Министерство образования и науки РФ
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования «НИТУ «МИСиС»
Институт ИТАСУ
Кафедра ИК

Отчёт по курсовой работе
по дисциплине
«Технологии программирования»

Приложение “Аугментатор изображений”

Выполнила:
Студентка группы БПМ-17-1
Куликова Д. М.
Проверил:
Полевой Д.В.

Москва 2018

Оглавление

Техническое задание **2**

Дополнительные технические требования **2**

Реализация **3**

Пользовательская документация 3

Основные функции 8

Инструкция по сборке **11**

Техническое задание

Необходимо написать программу для аугментации данных для тестирования нейронных сетей. Программа должна вырезать изображение автомобиля по маске и «вклеивать» его на изображение с дорогой (изображения с дорогами также имеют маски, на которых выделены участки дороги, подходящие для «вклеивания» автомобиля). В результате работы программы должно сгенерироваться указанное пользователем количество изображений. Программа должна допускать хранение конфигураций в виде внешних файлов и запуск из командной строки.

Пример работы программы:

Рисунок 1.

Дополнительные технические требования

Программа должна обрабатывать неправильное поведение пользователя. То есть должны выбрасываться исключения в следующих ситуациях:

- json файл конфигурации не найден либо пуст (с завершением работы программы);
- неправильный путь к изображению;
- неправильный формат файла изображения;
- размер маски изображения не соответствует размеру самого изображения.

Реализация

Пользовательская документация

Алгоритм обработки пары изображений “автомобиль – дорога” (+ маски):

1. На вход подаются изображения автомобиля (рис. 2) и дороги (рис. 4), а также их маски (рис. 3 и 5).

Рисунок 2.

Рисунок 3.

Рисунок 4.

Рисунок 5.

2. Для изображения с дорогой выбирается произвольная высота (в пределах маски дороги), на которой будет «вклеен» автомобиль (рис. 6 и 7).

Рисунок 6.

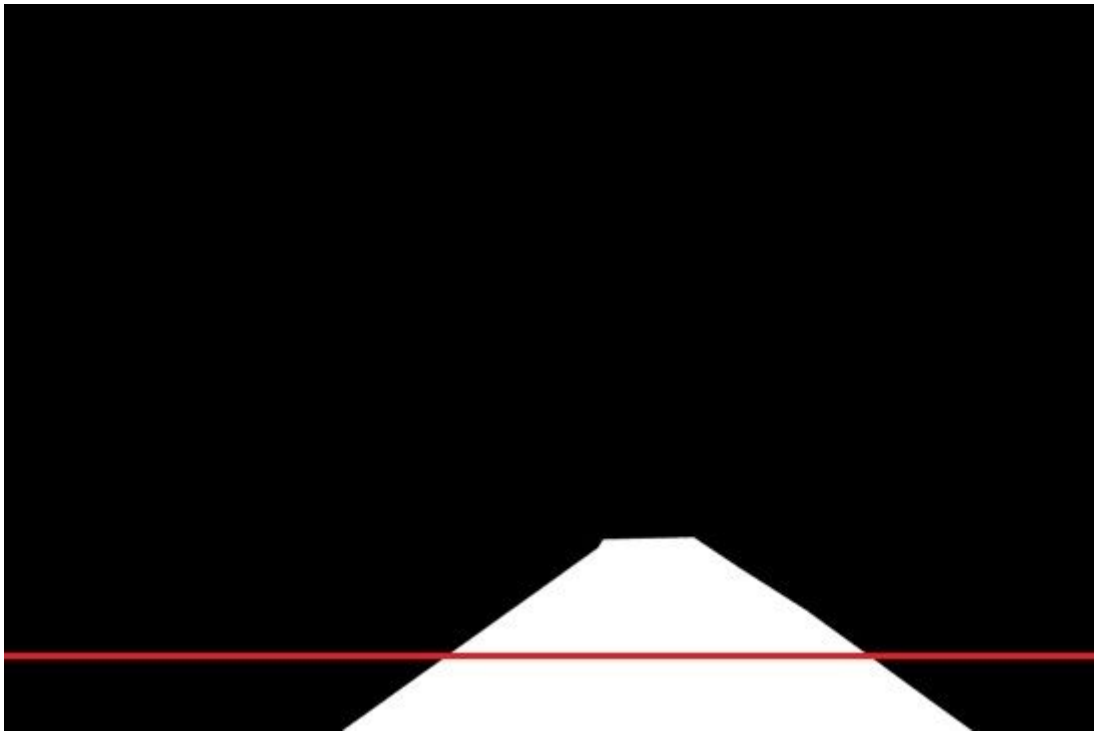

Рисунок 7.

3. Измеряется ширина дороги на данной высоте и ширина автомобиля для расчета коэффициента уменьшения размера автомобиля.
4. Изменяется размер автомобиля и его маски в соответствии с полученным коэффициентом (рис. 8).

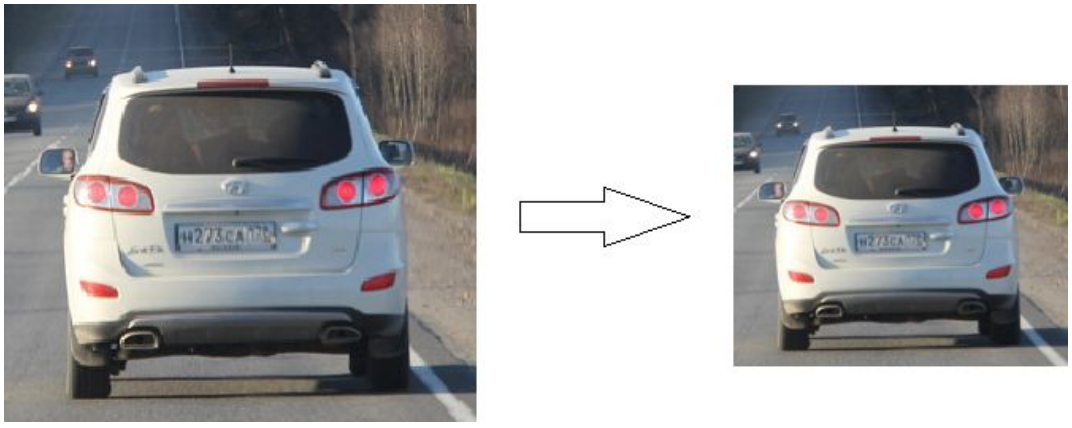

Рисунок 8.

5. Уменьшенное изображение автомобиля «вклеивается» на дорогу так, чтобы нижняя граница изображения автомобиля находилась на полученной ранее высоте (рис. 9).

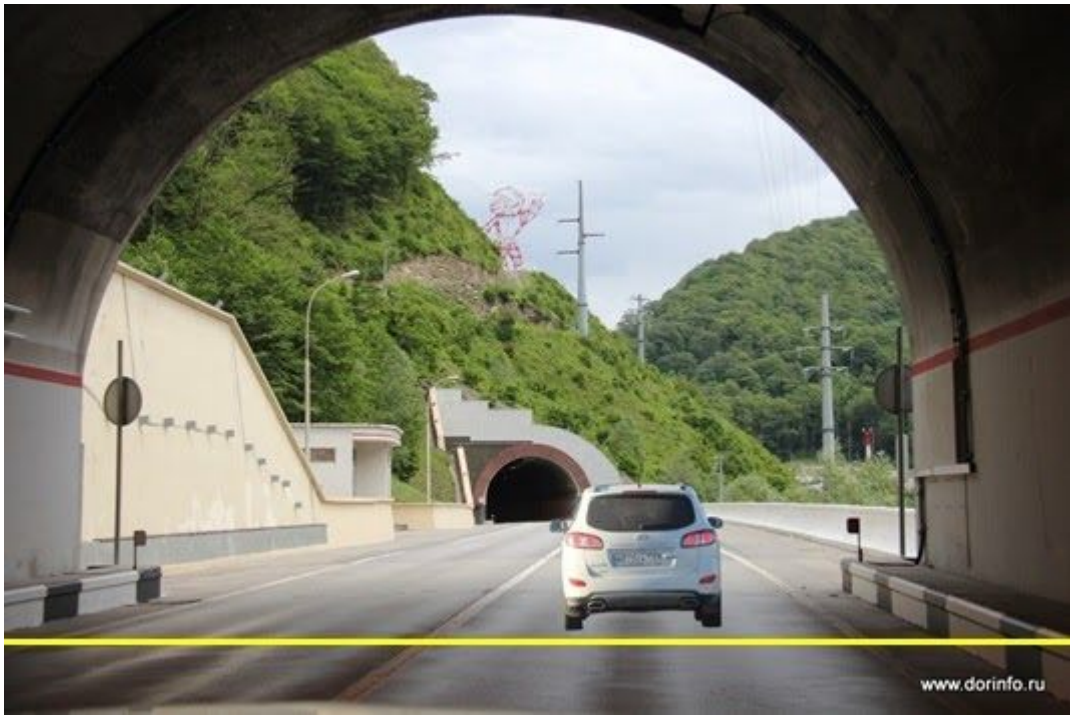

Рисунок 9.

Файлы конфигурации:

Для запуска программы пользователю необходимо создать json файл, имеющий следующую структуру:

```
{
   "cars": [
        {
            "image": "путь к изображению с автомобилем",
            "mask": "путь к маске для этого автомобиля"
        },
        {
            // Такой же блок для следующей пары автомобиль - маска
            "image": "...",
             "mask": "..."
        }
    ],
     "roads": [
        {
            "image": "путь к изображению с дорогой",
            "mask": [
                // Маски для этой дороги (количество масок может быть любым)
                "...",
```

```
                "...",
                "..."
            ]
        },
        {
            "image": "...",
            "mask": [
              "..."
            ]
        }
    ]
}
```

При запуске из командной строки можно указать следующие параметры:

Ключ	Значение по умолчанию	Описание
-j	dataset.json	Путь к json файлу, содержащему пути к входным изображениям
-r	result	Путь к папке, в которую сохраняются сгенерированные изображения
-n	Количество уникальных пар “автомобиль - маска дороги”	Желаемое количество сгенерированных изображений

Основные функции

Функции

- void ReadFromJson(const cv::FileStorage& dataset, cv::FileNode& roads, cv::FileNode& cars)

Считывает данные и файла типа json. В случае если файл пуст или не найден, выбрасывает исключение.

dataset	Данные из dataset.json
roads	Данные из dataset.json из блока “roads”
cars	Данные из dataset.json из блока “cars”

- cv::Mat ReadImage(const std::string name)

Считывает изображение по указанному пути. Если файл не найден или указан некорректный тип файла, выбрасывает исключение.

name	Путь к изображению

- void CheckMask(const cv::Mat& image, const cv::Mat mask, const std::string path_to_image, const std::string path_to_mask)

Проверяет соответствует ли размер маски размеру самого изображения, если нет, выбрасывает исключение.

image	Изображение
mask	Маска изображения
path_to_image	Путь к изображению
path_to_mask	Путь к маске

- int CountUniquePairs(cv::FileNode cars, cv::FileNode roads)

Считает количество уникальных пар "автомобиль - маска дороги"

cars	Данные из dataset.json из блока “cars”
roads	Данные из dataset.json из блока “roads”

Возвращает

Количество уникальных пар

- bool IsWhite(const cv::Mat mask, const int i, const int j)

  Проверяет, является ли пиксель белым.

mask	Изображение (маска)
i, j	Координаты пикселя

Возвращает

true, если пиксель белый, иначе - false

- std::pair<int, int> HeightOfMask(const cv::Mat mask)

  Определяет начало и конец дороги (в соответствии с маской).

mask	Маска

Возвращает

Пару значений - начало и конец дороги (номера первой и последней строк дороги).

- int RandomNumber(const std::pair<int, int> range)

  Вычисляет рандомное число из заданного диапазона.

range	Диапазон

Возвращает

Рандомное число из заданного диапазона.

- std::pair<int, int> WidthOfMask(const cv::Mat mask)

  Определяет левую и правую границы маски.

mask	Маска

Возвращает

Левую и правую границы маски (номера столбцов).

- std::pair<int, int> WidthOfMask(const cv::Mat mask, const int row)

  Определяет левую и правую границы маски на заданной строке.

mask	Маска
row	Номер строки

Возвращает

Левую и правую границы маски на заданной строке (номера столбцов).

- void AlphaBlending(cv::Mat car, cv::Mat car_mask, cv::Mat road, cv::Mat road_mask, const int i_out_image)

"Вклеивает" изображения автомобиля на дорогу в соответствии с алгоритмом, описанным выше.

car	Изображение автомобиля
car_mask	Маска автомобиля
road	Изображение дороги
road_mask	Маска дороги
i_out_image	Количество уже сгенерированных изображений

Инструкция по сборке

Установить, если не были установлены ранее CMake и OpenCV. Скачать файлы из репозитория: https://github.com/KulikovaDaria/data-augmentation. Собрать проект с помощью CMake. Создать и заполнить json файл в соответствии со структурой, указанной в разделе Реализация. Создать папку result, в ней будут сохранены полученные изображения.

Также в репозитории можно найти тестовый dataset (папка dataset).