Suitability of the Quadtree as a Spatial Index for Soundscape Maps

by

Jaime Lee Ka’iulani Lai Man Kiyoko Pier

A THESIS

Presented to the Department of Computer & Information Science
and the Robert D. Clark Honors College
in partial fulfillment of the requirements for the degree of
Bachelor of Science

June 2017

An Abstract of the Thesis of

Jaime Pier for the degree of Bachelor of Science
in the Department of Computer & Information Science to be taken June 2017

Title: Suitability of the Quadtree as a Spatial Index for Soundscape Maps

Approved: ______________________________________

Prof. Michal T. Young

When developing an application for blind and visually impaired users, additional features must be provided for that user to extrapolate the same data as a sighted user. The adaptation of thematic maps for blind and visually impaired users requires two major considerations: a way to convey trends that does not require sight (e.g. using sound), and a data structure that allows the map to respond quickly enough so that the user, if interacting with a touchscreen interface or similar, can drag their finger across the screen and receive information that corresponds to the location of their finger in real time.

This project evaluates the quadtree as a data structure that will these requirements, by testing the size of the quadtrees created by large amounts of data and the retrieval time of a node, given these trees. It shows that a quadtree structure allows objects to be found quickly and creates a tree that is small in relation to the size of the graphical canvas and the number of objects depicted, concluding that the quadtree is a reasonable data structure for this type of data.

Acknowledgements

I would like to thank Professors Young and Childs for their help in researching this topic and implementing the program I developed for it. Thank you for believing in my ability to complete this project and for continuing to push me to do my best. Thank you also to Professor Southworth and Kathleen Freeman for their support and advice over the course of my college career. This was not the journey I thought I would have, and without your encouragement and support, I would not be completing this project.

Thank you to everyone who has prayed for me over the last five years: the Trinity House girls, my CCF Life Group Leaders, the 1824 Girls, Moms in Prayer, Wives in Prayer, and the Friday Men’s Bible Study. A special thank you to Mr. & Mrs. George, Mr. & Mrs. Lopez, Mr. & Mrs. Walters, Mrs. Moore, Mrs. Adent, Mrs. Adkison, Mrs. Harris, Mrs. Keil, Mrs. Layton, Mrs. Shuholm, Mrs. Stockamp, Joe Tepe, Lisa Burton, and Becca Frost. I would not be here without all your prayers and support.

Finally, thank you to my parents, who have never given up on me. You have prayed with me and for me, continually supported me, and given me all the kicks in the pants and the wake-up calls that were necessary to get me to this point. Thank you for all the hours you’ve spent helping me with my homework, talking on the phone with me as I walk to class, celebrating every high and coaching me through all the lows. I love you.

Table of Contents

List of Accompanying Materials

1. Program Source Code - https://github.com/jpier7/chcthesis

List of Figures

List of Tables

1. Introduction

When depicting data graphically, sighted users are able to view trends and overviews of the data represented. For those with limited or no sight, these trends must be represented in an alternate way, and an application that allows blind or low vision users to utilize a map of trends will require the consideration of additional parameters and the accommodation of extra features.

In addition to adding features and parameters to an existing application, it is also important to look at how the data, along with these features, will be stored. There are many different data structures that can store data efficiently; those used to select data based on location are called spatial indexes. In particular, this paper explores the use of the quadtree data structure and its use as a viable data structure for storing this type of data.

1. **Terms & Conventions**

In this document, the term “blind and visually impaired users” will be used most often to refer to those users with either low vision, residual vision, or no vision. When referencing participants from outside studies, whichever terms were used in the original document will be used and explained. This project is designed with the blind user (no residual vision of any kind) in mind, but it can be used by those with low or residual vision.

For the purposes of this paper, the term “map” is used to refer to a representation of geographical area.

2. Background Research

1. **Introduction**

In order to develop an application that will allow a blind or visually impaired user to “view” the trends of a dataset, first we must understand the geographical and technical aspects of such an application.

Section 2 discusses the different types of maps used by geographers to display data, primarily the use of thematic maps in contrast to navigation maps. Section 3 covers two main types of adaptive maps for blind and visually impaired users, tactile maps, and soundscape maps. These are the types of maps that will be represented by the data structure. Section 4 covers the most complete version of a soundscape map, iSonic, and its subsequent variation for use with weather data as opposed to population data.

Sections 5 and 6 cover the software and data structures generally used to create a thematic map, and an overview of the quadtree, a potential data structure that we could use for representing geographical data.

2. Visual Maps

2.1 Thematic Maps – Choropleth & Isopleth

Any map that displays data that is related to geographic area is called a thematic map. A choropleth map, as defined by Esri, a prominent geographic software company, is “a type of thematic map in which areas are distinctly colored or shaded to represent classed values of a particular phenomenon” [1]. Basically, this refers to maps in which bounded areas (e.g. countries, states, or counties) are shaded or colored to represent the data. Data represented by choropleth maps can include population density, voter or election data (e.g. states or counties who tend to vote Democrat or Republican), or racial distribution (e.g. representing the most prevalent minority race per state).

A choropleth map represents data within the confines of a certain region. In contrast, an isopleth or isoline map uses the data to define the regions, using a single line to join the points of commonality. One common use of isopleth maps is in displaying weather, as seen in Figure 2.2 below.

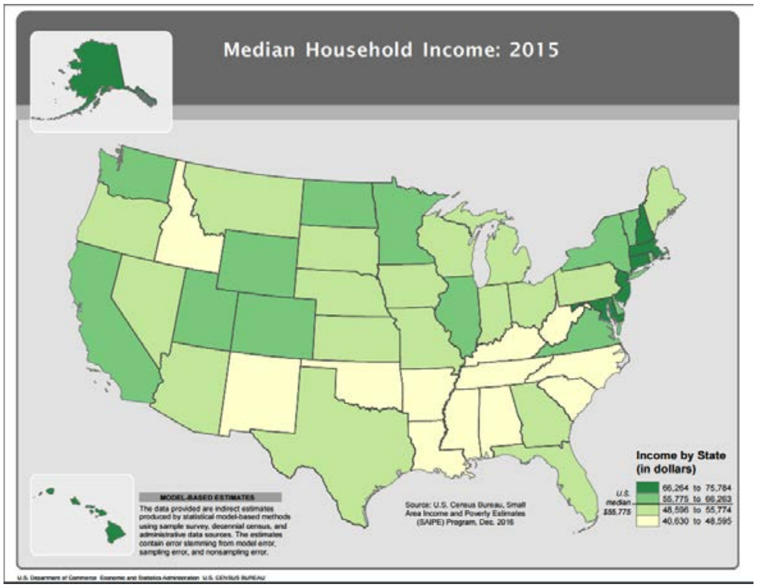

Median Household Income: 2015
Income by State
(in dollars)
U.S. median $55,775
66,264 to 75,784
55,775 to 66,263
48,596 to 55,774
40,630 to 48,595
MODEL-BASED ESTIMATES
The data provided are indirect estimates produced by statistical model-based methods using sample survey, decennial census, and administrative data sources. The estimates contain error stemming from model error, sampling error, and nonsampling error.
Source: U.S. Census Bureau, Small Area Income and Poverty Estimates (SAIPE) Program, Dec. 2016
U.S. Department of Commerce Economic and Statistics Administration U.S. CENSUS BUREAU

Fig. 2.1 A Choropleth Map (U.S. Census Bureau)

A choropleth map created by the U.S. Census Bureau illustrating the median household income by state in 2015. Each state is colored in a different shade of green, with lighter green representing a lower median income, and darker green indicating a higher median income. [10]

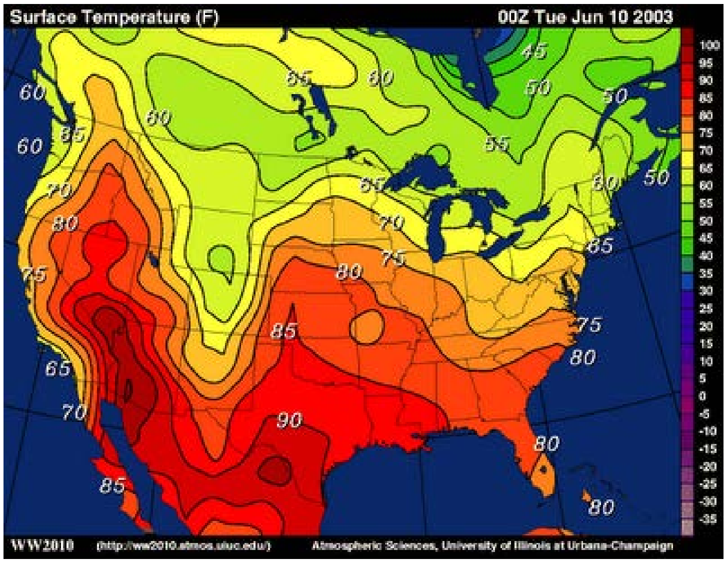


Fig 2.2 An Isopleth Map – Weather Data

An isoline or isopleth map displaying weather data. In an isopleth map, the data defines the regions—in this instance, which areas have a higher temperature, or a lower temperature. The boundaries of the states are depicted for reference only; they do not define the boundaries of the data. [5]

Visual choropleth maps provide a concise way to represent certain types of data, however they present a unique set of challenges when adapting to a visually impaired or blind user due to their reliance on colors, shading, and other visual properties.

When trying to view trends, a sighted user will generally prefer looking at the map to a table of data values. A blind or visually impaired user cannot make use of the advantages provided by looking at the map, and communicating the data to the user is more like looking at the table of values than extrapolating trends from the map.

2.2 Navigation Maps

A navigation map is one used mainly to guide a user from one destination to another. Navigation maps provide an overview of a geographic area, including streets, buildings, and other landmarks a user might need (such as rivers, bridges, walking paths, or public transportation lines).

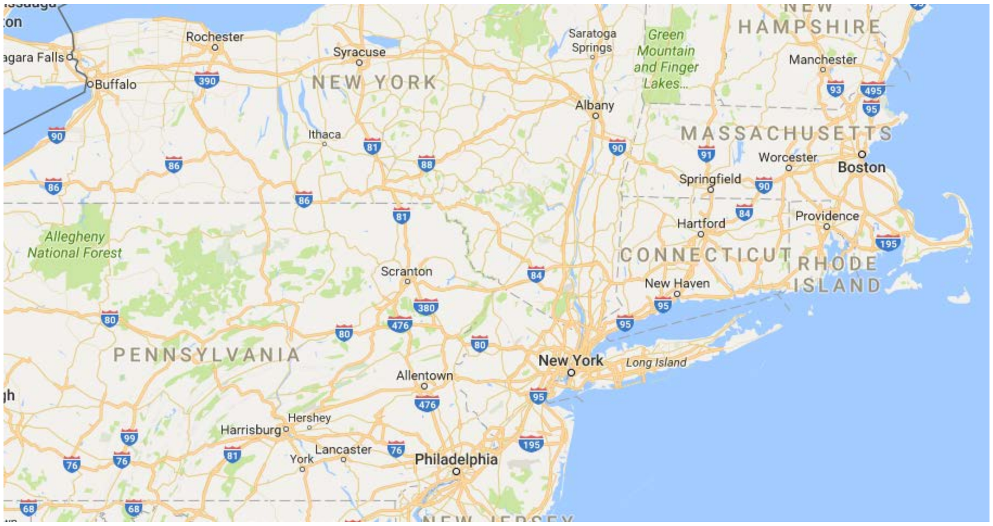


Figure 2.3 – Google Maps

This section of Google Maps containing Pennsylvania, New York, Connecticut, Rhode Island, and Massachusetts shows many attributes, however the most prominent feature is the highway system. [3]

Navigation maps are used primarily to guide the user from one place to another, although the highways and roads are not the only features on the map. In contrast, a thematic map places one attribute of higher importance, and shows only that attribute in relation to the geographic map. For example, in the choropleth map (Figure 2.1) above, the median population is the most important data; likewise in the isoline map (Figure 2.2) the temperature is the important attribute.

3. Adaptive Maps

In general, if a user cannot access an item because they lack full access to one sense, then the item must be adapted so that the user can access it through the substitution or augmentation of another sense. For instance, when adapting a visual map for a blind user, the goal is to use other senses (most notably sound and touch) in order to relay the same information or trends, as accurately as possible, as would be known to a sighted user.

3.1 Tactile Maps

One way to adapt maps for blind users is by creating a tactile map, which represents the information on the map by creating a raised surface. These raised surfaces can create symbols that represent roads, intersections, or buildings, and allow the user to explore the map using touch rather than sight or sound.

There are several challenges when it comes to producing and using tactile maps: First, tactile maps can be expensive to produce, due to the usage of raised visual elements and the quality needed for a blind user to be able to read the map successfully. One method of printing tactile maps, which uses microcapsule paper, is relatively inexpensive to produce (several hundred dollars for equipment and materials, versus several thousand). The microcapsule paper can be fed through an inkjet printer, and is then exposed to heat—the heat reacts with the chemicals in the paper as well as the black ink, and creates the raised relief [9].

Second, because tactile maps must be printed, they can only represent a fixed area and scale. If we want to see a larger area, or zoom in closer to a particular area, we have to print more maps. So even with relatively inexpensive technology, we may still make up the cost of the printing with the quantity of maps we have to print.

Third, the standardization of tactile map symbols is relatively new and not yet widely used, so a user will have to learn each map’s symbols individually, and cannot use symbols he or she recognizes from other maps to mean the same thing as on a map they are currently navigating. A paper presented by Pat van Geem of the Visual Impairment Educational Consultant in Texas, referenced Prof. Lobben’s set of symbols, which uses two parallel lines to represent a railroad. On the other hand, the Braille

Authority of North America uses a solid line with vertical cross-hatching to represent a railroad and discourages the use of double lines unless indicating a boulevard (two separate streets) [15]. The figures below illustrate the differences between symbols used by Prof. Lobben, and those covered by Mr. van Geem.

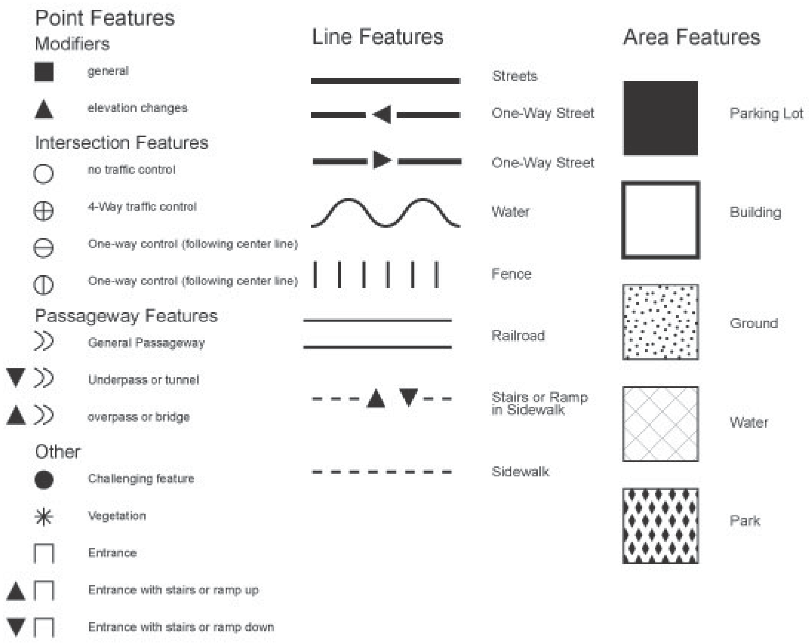
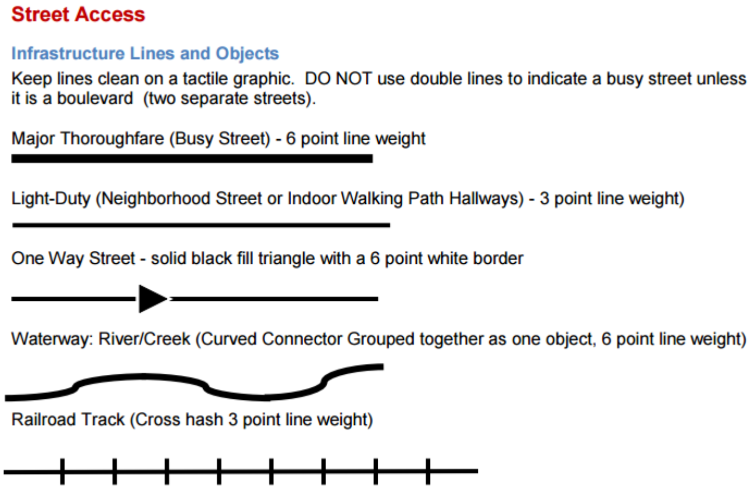


Figure 3.1 – Tactile Symbols, Prof. Lobben

The symbols used by Prof. Lobben in her tactile maps. Note the symbol for railroad, which consists of two parallel lines. [9]



Street Access

Infrastructure Lines and Objects

Keep lines clean on a tactile graphic. DO NOT use double lines to indicate a busy street unless it is a boulevard (two separate streets).

Major Thoroughfare (Busy Street) - 6 point line weight

Light-Duty (Neighborhood Street or Indoor Walking Path Hallways) - 3 point line weight)

One Way Street - solid black fill triangle with a 6 point white border

Waterway: River/Creek (Curved Connector Grouped together as one object, 6 point line weight)

Railroad Track (Cross hash 3 point line weight)

Figure 3.2 Tactile Symbols – Patrick van Geem

A portion of the symbol overview from Pat van Geem’s talk, which uses a cross-hatched line to symbolize a railroad track. Note the warning at the top restricting the use of double lines.

Additionally, thematic maps can be difficult to represent using tactile maps. Shading is much different than representing roads, buildings, or intersections, and variances in the shading (e.g. spacing of the fill pattern) can be difficult to detect by touch. Prof. Lobben, in a study conducted with Megan Lawrence, examined the use of tactile map symbols in thematic maps. Lobben and Lawrence looked at whether users could differentiate between patterns of raised fill dots on a thematic map. In this instance, the distance between the dots represents the gradation found in visual choropleth maps, and since population was used as the data set, dots closer together represented a higher population, and dots spaced farther apart represented a lower population. Lobben and Lawrence also specified that though the distance between the dots was varied, the size of all the dots were the same [8].

3.2 Soundscape Maps

Another way to allow blind or visually impaired users to experience an overview of a trend is by using a soundscape map. A soundscape, essentially a landscape of sound, is an aural representation of the data and can allow a user to perceive data trends more easily. In a soundscape map, we assign a different sound or pitch to each category of data, and then the sounds are overlaid onto the map. Different sounds can be used to indicate to a user that they have crossed some sort of boundary (e.g. between counties, between states, or into a body of water), or to indicate the end of a section (in a case where the user only wants to hear the trends in a certain area of the map). Varying pitch is the most often used attribute of sound for depicting trends aurally, as pitches have a set order in which they occur, and it is relatively easy to distinguish one pitch from another. When a user hears this combination of different sounds and pitches, they can form a mental image of the trends represented on the map.

Although a soundscape map can help blind and visually impaired users recognize trends more easily, it must be paired with a speech component to read raw data to the user in order to attempt to recreate the same effect a visual map has for a sighted user—much like using labels on a map.

4. iSonic

The most complete program used to adapt choropleth maps for visually impaired users is a program called iSonic, developed by researchers in the computer science department of the University of Maryland as well as a professor of computer and information sciences at Towson University [18]. Researchers Haixia Zhao, Catherine Plaisant, Ben Schneiderman, and Jonathan Lazar designed iSonic to address challenges in making geospatial data, specifically data provided by the U.S. government, accessible to blind and visually impaired users.

4.1 Development

Zhao, Plaisant, and Schneiderman utilized several already-researched concepts for their program, including the use of structured sounds (known as earcons) and spoken information.

They also utilized a feedback loop they called an Auditory Information-Seeking Action (AISA), in which the user gives the program a command through the interface and listens to the auditory response. Then, the user can build a mental representation of the data by continuing to issue commands and receive responses from the program.

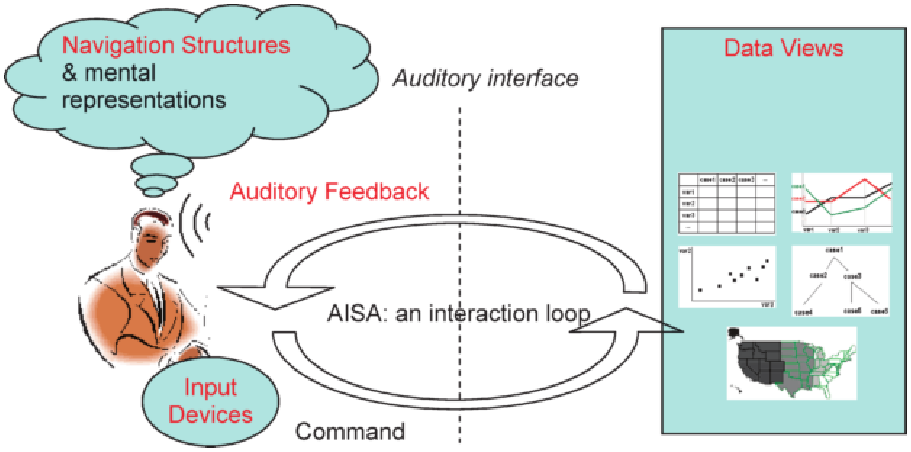


Figure 4.1 - The AISA Feedback Loop

Visual of the AISA feedback loop used by the creators of iSonic. [cite]

4.2 Interface & Navigation

iSonic's main data views are a table of values and a choropleth map. Both views utilize speech output as a primary means of conveying the data to the user, and the choropleth map also uses earcons to give the user a larger picture of the data represented. These earcons consist of a violin sound in various pitches, a bell sound, and a percussion sound. Each state is assigned one of approximately five violin pitches, which represent the relative population of each state. A higher pitch indicates a higher or more dense population, and a lower pitch a lower or less dense population. The percussion sound is used to indicate that a border has been crossed, which is helpful in the event that two states fall under the same pitch category. A chirp earcon is used to indicate that the user is off the map, or has crossed from a contiguous state to a noncontiguous state, such as from California to Hawaii.

If the user wants to read off every state in succession, the states are listed from right to left, and top to bottom in rows. Since the map of U.S. states is not arranged in exact rows or columns, the researchers approximated and grouped the regions as indicated in the figure below. Note that Hawaii is placed towards the bottom left side of the display and Alaska near the top left, thus Hawaii is included in the last row. In a complete sweep of all the states, the percussion sound indicates the end of a row, and the bell indicates the end of the sweep [6].

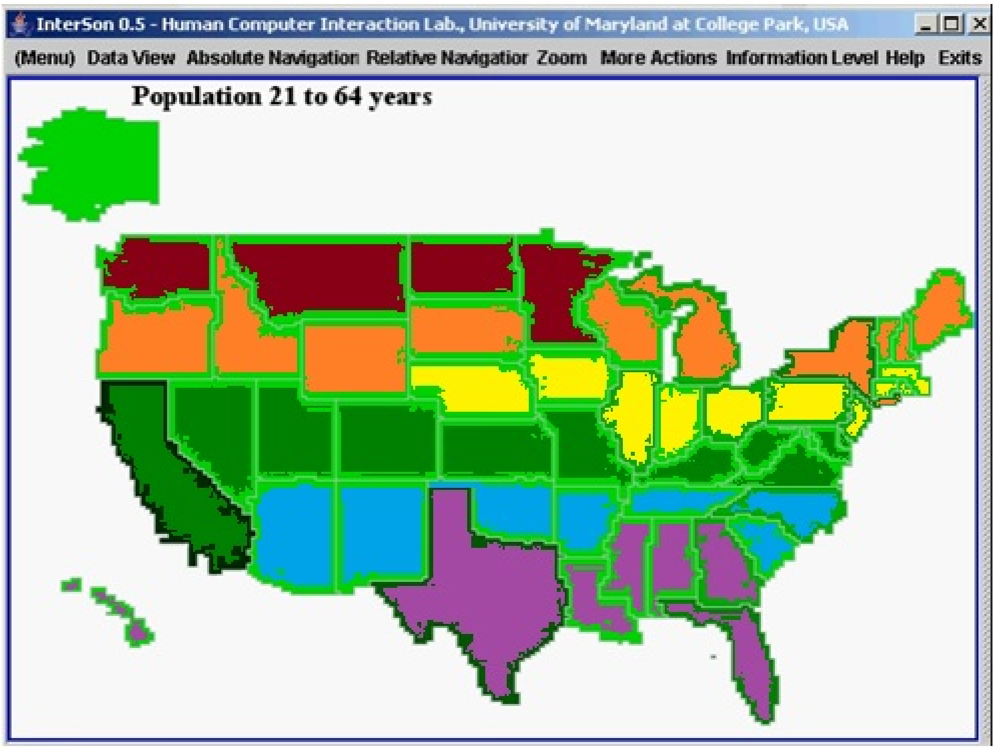

InterSon 0.5 - Human Computer Interaction Lab., University of Maryland at College Park, USA
(Menu)
Data View
Absolute Navigation
Relative Navigatior
Zoom
More Actions
Information Level
Help
Exits
Population 21 to 64 years

Figure 4.2 - iSonic Display

An edited version of the iSonic display, with each row colored according to how it is listed to the user. Note how some rows do not start on the west coast of the United States. [6]

Stereo panning is also used to help the user create a mental representation of the data, with 0 for the leftmost region(s) and 127 for the rightmost region(s). During testing, the users wore headphones to better hear the panning. For instance, if the user wanted to hear the sound associated with California, they would hear it in their left ear only. If they wanted the sound associated with Maine, they would hear it in their right ear. This assist the user in creating their mental image of the map, as they know by the sound whether a region is to the left, to the right, or close to the middle of the map.

The data is presented in four different levels: the first level contains sound only (no spoken information), which can help a user create a mental representation of the shading a sighted user takes in visually. The second level contains the sound associated with that state as well as the name of the state. In the third level, the user also gets an associated data value for that state—for instance, the number of people aged 21 to 64 in that state. A zeroth level includes only the state name, with no sounds.

When iSonic was initially developed, prior to 2008, it was designed for use with a keyboard only, or with the addition of a touchpad. The user can navigate the map with the number keypad, allowing for finer movement around the map, and use other strategically mapped keys (such as ‘I’ for information, or ‘L’ for lock/select) to gather more information from the map.

As seen in the figure below, the researchers also divided the map into a 3x3 grid, which they used for zooming in on the map. The grid can be controlled using the numeric keypad on the keyboard—when that number key is pressed, the sounds for the corresponding region are read off. This can also be used recursively—after zooming in to region 6, for instance, using the 9 key will read off only the northeast section of the zoomed in region.

In addition to the map view, iSonic also contains a table view. In this view, the user can hear raw data from several different variables as well as pitches associated with the data. The table view also features sorting, which can aid the user in finding the highest or lowest value for a given variable.

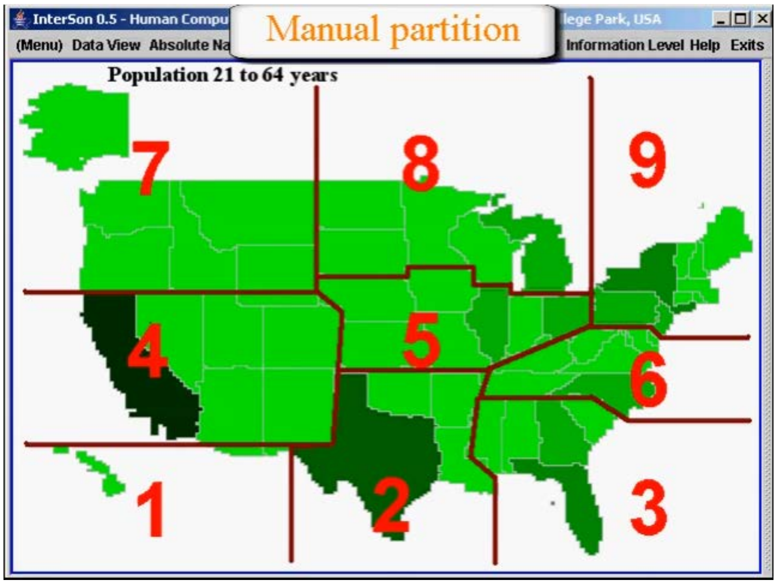

InterSon 0.5 - Human Compu
(Menu) Data View Absolute Na
Manual partition
lege Park, USA
Information Level Help Exits
Population 21 to 64 years
7
8
9
4
5
6
1
2
3

Figure 4.3 - Division of Display for Zooming

The manual partitioning of a map of the United States in the iSonic interface. The sections are controlled by using the numeric keypad on a keyboard, and can be used recursively to zoom in and read smaller sections of data [cite]

4.3 Testing

Initial user studies with iSonic were conducted in the summer of 2005 with seven local blind test subjects. All participants had computer experience and none had residual vision or were newly blind. Participants were given a brief training session on using iSonic, were then asked to perform several tasks, and finally, were allowed to explore the map freely using the keyboard controls. The majority of participants were able to complete the tasks successfully and found navigating the map relatively intuitive.

4.4 2013 Study with Weather Data

In 2013, one of the original iSonic developers Jonathan Lazar, along with colleagues from Towson University, also in Maryland, used iSonic as the basis for a project working to convey weather data to blind and visually impaired users [7].

When adapting the iSonic program for use with weather data, there are several additional parameters that needed consideration: first, the original iSonic program used population data, which is a static dataset. However, weather data requires frequent updating, and so the program must be adapted to accommodate a real-time weather feed.

Second, weather data is most often depicted as an isopleth map—that is, the data defines the geometry (such as an area of rain) rather than data bounded by geometry (such as population by state). In this case, the researchers chose to stay within the boundaries of the original iSonic program and modified their dataset to represent a larger area. The original data was taken from a representative city (usually the administrative center of the county) and used to represent the entire county—this was not considered to affect the integrity of the data too much, since the counties are not very large and have similar climate patterns throughout.

Lazar and his colleagues tried to keep as many of the features of the original iSonic, which included its grid-like nature. Although the data, and their usage of the state of Maryland does not fit as well into a row and column format as a map of the U.S., the modified iSonic still reads off the counties and their information in a west to east, north to south order.

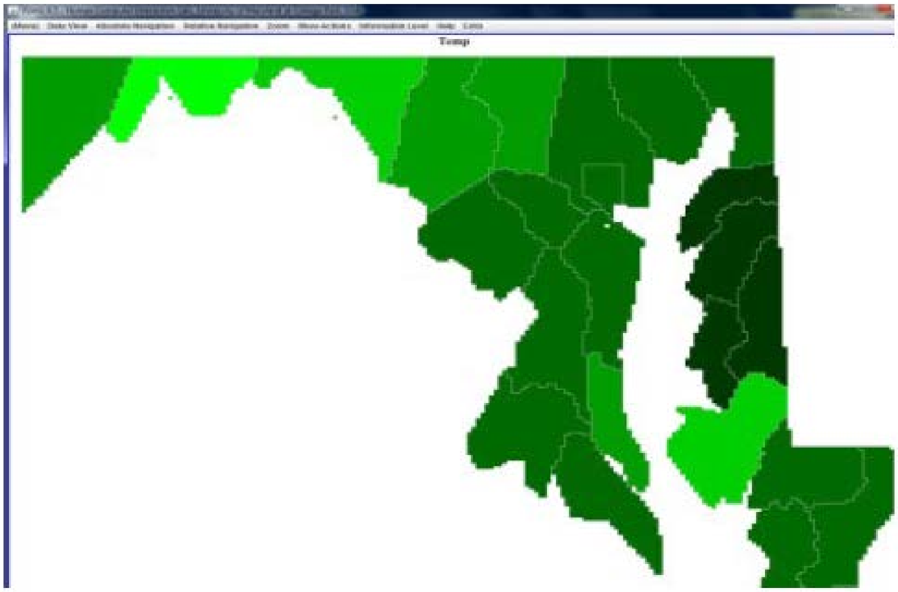


Figure 4.4 - Updated iSonic Display of Maryland

The updated iSonic program, illustrating weather data for Maryland; this data is harder to read off in a left to right, top to bottom order. (Lazar)

Lazar and the other researchers from Towson used the Web Content Accessibility Guidelines (WCAG), which provides a clear set of standards for adapting visual images, to inform their approach to this updated version of iSonic. Instead of a keyboard as a main control, Lazar and his colleagues also modified iSonic to be controlled via a touchscreen with a tactile map overlaid, indicating the boundaries for counties within the state of Maryland. This approach was useful since the map only represented Maryland; it would not be as effective with the original iSonic, which allowed a user to zoom in to different regions of the United States (thereby changing the boundaries displayed). Although the tactile map assisted in navigation of the map, the sounds were still necessary to help create a mental representation of trends on the map.

Although the team ran into issues with the technology during testing, all participants were able to navigate the map easily and were able to learn information from it. Some participants preferred the speech output to the sonification tones, but those who liked the sonification were excited to extrapolate trends they weren't able to sense before.

Lazar and his team also looked into another mapping software called EarthPlus, but it was only in its prototype stage and does not exist anymore. While iSonic is a nearly complete model for representing choropleth map data for visually impaired and blind users, the University of Maryland has not continued work on the project since its initial implementation. With touchscreen technology becoming largely accessible and in larger screen sizes, a new study could be conducted to analyze blind and visually impaired users' comfortability with a touchscreen (without a tactile map overlay, which was used with both the touchpad in the original iSonic and the touchscreen in the 2013 Lazar study) and adapt iSonic to be used with an Android or iOS tablet interface.

5. GIS software

GIS, or geospatial information system, is a tool used by geographers to visualize, analyze, and interpret data [16]. GIS software generally represents data in two ways: a vector model and a raster model.

A vector model represents data as a series of points and lines, or vertices and paths. A vector model uses a keyword (usually point, line, or polygon) followed by a series of coordinates which indicate where the paths and vertices lie on the map [2]. The figure below illustrates the difference between a raster and vector model of data. For the purposes of this project, the main focus is on storing and retrieving information from a vector model of data.

In contrast, a raster model is used to represent data that varies continuously, such as a satellite image or an aerial photograph [2]. The raster model uses pixels to display the data. A file of a raster model will generally contain coordinates in rows and columns georeferenced to the earth’s surface.

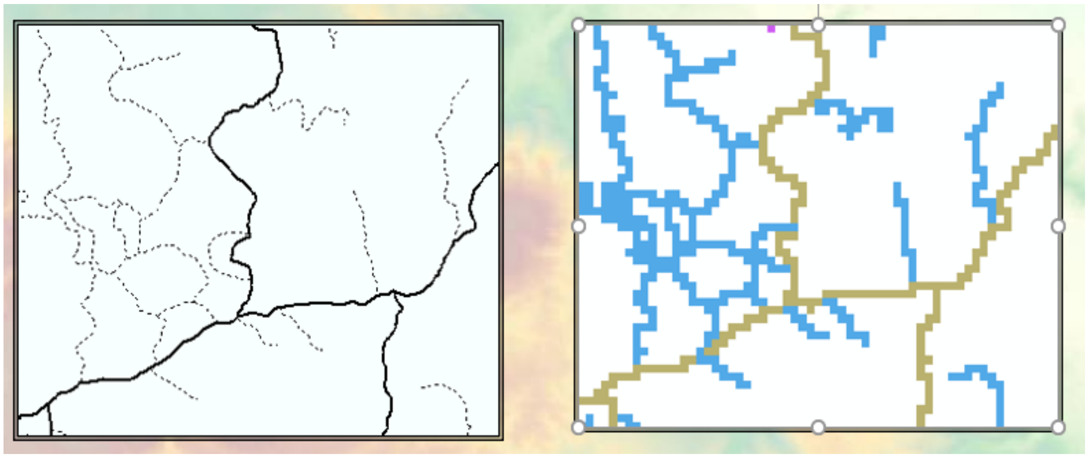


Figure 5.1 Vector vs. Raster model

Left image is a vector model of the data, the right image shows a raster model of the same data. From a PowerPoint lecture given by Maribeth A. Price at the University of Wisconsin – Green Bay. (https://www.uwgb.edu/dutchs/EnvSci454-PPT/VectorvsRaster.ppt)

5.1 GiST

Many GIS software utilize PostgreSQL, an open source object-relational database [14]. PostgreSQL uses several different types of trees for storing information, mainly a generalized search tree, or GiST [11].

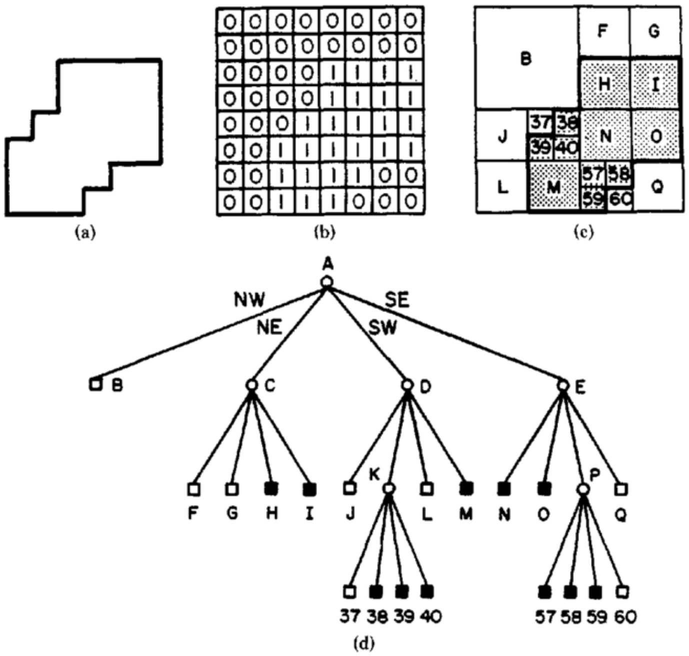
The generalized search tree is a variation of a B-tree, which is a self-balancing search tree used often in database structures. B-trees tend to be wide but not tall, making them easy and fast to search.

6. Quadtrees

A spatial index is a type of data structure used to represent graphical or geometric data. These data are usually first stored in a spatial database (also known as a geodatabase), and indexed to optimize the time it takes to query the database. The GiST and B-tree are not considered to be spatial indexes, as their use is not limited to graphical or geometric data. However, a GiST can be used as an underlying data structure for a spatial index, such as an R-tree, by using spatial coordinates as the search keys in the GiST structure.

6.1 Creating & Searching the Quadtree

One type of spatial index is a quadtree. In a quadtree, the desired area is divided recursively into fourths—that is, each parent node has four child nodes [13]. Division stops when the area has been divided into “small enough” units, for instance a single pixel. A merge operation can then be used to merge parent nodes whose children all belong to the same region—for instance, in the figure below, one could potentially map the image in (a) to the tree in (d) by first making all the leaves at the same level, and then merging up. This prevents “extraneous” nodes, such as all the “white” nodes in section B of (c).



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	0	0
0	0	1	1	1	0	0	0

Figure 6.1 – An Example of a Quadtree Created from an Image.

(a) the image; (b) a representation of the image on a grid, with 1s (black) indicating pixels which belong to the image, and 0s (white) indicating pixels which do not belong to the image. (c) the image as it would be mapped to the quadtree, represented in the same grid as (b). (d) the quadtree, with A as the root node. [13]

A node in the quadtree can be found much in the same way that we create the quadtree. The desired node can be recursively compared to the NW, NE, SW, and SE boundaries until a leaf is found. If the leaf matches or contains the desired node, it has been found; otherwise, it does not exist in this tree.

6.2 Quadtrees and GIS

There have been several forays into using quadtrees alongside geographic data and GIS software. One study conducted in China looked at encoding raster data by using what they termed a Fast Dynamic Linear Quadtree Encoding method, or FDLQE [17]. This method scans the raster data only once, and constructs the tree by utilizing a dynamic stack. The purpose of analyzing this method, however, was only to optimize the algorithm of the FDLQE, and it does not appear that this method was implemented alongside GIS software or tested with actual raster data. Instead, the researchers used decimal Morton code to express the location of the leaves in the quadtree.

Another study, conducted at the University of Maryland, implemented a brand new GIS software which utilized a quadtree and a B-tree instead of the GiST and other trees provided by PostgreSQL [12]. The researchers used data provided by the U.S. Army Engineer Topographic Library, and the quadtree-based GIS software was developed in part by Hanan Samet, who also published one of the generally accepted authoritative papers on quadtrees and other related spatial indexes. The implementation done by Samet and the other researchers included a quadtree editor, which allows a user to manipulate the tree themselves and see the effects of each step. The researchers found that the quadtree yielded a more efficient handling of many different queries, however their implementation was still limited in the number of different kinds of queries it could handle and the space available to store it.

3. Problem Statement

The additional challenges presented by a soundscape map may reveal the quadtree as an equally viable spatial index. Although a developer who wants to use a quadtree will have to build their own, the quadtree is relatively easy to build and its efficiency could make up for the inconvenience of building a separate data structure.

To evaluate if the quadtree is a viable spatial index for storing the information related to a soundscape map, several qualities must be considered:

1. How big is the produced tree? It must be reasonably sized, given the amount of data it represents. If the tree is unreasonably large, and it contains too many nodes to be searched efficiently, it will not be a viable data structure for this type of data.

2. How fast is the lookup time for a node in the tree? In correspondence with the size of the tree, the time it takes to find and return desired information must be nearly instantaneous. Otherwise, the sound does not correspond with the correct location of the user’s cursor or finger, and thus is not the correct data.

3. How easy is it to build a quadtree? Quadtrees are not currently used in GIS toolkits, so a geographer or developer who wishes to use one will have to use a software that is perhaps less supported or build one of their own. If the quadtree is too complicated to implement, it is not going to be worth it to produce separately from software a developer or geographer is already using.

If the quadtree satisfies these qualities—it is reasonably sized for the amount of data it represents, the lookup time is reasonably quick, and it is relatively easy to build—then it proves to be a viable spatial index for representing a soundscape map.

4. Tests & Results

1. Introduction

Two programs were developed for the purpose of testing whether the quadtree is a viable data structure for soundscape data. One program was used for converting and scaling the coordinates to fit the image size, and one built a quadtree using a given dataset and ran subsequent tests on the resulting tree.

The program was developed in a 64-bit Linux environment (Ubuntu 14.04 LTS), partly on a virtual machine (VirtualBox) and partly on a native Ubuntu machine. It was tested on only the native Ubuntu machine, an ASUS desktop computer with 15.5 GiB memory and an Intel Core i5 processor with 4 CPUs.

Sample datasets were provided by the advisor, Prof. Michal Young, and contained all the data necessary for mapping buildings in the nearby town of Springfield, OR. Since this original dataset was far too large for testing purposes, small sections of neighboring buildings were taken for use in the tests.

The sample datasets taken for use in tests fell into three size categories: 6 regions, 35 regions, and 196 regions. For the two smaller datasets, 6 and 35 regions, four different datasets were used, and the results are averaged over all the files.

2. Tree Size

One way to measure efficiency of a tree is to compute its size, which is the number of all internal and external nodes, including the root node. If a tree is not as deep, or not as wide, as another tree, the time it takes to find and retrieve the information stored at a node will be faster.

2.1 Test

In a quadtree, the image is divided recursively into fourths. This happens most easily when the dimensions of the image are powers of two and square--that is, 1024x1024, 2048x2048, etc. For images whose dimensions are not square or not powers of two, the program will round the largest dimension to the nearest power of two. Since the calculations of size are always made from images with square dimensions (given or calculated) of powers of two, test cases only included such dimensions.

2.2 Results

The expectation for these results were that the size of a given tree would grow logarithmically in relation to the size of the image. The size test showed that the size of the tree grows somewhere between linearly and logarithmically with the size of the image, which is not ideal but still within a reasonable bound for determining the quadtree's viability as a data structure. The graphs show the size of the quadtree (y-axis) plotted against the size of the image (x-axis). For this test, only square image sizes were tested, and only images sized 8x8 pixels or larger were tested. The maximum image size for this test was 4096x4096 pixels.

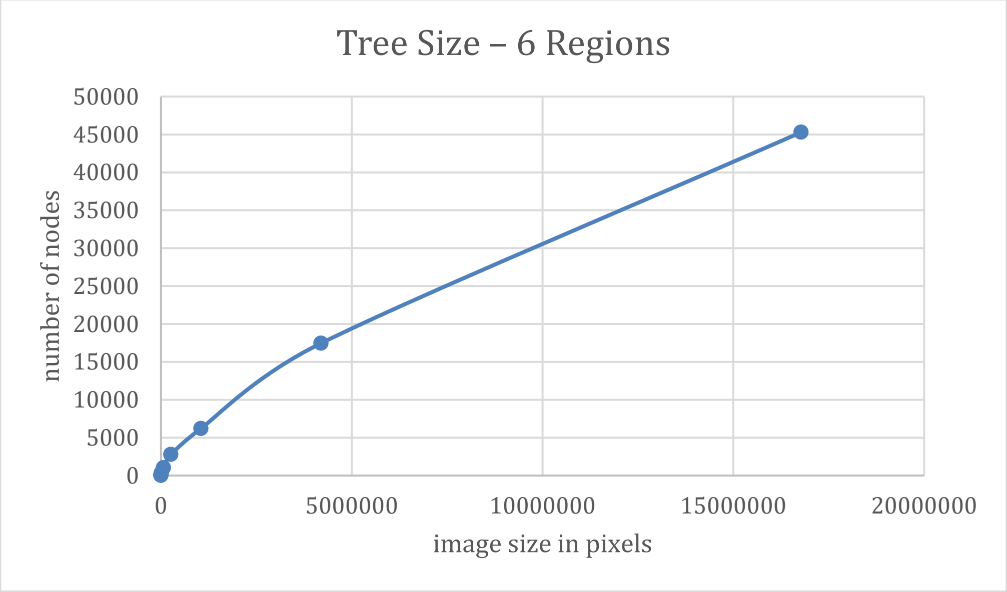


Figure 2.1 Tree Size for 6 Regions

Table 2.1 Size of Quadtree – 12 regions

Image Size (in pixels)	Average Size of Quadtree (number of nodes created)
8x8	21
16x16	39.25
32x32	75.75
64x646	191
128x128	411.25
256x256	1043.5
512x512	2790.25
1024x1024	6213.75
2048x2048	17458
4096x4096	44250.25

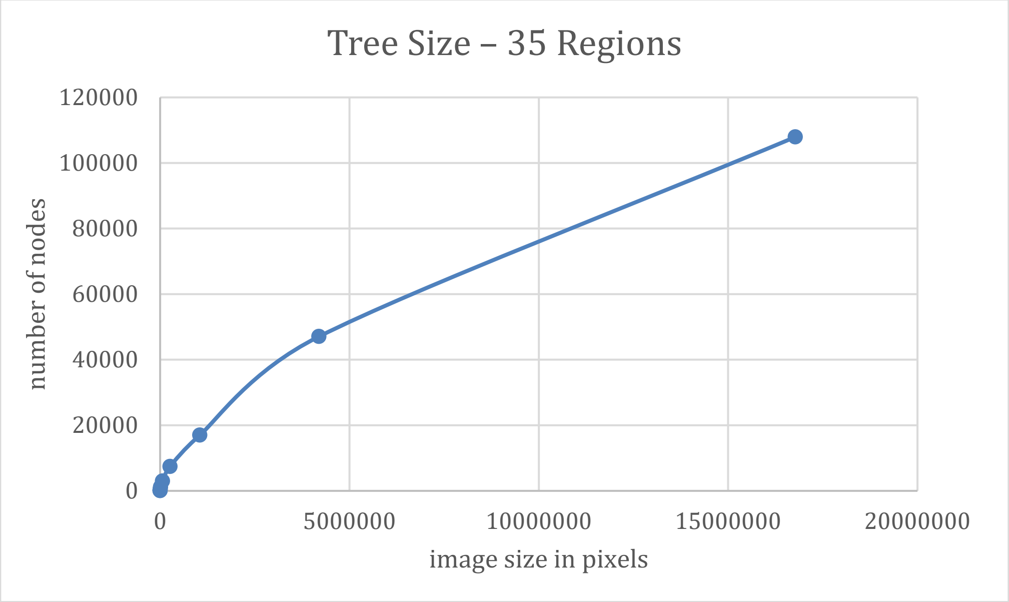


Figure 2.2 Tree Size for 35 Regions

Table 2.2 Size of Quadtree – 35 regions

Image Size (in pixels)	Average Size of Quadtree (number of nodes created)
8x8	37.5
16x16	96.25
32x32	216.5
64x646	517.25
128x128	1162.25
256x256	2974.25
512x512	7444.5
1024x1024	16984.25
2048x2048	47065.5
4096x4096	107927.8

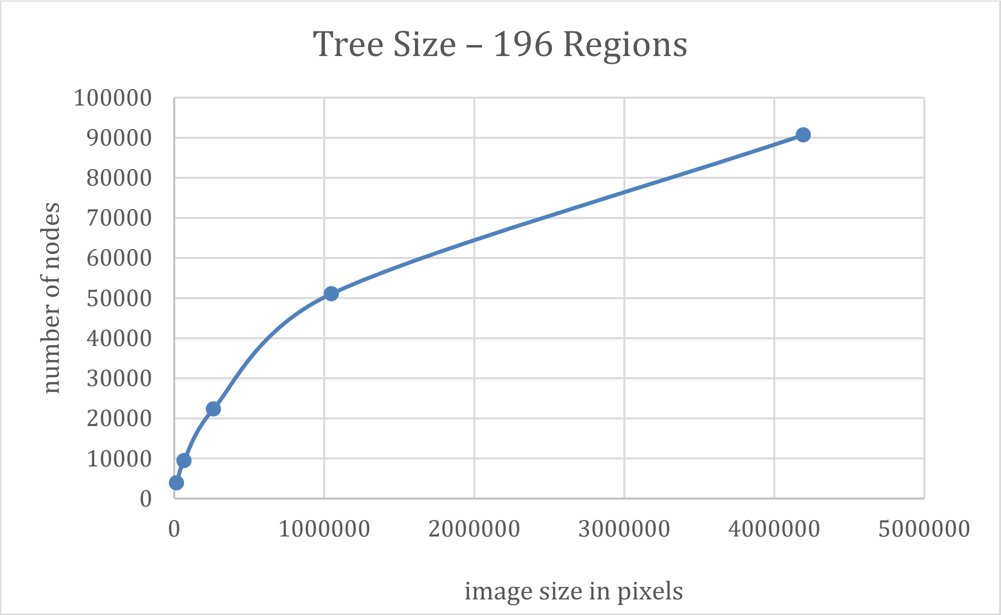


Figure 2.3 Tree Size for 196 Regions

Due to the number of regions in this dataset, the tree size was only computed for images size 128x128 (2^7x2^7) and larger.

Table 2.3 Size of Quadtree – 196 regions

Image Size (in pixels)	Size of Quadtree (number of nodes created)
128x128	3899
256x256	9434
512x512	22333
1024x1024	51074
2048x2048	90674
4096x4096	N/A

Table 2.4 Tree Size – Complete Results 6 Regions

Image dimensions	**6_1**	**6_2**	**6_3**	**6_4**
8x8	18	20	22	24
16x16	33	45	38	41
32x32	50	87	79	87
64x64	159	217	198	190
128x128	339	440	441	425
256x256	958	1107	1036	1073
512x512	2617	3014	2700	2830
1024x1024	6439	6806	6049	5561
2048x2048	14462	21445	18355	15570
4096x4096	34203	63618	44250	39130

Table 2.5 Tree Size – Complete Results 70 Regions & 200 Regions

Image dimensions	35	35_1	35_2	35_3	196regions
8x8	59	16	35	40	N/A
16x16	178	40	77	90	N/A
32x32	444	97	144	181	N/A
64x64	1090	243	328	408	N/A
128x128	2483	513	720	933	3899
256x256	6696	1102	1726	2373	9434
512x512	17370	2902	4228	5278	22333
1024x1024	39177	5805	10501	12454	51074
2048x2048	93701	21241	32685	40635	90674
4096x4096	160310	57770	89677	123954	N/A

3. Time Tests

Another way to measure efficiency is to look at how long it takes to find a desired node in the tree, to discover that a desired node is not in a tree, or that the desired node is not part of a region that contains relevant data. This test is crucial because it measures the response time for a query. In an application of a soundscape map, the program must respond to the user nearly instantaneously, so that the user can build the correct mental map of the data. The user’s finger or cursor must never be behind or too far in front of the point queried, as the data returned would then be incorrect in correlation with the position of the finger or cursor.

3.1 Test

Data for this test consisted of measuring, using CPU time, how long it took to query the tree for a given point, and return the object or node associated with that point. While this test used only one file of 217 sets of mouse coordinates, the file was used alongside the size test for each set of regions, and averages taken over the files containing the same number of regions (e.g. there were four datasets containing approximately twelve regions, so the results of the time tests for twelve regions is averaged over the four datasets). This test used the same power of 2 square image sizes as the size tests.

3.*2 Results*

In order for the quadtree to be a viable data structure, the retrieval times for a node must be nearly instantaneous. An article from PubNub, which produces an API for creating real-time apps, quantifies an “instantaneous” response time as 1 ms, or 0.001 seconds [4]. For these tests, the expectation was that the response times would fall between this instantaneous response time (1 ms) and an arbitrary upper bound of 10 ms.

The retrieval times from these tests, even in worst-case scenarios, fall well within these bounds. While this test was run with every image size starting with 8x8, only results for images size 512x512 and larger are reported here. Times are all given in milliseconds.

Table 3.1 – Average Response Time (ms) for 6 Regions

Image Size	**Average response time**	**Standard deviation**	**Minimum response time**	**Maximum response time**	**99th Percentile**
512x512	0.263	0.163	0.014	0.626	0.598
1024x1024	0.207	0.099	0.020	0.387	0.376
2048x2048	0.157	0.076	0.017	0.294	0.289
4096x4096	0.159	0.079	0.020	0.329	0.322

Table 3.2 – Average Response Time (ms) for 35 Regions

Image Size	**Average response time**	**Standard deviation**	**Minimum response time**	**Maximum response time**	**99th Percentile**
512x512	0.181	0.114	0.010	0.446	0.431
1024x1024	0.172	0.086	0.017	0.344	0.338
2048x2048	0.151	0.072	0.020	0.277	0.272
4096x4096	0.170	0.081	0.023	0.316	0.311

Table 3.3 – Average Response Time (ms) for 196 Regions

Image Size	**Average response time**	**Standard deviation**	**Minimum response time**	**Maximum response time**	**99th Percentile**
512x512	0.153	0.078	0.016	0.270	0.269
1024x1024	0.162	0.078	0.021	0.280	0.280
2048x2048	0.168	0.079	0.023	0.285	0.283
4096x4096	N/A	N/A	N/A	N/A	N/A

5. Analysis

1. Size Tests

For the size tests, all graphs of tree size plotted against image size showed somewhere between linear and logarithmic growth, with the smallest region size (6 regions) showing a more linear growth pattern and the largest region size (196 regions) showing a more logarithmic growth pattern. Based on this data, it seems that a quadtree will provide an efficient data structure for storing soundscape data.

2. Time Tests

In the time tests, results were generally not accurate until the 128x128 image size, and the program did not process all the points in the file until the 512x512 image size. Given the additional considerations for a soundscape map, it is unlikely that a map would be smaller than 512x512 pixels, and thus it is reasonable to assume that results for images smaller than 512x512 are irrelevant to the discussion of a quadtree as a viable data structure for this type of data.

Over all datasets, the maximum (worst-case) response time was 0.626 milliseconds, which still falls well within the bounds of 1-10 ms. The minimum response time, over all datasets, was 0.010 milliseconds. Therefore, in the area of response time, it appears that a quadtree is a viable data structure for storing soundscape map data.

3. Limitations & Future Plans

In the time tests, only one file of coordinates was used in testing. The results from this test show that the smaller image size (512x512 pixels) generally produced the longest response times, and the largest image size (4096x4096) showing a generally shorter response time. This could be due to the specific coordinates in the file—if more of the coordinates were not found in the tree, the find operation would search the entire tree, which would take more time than if the node was found somewhere in the tree. In a larger image size, it is likely that more of the coordinates were found in the tree, which would produce faster response times. In any case, the difference in worst-case times is 0.356 milliseconds, which is still quite small. Ideally, multiple different files should be tested, and aggregated over all sizes, which might produce a trend more like what might be expected—that as the size of the tree grows, it takes longer to find a node in the tree.

Additionally, since the numbers for response time are so small, they might have been affected by noise as a result of I/O (including reading from the file of coordinates and writing the response times to the output file). However, since this noise will generally worsen numbers, rather than improve them, this noise does not affect whether the quadtree is or is not a viable data structure for this type of data.

It may have been useful to use smaller coordinates with the smaller image sizes (to 512x512), however as stated above, since it is unlikely that a soundscape map would be smaller than 512x512 pixels, it is not necessary to gather results from smaller image sizes.

Lastly, it was not possible to gather data for the largest image size and largest region set (4096x4096 and 196 regions) most likely due to memory errors—the program stalled and caused the computer to freeze during multiple attempts. Knowing the response times and tree size for this largest dataset is useful, as it is closest to the size and amount of data that would be contained in a real soundscape map.

In the future, a more thorough investigation of the quadtree would be useful, particularly in comparing it with current, generally used GIS software. Other improvements might include using a wider set of regions—the jump from 35 regions to 196 regions is quite large, and with more time it might be useful to test sets of 50, 100, or 150 regions as well. Especially with regard to tree size, putting more data points between 35 and 196 regions would make clearer whether the growth of the tree was more linear or more logarithmic.

6. Conclusion

When viewing a choropleth map, a sighted user can extrapolate many trends from simply looking at the graphic. For a blind or visually impaired user, these trends must be conveyed using alternate senses. In creating an adaptive application for navigating a choropleth map, one of the most common implementations is using different sounds to represent the shading on a visual map, in conjunction with a touchscreen. The user can sweep his or her finger across the screen and the application continually responds with the sound or data associated with the current position of the user's finger. In order to support this real-time interaction, it is crucial to utilize a spatial index for storing this type of data that responds nearly instantaneously. If the data is not returned at the location of the user's finger, the user cannot get an accurate mental picture of the map and thus the application is not useful.

Although quadtrees are not generally built into GIS software, research has determined that it may be a good data structure for storing graphical data. To add to the existing research, this thesis tested the quadtree as a viable data structure by building a sample tree and examining two major factors: the size of the quadtree, and the retrieval time of a desired node in the quadtree. Results of these tests showed that, as predicted, the size of the quadtree grows logarithmically with the amount of data stored, and the retrieval time for nodes in the tree falls within the range that a human would perceive as instantaneous. As such, it appears that the quadtree is a viable data structure for storing data associated with soundscape maps. This research is important in furthering development of choropleth maps for blind or visually impaired users, because all people deserve the ability to access information in a way that they can understand.

Appendix

Testing Environment

All programs used for testing were written in C and are designed to run with the gcc compiler. Three main files were developed and used in this project: quadtree.c, scalefloat.c, and mouse.c.

Development & Conventions

quadtree.c

The quadtree program consists of a driver, quadtree_driver.c; a header file, quadtree.h; and a file of functions, quadtree.c. When run via the command line, it takes as arguments a file of type well-known text (*.wkt), a text file of mouse coordinates, and the x and y dimensions of the desired image.

When testing, the output was redirected to a text file so the results could be viewed more easily, and kept for analysis later. During development and testing, the program was run with the following commands:

```
gcc -std=c99 -o quadtree quadtree.c quadtree_driver.c
./quadtree [wktfile] [textfile] [x dimension] [y dimension]
>outputfile.txt
```

scale.c

The scale program takes in a *.wkt file of coordinates and outputs a *.wkt file of converted coordinates. In testing, the *.wkt file was first converted to a text file, put through an online conversion program (found at http://tool-online.com/en/coordinate-converter.php#) with the left-hand settings as WORLD > WGS84 (GPS) and the right-hand settings as WORLD > WGS84_XYZ (GEOCENTRIC). The commands for running the scale program are as follows:

```
gcc -std=c99 -o scale scalefloat.c
./scale [wktfile]
```

mouse.c

The mouse program was developed for collecting mouse coordinates to test with the quadtree program. It does not take any other files as arguments, and when run, produces a blank window. When the user drags their mouse across the window, the coordinates are recorded in an output file.

To run the mouse program, use the following commands:

```
gcc -std=c99 -o mouse mouse.c -lglut -lGLU -lGL
./mouse
```

When finished collecting coordinates, end the program by exiting out of the window.

Input Files

The quadtree program takes in a *.wkt file of integers only. It currently only recognizes the keyword “POLYGON” which precedes the coordinates in the *.wkt file, however it could be modified to recognize other keywords (e.g. “POINT” or “LINE”). The quadtree program is also only designed for use with integers, so the *.wkt file must contain points of only integer values.

The quadtree program also takes in a file of mouse coordinates. These were collected using an openGL program called mouse.c. This program tracks the location of the mouse and outputs the series of coordinates to a *.txt file.

Method of Testing

The scale program creates a scaling coefficient by using the minimum and maximum values in the *.wkt file and the desired size of the image. Because of this, the scale program must be run twice (once to determine the scaling coefficient, and another time to correctly scale the coordinates) at the beginning of a new file, and recompiled and run for each image size. In the future, this program could be automated to produce scaled coordinates for each image size without having to recompile.

The scale program outputs the minimum and maximum x and y for error-checking purposes, and a new minimum and maximum x and y. The new maximum x and y must then be put into the scalex and scaley functions within the program as the values for variables dimx and dimy, respectively. This ensures that the scale coefficient for each image size is correct, given a specific *.wkt file. For this project, the scaled coordinates were always output to a file called buildings.wkt, which allowed the quadtree program to be run multiple times with the same (or nearly the same) command.

The converted coordinates file may then be passed to the quadtree program. For this program, the command to run was always the same, except for the dimensions of the image (which increased from 8x8 to 4096x4096) and the name of the output file. Each output file contained the amount of time it took to find each point in the mouse coordinates file, whether that point was found or not found (and if found, where), the size of the image, and the size of the tree created.

Bibliography

[1] choropleth map | Definition - Esri Support GIS Dictionary: *http://support.esri.com/other-resources/gis-dictionary/term/choropleth map*. Accessed: 2017-04-26.

[2] GIS Spatial Data Types: Vector vs Raster - GIS Geography: *http://gisgeography.com/spatial-data-types-vector-raster/*. Accessed: 2017-05-06.

[3] Google Maps: *maps.google.com*. Accessed: 2017-01-01.

[4] How Fast Is Realtime? Human Perception and Technology | PubNub: 2015. *https://www.pubnub.com/blog/2015-02-09-how-fast-is-realtime-human-perception-and-technology/*. Accessed: 2017-05-07.

[5] Isoline Maps - Piggraphy Alevel Skills: *https://sites.google.com/site/piggraphy/home/cartographic-d/isoline-maps*. Accessed: 2017-04-26.

[6] iSonic: Interactive Data Sonification for Blind Users - YouTube: 2013. *https://www.youtube.com/watch?v=8hUIAnXtlc4&index=3&list=PLEiWrxYPZmLbVAtHoTUXrpORY1GRQgPIp&t=221s*. Accessed: 2017-04-26.

[7] Lazar, J. et al. 2013. Development and Evaluation of Two Prototypes for Providing Weather Map Data to Blind Users Through Sonification. *Journal of Usability Studies*. 8, 4 (2013), 93–110.

[8] Lobben, Amy; Lawrence, M.M. 2011. The Design of Tactile Thematic Symbols. *Journal of Visual Impairment & Blindness*. 105, 10 (2011), 681–691.

[9] Lobben, A. and Lawrence, M. 2012. Use of Environmental Features on Tactile Maps by Navigators Who Are Blind. *The Professional Geographer*. 64, (2012), 95–108.

[10] Median Household Income: 2015: 2016. *https://www.census.gov/did/www/saipe/data/statecounty/maps/iy2015/med_hh_inc2015.pdf*. Accessed: 2017-04-26.

[11] PostgreSQL: About: *https://www.postgresql.org/about/*. Accessed: 2017-05-06.

[12] Samet, H. et al. 1984. A Geographic Information System Using Quadtrees. *Pattern Recogjnition*. 17, 6 (1984), 647–656.

[13] Samet, H. 1984. The Quadtree and Related Hierarchical Data Structures. *ACM Computing Surveys*. 16, 2 (1984), 187–260.

[14] Section 8: Spatial Indexing — Introduction to PostGIS: *http://revenant.ca/www/postgis/workshop/indexing.html*. Accessed: 2017-05-06.

[15] Vector Draw Tactile Maps: *http://www.tsbvi.edu/images/downloads/SWOMA2015Handouts/VectorLineDrawTactileMaphandout.pdf*. Accessed: 2017-04-12.

[16] What is GIS? | The Power of Mapping - Esri: *http://www.esri.com/what-is-gis*. Accessed: 2017-04-26.

[17] Yehua, S. et al. Linear Quadtree Encoding of Raster Data and its Spatial Analysis Approaches. 1–7.

[18] Zhao, H. et al. 2008. Data Sonification for Users with Visual Impairment : A Case Study with Georeferenced Data. *ACM Transactions on Computer-Human Interaction*. 15, 1 (2008), 4.