浙江工业大学

本科毕业设计论文

题目：双目视觉立体匹配算法设计与实现

作者姓名 王 灏

指导教师 宣琦研究员

专业班级 通信工程 1301

学　　院 信息工程学院

提交日期 2017 年 6 月 6 日

浙江工业大学本科毕业设计论文

双目视觉立体匹配算法设计与实现

作者姓名：王　灏

指导教师：宣琦 研究员

浙江工业大学信息工程学院

2017 年 6 月

Dissertation Submitted to Zhejiang University of Technology for the Degree of Bachelor

Binocular Stereo Matching Algorithm Design and Implementation

Student: Hao Wang

Advisor: Professor Qi Xuan

College of Information Engineering

Zhejiang University of Technology

June 2017

浙江工业大学

本科生毕业设计(论文、创作)诚信承诺书

本人慎重承诺和声明：

1. 本人在毕业设计（论文、创作）撰写过程中，严格遵守学校有关规定，恪守学术规范，所呈交的毕业设计（论文、创作）是在指导教师指导下独立完成的；

2. 毕业设计（论文、创作）中无抄袭、剽窃或不正当引用他人学术观点、思想和学术成果，无虚构、篡改试验结果、统计资料、伪造数据和运算程序等情况；

3. 若有违反学术纪律的行为，本人愿意承担一切责任，并接受学校按有关规定给予的处理。

学生（签名）：

年　　月　　日

浙 江 工 业 大 学

本科生毕业设计（论文、创作）任务书

专 业 通信工程 班 级 通信 1301 学生姓名/学号 王灏/201303090119

一、设计（论文、创作）题目：

双目视觉立体匹配算法设计与实现

二、主要任务与目标：

双目视觉立体匹配系统，实现双目视觉图像数据采集及匹配，实现场景深度信息的计算，具有较快的运行速度。

三、主要内容与基本要求：

1.采集双目视觉图像 2.设计双目视觉立体匹配算法对双目视觉图像进行匹配，完成相关软件程序，达到指定性能指标 3.撰写毕业论文，提交相关的算法程序流程图及程序代码。

四、计划进度：

2016.12.20-2017.2.15 收集相关资料文献，学习数据库以及编程知识；完成外文翻译、文献综述；熟悉课题，做好开题准备，有初步设计方案；2017.2.16-3.5 完成开题报告，参加开题交流； 2017.3.6-4.23 完成数据采集，算法实现，并开发软件系统，接收中期检查； 2017.4.24-5.25 算法测试与改进，做出最终设计成品，撰写毕业论文初稿； 2017.5.26-6.12 论文修改，毕业答辩，提交相关文档资料。

五、主要参考文献：

[1]N.Mayer,E.Ilg,P.Hausser,P.Fischer,D.Cremers,A.Dosovitskiy,T.Brox.A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation[C].2015:arXiv preprint arXiv:1512.02134

[2]J.Zbontar,Y.LeCun.Stereo matching by training a convolutional neural network to compare image patches[J].Journal of Machine Learning Research.2016.17:1-32

[3]J.Zbontar,Y.LeCun.Computing the stereo matching cost with a convolutional neural network[C].IEEE Conference on Computer Vision and Pattern Recognition.2015:1592-1599

任务书下发日期 2016 年 12 月 20 日

设计（论文、创作）工作自 2016 年 12 月 20 日 至 2017 年 6 月 20 日

设计（论文、创作）指导教师 宣琦

系主任（专业负责人）

主管院长

双目视觉立体匹配算法设计与实现

摘　要

立体视觉技术一直是计算机视觉领域研究的一个重要分支，特别是近年来，随着无人驾驶、工业自动化、机器人技术的发展，如何从二维图像中提取出三维的深度信息成为一个重要的命题。一个完整的双目视觉系统包括离线标定，双目矫正，立体匹配和三维重建。其中立体匹配是双目视觉技术的核心问题，也是本文研究的重点。立体匹配算法的精度和速度对于双目立体视觉系统的性能起着至关重要的作用。

近年来深度学习技术开始被广泛应用到各个领域中，特别是在图像处理领域，深度学习的特征提取特性表现尤为优秀。随着深度学习技术的深入发展，研究人员开始尝试在一些早已被充分研究的问题中应用神经网络，试图提高图像处理的效率和准确率。

本文主要通过卷积神经网络来计算立体匹配代价，并通过左右一致性检查，半全局匹配，中值滤波和双边滤波等后处理步骤进一步优化视差图。本文使用立体视觉数据集 KITTI 2012 和 KITTI 2015 构造混合数据集来进行卷积神经网络模型的训练、验证和测试。此外，本文结合 OpenCV 和 Matlab 工具箱进行离线标定，基于 TensorFlow 框架构建卷积神经网络模型，最后利用 OpenCV 计算机视觉库进行三维重建。同时，我们使用 Flask 框架和 Ngrok 服务搭建 Web 服务器，并基于 Qt 框架构建客户端程序实现双目数据上传和结果显示。

关键词：双目立体视觉，立体匹配，卷积神经网络，TensorFlow，OpenCV

BINOCULAR STEREO MATCHING ALGORITHM DESIGN AND IMPLENTATION

ABSTRACT

Stereo vision technology has always been an important branch of computer vision research. Especially in recent years, with the development of self-driving, industrial automation and robot technology, how to extract three-dimensional depth information from two-dimensional images has become an important topic. A complete binocular vision system includes off-line calibration, binocular rectification, stereo matching and three-dimensional reconstruction. Stereo matching is the core of binocular vision technology and the focus of this paper. The accuracy and speed of the stereo matching algorithm play a crucial role in the performance of the binocular stereo vision system.

In recent years, deep learning technology has been widely applied in various fields. Especially in the field of image processing, deep learning performed quite well in extracting feature. With the development of deep learning, researchers began to try using neural networks in some issues that have already been fully studied in an attempt to improve the efficiency and accuracy of image processing.

In this paper, we compute the stereo matching cost with a convolution neural network, and optimize our disparity map with a left-right consistency check, semiglobal matching, a median filter and a bilateral filter. We construct a hybrid data set with KITTI 2012 and KITTI 2015 stereo data sets to training, evaluating and testing our network model. Besides, we calibrate camera with Matlab and OpenCV, build our network with TensorFlow, and finally use OpenCV computer vision program library for three-dimensional reconstruction. In the meanwhile, we build the web server with Flask and Ngrok service, and build the client based on Qt for binocular data upload and result display.

Key Words: binocular stereo vision, stereo matching, convolutional neural network, TensorFlow, OpenCV

目　　录

第 1 章　绪　论

1.1　研究背景及意义

众所周知，自然界中的物体都是三维立体的，人类可以通过双眼产生的二维图像来获取三维世界信息，并对各种物体和信息进行分类和识别。但是一般的摄影系统只能把三维的物体以二维的形式保存和记录下来，丢失了大量的深度信息。为了让计算机也能像人类一样“看到”并理解这个世界，人类一直在深入地研究自身的视觉系统，并衍生出来计算机视觉这一学科。立体视觉是计算机视觉的一个重要分支，是一种从二维平面图像中恢复出三维深度信息的技术。

表 1-1　激光雷达供应商 Velodyne 部分产品列表

激光雷达			
特性	HDL-64	HDL-32	VLP-16
售价	8 万美元左右	2 万美元左右	7999 美元
激光束	64	32	16
范围	120m	100m	100m
精度	±2cm	±2cm	±3cm
数据类型	距离/密度	距离/校准发射率	距离/校准发射率
数据频率	1.3M 像素/秒	700,000 像素/秒	300,000 像素/秒
垂直角度	26.8°	40°	30°
水平角度	360°	360°	360°
功率	60W	12W	8W
体积	203*284mm	86*145mm	104*72mm
重量	15kg	1kg	0.83kg

立体视觉技术在军事、工业领域中有着广泛的应用背景。在军事方面，现代化的战争中，通过侦察卫星，单兵侦察等手段获取二维图像数据，利用立体视觉技术进行三维重构，模拟出三维的战场信息，可以为作战指挥提供强有力的信息支持。在工业领域中，立体视觉技术为机器人、无人驾驶汽车提供了导航、避障和物体识别等应用，特别在工

业控制领域，大量的应用如工业机器人、机械手臂等都离不开立体视觉技术的支持。在智能安防中，立体视觉技术提取的深度信息还能帮助传统监控摄像头实现环境信息的结构化，从而更快预警，提高智能识别的准确性。

虽然自二十世纪六十年代以来，不断有人提出不同的方法来提取二维图像中的三维深度信息。但是目前准确率较高的传统方法如激光雷达，毫米波雷达等，提取设备的繁琐复杂，成本高昂，且视角存在局限性，无法得到真正的普及，如表 1-1 所示。而设备简单的传统立体视觉方法，如单目，双目的立体视觉方法，受限于立体匹配算法的不成熟，又无法提供准确的视差数据。在无人驾驶，机器人导航等技术迅猛发展的今天，我们亟需设计一个能从成本低廉的设备中提取的准确率较高的立体视觉算法。

1.2 国内外研究现状

计算机立体视觉的开创性工作是从 20 世纪 60 年代中期开始的，美国麻省理工学院的 Robert 把 2 维图像分析推广到 3 维景物分析，标志着计算机立体视觉技术的诞生，并在随后的 20 年中迅速发展成一门新的学科[1]。特别是 20 世纪 70 年代末，Marr 等人通过对生理学、计算机学、物理学和心理学等领域的研究与概括，提出了计算机视觉理论框架，将二维图像信息与三维场景信息联系在一起，对立体视觉的发展产生了巨大影响[1-2]。后来的研究者在此基础上建立了从图像获取到最终的景物可视表面重建的完整体系[2]。

立体匹配是双目视觉技术的核心问题，也是立体视觉研究中的难点。立体匹配算法的精度和速度对于双目立体视觉系统的性能起着决定性的作用。国内外的许多研究者对立体匹配算法进行了深入的研究。

国内的立体匹配研究起步较晚，研究成果主要集中在算法应用方面，如北理工开发的双目视觉空间测距平台，可应用于穿戴计算机人机交互系统中,以手指尖作为特征匹配点，根据三角测距方法确定空间手指坐标；哈工大开发的移动视觉系统，实现了足球机器人的自动导航；浙大利用双目视觉对机器臂进行改进，实现了动态定位与检测。

国外的研究起步较早，相对领先国内研究。2002 年，Scharstein 等人首次提出一个比较系统的立体匹配理论框架，将一个双目视觉立体匹配算法分为四个步骤：匹配代价计算，代价聚合，优化，视差精化，并提出了一个立体视觉评测数据集，最终衍变成 Middlebury 数据集[3]。2004 年 Kong 等人提出可以通过平方和（SAD）来初始化匹配代价，并训练了一个模型来预测三个类别的概率分布：初始视差正确的，由于前景目标过

大导致初始视差不正确的，并且由于其他原因导致初始视差不正确的。预测概率被用来调整初始匹配代价[4]。2007 年，Zhang 等人提出了一种替代优化算法来估算马尔科夫随机场超参数的最优值[5]。2007 年，Scharstein 等人构建了一个新的 30 个立体对的数据集，并使用它来得到条件随机场的参数[6]。2008 年，Li 等人提出了一个带非参数代价函数的条件随机场模型，并使用结构化支持向量机来得到模型参数[8]。2012 年，Peris 等人通过 AD-Census 方法初始化匹配代价，并使用多类线性判别分析来得到计算所得的匹配代价到最终视差的映射[7]。2012 年，德国卡尔斯鲁厄理工学院和丰田美国技术研究院共同提出立体视觉评测数据集 KITTI，该数据集是目前国际上最大的自动驾驶场景下的计算机视觉算法评测数据集，KITTI 包含市区、乡村和高速公路等场景采集的真实图像数据，可用于评测立体图像，光流，视觉测距，3D 物体检测和 3D 跟踪等计算机视觉技术在车载环境下的性能，并在 2015 年进一步完善[12,13]。2013 年，Haeusler 等人使用了一种随机森林分类器来组合若干置信度度量方式[9]。2014 年，Spyropoulos 等人训练了一个随机森林分类器来预测匹配代价的置信度，并且使用预测结果作为马尔科夫随机场中的软约束来减少后处理的误差[10]。2015 年，Zbontar 等人通过训练 Siamese 网络来计算图像块的匹配代价，并通过基于交叉的代价聚合和半全局匹配来优化视差，取得了良好的测试结果[11]。

近年来，随着深度学习技术开始被广泛应用到各个领域中，其在图像处理领域的特征提取表现尤为优秀。目前在立体匹配算法的主流评测库 KITTI[12,13]中，排名靠前的算法几乎都开始使用了卷积神经网络计算初始匹配代价。

1.3 主要研究内容与章节安排

本篇文章以双目立体视觉中的核心步骤立体匹配算法作为主要研究内容。本文通过训练卷积神经网络进行小图像块的相似性度量，初始化匹配代价，并通过左右一致性检查、半全局匹配、中值滤波和双边滤波等一系列后处理步骤来精化视差图。本文使用的数据集是来自 KITTI 2012 和 KITTI 2015 的混合数据集。此外，本文结合 OpenCV 和 Matlab 工具箱进行离线标定，基于 TensorFlow 框架构建卷积神经网络模型，最后利用 OpenCV 计算机视觉库进行三维重建。同时，我们使用 Flask 框架和 Ngrok 服务搭建 Web 服务器，并基于 Qt 框架构建客户端程序实现双目数据上传和结果显示。根据以上研究内容和步骤，本文的主要架构如下：

第一章：绪论

该章节详细介绍了双目立体视觉立体匹配算法的研究背景及意义、国内外研究现状，并简单介绍了本文的主要内容和章节安排。

第二章：双目立体视觉原理

该章节主要介绍了双目立体视觉的相关原理，重点介绍了双目立体视觉的基本原理，简要介绍了立体匹配的经典步骤，约束条件以及实现难点。

第三章：立体匹配算法设计

该章节简单介绍了卷积神经网络的发展历程，随后重点介绍了本文使用的网络结构，并详细介绍了立体匹配的后处理步骤。

第四章：双目立体视觉实现与运行

该章节介绍了整个立体视觉实现的过程，简单说明了双目立体视觉的离线标定、双目矫正和三维重建过程，介绍了相关数据集的构造，详细描述了立体匹配算法的实现细节，最后简单介绍了服务端的搭建过程和客户端的 UI 设计。

第五章：总结与展望

该章节总结了本论文所做的工作，并提出了不足之处以及在后续设计中可以改进的内容。

第 2 章　双目立体匹配原理

2.1　双目立体视觉基本原理

双目立体视觉系统通过模拟人的双眼成像原理，一般由两摄像机从不同角度同时获取目标场景的两幅图像，并基于视差原理恢复出场景的三维几何信息，重建场景中物体的三维轮廓及位置。相对于单目和多目的立体视觉方法，双目立体视觉设备较简单，成本低廉，提取深度信息较方便，故被作为立体视觉技术实现的首选。本节主要介绍了双目立体视觉的实现原理。

2.1.1　摄像机成像模型

摄像机的成像模型是一个将三维空间物体投影成二维数字平面图像的过程，光线从空间物体开始，通过透镜到达摄像机的成像平面，从而形成摄像机捕捉到的数字图像。摄像机的成像模型可以通过图像坐标系、世界坐标系和摄像机坐标系之间的空间几何关系，由二维数字平面图像反推出三维空间信息。下面我们将详细介绍这三种坐标系及其相互关系。

(1) 图像坐标系

图像坐标系可以分为两种：图像物理坐标系和图像像素坐标系，他们的区别主要在于原点的位置不同。

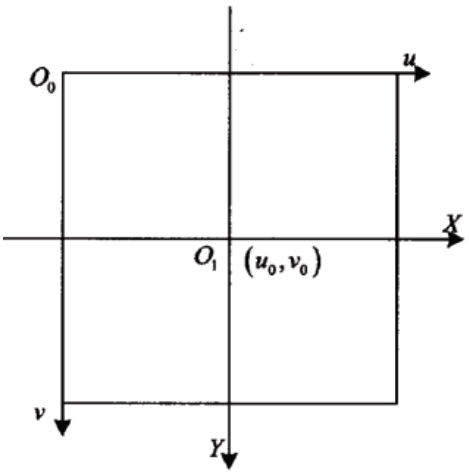

图 2-1　图像坐标系

如图 2-1 所示，图像像素坐标系原点O_0位于图像左上方的，它的 u，v 轴分别表示图像矩阵的列和行。而图像物理坐标系的原点O_1位于摄像机光轴和图像像素平面的交点，它的 X，Y 轴平行于 u，v 轴。设图像像素坐标系下的某点(u_0, v_0)，图像物理坐标系下每个像素在横轴 X 和纵轴 Y 上的物理尺寸为 dX，dY，则两坐标系之间的转换关系如公式 2-1 所示:

$$\begin{cases} u = \dfrac{X}{dX} + u_0 \\ v = \dfrac{Y}{dY} + v_0 \end{cases} \tag{2-1}$$

为方便后续计算，式 2-1 转换关系转换为矩阵表达形式如式 2-2 所示：

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \dfrac{1}{dX} & 0 & u_0 \\ 0 & \dfrac{1}{dY} & v_0 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{2-2}$$

其逆变换形式如下：

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} dX & 0 & -u_0 dX \\ 0 & dY & -v_{0dY} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2-3}$$

(2) 世界坐标系

为描述摄像机在真实场景中的相对位置信息，我们引入世界坐标系，如图 2-2 所示，O_w为世界坐标系的原点，坐标轴为X_w、X_w和Z_w。

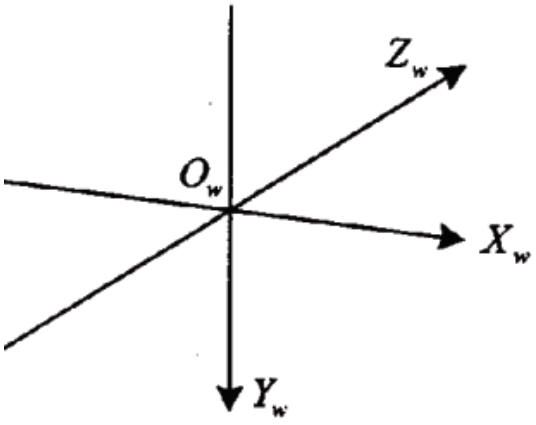

图 2-2　世界坐标系

(3) 摄像机坐标系

摄像机坐标系是联系图像坐标系和世界坐标系的桥梁。摄像机坐标系以摄像机镜头焦点为原点，摄像机光轴为 z 轴，其 x，y 轴平行于图像坐标系的 X，Y 轴。具体如图 2-3 所示，OO_1表示摄像机焦距。

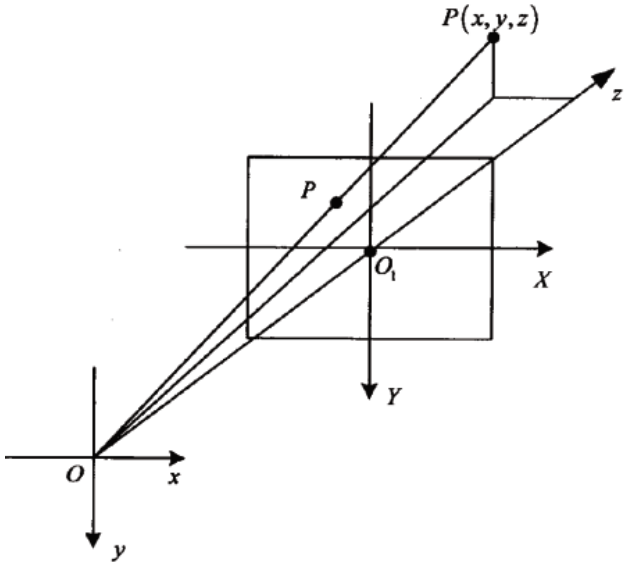

图 2-3　摄像机坐标系

摄像机坐标系和世界坐标系之间的转换需要经过矩阵平移和旋转操作，设某一像素点的摄像机坐标为$(x, y, z, 1)^T$，其对应的世界坐标系为$(X_w, Y_w, Z_w, 1)^T$，则两坐标系的关系如下：

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2-4}$$

其中，R 为3×3的正交单位旋转矩阵，T 为三维平移向量，0^T为$(0,0,0)$

2.1.2　对极几何

对极几何描述了两幅图像之间的视觉几何关系，与外部场景无关，只依赖于摄像机的内参数和左右摄像机的相对空间位置。对极几何是视图几何理论的基础，在双目视觉的双目矫正和三维恢复中有着广泛的应用。具体原理图如图 2-4 所示。

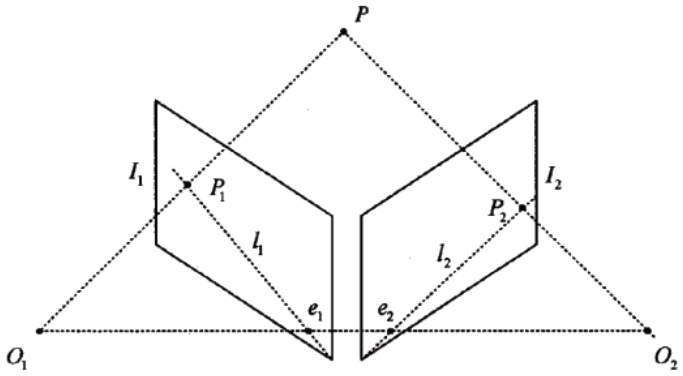

图 2-4　对极几何原理图

图中，O_1和O_2为左右两摄像机的光心，平面I_1和I_2是两摄像机的成像平面，P 点为现实空间中的某个待观测点，P_1和P_2为 P 点在两成像平面上的成像点。光心O_1O_2所在的直线和观测点 P 共同确定的平面为极平面，O_1和O_2在成像平面上的投影称为极点，极平面与两成像平面的交线为极线，极线l_1和l_2分别是P_2和P_1的外极线。根据对极几何原理，待匹配点的对应点一定在其对应的外极线上，即P_1点的匹配点，一定在其对应的外极线l_2上，P_2点的匹配点，一定在其对应的外极线l_1上。极线几何约束大大的减少了匹配点的搜索范围，降低了算法复杂度。

2.1.3　视差理论

假设存在一个理想的双目视觉模型，左右两摄像机平行摆放，摄像机内参一致，左右摄像机成像平面位于同一平面中。

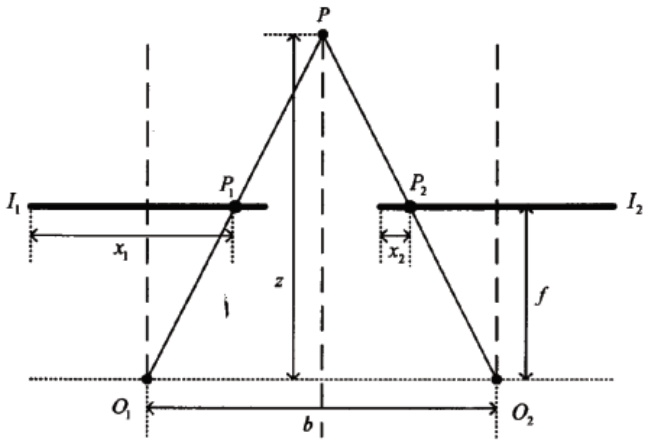

图 2-5　理想双目视觉模型

如图 2-5 所示，P 点为现实世界中的某观测点，b为两光心O_1和O_2的间距，z为点 P 的深度距离，两摄像机焦距为f。通过三角形相似原理，可以得出如下公式：

$$\frac{b}{z}=\frac{(b+x_2)-x_1}{z-f} \tag{2-5}$$

由公式 2-5 可以推导出深度距离为：

$$z=\frac{bf}{x_1-x_2}=\frac{bf}{d} \tag{2-6}$$

其中x_1-x_2为视差值d，f和b通过标定得到。这就是双目立体视觉的基本公式。我们常用视差图来表示立体匹配的结果，视差图中灰度值越小，目标点越远。

2.2 立体匹配的经典步骤

立体匹配算法是双目立体视觉技术的核心。立体匹配就是从左右摄像机获取的两幅图中找出每个像素的对应关系，从而提取出视差图。立体匹配算法的精度和速度常被用来评价立体匹配算法的性能。

立体匹配算法一般可以分为以下四个步骤：

(1) 匹配代价计算

立体匹配代价的计算即计算左右两像素之间的相似程度，根据代价函数的不同对应的匹配结果也会不同，常见的匹配代价计算方式包括灰度差的绝对值，灰度值的平方差等。

(2) 代价聚合

匹配代价的聚合通常是以待匹配点为中心，选择一定的区域作为聚合窗口，对窗口内的匹配代价进行叠加或平均来增强匹配代价的可靠性，提高匹配的准确度。目前常见的代价聚合方式有固定窗的代价聚合，基于交叉的代价聚合[14]等。

(3) 视差计算

视差计算通常采用胜者全拿的策略，即寻找使匹配代价C(p，d)最小的视差 d，如公式 2-7 所示。

$$\mathrm{D}(\mathbf{p})=\arg\min_{\mathrm{d}}\mathrm{C}(\mathbf{p},\mathrm{d}) \tag{2-7}$$

(4) 视差优化

视差优化的步骤是通过对计算所得的视差图进行错误校验，来得到精度更高的视差

图，通常会采用左右一致性检查，中值滤波和双边滤波等技术来消除视差图中的某些异常点，也可以通过半全局匹配[15]进行亚像素增强，来精化视差。

2.3 立体匹配的约束条件

为减少立体匹配的搜索范围，提高匹配精度与速度，我们通常需要参考某些约束条件，目前常见的约束条件有以下几种:

(1) 唯一性约束

左右两图像中的点一一对应，左图中的任何一点只能和右图中的一个点唯一对应。

(2) 相似性约束

左右图像中匹配的两个点具有相似的属性，如灰度，梯度变化，几何轮廓，颜色值等都具有相似性。

(3) 连续性约束

空间中的物体表面一般是连续的，除物体的边缘部分外，其他地方的视差值应该是一个缓慢变化的过程，可以以此来约束某点和领域像素的视差。

(4) 顺序性约束

立体匹配中某极线上的点，在对应极线上的匹配点的顺序是不变的。

(5) 视差范围约束

图像的视差值不可能无限大，在匹配时，仅需要搜索视差范围内的点即可。选择合适的视差范围可以提高匹配的精度和效率。

(6) 左右一致性约束

如果以左图为基准时，左图的点 p 在右图的匹配点为点 q，那么，以右图为基准时，右图的点 q 在左图的匹配点应为点 p。我们通过做左右一致性检查来消除视差图的毛刺。

(7) 极线约束

极线约束的原理已在 2.1.2 小节对极几何原理中详细介绍了，左图中的待匹配点在右图中的对应点一定在右图的极线上，匹配搜索时仅需要搜索该极线上的点即可。

2.4 立体匹配的难点

立体匹配算法经过不断的发展，已经有了极大的进步，但仍然面临众多挑战。目前立体匹配的难点主要集中在以下几个方面:

图 2-6　镜面区域

图 2-7　遮挡区域

(1) 镜面区域

由于双目视觉中的两摄像机是从不同角度拍摄的图像，所以会存在某些镜面区域因为光照问题而导致的灰度信息出现明显变化。如图 2-6 中光照所产生的镜面反射导致亮度、色调、饱和度等均出现变化。

(2) 遮挡区域

双目视觉中，同一场景在左右两幅图中并不完全一样，由于视线阻挡会出现遮挡区域，如图 2-7 所示，左图的部分区域被遮挡，在右图中无法观测到，自然也无法找到对应点。

(3) 弱纹理和重复纹理区域

弱纹理和重复纹理问题是立体匹配中的经典难题。由于目标场景的纹理特征单一或部分纹理特征非常相似导致匹配误差。如图 2-8 中，目标场景纹理特征不丰富，很难得到准确的视差值。

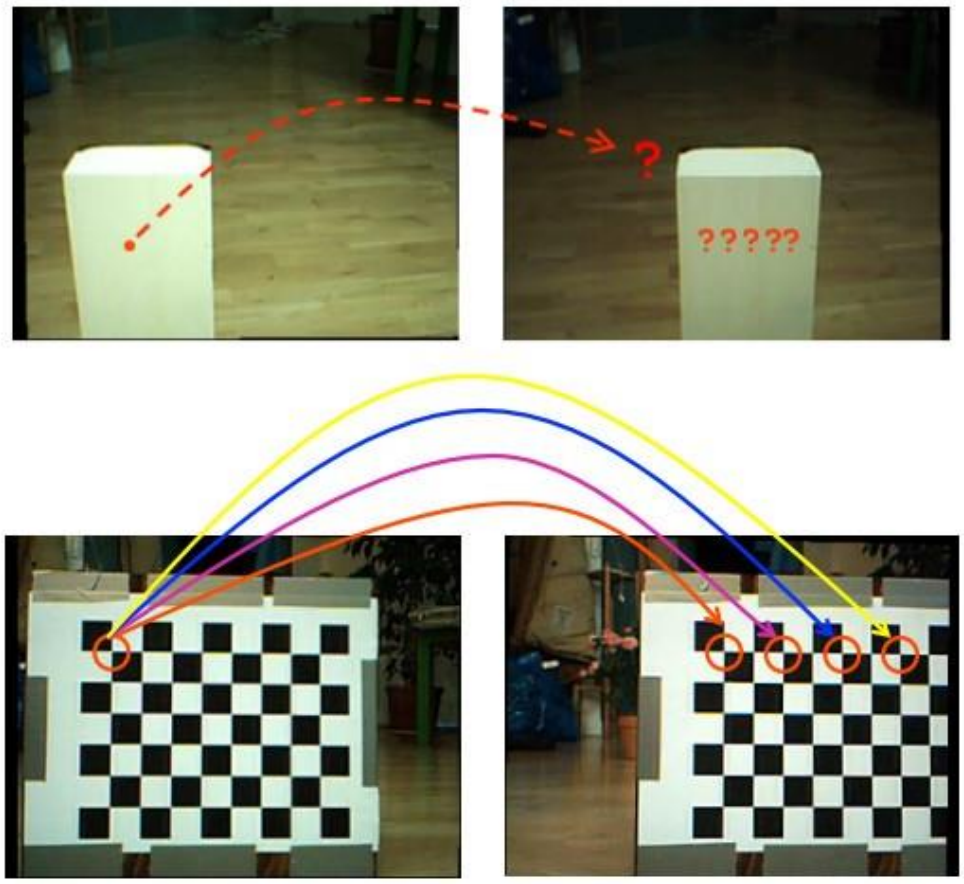

图 2-8　弱纹理和重复纹理区域

图 2-9　透明区域

(4) 透明区域

一般情况下，场景中的物体在边缘变化会比较剧烈，我们以此来判断物体边缘。但是也存在某些前景后景相似的情况，如出现透明物体时，这些区域的边缘灰度变化不大，导致所得视差的边缘模糊，甚至误匹配。

2.5 本章小结

本章节主要介绍了双目立体视觉的相关原理。首先本章详细介绍了双目立体视觉的基本原理，详细介绍了摄像机的成像模型、对极几何和视差理论。随后，本章对立体匹配的经典步骤，约束条件以及实现难点等做了概略说明。

第 3 章　立体匹配算法设计

3.1　卷积神经网络

卷积神经网络（Convolutional Neural Network, CNN）是当前深度学习技术中极具代表性的网络结构之一。卷积神经网络通过权值共享和局部连接的结构模拟生物神经网络，不仅避免了传统模式识别算法中复杂的特征提取过程，而且解决了神经网络中全部采用全连接导致参数量过大难以训练的问题。卷积神经网络的网络结构对平移、比例缩放、倾斜或者共他形式的变形具有高度不变性，其在图像识别、语音识别和自然语言处理等问题上表现尤为突出。

卷积神经网络的历史最早可以追溯到二十世纪六十年代。1962 年 Hubel 和 Wiesel 在研究猫的视觉皮层细胞时发现，每一个视觉神经元只会处理一小块区域的视觉图像，提出了感受野(Receptive Field)的概念，即猫的视觉系统是分级的，这种分级可以看成是逐层迭代、抽象的过程[19]。基于上述提出的感受野概念，日本学者 Fukushim 在 1984 年构建了神经认知机(Neocognitron) [20]。神经认知机是卷积神经网络的第一个实现网络，该网络中包含两类神经元，用来抽取特征的 S-cells，和用来抗形变的 C-cells，其中 S-cells 对应现在主流卷积神经网络中的卷积核滤波操作，而 C-cells 则对应激活函数、最大池化（Max-Pooling）等操作[20]。

随着后来研究者的不断改进，LeCun 等人在 1988 年结合 BP 算法实现了 LeNet-5 模型，其在数字识别领域的表现非常强大，在银行支票的手写识别应用中达到了 95.1%的识别精度[21]。

但是在接下来近十年的时间里，卷积神经网络的相关研究趋于停滞，主要原因有三点：一是当时互联网尚未广泛普及，很难取得足够的标签数据以训练网络；二是神经网络在进行训练时的计算量极其之大，以当时的硬件计算能力几乎不可能实现；三是包括 SVM 在内的浅层机器学习算法渐渐开始崭露头脚。直到 2006 年，Hinton 提出深度置信网络（DBN），使用一种贪心无监督训练方法来解决问题并取得了良好结果[22]。DBN（Deep Belief Networks）的训练方法降低了学习隐藏层参数的难度，并且该算法的训练时间和网络的大小和深度近乎线性关系[22]。CNN 重新回到研究者的视野中，并取得了长足发展。

随着大数据时代的到来和硬件性能的提升，研究者们不仅能够取得足够标签数据用于训练，而且能够借助图形处理单元（GPU）的强大运算能力快速的训练和优化模型。2012 年 Krizhevsky 等人凭借 AlexNet 网络赢得了 2012 ImageNet 竞赛的冠军（ImageNet 竞赛为 1000 类图片分类任务，难度较大），该网络将分类误差从 26%降到 15%，远远超越了之前的方法[23,24]。随着 AlexNet 的成功，卷积神经网络开始被广泛应用到各个领域中，例如谷歌将神经网络用于图片搜索引擎中，亚马逊将其用于商品推荐，Pinterest 将其用于个性化主页推送，还有打败李世石的 AlphaGo 也用到了卷积神经网络，可以说卷积神经网络在众多课题中都取得了突破性的成果。

随着卷积神经网络的深入发展，研究人员开始尝试在一些早已被充分研究的问题中应用神经网络。Zbontar 和 LeCun 在 2015 年提出使用卷积神经网络进行立体匹配[25]。他们使用相似和不相似的图像对构建二元分类数据集来进行有监督的训练[25]。卷积神经网络的输出用于初始化立体匹配代价，并配合一系列的后处理步骤：基于交叉的代价聚合，半全局匹配，左右一致性检查，亚像素增强，一个中值滤波器和一个双边滤波器来提高视差图的准确性[25]。他们的立体匹配算法在主流立体视觉评测库 KITTI 和 Middlebury 中表现优异，引起了广泛的关注。目前在 KITTI 的排行榜中排名前十的算法几乎都用到了卷积神经网络。

3.2 网络结构

鉴于卷积神经网络在图像处理领域的优秀表现，本文采用 Siamese 网络来比较图像块之间的相似度。Siamese 网络中包含两路结构相似的子网，它们分别训练，但共享各个层的参数，并在最后的非共享连接层进行相似度比较[26]。本文采用的网络结构如图 3-1 所示。

在图 3-1 网络结构中，两个子网由多个卷积层和整流线性单元（ReLu）组成。输入的左右图像块提取特征向量后通过计算它们之间的余弦相似性来获得相似度得分，其中余弦相似性的计算分为两步进行，归一化和点积。这样每个位置仅需要执行一次归一化操作，减少了运行时间。余弦相似性的计算如公式 3-1 所示。

$$similarity = \frac{\boldsymbol{A} \cdot \boldsymbol{B}}{\|\boldsymbol{A}\|\|\boldsymbol{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2}\sqrt{\sum_{i=1}^{n} {B_i}^2}} \tag{3-1}$$

其中 $\boldsymbol{A}$，$\boldsymbol{B}$ 分别表示左右特征向量。上式可以拆分为以下两步。

$$normal(X) = \frac{\boldsymbol{X}}{\|\boldsymbol{X}\|} = \frac{\boldsymbol{X}}{\sqrt{\sum_{i=1}^{n} {X_i}^2}} \tag{3-2}$$

$$similarity = normal(\boldsymbol{A}) \cdot normal(\boldsymbol{B}) \tag{3-3}$$

也就是上文所说的归一化和点积步骤。

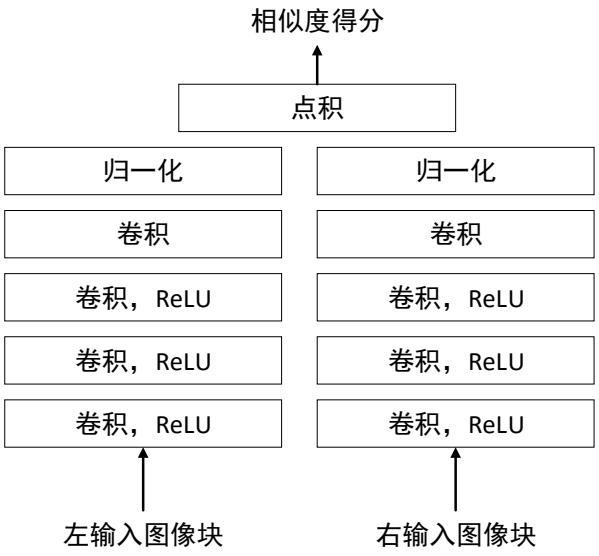

图 3-1　本文网络结构

本文选择铰链损失（Hinge loss）最小化来训练网络。损失函数如公式 3-4 所示。

$$L_{\text{Hinge}}(m, s_+, s_-) = \max\{0, m + s_- - s_+\} \tag{3-4}$$

其中s_+和s_-表示以同一图像位置为中心的积极和消极示例计算所得的相似度得分。s_+是积极示例的网络输出，s_-是消极示例的网络输出。余量 m 为正实数。当s_+大于s_-至少余量 m 时，网络损失为零。本文实验中设置余量 m 为 0.2。

Siamese 网络的输出用于初始化匹配代价，如公式 3-5 所示。

$$C_{CNN}(\boldsymbol{p}, d) = -s(< P^L(\boldsymbol{p}),\ P^R(\boldsymbol{p} - d) >) \tag{3-5}$$

其中$s(< P^L(\boldsymbol{p}),\ P^R(\boldsymbol{p} - d) >)$是当网络输入图像块$P^L(\boldsymbol{p})$和$P^R(\boldsymbol{p} - d)$时的输出。网络输出的相似度得分的负数表示匹配代价。

3.3　后处理步骤

卷积神经网络的原始输出不足以产生准确的视差图，特别是在某些低纹理和重复纹理区域存在明显的误差。通过应用一系列的后处理步骤可以有效的改善视差图的质量。

本文使用的后处理步骤主要参考了 Zbontar 和 LeCun 的论文[25]，其中包括左右一致性检查，半全局匹配，中值和双边滤波。

3.3.1 左右一致性检查

左右一致性检查是进行误匹配检测最为常见和有效的方式，可以有效的消除遮挡区域减少误匹配，得到精度更高的视差图。令D^L表示通过以左图为参考图得到的视差图，D^R表示以右图为参考图得到的视差图，在某些遮挡区域它们的视差是不同的。我们通过左右一致性检查来检测这些冲突，并应用以下规则来按顺序标记每个位置 p：

正确：当$d = D^L(p)$时，$|d - D^R(p-d)| \leq 1$

错误：当$d \neq D^L(p)$时，存在 d 使得$|d - D^R(p-d)| \leq 1$

遮挡：除上述两种情况外

对标记为“遮挡”的位置，我们从背景中获取新的视差值。我们向左遍历直到找到一个标记为“正确”的点并将该点的视差值作为遮挡点的视差。对于标记为“错误”的位置，我们以该点为中心的十六个不同方向中寻找最近的标记为“正确”的点，并取这些视差的中值作为错误点的视差。对于标记的“正确”的点，我们不进行操作。经过左右一致性检查后的视差图为D_{INT}。

3.3.2 半全局匹配

半全局匹配是 Hirschmuller 在 2008 年提出的，通过对视差图进行平滑约束来改善匹配代价[27]。半全局匹配通过使一个依赖于视差图 D 的能量函数 E(D)最小化来求解最优视差，能量函数形式如公式 3-6 所示。

$$\mathrm{E(D)} = \sum_{\boldsymbol{p}}\left(C_{CNN}\left(\boldsymbol{p}, D(\boldsymbol{p})\right) + \sum_{q\in N_{\boldsymbol{p}}} P_1 \cdot 1\{|D(\boldsymbol{p}) - D(\boldsymbol{q})| = 1\} + \sum_{q\in N_{\boldsymbol{p}}} P_2 \cdot 1\{|D(\boldsymbol{p}) - D(\boldsymbol{q})| > 1\}\right) \tag{3-6}$$

公式 3-6 中的粗体的 **p**，**q** 表示图中的某个像素点，$N_{\boldsymbol{p}}$表示以 **p** 点为中心的图像块，$C_{CNN}\left(\boldsymbol{p}, D(\boldsymbol{p})\right)$表示当 **p** 点的视差取$D(\boldsymbol{p})$时，该点的匹配代价。P1，P2 为惩罚系数，他们分别适用于与 **p** 点相邻像素的视差值和 p 的视差值相差为 1 和大于 1 的情况。1{·}表示如果函数中的参数为真，则返回 1，否则返回 0。

我们可以通过动态规划在单一方向上的能量函数最小化来实现 E(D)的最小化。虽然 Hirschmuller 选择了从十六个方向优化结果，但本文出于匹配速度上的考虑仅从水平

和垂直的四个方向上优化取平均数来最小化能量。为最小化方向 **r** 上的能量，可以得出公式 3-7 所示的具有递推关系的公式。

$$C_r(\boldsymbol{p},d)=C_{CNN}(\boldsymbol{p},d)+\min\begin{Bmatrix}C_r(\boldsymbol{p}-\boldsymbol{r},d),\\ C_r(\boldsymbol{p}-\boldsymbol{r},d-1)+P_1,\\ C_r(\boldsymbol{p}-\boldsymbol{r},d+1)+P_1,\\ \min\limits_k C_r(\boldsymbol{p}-\boldsymbol{r},k)+P_2\end{Bmatrix}-\min_k C_r(\boldsymbol{p}-\boldsymbol{r},k)) \tag{3-7}$$

公式 3-7 由三大项组成，第一项表示 Siamese 网络计算所得的匹配代价，第二项是四个候选值中的最小值。第一个候选值是之前所有像素视差值取 d 时的最小匹配代价，第二个候选值是之前所有像素视差值取 d-1 时的最小匹配代价与惩罚系数 P_1之和，第三个候选值是之前所有像素视差值取 d+1 时的最小匹配代价与惩罚系数 P_1之和，第四个候选值是之前所有像素视差值取其他值时的最小匹配代价与惩罚系数 P_2之和。第三项是之前所有像素视差值取其他值时的最小匹配代价，这一项是为了防止$C_r(\boldsymbol{p},d)$的值过大影响视差图的优化。

惩罚系数P_1和P_2是为优化图像边缘而设置的，令$D_1=|I^L(\boldsymbol{p})-I^L(\boldsymbol{p}-\boldsymbol{r})|$，$D_2=|I^R(\boldsymbol{p}-d)-I^R(\boldsymbol{p}-d-\boldsymbol{r})|$，$P_1$和$P_2$遵循以下规则：

如果D_1 <sgm_D，D_2 <sgm_D，则$P_1=$ sgm_P1, $P_2=$ sgm_P2

如果D_1 ≥sgm_D，D_2 ≥sgm_D，则$P_1=\frac{\text{sgm_P1}}{\text{sgm_Q2}}, P_2=\frac{\text{sgm_P2}}{\text{sgm_Q2}}$

其他情况，则$P_1=\frac{\text{sgm_P1}}{\text{sgm_Q1}}, P_2=\frac{\text{sgm_P2}}{\text{sgm_Q1}}$

在上述规则中超参数 sgm_P1 和 sgm_P2 为视差图中的不连续点设置了惩罚因子。如果D_1或D_2中的一个表示高图像梯度，惩罚因子为 sgm_Q1，如果D_1和D_2同时表示高图像梯度，则通过较大的 sgm_Q2 来惩罚。当计算两个垂直方向时，P_1的值还需要除以 sgm_V 惩罚因子。因为通常情况下垂直方向上的视差变化比水平方向更为频繁。

最终的匹配代价$C_{SGM}(\boldsymbol{p},d)$是所有四个方向取平均值计算得到的，如公式 3-8 所示。

$$C_{SGM}(\boldsymbol{p},d)=\frac{1}{4}\sum_{\boldsymbol{r}}C_r(\boldsymbol{p},d) \tag{3-8}$$

本小节涉及到的超参数有基础惩罚系数 sgm_P1、sgm_P2，惩罚因子 sgm_Q1、sgm_Q2、sgm_V，图像梯度阈值 sgm_D。

3.3.3 滤波

为消除视差图的某些异常点平滑视差图，本文通过中值滤波和双边滤波来进一步的精化视差。中值滤波的基本思想是用像素点邻域内灰度值的中值来代替该像素点的灰度值，在去除脉冲噪声、椒盐噪声的同时又保留了图像边缘细节。双边滤波是一种结合图像的空间邻近度和像素值相似度的折衷处理，同时考虑空域信息和灰度相似性，达到保边去噪的目的，具有简单、非迭代、局部的特点。

滤波过程中涉及到的超参数有中值滤波孔径线性尺寸 ksize，双边滤波邻域范围直径 d，颜色空间的标准差 sigmaColor 和坐标空间的标准差 sigmaSpace。

3.4 本章小结

本章节首先简单的介绍了卷积神经网络的发展历程，随后详细描述了本文使用的卷积神经网络结构和匹配代价的计算，最后本文详细介绍了立体匹配的后处理算法，重点介绍了左右一致性检查和半全局匹配算法，并简单说明了中值和双边滤波作用和涉及到的超参数。

第 4 章　立体视觉实现与运行

双目立体视觉的实现过程可分为离线标定、双目矫正、立体匹配和三维重建四个步骤。离线标定，双目矫正和三维重建均属于图像矫正部分，可借助 OpenCV 和 Matlab 等工具完成，不属于本文的核心研究内容。所以本章简要说明了图像矫正的流程，重点介绍立体匹配算法的实现过程。

4.1　图像矫正

在进行立体匹配算法研究时，我们的数据集中都是经过矫正的图像，可以直接用来实验。但是在实际应用中，我们的摄像机并不能如理想的双目视觉模型那样处于绝对水平的位置上，而且普通的摄像机往往存在一定程度的光学畸变，所以我们需要对图像进行离线标定，并在实际运行过程中进行双目矫正。

4.1.1　双目摄像机选取

本文使用的双目摄像机是由两个单目摄像机组成的，如图 4-1 所示，它们的配置如表 4-1 所示。

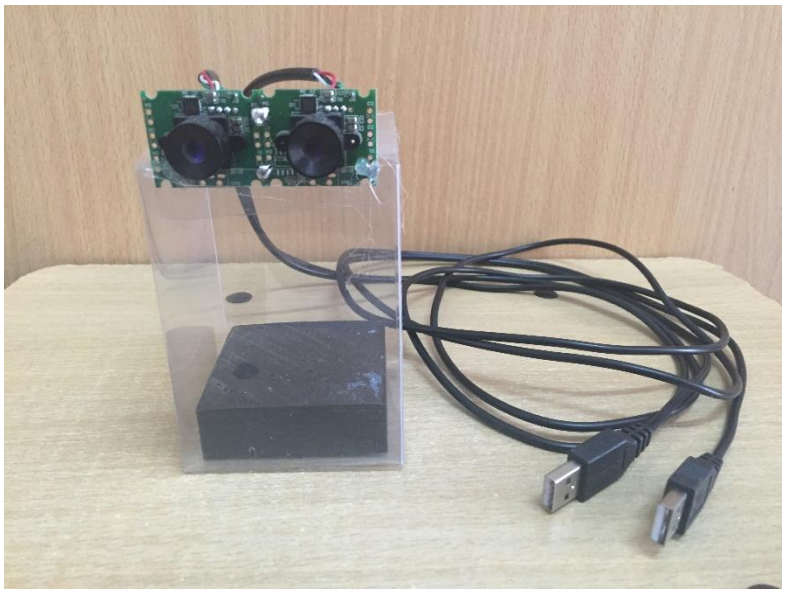

图 4-1　双目摄像机

表 4-1　摄像机配置参数

USB 接口	2.0 高速传输	视角	50 度
最大分辨率	640×480	摄像头尺寸	30 mm×25 mm
输出格式	YUV	焦距	2.8 mm

4.1.2　摄像机离线标定

摄像机的离线标定通常有两种方式可以实现：第一种是直接利用 OpenCV 图像处理库[28]的内置函数进行标定；第二种是通过 Matlab 的立体标定工具 Stereo Camera Calibrator 进行离线标定。由于第一种方法通常会产生过于夸张的矫正效果，所以文本采用第二种方式。

摄像机离线标定通常采用棋盘标定法实现。本文使用的棋盘如图 4-2 所示。

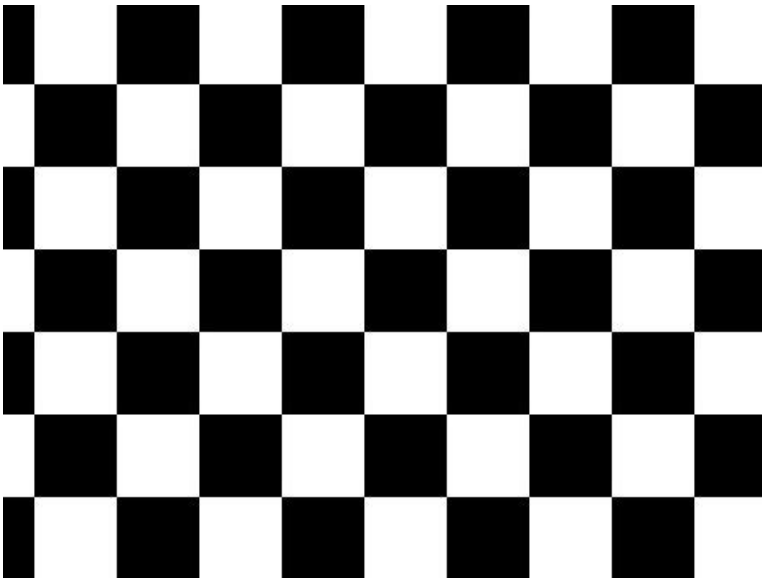

图 4-2　棋盘图

棋盘由黑白相间的正方形组成，为达到最好的标定效果，棋盘的长和宽的方格数必须要有一个为偶数，一个为奇数。本文的棋盘方格数为 6×9，实际打印后每个正方形的边长为 29mm，棋盘被固定在一硬纸箱底部。本文选取了 15 组不同的图像对进行标定，图像对左右图如 4-3 所示。

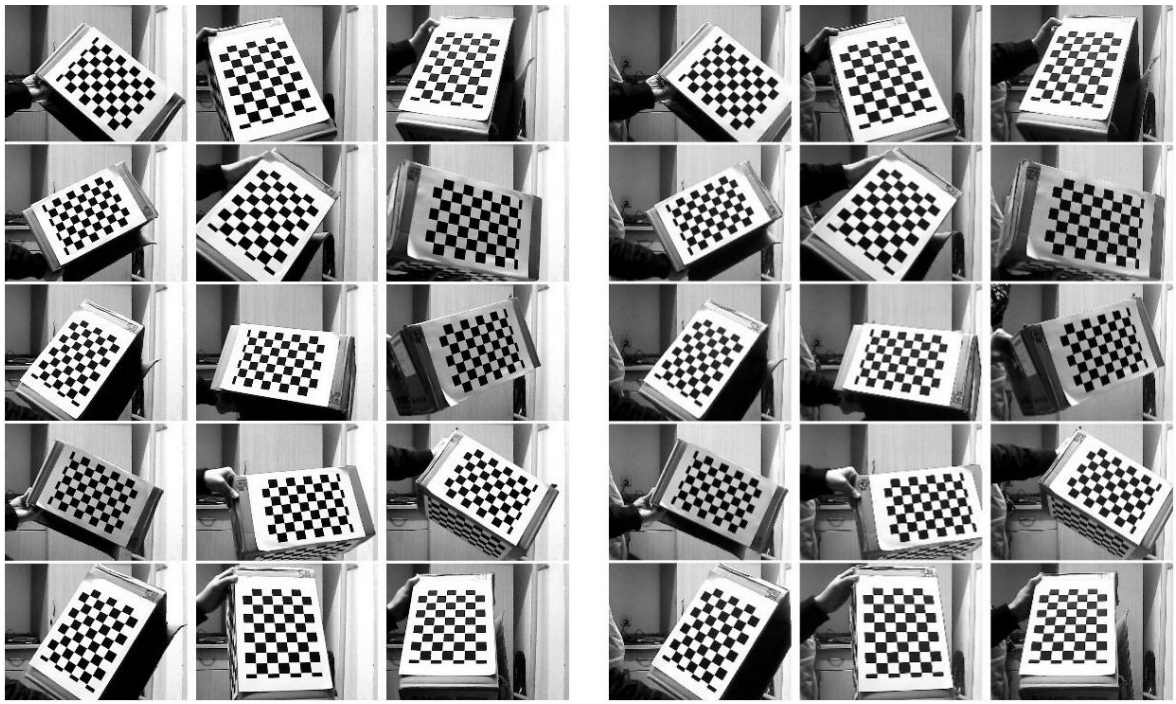

图 4-3　左起依次为：棋盘图像左图，棋盘图像右图

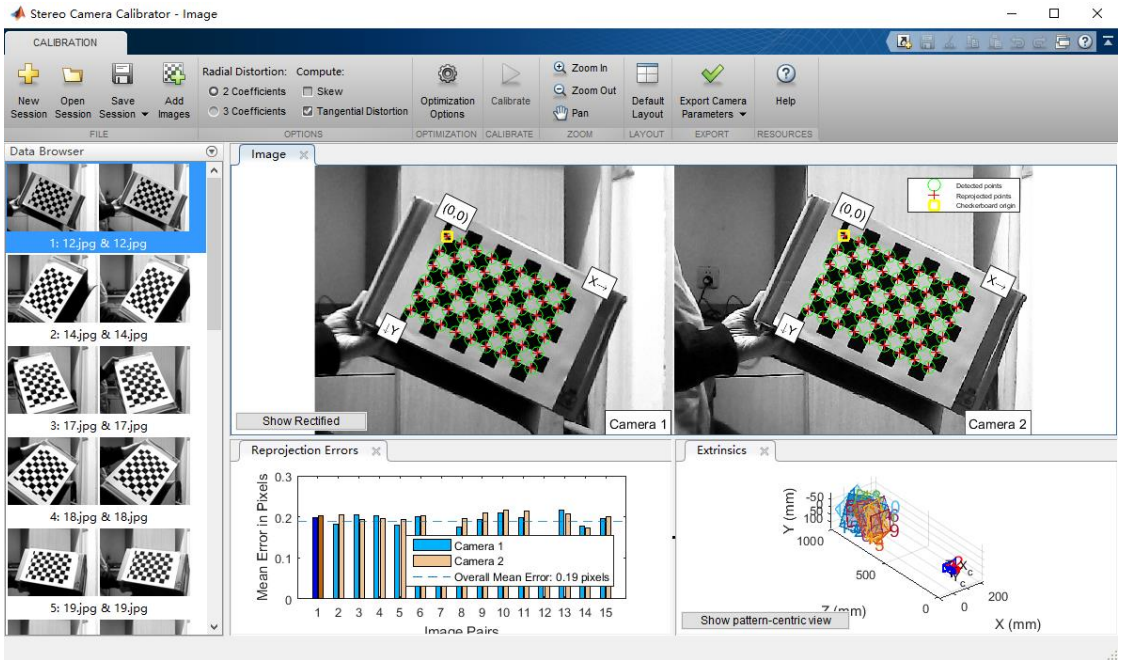

图 4-4　Matlab Stereo Camera Calibrator 操作界面

Matlab 的标定和矫正是一个自动化的过程，矫正过程中会自动修剪，只保留左右图像的共同区域。软件界面如图 4-4 所示，矫正前后的图像对如图 4-5 所示，矫正误差如图 4-6 所示，平均矫正误差为 0.19 个像素，基本满足实际应用需求。

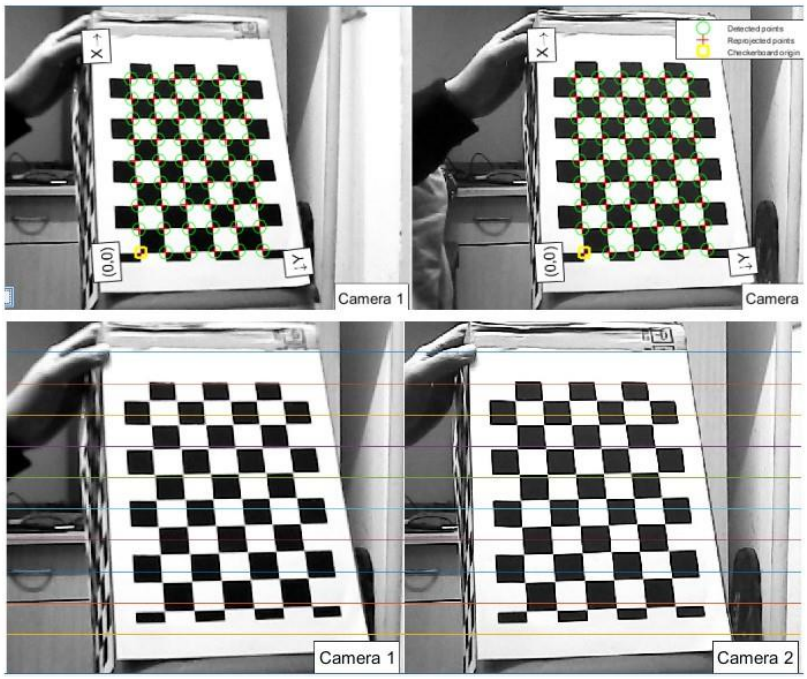

图 4-5　矫正前后图像对:从上到下依次为矫正前，矫正后

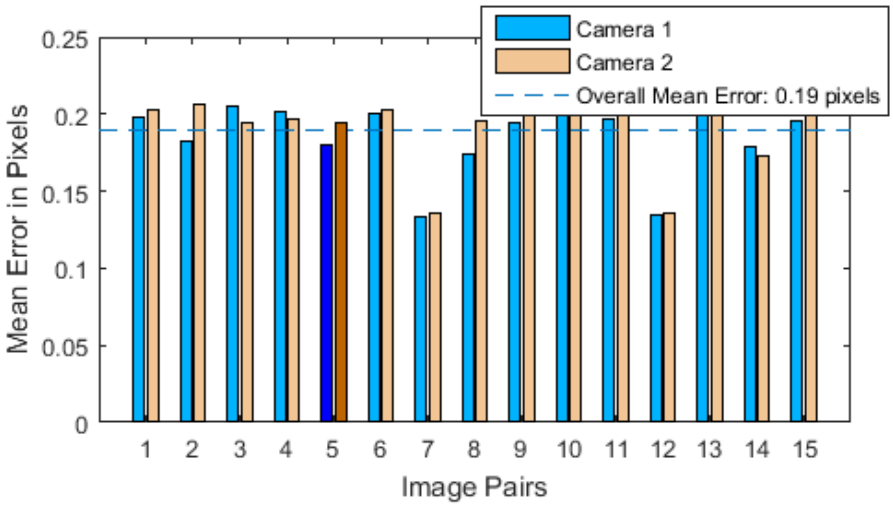

图 4-6　矫正误差

4.1.3　双目矫正与三维重建

经过 Matlab 离线标定所得的矫正参数，可以通过 OpenCV 的内置函数进行双目矫正和三维重建[28]。本文的矫正参数如图 4-7 所示，通过双目矫正的图像基本满足左右图的匹配点基本在同一水平线上。

```
## IntrinsicMa
left_camera_mat = np.array([[1123.7632, 0., 243.0600],
                            [0., 1121.3437, 220.1881],
                            [0., 0., 1.]])

## RadialDistortion and TangentialDistortion
left_distortion = np.array([[-0.1573, 0.6994, 0.0048, -0.0020, 0.]])

right_camera_mat = np.array([[1131.3162, 0, 357.7593],
                             [0., 1130.4406, 267.6306],
                             [0., 0., 1.]])

right_distortion = np.array([[-0.1746, 0.8458, 0.0068, 0.0003, 0.]])

## RotationOfCamera2
R = np.array([[0.9999, 0.0056, -0.0104],
              [-0.0060, 0.9991, -0.0430],
              [0.0101, 0.0430, 0.9990]])

## TranslationOfCamera2
T = np.array([-29.4093, -0.1669, -0.4104])

size = (640, 480)  ##长，宽
```

图 4-7　矫正参数

4.2　立体视觉数据集

高精度的立体视觉数据集对卷积神经网络的构建非常重要，优质的标签数据可以更好的训练模型。KITTI 数据集和 Middlebury 数据集是目前主流的立体视觉数据集。

4.2.1　KITTI 数据集

KITTI 立体视觉数据集[12,13]由德国卡尔斯鲁厄理工学院和丰田美国技术研究院联合创办，是目前国际上最大的自动驾驶场景下的计算机视觉算法评测数据集。该数据集可以用于评测立体图像，光流，3D 物体检测和 3D 追踪等计算机视觉技术。

数据集由安装在汽车车顶上两个相隔约 54 厘米的摄像机拍摄所得并经整流的灰度图像对构成。数据集中包含市区、乡村和高速公路等场景采集的真实图像数据，每张图像中最多达 15 辆车和 30 个行人，还有各种程度的遮挡与截断。原始数据集由 389 对立体图像和光流图，39.2 km 视觉测距序列以及超过 20 万 3D 标注物体的图像组成。图像分辨率为 1240×376。旋转激光扫描仪安装在左侧相机后面，记录地面的真实深度，标记了约 30％的图像像素。

数据集中包含训练数据集和测试数据集。测试数据集的地面真实视差并未公开，而是提供了一个在线排行榜供研究人员在测试集上评估他们的算法。每隔三天允许提交一次。错误率是计算那些真实视差和预测视差相差超过三个像素的像素的比例。这就意味

着，视差误差为 3 厘米则转换成物理距离为距离相机 2 米，而 80 厘米的误差，就是距离相机 10 米。KITTI 的每个子数据集中还提供了开发工具，用于视差数据的提取，保存以及评估，主要有 C++和 Matlab 两个版本。

目前共发布了两个 KITTI 立体视觉数据子集：KITTI 2012 和 KITTI 2015。2012 年的数据集包含 194 个训练图和 195 个测试图像，2015 年的数据集包含 200 个训练图和 200 个测试图。较新的数据集引入了一个微妙但重要的变化：运动车辆是密集标记的，并且汽车玻璃也被包括在评估中。

4.2.2 Middlebury 数据集

Middlebury 立体视觉数据集[3,16,17,18]最早由 Scharstein 等人在 2002 年提出，数据集中的图像对来自光照可控的室内场景。数据集通过结构光来测量真实视差，视差的密度和精度都比 KITTI 数据集更好。该数据集分别在 2001 年，2003 年，2005 年，2006 年和 2014 年发布了共五个独立的数据集。

在 2005 年，2006 年和 2014 年的数据集中，每一个场景都是根据多种光照条件和快门曝光方式拍摄的，其中有一组典型的图像对是同一个场景在四种光照条件和七种快门曝光方式下拍摄的共计 28 张图片。

该数据集也有一个类似 KITTI 的在线排行榜，显示了所有提交算法的排名表。参与者只有一次提交结果的机会，且需要通过作者审核。测试数据集包含 15 幅来自 2005 年和 2014 年数据集的图像。测试数据集提供全分辨率，半分辨率和四分之一分辨率的测试图像，全分辨率图像的分辨率约为 3000×2000。误差率的计算基于全分辨率，如果算法输出的是半分辨率或四分之一分辨率的视差图，需要在计算错误率之前上采样。

误差率通过计算真实视差与预测视差相差超过两个像素的像素点的百分比得到，使用半分辨率计算视差图时，错误容限仅为一个像素。在默认情况下，评估服务器上的误差率仅仅计算非遮挡的像素点。而在最终在线报告上的误差率为十五对测试图像的加权平均，权重由数据集的作者设置。

与 KITTI 数据集相比，Middlebury 在计算全分辨率时对实验计算机内存要求更高，图片较少，部分数据子集图像矫正并没有严格对应，即使通过 OpenCV 库中的标准矫正程序也会在 Middlebury 中造成最多九个像素的垂直视差错误[18]。

4.2.3 构造数据集

由于实验条件的限制，本文使用 KITTI 2012 和 KITTI 2015 数据集来构造实验数据集。本文从 KITTI 2012 和 KITTI 2015 中分别随机选出 162 张和 168 张图片构成训练数据集。KITTI 2012 和 KITTI 2015 剩余图片混合后，随机挑出 32 张构成验证数据集，混合数据集剩余的 32 张构成测试数据集。最终训练数据集，验证数据集和测试数据集的比例约为 10:1:1。

训练数据集仍需要进一步预处理转换成可训练的块数据。本文在训练数据集中每一个非零视差的图像位置处提取一对积极的和一对消极的示例。这种做法是为了确保训练数据集中包含相等数量的正例和负例。以下部分是积极示例和消极示例的具体提取步骤。

用$< P_{n\times n}^{L}(\mathrm{p}), P_{n\times n}^{R}(\mathrm{q}) >$表示一对图像块，其中$P_{n\times n}^{L}(\mathrm{p})$为左图中以位置 p=（x，y）为中心的$\mathrm{n}\times\mathrm{n}$图像块，$P_{n\times n}^{R}(\mathrm{q})$是右图中以位置 q 为中心的$\mathrm{n}\times\mathrm{n}$图像块，d 表示位置 p 处的正确视差。一对积极示例的获取是通过将右图像块的中心设置为

$$\mathrm{q} = (\mathrm{x} - \mathrm{d} + o_{pos}, \mathrm{y})$$

其中o_{pos}是从区间[−dataset_pos, dataset_pos]中随机挑选的。其中o_{pos}不为零是因为训练网络时，有一定程度的噪声可以降低网络的过拟合。

一对消极示例的获取是通过将右图像块的中心设置为

$$\mathrm{q} = (\mathrm{x} - \mathrm{d} + o_{neg}, \mathrm{y})$$

其中o_{neg}是从区间[dataset_neg_low, dataset_neg_high]或它的原点对应区间[−dataset_neg_high, −dataset_neg_low]中选出的。随机偏移o_{neg}确保了生成的图像块以不同与前者的 3D 点为中心。

积极示例和消极示例的图像块还需要通过将图像的像素强度值减去它们的平均值再除以标准差来归一化。本文将归一化后的积极示例和消极示例随机混合在一起，以 1:1 的比例构成一批 128 对的图像块，这就是我们最终的训练数据集。

构造数据集过程中的超参数包括积极示例的偏移（dataset_pos），消极示例的偏移（dataset_neg_low, dataset_neg_high）以及图像块的大小（input_patch_size）。

4.3 立体匹配

4.3.1 实验环境

随着近年来卷积神经网络的发展，涌现出了大量的深度学习框架，目前常见的有

Caffe, TensorFlow, MXNet, Torch, Theano 等五个主流框架[29,30,31]，这些框架的优缺点比较如表 4-2 所示。

表 4-2　深度学习框架比较

框架	开发语言	速度	预训练模型	RNN/CNN	多 GPU
Caffe	C++/Python	快	支持	CNN	支持
TensorFlow	Python/C++	中等	支持(Inception)	支持(最好)	支持(最好)
MXNet	C++/Python/R	快	支持	支持	支持
Torch	Lua/C	快	支持	支持	支持
Theano	Python/C++	中等	支持(Lasagne)	支持	支持

实验主要使用 Python 作为开发语言，需要 GPU 加速运算，综合考虑立体匹配算法的需求，本文选择 TensorFlow 作为模型训练框架。本文实验环境为 TensorFlow 框架，CentOs 系统，硬件: CPU(8 cores): Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz，Mem: 15990MB，GPU：NVIDIA Corporation GM200 [GeForce GTX TITAN X]，NVIDIA Corporation GK208 [GeForce GT 730]。

4.3.2　模型训练

本文使用 4.2.3 小节所说的训练数据集训练神经网络。训练期间，每次输入一批 128 对数量相等的积极和消极示例图像对。

表 4-3　子网卷积层参数

网络层	卷积核，特征映射
子网卷积层+ReLu	3×3，64
子网卷积层+ReLu	3×3，64
子网卷积层+ReLu	3×3，64
子网卷积	3×3，64

本文使用动量项为 0.9 的动量优化器来减少铰链损失。本文共训练了 20 次迭代，初始学习率为 0.001，在一次迭代后改为 0.0002，第 10 次迭代后改为 0.00002。训练中为了防止过拟合，本文还引入了权值衰减，衰减系数为 0.005，初始学习率和学习率的

衰减计划是通过交叉验证优化得到的。本文网络的卷积层参数如表 4-3 所示。

4.3.3 立体匹配细节

立体匹配的初始视差图 D(**p**)采用胜者全拿的策略得到，即对每个像素点 **p** 寻找使匹配代价$C_{CNN}(\boldsymbol{p},d)$最小的视差值 d，如公式 4-1 所示。

$$\mathrm{D}(\mathbf{p})=\arg\min_{d} C_{CNN}(\boldsymbol{p},d) \tag{4-1}$$

其中匹配代价通过 Siamese 网络的输出初始化得到。考虑到计算量较大，本文通过以下三个细节提高算法效率：

- 计算匹配代价时执行 2.3 节所说的视差范围约束，即每个像素点仅计算视差范围内的匹配代价。因为本文使用的数据集中的最大视差值为 228，所以本文视差范围为 228。
- 每个像素点的子网部分仅计算一次，而不是对每个视差值重新计算，即在计算匹配代价前先将每个像素点的子网部分计算一次，子网输出保存后用于后续的匹配代价计算
- 在运行期间，本文希望输入的是一整张的图片，但由于 GPU 内存的限制，本文每次仅输入一行图像数据（约 1220 个像素点）进入网络，并同时计算以左图为参考图的匹配代价和以右图为参考图的匹配代价，这样子网部分可以重复使用，节省运算时间。

运行过程中，每个输入 Siamese 网络的图像对都需要通过和训练数据一样的归一化处理，即将图像的像素强度值减去它们的平均值再除以标准差。

表 4-4 超参数

超参数	值	超参数	值
input_patch_size	9×9	ksize	5
dataset_pos	1	sgm_P1	3.5
dataset_neg_low	4	sgm_P2	223.0
dataset_neg_high	10	sgm_Q1	2.0
d	40	sgm_Q2	4.0
sigmaColor	5	sgm_V	1.75
sigmaSpace	5	sgm_D	0.02

验证数据集和测试数据集中的图片已经过双目矫正，可以直接计算视差图，但实际摄像头采集的图片存在一定程度的畸变，所以需要进行图像矫正，矫正所得的参数还可用于三维恢复。

本文通过公式 4-1 所说的胜者全拿（WTA）策略得到以左图为参考得到的视差图D^L和以右图为参考得到的视差图D^R视差图。D^L和D^R通过左右一致性检查得到视差图D_{INT}。D_{INT} 通过公式 4-2 所示的亚像素增强获得新的视差图$D_{SE}(\boldsymbol{p})$。最后通过一个中值滤波和双边滤波得到了我们的最终视差图。立体匹配流程图如图 4-8 所示。

$$D_{SE}(\boldsymbol{p}) = d - \frac{C_+ - C_-}{2(C_+ - 2C + C_-)} \tag{4-2}$$

式中 d=$D_{INT}(\boldsymbol{p})$，$C_- = C_{SGM}(\boldsymbol{p}, d-1)$，C=$C_{SGM}(\boldsymbol{p}, d)$，$C_+ = C_{SGM}(\boldsymbol{p}, d+1)$。

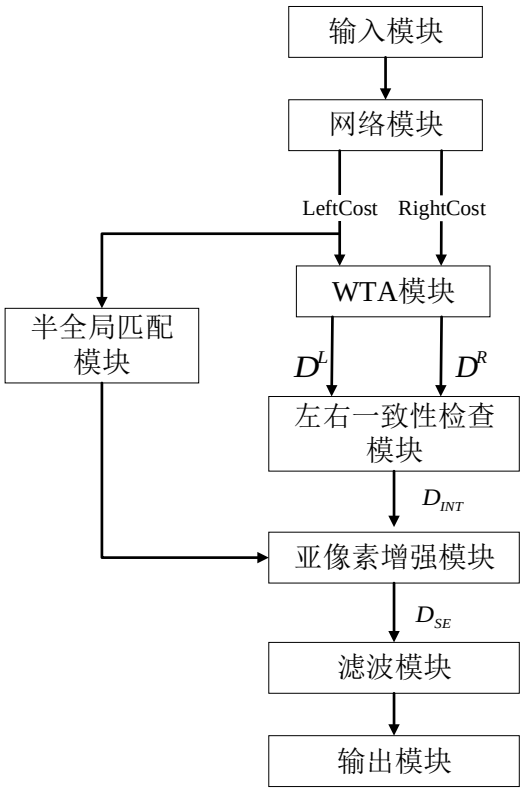

图 4-8　立体匹配过程框图

本文采用的超参数如表 4-4 所示，通过不同参数在验证数据集中的不同表现手动调整和简单的脚本优化得到。在测试数据集中，本文的立体匹配算法的视差误差仅为 8.0%，基本满足实际应用需求，视差图如图 4-9 所示，误差计算方法与 KITTI 立体视觉评测方式相同，如公式 4-3 所示。

$$\text{error} = \frac{N_1}{N_2} \tag{4-3}$$

其中N_1表示视差绝对误差超过 3 个像素的点个数，N_2表示参考视差图中有视差值的点个数。

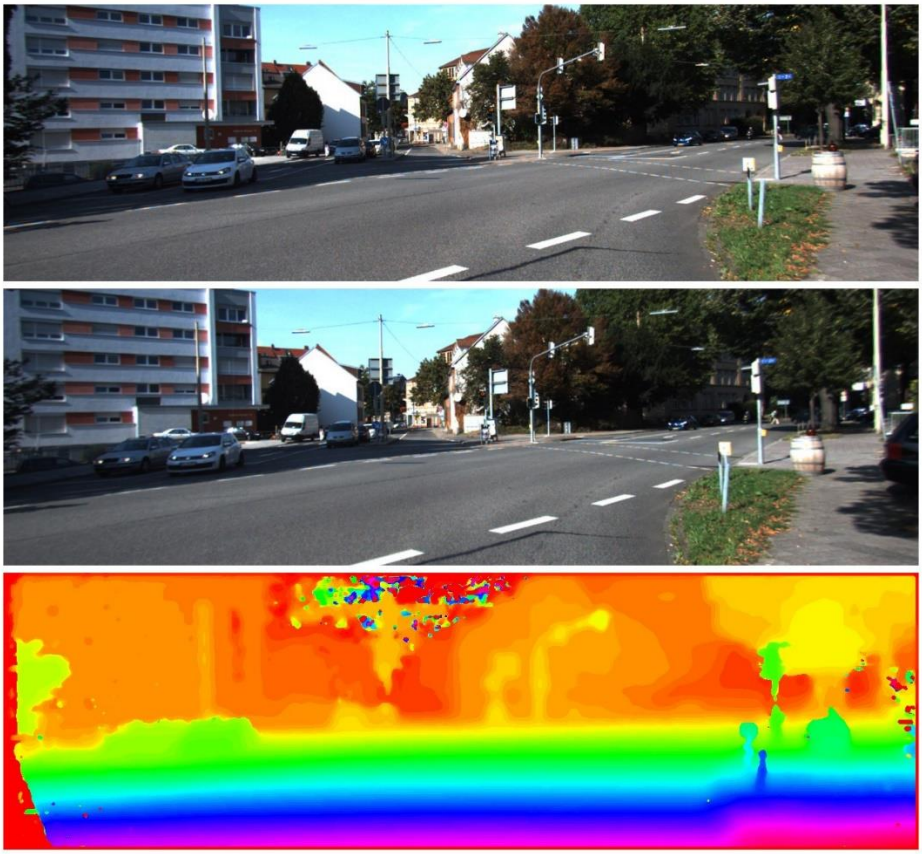

图 4-9　从上到下依次为：左图，右图，视差图

4.4　服务器搭建与客户端设计

4.4.1　服务端搭建

本文的立体匹配代价计算需要 GPU 加速计算，由于实验中使用的 GPU 服务器为内网服务器，所以本文需要搭建 Ngrok 服务实现内网穿透并通过 Flask 搭建 Web 服务器。

Ngrok 是目前常见的反向代理隧道工具，通过在内网主机和公网转接服务器之间建立安全隧道，让运行内网主机上的服务可以暴露给公网，即内网穿透。本文实验使用的是国内免费 Ngrok 服务器 ITTun.com，Ngrok 的配置信息如图 4-10 所示。

随着近年来互联网的发展，涌现出众多优秀的 Web 服务器开源框架，其中较为成

熟且应用较广的有 Nginx，Apache，Django，Flask 等。Flask 是一个轻量级的 Python 服务器框架，结构简单灵活可扩展性强，结合本文程序主要使用 Python 语言开发，所以本文选择 Flask 框架搭建 Web 服务器，便于后续集成。

```
server_addr: "ittun.com:36415"
trust_host_root_certs: false
tunnels:
        web:
           subdomain: "magic"
           proto:
            http: 172.0.0.1:7000
            https: 172.0.0.1:7000
```

图 4-10　Ngrok 配置信息

4.4.2　客户端设计

本文实验中的客户端是基于 Qt 框架搭建的。Qt 框架是一个面向对象的跨平台图形用户界面库，开发套件功能强大，应用广泛。本文通过 Qt 框架在 Python 的接口 PyQt 进行开发。初始界面如图 4-11，包含“Start”、“Capture”、“UpLoad”、“Disparity”四个按钮，“Left_Camera”，“Right_Camera”两个图像显示屏以及一个文本提示框。

实验中点击“Start”按钮启动连接的双目摄像头，开启视频录制，如图 4-12 所示。当出现视频中目标物体后，可以点击“Capture”按钮截取目标图片，同时进行双目矫正，如图 4-13 所示。确认图片后，可带点击“UpLoad”按钮上传图片至服务端，上传成功提示框显示“UpLoad Success”，否则显示“UpLoad Failed”，如图 4-14 所示。待服务端计算完毕后，可点击“Disparity”按钮，获取视差图片，获取的视差图会开启另一窗口显示，如图 4-15 所示，显示的视差图是由初始视差图归一化后经过 OpenCV 的 HSV 伪彩色处理得到[28]，点击视差图的某一位置，可以获取该位置的深度信息（以毫米为单位），如图 4-16 为先后点击图中两个水瓶的位置显示的情况，左侧水瓶约为 1300 毫米，右侧水平约为 1700 毫米，深度信息与实际基本相同。

图 4-11　初始页面

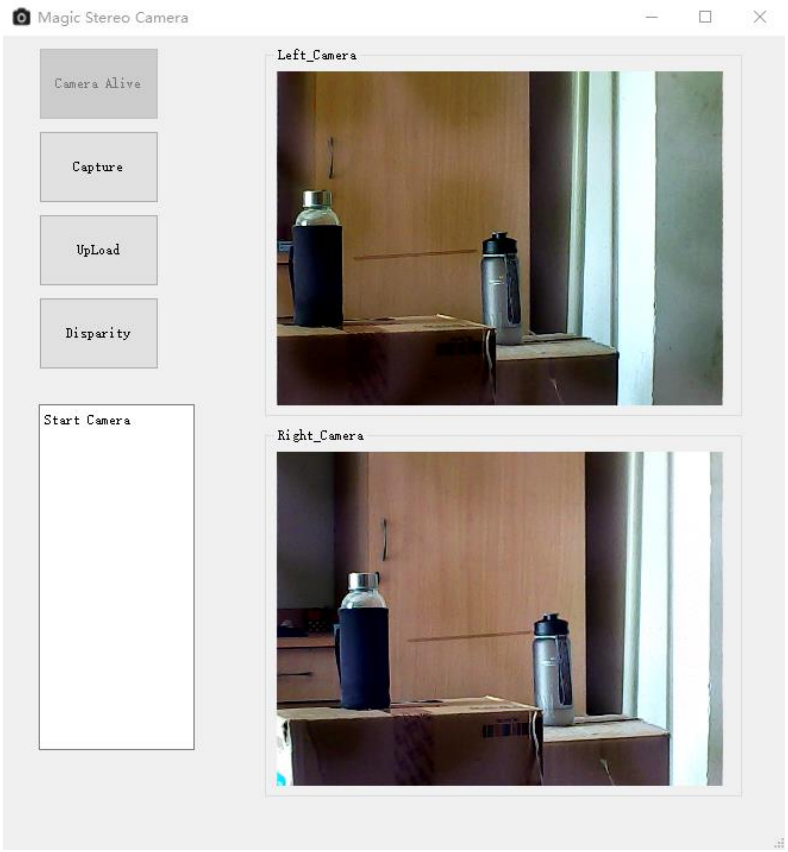

图 4-12　开启视频录制

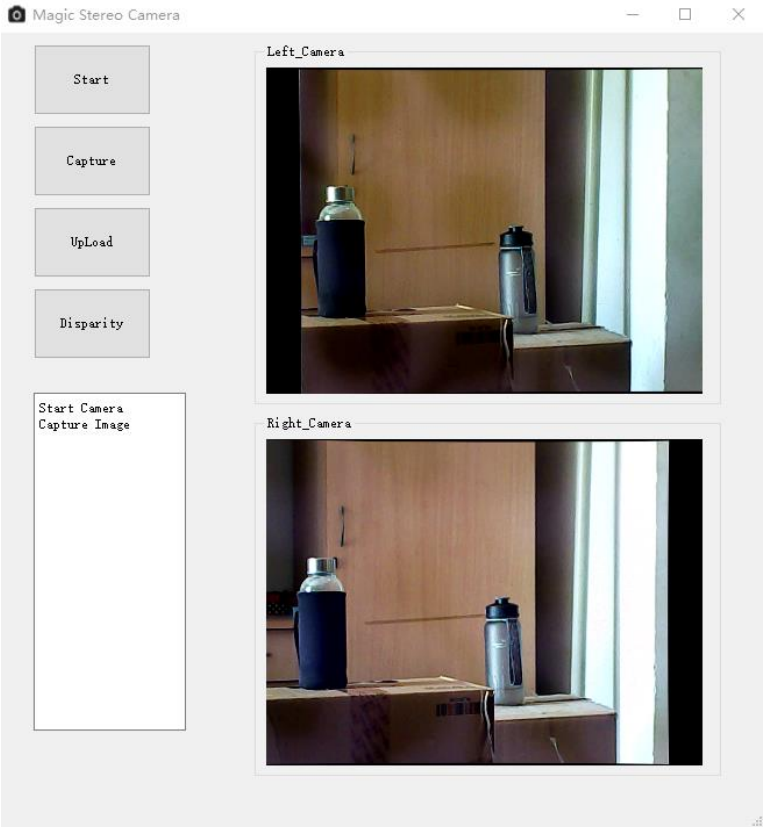

图 4-13　截取目标图片并矫正

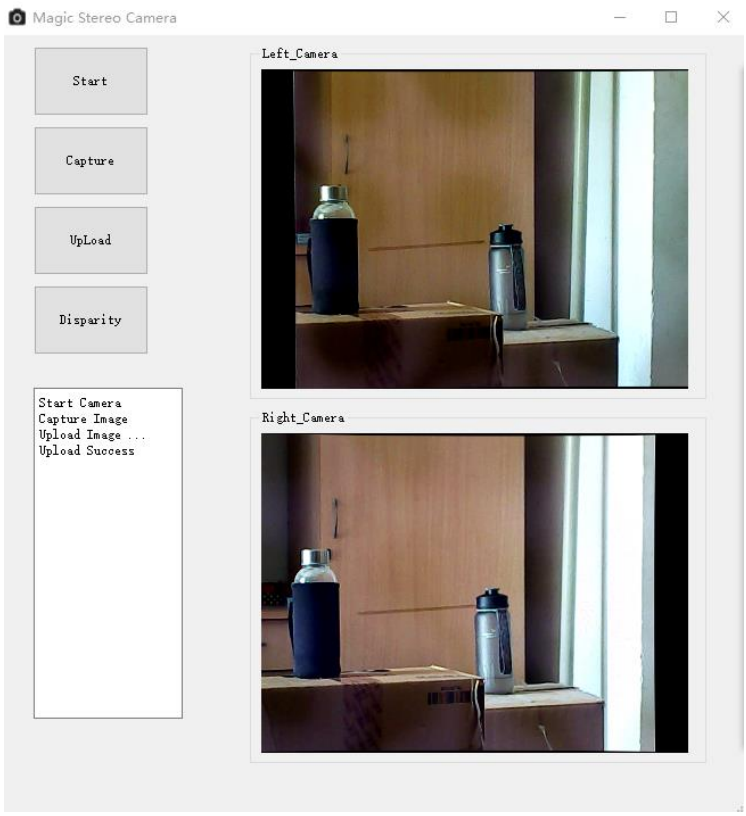

图 4-14　上传图片

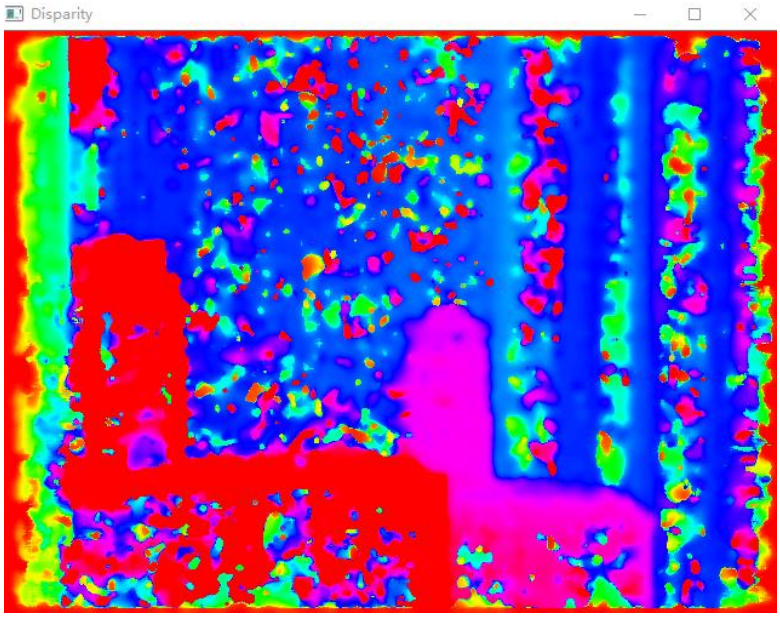

图 4-15　所得的视差图

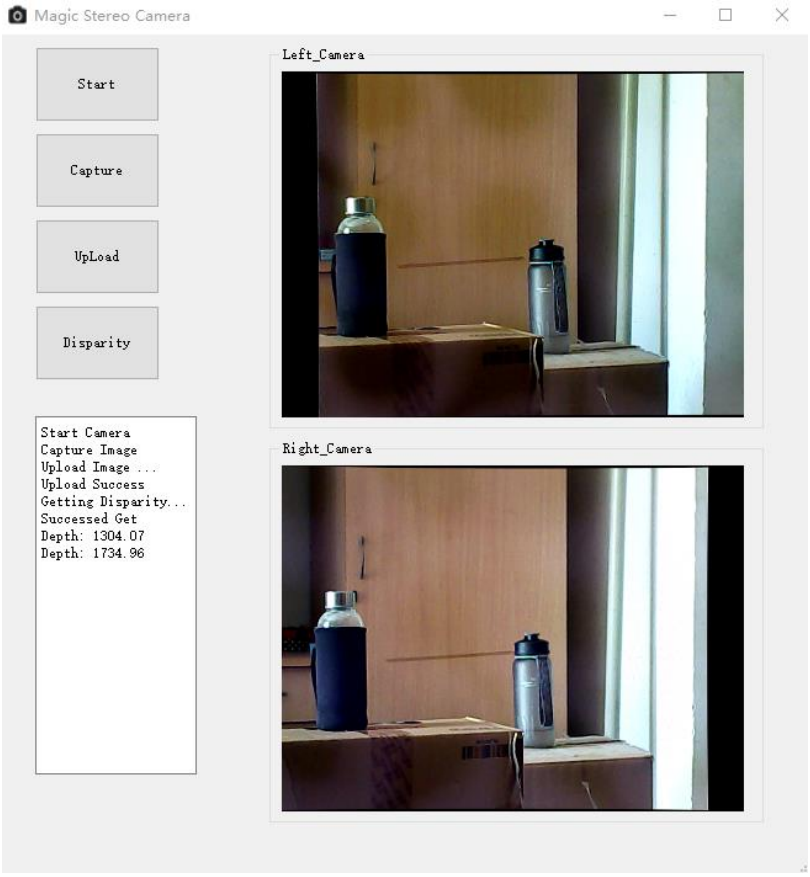

图 4-16　提示框显示深度信息

4.5 本章小结

本章节介绍了整个立体视觉的实现过程。首先本章介绍了实验中双目摄像机的离线标定、双目矫正和三维重建过程及相关参数，随后本章详细描述了立体匹配算法的实现细节，从环境搭建，模型训练到算法的实现细节。最后，本章简单介绍了服务端的搭建过程和客户端的 UI 设计。

第 5 章　总结与展望

5.1　论文总结

近年来，随着无人驾驶，机器人导航等技术的发展，计算机立体视觉作为其中重要的感知识别单元引起了研究者们广泛的关注。如何从二维的平面图像中提取出三维的深度信息成为一个重要的命题。双目立体匹配技术是双目立体视觉中的核心步骤，也是本文的研究重点。

本文在现有研究研究基础上，对立体匹配算法进行了深入研究，为提高匹配算法的精度，主要做了以下工作：

1) 本文对双目立体视觉的相关原理进行了深入研究，研究了摄像机成像原理，视差理论，立体匹配的经典步骤和约束条件。

2) 本文通过训练卷积神经网络进行图像块的相似性度量，在测试数据集中仅有8.02%的视差误差，已基本达到实际应用的需求。

3) 本文完成了基于 Qt 的双目视觉应用程序设计，用户可以连接双目摄像机，基本实现了双目测距。

5.2　论文研究的不足与展望

本文是在短时间内完成了双目立体视觉系统的设计和搭建，所以立体匹配算法中的许多超参数仍需要进一步优化，网络模型也有待于进一步的训练。实验中的许多模块仅仅出于调试的需求，并未考虑大规模的开发应用，系统容量十分有限。由于 GPU 内存的限制，本文双目视觉的计算时间超过了实际的应用的需求，后续研究中可以考虑应用多 GPU 或 TPU 来加速运算。本文双目立体视觉系统的双目矫正还不够完善，实际应用可以考虑采取严格的矫正步骤。

参 考 文 献

[1] 周星，高志军.立体视觉技术的应用与发展[J].工程图学学报,2010 年第 4 期，No.4.

[2] Marr D C.A Computational Investigation into the Human Representation and Processing of Visual Information [M]. San Francisco：W. H. Freeman and company，1982.

[3] Scharstein D, Szeliski R, Zabih R, et al. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms[J]. International Journal of Computer Vision, 2001, 47(1): 131-140.

[4] Kong D, Tao H. A method for learning matching errors for stereo computation.[C]. british machine vision conference, 2004.

[5] Zhang L, Seitz S M. Estimating Optimal Parameters for MRF Stereo from a Single Image Pair[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(2): 331-342.

[6] Scharstein D, Pal C. Learning conditional random fields for stereo[C]. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2007.

[7] Peris M, Martull S, Maki A, et al. Towards a simulation driven stereo vision system[C]//Pattern Recognition (ICPR), 2012 21st International Conference on. IEEE, 2012: 1038-1042.

[8] Li Y, Huttenlocher D P. Learning for stereo vision using the structured support vector machine[C]. computer vision and pattern recognition, 2008: 1-8.

[9] Haeusler R, Nair R, Kondermann D, et al. Ensemble Learning for Confidence Measures in Stereo Vision[C]. computer vision and pattern recognition, 2013: 305-312.

[10] Spyropoulos A, Komodakis N, Mordohai P, et al. Learning to Detect Ground Control Points for Improving the Accuracy of Stereo Matching[C]. computer vision and pattern recognition, 2014: 1621-1628.

[11] Zbontar J, Lecun Y. Computing the stereo matching cost with a convolutional neural network[C]. computer vision and pattern recognition, 2015: 1592-1599.

[12] Geiger A, Lenz P, Stiller C, et al. Vision meets robotics: The KITTI dataset[J]. The International Journal of Robotics Research, 2013, 32(11): 1231-1237.

[13] Guney F, Geiger A. Displets: Resolving stereo ambiguities using object knowledge[C]. computer vision and pattern recognition, 2015: 4165-4175.

[14] Zhang K, Lu J, Lafruit G, et al. Cross-Based Local Stereo Matching Using Orthogonal Integral Images[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2009, 19(7): 1073-1079.

[15] Hirschmuller H. Stereo Processing by Semiglobal Matching and Mutual Information[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 30(2): 328-341.

[16] Scharstein D, Pal C. Learning conditional random fields for stereo[C]//Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on. IEEE, 2007: 1-8.

[17] Scharstein D, Szeliski R. High-accuracy stereo depth maps using structured light[C]//Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. IEEE, 2003, 1: I-I.

[18] Scharstein D, Hirschmüller H, Kitajima Y, et al. High-resolution stereo datasets with subpixel-accurate ground truth[C]//German Conference on Pattern Recognition. Springer International Publishing, 2014: 31-42.

[19] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex[J]. The Journal of physiology, 1962, 160(1): 106-154.

[20] Fukushima K, Miyake S. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position[J]. Pattern recognition, 1982, 15(6): 455-469.

[21] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[22] Hinton G E, Osindero S, Teh Y, et al. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7): 1527-1554.

[23] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database[C]//Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009: 248-255.

[24] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[25] Zbontar J, LeCun Y. Stereo matching by training a convolutional neural network to compare image patches[J]. Journal of Machine Learning Research, 2016, 17(1-32): 2.

[26] Bromley J, Bentz J W, Bottou L, et al. Signature Verification Using A "Siamese" Time Delay Neural Network[J]. IJPRAI, 1993, 7(4): 669-688.

[27] Hirschmuller H. Stereo processing by semiglobal matching and mutual information[J]. IEEE Transactions on pattern analysis and machine intelligence, 2008, 30(2): 328-341.

[28] Bradski G, Kaehler A, 于仕琪, 等. 学习 OpenCV[J]. 清华大学出版社, 北京, 2009.

[29] Abadi M, Agarwal A, Barham P, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015[J]. Software available from tensorflow. org.

[30] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]//Proceedings of the ACM International Conference on Multimedia. ACM, 2014: 675-678.

[31] Chen T, Li M, Li Y, et al. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems[J]. arXiv preprint arXiv:1512.01274, 2015.

致　谢

经过几个月来的忙碌工作，我的毕业论文设计也要接近尾声了。作为一个本科学生的毕业设计，由于经验的缺乏，难免会有许多考虑不全之处，在这里，我要由衷的感谢指导老师宣琦研究员，正是在他的悉心指导和帮助下，我进入了人工智能这个全新的领域，学会了如何利用这项技术解决现实问题，他渊博的专业知识，严谨的治学态度，精益求精的工作作风和诲人不倦的高尚师德，给我留下了深刻的印象，使我受益匪浅。在此，谨向导师致以崇高的敬意和衷心的感谢。

离别在即，站在人生的又一转折点上，心中难免思绪万千，一种感恩之情油然而生。感谢所有鼓励支持过我的学院领导、任课老师和学长学姐，是他们给我带来了丰富多彩的大学生活，在他们的帮助下，我参加了各项文体科技竞赛，得到了一次次锻炼自己的机会，包括最后的毕业设计，也是在大家的共同努力下完成的。作为信息人，我不会忘记演讲台上的激昂青春，不会忘记实验室里辛勤汗水，这一次次的经历将会在日后的工作和生活中产生巨大的影响。风雨兼程，感谢一路有你们。别了我的大学，让我们江湖再见。