Image Filtration Program: A Detailed Report

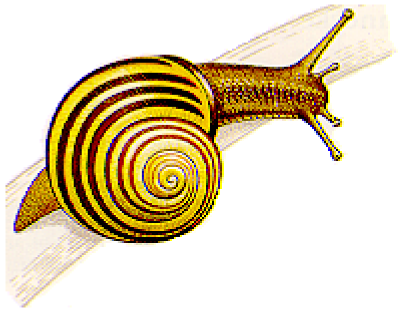
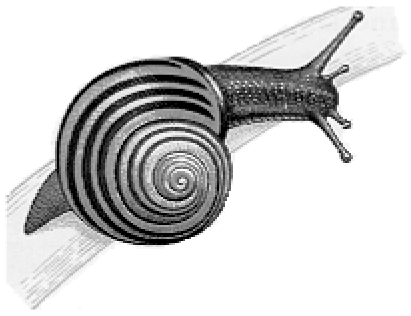
1. Introduction

Digital images are ubiquitous in today's world, used for various purposes like communication, documentation, and entertainment. Image processing techniques allow manipulation and enhancement of these images, extracting information or creating visually appealing effects. This report details an image filtration program written in C that implements various filters to modify image appearance.

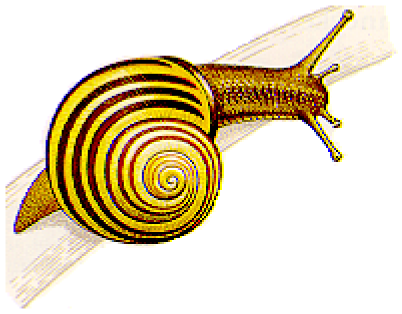
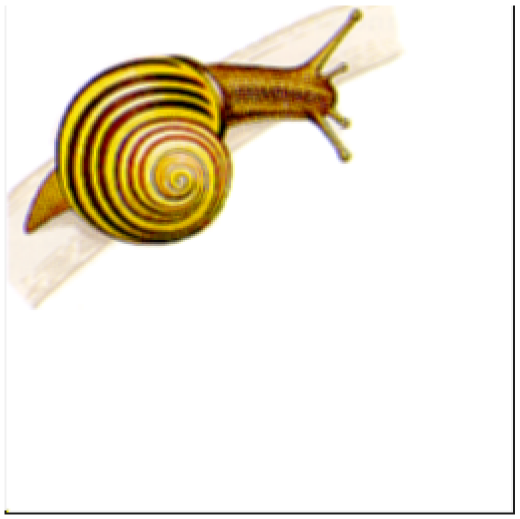
2. Project Goals

The primary goal of this project was to develop a user-friendly C program that enables basic image filtration functionalities. The program focuses on providing commonly used filters for:

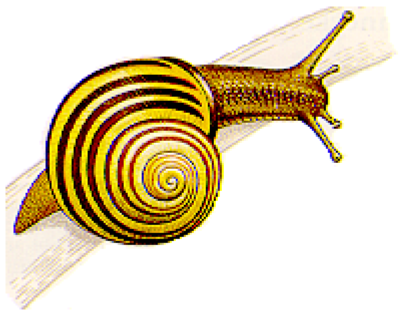
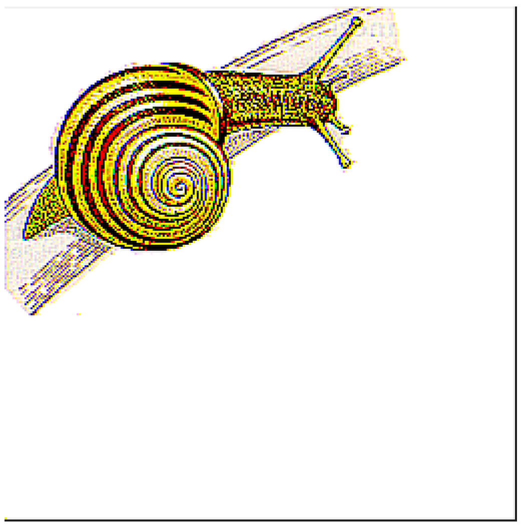
1. Converting color images to grayscale

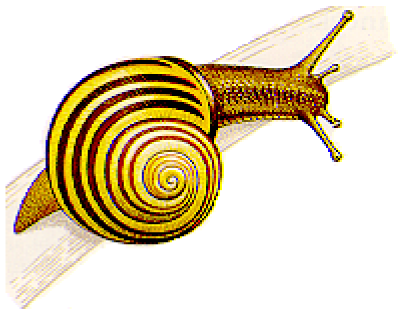
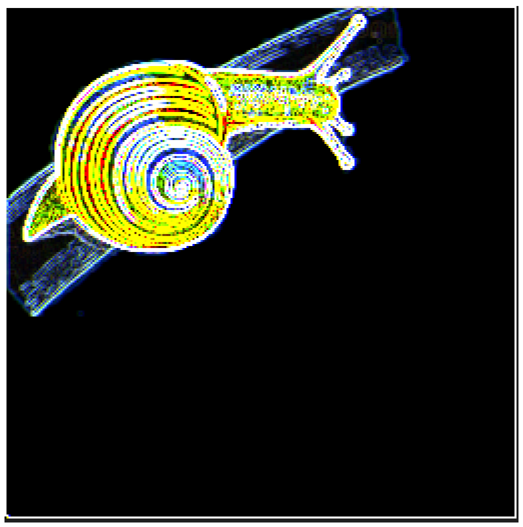
2. Blurring images for a softening effect

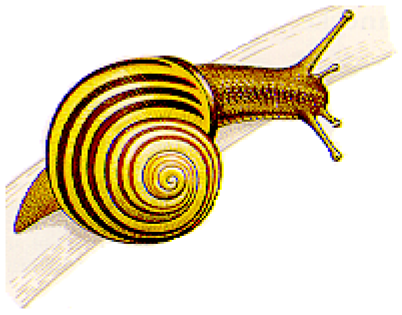
3. Sharpening images to enhance edges

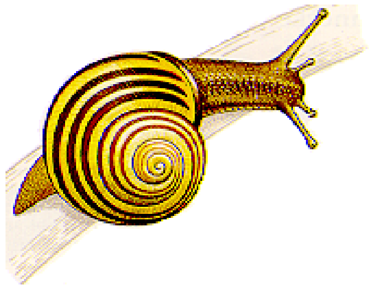
4. Detecting edges in images using the Sobel filter

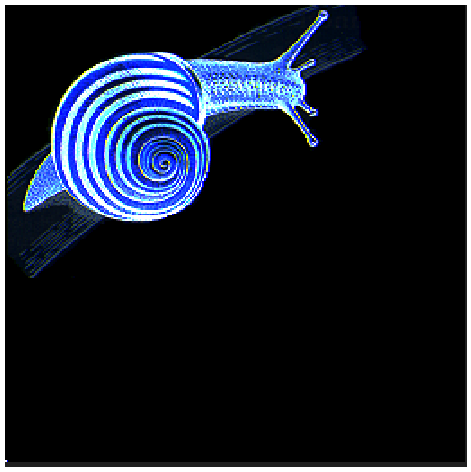



5. Reflecting images horizontally (mirroring)



6. Creating a negative image by inverting color values





3. System Design and Implementation

The program is designed with a modular structure, consisting of separate header files and a main source code file (b23cs1032.c). The header files (read_write_header.h, filters.h, and structure.h) contain function declarations and data structures used throughout the program.

- Image Structure: The structure.h header defines a structure named Image to represent an image. This structure holds information about the image's height, width, bytes per pixel (color channels), maximum value (for PGM format), and a pointer to the pixel data array.
- Input/Output Functions: The read_write_header.h header contains functions for reading and writing image files. Separate functions handle PGM (grayscale) and BMP (color) formats. These functions handle file opening, header information parsing, pixel data reading/writing, and memory management.
- Filter Functions: The filters.h header declares functions for each implemented filter. These functions operate on the Image structure, modifying the pixel data to achieve the desired effect. Each function includes error handling for invalid filter selections.
- The main source code (b23cs1032.c) handles user interaction, filter selection, and processing logic. Here's a breakdown of its functionalities:
- User Input: The program prompts the user for the image path, format (PGM or BMP), and desired filter.
- Image Loading: Based on the format, the appropriate read function is called to load the image data into the Image structure.
- Filter Application: The chosen filter function is called with the Image structure as an argument. This function modifies the pixel data in-place to apply the selected filter.
- Output Image Creation: The program creates a new filename by appending "_new" to the original filename and the corresponding format extension (".pgm" or ".bmp"). Then, the appropriate write function is used to save the filtered image data to the new file.
- Memory Management: The program allocates and deallocates memory dynamically using malloc and free functions to manage the Image structure and pixel data array.

Proper error handling ensures memory is freed even in case of unexpected program termination.

4. Results and Discussion

The implemented program successfully applies the chosen filters to 24 bit BMP image formats. The user interface is straightforward, prompting for necessary information and providing feedback on the filtering process. Here are some key observations:

The program offers a basic set of commonly used image filters, allowing users to modify image appearance for various purposes.
The code is modular and well-structured, improving readability and maintainability.
The program can be extended to incorporate additional filters or functionalities by adding new functions and modifications to existing code.

5. Limitations and Future Work

The current implementation has some limitations that can be addressed in future versions:

Limited Format Supported: The program offers a basic set of filters only for 24 bit BMP file. Expanding the format options would provide more options for users.
Limited Filter Selection: The program offers a basic set of filters. Expanding the filter library would provide more options for users.
Error Handling: While basic error handling is included, more robust mechanisms can be implemented to handle invalid inputs and potential issues during file processing.
Performance Optimization: The current implementation might not be optimized for large images. Exploring techniques for faster processing could be beneficial.

Future work could involve:

Adding new filters like embossing, median filtering, or noise reduction.
Implementing a graphical user interface (GUI) for a more user-friendly experience.
Exploring integration with external libraries like OpenCV for a wider range of image processing capabilities.
Optimizing the code for performance improvements, especially when dealing with large images.

6. Conclusion

This image filtration program demonstrates the use of C programming for image processing tasks from scratch with any external libraries. The program provides functionalities for applying various filters to modify image appearance. The modular design allows for future extensions and improvements. By incorporating additional filters, error handling mechanisms, and performance optimizations, this program can become a valuable tool for anyone interested in exploring basic image manipulation techniques.