本科生毕业论文（设计）

题　　目　几种立体匹配算法的实现

学　　院　计算机学院

专　　业　计算机科学与技术

学生姓名　曾志刚

学　　号　1043041258　年级　2010

指导教师　王明辉

教务处制表

二０一四年五月二十一日

几种立体匹配算法的实现

计算机科学与技术

学生 曾志刚 指导老师 王明辉

[摘要] 同人类获取环境的三维信息类似，计算机双目立体视觉通过双目摄像机获取目标的三维信息。与传统的测量技术相比，立体视觉具有准确度高、非接触等优点，基于双目立体视觉的三维重建可用于测量、建模等领域。论文主要研究了双目立体视觉三维重建的若干算法，即：摄像机标定、畸变矫正和立体矫正、立体匹配以及三维重建。主要研究工作有:1)实现了多种基于灰度图像的角点检测算法(Moravec、Harris、Nobel、Shi-Tomasi)以及二次曲面拟合和向量方法的亚像素角点提取算法 2）实现了半自动棋盘标定程序，完成摄像机单目标定和双目标定 3）实现了多种立体匹配算法(FW、AW、FBS、DP、DP+AW、DP+FBS、SGM、SGM+FW、SGM+AW)，并提出一种改进的迭代的 SGM 算法，该算法在平滑度上优于 SGM 算法 4）实现了基于 OpenGL 的三维点云重建。

[主题词] 摄像机标定；harris；立体匹配；DP；SGM；迭代；

Implementations of several Stereo Matching algorithms

Computer Science and Technology

Student： ZENG Zhi-gang **Adviser:** WANG Ming-hui

[Abstract] Similar to human beings obtaining 3D information from environment, binocular camera can be used for computer binocular stereo vision to get the target's 3D information.Compared with the traditional measurement techniques,stereo vision has the advantage of high accuracy,non-contact and so on.3D reconstruction based on binocular stereo vision can be used to measure,model and other fileds.Some core algorithms of 3D reconstruction of binocular stereo vision have been studied,namely:camera calibration,distortion correction and stereo correction, stereo matching and 3D reconstruction.The main research works are:1）Implementations of many corner detecting algorithms based on gray image(Moravec、Harris、Nobel、Shi-Tomas)and sub-pixel corner extraction algorithm using quadratic surface fitting method and vector method 2)Implementations of a semi-automatic chessboard calibration program,completion of single camera calibration and binocular camera calibration 3)Implementations of a variety of stereo matching algorithms(FW、AW、FBS、DP、DP+AW、DP+FBS、SGM、SGM+FW、SGM+AW),and an improved iterative SGM algorithm was proposed,which is more smooth than SGM algorithm 4)Implementation of 3D point cloud reconstruction based on OpenGL.

[Key Words] Camera Calibration；Harris；Stereo Matching；DP；SGM；Iteration；

目 录

1．综述

1.1 引言

视觉是人类感知周围环境的一个非常重要的途径，同其他感官相比，人类通过视觉系统获取的信息占所有信息的 75%[1]。从狭义的角度来说，对场景做出有意义的描述和解释是视觉对观察者而言的最终目的；从更为广义的角度上讲，则是在上述解释和描述的基础上依据观察者的意愿进一步做出相应的行为规划或决策[2]。随着科学技术的不断发展，尤其是计算机技术的提高，诞生了计算机视觉这一新兴学科,它致力于代替人脑去理解世界[3]。

1.2 研究意义

目前，立体视觉在诸多领域如：机器人导航，医学成像，工业检测甚至军事领域都得到了相当广泛的应用；在农业领域，如农产品的检查、分离，虚拟植物的三维重建等方面也展示了良好的应用前景。

美国NASA勇气号与机遇号火星探测器的成功开启了双目视觉在机器人导航领域的先河，由于火星离地球非常遥远，假如通过遥控的方式控制火星车，从地面发射电磁波需要近 20 分钟的时间才能抵达火星，不能通过遥控的方式来控制火星车，就需要火星车具备良好的自主能力。Atthena 火星车采用 4 对立体摄像机，通过立体相机获取图像并经过立体匹配和路经规划找到合适的导航路径。在我国的嫦娥工程二期工程中，样机 MR-3 上也采用了双目视觉导航技术[5]。

由于立体视觉具有测量者于被测量物不需要接触的特点，立体视觉被广泛用于工业检测。比如，现代飞机大多使用高涵道涡轮风扇发动机，由于航空发动机工作在高温、高压和告诉旋转的工作环境下，高压压气机、燃烧室和高压涡轮为故障多发部位，但由于这些零件不易拆卸且检测可达性较差[6]。文献[6]将立体视觉和运用于内窥技术，可以定性和定量地了解发动机内部损伤和内部工作情况，为损伤的预测和防治提供了重要依据。

在医学领域，立体视觉相对于传统的测量手段具有明显的优势，如：传统牙颌模型测量,采用游标卡尺、万能角度尺等测量工具在石膏模型上直接测量,不仅难以对牙颌的复杂几何形态给出全面的描述,而且对数据的测量和处理也较费时费力。而采用立体视觉对牙颌模型图像进行三维重建，之后再进行测量与分析，更加精确和方便[7]。

1.3 国内外研究现状

立体视觉按照摄像机个数可分为双目立体视觉、多目立体视觉。双目立体视觉作为研究多目立体视觉最重要的基础，从二十世纪 70 年代中期 Marr 提出第一个计算机视觉领域的理论框架开始，立体视觉研究的理论框架日臻完善。在双目立体视觉系统中，摄像机标定、立体匹配、三维重建是最为重要的三个研究方向。

1.3.1 摄像机标定

在立体视觉的三维重建过程中，需要建立起物体实际的三维坐标和摄像机采集到该物体图像的对应点的映射关系，也就是摄像机成像的几何模型参数的获取。摄像机标定就是通过具体实验操作计算出该几何模型的各个参数[8]。摄像机标定作为三维重建的首要步骤，其精度直接影响着三维重建的精度和效果，因此提高摄像机标定的精度是大部分标定算法的重点。摄像机标定一般可分为，传统的摄像机标定方法、主动视觉标定方法、摄像机自标定法[9]。

传统的摄像机标定方法利用已知的景物结构信息(标定块、标定棋盘)，Abdel-Aziz 和 Karara 于 1971 年提出直接线性标定，即将像点和物点成像几何关系在齐次坐标系下以透视矩阵的形式表示，通过求解线性方程组而求出透视矩阵[10]。Tsai 于 1986 年通过建立摄像机模型，利用径向一致约束，分两步求解出摄像机的内参和外参，能得到较高的精度[11]。张正友于 1999 年[12]提出一种灵活的标定方法:只需要打印一张 2D 平面棋盘，从不同的方向(至少两个，方向不需要已知)拍摄图片，在计算过程中，先计算出摄像机内参，再计算出外参，之后使用最大似然进行优化，张正友标定法简单、较精确、低成本。

摄像机自标定则不需要已知景物的结构信息，而是利用摄像机在运动过程中周围环境的图像与图像之间的对应关系进行标定。目前的自标定方法主要有：利用绝对二次曲线和极线变换性质解 Kruppa 方程、分层逐步标定、基于二次平面以及其他改进的摄像机自标定技术[13]。主动视觉系统是指摄像机被精确的安装在控制平台上，通过主动控制摄像机运动获取多幅图片，通过图像与已知的运动参数进行标定[13]。

1.3.2 立体匹配

立体匹配一直是双目立体视觉中最为重要的问题，立体匹配的目的即找出左右图片内信息对应的关系，从而得到视差图。立体匹配算法可分为基于特征的匹配、基于区域的匹配、基于相位的匹配。

基于特征的立体匹配算法首先在左右图片中提取图像特征，然后再进行匹配。可能使用的图像特征通常有：特征点（角点，边缘点，零交叉点）、特征线（直线、边缘线、轮廓）、闭合区域、统计特征（如重心、矩不变量）等[14]。基于特征的立体匹配算法具有计算量小、抗干扰能力强、精度较高等优点。但是基于特征的立体匹配算法只能获取稀疏的视差图(特征点的数目较少)，通常在匹配之后，还需要进行插值处理才能获得更加稠密的视差图。

近年来，基于区域的立体匹配算法发展十分迅速，基于区域的立体匹配方法算法能获取稠密的视差图，适合场景的重建。基于区域的立体匹配可分为如下几大过程[15]：

1、匹配代价计算，常用的匹配代价函数有：平方差(SD)，绝对差(AD)等。为了减少畸变、光线等误差对上述这些代价函数的影响，可采用基于梯度的度量方法[16]，基于 rank 和 census 变换的度量[17]，这些无参数的度量方式有效地提高了匹配的强健度。通过计算匹

配代价，从而得到视差空间图(DSI)的初始值。

2、匹配代价聚合，由于单个像素的匹配代价容易受到图片噪声的影响，而且可鉴别性差，通过聚合像素点周围的匹配代价能有效提高可鉴别性。一种简单的聚合方式即把像素点周围固定窗口区域（FW）的代价值相加作为该像素的聚合代价，这种方法简单快速、计算量小，由于窗口的尺寸固定，可以采用滑动窗口技术加速求和过程，使得这种方法被广泛的用于一些实时系统中。

通常地，较小的窗口使得视差图的边缘信息较完整、但同时在纹理较少的区域误匹配点多；采用较大的窗口得到的视差图过于平滑，在视差不连续的区域过于模糊，因此损失了大量的边缘信息。为了解决这一问题，采用可移动窗口，即参考像素的位置并不固定于窗口的中心位置，而是多个位置选取结果最好的作为聚合代价[15]。此外，还可采用多窗口技术[18]：最小子窗口的大小均相同，选取最优的部分子窗口代价只和作为聚合代价，从而使得在物体边缘处不会因为窗口过大而变得过于模糊。

以上两种算法均属于自适应窗口算法，即通过改变窗口的大小或者形状进行能量聚合。M.Gerrits[19]等人在 2006 年提出基于图像分割的立体匹配方法，其核心思想即同一分割区域内的像素代价应该聚合在一起，这个假设比较符合实际情况同一分割区域属于同一物体，该算法采用如下方法实现：对于支持窗口内，与中心像素为同一分割区域的像素代价权值为 1，其余的为 λ (λ <<1)。

K.Yoon[20]等在 2006 年提出自适应权重的立体匹配算法，与图像分割算法相比，自适应权重并非将支持窗口内的像素分为和中心像素同一区域、不同区域，即权值为 1 和 λ，而是采用双边滤波的算法，对于每一个像素，权值由同中心像素的空间距离以及同中心像素的色彩空间距离决定。

3、视差获取及优化，对于局部立体匹配算法而言，采用“赢家通吃”(WTA)算法，即选取具有最小匹配代价的视差作为视差(唯一性约束,即对于参考图像的每一个像素，目标图像最多只有一个像素点与之对应)。相反，对于全局匹配算法而言，算法的重点不是匹配代价聚合而是视差获取及优化。大部分的全局算法采用构造能量函数[15]：

$$E(\mathrm{d}) = E_{\mathrm{data}}(\mathrm{d}) + \lambda E_{smooth}(\mathrm{d}) \tag{1-1}$$

将视差的求取转化为一个能量最小化问题的求解，其中 $E_{\mathrm{data}}(\mathrm{d})$ 为数据项，为两张图片像素一致性程度。

$$E_{data}(\mathrm{d}) = \sum_{x,y} C(\mathrm{x},\mathrm{y},\mathrm{d}(\mathrm{x},\mathrm{y})) \tag{1-2}$$

其中 C 为匹配代价(初始值或者聚合值)。平滑项 $E_{smooth}(\mathrm{d})$ 表示像素视差与周围像素视差的平滑度。有时，为了便于计算，平滑项简化为只考虑临近的像素视差的差值，如：

$$E_{smooth}(\mathrm{d}) = \sum_{x,y} \rho(\mathrm{d}(\mathrm{x},\mathrm{y}) - d(\mathrm{x}+1,\mathrm{y})) + \rho(\mathrm{d}(\mathrm{x},\mathrm{y}) - \mathrm{d}(\mathrm{x},\mathrm{y}+1)) \tag{1-3}$$

当全局能量函数构造完成之后，采用能量最小化手段求取极小值。传统的能量最

小化方法有模拟退火算法、平均退火法、梯度下降法。传统的能量最小化方法由于收敛速度慢而较少在实际中应用。2001 年 Yuri Boykov[21]提出通过图割算法进行 α 扩展和 α - β 交换移动以进行能量最小化；此外，还有基于置信传播和动态规划等算法的全局立体匹配算法。

4、视差精化，在实际运用中，像素精度的视差通常无法满足应用需求，为了得到亚像素精度的视差，通常采用梯度下降法以及曲线拟合代价。除此之外，很多算法在这个步骤进行左右一致性校验，以除去误匹配点或者遮挡区域。另外也可采用滤波器(如中值滤波器)除去视差图中的噪声点。经过上面的处理过程，有些像素点可能会没有视差值，可以采用平面拟合的方式填补这些像素的视差值[22]。

1.3.3 三维重建

经过立体匹配得到视差图，结合标定的投影方程可以计算出各个像素点(具有有效视差值的像素点)的三维坐标。然而，这样得到的只是散乱的点云，基于点云的三维重建就是通过这些点云，首先进行点云数据的预处理(点云精简、点云重采样、点云过滤)，然后进行空间的点云网格化处理，生成近似物体原始表面的网格模型，最后通过纹理映射，将物体的图片纹理映射到三维模型上，得到重建的三维实体模型[23]。由于研究通过点云进行三维重建不是本文的重点所在，本文仅采用带有简单纹理的点云作为三维重建的参考效果。

2 摄像机标定

2.1 摄像机成像模型

摄像机成像模型用于模拟光学成像几何关系，在精度要求不高或者镜头畸变较小的情况下可以仅采用线性模型；当要求高精度结果时，就需要考虑非线性因素(镜头畸变)而采用非线性模型[24]。

2.1.1 描述坐标系

为了清楚地表达物体在 3 维空间中的实际位置与图像中像素的对应关系，定义以下 4 个参考坐标系，如图 2.1 所示：

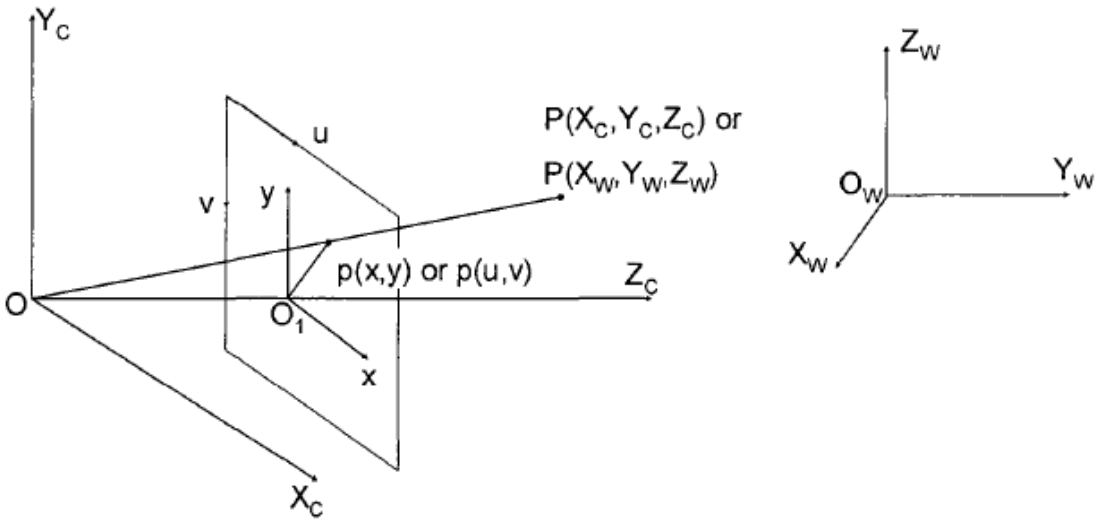


图 2.1 4 个参考坐标系关系图

(1)世界坐标系 $O_wX_wY_wZ_w$

在实际的三维空间中一个参考坐标系，可以根据需要任意选取。P（Xw,Yw,Zw）为该坐标系内一点。

(2)摄像机坐标系 $O_cX_cY_cZ_c$

摄像机坐标系 $O_cX_cY_cZ_c$ 的坐标原点定义为摄像机透镜的光心处，$O_cX_cY_c$ 平面平行于成像平面，Z 轴与光轴重合。

(3)成像平面坐标系 OXY

成像平面坐标系 OXY 的坐标原点定义为光轴与成像平面的交点。世界坐标系、摄像机坐标系、成像平面坐标系的单位均为实际的物理单位(如毫米 mm)。

(4)图像坐标系 UV

图像坐标系 UV 按照习惯定义坐标原点为图片的左上角，(u,v)代表第 u 行、第 v 列像素点，单位为像素。

2.1.2 线性模型

在不考虑摄像机畸变等非线性因素的情况下，摄像机成像模型可以采用我们熟悉的小孔成像模型来建模，如图 2.1 所示。下面逐个分析 4 个坐标系之间的变换关系：

(1)世界坐标系 $O_wX_wY_wZ_w$ 和摄像机坐标系 $O_cX_cY_cZ_c$ 的关系：由于摄像机坐标系和世界坐标系之间存在旋转变换和平移变换，某一点在摄像机坐标系内的坐标(x_c,y_c,z_c)和在世界坐标系内坐标(x_w,y_w,z_w)有如下关系：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + T \quad \text{其中} \quad R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2-1}$$

其中， **R** 和 **T** 和分别为旋转矩阵和平移矩阵，若采用齐次坐标的写法，有：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0_{3\times3} & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{2-2}$$

(2)摄像机坐标系 $O_cX_cY_cZ_c$ 和成像平面坐标系 OXY 坐标系关系，如图 2.2 所示，

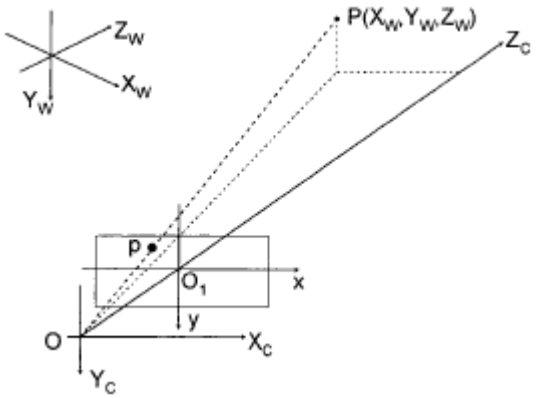


图 2.2 摄像机坐标系和成像平面坐标系

该变换为中心映射或透视投影，依据相似三角形的比例关系，某点在图像平面上的坐标(x,y)和在摄像机坐标系内坐标(x_c,y_c,z_c)有：

$$\begin{cases} x = f\dfrac{x_c}{z_c} \\ y = f\dfrac{y_c}{z_c} \end{cases} \tag{2-3}$$

其中，f 为相机透镜的焦距，改写为齐次坐标形式即：

$$z_c\begin{bmatrix}x\\y\\1\end{bmatrix}=\begin{bmatrix}f&0&0&0\\0&f&0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}x_c\\y_c\\z_c\\1\end{bmatrix} \tag{2-4}$$

(3)图像坐标系 UV 和成像平面坐标系 OXY 之间的关系如图 2.3 所示:

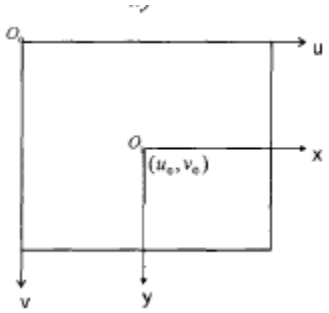


图 2.3 图像坐标系与成像平面坐标系

某一点在图像坐标系内的坐标(u,v)和在成像平面坐标系内的坐标(x,y)有：

$$\begin{cases}u=\dfrac{x}{dx}+u_0\\[2ex] v=\dfrac{y}{dy}+v_0\end{cases} \tag{2-5}$$

其中 dx，dy 为每个像素的物理宽度和长度，$(u_0,v0)$为成像坐标平面的原点坐标，改写为齐次坐标的形式即：

$$\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}\dfrac{1}{dx}&0&u_0\\[2ex]0&\dfrac{1}{dy}&v_0\\[2ex]0&0&1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix} \tag{2-6}$$

将(2-2)、(2-4)、(2-6)联立有：

$$\begin{aligned}z_c\begin{bmatrix}u\\v\\1\end{bmatrix}&=\begin{bmatrix}\dfrac{1}{dx}&0&u_0\\[2ex]0&\dfrac{1}{dy}&v_0\\[2ex]0&0&1\end{bmatrix}\begin{bmatrix}f&0&0&0\\0&f&0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}R&T\\0_{3\times3}&1\end{bmatrix}\begin{bmatrix}x_w\\y_w\\z_w\\1\end{bmatrix}\\&=\begin{bmatrix}f_x&0&u_0&0\\0&f_y&v_0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}R&T\\0_{3\times3}&1\end{bmatrix}\begin{bmatrix}x_w\\y_w\\z_w\\1\end{bmatrix}=M_1M_2X_w=MX_w\end{aligned} \tag{2-7}$$

其中，M_1 只与摄像机本身属性有关，故称 M_1 为内参，为摄像机的固有属性，一般不会因

(2-7)

为摄像机的摆放位置改变或者世界坐标系改变而改变(假设摄像机为固定焦距摄像机)；相反矩阵 M_2 只由摄像机与世界坐标系的相对位置(旋转或者平移)相关，故称 M_2 为外参矩阵。内参和外参矩阵的乘积 M 即为投影矩阵。

2.1.3 非线性模型

在需要精确的标定结果或者摄像机畸变较大(广角镜头)等原因时，需要考虑非线性因素。由于摄像头的类型不同、制造工艺不同，存在多种畸变，常用的畸变模型有：径向畸变、离心畸变和薄凌镜畸变,我们采用以下方式描述畸变：

$$\begin{cases} x_d = x_u + \delta_x(x_u, y_u) \\ y_d = y_u + \delta_y(x_u, y_u) \end{cases} \tag{2-8}$$

其中(x_u,y_u)为成像坐标系下理想的坐标，(x_d,y_d)为成像坐标系下实际(畸变)的点坐标。

(1)径向畸变的特点是畸变关于摄像机透镜光轴对称，桶形畸变和枕形畸变是两种常见的径向畸变，如图 2.4 所示：

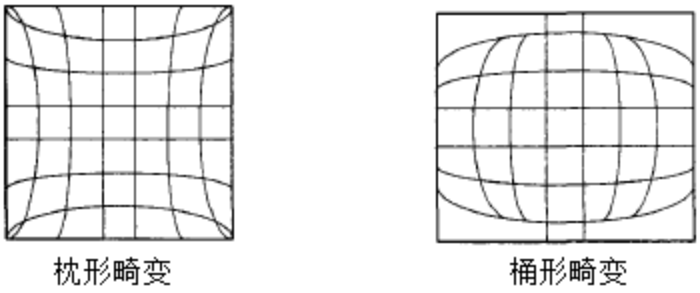


图 2.4 枕形畸变和桶形畸变[25]

使用高阶多项式数学模型描述为：

$$\begin{cases} \delta_x = x(k_1 r^2 + k_2 r^4 + k_3 r^6 + \cdots) \\ \delta_y = y(k_1 r^2 + k_2 r^4 + k_3 r^6 + \cdots) \end{cases} \tag{2-9}$$

其中 $r^2 = z^2 + y^2$,k_1,k_2,k_3 为径向畸变参数。

(2)离心畸变主要是由于光学系统光心与几何中心不重合而造成的畸变，使得各个透镜的光心不能严格共线，离心畸变不仅包含径向畸变还包含切向畸变，切向畸变采用高阶多项式数学模型描述为：

$$\begin{cases} \delta_x = 2p_1 xy + p_2(r^2 + 2x^2) + \cdots \\ \delta_y = 2p_1(r^2 + 2y^2) + 2p_2 xy + \cdots \end{cases} \tag{2-10}$$

其中 p_1，p_2 为切向畸变参数。

(3)薄凌镜畸变一般由于在设计、生产以及摄像机的组装过程中产生的误差而形成的。例如 CCD 阵列或镜头的微小倾斜等，薄凌镜畸变效果相当于在光学系统中附加了了一个薄凌镜，采用高阶多项式数学模型描述为：

$$\begin{cases}\delta_x = s_1 \mathrm{r}^2 + \cdots \\ \delta_y = s_2 \mathrm{r}^2 + \cdots\end{cases} \tag{2-11}$$

其中 s_1，s_2 为畸变参数。

Brown 于 1966 年提出一个包含上述三种畸变的综合模型，称为铅锤模型(Plumb Bob Model)，采用高阶多项式的数学模型描述为：

$$\begin{cases}\bar{x} = (1+\mathrm{k}_1 r^2 + \mathrm{k}_2 r^4 + \mathrm{k}_3 r^6)\,\mathrm{x} + 2\mathrm{k}_3 \mathrm{xy} + \mathrm{k}_4(\mathrm{r}^2 + 2\mathrm{x}^2) \\ \bar{y} = (1+\mathrm{k}_1 r^2 + \mathrm{k}_2 r^4 + \mathrm{k}_3 r^6)\,\mathrm{y} + \mathrm{k}_3(\mathrm{r}^2 + 2y^2) + 2\mathrm{k}_4 \mathrm{xy}\end{cases} \tag{2-12}$$

上述式子中，点 (x, y) 为成像坐标系中理想的点坐标，$(\bar{\mathrm{x}}, \bar{\mathrm{y}})$ 为实际(畸变)的成像点坐标。k_1 k_2 k_3 k_4 k_5 为畸变参数。

在实际应用情况中，通常考虑高阶(四阶以上)的径向分量的畸变模型是没有必要的，因为单纯地增加畸变模型的阶数并不一定能带来高的精度，Tsai[26]指出，引入过多的参数往往不能提高精度，反而使得在求解参数的时候使得解不稳定。

2.2 直接线性标定方法

直接线性变换方法(DLT)由 Abdel-Aziz 和 Karara[10]于 1971 年首先提出。该方法只采用线性模型作为计算模型，不考虑摄像头畸变的因素。

2.2.1 投影矩阵的求解

由式 2.1，将投影矩阵 M 展开写成如下形式：

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{2-13}$$

其中 $(\mathrm{x_w}, \mathrm{y_w}, \mathrm{z_w})$ 为世界坐标系中某一点的坐标，(u, v) 为对应点在图像上的坐标，上式包含三个方程：

$$\begin{cases} z_c u = m_{11} x_w + m_{12} y_w + m_{13} z_w + m_{14} \\ z_c v = m_{21} x_w + m_{22} y_w + m_{23} z_w + m_{24} \\ z_c = m_{31} x_w + m_{32} y_w + m_{33} z_w + 1 \end{cases} \tag{2-14}$$

令 $\mathbf{m_{34}}$=1，消去 $\mathrm{Z_c}$ 之后有：

$$\begin{cases} m_{11} x_w + m_{12} y_w + m_{13} z_w + m_{14} - u m_{31} x_w - u m_{32} y_w - u m_{33} z_w = u \\ m_{21} x_w + m_{22} y_w + m_{23} z_w + m_{24} - v m_{31} x_w - v m_{32} y_w - v m_{33} z_w = v \end{cases} \tag{2-15}$$

已知一个空间点的坐标和它对应的图像坐标可以得到上述两个方程，已知 n 个这样的点时，便能得到 2n 个方程的方程组，记为：

$$Km = U \tag{2-16}$$

其中

$$K=\begin{bmatrix} x_{w1} & y_{w1} & z_{w1} & 1 & 0 & 0 & 0 & 0 & -u_1x_{w1} & -u_1y_{w1} & -u_1z_{w1} \\ 0 & 0 & 0 & 0 & x_{w1} & y_{w1} & z_{w1} & 1 & -v_1x_{w1} & -v_1y_{w1} & -v_1z_{w1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{wn} & y_{wn} & z_{wn} & 1 & 0 & 0 & 0 & 1 & -u_nx_{wn} & -u_ny_{wn} & -u_nz_{wn} \\ 0 & 0 & 0 & 0 & x_{wn} & y_{wn} & y_{wn} & 1 & -v_nx_{wn} & -v_ny_{wn} & -v_1z_{wn} \end{bmatrix} \tag{2-17}$$

$$m=[m_{11} \quad m_{12} \quad m_{13} \quad m_{14} \quad m_{21} \quad m_{22} \quad m_{23} \quad m_{24} \quad m_{31} \quad m_{32} \quad m_{33}]^{\mathrm{T}} \tag{2-18}$$

$$U=\begin{bmatrix} u_1 & v_1 & \cdots & \cdots & u_n & v_n \end{bmatrix}^{\mathrm{T}} \tag{2-19}$$

(x_{wi}, y_{wi}, z_{wi})代表第 i 个点的三维坐标，(u_i, v_i)代表该点对应的图像坐标。当 n>=6 时，通过最小二乘法解超定线性方程组得到列向量 m 从而求得投影 M：

$$m=(K^{\mathrm{T}}K)^{-1}K^{\mathrm{T}}U \tag{2-20}$$

2.2.1 摄像机内参和外参的求解

求出投影矩阵 M 后，便能求得摄像机的内参和外参。将式 2.1 改写为如下形式[27]：

$$m_{34}\begin{bmatrix} m_1^T & m_{14} \\ m_1^T & m_{24} \\ m_1^T & 1 \end{bmatrix}=\begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} r_1^T & t_x \\ r_2^T & t_y \\ r_3^T & t_z \\ 0 & 1 \end{bmatrix}=\begin{bmatrix} f_xr_1^T+u_0r_3^T & f_xt_x+u_0t_z \\ f_yr_2^T+v_0r_3^T & f_yt_y+v_0t_z \\ r_3^T & t_z \end{bmatrix} \tag{2-21}$$

其中 m_i^T(i=1,2,3)表示矩阵 M 的第 i 行前 3 个元素组成的行向量，r_i^T (i=1,2,3) 表示旋转矩阵 R 的第 i 行元素组成的行向量。

对比左右两式，$m_{34}m_3=r_3$，由于 R 为单位正交矩阵，所以$\|r_3\|=1$，故

$$m_{34}=\frac{1}{\|m_3\|} \tag{2-22}$$

然后可以解出摄像机的内参参数：

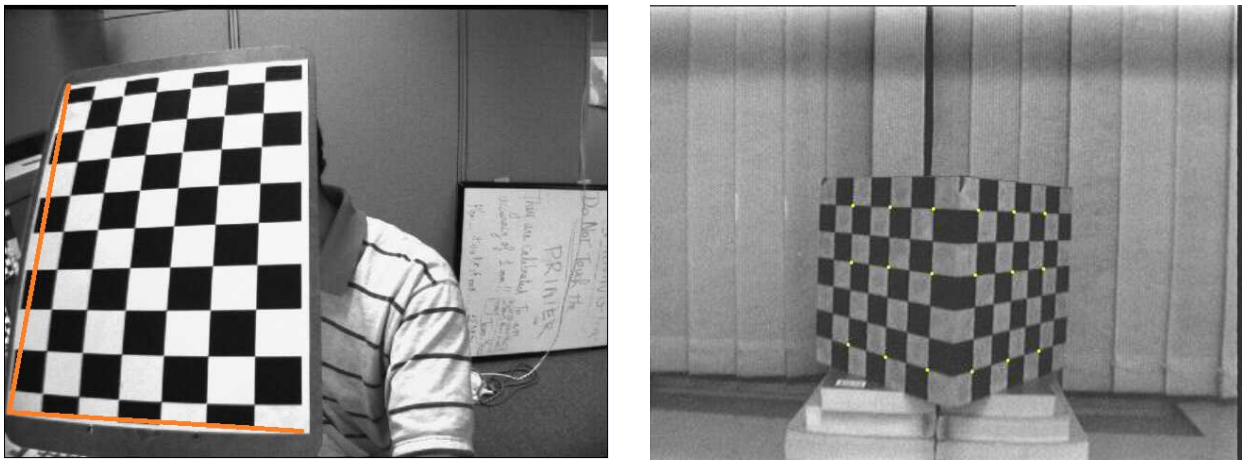
$$\begin{cases} r_3=m_{34}m_3 \\ u_0=(f_xr_1^T+u_0r_3^T)r_3=m_{34}^2m_1^Tm_3 \\ v_0=(f_yr_2^T+v_0r_3^T)r_3=m_{34}^2m_2^Tm_3 \\ f_x=m_{34}^2(m_1\times m_3) \\ f_y=m_{34}^2(m_2\times m_3) \end{cases} \tag{2-23}$$

接着求出摄像机的外参参数：

$$\begin{cases} r_1 = \dfrac{m_{34}(\mathrm{m}_1 - \mathrm{u}_0\mathrm{m}_3)}{f_x} \\ r_2 = \dfrac{m_{34}(\mathrm{m}_2 - v_0\,\mathrm{m}_3)}{f_y} \\ t_x = m_{34} \\ t_y = \dfrac{m_{34}(\mathrm{m}_{14} - \mathrm{u}_0)}{f_x} \\ t_z = \dfrac{m_{34}(\mathrm{m}_{24} - v_0)}{f_y} \end{cases} \tag{2-24}$$

2.3 张正友标定法

在上一节介绍的直接线性标定法中，标定采用线性模型，并没有考虑摄像机畸变，当镜头畸变较大时误差较大，并且在 2.2.1 中，求解投影矩阵需要 6 个(以上)三维空间点的坐标以及其对应的图像的点的坐标，还需要用到精密的标定块，标定过程十分繁琐。



(a)由于镜头畸变，棋盘边出现扭曲　　(b)标定块

图 2.5 镜头畸变和二维标定块

张正友[12]于 1999 年提出一种灵活的标定方法，该方法较传统的标定方法(DLT、Tsai 等)具有更加简单、低成本的优点。标定采用二维标定板(如打印棋盘)而不是精确的标定块。与摄像机自标定方法相比具有精度高、可靠性强的优点。

张正友标定法主要分为两大步骤：内参矩阵和外参矩阵的初始值求解、求取畸变参数以及参数精化。

2.3.1 摄像机内参矩阵和外参矩阵初始值求解

令图像上的二维点表示为 $m = [u \quad v]^T$，齐次坐标形式为 $\tilde{m} = [u \quad v \quad 1]^T$，世界坐标系中 3 维点表示为 $M = [X \quad Y \quad Z]^T$，齐次坐标形式为 $\tilde{M} = [X \quad Y \quad Z \quad 1]^T$，张正友标定方法采用的摄像机模型为：

$$s\tilde{m} = A\begin{bmatrix} R & t \end{bmatrix}\tilde{M} \tag{2-25}$$

其中，

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2-26}$$

即摄像机内参矩阵，***R*** 为旋转矩阵，***t*** 为平移矩阵，$\begin{bmatrix} R & t \end{bmatrix}$为摄像机外参矩阵，在下文的推导中，将 $(A^{-1})^T$ 和 $(A^T)^{-1}$ 简记为 A^{-T} 。

在计算机视觉中，平面的单应性被定义为一个平面到另外一个平面的投影映射。对于一块标定板(假定是平面)，标定板上的点和摄像机拍摄的图像上对应点之间的映射即为单应性变换。设标定板上某一点坐标为 (x_w, y_w, z_w),令标定板所在平面为 $z_w = 0$，则 $z_w = 0$,有：

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A\begin{bmatrix} r_1 & r2 & T \end{bmatrix}\begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = H\begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{2-27}$$

其中 $r_i(i=1,2)$ 为旋转矩阵 **R** 的第 i 列元素组成的列向量。**H** 矩阵为单应性矩阵，将 H 矩阵归一化:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \tag{2-28}$$

(2-28)

为将上式展开为方程组形式：

$$\begin{cases} z_c u = h_{11}x_w + h_{12}y_w + h_{13} \\ z_c v = h_{21}x_w + h_{22}y_w + h_{23} \\ z_c = h_{31}x_w + h_{32}y_w + 1 \end{cases} \tag{2-29}$$

消去 z_c 有:

$$\begin{cases} h_{11}x_w + h_{12}y_w + h_{13} - uh_{31}x_w - uh_{32}y_w = u \\ h_{21}x_w + h_{22}y_w + h_{23} - vh_{31}x_w - vh_{32}y_w = v \end{cases} \tag{2-30}$$

已知一个三维空间中的点的坐标 (x_w, y_w, z_w) 和其对应图像上点的坐标(u,v)可以得到一个包含 2 个方程的方程组，已知 n 个对应点可得到一个包含 2n 个方程的方程组，用矩阵形式表示为：

$$Kh = U \tag{2-31}$$

其中，

$$K = \begin{bmatrix} x_{w1} & y_{w1} & 1 & 0 & 0 & 0 & -u_1x_{w1} & -u_1y_{w1} \\ 0 & 0 & 0 & x_{w1} & y_{w1} & 1 & -v_1x_{w1} & -v_1y_{w1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{wn} & y_{wn} & 1 & 0 & 0 & 0 & -u_nx_{wn} & -u_ny_{wn} \\ 0 & 0 & 0 & x_{wn} & y_{wn} & 1 & -v_nx_{wn} & -v_ny_{wn} \end{bmatrix} \tag{2-32}$$

h 为 H 矩阵各个参数组成的列向量：$h=\begin{bmatrix}h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32}\end{bmatrix}^T$，$U=\begin{bmatrix}u_1 & v_1 & \cdots & \cdots & u_n & v_n\end{bmatrix}^T$，$(x_{wi},y_{wi},0)$ 为标定板上第 i 个点的坐标，对应图像上点的坐标为 (u_i,v_i)，当 2n>=8 时，可以通过最小二乘法解超定线性方程组得到列向量 **h** 从而得到 **H** 矩阵。令 $H=\begin{bmatrix}h_1 & h_2 & h_3\end{bmatrix}$,则

$$\begin{bmatrix}h_1 & h_2 & h_3\end{bmatrix}=\lambda A\begin{bmatrix}r_1 & r_2 & t\end{bmatrix} \tag{2-33}$$

由于旋转矩阵 **R** 为单位正交矩阵，故 $r_1^Tr_1=r_2^Tr_2=1$,且 $r_1^Tr_2=0$,联立(2-33)可得：

$$\begin{cases}h_1^TA^{-T}A^{-1}h_2=0\\ h_1^TA^{-T}A^{-1}h_1=h_2^TA^{-T}A^{-1}h_2\end{cases} \tag{2-34}$$

令

$$B=A^{-T}A^{-1}=\begin{bmatrix}B_{11} & B_{12} & B_{13}\\ B_{12} & B_{22} & B_{23}\\ B_{13} & B_{23} & B_{33}\end{bmatrix},b=\begin{bmatrix}B_{11} & B_{12} & B_{22} & B_{13} & B_{21} & B_{33}\end{bmatrix}^T \tag{2-35}$$

则：

$$h_i^TBh_j=v_{ij}^Tb \tag{2-36}$$

其中

$$v_{ij}=\begin{bmatrix}h_{i1}h_{j1} & h_{i1}h_{j2}+h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1}+h_{i1}h_{j3} & h_{i3}h_{j2}+h_{i2}h_{j3} & h_{i3}h_{j3}\end{bmatrix}^T \tag{2-37}$$

带入(2-33)有：

$$\begin{bmatrix}v_{12}^T\\ (v_{11}-v_{22})^T\end{bmatrix}b=0 \tag{2-38}$$

由一幅图片求取单应性矩阵 **H** 可以得到 2 个这样的方程，由 n 幅图片可以得到 2n 个方程组成的方程组，用矩阵形式表示为：

$$Vb=0 \tag{2-39}$$

其中 V 为 2n x 6 的矩阵，当 2n>=6 时可通过求取 V^TV 最小特征值对应的特征向量得到 b 的解，从而得到矩阵 **B**，由于

$$B=A^{-T}A^{-1}=\begin{bmatrix}B_{11} & B_{12} & B_{13}\\ B_{12} & B_{22} & B_{23}\\ B_{13} & B_{23} & B_{33}\end{bmatrix}=\begin{bmatrix}\frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma-u_0\beta}{\alpha^2\beta}\\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2}+\frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma-u_0\beta)}{\alpha^2\beta^2}-\frac{v_0}{\beta^2}\\ \frac{v_0\gamma-u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma-u_0\beta)}{\alpha^2\beta^2}-\frac{v_0}{\beta^2} & \frac{(v_0\gamma-u_0\beta)^2}{\alpha^2\beta^2}+\frac{v_0^2}{\beta^2}+1\end{bmatrix} \tag{2-40}$$

可以求出内参矩阵 **A** 矩阵的各个参数：

$$\begin{cases}r_1=\lambda A^{-1}h_1\\ r_2=\lambda A^{-1}h_2\\ r_3=r_1\times r_2\\ t=\lambda A^{-1}h_3\end{cases} \tag{2-41}$$

由(式 2.2)可以得到相应的外参矩阵参数(不同的单应性矩阵对应不同的外参矩阵)：

$$\begin{cases} v_0 = \dfrac{(B_{12}B_{13} - B_{11}B_{23})}{(B_{11}B_{22} - B_{12}^2)} \\ \lambda = B_{33} - \dfrac{B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})}{B_{11}} \\ \alpha = \sqrt{\dfrac{\lambda}{B_{11}}} \\ \beta = \sqrt{\dfrac{\lambda B_{11}}{B_{11}B_{22} - B_{12}^2}} \\ \gamma = \dfrac{-B_{12}\alpha^2\beta}{\lambda}, u0 = \dfrac{\gamma v_0}{\beta} - \dfrac{B_{13}\alpha^2}{\lambda} \end{cases} \tag{2-42}$$

至此，摄像机内参矩阵和外参矩阵的初始值(未考虑畸变)全部求解。

2.3.2 求取畸变参数

在上一小节中，通过线性模型求闭合形式解得到未考虑畸变的摄像机内参矩阵和外参矩阵。为了更加精确的描述 3 维物体与真实图像的关系，张正友标定方法还考虑了摄像头畸变、采用的非线性模型为：

$$\begin{cases} \tilde{x} = x + x\left[k_1(x^2+y^2) + k_2(x^2+y^2)^2\right] \\ \tilde{y} = y + y\left[k_1(x^2+y^2) + k_2(x^2+y^2)^2\right] \end{cases} \tag{2-43}$$

其中(x,y)为理想情况(线性模型理论值)下成像平面坐标系下点的坐标，$(\tilde{x},\tilde{y})$ 为实际(畸变)坐标，其中 k_1,k_2 畸变参数。可以看出该模型只考虑了径向畸变，并只考虑了 r^2 和 r^4 项的系数。相应地，(u,v)为理想情况(线性模型理论值)下图像上某一像素点的坐标，$(\tilde{u},\tilde{v})$ 为实际(畸变)坐标。根据 2.1.1 中成像平面坐标系和图像坐标系的转换关系有：

$$\begin{cases} \tilde{u} = u + (u - u_0)[k_1(x^2+y^2) + k_2(x^2+y^2)^2] \\ \tilde{v} = v + (v - v_0)[k_1(x^2+y^2) + k_2(x^2+y^2)^2] \end{cases} \tag{2-44}$$

矩阵形式为：

$$Dk = d \tag{2-45}$$

其中

$$D = \begin{bmatrix} (u-u_0)(x^2+y^2) & (u-u_0)(x^2+y^2)^2 \\ (v-v_0)(x^2+y^2) & (v-v_0)(x^2+y^2)^2 \end{bmatrix}, k = \begin{bmatrix} k_1 & k_2 \end{bmatrix}^T, d = \begin{bmatrix} \tilde{u}-u \\ \tilde{v}-v \end{bmatrix} \tag{2-46}$$

可以通过最小二乘法求出列向量 $k = (D^T D)^{-1} D^T d$，从而得到畸变参数 k_1、k_2。

2.3.3 摄像机参数精化

虽然在上一小节中考虑了摄像头的径向畸变，但是计算采用的参数是 2.3.1 中计算出的摄像机内参矩阵和外参矩阵，2.3.1 计算的摄像机内参矩阵和外参矩阵又是在假设没有畸变的前提下得到的，这些参数的误差应该还比较大。为了得到更加精确的参数(内参矩阵、

外参矩阵、畸变参数) ，张正友在其文章[12]中给出两种精化方法：

(1)迭代方法。即每次计算出畸变参数 k_1,k_2 之后进行矫正畸变，然后重新计算摄像机的内参和外参，再根据新的内参和外参重新计算 k_1,k_2,反复进行这两个步骤直到参数收敛。

(2)最大似然估计方法。使用 Levenberg-Marquardt(LM)算法对下式进行最优化：

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|m_{ij}-\tilde{m}(A,k1,k_2,R_i,t_i,M_j)\right\|^2 \tag{2-47}$$

其中 $\tilde{m}(A,k1,k_2,R_i,t_i,M_j)$ 为第 i 个标定板上点 M_j 的投影点，m_{ij} 为实际上对应点的坐标。$\mathbf{R_i},\mathbf{t_i}$ 为第 i 个标定板对应的旋转矩阵和平移矩阵。**A**、$\mathbf{R_i}$、$\mathbf{t_i}$ 使用 2.3.1 求得的结果初始化、k1,k2 可以采用 2.3.2 求得的结果初始化或者简单地初始化为 0。

2.3.4 小结

张正友标定方法包含如下几个过程：

(1)打印一张图案(黑白棋盘)并固定在一个较为平坦的物体表面(如玻璃)

(2)从多个(至少 3 个)不同的角度拍摄标定板，移动标定板或者摄像机均可。

(3)检测拍摄标定板图片上的特征点(如黑白棋盘的角点)的坐标。

(4)采用 2.3.1 节的方法求取摄像机的内参矩阵和外参矩阵。

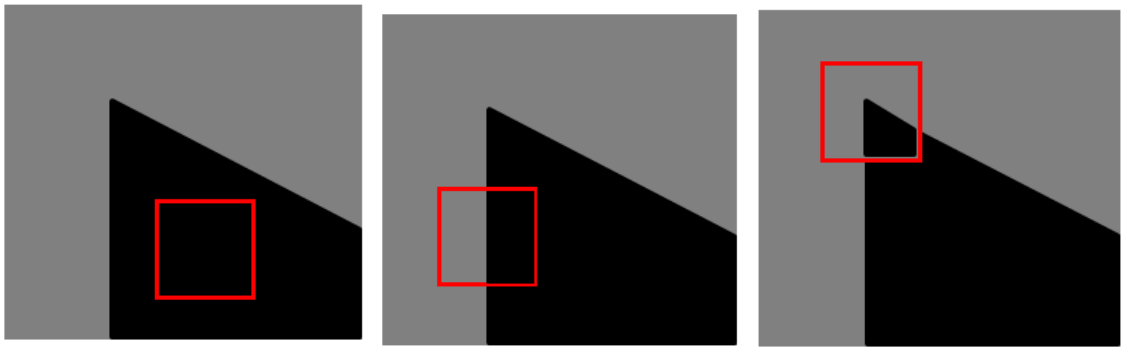
(5)采用 2.3.2 节的方法求取摄像机的畸变参数。

(6)通过(式 2.6)进行摄像机内外参数以及畸变参数精化。

2.4 基于灰度图像的角点检测

2.4.1 Moravec 角点检测算法

Moravec[28]角点检测子主要考虑一个局部窗口在若干个方向(4-8)移动产生的对应像素灰度值改变与图像特征的关系，如图 2.6 所示，考虑如下三类情况：



(a) 平坦区域 (b)边缘区域 (c)角点区域

图 2.6 Moravec 角点检测示意图

(1)如果图片在窗口附近变化平坦，窗口所有方向上的移动只会引起窗口内对应像素灰度值微小的变化，如图 2.6 a 所示。

(2)如果窗口位于图像边缘上，那么顺着边缘方向上的移动引起上述值微小的变化，垂

直于边缘方向上的移动将引起巨大的变化，如图 2.6 b 所示。

(3)如果窗口位于图像的角点上，则所有方向上的移动都将造成窗口内对应像素灰度值巨大的变化，如图 2.6 c 所示。

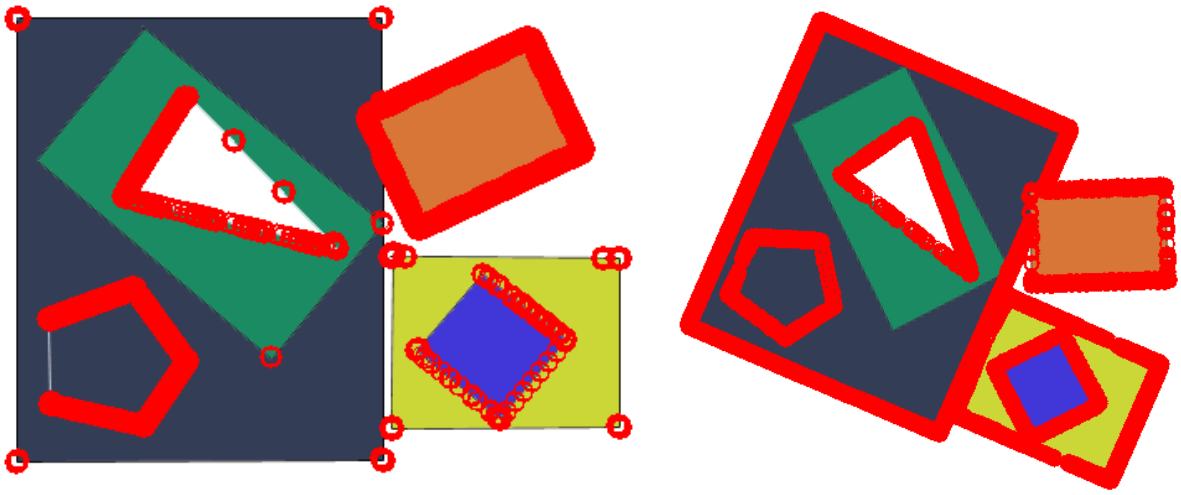
所以可以通过各个方向上移动引起的变化的最小值来判断是否任意方向的变化值都比较大。

向(x,y)方向移动产生的变化 $\mathbf{E_{x,y}}$ 可以通过以下方式度量：

$$E_{x,y}=\sum_{u,v} w_{u,v}\left|I_{x+u,y+v}-I_{u,v}\right|^2 \tag{2-48}$$

其中 w 表示所使用的二值化窗口，(x,y)包含[(1,0),(1,1),(0,1),(-1,1)]四个方向，Moravec 算法通过该点在 4 个方向上的 min $\mathbf{E_{x,y}}$ 是否大于某一阈值来判断该点是否为图像的角点。

Moravec 角点检测算法对边缘十分敏感，而且不具备旋转不变性,如图 2.7 所示:



(a)原图使用 Moravec 角点检测　　(b)旋转之后使用 Moravec 角点检测

图 2.7 Moravec 角点检测子效果图

2.4.2 Harris 角点检测算法

Harris[29]等人在 1988 年提出一种改进 Moravec 的角点检测算法，Moravec 算法存在如下不足：

(1)Moravec 角点检测算法只考虑了每个 45 度角方向，如图 2.7 中所示，不属于这几个方向的边缘会被检测为角点。

(2)Moravec 角点检测算法采用二值化窗口，对噪声较为敏感。

(3)Moravec 角点检测算法只关注这几个方向中 E 的最小值。

针对这三项不足，Harris 角点算法作了如下改进:

(1)Moravec 算法只计算了间隔 25 度的几个方向，Harris 算法使用 Taylor 展式近似计算任意方向:

$$E_{x,y}=\sum_{u,v} w_{u,v}\left|I_{x+u,y+v}-I_{u,v}\right|^2=\sum_{u,v} w_{u,v}\left|xX+yY+o(x^2,y^2)\right|^2 \tag{2-49}$$

其中 **X** 和 **Y** 分别为 x 和 y 方向上的梯度:

$$\begin{cases} X = I \otimes (-1,0,1) = \dfrac{\partial I}{\partial x} \\ Y = I \otimes (-1,0,1)^{\mathrm{T}} = \dfrac{\partial I}{\partial y} \end{cases} \tag{2-50}$$

则对于任意方向的微小移动(x,y):

$$E_{x,y} = Ax^2 + 2Cxy + By^2 \tag{2-51}$$

其中:

$$A = X^2 \otimes w, B = Y^2 \otimes w, C = (XY) \otimes w \tag{2-52}$$

w 为卷积窗口。

(2)Moravec 算法采用二值窗口，Harris 算法采用更加平滑的窗口，如高斯窗口:

$$w_{u,v} = e^{-\frac{(u^2+v^2)}{2\sigma^2}} \tag{2-53}$$

(3)Moravec 算法通过各个方向 $E_{x,y}$ 的最小值检测角点，Harris 使用一种新的判断方法:使用矩阵的形式表示 $E_{x,y}$ 为:

$$E(x,y) = (x,y)M(x,y)^T \tag{2-54}$$

其中

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix} \tag{2-55}$$

M 的两个特征值 α , β 和局部自相关函数的主曲率成正比，同 Moravec 算法一样，分如下三种情况讨论 α , β 和图像特征的关系:

(1)如果两个曲率都较小，即局部自相关函数整体较平坦，则所有方向上的移动将引起 E 的微小变化。

(2)如果其中一个曲率较大而另一个较小，即局部自相关函数呈山脊状。沿着山脊的移动引起 E 微小的变化，此时窗口处于图像边缘处。

(3)如果两个曲率都较大，则局部自相关函数呈山顶形状：所有方向的移动引起 **E** 剧烈的变化，此时窗口处于图像的角点处。

基于上述考虑，Harris 定义角点响应函数:

$$R = Det - kTr^2 \tag{2-56}$$

其中：

$$\begin{cases} Tr(\mathrm{M}) = \alpha + \beta = A + B \\ Det(\mathrm{M}) = \alpha\beta = AB - C^2 \end{cases} \tag{2-57}$$

使用 $Tr(\mathrm{M})$ 和 $Det(\mathrm{M})$ 而非 α、β，从而避免了对 **M** 进行显式的特征值分解。

2.4.3 Nobel 角点检测算法

由于 Harris 算法中参数 k 需要根据经验提前设定，Nobel[33]于 1988 年提出利用如下公

式计算角点的响应值：

$$cim = \frac{I_x^2 I_y^2 - (I_x I_y)^2}{I_x^2 + I_y^2} \tag{2-58}$$

采用上述公式计算角点的 CRF 值，从而避免的参数 k 对角点选取的影响，在实际应用中，通常选用这个改进的 Harris 角点检测算法进行检测：当 cim 值大于预定的阈值，则该点为角点候选点，通过非极大值抑制挑选出最终的角点。

2.4.4 Shi-Tomasi 角点检测算法

J.Shi 和 J.Tomasi[30]于 2000 年采用与 Harris 不同的角点响应函数。在 Harris 算法中角点响应函数定义为:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{2-59}$$

其中 λ_1 和 λ_2 分别为 **M** 矩阵的两个特征值。Shi-Tomasi 算法采用的角点响应函数为:即两个特征值中的最小值即:

$$R = \min(\lambda_1, \lambda_2) = \frac{A + B - \sqrt{(A-B)^2 + C^2}}{2} \tag{2-60}$$

2.4.5 亚像素精度角点检测

无论是 Moravec 算法、Harris 算法还是 Shi-Tomasi 算法，最终只能得到像素精度的角点坐标。为了得到更精确的标定结果，需要提高角点检测算法的精度，亚像素精度角点坐标的获取可以采用通用的(1)插值法 以及(2)向量法。

(1)插值法 在图像处理中，插值通常用来获取亚像素精度的结果。在文献[31]中，笔者采用二次多项式逼近 Harris 角点响应函数：

$$CRF(u,v) = a_1 u^2 + a_2 uv + a_3 v^2 + a_4 u + a_5 v + a_6 \tag{2-61}$$

其中，a_1-a_6 为拟合平面系数，通过(u,v)邻域内多个点组成超定线性方程组，通过最小二乘法求解出 a_1-a_6,用矩阵表示为：

$$AX = B \tag{2-63}$$

其中:

$$A = \begin{bmatrix} u_1^2 & u_1 v_1 & v_1^2 & u_1 & v_1 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ u_n^2 & u_n v_n & v_n^2 & u_n & v_n & 1 \end{bmatrix} \tag{2-64}$$

$$X = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{bmatrix} \tag{2-65}$$

$$B = \begin{bmatrix} CRF(u_1, v_1) \\ \cdots \\ CRF(u_n, v_n) \end{bmatrix} \tag{2-66}$$

(u_i, v_i) 为 (u, v) 邻域内的点， $CRF(u_i, v_i)$ 为对应的角点响应函数值。

拟合的二次平面的极大值处的坐标为亚像素精度角点坐标，在二次平面的极大值处有：

$$\begin{cases}\dfrac{\partial CRF(u,v)}{\partial u}=2a_1u+a_2v+a_4=0\\ \dfrac{\partial CRF(u,v)}{\partial v}=2a_3v+a_2u+a_5=0\end{cases} \tag{2-67}$$

则亚像素精度角点坐标为：

$$\begin{cases}u=\dfrac{2a_3a_4-a_2a_5}{a_2^2-4a_1a_3}\\ v=\dfrac{2a_1a_5-a_2a_4}{a_2^2-4a_1a_3}\end{cases} \tag{2-68}$$

(2)向量法 如图 2.8 所示，对于一个方格形状的角点 q 和 q 邻域内某点 p，图像在 p 处的梯度为$\nabla I(p)$ [32]。

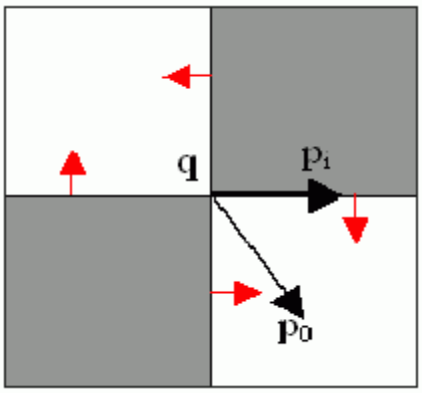


图 2.8 方格状角点亚像素提取

$\overrightarrow{pq}$的位置有两种情况:位于边缘上(如图中 p_1)、位于平坦区域(全黑或全白,如图中 p_2)。当$\overrightarrow{pq}$位于平坦区域时，$\|\nabla I(p)\|=0$，$\overrightarrow{pq}\cdot\nabla I(p)=0$ ；当$\overrightarrow{pq}$位于方格边缘时，由于$\nabla I(p)$和$\overrightarrow{pq}$垂直，故$\overrightarrow{pq}\cdot\nabla I(p)=0$。综上所述，对于 q 邻域内的点 p，均有$\overrightarrow{pq}\cdot\nabla I(p)=0$即：

$$(u_q-u_p,v_q-v_p)\cdot(du_p,dv_p)=0 \tag{2-69}$$

其中(u_q,v_q)和(u_p,v_p)分别为 q 和 p 两点的坐标，(du_p,dv_p)为图像在 p 点处的梯度。整理得：

$$u_q du_p+v_q dv_p=u_p du_p+v_p dv_p \tag{2-70}$$

取 q 邻域内的 n 个点可以得到 n 个上述方程，写成矩阵的形式为:

$$AX=B \tag{2-71}$$

其中

$$A=\begin{bmatrix}du_1 & dv_1\\ \cdots & \cdots\\ du_n & dv_n\end{bmatrix},B=\begin{bmatrix}udu_1+vdv_1\\ \cdots\\ udu_n+vdv_n\end{bmatrix} \tag{2-72}$$

X=[u v]^T,(u_i,v_i)为 q(初始化为像素级别角点坐标)邻域内第 i 个点的坐标，(du_i,dv_i)为图像在

该点处的梯度，(u,v) 为待解的亚像素精度的角点坐标。若采用最小二乘法解上述超定线性方程，解为：

$$X=(A^TA)^{-1}A^TB \tag{2-73}$$

将解出的坐标作为新的 q 的坐标重复以上过程，直到 q 的坐标收敛到某一范围，此时 q 的坐标即为亚像素精度的角点坐标。

2.5 亚像素精度角点检测的实现

2.5.1 像素精度角点提取的实现

根据 2.4 节中各个角点算法的数学原理，采用 Harris 算法、Nobel 算法和 Shi-Tomasi 算法检测角点分为如下几个步骤：

(1)彩色图片转为灰度图

(2)求取 x 和 y 方向上的梯度

(3)用高斯模板求卷积

(4)求角点响应值

(5)非极大值抑制

(1)彩色图片转为灰度图 通过公式

$$I=0.299R+0.587G+0.114B \tag{2-74}$$

其中 R、G、B 分别为 RGB 格式图像的像素点三个通道的值，I 为转化后的灰度，转换效果如图 2.9 所示：

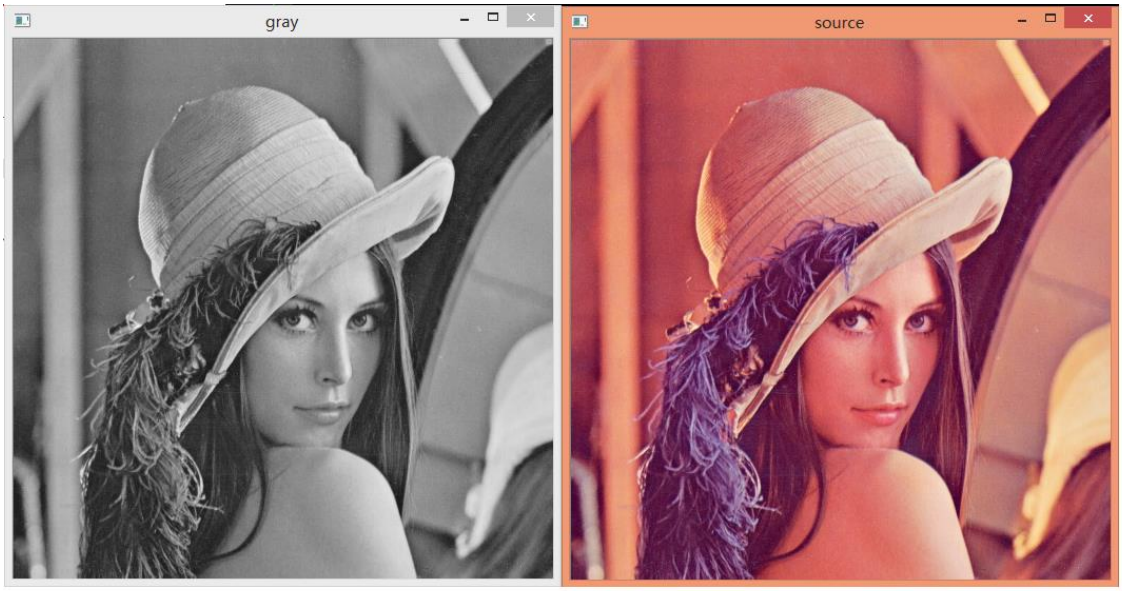


图 2.9 右图为 RGB 原图，左图为灰度图

(2)求取 x 和 y 方向上的梯度 将原图转化为灰度图之后，分别使用下面两个模板窗口求取两个方向上的梯度值：

-1		1
-1		1
-1		1

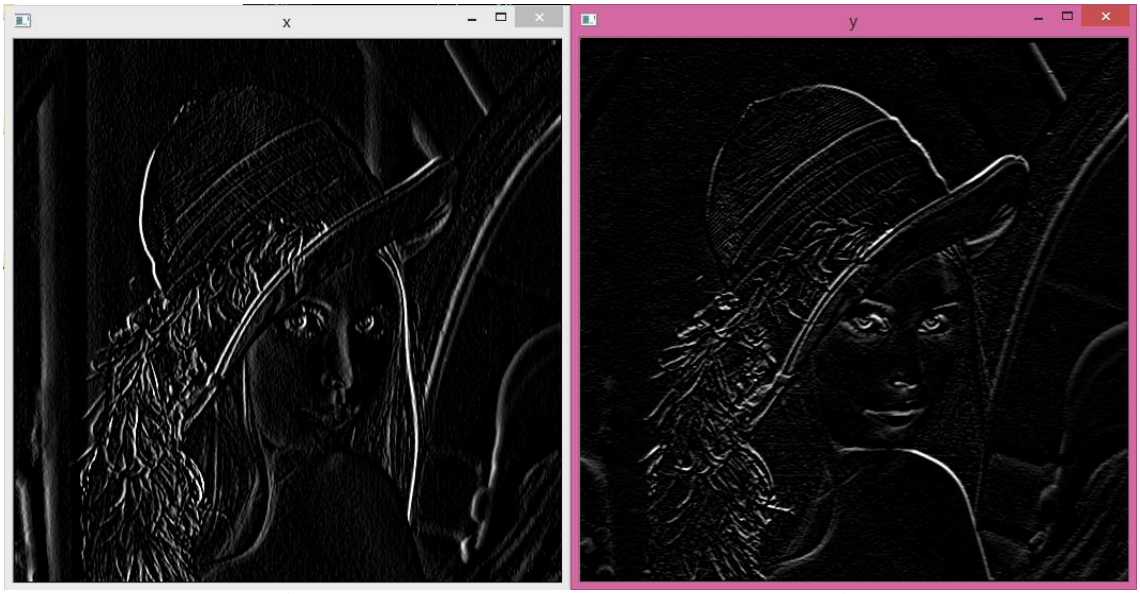
x 方向

-1	-1	-1
1	1	1

y 方向

图 2.10 使用两个 3*3 模板在两个方向求梯度

为了能直观的观察到求梯度的效果，将求得的梯度值归一化到[0，255]区间内，转化为灰度图便于显示，结果如图 2.11 所示：



x 方向 y 方向

图 2.11 两个方向上的梯度图

求得每个像素点 x、y 方向的梯度 Ix,Iy 值之后，便能得到该像素点 Ix*Ix,Ix*Iy,Iy*Iy 的值。

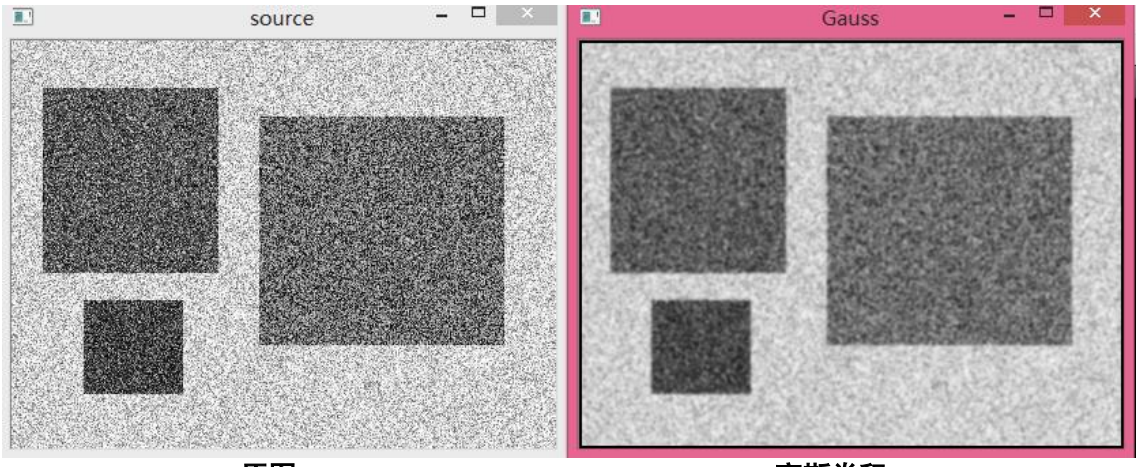
(3)使用高斯模板求卷积 采用以下 5*5 的高斯窗口对 Ix*Ix,Ix*Iy,Iy*Iy 进行卷积运算，为了避免浮点运算，采用整数窗口近似：

1/273 ＊

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

图 2.12 5*5 高斯卷积窗口

为了直观的展示高斯卷积的效果，对一幅图片进行高斯卷积测试，如图 2.13 所示，进行高斯卷积之后的图片变得模糊，细节和者噪声被平滑抑制。



原图 **高斯卷积**

图 2.13 对图片进行高斯卷积效果图

(4)对 IxIx,IxIy,IyIy 分别进行高斯卷积之后，采用不同的算法求取角点响应值:

Harris: $$cim = IxIx \times IyIy - IxIy \times IxIy - k \times (IxIx + IyIy)^2 \quad (2\text{-}75)$$

Nobel: $$cim = \frac{IxIx \quad IyIy \quad IxIy}{IxIx \quad IyIy} \quad (2\text{-}76)$$

Shi-Tomasi: $$cim = \frac{IxIx + IyIy - \sqrt{(IxIx - IyIy)^2 + 4(IxIy)^2}}{2} \quad (2\text{-}77)$$

当 cim 大于阈值 threshold 时，该点即被认为是角点，分别采用上述三种算法对一幅图片提取角点：

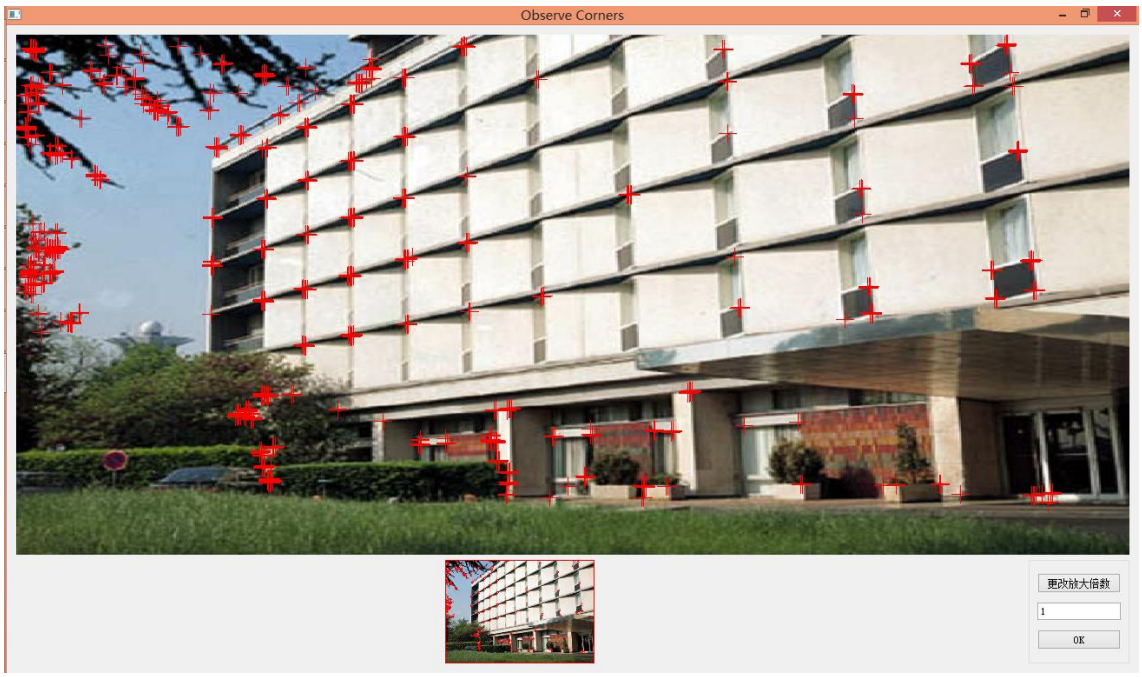


图 2.14 Harris 角点提取算法 k=0.04 threshold= 400000000

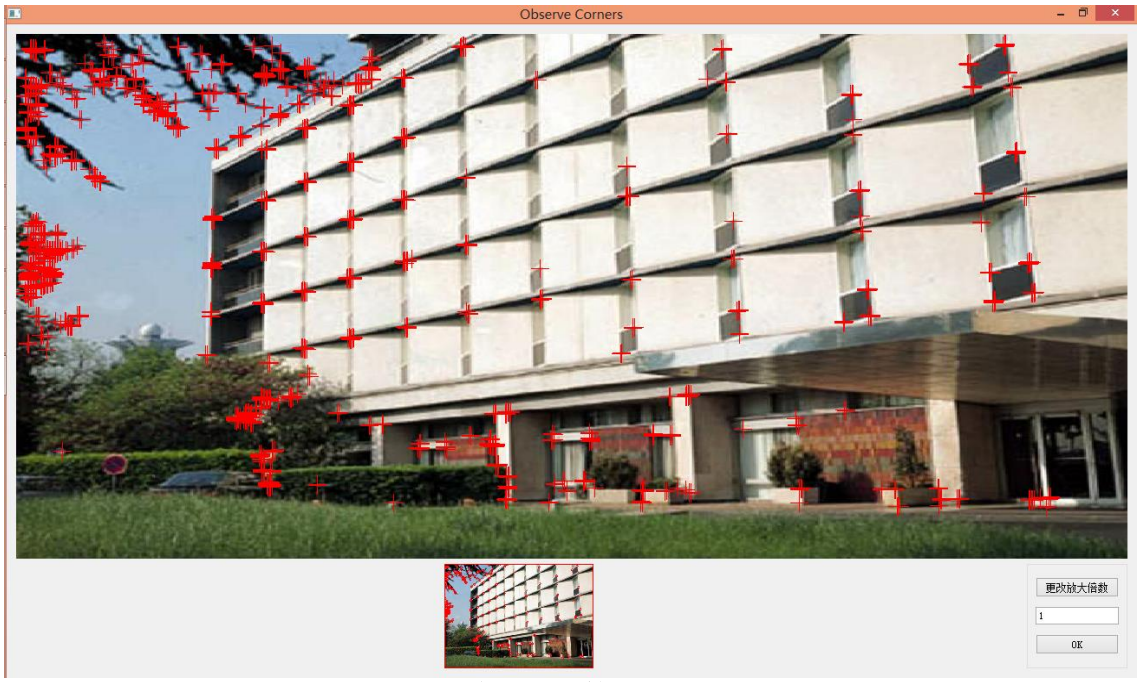


图 2.15 Nobel 角点提取算法 threshold= 8000

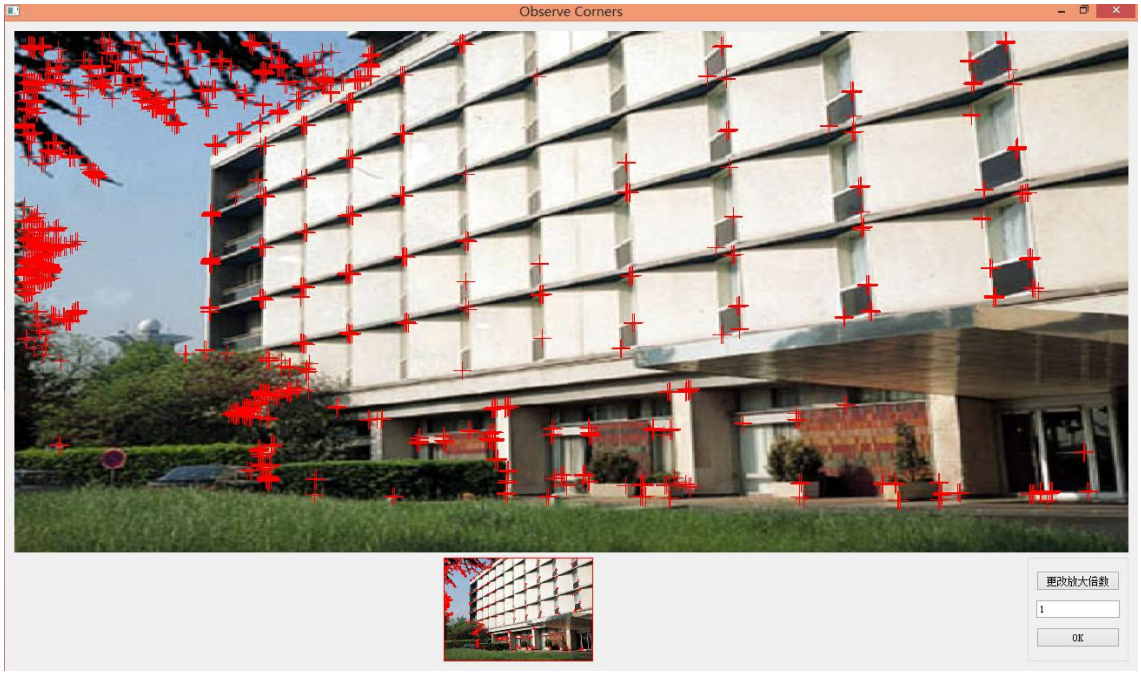


图 2.16 Shi-Tomasi 角点提取算法 threshold= 10000

(5) 非极大值抑制 若单纯的通过阈值判断像素是否为角点，容易在图像中形成角点群，即角点总是成群出现，在摄像机标定过程中，通常只希望棋盘角点的地方存在一个角点。非极大值抑制策略即像素点的角点响应值不仅需要大于阈值，还需要是邻域范围内角点响应值的最大值，同时满足上述两个条件则判断为角点。

采用 Nobel 角点提取算法以及采用了非极大值抑制的结果：

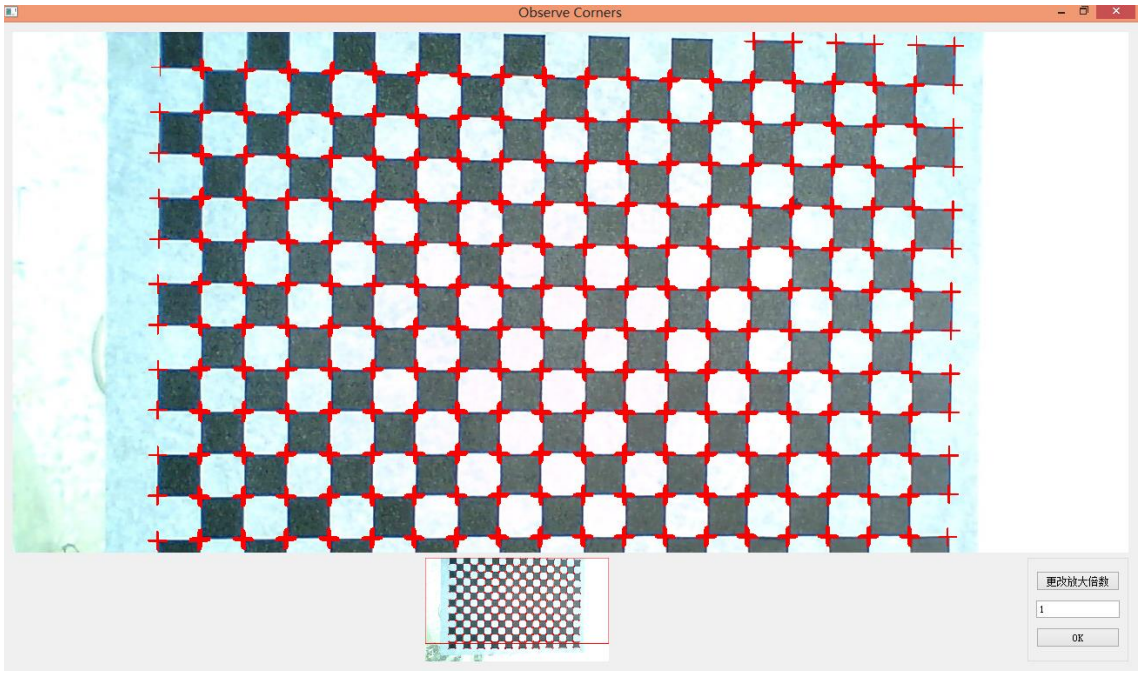


图 2.17 Nobel 角点提取算法 threshold= 15000

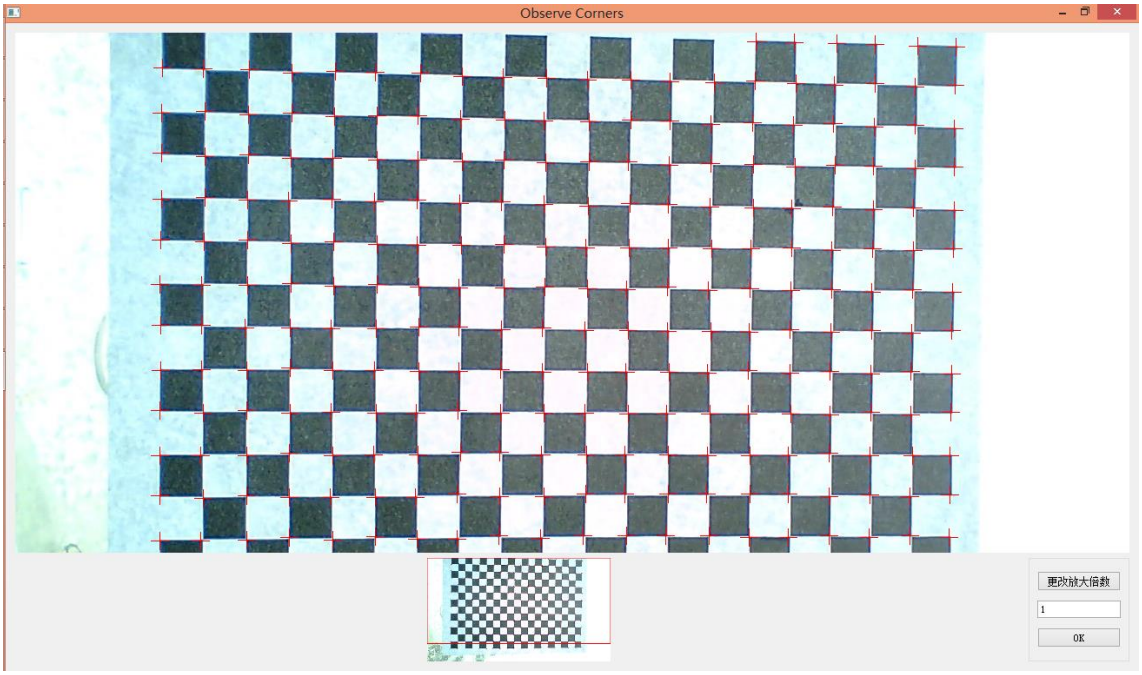


图 2.18 采用非极大值抑制的 Nobel 角点提取算法 threshold= 15000

2.5.2 亚像素精度精化实现

无论是使用平面拟合的方式还是采用向量法都涉及到最小二乘法求解超定线性方程组，需要求取矩阵的逆，由于需要计算的矩阵的规模并不非常大，采用高斯主消元法求取矩阵的逆,伪代码描述如下:

```
Mat func(Mat A){
Mat E;//单位矩阵
    for(i = 0;i<A.rows;i++){//化为上三角矩阵
        m = selectmax(A,i,rows-1);//选择 A(x,i)绝对值最大的 x(i<=x<rows)
        swap(A,i,m);                    //交换矩阵 A 的 i，m 两行
        swap(E,I,m);                    //对单位矩阵实施同样的操作
        for(j = i+1;j<A.rows;j++){
            k = -1 * A(j,i)/A(i,i);
            add(A,i,k,j);               //将 i 行元素乘以 k 加到 j 行元素上
            add(E,I,k,j);              //对单位矩阵实施同样的操作
        }
    }
    for(i = A.rows-1;i>=0;i--){//化为对角矩阵
        for(j = i-1;j>=0;j--){
            k = -1 * A(j,i)/A(i,i);
            add(A,i,k,j);               //将 i 行元素乘以 k 加到 j 行元素上
            add(E,I,k,j);              //对单位矩阵实施同样的操作
        }
    }
    for(i = 0;i<A.rows;i++){//化为单位矩阵
        k = 1/A(i,i);
        mul(A,i,k);                     //将 i 行元素全部乘以 k
        mul(E,i,k);                     //对单位矩阵实施同样的操作
    }
    return E;                           //该矩阵即为 A-1
}
```

(1)二次平面拟合方法求亚像素精度角点 在 2.4.5 节中提到如下矩阵方程计算二次拟合平面的各个参数：

$$AX = B \tag{2-78}$$

其中：

$$A = \begin{bmatrix} u_1^2 & u_1v_1 & v_1^2 & u_1 & v_1 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ u_n^2 & u_nv_n & v_n^2 & u_n & v_n & 1 \end{bmatrix}, B = \begin{bmatrix} CRF(u_1, v_1) \\ \cdots \\ CRF(u_n, v_n) \end{bmatrix} \tag{2-79}$$

在实际的计算中，若直接采用角点(u,v)邻域内点坐标(u_i,v_i)计算的话，$\mathbf{A^TA}$ 内的元素会变得非常大，从而影响计算。将参考坐标系的坐标原点平移到(u,v)，即角点(u,v)邻域内的坐标变成(u_i –u ,v_i–v),将最后的计算结果(x,y)再平移(u,v)即(x+u,y+v)得到真实的图像坐标。

(2)向量法求亚像素精度角点 在 2.4.5 节中通过求解矩阵方程解出亚像素角点的坐标

$$AX = B \tag{2-80}$$

其中

$$A = \begin{bmatrix} du_1 & dv_1 \\ \cdots & \cdots \\ du_n & dv_n \end{bmatrix}, B = \begin{bmatrix} udu_1 + vdv_1 \\ \cdots \\ udu_n + vdv_n \end{bmatrix} \tag{2-81}$$

对于浮点型的角点坐标(u_i,v_i)，该像素点处的梯度使用(Ix([u_i],[v_i]),Iy([u_i],[v_i]))近似，其中[x]为 x 取整数部分的值。同二次平面拟合的方式一样，将参考坐标系平移到(u,v), 角点(u,v)

邻域内的坐标变成(u_i –u ,v_i–v),将最后的计算结果(x,y)再平移(u,v)即(x+u,y+v)得到真实的亚像素图像坐标。

将得到的图像坐标作为新的角点坐标，重复以上处理过程，直到两次结果收敛(或者超过最大迭代次数)终止迭代。

图 2.19 采用二次平面拟合提取亚像素角点 上为局部放大 50 倍 下为缩略图

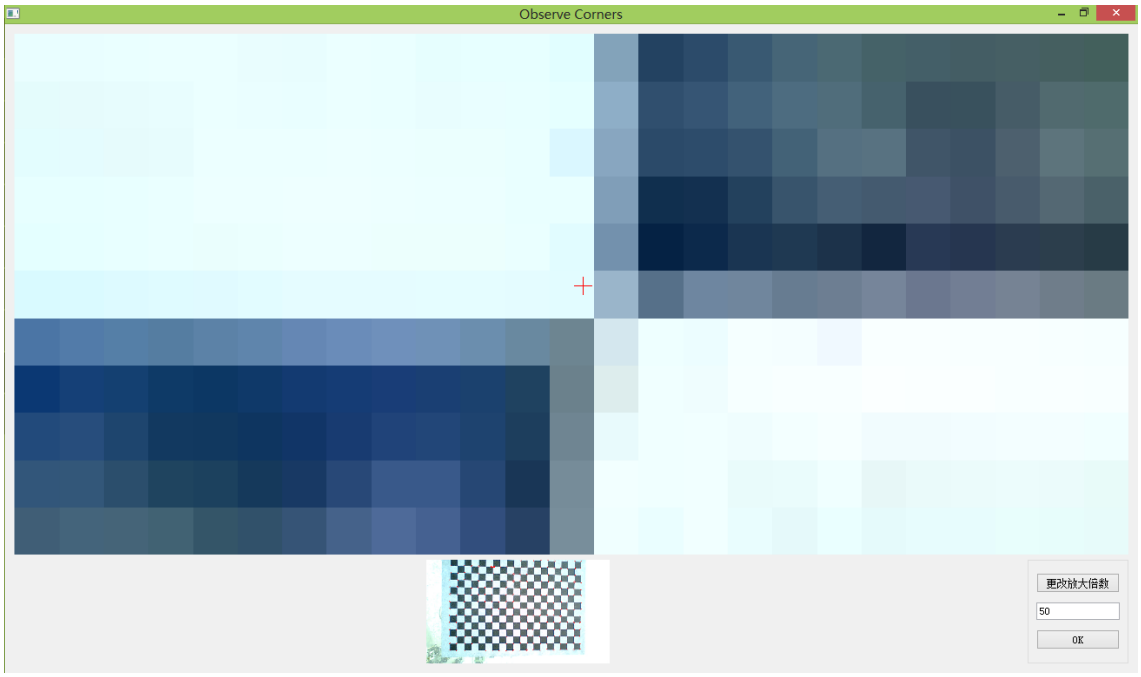


图 2.20 采用向量方法提取亚像素角点 上为局部放大 50 倍 下为缩略图

分别采用上述两种方法提取棋盘图片，两种算法的实验结果分别如图 2.19、2.20 所示。

2.6 标定实现

采用张正友的标定方法中关键的步骤即找到标定板(如标定棋盘)上的点和图片上像素的对应关系，本文中采用的标定板为黑白相间的方格棋盘标定板，使用黑色方格的角点作为标定的特征点。方格角点的三维坐标通过设置方格的打印宽度(如 40mm)得到(z 坐标为0)，图片上对应的角点坐标由 2.5 节中亚像素精度角点提取得到。

2.6.1 半自动提取角点

由于通常图片中不仅包含棋盘的信息，还有一部分环境信息，若直接对整张图片提取角点，容易包含图片中的非棋盘角点。另外，在实际的标定过程中，靠近打印纸张边缘的棋盘角点通常误差较大(纸张变形)，所以也并非所有的棋盘角点都要参与计算，若所有的角点全部通过人工选取，由于标定中一张图片上需要提取的点数就非常多(5*5-20*20)，标定通常需要 10-30 张图片，人工选取的工作量非常巨大。

本文采用半自动的角点提取方式：对于每张图片，人工选取 4 个角点，由于棋盘为方格形状，通过计算获取 4 个角点构成的矩形中包含的棋盘角点坐标，在这些坐标周围一定范围内搜索角点坐标得到若干角点坐标，另外，将所选矩形的左上角角点对应的棋盘角点位置定义为三维坐标系的原点坐标，根据棋盘方格的边长推算出所有角点的三维坐标。

实现流程如下：

(1)提取出整张图片上所有角点的亚像素精度坐标

(2)人工选取 4 个角点坐标

(3)通过计算得到选取矩形的规模(如 3*5)

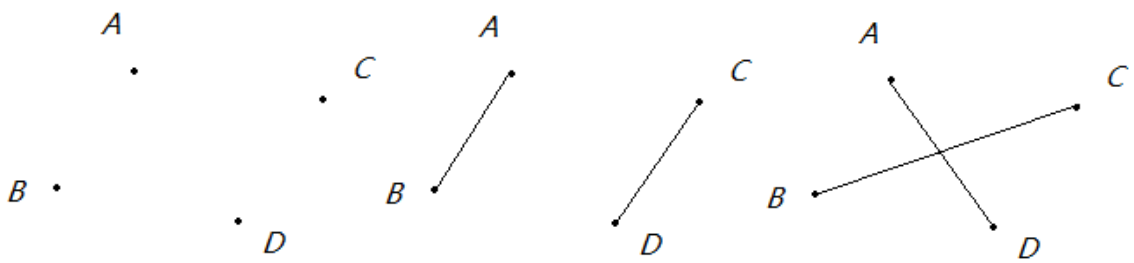
(4)得到图片角点和棋盘角点对应坐标序列

(5)通过张正友标定算法计算摄像机的内参矩阵和畸变矩阵

其中第(3)步骤的实现细节如下：

(1)通过交叉检测算法得到所选 4 个点的相对关系

如图 2.21 所示，假设提取的角点为 ABCD，为了判断两点之间的相对关系是为邻边关系(a)还是对角线关系(b)，通过判断两点连线线段是否和另外两点连线线段相交，如假设线段 AB 和线段 CD 相交，则 AB 两点属于对角边关系，否则属于相邻边关系。



人工选取 4 个角点　　(a)相邻边关系　　(b)对角线关系

图 2.21 判断 4 个角点相对关系

(2)通过检测 ABCD 四条边上的角点得到四边形网格

首先在所选的 A、B、C、D 四个点周围一定范围内搜索角点，设搜索到对应点的角点为 EFGH，再对每一条边 EF、FG、GH、HE 边进行如下操作：

a 遍历图片上所有的角点，若角点位于边(线段)上，则将该角点放入队列 q

b 将队列 q 按照 x 坐标或者 y 坐标进行排序

得到每一条边上的 n-1 个 n 等分(这里的等分指的是原棋盘标定板上意义的等分，图片中由于投影变换并非等分)点之后，可以得到 EFGH 的角点网格：

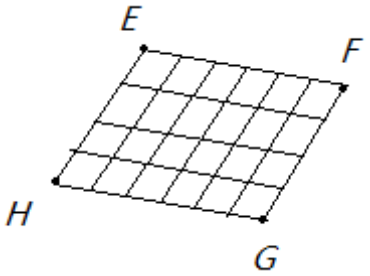
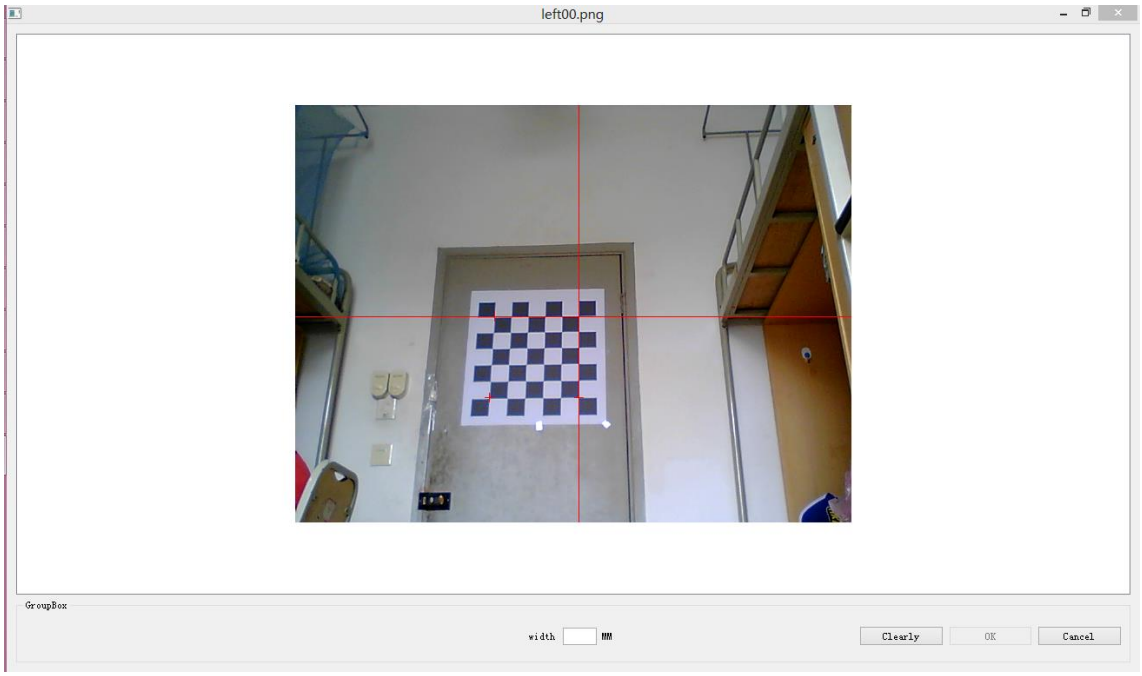


图 2.22 判断 4 个角点得到的角点网格

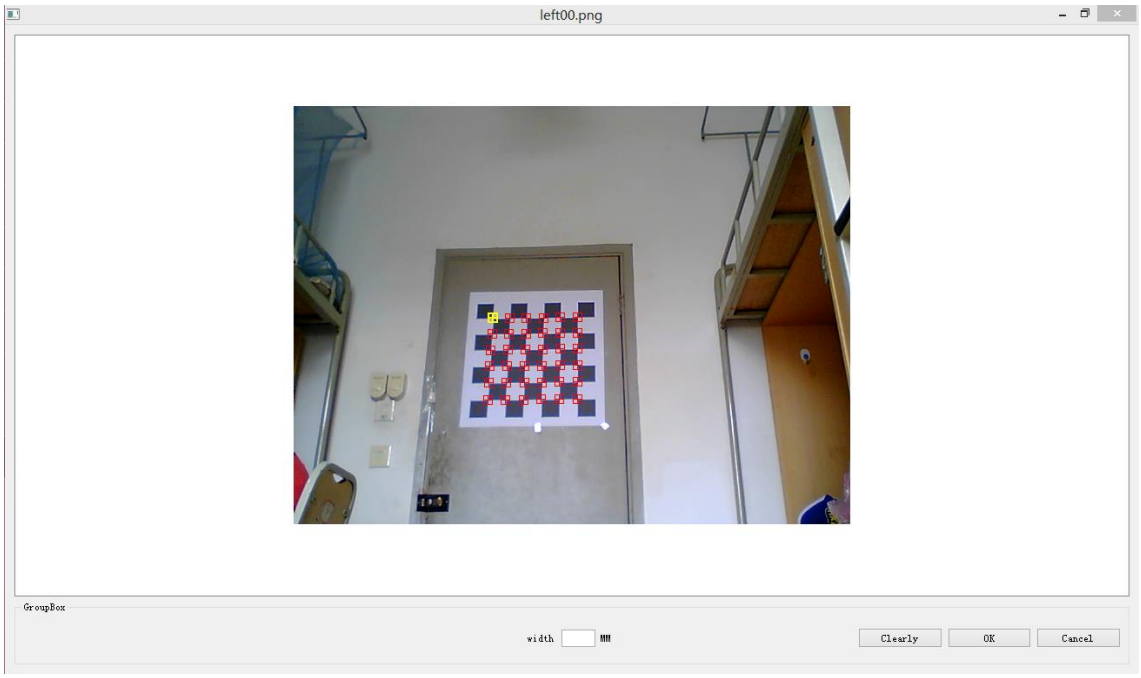
得到 EFGH 网格之后，同样在每个网格周围搜索角点得到角点坐标，以 E 所在棋盘角点为三维坐标原点可以得到角点对应的三维坐标(z 坐标为 0)。

2.6.2 标定实验结果

实验采用 75mm*75mm 的棋盘彩色(油墨)打印(粉墨打印机打印棋盘黑色区域不均匀)：从不同的角度拍摄 30 了张图片，标定过程如图 2.23 所示：



(a) 手工选取 4 个角点



(b) 得到角点网格

图 2.23 标定过程

对两个摄像头都进行标定，标定结果如图 2.24 所示：

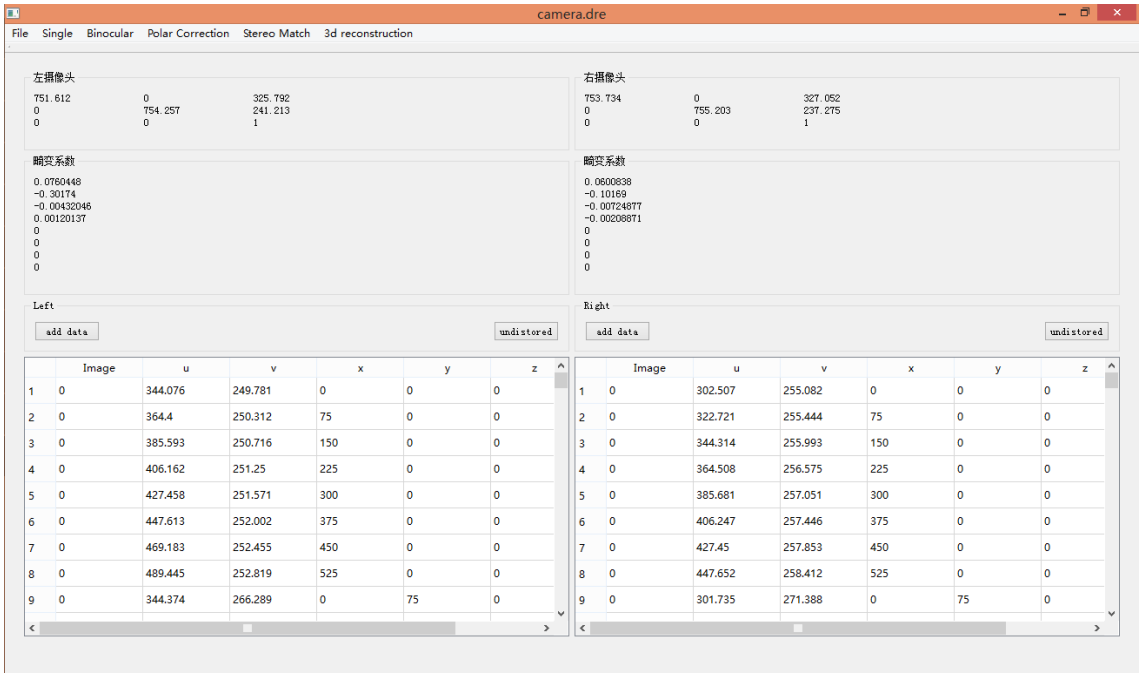


图 2.24 左右摄像机标定结果

其中左摄像机内参矩阵为：

$$\begin{bmatrix} 751.612 & 0 & 325.792 \\ 0 & 754.257 & 241.213 \\ 0 & 0 & 1 \end{bmatrix}$$

右摄像机内参矩阵为：

$$\begin{bmatrix} 753.734 & 0 & 327.052 \\ 0 & 755.203 & 237.275 \\ 0 & 0 & 1 \end{bmatrix}$$

左摄像机畸变系数 k1=0.0760448 k2=-0.30174 p1=-0.00432046 p2=0.00120137

右摄像机畸变系数 k1=0.0600838 k2=-0.10169 p1=-0.00724877 p2=0.00208871

k1 和 k2 为径向畸变系数，p1 和 p2 为切向畸变系数。

从标定结果可以看出，两个摄像头参数基本相同。另外，内参矩阵中 fx 和 fy 的值非常接近表明摄像机像素点几乎为方格形状。

3 畸变矫正以及立体矫正

3.1 畸变矫正

在 2.6 节中对摄像机进行了标定，并得到摄像机的内参和畸变系数，本小节主要研究通过摄像机内参以及畸变系数对畸变图像进行矫正。

在 2.1 节中，实际成像坐标系坐标(畸变)和理论(未发生畸变)成像坐标系坐标为如下关系：

$$\begin{cases} x'' = x'(1+k_1r^2+k_2r^4+k_3r^6)+2p_1x'y'+p_2(r^2+2x'^2) \\ y'' = y'(1+k_1r^2+k_2r^4+k_3r^6)+p_1(r^2+2y'^2)+2p_2x'y' \end{cases} \tag{3-1}$$

其中(x',y')为理论成像坐标，(x^n,y^n)为实际成像坐标。

以及成像坐标系 OXY 和图像坐标系 UV 的转换关系，可以建立起理论图像坐标(未畸变)和实际图像坐标(畸变)的映射关系：

对于未畸变图像的坐标(u,v)，可经过如下步骤计算出发生畸变之后坐标(u',v')，

$$\begin{cases} x' = \dfrac{u-u_0}{f_x} \\ y' = \dfrac{v-v_0}{f_y} \\ x'' = x'(1+k_1r^2+k_2r^4+k_3r^6)+2p_1x'y'+p_2(r^2+2x'^2) \\ y'' = y'(1+k_1r^2+k_2r^4+k_3r^6)+p_1(r^2+2y'^2)+2p_2x'y' \\ u' = x''f_x+u_0 \\ v' = y''f_y+v_0 \end{cases} \tag{3-2}$$

得到未畸变图像坐标(u,v)和实际图像坐标(u',v')的映射之后，便能得到未畸变的图像，即矫正之后的图像，图 3.1 为对 opencv 自带标定图片进行畸变矫正的效果：

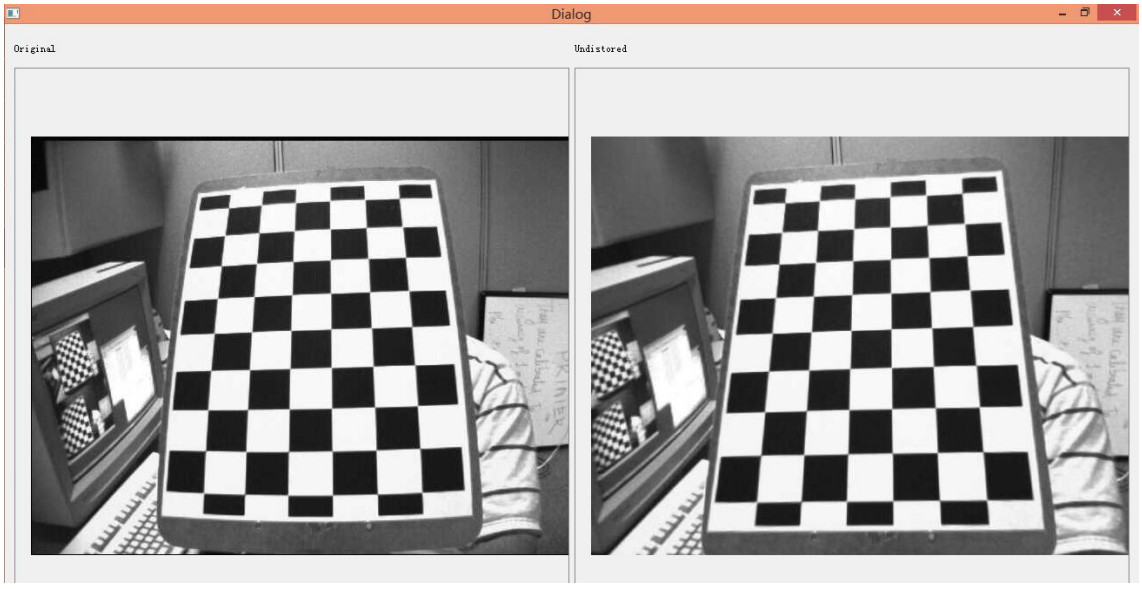


图 3.1 畸变矫正，左图为原始图像，右图为矫正后的图像

3.2 双目标定

为了进行极线矫正以及三维重建，需要得到左右摄像机的相对位置，即右摄像机坐标系相对于左摄像机坐标系的旋转矩阵 **R** 和平移矩阵 **T**。

假设左右摄像机坐标系相对于世界坐标系的旋转矩阵和平移矩阵分别为 R_l、R_r 以及 T_l、T_r，则对于世界坐标系中的一点 **P**,在左右摄像机坐标系内的坐标分别为 P_l 和 P_r 则：

$$\begin{cases} P_l = R_l P + T_l \\ P_r = R_r P + T_r \end{cases} \tag{3-3}$$

消去 P 有：

$$P_r = R_r R_r^{-1} P_l + T_r - R_r R_r^{-1} T_l \tag{3-4}$$

这样便得到左摄像机坐标系坐标 P_l 到右摄像机坐标系 P_r 的转换关系，由于 R_l 和 R_r 单位正交矩阵，则 $R_r R_r^{-1}$ 也为单位正交矩阵，由此可得右摄像机坐标系相对于左摄像机坐标系的旋转矩阵 $R = R_r R_r^{-1}$，平移矩阵 $T = T_r - R_r R_r^{-1} T_l$。

在进行双目标定实验中，可以直接进行双目标定实验，即左右摄像机同时拍摄不同角度的多张图片(在拍摄过程中左右摄像机的相对位置不能改变)，也可以先分别进行单目标定实验，将得到的摄像机内参参与双目标定时外参的计算，相比而言，后者结果更为精确。

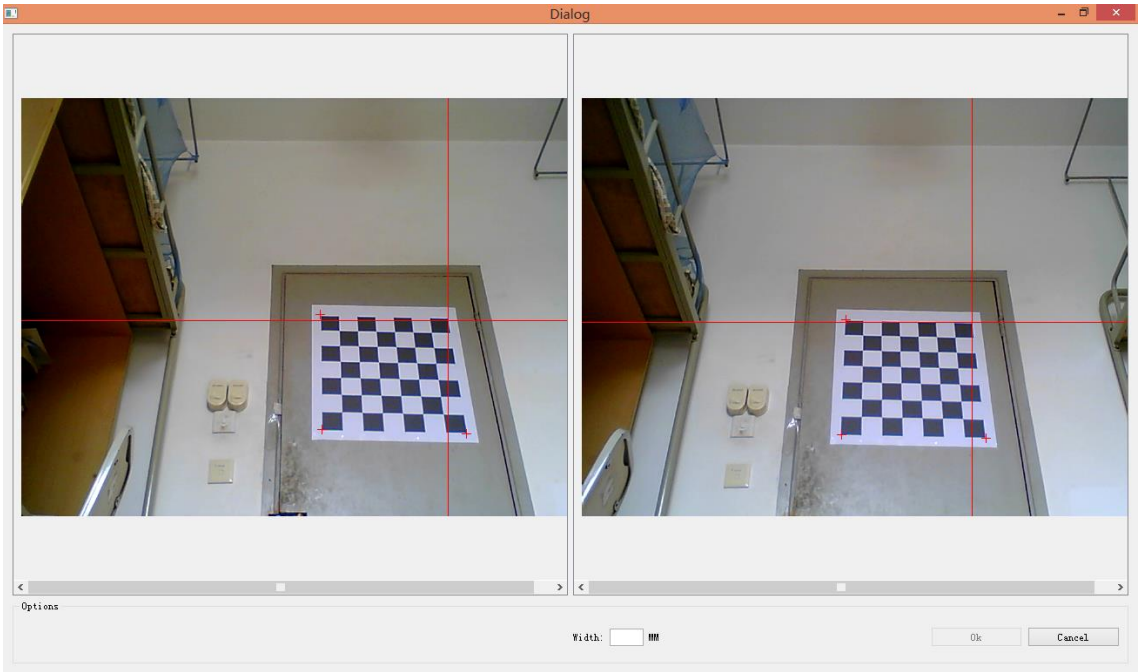


图 3.2 双目标定实验过程

双目标定程序仍然采用同单目标定相同的半自动角点检测方法，如图 3.2 所示，左右两副图片需人工选取 4 个角点，需要注意的是，左右图片选取的 4 个角点必须相同。

标定结果如图 3.3 所示，得到右摄像机坐标系相对于左摄像机坐标系的旋转矩阵 **R** 和平移矩阵 **T** 分别为：

$$R=\begin{bmatrix} 0.999995 & -0.001136 & 0.003064 \\ 0.001101 & 0.999933 & 0.011505 \\ -0.003077 & -0.011501 & 0.999929 \end{bmatrix}, T=\begin{bmatrix} 161.872 & 2.5572 & 8.33404 \end{bmatrix}^T$$

标定中采用的物理单位为 mm，两个摄像机几乎平行放置，所以 **R** 十分接近于单位矩阵。

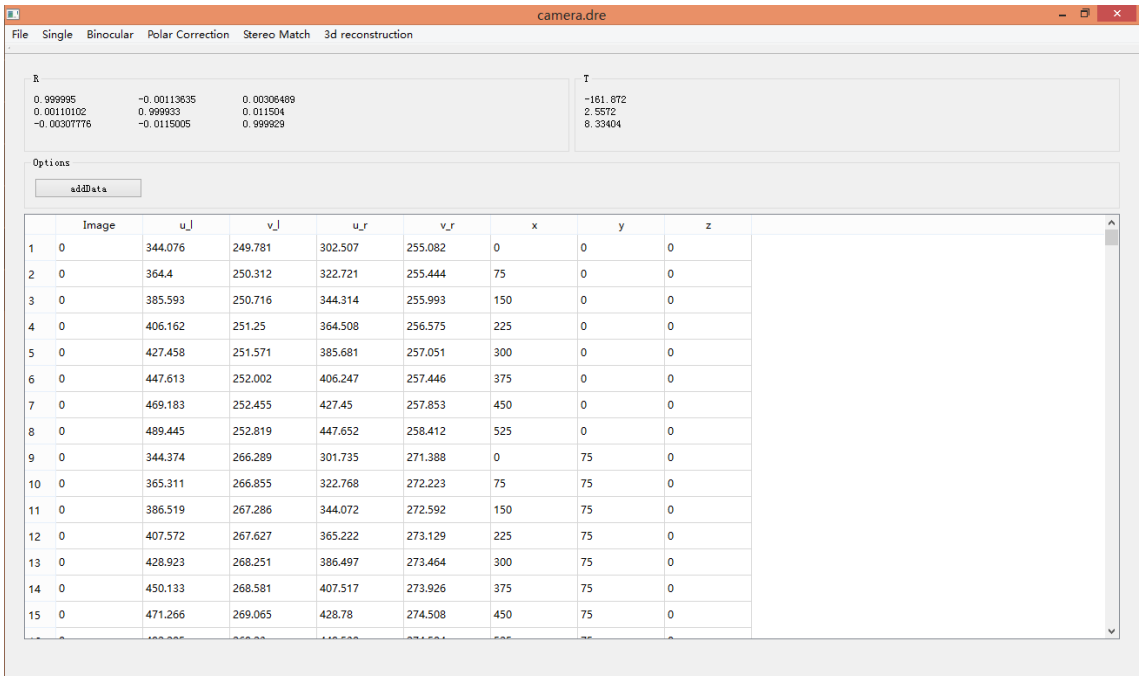


	Image	u_l	v_l	u_r	v_r	x	y	z
1	0	344.076	249.781	302.507	255.082	0	0	0
2	0	364.4	250.312	322.721	255.444	75	0	0
3	0	385.593	250.716	344.314	255.993	150	0	0
4	0	406.162	251.25	364.508	256.575	225	0	0
5	0	427.458	251.571	385.681	257.051	300	0	0
6	0	447.613	252.002	406.247	257.446	375	0	0
7	0	469.183	252.455	427.45	257.853	450	0	0
8	0	489.445	252.819	447.652	258.412	525	0	0
9	0	344.374	266.289	301.735	271.388	0	75	0
10	0	365.311	266.855	322.768	272.223	75	75	0
11	0	386.519	267.286	344.072	272.592	150	75	0
12	0	407.572	267.627	365.222	273.129	225	75	0
13	0	428.923	268.251	386.497	273.464	300	75	0
14	0	450.133	268.581	407.517	273.926	375	75	0
15	0	471.266	269.065	428.78	274.508	450	75	0

图 3.3 双目标定实验结果

3.3 立体矫正

3.3.1 平行摄像机成像模型

对于一对立体摄像机，假设两台摄像机的像平面精确位于同一平面上,光轴严格平行,距离一定,焦距相同,左右摄像机图像每一行严格对齐,平行摄像机成像模型如图 3.4 所示：

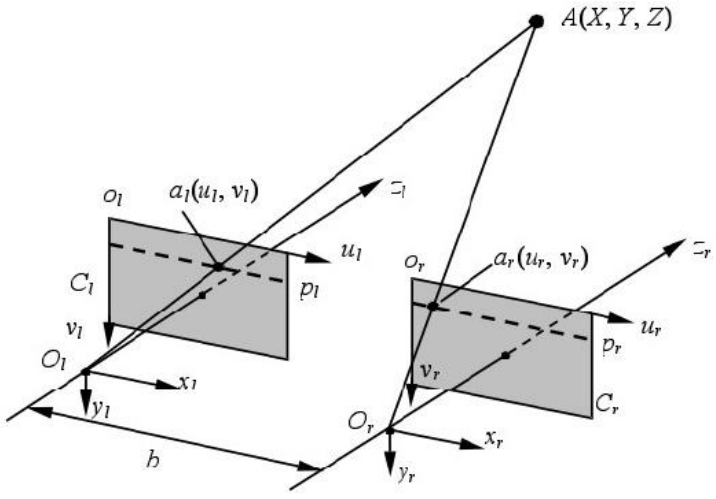


图 3.4 平行摄像机成像模型

3.3.2 立体矫正

平行摄像机模型无论是对于立体匹配还是三维重建，计算都得到了极大的简化。虽然在实际情况中，要求两个摄像机物理上为理想平行是非常困难的，但是通过立体矫正，可以从数学上得到平行的摄像机模型。

给定立体图像间的旋转矩阵和平移**(R,T)**，立体矫正的 Bouguet 算法[36]就是简单地使两图像中的每一幅重投影次数最小化，同时使得观测面积最大化。

Bouguet 算法分两步得到将旋转矩阵 R_l,R_r,左右摄像机通过 R_l,R_r 绕各自的投影中心旋转，使得极线变得水平，而且极点位于无穷远处。

(1)让左右摄像机绕着各自的投影中心旋转 **R** 的一半，使得两摄像机的主光轴平行(但不一定垂直于两个投影中心的连线)，如图 3.5 所示

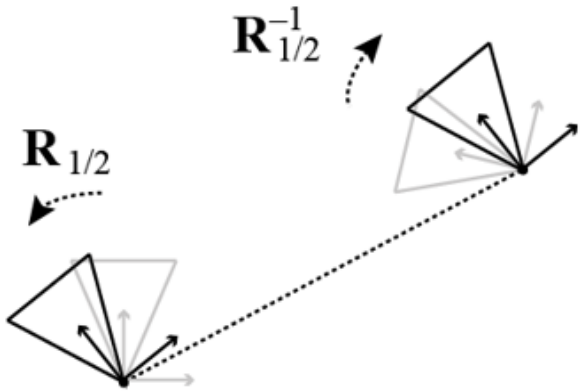


图 3.5 两个摄像机分别旋转一半

图中的 $\mathbf{R_{1/2}}$ 即旋转矩阵 **R** 的一半角度，即 $\mathbf{R_{1/2} * R_{1/2} = R}$，可以通过罗德格里斯(Rodrigues)变换求得。

(2)为了将主光轴平行的两个摄像机旋转到和投影中心连线垂直的位置，需要再做一次旋转变换，如图 3.6 所示：

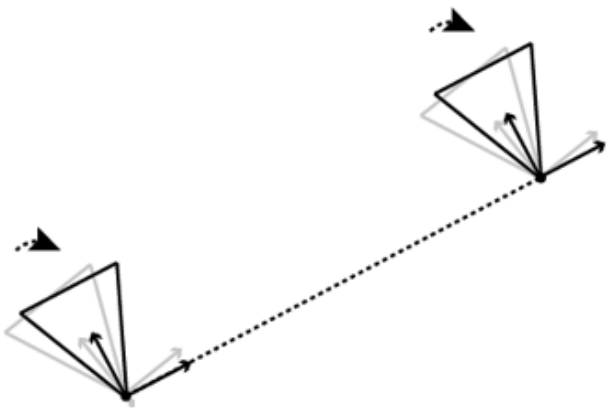


图 3.6 两个摄像机都旋转相同的角度

设这个旋转矩阵为 **Rr**，e_1,e_2,e_3分别为 Rr 矩阵的第一、二、三行向量。e_1为旋转之后的 x 坐标轴向量在当前坐标系中的向量表示，所以 e_1与两个摄像机投影点连线重合，即于 **T** 向量方向重合，T 向量经过(1)步骤的旋转，变为Tr:

$$Tr = R_{1/2}^{-1} \bullet T \tag{3-5}$$

则：

$$e_1 = \frac{Tr}{|Tr|} \tag{3-6}$$

e_2为旋转之后的 y 坐标轴向量在当前坐标系中的向量表示，由 3.6 图知 e_2的 z 坐标为 0，且 e_2和 e_1正交，故：

$$e_2 = \frac{\begin{bmatrix} e_{11} & e_{12} & 0 \end{bmatrix}}{\sqrt{e_{11}^2 + e_{12}^2}} \tag{3-7}$$

其中 e_{11}和 e_{12}为 e_1向量的 x 和 y 坐标。e_3向量和 e_1、e_2向量正交，通过叉积得到 e3:

$$e_3 = e_1 \times e_2 \tag{3-8}$$

通过(1)(2)两步得到左右摄像机的矫正旋转矩阵 $\mathbf{R_1}$ 和 $\mathbf{R_2}$:

$$R_1 = R_r \bullet R_{1/2}, R_2 = R_r \bullet R_{1/2}^{-1} \tag{3-9}$$

结合 3.1 节中的畸变矫正，以左摄像机为例，经过立体矫正后的图像中的像素(u,v)对应原图(未矫正图像)像素坐标为(u',v')，

由 2.1.2 节，对于(u,v)有：

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{3-10}$$

令：

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \bullet \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3-11}$$

则：

$$z_c \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{3-12}$$

同样对于(u',v')有：

$$z'_c \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} \tag{3-13}$$

矫正后的坐标为原坐标经过旋转矩阵 $\mathbf{R_1}$ 得到，即：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R_1 \bullet \begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} \tag{3-14}$$

则：

$$\frac{z'_c}{z_c}\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = R_1^{-1} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3-15}$$

整个计算流程为：

$$\begin{cases} x = \dfrac{u - u_0}{f_x} \\ \mathrm{y} = \dfrac{v - v_0}{f_y} \\ W\begin{bmatrix} x' & y' & 1 \end{bmatrix}^T = R_1^{-1} \bullet \begin{bmatrix} x & y & 1 \end{bmatrix}^T \\ \\ x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x' y' + p_2(r^2 + 2x'^2) \\ \mathrm{y}'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y'^2) + 2p_2 x' y' \\ u' = x'' f_x + u_0 \\ v' = y'' f_y + v_0 \end{cases} \tag{3-16}$$

得到 (u,v) 到 (u',v') 的映射之后，便能得到矫正之后的图像。采用 bouguet 进行立体矫正的结果如图 3.7 所示：

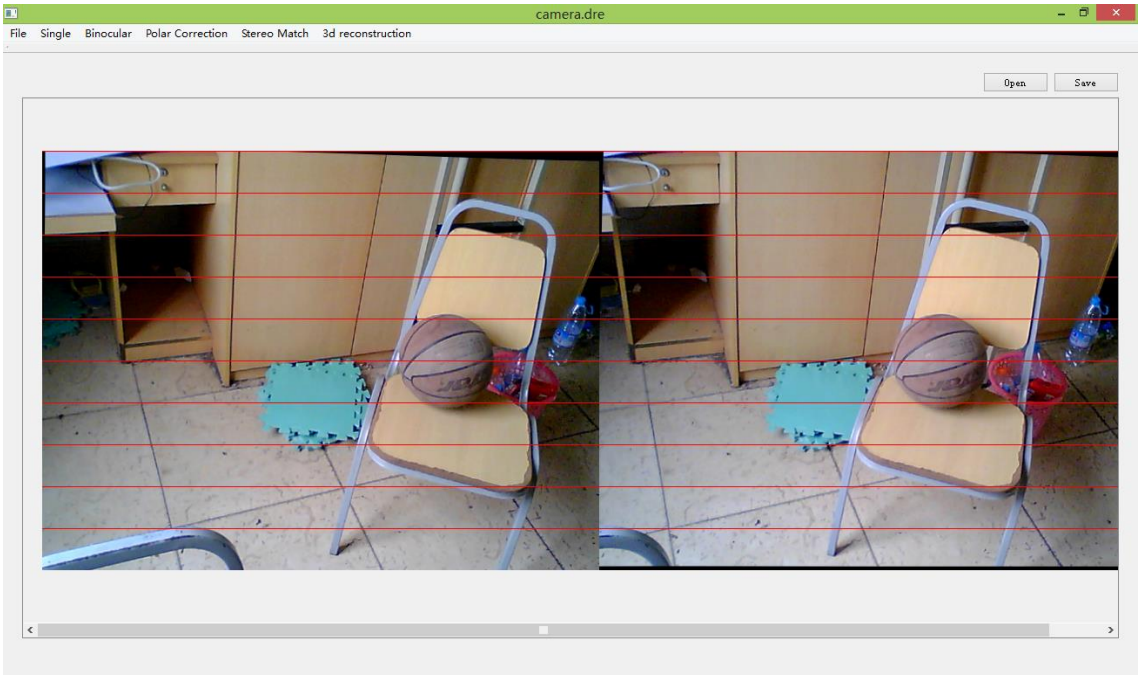


图 3.7 极线矫正结果

4 立体匹配

立体匹配是双目立体视觉中非常重要的一个环节，立体匹配的目标是找出左右图片(经过极线矫正处理)上的对应点，得到两张图片的视差图。通过视差图和标定结果可以确定这些对应点的三位空间坐标，换句话说，立体标定的结果直接影响着重建的结果。立体匹配算法可以分为基于特征的匹配、基于区域的匹配、基于相位的匹配算法。基于区域的匹配算法由于实现简单、能得到稠密的视差图，适合用于场景的重建，本文着重研究基于区域的立体匹配算法。

4.1 立体匹配约束

立体匹配本身为一个病态问题，为了降低匹配的错误率，立体匹配中常常使用如下一些约束条件[34]：

(1)极线约束 通过极线矫正，同一点在左右摄像头获得的图片中位于同一行，因此当参考图像和目标图像为矫正过的图片时，参考图像上的像素点与目标图像像素点位于同一行上。

(2)唯一性约束 对于大部分实际的图片，参考图像上每一个像素最多(也可能没有匹配,为遮挡点)和目标图像上的某一个像素匹配。

(3)相似性约束 对于参考图像和目标图像上对应点应该具有相似的属性(如灰度、灰度变化梯度等)，但是实际中，这一假设并不始终成立：由于左右摄像机所处的位置、自身的差异，使得两个摄像机拍摄的同一点可能属性并不相同(光照影响、遮挡影响等)。

(4)顺序性约束 对于参考图像和目标图像同一行上的匹配像素点在参考图像和目标图像中的相对顺序是相同的。

(5)平滑性约束 假设物体的表面通常是平滑的，所以除了在物体边界处之外的地方视差值变化应该较小。

(6)左右一致性约束 以左图为参考图像，右图为目标图像和以右图为参考图像左图为目标图像，得到的匹配点对应该是一致的。

4.2 固定大小窗口匹配算法

4.2.1 基本原理

固定窗口(Fixed Window)匹配算法实现的基本原理为：计算参考图像中的每一个像素点和目标图像中同一行的每一个像素点的 SAD(或 SSD 等)值，取值最小的像素点作为匹配点，如图 4.1 所示：

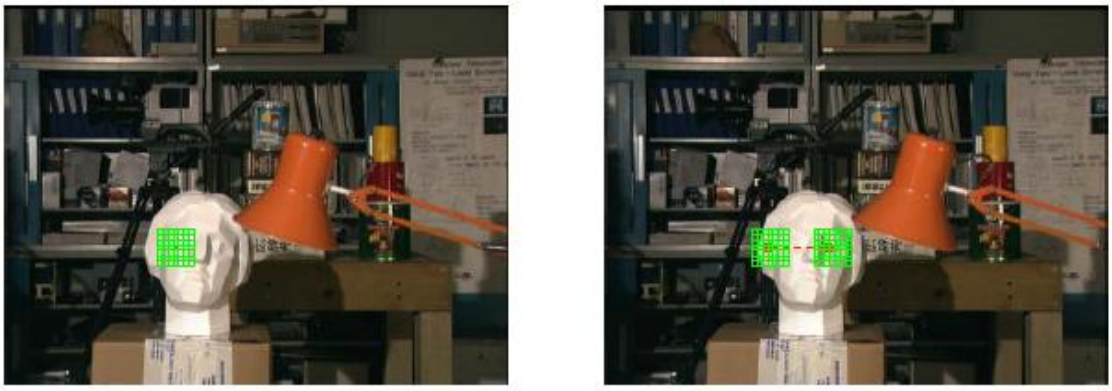


图 4.1 FW 算法

对于参考图像中的一点 (x, y)， $SAD(x,y,d)(\text{mind} < \text{d} < \text{maxd})$ 匹配代价函数定义如下：

$$SAD(x,y,d) = \sum_{i,j=-n}^{n} \left| L(x+j, y+i) - R(x+d+j, y+i) \right| \tag{4-1}$$

“固定窗口”在这里指的是以(x,y)点为中心，边长为 2n+1 的对成正方形窗口。采用 3*3 7*7 11*11 15*15 窗口的实验结果如图 4.2 所示：

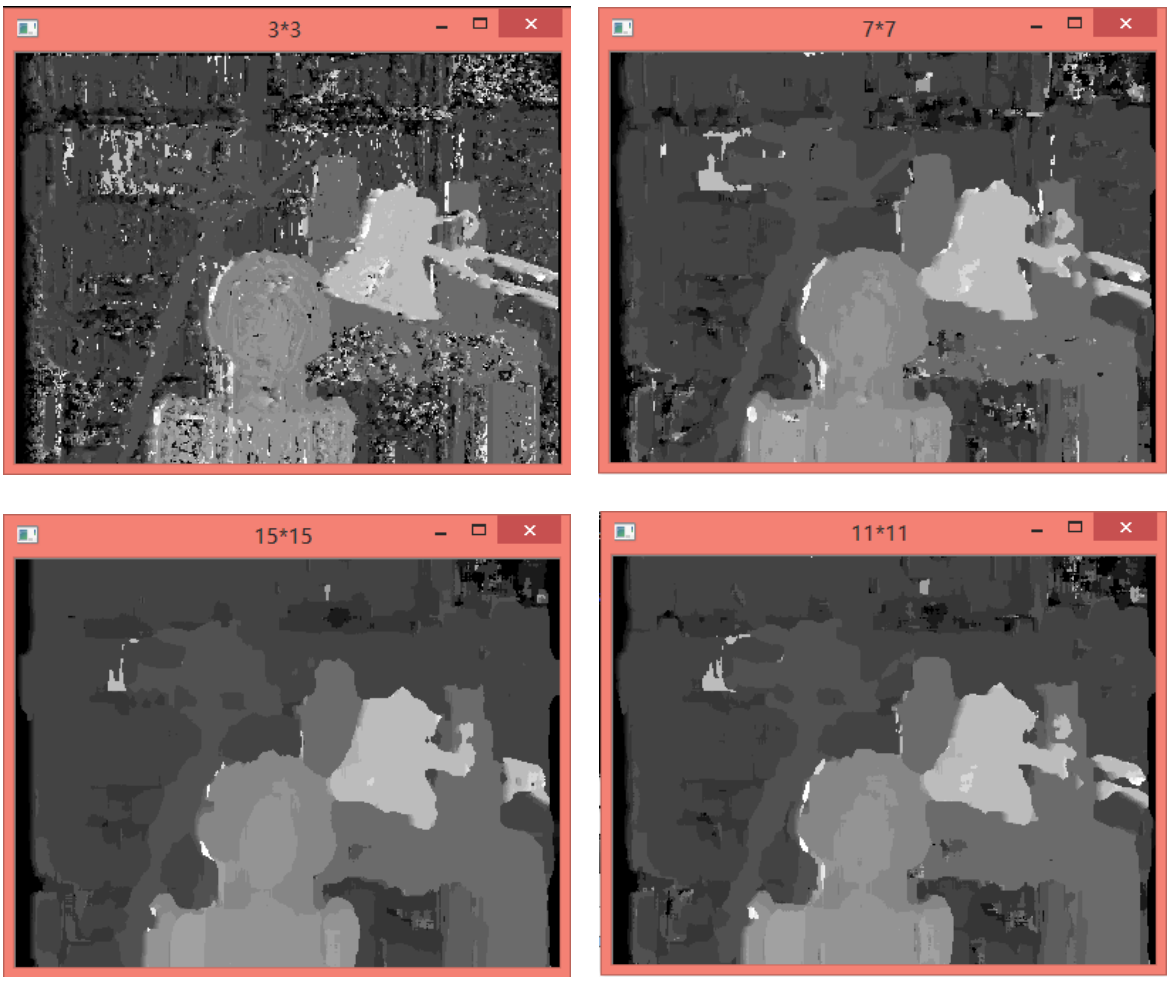


图 4.2 采用不同的窗口得到的视差图(0<d<20)

4.2.2 Box-filtering 加速算法

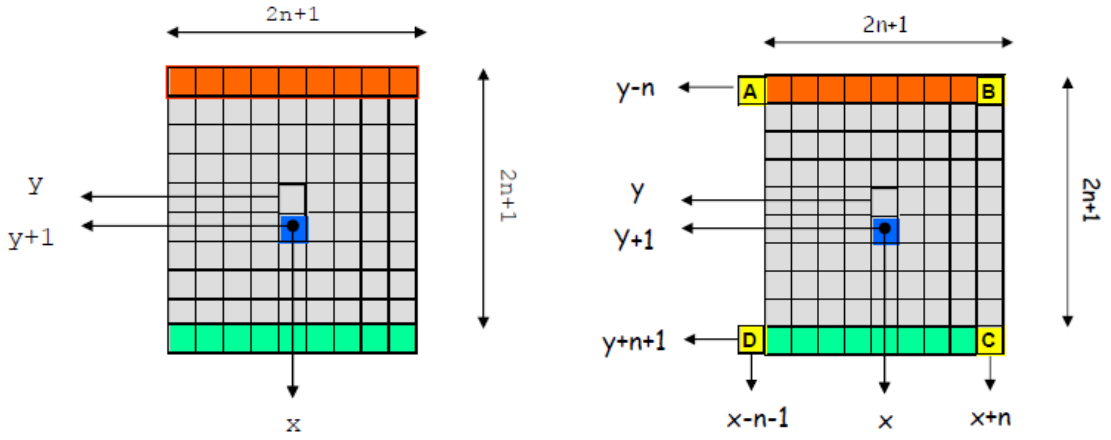
若对参考图像中每一个像素和目标图像中同一行的每一个像素分别计算邻域内绝对值差和，窗口大小直接影响计算时间，以 tsukuba(384*288)测试图像为例，不同窗口计算时间表如表 4.1 所示：

窗口大小	1*1	3*3	5*5	7*7	9*9	11*11	13*13	15*15	17*17
计算时间 ms	34	178	449	837	1358	1988	2929	3804	4881

表 4.1 不同的窗口计算花费时间表

M.Mc Donnel 在 1981 年提出 Box-filtering 加速 FW 算法[35]，通过减少 SAD 的冗余计算加快处理过程：

对对于相邻两列 SAD 值相差两行 SAD 窗口：



(a)SAD 函数移动示意图 **(b)S 函数移动示意图**

图 4.3 Box-filtering 算法示意图

SAD 函数具有递推关系：

$$SAD(x,y,d)=SAD(x,y-1,d)-S(x,y-1-n,d)+S(x,y+n,d) \tag{4-2}$$

S 函数具有递推关系：

$$S(x,y+n,d)=S(x-1,y,d)-I(x-n-1,y,d)+I(x+n,y,d) \tag{4-3}$$

其中:

$$I(x,y,d)=\left|I_L(x,y)-I_R(x-d,y)\right| \tag{4-4}$$

Box-filtering 算法在窗口移动的过程中，只需要多计算 4 个像素点的绝对差值，无论窗口大小，Box-filtering 算法采用不同的 SAD 窗口所需时间如表 4.2 所示：

窗口大小	1*1	3*3	5*5	7*7	9*9	11*11	13*13	15*15	17*17
计算时间 ms	48	48	51	49	53	57	56	59	60

表 4.2 Box-filtering 算法不同的窗口计算花费时间表

4.3 自适应权值（AW）匹配算法

4.3.1 基本思想

固定窗口(FW)算法是将固定窗口内像素点的匹配代价聚合起来，而实际上，与中心像素相似的像素贡献代价应该相对多一些，而与中心像素点差异较大的像素应该相对少一些，尤其是在物体边缘处，如图 4.3 所示：

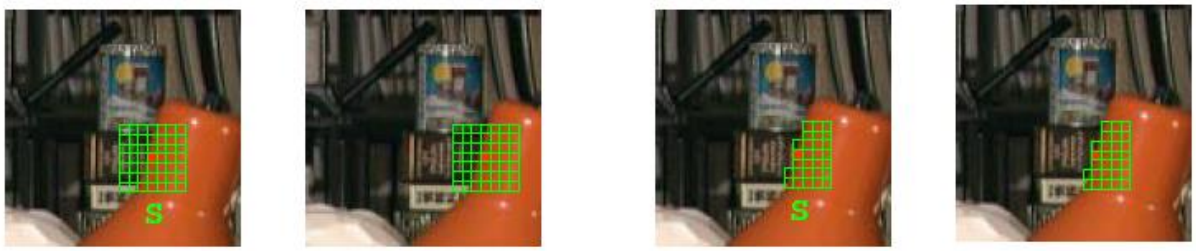


图 4.3 固定窗口算法(左) 理想的匹配窗口(右)

K.Yoon[20]等在 2006 年提出自适应权重的立体匹配算法，该算法采用窗口大小固定，但左右窗口内各个像素匹配代价的权值由和中心像素的距离以及相似程度决定(同双边滤波算法相似)，窗口内各个像素的权值为：

$$w(p,q)=k\exp(-\frac{\Delta c_{pq}}{\gamma_c}-\frac{\Delta g_{pq}}{\gamma_p}) \tag{4-5}$$

其中 p 为窗口中心像素，q 为窗口内的某一个像素。Δg_{pq} 为 pq 两点的像素距离，Δc_{pq} 为 pq 两点在色彩空间的距离，本文使用 CIELab 颜色空间：

$$\Delta c_{pq}=\sqrt{(L_p-L_q)^2+(a_p-a_q)^2+(b_p-b_q)^2} \tag{4-6}$$

采用自适应权重算法窗口内的权值分布如图 4.4 所示：

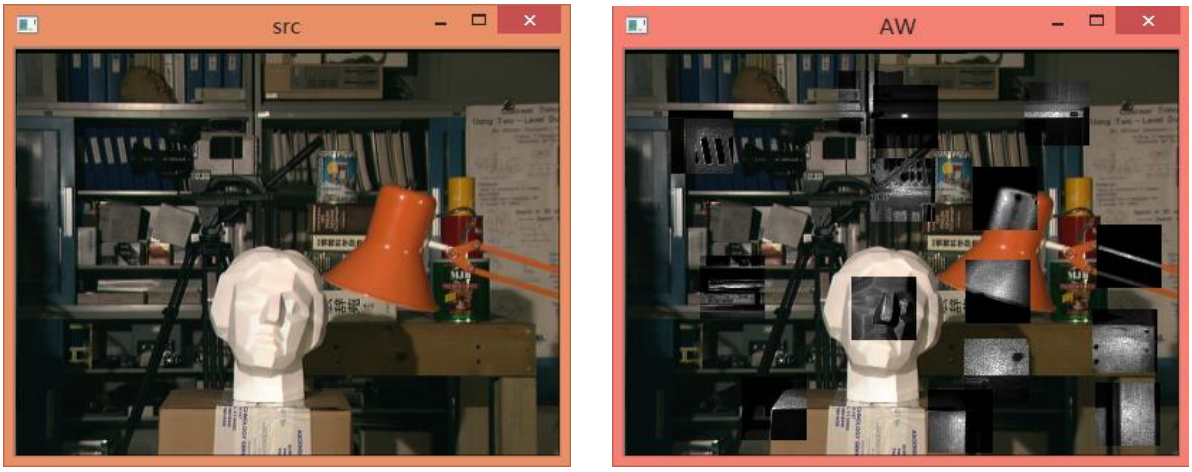


图 4.4 左为原图 右为若干个自适应 41*41 窗口 越亮代表权值越大

自适应权值算法参考图像上 p 点和目标图像上 $\bar{p}_d$ 点的匹配代价为：

$$E(p,\bar{p}_d)=\frac{\sum_{q\in N_p,\bar{q}_d\in N_{\bar{p}d}} w(p,q)w(\bar{p}_d,\bar{q}_d)e_0(q,\bar{q}_d)}{\sum_{q\in N_p,\bar{q}_d\in N_{\bar{p}d}} w(p,q)w(\bar{p}_d,\bar{q}_d)} \tag{4-7}$$

其中 N_p 为 p 点的邻域, $N_{\bar{p}d}$ 为 $\bar{p}_d$ 点的邻域， $e_0(q,\bar{q}_d)$ 为像素间差异度量函数，本文使用绝对差值(AD)：

$$e_0(q,\bar{q}_d)=\sum_{c\in\{r,g,b\}}\left|I_c(p)-I_c(\bar{q}_d)\right| \tag{4-8}$$

即将 RGB 三个通道的绝对值相加。

得到不同视差值的匹配代价后，采用 WTA(Winner-Takes-All)算法获取视差值，即取最小匹配代价对应的视差值：

$$d_p=\arg\min_{d\in S_d}E(p,\bar{p}_d) \tag{4-9}$$

采用自适应权值算法用不同窗口对 tsukuba 测试图片进行测试，测试结果如下：

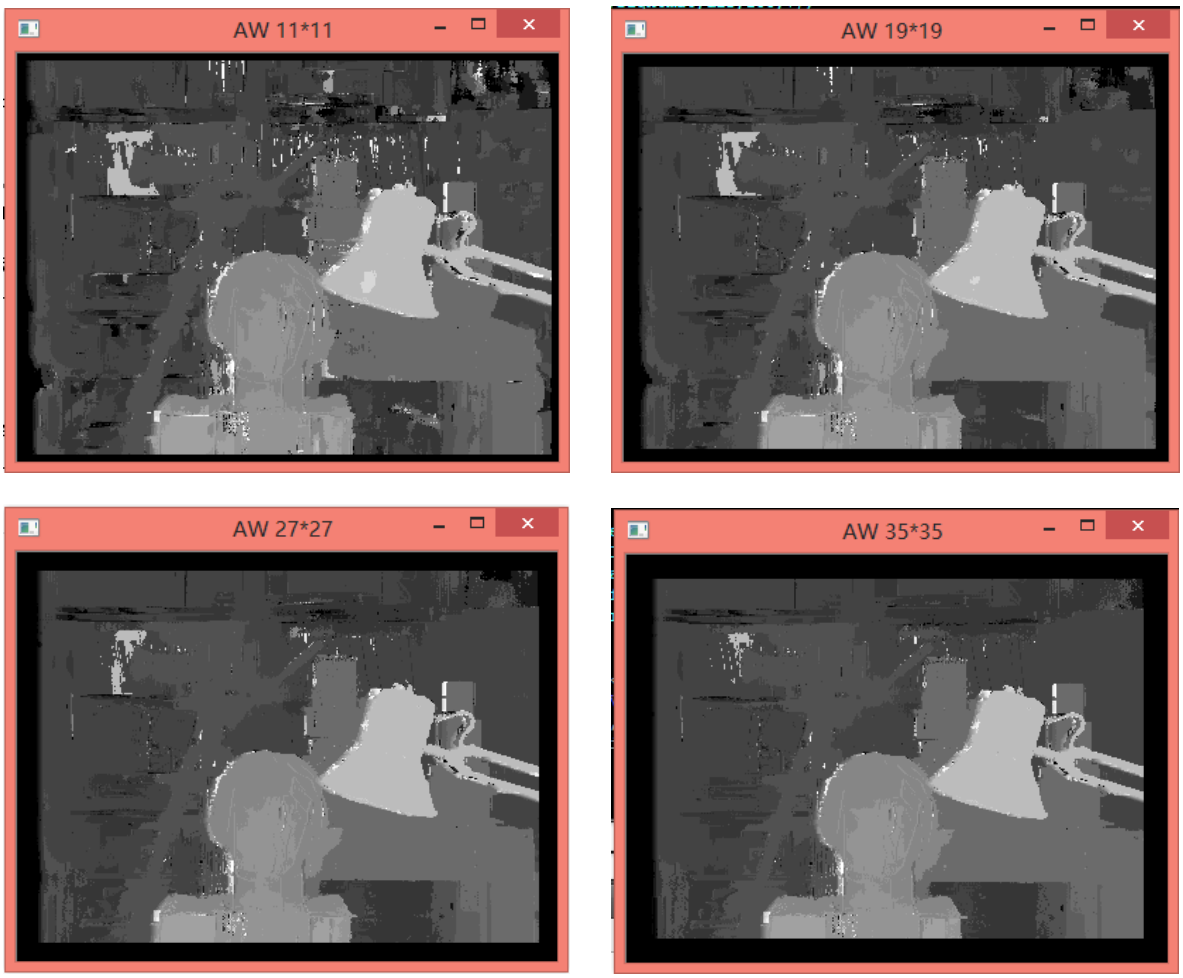


图 4.5 不同窗口大小的自适应权值算法(γ_c=7 γ_g=36)

4.3.2 快速双边滤波(FBS)算法

S. Mattoccia[37]等人于 2009 年提出快速双边滤波(FBS)算法，同 AW 算法类似，FBS 算法也采用类似双边滤波算法为窗口内的像素代价赋予权值，与 AW 算法不同的是，AW 算法为窗口内每一个像素赋予不同的权值，而 FBS 算法则将窗口分为若干个小窗口，小窗口内各个像素的权值相同，由小窗口内像素的平均值与中心像素的色彩空间距离和像素距离计算得到(当小窗口尺寸为 1*1 时,FBS 算法和 AW 算法等价),如图 4.6 所示。

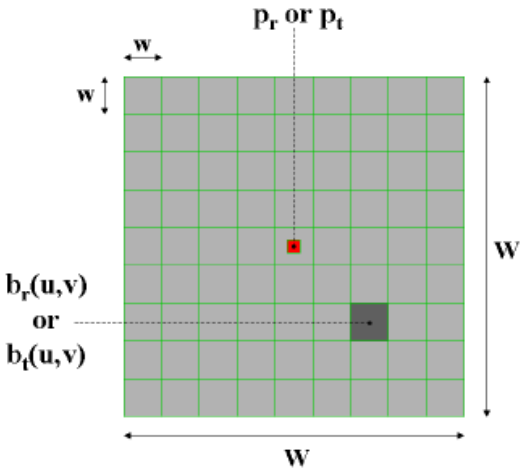


图 4.6 FBS 算法将窗口 W 分为若干个小窗口 w

若(u,v)为窗口内某一个小窗口的中心像素，则该小窗口内所有像素色彩权值均为：

$$W_c(I_r(p_r),\bar{I}_r(b_r(u,v))) = \exp(-\frac{\left\|I_r(p_r)-\bar{I}_r(b_r(u,v))\right\|}{\gamma_c}) \tag{4-10}$$

其中 $\bar{I}_r(b_r(u,v))$ 为 (u,v) 所在小窗口所有像素点的平均像素值。

采用窗口 W 为 45*45，w 为 3*3 的 FBS 算法窗口和 45*45 的 AW 算法窗口内权值分布对比如图 4.7 所示：

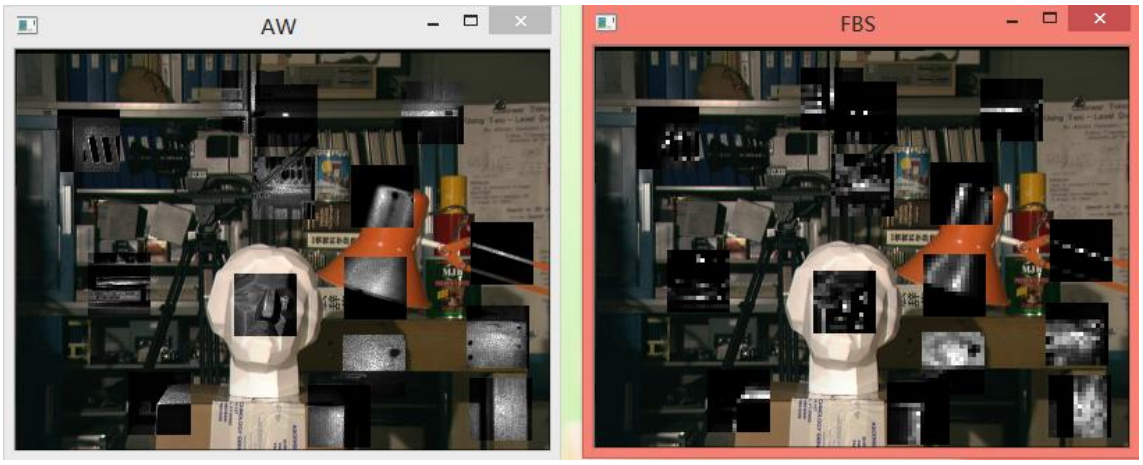


图 4.7 左为 AW 算法 右为 FBS 算法

BFS 相对于 AW 算法而言，具有如下优势:

(1) 由于使用了邻域均值处理，使得噪声对 BFS 算法影响比 AW 算法小

(2) 小窗口内均值、SAD 值可以采用 Box-Filtering 算法加速，小窗口越大，加速越明显，但是效果更差

对 21*21 窗口，分别采用 3*3 和 7*7 小窗口对 BFS 算法进行测试，测试结果如图 4.8 所示:

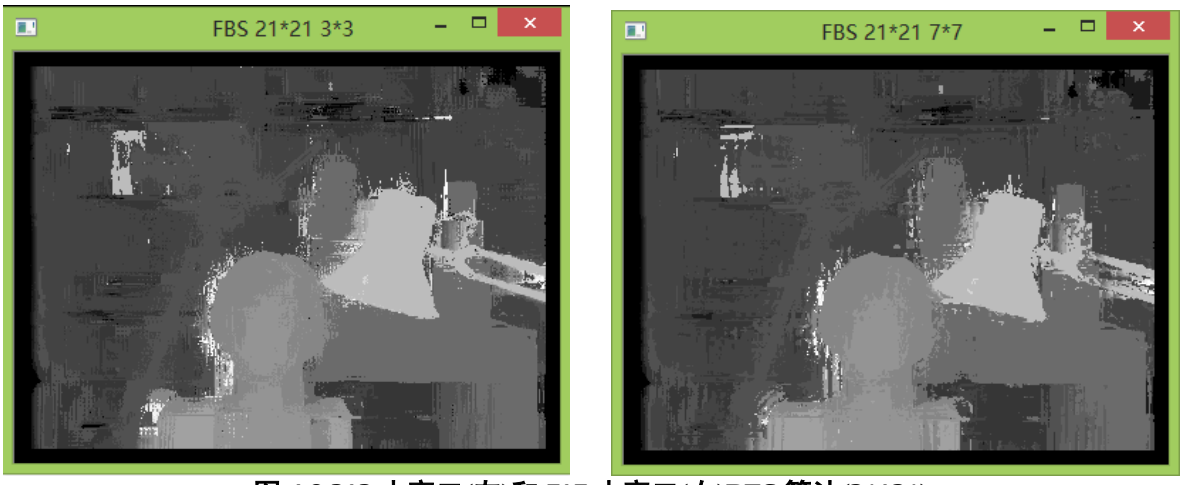


图 4.8 3*3 小窗口(左)和 7*7 小窗口(右)BFS 算法(21*21)

对 27*27 窗口，分别采用 3*3 和 9*9 小窗口对 BFS 算法进行测试，测试结果如图 4.9 所示：

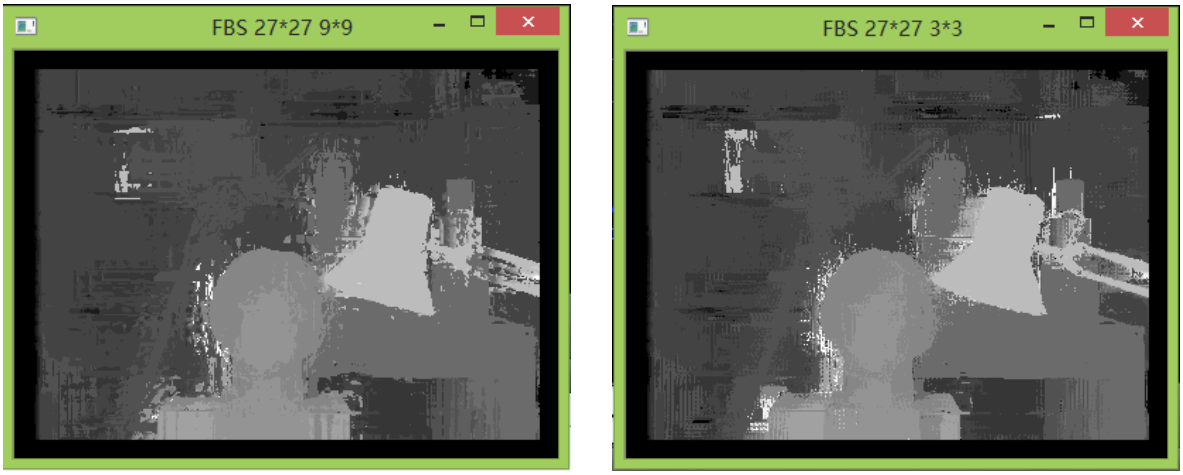


图 4.9 3*3 小窗口(左)和 9*9 小窗口(右)BFS 算法(27*27)

3*3 小窗口的 FBS 算法在实际测试中效果最好，但花费的时间最多。采用不同的窗口尺寸分别对 AW 算法和 FBS 算法处理时间进行测试，测试结果如表 4.3 所示：

窗口尺寸	9*9	15*15	21*21	27*27	33*33	39*39
AW 算法耗时(ms)	8260	24842	42256	77276	91554	132792
3*3 FBS 耗时(ms)	1046	2401	4337	7439	10729	13518

表 4.3 AW 算法和 FBS 算法耗时对比

4.4 动态规划(DP)匹配算法

4.4.1 动态规划原理

动态规划算法将图片按行扫描，对每一行寻求最优匹配，即使行代价函数 **E** 取得最小值：

$$E = E_{match} + E_{skip} \tag{4-11}$$

其中 E_{match} 为匹配代价 E_{skip} 为遮挡代价。

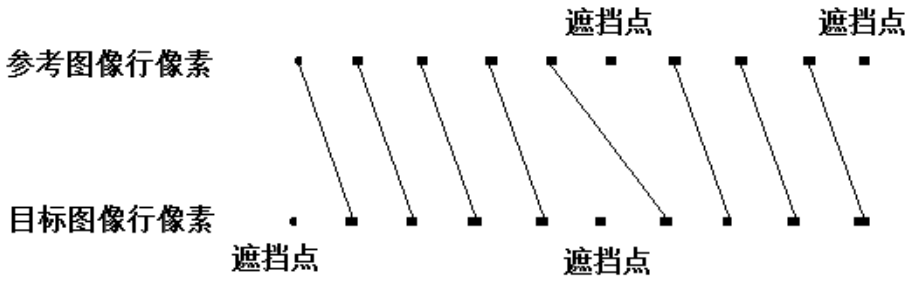


图 4.10 动态规划 匹配点和遮挡点

如图 4.10 所示，E_{match} 定义为参考图像行和目标图像行上所有匹配点对的匹配代价和：

$$E_{match} = \sum Cost(i, j) \tag{4-12}$$

其中(i,j)为两行像素的匹配点对，$Cost(i, j)$ 为 i 点和 j 点的匹配代价。E_{skip} 为所有遮挡点 i 的遮挡代价和：

$$E_{skip} = \sum Pen(i) \tag{4-13}$$

其中 i 为两行像素中的遮挡点，根据立体匹配的顺序性约束(匹配点对之间没有交叉)，得到状态转移方程：

$$\begin{cases} E(i, j) = \min(A, B, C) \\ A = E(i-1, j-1) + Cost(i, j) \\ B = E(i-1, j) + Pen(i) \\ C = E(i, j-1) + Pen(j) \end{cases} \tag{4-14}$$

A 情况代表像素点 i 和像素点 j 两点匹配，对应地，i 的视差值为 *i-j*；**B** 情况代表像素点 i 为遮挡点，**C** 情况代表像素点 j 为遮挡点。

4.4.2 采用单个像素代价作为匹配代价

若采用单个像素代价作为匹配代价，本小节采用代价函数 Cost(i,j)为：

$$Cost(i, j) = \sum_{c \in \{r,g,b\}} \left| I_{lc}(i) - I_{rc}(j) \right| \tag{4-15}$$

即三个通道绝对差值之和，本文采用的遮挡代价函数为常数函数，即：

$$Pen(i) = P \tag{4-16}$$

其中 **P** 为匹配参数。

分别采用不同的 P 值(遮挡代价)对 DP 算法进行测试，测试结果如图 4.11 所示：

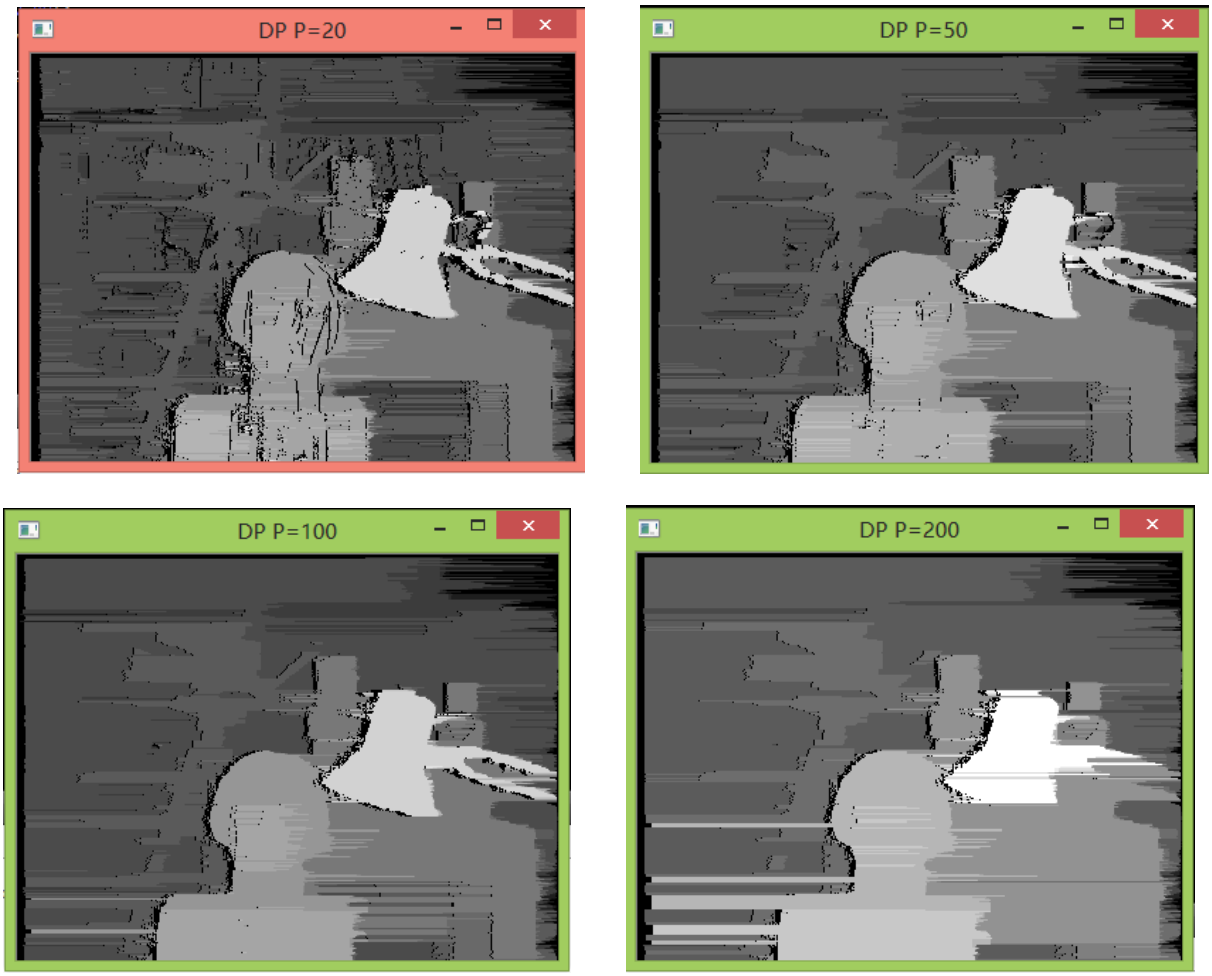


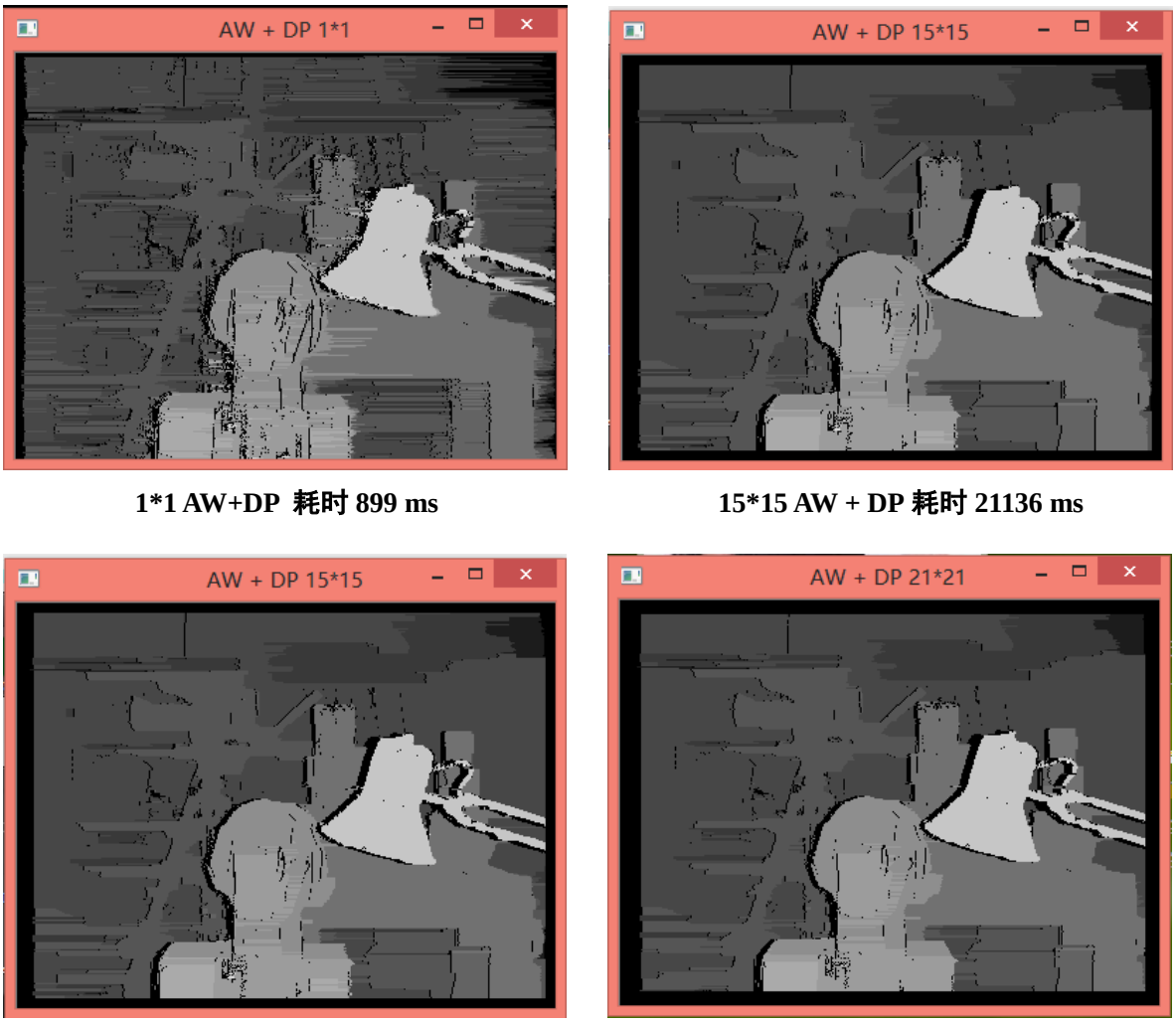
图 4.11 动态规划算法不同 P 值测试结果 执行时间 610ms

由于动态规划算法按行扫描进行最优化匹配，只考虑了单行像素之间的顺序约束，并没有考虑行间像素视差值得平滑度，所以相邻行间像素的视差值差异可能会很大，使得视差图有明显的条纹效应。随着遮挡代价的增加，遮挡点变少，而条纹效应愈加明显；而随着遮挡代价缩小，遮挡点变多，条纹效应减弱。

4.4.3 AW+DP 算法

由于 DP 算法本身并没有考虑不同像素行间的约束，而通过匹配代价聚合可以在一定程度上融入行间约束。本小节对 AW+DP 算法进行测试，即采用自适应权值(AW)算法进行匹配代价聚合，代价聚合函数为(4-7)。

其中，各个函数的配置同 4.3.1 节，采用不同的 AW 窗口的 DP 算法实验结果如图 4.22 所示(P=20)：



1*1 AW+DP 耗时 899 ms **15*15 AW + DP 耗时 21136 ms**

21*21 AW+DP 耗时 43861 ms **33*33 AW+DP 耗时 99782 ms**

图 4.12 采用不同的 AW 窗口的 DP 算法视差图

注意到 1*1 窗口的 AW 算法退化为单个像素的匹配代价，故效果和图 4.11 第一幅图效果一样。随着窗口的增大，DP 算法的条纹效应得到减弱，而时间开销增大。从实验结果可以看出，由于大尺度窗口的 AW 算法并不影响视差图轮廓，使得 AW+DP 算法极大地保留了物体的边缘信息。

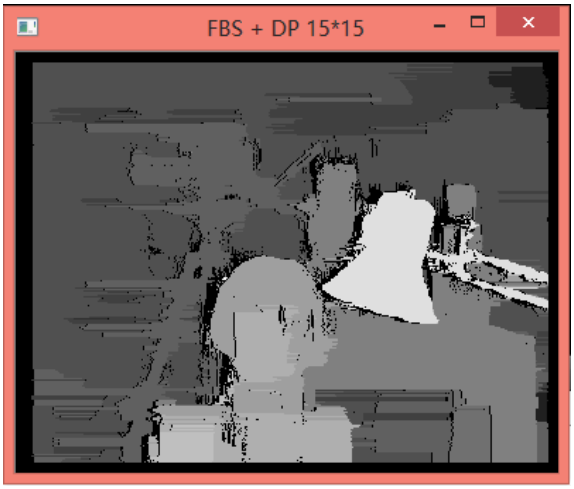
4.4.4 FBS+DP

AW+DP 算法由于 AW 算法本身计算代价较大使得算法整体的时间花销非常高，在 4.3 节中，采用子窗口的快速双边滤波(FBS)算法能有效地提高算法的处理速度，3*3 子窗口的 FBS 算法处理速度是相同大小窗口的 AW 算法处理速度的 9 倍左右(FBS 子窗口内代价采用了 Box-filter 加速计算)，5*5 子窗口的 FBS 算法处理速度则是 AW 算法的 25 倍左右，随着子窗口的增大，FBS 处理速度加快，但是相应地视差图的准确度下降，在实验中，采用 3*3 子窗口的 FBS 算法效果最佳。

在本小节中，采用 3*3 子窗口 FBS 算法，P=20*9,(第一幅图采用 1*1，P=20)计算匹配代价(其余配置同 4.3.2 节)，再采用动态规划获取视差值，不同窗口大小的 FBS 算法的测试

结果如下：

1*1 FBS+DP 耗时 998 ms



15*15 FBS + DP 耗时 3096 ms

21*21 FBS+DP 耗时 4809 ms

33*33 FBS + DP 耗时 12526 ms

图 4.13 不同窗口的 FBS+DP 算法实验结果

同 AW+DP 算法的结果类似，1*1 窗口的 BFS 算法退化为单个像素的代价匹配；随着窗口的增大，DP 算法的条纹效应减弱(台灯边缘误差主要由于 FBS 算法误差造成，参见 4.8 和 4.9)，但时间开销增大，33*33 的 FBS+DP 算法耗时 12s，比 AW+DP 99s 快了 9 倍左右，匹配效果比 AW+DP 算法稍逊。

4.5 半全局匹配(SGM)算法

4.5.1 全局匹配算法

上述的 FW，AW，FBS 算法在匹配代价聚合环节之后，仅仅是通过 WTA(Winner Takes All)算法得到每个像素点的视差值，并没有考虑该像素点周围其他像素的视差值之间的平滑约束，因此局部匹配算法产生的视差图视差并不平滑；而全局匹配算法则将平滑性约束考虑到数学模型当中。

局部匹配算法通常使用 WTA 算法，可以使用如下模型表示：

$$D(i,j)=\arg\min Cost(i,j,d)(\text{mind}<\text{d}<\text{maxd}) \tag{4-17}$$

其中 D(i,j)表示(i,j)像素点的视差值，Cost(i,j,d)为(i,j)点和(i,j-d)点的匹配代价，从这个模型也可以看出，各个像素点(i,j)的视差值 D(i,j)与其他点的视差值无关，因此局部算法另外一个优势是并行化计算实现较为简单。全局匹配算法的模型表示为：

$$E(\text{d})=E_{\text{data}}(\text{d})+\lambda E_{smooth}(\text{d}) \tag{4-18}$$

其中 d 表示视差函数，全局立体匹配算法将令 $E(d)$ 取得最小值的 d 函数作为最终的视差函数。其中 $E_{\text{data}}(\text{d})$ 表示所有像素点的匹配代价：

$$E_{\text{data}}(\text{d})=\sum_{i,j}Cost(i,j,d(i,j)) \tag{4-19}$$

$E_{smooth}(\text{d})$ 表示所有像素点的平滑代价，若只考虑 4 邻域平滑度，可表示为：

$$\begin{aligned}E_{smooth}(\text{d})=&\sum_{i,j}Smo(d(i,j)-d(i,j+1))+Smo(d(i,j)-d(i,j+1))\\&+Smo(d(i,j)-d(i,j+1))+Smo(d(i,j)-d(i,j+1))\end{aligned} \tag{4-20}$$

其中 *Smo* 为不同的平滑模型函数。

4.5.2 单扫描线算法

在全局匹配算法中，能量函数 **E(d)**中变量 d 可以看作一个 w*h(w 和 h 为图片的宽和高)维的向量，对于 100*100 的小图片而言，d 的维度为 100*100=10000，若采用穷举的方式，则有$(\text{maxd-mind})^{10000}$种可能，采用传统的能量最小化算法(如模拟退火算法),则可能会由于收敛速度慢使得算法耗时太长。

在全局匹配算法中，若平滑函数只考虑与之临近的一个像素,即：

$$E_{smooth}(\text{d})=\sum_{i,j}Smo(d(i,j)-d(i-i_r,j-j_r)) \tag{4-21}$$

其中 (j_r,i_r) 为扫描方向：(1,0)为自左向右扫描；(0,1)为自上向下扫描；(-1,0)为自右向左扫描，(0,-1)为自下往上扫描等。本小节采用的平滑模型函数为：

$$Smo(x)=\begin{cases}0 & |x|=0\\ P1 & |x|=1\\ P2 & |x|\geq 2\end{cases} \tag{4-22}$$

则全局函数可以通过该方向扫描进行动态规划得到最优解:

$$\begin{cases}E(i,j,d)=Cost(i,j,d)+\min(A,B,C)\\ A=E(i-ir,j-j_r,d)\\ B=\min(E(i-ir,j-j_r,d-1),E(i-ir,j-j_r,d+1))+\text{P1}\\ C=\min(E(i-ir,j-j_r,d-k),E(i-ir,j-j_r,d+k))+\text{P2}\ \ (\text{k}>1)\end{cases} \tag{4-23}$$

通常 P2〉P1，所以上式可以优化为：

$$C=\min(E(i-ir,j-j_r,k))+\text{P2}\ \ (\text{mind}<\text{k}<\text{maxd}) \tag{4-24}$$

从而避免了 C 式中对每一个不同 d 值遍历不同 k 值的花销(对所有的 d 值，只需提前计算一次。

本小节中使用三个通道的绝对差值作为匹配代价，即：

$$Cost(i,j,d)=\sum_{c\in\{r,g,b\}}\left|I_{lc}(i,j)-I_{rc}(i,j-d)\right| \tag{4-25}$$

分别采用向右→、右下↘、向下↓、左下↙四个扫描方向进行测试，实验结果如图 4.14 所示(P1=10,P2=40)：

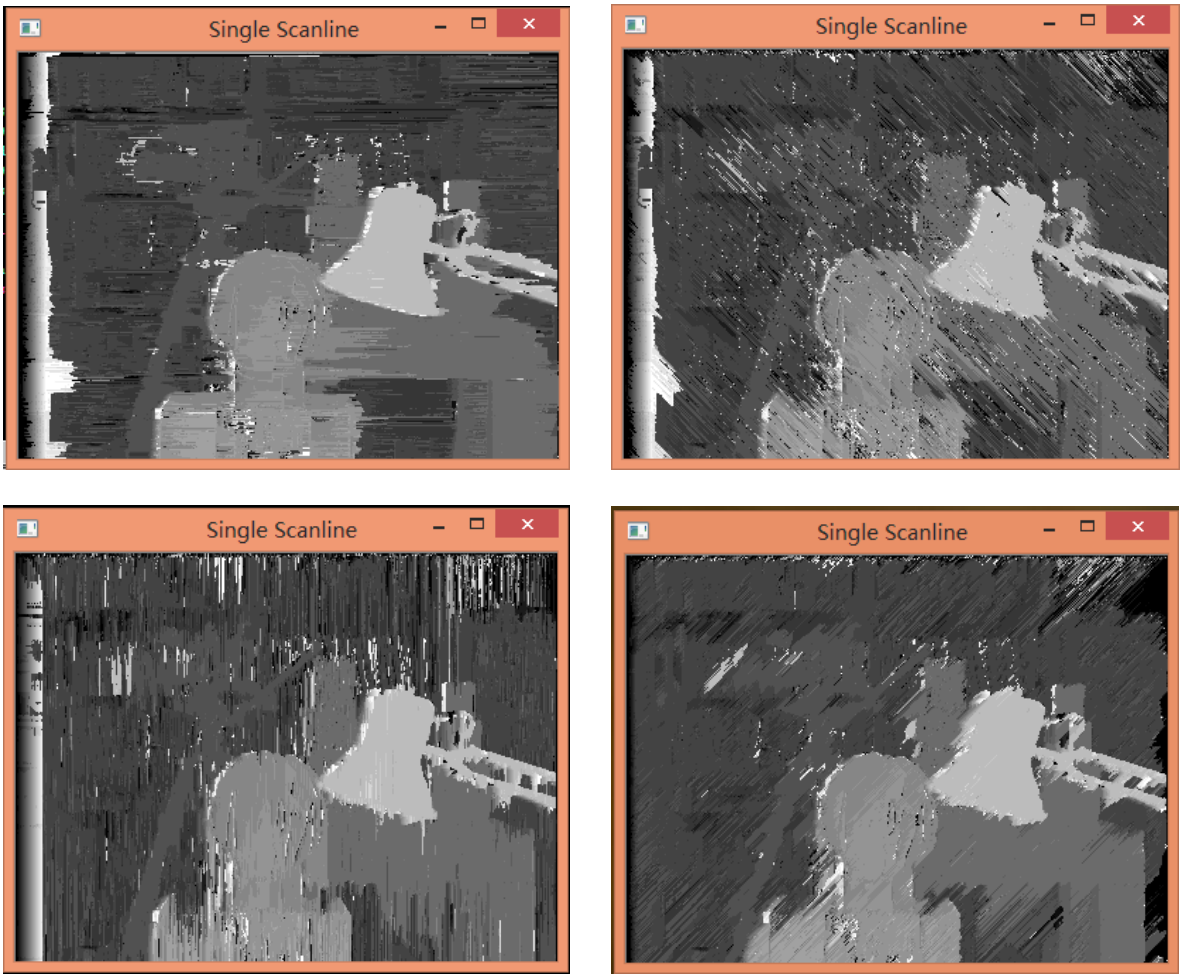


图 4.14 分别采用向右→、右下↘、向下↓、左下↙四个扫描方向结果 耗时 115ms

4.5.3 多扫描线(半全局 SGM)算法

从 4.14 实验结果可以看出，采用单一扫描线的算法和图 4.11 动态规划算法结果类似，均出现了明显的视差“条纹”，为了解决这一问题，H. Hirschmuller[38]于 2005 年提出一种半全局的立体匹配算法，该算法的核心思想就是通过从若干个方向进行扫描(4 个到 16 个)，取这几个方向能量函数之和最小的 d 为视差值，即：

$$D(i,j)=\arg\min\sum E_r(i,j,d) \tag{4-26}$$

其中 **Er(i,j,d)**表示像素点在不同的扫描方向上的能量值，考虑到 **Er** 本身会随着扫描线方向不断增大，为避免溢出，采用:

$$\begin{cases} E(i,j,d)=Cost(i,j,d)+\min(A,B,C)-\min(E(i-ir,j-j_r,k)) \ \ (\text{mind}<\text{k}<\text{maxd}) \\ A=E(i-ir,j-j_r,d) \\ B=\min(E(i-ir,j-j_r,d-1),E(i-ir,j-j_r,d+1))+\text{P1} \\ C=\min(E(i-ir,j-j_r,k))+\text{P2} \ \ (\text{mind}<\text{k}<\text{maxd}) \end{cases} \tag{4-27}$$

本小节使用和 4.5.2 小节相同的代价函数，使用 8 个方向(右→、左←、右下↘、左上↖、向下↓、向上↑、左下↙、右上↗)，用不同的 P1 和 P2 参数对 SGM 算法进行测试，测试结果如图 4.15 所示：

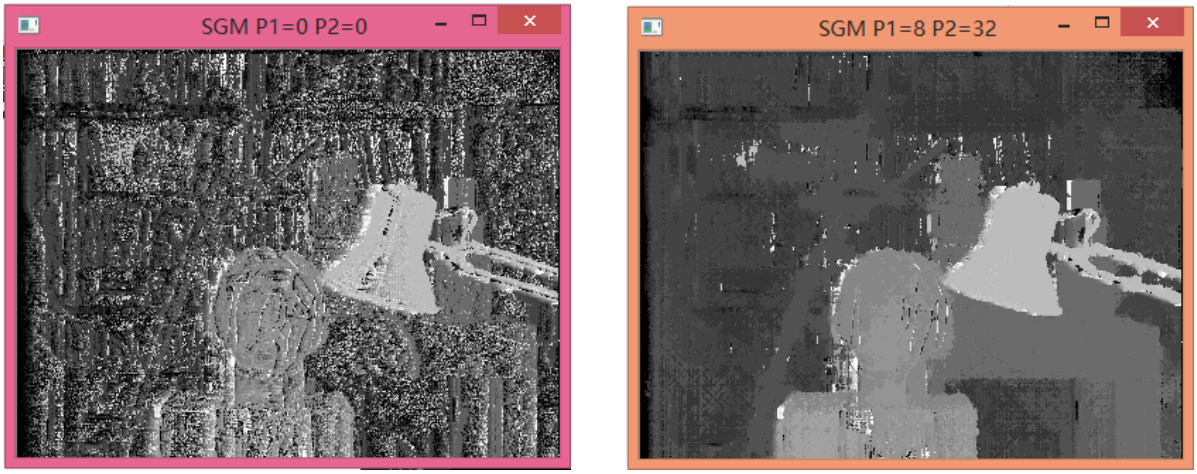


图 4.15 SGM 算法测试结果 左图 P1=P2=0 右图 P1=8 P2=32 耗时 692ms

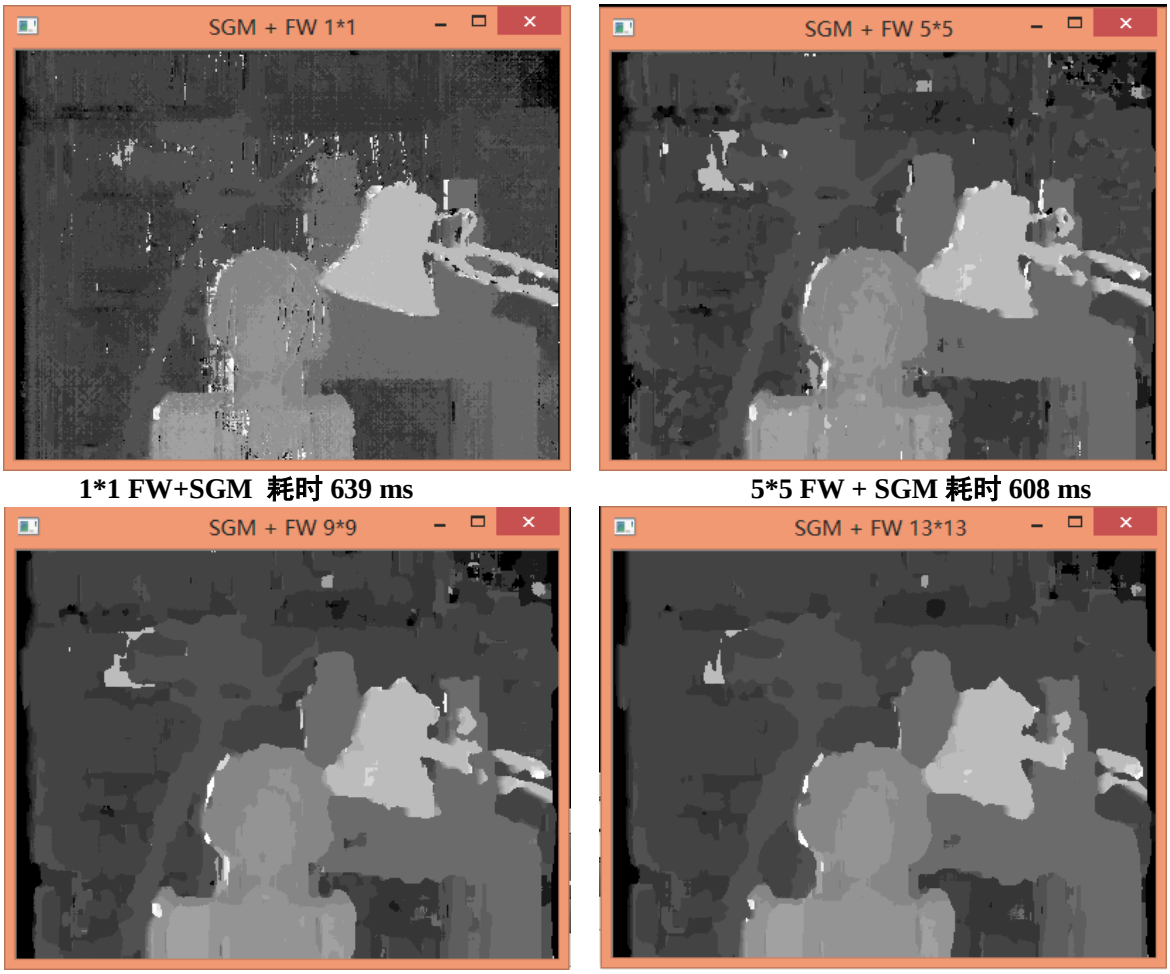
从图中可以看出，当 P1=P2=0 时，平滑约束为 0，SGM 算法退化为局部算法(窗口为 1*1)，使得视差图非常粗糙(物体边缘轮廓明显)，而当加入了平滑约束 P1=8 P2=32 时，SGM 算法得到的视差图则要明显平滑(边缘明显)于左图。

此外，由于采用了 8 个方向进行扫描，SGM 算法克服了单扫描线匹配算法的条纹效应，在时间开销上大约为单扫描线算法的 8 倍(8 个方向需要分别计算)，另外由于采用 8 个方向扫描需要储存整张图片大量的数据，若实际计算资源不足时，也可以采用 4 个方向扫描，只需要保存两行数据即可。

4.5.4 SGM+FW 算法

本小节对 SGM+FW 算法进行测试，即使用固定窗口(FW)算法聚合匹配代价，即 $Cost(i,j,d)=SAD(i,j,d)$，其中 SAD 函数定义为(4-1)：

由于 FW 算法可以通过 Box-Filtering 算法加速，使得 SGM 算法和 SGM+FW 算法时间消耗基本相同，采用不同尺寸 SAD 窗口的 SGM 算法实验结果如图 4.16 所示：



1*1 FW+SGM 耗时 639 ms **5*5 FW + SGM 耗时 608 ms**

9*9 FW+SGM 耗时 639 ms **13*13 FW + SGM 耗时 608 ms**

图 4.16 不同窗口的 FW+SGM 算法实验结果

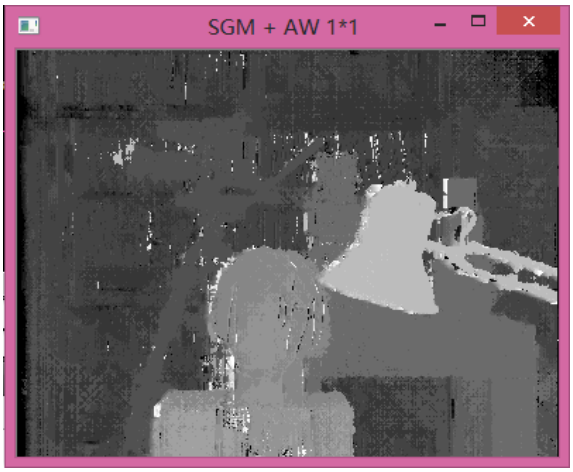
当 SAD 窗口为 1*1 时，算法退化为单个像素代价的 SGM 算法，从图中可以看出，随着 SAD 窗口的增大，视差图平滑度增加，缺失更多的边缘信息(物体的整齐的轮廓变得扭曲)。

4.5.5 SGM+AW 算法

本小节对 SGM+AW 算法进行测试，即使用自适应权值算法(AW)聚合匹配代价，$Cost(i,j,d)=E(p,\bar{p}_d)$,$p$ 为参考图像中的像素点(i,j)， $\bar{p}_d$ 为目标图像中的像素点$(i,j-d)$:

$$E(p,\bar{p}_d)=\frac{\sum_{q\in N_p,\bar{q}_d\in N_{\bar{p}d}} w(p,q)w(\bar{p}_d,\bar{q}_d)e_0(q,\bar{q}_d)}{\sum_{q\in N_p,\bar{q}_d\in N_{\bar{p}d}} w(p,q)w(\bar{p}_d,\bar{q}_d)} \tag{4-28}$$

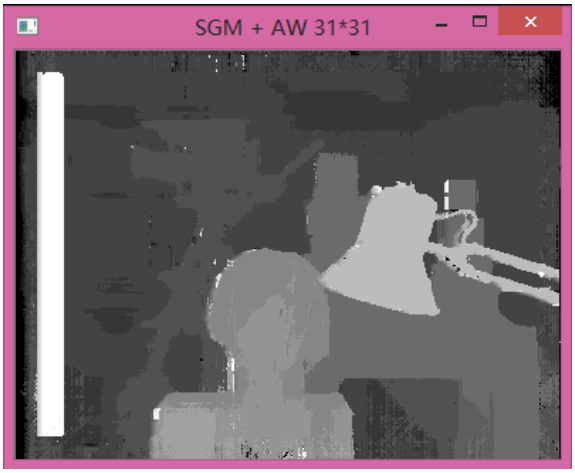
式中各个函数的配置同 4.3.1 节，采用不同窗口的 AW+SGM 算法的实验结果如下：



1*1 AW+SGM 耗时 842 ms

11*11 AW + SGM 耗时 12558 ms

21*21 AW+SGM 耗时 41449 ms



31*31 AW + SGM 耗时 83039 ms

图 4.17 不同窗口的 AW+SGM 算法实验结果

与 FW+SGM 算法一样，1*1 窗口的 AW+SGM 算法退化为单个像素代价的 SGM 算法，随着 AW 窗口的增大，视差图更加平滑、误差更小，与 FW 算法不同的是，物体的边缘仍然十分清晰，但是时间消耗巨大 31*31 窗口的 AW+SGM 算法为 1*1 窗口算法的近 100 倍耗时。

4.5.5 一种迭代的 SGM 算法

SGM 算法的核心思想是通过多个单方向扫描结果之和取得全局函数的近似值，本文提出一种迭代的 SGM 算法，迭代使用 SGM 算法对视差图进行平滑。算法流程如下：

(1)使用某一种匹配代价函数(FW、AW 等)计算匹配代价 Cost(i,j,d)

(2)对 n 个方向计算 $E_r(i,j,d)$

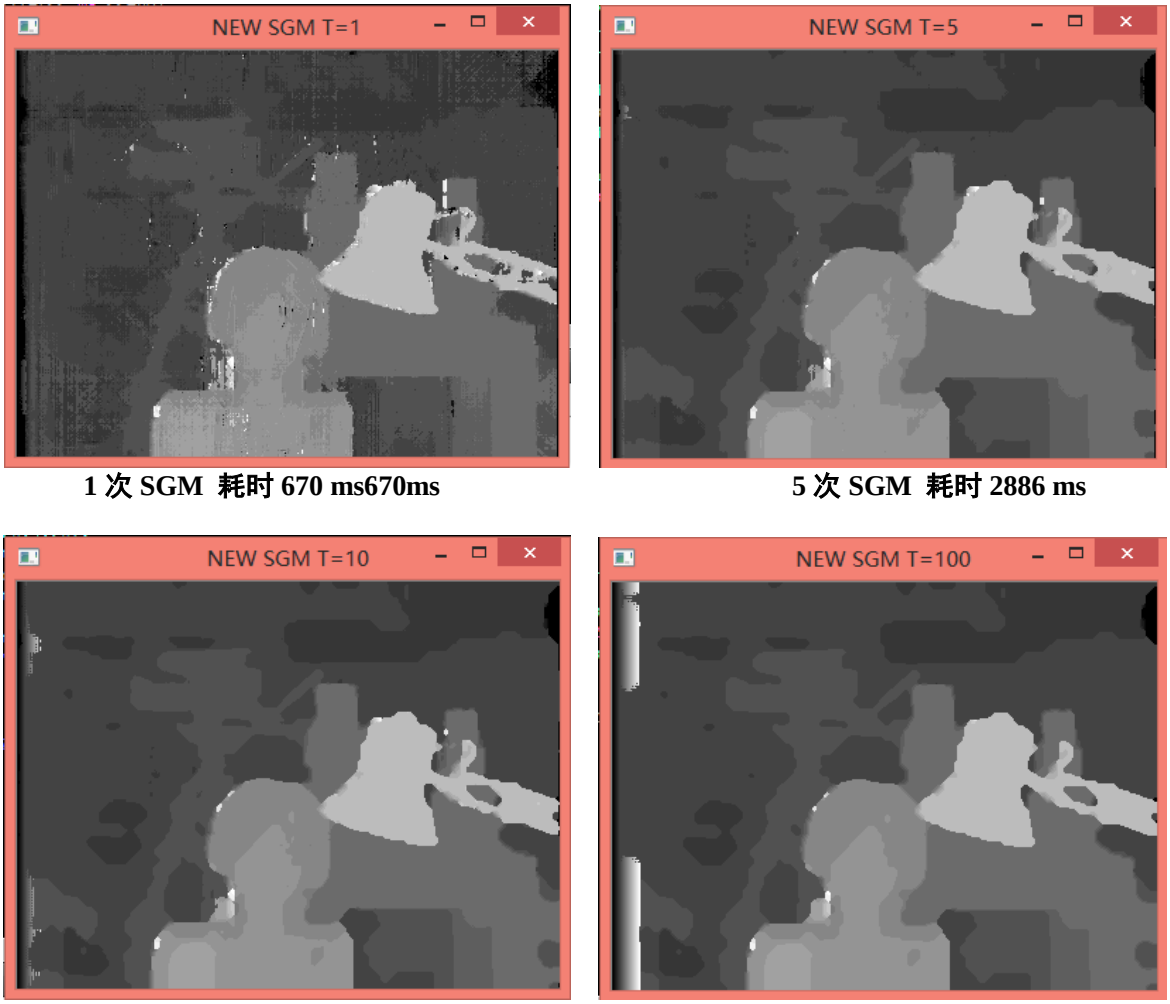
(3)令 $Cost(i,j,d) = \sum_r E_r(i,j,d)/n$

(4)若到达迭代次数上限则转(5)，否则继续迭代，转(2)

(5)采用 WTA 得到视差值 $D(i,j) = \arg\min Cost(i,j,d)$

本小节采用单像素不同通道绝对差值和作为匹配代价，迭代不同的次数 T 进行实验，

实验结果如图 4.18 所示：



1 次 SGM 耗时 670 ms670ms **5 次 SGM 耗时 2886 ms**

10 次 SGM 耗时 5651 ms **100 次 SGM 耗时 61558 ms**

4.18 不同迭代次数的 SGM 算法实验结果

T=1 时，本算法退化为 SGM 算法，当 T=5 时，视差图明显平滑于 SGM 算法的视差图，另外迭代次数为 5、10、100 的效果差异不大(不考虑左图边界遮挡区域)，说明算法在 5 次就几乎达到收敛了。由于采用单个像素代价作为匹配代价，得到的视差图虽然平滑，但是物体的边缘信息得到了保留，如雕塑的直线轮廓和桌子的直线轮廓等。

4.6 立体匹配优化技术

4.6.1 左右一致性检验(LRC)

利用立体匹配的左右一致性约束对视差图进行左右一致性检验，是遮挡检测常用的手段。LRC 过程为：先以左视图为参考图像，右视图为目标图像得到左图中像素点 i 的视差值 $D_L(i)$，再以右图像为参考图像，左图像为目标图像计算得到右图像素点 j 视差值 $D_R(j)$，若两次计算得到的视差值一致 $(D_L(i)=D_R(i-D(i)))$，则通过一致性检验(视差值有效)，否则过滤该点(视差值无效)。

本小节通过 7*7 的 FW 算法，采用左右值检验的结果如图 4.19 所示：

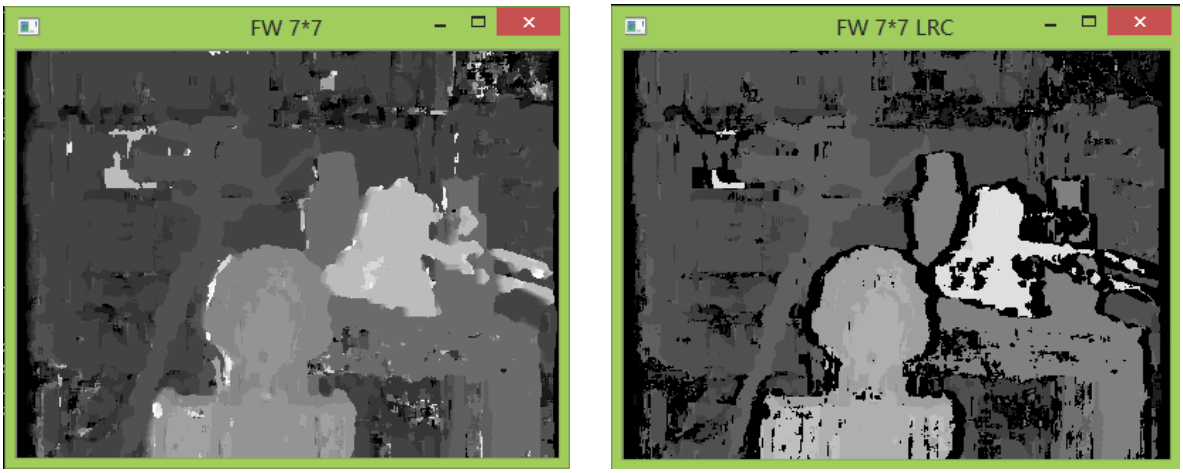


图 4.19 左右一致性检验(右图) FW 7*7(左图)

从图中可以看出，物体存在遮挡的地方(边界处居多)以及一些无匹配点通过左右一致性检验被赋予无效值(黑色像素)。

4.6.2 唯一性检验

在唯一性约束假设中，无论是对于参考图像还是目标图像，图像上的每一个像素点至始至终最多(发生遮挡时没有匹配)只能和另一图像上的一个像素点匹配。通常立体匹配算法本身满足参考图像的唯一性假设，但并不能保证目标图像的唯一性，因此唯一性检验主要检查目标图像上的每一点是否存在多个参考图像上的点与之对应，若同时有多个参考图像上的点 i_1,i_2...同时和目标图像上的某一点 j 匹配,即：

$$i_1 - D(i_1) = i_2 - Di_2 = \cdots = i_n - D(i_n) = j \tag{4-29}$$

程序中通常取和 j 匹配代价最小的点 i_i 之间的视差作为 i_i 的视差值，其余的 i 点视为与 j 的误匹配点。本小节在图 4.19 左右一致性检验的基础之上进行唯一性检验，结果如图 4.20 所示：

4.6.3 连通域阈值过滤

在实际情况中，通过左右一致性检验和唯一性检验并不能完全过滤掉无匹配的像素点。通常情况下，误匹配的像素点和与之视差值相似的临近像素点组成的连通域面积比较小，可以通过筛选视差图连通域面积较小的像素点去掉误匹配点。

本小节采用广度优先搜索算法进行连通域面积域值过滤，算法流程如下：

(1)初始化所有像素点 i 的标签 label[i]为未标记(=-1)，标签计数器 labelc=0

(2)遍历所有像素点 i，若 i 已经有标签 p，p=label[i]，则判断标签 p 的面积是否小于阈值 minarea（area[p]<minarea），若小于，则令 i 为无效点；若 i 没有标签，执行(3)

(3)初始化面积技术器 count=0，将像素点 i 放入队列 queue 中。

(4)从队列中取出像素点 j，count++，标记 j 为标签 labelc(label[j]=labelc)

(5)把 j 邻域的像素点中，没有标记过、和 j 点视差值相差不超过 diff 的像素点放入队

列 queue 中

(6)如果 queue 为空，则算法结束，否则转(4)

本小节使用 diff=1，minarea=200 对 4.6.2 得到的视差图进行连通域阈值过滤，为通过过滤的像素点如图 4.20 所示：

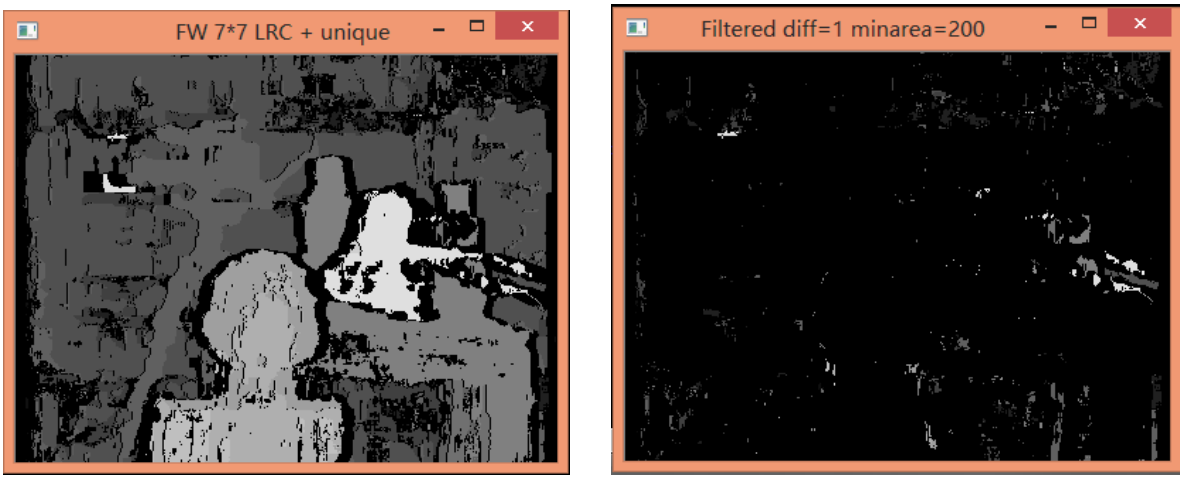


图 4.20 唯一性检验(左) 连通域阈值过滤(右)

4.6.4 亚像素精度视差

大部分立体匹配算法得到的视差值都为离散的整形数值，由此重建得到的物体表面通常呈鱼鳞块状(不够平滑)，为了提高视差图的精确度、平滑度(亚像素级的平滑度)，提高三位重建的精度，通常采用曲线拟合的方式得到视差值的亚像素精度值，在本文中使用抛物线拟合视差代价的方式获取亚像素精度的视差值：

假设通过计算得到了像素点 (i, j) 的匹配代价 $Cost(i, j, d)$,则像素级视差值 **D** 为当匹配代价取得最小值是对应的 d 的值,即

$$D = \min_{d} \arg Cost(i, j, d) \tag{4-30}$$

以 D 为参考点， $Cost(i, j, d)$ 在 D-1,D,D+1 三点的值为 C(-1),C(0),C(1), 通过抛物线拟合 D-1,D,D+1 三点的匹配代价值，如图 4.21 所示：

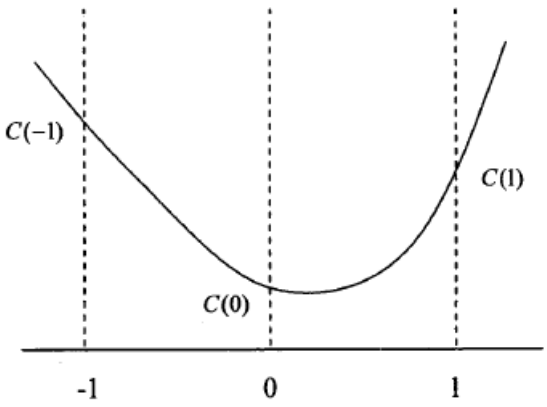


图 4.21 抛物线拟合视差代价

设拟合抛物线方程为： $y=ax^2+bx+c$,抛物线在 $x=-\frac{b}{2a}$ 处取得极小值，则有方程组：

$$\begin{cases} a\bullet 1+b\bullet(-1)+c=C(-1) \\ a\bullet 0+b\bullet 0+c=C(0) \\ a\bullet 1+b\bullet 1+c=C(1) \\ x=-\dfrac{b}{2a} \end{cases} \tag{4-31}$$

求解上述方程组，得到经过抛物线拟合的亚像素精度的视差值为：

$$\hat{d}=x+D=\frac{C(-1)-\mathrm{C}(1)}{2C(-1)-4\mathrm{C}(0)+2\mathrm{C}(1)}+D \tag{4-32}$$

5 三维重建

5.1 获取三维坐标

通过第三章的立体矫正方法，将通常情况下非平行摄像机模型摄像机对拍摄的图片对矫正为为平行摄像机模型摄像机对拍摄的图片。

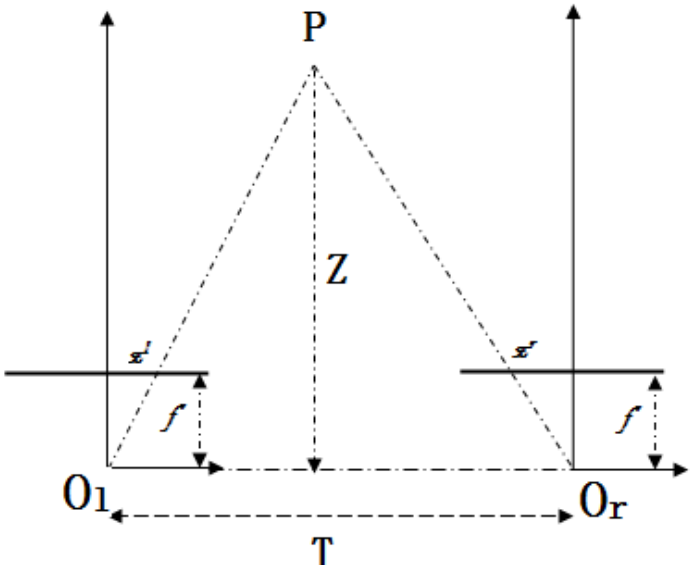


图 5.1 平行摄像机几何示意图

平行摄像机模型的简单几何示意图如图 5.1 所示，以左摄像机的成像坐标系为参考坐标系，通过相似三角形关系，空间中某一点 $P(X,Y,Z)$ 有如下关系：

$$\begin{cases} Z = \dfrac{fT}{x_l - x_r} = \dfrac{fT}{d} \\ X = \dfrac{x_l T}{d} \\ Y = \dfrac{y_l T}{d} \end{cases} \tag{5-1}$$

其中 d 为 P 点在左图中对应像素点的视差值(通过立体匹配得到)，$x_l = u - u_0, y_l = v - v_0$，$(u,v)$ 为 P 点在左图中坐标，(u_0,v_0) 为左摄像机光心的坐标，这些参数都通过预先对摄像机标定获取。

5.2 OpenGL 点云图

本文采用 Qt+OpenGL 绘制点云图，具体过程如下：

(1)对于左图中的每一个像素点 i，若通过立体匹配计算得到 i 点的视差值有效(非遮挡点或被过滤掉的像素点)，则进行(2)、(3)操作。

(2)通过 OpenGL 接口将绘制颜色设置为像素点 i 在左图中的颜色

(3)计算出 i 点的相对于左摄像机的三维坐标值(X,Y,Z),并通过 OpenGL 绘制该点(具有

物体的颜色信息)。

采用上述方式绘制的点云图带有简单的纹理，使得重建物体色彩丰富，辨认度较高，采用 DP+AW 立体匹配算法的三维重建结果如图 5.2 所示：

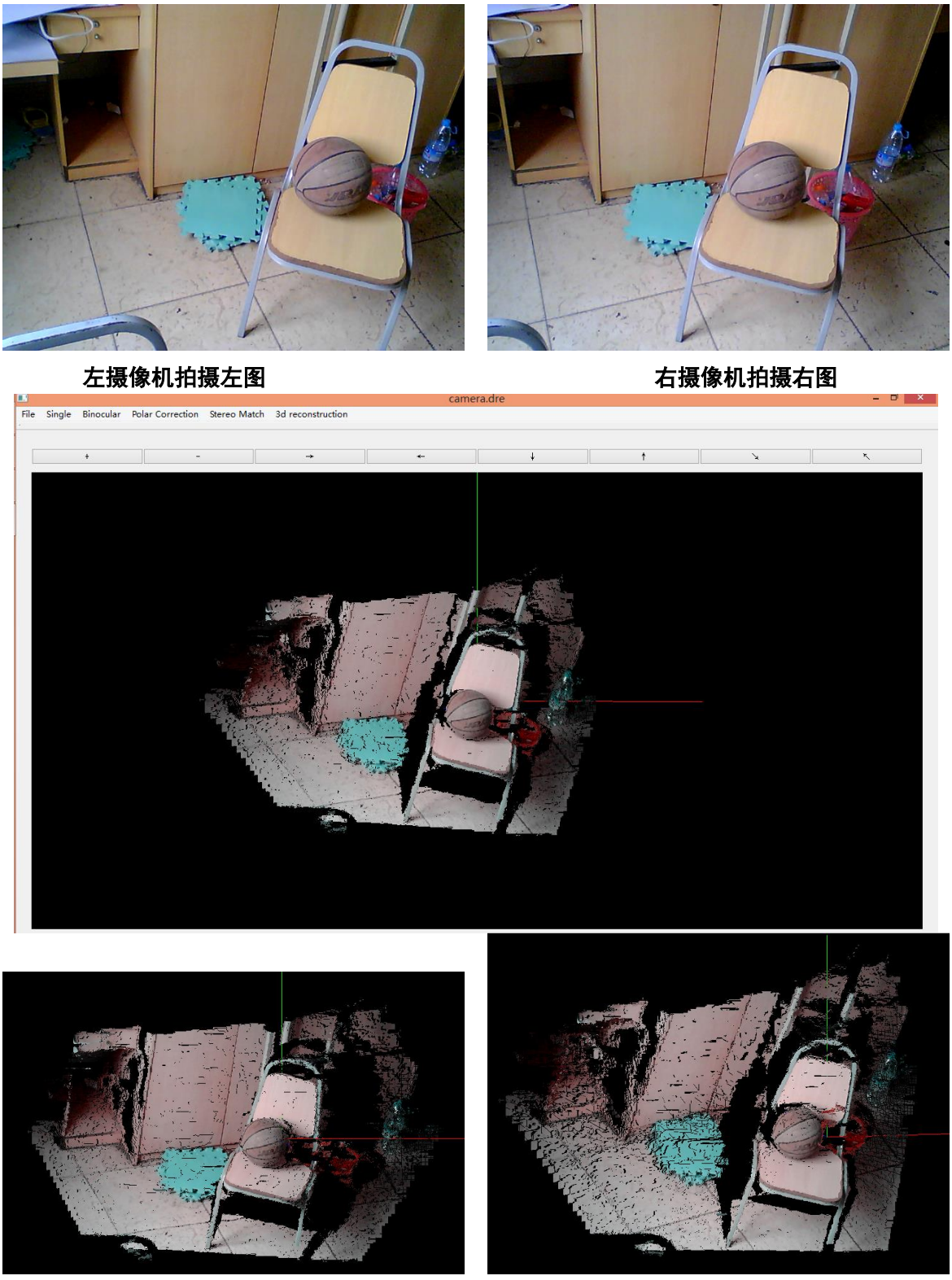


图 5.2 三维重建点云重建图

总 结

本文着重对双目立体视觉三维重建中的标定和立体匹配进行了研究，实现了经典的角点检测 Moravec、Harris、Nobel、Shi-Tomasi 算法，以及分别采用二次曲面拟合方法和基于向量点乘的方法提取亚像素精度角点。实现了经典的固定窗口 FW 算法，并采用 Box-filtering 加速、实现了自适应权值 AW 算法以及快速双边滤波立体匹配 FBS 算法、实现了动态规划 DP 立体匹配算法，并将 AW 和 FBS 算法作为 DP 算法的匹配代价聚合算法分别进行测试；实现了单扫描线、多扫描线算法，实现了 SGM+FW 以及 SGM+AW 算法。

本文提出一种迭代的 SGM 算法，该算法在 tsukuba 测试图片上表现良好，且时间消耗并不太多(为 SGM 算法的 2-5 倍处理时间)。

尽管如此，由于时间有限，本文仍存在着以下不足以及一些尚待改进的地方：

(1)对各个立体匹配算法的测试并不严格，在未来的工作中，将考虑使用 Middlebury 网站上提供的测试对算法的各项性能逐一进行测试，并得到定量(准确度、误匹配率等)的指标。

(2)本文提出的迭代的 SGM 算法在匹配代价和平滑代价量纲上还有改进的空间。

(3)本文最终通过三维点云图表现三维重建结果，点云图是离散化数据，随着重建比例的放大，点云图会变得稀疏，呈现明显的散点状。在将来考虑进行点云网格化，再进行纹理映射以解决这个问题。

作者在读期间科研成果介绍

参考文献

[1] 章毓晋.图像理解与一计算机视觉，北京:清华大学出版社，2000.

[2] 郑志刚. 高精度摄像机标定和鲁棒立体匹配算法研究[D].中国科学技术大学，2008

[3] 刘英杰. 基于动态规划和置信传播的立体匹配算法的研究[D].燕山大学,2011

[4] 赵宗涛. 计算机视觉三维重建理论与应用[D].西北大学，2004

[5] 周星,高志军. 立体视觉技术的应用与发展[J].工程图学学报,2010(4):50-55

[6] 于辉,左洪福,黄传奇. 立体视觉在航空发动机无损检测中的应用[J].无损检测,2003(5):229-233

[7] 曾一庭. 基于立体视觉的牙模三维重建系统的研究[D].重庆大学，2009

[8] 马颂德,张正友. 计算机视觉-计算理论与算法基础[M],科学出版社,1998.

[9] 中国科学院自动化研究所-模式识别国家重点实验室.摄像机标定[EB/OL]. http://nlpr-web.ia.ac.cn/course/calibration.pdf

[10] A bdel-Aziz Y I,Karara. HM Direct linear transformation into object space coordinates in Close-Range Photogrammetry[J].In:Proc Symposition on Close-Range Photogrammmetry,1971:1-18

[11] Tsai R Y.An efficient and accurate camera calibration technique for 3D machine vision[J].In:Proc CVPR'86,364-374

[12] Zhang zheng you. A Flexible Camera Calibration by Viewing a Plane from Unknown Orientations,ICCV99

[13] 李鹏,王军宁. 摄像机标定方法综述[J].山西电子技术，2007(4):77-79

[14] 宰小涛.基于 SIFT 特征描述子的立体匹配算法研究[D].上海交通大学.2007

[15] Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms[A]. International Journal of Computer Vision, 2002, 47(1-3):7–42.

[16] D. Scharstein. Matching images by comparing their gradient fields[A]. In ICPR, vol. 1, pp. 572–575, 1994.

[17] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence[A].In ECCV, vol. II, pp.151–158, 1994.

[18] H. Hirschmuller, P. Innocent, and J. Garibaldi, Real-time correlation-based stereo vision with reduced border errors[A] Int. Journ. of Computer Vision, 47:1–3, 2002

[19] M. Gerrits and P. Bekaert. Local Stereo Matching with Segmentation-based Outlier Rejection In Proc. Canadian Conf. on Computer and Robot Vision (CRV 2006),pages 66-66, 2006

[20] K. Yoon and I. Kweon, Adaptive support-weight approach for correspondence search, IEEE Trans. PAMI, 28(4):650–656, 2006

[21] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(11):1222–1239, November 2001

[22] 施陈博.快速图像配准和高精度立体匹配算法研究[D].清华大学，2011

[23] 王二柱.基于点云的三维重建系统研究与实现[D].哈尔滨工业大学，2011

[24] 苏晋. 摄像机标定方法研究[D].东北大学，2010

[25] 杨彦景. 摄像机标定与畸变图像矫正算法的设计与实现[D].东北大学,2008

[26] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. IEEE Journal of Robotics and Automation, 3(4):323–344, Aug. 1987.

[27] 姚静. 摄像机标定相关问题研究[D].南京理工大学,2012

[28] MORAVEC H P. Towards automatic visual obstacle avoidance[C]. Proceeding of IJCA I. Cambridge.1977:584-590

[29] C.Harris, M.Stephens. A Combined Corner and Edge Detector[C].Proc of 4th Alvey Vision Conference,1988:147-151

[30] J.Shi,J.Malik. Good features to track[C].IEEE Transactions on Pattern Analysis and Machine Intelligence 22,2000:810-813

[31] 陈光. 亚像素级角点提取算法[D].吉林大学,2009

[32] CHRISTOPH STOCK, ULRICH MÜHLMANN. Sub-pixel Corner Detection for Tracking Applications using CMOS Camera Technology[C].26th Workshop of the Austrian Association for Pattern Recognition (ÖAGM/AAPR), Graz Austria: [s. n.], 2002: 191－199

[33] J.A.Noble.Finding corners. Image and Vision Co mputing,1988,6 (2) :121-128.

[34] 殷虎 基于图像分割的立体匹配算法研究[D].南京航空航天大学，2010

[35] M.Mc Donnel. Box-filtering techniques[C]. Computer Graphics and Image Processing 17:65-70,1981

[36] Bouguet JY. Perona P, Camera calibration from points and lines in dual-space geometry. Proc. European Conference on Computer Vision, 1998. 2-6

[37] S. Mattoccia, S. Giardino,A. Gambini. Accurate and efficient cost aggregation strategy for stereo correspondence based on approximated joint bilateral filtering[C].Asian Conference on Computer Vision (ACCV2009)

[38] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information[C]. In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 807–814, San Diego, CA, USA, June 2005.

声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

本学位论文成果是本人在四川大学读书期间在导师指导下取得的，论文成果归四川大学所有，特此声明。

学位论文作者（签名）_____

论文指导教师（签名）_____

2014 年 05 月 21 日

致 谢

首先我在这里向所有给予我帮助、关心和指导的老师、领导、同学、师兄们表示衷心地感谢！是你们帮助我在学业上和人生道路上茁壮地成长。

在大四上学期，我有幸成为王老师“技术讨论”课的学生，王老师渊博的知识让我深深的折服了，在王老师的课堂上，前沿的研究课题以及王老师向我们传达的作为一名中国当代大学生应具有的历史使命感和责任感，都让我坚定了投身科研的决心。

感谢张磊师兄为我的论文提出宝贵的意见和建议，感谢王老师对我毕业设计和毕业论文的悉心指导，感谢所有给予我指导的老师们，是你们将计算机科学的魅力转化为我求知的动力。

附录 3　翻译（原文和译文）

翻译原文来自：

Z. Zhang, "A flexible new technique for camera calibration",IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330–1334, 2000.

Link: http://research.microsoft.com/en-us/um/people/zhang/Papers/TR98-71.pdf

译文：

一种新的灵活摄像机标定技术

摘要

论文提出一种新的灵活容易的摄像机标定技术，即使没有任何 3d 几何或者计算机视觉的相关知识，它只需要摄像机对一个模型平面采集若干（至少两个）不同方向的照片，即可出色的完成标定。无论是摄像机或者标定板都可以自由挪动，而并不需要知道摄像机的姿态。标定还考虑了摄像机的径向畸变。论文提出的程序由一个闭合形式的解，以及基于最大似然准则的非线性优化。文中提到的方法在计算机模拟和真实数据的测试中都表现优异。与使用精确的标定器材如两个或三个正交标定板的传统标定技术相比，论文提出的方法非常灵活和易用。它把 3D 计算机视觉由实验室环境带到了真实的应用场景。

关键词：摄像机标定，平板标定，2 维模型，绝对二次曲线，射影变换，镜头畸变，闭合解，最大似然估计，灵活设定

1 背景

摄像机标定是计算机视觉领域里从二维图像提取三维空间信息必不可少的步骤。从摄影社区（见文献 2,4），到最近的计算机视觉（见文献 9，8，23，7，26，24，17，6），已经进行了大量的研究。论文可以把这些技术大致分为两类：摄影测量标定和自标定。

摄影测量标定：通过观测一个已知 3d 空间几何参数的高精度标定物体来进行摄像机标定。通过这种方式，能过获取到精确的结果。标定物体通常由两个或者相互正交的标定板，有时候，仅一个已知精确转换的标定板也可以用于标定（见文献 23），这些方法都要求十分精确的标定仪器以及复杂的设置。

自标定：这一类的标定方法并不需要任何标定物。在一个静态的场景中移动摄像机，场景的刚性仅仅依靠靠图像信息，就能从这个摄像机的移动中，提供摄像机内参两个约束（见文献 17，15）。因此，如果由同一个内参不变的摄像机拍摄，三张照片之间的对应关系足够获取内参和外参，论文便能通过这些参数进行大致的 3 维重建。虽然这种方法非常的灵活，但由于这个过程中有太多的参数需要拟合，所以并未成熟应用起来，并不能得到可信的结果。

其他的一些技术：正交方向的消失点（文献 3，14）以及纯旋转法标定（文献 11，21）。

由于桌面视觉系统（DVS）的前景广阔，所以论文的研究主要针对 DVS。现在摄像机越来越普遍和便宜，而 DVS 旨在非计算机视觉专家的大众人群。一般的计算机用户只是时不时地从事视觉工作，而并不愿意为昂贵的设备投资。因此，灵活、鲁棒并且低成本是非常重要的。本文提出的标定技术正是基于上述考虑而开发的。

本文提出的方法只需要摄像机从一个平面图案的若干个（至少两个）方向获取图像。图案可以是由激光打印机打印、粘在一个硬度合适的平面上（如一本硬书壳），不论是摄像机或者图案都能用手移动。摄像机和图案的姿态不需要预先知道。由于论文只是用了 2D 度量信息而非 3D 或者隐式的，所以本文提出的方法介于摄影测量标定和自标定方法之间：在计算机模拟以及真是出局测试中都得到了非常好的结果。与传统标定技术相比，本文提出的方法非常的灵活；与自标定技相比，它得到非常鲁棒的结果。论文相信这项技术将让 3D 计算机视觉从实验室环境上升到实际应用的台阶。

需要指出的是，BillTriggs[22]最近开发出了一种通过至少 5 张平面场景视图的自标定技术，他的方法比本文提出的方法更为灵活，但是初始化却相当困难。Liebowitz 和 Zisserman[14]描述了一种矫正方法，它通过已知的度量信息比如，一个已知的角度、两个相等但未知的角度、一个已知的长度比率矫正图案的透视图。他们虽然没有展示任何实验结果，但是他们认为，通过提供 3 个这样矫正之后的图案可以标定摄像机的内部参数。

本文按照如下方式组织：第 2 节描述了观测单个平面基本的约束。第 3 节描述了标定步骤，从闭合形式解开始，然后是非线性优化，还有径向畸变的建模。第 4 节研究何种情况下该技术会失效。在实际操作中，可以非常容易避免这样的情况。第 5 节提供了实验结果。论文使用计算机模拟以及真实数据来验证该技术。在附录中，论文提供了一些实现的细节，包括拟合模型平面和它图像之间单应性矩阵的技术。

2 基本的方程

论文通过观测单一平面来考察摄像机内参的约束，本小节开始将列出本文用到了一些记号。

2.1 表示法

一个 2 维的点由 $m=\begin{bmatrix}u & v\end{bmatrix}^T$ 表示，一个 3D 点由 $M=\begin{bmatrix}X & Y & Z\end{bmatrix}^T$ 表示，论文使用 $\tilde{\mathbf{x}}$ 来表示在向量的最后一维元素之后加上 1 的向量：$\tilde{m}=\begin{bmatrix}u & v & 1\end{bmatrix}^T$ 和 $\tilde{M}=\begin{bmatrix}X & Y & Z & 1\end{bmatrix}^T$，摄像机使用常用的针孔模型：3D 中的点 M 和它的投影图像 m 的关系如下：

$$s\tilde{m}=A\begin{bmatrix}R & t\end{bmatrix}\tilde{M} \tag{1}$$

其中 s 是任意的比例因素，(R,t),叫做外参，分别为真实的世界坐标系到摄像机坐标系的旋转和平移变换，A，叫做摄像机的内参，形如：

$$A=\begin{bmatrix}\alpha & \gamma & u_0\\ 0 & \beta & v_0\\ 0 & 0 & 1\end{bmatrix}$$

其中(u_0,v_0)为光心的坐标，α 和 β 分别是图像坐标系 u，v 的比例因子，γ 表示两个坐标轴的偏斜度。

论文中使用 A^{-T} 缩略表示 $(A^T)^{-1}$ 或者 $(A^{-1})^T$。

2.2 平面模型和它的图像之间的单应性矩阵

不失一般性，论文假设模型平面位于世界坐标系中的 Z=0 平面上，让 r_i 表示 **R** 矩阵的第 i 列，从(1)有：

$$s\begin{bmatrix}u\\ v\\ 1\end{bmatrix}=A\begin{bmatrix}r_1 & r_2 & r_3 & t\end{bmatrix}\begin{bmatrix}X\\ Y\\ 0\\ 1\end{bmatrix}$$

$$=A\begin{bmatrix}r_1 & r_2 & t\end{bmatrix}\begin{bmatrix}X\\ Y\\ 1\end{bmatrix}$$

论文依然使用 M 表示模型平板上的点，但由于 **Z**=0，$M=\begin{bmatrix}X & Y\end{bmatrix}^T$,反过来，$\tilde{M}=\begin{bmatrix}X & Y & 1\end{bmatrix}^T$，所以模型上的点 M 和它对应的图像上的点 m 通过一个单应性矩阵 **H** 联系起来:

$$s\tilde{m}=H\tilde{M} \quad with \quad H=A\begin{bmatrix}r_1 & r_2 & t\end{bmatrix} \tag{2}$$

很明显，**H** 是由一个比例因子决定的 3*3 的矩阵。

2.3 内部参数的约束

给定一张模型平面的图片，可以估算出一个单应性矩阵(见附录 A)，论文记为 **H**=[h_1,h_2,h_3],由(2)有：

$$\begin{bmatrix}h_1 & h_2 & h_3\end{bmatrix}=\lambda A\begin{bmatrix}r_1 & r_2 & t\end{bmatrix}$$

其中 λ 是任意的缩放因子，注意到 r_1，r_2 是正交的，有：

$$h_1^T A^{-T}A^{-1}h_2=0 \tag{3}$$

$$h_1^T A^{-T}A^{-1}h_1=h_2^T A^{-T}A^{-1}h_2 \tag{4}$$

以上是对于给定的一个单应性矩阵，内参的两个基本约束。由于一个单应性矩阵有 8 个自由度，外参有 6 个参数（3 个旋转参数、3 个平移参数），论文只能得到内参的两个约束。注意到 $A^{-T}A^{-1}$ 实际上描述了图像的绝对二次型[16].在下一小章节中，论文将给出一个几何意义上的解释。

2.4 几何意义

论文现在把(3)(4)和绝对二次型联系起来。

在摄像机坐标系中，对于模型平面，按照论文的通常习惯，不难证明等式：

$$\begin{bmatrix} r_3 \\ r_3^T t \end{bmatrix}^T \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = 0$$

其中 w=0 表示无穷远点，反之，w=1。这两个平面在无穷处交于一条直线，论文很容易发现 $\begin{bmatrix} r_1 \\ 0 \end{bmatrix}$ 和 $\begin{bmatrix} r_2 \\ 0 \end{bmatrix}$ 是该条直线上的两个特殊点。该直线上的任何点均为这两点的线性组合，即：

$$x_\infty = a\begin{bmatrix} r_1 \\ 0 \end{bmatrix} + b\begin{bmatrix} r_2 \\ 0 \end{bmatrix} = \begin{bmatrix} ar_1 + br_2 \\ 0 \end{bmatrix}$$

现在，论文来计算上述直线和绝对二次型的交点，由定义有，点 x_∞ 即虚圆点，满足： $x_\infty^T x_\infty = 0$，即：

$$(ar_1 + br_2)^T (ar_1 + br_2) = 0 \text{ 或者 } a^2 + b^2 = 0$$

方程的解为 $b = \pm ai$,其中 $i^2 = -1$,即，两个交点即:

$$x_\infty = a\begin{bmatrix} r_1 \pm ir_2 \\ 0 \end{bmatrix}$$

它们在图像平面由比例因子确定的投影为：

$$\tilde{m}_\infty = A(r_1 \pm ir_2) = (h_1 \pm ih_2)$$

点 $\tilde{m}_\infty$ 在绝对二次型的图像上，可以由 $A^{-T}A^{-1}$ **[16]**表示成如下形式：

$$(h_1 \pm ih2)^T A^{-T}A^{-1}(h_1 \pm ih_2) = 0$$

实部和虚部必都为 0 即(3)(4).

3 摄像机标定

这一节将给出如何高效地解决摄像机标定问题的详细步骤,论文先从解析解开始，然后基于最大似然规则进行非线性优化。最后，计算镜头畸变，得到解析解和非线性解。

3.1 闭合形式解

$$B = A^{-T}A^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

$$\text{令} \quad = \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix} \tag{5}$$

注意到 B 是一个对称矩阵，定义一个六维的向量 b

$$b = \begin{bmatrix} B_{11} & B_{12} & B_{22} & B_{13} & B_{21} & B_{33} \end{bmatrix}^T \tag{6}$$

令 **H** 的第 i[th] 列向量为 $h_i = \begin{bmatrix} h_{i1} & h_{i2} & h_{i3} \end{bmatrix}^T$ ，有

$$h_i^T B h_j = v_{ij}^T b \tag{7}$$

其中

$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2}+h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1}+h_{i1}h_{j3} & h_{i3}h_{j2}+h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}^T$$

所以，给定一个单应性矩阵的两个基本约束(3)(4),用 **b** 表示为：

$$\begin{bmatrix} v_{12}^T \\ (v_{11}-v_{22})^T \end{bmatrix} b = 0 \tag{8}$$

若采集了 n 幅模型平面的图片，通过建立 n 个(8)这样的方程，有：

$$Vb = 0 \tag{9}$$

其中 **V** 是 2n x 6 的矩阵，如果 n≥3，可以通过比例因子唯一确定 **b** 的一组解，如果 n=2，可以加上倾斜因子 γ =0，如[0,1,0,0,0,0]**b**=0,相当于在方程组(9)中添加了一个等式。（如果 n=1，最多可以解出内参矩阵中的两个参数，比如，假设 u0,v0 是已知的(比如，位于图像中心),）并且 γ =0,这实际上就是论文基于眼睛和嘴共面这一合理假设的头部姿态估计中所用到的技术[19]。(9)的解是论文熟知 V^TV 的最小特征值所对应的特征向量(即 V 最小奇异值所对应的奇异向量)。

一旦求出 **b** 之后，便能计算出摄像机的内参矩阵 **A**，详细过程见附录 **B**。

而当 **A** 已知后，每张图片的外参矩阵就容易就计算出了。由(2)有：

$$r_1 = \lambda A^{-1} h_1$$

$$r_2 = \lambda A^{-1} h_2$$

$$r_3 = r_1 \times r_2$$

$$t = \lambda A^{-1} h_3$$

其中 $\lambda = 1/\left\|A^{-1}h_1\right\| = 1/\left\|A^{-1}h_2\right\|$，当然，由于数据中存在噪声,这样计算出的矩阵 $R = [r_1, r_2, r_3]$ 并不一定能满足旋转矩阵的所有性质。附录 C 有由普通 3*3 矩阵估计最佳的旋转举证的详细方法。

3.2 最大似然估计

由于上述解是通过最小化代数距离而求得的结果，所以物理意义并不明显。论文可以通过最大似然估计推导来重新解释一下它的意义。

设有 n 张模型平面的图像，每一个模型平面上有 m 个点。假设图像上的点存在独立分散的噪声。可以通过求解如下式子的最小值进行最大似然估计:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|m_{ij} - \tilde{m}(A, R_i, t_i, M_j)\right\|^2 \tag{10}$$

其中 $\tilde{m}(A, R_i, t_i, M_j)$ 是由(2)式计算出第 i 幅图像上点 M_j 的投影的理论值。旋转矩阵 **R** 可以通过一个 3 维向量参数化，即表示为 **r**，**r** 平行于旋转轴，大小等于旋转角度。**R** 和 **r** 满足毛里求斯方程式[5]。最小化(10)式本质上为非线性最小化问题，可以通过 Minpack[18]实现的 Levenberg-Marquardt(LM)算法求解。而 LM 算法需要初始化的 $\{R_i, t_i \mid i = 1..n\}$ ，**A** 矩阵的初始值就就可以使用上一章节介绍的算法得到。

3.3 径向畸变

至此，本文尚未考虑摄像机的径向畸变。然而，通用摄像机通常带有明显的镜头畸变，有其是径向畸变。本章中，只考虑径向畸变的前两个因素。读者可以参考[20,2,4,26]其他更加详细的模型。根据文献[2,23,25]中的报告，镜头失真主要是由径向畸变造成的，有其与第一个因子密切相关。另外，实验中还发现更精细的建模不仅没有多大效果(与传感器量相比，可以忽略不计)，反而会导致数值不稳定[2,25]。

令 (u, v) 为理想(不考虑畸变)的像素点坐标，$(\tilde{u}, \tilde{v})$ 为实际对应图像上的点坐标。理想点是模型点根据小孔成像模型的投影点。同理，(x, y) 和 $(\tilde{x}, \tilde{y})$ 分别是理想(不考虑畸变)的成像坐标和实际的成像坐标，有[2,25]

$$\breve{x} = x + x\left[k_1(x^2+y^2) + k_2(x^2+y^2)^2\right]$$
$$\breve{y} = y + y\left[k_1(x^2+y^2) + k_2(x^2+y^2)^2\right]$$

其中 k1，k2 为径向畸变系数。径向畸变和理论成像平面原点重合。由 $\breve{u} = u_0 + \alpha\breve{x} + \gamma\breve{y}$ 和 $\breve{v} = v_0 + \beta\breve{y}$，假设 γ =0，有：

$$\tilde{u} = u + (u - u_0)[k_1(x^2+y^2) + k_2(x^2+y^2)^2] \quad (11)$$
$$\tilde{v} = v + (v - v_0)[k_1(x^2+y^2) + k_2(x^2+y^2)^2] \quad (12)$$

迭代估计径向畸变。由于径向畸变非常的小，有的时候可以使用 3.2 节的技术，直接忽略畸变，求出 5 个内参参数。另外，可以在求出其他的参数之后再估计 k1,k2 的值；当其他参数已知时，得到理想的像素坐标(u,v)，然后每幅图片由(11,12)，有两个等式：

$$\begin{bmatrix} (u-u_0)(x^2+y^2) & (u-u_0)(x^2+y^2)^2 \\ (v-v_0)(x^2+y^2) & (v-v_0)(x^2+y^2)^2 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} \tilde{u}-u \\ \tilde{v}-v \end{bmatrix}$$

假设有 n 幅图片，每张图片上有 m 个点，就能得到 2mn 个等式，写成矩阵的形式为：$Dk = d$,其中 $k = \begin{bmatrix} k_1 & k_2 \end{bmatrix}^T$。最小二乘解为：

$$k = (D^T D)^{-1} D^T d \quad (13)$$

当 k1 和 k2 求得之后，就可以将(11)和(12)替换(10)中的 $\tilde{m}(A, R_i, t_i, M_j)$ 来重新求解其他的参数，论文可以反复迭代这两个过程直至收敛。

完整的最大似然估计 在实验中，论文发现上述方法的收敛速度太慢。而将(10)式自然地扩展为：

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\| m_{ij} - \tilde{m}(A, k1, k_2, R_i, t_i, M_j)\right\|^2 \quad (14)$$

问题转化为求上式得最小值问题。其中 $\tilde{m}(A, k1, k_2, R_i, t_i, M_j)$ 为 i 幅图片中 M_j 点的投影，见(2)以及(11)、(12)。这是一个常规的非线性优化问题，可以使用由 Minpack[18]实现的 LM 算法算出结果。旋转同样使用 3.2 节中提到的 3 维向量 **r** 参数化，**A** 矩阵以及 $\{R_i, t_i \mid i = 1..n\}$ 依然可以使用 3.1 或者 3.2 节中的方法。k1,k2 的初值可以使用上述最后一段的方法，或者直接令为 0。

3.4 小节

推荐标定过程如下：

1、 打印图案并固定在一个平整的表面上
2、 移动模型平面或者摄像机，从不同的方向拍摄若干照片
3、 检测图片上的特征点
4、 使用 3.1 节所述的方法估算 5 个内参参数以及所有图片对应的外参
5、 使用最小二乘法估算径向畸变系数(13)
6、 优化得到所有的参数(14)

4 退化配置

本节探讨附加的图片不为摄像机内参提供更多的约束的配置。由于(3)和(4)式均由旋转矩阵衍生而来，如果 R_2 与 R_1 相关，图片 2 就没有提供约束。更为特殊的情况是，假如平面仅仅经过平移变换，即 R_2=R_1，则图片 2 对于摄像机标定可有可无。接下来，论文考虑更为复杂的配置。

命题 1 如果第二张模型平面和第一张模型平面平行，则第二个单应性矩阵不产生额外的约束。

证明 假设 R_2 绕 z 轴旋转得到 R_1 ，即：

$$R_1 \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_2$$

其中 θ 为旋转角度。这里采用上标(1)(2) 分别表示图片 1 和图片 2 的向量，那么对两张图片分别有：

$$h_1^{(2)} = \lambda^{(2)}(Ar^{(1)}\cos\theta + Ar^{(2)}\sin\theta) = \frac{\lambda^{(2)}}{\lambda^{(1)}} h_1^{(1)}(\cos\theta + h_2^{(1)}\sin\theta)$$

$$h_2^{(2)} = \lambda^{(2)}(-Ar^{(1)}\sin\theta + Ar^{(2)}\cos\theta) = \frac{\lambda^{(2)}}{\lambda^{(1)}} h_1^{(1)}(-\sin\theta + h_2^{(1)}\cos\theta)$$

然后，图片 2 的第一个约束(3)变成：

$$h_1^{(2)T} A^{-T} A^{-1} h_2^{(2)} = \frac{\lambda^{(2)}}{\lambda^{(1)}}[(\cos^2\theta - \sin^2\theta)(h_1^{(1)T} A^{-T} A^{-1} h_2^{(1)})$$

$$-\cos\theta\sin\theta(h_1^{(1)T} A^{-T} A^{-1} h_1^{(1)} - h_2^{(1)T} A^{-T} A^{-1} h_2^{(1)})],$$

该式实际上与 $\mathbf{H_1}$ 的两个约束线性相关。同理，图片 2 的第二个约束也是与 $\mathbf{H_1}$ 的两个约束线性相关，因此，$\mathbf{H_2}$ 并没有增加更多的约束条件。

结果实际上是不言而喻的，因为平行的平面相交于无穷远处，根据 2.4 节，这两个平面产生相同的约束。

而在实际上是非常容易避免上述这样的退化配置：仅仅需要改变模型平面的方向就可以了。

尽管在模型平面做纯平移变换的情况下，本文提出的方法将失效，但是倘若这个变换时已知的，同样可以进行摄像机标定。详细过程参考附录 D。

5 实验结果

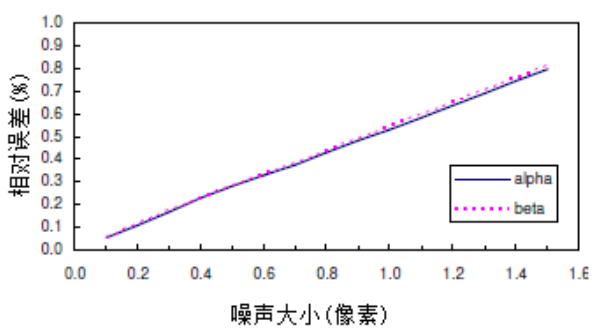
采用计算机模拟数据、实际数据测试本文提出的算法。闭合形式解主要包括奇异值分解一个 2n * 6 的小规模矩阵，其中 n 为图片的数目。采用 LM 非线性细化算法大概需要 3-5 次迭代即可收敛。

5.1 计算机模拟

模拟摄像机的参数如下:$\alpha = 1250, \beta = 900, \gamma = 1.09083$(即 89.95°)，$u_0$=255,$v_0$=255，图片的分辨率为 512*512。模型平面为包含 10*14=140 个交点的棋盘图案(因此，通常在 v 方向上会有比 u 方向上更多地数据)。模型平板为 18cm*25cm。平面的方向由一个 3 维向量 r 表示，其中 r 的方向和旋转轴平行，大小为旋转的角度。平面的位置由一个 3 维向量 t 表示。

不同级别噪声对性能的影响 在这个实验中，论文使用 3 个模型平面，参数分别为：$r_1 = [20^\circ, 0, 0]^T$，$t_1 = [-9, -12.5, 500]^T$，$r_2 = [0, 20^\circ, 0]^T$，$t_2 = [-9, -12.5, 510]^T$，$r_3 = \frac{1}{\sqrt{5}}[-30^\circ, -30^\circ, -15^\circ]^T$，$t_3 = [-10.5, -12.5, 525]^T$ 将均值为 0，标准差为 σ 的高斯噪声加到投影图像点上。之后把计算出的摄像机的参数和实际的参数相比较，并计算 α 和 β 的相对误差，以及 u_0,v_0 的绝对误差。实验中，在 0.1 像素到 1.5 像素范围内改变噪声的程度。对于每个不同程度的噪声，论文进行了 100 次独立的实验，并将实验的平均值作为结果。从表 1 中，可以看出，误差随着噪声的程度提高而线性增大(这里没有给出 γ 的误差表，但是也有相同的性质)。α 和 β 的相对误差不超过 0.3%，u_0,v_0 的绝对误差为 1 像素左右，u_0 的误差大于 v_0，主要是由于 u 方向上的数据量小于 v 方向上的数据量，上文也有提及。

平面数量对性能的影响 这个实验探究平面的数目(更准确地说，是模型平面的图片数目)和性能的关系。前 3 幅图片模型平面的方向和位置和上文中的一样。从第 4 幅图片开始，论文先随机的选择一个空间中的旋转轴，然后再绕着这个轴旋转 30 度。每组实验照片数目从 2 到 16 张不等。每组从不相关的角度(前 3 张除外)以及使用均值为 0，



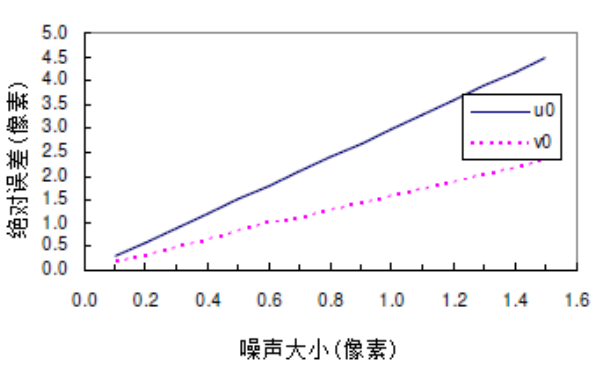


图 1 误差—图像点噪声水平 折线图

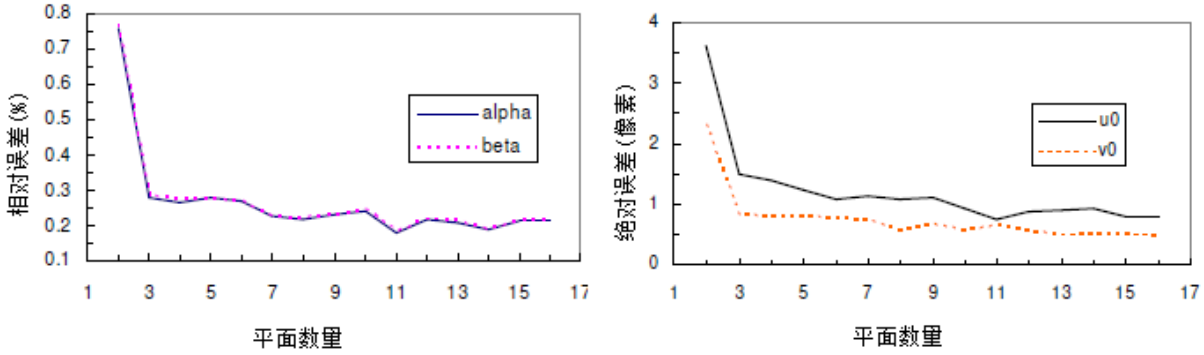


图 2 误差—图片数量 折线图

标准差为 0.5 像素的随机噪声进行 100 次独立实验。将均值作为结果，如图 2 所示。可以看到，随着图片数量的增加，误差逐渐减小，尤其当图片数量从 2 增加到 3 时，误差显著下降。

模型平面的朝向和性能的关系 本实验测试模型平面与成像平面之间的角度对性能的影响，实验使用 3 张图片，图片的角度由如下方式产生：平行于成像平面；随机选择一个空间旋转轴，然后将模型平面绕该轴旋转角度 θ 。同样将均值为 0、标准差为 0.5 像素的高斯噪声附加到图片，重复以上步骤 100 次求得平均误差。 θ 从 5 度到 75 度不等。结果如图 3 所示。当 θ =5 度时，由于平面几乎互相平行（退化配置），40%的测试失效。理论上在 45 度左右时，标定性能达到最佳水平。注意到在实际情况中，随着角度的增加，由于透视缩短，角点检测的精度下降，在本实验中，并未考虑到这个因素。

5.2 实际数据

本文提出的方法在笔者所在的视觉组以及微软研究院的图像组中广泛采用。这里提供一个实际样例的结果。

待标定摄像机为一现成的 6mm 镜头 PULNiXCCD 摄像机。图片分辨率为 640*480.模型平面图案有 8*8 个方格，即 256 个角点。图案尺寸为 17cm*17cm，由高质量打印机打印，并固定在玻璃板上。

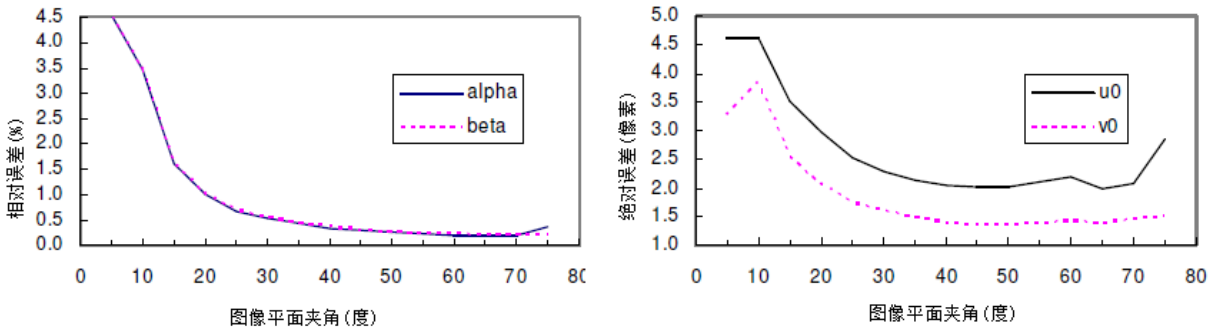


图 3 误差—模型平面与成像平面角度 折线图

nb	2 张图片			3 张图片			4 张图片			5 张图片		
	初始值	最终值	σ	初始值	最终值	σ	初始值	最终值	σ	初始值	最终值	σ
α	825.59	830.47	4.74	917.65	830.80	2.06	876.62	831.81	1.56	877.16	832.50	1.41
β	825.26	830.24	4.85	920.53	830.69	2.10	876.22	831.82	1.55	876.80	832.53	1.38
γ	0	0	0	2.2956	0.1676	0.109	0.0658	0.2867	0.095	0.1752	0.2045	0.078
u_0	295.79	307.03	1.37	277.09	305.77	1.45	301.31	304.53	0.86	301.04	303.96	0.71
v_0	217.69	206.55	0.93	223.36	206.42	1.00	220.06	206.79	0.78	220.41	206.59	0.66
k_1	0.161	−0.227	0.006	0.128	−0.229	0.006	0.145	−0.229	0.005	0.136	−0.228	0.003
k_2	−1.955	0.194	0.032	−1.986	0.196	0.034	−2.089	0.195	0.028	−2.042	0.190	0.025
RMS	0.761	0.295		0.987	0.393		0.927	0.361		0.881	0.335	

表 1 2-5 张图片的实际数据

如图 4 所示，从不同的角度拍摄 5 张模型平面的照片，可以看出图片中明显的镜头畸变。其中，角点在每个方格两两相交的直线处取得。

对前 2，3，4 以及全部的 5 张图片使用论文的标定算法，结果如表 1 所示。每组有 3 列值，第一列

(初始值)为闭合形式解。第二列(最终结果)为最大似然估计结果，第三列(σ)为标准差，代表最终结果的不确定度。由表中可以看出，闭合形式解为合理值，最终的估计结果不论是使用 2，3，4 或者 5 幅图片都非常的一致。表 1 的最后一行是实际图像点与理论投影点距离平均值的平方根。最大似然估计明显地改进了该项值。

细心的读者可能注意到闭合形式解与最大似然估计优化值中 k_1,k_2 不太一致，主要原因在于，在闭合形式解中，摄像机内参是在假定没有畸变下计算的，理论上的点将要比实际检测到的点更靠近图片中心。随后的畸变估计将尽量增加比例系数以延展外部点从而缩小误差值，而实际上畸变形状（正参数 k1，也称为枕形畸变）并不符合实际的畸变形状(负参数 k1，也叫做桶形畸变)。而通过非线性优化（MLE）矫正得到最后正确的畸变形状。有了畸变参数，便能对原始图片进行畸变矫正。图 5 展示前两幅这样矫正之后的图片，可与图 4 中前 2 幅图对比，可以清晰的看到原始图片中弯曲的图案变直了。

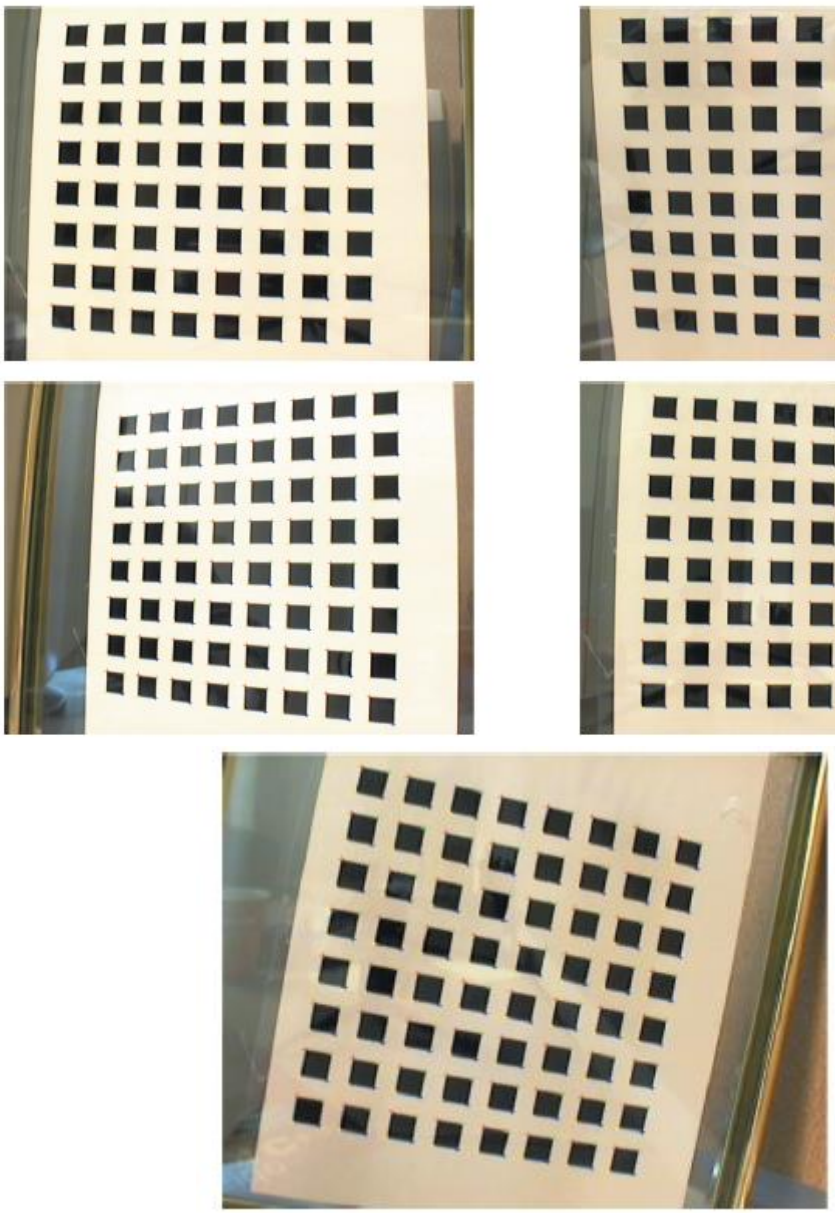


图 4 5 张模型平面，带有提取的角点(由十字叉标出)

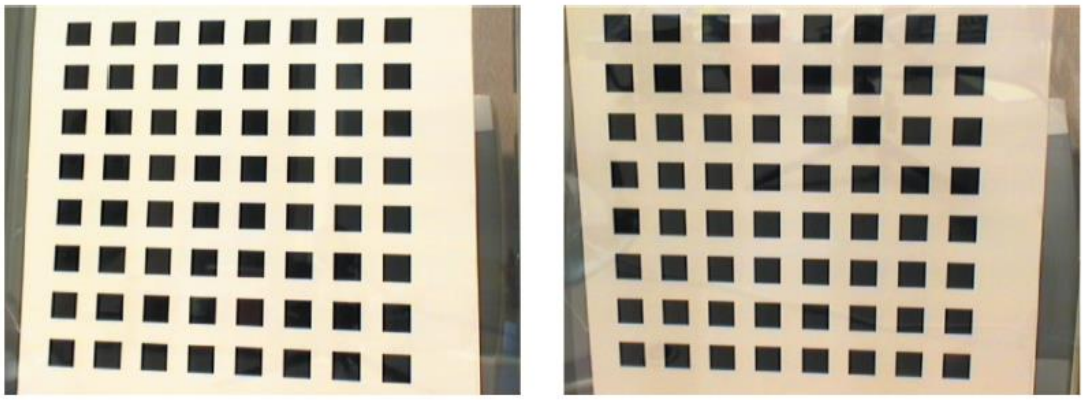


图 5 前 2 张经过径向畸变矫正后的图片

4张图片组合	(1234)	(1235)	(1245)	(1345)	(2345)	平均值	偏差
α	831.81	832.09	837.53	829.69	833.14	832.85	2.90
β	831.82	832.10	837.53	829.91	833.11	832.90	2.84
γ	0.2867	0.1069	0.0611	0.1363	0.1096	0.1401	0.086
u_0	304.53	304.32	304.57	303.95	303.53	304.18	0.44
v_0	206.79	206.23	207.30	207.16	206.33	206.76	0.48
k_1	−0.229	−0.228	−0.230	−0.227	−0.229	−0.229	0.001
k_2	0.195	0.191	0.193	0.179	0.190	0.190	0.006
RMS	0.361	0.357	0.262	0.358	0.334	0.334	0.04

表 2 所有 4 张图片组合的标定偏差

标定结果的变化 在表 1 中展示了 2 到 5 张照片的标定结果，论文发现标定结果非常的一致。为了进一步考察本文提出的算法的稳定性，论文又将该算法应用到 5 张照片所有的 4 张照片组合中。结果如表 2 所示，其中例如第三列(1235)代表第一、第二、第三、第五这四张图片。最后两列显示了这 5 组实验的均值和样本偏差。所有参数的样本偏差都非常小，由此可见，本文提出的算法非常的稳定。由于变化因子 0.086/0.1401=0.6，比较大，倾斜因子 γ 并不明显为 0，实际上， γ =0.1401， α =832.85 相当于 89.99 度，非常接近 90 度，也就是图片的两个坐标轴。论文还计算过每四幅图片 σ / β 的值，他们的平均值等于 0.99995，样本偏差为 0.00012，所以非常接近 1，也就是说，像素都是方形的。

基于图片的建模应用 论文用上文标定的摄像机拍摄了两张茶罐的图片(见图 6)，主要能看到两个侧面。在每个侧面上人工挑选了 8 个匹配的点，使用论文之前开发的[27]移动重建软件对这茶罐的 16 个匹配点进行局部建模。模型采用虚拟现实描述语言，3 张渲染后的图片如图 7 所示。每个侧面上重建的点实际上是共面的，论文还计算了两个重建的平面的角度为 94.7 度，虽然论文并不知道实际值到底是多少，

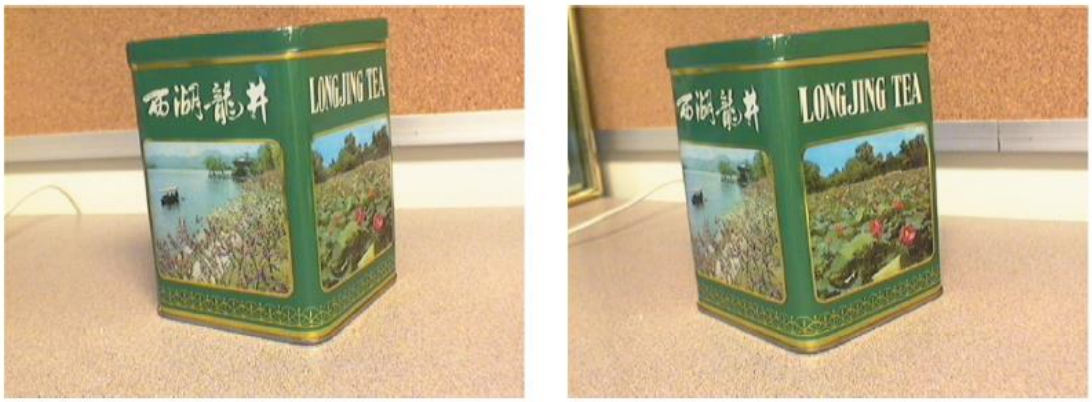


图 6 茶罐的两张图片

图 7 3 张茶罐重建的渲染图片

但是茶罐的两个侧面实际上几乎是相互垂直的。

所有的实际数据和结果可以去：http://research.microsoft.com/~zhang/Calib/ 下载

5.3 对模型误差的敏感度

如上文描述的例子一样，2D 模型平面是由高质量的打印机打印而成。尽管打印这样一张高质量的 2D 图案相对于传统的标定设备已经相当的廉价，但是假如论文使用普通的打印机打印，或者图案并不是固定在一个平坦的表面上，这个 2D 模型平面都有可能存在一些不精确之处。本节将要研究本文提出的标定算法对于模型不精确的敏感度。

5.3.1 模型点中的随机噪声

论文将进行同上一节一样的实际测试，并将 5 张照片都用于标定。为了模拟模型误差，论文在模型上每个方格的各个角点添加均值为 0 的高斯噪声。附加噪声的标准差由每个方格边长的 1%到 15%，方格的边长为 1.27cm(更精确地讲，0.5 英寸)，方格边长的 15%也即 2mm 的偏差，应该不会有人愿意用这么差的模型吧。对于每个级别的噪声，分别进行 100 次独立实验，计算平均误差（和表 1 中的真实模型的结果的偏差），

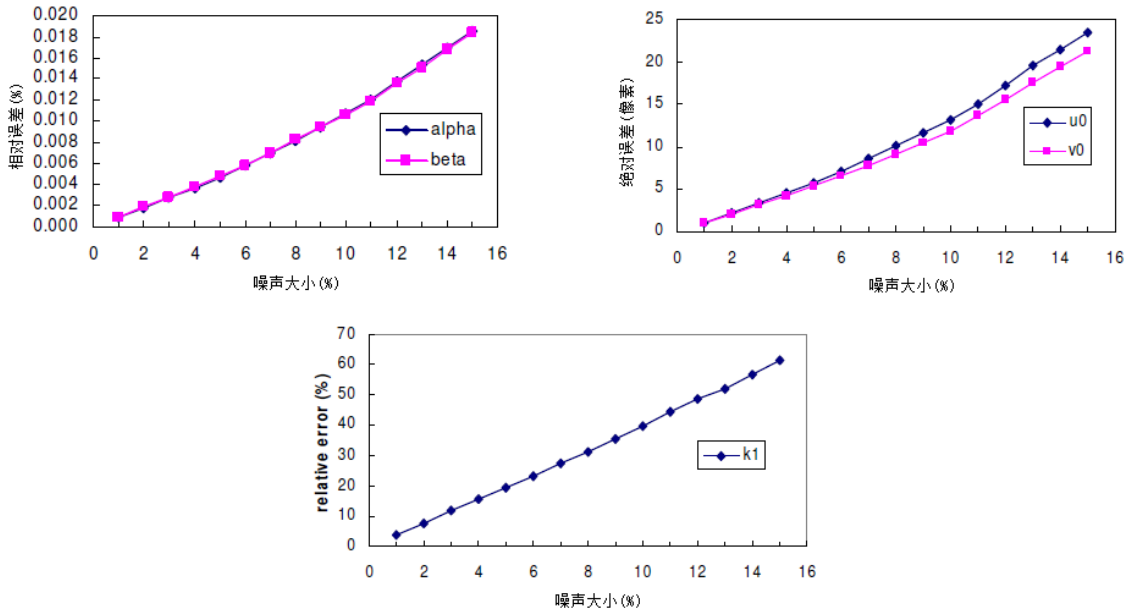


图 8 摄像机标定对模型点中高斯噪声的敏感度

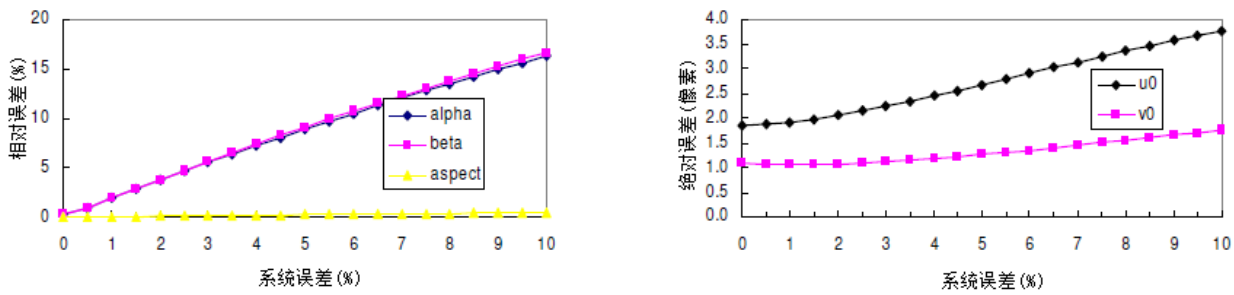


图 9 摄像机标定对系统性球形扭曲的敏感度

结果如图 8 所示。显然，随着模型点的噪声值得增强，所有参数的误差都增加了。但是像素比例因子(α和β)却依然十分稳定：误差不超过 0.02%。原点坐标同样非常稳定：噪声为 15%时，误差在 20 像素左右。径向畸变的估计值 k1 就有点不靠谱，第二畸变因素 k2(没有展示)就更不靠谱了。

在论文当前的推导中，论文假设模型平面上点提取的位置是已知的。如果模型平面上的点只是已知在一定精度的范围内，论文可以重新论述这个问题，可以得到比这里更为精确的结果。

5.3.2 模型平面系统性非平面误差

在本节中，论文将探讨模型平面系统性非平面误差，例如，当打印图案固定在一本相对较软的书封面上。实验中采用同 5.1 节相同的配置。

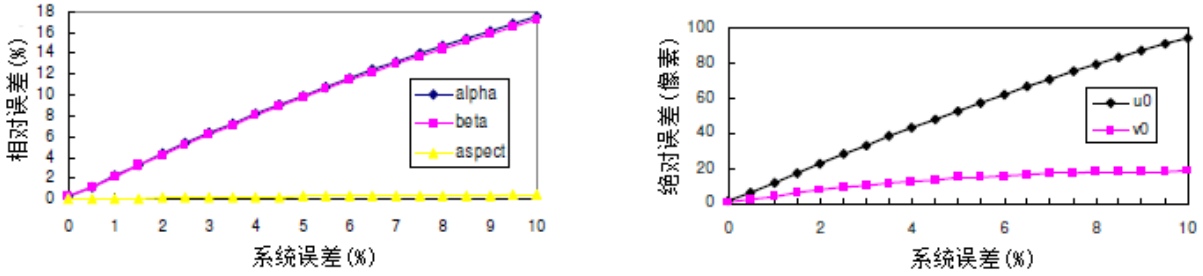


图 10 摄像机标定对系统性圆柱形扭曲的敏感度

采用两种系统性方法扭曲模型平面以模拟非平面性程度：球形和圆柱形扭曲。在球形扭曲中，图案非中心处的点的 z 坐标变成 $z=p\sqrt{x^2+y^2}$，其中 p 表示非平面程度(p=0 时，模型上的点在一个平面上)，并且扭曲关于中心点是均匀对称的。在圆柱形扭曲中，图案上的点的 z 坐标变成 z=px，其中，p 也表示非平面程度，采用这种方式模拟模型图案在竖直轴的扭曲。实验使用 4 张模型图案：第一张平行于成像平面；第二张图案由第一张图案绕水平坐标旋转 30 度；第三张由第一张图案绕竖直坐标轴旋转 30 度；第四张由第一张绕对角线轴旋转 30 度。尽管模型点并不是在一个平面上的，但论文把它当作是在一个平面上，并使用本文提出的算法处理。图片上仍然附加了标准差为 0.5 像素的高斯噪声，重复进行 100 次独立实验。球形实验的标定误差如图 9 所示，圆柱形实验的结果如图 10 所示。水平坐标轴表示非平面程度的增加，是由 z 坐标的最大偏差和图案的尺寸之比计算而来。因此，10%的非平面程度相当于 z 坐标最大偏差 2.5cm，这种情况在实际中很少发生。可以观察到：

模型的系统性非平面误差对标定精度的影响大于上一节中提到的平移随机误差。

α β 比值非常稳定（非平面程度为 10%时只有 0.4%的误差）

柱形系统非平面性比球形更离谱，尤其对原点坐标（u_0,v_0）,主要是因为柱形扭曲只关于一个坐标轴对称，这也是在论文的模拟实验中 u_0 的误差远大于 v_0 的误差的原因。

实际情况中，对于很小的系统性非平面误差（比如小于 3%），标定结果还是靠谱的

很多研究者发现，（u_0,v_0）的误差对 3D 重建的影响很小。Triggs[22]指出，（u_0,v_0）的绝对误差并没有多大几何意义，他提出测量和焦距的相对误差，比如 $\Delta u_0/\alpha$ 和 $\Delta v_0/\alpha$ 。这相当于测量理论光轴和实际光轴的夹角，那么，当柱形扭曲为 10%时（见图 10），u_0 相对误差只有 7.6%，比 α 和 β 的误差还小。

6 总结

在本文中，论文提出一种灵活易用的标定方法。该方法只需要摄像机从若干（至少 2）个不同角度拍摄平面图案。论文可以移动摄像机或者标定平面图案，并且移动不需要是已知的。该方法考虑了镜头径向畸变，主要步骤有：闭合形式解以及基于最大似然准则的非线性优化。文中采用计算机模拟数据以及真实数据来测试该算法，并取得了非常好的结果。与传统的标定技术相比，并不需要像 2-3 个正交平面这样昂贵的器材，本文提出的方法明显地提高了灵活性。

致谢

感谢 Brain Guenter 提取角点的软件，感谢他和我的多次讨论，感谢 Bill Tiggs 给我提出深刻的建议。感谢 Andrew Zisserman 与我分享他的 CVPR98 成果[14]，虽然论文使用了相同的约束，但形式却大不一样，另外，非常感谢他给我指出推导纯平移情况下的错误。还要感谢 Bill Triggs 和 Gideon Stein 向我建议 5.3 节中的实验。感谢 MSR 视觉小组所有同事对我的鼓励和支持。感谢 Anadan 和 Charles Loop 帮我审核英文。

附录

A 模型平面和对应图像的单应性矩阵估计

有很多种方式可以估算模型平面和它的图像之间的单应性矩阵。这里，论文给出一种基于最大似然准则的方案。令 M_i 和 m_i 分别表示模型平面和图像上的点。理论上他们满足等式（2），实际上由于提取的图像点存在一定的噪声，并不严格满足（2）式。假设 m_i 受高斯噪声影响，噪声均值为 0，协方差矩阵为Λ_{m_i}，于是，**H** 的最大似然估计可以通过求取函数

$$\sum_i (m_i - m_i)^T \Lambda_{mi}^{-1} (m_i - \hat{m}_i)$$

的最小值得到，其中 $\hat{m}i = \frac{1}{\bar{h}_3^T M_i}\begin{bmatrix}\bar{h}_1^T M_i \\ \bar{h}_2^T M_i\end{bmatrix}$，$\bar{h}_i$ 为 **H** 的第 i 行。

在实际计算过程中，由于角点是通过相同的步骤独立提取出来的，论文可以简单的假设对于任意的 i，$\Lambda m = \sigma^2 I$，这样，上述的问题便转化为一个非线性最小二乘问题，即最小化 $\min_H \sum_i \|m_i - \hat{m}_i\|^2$。非线性最小值由 Minpack[18]实现的 LM 算法求得。该算法需要一个初始值，可以通过如下方法获得：

令 $x = \begin{bmatrix}\bar{h}_1^T & \bar{h}_2^T & \bar{h}_3^T\end{bmatrix}^T$，（2）式可化为：

$$\begin{bmatrix}\tilde{M}^T & 0 & -u\tilde{M}^T \\ 0 & \tilde{M}^T & -v\tilde{M}^T\end{bmatrix} x = 0$$

假如给定 n 个点，就有 n 个上述等式，写成矩阵形式即 **Lx=0**,其中 **L** 为 2n*9 的矩阵，由于 x 由一个比例因子确定，该式的解即 **L** 的最小奇异值对应的奇异向量（也即 $L^T L$ 最小特征值对应的特征向量）。

在 **L** 中，有的元素为常数 1，有的是像素点，有的又是世界坐标系，有的还是两者兼有。这样使得 L 不适合数值计算，而在进行上述处理过程之前，简单地做归一化处理，如文献[12]提出的一样，将得到好得多的实验结果。

B 从 B 矩阵提取摄像机内参

在 3.1 节中提到的 **B** 矩阵，由一个比例因子确定，即 $B = \lambda A^{-T} A$，其中 λ 为任意的比例因子。论文可以非常容易的从 **B** 矩阵中解出内参参数

$$v_0 = \frac{(B_{12}B_{13} - B_{11}B_{23})}{(B_{11}B_{22} - B_{12}^2)}$$

$$\lambda = B_{33} - \frac{B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})}{B_{11}}$$

$$\alpha = \sqrt{\frac{\lambda}{B_{11}}}$$

$$\beta = \sqrt{\frac{\lambda B_{11}}{B_{11}B_{22} - B_{12}^2}}$$

$$\gamma = \frac{-B_{12}\alpha^2\beta}{\lambda}$$

$$u0 = \frac{\gamma v_0}{\beta} - \frac{B_{13}\alpha^2}{\lambda}$$

C 通过旋转矩阵逼近 3*3 矩阵

本节将要讨论的问题是，给定矩阵 **Q** 找出逼近 **Q** 的最优旋转矩阵 **R**。这里，“最优”定义为 **R-Q** 的最小 Frobenius 准则，也就相当于求解如下问题：

$$\min_R \|R-Q\|_F^2 \quad \text{满足约束 } R^TR=I \tag{15}$$

由于

$$\|R-Q\|_F^2 = trace((R-Q)^T(R-Q))$$
$$= 3+trace(Q^TQ)-2\,\mathrm{trace}(\mathrm{R^TQ})$$

问题 15 等价于最大化 $trace(R^TQ)$

令 **Q** 的奇异值分解为 USV^T，其中 $S=diag(\sigma_1,\sigma_2,\sigma_3)$,假如定义正交矩阵 **Z,** $Z=V^TR^TU$，那么

$$trace(R^TQ)=trace(R^TUSV^T)=trace(V^TR^TUS)$$
$$=trace(ZS)=\sum_{i=1}^{3}z_{ii}\sigma_i\le\sum_{i=1}^{3}\sigma_i$$

很明显令 $R=UV^T$ 则 **Z=I**，即可得到最大值。这样便得到（15）的解。

矩阵计算方面一个非常值得借鉴的是 Golub 和 vanLoan[10].

D 已知平移变换下的摄像机标定

在第 4 节论文提到，假如模型平面仅仅通过平移变换，本文提出的标定方法将失效。但是，假如变换是已知的，可以通过像 Tsai[23]方法一样进行初始化，同样可以进行摄像机标定。由(2)有，$t=\alpha A^{-1}h_3$，其中 $\alpha=1/\|A^{-1}h_1\|$，i 和 j 两点之间的变换写成如下形式：

$$t^{(ij)}=t^{(i)}-t^{(j)}=A^{-1}(\alpha^{(i)}h_3^{(i)}-\alpha^{(j)}h_3^{(j)})$$

（注意到虽然 $\mathbf{H}^{(i)}$和 $\mathbf{H}^{(j)}$都是由各自的比例因子确定，由于变换为纯平移变换，它们都可以缩放到一个相同的比例因子）只要变换的方向已知，便可以得到 **A** 矩阵的两个约束条件，如果还知道变换的大小，还能得到 **A** 矩阵的另外一个约束条件，这样便可以从两张平移的平面获得标定结果。

参考文献：

原文：

A Flexible New Technique for Camera Calibration

Abstract

We propose a flexible new technique to easily calibrate a camera. It is well suited for use without specialized knowledge of 3D geometry or computer vision. The technique only requires the camera to observe a planar pattern shown at a few (at least two) different orientations. Either the camera or the planar pattern can be freely moved. The motion need not be known. Radial lens distortion is modeled. The proposed procedure consists of a closed-form solution, followed by a nonlinear refinement based on the maximum likelihood criterion. Both computer simulation and real data have been used to test the proposed technique, and very good results have been obtained. Compared with classical techniques which use expensive equipment such as two or three orthog- onal planes, the proposed technique is easy to use and flexible. It advances 3D computer vision one step from laboratory environments to real world use.

Index Terms— Camera calibration, calibration from planes, 2D pattern, absolute conic, projective mapping, lens distortion, closed-form solution, maximum likelihood estimation, flexible setup.

1 Motivations

Camera calibration is a necessary step in 3D computer vision in order to extract metric information from 2D images. Much work has been done, starting in the photogrammetry community (see [2,4] to cite a few), and more recently in computer vision ([9, 8, 23, 7, 26, 24, 17, 6] to cite a few).We can classify those techniques roughly into two categories: photogrammetric calibration and self-calibration.

Photogrammetric calibration. Camera calibration is performed by observing a calibration object whose geometry in 3-D space is known with very good precision. Calibration can be done very efficiently [5]. The calibration object usually consists of two or three planes orthogonal to each other. Sometimes, a plane undergoing a precisely known translation is also used [23]. These approaches require an expensive calibration apparatus, and an elaborate setup.

Self-calibration. Techniques in this category do not use any calibration object. Just by moving a camera in a static scene, the rigidity of the scene provides in general two constraints [17, 15] on the cameras' internal parameters from one camera displacement by using image informa-tion alone. Therefore, if images are taken by the same camera with fixed internal parameters, correspondences between three images are sufficient to recover both the internal and external parameters which allow us to reconstruct 3-D structure up to a similarity [16, 13]. While this ap- proach is very flexible, it is not yet mature [1]. Because there are many parameters to estimate, we cannot always obtain reliable results.

Other techniques exist: vanishing points for orthogonal directions [3, 14], and calibration from pure rotation [11, 21].

Our current research is focused on a desktop vision system (DVS) since the potential for using DVSs is large. Cameras are becoming cheap and ubiquitous. A DVS aims at the general public, who are not experts in computer vision. A typical computer user will perform vision tasks only from time to time, so will not be willing to invest money for expensive equipment. Therefore, flexibility, robustness and low cost are important. The camera calibration technique described in this paper was developed with these considerations in mind.

The proposed technique only requires the camera to observe a planar pattern shown at a few (at least two) different orientations. The pattern can be printed on a laser printer and attached to a "reasonable" planar surface (e.g., a hard book cover). Either the camera or the planar pattern can be moved by hand. The motion need not be known. The proposed approach lies between the photogrammetric calibration and self-calibration, because we use 2D metric information rather than 3D or purely implicit one. Both computer simulation and real data have been used to test the proposed technique, and very good results have been obtained. Compared with classical techniques, the proposed technique is considerably more flexible. Compared with self-calibration, it gains considerable degree of robustness. We believe the new technique advances 3D computer vision one step from laboratory environments to the real world.

Note that Bill Triggs [22] recently developed a self-calibration technique from at least 5 views of a planar scene. His technique is more flexible than ours, but has difficulty to initialize. Liebowitz and

Zisserman [14] described a technique of metric rectification for perspective images of planes using metric information such as a known angle, two equal though unknown angles, and a known length ratio. They also mentioned that calibration of the internal camera parameters is possible provided at least three such rectified planes, although no experimental results were shown.

The paper is organized as follows. Section 2 describes the basic constraints from observing a single plane. Section 3 describes the calibration procedure. We start with a closed-form solution, followed by nonlinear optimization. Radial lens distortion is also modeled. Section 4 studies configurations in which the proposed calibration technique fails. It is very easy to avoid such situations in practice. Section 5 provides the experimental results. Both computer simulation and real data are used to validate the proposed technique. In the Appendix, we provides a number of details, including the techniques for estimating the homography between the model plane and its image.

2 Basic Equations

We examine the constraints on the camera's intrinsic parameters provided by observing a single plane. We start with the notation used in this paper.

2.1 Notation

A 2D point is denoted by $m = [u \quad v]^T$. A 3D point is denoted by $M = [X \quad Y \quad Z]^T$. We use $\tilde{x}$ to denote the augmented vector by adding 1 as the last element: $\tilde{m} = [u \quad v \quad 1]^T$ and $\tilde{M} = [X \quad Y \quad Z \quad 1]^T$.. A camera is modeled by the usual pinhole: the relationship between a 3D point M and its image projection **m** is given by

$$s\tilde{m} = A[R \quad t]\tilde{M} \tag{1}$$

where *s* is an arbitrary scale factor, (**R**; **t**), called the extrinsic parameters, is the rotation and translation which relates the world coordinate system to the camera coordinate system, and **A**, called the camera intrinsic matrix, is given by

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

with (u_0, v_0) the coordinates of the principal point, α and β the scale factors in image *u* and *v* axes, and γ the parameter describing the skewness of the two image axes.

We use the abbreviation A^{-T} for $(A^T)^{-1}$ or $(A^{-1})^T$.

2.2 Homography between the model plane and its image

Without loss of generality, we assume the model plane is on $Z = 0$ of the world coordinate system. Let's denote the i_{th} column of the rotation matrix **R** by $\mathbf{r}_i$. From (1), we have

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A[r_1 \quad r_2 \quad r_3 \quad t]\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$= A[r_1 \quad r_2 \quad t]\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

By abuse of notation, we still use M to denote a point on the model plane, but $M = [X \quad Y]^T$ since *Z* is always equal to 0. In turn $\tilde{M} = [X \quad Y \quad 1]^T$. Therefore, a model point M and its image **m** is related by a homography **H**:

$$s\tilde{m} = H\tilde{M} \quad with \quad H = A[r_1 \quad r_2 \quad t] \tag{2}$$

As is clear,the 3x3 matrix **H** is defined up to a scale factor.

2.3 Constraints on the intrinsic parameters

Given an image of the model plane, an homography can be estimated (see Appendix A). Let's denote it by **H** =[**h**1 **h**2 **h**3]. From (2), we have

$$[h_1 \quad h_2 \quad h_3]=\lambda A[r_1 \quad r_2 \quad t]$$

where λ is an arbitrary scalar. Using the knowledge that **r**1 and **r**2 are orthonormal, we have

$$h_1^T A^{-T} A^{-1} h_2 = 0 \tag{3}$$

$$h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \tag{4}$$

These are the two basic constraints on the intrinsic parameters, given one homography. Because a homography has 8 degrees of freedom and there are 6 extrinsic parameters (3 for rotation and 3 for translation), we can only obtain 2 constraints on the intrinsic parameters. Note that $A^{-T}A^{-1}$ actually describes the image of the absolute conic [16]. In the next subsection, we will give an geometric interpretation.

2.4 Geometric Interpretation

We are now relating (3) and (4) to the absolute conic.It is not difficult to verify that the model plane, under our convention, is described in the camera coordinate system by the following equation:

$$\begin{bmatrix} r_3 \\ r_3^T t \end{bmatrix}^T \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = 0$$

where w=*0* for points at infinity and *w=1* otherwise. This plane intersects the plane at infinity at a line, and we can easily see that $\begin{bmatrix} r_1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} r_2 \\ 0 \end{bmatrix}$ are two particular points on that line. Any point on it is a linear combination of these two points, i.e.:

$$x_\infty = a\begin{bmatrix} r_1 \\ 0 \end{bmatrix} + b\begin{bmatrix} r_2 \\ 0 \end{bmatrix} = \begin{bmatrix} ar_1 + br_2 \\ 0 \end{bmatrix}$$

Now, let's compute the intersection of the above line with the absolute conic. By definition, the point x_∞, known as the *circular point*, satisfies: $x_\infty^T x_\infty = 0$ i.e.:

$$(ar_1 + br_2)^T (ar_1 + br_2) = 0 \ \ or \ \ a^2 + b^2 = 0$$

The solution is $b = \pm ai$, where $i^2 = -1$. That is, the two intersection points are

$$x_\infty = a\begin{bmatrix} r_1 \pm ir_2 \\ 0 \end{bmatrix}$$

Their projection in the image plane is then given, up to a scale factor, by

$$\tilde{m}_\infty = A(r_1 \pm ir_2) = (h_1 \pm ih_2)$$

Point $\tilde{m}_\infty$ is on the image of the absolute conic, described by $A^{-T}A^{-1}$ [16]. This gives

$$(h_1 \pm ih2)^T A^{-T} A^{-1} (h_1 \pm ih_2) = 0$$

Requiring that both real and imaginary parts be zero yields (3) and (4).

3 Solving Camera Calibration

This section provides the details how to effectively solve the camera calibration problem. We start with an analytical solution, followed by a nonlinear optimization technique based on the maximum likelihood criterion. Finally, we take into account lens distortion, giving both analytical and nonlinear solutions.

3.1 Closed-form solution

Let

$$B = A^{-T}A^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix} \quad (5)$$

Note that **B** is symmetric, defined by a 6D vector

$$b = \begin{bmatrix} B_{11} & B_{12} & B_{22} & B_{13} & B_{21} & B_{33} \end{bmatrix}^T \quad (6)$$

Let the i^{th} column vector of **H** be $h_i = \begin{bmatrix} h_{i1} & h_{i2} & h_{i3} \end{bmatrix}^T$ Then, we have

$$h_i^T B h_j = v_{ij}^T b \quad (7)$$

with

$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2} + h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}^T$$

Therefore, the two fundamental constraints (3) and (4), from a given homography, can be rewritten as 2 homogeneous equations in **b**:

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \quad (8)$$

If *n* images of the model plane are observed, by stacking *n* such equations as (8) we have

$$Vb = 0 \quad (9)$$

where **V** is a $2n \times 6$ matrix. If $n \geq 3$, we will have in general a unique solution **b** defined up to a scale factor. If $n = 2$, we can impose the skewless constraint $\gamma = 0$, i.e., $[0, 1, 0, 0, 0, 0]\mathbf{b} = 0$, which is added as an additional equation to (9). (If $n = 1$, we can only solve two camera intrinsic parameters, e.g., α and β, assuming u_0 and v_0 are known (e.g., at the image center) and $\gamma = 0$, and that is indeed what we did in [19] for head pose determination based on the fact that eyes and mouth are reasonably coplanar.) The solution to (9) is well known as the eigenvector of V^TV associated with the smallest eigenvalue (equivalently, the right singular vector of **V** associated with the smallest singular value). Once **b** is estimated, we can compute all camera intrinsic matrix **A**. See Appendix B for the details. Once **A** is known, the extrinsic parameters for each image is readily computed. From (2), we have

$$r_1 = \lambda A^{-1}h_1$$
$$r_2 = \lambda A^{-1}h_2$$
$$r_3 = r_1 \times r_2$$
$$t = \lambda A^{-1}h_3$$

with $\lambda = 1/\left\|A^{-1}h_1\right\| = 1/\left\|A^{-1}h_2\right\|$ Of course, because of noise in data, the so-computed matrix $R = \begin{bmatrix} r_1, r_2, r_3 \end{bmatrix}$ does not in general satisfy the properties of a rotation matrix. Appendix C describes a method to estimate the best rotation matrix from a general 3×3 matrix.

3.2 Maximum likelihood estimation

The above solution is obtained through minimizing an algebraic distance which is not physically meaningful. We can refine it through maximum likelihood inference.

We are given *n* images of a model plane and there are *m* points on the model plane. Assume that the image points are corrupted by independent and identically distributed noise. The maximum likelihood estimate can be obtained by minimizing the following functional:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|m_{ij}-\tilde{m}(A,R_i,t_i,M_j)\right\|^2 \tag{10}$$

where $\tilde{m}(A,R_i,t_i,M_j)$ is the projection of point M*j* in image *i*, according to equation (2). A rotation **R** is parameterized by a vector of 3 parameters, denoted by **r**, which is parallel to the rotation axis and whose magnitude is equal to the rotation angle. **R** and **r** are related by the Rodrigues formula [5]. Minimizing (10) is a nonlinear minimization problem, which is solved with the Levenberg-Marquardt Algorithm as implemented in Minpack [18]. It requires an initial guess of **A**, $\{R_i,t_i \mid i=1..n\}$ which can be obtained using the technique described in the previous subsection.

3.3 Dealing with radial distortion

Up to now, we have not considered lens distortion of a camera. However, a desktop camera usually exhibits significant lens distortion, especially radial distortion. In this section, we only consider the first two terms of radial distortion. The reader is referred to [20, 2, 4, 26] for more elaborated models. Based on the reports in the literature [2, 23, 25], it is likely that the distortion function is totally dominated by the radial components, and especially dominated by the first term. It has also been found that any more elaborated modeling not only would not help (negligible when compared with sensor quantization), but also would cause numerical instability [23, 25].

Let (u,v) be the ideal (nonobservable distortion-free) pixel image coordinates, and $(\tilde{u},\tilde{v})$ the corresponding real observed image coordinates. The ideal points are the projection of the model points according to the pinhole model. Similarly, (x,y) and $(\tilde{x},\tilde{y})$ are the ideal (distortion-free) and real (distorted) normalized image coordinates. We have [2, 25]

$$\breve{x}=x+x\left[k_1(x^2+y^2)+k_2(x^2+y^2)^2\right]$$
$$\breve{y}=y+y\left[k_1(x^2+y^2)+k_2(x^2+y^2)^2\right]$$

where *k*1 and *k*2 are the coefficients of the radial distortion. The center of the radial distortion is the same as the principal point. From $\breve{u}=u_0+a\breve{x}+\gamma\breve{y}$ and $\breve{v}=v_0+\beta\breve{y}$ and assuming $\gamma=0$, we have

$$\tilde{\mathrm{u}}=\mathrm{u}+(\mathrm{u}-\mathrm{u}_0)[k_1(x^2+y^2)+k_2(x^2+y^2)^2] \tag{11}$$
$$\tilde{\mathrm{v}}=v+(\mathrm{v}-v_0)[k_1(x^2+y^2)+k_2(x^2+y^2)^2] \tag{12}$$

Estimating Radial Distortion by Alternation. As the radial distortion is expected to be small, one would expect to estimate the other five intrinsic parameters, using the technique described in Sect. 3.2, reasonable well by simply ignoring distortion. One strategy is then to estimate *k*1 and *k*2 after having estimated the other parameters, which will give us the ideal pixel coordinates (*u, v*). Then, from (11) and (12), we have two equations for each point in each image:

$$\begin{bmatrix}(\mathrm{u}-\mathrm{u}_0)(x^2+y^2) & (\mathrm{u}-\mathrm{u}_0)(x^2+y^2)^2\\ (\mathrm{v}-v_0)(x^2+y^2) & (\mathrm{v}-v_0)(x^2+y^2)^2\end{bmatrix}\begin{bmatrix}k_1\\k_2\end{bmatrix}=\begin{bmatrix}\tilde{u}-u\\ \tilde{v}-v\end{bmatrix}$$

Given *m* points in *n* images, we can stack all equations together to obtain in total $2mn$ equations, or in matrix form as $Dk=d$, where $k=\begin{bmatrix}k_1 & k_2\end{bmatrix}^T$. The linear least-squares solution is given by

$$k=(D^TD)^{-1}D^Td \tag{13}$$

Once *k*1 and *k*2 are estimated, one can refine the estimate of the other parameters by solving (10) with $\tilde{m}(A,R_i,t_i,M_j)$ replaced by (11) and (12). We can alternate these two procedures until convergence.

Complete Maximum Likelihood Estimation. Experimentally, we found the convergence of the above alternation technique is slow. A natural extension to (10) is then to estimate the complete set of parameters by minimizing the following functional:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|m_{ij}-\tilde{m}(A,k1,k_2,R_i,t_i,M_j)\right\|^2 \tag{14}$$

where $\tilde{m}(A,k1,k_2,R_i,t_i,M_j)$ is the projection of point M$_j$ in image i according to equation (2), followed by distortion according to (11) and (12). This is a nonlinear minimization problem, which is solved with the Levenberg-Marquardt Algorithm as implemented in Minpack [18]. A rotation is again parameterized by a 3-vector **r**, as in Sect. 3.2. An initial guess of **A** and $\{R_i,t_i \mid i=1..n\}$ can be obtained using the technique described in Sect. 3.1 or in Sect. 3.2. An initial guess of k_1 and k_2 can be obtained with the technique described in the last paragraph, or simply by setting them to 0.

3.4 Summary

The recommended calibration procedure is as follows:

1. Print a pattern and attach it to a planar surface;
2. Take a few images of the model plane under different orientations by moving either the plane or the camera;
3. Detect the feature points in the images;
4. Estimate the five intrinsic parameters and all the extrinsic parameters using the closed-form solution as described in Sect. 3.1;
5. Estimate the coefficients of the radial distortion by solving the linear least-squares (13);
6. Refine all parameters by minimizing (14).

4 Degenerate Configurations

We study in this section configurations in which additional images do not provide more constraints on the camera intrinsic parameters. Because (3) and (4) are derived from the properties of the rotation matrix, if $\mathbf{R}_2$ is not independent of $\mathbf{R}_1$, then image 2 does not provide additional constraints. In particular, if a plane undergoes a pure translation, then $\mathbf{R}_2 = \mathbf{R}_1$ and image 2 is not helpful for camera calibration. In the following, we consider a more complex configuration.

Proposition 1. *If the model plane at the second position is parallel to its first position, then the second homography does not provide additional constraints.*

Proof. Under our convention, $\mathbf{R}_2$ and $\mathbf{R}_1$ are related by a rotation around z-axis. That is,

$$R_1\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}=R_2$$

where θ is the angle of the relative rotation. We will use superscript (1) and (2) to denote vectors related to image 1 and 2, respectively. It is clear that we have

$$h_1^{(2)}=\lambda^{(2)}(Ar^{(1)}\cos\theta+Ar^{(2)}\sin\theta)=\frac{\lambda^{(2)}}{\lambda^{(1)}}h_1^{(1)}(\cos\theta+h_2^{(1)}\sin\theta)$$

$$h_2^{(2)}=\lambda^{(2)}(-Ar^{(1)}\sin\theta+Ar^{(2)}\cos\theta)=\frac{\lambda^{(2)}}{\lambda^{(1)}}h_1^{(1)}(-\sin\theta+h_2^{(1)}\cos\theta)$$

Then, the first constraint (3) from image 2 becomes:

$$h_1^{(2)T}A^{-T}A^{-1}h_2^{(2)}=\frac{\lambda^{(2)}}{\lambda^{(1)}}[(\cos^2\theta-\sin^2\theta)(h_1^{(1)T}A^{-T}A^{-1}h_2^{(1)})$$

$$-\cos\theta\sin\theta(h_1^{(1)T}A^{-T}A^{-1}h_1^{(1)}-h_2^{(1)T}A^{-T}A^{-1}h_2^{(1)})],$$

which is a linear combination of the two constraints provided by $\mathbf{H}_1$. Similarly, we can show that the second constraint from image 2 is also a linear combination of the two constraints provided by $\mathbf{H}_1$. Therefore, we do not gain any constraint from $\mathbf{H}_2$.

The result is self-evident because parallel planes intersect with the plane at infinity at the *same ircular points*, and thus according to Sect. 2.4 they provide the same constraints.

In practice, it is very easy to avoid the degenerate configuration: we only need to change the orientation of the model plane from one snapshot to another.

Although the proposed technique will not work if the model plane undergoes pure translation, camera calibration is still possible if the translation is known. Please refer to Appendix D.

5 Experimental Results

The proposed algorithm has been tested on both computer simulated data and real data. The closedform solution involves finding a singular value decomposition of a small $2n \times 6$ matrix, where n is the number of images. The nonlinear refinement within the Levenberg-Marquardt algorithm takes 3 to 5 iterations to converge.

5.1 Computer Simulations

The simulated camera has the following property: $\alpha = 1250$, $\beta = 900$, $\gamma = 1.09083$ (equivalent to 89.95°), $u_0 = 255$, $v_0 = 255$. The image resolution is 512×512. The model plane is a checker pattern containing $10 \times 14 = 140$ corner points (so we usually have more data in the v direction than in the u direction). The size of pattern is 18cm×25cm. The orientation of the plane is represented by a 3D vector **r**, which is parallel to the rotation axis and whose magnitude is equal to the rotation angle. Its position is represented by a 3D vector **t** (unit in centimeters).

Performance w.r.t. the noise level. In this experiment, we use three planes with $r_1 = \left[20^\circ, 0, 0\right]^T$, $t_1 = \left[-9, -12.5, 500\right]^T$, $r_2 = \left[0, 20^\circ, 0\right]^T$, $t_2 = \left[-9, -12.5, 510\right]^T$, $r_3 = \frac{1}{\sqrt{5}}\left[-30^\circ, -30^\circ, -15^\circ\right]^T$ $t_3 = \left[-10.5, -12.5, 525\right]^T$. Gaussian noise with 0 mean and σ standard deviation is added to the projected image points. The estimated camera parameters are then compared with the ground truth. We measure the relative error for α and β , and absolute error for u_0 and v_0. We vary the noise level from 0.1 pixels to 1.5 pixels. For each noise level, we perform 100 independent trials, and the results shown are the average. As we can see from Fig. 1, errors increase linearly with the noise level. (The error for γ is not shown, but has the same property.) For $\sigma = 0.5$ (which is larger than the normal noise in practical calibration), the errors in α and β are less than 0.3%, and the errors in u_0 and v_0 are around 1 pixel. The error in u_0 is larger than that in v_0. The main reason is that there are less data in the u direction than in the v direction, as we said before.

Performance w.r.t. the number of planes. This experiment investigates the performance with respect to the number of planes (more precisely, the number of images of the model plane). The orientation and position of the model plane for the first three images are the same as in the last subsection. From the fourth image, we first randomly choose a rotation axis in a uniform sphere, then apply a rotation angle of 30°. We vary the number of images from 2 to 16. For each number, 100 trials of independent plane orientations (except for the first three) and independent noise with mean 0 and standard deviation 0.5 pixels are conducted. The average result is shown in Fig. 2. The errors decrease
when more images are used. From 2 to 3, the errors decrease significantly.

Performance w.r.t. the orientation of the model plane. This experiment examines the influence of the orientation of the model plane with respect to the image plane. Three images are used. The orientation of the plane is chosen as follows: the plane is initially parallel to the image plane; a rotation axis is randomly chosen from a uniform sphere; the plane is then rotated around that axis with angle θ. Gaussian noise with mean 0 and standard deviation 0.5 pixels is added to the projected image points. We repeat this process 100 times and compute the average errors. The angle θ varies from 5° to 75°, and the result is shown in Fig. 3. When θ=5°, 40% of the trials failed because the planes are almost parallel to each other (degenerate configuration), and the result shown has excluded those trials. Best performance seems to be achieved with an angle around 45°. Note that in practice, when the angle increases, foreshortening makes

the corner detection less precise, but this is not considered in this experiment.

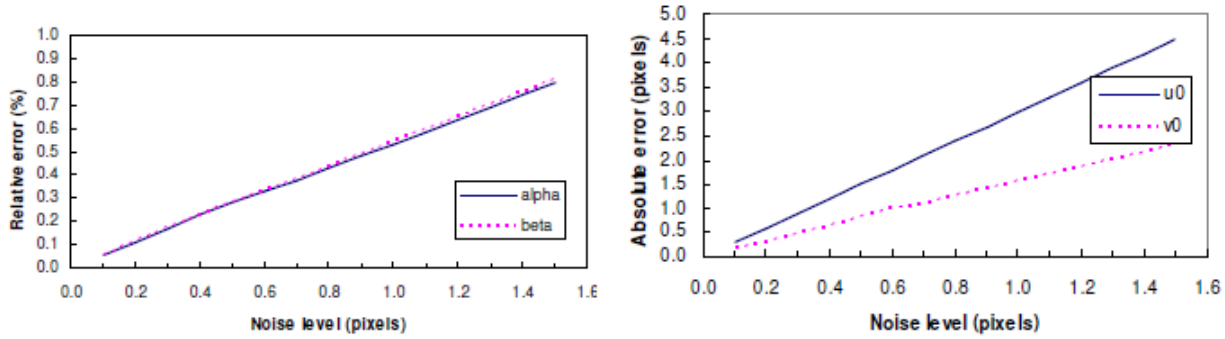


Figure 1: Errors vs. the noise level of the image points

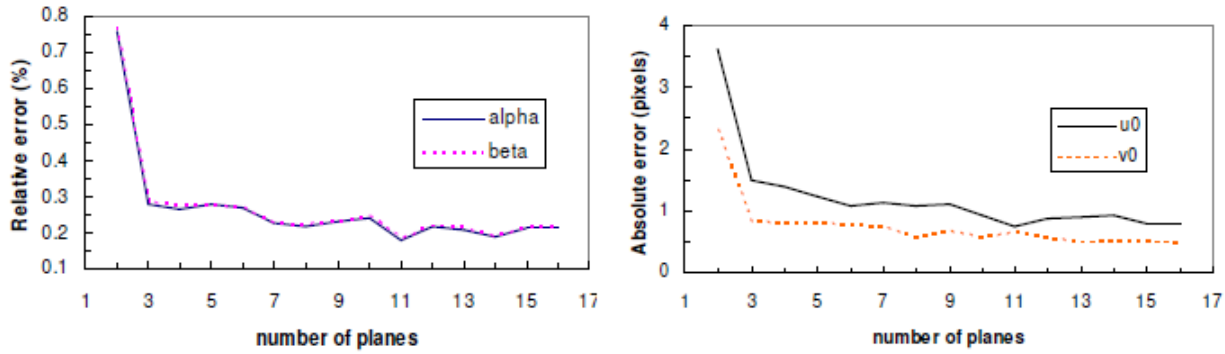


Figure 2: Errors vs. the number of images of the model plane

5.2 Real Data

The proposed technique is now routinely used in our vision group and also in the graphics group at Microsoft Research. Here, we provide the result with one example.

The camera to be calibrated is an off-the-shelf PULNiX CCD camera with 6 mm lens. The image resolution is 640 $\times$480. The model plane contains a pattern of 8×8 squares, so there are 256 corners. The size of the pattern is 17cm $\times$17cm. It was printed with a high-quality printer and put on a glass.

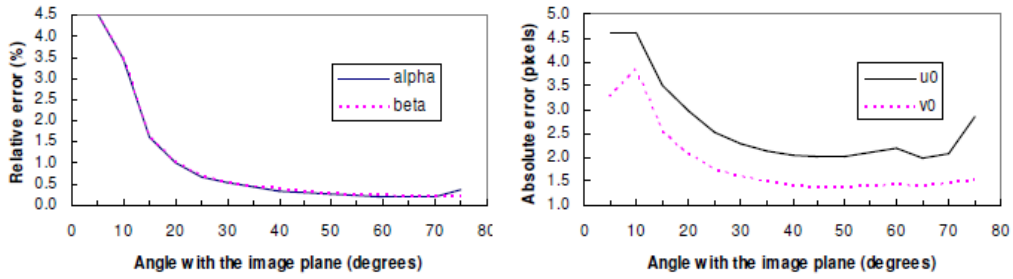


Figure 3: Errors vs. the angle of the model plane w.r.t. the image plane

Table 1: Results with real data of 2 through 5 images

nb	2 images			3 images			4 images			5 images		
	initial	final	σ	initial	final	σ	initial	final	σ	initial	final	σ
α	825.59	830.47	4.74	917.65	830.80	2.06	876.62	831.81	1.56	877.16	832.50	1.41
β	825.26	830.24	4.85	920.53	830.69	2.10	876.22	831.82	1.55	876.80	832.53	1.38
γ	0	0	0	2.2956	0.1676	0.109	0.0658	0.2867	0.095	0.1752	0.2045	0.078
u_0	295.79	307.03	1.37	277.09	305.77	1.45	301.31	304.53	0.86	301.04	303.96	0.71
v_0	217.69	206.55	0.93	223.36	206.42	1.00	220.06	206.79	0.78	220.41	206.59	0.66
k_1	0.161	−0.227	0.006	0.128	−0.229	0.006	0.145	−0.229	0.005	0.136	−0.228	0.003
k_2	−1.955	0.194	0.032	−1.986	0.196	0.034	−2.089	0.195	0.028	−2.042	0.190	0.025
RMS	0.761	0.295		0.987	0.393		0.927	0.361		0.881	0.335	

Five images of the plane under different orientations were taken, as shown in Fig. 4. We can observe a significant lens distortion in the images. The corners were detected as the intersection of straight lines fitted to each square.

We applied our calibration algorithm to the first 2, 3, 4 and all 5 images. The results are shown in Table 1. For each configuration, three columns are given. The first column (initial) is the estimation of the closed-form solution. The second column (final) is the maximum likelihood estimation (MLE), and the third column (σ) is the estimated standard deviation, representing the uncertainty of the final result. As is clear, the closed-form solution is reasonable, and the final estimates are very consistent with each other whether we use 2, 3, 4 or 5 images. We also note that the uncertainty of the final estimate decreases with the number of images. The last row of Table 1, indicated by RMS, displays the root of mean squared distances, in pixels, between detected image points and projected ones. The MLE improves considerably this measure.

The careful reader may remark the inconsistency for k_1 and k_2 between the closed-form solution and the MLE. The reason is that for the closed-form solution, camera intrinsic parameters are estimated assuming no distortion, and the predicted outer points lie closer to the image center than the detected ones. The subsequent distortion estimation tries to spread the outer points and increase the scale in order to reduce the distances, although the distortion shape (with positive k_1, called pincushion distortion) does not correspond to the real distortion (with negative k_1, called barrel distortion). The nonlinear refinement (MLE) finally recovers the correct distortion shape. The estimated distortion parameters allow us to correct the distortion in the original images. Figure 5 displays the first two such distortion-corrected images, which should be compared with the first two images shown in Figure 4. We see clearly that the curved pattern in the original images is straightened.

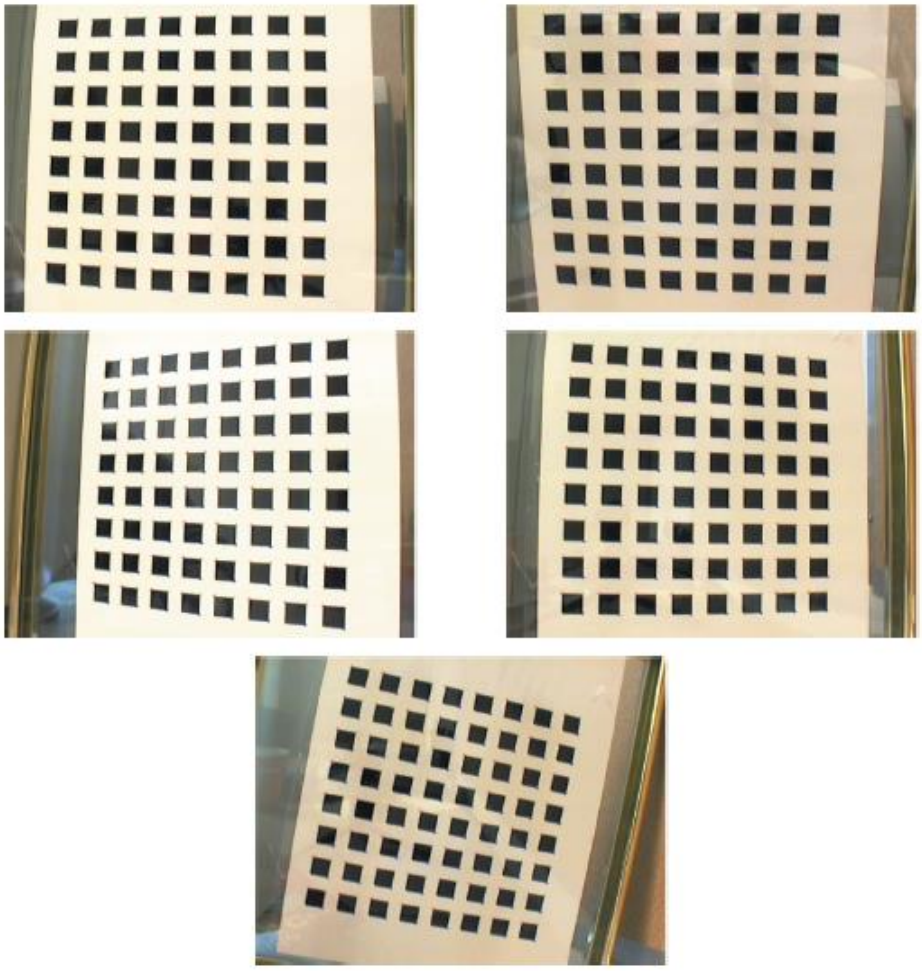


Figure 4: Five images of a model plane, together with the extracted corners (indicated by cross)

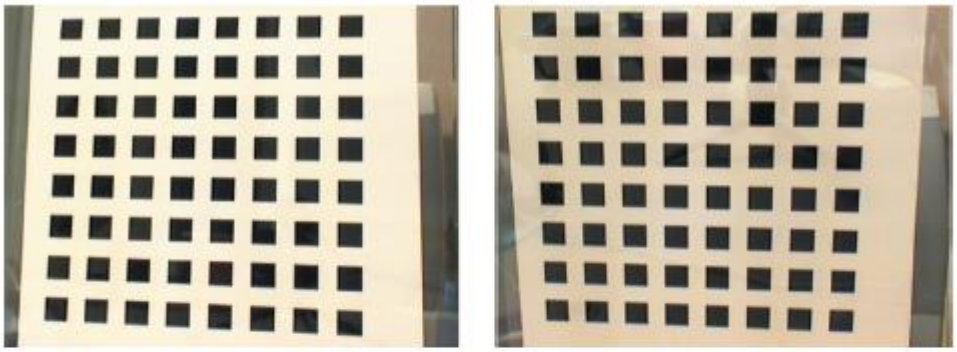


Figure 5: First and second images after having corrected radial distortion

Table 2: Variation of the calibration results among all quadruples of images

quadruple	(1234)	(1235)	(1245)	(1345)	(2345)	mean	deviation
α	831.81	832.09	837.53	829.69	833.14	832.85	2.90
β	831.82	832.10	837.53	829.91	833.11	832.90	2.84
γ	0.2867	0.1069	0.0611	0.1363	0.1096	0.1401	0.086
u_0	304.53	304.32	304.57	303.95	303.53	304.18	0.44
v_0	206.79	206.23	207.30	207.16	206.33	206.76	0.48
k_1	−0.229	−0.228	−0.230	−0.227	−0.229	−0.229	0.001
k_2	0.195	0.191	0.193	0.179	0.190	0.190	0.006
RMS	0.361	0.357	0.262	0.358	0.334	0.334	0.04

Variation of the calibration result. In Table 1, we have shown the calibration results with 2 through 5 images, and we have found that the results are very consistent with each other. In order to further investigate the stability of the proposed algorithm, we have applied it to all combinations of 4 images from the available 5 images. The results are shown in Table 2, where the third column (1235), for example, displays the result with the quadruple of the first, second, third, and fifth image. The last two columns display the mean and sample deviation of the five sets of results. The sample deviations for all parameters are quite small, which implies that the proposed algorithm is quite stable. The value of the skew parameter γ is not significant from 0, since the coefficient of variation, 0.086/0.1401 = 0.6, is large. Indeed, $\gamma = 0{:}1401$ with $\alpha = 832{:}85$ corresponds to 89.99 degrees, very close to 90 degrees, for the angle between the two image axes. We have also computed the aspect ratio α/β for each quadruple. The mean of the aspect ratio is equal to 0.99995 with sample deviation 0.00012. It is therefore very close to 1, i.e., the pixels are square.

Application to image-based modeling. Two images of a tea tin (see Fig. 6) were taken by the same camera as used above for calibration. Mainly two sides are visible. We manually picked 8 point matches on each side, and the structure-from-motion software we developed ealier [27] was run on these 16 point matches to build a partial model of the tea tin. The reconstructed model is in VRML, and three rendered views are shown in Fig. 7. The reconstructed points on each side are indeed coplanar, and we computed the angle between the two reconstructed planes which is 94.7 ◦. Although we do not have the ground truth, but the two sides of the tea tin are indeed almost orthogonal to each other. All the real data and results are available from the following Web page: http://research.microsoft.com/˜zhang/Calib/

Figure 6: Two images of a tea tin

Figure 7: Three rendered views of the reconstructed tea tin

5.3 Sensitivity with Respect to Model Imprecision

In the example described above, the 2D model pattern was printed on a paper with a high-quality printer. Although it is significantly cheaper to make such a high-quality 2D pattern than the classical calibration equipment, it is possible that there is some imprecision on the 2D model pattern if we print it on a normal printer, or the pattern is not on a flat surface. This section investigates the sensitivity of the proposed calibration technique with respect to model imprecision.

5.3.1 Random noise in the model points

We conducted this experiment on the same real data as in the last subsection. All five real images were used. To simulate model imprecision, we added Gaussian noise with zero mean to the corners of each square in the model. The standard deviation of the added noise varies from 1% to 15% of the side of each square, which is equal to 1.27cm (more precisely, 0.5inches). 15% corresponds to a standard deviation of 2mm, and people may not want to use such a poor model. For each noise level, 100 trials were conducted, and average errors (deviations from the results obtained with the true model as shown in Table 1) were alculated, and are depicted in Fig. 8. Obviously, all errors increase with the level of noise added to the model points. The pixel scale factors (α and β) remain very stable: the error is less than 0.02%. The coordinates of the principal point are quite stable: the errors are about 20 pixels for the noise level 15%. The estimated radial distortion coefficient k_1 becomes less useful, and the second term k_2 (not shown) is even less than k_1. In our current formulation, we assume that the exact position of the points in the model plane is known. If the model points are only known within certain precision, we can reformulate the problem and we could expect smaller errors than reported here.

5.3.2 Systematic non-planarity of the model pattern

In this section, we consider systematic non-planarity of the model pattern, e.g., when a printed pattern is attached to a soft book cover. We used the same configuration as in Sect. 5.1. The model planewas istorted in two systematic ways to simulate the non-planarity: spherical and cylindrical. With spherical distortion, points away from the center of the pattern are displaced in z according to $z = p\sqrt{x^2 + y^2}$, where p indicates the non-planarity (the model points are coplanar when $p = 0$). The displacement is

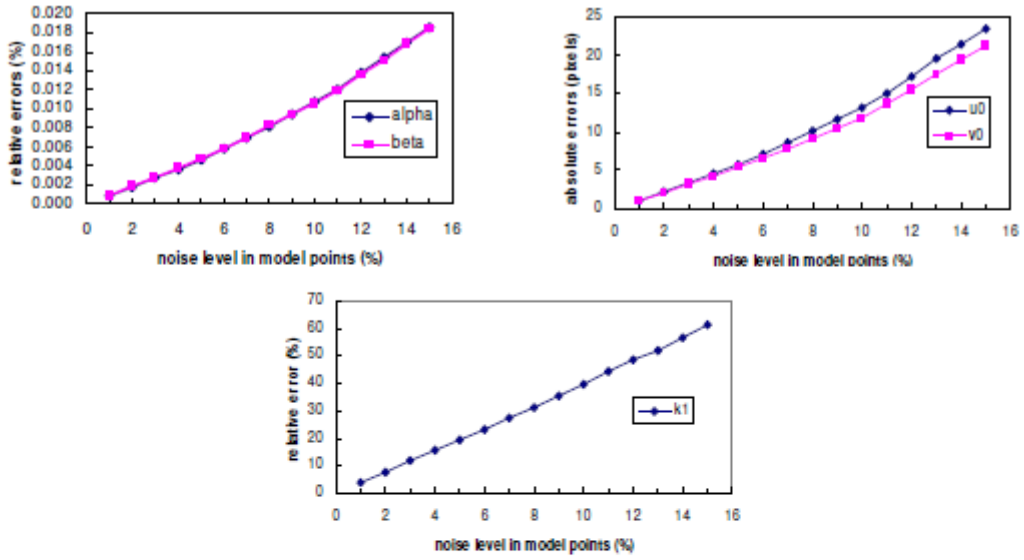


Figure 8: Sensitivity of camera calibration with respect to Gaussian noise in the model points

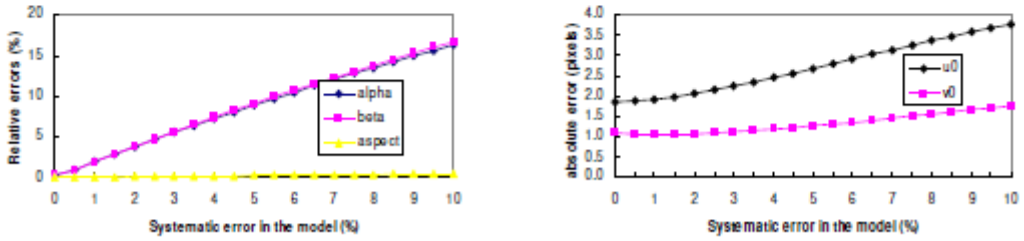


Figure 9: Sensitivity of camera calibration with respect to systematic spherical non-planarity

symmetric around the center. With Cylindrical distortion, points are displaced in z according to $z = px$. Again, p indicates the non-planarity. This simulates bending of the model pattern around the vertical axis. Four images of the model pattern were used: the first is parallel to the image plane; the second is rotated from the first around the horizontal axis by 30 degrees; the third is rotated from the first around the vertical axis by 30 degrees; the fourth is rotated from the first around the diagonal axis by 30 degrees. Although model points are not coplanar, they were treated as coplanar, and the proposed calibration technique was applied. Gaussian noise with standard deviation 0.5 pixels was added to the image points, and 100 independent trials were conducted. The average calibration errors of the 100 trials are shown in Fig. 9 for spherical non-planarity and in Fig. 10 for cylindrical non-planarity. The horizontal axis indicates the increase in the non-planarity, which is measured as the ratio of the maximum z displacement to the size of the pattern. Therefore, 10% of non-planarity is equivalent to maximum 2.5cm of displacement in z, which does not likely happen in practice. Several observations can be made:

- Systematic non-planarity of the model has more effect on the calibration precision than random errors in the positions as described in the last subsection;
- Aspect ratio is very stable (0.4% of error for 10% of non-planarity);
- Systematic cylindrical non-planarity is worse than systematic spherical non-planarity, especially for the coordinates of the principal point $(u_0; v_0)$. The reason is that cylindrical nonplanarity is only symmetric in one axis. That is also why the error in u_0 is much larger than in v_0 in our simulation;
- The result seems still usable in practice if there is only a few percents (say, less than 3%) of systematic non-planarity.

The error in $(u_0; v_0)$ has been found by many researchers to have little effect in 3D reconstruction. As pointed out by Triggs in [22], the absolute error in $(u_0; v_0)$ is not geometrically meaningful. He proposes

to measure the relative error with respect to the focal length, i.e., $\Delta u_0 / \alpha$ and $\Delta v_0 / \beta$ This is equivalent to measuring the angle between the true optical axis and the estimated one. Then, for 10% of cylindrical non-planarity (see Fig. 10), the relative error for u_0 is 7.6%, comparable with those of α and β.

6 Conclusion

In this paper, we have developed a flexible new technique to easily calibrate a camera. The technique only requires the camera to observe a planar pattern from a few (at least two) different orientations. We can move either the camera or the planar pattern. The motion does not need to be known. Radial lens distortion is modeled. The proposed procedure consists of a closed-form solution, followed by a nonlinear refinement based on maximum likelihood criterion. Both computer simulation and real data have been used to test the proposed technique, and very good results have been obtained. Compared with classical techniques which use expensive equipment such as two or three orthogonal planes, the proposed technique gains considerable flexibility.

Acknowledgment

Thanks go to Brian Guenter for his software of corner extraction and for many discussions, and to Bill Triggs for insightful comments. Thanks go to Andrew Zisserman for bringing his CVPR98 work [14] to my attention, which uses the same constraint but in different form, and for pointing out an error in my discussion on the case of pure translation. Thanks go to Bill Triggs and Gideon Stein for suggesting experiments described in Sect. 5.3. Thanks also go to the members of the Vision Group at MSR for encouragement and discussions. Anandan and Charles Loop have checked the English.

A Estimation of the Homography Between the Model Plane and its Image

There are many ways to estimate the homography between the model plane and its image. Here, we present a technique based on maximum likelihood criterion. Let M*i* and **m***i* be the model and image points, respectively. Ideally, they should satisfy (2). In practice, they don't because of noise in the extracted image points. Let's assume that **m***i* is corrupted by Gaussian noise with mean **0** and covariance matrix Λm . Then, the maximum likelihood estimation of **H** is obtained

$$\sum_i (m_i - m_i)^T \Lambda_{mi}^{-1} (m_i - \hat{m}_i)$$

where $\hat{m}i = \frac{1}{\bar{h}_3^T M_i} \begin{bmatrix} \bar{h}_1^T M_i \\ \bar{h}_2^T M_i \end{bmatrix}$ with $\bar{h}_i$,the i^{th} row of **H.**

In practice, we simply assume $\Lambda m = \sigma^2 I$ for all *i*. This is reasonable if points are extracted indepen-dently with the same procedure. In this case, the above problem becomes a nonlinear least-squares one, i.e. $\min_H \sum_i \|m_i - \hat{m}_i\|^2$ The nonlinear minimization is conducted with the Levenberg-Marquardt Algorithm as implemented in Minpack [18]. This requires an initial guess, which can be obtained as follows.

Let $x = \begin{bmatrix} \bar{h}_1^T & \bar{h}_2^T & \bar{h}_3^T \end{bmatrix}^T$ Then equation (2) can be rewritten as

$$\begin{bmatrix} \tilde{M}^T & 0 & -u\tilde{M}^T \\ 0 & \tilde{M}^T & -v\tilde{M}^T \end{bmatrix} x = 0$$

When we are given *n* points, we have *n* above equations, which can be written in matrix equation as **Lx** = **0**, where **L** is a $2n \times 9$ matrix. As **x** is defined up to a scale factor, the solution is well known to be the right singular vector of **L** associated with the smallest singular value (or equivalently, the eigenvector of $L^T L$ associated with the smallest eigenvalue).

In **L**, some elements are constant 1, some are in pixels, some are in world coordinates, and some are multiplication of both. This makes **L** poorly conditioned numerically. Much better results can be obtained

by performing a simple data normalization, such as the one proposed in [12], prior to running the above procedure.

B Extraction of the Intrinsic Parameters from Matrix B

Matrix **B**, as described in Sect. 3.1, is estimated up to a scale factor, i.e., , $B=\lambda A^{-T}A$ with λ an arbitrary scale. Without difficulty, we can uniquely extract the intrinsic parameters from matrix **B**.

$$v_0=\frac{(B_{12}B_{13}-B_{11}B_{23})}{(B_{11}B_{22}-B_{12}^2)}$$

$$\lambda=B_{33}-\frac{B_{13}^2+v_0(B_{12}B_{13}-B_{11}B_{23})}{B_{11}}$$

$$\alpha=\sqrt{\frac{\lambda}{B_{11}}}$$

$$\beta=\sqrt{\frac{\lambda B_{11}}{B_{11}B_{22}-B_{12}^2}}$$

$$\gamma=\frac{-B_{12}\alpha^2\beta}{\lambda}$$

$$u0=\frac{\gamma v_0}{\beta}-\frac{B_{13}\alpha^2}{\lambda}$$

C Approximating a 3 × 3 matrix by a Rotation Matrix

The problem considered in this section is to solve the best rotation matrix **R** to approximate a given 3 × 3 matrix **Q**. Here, "best" is in the sense of the smallest Frobenius norm of the difference **R**−**Q**. That is, we are solving the following problem:

$$\min_R \|R-Q\|_F^2 \quad \text{subject to} \quad R^TR=I \tag{15}$$

Since

$$\begin{aligned}\|R-Q\|_F^2 &= trace((R-Q)^T(R-Q))\\ &= 3+trace(Q^TQ)-2\,\mathrm{trace}(\mathrm{R^TQ})\end{aligned};$$

problem (15) is equivalent to the one of maximizing $trace(R^TQ)$.

Let the singular value decomposition of **Q** be USV^T , where $S=diag(\sigma_1,\sigma_2,\sigma_3)$. If we define an orthogonal matrix **Z** by $Z=V^TR^TU$, then

$$\begin{aligned}trace(R^TQ)&=trace(R^TUSV^T)=trace(V^TR^TUS)\\ &=trace(ZS)=\sum_{i=1}^{3}z_{ii}\sigma_i\le\sum_{i=1}^{3}\sigma_i\end{aligned}$$

It is clear that the maximum is achieved by setting $R=UV^T$ because then **Z** = **I**. This gives the solution to (15).

An excellent reference on matrix computations is the one by Golub and van Loan [10].

D Camera Calibration Under Known Pure Translation

As said in Sect. 4, if the model plane undergoes a pure translation, the technique proposed in this paper will not work. However, camera calibration is possible if the translation is known like the setup in Tsai's technique [23]. From (2), we have $t=\alpha A^{-1}h_3$, where $\alpha=1/\left\|A^{-1}h_1\right\|$. The translation between two positions *i* and *j* is then given by

$$t^{(ij)} = t^{(i)} - t^{(j)} = A^{-1}(\alpha^{(i)} h_3^{(i)} - \alpha^{(j)} h_3^{(j)})$$

(Note that although both $\mathbf{H}^{(i)}$ and $\mathbf{H}^{(j)}$ are estimated up to their own scale factors, they can be rescaled up to a single common scale factor using the fact that it is a pure translation.) If only the translation direction is known, we get two constraints on **A**. If we know additionally the translation magnitude, then we have another constraint on **A**. Full calibration is then possible from two planes.

References

[1] S. Bougnoux. From projective to euclidean space under any practical situation, a criticism of self-calibration. In *Proceedings of the 6th International Conference on Computer Vision*, pages 790–796, Jan. 1998.

[2] D. C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[3] B. Caprile and V. Torre. Using Vanishing Points for Camera Calibration. *The International Journal of Computer Vision*, 4(2):127–140, Mar. 1990.

[4] W. Faig. Calibration of close-range photogrammetry systems: Mathematical formulation. *Photogrammetric Engineering and Remote Sensing*, 41(12):1479–1486, 1975.

[5] O. Faugeras. *Three-Dimensional Computer Vision: a Geometric Viewpoint*. MIT Press, 1993.

[6] O. Faugeras, T. Luong, and S. Maybank. Camera self-calibration: theory and experiments. In G.Sandini, editor, *Proc 2nd ECCV*, volume 588 of *Lecture Notes in Computer Science*, pages 321–334, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[7] O. Faugeras and G. Toscani. The calibration problem for stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 15–20, Miami Beach, FL, June 1986. IEEE.

[8] S. Ganapathy. Decomposition of transformation matrices for robot vision. *Pattern Recognition Letters*, 2:401–412, Dec. 1984.

[9] D. Gennery. Stereo-camera calibration. In *Proceedings of the 10th Image Understanding Work-shop*, pages 101–108, 1979.

[10] G. Golub and C. van Loan. *Matrix Computations*. The John Hopkins University Press, Balti-more, Maryland, 3 edition, 1996.

[11] R. Hartley. Self-calibration from multiple views with a rotating camera. In J.-O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision*, volume 800-801 of *Lecture Notes in Computer Science*, pages 471–478, Stockholm, Sweden, May 1994. Springer-Verlag.

[12] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision*, pages 1064–1070, Boston, MA, June 1995. IEEE Computer Society Press.

[13] R. I. Hartley. An algorithm for self calibration from several views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 908–912, Seattle, WA, June 1994. IEEE.

[14] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Pro-ceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488,Santa Barbara, California, June 1998. IEEE Computer Society.20

[15] Q.-T. Luong. *Matrice Fondamentale et Calibration Visuelle sur l'Environnement-Vers une plus grande autonomie des syst`emes robotiques*. PhD thesis, Universit´e de Paris-Sud, Centre d'Orsay,Dec. 1992.

[16] Q.-T. Luong and O. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *The International Journal of Computer Vision*, 22(3):261–289, 1997. [17] S. J. Maybank and O. D. Faugeras. A theory of self-calibration of a moving camera. *The International Journal of Computer Vision*, 8(2):123–152, Aug. 1992.

[18] J. More. The levenberg-marquardt algorithm, implementation and theory. In G. A. Watson, editor, *Numerical Analysis*, Lecture Notes in Mathematics 630. Springer-Verlag, 1977. [19] I. Shimizu, Z. Zhang, S. Akamatsu, and K. Deguchi. Head pose determination from one image using a generic model. In *Proceedings of the IEEE Third International Conference on Automatic Face and Gesture Recognition*, pages 100–105, Nara, Japan, Apr. 1998.

[20] C. C. Slama, editor. *Manual of Photogrammetry*. American Society of Photogrammetry, fourth edition, 1980.

[21] G. Stein. Accurate internal camera calibration using rotation, with analysis of sources of error. In *Proc. Fifth International Conference on Computer Vision*, pages 230–236, Cambridge,Massachusetts, June 1995.
[22] B. Triggs. Autocalibration from planar scenes. In *Proceedings of the 5th European Conference on Computer Vision*, pages 89–105, Freiburg, Germany, June 1998.
[23] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, Aug. 1987.
[24] G. Wei and S. Ma. A complete two-plane camera calibration method and experimental comparisons. In *Proc. Fourth International Conference on Computer Vision*, pages 439–446, Berlin,May 1993.
[25] G. Wei and S. Ma. Implicit and explicit camera calibration: Theory and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):469–480, 1994.
[26] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980,Oct. 1992.
[27] Z. Zhang. Motion and structure from two perspective views: From essential parameters to euclidean motion via fundamental matrix. *Journal of the Optical Society of America A*, 14(11):2938–2950, 1997.