МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**Факультет программной инженерии и компьютерной техники**

**Задание №1**

по дисциплине

“Методы обработки изображений”

Студент:

Гуменник Петр Олегович

Группа P3333

Преподаватель:

Андреев Артем Станиславович

Санкт-Петербург, 2025

### **Цель:**

Ознакомиться с библиотекой OpenCV и её возможностями по визуализации.

Настроить рабочее окружение на базе Visual Studio 2022 Community Edition и OpenCV 4.5.5.

Создать и модифицировать приложение, демонстрирующее генерацию случайных графических объектов с использованием альтернативного генератора случайных чисел и пользовательских текстов.

**Процесс выполнения работы:**

#### ****1. Настройка среды разработки****

* Установлена **Visual Studio 2022 Community Edition** с компонентами:
  + Desktop development with C++
  + Windows application development
* Загружена и установлена OpenCV 4.5.5 с сайта: <https://sourceforge.net/projects/opencvlibrary/files/>
* Внесены изменения в переменные среды Windows:
  + Добавлен путь к bin директории OpenCV:  
    C:\opencv\build\x64\vc15\bin в PATH
* В **Visual Studio**:
  + Создан новый пустой проект.
  + В настройках проекта вручную добавлены:
    - Include Directories: C:\opencv\build\include
    - Library Directories: C:\opencv\build\x64\vc15\lib
    - Additional Dependencies: opencv\_world455.lib (Release), opencv\_world455d.lib (Debug)
    - Отключена инкрементная сборка
    - Установлен путь к DLL через Properties -> Debugging -> Environment

#### ****2. Разработка и модификация программы****

На основе урока [Basic Geometric Drawing](https://docs.opencv.org/4.x/d3/d96/tutorial_basic_geometric_drawing.html) была создана и расширена C++ программа.

##### Реализовано:

* Подключение альтернативного генератора случайных чисел — std::mt19937.
* Создание собственной палитры цветов (вместо случайного побитового цвета).
* Добавление собственных текстовых сообщений: "Hi,my friend!", "OpenCV rules!" и др.
* Реализация следующих функций:
  + Drawing\_Random\_Lines: рисование случайных линий
  + Drawing\_Random\_Circles: генерация окружностей
  + Displaying\_Random\_Text: вывод текста с разными шрифтами, размерами и цветами
  + Displaying\_Big\_End: финальный вывод текста с эффектом затемнения

#include <opencv2/opencv.hpp>

#include <iostream>

#include <random>

using namespace cv;

using namespace std;

const int NUMBER = 50;

const int DELAY = 20;

const int window\_width = 800;

const int window\_height = 600;

std::mt19937 gen(42);

std::uniform\_int\_distribution<> rand\_x(0, window\_width);

std::uniform\_int\_distribution<> rand\_y(0, window\_height);

std::uniform\_int\_distribution<> rand\_size(10, 100);

std::uniform\_int\_distribution<> rand\_thickness(1, 10);

std::uniform\_int\_distribution<> rand\_font(0, 7);

std::uniform\_real\_distribution<> rand\_scale(0.1, 5.0);

vector<Scalar> palette = {

Scalar(255, 0, 0), Scalar(0, 255, 0), Scalar(0, 0, 255),

Scalar(255, 255, 0), Scalar(255, 0, 255), Scalar(0, 255, 255)

};

Scalar randomColor() {

return palette[gen() % palette.size()];

}

int Drawing\_Random\_Lines(Mat& image, const string& window\_name) {

for (int i = 0; i < NUMBER; i++) {

Point pt1(rand\_x(gen), rand\_y(gen));

Point pt2(rand\_x(gen), rand\_y(gen));

line(image, pt1, pt2, randomColor(), rand\_thickness(gen), LINE\_8);

imshow(window\_name, image);

if (waitKey(DELAY) >= 0) return -1;

}

return 0;

}

int Drawing\_Random\_Circles(Mat& image, const string& window\_name) {

for (int i = 0; i < NUMBER; i++) {

Point center(rand\_x(gen), rand\_y(gen));

int radius = rand\_size(gen);

circle(image, center, radius, randomColor(), rand\_thickness(gen));

imshow(window\_name, image);

if (waitKey(DELAY) >= 0) return -1;

}

return 0;

}

int Displaying\_Random\_Text(Mat& image, const string& window\_name) {

vector<string> messages = { "Hi, my friend!", "OpenCV rules!", "Figure generator", "Random text" };

for (int i = 0; i < NUMBER; i++) {

Point org(rand\_x(gen), rand\_y(gen));

string txt = messages[gen() % messages.size()];

putText(image, txt, org, rand\_font(gen), rand\_scale(gen), randomColor(), rand\_thickness(gen), LINE\_8);

imshow(window\_name, image);

if (waitKey(DELAY) >= 0) return -1;

}

return 0;

}

int Displaying\_Big\_End(Mat& image, const string& window\_name) {

string msg = "OpenCV forever!";

Size textsize = getTextSize(msg, FONT\_HERSHEY\_COMPLEX, 3, 5, 0);

Point org((window\_width - textsize.width) / 2, (window\_height - textsize.height) / 2);

Mat image2;

for (int i = 0; i < 255; i += 2) {

image2 = image - Scalar::all(i);

putText(image2, msg, org, FONT\_HERSHEY\_COMPLEX, 3, Scalar(i, i, 255), 5, LINE\_8);

imshow(window\_name, image2);

if (waitKey(DELAY) >= 0) return -1;

}

return 0;

}

int main() {

const string window\_name = "Random OpenCV Drawing";

namedWindow(window\_name, WINDOW\_AUTOSIZE);

Mat image = Mat::zeros(window\_height, window\_width, CV\_8UC3);

if (Drawing\_Random\_Lines(image, window\_name) != 0) return 0;

if (Drawing\_Random\_Circles(image, window\_name) != 0) return 0;

if (Displaying\_Random\_Text(image, window\_name) != 0) return 0;

if (Displaying\_Big\_End(image, window\_name) != 0) return 0;

waitKey(0);

return 0;

}

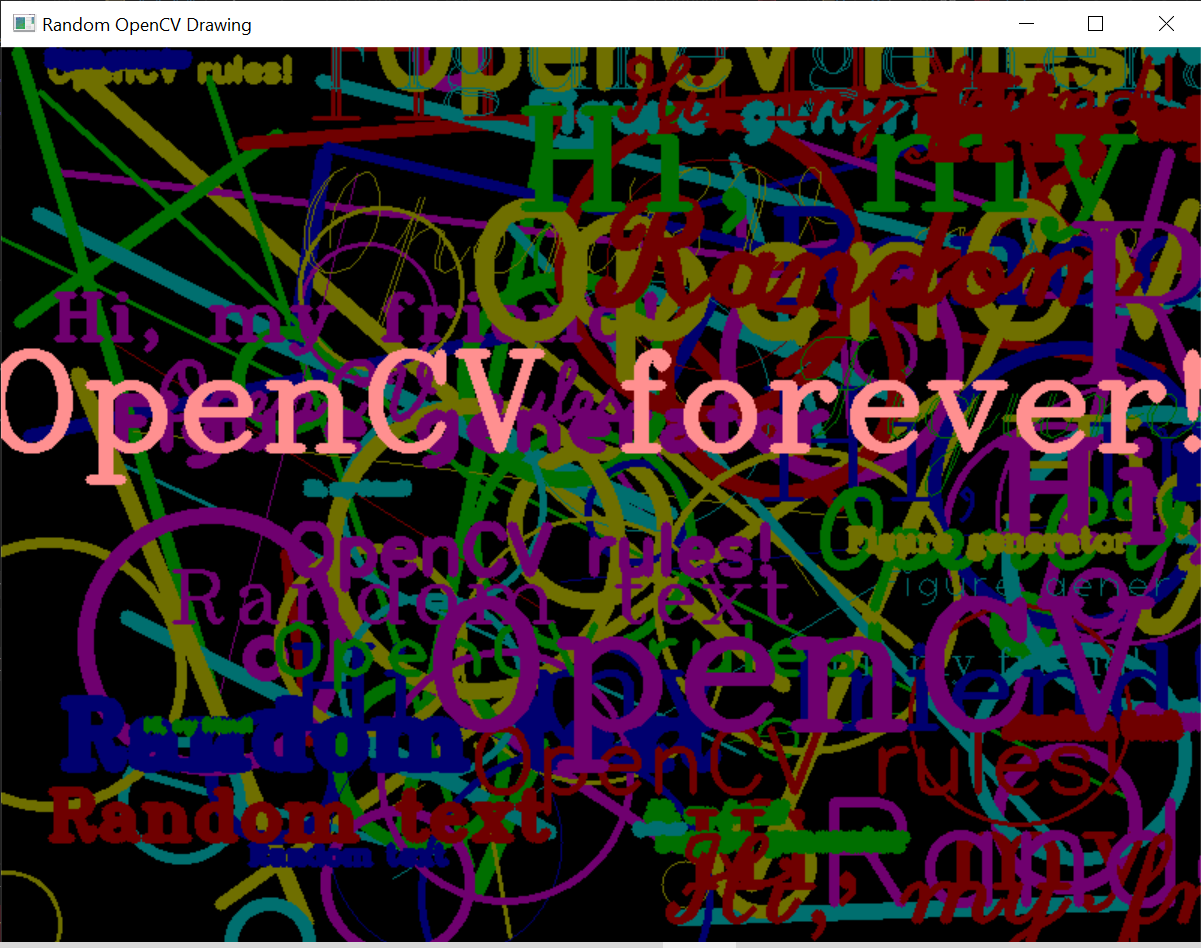
##### Отличия от оригинального примера:

* Все генерации случайных параметров перенесены в std::uniform\_\*\_distribution, что даёт лучшее управление над диапазонами.
* Вместо cv::RNG применён стандартный C++ генератор чисел.
* Разделение кода по функциям с чистой архитектурой и понятными интерфейсами.

**4. Результаты**

После выполнения всех шагов программа успешно скомпилирована и запущена. На экране поочерёдно отрисовываются:

* Случайные линии и окружности.
* Текстовые сообщения в разных позициях, цветах, шрифтах и размерах.
* Завершающее сообщение "OpenCV forever!" с эффектом осветления.



**5. Выводы**

В ходе работы была настроена профессиональная среда для разработки с использованием Visual Studio и библиотеки OpenCV. Благодаря интеграции с современными средствами генерации случайных чисел и кастомизации визуального вывода, был реализован наглядный и расширяемый пример по визуализации данных.

Этот опыт может быть полезен при разработке более сложных графических приложений, включая визуализацию данных, генерацию шаблонов, интерфейсов и обучающих материалов.