Министерство образования и науки Российской Федерации

Национальный исследовательский технологический университет «МИСиС»

Институт информационных технологий и автоматизированных систем управления

Кафедра Инженерной Кибернетики

УТВЕРЖДАЮ

Доцент \_\_\_\_\_\_\_\_\_\_\_\_ Полевой Д.В.

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**Курсовая работа   
по дисциплине «Технологии программирования»**

**«Пакетный обработчик изображений»**

Листов 9

Руководитель,

доцент кафедры ИК \_\_\_\_\_\_\_\_\_\_\_\_ Доцент Д.В.

Исполнитель,

студент гр. БПМ-17-2\_\_\_\_\_\_\_\_\_\_\_\_ Скибин К.С.

Москва 2018

**Оглавление**

1. Задачи ПО 3

2. Технические требования 3

3. Пользовательская документация 3

3.1. Файл конфигурации 4

3.2. Поддерживаемые операции 5

3.3. Запуск утилиты 8

3.4. Ошибки и сбои 8

3.5. Тестирование 8

4. Структура и функционал 9

4.1. Структура 9

4.2. Функционал 9

4.3. Используемые библиотеки 10

5. Инструкция по сборе и развертыванию 10

1. **Задачи ПО**

Разработанная утилита предназначена для обработки и модификации массивов изображений. Конечной целью является получение аугментированных изображений, пригодных для обучения нейронных сетей, работающих с картинками и фотографиями. Данный продукт позволяет осуществлять различные преобразования над изображениями, порядок и параметры которых регулирует пользователь, путем создания файлов конфигурации.

1. **Технические требования**
2. Кроссплатформенность
3. Язык программирования C++, стандарт C++11
4. Устойчивость к сбоям

**3. Пользовательская документация**

3.1. Файл конфигурации

Первым шагом при работе с данной утилитой является создание файла конфигурации. Для генерации шаблона файла конфигурации необходимо запустить исполняемый файл программы без передачи ему аргументов. Сгенерированный шаблон появится в папке с исполняемым файлом и будет иметь название sample.json. Редактирование файла конфигурации производится при помощи любого текстового редактора.

Файл конфигурации представляет собой массив операций, последовательно выполняемых над изображениями. Каждая операция является JSON объектом и имеет данную структуру :

*action* **(string)**  — наименование операции.

*src* **(string vector)** — вектор строк, указывающих, какие изображения должны быть подвергнуты метаморфозе.

*dst* **(string)** — строка, указывающая, куда будет отправлены аугментированные изображения.

*settings* **(string-string map)** — контейнер, связывающий названия параметров и их значения.

Каждая операция накладывает на src и dst ограничения

Ограничения SRC :

1. Только пути к каталогам и файлам

“src”: [

“input\_dir/\*”,

“input\_image.png”

]

1. Только placeholder’ы [[1]](#footnote-1)

“src”: [

“$input\_placeholder\_1”,

“$input\_placeholder\_2”

]

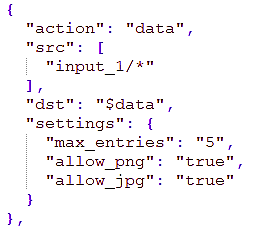
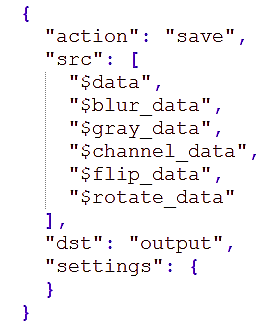
Ограничения DST :

1. Путь к каталогу

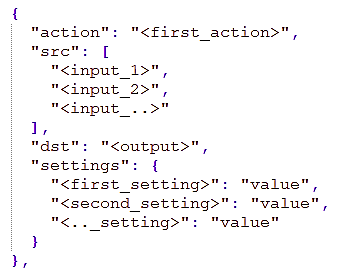
“dst” : “images/output\_dir”

1. Placeholder

“dst” : “$output\_placeholder”

****

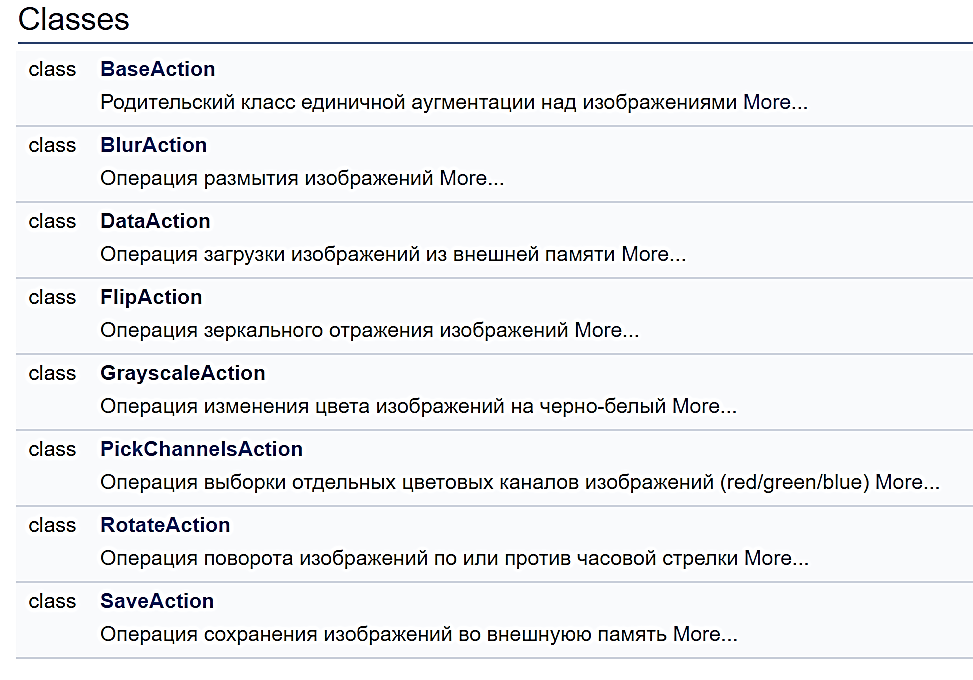
*Примеры операций*



*Шаблонная операция*

3.2. Поддерживаемые операции

Данная утилита предлагает пользователю перечень операций, которые могут быть осуществлены над массивами изображений



Остановимся подробнее на каждой из них

1.

Класс : DataAction

Наименование в файле конфигурации : “data”

SRC : только пути к каталогам и файлам

DST : placeholder

Параметры :

* max\_entries **int** (максимальное количество изображений, которые будут помещены в placeholder) ;
* allow\_png **bool** (разрешить загрузку изображений формата .png) ;
* allow\_jpg **bool** (разрешить загрузку изображений формата .jpg)

Описание : загружает массив изображений в оперативную память

2.

Класс : SaveAction

Наименование в файле конфигурации : “save”

SRC : только placeholder’ы

DST : каталог

Параметры : —

Описание: сохраняет массив изображений во внешнюю память компьютера

3.

Класс : GrayscaleAction

Наименование в файле конфигурации : “gray”

SRC : только placeholder’ы

DST : placeholder

Параметр :

* alpha\_channel **bool** (надо ли учитывать альфа-канал при обработке)

Описание: возвращает массив изображений в оттенках серого

4.

Класс : PickChannelsAction

Наименование в файле конфигурации : “pick\_channels”

SRC : только placeholder’ы

DST : placeholder

Параметры :

* red **bool** (сохранить канал красного цвета) ;
* green **bool** (сохранить канал зеленого цвета) ;
* blue **bool** (сохранить канал синего цвета) ;
* alpha **bool** (сохранить альфа-канал)

Описание: возвращает массив изображений с учетом включенных цветовых каналов

5.

Класс : RotateAction

Наименование в файле конфигурации : “rotate”

SRC : только placeholder’ы

DST : placeholder

Параметры :

* clockwise **bool** (по или против часовой стрелки) ;
* angle **int** (угол поворота изображения в градусах)

Описание: возвращает массив изображений, повернутых на указанное количество градусов по или против часовой стрелки

6.

Класс : FlipAction

Наименование в файле конфигурации : “flip”

SRC : только placeholder’ы

DST : placeholder

Параметр :

* type **string** (“vertically”, “horizontally”)

Описание: возвращает массив изображений, отраженных по вертикальной или горизонтальной оси

7.

Класс : BlurAction

Наименование в файле конфигурации : “blur”

SRC : только placeholder’ы

DST : placeholder

Параметры :

* filter **string** (“homogeneous”, “gaussian”, “median”, “bilateral”) ;
* kernel\_length **int** (размер ядра размытия 1-31)

Описание: возвращает массив изображений, размытых с использованием выбранного фильтра

3.3. Запуск утилиты

Для запуска утилиты необходимо запустить исполняемый .exe файл программы, передав в него путь к файлу конфигурации в качестве аргумента.

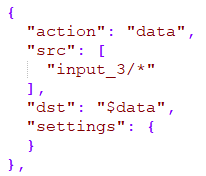
3.4. Ошибки и сбои

Информация обо всех ошибках и сбоях, возникших во время выполнения программы заносится в отдельный файл log\_output.txt

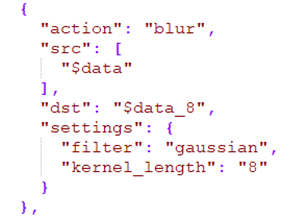
3.5. Тестирование

Для тестирования программы предусмотрены 4 файла конфигурации (test\_1.json, test\_2.json, …) и идущие вместе с ними каталоги с наборами изображений (input\_1, input\_2, …)

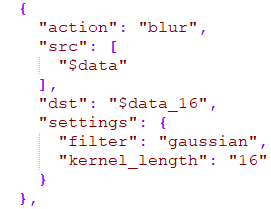
Разберём работу программы на примере третьего файла конфигурации :



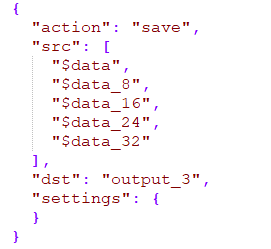
Операция 1



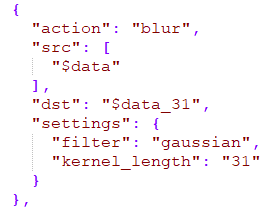
Операция 2



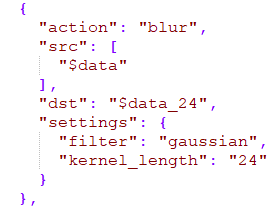
Операция 3



Операция 6



Операция 5



Операция 4

Операция 1 : загрузка изображений из папки input\_3

Операции 2-5 : размытие загруженных изображений фильтром Гаусса с размером ядром размытия 8-16-24-31

Операция 6 : сохранение исходного набора изображений и четырех наборов размытых изображений в папку output\_3

Примеры изображений на выходе

Размер ядра 16



Размер ядра 31

Исходное изображение

**4. Структура и функционал**

4.1. Структура

На уровне исходного кода в программе можно выделить 2 части :

1) Функциональную часть, вынесенную в отдельную библиотеку, подключаемую к проекту при сборке

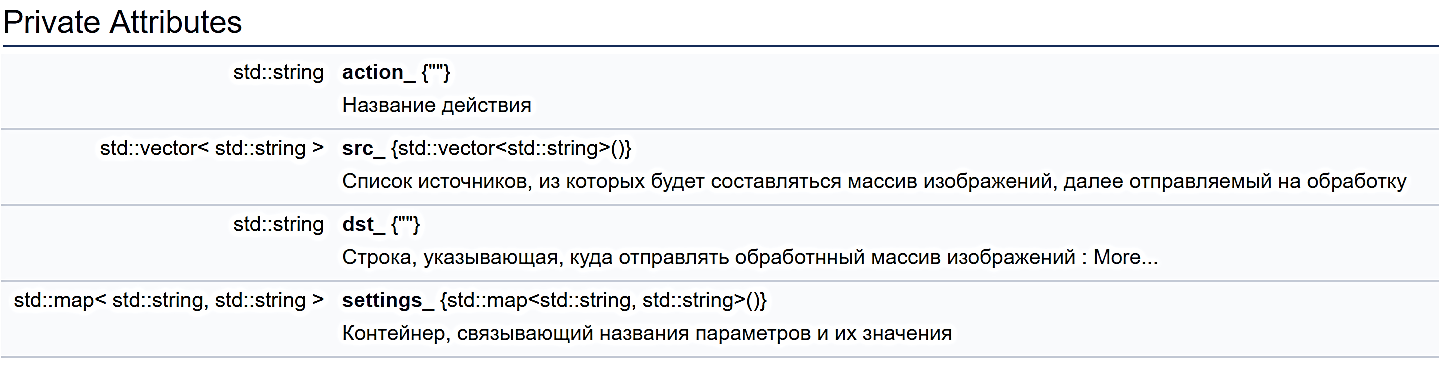
2) Тестовую часть, содержащую main.cpp, с которого непосредственно начинается выполнение программы

Связующим звеном между этими двумя компонентами является заголовочный файл entrance.h, который содержит основные функции библиотеки, доступные для пользователя.

4.2. Функционал

В качестве входных данных система принимает файл конфигурации формата json. При помощи массива объектов-действий, пользователь должен задать последовательность операций при обработке изображений. При этом пользователь может использовать только те инструкции и параметры, которые изложены в документации программы.

Каждый объект-действие на первом этапе обработки считывается в формальный класс Layer, имеющий данные атрибуты :



Перечень доступных действий, их функционал и параметры можно найти в п.3.2.

В случае безошибочного считывания объекта-действия, созданный экземпляр Layer, перенаправляется в ActionsFactory, где в соответствии с action\_ будет физически выполнено то или иное действие.

Для разграничения массивов изображений, находящихся во внешней и в оперативной памяти, был реализован механизм placeholder. Placeholder — это класс, хранящий массив именованных изображений . Он необходим для доступа к массивам изображений, временно хранящихся в куче. В файле конфигурации Placeholder обозначается в виде строки, которой предшествует знак ‘$’

4.3. Используемые библиотеки

Для разработки функциональной части программы были использованы сторонние библиотеки, доступные для бесплатного использования :

1. dirent.h — получение массива с названиями файлов, находящихся в каталоге
2. json.h — парсинг файлов конфигурации
3. OpenCV 3.4.3 — загрузка, обработка и сохранение растровых изображений

**5. Инструкция по сборке и развертыванию**

Ссылка на репозиторий : <https://mysvn.ru/lisiyrak/skibin_k_s/coursework/>

Стороннее ПО, требуемое для сборки:

1. CMake 3.13.1
2. OpenCV 3.4.3

1. Механизм Placeholder описан в п.4.2. [↑](#footnote-ref-1)