Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа программной инженерии

КУРСОВАЯ РАБОТА

по дисциплине: «Методы и средства обработки видеоинформации»

на тему:

«Обработка видеоинформации с использованием telegram bot»

Выполнил

студент группы 3540904/10101 Томилин И. С.

Преподаватель Молодяков С. А.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_ г.

Санкт-Петербург 2022

Содержание

[Цель работы и постановка задачи 3](#__RefHeading___Toc728_1223764712)

[Создание бота 4](#__RefHeading___Toc730_1223764712)

[Пример работы 6](#__RefHeading___Toc732_1223764712)

[Использованные функции API и библиотеки 9](#__RefHeading___Toc736_1223764712)

[Вывод 10](#__RefHeading___Toc738_1223764712)

[Исходный код 11](#__RefHeading___Toc588_2292772786)

## Цель работы и постановка задачи

Цель курсовой работы, создать интерактивное приложение использущее интерфейс telegram бота для обработки изображений на языке python.

Приложение должно удовлетворять следующим условиям:

* принимать на вход изображение с описанием;
* в зависимости от описания **number** или **background**, выполнять манипуляции на сервере;
* при выборе **number**, бот возвращает изображение найденного номера и его распознанное текстовое представление или сообщает о том, что распознать номер не удалось;
* при выборе **background**, бот возвращает видеофайл с наложенным изображением на фон.

## Создание бота

Воспользуемся ботом **@BotFather**, для создания нового бота необходимо перейти в **@BotFather,** ввести **/newbot** и следовать дальнейшим инструкциям в интерактивном режиме. как показано на рисунке 1.

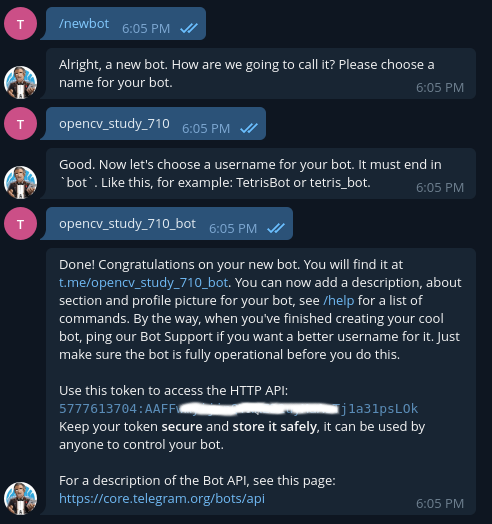


Рисунок 1 – создание нового бота.

После этого бот создан и можно проверить что бот создан, для этого перейти в чат бота **@opencv\_study\_710\_bot**

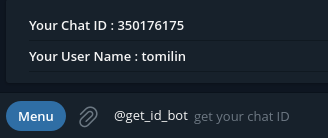
Все нижеописанные шаги выполнялись в дистрибутиве linux debian 11.

Первое что нужно сделать, добавить переменные окружения:

***export OPENCV\_BOT\_TOKEN=<token\_http\_api>*** # взять из @BotFather

***export OPENCV\_BOT\_CHAT\_ID=<your\_telegram\_chat\_id>***

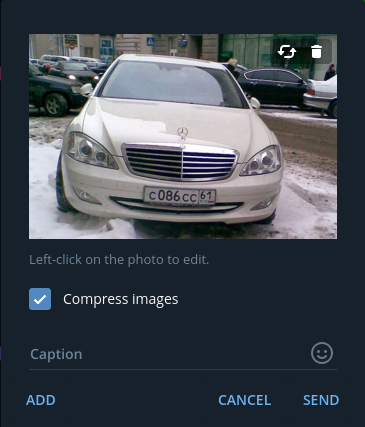
Получить chat\_id можно введя в любой чат телеграма строку **@get\_id\_bot**



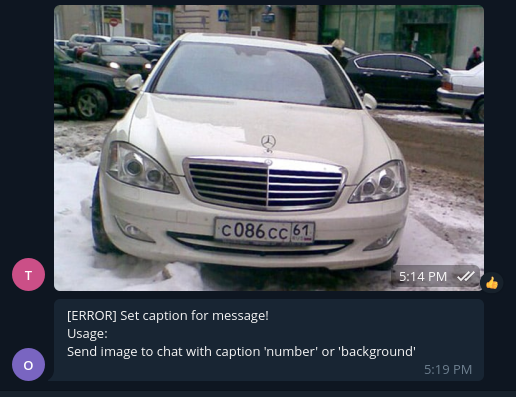
Для того, чтобы запустить бота, нужно скачать дистрибутив: <https://github.com/itomilin/opencv_play.git> перейти в скачанную директорию, установить зависимости выполнив команду **./run.sh** с правами root, выполнить команду **pip3 install -r requirements.txt** (на машине должен быть предустановлен **python3** и **pip3**). После того, как все зависимости были установлены, можно запустить бота выполнив команду **./start\_bot** на своей машине приложение.

## Пример работы

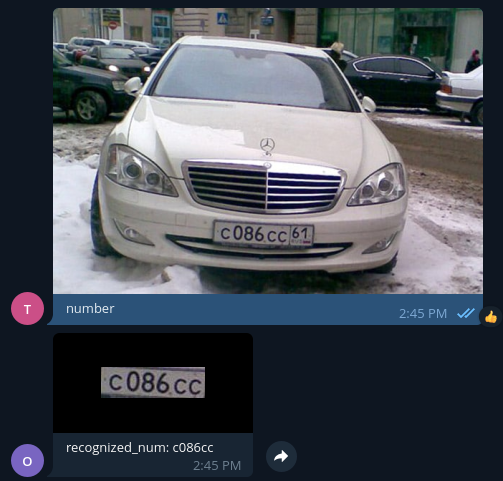
Изображения необходимо отправлять с выбранным **checkbox “compress images”** как на рисунке 2, посольку при отправке без выбранного checkbox тип файла будет распознан ботом как document и не будет обработан.

Рисунок 2 – правилный вариант загрузки изображения.

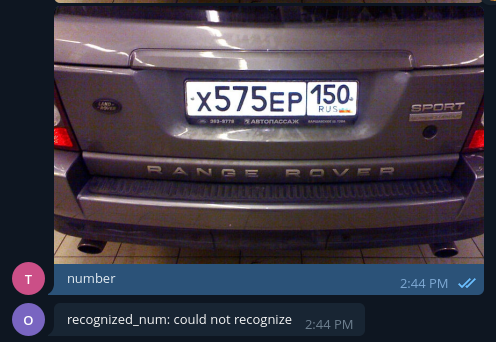
Если не задан caption, то бот сообщит о том, что его нужно ввести, рисунок 3.

Рисунок 3 – соощение бота, если не указан тип.

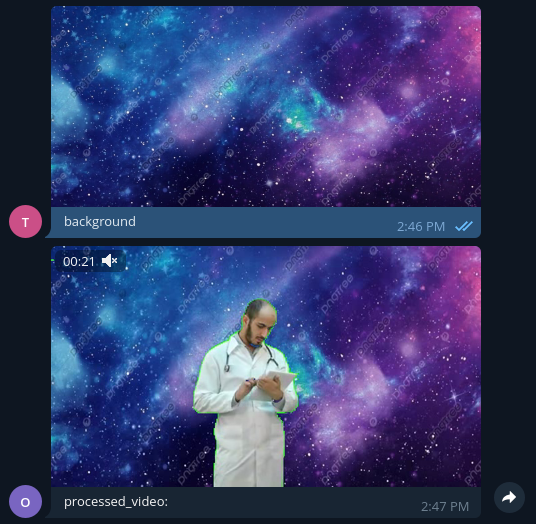
Добавим к caption **“number”** к отправляемому изображению и проверим результат обработки, рисунок 4.

Рисунок 4 – успешный результат распознавания номера.

Пример неудачного распознавания номера, рисунок 5.

Рисунок 5 – неудачное распознавание номера.

Теперь добавим второй вариант **“background”**, показанный на рисунке 6.

Рисунок 6 – кадр замененного фона у видео.

Как видно на рисунке, отправленное изображение стало фоном видео.

## Использованные функции API и библиотеки

В ходе курсовой работы была использована python библиотека для работы с telegram ботами **telepot –** <https://github.com/nickoala/telepot>.

Для того чтобы поставить библиотеку к себе ну машину нужно ввести команду: **pip3 install telepot**

Для запуска прослушивания входящих сообщений в чате, используется 2 функции.

Первая функция **Bot**, принемает на вход, HTTP\_API токен бота и возвращает рабочий объект для взаимодействия.

***bot = telepot.Bot( BOT\_TOKEN )***

Вторая функция **MessageLoop** принимает на вход экземпляр бота и функцию-обработчик входящих сообщений, данная функция запускается в фоновом режиме, поэтому необходимо сделать бесконечную задержку в программе.

***MessageLoop( bot, handle ).run\_as\_thread()***

Все входящие сообщения в чат, делятся на несколько типов: text, photo, video и.т.д, для обработки сообщений необходимо определить тип сообщения из поля входящего JSON и обработать его соответствующей функцией.

В данной работе производилась обработка только типа photo, использовались следующие функции для чтения и отправки файлов и сообщений.

Для отправки сообщения с сервера в чат, используется функция ***sendMessage( CHAT\_ID, STR )***, первый параметр id чата, второй параметр текст отправляемого сообщения.

Для отправки фотографии с сервера в чат, используется функция

***sendPhoto( CHAT\_ID, PATH\_TO\_IMG, caption=STR)***, первый параметр id чата, второй параметр путь до изображения, третий параметр опциональный, текст к отправляемому изображению.

Для отправки видеофайла с сервера в чат, используется функция

***sendVideo( CHAT\_ID, PATH\_TO\_VIDEO, caption=STR)***, первый параметр id чата, второй параметр путь до видеофайла, третий параметр опциональный, текст к отправляемому видео.

Для загрузки изображения отправленное в чат, используется функция ***download\_file( CHAT\_FILE\_ID, PATH\_TO\_SAVE)***, первый параметр id файла из чата, второй параметр путь на сервере для сохранения изображения.

Полный пример бота приведет в приложении.

## Вывод

В ходе выполнения курсовой работы было создано приложение, которое полностью соответствует всем поставленным задачам.

Проект доступен на github по ссылке: <https://github.com/itomilin/opencv_play/tree/master/course>

## Исходный код

Телеграм бот:

#!/usr/bin/env python3

import sys

import os

import time

import telepot

from telepot.loop import MessageLoop

from pprint import pprint

from number\_recognition import number\_recognition

from background\_remover import change\_background

BOT\_TOKEN = os.environ['OPENCV\_BOT\_TOKEN']

CHAT\_ID = int( os.environ['OPENCV\_BOT\_CHAT\_ID'] )

def usage():

return "Usage:\nSend image to chat with caption 'number' or 'background'"

def handle( msg ):

content\_type, chat\_type, chat\_id = telepot.glance( msg )

if content\_type == 'photo':

case = None

try:

case = msg['caption']

pprint( msg['caption'] )

except KeyError:

err\_msg = f"[ERROR] Set caption for message!\n{usage()}"

print( err\_msg )

bot.sendMessage( CHAT\_ID, err\_msg )

return

img\_path = f"./chat\_files/{msg['photo'][-1]['file\_unique\_id']}.png"

\_, file\_name = os.path.split( img\_path )

bot.download\_file( msg['photo'][-1]['file\_id'], img\_path )

if case == "number":

text\_number, status = number\_recognition( img\_path )

if status:

bot.sendPhoto( CHAT\_ID,

photo=open( f"./processed\_files/{file\_name}", 'rb'),

caption=f'recognized\_num: {text\_number}')

else:

bot.sendMessage( CHAT\_ID, f'recognized\_num: {text\_number}' )

elif case == "background":

change\_background( img\_path )

bot.sendVideo( CHAT\_ID,

video=open(f"./processed\_files/{file\_name.split('.')[0]}.mp4", 'rb'),

caption=f'processed\_video:')

else:

print( "Wrong case, try again!" )

return

print( "!!processing done!!" )

elif content\_type == 'video':

pass

elif content\_type == 'text':

pass

else:

print( "Type is not handled" )

bot = telepot.Bot( BOT\_TOKEN )

MessageLoop( bot, handle ).run\_as\_thread()

print( usage() )

# Keep the program running.

while True:

time.sleep( 1200 )

Распознавание номеров:

import os

import cv2

import pytesseract

from imutils import contours

def number\_recognition( img ):

\_, file\_name = os.path.split( img )

car\_number\_img = None

msg = "could not recognize"

status = 0

image = cv2.imread( img )

height, width = image.shape[:2]

image\_gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY) # преобразовать изображение в

#cv2.imshow("Gray Image", image\_gray) #вывести на экран для проверки

#cv2.waitKey()

# пороговая обработка для выделения контуров

threshold = cv2.threshold(image\_gray, 0, 255, cv2.THRESH\_OTSU)[1]

#cv2.imshow("Threshold", threshold) #вывести на экран для проверки

#cv2.waitKey()

# Найдем все контуры

contourss = cv2.findContours(threshold, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contourss, \_ = contours.sort\_contours(contourss[0]) # отсортируем их

chars = set('0123456789,') # создадим множество цифр для отсечения "мусорных" текстов

for c in contourss: # переберем все контуры

area = cv2.contourArea(c) # найдем площадь контура

x, y, w, h = cv2.boundingRect(c)

if area > 500: # если площадь соизмерима с номером

img = image[y:y+h, x:x+w] # получим этот контур из исходного изображения

# cv2.waitKey()

result = pytesseract.image\_to\_string(img, lang='rus+eng')

if len(result) > 7:

car\_number\_img = img.copy()

chars = "-`‘\/')( ,."

for char in chars:

result = result.replace( char, "" )

msg = result

cv2.imwrite( f"./processed\_files/{file\_name}", car\_number\_img )

status = 1

#cv2.imshow( "car\_n", car\_number\_img )

#cv2.waitKey()

return msg, status

Замена фона на видео

import cv2

import numpy as np

import os

def change\_background( img\_path ):

cap = cv2.VideoCapture( '../imgs/green\_screen\_360.mp4' )

\_, file\_name = os.path.split( img\_path )

drop\_ext = file\_name.split('.')[0]

bg = cv2.imread( img\_path )

bg = cv2.resize(bg, (640, 360))

l\_green = np.array([0, 0, 45])

u\_green = np.array([177, 216, 255])

frame\_width = int(cap.get(3))

frame\_height = int(cap.get(4))

out = cv2.VideoWriter( f'./processed\_files/{drop\_ext}.mp4',

0x7634706d,

20.0,

(frame\_width, frame\_height) )

while True:

ret, frame = cap.read()

if not ret:

break

if cv2.waitKey(25) & 0xFF == ord('q'):

break

#frame = cv2.resize(frame, (640, 360))

hsv = cv2.cvtColor( frame, cv2.COLOR\_BGR2HSV )

mask = cv2.inRange(hsv, l\_green, u\_green)

frame[mask == 0] = bg[mask == 0]

out.write(frame)

#cv2.imshow("video2", frame)

cap.release()

out.release()