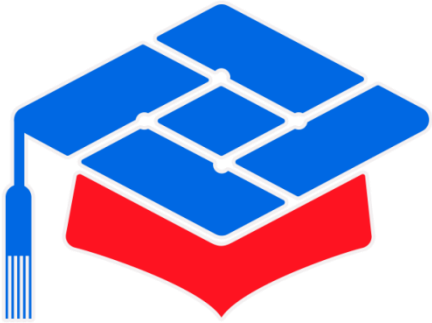

第九届

全国大学生集成电路创新创业大赛

报告类型：技术文档

参赛杯赛：海运捷讯杯

作品名称：基于 FPGA 的图像采集及识别和机械臂分析及控制系统设计

队伍编号：CICC0901373

团队名称：超绝队

目录

快速预览简介

关键技术

1. 多色目标分割技术

色彩空间转换：将 RGB565 数据转换至 YCbCr 空间，突出色差特征；

多阈值二值化：分别对蓝、红、黄、黑四色目标进行阈值分割，提高识别精度；

形态学滤波：开闭运算消除噪点与小斑块，分离粘连目标，保证后续连通域分析可靠。

2. 高效连通域与质心定位算法

两遍扫描 + 8 连通域：经典算法结合 FPGA 并行处理，实现实时标记与统计；

细粒度特征提取：自动寻找最大标签区域，计算质心和最高点坐标，用于后续运动学映射。

3. 查表式逆运动学与分段拟合

ROM 查表法：预存机械臂 0～4 号舵机角度映射，替代复杂运算，减少逻辑资源占用；

分段线性插值：在查表基础上实现插值调速，平滑运动，克服惯性影响。

4. 实时帧级状态机控制

帧间切换机制：顶层状态机根据图像处理结果触发抓放流程；

PWM 驱动模块：灵活调节频率与占空比，实现舵机与气泵同步控制。

作品亮点

1. 资源高效：

查表式逆运动学 + 并行连通域处理，节省 DSP 运算与逻辑资源；

2. **实时性强：**

66 ms 图像识别周期，20 s 完成多目标分拣，满足赛题高时效要求；

3. **精度可调：**

基于线性插值的 PWM 驱动可实现不同速度下的高精度控制；

4. **模块化设计：**

图像、运动、控制、驱动四大模块解耦，便于扩展与移植；

5. **自主扩展板：**

AWC_C4 & 升腾互联板 + 舵机驱动板，实现信号集成与电源管理一体化。

1 系统架构分析

1.1 系统总体设计方案

系统以完成第九届集创赛海云捷讯杯的难度二为目标进行设计，其中软件主要功能使用 AWC_C4 青春版开发板和野火升腾 mini 版开发板（XC7A100T）实现（皆为 FPGA）。其系统框图如图 1 所示。

此系统中的 FPGA 设计主要分为两大模块，即图像采集及识别模块和机械臂分析及控制模块，并且利用顶层模块将两者相结合。

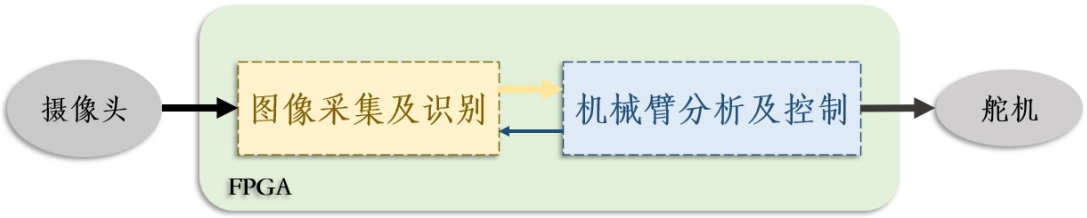

图 1 系统总体设计方案框图

1.2 技术方案分析比较

经队伍讨论得出，以实现难度二为设计目标，有多种方案可以达到理想效果，于是我们分别针对每一具体功能的实现方法进行列举，并对各方法进行分析比较，最终确定方案。**此小节旨于记录本队伍在此系统设计前确定方案的过程和设计过程中遇到的问题及其解决方法**。这里主要对图像颜色识别模块、图像形状识别及坐标获取模块、逆运动算法模块、机械臂控制方式等进行方案的列举与通过权衡后最终选择的方案。

1.2.1 图像颜色识别模块的论证与选择

方案 1：搭建卷积神经网络，通过数据预处理、构建模型、训练模型、评估模型的方式进行深度学习。

方案 2：利用摄像头采集直接得到的 RGB565 色域空间进行阈值分析。

方案 3：采用 YCbCr 色域空间对图像信息进行阈值分析，利用 Y 分量进行物体与背景分离，再根据 Cb 和 Cr 分量区分出四种颜色。

综合比较以上三种颜色识别方案，由于在卷积神经网络方面经验较少故没有选择方案 1；RGB565 色域空间受光照影响较大，在不同的环境下容易出现误判的情况，且经 MATLAB 仿真验证四种颜色在 YCbCr 色域空间内较好区分，故选择方案 3 实现图像的颜色识别。

1.2.2 连通域算法模块的论证与选择

a) 标签更新逻辑方案：

方案 1：使用种子填充法，将像素点 A(x,y)作为种子赋予一个 label，然后将该种子相邻的所有前景像素都压入栈中，弹出栈顶像素，赋予其相同的 label，然后再将与该栈顶像素相邻的所有前景像素都压入栈中。

方案 2：使用两遍扫描法，利用 fifo 进行行缓存。在第一行扫描中对每个像素 A(x,y)赋予 label 并往左回溯固定像素。在第二行扫描时，更新每个像素的 label。而后使用行计数器对应像素和标签输出。

标签更新逻辑的选择：对于方案 1 而言，像素点是一个接一个的传入的，无法将全部的像素点传入栈中，而且需要每个考虑从栈中读出的像素的领域，其在 fpga 中的实现相对来说比较困难；方案 2，在当前行对像素打上标签并往左回溯，下一行对上一行更新标签。不仅可以保证标签更新的稳定性和正确性。同时也具有很强的实时性，fpga 相对也容易实现。综上所述采用方案 2.

b) 连通域领域数量：

方案 A: 采用八连通域，在像素 A(x,y)传入后，考虑其上面 3 个，左边 1 个，右边 1 个，下面 3 个 。考虑到像素是一个一个传入的，下面 3 个和右边 1 个不用考虑。

方案 B: 采用四连通域，在像素 A(x,y)、传入后，考虑其正上方 1 个，左边 1 个，右边 1 个，正下方 1 个。同样正下方 1 个和右边 1 个不用考虑。

连通域领域数量的选择：对于连通域数量选择来说，自然是考虑的领域越多，稳定性和准确性也越高，但是消耗的逻辑资源也会变多。因为后面形状识别和求找质心和角度需要相对稳定和完整的图像。且比赛允许使用 fpga 外接平台。综

上所述，选择方案 A。

综合考量下来，选择方案 2 + 方案 A 来实现连通域算法。

1.2.3 形状识别及坐标获取模块的论证与选择

方案 1：搭建神经网络进行深度学习，来对目标进行形状识别及坐标获取。

方案 2：在颜色识别模块后提供的独特二值化图像下，采用水平、垂直投影获取边界，进行逻辑判断调整后得到三组三种形状各自的上下左右边界坐标，统计区块内各自面积进行形状的区分，利用边界坐标获取中心坐标。

方案 3：利用霍夫变换检测直线，利用物体轮廓的直线条数不同以区分出正方形和六边形，剩下的即为圆形。

方案 4：采用连通域算法，将二值化之后互不粘连的物体打上不同的标签，以此可以将不同的物体单独分离开来。然后用公式找到质心坐标：

$$\overline{x} = \frac{\sum_{i=1}^{N} x_i}{N}, \quad \overline{y} = \frac{\sum_{i=1}^{N} y_i}{N}$$

其中 N:前景像素的总数，$\sum_{i=1}^{N} x_i$所有前景像素x坐标之和，$\sum_{i=1}^{N} y_i$所有前景像素 y 坐标之和，$\overline{x}$:质心x坐标，$\overline{y}$:质心 y 坐标。

同时将摄像头固定在一个高度，此时不同形状的物体二值化后的面积有不同的范围，凭借面积的不同可以识别不同形状。

综合比较以上三种形状识别方案，我们队伍对神经网络相对不熟悉，故不选择方案 1；方案 3 则难以利用 FPGA 实现；由于难度二物体不规则摆放，水平垂直投影算法获得的边界和坐标不准确，故不选择方案二。利用连通域算法可以将不规则摆放的物体单个分离开来，以此获得的物体方便求找质心坐标和要旋转的角度，以及利用面积进行的形状识别也稳定很多。所以选择方案 4.

1.2.3 旋转角度获取模块的论证与选择

方案 1：连通域算法提取物体后，找到物体的最高点和质心。利用质心的 x 坐标，最高点的 y 坐标重新生成一个点，然后利用这三个点重新生成一个三角形。对质心所在的角进行反正切运算。而后经过运算旋转成不同的形状。

方案 2：由于是同样的正方形物块，外接圆的大小在理论上是一样的，则可以生成一个外接圆的最高点。利用物体的质心和最高点求出圆的半径。同时三个点也可以生成一个三角形，直接反正切求出的角度即要旋转的角度。

综合比较以上两种方案。对于方案 1 来说，三个点形成的是一个直角三角形，反正切公式比较简单也好求，但是Δy和Δx的比值太大会导致线性拟合的反正切误差较大，该种情况要额外考虑；对于方案 2 来说，要用到平方和开方运算，这个过程会额外造成误差，而且三个点形成的三角形不是直角三角形，角度求解的逆运算相对比较复杂和麻烦。综上所述，选择方案 1。

1.2.4 逆运动算法模块的论证与选择

方案 1：采用传统的机械臂逆运动解析法，即确定机械臂末端执行器的坐标，根据机械臂的几何结构和连杆之间的关系和旋转矩阵，通过矩阵乘法计算出各个关节的角度，从而得到机械臂末端到达目标位置时各关节的角度。

方案 2：根据机械臂连杆的几何关系，通过计算机用二元一次三角函数方程组解出所有舵机的角度，将结果存入 ROM。模块只需输入机械臂末端执行器的坐标，经简单运算得到 ROM 地址，即可读出各关节角度。（具体内容请参考 2.1.3）。

综合比较以上两种方案，方案 1 中利用 FPGA 进行矩阵运算与反正切运算所消耗逻辑资源、DSP 资源量极大，且 EP4CE6F17C8 芯片对应资源少，难以实现传统逆运算算法；方案 2 算法简单易懂，主要消耗存储资源，消耗逻辑资源极少，最终选择方案 2 实现机械臂逆运动算法。

1.2.5 机械臂控制方式的论证与选择

方案 1：采用总线通讯，用 UART 双线转单线，通过发指令的方式控制舵机，并用指令读取实际角度值，与预设值对比从而进行 PID 自动控制。

方案 2：采用结合线性插值的 PWM 驱动，用传统脉冲宽度调制的方法控制舵机角度，通过改变 PWM 波占空比实现角度控制，并且利用线性插值实现舵机速度可控。

综合比较以上两种方案，方案 1 中的指令发送涉及两级缓存器、指令的发送

和接收所涉及的时序冗杂，且异步时钟缓存器易发生数据丢失；方案 2 简单易懂，方便实现。并且舵机里自带单片机，可由商家上位机软件实现 PID 参数修改和运动控制，最终选择方案 2 实现速度可调的机械臂控制。

1.3 系统架构工作原理

本系统使用 Altera-Cyclone IV系列的 EP4CE6F17C8 芯片实现机械臂控制，使用 Xilinx Artix7 系列的 XC7A100T 芯片实现图像处理。

将摄像头 OV5640 模块采集到的 RGB565 色域的数据流信息转换成 ycbcr 颜色空间。接着根据 y,cb,cr 信号的阈值不同，先将蓝色物体以二值化的形式提取出来。然后经过消除粘连模块，再通过连通域算法，把最高的物体单独分离出来。然后传入形状识别和质心求找模块。同时找到物体传入的最高点。之后传入角度求取模块，在质心角度和形状信息均找到后。统一传入寄存器发送给机械臂。当接受到机械臂的复位信号后，开始下一帧图片的拍摄。当检测到图像中没有物体时，切换颜色重新拍一帧图像。识别颜色的顺序是 1 蓝—>2 红—>3 黄—>4 黑。

接收到图像识别完成信号以后，顶层模块读出寄存器中的坐标，将其输入逆运动解算模块，得到机械臂抓取物体的姿态信息；同时，颜色和形状输入给放置模块，得到机械臂放置物体的姿态信息。将抓取和放置姿态信息传输给状态控制模块，状态控制模块按:1 复位—>2 抓姿态—>3 复位—>4 放姿态—>5 复位的顺序，分别将舵机角度传给模式选择模块，当机械臂处于从 1 到 2 或从 3 到 4 状态时，选择线性插值模式，达到控制机械臂运动速度的效果；当机械臂处于从 2 到 3 或从 4 到 5 状态时，选择全速模式，达到使机械臂以最快速度运动的效果。最后，模式选择模块将经处理的角度信息分别输入 5 个 PWM 生成模块，分别实现 0、1、2、3、4 号舵机的控制，并同时实现气泵吸和放的控制。当状态控制模块刚进入“3 复位”状态时，向图像处理模块输出使能信号，识别下一个物块的信息。

当一个物体完成运送后，从寄存器即可读出下一组物块信息，进入下一个循环，直至所有规定物体全部运送完毕。系统流程图如**错误!未找到引用源。**所示。

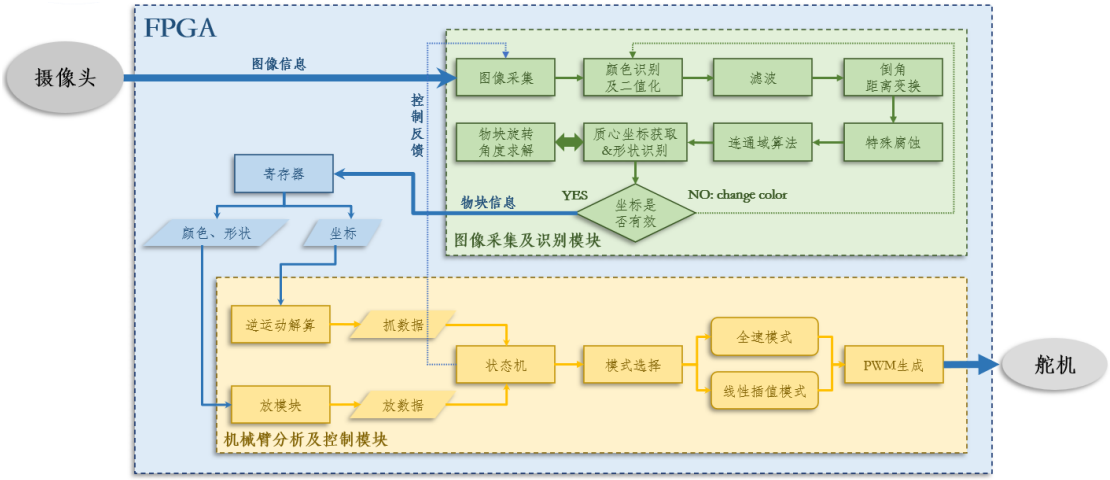
FPGA
摄像头
图像信息
控制反馈
图像采集
颜色识别及二值化
滤波
倒角距离变换
物块旋转角度求解
质心坐标获取&形状识别
连通域算法
特殊腐蚀
寄存器
颜色、形状
坐标
物块信息
YES
坐标是否有效
NO: change color
图像采集及识别模块
逆运动解算
抓数据
状态机
模式选择
全速模式
线性插值模式
PWM生成
舵机
放模块
放数据
机械臂分析及控制模块

图 2 系统流程图

2 关键技术分析

在此节中主要将介绍到此作品的软件设计、硬件设计与机械臂结构设计。

软件（FPGA）设计主要分为两大模块：图像采集及识别模块和机械臂控制模块。并利用顶层模块将两者相结合。

硬件设计为两块自制电路板：AWC_C4 & 升腾连接板与 PWM 舵机驱动板。

机械臂结构主要采用众灵科技的机械臂套件，并在此基础上进行改进。

2.1 基于 FPGA 的图像采集及识别系统设计

对于图像采集及识别模块，主要分析图像识别模块总体设计、颜色识别算法设计、连通域算法模块、形状识别及坐标获取算法设计、反正切求角度模块等。

2.1.1 图像识别模块总体设计

1 设计内容

图像识别模块总体设计主要分为四个模块：

a) 图像颜色识别模块：对红、黄、蓝、黑四种颜色进行区分。

b) 连通域算法模块：给同一帧图像中互不连通的物体打上不同的标签，仅将标签为 1 的物体提取出来。

c) 形状识别及坐标获取算法模块：根据二值化分离出的二值图片，找到物体的质心、最高点，同时依据前景像素的面积识别物体

d) 反正切求角度模块：利用质心的坐标，最高点 y 坐标生成一个点，构成一个三角形，而后求找角度。

2 图像识别系统流程

由图像采集模块输入图像信息。对数据流数据进行 YCbCr 颜色阈值分析后可分别得到单个颜色的二值化图像。而后进行腐蚀膨胀（开运算）进行滤波去除噪点和正方形物块的相互粘连，再经过特殊腐蚀完成图像的预处理。接着传入连通域算法，将标签为 1 的物体提取出来。传入形状识别及坐标获取算法模块时进

行判断是否有物体。如果是，则找到质心，最高点和形状信息；如果没有，那么就切换颜色。得到质心等信息后，传入反正切求角度模块。找到应该旋转的角度后通过寄存器传入机械臂。在机械臂传入复位信号后再拍一帧。如**错误!未找到引用源。**所示。

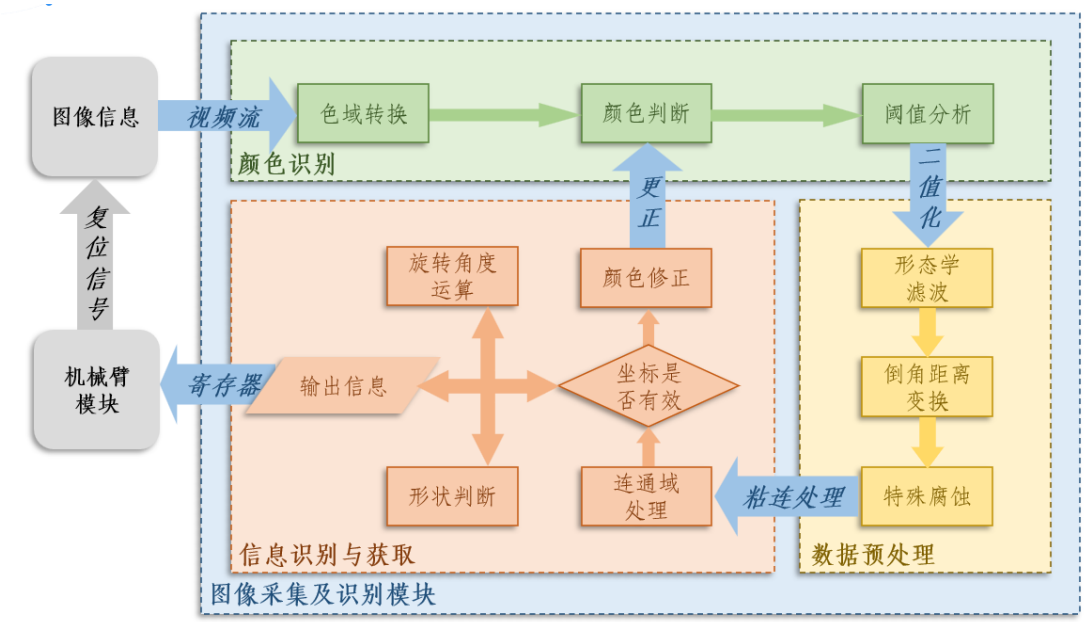

图 3 图像识别系统流程图

2.1.2 图像颜色识别算法设计

1 预想分析

由 OV5640 摄像头得到数据为 RGB565 格式，经实验此色域空间受光照影响较大，故采取 YCbCr 色域空间进行颜色识别。

2. 仿真与结论

如图 4 所示，对实物拍照后导入到 MATLAB 分析，得到 YCbCr 色域空间下不同颜色不同分量通道的结果。

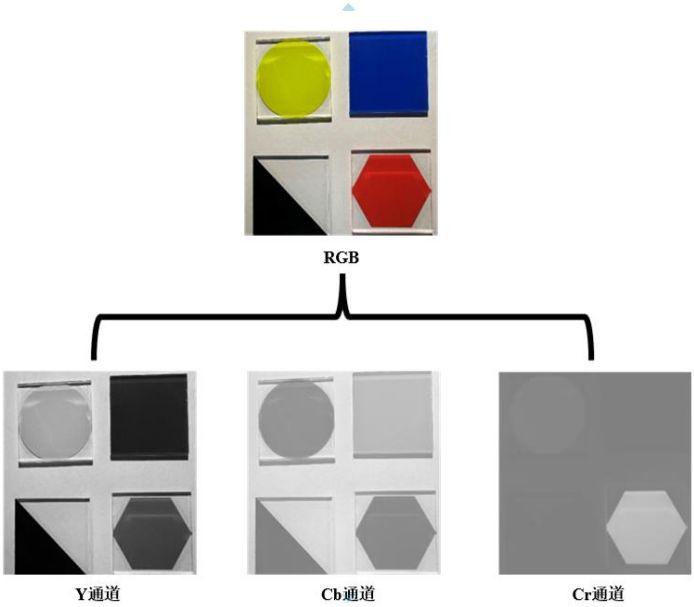

图 4　RGB 转 YCbCr 色域后结果图

从图 4 中可以得出以下结论（针对红、黄、蓝、黑四种颜色）：

a)　红色：Y 分量比黑色大且 Cr 分量在四种颜色里最大。

b)　黄色：Y 分量在四种颜色里最大且 Cb 分量在四种颜色里最小。

c)　蓝色：Y 分量比黑色大、比红色小且 Cb 分量在四种颜色里最大。

d)　黑色：Y 分量在四种颜色里最小。

以此进行条件设定，对每组数据进行选择，识别到为所需颜色时，输出 1，否则输出 0，以此分别得到对于红、黄、蓝、黑四种颜色的二值化图像，以供后续模块使用。

2.1.3 连通域算法模块设计

1 预想分析

给二值化图像中互不接触的物体打上不同的标签（1~15）最多识别 15 个物体。为了让标签打的更为准确，采用八连通域算法以及两遍扫描法。由于要对标签进行更新，利用两个数组来分别存储当前行以及上一行的标签。在一行结束时，用使用 for 循环来对存储上一行的数组进行更新。利用 FIFO 进行行缓存，保证时序对齐。在实际运用中由于二值化的图像不稳定在特定情况下可能出现要往左

进行标签“回溯”的需要，即在标签更新时对当前像素的左侧进行某些次数的便签更新。

a. **两遍扫描法：**在一帧图像的前一行打上标签，第二行进行上一行标签的更新。给互不相连的物体打上不同的标签，以此达成分离物体的目的。

b. **八连通域：**指定像素的上下左右以及对角线方向上的 8 个像素为相邻像素。在实现过程中对某一像素做具体判断的时候，只需要考虑头上三个以及左边的像素即可。

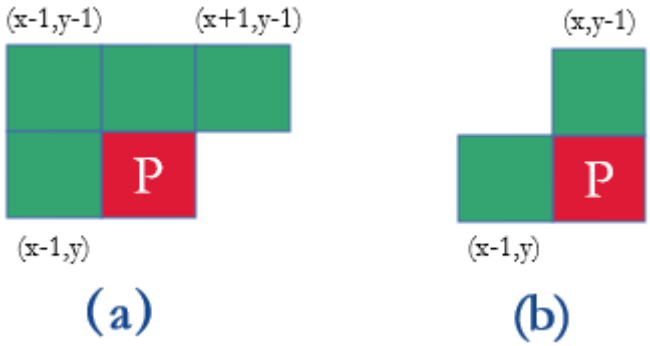

图 5 （a）为实现八连通域，（b）四连通域

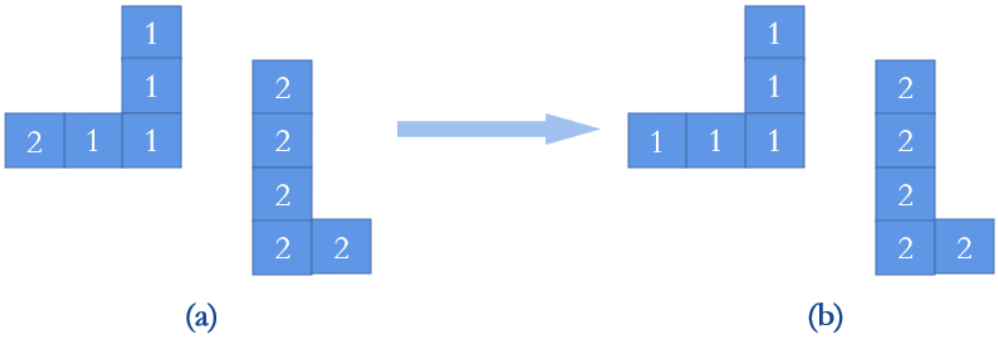

图 6 （a）为更新前标签图，（b）为更新后的标签图

经过实践表明，此预想分析可行，出现的图像稳定和完整可以使用于之后求坐标形状和旋转角度。

2 连通域算法仿真验证

下面是利用 **vivado 自带的仿真**进行仿真验证的图片：

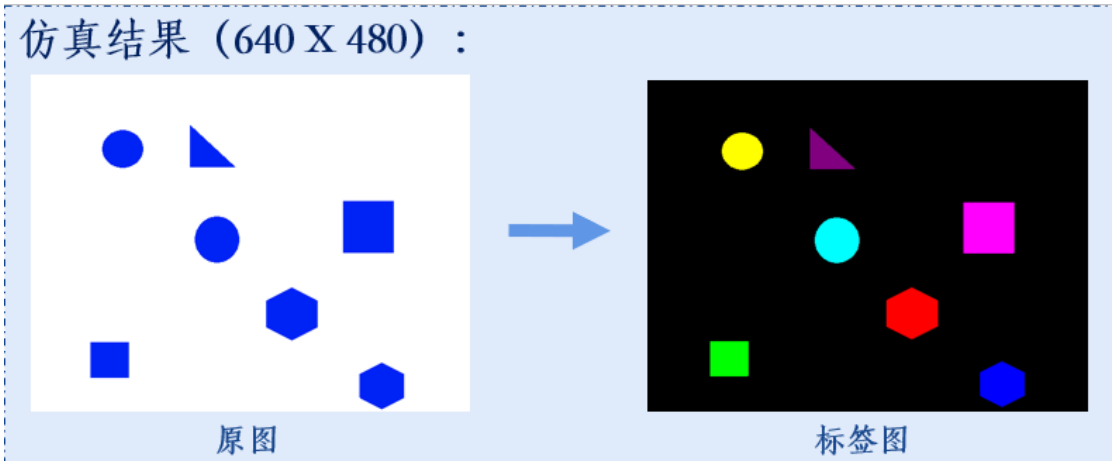

图 7 标签效果图

根据标签的不同打上不同的颜色。仿真表明，该连通域算法的模块正确的给不同的物体打上了不同的标签。如此便可单独将某一标签的物体单独提取出来:

图 8 标签为“1”的图片

在图片仿真中，像素数据是从右下角开始传入，所以给右下角物体打的标签为 1。

3 问题发现

由于二值化后的图像边缘不稳定可能出现反”L”的情况。在这种情况下，无论是上板还是二值化图像都会缺少一块，导致物体不完整，这种情况较为常见的出现在圆形和正方形上面，会较为严重地影响坐标和形状识别。经过实际验证，缺少的那一块物体的标签要大 1。比如：

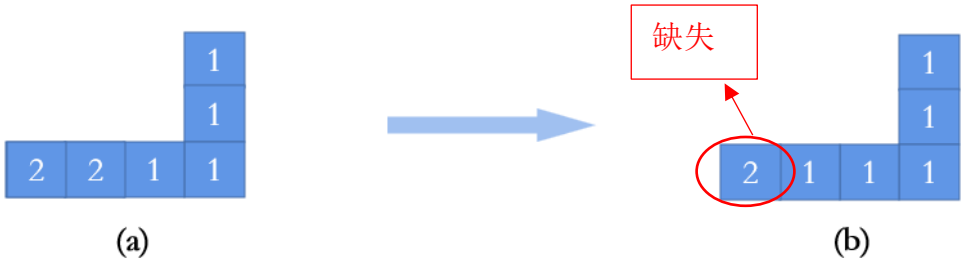

图 9 （a）无回溯更新前标签，（b）有回溯更新前标签

仿真的情况为：

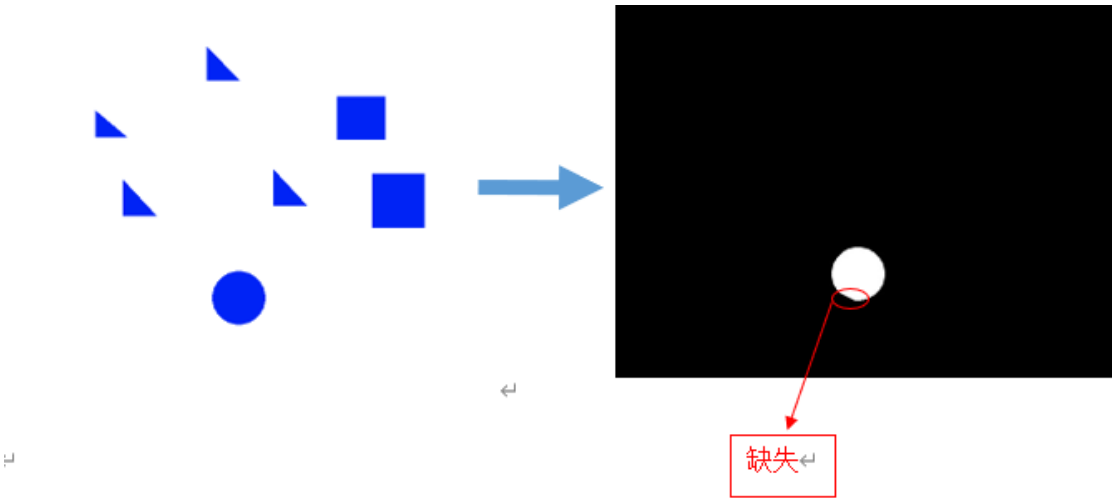

图 10 仿真展示物体缺失效果

实际上板验证的情况为：

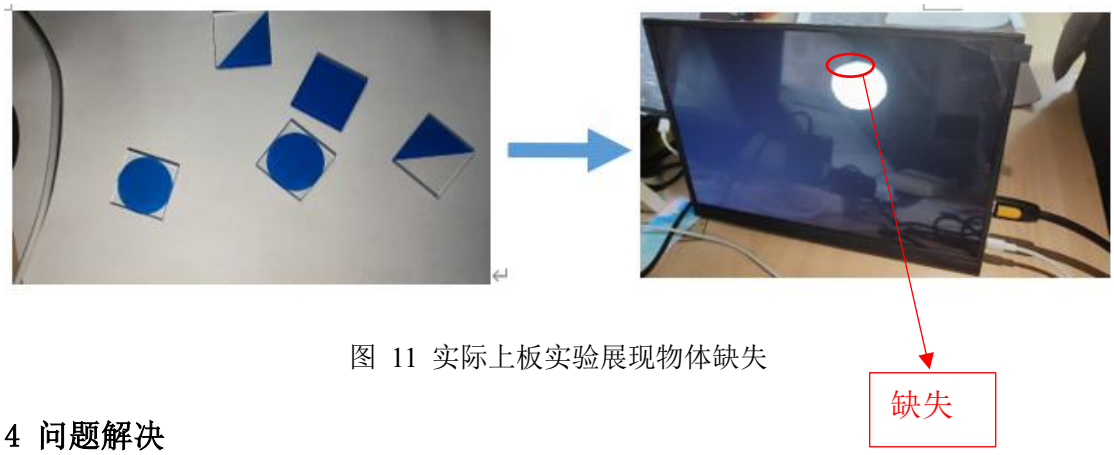

图 11 实际上板实验展现物体缺失

4 问题解决

这种问题解决起来十分容易。只需要增加往左方回溯的次数即可，这样得到的物体稳定且完整如：

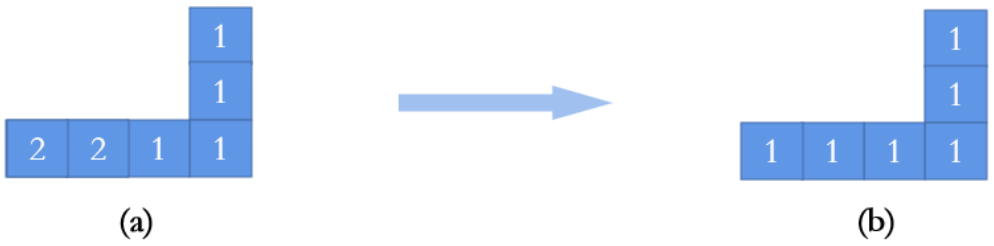

图 12 （a）有回溯更新前，（b）无回溯更新前

仿真的情况为：

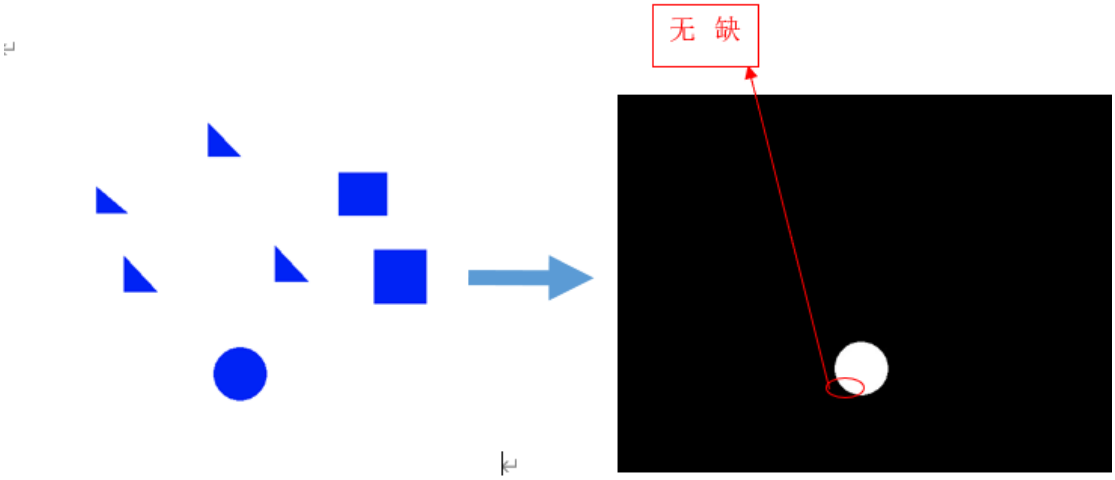

图 13 问题解决后的仿真图

实际上板实验的情况为：

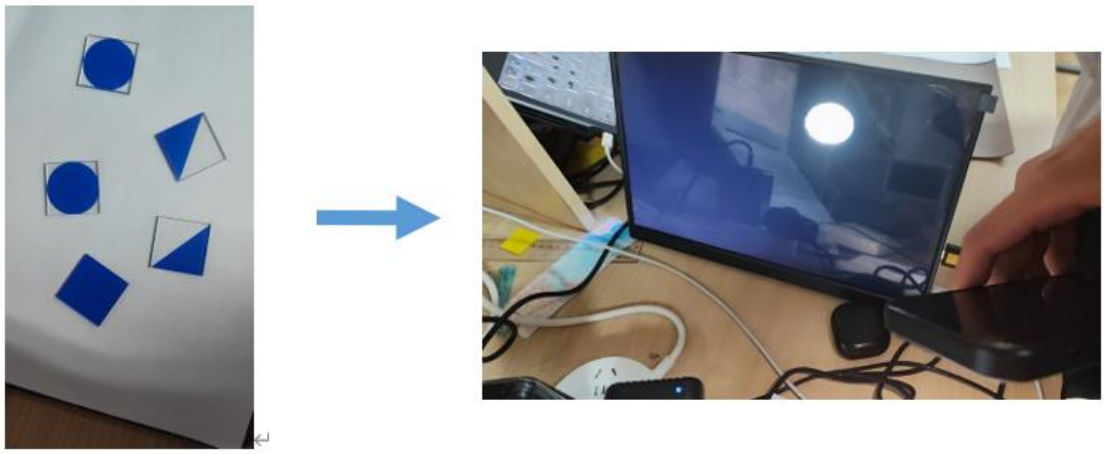

图 14 解决问题后无缺失的上板效果图

2.1.4 形状识别及坐标获取算法设计

1 预想分析

分别统计所有前景像素的 x 与 y 坐标之和，而后分别除前景像素的面积，以此得到该物块的质心坐标。同时找到输入像素的最高点，为之后求算角度做准备。同样总计前景像素的面积后，根据不同形状面积范围获得形状信息。

公式为：

$$\overline{x} = \frac{\sum_{i=1}^{N} x_i}{N},\ \overline{y} = \frac{\sum_{i=1}^{N} y_i}{N}$$

其中，N:前景像素的总数，$\sum_{i=1}^{N} x_i$是所有前景像素x之和，$\sum_{i=1}^{N} y_i$是所有前景像素 y 坐标之和，$\overline{x}$质心x坐标，$\overline{y}$:质心 y 坐标。

经过仿真和实际验证，在无噪点的情况下，该算法所求得的质心坐标以及形状信息正确且稳定。

2 仿真验证及结论

此处以正六边形为例，找出其最高点和最低点来验证求取质心的正确性。

原图：

图 15 传入该模块的图像数据

仿真波形：

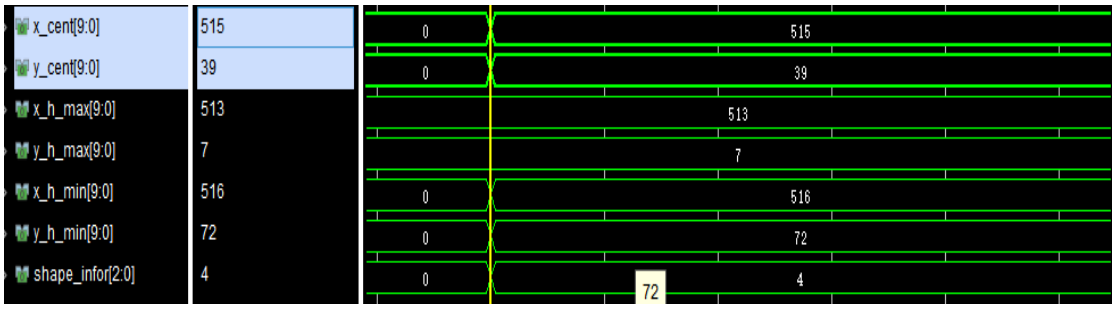

图 16 该模块的仿真波形图

其中，形状信息 shape_infor 的 1-4 对应的形状信息为：1-三角形；2-正方形；3-圆形；4-六边形。

x_h_max 与 y_h_max 是传入的第一个像素（最高点）的坐标。x_h_min 与 y_h_min 是第最后一个传入像素（最低点）的坐标。对于没有噪点的正六边形来说，其质心坐标 x_cent 和 y_cent 分别为最初于最后像素对应坐标相加除 2。由波形可以看出，求得的质心与最初最终像素坐标所求质心坐标相差无几。

shape_info 为形状信息。根据不同物块的面积确定是什么形状。下面的表格是不同形状对应的面积大小关系：

表 1 形状与面积大小对应表

形状	面积大小关系
三角形	最小
圆形	最大
正方形	大于六边形小于正方形
六边形	大于三角形小于圆形

同时，该模块还兼具传出使能反正切求角度模块，传出形状角度坐标数据，检测是否要切换颜色等功能。当最高点和质心找出后，拉高角度使能信号进行角度求解。在反正切角度模块传入完成信号后，传出数据。当一帧图像传入完成后，发现面积小于一定值，则切换颜色，同时向机械臂传入全零信号。

2.1.5 反正切求角度模块设计

1 预想分析

在得到图像最高点以及质心数据信息后，利用**质心 y 坐标**，**最高点 x 坐标**再**生成一个点**。三个点可以生成一个**直角三角形**。对质心所在的角进行反正切(cordic ip core)。

然后假设物体要旋转成一个固定的姿态。如下文“2 数学模型建立”中每张图的红色物体，即要旋转成的目标姿态。利用公式即可求出要旋转的角度。

2 数学模型建立

由于进行反正切的角如果是钝角，那么就会出现负值的问题。同样，如果两个点挨得很近则会导致用线性拟合达成的反正切求解不准确。所以对最高点的x坐标和质心的x坐标进行大小判断。分为$x_{高}$（最高点x坐标）在$x_{质心}$（最低点x坐标）左边，右边，附近（左右相差不超过六个像素）三种情况。对于每个形状来说，三种情况求解旋转角度的公式不一样：

① $x_{高}$在$x_{质心}$左边：

$$\theta = \tan^{-1}（（y_{高} - y_{质心}）/（x_{质心} - x_{高}））$$

② $x_{高}$在$x_{质心}$右边：

$$\theta = \tan^{-1}（（y_{高} - y_{质心}）/（x_{高} - x_{质心}））$$

③ $x_{高}$在$x_{质心}$附近

$$\theta = 常数$$

a) 圆形

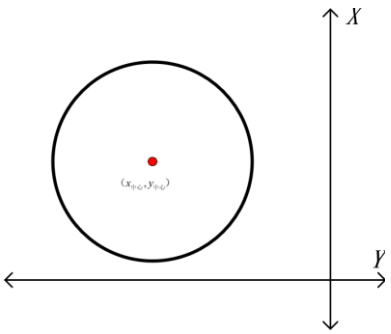

图 17 圆形

无论圆形处于什么姿态，都不需要旋转

b) 正方形

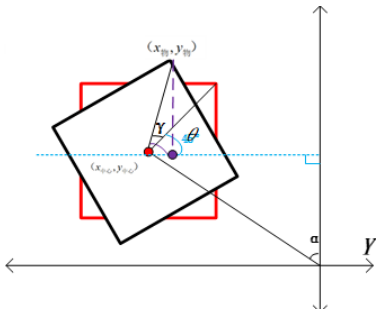

图 18 最高点 x 坐标在质心 x 坐标右边

如图 18 是最高点 x 坐标大于质心 x 坐标的情况，黑色物块是假定正方形的实际姿态，红色物块则是该正方形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = \theta - 45^\circ$

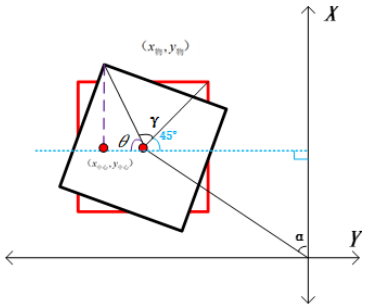

图 19 最高点 x 坐标在质心 x 坐标左边

如图 19 是最高点 x 坐标小于质心 x 坐标的情况，黑色物块是假定正方形的实际姿态，红色物块则是该正方形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = 135^\circ - \theta$

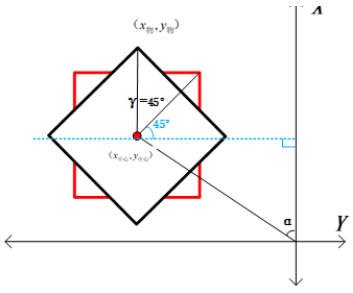

图 20 最高点 x 坐标约等于质心 x 坐标

如图 20 是最高点 x 坐标约等于质心的情况，黑色物块是假定正方形的实际姿态，红色物块则是该正方形要旋转成的目标姿态。此时线性拟合达成的反正切算法误差较大，让旋转的角度为一个固定的值。

旋转的角度为：$\gamma = 45^\circ$

c） 六边形

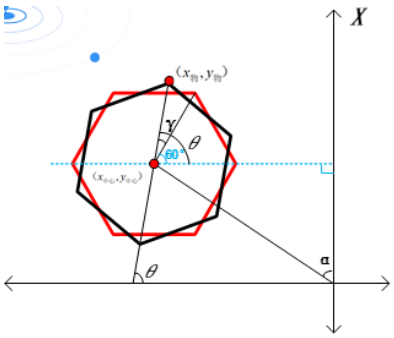

图 21 最高点 x 坐标在质心 x 坐标右边

如图 21 是最高点 x 坐标大于质心 x 坐标的情况，黑色物块是假定六边形的实际姿态，红色物块则是该六边形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = \theta - 60^\circ$

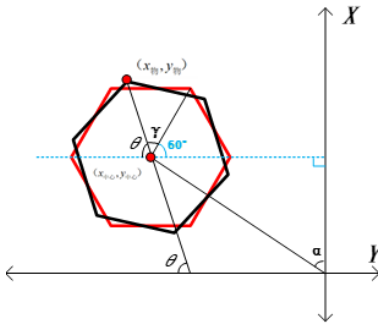

图 22 最高点 x 坐标在质心 x 坐标左边

如图 22 是最高点 x 坐标小于质心 x 坐标的情况，黑色物块是假定六边形的实际姿态，红色物块则是该六边形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = 120^\circ - \theta$

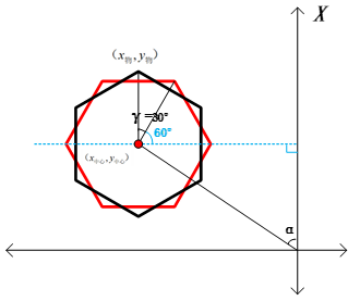

图 23 最高点 x 坐标约等于质心 x 坐标

如图 23 是最高点 x 坐标约等于质心的情况，黑色物块是假定六边形的实际姿态，红色物块则是该六边形要旋转成的目标姿态。此时线性拟合达成的反正切算法误差较大，让旋转的角度为一个固定的值。

旋转的角度为：$\gamma = 30^\circ$

d) 三角形

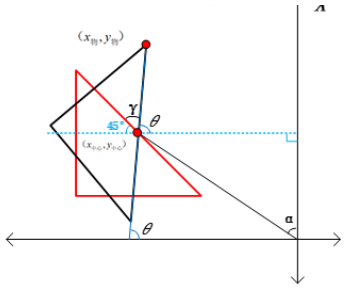

图 24 最高点 x 坐标在质心 x 坐标右边

如图 24 是最高点 x 坐标大于质心 x 坐标的情况，黑色物块是假定三角形的实际姿态，红色物块则是该三角形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = 135° - \theta$

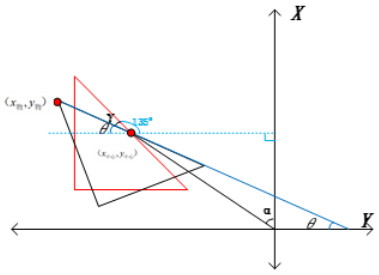

图 25 最高点 x 坐标在质心 x 坐标左边

如图 25 是最高点 x 坐标小于质心 x 坐标的情况，黑色物块是假定三角形的实际姿态，红色物块则是该三角形要旋转成的目标姿态。利用反正切求出的θ来求需要旋转的角度。

旋转的角度为：$\gamma = 45^\circ - \theta$

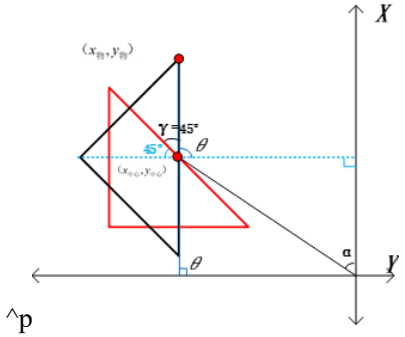

图 26 最高点 x 坐标约等于质心 x 坐标

如图 26 是最高点 x 坐标约等于质心的情况，黑色物块是假定三角形的实际姿态，红色物块则是该三角形要旋转成的目标姿态。此时线性拟合达成的反正切算法误差较大，让旋转的角度为一个固定的值。

旋转的角度为：$\gamma = 45^\circ$

3 仿真验证及分析

以六边形最高点 x 坐标在质心 x 坐标附近为例，其旋转的角度为：

原图：

图 27 传入该模块的图片

仿真波形：

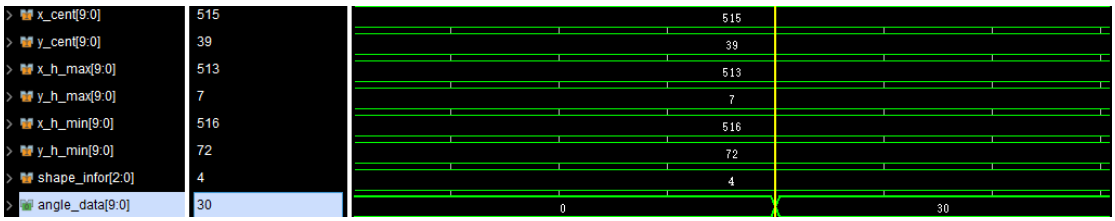

图 28 反正切仿真波形，六边形角度

可以看见，此时六边形最高点 x 坐标为 516，质心 x 坐标为 515。两者相互接近相差像素为 1 则旋转角度为 30，旋转角度正确。

2.2 基于 FPGA 的机械臂分析及控制系统设计

对于机械臂分析及控制模块，主要分析逆运动算法设计、机械臂控制模块总体设计、状态机控制模块设计、线性插值模块设计等。

2.2.1 逆运动算法设计

此系统对各关节角度求解基于二元一次三角方程组和反正切运算。并且在 FPGA 中，此模块只需输入物体的二维坐标，即可通过简单运算和 ROM 读取得到机械臂 5 个舵机的角度信息。

相比传统机械臂系统中利用旋转矩阵运算进行逆运动解算的方法，此方法大幅降低了逻辑资源和 DSP 资源的消耗，仅消耗少量存储资源即可完成逆运动解算，使逆运动算法在 EP4CE6F17C8 芯片（各资源紧缺）上的实现具有“高性价比”。

1 预想分析

以仓库右上角为原点O建立二维平面直角坐标系$X-Y$（如图 29 中蓝色坐标系所示），以机械臂中心为原点O'建立二维平面直角坐标系$X'-Y'$（如图 29 中红色坐标系所示）。根据X'/Y'的关系可得出 0 号舵机的角度，同时，由几何关系可知 4 号舵机偏转的角度与 0 号舵机的偏转角度相同。

以O'为圆心，O'到物体中心坐标为半径R（即$\sqrt{X'^2+Y'^2}$）做圆，机械臂在同一半径的圆周上运动时，1、2、3 号舵机的角度不变。故当R确定时，1、2、3 号

舵机的角度可随之确定。

由以上分析出的方法，即可分别通过X'/Y'的数值和R的数值确定所有舵机的角度。由此预想进行具体方程组的建立、求解。现已验证此预想可行。

2 数学模型建立

① 坐标系的建立

a) $X-Y$坐标系的建立

以仓库平面的右上角处为原点O，仓库的长为X轴，仓库的宽为Y轴，建立二维平面直角坐标系$X-Y$（如图 29 中蓝色坐标系所示）。

b) $X'-Y'$坐标系的建立

以机械臂中心轴与仓库平面的交点为原点O'，平行于X轴且穿过O'的直线为X'轴，平行于Y轴且穿过O'的直线为Y'轴建立二维平面直角坐标系$X'-Y'$（如图 29 中红色坐标系所示）。根据实际位置关系可得：

$$\begin{cases} X' = X + 85 \\ Y' = Y + 85 \end{cases}$$

c) $R-Z$坐标系的建立

以机械臂中心轴与仓库平面的交点为原点Oz，机械臂中心轴为Z轴，各个物体中心点与Oz的连线为R轴，分别建立n个（数量取决于仓库中物块的个数）二维平面直角坐标系R_1-Z、R_2-Z、…、R_n-Z（如图 30 为例）；

d) 舵机标号

图 29 中绿线为舵机的 0°与 270°的阈值。图 30 中蓝色数字为舵机标号。

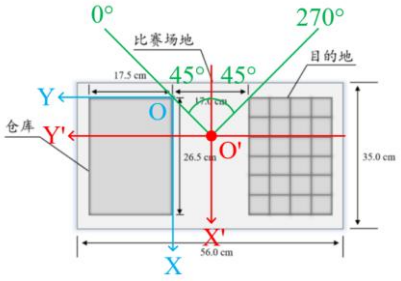

图 29 $X-Y$与$X'-Y'$坐标系图

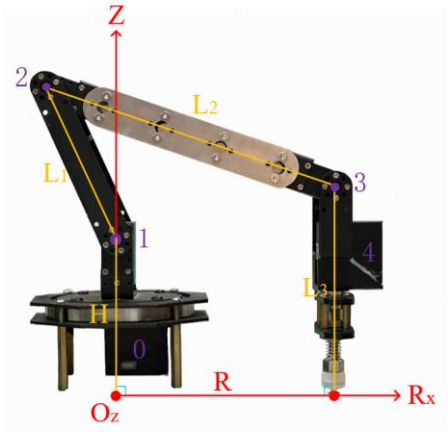

图 30 在$R_n < 175$时，舵机型号标号与$R_n - Z$坐标系图

② 0 号与 4 号舵机角度的模型

0 号舵机角度求解坐标图如图 31 所示，已知物体在$X' - Y'$坐标系下的中心坐标（$X_{物}$，$Y_{物}$），设物体中心点与Y'轴的夹角为θ，则由实际关系可知：

$$\tan\theta = X_{物}/Y_{物} \rightarrow \theta = \arctan\left(X_{物}/Y_{物}\right)$$

设 0 号舵机角度为θ_0，则由实际关系可得：

a） 当$X' \geq 0$时（即$X <= 85$时），$\theta_0 = \frac{\pi}{4} - \theta$；

b） 当$X' < 0$时（即$X > 85$时），$\theta_0 = \frac{\pi}{4} + \theta$。

测量并计算出物体中心点与Y'轴的最大夹角为$\theta_{\max} = 64^\circ$，故$\theta \in [0，64]^\circ$，得出$\theta_0 \in [0，107]^\circ$。

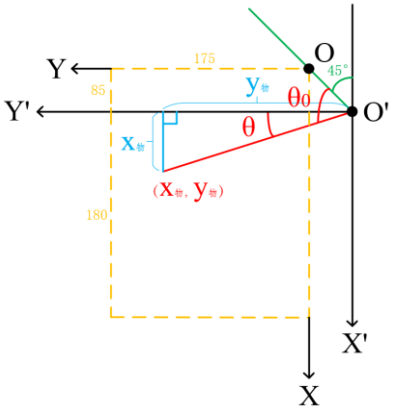

图 31 0 号舵机角度求解坐标图

设 4 号舵机偏置角度为θ_{4p}，由图 32 可得出几何关系：

$$\theta_{4p} = \theta$$

设 4 号舵机角度为θ_4，则由实际关系可得：

a） 当$X' \geq 0$时（即$X <= 85$时），$\theta_4 = \frac{3\pi}{4} - \theta_{4p} = \frac{3\pi}{4} - \theta$；

b） 当$X' < 0$时（即$X > 85$时），$\theta_4 = \frac{3\pi}{4} + \theta_{4p} = \frac{3\pi}{4} + \theta$。

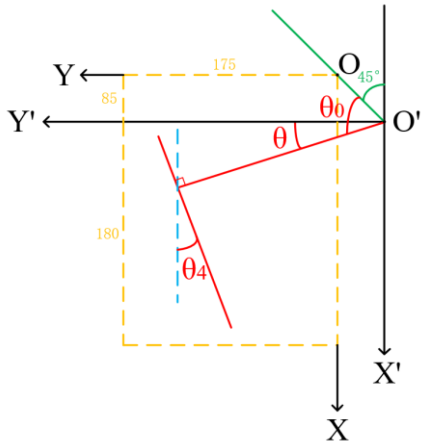

图 32 4 号舵机角度求解坐标图

③ 1 号、2 号、3 号舵机角度的模型

变量声明与参数测量：

设 1 号舵机的旋转轴距离地面的高度为H；1 号舵机与 2 号舵机旋转轴的距离为L_1；2 号舵机与 3 号舵机旋转轴的距离为L_2；3 号舵机与气泵末端（机械臂末端执行器）的距离为L_3；分别设 1、2、3 号舵机的角度为θ_1、θ_2、θ_3，设L_1与H延长线的夹角为α；L_2与L_1延长线的夹角为β；L_3与L_2延长线的夹角为γ。

实际测量出$H = 100mm$、$L_1 = 105mm$、$L_2 = 188.5mm$、$L_3 = 134.5mm$；计算出最小半径$R_{\min} = 85mm$、最大半径$R_{max} = 316mm$，故$R \in [85，316]mm$。

已知物体在$X' - Y'$坐标系下的中心坐标为（$X_{物}$，$Y_{物}$）,设物体与O'的距离为R，则

$$R = \sqrt{X_{物}{}^2 + Y_{物}{}^2}$$

根据R的数值分类讨论：

a) $R < 175$

此时机械臂的简化坐标图如图 33 所示，为使机械臂能够完成准确抓取，令气泵末端（机械臂末端执行器）与仓库表面垂直，则由各约束条件建立模型的几何关系得方程组：

$$\begin{cases} R = -L_1 \sin\alpha + L_2 \sin\gamma \\ L_3 - H = L_1 \cos\alpha - L_2 \cos\gamma \\ -\alpha + \beta + \gamma = \pi \\ \alpha, \beta, \gamma > 0 \end{cases}$$

其中α、β、γ为未知数，其余皆为已知数，三个方程三个未知数即可求出α、β、γ的值。

由实际几何关系可得：

$$\begin{cases} \theta_1 = \dfrac{3\pi}{4} + \alpha \\ \theta_2 = \dfrac{3\pi}{4} + \beta \\ \theta_3 = \dfrac{3\pi}{4} - \gamma \end{cases}$$

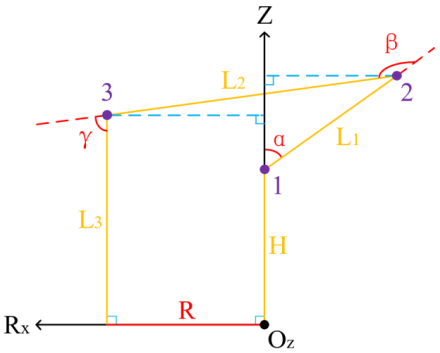

图 33 在$R < 175$时的$R - Z$坐标图

b) $R = 175$

此时机械臂的简化坐标图如图 34 所示，L_1与H两个连杆刚好组成一条直线，即$\alpha = 0$，不满足$\alpha > 0$的约束，故当$R \geq 175$时，此模型的方程组应当改变。$R = 175$即为两个方程组的临界值。

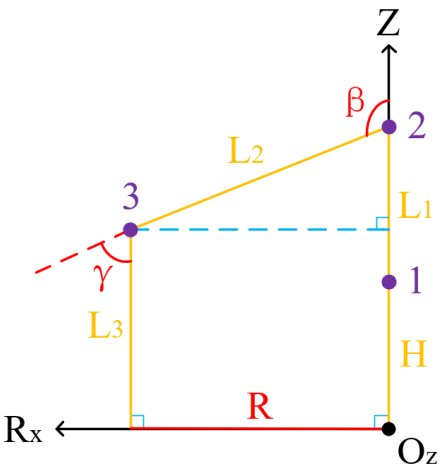

图 34 在$R=175$时的$R-Z$坐标图

c) $175<R<291$

此时机械臂的简化坐标图如图 35 所示，为使机械臂能够完成准确抓取，令气泵末端（机械臂末端执行器）与仓库表面垂直，则由各约束条件建立模型的几何关系得方程组：

$$\begin{cases} R=L_1\sin\alpha+L_2\sin\gamma \\ L_3-H=L_1\cos\alpha-L_2\cos\gamma \\ \alpha+\beta+\gamma=\pi \\ \alpha,\beta,\gamma>0 \end{cases}$$

其中α、β、γ为未知数，其余皆为已知数，三个方程三个未知数即可求出α、β、γ的值。

由实际几何关系可得：

$$\begin{cases} \theta_1=\dfrac{3\pi}{4}-\alpha \\ \theta_2=\dfrac{3\pi}{4}+\beta \\ \theta_3=\dfrac{3\pi}{4}-\gamma \end{cases}$$

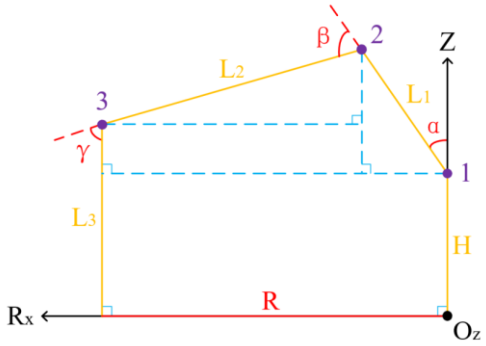

图 35 在$175 < R < 291$时的$R - Z$坐标图

d) $R = 291$

此时机械臂的简化坐标图如图 36 所示，L_2与L_3两个连杆刚好组成一条直线，即$\beta = 0$，不满足$\beta > 0$的约束，故当$R \geq 291$时，此模型的方程组应当改变。$R = 291$即为两个方程组的临界值。

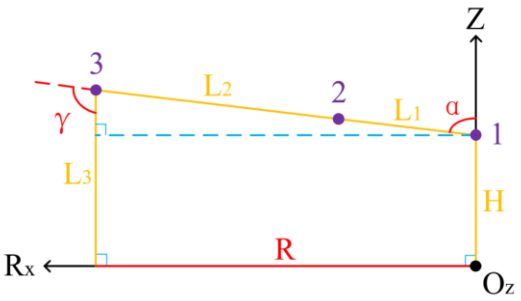

图 36 在$R = 291$时的$R - Z$坐标图

e) $291 < R < 316$

此时机械臂的简化坐标图如图 37 所示，由于此时 R 过大，气泵末端无法与仓库表面垂直，但 2 号舵机将不受 R 变化而变化，而是始终保持 135°，则由各约束条件建立模型的几何关系得方程组：

$$\begin{cases} R = L_1 \sin\alpha + L_2 \sin\gamma \\ L_3 - H = L_1 \cos\alpha - L_2 \cos\gamma \\ \alpha, \beta, \gamma > 0 \end{cases}$$

其中α、γ为未知数，其余皆为已知数，两个方程两个未知数即可求出α、γ的值。

由实际几何关系可得：

$$\begin{cases} \theta_1 = \dfrac{3\pi}{4} - \alpha \\ \theta_2 = \dfrac{3\pi}{4} \\ \theta_3 = \dfrac{3\pi}{4} - \gamma \end{cases}$$

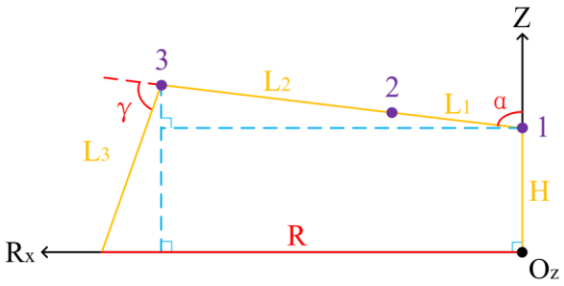

图 37 在$R > 291$时的$R - Z$坐标图

3 求解并将解存入 ROM

利用 MATLAB 对 0 号舵机角度和 1、2、3 号舵机角度的模型分别进行建模。

对于 0 号舵机的角度：通过理论分析得出，$\theta_0 \in [0，106]^\circ$，结合存储资源消耗和模型精度考虑，选取 1°为间隔（共 107 个），存入“.mif”文件（catch_servo0_rom_data.mif）中供 Quartus 的 ROM（IP 核）使用。

对于 1、2、3 号舵机的角度：通过理论分析得出$R \in [85，316]mm$，结合存储资源消耗和模型精度考虑，选取 1mm 为间隔（共 231 个），分别代入方程组后利用 solve 函数对未知数为α、β、γ的方程组求解。经求解，得每个R对应两组α、β、γ。选择其中舵机所需力矩较小（即类似图 30 所示姿态）的一组解作为最终结果。将α、β、γ由几何关系变换，最终得到 1、2、3 号舵机的角度（共 231 组），存入“.mif”文件（catch_servo123_rom_data.mif）中供 Quartus 的 ROM（IP 核）使用。

4 FPGA 实现

由预想分析中提及，以X'/Y'和R的数值分别确定 0 号与 4 号舵机和 1、2、3 号舵机的角度，即确定所有舵机的角度。现在已完成机械臂的数学模型搭建，通过 MATLAB 求解出所有不同 R 时各舵机的角度，并将角度信息存入 ROM 中。由此，需要 FPGA 实现的功能如下：

a) 将图像识别模块输出的在$X-Y$坐标系中的坐标转换到$X'-Y'$坐标系中，并判断物体所在仓库区域位于上/下半区；

b) 经过计算得到X'/Y'和R的值；

c) 将X'/Y'和R的值分别转换为 ROM 的地址；

d) 通过地址，将角度信息从 ROM 中读出；

对于“c)将X'/Y'的值转化为 ROM 的地址”，需进行反正切函数运算，若利用传统的 Cordic（IP 核）在 FPGA 中实现，则消耗资源大。而此次运用反正切函数的分段线性拟合函数在 FPGA 中实现反正切运算，可极大优化 FPGA 的资源利用。

据实验观察，可将定义域为[0,2.06]的反正切函数分为 8 段以达到理想误差。同时，由于使用移位相加的方式在 FPGA 中实现除法，并且在不使用浮点数运算 IP 核的情况下，FPGA 中只能进行整数加法运算，则需加入两个约束条件：一次函数中斜率系数的小数部分通过最高八位二进制表示；一次函数中截距为整数。可见拟合效果优秀。利用 MATLAB 进行此数学模型的建立，并求解出分段线性函数的最优顶点、各分段一次函数的斜率和截距。最终得出的分段线性拟合函数与原反正切函数的对比图如图 38 所示，可见拟合效果优秀。

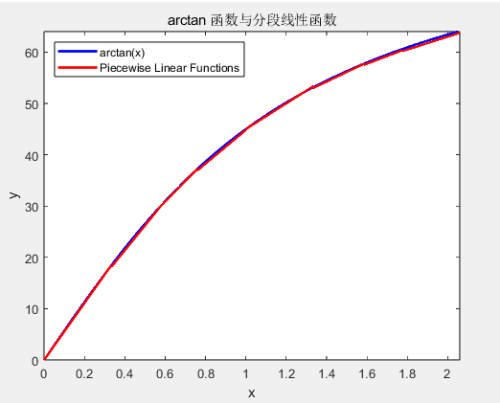

图 38 分段线性拟合函数与反正切函数

仿真：

若使用组合逻辑赋值，在信号的传输上产生的延时会有几率会造成时序紊乱，故利用对坐标有效标志信号（valid_axis_data）进行延时，分别延时一个、两个、

三个时钟周期，产生 valid_axis_data_reg 、 valid_axis_data_reg_1 、valid_axis_data_reg_2 信号作为触发信号，利用时序逻辑进行赋值。

其中 31~39 行即为利用移位相加的方式实现对反正切函数的分段线性拟合的具体实现代码(always 赋值语句)。

按照以上功能编写 Verilog HDL 代码并对其进行 Modelsim 仿真，由图 39 所示，上述功能已实现。

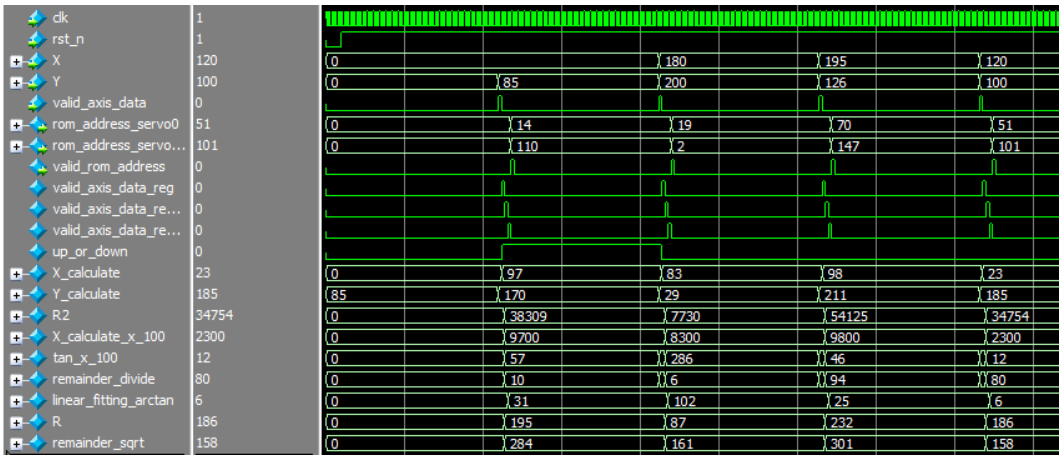

图 39 逆运动算法模块的时序仿真图

2.2.2 机械臂控制模块总体设计

1 设计内容

① 状态机控制模块：控制机械臂的状态；

② 线性插值模块：控制机械臂的速度；

③ 模式选择模块：控制机械臂的模式，分为全速模式和线性插值模式。

2 机械臂控制系统流程

状态机模块接收前面模块发送的抓姿态信息和放姿态信息，并将这些信息在对应的时间分别输送给模式选择模块，根据状态机模块输出的使能信号，模式选择模块分别以全速模式（不做处理）、线性插值模式（进行线性插值处理）进行，输出期望的 PWM 信息。如图 40 所示。

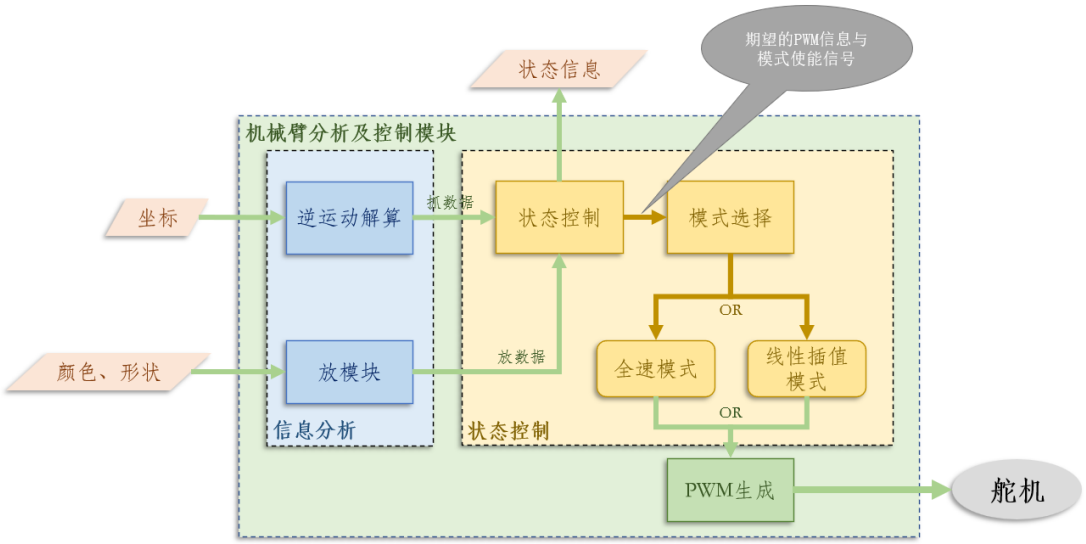

图 40 机械臂控制系统流程图

3 问题发现

① 若需要舵机发生较大角度改变时，直接输出期望角度的 PWM，舵机会以最快速度运行，受惯性影响，舵机往往会超出预想位置一段距离，进而造成把物体撞散的结果。

② 若不分时间顺序，同时对四个舵机分别输入 PWM，机械臂末端执行器距离仓库和目的地平面的高度往往会低于薄片的高度，而造成把物体撞散的结果。

③ 若设置机械臂复位姿态为竖直姿态，即四个舵机皆处于中间位置，则从复位姿态到抓取姿态的过程当中会碰撞到开发板。

4 解决方法

① 设计线性插值模块：每隔一段时间改变一次步进值，直到达到期望值，最大化减少惯性对系统造成的消极影响。

② 令 1 号舵机最先或最后动：利用线性插值模块的完成线性插值信号和状态机控制模块里的延迟计数器，达到使 1 号舵机从复位姿态到抓取姿态或放置姿态的过程中最后运动、从抓取姿态或放置姿态到复位姿态的过程中最先运动的目的，从而完全消除机械臂末端执行器撞散物体的可能性。

③ 改变复位姿态：在状态机控制模块里设置新的复位值，使复位姿态为“弯曲”

姿态。

5 FPGA 实现与仿真

选取 1500、2000、2100、800、1500 分别为 0、1、2、3、4 号舵机的复位值。

以随机物体为例编写仿真文件，如图 41 和图 42 所示，可见系统以实现上述功能。

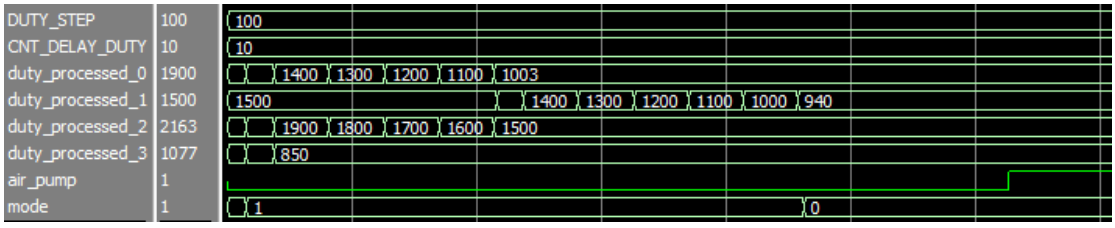

图 41 某个物体的从复位姿态到抓姿态机械臂控制系统输出的 PWM 信息

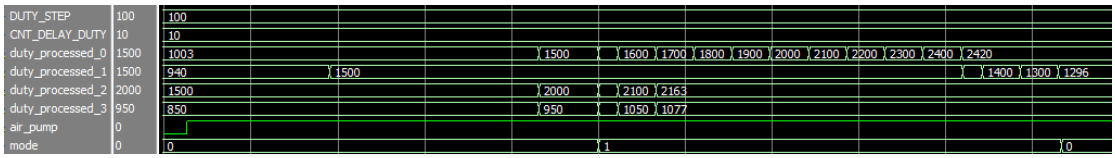

图 42 某个物体的从抓姿态到复位姿态机械臂控制系统输出的 PWM 信息

2.2.3 状态机控制机械臂模块设计

此模块实现了机械臂的状态控制，并且**每个步骤的时间可调**。

1 预想分析

若想要机械臂达到预想的行动流程，舵机需要严格地按照步骤在某一段时间内到达某一个角度，故需设计一个模块来控制机械臂的状态。并且，每个舵机只有一个控制端口，故需要将抓取姿态信息、放置姿态信息分别发送到状态机模块，并通过此模块实现在指定的时间内向下一个模块输出预想的 PWM 信息。

同时，由于 PWM 波输入给舵机以后，舵机需要一定时间达到指定位置，但 FPGA 的时钟远快于舵机所需运动时间，故需要在 FPGA 中添加延时，等待舵机到达指定位置后再做下一步状态的转换，以免发生机械臂运动紊乱的情况。

2 模块功能

① 按流程分别向抓模块和放模块发送使能，并接收抓数据和放数据；

② 按 1 复位—>2 抓姿态—>3 气泵吸—>4 复位—>5 放姿态—>6 气泵放—>7 复位的顺序，将每个状态所对应的 PWM 信息按流程传输给下一个模块；

③ 当状态为 1—>2 和 4—>5 时，输出线性插值使能信号；

④ 当状态为 3 和 6 时，分别控制气泵吸和放；

⑤ 接收开始信号，开始一个物体的移动，在“4 复位”状态时输出图像处理使能信号，并且再完成所有步骤后输出完成信号。

3 FPGA 实现与仿真

利用状态机与计数器结合，当计数器计满为触发信号，使状态转换。分别在各个状态下完成控制其他模块的使能、失能、完成信号的发送与接收。

按照以上功能编写 Verilog HDL 代码并对其进行 Modelsim 仿真，由图 43 所示，上述功能已实现。

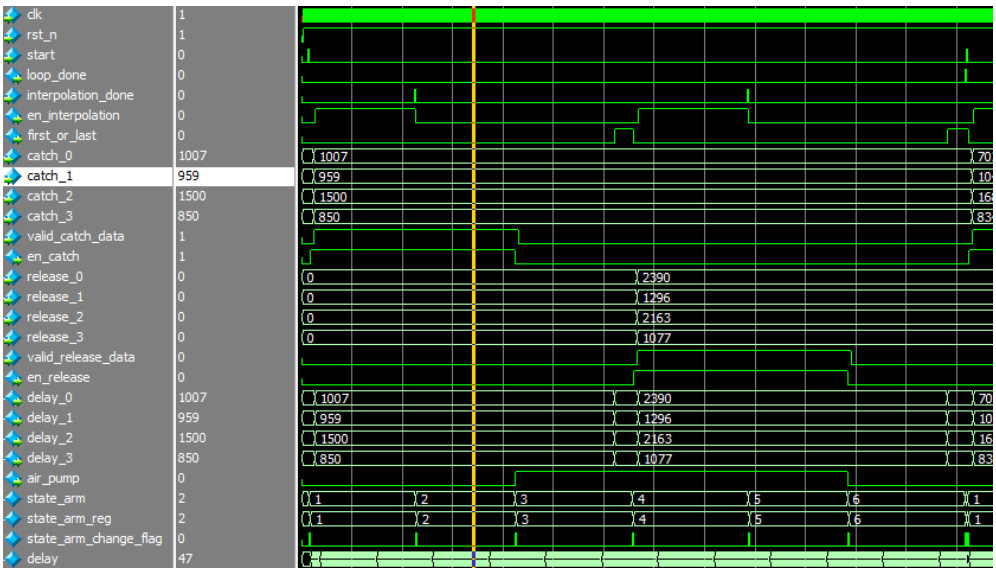

图 43 状态控制模块的时序仿真图

2.2.4 线性插值模块设计

此模块**实现了机械臂运动速度的可控**。

1 预想分析

由于使用 PWM 控制舵机，舵机会以最大速度到达指定位置，若改变 PWM

值使舵机转动较大角度时，受惯性影响，机械臂会超出指定位置的一定范围，这有可能会造成把物体撞散的结果，故若想要达到使机械臂稳定运动、尽量消除惯性对系统造成的危害、机械臂速度可变的目标，需设计一个可以控制舵机运动速度的模块，即线性插值模块。

2 模块功能

① 将输入的初始值和目的值对比；

② 若初始值大于目的值，则每隔一段时间减一个固定的值，直到达到目的值；若初始值小于目的值，则每隔一段时间加一个固定的值，直到达到目的值；

③ 向 PWM 生成模块输出实时的值；

④ 当初始值达到目的值时，输出一个线性插值完成信号；

⑤ 间隔时间可变、从初始值到目的值的步进值可变。

3 FPGA 实现与仿真

① 设置输入端口为线性插值使能信号 en_interpolation、初始值 duty_start、目标值 duty_target、步进值 duty_step、分频计数器最大值 CNT_DELAY_DUTY，输出端口为实时值 duty、线性插值完成 duty_done 信号；

② 为实现间隔时间可变的目标，需设计一个可变进制计数器 cnt 做模块主时钟 clk 的分频器，以生成另外一个时钟 clk_div，作产生步进值的触发条件；

③ 当线性插值使能信号 en_interpolation 拉高后，实时值 duty 每隔一段时间增加或减少一次步进值 duty_step，当实时值等于目标值时，将线性插值完成信号 duty_done 拉高。

时序仿真图如图 44 所示，上述功能已实现。

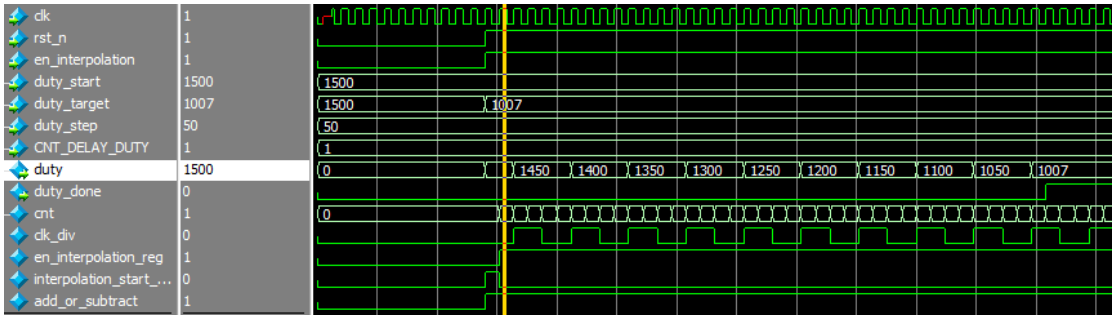

图 44 线性插值模块的时序仿真图

2.3 硬件电路设计

此作品为实现 AWC_C4 开发板与升腾 mini 开发板信号的通信，设计了 AWC_C4 & 升腾连接板。同时，为实现理想的控制目的，完成优秀的控制效果，并提高系统稳定性，设计了与连接板配套的 PWM 舵机驱动板。

2.3.1 AWC_C4 & 升腾连接板

AWC_C4 & 升腾连接板的主要功能为：将 AWC_C4 开发板上的 IO 端口引出，用以控制舵机、气泵等外设，同时将剩余所有端口与升腾开发板引出的端口相互连接，实现信号通讯。

电路板上搭载了按键、PWM 信号传输端口、气泵端口、开发板输入端口等硬件。如图 45 和图 46 所示。

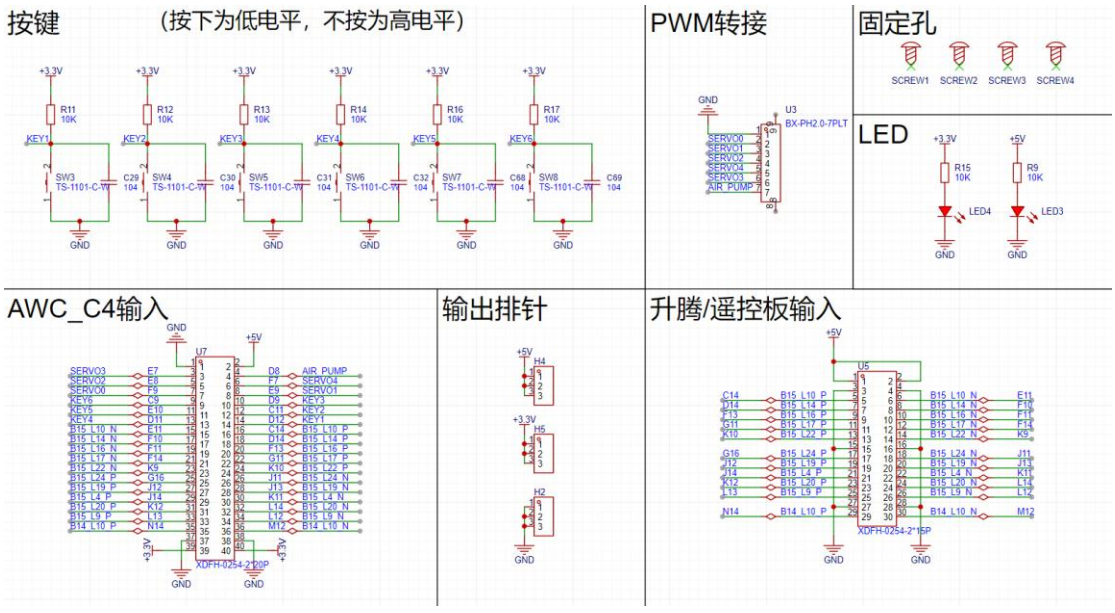

图 45 AWC_C4 & 升腾连接板电路原理图

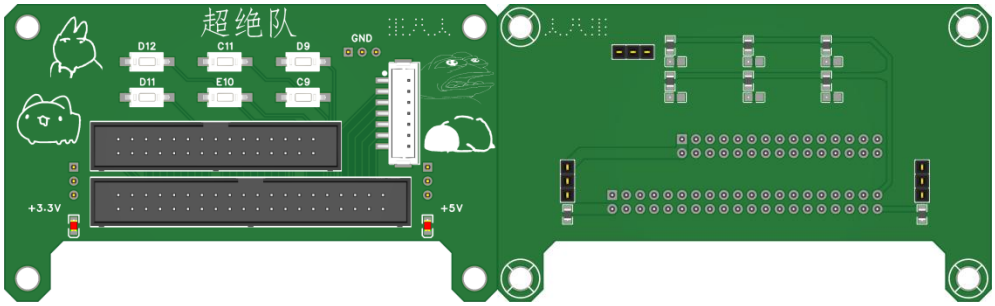

图 46 AWC_C4 & 升腾连接板 PCB 的正/反面平面图

2.3.2 PWM 舵机驱动板

PWM 转接板的主要功能为：接收 AWC_C4 & 升腾连接板上传输过来的 PWM 信号和气泵的使能信号以分别控制舵机角度和气泵的吸或放，并分别给 5 个舵机和气泵供电。

板上搭载了电源输入端口、LED、气泵端口、PWM 端口（5264）等。原理图如图 47，PCB 图如图 48。

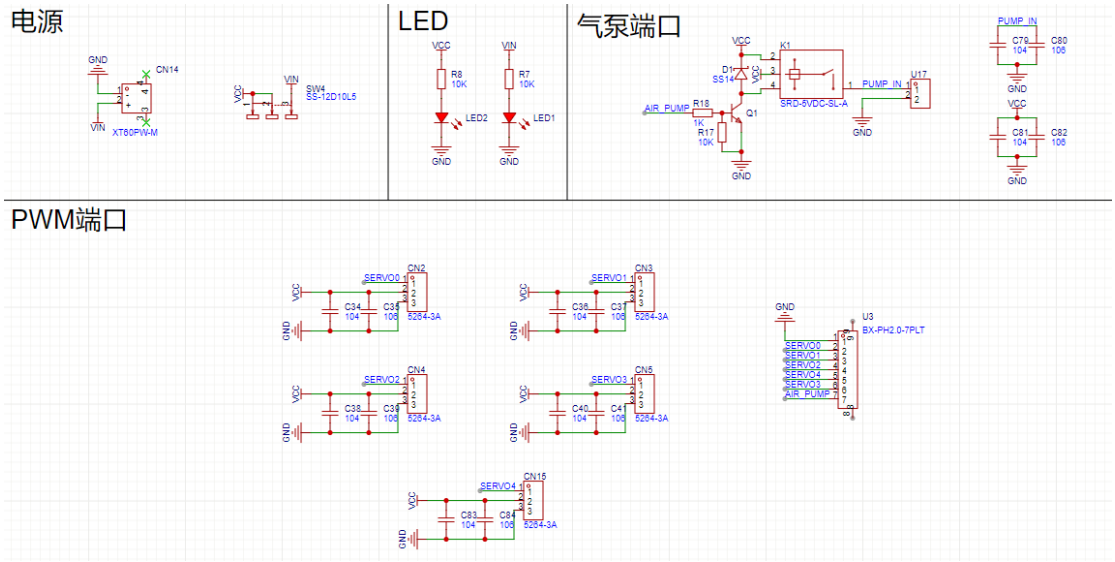

图 47 PWM 舵机驱动板的电路原理图

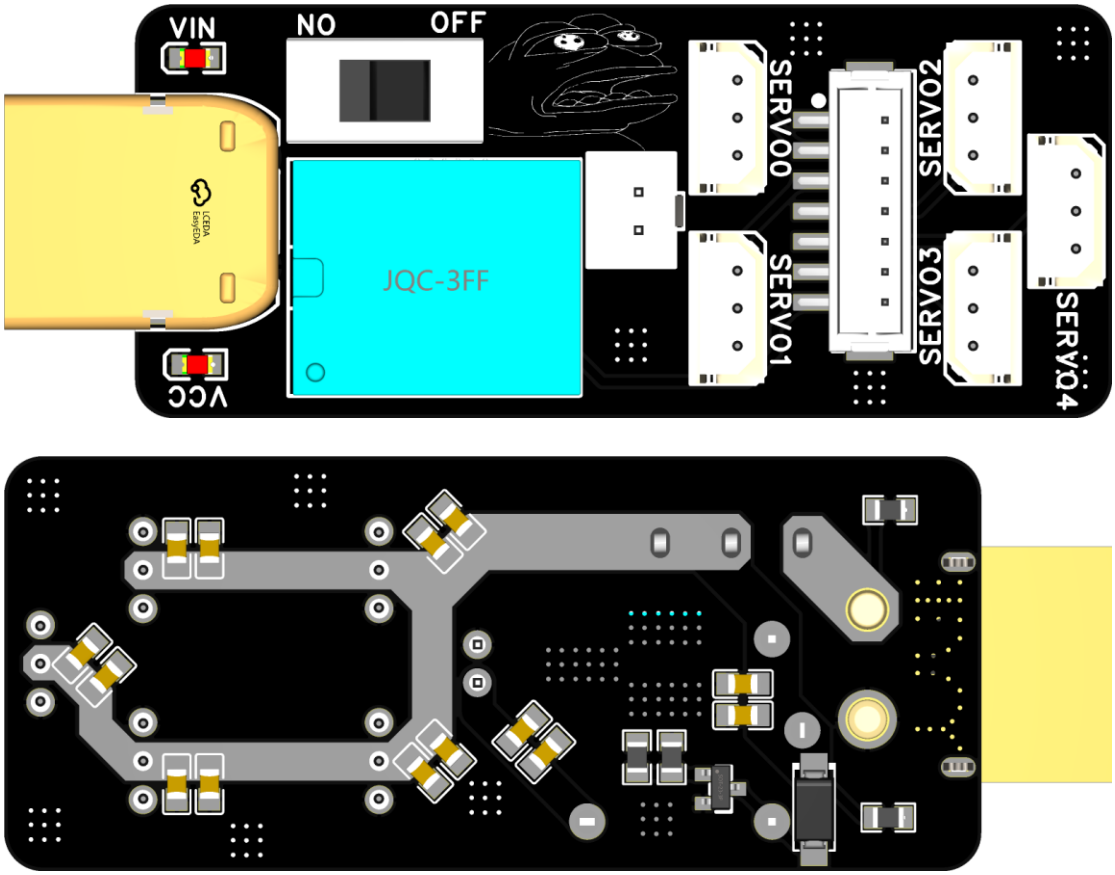

图 48 PWM 舵机驱动板的 PCB 正/反面图

2.4 机械臂结构设计

本作品机械臂结构主要由众灵科技机械臂套件（如图 49 所示）搭建而成，但原套件不足以完成目标功能，故对某一连杆进行改进，最终效果如图 50 所示。

图 49 众灵科技机械臂套件

图 50 本作品的机械臂结构图

其中改进的连杆是使用 SOLIDWORKS 二维建模而成，其二维图纸如图 51 所示。

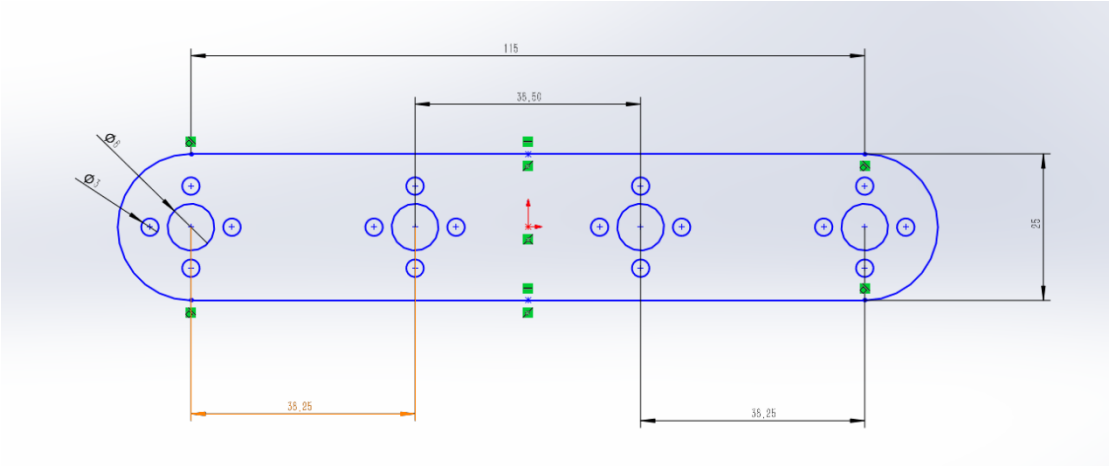

图 51 改进连杆的二维图纸

3 性能分析

本装置能够优秀完成第九届集创赛海云捷讯杯难度 2 的要求，即区分物体颜色、形状；随即乱序起始位置；按照颜色、形状进行目的地排列。且锁定时间和移动速度较快。

3.1 目标锁定时间

目标锁定时间由图像采集及识别模块确定，由于时间极短，无法使用秒表测量时间，故此处仅提供理论值。

由于系统使用摄像头 OV5640，根据其数据手册及实际应用情况可知，此系统所配置的摄像头帧率为 15fps，由此求得采集到每一帧图像信息所需时间

$$T_{\text{图像采集}} = 0.067s$$

由图像识别模块具体设计方案，利用 1 帧图像信息即可确定某一颜色中的处于图像坐标系最左上角的物体的中心坐标，并且同时将坐标信息存入寄存器中。设目的地有N个物块，锁定一种颜色中一个物体的中心坐标的时间为T_{single}，锁定所有物体中心坐标的时间为T_{all}，则通过计算可得

$$T_{\text{single}} = 1 \times 0.067s = 0.067s$$

$$T_{\text{all}} = N \times 0.067s$$

由于 FPGA 中 assign 语句综合后的连线会导致极其短暂时间的延迟，并且时序逻辑赋值会产生数个时钟周期的延时，故实际锁定时间会稍大于理论值，但由于图像处理模块的系统时钟一个周期为 ns 级，故忽略不计。

3.2 移动速度

本装置的移动速度可调，其中最快速度$V_{\max}$由舵机最快速度所限制，最慢速度$V_{\min}$理论上可以为无限接近于零。

由于难以测得机械臂末端执行器运动的距离，并且此值代表的实际意义不大，于是此实验以测试系统完成以难度 2 为标准完成N个物体运送所需的时间为参数，以评估系统的移动速度。

由于系统在最快速度下的工作精度相对较低，在此定义系统工作稳定的条件为系统将物体从仓库运送至目的地后物体中心点与期望中心点距离的标准差小于等于 2mm（$\sigma \leq 2mm$）。

设系统工作稳定状态下完成任务所需时间为T_{smin}；设最快速度$V_{\max}$所对应的最短时间为$T_{\min}$；最慢速度$V_{\min}$所对应的时间为$T_{\max}$。

经过十次实验测试（实验具体数据如表 2 所示），测试结果数据经计算处理后得到结果如表 4。

表 2 系统完成难度 2 任务的总时间情况

	T_{smin}	$T_{\min}$
时间（s）	$6.94N$	$5.27N$

3.3 定位精度

由于此系统的速度可调，但在不同速度下系统的定位精度不同，故选取三个典型的速度分别进行一次测试。测试结果如表 3 所示。

此系统不能控制物体的具体姿态，所以当物体为正方形时，由于正方形面积最大，若姿态稍偏就会与虚线相交，难以不触碰虚线，但圆形和六边形能够在90.33%的情况下处于虚线框内，并且不触及虚线框。

实验具体测试数据如附录 1 中表 5、表 6、表 7 所示，将三组实验测试数据经计算处理出均方根误差、方差和均值误差，如表 3 所示。

表 3 三个典型速度下的定位精度

总时间（s）	均方根误差（cm）	方差（cm^2）	均值误差（cm）
T_{smin}	1.45	2.27	1.30

$T_{\min}$	2.11	4.77	1.92
$6.94N$	0.35	0.13	0.27

3.4 总结

综上所述，本装置仅利用一块 AWC_C4 开发板就可以优秀的实现题目中目标三难度的要求，且在附加分部分设计了上位机模块和摇杆模块。

其中图像采集及识别模块所用到的距离变换解决了物块粘连的问题，连通域算法使系统能够区分出各个物块各自的区域，且坐标验证算法利用查面积的方法高效地验证坐标是否可供机械臂进行抓取。

其中机械臂分析及控制模块总共所用 Logic Cells 仅 1482（相对极少），代表其所用到的逆运动解算算法、反正切算法、状态控制算法利用了极少的逻辑资源、DSP 资源和存储资源达到了实际目的，并且精度可观。在逆运动结算中所涉及的反正切运算利用分段函数拟合的方式，实现了资源的高效利用。

4 附录

表 4 完成难度 2 任务的总时间测试数据

	T_{smin}	T_{min}	T_{max}
时间（s）	$6.932N$	$5.28N$	∞
	$6.954N$	$5.25N$	∞
	$6.942N$	$5.27N$	∞
	$6.948N$	$5.27N$	∞
	$6.926N$	$5.27N$	∞
	$6.929N$	$5.23N$	∞
	$6.937N$	$5.26N$	∞
	$6.943N$	$5.27N$	∞
	$6.938N$	$5.31N$	∞
	$6.951N$	$5.29N$	∞
平均时间（s）	$6.94N$	$5.27N$	∞

表 5 T_{smin}速度下的定位精度测试数据

	水平距离	垂直距离	直线距离
	0.70	1.42	1.58

误差（cm）	-0.30	2.41	-2.43
	-0.50	-1.44	1.52
	-0.20	-1.58	1.59
	-0.40	0.77	-0.87
	0.70	0.67	0.97
	0.80	0.70	1.06
	0.50	-0.66	-0.83
	-1.20	1.10	-1.63
	1.00	-2.11	-2.33
	-0.20	0.37	-0.42
	-0.40	0.00	0.40
均方根误差（cm）			1.45
方差（cm^2）			2.27
平均误差（cm）			1.30

表 6 $T_{\min}$速度下的定位精度测试数据

	水平距离	垂直距离	直线距离
误差（cm）	1.10	2.11	2.38

	-1.90	2.84	-3.42
	-0.90	-1.55	1.79
	-1.10	2.13	-2.40
	0.90	1.10	1.42
	-0.80	0.90	-1.20
	1.10	1.60	1.94
	0.97	-1.21	-1.55
	-0.70	0.89	-1.13
	3.10	2.10	3.74
	1.30	0.47	1.38
	0.40	0.60	0.72
均方根误差（cm）			2.11
方差（cm^2）			4.77
平均误差（cm）			1.92

表 7 $T = 12.96Ns$ 速度下的定位精度测试数据

	水平距离	垂直距离	直线距离
误差（cm）	0.05	0.20	0.21

	0.10	0.15	0.18
	-0.05	-0.20	0.21
	0.20	0.33	0.39
	0.00	0.01	0.01
	0.10	0.23	0.25
	0.40	-0.49	-0.63
	0.00	-0.13	-0.13
	0.15	-0.22	-0.27
	0.12	0.22	0.25
	-0.60	0.47	-0.76
	-0.10	0.00	-0.01
均方根误差（cm）			0.35
方差（cm^2）			0.13
平均误差（cm）			0.27