Steganography Tool Project Report

Introduction

Steganography is the art of hiding secret information within a cover medium, such as images, audio, or text, so that the existence of the message is concealed. Unlike encryption, which makes data unreadable, steganography hides the presence of the data itself. This project focuses on developing a GUI-based image steganography tool that allows users to securely hide and extract messages in images, using the Least Significant Bit (LSB) method.

Abstract

The objective of this project is to create a user-friendly and interactive tool for embedding and extracting hidden text messages in images. Users can encode sensitive information within PNG or BMP images and decode messages from previously encoded images. The tool ensures data integrity, ease of use, and prevents unwanted garbage during decoding, making it suitable for practical applications in secure communications.

Tools Used

- Python 3.x – Programming language used for logic and GUI development.
- Tkinter – For building the interactive graphical user interface (GUI).
- PIL (Pillow) – To handle image reading, writing, and pixel manipulation.
- String Module – For handling printable characters and preventing garbage in decoding.
- Operating System – Any platform supporting Python (Windows, Linux, macOS).
-

Steps Involved in Building the Project

1. Requirement Analysis

Defined the objectives: encode messages in images and decode them reliably.

Decided on supporting PNG and BMP images.

2. Designing the GUI

Used Tkinter for GUI elements: labels, text boxes, buttons, and frames.

Added features such as Browse button, Encode/Decode buttons, radio buttons to select decoding method.

Implemented a modern color theme and hover effects for better user experience.

3. Implementing Steganography Logic

Encoding: Converted the secret message to binary and embedded it in the least significant bits of RGB pixels. A delimiter ####END#### is used to mark the end of the message.

Decoding (New Images): Extracted binary data from the image pixels until the delimiter is found, converting it back to readable text.

Decoding (Existing Images): Extracted only printable ASCII characters to avoid garbage values for images without delimiters.

4. Testing and Validation

Verified encoding and decoding on multiple images.

Checked that hidden messages are invisible to normal viewers and correctly extracted without garbage.

Ensured GUI responsiveness and color scheme consistency.

5. Final Touches

Added features for enhanced GUI: colored frames, message box highlighting, modern buttons with hover effects.

Ensured cross-platform compatibility and user-friendly workflow.

Conclusion

The Steganography Tool project successfully demonstrates how secret messages can be securely hidden and retrieved from images. The GUI provides a simple and intuitive interface for users, while the backend ensures accurate encoding and decoding using the LSB method. By combining security and usability, this tool serves as a practical example of applied steganography, suitable for learning and real-world applications in secure communication.