浙江大学

计算机视觉作业报告

作业名称：完成相机参数的标定和俯瞰视角的转换

姓　　名：陈润健

学　　号：3160103989

电子邮箱：3160103989@zju.edu.cn

联系电话：18868104871

导　　师：潘纲

2018 年　　12 月　　25　日

完成相机参数的标定和俯瞰视角的转换

一、 作业已实现的功能简述及运行简要说明

1. 实现相机参数标定的程序。
2. 分别标定数据库的标定照片和自己拍的照片。
3. 分别对数据库的标定照片和自己拍的照片进行矫正和视角的转换。

二、 作业的开发与运行环境

IDE：Xcode 10.1

操作系统：MacOS 10.14.1

SDK：macosx10.14

三、 基本思路，用到的函数及流程

1. 相机参数的定义：

1) 内部参数：表达的是聚焦平面与成像平面之间的关系。

$$\begin{bmatrix} -f_u & 0 & u_0 & 0 \\ 0 & -f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -fk_u & 0 & u_0 & 0 \\ 0 & -fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

a) f表示焦距。

b) k_u，k_v分别表示从聚焦平面到成像平面投影的过程中 x，y 方向上的伸缩。

c) u_0，v_0分别表示从聚焦平面到成像平面投影的过程中 x，y 方向上的位移。

2) 畸变参数：

a) 对于透镜造成的畸形，有：

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

其中，k_1，k_2，k_3是参数。

b) 对于正切畸形，有：

$$x_{corrected} = x + 2p_1 y + p_2(r^2 + 2x^2)$$

$$y_{corrected} = y + 2p_2 x + p_1(r^2 + 2y^2)$$

3) 外部参数：表达的是相机的位姿，其中 R 是 3*3 的旋转矩阵，t 是 3*1 的位移向量。

$$\begin{bmatrix} R & t \\ 0 & 0 \end{bmatrix}$$

2. 计算相机内参的总体思路：

   1) 找到图像中棋盘的角点。

   2) 进一步计算图像中棋盘的亚像素级角点，使计算更加精确。（得到精确的 corners）

   3) 计算聚焦平面上的棋盘点的位置。（得到 object_points）。

   4) 通过 opencv 的函数计算相机的内外参。

   5) 通过畸形参数对图像进行矫正。

3. 做鸟瞰图转化的思路：

   1) 对斜拍照片进行矫正。

   2) 找到图像中棋盘的角点。

   3) 进一步计算图像中棋盘的亚像素级角点，使计算更加精确。（得到精确的 corners）

   4) 通过 opencv 的透射变换进行视角的转换。其原理是：先找到图片中棋盘的四个角，而我们的目标是要在新的图像上，这四个角位于一个正方形内，看起来才像是鸟瞰图，因此把这四个角的像素位置投射到期望的四个顶点的像素上，就得到了我们想要的鸟瞰图。

4. 程序的构架：

   1) 类的设计：

```
class ChenRJ_Camera_Calibrate
{
private:
    Mat Intrinsic_Parameter;
    Mat Distorn_Parameter;
    Mat Extrinsic_Parameter_r,Extrinsic_Parameter_t;

    vector<Mat> Raw_Data;

    void Calculate_Parameter();

    int Board_Width,Board_Height,Board_Size;

public:
    ChenRJ_Camera_Calibrate(vector<Mat>* Input_Data,int H,int W):Raw_Data(*Input_Data)
    {
        Raw_Data = *Input_Data;
        Board_Height = H;
        Board_Width = W;
        Board_Size = Board_Width * Board_Height;
        Calculate_Parameter();
    }
    ~ChenRJ_Camera_Calibrate(){};
    Mat Bird_View(Mat Source);
    Mat Get_Intrinsic_Parameter()
    {
        return Intrinsic_Parameter;
    }
    Mat Get_Distorn_Parameter()
    {
        return Distorn_Parameter;
    }
    void Get_Extrinsic_Parameter(Mat *R,Mat *t)
    {
        *R = Extrinsic_Parameter_r;
        *t = Extrinsic_Parameter_t;
    }
};
```

2） 参数计算函数。

```
void ChenRJ_Camera_Calibrate::Calculate_Parameter()
{
    bool* Proper_data = new bool[Raw_Data.size()];
    vector<vector<Point3f>> object_points(1);
    int count = -1;
    Size Board_sz = Size(Board_Width,Board_Height);
    vector<vector<Point2f>> corners(0);

    Size win = Size(11,11),Zeros = Size(-1,-1);

    TermCriteria criteria = TermCriteria( CV_TERMCRIT_EPS+CV_TERMCRIT_ITER, 30, 0.1 );

    //preprocess
    for (int i = 0 ; i < Raw_Data.size() ; i++)
    {
        vector<Point2f> corner;

        bool corners_found ;

        corners_found = findChessboardCorners(Raw_Data[i],Board_sz,corner);

        if (!corners_found)
        {
            cout << "Error in processing " << i <<"th photo" << endl;
            Proper_data[i] = false;
            continue;
        }

        Proper_data[i] = true;

        Mat *now_im = new Mat(Raw_Data[i]), gray;

        cvtColor(*now_im , gray, CV_BGR2GRAY);

        count++;

        object_points[0].clear();

        float squareSize = 0.005;

        for( int i = 0; i < Board_Height; ++i )
            for( int j = 0; j < Board_Width; ++j )
                object_points[0].push_back(Point3f(float( j*squareSize ), float( i*squareSize ), 0));

        object_points.resize(Raw_Data.size(), object_points[0]);

        cornerSubPix( gray , corner , win , Zeros , criteria );

        corners.push_back(corner);
    }

    //caculate camera parameters
    Mat cameraMatrix,distCoeffs,rvecs, tvecs;

    calibrateCamera(object_points, corners, Board_sz, cameraMatrix, distCoeffs,rvecs, tvecs);

    Intrinsic_Parameter = cameraMatrix;
    Extrinsic_Parameter_t = tvecs;
    Extrinsic_Parameter_r = rvecs;
    Distorn_Parameter = distCoeffs;
}
```

3） 视角变换函数：

```
Mat ChenRJ_Camera_Calibrate::Bird_View(Mat Source)
{
    Mat bird_view = Source;

    //undistort

    Size imageSize = Size(1200,1600);
    Mat map1 , map2;

    initUndistortRectifyMap(Intrinsic_Parameter, Distorn_Parameter, Mat(),
                            getOptimalNewCameraMatrix(Intrinsic_Parameter, Distorn_Parameter, imageSize, 1, imageSize, 0),
                            imageSize, CV_16SC2, map1, map2);

    remap(bird_view, bird_view, map1, map2, INTER_LINEAR);

    //find corners in source map

    Size Board_sz = Size(Board_Width,Board_Height);

    vector<Point2f> corner;

    bool corners_found ;

    Size win = Size(11,11),Zeros = Size(-1,-1);

    TermCriteria criteria = TermCriteria( CV_TERMCRIT_EPS+CV_TERMCRIT_ITER, 30, 0.1 );

    corners_found = findChessboardCorners(bird_view,Board_sz,corner);

    Mat bird_gray;

    cvtColor(bird_view , bird_gray, CV_BGR2GRAY);

    cornerSubPix( bird_gray , corner , win , Zeros , criteria );

    //tranform

    Point2f dst_points[] = {
        Point2f(400, 1200),
        Point2f(700, 1200),
        Point2f(400 ,1500),
        Point2f(700, 1500) };

    Point2f src_points[] = {
        corner[0],
        corner[Board_Width-1],
        corner[(Board_Height-1)*Board_Width],
        corner[(Board_Height-1)*Board_Width+Board_Width-1]};

    Mat M = getPerspectiveTransform(src_points, dst_points);

    warpPerspective(bird_view, bird_view, M, Size(1200,1600));


    return bird_view;
}
```

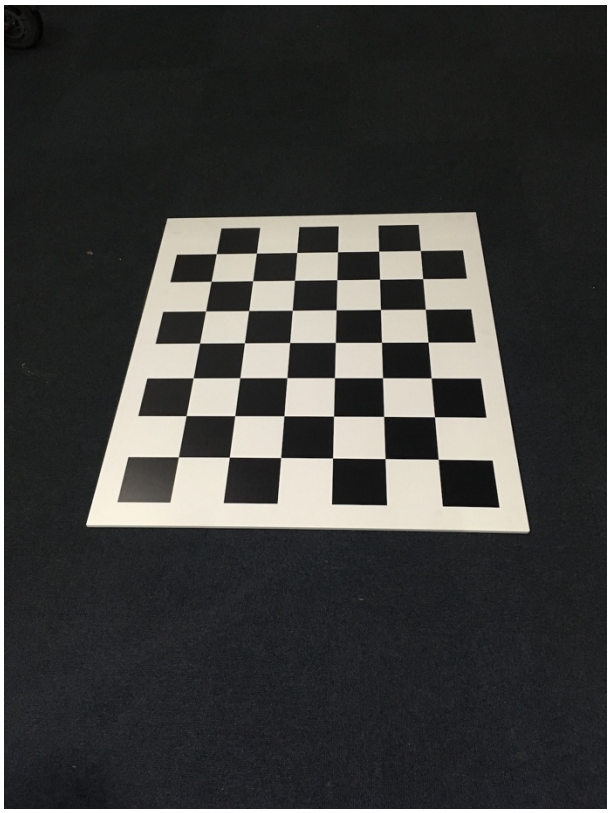
四、 实验结果与分析：

1. 使用手机拍的照片做出的结果：

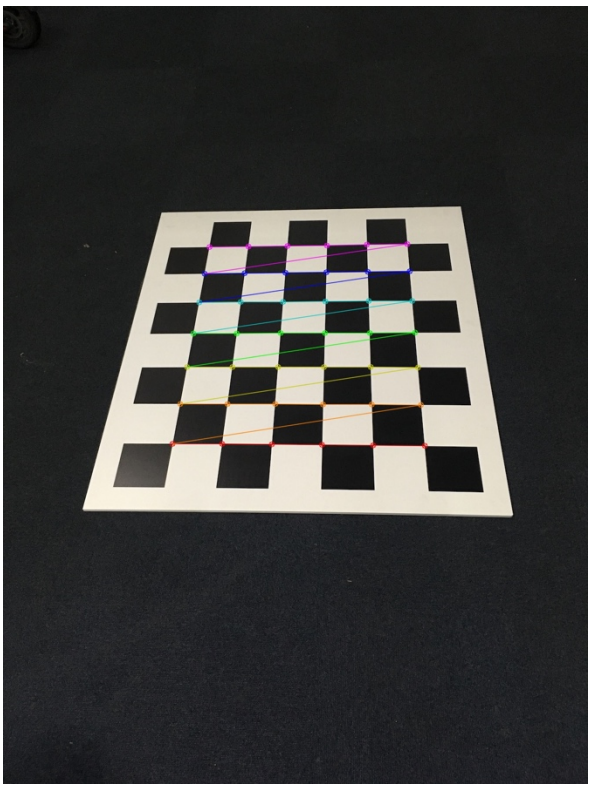
1） 相机参数：（11 张照片，外参有 11 组）

```
Intrinsic_Parameter :
[1716.946088640205, 0, 151.2699880479101;
 0, 1722.942872743044, 301.2600592727103;
 0, 0, 1]
Distorn_Parameter
[-0.3545453950554084, -0.2928413277406844, 0.03054397128532346, 0.02381544232882315, 0.5552179772703254]
Extrinsic_Parameter_R :
[-0.1132467409955921, -0.4213571329215172, 1.395451205376742;
 0.07951564330583979, -0.1946120049051225, 1.481949151538093;
 0.3653613225804763, 0.05778455365887349, 1.516788975854585;
 0.5734175966439443, 0.3422669655435872, 1.545215646971356;
 -0.9886003928177748, -0.4893805361527334, -1.314170865086046;
 -1.001681383833105, -0.4766916769763973, -1.276288006117752;
 -0.9938539569685936, -0.2580617991112275, -0.971551346118721;
 -0.7748865439528239, -0.7164376245985413, -1.737361571152897;
 -0.7209762203444957, -0.8899332091773484, -2.042366153257951;
 -0.6196026986006539, -1.186172452962646, -2.32821785322487;
 -0.1530644437693704, -0.4437368773042252, -0.8528610368715094]
Extrinsic_Parameter_t :
[0.06481838736416148, 0.01881908239879869, 0.1658318755936997;
 0.06895095477070835, 0.02613850113017384, 0.1649037355151082;
 0.07088960889546871, 0.02795154800494677, 0.1605612235283631;
 0.07296903762191734, 0.03791236306397564, 0.1604649505034881;
 0.01899538591091752, 0.05789395651338741, 0.1660776765595765;
 0.01942696114913984, 0.04841531709914368, 0.1570883274070403;
 0.01086044040724656, 0.0399564685317897, 0.182842899038293;
 0.03375226079097111, 0.0524462244863039, 0.1597161099996112;
 0.04701958555839489, 0.05291318945814867, 0.1600331328568484;
 0.04860695124290971, 0.05939605399898455, 0.1715734415175559;
 0.01747434873247461, 0.05769614250331753, 0.1860840907714324]
```

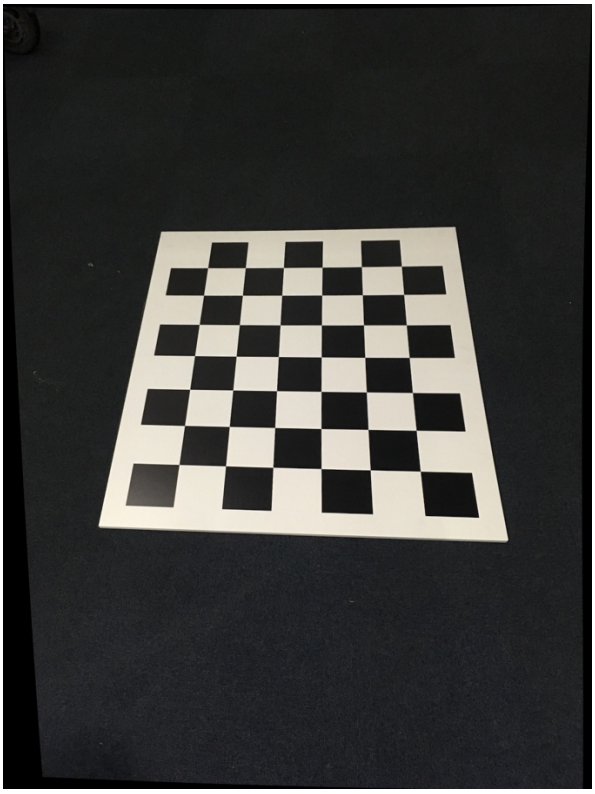
2） 输入的原始图像：



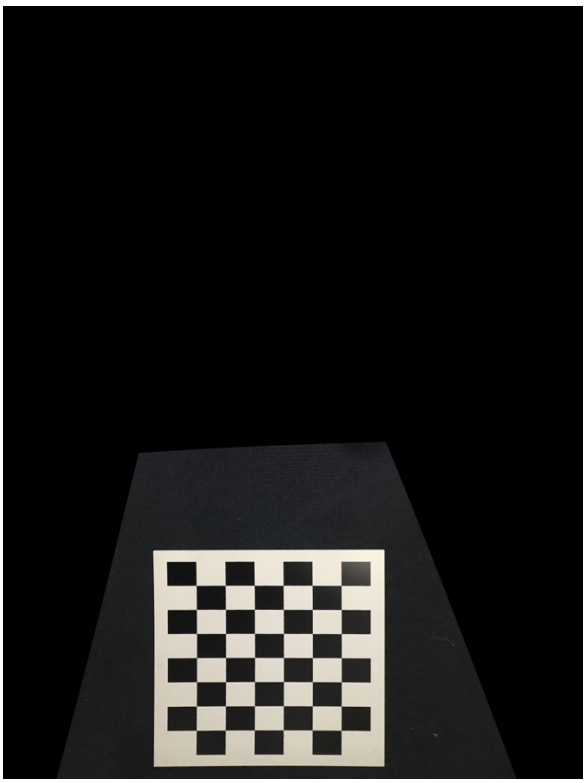
3） 角点检测图：



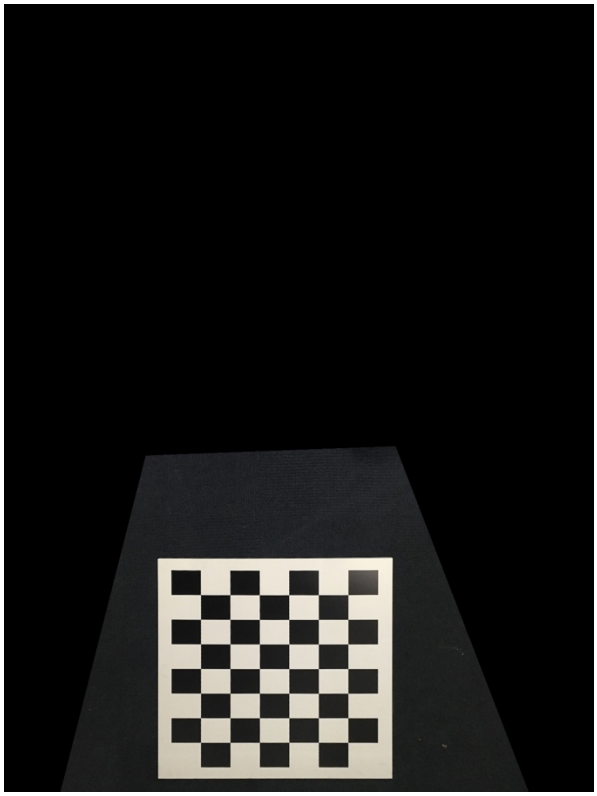
4） 矫正之后的结果：



5） 变换视角之后的结果：



6） 不作矫正得到的变换视角的效果：可以看到边缘有一点枕形突变。



2. opencv 数据库 1（cameracalibrate）的结果：

1） 相机参数（12 张照片，外参有 12 组）：

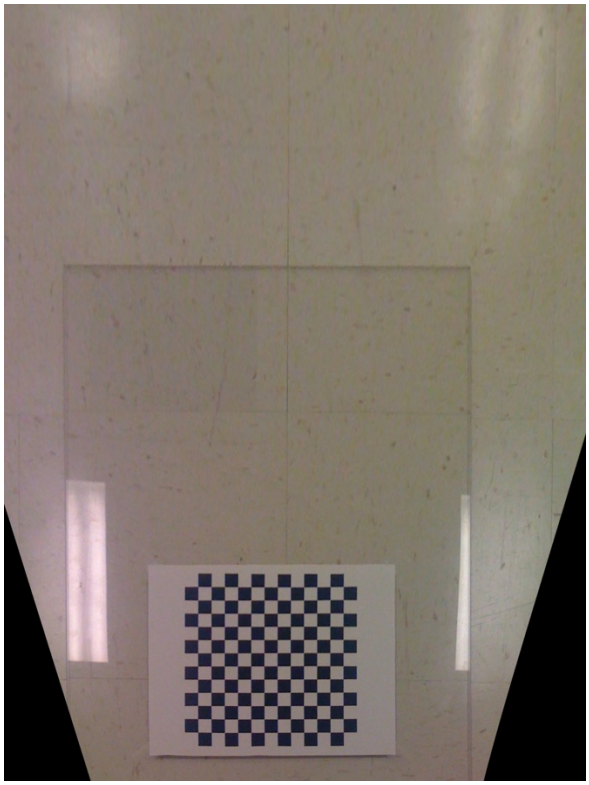
```
Intrinsic_Parameter :
[2577.105036123817, 0, 311.5869385820288;
 0, 565.6855667119128, 57.43166177808181;
 0, 0, 1]
Distorn_Parameter
[-0.007556133916657303, 0.006649306656008987, -0.03720373637585449, -0.01792593345580195, -0.0003484035787896298]
Extrinsic_Parameter_R :
[-1.200587255168263, -0.2365942768999238, 0.2104703759321478;
 -1.239105896602297, 0.3620584150651515, -0.2258506581692954;
 -1.893994968086178, 0.05163424865123325, 0.0699801147534236;
 -1.231202594511191, 0.2626725522000661, -0.1326759719108838;
 -1.894002839444866, 0.05162375808657313, 0.0699954319750378 3;
 -2.040439927221456, 0.02591209316791472, 0.08866546467720138;
 -2.040433701198666, 0.02590529616950313, 0.08867416104164878;
 -1.239105429141555, 0.3620865319756098, -0.2258712207210536;
 -1.397477368270685, 0.3025118115226522, -0.1645965701509445;
 -1.397450804747926, 0.3024927133907391, -0.1645879761655311;
 -1.231057454579687, 0.262714661042902, -0.1327069849025062;
 -1.200594932856769, -0.2365363856205889, 0.2104364480999134]
Extrinsic_Parameter_t :
[0.09767624951999636, 1.628683255009453, 0.7022818114814304;
 0.09185907024419 98, 1.699975330745669, 0.6942402999681371;
 0.07611168439442788, 1.449513884207495, 0.6589018491731376;
 0.1453018988440692, 2.211694391847146, 0.9358163400642227;
 0.07611193947443089, 1.449516694288958, 0.6589024877386461;
 0.05964517836187003, 1.488374676565738, 0.6486273638148801;
 0.05964439713731802, 1.488364592437811, 0.6486241044031383;
 0.09185741633372077, 1.699949748820878, 0.6942308563889177;
 0.1500669908541812, 2.404412863036877, 0.9553695390668177;
 0.1500667619638168, 2.404407293930221, 0.9553675131163428;
 0.1453082318126244, 2.211676229787778, 0.9358107543395785;
 0.09767757069349399, 1.628710117314145, 0.7022929598343607]
```

2） 输入的原始图像：

3） 角点检测得到的图像：

4） 矫正之后的图像：

5） 转换视角之后的图像：



3. opencv 数据库 2 的结果：

1） 相机参数（20 张照片，外参有 20 组）：

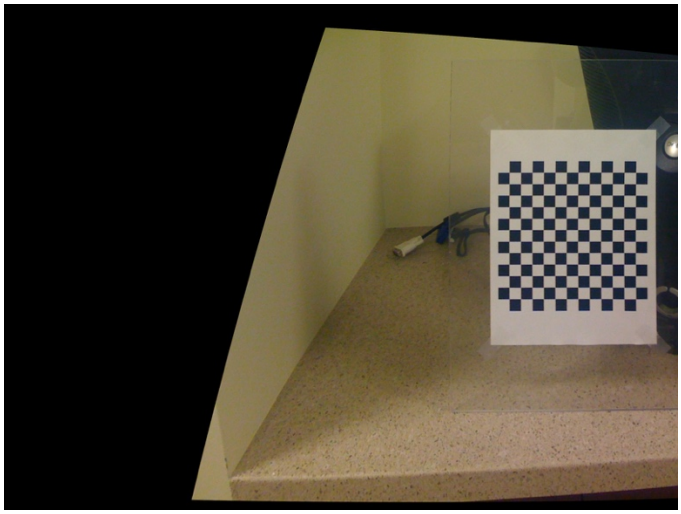
```
Intrinsic_Parameter :
[1805.788879238709, 0, 600.311989695371;
 0, 1794.805499860696, 566.9739476093789;
 0, 0, 1]
Distorn_Parameter
[0.05415408025664079, 0.1302274627789264, -0.005788378282827118, -0.05078103303453785, 0.15560799574809]
Extrinsic_Parameter_R :
[0.467457127199867, 0.8821540517038767, 0.4827920301527824;
 0.3228868257615145, 0.1767777220680538, 0.015270766540847;
 0.4526934147234565, -0.2715336318585206, -0.2645682504143106;
 0.4120045904607541, 0.2469862904456298, -0.09637232835918838;
 0.700658250239961, 0.2814913825713261, -0.08060318171000221;
 0.3438679590988122, 0.886561391857095, 0.3918497303265799;
 0.5485736590024175, 1.176133477229812, 0.5960318030650453;
 0.4133211216070631, 0.9787075137177984, 0.3033371801588425;
 0.3015385374079601, 0.1623010487469634, -0.01839190903013005;
 0.5733605942118311, 0.04165784538220346, -0.1502755946790283;
 0.2475022701485099, 0.6911054987373397, 0.8011684242754719;
 0.4960679603425964, -0.3335798130823255, -0.508071122970559 8;
 0.2145434149103659, -0.1913965497731192, -0.3576471313218211;
 0.9357744925727488, 0.1235151259016388, -0.04414265046923712;
 0.7958864307226126, 0.2026718184091573, -0.005023988034032369;
 0.4965525546690424, 0.4833908052445062, 0.2612536170205103;
 0.222505709622972, 0.01125010187173122, -0.1264977827291095;
 0.169479162908669, 0.5982948299736487, 0.4874490391163053;
 0.5314679760626408, 0.817124279455699, 0.6980342851644353;
 0.6105198907159928, 0.6159848722601596, 0.5586720744754962]
Extrinsic_Parameter_t :
[0.07832334712114414, -0.05184962837847487, 0.4350564689227527;
 -0.01822213963358209, 0.003339937587332928, 0.4516227218571855;
 -0.04484682642722947, -0.02636344071413211, 0.3994658503506165;
 0.01989679719612535, -0.04141427910319115, 0.5139708526302852;
 -0.03492449307903975, 0.000306000762104874, 0.2794356880438442;
 0.04356821952758087, -0.08140185944690832, 0.4694897695552586;
 0.05249649357419235, -0.07253624345854646, 0.3598037432034036;
 0.06409703815462334, -0.06950241482558414, 0.5800179195809323;
 0.009167492472832054, -0.03980560678913397, 0.7367949112508363;
 -0.01779326541017332, -0.02170105710355125, 0.2629579994400422;
 0.07932680266945827, -0.06740558925462181, 0.4297914634857372;
 -0.04134236061700295, 0.007681761991871235, 0.3323743353864911;
 -0.000327838534642838 7, -0.01583481567136139, 0.6121709646245902;
 -0.02862746901255537, -0.007632268492284552, 0.2304645839295693;
 -0.01055099707215 88, 0.02683540294296662, 0.3542089544705921;
 -0.1074184250358838, -0.04992118249 1387, 0.4353531020668447;
 -0.08839214400867222, -0.0852151922018957, 0.4524737918524734;
 0.05305751712585353, -0.05094749767652249, 0.3869371328728516;
 -0.04233804539758128, -0.07922613621496612, 0.3995728895494469;
 0.04935897017552528, -0.07897370466665982, 0.3958565660392297]
```

2） 输入的原始图像：

3） 角点检测得到的图像：

4） 矫正之后的图像：

5） 转换视角之后的效果：（由于这一组棋盘不是放在地上的，视角转换的效果不是很好）



五、 结论与心得体会

在这一次的实验中，我实现了相机参数的标定以及视角变换的程序，做了多组实验，实现效果比较好。

希望在接下来的课程和作业中，能够接触到更多实用的算法，同时学习到更多关于计算机视觉的理论知识，尝试自己实现更多有趣的功能。