OpenCV lab 2

Federico Luisetto - 2074282

April 2, 2023

Abstract

The aims of this laboratory was to become familiar with

- mouse callback and mouse events;
- color segmentation, color picked through mouse event.

1 Task 1

In the first task we only had to show the provided image.
The referred image is Figure 1

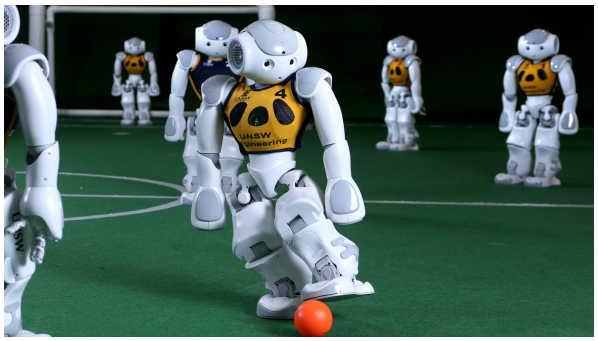


Figure 1: Referred image

2 Task 2

In the second task we had to display 1 and print the color BGR triplet of the pixel where the click occurred. The main difficulty was to understand passing image data to the function `onMouse`. In the main I invoked `cv::setMouseCallback("Img", onMouse, (void*)&img);`, instead in the function I managed it in this way:

```
void onMouse(int event, int x, int y, int f, void* userdata) {
    if(event == cv::EVENT_LBUTTONDOWN) {
            cv::Mat img = *(cv::Mat*) userdata;
            cv::Vec3b point = img.at<cv::Vec3b>(x,y);
        ...
    }
}
```

3 Task 3

In the third task we had to calculate the mean of the BGR values (separately) using the 9x9 neighborhood around the pixel where the click occurred. After the creation of the `cv::Mat img` object inside the `onMouse` function, I managed the task in this way:

```
...
cv::Mat img_out = img.clone();
if(y+NBHD_Y > img_out.rows || x+NBHD_X > img_out.cols)
    return;
cv::Rect rect(x, y, NBHD_X, NBHD_Y);
cv::Scalar mean = cv::mean(img_out(rect));
...
```

4 Task 4

In the forth task we had to calculate the mean of BGR values (as previously done in task 3), and then create a new image having

- a white pixel if the corresponding pixel in the input image has all the three B, G and R values having a distance from the reference color not greater than a threshold T;
- a black pixel otherwise.

To the code of task 3 I added the following code:

```
cv::Mat img_mask = cv::Mat::zeros(cv::Size(img.cols,img.rows), CV_8UC1);
uchar T = 75;
for(int i=0; i<img.rows; i++) {
    for(int j=0; j<img.cols; j++) {
        cv::Vec3b vec = img.at<cv::Vec3b>(i,j);
        if(std::abs(vec[0]-mean[0])<T && std::abs(vec[1]-mean[1])<T && std::abs(vec[2]-mean[2])<T)
            img_mask.at<uchar>(i,j)=255;
        else
            img_mask.at<uchar>(i,j)=0;
    }
}
```

The output image selecting the T-shirt of the robot in Figure 2. The main difficulty were to find the

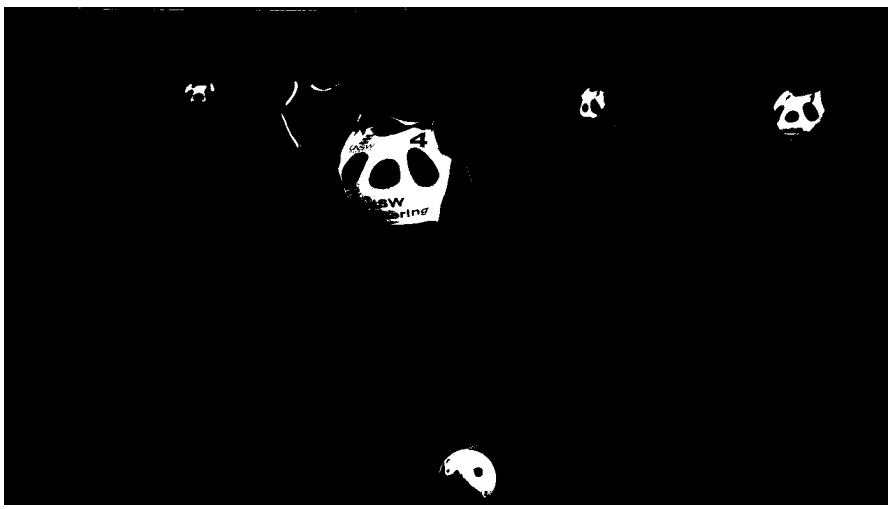


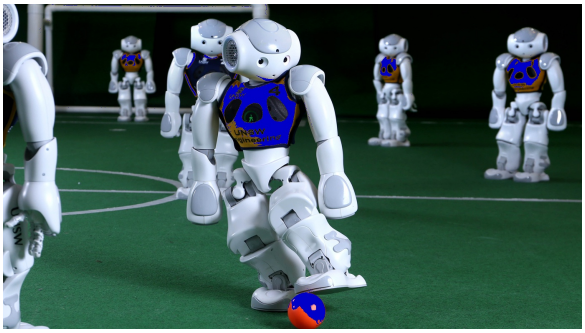
Figure 2: Mask image

right compromise of the threshold for selecting the right amount of pixels.

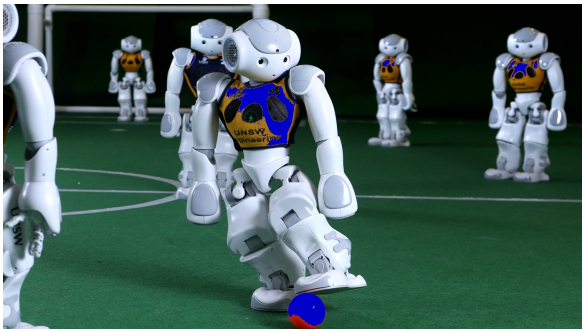
5 Task 5

In the fifth task we had to create, from the one in task 4, a new image whose pixels are

- equal to the input image if the corresponding pixel in the mask is black;



(a) New color when selecting T-shirt



(b) New color when selecting ball

Figure 3: New color images

- equal to the given color BGR = (92, 37, 201) otherwise.

The code for solving this task is similar to the one in task 4. The output images after selecting the T-shirt and the ball is in Figure 3