おまけ

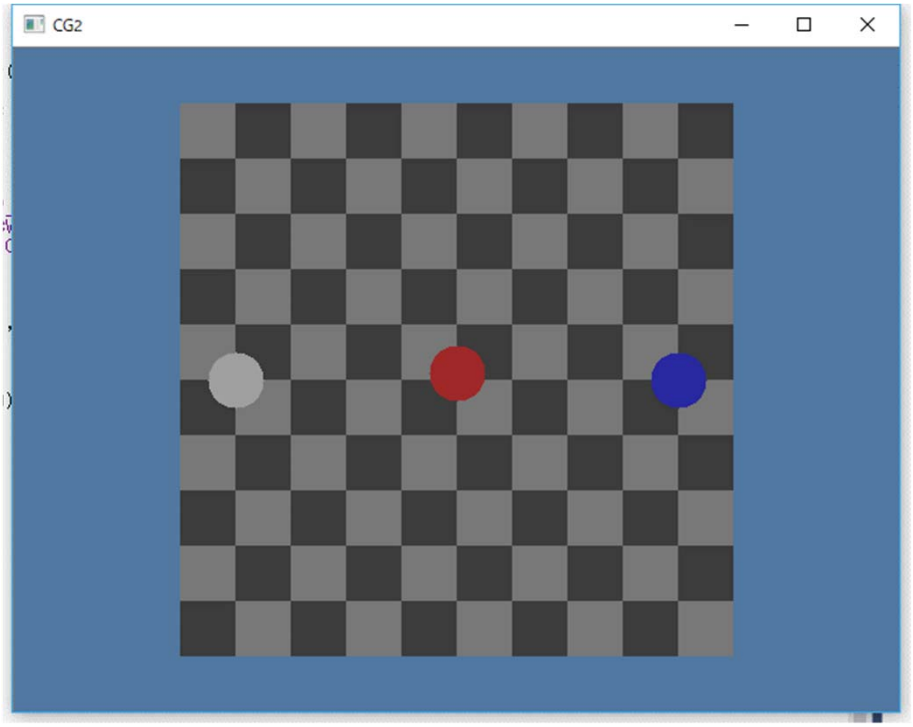
- この資料は授業とは関係ないので，参考程度に読んでください
- 解説はしません

OpenCV

- OpenCVはコンピュータビジョンのライブラリであってコンピュータグラフィックスのライブラリではない

サンプルプログラム

- 私の作成したサンプルソースコードを載せるが，参考にしないように
  - OpenCVはコンピュータビジョンのライブラリであってコンピュータグラフィックスのライブラリではない
  - このサンプルはかなり雑に作ってある



サンプルソースコード

```
// [プロジェクト]-[プロパティ]で[構成]を[すべての構成]に
// [構成プロパティ]-[C/C++]-[全般]-[追加のインクルード ディレクトリ]
// [構成プロパティ]-[リンカ]-[全般]-[追加のライブラリ ディレクトリ]
// Include directory C:¥OpenCV3.3.1¥build¥include
// Library directory (x64) C:¥OpenCV3.3.1¥build¥x64¥vc14¥lib
// PATH (x64) C:¥OpenCV3.3.1¥build¥x64¥vc14¥bin
#if _DEBUG
#pragma comment(lib, "opencv_world331d.lib")
#else
#pragma comment(lib, "opencv_world331.lib")
#endif

#include <opencv2/opencv.hpp>
using namespace cv;
```

```
int main()
{
    // ウィンドウの生成
    Mat img = Mat::zeros(480, 640, CV_8UC3);
    namedWindow("CG2", WINDOW_AUTOSIZE);

    // 描画
    Point2d item(4 * 40 + 320, 0 * 40 + 240);
    Point2d self(-4 * 40 + 320, 0 * 40 + 240);
    Point2d enemy(0 * 40 + 320, 0 * 40 + 240);
    Point2d enemy_v(0.0, 5.0);
    const double WALKSPEED = 5.0;
    bool nowgaming = true;
    int64 start = getTickCount();
    for(;;) {
        // フレームの表示時間
        int64 end = getTickCount();
        if ((end - start) * 1000 / getTickFrequency() < 10) continue;

        // ゲーム終了
        if (!nowgaming) {
            if ((end - start) * 1000 / getTickFrequency() < 3000) continue;
            break;
        }

        // 時間の更新
        start = end;

        // 消去
        img = Scalar(160, 120, 80);

        // 床
        for (int i = -5; i < 5; i++) {
            for (int j = -5; j < 5; j++) {
                Scalar col;
                if (abs(i + j) % 2 == 0) col = Scalar(120, 120, 120);
                else col = Scalar(60, 60, 60);
                rectangle(img, Rect(j * 40 + 320, i * 40 + 240, 40, 40), col, CV_FILLED);
            }
        }

        // 円
        circle(img, item, 20, Scalar(160, 40, 40), CV_FILLED);
        circle(img, enemy, 20, Scalar(40, 40, 160), CV_FILLED);
        circle(img, self, 20, Scalar(160, 160, 160), CV_FILLED);

        // 判定
        if (norm(self - enemy) < 40.0) {
            putText(img, "GAME OVER", Point(120, 240), FONT_HERSHEY_SIMPLEX, 2, Scalar(255, 255, 255), 2);
            nowgaming = false;
        }
        if (norm(self - item) < 40.0) {
            putText(img, "GAME CLEAR", Point(120, 240), FONT_HERSHEY_SIMPLEX, 2, Scalar(255, 255, 255), 2);
            nowgaming = false;
        }

        // ウィンドウに表示
        imshow("CG2", img);

        // キーボード
        int key = waitKeyEx(1);
        if (key == 0x1b) break;
        if (key == 2424832) self.x -= WALKSPEED;
        if (key == 2490368) self.y -= WALKSPEED;
        if (key == 2555904) self.x += WALKSPEED;
        if (key == 2621440) self.y += WALKSPEED;
        waitKey(1);

        // 敵の動作
        enemy += enemy_v;
        if (enemy.y > 5 * 40 + 240) enemy_v.y = -enemy_v.y;
        if (enemy.y < -5 * 40 + 240) enemy_v.y = -enemy_v.y;
    }

    // 終了処理
    return 0;
}
```