Hacettepe University Computer Engineering Internship Report

NAME – SURNAME : *Yavuz Baykut KAYA*

CLASS: *2nd Grade*

NUMBER: *21427085*

SUBJECT OF INTERNSHIP: *Software*

START - END DATE: *05/06/2017-18/07/2017*

DURATION OF INTERNSHIP: *6 weeks/30 work day*

COMPANY NAME/ ADRESS: Gazi University Gölbaşı Campus Techno Plaza BZ-16 Gölbaşı , Ankara

Table Of Contents

1.INTRODUCTION

This internship is a software internship. Main focuses of internship are image processing and biometrical analysis. Applications do image segmentation, real time object tracking based on color, real time face detection, real time face tracking and produce value for similarity between given two face images. At the end of internship how to image processing functions work and how to apply them in real applications were understood.

2. CORPORATION

2.1. Company Name

MIA Teknoloji Yazılım Tasarım Mühendislik Limited Şirketi

2.2. Company Location

Gazi University Gölbaşı Campus Techno Plaza BZ-16 Gölbaşı/ANKARA

e-Mail: info@miateknoloji.com

Phone: 0 (312) 286 17 77
Call Center: 444 4 642
Fax: 0 (312) 484 37 73

2.3. Short History of Company

MIA Technologies was founded by three computer engineer who are gruaduated from same university in 2006. Company taken care of research and development and has been following global trendings. The company also provides health information management for some local hospitals and controls approximately 325.000 patient's information each month.

2.4. Main Area of Company

2.4.a. Biometric Person Identification Systems

The company provides Multi-Biometric Identification solutions .These Multi-Biometric Identification devices occurs with combination of fingerprint, finger vein ,face and voice recognition systems. Multi-combinational system offers its customers different choices for different situtations.

2.4.b. Health Information Management

The corporation provides hospitals or medical centers a solution that can facilitate archiving the laboratory,radiology,clinic and other information about patients.

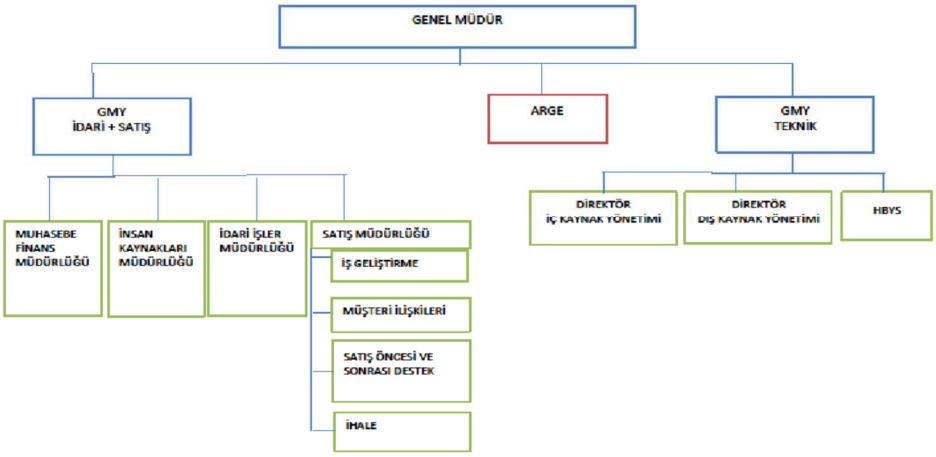
2.4.c. Border Security

The company produces gates which bases on BEOGS(biometric integrated automatic switching system). This system creates an easy and fast integrated e-Gate solution for border passing places such as airports or border gates.

2.4.d. Face Recognition in Crowded Areas

Providing the security in daily life is not an easy job unless we decrease its speed. But MİA Teknoloji offers a solution which can provide security when daily life runs. This system allows authorized people to detect criminals or suspicious people.

2.5. Organizaiton Scheme



2.6. Number of Employees

Company has 3 main parts. These are management-sales, R&D and Technical support departments. It has totally 195 workers in these three departments 25 of 195 are administrative staff, 70 out of 195 are engineers. Then technical stuff and others. For 70 engineers, 14 out of 70 are electrical and electronics engineers, 26 out of 70 are computer and software engineers and other engineers.

3. INTERNSHIP PROCESS

3.1. Real-Time Object Detection Based on Color

Aim of this Project is detect objects via using appropriate functions with given RGB color values from user while streaming.

Program basically has two functions.

3.1.a. Color Finder Function

Function takes 4 parameters. Given image, lowest threshold of color, highest threshold of color and color name as a string for printing current color on objects.

First of all, using inRange() function makes visible that only given color. After that using dilate() and erode() functions provides morphological operations for better segmentation of objects. These changes on given image saved in another Mat image that called imageThresholded.

Next, two vector variables defined for storing vector of points and vectors of hiearchy. Then using findContours() function with appropriate parametres.(Target image, countors, hiearchy , CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE).

CV_RETR_EXTERNAL : Retrieves only the extreme outer contours.

CV_CHAIN_APPROX_SIMPLE: Compresses horizontal, vertical, and diagonal segments and leaves only their end points.

Finally using for loop and getting coordinates of objects with boundingRect() function and storing via Rect variable ; printing color name on the middle of objects. White color(RGB: 255,255,255) using for drawing contours on new window via drawContours() function with proper parameters and imshow() function shows new window with drawed objects.

```
void colorFinder(const Mat image , const Scalar lowColor,const Scalar highColor,const String stColor)
{
    Mat imageThresholded;
    inRange(image, lowColor, highColor, imageThresholded);// threshold the image.

    erode(imageThresholded, imageThresholded, getStructuringElement(MORPH_ELLIPSE, Size(5, 5)));
    dilate(imageThresholded, imageThresholded, getStructuringElement(MORPH_ELLIPSE, Size(5, 5)));

    dilate(imageThresholded, imageThresholded, getStructuringElement(MORPH_ELLIPSE, Size(5, 5)));
    erode(imageThresholded, imageThresholded, getStructuringElement(MORPH_ELLIPSE, Size(5, 5)));

    vector<vector<Point> > contours;
    vector<Vec4i> hierarchy;

    findContours(imageThresholded, contours, hierarchy, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE);

    for (int i = 0; i < contours.size(); i++)
    {
        Point center;
        Rect re = boundingRect(contours[i]);
        center.x = re.x + re.width / 2 - 10;
        center.y = re.y + re.height / 2;

        putText(image, stColor, center, FONT_HERSHEY_PLAIN, 2, Scalar(255, 255, 255));

    }

    drawContours(image, contours, -1, Scalar(255, 255, 255), 6, 10, hierarchy, 0);

    imshow(stColor, imageThresholded); //show the thresholded image

}
```

3.1.b. Main Function

In main function; first off all, videoCapture object that called cap(0) created for getting stream from default camera(0 means embedded camera). In while loop, stream frame saving as a Mat object and checking if there is problem with frame via if statement. Finally colorFinder() function calling for wanted colors(RGB values used) , showing original stream in new window for comparing results and with “Esc” key while loop successfully ending.

3.2. Face Detection and Face Tracking with KCF Tracker

3.2.a. Face Detector Function

This function takes just one image parameter as Mat object. For storing multiple face rectangles in one vector of Rect variable which is defined. Detection is based on OpenCV's face cascades and necessary functions.For cascades one cascade object is defined and with load() function cascade is loaded.(load function takes path of cascade location as a parameter). Then, cascades detectMultiScale() function returns face rectangles from given image, detectMultiScale function takes target array of face rectangles, takes face detection method as a parameters. Finally function returns face rectangles array.

```
#include <thread>
#include <future>

using namespace std;
using namespace cv;

Rect getFaceRectangle(Mat image)
{
    vector<Rect> faces;
    CascadeClassifier face_cascade;
    face_cascade.load("C:\\opencv\\sources\\data\\haarcascades\\haarcascade_frontalfa

    face_cascade.detectMultiS                    1.1, 2, 0 | CV_HAAR_SCALE_IMAGE, Size

    return faces[0];
}

int main()
```

cv::String::String(const char *s)

+ 11 overloads

Ln 23 Col 52 Ch 49

3.2.b. Main Function(Tracker Function)

First of all, videoCapture object and Mat frame are defined for streaming from webcam then Rect variable defined for current face. Bool type initial rectangle declared as false default.

OpenCV provides different trackers for object tracking and in this program KCF tracker has used(considered as most suitable tracker for face tracking).

In while loop, first if statement checks if there is a error while streaming then next outer if statement of nested stataments checks whether face detected or not. Inner if statement initiliazes KCF tracker via init() function. Then tracker is updating and if tracker can not updated, program prints error message and terminates program properly. For drawing rectangle around a detected face rectangle() function used with proper parameters. Finally in new window, tracking process is ended with “Esc” key from keyboard.

```
int main()
{
    VideoCapture cap(0);
    Mat frame;
    Rect2d face;
    bool initialRect = false;

    Ptr<Tracker> tracker = Tracker::create("KCF");


    while (true)
    {
        cap >> frame;
        if (frame.rows == 0 || frame.cols == 0)
            break;
        if(initialRect==false)
        {
            if (getFaceRectangle(frame).height)
            {

                face = getFaceRectangle(frame);
                tracker->init(frame, face);
                initialRect = true;
```

```
        }



        if(!tracker->update(frame, face))
        {
            cout << "Tracker can not locate target in current frame " << endl;
            waitKey(0);
            break;
        }

        rectangle(frame, face, Scalar(255, 0, 0));


        imshow("tracking", frame);

        if (waitKey(1) == 27)
            break;

    }


    return 0;
```

3.3. Face Recognition

In this program, “OpenBR” face recognition function is used. OpenBR is a framework for investigating new modalities, improving existing algorithms, interfacing with commercial systems, measuring recognition performance, and deploying automated biometric systems.

Aim of this application is produce similarity rate between two given images. When input is just two images, calculating accurate rate is difficult. From my observation, OpenBr provides us better results then OpenCV’s facial recognition algorithms if input size is just two.

OpenBr installation is quite challenging because it requires other programs and required versions. OpenBr requirements are:

3.3.a. CMake 3.0.2

CMake is an extensible, open-source system that manages the build process in an operating system and in a compiler-independent manner.

3.3.b. OpenCV 2.4.11

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library.

3.3.c. Qt 5.4.11

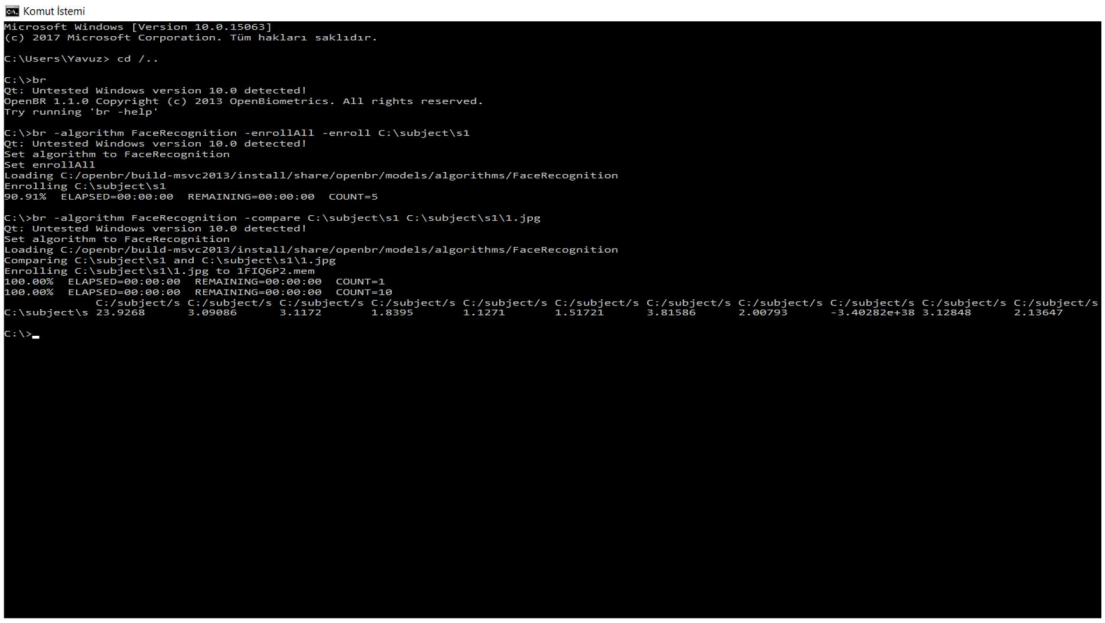
Qt is a cross-platform application development framework for desktop, embedded and mobile. Supported platforms include Linux, OS X, Windows, VxWorks, QNX, Android, iOS, BlackBerry, Sailfish OS and others.

3.3.d. GitHub

GitHub is a web-based Git or version control repository and Internet hosting service. It is mostly used for code. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features.

Results are quite confusing because of angle, light, skin color and race but threshold(Value for whether or not two images are belongs to same

person) can be approximately “1.15” and above out of 23.9268(Score of exact two images).



Results of comparing pictures of same person pictures. Threshold is approximately 1.15 can be seen.

4. CONCLUSION

This internship basically gave me idea about software development and business industry. After things i learned, i have gained vision about my future plans .

I learned so many things about image processing(especially OpenCV) from scratch. Topics that i became familiar:

-Image Segmentation.

-Erosion and Dilation.

-Finding and drawing contours.

-Face and Eyes Detection.

-Object Tracking.

-Facial Recognition.

Also i learned how to program with “C++” and making projects on “Visual Studio”. I learned some basics about “Qt” framework and how to make new applications on Qt.

Besides all of these topics, i have seen creating program process, teamwork importance, debugging and what developers environment really look like.

5. REFERENCES

http://www.miateknoloji.com.tr

https://docs.opencv.org

http://openbiometrics.org

https://cmake.org

https://www.qt.io

https://github.com