Javeriana Cali University

Computer and Systems Engineering

Exam 2

Parallel Programming

Authors:
Andrés Felipe Delgado and Ana Maria García

April 22 2020

1 Introduction

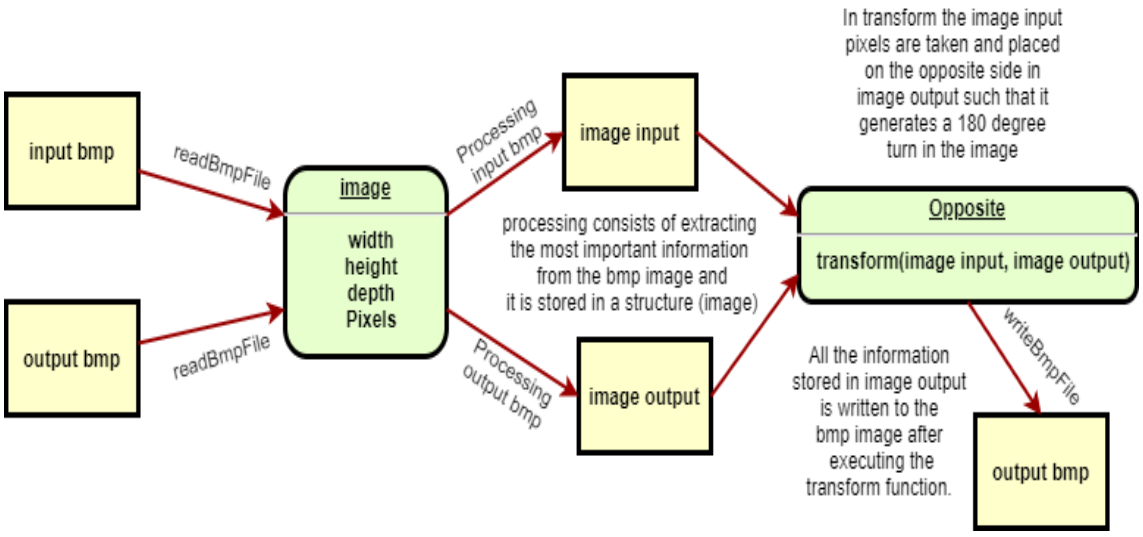
In the present part, a github repository was taken from a Chinese student, this repository belongs to an image analysis and processing course. The project basically consists of creating a program in C languaje that can process images in bmp format and then, with the information extracted from this image, modify it by playing mathematically with the pixels, thus generating deformations of many types.

Among all the transformations that are part of the code we select the one that rotates the original image 180 degrees. Next, a short description of the problem will be presented, then graphically the sequential solution to the problem will be shown to finally present the parallelization made using code threads, the results obtained and conclusions.

2 Description of the problem

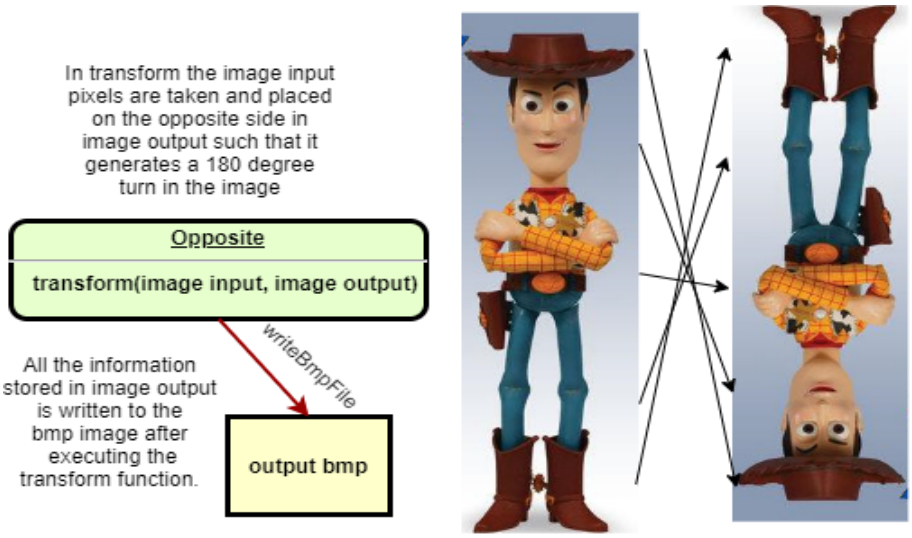
Given an input image with bmp extension and an output image with bmp extension, this last being a copy of the input image, the program analyzes the input image so that it obtains and stores each of its pixels to later invert the image making a 180 degree turn. The result will be saved in the same output image bmp file entered at the start of the program.

2.1 Sequential Solution



In the previous figure you can see graphically the operation of the sequential solution to the general problem, for this reason it includes the way in which

two bmp files are processed, one of which is going to obtain its very important properties such as its height, width, depth and the matrix of pixels that make it up, the second bmp is where the pixel matrix of the first one will be written, but in an opposite way to generate the effect of a 180 degree turn. In the following graph you can see how the pixels of one image are copied in the second.

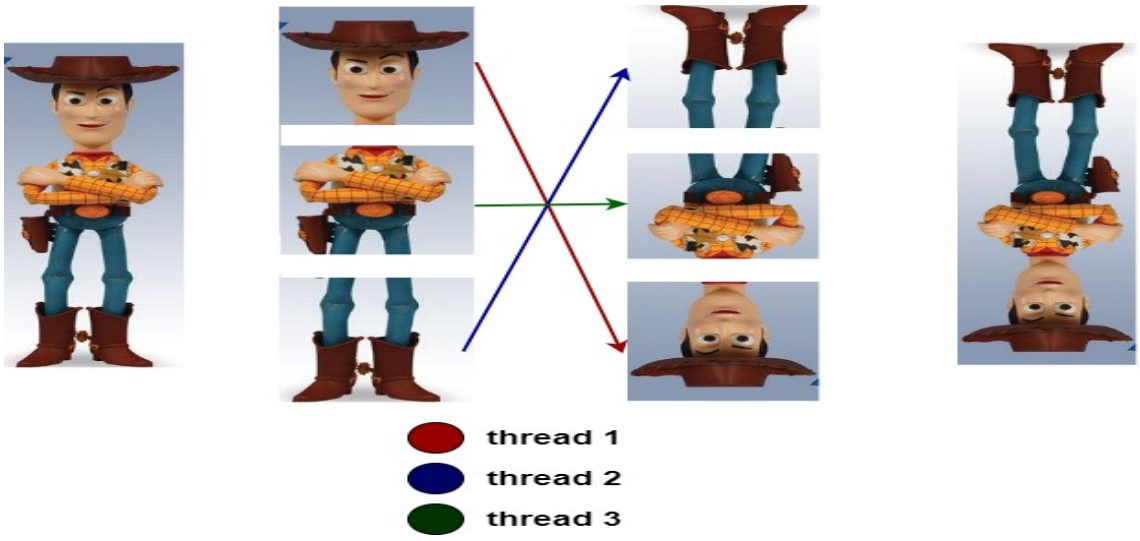


Our approach is the transformation algorithm that loops through the matrix of one image to copy each pixel to the opposite pixel of the second image.

Complexity: Since a matrix is traversed completely where the width is m and the height is n, the algorithm that we parallelize is O (n * m).

2.2 Parallel Solution

Our parallel solution assumes that there is no dependency between pixels when traversing the matrix, for this reason a data decomposition could be made where each existing thread is assigned a part of the image to invert. The following image shows graphically how the process would be in case of executing the algorithm with 3 threads.



3 Conclusions

- The parallelization exercise did not have the expected results, all because although in the first instance its potential to parallelize it was known, the communication that the threads have was not taken into account was going to be so high, because being this way The advantages of parallel programming over sequential programming are lost.
- The parallelization in this case is much more useful using the threads to execute different images since in this way the threads do not share too much information. This implementation can be useful to rotate a complete video frame by frame, this implementation is outside the scope of this partial.