论文题目：柱面畸变二维码矫正算法研究与设计

专　　业：通信工程

学生姓名：冯展鹏

学　　号：12346029

指导教师：谭洪舟 教授

摘 要

传统的二维码扫描软件一般都只能够识别贴在平面上的二维码，而不能识别柱面上的二维码。一方面，平面上的二维码，当拍摄角度倾斜致使二维码成像时变成一个不规则四边形时，会导致二维码识别失败。另一方面，对于贴在柱面上的二维码当拍照角度不对或者是柱体曲面曲度较大时，也会由于透视投影到图像平面时二维码外轮廓变成椭圆曲线，导致二维码识别失败。本文在参考了国内外相关论文的算法成果的同时，提出了一种全新的基于 OpenCV3 的分离二维码寻位图形的方法。并设计了以一整套算法用来完成柱面二维码从复杂背景下提取寻位图形、边缘拟合、二维码区域分割提取以及矫正。最终测试结果表明，本文所设计的算法能够取得较好的二维码区域分割能力和一定的图形矫正效果。

关键词：OpenCV3；柱面二维码；特征提取；图形矫正

Title: Cylindrical QR Code Rectification Algorithm Research And Design
Major: Communication Engineering
Name: ZhanPeng Feng
Student ID: 12346029
Supervisor: Prof. HongZhou Tan

Abstract

Traditional qr code recognition software usually do not recognize qr code on cylindrical surface, but only the one on flat plane. On the one hand, when the camera shooting angle is not correctly, the qr code on the flat surface would be transformed into irregular quadrilateral which will lead to the failure of recognition. On the other hand, if the qr code is on the cylindrical surface while the shooting angle is not correctly at the same time, the outer contours of the qr code will become elliptic curve due to the perspective transform. After learning many useful information from several algorithms proposed by related papers, this paper proposes a new way to extract finder pattern from qr code based on OpenCV3. This paper also designs a whole set of algorithms for the preprocessing of qr code including extracting finder pattern from complex background image, edge fitting, region segmentation of qr code and rectification. Testing results indicating that this algorithm is effective.

Keywords: OpenCV3, Cylindrical QR Code, Feature Extraction, Graphic Rectification

目 录

第 1 章 引言

1.1 选题背景与意义

二维码自其产生以来就被给予了高度的关注以及广泛的应用，它在产品溯源、防伪，媒体推广，电子票务，生产管理等领域都有着极为广泛的应用。而且在当前互联网+的时代下，二维码也可以说是互联网的入口。很多商业应用都已二维码作为它们网站推广的手段。如此多的应用，可以说都得力于二维码本身较高的数据存储能力以及较强的数据容错性。然而，虽然二维码的应用日益广泛，但是市面上流行的传统的二维码扫描识别软件却仍然只能够识别良好摆放的二维码，当二维码出现扭曲畸变时，往往都无法识别。如果能够实现二维码在发生一定程度上的畸变时，仍然能够通过图像处理，还原二维码信息的技术，那么必然能够增加二维码的应用价值。在识别二维码的时候，用户可以仅仅通过拍照，从复杂背景下提取二维码区域，并且通过内置软件中的图像矫正算法还原二维码图像，最终识别二维码信息，也省去了用户不必要的麻烦。

1.2 国内外研究现状和相关工作

近年来，国内外关于二维码畸变图像识别与矫正的论文就有不少，Tong L 等人[1]提出一种基于 LBP 描述子的二维码提取算法，Chen Q 等人[2]则介绍了二维码识别的一种处理流程，Sun H 等人[3]研究了二维码预处理，Qi H 等人[4]提出了一种畸变二维码寻位图形的识别方法，Kang E 等人[5]则提出了一种基于形态学的二维码透视畸变的矫正方法，而 Li X 等人[6]提到使用一种基于最小二乘椭圆拟合算法来获取柱面畸变二维码的椭圆外轮廓，并最后使用一种透视投影的方法还原柱面二维码图像的算法，该算法使用到了 Fitzgibbon A 等人[7]提出的直接最小二乘椭圆拟合算法。还有一些书籍也有相关介绍。虽然这些论文都提供了某种意义上的实现，但是由于论文限于篇幅的问题，没有提供太多的细节，而且这些论文偏重于理论，往往也不提供实现代码，对于应用程序解决实际问题方面没有

提供太多有效的信息。固然，这给予了应用灵活的实现可能，但是也带来了算法实现的困难以及算法实现效果的差异，对算法比较也带来较大的不确定性。

最近几年，OpenCV3 图形算法库得到了较大的发展，众多成熟的图像处理问题的解决算法都有了相应的实现。这给二维码图像提取带来的方便之处，很多基本问题无须重复劳动，然而 OpenCV3 并没有关于矩阵运算过多的实现内容，这使得当算法需要复杂的矩阵运算，譬如求广义特征值，求矩阵的广义逆时，无法借助于 OpenCV3。而且 OpenCV3 不可能对所有问题都面面具到。因而，在实现算法时，也就需要根据实际应用需求与情况考虑更多的细节。以及需要通过不断地实验解决遇到的问题，逐步修正程序。

1.3 本文的研究内容与主要工作

本文总结了参考论文中的理论与算法，结合 OpenCV3 实现了二维码在复杂背景下高精确的 Finder Pattern 的提取，使用最小二乘拟合等方法，在 C++结合 Qt 的桌面平台以及 Android 使用 Java 结合 JNI 的方式分别实现了本文的算法。流程包括，Finder Pattern(下文都称做寻位图形)提取、柱面二维码外轮廓的拟合、多重透视变换矫正图像。

1.4 本文的论文结构与章节安排

本文共分为六章，其中，第二、三、五章是本文的重点内容，具体章节内容安排如下：

第一章包含柱面畸变二维码背景意义，以及国内外研究现状和本文的研究内容和主要工作。

第二章包含柱面畸变二维码矫正的相关实现技术的简要阐述。

第三章包括本文算法设计的流程以及原理。

第四章包括编程实现的程序流程图和简单的实现介绍。

第五章包括在两个平台上的实验仿真过程以及结果分析。

第六章则是本文的工作总结以及研究展望。

第 2 章 相关技术与原理

本章阐述柱面畸变二维码特征提取，边缘拟合以及图像矫正所用到的相关核心技术以及基本原理。算法设计细节在第三章阐述。而算法实现的细节在第四章阐述。由于本章是支撑起本篇论文最为重要的理论计算基础，后续算法程序会使用到本章所述公式，所以这部分较为详细。

2.1 OpenCV3 图像轮廓特征提取

OpenCV3 是重要的图像处理工具[8]，它提供了众多成熟的图像处理算法的高效实现。在二维码预处理的过程中，往往需要定位二维码究竟在图像的哪一个区域。这就需要根据图像当中的特征，从复杂的背景当中定位二维码区域。

在轮廓提取之前，首先要对图像进行阈值化操作，只有这样才能够有效的提取图像的边缘轮廓。adaptiveThreshold 是 OpenCV3 提供的图像自适应阈值化的函数。对于处理二维码图像而言，为了提取出图像中的轮廓，必须排除噪声，以及光度不均匀对图像造成的影响。如果只是将灰度图像使用全局的二值化处理，这样往往得到的图像会不太理想。在阈值化完成后，通常也会进行中值滤波操作，下图 2-1 所示就是一般二维码图像预处理的常用操作流程图。

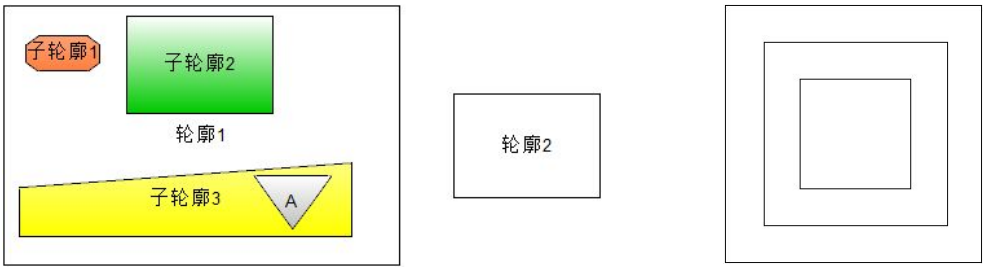
图 2-1 二维码预处理一般初始步骤

对一幅图像提取轮廓，必然需要从众多轮廓中筛选出满足预设条件和特征的轮廓。就封闭轮廓而言，其特征一般有两个方面。

一方面是轮廓与轮廓之间的关系，譬如父子关系，兄弟关系。父子关系是指一个封闭轮廓完全包含另外一个轮廓，兄弟关系是指它们没有重合的部分，并且它们处以同一个层次。

另外一方面，就是它们本身所包围区域的面积大小，所需要提取出的这些轮廓包围的面积很多时候它们的比例也是重要的筛选信息之一。

通过这两个方面的信息，往往能够在图像当中提取出所需要的轮廓边缘点以供后续处理来使用。如下图 2-2 (a)所示，轮廓 1 和 2，子轮廓 1 和子轮廓 2 是同一个层次的轮廓，它们互为兄弟轮廓。而轮廓 1 只有三个子轮廓，包含子轮廓 1、2、3。子轮廓 3 包含子轮廓 A。



(a) (b)

图 2-2 (a) 轮廓关系示意图 (b) 二维码寻位图形的边缘轮廓样式

对于二维码的寻位图形(Finder Pattern 以下都称为寻位图形)样式如图 2-2 (b)所示。最外边的轮廓有且只有一个子轮廓，而其子轮廓也有且只有一个子轮廓。而且由于它们的边缘满足 1:1:3:1:1 的关系，显然，不管如何畸变，它们的面积比例往往在一定的范围之内。设最外边的轮廓到里边的轮廓的面积依次为A、B、C，则它们的面积比例关系可以设为(数值上可以适当宽松或者收紧)：

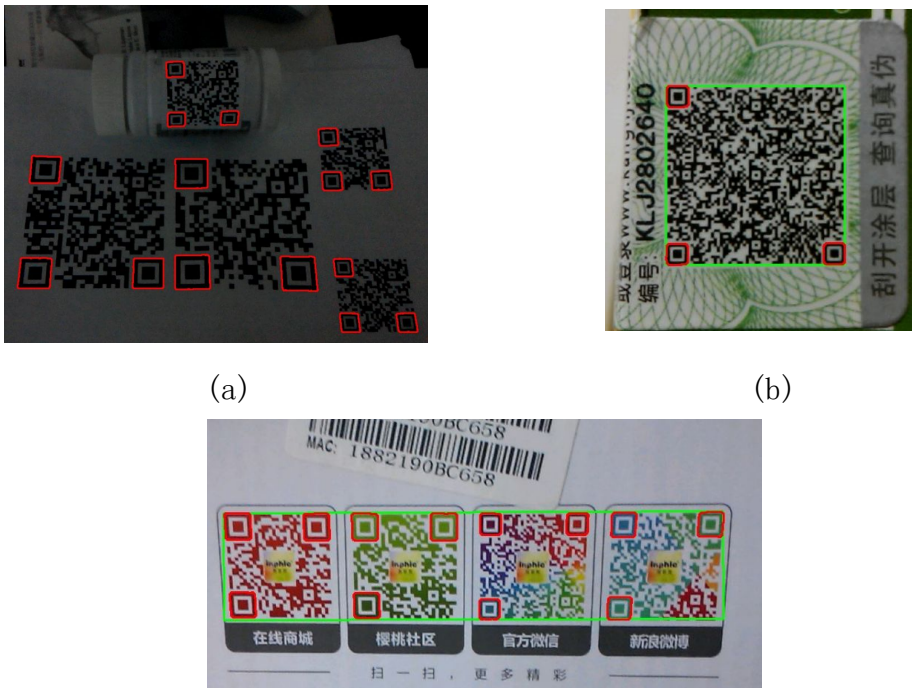
$$1.5*B < A < 4*B\,, \tag{2.1}$$

$$1.5*C < B < 4*C\,, \tag{2.2}$$

OpenCV3 提供了 findContours 函数来提取图像轮廓，并且会把它们组织到一个链表当中，每一个轮廓都是一个封闭边缘点相连的轮廓。当设置模式为 RETR_TREE 时，所有轮廓会被组织为一颗树。树中的一层代表兄弟轮廓，每一层往下遍历就是寻找子轮廓。通过这种遗传关系树，很方便就可以确定上述所说的关系。只要找到一个轮廓有且只有一个子轮廓，其子轮廓也有且只有一个子轮廓。则这个轮廓就是寻位图形的候选轮廓。此时依次计算封闭轮廓包围的面积，就可以得到它们的比值关系，满足条件的就是二维码的寻位图形。这种方法，如果图像中只有一个二维码，则只会有三个寻位图形，如果有多个二维码，每个二维码的寻味图形也都能够找到。值得注意的是，如果预处理后寻位图形出现边缘断线，也就是说无法封闭，这个时候这样做就会失败了。

诚然，如果只使用轮廓关系可能找到不正确的寻位图形。因为有这种关系的

图形还是有的，只是不多见罢了。进一步结合了面积比例关系后，就大大减少了识别错误的可能性。



(a) (b)

(c)

图 2-3 二维码寻位图形示例图

如图 2-3 所示，是由测试程序在安卓手机上利用 OpenCV3 处理的效果图。红色包围的是寻位图形，图(b)和(c)的绿色方框则包围了所有的寻位图形。

2.2 最小二乘拟合

在图像处理中，单单只是提取了所需图像当中的部分边缘点往往是不够的。在柱面二维码提取过程中，得到了一定数目的寻位图形的边缘点之后，还需要完整地包含整个二维码区域才行。在得到寻位图形的边缘点后，可以采用最小二乘的方法[9]来求取方程作图，包围二维码。

首先，来看直线的最小二乘拟合。对于直线而言，通常不垂直于 y 轴的直线可以表示为如下公式：

$$y = bx + c\,, \tag{2.3}$$

最小二乘就是想要找到一组这样的解 (b,c) 使得如下和式最小

$$Minimize: S = \sum_{i=1}^{n} (bx_i + c - y_i)^2 , \tag{2.4}$$

对于抛物线而言，

$$y = ax^2 + bx + c , \tag{2.5}$$

也就是最小化如下等式：

$$Minimize: S = \sum_{i=1}^{n} (ax_i^2 + bx_i + c - y_i)^2 , \tag{2.6}$$

转化为矩阵表达，对于抛物线，如果令

$$A = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ x_n^2 & x_n & 1 \end{bmatrix}, \quad x = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad b = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{bmatrix}, \quad x_i \neq x_j (i \neq j), \quad \text{则有：}$$

$$Ax = b , \tag{2.7}$$

可以知道，这样求解 x，由于这是超定方程组，通常 x 没有解，这是因为列向量 b 一般不再 A 的列空间当中。令：

$$Ax = b', \tag{2.8}$$

设 b' 在 A 的列空间中，则此时有解，为了使得误差最小，则应该有：

$$Minimize: S = |bb'|^2 = b - b' = \sum_{i=1}^{n} (y_i' - y_i)^2 , \tag{2.9}$$

这也就是最小二乘的条件，根据投影的原理，当 b' 是 b 在 A 的列空间的投影时，此时向量 bb' 模长最小。因而有：

$$A^T (b - b') = 0 , \tag{2.10}$$

结合(2.8)和(2.10)可以得到：

$$A^T Ax = A^T b \Rightarrow x = (A^T A)^{-1} A^T b , \tag{2.11}$$

其中由于 A 有线性无关的列向量，而且对于 $A^T Ax = 0$，两边同乘以 x 的转置，有 $x^T A^T Ax = (Ax)^T Ax = 0 \Rightarrow Ax = 0$，所以 $A^T Ax = 0$ 与 $Ax = 0$ 同解。有 $n - rank(A^T A) = n - rank(A) \Rightarrow rank(A^T A) = rank(A)$，此时 $A^T A$ 为对称满秩方阵，所以 $A^T A$ 的逆必然存在。

根据式(2.9)就可以求出方程的各个参数。实际上，任何形式上具有 $y=f(x)$ 的函数都能够这样拟合，只是说高次项由于乘数过大有时候这样处理可能导致程序变量溢出产生错误。

在实际使用直线和抛物线拟合进行图像处理的过程中，也不必一定非要把 y 当作因变量，完全可以使用 x 作为因变量来避免不好的情形发生。如下两式子所示：

$$x=by+c, \tag{2.12}$$

$$x=ay^2+by+c, \tag{2.13}$$

上边的最小二乘理论方法，都是针对于因变量只是一次的形式。对于其他一些二次曲线来说，比如椭圆，想要使用最小二乘拟合，却不能够使用上边的方法。对于柱面畸变的二维码，曲线边缘透视过来就是椭圆，所以考虑椭圆拟合具有一定的意义，故在此简述。椭圆的直接最小二乘方法如下所示：

椭圆的一般方程为

$$ax^2+bxy+cy^2+dx+ey+f=0, \tag{2.14}$$

$$f(x.y)=Da=0,\ D=\begin{pmatrix}x^2 & xy & y^2 & x & y & 1\end{pmatrix},\ a=\begin{pmatrix}a & b & c & d & e & f\end{pmatrix}, \tag{2.15}$$

$$Minimize:\Delta(a,D)=\sum_{i=1}^{n} f(x_i,y_i)^2, \tag{2.16}$$

其中，x 和 y 都是由多个点 (x,y) 坐标组成的列向量，而不是单个值。

通过上诉各个式子，可以得到：

$$\Delta(a,D)=a^TD^TDa=a^TSa,\quad \text{其中}\quad S=D_i^TD_i \tag{2.17}$$

S 是6x6的一个方阵，对于椭圆来说，有约束 $4ac-b^2>0$。令 $\Phi=4ac-b^2>0$，

$$C=\begin{bmatrix}0&0&2&0&0&0\\0&-1&0&0&0&0\\2&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\end{bmatrix},\ \text{则有}\ a^TCa=\Phi, \tag{2.18}$$

这个时候，式(2.17)最小化的问题，变成了具有约束式(2.18)的最小化问题。

此时由拉格朗日乘子法：

$$L(a)=\Delta(a,D)-\lambda(a^{T}Ca-\Phi), \tag{2.19}$$

为了最小化此式，求偏微分可得：

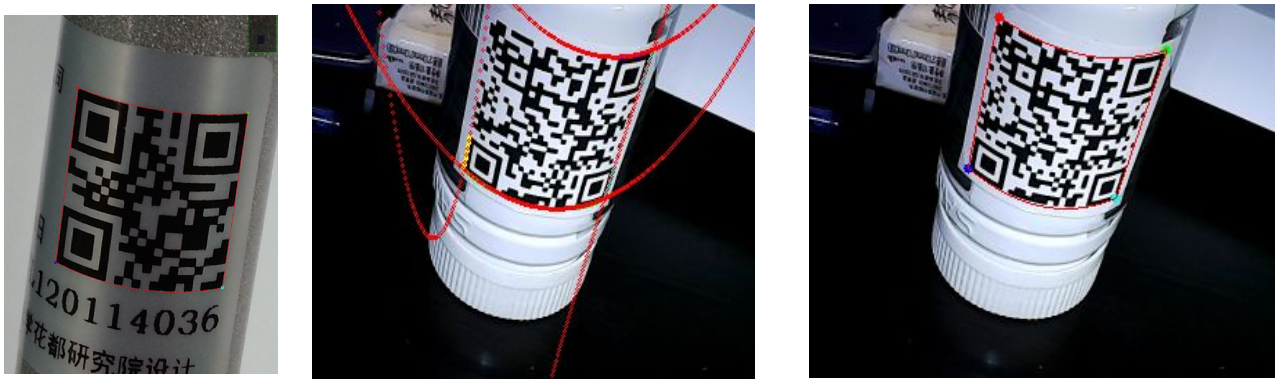
$$\frac{\partial L(a)}{\partial a}=0\Rightarrow Sa=\lambda Ca, \tag{2.20}$$

这把椭圆拟合变为了求解广义特征值问题。一般情况下，S 是对称正定矩阵，所以可以如下处理：

$$\frac{1}{\lambda}a=S^{-1}Ca, \tag{2.21}$$

这样就转化为了一般求特征值的问题。找到最大的 $\frac{1}{\lambda}$ 对应的特征向量就是所要求得椭圆的一般方程的各个参数。

使用上诉的方法，在实际编程过程中还要考虑数据归一化和反归一化的问题，矩阵运算也比较复杂，不能像依靠像抛物线和直线拟合那样只依靠求矩阵乘法，以及对称满秩矩阵的逆就可以解决。



(a) (b) (c)

图 2-4 寻位图形边缘拟合结果图

如图 2-4(a)(c)所示，是使用寻位图形外边缘点拟合的结果图，边缘用红色画出，各顶点使用了不同颜色标记。需要注意的是，二维码方位有很多种，这个算法在后几章阐述。而且，必然有一曲线边只有一个寻位图形的边缘点，这导致最小二乘拟合效果不好，会有错误产生，譬如线段覆盖了二维码区域，或者完全如图 2-4(b)就在外边。这就需要能够自动矫正的最小二乘拟合算法，这在后续章节阐述。

2.3 透视变换

透视变换是描述两个在不同空间的点的一种变换关系。设三维失真空间中的点为$[u,v,w]$，其齐次坐标为$[u,v,w,1]$。三维基准空间中对应的点为$[x,y,z]$，其齐次坐标为$[x',y',z',k]$。其中有：

$$[x,y,z]=[x'/k,y'/k,z'/k], \tag{2.22}$$

对于这两个点而言，透视变换使用了一个 4x4 的矩阵描述它们的映射关系：

$$\begin{pmatrix} x' \\ y' \\ z' \\ k \end{pmatrix}=\begin{bmatrix} A & B & C & D \\ E & F & G & H \\ I & J & K & L \\ M & N & O & P \end{bmatrix}\begin{pmatrix} u \\ v \\ w \\ 1 \end{pmatrix}, \tag{2.23}$$

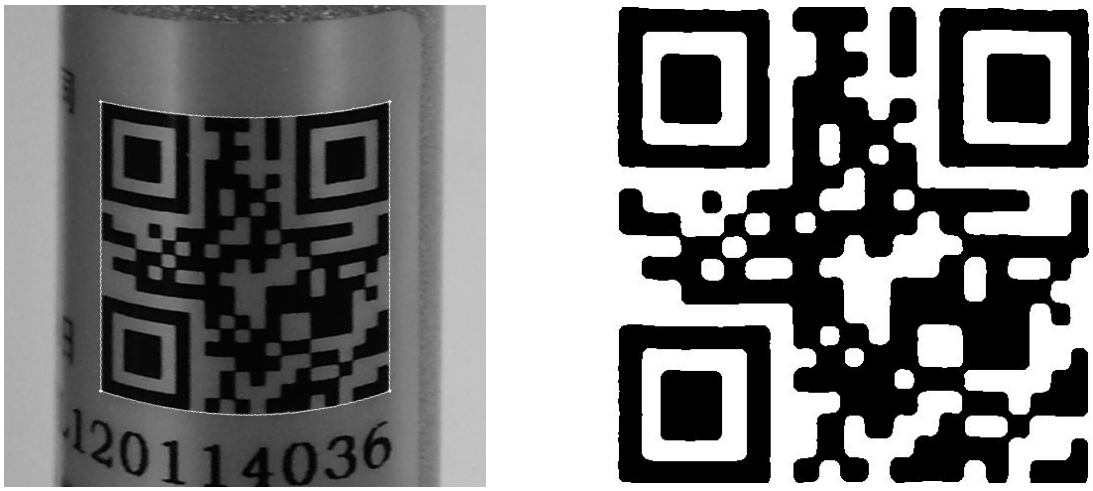
在图像处理过程中，认为两个图像变换前后处于同一个平面，因而可以变为如下 3x3 的矩阵表达式：

$$\begin{pmatrix} x' \\ y' \\ k \end{pmatrix}=\begin{bmatrix} A & B & D \\ E & F & H \\ M & N & P \end{bmatrix}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \tag{2.24}$$

在 OpenCV3 中，就是使用式(2.24)的形式来做图像透视变换的。在做透视变换时，只需要找到四对点就可以求出 3x3 的矩阵表达式。如果设变换前的点依次为$(u_1,v_1),(u_2,v_2),(u_3,v_3),(u_4,v_4)$，变换后的点为$(x_1,y_1),(x_2,y_2),(x_3,y_3),(x_4,y_4)$。则式(2.24)可以变为：

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 \\ u_4 & v_4 & 1 & 0 & 0 & 0 & -u_4x_4 & -v_4x_4 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3y_3 & -v_3y_3 \\ 0 & 0 & 0 & u_4 & v_4 & 1 & -u_4y_4 & -v_4y_4 \end{bmatrix}\begin{pmatrix} A \\ B \\ D \\ E \\ F \\ H \\ M \\ N \end{pmatrix}=P\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}, \tag{2.25}$$

对于式(2.25)一般也设$P=1$，然后根据四点直接解这个线性方程组即可[10,11]。得到变换矩阵之后，再依据式(2.24)和式(2.22)就可以把原图像逐个点变换到新图像对应的点，原点的像素值不变。一般得到的图像还需要进行插值操作。



(a) (b)

图 2-6 透视变换矫正后的柱面图像

如图 2-6(a)所示是歪斜的柱面二维码图像经过四个顶点提取后做一次透视变换的结果图。可以看到两边都变成了垂直于x轴的直线。上下两边变成了曲线。再经过多重基于图像分割的透视变换，经过阈值化和中值滤波后，变为如图2-6(b)所示的矫正后的图像。对于一般的透视畸变的二维码，往往也可以采用形态学方法霍夫变换得到外轮廓后来矫正[12]。

2.4 本章小结

本章介绍了 OpenCV3 里的轮廓特征提取算法以及本论文所涉及的基本知识。最小二乘拟合以及透视变换是本文使用到的核心数学方法及原理。然而，仅仅只知道基本的原理技术是不够的，在实际解决问题时，往往还需要根据实际情况考虑众多可能遇到的问题。这就涉及到算法如何设计，如何实现的问题。所以本文在后续章节进一步阐述算法的设计以及具体的实现。

第 3 章 柱面畸变矫正算法设计

本章节阐述本论文实现程序的算法流程。首先在第一小节给出了总的流程。在后续小节则分别给出了各个重要部分详细的流程。

3.1 总算法流程

整个程序的流程图如下图 3-1 所示，可以看到，核心的步骤有六步。

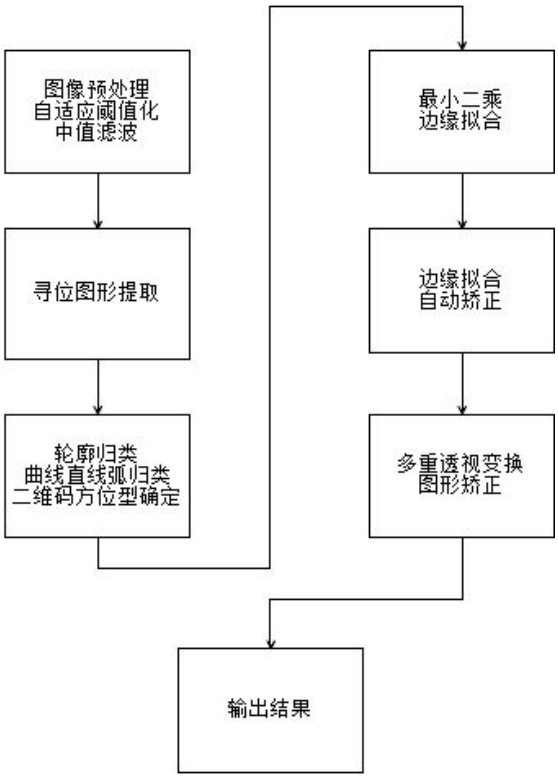


图 3-1 总算法流程图

第一步是一般的预处理步骤，这一步有时候往往也是最耗费时间的，因为很多时候图像分辨率很大，所以自然的运算量也很大。相应的参数譬如阈值化的掩模大小，中值滤波的掩模大小都需要根据图像分辨率调整。一般如果是摄像机拍照，得到的图像分辨率往往已经固定，所以这些参数依靠预先设置的就好了。本文不会对这一步过多叙述。

第二步和第三步在本章第二小节分析。第四步和第五步在本章第三小节分

析。第六步在第四小节分析。

3.2 轮廓提取归类算法

寻位图形的提取原理在 2.1 节已经详细阐述，这里只列出 C++代码示例图。

```
findContours(g_binImg, g_Contours, g_Hierarchy, RETR_TREE,CHAIN_APPROX_SIMPLE);
int c_index, cc_index, fp_count = 0;
size_t i;
for (i = 0; i < g_Hierarchy.size(); ++i) {
    c_index = g_Hierarchy[i][2];
    if (c_index != -1) {
      cc_index = g_Hierarchy[c_index][2];
      if (cc_index != -1 && g_Hierarchy[c_index][0] == -1
        && g_Hierarchy[c_index][1] == -1
        && g_Hierarchy[cc_index][0] == -1
        && g_Hierarchy[cc_index][1] == -1) {
        double area = contourArea(g_Contours[i]);
        double area_c = contourArea(g_Contours[c_index]);
        double area_cc = contourArea(g_Contours[cc_index]);
        if (area < area_c*4 && area > area_c*1.5 && area_c < area_cc*4 && area_c > area_cc*1.5) {
          g_Finder.push_back(g_Contours[i]);
          ++fp_count;
        }
      }
    }
}
```

图 3-2 寻位图像定位 C++代码截图

通过图 3-2，可以看到这种方法比使用 1:1:3:1:1 的方式的简洁与优雅。在实际测试时，准确性也大幅提高。

在进一步叙述之前，首先应当知道在实际处理二维码图像时的一些信息。

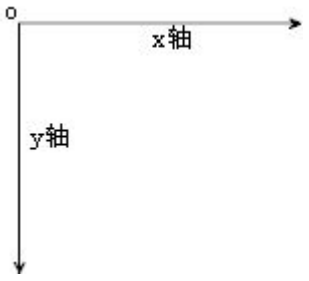
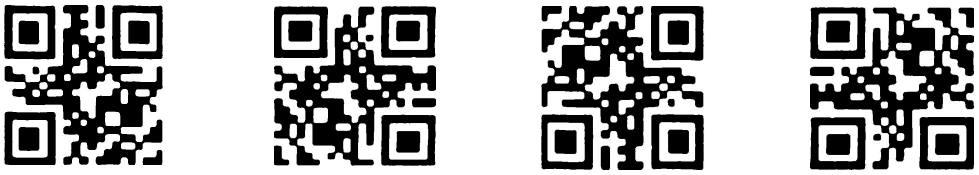


图 3-3 图像的存储表达

一般来说，图像中各个点的(x,y)坐标是由图 3-3 中所示来表达的。每一幅

存储图像，从(0,0)坐标开始，行代表 y 轴，列代表 x 轴。如果把 x 轴视为上边，y 轴视为左边，那么一个不规则四边形图像就有上下左右四条边。它们不一定严格的平行于 x 、 y 轴。只需要根据它们每一条边的重心来判断。最靠近 x 轴的是上边，最靠近 y 轴的是左边，其余最远离 y 轴的是右边，最远离 x 轴的就是下边。

对于二维码来说，实际拍摄得到的畸变二维码，有多个方位。但不管它怎么旋转，怎么畸变，总的来说只有四种方位，本文在这里将其定义为二维码在图 3-2 所示的图像坐标系下的方位型。



(a) (b) (c) (d)

图 3-4 二维码方位型

如上图 3-5 所示，在这里依次把(a)(b)(c)(d)视为 I、II、III、IV 型。

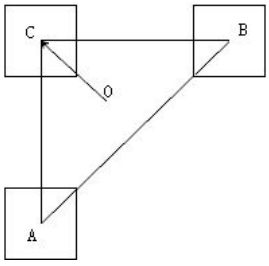


图 3-5 二维码示意图

对于图 3-5，点 A 、 B 、 C 为三个寻位图形的外轮廓点的重心，O 为 A 、 B 、 C 的重心。可以想象，二维码在沿着 O 顺时针旋转，那么向量 OC 的 x 、 y 坐标依次为负负，正负，正正，负正。通过坐标的符号知道，确实只有四种情形。这种状况并不需要寻位图像是严格的四边形。

而且可以确定每一个寻位图形上下左右四条边，结合方位型就可以知道在拟合二维码外轮廓时，哪一个二维码寻位图形的边需要参与拟合，那一边不需要。通过图 3-5 可以看到，根据寻位图形的外轮廓信息，有两条外边有两个寻位图形的边参与拟合，因此比较准确，事实上也确实如此。另外两个只有一个边，误差较大。

在确定 A、B、C 三点的方位时，可以根据三角形 ABC 的各边长度确定，最长的对应于 C 中心点。如何确定 OC 的左边点 A，和右边点 B 需要计算几何学的一个知识。在计算几何学里，有一个问题就是对一堆离散不重合的点集，将它们依照它们的中心点顺时针或者逆时针排序。它们都是离散的二维点，判断点处于顺时针还是逆时针恰恰可以依靠叉乘的性质。

设在三维坐标系下，xyz 轴构成右手坐标系。那么依照图 3-3 所示，z 轴向图像里边。

$$OC \times OP = \begin{vmatrix} x_c & y_c & 0 \\ x_p & y_p & 0 \\ i & j & k \end{vmatrix} = (x_c y_p - y_c x_p)k , \tag{3-1}$$

判断他们叉乘的正负。如果是正，说明 OP 在顺时针方向，反之则在逆时针方向。以基准向量为直线，把逆时针的都可以归到左边，顺时针的都归到右边。不断如此递归把点分到左右两边就可以实现点的顺时针或者逆时针排序。

结合上边的简单理论，如图 3-6 所示就是上述过程的流程图。

这样，按照上边所述，就可以把二维码的三个寻位图形定为 A、B、C 了。它们之中，C 是中间的寻位图形，A 是 C 左边的，B 则在 C 右边。具有方位型的信息就可以确定，究竟需要寻位图形的那一边作为拟合边。如下图 3-7 所示就确定了所需要用来拟合边缘的点的来源。

然而，柱面畸变的二维码有两条曲线边，也有两条直线边。这可以通过求上下左右各两边的包围矩形的窄边宽度判断。越是宽的，说明就是曲线边。这个编程实现并不难，在此不再赘述。在实际测试时，使用椭圆拟合发现效果只有在有两个边的时候才会效果好，用到抛物线来拟合反而效果不错。所以这里定义二维码类型为 UP_DOWN_PARABOLA 型和 LEFT_RIGHT_PARABOLA 型两种。在实际考虑图形时，也就有了八种情况。

在这一小节的最后，还需要对单个寻位图形如何把可能无序的点归类为上下左右四条边做进一步的分析。

对于得到的寻位图形外轮廓点，显然，为了分得出上下左右，需要找到它们的角点。此时，可以创建一幅小的能够包围所有该轮廓点的图像。这幅图像可以认为是原图像在这个寻位图形处的 ROI 图像。把轮廓点按照坐标变换后依次连线

画在图中，设置好参数，使用函数 goodFeatureToTrack 就可以获取四个角点了。

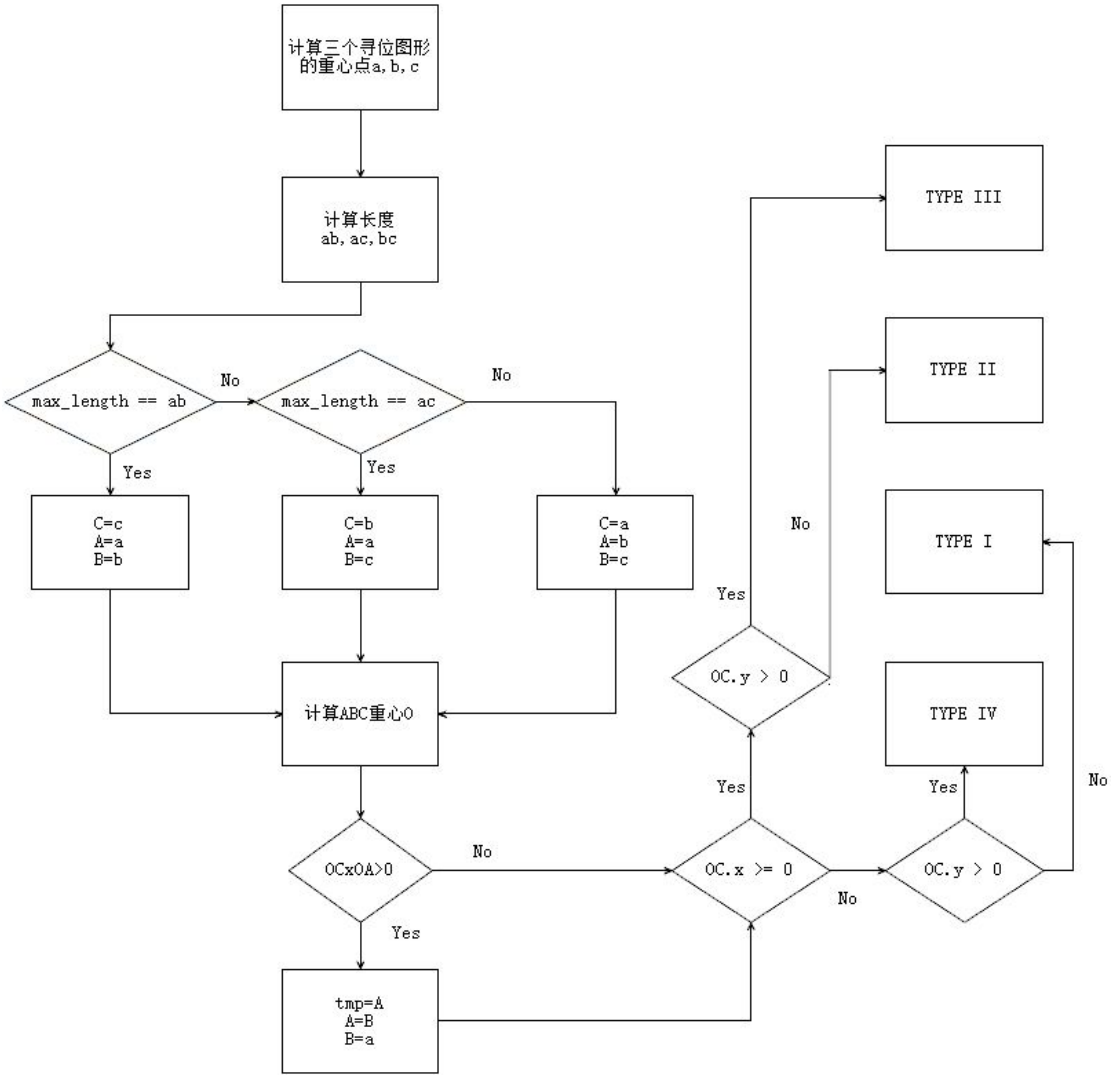


图 3-6 二维码方位型确定流程图

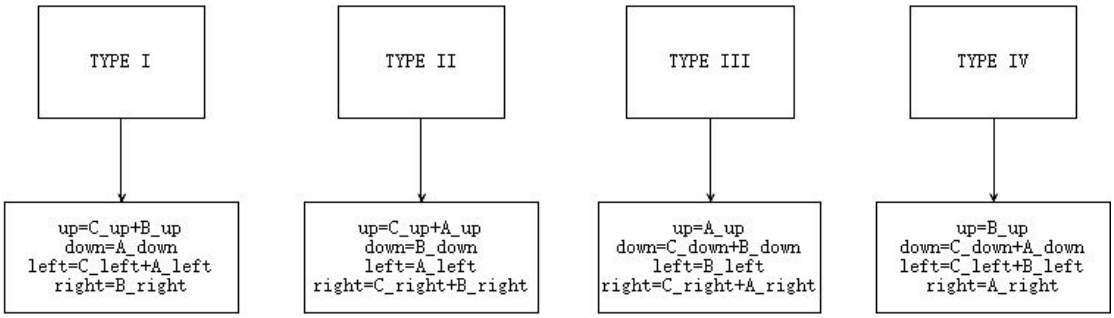


图 3-7 根据方位型确定拟合四边的点数据来源

期间，当然，点还是需要顺时针或者逆时针排好序的。区分了四个 up_left, up_right, down_left, down_right 角点。也就可以把他们之间的点找出来了。这样在按照上边的流程。二维码四边用于拟合的点就全部得到了。

需要注意的是，由于此时只使用了单个寻位图形来分类，这可能导致错误。还需要从三个寻位图形为一个整体来考虑边的分类。可以用三个寻位图形的中心点构成的三条线与各个寻位图形的边的夹角的正余弦值来判断他们是否出错。如果出错，则交换边就可以保证分类正确了。

3.3 边缘点拟合算法

在 3.2 节，阐述了二维码外轮廓拟合点的提取方法以及二维码类型的归类。

就边缘拟合来说，对于 UP_DOWN_PARABOLA 型，使用式(2.5)来拟合上下曲线边。使用式(2.12)来拟合左右直线边。这样做提高了包围准确性，同时避免了使用式(2.3)拟合可能出现的斜率过大或者垂直于 x 轴的情形发生。

同理，对于 LEFT_RIGHT_PARABOLA 型，使用式(2.13)来拟合左右曲线边，使用(2.3)来拟合上下直线边。

具体得到的方程参数是由 2.2 节式(2.11)计算得到的。观察直线和抛物线的方程，它们其实具有一致的形式。也就是二次项是否为零的差别。实际编程可以认为他们都是抛物线。统一了处理方式。在求解交点时，根据二维码归类来区分方程形式求解交点即可。

(a) (b)

图 3-8 二维码拟合错误示例

这里，详细阐述自动矫正误差的方法。

首先，可以通过图 3-8 看到，其实这种拟合错误只有两种情况。也就是误差线完全在二维码区域里，或者误差线完全在二维码区域以外。

(a) (b)

图 3-9 自动矫正后的二维码图像

沿着这条线，从起始点开始到终止点。如果把灰度值低于 50 认为黑色，大于为白色。它们的差别在于，沿线的黑白像素的比例不同。如果以白比黑作为衡量的比例。那么显然，正确的边缘拟合线也会有一定的范围内的白比黑比例。认为 50 以下是黑色是因为对比灰度条可以发现。人眼看到的十分确定是黑色的基本低于 50 灰度值。可以想象的是，对于图 3-8(a)，把一点沿着右边直线往二维码区域以外方向移动。然后把这一点放入原拟合点集中继续拟合，显然，得到的新抛物线又会往外移动。白黑像素比也会发生变换。不断这样循环，轮廓线正确时，白黑比会在预设的一个比值区间之中。此时中止循环。对于图 3-8(b)的情形，则把点向内移动。经过实验对比，发现：

$$6 < P_w / P_b < 12, \tag{3-2}$$

$$P_w / P_b > 12, \tag{3-3}$$

$$P_w / P_b < 6, \tag{3-4}$$

当式(3-2)满足时，轮廓线是基本正确的，如图 3-9 所示。当式(3-3)满足时，轮廓线在二维码区域以外，显然，此时白点过多。当式(3-4)满足时，轮廓线在

二维码区域以内，显然此时黑点很多。只有在边缘处，会有过度的现象发生。设置好上下区间，就可以筛选出正确的情况了。

这种方法应该不会受到分辨率影响，但显然会受到图像光度的干扰。上边的图像都是八百万像素的。二维码区域只占图像很小的一部分。下边是一些分辨率较低的图像的效果。

(a) (b)

图 3-10 自动矫正的低分辨率图像轮廓提取

图 3-11 不同类型二维码的方向移动 dir 与直线归类图

结合二维码可能具有的八种方位，定义 dir 为线上点向外移动时对应 x 或 y 坐标增减，1 代表递增，-1 代表递减。step 是一次移动像素步长，step 大于 0 表示向外移动，小于 0 表示向内移动。

根据图 3-11 还可以知道像素移动的使用哪条直线的 x 还是 y 坐标。这样，自动矫正算法的流程图如下图 3-12 所示。

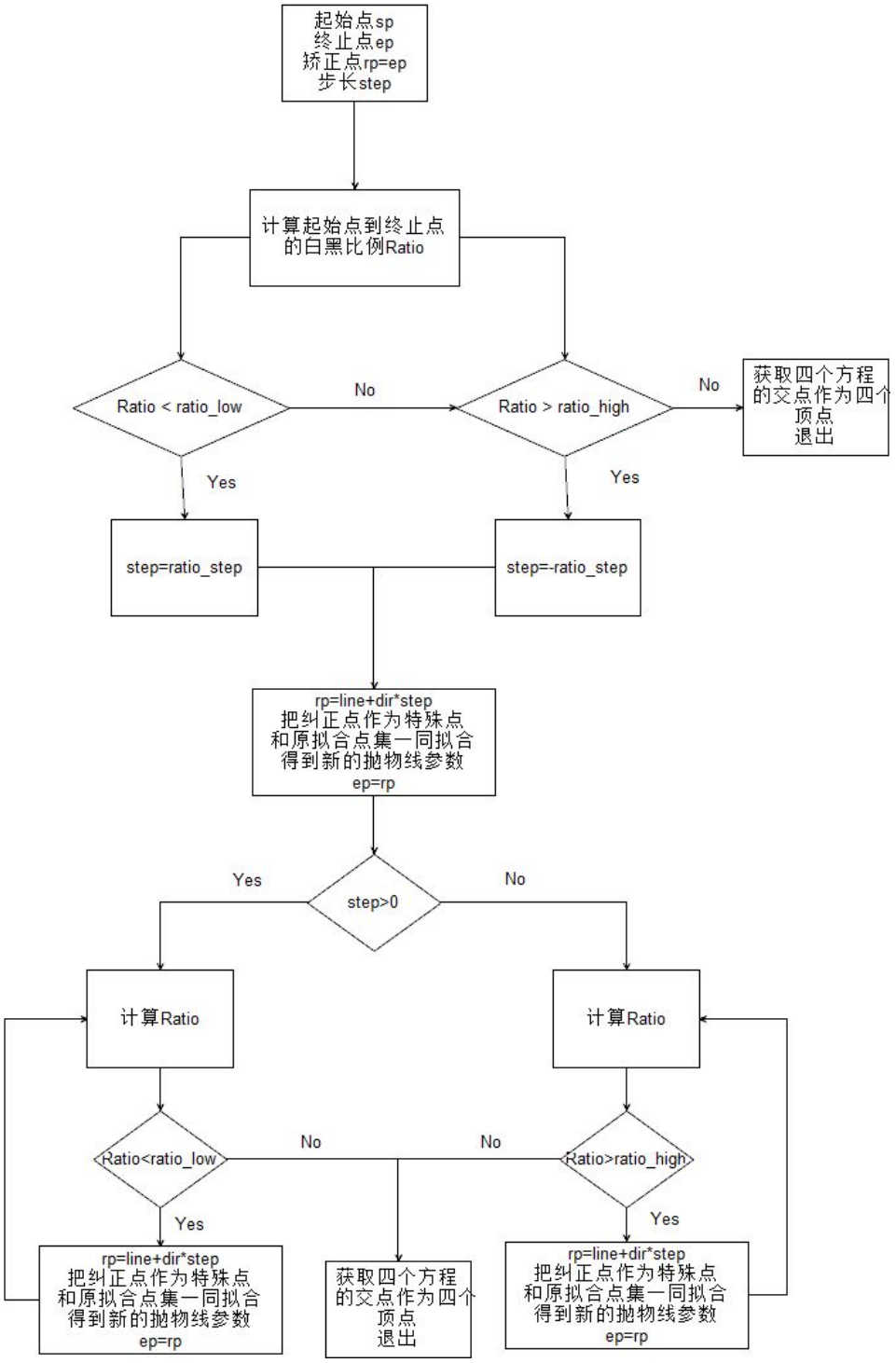


图 3-12 自动矫正算法流程

最终，算法会得到四个顶点作为二维码包围轮廓的顶点。利用这四个顶点就可以做一次透视变换，得到两边为直线，上下为曲边的情形。需要注意的是，上述算法没有指出求曲线检点可能出现的交点溢出(即两方程交点不再图像内部)等实际情况可能会导致算法出错的情形，也没有给出如何求方程交点的内容。

对于判断是否溢出，需要在获取交点时判断是否在图像范围之内，如果不在，那么就设置这角点为这条直线在图像边缘点的位置。求解方程需要注意的是，由于采用了上下边 y 作为因变量，左右边 x 作为因变量的拟合方式。因而它们的系数首项可能很小。在让他们的某个系数作为除数时，这个系数不能过于小，否则会出现计算结果溢出错误。通常设置小于 10 的负 7 次方就可以认为这一系数是 0 了，不能作为除数了。

3.4 图形矫正算法

二维码的图形矫正基于 2.3 节阐述的透视变换。该小节提到了图像分割成条状来矫正柱面图形的方法，但是那种方法针对于柱面是正对摄像头的情形[13]。本文所涉及的柱面并不具有如此简单的形式，需要首先针对四点先做透视变换。而且，如果需要精确地还原，需要更为复杂的数学模型来通过二维图形重建三维的柱面结构[14,15]，这是十分困难的，不利于实现。在二维码版本不是很高时，采用分割图形成条状。然后视作这个条带只发生了透视畸变，然后还原它们，最后重组，是对于实际情况的一种近似方法。切的份数约多，显然就越精确，此时曲线越接近直线！

在通过 3.3 节的算法计算后，会得到了二维码包围区域的四个顶点。此时以这四个顶点作为原始点，建立一幅图像，设置映射点的坐标，依据式(2.25)就可以得到坐标变换矩阵。最终会得到图 2-6(a)所示的图像。此时两边变为了垂直线。

如下图 3-13 是此时变换得到的外轮廓线。上下边是曲边，可以向下凹，也可以向上凹。通过该图，可以计算上下两边的总长度。把图像垂直切成 n 份，那么每一份的上下曲边长度应该就是总长度的 n 分之一。这样就可以把图像多次切割并分别作透视变换，得到一个个图像条带，然后组合，就可以得到矫正的图形了。

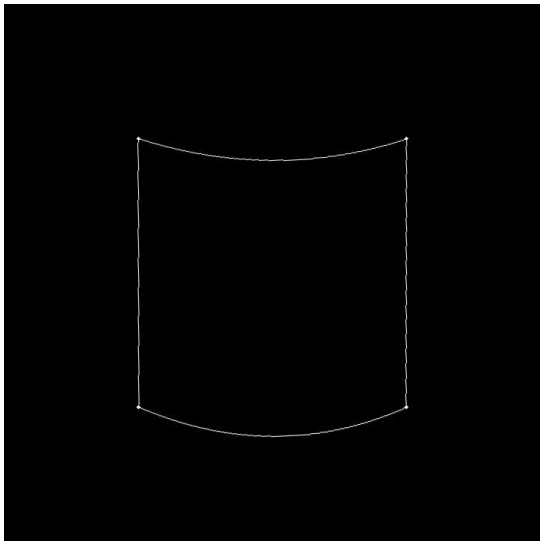


图 3-13 外轮廓线

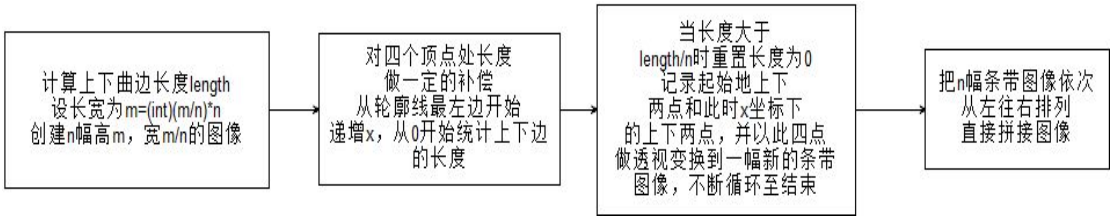


图 3-14 矫正流程图

这个方法需要对两边的点做一定的长度补偿。认为这是三维图像，每一点都具有场景深度。图像在这些地方是具有深度的，这个是 3D 的柱面投影到平面上的结果。一般曲度不大的情况下，长度补偿值是可以预先估计出来的。

最终处理结果类似于图 2-6(b)所示。

3.5 本章小结

本章阐述了柱面畸变二维码算法的设计以及对于这样设计的原因分析。在必要的地方添加了算法流程图作为补充。逐步深入地介绍了本文应用设计算法的设计思想以及怎样一步步处理实际情况中的问题的思路。

第 4 章 柱面畸变二维码算法实现

本章简要的提及算法实现中的各个参数设置、实现以及实现的平台。

4.1 算法参数设置与实现方法

在图像处理里，一般都是不会出现所有图像都能够处理的情况。要么有事先设计好的自动化算法。在本文所提及算法中，对于图像的预处理有这样的问题。譬如自适应阈值化，中值滤波。对于大分辨率图像。本文设置中值滤波的核大小为 9。小分辨率则为 3。对于自动拟合矫正，对于差别很大的二维码，两个高低参数的设置也是不确定的。只能根据实际情况做一定的调整。在透视变换矫正中，究竟把图像分成多少块，补偿的数值多少，并不是非常确定。本文设置补偿数值为 $5m/3/200$ 。这对于测试样例来说是合适的。

算法的实现往往依赖于具体语言的特性和实现者自身对于语言本身的考虑。本文所用算法使用 C++在桌面平台以及 Java 在安卓平台实现算法并且测试。使用面向对象的方法实现算法。

首先，针对本文提及算法。设计了如下几个基本类。

1、arc 类：弧类。用于边缘点拟合的一边是一条弧。弧类自身实现拟合算法，并且把输入点集排序等。

2、finder_pattern 类：寻位图形类。每个二维码都有三个这样的图形。使用类来管理，它有四条弧。管理自身的四条边，并将它们区分归类。

3、qr 类：二维码类。包含二维码的预处理，原始图像的灰度图等。由前边的类构建。包含二维码图像输入的初始化，以及做图像的透视变换等后期操作。

上述类都是层层包含的关系。arc 是 finder_pattern 的组成，后者又是 qr 的组成部分，把功能拆分，方便功能实现与调试。其次，还需要有管理类来管理各个核心类。

1、fp_manager 类：finder_pattern 的管理类。负责三个寻位图形的方位型确定以及外边缘点的重组。还包括根据原灰度图自动矫正最小二乘拟合的功能。

2、qr_manager 类：qr 码包装给用户使用的类。包括在图片中提取二维码，通过摄像头拍摄二维码等。由于上述算法都有一定预设的流程，错误的步骤往往导致未定义的结果。所以最好包装一下。

4.2 在桌面平台的实现

本应用设计使用实时摄像提取二维码区域，拍照后矫正图形。

设计流程如图 4-1 所示。这个流程在安卓手机上也是可行的。

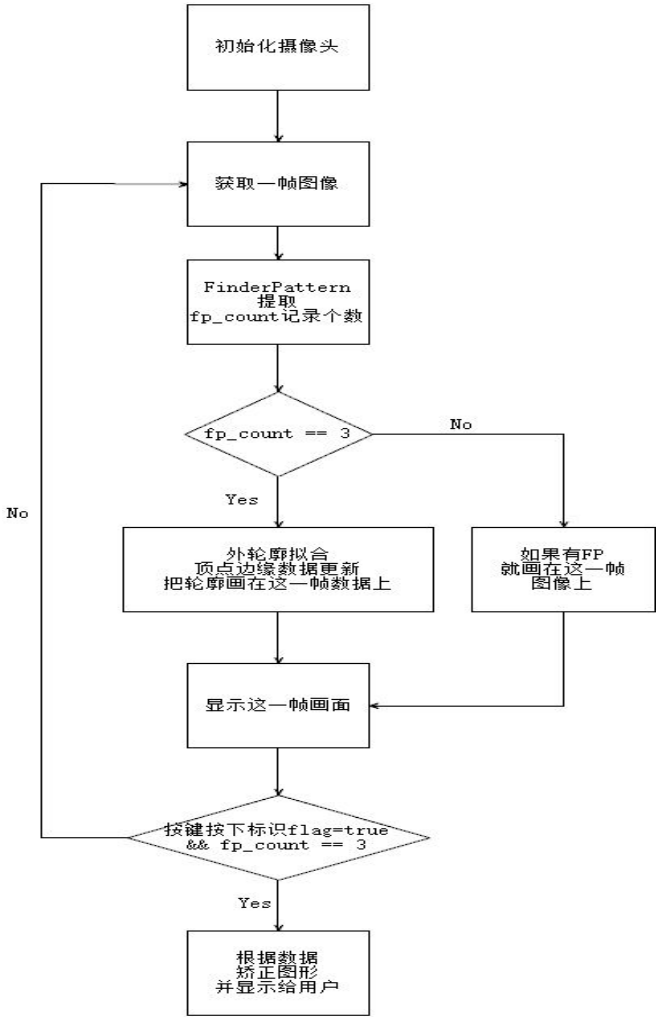


图 4-1 应用程序设计流程图

4.3 本章小结

在本章中，给出了应用设计的流程图及简要的提及了算法参数的设置问题。

第 5 章 柱面畸变矫正算法实验仿真

本章节阐述实现本文设计算法流程。通过在安卓平台实现算法，对算法效果进行实验测试。并分析算法的不足和优点。

5.1 仿真环境与内容

仿真运行环境：Android 5.1.1 CPU 为高通骁龙 650 的手机

程序开发环境：Android Studio 2.0 under Windows 10/Ubuntu 16.04

仿真内容：在安卓平台使用 Java 以及 JNI 调用 C++库的方式实现本文所述的柱面畸变矫正算法，并通过手机摄像头实时拍摄画面进行柱面二维码的识别与矫正。仿真测试整个算法流程的效果以及实际中的不足。

图 5-1 程序主界面图

如图 5-1 所示是本人实现的一个二维码预处理安卓应用，是一段时间的积累结果。这个应用选中自适应二值化后提取二维码的 FinderPattern 选项，并设置好掩模大小和偏移值，点击自适应阈值二值化按钮后可以查看在彩色图片上提取二维码的寻位图形的处理效果，用户可以自己调整掩模大小来查看效果变化，一

般默认即可。点击二维码边缘提取则开始本文所述的算法流程。这在本章第二小节阐述。其余的则是本人用于在安卓平台测试各个图形处理功能所额外添加的。

5.2 仿真方法与步骤

本章 5.1 小节简单介绍了仿真测试环境。仿真方法就是依据本文的算法流程一步步的执行。具体步骤如下所示：

图 5-2 二维码区域初提取

首先，如图 5-2 所示，手机显示变为横屏，开始打开摄像头提取二维码区域，在没有二维码时，不会画出任何图形。当有不是 3 个二维码寻位图形出现时，它们都被用红色的线画出。当寻位图形数目刚好是 3 个时，三个寻位图形四条边以四种不同颜色的画出，并且会用拟合的红线圈出二维码区域，可以看到，并不是边缘红线完全准确，此时没有进行拟合矫正。屏幕左上角显示分辨率和 FPS 信息。

其次，如图 5-3(a)所示，当用户看到 5-2 中已经基本圈出二维码区域后点击屏幕，进入到下一个 activity 中，进行进一步处理。此时的图片显示为原始图像的二维码的 ROI 图像，白线是原来的外围拟合边缘线。此时点击抛物线矫正按钮，按钮变为透视变换矫正按钮，图片也接着被处理并显示了出来。图片中的白线显示出了边缘线的最终处理结果。可以看到，这个是成功的。基本包围了二维码区域。

(a) (b)

图 5-3 边缘线矫正

(a) (b)

图 5-4 进一步处理结果图

接着，不断按下按钮，图形得到处理，如图 5-4 所示。

当然，本次演示图中，最终处理效果不是非常理想。但重复尝试后，可以得到如下图 5-5 所示的成功的结果图。

图 5-5 成功的矫正示例

图 5-5 所示的二维码是可以用另一台手机上一般的二维码扫描软件扫描后解码成功的。但是图 5-4(b)的则还是不行。多次测试，发现在某些情况下矫正效果是可以识别的。但是本文所设计的整套算法还是存在不足之处。这在本章下一小节全面的阐述。

5.3 仿真结果与分析

经过不断地按照 5.2 小节的方式测试，并且在多个安卓设备上测试程序，发现了如下的问题以及优点。下文一一地标明问题以及原因分析。

问题 1：二维码初提取时，直线边缘线和抛物线边缘线可能没有交点。依次顺时针或逆时针旋转手机，使得二维码逐渐呈现不同方位时，发现二维码倾斜较为接近角度绝对值 45 度于 x 轴或 y 轴时，会参数由于抛物线的曲率较大，出现错误。但是后期添加修正程序后寻位图形则完全不会出现分类错误，可以从图像

中边的颜色判断。

问题 1 分析：这个问题是这个方法本身的缺陷，抛物线拟合外边缘线在抛物线曲率较大时，单个寻位图形的一边带来的误差较大。只能避免这种情形，除非找到一种更好的方法可以得到某些准确的非二维码寻位图形的外边缘点用于拟合。但是实际上由于二维码各种各样其它特征并不明显，较难以实现这一点。目前只能避免这种情况。

问题 2：抛物线拟合矫正受到拍摄图片光度影响，可能出现错误。图片明暗不均匀，采用本文的 3.3 节这种矫正方法理论上可行，但是实际中黑白区分的边界值确定有一定的随机性。而且上下比值不一定各个图像都一样。然而实验发现，如果人工不断调整这些取值，确实是都可以矫正正确的。毕竟这是用于程序，人工调整还是缺少自动化。

问题 2 分析：在取黑白分界为 50 时，本身就是属于一种全局二值化方法了。当然受到光度的影响。可能如果能够找到一种方法，根据灰度值变化来区分会更好。而且二维码内容不同，黑白方块的分布概率不会一样，所以恐怕需要一种基于统计的方法来设计 3.3 节中两边界值的取值。

问题 3：矫正的结果受到边缘点的精度的影响。

问题 3 分析：透视变换存在一个较大的问题就是如果四个点略有偏差都会导致矫正出来的图形结果存在较大误差。即使对于平面上的二维码，在拍摄角度不正变为为一个不规则四边形时，如果无法精确的定位四个角点，也会矫正后识别失败。改善这个问题只能够使用高分辨率图像以及尽可能优化透视变换点的提取算法。

通过实验也发现了如下的优点。

优点 1：二维码的寻位图形识别准确性很高，基本不会出现错误的情况。

优点 2：寻位图形的四条边不管图形在那个方位上，基本都能够正确的分类。

优点 3：在偏斜角度不大时，二维码的方位型都能够正确的确定。拟合曲线能够基本包含二维码区域。算法能够在一定程度上适应不同的拍摄旋转角度。

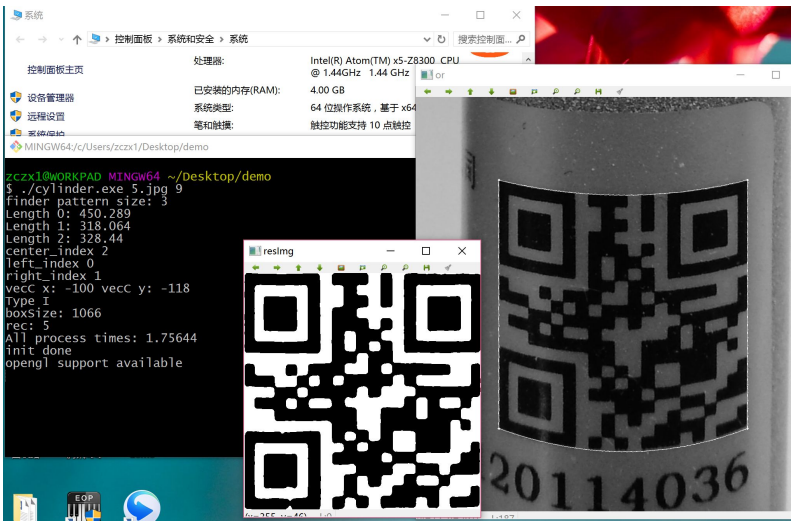
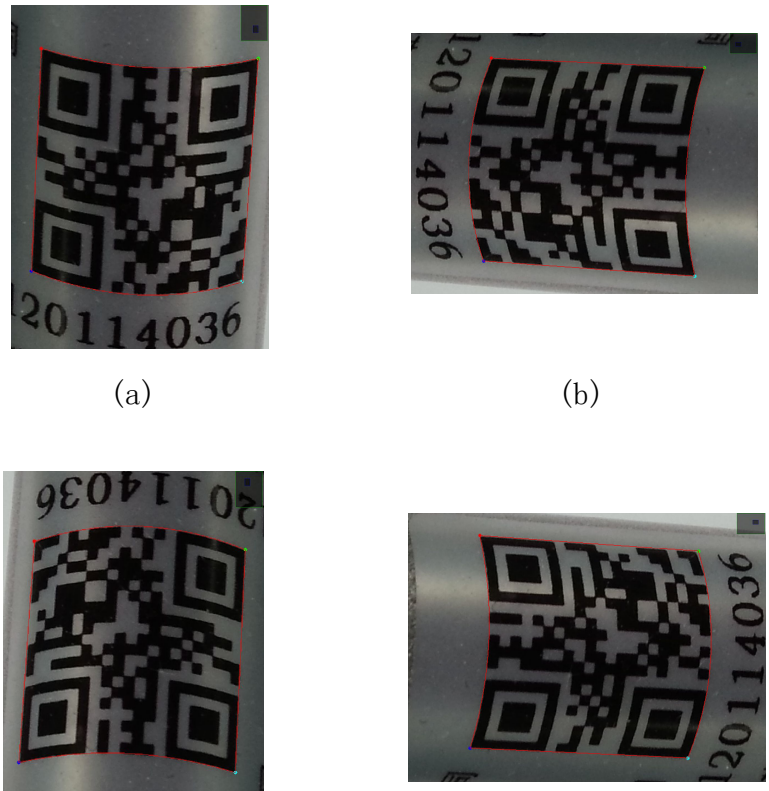


图 5-6 桌面平台图片处理测试图



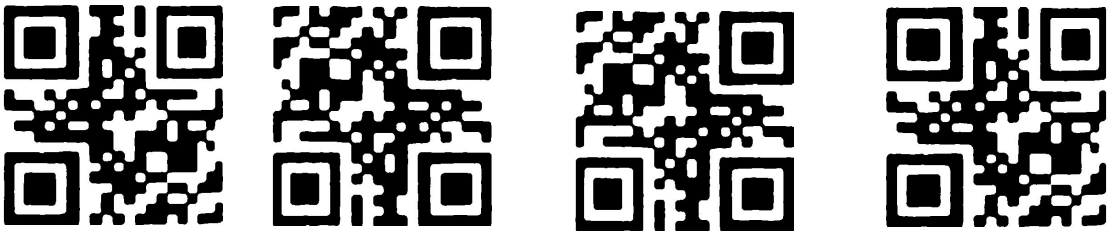
(a) (b)

(c) (d)

图 5-7 图片多方位处理结果图

在仿真实验过程中，还用了 C++按照类似 5.2 节的流程，测试了针对相机拍摄的图片的效果。实验中使用了 800 万像素的高精度图片进行处理。测试使用英特尔搭载 Atom 处理器 Z8300 的 Windows10 平板，一幅图像处理用时不到 2 秒。

图 5-6 显示了测试结果图。需要注意的是，原始图像二维码区域并不是两边垂直的，其中 or 图显示的是处理后的结果。在测试中，还需要把同一幅图像不断旋转 90 度，处理后得到如图 5-7 所示意的不同方位的原始二维码 ROI 图像，可以发现，算法仍然能够较好地正确包围二维码区域。



(a) (b) (c) (d)

图 5-8 图片处理矫正结果图

图 5-8 则依次对应于图 5-7 所示的矫正后的各个图像。可以看到矫正也不完美，有一边有一点窄。但是已经可以扫码成功了。

通过类似的图片实验发现，本文所述算法在实际处理图片时，如果可以手动根据实际处理情况来逐步调整拟合矫正参数，还是可以得到相对理想的结果的。

5.4 本章小结

在本章之中，在两个平台给出了算法实现的仿真结果与分析。通过仿真实验发现，本文的算法在外边缘拟合矫正以及整体二维码透视变换矫正上有其先天的不足之处，但是在二维码初提取，寻位图形获取，自动分类方面却表现出了较好的效果。最后的处理结果识别失败往往也是由于透视变换取点不够精确导致出现波浪形的矫正失真。但是二维码的所有信息已经被从复杂的背景中完全分离了出来。多次尝试或调整参数优化实现能够提升矫正效果。

第 6 章 总结与展望

本文设计以一整套算法流程来处理柱面畸变的二维码，并且在安卓平台和桌面平台都进行了实验仿真。本章节对本文的工作与研究进行总结与展望。

6.1 工作总结

本文的工作主要有两个方面。

一方面，本文总结了以前对于二维码寻位图形识别的研究成果，结合OpenCV3提出了一种寻位图形识别的算法，这是本文的一个小创新点。这种基于轮廓包含特征和面积特征的分离方法相比于使用比例的方法在图形畸变的情况下也能够较好地识别出寻位图形区域。当前，也有提到使用轮廓特征而非比例来分离二维码寻位图形的论文。只是使用的方法不同，而且也没有使用包围面积的比例特征。当然，实际应用为了精简，开发者应该自己实现图像预处理以及全局轮廓提取方法，而不是依赖于 OpenCV3 图形库。

另外一方面，本文尝试设计一整套流程来从复杂背景中矫正识别柱面畸变二维码。很多相关二维码研究论文则只是针对于二维码提取过程中的某个方面进行研究，全面阐述整个流程的基本没有。然而，想要做到尽善尽美比较困难，特别是还需要考虑到实际编程时的各种可能状况。因而本文把重点放在应用编程上，在一些已有的成熟的图像处理方法则使用了现成的 OpenCV3 函数。本文把整个过程都基本阐述了出来，并且实验仿真结果也呈现了出来。自然，本文的方法最后矫正结果并不是非常理想，只是在寻位图形提取与边分类上做到了基本完善。但是本文仍然可以作为后续研究者的参考，并为初涉二维码矫正这个处理问题的初学者提供其它论文没有的一些重要问题细节，以避免重复劳动。

6.2 研究展望

本文在二维码寻位图形和二维码区域包围分割和分类上效果较好。在矫正的

上却还有较多不足。对于高版本的二维码图像，需要高分辨率的图像源，然后切分成更多份组合才有良好的效果。而且曲度过大往往也会导致出来的结果不理想。所以仍然需要寻求一种好的方法。

$$\begin{aligned} x &= f(u,v) \\ y &= g(u,v) \end{aligned}, \tag{6-1}$$

如上述式子，设(u,v)为矫正前坐标，(x,y)为矫正后的坐标。那么柱面畸变矫正就是希望找到这样的函数映射关系，使得前后坐标有一定的对应。可以把本文掩码中上下曲边到另一幅图像中的点一一对应，然后拟合求这种变换关系。透视变换也是具有这样的表达式，只不过只是二元一次的形式。对于柱面的畸变图形，这样的函数不好找。目前还没有头绪，想必在以后，会发现一种变换关系，使得依靠已知的原图中的点，设置目标图像中的点，并结合这种变换关系，拟合求出变换函数各个参数，然后逐点变换，最后得到结果。但这只是个人的一种想法，目前还没有看到有学者提出数学模型能够完整地表达这种关系。已有的几何变换效果未知，也缺乏数学推导。不过一旦找到合适的关系。这个问题也就彻底解决了。不过目前有使用格子法来重构二维码，不用图形矫正，这自然是目前最好的方法了。

参考文献

[1] Tong L, Gu X, Dai F, QR Code Detection Based on Local Features[C], Proceedings of International Conference on Internet Multimedia Computing and Service, ACM, 2014, 319

[2] Chen Q, Du Y, Lin R, et al, Fast QR Code Image Process and Detection[M], Internet of Things, Springer Berlin Heidelberg, 2012, 305~312

[3] Sun H, Xia H, Dong N, Research on pre-processing of QR Code[J], Proc Spie, 8916, 2013, 89164B~89164B-7

[4] Qi H, Lu X, Lu L, A localization algorithm for distorted or rotated QR code[C], International Conference on Computing, Communication and NETWORKING Technologies, IEEE, 2014, 1~4

[5] Kang E, A rectification method for quick response code image[C], International Symposium on Consumer Electronics, 2014, 1~2

[6] Li X, Shi Z, Guo D, et al, Reconstruct argorithm of 2D barcode for reading the QR code on cylindrical surface[C], International Conference on Anti-Counterfeiting, Security and Identification., 2013, 1~5

[7] Fitzgibbon A, Pilu M, Fisher R B, Direct least square fitting of elipses[J], IEEE Transactions on Pattern Analysis & Machine Intelligence, 21(5), 1999, 476～480

[8] 毛星云，冷雪飞，OpenCV3 编程入门[M]，电子工业出版社，北京，2015

[9] 任广千，谢聪，胡翠芳，线性代数的几何意义[M]，西安电子科技大学出版社，西安，2015

[10] 李建华，二维条码图像处理算法及其 VLSI 设计研究[D]，博士学位论文，电子科技大学，2013

[11] Richard Szeliski，计算机视觉算法与应用[M]，清华大学出版社，北京，2012

[12] 卫晋伟，戴曙光，穆平安，基于形态学和 Hough 变换的 QR 码校正与定位方法[J]，电脑与信息技术，18(6)，2010，32~35

[13] 司国东，陈仲，柱面二维码识别算法的设计与实现[J]，现代计算机:普及版，(8)，2013，25~27

[14] 孙富明，王沁，李笑盈，圆柱面深度图像变换的仿真与实现[J]，系统仿真学报，21(21)，2009，6925~6929

[15] 苏成志，王恩国，郝江涛等，平面几何测量中的图像畸变校正[J]，光学精密工程，19(1)，2011，161~167

致　谢

由衷感谢我的导师谭洪舟教授和项目组陈荣军老师，本文是在他们的指导下完成的。特别是陈荣军老师，提供了很多书籍资料给我参考学习。并且在我遇到问题是给出了众多宝贵的建议。谭洪舟教授带领的实验室团队氛围非常好，在实验室学习期间，学习到了不少宝贵的经验和知识。

冯展鹏

2016 年 4 月