04/16/2019

Armandas Rokas (s185144)

C assignment 4

Rapport animal activity in forest - recording animal activity with us of Raspberry Pi Camera

Table of contents

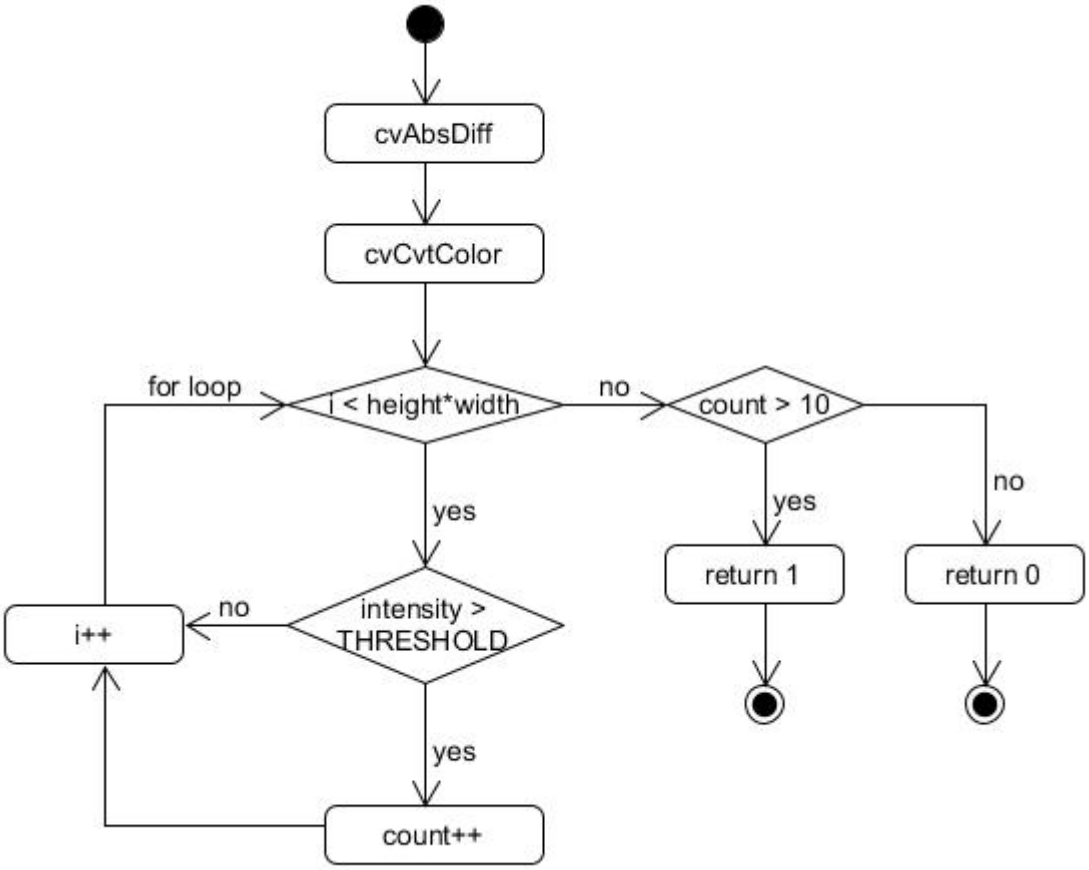
Intro

This document describes the implementation of "Rapport animal activity in forest - recording animal activity with us of Raspberry Pi Camera " solution.

Detect movement

Firstly, the function finds the difference between pictures with `cvAbsDiff` function from `openCv` library. Next, the resulting picture is converted to grayscale with `cvCvtColor` function in order to facilitate an examination of the picture, because in this way every pixel will have only intensities between 0 and 255.

After that, every pixel is checked if it has more intensity than the user-defined `THRESHOLD` constant. If yes, it adds 1 to `count` variable. The count variable is used to minimize the chance of reporting the false movement.




```
int compareImages(IplImage * image1, IplImage * image2){
        IplImage* res = cvCreateImage(cvGetSize(image1), 8, 3);

        cvAbsDiff(image1,image2,res);

    IplImage* gray_res = cvCreateImage(cvGetSize(res), 8, 1);
    cvCvtColor(res, gray_res, CV_RGB2GRAY );
        int count = 0;
        for(int i = 0; i < res->height*res->width;i++){
          if(gray_res->imageData[i] > 50){
```

```
                count++;
          }
        }
        cvReleaseImage(&res);
        cvReleaseImage(&gray_res);
        if(count > 10){
                return 1;
        } else{
                return 0;
        }
}
```

Rapport time and date

When the movement is detected it has to be logged in a text file. It has been done by opening file for appending with `fopen` function using an `a+` mode. There was used `sprintf` to combine the file name string with a current date. Finally, there is used `fprintf` for writing to the file.

```
void logTxt(struct tm * timeinfo){
        static const char mon_name[][4] = {
        "JAN", "FEB", "MAR", "APR","MAY", "JUN",
        "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};

        FILE * cfPtr;
        char * file_name = (char *) malloc(15*sizeof(char));
        int year = timeinfo->tm_year + 1900;
        sprintf(file_name, "log%d%s.txt", year, mon_name[timeinfo->tm_mon]);
        if ((cfPtr = fopen(file_name, "a+")) == NULL){
                printf("File could not be opened");
                exit(1);
        }
        fprintf(cfPtr, "%d-%02d-%02d %02d:%02d:%02d\n",
                        year,
                        timeinfo->tm_mon+1,
                        timeinfo->tm_mday,
                        timeinfo->tm_hour,
                        timeinfo->tm_min,
                        timeinfo->tm_sec);
        free(file_name);
        fclose(cfPtr);
}
```

Saving a picture

In order to save the picture is used `cvSaveImage` function. `sprintf` combines a file name string and a text information string that is used to embed into a picture.

`cvPutText` puts the text on the picture with defined `color` in `CvScaler` and `base_font` in `CvFont`. It is important to notice here that there is making a clone of the original picture and the text is put on the clone in order to not affect the original picture. It's has been implemented like this, because if the text is put on the original picture, it could affect the next comparison and the text will be calculated as a difference in the pixels,

which is not desired.

```
void logJpg(struct tm * timeinfo, IplImage * image){

        CvScalar color;
        CvFont base_font;
        color = CV_RGB(36,0,250);
        cvInitFont( &base_font, CV_FONT_HERSHEY_SIMPLEX, 1.5, 1.5, 0, 1, 8);

        char * currTime = (char *) malloc(19*sizeof(char));
        sprintf(currTime, "%d-%02d-%02d %02d:%02d:%02d",
                        timeinfo->tm_year+1900,
                        timeinfo->tm_mon+1,
                        timeinfo->tm_mday,
                        timeinfo->tm_hour,
                        timeinfo->tm_min,
                        timeinfo->tm_sec);

        IplImage * edited_image = cvCloneImage(image);
        cvPutText(edited_image, currTime, cvPoint(10, 400), &base_font, color);

        char * file_name = (char *) malloc(23*sizeof(char));
        sprintf(file_name, "%s.jpg", currTime);

        cvSaveImage(file_name, edited_image, 0);

        cvReleaseImage(&edited_image);
        free(currTime);
        free(file_name);
}
```

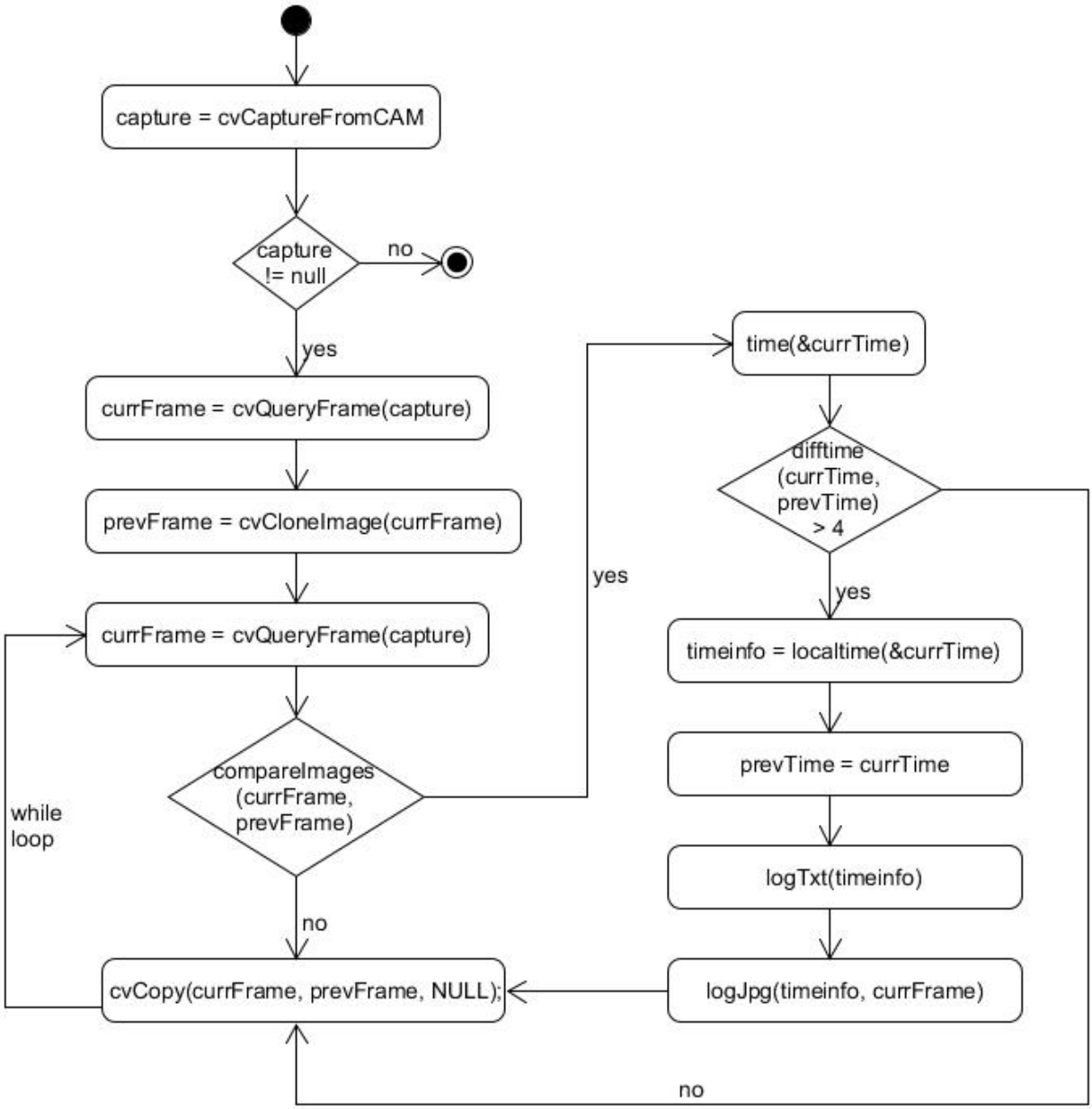
Main function

The main function sets all functions working together.

Firstly It uses `cvCaptureFromCAM` to capture a video from `RaspberryPi` camera.

After that there are making two frames (the current frame and the previous frame) by cloning the current frame with `cvCloneImage` function. These two frames is compared with `compareImages` function described above.

If there a difference in the frames, the function checks if there are more than 4 sec from the last recorded activity with `diffTime` function. If yes, so the time is registered and logged in the text file with `logTxt` function and a picture is saved with `logJpg` function.




```
int main()
{
        CvCapture* capture;
        time_t prevTime;
        time_t currTime;
        struct tm * timeinfo;
        IplImage* currFrame = 0;
        IplImage* prevFrame = 0;
        // Read the video stream
        capture = cvCaptureFromCAM(0);
        if( capture )
        {
                currFrame = cvQueryFrame( capture );
```

```
                prevFrame = cvCloneImage(currFrame);
                while(1)
                {
                     currFrame = cvQueryFrame(capture);
                     if(compareImages(currFrame,prevFrame)){
                           time(&currTime);
                           if (difftime(currTime, prevTime) > 4){
                                  timeinfo = localtime(&currTime);
                                  prevTime = currTime;
                                  logTxt(timeinfo);
                                  logJpg(timeinfo, currFrame);
                               }
                        }
                        cvCopy(currFrame, prevFrame, NULL);
                }
        } else {
                printf("Error capturing camera\n");
        }
    cvReleaseCapture( &capture ); // release memory.
    cvDestroyWindow("Video"); //destroy windows
    return 0;
}
```

Conclusion

The main goal to detect and record the activity was achieved. The program works like it's expected, however it''s only tested indoors.