Отчет по выполнению тестового задания пониженной сложности

Содержание

1. Постановка задачи

1.1. Задача

Задание пониженной сложности. Распознавание знака. В архиве individual.zip находятся изображения различных дорожных знаков. Ваша задача - написать программу, выполняющую распознавание (идентификацию, классификацию) предложенных знаков. Аргументом программы является путь к каталогу с изображениями знаков в предложенном формате, в консоль программа должна выводить в каждой строке: имя файла (без полного пути), название найденного программой знака (и найден ли знак вообще) - из набора символов O/D/R/N (N для “none”) и, опционально, вероятность верного распознавания (с точки зрения программы) - например, “D_02996.ppm D 0.956”.

2. Анализ существующих решений

Так как выполняется лишь тестовое задание по распознованию ограниченного колличества типов доожных знаков. В данном разделе приводится краткий обзор существующих системм распознования дорожных знаков.

2.1. Система для распознавания дорожных знаков FOSTS (Foveal Systems for Traffic Sign Recognition)

Система предназначена для распознавания дорожных знаков инвариантно к погодным условиям, скорости и полосы движения автомобиля, а также расстоянию до знака.

В основе работы системы лежат алгоритмы определения формы знака, классификации по цвету и форме внешнего контура и формирования специфического описания с помощью оригинального подхода, основанного на позиционировании пространственно-неоднородного сенсора в центре информативной части знака.

Технические характеристики системы:

1. Точность распознавания ~ на уровне 97%.
2. Степень зашумления знака – до 50% информативной части.
3. Время распознавания – не более 2 с.
4. Изменение ракурса – до 30º.
5. Максимальное расстояние до знака – 50 м.

Оперирует со всеми типами дорожных знаков (предупредительные, запрещающие, предписывающие и т.д.).

2.2. Opel Eye TSR

Видеокамера Opel Eye включает несколько систем, одна из которых распознает дорожные знаки и линии разметки.

Характеристики:

1. 2 процессора обработки сигналов
2. Распознает только знаки, указывающие на ограничение скорости и запрещающие обгон
3. Максимальное расстояние до знака - 100 м.

3. Высказывание и гипотезы

Не смотря на то, что в поставленной задаче фигурирует всего 3 типа знаков, сложность в их обнаружении сохраняется из-за «помех» на изображении.

Из обзора аналогов следует, что системы распознавания знаков основываются на таких параметрах знака как:

1. Цвет
2. Форма
3. Контуры

Следовательно, необходимо выявить данные параметры на анализируемых изображениях.

Это можно проделать при исключении помех с помощью увеличения-

уменьшения яркости и контраста изображения, ликвидации шумов, ликвидации цветного источника света, увеличения четкости и тому подобных действий. После того как изображение скорректировано, можно приступать к поиску контуров.

Как только контуры обнаружены нужно их отфильтровать, убрать слишком малые контуры, не замкнутые, самопересекающиеся. После фильтрации контуров можно провести контурный анализ по признакам искомых фигур(Отношения сторон, площадей и окружностей) и выявить знаки.

4. Выбор пути решения задачи

Путь решения выбран самый простой – рассмотреть изображение с нескольких сторон и выносить решение из совокупности обнаруженных значений..

Рассмотрение производится со сторон:

1. Количества помех – устранить или обойти(не рассматривать) помехи
2. Цветов– анализировать RBG
3. Света – анализировать HSV
4. Контуров – получить контуры из вариаций RGB и HSV
5. Результата анализа контуров – выявить признаки знаков.

5. Детали способа реализации

Поступающее изображение рассматривается с нескольких сторон.

Изображение корректируется:

1. Увеличивается размер
2. Увеличивается чёткость
3. Увеличивается контраст
4. Нормализуется яркость
5. На изображении определяются
    1. Жёлтый цвет – главная дорога

2. Красный цвет – уступи дорогу и движение запрещено
3. Белый цвет – контуры знаков и остальной белый цвет

При выделении цветов, цвета абсолютизируются. Т.е если обнаружен жёлтый оттенок, то он становится в BGR равным (0, 255, 255). Остальные цвета обнуляются, делаясь чёрным.

Почти каждый этап фильтрации сохраняется по этому на выходе из этапа нормализации существует уже не одно, а несколько анализируемых изображений:

1. Исходное
2. Яркое
3. Контрастное
4. Чёрно-Красное
5. Чёрно-Жёлтое
6. Чёрно-Белое

Такое количество изображений позволяет лучше проанализировать интересующие нас детали. Исходное изображение может быть хорошего качества и все коррекции приведут к его ухудшению, что скажется на анализе. По этому исходное изображение сохраняется почти без коррекций. Все изображения имеют коррекцию размера, чёткости и снижения влияния цветного источника света(GrayWorld).

Так же изображение рассматривается с точки зрения пространства цветов HSV. И исходное, яркое и контрастное разделяются на HSV каналы и анализируются отдельно.

Далее производится выявление контуров. Для этого используется Canny Edge Detector. Так как анализируемых изображений к данному моменту довольно много(15), то детектор выявляет большое количество контуров, многие из которых не подходят на роль знака по очевидным причинам. К примеру контур длинной в 1 пиксель не является знаком.

После фильтрации контуров производится их анализ на принадлежность

одной из фигур. Критерии для поиска выбраны довольно простые:

1. Треугольник:
    1. Отношения длин сторон друг к другу т.к треугольник равносторонний
    2. Присутствие отсутствие высоты в предполагаемом месте
    3. Сумма длин двух сторон относительно длинны третьей
2. Круг:
    1. Отношение длинны окружности к площади
    2. Нахождение по HoughCircles
3. Ромб
    1. Близость высот ромба друг к другу
    2. Присутствие - отсутствие высоты с минимальным значением кардинаты Y. см. Документацию (rombDetection).
    3. Отношение площади найденной по точкам и при помощи функции contourArea

6. Описание достигнутых результатов

Полученные контуры разнятся от изображение к изображению. Бывают как на рисункая 1,2.

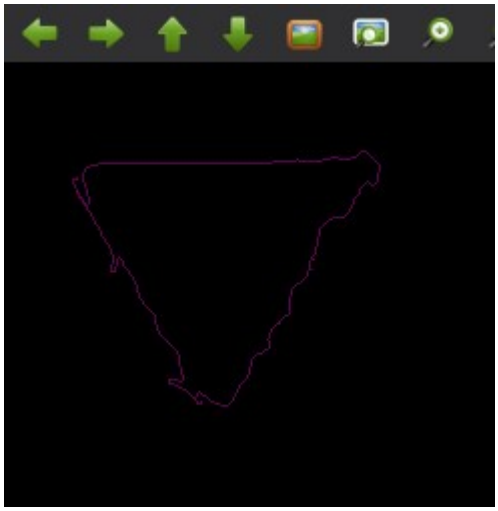

Рисунок 1: контур треугольника

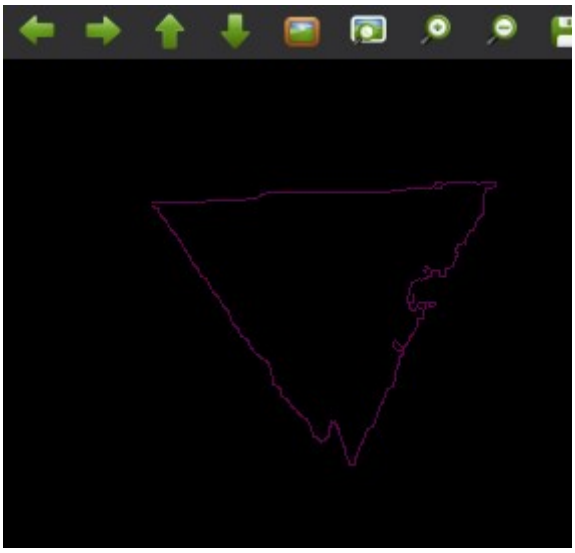

Рисунок 2: контур треугольника пример 2

Но бывают контуры менее очевидные см. Рисунок 3.

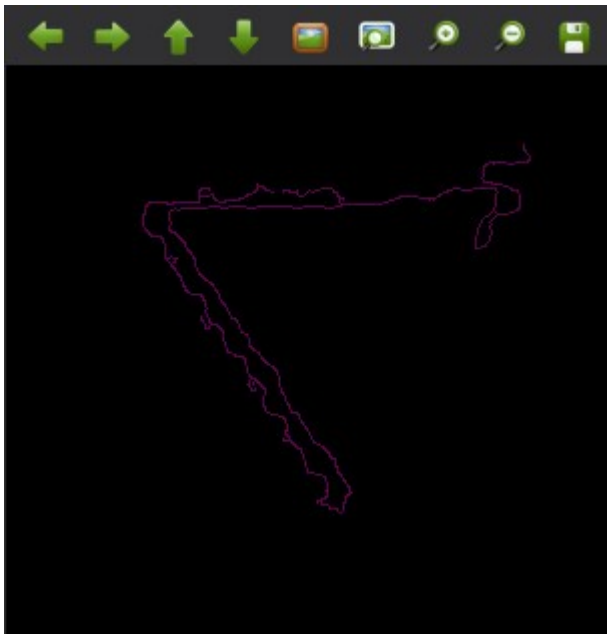

Рисунок 3: Разорванный контур

Ранее принятое решение оправдало себя. Рассмотрение изображений с разных сторон с целью получения большего количества информации позволяет выполнить поставленную задачу, но вносит, возможно, больше помех чем

полезной информации.

Возможно более тщательный анализ и более адаптивная обработка на этапе фильтрации даст более эффективный результат.

Хотя часто выявляются контуры похожие на контур на рисунке 4. Что не может не радовать.

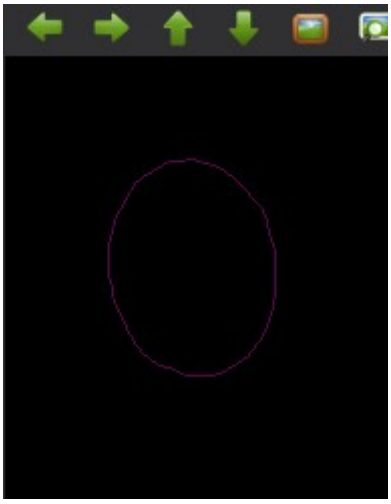

Рисунок 4: Четкий контур

Из-за большого количества анализируемых изображений программа работает довольно медленно. Но распознавание в большинстве случаев успешно.

Результат выполнения программы на изображениях из архива individual.zip:

[takava@linux-tjhe build-SignDetermination-SelfKit-Release] $./SignDetermination /home/takava/Documents/Study/Works/individual/

```
0) Processing for: D_00280.ppm:  D 1
1) Processing for: D_00555.ppm:  D 1
2) Processing for: D_00569.ppm:  D 0.888889
3) Processing for: D_00739.ppm:  D 1
4) Processing for: D_00863.ppm:  N
5) Processing for: D_00902.ppm:  D 1
6) Processing for: D_01211.ppm:  D 0.666667
7) Processing for: D_01280.ppm:  D 1
8) Processing for: D_02213.ppm:  O 0.5
9) Processing for: D_02214.ppm:  D 1
10) Processing for: D_02280.ppm:  D 1
11) Processing for: D_02371.ppm:  D 0.8
12) Processing for: D_02604.ppm:  D 1
13) Processing for: D_02748.ppm:  D 0.909091
14) Processing for: D_02996.ppm:  D 1
15) Processing for: D_02999.ppm:  D 0.666667
16) Processing for: D_03086.ppm:  N
```

```
17) Processing for: D_04235.ppm:  O 1
18) Processing for: D_07515.ppm:  N
19) Processing for: D_08381.ppm:  N
20) Processing for: D_08477.ppm:  O 1
21) Processing for: D_08537.ppm:  N
22) Processing for: D_08619.ppm:  R 0.875
23) Processing for: D_09279.ppm:  D 0.857143
24) Processing for: D_09285.ppm:  N
25) Processing for: D_09426.ppm:  D 0.823529
26) Processing for: D_09594.ppm:  D 1
27) Processing for: D_10108.ppm:  N
28) Processing for: O_00063.ppm:  O 1
29) Processing for: O_00185.ppm:  O 1
30) Processing for: O_00522.ppm:  O 1
31) Processing for: O_01244.ppm:  O 1
32) Processing for: O_01591.ppm:  O 1
33) Processing for: O_02365.ppm:  O 1
34) Processing for: O_02617.ppm:  O 1
35) Processing for: O_02780.ppm:  O 1
36) Processing for: O_02797.ppm:  O 1
37) Processing for: O_03120.ppm:  O 1
38) Processing for: O_03682.ppm:  O 1
39) Processing for: O_04254.ppm:  O 1
40) Processing for: O_04556.ppm:  O 1
41) Processing for: O_05305.ppm:  O 0.666667
42) Processing for: O_05995.ppm:  O 1
43) Processing for: O_06256.ppm:  O 1
44) Processing for: O_06741.ppm:  O 1
45) Processing for: O_07142.ppm:  O 1
46) Processing for: O_07242.ppm:  O 1
47) Processing for: O_08441.ppm:  O 1
48) Processing for: O_08707.ppm:  O 1
49) Processing for: O_08751.ppm:  O 1
50) Processing for: O_09220.ppm:  O 1
51) Processing for: O_09489.ppm:  O 1
52) Processing for: O_09593.ppm:  O 1
53) Processing for: O_09685.ppm:  O 1
54) Processing for: O_09754.ppm:  O 0.6
55) Processing for: O_10199.ppm:  O 1
56) Processing for: O_12187.ppm:  O 1
57) Processing for: O_12456.ppm:  N
58) Processing for: R_00012.ppm:  R 1
59) Processing for: R_00032.ppm:  R 0.666667
60) Processing for: R_00077.ppm:  R 1
61) Processing for: R_00098.ppm:  R 1
62) Processing for: R_00434.ppm:  R 1
63) Processing for: R_00450.ppm:  N
64) Processing for: R_00552.ppm:  N
65) Processing for: R_00997.ppm:  R 1
66) Processing for: R_01486.ppm:  R 1
67) Processing for: R_01843.ppm:  R 1
68) Processing for: R_01920.ppm:  R 1
69) Processing for: R_02105.ppm:  R 1
70) Processing for: R_02136.ppm:  R 0.909091
71) Processing for: R_02151.ppm:  R 1
72) Processing for: R_02169.ppm:  R 1
73) Processing for: R_02281.ppm:  N
74) Processing for: R_02453.ppm:  R 0.538462
75) Processing for: R_02457.ppm:  R 1
76) Processing for: R_02886.ppm:  R 1
77) Processing for: R_03596.ppm:  O 0.8
78) Processing for: R_03674.ppm:  N
79) Processing for: R_03697.ppm:  R 0.8
80) Processing for: R_03729.ppm:  N
```

```
81) Processing for: R_03913.ppm:  N
82) Processing for: R_04195.ppm:  R 1
83) Processing for: R_04253.ppm:  R 1
84) Processing for: R_07209.ppm:  N
85) Processing for: R_07408.ppm:  R 1
86) Processing for: R_07417.ppm:  N
87) Processing for: R_09992.ppm:  R 0.933333
88) Processing for: R_11310.ppm:  R 1
89) Processing for: D_00116.ppm:  D 0.941176
90) Processing for: O_06777.ppm:  O 1
91) Processing for: R_10409.ppm:  N
```