INTEGRATING AND REASONING ABOUT ONLINE SOURCES TO ACCURATELY GEOCODE ADDRESSES

by

Rahul Bakshi

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
Master of Science
(Computer Science)

August 2004

Dedication

To my parents and professors

Acknowledgements

I would like to sincerely thank my advisor, Craig A. Knoblock for all his guidance and support. I would also like to thank the members of my thesis committee Cyrus Shahabi and John P. Wilson for their comments on various issues in the thesis. This work couldn't have been done without the intellectual and social environment created by the members of the Info-Agents research group at the Information Sciences Institute: Snehal Thakkar, Ratapoom Tuchinda, Martin Michalowski and Salim Khan. I want to especially thank Snehal Thakkar for his help on various stages of the design of the algorithm and the implementation of the geocoder. I would also like to acknowledge the help of Shou-de Lin for the calculations in the Actual lot-size method.

This material is based upon work supported in part by the National Science Foundation under Award No. IIS-0324955, in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010, in part by the Air Force Office of Scientific Research under grant numbers F49620-01-1-0053 and FA9550-04-1-0105, in part by the United States Air Force under contract number F49620-02-C-0103, and in part by a gift from the Microsoft Corporation.

The U.S. Government is authorized to reproduce and distribute reports for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

Contents

DEDICATION II

ACKNOWLEDGEMENTS III

LIST OF TABLES..V

LIST OF FIGURES..VI

ABSTRACT VIII

INTRODUCTION 1

1.1 MOTIVATING PROBLEM..1
1.2 MY APPROACH..2
1.3 THESIS STATEMENT..3
1.4 CONTRIBUTION..3
1.5 THESIS OVERVIEW 4

GEOCODING: AN OVERVIEW 5

2.1 DATA SOURCES FOR GEOCODING 5
2.2 EXISTING METHOD FOR GEOCODING..7
2.3 DRAWBACKS OF EXISTING METHOD 9

APPROACHES TO ACCURATE GEOCODING 13

3.1 UNIFORM LOT SIZE METHOD..14
3.2 ACTUAL LOT SIZE METHOD 20

INTEGRATING SOURCES FOR GEOCODING..27

4.1 MODELING WEB SERVICES AS DATA SOURCES 29
4.2 DESCRIPTION OF THE DOMAIN MODEL..31
4.3 QUERYING COLUMBUS..36
4.4 ADDING NEW SOURCES..40

RESULTS..42

5.1 THE ADDRESS-RANGE METHOD..46
5.2 THE UNIFORM LOT-SIZE METHOD..48
5.3 THE ACTUAL LOT-SIZE METHOD..51
5.4 COMPARISON OF THE METHODS..53

RELATED WORK..57

CONCLUSIONS..59

7.1 CONTRIBUTION..59
7.2 LIMITATIONS AND FUTURE WORK 60

REFERENCE LIST 61

APPENDIX 63

List of Tables

TABLE 5.1 ERROR IN THE ADDRESS-RANGE METHOD OF GEOCODING......................47

TABLE 5.2 ERROR IN UNIFORM LOT-SIZE METHOD – FOR THE ENTIRE BLOCK 50

TABLE 5.3 ERROR IN ACTUAL LOT-SIZE METHOD – FOR ENTIRE BLOCK 52

TABLE 5.4: COMPARISON OF ERROR FROM ALL THE THREE METHODS...................56

TABLE A.1: COMPARISON OF ERROR..64

TABLE A.2: COMPARISON OF ERROR..66

TABLE A.3: COMPARISON OF ERROR..68

TABLE A.4: COMPARISON OF ERROR..70

TABLE A.5: COMPARISON OF ERROR..72

TABLE A.6: COMPARISON OF ERROR..74

TABLE A.7: COMPARISON OF ERROR..76

TABLE A.8: COMPARISON OF ERROR..78

TABLE A.9: COMPARISON OF ERROR..80

TABLE A.10: COMPARISON OF ERROR..82

TABLE A.11: COMPARISON OF ERROR..84

TABLE A.12: COMPARISON OF ERROR..86

TABLE A.13: COMPARISON OF ERROR..88

List of Figures

FIGURE 2.1 SAMPLE STREET DATA SOURCE FOR A GIVEN SEGMENT............................6

FIGURE 2.2 ALGORITHM FOR THE ADDRESS-RANGE GEOCODING METHOD.............8

FIGURE 2.3 A NON-EXISTING ADDRESS IS GEOCODED BY YAHOO.COM MAP SERVICE 10

FIGURE 2.4 645 SIERRA ST– BY MAPPOINT.COM: INACCURATE LOCATION 12

FIGURE 3.1 ALGORITHM FOR THE UNIFORM LOT-SIZE METHOD 15

FIGURE 3.2 TWO CORNER LOTS ON A STREET...17

FIGURE 3.4 THE CORNER LOT PROBLEM AND LOT SIZE PROBLEM.............................21

FIGURE 3.5 ALGORITHM FOR THE ACTUAL LOT-SIZE APPROACH TO GEOCODING 23

FIGURE 3.6 DIFFERENT ORIENTATIONS POSSIBLE FOR A LOT.......................................24

FIGURE 3.7 ACTUAL DIMENSIONS OF THE BLOCK...25

FIGURE 4.1 DOMAIN MODEL FOR COLUMBUS 33

FIGURE 4.2 SOURCE DESCRIPTIONS FOR COLUMBUS..35

FIGURE 4.3 VIEW DEFINITIONS FOR VARIOUS PROPERTY SOURCES 36

FIGURE 4.4 INVERTED SOURCE DESCRIPTIONS 38

FIGURE 4.5 DATALOG PROGRAM GENERATED BY THE MEDIATOR 39

FIGURE 5.1 REGION SELECTED FOR GEOCODING...43

FIGURE 5.2 CALCULATING ERROR USING DRIVING-DISTANCE TECHNIQUE 44

FIGURE 5.3 CALCULATIONS FOR SINNOTT'S FORMULA 45

FIGURE 5.4 RESULTS OF THE ADDRESS-RANGE METHOD 48

FIGURE 5.5 RESULTS OF THE UNIFORM LOT-SIZE METHOD 51

FIGURE 5.6 RESULTS OF THE ACTUAL LOT-SIZE METHOD...53

FIGURE 5.7 NORMAL DISTRIBUTION OF ERROR FROM ALL THE THREE METHODS 56

FIGURE A.1: MAP FOR AREA GEOCODED IN TABLE A.1...65

FIGURE A.2: MAP FOR AREA GEOCODED IN TABLE A.2...67

FIGURE A.3: MAP FOR AREA GEOCODED IN TABLE A.3...69

FIGURE A.4: MAP FOR AREA GEOCODED IN TABLE A.4...71

FIGURE A.5: MAP FOR AREA GEOCODED IN TABLE A.5...73

FIGURE A.6: MAP FOR AREA GEOCODED IN TABLE A.6...75

FIGURE A.7: MAP FOR AREA GEOCODED IN TABLE A.7...77

FIGURE A.8: MAP FOR AREA GEOCODED IN TABLE A.8...79

FIGURE A.9: MAP FOR AREA GEOCODED IN TABLE A.9...81

FIGURE A.10: MAP FOR AREA GEOCODED IN TABLE A.10 83

FIGURE A.11: MAP FOR AREA GEOCODED IN TABLE A.11 85

FIGURE A.12: MAP FOR AREA GEOCODED IN TABLE A.12 87

FIGURE A.13: MAP FOR AREA GEOCODED IN TABLE A.13 89

Abstract

Many Geographic Information System (GIS) applications require the conversion of an address to its geographic coordinates. This process is called geocoding. The existing methods rely on a technique which queries the data from a street data source. This data source typically provides the coordinates of the end points of the street segment and a range of addresses between those points. The existing methods approximate the location of the address based on this information. Some of these addresses may not even be present on the street. This provides inaccurate results since the approximation does not consider the actual number of addresses and is based on an estimate of possible addresses existing on the street segment. I propose two new methods for geocoding. The first method is called the *Uniform lot-size method,* which uses the number of addresses/lots present on the street segment to approximate the location of an address. The second method is called the *Actual lot-size* method which takes into consideration the lot sizes on the street segment and orientation of the lots as well. These methods gather information about the actual number of lots and sizes of the lots on the streets from property tax web sites. The two new methods yield much more accurate results compared to the existing techniques. In the region chosen as the test-set, the traditional geocoding approach had an average error of 36.85 meters. The *Uniform lot-size method* reduced this error to 7.87 meters, an improvement of 79% while the *Actual lot-size method* reduced the error to 1.62 meters, an improvement of 96%.

Chapter 1

Introduction

As we move to the next generation of the Internet, the World Wide Web is becoming a data source that can be queried. The challenge lies in using these data sources to solve many of the existing problems. One such challenge is to geocode addresses more accurately. Geocoding is the process of obtaining the geographic coordinates (latitude/longitude) of a given address. The software which does this is called a geocoder. The existing approaches to geocoding provide values which have a significant error in them as most of the time they assume an incorrect number of lots are present on a street. This error in the values can be appreciably reduced if property related information is integrated with existing techniques. This dissertation is an interesting example on how information integration techniques can be used to build a geocoder that has a much higher accuracy compared to existing geocoder.[1]

1.1 Motivating Problem

There is considerable inaccuracy in the existing services on the Internet that locate a map of a particular area or provide the latitude and longitude information of a given address. They approximate their results based on the data from some street data

[1] Some typical examples of the existing geocoding services are the Federal Financial Institutions Examination Council's http://www.ffiec.gov/geocode/default.htm, http://www.geocode.com/, http://www.geographic.com/ezgeo/

source like NavTech,[2] TIGER/Line (TIGER/Lines 2000), etc. Their approach to approximation yields inaccurate results. The information is not only inaccurate, but sometimes misleading. For example, these services provide the map or give the latitude/longitude information for an address which does not exist in reality. The error in the coordinate values provided by these services is significant compared to the actual latitude/longitude values (ground-truth).

It is important to have accurate values for the geographic coordinates for some applications. For example, accurate geocoding is essential for urban rescue and recovery operations. Accurate geocoding is important for a variety of applications, such as environmental health studies to demarcate areas with potential hazardous exposure in relation to where people live (Cayo and Talbot 2003). According to the journal article by the US Federal Geographic Data Committee (FGDC), the geographic location is a key feature of 80-90% of all government data (Committee 2003). It is important for other applications such as aligning vector data with imagery (Chen, Knoblock et al. 2003). Therefore it is important to have geocoding methods that provide results with maximum accuracy.

1.2 My Approach

My approach to geocoding with higher accuracy focuses around getting accurate property-related information for and around the address to be geocoded. Sources like US Postal Services web site and property tax websites are good candidates to get

[2] http://www.navteq.com/

property related data. I exploit some of these sources to get the property related information. I use a mediator-based architecture to query the sources on the web, and get the number of addresses around the location to be geocoded. Based on the availability of these sources and the details they provide about the addresses, I use new algorithms which give more accurate results for geocoded values compared to the existing geocoding methods. I describe my approach in detail in this dissertation.

1.3 Thesis Statement

In this dissertation, I propose a novel way to exploit online sources to build a geocoder with higher accuracy compared to current methods. The following is my thesis statement:

> **The accuracy of the geocoded values of a location can be significantly improved by exploiting online property-related data.**

1.4 Contribution

With this Thesis, I make the following contributions:

- I develop novel algorithms to exploit online data to perform geocoding with higher accuracy than existing methods

- I apply data integration techniques to organize and integrate a large number of online sources for geocoding, in an extensible framework.

1.5 Thesis Overview

The remainder of the dissertation is organized as follows:

Chapter 2 provides an overview of the existing techniques to estimate the latitude and longitude of a given location. Chapter 3 introduces two new techniques for geocoding and explains them in detail. Chapter 4 explains the information mediator and its role in successfully realizing the new geocoder. Chapter 5 presents the results of my approach compared to the traditional approach. Chapter 6 gives details about related work. Chapter 7 summarizes the dissertation and gives information about future work possible in this field.

Chapter 2

Geocoding: An Overview

The goal of this research work is to improve the accuracy of geocoding. This chapter describes the existing approach to geocoding and explains its limitations. Section 2.1 discusses the data sources that are typically used for geocoding. Section 2.2 discusses the existing approach in detail. Section 2.3 gives an analysis of the existing techniques and shows the errors produced by these techniques.

2.1 Data Sources for geocoding

The main sources of data that the existing services use are commercially available products like the TIGER/Line data from the Bureau of Census,[3] NavTech data from Navigation Technologies,[4] and the GDT data from the Geographic Data Technology (GDT).[5] These sources have street information for the entire United States.

The data sources provide the geographic coordinates (latitude and longitude) of the street intersections. They also provide the possible address ranges on each side of the street between the two sets of coordinates for the street segment. These data sources give a very good estimate, but do not give information about the exact

[3] http://www.census.gov/geo/www/maps/
[4] http://www.navteq.com/
[5] http://www.geographic.com/

number of addresses present on the street segment. For example, if an address "625 Sierra St., El Segundo, 90245" is queried in the TIGER/Line data source, it returns a tuple which has the end-points of the street segment on which the address is located and the possible addresses. For this address, the range on the left side of the street is 601 – 699 and on the right side of the street is 600 – 698.[6] Figure 2.1 shows some of the sample values associated with a street segment in a typical street data source.

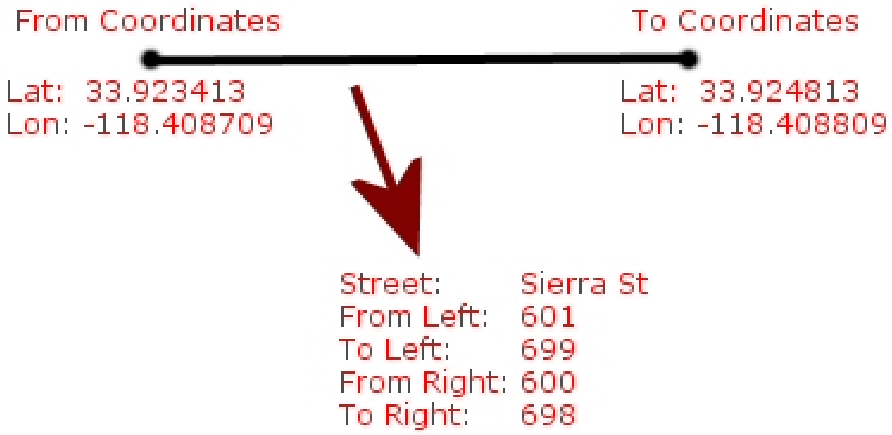


Figure 2.1 Sample street data source for a given segment

This information is not inaccurate, but it does not give us any detail about how many of those addresses actually exist. The other data sources like Navtech and GDT provide similar information as the TIGER/Line data source. The accuracy of data in each of these sources is different, with NavTech being more accurate than TIGER/Line source (TIGER/Lines 2000). However, the base problem remains the same

[6] The left and right are the directions taken in the sense when one travels from the 'from' coordinates to the 'to' coordinates in the TIGER/Line data source.

– there is no information about the actual addresses present in the specified address range. Further, there is no information about the size and location of each address/lot present on the street.

2.2 Existing method for geocoding

The problem described in Section 2.1 leads the existing services to geocode the address location based on the end points of the street segment and the address range provided in these data sources.

Figure 2.2 describes the algorithm for the traditional a*ddress range* geocoding method. As a first step, the algorithm parses the given address into individual tokens representing the street address, street name, city, state and zip. Based on these parameters, at the second step, the algorithm queries the street data source being used. The data source returns the coordinates of the end points of the street segment in which the current address is located. It also gives the address ranges present on both the sides of the street. The third step selects the appropriate side of the street depending on the current street addresses and associating it with the side of street which has the corresponding even or odd addresses. This is accomplished by the Modulo 2 (Mod2) operation. The Mod2 returns 0 if its operand is even, otherwise it returns 1. The result of Mod2 operation on the address-range of the left side of the street is compared with the result of Mod2 operation of the current address, and the side on which the current address belongs is decided.

```
Step 1:  currentaddress ← parse the given address to get street address

Step 2: Query street data source:
                 fromlatitude, fromlongitude, tolatitude, tolongitude ←
                                                            coordinates of end points
                 fromaddrleft, toaddrleft, fromaddrright, toaddrright ←
                                          address ranges on either side of the street

Step 3:  street_side ← fromaddrleft % 2

Step 4: If street_side == 0
                 toaddress ← toaddrleft
                 fromaddress ← fromaddrleft
         Else
                 toaddress ← toaddrright
                 fromaddress ← fromaddrright

Step 5:  rel_loc ← ABS((toaddress - currentaddress)/(toaddress - fromaddress))

Step 6:  Calculate the latitude and longitude based on the ratio

            currentlatitude ← tolatitude - (rel_loc * (tolatitude - fromlatitde))
            currentlongitude ← tolongitude - (rel_loc * (tolongitude - fromlongitude))
```

Figure 2.2 Algorithm for the Address-range geocoding method

Once the side to which the current address exists is decided, a ratio is taken (Step 5) which gives the relative location of the current address (address to be geocoded) on the street segment. This ratio is based on the number of addresses/lots that may be present on the street. For example, if the street data source returns addresses 601 – 699 present on the left side which is also the side where the current address exists, this method would assume that 50 addresses are present on the left side of the street. It then calculates the relative location of the current address in the range of 50 addresses. The relative location calculated is then interpolated between the street end points to get the geographic coordinates of the current address (Step6). Some geocoding techniques include offsets as well. However for all the methods that will be described in this dissertation, I do not offset the locations and the geocoded locations are on the center of street.

2.3 Drawbacks of existing method

The *Address-range* method described in section 2.2 has some drawbacks. First, it assumes that all the lots/addresses specified by the street data source in the address range actually exist. Second, it assumes that all these lots are of equal size. And lastly, it does not take into account the dimension occupied by the corner lots which actually may be a part of the other intersecting street segments.

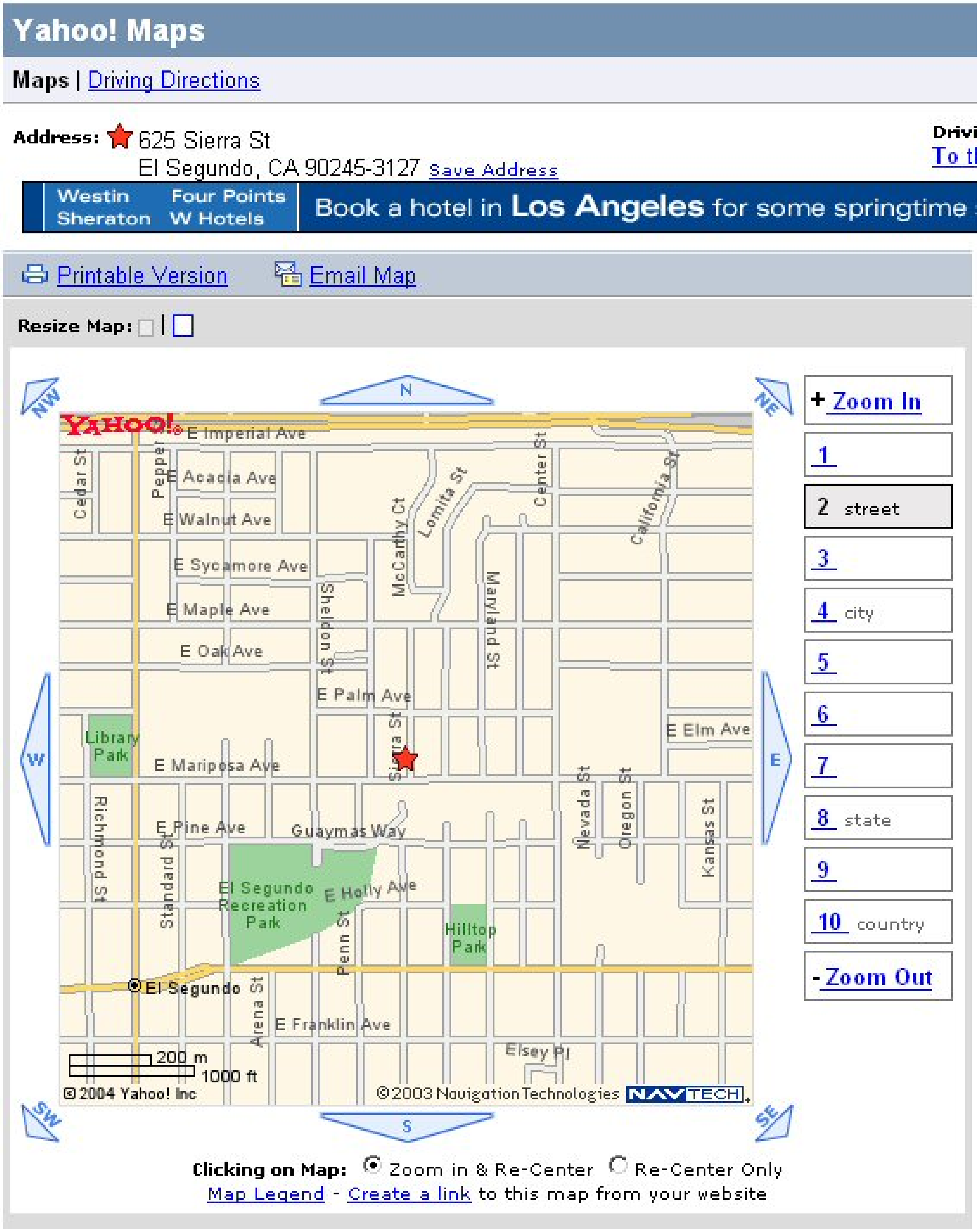
Consider the example of finding the location of a nonexistent address in Los Angeles County: "625 Sierra St, El Segundo, CA, 90245." I used this address to query a number of popular mapping services on the Internet. All of these services returned a location for this nonexistent address. Figure 2.3 show the result of Yahoo! Map Service[7]. The other popular services like Geocode[8], MapQuest[9] and MapPoint[10] also geocoded this address. Thus the present method can be misleading at times, as in this case when it gives the location of a non existent address.

[7] http://maps.yahoo.com

[8] http://www.geocode.com

[9] http://www.mapquest.com

[10] http://www.mappoint.com



Yahoo! Maps

Maps | Driving Directions

Address: 625 Sierra St
El Segundo, CA 90245-3127 Save Address

Drivi
To tl

Westin Four Points
Sheraton W Hotels
Book a hotel in Los Angeles for some springtime

Printable Version Email Map

Resize Map: □ | □

Clicking on Map: ● Zoom in & Re-Center ○ Re-Center Only

Map Legend - Create a link to this map from your website

Figure 2.3 A non-existing address is geocoded by Yahoo.com Map Service

Consider another example. The address 645 Sierra St, El Segundo, CA – 90245 is present on the intersection of Sierra St. and E. Palm Ave. Figure 2.4 shows the results generated by MapPoint[11] for this address. This lot is the last lot (towards the North) on that segment of that street. However, MapPoint shows this address on the middle of the street when queried. The apparent reason is that the data source which they use returns a result which has addresses 601 to 699 present on the side of the street where 645 Sierra St is located. This range implies that there are 50 lots present on the selected side of the street. In reality, there are 7 lots present on this street segment. So when the interpolation is done by taking 50 addresses, it leads to results with a large error.

Theses observations validate the claim that the existing services for geocoding do not check for validity of addresses and approximate the given address based on the information about the end-point of the street and an approximation of the address range present on the street. This also implies that the existing services do not consider the size of the lots on the street. This is the motivating problem for this dissertation.

[11] http://www.MapPoint.com/

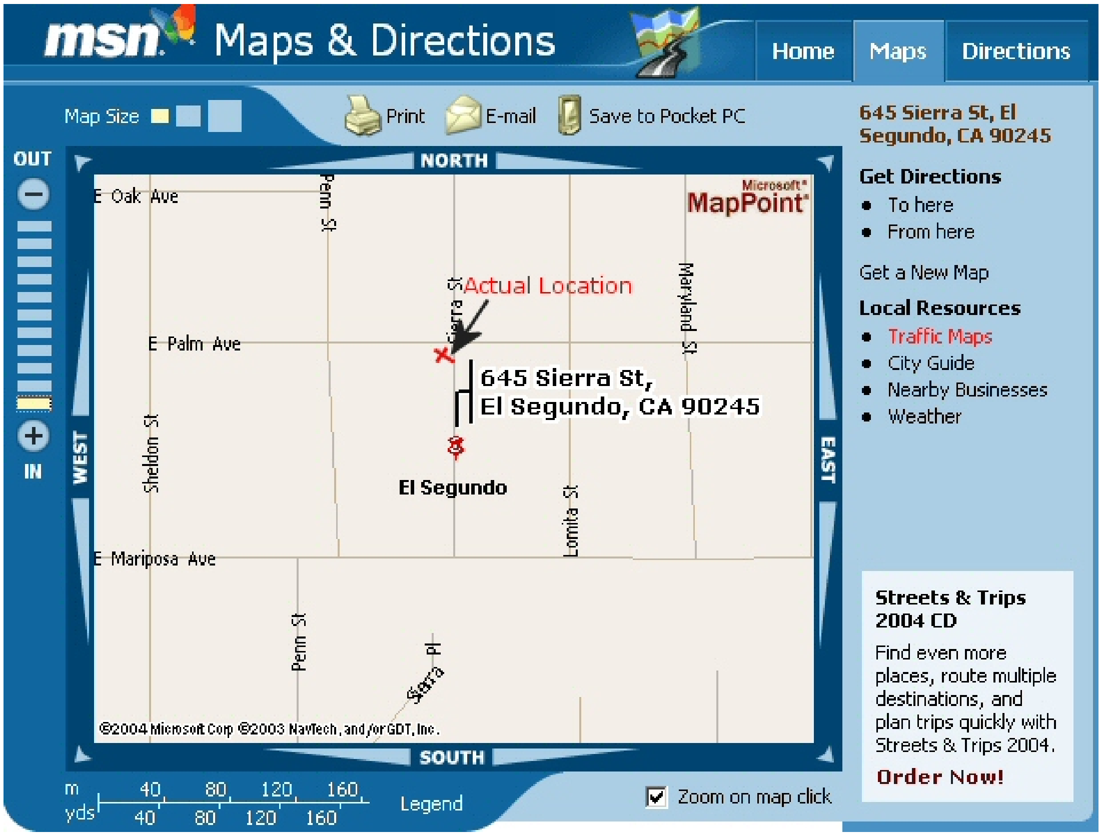


Figure 2.4 645 Sierra St– by Mappoint.com: inaccurate location

In the next section I address these issues and present my approach to geocode addresses. The goal of this dissertation is to overcome this problem and combine property related information with the existing information sources to yield better results.

Chapter 3

Approaches to accurate geocoding

More accurate geocoding can be performed by utilizing the number of properties on a given street and their dimensions. My approach for increasing the accuracy takes into account these facts and shows a remarkable improvement in the geocoded values. I call the new geocoder Columbus.[12] This chapter discusses the methodology and approach to accurate Geocoding. Section 3.1 describes the *Uniform lot-size* approach which takes into account the number of lots on the street. Section 3.2 describes the *Actual lot-size* approach which also takes into account the lot dimensions and orientations in addition to the number of lots on the street.

The main reason why the address range method provided results with significant error is because it infers the number of houses/lots present on the street segment from the street address range. It is seldom the case that all the address in this range actually exists on the street. If there was any method where-by the number of lots on the given segment are known, it would show significant improvement in the geocoded values. Further if the orientation and sizes of the lots on the corner of the street are known, it would result in further improvement in accuracy.

[12] The geocoder is named Columbus after the famous traveler Christopher Columbus.

3.1 Uniform lot size method

The idea behind the Uniform lot-size method is to use the actual number of houses/lots on a street segment. There are two main sources on the Internet which give property related information. They are the US Postal Service website[13] and the property tax[14] websites for different regions. While from the US Postal Service website, it is possible to extract the number of properties between an address-range, some of the property tax sites have information about the size of the lots as well.

The Uniform lot-size approach requires the number of houses/lots on a given street segment. I use the property tax websites as my source for finding the number of lots on a street segment. This is due to the fact that some of the property tax websites provide the lot dimensions, which becomes useful in the next approach to accurate geocoding (discussed in Section 3.2). These resources have the most accurate property data with regard to the size of the lots and number of lots on every street. Hence, they are an excellent data source to use in conjunction with the existing street data sources to improve accuracy. Most of these property tax websites have data about the number of lots for a given address range and some of them even provide the dimensions of the properties.[15]

[13] http://www.usps.com/

[14] A list of property tax websites can be found at http://indorgs.virginia.edu/portico/personalproperty.html

[15] The property tax website for Lubbock County in Texas (http://www.lubbockcad.org/) gives the lot dimensions for the property parcels.

The Uniform lot-size method queries the number of houses from a property tax data source. Based on this result, it interpolates the latitude and longitude of the lots on the street. This method assumes that all the lots on the street are of equal size and hence the name "Uniform lot-size" method.

Figure 3.1 gives the algorithm for the uniform lot-size method. As the first step, the address to be geocoded (*currentaddress*) is separated into street address, street name, city, state and zip. The underlying data source which has the street information is then queried at the next step (Step 2). It gives the street segment to which the current address belongs and also the address ranges present on either side of the street. Based on this information, the side on which the given address belongs is calculated at step 4.

```
Step 1:  currentaddress ← parse the given address to get street address
Step 2:  Query street data source:
                 fromlatitude, fromlongitude, tolatitude, tolongitude ←
                                                  coordinates of end points
                 fromaddrleft, toaddrleft, fromaddrright, toaddrright ←
                                                  address ranges on either side of the street
Step 3: street_side ← fromaddrleft % 2
Step 4: If street_side == 0
                 toaddress ← toaddrleft
                 fromaddress ← fromaddrleft
        Else
                 toaddress ← toaddrright
                 fromaddress ← fromaddrright
Step  5: Query the property tax data source for the selected side:
                 nb ← number of lots between fromaddress and currentaddress
                 na ← number of lots between currentaddress and toaddress
Step  6: Calculate the length of the street segment obtained in step 2 using the distance formula
                 street_len ← SQRT((fromlatitude - tolatitude)^2 + (fromlongitude - tolongitude)^2)
Step  7: Assume uniform size for all lots and divide the value of 'street_len' Obtained
         in Step 5 by the number of lots present on the street + 1:The additional lot
         is added to account for the corner lot that may be on an intersecting street
                 lotsize ← street_len/(nb + 1 + na + 1)
Step  8: Divide the lot size obtained in Step 6 by two, to get the increment factor 'offset'
                 offset ← lotsize/2
Step  9: Calculate the slope θ (theta) for the street segment
                 θ ← Tan^-1 ((tolongitude - fromlongitude)/(tolatitude - fromlatatitude))
Step  10:Calculate the latitude of the currentaddress
                 currentlatitude ← fromlatitude + (offset + nb * lotsize + offset) * Cos (θ)
                 currentlongitude ← fromlongitude + (offset + nb * lotsize + offset) * Sin(θ)
```

Figure 3.1 Algorithm for the Uniform lot-size method

At the fifth step, we query the property tax data source to get the number of houses before (*nb*) and after (*na*) the current address on the street segment. The sixth step calculates the length of the street segment (*street_len*) using the Euclidian distance formula. This formula is valid for planar surfaces. Since the segments on the street data source are very small compared to the size of the Earth, I can use this formula without significantly affecting the accuracy of my results. At the next step (step 7), I calculate the size of each lot.

Here I come across a challenge of deciding the orientation of the lots on the corners of the street segment. It is not known to which street segment the corner lots belong from the property tax source. For example, consider Sierra St. in Figure 3.2. It cannot be determined from the property tax data source whether lot 12 belongs to Sierra St. or E. Palm Ave. Similarly, lot 19 could belong to Sierra St. and E. Mariposa Ave. I refer to this as the 'c*orner-lot problem*'. For the *Uniform lot-size* method, I generalize and assume that for 2 corner lots on a street, one of them belongs to the street and the other one belongs to the other intersecting street which would hold on average, given four streets and four corners. Thus if there are 'n' lots/addresses existing on one side of a street, there are actually 'n+1' dimensions present on the street where the one extra dimension is a corner lot which is actually part of another street.

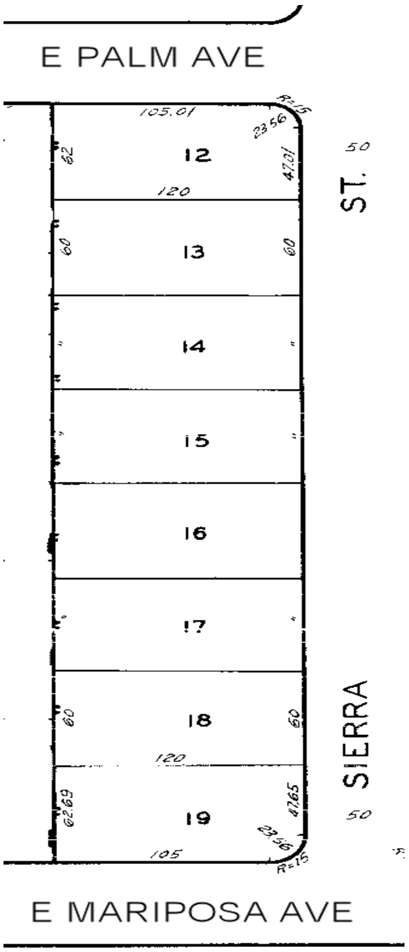


Figure 3.2 Two corner lots on a street

The value of '*street_len*' obtained is now divided by the number of lot dimensions present on the street. As explained above, if there are 'n' lots/addresses on the street, this method assumes that there are n+1 lot-dimensions present on the street, the extra dimension being of the lot which is a part on the intersecting street. This is achieved in step seven of the algorithm shown. The '*street_len*' is divided by the sum of number of addresses before the current address on the street segment, the number of address after the current address on the street segment, the current address and one extra lot which is a part of the intersecting street segment. This method also assumes that all the lots are equal in size. Since it is not known on which corner of the street is the lot of the other street, I begin calculating the first lot to be located at an offset factor, which is half the average lot size on the street. This factor is referred to as '*offset*'. The eighth step in the algorithm calculates the value of *offset*. Figure 3.3 shows the calculations used to determine the latitude and longitude of the lots present on the street with this method. The slope of the street segment is calculated and the angle theta (θ) is calculated by the formula given in step 8 of the algorithm. Once the angle is known, the latitude and longitude are calculated. sin(θ) gives the projection of the longitude and cos(θ) gives the projection for the latitude. The equations to calculate the latitude and longitude are described in step 10 of the algorithm. I add another *offset* value so that we get to the center of the lot.

Columbus is realized with a set of web-services that perform atomic and independent operations. There are three web services that are used for the Uniform lot-size method for getting latitude and longitude – *Streets*, *PropertyTax* and *UniformLotSizeApproximation*.

The *Street* service takes the street address, city, state and zip as input and queries the TIGER/Line data source. It gives the following as output:

1. The latitude and longitude of the end points of a street
2. The street name and street type
3. The zip codes on each side of the street
4. The address ranges on the left and right side of the street

The *PropertyTax* web service takes the street address, city, state, zip and the address ranges on both the sides of the street. It gives an output which has the number of lots before and after the given lot.

The *UniformLotSizeApproximation* web service approximates the location of the current address based on the number of houses present on the street and the coordinates of the end points of the street segment.

In the algorithm just described, the address to be geocoded is taken as the input. Then the *Street* web service is queried to get the geographic coordinates of the end points and the address ranges. It then decides on which side of the street the lot is located. The *PropertyTax* web service is used to get the number of lots before and after the current lot. The *UniformLotApproximation* web service then calculates the latitude and longitude based on this data

For example, consider the street address 623 Sierra St, El Segundo CA, 90245. A query to the TIGER/Line data source returns the address range of 601 – 699 on the left side and 600-698 on the right side. If these values are taken as the basis of calculation, it would imply that 50 lots are present on each side of the street. A query on the *PropertyTax* web service tells us that only 7 lots present on each side of the street. Thus interpolating on the basis of 50 streets will definitely give a very inaccurate result compared to only 7 lots.

However, the problem of deciding which street segment the corner house belongs to still remains. There are some number of lots on a street, '*n*'. However there may be up to two more lots on each of the corners that also occupy space on the present street, but are a part of the other intersecting street. E.g.: Figure 3.4 shows a block formed by the intersection of 4 streets: *Mariposa Ave.*, *Palm Ave.*, *Sierra St.* and *Penn St*. However, the figure does not indicate if Lot 12 and Lot 19 belong to *Sierra St*.

Thus in this case, the geocoding by Uniform lot-size method first gets the number of lots present on the street, which is 7. It then makes the assumption that there is another lot on the street which is a part on an intersecting street. Thus the lot size is calculated by dividing the street length into 8 equal parts and the geographic coordinates are calculated as per the algorithm described above. The results in Chapter 5 show that this method provides better results over the traditional address range approach.

3.2 Actual Lot Size Method

Although the corner lot problem is dealt with in the Uniform lot-size method, it is again only an estimate and not the most accurate method of solving the problem. The method does not take into account the orientation of the corner lots. Figure 3.4 describes a block formed by 4 streets: *Sierra St*, *E Mariposa Ave*, *Penn St* and *E*

Palm Ave. Just looking at the figure, one cannot determine to which streets the corner lots belong.

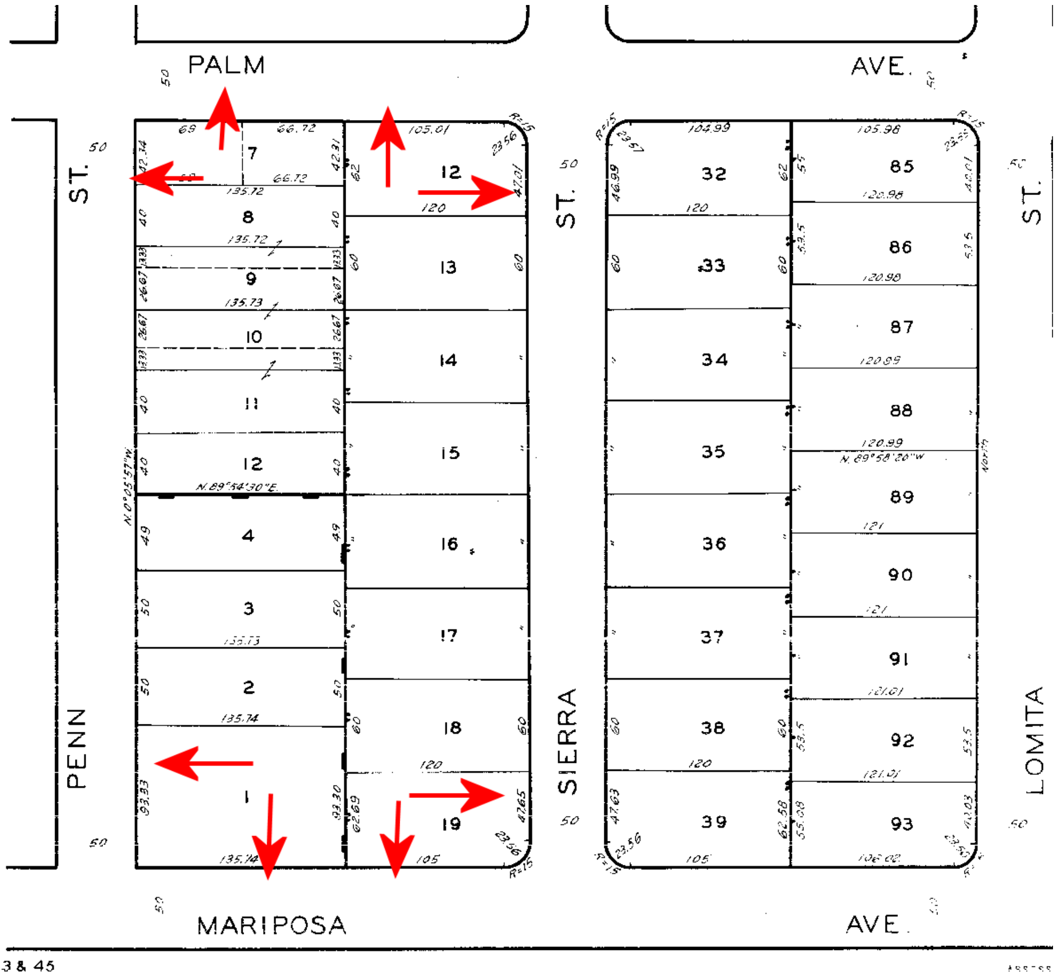


Figure 3.4 The Corner lot problem and lot size problem

Thus one cannot conclude if Lot 19 belongs to Mariposa Ave. or Sierra St. directly from the data in the property tax source. Also the Uniform lot-size method assumes that all the lots on a given street have the same dimensions. Consider lots 1 and 2 in Figure 3.4. Both these lots belong to Penn St. However, the size of lot 1 is almost twice the size of lot 2. If the information about the orientation of the corner lots and the size of each of the lots is combined with the Uniform lot-size approach, intuitively it will give a much better result compared to the traditional Address-range method or the new Uniform lot-size approach. This is the motivation for the next method of Geocoding called the Actual lot-size method, which takes into account the orientation of the corner lots and size of the lots.

This geocoding approach solves the problem of lots on the same street having different sizes. It also determines the orientation of the corner lots on the street. Given the address, this method calculates the street and the block to which the lot belongs. This approach relies on the availability of an online source which has information on the sizes of the addresses/lots.

Figure 3.5 gives the algorithm for the Actual lot-size method. Similar to the other two methods described earlier, the initial steps of this method separate the address into individual tokens representing the street address, city, state and zip. Then, the street segment information to which the current address belongs is obtained. Step 4 decides on which side of the street the address is located.

The fifth step gets the coordinates of the end points of the other streets that form the block. After obtaining the coordinates of all the four corners of the block, at step 6 the algorithm determines if the block is rectangular. If it indeed is rectangular, the algorithm proceeds to the next step, otherwise it reverts to Uniform lot-size geocoding method. The next step queries the property tax source and gets the dimensions of all the lots on the block. Step 8 calculates the actual lengths of street segments that form the block. We use the great circle distance formula to calculate the length.

```
Step  1: currentaddress ← parse the given address to get street address
Step  2: Query street data source:
             fromlatitude, fromlongitude, tolatitude, tolongitude ←
                                                 coordinates of end points
             fromaddrleft, toaddrleft, fromaddrright, toaddrright ←
                                                 address ranges on either side of the street
Step  3: street_side ← fromaddrleft % 2
Step  4: If street_side == 0
             toaddress ← toaddrleft
             fromaddress ← fromaddrleft
         Else
             toaddress ← toaddrright
             fromaddress ← fromaddrright
Step  5: Query street data source:
             fromlatitudeP, fromlongitudeP, tolatitudeP, tolongitudeP ←
                                 end points of the street segments that form a block
Step  6: If block not rectangular, perform Uniform lot-size geocoding
Step  7: Query the property tax data source and get the dimensions of  each of the lots
         present on the block
Step  8: Calculate the actual dimensions of the streets in the block based on the data from the
         source used in Step 2 and Step 4 using the Great Circle Distance Formula:

         EarthRadius = 6378137.0
         street_len  ← EarthRadius * (Cos$^{-1}$(Sin(tolatitude) * Sin(fromlatitude) + Cos(tolatitude)
                                 * Cos(fromlatitude) * Cos(tolongitude - fromlongitude)))
Step  9: There are 2 possible assignments for each conrer lot and there are 4 corner lots. So,
         there are 16 possible combinations of assignments of corner lots in a given rectangular
         block.
             orientations[1..16] //array with all 16 possible orientations
             error[1..16] //error in street length for each orientation
             For i ← 1 to 16 do:  //for all 16 orientations
                     estimated_len = Σ length of all lots on the street in orientations[i]  +
                                      Σ depth of corner lots (if present in orientation[i])
                     For k ← 1 to 4
                             errorstreet[k] = ABS(street_len of street[k] -
                                                          estimated_len of street[k])

                     error[i] ← Σ errorstreet[1..4]
Step 10: Select the orientation with minimum error in step 9
             j = indexOf(min(error), error) //find element in error with minimum error
Step 11: Based on the assignement selected, obtain the center point of the lot to be geocoded
             relXcoord, relYcoord ← orientation[j]
Step 12:Convert the relative position in Step 11 to absolute latitude and longitude
             latitude  = toplat - ((relYcoord)*(toplat - bottomlat)/(relBlocklen))
             longitude = leftlon + ((relXcoord)*(rightlon - leftlon)/(relBlockwid))
```

Figure 3.5 Algorithm for the Actual lot-size approach to geocoding

Once the number of houses/lots on each street segment and their dimensions are known, the challenge is to decide on which street segment the corner lots belong. For a rectangular block, there are four corner lots and each of these could belong to either of the two streets which intersect on the corner. This leads to sixteen possible combinations for the orientation of the corner lots for the given block. The corner lots can belong to any of the street segments giving rise to 16 different combinations as shown in Figure 3.6. The different orientations of the corner lots would change the size of the streets on the block.

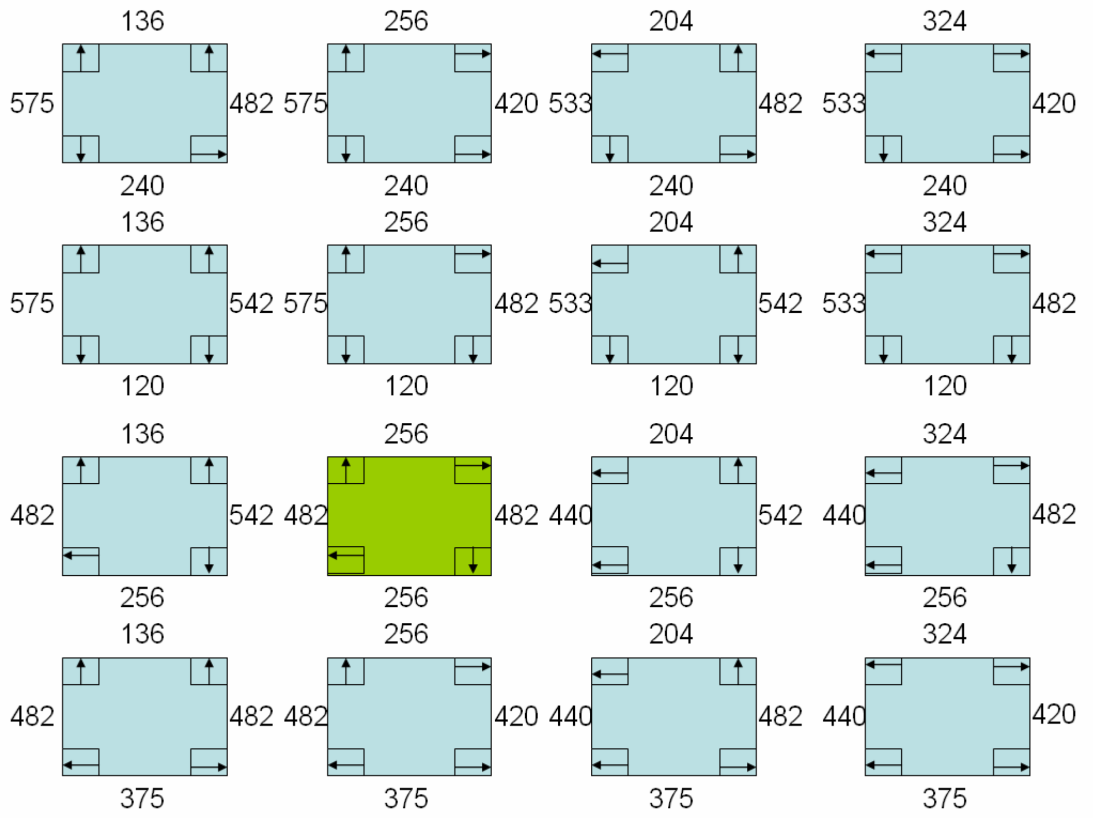


Figure 3.6 Different orientations possible for a lot

In step nine, an error value is calculated, which is the difference between the sum of the actual lengths of the street segments and the calculated length of the street for a particular orientation. This error is calculated for all possible sixteen orientations for the block. The orientation which gives the least error value (which most closely matches the actual dimension of the block) is selected as the one for the current block. Figure 3.7 shows the actual dimensions of the block computed from the end points of the street data. These are compared with each of the sixteen combinations shown in Figure 3.6. At this step of the algorithm, the true dimensions of the block are compared with all the sixteen possible orientations and the selection with minimum error is chosen as the orientation of the current block. Thus at the end of step ten, the exact layout of the block and the orientations of all the four corner lots for the block are known.

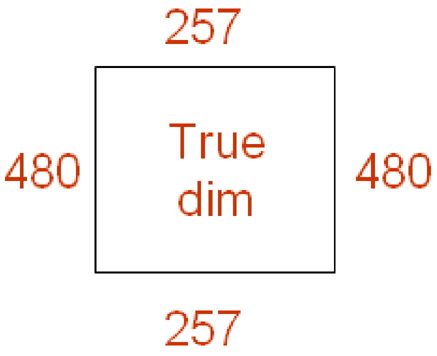


Figure 3.7 Actual dimensions of the block

Once the layout of the block is known, we obtain the center point for the lot to be geocoded in terms of relative coordinates for the block. The relative coordinates are with respect to the top left corner of the block being the origin (0,0). These relative coordinates are converted into latitude and longitude values by a simple mapping function. Step twelve shows a sample mapping function which assumes that the latitude of the block increases as we move from south to north and the longitude increases as we move from west to east. A trivial change is needed for blocks which do not have this type of layout. Thus we obtain the latitude and longitude for the lot.

The results presented in Chapter 5 show that this method gives us even better results than any of the previously discussed methods.

Chapter 4

Integrating Sources for Geocoding

In section 3, I described two algorithms that perform geocoding with higher accuracy than the traditional Address-range approach. To explain the algorithms I assumed that a single source exists for getting property data. However, that is not actually the case. There are over two thousand property tax assessment districts in the US and each of these districts has their own property tax web site.[16] The property tax data may be organized by state, county, city or some other geographic region. For example, the property tax information from the all the properties in the state of New York can be found at the USPDR[17] web site, while the property information for Los Angeles County[18] is on a separate website. There is no single web source to obtain the property tax information for the state of California. The challenge is to determine the appropriate source to query the property information for geocoding a given address. Each source may have a different set of attribute names and different structure in which the data is represented. The same is true for the street data. The data may be organized by state, county or other geographic region. The challenge is the need to generate one unified property tax service for geocoding.

[16] A list of property tax sites can be found at http://indorgs.virginia.edu/portico/personalproperty.html

[17] http://www.uspdr.com/

[18] http://www.lacountyassessor.com/extranet/default.asp

Integration systems such as Information Manifold (Levy, Rajaraman et al. 1996), InfoMaster (Genesereth, Keller et al. 1997), InfoSleuth (Bayardo Jr., Bohrer et al. 1997) and Ariadne (Knoblock, Minton et al. 2001) solve this exact problem. That is, they provide a uniform query interface to various data sources. In Columbus, I use the Prometheus mediator (Thakkar, Ambite et al. 2003) to access different property tax data sources as well as different street data sources as if they were in one database.

The geocoding in Columbus is realized with a set of web services described as data sources in the mediator domain model. If the data is in the form of web pages such as the Los Angeles Property Tax web site,[19] then I use the Fetch Agent Builder[20] to convert it into an XML web service. Section 4.1 of this chapter describes how web services can be modeled as data sources in the mediator. Section 4.2 describes how I define the domain model for the three geocoding methods and various sources. Section 4.3 provides details of how a query to geocode an address is processed in Columbus. Section 4.4 gives details on how new sources can be added to the geocoder.

[19] http://www.lacountyassessor.com/extranet/default.asp

[20] http://www.fetch.com/

4.1 Modeling web services as data sources

Columbus consists of a set of web services which are used to query the data needed for geocoding. Depending on the given addresses and available data sources, a different set of web services will need to be executed every time an address is geocoded. For example, if the address is in Los Angeles County, and I am using the Uniform lot-size method or the Actual lot-size method, then the web service for the Los Angeles County property tax data would be required whereas if the address was located in New York, a different web service would be queried. A mediator system such as Prometheus can offer a unified interface to these different sources.

The Prometheus mediator is a data integration system that builds on previous work on data integration (Garcia-Molina, Hammer et al. 1995; Genesereth, Keller et al. 1997; Knoblock, Minton et al. 1998; Levy 2000; Knoblock, Minton et al. 2001; Lenzerini 2002). Traditionally, data integration systems have a set of domain relations on which the users can specify queries. The task of the data integration system is to translate the user's query into a set of queries on the source relations.

In order to support a unified interface, the mediator needs all the web services modeled as data sources. The available data sources for Columbus are a set of property tax web services generated from various assessors' web pages, a set of street information web services such as, the *Tigerlines* street information web service, and a set of services to approximate location of the given address on the given street segment. Each web service is modeled as a source relation with binding restrictions. That is, in order to obtain information from the source relation, the user must provide values of all attributes with binding restrictions. The input attributes of the web services are modeled as attributes in the corresponding source relations with binding restrictions. For example, *Tigerlines* service that accepts the *streetaddress*, *city*, *state*, and *zip* attributes and returns *streetname*, *streettype*, *frlat*, *frlon*, *tolat*, *tolon*, *zipl*, *zipr*, *fraddr*, *fraddl*, *toaddr*, *toaddl* attributes is modeled as the following source relation. The '$' symbol before an attribute denotes attribute with a binding restriction.

LAProperty($sa, $ci, $st, $zi, frlat, frlon, tolat, tolon, fename,
fetype, zipl, zipr, fraddr, fraddl, toaddr, toaddl)

Once I have modeled all available web services as source relations, I need to determine a set of domain relations for Columbus. I define *PropertyTax* and *Street* domain relations in Columbus as virtual relations representing all available property tax and street information web services respectively. The three different methods to geocode given addresses are modeled as the following three domain relations that user's can query: (1) *AddressRangeGeocoder*, (2) *UniformLotSizeGeocoder*, and (3) *ActualLotSizeGeocoder*.

Now that I have modeled all available web services as data sources and determined domain relations, I need to define a set of rules to relate the source relations with the domain relations.

4.2 Description of the Domain Model

An integral part of any data integration system is a set of rules that relate the domain relations to available source relations. Traditionally, data integration systems have utilized three approaches to relate domain relations to available source relations. In a Global-As-View (GAV) approach, a domain expert defines the domain relations as views over the available source relations. In the Local-As-View (LAV) approach, available source relations are defined as views over the domain relations. In GAV model query reformulation is trivial. However, adding additional data sources in GAV model may require modifying definitions of all domain relations. In Local-As-View one only needs to add the view definition for the new source to add additional sources. Duschka (Duschka 1997) and Levy et.al. (Levy, Rajaraman et al. 1996) have described algorithms to translate user queries into a set of source queries using the LAV approach. More recently, there has been another approach termed GLAV (Lenzerini 2002) that allows user to combine both GAV and LAV approaches. The Prometheus mediator supports all three approaches. I utilize the flexibility provided by the GLAV approach by defining some source relations as views over domain relations, i.e. using the Local-As-View approach and other domain relations as views over source relations, i.e. using the Global-As-View approach.

As shown in Figure 4.1, the three domain predicates representing different geocoding methods are defined as views on the available source relations or other domain relations. For example, the *UniformLotSizeGeocoder* domain relation is defined as a join over *Street* and *PropertyTax* domain relations and *UniformLotSizeApproximation* source relation. *ActualLotSizeGeocoder* and *AddressRangeGeocoder* implement the actual lot size approach and the address range approach for geocoding respectively. The key advantages of utilizing the GLAV approach are to enable easy addition of new property tax and street information web services and avoid the complexity of defining LAV rules to model geocoding algorithms.

The rule R1 is for the *Address-range* method. The predicate *Street* maps to the service required to get the street data in this method. The predicate *AddressRangeApproximation* maps to the service that approximates the coordinates of the address to be geocoded based on the street end-point coordinates and the range of addresses present on the street.

The Rule R2 defines the *Uniform lot-size* geocoder. The predicate *Street* maps to the service required to get the street data in this method. The predicate *PropertyTax* maps to the web service that queries the property tax data source and provides the number of lots before and after the current address. The predicate

UniformLotApproximation maps to the web service which approximates the location of the address based on the algorithm described in Section 3.2.

```
Domain Rules:

R1:

AddressRangeGeocoder(sa, ci, st, zi, lat, lon):-
           Street(sa, ci, st, zi, frlat, frlon, tolat, tolon,
                  fename, fetype, zipl, zipr, fraddr, fraddl,
                  toaddr, toaddl)^
           AddressRangeApproximation(sa, fraddr, fraddl,
                  toaddr, toaddl,frlat, frlon,
                  tolat, tolon, lat, lon)

R2:

UniformLotSizeGeocoder(sa, ci, co, st, zi, lat, lon):-
           Street(sa, ci, co, st, zi, frlat, frlon,
                  tolat, tolon, fename, fetype, zipl, zipr,
                  fraddr, fraddl, toaddr, toaddl)^
           PropertyTax(sa, ci, co, st, zi, fraddr, fraddl,
                  toaddr, toaddl, addrsbefore, addrsafter,
                  frontage, depth)^
           UniformLotApproximation(frlat, frlon, tolat, tolon,
                  addrsbefore, addrsafter, lat, lon)

R3:

ActualLotSizeGeocoder(sa, ci, co, st, zi, lat, lon):-
           Street(sa, ci, co, st, zi, frlat, frlon,
                  tolat, tolon, fename, fetype, zipl, zipr,
                  fraddr, fraddl, toaddr, toaddl)^
           PropertyTax (sa, ci, co, st, zi, fraddr, fraddl,
                  toaddr, toaddl, addrsbefore, addrsafter,
                  frontage, depth)^
           ActualLotApproximation(sa, fename, fetype frlat, frlon,
                  tolat, tolon, addrsbefore, addrsafter,
                  frontage, depth, lat, lon)
```

Figure 4.1 Domain Model for Columbus

Similarly Rule R3 defines the *Actual lot-size* geocoder. The predicate *PropertyTax* maps to the web service which queries the property tax source and gets the lot dimensions. The *ActualLotApproximation* predicate maps to the web service which calculates the coordinates of the address to be geocoded based on the lot dimensions.

Figure 4.2 gives the source descriptions for the Columbus geocoder. The rule D1 states that the source *LAProperty* provides property information for the properties in Los Angeles County in the state of California. Similarly, the rule D3 states that the *NYProperty* data source provides information for the properties in the state of New York. The rule D2 states that there is a source called *LAProperty_detailed* which gives the detailed property information (the dimensions of the lots) for Los Angeles County in California State. The rule D4 conveys similar information as D3, but for the state of New York. The rule D5 gives detail about a source *SFProperty* which has property information for the city of San Francisco. The rules D6 states that a source *TigerLinesCA* exists to query street data for the state of California. Rule D7 states that there is a source called *NavTechLinesNY* which provides street data for the state of New York

Figure 4.3 shows the graphical representation of the view definitions for three property tax web services and street information web services. Different property tax web services are defined as views over the PropertyTax domain relation. Therefore, when a new property tax web service becomes available, I only need to provide a view definition for the new web service.

```
D1:
LAProperty(streetaddress, city, county, state,  zip, addrsbefore,
                  addrsafter, fraddr, fraddl, toaddr, toaddl ):-
            PropertyTax(streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth) ^
                  (state = "CA") ^ (county = "Los Angeles")
D2:
LAProperty_detailed(streetaddress, city, county, state, zip,
                  addrsbefore, addrsafter, fraddr, fraddl,
                  toaddr, toaddl, lotwidth, lotdepth ):-
            PropertyTax (streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth) ^
                  (state = "CA") ^ (county = "Los Angeles")
D3:
NYProperty(streetaddress, city, county, state, zip, addrsbefore,
                  addrsafter, fraddr, fraddl, toaddr, toaddl ):-
            PropertyTax(streetaddress, city, county, state, zip,
                  fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth) ^
                  (state = "NY")
D4:
NYProperty_detailed(streetaddress, city, county, state, zip,
                  addrsbefore,  addrsafter, fraddr, fraddl,
                  toaddr, toaddl, lotwidth, lotdepth ):-
            PropertyTax (streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth) ^
                  (state = "NY")
D5:
SFProperty(streetaddress, city, county, state, zip,
                  addrsbefore, addrsafter, fraddr, fraddl,
                  toaddr, toaddl, lotwidth, lotdepth ):-
            PropertyTax (streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth) ^
                  (state = "CA") ^ (city = "San Francisco")
D6:
TigerLinesCA(streetaddress, city, state, zip, frlat, frlon, tolat,
            tolon, fename, fetype, zipl, zipr,  fraddr, fraddl,
            toaddr, toaddl):-
            Street(streetaddress, city, state, zip,
                  frlat, frlon, tolat, tolon, fename, fetype,
                  zipl, zipr, fraddr, fraddl, toaddr, toaddl) ^
                  (state = "CA")
D7:
NavTechLinesNY(streetaddress, city, state, zip, frlat, frlon,
            tolat, tolon, fename, fetype, zipl, zipr,
            fraddr, fraddl, toaddr, toaddl):-
            Street(streetaddress, city, state, zip,
                  frlat, frlon, tolat, tolon, fename, fetype,
                  zipl, zipr, fraddr, fraddl, toaddr, toaddl) ^
                  (state = "NY")
```

Figure 4.2 Source Descriptions for Columbus

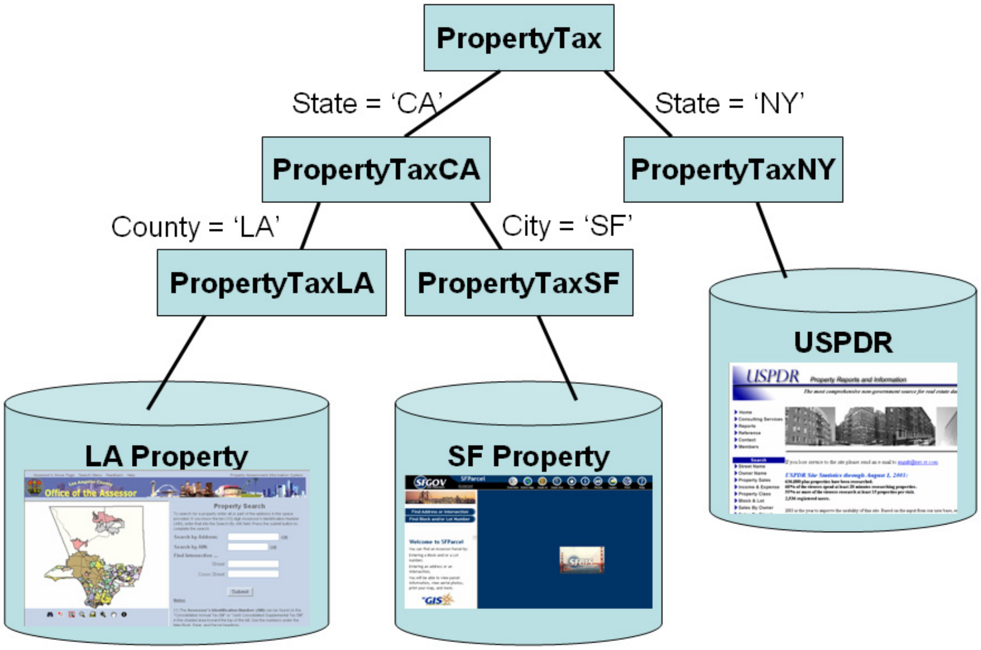


Figure 4.3 View definitions for various property sources

4.3 Querying Columbus

Having described the sources and the domain rules, Columbus can now be queried to geocode an address. For example, to geocode an address “645 Sierra St, El Segundo, Los Angeles, CA, 90245,” I would specify the following query:

```
Q1(streetaddress, city, state, zip, lat, lon):-
        UniformLotAccurateGeocoder (streetaddress, city, state, zip) ^
                streetaddress = “645 Sierra St” ^
                city = “El Segundo” ^
                state = “CA”^
                zip = “90245”
```

The mediator gets this query and inverts the source descriptions using the Inverse-rules algorithm described in (Duschka 1997). By inverting the sources, the mediator obtains the definitions for the domain predicates as views over source predicates.

Figure 4.4 shows the inverted rules generated by the mediator. These rules are obtained by inverting the source descriptions in Figure 4.2.

In this case we get a definition of the *Street* and *PropertyTax* domain predicates. An inverted rule IR1 implies that the *PropertyTax* domain is a result of a join of the predicates *LAProperty* and *LAProperty_detailed.* IR2 implies the *PropertyTax* domain is a result of join of the predicates *NYProperty* and *NYProperty_detailed*. IR1, IR2 and IR3 have the same head relation, which implies a union in datalog. Therefore *PropertyTax* domain predicate is a union of, a join between Los Angeles property tax sources, a join between New York property tax sources and the San Francisco property tax source.

Similarly, the Inverted rules IR4 and IR5 are obtained from inverting the rules D6 and D7. They show that the *Street* domain predicate is a union of the *TigerLinesCA* and *NavTechLinesNY* source predicates.

```
IR1:
PropertyTax(streetaddress, city, “Los Angeles”, “CA”, zip,
            fraddr, fraddl, toaddr, toaddl,
            addrsbefore, addrsafter, frontage, depth ):-
            LAProperty (streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter)^
            LAProperty_detailed(streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth)

IR2:
PropertyTax(streetaddress, city, county, “NY”, zip,
                  fraddr, fraddl, toaddr, toaddl,addrsbefore,
                  addrsafter, frontage, depth ):-
            NYProperty (streetaddress, city, county, state, zip,
                  fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter)^
            NYProperty_detailed(streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth)

IR3:
PropertyTax(streetaddress, “San Francisco”, county, “CA”,
            zip, fraddr, fraddl, toaddr, toaddl ,
            addrsbefore, addrsafter, frontage, depth):-
            SFProperty (streetaddress, city, county, state, zip,
                  fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth)

IR4:
Street(streetaddress, city, “CA”, zip, frlat, frlon, tolat,
            tolon, fename, fetype, zipl, zipr,  fraddr, fraddl,
            toaddr, toaddl):-
            TigerLinesCA(streetaddress, city, state, zip,
                  frlat, frlon, tolat, tolon, fename, fetype,
                  zipl, zipr, fraddr, fraddl, toaddr, toaddl)


IR5:
Street(streetaddress, city, “NY”, zip, frlat, frlon,
            tolat, tolon, fename, fetype, zipl, zipr,
            fraddr, fraddl, toaddr, toaddl):-
            NavTechLinesNY(streetaddress, city, state, zip,
                  frlat, frlon, tolat, tolon, fename, fetype,
                  zipl, zipr, fraddr, fraddl, toaddr, toaddl)
```

Figure 4.4 Inverted Source Descriptions

Next, the mediator combines the inverted source descriptions, the domain rules, and the query to generate a datalog program to answer the user query. Figure 4.5 shows the datalog program that would be generated for the query Q1. When the mediator evaluates the rules in the program, it first queries the *TigerLinesCA* source to obtain the street information for the given address. Next, the *LAProperty* tax source is queried to get the property related information, more specifically, the number of addresses before and after the given address on the street segment. It then calls the uniform lot approximation web service which takes these parameters and returns the latitude and longitude values which in this case are 33.92491 degrees and -118.40869 degrees respectively.

```
Q1(streetaddress, city, state, zip, lat, lon):-
            UniformLotAccurateGeocoder(sa, ci, co,  st, zi,
            lat, lon) ^
            sa = “645 Sierra St” ^
            ci = “El Segundo” ^
            st = “CA”^
            zi = “90245”

UniformLotSizeGeocoder(sa, ci, co, st, zi, lat, lon):-
           Street(sa, ci, co, st, zi, frlat, frlon,
                  tolat, tolon, fename, fetype, zipl, zipr,
                  fraddr, fraddl, toaddr, toaddl)^
           PropertyTax(sa, ci, co, st, zi, fraddr, fraddl,
                  toaddr, toaddl, addrsbefore, addrsafter,
                  frontage, depth)^
           UniformLotApproximation(frlat, frlon, tolat, tolon,
                  addrsbefore, addrsafter, lat, lon)

Street(streetaddress, city, “CA”, zip, frlat, frlon, tolat,
            tolon, fename, fetype, zipl, zipr,  fraddr, fraddl,
            toaddr, toaddl):-
            TigerLinesCA(streetaddress, city, state, zip,
                  frlat, frlon, tolat, tolon, fename, fetype,
                  zipl, zipr, fraddr, fraddl, toaddr, toaddl)

PropertyTax(streetaddress, city, “Los Angeles”, “CA”, zip,
            fraddr, fraddl, toaddr, toaddl,
            addressbefore, addressafter, frontage, depth):-
            LAProperty (streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter) ^
            LAProperty_detailed(streetaddress, city, county,
                  state, zip, fraddr, fraddl, toaddr, toaddl,
                  addrsbefore, addrsafter, frontage, depth)
```

Figure 4.5 Datalog program generated by the mediator

4.4 Adding New Sources

As more and more property data sources become available online, their descriptions need to be incrementally added to the mediator's domain model. The GLAV data-model is used to define the data sources in the mediator. This is very convenient when new sources have to be added in the system. For example, if a new county data (say Fresno) is available online, it is defined by the following predicate:

Fresno(streetaddress, city, county, state, zip, fraddr, fraddl, toaddr, toaddr)

After defining the predicate for the new data source, I add the source description in terms of the source descriptions:

```
Fresno(streetaddress, city, county, state, zip, before,
               after, fraddr, fraddl, toaddr, toaddl ):-
               PropertyTax(streetaddress, city, county, state, zip, fraddr, fraddl,
                          toaddr, toaddl, addrsbefore, addrsafter, frontage, depth) ^
                          (state = "CA") ^
                          (county = "Fresno")
```

Suppose we add this source description to the model in Figure 4.2, and a new query is requested from Columbus. The Columbus now inverts the rules and a new inverted rule is created:

```
PropertyTax(streetaddress, city, "Fresno", "CA", zip,
        fraddr, fraddl, toaddr, toaddl, addrsbefore, addrsafter, frontage, depth ):-
                          Fresno (streetaddress, city, county,
                                  state, zip, fraddr, fraddl, toaddr, toaddl,
                                  addrsbefore, addrsafter)
```

Therefore the *PropertyTax* is now a union of the Los Angeles County, Fresno County, New York State and San Francisco city sources. If a new query is given to Columbus for an address in Fresno County, it would make use of the Fresno property tax information to calculate the coordinates. It can be clearly seen that the choice of GLAV to describe the data sources makes the geocoder very scalable.

Chapter 5

Results

In this chapter, I present empirical results for the methods that I have described. To run the experiment, I selected the region bounded by the intersection of Sheldon St., Center St., E. Mariposa Ave., E. Maple Ave. in El Segundo, CA as shown in Figure 5.1. This area has 267 addresses spread over thirteen well-defined blocks. I selected this region due to the availability of conflated (Saalfeld 1993) TIGER/Line data source. These lines were automatically conflated by methods described in (Chen, Thakkar et al. 2003). This data is much accurate than the original TIGER/Line data. The underlying data source can be any other repository like NavTech, GDT, etc as well. The sources (property tax websites) are converted into XML format by wrapping them using Fetch Technology's Agent Platform.[21] The information mediator used is Prometheus 2.0 (Thakkar, Ambite et al. 2003).

[21] http://www.fetch.com/

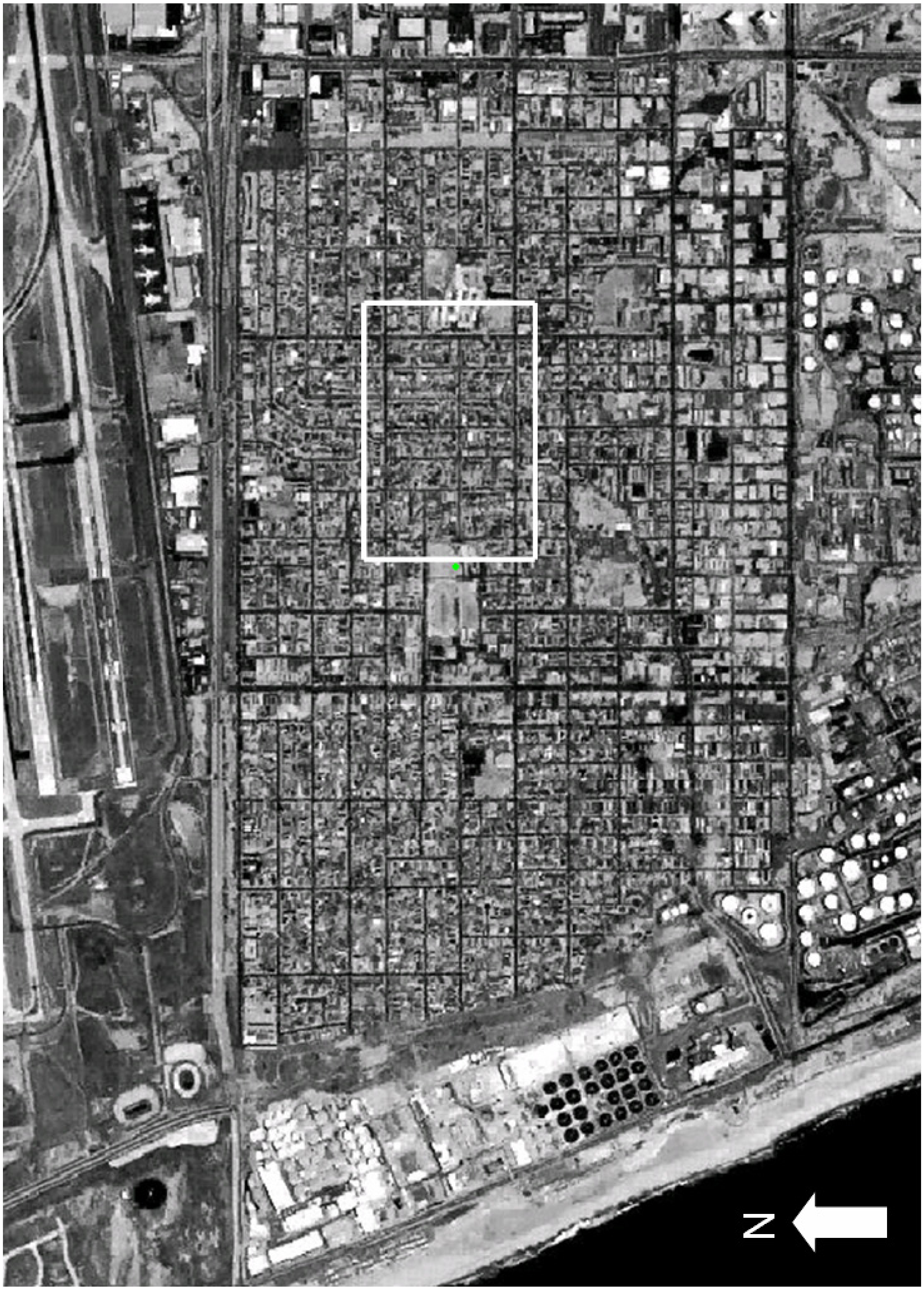


Figure 5.1 Region selected for geocoding

The three methods are evaluated based on the error in the geocoded coordinates compared to the actual locations. This error is measured as the driving distance on the street from the geocoded location, to the center on the actual location of the address projected on the street. I call this the 'driving-distance' technique for estimating error. For example, in Figure 5.2, the cross indicates the geocoded value of lot 14. The error is calculated as the driving distance from the geocoded location to the actual center of the lot.

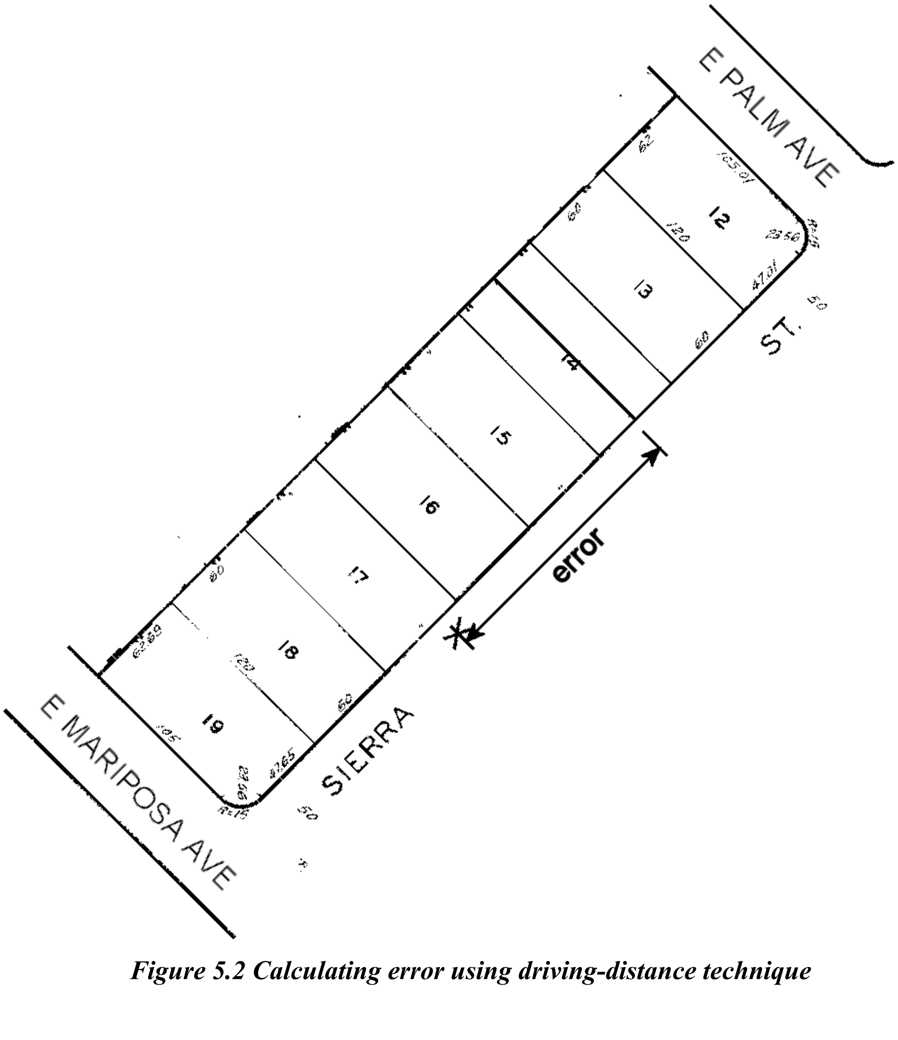


Figure 5.2 Calculating error using driving-distance technique

To measure the error in meters, I use the Sinnott's Formula (Sinnott 1984). Figure 5.3 shows the calculations needed for this formula. The coordinates of the two points between which the distance is to be calculated are (lat1, lon1) and (lat2, lon2) respectively. For calculations, the average earth radius is taken as 6,378,137 meters. I calculate the actual coordinates for all the lots on the block by mapping the conflated TIGER/Line on the assessor's map for the region. I then calculate the center points of the frontage of the lots and project it on the center of the street, perpendicular to the frontage.

```
EarthRadius = 6378137.0

dlon = lon2 - lon1;
dlat = lat2 - lat1;

a = (Sine(dlat/2))^2 + Cosine(lat1) * Cosine(lat2) *
           (Sine(dlon/2))^2

c = 2 * Sine^-1(√(a));

dist = EarthRadius * c;
```

Figure 5.3 Calculations for Sinnott's formula

In Sections 5.1 to 5.3, I show an analysis of the error by the three geocoding methods for one block in the selected region. This block is formed by the streets: Penn St., E. Mariposa Ave, Sierra St. and E. Palm Ave. I also provide the satellite imagery of this block with the geocoded locations plotted on it. In Section 5.4, I present the comparison and analysis of a more comprehensive set of results consisting of the geocoded locations of the addresses in the region selected.

5.1 The Address-range method

As explained in detail earlier, the traditional address range method which is the one used by most of the existing online services, assumes that all the houses/lots mentioned in a range of values provided by the data sources like TIGER/Line exist. This gives us results which have a very high error margin in them. Most of the existing geocoding services use this method to geocode the addresses.

Table 5.1 gives the results for the address range method of geocoding for the houses on the Sierra, Mariposa, Penn and Palm streets. The average error for entire block is 41.09 m (137.14 ft). Also the maximum error for the lot is 76.48 m or 245.33 ft. The average lot width on the given block is 17.69 m or 59.04 ft. Thus the method gives us results which are on an average 2 lots off from the original and in the worst case scenario, the geocoded values are 4 lots off from the original lot. Figure 5.4 shows the points plotted on the image of the block. The circular points are the center points of the actual lots and the crosses are the geocoded location of the lots from this method.

Table 5.1 Error in the Address-range method of geocoding

	Actual		**AddressRangeMethod**		**Error in**
Address	**Latitude**	**Longitude**	**Latitude**	**Longitude**	**Meters**
611 Sierra St	33.92400	-118.40890	33.92384	-118.40869	24.10
617 Sierra St	33.92416	-118.40890	33.92392	-118.40869	31.43
623 Sierra St	33.92432	-118.40890	33.92401	-118.40869	39.77
629 Sierra St	33.92448	-118.40890	33.92409	-118.40869	48.46
633 Sierra St	33.92464	-118.40890	33.92415	-118.40869	60.14
639 Sierra St	33.92480	-118.40890	33.92423	-118.40869	66.10
645 Sierra St	33.92495	-118.40890	33.92432	-118.40869	76.48
				Avg Error	**49.50**
				Max Error	**76.48**
606 Penn St	33.92389	-118.40958	33.92378	-118.40975	13.10
610 Penn St	33.92407	-118.40958	33.92384	-118.40975	28.52
618 Penn St	33.92420	-118.40958	33.92395	-118.40975	31.49
624 Penn St	33.92433	-118.40958	33.92404	-118.40975	38.47
628 Penn St	33.92445	-118.40958	33.92409	-118.40975	46.08
630 Penn St	33.92458	-118.40958	33.92412	-118.40975	56.33
636 Penn St	33.92472	-118.40958	33.92421	-118.40975	61.62
642 Penn St	33.92485	-118.40958	33.92429	-118.40975	67.92
				Avg Error	**42.94**
				Max Error	**67.92**
604 E Palm Ave	33.92497	-118.40955	33.92509	-118.40951	0.51
610 E Palm Ave	33.92497	-118.40931	33.92509	-118.40916	12.68
				Avg Error	**6.60**
				Max Error	**12.68**
633 E Mariposa Ave	33.92384	-118.40898	33.92369	-118.40940	36.37
				Avg Error	**36.37**
				Max Error	**36.37**
		For Entire Block:		**Avg Error**	**41.09**
				Max Error	**76.48**

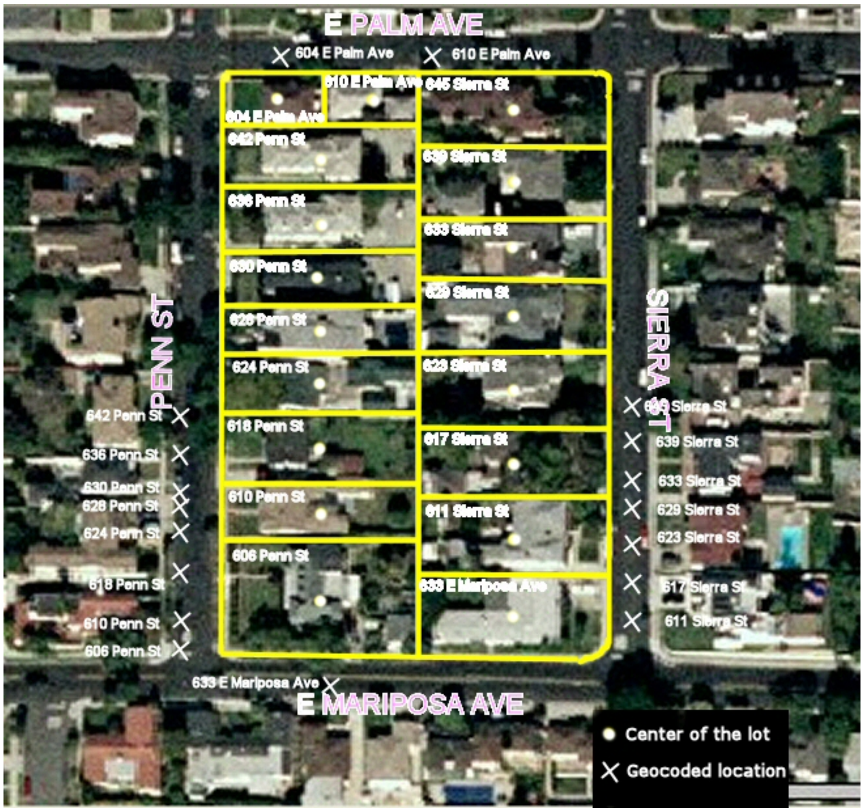


Figure 5.4 Results of the Address-range method

5.2 The Uniform lot-size method

The Uniform lot-size method uses information from the Los Angeles County's property tax site to get the number of lots present in the address range provided by the TIGER/Line data source. This method gives better results over the traditional approach, as the number of lots on which we approximate our results is correct.

There is a significant reduction in error compared to the previous approach. The average error is reduced with this method from 41.08 m (137.14 ft) to 10.55 m (34.6 ft) and the maximum error reduces from 76.48 m (245.33 ft) to 23.01 m (75.5 ft).

Given the average width of the lot for this block 17.69 m, we are on an average one lot off from the original and in the worst case around three lots. Although there is a considerable improvement in the geocoded values compared to the previous approach, there is still some error because this method assumes the sizes of each of the lots to be the same which is often not the case. Also as previously discussed, the orientation of the corner lots is also not known. This adds to the error of the results from this method.

Table 5.2 shows the comparative errors for this method on Sierra, Penn, Palm and Mariposa streets. It can be seen the result from this approach are much more accurate compared to the Address-range method. An interesting observation is that on E. Palm Ave., this method does not perform as good as the traditional method. This is due to the fact that we assume that all the lots are equal in size which is not the case on this street segment. However, overall this method yields better results than the traditional approach. Figure 5.5 shows the points plotted on the image of the block. The circular points are the actual lots and the crosses are the geocoded location of the lots from this method.

Table 5.2 Error in Uniform lot-size method – for the entire block

	Actual		**UniformLotSize**		**Error in**
Address	**Latitude**	**Longitude**	**Latitude**	**Longitude**	**Meter**
611 Sierra St	33.92400	-118.40890	33.92387	-118.40869	20.35
617 Sierra St	33.92416	-118.40890	33.92404	-118.40869	17.25
623 Sierra St	33.92432	-118.40890	33.92422	-118.40869	15.17
629 Sierra St	33.92448	-118.40890	33.92439	-118.40869	13.42
633 Sierra St	33.92464	-118.40890	33.92457	-118.40869	11.34
639 Sierra St	33.92480	-118.40890	33.92474	-118.40869	6.88
645 Sierra St	33.92495	-118.40890	33.92491	-118.40869	6.84
				Avg Error	**13.04**
				Max Error	**20.35**
606 Penn St	33.92389	-118.40958	33.92385	-118.40975	4.95
610 Penn St	33.92407	-118.40958	33.92400	-118.40975	8.87
618 Penn St	33.92420	-118.40958	33.92416	-118.40975	7.02
624 Penn St	33.92433	-118.40958	33.92431	-118.40975	5.85
628 Penn St	33.92445	-118.40958	33.92447	-118.40975	1.96
630 Penn St	33.92458	-118.40958	33.92462	-118.40975	2.61
636 Penn St	33.92472	-118.40958	33.92478	-118.40975	5.48
642 Penn St	33.92485	-118.40958	33.92493	-118.40975	7.33
				Avg Error	**5.51**
				Max Error	**8.87**
604 E Palm Ave	33.92497	-118.40955	33.92509	-118.40939	10.83
610 E Palm Ave	33.92497	-118.40931	33.92509	-118.40904	23.01
				Avg Error	**16.92**
				Max Error	**23.01**
633 E Mariposa Ave	33.92384	-118.40898	33.92369	-118.40922	**20.79**
				Avg Error	**20.79**
				Max Error	**20.79**
		For Entire Block:		**Avg Error**	**10.55**
				Max	**23.01**

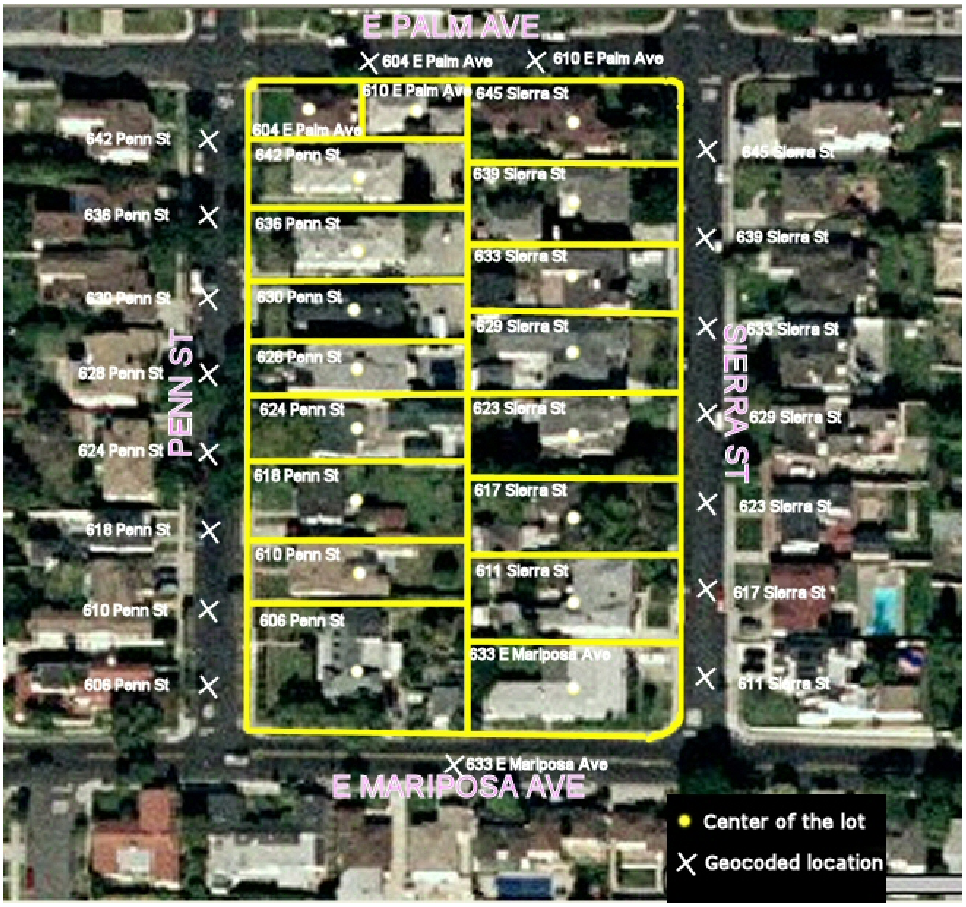


Figure 5.5 Results of the Uniform lot-size method

5.3 The Actual lot-size method

The Actual lot-size method estimates the orientation of the corner lots on the given block and the size of the individual lots. This approach further reduces the error. The average error is 3.52 m (11.54 ft) and in the worst case it is 7.31 m (23.98 ft). This partially accomplishes the goal as we are now in the same lot even in the worst case. The nominal error that exists could have been further reduced if not eliminated if the underlying data source (in this case TIGER/Line) aligned perfectly with streets and had no noise in it. Table 5.3 shows the error in each of the street for the block formed by Sierra, Penn, Palm and Mariposa streets. Figure 5.6 shows the points plotted on the image of the block.

Table 5.3 Error in Actual lot-size method – for entire block

	Actual		**Block Accurate Calculated**		**Error in**
Address	Latitude	Longitude	Latitude	Longitude	**Meter**
611 Sierra St	33.92400	-118.40890	33.92395	-118.40894	7.31
617 Sierra St	33.92416	-118.40890	33.92412	-118.40894	5.52
623 Sierra St	33.92432	-118.40890	33.92430	-118.40894	4.76
629 Sierra St	33.92448	-118.40890	33.92447	-118.40894	4.33
633 Sierra St	33.92464	-118.40890	33.92465	-118.40894	3.57
639 Sierra St	33.92480	-118.40890	33.92482	-118.40894	0.42
645 Sierra St	33.92495	-118.40890	33.92500	-118.40894	1.38
				Average Error	**3.90**
				Max Error	**7.31**
Penn St					
606 Penn St	33.92389	-118.40958	33.92382	-118.40947	3.81
610 Penn St	33.92407	-118.40958	33.92403	-118.40947	3.12
618 Penn St	33.92420	-118.40958	33.92417	-118.40947	3.51
624 Penn St	33.92433	-118.40958	33.92432	-118.40947	4.73
628 Penn St	33.92445	-118.40958	33.92445	-118.40947	4.82
630 Penn St	33.92458	-118.40958	33.92458	-118.40947	3.60
636 Penn St	33.92472	-118.40958	33.92474	-118.40947	2.01
642 Penn St	33.92485	-118.40958	33.92489	-118.40947	1.43
				Average Error	**3.38**
				Max Error	**4.82**
E Palm Ave					
604 E Palm Ave	33.92497	-118.40955	33.92503	-118.40961	1.29
610 E Palm Ave	33.92497	-118.40931	33.92503	-118.40933	3.56
				Average Error	**2.43**
				Max Error	**3.56**
E Mariposa Ave					
633 E Mariposa Ave	33.92384	-118.40898	33.92378	-118.40895	4.15
				Average Error	**4.15**
				Max Error	**4.15**
		For Entire Block:		**Avg Error**	**3.52**
				Max Error	**7.31**

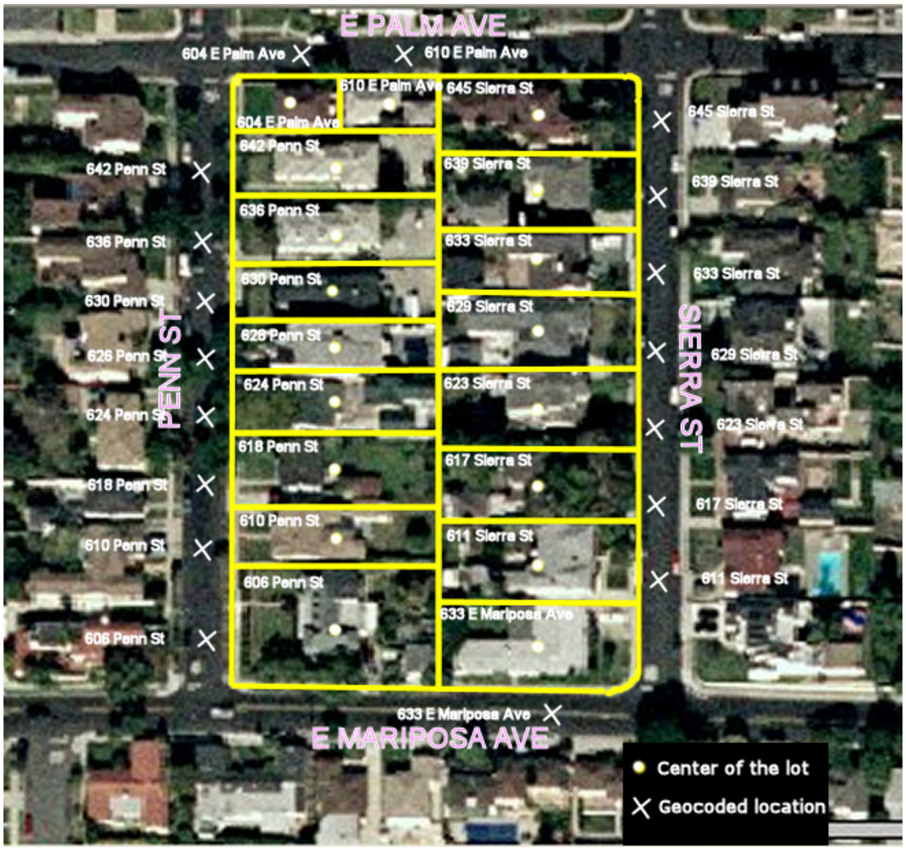


Figure 5.6 Results of the Actual lot-size method

5.4 Comparison of the Methods

In this section I compare the accuracy of the three methods of geocoding. The basis of this comparison is the geocoding of the addresses in the area selected for geocoding. The region selected for geocoding has thirteen well defined blocks consisting of 267 addresses. The comparison of errors and the maps of each of the blocks are presented in the Appendix. Out of the 267 addresses selected, 208 addresses could be geocoded by all the three methods. 58 addresses which were parts of blocks shown in Figures A.11, A.12 and A.13 could not be geocoded by the Actual lot-size method, while one address could not be geocoded by any of the three methods. These three blocks were completely excluded from the test-set. This was done so that we could evaluate the performance of all the three methods on a common set of addresses.

As mentioned earlier, the Actual lot-size method requires that the block formed by the intersection of the streets is rectangular and lots are rectangular as well. In Figure A.11, it can be seen that there is a small alley perpendicular to Sheldon Street and the address 519 E Palm Ave is not rectangular in shape. Hence the Actual lot-size method could not be applied here. However the Uniform lot-size method could be applied in this case and Table A.11 has a comparison of error over the traditional approach.

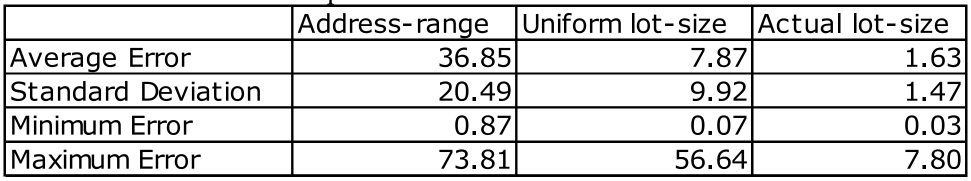
The block formed by the streets shown in Figure A.12, also could not be geocoded by the Actual lot-size method since the address 501 and 511 Mariposa Ave are not rectangular in shape. Also the addresses 523, 525 and 527 have a peculiar layout and the Actual lot-size algorithm as of now does not handle this case. Uniform lot-size method could, however, be applied to this block and Table A.12 gives a comparison of the error of this method. Similarly, the block of streets shown in Figure A.13 could not be geocoded since it does not form a rectangular block because of an alley (Irene Ct). The Actual lot-size method could be extended to handle these new geometric shapes and is a part of the future work for this research.

There was one address in the test-region that could not be geocoded by any of the three methods. The address 708 E Palm Ave could not be geocoded due to inaccurate address range provided in the TIGER/Line data source. According to the data source, the address ranges 620-698 exist on this street segment which is not the case.

To do an analysis of error in geocoding, I select the 208 addresses in this region on which all the three methods of geocoding described in this dissertation were applicable. This comprised of ten well-defined blocks. The error is calculated using the driving-distance technique described earlier in this chapter. Table 5.4 gives a summary of the average error for all the 208 addresses. Figure 5.7 gives the normal distribution of the error from all the three methods. With the Uniform lot-size method, the average error reduces from 36.85m to 7.87m and improvement of 79% over the traditional approach. The Actual lot-size method reduces the error further to 1.63m on an average. This gives an improvement of 96% over the traditional method. Further, for all the ten blocks that were a part of the test-set, the Actual lot-size method determined the orientation of the corner-lots in the block correctly. The average error for each of the street segments along with the Assessor's map for those streets is shown in the Appendix.

The average response time for the query was 410 ms for the Address-range method, 511 ms for the Uniform lot-size method and 3415 ms for the Actual lot-size method. The property related data was cached locally for these experiments and was not retrieved in real-time. The Actual lot-size method is more expensive because of the computation of the corner lots and there is a considerable room for optimizing this further and is a part of future work for this research.

Table 5.4: Comparison of error from all the three methods



	Address-range	Uniform lot-size	Actual lot-size
Average Error	36.85	7.87	1.63
Standard Deviation	20.49	9.92	1.47
Minimum Error	0.87	0.07	0.03
Maximum Error	73.81	56.64	7.80

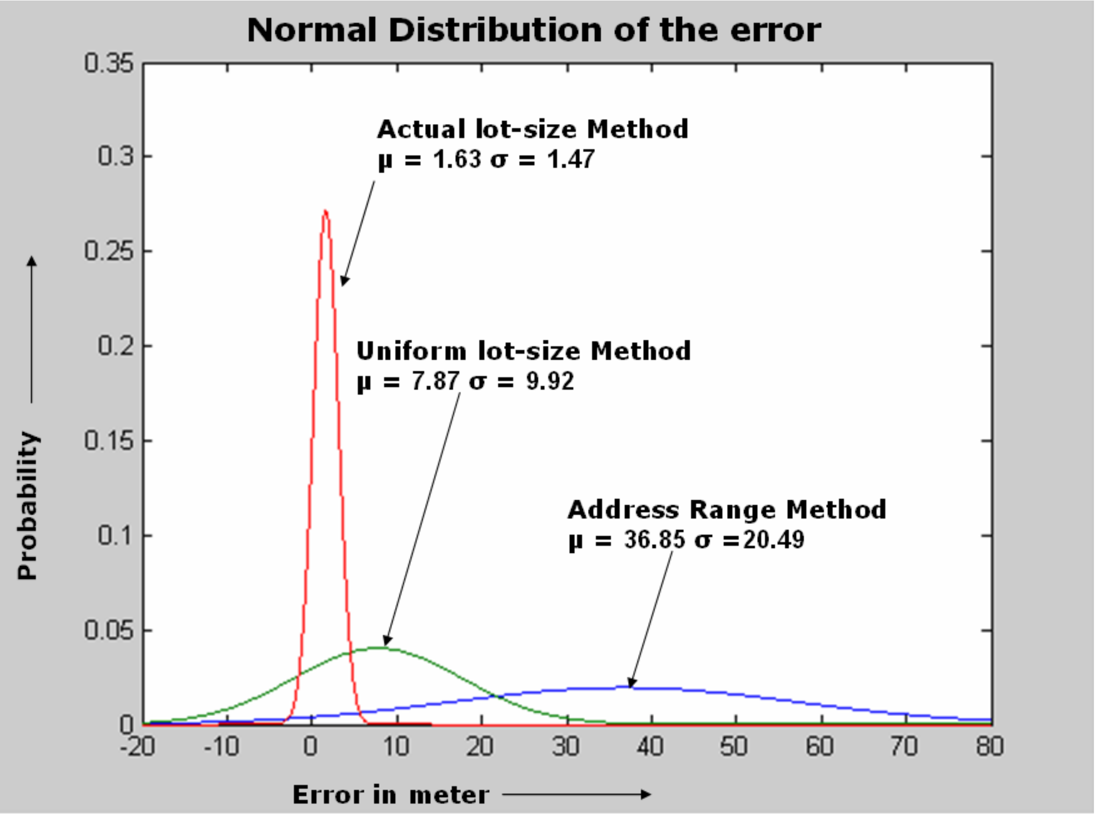


Figure 5.7 Normal distribution of error from all the three methods

Chapter 6

Related Work

The research works related to this paper can be broadly classified in two categories. The first category of research work is in the area of measuring the inaccuracy of available street data and existing geocoding web sites (Ratcliffe 2001) (Cayo and Talbot 2003) (Krieger, Waterman et al. 2001), while the second category of research is in the area of geo-spatial data integration using data integration systems .

All geocoding algorithms rely on some street vector data to identify the location of the given address. A study by Ratcliffe (Ratcliffe 2001) about accuracy of Tigerline files in Australia showed that out of 20,000 addresses geocoded using Tigerlines data, less than 5% of geocoded points were on the correct lot. The two key factors behind the error are inaccuracy of the Tigerlines and inaccuracy introduced by the approximation performed by the geocoding algorithm. In this paper, we reduce the uncertainty introduced by the geocoding algorithm by 89% by utilizing online data sources. In past work our group has introduced automated conflation techniques to align street vector data with satellite imagery or maps (Chen, Thakkar et al. 2003). For the experiments with Columbus, we used the conflated TIGER/Lines obtained from these techniques. Cayo and Talbot and Krieger et. al. have studied the accuracy of commercial geocoding sites for addresses in the U.S.A. Both studies support our claims that the traditional geocoding methods may provide geographic coordinates for inaccurate addresses and the geographic coordinates provided for existing addresses by the traditional methods are often very inaccurate.

In the data integration community there has been some work on integrating geo-spatial datasets. The goal of MIX mediator (Gupta, Marciano et al. 1999) and TSIMMIS mediator (Garcia-Molina, Hammer et al. 1995) is to provide unified access to a wide variety of data sources. Both mediator systems utilize Global-As-View approach to integrate data. Adding new sources to Global-As-View model may require changing all the rules in the domain model. In case of Columbus new property tax web sites become available everyday, therefore, the GLAV approach is more suitable. In general, as geo-spatial data sources often vary in coverage and new data sources with different coverage become available every day GLAV approach is more suitable.

Chapter 7

Conclusions

This dissertation shows how Information integration techniques can be used to combine different web-services and achieve a substantial improvement in the geocoding process. I proposed two new approaches that used Information Integration techniques to gather property related data available on the Internet and used this information to perform more accurate Geocoding. The various sources were integrated using the Prometheus information mediator. The sources were modeled as Local-As-View which made the architecture scalable. The property sources were converted into XML webservices using Fetch Agent Builder.

These methods provide more accurate results compared to the existing Geocoding techniques. The first method, Uniform lot-size resulted in an improvement of 79% over the existing methods, while the Actual lot-size method resulted in and improvement of 96% over the existing methods.

7.1 Contribution

With this research, I have realized a geocoder Columbus which exploits online data sources to geocode addresses with higher accuracy. The two main contributions that I have made are:

1. I developed novel algorithms to exploit online data to perform geocoding with higher accuracy than the existing method.

2. I applied data integration techniques to organize and integrate a large number of online sources, relevant for geocoding, in an extensible framework.

7.2 Limitations and Future Work

The Actual lot-size method assumes the lot to be a rectangular block. If that is not the case, Columbus reverts to Uniform lot-size method. As a next level of accuracy, I would like to extend the Actual lot-size method to handle cases where the block is not rectangular in shape. This can be done by determining the shape of the block to which the address belongs and modifying the algorithm, to detect the street the corner lots belong. If the property tax source has the dimensions for the address, but the address does not belong to a well defined shape of a block, then the Uniform lot-size method can be extended to account for the size of all the lots on the street segment and approximate the corner lot size as the average of the other lots present on the street.

Also as a future work, I would like to integrate more data sources. For some areas, the Property Tax websites may not be available. In these cases, it will be a good idea to combine the data from the US Postal Services websites. At the time this dissertation was written, I have only used Los Angeles property tax website as data source for property information. There are many other property tax websites that can be incorporated in Columbus.

Reference List

Bayardo Jr., R. J., W. Bohrer, et al. (1997). Infosleuth: Agent-based semantic integration of information in open and dynamic environments. In Proceedings of ACM SIGMOD-97.

Cayo, M. R. and T. O. Talbot (2003). "Positional error in automated geocoding of residential addresses." International Journal of Health Geographics **2**(10).

Chen, C.-C., C. A. Knoblock, et al. (2003). Automatically and Accurately Conflating Satellite Imagery and Maps. International Workshop on Next Generation Geospatial Information, Cambridge, MA.

Chen, C.-C., S. Thakkar, et al. (2003). Automatically Annotating and Integrating Spatial Datasets. In the Proceedings of International Symposium on Spatial and Temporal Databases, Santorini Island, Greece.

Committee, US Federal Geographic Data Committee (2003). "Homeland Security and Geographic Information Systems – How GIS and mapping technology can save lives and protect property in post-September 11th America." Public Health GIS News and Information **52**: 20-23.

Duschka, O. M. (1997). Query Planning and Optimization in Information Integration. Ph.D. Thesis, Computer Science, Stanford University.

Garcia-Molina, H., J. Hammer, et al. (1995). Integrating and Accessing Heterogeneous Information Sources in TSIMMIS. Proceedings of the AAAI Symposium on Information Gathering, Stanford, CA.

Genesereth, M. R., A. M. Keller, et al. (1997). InfoMaster: An information integration system. In Proceedings of ACM SIGMOD-97.

Gupta, A., R. Marciano, et al. (1999). Integrating GIS and Imagery through XML-Based Information Mediation. Proc. NSF International Workshop on Integrated Spatial Databases: Digital Images and GIS.

Knoblock, C., S. Minton, et al. (2001). "The ARIADNE Approach to Web-Based Information Integration." International Journal on Intelligent Cooperative Information Systems (IJCIS) **10**(1-2): 145-169.

Knoblock, C. A., S. Minton, et al. (1998). Modeling web sources for information integration. In Preceedings of the Fifteenth National Conference on Artificial Intelligence, Madison, WI.

Krieger, A., P. Waterman, et al. (2001). " On the wrong side of the tracts? Evaluating the accuracy of geocoding in public health research." American Journal of Public Health **91**(7): 1114-1116.

Lenzerini, M. (2002). Data integration: A theoretical perspective. In Proceedings of ACM Symposium on Principles of Database Systems. Madison, Winsconsin, USA.

Levy, A. (2000). Logic-Based Techniques in Data Integration. Logic Based Artificial Intelligence. J. Minker, Kluwer Publishers.

Levy, A. Y., A. Rajaraman, et al. (1996). Querying Heterogeneous Information Sources Using Source Descriptions. Proceedings of the 22nd VLDB Conference, Bombay, India.

Ratcliffe, J. H. (2001). "On the accuracy of TIGER-type geocoded address data in relation to cadastral and census areal units." Int. J Gographical Information Science **15**(5): 473-485.

Saalfeld, A. (1993). Conflation: Automated Map Compilation. Computer Vision Laboratory, Center for Automation Research, University of Maryland.

Sinnott, R. W. (1984). "Virtues of the Haversine." Sky and Telescope **68**(2): 159.

Thakkar, S., J. L. Ambite, et al. (2003). A view integration approach to dynamic composition of web services. In Proceedings of 2003 ICAPS Workshop on Planning for Web Services, Trento, Italy.

TIGER/Lines, U.S.Census Bureau.-. (2000). U.S.Census Bureau - TIGER/Lines. **2002**.

Appendix

In this appendix, I present the result from a comprehensive set of experiments which show the accuracy of the methods described in this dissertation. Tables A.1 to A.13 give a comparison of error of all the addresses geocoded, while figures A.1 to A.13 give the maps for the areas in the corresponding tables.

Table A.1: Comparison of Error

		Error in Meters	
Address	**Address Range**	**Uniform Lot Size**	**Actual Lot Size**
611 Sierra St	24.10	20.35	7.31
617 Sierra St	31.43	17.25	5.52
623 Sierra St	39.77	15.17	4.76
629 Sierra St	48.46	13.42	4.33
633 Sierra St	60.14	11.34	3.57
639 Sierra St	66.10	6.88	0.42
645 Sierra St	76.48	6.84	1.38
Average Error	49.50	13.04	3.90
Maximum Error	76.48	20.35	7.31
Penn St			
606 Penn St	13.10	4.95	3.81
610 Penn St	28.52	8.87	3.12
618 Penn St	31.49	7.02	3.51
624 Penn St	38.47	5.85	4.73
628 Penn St	46.08	1.96	4.82
630 Penn St	56.33	2.61	3.60
636 Penn St	61.62	5.48	2.01
642 Penn St	67.92	7.33	1.43
Average Error	42.94	5.51	3.38
Maximum Error	67.92	8.87	4.82
E Palm Ave			
604 E Palm Ave	0.51	10.83	1.29
610 E Palm Ave	12.68	23.01	3.56
Average Error	6.60	16.92	2.43
Maximum Error	12.68	23.01	3.56
E Mariposa Ave			
633 E Mariposa Ave	36.37	20.79	4.15
Average Error	36.37	20.79	4.15
Maximum Error	36.37	20.79	4.15
For Entire Block			
Average Error	41.09	10.55	3.52
Maximum Error	76.48	23.01	7.31

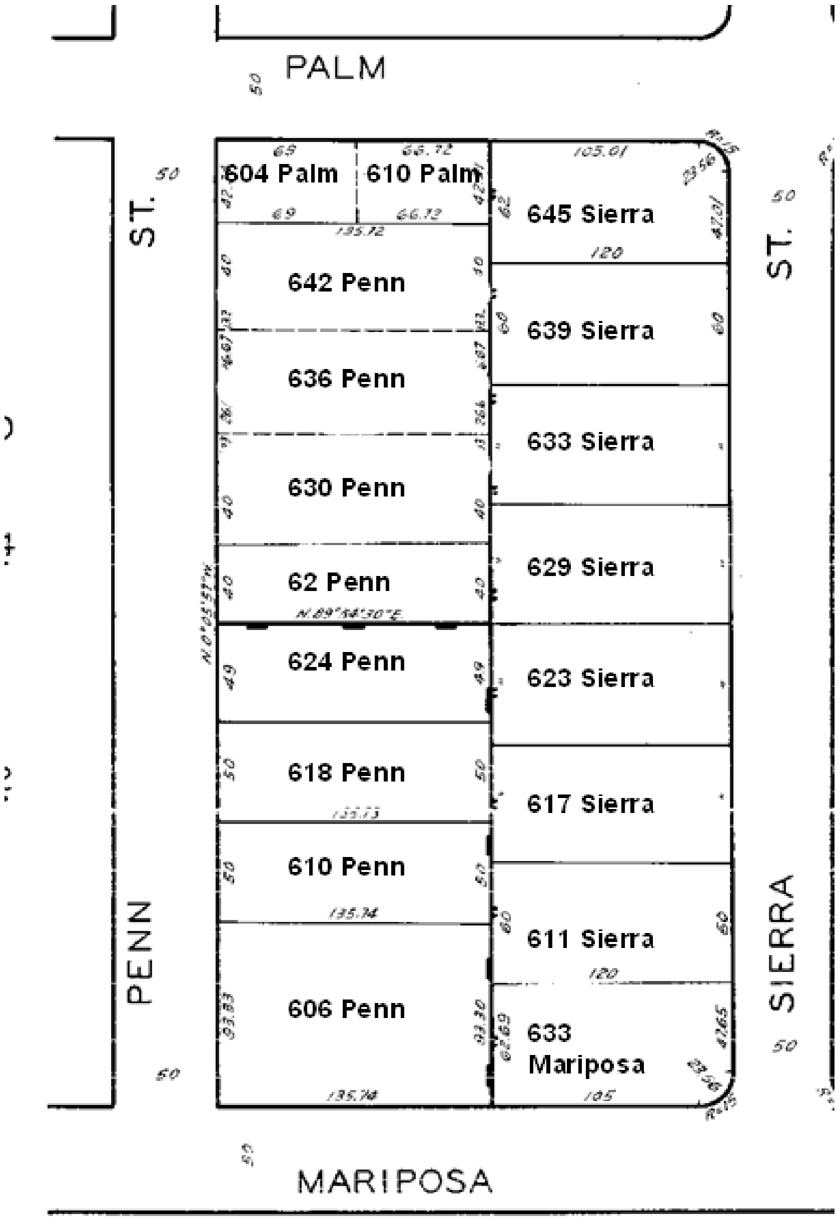


Figure A.1: Map for area geocoded in Table A.1

Table A.2: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Penn St			
754 Penn St	11.46	5.64	0.36
750 Penn St	3.84	12.37	0.78
742 Penn St	11.48	6.50	0.76
738 Penn St	12.68	11.02	1.00
734 Penn St	12.03	13.69	0.07
730 Penn St	7.20	12.19	0.30
726 Penn St	12.08	3.68	1.40
724 Penn St	20.97	2.06	0.59
716 Penn St	3.71	1.14	0.47
712 Penn St	3.42	0.29	0.36
708 Penn St	2.45	0.32	0.19
704 Penn St	0.87	0.63	0.71
Average	**9.21**	**6.26**	**0.57**
Max	**20.97**	**13.69**	**1.40**
Sierra St			
763 Sierra St	61.29	3.09	0.06
757 Sierra St	58.24	3.70	0.49
753 Sierra St	49.73	3.82	0.39
747 Sierra St	46.22	4.72	0.58
741 Sierra St	43.65	6.67	0.16
737 Sierra St	32.82	4.79	0.73
725 Sierra St	34.17	2.17	0.12
721 Sierra St	22.67	4.73	0.29
711 Sierra St	27.60	3.88	0.17
707 Sierra St	18.90	3.80	0.64
703 Sierra St	12.12	2.38	0.25
Average	**37.04**	**3.98**	**0.35**
Max	**61.29**	**6.67**	**0.73**
E Maple Ave			
612 E Maple Ave	17.76	2.35	0.83
Average	**17.76**	**2.35**	**0.83**
Max	**17.76**	**2.35**	**0.83**

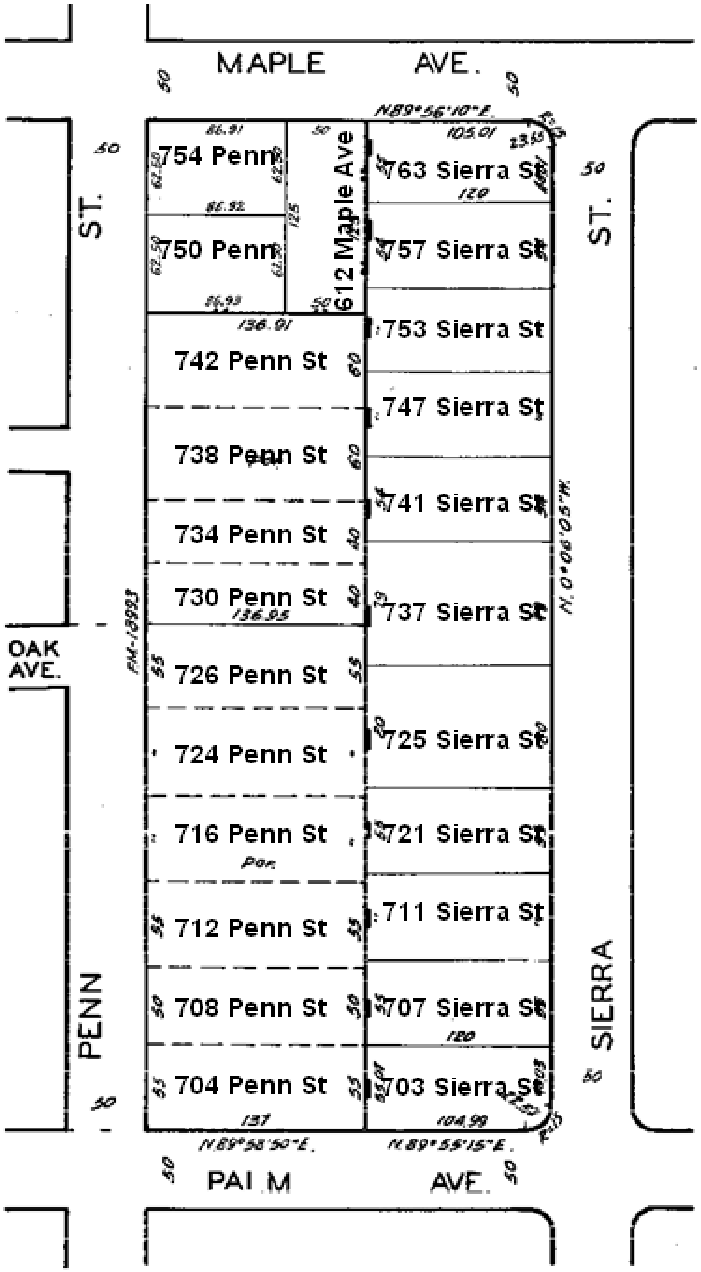


Figure A.2: Map for area geocoded in Table A.2

Table A.3: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Sierra St			
760 Sierra St	67.58	3.11	0.87
756 Sierra St	59.01	2.26	0.30
750 Sierra St	55.92	1.99	0.72
746 Sierra St	47.82	1.37	0.45
740 Sierra St	44.50	1.01	0.63
736 Sierra St	36.87	0.74	0.82
730 Sierra St	33.55	0.68	1.01
724 Sierra St	30.25	0.88	1.20
720 Sierra St	22.64	1.21	1.38
714 Sierra St	19.36	1.60	1.57
708 Sierra St	16.11	2.00	1.76
702 Sierra St	12.90	2.42	2.10
Average	**37.21**	**1.61**	**1.07**
Max	**67.58**	**3.11**	**2.10**
Lomita St			
763 Lomita St	13.78	3.62	2.97
757 Lomita St	14.02	3.02	2.85
753 Lomita St	9.70	2.35	2.93
747 Lomita St	10.43	1.90	2.87
741 Lomita St	11.69	1.39	2.91
735 Lomita St	13.00	1.05	2.95
731 Lomita St	8.48	1.06	3.01
725 Lomita St	9.86	1.40	3.08
719 Lomita St	11.28	1.91	3.16
715 Lomita St	7.01	2.48	3.24
707 Lomita St	14.15	3.09	3.34
701 Lomita St	15.61	3.71	3.53
Average	**11.58**	**2.25**	**3.07**
Max	**15.61**	**3.71**	**3.53**

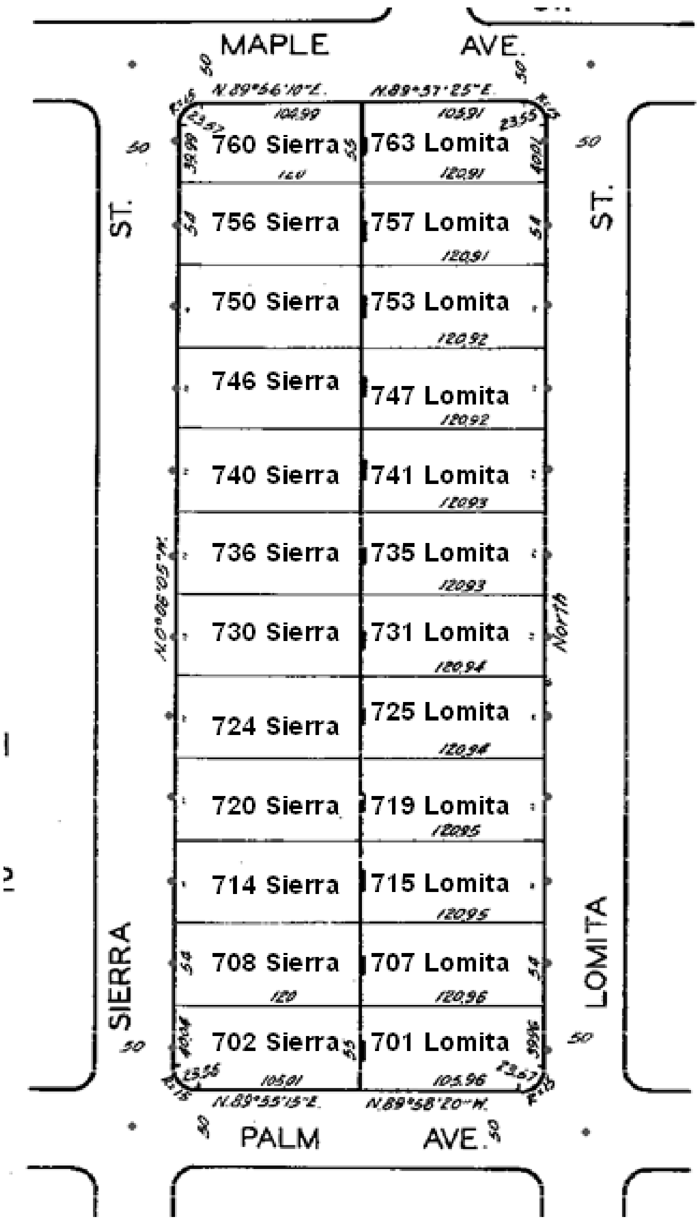


Figure A.3: Map for area geocoded in Table A.3

Table A.4: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Sierra St			
640 Sierra St	59.19	13.30	0.57
636 Sierra St	48.12	11.31	0.62
630 Sierra St	39.87	9.64	0.33
624 Sierra St	31.80	7.82	0.21
616 Sierra St	26.56	6.32	0.24
610 Sierra St	19.32	3.66	0.48
602 Sierra St	13.91	2.33	0.15
Average	**34.11**	**7.77**	**0.37**
Max	**59.19**	**13.30**	**0.62**
Lomita St			
647 Lomita St	67.56	4.91	0.21
641 Lomita St	62.22	5.89	1.00
635 Lomita St	56.11	6.90	0.82
631 Lomita St	47.70	7.92	1.46
625 Lomita St	41.57	8.91	1.11
619 Lomita St	35.91	9.92	1.08
615 Lomita St	28.07	10.94	1.73
609 Lomita St	22.94	11.94	1.54
601 Lomita St	20.37	12.97	0.66
Average	**42.49**	**8.92**	**1.07**
Max	**67.56**	**12.97**	**1.73**

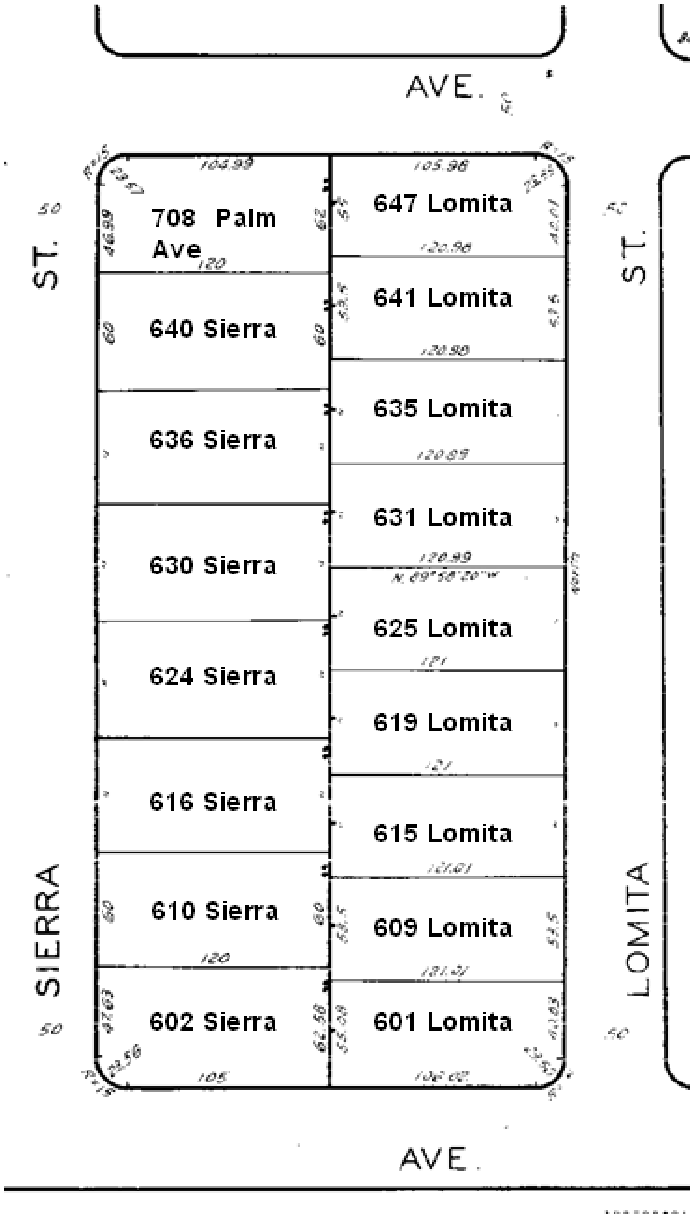


Figure A.4: Map for area geocoded in Table A.4

Table A.5: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Lomita St			
764 Lomita St	61.72	5.88	0.06
758 Lomita St	62.21	8.99	0.22
752 Lomita St	62.70	12.53	0.27
746 Lomita St	63.19	16.22	0.32
740 Lomita St	63.69	19.97	0.37
736 Lomita St	59.88	23.76	0.42
730 Lomita St	60.39	27.56	0.47
726 Lomita St	56.59	31.38	0.52
720 Lomita St	57.10	35.21	0.57
714 Lomita St	57.62	39.05	0.62
708 Lomita St	58.14	42.89	0.67
702 Lomita St	58.67	46.74	0.83
Average	**60.16**	**25.85**	**0.44**
Max	**63.69**	**46.74**	**0.83**
Maryland St			
763 Maryland St	69.29	6.36	0.06
757 Maryland St	70.51	9.95	0.22
753 Maryland St	67.18	14.26	0.27
747 Maryland St	68.40	18.81	0.32
741 Maryland St	69.63	23.45	0.37
735 Maryland St	70.86	28.14	0.42
731 Maryland St	67.54	32.86	0.47
725 Maryland St	68.79	37.60	0.52
719 Maryland St	70.03	42.35	0.57
715 Maryland St	66.73	47.11	0.62
707 Maryland St	72.54	51.87	0.67
701 Maryland St	73.81	56.64	0.83
Average	**69.61**	**30.78**	**0.44**
Max	**73.81**	**56.64**	**0.83**

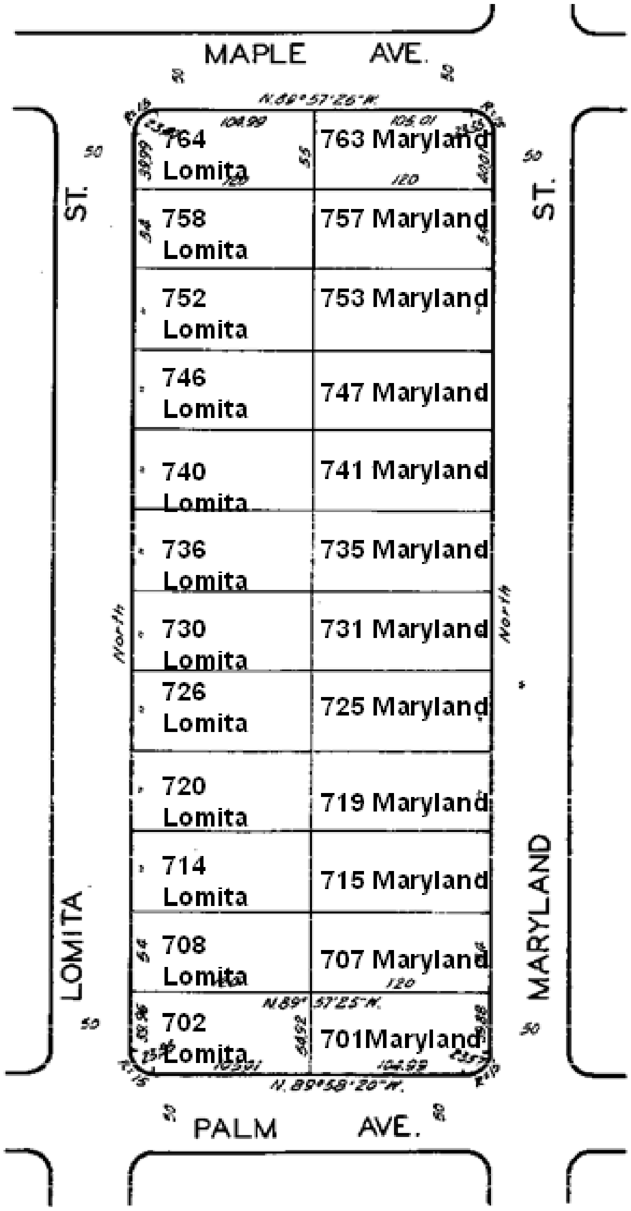


Figure A.5: Map for area geocoded in Table A.5

Table A.6: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Lomita St			
646 Lomita St	63.17	5.84	0.19
640 Lomita St	57.94	4.55	0.94
634 Lomita St	51.90	3.95	0.74
630 Lomita St	43.50	2.75	1.35
624 Lomita St	37.32	2.73	0.98
618 Lomita St	31.51	2.75	0.94
614 Lomita St	23.20	2.67	1.55
608 Lomita St	17.39	3.50	1.34
602 Lomita St	11.11	4.77	0.45
Average	**37.45**	**3.72**	**0.94**
Max	**63.17**	**5.84**	**1.55**
Maryland St			
647 Maryland St	68.29	5.46	0.19
641 Maryland St	62.41	5.23	0.94
635 Maryland St	55.70	4.18	0.74
631 Maryland St	46.83	3.95	1.35
625 Maryland St	39.96	2.73	0.98
619 Maryland St	33.41	1.84	0.94
615 Maryland St	24.55	1.62	1.55
609 Maryland St	17.84	0.57	1.34
601 Maryland St	13.29	1.31	0.45
Average	**40.25**	**2.99**	**0.94**
Max	**68.29**	**5.46**	**1.55**

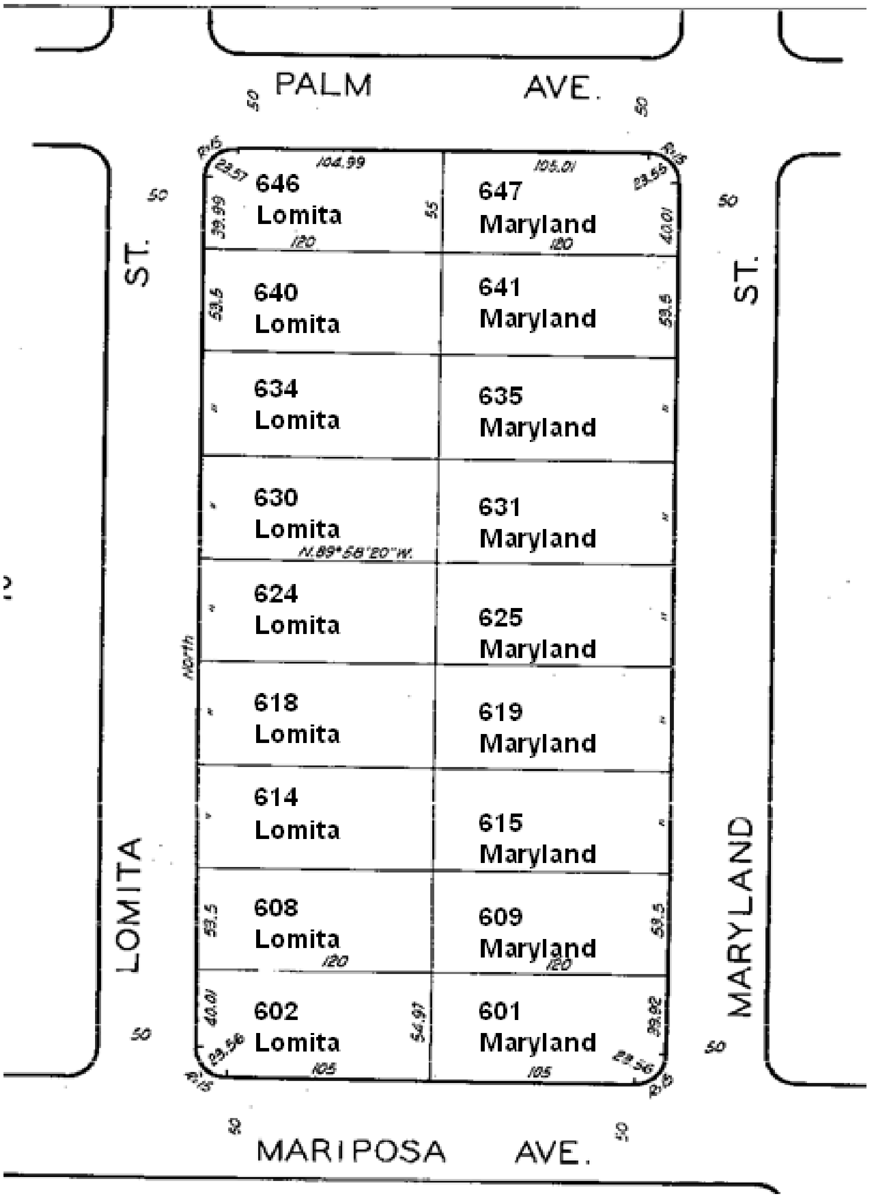


Figure A.6: Map for area geocoded in Table A.6

Table A.7: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Maryland St			
764 Maryland St	63.73	5.68	2.04
758 Maryland St	60.55	5.31	2.20
752 Maryland St	57.37	5.06	2.20
746 Maryland St	54.20	4.96	2.20
740 Maryland St	51.04	5.00	2.20
736 Maryland St	43.33	5.18	2.21
730 Maryland St	40.19	5.50	2.21
726 Maryland St	32.51	5.92	2.21
720 Maryland St	29.40	6.44	2.21
714 Maryland St	26.33	7.02	2.21
708 Maryland St	23.30	7.65	2.22
702 Maryland St	20.34	8.33	2.37
Average	**41.86**	**6.00**	**2.21**
Max	**63.73**	**8.33**	**2.37**
Bungalow Dr			
763 Bungalow Dr	64.02	9.39	2.04
757 Bungalow Dr	60.29	9.30	2.20
753 Bungalow Dr	52.28	9.22	2.20
747 Bungalow Dr	48.57	9.16	2.20
741 Bungalow Dr	44.88	9.12	2.20
735 Bungalow Dr	41.19	9.10	2.21
731 Bungalow Dr	33.31	9.09	2.21
725 Bungalow Dr	29.69	9.11	2.21
719 Bungalow Dr	26.12	9.14	2.21
715 Bungalow Dr	18.68	9.20	2.21
707 Bungalow Dr	19.18	9.27	2.22
701 Bungalow Dr	15.91	9.36	2.37
Average	**37.84**	**9.20**	**2.21**
Max	**64.02**	**9.39**	**2.37**

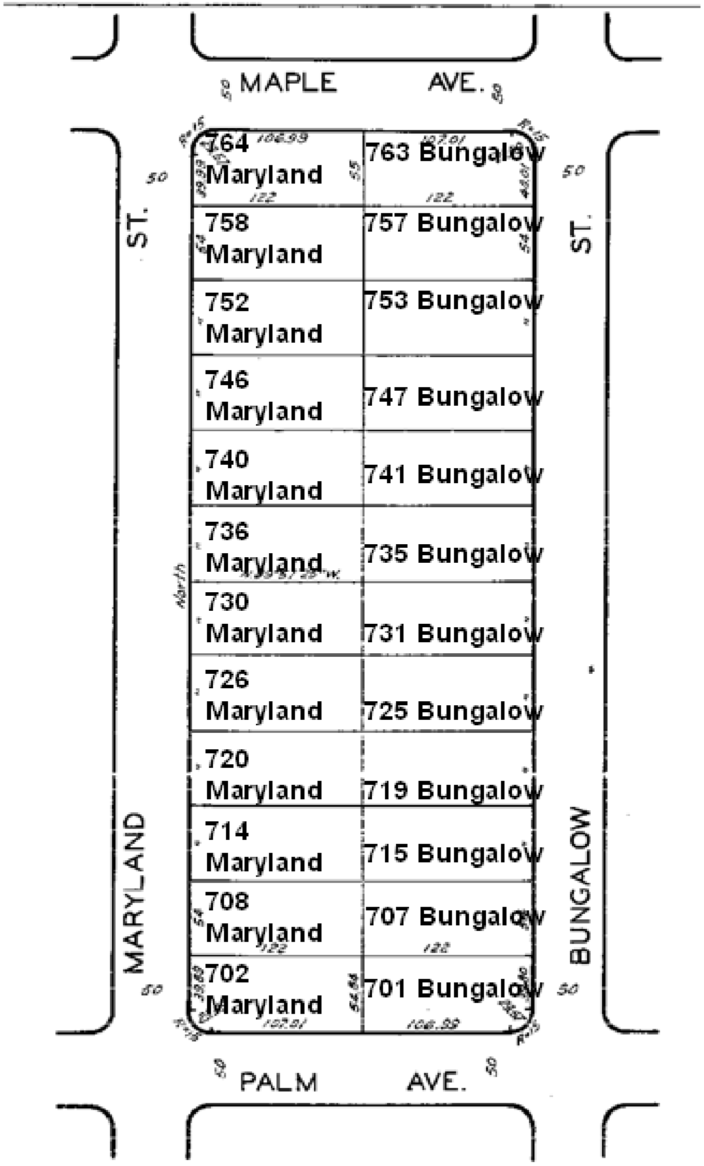


Figure A.7: Map for area geocoded in Table A.7

Table A.8: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Maryland St			
646 Maryland St	68.19	5.35	0.03
640 Maryland St	62.20	5.02	0.50
634 Maryland St	55.38	3.86	0.06
630 Maryland St	46.42	3.53	0.45
624 Maryland St	39.43	2.20	0.15
618 Maryland St	32.78	1.21	0.43
614 Maryland St	23.81	0.88	0.04
608 Maryland St	16.99	0.28	0.48
602 Maryland St	9.35	2.27	1.60
Average	**39.40**	**2.73**	**0.42**
Max	**68.19**	**5.35**	**1.60**
Bungalow Dr			
647 Bungalow Dr	63.19	5.21	2.68
641 Bungalow Dr	57.90	4.55	2.73
635 Bungalow Dr	51.78	4.60	2.68
631 Bungalow Dr	43.28	4.00	2.72
625 Bungalow Dr	37.01	4.17	2.69
619 Bungalow Dr	31.06	4.10	2.72
615 Bungalow Dr	22.60	3.57	2.68
604 Bungalow Dr	22.89	2.85	2.72
601 Bungalow Dr	12.87	4.25	3.12
Average	**38.07**	**4.14**	**2.75**
Max	**63.19**	**5.21**	**3.12**

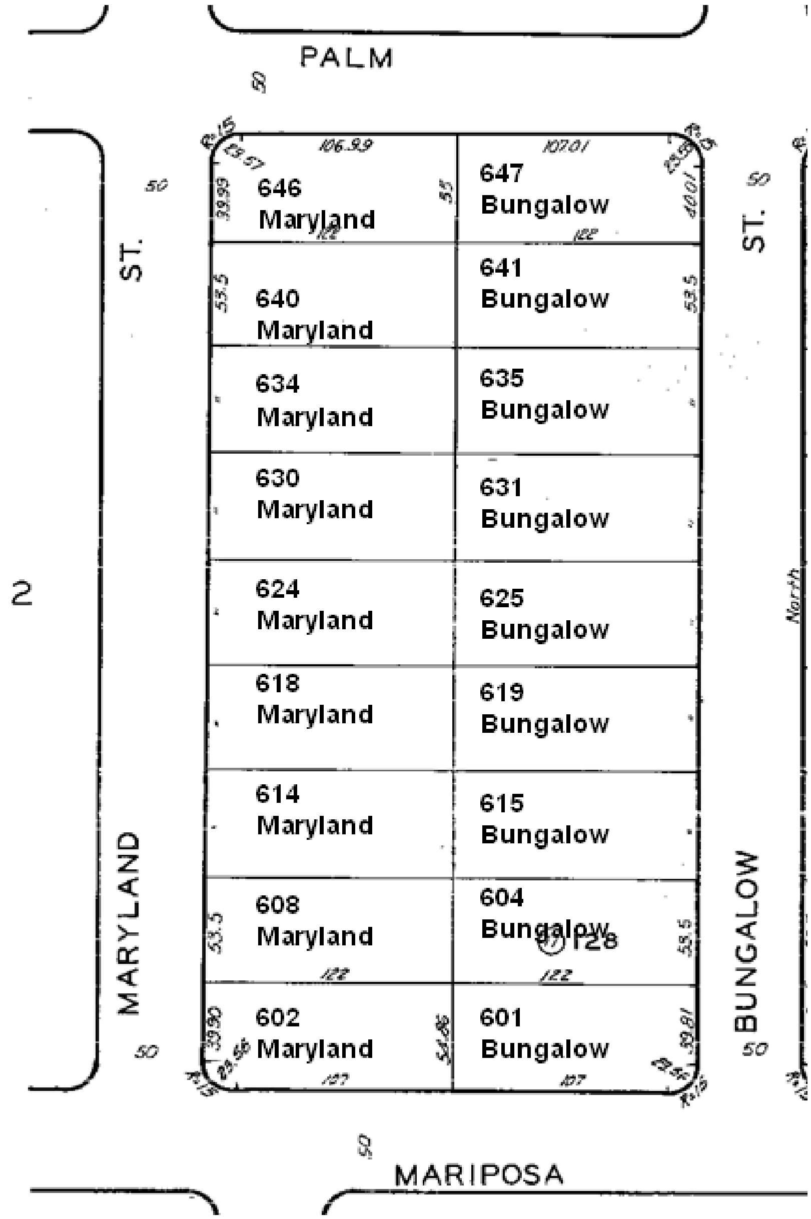


Figure A.8: Map for area geocoded in Table A.8

Table A.9: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Bungalow Dr			
764 Bungalow Dr	59.62	2.04	2.00
758 Bungalow Dr	56.17	1.94	2.16
752 Bungalow Dr	52.72	1.85	2.16
746 Bungalow Dr	49.27	1.76	2.16
740 Bungalow Dr	45.83	1.67	2.16
736 Bungalow Dr	38.03	1.59	2.16
730 Bungalow Dr	34.58	1.50	2.17
726 Bungalow Dr	26.78	1.43	2.17
720 Bungalow Dr	23.33	1.35	2.17
714 Bungalow Dr	19.89	1.28	2.17
708 Bungalow Dr	16.44	1.22	2.17
702 Bungalow Dr	13.00	1.16	2.33
Average	**36.30**	**1.57**	**2.17**
Max	**59.62**	**2.04**	**2.33**
Center St			
763 Center St	39.70	1.53	2.00
757 Center St	32.29	5.62	2.16
753 Center St	22.13	9.76	2.16
747 Center St	15.40	13.90	2.16
741 Center St	10.14	18.03	2.16
735 Center St	9.32	22.17	2.16
731 Center St	15.73	26.32	2.17
725 Center St	15.38	13.11	2.17
719 Center St	8.50	10.49	2.17
715 Center St	7.89	7.89	2.17
707 Center St	10.43	5.33	2.17
701 Center St	17.60	2.92	2.33
Average	**17.04**	**11.42**	**2.17**
Max	**39.70**	**26.32**	**2.33**

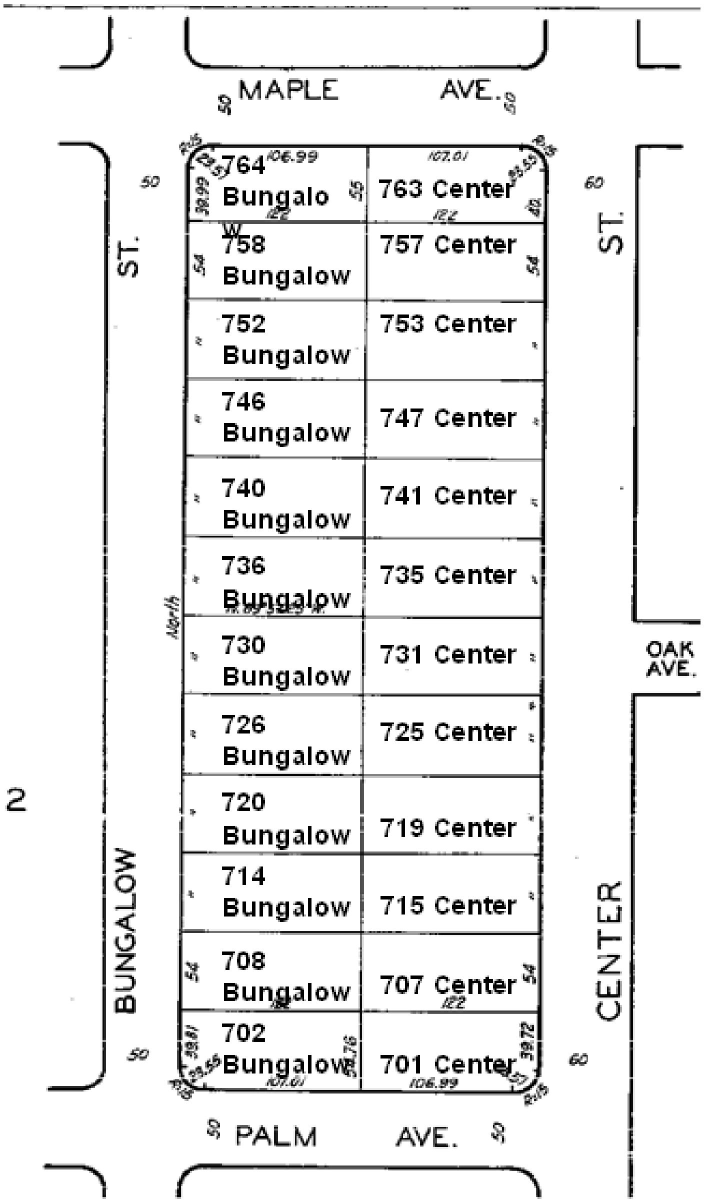


Figure A.9: Map for area geocoded in Table A.9

Table A.10: Comparison of Error

Error:	Address-range	Uniform lot-size	Actual lot-size
Bungalow Dr			
646 Bungalow Dr	67.83	0.21	0.19
640 Bungalow Dr	61.99	0.42	0.99
634 Bungalow Dr	55.30	0.21	0.77
630 Bungalow Dr	46.25	0.07	1.41
624 Bungalow Dr	39.39	0.71	1.02
618 Bungalow Dr	32.86	1.14	0.98
614 Bungalow Dr	23.82	0.89	1.62
608 Bungalow Dr	17.12	1.49	1.40
602 Bungalow Dr	9.57	2.95	0.47
Average	**39.35**	**0.90**	**0.98**
Max	**67.83**	**2.95**	**1.62**
Center St			
647 Center St	68.13	1.09	0.19
641 Center St	62.33	0.63	0.99
635 Center St	55.69	0.90	0.77
631 Center St	46.71	1.88	1.41
625 Center St	39.96	2.86	1.02
619 Center St	33.60	3.89	0.98
615 Center St	24.84	4.79	1.62
609 Center St	18.64	5.84	1.40
601 Center St	15.07	7.20	0.47
Average	**40.55**	**3.23**	**0.98**
Max	**68.13**	**7.20**	**1.62**

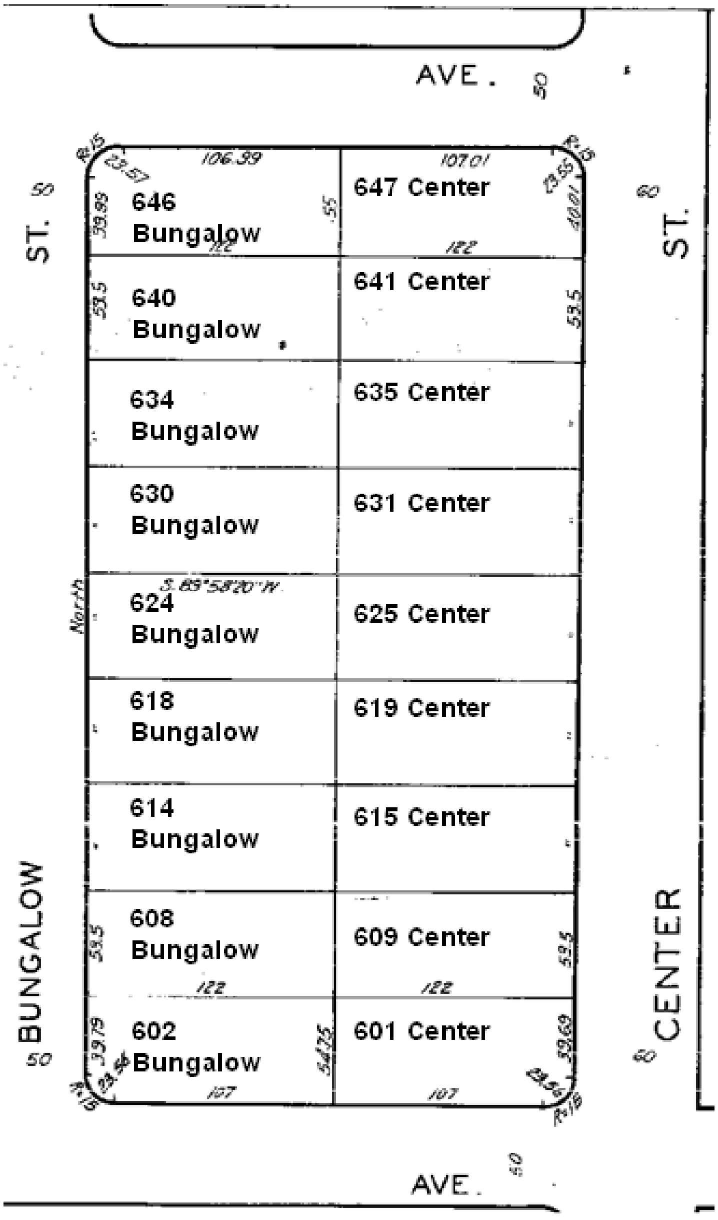


Figure A.10: Map for area geocoded in Table A.10

Table A.11: Comparison of Error

Error:	Address-range	Uniform lot-size
Oak Ave		
504 Oak Ave	9.87	1.97
506 Oak Ave	20.45	5.73
508 Oak Ave	33.87	6.76
510 Oak Ave	45.71	9.40
512 Oak Ave	55.79	13.81
514 Oak Ave	65.33	18.75
518 Oak Ave	78.76	16.98
Average	**44.25**	**10.49**
Max	**78.76**	**18.75**
E Palm Ave		
501 E Palm Ave	15.01	6.55
505 E Palm Ave	22.88	7.84
509 E Palm Ave	30.48	10.91
513 E Palm Ave	39.81	12.92
519 E Palm Ave	45.32	16.21
521 E Palm Ave	54.80	21.37
591 E Palm Ave	18.18	12.36
Average	**32.35**	**12.59**
Max	**54.80**	**21.37**

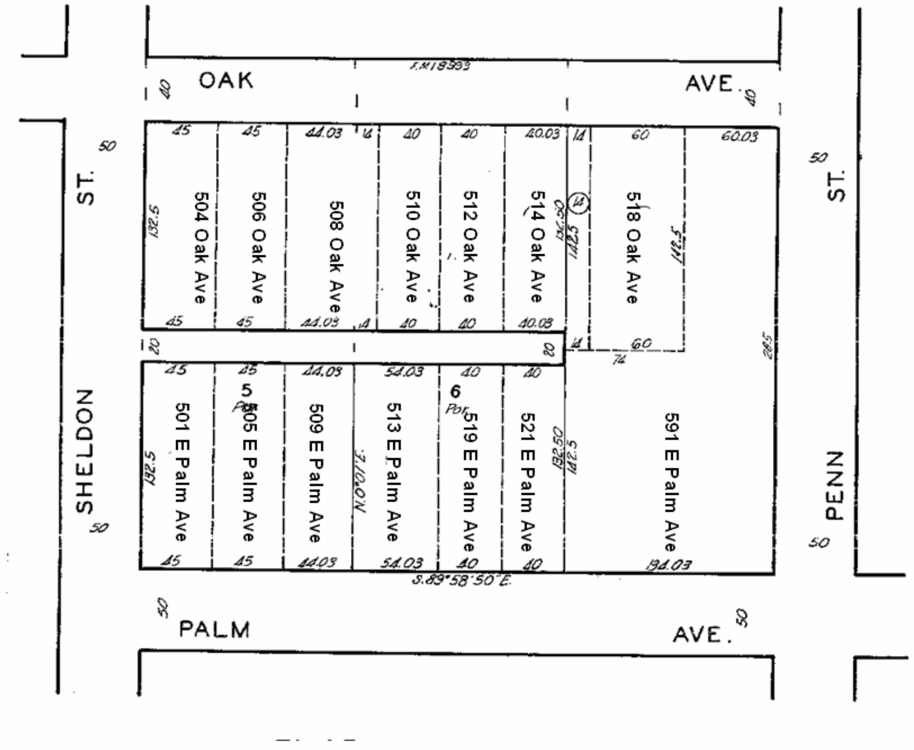


Figure A.11: Map for area geocoded in Table A.11

Table A.12: Comparison of Error

Error:	Address-range	Uniform lot-size
Sheldon St		
646 Sheldon St	69.39	11.30
640 Sheldon St	62.97	13.78
634 Sheldon St	56.24	16.11
628 Sheldon St	49.52	18.54
622 Sheldon St	42.00	20.23
616 Sheldon St	36.28	23.74
610 Sheldon St	30.27	26.95
Average	**49.52**	**18.66**
Max	**69.39**	**26.95**
Penn St		
645 Penn St	62.76	3.40
639 Penn St	55.54	0.71
633 Penn St	48.17	1.82
627 Penn St	40.63	4.19
621 Penn St	30.33	3.79
615 Penn St	22.80	6.16
609 Penn St	18.83	12.10
Average	**39.87**	**4.59**
Max	**62.76**	**12.10**
E Mariposa Ave		
535 E Mariposa Ave	4.00	15.34
527 E Mariposa Ave	33.40	13.37
525 E Mariposa Ave	24.16	6.67
523 E Mariposa Ave	14.91	26.70
517 E Mariposa Ave	9.71	24.21
511 E Mariposa Ave	3.66	22.65
501 E Mariposa Ave	21.65	4.60
Average	**15.93**	**14.35**
Max	**33.40**	**15.34**
E Palm Ave		
524 E Palm Ave	44.90	13.95
520 E Palm Ave	28.56	10.59
Average	**36.73**	**12.27**
Max	**44.90**	**13.95**

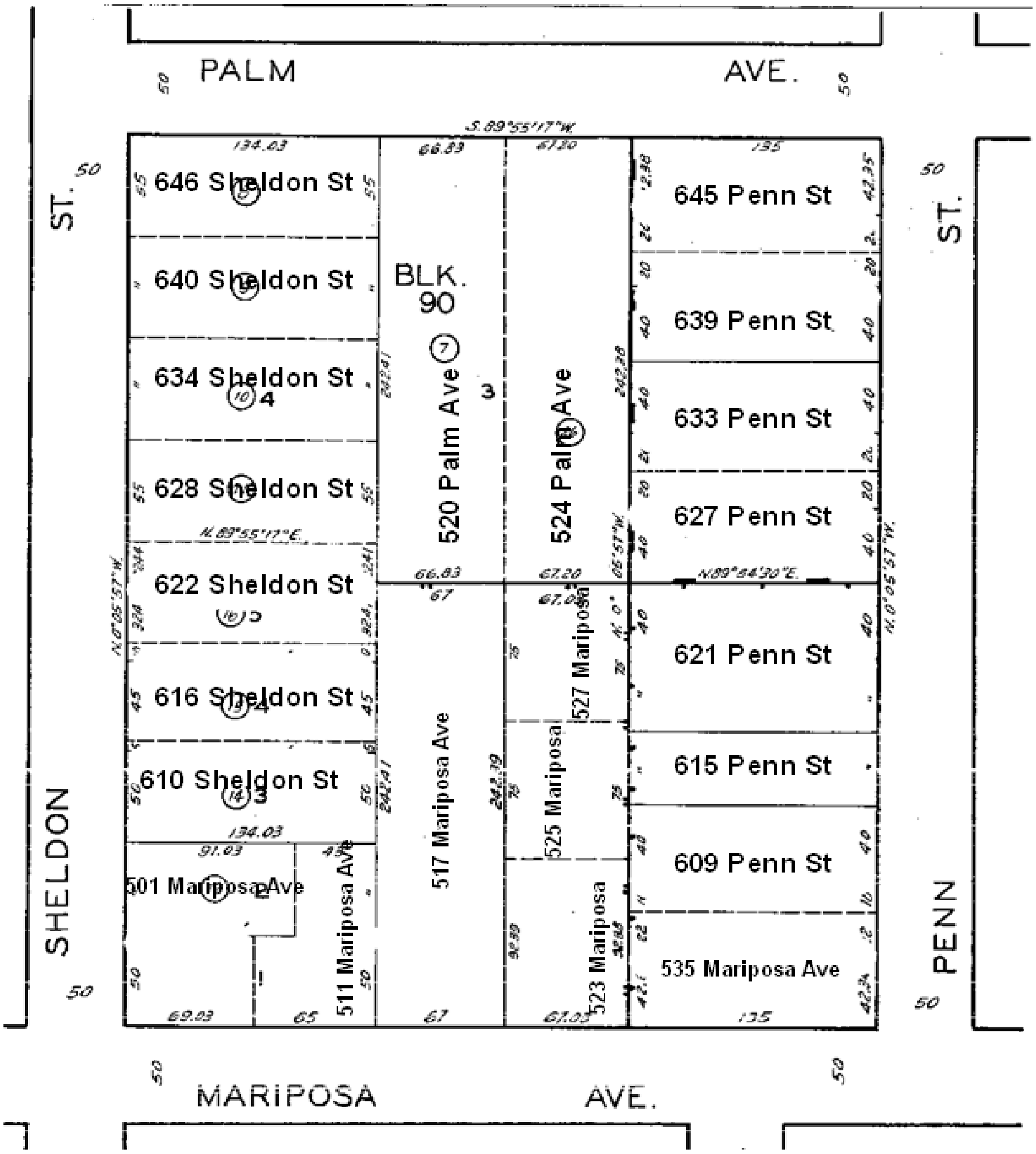


Figure A.12: Map for area geocoded in Table A.12

Table A.13: Comparison of Error

Error:	Address-range	Uniform lot-size
Oak Ave		
529 Oak Ave	81.27	2.41
527 Oak Ave	63.58	7.41
523 Oak Ave	54.32	6.92
521 Oak Ave	46.98	1.70
519 Oak Ave	39.10	2.99
517 Oak Ave	30.71	7.16
511 Oak Ave	25.16	8.52
501 Oak Ave	18.26	2.95
Average	**44.92**	**5.01**
Max	**81.27**	**8.52**
Sheldon St		
760 Sheldon St	36.93	3.61
754 Sheldon St	31.39	6.83
750 Sheldon St	23.21	10.06
744 Sheldon St	17.99	13.60
740 Sheldon St	9.97	16.99
Average	**23.90**	**10.22**
Max	**36.93**	**16.99**
E Maple Ave		
536 E Maple Ave	72.06	12.13
526 E Maple Ave	66.03	19.65
522 E Maple Ave	50.64	26.32
514 E Maple Ave	40.90	32.99
Average	**57.41**	**22.77**
Max	**72.06**	**32.99**
Penn St		
745 Penn St	10.30	25.75
741 Penn St	22.54	10.74
Average	**16.42**	**18.25**
Max	**22.54**	**25.75**

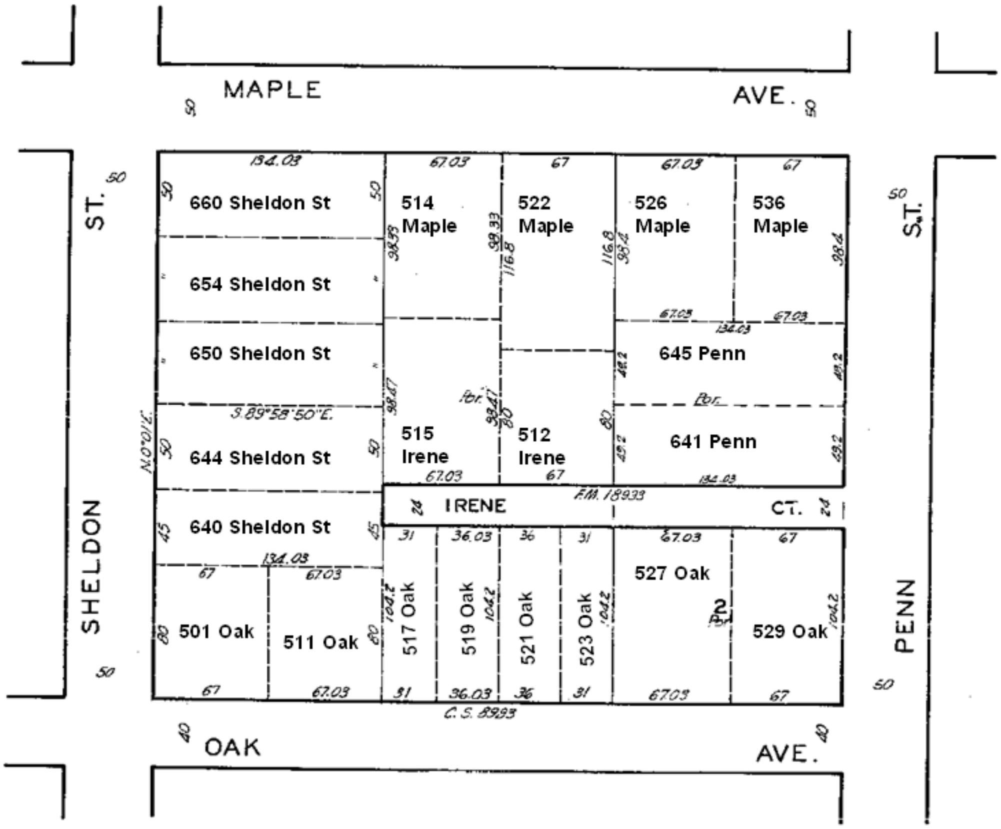


Figure A.13: Map for area geocoded in Table A.13