

# VR Player 畸变系数 $k_1k_2k_3$ 标定过程 (初版)

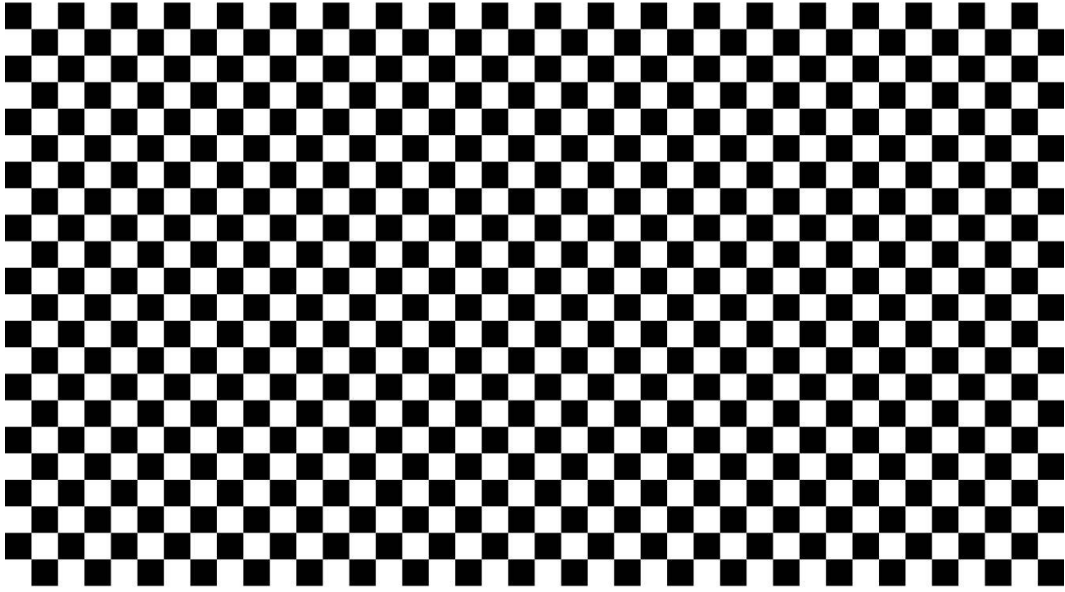


图 1 校正视频画面

- 1.以 2D 模式打开校正视频，将手机放入 VR 眼镜盒子
- 2.此时需要另外一个手机，模拟眼睛通过 VR 眼镜盒子看校正视频
- 3.将照片应用设置成正方形，拍照(5 张左右)，角落要拍到，如下图这里我们得到一个 12 乘 12 顶点的畸变画面

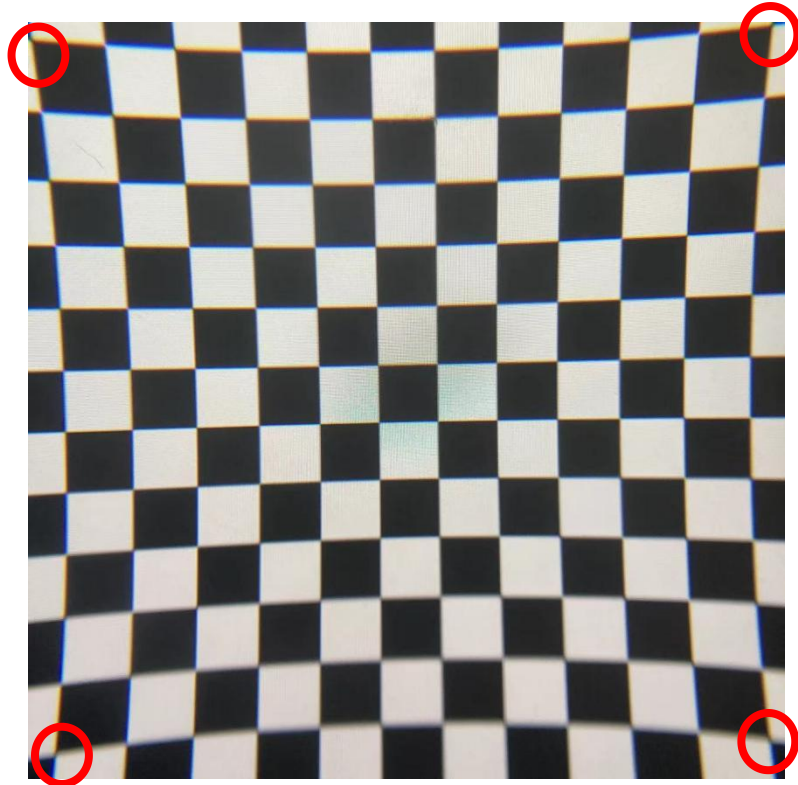


图 2 手机拍到的畸变画面 (也相当于人眼通过 VR 眼镜看到的画面)

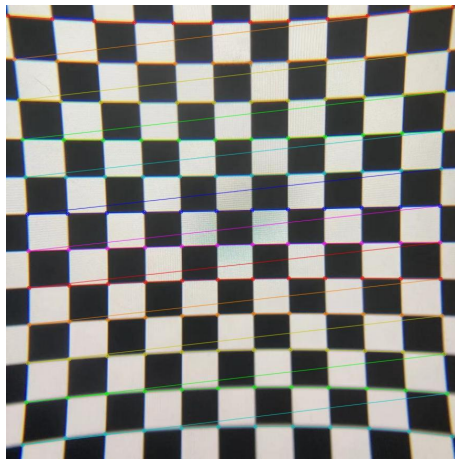


图 3 程序判定画面中的格子

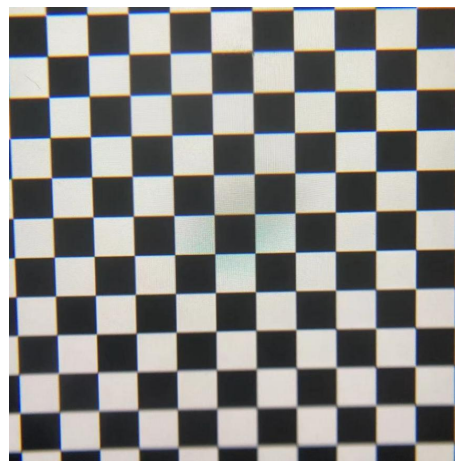


图 4 程序校正后的画面 (2.py 用来验证畸变系数是否正确)

打开 python 程序，设置标定板交叉点数

```
# 标定板交叉点的个数
row = 12
column = 12
```

将照片放入下面 source 文件夹中

```
# 批量读取图片
images = glob.glob('./pic/source/*.jpeg')
```

运行 python 程序 1.py 在下方得到畸变系数

```
畸变k1, k2, p1, p2, k3 :
[[ 9.83241293e+00 -3.61692224e+02 -2.04406978e-02 -2.77445246e-02
 1.57106091e+04]]
```

这里我们只需要  $k_1 k_2 k_3$

计算实际的畸变系数  $k_1 k_2 k_3$ ，需要用尺子测量 5 个照片中格子大小和手机屏幕中 5 个格子的大小，相除得到一个放大比例  $a$

在 OpenCV 中，相机标定的基本模型假设相机遵循针孔摄像机模型，并且图像可能会存在径向和切向畸变。径向畸变是指图像的变形是以图像中心为原点的，而切向畸变则是由于镜头和成像平面不完全平行引起的。

在这个模型中，径向畸变主要由三个参数  $k_1$ 、 $k_2$  和  $k_3$  来描述， $k_1$ 、 $k_2$  和  $k_3$  分别对应于径向畸变的一次、二次和三次项。切向畸变则由两个参数  $p_1$  和  $p_2$  来描述。

具体来说，假设 $(x, y)$ 是畸变图像中的一个点， $(x', y')$ 是校正后的点，那么他们之间的关系可以用下面的公式来描述：

...

$$x' = x + (2 * p1 * x * y + p2 * (r^2 + 2 * x^2))$$

$$y' = y + (p1 * (r^2 + 2 * y^2) + 2 * p2 * x * y)$$

...

其中  $r^2 = x^2 + y^2$ ，是从图像中心到点 $(x, y)$ 的欧氏距离的平方。

然后，再对 $(x', y')$ 应用径向畸变的修正：

...

$$x'' = x' * (1 + k1 * r'^2 + k2 * r'^4 + k3 * r'^6)$$

$$y'' = y' * (1 + k1 * r'^2 + k2 * r'^4 + k3 * r'^6)$$

...

其中  $r'^2 = x'^2 + y'^2$ 。

这个过程就是 OpenCV 中的相机标定的基本步骤，通过这种方式，我们可以得到每个像素点畸变的程度，从而对整个图像进行校正。这个过程通常需要一些已知的标定图像，比如棋盘格，通过观察棋盘格上的点在图像中的畸变，我们可以得到上述的  $k1$ 、 $k2$ 、 $k3$ 、 $p1$  和  $p2$  等参数。

这里我们需要将  $r=a*r$  带入到  $x'' = x' * (1 + k1 * r'^2 + k2 * r'^4 + k3 * r'^6)$ ，化简最终得到新的  $k1k2k3$  系数

$$k1\_new = k1 * a^2$$

$$k2\_new = k2 * a^4$$

$$k3\_new = k3 * a^6$$

$$\begin{array}{cc} \text{照片} & \text{屏幕} \\ 58.085 & / \quad 7.441 = 7.8060744524 \end{array}$$

$$\begin{array}{cc} \text{屏幕} & \text{照片} \\ 7.441 & / \quad 58.085 = \text{取5个格子计算放大比例} \end{array}$$

$$\begin{array}{cc} \text{取5个格子计算放大比例} & \\ 0.1281053628 & ^2 = \end{array}$$

$$\begin{array}{ccccc} & \text{程序得到的k1} & & \text{最终k1} & \\ 0.016410984 & \times & 9.83241293 & = & 0.1613595711 \end{array}$$

$$\begin{array}{cc} \text{取5个格子计算放大比例} & \\ 0.1281053628 & ^4 = \end{array}$$

$$\begin{array}{ccccc} & \text{程序得到的k2} & & \text{最终k2} & \\ 0.0002693204 & \times & ( - 3.61692224 ) & \times & 10^2 = -0.0974110928 \end{array}$$

$$\begin{array}{cc} \text{取5个格子计算放大比例} & \\ 0.1281053628 & ^6 = \end{array}$$

$$\begin{array}{ccccc} & \text{程序得到的k3} & & \text{最终k3} & \\ 0.0000044198 & \times & 1.57106091 & \times & 10^4 = 0.0694379496 \end{array}$$

最终我们近似得到新的  $k1=0.1614$ ， $k2=-0.0974$ ， $k3=0.0694$

参考程序链接: <https://pan.baidu.com/s/1ZDdKrYJ33-QoEPrNS6onWA> 密码: ujvv