Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Летучка №4

по курсу «Языки и методы программирования»

«Введение в CV на примере распознавания ArUco маркеров»

Студент группы ИУ9-22Б Лавров Р. Д.

Преподаватель Посевин Д. П.

Москва 2025

1 Задание

4.1. Реализовать пример detect-markers.cpp 4.2. Реализовать вывод координат углов ArUco меток Задачи 4.1 в консоль.

2 Результаты

Листинг 1 — detect-markers.cpp

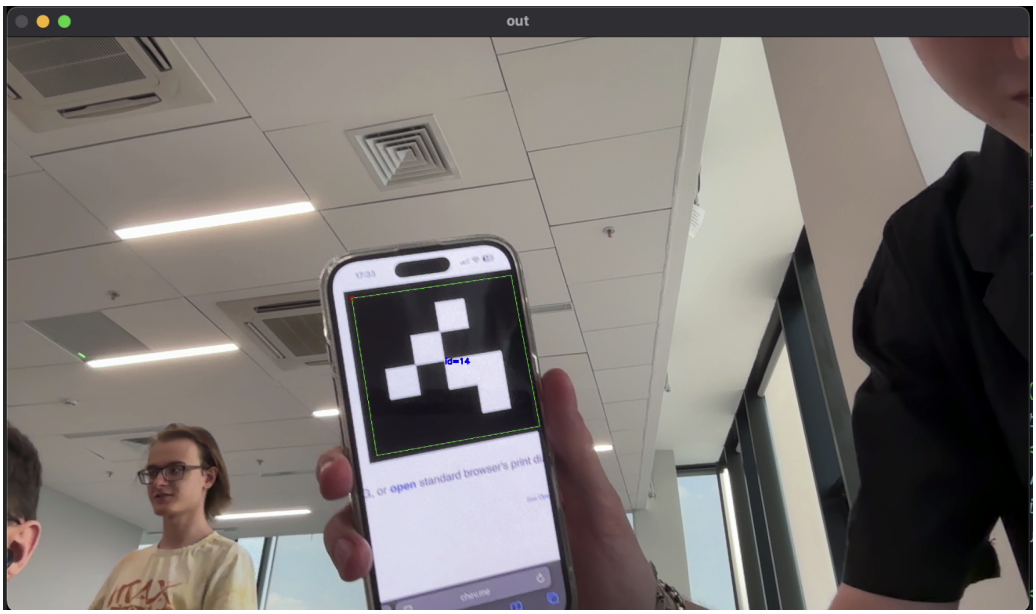
```cpp
int main(int argc, char *argv[]) {
    CommandLineParser parser(argc, argv, keys);
    parser.about(about);

    if(argc < 2) {
        parser.printMessage();
        return 0;
    }

    bool showRejected = parser.has("r");
    bool estimatePose = parser.has("c");
    float markerLength = parser.get<float>("l");

    aruco::DetectorParameters detectorParams;
    if(parser.has("dp")) {
        FileStorage fs(parser.get<string>("dp"), FileStorage::READ);
        bool readOk = detectorParams.readDetectorParameters(fs.root());
        if(!readOk) {
            cerr << "Invalid detector parameters file" << endl;
            return 0;
        }
    }

    if (parser.has("refine")) {
        //override cornerRefinementMethod read from config file
        detectorParams.cornerRefinementMethod = parser.get<aruco::CornerRefineMethod>("
refine");
    }
    std::cout << "Corner refinement method (0: None, 1: Subpixel, 2:contour, 3: AprilTag 2): "
 << (int)detectorParams.cornerRefinementMethod << std::endl;

    int camId = parser.get<int>("ci");

    String video;
    if(parser.has("v")) {
        video = parser.get<String>("v");
    }
```

Листинг 2 — detect-markers.cpp (продолжение 1)

```
    if (!parser.check()) {
        parser.printErrors();
        return 0;
    }

    aruco::Dictionary dictionary = aruco::getPredefinedDictionary(0);
    if (parser.has("d")) {
        int dictionaryId = parser.get<int>("d");
        dictionary = aruco::getPredefinedDictionary(aruco::PredefinedDictionaryType(
    dictionaryId));
    }
    else if (parser.has("cd")) {
        FileStorage fs(parser.get<std::string>("cd"), FileStorage::READ);
        bool readOk = dictionary.aruco::Dictionary::readDictionary(fs.root());
        if (!readOk) {
            std::cerr << "Invalid dictionary file" << std::endl;
            return 0;
        }
    }
    else {
        std::cerr << "Dictionary not specified" << std::endl;
        return 0;
    }

    Mat camMatrix, distCoeffs;
    if (estimatePose) {
        bool readOk = readCameraParameters(parser.get<string>("c"), camMatrix, distCoeffs);
        if (!readOk) {
            cerr << "Invalid camera file" << endl;
            return 0;
        }
    }
    aruco::ArucoDetector detector(dictionary, detectorParams);
    VideoCapture inputVideo;
    int waitTime;
    if (!video.empty()) {
        inputVideo.open(video);
        waitTime = 0;
    } else {
        inputVideo.open(camId);
        waitTime = 10;
    }

    double totalTime = 0;
    int totalIterations = 0;

    // Set coordinate system
    cv::Mat objPoints(4, 1, CV_32FC3);
    objPoints.ptr<Vec3f>(0)[0] = Vec3f(-markerLength/2.f, markerLength/2.f, 0);
    objPoints.ptr<Vec3f>(0)[1] = Vec3f(markerLength/2.f, markerLength/2.f, 0);
    objPoints.ptr<Vec3f>(0)[2] = Vec3f(markerLength/2.f, -markerLength/2.f, 0);
    objPoints.ptr<Vec3f>(0)[3] = Vec3f(-markerLength/2.f, -markerLength/2.f, 0);
```

Листинг 3 — detect-markers.cpp (продолжение 2)

```
    while(inputVideo.grab()) {
        Mat image, imageCopy;
        inputVideo.retrieve(image);

        double tick = (double)getTickCount();

        vector< int > ids;
        vector< vector< Point2f > > corners, rejected;

        // detect markers and estimate pose
        detector.detectMarkers(image, corners, ids, rejected);

        size_t nMarkers = corners.size();
        vector<Vec3d> rvecs(nMarkers), tvecs(nMarkers);

        if (estimatePose && !ids.empty()) {
            // Calculate pose for each marker
            for (size_t i = 0; i < nMarkers; i++) {
                solvePnP(objPoints, corners.at(i), camMatrix, distCoeffs, rvecs.at(i), tvecs.at(
    i));
            }
        }
        double currentTime = ((double)getTickCount() - tick) / getTickFrequency();
        totalTime += currentTime;
        totalIterations ++;
        // if( totalIterations % 30 == 0) {
        //     cout << "Detection Time = " << currentTime * 1000 << " ms "
        //          << "(Mean = " << 1000 * totalTime / double(totalIterations) << " ms)"
    << endl;
        // }
        if (!ids.empty()) {
            for (size_t i = 0; i < ids.size(); ++i) {
                cout << "Marker ID: " << ids[i] << endl;
                const vector<Point2f>& corner = corners[i];
                for (size_t j = 0; j < corner.size(); ++j) {
                    cout << " Corner " << j << ": (" << corner[j].x << ", " << corner[j].y
    << ")" << endl;
                }
                cout << endl;
            }
        }
        // draw results
        image.copyTo(imageCopy);
        if (!ids.empty()) {
            aruco::drawDetectedMarkers(imageCopy, corners, ids);

            if(estimatePose) {
                for(unsigned int i = 0; i < ids.size(); i++)
                    cv::drawFrameAxes(imageCopy, camMatrix, distCoeffs, rvecs[i], tvecs[i],
    markerLength * 1.5f, 2);
            }
        }
        if(showRejected && !rejected.empty())
            aruco::drawDetectedMarkers(imageCopy, rejected, noArray(), Scalar(100, 0, 255));

        imshow("out", imageCopy);
        char key = (char)waitKey(waitTime);
        if(key == 27) break;
    }
    return 0;
}
```



3 Вывод

Благодаря этой летучке, я познакомисля с библиотекой OpenCV и её возможностями.