HUMAN DETECTION WITH QUADROTOR

By

151220102042 - Sinan KARACA

151220102024 - Muhammet Ali DEMİR

151220102026 - Çağlar DUMAN

151220102085 - Mustafa Said YILDIZ

151220112020 - Mert BÜYÜKARSLAN

An Engineering Synthesis and Design Project Report

Electrical Engineering Department

June, 2015

HUMAN DETECTION WITH QUADROTOR

By

151220102042 - Sinan KARACA

151220102024 - Muhammet Ali DEMİR

151220102026 - Çağlar DUMAN

151220102085 - Mustafa Said YILDIZ

151220112020 - Mert BÜYÜKARSLAN

An Engineering Synthesis and Design Project Report

Electrical Engineering Department

June, 2015

A Report Presented in Partial Fulfillment of
the Requirements for the Degree
Bachelor of Science in Electrical Engineering

OSMANGAZI UNIVERSITY

June, 2015

HUMAN DETECTION WITH QUADROTOR

By

151220102042 - Sinan KARACA

151220102024 - Muhammet Ali DEMİR

151220102026 - Çağlar DUMAN

151220102085 - Mustafa Said YILDIZ

151220112020 - Mert BÜYÜKARSLAN

has been approved

June, 2015

APPROVED:

_____

(Full Name of the Chair of the Committee), Chairperson

_____

(Full Name of the other member)

Supervisory Committee

ABSTRACT

In the present study, it is aimed to detect people in received data from onboard embedded camera of Parrot AR.Drone 2.0 quadcopter. In order to fulfill this purpose, video data is received from quadcopter by using Robot Operating System (ROS). Received video is separated into frames and in the meantime, frames are grouped as positive or negative images. In MATLAB environment, these images are used to train a detector where Histogram of Oriented Gradients (HOG) method takes place in the background. Constructed detector ('.xml' file) can always be added or embedded into other OpenCV or MATLAB image processing applications.

ÖZET

Bu çalışmada; Parrot AR.Drone 2.0 dört pervaneli robot helikopter üzerindeki kameradan alınan veriler üzerinden insan algılanması amaçlanmıştır. Bu amacı gerçekleştirmek üzere, Robot Operating System (ROS) vasıtasıyla video verisi alınır. Alınan video çerçevelere (frames) ayrılır ve bu aşamada pozitif ve negatif imge olarak ayrılarak gruplanır. MATLAB ortamında, arkaplanda Histogram of Oriented Gradients (HOG) methodu kullanılarak, önceden gruplanan imgeler üzerinden bir algılayıcı eğitilir. Oluşturulan algılayıcı ('.xml' dosyası), herhangi bir başka OpenCV yada MATLAB ortamında geliştirilmiş görüntü işleme uygulamasında kullanılabilir.

TABLE OF CONTENTS

INTRODUCTION

Quadcopter, also known as quadrotor, is a helicopter with four rotors. The rotors are directed upwards and they are placed in a square configuration with equal distance from the center of mass of the quadcopter. The quadcopter is controlled by adjusting the angular velocities of the rotors which are spun by electric motors. Quadcopter is a usual design for small unmanned aerial vehicles (UAV) because of the simple structure. This robot has been used in supervision, search and rescue, construction inspections and several other applications [1, 2, 3]. Regarding their complicate structure, the quadcopter is these days taken into consideration by many of the robotics researches and its complication causes special abilities which can be used in broad range of usages [4,5]. Quadcopter unmanned aerial vehicles (UAV) are used for supervision and reconnaissance by military and law enforcement agencies, as well as search and rescue missions in urban environments, which is a small UAV that can quietly hover in place and use camera to tracking people on the ground. Recently, many studies are being done for providing a proper detecting method which can track the persons in different situations with high efficiency. The main modules which are vital for image processing using a quadcopter are; wireless virtual interfaces, low bandwidth video compression [2].

The main object of implementing this work is to receive number of people in a specific disaster area, also making a new area of utilization for quadcopters.

IMPLEMENTATION

- Obtaining video datas from quadrotor embedded camera

First of all, in order to obtain video data from quadrotor we needed to communicate quadrotor and computer via Robot Operating System. The obtained video data were extracted to their frames in MATLAB environment. Totally, 3186 images were obtained from 9 videos, 412 of 3186 consist of humans are called positive images. Similarly, 2774 of 3186 do not consist of humans are called negative images. These numbers can be increased to get more trustworthy results and also these frames were obtained from a single indoor place, for other indoor or outdoor places, videos are needed to be recorded in each place.

Figure 1 - Obtaining video datas from quadrotor embedded camera

Figure 2 Negative Image

Figure 3 Negavite Image

Figure 4 Positive Image

Figure 5 Positive Image

- Training Image Labeler

The obtained positive images were prepared for 'trainCascadeObjectDetector' function of MATLAB. It means that humans in positive images were marked to specify rectangle region of interests (ROI) by using 'trainingImageLabeler' MATLAB toolbox.

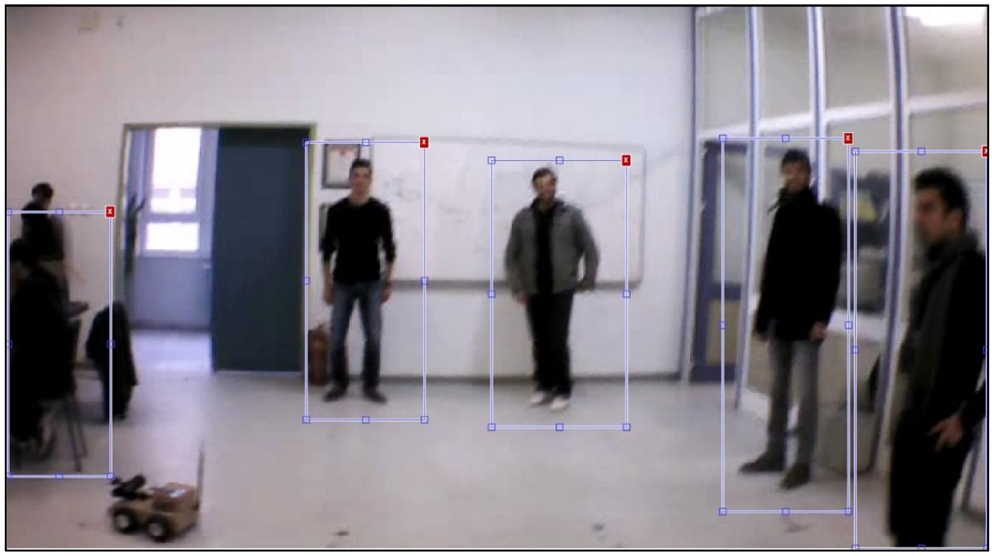


Figure 6 Example of ROIs – Screen Shot of Training Image Labeler

As it has seen in figure 6, the training image labeler is an easy way to label positive images that the 'trainCascadeObjectDetector' function uses to create a cascade classifier.

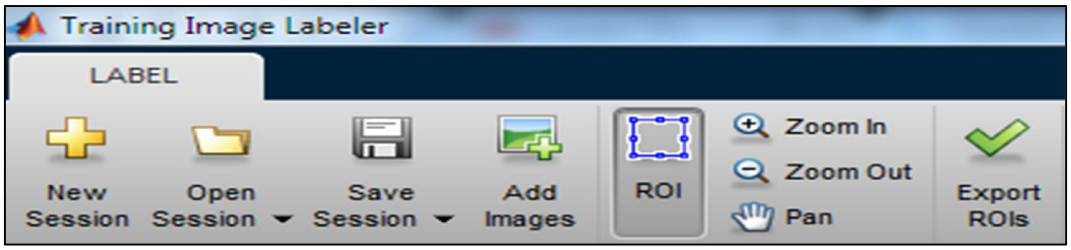


Figure 7 Training image labeler

All of the positive images were labeled and ROIs were exported. Finally, ROI file was exported as a MAT-file which contains the [x, y, width, height] informations of each sample. Also the app does not copy and resave images, so the stored path refers to the original image and folder. Shown in Figure 8 below.

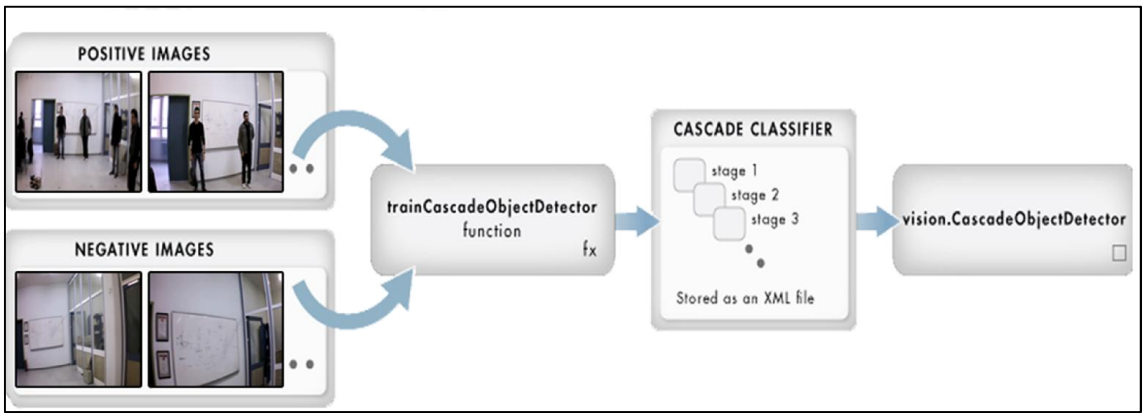
imageFilename	objectBoundingBoxes
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\032.png'	[316 131 147 229;19 126 7...
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\033.png'	[316 131 146 229;19 125 7...
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\034.png'	[316 131 147 229;18 126 7...
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\035.png'	[317 133 144 227;20 127 7...
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\036.png'	[310 131 152 230;21 126 7...
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\037.png'	3x4 double
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\038.png'	3x4 double
'C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\2\039.png'	3x4 double

Figure 8 ROI's.mat

- Train Cascade Object Detector

Cascade classifier training requires a set of positive samples and a set of negative images. Training image labeler outputs were used as positive samples. Instead of this 2 input , it has adjustable number of stages, feature type and other parameters. Such as 'FalseAlarmRate' , 'TruePositiveRate', 'NegativeSampleFactor' , 'ObjectTrainingSize' .Number of cascade stage was taken '7' value and false alarm rate value were determined after some test result. False alarm rate was adjusted so we didnt need to adjust true positive rate

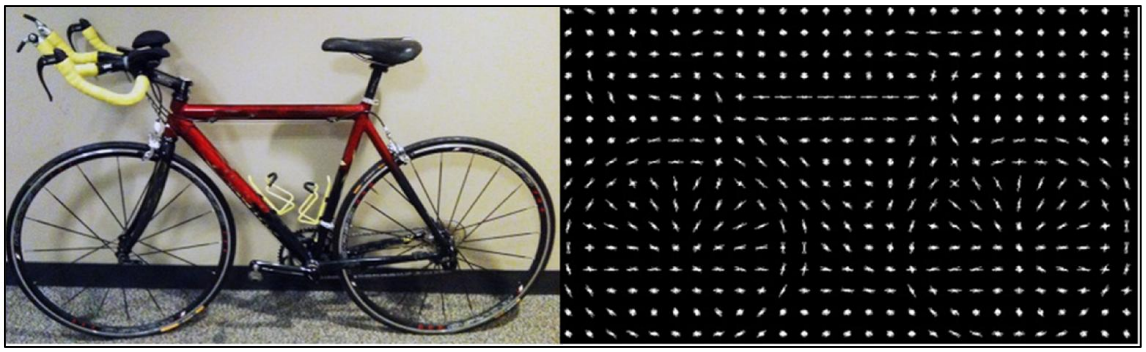
because they are inversely proportional to each other. Other parameters were taken 'default' value.



- Feature Type (Histogram of Gradients)

The essential thought behind the HOG descriptors is that local object apparance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled [6].

In addition, HOG method has an advantage is that it didn't take much longer then other feature types.



Şekil 1An example of HOG method applied on an image

- Test Results

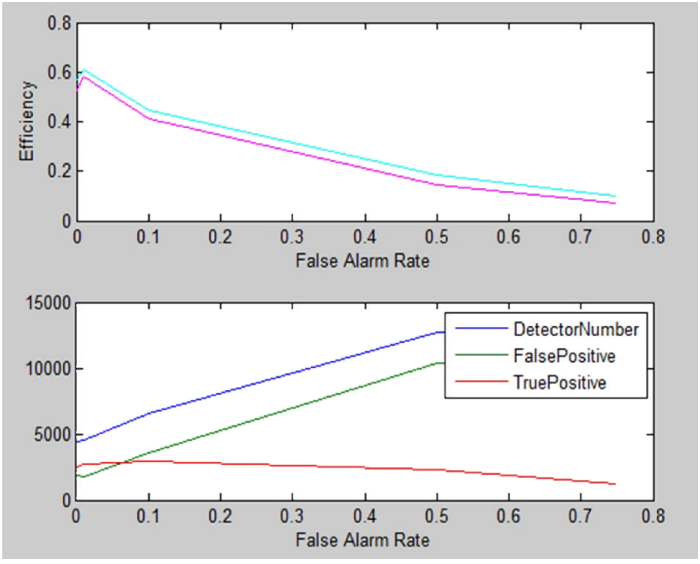
In this stage , first half of 412 positive images were labeled and trained , after that the created detectors were applied to second half of 412 positive images. Test was done with 5 different false alarm rate values , '0.0001' ,'0.01', '0.1' , '0.5' and '0.75'. As expected , most efficient false alarm rate value was '0.0001'. These processes based on HOG method, other feature types were applied but they took much longer than HOG method, so HOG method was prefered.

$$efficiency = \frac{True\ positive\ bounding\ boxes}{Total\ detected\ bounding\ boxes}$$

$$Total\ detected\ bounding\ boxes = True\ positive + False\ positive$$

Şekil 2True Postives - False positives

In the figure , red bounding boxes represent false positives and green bounding boxes represent true positives . These boxes were determined by comparison of detector's bounding boxes and ROIs which were labeled.



As it can be seen , efficency and true positive are inversely proportional with false alarm rate.

CONCLUSION

The application of image processing on a quadrotor is a challenging task in many ways. An optimal method in order to construct a detector file, is observed at the end of several experiments, and the best selection of parameters w.r.t desired results are performed. Since real time applications require high computation and communication speed, detector which is obtained as a result of this research can be implemented in another image processing application which is capable of running higher computational rates, and correspondingly capable of performing real time applications with quadrotor.

REFERENCES

[1] Lee, K. D., Nam, M. Y., Chung, K. Y., Lee, Y. H., & Kang, U. G. (2013), "Context and profile based cascade classifier for efficient people detection and safety care system", Multimedia Tools and Applications, 1-18.

[2] Wren, C. R., Azarbayejani, A., Darrell, T., & Pentland, A. P. (1997). Pfinder: Real-time tracking of the human body. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 19(7), 780-785.

[3] Fatan,M.,Lavi,B.,Vatankhah,A.," An Adaptive Neuro PID for Controlling the Altitudeof Quadcopter Robot",18th international conference on methods and models in automation and robotics, August 2013.

[4] Teppo Luukkonen, "Modelling and control of quadcopter," Independent research project in applied mathematics, Espoo, August 22, 2011.

[5] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," IEEE International Conference on Robotics and Automation, pp. 3277–3282, May 2009.

[6]

APPENDICES

- Appendix-1 : Obtaining video datas from quadrotor's embedded camera.
- Appendix-2 : Creating test .xml files.
- Appendix-3 : Testing .xml files' efficiency.
- Appendix-4 : Testing .xml file in real time video processing by webcam.

APPENDIX-1

```
clc;
close all;
clear all;
s = 2666;
% assigning the name of sample avi file to a variable
filename = '8.avi';

%reading a video file
mov = VideoReader(filename);

% Defining Output folder as 'snaps'
opFolder = fullfile(cd, 'snaps');
%if  not existing
if ~exist(opFolder, 'dir')
%make directory & execute as indicated in opfolder variable
mkdir(opFolder);
end

%getting no of frames
lastFrame = read(mov, inf);
numFrames = mov.NumberOfFrames;

%setting current status of number of frames written to zero

numFramesWritten = 0;

%for loop to traverse & process from frame '1' to 'last' frames
for t = 1 : 1 : numFrames
currFrame = read(mov, t);    %reading individual frames
opBaseFileName = sprintf('%d.png', s);
s = s+1;
opFullFileName = fullfile(opFolder, opBaseFileName);
imwrite(currFrame, opFullFileName, 'png');   %saving as 'png' file
%indicating the current progress of the file/frame written
progIndication = sprintf('Wrote frame %4d of %d.', t, numFrames);
disp(progIndication);
numFramesWritten = numFramesWritten + 1;
end      %end of 'for' loop
```

```
progIndication = sprintf('Wrote %d frames to folder
"%s"',numFramesWritten, opFolder);
disp(progIndication);
%End of the code
```

APPENDIX-2

```
% Creating .xml files

load( 'C:\Users\Sinan\Desktop\proje\proje\Roıs\session01_ROIs');
% Loading TrainingImageLabeler output as positiveInstances.(Positive
images ROIs)

negativeFolder =
fullfile('C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\1');
% Negative folder path

trainCascadeObjectDetector('detectors_0.0001.xml', positiveInstances,
negativeFolder,'FalseAlarmRate',0.01,'NumCascadeStages',7);
```

```
trainCascadeObjectDetector('detectors_0.01.xml', positiveInstances,
negativeFolder,'FalseAlarmRate',0.1,'NumCascadeStages',7);
trainCascadeObjectDetector('detectors_0.1.xml', positiveInstances,
negativeFolder,'FalseAlarmRate',0.5,'NumCascadeStages',7);
trainCascadeObjectDetector('detectors_0.5.xml', positiveInstances,
negativeFolder,'FalseAlarmRate',0.75,'NumCascadeStages',7);
trainCascadeObjectDetector('detectors_0.75.xml', positiveInstances,
negativeFolder,'FalseAlarmRate',0.75,'NumCascadeStages',7);
% Creating test .xml files by using trainCascadeObjectDetector
function

detector1 = vision.CascadeObjectDetector('detector_0.0001.xml');
detector2 = vision.CascadeObjectDetector('detector_0.01.xml');
detector3 = vision.CascadeObjectDetector('detector_0.1.xml');
detector4 = vision.CascadeObjectDetector('detector_0.5.xml');
detector5 = vision.CascadeObjectDetector('detector_0.75.xml');
% .xml files were selected as detector1,detector2,detector3……
```

APPENDIX-3

```
% Test Cascade
clear all
load('roi_son_test.mat');
klasor = [2 3 5 7 9];
z = 1;
count = 1;
true_positive = 0;
false_positive = 0;
% detector = vision.CascadeObjectDetector('detectors_0.0001.xml');
% detector = vision.CascadeObjectDetector('detector_0.01.xml');
% detector = vision.CascadeObjectDetector('detector_0.1.xml');
% detector = vision.CascadeObjectDetector('detector_0.5.xml');
```

```
detector = vision.CascadeObjectDetector('detector_0.75.xml');

for H = 1:5
    klasor_isim = int2str(klasor(H));
    filename1 =
['C:\Users\Sinan\Desktop\proje\ardrone_images\FRAMES\' klasor_isim ];
    for i =  1:166
        A = sprintf( '%3.3d' , i);
        Boundary = positiveInstances(i).objectBoundingBoxes;
        BoundaryNames = positiveInstances(i).imageFilename;
        x=size(Boundary);
        I = imread(BoundaryNames);
        B = I;
        I(1:360,1:640,1:3) = 255;
    for y=1:1:x(1)

I(Boundary(y+x(1)):Boundary(y+x(1))+Boundary(y+3*x(1)),Boundary(y):(Bo
undary(y)+Boundary(y+2*x(1))),:)=0;
    end
    bbox = step(detector, B);
    k = size (bbox);
    for f=1:1:k(1)
         O(z) =
sum(sum(I(bbox(f+k(1)):bbox(f+k(1))+bbox(f+3*k(1)),bbox(f):(bbox(f)+bb
ox(f+2*k(1))),1)));
         if( O(z) < ((bbox(f+3*k(1))* bbox(f+2*k(1)))*255*0.2))
             true_positive = true_positive +1;
         else
             false_positive = false_positive + 1;
         end
         z = z + 1;
    end
    end
end
% Firstly, selected ROIs pixels value were changed to '0' , it means
their colour turned to black. Outside of ROIs values were changed to
'255' , it means white. After compression between ROIs and bounding
boxes of detector , true positive and false positive numbers were
determined.
```

APPENDIX-4

```
% Real Time Video Processing by using detector.xml file
imaqreset;clear all;
NumberFrameDisplayPerSecond=10;
hFigure=figure(1);

% Set up web camera
vid = videoinput('winvideo', 1,'YUY2_640x480');
```

```
set(vid,'FramesPerTrigger',1);

% Go on forever until stopped
set(vid,'TriggerRepeat',Inf);

% Get a grayscale image
set(vid,'ReturnedColorSpace','RGB');
triggerconfig(vid, 'Manual');

% set up timer object
TimerData=timer('TimerFcn',
{@FrameRateDisplay,vid},'Period',1/NumberFrameDisplayPerSecond,'Execut
ionMode','fixedRate','BusyMode','drop');

% Start video and timer object
start(vid);
start(TimerData);

% We go on until the figure is closed
uiwait(hFigure);
Clean up everythin
stop(TimerData);
delete(TimerData);
stop(vid);
delete(vid);
imaqreset;
```

```
% This function is called by the timer to display one frame of the
figure(FrameRateDisplay)

function FrameRateDisplay(obj, event,vid)
persistent IM;
persistent im1;
persistent handlesRaw;
detector =
vision.CascadeObjectDetector('C:\Users\Sinan\Desktop\proje\ardrone_ima
ges\Detectorler\detertor_0.0001.xml');
```

```
trigger(vid);
IM=getdata(vid,1,'uint8');

if isempty(handlesRaw)
    bbox = step(detector, IM);

   im1 = insertObjectAnnotation(IM, 'rectangle',bbox,'Face');
   title('CurrentImage');
   handlesRaw=imagesc(im1);
else
   set(handlesRaw,'CData',im1);
   bbox = step(detector, IM);
   im1 = insertObjectAnnotation(IM, 'rectangle', bbox,'Face');
end
```