

INCREASING USABILITY AND ACCESSIBILITY OF GIS TECHNOLOGY THROUGH
EXTENSION OF EXISTING SOFTWARE TOOLS

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Elizabeth Joan Noel

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Program:
Software Engineering

June 2015

Fargo, North Dakota

North Dakota State University
Graduate School

Title

Increasing Usability and Accessibility of GIS Technology through
Extension of Existing Software Tools

By

Elizabeth Joan Noel

The Supervisory Committee certifies that this *disquisition* complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Anne Denton

Chair

Kenneth Magel

Stephanie Day

Approved:

06/08/15

Date

Brian Slator

Department Chair

ABSTRACT

As the popularity of Geographic Information System (GIS) technology increases and its use extends from specialized users to the general public, tools available to users must adapt to take these changes into account. Virtual globes, such as Google Earth, are simplified GIS tools which do not include any of the data analysis and visualization capabilities of traditional GIS. However, the use of these applications can be extended by shifting the GIS functionality out of the application and into KML, through the dynamic manipulation of KML objects, allowing for some of the functionality of GIS to be reintroduced. Because KML objects include coordinates, they can be manipulated in a way which allows georeferenced data to be visualized within virtual globes in a way that is user-accessible, allowing for GIS to be more accessible to a greater number of users.

TABLE OF CONTENTS

ABSTRACT.....	iii
LIST OF TABLES.....	viii
LIST OF FIGURES.....	ix
1. INTRODUCTION.....	1
1.1. Rationale.....	1
1.2. Example application.....	2
1.3. The potential of KML for encapsulating data presentation.....	3
1.4. Contribution.....	3
2. BACKGROUND.....	5
2.1. Implementing GIS technology in education.....	5
2.2. Issues with GIS technology implementation.....	7
2.3. GIS user interface evaluation.....	8
3. CONCEPTS.....	13
3.1. Virtual globes.....	13
3.2. Comparison of virtual globe and traditional GIS functionality.....	16
3.2.1. Data management.....	16
3.2.2. Data analysis and visualization.....	17
3.3.3. Macros in GIS.....	18
3.3. KML.....	18
3.3.1. KML format.....	19
3.3.2. KML vs. Shapefile.....	20

4. GOALS	21
4.1. Virtual globes	21
4.2. Sample data sets	22
4.3. Querying capability	22
4.4. Google Earth	23
4.5. KML	23
4.6. Data visualization	24
4.7. Summary of goals	24
5. SOFTWARE DESIGN	26
5.1. Design overview	26
5.1.1. File reader	26
5.1.2. Database connection	27
5.1.3. User options	28
5.1.4. KML generation	29
5.2. User interface flow	29
5.2.1. Initiating the application	29
5.2.2. Select user file	29
5.2.3. Editing and reviewing data	31
5.2.4. Setting chart options	32
5.3. Class overview	32
5.3.1. Reading user files	34
5.3.2. Chart generation	35
5.4. KML logic	37

5.3. Projecting circles on Earth's surface.....	39
6. ANALYSIS.....	42
6.1. Sample data generation comparison	42
6.1.1. Generating a bar chart in GRASS GIS using sample data	42
6.1.2. Generating a bar chart in the chart generator application	44
6.2. Creating a bar chart from external data in GRASS GIS	49
6.2.1. Accessing Shapefiles	49
6.2.2. Importing Shapefiles into GRASS GIS.....	50
6.2.3. Viewing the Shapefile in GRASS	52
6.2.4. Cleaning the data.....	54
6.2.5. Displaying the bar chart	56
6.3. Creating a bar chart using external data in the KML generating application	59
6.3.1. Data preprocessing	60
6.3.2. Uploading and cleaning data	62
6.3.4. Opening KML in Google Earth.....	64
6.4. Creating a pie chart using external data in GRASS GIS	66
6.4.1. Data preprocessing	66
6.4.2. Displaying the pie chart.....	67
6.5. Creating a pie chart using external data in KML generating application	68
6.5.1. Data preprocessing	68
6.5.2. Selecting options for pie chart.....	68
6.6. General comparison of GRASS GIS and KML generator.....	71
6.6.1. Functionality.....	71

6.6.2. Required knowledge.....	71
6.6.3. Cleaning data.....	72
6.6.4. User interface interaction	72
6.7. Performance	73
6.7.1. Reading Excel files.....	73
6.7.2. Creating KML files	75
6.8. Sample output	76
6.8.1. Stacked bar chart	77
6.8.2. 100% bar chart.....	78
6.8.3. Pie chart.....	79
6.9. Other software design considerations	80
7. CONCLUSION.....	82
REFERENCES	84

LIST OF TABLES

<u>Table</u>	<u>Page</u>
1: Summary of Teacher Responses to GIS-Based Surveys	7
2: Comparison of Basic Features of Three Major Virtual Globes	15
3: Graph Types Available in Two Major GIS Software Packages	17
4: Functional Requirements for Chart Generating Application	25
5: Valid User Input Formats	26

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
1: GRASS GIS Layer Manager Window	9
2: GRASS GIS Default Map Display	10
3: GRASS GIS Raster Map Display Window	11
4: GRASS GIS Map Display with Roads Displayed	11
5: Google Earth with Placemark Added at NDSU Campus Location	14
6: Example of KML Language	19
7: Database Tables for Querying Latitude and Longitude.....	27
8: Example of a Form Which Allows User to Set Chart Display Options	28
9: Description of User Flow through Forms within the Application	30
10: Example of the Review Step within the Application.....	31
11: Classes within the Chart Generating Application.....	33
12: Programming Logic for Creating Points in a Circle around a Location.....	38
13: Example of How Latitude and Longitude Distance Can Vary with Location.....	39
14: Finding Latitude and Longitude Using Distance and Bearing	40
15: Method for Finding Latitude and Longitude around a Circle.....	41
16: Starting Point for GRASS GIS version 6.4.4.....	43
17: Commands for Setting and Creating a Bar Chart in GRASS GIS	43
18: Bar Chart Output in GRASS GIS Using the Example Provided	44
19: Choosing from Sample Data Options	45
20: Selecting a Chart Type.....	46
21: Selecting a Title and Category.....	46

22: Selecting User Options	47
23: Output from Malaria Sample Data Set Viewed in Google Earth	47
24: Google Earth Navigation Pane Showing User-Selected Display Values	48
25: Initial GRASS GIS Form.....	50
26: Selecting the Correct Import Option in GRASS.....	51
27: Result of Initial Shapefile Import	52
28: Example of a High Snap Parameter Value in GRASS	53
29: Vector Cleaning Window in GRASS	54
30: Remaining Extraneous Centroids after Running Rmarea Tool in v.clean Command.....	55
31: Initial Chart Creation in GRASS GIS.....	57
32: SELECT Query Output.....	58
33: Bar Chart Output for Energy Use in GRASS GIS.....	59
34: Form Showing Valid Formats and Term Definitions	60
35: Adjusted Data with Two Worksheets for Each Input Format	61
36: Selecting a Worksheet from the Chosen File.....	61
37: Selecting the Format of the User File	62
38: Example of a Successful File Upload.....	62
39: Uploaded Data Review	63
40: Google Earth Output of Energy Usage.....	65
41: Output for Pie Chart in GRASS GIS	67
42: Selecting Pie Chart Options.....	68
43: Pie Chart Display Options	69
44: Example of a Pie Chart with the Default Radius	70

45: Example of a Pie Chart with a Smaller Radius.....	70
46: Application Performance for File Reading When No Querying is Necessary	73
47: Application Performance When Querying by Country Name.....	74
48: Application Performance for KML Creation.....	75
49: Comparison of File Reading with and without Querying, As Well as KML Creation.....	76
50: Stacked Bar Chart Example Using CO2 Emissions Data Set.....	77
51: 100% Bar Chart Example Using Income Inequality Data Set.....	78
52: Pie Chart Example Using Income Inequality Data Set.....	79

1. INTRODUCTION

As the world becomes more connected through technology, it becomes increasingly important to understand global trends that affect our lives and environments. Global health and environmental issues are at the forefront of global sociopolitical discourse. In order to provide the tools necessary for the general public to understand the global issues facing their lives and their futures, technology must adapt, becoming more easily accessible and usable to a greater number of users.

1.1. Rationale

GIS (Geographic Information System) technology allows users to map, view and analyze geographic information through technology. In his 1998 speech, Al Gore discussed the virtues and importance of technology which allows users to view and manipulate the increasing amount of georeferenced data which is becoming available, and pointed out that “the hard part of taking advantage of this flood of geospatial information will be making sense of it” [1]. While the application of mapping technology is primarily one of geographical inquiry, the information that can be derived using GIS is multidisciplinary. For example, GIS can be applied to health, environmental, and social issues worldwide by mapping data associated with each of these fields in a geospatial context.

However, the practice of implementing GIS among non-specialized users has proven difficult for various reasons. These users currently can only make use of GIS through use of the same technology used by professionals, which assumes knowledge about GIS data formats and concepts that the general populace is unlikely to have. This lack of usability of software when taking untrained users into account means that the use of traditional GIS software is limited. The user interface is not only generally difficult to navigate, but it is not geared toward beginners, or

those without knowledge of the data formats associated with GIS, making it difficult for users to know which steps to take in order to view georeferenced data and make use of the tools provided by the application.

By focusing on improving the usability and accessibility of GIS technology through the extension of existing tools, each of these concerns could be dealt with concurrently. To make GIS more accessible to a greater number of users, software must be available that is free to download. However, addressing this first issue must also take into account the perceived complexity of the software. That is to say, a feasible solution to a simplified GIS cannot just be a free version of typical GIS software, as the perceived complexity of the software would not be dealt with in this case.

1.2. Example application

Virtual globes, such as Google Earth are both free and easy to use, allowing for greater accessibility than with most traditional GIS software. However, when working with virtual globes, you lose spatial analysis and data visualization capabilities which are typical in traditional GIS software. This loss inhibits the use of virtual globes, as they exist in their basic form, as tools for spatial thinking and geographic awareness.

Despite their limitations, virtual globes are ideal candidates for a simplified geographic tool which can be engaging for users with little to no GIS experience. In order to make this possible, functionality must be added back in to virtual globes. This can be through the use of a standalone application which separates out the functionality which exists in traditional GIS, rather than making changes to virtual globe software. This keeps the application modular in the sense that it does not have to be changed as virtual globe software updates, as long as the data format remains the same.

1.3. The potential of KML for encapsulating data presentation

Through the use of KML, it is possible to add layers to virtual globes which allow users to visualize data. KML files can be easily imported into virtual globes; in fact, the process for opening a KML file in Google Earth is the same as it is for opening any type of document in a typical application (i.e. by selecting “File”, then “Open”). However, the files require additional knowledge to create, limiting their use. A solution to the current limitations of virtual globes is through extension of existing tools by separating out some of the functionality of traditional GIS technology in a way that is user-friendly, which results in the extension of the use of KML objects beyond their original intended functionality. As long as user data has some geographic element, it can be viewed in a geospatial context by converting the data into 3-dimensional charts through the use of KML Polygon objects.

By isolating and simplifying content which is common in traditional GIS technology through the extension of existing technology, users can be introduced to GIS concepts without the necessity of relying on technology which is used by professionals. A sample application is given which implements this idea, allowing users to create KML files which can be opened in commonly used virtual globe applications, such as Google Earth. These KML files project Polygon objects onto the surface of the Earth, allowing the user to view their data in a geospatial context.

1.4. Contribution

Modern mapping applications, such as Google Earth and Google Maps have made GIS more accessible to the general public than ever before. Despite the additional accessibility, Google has stripped out much of the GIS capability of data analysis and visualization. This thesis reintroduces GIS functionality in a way which is fundamentally different than the traditional GIS

approach. By shifting the functionality out of the virtual globe implementation and into KML, which is user accessible and human readable, a more modular design is created which requires less specialized knowledge on the part of the user. By dynamically converting georeferenced data into KML, some of the data visualization functionality of KML is reintroduced to virtual globes without the need for any changes to the current virtual globe format.

2. BACKGROUND

2.1. Implementing GIS technology in education

A common case in which beginners have dealt with GIS is in primary and secondary educational environments. As technology expands into different corners of the world and our lives, it affects how we view and interact with the world, thus becoming increasingly important in education. The National Academy of Sciences asserted in 2006 that this “global reach of the Internet will make the understanding of space even more important” [2]. This assembly of scientists agreed that the use of GIS in education is a way of supporting spatial thinking in K-12 education. This assertion has been supported in research. A 2009 study showed that students’ spatial thinking skills improved over the course of a semester of GIS education [3]. In addition to spatial thinking, GIS has been shown to modestly improve data analysis skills [4]. However, the implementation of GIS education in schools has been a focus since the late 20th century. In 1998, Al Gore made a speech at the California Science Center which recommended the use of GIS in education in order to “spark the development of a Digital Earth” [1]. More recently, The Beijing Declaration on Digital Earth, in 2009, called for “investments in scientific research, technology development, education and popular promotion of the benefits of Digital Earth” [5].

Not only are suggestions that GIS should be implemented a global trend, attempts at implementing GIS technology into educational programs are also widespread. Countries from The US [6], to Turkey [7] [8], to Singapore [9] have implemented or attempted to implement GIS programs in education. Each of these locations have encountered similar difficulties, which will be looked at in turn, and which are summarized in table 1.

In 2003, Kerski [6] reviewed the current state of GIS in US schools. He found that fewer than 5% of US secondary schools owned GIS software, and that nearly half of those schools

weren't using it. However, among the schools surveyed, 88% agreed that "the use of GIS makes a significant contribution to learning" [6]. The instructors cited three major reasons for the dichotomy between usefulness of GIS and its actual use: lack of time, little support/training, and the complexity of the software.

A similar survey was conducted in Turkey in 2009 [7], in which researchers sent questionnaires to a sample of 200 high schools chosen at random throughout the country, and received a response from 79 teachers in 55 different schools. In this case, the respondents indicated that none of the schools had obtained GIS software, however 13 of the teachers indicated that they had used GIS software before, and 7 of these teachers had used the software in lessons. 66% did not have exact knowledge of what GIS is, and 82% indicated that they were unsure how to implement GIS in lessons. Again in this case, the majority (92%) of teachers said that they would use GIS in the future. In an attempt to improve the use of GIS software in Turkey, a book was written to provide teachers with the tools to implement GIS in classrooms [8]. In the first six months after its distribution, it was determined that supplying teachers with materials is not enough to encourage them to implement it in their courses.

Finally, in Singapore, a similar survey was completed in order to understand the status of GIS in the country's educational programs [9]. The results of this survey showed that 69% of respondents had some knowledge of GIS, but fewer than half had any training in GIS. Again, the majority (90.8%) supported the use of GIS in classrooms, feeling that it would enhance teaching and learning. The researchers suggest that intensive training is necessary to instruct teachers in the use of GIS technology, pointing out that entire undergraduate and postgraduate courses are taught in the use of GIS as an example of how challenging it can be to learn to use traditional GIS software.

2.2. Issues with GIS technology implementation

As can be seen in table 1, the support for the use of GIS software use in schools in all three countries was high, but the successful implementation of GIS in classes was low. The reasons for the low success rates can be divided into three major categories. The first category is lack of availability. In the US, Turkey, and Singapore, many of the schools surveyed did not have any GIS software available to them through their respective schools [6] [7] [9]. The second reason for low success can be attributed to lack of time. Teachers in the US and Singapore stated lack of time was a major hurdle in implementing GIS lessons in their classrooms. The final category could be described as a lack of knowledge. Teachers stated that they don't know how to implement GIS, that they have not been provided adequate training, or that they perceived the software to be too complex to implement effectively in their programs.

Table 1: Summary of Teacher Responses to GIS-Based Surveys

Teacher GIS-Based Questionnaire Responses By Country			
	United States	Turkey	Singapore
Sample Size	376	79	323
% Respondent schools with GIS software	100	0	43.8
% Believe GIS makes a positive contribution to education	88	76	90.7
% Who have used GIS software in lessons	54.9	8.9	11.8
Reasons stated for lack of use	Lack of time to develop GIS-based lessons	Teachers are unsure of what GIS is	Lack of curriculum time
	Little support for training and implementation	Teachers are unsure how to Implement GIS in lessons	Need for extra preparation time
	Perceived complexity of software	Teachers lack knowledge and skill to implement GIS technology	Lack of suitable instructional packages

The issues faced by educators and students can be generalized to the issues that are faced by any non-specialized user when attempting to use GIS technology. To increase the use of GIS software in among beginners, all three of the major issues need to be dealt with. First, the software must be available to the general populace and easily accessible. Second, it should not take too much time to access and understand. Finally, either more training should be provided to beginners, or there should not be a high level of specialized knowledge required in order to use the software. To generalize, GIS needs to be more usable in order to make it more accessible to beginners. In order to deal with both the time and knowledge issues, and considering the lack of success in providing additional training in Turkey [8], this paper focuses on the use of software which requires less specialized knowledge. The software should be easy for both teachers and students to understand and use. In reviewing the Singapore data, Suxia Liu and Xuan Zhu [10] note that many instructors feel the current software lacks flexibility and user friendliness. They point out that teachers and students are using the same software that is used by GIS industry professionals, and suggest that GIS technology should be adapted to support teaching and learning environments. This suggests that adaptations to traditional GIS software may not solve the issues of GIS lesson implementation, but adapting GIS technologies to suit the needs of non-specialized users may provide an easier and more time efficient way to introduce GIS concepts to beginners, and increase its use among those who have no interest in becoming specialists.

2.3. GIS user interface evaluation

In order to understand why traditional GIS technology may be viewed as complex by beginner or non-expert users, an example user interface can be reviewed and evaluated. A common method for evaluating user interfaces is via a cognitive walkthrough [11]. Cognitive walkthroughs evaluate the usability of user interfaces by asking questions from the perspective

of the user, in order to understand the system's learnability. This method requires a set of common tasks to be evaluated. Each task is then described from the point of view of the first time user, and the steps taken are evaluated based on the intuitiveness of the actions necessary from the user's perspective. Spencer [12] streamlined the process of the cognitive walkthrough by requiring only two questions for each step in the current task being evaluated:

1. Will the user know what to do at this step?
2. If the user does the right thing, will they know that they did the right thing, and are making progress toward their goal?

These questions can be used to evaluate the usability of traditional GIS user-interfaces. In order to do this, a simple task should be chosen. For example, viewing the roads in a given map.

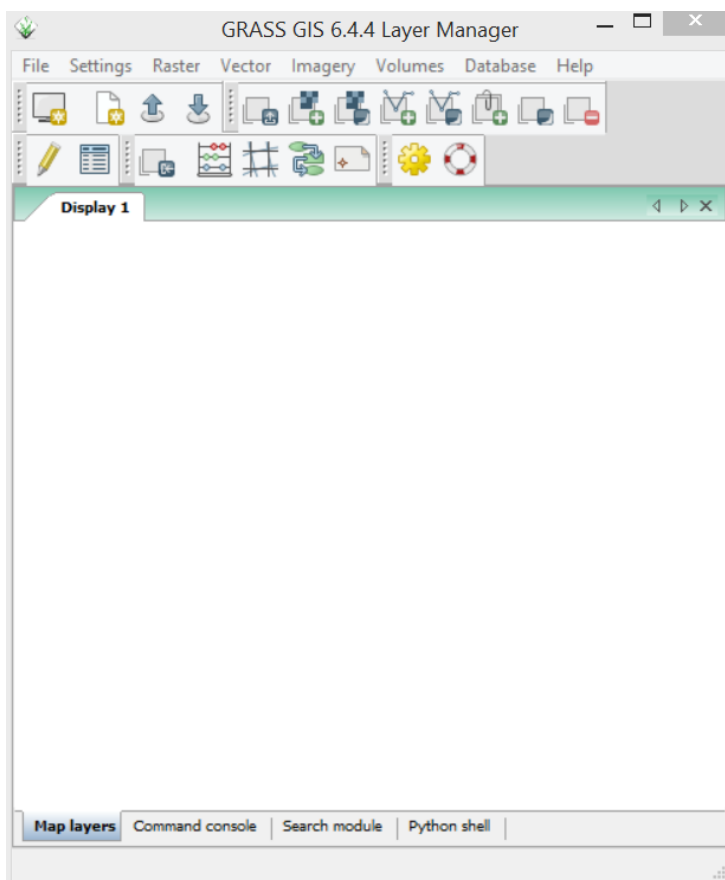


Figure 1: GRASS GIS Layer Manager Window

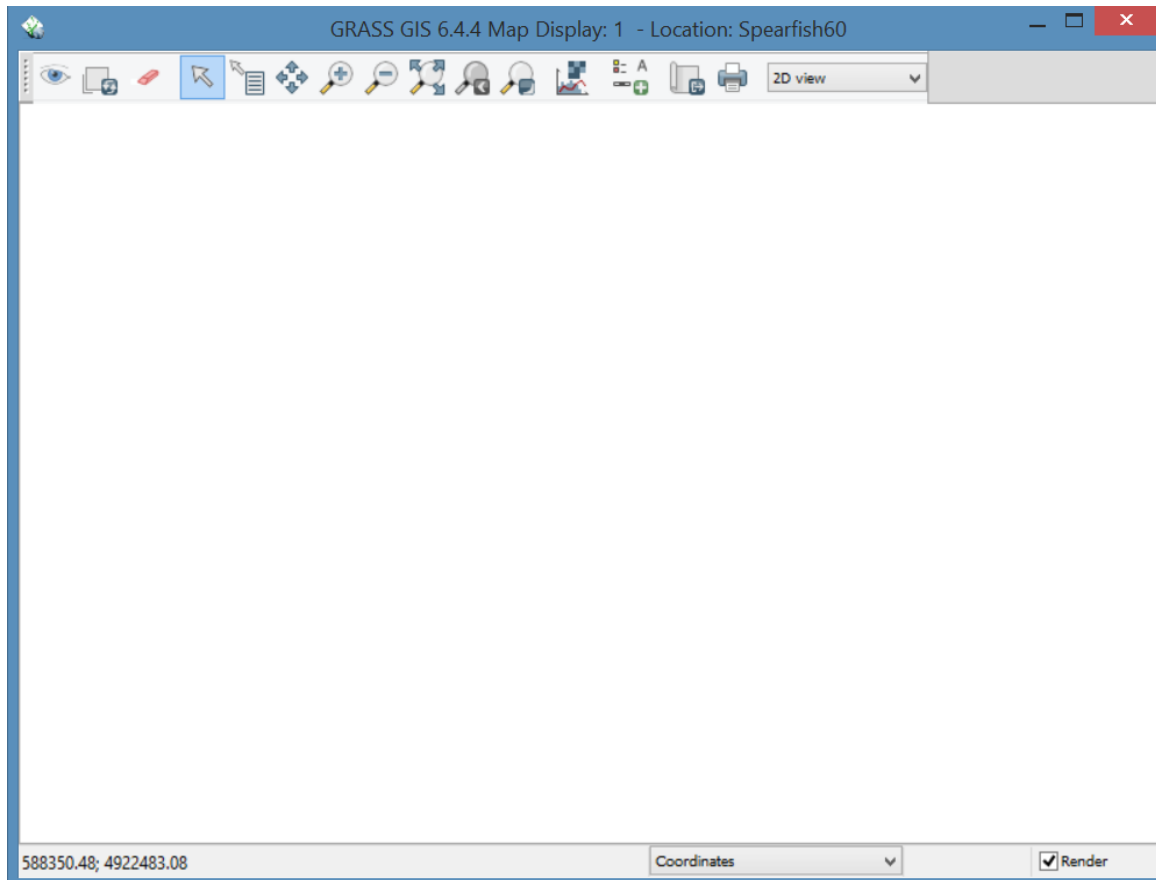


Figure 2: GRASS GIS Default Map Display

To evaluate this, a sample map set provided with GRASS GIS will be used. Upon opening the Spearfish data set in GRASS GIS, the user is presented with the two window interface shown in figures 1 and 2.

From the perspective of the beginner user, the question of whether or not the user will know what to do at this point would likely be no. However, through browsing menus, the user can find an option under “File” called “Map display”. Upon opening the sub menu for this option, the user is given the option to “Add raster” or to “Add vector”. From the perspective of a beginner, the meaning of raster and vector may be unknown, so again the answer to the first question is likely negative. Once a user determines which option they would like to choose, they are brought to the window shown in figure 3. This window allows the user to select map names

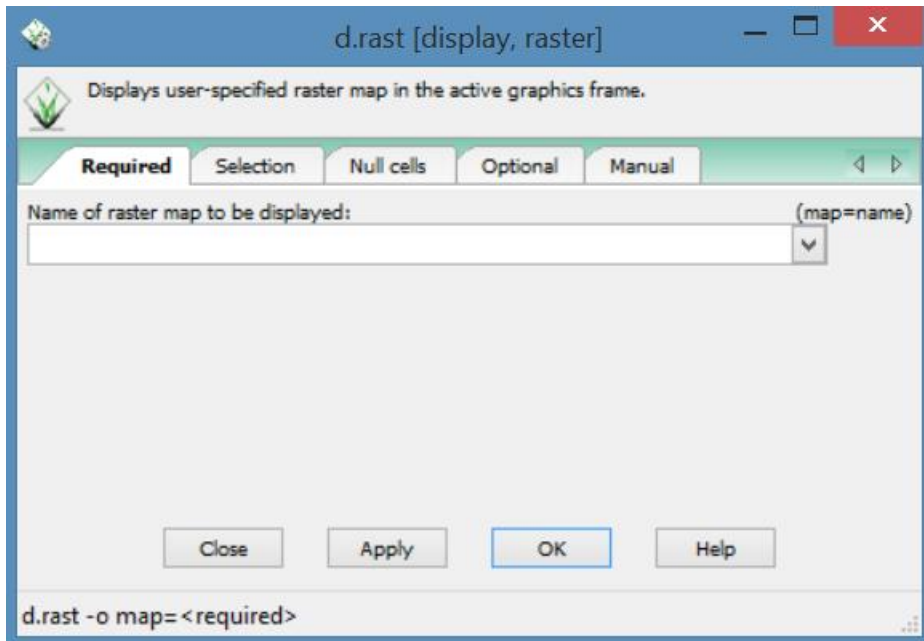


Figure 3: GRASS GIS Raster Map Display Window

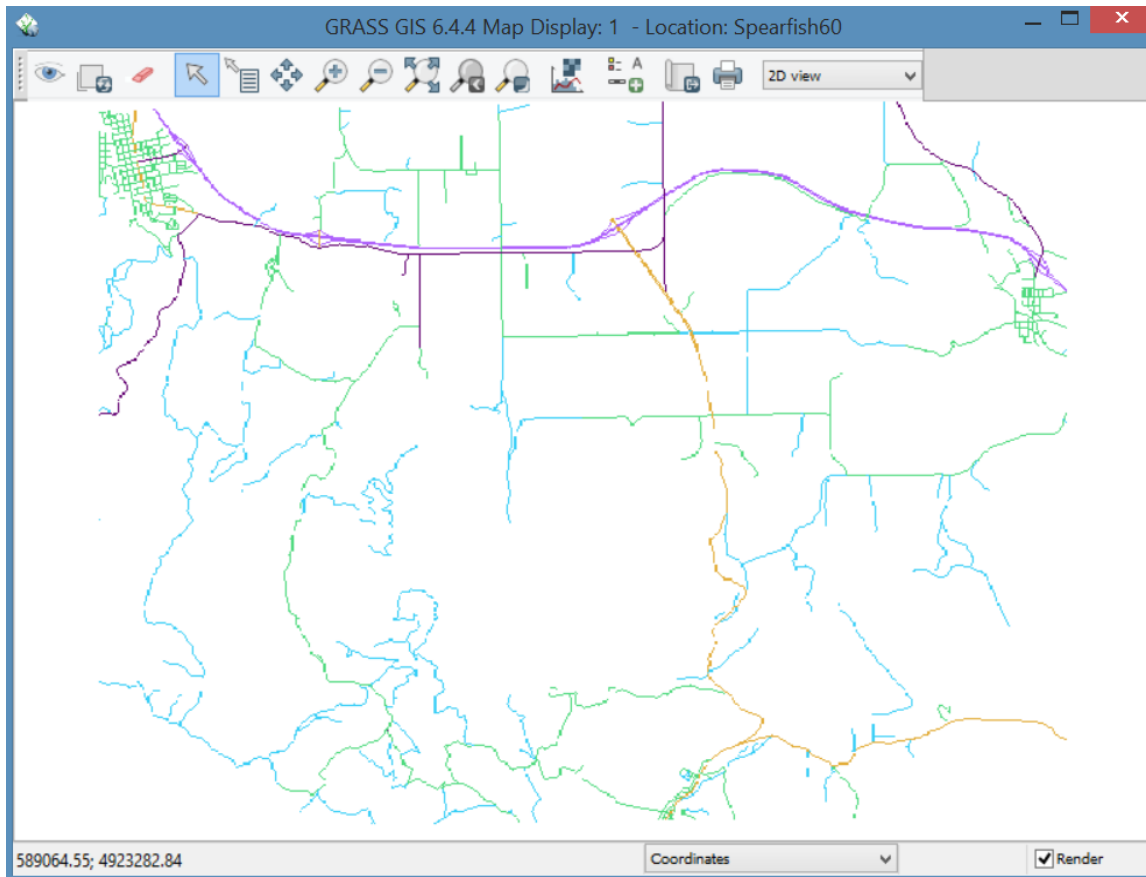


Figure 4: GRASS GIS Map Display with Roads Displayed

from a drop down box, one of which is “roads”. Upon selecting and applying the roads layer, the user is presented with the map display window shown in figure 4. At this step, the user can answer affirmatively to the second question posed by Spencer [12], as it is the first time that output is shown from a user action which relates to the user goal.

Many of the steps involved in this process assume knowledge on the part of the user. Gao and Goodchild [13] state that the interface “relies on the assumption that the user has a good sense of data format and knows exactly which operation is suitable for their needs.” They suggest that a simplified user interface is necessary for non-expert users, allowing them to get answers to spatial questions they may have, without needing specialized knowledge of GIS concepts. This method of simplifying the user interface would alleviate the issues beginners face when first exposing themselves to GIS concepts. By making sure that a beginner knows what to do next and also knows that they are making progress toward their goal throughout the process, they will be less likely to feel that the software is overly complex.

3. CONCEPTS

3.1. Virtual globes

Virtual globes are digital representations of earth in three dimensions. Virtual globes date back to VRML (Virtual Reality Modeling Language), which was released in 1995 and was designed as a language which allowed for representing animated 3D virtual worlds [14]. However, the use of VRML to construct 3D environments required a high degree of programming skill. Due to this, it was not until the release of modern virtual globes such as Google Earth, NASA World Wind, and ArcGIS Explorer that virtual globes began to enjoy popularity that they currently maintain.

Virtual globes have many potential uses. Many include global weather information in addition to high resolution images of the Earth's surface. They have been used for tracking animal movement over the course of several years, as well as tracking temporal changes in ice density and drift [15]. While the potential uses of virtual globes are varied, a survey of non-expert users showed that most people use them for observational purposes; this includes activities such as viewing one's house, which does not require the type of questioning about why phenomena occur that improves spatial thinking [16].

Google Earth is the most popular example of a virtual globe. As of 2011, Google Earth reached and surpassed one billion downloads [17]. Google Earth was originally an application which was created by Keyhole, Inc., which was founded in 2001 [14]. While Keyhole's virtual globe was used by news agencies and the United Nations, it wasn't until Google purchased Keyhole, Inc. in 2004 that Google Earth started to achieve its current status. In addition to imagery of the Earth's surface, Google Earth offers users a chance to view Mars, the Moon, and other celestial bodies as part of the desktop application. Google Earth can be used to lay

Placemarks and Polygons on top of the background imagery, allowing users to add their own data to the application. Figure 5 shows an example of a virtual globe, in this case Google Earth, with a Placemark added showing the location of the NDSU campus.

NASA World Wind and Esri ArcGIS Explorer have many of the same features as Google Earth (see table 2). All virtual globes include high resolution satellite imagery and support for multiple layers. All three are free to download and use, however NASA World Wind is the only one which is open source, allowing for more modification of the software. It is possible to import KML/KMZ and Shapefiles into all three, provided that the user owns Google Earth Pro, as Shapefile importing is not possible using the basic version of Google Earth. The differences between the applications come from the details in focus and implementation. Because NASA World Wind is open source, it focuses on its role as a software development toolkit and is aimed at developers [18]. Esri's ArcGIS Explorer is built as a crossover tool where you can easily



Figure 5: Google Earth with Placemark Added at NDSU Campus Location

Table 2: Comparison of Basic Features of Three Major Virtual Globes

Virtual Globe Feature Comparison			
	Esri ArcExplorer	Google Earth	NASA World Wind
High resolution satellite imagery	●	●	●
Support for multiple layers	●	●	●
Open Source			●
Free	●	●	●
Shapefile import	●	●*	●
KML/KMZ import	●	●	●

*Available only through Google Earth Pro

import and share data from other Esri applications [19]. Google Earth provides more navigation and query functionality and a greater number of default layers available for use.

There are several potential benefits to the availability and use of virtual globes. One proposed reason for the success of virtual globes is that they provide a pseudo-3D environment which “allows people to interact in an environment that they naturally comprehend and that makes the data and information presented easier to understand” [14]. In his *Nature* article, Declan Butler describes virtual globes as “an easy way into GIS software” [15]. One of the major hurdles of implementing GIS among non-specialized users is that the software is viewed as complex, or more generally that users don’t have the background knowledge and experience to use GIS. Since virtual globes are user friendly, non-specialized users can spend more time using the application and less time learning how to do so.

Not only are virtual globes easier to understand, but it is also typically very easy to gain access to these products. Google announced in early 2015 that in addition to their regular desktop version being free, they were also making Google Earth Pro free [20]. NASA World Wind is not only free, but also open source [21]. Esri’s ArcGIS Explorer can also be obtained at no cost, and since Esri also provides traditional GIS software, the cost in this case can be directly compared. Esri’s ArcGIS Online currently costs \$2,500 per year for five users, while their desktop version

costs \$1,500 for a single license [22]. There are, however, free versions of traditional GIS software. One example of this is GRASS GIS, which, like NASA World Wind, is both free and open source [23]. While free traditional GIS software exists, the most commonly owned traditional GIS software is not free. Esri is the market share leader in GIS software products. ARC Advisory Group reports that Esri has a 43% share in the current GIS market [24], which is a strong lead before any other company. So, while free GIS software exists, it is not as commonly used as the traditional fee-based GIS software licenses.

3.2. Comparison of virtual globe and traditional GIS functionality

Virtual globes provide high resolution images of the Earth's surface along with some layers and tools for creating information, however they lack data management and analysis functions. In traditional GIS software, these two features form a large part of the functionality of the application.

3.2.1. Data management

While the explicit implementation of data management varies between GIS software, traditional GIS software provides some form of data management. In the case of the free and open source GRASS GIS, basic SQL support is provided with commands available within GRASS GIS to connect to various DBMS backends as well as to import and export data [25]. Esri provides additional data management functionality, allowing users to store data in the Esri cloud in addition to databases [26]. In addition to this, Esri provides functionality such as support for achieving a company's security requirements and managing metadata. In comparison with traditional GIS software, virtual globes typically provide little to no support for data management. For example, Google Earth provides a data management guide as part of their

online support, which gives insight into best practices regarding data management based on insights provided by customers, but the tools provided are limited [27].

3.2.2. Data analysis and visualization

Traditional GIS software provides numerous tools for data analysis, from statistical analysis to the display of graphs and thematic maps. GRASS GIS, for example, provides capabilities for both raster and vector analysis [28], as well as an interface to the R programming language which allows for statistical analysis. GRASS GIS provides functionality in order to create charts such as bar charts, pie charts, and histograms as part of their base application. Esri ArcGIS Desktop provides additional graphing tools, as well as spatial calculations and statistical analysis of spatial data [29]. Table 3 shows a comparison of the graphing capabilities of these two applications. Spatial data visualization and analysis tools are not available in virtual globe software. Despite the lack of analysis tools, it is possible to add layers of pre-processed data to Google Earth and other virtual globes through the use of KML files. Though even in doing so, the additional functionality is more limited than with traditional GIS, as the additions are limited to KML objects and their functionality.

Table 3: Graph Types Available in Two Major GIS Software Packages

Graph Types Available in Base GIS Software		
	ArcGIS Desktop	GRASS GIS
Bar	●	●
Line	●	
Area	●	
Histogram	●	●
Scatterplot	●	
Box Plot	●	
Bubble	●	
Polar	●	
Pie	●	●

3.3.3. Macros in GIS

Through the use of macros, users of traditional GIS can add usability back into the application. Macros allow for common steps to be performed in sequence, as well as customizing the user interface so that simplified views are made available to users based on their needs or knowledge base. This is done programmatically, so it must be done by someone who can write commands in the language specified by the application (e.g. Python or Visual Basic). The knowledge of how to implement macros is limited to specialized users who have knowledge of that language.

Virtual globes, on the other hand, have a limited functionality, and in general the user interface cannot be edited through the use of macros. Using implementations such as the deprecated Google Earth API allow programmers to edit these interfaces to suit their needs by inserting Google Earth into their web page, however applications such as Google Earth do not allow for this. While this limits specialized users, it means that the user interface of virtual globes is kept simpler for non-specialized users. Increased options might confuse users regarding functionality, whereas a simple user interface that is easy to learn and use.

3.3. KML

KML (Keyhole Markup Language) is an XML language file format which is used to display geographic data. It was created by Keyhole, Inc., and was kept in use by Google when they bought the company. Google submitted KML to the OGC (Open Geospatial Consortium), a step that was approved in 2008 [30]. This shifted KML into an open standard for visualizing geographic information, which is maintained by the OGC. The standardization of KML makes it an ideal tool for geographic visualization. It is unlikely to change, and is not controlled by Google. Esri's Shapefile is standardized, but still maintained by Esri.

3.3.1. KML format

KML allows for the creation of Placemarks, Ground overlays, Paths and Polygons within Google Earth. Placemarks are single points which are placed within a map. Ground overlays allow images to be placed on the surface of the Earth, following its terrain. Paths are two dimensional lines which can include a number of points, while Polygons allow for the creation of basic 3D shapes, such as buildings. Any information entered in Google Earth can be saved as a KML file, or as a zipped KML file, known as a KMZ file. KML files must include XML headers and follow a hierarchical order of elements. The structure of this hierarchy must follow KML standards.

Figure 6 shows an example of a Placemark which is set at the location of the NDSU campus. Line 1 is the XML header, which is included in all KML files. Line 2 gives the KML namespace declaration, which is also part of every KML file. Lines 3 and 9 are the starting and ending statements for the Placemark object. Within the Placemark, the name is the label for the Placemark, in this case “NDSU Campus”. The description is optional and can include a description of the Placemark. A Placemark object includes a Point object, which contains coordinates for the Placemark. The coordinates include three values for longitude, latitude, and altitude.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <kml xmlns=http://www.opengis.net/kml/2.2>
3   <Placemark>
4     <name>NDSU Campus</name>
5     <description>Placemark showing the location of the NDSU campus</description>
6     <Point>
7       <coordinates>-96.8033967776714,46.89361084996413,0</coordinates>
8     </Point>
9   </Placemark>
```

Figure 6: Example of KML Language

3.3.2. KML vs. Shapefile

Shapefiles are similar to KML files in that they can be used to create points, lines, and Polygons within GIS software. Shapefiles were created by ESRI, and contain three files – .shp, .shx and .dbf. Shapefiles are binary, so the files are smaller than KML, but cannot be read as text files. Because they include multiple files, they are not as easy to transfer as KML, being that the XML language is made for simple usability across the internet.

4. GOALS

The primary goal of an introductory GIS application is to provide an introduction to some tools which are similar to those available in traditional GIS software, while also addressing the issues currently faced by organizations as well as individuals when attempting to learn and/or teach GIS technology concepts to non-expert users. Improving accessibility and usability of technology will simplify the introduction to GIS concepts, opening up an introductory route to GIS concepts, as well as expanding the use of currently available technology. The following sections discuss how different concepts address the issues of usability and accessibility, as well as the current limitations which should be addressed by the application.

4.1. Virtual globes

To assure that the technology is accessible to a broad user-base, the software should be easy to obtain at little to no cost to the user. Virtual globes address this, as all of the major virtual globe products are free to download. The only limitation to the accessibility of virtual globes is that the user must have internet access in order to attain them. Virtual globes are also considered to be “an easy way into GIS software” [15], and are generally viewed as user-friendly. This addresses the issue of knowledge, as less knowledge is required in order to use virtual globes.

The limitation of virtual globes is that they lack the data analysis and visualization capabilities of traditional GIS software. The software must address this limitation by implementing some data visualization capabilities. Schöning et al. [16] state that the “the spatial representation of data permits the individual to ask ‘why’ questions.” The assumption is that asking “why?” of the data improves spatial thinking. Implementing data visualization would help users ask questions of the data by allowing them to view patterns in geographic data. However,

the implementation of geographic data visualization must be done in a way that maintains the usability of the virtual globe software.

4.2. Sample data sets

The increasing availability of data online allows for easy access to geographically based information. For example, the WHO (World Health Organization) provides a data repository with information regarding global health and disease [31], and UNEP (United Nations Environment Programme) provides environmental data for download from their data sets [32]. The availability of data sets such as these allow users to easily access geographic data from multiple sources. However, instructors note that lack of time is an issue in implementing GIS concepts in the classroom. Despite the availability of online data sets, the software should address the issue of time by providing sample data sets which can immediately be used by the application, without the need for any modification. This makes the application usable immediately after its acquisition.

Additionally, as more data sets are created and used by instructors, the data should be in a format which is commonly used and easy to share, such as .xls/.xlsx file format. The additional benefit of using this format is that many online data sets allow data to be downloaded in this format, or as a .csv file, which can then be converted for use in software applications such as Microsoft Excel. Data in spreadsheet form is easily readable and easy to manipulate.

4.3. Querying capability

Geographic data requires geographic coordinates to be placed on a virtual globe. Users should be able to enter this information into or use data which already includes this information in order to easily and dynamically place their data in a geographic location. However, simplifying use in order to save time for beginner users is one of the main goals of this

application. Because of this, some querying capabilities should be included in the application in order to allow the user to input names of some location and receive geographic coordinates based on the name value. This can be achieved by adding country names, for example, to a database along with that country's latitude and longitude.

4.4. Google Earth

Because Google Earth is the most popular virtual globe, it can be said that it is also the virtual globe with which the greatest number of users are familiar. Familiarity with the application allows for users to spend less time learning how to use it; it is also more likely that instructors are already familiar with the basics of its use. This partially addresses the problems teachers have stated of both lack of time and lack of knowledge. Because of this, the application should focus on its capacity to integrate with Google Earth. The application output will be demonstrated in Google Earth to show that it integrates effectively with Google Earth.

Because the Google Earth API is being retired in late 2015 [33], and may lose browser support before that time, the application will not integrate with the Google Earth API, but will instead be a standalone desktop application which can be used with Google Earth Desktop.

4.5. KML

KML is an OGC standard, and was also developed for use with Google Earth. KML files are also easy to open within Google Earth. To open a KML file in Google Earth, the user only has to go to "File" in the menu bar, and from there select "Open". The user can then browse their computer and select a KML/KMZ file to open within the application. For these reasons, little knowledge is required in order to use these files. The application can therefore generate KML files which can be opened in Google Earth, or any virtual globe which supports KML files, including NASA World Wind and Esri ArcExplorer, without the need for any knowledge of how

the files function. However, since KML files can be viewed as text files, those interested in viewing them can do so.

4.6. Data visualization

The application should provide some basic functionality which is also available in traditional GIS software, and which can also be implemented using KML. Since data analysis (such as statistical analysis) tools could not be added using KML objects, the application should focus on data visualization. A good candidate for this is the use of KML Polygon objects to create charts, such as bar and pie charts, which are available in traditional GIS software. Basic bar charts have been created in Google Earth [34], but the use of this idea can be extended to additional chart styles. The use of KML Polygons can be extended to create stacked bar charts, 100% bar charts, and pie charts, for example. These charts can be created dynamically, giving the user more control over what is placed where, and how it looks within Google Earth. Since Google Earth supports adding multiple KML layers, multiple charts could be created and opened in Google Earth, then toggled on and off easily by the user.

4.7. Summary of goals

The focus of the application is creating a standalone desktop application which allows users to dynamically place geographic data within Google Earth in the form of bar charts, stacked bar charts, 100% bar charts, and pie charts. The charts will be imported into Google Earth by the user, using KML files generated by the application. The application will read .xls/xlsx files generated or attained by the user. These files will be in specified formats, which will be specified within the application. If the data is based on country or US state data, the application will perform a query which looks up the latitude and longitude of the location. The principal goal of the application is to provide users an introduction to GIS concepts in a way that

Table 4: Functional Requirements for Chart Generating Application

Functional Requirements	
Req #	Requirement
1	The application shall be a standalone desktop application
2	The application shall read Microsoft Excel files
3	The application output shall be compatible with Google Earth
4	The application shall produce KML files
5	The application shall allow the user to produce bar charts, stacked bar charts, 100% bar charts, and pie charts based on geographic data
6	The application shall provide basic automated query functionality for finding country and US state latitude and longitude
7	The application shall allow the user to input their location's latitude and longitude
8	The application shall provide sample data sets from publically available data

is easy to learn and understand for non-specialized users. A summary of the functional requirements of the application can be seen in table 4.

5. SOFTWARE DESIGN

The software is written as a Windows Form application based in C#, due to C# having the necessary libraries available for implementation, having the tools available to implement a form application in C#, and familiarity with the language. This chapter gives an overview of the software architecture and implementation.

5.1. Design overview

In addition to guiding users through forms and allowing users to choose specific details about how the KML will be implemented, the application has four major components.

5.1.1. File reader

The application reads files in Microsoft Excel's .xls/.xlsx format. The decision to use this format is based on the readability of spreadsheets. Additionally, spreadsheets are commonly used, easy to edit, and can often be downloaded from online data sources. Additionally, .csv type files can be converted to .xls/.xlsx files using applications such as Microsoft Excel. The application allows for a series of specific formatting options, based on manual input of latitude and longitude, country information excluding latitude and longitude, and US state abbreviations excluding latitude and longitude. The valid formats that can be read by the data file reader function can be seen in table 5. The Category value is required, and provides the program information regarding the type of information being uploaded. This can be any information

Table 5: Valid User Input Formats

Valid Input Formats for Data File				
Major Input Category	Column 1	Column 2	Column 3	Column 4
Manual	Category	Latitude	Longitude	Value
Country	Category	Country Name	Value	
	Category	Country Code (ISO2)	Value	
State	Category	State Abbreviation	Value	
	Category	State Name	State Abbreviation	Value

regarding the type of information being uploaded, such as the category or year of the information provided. The latitude and longitude must be in signed degrees format, which means that the values are decimals ranging from -90 to 90 in the case of latitude, and -180 to 180 in the case of longitude. Both country codes and state abbreviation must be two characters long. The format being used can be set by the user prior to the call to the function which reads the file. If the application is not read successfully due to incorrect formatting, the user is shown an error and allowed to review their file format and try uploading the file again. Once the file has successfully been read, the database can be queried, if necessary.

5.1.2. Database connection

The second major component of the application includes a database connection and data cleanup. In any case where latitude and longitude are not included in the data file, a database is available with some queries in order to find the latitude and longitude values for countries and states. The country and state latitude and longitude information is stored in a SQL Server .mdf file within the project itself. Figure 7 shows the tables included in the project. Country queries allows for querying by the name of the country or the two digit ISO code associated by the country. US state queries allow for querying by the two character state abbreviation. After loading the file information into a list, a query is performed when necessary to retrieve latitude and longitude values. The user is then shown the results of this process in an editable table. Any

Country	
PK	<u>countryID</u>
	countryName latitude longitude

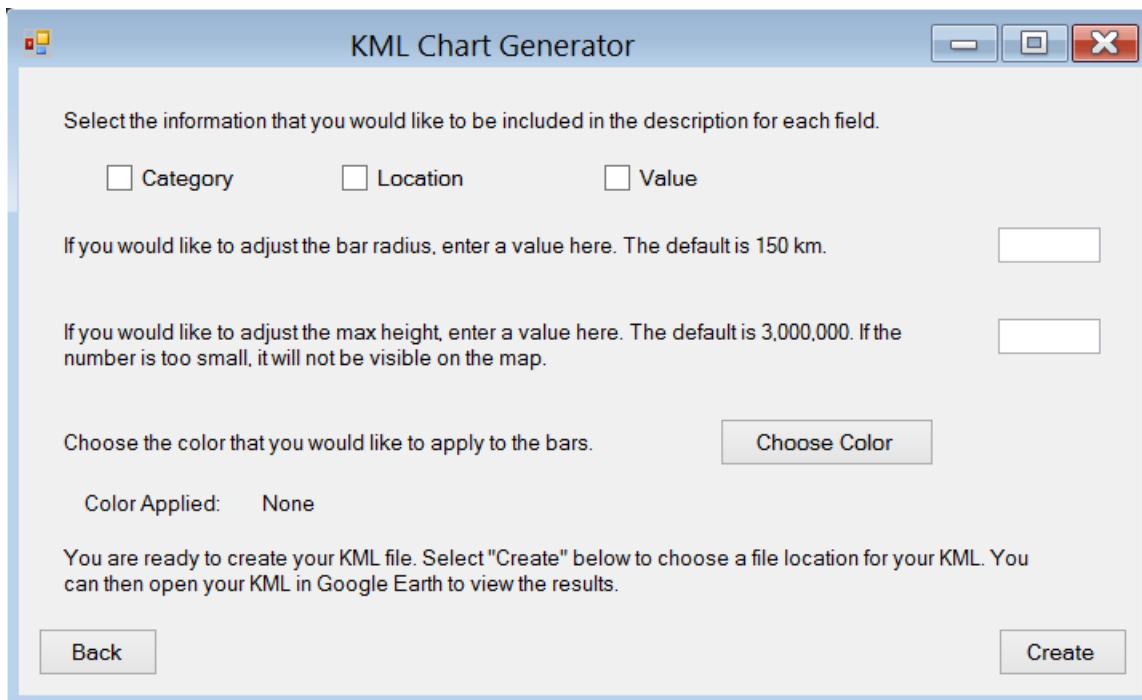
USState	
PK	<u>stateID</u>
	latitude longitude

Figure 7: Database Tables for Querying Latitude and Longitude

time the query returns zero or multiple results, the user is prompted to manually input the latitude and longitude values. Once the user has reviewed the data, they are prompted to choose a graph type and the corresponding options for that graph.

5.1.3. User options

The third component of the application allows users to set graph options. In addition to choosing the type of graph that they desire, they can also input options such as graph title, colors, and the resulting radius of the graph. In the case of the radius, a default value is included. This gives the user some control over the resulting graph, allowing for personalization by the user. In figure 8, some of the options associated with the generation of bar charts can be seen. The first set of options allows users to choose which text values will be included in the chart. After that the bar radius and maximum height of the chart can be set. The height is based on altitude values, and is set to a default of 3,000,000. The color of the bars can also be set here; once all options have been set, the user generates the file.



The screenshot shows a window titled "KML Chart Generator" with standard Windows window controls (minimize, maximize, close). The main content area contains the following elements:

- A heading: "Select the information that you would like to be included in the description for each field."
- Three checkboxes: "Category", "Location", and "Value", all of which are currently unchecked.
- A text input field with the label: "If you would like to adjust the bar radius, enter a value here. The default is 150 km."
- A text input field with the label: "If you would like to adjust the max height, enter a value here. The default is 3,000,000. If the number is too small, it will not be visible on the map."
- A "Choose Color" button.
- A label "Color Applied:" followed by the text "None".
- A paragraph of instructions: "You are ready to create your KML file. Select 'Create' below to choose a file location for your KML. You can then open your KML in Google Earth to view the results."
- Two buttons at the bottom: "Back" on the left and "Create" on the right.

Figure 8: Example of a Form Which Allows User to Set Chart Display Options

5.1.4. KML generation

One of the prerequisites for a programming language is that it have a library which supports KML generation. SharpKML [35] is a library developed in C# which supports the reading and writing of KML/KMZ files. This library was used in order to generate the final KML files. The final major component of the application is the generation of these files. When the user completes the process and chooses to generate their file, the application uses the KML library to generate a file based on the inputted values.

5.2. User interface flow

The flow of the application can be understood by viewing and understanding the structure of and relationships between the forms in the application. Figure 9 shows the forms and their relationships. The following section describes these forms, in order to provide an outline of the overall structure of the application.

5.2.1. Initiating the application

Upon opening the application, the user begins with the Start form. This form provides a short introduction to the application and its use, along with two options for data to use. The first option allows the user to use the sample data. Choosing this option opens a separate form, which allows the user to select from multiple options from the sample data file. If the user selects this option, they then move to reviewing the data. The second option allows users to select their own data file, which includes the selection of format options.

5.2.2. Select user file

Selecting a user file allows the user to select from multiple format options. The user can select an option to view information regarding the format, if they are unfamiliar with the application. After selecting the file from a dialog box, the application checks the worksheets in

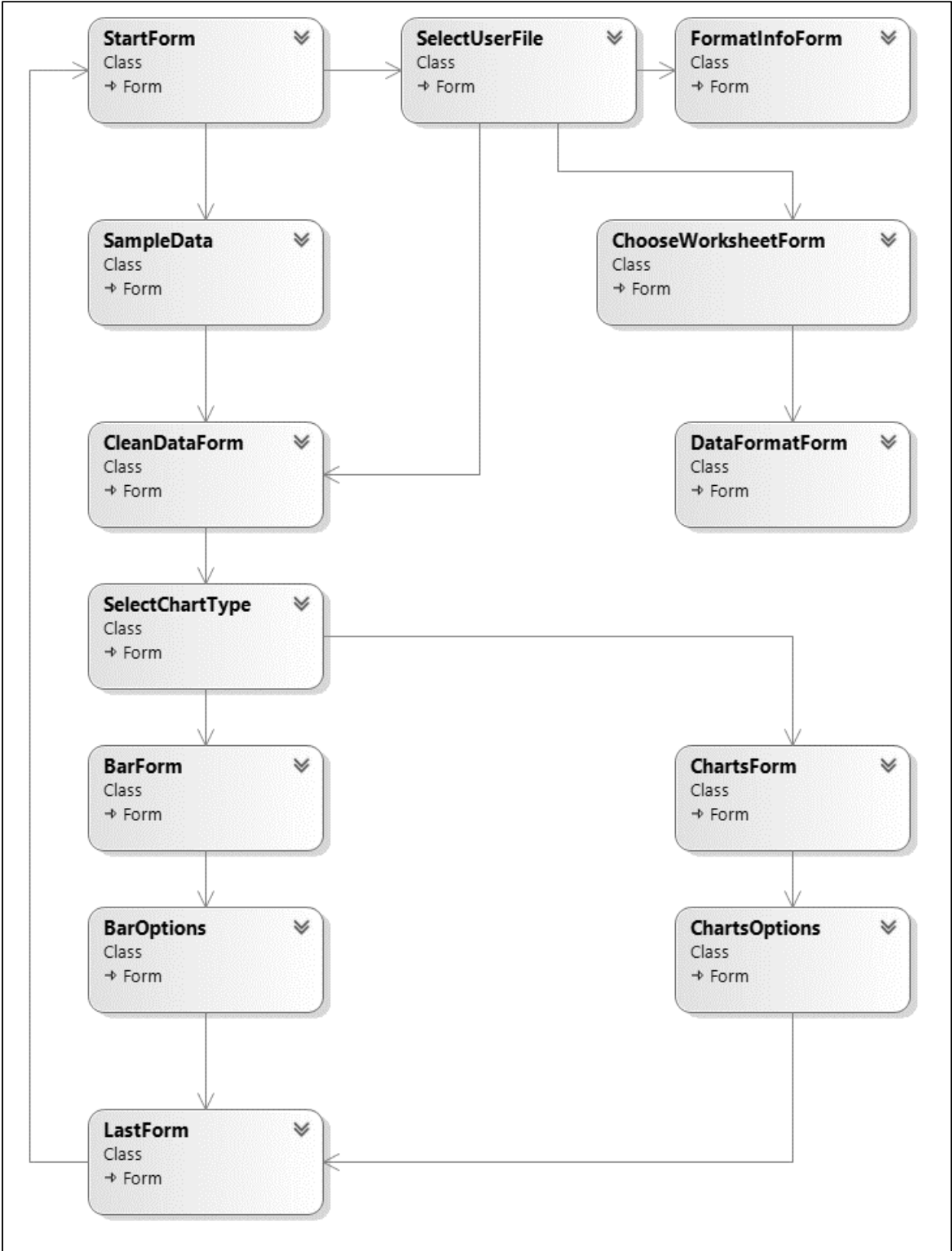


Figure 9: Description of User Flow through Forms within the Application

the file. If the data file contains multiple worksheets, the user must first select the worksheet that they would like to use from a drop down box. Once the user has selected their worksheet, they then select the format of their file. The application then attempts to upload the information in the selected format. If the data file does not match the expected format, the user is prompted to review their data format, then try again. Once the file has successfully been uploaded, the user can continue to the next step in the application. Any querying that is necessary occurs during this step.

5.2.3. Editing and reviewing data

Whether the user has selected their own file, or the sample file, they have a chance to review and edit the data once it has been uploaded. If the database query does not provide a match for the data input, the application highlights the blank fields and prompts the user to input

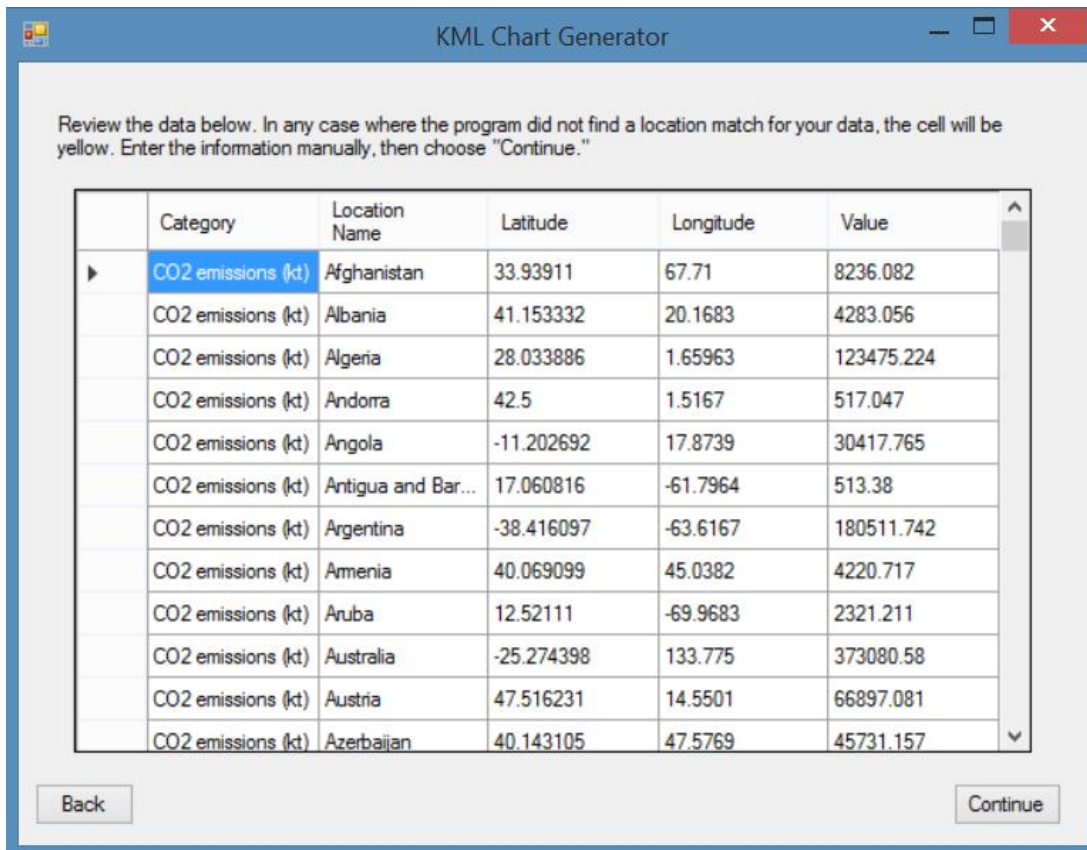


Figure 10: Example of the Review Step within the Application

the values manually. An example of this form can be seen in figure 10. In this case, any missing data would have blank entries for latitude and longitude, which would be highlighted in yellow until the user adds the necessary information. Once the user assesses that all data is correct, they can continue on to the chart options steps.

5.2.4. Setting chart options

Once a user has selected the chart type that they would like to create, they are brought to one of two forms. The first is the form which handles simple bar charts. This form is separate from the others because it does not handle subfields, as there is only one data category that is represented in the chart. The charts form handles all other charts. In both forms, the user selects the category or categories from the dataset which will be represented in their chart along with the title of the chart. Once this has been completed, the user moves on to the options forms. In both the bar and other charts forms, the user can set which text fields to include in the description field of the chart, along with color and size options for the resulting Polygons. Once all fields have been set, the user can generate their KML file. The user then has the option to start again with a new file or exit the program.

5.3. Class overview

Each time the application performs operations such as reading files, setting formatting options, and generating KML files, backend logic must be performed by additional classes. An overview of the classes, along with their properties and methods can be seen in the class diagram shown in figure 11. This diagram does not show the methods associated with forms, because these methods are associated with specific actions (e.g. *FwdBtn_Click()*), and are often repeated from form to form. This maintains greater readability of the diagram, as well, as including form methods causes the classes to be crowded and necessitates that they be smaller to fit on a single

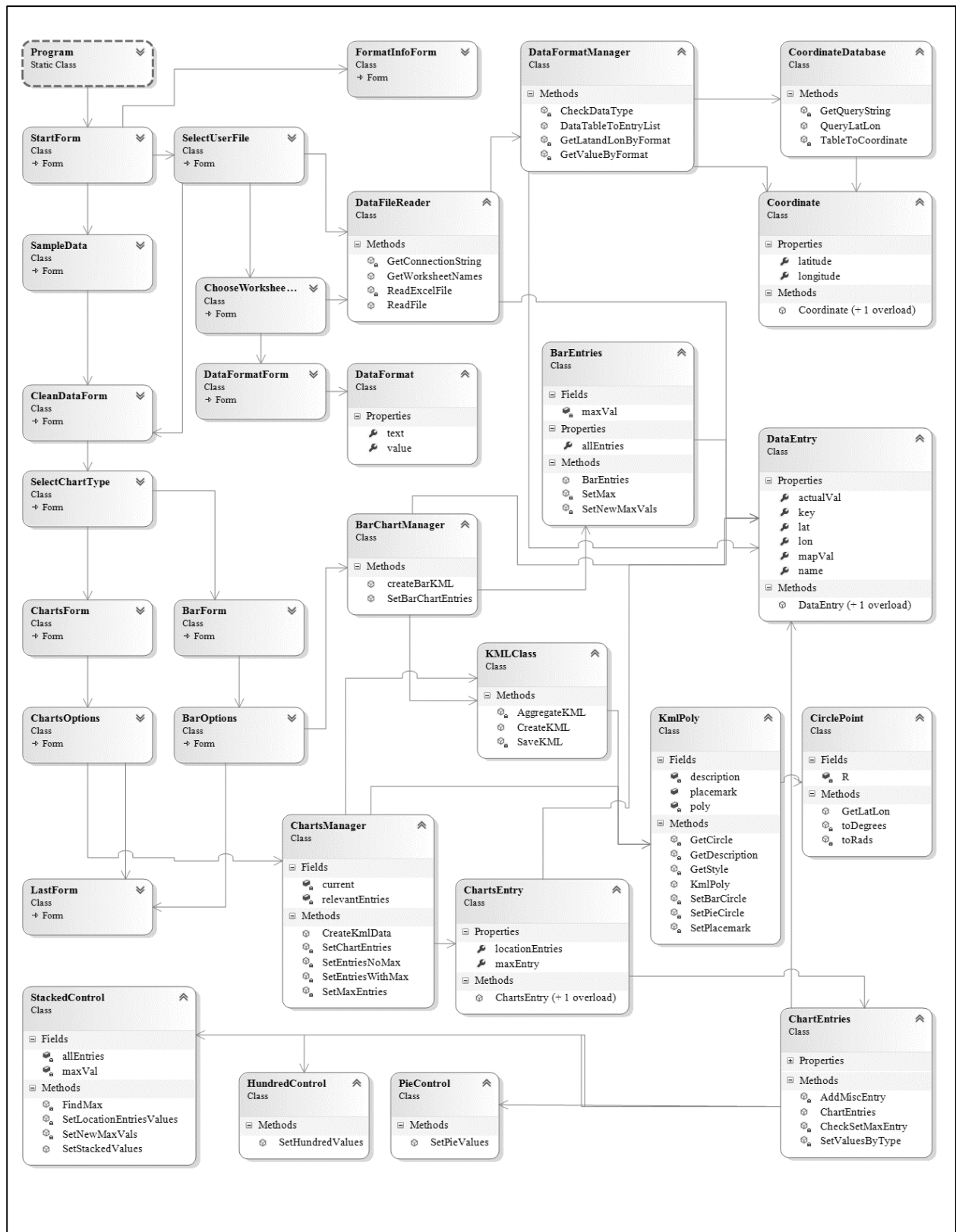


Figure 11: Classes within the Chart Generating Application

page. This section will describe the application with regards to the classes and the methods within those classes, focusing only on the classes which are not forms.

5.3.1. Reading user files

The *DataFileReader* class functions as an interface between the forms and the file reading and database querying logic. This class first allows *GetWorksheetNames()* to be called, which checks for all of the worksheets associated with the file. The worksheets are ready before any of the information within the worksheets is reviewed. Once the user has selected a worksheet, *ReadFile()* is called, which queries the file and selects all rows in the user-selected worksheet, then calls the *DataFormatManager* class in order to structure the file in the format selected by the user. Calling *GetLatandLonByFormat()* uses the selected format to query the database for coordinates which match the given identifier, through the use of the *CoordinateDatabase* class. Both of these classes use the *Coordinate* class in order to pass latitude and longitude values as a single object. Calling the *QueryLatLon()* method first gets a standardized query string based on the given format, queries the database, then converts the results to a coordinate value. The resulting data table is converted to a list of instances of the *DataEntry* class, which includes all of the information that will be used to create each KML Polygon. This class is used as a basic structural class for the data throughout the application. All values of this class are set once the database has been queried, except for the *mapVal* property, which is set after the user has selected the type of chart and chart options for the file, because this value depends on the type of chart. Similar to the *Coordinate* class, the *DataFormat* class is primarily used in order to pass information as a single object. In this case, it passes the user-selected format with text and associated numeric values, so that the program can perform equality checks based on integer rather than string values.

5.3.2. Chart generation

The rest of the classes are associated with the KML generating portion of the application, and are called after the user has set all desired options for their chart. While the methods for each chart type are similar, slightly different approaches are taken for each.

In the case of bar charts, the *BarChartManager* class uses the *CreateBarKML()* method to first select all values from the category specified by the user and create a *BarEntries* object. This object finds the highest value within the bar chart, and sets that to the maximum value. The *mapVal* field is then set based on this maximum value and its proportionate value in relation to the maximum value. For example, if the maximum value in the list is 100, and the current *DataEntry* object has an *actualVal* value of 30, then the *mapVal* for that entry would be 900,000, assuming that the maximum altitude remains its default value of 3,000,000. This equation can be defined as follows:

$$\text{mapVal} = \frac{\text{actualVal} * \text{maxAltitude}}{\text{maxVal}}$$

Once the map values for each entry have been set, the *BarChartsManager* class then handles adding each entry to a new list of *KMLPoly* class instances. This class uses the *DataEntry* values to create a KML Polygon. Each *KMLPoly* object contains a *description* field, a *Placemark* field, and a *poly* field. These fields are set based on the user input. For example, the description field can contain any of the following three: the title or category value of the chart, the name of the location, and the actual value of the data point. In the case of the bar chart, each location corresponds with a single Polygon, and each Polygon's boundaries are set as a cylinder surrounding the latitude and longitude values of the location. The radius of the cylinder has a default value of 150 km, but this can be set by the user as well. Once the *BarChartsManager* has

created *KMLPoly* objects and added each to a list, it then calls the *KMLClass*, which aggregates the Polygons, then saves this aggregation as a file in the location chosen by the user.

In the case of all other chart types, the *ChartsManager* class is called. In the case of stacked bar charts, 100% bar charts, and pie charts, the user has an option of setting their own total value field, or using the sum of all of the fields included as a total value field. Because of this, the *ChartsManager* class has two methods which handle each of these cases. The *SetEntriesWithMax()* finds each relevant record for the given location, and adds it to its own list using the *ChartsEntry* class, which consists of each subfield of the location, as well as a maximum value field. The *SetEntriesNoMax()* method adds each subfield to the *ChartsEntry* object, and calculates the maximum value field by summing the value fields for the location.

Once each *DataEntry* object for a specified location is added to a *ChartsEntry* object, the *ChartEntries* class is called. This class includes a method which calculates any difference between the maximum value field and the sum of the subfields, and adds an “other” field to the data, if necessary. The map values for each *DataEntry* is then calculated, by calling a separate class for each of the types of charts. Stacked bar charts use the *StackedControl*, which finds the largest total value for all of the locations, and sets that to the maximum altitude. The *HundredControl* class sets each maximum value for a location to the default maximum altitude, or the maximum altitude selected by the user. The *PieControl* class sets the maximum value equal to 360 degrees, and each subfield as a proportion of that.

At this point, the *ChartsManager* class calls the *KMLClass*, which again handles the creation of the KML Polygons for the given chart type. Once the Polygons are created, the methods are the same as those for bar charts. However, the methods for setting the Polygons is slightly unique for each type of chart. The methods for each type are managed within the

KmlPoly class. Pie charts are created as multiple Polygons placed next to one another until a pie is formed. For stacked bar charts and 100% bar charts, the logic is slightly different. For these, the radius must be set at a minor degree smaller for each subfield above lowest. The logic for this method is

$$\text{Polygon Radius} = \text{Total Radius} - \text{Array Index}$$

This means that for a specified location, the first field added to the array will be the field which appears closest to the Earth's surface and will have an array index of 0. So, if the total radius is set to a default value of 150, then that Polygon will have a radius of 150 - 0, or 150. The next Polygon placed in that location will have an index value of 1, so its radius will be 150 - 1, or 149, and so on. This is necessary because each Polygon is attached to the surface of the Earth, and making the radius of each Polygon slightly smaller than the one below it assures that there will be no overlap in color visibility.

5.4. KML logic

In order to create a KML file, multiple Polygons must be added to a document which can then be saved as a KML file. The *KMLPoly* class handles the creation of each individual Polygon. Each *KMLPoly* object consists of a Placemark, a Polygon, and a description. The description includes the information selected by the user to be included in this description. The Polygon includes an *OuterBoundary* object, which includes a collection of coordinates which makes up the outer boundaries of the Polygon object.

The methods for creating the outer boundaries of the Polygon vary only slightly between all types of charts. From the latitude and longitude of the user-selected location, a Point object is set along the outer boundary at 0 degrees at a distance of the user-selected or default radius. From there, a Point is set every 18 degrees around the circle until it returns to the location of the


```

double[] latlon = new double[2];

for (int i = 0; i <= 20; i++)
{
    latlon = CirclePoint.GetLatLon(lat, lon, i * 18);
    outer.LinearRing.Coordinates.Add(new Vector(latlon[0], latlon[1], val));
}

return outer;

```

Figure 12: Programming Logic for Creating Points in a Circle around a Location

first Point. A second overlapping Point must be placed at this starting point (i.e. 0 degrees) in order to complete the Polygon. Figure 12 shows some of the programming logic associated with this method. In this method, a total of twenty-one points are created in a circular pattern around the location's latitude and longitude. The final point (i.e. when $i = 20$) completes the circle and overlaps with the first point ($i = 0$). The degrees from 0 for each circle is then calculated as $i * 18$, thus creating a new Point at every 18 degrees around the circle. Each new point is added to the outer boundary of the Polygon, then that boundary is returned.

The method for creating a pie chart is similar. However, instead of each Polygon starting and ending at 0 degrees on the outer edge of the circle, each boundary begins and ends at the user's latitude and longitude location. That is to say, if the user-selected location is at a latitude and longitude of (x,y) , the first and last Point in the Polygon's outer boundary will be located at (x,y) . Each subsequent Polygon must have information regarding the final degree measure of the previous Polygon for the given location. For example, if a specific Polygon begins at 0 degrees and ends at 37 degrees before returning to the location's latitude and longitude, the subsequent Polygon must begin at 37 degrees and end at a degree that matches the value of that field.

Once the outer boundaries for the Polygon have been set in this manner, the rest of the Placemark's values can be set. The altitude mode for all Placemarks is set to absolute, meaning

that each Polygon is clamped to sea level. The outer boundaries which have already been created are then associated with the Polygon object. The Placemark's geometry is set to the Polygon object which has been created. Finally, style is added to the Placemark. Style includes setting the lines of the Polygon to width of zero pixels, so that they are not visible. It also includes setting the color of the Polygon based on the user's selected color option.

The above process is applied for each location and Polygon in the user's data set. Once the application has a complete list of *KmlPoly* objects, each Placemark is added to a Document object, which encompasses all items in the KML file. This document is added to a Kml object, which can then be saved as a KML file in the location which the user has chosen.

5.3. Projecting circles on Earth's surface

In order to project circles on the Earth's surface, it is not enough to use latitude and longitude values to derive distance from a given latitude or longitude. While latitude values are

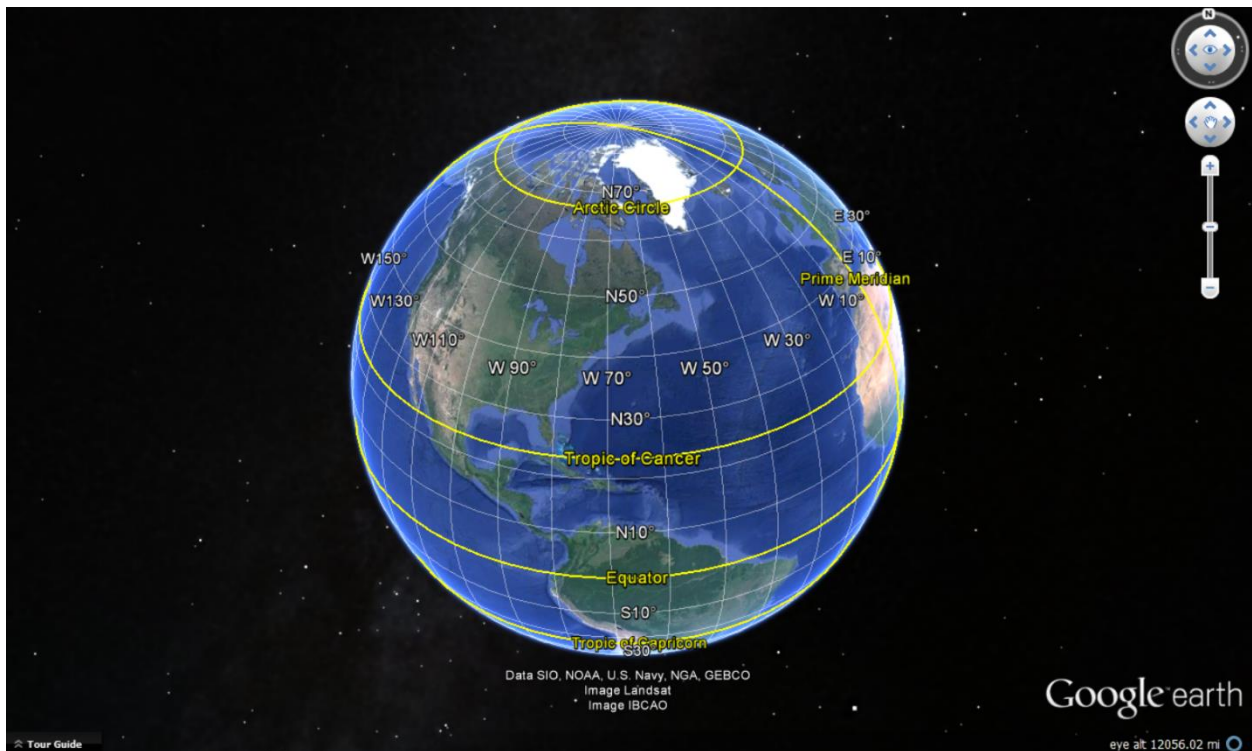


Figure 13: Example of How Latitude and Longitude Distance Can Vary with Location

equidistant from one another, longitudinal distance vary based on latitude. Figure 13 shows how geographic distance and distance in degrees of longitude can vary based on the location on the Earth's surface. Longitudinal degrees are much closer together geographically as one travels closer to the poles than they are closer to the equator. Using latitude and longitude to create circles on the Earth's surface would create warped circles which would become more and more oval shaped as one travels closer to the poles.

Because of this, a method had to be used to calculate a latitude and longitude value based on its distance and bearing from a given point. The distance from a given point will be the radius of the resulting circle. The bearing varies for each point, calculated in increments of 18 degrees around the location being mapped. An equation exists for calculating latitude and longitude based on distance and bearing. This equation can be seen in figure 14 (taken from [36]). The equation given assumes that the Earth is a sphere, when in fact it is an ellipsoid. Because of this, the calculations show some small degree of error. However, because the goal of this calculation is to approximate a circle on the Earth's surface, this does not have any negative impact on the quality of the output.

The actual implementation of this equation can be seen in figure 15, which shows the method which calculates each new distance using the equation given in figure 14. In this method, dKm is the radius of the circle, while R is the radius of the Earth, or 6378.14 kilometers. The

$$\varphi_2 = \text{asin}(\sin \varphi_1 \cdot \cos \delta + \cos \varphi_1 \cdot \sin \delta \cdot \cos \theta)$$

$$\lambda_2 = \lambda_1 + \text{atan2}(\sin \theta \cdot \sin \delta \cdot \cos \varphi_1, \cos \delta - \sin \varphi_1 \cdot \sin \varphi_2)$$

where φ is latitude, λ is longitude, θ is the bearing (clockwise from north), δ is the angular distance d/R ; d being the distance travelled, R the earth's radius

Figure 14: Finding Latitude and Longitude Using Distance and Bearing

```

public static double[] GetLatLon(double lat, double lon, double angle)
{
    int dKm = 150;
    double lat1 = toRads(lat);
    double lon1 = toRads(lon);
    double bearing = toRads(angle);

    double lat2 = Math.Asin(Math.Sin(lat1) * Math.Cos(dKm / R) +
        Math.Cos(lat1) * Math.Sin(dKm / R) * Math.Cos(bearing));
    double lon2 = lon1 + Math.Atan2(Math.Sin(bearing) * Math.Sin(dKm / R) * Math.Cos(lat1),
        Math.Cos(dKm / R) - Math.Sin(lat1) * Math.Sin(lat2));

    double[] latlon = new double[2];
    latlon[0] = toDegrees(lat2);
    latlon[1] = toDegrees(lon2);
    return latlon;
}

```

Figure 15: Method for Finding Latitude and Longitude around a Circle

latitude, longitude, and bearing, which are given as input for the equation, must be converted radians. This occurs in a separate method. The latitude for the point being determined is then calculated using the equation from figure 14. Then, the longitude is calculated for the output location. These two values then must be converted back to degrees. Again, this occurs in a separate method. The new latitude and longitude is then returned.

In the case of a basic circle, this occurs for each point every 18 degrees around the circle, which makes up the outer boundaries of this circle. This method allows for the creation of uniform circles across the surface of the Earth, which no visible distortion.

6. ANALYSIS

To investigate the efficacy of this application with regards to expanding the integration of GIS technology among non-specialized users, different properties of the application will be reviewed and compared. First, this paper will compare the use of sample data for creating charts in typical GIS software compared with this application. Second, performance will be reviewed. Finally, output for each of the charts generated will be reviewed.

6.1. Sample data generation comparison

The use of virtual globes such as Google Earth are free to download and highly accessible, provided that the user has an internet connection to download, so the goal of this section is to review the usability of the KML chart generating application in comparison with a similar task in GRASS GIS.

6.1.1. Generating a bar chart in GRASS GIS using sample data

GRASS GIS is open source and has bar and pie chart generating capabilities. The documentation provided by the company includes a description of all of the chart parameters, along with two examples of how bar charts can be created using the sample datasets provided by GRASS GIS [37]. This sample data set includes geographic data from Spearfish, South Dakota. The sample chart was created using GRASS GIS version 6.4.4.

Upon opening GRASS GIS, the user must first select the project location; in this case, this is the Spearfish data set. From this point, two windows open. This can be seen in figure 16. The window on the left is the layer manager, while the window on the right is the map display window. The example presented in the documentation provides commands which can be entered in order to generate a bar chart using the Spearfish data set. In order to enter this information, the user must select “Command console” from the layer manager. The user can then enter the

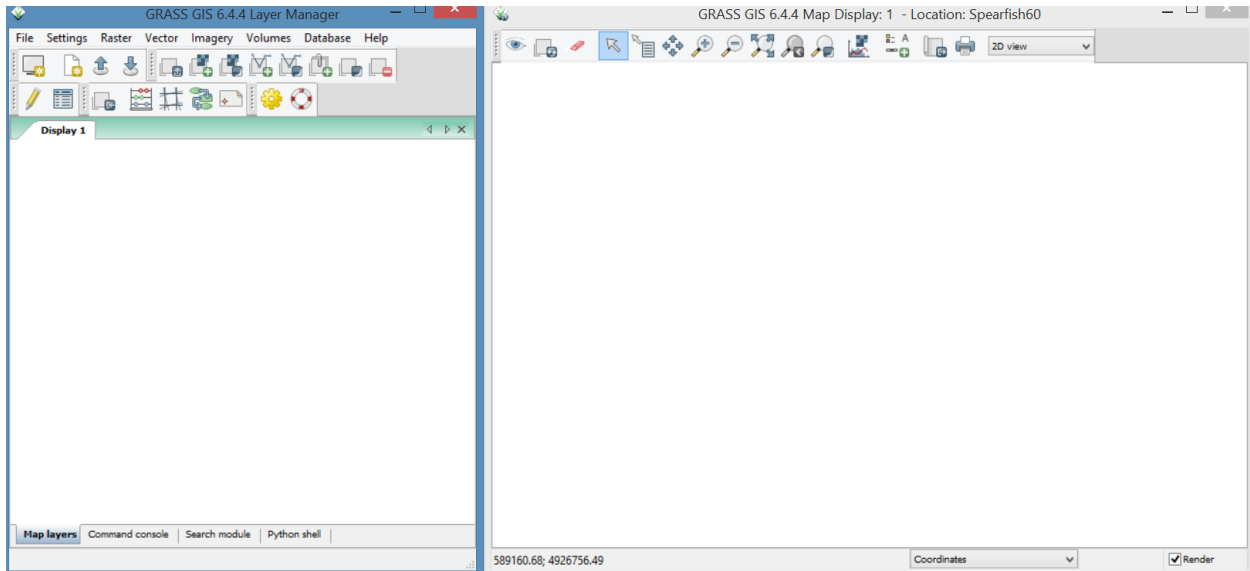


Figure 16: Starting Point for GRASS GIS version 6.4.4

information provided in the example from the documentation. The commands which should be entered can be seen in figure 17, the text and definition of parameters for which can be found in the `d.vect.chart` documentation page [37]. Inputting these parameters create a bar chart of the erodibility index of locations within the Spearfish data set. Once each line of the commands in figure 17 have been executed, a map will appear in the map display section. The map which is generated from the input of these commands using the Spearfish data set can be viewed in figure 18.

Assuming that the user has some knowledge of what a command line is, and how to input commands into the command line, the instructions for creating a bar graph are straightforward.

```
r.to.vect -s -v in=erode.index out=erode_index feature=area
v.extract in=erode_index out=erode_index_ctrds type=centroid
d.rast aspect
d.vect.chart map=erode_index_ctrds ctype=bar columns=cat \
    size=10 max_ref=12 scale=1.5 colors=yellow
d.vect erode_index_ctrds icon=basic/circle fcol=black col=black size=5
```

Figure 17: Commands for Setting and Creating a Bar Chart in GRASS GIS

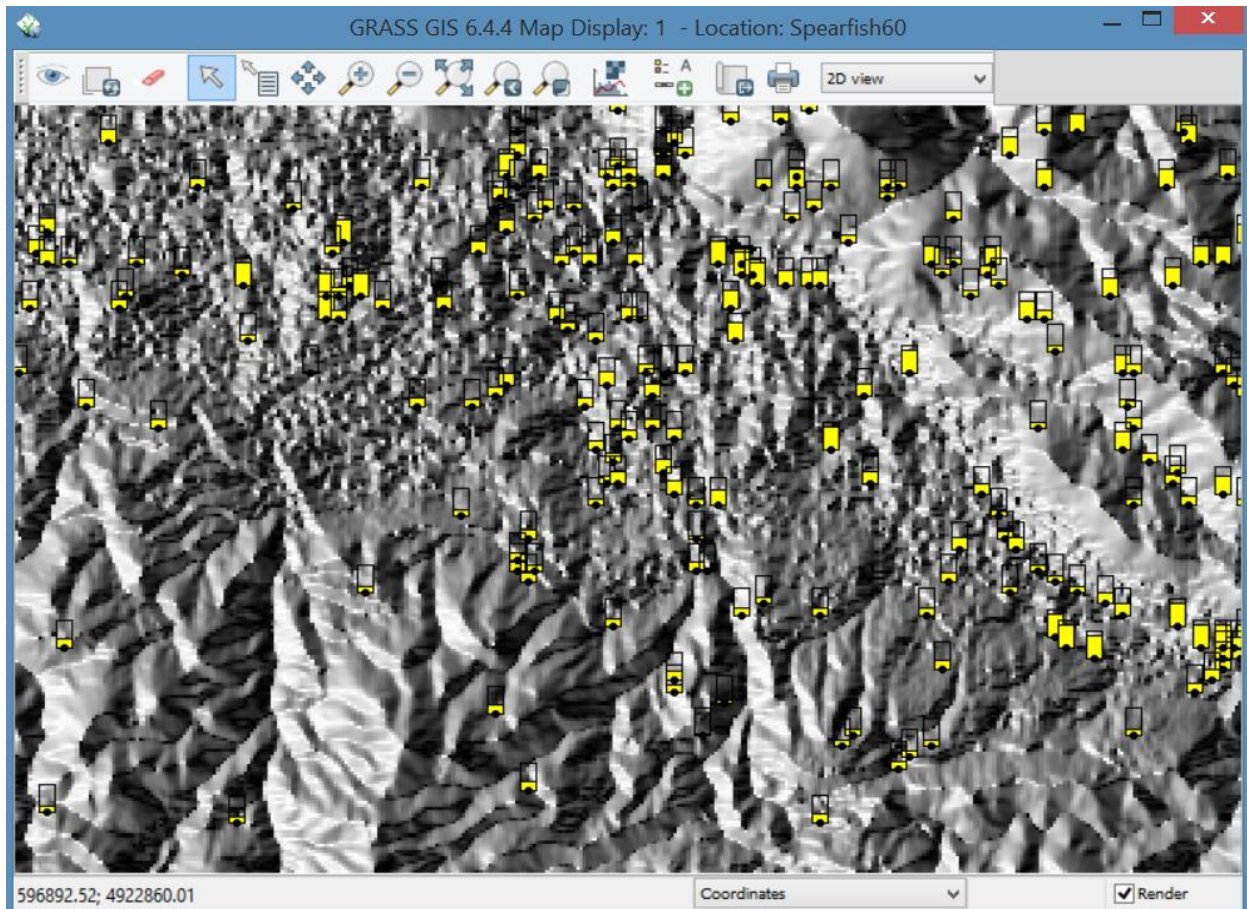


Figure 18: Bar Chart Output in GRASS GIS Using the Example Provided

However, the documentation does not explicitly lead the user through the steps required in order to create the graph. They provide commands, but assume knowledge of command lines and the nature of the input text provided, which the user must use parameter descriptions to understand, which in itself assumes knowledge of terminology. This may or may not be an issue, depending on the level of experience of the users implementing it; however, it is likely to be an issue among typical beginner users.

6.1.2. Generating a bar chart in the chart generator application

The chart generator application provides three sources of sample data for the user to choose from in order to create different charts without having to download and format their own

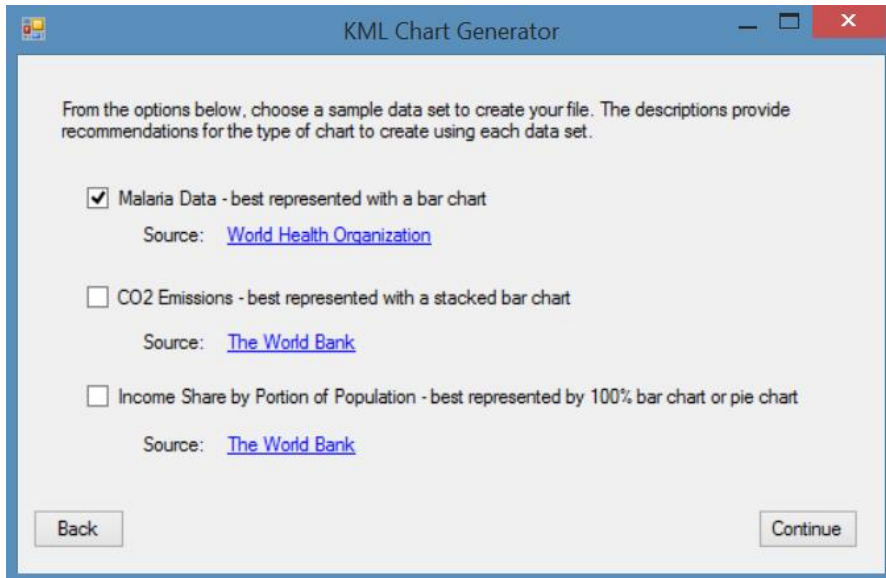


Figure 19: Choosing from Sample Data Options

data. To reflect the process in the traditional GIS application, a simple bar chart will be created using the sample Malaria data set [38] in order to show how the processes differ.

Upon executing the program, a user is given the option to use a sample data file, or to choose from their own file. Once the user selects to use a sample file, the form shown in figure 19 is displayed. This form gives the users sample data options, along with recommendations for the type of chart to create with each option, as well as a link to the source from which the data was taken. As can be seen in figure 19, the form guides users through the process, giving instructions which help them understand what they need to do in order to achieve their goal, allowing them to answer affirmatively to the first question in the streamlined cognitive walkthrough [12] (i.e. “Will the user know what to do at this step?”). If the user does not select an option and attempts to continue, a dialog box opens prompting the user to select a file, or choose the “Back” option to select the option to choose their own file. If the user receives no prompts when selecting “Continue”, they will know that the current step has executed successfully and they can continue on.

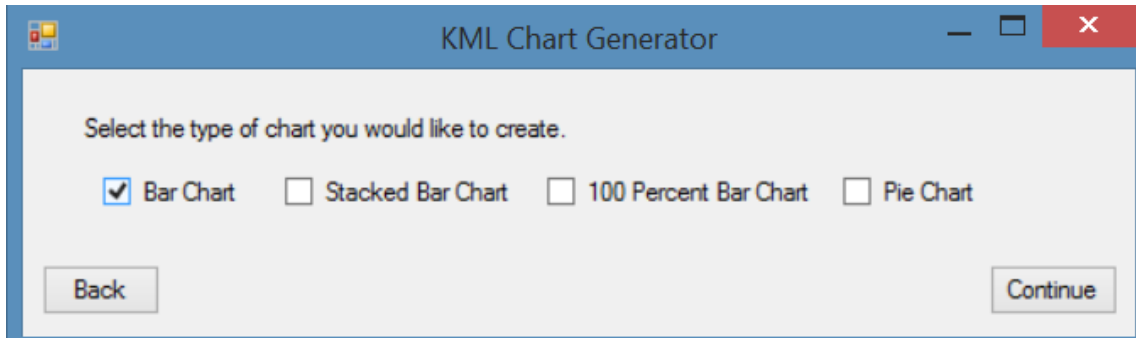


Figure 20: Selecting a Chart Type

Once the user has selected the sample data file to use, they are allowed to view/review the data before creating the chart. This step requires no editing in the case of sample data files, as all necessary information is included. Once the user has reviewed the data, they select the type of chart that they would like to create. This should be the type recommended to them in the options menu. The form with the possible selections can be seen in figure 20. Figure 21 shows the following screen, which allows the user to select the title for their chart, along with the field to include in the chart. In the case of the bar chart, there is only a single field to include. However, user data might include several fields in the same Excel worksheet, so at this point they would need to choose from these options.

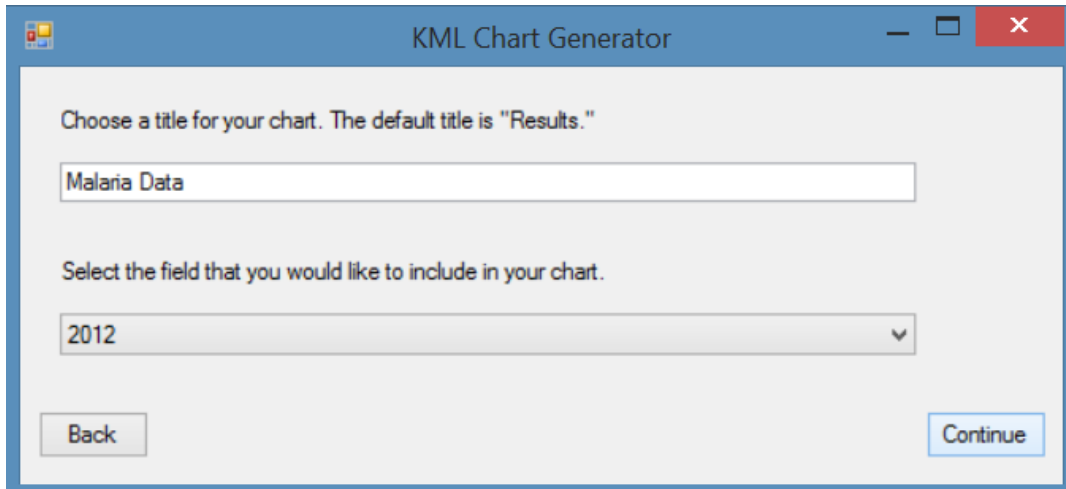


Figure 21: Selecting a Title and Category

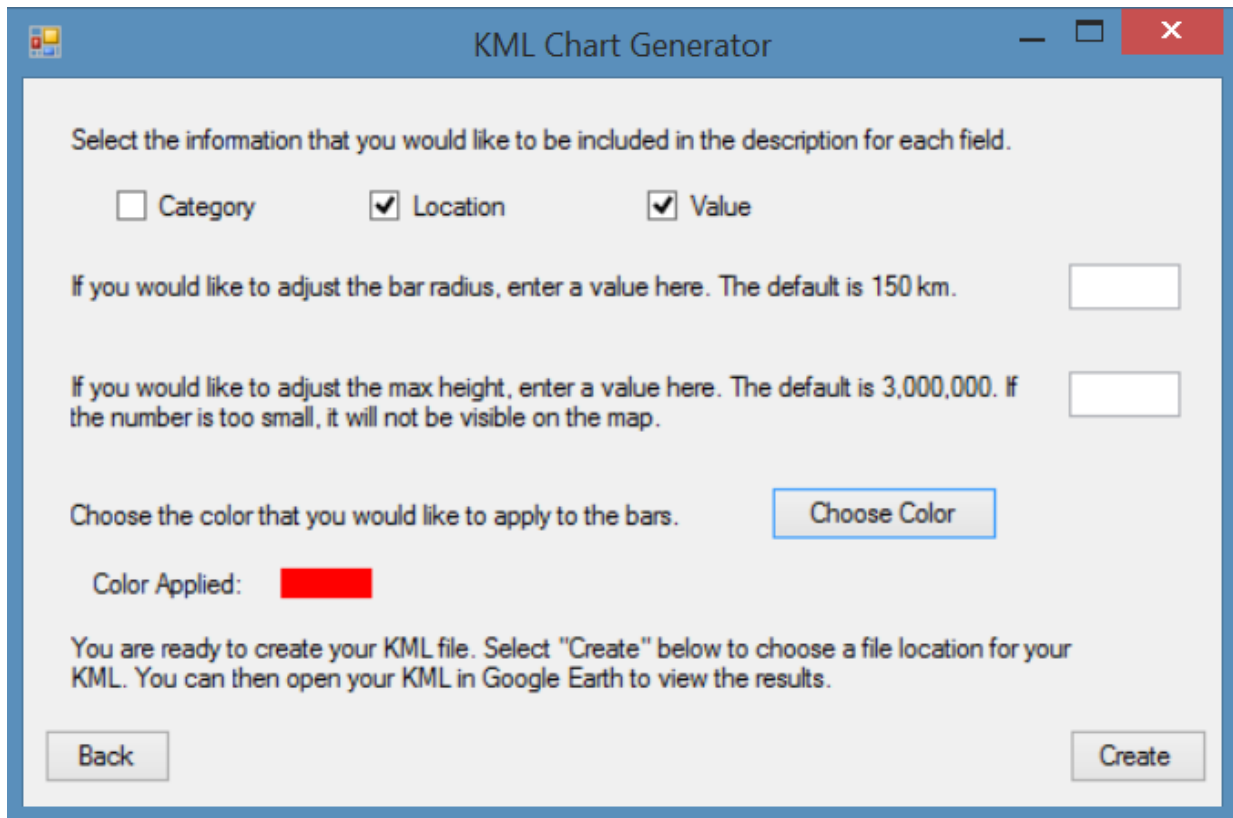


Figure 22: Selecting User Options

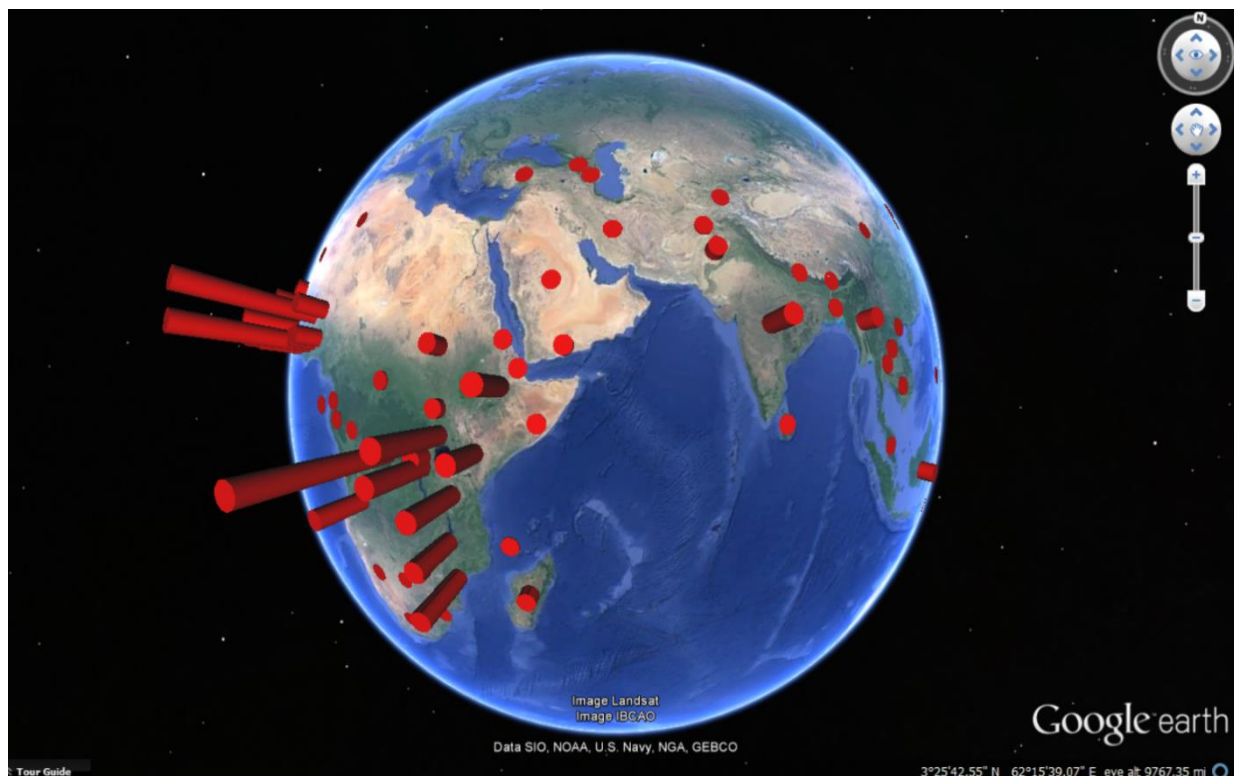


Figure 23: Output from Malaria Sample Data Set Viewed in Google Earth

The final form before generating a KML file allows users to set options such as color, as well as fields included in the description field of each KML Placemark. This form can be seen in figure 22. Once the KML has been generated, the user can go into Google Earth and select File, then Open to open the KML file that they have created. The output for the data can be seen in figure 23. Within Google Earth, the user can then adjust the virtual globe to see how malaria affects different populations throughout the world.

The goal of this step by step process is to allow the user the freedom to create a chart creatively while leading them through the process. The use of sample data files allows the user to skip the step of formatting their own Excel file, while still making use of the application. This

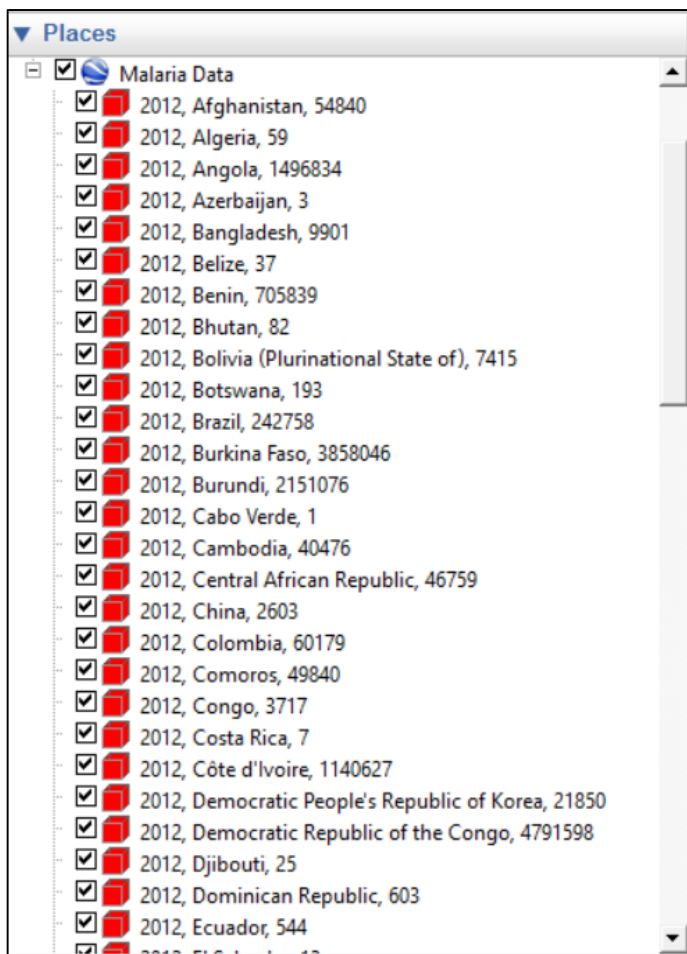


Figure 24: Google Earth Navigation Pane Showing User-Selected Display Values

process requires no knowledge of command lines, as well as limiting extraneous information and options, allowing the user to focus on a single task. After going step-by-step through the chart generator application, the steps required within Google Earth are limited. The goal of this is to allow the user to view their data with as little necessary prior knowledge of Google Earth as possible, thereby increasing the usability of the program, while decreasing the time spent learning to use it.

In addition to viewing data trends in the main screen of the Google Earth user interface, the navigation pane on the left-hand side allows the user to navigate to each data point separately. Double clicking any of the items from the list in figure 24 allows the user to navigate directly to that location. One use of this might be to see how that country compares to those surrounding it. The information which appears in this pane is chosen by the user. For example, in this case including the year in each of the descriptions is not necessary, and could be excluded by the user and instead added to the title of the chart data.

6.2. Creating a bar chart from external data in GRASS GIS

Some data sources provide Shapefiles which can be loaded into traditional GIS and converted into Bar Charts. The United Nations Environment Programme (UNEP) data repository [32] is one of these sources. Using the data provided, it is possible to create bar charts in software such as GRASS GIS.

6.2.1. Accessing Shapefiles

In the case of the UNEP data repository, the files downloaded are compressed in a .tar file. A .tar file is a compressed file format for UNIX type operating systems. In order to open this file in Windows, a third party tool is necessary. Upon downloading the file, the user can open the file using this third party tool and copy the files within the folder to the desired location.



Figure 25: Initial GRASS GIS Form

6.2.2. Importing Shapefiles into GRASS GIS

Upon opening GRASS GIS, the user is presented with the form seen in Figure 25. If it is the user's first time using the application, they will have the North Carolina and Spearfish data set available to them, depending on the data sets they choose to include in the application download. If they have opted not to include sample data sets, the window may be empty of any existing Locations or mapsets. The user can then create a new Location, or choose to import the Shapefiles into an existing Location. The user should then select the user mapset, and then select the "Start GRASS" option. In order to understand the relationship between the PERMANENT and the user mapsets, the user needs to refer to the documentation or other sources of information.

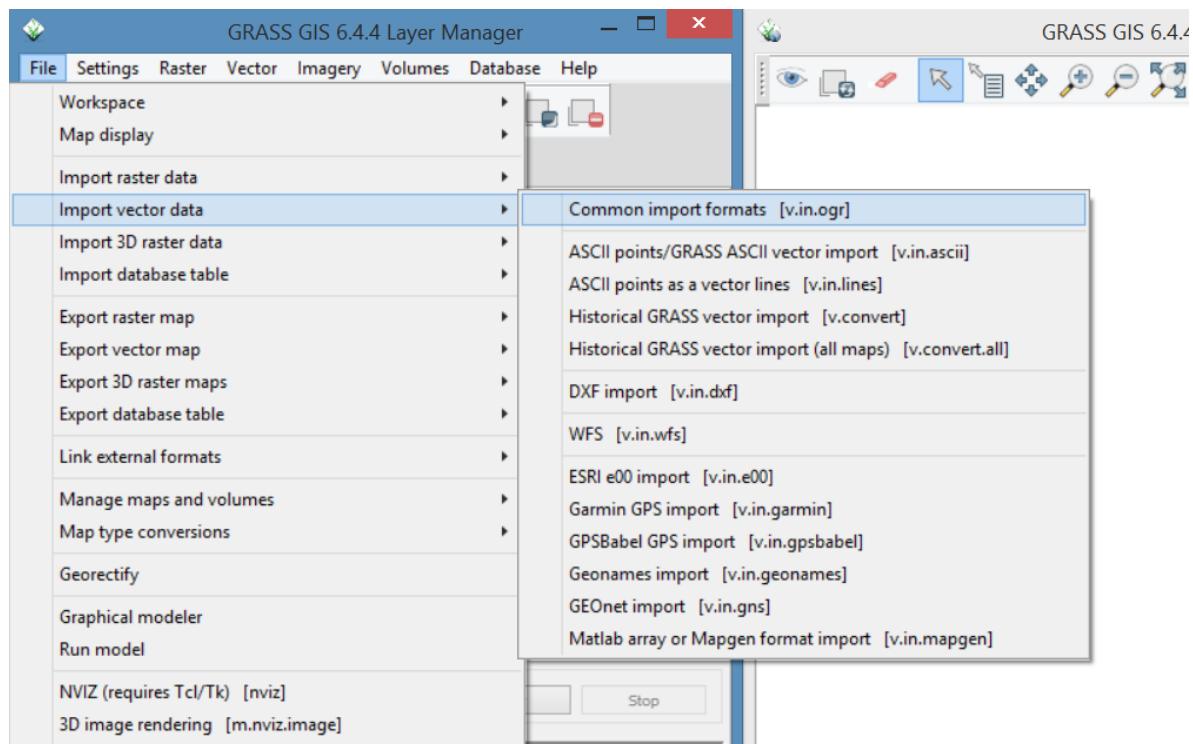


Figure 26: Selecting the Correct Import Option in GRASS

Once the user has started GRASS, they need to import the Shapefile into the mapset. There are two ways to do this, either through the command console, or through the menu options. These options can be seen in Figure 26. Some searching through documentation and other online guides will let the user know that they want to use the v.in.ogr function. The documentation provides all of the flags and parameters that one can use to import the data. There is a simple option for importing the file in the v.in.ogr form which opens when the option from figure 26 is selected. This allows the user to select the Shapefile that they would like to use, then select “Import” from the menu.

However, there are three Shapefiles provided with the .tar file downloaded from the UNEP data repository. Some trial and error shows that only one must be imported. Upon executing the simple import command, the command console output window shows the

following warning “WARNING: Errors were encountered during the import. Try to import again, snapping with at least 1e-013: ‘snap=1e-013’”. In order to understand this error, the user can refer to the documentation.

The documentation shows that the “snap” parameter sets the snapping threshold for boundaries and is a float. The description section of the documentation shows that the snap parameter snaps vertices together if they are within the snap threshold distance from one another. In order to see what this means within the application, the user can use some trial and error. But first, the map should be displayed.

6.2.3. Viewing the Shapefile in GRASS

Despite being imported with errors, the imported map can be viewed in the map display window. The user can select “File”, then “Map Display”, then “Add vector” in order to view the map. They can then select the map name from the drop down list provided. The map will then display in the window. The imported map can be seen in figure 27. The map shows that there are

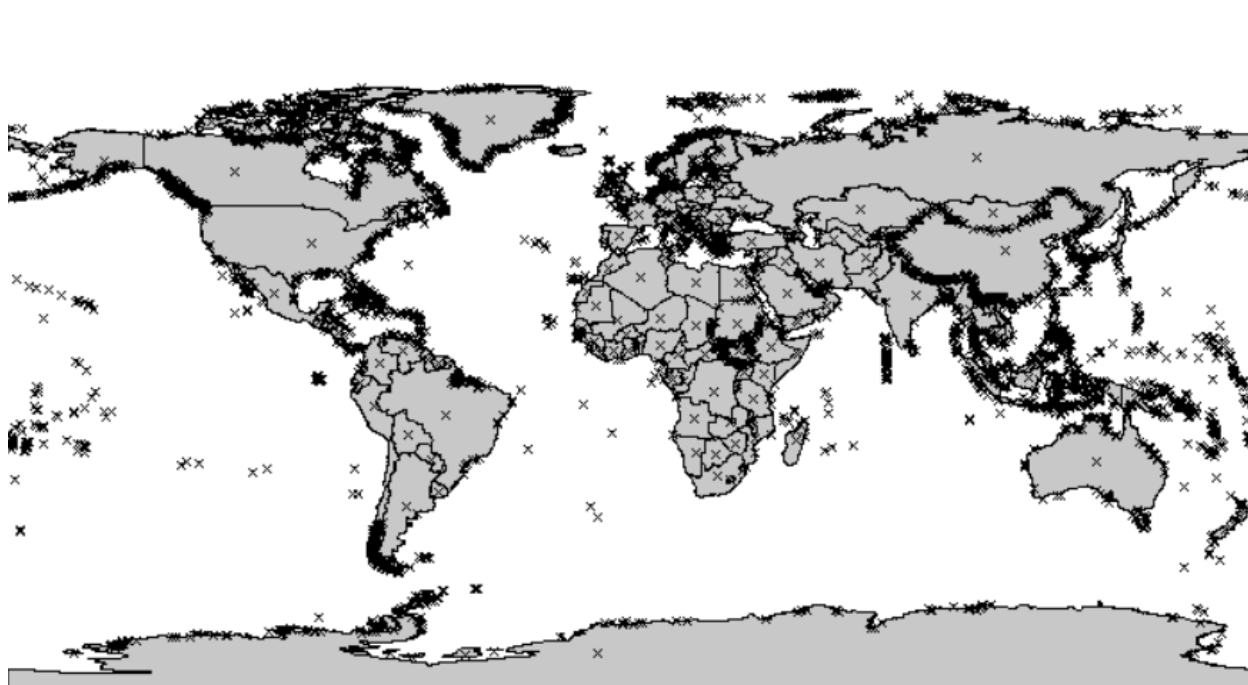


Figure 27: Result of Initial Shapefile Import

a large number of superfluous centroids located mostly along the boundaries of the map. Some searching shows that you can adjust the shape of these centroids so that they are less obvious, but they act as centroids and therefore cause major issues when trying to create bar charts at area centroids. Because the purpose of the Shapefile import is not purely aesthetic, it is necessary to remove any extraneous centroids from the map.

The first step is to adjust the snap parameter and see how it affects the errors on the map. In order to access this, the user must choose to import the map again, using v.in.ogr. Once the user has selected this, they must select the “Command dialog” option from the import window. This gives the user extra options for setting flags and parameters. Trial and error will show that running a new import on the same file does not allow for automatic overwrite. There is a parameter which can be set by the user in one of the command dialog tabs. Another option is to create a new name for the output vector map. To see how changing the snap value affects the output, the user can try different values.

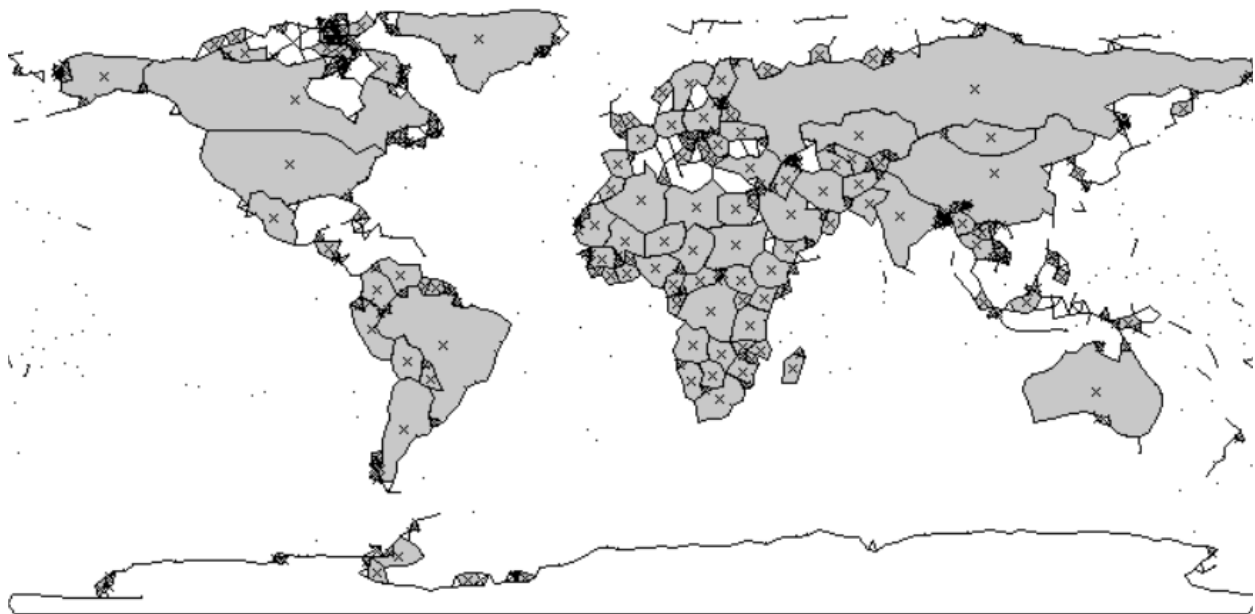


Figure 28: Example of a High Snap Parameter Value in GRASS

Some trial and error will show that a higher snap value will remove many of the extra centroids, but will distort the map. The higher the value, the fewer the extra centroids, but the more the map is distorted. An extreme example of this can be seen in figure 28, where the snap parameter is set to a value of 3. While the map is distorted beyond what the user would want, many of the extraneous centroids are still there. Thus, just adjusting the snap parameter is not a solution.

6.2.4. Cleaning the data

At this point, some research is necessary for other options for removing the extra centroids without distorting the map too much. Searching the documentation will show that there is a `v.clean` function. Again in this case, the user may choose to find the window which allows them to input the data, or they can run the commands on the command console. The benefit of

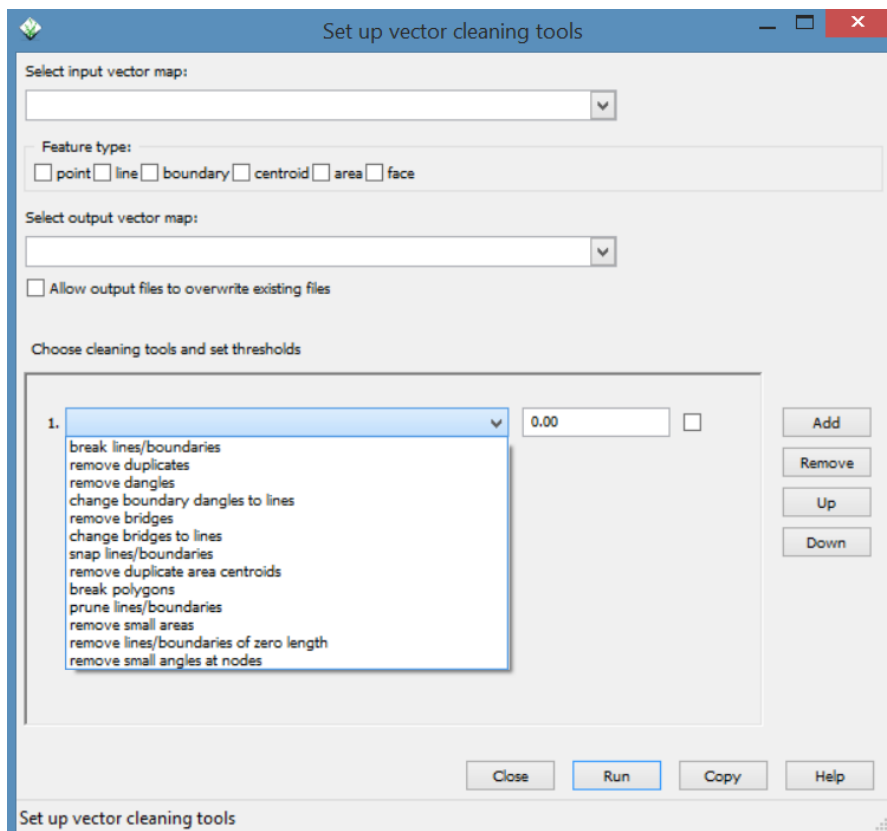


Figure 29: Vector Cleaning Window in GRASS

using the command console is that the documentation uses this format, and provides examples of how things can be done. The window can be seen in figure 29, and shows the tools available for cleaning.

A first step might be to try removing duplicate area centroids. The documentation shows that it is not necessary to set a threshold for this tool. Upon executing the “Run” command, the user then has to go back to the map display option, and choose to add the map layer to be viewed. Viewing this shows no visible difference in the number of centroids shown on the map. Without knowing what is causing the extraneous centroids to appear on the map, some trial and error is necessary.

Eventually the user will find that selecting the clean option which removes small areas from the map, and setting the threshold value to 10 will remove almost all of the unnecessary centroids. However, as can be seen in figure 30, many centroids still remain along the border of

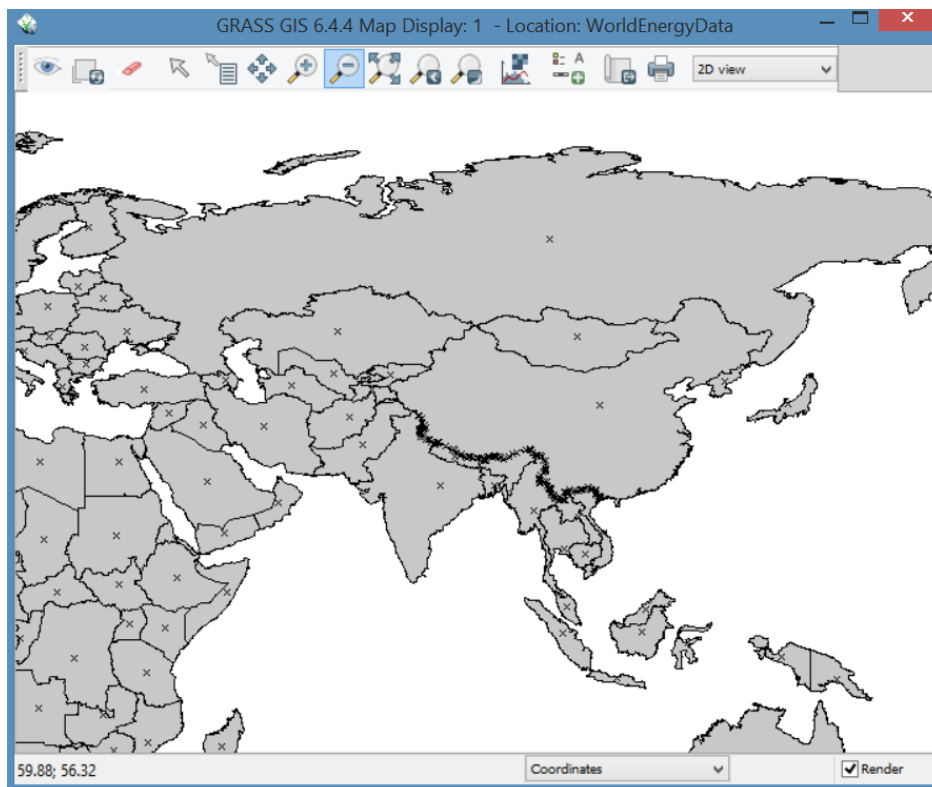


Figure 30: Remaining Extraneous Centroids after Running Rmarea Tool in v.clean Command

China, so additional cleaning is necessary. More trial and error will show that at this point, the user can run the remove duplicate centroid tool again to remove the remaining centroids, resulting in a clean map.

One important note here is that while the error message upon import suggested that the user set the snap value in order to remove the error, in practice this was not necessary in order to produce a map free of unnecessary centroids. Despite avoiding using this, removing small areas from the map still caused some distortion in areas where there are small countries. For example, much of Central America was removed in this process. Additionally, small islands were removed from the map. However, coastlines remained largely intact, as opposed to the distortion caused by using the snapping tool.

6.2.5. Displaying the bar chart

In order to display the chart, the `d.vect.chart` command can be used. Again, the user can reference the documentation in order to know which parameters to set in order to display their chart. The parameters that need to be set are as follows. The `map` parameter must be set, which gives the name of the input map. The `ctype` parameter must be set, as the default `ctype` (chart type) is a pie chart. The `size` should be set, which determines the width of the bar, as the default is 40, and while the user can decide which value works best for their goals, but for the purposes of this map, the value should probably be somewhere between 5 and 10. The user must select the column from the database which contains the information that they would like to map, and set that as the `columns` value. The final value which must be set is the scale. The default value for the scale is 1, and in order to be able to view the data, it should be set somewhere around .01 to .05. If this value is not changed, the bars will be extremely tall and will be difficult to view within the map.

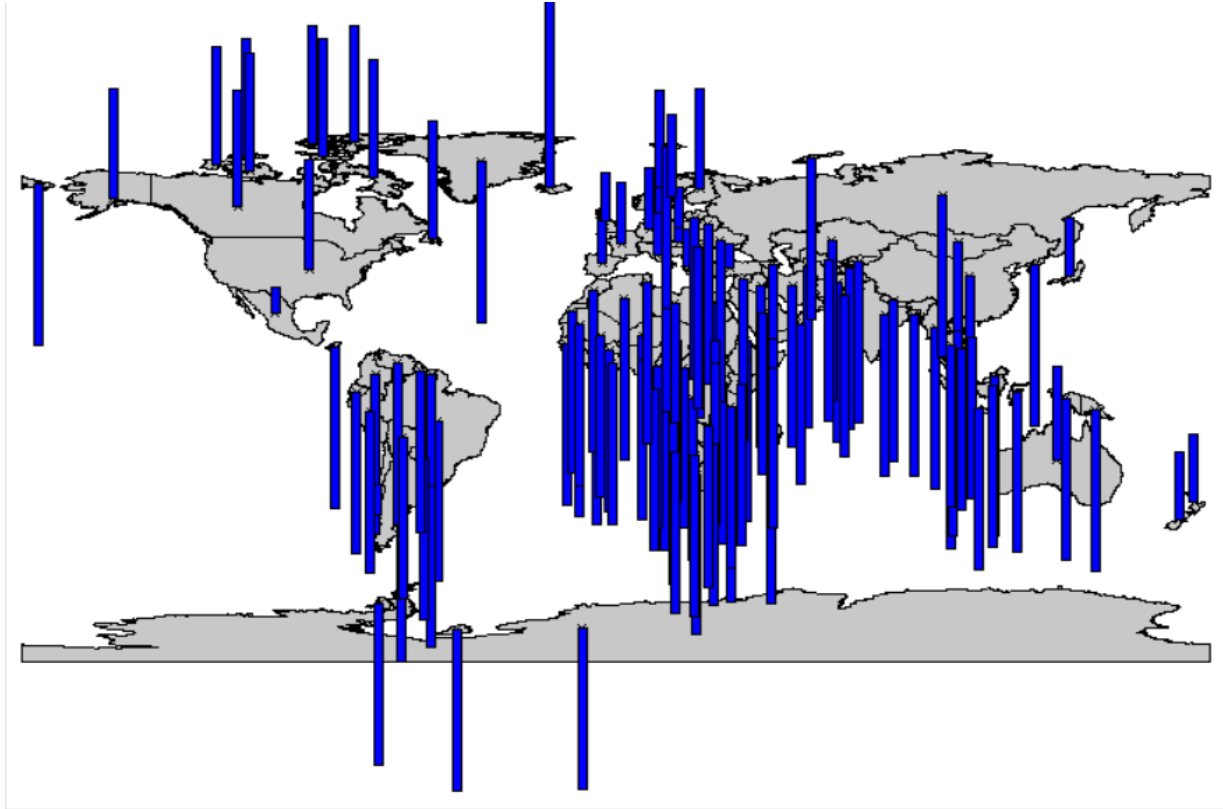


Figure 31: Initial Chart Creation in GRASS GIS

The `d.vect.chart` command must be run through the command console. One thing to note is that, when using the command console, sometimes the map values will not show up as options by default after they have been recently added to the mapset. The user can still manually enter the values, but closing and reopening GRASS GIS will cause them to appear. After setting all these values and running the command through the command console, the user will see the map seen in figure 31.

It is apparent from the output that this is not the output that the user would like. In order to understand what's going on, the user must view the data being used. Because Shapefiles are not human readable, and the database files associated with them can be opened in some programs, but are not formatted in a way that will be meaningful to the user, the user must run

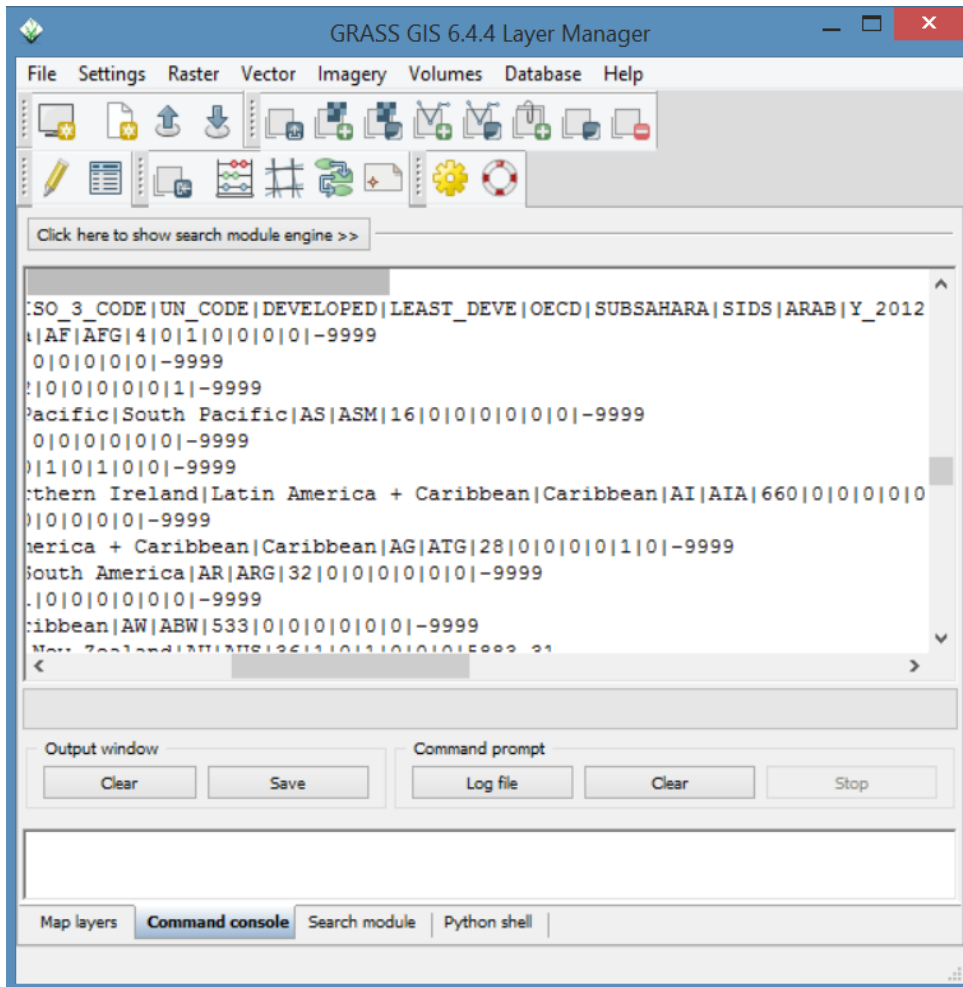


Figure 32: SELECT Query Output

SQL commands on the database. In order to view the data, the user can run a SELECT command using the db.select command in GRASS. Using the documentation, the user can find how this is done. The results of the query can be found in the command output window. A small portion of this output can be seen in figure 32. The final column in the table, named “Y_2012” is the data that is being charted. This column shows the energy output of each country in 2012. However, as can be seen in figure 32, the data has set countries where there is no data to a value of -9999 rather than 0 or NULL. This shows why the output in figure 31 looks the way that it does; many of the countries have negative values associated with them, which show up on the map.

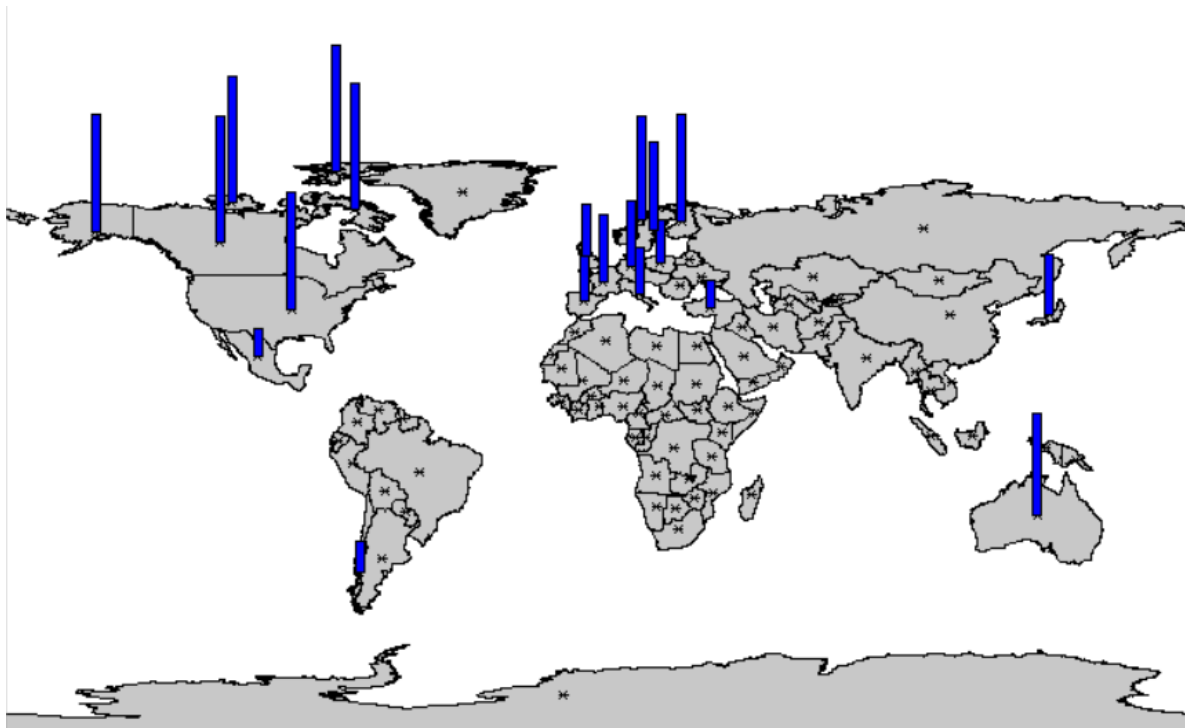


Figure 33: Bar Chart Output for Energy Use in GRASS GIS

In order to remove these values, another SQL command can be used to change the values in the database. Using the documentation, the user can see how to perform a SQL Update command. The user can use this to set the values in that column to 0 if they are -9999. After doing this, the user can run the Select command to see whether or not the columns have been changed successfully. Finally, the user can run the `d.vect.chart` command again to view the bar chart. The output from this final chart can be seen in figure 33. The negative bars have been removed, and the bar chart shows the positive values in bars throughout the map.

6.3. Creating a bar chart using external data in the KML generating application

The UNEP data repository provides the user with multiple download options. In addition to Shapefiles, the user can download an Excel file of the same data. Because of this, a direct comparison can be made of the process involved in generating charts in the KML generating application and GRASS GIS.

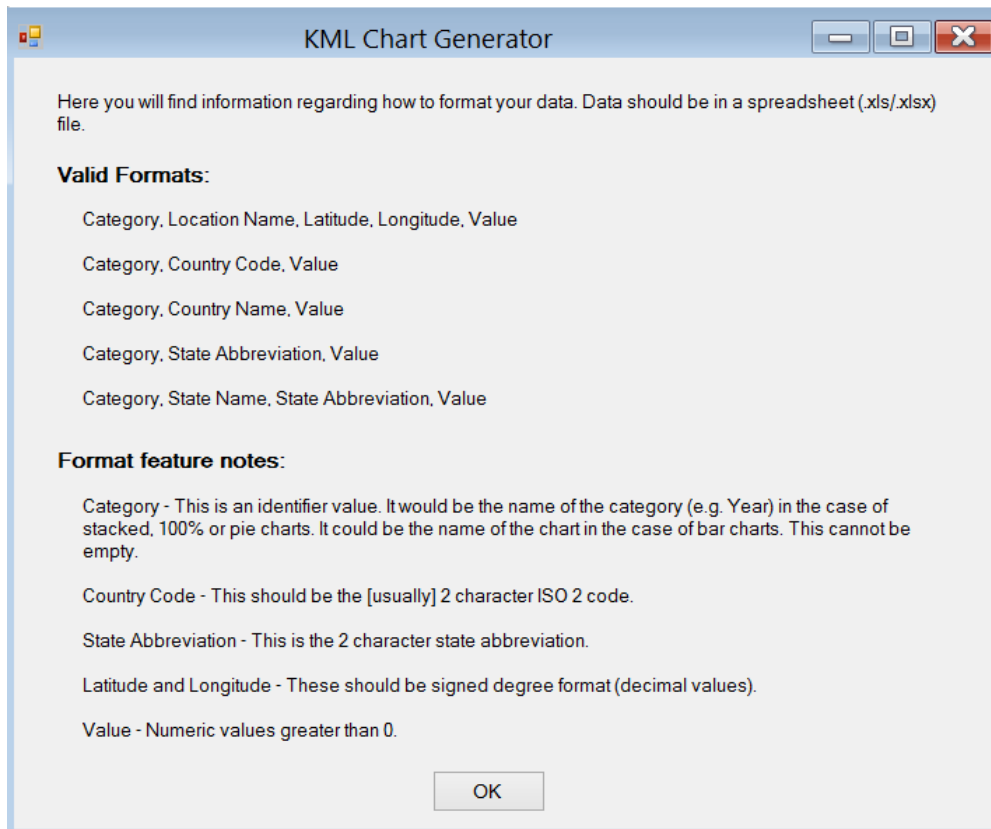


Figure 34: Form Showing Valid Formats and Term Definitions

6.3.1. Data preprocessing

After downloading the file, the first step in using the application is to make sure that the data is formatted in a way that allows the application to read it. The user can view these formats within the application. The form which shows the users the valid formats and definitions for terms used can be seen in figure 34.

The downloaded spreadsheet includes both country names and ISO2 codes. Because there is no variation between country codes like there may be for country names (e.g. US, United States, USA, etc.), using ISO2 codes queries tend to return more accurate results. Because the user may not have this information, this section will look at both variations. In both cases, the user can remove extra columns to match one of the valid formats.

17	2012	Bahrain	-9999
18	2012	Bangladesh	-9999
19	2012	Barbados	-9999
20	2012	Belarus	-9999
21	2012	Belgium	5147.84
22	2012	Belize	-9999
23	2012	Benin	-9999
24	2012	Bermuda	-9999
25	2012	Bhutan	-9999
26	2012	Bolivia	-9999
27	2012	Bosnia and Herzegovina	-9999
28	2012	Botswana	-9999
29	2012	Brazil	-9999
30	2012	British Virgin Islands	-9999

Figure 35: Adjusted Data with Two Worksheets for Each Input Format

A portion of a single file which contains both the country name and the country code versions of the information can be seen in figure 35. Both of these formats include the category, country information, and the chart value. One of the limitations of this application is the file must be formatted in a specific way, meaning that the user must either remove extraneous information or copy relevant information to a new file. However, as can be seen in figure 35, it is evident immediately that many of the values are set to -9999. The user can remove the rows containing this information, or use Excel commands to edit the rows; however, neither of these is necessary, because the application will ignore any values of zero or less.

Figure 36: Selecting a Worksheet from the Chosen File

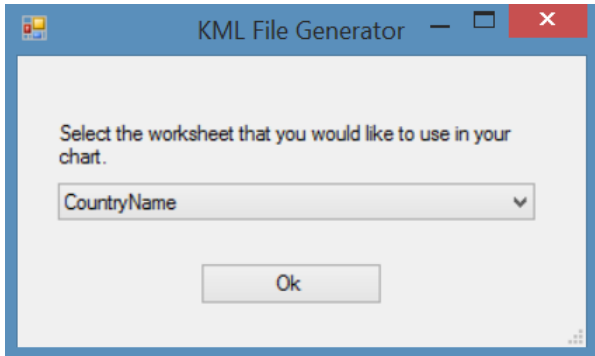


Figure 37: Selecting the Format of the User File

6.3.2. Uploading and cleaning data

After editing the data file, the user can select the file in the application. Because the spreadsheet contains multiple worksheets, the form seen in figure 36 will appear, letting the user select the worksheet that they would like to use. After selecting the worksheet, the user can select the format that they set their data in, from the list of valid formats. This can be seen in figure 37. As long as the user information is in the correct format, and this format matches the selection of the user, the application will upload the information. If the upload is successful, the status will be as shown in figure 38, and the user will be allowed to continue. If unsuccessful, the user will be prompted to review the format of their data and try again.

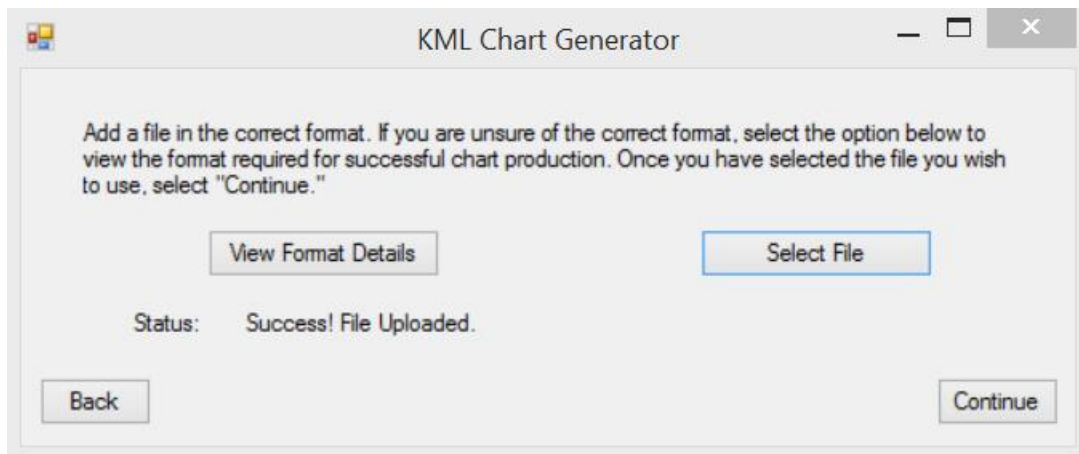


Figure 38: Example of a Successful File Upload

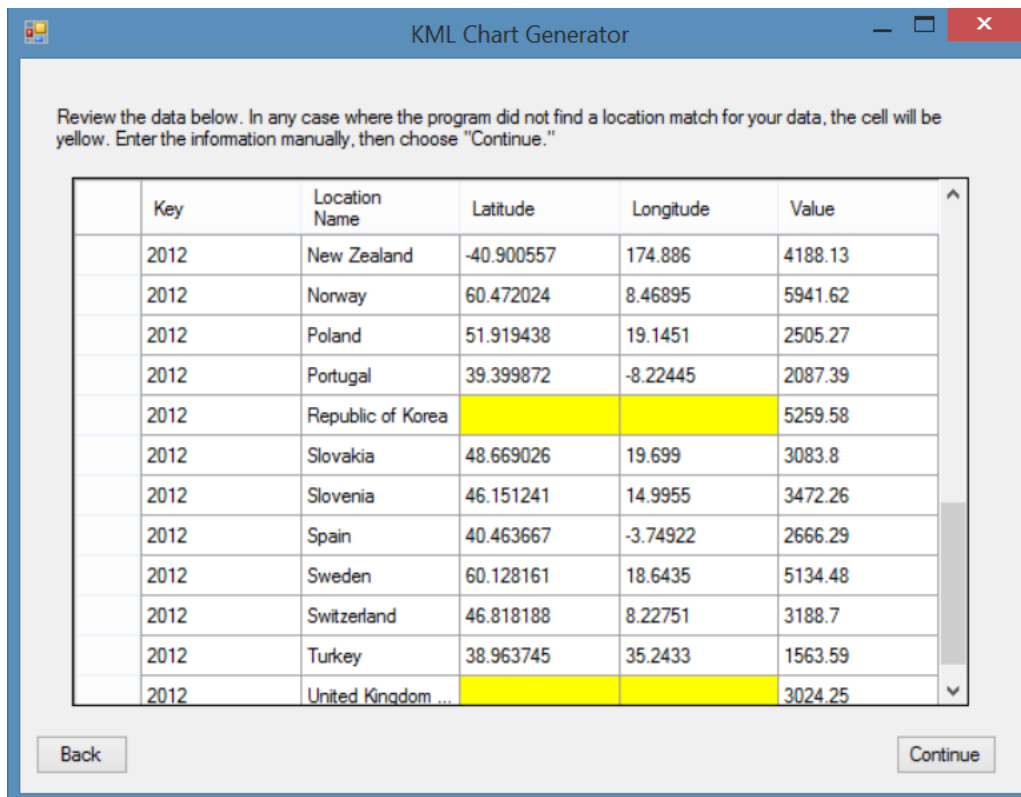


Figure 39: Uploaded Data Review

Upon continuing, the user is allowed to review the uploaded data. In the example shown in figure 39, querying by country name did not return results for all names. The case shown is a good example of why this might happen. Because there are two countries with Korea in their names, and the names used may vary, the query is more likely to return zero or multiple results. In both of these cases, the user will be prompted to input the information, to assure greater accuracy. A side benefit of this form is that once the user query returns the latitude and longitude and the user has inputted any missing information, they can then copy the latitude and longitude columns into their data, so they can run the manual option if they would like to run the program multiple times with the same file. Another benefit of this form is that it allows the user to see the output of the query without having to have any knowledge of databases or SQL queries themselves. This simplifies the interaction for the user.

This form also gives the user a chance to view and review the data which will be included in the KML file. The negative values have already been excluded from the file, so the user will know that this has occurred prior to generation of the KML. Once the user is confident that their data is correct, they can continue.

The next form simply allows the user to select the type of chart that they will be creating. After this, the user selects the title and category for their chart. Since there is only a single category in the data file, the user will only have one option. In cases where the user has multiple data categories in the same worksheet, they would have to select between them. If the user does not select a title, it will have a default value. The final form allows the user to select the display options, such as color and description values. An example of the last three forms can be seen in figures 20-22.

6.3.4. Opening KML in Google Earth

In Google Earth, the user can select File, then Open. A typical open file dialog box allows the user to select the KML file which they have just generated and view it in the Google Earth window. The energy output data can be seen in figure 40. One limitation to using the country codes instead of country names is that the descriptions contain less readable codes rather than names. The user can still zoom into the country to see the name, but it does create a slightly less readable option.

One thing to note is that in comparing the output here with the output in GRASS GIS, some discrepancies can be seen. Most notably, in GRASS GIS, Iceland was removed from the map during the cleaning process. Because it, along with other small countries, does not exist on the map, the output is not shown for the chart generation. In the case of Energy Output, the user can see that Iceland has a surprisingly large energy output. When considering asking questions

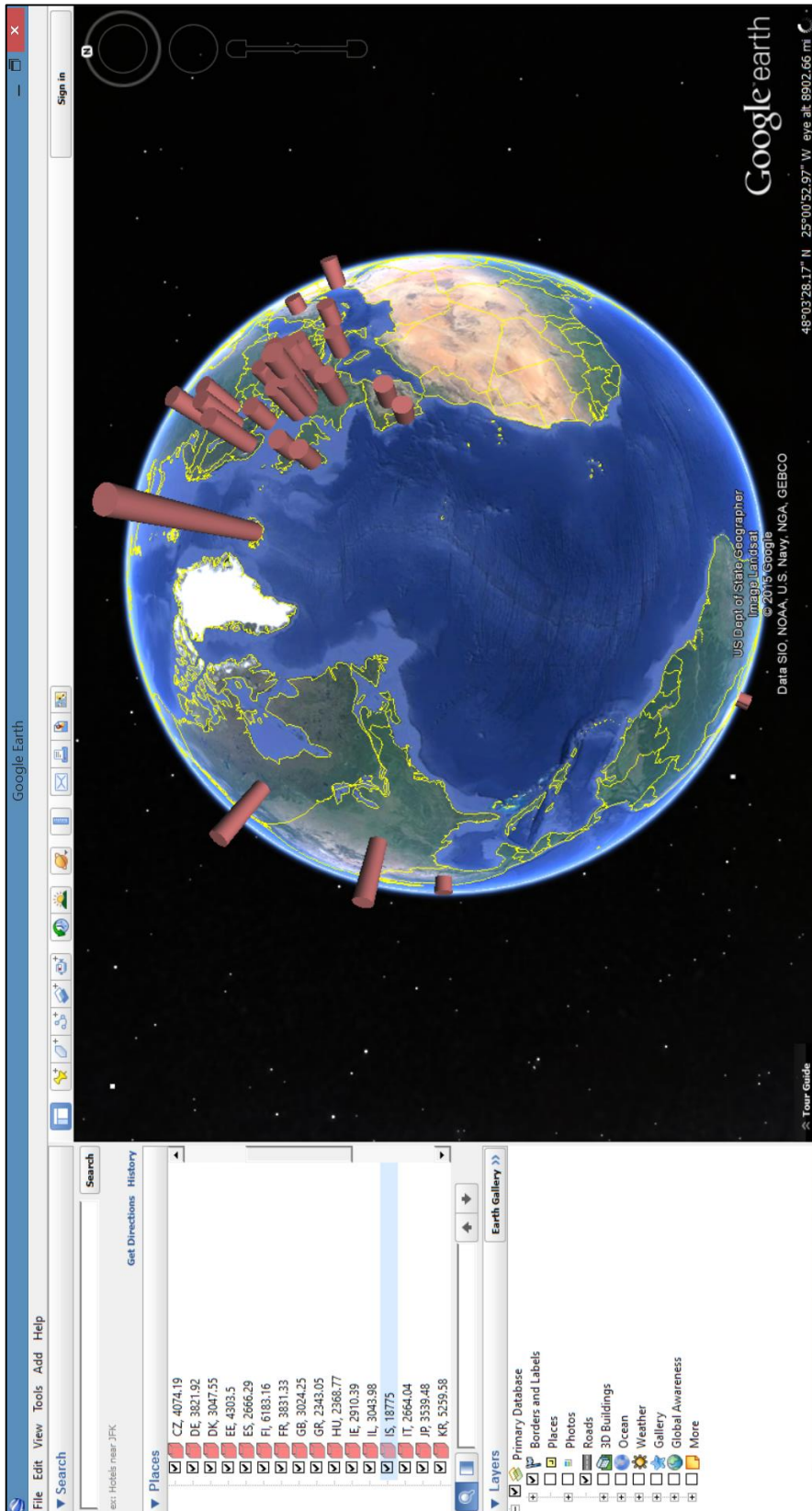


Figure 40: Google Earth Output of Energy Usage

of the data and promoting global awareness, this is significant. It is likely to prompt the user to question and subsequently discover why Iceland's energy usage is so large in comparison to other countries.

Another difference between the KML chart generator and GRASS GIS is that in cases where a country may have two separate, but large land masses, the `d.vect.chart` command will produce a bar in each of these areas. This can be seen in the case of Alaska being separate from the rest of the United States, as well as multiple examples in the north of Canada.

6.4. Creating a pie chart using external data in GRASS GIS

The process for creating a pie chart in GRASS is very similar to the process of creating a bar chart. The only major difference is that data preprocessing must be done if multiple columns are not contained in a single Shapefile. In the case of the UNEP data repository, each data set is contained in its own file.

6.4.1. Data preprocessing

The `.dbf` file associated with the `.shp` file contains all of the database information. This means that the `.dbf` file contains the information that the user would need to chart. Some applications, such as OpenOffice Calc, allow for `.dbf` files to be open and viewed in a spreadsheet application, then saved again in the same format. In order to integrate the information from multiple data sets so that it can be viewed as a pie chart in GRASS, the user must open one of these files, then copy the relevant columns from the other files into that file, renaming them if necessary. Once this has been completed, the user can open the Shapefile associated with that information in GRASS.

From this point, the process is largely the same. It differs only once the map has been cleaned and extra centroids have been removed. Again, this information contains fields with a

value of -9999 if there is no information for the given country. So the user must execute the `v.dv.update` command in order to change any negative values to zero values. The difference in this case is that the user must do this for all relevant columns.

6.4.2. Displaying the pie chart

The `d.vect.chart` command is again used for displaying the pie chart. In this case, the user does not need to specify the `ctype`, as pie chart is the default. The user will also need to specify multiple columns in the `columns` parameter. The user may want to change the size, which in this case is the diameter of the chart, and is set to a default value of 40. The data used in this example is a comparison of the types of biofuels used by different countries. The three categories are: biodiesel, biogasoline, and other. The command which a user might use to generate a pie chart is: `d.vect.chart map=EC3 columns=BIODIESEL,BIOGASOLIN,OTHER size=20`. The output for this command can be seen in figure 41.

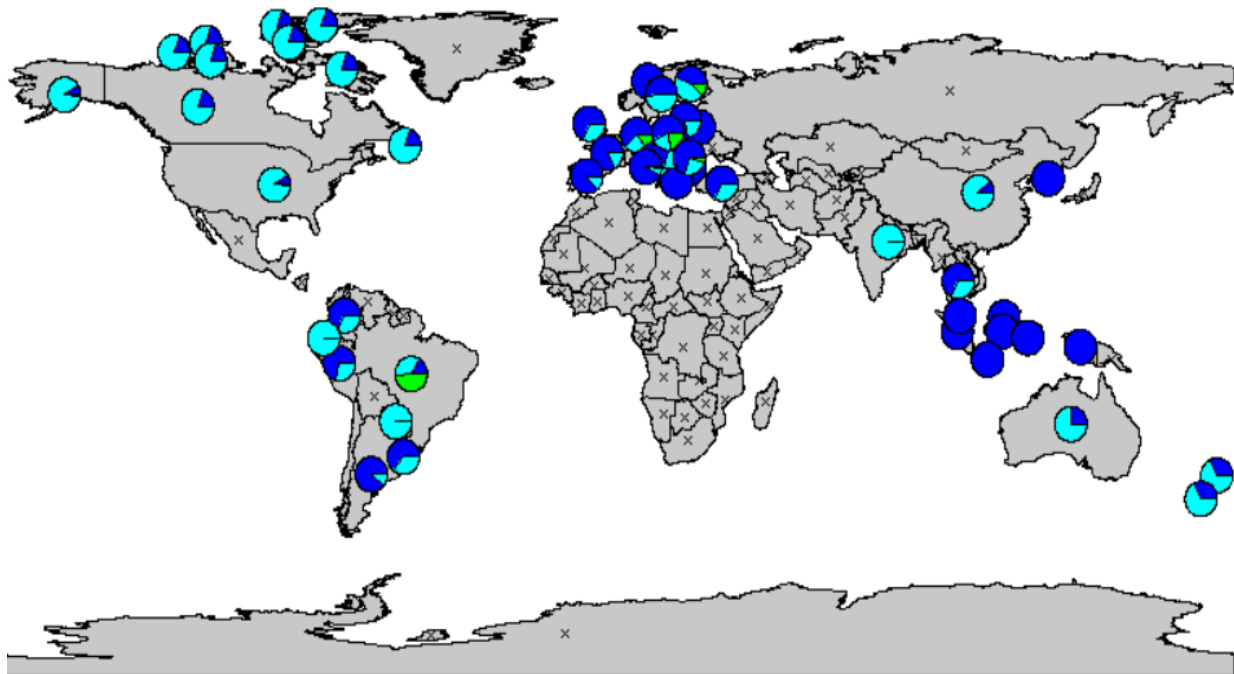


Figure 41: Output for Pie Chart in GRASS GIS

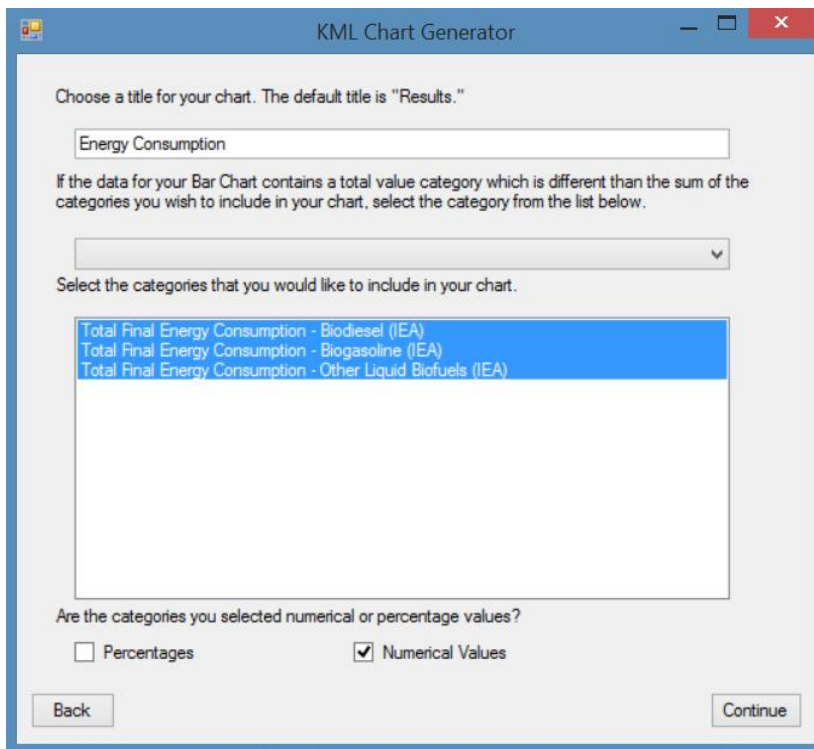
6.5. Creating a pie chart using external data in KML generating application

6.5.1. Data preprocessing

Again in this case, the process of creating a pie chart is very similar to that of creating a bar chart. The preprocessing that must be done is slightly different. However, because the application requires preprocessing for all files, the process only differs in that the category values for each data set must be different, and the data must be concatenated into a single worksheet.

6.5.2. Selecting options for pie chart

The forms used for data which contain multiple fields differ from those used for bar charts, so once a user selects a stacked bar chart, a 100% bar chart, or a pie chart, they will be brought to the form seen in figure 42. This form prompts a user to select the chart title, then select the fields that they would like to conclude. In some cases, a user's data may contain a total field which differs from the sum of the subfields. The user has the option to select that field here



The screenshot shows a dialog box titled "KML Chart Generator". It contains the following elements:

- A text input field with the text "Energy Consumption".
- A dropdown menu with a downward arrow.
- A list box containing three items: "Total Final Energy Consumption - Biodiesel (IEA)", "Total Final Energy Consumption - Biogasoline (IEA)", and "Total Final Energy Consumption - Other Liquid Biofuels (IEA)". The first item is selected.
- Two radio buttons: "Percentages" (unchecked) and "Numerical Values" (checked).
- "Back" and "Continue" buttons at the bottom.

Figure 42: Selecting Pie Chart Options

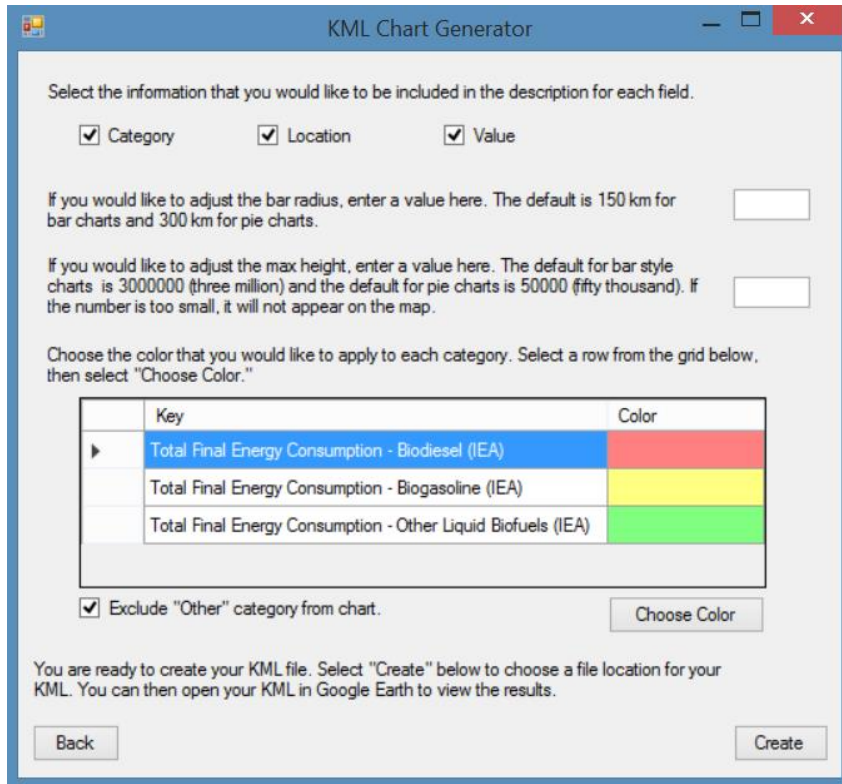


Figure 43: Pie Chart Display Options

as well, then any difference between the total and the subfields will be included as an “Other” field. Another option at this point is to specify whether the data is in percentage or numerical form. Numerical values have no expected maximum value, whereas percentages should add up to 100%.

Once the user has selected these options, they can then select the more aesthetic options for their chart. Again, this form is similar to the bar chart options, but contains options for more fields. This can be seen in figure 43. Once the user has selected the options, they can generate the KML file and again, open it in Google Earth.

One issue faced when dealing with pie charts is that smaller countries tend to overlap, both in GRASS and in the KML generator. In both cases, the functionality is not there to have multiple chart diameters in the same layer. However, one solution is to generate multiple KML

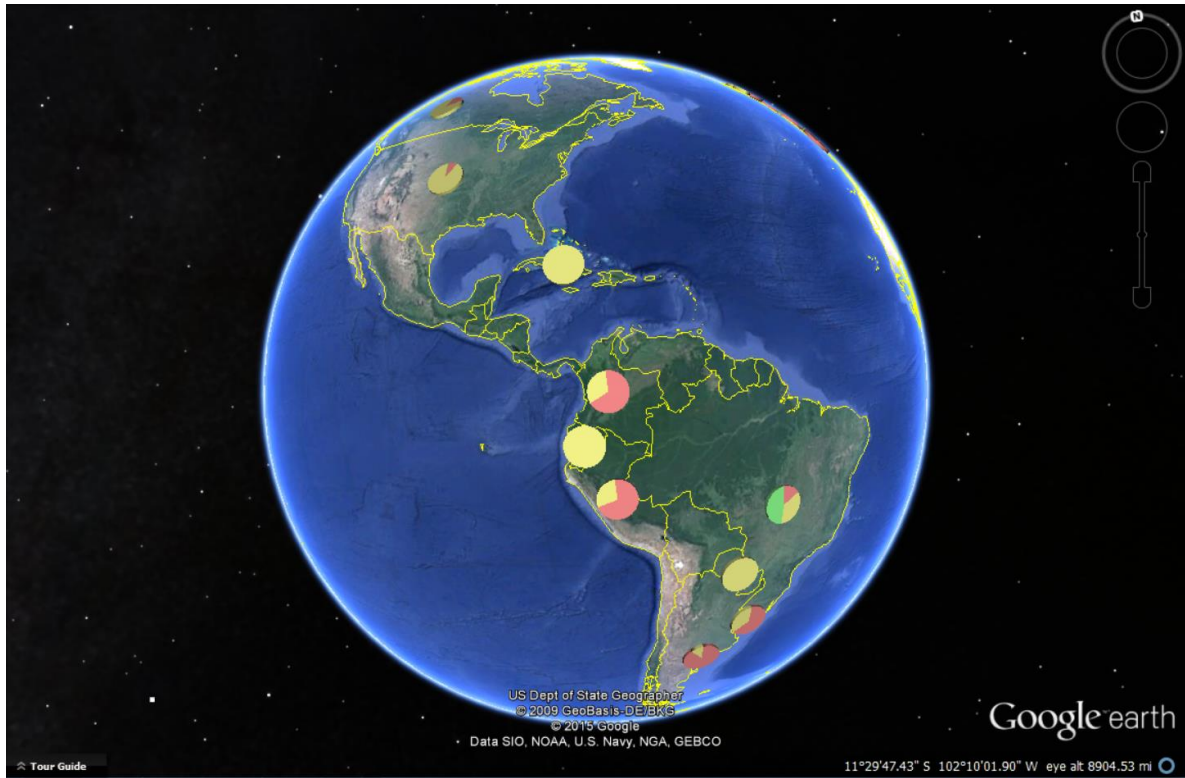


Figure 44: Example of a Pie Chart with the Default Radius

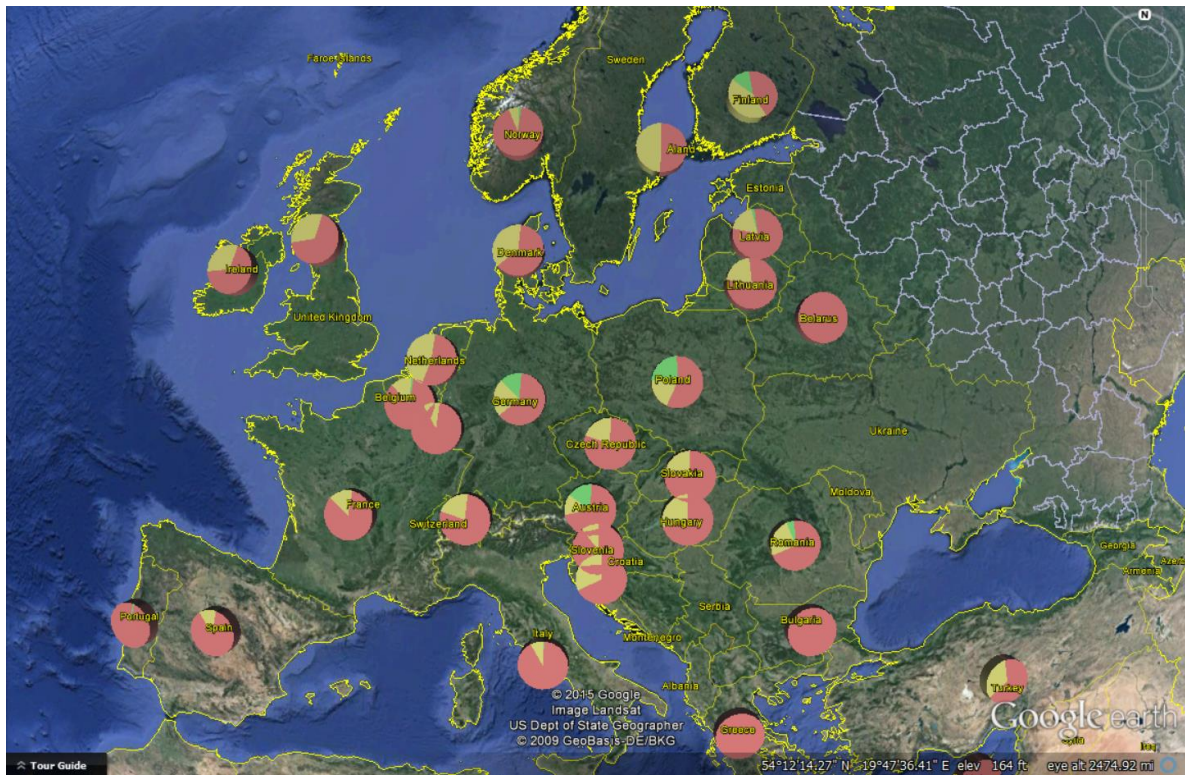


Figure 45: Example of a Pie Chart with a Smaller Radius

layers. In the KML chart generating application, the user can select “Back” after generating the KML file, change the radius, and generate a second KML file. This allows the user to view different parts of the world differently. For example, figure 44 uses the default radius for a pie chart, which works well when the data is spaced out or large countries, but makes the data unintelligible in places where countries are small and data is condensed geographically. In this case, the user could set the radius to a smaller value, which can be seen in figure 45. There is still danger of overlap, in which case the user may choose to use a stacked or 100% bar chart instead.

6.6. General comparison of GRASS GIS and KML generator

6.6.1. Functionality

GRASS GIS differs from the KML generating application in many ways. First, the KML generator is much more limited in its use, restricting the user to a single task. This can be seen as both positive and negative. It does limit what the user can do with the application, but it also keeps the options available from being overwhelming. The GRASS interface has a large number of possible commands that can be executed, and it is up to the user to find the one which works for them. The KML generator restricts the freedom of the user by presenting them with only what they need to get to the next step of the application.

6.6.2. Required knowledge

The KML generating application requires some basic knowledge of how to format data in Excel, along with assuming some knowledge of charts on the part of the user. GRASS GIS requires a much greater knowledge of data formats as well as database concepts. In order to create a chart in the examples given, the user had to manipulate .dbf files along with executing SQL commands and queries. While the KML generating application does perform some querying, that part of the application is hidden from the user, keeping the actual interaction with

the database limited. Additionally, while the application generates KML files, the application requires no knowledge on the part of the user with regards to the file. The user never has to look at the contents of the file in order to use it.

Additionally, the documentation for GRASS GIS is written in a way which assumes knowledge on the part of the user of how to run commands on the command console, how to set flags and parameters, and what those are. Most functions do, however, include a window which simplifies this process for the user. The windows are not shown in the documentation, so it takes some work on the part of the user to find the counterparts of the flags and parameters in the window, along with some trial and error.

6.6.3. Cleaning data

One of the major issues with using GRASS is figuring out how to clean the data once it has been loaded. The Shapefiles in the examples given loaded with many errors which had to be fixed before a chart could be generated. Using the KML generating application, this is not an issue. Google Earth takes care of the actual map and making sure that it loads correctly within the application, so the user only has to worry about their data. This modularity between the two applications allows the user to focus on the layers they would like to add, without worrying about the map.

6.6.4. User interface interaction

Interacting with the map display in GRASS GIS is very different than that of Google Earth. GRASS loads slowly between commands, does not allow for simple panning in the way that is expected from most modern interfaces. In order to zoom, the user must select the zoom icons from the interface menu before clicking and waiting for the new data to load. Google Earth not only allows the user to grab and spin the globe, but the interface of Google Earth Desktop is

also partially touch screen compatible. The application doesn't include quite as much of the capabilities as Google Maps does, but much of the touch screen commands are implemented in Google Earth. The interface reaction time is also significantly shorter when it comes to loading new areas of the map, as well as with zooming.

6.7. Performance

The performance of the system can be measured by the major functions of the application. The two primary functions are reading Excel files, and creating KML files. However, reading Excel files can be broken into two categories. The first category is reading Excel files when no querying is necessary. The second category is reading Excel files with querying included.

6.7.1. Reading Excel files

In order to measure the performance of file reading, first the malaria sample data was used, and the number of milliseconds it took to run the file was measured. This was performed

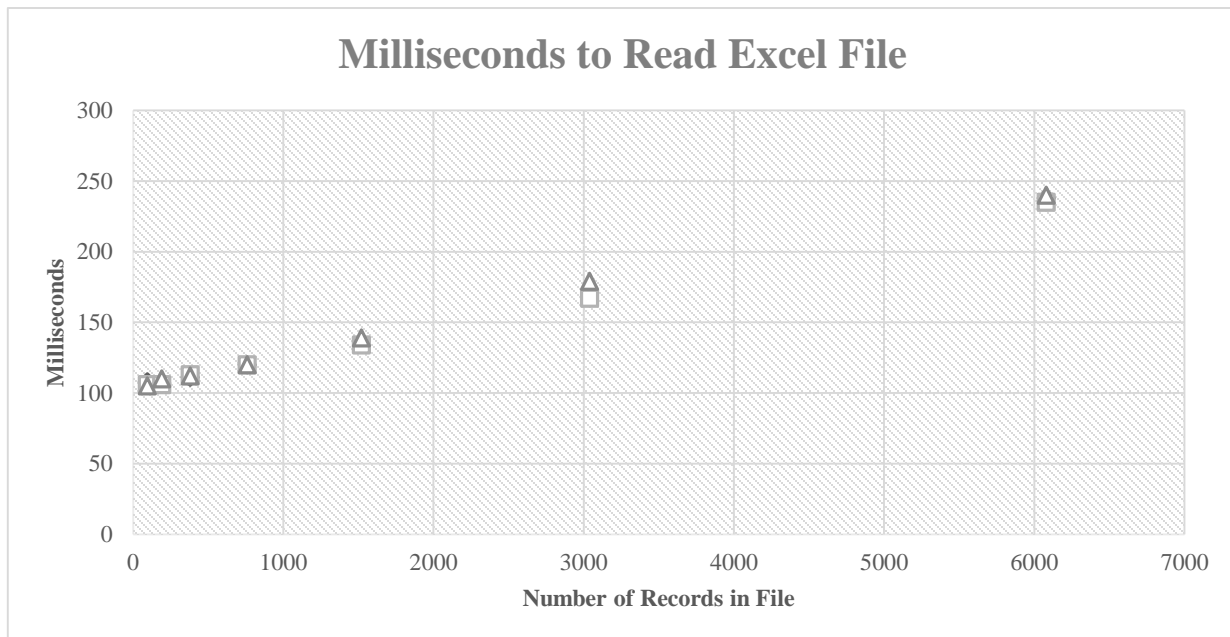


Figure 46: Application Performance for File Reading When No Querying is Necessary

three times in order to ensure accurate and consistent measurements. The file was then doubled in size, and the process was repeated. This was tested for 95, 190, 380, 760, 1,520, 3,040, and 6,080 records. The tests were capped here because it is unlikely that a user would be able to create a readable map with more than this number of Polygons in a single KML file. The results of this test can be seen in figure 46. The simple reading of Excel files without any querying of the database does not create a large bottleneck in performance. At 6,080 records, file reading took less than 250 milliseconds.

A much greater bottleneck occurs when information must be queried for each record in the Excel file. In this case, the latitude and longitude values were stripped from the malaria data set, and the performance was tested again while querying by country name for each of the values.

The results of this analysis can be viewed in figure 47. There is a linear increase in the amount of time taken to finish the process of reading the file when a query must be performed for each record in the data set. It took an average of 258 milliseconds to read and query latitude

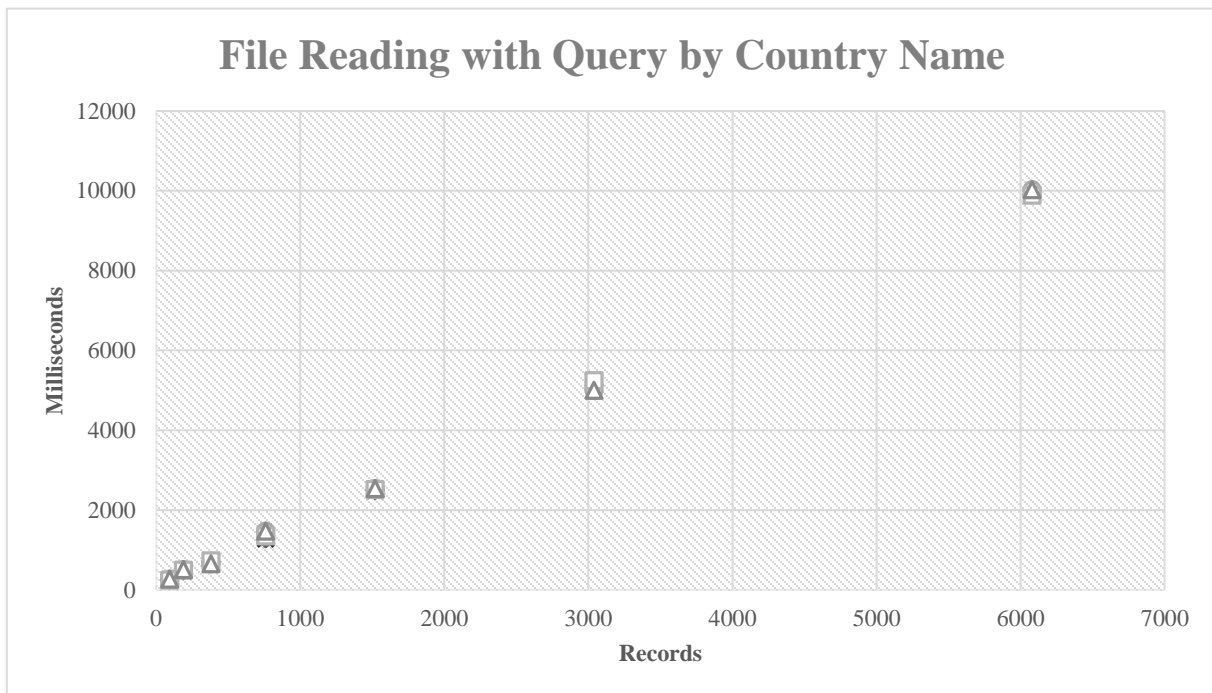


Figure 47: Application Performance When Querying by Country Name

and longitude by country name for a file containing 95 records. However, in a file of 6,080 records, the average grows to 9,940.67 milliseconds. The delay caused by querying latitude and longitude, especially for large data sets, is the main bottleneck that occurs in the application.

6.7.2. Creating KML files

The final step in the application is creating KML files. Once a user has selected the file location, the function is called which creates the KML file. The same file sizes from the file reading performance analysis were used in order to analyze the performance of KML creation. The average time taken across three trials for a file with 95 records was 54.3 milliseconds. This increases to 100 milliseconds for a file with 380 records, and 1,136 milliseconds for a file of 6,080 records. Figure 48 shows the increase in time taken to create KML files by the number of records included in the data file.

Because of the step-by-step nature of the application, these bottlenecks are not problematic in smaller values, as they do not accumulate between steps taken by the user.

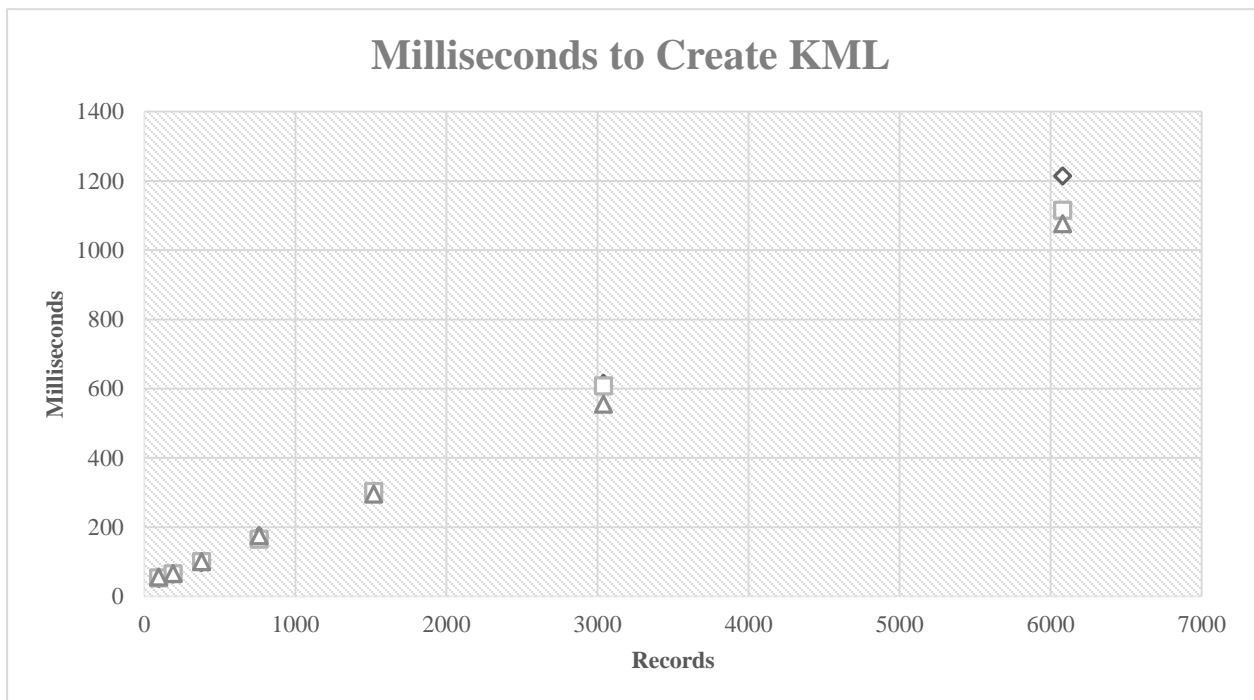


Figure 48: Application Performance for KML Creation

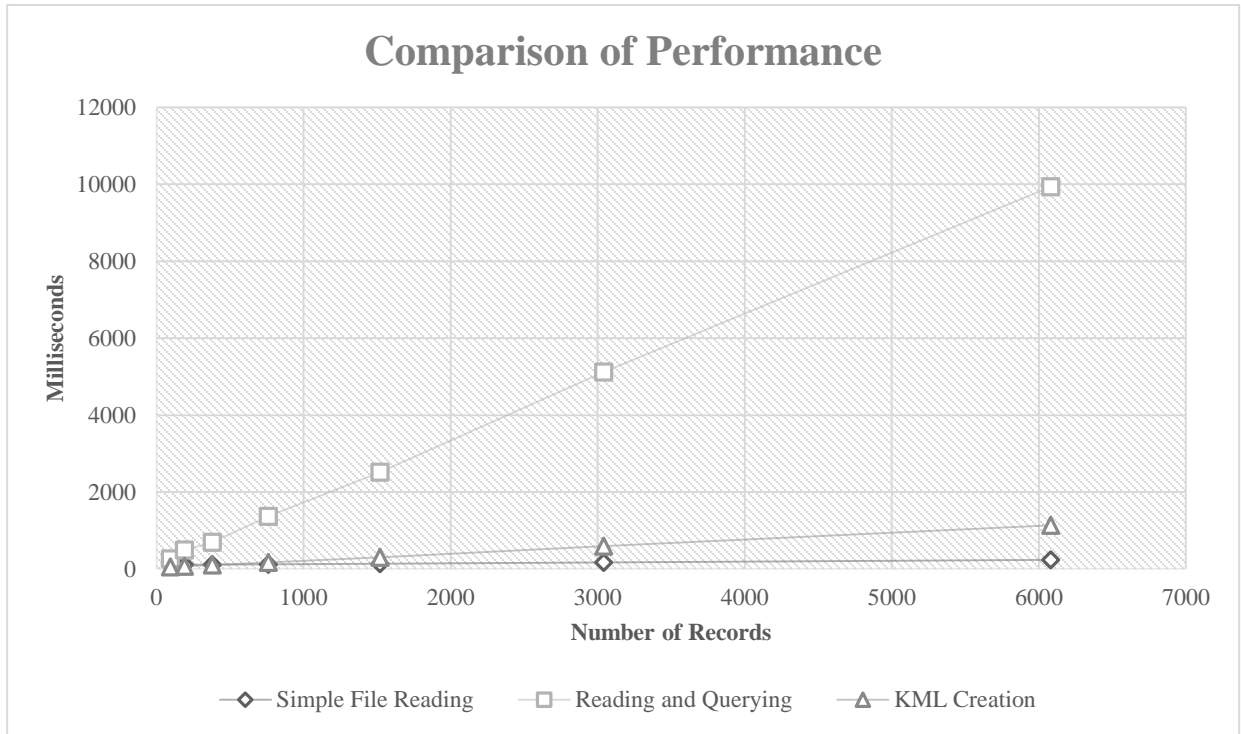


Figure 49: Comparison of File Reading with and without Querying, As Well as KML Creation

However, they become more noticeable as file size increases, especially when querying is necessary. In figure 49, a comparison of the average processing time required for each of the activities can be seen. This shows the significant processing time that querying takes relative to file reading and KML creation.

6.8. Sample output

The sample data provided in the application can be used to create a chart in each of the styles provided by the application. In addition to the simple bar chart and the pie chart, which have already been reviewed, the user has the option to create a stacked bar chart, a pie chart, and a 100% bar chart. While an example of pie chart creation has been shown, this section will differ in that it shows the sample data output using the pie chart so that it can be viewed in comparison to the 100% stacked bar chart. The output using the sample data set for each of these options will be reviewed in the following sub-sections.

6.8.1. Stacked bar chart

The stacked bar chart can be created using the CO2 Emissions data set [39], which shows the total CO2 emissions by country, along with the type of sources of these emissions (liquid, solid, or gaseous fuel consumption). The output for this sample data file can be seen in figure 50. Each type of fuel consumption can be viewed separately as subcategories of the total CO2 emissions for each countries. One drawback of this type of chart can be seen in this example. The CO2 emissions for China are so much greater than all of the other surrounding countries, that it becomes difficult to view the separate categories of countries with lower total CO2 emissions. The categories corresponding with each color can be seen in the navigation pane, if the user chooses to include that information. In this case, green corresponds with CO2 emissions from solid fuel consumption, yellow corresponds with CO2 emissions from liquid fuel consumption, and the reddish color corresponds with CO2 emissions from gaseous fuel

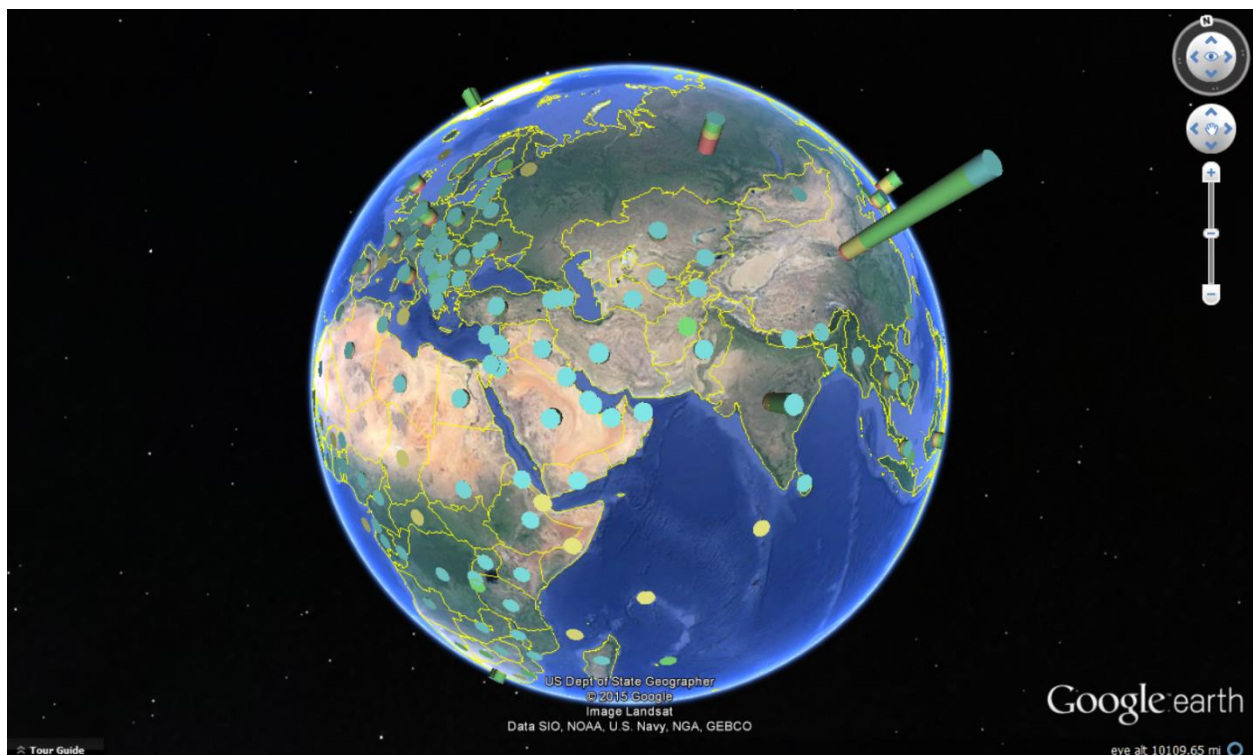


Figure 50: Stacked Bar Chart Example Using CO2 Emissions Data Set

consumption. The aqua color corresponds with the “Other” category, which is an optional field which can be included by the user. Because there is a total CO2 emissions category, in any case where the sum of the total emissions for each of the subcategories does not equal the total CO2 emissions, the difference is categorized as “Other”.

6.8.2. 100% bar chart

The income share sample data set [39] can be used to create both the 100% bar chart and the pie chart. This data set shows each country’s income share by portion of the population. The results of using this data set to create a 100% bar chart can be seen in figure 51. In this chart, the highest earning 20% of the population shows up in yellow, while the lowest 20% of the population shows up in green.

Using the 100% bar chart makes comparisons between countries across the world more difficult in this case, as many of the proportions of wealth carried by portions of the population

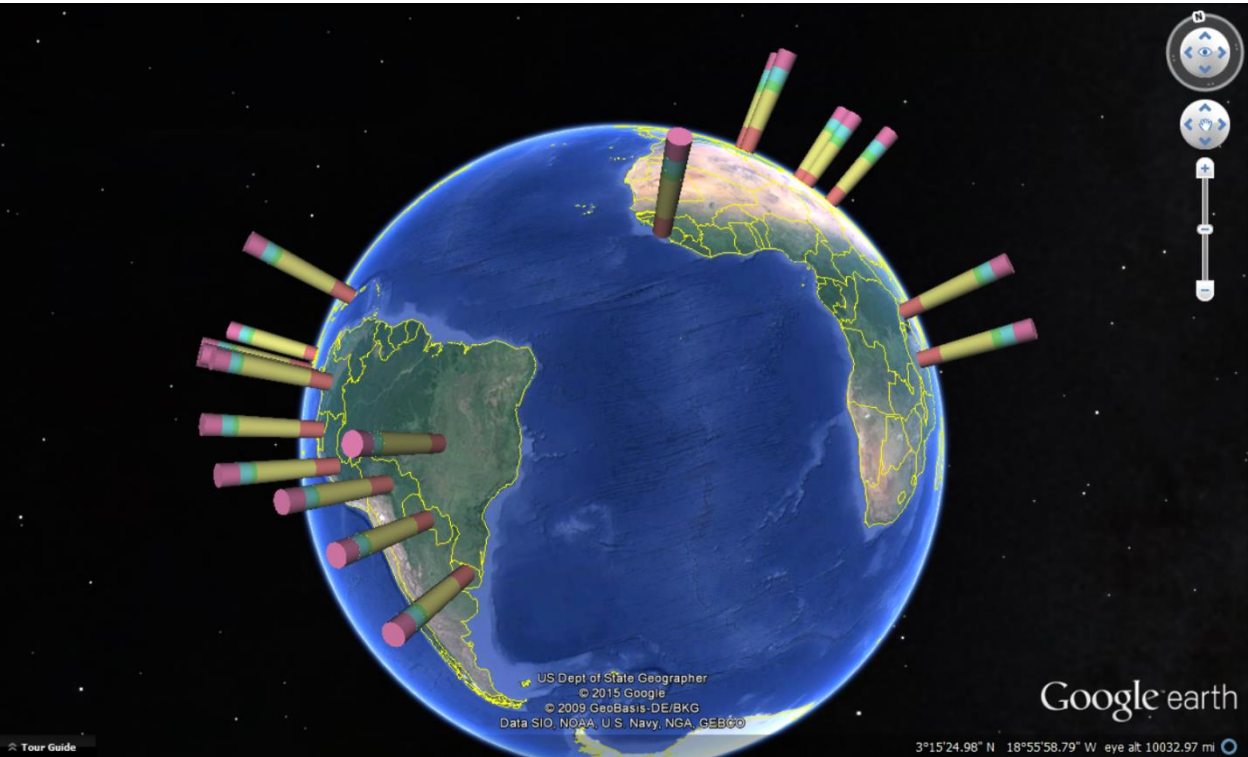


Figure 51: 100% Bar Chart Example Using Income Inequality Data Set

are similar across countries. However, it does give the user an idea of how income is dispersed across populations throughout the world. More generally, this option allows users to compare multiple internal variations in trends within locations. These variations can be percentages or numeric values.

6.8.3. Pie chart

The pie chart can also be used to view the income share sample data set [39]. The results of using this data set can be seen in figure 52. While the application allows the user to create multiple pie charts, the major drawback of using this type of chart is that if countries are too close together, the Polygons may overlap, making it difficult to view the data for smaller countries with close geographic proximity to one another. In figure 52, this is visible in Central America. This is a greater issue in Pie Charts than in any of the bar charts, because the data is laid out horizontally rather than vertically. This means that horizontal overlap causes a loss of

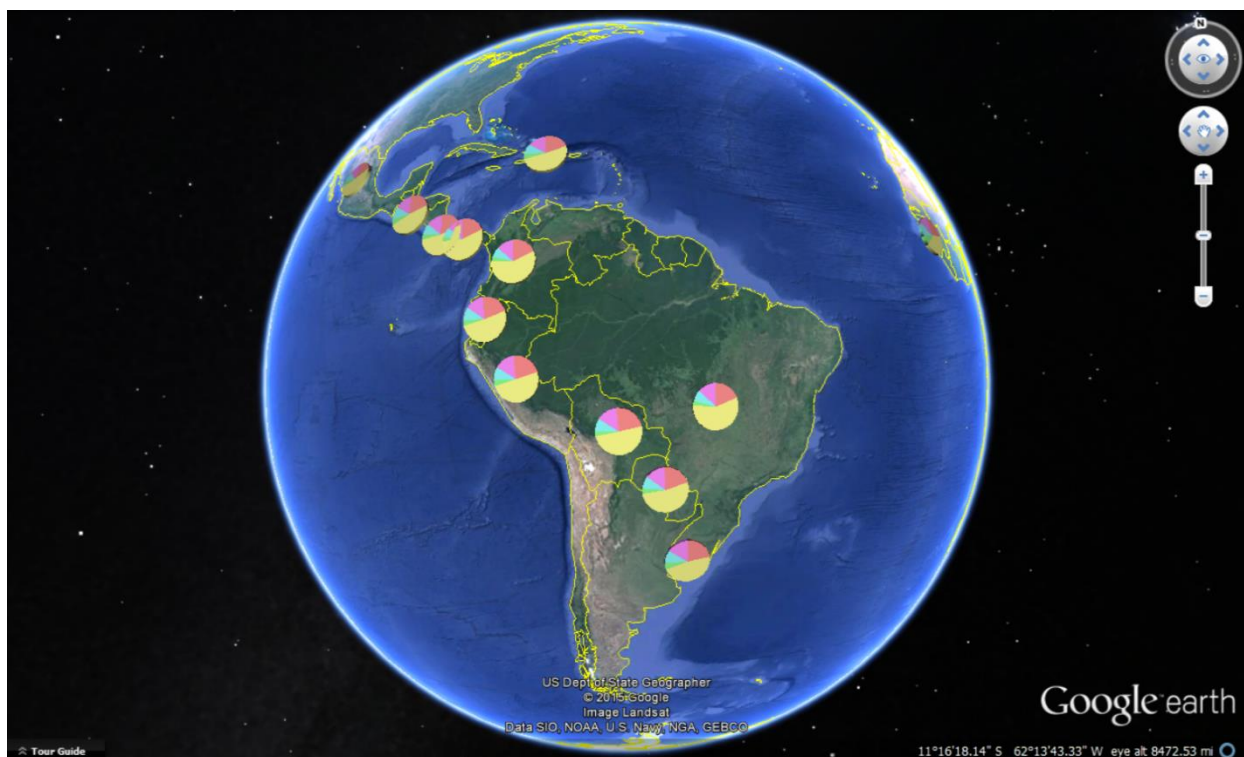


Figure 52: Pie Chart Example Using Income Inequality Data Set

clarity. The user can overcome this issue by choosing a chart radius which allows the data to be viewed without overlapping. They also have the option of choosing a 100% bar chart for their purposes if the first option is not feasible. If the user knows that the countries being mapped are close together geographically, and have a smaller land area, they should consider making the pie chart smaller, or using a 100% bar chart to represent the data.

6.9. Other software design considerations

Because of its nature as a KML generating application, the functionality of the output is highly portable. The files generated can be opened in multiple applications which support KML. They can also be shared between users, as the files generated can be viewed as text files. Additionally, because KML has been standardized by the OGC, it is unlikely to change soon or drastically, meaning that the output of this application will be relevant for a greater period of time.

In traditional GIS applications, the functionality of chart generation is integrated with the application itself. By separating this functionality out in the KML chart generating application, while using a standard visualization language, a greater degree of modularity is created between the conversion of data into geospatial coordinates and the display of the generated layers. As long as the KML standards remain the same, the virtual globe can be updated and adapted without affecting the ability to open the KML files generated by the application. Additionally, the application does not rely on the virtual globe in any way as long as the virtual globe uses KML, so it can be updated and extended without considerations of virtual globe technology in mind.

Another benefit of this application is that, because it is kept completely detached from any specific GIS, it is unaffected by changes in technology or support for the applications which

allow for the opening of KML files. For example, Google announced that they were stopping support for the Google Earth API this year. While this means that the API is unavailable to users, it does not affect the KML generator. The files created using this application can still be opened in Google Earth Desktop. Google has also added an “Earth View” to the Google Maps engine. When the user zooms out in satellite view, they can see the Earth in 3-D. Currently, this has not been extended to the custom maps that the user can create. So while the KML files created by the user can be opened in the custom maps part of Google Maps, they cannot be viewed in 3-D. However, should this change, the user can easily open up their file and create a custom map. Not only that, but if a user would prefer to use NASA World Wind, or Esri ArcExplorer, they are able to. Any future virtual globe technology would be open to the user as well. By keeping the data visualization separate from the GIS, the user gains a great deal of flexibility.

7. CONCLUSION

Despite its growing popularity and usefulness as a tool in a globalizing world, implementing traditional GIS technology has proven problematic for non-specialized users. More commonly used applications such as Google Earth and Google Maps have removed much of the GIS functionality and completely changed the data format which the platform is based on. Non-specialized users view traditional GIS tools as too complex or time consuming to learn. While past efforts have focused on increasing the level of support for implementing traditional GIS technology, this thesis has proposed simplifying GIS by changing the way it is typically structured.

By separating out some of the functionality of GIS in a usable way through the use of virtual globe technology, users can be introduced to GIS concepts with no need for any extension of the virtual globe itself. Through the use of standardized languages, such as KML, functionality can be added back into applications while still keeping it modular. This allows for changes to be made on both ends, whether it be updates to the virtual globe or the added functionality, without changing the interaction between the two. Because KML is an OGC standardized language, it is a good example of how this can be achieved.

The chart generating application developed here provides an example of how GIS concepts can be made more usable for and accessible to a greater number of people, by providing an accessible introduction to some of the functionality of GIS. The step-by-step nature of the application requires little knowledge or planning to use, guiding the user through each step in each form which comprises the user interface, so they know what to do next in order to achieve their goals. It also removes the need for the user to have any knowledge of database concepts, which is sometimes necessary when dealing with GIS. This procedural technique is one which

takes the knowledge and experience of the user in mind, with the goal of making the application more understandable and less seemingly complex to non-specialized users.

The charts developed using this application can be used to generate an understanding of global issues within a geospatial context. Using the sample data, users have the ability to view data mapped using the application without the need for generating their own spreadsheet with relevant information. This allows the user to easily map and view trends which allow them to answer questions such as how health issues affect different countries at different rates, which countries are emitting the most CO₂, and how income is dispersed across the population of a country. While the application does not provide any of the statistical tools available in GIS software, its goal is to provide an introduction to mapping technology, and expand the use of virtual globes in a way that is interesting and engaging. It also provides flexibility by allowing the user to make simple bar charts, stacked bar charts, 100% bar charts, and pie charts, all centered at any latitude and longitude which the user wishes.

By separating out some of the functionality of GIS, users can take advantage of the user-friendly interface provided by Google Earth, allowing them to view information and satellite images in great detail. While it is possible that virtual globes may introduce data visualization capabilities back into their applications, through the use of KML generating applications, they can keep their current capabilities intact while allowing for the addition of functionality for the user.

REFERENCES

- [1] A. Gore, *The Digital Earth: Understanding our planet in the 21st Century*, Los Angeles, CA, 1998.
- [2] National Academy of Sciences, *Learning to think spatially-GIS as a support system in the K-12 curriculum*, Washington DC: The National Academies Press, 2006.
- [3] J. Lee and R. Bednarz, "Effect of GIS Learning on Spatial Thinking," *Journal of Geography in Higher Education*, vol. 33, no. 2, pp. 183-198, 2009.
- [4] T. R. Baker and S. H. White, "The Effects of G.I.S. on Students' Attitudes, Self-efficacy, and Achievement in Middle School Classrooms," *Journal of Geography*, vol. 102, no. 6, pp. 243-254, 2003.
- [5] The 6th International Symposium on Digital Earth -- Digital Earth in Action, "Beijing Declaration on Digital Earth," 12 09 2009. [Online]. Available: english.ceode.cas.cn/zt/isde6en/hyqx11.html. [Accessed 19 02 2015].
- [6] J. J. Kerski, "The Implementation and Effectiveness of Geographic Information Systems Technology and Methods in Secondary Education," *Journal of Geography*, vol. 102, no. 3, pp. 128-137, 2003.
- [7] A. Demirci, "How do Teachers Approach New Technologies: Geography Teachers' Attitudes towards Geographic Information Systems (GIS)," *European Journal of Educational Studies*, vol. 1, no. 1, pp. 43-53, 2009.
- [8] A. Demirci and A. Karaburun, "How to Make GIS a Common Educational Tool in Schools: Potentials and Implications of the GIS for Teachers Book for Geography

- Education in Turkey," *Ozean Journal of Applied Sciences*, vol. 2, no. 2, pp. 205-215, 2009.
- [9] L. Y. Yap, G. C. I. Tan, X. Zhu and M. C. Wettasinghe, "An Assessment of the Use of Geographical Information Systems (GIS) in Teaching Geography in Singapore Schools," *Journal of Geography*, vol. 107, pp. 52-60, 2008.
- [10] S. Liu and X. Zhu, "Designing a Structured and Interactive Learning Environment Based on GIS for Secondary Geography Education," *Journal of Geography*, vol. 107, no. 1, pp. 12-19, 2008.
- [11] P. G. Polson, C. Lewis, J. Rieman and C. Wharton, "Cognitive walkthroughs: a method for theory-based evaluation of user interfaces," *Int. J. Man-Machine Studies*, vol. 36, pp. 741-773, 1992.
- [12] R. Spencer, "The Streamlined Cognitive Walkthrough Method, Working Around Social Constraints Encountered in a Software Development Company," Microsoft Corporation, Redmond, WA, 2000.
- [13] S. Gao and M. F. Goodchild, "Asking Spatial Questions to Identify GIS Functionality," in *Fourth International Conference on Computing for Geospatial Research and Application*, San Francisco, CA, 2013.
- [14] B. T. Tuttle, S. Anderson and R. Huff, "Virtual Globes: An Overview of Their History, Uses, and Future Challenges," *Geography Compass*, vol. 2, no. 5, pp. 1478-1505, 2008.
- [15] D. Butler, "The web-wide world," *Nature*, vol. 439, pp. 776-778, 2006.

- [16] J. Schöning, B. Hecht, M. Raubal, A. Kruger, M. Marsh and M. Rohs, "Improving Interaction with Virtual Globes through Spatial Thinking: Helping Users Ask "Why?"," in *Proceedings of the 13th international conference on Intelligent user interfaces*, 2008.
- [17] B. McClendon, "Google Official Blog," 05 October 2011. [Online]. Available: <http://googleblog.blogspot.com/2011/10/google-earth-downloaded-more-than-one.html>. [Accessed 24 02 2015].
- [18] NASA World Wind, "Get Started," Wordpress, [Online]. Available: <http://goworldwind.org/getting-started/>. [Accessed 08 March 2015].
- [19] Esri, "Explorer for ArcGIS," Esri, [Online]. Available: <http://www.esri.com/software/explorer-for-arcgis>. [Accessed 08 March 2015].
- [20] S. Marquardt, "Google Maps," 30 January 2015. [Online]. Available: <http://google-latlong.blogspot.com/2015/01/google-earth-pro-is-now-free.html>. [Accessed 3 March 2015].
- [21] R. Kim, "NASA World Wind," NASA, 18 July 2011. [Online]. Available: <http://worldwind.arc.nasa.gov/>. [Accessed 03 March 2015].
- [22] Esri, "ArcGIS Platform," Esri, [Online]. Available: <http://www.esri.com/software/arcgis/>. [Accessed 03 March 2015].
- [23] GRASS Development Team, "GRASS GIS," GRASS Development Team, [Online]. Available: <http://grass.osgeo.org/>. [Accessed 03 March 2015].
- [24] S. Alban, "Independent Report Highlights Esri as Leader in Global GIS Market," 02 March 2015. [Online]. Available: <http://www.esri.com/esri-news/releases/15->

- 1qtr/independent-report-highlights-esri-as-leader-in-global-gis-market?_ga=1.117602697.1673125595.1421177408. [Accessed 03 March 2015].
- [25] GRASS Development Team, "Database management in GRASS GIS," GRASS GIS, [Online]. Available: <http://grass.osgeo.org/grass64/manuals/databaseintro.html>. [Accessed 8 3 2015].
- [26] Esri, "Data Management," Esri, [Online]. Available: <http://www.esri.com/products/arcgis-capabilities/data-management>. [Accessed 8 March 2015].
- [27] Google, "Data management: Best practices," 17 April 2013. [Online]. Available: <https://support.google.com/earthenterprise/answer/3060030?hl=en>. [Accessed 08 March 2015].
- [28] GRASS Development Team, "General Overview," 24 February 2015. [Online]. Available: <http://grass.osgeo.org/documentation/general-overview/>. [Accessed 08 March 2015].
- [29] Esri, "ArcGIS for Desktop Functionality Matrix," 2012. [Online]. Available: <http://www.esri.com/library/brochures/pdfs/arcgis10-functionality-matrix.pdf>. [Accessed 8 March 2015].
- [30] S. Bacharach, "OGC Approves KML as Open Standard," 14 April 2008. [Online]. Available: <http://www.opengeospatial.org/node/857>. [Accessed 09 March 2015].
- [31] WHO, "Global Health Observatory Data Repository," World Health Organization, 2014. [Online]. Available: <http://apps.who.int/gho/data/?theme=main>. [Accessed 10 March 2015].
- [32] UNEP, "Environmental Data Explorer," United Nations Environment Programme, 2014. [Online]. Available: <http://geodata.grid.unep.ch/#>. [Accessed 10 March 2015].

- [33] K. Hoetmer, "Geo Developers Blog: Announcing deprecation of the Google Earth API," 12 December 2014. [Online]. Available: <http://googlegeodevelopers.blogspot.com/2014/12/announcing-deprecation-of-google-earth.html>. [Accessed 10 March 2015].
- [34] B. Sandvik, "Using KML for Thematic Mapping," University of Edinburgh, Edinburgh, 2008.
- [35] samcragg, "SharpKml," CodePlex, 18 November 2013. [Online]. Available: <https://sharpkml.codeplex.com/>. [Accessed 18 April 2015].
- [36] "Calculate distance, bearing and more between Latitude/Longitude points," Movable Type Scripts, [Online]. Available: <http://www.movable-type.co.uk/scripts/latlong.html>. [Accessed 27 April 2015].
- [37] R. Blazek, "d.vect.chart," GRASS GIS, 08 November 2011. [Online]. Available: <http://grass.osgeo.org/grass64/manuals/d.vect.chart.html>. [Accessed 28 April 2015].
- [38] World Health Organization, "Global Health Observatory Data Repository," WHO, 2014. [Online]. Available: http://apps.who.int/gho/data/node.main.WHS3_48?lang=en. [Accessed 29 April 2015].
- [39] The World Bank, "World DataBank Poverty and Inequality Database," The World Bank Group, 2015. [Online]. Available: <http://databank.worldbank.org/data/views/variableselection/selectvariables.aspx?source=poverity-and-inequality-database#>. [Accessed 29 April 2015].