

学术型硕士(打印时删除)



同濟大學
TONGJI UNIVERSITY

硕士学位论文

基于 RGB-D 图像的三维物体识别算法
的研究与实现

姓名：李勇奇

学号：1531620

所在院系：电子与信息工程学院

学科门类：工学

学科专业：控制科学与工程

指导教师：陈启军 教授

二〇一八年三月



同濟大學
TONGJI UNIVERSITY

A dissertation submitted to
Tongji University in conformity with the requirements for
the degree of Master of Engineering

3D Object Recognition and Pose Estimation Based on RGB-D Images

Candidate : Li Yongqi
Student Number : 1531620
School/Department : College of Electronics and
Information Engineering
Discipline : Engineering
Major : Control Science and Engi-
neering
Supervisor : Prof. Chen Qijun

March, 2018

学位论文版权使用授权书

本人完全了解同济大学关于收集、保存、使用学位论文的规定，同意如下各项内容：按照学校要求提交学位论文的印刷本和电子版本；学校有权保存学位论文的印刷本和电子版，并采用影印、缩印、扫描、数字化或其它手段保存论文；学校有权提供目录检索以及提供本学位论文全文或者部分的阅览服务；学校有权按有关规定向国家有关部门或者机构送交论文的复印件和电子版；在不以赢利为目的的前提下，学校可以适当复制论文的部分或全部内容用于学术活动。

学位论文作者签名：

年 月 日

同济大学学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：

年 月 日

摘要

@todo: 摘要

关键词: 三维视觉, 目标检测, 位姿估计, 随机分拣

ABSTRACT

@todo: abstract

D vision, object detection, pose estimation, Bin-Picking

目录

第 1 章 引言	1
第 2 章 RGB-D 图像的获取与融合	2
2.1 3D 相机现状与分析	2
2.2 RGB-D 相机	3
2.2.1 RGB-D 相机原理与结构	3
2.2.2 RGB-D 相机的数学模型	4
2.2.3 RGB-D 相机的标定流程	6
2.3 对偶 RGB-D 相机	9
2.3.1 对偶 RGB-D 相机原理与结构	9
2.3.2 对偶 RGB-D 相机的标定流程	14
2.4 深度图质量测试实验	17
2.4.1 实验流程	17
2.4.2 实验原理	18
2.4.3 实验结果	20
2.5 本章小结	20
第 3 章 基于 RGB-D 图像的目标检测算法	21
3.1 3D Faster R-CNN	21
3.1.1 Faster R-CNN	23
3.1.2 HHA	23
3.1.3 Spatial Transformer	25
3.2 3D Mask R-CNN	28
3.2.1 特征提取网络	29
3.2.2 ROIAlign	30
3.2.3 Mask 损失函数	30
3.3 目标检测实验	31
3.3.1 数据集	32
3.3.2 实验内容	35
3.3.3 实验结果	36
3.4 本章小结	37
第 4 章 基于 4PCS 的点云匹配算法	38
4.1 点云匹配算法概述	38
4.1.1 问题描述	38
4.1.2 背景介绍	39

4.2 A4PCS-ICP 算法	40
4.2.1 算法框架.....	40
4.2.2 Angle-fixed 4PCS 算法.....	40
4.2.3 Outlier filter	44
4.2.4 ICP 算法.....	45
4.3 点云匹配实验.....	47
4.4 本章小结.....	47
第 5 章 3D 目标位姿估计算法	48
5.1 3D-MRAI 框架设计.....	48
5.2 3D-MRAI 具体实现.....	49
5.3 3D 目标位姿估计实验	51
5.3.1 数据集	51
5.3.2 实验内容.....	52
5.3.3 实验结果.....	52
5.4 本章小结.....	54
第 6 章 算法应用——Bin-Picking.....	55
6.1 Bin-Picking 背景与现状.....	55
6.2 基于 3D-MRAI 的随机分拣系统.....	57
6.2.1 系统硬件设计	57
6.2.2 系统软件设计	59
6.3 随机分拣实验	64
6.3.1 实验内容	64
6.3.2 实验结果	66
6.4 本章小结.....	67
第 7 章 结论与展望	68
7.1 结论	68
7.2 进一步工作的方向	68
致谢	69
参考文献	70
附录 A 补充资料	73
个人简历、在学期间发表的学术论文与研究成果	74

符号说明

GNU	GNU's Not Unix /'gnu:/
GFDL	GNU Free Documentation License
GPL	GNU General Public License
FSF	Free Software Foundation

第1章 引言

@todo: 引言

第 2 章 RGB-D 图像的获取与融合

RGB-D 图像是所设计的三维物体识别算法的输入, 其质量对算法结果有着至关重要的影响, 以此获取高质量的 RGB-D 图像也十分重要。本章首先分析了 3D 相机的现状, 然后详细介绍了选取的 RGB-D 相机的原理和标定流程, 并且针对其缺点提出了对偶 RGB-D 相机结构, 最后通过实验证明了对偶 RGB-D 相机可以获取更高质量的 RGB-D 图像。

2.1 3D 相机现状与分析

3D 相机能够获取相机到物体表面每一点的距离, 从而感知物体的形状和距离, 近几年 3D 成像技术的应用越来越多, 如用于体感游戏的 Kinect(Microsoft 2012), Google 的 Project Tango(Google 2012), 以及 Apple 公司的 iPhone X 前置摄像头的人脸识别。

目前市面上的 3D 相机要么价格昂贵, 要么精度低下, 很难找到一款性价比较高的 3D 相机。如表2.1和图2.1所示, 列举了一些市面上较为常见的 3D 相机,

品牌	价格	精度	速度
SICK	大于 30 万	高	很慢
Enshape	大于 30 万	高	较快
Ensenso	约 10 万	较高	较快
Realsense	约 1.5 千	中等	快
Kinect V2	约 1.5 千	低	快

表 2.1 市面上主要 3D 相机价格和性能

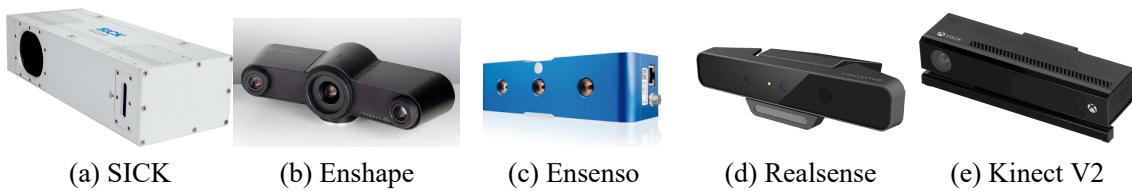


图 2.1 市面上主要 3D 相机

从表中可以发现很多精度高的 3D 相机价格及其昂贵, 并且采集速度很慢, 价格低的 3D 相机精度有相对较低。出于实际应用考虑, 我们需要相机的采集速度相

对较快,再加上成本上的限制,最终选择了精度中等、采集速度较快、价格较为便宜的 Realsense 系列相机。

2.2 RGB-D 相机

2.2.1 RGB-D 相机原理与结构

RGB-D 相机获取深度的原理大致可以分为三种:

- Structure Light
- Time of Flight(ToF)
- Stereo

Structure Light 获取深度信息的原理是通过激光发射器投射带有特定编码的结构光到物体表面后,由 IR Camera 采集,根据采集到的光信号量的变化来计算物体的深度。举一个形象的例子,将手电筒照向墙面,手电筒离墙面越远,墙面上所形成的光斑的直径就越大,所以可以通过光斑的直径来计算手电筒距离墙面的距离。ToF 获取深度信息的原理是通过专有的传感器捕捉红外光发射到接收的飞行时间来计算物体的深度。Stereo 是通过双摄像头拍摄物体,再通过特征点匹配,根据三角测量原理来计算物体的深度。

三种原理的深度相机各有其特点,采用 Structure Light 原理的深度相机一般精度比较高,但景深比较短并且受光线影响比较大,适合室内场景;ToF 原理的深度相机获取深度图的精度和分辨率一般都比较低,但帧率高,并且具有一定的抗光照性能;Stereo 获取深度精度适中,帧率相对来说较低,并且需要较强的计算性能,但抗光照能力强,适合室外场景。

本文所使用的 RGB-D 相机是 Intel 的 Realsense SR300 相机,SR300 采用的结构光的原理获取深度^①,其内部结构如图2.2所示。从图2.2可以看出,SR300 内部的传感器主要有彩色摄像头(Color Camera)、红外激光发射器(Infrared Laser Projector)和红外摄像头(Infrared Camera)。Color Camera 是 1920×1080 像素的普通针孔摄像头,用来获取彩色图像;Infrared Laser Projector 和 Infrared Camera 用来获取深度图像或者红外成像图,两种成像流程如图2.3所示。其中当 Infrared Laser Projector 投射带有编码的结构光时,Infrared Camera 可以获取深度图;当投射不带编码的红外光时,Infrared Camera 可以获取红外成像图。正常使用时,往往设置 Infrared Laser Projector 投射带有编码的结构光来获取深度信息。因此,从 RGB-D 相机的使用来看,可以忽略其内部具体结构,将其看成由一个彩色摄像头

^① 此后所提到的 RGB-D 相机均指与 SR300 相机类似的采用结构光原理获取深度的 RGB-D 相机

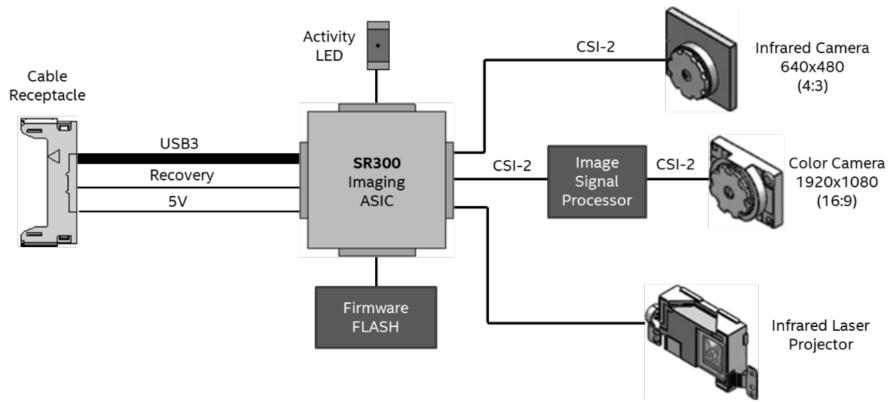


图 2.2 Realsense SR300 内部结构图

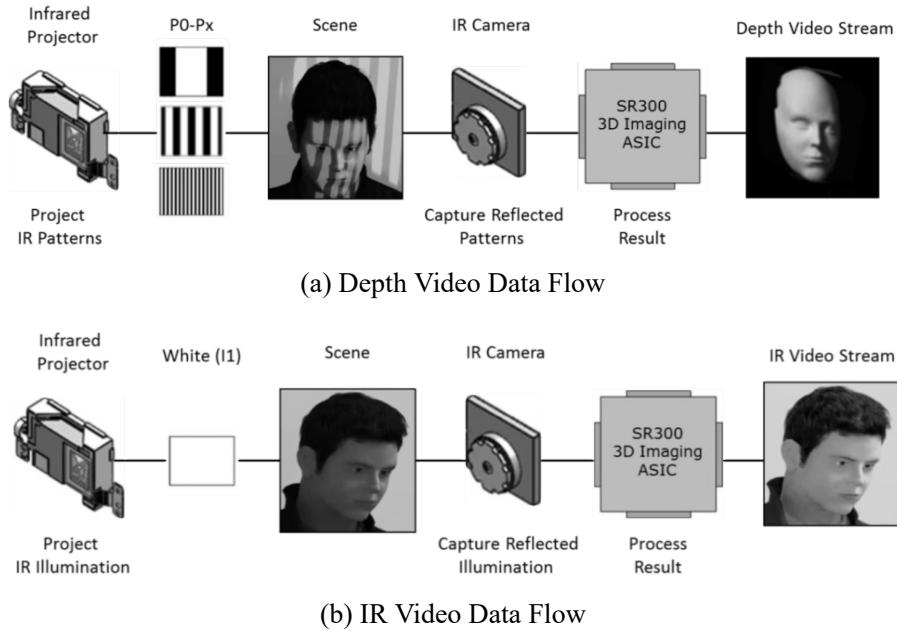


图 2.3 Realsense SR300 深度成像流程

和一个深度摄像头构成,其中彩色摄像获取彩色(RGB)信息,深度摄像头获取深度(depth)信息。

2.2.2 RGB-D 相机的数学模型

图2.4展示了本文所使用的RGB-D相机的基本物理模型,其中彩色摄像头和深度摄像头都使用了针孔(pin-hole)相机模型(Heikkilä 2000)。先考虑普通针孔相机的模型,相机图像坐标系下一点 $\mathbf{u} := [u, v]^T$,对应的三维世界中的一点在相机坐标系下表示为 $\mathbf{X} := [x, y, z]^T$ 。根据针孔相机模型有:

$$z\tilde{\mathbf{u}} = \mathbf{K}\mathbf{X} \quad (2.1)$$

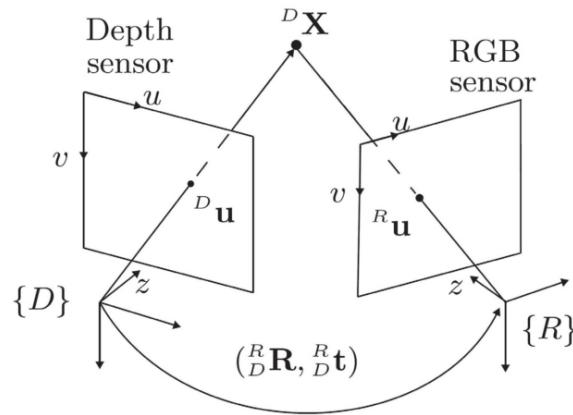


图 2.4 RGB-D 相机模型

其中 $\tilde{\mathbf{u}}$ 表示 \mathbf{u} 的齐次变换形式, 彩色相机的内参矩阵 \mathbf{K} 的定义如下:

$$\mathbf{K} := \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

其中 f_u 和 f_v 分别表示彩色相机在图像坐标轴上的焦距(以像素为单位), u_0 和 v_0 表示彩色相机光心在图像平面的投影中心。

公式2.1还未考虑镜头的畸变, 为了提高相机的精度, 现引入径向畸变(radial distortion)和切向畸变(tangential distortion):

- 径向畸变是由相机透镜的不完善和表面曲率存在误差造成的, 径向畸变的数学模型可以表示为:

$$\begin{cases} \hat{x} = \bar{x}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ \hat{y} = \bar{y}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{cases} \quad (2.3)$$

其中

$$\bar{x} = x/z \quad (2.4)$$

$$\bar{y} = y/z \quad (2.5)$$

$$r = \sqrt{\bar{x}^2 + \bar{y}^2} \quad (2.6)$$

\bar{x}, \bar{y} 表示点 X 在归一化平面上的坐标, \hat{x}, \hat{y} 表示修正径向畸变后的的坐标, k_1, k_2, k_3 表示径向畸变的参数。

- 切向畸变是由于相机透镜与图像平面不平行造成的, 其数字模型可以表示为:

$$\begin{cases} \hat{x} = \bar{x} + (2p_1\bar{x}\bar{y} + p_2(r^2 + 2\bar{x}^2)) \\ \hat{y} = \bar{y} + (p_1(r^2 + 2\bar{y}^2) + 2p_2\bar{x}\bar{y}) \end{cases} \quad (2.7)$$

其中 p_1, p_2 是切向畸变的参数。

- 结合公式2.3和2.7可以得到修正径向畸变和切向畸变的 Brown-Conrady 模型 (BROWN 1966):

$$\begin{cases} \hat{x} = \bar{x}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (2p_1 \bar{x}\bar{y} + p_2(r^2 + 2\bar{x}^2)) \\ \hat{y} = \bar{y}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (p_1(r^2 + 2\bar{y}^2) + 2p_2 \bar{x}\bar{y}) \end{cases} \quad (2.8)$$

通过以上分析,根据公式2.1和2.8可以推导出带有畸变的针孔相机模型:

$$\begin{cases} u = f_u(\bar{x}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (2p_1 \bar{x}\bar{y} + p_2(r^2 + 2\bar{x}^2))) + u_0 \\ v = f_v(\bar{y}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (p_1(r^2 + 2\bar{y}^2) + 2p_2 \bar{x}\bar{y})) + v_0 \end{cases} \quad (2.9)$$

为方便起见,记 $\mathbf{d} := [k_1, k_2, p_1, p_2, k_3]^T$, 定义函数

$$f_{undist}(\mathbf{d}, \mathbf{X}) := \begin{bmatrix} \bar{x}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (2p_1 \bar{x}\bar{y} + p_2(r^2 + 2\bar{x}^2)) \\ \bar{y}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (p_1(r^2 + 2\bar{y}^2) + 2p_2 \bar{x}\bar{y}) \end{bmatrix} \quad (2.10)$$

$$\tilde{f}_{undist}(\mathbf{d}, \mathbf{X}) := \begin{bmatrix} f_{undist}(\mathbf{d}, \mathbf{X}) \\ 1 \end{bmatrix} \quad (2.11)$$

则公式2.9可简化为:

$$\tilde{\mathbf{u}} = \mathbf{K} \cdot \tilde{f}_{undist}(\mathbf{d}, \mathbf{X}) \quad (2.12)$$

其中需要标定的参数有相机内参矩阵 \mathbf{K} (包含未知参数 f_u, f_v, u_0, v_0)以及畸变参数 \mathbf{d} (包含未知参数 k_1, k_2, p_1, p_2, k_3),共 9 个参数。

明确了针孔相机的数学模型后,很容易推出 SR300 的相机模型:

$$\begin{cases} {}^R\tilde{\mathbf{u}} = {}^R\mathbf{K} \cdot \tilde{f}_{undist}({}^R\mathbf{d}, {}^R\mathbf{X}) \\ {}^D\tilde{\mathbf{u}} = {}^D\mathbf{K} \cdot \tilde{f}_{undist}({}^D\mathbf{d}, {}^D\mathbf{X}) \\ {}^R\mathbf{X} = {}_D^R\mathbf{R} {}^D\mathbf{X} + {}_D^R\mathbf{t} \end{cases} \quad (2.13)$$

其中左上标 $\{R\}$ 表示 SR300 相机中的彩色相机(RGB), $\{D\}$ 表示 SR300 相机中的深度相机(Depth), ${}_D^R\mathbf{R}$ 和 ${}_D^R\mathbf{t}$ 表示了彩色相机坐标系和深度相机坐标系之间的齐次变换关系。

2.2.3 RGB-D 相机的标定流程

根据上文所述的 RGB-D 相机的结构及数学模型,RGB-D 相机的标定主要涉及到彩色摄像头内参和畸变的标定,深度摄像头内参和畸变的标定,以及彩色摄像头和深度摄像头之间位姿变换的标定。由于 RGB-D 相机是一种较为新颖的相

机, 所以市面上基本上没有较为成熟通用的标定 RGB-D 相机的方法以及对应的工具。因此本文针对所使用的 Realsense SR300 相机, 设计了一套标定方法。

根据公式2.13可知相机需要标定的参数有彩色相机内参和畸变参数 9 个, 深度相机内参和畸变参数 9 个, 彩色相机和深度相机之间的位姿关系 6 个, 一共 24 个参数。一起标定这 24 个参数理论上是相当困难的, 考虑到普通针孔相机的标定技术已经相当成熟(如张正友的棋盘格标定 (Zhang 2002), 以及 RGB-D 相机中彩色相机和深度相机的解耦性, 因此所设计的标定方法分为三步:

Step 1 标定彩色相机内参以及畸变参数

Step 2 标定深度相机内参以及畸变参数

Step 3 标定彩色相机和深度相机之间的齐次变换关系

步骤 1 标定彩色相机内参以及畸变参数相对来说比较简单, 主要参考文献 (Zhang 2002), 但所使用的标定板是不对称圆盘标定板 (Asymmetrical Circle Board), 如图2.5是 4×11 的不对称圆盘标定板。使用圆盘标定板而非棋盘格标定

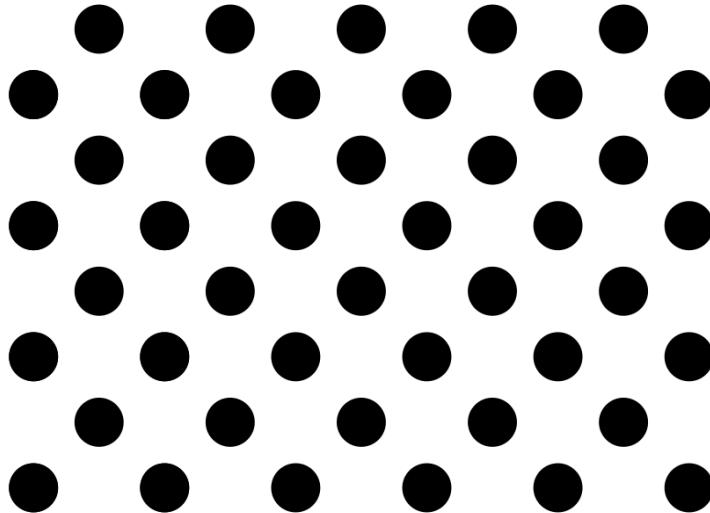


图 2.5 Asymmetrical Circle Board

板的原因是圆盘相对于棋盘格有更高的检测精度, 在某些情况下可以达到 0.1 到 0.01 像素的亚像素精度, 当然代价是相比计算棋盘格的角点, 计算椭圆(圆形经过投影变换后退化为椭圆)的中心会涉及到较为复杂的数学运算, 这也是为什么工业上大多使用圆盘作为标定板的原因。

步骤 2 标定深度相机内参以及畸变参数的方法和步骤 1 类似, 区别在于深度相机并不能直接获得颜色信息, 因此也不能直接检测图2.5所示的标定板。但是, 幸运的是, 根据前文所述的 SR300 深度相机的原理, 其本质上也是个普通的针孔相机, 只不过在其镜头上加上了滤波片, 可以认为其只对红外光成像。因此, 只要

使用图2.3中的红外成像模式获取红外成像图,在红外成像图上检测标定板。如图2.6所示,在红外成像图中检测出了标定板。

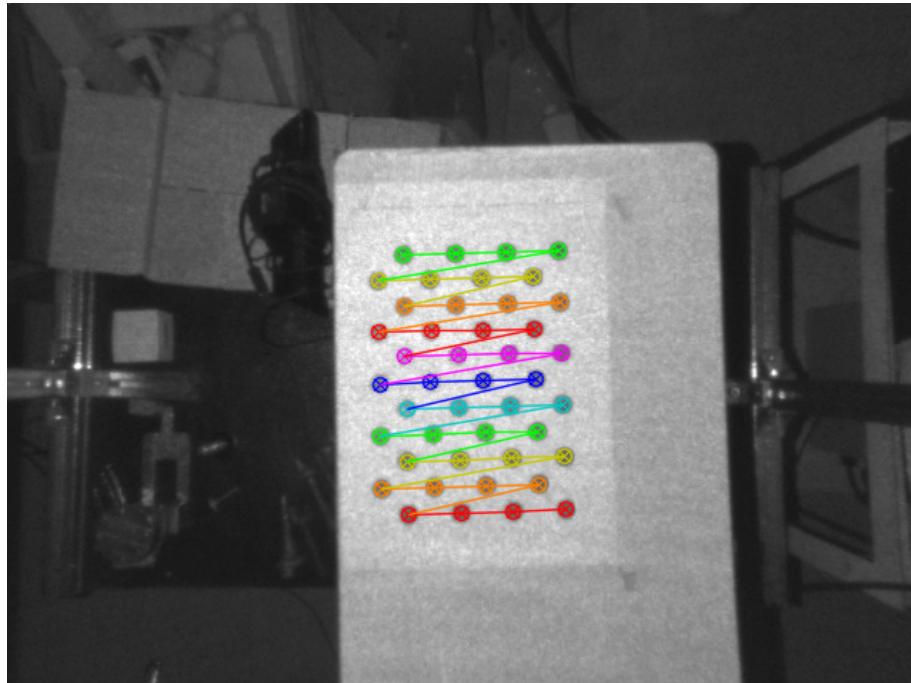


图 2.6 红外成像图中检测标定板

步骤 3 标定彩色相机和深度相机之间的齐次变换关系需要依赖于步骤 1 和步骤 2 中标定出的彩色相机和深度相机的内参和畸变参数,具体做法是将标定板放在彩色相机和深度相机下,使彩色相机和深度相机能够同时检测到标定板,然后分别根据各自的内参和畸变参数计算出标定板的位姿 ${}^R_B \mathbf{H}$ 和 ${}^D_B \mathbf{H}$,其中 ${}^R_B \mathbf{H}$ 是 4×4 的齐次变换矩阵,表示标定板在彩色相机坐标系下的位姿,也是彩色相机坐标系变换到标定板坐标系的齐次变换矩阵; ${}^D_B \mathbf{H}$ 也是 4×4 的齐次变换矩阵,表示标定板在深度相机坐标系下的位姿,也是深度相机坐标系变换到标定板坐标系的齐次变换矩阵。从而所要求的彩色相机坐标系变换到深度相机坐标系的齐次变换矩阵为:

$${}^D_R \mathbf{H} = {}^R_B \mathbf{H} {}^D_B \mathbf{H}^{-1} \quad (2.14)$$

其中

$${}^R_D \mathbf{H} := \begin{bmatrix} {}^R \mathbf{R} & {}^R \mathbf{t} \\ {}^D \mathbf{R} & {}^D \mathbf{t} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix} \quad (2.15)$$

当然,实际标定时,往往采取多组 ${}^R_B \mathbf{H}$ 和 ${}^D_B \mathbf{H}$ 来提高标定的精度。

2.3 对偶 RGB-D 相机

使用 SR300 相机时,发现相机在某些情况下,对一些反光的物体的深度图有严重的缺失,具体如图2.7所示。经过实验,发现这种缺失情况的出现和拍摄的

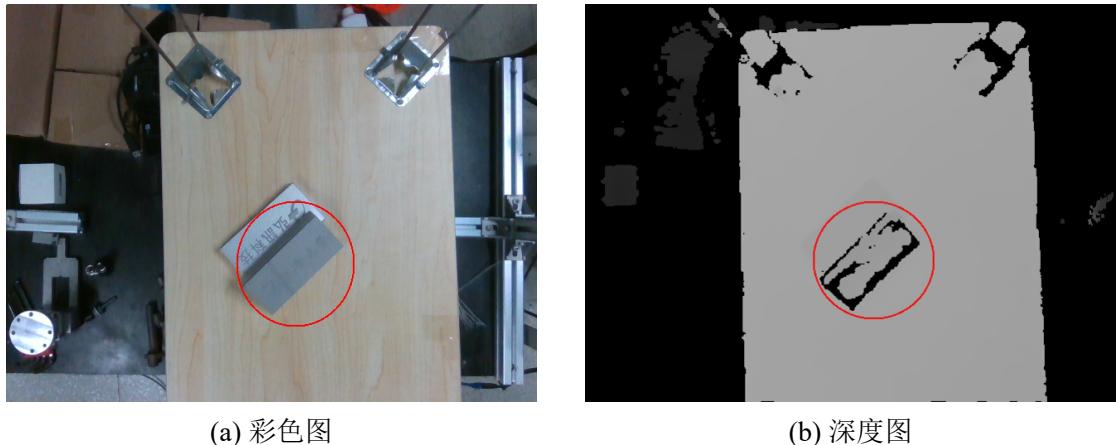


图 2.7 SR300 采集的物体深度信息部分缺失情况下的深度图

角度以及光线有关,因此本文提出一种组合相机对偶 RGB-D 相机(Dual RGB-D Camera)。

2.3.1 对偶 RGB-D 相机原理与结构

对偶 RGB-D 相机在原 RGB-D 相机的基础上,通过增加一个与原相机呈 180 度夹角的 RGB-D 相机构成,实际物理结构如图2.8所示。

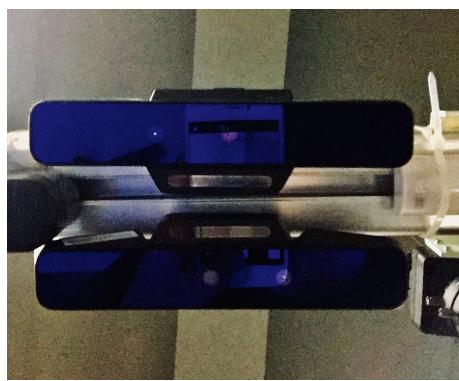


图 2.8 对偶 RGB-D 相机实际物理结构

对于对偶 RGB-D 相机,当其中一个相机深度图出现严重缺失时,另外一个相机的深度图往往不会在相同的地方深度信息出现严重的缺失,如图2.9所示^②,有

^② 实际上相机采集的图像与上相机采集的图像相差了 180 度,为了方便起见,都将下相机采集的图像旋转了 180 度

效的避免了单个 RGB-D 相机某些情况下深度信息严重缺失的情况。

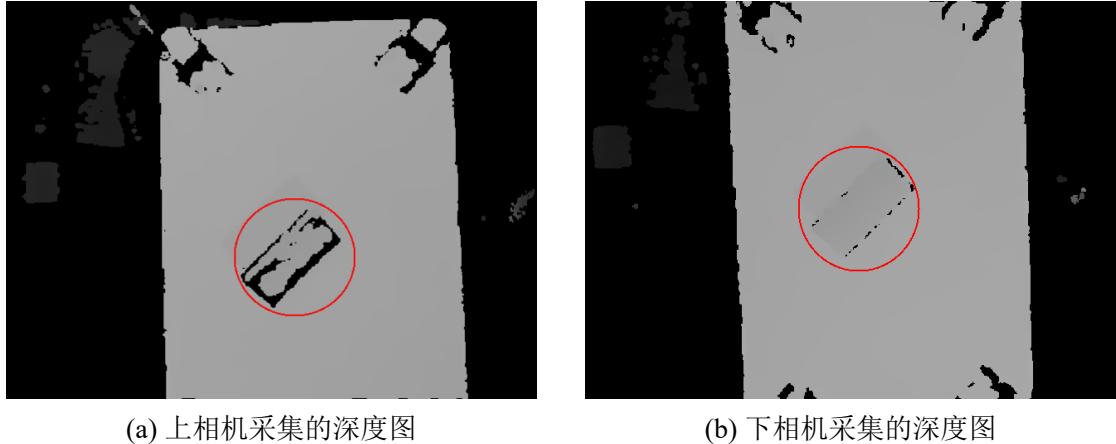


图 2.9 对偶 RGB-D 相机采集的左右两张深度图

除此之外,对偶 RGB-D 相机还可以利用两个相机的彩色图构成双目,生成第三张深度图,从而通过设计的深度的融合算法将三张深度图融合成为一张质量更高的深度图,其内部原理如图2.10所示。

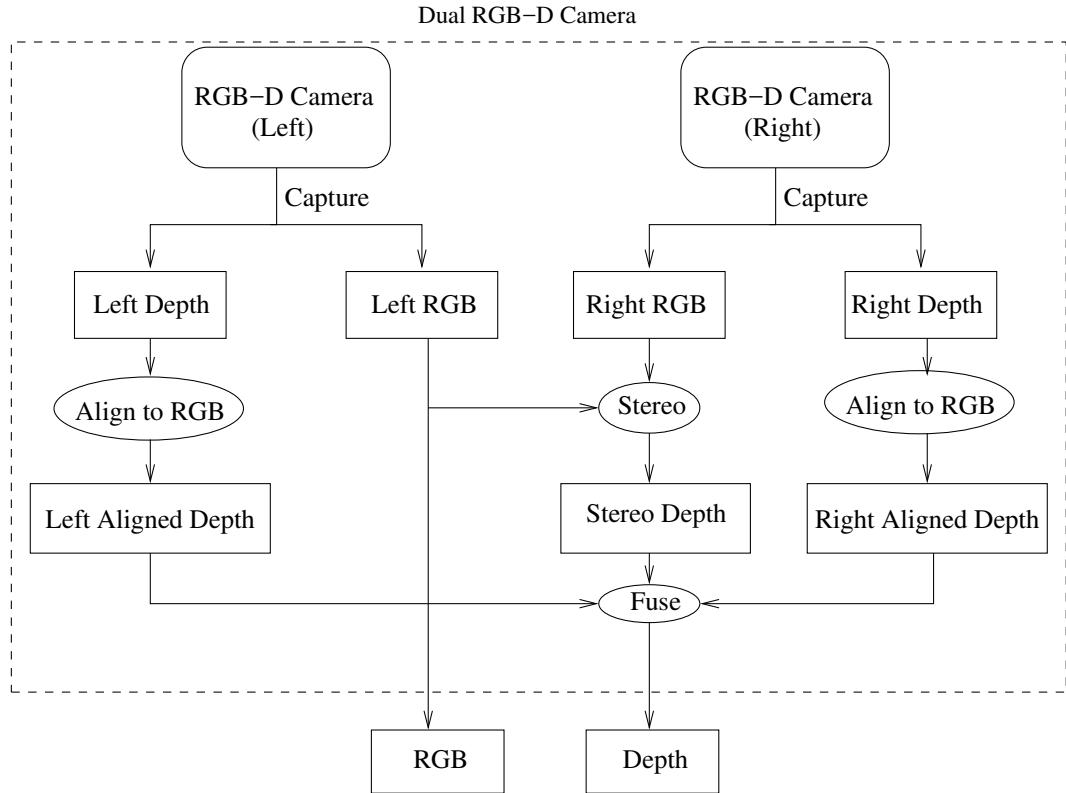


图 2.10 对偶 RGB-D 相机内部原理图

从外部使用来看,对偶 RGB-D 相机也输出一张彩色图、一张深度图。输出的彩色图就是从上相机采集到的彩色图;输出的深度图是由三张深度图融合而成,

并且与输出的彩色图相对齐,对齐的意思是彩色图和深度图相同图像坐标下的颜色信息和深度信息对应的实际物理世界中相同的一点,对齐的意义在于方便后续的一些图像处理的算法。

从内部实现来看,主要涉及到三个部分:

- 将深度图与输出的彩色图对齐 (Align to RGB)
- 利用上相机采集的彩色图和下相机采集的彩色图,通过双目匹配算法形成一张新的深度图
- 融合上相机对齐后的深度图、下相机对齐后的深度图和双目匹配得到的深度图

将深度图与彩色图对齐,相对来讲实现还是比较简单的,对齐深度图的具体流程如算法1所示。算法1主要将深度图中每个点的图像坐标利用该点的深度信息反

算法 1: Align Depth Frame

Input: Raw Depth Frame $Raw_D_{dh \times dw}$

Output: Aligned Depth Frame $Aligned_D_{ch \times cw}$

```

1 for  $p$  in  $Aligned\_D$  do
2    $p = 0$ 
3 for  $dy = 1; dy <= dh; ++dy$  do
4   for  $dx = 1; dx <= dw; ++dx$  do
5     通过深度相机内参将点  $(dx, dy)$  反投影到三维空间一点  ${}^D X$ ;
6     坐标变换  ${}^R X = {}_D^R R {}^D X + {}_D^R t$ ;
7     通过彩色相机内参将点  ${}^R X$  投影变换到彩色图像坐标系下一点
8      $(cx, cy)$ ;
9     if  $cx$  in  $(0, cw]$  and  $cy$  in  $(0, ch]$  then
10     $Aligned\_D(cx, cy) = Raw\_D(dx, dy);$ 

```

投影变换到实际三维空间中一点,然后将该点坐标变换到彩色相机坐标系下,最后通过彩色相机的内参将该点在彩色相机坐标系下的三维坐标投影变换到彩色图像上的二维坐标。实际对齐三张深度图时,对于上相机深度图对齐到上相机彩色图,需要分别知道上相机深度相机和彩色相机的内参和畸变参数以及深度相机与彩色相机之间的齐次变换关系(通过相机标定这些参数都可以得到);双目匹配得到的深度图理论上可以有两张,一张与上相机校准后的彩色图像对齐,另一张与下相机校准后的彩色图像对齐,简单起见,选择与上相机对齐的深度图,然后通过上相机校准所使用的旋转矩阵的逆矩阵即可得到与原上相机彩色图像对齐的

深度图;对齐下相机到上相机彩色图,除了要知道下相机标定的参数外,还需要知道下相机与上相机之间的齐次变换关系(通过对偶 RGB-D 相机的标定得到)。

利用上下相机采集到的两张彩色图获取深度信息主要分为三步:

- 分别对两张原始图像进行校准
- 在校准后的两张图像上通过匹配算法得到视差图
- 通过视差图获取深度图

对两张原始图像进行校准主要通过双目相机的标定实现,使得校准后的两张图像的极线对齐,如图2.11所示,其中绿色的直线便是图像对齐后的部分极线,可以看

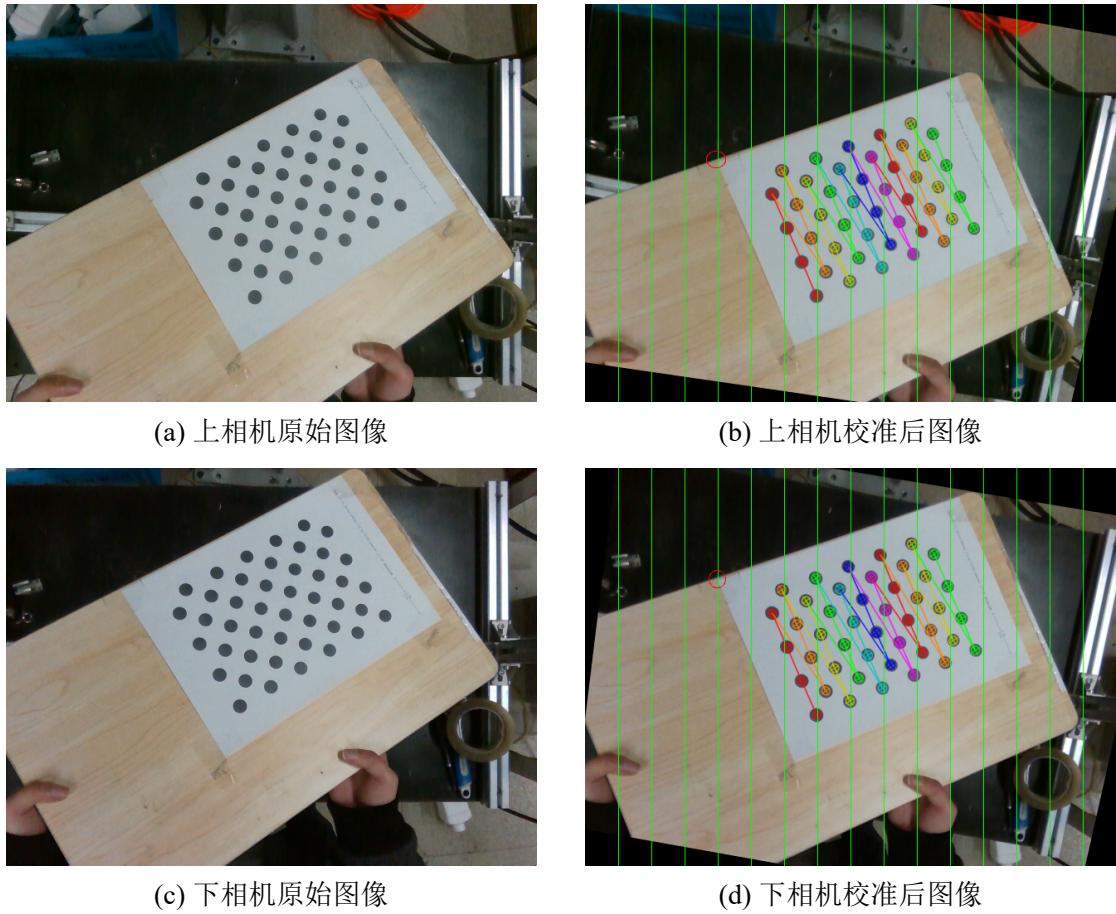


图 2.11 双目相机原始图像和校准后图像

出校准后的图像的对应点都分布在对齐的极线上(如图中用红色圈出的一对对应点所示),这样使得双目的匹配算法的搜索从二维缩小到了一维,只需要在极线上找对应点即可,能更快更稳定地在两张图中找到对应点。双目匹配算法使用的是 ELSA 算法 (Geiger et al. 2010),通过 ELSA 算法可以从两张校准后彩色图像上得到对应的视差图,视差图到深度图的变化可以通过公式2.16得到:

$$z = \frac{\{T, R\} f B}{-(\{T, R\} v_0 - \{B, R\} v_0) + \{T, R\} d} \quad (2.16)$$

其中上标 $\{T, R\}$ (Top,RGB) 表示上相机的 RGB 摄像头, $\{B, R\}$ (Bottom,RGB) 表示下相机的 RGB 摄像头, B 表示基线长度, ${}^{\{T, R\}}d$ 表示视差。一般地, 会人为地校准过程中使得 ${}^{\{T, R\}}v_0 - {}^{\{B, R\}}v_0 = 0$, 从而公式2.16可以简化为:

$$z = \frac{{}^{\{T, R\}}fB}{{}^{\{T, R\}}d} \quad (2.17)$$

融合上相机对齐后的深度图、下相机对齐后的深度图以及双目匹配得到的深度这三张深度图的算法首先做的是分别对这三张深度图进行预处理, 填补一些深度缺失的像素, 因为对齐后的深度图和双目匹配得到的深度图深度信息都有细微的缺失, 填补深度信息缺失的方法如算法2所示。算法2主要实现对于深度缺失的

算法 2: Fill Holes in Depth Frame

Input: Depth Frame $D_{h \times w}$

Output: Filled Depth Frame $FD_{h \times w}$

```

1 for  $y = 1; y <= h; ++y$  do
2   for  $x = 1; x <= w; ++x$  do
3     if  $valid(D_{x,y})$  then
4        $FD_{x,y} = D_{x,y};$ 
5     else
6        $FD_{x,y} = NAN;$ 
7       bool leftTop =  $valid(D_{x-1,y-1})$  or  $valid(D_{x,y-1})$  or  $valid(D_{x-1,y});$ 
8       bool leftBottom =  $valid(D_{x-1,y+1})$  or  $valid(D_{x,y+1})$  or  $valid(D_{x-1,y});$ 
9       bool rightTop =  $valid(D_{x+1,y-1})$  or  $valid(D_{x,y-1})$  or  $valid(D_{x+1,y});$ 
10      bool rightBottom =  $valid(D_{x+1,y+1})$  or  $valid(D_{x,y+1})$  or  $valid(D_{x+1,y});$ 
11      if  $leftTop$  and  $leftBottom$  and  $rightTop$  and  $rightBottom$  then
12        validPoints = {};
13        for  $dy = -1; dy <= 1; ++dy$  do
14          for  $dx = -1; dx <= 1; ++dx$  do
15            if  $valid(D_{dx,dy})$  then
16              push back  $D_{x,y}$  to validPoints;
17            if  $max(validPoints) - min(validPoints) < 0.05$  then
18               $FD_{x,y} = mean(validPoints);$ 

```

点, 将检查其周围的深度信息, 当其四个角上都有有效的深度信息时, 并且周围有

效深度信息的极值小于一定阈值时,会用周围有效深度信息的均值填充该缺失的点。实际的效果如图2.12所示。分别对深度图进行预处理后,将会对三张深度图

@TODO: fill hole 效果图

图 2.12 填补深度信息缺失算法效果图

进行线性叠加得到最终的深度图,基本叠加的公式如2.18所示。

$$d_{fuse} = \frac{w_1 d_{left} + w_2 d_{right} + w_3 d_{stereo}}{w_1 + w_2 + w_3} \quad (2.18)$$

其中 w_1, w_2, w_3 分别表示上相机深度、下相机深度以及双目匹配深度的权重, SR300 相机得到深度的精度比双目计算得到的深度要高, 所以实际使用时 w_1, w_2 要比 w_3 大许多。融合三张深度图的理论相对简单,但实际上,三张深度图的深度信息并非都会永远有效,因此根据实际情况实际的融合算法如3所示。算法3不仅考虑了深度缺失的情况,对于深度信息差值过大的情况也进行了处理。实际处理的效果如图2.13所示。

@TODO 深度融合算法效果图

图 2.13 深度融合算法效果图

2.3.2 对偶 RGB-D 相机的标定流程

对偶 RGB-D 相机的标定流程可以分为三步:

Step 1 分别标定好单个 RGB-D 相机

Step 2 标定出两个彩色相机之间的齐次变换关系

Step 3 标定出矫正彩色图像的旋转矩阵以及矫正后图像的投影矩阵

单个 RGB-D 相机的标定在2.2.3小节中已经详细叙述过了,分别标定完单个 RGB-D 相机后,后面的步骤其实就等价于双目标定了。双目的几何结构如图2.14所示,标定出两个彩色相机之间的齐次变换关系,即图2.14中的 H ,简单地可以通过 8 点法 (Sur et al. 2008) 先求出基础矩阵 (Fundamental Matrix) F , 即所谓的“弱标定”,然后根据相机的内参矩阵可求得本质矩阵 (Essential Matrix) E :

$$E = K^T F K' \quad (2.19)$$

其中 K 和 K' 分别是两个相机的内参矩阵。求得本质矩阵后可以通过奇异值分解求得齐次变换矩阵的旋转矩阵 R 和平移向量 T :

$$\begin{cases} E &= U \Sigma V^T \\ R &= U R_Z(\frac{\pi}{2}) V^T \\ [T]_x &= U R_Z(\frac{\pi}{2}) \Sigma U^T \end{cases} \quad (2.20)$$

算法 3: Fuse Depth Frames

Input: leftDepth, rightDepth, stereoDepth**Output:** fuseDepth

```

1 Initialize w1,w2,w3;
2 for ( $d_1, d_2, d_3, d_4$ ) in ( $leftDepth, rightDepth, stereoDepth, fuseDepth$ ) do
3   validDepth = [], validWeight = [];
4   for  $i = 1$  to  $3$  do
5     if  $d_i$  is valid then
6       push back  $d_i$  to validDepth,  $w_i$  to validWeight;
7   if size of validDepth == 0 then
8      $d_4 = \text{NAN}$ ;
9   else if size of validDepth == 1 then
10     $d_4 = \text{validDepth}[1]$ ;
11   else if size of validDepth == 2 then
12     if extremum of validDepth < 0.03 then
13        $d_4 = \text{validDepth} \cdot \text{validWeight} / \text{sum of validWeight};$ 
14     else
15        $d_4 = \text{NAN}$ ;
16   else
17     mediumDepth = medium(validDepth);
18      $d_4 = 0$ , sum = 0;
19     for ( $d, w$ ) in (validDepth, validWeight) do
20       if  $abs(d - \text{mediumDepth}) < 0.03$  then
21          $d_4 += d * w;$ 
22         sum += w;
23       if sum > 0 then
24          $d_4 = d_4 / \text{sum};$ 
25       else
26          $d_4 = \text{NAN};$ 

```

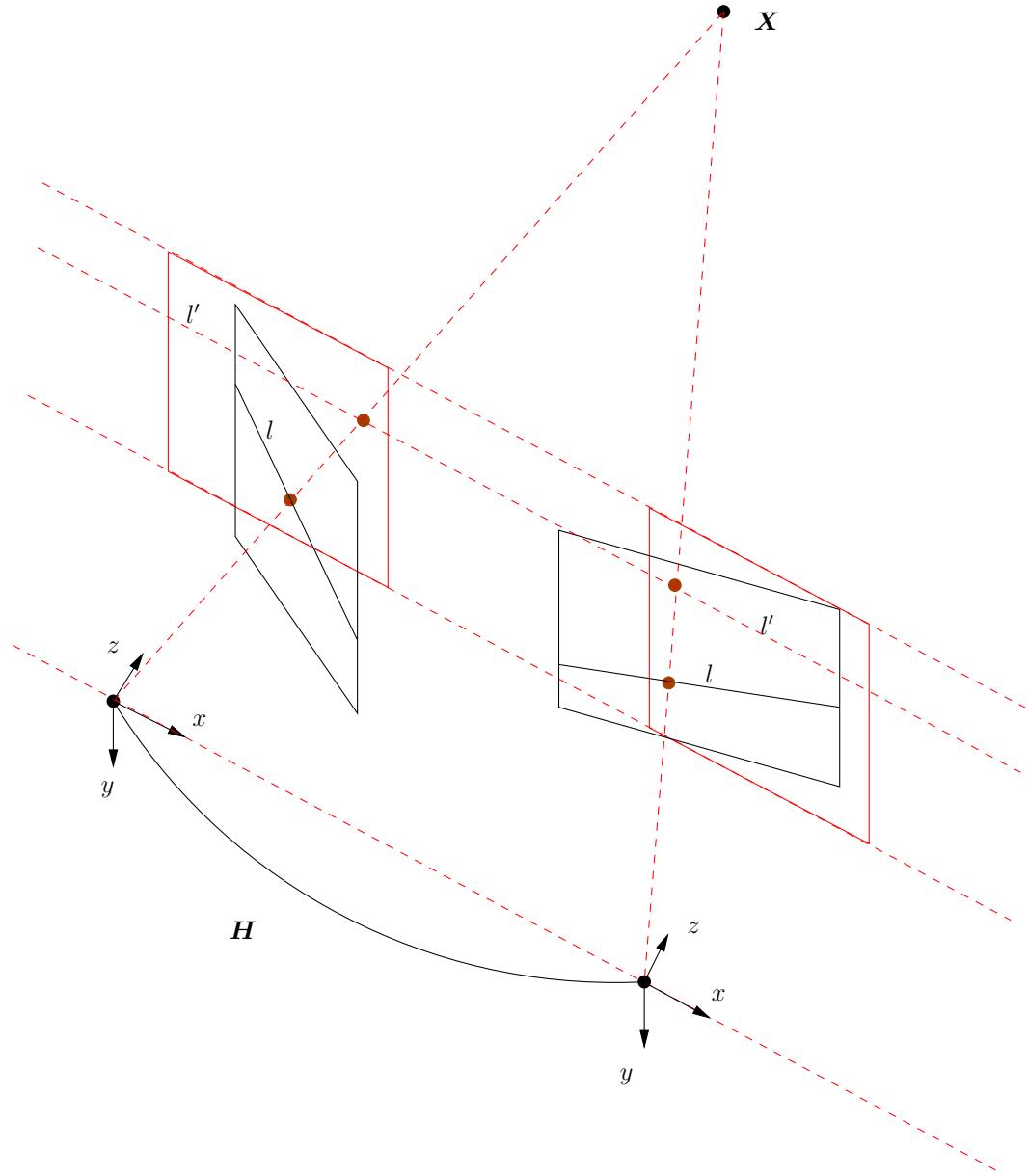


图 2.14 双目几何结构

其中 $R_Z(\theta)$ 表示绕 Z 轴旋转 θ 角的旋转矩阵, $[T]_x$ 的定义如下:

$$[T]_x = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \quad (2.21)$$

矫正彩色图像的旋转矩阵会将图2.14中黑色线框的图像平面变换到红色线框的图像平面上,使得对应点在两张图像的同一条极线上。矫正彩色图像的旋转矩阵的计算参考文献 (Loop et al. 2001), 此步标定完最终可以得到:

- 两个相机的矫正旋转矩阵 R_1, R_2
- 两个矫正坐标系下的投影矩阵 P_1, P_2

TODO 深度测试实验装置图

图 2.15 深度测试实验装置

- 主相机^③的投影变换矩阵 Q

其中

$$Q = \begin{bmatrix} 1 & 0 & 0 & -u_0 \\ 0 & 1 & 0 & -v_0 \\ 0 & 0 & 0 & f \\ 0 & 0 & 1/B & 0 \end{bmatrix} \quad (2.22)$$

包含了公式2.17由视差计算深度的所有参数。

2.4 深度图质量测试实验

RGB-D 相机相对于彩色图我们更关心其深度图的质量,因此设计实验测试了所采用的 SR300 相机深度图的质量以及改进的由两个 SR300 相机所组成的对偶 RGB-D 相机深度图的质量。通过实验,主要考察相机采集的深度在不同距离下的填充率、精度和噪声这三个指标。实验器材除了测试所用的相机,还需要沿固定方向运动的导轨,以及固定在导轨上的平板,相机通过采集平板上的深度信息来计算填充率、精度和噪声这三个指标。实际实验时,由于实验室没有沿固定方向运动的导轨,但是有六轴机械臂,所以将平板固定在机械臂末端,然后通过机械臂示教器控制机械臂末端沿固定方向移动,并且移动的距离可在示教器上读出,整个实验装置如图2.15所示。

2.4.1 实验流程

实验流程示意图如图2.16所示,其中 z_c 是相机坐标系的 z 轴方向, z_r 是机械臂末端运动方向,也是平板运动方向, z_b 是垂直于平板的方向, $D_i := \{d_1, d_2, \dots, d_{n_i}\}$ 是相机采集到平板的深度信息。具体实验步骤如下:

- 固定机械臂和相机,通过手眼标定(具体见??小节)得到相机坐标系和机器人坐标系之间的齐次变换关系
- 固定平板到机械臂末端
- 通过相机采集平板的深度信息 D_0 ,记录此时示教器上机械臂末端在机器人坐标系 z 轴上的值 r_0

^③ 另外一个相机的投影变换矩阵也可以得到,但没有必要。

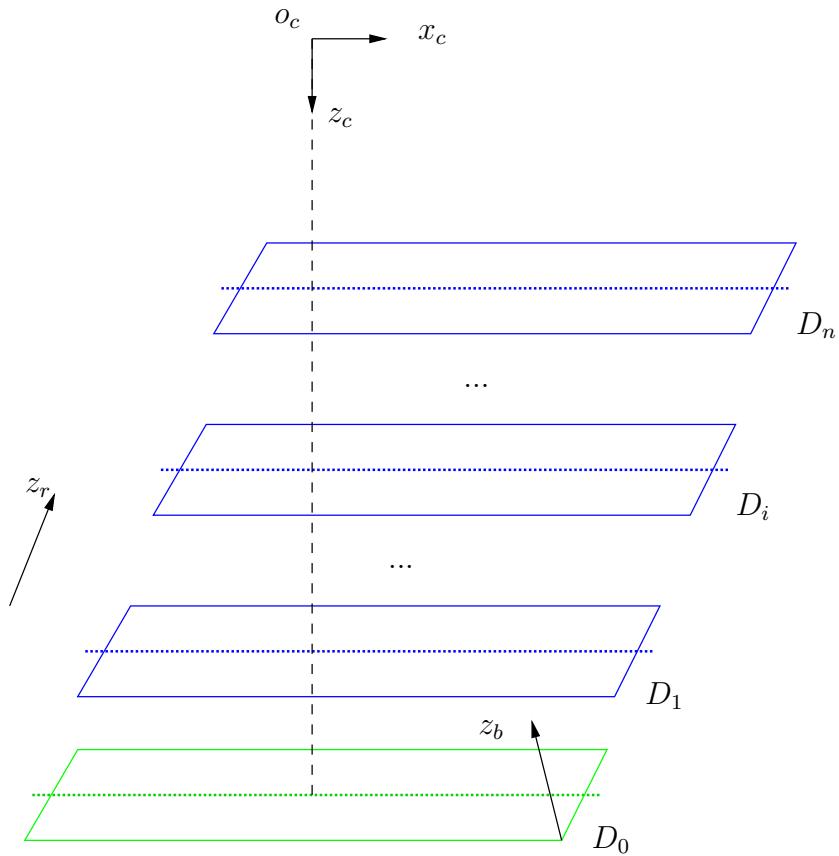


图 2.16 实验流程图

- 控制机械臂示教器使机械臂末端沿机器人坐标系 z 轴运动, 分别记录此时平板的深度信息 D_i 和机械臂末端位置在 z 方向上的值 r_i
- 重复上述步骤, 直到采集满 n 组数据

2.4.2 实验原理

通过上述实验步骤采集完数据后, 需要计算深度信息的填充率、精度和噪声这三个指标, 下面分别介绍这三个指标的定义和计算方式。

填充率表示深度图中有效深度信息的百分比, 计算相对简单, 首先在采集到的深度图中拟合出平板的平面方程 $\theta_i^T \bar{p} = 0$, 其中 $\theta_i := [\theta_i(1), \theta_i(2), \theta_i(3), \theta_i(4)]$ 为平面方程参数, 然后通过深度图中每个像素到平面的距离确定平板在深度图中的闭合区域。定义像素表示的点到平面的距离小于规定的阈值 δ 时该像素深度信息有效, 最后统计在该闭合区域中像素的总点数 M_i 和有效深度信息的像素点个数 M'_i , 则第 i 组数据测得的填充率为

$$FillRate_i = \frac{M'_i}{M_i} \times 100\% \quad (2.23)$$

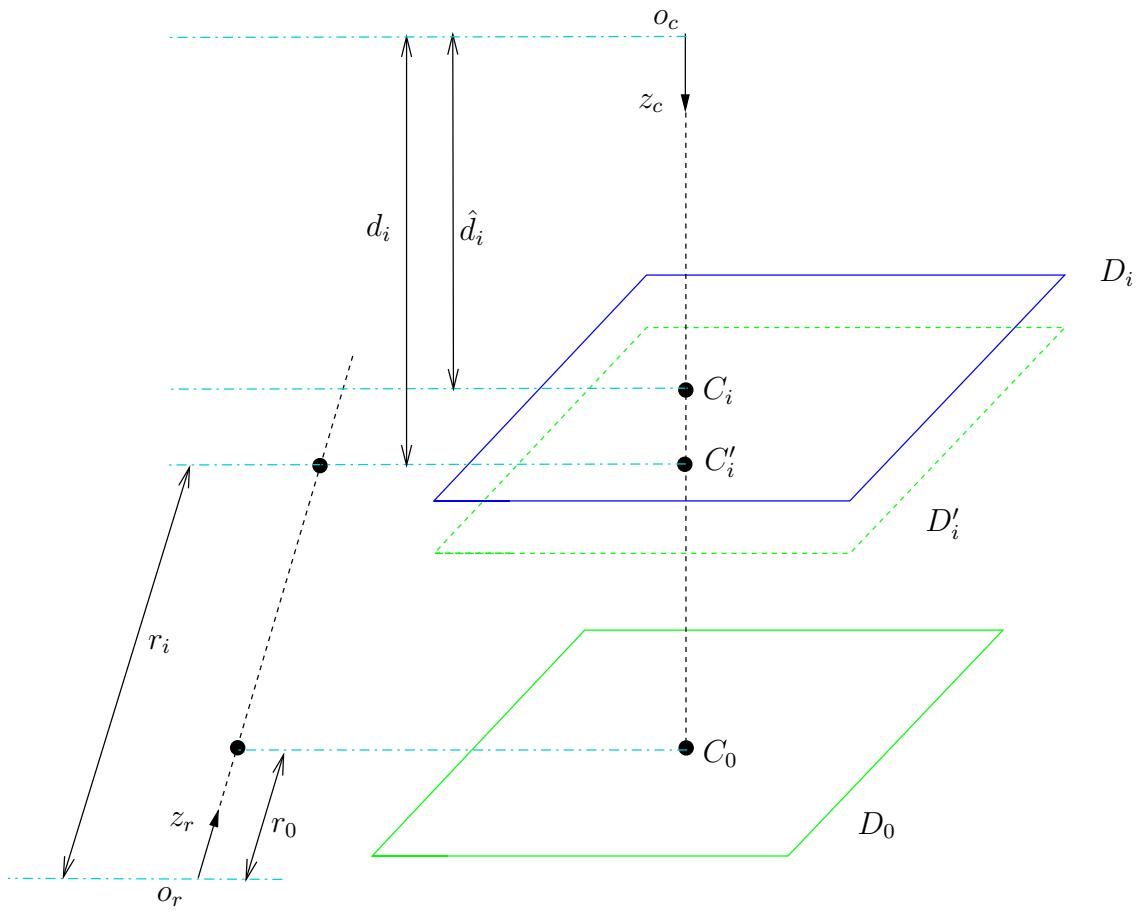


图 2.17 精度测量原理图

精度表示深度图测量的深度的精度, 定义如下:

$$Precision_i = \frac{|\hat{d}_i - d_i|}{d_i} \times 100\% \quad (2.24)$$

其中 \hat{d}_i 表示相机坐标系原点到拟合平面的距离, d_i 表示相机坐标系原点到实际平板的距离, 详细如图2.17所示。 \hat{d}_i 的计算相对简单, 根据 D_i 中拟合出相机坐标系下平板的平面方程的参数 Θ_i , 容易得到

$$\hat{d}_i = -\frac{\theta_i(4)}{\theta_i(3)} \quad (2.25)$$

为了获得实际平板的与相机坐标系的原点的距离, 首先以 D_0 平面为基准, 将其沿机器人坐标系 z 轴方向移动 $r_i - r_0$, 得到新的平面方程 θ'_i , 则

$$d_i = -\frac{\theta'_i(4)}{\theta'_i(3)} \quad (2.26)$$

噪声定义为平板闭合区域内点距离平面拟合方程的距离的均方根 (RMS):

$$Noise_i = \sqrt{\frac{1}{J} \sum_{j=1}^J \delta_j^2} \quad (2.27)$$

@TODO: sr300 and dual-rgb-d camera's fill rate, precision, nosie plot

图 2.18 深度图质量测试实验结果

其中 δ_j 为点到拟合平面的距离。

2.4.3 实验结果

实验分别对 SR300 和对偶 RGB-D 相机在平板距离相机 0.2 到 1.2m 范围内采集了 $1 + 19$ 组数据，测得每组数据深度信息的三个指标，如图2.18所示。从图2.18可以看出，随着距离的增加，深度相机的填充率、精度下降，噪声增加。所设计的对偶 RGB-D 相机相比 SR300 相机有更高的填充率，与设计时的初衷一致，毕竟结合了两个相机的深度信息，理论上深度信息的填充率就应该有所增加；对偶 RGB-D 相机的精度与 SR300 相机相比没有太显著的提升，但噪声有着明显的下降。综上可以得出对偶 RGB-D 相机所采集的深度图相比单个 RGB-D 相机有着更高的填充率、更低的噪声，深度图的质量更好。

2.5 本章小结

本章首先介绍了 RGB-D 相机的现状，然后详细介绍了以结构光为原理的 RGB-D 相机 SR300 的原理以及标定方法，针对 SR300 对于反光物体深度信息缺失的情况，通过组合两个 RGB-D 相机构成对偶 RGB-D 相机实现采集高质量的深度图，并给出了对偶 RGB-D 相机的标定流程。最后设计了深度图质量测试的实验，证明了对偶 RGB-D 相机比单个 RGB-D 相机有更高的填充率，更低的噪声。

第3章 基于RGB-D图像的目标检测算法

本章主要介绍所提出的两种基于RGB-D图像的目标检测算法3D Faster R-CNN和3D Mask R-CNN。3D Faster R-CNN是在Faster R-CNN(Ren et al. 2016)的基础上,通过引入深度图以解决单从RGB图难以检测缺少纹理物体(Textureless Object)的问题,并且还引入了Spatial Transformer结构使得提取的特征具有旋转不变性。由于3D Faster R-CNN目标检测的结果是框出目标的Bounding Box,因此使得一些框住细长目标的Bounding Box内大部分像素并不属于该目标,这就使得后面的点云匹配算法难以得到满意的结果。因此3D Mask R-CNN根据Mask R-CNN(He et al. 2017)对Faster R-CNN的改进思路,对3D Faster R-CNN进行了改进,使得其不仅能得到目标的Bounding Box,还能得到目标的Mask(可以知道Bounding Box内属于检测目标的像素),大大减少了后续匹配算法的难度。

3.1 3D Faster R-CNN

3D Faster R-CNN算法的整体结构如图3.1所示。

相比于Faster R-CNN,本文所提出的3D Faster R-CNN主要增加对深度信息的处理和Spatial Transformer,分别用于解决Faster R-CNN在实际应用时所不能解决的问题:

- 难以检测出缺少纹理的物体
- 对物体的旋转敏感,提取的特征不具有旋转不变性

对于缺少纹理的物体,单从RGB图中很难检测出目标,这是一个很显然的问题,但是现在我们可以从对偶RGB-D相机中获取深度图,对于纹理少的物体,可以从深度图中提取特征检测出目标,所以现在的关键问题是如何从深度图中提取特征,并结合到Faster R-CNN中,本文所提出的方法是将深度图转换到HHA,然后再使用CNN提取特征,具体后文会详细介绍。

Faster R-CNN对于物体旋转敏感的问题,归根到底是因为CNN所提取的特征不具有旋转不变性,实际出现这种问题的情况,如图3.2所示,其中图(b)只是将图(a)旋转了180度,由于CNN所提取的特征不具有旋转不变性,并且训练所实验的图片中的宠物都是头朝上的,即使图(a)在训练集中,将其旋转180度后,也无法从中检测出目标来。解决这个问题有两个思路:

- Data Augmentation

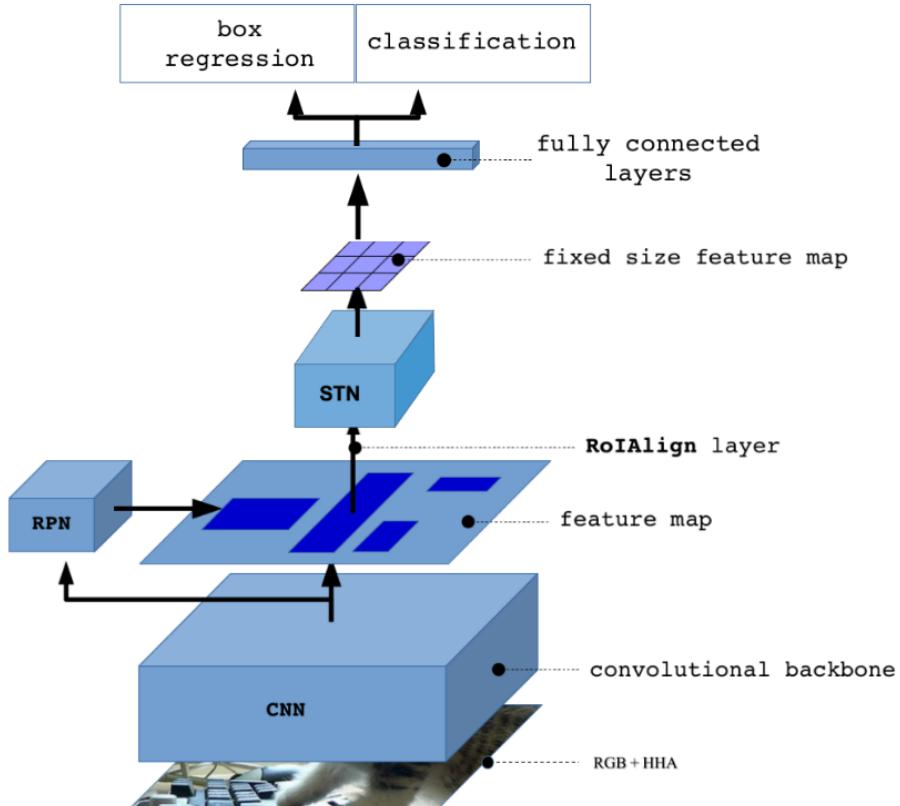


图 3.1 3D Faster R-CNN 结构

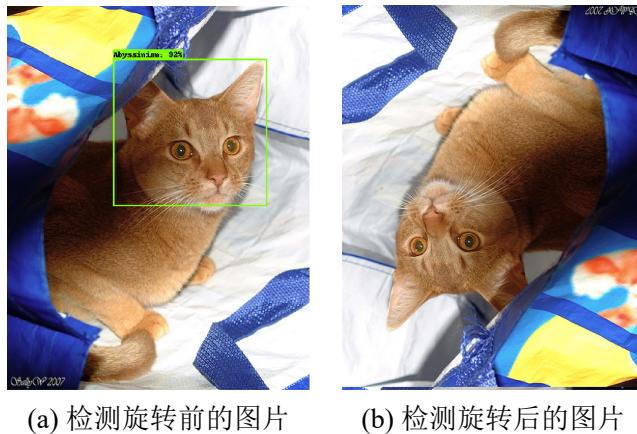


图 3.2 Faster R-CNN 检测识别宠物猫示例

- Spatial Transformer

Data Augmentation 是通过对训练集中的图片进行旋转以获取不同角度的图片, 通过这种方式增大数据集从而使得最终训练得到的模型对各种角度的图片都能识别; Spatial Transformer 是一种特殊的网络结构, 本文所使用的就这种方式, 后文会详细介绍。

3.1.1 Faster R-CNN

为了更好地介绍所提出的3D Faster RCNN算法,先回顾一下Faster R-CNN算法。Faster R-CNN的网络结构如图3.3所示,Faster R-CNN的由两个核心模块构

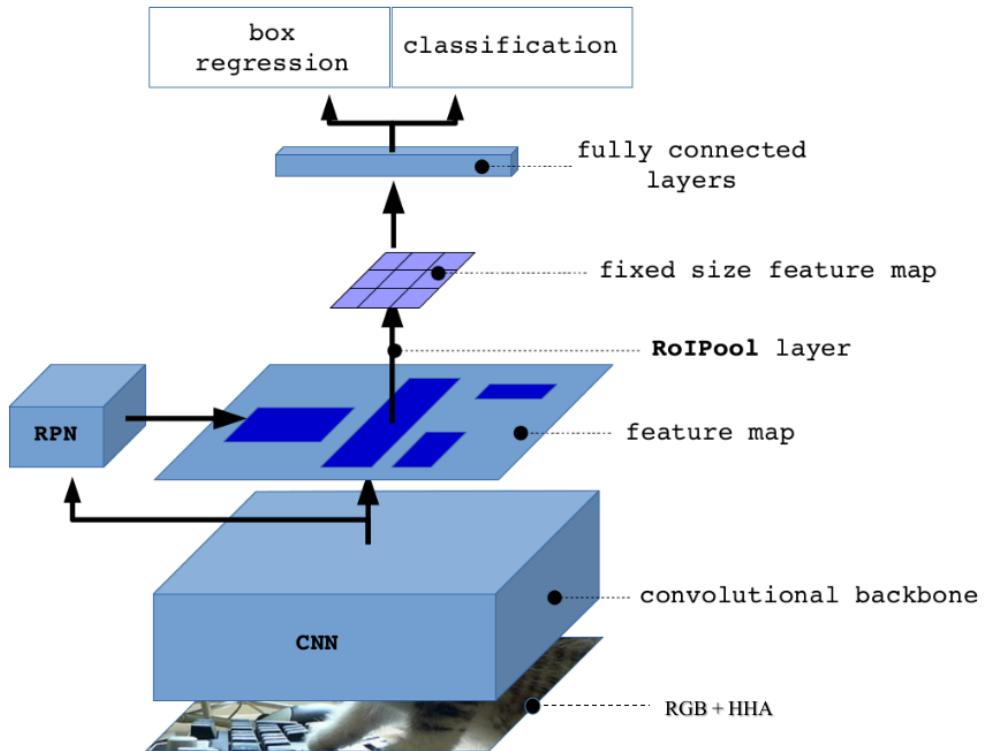


图 3.3 Faster R-CNN 结构

成:

- RPN(Region Proposal Network)
- Fast R-CNN

整个网络是一个端到端(end-to-end)的目标检测网络,输入图片,输出图片中检测到的目标的类别和Bounding Box。RPN模块输出候选框,形象地说,RPN模块告诉Fast R-CNN模块去哪里检测目标,Fast R-CNN模块输出检测结果。

@todo: faster r-cnn more specific ?

3.1.2 HHA

有了与彩色图对应的深度图,如何有效地利用深度图是一个值得思考的问题。从2012年AlexNet(Krizhevsky et al. 2012)在ImageNet(ImageNet 2011)数据集上的应用开始,深度学习在计算机视觉领域其准确率相比传统方法有了一个很大的提升,因此,本文考虑通过深度学习的方法结合深度图和彩色图进行目标检测。深度学习在彩色图上的应用已经相当成熟,但对于深度图的应用还比较少,如

何使用 CNN 在深度图上提取特征也是一个值得探讨的问题, 是将深度图直接作为一个通道使用 CNN 提取特征? 还是将深度图变换到三维坐标(x, y, z), 然后再在这三个通道上通过 CNN 提取特征? 经过实验和相关调研, 发现将深度图转换为 HHA 图后进行训练的模型有较高的准确率 (Gupta et al. 2014), 因此本文将深度图转换为 HHA 三个通道, 然后再通过 CNN 提取特征。HHA 三个通道分别为:

- 水平方向上视差(Horizontal disparity)
- 距离地面的高度(Height above ground)
- 法向量与重力的夹角(Angle with gravity)

Horizontal disparity: 深度图到视差的转换相对来说十分简单, 理论上视差与深度呈倒数关系, 因此水平方向上的视差计算具体如算法4所示。

算法 4: 计算水平方向上视差

Input: Depth Frame $D_{h \times w}$

Output: Horizontal disparity Frame $H_{h \times w}$

```

1  $h_{floor} = 1/d_{ceil}, h_{ceil} = 1/d_{floor};$ 
2 for  $y \leftarrow 1$  to  $h$  do
3   for  $x \leftarrow 1$  to  $w$  do
4      $H[y, x] = 1/D[y, x];$ 
5      $H[y, x] = (H[y, x] - h_{floor}) / (h_{ceil} - h_{floor});$ 

```

Height above ground: 计算距离地面的高度首先要确定一个世界坐标系, 然后得到世界坐标系到相机坐标系的旋转矩阵 ${}^W_C R$ 和平移向量 ${}^W_C T$, 最后通过坐标变换得到距离地面的高度, 具体如算法5所示。

算法 5: 计算距离地面的高度

Input: Point Cloud $P_{h \times w}$

Output: Height Frame $H_{h \times w}$

```

1 for  $y \leftarrow 1$  to  $h$  do
2   for  $x \leftarrow 1$  to  $w$  do
3      $p = {}^W_C R P[y, x] + {}^W_C T;$ 
4      $H[y, x] = p.z;$ 

```

Angle with gravity: 法向量与重力的夹角的计算相对来说稍微复杂一点, 重力的方向在工作区间内一般与所设的世界坐标系的 z 轴负方向相同, 因此原问题就是求法向量与世界坐标系 z 轴负方向之间的夹角。参考文献 (Gupta et al. 2013),

首先计算深度图中每个点上的法向量,计算点云中一点 p_0 的法向量 \vec{n} 的简单思路如下:

- 找出距离点 p_0 最近的 k 个点: p_1, p_2, \dots, p_k
- 通过最小二乘在点 $\{p_i | i = 0, 1, \dots, k\}$ 中拟合出平面 $Ax + By + Cz + D = 0$
- 点 p_0 的法向量 $\vec{n} = [A, B, C]^T$

考虑到所采集的深度图转换的点云是有序的(Organized Point Cloud),意味着坐标索引相近的点实际物理距离也相近,因此找出距离点 p_0 最近的 k 个点可以通过选取点 p_0 坐标索引附近的点代替,具体地,记点 p_0 在深度图中图像坐标为 (x_0, y_0) ,取点集 $S = \{p_i | x_0 - R \leq x_i \leq x_0 + R, y_0 - R \leq y_i \leq y_0 + R\}$,其中 R 是选取区域的半径。得到法向量后计算法向量与世界坐标 z 轴负方向的角度就十分简单了,整个计算法向量与重力的夹角的算法如6所示。

算法6: 计算法向量与重力的夹角

Input: Point Cloud $P_{h \times w}$

Output: Angle Frame $A_{h \times w}$

```

1 for  $y \leftarrow 1$  to  $h$  do
2   for  $x \leftarrow 1$  to  $w$  do
3     Calculate surface normal  ${}^C\vec{n}$  at point  $P[y, x]$ ;
4      ${}^W\vec{n} = {}^W_C R {}^C\vec{n} + {}^W_C T$ ;
5      $A[y, x] = \arccos(-(\vec{n} \cdot \vec{o}_z) / (\|\vec{n}\| \|\vec{o}_z\|))$ ;

```

计算完上述HHA三个通道后,为了计算和存储方便,分别将三个通道的值线性变换到0到255之间,可视化如图3.4所示。

3.1.3 Spatial Transformer

Spatial Transformer是一个可微模块,根据输入的特征对其进行相应空间变化,输出变换后的特征,如图3.5所示,输入特征 U 经过Spatial Transformer模块后输出特征 V 。Spatial Transformer模块具体可以分为三个部分,如图3.6。简单来讲,第一部分是一个定位网络(localisation network),输入特征 U ,输出需要进行空间变换的参数;第二部分是一个网格生成器(grid generator),根据空间变换的参数生成输入特征中需要变换的点的网格;第三部分是个采样器,根据网格生成器的输出对输入特征进行采样并进行空间变换,生成输出特征。

具体地,记定位网络的输入为特征 $U \in \mathbb{R}^{H \times W \times C}$,其中 W, H, C 分别为长、宽和通道数,网络的输出为空间变化 \mathcal{T}_θ 的参数 θ ,参数 θ 的个数由空间变换的类型决

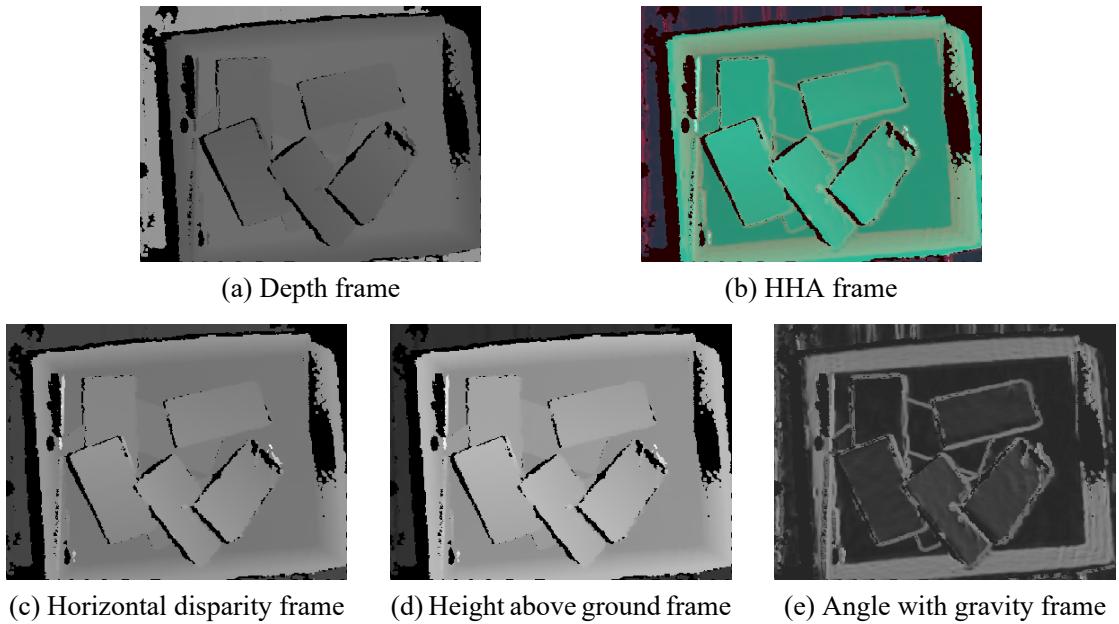


图 3.4 HHA 可视化效果图

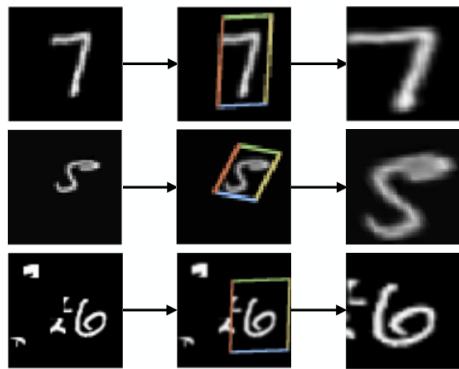


图 3.5 Spatial Transformer 效果图

定,本文所采用的空间变换为 2D 仿射变换,则

$$\mathcal{T}_\theta = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \quad (3.1)$$

定位网络内部可以由一些全连接层或者卷积层再加一个回归层组成。

网格生成器本质上就是在输入特征中选取需要进行空间变化的点,如图??中绿色点便是网格生成器所选取的点,记 Spatial Transformer 的输出特征为 $V \in \mathbb{R}^{H' \times W' \times C}$,其中 W', H', C 分别为输出特征的长、宽和通道数,输出特征的通道数和输入特征的通道数相同,不能改变,并且空间变换 \mathcal{T}_θ 将分别作用于输入 U 的各个通道以保证每个通道上的变换一致。并记点集 $G = \{G_i | G_i = (x_i^t, y_i^t)\}$,其中 (x_i^s, y_i^s) 为输出特征图中点的坐标,由定位网络输出的参数 θ 和 G 我们就可以在输

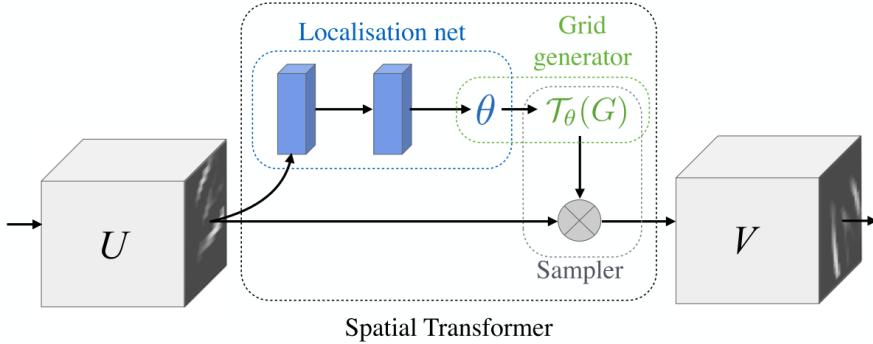


图 3.6 Spatial Transformer 结构图

入特征中确定需要进行空间变换的点的集合 $\mathcal{T}_\theta(G)$:

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \mathcal{T}_\theta(G_i) = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \quad (3.2)$$

其中 (x_i^s, y_i^s) 是输入特征中点的坐标,也是图??中的绿色点。

采样器输入网格生成器生成的点集 \mathcal{T}_θ ,和输入特征 U ,最终输出经过空间变换后的特征 V ,具体如公式3.3所示:

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c k(x_i^s - m; \Phi_x) k(y_i^s - n; \Phi_y) \quad \forall i \in [1 \dots H'W'] \quad \forall c \in [1 \dots C] \quad (3.3)$$

其中 Φ_x 和 Φ_y 是采样核函数 $k()$ 的参数, U_{nm}^c 表示输入特征 U 在坐标 (n, m) 下第 c 个通道上的值, V_i^c 表示输出特征在坐标 (x_i^s, y_i^s) 下第 c 个通道上的值。理论上可以使用任何采样核函数,只要可以对 x_i^s 和 y_i^s 求导,因为网络训练需要对公式3.3求导。以双线性采样核函数为例,公式3.3变为

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|) \quad (3.4)$$

则 V 对 U 和 G 的梯度为

$$\frac{\partial V_i^c}{\partial U_{nm}^c} = \sum_n^H \sum_m^W \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|) \quad (3.5)$$

$$\frac{\partial V_i^c}{\partial x_i^s} = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |y_i^s - n|) \begin{cases} 0 & \text{if } |m - x_i^s| \geq 1 \\ 1 & \text{if } m \geq x_i^s \\ -1 & \text{if } m < x_i^s \end{cases} \quad (3.6)$$

$$\frac{\partial V_i^c}{\partial y_i^s} = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |x_i^s - m|) \begin{cases} 0 & \text{if } |n - y_i^s| \geq 1 \\ 1 & \text{if } n \geq y_i^s \\ -1 & \text{if } n < y_i^s \end{cases} \quad (3.7)$$

3.2 3D Mask R-CNN

Mask R-CNN 相比 Faster R-CNN 不仅可以输出目标的 Class 和 Bounding Box, 还可以输出目标的 Mask。Mask R-CNN 相比 Faster R-CNN 主要的技术要点有:

- 强化了特征提取网络
- 采用 ROIAlign 代替 ROIPooling
- Mask 的损失函数

因此 3D Mask R-CNN 也针对上述三个要点对 3D Faster R-CNN 进行改进, 其结构框架如图3.7所示。

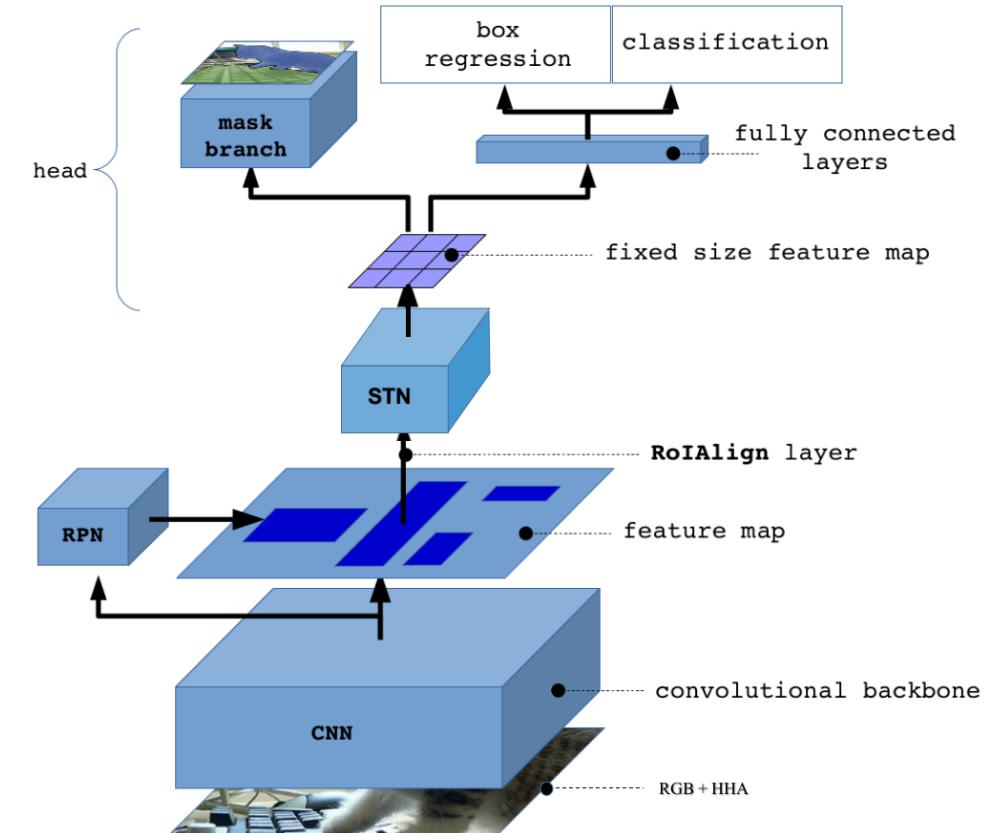


图 3.7 3D mask R-CNN 结构

3.2.1 特征提取网络

3D Faster R-CNN 的特征提取网络使用的是 VGG-16(Simonyan et al. 2014), VGG16 是牛津大学 VGG 组提出的。VGG16 相比最早的 AlexNet 的一个改进是采用连续的几个 3×3 的卷积核代替 AlexNet 中的较大卷积核($11 \times 11, 5 \times 5$)。对于给定的感受野(与输出有关的输入图片的局部大小),采用堆积的小卷积核是优于采用大的卷积核,因为多层非线性层可以增加网络深度来保证学习更复杂的模式,而且代价还比较小(参数更少)。比如,3 个步长为 1 的 3×3 卷积核连续作用在一个大小为 7 的感受野,其参数总量为 $3 \times 9C^2$,其中 C 是通道数,如果直接使用 7×7 卷积核,其参数总量为 $49C^2$ 。而且 3×3 卷积核有利于更好地保持图像性质。

3D Mask R-CNN 改用了 ResNeXt-101+FPN 网络提取特征,该网络主要由 ResNeXt(Xie et al. 2017) 和 FPN(Lin et al. 2017) 两部分构成。ResNeXt 是对残差网络 ResNet(He et al. 2016) 的改进,在介绍 ResNeXt 之前先介绍一下 ResNet。ResNet 为了解决随着网络层数增加,靠前的层梯度会很小,导致训练时学习停滞、梯度消失的问题,引入了残差模块,如图3.8所示。残差单元可以解决学习停滞问

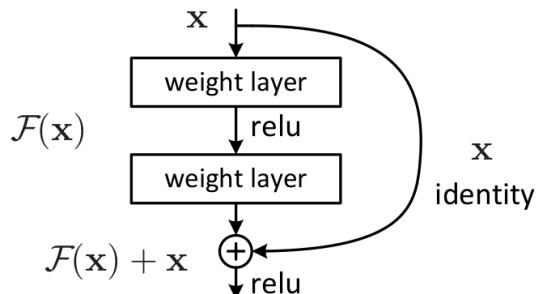


图 3.8 残差模块

题的背后逻辑在于此:想象一个网络 A,其训练误差为 x 。现在通过在 A 上面堆积更多的层来构建网络 B,这些新增的层什么也不做,仅仅复制前面 A 的输出。这些新增的层称为 C。这意味着网络 B 应该和 A 的训练误差一样。那么,如果训练网络 B 其训练误差应该不会差于 A。但是实际上却是更差,唯一的原因是让增加的层 C 学习恒等映射不容易。为了解决这个退化问题,残差模块在输入和输出之间建立了一个直接连接,这样新增的层 C 仅仅需要在原来的输入层基础上学习新的特征,即学习残差,会比较容易。ResNeXt 在 ResNet 的基础上,提出 cardinality 的概念,如图3.9,其中左右两个网络结构有相同的参数个数,左边是 ResNet 的一个区块,右边的 ResNeXt 中每个分支一模一样,分支的个数就是 cardinality,其通过在大卷积核层两侧加入 1×1 的网络层来控制核个数、减少参数个数。因此,与 ResNet 相比,相同的参数个数,ResNeXt 结果更好:一个 101 层的 ResNeXt 网络,

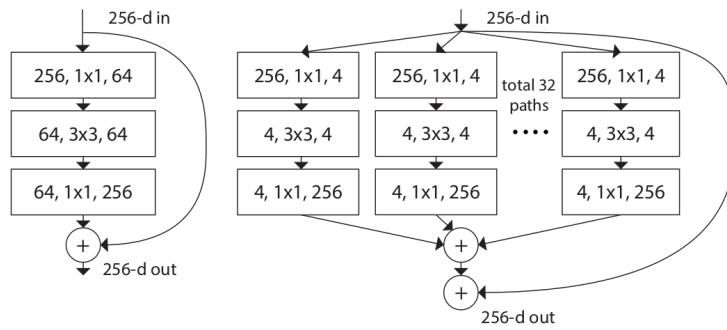


图 3.9 ResNeXt 对 ResNet 的改进

和 200 层的 ResNet 准确度差不多,但是计算量只有后者的一半。

3.2.2 ROIAlign

ROIPooling 采用的是最近邻插值(Nearest neighbor interpolation),即在 resize 时,对于缩放后坐标不能刚好为整数的情况,采用四舍五入的方法,相当于选取离目标点最近的点。虽然这种处理方法对分类问题影响不大,但是现在 Mask R-CNN 需要预测 Pixel 级别的 Mask, ROIPooling 造成的像素不对齐问题对 Mask 的精确度影响很大,因此提出 ROIAlign 代替 ROIPooling, ROIAlign 使用双线性插值(Bilinear interpolation)来获得像素级别的对齐。举例来说,假设在 8×8 的特征图中提取 2×2 的输出, ROIPooling 的示意图如3.10所示, ROIAlign 的示意图如3.11所示。

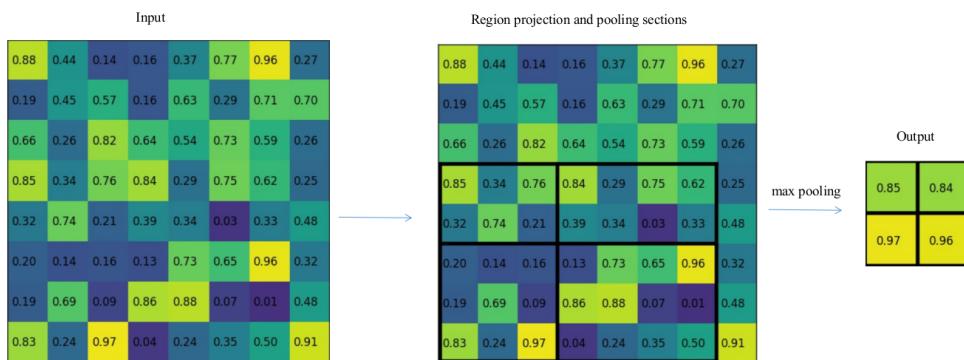


图 3.10 ROIPooling 示意图

3.2.3 Mask 损失函数

为了有效的避免类之间的竞争,使得其他 class 不贡献损失,Mask 的损失函数使用平均二值交叉熵(average binary cross-entropy)。具体的,对于每个 ROIAlign 的 $K \times m^2$ 维输出, K 表示总类别个数, m 对应 mask 分辨率,即输出 K 个 mask, 定

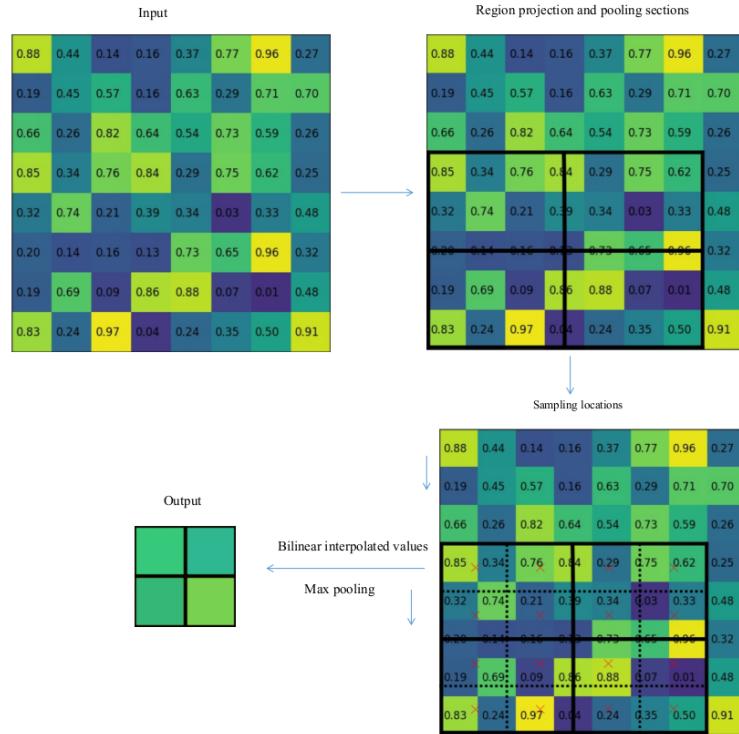


图 3.11 ROIAlign 示意图

义 Mask 的损失函数为

$$L_{mask} = -\frac{1}{K} \sum_{i=1}^K \sum_j (y'_j \lg(y_j) + (1 - y'_j) \lg(1 - y_j)) \quad (3.8)$$

其中

$$j = 1, 2, \dots, m^2 \quad (3.9)$$

表示 mask 的像素索引。通过上述损失函数的定义有效的避免了类间的竞争, 将 mask 分支与 class 分支并行区分开来, 通过 class 分支最终的输出在 K 个 mask 中选择对应的 mask 输出。实验表明, 相比将 mask 和 class 混在一起, 根据 mask 的结果来判断类别的方法, 这种方法对算法最终的精确度有着重要意义。

3.3 目标检测实验

为了评价所设计的 3D Faster R-CNN 和 3d Mask R-CNN 算法的性能, 分别在一个现有的数据集和一个自己采集的实际应用的数据集上进行了网络的训练和测试, 并与原始的 Faster R-CNN 和 Mask R-CNN 相比较, 验证了所设计算法的性能。

3.3.1 数据集

实验所采用的数据集一个是参加 APC(Amazon Picking Challenge) 的 MIT-Princeton 队伍所采集的数据集”Shelf & Tote” Benchmark Dataset(MIT-Princeton 2016), 此处简单记为 APC 数据集, 另外一个数据集是实际用于 bin-picking 在实验室采集的数据集, 记为 workpiece 数据集。

APC 数据集: 该数据集共有 39 类不同的物体, 452 个场景, 每个场景有不同的物体, 一共 7281 组图片, 通过在多个场景下, 不同的视角下使用 Intel Realsense SR300 相机所拍摄。标注的数据是每个场景下物体在世界坐标系下的位姿, 以及每个场景下相机在世界坐标系下的位姿, 是一个半自动标注的数据集, 通过物体的位姿和相机的位姿就可以得到每个物体在相机坐标系下的位姿, 数据集中部分数据如图3.12所示。

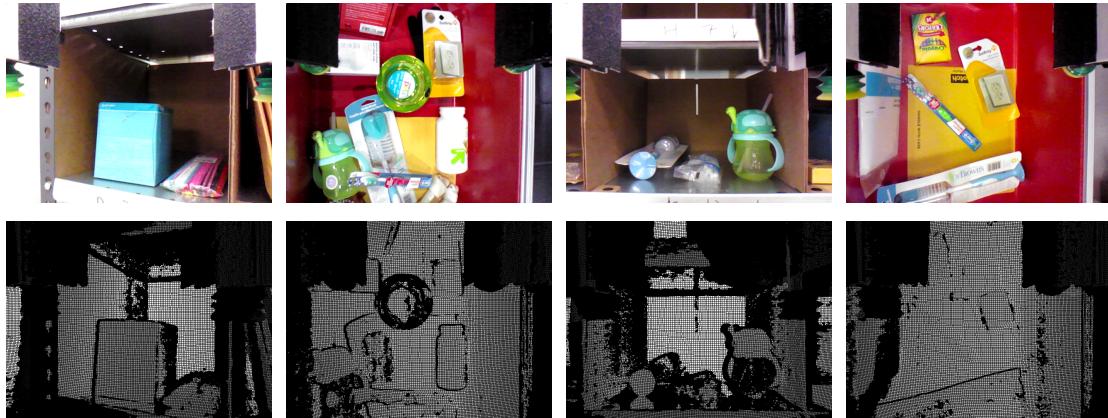


图 3.12 APC 数据集部分数据: 第一栏为彩色图像, 第二栏为与彩色图像相匹配的深度图

APC 数据集中标注的标签可以认为是每个物体在相机坐标系下的位姿和类别, 对于设计的算法来说需要的是物体的类别(class)、边界框(bounding box) 和掩模(mask), 因此需要对原始标注数据进行一些处理, 因为 APC 还提供了每类物体的 CAD 模型, 并且相机的内参矩阵也在数据集中提供了, 因此可以将 CAD 模型转换为点云后齐次变换到所标注的对应物体在相机坐标系下的位姿, 然后利用相机内参矩阵将物体点云投影到图像平面, 从而获得物体的 mask, 进而可以得到物体的 bounding box。需要注意的是由于一个场景中有多个物体, 在不同相机位姿下会出现遮挡, 因此需要对被遮挡物体的 mask 进行相应的裁剪, 对于几乎被完全遮挡的物体可以去除, 判断物体是否遮挡可以通过物体点云距离相机原点的距离远近判断。将一张图中物体位姿得到 mask 和 boudning box 的处理流程如下所示:

1. 对于图中标注的每个物体:
 - 将对应物体的 3D 点云变换到物体标注的位姿

- 根据相机内参矩阵将3D点云投影到图像平面,获得物体的mask以及mask对应的深度图depth
- 遍历像素索引i:
 - 如果在索引i出存在多张mask的值有效,保留depth值最小的mask,将其余mask在索引i处置为无效
 - 对于每个物体的mask:
 - 如果mask中有效像素点小于阈值T,删除该mask
 - 根据mask有效像素点的坐标计算对应的bounding box

处理后的部分图片的ground truth(class, mask, boudning box)如图3.13所示。从



图3.13 APC数据集部分标注数据

图3.13可以看出处理后的mask基本覆盖了物体,boudning box也正确框出了物体,唯一的缺点是所生成的mask有时候有些缺失,没有人工标注的完美,如图3.13中第一张图中的瓶子(easter turtle sippy cup)标注的mask有很多缺失,根本原因是所使用的物体的CAD模型是通过相机采集生成的,其转换的3D点云质量并不是十分理想,其3D点云比较稀疏并且有部分缺失,如图3.14所示,这个瓶子的点云有严重的缺失,主要原因是瓶子透明,所以相机难以采集其深度信息。模型点云的缺失,因此将点云投影到图像平面生成的mask也有部分缺失,尽管已经对生成的mask进行了一些滤波处理,但部分mask还是有明显的缺失。

总体来说,尽管生成的ground truth的质量没有人工标注的ground truth质量

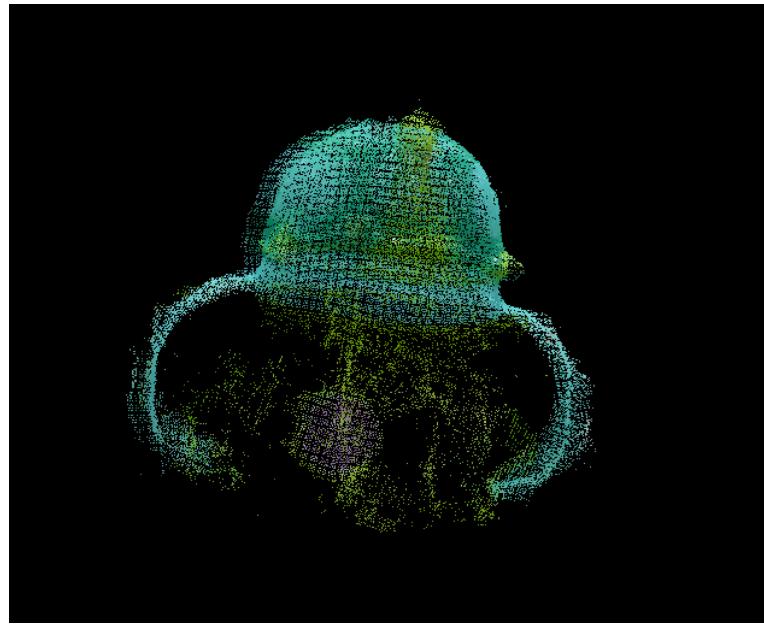


图 3.14 easter turtle sippy cup point cloud

好,但对本实验来说已经够用,并且相比人工标注这种半自动化的标注方式节省了大量时间和金钱成本。

workpiece 数据集: 该数据集有三类物体,共 2k 组图片。该数据集与 APC 数据集最大的不同是,同一张图片中存在大量不同位姿的同种物体,并且三类物体都缺少纹理 (textureless),因此 Faster R-CNN 和 Mask R-CNN 在该数据集上的表现理论上应该大大不如 3D Faster R-CNN 和 3D Mask R-CNN。部分数据集中的图片如图3.15所示。**workpiece** 数据集的 ground truth 由人工标定,其中据测试集中

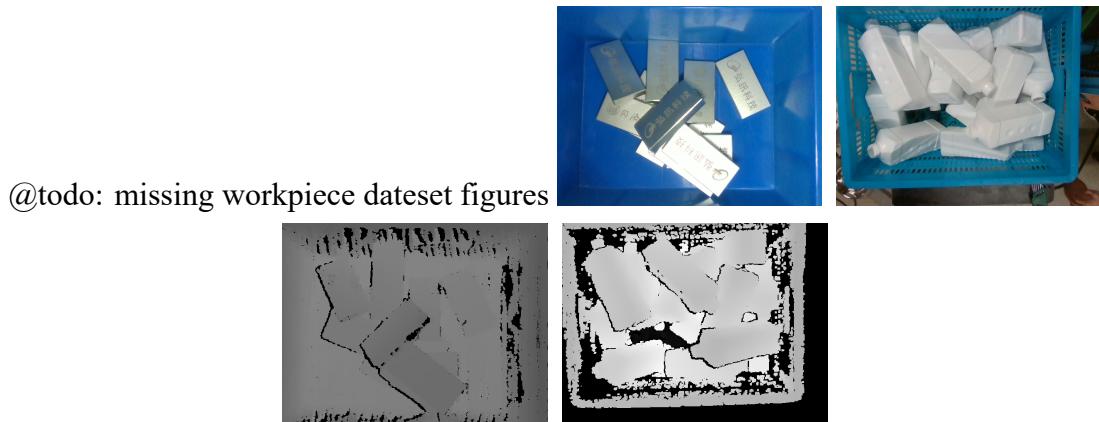


图 3.15 workpiece 数据集部分图片

有的 ground truth 不仅包括了物体的 class, mask, bounding box, 还有物体的位姿,并且由于三类物体都是工厂中的工件,因此也提供三类物体精确的 CAD 模型。

3.3.2 实验内容

实验在 APC 数据集和 workpiece 数据集上比较 Faster R-CNN 和 3D Faster R-CNN、Mask R-CNN 和 3D Mask R-CNN 的性能。

算法实现主要通过 Tensorflow 框架使用 python 语言实现, 详细代码见 Github 项目地址^①。

评价的指标主要是检测的精确度 AP 以及算法的时间性能 FPS。FPS 是每秒能检测的图片数比较好理解, AP 是 bounding box 或者 mask 交并比的精确度。具体地, 如图3.16所示, 两个 bounding box 的交并比定义为:

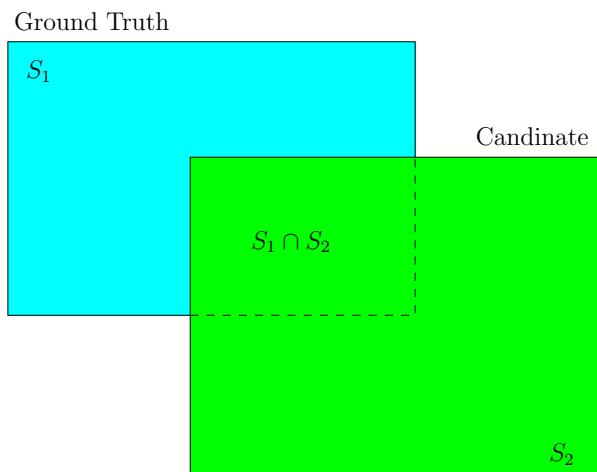


图 3.16 bounding box 交并比

$$IoU = \frac{S_1 \cap S_2}{S_1 \cup S_2} \quad (3.10)$$

$AP_{0.5}$ 表示检测的结果与 ground truth 的交并比大于 0.5 的个数占总体检测个数的比例, 显然定义精度的 IoU 大小会影响最终评价的质量, 过小和过大的最小 IoU 都不能很好地反应算法的精缺度, 因此将评价的主要精确度定义如下:

$$AP = \frac{1}{10} \sum_{i=0}^9 AP_{0.5+0.5i} \quad (3.11)$$

检测结果换为 mask 精确度的定义也类似, 只需用 mask 的交并比代替 bounding box 的交并比。

模型训练在实验室的服务器上进行, 服务器有两块 Intel(R) Xeon(R) E5-2683 v3(2.00GHz) 的 CPU, 4 块 TITAN X GPU。模型训练时为了减少训练时间, 4 块 GPU 都使用了。在 APC 数据集上, 训练用了约 6k 组图片, 剩下的 1k 多组图片用于测试, 3D Faster R-CNN 训练用了 40 个小时左右, 3D Mask R-CNN 用了 48 个小

^① https://github.com/freealong/Mask_RCNN

时左右；在 workpiece 数据集上，训练用了约 1.6k 组图片，剩下的 400 组图片用于测试，3D Faster R-CNN 训练用了 30 个小时左右，3D Mask R-CNN 用了 36 小时左右。

3.3.3 实验结果

在 APC 数据集上，本文算法 Faster R-CNN 和 Mask R-CNN 的精确度如表3.1所示，在测试集上的部分图片检测结果见图??。从表3.1中可以看出在 APC

表 3.1 算法在 APC 数据集上的精确度

	input	output	AP	$AP_{0.5}$	$AP_{0.75}$
Faster R-CNN	RGB	bbox	33.26	56.29	34.03
3D Faster R-CNN	RGB+HHA	bbox	34.55	57.99	34.69
Mask R-CNN	RGB	mask	32.34	55.78	33.12
3D Mask R-CNN	RGB+HHA	mask	33.94	56.45	33.99

@todo: apc figure results

图 3.17 算法 APC 数据集上部分检测结果

数据集上 3D Faster R-CNN 相比 Faster R-CNN 的精确度提高了 1.3 个百分点左右，3D Mask R-CNN 相比 Mask R-CNN 提高了 0.8 个百分点左右。整体来说对精确度的提高并不是十分明显，究其原因，从图3.12可以看到 APC 数据集中的物体大多也是纹理丰富的，单从 RGB 图就可以训练出一个很好的模型，因此增加 HHA 通道，对模型精确度的提升十分有限，反而降低了算法的 FPS。

在 workpiece 数据集上，本文算法 Faster R-CNN 和 Mask R-CNN 的精确度如表3.2所示，在测试集上的部分图片检测结果见图3.18。从表3.2可以看出在

表 3.2 算法在 workpiece 数据集上的精确度

	input	output	AP	$AP_{0.5}$	$AP_{0.75}$
Faster R-CNN	RGB	bbox	18.78	37.49	19.46
3D Faster R-CNN	RGB+HHA	bbox	32.39	56.37	33.54
Mask R-CNN	RGB	mask	16.12	35.95	18.74
3D Mask R-CNN	RGB+HHA	mask	30.98	53.74	32.19

workpiece 数据集上，3D Faster R-CNN 相比 Faster R-CNN 的精确度提高了 13.6 个百分点左右，3D Mask R-CNN 相比 Mask R-CNN 提高了约 14.8 个百分点。显然，无论是 3D Faster R-CNN 还是 3D Mask R-CNN，在 workpiece 数据集上精确度

@todo: workpiece figure results

图3.18 算法workpiece数据集上部分检测结果

相比原算法有了大大的提高,从图3.15可以发现workpiece数据集中的图片包含的都是一些缺少纹理的物体,并且有大量同种物体混杂在一起,有时候人眼也很难从中区分单个目标,因此可能单从RGB图难以训练出一个准确率较高的模型来检测目标。而这些缺少纹理的大量物体在深度图,尤其是变换后的HHA图上十分容易区分出来,因此3D Faster R-CNN和3D Mask R-CNN引入HHA后,增加了更多信息,最终训练得到的模型的准确度相比原算法有了巨大的提升。

3D Faster R-CNN和3D Mask R-CNN算法的时间性能见表3.3,由于两个数据集内图片的大小都是一样的,因此算法的时间性能在两个数据集上并不会有什么差异,因此表3.3中直接统计了算法在两个数据集测试样本上FPS的平均值。从表3.3可以看出3D Faster R-CNN和3D Mask R-CNN相比原算法,普遍具有更低的FPS,因为增加了HHA数据并且增加了STN模块。但考虑到本文算法的具体应用,适当降低的FPS并不会对具体使用造成什么影响。

	Faster R-CNN	3D Faster R-CNN	Mask R-CNN	3D Mask R-CNN
FPS	5.5	3.2	4.1	2.5

表3.3 算法时间性能

3.4 本章小结

本章主要介绍了两个目标检测的算法3D Faster R-CNN和3D Mask R-CNN,
 @todo:detection本章小结

第 4 章 基于 4PCS 的点云匹配算法

本章主要介绍为了估计目标的位姿所提出的一种基于 4PCS 的点云匹配算法 A4PCS-ICP (Angle-fixed-4PCS-ICP), A4PCS-ICP 主要基于全局匹配算法 4PCS(Aiger et al. 2008) 和局部匹配算法 ICP(Besl et al. 1992)。为了详细介绍 A4PCS-ICP 算法, 本章首先从整体上简单介绍了 A4PCS-ICP 算法, 包括算法所要解决的问题的具体数学描述以及相关算法的背景; 然后介绍 A4PCS-ICP 算法的基础 4PCS 算法, 并分析了其不足, 进而提出 A4PCS 算法对其进行改进; 接着介绍与 A4PCS 算法相结合的 ICP 算法, ICP 算法主要用于提高最终点云匹配的精度; 最后进行了点云匹配的实验, 将本文的 A4PCS-ICP 算法与其他几个匹配算法相比较。

4.1 点云匹配算法概述

4.1.1 问题描述

通过第 3 章中的目标检测算法可以得到目标的 bounding box 或者 mask, 根据 bounding box 或者 mask 可以在深度图中提取对应的区域, 从而获得包含目标的点云。所以现在的问题是如何通过目标的点云计算出目标的位姿, 由于可以得到目标的三维模型, 因此将目标的三维模型经过一个刚体变换 T , 使之与目标点云重合, 然后目标的三维模型在相机坐标系下的位姿也是已知并且可调的, 为方便起见将三维模型坐标系与相机坐标系重合, 则目标的位姿便等于三维模型与目标点云之间的齐次变换关系, 即 T 。所以, 要计算目标的位姿, 就要求解目标三维模型与相机采集到的目标点云之间的刚体变换 T , 如图 4.1, 这也是 A4PCS-ICP 主要要解决的问题: 两个点云之间的匹配问题。

相机所采集到的目标点云是一组包含空间三维坐标(x, y, z)以及颜色(r, g, b)的点集^①, 由于此处并不需要颜色信息, 因此对目标点云只保留空间位置信息, 去除颜色信息后的目标点云记为点集 P 。三维模型亦可通过采样得到一组包含空间三维坐标的点集, 记为 Q 。A4PCS-ICP 算法就可以简化为求解一个刚体变换 T 使得点集 P 中的点经过矩阵 T 变换后, 尽可能与点集 Q 重合。更为准确地, A4PCS-ICP 算法就可以简化为求解一个 LCP(Largest Common Pointset)问题:

^① 对点集(point set)与点云(point cloud)不做区分, 都指包含坐标点的集合



图 4.1 位姿估计示意图

LCP 问题: 给定两个点集 P 和 Q , 在给定距离误差 δ 下, 求解点集 P 的最大子集 P' , 使得 $T(P')$ 和点集 Q 之间的距离在合适的距离度量下小于 δ , 其中 T 是一个刚体变换。

4.1.2 背景介绍

LCP 问题并不是一个新的问题, 解决该问题的算法也有很多, 尤其是近些年来, 随着几何扫描相关技术的发展, 如何将多次扫描或者多个设备采集的三维信息统一到一个坐标系下成为研究的热点, 其本质上可以归结为 LCP 问题或其衍生问题, 这些问题是计算机几何学和计算机视觉中的基础问题。

其中一个比较流行的算法是通过使用稳定的局部几何描述子来匹配得到粗略的刚体变换, 然后紧接着使用 ICP 算法迭代获取较为精确的刚体变换 (Li et al. 2005)。这种算法的效果十分取决于所选取的描述子, 通常一般的描述子对传感器噪声都比较敏感, 尤其是一些低精度的传感器, 常用的局部几何特征描述子有 SHOT(Salti et al. 2014)、FFPH(Rusu et al. 2009) 等; 还有一种比较流行的方法是通过几何希哈方法从事先设置好的候选集中来选择合适的刚体变换 (Wolfson et al. 1997); 一些随机算法, 如 RANSAC(Random Sample Consensus)(Bolles et al. 1981) 通常需要足够长的时间才能保证得到合适的解。

上述介绍的一些算法, 有些对噪声的鲁棒性不强, 有些时间复杂度极高, 有些也难以处理部分重叠的情况, 即点集 P 和 Q 之间只有一部分点集是相匹配的, 因此难以实际直接应用到本文所需要解决的问题, 其效果也难以让人满意。对此, 本文基于 4PCS(4-Points Congruent Sets)算法设计了有效解决点云匹配的算法 A4PCS-ICP。

4.2 A4PCS-ICP 算法

4.2.1 算法框架

A4PCS-ICP 算法基于 4PCS，针对 4PCS 的瓶颈，有效地降低了其时间复杂度，然后通过与 ICP 算法配合提高匹配的精度，其整体框架如图 4.2 所示。A4PCS-ICP

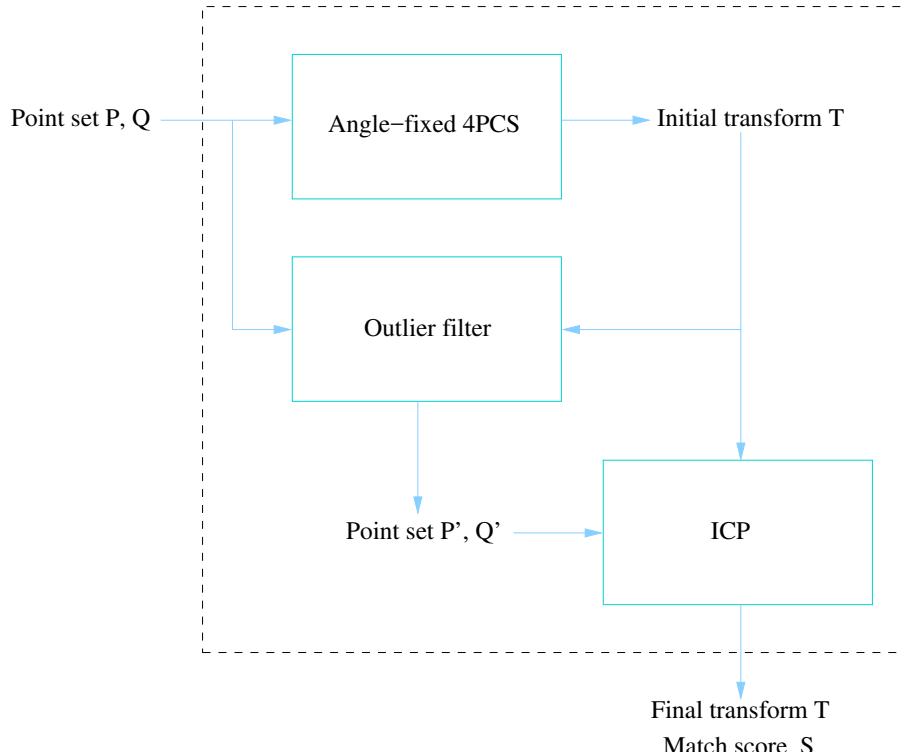


图 4.2 A4PCS-ICP 算法框架

由三个模块组成：Angle-fixed 4PCS、Outlier filter 和 ICP，Angle-fixed 4PCS 是 4PCS 算法的优化版，也是 A4PCS-ICP 算法的核心，根据输入的两个点集，输出两个点集之间的粗略的变换关系；Outlier filter 根据 Angle-fixed 4PCS 的输出对点集 P' 和 Q'' 进行滤波，去掉一些离群点，用以提高下一步 ICP 算法的精度；ICP 模块通过以 Angle-fixed filter 输出的变换关系为初始值对滤波后的两个点集进行迭代求解最佳的刚体变换关系，输出最终的变换关系 T ，也是目标的位姿。

4.2.2 Angle-fixed 4PCS 算法

介绍 Angle-fixed 4PCS 算法之前，首先先详细介绍一下 4PCS 算法，4PCS 算法是一个对 3D 点集全局匹配的算法，即使所给的两个 3D 点集有小的重叠，4PCS 都能给出较好的结果，并且对噪声有一定的鲁棒性。这种方法对初始位姿没有任何要求，其核心是从 3D 点集中提取出所有与给定平面 4-points 近似全等的共面

4-points, 该算法时间复杂度为 $O(n^2 + k)$, 其中 n 是 3D 点集中点的个数, k 是提取出的 4-points 个数。4PCS 使用十分广泛, 并且引申出许多相关的变种 (Corsini et al. 2013)。

4PCS 算法流程: 算法流程如7所示, 该算法输入两个点集 P 和 Q , 还有距离参数 δ , 返回两个点集之间的刚体变换 T 。4PCS 基于以下事实: 共面点集中定义的比例在仿射变换, 包括刚体运动中保持不变。举例来说, 定义点集 $X := \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}\}$, 其中 4 个点不都在同一条直线上, 设直线 ab 和 cd 相交于点 \mathbf{e} , 定义两个比例:

$$\begin{aligned} r_1 &= \|\mathbf{a} - \mathbf{e}\| / \|\mathbf{a} - \mathbf{b}\| \\ r_2 &= \|\mathbf{c} - \mathbf{e}\| / \|\mathbf{c} - \mathbf{d}\| \end{aligned} \quad (4.1)$$

则在仿射变换下所定义的 r_1 和 r_2 均保持不变, 如图4.3。如果曲面 S_1 和 S_2 匹配,

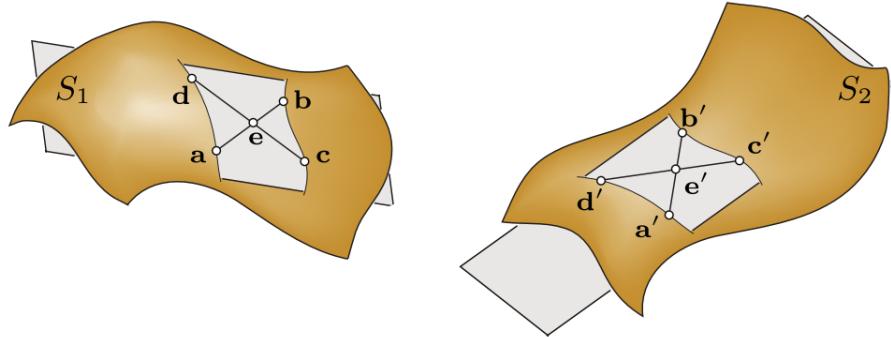


图 4.3 4-points 比例的仿射不变性

并且 4-points 共面基在重叠区域, 则 $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ 对应的四个点 $\mathbf{a}', \mathbf{b}', \mathbf{c}', \mathbf{d}'$ 也共面, 并且

$$\begin{aligned} \|\mathbf{a}' - \mathbf{e}'\| / \|\mathbf{a}' - \mathbf{b}'\| &= r_1 \\ \|\mathbf{c}' - \mathbf{e}'\| / \|\mathbf{c}' - \mathbf{d}'\| &= r_2 \end{aligned} \quad (4.2)$$

4PCS 算法另一个关键技术是使用了宽基 (*wide-base*), 相比于一般的基, 宽基中基的长度更长, 如图4.4所示, 图片上半部分是使用宽基匹配的曲线, 图片下半部分是使用一般的基匹配的曲线, 显然, 通过比较可以发现宽基相比普通基有更稳定的匹配结果, 相关理论证明见文献 (Goodrich et al. 1994)。

回到 4PCS 算法具体实现, 算法的主体其实是一个 RANSAC 循环, 每次循环首先会从点集 P 中挑选共面的宽基 B , 具体实现时, 先从点集 P 中随机选取 3 个点, 然后在剩下的点中选取第四个点构成共面的四点, 第四个点的选取尽可能使得每个点之间的距离最大(因为我们要使用宽基), 并且与前 3 个点近似共面(显然由于噪声的存在, 完全共面并不现实), 但如果第四个点选取的过远也会出现问

算法 7: 4PCS 算法

Input: Point sets P and Q , measure level δ **Output:** Rigid transform T

```

1  $h \leftarrow 0;$ 
2 for  $i = 1$  to  $L$  do
3    $B \leftarrow \text{SELECTCOPLANARBASE}(P);$ 
4    $U \leftarrow \text{FINDCONGRUENT}(B, Q, \delta);$ 
5   forall 4-points coplannar sets  $U_i \in U$  do
6      $T_i \leftarrow$  best rigid transform that aligns  $B$  to  $U_i$  in the least square sense;
7     Find  $S_i \subseteq P$ , such that  $d(T_i(S_i), Q) \leq \delta$ ;
8      $k \leftarrow \arg \max_i \{|S_i|\};$ 
9     if  $|S_k| > h$  then
10        $h \leftarrow |S_k|;$ 
11        $T \leftarrow T_k;$ 
12   return  $T;$ 

```

```

13 def FINDCONGRUENT( $B \equiv \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4\}, Q, \delta$ ):
14    $d_1 \leftarrow \|\mathbf{b}_1 - \mathbf{b}_2\|;$ 
15    $d_2 \leftarrow \|\mathbf{b}_3 - \mathbf{b}_4\|;$ 
16   计算  $R_1 \equiv \{(\mathbf{p}_i, \mathbf{p}_j) \mid \mathbf{p}_i, \mathbf{p}_j \in Q\}$ , 使得  $\|\mathbf{p}_i - \mathbf{p}_j\| \in [d_1 - \delta, d_1 + \delta]$ ;
17   计算  $R_2 \equiv \{(\mathbf{p}_i, \mathbf{p}_j) \mid \mathbf{p}_i, \mathbf{p}_j \in Q\}$ , 使得  $\|\mathbf{p}_i - \mathbf{p}_j\| \in [d_2 - \delta, d_2 + \delta]$ ;
18   forall  $r_{1i} \in R_1$  do
19     计算与定量  $r_1$  和  $r_2$  相关的四个点  $\{\mathbf{e}_{1i}^1, \mathbf{e}_{1i}^2, \mathbf{e}_{1i}^3, \mathbf{e}_{1i}^4\}$ , 记  $\Pi(\mathbf{e}_{1i}^j) = r_{1i}$ ;
20     对点集  $\{\mathbf{e}_{1i}^j\}$  在  $\mathbb{R}^3$  空间建立 range tree 的数据结构;
21   forall  $r_{2i} \in R_1$  do
22     计算与定量  $r_1$  和  $r_2$  相关的四个点  $\{\mathbf{e}_{2i}^1, \mathbf{e}_{2i}^2, \mathbf{e}_{2i}^3, \mathbf{e}_{2i}^4\}$ , 记  $\Pi(\mathbf{e}_{2i}^j) = r_{2i}$ ;
23    $U' \leftarrow \emptyset;$ 
24   forall  $\mathbf{e}_{2i}^j$  do
25     在 range tree 中以  $\delta$  为领域检索点  $\mathbf{e}_{2i}^j$  附近的点, 对于每个检索到的
      点  $\mathbf{q}$ , 建立与  $B$  相对应的 4 个点的点集  $U' \leftarrow \{U', (\Pi(\mathbf{q}), \Pi(\mathbf{e}_{2i}^j))\}$ ;
26   return  $U'$ ;

```

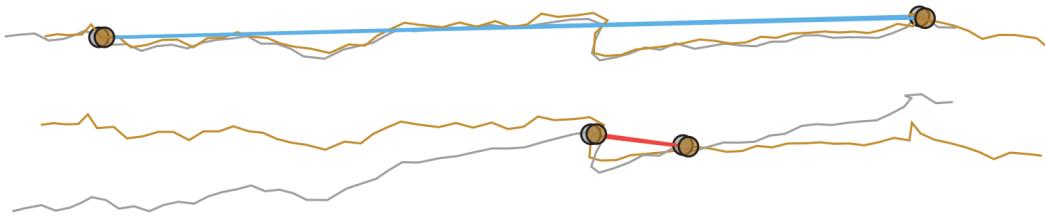


图 4.4 宽基的匹配稳定性

题,因为如果宽基超过两个点集的重叠区域则难以匹配,因此当选以最大距离取宽基造成误差变大时以 $f = 1, 0.5, 0.25, \dots$ 的比率降低最大距离来选取宽基。

在点集 P 中选取好宽基 B 后, 算法下一步会在点集 Q 中通过 4-points 的仿射不变性找出所有与宽基 B “全等”的基, 构成集合 U 。在 Q 中选取基的方法见算法7中的 FINDCONGRUENT 函数, 函数首先使用基 B 中的点先定义两个仿射无关的比例, 如图4.5中左边的图所示。假设在点集 Q 中找到两点 \mathbf{q}_1 和 \mathbf{q}_2 , 并且

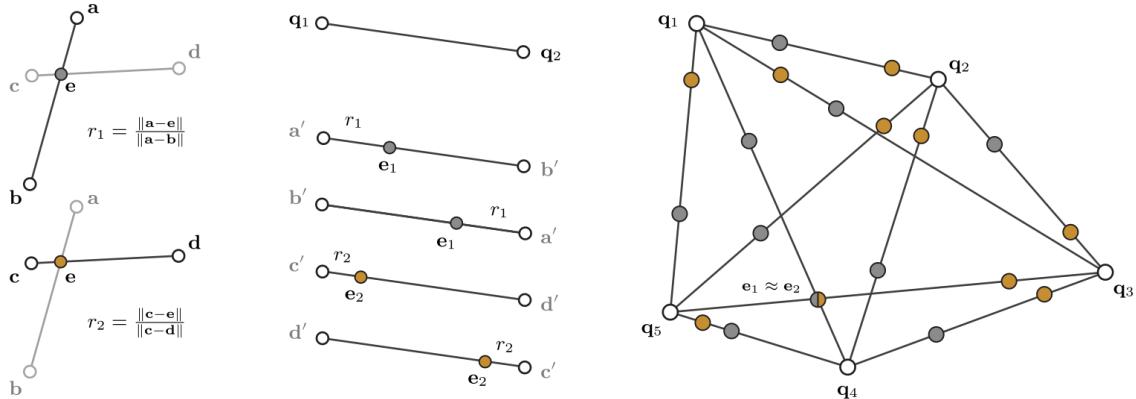


图 4.5 寻找近似“全等”的基示意图

$\|\mathbf{q}_1 - \mathbf{q}_2\| - \|\mathbf{a} - \mathbf{b}\| \leq \delta$, 则点 $\mathbf{q}_1, \mathbf{q}_2$ 有可能与点 \mathbf{a}, \mathbf{b} 对应, 则直线 \mathbf{ab} 与 \mathbf{cd} 相交的点 \mathbf{e} 的对应点可能为

$$\mathbf{e}_1 = \mathbf{q}_1 + r_1(\mathbf{q}_2 - \mathbf{q}_1) \quad (4.3)$$

或者

$$\mathbf{e}_1 = \mathbf{q}_2 + r_1(\mathbf{q}_1 - \mathbf{q}_2) \quad (4.4)$$

同理也可以根据 \mathbf{c}, \mathbf{d} 的对应点(设为 $\mathbf{q}_3, \mathbf{q}_4$)求得 \mathbf{e} 的对应点

$$\mathbf{e}_2 = \mathbf{q}_3 + r_1(\mathbf{q}_4 - \mathbf{q}_3) \quad (4.5)$$

或者

$$\mathbf{e}_2 = \mathbf{q}_4 + r_1(\mathbf{q}_3 - \mathbf{q}_4) \quad (4.6)$$

则当 $\mathbf{e}_1 \approx \mathbf{e}_2$ 时, $\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4$ 就是我们所要找的一组与点 $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ 近似“全等”的基, 如图4.5中右边图中的 $\mathbf{q}_5, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_1$ 。

具体实现时, 当我们在点集 Q 中找出了所有可能的 \mathbf{e}_1 和 \mathbf{e}_2 后, 找出其中近似相等的 \mathbf{e}_1 和 \mathbf{e}_2 可以通过 range 树 (Arya et al. 1998) 来实现, 对于大小为 n 的点集, range 树的建立时间复杂度为 $O(n \lg n)$, 查询附近点的时间复杂度为 $O(\lg n + k)$, 其中 k 是查询到点的个数。

在 Q 中找出所有与基 B 近似“全等”的基后, 下一步就是计算出最优的刚体变换 T 。对于 U 中的每个基 U_i , 我们可以利用最小二乘 (Horn 1987) 的思想计算 B 到 U_i 的刚体变换 T_i 。得到刚体变换 T_i 后, 我们将点集 P 进行变换 T_i , 然后对变换后的点集中的点在 Q 中查找最近点, 统计最近点距离小于 δ 的个数 S_i , S_i 也是评价 T_i 效果的分数, 分数越高的 T_i 就是我们要求的最优刚体变换 T 。

4PCS 算法时间复杂度: 设输入的两个点集 P, Q 的大小分别为 m, n 。算法中最耗时的部分是 FINDCONGRUENT 函数: 从点集 Q 中选取距离为 d_1 和 d_2 的点对, 其时间复杂度为 $O(n^2)$, 然后建立和查询 range 树, 其复杂度为 $O(n^2 + k)$, 其中 k 是满足条件的基个数, 因此 4PCS 算法整体的时间复杂度为 $O(n^2 + k)$, 空间复杂度显然为 $O(n)$ 。

4PCS 算法不足: 仔细研究 4PCS 算法, 可以发现从点集 Q 中提取的基与 B 并不是全等的, 如图4.6所示, 将线段 $\mathbf{q}_1\mathbf{q}_2$ 绕交点转动一定角度后便不再与原基全

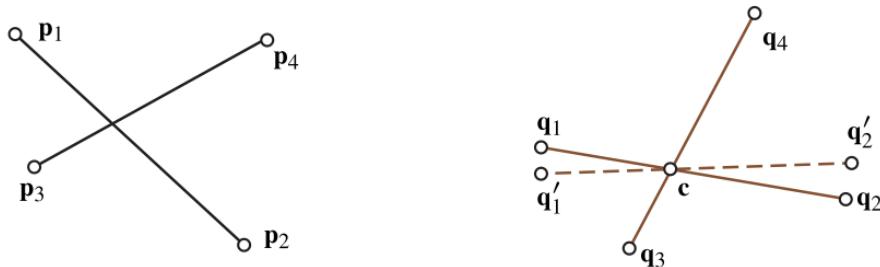


图 4.6 4PCS 中“全等”的基

等, 但是 4PCS 仍然会找出点 $\mathbf{q}'_1, \mathbf{q}'_2, \mathbf{q}_3, \mathbf{q}_4$ 作为与 $\mathbf{p}'_1, \mathbf{p}'_2, \mathbf{p}_3, \mathbf{p}_4$ 全等的基。这一缺点会导致 4PCS 算法需要更多的求解时间, 并且还有可能影响最终的匹配结果。

@todo: 4pcs 算法改进

4.2.3 Outlier filter

从图 4.1其实可以看出目标点云有许多点并不属于目标物体, 尤其是检测结果没有 mask 只能使用 bbox 分割目标时, 不属于目标物体的离群点就尤其的多。因此, 为了提高最终的匹配精度, 除去这些离群点十分有必要, Outlier filter 模块

的作用就是根据 Angle-fixed 4PCS 输出的初始刚体变换来去除点云数据中的离群点,从而提升下一步 ICP 算法匹配的精度,使最终估计的位姿精度提高。

Outlier filter 模块的核心算法如算法8所示。算法输入点集 P 和 Q ,需要参数

算法 8: Outlier filter 算法

Input: Point sets P and Q , Initial transform T , Distance tolerance δ

Output: Point sets P' and Q'

```

1  $P' \leftarrow P;$ 
2  $Q' \leftarrow \emptyset;$ 
3 forall point  $p_i \in P$  do
4    $p_i \leftarrow T(p_i);$ 
5 对点集  $P$  在  $\mathbb{R}^3$  空间建立 kd 树的数据结构;
6 forall point  $q_i \in Q$  do
7   在 kd 树中检索出距离点  $q_i$  最近的点  $p$ ;
8    $d \leftarrow \| q_i - p \|;$ 
9   if  $d \leq \delta$  then
10     $Q' \leftarrow \{Q', q_i\};$ 
11 return  $P', Q';$ 

```

初始刚体变换 T ,以及允许的距离误差 δ ,由于点集 P 是由物体的 CAD 模型转换过来的,因此不对其进行滤波,只对点集 Q 进行离群点去除。具体方法是,首先使用 T 对点集 P 进行刚体变换;然后对变换后的点集建立 kd 树,建立 kd 树的目的是为了快速在点集 P 中找到距离某点最近的点,其查找的时间复杂度为 $O(kN^{1-1/k})$,其中 k 是所建立 kd 树的维数,显然对于三维空间中点集为 3, N 是建立的 kd 树的节点个数;建立好 kd 树后,对点集 Q 中的每个点在 kd 树中找到与之距离最近的点,如果两点间的距离大于所设的参数 δ ,则在点集 Q 中去除该点。实际运行 Outlier filter 算法的效果如图4.7所示。

4.2.4 ICP 算法

ICP(Iterative Closest Point)算法,即最近点迭代算法,是最为经典的数据配准算法。ICP 算法本质上是基于最小二乘法的最优配准方法。该算法重复进行选择对应关系点对,计算最优刚体变换,直到满足正确配准的收敛精度要求。由于 ICP 算法是一种迭代算法,因此只要时间允许便可以获取足够精度的解,但也正因为如此,ICP 也容易陷入局部最优解。本文充分考虑了 ICP 算法的这个特点,通过

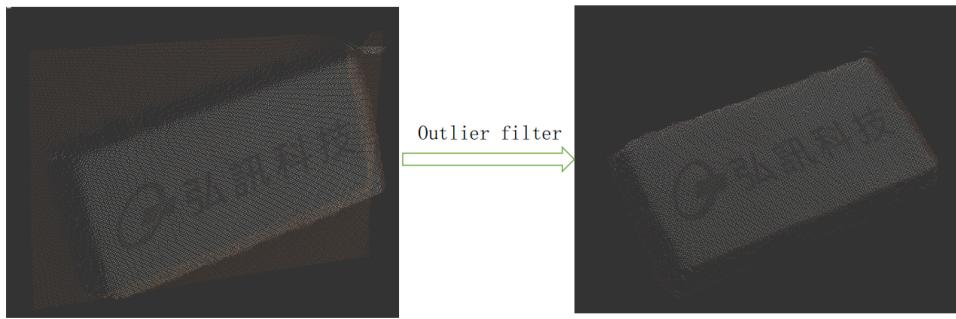


图 4.7 Outlier filter 效果图

Angle-fixed 4PCS 算法给出初始的刚体变换来避免 ICP 算法陷入局部最优解，同时通过迭代来提高最后输出的刚体变换精度。下面介绍一下 ICP 算法的基本原理。

给定两个点集 $P_n := \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n\}$ 和 $Q_m := \{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_m\}$ ，以及初始旋转变换 R 和平移变换 t ，以及迭代结束额距离误差 δ ，ICP 算法迭代步骤如下：

- 步骤 1：根据当前 R 和 t ，对于点集 P_n 中的每个点在 Q_m 中找出距离最近的点，构成点集 Q_n ；
- 步骤 2：计算 P_n 和 Q_n 之间的距离的均方根误差：

$$E(R, t) = \frac{1}{n} \sum_{i=1}^n \| \mathbf{q}_i - R\mathbf{p}_i - t \| \quad (4.7)$$

通过奇异值分解求得使得 $E(R, t)$ 最小的 R' 和 t' ；

- 步骤 3：如果 $E(R, t) < \delta$ ，结束迭代；否则 $R \leftarrow R'$, $t \leftarrow t'$ ，跳转至步骤 1。

ICP 算法的迭代过程还是相对来说十分简单的，唯一需要思考一下的是如何求得最小化 $E(R, t)$ 的 R' 和 t' ，通过奇异值分解求解 R' 和 t' 的方法如下：

首先，记

$$\begin{cases} P'_n &= \{\mathbf{p}_i - \mu_p \mid \forall \mathbf{p}_i \in P_n\} := \{\mathbf{p}'_i\} \\ Q'_n &= \{\mathbf{q}_i - \mu_q \mid \forall \mathbf{q}_i \in Q_n\} := \{\mathbf{q}'_i\} \end{cases} \quad (4.8)$$

其中

$$\begin{cases} \mu_p &= \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i \\ \mu_q &= \frac{1}{n} \sum_{i=1}^n \mathbf{q}_i \end{cases} \quad (4.9)$$

再另

$$W = \sum_{i=1}^n \mathbf{p}'_i \mathbf{q}'_i \quad (4.10)$$

然后对矩阵 W 进行奇异值分解:

$$W = U\Sigma V^T \quad (4.11)$$

则

$$\begin{cases} R' &= UV^T \\ t' &= \mu_p - R'\mu_q \end{cases} \quad (4.12)$$

4.3 点云匹配实验

为了评价所设计的算法 A4PCS-ICP, 考察其匹配精度以及时间性能, 本文设计了位姿估计的实验, 不但考察了 A4PCS-ICP 的性能, 还与其他一些算法作比较, 验证了 A4PCS-ICP 算法的匹配的精确度。

@todo: 点云匹配实验

4.4 本章小结

@todo: matcher 小结

第 5 章 3D 目标位姿估计算法

本章提出了一种 3D 目标位姿估计算法 3D-MRAI (3D Mask R-CNN & A4PCS-ICP)，该算法根据所提供目标的 CAD 模型，可以在 RGB-D 图中检测出目标，并给出目标的位姿。3D-ODPE 算法主要基于第 3 章中基于 RGB-D 图的目标检测算法 3D Faster R-CNN/3D Mask R-CNN，以及第 4 章中的点云匹配算法 A4PCS-ICP，通过将这两个算法结合，分两步计算出 RGB-D 图中目标的位姿。为了评价 3D-MRAI 算法的性能，本章还设计了相关实验，并与同类算法做了比较。

5.1 3D-MRAI 框架设计

3D-MRAI 主要解决三维空间中的目标检测和位姿估计问题，根据 RGB-D 图像，给出图像中目标的种类和其在三维空间中的位姿，区别与常见的在 2D 图像中的目标检测（如第 3 章中的算法）给出的是目标的种类和其在图像坐标中的 bounding box 或者 mask。给出目标在三维空间中的位姿的意义十分巨大，尤其在机器人领域中，图像层面的结果往往难以满足要求，但难度也很大。

一些给出 3D 目标位姿的传统算法，如 3DMatch(Zeng et al. 2016)，3DMatch 通过匹配局部几何特征来计算目标的位姿，缺点是对采集的 3d 数据质量要求很高，往往需要使用激光采集，因此整个识别过程的时间很久；通过 SIFT 描述子来匹配目标位姿 (Dias et al. 2015) 也是一种方法，但其对纹理较少的物体往往难以匹配，效果很差；另外如 LINEMOD(Hinterstoisser et al. 2012) 和 MOPED(Collet et al. 2011) 这些位姿估计框架，在某些情况下如目标在平整的桌面上并且光照条件较好的情况下才能取得满意的效果。因此，急需一种鲁棒性较强，精度较高，计算时间较短的 3D 目标位姿估计算法。

3D-MRAI 的框架如图 5.1 所示，从图 5.1 可以看出算法的输入是 RGB 图像、深度图，以及目标物体的 CAD 模型，输出是图像中检测到的目标的位姿。3D-MRAI 的核心部分是 3D Faster/Mask R-CNN 和 A4PCS-ICP 算法，3D Faster/Mask R-CNN 以及在第 3 章详细介绍过，A4PCS-ICP 也在第 4 章详细介绍过了。因此，3D-MRAI 估计目标的位姿流程上也是分为两步 (two-stage)：

- **目标检测：**利用 3D Faster/Mask R-CNN 检测目标，得到目标 BBox 或者 Mask
- **点云匹配：**将 CAD 模型与目标检测对应的点云进行匹配，得到目标位姿

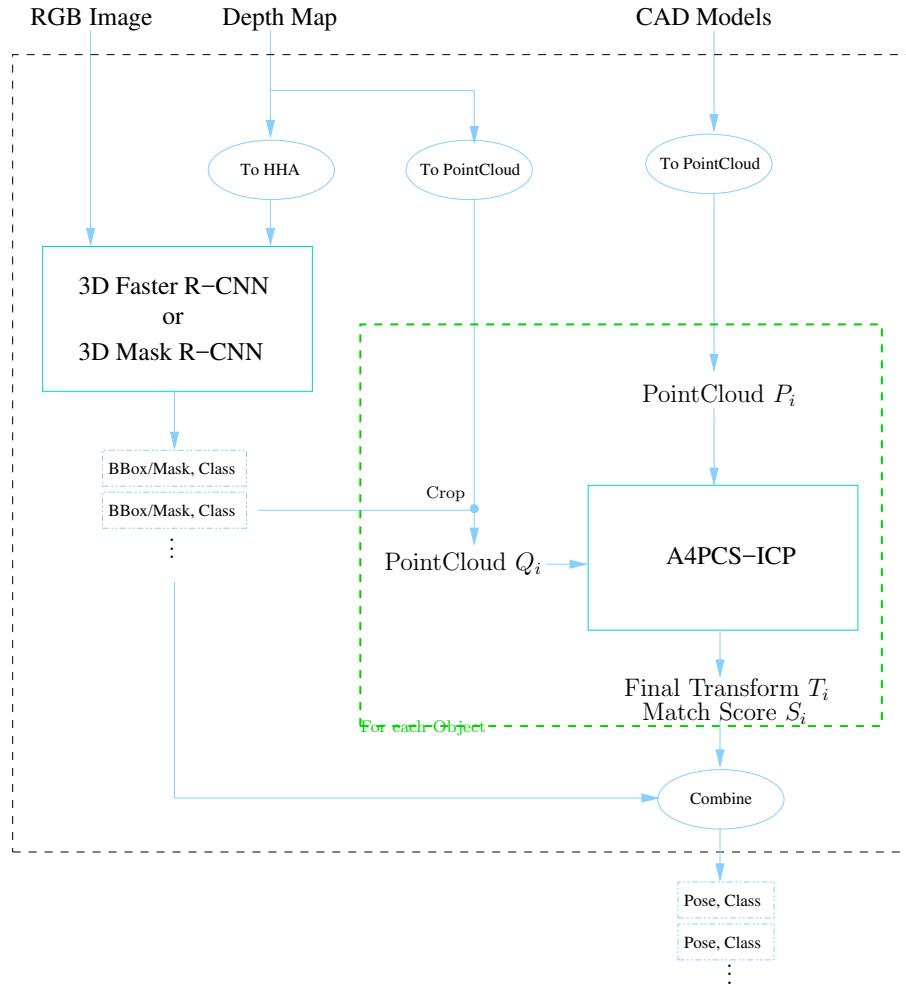


图 5.1 3D-MRAI 算法框架

5.2 3D-MRAI 具体实现

对与 3D-MRAI 输入的 RGB 图像和深度图,可以由对偶 RGB-D 获得,具体见第 2 章。目标物体的 CAD 模型也容易获得,对于工厂中的工件,往往是有其 CAD 模型的,对于一般物体,可以通过 3D 扫描仪重构出来,当然也可以使用本文设计的对偶 RGB-D 相机采集重构出来。

获得目标物体的 CAD 模型后,为了方便后续需处理,我们需要将其转换为点云。具体如何转换的话,基本思想是参考 Uniform Sampling 算法,Uniform Sampling 算法的核心思想是以 3D 栅格中所有点的质心代替这些点,从而达到降采样。类似地,对于 CAD 模型也建立 3D 栅格,但由于无法获得 3D 栅格总所有点,因此判断 CAD 模型是否穿过 3D 栅格,如果穿过 3D 栅格,则在该 3D 栅格中心处增加一个点。显然 3D 栅格的边长越大,转换后的点云数量越小,精度越低,考虑到所使用相机生成点云的精度,因使 CAD 模型转换后的点云的精度与相机采集的点云的精度近似,实际取 3D 栅格边长为 1mm,一个实际工件的 CAD 模型

和以 1mm 为边长进行采样转换后的模型点云如图 ?? 所示。此外,还需要将深度

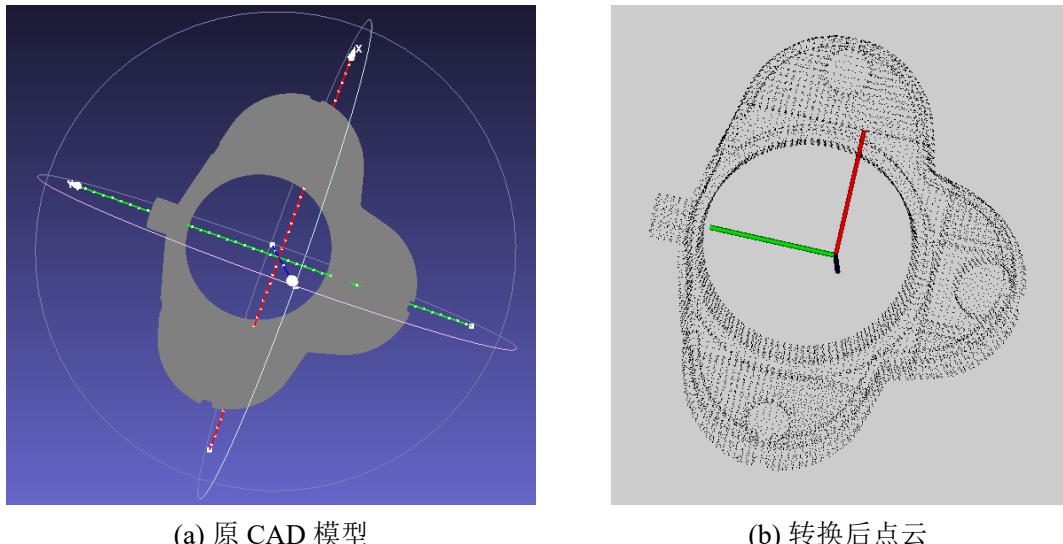


图 5.2 CAD 模型和转换后的点云

图转换为点云,这一步则相对简单,只要通过深度摄像头的内参矩阵反投影到三维空间即可,详细见第 2 章。

对于 3D Faster/Mask R-CNN 的输入,还需要将深度图转换为 HHA,具体见 3.1.2 小节。3D Faster/Mask R-CNN 模块的实现,由于一个深度神经网络,只要训练好后将网络模型导出成 Tensorflow 的 pb 文件,然后此处导入该网络模型,给定输入,网络输出便是图片中目标的 BBox/Mask 和 Class。

得到目标物体的 BBox/Mask 后,需要从深度图对应的点云中抠出目标点云,由于深度图转换的点云是有序的,因此 BBox/Mask 在深度图中的索引坐标与深度图转换的点云的索引坐标是一致的,只要将点云中对应的点提取出来就行,并滤去无效的点然后适当降采样即可,尽管滤波和降采样之后的目标点云是无序的,但后续匹配算法并不需要输入点云有序,而且降采样后点云数量减少,将会减少后续匹配算法的时间。

裁剪得到目标物体的点云 Q_i 后,找出对应物体的 CAD 模型转换的点云 P_i ,将 P_i 和 Q_i 输入到 A4PCS-ICP 模型,即可得到 CAD 模型到目标点云之间的刚体变换 T_i ,由于 CAD 模型坐标系与相机坐标系重叠,因此将矩阵 T 转换为 X, Y, Z, r, p, y 就是目标点云在相机坐标系下的位姿 Pose,最后将所有匹配得到的结果与目标检测的结果组合,并滤去匹配或检测分数较低的结果。详细的算法流程如算法9所示。

算法 9: 3D-MRAI 算法

Input: RGB Image I , Depth Map D , CAD Models M **Output:** Set of Pose and Class Res

```

1  $Res \leftarrow \emptyset;$ 
2  $P \leftarrow \emptyset;$ 
3 forall  $M_i \in M$  do
4    $P \leftarrow \{P, CAD2PointCloud(M_i)\};$ 
5  $H = Depth2HHA(D);$ 
6  $Q = Depth2PointCloud(D);$ 
7  $Mask, Class \leftarrow 3DMAS\ KRCNN(I, H);$            // Same with
     3DFASTERCNN
8 forall  $m_i \in Mask, c_i \in Class$  do
9    $Q_i \leftarrow Crop(Q, m_i);$ 
10   $P_i \leftarrow P(c_i);$ 
11   $T_i, S_i \leftarrow A4PCSICP(P_i, Q_i);$ 
12  if  $S_i > S_{min}$  then
13     $Res \leftarrow \{Res, [T_i, c_i]\};$ 
14 return  $Res$ 

```

5.3 3D 目标位姿估计实验

为了评价所提出的 3D-MRAI 的性能, 设计了 3D 目标位姿估计的实验, 并与文献 (Hinterstoisser et al. 2012) 所提出的基于 LINEMOD 算法的 3D 目标位姿估计框架相比。

5.3.1 数据集

实验所使用的数据集是 workpiece 数据集, 在第3.3.1小节中已经部分介绍过了, 该数据集是在实验室采集的三类物体, 第 3 章中实验所用 workpiece 数据集中的 ground truth 是物体的种类、BBox 和 Mask, workpiece 数据集中测试集中的图片的 ground truth 除了物体的种类、BBox 和 Mask, 还有物体的位姿, 物体的位姿是通过在物体旁固定标定板采集的。具体方法是, 通过固定标定板在目标物体旁, 我们可以记录标定板到目标的刚体变换关系 T_1 , 然后我们通过彩色摄像头可以

检测出标定板的位姿 T_2 , 则物体的位姿可以通过下式得到

$$T = T_2 T_1 \quad (5.1)$$

5.3.2 实验内容

为了有效的评价 3D-MRAI 算法, 我们先定义一个合适的评价指标姿态误差:

$$m = \text{avg}_{\mathbf{x} \in M} \|(\mathbf{R}\mathbf{x} + \mathbf{t}) - (\tilde{\mathbf{R}}\mathbf{x} + \tilde{\mathbf{t}})\| \quad (5.2)$$

其中 M 表示算法运行结果得到的物体种类对应的 CAD 模型转换得到的点云, R 和 t 分别表示从 ground truth 物体位姿分解得到的旋转变换和平移变换, \tilde{R} 和 \tilde{t} 分别表示从算法运行结果得到的物体位姿分解得到的旋转变换和平移变换。显然, 如果算法运行结果和 ground truth 越接近, 所定义的姿态误差就越小。对于一些对称的物体(如圆柱体的被子), 显然不同角度下相机看到的目标物体可能近似, 会造成算法运行的结果正确的情况下与 ground truth 相差很大, 造成姿态误差很大, 与我们所定义的评价指标的宗旨相违背。因此, 针对一些对称的物体, 重新定义姿态误差为

$$m = \text{avg}_{\mathbf{x}_1 \in M} \min_{\mathbf{x}_2 \in M} \|(\mathbf{R}\mathbf{x}_1 + \mathbf{t}) - (\tilde{\mathbf{R}}\mathbf{x}_2 + \mathbf{t})\| \quad (5.3)$$

此外, 如果 $k_m d > m$, 我们就认为目标物体准确检测到了, 并且估计的位姿正确, 其中 d 是目标物体对应模型的直径, k_m 是系数。因此, 还可以定义一个正确检测目标并正确估计目标位姿的准确率。

实验在 workpiece 数据集的测试集上分别运行了 3D-MRAI 和文献 (Hinterstoisser et al. 2012) 中的基于 LINEMOD 算法的检测框架, 运行实验的计算机有两块 Intel(R) Xeon(R) E5-2683 v3(2.00GHz) 的 CPU, 4 块 TITAN X GPU, 由于 3D-MRAI 有深度神经网络所以使用了一块 GPU 和一块 CPU, Hinterstorisser 等人的算法不需要 GPU, 只使用了一块 CPU。

5.3.3 实验结果

@TODO: exp detect results figures 分别统计 3D-MRAI 和 LINEMOD 在测试集上的运行结果, 变化系数 k_m 统计算法的准确率如表5.2所示。表中 k_m 从 5% 变化到 15%, 表示物体直径的占比, k_m 越大, 允许的姿态误差就越大, 因此准确率就越高。实际实验时, 发现 $k_m \approx 10\%$ 时基本上肉眼可以看出匹配的姿态误差。将表5.2绘制成图如5.3所示, 从图中可以发现本文所提出的 3D-MRAI 算法的准确率

$k_m [\%]$	5	7	9	11	13	15
Hinterstoisser et al.	75.63	83.84	89.13	93.48	96.83	98.12
3D-MRAI	95.12	97.35	98.10	98.69	99.22	100.00

表 5.1 3D-MRAI 和 Hinterstoisser 等人的算法准确率

大大超过了 Hinterstoisser 等人的算法, 在 $k_m = 13\%$ 是 3D-MRAI 算法的准确率已经接近 100% 了, 以肉眼可以看出匹配的姿态误差 $k_m = 9\%$ 为标准时 3D-MRAI 算法的准确达到了 98.10%, 比 Hinterstoisser 等人提出的算法高了大约 9 个百分点。

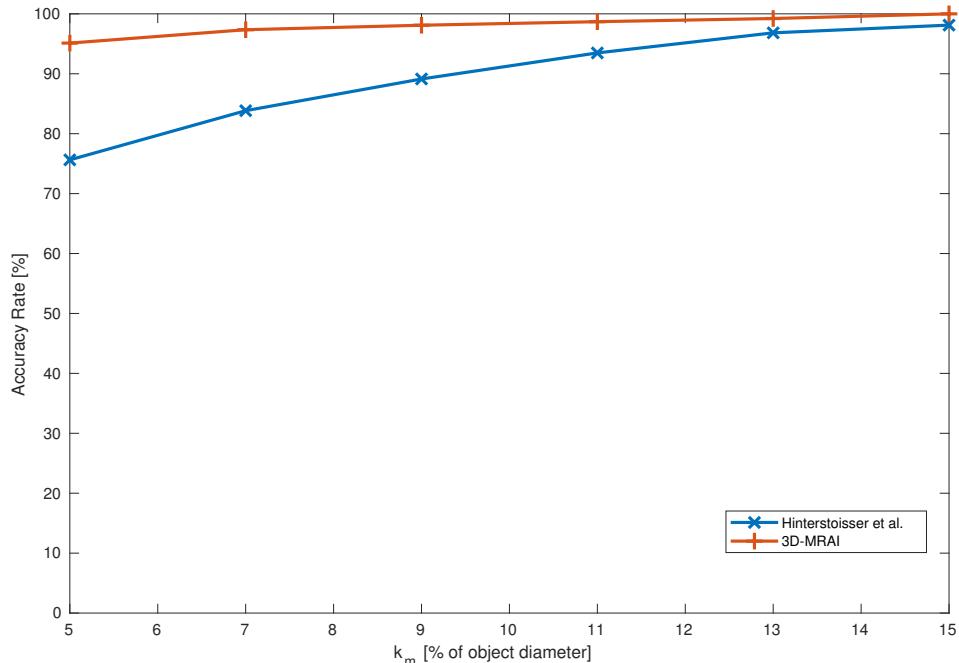


图 5.3 3D-MRAI 和 Hinterstoisser 等人的算法准确率曲线

为了进一步观察算法给出位姿的精度, 取 $k_m = 9\%$ 时 3D-MRAI 准确检测的例子, 将目标物体的位姿转换为 X, Y, Z, r, p, y 六个直观的变量三维位置和姿态欧拉角, 然后于 ground truth 相比较, 得到算法结果在距离和角度上的误差的频率直方图如图5.4所示。从图中可以看出 3D-MRAI 正确检测和估计目标位姿的情况下, 在 X, Y, Z 方向下的位置误差大部分分布在 $0 \sim 1mm$ 之间, 其距离精度在 $1mm$ 左右; 在 r, p, y 三个角度下的角度误差也大部分分布在 $0 \sim 1deg$ 之间, 其角度精度在 $1deg$ 左右。统计图5.4中的数据, 可以算出距离误差和角度误差的均值和

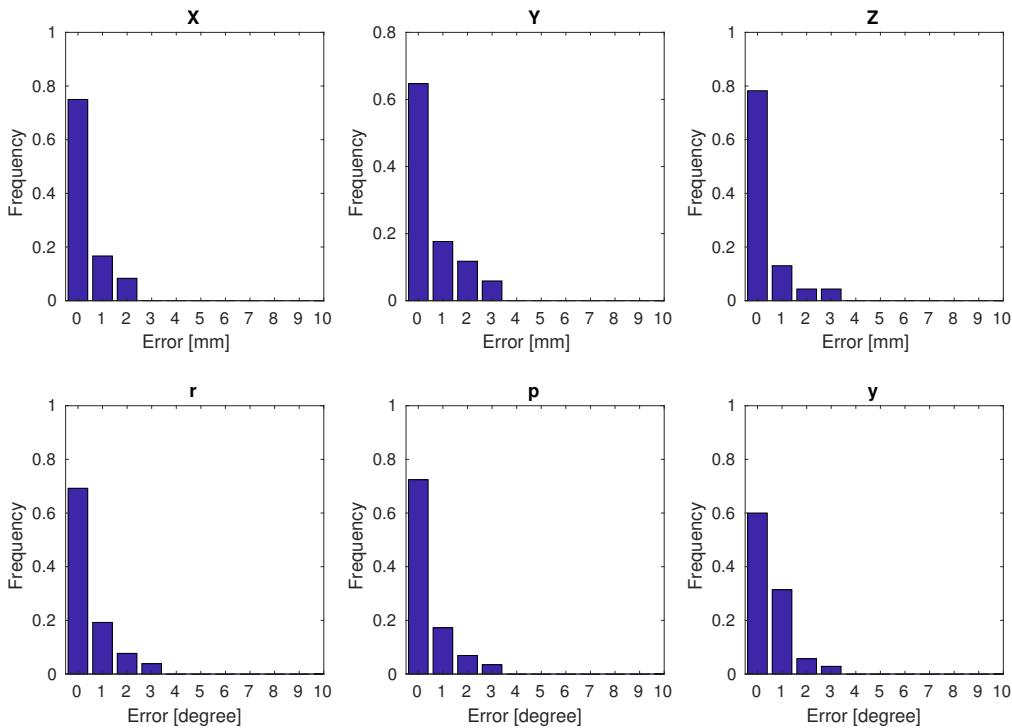


图 5.4 3D-MRAI 算法精度

方差为:

$$\begin{aligned} e_d &= 0.82 \pm 0.21\text{mm} \\ e_a &= 0.91 \pm 0.29\text{deg} \end{aligned} \quad (5.4)$$

除了关心算法的准确率和精度, 算法的运行时间也是我们所关心的。在实验所用的计算机上, 本文所设计的 3D-MRAI 算法与 Hinterstoisser 等人设计的基于 LINEMOD 算法的平均帧率如表??所示。表中可以看出 3D-MRAI 的 FPS 相比

Hinterstoisser et al. 3D-MRAI		
FPS	7.6	2.2

表 5.2 3D-MRAI 和 Hinterstoisser 等人的算法准确率

Hinterstoisser 等人的算法的 FPS 相对较低, 由于使用了深度神经网络相关的算法, 涉及到较大的计算量, 因此较低的 FPS 也在情理之中。

综上所述, 在 $k_m = 9\%$ 时, 3D-MRAI 算法的准确率为 98.10% 左右, 其估计的位姿的距离精度为 $0.82 \pm 0.21\text{mm}$, 角度精度为 $0.91 \pm 0.29\text{deg}$ 。

5.4 本章小结

@todo: pose 本章小结

第 6 章 算法应用——Bin-Picking

本章主要介绍 3D-MRAI 算法的实际应用——Bin-Picking。首先介绍一下 Bin-Picking 相关背景，以及近些年的具体研究与发展；然后详细介绍基于 3D-MRAI 算法所开发的一套解决 Bin-Picking 问题的视觉系统，包括硬件的选型、开发环境、系统的框架设计以及算法的具体实现；最后介绍了针对所开发的 Bin-Picking 视觉系统，进行了随机抓取的实验，测试了系统抓取的成功率以及系统的抓取速度。

6.1 Bin-Picking 背景与现状

使用机器人分拣散乱的工件的问题，在学术上我们称之为 Bin-Picking，Bin-Picking 并不是一个崭新的问题，学术上对这个问题的研究以及有了五十多年的历史。典型的 Bin-Picking 系统主要包括三部分：机器人、视觉检测模块和计算机控制模块。其中，视觉检测模块是整个 Bin-Picking 系统的核心部分，通过视觉检测模块对存放散乱工件的物料箱进行分析，获取目标工件的位姿，计算机控制模块根据视觉检测模块的检测结果规划机器人的运动路径，然后机器人执行完成工件的抓取。

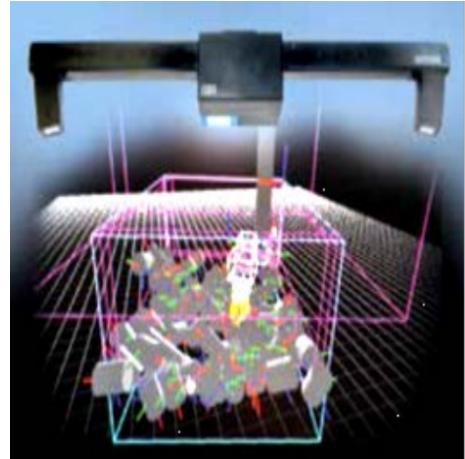
传统的 Bin-Picking 中检测估计目标工件的算法大致可以分为两类：一类是基于特征匹配的算法，另一类是基于模板匹配的算法。基于特征匹配的算法，通过某些特征描述目标工件，如边角、空洞等特征，然后通过分析特征在空间中的旋转变换和平移变换来估计目标零件的位姿。这一类方法受工件纹理或者结构以及传感器的精度影响很大。另外一类基于模板匹配的方法的精度受限与模板的数量，要获得较高的精度就需要大量的模板，而大量的模板会造成算法运行时间过长。

工业上用于解决 Bin-Picking 问题的视觉系统也有许多，如图6.1所示。日本的 Fanuc 公司推出了基于 iRVision 的 Bin-Picking 系统，该系统通过四个相机进行三维视觉重建，然后进行目标定位 (Connolly 2007)。德国的 ISRA Vision 公司推出了 3D Shape Scan 系统，丹麦的 Scape Technologies 公司推出了 Scape-Tech Discs 系统，德国的 Sick 公司退出了 PLB-500 系统，诸如类似的 Bin-Picking 系统还有许多，这些 Bin-Picking 系统的价格大多二十万以上，并且抓取成功率和速度也往往难以满足客户需求，因此工业上还是缺少成熟的、价格便宜的、抓取成功率高、速

度快的 Bin-Picking 解决方案。



(a) Fanuc 的 iRVision 系统



(b) ISRA Vision 的 3D Shape Scan 系统



(c) Scape Technologies 的 Scape-Tech Discs 系统



(d) Sick 的 PLB-500 系统

图 6.1 工业上典型的 Bin-Picking 解决方案

随着近几年一些高性价比的 3D 相机的出现,如微软的 Kinect 系列、Intel 的 RealSense 系列,加上近几年深度学习的巨大发展,使得开发一种性价比高的、抓取成功率高的、速度快的 Bin-Picking 系统成为可能。但尽管深度学习在计算机视觉领域(Computer Vision)有了大量的研究,但在机器人感知(Robot Perception)领域的应用还比较少,因此本文将深度学习在计算机视觉领域内的成果通过一些改进引入到 Robot Perception 领域,再结合传统的全局点云匹配算法,剔除了 3D-MRAI 算法,可以用于解决 Bin-Picking 相关问题。此外,本文基于 Intel 的 RealSense SR300 相机所提出的对偶 RGB-D 相机构建也为整个 Bin-Picking 系统提供了高性价比的相机解决方案。

6.2 基于 3D-MRAI 的随机分拣系统

6.2.1 系统硬件设计

本文所设计的基于 3D-MRAI 的随机分拣系统的硬件系统如图6.2所示。从

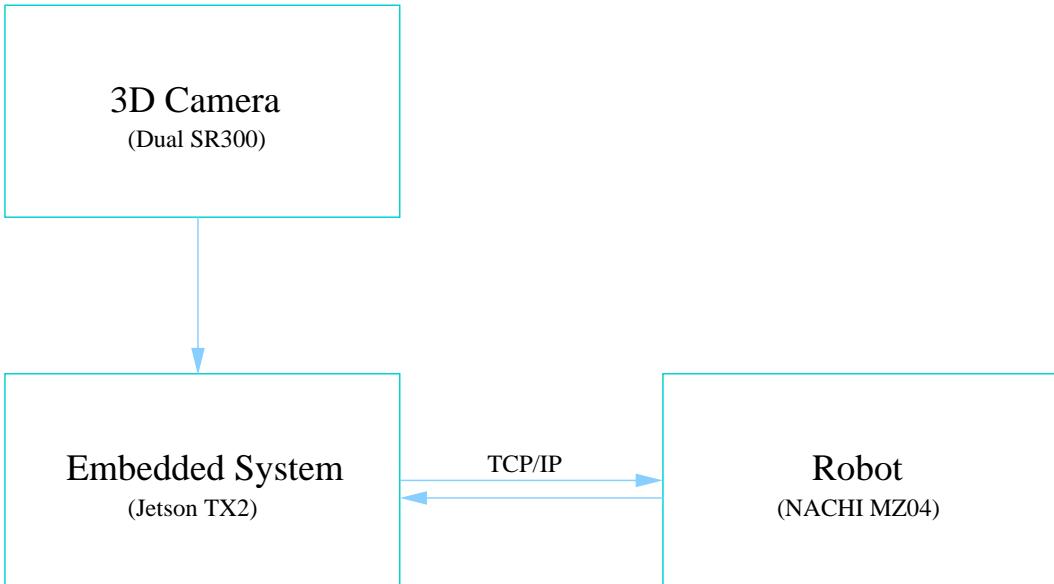


图 6.2 基于 3D-MRAI 的随机分拣系统硬件框架

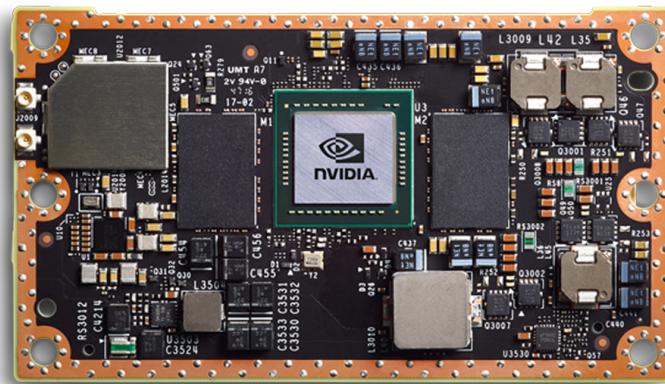
图6.2可以看出,所设计的随机分拣系统的硬件系统由三个部分构成: 相机模块、嵌入式计算模块以及机器人模块。对于相机模块,根据 3D-MRAI 算法的输入,需要相机能采集 RGB-D 图像,并且考虑整个系统的响应时间以及价格因此,希望相机模块的采集时间尽可能短,性价比尽可能高,因此选用了以结构光为原理的 3D 相机,并根据第 2 章所设计的对偶 RGB-D 相机,用两个 SR300 相机构成了对偶 RGB-D 相机模块。

嵌入式计算模块选用了搭载了 NVIDIA 公司的 Jetson TX2 模块的嵌入式计算机,如图6.3(a), Jetson TX2 如图6.3(b)所示。由于 3D-MRAI 算法使用了深度神经网络,因此所选用的计算机最好要搭载一块 GPU,当然由于模型的训练可以在服务器上完成,只需要在选用的计算机上跑模型的 Interface,因此其 GPU 性能也不需要特别好。另外,考虑到系统需要长时间运行,因此选用了低功耗的嵌入式计算机。所选用的搭载 Jetson TX2 模块的嵌入式计算机拥有一块 Pascal 架构的 GPU, 256 个 CUDA cores, CPU 是 HMP Dual Denver 加四块 ARM A57, 内存 8G (LPDDR4), 还拥有 1 Gigabit Ethernet, 802.11ac WLAN 以及 Bluetoothd, 在系统计算资源、功耗以及通信上完全满足整个 Bin-Picking 系统的要求。

机器人模块选用了 NACHI 的六轴机械臂 MZ04, 如图6.4所示。由于一般正常的随机分拣系统所要抓取的工件各种位姿都有,意味着所要抓取的工件有六个



(a) 搭载 Jetson TX2 模块的嵌入式计算机



(b) Jetson TX2 模块

图 6.3 嵌入式计算模块

自由度，因此所选用的机器人末端至少也要有六个自由度，不然难以完成各种姿态工件的抓取任务，因此选用了工业上常见的六轴机械臂，至于为何选用 NACHI 的 MZ04 这个型号，是出于合作方的需要，并不由个人意志决定，当然何种机械臂也不是本文的重点，所设计的视觉算法对机械臂也没什么特殊的要求，因此此处不作详细介绍。

实际搭建随机分拣系统环境如图6.5所示，图中相机固定在支架上，与机械臂构成了 eye-to-hand 的形式，当然也可以将相机固定在机械臂末端构成 eye-in-hand 形式，两种固定相机的形式略有不同，但对视觉识别算法那没有影响，只与控制流



图 6.4 六轴机械臂 NACHI MZ04

程和相机与机器人之间的标定有关系,后文会具体介绍到。嵌入式计算机通过相机采集物料箱内的图像,然后运行基于 3D-MRAI 的视觉系统,得出目标工件位姿,然后规划机械臂路径,通过 TCP/IP 通信,控制机械臂完成抓取任务,并根据机械臂的运动状态控制整个系统的流程。

图 6.5 实际搭建随机分拣系统环境

6.2.2 系统软件设计

需求分析: 基于 3D-MRAI 的随机分拣系统的软件运行在所采用的嵌入式计算机上,主要需要以下几点功能:

- 图像的采集与处理
- 与机器人的通信
- 3D-MRAI 算法的实现
- 机器人相关的处理
- 可视化界面

针对以上几点需求,整个软件分为五个模块,如图6.6所示。Camera 模块主要实现对偶 RGB-D 图像的匹配与合成,旨在提供高质量的 RGB-D 图像;Communication 模块主要实现一个 TCP 服务器,并且定义了与机器人的通信协议,旨在提供与机器人高效稳定的通讯服务;Vision 模块主要实现了 3D-MRAI 算法,旨在根据输入的 RGB-D 图像,输出目标工件的位姿;Robot 模块主要实现与机器人相关的一些路径规划,根据目标工件的位姿生成机械臂抓取的位姿;GUI 模块主要实现对检

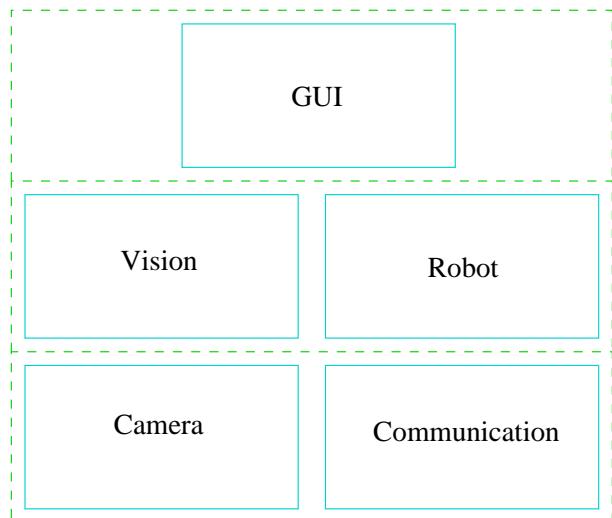


图 6.6 系统软件模块设计

测结果的可视化，以及一些简单的流程控制。每个模块具体的实现后文会详细介绍。

开发环境：所采用的嵌入式计算机使用的是嵌入式 Linux 操作系统，当然由于嵌入式计算机特性，软件的开发主要在通用计算机上，通用计算机也使用了 Linux 操作系统，这样在编写完成的软件可以无缝拷贝到嵌入式 Linux 操作系统上，但由于通用计算机和嵌入式计算机的 CPU 架构不同，将程序拷贝到嵌入式计算机上后，需要重新编译。当然也可以考虑直接在通用计算机上交叉编译嵌入式计算机上的可执行程序，但考虑到调试方便，并且所使用的嵌入式计算机性能强劲，并未使用交叉编译的方式。考虑到系统的性能以及算法的复杂性，整个系统软件使用 C++ 11 编写，Clang 作为编译器，CMake 作为自动化编译工具，git 作为版本控制，具体如表6.1所示。

操作系统	Ubuntu 16.04
编译器	Clang 3.8.0
构建系统	CMake 3.5.1
版本控制	git 2.7.4

表 6.1 系统开发环境

软件依赖：系统软件的依赖如下所示：

- librealsense < 2.0.0
- OpenCV >= 3.0.0
- PCL(Point Cloud Library) >= 1.7.0
- Tensorflow >= 1.2.0
- Glog >= 0.3.4

- glfw >= 3.1.2

上述所有的软件都是跨平台、开源的软件, librealsense 是所使用的 RGB-D 相机 SR300 的驱动以及 SDK, 系统主要使用它获取相机采集的图片; OpenCV 是一个基于 BSD 许可(开源)发行的跨平台计算机视觉库, 系统主要使用 OpenCV 完成对相机采集图像的处理以及对偶 RGB-D 相机图像的合成与匹配算法; PCL 是一个通用的开源点云库, 它实现了大量点云相关的通用算法和高效数据结构, 涉及到点云获取、滤波、分割、配准、检索、特征提取、识别、追踪、曲面重建、可视化等。支持多种操作系统平台, 可在 Windows、Linux、Android、Mac OS X、部分嵌入式实时系统上运行, 系统主要使用 PCL 完成一些点云相关的算法; TensorFlow 是谷歌基于 DistBelief 进行研发的深度学习框架, 系统主要使用 Tensorflow 完成 3D-MRAI 算法中的深度神经网络; Glog 是谷歌开发的一个 C++ 语言的应用级日志记录框架, 提供了 C++ 风格的流操作和各种助手宏, 系统主要使用 Glog 完成软件的日志记录; glfw 是一个 OpenGL 图形库, 系统主要使用 glfw 完成 GUI 模块的设计。

Camera module: Camera 模块的主要接口是一个虚基类 Camera, 如图6.7所示。相机模块通过虚基类 Camera 定义了一些通用的接口函数, 其他具体的相机

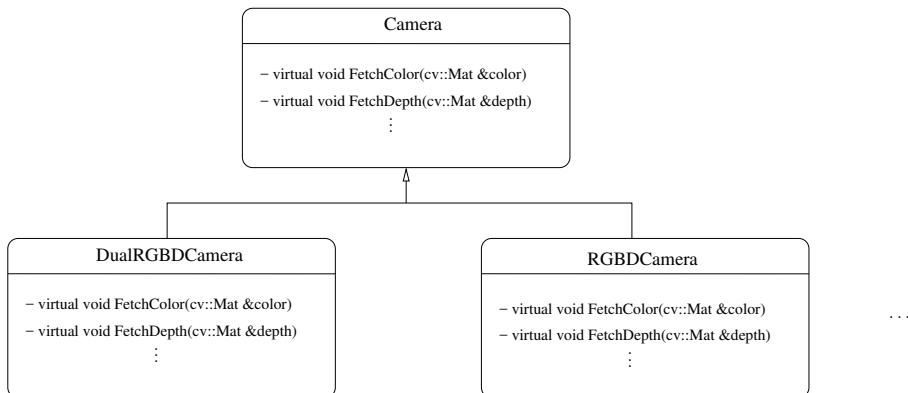


图 6.7 Camera module UML

通过继承这个基类来实现, 对于外部使用者来说并不需要关心这些继承 Camera 类的具体实现, 只需要调用 Camera 中定义的接口即可。整个模块具有很强的扩展性, 如增加一个新的相机可以通过增加一个继承 Camera 的类, 外部调用的模块无需改写。实际上, 使用何种相机通过设置配置文件可以由用户选择。

Communication module: Communication 模块主要使用 C++ 构建了一个 TCP 服务器, 并且规定了通信的协议。讲道理, 系统的通讯并不复杂, 并且数据量也很小, 基本上就视觉系统告诉机器人运动到哪, 然后机器人告诉视觉系统是否运动到了目的地这些简单的信息交流。在仔细研究机器人上具体编程后, 由于机器人上编程比较单调且繁琐, 因此通讯服务的服务器运行在嵌入式计算机上, 并且, 定

义了接收和发送两类信息,发送信息指的是从嵌入式计算机发送到机器人控制器上的信息,接收信息类似。具体定义了一个类模板,如代码6.1所示。

Listing 6.1 TCP server template

```
template <typename RecvMsgT, typename SendMsgT>
class SyncTCPServer {
public:
    SyncTCPServer(std::string address = "127.0.0.1", unsigned
                  short port = 8000);

    void WaitingClient() {
        acceptor_ ->accept(*socket_);
    }

    /**
     * Receive message from client
     * @param msg the received message
     * @return read bytes size
     */
    int RecvMsg(RecvMsgT &msg);

    /**
     * Send message to client
     * @param msg the message will be sent
     * @return write bytes size
     */
    int SendMsg(const SendMsgT &msg);

private:
    boost::asio::io_service io_service_;
    boost::shared_ptr<boost::asio::ip::tcp::acceptor> acceptor_;
    boost::shared_ptr<boost::asio::ip::tcp::socket> socket_;
};
```

Vision module: Vision 模块的主要类的 UML 图如图6.8所示。Vision 类提供在图像中找出目标工件位姿的接口,其核心是 3D-MRAI 算法,因此也有两个模块: Detector 和 Matcher。Detector 和 Matcher 的设计思想和 Camera 模块类似,通过定义接口屏蔽具体实现,从而提高了程序的扩展性,因为显然可以有多种检测的方法,比如本文就有 3D Faster R-CNN 和 3D Mask R-CNN 两种实现,通过这种方式可以在不改动程序代码的情况下,通过配置文件快速切换所想要使用的算法。

Vision 类的核心就是 3D-MRAI 算法,算法具体内容已经在第5 章中详细叙述了,但此处运用于 Bin-Picking 系统,为了提高系统的效率,针对 Bin-Picking 这个系统,在实现细节上对 3D-MRAI 做了些改动,或者说 trick。其中最主要的 trick 是不再对 3D Faster/Mask R-CNN 中的每个检测结果都去运行 A4PCS-ICP 算

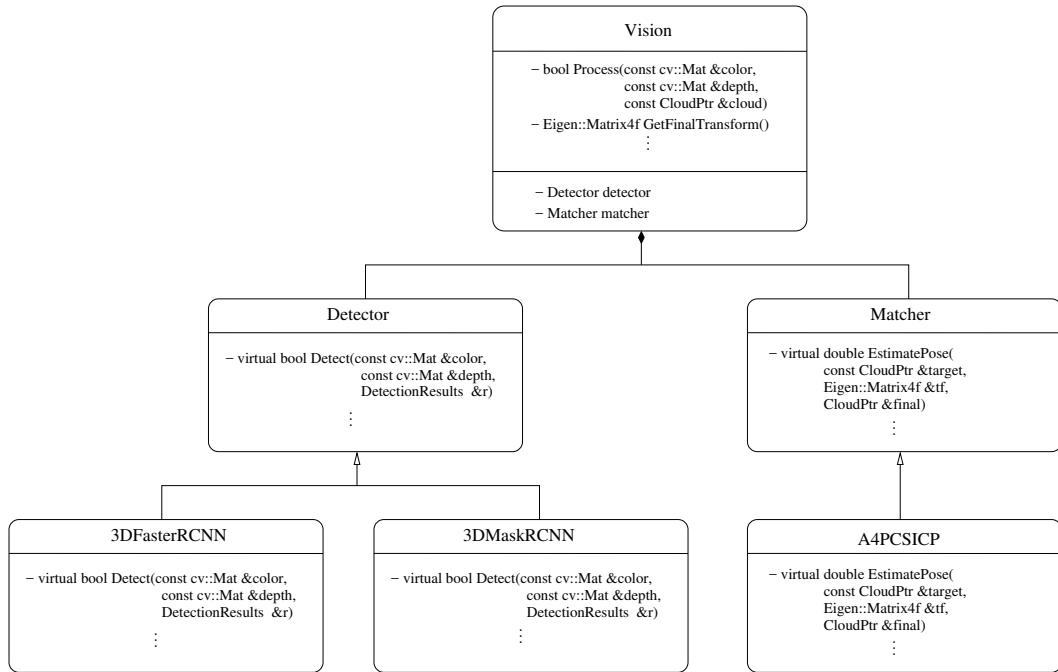


图 6.8 Vision module UML

法, 取而代之的是通过对每个 BBox/Mask 提取的点云根据距离工作台的高度排序, 从位置最高的点云开始运行匹配算法, 满足条件就返回。这么做的原因是, Bin-Picking 系统每次抓取只能抓取一个工件, 并且, 理论上位于物料堆最上面的工件显然最好抓取。增加这个 trick 后的 3D-MRAI 算法的流程如算法10所示, 这么做大大减少了视觉系统的运算时间, 提高了整个系统的抓取工作效率。除此之外, 为了避免抓取时不必要的碰撞, 在工件的抓取点附近还会检查有没有障碍物。

TODO: 避免碰撞具体实现。

Robot module: Robot 模块根据 Vision 模块输出的工件位姿, 将其变换到机器人坐标系下, 然后生成轨迹, 发送给机器人。由于 Vision 模块与 Robot 模块之间是解耦的, Vision 模块输出的工件位姿是在相机坐标系下的, 为了将相机坐标系下的位姿变换到机器人坐标系下, 还需要进行相机和机器人之间的标定(手眼标定), 所设计的系统的相机固定在支架上, 与机器人分离, 因此是一个典型的 eye-to-hand calibration。

TODO: 具体标定的方法

将工件位姿从相机坐标系变换到机器人坐标系后, 便可得机器人抓取的位姿, 机器人抓取的位姿与工件的位姿是事先标定好的, 并且为了提高抓取的成功率, 对一个工件设了多组抓取位姿, 根据工件的位姿选取合适的抓取位姿。

GUI module: 为了可视化视觉系统的检测情况, 设计了 GUI 模块实时展示检测结果, 由于系统中存在 3D 的点云, 因此采用 OpenGL 库在三维空间中可视化结果, 并且可以对程序进行简单的控制。三维可视化的界面如图6.9所示。三维可

算法10: 3D-MRAI with Tricks**Input:** RGB Image I , Depth Map D , CAD Models M **Output:** Set of Pose and Class Res

```

1  $P \leftarrow \emptyset;$ 
2 forall  $M_i \in M$  do
3    $P \leftarrow \{P, CAD2PointCloud(M_i)\};$ 
4  $H = Depth2HHA(D);$ 
5  $Q = Depth2PointCloud(D);$ 
6  $Mask, Class \leftarrow 3DMAS KRCNN(I, H); // \text{ Same with } 3DFASTERRCNN$ 
7  $Q_{sorted} \leftarrow \emptyset;$ 
8 forall  $m_i \in Mask, c_i \in Class$  do
9    $Q_i \leftarrow Crop(Q, m_i);$ 
10   $Q_{sorted} \leftarrow Q_{sorted}, [Q_i, c_i];$ 
11  $SORTBAS EDONHEIGHT(Q_{sort});$ 
12 forall  $[Q_i, c_i] \in Q_{sort}$  do
13    $P_i \leftarrow P(c_i);$ 
14    $T_i, S_i \leftarrow A4PCSI CP(P_i, Q_i);$ 
15   if  $S_i > S_{min}$  then
16     return  $[T_i, c_i];$ 

```

化界面中标出了相机坐标系、检测出要抓取的工件的位姿，整个场景是相机采集到的3D数据，红色部分点云是将目标工件CAD模型变换到所计算得到的工件位姿，因此红色点云与相机采集到的3D目标点云相重合说明视觉系统的检测正确。整个界面是3D的，可以通过鼠标放大、缩小、旋转，也可通过键盘控制视角的前进后退左右移动，全方位360度观察当前检测结果，如图6.10所示，在不同视角下观察检测结果。除此之外，还可以通过键盘保存当前所展示的点云和其他一些数据，方便进一步分析和观察。

6.3 随机分拣实验

6.3.1 实验内容

设计的随机分拣实验在所搭建的工作台上进行，在物料箱中随机放慢物料，如图??所示，然后运行整个随机分拣系统，将物料中的全部工件分拣到另外一个

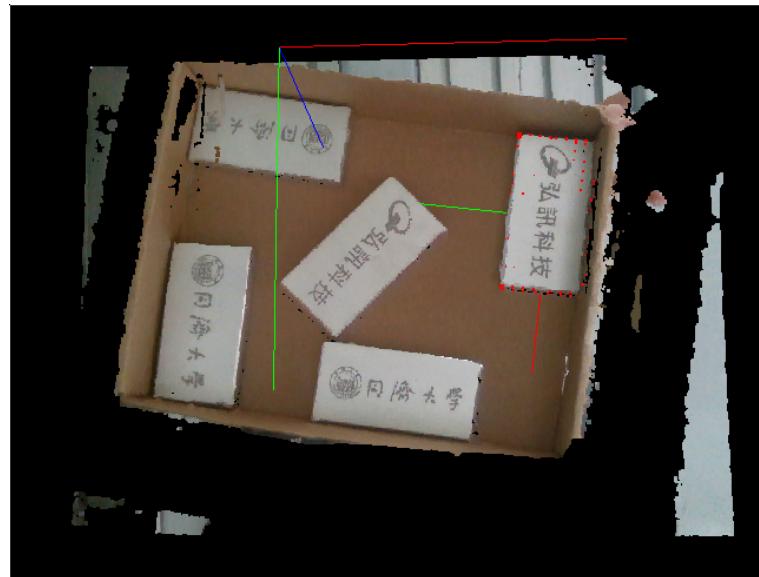


图 6.9 视觉系统三维可视化

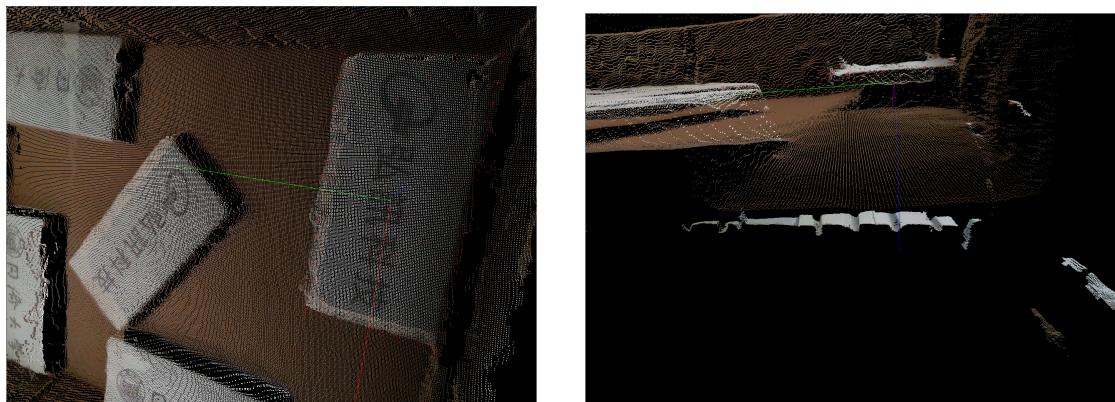


图 6.10 不同视角下观察检测结果

箱子中，分拣完一箱后，再随机填充物料，一共统计十箱物料的抓取结果。

评价系统的指标主要是系统的抓取成功率：

$$R = \frac{m}{n} \times 100\% \quad (6.1)$$

其中 n 表示总的抓取次数， m 表示成功抓取次数。一次成功的抓取是指机械臂成功将一个物料从物料箱中取出，然后放置到另外一个箱子中。除了系统的成功抓取率，为了考察系统的快速性，定义系统的响应时间 T_r 为从机器人请求开始抓取到机器人收到工件位姿，这个时间也是整个软件的响应时间，主要包括了：

- 相机采集时间
- 视觉算法计算时间
- 通讯时间

另外再定义机械臂从开始抓取到将物料抓出箱子的时间为抓取时间 T_1 ，机械臂从将物料抓出箱子到放置完物料回到抓取起始点的时间为放置时间 T_2 ，如果机

械臂只要回到抓取起始点就立即请求下一个抓取位姿，则一个工作周期的总时间为

$$T = T_r + T_1 + T_2 \quad (6.2)$$

但是，考虑到系统的相机是固定在支架上的，因此，只要机械臂将物料抓取出箱子就可以请求下一个抓取位姿，所以一个工作周期的总时间可以缩减为

$$T = T_1 + \max(T_r, T_2) \quad (6.3)$$

对于评价视觉系统来说，我们更关心系统响应时间 T_r ，对于评价整个 Bin-Picking 系统来说，显然工作周期 T 更重要。

6.3.2 实验结果

所设计的随机分拣系统的硬件系统抓取完十箱物料后，统计每箱的成功率、平均响应时间、平均抓取时间、平均放置时间和平均工作周期，绘制成表6.2以及图6.11。从表中可以发现所设计的基于 3D-MRAI 的随机分拣系统的平均抓取成

	成功率	响应时间 T_r	抓取时间 T_1	放置时间 T_2	工作周期 T
1	100%	711ms	7.6s	4.2s	12.5s
2	100%	729ms	6.8s	4.2s	11.0s
3	100%	708ms	6.5s	4.2s	10.7s
4	100%	701ms	8.1s	4.2s	12.3s
5	100%	713ms	9.3s	4.2s	13.5s
6	100%	722ms	6.6s	4.2s	10.8s
7	100%	693ms	7.9s	4.2s	12.1s
8	100%	732ms	9.1s	4.2s	13.3s
9	100%	718ms	8.9s	4.2s	13.1s
10	100%	723ms	6.9s	4.2s	11.1s
Avg.	100%	715ms	7.77s	4.20s	12.04s

表 6.2 随机分拣实验结果

功率为 100%，但理论上随着抓取次数的增加，个人认为系统一定会出现抓取失败的情况，并且实验中所使用的物料也相对单一，不同的物料理论上也会影响视觉的识别效果，因此换一种物料系统的抓取成功率也可能达不到 100%。但总体上来说，此次实验 100% 的成功率充分说明了所设计的 Bin-Picking 视觉系统完全能满足一般随机分拣任务，成功率高。

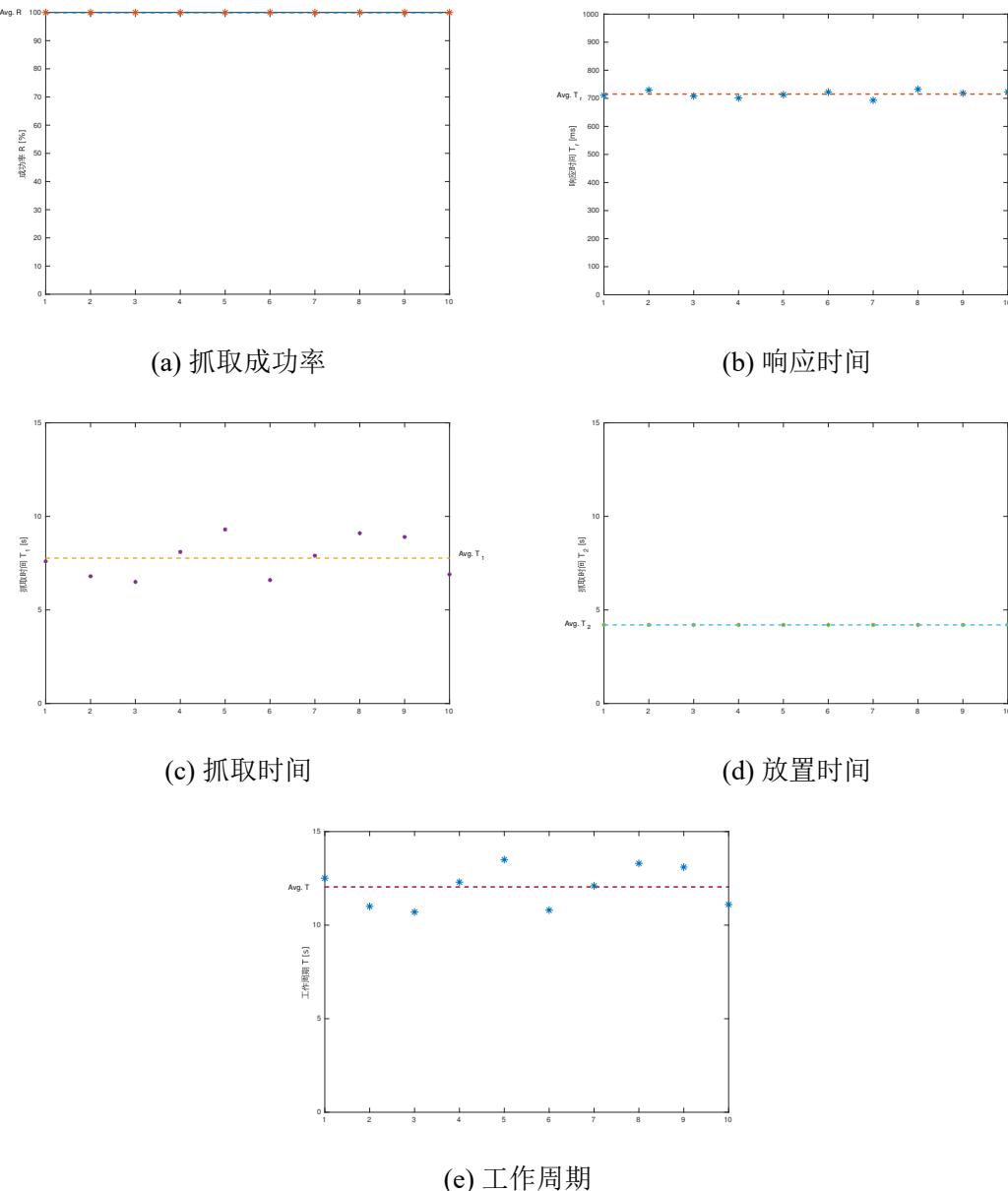


图 6.11 随机分拣实验结果

另外,视觉系统的响应时间为 $715ms$ 左右,分拣系统的工作周期为 $12.04s$ 左右,从表中数据可以发现工作周期基本上为抓取时间和放置时间之和,也就是完全是机械臂运动的时间,除了一开始系统启动时会增加额外的响应时间,但这对真个工作时间来说可以忽略不计,因此,所设计的基于 3D-MRAI 算法的 Bin-Picking 视觉系统在时间上完全满足一般分拣任务的要求,实时性高。

6.4 本章小结

@todo: 实验本章小结

第 7 章 结论与展望

@TODO: 结论与展望

7.1 结论

7.2 进一步工作的方向

致谢

逾尺的札记和研究纪录凝聚成这么薄薄的一本，高兴和欣慰之余，不禁感慨系之。记得鲁迅在一篇文章里写道：“人类的奋战前行的历史，正如煤的形成，当时用大量的木材，结果却只是一小块”。倘若这一小块有点意义的话，则是我读书生活的最好纪念，也令我对于即将迈入的新生活更加充满信心。回想读书生活，已经整整二十个年头，到同济求学将近五年，攻读博士学位也已三年了。进入同济大学以来，深深醉心于一流学府的大家风范。名师巨擘，各具特点；中西融合，文质相顾。处如此佳境以陶铸自我，实乃人生幸事。

2018 年 3 月

参考文献

- [1] AIGER D, MITRA N J, COHEN-OR D, 2008. 4-points congruent sets for robust pairwise surface registration[C]//ACM Transactions on Graphics (TOG): volume 27. [S.l.]: ACM: 85.
- [2] ARYA S, MOUNT D M, NETANYAHU N S, et al. An optimal algorithm for approximate nearest neighbor searching fixed dimensions[J]. Journal of the ACM (JACM), 1998, 45(6): 891–923.
- [3] BESL P J, MCKAY N D, 1992. Method for registration of 3-d shapes[C]//Sensor Fusion IV: Control Paradigms and Data Structures: volume 1611. [S.l.]: International Society for Optics and Photonics: 586–607.
- [4] BOLLES R C, FISCHLER M A, 1981. A ransac-based approach to model fitting and its application to finding cylinders in range data.[C]//IJCAI: volume 1981. [S.l.: s.n.]: 637–643.
- [5] BROWN D C. Decentering distortion of lenses[J/OL]. Photogrammetric Engineering and Remote Sensing, 1966. <https://ci.nii.ac.jp/naid/10022411406/en/>.
- [6] COLLET A, MARTINEZ M, SRINIVASA S S. The moped framework: Object recognition and pose estimation for manipulation[J]. The International Journal of Robotics Research, 2011, 30(10): 1284–1306.
- [7] CONNOLLY C. A new integrated robot vision system from fanuc robotics[J]. Industrial Robot: An International Journal, 2007, 34(2): 103–106.
- [8] CORSINI M, DELLEPIANE M, GANOVELLI F, et al. Fully automatic registration of image sets on approximate geometry[J]. International journal of computer vision, 2013, 102(1-3): 91–111.
- [9] DIAS A S, BRITES C, ASCENSO J, et al. Sift-based homographies for efficient multiview distributed visual sensing[J]. IEEE Sensors Journal, 2015, 15(5): 2643–2656.
- [10] GEIGER A, ROSER M, URTASUN R. Efficient Large-Scale Stereo Matching[J]. Accv, 2010.
- [11] GOODRICH M T, MITCHELL J S, ORLETSKY M W, 1994. Practical methods for approximate geometric pattern matching under rigid motions:(preliminary version)[C]//Proceedings of the tenth annual symposium on Computational geometry. [S.l.]: ACM: 103–112.
- [12] GOOGLE, 2012. Tango[EB/OL]. <https://developers.google.com/tango>.
- [13] GUPTA S, ARBELAEZ P, MALIK J, 2013. Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images[C/OL]//2013 IEEE Conference on Computer Vision and Pattern Recognition. IEEE: 564–571. <http://ieeexplore.ieee.org/document/6618923/>. DOI: 10.1109/CVPR.2013.79.
- [14] GUPTA S, GIRSHICK R B, ARBELÁEZ P A, et al. Learning Rich Features from {RGB-D} Images for Object Detection and Segmentation[J/OL]. Computer Vision - {ECCV} 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part {VII}, 2014, 8695: 345–360. http://dx.doi.org/10.1007/978-3-319-10584-0_23. DOI: 10.1007/978-3-319-10584-0_23
- [15] HE K, ZHANG X, REN S, et al., 2016. Deep residual learning for image recognition[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.]: 770–778.

- [16] HE K, GKIOXARI G, DOLLÁR P, et al., 2017. Mask R-CNN[EB/OL]. <http://arxiv.org/abs/1703.06870>.
- [17] HEIKKILÄ J. Geometric camera calibration using circular control points[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(10): 1066–1077. DOI: 10.1109/34.879788.
- [18] HINTERSTOISSE S, CAGNIART C, ILIC S, et al. Gradient response maps for real-time detection of textureless objects[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(5): 876–888.
- [19] HINTERSTOISSE S, LEPETIT V, ILIC S, et al., 2012. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes[C]//Asian conference on computer vision. [S.l.]: Springer: 548–562.
- [20] HORN B K. Closed-form solution of absolute orientation using unit quaternions[J]. JOSA A, 1987, 4(4): 629–642.
- [21] IMAGENET, 2011. Imagenet[EB/OL]. <http://www.image-net.org>.
- [22] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Alexnet[J]. Advances In Neural Information Processing Systems, 2012: 1–9. DOI: <http://dx.doi.org/10.1016/j.protcy.2014.09.007>.
- [23] LI X, GUSKOV I, 2005. Multiscale features for approximate alignment of point-based surfaces.[C]//Symposium on geometry processing: volume 255. [S.l.: s.n.]: 217.
- [24] LIN T Y, DOLLÁR P, GIRSHICK R, et al., 2017. Feature pyramid networks for object detection[C]//CVPR: volume 1. [S.l.: s.n.]: 4.
- [25] LOOP C, Zhengyou Zhang. Computing rectifying homographies for stereo vision[J/OL]. Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), 2001, 1: 125–131. <http://ieeexplore.ieee.org/document/786928/>. DOI: 10.1109/CVPR.1999.786928.
- [26] MICROSOFT, 2012. Kinect[EB/OL]. <https://www.xbox.com/en-US/xbox-one/accessories/kinect>.
- [27] MIT-PRINCETON, 2016. ”shelf & tote” benchmark dataset for 6d object pose estimation [EB/OL]. <http://apc.cs.princeton.edu/#shelf-and-tote-benchmark-dataset>.
- [28] REN S, HE K, GIRSHICK R, et al. Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks[M]. [S.l.: s.n.], 2016: 1–14.
- [29] RUSU R B, BLODOW N, BEETZ M, 2009. Fast point feature histograms (fpfh) for 3d registration[C]//Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. [S.l.]: IEEE: 3212–3217.
- [30] SALTÌ S, TOMBARI F, DI STEFANO L. Shot: Unique signatures of histograms for surface and texture description[J]. Computer Vision and Image Understanding, 2014, 125: 251–264.
- [31] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
- [32] SUR F, NOURY N, BERGER M O, 2008. Computing the Uncertainty of the 8 point Algorithm for Fundamental Matrix Estimation[C/OL]//Proceedings of the British Machine Vision Conference 2008. 96.1–96.10. <http://www.bmva.org/bmvc/2008/papers/269.html>. DOI: 10.5244/C.22.96.
- [33] WOLFSON H J, RIGOUTSOS I. Geometric hashing: An overview[J]. IEEE computational science and engineering, 1997, 4(4): 10–21.

- [34] XIE S, GIRSHICK R, DOLLÁR P, et al., 2017. Aggregated residual transformations for deep neural networks[C]//Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on. [S.l.]: IEEE: 5987–5995.
- [35] ZENG A, SONG S, NIESSNER M, et al. 3dmatch: Learning the matching of local 3d geometry in range scans[J]. arXiv, 2016, 1603.
- [36] ZHANG Z. A Flexible New Technique for Camera Calibration (Technical Report)[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 22(11): 1330–1334. DOI: 10.1109/34.888718.

附录 A 补充资料

可能需要补充的内容……

个人简历、在学期间发表的学术论文与研究成果

@todo: resume

个人简历

同济人,男/女,xxxx年xx月生。

xxxx年xx月毕业于xxxx大学xxxx专业获xx学位。

xxxx年xx月入同济大学攻读xx学位。

已发表论文

[1] ...

[2] ...

[3] ...

已获得专利

[1] ...

[2] ...