Gerardo Daniel Naranjo Gallegos, A01209499

Robotic Vision

Assignment 03

**Geometric Transformations of images in ANSI C.**

Contenido

[Background: 2](#_Toc1586040)

[Objective: 2](#_Toc1586041)

[Upload the following: 2](#_Toc1586042)

[Solution: 3](#_Toc1586043)

[Flip Horizontal: 3](#_Toc1586044)

[Flip Vertical: 4](#_Toc1586045)

[Rotate 90 degrees to the left: 5](#_Toc1586046)

[Rotate 90 degrees to the right: 6](#_Toc1586047)

[Rotate 180 degrees: 7](#_Toc1586048)

# Background:

To have a basic understanding of how images are handled and manipulated, ANSI C provides an excellent programming language. Geometric transformation is the manipulation of images in order to rotate, mirror, scale down, scale of, and flip them.

# Objective:

Using LenaGrey 512x512.pgm image file we have been using in class and the template I gave you, write 5 different C programs that do the following image geometric transformations:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Original image | Flip horizontal | Flip vertical |
|  |  |  |
| Rotate left | Rotate right | Rotate 180 |
| Image geometrics transformations to perform | | |

# Upload the following:

* Upload to the 5 programs written in C.
* Upload your 5 manipulated images as evidence.
* Include a report, written in word, which explains how your algorithm works for each case. This could be done using a flow chart, block diagram or any other representation.

# Solution:

## Flip Horizontal:

For this part I use the following code inside the USER DEFINED section:

// Declare local variables:

unsigned char Matrix[NCols][MRows];

int Column, Row;

// Fill the Matrix with the input image, using for:

for ( Row = 0; Row < MRows; Row++ ){

for ( Column = 0; Column < NCols; Column++ ){

Matrix[Column][Row] = fgetc( infptr );

}

}

// Flip horizontal and Save output image:

for ( Row = 0; Row < MRows; Row++ ){

for ( Column = 0; Column < NCols; Column++ ){

fputc( Matrix[(NCols-1)-Column][Row], outfptr );

}

}

The resulting image is:



## Flip Vertical:

For this part I use the following code inside the USER DEFINED section:

// Define the local variables:

unsigned char Matrix[NCols][MRows];

int Column, Row;

// Import the original image into Matrix:

for( Row = 0; Row < MRows; Row++ ){

for( Column = 0; Column < NCols; Column++ ){

Matrix[Column][Row] = fgetc( infptr );

}

}

// Flip vertically the image and save it in output image:

for( Row = 0; Row < MRows; Row++ ){

for( Column = 0; Column < NCols; Column++ ){

fputc( Matrix[Column][(MRows-1)-Row], outfptr );

}

}

The resulting image is:



## Rotate 90 degrees to the left:

For this part I use the following code inside the USER DEFINED section:

// Define the local variables:

unsigned char Matrix[NCols][MRows];

int Column, Row;

// Import the original image an store it in Matrix:

for( Column = 0; Column < NCols; Column++ ){

for( Row = 0; Row < MRows; Row++ ){

Matrix[Column][Row] = fgetc( infptr );

}

}

// Rotate 90 degrees to the left and save the output image:

for( Column = 0; Column < NCols; Column++ ){

for ( Row = 0; Row < MRows; Row++ ){

fputc( Matrix[Row][(NCols-1)-Column], outfptr );

}

}

The resulting image is:



## Rotate 90 degrees to the right:

For this part I use the following code inside the USER DEFINED section:

// Define the local variables:

unsigned char Matrix[NCols][MRows];

int Column, Row;

// Import the original image in Matrix:

for( Column = 0; Column < NCols; Column++ ){

for( Row = 0; Row < MRows; Row++ ){

Matrix[Column][Row] = fgetc( infptr );

}

}

// Rotate 90 degrees to the right & save the output image:

for( Column = 0; Column < NCols; Column++ ){

for( Row = 0; Row < MRows; Row++ ){

fputc( Matrix[(MRows-1)-Row][Column], outfptr );

}

}

The resulting image is:



## Rotate 180 degrees:

For this part I use the following code inside the USER DEFINED section:

// Define the local variables:

unsigned char Matrix[NCols][MRows];

int Column, Row;

// Import the original image in Matrix:

for( Column = 0; Column < NCols; Column++ ){

for( Row = 0; Row < MRows; Row++ ){

Matrix[Column][Row] = fgetc( infptr );

}

}

// Rotate 180 degrees and save the output image:

for( Column = 0; Column < NCols; Column++ ){

for( Row = 0; Row < MRows; Row++ ){

fputc(Matrix[(NCols-1)-Column][(MRows-1)-Row],outfptr);

}

}

The resulting image is:

