Yerdaulet Zeinolla

Report(Project3)

Firstly, the program was started by making some google search about the operations with color in opencv. Fortunately, it was described as one of the basic tasks, which is not so difficult and I started to write the program. I have learned how to deal with BGR values of pixels, how to deal with trackbars and understood how to work with matrices in OpenCV.

Main Task(color selecting)

My final code consists of five helping functions and one main. Firstly, I will explain the Main one.

```
}

//Gives Initial Annotations
static void Guide() {
    cout << "/////////////////-------READ ME-------//////////////////" << endl;

    cout << "\t Make a click with the Left Button of a mouse for a Color Selec
        "\t B using the trackbar you can adjust the R for colo enclosure\n"<< endl;

    cout << "\t Use the Hotkeys on a keyboard:\n"
        "\t 'r'   - switchess On/Off the color selection\n"
        "\t 's'   - Calls and Stops the SURPRISE\n"
        "\t 'ESC' - Stop The Program\n" << endl;

    cout << "\t In SURPISE! window:\n"
        "\t Mouse Left click      - Select the First image\n"
        "\t Mouse Right click     - Select the Second image\n"
        "\t 'ESC'                 - Stop The Program\n" << endl;
    cout << "/////////////////-------READ ME-------//////////////////" << endl;
    return;
}

int main(int, char**)
{
    Guide();
```

```
C:\Windows\system32\cmd.exe
***** VIDEOINPUT LIBRARY - 0.1995 - TFW07 *****
/////////////////-------READ ME-------//////////////////
        Make a click with the Left Button of a mouse for a Color Selection
        B using the trackbar you can adjust the R for colo enclosure

        Use the Hotkeys on a keyboard:
        'r'     - switchess On/Off the color selection
        's'     - Calls and Stops the SURPRISE
        'ESC'   - Stop The Program

        In SURPISE! window:
        Mouse Left click        - Select the First image
        Mouse Right click       - Select the Second image
        'ESC'                   - Stop The Program
```

As you can see, I started from writing the short Guide for user on how the program works. Thus, the main function starts from these annotations.

```
VideoCapture cap(0);
if (!cap.isOpened())
    return -1;

namedWindow("Color Detection", 1);
setMouseCallback("Color Detection", clickMouse, 0);
createTrackbar("Enclosure", "Color Detection", &radius, 255, 0, 0);
while (1) {
    cap >> frame;
    temp = frame*1.3;
    if (colorSelected) {
        for (int y = 0; y<frame.rows; y++) {
            for (int x = 0; x<frame.cols; x++) {
                Vec3b pixel = temp.at<Vec3b>(Point(x, y));
                if ((pow(pixel[2] - bgr[2], 2) + pow(pixel[1] - bgr[1], 2) + pow(pixel[0] - bgr[0], 2))>pow(radius, 2)) {
                    pixel[2] = int((pixel[2] + pixel[1] + pixel[0]) / 3);
                    pixel[1] = pixel[2]; pixel[0] = pixel[1];
                }
                pixel /= 1.3;
                frame.at<Vec3b>(Point(x, y)) = pixel;

            }
        }
    }
```

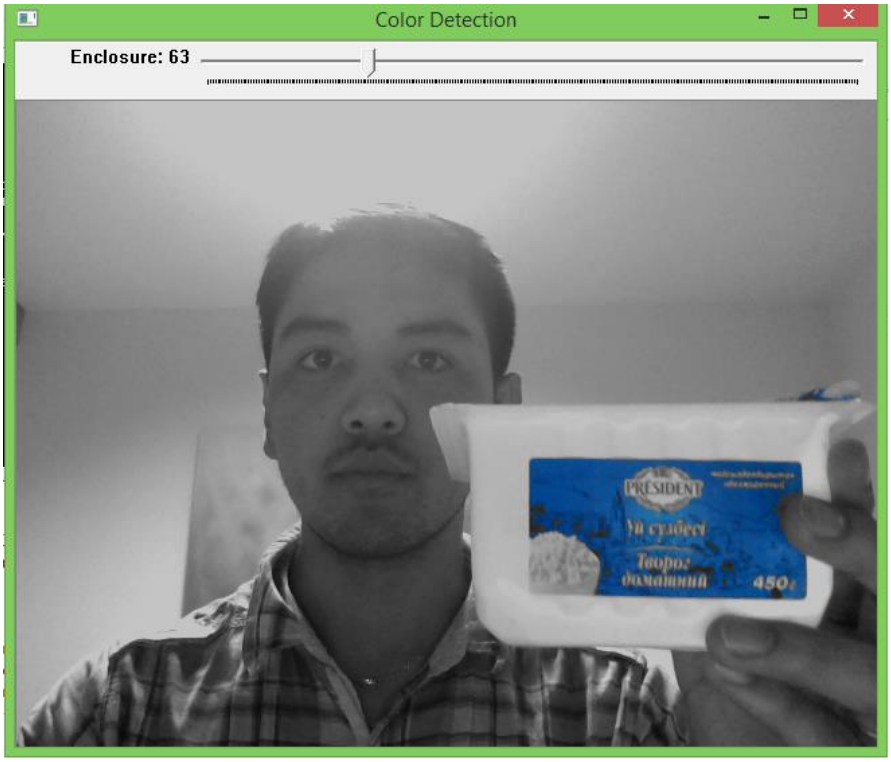
The next step was to connect to the Video Camera, create main Window, adjust mouse function and capture a frame. Just after that, I use variable **temp** in order to increase the contrast of main image and store in it. Further the program goes through each pixel(variable

pixel is created) in an Image with two for loops and proceeds each of them by a formula for the Sphere, given in an Assignment Description. Here the variable **bgr** is global and is a type of Vec3b, it adjusted in **clickMouse** function. It stores the information about the clicked pixel. The variable **radius** changes according to the trackbar, and if the squared values of all colors are higher than the adjusted radius, then these colors are averaged(gray scaled). Otherwise, the same color values for pixel are preserved.

```
//Mouse Function
static void clickMouse(int event, int x, int y, int flag, void* param)
{
    if (event == CV_EVENT_LBUTTONDOWN) {
        bgr = temp.at<Vec3b>(Point(x, y));
        colorSelected = true;
    }
    return;
}
```

Only one function is created for the main task. It picks the value of clicked pixel and stores it in **bgr.**

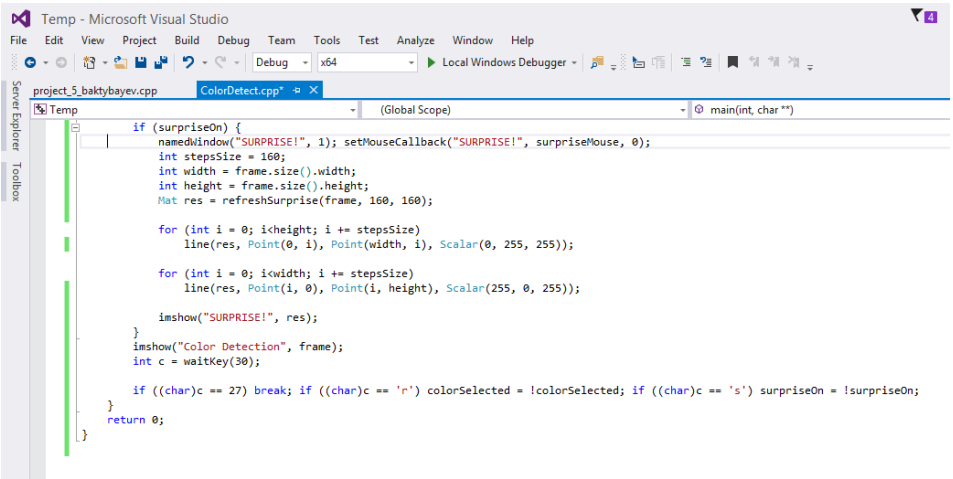
How it Works:



Yerdaulet Zeinolla

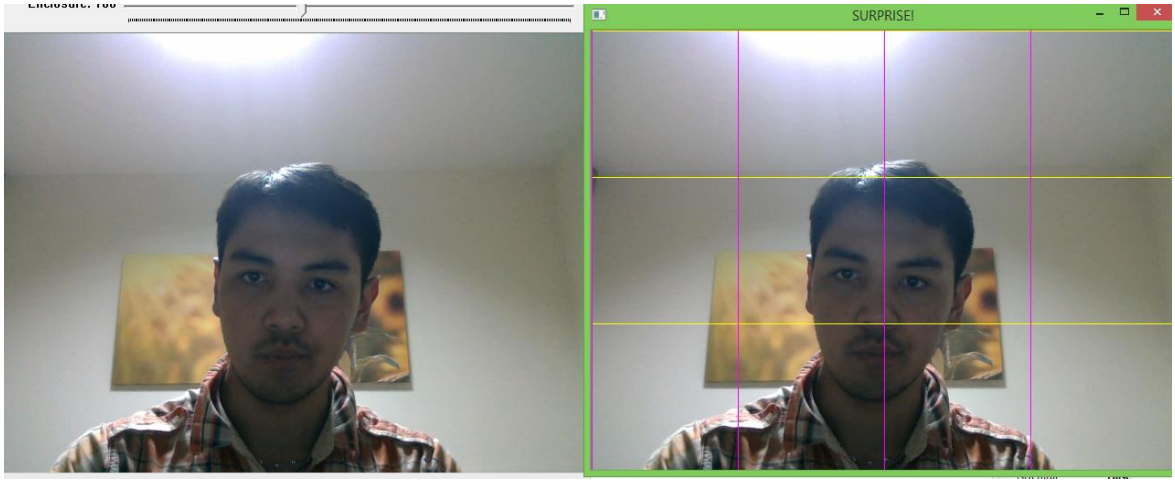
Surprise function:

The difficult part of this project was to create a Surprise button. I represented it as a HotKey ‘s’, after pressing which the new window will appear. Moreover, this task required me to find out how to work with separate blocks from a original matrix.




```
        if (surpriseOn) {
            namedWindow("SURPRISE!", 1); setMouseCallback("SURPRISE!", surpriseMouse, 0);
            int stepsSize = 160;
            int width = frame.size().width;
            int height = frame.size().height;
            Mat res = refreshSurprise(frame, 160, 160);

            for (int i = 0; i<height; i += stepsSize)
                line(res, Point(0, i), Point(width, i), Scalar(0, 255, 255));

            for (int i = 0; i<width; i += stepsSize)
                line(res, Point(i, 0), Point(i, height), Scalar(255, 0, 255));

            imshow("SURPRISE!", res);
        }
        imshow("Color Detection", frame);
        int c = waitKey(30);

        if ((char)c == 27) break; if ((char)c == 'r') colorSelected = !colorSelected; if ((char)c == 's') surpriseOn = !surpriseOn;
    }
    return 0;
}
```

Firstly, the global Boolean variable surpriseOn determines whether the function starts or no. Just after that the new window is created with the appropriate mouse function for it.**StepsSize** I used as 160, which divides 640 by and 480 by 3, it is the size of grid’s cells which will divide the Video Frame. In the middle I used two for loops in order to draw division lines.



For Surprise I created a separate mouse function. It reads two clicks: 1st reads the coordinates of first block and 2nd click reads the coordinates of the second block. After these values are recorded in **iX1, iY1** and in **oX1, oY1,** the **SwapPic(iX1, iY1, oX1, oY1)** is called.

```
//Function for mouse operation in Surprise window
static void surpriseMouse(int event, int x, int y, int flag, void* param){
    if (event == CV_EVENT_LBUTTONDOWN && surpriseOn == true) {

        if (!choosePic) { iX1 = x; iY1 = y; choosePic = true; }

    }
    else if (event == CV_EVENT_RBUTTONDOWN && surpriseOn == true) {
        if (choosePic) { oX1 = x; oY1 = y; choosePic = false; SwapPic(iX1, iY1, oX1, oY1); }
    }
}
```

Yerdaulet Zeinolla

```
//function Swaps the coordinates of blocks
static void SwapPic(int iX, int iY, int oX, int oY)//iX - iniial X, oX - output X
{
    int rowNi, colNi, rowNf, colNf; //rowNi - initial Row Number, rowNf - final Row Number
    colNi = (iX / 160); rowNi = (iY / 160);
    colNf = (oX / 160); rowNf = (oY / 160);
    int tempx = swapInfoX[colNi][rowNi], tempy = swapInfoY[colNi][rowNi];
    swapInfoX[colNi][rowNi] = swapInfoX[colNf][rowNf];
    swapInfoY[colNi][rowNi] = swapInfoY[colNf][rowNf];

    swapInfoX[colNf][rowNf] = tempx;
    swapInfoY[colNf][rowNf] = tempy;
}
```

The function **SwapPic** swaps the blocks with each other. Previously, the coordinates of clicks were read and at this function these coordinates are processed. The **swapInfoX[][]** and **swapInfoY[][]** are two global variables, together they store the position of each block. The former indicates x coordinates of each block and the latter y coordinates. Example is given below, the numbers indicate the number of Block. For example: the Block #0 should be in a first cell, by default. But here it's **swapInfoX[0][0]=3** and **swapInfoY[0][0]=0.**

2	**3**	**5**	**0**
1	**11**	**10**	**9**
6	**7**	**8**	**4**

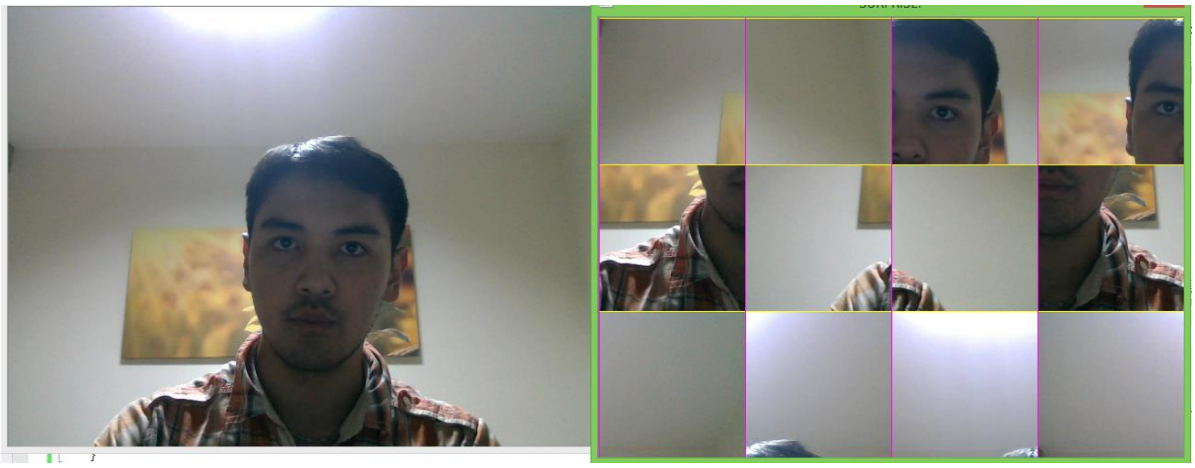
So, in a function swapPic the blocks get their new coordinates in **swapInfoX[][]** and **swapInfoY[][].**

Yerdaulet Zeinolla

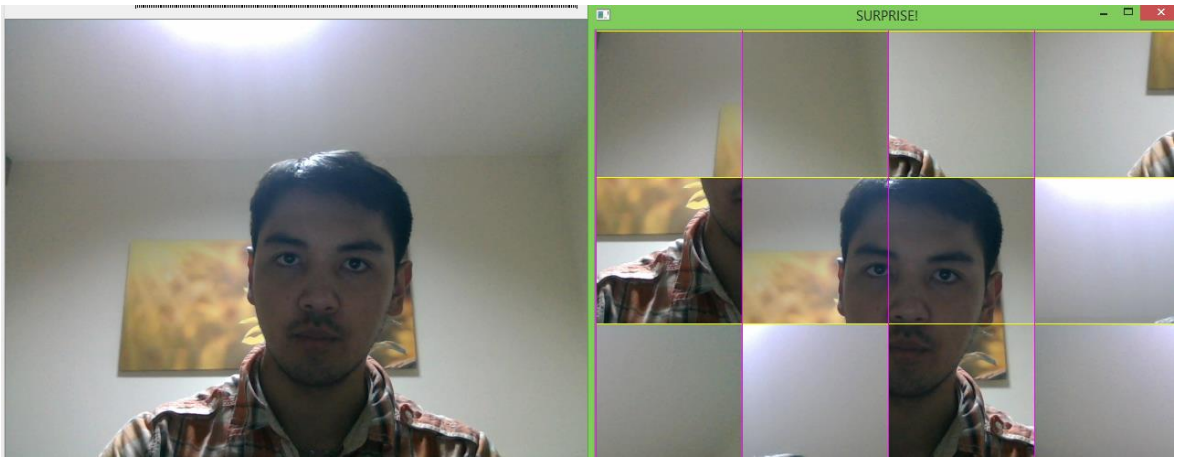
```
//function which updates the image called by Surprise button
Mat refreshSurprise(const cv::Mat& src, int cw, int ch)
{
    using namespace cv;
    int w(frame.cols);
    int h(frame.rows);
    CV_Assert(w >= cw);
    CV_Assert(h >= ch);
    Mat dst(3 * ch, 4 * cw, frame.type());
    int k = 0;
    for (int i = 0; i < 4;i++)
        for (int j = 0; j < 3;j++)
        {
            Mat(src, Rect(swapInfoX[i][j]*160, swapInfoY[i][j]*160, cw, ch)).copyTo(Mat(dst, Rect(i*160, j*160, cw, ch)));
            k++;
        }
    return dst;
}
```

In Main function of Surprise the only called function is refreshSurprise. It updates the position of each separate block in an Image. Here the main operation is done in two for loops, it outlines the ROI of a frame from Web Cam, puts the particular ROI to another place in a new mastrix **dst.** You can see that **swapInfoX** and **swapInfoY** are used in order to determine the block from which part of a frame will be picked.

How it Looks:



Moreover, you can assemble this mosaic by LeftClicking on one block and after that RightClicking on a second.



Conclusion

This assignment gave us opportunity to work with OpenCV, which I find more comfortable to use in comparison to MatLab. Furthermore, it let me study the new programming language, C++ and let me look at the images from a numerical side and understand how they are constructed. In a future the Surprise function could be improved, because I did not have enough time to do so and as a result it works only for images of size 640x480.