工学硕士学位论文

基于双目视觉的立体匹配算法研究与 FPGA 实现

吴振

哈尔滨理工大学

2016 年 3 月

国内图书分类号：TP391.41

工学硕士学位论文

基于双目视觉的立体匹配算法研究与 FPGA 实现

硕士研究生：吴振

导　　师：韩剑辉

申请学位级别：工学硕士

学科、专业：计算机科学与技术

所在单位：计算机科学与技术学院

答辩日期：2016 年 3 月

授予学位单位：哈尔滨理工大学

Classified Index：TP391.41

Dissertation for the Master Degree in Engineering

Stereo Matching Algorithm Based on Binocular Vision and its Implementation with FPGA

Candidate： Wu Zhen

Supervisor： Han Jianhui

Academic Degree Applied for： Master of Engineering

Specialty： Computer Science and Technology

Date of Oral Examination： March，2016

University： Harbin University of Science and Technology

哈尔滨理工大学硕士学位论文原创性声明

本人郑重声明：此处所提交的硕士学位论文《基于双目视觉的立体匹配算法研究与 FPGA 实现》，是本人在导师指导下，在哈尔滨理工大学攻读硕士学位期间独立进行研究工作所取得的成果。据本人所知，论文中除已注明部分外不包含他人已发表或撰写过的研究成果。对本文研究工作做出贡献的个人和集体，均已在文中以明确方式注明。本声明的法律结果将完全由本人承担。

作者签名：吴振　　日期： 2016 年 3 月 15 日

哈尔滨理工大学硕士学位论文使用授权书

《基于双目视觉的立体匹配算法研究与 FPGA 实现》系本人在哈尔滨理工大学攻读硕士学位期间在导师指导下完成的硕士学位论文。本论文的研究成果归哈尔滨理工大学所有，本论文的研究内容不得以其他单位的名义发表。本人完全了解哈尔滨理工大学关于保存、使用学位论文的规定，同意学校保留并向有关部门提交论文和电子版本，允许论文被查阅和借阅。本人授权哈尔滨理工大学可以采用影印、缩印或其他复制手段保存论文，可以公布论文的全部或部分内容。

本学位论文属于

保密，□ 在　　年解密后适用授权书。

不保密 √ 。

(请在以上相应方框内打 √)

作者签名：吴振　　日期： 2016 年 3 月 15 日

导师签名：韩剑辉　　日期： 2016 年 3 月 15 日

基于双目视觉的立体匹配算法研究与 FPGA 实现

摘 要

双目立体视觉是计算机视觉技术研究领域中的热点问题之一，主要采用两台摄像机对同一景物进行拍摄，根据几何原理将不同位置获取的二维图像信息进行三维重建，从而恢复原景物的三维信息。由于其原理是直接模拟人眼观察事物的方式，是获取三维场景中深度信息的重要手段。立体匹配是双目立体视觉中的一个重要研究方向，在机器人导航、三维测距、虚拟现实等领域具有广泛的应用。

改进匹配算法来提高匹配的精度，首先，在进行 Mini-Census 变换时，邻域像素需要与中心像素及其平均值进行比较，得到一串两位数码；然后，利用图像的梯度信息动态地调节匹配窗口的尺寸，根据匹配窗口尺寸的差异分别使用改进稀疏 Census 变换；最后，将左右一致性检测、投票算法相结合，对视差图进行细化处理，从而得到较为准确和稠密的视差图。

双目立体匹配算法的计算量大、复杂度高，一些传统平台无法满足实时性要求，从而在很大程度上制约了在实际中的应用。本文采用现场可编程门阵列（Field Programmable Gate Arrays FPGA）作为硬件实现平台，采用 Verilog HDL 语言设计匹配算法中图像数据缓存、Census 变换、梯度计算、左右一致性检测等相关功能模块，通过功能模块间的并行化和流水线化，提高匹配算法的处理速度。

本文选用 Altera 公司的 Cyclone II 系列芯片中的的 EP2C35F672 的 FPGA 芯片作为处理核心，Quartus II 11.1 作为主要开发工具，使用线逻辑分析仪 SignalTap II 及 Modelsim SE6.5 等仿真工具进行在线调试。选择 Middlebury 图像库中的标准测试立体图像对进行试验。验证本文算法在匹配精度和实时性两方面的可行性和有效性。

关键词 双目视觉；立体匹配；稀疏 Census 变换；实时；可编程门阵列

Stereo Matching Algorithm Based on Binocular Vision and its Implementation with FPGA

Abstract

Binocular stereo vision is one of the hot issues in the field of computer vision technology research. It mainly uses two cameras to shoot the same scene.According to the geometrical principle of different locations of two-dimensional image information of 3D reconstruction, the three-dimensional information of the original scene can be restored. It is an important means to obtain the depth information of the 3D scene due to the way to directly simulate the human eye. Stereo matching is an important research direction in binocular stereo vision and mainly used in the field of robot navigation, hree-dimensional measurement and virtual reality.

The improved matching algorithm improves the matching accuracy. Firstly ， in the Mini-Census transform, the neighborhood pixels need to be compared with the average values of the pixels in the center and the neighboring pixels, and a series of two digits are obtained.Then the size of the matching window is dynamically adjusted by using the gradient information of the image. Finally, the left and right consistency checking, voting algorithm and histogram are combined to refine the disparity map to get more accurate and dense disparity map.

The binocular vision stereo matching algorithms with computational complexity and large amount of data are unable to meet the real-time requirement,so they are restricted in the practical applications.This paper chooses Field Programmable Gate Arrays(FPGA) as the platform of hardware and the Verilog HDL is used to design the functional modules of the matching algorithm such as the image data cache, Census transform, gradient calculation, the left and right consistency checking.

This paper chooses Altera’s EP2C35F672 of Cyclone II series as the processing core, Quartus II 11.1 as the development tool and SignalTap II and

Modelsim SE6.5 as the debugging tools.The stereo images from the Middlebury datasets are chosen as the experimental subjects and this algorithm is verified the feasibility and effectiveness in the matching accuracy and processing speed.

Keywords binocular vision,stereo matching,sparse Census Transform,real time,field programmable gate array

目 录

第 1 章 绪论

1.1 课题研究的目的及意义

让计算机或机器人具有像人或其他生物那样高效、灵敏的视觉，是人们多年的梦想，计算机立体视觉技术应运而生。计算机立体视觉一般可分为双目立体视觉和多目立体视觉，其中，双目立体视觉是计算机立体视觉研究技术的最小系统。双目立体视觉技术采用直接模拟人类双眼处理景物的方式，简单可靠，而且是多目立体视觉系统的基础。双目立体视觉技术已经广泛的应用于虚拟现实、三维测量、立体摄像机、目标识别和机器人导航等诸多领域。

人的双眼获取空间内同一物体的两幅图像，视觉神经对两幅图像中对应物体具有的位置差异，即视差，进行处理，从而感知物体在场景中的深度信息。双目立体视觉系统接收双摄像机采集的左右图像，经过立体匹配，获取视差图，从而确定深度图像[1]。立体匹配是整个双目立体视觉系统中最耗时、最困难的一个步骤，匹配的精度和速度对于双目立体视觉的形成具有很大影响，因此在计算机立体视觉技术研究中，双目视觉的立体匹配是热点也是难点[2]。

要完成基于双目视觉的立体匹配，涉及到的相关算法复杂度都很高，需要处理大量数据。而且，随着数字图像技术的发展，高帧数、高分辨率的视频流对立体匹配的处理速度带来了更大的挑战，采用通用的计算机串行处理方式很难满足实时性的要求，因而需要开发专用的硬件并行处理系统[3,4]。在以往设计中常采用专用集成电路 ASIC（Application Specific Integrated Circuit）结合计算机或数字信号处理器（DSP）来实现双目立体视觉处理算法，但是 ASIC 相对固定且开发周期较长，无法随着最新的技术发展及时的对系统进行调整和更新。另外过多的分立元件必然增加电路板的尺寸和功率消耗，降低系统的吞吐率，增加系统不稳定的风险。

可编程门阵列（Field Programmable Gate Array，FPGA）是近年来主流的大规模可编程专用集成电路，为图像处理技术提供了新的思路和方法。FPGA 具有较强的并行处理能力，打破了顺序处理的模式，超越了多种微处理器的运算能力。而且 FPGA 可靠性高、应用灵活、可现场升级、成本低，相比于 ASIC、DSP 以及其他由分立元件组成的系统具有开发周期短、成本低、维护升级简便等优点。利用 FPGA 实现双目立体匹配算法既实现了高速处理，又保证系统的

实时性和准确性。

1.2 双目立体匹配算法的研究现状与展望

1.2.1 国内外研究现状

从 20 世纪 60 年代开始，众多学者对立体视觉技术的发展做出了自己的贡献。上世纪 70 年代末，Marr 结合图像处理、心理物理学、神经生理学及临床精神病学等诸多领域的研究成果，第一个提出了一个较为完备的视觉系统框架，为立体视觉技术后续的发展提供了稳固的理论基础。经历数十年发展，立体视觉已经成为一门新兴学科，并且被广泛应用在机器人、医学诊断、航天测控、军事应用等领域中[5,6]。

双目立体视觉的基础是双目视差，因而立体匹配成为立体视觉中极为重要的一个步骤。20 世纪 80 年代，Barnard 和 Fischler 回顾了之前立体视觉的研究成果，总结了一套在立体视觉中图像获取、相机模型、特征提取、立体匹配、深度信息获取等多方面的理论，并且对当时流行的匹配方法进行了性能评估，为以后立体视觉技术的发展奠定了坚实的基础[7,8]。20 世纪 80 年代末，Dhond 和 Aggarwal 提出了了许多立体匹配的新算法[9,10]。20 世纪 90 年代以后，通用的立体匹配的算法依然是研究的热门。文献[11]总结了 1989 年以来 86 篇论文的研究成果，文中收录的论文采用了各种各样的立体匹配算法，如基于邻域、特征的匹配方法，基于彩色信息的匹配方法，多目图像匹配方法等，并且对这些算法进行了性能评估和比较。目前很多立体匹配的研究都是建立在这一研究成果之上的。值得一提的是，该论文还总结了匹配算法的实时性处理问题，即采用特殊硬件实现立体匹配算法。

20 世纪 90 年代，Bhat 和 Nayar 改进了传统的局部相关度匹配算法，提出了基于窗口灰度值等级序列的匹配方法，提高了算法的鲁棒性[12,13]；Zabin 和 Woodfill 提出了两种非参数化的局部相关度转换方法：rank 变换和 census 变换，这两种方法实现简单，鲁棒性强，在之后的发展中逐渐被改进，成为目前使用的最多的两种局部匹配算法[14,15]。

2001 年，Scharstein 和 Szeliski 通过研究各类立体匹配算法，将立体匹配算法进行了分类，提出了现有立体匹配算法一般由 4 个独立模块组成：匹配代价计算、代价聚合、视差计算和视差优化[16]。著名的美国的 Middlebury 学院，院创建了立体匹配算法研究的交流网站 http://vision.middlebury.edu/，其所提供

的 Middlebury 双目立体视觉测试数据集也成为了当前主流的算法评估对象。而且目前的对于立体匹配算法的研究大多采用该站点提供的图像样本数据，并将立体匹配实验的结果上传到该网站，来获取权威的性能分析[17]。

近年来国内的学者也高度关注立体匹配算法的研究，以中科院、清华大学、浙江大学为代表的众多高校也正在研究高效快速的立体匹配算法，并取得了一系列的研究成果[18]。中科院自动化所的机器人视觉研究组基于特征点的匹配算法，开发了 CVSuite 系统软件[19]。三目立体匹配算法由北京理工大学实现，并将其固化在片上系统中，已经成功地应用在了机器人避障检测的领域。香港大学的 Maxime Lhuillier 博士和龙泉教授[20]，对于基于区域分割的区域增长算法的研究取得突破性进展，采用基于区域生长的种子点的选择的算法得到了稠密的视差图与数目较多的匹配点。

1.2.2 立体匹配算法硬件实现的研究现状

目前，针对立体匹配算法的研究主要集中在两个方面：第一是找到精度更高的匹配算法，减少误匹配；第二是在保证一定的匹配精度前提下，提高算法的运算速度，从而满足实时性要求。文献[21]提出的一种基于分割的加速算法，虽然速度有所提高，但处理时间还在秒级，无法满足实时性的要求；文献[22]提出了基于自适应窗口局部匹配的加速匹配算法，但是速度也没有大幅提升。通用计算机上运行的软件方法无法满足实时性的要求，需要专用的硬件设备对算法进行加速，才能满足实际需要。现在，常用的硬件加速方案有 4 种：第一，全定制的专用集成电路（ASIC）；第二，数字信号处理器（DSP）；第三，图形处理器（GPU）；第四，现场可编程门阵列（FPGA）。

ASIC 是一种为特定目标设计的集成电路，体积小、功耗低、速度快。文献[23]实现了一种基于 ASIC 的立体匹配系统，采用 SSD 算法，针对 256*192 分辨率的图像可达到 50fps 的处理速度；文献[24]采用了基于自适应窗口权重的匹配算法，利用 ASIC 实现，相对于软件处理其速度具有大幅提高。ASIC 由于属于纯硬件结构，适合于立体匹配算法的加速研究中，但是 ASIC 具有开发周期长、成本高、灵活性低等缺点，无法在实际应用中大规模使用。

DSP 是专门的数字信号处理微处理器，广泛地应用在各种各样数字信号处理算法的硬件实现中[25,26]。然而从本质上说，DSP 体系依然是串行指令系统，只能对一些特殊的计算实现加速处理，但是这些加速处理还是无法满足用户对于速度的要求，这导致了 DSP 在立体匹配领域没有得到广泛应用。文献[27,28]

使用 DSP 实现了局部立体匹配算法，但是处理速度仍然没有达到理想水平，无法应用于实际中。

近几年 GPU 发展迅速，性能有了很大的提升。GPU 采用流水线体系结构，可以看作是单令流多数据流（SIMD）的并行处理机制[29]。流水线结构和并行处理使 GPU 符合局部立体匹配算法的速度要求，而且高级语言的使用可以使开发人员将精力更多地集中在匹配算法的研究上，因而成为目前立体匹配算法硬件加速领域的一个重要研究方向。文献[30,31]使用 GPU 作为立体匹配的硬件加速平台，根据实验结果，其处理速度已经超过 DSP，其可编程的特性也优于 ASIC，但是 GPU 能耗高，可扩展的外围接口少，在某些应用环境或者大规模使用上难以推广。

FPGA 近些年来发展迅猛，随着性能的不断提升，在高速图像处理、数字通信等领域逐渐占据主导地位。而且，其并行特性和可流水线的结构，FPGA 已经成为立体匹配硬件加速最为理想的平台。文献[32]实现了一套基于 FPGA 的立体匹配系统，该系统采用 SAD 局部匹配算法，在 286MHz 的时钟频率下能够以 25.6fps 的速度 512∗512 的图像进行处理，基本可以满足实时性要求。文献[33]采用自适应窗口的 SAD 立体匹配算法，在 FPGA 上实现，采用窗口并行和像素并行的双并行策略，可以将 64∗64 灰度图像的处理速度提高到 0.19ms。文献[34]利用 FPGA 处理 640∗640 的图像，其速度已经达到 60fps。

综上所述，FPGA 适合作为硬件平台实现立体匹配算法的加速，而且加速效果明显，是将立体匹配向实际应用发展的有效途径。随着 FPGA 性能的不断提高和立体匹配算法的不断深入研究，处理速度将会越来越快。在立体匹配算法的硬件加速研究中，FPGA 由于并行性能好、通用性好、流水线设计等优点，成为立体匹配算法加速处理的热门平台。

1.3 课题的来源及研究内容

1.3.1 课题来源

题目自拟。

1.3.2 课题的主要研究内容

课题主要研究基于双目视觉的立体匹配算法，并利用 FPGA 平台硬件实现

立体匹配算法。本课题的研究内容主要由以下几个方面组成。

1. 介绍双目立体视觉原理及相关理论，分析各种立体匹配算法的优缺点。

2. 选择 FPGA 作为核心控制器，硬件实现立体匹配算法。根据所设计的双目视觉立体匹配算法规划子模块，使用 Verilog HDL 语言实现各个子模块，充分利用 FPGA 的并行特性，最大程度地对匹配算法进行加速。

3. 对基于 FPGA 的双目视觉立体匹配算法进行功能仿真，通过 Middlebury 网站提供的标准测试图像进行试验，对试验结果进行分析，重点主要集中在立体匹配的精度和实时性两个方面。

本文章节安排如下：

第一章介绍课题的背景、目的和意义，介绍了立体匹配算法的发展现状和算法硬件实现的研究概况，确定选用 FPGA 作为硬件实现算法的核心控制芯片。

第二章主要介绍三种图像坐标及其之间的转换关系、双目立体视差理论、空间三维坐标计算等相关理论，并分析当前流行的各种立体匹配算法的优缺点。

第三章介绍基于 Census 变换的区域立体匹配算法，针对算法的不足提出改进方案。提出一种自适应匹配窗口的生成算法，并研究视差图细化提出一种能够填补遮挡区域的算法。

第四章具体介绍基于 FPGA 的立体匹配算法的硬件实现。首先介绍整体框架和子模块划分，然后详细介绍立体匹配中各个子模块的实现。

第五章进行实验和结果分析，对各个模块进行功能仿真，重点分析对比了本文算法的匹配精度和实时性。

第 2 章 相关理论概述

2.1 双目立体视觉基本原理

2.1.1 双目立体视觉坐标系

摄像机可以通过镜头将三维空间中的景物投影到二维的成像平面上，因而三维空间像素点的坐标与二维图像平面中像素点的坐标之间存在着转换关系，成像系统的坐标系包括以下几个[35]：

（1）世界坐标系$(O_w - x_w y_w z_w)$：实际场景中摄像机与物体的位置坐标信息，是所有的坐标的参考坐标系。

（2）摄像机坐标系$(O_c - x_c y_c z_c)$：原点处于光心，摄像机的光轴为 z 轴，其 x 轴和 y 轴平行于像平面坐标系 x 轴和 y 轴。

（3）像平面坐标系$(O_i - xy)$：原点是光轴与像平面的交点，x 轴与 x_c 平行，y 轴与 y_c 轴平行，方向相同。

（4）像素坐标系$(O-uv)$：数字图像通常是用 $M*N$ 的二维数组来描述，M 表示图像的行数，N 表示图像的列数。(u,v) 表示图像中的某个像素点，其值为该像素的亮度或灰度值，u 和 v 分别表示该像素点所处的行号和列号，u 和 v 均为正值。图像坐标原点是图像左上角的像素点，规定坐标系的行坐标向右增加，列坐标向下增加。

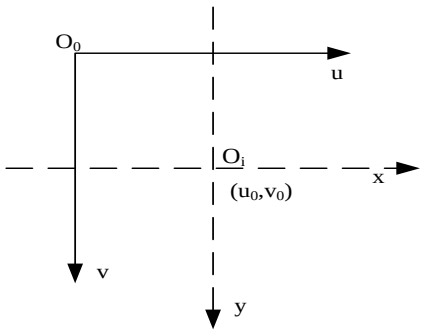


图 2-1 像平面坐标和像素坐标系的关系

Fig.2-1 Relationship between image coordinates and pixel coordinate system

图 2-1 表示的是像平面坐标系与像素坐标之间的关系，像平面坐标表示像

素点的坐标采用的是物理长度单位（如毫米、微米等），因而是实际的物理位置信息。像素坐标系是以像素为单位的，像素点的坐标是相对于原点的相隔的像素个数，不携带任何物理信息[36]。

图 2-1 中，(u,v)表示某点在像素坐标系的内坐标，(x,y)是该点在像平面坐标系中内的坐标。原点O_i处于镜头光轴与图像平面的交点，(u_0,v_0)是像素坐标系的原点坐标，dx和dy表示像素在x轴和y轴上的物理增量。两个坐标系的转换关系如公式(2-1)所示。

$$\begin{cases} u = \dfrac{x}{dx} + u_0 \\ v = \dfrac{y}{dy} + v_0 \end{cases} \tag{2-1}$$

通常采用齐次坐标矩阵表示：

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \dfrac{1}{dx} & 0 & u_0 \\ 0 & \dfrac{1}{dy} & v_0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2-2}$$

图 2-2 表示的是世界坐标系(x_w,y_w,z_w)、摄像机坐标系(x_c,y_c,z_c)和像平面坐标系(x_i,y_i,z_i)三者之间的关系。

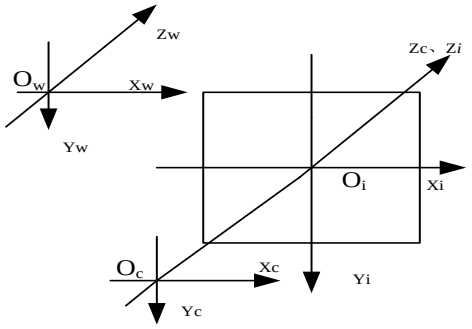


图 2-2 世界坐标系、摄像机坐标系和像平面坐标系的关系

Fig.2-2 Relationship between world coordinate system, camera coordinate system and image plane coordinate system

从摄像机坐标(x_c,y_c,z_c)到像平面坐标(x,y)的转换关系如公式(2-3)所示：

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

用齐次矩阵表示如下：

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

其中 s 为比例因子，f 为相机的焦距。

像素点的摄像机坐标转换为世界坐标，需要利用矩阵的平移和旋转操作。某个像素点 P 的世界坐标为 (x_w, y_w, z_w)、摄像机坐标为 (x_c, y_c, z_c)，那么转换公式如式(2-5)所示：

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = R\begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} + T \tag{2-5}$$

其中 $R = \begin{pmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{pmatrix}$ 称为旋转矩阵，为3×3正交矩阵；$T = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$ 为平移矩阵。将公式(2-2)、(2-4)、(2-5)结合起来可以得到表达式如式(2-6)所示：

$$s\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \dfrac{f}{dx} & 0 & u_0 & 0 \\ 0 & \dfrac{f}{dy} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \tag{2-6}$$

其中，$\dfrac{f}{dx}$ 与 $\dfrac{f}{dy}$ 分别为 u 轴和 v 轴上的比例因子。

2.1.2 视差理论

人眼在观察物体时，由于对于同一个物体，人眼与其所形成的角度是不同的，从而该物体在人眼视网膜上的投影位置不同，这个位置差就称为视差（disparity），人眼正是因为视差才能感受到客观世界的深度信息。视差反应了深度信息，计算机双目立体视觉采用左右两个摄像头模拟人的双眼，根据立体匹配求得视差图，根据视差还三维场景的深度图[37]。

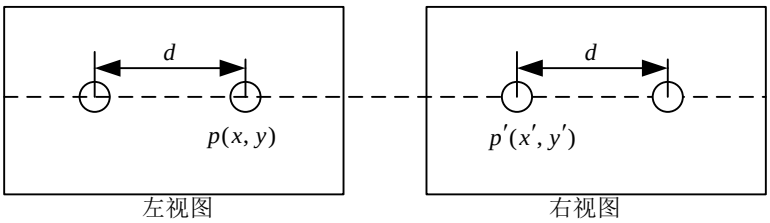


图 2-3 立体图像对中 $P(x,y)$ 的视差

Fig.2-3 Disparity of $P(x,y)$ in stereo image pair

如图 2-3 所示，左图和右图分别为左相机和右相机生成的图像，左图上有一点 $P(x,y)$，该点在右图上的对应点为 $P'(x',y')$。两点在同一水平线上，则以右图为标准，设该点的视差为 d，则有：

$$\begin{cases} y = y' \\ d = x - x' \end{cases} \tag{2-7}$$

由此可见，立体匹配的实质是求取立体图像对中所有的匹配点，并且计算出对应匹配点的位置差，即视差，然后将视差以灰度的形式保存起来，从而得到视差图。立体匹配的目的就是求得视差图。

求得视差图以后，需要根据视差信息求取实际的深度信息，这里就涉及到视差与深度的关系。如图 2-4 所示，C_L' 和 C_R' 为左右两台相机的光心，它们之间的距离为 B。C_L 和 C_R 是成像的中心，光心与中心的距离为焦距 f。P 是场景中的任意一点，该点在左相机中的成像点为 P_L，在右相机中的成像点为 P_R，令 $|C_LC_L| = l_L$，$|C_RC_R| = l_R$，则以 C_L 和 C_R 为原点，P_L 的坐标为 (I_R, y)，则根据上面求视差的方法，该点的视差为：

$$d = l_L - l_R \tag{2-8}$$

令 $|P_RA| = a$，设点 P 到光心的距离，即深度 Z，由 $\Delta OPC_R'$ 与 ΔAPP_R 相似，可以得到式(2-9)：

$$\frac{Z-f}{Z} = \frac{a}{a+l_R} \tag{2-9}$$

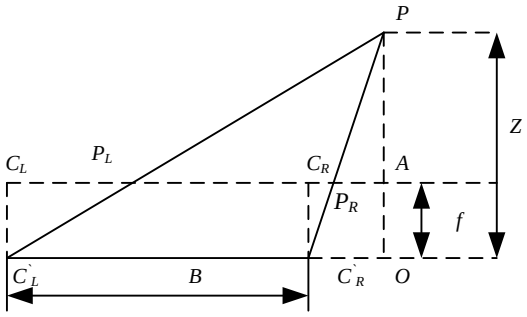


图 2-4 双目视觉成像模型

Fig.2-4 Imaging model of binocular vision

由 $\Delta OPC'_L$ 与 ΔAPP_L 相似，可以得到：

$$\frac{Z-f}{Z}=\frac{B-l_L+l_R+a}{B+a+l_R} \tag{2-10}$$

由上面两个公式，可以得到：

$$\frac{a}{a+l_R}=\frac{B-l_L+l_R+a}{B+a+l_R}=1-\frac{l_L}{B+a+l_R} \tag{2-11}$$

从而可得：

$$a=\frac{B*l_R}{l_L-l_R}-l_R \tag{2-12}$$

将公式(2-12)带入公式(2-9)，可得：

$$Z=\frac{B*f}{l_L-l_R} \tag{2-13}$$

将公式(2-8)带入公式(2-13)，可得：

$$Z=\frac{B*f}{d} \tag{2-14}$$

由公式(2-14)可以看出，深度与视差成反比关系，与两个相机的间距 B 和焦距 f 成正比关系，在实际的应用中，相机的相对位置固定，相机的焦距也是固定的，那么得到视差就可以求出实际的深度信息。根据视差与深度成反比这一结论，可以知道距离相机距离近的物体视差大，而距离相机远的物体视差小。

2.1.3 空间三维坐标

实际场景中物体的三维坐标可以通过双目立体视觉技术来确定。图 2-5 是

平行的双目立体视觉模型原理图，假设两摄像机的内部和外部参数完全相同，且焦距为 f，两摄像机镜头光心（O_L 和 O_R）之间的距离为基线距 B，两台摄像机在同一平面上，它们的投影中心的图像坐标 Y 值相同即 $Y_{left}=Y_{right}$，同一时刻拍摄到空间物体中的同一特征点 $P(x,y,z)$，分别在左摄像机和右摄像机上获取了点 P 的图像点 $P_{left}(x_{left},y_{right})$ 和 $P_{right}(x_{right},y_{left})$。

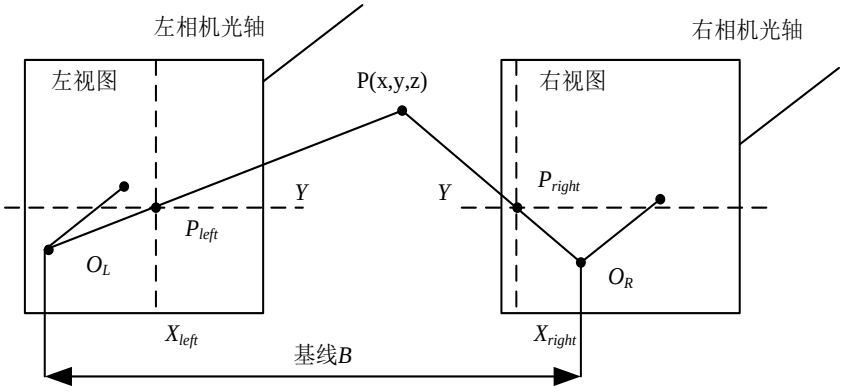


图 2-5 平视双目立体视觉模型

Fig.2-5 Head model binocular stereo vision

由三角几何关系可得摄像机上 P 点的坐标值，如式(2-15)所示：

$$\begin{cases} X_{left}=f\dfrac{x}{z} \\ X_{right}=f\dfrac{(x-B)}{z} \\ Y=f\dfrac{y}{z} \end{cases} \tag{2-15}$$

视差 D 为 P_{left} 和 P_{right} 两点之间的距离，即 $D=X_{left}-X_{right}$，则点 P 在平视模型系统下的三维坐标如式(2-16)所示：

$$\begin{cases} x=\dfrac{BX_{left}}{D} \\ y=\dfrac{BY}{D} \\ z=\dfrac{Bf}{D} \end{cases} \tag{2-16}$$

2.2 立体匹配约束条件

在摄像机拍摄图像的过程中，会丢失一些信息。立体匹配的最终目的是重建三维场景，如果不附加一些约束条件，很可能得不出可行解。立体图像对中，一幅图称为参考图，作为待匹配图；另一幅图称为目标图，而所生成的视差图以参考图为标准。对于参考图中的某个点，在与其共轭的目标图中可能有多个候选匹配点，因此需要加入一些约束条件，才能得到唯一的匹配点。立体匹配的基本约束条件有如下几条[38]：

1. 外极线约束(epipolar constraint)：外极线约束主要是为较小视差的搜索的难度，依据透视投影成像的几何原理，外极线是一条特定的直线，匹配图像对中对应的像素点都位于同一条外极线上。这样就将搜索范围从二维降到了一维，大大提高了搜索效率，降低了误匹配率。

2. 唯一性约束(uniqueness constraint)：目标图像中的每一个点在参考图像中至多只能有一个匹配点，而遮挡区域的点可能没有匹配点，即除遮挡区域外，两幅图像中所有的点的对应关系是唯一的。

3. 相似性约束(similarity constraint)：空间内某一物体在左右图像中的投影应具有相似性，即两个匹配基元（像素点、像素集或相位信息)具有相同或相近的属性。

4. 连续性约束(smoothness/coherence constraint)：主要的依据是现实世界中物体的表面一般都很平，除遮挡区域或物体自身视差存在不连续，视差变化范围内很小，可以消除目标物体由于表面不平整而引起的变化。

5. 顺序约束(ordering constraint)：在立体图像对中，目标图中的一系列待匹配点在参考图中对应极线上的位置是顺序的。顺序约束符合客观事实，在空间位置上为立体匹配添加了约束，减少了立体匹配的计算量和歧义性。

6. 遮挡约束(occlusion constraint)：因为双目立体匹配系统采用不同视角拍摄场景，所以存在遮挡现象。遮挡约束的的定义为，当图像对中一幅图像存在视差不连续，则该区域在另一幅图像中为遮挡区域，即背景区域被前景景物遮挡。

7. 视差范围约束(disparity limit constraint)：在立体匹配的过程中，为视差的搜索范围设定一个阈值是有科学依据的，并且限定视差范围可以在尽量保证精度的情况下，大大提高处理速度。

8. 左右一致性约束(mutual correspondence constraint)：在一般情况下，以左

图为参考图和以右图为参考图使用相同的算法做双向匹配时，所得到的匹配对应该是相同的。若以左图为参考图，P 点在右图的匹配点为 Q 点，则以右图为参考图，Q 点在左图的匹配点应该为 P 点。不满足这一约束的点一般是遮挡区域，该约束可以用来检测遮挡区域。

2.3 立体匹配算法研究

立体匹配算法的分类没有统一的标准，目前应用最广的，也是比较公认的分类方法为，根据是否采用全局优化的算法将立体匹配算法划分成局部立体匹配算法和全局立体匹配算法[39]。

2.3.1 局部立体匹配匹配算法

局部立体匹配算法没有采用全局优化算法，仅仅根据待匹配像素周围区域的图像信息进行计算。局部立体匹配算法具有实现简单、速度快、效率高等优点，非常适合实时性要求高的系统。但是，对于弱纹理区域、遮挡区域的会产生很高的误匹配率，并且对噪声敏感。局部立体匹配算法依据选取的匹配基元的差异可以划分成区域立体匹配算法、特征立体匹配算法和相位立体匹配算法[40]。

1. 区域立体匹配算法

区域匹配算法是以待匹配点周围的窗口区域作为匹配基元，利用相似度测量函数计算匹配点之间的相似度，选取相似度最大的点作为匹配结果，从而计算出视差。因此，区域匹配算法包括两个重要问题：匹配窗口的选取和相似度测量函数。

匹配窗口的选择是区域匹配算法的一个重要问题，如果采用固定窗口，虽然实现简单、速度快，但对于深度不连续区域或者弱纹理区域比较容易产生误匹配；如果采用可变窗口，窗口大小需要根据实际的位置进行变化，增加了复杂度，但能获得更好的匹配效果[41]。

在确定匹配窗口以后，区域立体匹配算法需要对窗口内的像素进行匹配代价计算和代价聚合。良好的相似度函数可以提高处理速度和匹配精度。定义 W 为匹配窗口，$C(u,v,d)$ 为窗口内的匹配代价，$I_L(u,v)$ 和 $I_R(u,v)$ 分别是像素点 (u,v) 在左右图像中的像素值。当前广泛使用的相似度测量函数有：

(1) 归一化互相关（Normalied Cross Correlation，NCC）

$$C(u,v,d)=\frac{\sum_{(u,v)\in W}((I_L(u,v)-\bar{I_L})\bullet(I_R(u+d,v)-\bar{I_R}))}{\sqrt{\sum_{(u,v)\in W}((I_L(u,v)-\bar{I_L})^2\bullet(I_R(u+d,v)-\bar{I_R})^2)}} \tag{2-17}$$

(2) 差的平方和（Sum of Squared Difference，SSD）

$$C(u,v,d)=\sum_{(u,v)\in W}(I_L(u,v)-I_R(u+d,v))^2 \tag{2-18}$$

(3) 归一化差的平方（Normalized SSD）

$$C(u,v,d)=(\frac{I_L(u,v)-\overline{I_L}}{\sqrt{\sum_{(u,v)\in W}(I_L(u,v)-\overline{I_L})^2}}-\frac{I_R(u+d,v)-\overline{I_R}}{\sqrt{\sum_{(u,v)\in W}(I_R(u,v)-\overline{I_R})^2}} \tag{2-19}$$

(4) 差的绝对值和（Sum of Absolute Differences，SAD）

$$C(u,v,d)=\sum_{(u,v)\in W}\left|I_L(u,v)-I_R(u+d,v)\right| \tag{2-20}$$

(5) Rank 变换

$$R_k(x_c,y_c)=\left\|\{(x,y)\,|\,I_k(x_c,y_c),(x,y)\in W_c,(x,y)\neq(x_c,y_c)\}\right\| \tag{2-21}$$

$$C(u,v,d)=\sum_{(u,v)\in W}\left|R_L(u,v)-R_R(u+d,v)\right| \tag{2-22}$$

(6) Census 变换

$$S_k(x_c,y_c)=Bitstring_{(x,y)\in W}(I_k(x,y)<I_k(x_c,y_c)) \tag{2-23}$$

$$C(u,v,d)=Ham\min g(S_L(u,v),S_R(u+d,v)) \tag{2-24}$$

公式(2-17)和(2-19)属于归一化度量算法，公式(2-18)和(2-20)属于灰度差度量算法，公式(2-21)、(2-22)、(2-23)和(2-24)是非参数变换算法[42]。灰度差度量算法的运算量比较小，实现也相对简单，但是对于光照不均和遮挡区域的匹配结果误差较大。归一化度量考虑到整个窗口的亮度信息，受光照影响较小，鲁棒性好于灰度差度量方法，但是这类算法复杂度较高，包括开方这类复杂运算，不适合使用硬件实现。非参数变换算法根据窗口元素与中心元素的关系代替像素本身的灰度值，有效解决了光照不均的影响。并且，非参数变换算法计算简单，易于硬件实现[43]。

2. 特征立体匹配算法

特征匹配选取的是图像中的强特征点作为待匹配点，如轮廓、角点、拐点

等，因此匹配精度比较高，并且强特征点数量一般较少，从而计算速度快，但是由于图像中的特征点具有间断性、稀疏性的特点，所以特征匹配所得到的一般是稀疏得视差图，还需进行插值运算才能得到完整的视差图，也加大了计算量。虽然这种算法对于所选取特征点的匹配精度很高，但经过插值所得到的其它点的匹配精度会受到已匹配点个数和插值算法精度的影响较大，而且性能也很不稳定。如果图中强特征点个数较多，所得到的视差图效果较好，否则，结果一般不会很理想[44]。

3. 相位立体匹配算

假设图像对中的对应像素的局部相位相等，相位匹配算法就是将图像进行带通滤波，之后将滤波所得信号的相位信息进行处理，最终得到图像对匹配的视差图。相位匹配算法对高频噪声和畸变具有很好的抑制作用，但这种算法的计算量偏大，不适于硬件实现法[45]。

2.3.2 全局立体匹配算法

和局部立体匹配算法相比，全局匹配算法的不同之处在于是从全局出发[46]，通过全局的能量函数对图像信息进行表述，通过对该全局能量函数进行最小化处理得出最优视差值。全局立体匹配算法的匹配效果要好于局部立体匹配算法，在弱纹理区域和遮挡区域的匹配效果明显更好。但是，由于需要建立全局能量函数并优化，全局匹配算法在计算速度上远落后于局部匹配算法[47]。

全局能量函数定义为式(2-25)

$$E(d)=E_{data}(d)+E_{smooth}(d) \tag{2-25}$$

公式(3-11)中，$E_{data}(d)$ 称为数据项，代表图像对之间的相似程度，即整幅图的匹配代价总和；$E_{smooth}(d)$ 称为平滑项，表示相邻像素之间在视差上的相互影响的关系；如果相邻像素间的视差差异愈大，$E_{smooth}(d)$ 就愈大；如果相邻像素的之间，其视差的差异越小，$E_{smooth}(d)$ 就越小。

常用的全局匹配算法有基于马尔科夫随机场的算法、基于动态规划的算法、基于遗传算法的算法、基于神经网络的算法和基于置信传播的算法等[48]。全局匹配算法相比于局部匹配算法，效果更好、误差更小，但是计算普遍较为复杂，目前没有实时性较高的算法，并且不利于硬件实现，通常应用于研究领域或者对处理速度要求不高的环境中。

2.4 本章小结

本章主要介绍了双目立体视觉的相关理论，双目立体视觉的基本原理、成像坐标系及其相互转换关系及视差理论，论述了视差的由来，给出了视差与深度的关系，并介绍立体匹配的约束条件，分析了不同立体匹配算法的优缺点。

第 3 章 基于 Census 变换的区域匹配算法研究与改进

3.1 基于 Census 变换的立体匹配算法

非参数化的区域匹配算法在一定程度上解决了光照不均所引起的问题，而且运算相对简单，易于利用 FPGA 实现。基于 Census 变换的区域立体匹配算法对于光照不敏感，鲁棒性强，目前已经成为应用广泛的立体匹配算法。

3.1.1 Census 变换

1994 年，Ramin Zabih 和 John Woodfill 提出了 Census 变换[49]，并且在之后的十几年的时间内得到了广泛的应用。Census 变换采用待匹配点的领域元素与待匹配点之间的大小关系代替灰度值作为特征，提高了匹配率。

Census 变换将一个匹配窗口中的各元素与中心元素进行比较，如果小于中心元素，就置为 1，否则置为 0。这样就可以把一个匹配窗口转换为一个比特串（bit string），如式(3-12)所示：

$$R_T(P(x,y)) = \underset{(i,j)\in W}{\otimes} \xi(P(x+i,y+j)) \tag{3-1}$$

公式(3-1)中，$P(x,y)$ 是图像中的任一像素点，W 是 P 的邻域窗口。$R_T(P)$ 表示 Census 变换，将 P 的邻域转换成一个比特串，这个比特串邻域像素与像素 P 之间的关系。$\otimes$ 表示连接操作，即将每一个属于 W 的像素经过 $\xi(A,B)$ 运算得到的结果连接在一起，形成一个比特串。Census 变换的具体使用方法如图 3-1 所示。$\xi(A,B)$ 表示比较运算，公式如(3-13)所示：

$$\xi(A,B) = \begin{cases} 1 & A < B \\ 0 & A \geq B \end{cases} \tag{3-2}$$

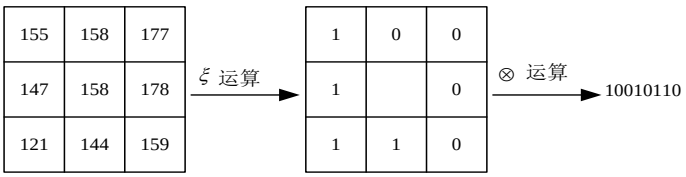


图 3-1 Census 变换过程

Fig.3-1 Process of Census transform

获得比特串以后，采用海明距离计算相似度。通常是先将两个比特串异或，

然后统计 1 的个数，如图 3-2 所示：

图 3-2 海明距离计算

Fig.3-2 Hamming distance calculation

确定了相似度测量函数之后，需要确定窗口的大小和视差搜索范围。当采用固定的窗口进行测量，假设窗口大小为 $N*N$。由于采用标准立体相机模型，只存在水平视差，因此视差搜索沿水平方向进行，降低了搜索维度。为了减少搜索的复杂度，并且要尽可能地保证精确度，本文的视差范围被假设为 $d_{\max}$。

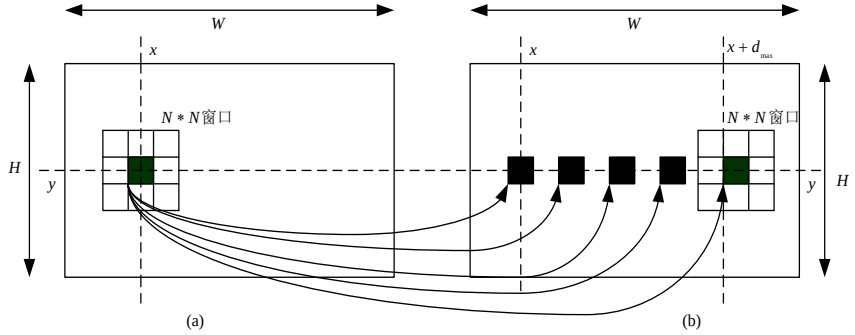


图 3-3 视差搜素过程

Fig.3-3 Processof disparity search

如图 3-3 所示，其中(a)为参考图，(b)为目标图。要求参考图中某点 (x,y) 的视差，则在目标图的 (x,y) 位置处开始，直到 $(x+d_{\max},y)$ 的位置结束。对其中的每一点采用 Census 变换算法求相似度，格局赢家通吃（Winner-Take-All，WTA）的方法，选择相似度最大的点作为匹配点，如果该点的位置在 $(x+d,y)$ 处，则点 (x,y) 的视差为 d。采用这种方法，对图像中的每一个点采用相同的处理方法，最终得到完整的视差图。

3.1.2 稀疏 Census 变换及改进

图像在经过 Census 变换后，所得比特串维数过大是 Census 变换的不足之处。例如，对于以一个 $5*5$ 的 Census 变换为例，进行变换前立体图对的每个像素灰度值只占用一个字节，但是经过变换后所得到比特串由 24 个比特组成，这就导致存储器占用量增加了 2 倍，因此匹配代价的计算量也成倍增加。这会在

很大程度占用硬件资源。为了这一问题，学者们提出了一些稀疏的 Census 变换方法(Sparse Census)[50]，其主要思想是在变换窗口中只选取一部分有代表性的像素与中心像素进行灰度值比较，从而减少产生比特串的维数，来达到降低计算量和资源占用率的目的。具有代表性的稀疏 Census 变换是 Mini-Census 变换，它的变换过程如图 3-4 所示：

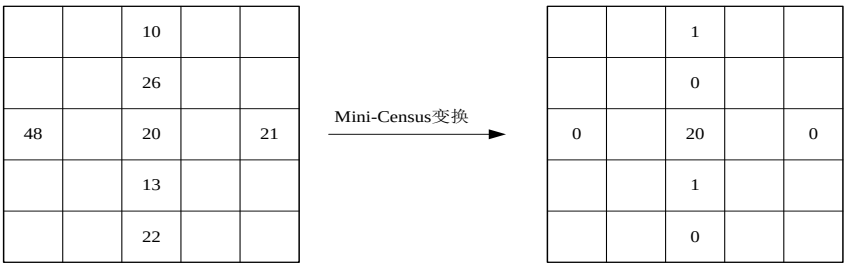


图 3-4 Mini-Census 变换过程

Fig.3-4 Process of Mini-Census transform

由图 3-4 可以看出，对于5*5的变换窗口，仅选择窗口内的一部分像素点，进行 Mini-Census 变换，所生产比特串的资源占用率不但低于 Census 变换所产生比特串的占用率，而且比原灰度图所占用的资源还要少。但是由于转化窗口内的大部分像素点没有参与变换，这就导致了经过 Mini-Census 变换得到的比特串所蕴含的相关信息量较低，会增大误匹配率。

为了克服 Mini-Census 变换中相关信息量不足的缺点，本文对 Mini-Census 变换提出一种改进，即将变换窗口内的某个像素点同时与邻域内所用像素点的灰度平均值、中心像素灰度值这两者进行比较，结果是一串 2 位的比特串。减小因为中心像素受到干扰和相关信息不足而带来的误匹配，从而提高匹配精度。求解变换窗口内任意一个像素点邻域内所有像素的平均值 $\overline{p}$，则邻域像素 p' 与邻域灰度均值 $\overline{p}$ 的灰度差异 δ 表示为：

$$\delta = p' - \overline{p} \tag{3-3}$$

邻域灰度平均值 $\overline{p}$ 作为补充信息，帮助分析窗口内各个像素之间的相关性，从而更准确地判断窗口内视差的变化。当中心像素 p 所属的窗口位于单一纹理区域时，中心像素 p 基本等于邻域灰度均值 $\overline{p}$，邻域像素 p' 的灰度值也接近邻域灰度均值 $\overline{p}$，此时 δ 值的会很小；当中心像素 p 所属的窗口位于视差不连续区域时，邻域像素 p' 与邻域灰度均值 $\overline{p}$ 的灰度会相差很大，δ 值也会很大。因此，邻域像素 p' 与邻域灰度均值 $\overline{p}$ 的灰度差 δ 的变化可以认为是图像视差变化的反映。而且当中心像素 p 受到噪声等影响而产生幅度失真的时，通过在变换

过程中引入邻域像素 p' 与邻域灰度均值 $\overline{p}$ 也可以减少失真对匹配结果的影响。

增加邻域像素的灰度平均值 $\overline{p}$ 作为一条约束条件，让邻域像素 p' 将同时与 $\overline{p}$ 和中心像素 p 做比较， p'、 p 和 $\overline{p}$ 之间的关系定义如式(3-4)所示。

$$\xi(p,p',\overline{p})=\begin{cases}00 & p'\le \min(p,\overline{p})\\ 01 & p<p'<\overline{p}\\ 10 & \overline{p}<p'<p\\ 11 & p'\ge \max(p,\overline{p})\end{cases} \tag{3-4}$$

在理论上，原 Mini-Census 变换经过改进，变换后利用两位二进制码来代替原来的一位码，增加了一个约束条件，使邻域内像素的相关信息得到体现，经过改进 Mini-Census 变换后所得的图像在视差不连续区域的可以得到更丰富的表示，减少了噪声对匹配质量的影响，提高了匹配精度。

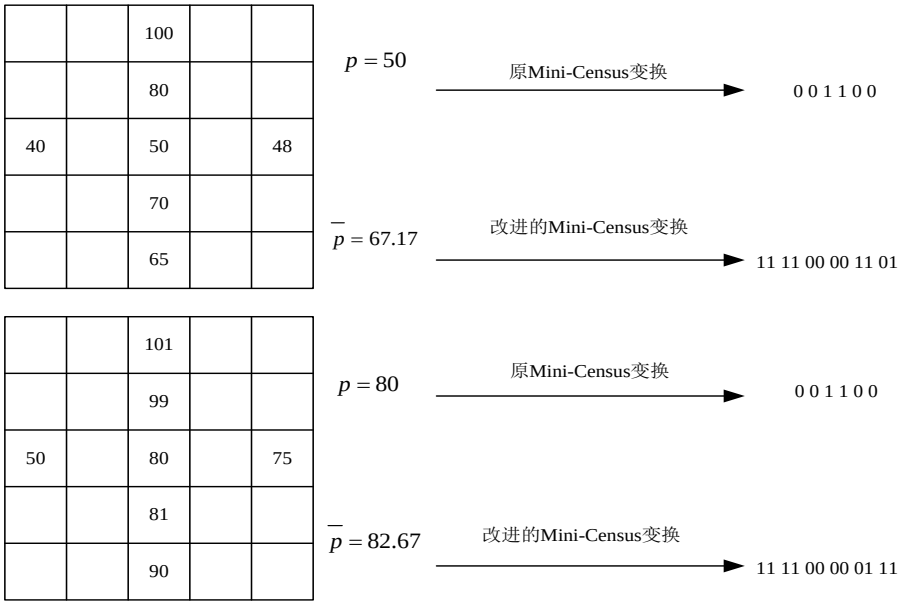


图 3-5 改进的 Mini-Census 变换与传统的 Mini-Census 变换比较

Fig.3-5 Comparison between improved Mini-Census and Mini-Census

选取 Tsukuba 测试图像中两个5*5区域分别进行 Mini-Census 变换和改进的 Mini-Census 变换，从图 3-5 中可以看出，当对两个区域的像素进行原 Mini-Census 变换时，所得到结果都是 001100，而当应用改进后的 Mini-Census 变换时，可以更加完整地描述匹配窗口区域内的像素信息，清晰地区分出所选择的两个区域之间的不同，提高了匹配精度。而且，加入邻域像素平均值这一约束条件，并没增大算法复杂度和运算量，改进后的 Mini-Census 仍然适于硬件实现。

3.2 自适应匹配窗口研究

3.2.1 匹配窗口研究

匹配窗口的选取是影响区域匹配算法精度的重要因素[51]：选择大窗口，会包含丰富的信息，在低纹理区域和重复区域匹配精度高，但是会在细节复杂、视差不连续的区域产生较大误差；选择的窗口小了，虽然可以在视差不连续区域获得较好的匹配效果，但是由于信息不足在低纹理区域的误匹配率高。常用的匹配窗口可以大致分为固定式和可变式[52]。采用固定窗口的匹配算法虽然简单、计算量，利于硬件实现，但误匹配率很高。当采用可变式窗口，会根据图像的细节信息、视差是否连续等情况动态地调节窗口的形状和大小，这样就大大提高匹配精度，而算法的复杂度却很高，不利于硬件实现。本文提出一种可以动态调节窗口大小自适应窗口的生成算法，算法不改变匹配窗口的形状，易于算法的硬件实现。

一般在像素灰度变化比较大的区域，对应的像素梯度值变化也比较剧烈。梯度值比较大的地方一般是视差变化不连续区域，匹配窗口大小应该设置得比较小；而对于梯度值较小的区域，往往是低纹理区域，此处匹配窗口应该设置得比较大。

-1	0	1
-2	0	2
-1	0	1

a)

-1	-2	-1
0	0	0
1	2	1

b)

图 3-6 水平梯度算子和垂直梯度算子

Fig.3-6 The horizontal gradient operator and vertical gradient operator

匹配窗口大小的设定与梯度有关，因此应用 Sobel 梯度算子计算每个像素点的梯度估计值。Sobel 是一阶导数边缘检测算子，是一种有效的梯度计算方法，具体方法是通过3*3模板作为核与图像中的每个像素点做卷积和运算。一个核对水平方向影响大、另一个对垂直方向影响大[52]。Sobel 算子的水平和垂直的运算核如图 3-6 所示，其中 a)为水平梯度算子，b)为垂直梯度算子。

利用 Sobel 算子计算图像中一点 $p(x,y)$ 的梯度幅值，需要以该像素点为中

心建立一个3*3的窗口，然后将窗口内所有像素分别与两个核做卷积获得水平分G_x和垂直分量G_y如公式(3-5)~(3-8)所示。

$$G_x=\begin{pmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{pmatrix}*\begin{pmatrix}p_1 & p_2 & p_3\\ p_4 & p(x,y) & p_5\\ p_6 & p_7 & p_8\end{pmatrix} \tag{3-5}$$

$$G_x=-p_1+p_3-2p_4+2p_5-p_6+p_8 \tag{3-6}$$

$$G_y=\begin{pmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & 1\end{pmatrix}*\begin{pmatrix}p_1 & p_2 & p_3\\ p_4 & p(x,y) & p_5\\ p_6 & p_7 & p_8\end{pmatrix} \tag{3-7}$$

$$G_y=p_1+2p_2+p_3-p_6-2p_7-p_8 \tag{3-8}$$

通过两个分量就可以计算像素点$p(x,y)$的梯度幅值$G(x,y)$，如公式(3-9)所示：

$$G(x,y)=\sqrt{G_x^2+G_y^2} \tag{3-9}$$

因此在动态调节匹配窗口的大小，需要设定了一个阈值T，并分别设置两个匹配窗口尺寸：N_1和N_2（$N_1<N_2$）。当某个像素点的梯度幅值$G(i,j)$大于或等于阈值T时，匹配窗口的尺寸设定为N_1*N_1，否则匹配窗口的尺寸设定为N_2*N_2。

3.2.2 自适应阈值设定

如果每次都人为地设定阈值T，不仅不利于立体匹配算法的硬件实现，而且影响了算法的实时性。提出一种引入中值滤波的自适应阈值生成算法。中值滤波是一种线性滤波，是把滤波窗口内的所有像素点进行排列，取中间值作为滤波结果，可以很好地保护图像的边缘信息[53]。

在一个阈值判断的模块中，设像素点为$p_1,p_2,\cdots,p_n$，其中$n\geq 3$，求自适应阈值的主要步骤如下所示。

第一步：第一次分组，将每 3 个连续的像素点分为一组，得到集合$X_1,X_2,\cdots,X_n$：

$$X_1=\{p_1,p_2,p_3\} \tag{3-10}$$

$$X_2 = \{p_4, p_5, p_6\} \tag{3-11}$$

……

$$X_n = \{p_{3n-2}, p_{3n-1}, p_{3n}\} \tag{3-12}$$

第二步：对第一组数据进行第一次排序：

$$Y_1 = Min\{p_1, p_2, p_3\} \tag{3-13}$$

$$Y_3 = Max\{p_1, p_2, p_3\} \tag{3-14}$$

$$Y_2 = \overline{\{p_1, p_2, p_3\} \cap \{Y_1, Y_3\}} \tag{3-15}$$

同理可以计算得到第 n 组的排序：

$$Y_{3n-2} = Min\{p_{3n-2}, p_{3n-1}, p_{3n}\} \tag{3-16}$$

$$Y_{3n} = Max\{p_{3n-2}, p_{3n-1}, p_{3n}\} \tag{3-17}$$

$$Y_{3n-1} = \overline{\{p_{3n-2}, p_{3n-1}, p_{3n}\} \cap \{Y_{3n-2}, Y_{3n}\}} \tag{3-18}$$

第三步，第二组分组，将第二步排序再分成三组，设集合为：

$$Z_1 = \{Y_1, Y_4, \cdots, Y_{3n-2}\} \tag{3-19}$$

$$Z_2 = \{Y_2, Y_5, \cdots, Y_{3n-1}\} \tag{3-20}$$

$$Z_3 = \{Y_3, Y_6, \cdots, Y_{3n}\} \tag{3-21}$$

第四步，第二次排序，在第三步中完成的分组中计算：设，

$$K_1 = Min\{Y_1, Y_4, \cdots, Y_{3n-2}\} \tag{3-22}$$

对 Z_2 先进行升序排列，可以得到：

$$Z_2 = \{L_2, L_5, \cdots, L_{3n-1}\} \tag{3-23}$$

再取出排在中间的值：

$$K_2 = L_{\frac{3n+1}{2}} \tag{3-24}$$

$$K_3 = Max\{Y_3, Y_6, \cdots, Y_{3n}\} \tag{3-25}$$

第五步，求出自适应阈值：

$$T = \sum_{i=1}^{3} \frac{K_1 + \cdots + K_i}{3} \tag{3-26}$$

K_i 的值随着图像中的各个像素点的灰度值在不断地变化，此时阈值的设定可以达到求得自适应的效果。自适应阈值生成算法的执行过程如图 3-7 所示。

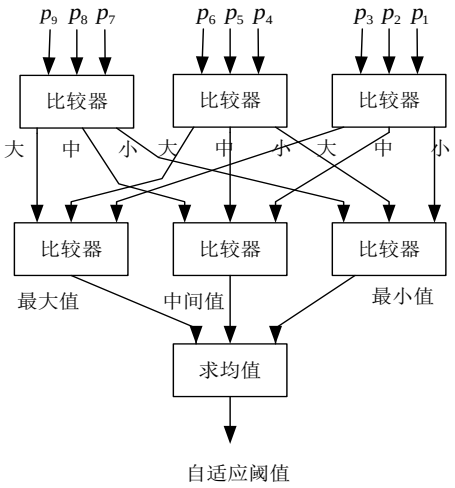


图 3-7 自适应阈值算法框图

Fig.3-7 Diagram of adaptive threshold algorithm

当计算图像中某一像素点 p 的自适应动态阈值时，先使其成为一个3∗3的滑动窗口的中心像素，然后对 $p_1 \sim p_9$ 这 9 个点根据公式(3-10)到公式(3-26)计算该点的自适应阈值，最后比较该点的梯度与阈值间的关系确定匹配窗口大小。

3.3 视差图后处理

3.3.1 左右一致性检测

遮挡问题是困扰立体匹配的“病态”问题如图 3-9 所示。由于双目相机在不同的角度拍摄，会出现在其中一幅图像中出现的物体而在另一幅图像中被前景物体遮挡的情况，这就是遮挡问题产生的原因。如图 3-9 所示，Cones 左右图

像中，由于两幅图像的拍摄角度不同，导致图像中的一部分在另一幅图像中被遮挡了，这就是典型的遮挡问题。因此为得到精确的视差图，需要将遮挡部分检测出来，然后进行填补。检测遮挡部分就用到了左右一致性检测（Left-Right，LRC check）。

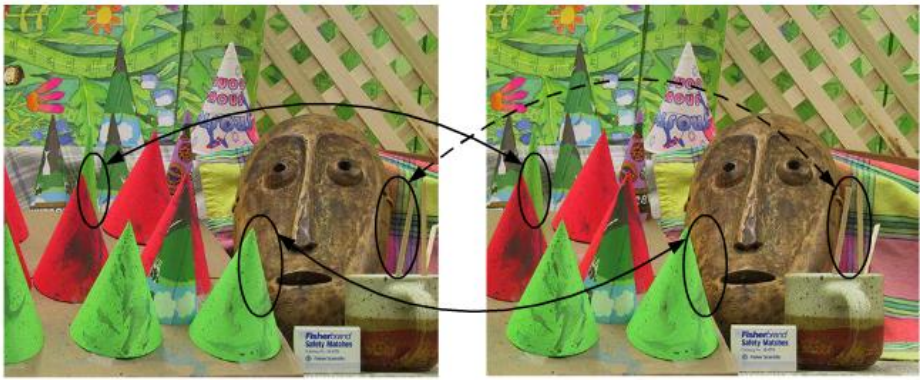


图 3-9 遮挡问题示意图

Fig.3-9 Schematic diagram of the occlusion problem

在一般情况下，以左图为参考图和右图为参考图使用相同的算法做双向匹配时，所得到的匹配点对应该是相同的。即若以左图为参考图，A 点在右图的匹配点为 B，则以右图为参考图时点 B 在左图的对应匹配点应为点 A，不满足这一约束条件的部分就是遮挡区域。通过双向匹配，可以完成左右一致性检测。

$$\left|d_{RL}(x,y)-d_{LR}(x+d_{RL}(x+d,y)\right|<T \tag{3-27}$$

公式(3-27)中，d_{LR} 和 d_{RL} 分别是以左图为参考图和以右图为参考图所得到的视差图，T 是允许的误差，满足公式(3-27)的点可以认为是正确的点，否则认为是遮挡区域，需要进行填补。

3.3.2 遮挡区域的填补

经过 LRC check 检测出遮挡区域以后，为了得到完整的视差图，需要填补，常用的插值算法大多需要使用一些复杂的计算，不适用于 FPGA 实现。

由立体匹配的约束条件中的连续性约束和顺序约束可知，相邻像素的视差值一般相差不大，因此可以使用遮挡点周围领域的像素的视差值对遮挡点进行填补。针对这一点，本文采用投票表决算法（Voting Method）对遮挡区域进行差值填补。

投票表决算法通过分析目标点邻域的像素信息，选取领域中出现次数最多的视差值代替目标点的视差值，通常选择一个以目标点为中心的窗口作为邻域。

此时的邻域窗口选择仍然按照前面生成的自适应窗口。如图 3-10 所示，针对目标点的邻域，统计邻域内的每一个视差值出现的次数，然后选取其中出现次数最多的视差值（如图中的 50），将该值代替目标点的视差。

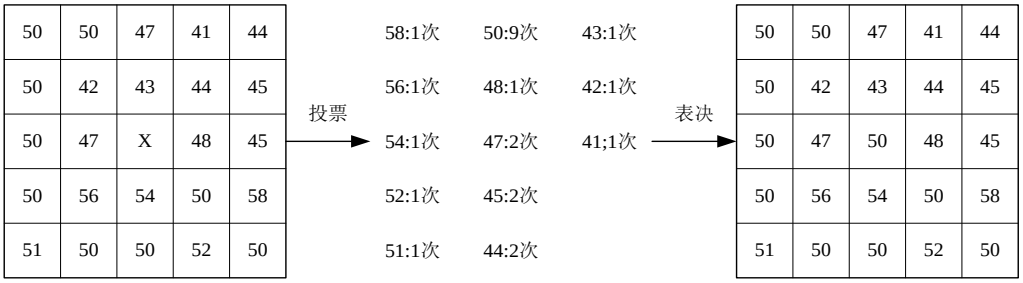


图 3-10 投票表决算法流程实例

Fig.3-10 The process instance of voting algorithm

投票表决算法在实现时只需要用到加法运算和比较运算，适合 FPGA 快速实现。与此同时，该算法充分利用到目标点的邻域信息，在实际应用中有很好的效果。

3.4 本章小结

本章详细介绍分析了基于 Census 变换的区域匹配算法，并根据存在的问题，提出了改进方案，包括对变换方式、匹配窗口选取、视差图细化及解决遮挡问题的相关算法的改进。

第 4 章 基于 FPGA 的立体匹配算法实现

4.1 系统整体结构

基于 FPGA 的立体匹配算法硬件实现系统的整体结构如图 4-1 所示，主要分成三个部分，即图像缓存模块、改进的 Mini-Census 变换模块和后处理模块。其中，图像缓存模块由 DDR2 储存已有的标准测试图像对，然后传送给改进的 Mini-Census 变换模块计算视差，得到左视差图和右视差图，通过左右一致性检测(left-right consistency，LRC)模块进行左右一致性检测，最后将检测到的遮挡区域经过填补模块进行插值，从而得到完成的视差图。各个模块的设计充分利用 FPGA 可以并行工作的特点，并采用流水线思想，最大限度地利用每一个时钟周期，提高算法的处理速度。

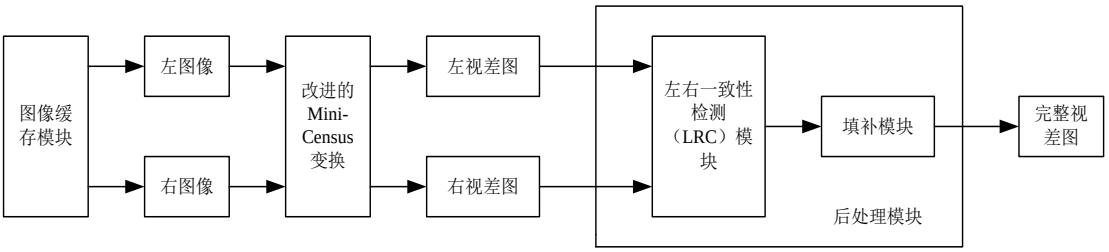


图 4-1 基于 FPGA 的立体匹配算法的整体框图

Fig.4-1 Overall block diagram of the stereo matching algorithm based on FPGA

4.2 图像缓存模块

图像缓存模使用 SDRAM 实现，SDRAM 具有容量大、速度快、价格低等优点，但是它由于逻辑复杂、时序严格，其驱动很复杂。文本利用 FPGA 设计一种 SDRAM 的控制器。为了提高图像数据传输的速度和稳定性，利用分时复用的办法使一片 SDRAM 上的不同 Bank 实现乒乓操作。

4.2.1 SDRAM 控制器设计

对 SDRAM 进行读写操作前，需要先对 SDRAM 进行初始化，初始过程如图 4-2 所示。即设置 SDRAM 的寄存器是普通模式还是扩展模式，确定其工作

方式，包括突发长度、突发类型、CAS 潜伏期等。

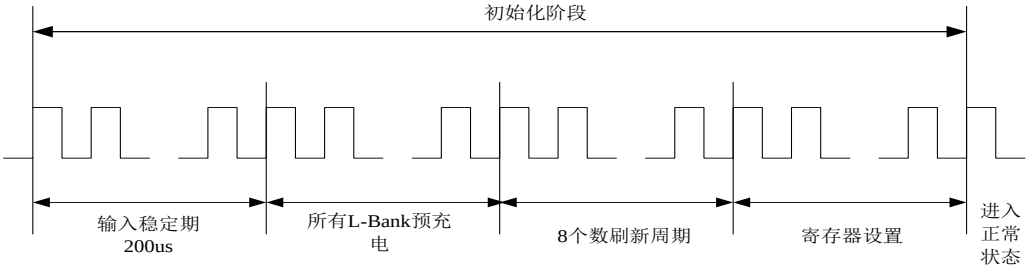


图 4-2 SDRAM 的初始化

Fig.4-2 The initialization of SDRAM

首先上电后，需要等待 200us，SDRAM 进入输入稳定期，在此期间不能对 SDRAM 做任何操作；200us 以后对所有的 L-Bank 预充电，之后然后对 SDRAM 刷新 8 次刷;最后设置 SDRAM 的模式寄存器。这样就可以对 SDRAM 进行正常读写了。SDRAM 的数据读写时序如图 4-3 所示。

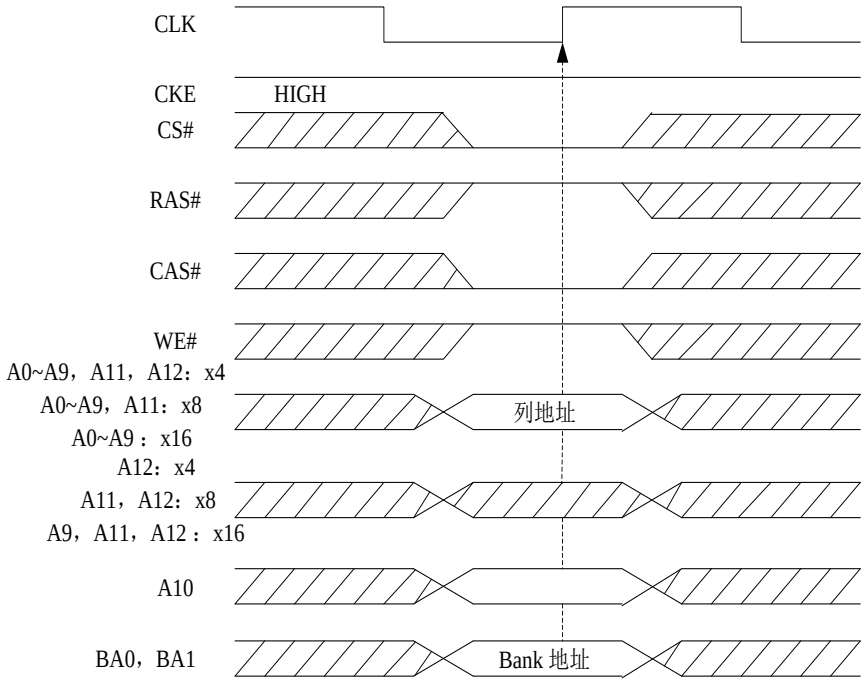


图 4-3 SDRAM 的读写时序

Fig.4-3 Read and write timing of SDRAM

利用 FPGA 设计 SDRAM 控制器，需要对所要实现的逻辑功能做细分，将

其划分成多个子模块。如图 4-4 所示。

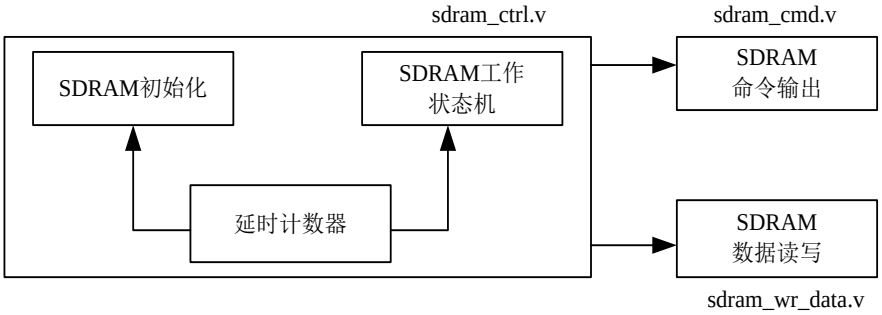


图 4-4 SDRAM 控制器功能模块

Fig. 4-4 The controller's function module of SDRAM

SDRAM 控制器用到了 3 个底层模块，及 sdram_ctrl.v、sdram_cmd.v、sdram_wr_data.v。sdram_ctrl.v 是 SDRAM 状态控制模块，主要完成 SDRAM 的初始化、刷新、读写等状态的转移。sdram_cmd.v 是 SDRAM 的命令模块，用来根据不同状态输出相应的控制命令和地址。sdram_wr_data.v 是读写模块，根据不同状态对 SDRAM 进行总线控制完成读写操作。SDRAM 的控制器的状态机如图 4-5 所示。

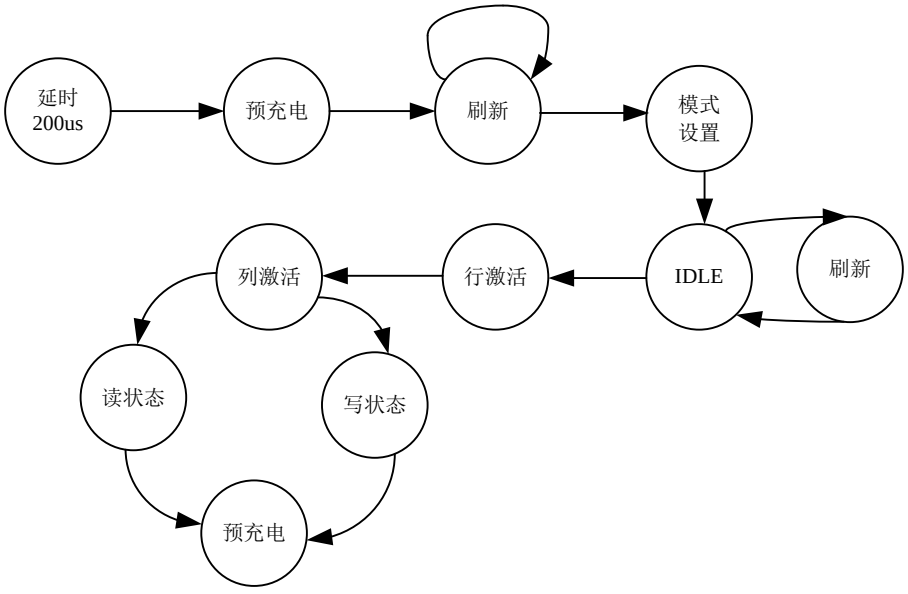


图 4-5 SDRAM 控制器状态机

Fig.4-5 The state machine of SDRAM’s controller

4.2.2 单片 SDRAM 实现乒乓操作

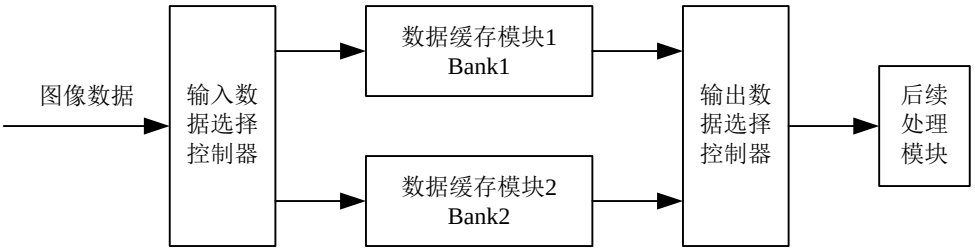


图 4-6 SDRAM 不同 Bank 实现乒乓操作

Fig.4-6 SDRAM Bank achieves the ping-pong operation

图 4-6 是利用一片 SDRAM 实现乒乓操作的示意图，其中利用 2 选 1 数据选择器作为输入数据选择控制器和输出数据选择控制器。用 SDRAM 的不同 Bank 作为数据缓存。

在第 1 个缓冲周期，将输入的图像数据缓存到数据缓存模块 1，即 Bank1 中。在第 2 个换冲周期，输入数据选择控制器将输入的图像数据缓存到数据缓存模块 2，即 Bank2 中；同时，输出数据选择控制器将 Bank1 中的在第 1 个周期缓存的数据传送到后续处理模块。在 3 个缓冲周期，输入数据选择控制器再次切换，将输入数据缓存到 Bank1 中；与此同时，输出数据选择控制器也进行切换，将 Bank2 在第 2 个缓冲周期存储的数据传送到后续处理模块中。如此不断循环，实现了数据的无缝传输，又提高了数据的处理速度。

由于只是用一块 SDRAM 来实现乒乓操作，其数据线和地址线是共用的，所以需要采用分时复用的办法，来达到读写时数据线和地址线的调用，即当一帧图像数据有效时，写操作优先，若写操作完成时，则开始读操作，同时写操作和读操作完成时进行 Bank 的切换。

4.3 Census 变换模块

改进的 Mini-Census 变换模块是系统的核心模块，采用并行化策略对改进的 Mini-Census 变换模块进行并行化加速，并行化的变换模块的硬件框图如图 4-7 所示。本系统以右图像作为参考图，左图像作为目标图，最大视差为 $d_{\max}$，则对于右图像中的像素 (x,y)，将同时与左图像中的像素 (x,y)~像素 $(x+d_{\max},y)$ 比较相似度，从中选择出相似度最高的匹配对，将两个点的位置差作为视差结果输出。其中左右图像经过缓存模块缓存，形成需要的 $N*N$ 的窗口，之后经过

变换转换成比特串，通过海明距离比较相似度，最终采用 WTA 策略选择出相似度最大的点，输出视差值。

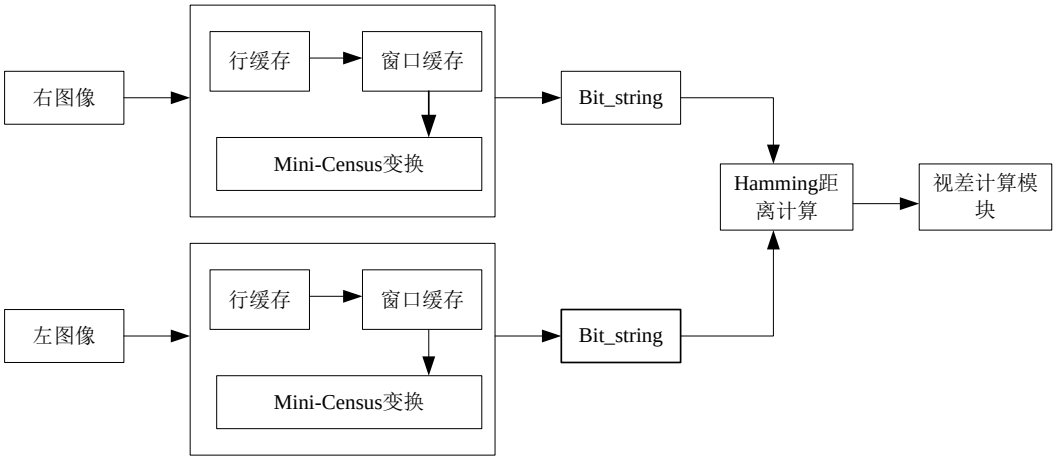


图 4-7 变换模块的硬件框图

Fig.4-7 The hardware diagram of transform module

4.3.1 Mini-Census 变换模块

变换模块将窗口的像素与中心像素和像素平均值进行比较，得出海明距离计算模块需要的比特串。变换模块的功能是缓存数据以及 Census 变换，图 4-8 为右图像的变换模块的状态机。

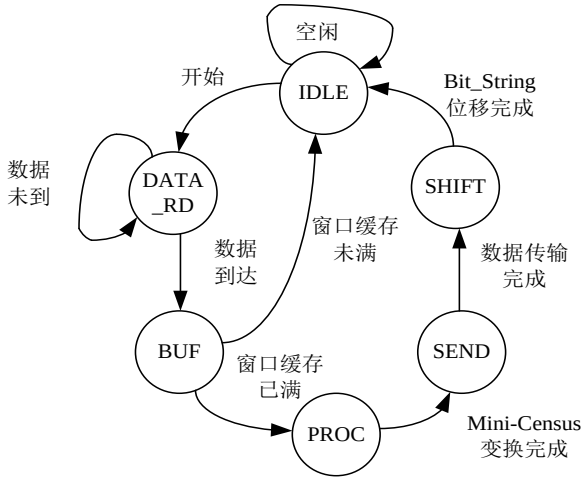


图 4-8 变换模块状态机转移图

Fig.4-8 State machine diagram of transform module

系统初始化之后，该模块处于 IDLE 状态，之后转入 DATA_RD 状态，向图像获取模块请求左图像数据，直到等到需要的数据，转入 BUF 状态。BUF 状态下，变换模块执行数据缓存，如果窗口缓存未满，转入 IDLE 状态，准备读入下一个像素；如果窗口缓存已满，则转入 PROC 状态。PROC 状态执行比较操作，得到比特串，完成后转入 SEND 状态。SEND 状态下该模块将比特串传入海明距离计算模块，之后转入 SHIFT 状态。SHIFT 状态是左图像变换模块特有的，用于完成公式(4-1) 的操作，完成后转入 IDLE 状态开始下一个像素的处理。对于右图像变换模块，不存在 SHIFT 状态，SEND 状态直接转入 IDLE 状态。

设新来的像素为 (x, y)，则有：

$$\begin{cases} bit\_string(i) \leftarrow bit\_string(i+1) & i \in [x - d_{\max}, x-1] \\ bit\_string(x) \leftarrow new \end{cases} \tag{4-1}$$

公式(4-1)， new 是右图像中以 (x, y) 为中心的窗口经过变换得到的比特串。

当窗口缓存已满，需要将窗口中的各个像素与中心像素和像素均值进行比较，需要 N^2-1 个 8bit 比较器，然后结果保存在一个 N^2-1 位的存储器中，硬件实现如图 4-9 所示。图中 N^2-1 个 8bit 比较器同时工作，在一个时钟周期内完成该工作，充分利用了 FPGA 的并行特性，如果采用串行指令结构，该操作至少需要 N^2-1 个时钟周期。

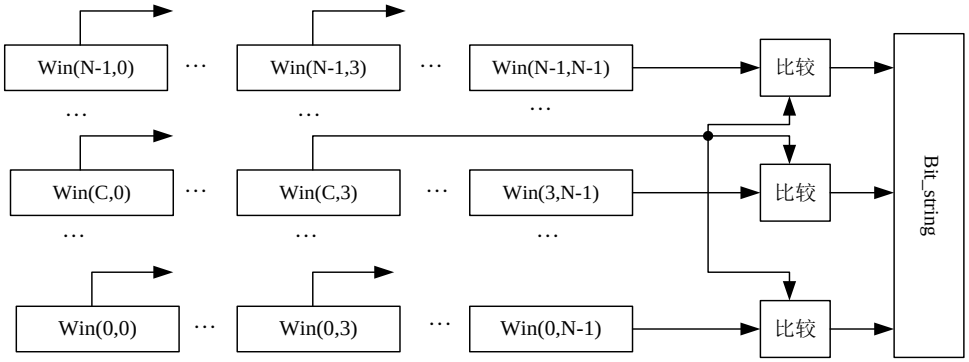


图 4-9 比较模块的硬件图

Fig.4-9 The hardware diagram of comparison module

4.3.2 匹配窗口设计

由于 FPGA 自身编程特点，无法将一个窗口内的数据通过数组来表示出来。

因此在使用硬件实现算法时需要一组与窗口尺寸相同的寄存器组来对图像数据进行缓存。使用 QuartusII11.1 软件中基于 RAM 的移位寄存器宏模块 Shift Register(RAM-based)来实现缓存窗口的设计，宏模块 Shift Register(RAM-based)可以定义数据宽度、移位的行数、每行的深度，同时还可以选择时钟使能端口 clken。如图 4-10 所示。

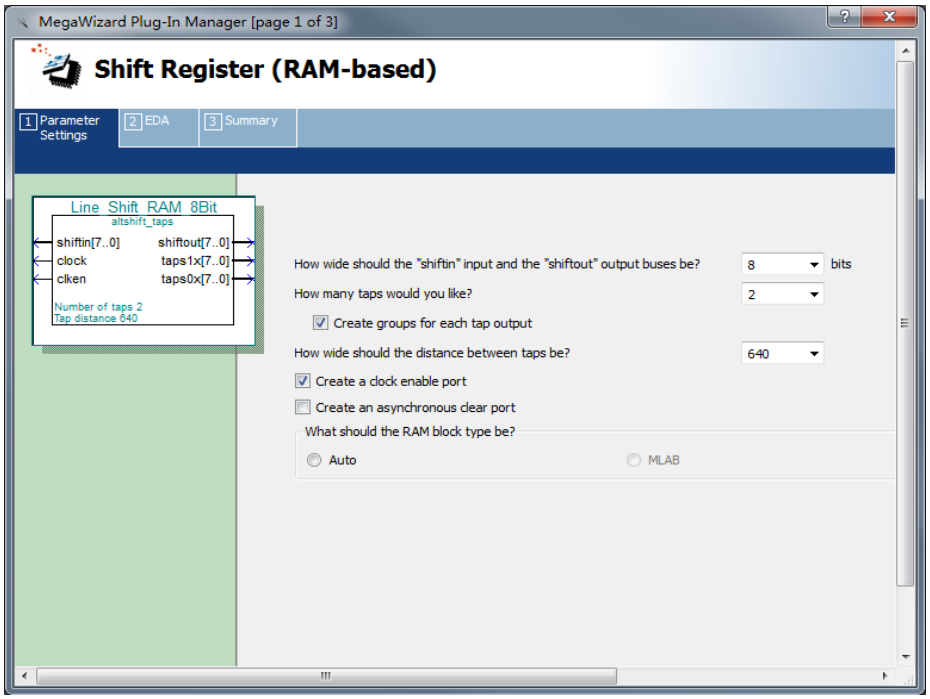


图 4-10 Shift Register 宏模块

Fig.4-10 Macro module of Shift Register

4.3.3 海明距离计算模块

海明距离计算模块的逻辑比较简单，状态机如图 4-11 所示。系统初始化之后，该模块处于 IDLE 状态，当变换模块的数据到达之后，转入 PROC 状态。在 PROC 状态下，进行海明距离的计算，计算完成后转入 SEND 状态。SEND 状态下，将数据传送给后续的视差计算模块，用于计算视差值，之后转入 IDLE 状态，开始下一次处理。

海明距离计算模块的硬件实现图如图 4-12 所示。Left_bitString 和 Right_bitString 为经过 Mini-Census 变换得到的左图和右图的比特串，之后经过异或运算，进入加法树统计 1 的个数，由于比特串为 N^2-1 位(N 为窗口大小)，

则得到的海明距离可以用 $\log_2(N^2-1)$ 位表示。假设 $N=7$，则海明距离可以用 6 位表示，加法树模块接收 48 位的字符串，可以采用三级加法树，第一级为 16 个 3 输入 1 位加法器，输出 16 个 2 位结果；第二级为 4 个 4 输入 2 位加法器，输出 4 个 4 位结果；第三级为 1 个四输入 4 位加法器，输出最终的 6 位结果。

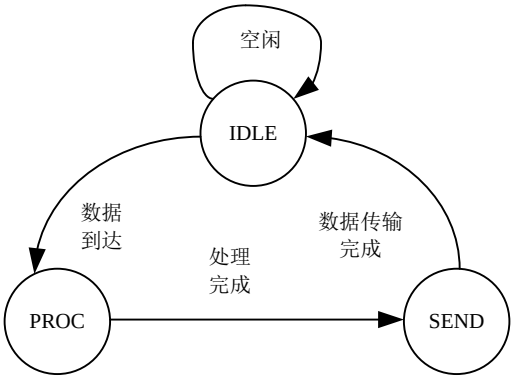


图 4-11 海明距离计算模块状态机

Fig.4-11 State machine diagram of the Hamming distance calculation module

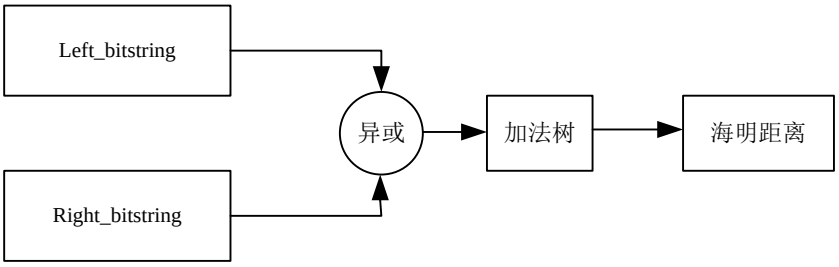


图 4-12 海明距离计算的硬件实现

Fig.4-12 The hardware diagram of the Hamming distance calculation module

4.4 视差计算模块

4.4.1 右视差计算模块

右视差计算模块的状态机如图 4-13 所示。系统初始化之后，该模块处于 IDLE 状态，等待数据到达，数据到达后转入 PROC 状态。PROC 状态下，该模块使用如图 4-14 所示的硬件模块计算视差值，计算完成后转入 SEND 状态。SEND 状态将右视差传送给后处理模块，之后转入 IDLE 状态，处理下一个数据。

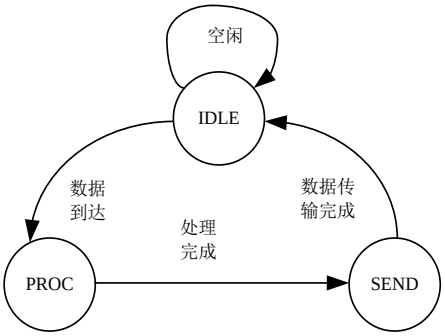


图 4-13 右视差图计算状态机

Fig.4-13 The state machine of right disparity map calculation

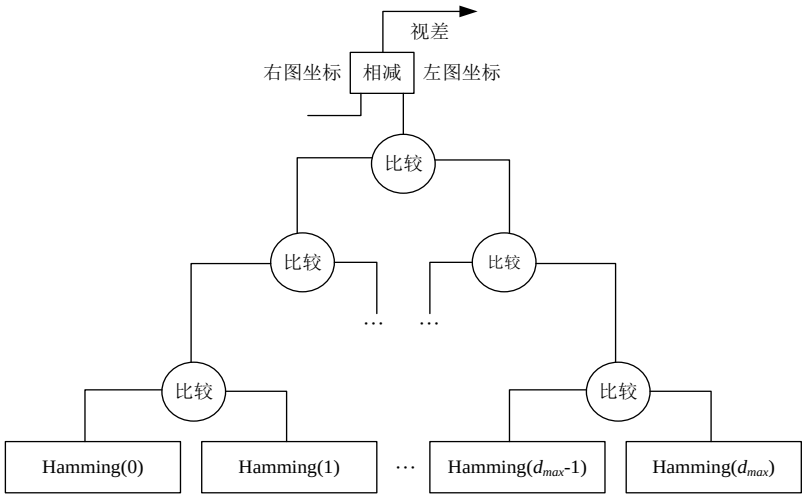


图 4-14 WTA 硬件实现

Fig.4-14 The hardware of WTA

右视差计算模块采用 WTA 策略，输入是海明距离计算模块的$d_{\max}+1$个海明距离，选取最小的海明距离，根据该点对的水平位置差计算视差，其硬件实现图如图 4-14 所示。WTA 的实现采用双路比较电路，$d_{\max}+1$个海明距离经过$\frac{d_{\max}+1}{2}$个比较器得出$\frac{d_{\max}+1}{2}$ 个结果，后面依次得到$\frac{d_{\max}+1}{4}$、$\frac{d_{\max}+1}{8}$… 1个结果，最终得到左图中与目标点相似度最大(海明距离最小)的点的坐标，与目标点坐标经过减法器运算，得到视差值。在每一级比较的时候，多个比较器并行工作，提高了计算效率。

4.4.2 左视差计算模块

如果计算左视差采用同计算右视差同样的硬件实现，不仅浪费了资源，而且会造成计算冗余。在计算右视差时，对于右图中的某个目标点 $P_R(x,y)$，要计算其对应的海明距离 $Ham\min g(P_R(x,y),P_L(x+i,y))\quad (i\in[0,d_{\max}])$。计算左视差时，对于左图中的某个点 $P_L(x+d_{\max},y)$，与之对应的海明距离是 $Ham\min g(P_R(x+d_{\max}-i,y),P_L(x+d_{\max},y))\quad (i\in[0,d_{\max}])$，这些数据在右视差计算 $P_R(x,y)\sim P_R(x+d_{\max},y)$ 时已经存在，只需要缓存起来用于左视差计算即可，如图 4-15 所示。采用这种方法可以有效地利用已计算数据，降低计算冗余，提高效率。

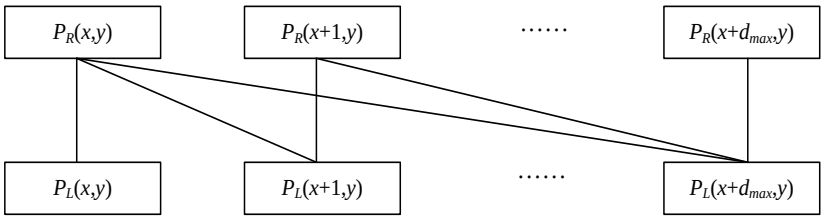


图 4-15 减少计算冗余

Fig.4-15 To reduce the computational redundancy

为了减少计算的冗余，需要将海明距离的计算结果保存起来。对于一个像素点 $P(x,y)$，会得到 $d_{\max}+1$ 个海明距离，而要计算做图像中某点 $P_L(x+d_{\max},y)$ 的视差，需要已知右图像中 $P_R(x,y)\sim P_R(x+d_{\max},y)$ 对应的海明距离，因而要缓存 $(d_{\max}+1)^2$ 个海明距离。可以采用 $d_{\max}+1$ 个双端口 RAM，每个双端口 RAM 最多存储 $d_{\max}+1$ 个数据，来实现缓存。

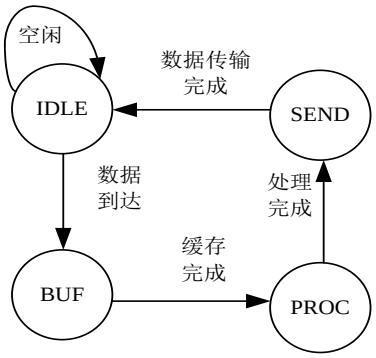


图 4-16 左视差计算状态机

Fig.4-16 The state machine of left disparity computation

与右视差计算模块相比，左视差计算模块需要加入一个数据缓存模块，对

应的状态机也略微复杂一些，如图 4-16 所示。其中，IDLE、PROC、SEND 状态所进行的操作均与右视差计算模块相同。不同的是加入了一个 BUF 状态，用于将数据缓存起来，并且选取正确的数据用于左视差计算。

4.5 后处理模块

4.5.1 左右一致性检测模块

左右一致性检测模块的目的是检测遮挡区域，需要根据左右视差执行双向匹配，将匹配不正确的点标记为遮挡区域的点，的状态机如图 4-17 所示。

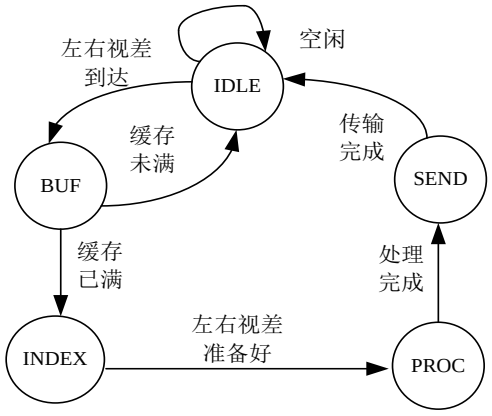


图 4-17 左右一致性检测状态机

Fig.4-17 The state machine of LRC

系统初始化之后，处于 IDLE 状态，左右视差到达后，转入 BUF 状态。BUF 状态下，将左右视差数据存储在左右视差缓存中，如果缓存未满，转入 IDLE 状态等待下一对视差数据到来;如果缓存已满,进入 INDEX 状态。INDEX 状态下，以右视差为索引，在左视差缓存中寻找对应的左视差，之后进入 PROC 状态。PROC 状态下，对左右视差进行比较，判断是否为遮挡区域的点，并且为遮挡区域的点加上标记，完成后转入 SEND 状态。SEND 状态将结果传送到后续模块，状态机转入 IDLE 状态，等待下一对视差数据到来。

针对右图中某个点 $(x,y$ 的视差 $d_{RL}(x,y$ ，需知道左图中的点 $(x+d_{RL}(x,y),y)$ 的视差 $d_{LR}(x+d_{RL}(x,y),y)$，。由于 $d_{RL}(x,y)$ 的取值范围是 $[0,d_{\max}]$，在进行左右一致性检测之前，需要知道视差 $d_{LR}(x,y)\sim d_{LR}(x+d_{\max},y)$，可以将这部分视差数据缓存起来。

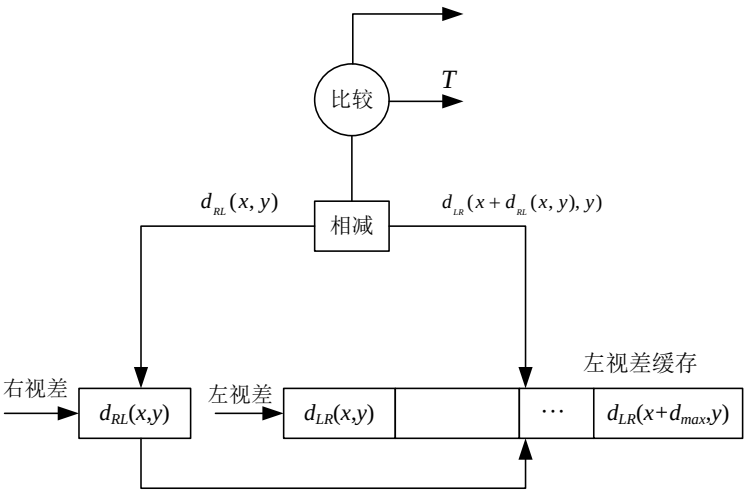


图 4-18 左右一致性的硬件实现图

Fig.4-18 The hardware of LRC

图 4-18 是左右一致性检测模块的硬件实现图。图中，对于右视差 $d_{RL}(x,y)$，先将该视差作为索引，在左视差缓存中找到对应的视差值 $d_{LR}(x+d_{RL}(x,y),y)$，之后 $d_{RL}(x,y)$ 与 $d_{LR}(x+d_{RL}(x,y),y)$ 进入减法器计算两者之间的差值，最后与阈值 T 作比较，如果小于 T 认为该点不是遮挡区域的点，否则，认为该点是遮挡区域的点。

4.5.2 投票算法模块

投票算法模块的状态机如图 4-19 所示，投票表决子模块的硬件实现图如图 4-20 所示。

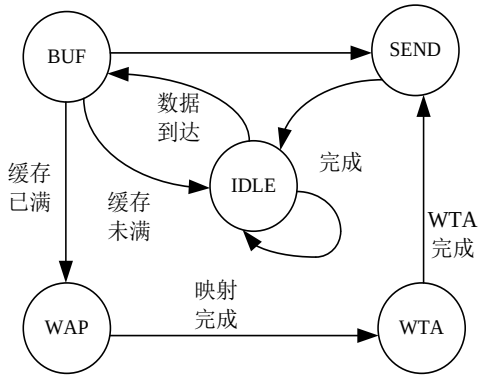


图 4-19 投票算法状态机

Fig.4-19 The state machine of voting algorithm

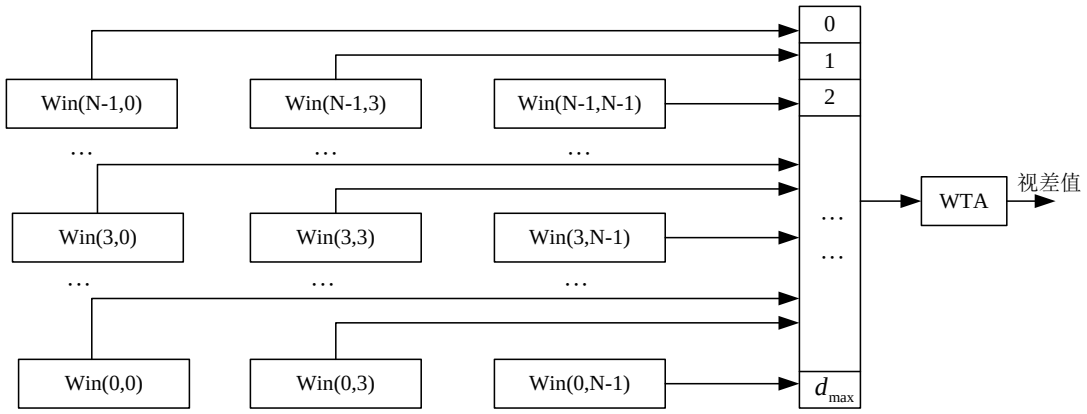


图 4-20 投票算法硬件实现图

Fig.4-20 The hardware of voting module

系统初始化之后，该模块处于 IDLE 状态，等待数据到达，数据到达后转入 BUF 状态。BUF 状态下，执行行数据缓存和窗口数据缓存，如果接收的视差值标记的是非遮挡区域，则转入 SEND 状态；否则，如果窗口缓存未满，则转入 IDLE 状态等待下一个数据的到来，如果窗口缓存已满，转入 MAP 状态。MAP 状态下，该模块根据视差值进行映射，统计窗口中每一个视差值出现的次数，然后转入 WTA 状态。WTA 状态下，根据赢家通吃策略选择出现次数最多的视差值，作为最终的视差结果，转入 SEND 状态。SEND 状态将最终的视差结果输出，完成后转入 IDLE 状态，等待下一个数据的到来。

图 4-20 中，系统采用 $d_{max}+1$ 个寄存器保存每个视差值出现的次数，根据窗口中的视差值直接映射到对应下标的寄存器中，将其中的次数加一，之后采用 WTA 选择出现次数最多的视差值作为结果输出。

4.6 本章小结

本章介绍了基于 FPGA 的立体匹配算法的实现，主要分为三个大模块，即图像缓存模块、Mini-Census 变换模块和后处理模块。每个大模块又细分为多个小模块，针对每个小模块，讲述了其硬件实现图和状态机，从而详细论述了整个系统的 FPGA 实现。

第 5 章 实验仿真结果分析

5.1 实验环境与评估标准

5.1.1 硬件平台

本课题的硬件选择的是 Altera 公司的 DE2 FPGA 开发平台，FPGA 芯片是 CycloneII 系列的 EP2C35F672，该芯片具有 33，216 个逻辑单元（Logic Element，LE）、105 个 M4K RAM 块、483,840 比特 RAM、35 个嵌入式乘法器及 4 个锁相环（ Phase Locked Loop，PLL）片上资源丰富，满足课题要求。选择 Altera 公司的 EPCS16 串行 FLASH 作为配置芯片，Zentel 公司的 8-MB SDRAM 作为 DDR2 存储芯片。

5.1.2 软件环境

由于选用的 FPGA 芯片的制造商是 Altera 公司，本系统采用 QuartusII 11.1 作为主要开发工具，Modelsim SE6.5 作为仿真工具，嵌入式在线逻辑分析仪 SignalTapII 作为在线调试工具。

5.1.3 评估方法

本实验主要针对匹配精度和实时性两个方面性能进行评估。Middlebury 提供了标准测试图片库，可以用于评估立体匹配算法，比较常用的 Tsukuba、Venus、Teddy 和 Cones 四种立体图像对，视差搜索范围分别为 $0\sim15$、$0\sim19$、$0\sim59$，如图 5-1 所示，其中 a)为 Tsukuba 图像对、b)为 Venus 图像对、c)为 Teddy 图像对、d)为 Cones 图像对。

目前通用的立体匹配算法精度的评估方法是根据计算所得的视差图与真实视差图作比较，计算误匹配率，误匹配率的计算公式如（5-1）所示：

$$B=\frac{1}{N}\sum_{(x,y)\in I}\left(\left|d_C(x,y)-d_T(x,y)\right|>\delta_d\right) \tag{5-1}$$

其中，N 是用于统计的像素总数，I 是计算误匹配率的区域，$d_C(x,y)$ 表示立体匹配算法得到的视差，$d_T(x,y)$ 表示视差实际值，δ_d 是误差阈值，一般

取 $\delta_d = 0.1$。

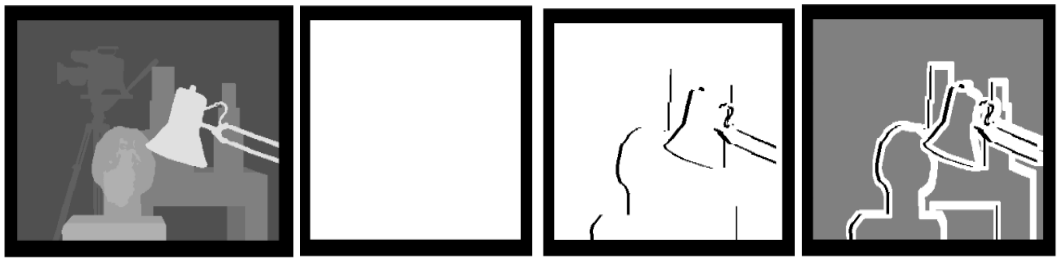
a) b)

c) d)

图 5-1 Middlebury 网站图像对

Fig.5-1 Image pairs of Middlebury website

在计算误匹配率时，主要采用 all、nonocc、disc 三种指标来衡量所得到的视差图。all 表示计算全图的误差率、nonocc 表示非遮挡区域的误差率、disc 表示深度不连续区域的无匹配率。Tsukuba 的视差图和三种指标对应的区域如图 5-2 所示，其中，a)为真实视差图、b)为 all 指标下的视差图、c)为 nonocc 指标下的视差图、d)为 disc 指标下的视差图，黑色部分表示不计算误匹配率，白色部分表示误匹配率。



a) b) c) d)

图 5-2 误匹配率计算指标

Fig.5-2 False matching rate

对于实时性的评估，本系统采用串口将左右图像传送到 FPGA，并将数据存储到 SDRAM 中，FPGA 从 SDRAM 中读取图像数据处理 10 次，记录处理时间，取平均值作为一次立体匹配处理的时间，来计算帧速率。

5.2 主要模块仿真

5.2.1 Mini-Census 变换模块

这个模块的主要功能是实现改进的稀疏 Census 变换，得到比特串，用于计算海明距离。图 5-3 是左图变换模块的仿真图，右图变换模块的仿真图与之相同。

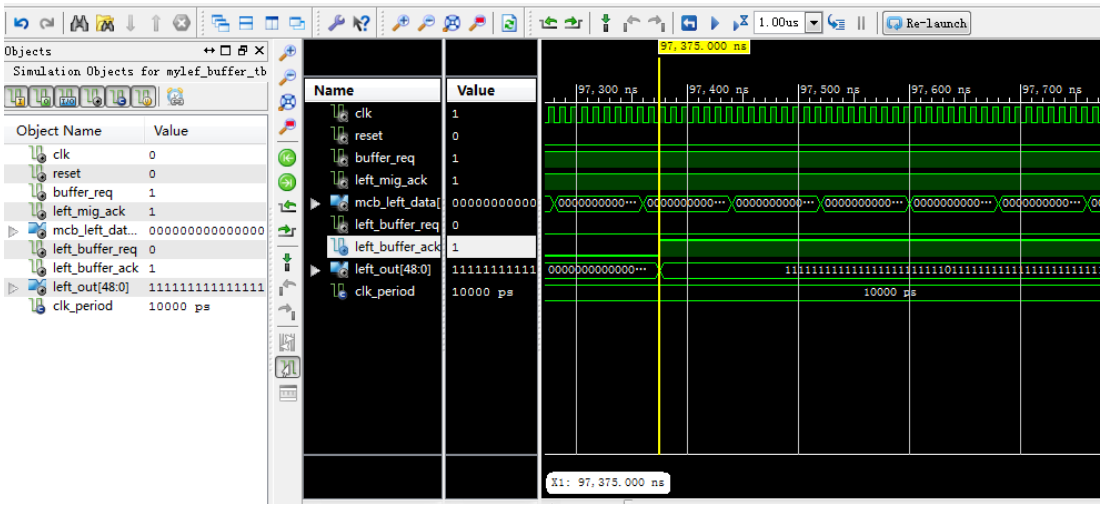


图 5-3 变换模块仿真图

Fig.5-3 Simulation diagram of transformation module

其中，*buffer_ re*是海明距离计算模块对变换模块的数据请求信号；*left _mig _ack* 和 *mcb _left _data* 是图像获取模块对变换模块的数据响应信号和图像数据；*left _buffer _req* 是变换模块向图像获取模块发送数据的请求信号；*left _buffer _ack* 是变换模块向海明距离计算模块发送数据的响应信号；*left _out* 是变换模块向海明距离计算模块发送的比特串数据。当 *left _buffer _ack* 为 1 时，表示 *left _out* 是有效数据，海明距离计算模块可以使用变换模块发送来的数据。

5.2.2 海明距离计算模块

海明距离计算模块接受左右变换模块发送来的比特串数据，计算相应的海明距离。本文的最大视差为 $d_{\max}=59$，因此需要计算 60 个海明距离，然后将结果传送给后续模块。海明距离计算模块的功能仿真图如图 5-4 所示。

图 5-4 中，*left _buffer _ack* 和 *right _buffer _ack* 分别是左右变换模块的数据传送响应信号；*ham _req* 是视差计算模块发送数据的请求信号；*ham _ack* 是海明距离计算模块向视差计算模块发送数据的响应信号；*buffer _req* 是海明距离计算模块向变换模块传送请求信号；*eff* 是边界掩码，用于标定边界数据；$r\_in0, r\_in1, \cdots\cdots r\_in59$ 是右图像变换模块发送的比特串，*1_in* 是左图像变换模块发送的比特串数据，通过海明距离计算模块分别计算 *1_in* 与 $r\_in0, r\_in1, \cdots\cdots r\_in59$ 的海明距离，60 个海明距离计算模块并行工作，将结果存放 *ham* 中发送给后续模块。

当 *left _buffer _ack* 和 *right _buffer _ack* 都是 1 时，表明左右变换模块的数据有效，海明距离计算模块开始计算海明距离，并且将结果放在 *ham* 里，*ham_ ack*置 1，向视差计算模块发送响应信号，之后 *buffer _req* 置 1，向变换模块请求新的数据。

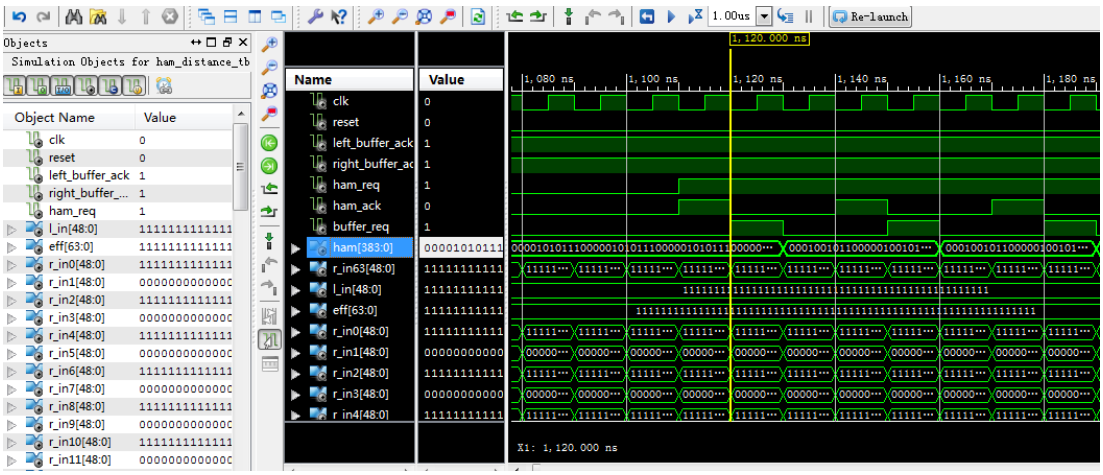


图 5-4 海明距离计算模块仿真图

Fig.5-4 Simulation diagram of hamming distance calculation module

5.2.3 视差计算模块

视差计算模块接收海明距离计算模块的结果，并且采用 WTA 策略选择最小的海明距离，并计算视差值。图 5-5 给出了右视差计算模块的功能仿真图。

图中，*ham_ ack*是海明距离计算模块向视差计算模块发送数据的响应信号；*r _req* 是左视差数据请求信号；*ham* 是海明距离计算模块发送来的计算结果；*ham_ req*是右视差计算模块向海明距离计算模块发送的数据请求信号；*r _ack* 是左视差计算模块向左右一致性检测模块发送的请求信号；*r _dis* 是左

视差计算模块向左右一致性检测模块发送的左视差数据。

当 *ham_ack* 为 1 时，表明海明距离已经计算完成，之后左视差计算模块采用 WTA 策略从 *ham* 中选择海明距离最小的点计算视差值，计算完成以后的左视差数据在 *dis* 中，并且1_*ack* 置 1，表明左视差数据有效，之后 *ham_req* 置 1，向海明距离计算模块请求数据。

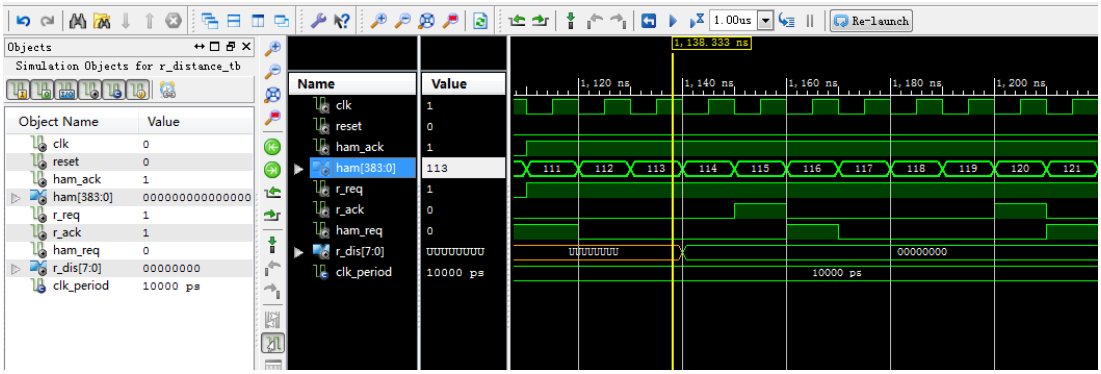


图 5-5 右视差计算模块的功能仿真图

Fig.5-5 Function simulation diagram of right parallax calculation module

左视差计算模块与右视差计算模块的功能仿真图相似，如图 5-6 所示。

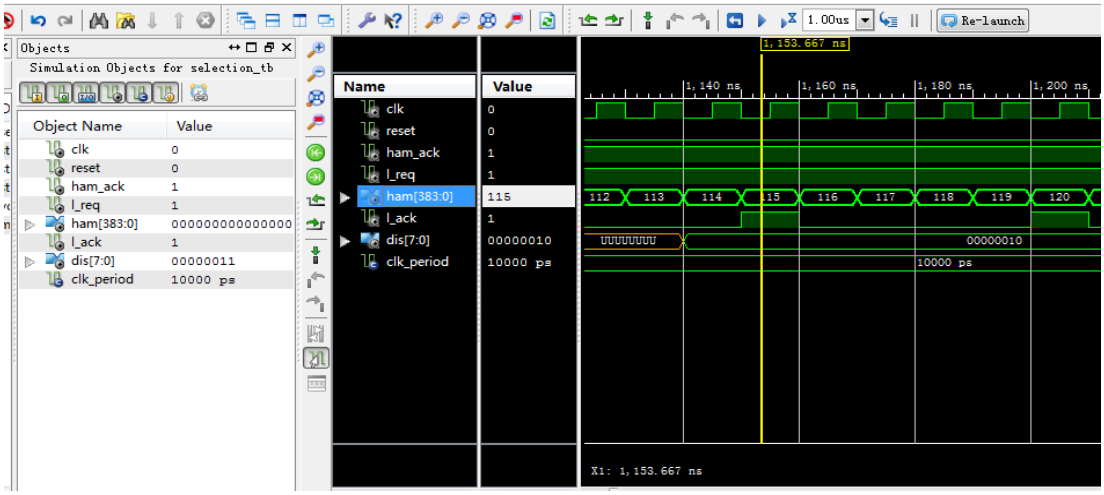


图 5-6 左视差计算模块的功能仿真图

Fig.5-6 Function simulation diagram of left parallax calculation module

5.2.4 左右一致性检测模块

左右一致检测模块检测遮挡区域，其功能仿真图如图 5-7 所示。图中，1_*dis*

与 *r_dis* 分别是左右视差计算模块得到的左右视差数据；*1_ack* 与 *r_ack* 分别是左右视差计算模块发送的数据请求信号；*1r_ack* 是左右一致性检测模块向投票表决模块发送的数据响应信号； *dis* 是左右一致性检测模块的检测结果。

当 *1_ack* 与 *r_ack* 均为 1 时，代表输入数据信号有效，左右一致性检测根据左右视差数据 *1_dis* 和 *r_dis* 进行检测，完成后将数据通过 *dis* 传送；若果是遮挡区域的点，置为全 0，否则为原数据，置 *1r_ack* 为 1，表明有效，之后，*1_reg* 与 *r_req* 置 1，向左右视差计算模块请求新的数据。

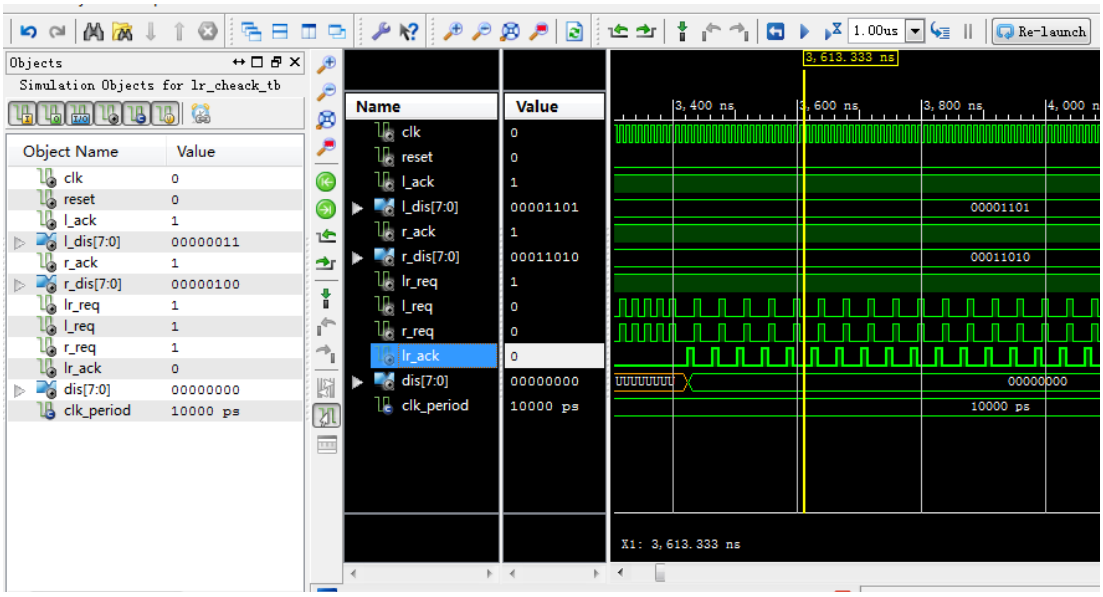


图 5-7 左右一致性检测模块仿真图

Fig.5-7 Simulation diagram of the left and right consistency checking module

5.2.5 投票模块仿真

投票表决模块功能仿真图如图 5-8 所示，图中，*1r_ack* 是左右一致检测模块向投票表决模块发送数据的响应信号；*1r_dis* 是左右一致检测模块的检测结果数据；*1r_reg* 是投票表决模块向左右一致检测模块发送的请求信号；*v_ack* 是投票表决模块对后续模块的相应信号；*v_req* 是后续模块对整个立体匹配算法实现模块的请求信号；*v_dis* 是整个立体匹配算法实现模块的最终视差结果。

当 *1r_ack* 为 1 时，表示左右一致检测模块完成，投票表决模块判断输入的视差数据是否为遮挡区域的视差，如果是遮挡区域，那么进行填补，否则直接输出，最终视差数据由 *v_dis* 传送，置 *1r_reg* 为 1，向左右一致检测模块请求

新的数据。

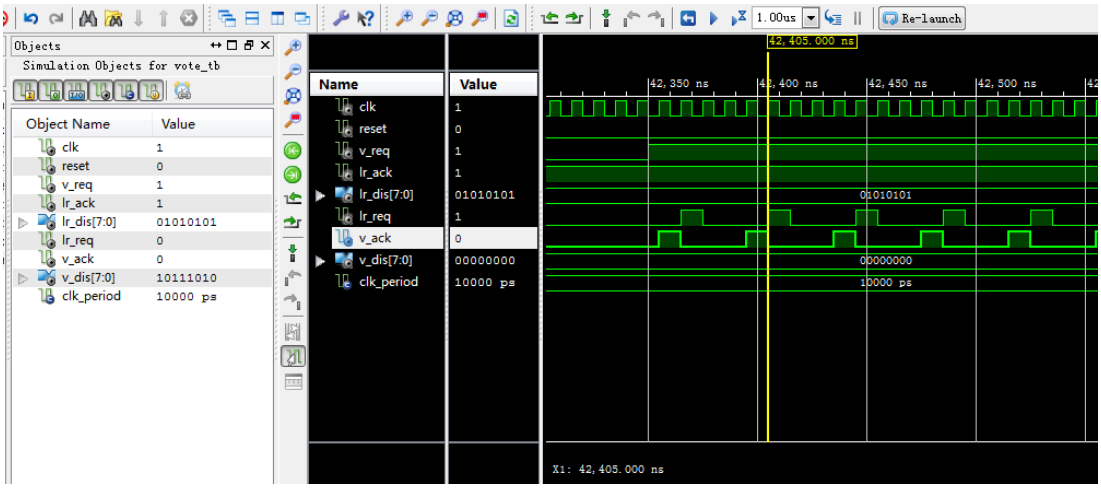


图 5-8 投票表决模块功能仿真图

Fig.5-8 Simulation diagram of voting module

5.3 实验结果分析

5.3.1 改进 Mini-Census 变换与原 Mini-Census 变换比较

首先，选择大小从3*3到23*23的固定匹配窗口做实验，通过误匹配率来比较原 Mini-Census 变换和改进的 Mini-Census 变换对匹配结果的影响，为了简化实验步骤及客观，实验没有进行代价聚合，仅对比较像素的匹配代价。误点率计算的阈值 δ_d 设置为 1。实验结果如图 5-9 所示，其中(a)为两种算法在处理图像 Tsukuba 时的比较结果、(b)为两种算法在处理图像 Venus 时的比较结果、(c)为两种算法在处理图像 Teddy 时的比较结果、(d)为两种算法在处理图像 Cones 时的比较结果。

由图 5-9 可以看出，改进的 Mini-Census 的匹配精度要明显高于原 Mini-Census。根据第 3 章可知，匹配窗口的大小直接影响着匹配的精度，窗口过小，则窗口内的像素信息不充分，导致匹配不精确；窗口过大，会包含过多的无关信息，也会降低匹配精度。为了比较不同窗口尺寸对匹配结果的影响，利用大小分别为3*3、5*5、7*7、9*9、11*11、13*13、15*15、17*17和19*19的匹配窗口使用改进的 Mini-Census，不同窗口下对 Tsukuba、Venus、Teddy 和 Cones 的匹配误差率，实验结果如图 5-10 所示，其中 a)为该算法在处理图像 Tsukuba 时的结果、b)为该算法处理图像 Venus 时的结果、c)为该算法处理图像

Teddy 时的结果、d)为该算法处理 Cones 时的结果

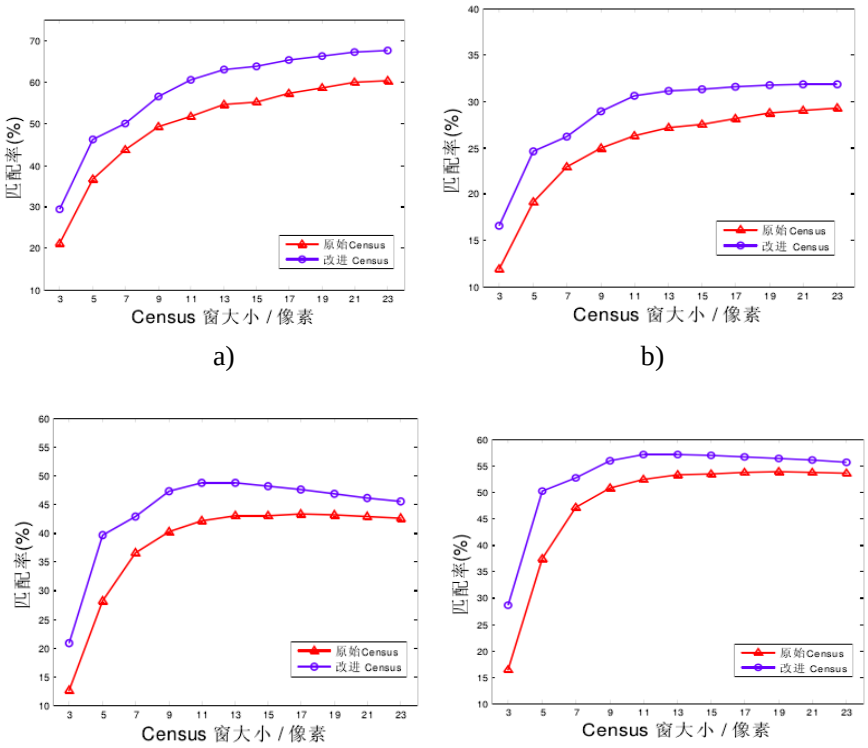


图 5-9 原 Mini-Census 变换和改进的 Mini-Census 变换的比较

Fig.5-9 Comparison of the original Mini-Census transform and the improved Mini-Census transform

由 5-10 可以看出，Census 窗口的大小对于匹配结果的影响很大，当窗口过小，比如3*3时，nonocc、all、disc 三个部分的误匹配率均很大。随着窗口逐渐扩大，nonocc 和 all 的误匹配率逐渐减少，直到窗口过大，比如19*19时，误匹配率又开始上升；而 disc 的误匹配率在窗口比较小，如7*7时，一般误匹配率比较低，之后随着窗口扩大误匹配率也随之增大，主要因为窗口在深度不连续区域的尺寸一般应该设置得较小，如果选择的窗口过大，会包含过多的深度连续区域的像素信息，从而降低匹配精度。

综上所述，匹配窗口大小的选择应该注意，既不能过小使得 nonocc 和 all 区域的误匹配率较大，也不能过大使得 disc 的误匹配率较大。所以本算法的自适匹配窗口的大小可设置为7*7、9*9和11*11，三者中任选二者进行搭配，这个窗口大小既保证了 nonocc 和 all 的误匹配率较低，也可以保证 disc 的误匹配率较低。

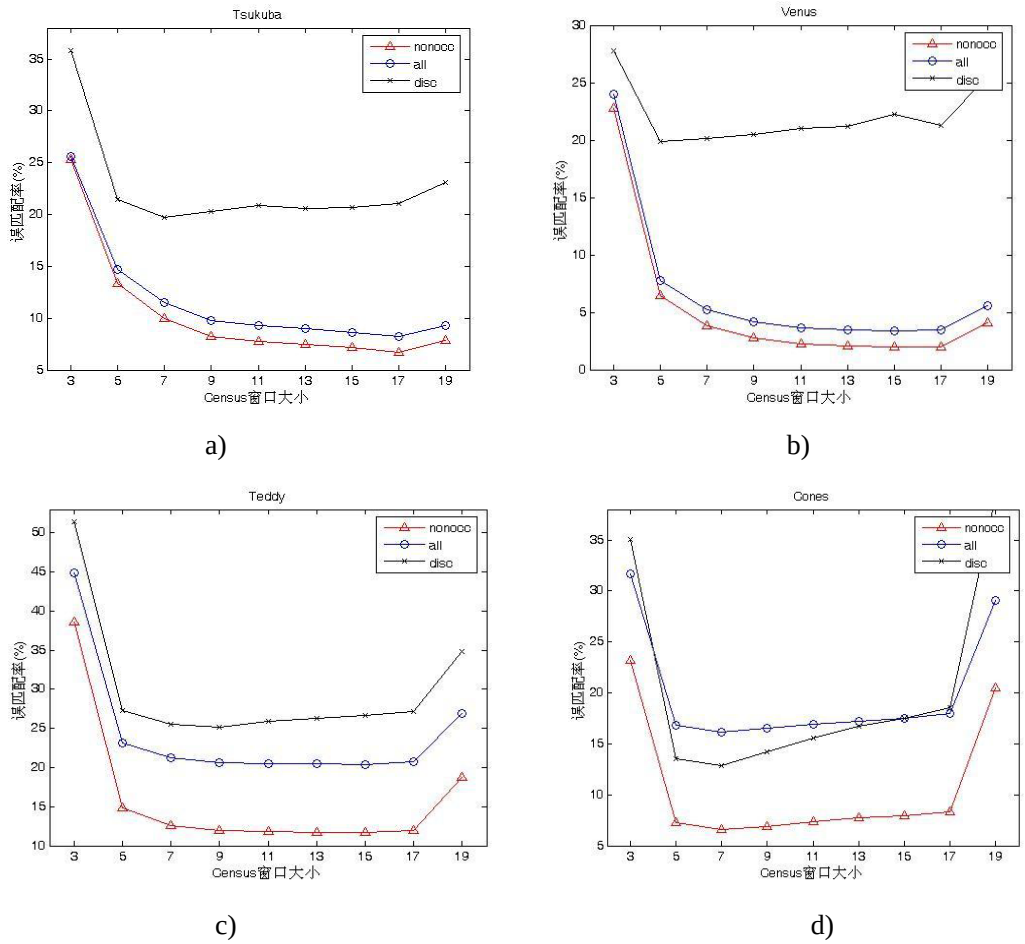
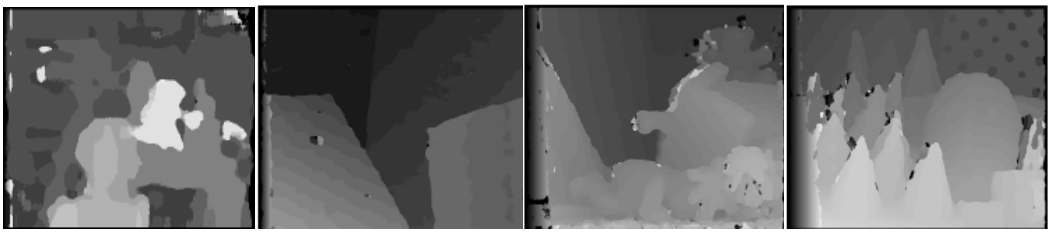


图 5-10 不同窗口的匹配误差

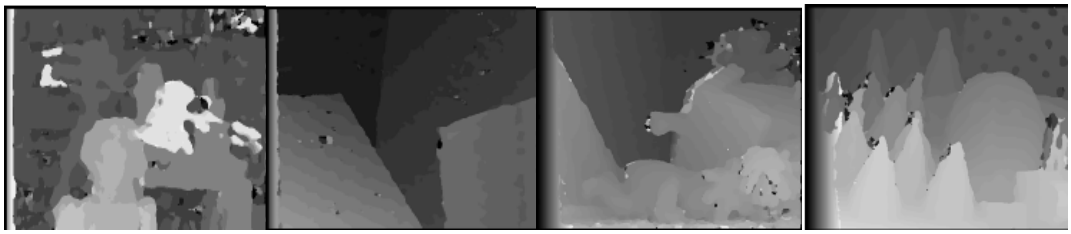
Fig.5-10 Matching error of different windows

基于改进的 Mini-Census 变换，自适应匹配窗口的大小选择7*7与9*9、9*9与11*11、7*7和11*11，三种情况下的匹配结果如图 5-11 所示，其中 a)为自适应匹配窗口的大小选择7*7与9*9时获得的四幅图像的视差图、b)为自适应匹配窗口的大小选择9*9与11*11时获得的四幅图像的视差图、c)为自适应匹配窗口的大小选择7*7和11*11时获得的四幅图像的视差图、d)为真实视差图。

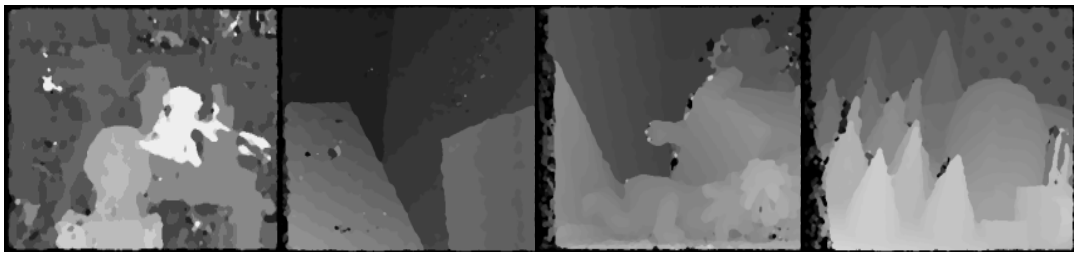
由图 5-11 的实验结果可以看出，选择7*7与9*9的窗口、9*9与11*11的窗口时的匹配结果比较接近，相比之下，9*9与11*11的窗口的匹配结果更接近真实视差，而选择7*7和11*11的窗口时，匹配结果相对前两者的误差较大。所以，在实现本文算法时，自适应窗口的大小可以选择9*9与11*11搭配所形成的匹配窗口。



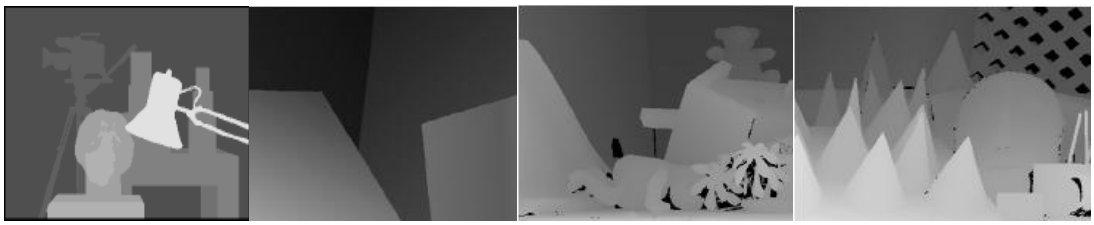
a)



b)



c)



d)

图 5-11 不同窗口的视差图

Fig. 5-11 Disparity map of difference windows

5.3.2 匹配精度分析

在评估本文算法时，选用 Middlebury 网站的测评平台及其图像数据库中的标准测试图像：Tsukuba、Venus、Teddy、Cones。并同一些相似的算法分别测试图像进行立体匹配，记录匹配误差率，测试得到的数据如表 5-1 所示。

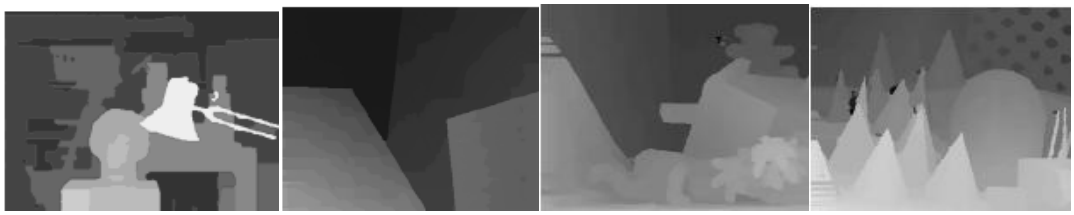
表 5-1 立体匹配算法精度比较

Table 5-1 Comparison of stereo matching algorithm accuracy

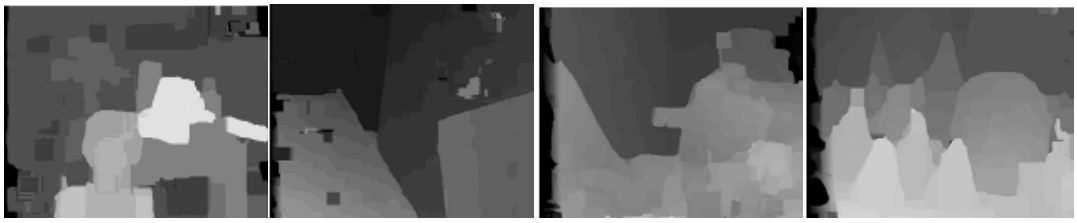
$\delta_d = 1$ 算法		Tsukuba	Venus	Teddy	Cones	平均误差率
ADCensus	nonocc	1.07	0.09	4.1	2.41	3.97
	all	1.48	0.25	6.22	7.25	
	disc	5.73	1.15	10.9	6.95	
RTCensus	nonocc	5.08	1.58	7.96	4.1	9.73
	all	6.25	2.42	13.8	9.54	
	disc	19.2	14.2	20.3	12.2	
SAD-IGMCT	nonocc	5.81	2.61	9.79	5.08	12.5
	all	7.14	3.33	15.5	11.5	
	disc	22.6	25.3	25.7	15	
SSD+MF	nonocc	5.23	3.74	16.5	10.6	15.7
	all	7.07	5.16	24.8	19.8	
	disc	24.1	11.9	32.9	26.3	
Rank+ASW	nonocc	6.51	10.5	15.7	14.1	18.4
	all	8.43	12	24.1	23.1	
	disc	19.7	32.7	32.8	24.7	
SemiGlob	nonocc	3.26	1	6.02	3.06	7.5
	all	3.96	1.57	12.2	9.75	
	disc	12.8	11.3	16.3	8.9	
MI-nonpara	nonocc	5.59	7.5	17.4	10.2	18
	all	7.54	8.99	25.7	19.9	
	disc	18.8	35	36.9	22.6	
本文算法	nonocc	7.71	2.24	11.8	7.34	13.6
	all	9.27	3.68	20.5	16.9	
	disc	20.9	21.0	25.9	15.6	

本算法对图像 Tsukuba、Venus、Teddy 和 Cones 的匹配结果如图 5-12 所示，其中， a)为应用 ADCensus 算法所得视差图、b)为应用 SSD+MF 算法所得视差

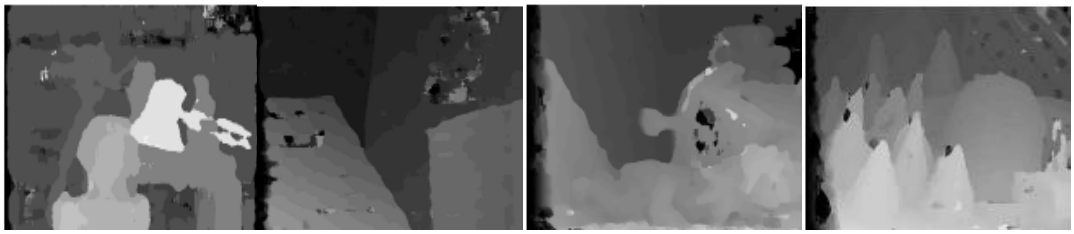
图、c)为应用 MI-nonpara 算法所得视差图、d)为应用本文算法所得视差图。



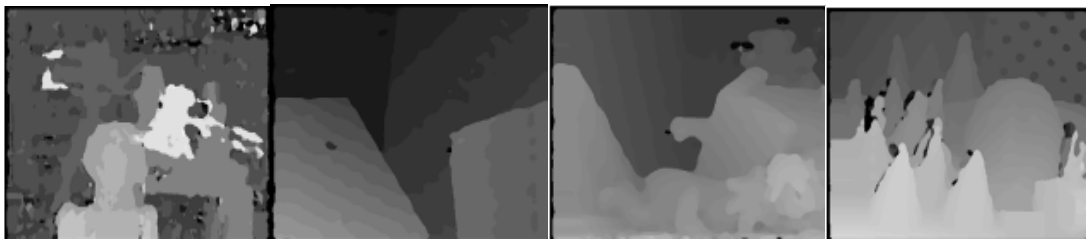
a)



b)



c)



d)

图 5-12 不同算法的视差图

Fig.5-12 Disparity maps of different algorithms

通过图 5-12 可以看出，本文算法的匹配效果在遮挡区域的处理方面优于 SSD+MF 算法，在视差不连续区域方面好于 MI-nonpara 算法。ADCensus 算法由于引入了颜色信息，匹配精度高于其他算法，但其复杂性远远高于本算法，不易于在实时系统上实现。因此，本文算法能够满足双目立体匹配系统的精度要求。

5.4 算法实时性分析

实时性评估以640*360大小的立体图像对作为目标图像。本系统将储存在DDR2中，然后处理该图像对10次，记录整体时间，取平均值，得到处理每一帧图像所需要的时间。同时，本文的算法将在PC上实现，PC机的环境为：Intel core i3处理器、2GB内存、Microsoft Windows XP SP2。两者处理速度如表5-2所示。

表5-2 本系统在PC和FPGA上处理速度表

Table 5-2 Processing speed of the system in the PC and FPGA

平台	图像分辨率	视差范围	每帧耗时	帧率
PC	640*360	60	7760ms	0.13fps
FPGA	640*360	60	32ms	30fps

表5-3列出了本文算法与其他文献里的立体匹配算法的实时性作比较，本文选择的图像大小为640*360视察搜索范围为60，FPGA工作频率为50MHz。本文算法在FPGA上实现的实时性优于大部分算法，与文献[55]的实时性持平，落后于文献[56]的实时性。然而，文献[55]在FPGA上的工作平率是148.5MHz，远远高于本文算法的工作频率。文献[56]采用的硬件是GPU，使用CUDA实现，其复杂计算上的能力远强于FPGA。

表5-3 实时性比较

Table 5-3 Comparison of real time

算法	平台	分辨率	视差范围	帧率(fps)
SSAD[46]	FPGA	320*240	25	20
Census+AW[54]	FPGA	358*288	60	26
SAD[55]	FPGA	480*270	30	60
Census[56]	GPU	450*375	60	105.4
本文算法	FPGA	640*360	60	30

FPGA的处理速度比PC机快是通过并行处理实现的。在PC机中，11*11的窗口，实现算法需要进行121次比较，而在FPGA上，这121个比较器可以并行工作，一次得到运算结果；在视差搜索范围内计算海明距离时，PC机上需要一个个计算，而在FPGA上可以并行工作，一次性计算出所有的海明距离，然

后通过 WTA 模块计算出视差，WTA 模块使用组合逻辑实现，可以在 1 个时钟周期内计算出视差。综上，本文算法的 FPGA 实现可以满足双目立体匹配系统的实时性要求。

5.5 本章小结

本章对改进的立体匹配算法进行实验和结果分析。首先，对各个模块进行了功能仿真，并且详细地介绍了每个模块的变换信号功能以及工作流程。然后，分析了变换窗口的尺寸对实验结果的影响。最后，从立体匹配精度和实时性两个方面进行实验并分析实验结果，将本文的实验结果与目前的研究成果做比较，从而验证了本文的算法在立体匹配精度和实时性两个方面的性能。

结论

本文的主要研究内容集中在以下几个方面：

1. 介绍了本课题的研究背景、目的和意义，分析比较当前流行的立体匹配算法的优缺点，阐述了各类算法硬件实现在实现立体匹配算法时的优缺点。本文确定使用基于 Mini-Census 变换的区域匹配算法，并通过 FPGA 硬件实现。

2. 匹配算法的改进。为了提高匹配的精度和算法的处理速度，本文采用 Mini-Census 变换并对其进行改进，计算邻域像素的灰度平均值，将邻域像素与中心像素和平均值分别比较，得到两位二进制的编码串。本文提出一种自适应的匹配窗口生成算法，结合 Sobel 梯度算子和中值滤波算法计算自适应阈值。然后对得到的左右一致性检测及投票算法相结合的方法进行精化。

3. 立体匹配算法的硬件实现。本文利用 Altera 公司的 DE2 FPGA 开发平台作为核心运算器件，采用模块化的实际方法，硬件实现了图像数据缓存、Mini-Census 变换和视差图后处理三个主要模块，及其包含的子模块。设计过程中，充分利用了 FPGA 可以并行处理的特点和流水线思想，有效地提高了立体匹配算法的实时性。同时分析仿真及实验结果。

本位还有一些不足，需要改进和提高，主要集中在以下几个方面：

1. 由于采用采用 Middlebury 网站提供的标准测试图片对本文算法进行评估，没有对现实环境中的真实图片进行测试，课题的后续需要基于现实情况对算法进一步完善。

2. 本文由于采用纯硬件设计，实现的算法比较简单，可以充分利用 Altera 公司 FPGA 的 NiosII 软核实现更复杂的算法。

3. 本文所提出双目立体匹配算法虽然在 640∗360 分辨率的图像能够满足实时性，但是对于某些分辨率要求较高的场合处理速度不够快。为了解决这一问题，可以采用多点并行和提高工作频率的方法，将处理速度成倍提高。

参考文献

[1] 白明，庄严.双目立体匹配算法的研究与发展[J].控制与决策，2008，23(7)：721-729.

[2] 陈小天，沈振康.机器人视觉导航[J].系统仿真学报，2008，20(10)：5501-5503.

[3] 陈艺峰.CMOS 摄像机标定实验研究[J].机电技术, 2011, 2(4)：25-28.

[4] Jonathan Gervais，Austin Younghlood，Walter H. Delashmit. Infrared-Based Object Tracking[J]. Proc. Of SPIE，2009，72(98)：1-9.

[5] 马颂德，张正友.计算机视觉——计算理论与算法基础[M].北京：科学出版社，2003：45-57.

[6] 田海锋.基于 FPGA 的双目立体视觉 SOPC 设计[D].西安：西安电子科技大学，2012：3-7.

[7] S.Mattoccia.Fast Locally Consistent Dense Stereo on Multicore[C].6th IEEE Embedded Computer Vision Workshop (ECVW2010)，2010： 69-76.

[8] Qingxiong Yang, Liang Wang, Ruigang Yang et al. Stereo Matching with Color-Weighted Correlation, Hierarchical Belief Propagation, and Occlusion Handling[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2009，31(3): 492-504.

[9] 葛宏升.3D 立体拍摄系统设计[D].北京：北京邮电大学，2013：20-28.

[10] 梁发云，丁鉴，张华等.自动多视点技术的眼暗跟踪方法[J].南昌大学学报(工科版)，2011，33(1)：79-82.

[11] JinZhou,BaoxinLi.Image Rectification for Stereoscopic Visualization[J].Journal of the Optical Society of America，2008，25(11)：2721-2733.

[12] 陈天飞，马孜.基于主动视觉标定线结构光传感器中的光平面[J].光学精密工程，2012，20(2)：256-263.

[13] 武斌，姬红兵，李鹏.人体三维无接触测量系统的研究[J].红外与毫米波学报，2006，25(5)：364-367.

[14] 施建盛.车载立体视觉技术的应用研究[D].南昌：南昌大学，2014:2-5.

[15] Yunseok Lee.Stereo Image Rectification Based on Polar Transformation[J].Optical Engineering，2008，47(8)：1-12.

[16] Hsien-Huang P Wu.Rectification of Stereo Scopic Video for Planar Catadioptric Stereo Systems[J].IEEE Transactions on Circuits and Systems for Video

Technoloy.2007，17(6)：686-698.

[17] Asif Mehmood，Nasser M. Nasrabadi. Kernel Wavelet-Reed-Xiaoli： an Anomaly Detection for Forward-looking Infrared Imagery[J]. Applied Optics 50，2011：2744-2751.

[18] 曹量，李太君.基于普通摄像机的双目立体视觉三维重构技术研究[J].通信技术，2011，44(9)：77-82.

[19] Wang Pei，Wu Fang，A Local Stereo Matching Algorithm Based on Region Growing[C]. Communications in Computer and Information Science，2012：459-464.

[20] 李晓琼，史彩成，毛二可.基简化的图像三维重建算法[J].光学技术，2008，34(5)：696-698.

[21] Kang Sunqil. Near-real-time Stereo Matching Method Using Temporal and Spatial Propagation of Reliable Disparity[J]. Optical Engineering，2014，5(6)：354-356.

[22] Xiao Jun. A Segment-based Stereo Matching Method with Ground Control Points [C]. 2010 2nd Conference on Environmental Science and Information Application Technology ，2010：306-309.

[23] Kuhn M, Moser S, Isler O et al. Efficient ASIC Implementation of A Real-Time DepthMapping Stereo Vision System[C].Proceedings of the IEEE 46th Midwest Symposium on Circuits and Systems，2003：1478~1481.

[24] Kim，Tae June. A Feature-based and Hierarchical Stereo Matching Method[J]. International Journal of Innovative Computing, Information and Control，2011：6785-6797.

[25] 侯洁，辛云宏.度数据重构三维物体方法[J].计量技术，2013，43(6)：683-688.

[26] 张双垒，林剑春，段东等.多个曲面拓扑模型及光滑重建方法的研究[J].红外技术，2012，34(8)： 472-481.

[27] Chang N，Ting-Min Lin，Tsung-Hsien Tsai. Real-Time DSP Implementation on Local Stereo Matching[C]. Proceedings of the 2007 IEEE International Conference on Multimedia and Expo，2007：2090-2093.

[28] Larab，Slimane.DSP Implementation for Stereo Matching[C].Proceedings of 2013 and Information Conference，2013：448-454.

[29] Vellanki，Pratibha.Enhanced Stereo Matching Technique Using Image Gradient for Improved Search Time[J].International Journal of Computer Science Issues，

2011，8(5)：483-486.
[30] 朱宾.基于 GPU 的的双目立体匹配技术研究[D].南京：南京航空航天大学，2010：23-25.
[31] 张永平.嵌入式双目视觉系统和三维重建技术研究[D].杭州：杭州电子科技大学，2011:8-10.
[32] Jinglin Zhang，Nezan J-F，Pelcat M et al. Real-Time GPU-Based Local Stereo Matching Method[C].Proceedings of the 2013 Conference on Design and Architectures for Signal and Image Processing，2013：209-214.
[33] 张翔.基于可编程片上系统的实时立体匹配算法研究[D].杭州：浙江大学，2014:23-35..
[34] Lu Zhang，Ke Zhang，Tian Sheuan Chang. Real-Time High Definition Stereo Matching on FPGA[C].Proceedings of the 19th ACM/SIGDA International Symposium on Field programmable gate arrays，2011：55-64.
[35] 梁伟涛.基于双目立体视觉三维重建方法的研究[D].哈尔滨：哈尔滨理工大学，2014:5-11.
[36] 龚文彪.立体匹配技术的研究及硬件化实现[D].南京：南京理工大学，2015：30-32.
[37] 王军政，朱华健，李静.一种基于 Census 变换的可变权值立体匹配算法[J].北京理工大学学报，2013，33(7)：704-710.
[38] 马利，李晶皎，马技.邻域相关信息的改进 Census 变换立体匹配算法[J].计算机工程与应用，2014，50(24)：16-20.
[39] 何欣荣，张刚，董建园.基于图像分割的改进立体匹配算法[J].微电子学与计算机 2014，23(12)：61-66.
[40] 王昭娜，赵西安，燕青浩.一种基于置信传播的全局立体匹配算法[J].北京建筑大学学报，2015，31(4)：58-63.
[41] 王莉，戴芳，郭文艳等.应用分形维数的自适应张量投票算法[J].计算机工程与应用，2013，49(12)：168-171.
[42] 王道累，吴懋亮，陈军.从双视图到多视图的协同优化立体视觉匹配算法[J].华侨大学学报，2015，36(3)：286-291.
[43] 刘金鑫，张祺.基于梯度值的自适应窗口立体图像匹配算法[J].计算机与现代化，2012，23(1)：67-69.
[44] 祝世平，李政.基于改进梯度和自适应窗口的立体匹配算法[J].光学学报，2015，24(1)：1-9.

[45] Chang N，Tsung-Hsien Tsai，Bo-Hsiung Hsu et al.Algorithm and Architectrue of Disparity Estimation with Mini-Census Adaptive Support Weight[J].IEEE Transactions on Circuits and Sysytem for Video Technology，2010，20(6)：792-805.

[46] Aysu A，Sayinta M，Cigla C.Low Cost FPGA Design and Implemetation of a Stereo Matching System for 3D-TV Applications[C].Proceedings of the 2013 IFIP/IEEE 21st International Conference on Very Large Scale Integration，2013：204-209.

[47] Jianbin Fang，Varbanescu A L， Jie Shen et al. Accelerating Cost Aggregation for Real-Time Stereo Matching[C].Proceedings of the 2012 IEEE International Conference on Parallel and Distributed Systems，2012：472-481.

[48] 宁静静，孔令德.一种基于双向互匹配的视差图算法[J].电脑开发与应用，2011，25(1)：17-20.

[49] 化春键，方程俊，陈莹.基于改进的 Census 变换的自由曲面立体匹配方法[J].计算机与现代化，2015,7(12)：43-47.

[50] 张丞，侯春萍，王晓燕等.立体图像视差自适应调整算法[J].光电子激光，2014，25(3)：581-587.

[51] 张超，王琼华，李大海.基于 SIFT 匹配算法的多视点自由立体显示视差图像的生成[J].光学学报，2010，30(7)：1989-1993.

[52] 靖固，任晓宇，纪颖.盲道识别系统算法设计及 FPGA 实现[J].哈尔滨理工大学学报，2014，19(6)：38-43.

[53] 葛威.改进型位级中值滤波硬件算法的实现[J]. 哈尔滨理工大学学报，2015，20(3)：35-39.

[54] Kristian Ambrosch， Wilfried Kubinger.Accurate Hardware-Based Stereo Vision[J].Computer Vision and Image Understanding， 2010， 114(11)：1303-1306.

[55] Yong Seok Heo, Kyoung Mu Lee, Sang Uk Lee. Robust Stereo Matching Using Adaptive Normalized Cross-Correlation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2011，33(4)：807-822.

[56] Ke Zhang, Jiangbo Lu, Lafruit G. Cross-Based Local Stereo Matching Using Orthogonal Integral Images[J]. IEEE Transactions on Circuits and Systems for VideoTechnology，2009，19(7)： 1073-1079.

攻读硕士学位期间发表的学术论文

[1] Jianhui Han，Zhen Wu，Lanying Li，Ying Ji.FPGA Implementation for Binocular Stereo Matching Algorithm Based on Sobel Operator[J]. International Journal of Database Theory and Application（已录用，2016 年 4 月刊出）

[2] 韩剑辉，吴振，李兰英.双目立体匹配算法的 FPGA 实现[J].哈尔滨理工大学学报（已录用，2016 年 4 月刊出）

致谢

时光飞逝，研究生学习生活马上就要结束，即将离开校园。此刻，我要对身边的老师、同学、朋友和家人表达由衷的谢意。

我衷心地感谢我的导师韩剑辉教授在研究生期间对我关心和指导。韩老师教学态度严谨、研究思想敏捷、态度平易近人，这都使我受益匪浅。课题的各个研究阶段在韩老师精心指导下展开，使我对所研究的课题有了深刻的认识。在实验过程中所遇到难题在韩老师的帮助下顺利解决。

感谢其他教授专业课的老师们，他们的授课给我的研究生课题研究及学位论文的撰写提供了扎实的理论基础。

我由衷地感谢寝室里的室友、实验室里的同学和寝室以及相识的朋友，感谢他们在我遇到学习或者是生活中的难题和困难时热心无私的帮助，使我愉快地度过近三年的研究生时光。

我要感谢我的家人，感谢他们对我无微不至的关怀与照顾，坚定了我求学的信念，使我能够全心全意地投入到学习中，祝愿他们健康快乐，我将努力回报他们。