Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана (Национальный исследовательский университет)»   
(МГТУ им. Н.Э. Баумана)

Факультет Информатики и систем управления (ИУ)

Кафедра Информационные системы и телекоммуникации (ИУ-3)

**Отчёт**

**по лабораторной работе №2**

**«Разработка программы распознавания объектов на спутниковых**

**изображениях с использованием характерных признаков Хаара»**

**по курсу**

**«Цифровая обработка изображений»,**

**1 курс, 2-й семестр.**

Выполнил: студентка группы ИУ3-21М

Смирнова К.Д.,

Проверил: Алфимцев А.Н.

Москва, 2020

**Содержание**

[Цель работы и задание 3](#_Toc43075738)

[Теоретическая часть 3](#_Toc43075739)

[Выполнение 7](#_Toc43075740)

[Листинг программы 10](#_Toc43075741)

[Список использованных источников 13](#_Toc43075742)

# Цель работы и задание

Используя выборку цветных цифровых изображений запрограммировать алгоритм определения характерных признаков Хаара для объекта в соответствии с вариантом.

Вариант задания 2: Грузовая машина

# Теоретическая часть

Характерные признаки Хаара – это некоторая величина, вычисляемая для прямоугольной области.

Каскады из признаков обычно упоминаются как база для построения систем выделения сложных объектов, таких как лица, руки, или другие предметы. В большинстве статей этот подход неразрывно связывают с алгоритмом обучения AdaBoost [1].

Сам по себе каскад Хаара — это набор примитивов, для которых считается их свёртка с изображением. Используются самые простые примитивы, состоящих из прямоугольников и имеющих всего два уровня, +1 и -1 (рисунок 1). При этом каждый прямоугольник используется несколько раз разного размера. Под свёрткой тут подразумевается *s = X-Y*, где *Y* — сумма элементов изображения в тёмной области, а *X* — сумма элементов изображения в светлой области (можно так же брать *X/Y*, тогда будет устойчивость при изменении масштаба).

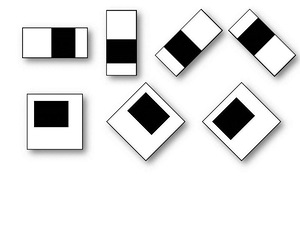
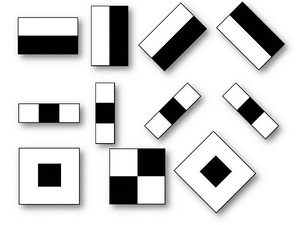


Рисунок 1 – Примеры признаков Хаара

Такие свёртки подчёркивают структурную информацию объекта. Например для центра лица человека будет всегда отрицательна свёртка на рисунке 2.



Рисунок 2 – Нахождение признака

Глаза будут темнее, чем область между ними, так же как область рта будет темнее чем лоб. Чем больше используется различных примитивов, тем точнее можно потом классифицировать объект. При этом если точная классификация не нужна — можно использовать меньшее количество примитивов.

Плюсом таких признаков является то, что каскады Хаара очень быстро считаются через интегральное представление изображений [2].

Интегральное представление позволяет быстро рассчитывать суммарную яркость произвольного прямоугольника на данном изображении, причем какой бы прямоугольник не был, время расчета неизменно.

Интегральное представление изображения – это матрица, совпадающая по размерам с исходным изображением. В каждом элементе ее хранится сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента. Элементы матрицы рассчитываются по следующей формуле:

image

Значение в точке *X,Y* матрицы (*II*), полученной из исходного изображения (*I*) это сумма всех точек в прямоугольнике (*0,0,X,Y*). Тогда интеграл по любому прямоугольнику (*ABCD*) в изображении представим как:

*SumOfRect(ABCD) = II(A) + II(С) — II(B) — II(D)*

Это даёт всего лишь 4 обращения к памяти и 3 математических действия для подсчёта суммы всех элементов прямоугольника вне зависимости от его размера. При расчёте других свёрток, отличных от свёрток примитивов Хаара, требуется количество действий пропорциональное квадрату размера примитива (если не рассчитывать через БПФ, что возможно не для любых паттернов).

Задача обнаружения характерных признаков на цифровом изображении выглядит так: имеется изображение, на котором есть искомые объекты. Оно представлено двумерной матрицей пикселей размером *Width\*Height*, в которой каждый пиксель имеет значение:

* от 0 до 255, если это черно-белое изображение;
* от (0, 0, 0) до (255, 255, 255), если это цветное изображение (компоненты R, G, B).

В результате своей работы, алгоритм должен определить признаки и пометить их. Поиск осуществляется в активной области изображения прямоугольными признаками Хаара, с помощью которых и описывается найденный объект

Иными словами, применительно к рисункам и фотографиям используется подход на основе сканирующего окна (scanning window): сканируется изображение окном поиска (так называемое, окно сканирования), а затем применяется классификатор к каждому положению.

Существуют модификации алгоритма для применения различных углов к прямоугольному признаку Хаара [3].

# Выполнение

В начале работы в программу загружаются изображения. Затем создается прямоугольный признак Хаара (рисунок 3).

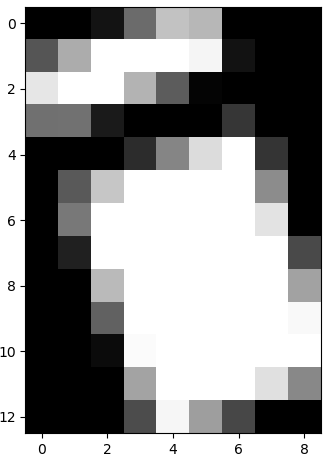


Рисунок 3 – Признак Хаара

Он состоит из трех областей: светлая, темная, светлая. По задумке, с помощью такого признака можно определить переднюю область кузова автомобиля, где капот, переднее стекло и крыша схожи с этим признаком (рисунок 4).



Рисунок 4 – Вид автомобиля сверху

Контрастность исходного изображения повышается. С помощью сканирующего окна изображение проверяется, и высчитывается сумма элементов в темной и светлой областях, после чего область с самым подходящим участком изображения признается автомобилем и обводится прямоугольником (рисунок 5).

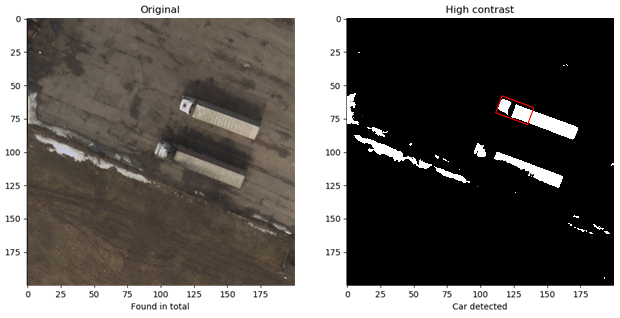


Рисунок 5 – Найденный участок

Угол наклона признака изменяется и со временем находится все больше автомобилей (рисунок 6).

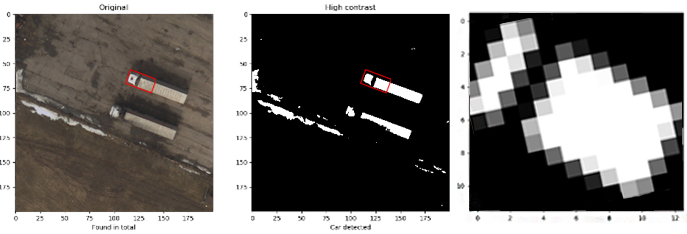


Рисунок 6 – Поиск и пометка автомобилей

В результате работы программы мы получаем изображения с отмеченным автомобилями, которые удалось определить по характерному признаку Хаара (рисунок 7). Не удается определить объекты тёмного цвета или те, что сливаются с фоном.

Листинг кода и результаты доступны на <https://github.com/karinoizerr/Digital-image-processing/tree/master/lab2>





На этом изображении видно, как влияет на обнаружение цвет прицепа.



На этом изображении видно, что такой признак Хаара присущ и автомобилям среднего размера и они обнаруживаются, если они достаточно светлые.

Рисунок 7 – Результаты нахождения автомобилей

# Листинг программы

# -\*- coding: utf-8 -\*-

"""

Created on Mon Jun 03 14:12:53 2020

@author: karina

"""

from skimage import draw, transform, io, color, exposure

import numpy as np

import matplotlib.pyplot as plt

import os

import cv2

import math

BRIGHT\_RECTANGLE = 255

DARK\_RECTANGLE = 50

def draw\_haar\_feature(w1, w2, w3, h, angle):

img = np.zeros((h, w1 + w2 + w3), dtype=np.uint8)

rr, cc = draw.rectangle((0, 0), extent=(h, w1), shape=img.shape)

img[rr, cc] = BRIGHT\_RECTANGLE

rr, cc = draw.rectangle((0, w1), extent=(h, w2), shape=img.shape)

img[rr, cc] = DARK\_RECTANGLE

if w3:

rr, cc = draw.rectangle((0, w1 + w2), extent=(h, w3), shape=img.shape)

img[rr, cc] = BRIGHT\_RECTANGLE

img = transform.rotate(img, 360 - angle, resize=True, preserve\_range=False)

return img

def create\_dataset():

dataset = []

os.chdir("Car")

img\_list = os.listdir()

for img in img\_list:

if os.path.isdir(img):

continue

image = io.imread(img)

img\_gray = color.rgb2gray(image)

img\_contrast = exposure.adjust\_sigmoid(img\_gray, cutoff=0.5, gain=60, inv=False)

dataset.append((image, img\_contrast))

os.chdir(os.getcwd() + "/../")

return dataset

def feature\_detection(img, haar\_feature\_img):

coord = (-1, -1)

max\_value = -1

size = haar\_feature\_img.shape

for x in range(img.shape[0] - size[0]):

for y in range(img.shape[1] - size[1]):

cur\_value = detect(img[x:x + size[0], y:y + size[1]], haar\_feature\_img)

if cur\_value > max\_value:

max\_value = cur\_value

coord = x, y

return coord, max\_value

def detect(img, haar\_feature\_img):

if img.shape != haar\_feature\_img.shape:

print("ERROR: SIZES NOT EQUAL")

raise IndexError

# этот параметр подбирается. есть граница, выше которой ничего определятся не будет

# если он слишком мал, то будет находится ненужное (по идее)

threshold = 35 \* haar\_feature\_img.shape[0] \* haar\_feature\_img.shape[1] / 255 # ~17.5

bright, dark = 0, 0

for x in range(img.shape[0]):

for y in range(img.shape[1]):

if haar\_feature\_img[x][y] == BRIGHT\_RECTANGLE / 255:

bright += img[x][y]

elif haar\_feature\_img[x][y] == DARK\_RECTANGLE / 255:

dark += img[x][y]

if bright - dark > threshold:

return bright - dark

return -1

def draw\_angled\_rec(x0, y0, width, height, angle, img, color):

\_angle = angle \* math.pi / 180.0

b = math.cos(\_angle) \* 0.5

a = math.sin(\_angle) \* 0.5

pt0 = (int(x0 - a \* height - b \* width),

int(y0 + b \* height - a \* width))

pt1 = (int(x0 + a \* height - b \* width),

int(y0 - b \* height - a \* width))

pt2 = (int(2 \* x0 - pt0[0]), int(2 \* y0 - pt0[1]))

pt3 = (int(2 \* x0 - pt1[0]), int(2 \* y0 - pt1[1]))

cv2.line(img, pt0, pt1, color, 1)

cv2.line(img, pt1, pt2, color, 1)

cv2.line(img, pt2, pt3, color, 1)

cv2.line(img, pt3, pt0, color, 1)

def highlight\_feature(img, c\_img, coord, size, angle, haar\_feature, n):

c\_img = color.gray2rgb(c\_img)

if coord != (-1, -1):

if size[0] > size[1]:

x0 = coord[1] + size[1] / 2

y0 = coord[0] + size[0] / 2

draw\_angled\_rec(x0, y0, size[0], size[1], angle, img, (255, 0, 0))

draw\_angled\_rec(x0, y0, size[0], size[1], angle, c\_img, (1, 0, 0))

else:

x0 = coord[1] + size[0] / 2

y0 = coord[0] + size[1] / 2

draw\_angled\_rec(x0, y0, size[1], size[0], angle, img, (255, 0, 0))

draw\_angled\_rec(x0, y0, size[1], size[0], angle, c\_img, (1, 0, 0))

label = "Car detected" if coord[0] != -1 else "No car detected"

fig, ax = plt.subplots(1, 3, squeeze=False)

ax[0][0].set(title='Original', xlabel='Found in total')

ax[0][0].imshow(img)

ax[0][1].set(title='High contrast', xlabel=label)

ax[0][1].imshow(c\_img)

ax[0][2].set(title='Haar-like feature', xlabel="Angle: {}".format(angle))

ax[0][2].imshow(haar\_feature)

io.show()

os.chdir("Car")

os.chdir("Detected")

img = img[:, :, ::-1]

cv2.imwrite(str(n) + " " + str(angle) + ".jpg", img)

os.chdir(os.getcwd() + "/../../")

if \_\_name\_\_ == '\_\_main\_\_':

car\_dataset = create\_dataset()

# диапазоны углов для каждой фотки, чтобы не проверять все углы (очень долго)

for n, image in enumerate(car\_dataset):

if n == 0:

start, end = -90, -5

elif n == 1:

start, end = -120, -80

elif n == 2:

start, end = -40, -20

else:

start, end = 0, 30

for i in range(start, end, 3):

haar\_feature = draw\_haar\_feature(w1=5, w2=5, w3=5, h=7, angle=i)

coordinates, value = feature\_detection(image[1], haar\_feature)

print('Image', n, ', Angle: ', i)

print(coordinates, value)

highlight\_feature(image[0], image[1], coordinates, haar\_feature.shape, i, haar\_feature, n)

# Список использованных источников

1. Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In Computational Learning Theory: Eurocolt ’95, pages 23–37. Springer-Verlag, 1995.
2. Paul Viola, Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. 2001.
3. A. L. C. Barczak, M. J. Johnson & C. H. Messom. Real-time Computation of Haar-like features at generic angles for detection algorithms. 2006.