Санкт-Петербургский политехнический университет Петра Великого
Институт машиностроения, материалов и транспорта
Высшая школа автоматизации и робототехники

Отчёт

по лабораторной работе №4

Дисциплина: Техническое зрение

Тема: Распознавание образов на изображении при помощи контурного анализа

Студент гр. 3331506/70401 Ляпцев И.А.
Преподаватель Варлашин В.В.

« »_____2020 г.

Санкт-Петербург
2020

Задание 1

Необходимо обнаружить светлое пятно на однотонном изображении и найти его центр. Для решения этой задачи используется функция *Task1(string fileName)*, на вход подается имя файла, которое потом расширяется для поиска файла в папке.

Алгоритм работы данной функции:

1. Выполняется пороговая фильтрация библиотечной функцией *threshold(inputImage, ImageToWork, thresh, 255,THRESH_BINARY);*
2. Создается и находится контур из маски, полученной в предыдущем пункте, с помощью ф-и *Canny(ImageToWork, edges, 0, 0, 3, false)*;
3. Определяются моменты найденного контура с помощью библиотечной ф-и *moments(contours[0])*;
4. Находим центр масс контура;
5. Отрисовываем прицел пот координатам центра контура с помощью библиотечной ф-и *line(outputImage, Point(center.x- 15, center.y), Point(center.x+ 15, center.y), Scalar(0, 0, 255), 2)*.

Результат работы программы показан на рис. 1

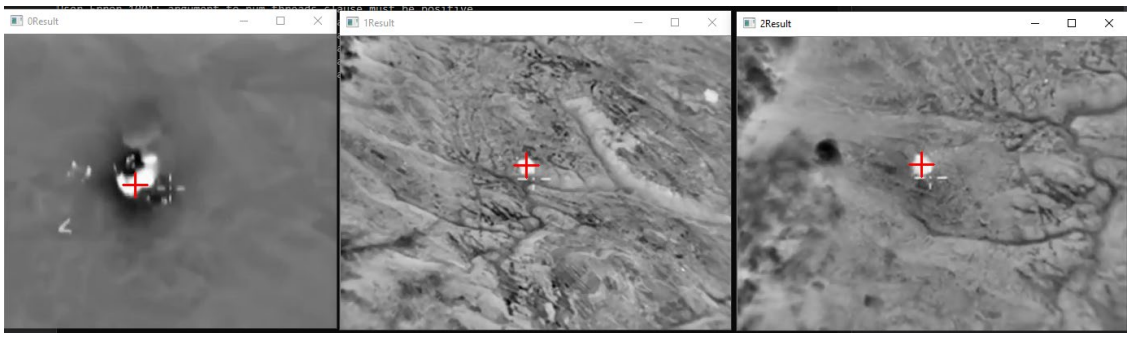

Рис. 1 — Результат работы функции *Task1*

Задание 2

Аналогично заданию 1, необходимо найти светлое пятно, но изображение подается цветное, из-за чего общий алгоритм отличается. Для решения этой задачи используется функция *Task2(string fileName)*, на вход подается имя файла, которое потом расширяется для поиска файла в папке.

Алгоритм функции:

1. Перевод изображения в цветовое пространство *HSV* ф-ей *cvtColor(inputImage, inputImage, COLOR_BGR2HSV*);
2. Пороговая фильтрация библиотечной ф-ей *inRange(inputImage, Scalar(34, 101, 0), Scalar(164, 255, 255), ImageToWork);*
3. Создается и находится контур из маски, полученной в предыдущем пункте, с помощью ф-и *Canny(ImageToWork, edges, 0, 0, 3, false)*;
4. Определяются моменты найденных контуров с помощью библиотечной ф-и для каждого найденного контура;
5. Находим период каждого контура через его момент и находим контур с наибольшим моментом;
6. Находим центр масс этого контура;
7. Отрисовываем прицел пот координатам центра контура с помощью библиотечной ф-и *line(outputImage, Point(center.x- 15, center.y), Point(center.x+ 15, center.y), Scalar(0, 0, 255), 2)*.

Фрагмент функции с поиском наибольшего периметра контура представлен ниже, здесь *contours* — вектор с контурами, *tempArea*- максимальный периметр, *mnts*- вектор с моментами каждого контура:

```
for (int i = 0; i < contours.size(); i++)
{
    if (mnts[i].m00 >= tempArea)
    {
        i_max = i;
        tempArea = mnts[i].m00;
    }
}
```

Результат работы программы показан на рис. 2

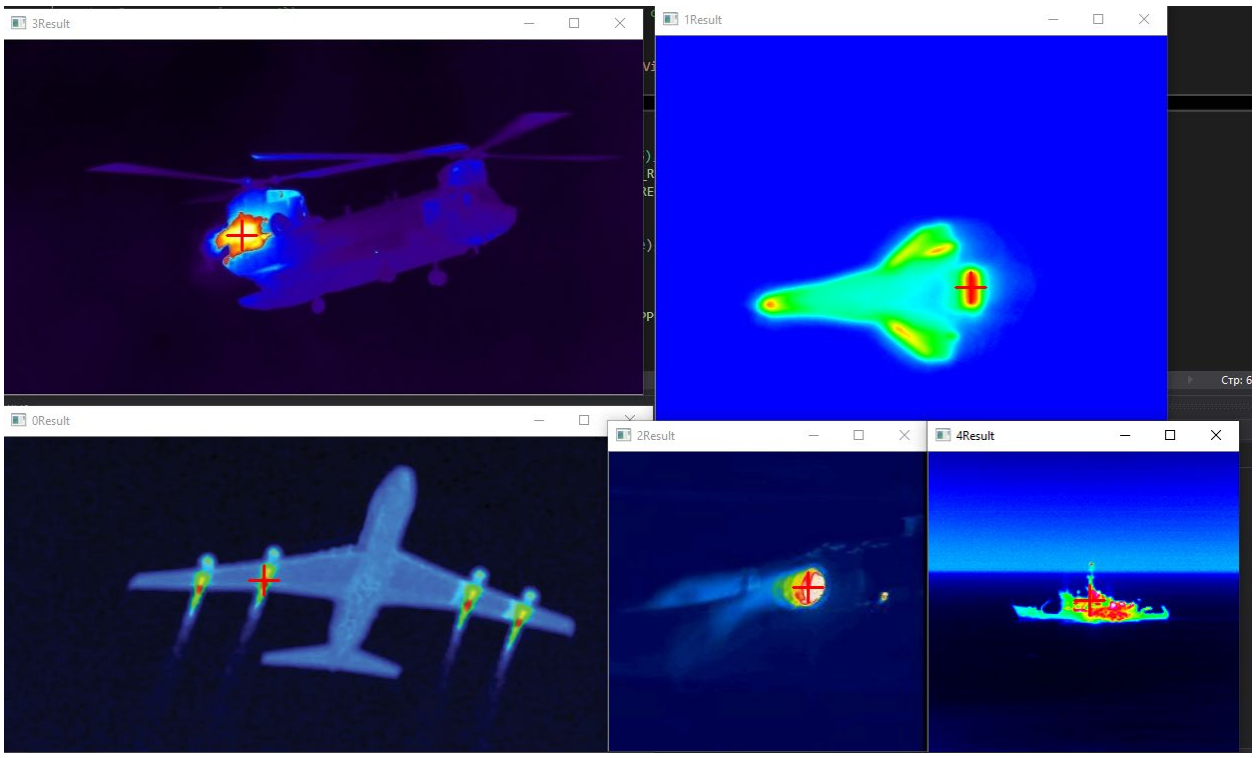

Рис. 2 — Результат работы функции *Task2*

Задание 3

В задании 3 необходимо на изображении с роботами нескольких цветов обозначить робота каждой группы, лампу и построить путь для ближайшего робота каждой группы до лампы. Для решения этой задачи используется функция *Task3(string fileName)*, на вход подается имя файла, которое потом расширяется для поиска файла в папке.

Алгоритм работы функции:

1. Перевод изображения в цветовое пространство *HSV* ф-ей *cvtColor(inputImage, inputImage, COLOR_BGR2HSV*);
2. Нахождение маски роботов функцией *maskDrawing(const int flag, const Mat inputImage)*. Эта функция определяет, маску чего нужно нарисовать, используя переменную *flag*, и возвращает маску изображения. Так как цвет лампы совпадает с цветом роботов, сначала она обнаруживается и вокруг нее рисуется маска (рис. 3);
3. Создание контуров из маски для каждого цвета;
4. Определение центра масс для робота из каждой группы;

5. Определение ближайшего робота для каждой группы с помощью ф-и *findNearest(vector<Point> centerRobot, Point centerLamp);*

6. Отрисовка контуров всех роботов и кратчайшего расстояния до лампы для трех ближайших роботов;

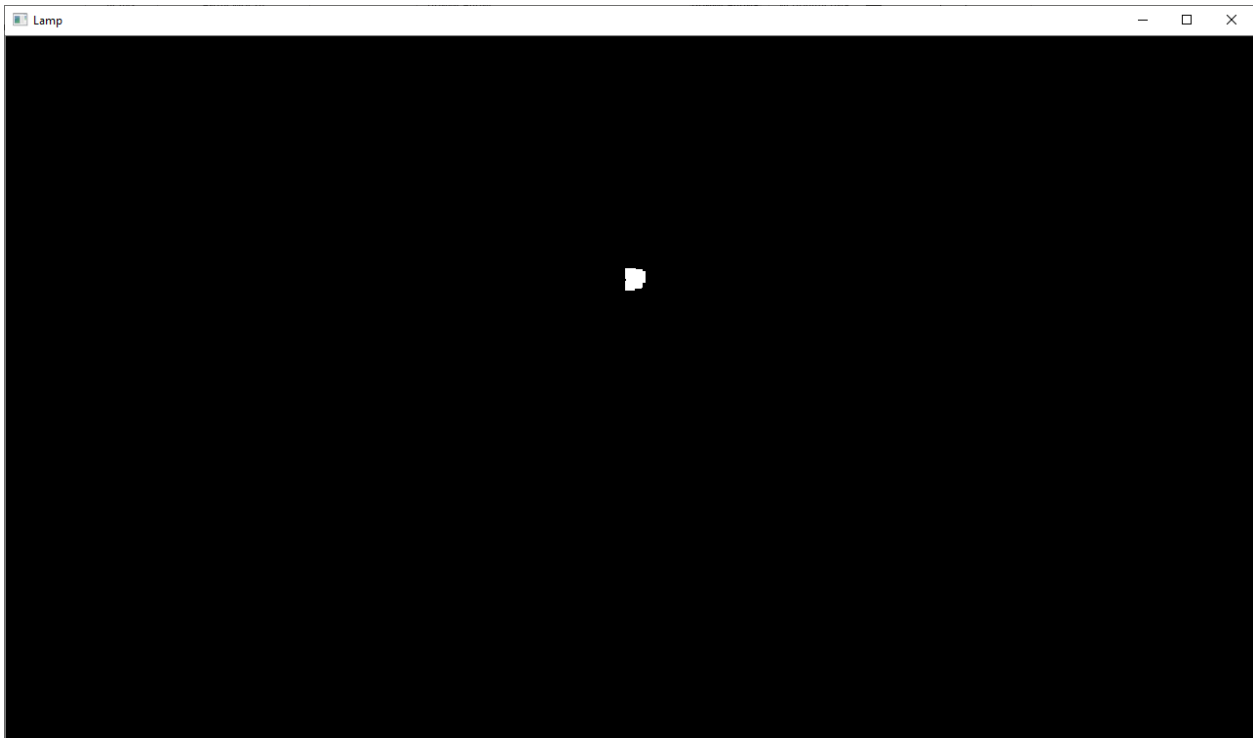

Рис. 3 — Маска лампы

Результат работы функции показан на рис. 4.

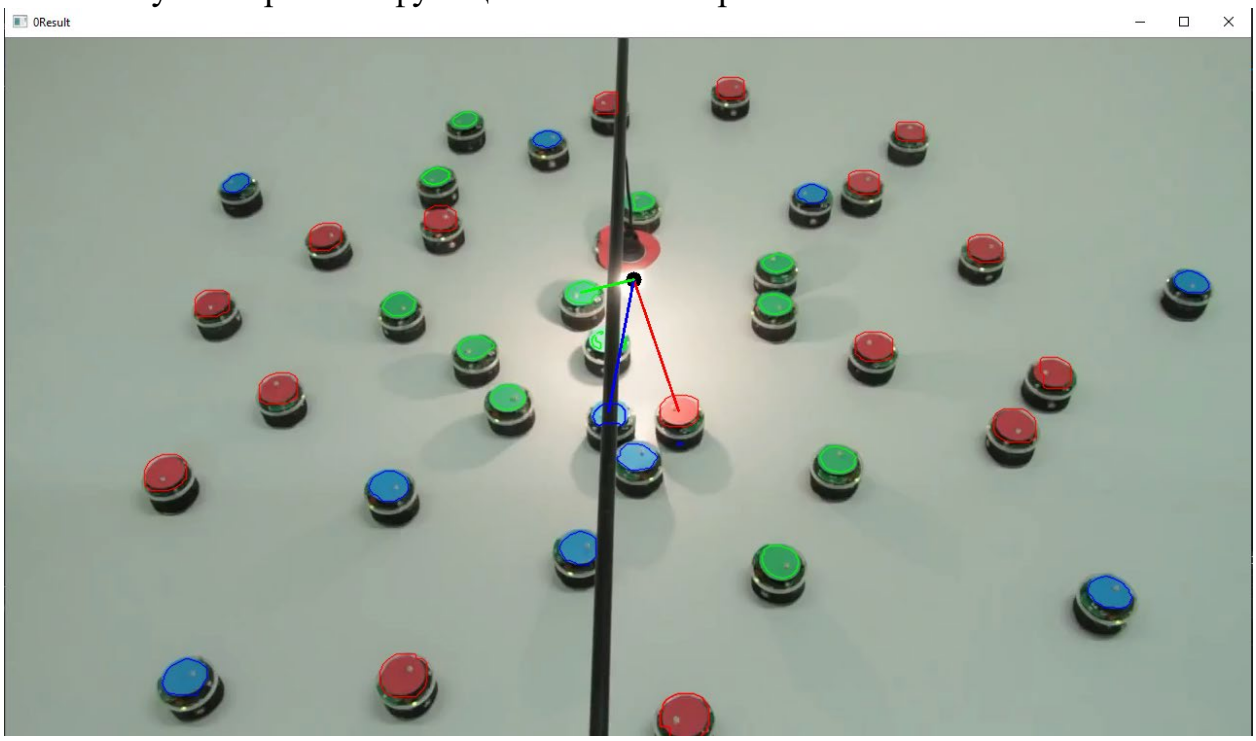

Рис. 4 — Результат работы функции *Task3*

Задание 4

В задании 4 необходимо на изображении с несколькими гаечными ключами отфильтровать их, используя шаблон. Для решения этой задачи используется функция *Task4*.

Алгоритм функции:

1. Находим маску для изображения со всеми ключами и для шаблона с помощью ф-и *threshold(inputImage, maskAll, 229, 255, THRESH_BINARY);*
2. Находим контур каждого ключа, в том числе и шаблона, с помощью функции *Canny(maskAll, edgesAll, 0, 0, 3, false);*
3. Находим различия между контуром каждого ключа и контуром шаблона с помощью ф-и *matchShapes(cntsTemplate[0], cntsAll[i], ShapeMatchModes::CONTOURS_MATCH_I2, 0)*;
4. Если разница контуров превышает единицу, контур «отбраковывается», и соответствующий ключ считается браком;
5. Каждый контур обводится в соответствии со своим качеством.

Результат работы функции показан на рис. 5.

Рис. 5 — Результат работы функции *Task4*