



浙江工业大学

本科毕业设计论文

附件

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

目 录

- 一、文献综述
- 二、外文翻译
- 三、开题报告
- 四、指导教师评语
- 五、论文评阅人评语
- 六、答辩记录
- 七、毕业设计成果演示记录表
- 八、教师指导记录表
- 九、毕业设计进程考核表
- 十、毕业论文的“知网”检测结果

一、文献综述

双目视觉立体匹配算法设计与实现

姓名 王灏 专业班级 通信工程 1301 .

摘要：本文回顾了立体视觉研究的发展和现状，简述了立体视觉技术在工业军事等领域的广泛应用，并介绍了双目立体视觉技术的研究背景和基本原理。随后，本文回顾了卷积神经网络（Convolution Neural Network, CNN）的发展历史，并简单描述了一个卷积神经网络的基本结构。此外本文还回顾了立体匹配算法的发展现状，以及卷积神经网络在立体匹配中的应用。最后，本文介绍了当前主流的立体视觉数据集 KITTI 和 Middlebury 及其对应的评测库。

关键词：卷积神经网络、立体视觉、立体匹配

1 引言

1.1 立体视觉研究的发展历史

众所周知，自然界中的物体都是三维立体的，人类通过双眼可以获取物体的三维立体信息。但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。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 等创立的视觉计算理论对立体视觉的发展产生了巨大影响，现已形成了从图像获取到最终的景物可视表面重建的比较完整的体系^[1-2]。

1.2 双目立体视觉的研究背景和基本原理

立体视觉技术在军事、工业的各个方面都有着非常广泛的应用背景，具有很大的商业价值和使用价值。在军事领域中，众所周知，现代战争是信息化战争，对信息的搜

索和利用率往往决定了一场战役的胜负。利用侦察卫星，单兵侦察等手段获取二维图像数据，并通过立体视觉算法进行三维重构，模拟出三维的战场环境，可以为作战指挥提供信息支持。在机器人导航系统中立体视觉技术可以用于环境检测、障碍识别。在工业检测系统中立体视觉技术常用于产品质量检测。在无人驾驶技术中立体视觉技术还可用于道路环境的检测。

目前对立体视觉的研究主要有三类方法：第一类方法是直接利用测距器（如激光雷达，结构光等）获得深度信息，重建三维模型的方法。这类方法通常需要特殊仪器，成本较高，而且由于设备复杂，应用环境受限；第二类方法是根据光学成像原理及统计假设，仅利用一幅图像所提供的信息推断三维模型的方法。这种方法由于受到单一图像所能提供信息的局限性，难以提供准确的深度信息；第三类方法是利用不同视点上的，是不同时间或不同位置拍摄的两幅或多幅二维图像重构三维模型的方法。第三类方法是目前较为常见的立体视觉方法。双目立体视觉就是第三类方法的常见实现方式。

双目立体视觉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，或由单摄像机在不同时刻从不同角度获取目标场景的两幅数字图像，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。

双目立体视觉技术实现的一般步骤为：相机内参外参的离线标定，双目相机图像矫正，立体匹配以及光学三角形计算深度信息。通常采用不同水平位置的相机拍摄的两个图像，通过立体匹配找出对应点，计算视差 d ，视差指的是同一对象在左图像和右图像中的水平位置的差异——同一对象在左图像中的位置为 (x, y) ，在右图像中的位置为 $(x-d, y)$ 。已知视差可以根据以下公式

$$z = \frac{fB}{d}$$

计算它的深度 z ，其中 f 是相机的焦距， B 是相机中心之间的距离。上述步骤中的双目立体匹配算法是双目视觉技术的核心问题。立体匹配的精度和速度对于立体视觉系统有着很大的影响^[3-4]。

目前立体匹配面临许多挑战性的问题：遮挡匹配问题、弱纹理或重复匹配问题、深度不连续匹配问题、光照变化引起匹配问题等。近几年来随着深度学习（Deep Learning）技术的发展，其在图片、语音方面强大的特征提取表现使得利用 CNN 进行立体视觉匹配变成可能。

2 卷积神经网络

2.1 卷积神经网络的发展历史

1962 年 Hubel 和 Wiesel 在研究猫脑皮层中用于局部敏感和方向选择的神经元时发现其独特的网络结构可以有效地降低反馈神经网络的复杂性，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^[5]。后来研究者便将这种逐步抽象的分层模型命名为深度学习模型。1984 年日本学者 Fukushima 基于上述提出的感受野概念，构建了神经认知机(Neocognitron)，神经认知机是卷积神经网络的第一个实现网络，其将视觉模式分为多个子模式（特征），然后进入分层连接的特征平面处理^[6]。随后，更多的科研工作者对该网络进行了改进。1988 年 LeCun 等人将 BP 神经网络算法引入 CNN，LeCun 等人结合 BP 算法实现的 LeNet-5 模型在数字识别领域的表现强大，在银行支票的手写体字符识别中，识别正确率达到商用级别^[7]。这是第一个真正多层结构的学习算法，它利用空间相对关系减少参数数目以提高训练性能。2006 年，Hinton 提出了深度置信网络（DBN），一种深层网络模型。使用一种贪心无监督训练方法来解决训练问题并取得良好结果^[8]。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^[8]。

2.2 卷积神经网络的网络结构

如图 1 所示，卷积神经网络是一个多层的神经网络，每层由多个二维平面组成，每个平面由多个独立神经元组成。

C 层为特征提取层（卷积层），每个神经元的输入与前一层的局部感受野相连，并提取该局部的特征，一旦该局部特征被提取后，它与其它特征间的位置关系也随之确定下来；S 层是特征映射层（池化层），网络的每个计算层由多个特征映射组成，每个特征映射为一个平面，平面上所有神经元的权值相等。特征映射结构采用影响函数核小的 sigmoid 函数作为卷积网络的激活函数，使得特征映射具有位移不变性。

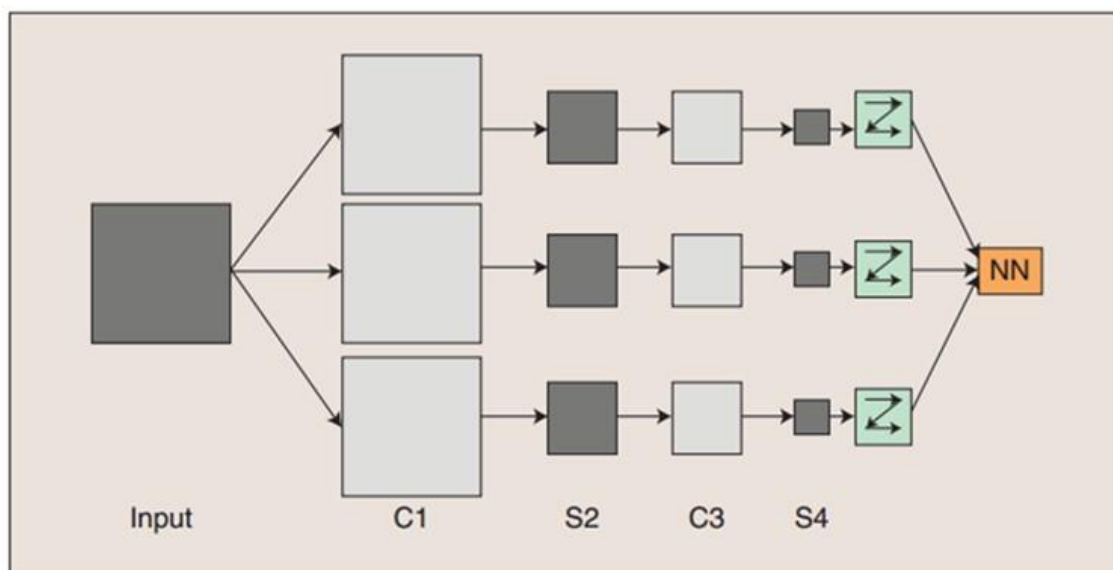


图 1 卷积神经网络的概念示范

此外，由于一个映射面上的神经元共享权值，因而减少了网络自由参数的个数，降低了网络参数选择的复杂度，所以卷积神经网络是一种权值共享网络，网络参数的减少使得能构建更加深（层数）以及更加宽（特征树）的网络结构。卷积神经网络中的每一个特征提取层（C-层）都紧跟着一个用来求局部平均与二次提取的计算层（S-层），这种特有的两次特征提取结构使网络在识别时对输入样本有较高的畸变容忍能力。

3 立体匹配研究及发展现状

3.1 立体匹配算法

3.1.1 区域匹配法

区域匹配算法是以参考图像中某一待匹配像素点为中心，选择一个矩形窗口作为其约束区域，然后，从目标图像中，寻找与其匹配的像素点，同样也以寻找的像素点为中心，选择同样大小一个矩形窗口，用此区域内像素点的属性信息约束该像素点的相似性计算。在左右图像中，两个像素之间的相似性必须满足一定相似性条件，才认为它们是相似的，区域匹配算法的目的是获取稠密视差图。区域匹配算法的缺点在于受图像的仿射畸变和辐射畸变的影响较大，而且像素点约束窗口的大小与形状选择比较困难，选择过大，在深度不连续处，视差图中会出现过度平滑现象；选择过小，对像素点的约束比较少，图像信息没有得到充分利用，容易产生误匹配。

随着研究的深入，开始出现各种改进的算法，如自适应窗体算法，自适应权重算法

等。目前较先进的 PMSC (PatchMatch-based Superpixel Cut) 算法采用双层匹配代价计算, 首先匹配小正方形的窗口然后在更大的不规则窗口上进行代价聚合, 最后进行全局优化。该算法截至 2016 年 12 月在 Middlebury 立体视觉测评中排名第一。^[9]

3.1.2 特征匹配算法

特征的匹配算法, 主要是基于几何特征信息 (边缘、线、轮廓、兴趣点、角点和几何基元等), 针对几何特征点进行视差估计, 所以先要提取图像的特征点, 进而利用这些特征点的视差值信息来重建三维空间场景。目前较流行的 SIFT (Scale Invariant Feature Transform) 就是一种提取局部特征的算法, 在尺度空间中寻找极值点, 提取位置、尺度、旋转不变量, 生成关键点特征描述符, 然后根据这些不变量特征进行匹配^[10]。SIFT 特征是基于物体上的一些局部外观的兴趣点而与影像的大小和旋转无关^[10]。对于光线、噪声、些微视角改变的容忍度也相当高。基于这些特性, 它们是高度显著而且相对容易提取, 在母数庞大的特征数据库中, 很容易辨识物体而且鲜有误认^[10]。使用 SIFT 特征描述对于部分物体遮蔽的侦测率也相当高, 甚至只需要 3 个以上的 SIFT 物体特征就足以计算出位置与方位^[10]。现今的电脑硬件速度下和小型的特征数据库条件下, 辨识速度可接近即时运算^[10]。但是基于特征的匹配算法, 得到的匹配点对数量较少, 无法满足生成视差图的要求, 所以使得它的应用领域受到了限制^[10]。

3.1.3 CNN 立体匹配

2015 年, Zbontar 和 LeCun 在 CVPR 上提出使用卷积神经网络来计算立体匹配代价, 并在 2016 年进一步完善。CNN 立体匹配是一种局部匹配算法, 主要是针对立体匹配算法的第一阶段: 匹配代价的计算。CNN 立体匹配使用相似和不相似的图像对构建二元分类数据集对卷积神经网络进行有监督的训练, 然后通过卷积神经网络进行小图像块的相似性度量。卷积神经网络的输出用于初始化立体匹配代价, 最后经过一系列的后处理步骤来进行代价聚合, 视差计算以及视差精化^[3-4]。

目前在立体视觉评测库 Middlebury 和 KITTI 中, 基于卷积神经网络的立体视觉匹配算法仍然排在前列。

3.2 立体视觉数据集

立体视觉真实标签数据对于卷积神经网络的构建非常重要, 优质的标签数据可以获得更好的训练模型。目前主流的立体视觉数据集是 KITTI 和 Middlebury 数据集。

3.2.1 KITTI 数据集

KITTI 立体数据集^[11-12]是从安装在汽车车顶上两个相隔约 54 厘米的摄像机拍摄的经整流的灰度图像对的集合。图像是在白天晴朗和多云的天气下，在 Karlsruhe 市中心和郊区开车记录的。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。

测试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，例如，错误的容许范围为 3 厘米则转换成物理距离为距离相机 2 米，而 80 厘米的容许范围，就是距离相机 10 米。

目前存在两个 KITTI 立体视觉数据集：KITTI 2012 和较新的 KITTI 2015。为了完成立体视觉的计算任务，他们几乎是相同的，但较新的数据集的改进了光流任务的一些方面。2012 年的数据集包含 194 个训练图和 195 个测试图像，而 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区别：运动车辆是密集标记的，并且汽车玻璃也被包括在评估中。

3.2.2 Middlebury 数据集

Middlebury 立体视觉数据集^[13-16]的图像对来自光照可控的室内场景。数据通过结构光来测量真实视差，视差的密度和精度都比在 KITTI 数据集好。该数据集共发布了五个独立的数据集，分别是在 2001 年，2003 年，2005 年，2006 年和 2014 年。

在 2005 年，2006 年，和 2014 年数据集的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，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。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。测试数据集包含 15 幅来自 2005 年和 2014 年数据集的图像。数据集提供全分辨率，半分辨率和四分之一分辨率的图片。误差率的计算是根据全分辨率的，如果算法输出的是半分辨率或四分之一分辨率的视差图，它们在计算错误率之前上采样。

4 总结

双目立体视觉经过几十年的研究已经取得了显著的成果，出现了各种专门的硬件设计和视频速率实时的立体视觉系统，目前立体视觉技术被广泛应用于军事侦察，无人驾

驶，机器人导航以及工业自动化系统中。但是，从普遍的意义来讲，由于难以彻底解决立体匹配的对应点问题，具体的立体视觉系统一般都是有针对性的、不普遍适用的，还无法与人类的双目视觉系统相媲美。

但近年来，随着深度学习技术在计算机视觉处理领域的快速发展，使用深度学习算法建立一个普遍适用的立体匹配网络也成为重要的趋势。

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] Zbontar J, Lecun Y. Computing the stereo matching cost with a convolutional neural network[C]. computer vision and pattern recognition, 2015: 1592-1599.
- [4] Žbontar J, Lecun Y. Stereo matching by training a convolutional neural network to compare image patches[J]. Journal of Machine Learning Research, 2016, 17(65): 1-32.
- [5] 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.
- [6] 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.
- [7] 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.
- [8] 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.
- [9] Li L, Zhang S, Yu X, et al. PMSC: PatchMatch-Based Superpixel Cut for Accurate Stereo Matching[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2016.
- [10] Lowe D G. Distinctive Image Features from Scale-Invariant Keypoints[J]. International Journal of Computer Vision, 2004, 60(2): 91-110.
- [11] 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.
- [12] Guney F, Geiger A. Displets: Resolving stereo ambiguities using object knowledge[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 4165-4175.
- [13] 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.

- [14]Scharstein D, Szeliski R, Zabih R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms[C]//Stereo and Multi-Baseline Vision, 2001.(SMBV 2001). Proceedings. IEEE Workshop on. IEEE, 2001: 131-140.
- [15]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.
- [16]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.

二、外文翻译

基于计算机视觉的自动植物物种识别系统——Leafsnap

Jure Zbontar, 卢布尔雅那大学计算机与信息科学学院

Yann LeCun, 纽约大学 Courant 数学科学研究所

翻译人 王灏 专业班级 通信工程 1301

摘要:我们提出了一种从经过矫正的图像对中提取深度信息的方法。我们的方法集中在许多立体视觉算法的第一阶段：匹配代价计算。我们通过使用卷积神经网络进行小图像块的相似性度量来解决这个问题。我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。针对这个任务，我们检测了两个网络架构：一个是速度导向的，另一个是准确性导向的。卷积神经网络的输出用于初始化立体匹配代价。一系列后处理步骤如下：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。我们用 KITTI 2012, KITTI 2015 和 Middlebury 的立体视觉数据集来评估我们的方法，并发现它在所有的三大数据集中的表现均优于其他方法。

关键词: 立体视觉，匹配代价，相似性学习，监督学习，卷积神经网络

1. 介绍

考虑以下问题：给定由不同水平位置的相机拍摄的两个图像，我们希望计算左图像中每个像素的视差 d 。视差指的是同一对象在左图像和右图像中的水平位置的差异——同一对象在左图像中的位置为 (x, y) ，在右图像中的位置为 $(x-d, y)$ 。如果我们知道一个对象的视差，我们可以使用以下关系计算它的深度 z ：

$$z = \frac{fB}{d}$$

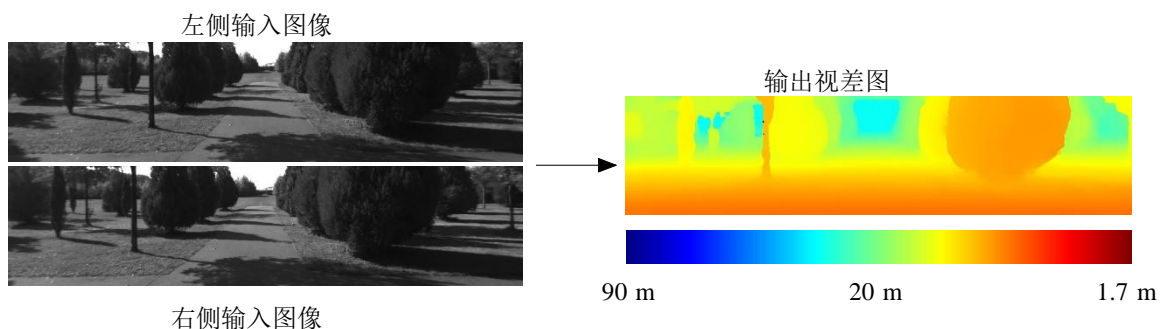


图 1: 输入是来自左侧和右侧相机的一对图像。两个输入图像的差异主要在对象的水平位置上(其他的差异是由反射, 遮挡和透视失真引起的)。注意靠近相机的对象比远离相机的对象有更大的视差。右图的输出是一个密集的视差图, 暖色表示更大的视差值(和较小的深度值)

其中 f 是相机的焦距, B 是相机中心之间的距离。图 1 描述了输入和通过我们的方法处理的输出。

上述的立体匹配问题在许多领域中都是非常重要的, 例如自动驾驶, 机器人技术, 中间视图生成和 3D 场景重建。根据 Scharstein 和 Szeliski (2002) 的分类, 一个典型的视觉算法包括四个步骤: 匹配代价计算, 代价聚合, 优化, 视差精化。根据 Hirschmüller 和 Scharstein (2009) 的分类, 我们认为前两个步骤是计算匹配代价和后两个步骤为立体视觉法。我们工作的重点是计算好匹配代价。

我们建议对图像块对训练卷积神经网络 (LeCun 等人, 1998 年), 其中真实的视差是已知的 (例如, 由激光雷达或结构光获得)。网络的输出用于初始化匹配代价。我们还需进行许多不是新颖的但是必要的后处理步骤以取得良好效果。匹配代价是由具有相似图像强度的邻近像素通过基于交叉的代价聚合的方式组合而成。平滑约束通过半全局匹配进行, 同时左右一致性检查被用来检测和消除遮挡区域中的误差。我们进行亚像素增强并应用中值滤波器和双边滤波器以获得最终视差图。

本文的贡献是:

- 基于卷积神经网络的为计算立体匹配代价的两种架构的描述;
- 一种方法, 伴随其源代码, 在 KITTI 2012, KITTI 2015 和 Middlebury 立体视觉数据集中具有最低的错误率;
- 实验分析了数据集大小的重要性, 与其他方法相比的错误率, 以及不同超参数设置下精度和运行时间之间的权衡。

本文延伸了我们以前的工作 (Zbontar 和 LeCun, 2015 年), 包括一个新架构的描述, 两个新数据集的结果, 更低的错误率以及更彻底的实验。

2. 相关工作

在引入大型立体数据集如 KITTI 和 Middlebury 之前, 相对较少立体视觉算法使用标签信息来得到他们模型的参数; 在这一节, 我们回顾一下一些以前的做法。有关立体视觉算法的一般概述, 请参阅 Scharstein 和 Szeliski (2002)。

Kong 和 Tao (2004) 采用平方距离的总和来计算初始匹配代价。然后他们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。Kong 和 Tao (2006) 随后延伸了他们的工作, 通过组合由计算归一化的不同的窗口大小和中心的互相关获得的预测。Peris 等人 (2012) 用 AD-Census (Mei 等, 2011) 进行初始化匹配代价, 并使用多类线性判别分析来得知从计算的匹配代价到最终视差的映射。

标签数据还用于得到概率图形模型的参数。Zhang 和 Seitz (2007) 使用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。Scharstein 和 Pal (2007) 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。Li 和 Huttenlocher (2008) 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。

最近的工作 (Haeusler 等人, 2013; Spyropoulos 等人, 2014) 集中在估计计算的匹配代价的置信度。Haeusler 等人 (2013) 使用了一种随机森林分类器来组合若干置信度量方式。同样的, Spyropoulos 等人 (2014) 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立体视觉法的误差。

一个计算匹配代价的相关问题是得到局部图像描述符 (Brown 等人, 2011; Trzcinski 等人, 2012; Simonyan 等人, 2014; Revaud 等人, 2015; Paulin 等人, 2015; Han 等人, 2015; Zagoruyko 和 Komodakis, 2015)。这两个问题共享一个公共子任务: 测量图像块之间的相似性。Brown 等人 (2011) 提出了一个总体框架来得到图像描述符并使用鲍威尔的方法来选择良好的超参数。为得到局部图像描述符已经提出了几种解决方法, 例如 Boosting 优化 (Trzcinski 等人, 2012), 凸优化 (Simonyan 等人, 2014 年), 分层移动象限的相似性 (Revaud 等人, 2015), 卷积内核网络 (Paulin 等人, 2015), 和卷积神经网络 (Zagoruyko 和 Komodakis, 2015; Han 等人, 2015)。Zagoruyko 和 Komodakis (2015) 的工作, Han 等人 (2015), 特别是, 非常类似于我们自己, 不同的主要是网

络的架构;具体地，包括合并和子采样以考虑更大的块尺寸和更大的视点变化。

3. 匹配代价

典型的立体视觉算法首先计算在每个位置 \mathbf{p} 考虑所有视差 \mathbf{d} 的匹配代价。一种计算匹配代价的简单方法是绝对差的总和：

$$C_{SAD}(\mathbf{p}, \mathbf{d}) = \sum_{\mathbf{q} \in N_{\mathbf{p}}} |I^L(\mathbf{q}) - I^R(\mathbf{q} - \mathbf{d})| \quad (1)$$

其中， $I^L(\mathbf{q})$ 和 $I^R(\mathbf{p})$ 是位置 \mathbf{p} 在左右图像中的图像强度， $N_{\mathbf{p}}$ 是以 \mathbf{p} 为中心的固定的矩形窗口内的位置集合。

我们使用粗体小写字母 \mathbf{p} 和 \mathbf{q} 表示图像位置。粗体小写 \mathbf{d} 表示向量的视差 \mathbf{d} ，即 $\mathbf{d} = (d, 0)$ 。我们为超参数的名称使用 typewriter 字体。例如，我们将使用 patch size 字体来表示附近区 $N_{\mathbf{p}}$ 的大小

等式（1）可以理解为测量匹配左图以位置 \mathbf{p} 为中心的图像块与右图以位置 $\mathbf{p}-\mathbf{d}$ 为中心的图像块的代价。

我们希望当两个图像块围绕相同的 3D 点时代价小，反之则代价大。

既然好的和坏的匹配的示例可以从公开可用数据集构建(例如, KITTI 和 Middlebury 立体数据集)，我们可以尝试通过监督学习方法来解决匹配问题。受到卷积神经网络在视觉问题中成功应用的启发，我们用它来评估两个小图像块匹配程度。

3.1 构造数据集

我们使用来自 KITTI 或 Middlebury 立体视觉数据集的地面实况视差图来构建二元分类数据集。在每个真实视差已知的图像位置处，我们提取一个消极的和一個积极的训练示例。这确保了数据集包含相等数量的正例和负例。一个积极的示例是指一对图像块，其中一个来自左边图像，一个来自右边图像，两者的中心像素是同一个 3D 点，而一个消极的示例是指一对不同于前者的图像块。以下部分详细描述数据集构建步骤。

用 $\langle P_{n \times n}^L(\mathbf{p}), P_{n \times n}^R(\mathbf{q}) \rangle$ 表示一对图像块，其中 $P_{n \times n}^L(\mathbf{p})$ 为左图中以位置 $\mathbf{p} = (x, y)$ 为中心的 $n \times n$ 图像块， $P_{n \times n}^R(\mathbf{q})$ 是右图中以位置 \mathbf{q} 为中心的 $n \times n$ 图像块， \mathbf{d} 表示位置 \mathbf{p} 处的正确视差。一个消极的示例的获取是通过将右图像块的中心设置为

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

其中 o_{neg} 是从区间 $[\text{dataset_neg_low}, \text{dataset_neg_high}]$ 或它的原点对应区间 $[-\text{dataset_neg_high}, -\text{dataset_neg_low}]$ 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。

一个积极示例的获取则是通过设置

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

其中 o_{pos} 是从区间 $[-\text{dataset_pos}, \text{dataset_pos}]$ 中随机挑选的。上式中包含 o_{pos} 而不是将其设为 0 的原因与后续使用的立体视觉法有关。特别是，我们发现当网络将低匹配代价分配给良好匹配和相近的匹配时基于交叉的代价聚合表现更好。在我们的实验中，超参数 dataset_pos 永远不会大于一个像素。

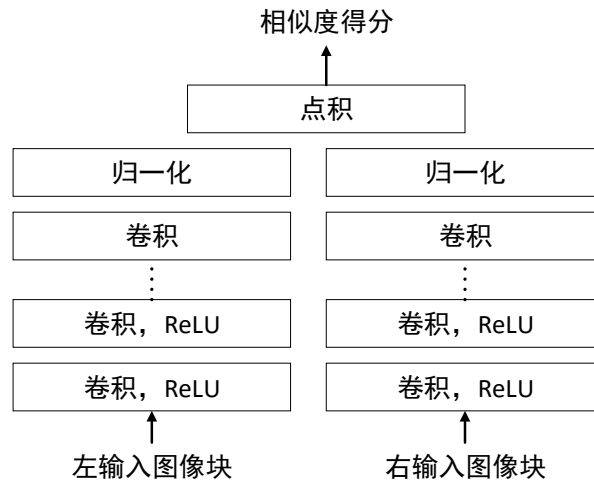


图 2: 这个快速架构是一个孪生网络。这两个子网络由多个卷积层和紧随其后的整流线性单元（缩写为“ReLU”）组成。通过从每次输入的两个图像块中提取向量并计算它们之间的余弦相似性来获得相似度得分。在上图和我们的具体实现中，余弦相似性计算可分为两个步骤：归一化和点积。因为每个位置仅需要执行一次归一化操作，所以这种方法减少了运行时间（详见 3.3 节）

3.2 网络架构

为了比较图像块上的相似度，我们提出了两种网络架构。第一种网络架构比第二种运行速度更快，但产生的视差图不太准确。在这两种架构下，网络的输入是一对小图像

块，输出是它们之间的相似性的度量。两个架构都包含一个可训练的用特征向量表示每个图像块的特征提取器。图像块之间的相似性通过特征向量上而不是原始图像强度值衡量。快速架构使用固定的相似性度量来比较两个特征向量，而精确架构试图从特征向量中得到一个更好的相似性度量。

3.2.1 快速架构

第一种架构是孪罗网络，即两个共享权重子网加在头部（Bromley 等人，1993）。子网络由多个带整流线性单元（除最后一层外）的卷积层组成。两个子网输出一个带有输入图像块属性的向量。得到的两个向量比较使用余弦相似性度量来产生网络的最终输出的。图 2 提供了该架构的概述。

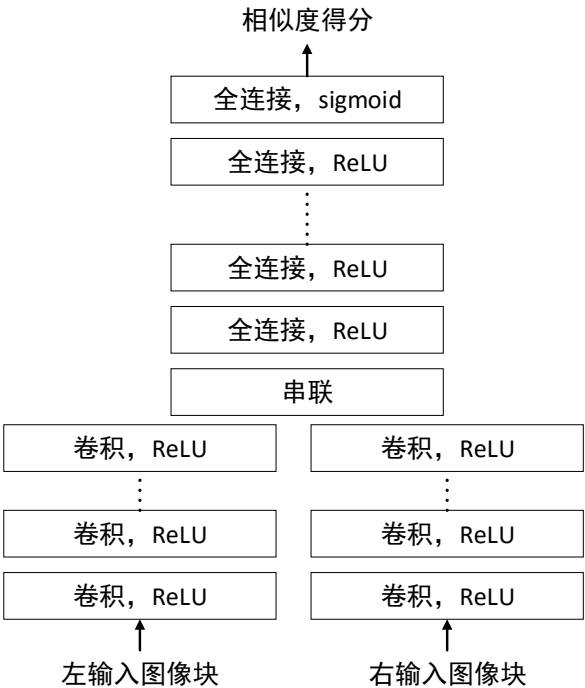


图 3：精确架构从两个卷积特征提取器开始。提取的特征向量串联并通过多个全连接层比较。输入是两个图像块，输出是介于 0 和 1 之间的单个实数，我们称这个实数为输入图像相似性的度量。

通过使铰链损耗最小化来训练网络。损失是通过以同一图像位置为中心的含积极和消极两类的示例对来计算的。设 s_+ 是积极实例网络的输出， s_- 是消极实例网络的输出，并让余量 m 为正实数。该实例对的铰链损失定义为 $\max(0, m + s_- - s_+)$ 。当积极实例的相似性大于消极实例至少余量 m 时，损失为零。在我们的实验中，我们设置余量为

0.2。

该架构的超参数是每个子网中卷积层的数量 (`num_conv_layers`) ,卷积核的大小 (`conv_kernel_size`) ,每层中的特征映射的数量 (`num_conv_feature_maps`) ,以及输入图像块的大小 (`input_patch_size`)。

3.2.2 精确架构

第二种架构是从第一种架构中通过用一些全连接层替换余弦相似性度量得到的(见图3)。

这种架构的变化增加了运行时间,但降低了错误率。两个子网包括很多卷积层,在每层之后紧跟着整流线性单元。两个结果向量串联并通过多个紧跟整流线性单元全连接层前向传播。最后一个全连接层经过 `sigmoid` 非线性变换输出一个数字,这个数字就是两图像块的相似度得分。

我们使用二元交叉熵损失函数来进行训练。令 s 表示一个训练示例的网络输出, t 表示该训练示例的类别;如果示例属于积极示例的类别,则 $t = 1$, 如果示例属于消极示例的类别,则 $t = 0$ 。该示例的二元交叉熵损失被定义为 $t \log(s) + (1 - t) \log(1 - s)$ 。

两个架构使用不同的损失函数是基于实验性的证据。虽然我们宁愿对两个架构使用相同的损失函数,但实验表明在精确架构中,二元交叉熵损失函数表现得比链损耗函数更好。另一方面,虽然快速架构的最后一步是余弦相似度计算,但交叉熵损失不直接适用。

精确架构的超参数是在每个子网络中卷积层的数量 (`num_conv_layers`) ,每个层中的特征映射的数量 (`num_conv_feature_maps`) ,卷积内核的大小 (`conv_kernel_size`) ,输入图像块的大小 (`input_patch_size`) ,每个全连接层中的单元数 (`num_fc_units`) ,以及全连接层数 (`num_fc_layers`)。

3.3 计算匹配代价

网络的输出用于初始化匹配代价:

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

其中, $s(< P^L(\mathbf{p}), P^R(\mathbf{p} - d) >)$ 是当网络输入图像块 $P^L(\mathbf{p})$ 和 $P^R(\mathbf{p} - d)$ 时的输出。

负号将相似度分数转换为匹配代价。要计算整个匹配的成本代价张量 $C_{CNN}(\mathbf{p}, d)$ ，我们仅需要对每个图像位置和每个需考虑的视差执行前向传递。以下三个实现细节保证了运行时间可管理：

- 两个子网络的输出每个位置只需计算一次，并且不需要针对每个需考虑的差异重新计算。
- 通过传播全分辨率图像，可以为单次前向传递中的所有像素点计算两个子网络的输出，而不仅仅是一个小图像块。在整个 $w \times h$ 图像上执行单次前向传递比执行 $w \times h$ 次小图像块的前向传递更快，因为许多中间结果可以重复使用。
- 全连接层在精确架构中的输出也可以在单次前向传递中计算。这是通过用一个 1×1 内核的卷积层替换每个全连接层实现的。我们还需要执行每个需考虑的视差的前向传递；对于 KITTI 数据集，最大视差 d 为 228，对于 Middlebury 数据集，最大视差为 400。因此，网络的全连接部分需要运行 d 次，这是精确架构的瓶颈。

为计算一对图像的匹配代价，我们每张图片在子网上运行 1 次，在全连接层运行 d 次，其中 d 是需考虑的最大视差。这个理解在设计网络的架构中是重要的。我们可能会选择两个图像在提交给网络之前级联的架构，但这意味着在运行时的大量代价，因为整个网络将需要运行 d 次。这种理解也导致了快速架构的发展，快速架构中唯一需要运行 d 次的层是特征向量的点积。

4. 立体视觉法

卷积神经网络的原始输出不足以产生准确的视差图，特别是在低纹理区域和遮蔽区域存在明显的误差。通过应用一系列后处理来改善视差图的质量，这就是我们称之为立体视觉法的步骤。我们所使用的立体视觉方法是受到 Mei 等（2011）的论文的影响，其中包括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，中值和双边滤波器。

4.1 基于交叉的代价聚合

来自相邻像素的信息可以通过对一个固定窗口内的匹配代价进行平均来组合。但这

种方法在深度不连续点附近失效，在那些点违反了窗口内深度恒定的设定。我们可能更喜欢自适应为每个像素选择邻域的方法，那样信息仅从相同的物理对象的像素中获取。在基于交叉的代价聚合（Zhang 等人，2009）中，我们在每个位置周围建立了一个本地邻域，邻域中包括类似图像强度值的像素，我们希望这些像素属于同一个对象。

该方法开始于在每个位置构建直立十字；用这个十字来定义本地支持区域。只要以下两个条件成立，左臂 P_l 就在位置 \mathbf{p} 向左延伸：

- $|I(\mathbf{p}) - I(P_l)| < \text{cbca_intensity}$; 位置 \mathbf{p} 和 P_l 的图像强度应该是类似，它们的差异应小于 cbca_intensity 。
- 对 $\|\mathbf{p} - P_l\| < \text{cbca_distance}$; 位置 \mathbf{p} 和 P_l 的水平距离（在上臂或下臂时为垂直距离）小于 cbca_distance 像素。

右臂，下臂和上臂的构建都是与前者类似的。一旦四个臂确定的，我们就可以计算支持区域 $U(\mathbf{p})$ 为铺设在位置 \mathbf{p} 垂直臂上所有的位置 \mathbf{q} 的水平臂的并集（见图 4）。

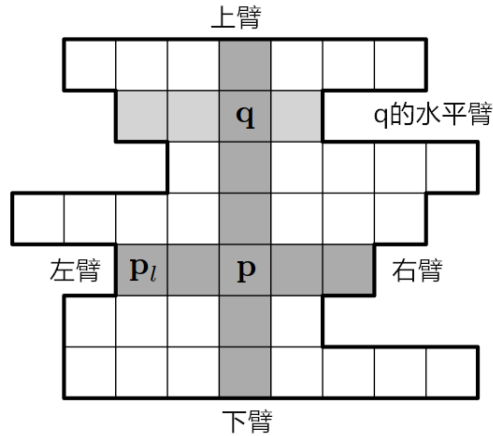


图 4: \mathbf{p} 的支持区域是铺设在位置 \mathbf{p} 垂直臂上所有的位置 \mathbf{q} 的水平臂的并集

Zhang 等人（2009）认为，聚合应考虑立体对的两张图片的支持区域。令 U^L 型和 U^R 表示在左，右图片的支持区域。我们定义共同支持区域 U_d 为：

$$U_d(\mathbf{p}) = \{\mathbf{q} | \mathbf{q} \in U^L(\mathbf{p}), \mathbf{q} - \mathbf{d} \in U^R(\mathbf{p} - \mathbf{d})\},$$

在共同支持区域内对匹配成本进行平均：

$$C_{CBCA}^0(\mathbf{p}, \mathbf{d}) = C_{CNN}(\mathbf{p}, \mathbf{d}),$$

$$C_{CBCA}^i(\mathbf{p}, \mathbf{d}) = \frac{1}{|U_d(\mathbf{p})|} \sum_{\mathbf{q} \in U_d(\mathbf{p})} C_{CBCA}^{i-1}(\mathbf{q}, \mathbf{d}),$$

其中 i 是迭代数。我们重复平均次数。因为支持区域是重叠的，每次迭代计算结

果都可能改变。我们在快速架构中跳过基于交叉的代价聚合，因为它对于实现低错误率没有决定性作用而且计算开销相对较大。

4.2 半全局匹配

我们通过对视差图像实施平滑约束来改善匹配代价。根据 Hirschmüller (2008) 所说，我们定义一个依赖于视差图 D 的能量函数 $E(D)$ 的：

$$E(D) = \sum_p (C_{BCA}^4(p, D(p)) + \sum_{q \in N_p} P_1 \cdot 1\{|D(p) - D(q)| = 1\} \\ + \sum_{q \in N_p} P_2 \cdot 1\{|D(p) - D(q)| > 1\}),$$

其中 $1\{\cdot\}$ 表示指标函数。第一项惩罚了高匹配代价的视差。当相邻像素的出现一个视差不同点时，第二项增加一个罚因子 P_1 。当相邻像素的视差不同点超过一个时，第三项增加了一个更大的罚因子 P_2 。

我们可以通过动态规划在单一方向上实现 $E(D)$ 最小化，而不是同时所有方向上最小化。因为在我们没有优化的方向上没有激励来使视差图平滑，所以这个解决方案会引入不必要的拖尾效应。在半全局匹配中，我们最小化单个方向的能量，并重复几个方向，最后取平均来得到最终结果。虽然 Hirschmüller (2008) 建议选择 16 个方向，但我们只沿两个水平方向和两个垂直方向优化；添加对角线方向并没有提高我们系统的准确性。为了最小化方向 r 上的 $E(D)$ ，我们定义了一个具有以下递推关系的匹配代价 $C_r(p, d)$ ：

$$C_r(p, d) = C_{BCA}^4(p, d) - \min_k C_r(p - r, k) + \min\{C_r(p - r, d), C_r(p - r, d - 1) + \\ P_1, C_r(p - r, d + 1) + P_1, \min_k C_r(p - r, k) + P_2\},$$

第二项是减法是为了防止 $C_r(p, d)$ 的值过大以致于影响优化视差图。

惩罚因子 P_1 和 P_2 是根据图像梯度为使视差中的跳跃与图像的边缘一致而设置的。

令 $D_1 = |I^L(p) - I^L(p - r)|$ 和 $D_2 = |I^R(p - d) - I^R(p - d - r)|$ 是我们优化方向上两个相邻位置之间的图像强度的差异。我们按照以下规则设定 P_1 和 P_2 ：

$$\begin{aligned} P_1 &= \text{sgm_P1}, & P_2 &= \text{sgm_P2} & \text{if } D_1 < \text{sgm_D}, D_2 < \text{sgm_D}; \\ P_1 &= \text{sgm_P1}/\text{sgm_Q2}, & P_2 &= \text{sgm_P2}/\text{sgm_Q2} & \text{if } D_1 \geq \text{sgm_D}, D_2 \geq \text{sgm_D}; \\ P_1 &= \text{sgm_P1}/\text{sgm_Q1}, & P_2 &= \text{sgm_P2}/\text{sgm_Q1} & \text{otherwise.} \end{aligned}$$

超参数 sgm_P1 和 sgm_P2 为视差图中的不连续点设置了基础惩罚因子。如果 D_1 或 D_2 中的一个表示高图像梯度，基础惩罚因子由 sgm_Q1 因子减少，如果 D_1 和 D_2 同时表示高图像梯度，则通过较大的 sgm_Q2 因子来减少。当进一步考虑两个垂直方向时， P_1 的值将通过 sgm_V 因子减小；在地面的真实数据中，视差的微小变化在垂直方向上比在水平方向上更频繁，而且惩罚因子应该更小。

最终代价 $C_{SGM}(\mathbf{p}, d)$ 是在所有四个方向取平均值计算得到的：

$$C_{SGM}(\mathbf{p}, d) = \frac{1}{4} \sum_r C_r(\mathbf{p}, d).$$

正如在上一节所说，在半全局匹配之后，我们重复进行基于交叉的代价聚合。超参数 $\text{cbca_num_iterations_1}$ 和 $\text{cbca_num_iterations_2}$ 确定半全局前后基于交叉的代价聚合的迭代次数。

4.3 计算视差图

视差图 $D(\mathbf{p})$ 是通过胜者全拿的策略计算得到的，即寻找使 $C(\mathbf{p}, d)$ 最小的视差 d ,

$$D(\mathbf{p}) = \arg \min_d C(\mathbf{p}, d).$$

4.3.1 插值

插值步骤试图解决从左右两张图片中预测得到的视差图之间的冲突。令 D^L 表示通过以左图为参考得到的视差图——这是到目前为止情况，即， $D^L(\mathbf{p})=D(\mathbf{p})$ ，令 D^R 表示以右图为参考得到的视差图。 D^L 和 D^R 有时在一些特殊点的视差是不同的。我们通过左右一致性检查来检测这些冲突。我们通过应用以下规则来按顺序标记每个位置 \mathbf{p} ：

$$\begin{array}{ll} \text{正确} & \text{if } |d - D^R(\mathbf{p} - \mathbf{d})| \leq 1 \text{ for } d = D^L(\mathbf{p}), \\ \text{不匹配} & \text{if } |d - D^R(\mathbf{p} - \mathbf{d})| \leq 1 \text{ for any other } d, \\ \text{遮挡} & \text{其他情况} \end{array}$$

对于标记为遮挡的位置，我们希望从背景获取新的视差值。我们向左移直到我们找到一个标记为正确的位置并用它的值来内插。对于标记为不匹配的位置，我们在 16 个不同方向中寻找最近的正确像素，并取这些视差的中值进行插值。我们称插值的视差图为 D_{INT} 。

4.3.2 亚像素增强

亚像素增强提供了增加立体视觉算法的分辨率的简单方法。我们通过相邻代价拟合二次曲线来获得新的视差图：

$$D_{SE}(\mathbf{p}) = d - \frac{C_+ - C_-}{2(C_+ - 2C + C_-)},$$

其中 $d = D_{INT}(\mathbf{p})$, $C_- = C_{SGM}(\mathbf{p}, d - 1)$, $C = C_{SGM}(\mathbf{p}, d)$, $C_+ = C_{SGM}(\mathbf{p}, d + 1)$.

4.3.3 细化

立体视觉方法的最后步骤包括一个 5×5 中值滤波器和下列的双边滤波器：

$$D_{BF}(\mathbf{p}) = \frac{1}{W(\mathbf{p})} \sum_{\mathbf{q} \in N_p} D_{SE}(\mathbf{q}) \cdot g(\|\mathbf{p} - \mathbf{q}\|) \cdot 1\{|I^L(\mathbf{p}) - I^L(\mathbf{q})| < blur_threshold\}$$

其中 $g(x)$ 是标准差为 $blur_threshold$ 的零均值正态分布概率密度函数， $W(\mathbf{p})$ 是归一化常数，

$$W(\mathbf{p}) = \sum_{\mathbf{q} \in N_p} g(\|\mathbf{p} - \mathbf{q}\|) \cdot 1\{|I^L(\mathbf{p}) - I^L(\mathbf{q})| < blur_threshold\}$$

双边滤波器的作用是平滑视差图而不模糊边缘。 D_{BF} 为我们的立体视觉方法的最终输出。

5. 实验

我们在实验中使用了三个立体视觉数据集：**KITTI 2012**, **KITTI 2015** 和 **Middlebury**。表 1, 2 和 4 中的测试集误差率是通过提交所生成的视差图到在线评估服务器后获得。所有的其他错误率是通过将数据集分成两部分计算得到，我们使用一部分用于训练，另一部分用于验证。

表 1: 截至 2015 年 10 月 KITTI 2012 数据集的最高排名方法。“设置”列提供对视差图如何计算的解释：“F”表示光流的使用，“MV”表示两个以上的时间相邻图像，“MS”表示使用对极几何计算光流。“错误率”列表示错误分类像素的百分比，“运行时间”列表示处理一对图像所需的时间（以秒为单位）。

排名	方法	设置	错误率	运行时间
1	MC-CNN-acrt Accurate architecture		2.43	67
2	Displets Güney and Geiger (2015)		2.47	265
3	MC-CNN Zbontar and LeCun (2015)		2.61	100
4	PRSM Vogel et al. (2015)	F, MV	2.78	300
	MC-CNN-fst Fast architecture		2.82	0.8
5	SPS-StFl Yamaguchi et al. (2014)	F, MS	2.83	35
6	VC-SF Vogel et al. (2014)	F, MV	3.05	300
7	Deep Embed Chen et al. (2015)		3.10	3
8	JSOSM Unpublished work		3.15	105
9	OSF Menze and Geiger (2015)	F	3.28	3000
10	CoR Chakrabarti et al. (2015)		3.30	6

表 2: 截至 2015 年 10 月 KITTI 2015 排行榜上的领先排名。“设置”，“错误率”和“运行”列意义和表 1 相同。

排名	方法	设置	错误率	运行时间
1	MC-CNN-acrt Accurate architecture		3.89	67
	MC-CNN-fst Fast architecture		4.62	0.8
2	SPS-St Yamaguchi et al. (2014)		5.31	2
3	OSF Menze and Geiger (2015)	F	5.79	3000
4	PR-Sceneflow Vogel et al. (2013)	F	6.24	150
5	SGM+C+NL Hirschmüller (2008); Sun et al. (2014)	F	6.84	270
6	SGM+LDOF Hirschmüller (2008); Brox and Malik (2011)	F	6.84	86
7	SGM+SF Hirschmüller (2008); Hornacek et al. (2014)	F	6.84	2700
8	ELAS Geiger et al. (2011)		9.72	0.3
9	OCV-SGBM Hirschmüller (2008)		10.86	1.1
10	SDM Kostková and Sára (2003)		11.96	60

5.1 KITTI 立体视觉数据集

KITTI 立体数据集（Geiger 等人，2013; Menze 和 Geiger，2015）是从安装在汽车车顶上两个相隔约 54 厘米的摄像机拍摄的经整流的图像对的集合。图像是在白天晴朗和多云的天气下，在 Karlsruhe 市中心和郊区开车记录的。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。

测试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，例如，错误的容许范围为 3 厘米则转换成物理距离为距离相机 2 米和 80 厘米的容许范围，就是距离相机 10 米。

存在两个 KITTI 立体视觉数据集：KITTI 2012 和较新的 KITTI 2015。为了完成立

体视觉的计算任务，他们几乎是相同的，较新的数据集的改进了光流任务的一些方面。2012 年的数据集包含 194 个训练图和 195 个测试图像，而 2015 年的数据集包含 200 个训练图和 200 个测试图。较新的数据集介绍了一个微妙但重要的区别：运动车辆是密集标记的，并且汽车玻璃也被包括在评估中。这强调了该方法在反射面上的表现。

在 KITTI 2012 数据集表现最好的方法已经列在表 1 上。我们的精确架构排名第一，仅有 2.43% 的误差率。在排行榜第三名是我们之前的工作成果 (Zbontar 和 LeCun, 2015)，误差率为 2.61%。误差从 2.61% 减少到 2.43% 得益于两个改变——扩大了数据集（见第 5.4 节）和倍增了卷积层的数量，同时减少内核大小从 5×5 到 3×3 。排在第二位的方法 (Güney 和 Geiger, 2015) 用的是我们之前的方法计算匹配代价 (Zbontar 和 LeCun, 2015)。测试快速架构的错误率是 2.82%，这将是足够在排行榜上排到第五位。精确结构处理单一图像对的运行时间是 67 秒，而快速架构为 0.8 秒。图 5 包含了一对来自 KITTI 2012 数据集的例子，以及按我们的方法进行预测的结果。

表 2 列出了 KITTI 2015 数据集的领先者。我们方法中精确架构的错误率为 3.89%，而快速架构为 4.46%，占据了排行榜的第一和第二位。由于每篇论文只被允许提交一次，所以只有精确架构的结果出现在排行榜上。图 6 是 KITTI 2015 数据集按我们的方法产生视差图。

表 3: 五个 Middlebury 立体视觉数据集的总结。“Number of Image Pairs” 列只计算有地面真实视差的图像对的数量。2005 年和 2014 年的数据集还包含了一些保留地面真实视差的图像对;这些图像对构成了测试数据集。

Year	Number of Image Pairs	Resolution	Maximum Disparity
2001	8	380×430	30
2003	2	1800×1500	220
2005	6	1400×1100	230
2006	21	1400×1100	230
2014	23	3000×2000	800

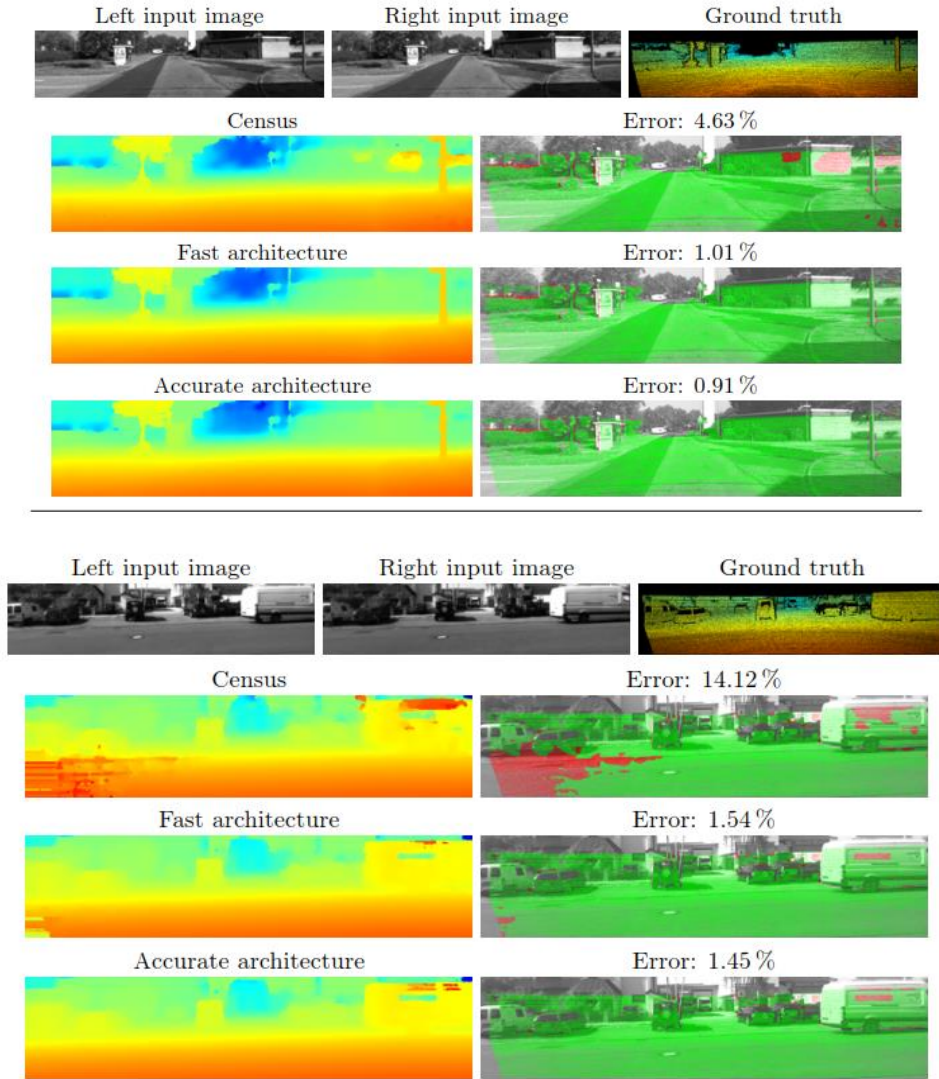


图 5: 对 KITTI 2012 数据集预测视差图的实例。注意图像的某些区域（顶部例子的白墙，底部例子的沥青）如何导致统计变换出现问题。快速和精确架构都有更好的表现，精确架构平均更少出现错误。

5.2 Middlebury 立体视觉数据集

Middlebury 立体视觉数据集的图像对来自光照可控的室内场景。数据通过结构光来测量真实视差，视差的密度和精度都比在 KITTI 数据集好。该数据集共发布了五个独立的数据集，分别是在 2001 年，2003 年，2005 年，2006 年和 2014 年（Scharstein 和 Szeliski, 2002, 2003; Scharstein 和 Paul, 2007; Hirschmüller 和 Scharstein, 2007; Scharstein 等人, 2014）。在本文中，我们指的 Middlebury 数据集是所有的五个数据集的串联;表 3 是各个数据集的情况总结。

在 2005 年, 2006 年, 和 2014 年数据集的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, 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。

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。这条规则比允许每三天提交一次的 KITTI 数据集更严格。测试数据集包含 15 幅来自 2005 年和 2014 年数据集的图像。

数据集提供全分辨率, 半分辨率和四分之一分辨率的图片。误差率的计算是根据全分辨率的; 如果算法输出的是半分辨率或四分之一分辨率的视差图, 它们在计算错误率之前上采样。由于显卡内存有限, 我们选择半分辨率的图片来运行我们的算法。

通过标准的校准程序来矫正图像, 像 OpenCV 库中的那样, 会在 Middlebury 数据集中导致多达九个像素的垂直视差错误 (Scharstein 等人, 2014)。2014 年数据集中的每个立体图像对都整流了两次: 一次是使用标准的不完美的方法, 另一次是采用严格的 2D 对应来达到完美的矫正 (Scharstein 等, 2014)。我们使用不完美的矫正图像对来训练深度学习网络, 因为十五对测试图像中仅有 2 对是完美矫正的。

误差率是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; 这相当于半分辨率中的错误容限是一个像素。在默认情况下, 评估服务器上的错误率仅仅计算非遮挡的像素点。最终在线报告上的错误率是十五对测试图像的加权平均, 权重由数据集的作者设置。

表 4 包含了一张第三名的快照, 以及最新版本的 Middlebury 排行榜。我们的算法排在第一, 是 8.29% 的误差率, 并在大幅领先第二名 MeshStereo 的算法, 它的误差率是 13.4%。图 7 是我们根据 Middlebury 数据集中的一个图像对计算的视差图。

表 4: 截至 2015 年 10 月, 在 Middlebury 立体视觉数据集中的排名前十的算法, “Error” 列是上采样成全分辨率图片后的加权平均误差。“Runtime” 是处理一对图像的运行时间, 以秒为单位。

Rank	Method		Resolution	Error	Runtime
1	MC-CNN-acrt	Accurate architecture	Half	8.29	150
2	MeshStereo	Zhang et al. (2015)	Half	13.4	65.3
3	LCU	Unpublished work	Quarter	17.0	6567
4	TMAP	Psota et al. (2015)	Half	17.1	2435
5	IDR	Kowalczyk et al. (2013)	Half	18.4	0.49
6	SGM	Hirschmüller (2008)	Half	18.7	9.90
7	LPS	Sinha et al. (2014)	Half	19.4	9.52
8	LPS	Sinha et al. (2014)	Full	20.3	25.8
9	SGM	Hirschmüller (2008)	Quarter	21.2	1.48
10	SNCC	Einecke and Eggert (2010)	Half	22.2	1.38

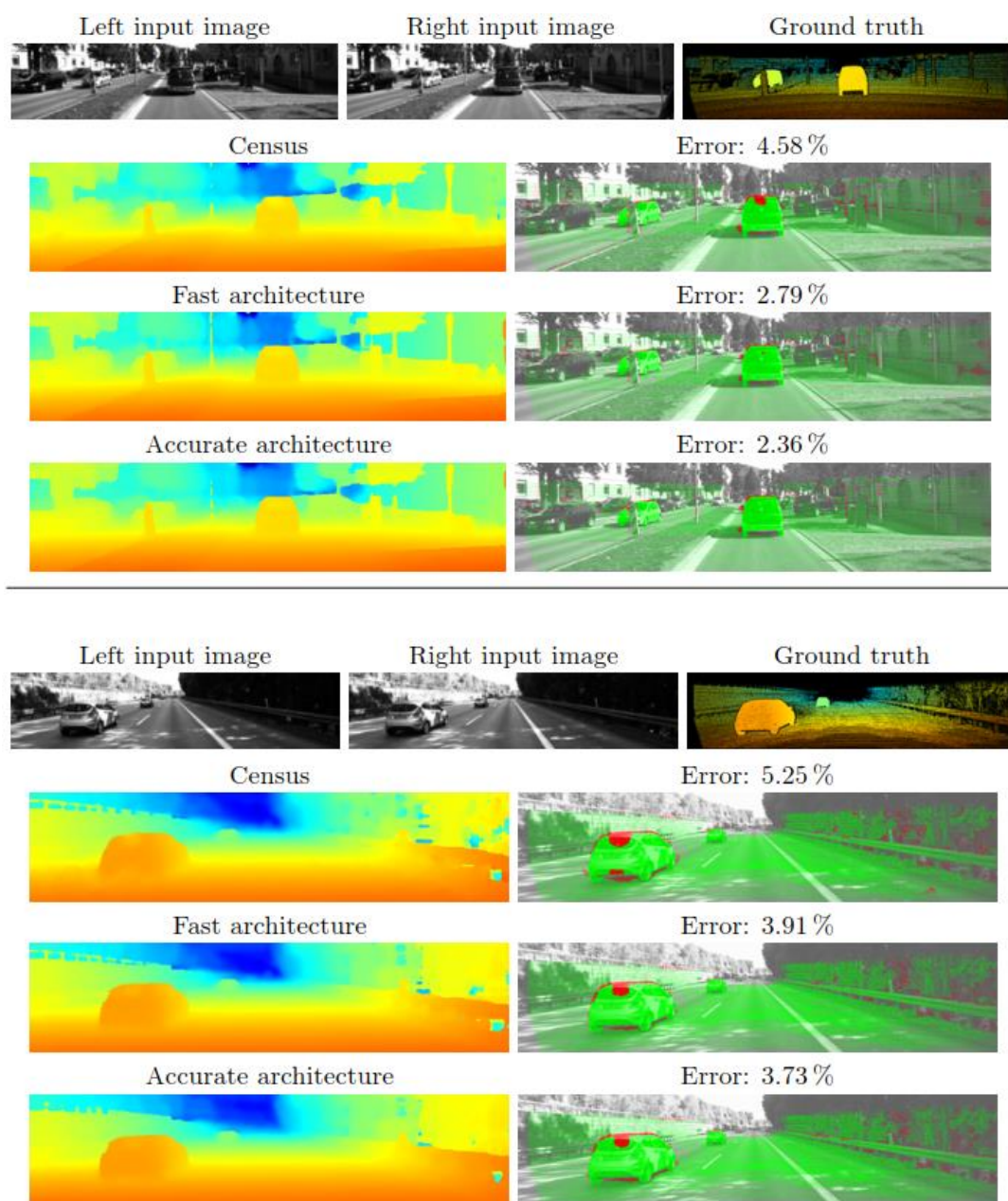


图 6: 在 KITTI 2015 数据集的预测实例。可以看出运动车辆在 KITTI 2015 数据集中是密集标记的。

5.3 深度学习细节

我们用训练数据集中所有可用的图像对构建了一个二元分类数据集。该数据集包含了 KITTI 2012 的 2500 万实例和 KITTI 2015 的 1700 万实例，以及 Middlebury 数据集的 3800 万实例。

在训练时，网络的输入是一批 128 对的图像块。在测试期间，输入是整个的左和右

图像。在训练期间，我们也可以使用整个的图像，这将使我们能够实现 3.3 节中所说的速度优化。我们更愿意使用图像块训练的原因是：这是更容易控制批次的大小，这些例子可以被混合在一起，使得同一个批次的图像块可以分别来自几个不同图像，同时这也比较容易维持同一批内积极实例和消极实例的数量相同。

我们使用动量项为 0.9 的小批梯度下降来减少损失。

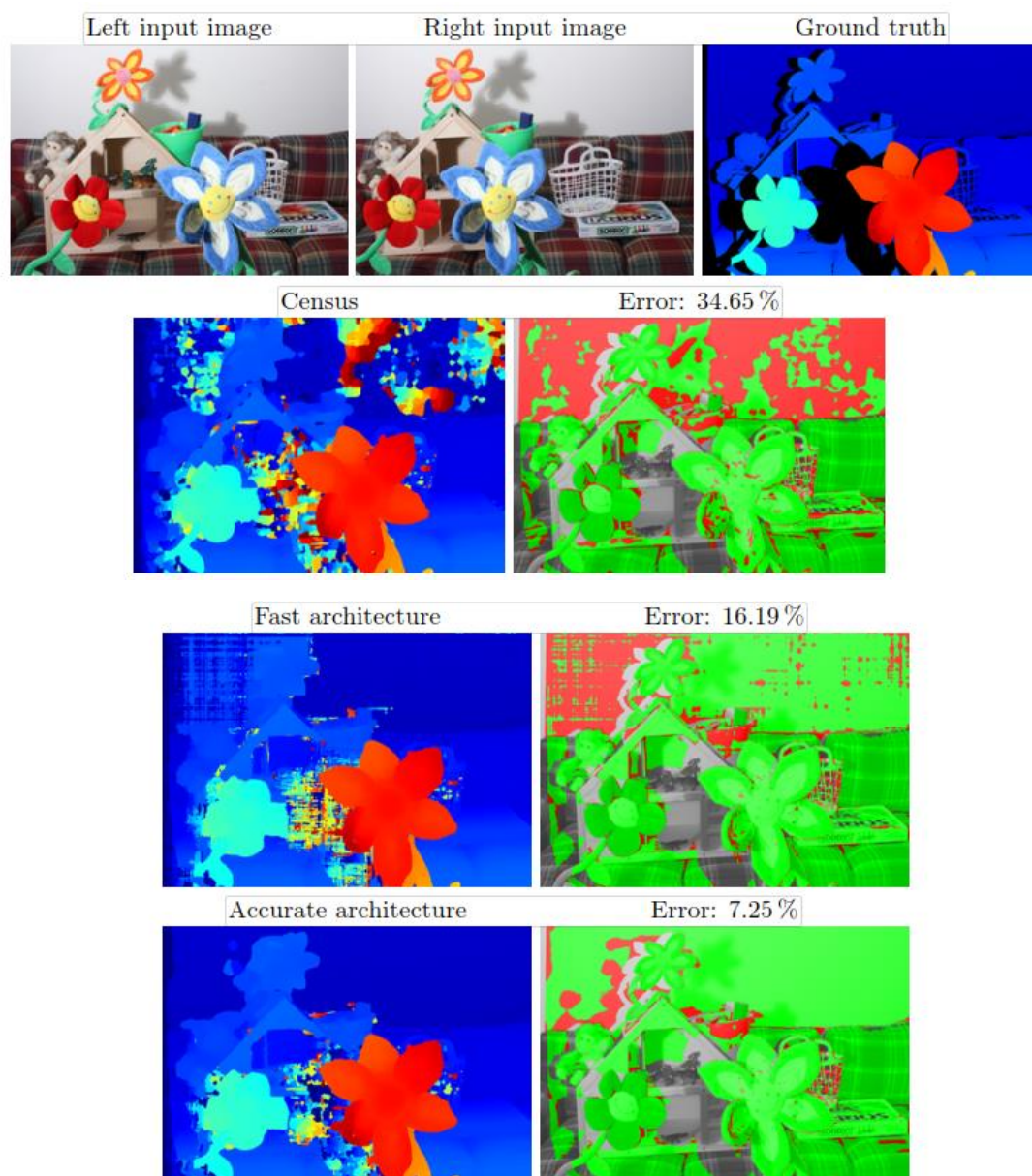


图 7: Middlebury 数据集中一个特别难的图像对示例；白色的背景墙几乎是无纹理。精确架构能够区分大多数的内容。快速架构做的并没有那么好但仍比统计方法更好。

我们训练迭代 14 次，对于精确架构学习率的初始设置为 0.003，快速架构的初始学

习率为 0.002。学习率在 11 次迭代时减少 10 倍。迭代次数，初始学习率以及学习率衰减计划，都作为超参数并用交叉验证进行优化。每个图像都通过减去平均值并除以像素强度值的标准差来预处理。立体图像对的左和右图分别预处理。我们的初步实验表明，使用彩色信息不会改善视差图的质量;因此，我们把所有的彩色图片都转换成灰度图。

立体视觉方法的后处理步骤在 CUDA (Nickolls 等人,2008) 中得到实施，该网络的训练是使用从 cuDNN 库 (Chetlur 等人, 2014) 的卷积例程在 Torch 环境中完成的 (Collobert 等人, 2011)。OpenCV 库 (Bradski, 2000) 被用于在数据集扩增步骤的仿射变换。

超参数通过手动搜索和简单的自动化脚本优化。表 5 是我们选择的超参数。

表 5: 我们用于快速和精确架构的超参数值 (缩写 “FST” 和 “ACRT”)。需要注意的是超参数有关图像强度的值 (cbca_intensity 和 sgm_D) 仅适用于经过预处理的图像，而不是强度值范围为 0 到 255 的原始图像。

Hyperparameter	KITTI 2012		KITTI 2015		Middlebury	
	fst	acrt	fst	acrt	fst	acrt
input_patch_size	9 × 9	9 × 9	9 × 9	9 × 9	11 × 11	11 × 11
num_conv_layers	4	4	4	4	5	5
num_conv_feature_maps	64	112	64	112	64	112
conv_kernel_size	3	3	3	3	3	3
num_fc_layers		4		4		3
num_fc_units		384		384		384
dataset_neg_low	4	4	4	4	1.5	1.5
dataset_neg_high	10	10	10	10	6	18
dataset_pos	1	1	1	1	0.5	0.5
cbca_intensity		0.13		0.03		0.02
cbca_distance		5		5		14
cbca_num_iterations_1		2		2		2
cbca_num_iterations_2		0		4		16
sgm_P1	4	1.32	2.3	2.3	2.3	1.3
sgm_P2	223	32	42.3	55.8	55.9	18.1
sgm_Q1	3	3	3	3	4	4.5
sgm_Q2	7.5	6	6	6	8	9
sgm_V	1.5	2	1.25	1.75	1.5	2.75
sgm_D	0.02	0.08	0.08	0.08	0.08	0.13
blur_sigma	7.74	6	4.64	6	6	1.7
blur_threshold	5	6	5	5	2	2

5.4 数据集扩增

通过反复训练实例的变换来扩增数据集是一种常用的技术，用来减少网络的泛化误差。这种变换是在训练时进行，不会影响运行时性能。我们随机旋转，缩放和剪切训练

的图像块;我们还改变其亮度和对比度。因为是在提取图像后才对图像块变换,所以扩增数据集的步骤不改变地面真实视差图或破坏矫正。

变换的参数是每对图像对随机选取,每经过一次迭代后,相同的实例被第二次提供给网络,新的随机参数也被选择。我们对左右图像选择略有不同的变换参数;例如,我们会旋转左图 10 度而右图是 14 度。不同的数据集从不同类型的变换中受益,在某些情况下,使用错误的转换会增加错误率。在 Middlebury 数据集中,我们充分利用的既有事实,即所有的可用图像是在不同的光照条件和不同的快门曝光下拍摄的。这些数据集的扩增参数同样被用于 KITTI 2012 和 KITTI 2015 数据集中。

Middlebury 测试数据集中有两张值得一提图片:图片“教室”,它的右图曝光不足,因此,比左侧更暗;图片“非洲鼓”,它的左和右图像分别在不同光照条件下拍摄的。为了处理这两种情况,我们花了 20%的时间训练快门曝光或灯光安排不同的情况下的左右图像。

我们通过在左右图像块之间加一个小的垂直视差来改善 Middlebury 数据集中的不完美矫正。

在描述数据扩增的步骤之前,让我们介绍一些符号:在下面,typewriter 字体的词被用来表示定义一个集合的超参数的名称,而 italic (斜体)字体的同一个词被用来表示从集合中随机抽取的数字。例如,rotate 是定义一组可能的旋转角度的集合,rotate 是从集合中随机抽取的数字。下表是数据扩增步骤的介绍:

- 左图像块旋转 rotate 度,右图像块旋转 rotate+rotate_diff 度。
- 左图像块缩放 scale 倍,右图像块缩放 scale*scale_diff 倍。
- 左图像块在水平方向上拉伸 horizontal_scalescale 倍,右图像块拉伸 horizontal_scalescale*horizontal_scalescale_diff 倍。
- 左图像块在水平方向上剪切 horizontal_shear,右图像块在水平方向上剪切 horizontal_shear+horizontal_shear_diff。
- 在垂直方向上调整右图像块 vertical_disparity.的视差
- 调整由左和右图像的补丁的亮度和对比度设定为:

$$\begin{aligned} \mathcal{P}^L &\leftarrow \mathcal{P}^L \cdot \text{contrast} + \text{brightness} \text{ and} \\ \mathcal{P}^R &\leftarrow \mathcal{P}^R \cdot (\text{contrast} \cdot \text{contrast_diff}) + (\text{brightness} + \text{brightness_diff}), \end{aligned}$$

逐项适当的加法和乘法。

表 6 包含使用的超参数和数据扩增步骤是如何影响验证误差的。

表 6: 管理数据扩增的超参数以及它们是如何影响验证误差的。“Error”列统计未使用某个特殊数据扩增步骤时的验证误差。最后两行统计使用或不使用数据扩增的验证错误。例如，如果没有数据扩增，KITTI 2012 的验证误差是 2.73%；如果除了旋转以外的所有步骤都被使用，验证误差为 2.65%；如果使用全部数据扩增步骤则验证误差为 2.61%。

Hyperparameter	KITTI 2012		Middlebury	
	Range	Error	Range	Error
rotate	[-7, 7]	2.65	[-28, 28]	7.99
scale			[0.8, 1]	8.17
horizontal_scale	[0.9, 1]	2.62	[0.8, 1]	8.08
horizontal_shear	[0, 0.1]	2.61	[0, 0.1]	7.91
brightness	[0, 0.7]	2.61	[0, 1.3]	8.16
contrast	[1, 1.3]	2.63	[1, 1.1]	7.95
vertical_disparity			[0, 1]	8.05
rotate_diff			[-3, 3]	8.00
horizontal_scale_diff			[0.9, 1]	7.97
horizontal_shear_diff			[0, 0.3]	8.05
brightness_diff	[0, 0.3]	2.63	[0, 0.7]	7.92
contrast_diff			[1, 1.1]	8.01
No data set augmentation		2.73		8.75
Full data set augmentation		2.61		7.91

在 KITTI 2012 数据集中，数据扩增把验证误差从 2.73%降低到了 2.61%，而在 Middlebury 数据集中是从 8.75%降至 7.91%。

5.5 运行时间

我们在一台带有图形处理器单元 NVIDIA Titan X 的电脑上运行我们的应用并测量运行时间。表 7 包含三个数据集在不同的超参数设置下的运行时间：KITTI，Middlebury 半分辨率，和一个新的，称为 Tiny 的虚拟数据集，这是我们用来验证我们算法用于自动驾驶或机器人时的性能。我们测量运行时间的图片大小是：针对 KITTI 数据集的是 1242×350 并含有 228 个视差水平，针对 Middlebury 数据集的是 1500×1000 并含有 200 个视差水平，以及针对 Tiny 数据集的是 320×240 并含有 32 个视差水平。

表 7 表明，快速架构比精确架构速度快高达 90 倍。此外，快速架构的运行时间在 KITTI 中为 0.78 秒，在 Middlebury 中为 2.03 秒，在 Tiny 数据集中为 0.06 秒。我们还可以看到在精确架构中全连接层占据了大多数的运行时间，控制卷积层的数量和特征映射的数目的超参数对运行时间仅有很小的影响。

训练次数取决于数据集的大小和结构，但从来没有超过两天。

表 7: 用来计算匹配代价的时间（秒），即花的没有任何后处理步骤卷积神经网络中的时间。这个时间不包括计算匹配代价两次：一次是当左图像被取为参考图像，另一次是右图像被取为参考图像。我们测量作为四个控制网络架构的超参数的函数的运行时间；例如，前六个行包含网络中卷积层的数量从一个增加到六个的运行时间。表中的最后一行包含整个算法的运行时间，包括后处理步骤。和前面一样，快速和精确架构的缩写为“fst”和“acrt”。

Hyperparameter		KITTI		Middlebury		Tiny	
		fst	acrt	fst	acrt	fst	acrt
num_conv_layers	1	0.23	66.1	0.70	74.9	0.01	1.7
	2	0.26	66.2	0.82	75.2	0.01	1.8
	3	0.30	66.3	0.97	75.4	0.02	1.8
	4	0.34	66.4	1.11	75.6	0.03	1.8
	5	0.38	66.5	1.24	75.7	0.03	1.9
	6	0.42	66.7	1.37	76.0	0.04	1.9
num_conv_feature_maps	16	0.09	59.4	0.27	64.8	0.01	1.6
	32	0.15	60.4	0.51	66.2	0.01	1.6
	48	0.25	61.5	0.94	68.2	0.02	1.7
	64	0.34	62.7	1.24	70.0	0.03	1.7
	80	0.44	64.0	1.63	72.0	0.04	1.8
	96	0.53	65.3	1.93	73.9	0.04	1.8
	112	0.61	66.4	2.28	75.7	0.05	1.8
	128	0.71	67.7	2.61	77.8	0.06	1.9
num_fc_layers	1		16.3		25.3		0.5
	2		32.9		50.7		0.9
	3		49.6		75.7		1.4
	4		66.4		101.2		1.8
	5		82.9		126.4		2.3
num_fc_units	128		17.4		21.4		0.6
	256		38.5		44.9		1.1
	384		66.4		75.7		1.8
	512		101.0		113.3		2.7
No stereo method		0.34	66.4	1.24	75.7	0.03	1.8
Full stereo method		0.78	67.1	2.03	84.8	0.06	1.9

5.6 匹配代价

我们认为，我们方法的低错误率是由于卷积神经网络而不是更优秀的立体视觉方法。我们通过用三个标准的计算匹配代价的方法更换卷积神经网络验证这种说法：

- 绝对差之和方法根据等式（1）计算匹配代价，也就是说，两个图像块之间的匹配代价是计算在相应的位置之间的图像强度的绝对差值之和。我们使用 9×9 的图像块。
- 统计变换方法（Zabih 和 Woodfill, 1994）认为每个图像的位置是比特向量。此

向量的大小是一个超参数，它的值，在测试几次后，我们设置为 81。这个向量通过计算被裁剪成围绕感兴趣位置的 9×9 的图像块，并比较图像块中每个像素的强度值和中心像素的强度值。如果中心像素更亮，相应位就被置位。匹配代价被计算为两次统计变换所得向量之间的汉明距离。

- 归一化的互相关方法是用下面等式定义的基于窗口的方法：

$$C_{NCC}(\mathbf{p}, d) = \frac{\sum_{\mathbf{q} \in N_p} I^L(\mathbf{q}) I^R(\mathbf{q} - d)}{\sqrt{\sum_{\mathbf{q} \in N_p} I^L(\mathbf{q})^2 \sum_{\mathbf{q} \in N_p} I^R(\mathbf{q} - d)^2}}$$

归一化互相关匹配代价是计算左右图像块的余弦相似度，左和右图像块被看作是向量，而不是矩阵。这是在快速架构的最后两层（归一化和点积）计算的同一个函数。

邻域 N_p 被设定为围绕 \mathbf{p} 的 11×11 的方形窗口。

表 8 的“sad”，“cens”和“ncc”列是上述三个方法在 KITTI 2012, KITTI 2015 和 Middlebury 数据集的结果。表 8 中最后一排的验证误差被用来比较五种方法。对所有这三个数据集，准确架构的效果最好，其次是快速架构，其后是统计变换。这是对所有三个数据集表现最好的三个方法。他们的错误率分别在 KITTI 2012 中为 2.61%，3.02% 和 4.90%；在 KITTI 2015 中为 3.25%，3.99% 和 5.03% 2015 年；在 Middlebury 中是 7.91%，9.87% 和 16.72%。绝对差之和的方法和归一化互相关方法的匹配代价产生较大错误率的视差图。我们的方法和统计变换的视觉对比见图 5,6,7。

5.7 立体视觉方法

立体视觉方法包括多个后处理步骤：基于交叉的代价聚合，半全局匹配，插值，亚像素增强，中值和双边滤波器。我们进行了一系列实验，在实验中我们依次排除上述的步骤并记录验证误差（见表 8）。

表 8 的最后两行暗示了立体视觉方法中后处理步骤的重要性。我们看到，如果所有的后处理步骤都被去除，精确架构的验证误差在 KITTI 2012 中从 2.61% 提高到 13.49%，在 KITTI 2015 中从 3.25% 升到 13.38%，在 Middlebury 中从 7.91% 升至 28.33%。

在立体视觉方法的所有后处理步骤中，半全局匹配对验证误差的影响最强。如果我们将它去除，在 KITTI 2012 中验证误差从 2.61% 上升到 4.26%，在 KITTI 2015 中从 3.25% 上升到 4.51%，在 Middlebury 中从 7.91% 上升到 11.99%。

我们没有使用左右一致性检查来消除 Middlebury 中遮蔽区域的误差。在精确架构中使用左右一致性检查，错误率从 7.91% 提高到了 8.22%，这就是我们决定将其删除的原因。

表 8: 当从立体视觉方法中去除某个特定的后处理步骤时，验证误差的数值。表的最后两行应有不同于之前的解释：它们是原始卷积神经网络的验证误差和完整的立体视觉方法后的验证误差。例如，如果我们排除半全局匹配，在 KITTI 2012 数据集中快速架构的错误率为 8.78% 和在应用整个立体视觉算法后为 3.02%。我们的简写“fst”为快速架构“acrt”的精确建筑，“sad”为绝对值之和的方法，“cens”为统计变换方法，而“ncc”为归一化互相关匹配代价。

	KITTI 2012				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	3.02	2.73	8.22	5.21	8.93
Semiglobal matching	8.78	4.26	19.58	8.84	10.72
Interpolation	3.48	2.96	9.21	5.96	11.16
Subpixel Enhancement	3.03	2.65	8.16	4.95	8.93
Median filter	3.03	2.63	8.16	4.92	9.00
Bilateral filter	3.26	2.79	8.75	5.70	9.76
No stereo method	15.70	13.49	32.30	53.55	22.21
Full stereo method	3.02	2.61	8.16	4.90	8.93
	KITTI 2015				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	3.99	3.39	9.94	5.20	8.89
Semiglobal matching	8.40	4.51	19.80	7.25	9.36
Interpolation	4.47	3.33	10.39	5.83	10.98
Subpixel Enhancement	4.02	3.28	9.44	5.03	8.91
Median filter	4.05	3.25	9.44	5.05	8.96
Bilateral filter	4.20	3.43	9.95	5.84	9.77
No stereo method	15.66	13.38	30.67	50.35	18.95
Full stereo method	3.99	3.25	9.44	5.03	8.89
	Middlebury				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	9.87	10.63	43.09	29.28	33.89
Semiglobal matching	25.50	11.99	51.25	19.51	35.36
Interpolation	9.87	7.91	41.86	16.72	33.89
Subpixel Enhancement	10.29	8.44	42.71	17.18	34.12
Median filter	10.16	7.91	41.90	16.73	34.17
Bilateral filter	10.39	7.96	41.97	16.96	34.43
No stereo method	30.84	28.33	59.57	64.53	39.23
Full stereo method	9.87	7.91	41.86	16.72	33.89

5.8 数据集大小

我们使用了一个监督学习的方法来衡量图像块之间的相似性。因此，我们自然要问数据集的大小是如何影响视差图质量的。要回答这个问题，我们用较小的训练集重新训练我们的网络，这个小训练集是通过选择一组随机的实例获得（见表 9）。

我们观察到，随着我们增加训练示例的数量，验证误差减少了。这些实验表明了一个改善我们立体视觉方法结果的简单策略：收集更大的数据集。

表 9： 验证误差是与训练集大小相关的函数

Data Set Size (%)	KITTI 2012		KITTI 2015		Middlebury	
	fst	acrt	fst	acrt	fst	acrt
20	3.17	2.84	4.13	3.53	11.14	9.73
40	3.11	2.75	4.10	3.40	10.35	8.71
60	3.09	2.67	4.05	3.34	10.14	8.36
80	3.05	2.65	4.02	3.29	10.09	8.21
100	3.02	2.61	3.99	3.25	9.87	7.91

5.9 迁移学习

到目前为止，训练集和验证集都是从同一个立体视觉数据集中产生的，无论是在 KITTI 2012，还是 KITTI 2015 或 Middlebury 数据集。为了评估我们算法在迁移学习方面的表现，我们再次进行实验，其中验证误差是计算在一个不同于之前训练时的数据集。例如，我们使用 Middlebury 数据集训练匹配代价神经网络，并在 KITTI 2012 数据集中评估它的性能。这些实验展示给我们一些算法在现实世界中的应用时预期的表现，因为没有可用的标签数据，所以在现实世界中是不可能训练出专门的网络的。这些实验的结果展示于表 10 中。

表 10 中的一些结果出人意料。例如，当使用 Middlebury 训练数据并在 KITTI 2012 中验证的误差比用 KITTI 2015 训练时更低，即使 KITTI 2012 数据集相比于 Middlebury 明显更类似 KITTI 2015。此外，当使用 KITTI 2015 训练网络时，使用快速架构的 KITTI 2012 的验证误差比使用精确架构时更低。

由 Middlebury 数据集训练的匹配代价神经网络，能更好的迁移到 KITTI 数据集中。其验证误差类似于 KITTI 数据集训练的网络的验证误差。

表 10: 当训练集和测试集不同时的验证误差。例如，当 Middlebury 数据集用于训练快速架构并使用 KITTI 2012 数据集测试时，验证误差为 3.16%。

		Test Set					
		KITTI 2012		KITTI 2015		Middlebury	
		fst	acrt	fst	acrt	fst	acrt
Training Set	KITTI 2012	3.02	2.61	4.12	3.99	12.78	11.09
	KITTI 2015	3.60	4.28	3.99	3.25	13.70	14.19
	Middlebury	3.16	3.07	4.48	4.49	9.87	7.91

5. 10 超参数

寻找一组好的超参数是一个艰巨的任务——搜索空间随着超参数的数量成指数增长并且没有梯度可循。为了更好地理解每个超参数对验证误差的影响，我们进行了一系列的实验，在实验中我们改变一个超参数的值，同时保持其他超参数固定为默认值。结果在表 11 中，我们可以观察到增加网络的大小提高了泛化性能，但只在一种特殊情况下，因为数据集的大小，泛化性能开始真的下降。

注意，`num_conv_layers` 超参数隐性控制着图像块的大小。例如，一个 3×3 内核的卷积层的网络意味着尺寸 3×3 的图像块，而有五个卷积层的网络意味着尺寸 11×11 的图像块。

6. 总结

我们提出了两个卷积神经网络架构来学习图像块相似性的测量并应用到立体匹配的问题中。

我们实现方法的源代码在 <https://github.com/jzbontar/mc-cnn>。在线仓库中包含计算视差图，训练网络以及立体视觉方法的后处理步骤的过程。

精确架构产生更低错误率的视差图，比任何先前公布在 KITTI 2012, KITTI 2015 和 Middlebury 数据集的方法更低。快速架构计算视差图比精确架构快将近 90 倍，并且只有小幅的错误率增长。这些结果表明，卷积神经网络非常适合计算立体视觉匹配成本，甚至可以适应那些需要实时性能的应用中。

在一个已经被充分研究的问题中，一个相对简单的卷积神经网络胜过先前所有的立体视觉算法的事实是对现代机器学习方法力量的一个相对重要的示范。

表 11: 一系列超参数设置下的验证误差。

Hyperparameter		KITTI 2012		KITTI 2015		Middlebury	
		fst	acrt	fst	acrt	fst	acrt
num_conv_layers	1	5.96	3.97	5.61	4.06	20.74	12.37
	2	3.52	2.98	4.19	3.45	12.11	9.20
	3	3.10	2.72	4.04	3.27	10.81	8.56
	4	3.02	2.61	3.99	3.25	10.26	8.21
	5	3.03	2.64	3.99	3.30	9.87	7.91
	6	3.05	2.70	4.01	3.38	9.71	8.11
num_conv_feature_maps	16	3.33	2.84	4.32	3.51	11.79	10.06
	32	3.15	2.68	4.12	3.35	10.48	8.67
	48	3.07	2.66	4.06	3.32	10.20	8.47
	64	3.02	2.64	3.99	3.30	9.87	8.12
	80	3.02	2.64	3.99	3.29	9.81	7.95
	96	2.99	2.68	3.97	3.27	9.62	8.03
	112	2.98	2.61	3.96	3.25	9.59	7.91
	128	2.97	2.63	3.95	3.23	9.45	7.92
num_fc_layers	1		2.83		3.50		8.52
	2		2.70		3.31		8.33
	3		2.62		3.30		8.06
	4		2.61		3.25		8.00
	5		2.62		3.29		7.91
num_fc_units	128		2.72		3.36		8.44
	256		2.65		3.28		8.03
	384		2.61		3.25		7.91
	512		2.60		3.23		7.90
dataset_neg_low	1.0	3.00	2.76	3.97	3.35	9.84	8.00
	1.5	3.00	2.71	3.97	3.33	9.87	7.91
	2.0	2.99	2.63	3.98	3.31	9.98	8.08
	4.0	3.02	2.61	3.99	3.25	10.20	8.66
	6.0	3.06	2.63	4.05	3.28	10.13	8.86
dataset_neg_high	6	3.00	2.72	3.98	3.30	9.87	8.59
	10	3.02	2.61	3.99	3.25	9.97	8.23
	14	3.04	2.61	4.02	3.25	10.00	8.05
	18	3.07	2.60	4.06	3.23	9.98	8.11
	22	3.07	2.61	4.05	3.24	10.16	7.91
dataset_pos	0.0	3.04	2.67	4.00	3.26	9.92	7.97
	0.5	3.02	2.65	3.99	3.28	9.87	7.91
	1.0	3.02	2.61	3.99	3.25	9.86	8.04
	1.5	3.04	2.62	4.04	3.27	10.00	8.34
	2.0	3.04	2.66	4.04	3.29	10.16	8.51

参考文献

- [1] G. Bradski. The OpenCV library. Dr. Dobb's Journal of Software Tools, 2000.
- [2] Jane Bromley, James W Bentz, L'eon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Sackinger, and Roopak Shah. Signature verification using a siamese time delay neural network. International Journal of Pattern Recognition and Artificial Intelligence, 7(04):669–688, 1993.
- [3] Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(1): 43–57, 2011.
- [4] Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(3):500–513, 2011.
- [5] Ayan Chakrabarti, Ying Xiong, Steven J. Gortler, and Todd Zickler. Low-level vision by consensus in a spatial hierarchy of regions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.
- [6] Zhuoyuan Chen, Xun Sun, Yinan Yu, Liang Wang, and Chang Huang. A deep visual correspondence embedding model for stereo matching costs. IEEE International Conference on Computer Vision (ICCV), 2015.
- [7] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. cuDNN: Efficient primitives for deep learning. CoRR, abs/1410.0759, 2014.
- [8] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In BigLearn, NIPS Workshop, 2011.
- [9] Nils Einecke and Julian Eggert. A two-stage correlation method for stereoscopic depth estimation. In Digital Image Computing: International Conference on Techniques and Applications (DICTA), pages 227–234, 2010.
- [10] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In Proceedings of the 10th Asian Conference on Computer Vision - Volume Part I, ACCV'10, pages 25–38. Springer-Verlag, Berlin, Heidelberg, 2011.

- [11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: the KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013.
- [12] Fatma Güney and Andreas Geiger. Displets: Resolving stereo ambiguities using object knowledge. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [13] Ralf Haeusler, Rahul Nair, and Daniel Kondermann. Ensemble learning for confidence measures in stereo vision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.
- [14] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. MatchNet: Unifying feature and metric learning for patch-based matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [15] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008.
- [16] Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [17] Heiko Hirschmüller and Daniel Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599, 2009.
- [18] Michael Hornacek, Andrew Fitzgibbon, and Carsten Rother. SphereFlow: 6 DoF scene flow from RGB-D pairs. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
- [19] Dan Kong and Hai Tao. A method for learning matching errors for stereo computation. *British Machine Vision Conference (BMVC)*, 2004.
- [20] Dan Kong and Hai Tao. Stereo matching via learning multiple experts behaviors. *British Machine Vision Conference (BMVC)*, 2006.
- [21] Jana Kostková and Radim Šára. Stratified dense matching for stereopsis in complex scenes. *British Machine Vision Conference (BMVC)*, 2003.
- [22] Jędrzej Kowalczyk, Eric T Psota, and Lance C Perez. Real-time stereo matching on CUDA using an iterative refinement method for adaptive support-weight correspondences. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(1):94–104, 2013.

- [23] Yann LeCun, L eon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [24] Yunpeng Li and Daniel P Huttenlocher. Learning for stereo vision using the structured support vector machine. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.
- [25] Xing Mei, Xun Sun, Mingcai Zhou, Haitao Wang, Xiaopeng Zhang, et al. On building an accurate stereo matching system on graphics hardware. *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 467–474, 2011.
- [26] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [27] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with CUDA. *Queue*, 6(2):40–53, 2008.
- [28] Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *IEEE International Conference on Computer Vision (ICCV)*, pages 91–99, 2015.
- [29] Martin Peris, Atsuto Maki, Sara Martull, Yasuhiro Ohkawa, and Kazuhiro Fukui. Towards a simulation driven stereo vision system. In *21st International Conference on Pattern Recognition (ICPR)*, pages 1038–1042, 2012.
- [30] Eric T Psota, J drzej Kowalczyk, Mateusz Mittek, and Lance C Perez. Map disparity estimation using hidden markov trees. *IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [31] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Deepmatching: Hierarchical deformable dense matching. *ArXiv e-prints*, 1(7):8, 2015.
- [32] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007.
- [33] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3): 7–42, 2002.
- [34] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured

- light. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June.2003.
- [35]Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. German Conference on Pattern Recognition (GCPR), September 2014.
- [36]Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning local feature descriptors using convex optimisation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(8):1573–1585, 2014.
- [37]Sudipta N Sinha, Daniel Scharstein, and Richard Szeliski. Efficient high-resolution stereo matching using local plane sweeps. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014.
- [38]Aristotle Spyropoulos, Nikos Komodakis, and Philippos Mordohai. Learning to detect ground control points for improving the accuracy of stereo matching. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014.
- [39]Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. International Journal of Computer Vision, 106(2):115–137, 2014.
- [40]Tomasz Trzcinski, Mario Christoudias, Vincent Lepetit, and Pascal Fua. Learning image descriptors with the boosting-trick. In Advances in neural information processing systems, pages 269–277, 2012.
- [41]Christoph Vogel, Konrad Schindler, and Stefan Roth. Piecewise rigid scene flow. IEEE International Conference on Computer Vision (ICCV), 2013.
- [42]Christoph Vogel, Stefan Roth, and Konrad Schindler. View-consistent 3D scene flow estimation over multiple frames. European Conference on Computer Vision (ECCV), September 2014.
- [43]Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D scene flow estimation with a piecewise rigid scene model. International Journal of Computer Vision, pages 1–28, 2015.
- [44]Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. European Conference on Computer Vision (ECCV), September 2014.
- [45]Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual

- correspondence. European Conference on Computer Vision (ECCV), 1994.
- [46]Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.
- [47]Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.
- [48]Chi Zhang, Zhiwei Li, Yanhua Cheng, Rui Cai, Hongyang Chao, and Yong Rui. Meshstereo: A global stereo model with mesh alignment regularization for view interpolation. IEEE International Conference on Computer Vision (ICCV), 2015.
- [49]Ke Zhang, Jiangbo Lu, and Gauthier Lafruit. Cross-based local stereo matching using orthogonal integral images. IEEE Transactions on Circuits and Systems for Video Technology, 19(7):1073–1079, 2009.
- [50]Li Zhang and Steven M Seitz. Estimating optimal parameters for MRF stereo from a single image pair. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(2): 331–342, 2007.

一个训练视差，光流，场景流估计卷积网络的大型数据集

Philip Hausser , Philipp Fischer , Daniel Cremers, 慕尼黑工业大学

Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, Thomas Brox, 弗赖堡大学

翻译人 王灏 专业班级 通信工程 1301

摘要：最近的工作表明，光流估计可以被制定为监督学习的任务，并且可以用卷积网络解决。训练所谓的 FlowNet 是由大型的综合生成的数据集实现的。本文将利用卷积网络进行光流估计的概念扩展到视差和场景流估计中。为此，我们提出 三个合成立体视频数据集，它们具有充足的真实性，变化情况和大小范围来成功地训练大型网络。我们的数据集是第一个允许训练和评估场景流方法的大规模数据集。除了数据集，我们还提出了一个实时视差估计的卷积网络来提供最先进的结果。通过组合流和视差估计网络并对其进行训练它们，我们演示了第一个使用卷积网络的场景流估计。

1. 介绍

估计场景流意味着提供立体视频中所有可见点的深度和 3D 运动矢量。当谈到重建和运动估计时，它是“皇家联盟”任务并提供了一个重要的基础给很多更高层次的挑战，例如先进的驾驶员的辅助系统和自主系统。最近几十年来的研究一直专注于它的子任务，即视差估计和光流估计，并取得了相当大的成功。整个的场景流问题还没有开发到这种程度。既然局部场景流可以简单的从子任务结果中集合出来，我们可以猜测所有组件的联合估计对效率和精确率都是有利的。场景流比其子任务更少被探究的原因之一似乎是缺乏完全注释的地面实况数据。

在卷积网络的时代，这种数据的可用性已经变得更加重要。论文 Dosovitskiy^[4]表明光流估计可以看作一个监督学习问题并且可以用一个大型网络解决。为了训练他们的网络，他们创造了一个简单的合成的飞行椅的 2D 数据集并被证明足以预测一般视频的精

确的光流。这些结果表明，视差和现场流可以由卷积网络理想地联合的有效的实时的估计出来。实现这个想法缺少的是一个具有足够真实性和变化情况的大型数据集来训练这样的网络并评估它的性能。



图 1: 我们的数据集提供超过 35000 个立体帧，带有光流，视差和视差变化以及诸如对象间隔的其他数据的密集标签数据

在本文中，我们提出了三个这样数据集的集合，它们是通过使用开源的 3D 创建套件 Blender 的自定义版本创建的。我们的努力在本质上是和 Sintel benchmark^[2]类似的。与 Sintel 的工作相比，我们的数据集足够大来帮助训练卷积网络，而且它为场景流提供了地面实况。特别地，它包括立体彩色图像和双向视差，双向光流，视差变化，运动边界和对象间隔的地面真实情况。此外，相机完全标定和 3D 点定位是可用的，即我们的数据集也包括 RGBD 数据。

我们不能在单篇论文中利用这个数据集的全部潜力，但我们已经展示了各种与卷积网络训练相结合的用途示例。我们训练一个用于视差估计的网络，其在以前的基准上产生了有竞争力的性能，特别是在那些实时运行的方法中。最后，我们还提出了一个场景流估计的网络并提供在一个足够大的测试集上全场景流的第一个定量数字。

表 1: 可用数据集的比较: 我们的新集合提供了比任何现有选择更多的注释数据和更多的数据变化。我们的所有数据具有完全连续, 密集, 准确的地面实况。

Dataset	MPI Sintel [2]	KITTI Benchmark Suite [17]		SUN3D[27]	NYU2[18]	Ours		
		2012	2015			FlyingThings3D	Monkaa	Driving
#Training frames	1 064	194	800	2.5M	1 449	21 818	8 591	4 392
#Test frames	564	195	800	—	—	4 248	—	—
#Training scenes	25	194	200	415	464	2 247	8	1
Resolution	1024 × 436	1226 × 370	1242 × 375	640 × 480	640 × 480	960 × 540	960 × 540	960 × 540
Disparity/Depth	✓	sparse	sparse	✓	✓	✓	✓	✓
Disparity change	✗	✗	✗	✗	✗	✓	✓	✓
Optical flow	✓	(sparse)	(sparse)	✗	✗	✓	✓	✓
Segmentation	✓	✗	✗	(✓)	✓	✓	✓	✓
Motion boundaries	✓	✗	✗	✗	✗	✓	✓	✓
Naturalism	(✓)	✓	✓	✓	✓	✗	✗	(✓)

2. 相关工作

数据集。第一个做出重要的努力创建标准数据集的是用于立体视差估计^[22]和光流估计^[1]的 Middlebury 数据集。Middlebury 立体数据集由真实场景组成的, 光流数据集则是真实场景和渲染场景的混合。两个数据集在今天看来都非常小。特别是小测试集会严重的手动过拟合。立体数据集的一个优点是可用于相关真实的场景, 特别是在自 2014 年以来的最新的高分辨率版本^[21]。

MPI Sintel ^[2]是一个完全合成的数据集来源于一个短的开源动画 3D 电影。它提供了光流的密集地面实况。最近, 一个视差的测试版本可用于训练。Sintel 数据集有 1064 个训练帧是目前最大的可用数据集。它包含足够的现实场景, 其中包括自然的图像退化例如雾和运动模糊。作者将大量的努力投入到地面实况的所有帧和像素的矫正中。这使得这个数据集成为一个非常可靠的测试集用来进行算法的比较。然而, 对于卷积网络的训练, 数据集仍然太小。

KITTI 数据集制作于 2012 年^[8]并在 2015^[17]得以扩展。它包含来自安装在汽车上的已校准的一对相机的道路场景的立体视频。光流和视差的地面实况信息通过 3D 激光扫描仪结合汽车的运动数据获得。虽然这个数据集包含真实数据, 但是采集方法将地面实况信息限制在场景的某个静态部分。此外, 激光器仅提供高达某一特定距离和高度的稀疏数据。至于最新的版本, 汽车的 3D 模型与点云拟合以获得更致密的标签并且包括了移动物体。但是, 这些区域的地面实况仍然是一个近似值。

Dosovitskiy 等人^[4]为光流估计训练的卷积网络, 数据集是在移动的 2D 椅子图像叠加自然背景图像的合成数据集。这个数据集很大, 但仅限于单视图光流。它不包含 3D 动作, 并且尚未公开可用。

最新的 Sintel 数据集和 KITTI 数据集都可以用于估计具有一些限制的场景流。在遮挡区域（在一帧中可见，但在另一帧中不可见），场景流的地面实况不可用。在 KITTI 中，场景流最有趣的部分是 3D 点的运动，这部分是缺失的或者是通过拟合汽车 CAD 模型近似的。一个关于最重要的可比数据集及其特征的全面概述 在表 1 中给出了。

卷积网络。卷积网络^[16]已经证明对于各种识别任务是非常成功的，如图像分类^[15]。最近卷积网络的应用还包括单个图像的深度估计^[6]，立体匹配^[28]和光流估计^[4]。

Dosovitskiy 等人的 FlowNet^[4]是和我们的工作最相关的。它使用一个编码器--解码器架构，带有额外的收缩和扩展网络部分的交叉链路，其中编码器计算来自增大范围的接收场的抽象特征，解码器通过扩展的上卷积架构^[5]重建原始分辨率。我们采用这种方法来视差估计。

Zbontar^[28]等人的视差估计方法使用 Siamese 网络来计算图像块之间的匹配距离。为了实际估计视差，作者执行了基于交叉的代价聚合^[28]和半全局匹配（SGM）^[11]。与我们的工作相比，Zbontar 等人没有端到端的对视差估计任务训练卷积网络，相应的后果是计算效率和优雅。

场景流。虽然有数百篇关于视差和光流估计的论文，只有几篇是关于场景流的。它们之中没有人使用深度学习的方法。

场景流估计第一次通过 Vedula 等人的工作^[23]被推广，他们分析了不同的可能的问题设置。后来的工作是由各种变分方法主导的。Huguet 和 Devernay^[12]制定利用联合变分法的场景流估计。Wedel 等人^[26]遵循变分框架但是分离视差估计已获得更高的效率和准确性。Vogel 等人^[25]利用使用分段刚性模型正则化的超像素分割组合场景流估计的任务。Quiroga 等人^[19]扩展正则化矩阵到刚性运动的平滑场。像 Wedel 等人^[26]那样，他们分离视差估计并通过 RGBD 视频的深度值来代替。

在 KITTI 的场景流前七名中最快的方法是 Cech 等人^[23]，运行时间为 2.4 秒。他们方法采用种子生长算法同时视差和光流估计。

3. 场景流的定义

光流是真实世界中 3D 运动到图像平面上的投影。通常情况下，场景流被认为是从立体视频或 RGBD 视频计算潜在的 3D 运动场。假设立体对的两个连续的时间帧 t 和 $t+1$ ，产生四个图像 $(I_L^t, I_R^t, I_L^{t+1}, I_R^{t+1})$ 。场景流提供给在四个图像之一的每个可见点

的 3D 点定位及其 3D 运动矢量^[24]。

这些 3D 量只能在已知的相机内在和外在情况下计算。一个相机关于场景流的独立定义是通过分离光流，视差和视差变化分量^[12]获得的，见图 2。这种表述在这个意义上得以完善——如果相机参数是已知的，可见的 3D 点和他们的 3D 运动向量可以从上述分量中计算出来。

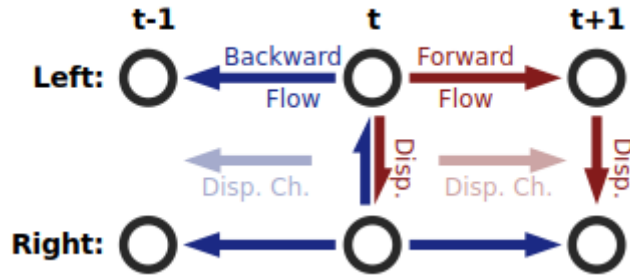


图 2: 给定时间 $t-1$, t 和 $t+1$ 的立体图像，箭头表示它们之间的视差和光流关系。红色的箭头通常用于估计场景流。在我们的数据集中，我们提供所有的关系，包括蓝色箭头。

给定 t 和 $t+1$ 的视差，视差变化几乎可以忽略。因此，在 KITTI 2015 场景流基准中^[17]，只评估光流和视差。在这种情况下，场景流只能对同时在左侧和右侧帧可见的表面点重建。特别是在卷积网络的背景下，在部分遮挡区域估计深度和运动是特别有趣的。此外，从流和视差中重建 3D 运动是对噪声更敏感的，因为光流中的一个小误差可能导致 3D 运动矢量中的巨大误差。

4. 三个已渲染的数据集

我们创建了一个由三个子集组成的合成数据集套件并提供了完整的前后方向的地面实况场景流（包括视差变化）。为此，我们使用了开源 3D 创建套件 Blender 来动画绘制大量复杂运动的对象和使结果呈现出成千上万帧。我们修改了 Blender 的内置渲染引擎的流水线生成——除立体 RGB 图像外——逐帧逐视图的三个额外的数据传递。这些数据提供了所有可见表面点的 3D 位置，以及它们未来和过去的 3D 位置。两个这样的为给定的相机视图结果的数据传递之间的像素差异——在 3D 运动矢量的“图像”中——完整的场景流地面实况正如这台相机所见。注意，即使在遮挡区域中信息也是完整的，因为渲染引擎总是知道所有的（可见的和不可见的）场景点。所有的透光材料——

明显地，大多数车窗——是被渲染为完全透明的以避免 3D 数据中的一致性问题。

给定相机的内在参数（焦距，主点）和渲染设置（图像大小，虚拟传感器尺寸和格式），我们投影每个像素点的 3D 运动矢量到与成像平面共面的 2D 像素运动向量：光流。深度是直接从像素的 3D 位置检索的，并通过虚拟立体视觉平台的已知配置转换为视差。我们从 3D 运动矢量的深度分量中计算出视差变化。结果的示例如图 1 所示。

此外，我们渲染了对象的分割掩码，其中每个像素值对应唯一的对象索引。对象可以由多个子分量组成，其中每个都可以有一个单独的材料（有自己的外在性质例如纹理）。我们利用这一点并渲染额外的分割掩码，其中每个像素编码其材料的索引。最近可用的 Sintel 的测试版本也包括这个数据。

类似于 Sintel 数据集，我们还提供运动边界，在两个或两个以上的移动对象间加亮像素，如果以下条件成立：两个帧之间的运动差异至少为 1.5 个像素，边界段覆盖至少 10 个像素的区域。阈值是根据 Sintel 的分割结果选定的。

对于所有的帧和视图，我们提供相机完整的内在和外在模型。那些模型可以用于运动的或者其他任务的相机追踪的结构。我们使用虚拟的 35mm 焦距的 32mm 宽的模拟传感器来渲染所有的图像数据。为了 Driving 数据集，我们添加了一个焦距为 15mm 的广角版本，其在视觉上更接近现有的 KITTI 数据集。

像 Sintel 数据集，我们的数据集还包括两个每个图像的特别版本：clean pass 版本显示颜色，纹理和场景光照但没有图像退化，而 final pass 版本还包括后处理工作诸如模拟的景深模糊，运动模糊，阳光眩光和伽马曲线操作。

为了处理大量的数据（2.5 TB），我们将所有 RGB 图像数据压缩到无损但高质量的 WebP 格式。非 RGB 数据通过 LZO 无损压缩。

4.1 FlyingThings3D

新数据集的主要部分由日常物体沿着随机的 3D 轨迹飞行组成。我们生成了约 25 000 个带地面实况数据的立体帧。我们不是专注于一个特定的任务（如 KITTI）或执行严格的自然主义（如 Sintel），而是依赖于随机性和一个巨大的渲染资源池来生成数量比任何现有的选项更多的数据，而且不存在重复或饱和的风险。数据生成是快速的，全自动的，并为完整的场景流任务产生密集精确的地面实况。创造这个数据集的动机是为了方便训练得益于大量的种类的大卷积网络。

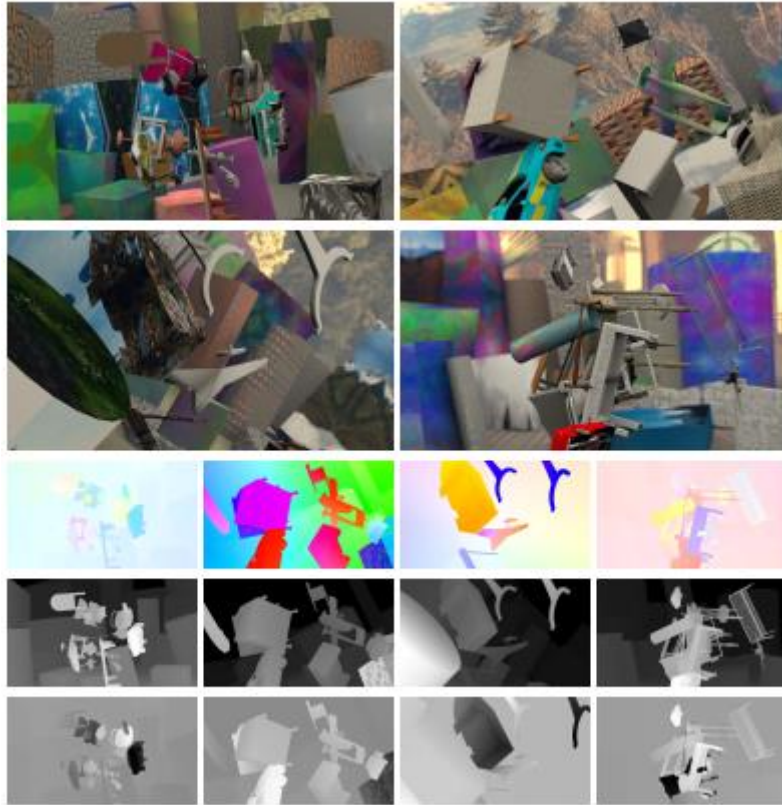


图 3: 来自我们的 FlyingThings3D 数据集的示例场景。**第三行:** 光流图, **第四行:** 视差图, **第五行:** 视差变化图。最好在高分辨率的彩色屏幕上观看 (图像归一化显示)。

每个场景的基础是一个大的纹理地平面。我们生成了 200 个静态背景对象, 它们的形状是从长方体和圆柱体中随机选择。每个对象随机缩放, 旋转, 纹理化然后放置在地平面上。

要填充场景, 我们下载了 35927 个来自斯坦福 ShapeNet 数据库的详细的 3D 模型^[20]。从这些模型中, 我们组合了一套 32872 个训练模型和一组大小为 3055 的测试模型。当然, 模型的类别是不相交。

我们从来自这个对象集的 5 到 20 个的随机对象中进行采样, 并随机的纹理化每个对象的每个材料。每个 ShapeNet 对象都已转化并沿着平滑的 3D 轨迹旋转建模, 使得相机可以看到对象, 但带有随机的位移。相机也是移动的。

纹理集的组成通过 ImageMagick 创建的过程图像, 来自 Flickr 的自然风景和城市景观照片以及来自 Image * After 的纹理样式图像。像 3D 模型一样, 纹理被分为不相交的训练和测试部分。

对于最终的通过图像, 场景存在不同强度的运动模糊和散焦模糊。

4.2 Monkaa

我们的数据集的第二部分来自开源的 Blender 的动画短片 Monkaa 资源。在这方面，它类似于 MPI Sintel 数据集。Monkaa 包含非刚性和软关节运动以及可见的有挑战性的皮毛。除此之外，几乎没有可见的与 Sintel 的相似之处；Monkaa 电影不追求相同数量的自然性。

我们选择了一些合适的电影场景并额外使用 Monkaa 的片段创建了全新的场景。为了增加数据量，我们渲染多个版本的自制场景，每个都有随机增量更改摄像机的平移和旋转的关键帧。

4.3 Driving

Driving 场景几乎都是从驾驶汽车的角度看去的自然的动态的街道场景，类似于 KITTI 数据集。它使用来自 FlyingThings3D 数据集相同资源池中的汽车模型，并额外使用来自 3D Warehouse 的高度详细的树模型和简单的路灯。在图 4 中，我们展示了从 Driving 中选定的帧以及 KITTI 2015 中相似的帧。

我们的立体视觉基线设置为 1 个 Blender 单元，还有典型的车型宽度约 2 个单位相当于 KITTI 的设置（54 厘米基线，186 厘米车宽^[8]）。

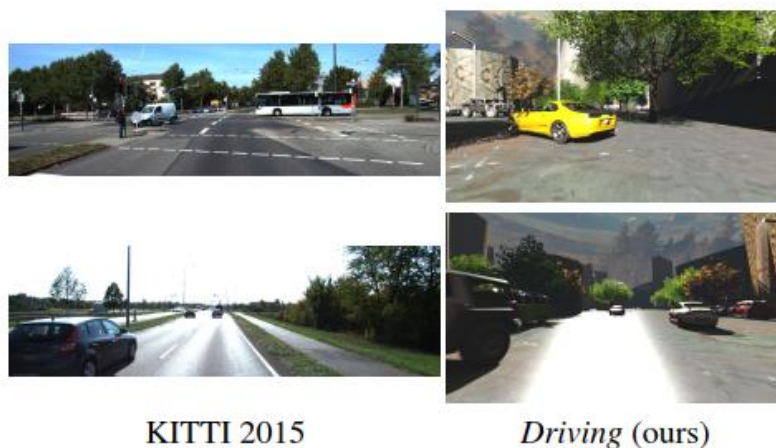


图 4: 来自 KITTI 2015 基准套件^[17] 和我们的新的 Driving 数据集的示例帧。两个都显示了许多从各种现实的视角的静态和移动汽车，稀薄的对象，复杂的阴影，纹理化的地面，具有挑战性的镜面反射。

5. 网络

为了证明我们新的合成数据集在场景流估计中的适用性，我们使用它来训练卷积网络。一般来说，我们遵循 FlowNet^[4]架构。也就是说，每个网络包含收缩部分和扩展部分以及两部分之间的远程链路。收缩部分包含步长为偶尔使用的 2 的卷积层，导致总的下采样因子为 64。这允许网络来估计大位移。网络的扩展部分接下来逐步和非线性地上采样特征图，同时考虑到来自收缩部分的特征。这是通过一系列上卷积层和卷积层实现。注意，网络中没有数据瓶颈，信息也可以通过收缩层和扩展层之间的远程链路。整体架构的说明我们可以参考论文 Dosovitskiy 等人^[4]的图。

对于视差估计，我们建议使用基本的 DispNet 架构，如表 2 所示。我们发现在扩展部分中的额外卷积产生比 FlowNet 架构更平滑的视差图；见图 6。

表 2: DispNet 架构的说明。收缩部分由卷积 conv1 至 conv6b 组成。在扩展部分，上卷积(upconvN)，卷积(iconvN, prN)和损失层是交替的。来自早期层的特征连接着更高层的特征。预测视差图由 pr1 输出。

Name	Kernel	Str.	Ch I/O	InpRes	OutRes	Input
conv1	7×7	2	6/64	768×384	384×192	Images
conv2	5×5	2	64/128	384×192	192×96	conv1
conv3a	5×5	2	128/256	192×96	96×48	conv2
conv3b	3×3	1	256/256	96×48	96×48	conv3a
conv4a	3×3	2	256/512	96×48	48×24	conv3b
conv4b	3×3	1	512/512	48×24	48×24	conv4a
conv5a	3×3	2	512/512	48×24	24×12	conv4b
conv5b	3×3	1	512/512	24×12	24×12	conv5a
conv6a	3×3	2	512/1024	24×12	12×6	conv5b
conv6b	3×3	1	1024/1024	12×6	12×6	conv6a
pr6+loss6	3×3	1	1024/1	12×6	12×6	conv6b
upconv5	4×4	2	1024/512	12×6	24×12	conv6b
iconv5	3×3	1	1025/512	24×12	24×12	upconv5+pr6+conv5b
pr5+loss5	3×3	1	512/1	24×12	24×12	iconv5
upconv4	4×4	2	512/256	24×12	48×24	iconv5
iconv4	3×3	1	769/256	48×24	48×24	upconv4+pr5+conv4b
pr4+loss4	3×3	1	256/1	48×24	48×24	iconv4
upconv3	4×4	2	256/128	48×24	96×48	iconv4
iconv3	3×3	1	385/128	96×48	96×48	upconv3+pr4+conv3b
pr3+loss3	3×3	1	128/1	96×48	96×48	iconv3
upconv2	4×4	2	128/64	96×48	192×96	iconv3
iconv2	3×3	1	193/64	192×96	192×96	upconv2+pr3+conv2
pr2+loss2	3×3	1	64/1	192×96	192×96	iconv2
upconv1	4×4	2	64/32	192×96	384×192	iconv2
iconv1	3×3	1	97/32	384×192	384×192	upconv1+pr2+conv1
pr1+loss1	3×3	1	32/1	384×192	384×192	iconv1

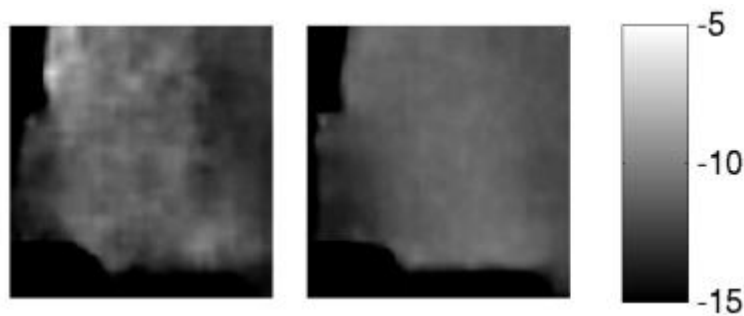


图 6: 上卷积之间没有（左图）和有（右图）额外卷积的预测视差图的特写。注意右图的预测更平滑。

我们还测试了一个利用显式相关层^[4]的架构，我们称之为 **DispNetCorr**。在这个网络中，两个图像单独处理，直到 **conv2** 层，然后将所得到的特征水平相关。我们认为最大位移为 40 像素，其对应于输入图像中的 160 个像素。与 Dosovitskiy 等人^[4]的 2D 相关相比，1D 相关在计算上更简洁并且允许我们用比 **FlowNet** 更精细的采样来覆盖更大的位移，对这个相关，我们使用步长为 2。

我们训练一个用于场景流估计的联合网络，它组合并且微调预训练的视差和光流网络。如图 5 所示。我们使用我们的 **FlowNet** 实现方法来预测左右图之间的光流和两个 **DispNets** 来预测 t 和 $t+1$ 图的视差。然后，我们微调这个大组合网来估计光流，视差和额外的视差变化。

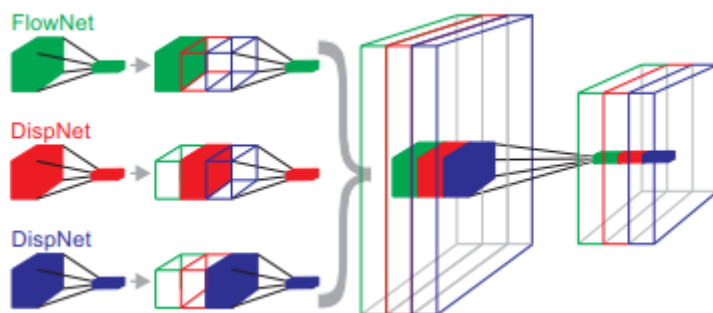


图 5: 交叉 **FlowNet**（绿色）和两个 **DispNets**（红色和蓝色）的权重到 **SceneFlowNet**。对于每一层，过滤器掩码是通过设置一个网络的权重为 1 并将其他网络的权重分别设为零（中间）。然后将来自每个网络的输出连接产生一个大网络，这个网络具有三倍的输入数和输出数（右）。

训练。所有网络都是端到端的训练，给定图像作为输入和地面实况（光流，视差，或场景流）作为输出。我们使用自定义版本的 Caffe^[13]和利用 Adam optimizer 优化器^[14]。我们设置 $\beta_1=0.9$ ， $\beta_2=0.999$ 如论文 Kingma 等人^[14]。同时我们使用的学习率 $\lambda=1e-4$ 并

且从迭代 400000 次开始的每 200000 次迭代学习率除以 2。

由于网络的深度以及收缩层和扩展层之间的直接连接（见表 2），如果所有的六个损失都是激活的，更低的层得到混合梯度。我们发现使用损失权重法是有用的：我们开始训练时，将损失权重 1 分配给最低分辨率损失 $loss_6$ ，将权重 0 设为所有其他损失（即所有其他损失均被关闭）。在训练期间，我们逐步增加具有更高分辨率的损失的权重，停用低分辨率的损失。这使得网络能够首先学习粗略表示然后继续更精细的分辨率的处理，没有损失限制中间特征。

数据扩增。尽管有大量的训练集，我们选择执行数据扩增以几乎没有额外代价的引入更多的种类到训练数据中。我们执行空间变换（旋转，平移，剪切，缩放）和色彩变换（颜色，对比度，亮度），我们对所有 2 或 4 输入的图像使用相同的变换。

对于视差，引入任何旋转或垂直偏移都会打破对极约束。水平移位会导致负视差或者移至相机的无穷远。

6. 实验

现有方法的评估。我们用我们的数据集评估了几个现有的视差和光流估计方法。对于差异，我们评估 Zbontar 和 LeCun 的最先进的方法^[28]和流行的半全局匹配^[11]方法与 OpenCV 的块匹配方法。结果与那些我们的 DispNets 一起展示在表 3 中。我们使用端点误差（EPE）作为大多数情况的误差测量，唯一的例外是 KITTI 2015 测试集只有 D1-all 误差是由 KITTI 的评估服务器提供的。它是估计误差比地面实况视差大 3px 和大 5% 的像素的百分比。

表 3: 视差误差。所有测量值都是端点误差，除了 KITTI-2015 测试的 D1-all 测量值（参见文本解释）。它的结果来自 KITTI 2012 训练的微调网络。

Method	KITTI 2012		KITTI 2015		Driving	FlyingThings3D test	Monkaa	Sintel Clean train	Time
	train	test	train	test (D1)					
DispNet	2.38	—	2.19	—	15.62	2.02	5.99	5.38	0.06s
DispNetCorr1D	1.75	—	1.59	—	16.12	1.68	5.78	5.66	0.06s
DispNet-K	1.77	—	(0.77)	—	19.67	7.14	14.09	21.29	0.06s
DispNetCorr1D-K	1.48	1.0 [†]	(0.68)	4.34%	20.40	7.46	14.93	21.88	0.06s
SGM	10.06	—	7.21	10.86%	40.19	8.70	20.16	19.62	1.1s
MC-CNN-fst	—	—	—	4.62%	19.58	4.09	6.71	11.94	0.8s
MC-CNN-acrt	—	0.9	—	3.89%	—	—	—	—	67s

DispNet。我们在 FlightThings3D 数据集上训练 DispNets 网络，然后在 KITTI 上对

其进行选择性的微调。微调后的网络在表中用“-K”后缀表示。DispNetCorr 是在 KITTI 2015 上微调的,是目前在 KITTI 2015 年前几名表中的第二名,略低于 MC-CNN-acrt^[28], MC-CNN-acrt 大约快 1000 倍。关于 KITTI 分辨率,它以每秒 15 帧的速度运行在 Nvidia GTX TitanX GPU 上。对于前景像素(属于汽车模型)它的误差大约是文献^[28]的一半。该网络误差比表中最好的实时方法 Multi-Block-Matching^[7]还要低 30%。在其他数据集中 DispNet 也表现良好,并且优于 SGM 和 MC-CNN 网络。

虽然 KITTI 的微调改善了在这个数据集中的结果,但这也会增加在其他数据集中的误差。我们认为这个显著的性能下降,是由于 KITTI 2015 中只包含相对较小的是视差,最高约 150 像素,而其他数据集包含的一些 500 像素及以上的视差。当在 KITTI 上进行微调时,网络似乎失去了预测大位移的能力,因此在其他数据集中导致了巨大的错误。

对比着 FlowNet^[4],我们对网络架构提出了几个修改。首先,我们在网络扩展部分的上卷积层之间的添加了一层卷积层。正如预期,这允许网络更好地调整视差图并预测出更平滑的结果,如在图 6 所示。在数量上,这导致在 KITTI 2015 中相对 EPE 评估大约降低了 15%。

第二,我们使用一个 1D 相关层训练了我们网络的一个版本。与 Dosovitskiy 等人^[4]相比,我们发现具有相关性的网络整体更好,(见表 3)。一个可能的解释是,1D 性质的视差估计问题允许我们在比 FlowNet 更细的网格上计算相关性。

SceneFlowNet。我们提供用卷积网络场景流估计的完整的早期结果。图 8 显示了一个 FlyingThings 场景的网络的结果。这个网络能够很好地预测视差变化,即使在被遮挡的区域。由于训练场景流时必须处理的数据量很大,网络训练相对较慢(一个前向传递的网络需要 0.28s,是 DispNet 上时间的 5 倍),而且还没有收敛。随着我们允许网络训练更长时间,我们希望结果进一步改善。表 4 是我们数据集的定量评估

表 4: 在上述数据集中评估我们的 SceneFlowNet 的端点误差。Driving 数据集包含最大的视差,光流和视差变化,导致了较大的误差。FlyingThings3D 数据集包含大光流,而 Monkaa 包含较小的光流和较大的视差。

SceneFlowNet	Driving	FlyingThings3D	Monkaa
Flow	22.01	13.45	7.68
Disparity	17.56	2.37	6.16
Disp. change	16.89	0.91	0.81

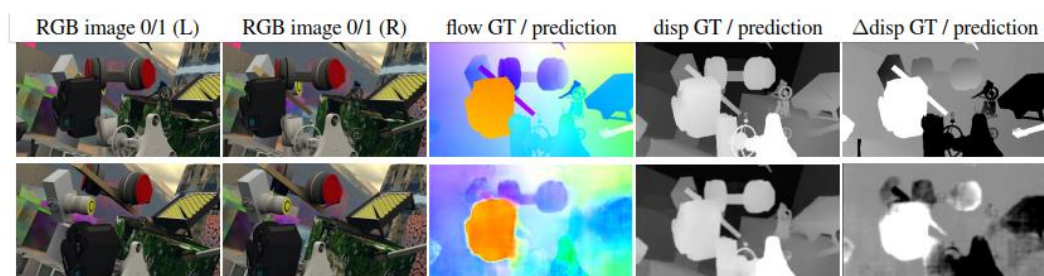


图 8: 预训练的 FlowNet 和 DispNets 产生的 SceneFlowNet 结果。已经加入视差变化并在 FlyingThings3D 的 80000 次迭代中进行了网络微调。视差变化的预测在经过这几次训练的迭代已经相当不错了。

7. 总结

我们已经介绍了一个包含超过 35000 个带地面实况视差, 光流和场景流的立体图像对的合成数据集。虽然我们的动机是创建一个足够大的适合于训练卷积网络估计的数据集, 但这个数据集也可以用于评价其他方法。特别是对场景流来说很有趣, 那里一直缺乏带地面实况信息的数据集。

我们已经证明, 这个数据集确实可以成功地训练大型卷积网络: 我们训练的视差估计的网络是和当前最先进的网络同等水平的, 当前最先进的网络比我们的运行速度快 1000 倍。率先使用标准的网络架构训练场景流估计网络的方法也显示出了不错的效果。我们相信我们的数据集将有助于提高深入学习研究中立体视觉, 光流和场景流估计这样具有挑战性的视觉任务。

8. 致谢

这项工作的一部分资金来自 ERC Starting Grant VideoLearn, the ERC Consolidator Grant 3D Reloaded, DFG Grants BR3815 / 7-1 和 CR 250 / 13-1。

参考文献

- [51]S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. Technical Report MSR-TR-2009-179, December 2009. 2
- [52]D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In ECCV, Part IV, LNCS 7577, pages 611–625, Oct. 2012. 2
- [53]J. Cech, J. Sanchez-Riera, and R. P. Horaud. Scene flow estimation by growing correspondence seeds. In CVPR, 2011.3
- [54]A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In ICCV, 2015. 1, 2, 3, 5, 7, 12
- [55]A. Dosovitskiy, J. T. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In CVPR, 2015. 3
- [56]D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. NIPS, 2014. 3
- [57]N. Einecke and J. Eggert. A multi-block-matching approach for stereo. In Intelligent Vehicles Symposium, pages 585– 592, 2015. 7
- [58]A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. International Journal of Robotics Research (IJRR), 2013. 2, 5
- [59]R. Hartley and A. Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003. 12
- [60]J. Hays and A. A. Efros. im2gps: estimating geographic information from a single image. In CVPR, 2008. 4
- [61]H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. PAMI, 30(2):328–341, 2008. 3, 6
- [62]F. Huguet and F. Deverney. A variational method for scene flow estimation from stereo sequences. In ICCV, 2007. 3
- [63]Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv preprint

arXiv:1408.5093, 2014. 5

- [64]D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In ICLR, 2015. 5
- [65]A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pages 1106–1114, 2012. 2
- [66]Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 2
- [67]M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 3, 5
- [68]P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgb-d images. In *ECCV*, 2012. 2
- [69]J. Quiroga, F. Devernay, and J. Crowley. Scene flow by tracking in intensity and depth data. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012 IEEE Computer Society Conference on, pages 50–57. IEEE, 2012. 3
- [70]M. Savva, A. X. Chang, and P. Hanrahan. Semantically-Enriched 3D Models for Common-sense Knowledge. *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality*, 2015. 4
- [71]D. Scharstein, H. Hirschmuller, Y. Kitajima, G. Krathwohl, N. Neřc, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition*, pages 31–42. Springer, 2014. 2
- [72]D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.2
- [73]S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475–480, 2005. 3
- [74]S. Vedula, S. Baker, P. Rander, R. T. Collins, and T. Kanade. Three-dimensional scene flow. In *ICCV*, pages 722–729, 1999. 3
- [75]C. Vogel, K. Schindler, and S. Roth. Piecewise rigid scene flow. In *ICCV*, 2013. 3
- [76]A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers. Efficient dense scene flow from sparse or dense stereo data. Springer, 2008. 3
- [77]J. Xiao, A. Owens, and A. Torralba. Sun3d: A database of big spaces reconstructed using

- sfm and object labels. In Computer Vision (ICCV), 2013 IEEE International Conference on, pages 1625–1632, Dec 2013. 2
- [78]J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. arXiv preprint arXiv:1510.05970, 2015. 3, 6, 7
- [79]K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. IEEE Trans. Circuits Syst. Video Techn., 19(7):1073–1079, 2009. 3

一个训练视差，光流，场景流估计卷积网络的大型数据集：补充材料

1. 介绍

由于本文篇幅的限制，补充材料包含了数据集产生过程更详细的描述（第 2 节），以及更多的细节和 DispNet 更多的定性结果（第 3 节）。



图 1: Driving 场景的鸟瞰图。相机沿着街道上的一条羊肠小道，并遇到很多转角，道口，其他车辆和不同的照明条件。

2. 数据集产生细节

我们修改 Blender 的的通道的内部渲染引擎来产生——除了立体 RGB 图像——每帧每视图三个额外的数据传递。图 2 给出了这个数据的可视明细：

- 在基础传递中 ($3DPos_t$)，每个像素存储现场点的真实 3D 位置，它投影那个像素（3D 位置在相机坐标系统中给出）。
- 对于第二次传递 ($3DPos_{t-1}$)，我们恢复时间到前一帧 $t-1$ ，并保存那个时候的所有顶点的 3D 位置。然后我们返回到当前帧 t 并使用时间 t 的顶点 3D 位置来投影时间 $t-1$ 的 3D 顶点到图像空间中。因此，我们再次存储每个像素的 3D 位置，但此时从时间 $t-1$ 的 3D 位置，使用在时间 t 的投影。
- 第三次传递 ($3DPos_{t+1}$) 类似第二次传递，但这次我们使用的后续的 $t+1$ 帧，而不是以前的 $t-1$ 帧。

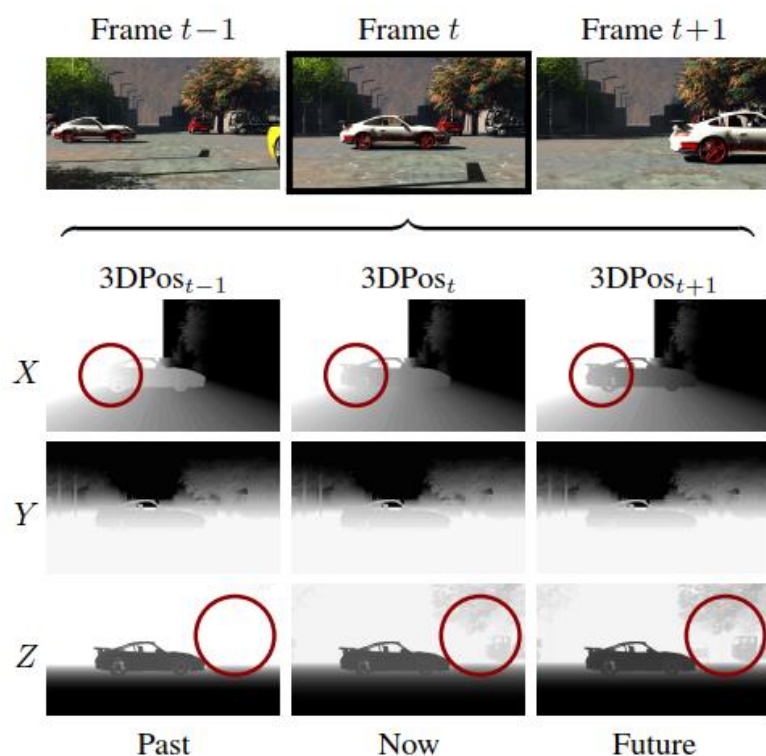


图 2: 我们对 t 帧的中间渲染数据：在 $X / Y / Z$ 通道编码所有帧 t 处视点（中间列）的 3D 位置（相对于相机），以及他们各自的 3D 位置是/将是在前一帧/下一帧（左/右列）。前一帧和下一帧的 3D 位置是存储在如在帧 t 相同的图像位置。因此，分析来自帧 t 的位置可以得到相应的 3D 点的过去，当前和未来的位置信息。所有场景流数据可以从该信息导出。例如：汽车向右移动改变其 X 值（注意，该透视投影压缩远处天空的强度梯度到一个位于 $X=0$ 的明显的步骤）。没有什么是垂直运动的，所以所有的 Y 值是随时间恒定的。相机向前移动，所有的 Z 值均匀改变（注意，右侧的对象是如何变得可见的）。

这三个数据结构包含有关的所有关于三维结构和从当前视点可见的场景的三维运动的信息。从 3DPos 数据中，我们生成场景流数据。图 3 描述了从 blender 输出到所得数据集的数据转换步骤。注意彩色图像和分割掩码是直接通过 Blender 产生的，并且不需要任何后期处理。与相机的内在和外在性质一起，各种数据可以产生出来，包括校准的 RGBD 图像。

图 4 展示了对我们数据集中一帧的分割掩码示例。材料可以穿过对象共享，但对对象索引和材料索引的组合产生一个场景的独特过度分割（整个场景中的所有帧一致的）。虽然我们的实验不利用这些数据，但对于其它应用，我们还在我们的数据集中包括对象和材质 ID。

与这份补充材料一起，我们还提供一个视频来演示我们创建的数据集和最终的输出管道，即光流，视差，视差改变，对象和材料索引地面实况。

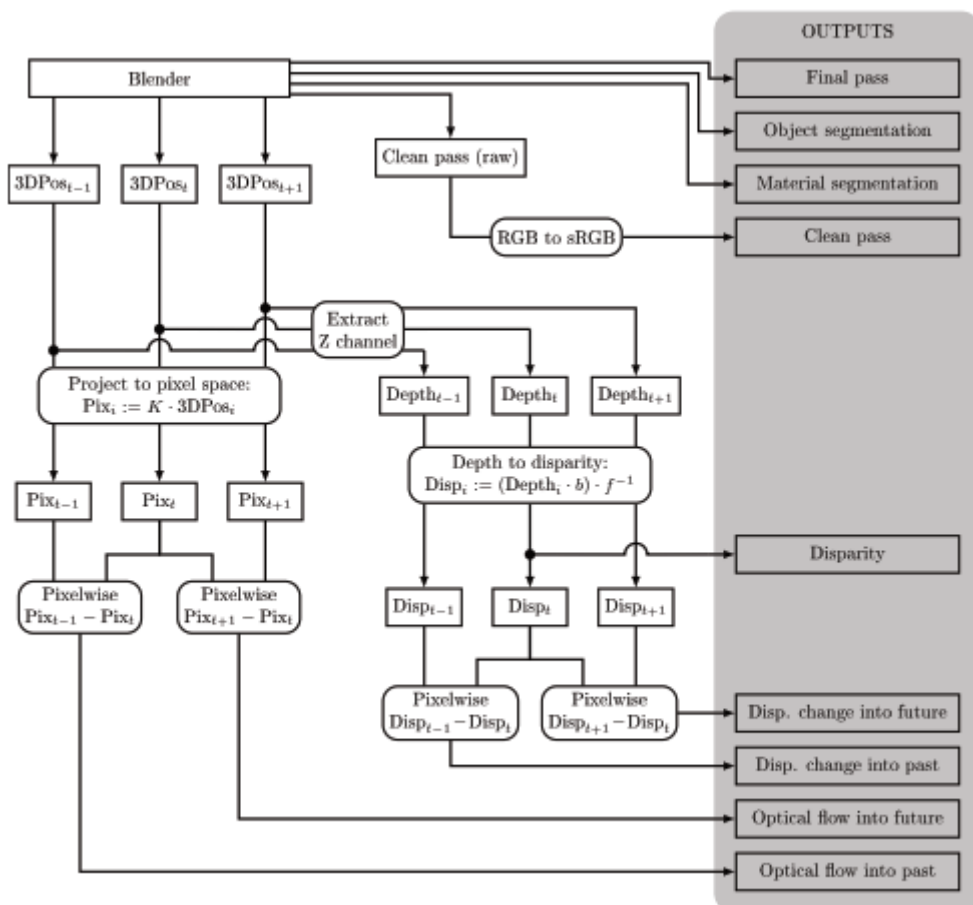


图 3: 在帧时间 t 的单一视图的数据生成概述: Blender 直接输出 Final pass 和 Clean pass 图像, 以及对象和材料的分割掩码。视差是直接从深度中获取, 深度是由图 2 中的当前 3DPos 图的 Z 通道给出的 (b 是立体视觉基线, f 表示焦距)。在未来/过去方向上的视差变化, 是未来/过去的视差图的结果减去当前视差图得到的。原始 3DPos 图像从相机空间使用相机内部矩阵 K 转换为像素空间的投影。从未来/过去像素位置图像减去当前的像素位置图像产生了未来/过去的光流。

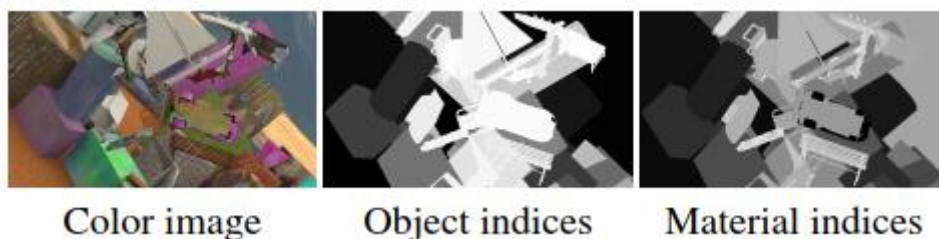


图 4: 分割数据: 对象索引是每个场景独一无二的。材料索引可以跨对象共享, 但是可以与对象索引组合以产生过度分割部分。

3. DispNetCorr

直观地看, 简单的 DispNet 视差估计架构 (如在主论文中所述) 必须学习在矫正立

体图像中匹配不同的图像部分的概念。由于这个问题的架构是众所周知的（对应只能根据对极几何^[9]找到），我们引入另一种可选的架构——DispNetCorr——我们在其中明确沿水平扫描线的相关特性。

虽然 DispNet 使用两个堆叠的 RGB 图像作为单输入（即一个六声道输入的对象），DispNetCorr 架构首先独立处理输入图像，那么关联两个图像之间的特征 并进一步处理结果。此行为类似于关联架构^[4]中使用的那样，其中 Dosovitskiy 等人用有限相邻大小和每张图不同的步长来构建一个二维相关层。对于视差估计，我们可以用一个没有步长而有较大的邻域大小的更简单的方法，因为沿着一个维度相关性计算上要求更少。我们通过限制仅搜索一个方向，可以额外降低比较的次数。例如，如果我们都给定左摄像机图像，在右图查找对应关系，那么所有的视差位移都是到左边的。

给定两个特征对象 a 和 b ，它们有多个信道和相同的大小，我们使用 D 通道计算相同的宽度和高度的关联图，其中 D 是可能的差异值的数量。对于在第一个对象 a 的位置 (x, y) 的一个像素，在通道 $d \in [0, D - 1]$ 的产生的关联入口是两个特征矢量 $a_{(x,y)}$ 和 $b_{(x-d,y)}$ 的数积。

4. 定性示例

我们的视差估计网络的定性评估图，并与其他方法的比较，都展示在图 5 到 10 中。

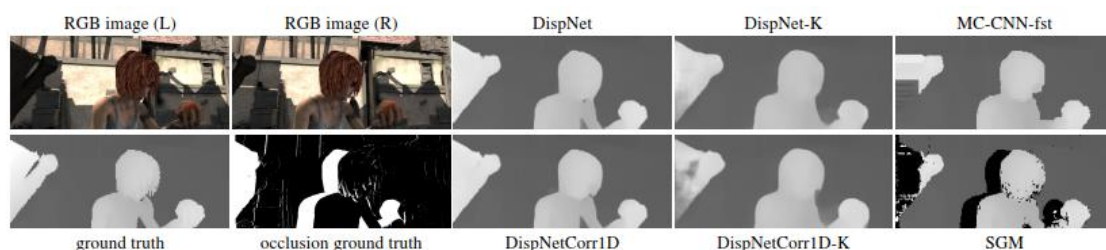


图 5： 一个 Sintel 帧的视差：DispNet 和 DispNetCorr1D 以更合理的方式填充遮挡区域，对比其他方法。

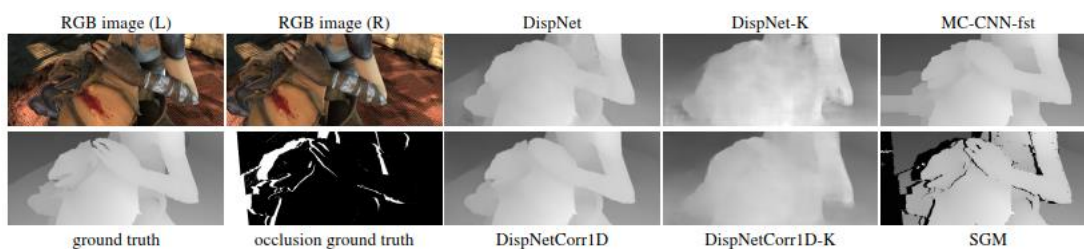


图 6: 一个 Sintel 帧视差: DispNetCorr1D 提供更清晰的估计和对龙首的平滑区域估计比 DispNet 更好。

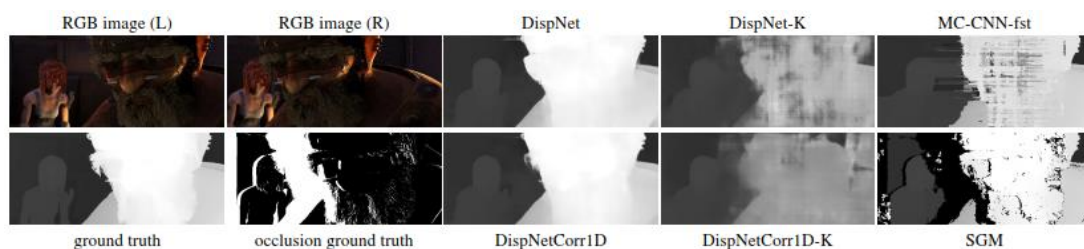


图 7: 一个 Sintel 帧视差: 微调网络在 KITTI 2015 数据集中无法估计的巨大视差了 (大视差不存在于 KITTI 中)。另外 MC-CNN-FST 也对大视差处理有问题。

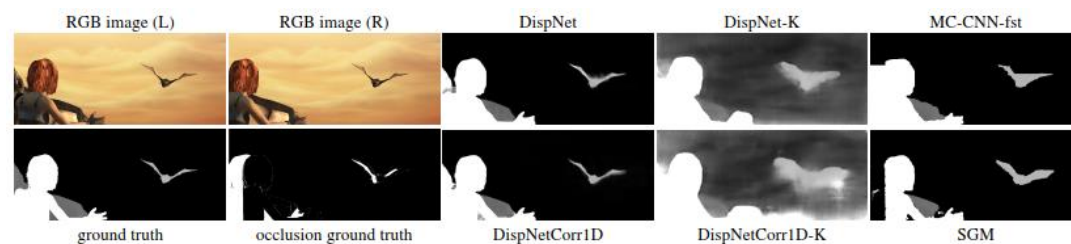


图 8: 一个 Sintel 帧视差: DispNet 和 DispNetCorr1D 可以用一个不错的方式处理遮挡区域。在 KITTI 2015 中微调后, 在天空区域失败了 (地面对天空真实数据和其他不在 KITTI 中的小视差)。

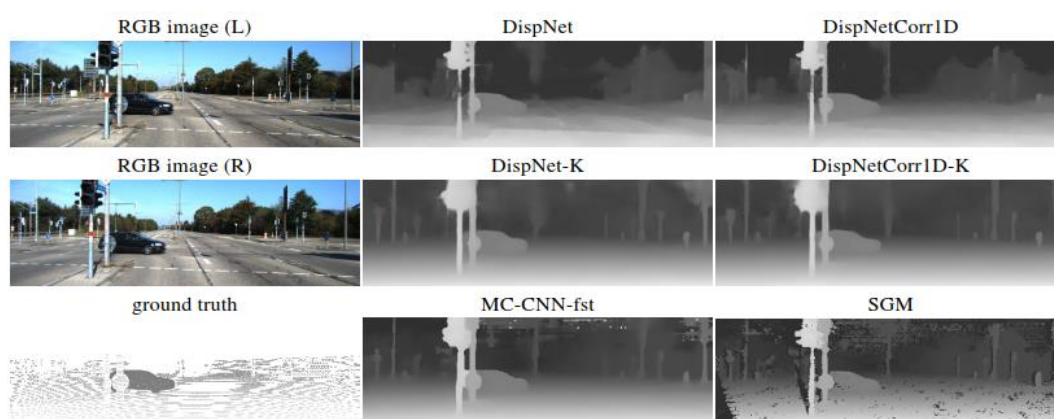


图 9: KITTI 2015 帧的视差: 当使用这样的地面实况信息微调时, KITTI 2015 数据集的稀疏性导致了非常平滑的预测。虽然未微调的 DispNet 和 DispNetCorr1D 准确地估计精细细节, 但它们对 KITTI 数据集中常见的平滑道路和地面区域估计不太准确。

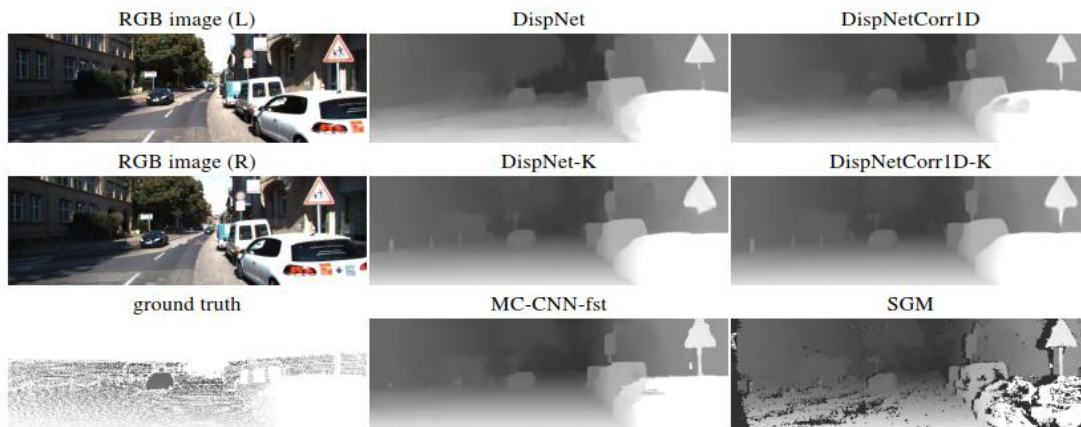


图 10: KITTI 2015 帧的视差: KITTI 的微调网络导致更平滑的估计。然而, DispNet-K 和 DispNetCorr1D-K 仍可以识别在左下角的标识, 这是 DispNet 和 DispNetCorr1D 完全忽略的。这表明微调网络不仅仅是过度平滑, 但仍能找到小的结构和视差不连续性。

三、开题报告

双目视觉立体匹配算法设计与实现

姓名 王灏 专业班级 通信工程 1301

1 课题研究背景及意义

近年来深度学习技术开始被广泛应用到各个领域，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。随着无人驾驶、机器人导航技术的发展，如何从二维的图像中提取出三维的深度信息成为了一个重要的命题。虽然自20世纪60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。准确率较高的传统的方法如激光测距仪，结构光等，受限于提取设备的繁琐复杂，无法得到广泛应用。而设备简单的传统双目视觉方法，受限于立体匹配算法（如SAD块匹配和SIFT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出准确率较高的立体视觉算法。

立体视觉是一种从二维平面图像中恢复出三维深度信息的技术。双目立体视觉系统是通过模拟人的双眼，仅需要两台安装在同一水平线上的数字摄像机，所得图像对经过立体矫正就可以投入使用，具有实现设备简单，成本低廉，并且可以在非接触条件下测量距离的优点。在机器人导航系统中双目立体视觉技术可以用于环境检测、障碍识别。在工业自动化控制系统中双目立体视觉技术可用于零部件的识别安装、产品质量检测。在安防监控系统中双目立体视觉技术常用于人流检测，危害报警。在无人驾驶技术中双目视觉技术还可用于道路环境的检测。

经典的双目立体视觉方法实现的一般步骤为：相机内参外参的离线标定，双目相机图像矫正，立体匹配以及光学三角形计算深度信息。通常采用不同水平位置的相机拍摄的两个图像，通过立体匹配找出对应点，计算视差 d ，视差指的是同一对象在左图像和右图像中的水平位置的差异——同一对象在左图像中的位置为 (x, y) ，在右图像中的位置为 $(x-d, y)$ 。已知视差，我们可以使用以下关系计算它的深度 z ：

$$z = \frac{fB}{d}$$

其中 f 是相机的焦距， B 是相机中心之间的距离。上述步骤中的立体匹配算法是双目视觉技术的核心问题，也是我们目前研究的重点。立体匹配的精度和速度对于立体视觉系统有着很大的影响。为提高双目视觉方法的性能，基于卷积神经网络进行立体匹配的双目视觉算法由下述几个部分组成。

(1) 训练模型

在KITTI 2012数据集^[1]中，从每个真实视差已知的图像位置处，提取出一对消极的和一对消极的 9×9 的训练图像对，构成二元分类数据集。并将等量的积极和消极图像对，作为输入数据，训练基于Tensorflow的卷积神经网络模型。

(2) 离线标定

通过常见的棋盘标定法，利用Matlab的stereo camera模块对双目摄像头进行标定。

(3) 数据获取

通过双目摄像头获取图像数据对。

(4) 双目矫正

利用Matlab标定所得的畸变参数和位置矩阵，在Opencv图像处理库中对双目摄像头所获取的左右图像对进行立体矫正，消除由于摄像头位置或摄像头本身性能导致的光学畸变。

(5) 立体匹配

将已矫正的左右图像对中的每个像素点，在可能的视差范围内，从左右图像对中依次提取出 9×9 的图像块，输入训练好的卷积神经网络模型，选取相似性得分最高的点，作为匹配点，并通过半全局匹配，左右一致性检查，中值滤波和双边滤波等后处理进一步优化视差图。

(6) 3D恢复

根据所得的视差图，在Opencv图像处理库中进行3D恢复，计算3D坐标。

2 国内外现状

计算机立体视觉理论开始于20世纪60年代，美国麻省理工学院的Robert最先提出把二维图像分析推广到三维景物分析，标志着计算机立体视觉技术的诞生，并逐步发展成

一门新的学科^[2]。特别是20世纪70年代末，Marr等创立的视觉计算理论对立体视觉的发展产生了巨大影响，现已形成了从图像获取到最终的景物可视表面重建的比较完整的体系^[2]。目前常见的立体视觉算法是双目立体视觉算法。立体视觉算法的核心是立体匹配算法。

Kong和Tao在2004年提出采用平方距离的总和（SAD）来计算初始匹配代价。他们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，以及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^[3]，并在2006年提出通过组合由计算归一化的不同的窗口大小和中心的互相关获得的预测^[4]。

Zhang和Seitz在2007年提出使用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]。Scharstein和Pal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]。Li和Huttenlocher在2008年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^[7]。

Haeusler等人于2013年使用了一种随机森林分类器来组合若干置信度度量的方式^[8]。同样的，Spyropoulos等人于2014年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立体视觉法的误差^[9]。

Zbontar和LeCun在2015年提出使用卷积神经网络进行立体匹配。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。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。^[10]

目前在主流的立体视觉评测库KITTI和Middlebury中，Zbontar和LeCun提出的MC-CNN-acrt架构准确率高达2.43%（排名第五），所以使用卷积神经网络进行立体匹配算法的设计非常适合。

卷积神经网络是深度学习模型的一种，目前众多的研究院校和企业（Google，Facebook，阿里巴巴，百度，腾讯）都致力于深度学习模型的研究、应用、推广和优化。谷歌著名的AlphaGo围棋程序正是基于两个神经网络的算法设计的。深度学习模型在图像匹配中表现优异，其基本思想是通过有监督或者无监督的方式学习层次化的特征表达，从而达到从底层到高层的描述。主流的深度学习模型包括自动编码器、受限玻尔兹曼机、

深度置信网络以及卷积神经网络等。下面详细介绍目前较为火热并强大的卷积神经网络，这也是在立体匹配算法中准备使用的技术。

1962年Hubel和Wiesel^[11]通过对猫视觉皮层细胞的研究，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。后来研究者便将这种逐步抽象的分层模型命名为深度学习模型。1984年日本学者Fukushima基于感受野概念提出的神经认知机(Neocognitron)^[12]可以看作是卷积神经网络的第一个实现网络。

1988年LeCun等人将BP神经网络算法引入CNN，LeCun等人结合BP算法实现的LeNet-5模型在数字识别领域的表现强大，在银行支票的手写体字符识别中，识别正确率达到商用级别^[13]。这是第一个真正多层结构的学习算法，它利用空间相对关系减少参数数目以提高训练性能。2006年，Hinton提出了深度置信网络(DBN)，一种深层网络模型。使用一种贪心无监督训练方法来解决训练问题并取得良好结果。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^[14]。Siamese网络通过在全连接层前添加两个共享权重的子网来得到两张图片的特征，并计算特征间的距离来比较图像的相似度，非常适合于双目视觉算法中的立体匹配^[15]。

3 研究内容及方法

3.1 研究内容

本篇文章的研究目标为设计一个双目视觉立体匹配算法。其主要内容包含如下：

- (1) 通过文献检索查找国内外分类领域的最新研究，阅读比较各个算法的优劣，并针对本篇文章的研究目标设计可行的网络模型和整体方案。
- (2) 从KITTI 2012已知视差的立体数据集中构建二元分类数据集（含数量相等的匹配的数据和不匹配的数据）。
- (3) 基于Tensorflow框架，设计并训练卷积神经网络模型。本文初步选定的模型为Siamese网络模型，如图3-1所示。
- (4) 双目摄像头采集标定图片，并在Matlab中进行双目标定。
- (5) 根据Matlab标定所得的相机参数进行利用Opencv对采集的图片进行双目矫正。

- (6) 将矫正后的图片上传到服务器中，利用训练好的卷积神经网络模型进行立体匹配和后处理。
- (7) 根据所得的视差图进行3D恢复，提取深度信息。

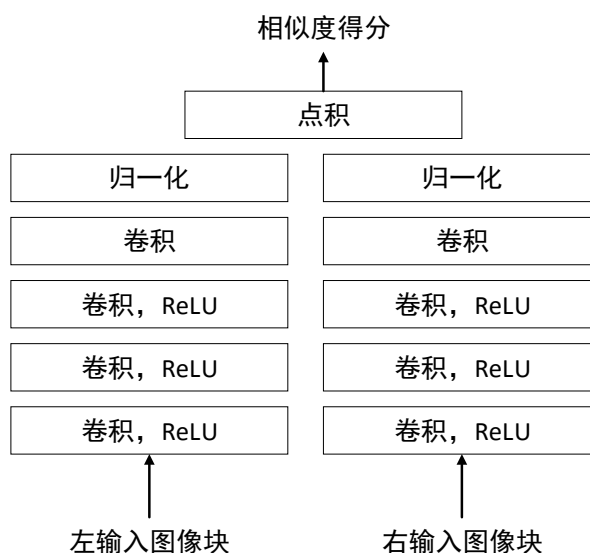


图3-1 Siamese网络模型

3.2 研究方法

- (1) 文献：利用各个文献检索数据库，互联网检索以及博客文章的阅读，查询相关领域的最新技术以及发展趋势。
- (2) 调研：通过与该领域公司内的技术人员交流沟通，了解目前基本解决方案以及最新的发展方向，并咨询目前该相关领域的难点和突破。
- (3) 实验：通过对初步方案的不断实验完善，对于模型的摸索，调整，通过实验数据优化模型并寻找改进的方案。

4 进度安排

2016.12.20--2017.3.1：收集相关资料文献，学习Tensorflow框架，从KITTI2012数据集中构建二元分类数据集，初步训练网络模型。完成外文翻译、文献综述；熟悉课题，有初步设计方案；

2017.3.1--2017.3.15 : 完成开题报告并有初步算法实现结果。

2017.3.16--2017.4.30 : 完成立体视觉算法的初始版本, 有较高的准确率。

2017.5.1--2017.5.31 : 算法的优化。撰写毕业论文初稿。

2017.6.1--2017.6.20 : 论文修改, 毕业答辩, 提交相关文档资料。

参考文献

- [1] 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.
- [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] Kong D, Tao H. A method for learning matching errors for stereo computation[C]//BMVC. 2004, 1: 2.
- [4] Kong D, Tao H. Stereo Matching via Learning Multiple Experts Behaviors[C]//BMVC. 2006, 1: 2.
- [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]//Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on. IEEE, 2007: 1-8.
- [7] Li Y, Huttenlocher D P. Learning for stereo vision using the structured support vector machine[C]. computer vision and pattern recognition, 2008: 1-8.
- [8] 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.
- [9] 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.
- [10] Zbontar J, Lecun Y. Computing the stereo matching cost with a convolutional neural network[C]. computer vision and pattern recognition, 2015: 1592-1599.
- [11] 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.
- [12] 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.
- [13] 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.

- [14]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.
- [15]Bromley J, Guyon I, Lecun Y, et al. Signature verification using a Siamese time delay
neural network[C]. neural information processing systems, 1993: 737-744.

四、指导教师评语

浙 江 工 业 大 学

毕业设计（论文）指导教师评语

专业班级	通信工程 1301	学生姓名	王灏
题 目	双目视觉立体匹配算法设计与实现		
<p>设计（论文）指导教师评语：</p> <p>王灏同学在整个毕业设计过程中，态度认真，工作踏实，独自钻研，勤奋刻苦。毕业设计的选题结合结合深度学习和双目视觉，是机器视觉领域的热点课题之一，具有较强的应用背景。所完成的工作主要有：</p> <p>(1) 设计了基于卷积神经网络的双目立体匹配算法；</p> <p>(2) 搭建了一套双目立体视觉系统；</p> <p>所完成的工作可实际应用，质量高。毕业设计说明书格式规范、逻辑性强、条理清晰、表述准确，是一份优秀的本科生毕业设计论文。</p> <p>建议成绩： 优</p> <p>指导教师： (签字)</p> <p>2017 年 6 月 3 日</p>			

注：此表一式一份，各学院（系）自行归档，保留三年。

五、论文评阅人评语

浙 江 工 业 大 学
毕业设计（论文）评阅人评语

专业班级	通信工程 1301	学生姓名	王灏
题 目	双目视觉立体匹配算法设计与实现		
设计（论文）评阅人评语：			
建议成绩：			
设计（论文）评阅人：			（签字）
2017 年 6 月 日			

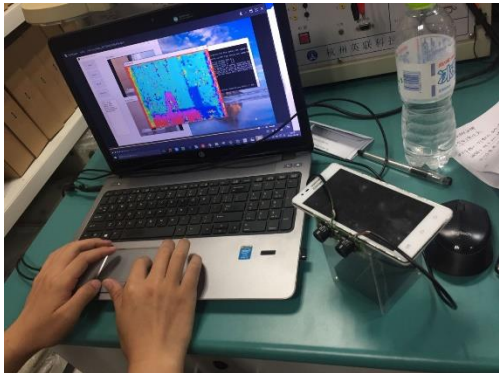
注：此表一式一份，各学院（系）自行归档，保留三年。

六、答辩记录

七、毕业设计成果演示记录表

浙 江 工 业 大 学

毕业设计（论文）成果演示记录表

专业班级	通信工程 1301	学生姓名	王灏	指导教师	宣琦
题 目	双目视觉立体匹配算法设计与实现				
根据设计任务书要求，毕业设计（论文）应完成的内容和应达到的功能	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。				
成果演示时间	2017.6.6	成果演示地点	广 C105		
实际演示结果	1.设计了一套双目立体视觉系统并实现三维场景恢复； 2.搭建了 Web 服务器和 Windows 客户端应用。 				

答辩小组长签名：

注：此表一式一份，各学院（系）自行归档，保留三年。

八、教师指导记录表

浙 江 工 业 大 学

毕业设计（论文）教师指导记录表

专业班级	通信工程 1301	学生姓名	王灏
题 目	双目视觉立体匹配算法设计与实现		
时间	2016.12.20-2016.12.27		
指 导 内 容 记 录	讲解课题内容，说明这次毕设的难度； 布置这一阶段要做的事情；		
时间	2016.12.28-2017.1.3		
指 导 内 容 记 录	讲解文献检索的相关内容，说明查找资料的大致方向；		
时间	2017.1.4-2017.1.10		
指 导 内 容 记 录	确定所要翻译的两篇外文文献； 指导学生收集相关资料文献；		
时间	2017.1.11-2017.1.17		

指导 内容 记录	<p>指导学生在寒假期间学习相关编程知识；</p> <p>督促学生完成外文翻译、文献综述，熟悉课题，并做好开题准备</p>
时间	2017.2.20-2017.2.27
指导 内容 记录	<p>讲解开题报告的相关内容，查看外文翻译和文献综述；</p> <p>督促学生完成开题报告</p>
时间	2017.2.28-2017.3.6
指导 内容 记录	<p>开题交流，检查文献综述，外文翻译和开题报告，指出要修改的地方；</p>
时间	2017.3.7-2017.3.13
指导 内容 记录	<p>指导确定课题的实施方案；</p> <p>督促学生完成数据采集</p>
时间	2017.3.14-2017.3.20

指导内容记录	指导学生使用 TensorFlow 搭建卷积神经网络，加深学生对 CNN 的理解；
时间	2017.3.21-2017.3.27
指导内容记录	指导学生训练卷积神经网络，调试参数；
时间	2017.3.28-2017.4.3
指导内容记录	指导学生实现后处理算法；
时间	2017.4.4-2017.4.10
指导内容记录	指导学生实现整套双目视觉立体匹配算法；
时间	2017.4.11-2017.4.17

指 导 内 容 记 录	指导学生搭建 Web 服务器，设计 UI 界面；
时间	2017.4.18-2017.4.24
指 导 内 容 记 录	督促学生接受中期检查，提出不足；
时间	2017.4.25-2017.5.1
指 导 内 容 记 录	指导学生整合双目立体视觉系统的各个模块；
时间	2017.5.2-2017.5.8
指 导 内 容 记 录	指导学生对立体匹配算法进行测试并改进；
时间	2017.5.9-2017.5.15

指 导 内 容 记 录	督促学生完成整套双目立体视觉系统；
时间	2017.5.16-2017.5.22
指 导 内 容 记 录	指导如何撰写毕业论文； 督促学生完成毕设初稿；
时间	2017.5.23-2017.5.30
指 导 内 容 记 录	对毕业论文初稿提出修改意见；
时间	2017.5.31-2017.6.3
指 导 内 容 记 录	完成毕业论文评阅，写好“指导教师评语”；
时间	2017.6.4-2017.6.5

指 导 内 容 记 录	指导学生制作 PPT 文稿，对答辩注意事项进行指导；
----------------------------	----------------------------

指导教师签名：

注：根据教师实际指导情况，添加相应栏目。

此表一式一份，各学院（系）自行归档，保留三年。

九、毕业设计进程考核表

通信工程专业_2013_届毕业设计（论文）进程安排与考核表

班级	通信工程 1301	学生姓名	王灏	总进程	2016 年 12 月—2017 年 6 月 总计 20 周	
毕业设计题目	双目视觉立体匹配算法设计与实现					
安排与考核						
起 止 时 间	阶 段 任 务 要 点			完 成 情 况	*阶段成绩	备 注
2016.12.20~2016.12.27	熟悉课题内容			全面完成	A	
2016.12.28-2017.1.3	进行相关文献检索			全面完成	A	
2017.1.4-2017.1.10	翻译外文文献，储备相关知识			全面完成	A	
2017.1.11-2017.1.17	撰写文献综述			全面完成	A	
2017.2.20-2017.2.27	撰写开题报告			全面完成	A	
2017.2.28-2017.3.6	修改文献综述和开题报告			全面完成	A	
2017.3.7-2017.3.13	完成三大件的修改，并规范好格式。			全面完成	A	
2017.3.14-2017.3.20	搭建卷积神经网络			全面完成	A	
2017.3.21-2017.3.27	训练卷积神经网络			全面完成	A	
2017.3.28-2017.4.3	完成后处理算法			全面完成	A	
2017.4.4-2017.4.10	实现立体匹配算法			全面完成	A	
2017.4.11-2017.4.17	完成 Web 服务器搭建			全面完成	A	
2017.4.18-2017.4.24	接受中期检查，完善不足			全面完成	A	
2017.4.25-2017.5.8	改进立体匹配算法			全面完成	A	
2017.5.9-2017.5.15	完成双目立体视觉系统			全面完成	A	
2017.5.16-2017.5.27	撰写毕业论文			全面完成	A	
2017.5.28-2017.6.3	根据指导老师意见完成终稿			全面完成	A	
2017.6.4-2017.6.5	准备答辩资料和 PPT			全面完成	A	

*注：阶段成绩分 A、B、C 三级：A 为全面完成任务、B 为完成任务、C 为完成任务不好

指导教师_____

2017 年 6 月 6 日

十、毕业论文的“知网”检测结果