УДК 004.932

ОСНОВНЫЕ МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ В OPENCV

ШАВКАТ кызы Аида
магистрант
Ошский государственный университет
КАЛБАЕВА Динара Исматиллаевна
преподаватель
Ошский технологический университет им. академика М.М. Адышева
г. Ош, Кыргызстан

В данной статье рассматриваются методы и подходы к обработке изображений с использованием библиотеки OpenCV языка Python. Приводится обзор теоретических основ цифровой обработки изображений и описываются практические инструменты библиотеки, обеспечивающие решение задач компьютерного зрения, включая фильтрацию, изменение цветовых пространств, выделение контуров и морфологические операции. Работа ориентирована на исследователей и разработчиков, занимающихся автоматизацией обработки визуальных данных и внедрением технологий компьютерного зрения в различные сферы деятельности.

Ключевые слова: обработка изображений, компьютерное зрение, фильтрация, визуализация, методы.

Введение. Современные информационные технологии активно используют методы компьютерного зрения для автоматизации анализа визуальных данных. Обработка изображений играет ключевую роль в таких областях, как медицина, промышленная автоматизация, робототехника, системы безопасности и искусственный интеллект. Одним из наиболее востребованных инструментов для решения подобных задач является библиотека OpenCV (Open Source Computer Vision Library) – открытая программная платформа, предоставляющая широкий набор функций для обработки изображений и видео [1; 2].

Язык Python стал основным средством для работы с OpenCV благодаря простоте синтаксиса, обширной экосистеме научных библиотек (NumPy, SciPy, Matplotlib) и удобной интеграции с алгоритмами машинного обучения. Это делает OpenCV на Python эффективным инструментом для как учебных, так и научно-практических проектов, связанных с анализом изображений.

Основной целью данной работы является рассмотрение ключевых методов обработки изображений, реализованных в библиотеке OpenCV [3].

Основные методы. Библиотека OpenCV предоставляет обширный набор функций для выполнения основных операций цифровой обработки изображений. Рассмотрим наиболее важные из них, которые чаще всего исполь-зуются при анализе визуальных данных.

1. Чтение и отображение изображений

Для начала работы с изображениями в OpenCV используются функции cv2.imread()**,** cv2.imshow() и cv2.imwrite()**.**

Функция cv2.imread() позволяет загрузить изображение в виде матрицы пикселей (NumPy-массив), что обеспечивает удобную работу с данными. При необходимости можно задать режим чтения (цветное, градации серого, без альфа-канала).

2. Изменение размеров и преобразование цветовых пространств.

Для подготовки изображений к дальнейшей обработке часто необходимо изменить их размеры или цветовую модель.

Функции cv2.resize() и cv2.cvtColor() обеспечивают эти возможности:

```
resized = cv2.resize(image, (300, 300))
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

Цветовые преобразования особенно важны, когда требуется выделить интенсивность, контраст или провести бинаризацию изображения [3; https://docs.python.org/3/].

3. Фильтрация изображений.

Одной из важнейших задач при предварительной обработке изображений является

фильтрация шумов. Шумы могут возникать в процессе съёмки, передачи или цифрового преобразования изображения. В библиотеке OpenCV реализованы различные типы фильтров, включая линейные (усредняющие) и нелинейные (медианные, гауссовы).

Пример применения фильтров:

```
# Применение усредняющего фильтра
blur = cv2.blur(image, (5, 5))
# Применение гауссова фильтра
gaussian = cv2.GaussianBlur(image, (5, 5), 0)
# Применение медианного фильтра
median = cv2.medianBlur(image, 5)
```

Гауссов фильтр обеспечивает сглаживание изображения с минимальной потерей резкости, что делает его наиболее часто используемым для предварительной обработки данных перед выделением контуров и сегментацией [1, https://docs.opencv.org/4.x/].

4. Выделение контуров.

Выделение контуров – важный этап анализа изображений, позволяющий определить границы объектов и их форму. Один из наиболее эффективных методов – алгоритм Кэнни (Canny edge detector). Он основан на градиентных преобразованиях и позволяет обнаружить значимые контуры при минимальном количестве ложных срабатываний.

Пример выделения контуров с помощью Canny:

```
# Преобразуем изображение в градации серого
gray = cv2.cvtColor(image, cv2.COLOR_BGR2
GRAY)
# Применяем размытие для снижения шумов
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
# Выделяем контуры
edges = cv2.Canny(blurred, 100, 200)
cv2.imshow('Edges', edges)
```

Функция cv2.Canny() использует два порога (нижний и верхний), что позволяет контролировать чувствительность метода к изменениям яркости. Этот метод широко применяется при распознавании объектов и сегментации изображений.

5. Морфологические операции.

Морфологические операции применяются для улучшения бинарных изображений, устранения мелких шумов и выделения структурных особенностей объектов. Основные операции – эрозия, дилатация, открытие и закрытие.

```
erosion = cv2.erode(binary, kernel, iterations=1)
dilation = cv2.dilate(binary, kernel, iterations=1)
opening = cv2.morphologyEx(binary,
cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(binary,
cv2.MORPH_CLOSE, kernel)
```

Морфологическая обработка особенно эффективна при подготовке изображений к распознаванию символов (OCR), анализу структуры тканей в медицинских снимках и обработке технических изображений [3].

Заключение. Библиотека OpenCV является мощным и гибким инструментом для обработки изображений на языке Python. Она предоставляет широкий спектр функций для:

– чтения и преобразования изображений;

– фильтрации и сглаживания;

– выделения контуров и морфологической обработки;

– подготовки данных для анализа и распознавания объектов.

Таким образом, OpenCV на Python является оптимальным выбором для исследователей и разработчиков, работающих в области компьютерного зрения, машинного обучения и анализа визуальных данных. Дальнейшие исследования могут быть направлены на интеграцию методов обработки изображений с нейронными сетями и автоматизированными системами распознавания.

СПИСОК ЛИТЕРАТУРЫ

1. *Брадски Г., Келер А.* Обучение OpenCV: компьютерное зрение с помощью библиотеки OpenCV. O'ReillyMedia, 2008 – 555 с.

2. *Келер А., Брадски Г.* Обучение OpenCV 3: Компьютерное зрение на C++ с библиотекой OpenCV. O'ReillyMedia, 2016 – 1024 с.

3. *Шелиски Р.* ComputerVision: алгоритмы и приложения. – Спрингер, 2010. 2-е издание, 2022. – 1232 с.

REFERENCES
1. *Bradski G., Kaehler A.* Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media, 2008. 555 p.
2. *Kaehler A., Bradski G.* Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. O'Reilly Media, 2016. 1024 p.
3. *Szeliski R.* Computer Vision: Algorithms and Applications. Springer, 2010, 2022. 1232 p.

BASIC IMAGE PROCESSING METHODS IN OPENCV

SHAVKAT kyzy Aida
Undergraduate Student
Osh State University
KALBAEVA Dinara Ismatillaevna
Lecturer
Osh Technological University named after academician M.M. Adyshev
Osh, Kyrgyzstan

This article examines methods and approaches to image processing using the OpenCV Python library. It provides an overview of the theoretical foundations of digital image processing and describes the library's practical tools for solving computer vision problems, including filtering, color space manipulation, edge detection, and morphological operations. This paper is intended for researchers and developers working on the automation of visual data processing and the implementation of computer vision technologies in various fields.
Keywords: image processing, computer vision, filtering, visualization, methods.