A parallel algorithm for finding the human face in the image

H N Zaynidinov¹, O U Mallaev¹ and B B Anvarjonov²

¹Tashkent university of Information technologies named after Muhammad al-Khwarizmi (TUIT), Tashkent, Uzbekistan
²Tashkent institute of design, construction and maintenance of automobile roads, Tashkent, Uzbekistan

E-mail: info-oybek@rambler.ru

Abstract. It is well-known that anyone who processes and analyzes image information needs a large amount of memory for storing images. For example, for storing a standard TV frame, if two of the three colors contain 2 bits for each of the 3 bits, the memory of at least 512x512 bytes is required. A passport image must be at least 64x64x64 bits (this requires more space than any other passport data). The problems associated with storage, search, and discovery are made more complex by processing and analyzing large numbers of images. The image contains a large amount of information, so it is natural for them to have excessive statistical, psycho-visual and content information. Due to this redundancy, the process of digital image processing, image retrieval time, and storage capacity increase. The use of parallel technologies in digital processing of images plays an important role in solving the above problems. Therefore, the article presents an algorithm for calculating the process of extracting a human face from a color image using MPI technology.

1. Introduction

It is well-known that anyone who processes and analyzes image information needs a large amount of memory for storing images. For example, for storing a standard TV frame, if two of the three colors contain 2 bits for each of the 3 bits, the memory of at least 512x512 bytes is required. A passport image must be at least 64x64x64 bits (this requires more space than any other passport data). The problems associated with storage, search, and discovery are made more complex by processing and analyzing large numbers of images. The image contains a large amount of information, so it is natural for them to have excessive statistical, psycho-visual and content information. Due to these redundancy, the process of digital image processing, image retrieval time, and storage capacity increase. The use of parallel technologies in digital processing of images plays an important role in solving the above problems. Therefore, the article presents an algorithm for calculating the process of extracting a human face from a color image using MPI technology.

The images are defined by a two-dimensional function f (x, y), where x and y are in a particular space (or more precisely a plane), where the value of f is called the intensity or gray phase at the coordinate point (x, y). Some publications call brilliance. If x, y, and f accept discrete values, they are called digital images. Digital image processing refers to the processing of digital images by computers. Any digital image is subdivided into finite elements in specific fields and accepts the defined values.
These elements are referred to as image elements or pixels (the word pixel means English picture element and is translated as an image element.

Color images. You need at least 2 bits per pixel of these images.

Each pixel has a 16 bit (65,536 color) image with a High Color name and a 24 bit (16.7 million color) image at each pixel called True Color. Computer graphics systems also use images with 32.44 pixels or more per pixel.

Types of information provided in the image form. When it comes to computer image processing, it is desirable to divide the images into four groups:

- Grayscale and color images.
- Two and several color images.
- Double curves and straight lines.
- Dots or polygons.

A color image can be stored using three matrices (for red, green and blue), or by using a single matrix that separates specific bits for each color. It is known that if the difference in illumination is less than 1 percent, the eyes of the person usually cannot perceive it. Therefore, when storing color images, it is enough to allocate one byte for the color of each pixel. However, to achieve the best results, one byte of information is sufficient when storing each pixel of the color image in memory. In this case 3 bits for each of the two colors and 2 bits for the third color. From a mathematical point of view, it is convenient to view a color image as a three-dimensional vector matrix in some cases.

2. Algorithm

The proposed method for detecting faces refers to methods using invariant features, and is based on the allocation of image areas according to the color of human skin. The input is a color image of the size. The algorithm consists of 4 stages [1].

Stage 1. Convert the original image from the RGB color model to YCrCb using the following formula (1).

\[
\begin{align*}
    Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\
    Cr &= 0.5 \cdot R - 0.4187 \cdot G - 0.0813 \cdot B + 128 \\
    Cb &= -0.1687 \cdot R - 0.3313 \cdot G + 0.5 \cdot B + 128
\end{align*}
\]

Here, Y is the brightness of the original image, Cr is the difference of the channel R of the original image and brightness Y, Cb is the difference of the channel B of the original image and brightness Y.

Stage 2. As a result of the experiments, it was revealed that in the ranges of the Cr channel [133; 173] and the Cb channel [77; 127], the color of human skin changes, i.e. the selection of the human skin region S1 in the image is as following formula(2) [2]:

\[
S_1 = \begin{cases} 
1, & \text{if } \left[ Cr(x, y) \in [133;173] \right] \cap \left[ Cb(x, y) \in [77;127] \right] \\
0, & \text{otherwise}
\end{cases}
\]

Stage 3. At this stage, the task is set of eliminating the noise generated as a result of applying the actions of the 1st stage. Noise can form in the form of dots and small holes. To eliminate noise, we calculate the image density map according to the following formula (3):

\[
D(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} S_i (4x + i, 4y + j)
\]
Points with a zero density value (D(x, y) = 0) do not apply to the face area. Points with a full density value (D(x, y) = 16) refer to the face area. Any point in the intermediate density value (0 < D(x, y) < 16) indicates the presence of noise.

For the density map D(x, y), we perform the following steps:

- zero the values of all extreme points – formula (4), i.e.
  \[ D(0, 0) = D(x, 0) = D(m, y) = D(x, n) = 0 \]  
  \[ D(n, y) = D(m, 0) = D(0, n) = 0 \]  

- zero out the values of all points of total density () if they are surrounded by less than five points of total density;
- assign a value of 16 to all density points if there are more than two points of total density in their vicinity.

At this stage, the collection process consists of four cycles. That is, in two external cycles, the calculation is performed on 362 x 468 = 168416 pixels of a two-dimensional array derived from the image. In two internal cycles, the calculation is done at 4x4 = 16 pixels. This computational process is thrown out of the MPI technology for parallel computing to the parallel streams of the operating system \[3,4,5\].

After these actions, the density map is converted into an image as following formula (5):

\[ S_2 = \begin{cases} 
1, & \text{if } D(x, y) = 16 \\
0, & \text{otherwise} 
\end{cases} \]  

Stage 4. It is known that, basically, in contrast to the background, the region of the face in the image is heterogeneous. Therefore, based on this property, it is possible to more accurately determine the boundary between the background and the face area – formula (6). To do this, we calculate the standard deviation of the value of each aperture W of size 8x8:

\[ \delta(x, y) = \sqrt{E(W^2) - E(W)^2} \]  

At this stage, the process performed in Stage 3 will be followed \[6,7\].

The velocity of digital processing of the matrix elements increases even further when the vectorization in steps 3 and 4 is carried out. The vectorization process is the addition, subtraction, multiplication, or division of the N elements of the matrix simultaneously.

If the standard deviation of \( \delta(x, y) \) is less than two, then the window in question is too homogeneous and therefore can hardly be part of the face area. We apply this action to all apertures W – formula (7):

\[ S_3(x, y) = \begin{cases} 
1, & \text{if } S_2(x, y) = 1 \text{ and } \delta(x, y) \geq 2 \\
0, & \text{otherwise} 
\end{cases} \]  

As a result, we get an image containing an area of the face with minimal content or no noise.

3. Results:

The time spent in consecutive and parallel calculations in the third and fourth stages of the algorithm is shown in the table 1 below:

| Number of images | Time spent on the calculations | Acceleration factor (S_4(n)) |
|------------------|-------------------------------|-----------------------------|
|                  | Sequence (sec.) | Parallel (sec.) | |
| 1                | 0.00021         | 0.00017           | 1.235             |
| 5                | 0.00175         | 0.00125           | 1.400             |
According to the data in table 1, 0.00021 seconds for 1 image, 0.00017 seconds for parallelization, 1.152 seconds for 20 images, 0.391 seconds for parallelization and 2.354 seconds for 40 images, and 72727 seconds for parallelization. The acceleration factor is calculated by formula 8:

\[
S_p(n) = \frac{T(n)}{T_p(n)}
\]  

(8)

Where \( T(n) \) is the amount of time spent performing a consecutive program; \( T_p(n) \) - the time spent on running a parallel program.

Figure 1 shows the interface of the algorithm based on the proposed algorithm. Software is created using MPI technology in C++ programming language. The process of separating informative signs of the human face from the program image and removing the inactive characters are shown in the sequence.

4. Conclusion
The time spent extracting a person’s face in color images can be reduced by the parallel algorithm proposed in the article. If this process is used in real-time to solve human identification issues, it can provide information about the person you are looking for shortly. The process of comparing the wanted information with the pre-processed informative markers of N human faces in the database enhances the effectiveness of the search. If this parallel algorithm is used in car alarm systems, it will reduce the car crash speed. If this parallel algorithm is used in car alarm systems, it will reduce the car crash speed. It also allows you to predict people who accidentally get in front of the car and automatically stop the car. Used in surveillance and search engines, it is possible to quickly and efficiently analyze the person being sought.

References
[1] Fazylov Sh Kh, Tukhtasinov M T, Starovoytov V V, Samal D I and Rigol G 2004 Lokalizatsiya fragmentov litsa na tsvetnykh fotoportretakh. Obrabotka informatsii i upravleniye v chrezvychnykh i ekstremalykh situatsiyakh: tez.dokl. 4-uy Mezh. nauchn. konf. Minsk. 1 218–23
[2] Al-Akkad M A 2016 Using curvature and neural networks for recognizing faces. журнале Дамасского Университета в 1 22-35
[3] Antonov A S 2004 Parallel programming using MPI technology (Moscow: Moscow State University Press) 71
[4] Billig V A 2016 Parallel computing and multi-threaded programming (Moscow: Moscow State University Press) 311
[5] Burova I G and Demjanovich Yu K 2010 Minimal splines and their applications (Moscow: Moscow State University Press) 364
[6] Mallaev O U 2018 Algorithm and software for speeding up computer memory using OpenMP technology. Descendants of Mohammed al-Khwarizmi 4 71-5
[7] Gergel V P 2010 High-performance computing for multi-core multiprocessor (Nizhny Novgorod: Nizhny Novgorod State University Press) 421