УДК 62-50

ВОЗМОЖНОСТЬ ПРИМЕНЕНИЯ АЛГОРИТМОВ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ ИЛИ СЛЕЖЕНИЯ ЗА ОБЪЕКТАМИ С БЕСПИЛОТНОГО ЛЕТАТЕЛЬНОГО АППАРАТА ПРИ ПОМОЩИ СИСТЕМЫ КОМПЬЮТЕРНОГО ЗРЕНИЯ OPENCV

Степанов В.Ю., Зуёнок А.В.
Белорусский национальный технический университет, Минск, Беларусь,
vovchik-13a@yandex.ru, itu.mido@gmail.com

Применение беспилотных летательных аппаратов (БПЛА) для слежения за объектами представляет большой интерес в различных сферах человеческой деятельности. Например, могут возникнуть задачи поиска возгораний в лесных массивах. В таком случае следует воспользоваться следующим алгоритмом распознавания [1]:

- получение данных;
- общий анализ изображения;
- поиск объектов, соответствующих определённым статическим или динамическим описаниям;
- оповещение оператора, следящего за БПЛА о результатах обработки.

Существуют специальные методы, на основе которых выявляется пламя в цветовой модели RGB (R – красный канал, G – зелёный канал, B – синий канал). Решение о принадлежности анализируемого пикселя изображению пламени принимается на основе правил, описываемых при помощи следующей системы уравнений [1]:

$$\left.\begin{aligned} & R(x,y) > R_{\text{mean}}, \\ & R_{\text{mean}} = \frac{1}{K}\sum_{i=1}^{K} R(x_i, y_i), \\ & R(x,y) > G(x,y) > B(x,y), \end{aligned}\right\}$$

где: R(x, y), G(x, y), B(x, y) – значения красного, зелёного и синего каналов в пикселе (x, y), K – общее количество пикселей, R_{mean} – среднее значение интенсивности красного цвета либо на основе трёх выделенных правил, в соответствии с которыми насыщенность каждого возможного пикселя изображения пламени должна быть больше определённого порогового значения [1].

Для обеспечения требуемой достоверности классификации можно использовать последовательные процедуры с накоплением информации.

Также часто стоит задача слежения за некоторым объектом. Упростим данную задачу и рассмотрим её на примере слежения за цветовым пятном, так как данный алгоритм, после соответствующих доработок, может без труда быть применён для поиска реальных объектов на примере возможностей библиотеки с открытым исходным кодом OpenCV. Стандартный приём в OpenCV для решения поставленной задачи складывается из двух этапов:

1) на первом этапе применяется цветовой фильтр и каждый кадр превращается в бинарное изображение. После наложения фильтра объект заданного цвета превращается в белое пятно, а всё остальное делается чёрным;

2) на втором этапе применяется алгоритм вычисления моментов. Момент изображения – это суммарная характеристика пятна, представляющая собой сумму всех точек (пикселей) этого пятна. При этом, имеется множество подвидов моментов, характеризующие разные свойства изображения.

Формула для вычисления геометрических моментов:

$$m_{pq} = \iint_{\zeta} x^p y^q f(x,y)dxdy, \qquad p,q = 0,1,2,3\ldots$$

Для бинарного изображения функция интенсивности f(x, y) будет принимать значение 1 в точках контура и значение 0 в точках фона изображения.

Программа, использующая библиотеку OpenCV для слежения за объектами и/или анализа траектории объектов, в зависимости от задачи, может быть такой, как представлено в [2]. Фрагмент листинга программы на языке Python (рисунок 1):

```
...
# HSV фильтр для цветностей искомого объекта
hsv_min = ...
hsv_max = ...

lastx = 0
lasty = 0

path_color = (0,0,255)
flag, img = cap.read()
path = createPath(img)

while True:
    flag, img = cap.read()

    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV )
    thresh = cv2.inRange(hsv, hsv_min, hsv_max)

    moments = cv2.moments(thresh, 1)

    dM01 = moments['m01']
    dM10 = moments['m10']
    dArea = moments['m00']

    if dArea > 10:
        x = int(dM10 / dArea)
        y = int(dM01 / dArea)
        cv2.circle(img, (x, y), 10, (0,0,255), -1)

    if lastx > 0 and lasty > 0:
        cv2.line(path, (lastx, lasty), (x,y), path_color, 5)

    lastx = x
    lasty = y

    # накладываем линию траектории поверх изображения
    img = cv2.add( img, path)
    cv2.imshow('result', img)
...
```

Рисунок 1 – Фрагмент листинга кода слежения за объектом на языке Python (библиотека OpenCV)

Функция moments библиотеки OpenCV возвращает массив моментов вплоть до третьего порядка и имеет следующий формат:

```
moments( кадр, двоичный )
```

Аргумент кадр представляет собой предобработанную картинку. Аргумент двоичный определяет то, как алгоритм будет вычислять вес каждой точки.

Для вычисления координат центра пятна потребуются только моменты первого порядка m01 и m10, а также момент m00.

Чтобы получить координаты X и Y искомого пятна, следует поделить полученные моменты m10 и m01 на нулевой момент m00. Таким образом находятся средние координаты X и Y всех точек, а это и есть центр пятна.

В программе центр пятна выделен при помощи красной окружности.

Важной задачей в обработке изображений (применительно к БПЛА) является оценка движения между кадрами. Для данной цели можно использовать методы оценки параметров движения кадра (при помощи возможностей одной из дополнительных библиотек OpenCV) на основе алгоритмов поиска подобия изображений: SURF [3], методов шаблонного сопоставления [4] и корреляционного анализа изображения.

ЛИТЕРАТУРА

1 А.О. Кузнецов, В.М. Мусалимов, А.П. Саенко, К.В. Трамбицкий. Применение алгоритмов анализа изображений для обнаружений пожаров. – Изв. вузов. приборостроение. – 2012. Т.55. – № 6.

2 RobotClass [Электронный ресурс]. – Режим доступа: http://robotclass.ru/tutorials/opencv-moments-color-object-search/. – Дата доступа: 05.11.2017.

3 Bay, Herbert SURF: Speeded Up Robust Features / Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool // Computer Vision and Image Understanding (CVIU). – 2008. – N 3. – P. – 346–359.

4 Brunelli, R. Template Matching Techniques in Computer Vision: Theory and Practice/ R. Brunelli // Wiley. – 2009. – P. – 239.