ABSTRACT

Live Oak Peninsula is located within the Corpus Christi/Aransas Bay system, hydrologically connected to Nueces and Aransas/Mission watersheds, as well as Gulf of Mexico through Aransas Pass. Three municipalities located along the eastern shoreline are: Fulton-Rockport, Aransas Pass, and Ingleside. Urban growths as well as industrial operations have historically occurred along this shoreline. The purpose of this study is to quantify the change in land use that have occurred in the past ten years in this area by developing a Geographic Information System application using different data sets of different time period and integrate, analyze and process these data sets and display the changes in the form of dynamic reports, dynamic graphs and comprising of all the tools like pan, zoom-in, zoom-out, overview, magnifier etc which are required to manipulate and access the data.
# TABLE OF CONTENTS

Abstract .................................................................................................................. ii

Table of Contents .................................................................................................. iii

List of Figures ....................................................................................................... vi

List of Tables ........................................................................................................ vii

1. Introduction and Background .......................................................................... 1
   1.1 Live Oak Peninsula ....................................................................................... 2

1.2 Data .................................................................................................................. 5
   1.2.1 1995 DOQQ’s ............................................................................................. 5
   1.2.2 2001 Landsat Image ................................................................................ 7

1.3 Image Correction, Processing and Clipping .................................................. 8

1.4 Change Detection Approach ......................................................................... 9
   1.4.1 Classified maps ......................................................................................... 9
   1.4.2 Statistics .................................................................................................. 10
   1.4.3 Image Maps ............................................................................................. 10

2. LiveOakPro Application for Monitoring and Mapping of Live Oak Peninsula .. 12
   2.1 Monitoring And Mapping LiveOak Peninsula Application ......................... 12

3. System Design and Research ........................................................................... 13
   3.1 Requirement Definition .............................................................................. 14
       3.1.1 Analysis ............................................................................................... 15
LIST OF FIGURES

Figure 1.1 Location of study area, Live Oak Peninsula, Texas, located in Aransas/Corpus Christi Bay system .................................................................4

Figure 1.2 LiveOak Peninsula DOQQ’s 1995......................................................6

Figure 1.3 LiveOak Peninsula Landsat 2001.......................................................7

Figure 3.1 Application Development Process Cycle............................................13

Figure 3.2 Basic Objects in the Geodatabase Model...........................................20

Figure 3.3 Map Viewer Interface.....................................................................22

Figure 3.4 Toolbar.............................................................................................27

Figure 3.5 Status bar..........................................................................................28

Figure 3.6 Magnifying Glass..............................................................................29

Figure 3.7 Overview Windows.........................................................................40
LIST OF TABLES

Table 1.1 LiveOak Data of Different Time Periods……………………………………….5
Table 1.2 1995 Digital Ortho Quarter Quadrangle……………………………………..7
Table 1.3 LiveOak Peninsula Landsat 2001…………………………………………….8
1. INTRODUCTION AND BACKGROUND

Geographic Information System is a technology used to view and analyze data from a geographic perspective. The technology is a piece of an organization's overall information system framework. GIS links location to information (such as people to addresses, buildings to parcels, or streets within a network) and layers that information to give a better understanding of how it all interrelates.

The roads are designed using GIS to analyze traffic volumes and other pertinent data to determine the best locations, materials, and maintenance schedules. The forests are managed using GIS to analyze health issues, harvesting capabilities, reforestation opportunities, fire dangers, and more. Most utility companies rely on GIS to develop the infrastructure that allows their companies to function; water, electrical, phone and oil and gas lines are mapped, monitored, and analyzed all using GIS. Law enforcement is using GIS to plot and track crimes and correlate crime statistics helping to monitor and even predict the probability of a crime occurring.

The overall goals of this project were to develop, test, and demonstrate an accessible methodology for mapping and monitoring Live Oak Peninsula with aerial imagery and shapefiles for changed area. The main objective of this research project was to assess change-detection approaches in digital land use change display by integrating computer science approach and geographic information system. The importance of change-detection methods relies on the possibility of identifying changes that occurred in land covers (e.g., due to urban expansion, deforestation, habitat, etc.) by analyzing aerial images acquired at different dates by visual interpretation.
1.1 Live Oak Peninsula

Live Oak Peninsula is located within the Corpus Christi/Aransas Bay system, hydrologically connected to Nueces and Aransas/Mission watersheds, as well as Gulf of Mexico through Aransas Pass (Figure 1.1). Geologically, the peninsula was formed by longshore Gulf processes and is a Pleistocene barrier-strandplain. The elevations are uniformly below 30 feet above mean sea level, with the gulfward portion overlain by varying depths of sand. The vegetation is comprised of coastal woodland (primarily Live Oak-Red Bay Association), coastal prairie, and freshwater depressional wetlands. These habitats harbor a high diversity of resident wildlife, as well as support high numbers of migratory neotropical birds.

Humans have occupied Live Oak Peninsula for several hundred years, primarily due to its proximity to Gulf and bay waters. Fishery communities, processing plants, and harbors maintained the local economy for several decades. As fishery population declined, tourism increased as an economic alternative. Three municipalities are located along the eastern shoreline, Fulton-Rockport, Aransas Pass, and Ingleside. Urban growth has historically occurred along this shoreline, as well as industrial operations. Rural residents were typically located close to Hwy 35. The construction of the Hwy 35 bypass effectively divides the peninsula from north to south. The native vegetation was cleared for several hundred yards along the highway, fragmenting coastal woodland complexes. The predicted increase in rural growth has been realized, as roads are developed perpendicular to the highway and into the coastal woodland complex. With the inception of Solid Waste
Agency of Northern Cook County decision, it is likely several hundred wetlands have been filled with this development surge.

Since most of the Peninsula is not located within municipal city limits, rural growth remains unplanned. Native habitat continues to be fragmented by roads, right of ways, and residential and commercial development. The purpose of this study is to quantify the change in land use that have occurred in the past ten years, as well as identify habitat tracts that still have functional size and quality for conservation actions. This information will be available to federal and state agencies as well as regional conservation organizations [CBBEP 2002].
Figure 1.1 Location of study area, Live Oak Peninsula, Texas, located in Aransas/Corpus Christi Bay system. (Adapted from [CCS 2001])
1.2 Data

Aerial data of Live Oak Peninsula of different time period has been analyzed in this research project along with some shapefile. The more elaborated data table is shown below in Table 1.1.

Table 1.1 Live Oak Data of Different Time Periods

<table>
<thead>
<tr>
<th>Data Used</th>
<th>Scale</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1995 Digital Ortho Quarter Quadrangle (DOQQ)</td>
<td>1 meter resolution</td>
<td>US Geological Survey</td>
</tr>
<tr>
<td>2001 Landsat Image</td>
<td>40 meter resolution</td>
<td>US Geological Survey</td>
</tr>
<tr>
<td>Panchromatic (black-and-white)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shapefiles of Land Use/Land Cover Polygons of new construction.</td>
<td>1:24000</td>
<td>Digitized in ArcView</td>
</tr>
<tr>
<td>Shapefiles of Land Use/Land Cover Polygons of Excavated Sand.</td>
<td>1:24000</td>
<td>Digitized in ArcView</td>
</tr>
<tr>
<td>Shapefiles of Land Use/Land Cover Polygons of Cleared Area.</td>
<td>1:24000</td>
<td>Digitized in ArcView</td>
</tr>
<tr>
<td>Shapefiles of Land Use/Land Cover Lines of new roads</td>
<td>1:12000</td>
<td>Digitized in ArcView</td>
</tr>
</tbody>
</table>

1.2.1 1995 DOQQ’s

The 1995 Digital Ortho Quarter Quadrangle (DOQQ) image of Live Oak Peninsula was obtained from United States Geological (USGS) as an MrSid file at a resolution of 1 meter consists of 13 squares. A Digital Orthophoto Quadrangle (DOQQ) is a computer-generated image of an aerial photograph in which the image displacement caused by terrain relief and camera tilt has been removed. The DOQQ combines the image characteristics of the original photograph with the georeferenced qualities of a map.
DOQQs are black and white (B/W), natural color, or color-infrared (CIR) images with 1-meter ground resolution, table 1.2 gives more information about 1995 DOQQ.

Table 1.2 1995 Digital Ortho Quarter Quadrangle

<table>
<thead>
<tr>
<th>Coordinate System</th>
<th>UTM, Zone 16 North</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Datum</td>
<td>NAD83</td>
</tr>
<tr>
<td>Pixel Resolution</td>
<td>1 meter</td>
</tr>
<tr>
<td>File Format</td>
<td>MrSID</td>
</tr>
</tbody>
</table>
1.2.2 2001 Landsat Image

Landsat is a series of satellites that produce images of the earth. The LANDSAT remote sensing satellite program was developed by NASA (National Aeronautics and Space Administration). Landsat images are not high-resolution images like Digital Ortho Quarter Quadrangle (DOQQ); 2001 Live Oak Peninsula (Figure 1.3) is a mosaic with resolution of 30 meters with 3 bands of color.

Figure 1.3 Live Oak Peninsula Landsat 2001
1.3 Image Correction, Processing and Clipping

The aerial data of 2001 was processed using Arcview 9.0 software including preprocessing, enhancement, georectification and clipping of the size equivalent to the 1995 DOQQ. In a first step the 2001 images were georectified in order to remove geometric distortions introduced by the image acquisition. Geometric distortions are for example, caused by topographical variations in the earth surface and the tilt of the camera affecting the distance with which features on the aerial image are displayed. Then the image is georeferenced to the surface of the earth and fit to the Universal Transverse Mercator projection and finally, they were clipped to the size of DOQQ block.

<table>
<thead>
<tr>
<th>Coordinate System</th>
<th>UTM, Zone 16 North</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Datum</td>
<td>NAD83</td>
</tr>
<tr>
<td>Pixel Resolution</td>
<td>30 meter</td>
</tr>
<tr>
<td>File Format</td>
<td>tif</td>
</tr>
</tbody>
</table>
1.4 Change Detection Approach

There are different approaches to achieve any land use change project using aerial imagery. Before starting a land cover change project, it is important to define some objectives concerning what one plan to gain from the analysis. Reviewing these goals provides insight into what methods were necessary to achieve specific objectives. Here is a list of common objectives:

- Identify areas of deforestation/reforestation
- Monitor growth of urban or rural populations
- Predict future change based on past change
- Monitor changes in species habitat
- Monitor changes in agriculture patterns.

Another question that was answered before designing a land cover change project deals with the type of output that can be generated from a land cover project. Land cover change output can be grouped into three categories: classified maps, statistics, and image maps.

1.4.1 Classified maps

Classified maps are the most typical form of land cover change output. These maps group the landscape into discrete change classes, such as: forest to non-forest, forest unchanged, nonforest to forest, and non-forest unchanged. Perhaps the main advantage of this method is that it provides mapped output that is a necessary format for automated spatial analysis, such as modeling and analysis of landscape metrics. Once an accurate baseline change map is generated, it can be updated periodically, often at a fraction of the time and money required to produce the baseline map. Statistics can be easily generated
from these mapped products, so in some sense, one effectively obtains all three output products when producing classified maps. In spite of all these advantages, classified maps have some disadvantages. For example, the classification process can be costly and time-consuming, especially if robust error analysis is performed to assess the quality of the output [CBC 2004].

1.4.2 Statistics

During the early years of applied satellite remote sensing, the most common approach to change detection relied on statistics, simply because creating classified maps for large areas requires too much computer power. To create change statistics, a sampling strategy is developed whereby small portions of the image are accurately classified. Then, using statistics, estimates for the various cover types are generated for the entire study area. This approach tends to produce more accurate statistics, for a given level of effort, as compared to those generated from a classified map, even though a classified map effectively samples the entire population of pixels. This conclusion is based on the assumption that the accuracy of the land cover change estimates for the small portions (sample sites) are more accurate than the results from a classified image.

The major disadvantage to the statistics-only approach is that there is no mapped output. In this age of spatial analysis, this often rules out the use of the statistics-only approach [CBC 2004].

1.4.3 Image Maps

An often shunned approach to monitoring changes in land cover is a simple visual approach. With this approach, two images from different dates are viewed simultaneously. This can be achieved by overlaying bands from the different dates, displaying the images
side-by-side, or rapidly switching between images acquired at different times using flicker or swipe options offered by many image processing software products. The primary advantage to this approach is that the results are nearly instantaneous. Another advantage is that a better sense of the actual landscape can be because an effectively picture of the landscape can be depicted, rather than a map of discrete categories, as with a classified map [CBC 2004].

In this project, the knowledge-based classification and multi-source information approach is achieved with conventional visual interpretation and maximum likelihood classification for change detection. On screen digitizing for visual interpretation was done for images of years 1995 and 2001. Image characteristics such as tone, texture color, and pattern are translated into land use attributes. The translation process (transfer function) is guided by local knowledge (e.g. land use map, land cover map, or agriculture statistic), which were collected during fieldwork or background studies. Polygons are drawn around features (cleared land, sand excavation, new roads and wetland) and a label was assigned to each shapefile, characterizing it by attributes (the legend) and finally all four new shapefiles are created each for construction, new roads, sand excavation, and cleared area.
2. LiveOakPro Application for Monitoring and Mapping of Live Oak Peninsula

The Monitoring and Mapping of Live Oak Peninsula project extends the use of GIS and computer science to know the changes occurred in LiveOak Peninsula and also to know the potential for the development in those areas. The objective is to develop a GIS system (LiveOakPro) for integrating, processing, viewing and analyzing various data sets concerning Live Oak Peninsula area.

2.1 Monitoring And Mapping Live Oak Peninsula Application

The complete, self-contained standalone application named LiveOakPro has been developed for Mapping and Monitoring of Live Oak Peninsula which has the following basic functionalities:

• Consists of an electronic map for geographic data display, query, and analysis.

• Display and change the results dynamically in a chart and tabular form.

• Identifies the changes in terms of new roads, construction, sand excavation and cleared area.

• Creating the dynamic detailed report for change occurred of any particular area in Live Oak Peninsula.

• Overview window and magnifier to see the aerial imagery in more detail.

• Ability to print maps with scalebar, north arrow and text include them in reports.
3. SYSTEM DESIGN AND RESEARCH

This chapter gives a brief overview of the analysis, design and implementation phases of this project.

The development of LiveOakPro comprises the elaboration and implementation of a suitable domain model and the realization of a graphical user interface with sophisticated graphical facilities, which is user-friendly. Such a development is a demanding enterprise, which consumes a substantial amount of time and money [Pressman 2001]. Figure 3.1 shows the whole application development process, which was followed to design, develop, and implement the project.

![Application Development Process Cycle](image.png)

Figure 3.1 Application Development Process Cycle
3.1 Requirement Definition

In this stage, user requirements were gathered. This was done on the basis of information provided by the user in the form of documents, required user interface & process specs, on-site analysis interviews with end-users, other project research and analysis. This project was the idea of, Dr. Elizabeth Smith, Assistant Director Center for Coastal Studies at Texas A&M University - Corpus Christi. Several meetings were held with her and some other employee’s and technical personnel related with this project. This stage has had the following steps:

1. Requirements Analysis with Business Application Goal and High Level Requirement gathering

2. Creation of Visual Scope Document and Feature List

3. Providing technical recommendations and High Level Requirement Specification

Dr. Smith provided the background and geographic importance about the studied area. She explained the need for this application and the projected scope of this project. She suggested sources for getting information for the design of the database. In her terms, she explained her vision of stand-alone application to access this data. She also explained the targeted audience and an approximate number of different users of the system.

Detailed excerpts of these meetings can be found in the Appendix A. Based on these excerpts and initial requirement analysis, the design of the database schema and user interface were started.
3.1.1 Analysis

In this stage, a detailed analysis was carried out from the information in the vision & scope document and feature list. This stage has had the following steps:

1. Analysis and creation of Software Requirement Specification
2. Creation of Use Case Specification & Diagram and generating the Use Case Model
3. Design business rules and flow diagrams
4. Establishment of Requirement Traceability Matrix
5. Validation of the Scope and estimates against the contract and revisions made if necessary.

3.1.2 Design

In this stage, the application design is developed on the basis of the software requirement specification; user specification, business rules and diagrams and the scope agreed upon in the Requirement Definition stage. This stage has the following steps:

- Based on the initial requirement analysis a project proposal document was prepared. Mrs. Funk-Baxter was shown the document to find out whether the problem was properly addressed or not
- User Interface - The prototype is developed and validated against the requirements and presented to the client for approval
- Researched MapObjects, VisualBasic 6.0 and other ActiveX components from various sources such as books, Web sites and listserves to determine the functionality provided by the languages.
- Studied the features of MapObjects and Visual Basic 6.0 from various sources such as books, Web sites and listserves. During that research, learned about the different
libraries included in MapObject, which proved to be very important to create the application.

- Designed the basic components of the user interface.

### 3.1.3 Development

In this stage, the actual code based on the design was created and tested against design requirements and test cases. This stage had the following steps:

1. The development of code base proceeds as per implementation and design models
2. The application source code is tested according to the test cases and test plan
3. Documentation of processes
4. Creation of User Case Test results

### 3.1.4 Integration and Testing

In this stage, the developed application was tested through test cases, test plans and user acceptance criteria in the deployed environment.

1. Integration and quality testing was carried out resulting in test reports
2. End User acceptance tests carried out
3. End User feedback and debugging
4. End User acceptance

### 3.1.5 Deployment and Acceptance

In this stage, the developed application was deployed on the end user computer and these were the steps followed in the whole process:

1. Integrated Application and Deployment plan
2. Implementation on user Premises
3. Implementation Signoff by user

4. Creation of Archived Software Articrafts

5. Transition of application to the maintenance team

3.2 Database models

The database is the foundation for a geographic information system. Database systems are usually classified according to the data model they use. In the past, there were the hierarchical and network data models, which have now being replaced by three models, namely, relational, object-oriented and object-relational. The following sections briefly describe these three models.

3.2.1 The Relational model

Relational databases are based on the relational model, which organizes the data into relations or tables [Elmasri 2000]. A relational schema is a set of attribute names and mapping from each attribute name to a domain [Worboys 1995]. The relational schema gives the structure of the relation, and does not include the data. A relation is a finite set of tuples associated with a relational schema according to the number of attribute names in the schema. Some properties of a relation include:

- The ordering of the tuples in the relation is not significant
- Tuples in a relation are distinct from each other
- Ordering of the tuples in the relation is not significant
- Columns are ordered so that data items correspond to attribute in the relational schema with which they are labeled. Inherent to relational databases are the notions of primary key and foreign key, which are used to provide links between tuples in different relations
3.2.2. The object-oriented model

Object-oriented database management systems (OODBMS) integrate object orientation and database functionality [Khoshafian, 1993]. An object-oriented database is characterized by having an object-oriented logical data model and by using an object-oriented programming language as its principal interface [Cooper, 1997]. In object-oriented databases, the basic unit of data is an indivisible object.

These objects have identity, attributes and behavior defined through methods, all encapsulated within the object itself. Object-oriented databases were introduced as a means of overcoming the limitations of the relational model and to handle more complex application requirements, which the relational model cannot capture. However, OODBMS have their own limitations. Both the relational and object oriented databases have their advantages and disadvantages.

3.2.3. The object-relational model

The object-relational approach, sometimes called the extended relational model, is a compromise between the concepts of the object-oriented and relational models. In this approach, object-oriented features are incorporated into relational databases or (built on top of it) in a manner that utilizes the power of object orientation while maintaining the full functionality of the relational model [Elmasri 2000]. Currently, many database management systems and GIS systems are beginning to incorporate object-oriented features into the existing relational database functionality. For example, Oracle 8 introduced this concept to overcome the limitations of the relational model. ESRI’s recent model called the geodatabase fully combines object-oriented concepts with the relational model. This model is the one used for implementation in this research. The object-
relational model has been adopted in this research because it utilizes the power and semantics of object orientation and the full functionalities of the underlying relational database system.

3.3 Geodatabases

Traditionally, most GIS systems have stored the spatial and attribute components of geographic data in separate files linked together by unique identifiers. The data model used often isolates the ‘graphics’ data from the attribute data. For example, in the old ArcInfo coverage (georelational) model, the spatial data is stored in indexed binary files, and the attributes are stored in separate files (INFO tables). Recent advances in hardware and software engineering have now made it possible to integrate GIS and database systems to create geodatabases. Geodatabases (short for geographic databases) store both the spatial and attribute data together in a single database management system. The spatial data is stored just like an attribute in the database. This possibility has further been enhanced by the integration of the relational model and object-oriented concepts. This section gives a general introduction to ESRI geodatabase concept in the context of the foregoing discussion.

3.3.1. ESRI Geodatabase Model

With the recent release of their integrated ArcGIS software, ESRI introduced a new object-relational model called the geodatabase. This new model is implemented as an extension to the standard relational model by integrating it with object-oriented concepts in a manner that allows geographic objects to be modeled with their behaviors attached to them. The geodatabase model serves as a generic model for geographic information, and provides a framework for creating an object-relational database schema and implementing
behaviors on objects in the database [ESRI 1999]. The geodatabase supports many object-oriented concepts such as: inheritance, encapsulation, polymorphism, etc [Zeiler 1999]. Entities in the geodatabase are represented as objects with attributes, relationships, and behavior. ArcInfo also provides the ability for objects in the geodatabase to message each other. This provides database developers the ability to build more complex behaviors into objects in the database. The ArcInfo software manages both the attributes and behavior of objects, and database developers work in an object-component environment that is abstracted from the underlying physical database model and augmented by a programming framework of interfaces and methods. ArcInfo automatically provides the object-relational mapping and manages the integrity of the data in the underlying database.

The basic objects in the geodatabase model are shown in Figure 3.2. All the objects in the model are implemented as COM components with attributes and behavior attached to them.
3.3.2. Personal Geodatabases

LiveOakPro is a single user application using the personal geodatabase to store the spatial and attribute components of geographic data. The geodatabase comes in two variants, namely, personal and multi-user geodatabases. The personal geodatabase is implemented on the Microsoft Jet Engine, which stores data in Microsoft Access database, and is built into the ArcInfo software. The personal geodatabase suitable for project-oriented GIS. A Personal geodatabase supports single editing and several simultaneous viewers. For large enterprise databases, ArcInfo provides a multi-user data access extension called the ArcSDE. ArcSDE provides the gateway between the ArcInfo software and the DBMS to share and manage the spatial data as tables. It allows remote access to spatial data and allows many concurrent editing of the same database.

3.4 Programming and Application Development

This section of the project report focuses on translating the design previously described into machine readable code. LiveOakPro needed a map display interface, a data display interface, charting capability, map layout tools, magnifier, overview and report functions. The coding is done using ESRI’s MapObjects, Visual Basic 6.0, and a reporting tool. MapObjects is component based software; developers can create applications that have dynamic, interactive maps that have geographic information system (GIS) capabilities. MapObjects gives developers a powerful collection of mapping components that can be plugged into many standard development environment [Ralston 2001]. With nearly 50 ActiveX® automation objects, Map Objects provides developers with all the tools they
need to build customized mapping and GIS applications [ESRI 2004]. A rapid application
development approach is used to develop and deploy the application on a Windows
platform.

The GIS software application is designed to have the following interfaces:

1. A Live Oak Peninsula Map Viewer interface
2. An Overview interface, which gives the overview of the Live Oak map.
3. A Magnifier interface which can magnify the map to have a more detailed view.
3.4.1 LiveOakPro Map Viewer Interface

Figure 3.3 Map Viewer Interface

The LiveOakPro Map Viewer interface is designed to display the cartographically prepared maps. It is the visual representation of the geographic features in the geodatabase. On the start up form, it loads all the geographic data into the viewer with the other features of the application. The following code loads the shapefiles and imagefiles into the start up window.
3.4.2 Advanced Toolbar

Undocking advanced toolbar is developed which floats about the desktop but function as a part of the application. The toolbar has certain properties that allow users to manipulate the data in order to produce the user interface they want. The toolbar can be displayed or hidden from the user interface. The toolbar functionalities starts from left to right as shown in figure 3.4:

1. Zoom to Full Extent. “Globe”. Click on tool to zoom to full extent of all map layers.

2. Zoom-In with “Magnifying Glass”. “Magnifying glass” with plus sign. Select the tool, place it on the map, depress the mouse button, and drag a square across the area to view closer.

3. Zoom-Out with “Magnifying Glass”. Magnifying glass with minus sign. Select the tool, place it on the map, depress the mouse button, and drag a square across the area to view at a farther range.

4. Pan. Hand. Click on tool, place on map, depress mouse and drag across screen in the direction you wish to move.

5. Fixed Zoom-in. Tool with four arrows pointing in. Click on tool and zoom in.

6. Fixed Zoom-Out. Tool with four arrows pointing out. Click on tool and zoom out

7. Zoom to Previous Extent. Blue arrow pointing backwards. Click on arrow to go to previous view extent.

8. Print. A print logo. Click on print logo to print the map in the current window.
3.4.3 Status bar

Figure 3.5 Status bar

A Status Bar control is a frame that can consist of several panels, which inform the user of the status of an application. The control can hold up to sixteen frames. Additionally, the control has a "simple" style (set with the Style property), which switches from multi-panels to a single panel for special messages.

The Status Bar control can be placed at the top, bottom, or sides of an application. Optionally, the control can "float" within the application's client area.

Status bar has been developed for the notification purpose to alert users that an action or event has occurred, to give the map coordinates to user, current time and date, total area in the window and also the map scale. The code was written in the Click event handler or any of the event handlers to associate application. The messages show in the status bar from left to right are:

- The message in the first panel shows the total area of the map in the current window
- The second panel gives the X coordinate of the mouse
- The Third panel displays the Y coordinate of the mouse
- The Fourth panel shows the map scale
- The Fifth panel displays the current time
- The Sixth panel shows the date
3.4.4 Virtual Magnifying Glass

The Virtual Magnifying Glass enlarges the view of the Map View window. The magnifier's window automatically follows the movement of the mouse. A semi-transparent crosshair in the center of the magnifier window represents the current position of the mouse. The user can move the lens around the screen to view magnification of any map screen area. The Virtual Magnifying Glass can be dragged to different locations on the screen, remain stationary or follow mouse movement and the Virtual Magnifying Glass will always be remain on the top of the Map window. It can magnify the map view by almost 200%.

Figure 3.6 Magnifying Glass

The underlying technology and coding was developed using different windows in built Application Programming Interface (API).
3.4.5 Overview Window

The Overview window features display an additional view of the Live Oak Peninsula with a position indicator that corresponds to the current view inside the Overview Window. When the cursor is moved over the Overview window, the rectangle in red can be seen, which indicates the current view. In this window a rectangle can be drawn around to change the current view. Cursor can be drawn anywhere else in the Overview window and drag a rectangle to create a new view of the LiveOakPro Map.

![Figure 3.7 Overview Window](image)

The coding was done using different windows Application Programming Interface (API) along with MapObject libraries.
3.4.6 Dynamic Chart:

A charting facility is provided in the application which dynamically provides a dynamic pie chart for the changed area in terms of construction, new roads, wetland, and cleared area as shown in Figure 3.8, which gives the chart for the area picked by the user in the map interface windows. It was developed using SimpleChart V2.1 ActiveX control.

![Figure 3.8 Charting Window](image)

3.4.7 Reporting Tool

A Reporting tool is integrated in the application, which can build on-the-fly printable reports for the end user. A simple COM object is integrated within the application.
to create the reporting wizard that creates report for the changed area in the map interface window. The reporting tool pulls the data from the map interface window and creates a report to show results in a well formatted form.

The software application also includes help information such as tool tips for various tool bar controls in the Map Viewer, and About Form that shows information about the application such as its version number and copyright date. Also included is a help system that introduces a user to the software application and presents a quick start-up tutorial for the user.
4. TESTING AND EVALUATION

The testing and evaluation of this software was a very important factor in the success of the project. A major effort was made to test the usability of the system and insure security. Also, during each step of the development process, the system was verified for its actual purpose and final results.

The three basic functions of software testing, verification, validation, and debugging were adhered to throughout the development cycle. Software testing concerned with exercising and observing product behavior. The verification attribute specifies an objective and quantitative strategy for assessing whether a requirement was satisfied. Typical values included code inspection, unit test, and system test. Validation was done at the end of the development stage to confirm that the product meets the user needs and requirements.

During software debugging the coding defects were removed from the application.

The testing and evaluation of this system was a very important factor in the success of the system. A major effort was made to test the usability of the system and insure security. Also, during each step of the development process, the system was verified for its actual purpose and final results.

4.1 Usability Testing

The system was designed so that it is simple and friendly to use. This project was developed for The Center for Coastal Studies. Once the pilot module was developed it was shown to users in the The Center for Coastal Studies department of our University.

Feedback was taken from the users regarding the various interfaces of this system. For example. A Graduate Research Student suggested if an overview window was provided to
the user so that user may know what part of map he/she is at then I provided the overview window and also Dr. Smith asked me if some reporting tool would be provided in the application then it would solve lot of her hassle to copy and paste maps and pie chart from the application and create a report for future records. Based on the feedback, changes were made to the system to satisfy maximum users of this system.

4.2 Component Testing

Component testing was done to uncover software errors and validate the quality of software components at the unit level. The main objective was to validate and confirm the quality of a software component by checking its functions, behaviors, and performance to make sure a module under test meets its functional requirements and specified design in a given operational environment.

The main objectives of component testing were to:

- Design high-quality unit tests to uncover as many errors as possible
- Conduct unit tests to demonstrate its functions and behaviors matching the given requirements and design [Gao 2003].

A component was tested to validate the area calculated by the developed application. The area was calculated in ArcView 8.3 and was checked with the results in LiveOakPro and both the results came out to be the same.
5. CONCLUSION AND FUTURE WORK

LiveOakPro is a successful attempt to map and monitor Live Oak Peninsula through computer science and GIS technologies. LiveOakPro not only proved to be very useful for readily accessible data, integration of GIS data, display of change detection results, and better data control, but an easier interface for non-GIS users and proved to be very successful for the agencies that funded this project to monitor this area.

The current limitation in LiveOakPro is that it cannot be used on different datasets so in future it can be made a generic application to monitor any area using any kind of data set and efforts can also be made to make it a Web based application work around ArcIMS so that it can be a part of an organization's internet and intranet platform strategy, acting as a plug-in to a browser, extending its reach throughout the organization, even to palmtops in the field.
ACKNOWLEDGEMENTS

My sincere thanks and appreciation goes to all who contributed in various capacities to make this project a success.

I am greatly indebted to Dr. Elizabeth Smith for being one of my advisors and for granting me the opportunity to do GIS related research in the coastal and marine sciences and for the support she has extended to me throughout my work towards this project, and for her informed guidance and advice. Her positive outlook and confidence in my research inspired me and gave me the confidence.

Special thanks go to my project chair, Dr. Dulal Kar for his support and encouragement during this project. At many stages in the course of this research project I benefited from his advice, particularly so when exploring new ideas. His careful editing contributed enormously to the production of this report.

I would like to express my thanks to one of my committee, Dr. Mario Garcia who directed me at every stage of the project until its completion and his help in writing the statement of my research agenda for the Masters Program.

I would also like to express my thanks to Gail Sutton, Director of Finance and Administration (HRI) for her support, encouragement and to have confidence in me as well as recommending me to Dr. Smith.

Many thanks to the Department, Center for Coastal Studies at Texas A&M University Corpus Christi for providing me the entire computing environment needed for this project.

Finally, thanks to my family and friends on both continents who helped me in ways unknown to them. Their encouragement is deeply appreciated.
BIBLIOGRAPHY AND REFERENCES


Appendix A: Sample Codes

A.1. Code for LiveOak Map Viewer

A.2. Code for LiveOak Magnifier
A.1. Code for Live Oak Map Viewer

Private Sub Form_Load()

legMapView.Visible = False

sbMapView.Visible = False

Set sym = New MapObjects2.Symbol

With sym

.SymbolType = moFillSymbol

.Style = moTransparentFill

.OutlineColor = moRed

.Size = 4

End With

'Code Adding Imagefiles Begins

Dim ilyr As New MapObjects2.ImageLayer

ilyr.file = App.Path & "\2797072a.sid"

Map1.Layers.Add ilyr

'Set ilyr = Nothing

Dim ilyr1 As New MapObjects2.ImageLayer

ilyr1.file = App.Path & "\2797074a.sid"

Map1.Layers.Add ilyr1

'Set ilyr = Nothing

Dim ilyr2 As New MapObjects2.ImageLayer

ilyr2.file = App.Path & "\2797083a.sid"

Map1.Layers.Add ilyr2
Dim ilyr3 As New MapObjects2.ImageLayer
ilyr3.file = App.Path & "\2797151a.sid"
Map1.Layers.Add ilyr3

Dim ilyr4 As New MapObjects2.ImageLayer
ilyr4.file = App.Path & "\2797081a.sid"
Map1.Layers.Add ilyr4

Dim ilyr5 As New MapObjects2.ImageLayer
ilyr5.file = App.Path & "\2797082a.sid"
Map1.Layers.Add ilyr5

'Load data

Dim ilyr6 As New MapObjects2.ImageLayer
ilyr6.file = App.Path & "\2897642a.sid"
Map1.Layers.Add ilyr6

Map1.Refresh

Dim ilyr7 As New MapObjects2.ImageLayer
ilyr7.file = App.Path & "\2897643a.sid"
Map1.Layers.Add ilyr7

Map1.Refresh

Dim ilyr8 As New MapObjects2.ImageLayer
ilyr8.file = App.Path & "\2897634a.sid"
Map1.Layers.Add ilyr8

Dim ilyr9 As New MapObjects2.ImageLayer
ilyr9.file = App.Path & "\2797071a.sid"
Map1.Layers.Add ilyr9

Dim ilyr10 As New MapObjects2.ImageLayer
ilyr10.file = App.Path & "\2797073a.sid"
Map1.Layers.Add ilyr10

Dim ilyr11 As New MapObjects2.ImageLayer
ilyr11.file = App.Path & "\2797152a.sid"
Map1.Layers.Add ilyr11

Dim ilyr12 As New MapObjects2.ImageLayer
ilyr12.file = App.Path & "\2897641a.sid"
Map1.Layers.Add ilyr12

Dim ilyr13 As New MapObjects2.ImageLayer
ilyr13.file = App.Path & "\2897644a.sid"
Map1.Layers.Add ilyr13

'Code Adding Imagefiles Ends

'Code Adding Shapefiles begins

mouseclick = True
dc.Database = App.Path
dc.Connect

Set mlyr1.GeoDataset = dc.FindGeoDataset("Cleared")
    mlyr1.Symbol.Color = moGreen
Map1.Layers.Add mlyr1

Set mlyr2.GeoDataset = dc.FindGeoDataset("NewRoad")
mlyr2.Symbol.Color = moRed
Map1.Layers.Add mlyr2
Set mlyr3.GeoDataset = dc.FindGeoDataset("Construction")
mlyr3.Symbol.Color = moPurple
Map1.Layers.Add mlyr3
Set mlyr4.GeoDataset = dc.FindGeoDataset("Sandexcavation")
mlyr4.Symbol.Color = moTeal
Map1.Layers.Add mlyr4
‘Code Adding shapefiles Ends
'Code MapScale
Dim mapscale As String
dMapScale = Map1.Extent.Width * DEGREES2INCHES / (Map1.Width / 1440)
If Form1.ScaleMode = 1 Then
mapscale = CStr (Format (dMapScale, ",#, 000"))
Else
mapscale = "ScaleMode not Twips"
End If
Text1.text = "Map Scale 1:" & mapscale
'Code Mapscale End
Dim i As Integer
Dim fntScalebar As New stdole.StdFont
fntScalebar.Name = "Arial Narrow"
fntScalebar.Size = 10
With sbMapView
AdjustForLatitude = True
BackColor = moWhite
BarColor1 = &H808080 ‘medium gray
BarColor2 = &HFFC0C0 ‘light blue
BarWidth = 4
BorderStyle = vbBSSolid
Set .Font = fntScalebar
.MapUnits = ScaleBar.sbMapUnits.muDecimalDegrees
.MinTicSpace = 0.6
.ScaleBarUnits = ScaleBar.sbScaleBarUnits.suMeters
End With
ReDim lColor(Map1.Layers.Count - 1) As Double
For i = 0 To UBound(lColor)
If TypeOf Map1.Layers(i) Is MapLayer Then
lColor(i) = Map1.Layers(i).Symbol.Color
End If
Next i
'Initialize Zoom Previous rectangle
Set rectPrevious = Map1.FullExtent
Set Map1.Extent = Map1.FullExtent
'Setup label caption
Form2.Show
End Sub
A.2. Code for LiveOak Magnifier

This code loads the magnifier, which magnifies the view in the Map window by almost two hundred percent and shows the area in more detail to the viewer.

'Module Name:  frmMagnifier

'Description:  Magnifier form

'Requires:  (nothing)

'Methods:  SetFormAndMap - sets underlying form and map

'Update - updates the internal snapshot of the underlying map and
draws on the magnifier whatever is currently underneath the magnifier

'StayOnTop - sets the "always on top" mode for this form

Option Explicit

'== Windows API calls and constants

'-- window position and state

Private Declare Function SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal hWndInsertAfter As Long, ByVal X As Long, ByVal Y As Long, ByVal cx As Long, ByVal cy As Long, ByVal wFlags As Long) As Long

Private Const HWND_TOPMOST = -1
Private Const HWND_NOTOPMOST = -2

'-- device contexts

Private Declare Function GetDC Lib "user32" (ByVal hWnd As Long) As Long

Private Declare Function CreateCompatibleDC Lib "gdi32" (ByVal hDC As Long) As Long

Private Declare Function ReleaseDC Lib "user32" (ByVal hWnd As Long, ByVal hDC As Long) As Long

Private Const HWND_TOPMOST = -1
Private Const HWND_NOTOPMOST = -2
Private Declare Function DeleteDC Lib "gdi32" (ByVal hDC As Long) As Long

Private Declare Function SelectObject Lib "gdi32" (ByVal hDC As Long, ByVal hObject As Long) As Long

Private Declare Function DeleteObject Lib "gdi32" (ByVal hObject As Long) As Long

' -- Bit Map Manipulation

Private Declare Function CreateCompatibleBitmap Lib "gdi32" (ByVal hDC As Long, ByVal nWidth As Long, ByVal nHeight As Long) As Long

Private Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal dwRop As Long) As Long

' -- Drawing

Private Declare Function CreatePen Lib "gdi32" (ByVal nPenStyle As Long, ByVal nWidth As Long, ByVal crColor As Long) As Long

Private Declare Function CreateSolidBrush Lib "gdi32" (ByVal crColor As Long) As Long

Private Declare Function MoveToEx Lib "gdi32" (ByVal hDC As Long, ByVal X As Long, ByVal Y As Long, lpPoint As POINTAPI) As Long

Private Declare Function LineTo Lib "gdi32" (ByVal hDC As Long, ByVal X As Long, ByVal Y As Long) As Long

Private Declare Function Rectangle Lib "gdi32" (ByVal hDC As Long, ByVal X1 As Long, ByVal Y1 As Long, ByVal X2 As Long, ByVal Y2 As Long) As Long

Private Declare Function GetROP2 Lib "gdi32" (ByVal hDC As Long) As Long

Private Declare Function SetROP2 Lib "gdi32" (ByVal hDC As Long, ByVal nDrawMode As Long) As Long

Private Type POINTAPI

X As Long
Y As Long
End Type

'== Module Variables
Dim mForm As Form
Dim mMap As MapObjects2.Map
Dim mWidth As Long, mHeight As Long
Dim mMagnification As Double
Dim mDC As Long           ' device context for the snapshot
Dim mBitmap As Long       ' bitmap snapshot of the underlying map
Dim mOldBitmap As Long

Private Sub Form_Load ()
StayOnTop True
mMagnification = 4
End Sub

Private Sub Form_Unload (Cancel As Integer)
'-- free the memory used by the snapshot and its device context
If mOldBitmap > 0 Then
SelectObject mDC, mOldBitmap
DeleteObject mBitmap
DeleteDC mDC
End If
End Sub
Private Sub mapMagnify_MouseDown (Button As Integer, Shift As Integer, X As Single, Y As Single)

If Button = 1 Then

    MoveMagnifier X, Y

End If

End Sub

Private Sub mapMagnify_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

If Button = 1 Then

    MoveMagnifier X, Y

End If

End Sub

Private Sub mapMagnify_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As Single)

DrawMap

End Sub

Private Sub DrawMap()

Dim sx As Single, sy As Single

Dim p As MapObjects2.Point, e As MapObjects2.Rectangle

'-- draw the magnified map; determine the screen location of mapMagnify

'-- relative to the underlying map

Set e = New MapObjects2.Rectangle

sx = (frmMagnifier.Left + mapMagnify.Left) - (mForm.Left + mMap.Left) - (3 * Screen.TwipsPerPixelX)

sy = (frmMagnifier.Top + mapMagnify.Top) - (mForm.Top + mMap.Top) - (6 * Screen.TwipsPerPixelY)
'-- get the map extent from the underlying map based on the pixel extent
'-- of mapMagnify
Set p = mMap.ToMapPoint (sx, sy)
e.Left = p.X
e.Top = p.Y
Set p = mMap.ToMapPoint (sx + mapMagnify.Width, sy + mapMagnify.Height)
e.Right = p.X
e.Bottom = p.Y
'-- scale mapMagnify to the set magnification and display
e.ScaleRectangle 1 / mMagnification
mapMagnify.Extent = e
End Sub

Private Sub Form_Resize()
    ResizeFormFor Me
End Sub

Private Sub MoveMagnifier(X As Single, Y As Single)
    Dim mapDC As Long, tempDC As Long, bitmap As Long, oldBitmap As Long
    Dim dx As Single, dy As Single
    Dim sx As Long, sy As Long, w As Long, h As Long
    Dim tx As Long, ty As Long
    Dim mag As Single
    Dim oldBrush As Long, hndBrush As Long
    Dim oldPen As Long, hndPen As Long
    Dim lastPoint As POINTAPI

Dim oldDrawMode As Long

'-- if the specified coordinates do not represent the center of mapMagnify, move
'-- the form to recenter

dx = X - (mapMagnify.Width / 2)
dy = Y - (mapMagnify.Height / 2)

If dx <> 0 Or dy <> 0 Then
    Me.Move Me.Left + dx, Me.Top + dy
End If

'-- determine the screen location of mapMagnify relative to the underlying map

tx = Screen.TwipsPerPixelX

ty = Screen.TwipsPerPixelY

sx = (((frmMagnifier.Left + mapMagnify.Left) - (mForm.Left + mMap.Left)) \ tx) - 3

sy = (((frmMagnifier.Top + mapMagnify.Top) - (mForm.Top + mMap.Top)) \ ty) - 6

w = (mapMagnify.Width / tx) - 2

h = (mapMagnify.Height / ty) - 2

mapDC = GetDC(mapMagnify.hWnd)

'-- if mapMagnify goes beyond the edge of the underlying map

If sx < 0 Or sy < 0 Or sx + w > mWidth Or sy + h > mHeight Then

    '-- create a temporary device context and bitmap that is the same pixel size

    '-- as mapMagnify

    tempDC = CreateCompatibleDC (mapDC)

    bitmap = CreateCompatibleBitmap (mapDC, mapMagnify.Width / tx, mapMagnify.Height / ty)

    oldBitmap = SelectObject(tempDC, bitmap)
'-- paint the temporary bitmap a medium gray

hndBrush = CreateSolidBrush(RGB(128, 128, 128))

oldBrush = SelectObject(tempDC, hndBrush)

Rectangle tempDC, -2, -2, mapMagnify.Width / tx + 2, mapMagnify.Height / ty + 2

SelectObject mapDC, oldBrush

DeleteObject hndBrush

'-- copy the underlying map graphics from the snapshot to the temporary bitmap,

'-- then copy the temporary bitmap to mapMagnify's device context; this entire

'-- process avoids flicker

BitBlt tempDC, 0, 0, w, h, mDC, sx, sy, vbSrcCopy

BitBlt mapDC, 0, 0, w, h, tempDC, 0, 0, vbSrcCopy

'-- otherwise, copy the underlying map graphics directly from the snapshot to

'-- mapMagnify's device context

Else

BitBlt mapDC, 0, 0, w, h, mDC, sx, sy, vbSrcCopy

End If

'-- draw the outline of the area to magnify

hndPen = CreatePen(0, 1, RGB(0, 0, 0))

oldPen = SelectObject(mapDC, hndPen)

oldDrawMode = GetROP2(mapDC)

SetROP2 mapDC, vbInvert

mag = mMagnification * 2

MoveToEx mapDC, (w / 2) - (w / mag), (h / 2) - (h / mag), lastPoint
LineTo mapDC, (w / 2) + (w / mag), (h / 2) - (h / mag)
MoveToEx mapDC, (w / 2) + (w / mag), (h / 2) - (h / mag), lastPoint
LineTo mapDC, (w / 2) + (w / mag), (h / 2) + (h / mag)
MoveToEx mapDC, (w / 2) + (w / mag), (h / 2) + (h / mag), lastPoint
LineTo mapDC, (w / 2) - (w / mag), (h / 2) + (h / mag)
MoveToEx mapDC, (w / 2) - (w / mag), (h / 2) + (h / mag), lastPoint
LineTo mapDC, (w / 2) - (w / mag), (h / 2) - (h / mag)

'-- reset device contexts and free memory
If bitmap > 0 Then
  SelectObject tempDC, oldBitmap
  DeleteObject bitmap
  DeleteDC tempDC
End If
SelectObject mapDC, oldPen
DeleteObject hndPen
SetROP2 mapDC, oldDrawMode
ReleaseDC mapMagnify hWnd, mapDC
End Sub

Private Sub UpdateBitmap ()
  Dim baseMapDC As Long

  '-- create a new snapshot of the underlying map
  mWidth = (mMap.Width / Screen.TwipsPerPixelX) - 6
  mHeight = (mMap.Height / Screen.TwipsPerPixelY) - 7
baseMapDC = GetDC(mMap.hWnd)

mDC = CreateCompatibleDC(baseMapDC)

mBitmap = CreateCompatibleBitmap(baseMapDC, mWidth, mHeight)

mOldBitmap = SelectObject(mDC, mBitmap)

BitBlt mDC, 0, 0, mWidth, mHeight, baseMapDC, 0, 0, vbSrcCopy

ReleaseDC mMap.hWnd, baseMapDC

End Sub

Public Sub SetFormAndMap(f As Form, m As MapObjects2.Map)

Dim e As MapObjects2.Rectangle, i As Integer

'-- set underlying form and map

Set mForm = f

Set mMap = m

For i = mMap.Layers.Count - 1 To 0 Step -1
    mapMagnify.Layers.Add mMap.Layers(i)

Next i

Set e = mMap.FullExtent

e.ScaleRectangle 3

mapMagnify.FullExtent = e

UpdateBitmap

DrawMap

End Sub

Public Sub Update()

'-- update the snapshot of the underlying map and draw whatever is now
'-- underneath the magnifier
UpdateBitmap
DrawMap
End Sub

Public Sub StayOnTop (onTop As Boolean)
Dim fLeft As Long, fTop As Long, fWidth As Long, fHeight As Long
Dim fState As Long
'-- set the "always on top" mode for this form
fLeft = Me.Left / Screen.TwipsPerPixelX
fTop = Me.Top / Screen.TwipsPerPixelY
fWidth = Me.Width / Screen.TwipsPerPixelX
fHeight = Me.Height / Screen.TwipsPerPixelY
If onTop Then
  fState = HWND_TOPMOST
Else
  fState = HWND_NOTOPMOST
End If
SetWindowPos Me.hWnd, fState, fLeft, fTop, fWidth, fHeight, 0
End Sub
Appendix B: Requirement Analysis Meetings

- Meeting with Dr. Elizabeth Smith

  Dr. Smith is the Associate Director for Research and Education, as well as a Research Scientist with Center for Coastal Studies, and an Adjunct Professor of Biology, Texas A&M University-Corpus Christi. Her areas of expertise are Wetland Ecology, Ornithology, Mammalogy, and Wetlands Vegetation. Following are the excerpts from the interview:

  1. Historical and geographic importance of Live Oak Peninsula.
  2. The purpose and importance of the study.
  3. Background information about the existing change detection methodologies such as traditional post-classification cross-tabulation and cross-correlation analysis.
  4. Suggested the sources to obtain the geographic data.
  5. She gave a general feeling of how the whole application might look. This was just an idea that had helped me design the system.
  6. The platform and languages that will be used were also discussed. Initially the idea suggested was to build a web-based system using ArcIMS and ASP.net but finally we decided to go for a standalone application using visual basic 6.0 and MapObjects because of the unavailability of hardware and software resources.