Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный технический университет
имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления
Кафедра: Теоретическая информатика и компьютерные технологии

ДЗ №2
«Введение в CV на примере реализации задачи Key point detection на C++ и Python»
по курсу: «Языки и методы программирования»

Выполнил:
Студент группы ИУ9-21Б
Старовойтов А. И.

Проверил:
Посевин Д. П.

Москва, 2022

Цели

Знакомство с возможностями языка C++ и Python для реализации задач машинного зрения.

Задачи

Реализовать на C++ (см. п. 2.1.) и Python (см. п. 2.2.) под любую ОС по желанию студента следующие задачи:

1. Распознавание координат точек кисти со снимков получаемых с камеры, координаты точек выводятся списком в консоль в формате JSON.
2. Распознавание координат точек тела со снимков получаемых с камеры, координаты точек выводятся списком в консоль в формате JSON.
3. Сравнить скорость работы алгоритма распознавания кисти руки выполненного на C++ со скоростью распознавания выполненного на Python. В отчете привести сравнение скоростей.
4. Сравнить скорость распознавания кисти руки алгоритмом выполненным на языке Python в этом Модуле со скоростью алгоритма распознавания кисти руки на базе Mediapipe выполненным на языке Python в предыдущем Модуле №1. В отчете привести сравнение скоростей.
5. Сделать выводы.

Решение

Распознование кисти

```
python handPoseImage.py
time taken by network : 0.510
Total time taken : 0.620
```

```
./handPoseImage
Time Taken = 0.7435
```

Код для вывода в `JSON` на `Python`:

```
with open('output.json', 'w') as outfile:
    json.dump({"points": points}, outfile,
 ↪  ensure_ascii=False, indent=4)
```

Пример JSON вывода для Python:


```
{
    "points": [
        [
            263,
            638
        ],
        [
            365,
            570
        ],
        [
            450,
            519
        ],
        [
            518,
            434
        ],
        [
            518,
            349
        ],
        [
            348,
            366
        ],
        [
            365,
            230
        ],
        [
            382,
            161
        ],
        [
```

```
        382,
        76
    ],
    [
        280,
        366
    ],
    [
        297,
        212
    ],
    [
        280,
        111
    ],
    [
        280,
        25
    ],
    [
        229,
        383
    ],
    [
        212,
        230
    ],
    [
        195,
        161
    ],
    [
        195,
        76
    ],
    [
        161,
        400
    ],
    [
```

```
            127,
            315
        ],
        [
            127,
            247
        ],
        [
            110,
            161
        ],
        null
    ]
}
```

Код для вывода в JSON на C++:

```
    FileStorage fs("output.json", FileStorage::WRITE);

    fs << "points" << "[";
    for (auto& el : points) {
        fs << el;
    }
    fs << "]";
```

Пример JSON вывода для C++:

```
{
    "points": [
        [ 263, 638 ],
        [ 365, 570 ],
        [ 450, 519 ],
        [ 518, 434 ],
        [ 518, 349 ],
        [ 348, 366 ],
        [ 365, 230 ],
        [ 382, 161 ],
        [ 382, 76 ],
        [ 280, 366 ],
        [ 297, 212 ],
        [ 280, 111 ],
        [ 280, 25 ],
```

```
            [ 229, 383 ],
            [ 212, 230 ],
            [ 195, 161 ],
            [ 195, 76 ],
            [ 161, 400 ],
            [ 127, 315 ],
            [ 127, 247 ],
            [ 110, 161 ],
            [ 127, 247 ]
        ]
}
```

Распознование тела

```
python3 OpenPoseImage.py
Using CPU device
time taken by network : 0.979
Total time taken : 1.352
```

```
./OpenPoseImage
USAGE : ./OpenPose <imageFile>
USAGE : ./OpenPose <imageFile> <device>
Using CPU device
Time Taken = 1.0332
```

Код для вывода в JSON одинаков.

Пример вывода в JSON из C++:

```
{
    "points": [
            [ 348, 125 ],
            [ 334, 230 ],
            [ 250, 271 ],
            [ 237, 376 ],
            [ 223, 480 ],
            [ 417, 271 ],
            [ 417, 397 ],
            [ 390, 501 ],
            [ 292, 501 ],
            [ 306, 668 ],
            [ 306, 835 ],
```

```
        [ 348, 501 ],
        [ 348, 668 ],
        [ 334, 793 ],
        [ 334, 376 ]
    ]
}
```

Пример вывода в JSON из Python:

```
{
    "points": [
        [
            361,
            187
        ],
        [
            333,
            271
        ],
        [
            250,
            271
        ],
        [
            236,
            375
        ],
        [
            208,
            480
        ],
        [
            417,
            271
        ],
        [
            417,
            396
        ],
        [
            389,
```

```
            500
        ],
        [
            292,
            500
        ],
        [
            306,
            667
        ],
        [
            306,
            855
        ],
        [
            361,
            500
        ],
        [
            361,
            667
        ],
        [
            333,
            813
        ],
        [
            347,
            187
        ],
        [
            375,
            166
        ],
        [
            306,
            166
        ],
        null
    ]
```

```
}
```

Вывод

Т.к. python-opencv это лишь “обертка” для обычных C++ функций из opencv, разницы в скорости нет.