INVESTIGATION OF GPS AND GIS FOR TRAVELER INFORMATION

WA-RD 332.1

Final Technical Report
March 1994

Washington State
Department of Transportation
Washington State Transportation Commission
Transit, Research, and Intermodal Planning (TRIP) Division
in cooperation with the U.S. Department of Transportation
Federal Highway Administration

87/9

64229

TECHNICAL REPORT STANDARD TI

1. REPORT NO. WA-RD 332.1	2. GOVERNMENT ACCESSION NO.
4. TITLE AND SUBTITLE INVESTIGATION OF GPS AND GIS FOR TRAVELER INFORMATION	5. REPORT DATE March 1994
	6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) Daniel J. Dailey and Po-Jung Lin	8. PERFORMING ORGANIZATION REPORT NO.
9. PERFORMING ORGANIZATION NAME AND ADDRESS Washington State Transportation Center (TRAC) University of Washington, JD-10 University District Building; 1107 NE 45th Street, Suite 535 Seattle, Washington 98105-4631	10. WORK UNIT NO.
	11. CONTRACT OR GRANT NO. Agreement T9233, Task 35
12. SPONSORING AGENCY NAME AND ADDRESS Washington State Department of Transportation Transportation Building, MS 7370 Olympia, Washington 98504-7370	13. TYPE OF REPORT AND PERIOD COVERED Final technical report
	14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

This study was conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. ABSTRACT

Traffic congestion is an increasing problem in many areas of Washington state. Efforts to control traffic flows and mitigate congestion must rely on the ability to accurately monitor the state of traffic flow on highways and arterials. This project, sponsored by WSDOT and executed at the University of Washington, combines two important aspects of geodesy, geographical information systems (GIS) and the global positioning system (GPS), to produce a traveler information system. This project produced a computer application that draws a graphical representation of the transportation data (a map) on an X-terminal. This project also produced a second computer application to perform real-time vehicle positioning using GPS. These two applications operate in a client/server distributed computing environment and share data over the Internet. The client/server paradigm is used to combine location and congestion information on one, digitally generated map display.

17. KEY WORDS GPS, GIS, traveler information, maximum likelihood, digital maps, distributed computing, client server	18. DISTRIBUTION STATEMENT No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22616

19. SECURITY CLASSIF. (of this report) None	20. SECURITY CLASSIF. (of this page) None	21. NO. OF PAGES 60	22. PRICE

Final Technical Report
Research Project T9233, Task 35
GPS and GIS for Traveler Information

INVESTIGATION OF GPS AND GIS FOR TRAVELER INFORMATION

by

Daniel J. Dailey
Assistant Professor
Principal Investigator

Po-Jung Lin
Research Assistant

Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

Washington State Transportation Center (TRAC)
University of Washington, JD-10
University District Building
1107 NE 45th Street, Suite 535
Seattle, Washington 98105-4631

Washington State Department of Transportation
Technical Monitor
Larry Senn, Urban Systems Engineer
Advanced Technology Branch

Prepared for

Washington State Transportation Commission
Department of Transportation
and in cooperation with
U.S. Department of Transportation
Federal Highway Administration

March 1994

Contents

List of Figures

Executive Summary

Traffic congestion is an increasing problem in many areas of Washington state. Efforts to control traffic flows and mitigate congestion must rely on the ability to accurately monitor the state of traffic flow on highways and arterials. This project, sponsored by WSDOT and executed at the University of Washington, combines two important aspects of geodesy, geographical information systems (GIS) and the global positioning system (GPS) to produce a traveler information system. This project produced a computer application that draws a graphical representation of the transportation data (a map) on an X-terminal. This project also produced a second computer application to perform real time vehicle positioning using GPS. These two applications operate in a client/server distributed computing environment and share data over the internet. The source code for these two applications are available to WSDOT personnel who may have an interest in this arena.

This report is presented in two major chapters corresponding to the two major theses asserted in the implementation of the *Investigation of GPS and GIS for Traveler Information* project.

The first thesis was that two advanced technologies, GIS and GPS, can be combined in a distributed computing environment to deliver traveler information. The first chapter described the architecture and implementation of such a traveler information system that combines GPS probe vehicle locations, traffic congestion information, and digital maps in a distributed computing environment. The components reported on include, (1) a GIS system we have called the IVHS map display client, (2) a GPS server system to provide the location of probe vehicles in real time, and (3) a traffic congestion server to provide a network interface to the WSDOT's inductance loop system located in the freeways surrounding metropolitan Seattle. This project successfully

implemented a demonstration system that combines the technologies mentioned. Further this project defined a general architecture that is extensible. The architecture and the implementation can easily include additional sensors or information sources and can be extended to a larger geographic area or even used in other geographic regions (other cities or states).

The second thesis of this project was that a maximum likelihood approach can be used to solve the GPS positioning problem. This is the topic of the second major chapter of this report. The second chapter presents a comprehensive methodology for positioning that uses the basic information, ephemeris and code phase, from the global positioning system satellites. It first developed a positioning algorithm that uses unbiased range estimates when four or more satellites are available. It then provided an overview of the bias correction inherent in the GPS positioning problem and enumerates an algorithm for positioning that includes corrections for the biases. Finally, it outlines a method for approximating the second order statistics of the errors in the position estimation. and provided a closed form for this approximation. The methods presented provide a uniform positioning solution that can be used with any GPS receiver.

The traffic engineers and maintenance engineers from WSDOT can potentially use this technology to perform such tasks as fleet management, vehicle tracking and digital mapping as well as for the traffic information application of the present project.

1. Introduction

Traffic congestion is an increasing problem in many areas of Washington state. Efforts to control traffic flows and mitigate congestion must rely on the ability to accurately monitor the state of traffic flow on highways and arterials. This project investigates the feasibility of combining advanced technologies in both vehicle location systems and digital geographic information systems (GIS) to produce a better tool for real-time traffic monitoring.

The status of traffic congestion in the Seattle metropolitan area can be understood in greater detail by combining probe vehicle position and congestion data on a digital map. This project designed, constructed and demonstrated a system that combines

- real-time location system data from the global positioning system (GPS) satellites;
- real-time onroad congestion, flow, and speed data;
- detailed regional digital maps, and
- high resolution graphics displays suitable to scale for wide distribution of this information

to provide an information system that is flexible enough to be valuable to several types of users. Previous work has combined flow and congestion data with a symbolic representation of the status of major highway systems.[1] Other projects have implemented (or plan to implement) an in-vehicle display of similar information.[2, 3] This effort focuses on providing a flexible information system that combines available data and is extensible to other possible data sources. While this project has no specific focus on in-vehicle graphical presentation, the system is designed to be capable of such an application with appropriate extensions.

The possible benefactors are: (1) commuters (pretrip or at traffic facilities) who can get accurate traffic information from probe vehicles, (2) traffic management personnel who might use the geolocation and display for maintenance fleet management or remote mapping, and (3) public transit operators who might also use it for fleet management or even dynamic route assignment.

The body of this work consists of two nearly independent theses. The first thesis is that several advanced technologies, such as GIS and GPS, can be combined in a distributed computing environment to deliver traveler information. The second thesis is that a maximum likelihood approach can be used to solve the GPS positioning problem. We present the work performed during this project in two major chapters and four appendices. Chapter 2 describes and documents the architecture used to construct the client and server systems that operate in a distributed computing environment to deliver real-time congestion and probe vehicle position information. Chapter 3, in conjunction with the appendices, provides a detailed prescription for positioning using the GPS satellite constellation.

2. System Architecture

The central issue for the portion of the project described in this chapter is obtaining traffic information from varied sources and presenting it in a meaningful way. This project uses geographically distributed computing architecture to implement a traffic information system that collects traffic data from several sources and presents it on a digital map.

The traveler information system developed in this project is divided into three subsystems: (1) the IVHS client digital map display system, (2) the GPS server system, and (3) the traffic congestion server system.

The first subsystem, the IVHS client digital map display system, operates in a UNIX X-window environment and is an example of a basic geographical information system (GIS). This subsystem produces a representation of map data on an X-terminal that may be located anywhere on the Internet. In this project we selected two publicly available map databases. The first is the U.S. Census Bureau data in the form of its TIGER files. The second source of digital maps is the U.S. Geological Survey's (USGS) digital line graph (DLG) files at the 1:100,000 scale. While this project is implemented using these two formats, it is designed to be independent of the map data source. This independence is a result of the fact that the system converts the location data from the source specific format (ASCII in both the TIGER and DLG cases) to a data structure developed in this project and then stores that data structure as a binary file. The display application reads the binary file structure. The format is converted for two reasons: (1) to improve the speed of accessing items that are part of the map data, and (2) to reduce the memory requirements placed on the display client when the system displays either maps of a large area (e.g., multicounty or statewide) or maps containing a high density of information (e.g., urban areas with all the streets, arterials, connectors, highways, waterways, and political boundaries).

The IVHS client display system reads the binary map file and displays the entire map on an X-terminal. It also establishes connections to information servers (the GPS information server and the traffic congestion information server are presently implemented). The user interface provides some simple GIS functions, such as

- display or hide dynamic information from servers,
- zoom,
- cut and save a portion of the map to a file,
- display or hide static features, and
- point to a feature and get information.

These basic GIS functions provide enough functionality to implement this demonstration system.

The second subsystem is the GPS server system. This subsystem provides location data (real time or recorded) from global positioning system (GPS) receivers located on vehicles or at preselected sites. The GPS receivers are either connected directly to the subsystem or placed on a vehicle, and a cellular phone and modem are used to retrieve the raw GPS measurements[1]. A GPS receiver obtains the data broadcast from the satellite constellation and measures the "code phase"[2] and current GPS time. The GPS server estimates the approximate locations of the receivers on the basis of these measurements. The present GPS server is capable of operating up to 32 receivers simultaneously and operates the GPS receivers in parallel. At initialization time, the server announces its well-known Internet socket name and awaits any IVHS client requesting a connection. After each connection has been confirmed by both the IVHS client and the GPS server, the GPS server begins sending the location of all operational GPS receivers to every connected IVHS client. The clients display a symbol

[1]See section 3 for the details.

[2]See section 3 for a description of the code phase.

that corresponds to the receiver location whenever new data are available. This is a real-time GPS positioning system that operates in a parallel processing mode within a distributed computing environment.

The third subsystem is the traffic congestion server system. This subsystem provides a network interface for the congestion information gathered by WSDOT's inductance loop system. This subsystem uses a modem to dial the WSDOT traffic management computer located at the Traffic System Management Center (TSMC) and updates the traffic congestion information (volume and occupancy) every minute. The methodology for communicating the traffic information to each IVHS map client is similar to that of the GPS server system. At the initialization time, the traffic congestion server announces its well-known Internet socket name and awaits requests from IVHS client applications. After each connection has been confirmed by both the traffic congestion server and IVHS client system, the traffic congestion server sends traffic information to the newly connected IVHS client system as well as those already connected.

The entire GIS/GPS system is interconnected over a network that uses an interprocess communication (IPC) paradigm to form the backbone of the system. As shown in Figure 2.1, there are two information servers in our present system, the GPS server system and the traffic congestion server system.

Each information server uses an interprocess communication channel to communicate with IVHS client systems. The information server uses an interprocess communication channel to announce its well-known Internet socket name. Any IVHS client that desires this type of information may send a connection request packet to the well-known Internet socket requesting a connection. When a server receives a request for data from a new IVHS display client, it first verifies the acceptability of establishing the connection (adjudication) and then opens a new IPC channel with the requester. The information about the new IPC channel is passed to the server process responsible for broadcasting the data. After both sides have agreed and confirmed this connection, the information server adds the new client to the list of clients and continues sending data records to all connected IVHS clients whenever new data are available. The newly

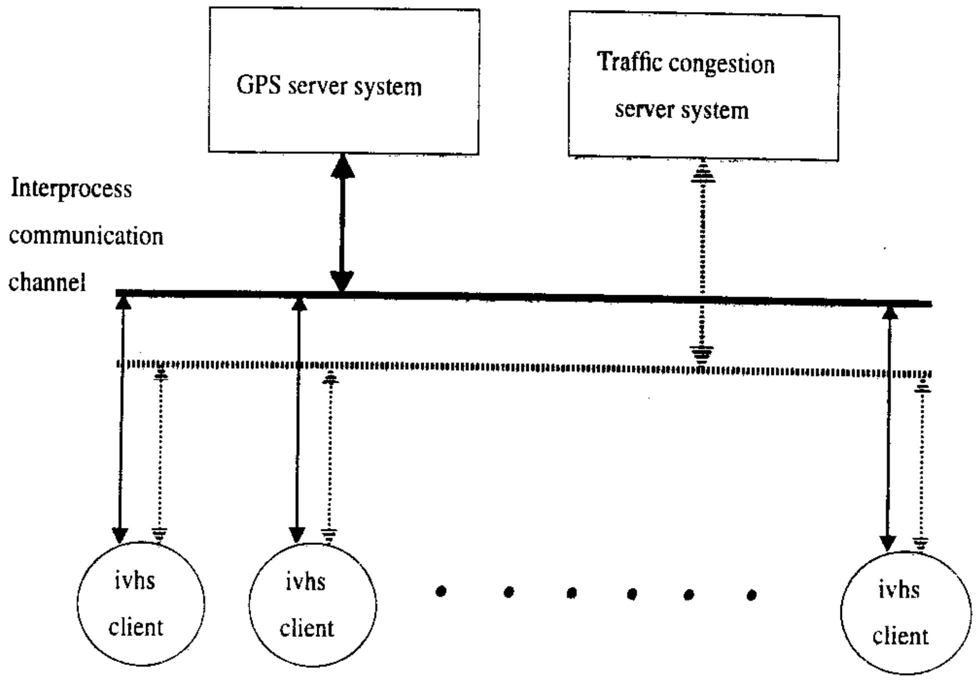
GPS server system
Traffic congestion server system
Interprocess communication channel
ivhs client
ivhs client
ivhs client

Figure 2.1: Client/Server Architecture.

added client receives the data from the information server on the newly established IPC channel. In the case of our GIS client, it displays the data (location or congestion) on a digital map. Figure 2.2 is a diagram of the communications structure used by the IVHS client and servers in which each of the processes is represented by an oval and the interprocess communication is represented by an arrow.

The combination of the three subsystems identified above, in conjunction with the backbone, comprise the body of the GIS/GPS project. Each of the subsystems is discussed in detail in the following sections.

2.1. IVHS Map Display Client System

The first major subsystem is the IVHS map display client. The following description of the IVHS map display client has several components, (1) the functionality built into the client, (2) the configuration of the client, and (3) a description of the source code

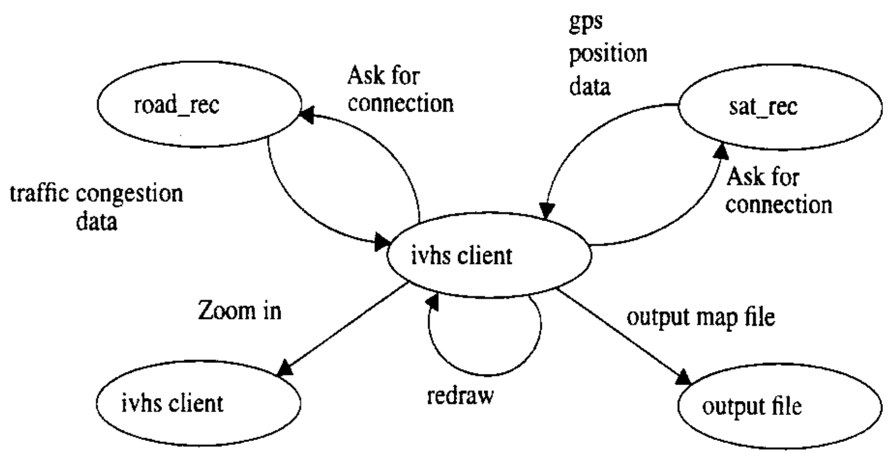

Figure 2.2: Communication state diagram for the client/server system.

components that make up the subsystems.

2.1.1. Functionality

The purpose of the IVHS map display client system is to pictorially represent the transportation data layer from the TIGER or USGS-DLG databases as a map on a computer display. The IVHS map display client reads a binary representation of the transportation layer data, taken from either TIGER or USGS-DLG data files, and draws a map on an X11 display. The transportation layer information includes highways, streets, arterials, connectors, trails, railroads, streams, and water and political boundaries. Representing these features as lines with end points and names requires a large amount of data for a metropolitan area. For example, the TIGER files for King County, Wash., contain 47 megabytes of data for the transportation information alone. Even the binary file format we are using requires 4 megabytes to represent King County.

Displaying the multiple megabytes of transportation layer data in a meaningful

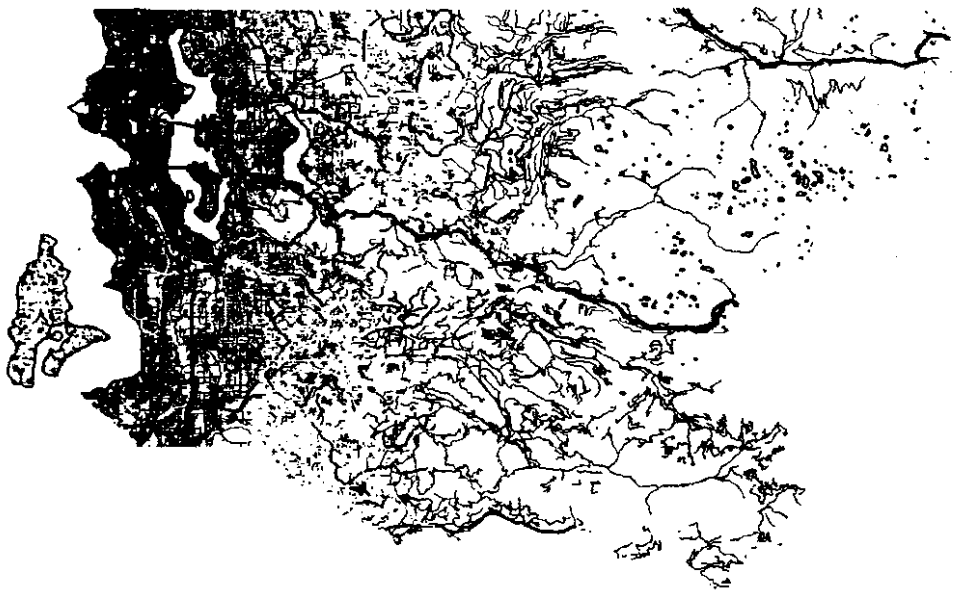

Figure 2.3: King County Map.

way is a difficult problem. Figure 2.3 is a map of the transportation data for King County. Because of the density of information, details of specific streets are lost, and only the gross features such as Lake Washington, Bainbridge and Mercer islands, I-90 east, the floating bridges, and perhaps parts of I-5 are discernible. The need to make the display more informative led us to build some basic GIS functionality into the IVHS map display client system.

The functions presently supported by the system and activated with a pop-up window in response to a mouse "left button down" are listed below.

Zoom in: Open a new window displaying the rectangular map area specified by dragging the cursor in the current window. Figure 2.4 is a submap created by zooming on a map of the density used for Figure 2.3. Pictorial representations of the region shown in Figure 2.4 have been used by previous traffic information systems.[1] This map includes I-5, I-405, and the connecting highways that contain the bulk of the WSDOT loop detectors used for congestion monitoring. Even this zoom level does not provide sufficient detail for an application that is capable of display-

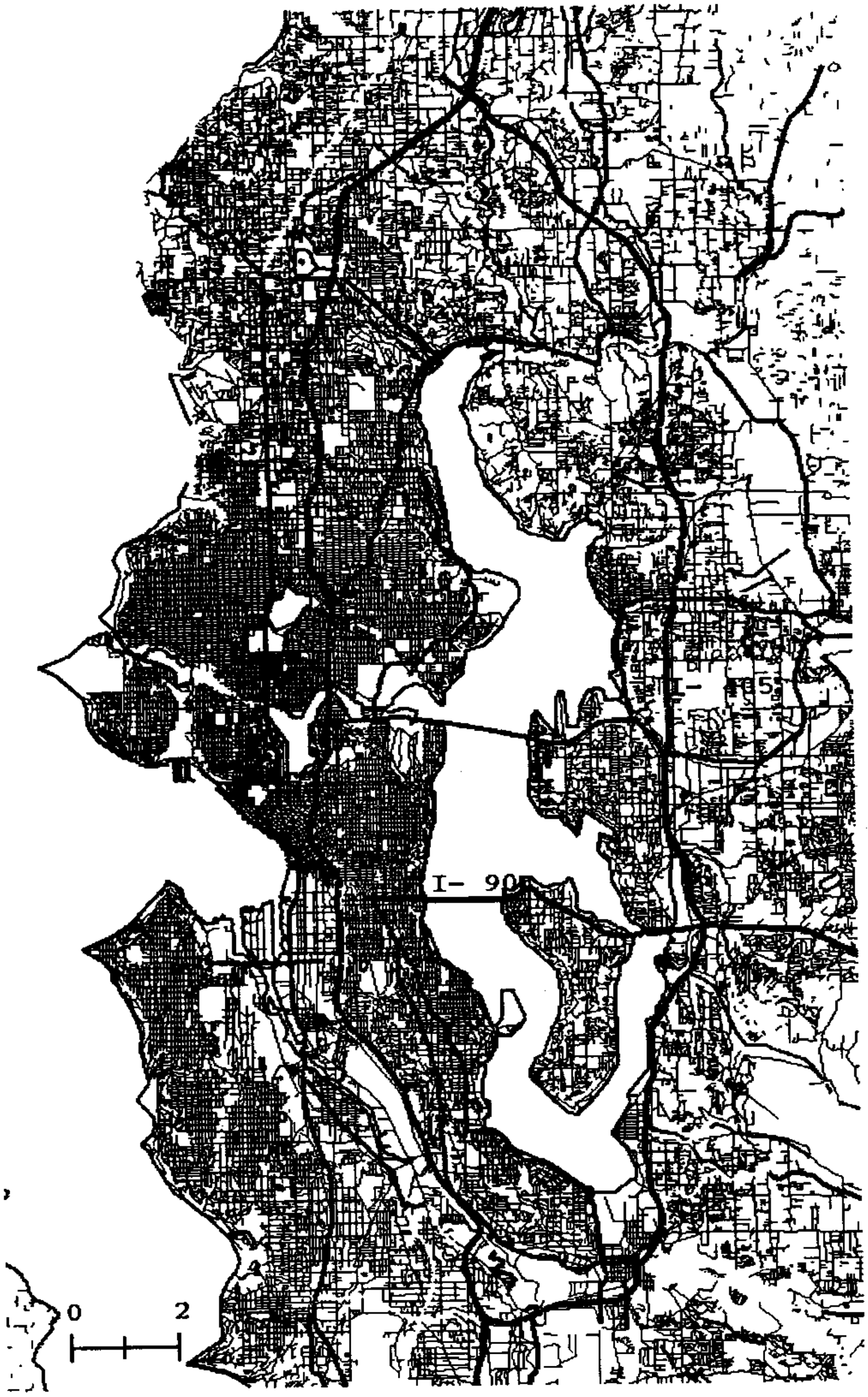

Figure 2.4: Seattle metropolitan area including I-5 and I-405.

ing individual vehicles. Because of the need for a variety of levels of detail for this project, we implemented a zoom function that operates over an arbitrary region rather than a standard set of magnifications, as is often used in transportation GIS applications.

Symbol on: Turn on the display of symbols associated with the GPS data. After setting the symbol on, the pop-up menu displays "Symbol off." Selecting "symbol off" causes all the symbols displayed on the screen to be removed. It also sets the pop-up menu option back to "Symbol on."

Output file: This option creates a file that contains the map data from the current window. In response to selecting this option a dialog menu appears in which the user first specifies the output filename and then selects the output format for the file (either ASCII or binary). The output file created contains the map data from the current window in the format the user has chosen. This is useful for creating custom maps of specific areas if the total geographic area of interest is known in advance.

Display set: The contents of the map can be set by selecting this option. It displays a control menu on which the user may select the components of the map data to display on the screen. The available components include freeways, highways, local streets, trails, railroads, and political and water boundaries. Each of these attributes can be individually set to be displayed or hidden. Figure 2.5 is metropolitan Seattle highways and water boundaries; it is the same map as in Figure 2.4 with the local street information hidden.

Redraw: Redraw the current display window.

Sat on: Set the satellite data receiving mode on. This invokes a process to communicate with the GPS server through an interprocess communication channel. The function also changes the selection in the pop up menu to "Sat off." When "Sat off" is selected on the pop-up menu, it sends a disconnection request to the GPS server, and the menu selection is reset to "Sat on."

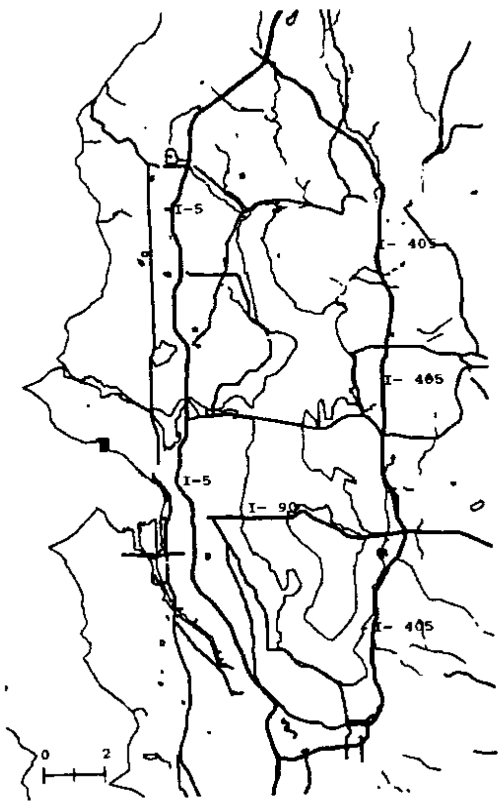

Figure 2.5: Seattle metropolitan area excluding local streets.

Road on: This selection turns on the traffic congestion data option. The IVHS client system invokes a process to communicate with the traffic congestion server using an interprocess communication channel. It also changes the pop-up menu selection to "Road off." When "Road off" is selected on the menu, it sends a disconnection request packet to the traffic congestion server. The menu selection is then reset to "Road on."

Close: This selection closes the current display window. If there is a parent window, the focus returns to the parent window. Otherwise, closing the window exits the IVHS client display system.

Quit: This selection causes the IVHS client display system to clean up and exit.

The functionality in the IVHS map display client system is, while in its infancy, sufficient to provide a testbed and demonstration of the utility of combining location data, congestion data, and digital maps on a communications backbone.

2.1.2. Configuration

The IVHS client display system is a geographical map display system that operates in an X-window environment. The configuration information is obtained from two sources, the command line used when the application is invoked, and the X-server database. On the command line the file name for the binary map data file and the optional shared memory address are input. The invocation command is

```
ivhs [-s SHARED-MEMORY-ID] [-fi] binary-file-name.
```

In addition, each map element's color is specified in a single resource file named "`.ivhsrc`". The X11 system command

```
xrdb -merge .ivhsrc
```

issued before the **`ivhs`** program is invoked causes the customized configuration information from the **`.ivhs`** file to be used in the map displayed by the **`ivhs`** program. The user interface is mouse based; moving the pointer and clicking the mouse buttons initiate various operations on the map display. This subsystem reads a binary map file and stores the map information in shared memory segments. The **`ivhs`** program displays map features such as highways, arterials, surface streets, political and water boundaries. and railroads on an X-terminal screen. Colors are assigned in the resource file **`.ivhsrc`**. Each type of feature is assigned to a specific class, and each class is stored in a different memory segment. The shared memory address option allows multiple clients, operating on the same computer, to share the memory resident binary map data. This feature enhances the speed of the client initialization, as the map data (often tens of megabytes) does not need to be loaded from disk. In addition, the shared memory implementation reduces the total memory required for multiple clients to operate simultaneously.

2.1.3. Source Code File Description

This section describes each of the source files (written in the C language) that, along with the Makefile that controls the compilation and linking of the applications, make up the software developed for the IVHS map display client.

main.c: This is the central routine for the IVHS map client display system. This routine handles the X-window resource allocation, such as segment color, display window height, display window width, foreground color, background color, and window color depth. It also provides an event handler to respond to events such as pointing or clicking with the mouse. The default resource values are assigned in this routine; however, the defaults are overridden by the "`xrdb`" command.

disp.c: This file contains subroutines to select what is displayed on the current window.

draw.c: This file contains subroutines to process events, such as Expose, Configure/Notify, and MapNotify. They include routines to draw or redraw when needed.

out.c: This file contains subroutines to create either ASCII or binary files from the current map window.

rdbf.c: This routine reads binary map data from a file to memory and arranges the memory into a shared memory segment. Several IVHS client systems can simultaneously read the shared memory segments, which minimizes the memory requirements for multiple clients.

real.c: This routine opens a new sub-map window using the contents of the current map window. It is invoked by selecting the `Zoom in` option.

rubber.c: This file includes subroutines to select a sub-map from an existing map window and create a new map based on the subarea selected. It uses rubber lines when selecting a subarea from the map window.

sat.c: This file includes several subroutines that handle the communication with the GPS server system. It also includes programs to display symbols on the map using location data received from the GPS server system.

symbol.c: This file contains subroutines to set displayed symbols ON (or OFF) in the displayed window. It includes a routine to draw these symbols, to redraw these symbols when needed, and to get the symbol's information file.

2.2. GPS Server System

The second major subsystem developed in this work is the GPS server system. The purpose of GPS server system is to communicate with GPS receivers, process the satellite data obtained from the receivers, calculate the receivers correct location, and send the location information to all of the connected IVHS client display systems. As shown in Figure 2.6, each receiver has a unique process that calculates the physical location of using the GPS satellite measurement data.[3] Each position calculation process has a dedicated Kalman filter process to minimize the errors in the positioning solution. The location tuple [latitude, longitude, height] and GPS time are sent to the network service application. The network service application distributes the GPS information to connected IVHS clients and also deals with the addition and removal of clients. The GPS server system uses a pipeline architecture; that is to say, all three parts of the application are running in parallel. This parallelism allows us to perform the GPS positioning in real time. In this case, real time means the ability to produce a position estimate for each GPS receiver at an interval of approximately one second.

This subsystem is divided into three parts; each part implements different functions. The parts are (1) GPS receiver position calculation, (2) Kalman filter, and (3) network service. The following subsections describe each of these parts' functionality.

[3]See section 3 for the details of the position calculation.

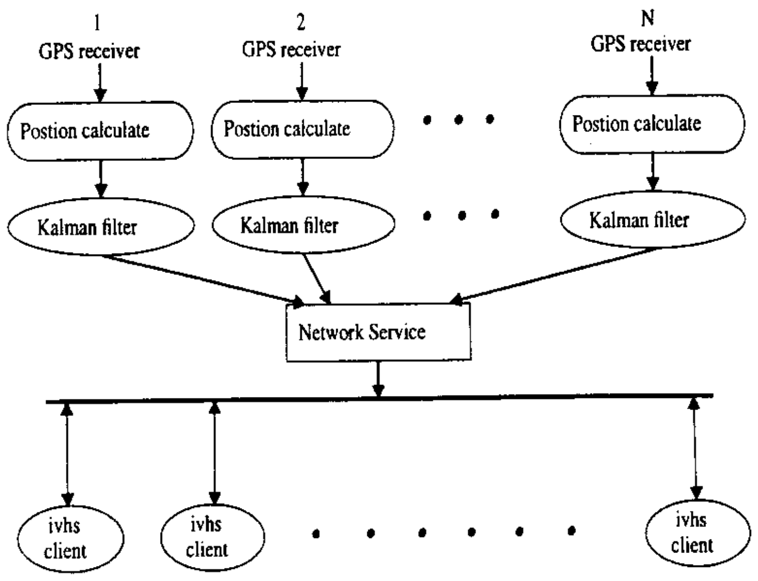

Figure 2.6: Architecture for combining GPS location data and displaying it on the IVHS client display system.

2.2.1. GPS receiver position calculation

This part of the subsystem comprises two types of GPS information servers, (1) a relative correction server and (2) a vehicle location server. The location solution in both cases uses the raw GPS measurements of code phase and GPS time. This position solution provides better accuracy than the vendor supplied software onboard the GPS receiver. It also provides knowledge of the statistics associated with the solution that allow the results to be used for later processing.

The correction server consists of a set of GPS receivers placed at known sites, a data link to the computer on which the server software runs, and a network connection to share the correction factor with the location servers. Pseudo range data calculated from the GPS code phase are used with the satellite ephemeris[4] information broadcast by the satellites and the known location of the receivers to generate per-satellite cor-

[4]Ephemeris information is a set of values used to calculate the orbit parameters of a satellite. This is necessary to calculate the absolute position of a satellite at a given time.

rection factors to remove common biases such as the selective availability (SA) biases introduced by the U.S. Department of Defense. These corrections are available to other applications doing real-time positioning.

The vehicle location server provides the location of a particular vehicle in real time to any application on the network that requests that information. Each vehicle location server is in communication (via cellular modem) with a GPS-equipped vehicle as well as the correction server (via network). The vehicle location process requires the vehicle to report only the tuple [satellite vehicle number, code phase, GPS time] for each satellite that is visible. The location server uses the code phase from the available satellites with the ephemeris information and correction factors to perform a maximum likelihood estimate of the point position.

2.2.2. Kalman filter

The point estimate of position is used with a dynamics model implemented as a Kalman filter to estimate the state variable [position, speed, acceleration]. The point estimate of position has a standard deviation of just under 3 meters, and the Kalman filtering reduces this error. The Kalman filter receives the estimated position as input and uses maximum likelihood estimation to minimize the errors in the position estimate. The output position is guaranteed to be a minimum variance estimate of the state variables. This position information is available to any other application on approximately a 1-second interval.

2.2.3. Network Service

The purpose of network service in the GPS server subsystem is to handle the connection requests from IVHS clients and to send the position tuple [latitude, longitude, height] and GPS time to the connected IVHS client system. The process labeled "`sat_serv`" in Figure 2.7 receives the position data from all of the operating Kalman filter processes and sends the combined single data stream to the process "`sat_rec`." Process "`sat_rec`" listens to the interprocess communication channel associated with the well-

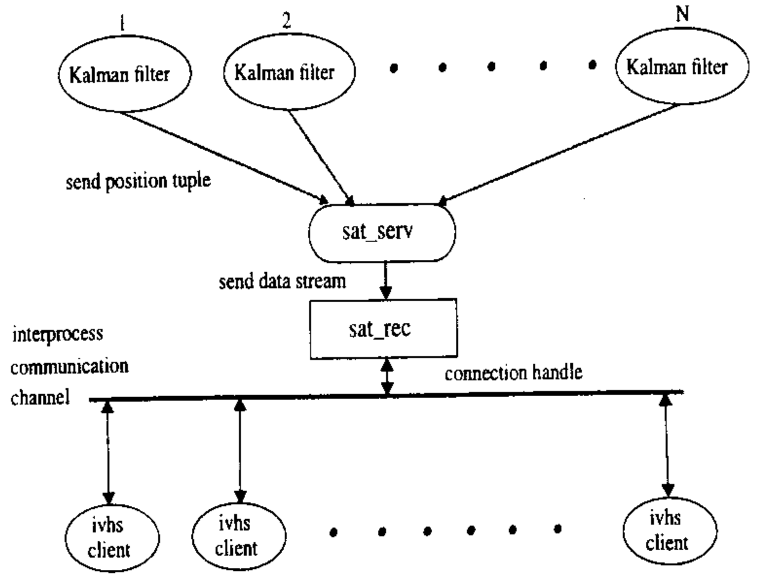

Figure 2.7: Processes involved in providing network access to the GPS position data.

known name of the GPS server and services any IVHS clients that request position data. In addition, the process "sat_rec" listens to the interprocess socket connected to the process "sat_serv." If position data are received, they are immediately redistributed to all connected IVHS client systems each of which returns an acknowledgment.

The GPS server system is constructed from a group of processes interconnected by interprocess communication channels.

2.2.4. Example

Figure 2.8 is composed of four snapshots of a small area of Seattle near 45th Street N.E. and I-5 on which is superimposed the location of a GPS equipped vehicle. The vehicle is moving east (from left to right), and the chronology begins at the top.

2.2.5. Source File Description

This section describes each of the source files (written in the C language) that, along with the Makefile that controls the compilation and linking of the applications, make

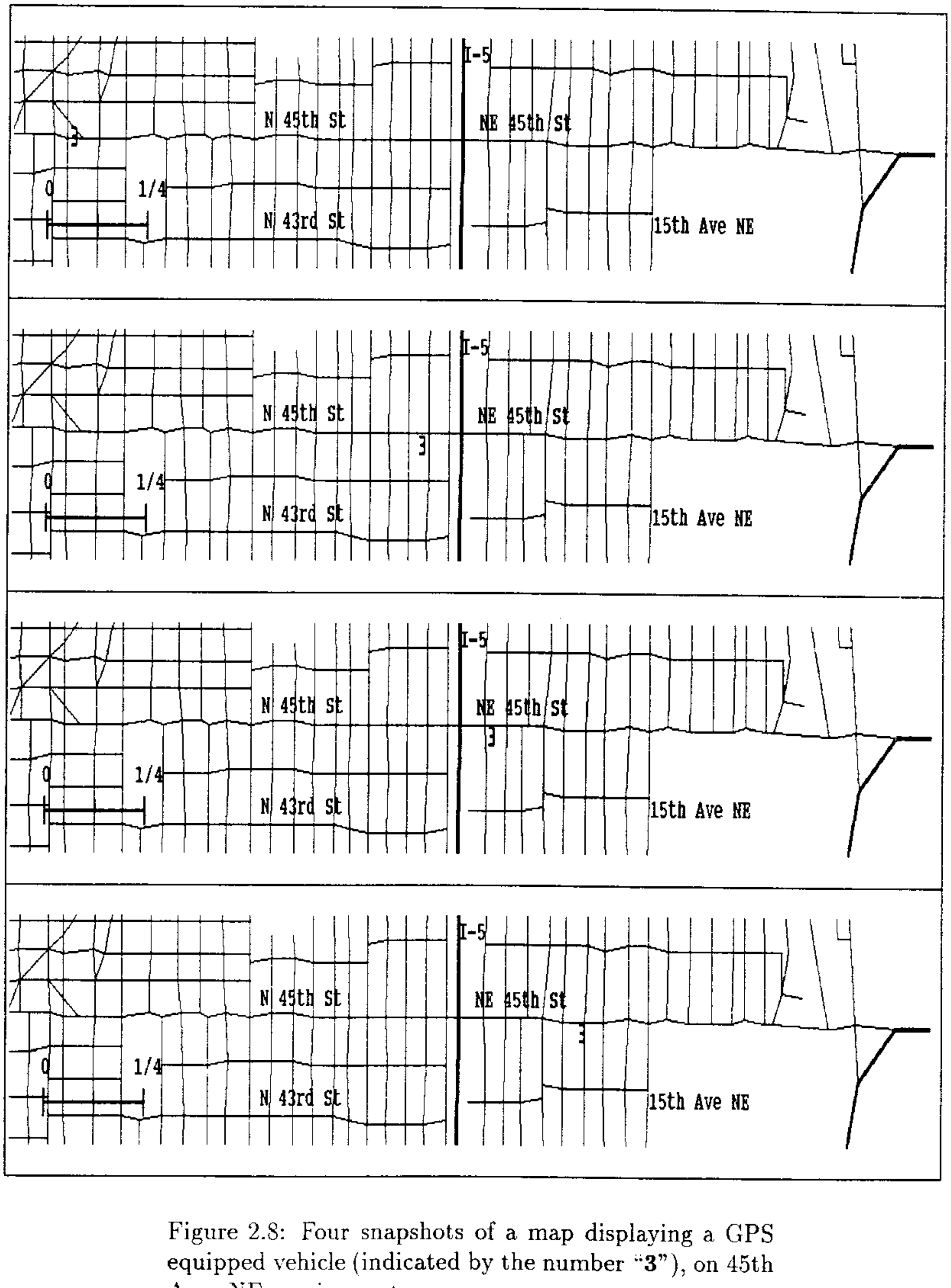

Figure 2.8: Four snapshots of a map displaying a GPS equipped vehicle (indicated by the number "**3**"), on 45th Ave. NE moving east.

up the software developed for the IVHS GPS server system.

sat_rec.c: Subroutine to handle the data stream from `sat_serv` and service its well-known Internet socket name.

sat_serv.c: Subroutine to receive data from various Kalman filter outputs and combine them into a single data stream that is sent to `sat_rec`.

sat_echo.c: Subroutine to handle the Internet socket connection between the Kalman filter processes and the `sat_serv` process.

net_echo.c: Subroutine to handle the Internet connection between `sat_rec` and the IVHS client display system.

gps_refer.c: Subroutine to get the correction factor from receivers at known sites.

gps_cli.c: Subroutines to get the approximate locations of the GPS receivers.

gps_cli_file.c: Subroutines to get the approximate locations of the GPS receiver from the pre-recorded data file.

gps_serv.c: Routine that reads the file `reftbl` and initializes as many child processes as there are records in the file `reftbl`. Each process operates a GPS reference receiver. The subroutine reads the file `gpstbl` and initializes as many child processes as there are records in the file `gpstbl`. These processes service the remote GPS receivers. This routine also determines how a GPS receiver is connected, directly or by cellular phone modem.

gps_read.c: Routine that handles all the read and write commands used by the GPS receiver.

gps_check.c: Subroutine that verifies that the current satellites tracked by a receiver and those tracked by the reference receivers are the same. If the same satellites are available, the subroutine corrects the common biases in the pseudo ranges using the data supplied by the GPS reference server.

gps_ref_locate.c: Subroutine to calculate the approximate location of the reference receiver on the basis of the satellite data received by the reference receiver.

gps_cli_locate.c: Subroutine to calculate the approximate location of a GPS receiver on the basis of the satellite data received by the the GPS receiver after the common biases have been corrected.

modem.c: All the subroutines necessary to handle the modem dialing and connection.

kalman_locate.c: Routine that uses data from `gps_cli_locate` and computes an improved location estimate with the Kalman filtering algorithm.

xyz_ephem.c: Routine to calculate the distance between the GPS receiver and the tracked satellite on the basis of the satellite's ephemeris data.

gps_locate.c: Routine to estimate the 3-D position of a GPS receiver using the distances passed from `xyz_ephem`.

fit.c: Routine that implements a maximum likelihood method to estimate the 3-D positions of the GPS receiver using the distances between the receiver and tracked satellites.

implicit.c: Routine that calculates the gradient and Hessian of the objective function for use in the routine **fit.c**

geoxyz.c: Routine to transform the location tuple [latitude, longitude, height] into a 3-D position tuple [X, Y, Z].

xyzgeo.c: Routine to transform the 3-D position tuple [X, Y, Z] into a location tuple [latitude, longitude, height].

tools.c: File that includes miscellaneous subroutines that support the routines mentioned above.

matrix.c: Subroutines to support matrix calculations, such as matrix adds, matrix subtract, matrix multiplies, matrix divides, inverse matrix.

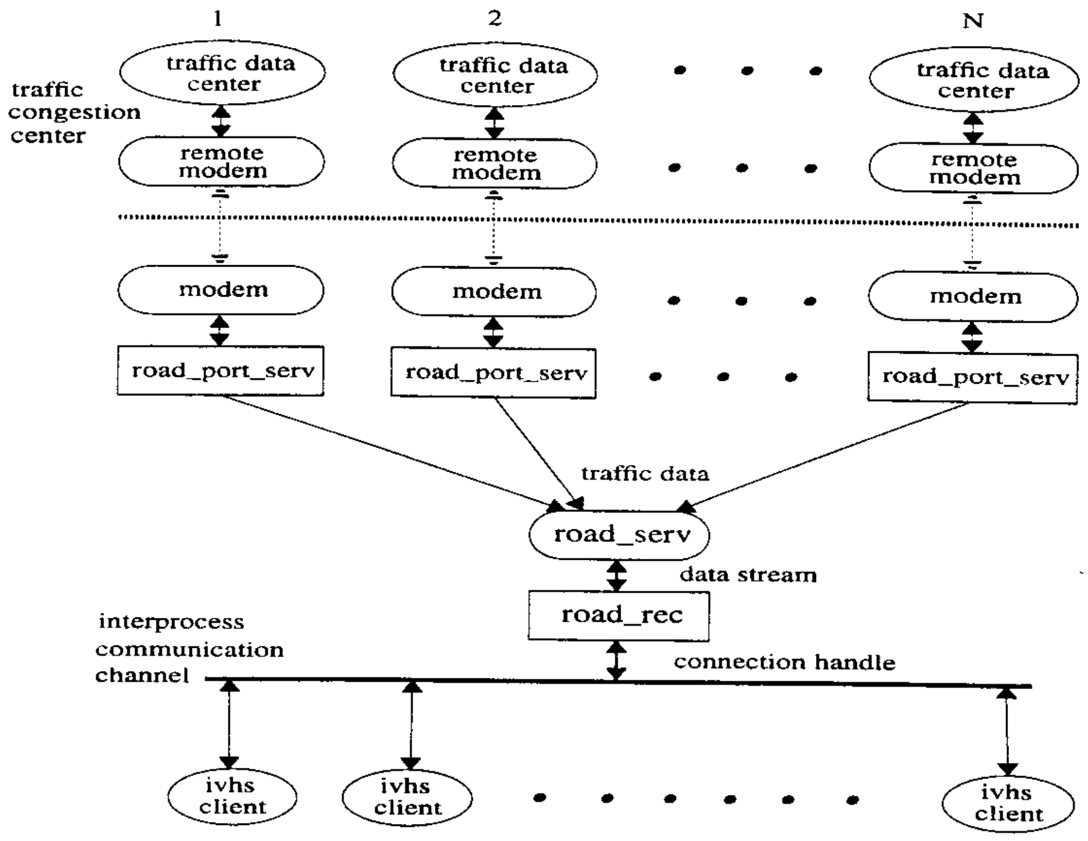

Figure 2.9: Congestion Server Architecture.

In addition, the routines **dassum.f, dzxpy.f, dcopy.f, ddot.f, dgeco.f, dgedi.f, dgefa.f, dgesl.f, dnrm2.f, dscal.f, dswap.f, idmax.f** from LINPACK [4] are used to perform matrix operations.

2.3. Traffic Congestion Server System

Presently, WSDOT has an extensive loop and camera system for traffic management. The loop system measures occupancy and volume data to quantify the traffic congestion level on highways. The Traffic System Management Center (TSMC) collects the loop data from the loop system. The TSMC computer combines the data from the individual loops into one data stream that has as many records as there are loop stations. These data are available at 1-minute intervals.

The purpose of the traffic congestion server system is to make the loop data available to the IVHS clients on the backbone. As shown in Figure 2.9, a modem connection is established to a remote modem located at the TSMC. Our system requests and receives the loop data from all the stations. It then extracts the valid data records from those received. These traffic congestion data are sent to the process "`road_serv`." The process "`road_serv`" collects all valid loop data records from any available traffic congestion centers and combines them into a single data stream that will be sent to the process "`road_rec`." The process "`road_rec`" handles the IVHS client interprocess communication connect requests in a manor analogous to the method used by the process "`sat_rec`" in the GPS server system. Finally, the process "`road_serv`" sends the traffic volume/occupancy data to the IVHS clients connected to the traffic congestion server system as the data become available.

Below is a description of each of the source files (written in the C language) that along with the Makefile that controls the compilation and linking of the applications, comprise the software developed for the IVHS congestion server system.

road_rec.c: Subroutine to handle the data stream from `road_serv` and service its well-known internet socket.

road_serv.c: Subroutine to receive traffic volume/occupancy data from every `road_port_read` process and combine these data records into a single data stream to be sent to `road_port_serv` via the Internet socket.

road_echo.c: Subroutine to handle the Internet socket between `road_port_serv` and `road_serv`.

net_echo.c: Subroutine to handle the Internet connection between `sat_rec` and the IVHS client display system.

road_port_serv.c: Routine that checks the file `roadtbl` to determine which devices are connected to modems and how many traffic congestion centers need to be connected. The routine forks as many child processes as there are records in file `roadtbl`. Each of these processes is dedicated to obtaining traffic congestion information from a traffic management center over a phone line with a modem.

road_port_cli.c: Routine that obtains processed traffic data from the process `road_port_read` and sends it to process `road_port_serv`.

road_port_read.c: Subroutines to (1) connect to the remote modem at the Traffic Management Center, (2) get the traffic data, and (3) extract the valid data records. These data are then sent to the process `road_port_cli` via the Internet communication socket.

modem.c: File that includes all the subroutines necessary to handle modem dialing and connection maintenance.

2.4. Architecture Summary

This chapter presented a traffic information system that operates in a distributed computing environment. This system includes a GIS client to display digital map data and two information servers. The two information servers are the GPS server that provides real-time probe vehicle locations and the traffic congestion server that provides real-time congestion information from traffic management centers. The client and servers are implemented as processes that execute on Unix[5] workstations and communicate using an interprocess communications channel. The interprocess communication channel uses the Internet as the network transport medium. The reliance on Internet style interprocess communication enables the processes that implement the client and servers to operate without regard to geographic location. For example, the location server operates on one computer at the University of Washington, the GPS server operates on another, the GIS clients operate on yet other computers, and the map can be displayed on any X-terminal anywhere on the international Internet. The combination of the client/server paradigm and the interprocess communication paradigm allows the

[5] Unix was chosen for use based on its multitasking support, support for interprocess communication over a network and portability of any software developed.

traffic information system to operate without regard to physical location and to scale to regional size.

3. GPS Positioning Prescription

The proliferation of global positioning system (GPS) applications in the scientific and engineering community has made a complete prescription for calculating positioning useful. We were unable to find such a prescription and, in the course of our work, we have developed such.[1] In this chapter we present a method for the calculation of absolute and relative position using the C/A code phase measurements. The method can use GPS receivers from several vendors to obtain the position solution in a uniform way. It also provides the statistics of the position estimates. Many applications use position estimates as an input. The statistics of these estimates are necessary for applications that do estimation and control processing based on the position estimates.

We present the method in five sections and four appendices. The first section provides some background on GPS positioning. The second section describes the use of the Gauss-Newton method to solve for position given unbiased satellite ranges. The third section describes biases in the measurement of the ranges to the satellites. The fourth section provides a step by step algorithm for position estimation. The fifth section discusses the second order statistics of the position estimate. Finally, the appendices provide the details necessary to implement the method.

3.1. Background

The global positioning system (GPS) consists of two components: first a set of satellites orbiting 20,200 km above the earth that transmit signals which contain positioning information, and second a receiver that can decode and display the information. The basic idea used in positioning is that given N ranges from known points, a position in N space can be calculated. In GPS the known points are the location of a set of

[1] The contents of this chapter are taken largely from [5].

satellites. However, since the satellites are moving at a rate of approximately 3,500 meters per second[2] the position of the satellites must be calculated at an instant in time.

Calculating the positions of the GPS satellites requires ephemeris information about the satellite orbits. This information is included in the signal transmitted by the satellite. The GPS navigation data is transmitted in 1500 bit frames at a rate of 50 bits per second.[3] Each satellite broadcasts this message on two carriers called L1 (1575.42 MHz) and L2 (1227.60 MHz).[4] Using the ephemeris information an accurate position for each of the satellites, as a function of time, can be calculated. The range to each of the satellites is the remaining information necessary to calculate the receiver position.

The range to each satellite is established using a signal transmitted by the satellites, highly accurate clocks and the speed of light. Each satellite transmits a known unique pseudo random sequence (the C/A code) of 1023 bits. This sequence repeats every millisecond. The resulting wave length (Λ) of the C/A code is approximately 300 kilometers. The receiver measures the time delay necessary to align the copy of the code generated in the receiver with the code received from the satellite. We define delay for the *ith* satellite times the speed of light as the code phase (C_i). The code phase, when corrected for the known deviation of the satellite clock from GPS time is denoted by $\hat{C}_i$. This is the distance that separates the receiver from that satellite modulo the wavelength (Λ) of the C/A code. The code phase and the number of wavelengths (n_i) between the satellite and the receiver define the pseudo range to the *ith* satellite,

$$r_i = \hat{C}_i + n_i \Lambda. \tag{3.1}$$

Pseudo range estimates from three satellites would be sufficient to estimate position if the satellite clocks and the receiver clocks were perfectly aligned. The satellite clocks can be aligned in GPS time using the ephemeris information. However, the accuracy of the clock used in commercial receivers is inadequate for this alignment. This adds

[2]Based on a twelve hour orbital period.
[3]For the details of this encoding see section 20.3 of reference [6] and section 4.1 of reference [7].
[4]See reference [6] section 3.3.1.1 or section 6.7 of [7]

a bias (t_b) to the range estimate that is associated with the alignment of the receiver clock with GPS time. The approximation that relates pseudo range to receiver position (**R**) is,

$$\|\mathbf{R} - \mathbf{S}_i\| \approx r_i - t_b, \tag{3.2}$$

where $\mathbf{S}_i$ is the position of the *ith* satellite. The bias term is equal for all the satellites and thus only one additional piece of information is required to generate a position solution (e.g. four satellite ranges are sufficient to determine a position solution). The next section provides a method for estimating the receiver position using four[5] or more[6] satellites.

3.2. Estimating Position using Unbiased Pseudo Ranges

This section presents the theory and methodology for calculating position based on unbiased pseudo ranges. The position solution is obtained by optimizing an objective function equivalent to a likelihood function. We use a Gauss-Newton method to solve the optimization problem. The presentation in this section accounts for the error in the receiver clock but does not address other time biases.

The likelihood is a function of the pseudo ranges and receiver position. Given a set of pseudo ranges our estimate chooses the position that maximizes the likelihood function. The error in the pseudo range to the *ith* satellite is written

$$\varepsilon_i = \| \mathbf{R} - \mathbf{S}_i \| - r_i + t_b \tag{3.3}$$

where,

$\mathbf{R}$	is the position of the receiver in earth centered, earth fixed XYZ coordinates (e.g. $[R_x, R_y, R_z]^T$),
$\mathbf{S}_i$	is the XYZ position of the *ith* satellite,
$\|\cdot\|$	is the Euclidean distance,
r_i	is the pseudo range to the *ith* satellite, and

[5]the fully determined problem
[6]the over determined problem

t_b is the timing bias (which is defined to be the error in the receiver clock times the speed of light).

We assume that the error ε_i is normally distributed, with zero mean and standard deviation σ_i, to obtain an expression for the probability density as a function of ε_i,

$$\frac{1}{\sigma_i\sqrt{2\pi}}\exp\left\{-\frac{1}{2}\frac{\varepsilon_i^2}{\sigma_i^2}\right\}. \tag{3.4}$$

In addition if we assume that the errors in the range to each of the satellites are independent but have the same variance so that the likelihood function for positioning with a set of N satellites is,

$$\prod_{i=1}^{N}\frac{1}{\sigma\sqrt{2\pi}}\exp\left\{-\frac{(\| \mathbf{R}-\mathbf{S}_i \| - r_i + t_b)^2}{2\sigma^2}\right\}. \tag{3.5}$$

Maximizing this likelihood is equivalent to minimizing the the negative of its log. This is equivalent to minimizing,

$$f(\mathbf{R}, t_b, \mathbf{r}) = \frac{1}{2}\sum_{i=1}^{N}(\| \mathbf{R}-\mathbf{S}_i \| - r_i + t_b)^2, \tag{3.6}$$

where $\mathbf{r} = [r_1, r_2 \ldots r_N]^T$ is the vector of observed pseudo ranges. The positioning problem can be expressed as:

$$\text{Minimize } f(\mathbf{R}, t_b, \mathbf{r}) \text{ with respect to } \mathbf{R} \text{ and } t_b. \tag{3.7}$$

To simplify the notation define,

$$\mathbf{x} = \begin{bmatrix} \mathbf{R} \\ t_b \end{bmatrix}, \quad q_i(\mathbf{x}, r_i) = \| \mathbf{R}-\mathbf{S}_i \| - r_i + t_b, \tag{3.8}$$

$$f(\mathbf{x}, \mathbf{r}) = \frac{1}{2}\sum_{i=1}^{N} q_i(\mathbf{x}, r_i)^2 = \frac{1}{2}\mathbf{q}^T\mathbf{q}, \tag{3.9}$$

where $\mathbf{q} = [q_1(\mathbf{x}, r_1), q_2(\mathbf{x}, r_2), \ldots q_N(\mathbf{x}, r_N)]^T$. The minimization is performed using the Gauss-Newton method to find a zero in the gradient of the objective function. The Gauss-Newton formulation uses values for the gradient (the first derivative) of the objective function and an approximation of the Hessian (the second derivative) of the objective function. The gradient [7] of the objective function with respect to the parameter vector $\mathbf{x}$ is,

$$\mathbf{g}(\mathbf{x},\mathbf{r}) = \nabla_x f(\mathbf{x},\mathbf{r}) = \begin{bmatrix} \frac{\partial f}{\partial R_x} \\ \frac{\partial f}{\partial R_y} \\ \frac{\partial f}{\partial R_z} \\ \frac{\partial f}{\partial t_b} \end{bmatrix} \tag{3.10}$$

Using equations (3.9) and (3.10), with the definitions [8]

$$\mathbf{Q}_i(\mathbf{x}) = \nabla_x q_i(\mathbf{x},\mathbf{r}), \qquad \mathbf{Q}(\mathbf{x}) = [\mathbf{Q}_1, \mathbf{Q}_2 \dots \mathbf{Q}_N], \tag{3.11}$$

the gradient of the objective function can be written,

$$\mathbf{g}(\mathbf{x},\mathbf{r}) = \mathbf{Q}(\mathbf{x})\mathbf{q}(\mathbf{x},\mathbf{r}). \tag{3.12}$$

The Hessian of the objective function is,

$$\nabla^2_{xx} f(\mathbf{x},\mathbf{r}) = \mathbf{Q}(\mathbf{x})\mathbf{Q}^T(\mathbf{x}) + \sum_{i=1}^{N} q_i(\mathbf{x},r_i)\nabla^2_{xx} q_i(\mathbf{x},r_i) \tag{3.13}$$

where the (l,m) element of the matrix $\nabla^2_{xx} q_i(\mathbf{x},r_i)$ is $(\frac{\partial^2 q_i}{\partial x_l \partial x_m})$. The approximation of the Hessian used in the Gauss-Newton method is[9],

$$\nabla^2_{xx} f(\mathbf{x},\mathbf{r}) \approx \mathbf{G}(\mathbf{x}) = \mathbf{Q}(\mathbf{x})\mathbf{Q}^T(\mathbf{x}) \tag{3.14}$$

Using equations (3.12) and (3.14) the functions $\mathbf{g}(\mathbf{x},\mathbf{r})$ and $\mathbf{G}(\mathbf{x})$ can be calculated using only $\mathbf{q}(\mathbf{x},\mathbf{r})$ and its gradient $\mathbf{Q}(\mathbf{x})$. Thus no explicit second derivative formulas are necessary. The analytical expressions for the calculation of the elements of $\mathbf{q}(\mathbf{x},\mathbf{r})$ and $\mathbf{Q}(\mathbf{x})$ are given in equations (A.1) and (A.2) of appendix A.

[7]Note that the gradient of a scalar function is a column vector.
[8]See appendix A for the details of calculating $\nabla_x q_i(\mathbf{x},\mathbf{r})$ which is independent of $\mathbf{r}$.
[9]See equation (6.1.8) of [8].

Given a range vector $\mathbf{r}$. the first order necessary condition for a solution $\mathbf{x}_0$, is that $\mathbf{g}(\mathbf{x}_0, \mathbf{r}) = 0$. We use the Gauss-Newton method to solve for $\mathbf{x}_0$ in this problem. The Gauss-Newton iterate for the $(k+1)$ iteration, given $\mathbf{x}^{(k)}$, is[10]

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{G}(\mathbf{x}^{(k)})^{-1}\mathbf{g}(\mathbf{x}^{(k)}, \mathbf{r}). \tag{3.15}$$

This iteration is initialized with an approximate solution to problem (3.7). The iterative scheme is continued until

$$\| \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \| < \delta \tag{3.16}$$

where δ is the desired accuracy of the solution (we use 0.1 meters as our criteria). This methodology provides a mechanism for positioning with unbiased range data. However, the observed pseudo range data contains biases that can be removed. In the next section we describe the necessary pseudo range corrections.

3.3. Removable Biases

In this section we discuss several biases in the pseudo range measurements that need to be removed before calculating the position solution presented in section 3.2.

The first problem is determining the number of wavelengths (Λ) between the satellites and the receiver. The raw data consists of the C/A code phase $(\hat{C}_i)$ whose wavelength (Λ) is approximately 300 km. From equation (3.1) the raw pseudo range (r_i) to the ith satellite is related to the code phase by

$$r_i = \hat{C}_i + n_i\Lambda, \tag{3.17}$$

where (n_i) is an integer. Since the receiver can only measure the code phase, this pseudo range measurement has an ambiguity associated with the determination of n_i.

A second correction concerns the coordinate system in use and the time it takes the signal to travel from the ith satellite to the receiver (t_i). While the C/A signal is propagating from the satellite to the receiver the earth is spinning and the satellite is moving. This is important because the coordinate system used in the solution rotates

[10]See equation (6.1.9) of [8].

with the earth. The Z axis of this coordinate system passes through the poles, and the X and Y axis spin with the rotation of the earth. Since this coordinate system is moving a fixed time must be chosen to perform the position solution. The position solution is done in the coordinate system corresponding to the time of reception. When the signal from the satellite is received the coordinate system has rotated $t_i\dot{\Omega}_e$ radians since the time of transmission, where $\dot{\Omega}_e$ the rotation rate of the earth. The satellite position at the time of transmission can be calculated using the broadcast ephemeris information. This must be rotated from earth coordinates at the time of transmission to the earth coordinates at the time of reception using the travel time of the signal.

A third correction is associated with the satellite clock. This uses part of the information broadcast by the satellites. (See section 20.3.3.3.3.1 of reference [6].)[11]

There are further biases on the order of 100 meters associated with delays in the ionosphere, the troposphere and the DOD introduced "Selective Availability" (SA). These cannot be easily modeled. However, if a second receiver is placed at a known location ($\mathbf{R}^0$), and the same ephemeris data is used to determine the satellite locations ($\mathbf{S}_i^0$) at the time of transmission, a correction to the pseudo range can be generated,

$$\Delta r_i = \| \mathbf{R}^0 - \mathbf{S}_i^0 \| - r_i^0 \tag{3.18}$$

where r_i^0 is the pseudo range obtained from the observed code phase $\hat{C}_i$ at the known receiver location (see equation (3.17) above). This is the differential correction factor we use to correct the pseudo range. If the receiver, for which the position solution is desired, is "near" [12] the reference receiver this correction will improve the final position estimate by removing ionospheric, tropospheric and "selective availability" effects. If a second receiver, at a known site, is not available these effects must be considered noise which will lead to larger values for σ and larger errors in the final position estimate. Furthermore, if this correction is not made the selective availability component may make the model of independent normal errors in the pseudo range values a poor assumption. Finally, without the differential correction, the resulting position estimates will have a large correlation from one time to another.

[11]Appendix C lists the information needed from the ephemeris message.

[12]Section 10.10 of Reference [7] suggests this is of the order $\leq 100km$.

The next section provides a prescription for correcting biases and estimating position.

3.4. Algorithm

In this section we present a step by step algorithm for computing GPS position using the "raw" C/A code data available from a GPS receiver. To do this we need to introduce some notation. Table 3.1 provides a list of the variables and constants that appear in the algorithm.

The variables are divided into roughly three categories; first those input to the algorithm from external sources labeled *input* in table 3.1, second those calculated and used internal to the algorithm labeled *calculated* and, third the results labeled *output*.[13]

The positioning solution is obtained by the following steps:

1. Obtain ephemeris information, the values in table C.1 in appendix C, and differential range correction factor for all visible satellites (Δr_i). (This is computed using equation (3.18)).

2. Obtain the code phase (C_i) for the satellites to be used in the position solution.

3. For each satellite, initialize the travel time for the GPS signal to zero ($t_i^{(1)} = 0$), select an initial position $\mathbf{R}^{(1)}$, and set $k = 1$.

The iteration loop begins here:

4. Calculate the position of the satellites $\mathbf{P}_i^{(k)}$ (see appendix C for details) at the time that the signal was transmitted from each satellite ($t - t_i^{(k)}$).

5. Rotate the position of each of the satellites from XYZ coordinates at transmission time to XYZ coordinates at reception time. This corrects the satellite position for earth's rotation at the rate of $\dot{\Omega}_e$ during the travel time estimated in this

[13]The units of the variables in table 3.1 are in the mks (meter-kilogram-second) system and abbreviated meters (m), kilometers (km), and seconds (s).

Table 3.1: Algorithm Variables and Constants

V	Approximate satellite speed (3500 m/s) Based on a twelve hour orbital period and a height of 20,200 km	*constant*
Λ	2.99792458×10^5 meters, the wavelength of the C/A code C/A code repeats every millisecond ([7] 6.7) and $\Lambda = c \cdot 10^{-3}$	*constant*
F	$-4.442809305 \times 10^{-10}$ $(\mathrm{s/m^{1/2}})$ See section 20.3.3.3.3.1 of reference [6]	*constant*
c	2.99792458×10^8 meters/second the speed of light See section 20.3.4.3 of reference [6]	*constant*
$\dot{\Omega}_e$	Rotation rate of the earth $7.2921151467 \times 10^{-5}$ radians/s See table 20-IV of reference [6]	*constant*
δ	Convergence criteria (m)	*input*
C_i	Code phase for satellite i from the receiver (m)	*input*
t	GPS time at which the code phases are measured (s)	*input*
$\mathbf{a}_i$	Satellite clock correction vector $[a_{0i}, a_{1i}, a_{2i}]^T$ for satellite i	*input*
t^i_{oe}	Reference time for satellite i ephemeris (s)	*input*
A_i	Semi major axis of the orbit for satellite (m) i	*input*
e_i	Eccentricity of the orbit for satellite i	*input*
$\mathbf{R}^{(1)}$	The initial approximate receiver location (m)	*input*
i	Satellite index	*temporary*
k	Iteration index	*temporary*
Δr_i	Differential pseudo range correction for satellite i (see Equation (3.18)), value is zero if unavailable (m)	*calculated*
n_i	The number of C/A code cycles between transmission and reception for satellite i	*calculated*
$\mathbf{S}_i^{(k)}$	Position of satellite i at transmission time in earth coordinates at time of reception, time t (m)	*calculated*
$\mathbf{P}_i^{(k)}$	Position of satellite i at transmission time in earth coordinates at time of transmission, see appendix C (m)	*calculated*
Δt_i^r	The relativistic time correction for satellite i (s)	*calculated*
$t_i^{(k)}$	Travel time from satellite i to the receiver at iteration k (s)	*calculated*
$\Delta t_i^{(k)}$	The total time correction for satellite i	*calculated*
$E_i^{(k)}$	Eccentric anomaly for satellite i, see appendix C	*calculated*
$\mathbf{R}^{(k)}$	For $k > 1$, the approximate receiver location at iteration k	*output*
$t_b^{(k)}$	Approximate receiver clock bias estimate at iteration k (m)	*output*

iteration $(t_i^{(k)})$

$$\theta_i^{(k)} = -t_i^{(k)}\dot{\Omega}_e \tag{3.19}$$

$$\mathbf{S}_i^{(k)} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}_i^{(k)}. \tag{3.20}$$

6. Calculate the satellite clock bias $(\Delta t_i^{(k)})$ using the correction vector broadcast by the satellites $(\mathbf{a}_i)$ with the difference between the ephemeris reference time (t_{oe}), and the time of the code phase transmission $s_i^{(k)} = t - t_i^{(k)} - t_{oe}^i$,

$$\Delta t_i^{(k)} = a_{0i} + a_{1i}s_i^{(k)} + a_{2i}(s_i^{(k)})^2 + \Delta t_r^r \tag{3.21}$$

where,

$$\Delta t_i^r = F e_i A_i^{\frac{1}{2}} \sin E_i^{(k)}$$

(See appendix C for the calculation of $E_i^{(k)}$.)

7. Construct the corrected code phase for each satellite,

$$\hat{C}_i^{(k)} = C_i + c\Delta t_i^{(k)}. \tag{3.22}$$

8. If $k = 1$ then calculate the integer number of cycles (n_i) in the range to each satellite and the initial timing bias estimate $(t_b^{(1)})$ using the method presented in appendix B.

9. Construct the corrected pseudo range for each of the satellites using the differential pseudo range correction Δr_i,[14]

$$r_i^{(k)} = \hat{C}_i^{(k)} + n_i\Lambda + \Delta r_i \tag{3.23}$$

and denote $\left[r_1^{(k)}, r_2^{(k)}, \dots r_N^{(k)}\right]^T$ by $\mathbf{r}^{(k)}$.

[14]Note that if this is the reference station, $\|\mathbf{R}^0 - \mathbf{S}_i\| - (\hat{C}_i + n_i\Lambda)$ is output as the differential pseudo range correction Δr_i.

10. Use the Gauss Newton method described in the last section with the notation $\mathbf{x}^{(k)} = \begin{bmatrix} \mathbf{R}^{(k)} \\ t_b^{(k)} \end{bmatrix}$ and equation (3.15),

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{G}(\mathbf{x}^{(k)})^{-1}\mathbf{g}(\mathbf{x}^{(k)}, \mathbf{r}^{(k)}),$$

to determine $\mathbf{R}^{(k+1)}$ and $t_b^{(k+1)}$.

11. Calculate the new travel time estimate based on the new position,

$$t_i^{(k+1)} = \frac{\| \mathbf{R}^{(k)} - \mathbf{S}_i^{(k)} \|}{c} \tag{3.24}$$

12. Check for convergence.

If the new position, travel time and bias estimate differ from the previous by an amount greater than the selected accuracy of δ meters,

$$|t_b^{(k+1)} - t_b^{(k)}| + \| \mathbf{R}^{(k+1)} - \mathbf{R}^{(k)} \| + \sum_{i=1}^{N} V|t_i^{(k+1)} - t_i^{(k)}| > \delta \tag{3.25}$$

set $k = k + 1$ and *go back to step 4*.

13. Sanity check, if $\| \mathbf{R}^{(k+1)} - \mathbf{R}^{(1)} \| > \frac{\Lambda}{2}$ the positioning has failed, otherwise it was successful and the solution is $\mathbf{R}^{(k+1)}$.

Using the steps in the prescription above the position of a receiver can be calculated. This position depends on the measurement of the pseudo ranges to a set of satellites. The random errors in the measurements result in errors in the position estimates. Our assumption that the errors in the pseudo range to each satellite are independent does not imply that the statistics of the components of the position estimate are independent. The statistics of the errors in the position estimate are derived in the next section.

3.5. Statistics and Errors in the Solution

This section considers the second order statistics of the position estimate. We have assumed independence of the errors in the measured pseudo ranges. However, errors in the components of the timing bias and position estimate

$$\hat{\mathbf{x}} = \begin{bmatrix} x^{(k+1)} \\ y^{(k+1)} \\ z^{(k+1)} \\ t_b^{(k+1)} \end{bmatrix} \tag{3.26}$$

resulting from the maximum likelihood methodology are not necessarily independent. The error in the estimate is the difference between the true position and timing bias (denoted by $\bar{\mathbf{x}}$) and the maximum likelihood estimate (denoted by $\hat{\mathbf{x}}$). If the position estimate is to be used in further processing, such as Kalman filtering for the position and velocity, the relationship between the statistics of the components of the calculated position must be quantified.

The covariance between the errors in the components of the estimated position is,

$$\mathbf{C} = E\{(\hat{\mathbf{x}} - \bar{\mathbf{x}})(\hat{\mathbf{x}} - \bar{\mathbf{x}})^T\}, \tag{3.27}$$

where $E\{\cdot\}$ is the expected value operator. The diagonal elements of this covariance matrix are the variance values for each of the estimated components and the off diagonal elements represent the covariance between the components (if an off diagonal element is zero the corresponding components are independent[15]). The matrix $\mathbf{C}$ expresses the variability of the position estimate. However, in the positioning problem the observables are the pseudo ranges and not the positions. In this section, we present a method for approximating the covariance of the position solution from the covariance of the pseudo ranges. Our estimation procedure maps the observables, the (pseudo ranges to the satellites, $\mathbf{r} = [r_1, r_2, \ldots r_N]^T$) to the position estimates and receiver clock bias. For each observed $\hat{\mathbf{r}}$ our estimate of $\hat{\mathbf{x}}$ minimizes $f(\mathbf{x}, \hat{\mathbf{r}})$ with respect to $\mathbf{x}$, thus $\nabla_x f(\hat{\mathbf{x}}, \hat{\mathbf{r}}) = 0$. If we expand the gradient of the objective function, $\nabla_x f(\mathbf{x}, \mathbf{r})$, in a first order Taylor series about the $(\bar{\mathbf{x}}, \bar{\mathbf{r}})$ we get,

$$\nabla_x f(\hat{\mathbf{x}}, \hat{\mathbf{r}}) \approx \nabla_x f(\bar{\mathbf{x}}, \bar{\mathbf{r}}) + \nabla_{xx} f(\bar{\mathbf{x}}, \bar{\mathbf{r}})\,(\hat{\mathbf{x}} - \bar{\mathbf{x}}) + \nabla_{xr} f(\bar{\mathbf{x}}, \bar{\mathbf{r}})\,(\hat{\mathbf{r}} - \bar{\mathbf{r}}), \tag{3.28}$$

where $\hat{\mathbf{r}}$ is the observed pseudo range vector, $\hat{\mathbf{x}}$ is the corresponding estimate of position and clock bias, $\bar{\mathbf{x}}$ is the true receiver position and clock bias, and $\bar{\mathbf{r}}$ is the true satellite

[15]This is true up to the approximation that the position errors are normally distributed.

range vector. The gradient $\nabla_x f(\bar{\mathbf{x}}, \bar{\mathbf{r}})$ is zero at the true location and clock bias $\bar{\mathbf{x}}$ (see equation (3.6)), and $\nabla_x f(\hat{\mathbf{x}}, \hat{\mathbf{r}}) = 0$ is enforced at $\hat{\mathbf{x}}$. Substituting these values in equation (3.28) we obtain

$$\nabla^2_{xx} f(\bar{\mathbf{x}}, \bar{\mathbf{r}})\,(\hat{\mathbf{x}} - \bar{\mathbf{x}}) \approx -\nabla^2_{xr} f(\bar{\mathbf{x}}. \bar{\mathbf{r}})\,(\hat{\mathbf{r}} - \bar{\mathbf{r}})\,. \tag{3.29}$$

To simplify notation we can define

$$\mathbf{A} = \nabla^2_{xx} f(\bar{\mathbf{x}}, \bar{\mathbf{r}}) \qquad \mathbf{B} = \nabla^2_{xr} f(\bar{\mathbf{x}}, \bar{\mathbf{r}}) \tag{3.30}$$

so that approximation (3.29) becomes,

$$\mathbf{A}(\hat{\mathbf{x}} - \bar{\mathbf{x}}) \approx -\mathbf{B}(\hat{\mathbf{r}} - \bar{\mathbf{r}}). \tag{3.31}$$

The errors in the position can be approximated by,

$$(\hat{\mathbf{x}} - \bar{\mathbf{x}}) \approx -\mathbf{A}^{-1}\mathbf{B}(\hat{\mathbf{r}} - \bar{\mathbf{r}}). \tag{3.32}$$

The covariance matrix for the errors in the position as a function of the observed pseudo ranges is approximated by,

$$E\{(\hat{\mathbf{x}} - \bar{\mathbf{x}})(\hat{\mathbf{x}} - \bar{\mathbf{x}})^T\} \approx \mathbf{A}^{-1}\mathbf{B}\ E\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\}\ \mathbf{B}^T(\mathbf{A}^{-1})^T \tag{3.33}$$

We assume that the errors in the pseudo ranges are independent which implies that $E\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\}$ is a diagonal matrix with the variance of each satellite's pseudo range as the elements. In section 3.2 we assumed that the variances are equal so that $E\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\} = \sigma^2\mathbf{I}$, and equation (3.33) becomes,

$$E\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\} \approx \sigma^2\mathbf{A}^{-1}\mathbf{B}\mathbf{B}^T(\mathbf{A}^{-1})^T. \tag{3.34}$$

Which can be simplified by noting that the matrix $\mathbf{A}^{-1}$ is symmetric (see equation (3.30)) so that,

$$E\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\} \approx \sigma^2\mathbf{A}^{-1}\mathbf{B}\mathbf{B}^T\mathbf{A}^{-1}. \tag{3.35}$$

In addition we approximate the matrix $\mathbf{A}$ by $\mathbf{G}(\bar{\mathbf{x}})$ (see equation (3.14)) and the matrix $\mathbf{B}$ is[16]

$$\mathbf{B} = \nabla^2_{xr} f(\bar{\mathbf{x}}) = -\mathbf{Q}(\bar{\mathbf{x}}) \tag{3.36}$$

[16]See appendix A for a derivation.

Approximating $\mathbf{G}(\bar{\mathbf{x}})$ by $\mathbf{G}(\hat{\mathbf{x}})$ and $\mathbf{Q}(\bar{\mathbf{x}})$ by $\mathbf{Q}(\hat{\mathbf{x}})$ we obtain,

$$E\left\{(\hat{\mathbf{r}} - \bar{\mathbf{r}})(\hat{\mathbf{r}} - \bar{\mathbf{r}})^T\right\} \approx \sigma^2 \mathbf{G}(\hat{\mathbf{x}})^{-1}\mathbf{Q}(\hat{\mathbf{x}})\mathbf{Q}(\hat{\mathbf{x}})^T\mathbf{G}(\hat{\mathbf{x}})^{-1}. \tag{3.37}$$

Using equation (3.14) to replace $\mathbf{G}(\hat{\mathbf{x}})$ yields,

$$E\{(\hat{\mathbf{x}} - \bar{\mathbf{x}})(\hat{\mathbf{x}} - \bar{\mathbf{x}})^T\} \approx \sigma^2(\mathbf{Q}(\hat{\mathbf{x}})\mathbf{Q}(\hat{\mathbf{x}})^T)^{-1} \tag{3.38}$$

This is an approximation of the covariance matrix for the position solution, given that the variance of the pseudo range data is $\sigma^2\mathbf{I}$. The covariance matrix is constructed using only a few additional calculations and as such does not add a significant computational burden to the positioning solution. The covariance matrix is appropriate for use with subsequent estimation and control processing such as Kalman filtering.

4. Conclusion

This report was presented in two major chapters corresponding to the two major theses asserted in the implementation of the *Investigation of GPS and GIS for Traveler Information* project.

The first thesis was that two advanced technologies, GIS and GPS, can be combined in a distributed computing environment to deliver traveler information. The second chapter of this report described the architecture and implementation of such a traveler information system that combines GPS probe vehicle locations, traffic congestion information, and digital maps in a distributed computing environment. The components reported on include (1) a GIS system we have called the IVHS map display client, (2) a GPS server system to provide the location of probe vehicles in real time, and (3) a traffic congestion server to provide a network interface to the WSDOT's inductance loop system located in the freeways surrounding metropolitan Seattle. This project successfully implemented a demonstration system that combines the technologies mentioned. Further this project defined a general architecture that is extensible. The architecture and the implementation can easily include additional sensors or information sources and can be extended to a larger geographic area or even used in other geographic regions (other cities or states).

The second thesis of this project was that a maximum likelihood approach can be used to solve the GPS positioning problem. This is the topic of the third major chapter of this report. The third chapter presents a comprehensive methodology for positioning that uses the basic information, ephemeris and code phase, from the global positioning system satellites. It first developed a positioning algorithm that uses unbiased range estimates when four or more satellites are available. It then provided an overview of the bias correction inherent in the GPS positioning problem and enumerated an algorithm for positioning that includes corrections for the biases. Finally, it outlined a method for approximating the second order statistics of the errors in the position estimation and provided a closed form for this approximation. The methods presented provide a uniform positioning solution that can be used with any GPS receiver that provides C/A code phase measurements.[1]

The methodology and example software to combine GPS position solutions with digital maps can potentially be used by WSDOT for a range of applications. These might include fleet management for a maintenance fleet, vehicle tracking for IVHS project evaluation, real time mapping of remote areas as well as the traveler information application which was the starting point for this project. The overall architecture developed to implement this project provides a framework for combining information from a number of sources in a structured way and could form the basis for a larger information backbone to implement future IVHS projects.

WSDOT could use the software developed with GPS receivers from a variety of vendors with limited modifications. This project has demonstrated that the technology to combine GIS and GPS is practical and, by the software produces as part of the project, made such technology directly available to WSDOT. The existence of the demonstration software and the expertise produced by the project will help to mitigate the cost of any possible future forays into the use of GIS and GPS for traveler information.

[1]P-code phase measurements can be used with the methodology presented. The P code phase measurements do not have the ambiguity associated with the number of integer wavelengths between the satellite and the receiver (see equation (3.2)

BIBLIOGRAPHY

[1] M. Haselkorn and J. Spyridakis and W. Barfield and B. Goble and M. Garner. Designing and Implementing a PC-Based, Graphical, Interactive, Real-Time Advanced Traveler Information System that Meets Commuter Needs . In *Proceedings of the 1991 Vehicle Navigation and Information Systems Conference*, pages 1045–1048, 1991.

[2] F.J. Mammano and R. Sumner . Pathfinder Status and Implementation Experience . In *Proceedings of the 1991 Vehicle Navigation and Information Systems Conference*, 1991.

[3] D.E. Boyce and A. Kirson and J.L. Schofer . Design and Implementation of ADVANCE: The Illinois Dynamic Route Guidance Demonstration Program . In *Proceedings of the 1991 Vehicle Navigation and Information Systems Conference*, 1991.

[4] J.J. Dongarra, C.B. Moler, J.R. Bunch, and G.W. Stewart. *LINPACK*. SIAM, 4th edition, 1984.

[5] D.J. Dailey and B.M. Bell. A Prescription for GPS positioning . *Transactions on Aeroapce and Electronic Systems*, In Review.

[6] Rockwell International. *Navstar GPS Space Segment/Navigation User Interface.* Rockwell International, 1984.

[7] D. Wells. *Guide to GPS Positioning.* Canadian GPS Associates, 1987.

[8] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, second edition, 1987.

[9] D.G. Moursund and C.S. Duris. *Elementary theory and Applications of Numerical Methods.* McGraw-Hill, 1967.

[10] G. Bomford. *Geodesy.* Clarendon Press, 4th edition, 1980.

[11] M. Kumar. World geodetic system 1984: A modern and accurate global reference frame. *Marine Geodesy*, 12:117–126, 1984.

[12] Defense Mapping Agency. Supplement to DOD World Geodetic System 1984 Technical Report. Technical Report DMATR8350.2-A, Department of Defense, December 1987.

A. Appendix - Derivative Calculations

This appendix provides details for calculating $\nabla_x q_i(\mathbf{x}, \mathbf{r})$ which is used to define $\mathbf{Q}(\mathbf{x})$ in equation (3.14) and $\nabla^2_{xr} f(\mathbf{x}, \mathbf{r})$ which is used to define $\mathbf{B}$ in equation (3.30). The vector $\mathbf{q}$ is defined as,

$$\mathbf{q}(\mathbf{R}, t_b, \mathbf{r}) = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_N \end{bmatrix} = \begin{bmatrix} (\| \mathbf{R} - \mathbf{S}_1 \| - r_1 + t_b) \\ (\| \mathbf{R} - \mathbf{S}_2 \| - r_2 + t_b) \\ \vdots \\ (\| \mathbf{R} - \mathbf{S}_N \| - r_N + t_b) \end{bmatrix}. \tag{A.1}$$

and where $\| \cdot \|$ is the Euclidean distance.

$\mathbf{Q}_i(\mathbf{x})$, which is the partial derivative of q_i with respect to the position and timing bias, is[1]

$$\mathbf{Q}_i(\mathbf{x}) = \nabla_x q_i(\mathbf{x}, \mathbf{r}) = \begin{bmatrix} \dfrac{\partial q_i}{\partial R_x} \\ \\ \dfrac{\partial q_i}{\partial R_y} \\ \\ \dfrac{\partial q_i}{\partial R_z} \\ \\ \dfrac{\partial q_i}{\partial t_b} \end{bmatrix} = \begin{bmatrix} \dfrac{(R_x - S_i^x)}{\| \mathbf{R} - \mathbf{S}_i \|} \\ \\ \dfrac{(R_y - S_i^y)}{\| \mathbf{R} - \mathbf{S}_i \|} \\ \\ \dfrac{(R_z - S_i^z)}{\| \mathbf{R} - \mathbf{S}_i \|} \\ \\ 1 \end{bmatrix} \tag{A.2}$$

where R_x, R_y, R_z are the components of the receiver position $\mathbf{R}$, S_i^X, S_i^y, S_i^z are the components of $\mathbf{S}_i$, the position of the ith satellite, $\mathbf{x} = [\mathbf{R},\ t_b]^T$, and $\mathbf{r} = [r_1, r_2, \ldots r_N]^T$. As defined in the equation (3.11) the matrix $\mathbf{Q}(\mathbf{x})$ is $[\mathbf{Q}_1, \mathbf{Q}_2 \ldots \mathbf{Q}_N]$.

The $\mathbf{B}$ matrix, defined in equation (3.30) is constructed from the partial derivative

[1]See equations (3.8) and (3.11).

of the gradient $\mathbf{g}(\mathbf{x}, \mathbf{r}) = \nabla_x f(\mathbf{x}, \mathbf{r})$ with respect to each of the pseudo ranges,

$$\mathbf{B} = \nabla^2_{xr} f(\bar{\mathbf{x}}, \bar{\mathbf{r}}) = \left[\frac{\partial \mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{r}})}{\partial r_1}, \frac{\partial \mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{r}})}{\partial r_2}, \ldots \frac{\partial \mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{r}})}{\partial r_N} \right]. \tag{A.3}$$

Using equation (3.12) the partial derivative with respect to r_i can be written,

$$\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{r})}{\partial r_i} = \left[\frac{\partial \mathbf{Q}(\mathbf{x})}{\partial r_i} \mathbf{q}(\mathbf{x}, \mathbf{r}) + \mathbf{Q}(\mathbf{x}) \frac{\partial \mathbf{q}(\mathbf{x}, \mathbf{r})}{\partial r_i} \right]. \tag{A.4}$$

From equation (A.2)

$$\frac{\partial \mathbf{Q}(\mathbf{x})}{\partial r_i} = 0 \tag{A.5}$$

and therefore

$$\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{r})}{\partial r_i} = \mathbf{Q}(\mathbf{x}) \left[\frac{\partial \mathbf{q}(\mathbf{x}, \mathbf{r})}{\partial r_i} \right]. \tag{A.6}$$

The components of the vectors $\frac{\partial \mathbf{q}}{\partial r_i}$ are equal to zero with the exception of the *ith* component which is -1. Thus,

$$\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{r})}{\partial r_i} = -\mathbf{Q}_i(\mathbf{x}). \tag{A.7}$$

Combining this with equation (A.3), the matrix **B** can then be written in terms of $\mathbf{Q}(\bar{\mathbf{x}})$,

$$\mathbf{B} = -\mathbf{Q}(\bar{\mathbf{x}}). \tag{A.8}$$

B. Appendix - Range Ambiguity Solution

The true number of integer wavelengths (Λ) of the C/A code between the position of the ith satellite ($\mathbf{S}_i$) and the initial estimate of the receiver position ($\mathbf{R}^{(1)}$) is the range ambiguity (n_i^t). The approximations for the range are,

$$\|\mathbf{R}^{(1)} - \mathbf{S}_i\| \approx \hat{C}_i^{(1)} + n_i^t\Lambda - t_b^t \qquad i = 1, 2, \ldots N \tag{B.1}$$

where $\hat{C}_i^{(1)}$ is the code phase measurement, corrected using the satellite clock error and the travel time bias from the first iteration of the positioning algorithm, and the superscript t indicates the true values. As noted in the text the value of t_b^t, the bias of the receiver clock relative to GPS time, is unique to a receiver and equal for all of the satellites. Our goal in this appendix is to arrive at values for the set of range ambiguities $[n_1, n_2, \ldots n_N]$ and a corresponding receiver timing bias approximation, t_b. For N satellites this problem has $N + 1$ unknowns.

We add and subtract $m\Lambda$ to approximation (B.1) to get

$$\|\mathbf{R}^{(1)} - \mathbf{S}_i\| \approx \hat{C}_i^{(1)} + (n_i^t + m)\Lambda - (t_b^t + m\Lambda) \qquad i = 1, 2, \ldots N, \tag{B.2}$$

which is true for all integer values of m. This suggests that adding m to each of the range ambiguities is equivalent to adding $m\Lambda$ to the timing bias. Define,

$$n_i^m = n_i^t + m, \tag{B.3}$$

$$t_b^m = t_b^t + m\Lambda. \tag{B.4}$$

Approximation (B.2) can be written as,

$$\|\mathbf{R}^{(1)} - \mathbf{S}_i\| \approx \hat{C}_i^{(1)} + n_i^m\Lambda - t_b^m \qquad i = 1, 2, \ldots N. \tag{B.5}$$

We choose m to be the integer such that,

$$n_1^m = n_1^t + m = int\left[\frac{\|\mathbf{R}^{(1)} - \mathbf{S}_1\| - \hat{C}_1^{(1)}}{\Lambda}\right] \tag{B.6}$$

where $int[x]$ is the integer closest to x. With this value for n_1^m the approximations in (B.5) become,

$$\mathbf{A}\mathbf{y}_m \approx \mathbf{b} \tag{B.7}$$

where,

$$\mathbf{y}_m = \begin{bmatrix} n_2^m \\ n_3^m \\ \vdots \\ n_N^m \\ t_b^m \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & 0 & \dots & 0 & -1 \\ \Lambda & 0 & \dots & 0 & -1 \\ 0 & \Lambda & \dots & 0 & -1 \\ 0 & 0 & \ddots & 0 & -1 \\ 0 & 0 & \dots & \Lambda & -1 \end{bmatrix} \tag{B.8}$$

$$\text{and,} \quad \mathbf{b} = \begin{bmatrix} \|\mathbf{R}^{(1)} - \mathbf{S}_1\| - C_1 - n_1^m\Lambda \\ \|\mathbf{R}^{(1)} - \mathbf{S}_2\| - C_2 \\ \|\mathbf{R}^{(1)} - \mathbf{S}_3\| - C_3 \\ \vdots \\ \|\mathbf{R}^{(1)} - \mathbf{S}_N\| - C_N \end{bmatrix}. \tag{B.9}$$

The exact solution of $\mathbf{A}\mathbf{y}_e = \mathbf{b}$ is

$$\mathbf{y}_e = \begin{bmatrix} n_2^e \\ n_3^e \\ \vdots \\ n_N^e \\ t_b^e \end{bmatrix} = \mathbf{A}^{-1}\mathbf{b} \tag{B.10}$$

where,

$$\mathbf{A}^{-1} = \frac{1}{\Lambda}\begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ -1 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ -1 & 0 & 0 & \dots & 1 \\ -\Lambda & 0 & 0 & \dots & 0 \end{bmatrix}. \tag{B.11}$$

We assume that $|n_i^e - n_i^m| < 0.5$ so that y_m is close to y_e. The range ambiguity for the ith satellite and the estimate of the timing bias can be written,

$$t_b^m = \hat{C}_1^{(1)} + n_1^m \Lambda - \|\mathbf{R}^{(1)} - \mathbf{S}_1\|. \tag{B.12}$$

$$n_i^m = int\left\{\frac{1}{\Lambda}\left(\|\mathbf{R}^{(1)} - \mathbf{S}_i\| - \hat{C}_i^{(1)} + t_b^m\right)\right\} \tag{B.13}$$

This solution provides the range ambiguities (n_i^m), and a corresponding initial estimate of the timing bias t_b^m which are referred to as n_i and $t_b^{(1)}$ in step 7 of the algorithm.

C. Appendix - Satellite Position Calculation

This appendix summarizes the calculation of the position for the *ith* satellite (referred to as $\mathbf{P}_i^{(k)}$ in step 4 of the positioning algorithm), and eccentric anomaly (referred to as $E_i^{(k)}$ in step 6 of the positioning algorithm), from the coefficients and the GPS time transmitted by the satellite. This is done based on the recommendation made in table 20-IV of reference [6]. The notation used comes directly form that source. The reader also referred to section 5.23 of reference [7] for an overview of the subject. The values needed from each satellite are shown in table C.1:[1]

The individual satellite location is determined by going through the following calculations sequentially.

1. The time from the ephemeris reference epoch,

$$t_k = s - t_{oe}^{(i)} \tag{C.1}$$

modulo one week (604,800 seconds), where s is the GPS second of the week at which the position is calculated.[2]

2. The computed mean motion,

$$n_0 = \left(\frac{\mu}{A^3}\right)^{\frac{1}{2}} \tag{C.2}$$

3. Mean anomaly,

$$M = M_0 + (n_0 + \Delta n)t_k \tag{C.3}$$

[1]See table 20-II of reference [6]
[2]In step 4 of the algorithm $s = (t - t_i^k)$

Table C.1: Values from GPS ephemeris message.

$t_{oe}^{(k)}$	= Reference time (s)
Δn	= Mean motion difference from computed value (1/s)
e	= Eccentricity of satellite orbit
A	= Semi major axis (m)
M_0	= Mean anomaly at the reference time
ω	= Argument of perigee (radians)
C_{us}	= Amplitude of sine correction term for argument of latitude
C_{uc}	= Amplitude of cosine correction term for argument of latitude
C_{rs}	= Amplitude of sine correction term for orbital radius
C_{rc}	= Amplitude of cosine correction term for orbital radius
C_{is}	= Amplitude of sine correction term for inclination
C_{ic}	= Amplitude of cosine correction term for inclination
$\dot{\Omega}$	= Rate of right ascension (radians/s)
Ω_0	= Longitude of ascending node of orbit plane at weekly epoch at the reference time (radians)
i_0	= Inclination angle at reference time (radians)
$\dot{I}$	= Rate of inclination angle (radians/s)

Table C.2: Other input for Satellite position calculation.

s	= GPS second of the week at which the position is calculated
μ	= the universal gravitational parameter $3.986008 \times 10^{14}\text{m}^3/\text{s}^2$
$\dot{\Omega}_e$	= the rotation rate of the earth is $7.292115147 \times 10^{-5}$ radians/s

4. Next solve Kepler's equation for the eccentric anomaly (E)[3],

$$M = E - e\sin(E). \tag{C.4}$$

The mapping $h(E) = M + e\sin(E)$ is a contraction mapping (because e is small) and its fixed point solves equation (C.4). A direct application of Theorem 1-5-4 of reference [9] yields the following method for solving (C.4).

Set $E_1 = M$

[3]Note that the value E presented here is equal to the value $E_i^{(k)}$ in step 6 of the receiver positioning algorithm.

for $j = 1$ to 4, Set $E_j = M + e\sin(E_{j-1})$

Set $E = E_4$

5. The true anomaly is then calculated,

$$v = \tan^{-1}\left(\frac{\sin(E)\sqrt{1-e^2}}{\cos(E) - e}\right) \tag{C.5}$$

6. The argument of latitude is calculated,

$$\phi = v + \omega. \tag{C.6}$$

7. The corrected argument of latitude is

$$u = \phi + C_{uc}\cos 2\phi + C_{us}\sin 2\phi. \tag{C.7}$$

8. The corrected radius is,

$$r = A(1 - e\cos E) + C_{rc}\cos 2\phi + C_{rs}\sin 2\phi. \tag{C.8}$$

9. The corrected inclination is,

$$i_k = i_0 + C_{ic}\cos 2\phi + C_{is}\sin 2\phi + \dot{I}t_k. \tag{C.9}$$

10. The coordinates in the orbital plane are calculated,

$$x' = r\cos u \tag{C.10}$$

$$y' = r\sin u. \tag{C.11}$$

11. The corrected longitude of the ascending node,

$$\Omega = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)t_k - \dot{\Omega}_e t_{oe} \tag{C.12}$$

12. The position is then rotated to earth fixed coordinates,

$$x = x'\cos\Omega - y'\cos i_k \sin\Omega \tag{C.13}$$

$$y = x'\sin\Omega + y'\cos i_k \cos\Omega \tag{C.14}$$

$$z = y'\sin i_k \tag{C.15}$$

This is the position of the satellite in the earth fixed XYZ coordinates at time s. The satellite position in step 5 of the positioning algorithm is, $\mathbf{P}_i^{(k)} = [x, y, z]^T$, where x, y, and z are defined by equations (C.13), (C.14), and (C.15).

D. Appendix - Coordinate Transformations

D.1. Latitude, longitude, height to rectangular coordinates

The transformation from latitude ϕ, longitude λ and height h to an earth fixed rectangular coordinate system is a straight forward calculation. The equations presented here are taken from appendix C of reference [10], whose notation conventions are used (with the exception that h is the height above the spheroid instead of the height above the geoid). The relationship between geodetic coordinates ϕ, λ, h and cartesian coordinates x, y, z is given by,

$$\begin{aligned} x(\phi,\lambda,h) &= (\nu + h)\cos\phi\cos\lambda && \text{(D.1)} \\ y(\phi,\lambda,h) &= (\nu + h)\cos\phi\sin\lambda && \text{(D.2)} \\ z(\phi,\lambda,h) &= \Big(\nu(1-e^2) + h\Big)\sin\phi && \text{(D.3)} \end{aligned}$$

where,

$$\nu = \frac{a}{(1 - e^2\sin^2\phi)^{1/2}},$$

x, y, z	–	Rectangular Coordinates,
ϕ, λ, h	–	Latitude, longitude, and height above the spheroid
a	–	Semi-major axis of the spheroid (6378137 meters)[1]
e	–	Eccentricity of the earth as a spheroid (0.081819191)[2]

In addition, the partial derivatives of the position transformation with respect to the variables are used for the inverse mapping. Noting

$$\frac{\partial \nu}{\partial \phi} = \frac{e^2 \sin(\phi)\cos(\phi)\nu}{(1 - e^2 \sin^2 \phi)} \tag{D.4}$$

the partial derivatives of the cartesian coordinate transformations with respect to each of the geodetic coordinated are shown in table D.1.

Table D.1: Derivatives

	$\frac{\partial}{\partial \phi}$	$\frac{\partial}{\partial \lambda}$	$\frac{\partial}{\partial h}$
x	$\frac{\partial \nu}{\partial \phi}\cos\phi\cos\lambda - (\nu + h)\sin\phi\cos\lambda$	$-(\nu + h)\cos\phi\sin\lambda$	$\cos\phi\cos\lambda$
y	$\frac{\partial \nu}{\partial \phi}\cos\phi\sin\lambda - (\nu + h)\sin\phi\sin\lambda$	$(\nu + h)\cos\phi\cos\lambda$	$\cos\phi\sin\lambda$
z	$\frac{\partial \nu}{\partial \phi}(1 - e^2)\sin\phi + (\nu(1 - e^2) + h)\cos\phi$	0	$\sin\phi$

Equations (D.1), (D.2), (D.3) provide a closed form solution to the transformation from geodetic to rectangular coordinates. Table D.1 provides a closed form expression for the partial derivatives which are used to obtain the inverse transformation.

D.2. Rectangular coordinates to latitude, longitude, height

The mapping from cartesian earth fixed coordinates ($\mathbf{X} = [x\ y\ z]^T$) to geodetic coordinates of latitude, longitude and height ($\mathbf{L} = [\phi\ \lambda\ h]^T$) is done using Newton's method. Define

$$f(\mathbf{L}) = \mathbf{X} = \begin{bmatrix} x(\phi, \lambda, h) \\ y(\phi, \lambda, h) \\ z(\phi, \lambda, h) \end{bmatrix}. \tag{D.5}$$

[1]See reference [11] and page 3-46 of reference [12].
[2]See reference [11] and page 3-46 of reference [12].

Given $\mathbf{X}^0$ we need to find $\mathbf{L}^0$ such that

$$f(\mathbf{L}^0) - \mathbf{X}^0 = 0 \tag{D.6}$$

The Newton's method iteration for solving (D.6) is,

$$\mathbf{L}_{j+1} = \mathbf{L}_j - (\nabla f(\mathbf{L}_j)^T)^{-1} \left[f(\mathbf{L}_j) - \mathbf{X}^0\right] \tag{D.7}$$

where

$$\nabla f(\mathbf{L}) = \begin{bmatrix} \frac{\partial x}{\partial \phi} & \frac{\partial y}{\partial \phi} & \frac{\partial z}{\partial \phi} \\ \frac{\partial x}{\partial \lambda} & \frac{\partial y}{\partial \lambda} & \frac{\partial z}{\partial \lambda} \\ \frac{\partial x}{\partial h} & \frac{\partial y}{\partial h} & \frac{\partial z}{\partial h} \end{bmatrix} \tag{D.8}$$

The iteration is initialized by setting $L_1 = [\phi_1, \lambda_1, h_1]$ where

$$\phi_1 = \sin^{-1}(\frac{z^0}{r^0}) \qquad \lambda_1 = \tan^{-1}\left(\frac{y^0}{x^0}\right) \qquad h_1 = r^0 - a, \tag{D.9}$$

and where x^0, y^0, z^0 are the components of $\mathbf{X}^0$, and $r^0 = \| \mathbf{X}^0 \|$.

The iteration scheme defined in (D.7) converges to the geodetic coordinates.