题目：基于机器视觉的水准仪补偿误差。基于亚像素定位的水准仪补偿误差检定。

摘要：

关键字：

**0引言：**

水准仪是以仪器的水平视准线作为基准线，进行高差测量的计量器具。它广泛地使用于大地水准测量，地形变测量，各种工程水准测量与大型精密机械安装等。因其灵敏构件的不同又分为水准管水准仪，自动安平水准仪和数字水准仪。

自动安平水准仪补偿误差是水准仪计量性能要求中关键的一项，在标称补偿范围内，DSZ05级的补偿误差应不大于0.20″，DSZ1级的应不大于0.30″，DSZ3级的应不大于0.50″。现在的水准仪补偿误差检定方法为：将待检仪器放到微顷台上，整平，对准测微光管使仪器十字丝与测微光管横丝吻合，旋转微顷台纵向测微器，每次按约等于2′角度值倾斜，在补偿工作范围内，从，再由的顺序进行检定。每次倾斜一个角度时，读数两次，取平均值，求得仪器纵向补偿误差，同理，用横向测微器重复以上操作，测得仪器的横向补偿误差，分别取各方向的最大偏差值作为检定结果。



图像测量技术是一种以图像处理技术为核心的检测技术，本文提出一种以CCD摄像机代替人眼对水准仪补偿误差进行检定的方法。利用亚像素定位技术对图像十字丝中心位置进行定位，进而得到仪器的补偿误差。

几种常见的水准仪十字丝。



**1原理：**

使用CCD摄像头代替人眼进行水准仪补偿误差检定的基本思想是：使用高分辨率的摄像头，捕捉微顷台倾斜后十字丝的细微变化，使用亚像素级别的图像中心位置坐标的改变量，计算水准仪的竖直角改变量，从而进行水准仪的补偿误差检定。

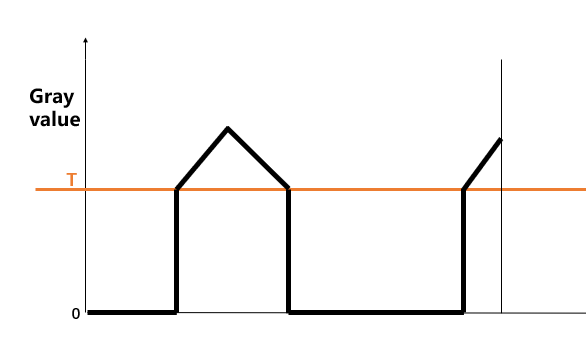
* 1. 基于灰度重心法的亚像素图像十字丝坐标计算模型。

经过灰度化处理的图像都可以看成一个由图像灰度值组成的二维矩阵。假设一副图像由行列像素组成，那么，图像第行的像素分布模型为，

其中为对应位置上的灰度值。图像第列的像素分布模型为，

其中为对应位置上的灰度值。

多种以重心法为基础的算法可以对目标图像进行亚像素定位，包括二值重心法，灰度重心法以及灰度值平方重心法。重心法进行亚像素定位时需设定阈值为，即只有灰度值大于的像素点会参与运算。对原图像阈值进行处理，即图像对应像素位置灰度值进行如下变换：



大于阈值的像素，其灰度值保存不变，对于小于阈值的像素，其灰度值则置为0。

提取出了图像的目标区域，设目标区域起始位置为，结束位置为，所以，在目标区域内，用灰度重心法计算第行的重心坐标为：

用灰度重心法计算第列的重心坐标为：

本文采用的是基于灰度值平方的重心法，即，第行的重心坐标为：

第列的重心坐标为：

平方加权重心法使用灰度值的平方代替原始灰度值，加强了灰度值较大（距离中心比较近）的像素对重心位置的影响，加大了信噪比高的像素点的权重，使算法抗干扰能力更强，精度更高。

记录一系列和点，对和进行最小二乘法拟合，拟合出两条直线，，。得到图像十字丝中心位置亚像素坐标为：

* 1. 图像十字丝坐标与水准仪水平角变换模型。

图像十字丝坐标与水准仪水平角以及竖直角的转换模型为：

其中，为相机轴的旋转角，、分别为CCD光管方向的角度格值，为CCD的轴与轴不垂直误差角，和为原点。

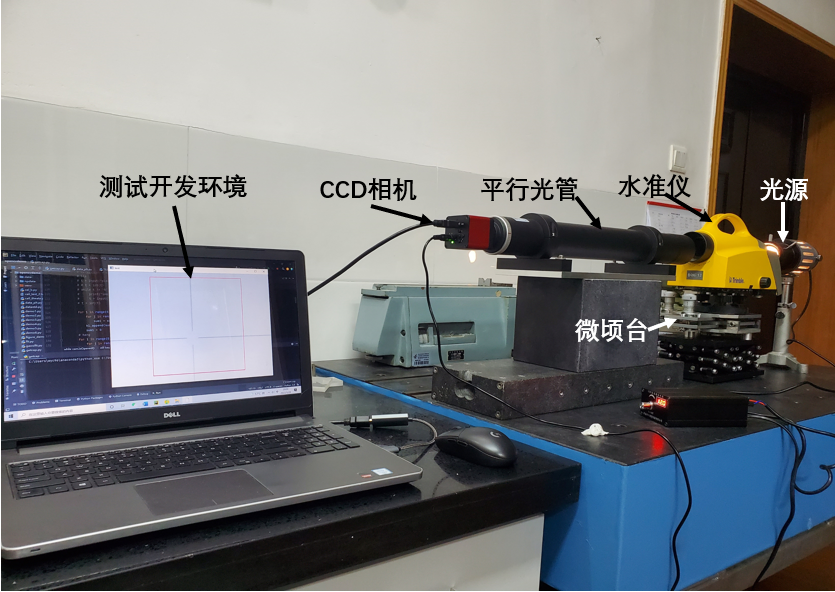
所以，用灰度重心法计算选定区域内每一行，每一列的中心位置坐标，利用最小二乘法拟合出直线，计算两直线的交点即为图像十字丝的中心坐标。得到亚像素级的中心位置坐标，利用两次中心位置坐标值，使用转换关系模型即可得出水准仪竖直角的改变量，从而衡量水准仪的补偿误差大小。

**2 实现**：

摄像头信息

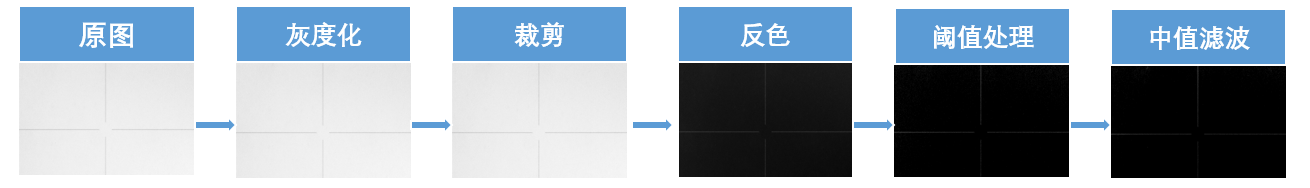
摄像头分辨率设置为19201080。

本算法采用python语言实现，开发环境为Pycharm，并使用OpenCV开源库构建部分基础功能函数。

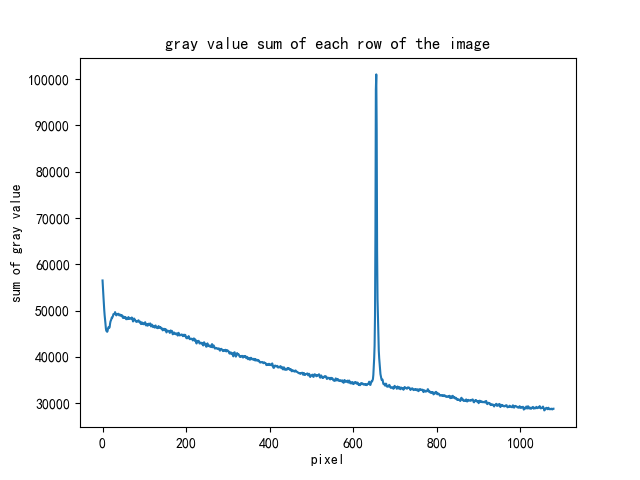
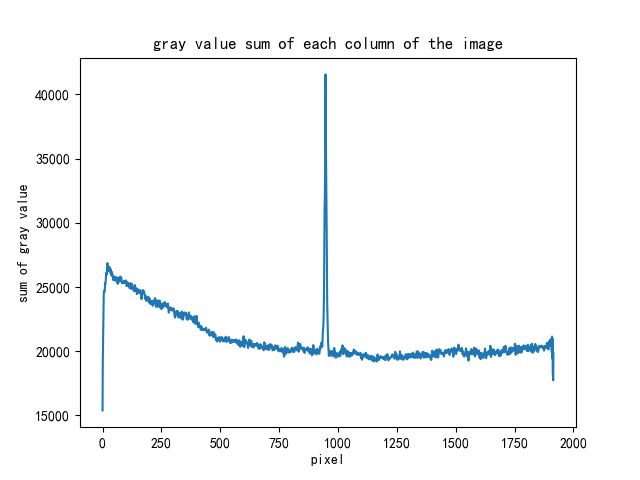


2.1 十字丝图像预处理

图像预处理步骤：

将图像转化为灰度图后，对图像进行取反操作。之后根据设定好的阈值，将灰度值小于的像素置为0。对图像进行去噪，采用一个的中值滤波器对图像进行滤波操作。

对图像分别按行，按列进行求和，求和的结果如图所示。取峰值点作为像素级的十字丝中心位置坐标。



根据提取目标区域，如图所示，以为中心，分别向左右（上下）拓展，并确定高度（长度）为的一片区域。

2.2 基于Canny算子和Otsu阈值分割算法的目标区域提取

2.2.1 Canny算子

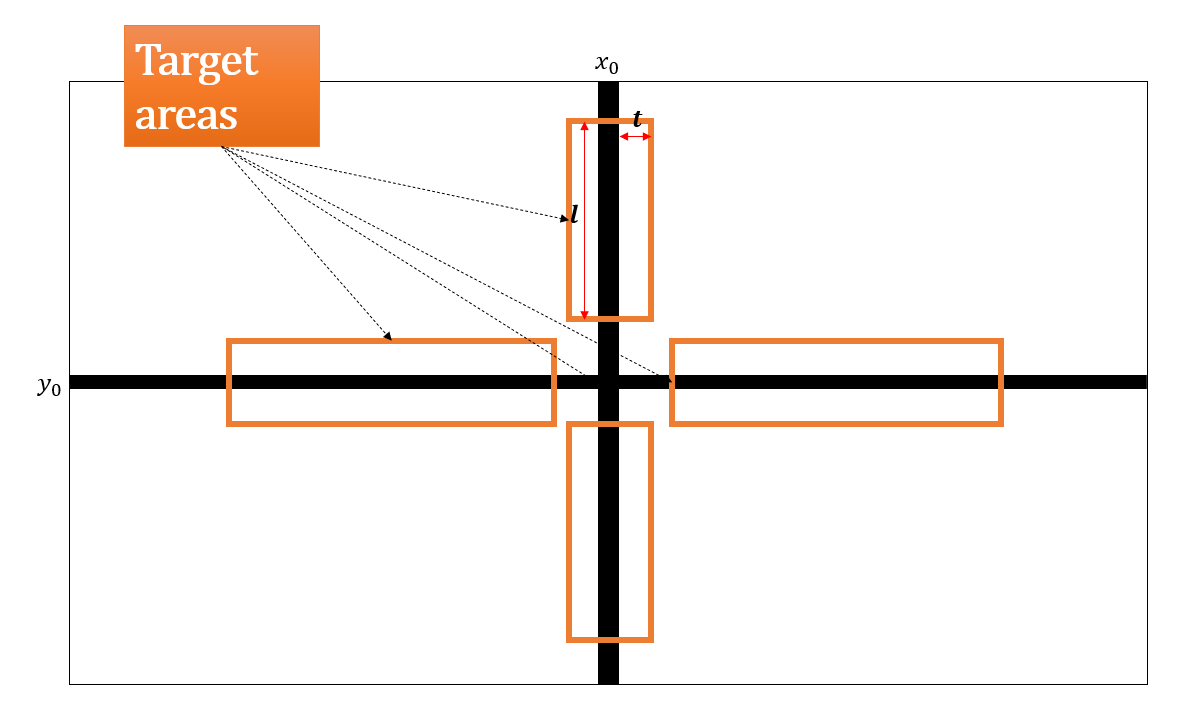
Canny算子是一种常见的边缘检测算法，是1986年John Canny提出。Canny算子是一阶算子，其方法的实质是利用一个准高斯函数做平滑运算，然后用带方向的一阶微分算子来定位导数最大值，它可用高斯函数的梯度来近似，在理论上很接近四个指数函数线性组合形成的边缘算子。

2.2.2 Otsu阈值分割算法

即大津法，其原理是把图像的像素点根据阈值分成两类。计算第一类像素的类内方差和第二类像素的类内方差，并计算这两个像素类的类间方差。根据类内方差最小同时类间方差最大这一准则，即可达到最好的分类效果，此时的阈值即为分割阈值。Otsu算法一般适用于灰度级分布差异较为明显的图像。

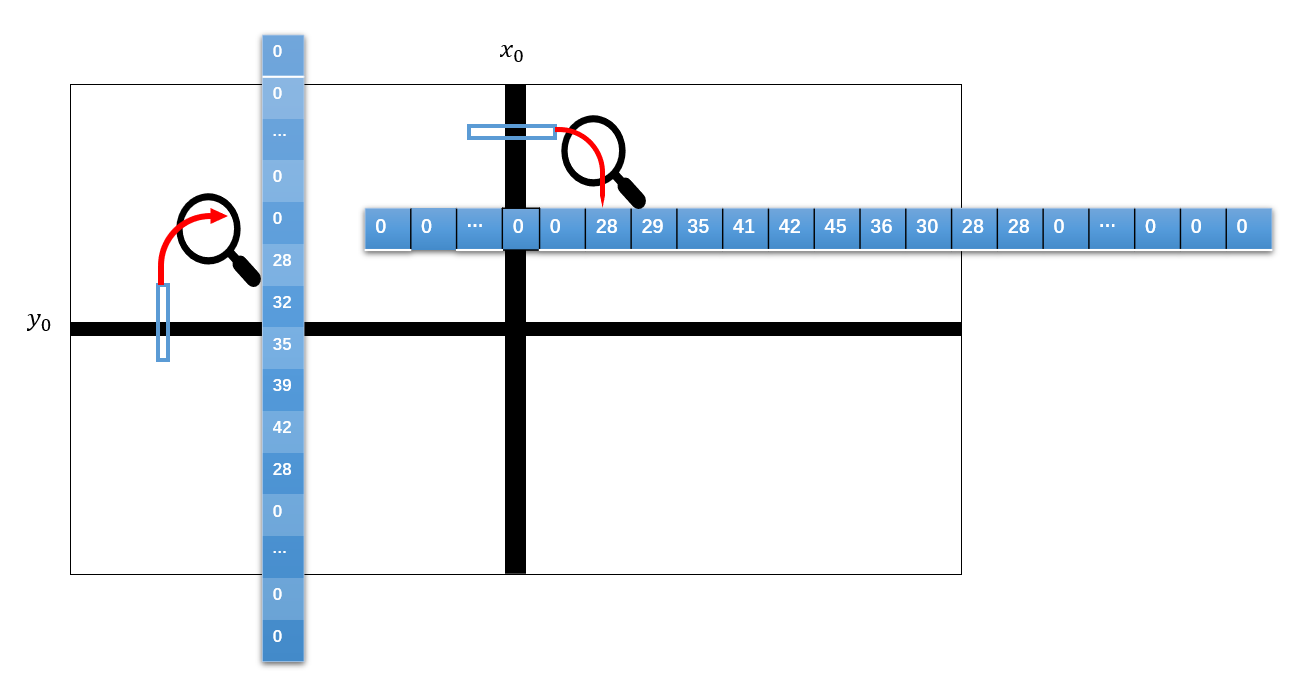
十字丝图像组成较为简单，一般只包含十字丝图像和背景两部分，但往往十字丝图像所占的像素面积较小，直接对整个图像使用Otsu算法的话，效果不太理想，因此，需要对图像进行处理之后才能进行分割。

对经过预处理后的图像进行Canny算子边缘检测，分别确定图像中横丝和竖丝边缘的位置，即对进行Canny算子边缘检测后的二值图像分别按行，按列进行灰度值的累加，取出其方向上最大的两个值作为边缘坐标值。根据边缘坐标值的插值计算边缘之间的距离，即边缘的宽度，分别向前，向后（对横丝为向上，向下）拓宽。这样可以保证该区域内目标区域像素所占的面积和背景像素所占的面积基本一致。在该区域内使用算法计算最佳分割阈值，将图像分成背景和目标区域。

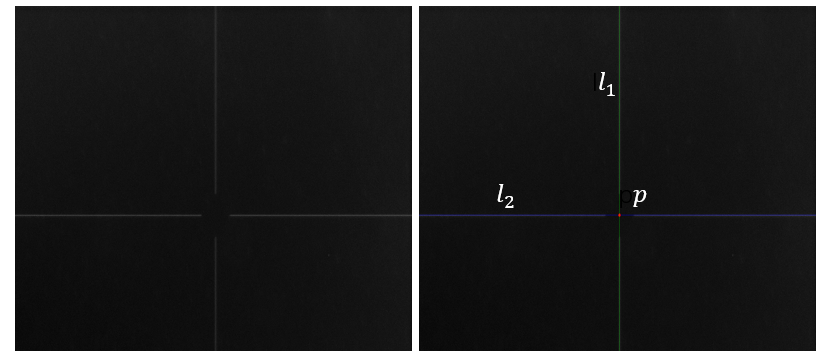


2.2 拟合

在目标区域内，利用灰度重心法进行亚像素定位并对定位的结果进行最小二乘拟合。



拟合得到的直线和直线以及交点，如下图右图所示



2.3 CCD光管的标定

利用全站仪作为角度标准，对CCD光管进行标定。测量得到组校准点，每组数据为十字丝中心位置的亚像素坐标以及对应的水平角以及天顶距，利用公式1对参数进行标定。

相机轴旋转角标定方法：

有个校准点。有个点的图像中十字丝坐标。有转换关系：

其中为测得点形成直线的斜率，为直线截距。写成矩阵形式：

即，

拟合值：

使残差平方和最小，

得：

**3实验数据分析**

3.1 对十字丝图像中心位置的模拟定位。

使用制作好的已知十字丝图像中心位置坐标的图像对上述亚像素定位算法就行仿真实验。共有10幅已知十字丝中心位置坐标的图像，每幅图像的尺寸大小为800×600。仿真结果如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 理论坐标 | (400.00,300.00) | (399.90,299.90) | (399.80,299.80) | (399.70,299.70) | (399.60,299.60) |
| 算法定位 | (400.00,300.00) | (399.89,299.89) | (399.80,299.80) | (399.70,299.70) | (399.60,299.60) |
| 误差 | <(0.005,0.005) | ≈(0.006,0.006) | <(0.005,0.005) |  |  |

3.2 测量系统测试

实打实打算

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
| 1 | 118.374 | 118.36 | 118.544 | 118.976 | 118.749 | 118.815 | 119.965 | 119.298 | 119.245 |
| 2 | 118.328 | 118.507 | 118.615 | 118.572 | 118.539 | 118.604 | 118.972 | 119.443 | 118.791 |
| 3 | 118.415 | 118.498 | 118.294 | 118.637 | 119.488 | 119.557 | 119.434 | 118.975 | 119.311 |
| 4 | 118.363 | 118.044 | 118.439 | 118.769 | 118.846 | 118.835 | 119.027 | 119.548 | 120.019 |
| 5 | 118.489 | 118.166 | 119.483 | 118.657 | 118.885 | 119.26 | 119.043 | 119.318 | 119.042 |
| 6 | 118.475 | 118.632 | 118.523 | 118.536 | 118.799 | 119.149 | 119.031 | 119.724 | 119.473 |
| 7 | 118.148 | 119.756 | 118.436 | 119.181 | 119.201 | 118.876 | 119.032 | 118.718 | 119.459 |
| 8 | 118.625 | 118.776 | 118.811 | 118.47 | 118.728 | 118.852 | 119.05 | 119.062 | 119.267 |
| 9 | 118.241 | 118.591 | 118.407 | 118.339 | 118.654 | 119.196 | 119.367 | 119.464 | 119.923 |
| 10 | 118.409 | 118.461 | 118.854 | 119.111 | 118.671 | 118.799 | 118.856 | 119.399 | 119.743 |
| 均值 | 118.39 | 118.58 | 118.64 | 118.72 | 118.86 | 118.99 | 119.18 | 119.29 | 119.43 |
| 标准差 | 0.133 | 0.466 | 0.344 | 0.28 | 0.284 | 0.286 | 0.328 | 0.298 | 0.384 |

对比

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
| 1 | 118.410 | 118.842 | 118.206 | 118.091 | 118.584 | 117.928 | 117.718 | 117.713 | 117.544 |
| 2 | 119.032 | 118.32 | 118.443 | 117.972 | 118.234 | 118.261 | 117.969 | 117.804 | 118.08 |
| 3 | 118.581 | 118.08 | 118.148 | 118.042 | 118.072 | 118.353 | 117.772 | 118.199 | 117.846 |
| 4 | 118.43 | 118.261 | 118.506 | 118.328 | 118.192 | 118.184 | 117.682 | 117.866 | 117.66 |
| 5 | 118.414 | 117.934 | 117.808 | 118.5 | 118.88 | 118.212 | 117.923 | 117.885 | 117.741 |
| 6 | 118.984 | 118.265 | 118.024 | 118.131 | 118.084 | 118.314 | 118.303 | 117.964 | 118.141 |
| 7 | 118.711 | 118.839 | 118.177 | 118.091 | 117.889 | 117.923 | 118.351 | 118.032 | 117.459 |
| 8 | 118.581 | 118.056 | 118.295 | 118.385 | 117.761 | 117.814 | 118.34 | 118.17 | 117.403 |
| 9 | 118.752 | 118.29 | 118.29 | 118.281 | 117.602 | 117.813 | 117.859 | 117.654 | 117.851 |
| 10 | 118.154 | 118.74 | 118.521 | 118.265 | 118.29 | 118.305 | 117.898 | 117.949 | 117.951 |
| 均值 | 118.60 | 118.36 | 118.24 | 118.21 | 118.16 | 118.11 | 117.98 | 117.92 | 117.77 |
| 标准差 | 0.273 | 0.331 | 0.222 | 0.169 | 0.377 | 0.216 | 0.258 | 0.178 | 0.253 |

对比

|  |  |  |
| --- | --- | --- |
| 后倾 | 本文算法 | 人眼 |
| 2 | 0.19 | 0.1 |
| 4 | 0.25 | 0.1 |
| 6 | 0.33 | 0.1 |
| 8 | 0.47 | 0.5 |
| 10 | 0.60 | 0.2 |
| 12 | 0.79 | 0.8 |
| 14 | 0.90 | 1.1 |
| 16 | 1.04 | 0.9 |

|  |  |  |
| --- | --- | --- |
| 前倾 | 本文算法 | 人眼 |
| 2 | 0.24 | 0.0 |
| 4 | 0.36 | 0.1 |
| 6 | 0.39 | 0.6 |
| 8 | 0.44 | 0.1 |
| 10 | 0.49 | 0.1 |
| 12 | 0.62 | 1.1 |
| 14 | 0.68 | 1.0 |
| 16 | 0.83 | 0.7 |

补偿误差：

|  |  |  |
| --- | --- | --- |
| 后倾 | 本文算法 | 人眼 |
| 2 | 0.080 | 0.050 |
| 4 | 0.063 | 0.025 |
| 6 | 0.055 | 0.017 |
| 8 | 0.059 | 0.063 |
| 10 | 0.060 | 0.020 |
| 12 | 0.066 | 0.067 |
| 14 | 0.064 | 0.079 |
| 16 | 0.065 | 0.056 |
| 最大值 | 0.080 | 0.079 |
| 均值 | 0.0640 | 0.0471 |
| 标准差 | 0.00693 | 0.0220 |

|  |  |  |
| --- | --- | --- |
| 前倾 | 本文算法 | 人眼 |
| 2 | 0.12 | 0.0 |
| 4 | 0.090 | 0.025 |
| 6 | 0.065 | 0.10 |
| 8 | 0.055 | 0.013 |
| 10 | 0.049 | 0.010 |
| 12 | 0.052 | 0.092 |
| 14 | 0.046 | 0.071 |
| 16 | 0.052 | 0.044 |
| 最大值 | 0.12 | 0.10 |
| 均值 | 0.0661 | 0.0444 |
| 标准差 | 0.0242 | 0.0364 |

**4结论**

参考文献