## 基于匹配代价卷融合和多步插值策略的立体匹配

## 1.摘要

针对双目立体匹配算法中的自适应窗口法在重复纹理区表现不好的问题提出了一种基于匹配代价卷融合的策略；在后处理步骤中，提出一种基于分割的插值策略，提高了图片左右边界处的匹配精度，并且提出一种联合三种插值策略的方法，比已有的双插值策略匹配精度更高。与其他经典立体匹配算法的对比证明了本方法在计算视差精度上的优势。

## 引言

立体匹配通过搜寻左右图片对的对应像素点来计算视差，根据三角相似性原理，从视差值可求出物体到相机的距离。立体匹配算法被广泛应用于自动驾驶，智能机器人，Slam等领域中，是计算机视觉领域的研究热点[参考文献]。根据\*\*的分类，立体匹配算法被分为全局立体匹配和局部立体匹配[ ]。全局算法将视差计算表述为一个能量最小化的问题，可以通过置信度传播、图割等算法来求解，匹配精度高，但计算耗时；局部匹配算法通过为目标像素点构建局部支持窗口，利用邻域信息来降低匹配的模糊性，具有实现简单，运行速度块的优点，一些局部算法的精度可以媲美全局算法，使得局部匹配算法广泛地应用于现实任务，尤其是实时任务中。局部立体匹配流程通常被分为四个步骤：匹配代价计算、代价聚合、视差计算（也称为视差优化或视差细化）。

匹配代价用来表征两个像素点之间的相似性，根据相似性测量策略不同，匹配代价函数可分为两类：基于相关性的和基于差异的。归一化互相关NCC[ ]、零均值归一化互相关ZNCC[ ]是基于相关性的典型方法，实质上都是通过计算两个像素点所在的支持窗口内部像素点强度值的皮尔逊相关系数来比较两点间的相似性，对图像强度偏差和对比度改变有较高的鲁棒性、但是计算耗时；基于差异的方法是通过计算两个像素间的颜色或者灰度的欧氏距离来测量差异，有AD法[ ]、SD法[ ]等，这类方法计算简单，对丰富纹理或重复纹理区的表征较好，相反在弱纹理区表现不好，且对光照变化敏感；非参数变换法不直接计算像素点间的强度差异，而是先通过比较中心点与周围点强度值的大小来获取两个像素点的描述符，然后计算两个描述符的差异来得到两个像素点的差异，Census变换[ ]是其中应用最广的方法，它对弱纹理、噪声和光照的鲁棒性较高，但是由于其丢失了灰度信息，在重复纹理区表现不好[ ]。还有很多种匹配代价相结合的方法，其中Mei等人提出的AD-Census匹配代价函数[ ]，融合了AD和Census的优势，在复杂纹理和弱纹理区都有较好的表现。

代价聚合利用局部支持区域的信息来降低匹配的歧义性，可以把它看作对匹配代价卷的每一个视差层的滤波操作。局部支持区域的构建是该步骤的关键，用覆盖目标点的窗口来表示该支持区域，该方法的一个假设是窗口内的视差必须满足平滑性限制，当窗口覆盖两个或者多个视差平面时，聚合操作就会带来误差，引起前景区域的扩展或者缩小【多窗口法2002】，所以窗口不能太大以降低其包含不同视差区域的可能性；另一方面，在重复纹理或者复杂纹理区，窗口必须足够大才能保证匹配的准确性。许多学者对于窗口的构建策略进行了研究。

【多窗口法】【移动窗口法】等。

Zhang等人[ ]提出的基于十字交叉的自适应支持窗口构建策略，该算法通过颜色和距离限制构建目标像素点的水平垂直臂进而构建整个支持窗口，对视差边缘有较好的检测能力。Mei等人对此进行了改进，根据臂长的颜色限制对距离进行分段，在增大支持窗口面积的同时尽可能降低误差(邻域限制原方法有吗？) [ ]。但是，基于十字交叉的自适应窗口法的一个重要缺陷是：它在纹理重复区生成的窗口过小，这是由于该区域颜色的频繁变化造成的。

视差细化是对生成的初始视差图进行图像处理，比如平滑、插值、亚像素增强等。其中，插值策略是为标记为不合理的点选取周围合理的点来填充，它对于遮挡区的视差估计具有重要意义，同时也能降低非遮挡区的误匹配率。最常用的插值策略是背景插值[ ]，通过选择不合理点左右两边第一个合理点中视差大的值来填充不合理点，其基本思想是：视差值大的点更可能属于背景区，而遮挡点往往位于背景区。该方法简单高效，适用于对遮挡点的插值。区域投票插值[ ]先计算不合理点的同质区域内视差值的直方图，然后选择直方图中最高的视差值作为填充值，该方法鲁棒性强，正确率高，但是直方图的面积必须高于一定的阈值才能保证插值的可信性，遮挡区的点的同质区可能出现大部分位于遮挡区的情况，这就造成该点的直方图面积过小而策略失败。合理插值[ ]是背景插值的改进，它在不合理点周围的16个方向寻找合理视差来完成插值，但因为方向的增加以及每个方向搜索深度的加大，搜索噪声也增加，导致实际插值的正确率要低于背景插值。（有没有针对边缘问题的解决方法，要介绍一下）

局部立体匹配中除代价函数外，代价聚合策略在寻找最佳匹配视差时也起到至关重要的作用，对重复纹理仍缺乏有效的解决方案；在视差细化方面不同方法都有各自局限性，对于边缘插值的效果仍待提高。

针对这些问题，本文提出一种融合十字交叉自适应窗口和固定窗口的代价聚合策略，比单独使用自适应十字交叉窗口在重复纹理区表现好，并且提出一种联合背景插值和区域投票插值的策略，比单独使用区域投票得到了更低的误匹配率。此外，为提高在图像边缘的匹配效果，提出一种基于分割的插值策略。

## 2.本文算法

本文提出的立体匹配方法流程如图1所示。代价函数以性能较优的AD-Census为基础，主要在代价聚合、视差细化方面提出更有效的算法。

### 2.1 AD-Census代价函数

AD-Census代价函数是Mei【】提出，公式如下：

AD-Census公式

### 2.2 基于十字交叉窗口和固定窗口融合的代价聚合

对于每个图像像素点，需要从局部支持的区域中根据对初始代价卷上聚合的代价来选择最优匹配视差。这一步骤的关键在于如何确定局部支持区域的窗口。最简单的方法是矩形窗口法，它在以目标像素点为中心的大小固定的矩形窗口区内进行代价聚合计算。当该窗口包含的区域是视差连续的像素时，该方法能够取得较好的聚合效果，但是当窗口中存在视差跳跃现象，即窗口包含两个或多个值不同的视差面，聚合操作会引入误差，降低匹配精度。针对这一问题，十字交叉窗口法通过颜色和距离限制来构建不规则的目标像素点的局部支持窗口，使窗口的大小和形状能够尽可能地贴合视差不连续边缘。但是该方法在重复纹理区表现不好，原因是致密的纹理会阻断臂长的生长，导致支持区域过小，增大了匹配的模糊性。

本文考虑了两类方法优缺点，提出了十字交叉窗口和固定窗口融合的代价聚合策略，能够有效降低重复纹理区域的匹配误差，并且避免矩形窗口在视差不连续区域的引入误差。具体方法如下。

矩形窗口的构建非常简单，以。。。wxh

本文使用了Mei等人提出。。方法构建目标像素的十字交叉窗口。

首先计算像素点的上、下、左、右四个方向的支持臂的长度，公式如下：

颜色限制公式

距离限制公式

其中。。表示 。。。；。。表示 。。

再以同样的方式计算水平臂（左臂和右臂）或者竖直臂（上臂和下臂）上的每个像素点的垂直臂长或者水平臂长，完成支持区域的构建，整个过程如图1所示。

融合策略的基本思想是在重复纹理区使用矩形窗口聚合代价，而在其他区域使用十字交叉窗口聚合代价。由十字交叉窗口构建的方法可知，在重复纹理区该窗口表现为较小的区域，而在视差变化区域，其窗口的单方向臂长可能会较长，因此可要通过臂长来确定是否使用矩形窗口。此外，还考虑以总的匹配代价来限制矩形窗口的使用。

计算伪代码如下：

1. 分别计算矩形窗口聚合代价和十字交叉窗口聚合代价
2. 统计十字交叉窗口的最大臂长长度, 左右和上下总臂长长度
3. 如果，使用计算聚合代价；否则使用。

其中，和为长度阈值，其值不应超过矩形窗口的最长和最短边长。

### 2.3 针对遮挡区域的视差细化策略

遮挡区是只在一侧图片存在而在另一侧图片不存在的区域，根据形成原因的不同，其可以分为两类，第一类是由于背景物体被前景物体挡住，使得一侧相机可以拍到而另一侧相机无法拍到，这类区域一般在图片的内部，第二类是由于相机所处位置的不同，使得左侧相机拍摄到的偏左的区域，而右侧相机拍摄到偏右的区域，这就造成在左图的左侧和右图的右侧各存在一块区域在另一侧图片上不存在匹配点。

针对第一类遮挡区的处理，一般是通过插值的策略，常见的有背景插值，基于区域投票的插值，合理插值等。背景插值操作简单而有效，它从不合理点向左右两侧分别寻找第一个合理点，并选择代表来自背景点，即视差值大的点的视差来填充不合理点，公式如下：

背景插值公式：

区域投票策略，首先，构建不合理点的支持窗口，该支持窗口必须尽可能只包含相同或者连续视差的点，一般通过自适应窗口法来构建，然后，通过计算支持窗口内的所有点的视差值的投票直方图，并选择投票最多的视差作为候选插值的值，公式如下

区域投票插值公式：

若满足以下两个限制则进行插值，（1）构建的支持窗口内的合理点的数目必须超过设定的阈值，（2）投票最高的视差值的得票率必须高于设定的阈值。但该算法有个问题，遮挡区域的点构建的支持窗口部分位于遮挡区，相对非遮挡区，其更可能违反限制条件(1)，其在遮挡区域的插值成功率不如背景插值高。

背景插值在非遮挡区的插值精度低于区域投票，并且对比合理插值策略其只在水平方向寻找合理视差，在规定范围内若没有找到则插值失败。合理插值通过寻找不合理点16个方向上的合理视差值，并选择其中视差最小的点作为遮挡点的填充值，这个策略保证了插值的完成率，即通过增加方向数以及加大搜索深度增大了找到填充值的概率，但同时也因为放宽搜索范围增加了错误插值的可能性。所以这步插值应该在执行完区域投票或者背景插值后再执行。

针对第二类遮挡区域的插值，一般是沿用第一类遮挡区域的插值策略，但是第二类遮挡区域一般面积更大，对于靠近边缘处的遮挡点，需要更大的搜索深度，增加了错误插值的可能。（待续。。。。。）

整个算法的流程图：

## 4.实验结果与分析

实验平台配置为Intel(R) Core(TM) i5-7300U CPU，2.60GHz，8GB内存。利用visual studio 2019IDEA，使用C++语言编程实现算法。数据集选用Middleburry基准测试平台的第二版的数据集上，包括Teddy、Cones、Venus、Tsukuba四对图片，并设置视差搜索范围分别为0~59、0~59、0~19、0~15px。

评价标准和Middleburry平台保持一致，即误匹配像素百分比，并且设置误差限为1个像素，当某个像素点的预测视差值和真实视差值的差距大于1时，该点被识别为误匹配像素，分别计算在非遮挡区、所有区域和视差不连续区的误匹配像素百分比，区域的选择通过掩膜图来实现，掩膜图为二值灰度图，分别为0和255，其中255表示需要计算的点，而0表示不计算的点。

4.1 匹配代价卷融合策略的有效性

为了更好地观察匹配代价卷融合策略的效果，实验只保留匹配代价计算、代价聚合步骤，通过WTA获取视差图，不进行后处理步骤。设置参数，总臂长长度，单臂长长度。在代价聚合步骤中，分别使用基于十字交叉窗口的代价聚合和本文提出的代价卷融合策略，结果如表6 所示。

不同代价聚合方法的误匹配百分比

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Aggregation method | Tsukuba | | | Venus | | | Teddy | | | Cones | | | Ave |
| no-occ | all | disc | no-occ | all | disc | no-occ | all | disc | no-occ | all | disc |
| CBCA |  |  |  | 0.62 | 2.13 | 5.10 | 6.26 | 15.83 | 16.08 | 4.59 | 15.20 | 11.76 |  |
| Proposed |  |  |  | 0.58 | 2.05 | 4.63 | 6.25 | 15.79 | 15.99 | 4.53 | 15.06 | 11.61 |  |

结果如下，

两个方法的精度表

结果图

时间对比

对结果进行讨论

4.2 针对特定区域插值策略的有效性

4.2.1 背景插值和基于区域投票的联合插值策略的有效性

对比联合插值策略和区域投票的结果

4.2.2 基于图像分割的图片左边缘插值的有效性

在上一个步骤后，跟这个步骤，对比两者的结果

4.3 总的结果的对比

对比该方法和其他经典方法的结果，数据的和图片的

## 5.结论