Таблица 2.

Результаты тестирования программы

Вид теста	Содержание теста	Результат теста
Организация входных данных	1. Дата рождения должна вводиться в формате дд/мм/гггг. 2. Для определения совместимости данные указываются с помощью выбора из специального списка. 3. Адрес электронной почты должен состоять из двух частей, разделённых символом «@». Левая часть указывает имя почтового ящика. Правая часть адреса указывает доменное имя того сервера, на котором расположен почтовый ящик	Выполнен
Организация выходных данных	1. Данные будут выведены на экран в специальном окне в соответствии с запросом пользователя. 2. Персональный гороскоп будет приходить на почту ежедневно в виде сообщения	Выполнен

Таким образом, разрабатываемый проект «Гороскоп» может быть популярен среди большого круга людей молодого возраста, в особенности женского пола. Приложение имеет интересный дизайн и удобный функционал, что весьма важно для будущих пользователей, судя по проведенному опросу. Проект «Гороскоп» имеет предпосылки для дальнейшего развития, а именно планируется добавить определение знака зодиака по дате рождения, изменение интерфейса зависимости от времени суток, создание мобильной версии приложения.

1. Pyinstaller [Электронный ресурс]. URL: <http://www.pyinstaller.org/> (дата обращения: 15.05.2020).
2. Бэрри Пол. Изучаем программирование на Python. М.: Эксмо, 2017, 624 с.
3. Васева Е.С., Бужинская Н.В. Использование средств визуализации в организации проектной деятельности. Учебно-методическое пособие. Российский государственный профессионально-педагогический университет. Нижний Тагил, 2020.
4. Векторные иконки [Электронный ресурс]. URL: www.flaticon.com (дата обращения: 15.05.2020).
5. ГОСТ Р 56939-2016. Защита информации. Разработка безопасного программного обеспечения. Общие требования [Электронный ресурс]. URL: <http://docs.cntd.ru/document/1200135525> (дата обращения: 24.05.2020).
6. Доусон Майкл. Программируем на Python. М.: Прогресс-книга, 416 с.
7. Каюмова А. В. Визуальное моделирование систем в StarUML: учебное пособие. Казань. Казанский федеральный университет, 2013. 104с.
8. Метод диаграммы «Исикавы» [Электронный ресурс]. URL: <https://www.inventech.ru/pub/methods/metod-0019/> (дата обращения: 15.05.2020).
9. РД 50-34.698-90. Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов [Электронный ресурс]. URL: <http://docs.cntd.ru/document/1200006978> (дата обращения: 24.05.2020).
10. Что такое PyQT? [Электронный ресурс]. URL: <https://riverbankcomputing.com/software/> (дата обращения: 31.05.2020).

Магамедова Д.М.

OpenCV – инструмент компьютерного зрения

ФГБОУ ВО «Чеченский государственный университет»
(Россия, Грозный)

doi: 10.18411/lj-07-2020-68

idsp: ljjournal-07-2020-68

Аннотация

В современном мире тематика компьютерного зрения и область разработки интеллектуальных технологий перестали быть прерогативой сугубо научного сообщества. Очевидны значительные успехи IT-разработчиков, нейробиологов,

психологов, физиков и прочих специалистов, долгое время ограниченных рамками отдельных научных дисциплин, а теперь объединённых в контексте междисциплинарности. Важнейшими источниками информации о внешнем мире для робота являются его оптические датчики и камеры. После получения изображения необходима его обработка для анализа обстановки и принятия решения. В данной статье акцентируется внимание на методах библиотеки OpenCV и применения их в приложениях компьютерного зрения.

Ключевые слова: компьютерное зрение, OpenCV, программирование, методы, библиотеки.

Abstract

In the modern world, the subject of computer vision and the field of development of intelligent technologies have ceased to be the prerogative of the purely scientific community. It developers, neuroscientists, psychologists, physicists, and other specialists who have long been confined to separate scientific disciplines, but are now United in the context of interdisciplinarity, are clearly making significant progress. The most important sources of information about the outside world for a robot are its optical sensors and cameras. After receiving the image, you need to process it to analyze the situation and make a decision. This article focuses on the methods of the OpenCV library and their application in computer vision applications.

Keywords: computer vision, OpenCV, programming, methods, libraries.

Компьютерное зрение объединяет множество методов работы с изображениями. При функционировании робота предполагается, что видеoinформация с камер обрабатывается какой-либо программой, запущенной на контроллере. Чтобы не писать код с нуля, программист может воспользоваться готовыми программными решениями. На текущий момент существует множество готовых библиотек компьютерного зрения: Matrox Imaging Library, Camellia Library, Halcon, VXL, IVT, OpenCV и другие.

В данной работе рассмотрим библиотеку OpenCV. Она предоставляется бесплатно как для учебных целей, так и для коммерческого использования.

OpenCV (от англ. Open Source Computer Vision Library) — это библиотека алгоритмов компьютерного зрения с открытым исходным кодом.

Типичными задачами компьютерного зрения являются обнаружение, отслеживание и классификация объектов. В OpenCV реализовано очень большое количество алгоритмов для обработки изображений и компьютерного зрения. Некоторые алгоритмы реализованы в виде отдельных классов, а большинство — в виде статических методов какого-либо класса.

OpenCV написана на оптимизированном C/C++ в 1999 году фирмой Intel, поддерживает интерфейсы C, C++, Python, Java и включает в себя реализации свыше 2500 алгоритмов. Библиотека поддерживается платформами Windows, Linux и MacOS [1]. Помимо стандартных функций обработки изображений (фильтрация, размытие, геометрические преобразования и т.д.) данный SDK позволяет решать более сложные задачи, к которым относятся обнаружение объекта на фотографии и его «узнавание» [2].

Рассмотрим в нашей работе следующие понятия:

- Компьютерное зрение;
- Основы OpenCV;
- Обнаружение изображения с использованием OpenCV.

Компьютерное зрение – это междисциплинарная область, которая касается того, как использовать компьютеры для распознавания объектов на цифровых изображениях или видео. Идея здесь состоит в том, чтобы автоматизировать задачи, которые могут выполнять, например, зрительные системы человека. Таким образом, компьютер

должен иметь возможность распознавать такие объекты, как лицо человека, жесты рук в сурдопереводе, фонарный столб или даже статуя [3]. **Например**, многие из нас могут определить по изображению объект на фотографии. Но для компьютера данная информация недоступна. Компьютер считывает любое изображение в диапазоне значений от 0 до 255. Для любого цветного изображения есть 3 основных канала – красный, зеленый и синий. Матрица формируется для каждого основного цвета, и позже эти матрицы объединяются, чтобы обеспечить значение пикселя для отдельных цветов. Каждый элемент матрицы предоставляет данные, относящиеся к интенсивности яркости пикселя. Для черно-белого изображения используется только один канал.

Рассмотрим различные концепции: от загрузки изображений до изменения их размера [4].

Загрузка изображения с использованием OpenCV:

1. `import cv2`
2. `# цветное изображение`
3. `img = cv2.imread("city.jpg",1)`
4. `# черно-белый (серая шкала)`
5. `img_1 = cv2.imread("city.jpg",0)`

Как видно из приведенного выше фрагмента кода, первым требованием является импорт модуля OpenCV. Причем, код читает цветное и черно-белое изображение с помощью модуля `imread`. Если второй параметр метода `imread` равен 0, это означает, что импортируемое изображение является черно-белым. Название изображения здесь – «city».

Подфункцию `shape` используют для вывода формы изображения:

- `import cv2`
- `# черно-белый (серая шкала)`
- `img = cv2.imread("city.jpg",0)`
- `print(img.shape)`

Форма изображения – это форма NumPy. Главной особенностью NumPy является объект `array` (массив). Матрица кода состоит из 768 строк и 1024 столбцов.

Отображение изображения (пример):

- `import cv2`
- `# черно-белый (серая шкала)`
- `img = cv2.imread("city.jpg",0)`
- `cv2.imshow("city", img)`
- `cv2.waitKey(0)`
- `# cv2.waitKey(2000)`
- `cv2.destroyAllWindows()`

Первым шагом импортируем изображение с помощью `imread`. Для отображения изображений в окне используем функцию `imshow`. Данная функция имеет 2 параметра, которые представляют собой имя окна и отображаемый объект изображения. Метод `waitKey` делает окно статическим, то есть оно открыто пока пользователь не нажмет любую клавишу. Переданный ему параметр – это время в миллисекундах. Метод в последней строке кода – `destroyAllWindows`, закрывает окно на основе параметра `waitForKey`.

Изменение размера изображения аналогично (пример):

- `import cv2`
- `# черно-белый (серая шкала)`
- `img = cv2.imread("city.jpg",0)`
- `resized_image = cv2.resize(img, (650,500))`
- `cv2.imshow("city", resized_image)`
- `cv2.waitKey(0)`

— `cv2.destroyAllWindows()`

Функция изменения размера используется для изменения размера изображения до желаемой формы. Параметром является форма нового измененного изображения. Для изменения объект изображения `img` меняем на `resized_image`.

Рассмотрим методы, применяемые при обнаружении лица с использованием OpenCV

Для этого, составим следующий алгоритм действий:

Первое – загрузка изображения, второе – использование OpenCV, который будет читать изображение и файл функций. Именно, на этом этапе в первичных точках данных есть массивы NumPy. Находим значения строк и столбцов грани NumPy ndarray. Это массив с координатами прямоугольника лица. И последний, третий пункт в нашем алгоритме – это отображение изображения с прямоугольной рамкой лица.

Для первого шага создаем объект CascadeClassifier для извлечения особенностей лица. Параметр здесь указывает путь к XML-файлу, который содержит элементы лица. Следующим шагом будет чтение изображения с лицом на нем и преобразование его в черно-белое изображение с помощью COLOR_BGR2GREY. Затем находим координаты для изображения (прямоугольника лица). Это делается с помощью метода DetectMultiScale. Функция ScaleFactor используется для уменьшения значения формы на 5%, пока лицо не будет найдено. Итак, в целом – чем меньше значение, тем выше точность и лицо будет отображено в окне.

Определяем метод для создания прямоугольника, используя `cv2.rectangle`, передавая такие параметры, как объект изображения, значения RGB контура поля и ширину прямоугольника.

```
— import cv2
— # Создать объект CascadeClassifier
— face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
— # Чтение изображения в заданном формате
— img = cv2.imread("photo.jpg")
— # Чтение изображения как изображения в оттенках серого
— gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
— # Поиск координат изображения
— faces = face_cascade.detectMultiScale(gray_img, scaleFactor = 1.05, minNeighbors=5)
— for x,y,w,h in faces:
—     img = cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 3)
—     resized = cv2.resize(img, (int(img.shape[1]/7), int(img.shape[0]/7)))
—     cv2.imshow("Gray", resized)
—     cv2.waitKey(0)
—     cv2.destroyAllWindows()
```

За последние годы OpenCV сильно расширилась и стала в некоторой степени «тяжеловесной». В связи с этим, разными группами программистов создаются более «легкие» библиотеки, основанные на OpenCV. Примеры: SimpleCV, liuliu CCV и другие.

Разберём несколько примеров использования библиотеки OpenCV.

Пример 1: Вывод текста на экран

```
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <iostream>
using namespace cv;
using namespace std;
```

```

int main(int argc, char** argv)
{
    int w = 600, h = 200; //размеры картинки
    Point pt = Point(100, 100); //точка, куда выводим
    IplImage* hw = cvCreateImage(cvSize(w, h), 8, 3); //создаём картинку
    cvSet(hw, cvScalar(0, 0, 0)); // заливаем чёрным цветом
    CvFont font;
    cvInitFont(&font, CV_FONT_HERSHEY_COMPLEX, 1.0, 1.0, 0, 1,
CV_AA);
    // инициализируем шрифт
    cvPutText(hw, "OpenCV for sitev.ru", pt, &font, CV_RGB(255, 0, 0));
    // выводим на картинку текст
    namedWindow("sitev.ru", WINDOW_AUTOSIZE); // создаём окошко
    cvShowImage("sitev.ru", hw); // выводим картинку в окошко
    waitKey(0); // ждём любую клавишу
    // освобождаем ресурсы
    cvReleaseImage(&hw);
    cvDestroyWindow("sitev.ru");
    return 0;
}

```

Приложение создаёт пустую картинку с текстом и выводит ее на экран.

Пример 2. Вывод видео на экран

```

#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <iostream>
using namespace cv;
using namespace std;
IplImage* frame = 0;
int main(int argc, char* argv[])
{
    char* filename = "test.avi";
    namedWindow("sitev.ru", CV_WINDOW_AUTOSIZE);
    CvCapture* capture = cvCreateFileCapture(filename);
    // инициализация захвата видео из файла
    while (1) {
        frame = cvQueryFrame(capture); // получить очередной кадр
        if (!frame) break;
        cvShowImage("sitev.ru", frame); // показать кадр
        char c = waitKey(33);
        if (c == 27) break; // выходим по ESC
    }
    // освобождаем ресурсы
    cvReleaseCapture(&capture);
    cvDestroyWindow("sitev.ru");
    return 0;
}

```

В OpenCV можно работать с прямоугольной областью изображения Region Of Interest (ROI).

Пример 3: Посмотрим на примере

```

#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>

```

```

#include <iostream>
using namespace cv;
using namespace std;
int main(int argc, char* argv[])
{
    char* filename = "Diana.jpg";
    IplImage* image = cvLoadImage(filename, 1);
    namedWindow("dok.ru", CV_WINDOW_AUTOSIZE);
    namedWindow("ROI", CV_WINDOW_AUTOSIZE);
    // задаём ROI
    int x = 250;
    int y = 250;
    int w = 100;
    int h = 100;
    int add = 100; // добавочная величина
    cvShowImage("sitev.ru", image);
    cvSetImageROI(image, cvRect(x, y, w, h)); // устанавливаем ROI
    cvAddS(image, cvScalar(add), image); // добавить скалярную величину
    cvResetImageROI(image); // сбрасываем ROI
    cvShowImage("ROI", image); // показываем изображение
    waitKey(0);
    // освобождаем ресурсы
    cvReleaseImage(&image);
    cvDestroyAllWindows();
    return 0;
}

```

Функция `cvAddS` добавляет к пикселям изображения некую скалярную величину, выделяя цветом прямоугольник ROI. С этим понятно.

Пример 4: Захват с помощью OpenCV видео с камеры:

```

#define _CRT_SECURE_NO_WARNINGS
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <iostream>
using namespace cv;
using namespace std;
int main(int argc, char* argv[])
{
    VideoCapture cap(0); //открыть камеру по-умолчанию
    if (cap.isOpened() == false) {
        cout << "Cannot open the video camera" << endl;
        cin.get();
        return -1;
    }
    // ширина кадра
    double w = cap.get(CV_CAP_PROP_FRAME_WIDTH);
    // высота кадра
    double h = cap.get(CV_CAP_PROP_FRAME_HEIGHT);
    cout << "Resolution of the video : " << w << " x " << h << endl;
    string window_name = "My Camera Feed";
    namedWindow(window_name);
    while (true)
    {

```

```
Mat frame;
bool success = cap.read(frame); // чтение кадр
if (success == false) {
    cout << "Video camera is disconnected" << endl;
    cin.get();
    break;
}
imshow(window_name, frame); // показать кадр
if (waitKey(10) == 27) {
    cout << "Esc key is pressed by user. Stoppig the video" << endl;
    break;
}
}
return 0;
```

Помимо приведенных методик в OpenCV имеются и другие алгоритмы для детектирования и распознавания, а также набор алгоритмов, относящихся к машинному обучению, таких как метод k ближайших соседей, нейронные сети, метод опорных векторов и т.д.

В целом, OpenCV предоставляет инструментарий, достаточный для решения подавляющего большинства задач компьютерного зрения. Если алгоритм не имеется в составе SDK, то, как правило, он может быть без проблем запрограммирован. Кроме того, существует множество авторских версий алгоритмов, написанных пользователями на основе OpenCV.

1. Васильев А.И. Программное и алгоритмическое обеспечение систем компьютерного видения с несколькими полями зрения. Дисс. ... канд. Физ.-матем. наук – Москва – 2013. – 116 с.
2. Прохоренок Н.А. OpenCV и Java. Обработка изображений и компьютерное зрение. – СПб.: БХВ-Петербург, 2018. – 320 с.
3. Орлова Ю.А. Модели и методы информационной поддержки коммуникаций людей с ограниченными возможностями. Дисс. ... докт. техн. наук – Волгоград – 2016. – 346 с.
4. J.Solem. Programming Computer Vision with Python. O'Reilly, 2012. – 352 p.

Макаревич И.В., Костикова А.В.

Цифровизация химической промышленности: на пути к искусственному интеллекту

*Волгоградский государственный технический университет
(Россия, Волгоград)*

doi: 10.18411/lj-07-2020-69

idsp: ljournal-07-2020-69

Аннотация

В период цифровой трансформации крупные промышленные предприятия уделяют большое внимание разработке и внедрению IT-технологий, позволяющих улучшить и обезопасить технологические процессы производства. Предлагаемая IT-технология с применением искусственного интеллекта на химическом предприятии усовершенствует контроль за техническими показателями и состоянием окружающей среды.

Ключевые слова: IT-технологии, ИИ, химическое предприятие