
# *BLM 102- Algoritmalar ve Programlama II*
# *2019-2020 Bahar Dönemi*

## *Görüntü İşleme Yöntemleri ile Üretim Hattı Analizi*

## *Proje Teslim Raporu*
## *10 Haziran 2020*

***Evrim Arda Kalafat, Arda Alhan***

***Öğretim Elemanı: Dr. Vecdi Emre Levent, emre.levent@fbu.edu.tr,*** 

# 1 Giriş

## 1.1 Projenin Amacı

Fabrikalarda üretim hatlarından çıkan ürünlerin analizi yüksek çözünürlüklü, yüksek FPS (Frame per Second) özellikli kameralar ile yapılmaktadır. Bir üretim hattından çıkan üç farklı tür ürünün, video kayıtı boyunca kaç'ar adet üretildiğinin tespit edileceği bir sistem geliştirilecektir. OpenCV (Open Computer Vision) kütüphanesi kullanılacaktır.

## 1.2 Proje Ekibi

Evrim Arda KALAFAT, 25.09.2001 yılında istanbulda doğdu. 2019 yılında Kadıköy Final Temel Lisesinden mezun oldu. Şu anda Fenerbahçe Üniversitesi Bilgisayar Mühendisliği bölümünde lisans eğitimi almakta. C ve C++ diline hakim, Başlangıç seviyesinde Pyhton biliyor. Programlama, yapay zeka ve siber güvenlik ile ilgileniyor.

Arda ALHAN, 18.05.2001 yılında doğdu. 2019 yılında Eyüp Anadolu Lisesi'nden mezun oldu. Şuan da Fenerbahçe Üniversitesi Bilgisayar Mühendisliği bölümünde Lisans eğitimi almakta. C, C#, Python dilleriyle ve Unity oyun motoruyla ilgileniyor.

# 2 Geliştirilen Uygulama

## 2.1 Kullanılan Araçlar

- Microsoft Visual Studio
- OpenCV Kütüphanesi

## 2.2 Tasarım

- İlk olarak kütüphanelerimizi ve renklerimizi tanımladık.

```
#include <opencv2/opencv.hpp>
#include<opencv2/core/core.hpp>
#include<opencv2/highgui/highgui.hpp>
#include<opencv2/imgproc/imgproc.hpp>
#include <opencv2/video/background_segm.hpp>
#include <opencv2/video/tracking.hpp>
#include <opencv2/video/video.hpp>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include<conio.h>

using namespace std;
using namespace cv;

//-------------------------RENKLER-----------------------------//
const Scalar COLOR_BLACK = Scalar(0.0, 0.0, 0.0);
const Scalar COLOR_WHITE = Scalar(255.0, 255.0, 255.0);
const Scalar COLOR_BLUE = Scalar(255.0, 0.0, 0.0);
const Scalar COLOR_GREEN = Scalar(0.0, 200.0, 0.0);
const Scalar COLOR_RED = Scalar(0.0, 0.0, 255.0);
//-------------------------RENKLER-----------------------------//
```

- Mainde VideoCapture kullanarak videoyu açtık. Frame1, Background ve foreground isimli üç Mat oluşturduk ve videonun ilk karesini yakaladık bu da videonun arkaplanı oluyor.

```
int main() {

    VideoCapture capture("C:\\Users\\evrim\\Desktop\\uretimHatti.mp4");

    Mat Frame1, Background, foreground;

    int nesnesayisi = 0, küçük = 0, büyük = 0, daire = 0, test = 0;

    if (!capture.isOpened()) {
        cout << "Error: Video acilamadi" << endl;
        return -1;
    }

    capture.read(Background);
```

- Video açıldıktan sonra, videoyu kare kare Frame1 e kaydetmeye başlıyor ve filtrelemeleri yapıyoruz. Grileştirme, bulanıklaştırma yaptıktan sonra her bir karenin ayrı ayrı arkaplan ile arasındaki farkı alıyoruz bu fark bize geçen nesneleri veriyor. Sonra treshold ekliyoruz ve nesneyi genişletiyoruz. Son olarak da MOG2 filtresini koyuyoruz.

```
39. while (capture.isOpened()) {
40.
41.         // Capture frame-by-frame
42.         capture >> Frame1;
43.         if (Frame1.empty())
44.             break;
45.         imshow("RAW", Frame1);
46.
47.
48.         //Frame1 ve arkaplan kopyalama
49.         Mat Frame1Copy = Frame1.clone();
50.         Mat BackgroundCopy = Background.clone();
51.
52.
53.         // Grilestirme
54.         cvtColor(Frame1Copy, Frame1Copy, CV_BGR2GRAY);
55.         cvtColor(BackgroundCopy, BackgroundCopy, CV_BGR2GRAY);
56.         // imshow("CVTCOLOR", Frame1Copy);
57.
58.
59.         //Blurlama
60.         GaussianBlur(Frame1Copy, Frame1Copy, Size(9, 9), 0);
61.         GaussianBlur(BackgroundCopy, BackgroundCopy, Size(9, 9), 0);
62.         // imshow("BLUR", Frame1Copy);
63.
64.
65.         // fark
66.         Mat imgDifference;
67.         absdiff(Frame1Copy, BackgroundCopy, imgDifference);
68.         // imshow("DIFFERENCE", imgDifference);
69.
70.
71.         // Threshold
72.         Mat imgThresh;
73.         threshold(imgDifference, imgThresh, 64, 255.0, CV_THRESH_BINARY);
74.         // imshow("THRESHOLD", imgThresh);
75.
76.
77.         // Genisletme
78.         Mat structuringElement9x9 = getStructuringElement(MORPH_RECT, Size(9, 9));
79.         dilate(imgThresh, imgThresh, structuringElement9x9);
80.
81.
82.         // BackgraundSubtracktorMOG2
83.         BackgroundSubtractorMOG2 mog;
84.         mog(imgThresh, foreground);
85.         // imshow("Fore", foreground);
```

- Sonra parametrelerini ayarlayarak nesneleri tespit etmemizi sağlayan SimpleBlobDetector'ı kullanarak nesneleri saydırıyoruz.

```
88.  //--------------PARAMETRELER----------------//
89.         SimpleBlobDetector::Params params;
90.
91.         params.minThreshold = 10;
92.         params.maxThreshold = 500;
93.
94.         params.filterByArea = true;
95.         params.minArea = 200;
96.
97.         params.filterByCircularity = false;
98.         params.minCircularity = 0.7;
99.
100.        params.filterByConvexity = false;
101.        params.minConvexity = 0.87;
102.
103.        params.filterByInertia = false;
104.        params.minInertiaRatio = 0.01;
105.       //--------------PARAMETRELER----------------//
106.
107.
108.       //--------------SimpleBlobDetection--------------//
109.        SimpleBlobDetector detector(params);
110.        vector<KeyPoint> keypoints;
111.
112.        detector.detect(foreground, keypoints);
113.
114.        for (int i = 0; i < keypoints.size(); i++)
115.        {
116.                    test++;
117.                    if (((15 < floor(keypoints[i].size)) && (floor(keypoints[i].size
    < 19))) && ((32 < floor(keypoints[i].pt.x)) && (floor(keypoints[i].pt.x) < 36)) &&
    ((216 < floor(keypoints[i].pt.y)) && (floor(keypoints[i].pt.y) < 220)))
118.                    {
119.                        nesnesayisi++;
120.                        küçük++;
121.
122.                    }
123.                    else if (((29 < floor(keypoints[i].size)) && (floor(keypoints[i]
    .size < 33))) && ((50 < floor(keypoints[i].pt.x)) && (floor(keypoints[i].pt.x) < 53
    )) && ((199 < floor(keypoints[i].pt.y)) && (floor(keypoints[i].pt.y) < 203)))
124.                    {
125.                        nesnesayisi++;
126.                        büyük++;
127.                    }
128.
129.                    else if (((36 < floor(keypoints[i].size)) && (floor(keypoints[i]
    .size < 40))) && ((96 < floor(keypoints[i].pt.x)) && (floor(keypoints[i].pt.x) < 10
    0)) && ((145 < floor(keypoints[i].pt.y)) && (floor(keypoints[i].pt.y) < 149)))
130.                    {
131.                        nesnesayisi++;
132.                        daire++;
133.                    }
134.                    cout << "Sekilin " << " Kordinat X: " << keypoints[i].pt.x << "
    Y: " << keypoints[i].pt.y << " Boyut:" << keypoints[i].size << "\n";
135.        }
136.
137.        Mat im_with_keypoints;
138.        drawKeypoints(imgThresh, keypoints, im_with_keypoints, COLOR_RED, DrawMatche
    sFlags::DRAW_RICH_KEYPOINTS);
139.        imshow("keypoints", im_with_keypoints);
140.        //--------------SimpleBlobDetection--------------//
```

- Son olarak videoyu bitiriyoruz ve nesnelerin sayılarını ekrana bastırıyoruz.

```
143.        //----------Bitis----------//
144.                char c = (char)waitKey(10);
145.                if (c == 27) break;
146.                //----------Bitis----------//
147.            }
148.
149.            cout << endl << "Toplam nesne sayisi: " << nesnesayisi << endl;
150.
151.            cout << "Kucuk nesne sayisi: " << küçük << endl;
152.            cout << "Sari nesne sayisi: " << büyük << endl;
153.            cout << "Yuvarlak nesne sayisi: " << daire << endl;
154.
155.            capture.release();
156.            return 0;
157.        }
```

# 3 Sonuçlar

Bu proje sayesinde bir C++ kütüphanesi olan OpenCV'yi iyice araştırmış ve incelemiş olduk. C++ diline hakimiyetimiz arttı.

Hazırlanan sunum video'su adresi: https://www.youtube.com/watch?v=82WAORLat_w

Dosyaların github adresi: https://github.com/rhgod/Goruntu_isleme_projesi