在图形硬件上建立精确的立体匹配系统

Xing Mei[1,2], Xun Sun[1], Mingcai Zhou[1], Shaohui Jiao[1], Haitao Wang[1], Xiaopeng Zhang[2]

[1]三星先进技术研究所中国实验
[2]中国科学院自动化研究所

{xing.mei,xunshine.sun,mingcai.zhou,sh.jiao,ht.wang}@samsung.com,xpzhang@nlpr.ia.ac.cn

Abstract

This paper presents a GPU-based stereo matching system with good performance in both accuracy and speed. The matching cost volume is initialized with an AD-Census measure, aggregated in dynamic cross-based regions, and updated in a scanline optimization framework to produce the disparity results. Various errors in the disparity results are effectively handled in a multi-step refinement process. Each stage of the system is designed with parallelism considerations such that the computations can be accelerated with CUDA implementations. Experimental results demonstrate the accuracy and the efficiency of the system: currently it is the top performer in the Middlebury benchmark, and the results are achieved on GPU within 0.1 *seconds. We also provide extra examples on stereo video sequences and discuss the limitations of the system.*

1. Introduction

立体匹配是计算机视觉中研究最广泛的问题之一[11]。立体匹配算法设计中的两个主要问题是匹配精度和处理效率。尽管每年都会引入许多算法，但这两个问题在报告的结果中往往是相互矛盾的：准确的立体方法通常耗费时间[6,17,20]，而基于GPU的方法实现了高处理速度和相对较低的视差精度[10,18,24]。据我们所知，Middlebury排名前十的算法大多需要至少10秒才能处理384×288图像对，而前20名中仅有的两种基于GPU的方法:CostFilter[9]和Plane-FitBP[19]都是近乎实时的。

这种矛盾背后的原因很简单：精确立体算法采用的一些关键技术不适合GPU实现。对领先的Middlebury算法[6,17,20]的分析表明，这些算法在匹配过程中有几种常用技术：they use large support windows for robust cost aggregation [5, 16, 21]; 他们将视差计算步骤表示为能量最小化问题，并用慢收敛优化器求解[14];他们广泛使用图像区域分割作为匹配单元[17]，表面约束[6,20]或后处理[2]。这些技术以计算成本为代价显着提高了匹配质量。然而，直接将这些技术移植到GPU或其他多核平台上是棘手且麻烦的[4,7,18,19]：大型聚合窗口需要在每个像素上进行大量迭代;一些优化、分割和后处理方法需要复杂的数据结构和顺序处理。因此，简单的技术对于GPU和嵌入式立体匹配系统来说更为流行。设计一个在精度和效率之间取得良好平衡的立体匹配系统仍然是一个具有挑战性的问题。

在本文中，我们的目标是通过提供具有近实时性能的精确立体匹配系统来应对这一挑战。目前（2011年8月），我们的系统是Middlebury基准测试中表现最佳的系统。简而言之，我们将几种技术集成到一个有效的立体框架中。这些技术可确保高匹配质量，而无需高开销的分割和聚合。此外，它们显示出适度的并行性，因此整个系统可以映射到GPU上以进行计算加速。我们系统的关键技术包括：

- AD-census代价测度有效地结合了绝对差异（AD）测度和census变换。与具有robust的聚合方法的常见单一测度相比，此测度提供了更准确的匹配结果。在最近的立体算法[13]中采用了类似的测度方法。
- 基于cross区域实现高效的代价聚合。Zhang[23]等人首先提出了基于cross skeletons的Support区域。允许快速聚合middle-ranking的视差结果。我们通过更准确的区域构建和成本聚合策略来增强此技术。
- 基于Hirschmüller的半全局匹配（SGM）的扫描线优化器，减少了路径方向
- 一系列系统的改进，通过迭代区域投票，插值，深度不连续调整和亚像素增强来处理各种视差误差。这种多步骤过程证明对改善视差结果非常有效。
- 使用CUDA在GPU上实现了高效的系统.

2. 算法

遵循Scharstein和Szeliski的分类法[11]，我们的系统包括四个步骤：代价初始化、代价聚合、视差计算和后处理。我们提供了这些步骤的详细说明。

2.1. AD-Census 代价初始化

此步骤计算初始的代价值。由于计算可以在每个像素和每个视差级别上同时进行，因此该步骤本质上是并行的。我们主要关注的是开发一种高匹配质量的代价测度。常用的代价测度包括绝对差(ad)、Birchfield和Tomasi的抽样不敏感度(bt)、基于梯度的测度和非参数变换，如秩和中心值[22]。在　　　　和Scharstein[3]最近的评估中，census ﺲ局部和全局立体匹配方法中表现最佳。虽然将代价测度结合起来以提高准确性的想法似乎有了新的进展，但对这一问题的探讨相对较少。Klaus等人[6]建议将SAD和基于梯度的测度线性结合起来进行代价计算。他们的视差结果令人印象深刻，但他们没有明确阐述这种组合的好处。

Census使用除强度值本身之外的像素强度的相对或相关来编码局部图像结构，因此容忍由于辐射差异和图像噪声引起的异常值。然而，该方法还可能在具有重复或类似局部结构的图像区域中引入匹配的模糊。为了处理这个问题，应该加入更多细节信息。对于具有相似局部结构的图像区域，颜色（或强度）信息可能有助于减轻匹配的模糊性;而对于具有相似颜色分布的区域，Census变换比基于像素的强度差异更稳定。这正式组合测度思想的由来。

给定左图中的像素 $\mathbf{p} = (x, y)$ 和视差 d,计算两个独立的代价值$C_{census}(\mathbf{p}, d)$ 和 $C_{AD}(\mathbf{p}, d)$.

对于 C_{census}, 使用一个 9×7的窗口把每个像素的局部结构编码到一个64-bit的string中. $C_{Census}(\mathbf{p}, d)$ 定义为左图像素 $\mathbf{p}$ 和它对应的右图像素$\mathbf{pd} = (x - d, y)$ 编码生成的为串的海明距离[22].

C_{AD}被定义为RGB通道中p和pd的平均强度差：

$$C_{AD}(\mathbf{p}, d) = \frac{1}{3} \sum_{i=R,G,B} |I_i^{Left}(\mathbf{p}) - I_i^{Right}(\mathbf{pd})| \quad (1)$$

AD-Census代价$C(\mathbf{p}, d)$的计算方法如下:

$$\begin{aligned} C(\mathbf{p}, d) = \rho(C_{census}(\mathbf{p}, d), \lambda_{census}) + \\ \rho(C_{AD}(\mathbf{p}, d), \lambda_{AD}) \end{aligned} \quad (2)$$

其中 $\rho(c, \lambda)$ 是关于c的函数:

$$\rho(c, \lambda) = 1 - \exp(-\frac{c}{\lambda}) \quad (3)$$

该函数的目的有两个：首先，它将代价值到[0,1]的范围，使得等式（2）不会受到其中一个代价函数的严重影像(归一化);第二，它提供了一个参数 λ 用于控制离群点的影响。

为了验证组合的效果，图1中显示了在Middlebury数据集上使用AD、Census和AD-Census的一些视差结果特写。这些实验使用了Cross-based代价聚合。Census在具有重复局部结构的区域中产生错误匹配，而基于像素的AD不能处理大的无纹理区域。综合二者的AD-Census成功地减少了使用单独的代价函数引起的误差。对于定量比较，AD-Census将Census的非遮挡误差分别降低1.96%（Tsukuba），0.4%(Venus)，1.36%（Ted-dy）和1.52%（Cones）。而这种改进来自于额外添加的AD代价函数。

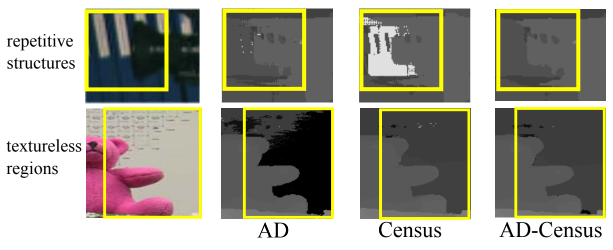

Figure 1. Some close-up disparity results on Tsukuba and Teddy image pair, which are computed with AD, Census, AD-Census cost measures and cross-based aggregation. AD-Census measure produces proper disparity results for both repetitive structures and textureless regions.

2.2. Cross-Based代价聚合

此步骤聚合每个像素在支撑区域(support region)上的匹配代价，以减少初始代价中的匹配模糊度和噪声。一个简单但有效的聚合假设是具有相似颜色的相邻像素应该具有相似的视差。这种假设已被最近的聚合方法所采用，例如分割支持(segment support)[16]，自适应权重(adaptive weight)[21]和测地线权重(geodesic weight)[5]。如引言中所述，这些聚合方法需要进行分割操作或逐像素迭代等耗时的操作，这对于高效的GPU实现来说是不可行的。尽管已经针对GPU系统提出了简化的自适应权重技术(1D聚合[13,18]和颜色平均[4,19])，但聚合精度通常会退化。最近，Rhemann等人[9]将聚合步骤制定为代价过滤(cost filtering)问题。通过使用设计好的(guided)滤波器平滑每个代价切片[1]，可以实现良好的视差结果。

我们重点关注Zhang等人最近提出的cross-based聚合方法[23]。我们证明，通过改进支撑区域构建和聚合的策略，该方法可以产生与自适应权重方法相当的聚合结果，并且计算时间更短。相对于自适应权重方法的另一个优点是为每个像素构造的支撑区域可以在稍后的后处理步骤中使用。

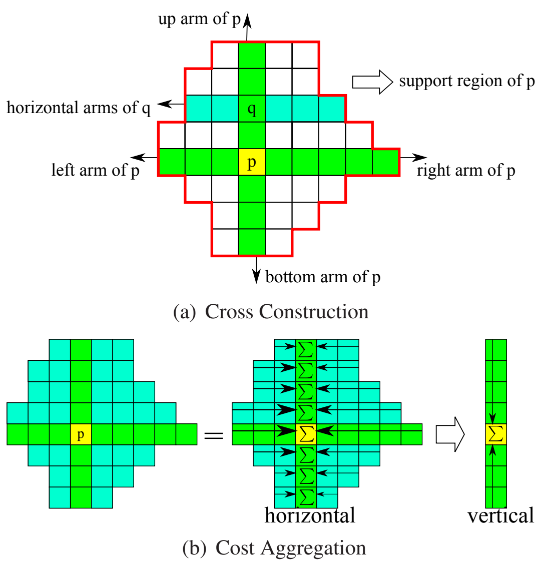

Figure 2. Cross-based aggregation: *在第一步中，为每个像素构造直立的十字支撑区域。像素p的支撑区域通过合并位于像素p的垂直方向上的像素（例如，q）的水平方向来构造。在第二步中，支撑区域中的代价沿水平和垂直方向聚合。*

基于交叉的聚合通过两个步骤进行，如图2所示。在第一步（图2（a））中，为每个像素构造具有四个方向的十字区域。以向左侧扩展为例，给定一个像素p，当它左侧的某个像素pl违反以下两条规则之一时，向左扩展就停止：

1. $D_c(\mathbf{p}_l, \mathbf{p}) < \tau$, 其中$D_c(\mathbf{p}_l, \mathbf{p})$是$\mathbf{p}_l$和$\mathbf{p}$之间的色差，$\tau$是色差阈值. 色差定义为 $D_c(\mathbf{p}_l, \mathbf{p}) = \max_{i=R,G,B} |I_i(\mathbf{p}_l) - I_i(\mathbf{p})|$.

2. $D_s(\mathbf{p}_l, \mathbf{p}) < L$, 其中$D_s(\mathbf{p}_l, \mathbf{p})$是P1和P之间的空间距离 L是最大长度L(以像素计). 空间距离定义为: $D_s(\mathbf{p}_l, \mathbf{p}) = |\mathbf{p}_l - \mathbf{p}|$.

这两条规则限值了支撑区域的颜色相似性和尺寸(通过超参数τ和L)。在第二步（图2（b））中，代价聚合分为两步计算：第一步计算横向总和并存储中间结果；第二步把中间求和结果以获得最终代价。两个过程都可以用1D积分图像有效地计算。为了获得稳定的代价值，聚合步骤通常运行2-4次迭代，这可以被视为各向异性扩散过程。关于该方法的更多细节可以在[23]中找到。

cross-based代价聚合的准确性与参数L和τ密切相关，因为它们控制支撑区域的形状。大的无纹理区域可能需要大的L和τ值以包括足够的强度变化，但是简单地增加所有像素的这些参数将在暗区域或深度不连续处引入更多误差。因此，我们采用了以下增强规则：

1. $D_c(\mathbf{p}_l, \mathbf{p}) < \tau_1$ and $D_c(\mathbf{p}_l, \mathbf{p}_l + (1, 0)) < \tau_1$

2. $D_s(\mathbf{p}_l, \mathbf{p}) < L_1$

3. $D_c(\mathbf{p}_l, \mathbf{p}) < \tau_2$, if $L_2 < D_s(\mathbf{p}_l, \mathbf{p}) < L_1$.

规则1不仅限制了p_l和p，之间的色差，而且还限制了p_l和它的前置p_l +(1, 0) o在同一个方向上的色差，这样就不会越过边缘。规则2和3允许对单方向长度进行更多的可控制。我们使用大的L_1值来为无纹理区域包含足够的像素。 但是当超过预设值L_2 (L_2 < L_1),时，更严格的阈值τ_2 (τ_2 < τ_1) 用 于 D_c(p_l, p) 以确保仅在具有非常相似的颜色模式的区域中延伸。

对于聚合步骤，我们还提出了不同的策略。我们仍然在此步骤执行4次迭代以获得稳定的代价值。对于第1和第3次迭代，我们遵循原始方法：先水平后垂直。但是对于第2和第4次迭代，先垂直后水平。对于每个像素，这种新的聚合顺序导致支撑区域与原始方法中的不同。通过改变聚合方向，在迭代过程中使用两个支持区域。我们发现这种聚合策略可以显着减少深度不连续处的误差。

通过原始的聚合方法和我们的改进方法计算的Tsukuba视差结果如图3所示，这表明增强的构造规则和聚合策略可以在大的无纹理区域和近深度不连续处产生更准确的结果。

使用三种聚合方法（自适应权重，原始的交叉聚合方法和我们的增强方法）评估WTA差异结果。对于自适应重量，参数遵循[21]中的设置。对于原始的交叉聚合方法，L=17， τ= 20并且使用4次迭代。四个数据集（非遮挡，不连续和所有区域）的平均误差百分比如图4所示。我们的增强方法在各种区域产生最准确的结果，特别是深度不连续区域。我们在自适应权重方法上的实现通常需要超过1分钟的CPU时间才能产生聚合值，而我们的方法只需要几秒钟。

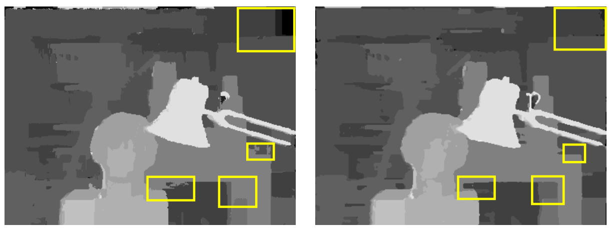
(a) original aggregation (b) improved aggregation

Figure 3. Comparison of the original cross-based aggregation method and our improved method on the Tsukuba image pair. Our aggregation method can better handle large textureless regions and depth discontinuities.

2.3. 扫描线优化

该步骤输入聚合的匹配代价值（表示为C_1）输出中间视差结果。为了进一步减轻匹配的模糊性，应采用具有平滑约束和中等平行度的优化器。我们采用基于Hirschmüller的半全局匹配方法的多方向扫描线优化器[2]。

四个扫描线优化过程独立地执行：2个沿水平方向，2个沿着垂直方向。给定扫描线方向 $\mathbf{r}$，像素p处的路径成本

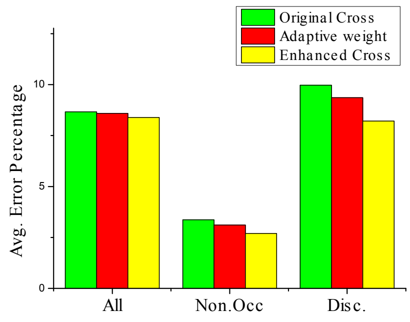

Figure 4. The average disparity error percentages in various regions for adaptive weight, the original cross-based aggregation method and our enhanced method.

$C_{\mathbf{r}}(\mathbf{p},d)$ 和视差d按照如下规则更新：

$$C_{\mathbf{r}}(\mathbf{p},d) = C_1(\mathbf{p},d) + \min(C_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d), \\ C_{\mathbf{r}}(\mathbf{p}-\mathbf{r},d\pm 1)+P_1, \\ \min_k C_{\mathbf{r}}(\mathbf{p}-\mathbf{r},k)+P_2) - \min_k C_{\mathbf{r}}(\mathbf{p}-\mathbf{r},k) \tag{4}$$

其中$\mathbf{p}-\mathbf{r}$是同方向上的前一个像素，$P_1, P_2\,(P_1 \le P_2)$ 是相邻像素之间的视差变化的两个惩罚参数。实验中，P_1, P_2根据左图中的色差$D_1 = D_c(\mathbf{p}, \mathbf{p}-\mathbf{r})$ 和右图中的色差$D_2 = D_c(\mathbf{pd}, \mathbf{pd}-\mathbf{r})$进行对称设置[8]:

1. $P_1 = \Pi_1, P_2 = \Pi_2$, if $D_1 < \tau_{SO}, D_2 < \tau_{SO}$.
2. $P_1 = \Pi_1/4, P_2 = \Pi_2/4$, if $D_1 < \tau_{SO}, D_2 > \tau_{SO}$.
3. $P_1 = \Pi_1/4, P_2 = \Pi_2/4$, if $D_1 > \tau_{SO}, D_2 < \tau_{SO}$.
4. $P_1 = \Pi_1/10, P_2 = \Pi_1/10$, if $D_1 > \tau_{SO}, D_2 > \tau_{SO}$.

其中Π_1，Π_2 是常量,τ_{SO}是色差阈值. 像素p的最终代价$C_2(\mathbf{p},d)$和视差d是通过对四个方向上的路径成本进行平均得到:

$$C_2(\mathbf{p},d) = \frac{1}{4}\sum_{\mathbf{r}} C_{\mathbf{r}}(\mathbf{p},d) \tag{5}$$

具有最小C_2值的视差被选中为$\mathbf{p}$的中间视差结果.

2.4. 多步视差修复

由前三个步骤计算的两个图像(表示为D_L和D_R)视差结果包含遮挡区域和深度不连续处的异常值。在对这些异常值进行检测之后，最简单的改进方法是用最接近的可靠差异来填充它们[11]，这只适用于小的遮挡区域。相反，我们在多步骤过程中系统地处理差异错误。 每个步骤都试图消除由各种因素引起的错误。

Outlier Detection: The outliers in D_L are first detect-ed with left-right consistency check: pixel $\mathbf{p}$ is an outlier if $D_L(\mathbf{p}) = D_R(\mathbf{p} - (D_L(\mathbf{p}), 0))$ doesn't hold. Outliers are further classified into occlusion and mismatch points, since they require different interpolation strategy. We follow the method proposed by Hirschmüller [2]: for outlier $\mathbf{p}$ at disparity $D_L(\mathbf{p})$, the intersection of its epipolar line and D_R is checked. If no intersection is detected, $\mathbf{p}$ is labelled as 'occlusion', otherwise 'mismatch'.

Iterative Region Voting: The detected outliers should be filled with reliable neighboring disparities. Most accu-rate stereo algorithms employ segmented regions for outlier handling [2, 20], which are not suitable for GPU implementation. We process these outliers with the constructed cross-based regions and a robust voting scheme.

For an outlier pixel $\mathbf{p}$, all the reliable disparities in its its cross-based support region are collected to build a histogram $H_{\mathbf{p}}$ with $d_{\max} + 1$ bins. The disparity with the highest bin value (most votes) is denoted as d^*. And the

$\mathbf{p}$

total number of the reliable pixels is denoted as $S_{\mathbf{p}} = \sum_{d=0}^{d=d_{\max}} H_{\mathbf{p}}(d)$. $\mathbf{p}$'s disparity is then updated with $d^*_{\mathbf{p}}$ if enough reliable pixels and votes are found in the support region:

$$S_{\mathbf{p}} > \tau_S, \frac{H_{\mathbf{p}}(d^*_{\mathbf{p}})}{S_{\mathbf{p}}} > \tau_H \tag{6}$$

where τ_S, τ_H are two threshold values.

To process as many outliers as possible, the voting process runs for 5 iterations. The filled outliers are marked as 'reliable' pixels and used in the next iteration, such that valid disparity information can gradually propagate into occlusion regions.

Proper Interpolation: The remaining outliers are filled with a interpolation strategy that treats occlusion and mismatch points differently. For outlier $\mathbf{p}$, we find the nearest reliable pixels in 16 different directions. If $\mathbf{p}$ is an occlusion point, the pixel with the lowest disparity value is selected for interpolation, since $\mathbf{p}$ most likely comes from the background; otherwise the pixel with the most similar color is selected for interpolation. With region voting and interpolation, most outliers are effectively removed from the disparity results, as shown in Figure 5.

Depth Discontinuity Adjustment: In this step, the disparities around the depth discontinuities are further refined with neighboring pixel information. We first detect all the edges in the disparity image. For each pixel $\mathbf{p}$ on the disparity edge, two pixels $\mathbf{p}_1$, $\mathbf{p}_2$ from both sides of the edge are collected. $D_L(\mathbf{p})$ is replaced by $D_L(\mathbf{p}_1)$ or $D_L(\mathbf{p}_2)$ if one of the two pixels has smaller matching cost than $C_2(\mathbf{p}, D_L(\mathbf{p}))$. This simple method helps to reduce the small errors around discontinuities, as shown by the error maps in Figure 6.

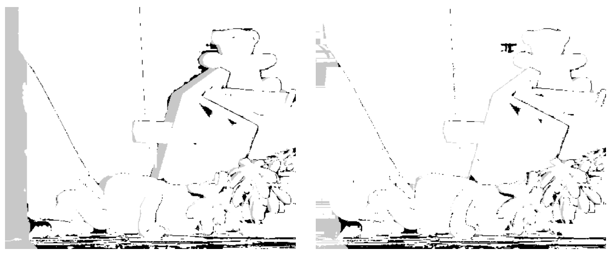

Figure 5. The disparity error maps for the Teddy image pair. The errors are marked in gray (occlusion) and black (non occlusion). The disparity errors are significantly reduced in the outlier handling process.

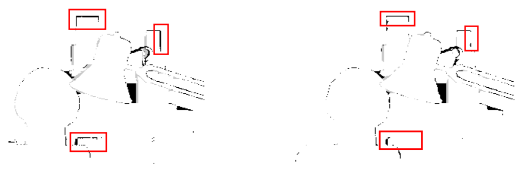
(a) before discontinuity adjustment
(b) after discontinuity adjustment

Figure 6. The errors around depth discontinuities are reduced after the adjustment step.

Sub-pixel Enhancement: Finally, a sub-pixel enhancement process based on quadratic polynomial interpolation is performed to reduce the errors caused by discrete disparity levels [20]. For pixel $\mathbf{p}$, its interpolated disparity d^* is computed as follows:

$$d^* = d - \frac{C_2(\mathbf{p}, d_+) - C_2(\mathbf{p}, d_-)}{2(C_2(\mathbf{p}, d_+) + C_2(\mathbf{p}, d_-) - 2C_2(\mathbf{p}, d))} \tag{7}$$

where $d = D_L(\mathbf{p}), d_+ = d + 1, d_- = d - 1$. The final disparity results are obtained by smoothing the interpolated disparity results with a 3×3 median filter.

To verify the effectiveness of the refinement process, the average error percentages in various regions after performing each refinement step are presented in Figure 7. The four refinement steps successfully reduce the error percentage in *all* regions by 3.8%, but their contributions are distinct for different regions: for *non-occluded* regions, voting and sub-pixel enhancement are most effective for handling the mismatch outliers; for *discontinuity* regions, the errors are significantly reduced by voting, discontinuity adjustment and sub-pixel enhancement; most outliers in *all* regions are removed with voting and interpolation, and small errors due to discontinuities and quantization are reduced by adjustment and sub-pixel enhancement. A systematic integration of these steps guarantees a strong post-processing method.

3. CUDA Implementation

Compute Unified Device Architecture (CUDA) is a programming interface for parallel computation tasks on NVIDIA graphics hardware. The computation task is coded into a *kernel* function, which is performed concurrently on data elements by multiple threads. The allocation of the threads is controlled with two hierarchical concepts: *grid* and *block*. A $kernel$ creates a *grid* with multiple *block*s, and each *block* consists of multiple threads. The performance of the CUDA implementation is closely related to thread allocation and memory accesses, which needs careful tuning in various computation tasks and hardware platforms. Given image resolution $W \times H$ and disparity range D, we briefly describe the implementation issues of our algorithm.

Cost Initialization: This step is parallelized with $W \times H$ threads. The threads are organized into a 2D grid and the block size is set to 32×32. Each thread takes care of computing a cost value for a pixel at a given disparity. For census transform, a square window is require for each pixel, which requires loading more data into the shared memory for fast access.

Cost Aggregation: A grid with $W \times H$ threads is created for both steps of the aggregation process. For cross construction, we set the block size to W or H, such that each block can efficiently handle a scanline. For cost aggregation, we follow the method proposed by Zhang *et al*. [24], which works similar to the first step. Each thread sums up a pixel's cost values horizontally and vertically in two passes. Data reuse with shared memory is considered in both steps.

Scanline Optimization: This step is different from the previous steps, because the process is sequential in the scanline direction and parallel in the orthogonal direction. A grid with $W \times D$ or $H \times D$ threads is created according to the scanline direction. D threads are allocated for each scanline, such that path costs on all disparity levels can be computed concurrently. Synchronization between the D threads is needed for finding the minimum cost of the previous pixel on the same path.

Disparity Refinement: Each step of the refinement process works on the intermediate disparity images, which can be efficiently processed with $W \times H$ threads.

4. Experimental Results

We test our system with the Middlebury benchmark [12]. The test platform is a PC with Core2Duo 2.20GHz CPU and NVIDIA GeForce GTX 480 graphics card. The parameters are given in Table 1, which are kept constant for all the data sets.

The disparity results are presented in Figure 8. Our system ranks first in the Middlebury evaluation, as shown in Table 2. Our algorithm performs well on all the data sets, and gives the best results on the Venus image pair with minimum errors both in non-occluded regions and near depth discontinuities. Compared to algorithms such as CoopRegion [17], the results on the Tsukuba image pair are not competitive. The Tsukuba image pair contains some very dark and noisy regions near the lamp and the desk, which lead to incorrect cross-based support regions for aggregation and refinement.

λ_{AD}	λ_{Census}	L_1	L_2	τ_1	τ_2
10	30	34	17	20	6
Π_1	Π_2	τ_{SO}	τ_S	τ_H	
1.0	3.0	15	20	0.4	

Table 1. Parameter settings for the Middlebury experiments

We run the algorithm both on CPU and on graphics hardware. For the four data sets (Tsukuba, Venus, Teddy and Cones), the CPU implementation requires 2.5 seconds, 4.5 seconds, 15 seconds and 15 seconds respectively, while the GPU implementation requires only 0.016 seconds, 0.032 seconds, 0.095 seconds and 0.094 seconds respectively. The GPU-friendly system design brings an impressive $140\times$ speedup in the processing speed. The average proportions of the GPU running time for the four computation steps are 1%, 70%, 28% and 1% respectively. The iterative cost aggregation step and the scanline optimization process dominate the running time.

Finally, we test our system on two stereo video sequences: a 'book arrival' scene from the HHI database (512×384, 60 disparity levels), and an 'Ilkay' scene from Microsoft i2i database (320×240, 50 disparity levels). To test the generalization ability of the system, we use the same set of parameters as the Middlebury datasets, and no temporal coherence information is employed in the computation process. The snapshots for the two examples are presented in Figure 9, and a video demo that runs at about 10FPS is available at `http://xing-mei.net/resource/video/adcensus.avi`. Our system performs reasonably well on these examples, but the results are not as convincing as the Middlebury datasets: artifacts are visible around depth boarders and occlusion regions.

We briefly discuss the limitations of the current system with the video examples. The disparity errors come from several aspects: first, the support regions defined by the cross skeleton rely heavily on color and connectivity constraints. For practical scenes the cross construction process can be easily corrupted by dark regions and image noise. Small regions without enough support area can be produced, which brings significant errors for later computation steps such as cost computation and region voting. Bilateral filtering might be used as a pre-process to reduce the noise while preserving the image edges [1, 15]. Second, the well-designed multi-stage mechanism is a double-edged sword. It help us to get accurate results and remove the errors step by step in a systematic way, but it also brings a large set of parameters. By carefully tuning individual parameters, the disparity quality can be improved, but such a scheme is usually laborious and impractical for various real-world applications. A possible solution is to analyze the robustness of the parameters with ground truth data and adaptively set the 'unstable' parameters with different visual contents. Automatic parameter estimation within an iterative framework [25] might also be used to avoid the tricky parameter tuning process.

5. Conclusions

This paper has presented a near real-time stereo system with accurate disparity results. Our system is based on several key techniques: AD-Census cost measure, cross-based support regions, scanline optimization and a systematic refinement process. These techniques significantly improve the disparity quality without sacrificing performance and parallelism, which are suitable for GPU implementation. Although our system presents nice results for the Middlebury data sets, applying it in real world applications is still a challenging task, as shown by the video examples. Real world data usually contains significant image noise, rectification errors and illumination variation, which might cause serious problems for cost computation and support region construction. And robust parameter setting methods are also important to produce satisfactory results. We would like to explore these topics in the future.

Acknowledgment

The authors would like to thank Daniel Scharstein for the Middlebury test bed and personal communications.

References

[1] K. He, J. Sun, and X. Tang. Guided image filtering. In *Proc. ECCV*, 2010.

[2] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE TPAMI*, 30(2):328–341, 2008.

[3] H. Hirschmüller and D. Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE TPAMI*, 31(9):1582–1599, 2009.

[4] A. Hosni, M. Bleyer, and M. Gelautz. Near real-time stereo with adaptive support weight approaches. In *Proc. 3DPVT*, 2010.

[5] A. Hosni, M. Bleyer, M. Gelautz, and C. Rheman. Local stereo matching using geodesic support weights. In *Proc. ICIP*, pages 2093–2096, 2009.

[6] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *ICPR*, pages 15–18, 2006.

[7] J. Liu and J. Sun. Parallel graph-cuts by adaptive bottom-up merging. In *Proc. CVPR*, pages 2181 – 2188, 2010.

[8] S. Mattoccia, F. Tombari, and L. D. Stefano. Stereo vision enabling precise border localization within a scanline optimization framework. In *Proc. ACCV*, pages 517–527, 2007.

[9] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *Proc. CVPR*, 2011.

[10] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *Proc. ECCV*, pages 6311–6316, 2010.

[11] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3):7–42, 2002.

[12] D. Scharstein and R. Szeliski. Middlebury stereo evaluation - version 2, 2010. `http://vision.middlebury.edu/stereo/eval/`.

[13] X. Sun, X. Mei, S. Jiao, M. Zhou, and H. Wang. Stereo matching with reliable disparity propagation. In *Proc. 3DIMPVT*, pages 132–139, 2011.

[14] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE TPAMI*, 30(6):1068–1080, 2008.

[15] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. ICCV*, pages 839–846, 1998.

[16] F. Tombari, S. Mattoccia, and L. D. Stefano. Segmentation-based adaptive support for accurate stereo correspondence. In *Proc. PSIVT*, pages 427–438, 2007.

[17] Z. Wang and Z. Zheng. A region based stereo matching algorithm using cooperative optimization. In *Proc. CVPR*, pages 1–8, 2008.

[18] Y. Wei, C. Tsuhan, F. Franz, and C. H. James. High performance stereo vision designed for massively data parallel platforms. *IEEE TCSVT*, 99:1–11, 2010.

[19] Q. Yang, C. Engels, and A. Akbarzadeh. Near real-time stereo for weakly-textured scenes. In *Proc. BMVC*, pages 80–87, 2008.

[20] Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling. *IEEE TPAMI*, 31(3):492–504, 2009.

[21] K.-J. Yoon and I.-S. Kweon. Adaptive support-weight approach for correspondence search. *IEEE TPAMI*, 28(4):650–656, 2006.

[22] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proc. ECCV*, pages 151–158, 1994.

[23] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE TCSVT*, 19(7):1073–1079, 2009.

[24] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. V. Gool. Real-time accurate stereo with bitwise fast voting on cuda. In *Proc. ICCV Workshop*, 2009.

[25] L. Zhang and S. M. Seitz. Estimating optimal parameters for mrf stereo from a single image pair. *IEEE TPAMI*, 29(2):331–342, 2007.

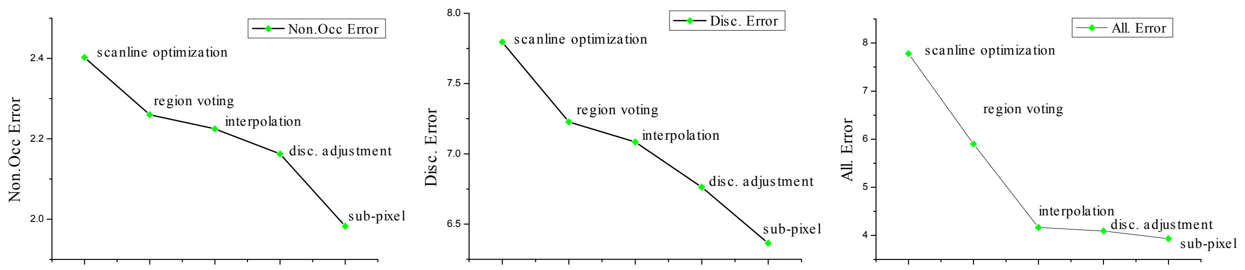

Figure 7. The average error percentages in *non-occlusion*, *discontinuity* and *all* regions after performing each refinement step.

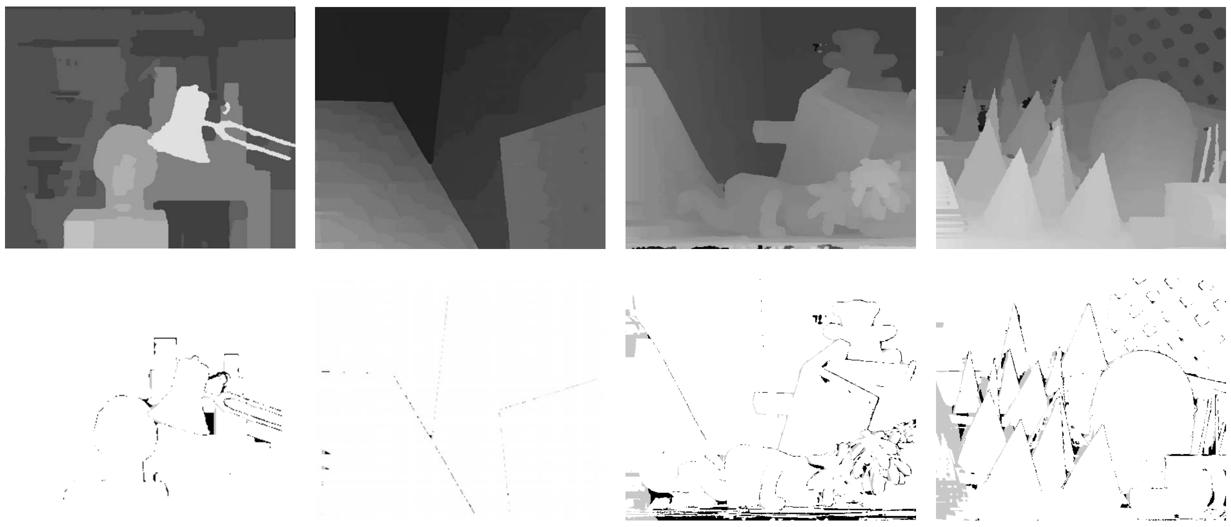

Figure 8. Results on the Middlebury data sets. First row: disparity maps generated with our system. Second row: disparity error maps with threshold 1. Errors in unoccluded and occluded regions are marked in black and gray respectively.

Algorithm	Avg. Rank	Tsukuba			Venus			Teddy			Cones		
		nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc
Our method	5.8	1.07_{12}	1.48_{10}	5.73_{14}	$\mathbf{0.09}_{2}$	0.25_{7}	$\mathbf{1.15}_{2}$	4.10_{4}	6.22_{3}	10.9_{4}	2.42_{3}	7.25_{5}	6.95_{4}
AdaptingBP [6]	7.2	1.11_{15}	1.37_{6}	5.79_{15}	0.10_{3}	0.21_{4}	1.44_{4}	4.22_{6}	7.06_{6}	11.8_{7}	2.48_{4}	7.92_{9}	7.32_{7}
CoopRegion [17]	7.2	0.87_{3}	1.16_{1}	4.61_{2}	0.11_{4}	0.21_{3}	1.54_{6}	5.16_{14}	8.31_{10}	13.0_{11}	2.79_{12}	7.18_{4}	8.01_{16}
DoubleBP [20]	9.7	0.88_{5}	1.29_{3}	4.76_{5}	0.13_{7}	0.45_{17}	1.87_{11}	3.53_{3}	8.30_{9}	9.63_{2}	2.90_{17}	8.78_{24}	7.79_{13}

Table 2. The rankings in the Middlebury benchmark. The error percentages in different regions for the four data sets are presented.

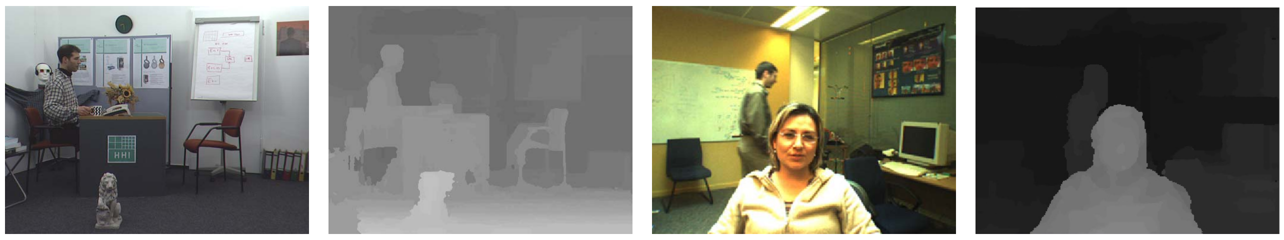

Figure 9. Snapshots on 'book arrival' and 'Ilkay' stereo video sequences.