基于图像分割的稠密立体匹配算法

马瑞浩[1,2,3]，朱枫[1,3]*，吴清潇[1,3]，鲁荣荣[1,3]，魏景阳[1,3]

[1]中国科学院沈阳自动化研究所，辽宁 沈阳 110016；

[2]东北大学信息科学与工程学院，辽宁 沈阳 110819；

[3]中国科学院光电信息处理重点实验室，辽宁 沈阳 110016

摘要 提出一种基于图像分割的稠密立体匹配算法，该算法将灰度-梯度算法与零均值归一化互相关(ZNCC)算法相结合生成匹配代价，利用 SLIC(Simple Liner Iterative Cluster)算法对图像进行分割，基于视差图和超像素更新了匹配代价。在视差后处理阶段，基于左右一致性检验(LRC)、孔洞填充和十字交叉自适应窗口加权中值滤波的方法减小视差图的误匹配率。利用 Middlebury 数据集的 4 组图像进行测试，测试结果表明，平均误匹配率为4.99%。

关键词 机器视觉；立体匹配算法；匹配代价计算方法融合；十字交叉自适应窗口加权中值滤波

中图分类号 TN911.73 **文献标识码** A **doi**：10.3788/AOS201939.0315001

Dense Stereo Matching Algorithm Based on Image Segmentation

Ma Ruihao[1,2,3], Zhu Feng[1,3]*, Wu Qingxiao[1,3], Lu Rongrong[1,3], Wei Jingyang[1,3]

[1] *Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, Liaoning* 110016, *China*;

[2] *College of Information Science and Engineering, Northeastern University, Shenyang, Liaoning* 110819, *China*;

[3] *Key Laboratory of Opto-Electronic Information Processing, Chinese Academy of Sciences, Shenyang, Liaoning* 110016, *China*

Abstract A dense stereo matching algorithm is proposed based on image segmentation. This algorithm combines the gray-gradient algorithm and the zero-mean normalized cross-correlation (ZNCC) algorithm to generate matching cost. The SLIC (Simple Liner Iterative Cluster) algorithm is used for image segmentation. A method based disparity map and superpixels is proposed to update the matching cost. At the disparity post-processing stage, the LRC (Left Right Check), hole filling and cross adaptive window weighted median filtering methods are used to reduce the error matching rate of the disparity map. The performance evaluation experiments on four Middlebury stereo pairs demonstrate that the proposed algorithm achieves an average error matching rate of 4.99%.

Key words machine vision; stereo matching algorithm; matching cost computation method fusion; cross adaptive window weighted median filtering

OCIS codes 150.1135; 330.1400; 100.6890

1 引 言

立体匹配是计算机视觉中的热门研究方向，广泛应用于机器人导航、虚拟现实、三维重建等领域[1]。Scharstein 等[2]将立体匹配算法分为 4 个步骤：1)匹配代价计算；2)代价聚合；3)视差计算或优化；4)视差细化(视差后处理)。根据是否采用代价聚合步骤，可以将传统的立体匹配算法分为全局、半全局和局部立体匹配算法。全局立体匹配算法通常跳过匹配代价聚合步骤，直接进行视差的计算和优化，如图割法[3]、置信度传播法[4]等。全局立体匹配算法虽然精度高，但是计算效率低。半全局立体匹配算法中的动态规划算法[5-6]是最为典型的方法，传统动态规划算法的结果会出现扫描线效应。局部立体匹配算法一般利用匹配点的局部信息计算匹配代价，采用代价聚合方法来改善匹配代价，然后利用 WTA(Winner Take All)算法得到视差图，虽然精度低，但是计算效率高。

近几年，以深度学习为代表的机器学习被用于解决立体匹配问题，并且取得了非常好的效果。机

收稿日期：2018-09-13；**修回日期**：2018-10-10；**录用日期**：2018-10-21

基金项目：国家自然科学基金(U1713216)

* **E-mail**：fzhu@sia.cn

器学习主要是将卷积神经网络(CNN)应用在立体匹配中。基于CNN解决立体匹配问题的方法也大致分为三类[7]：1)利用CNN学习匹配代价，然后利用传统方法进行视差后处理，例如MC-CNN(Matching Cost-Convolutional Neural Network)及其改进网络[8]；2)从端到端训练CNN，直接从图像对估计视差，例如DispNets[9]网络；3)利用多个网络得到视差图，例如CRL(Cascade Residual Learning)[10]网络，由DisFullNet和DisResNet两个网络组成。基于CNN网络的方法取得了非常好的效果，但是也具有一定的局限性：首先遮挡区域的像素点不能用来训练，这意味着很难在这些区域获得可靠的视差估计[7]；其次，训练神经网络需要大量的数据，在某些特定场合，无法得到训练网络所需要的数据，这使得基于CNN的方法受到限制。

传统方法不需要训练数据，且已取得了较好的结果。Rhemann等[11]将图像引导滤波器(GF)应用在立体匹配中，不仅能够减轻视差图边缘模糊现象，而且计算效率高。Yang[12]将最小生成树(MST)引入匹配代价聚合，不仅速度快，而且精度高，但是在遮挡处容易误匹配。Zhang等[13]提出跨尺度代价聚合模型，优化了GF、MST等算法的结果，但是对遮挡处的误匹配问题优化效果不佳。刘艳等[14]提出结合局部二进制表示和超像素分割求精的方法，改善了弱纹理区域的阶梯效应，但是在平均误匹配率较高。龚文彪等[15]利用mean shift分割算法，分割出不同深度区域的匹配点，根据匹配点所在的深度区域进行匹配代价重定义，在遮挡处的效果较好，但是在其他区域的误匹配率较高。为降低所有区域的误匹配率，本文提出基于图像分割的稠密立体匹配方法。

2　基于图像分割的立体匹配算法

所提算法的流程分为三个部分：1)匹配代价融合算法和SLIC(Simple Liner Iterative Cluster)[16]算法；2)匹配代价聚合方法，包括基于视差图和超像素更新匹配代价和GF算法；3)视差后处理方法，包括左右一致性检验(LRC)、孔洞填充和十字交叉自适应窗口加权中值滤波。匹配代价聚合、视差后处理以及整个算法的流程图如图1所示。

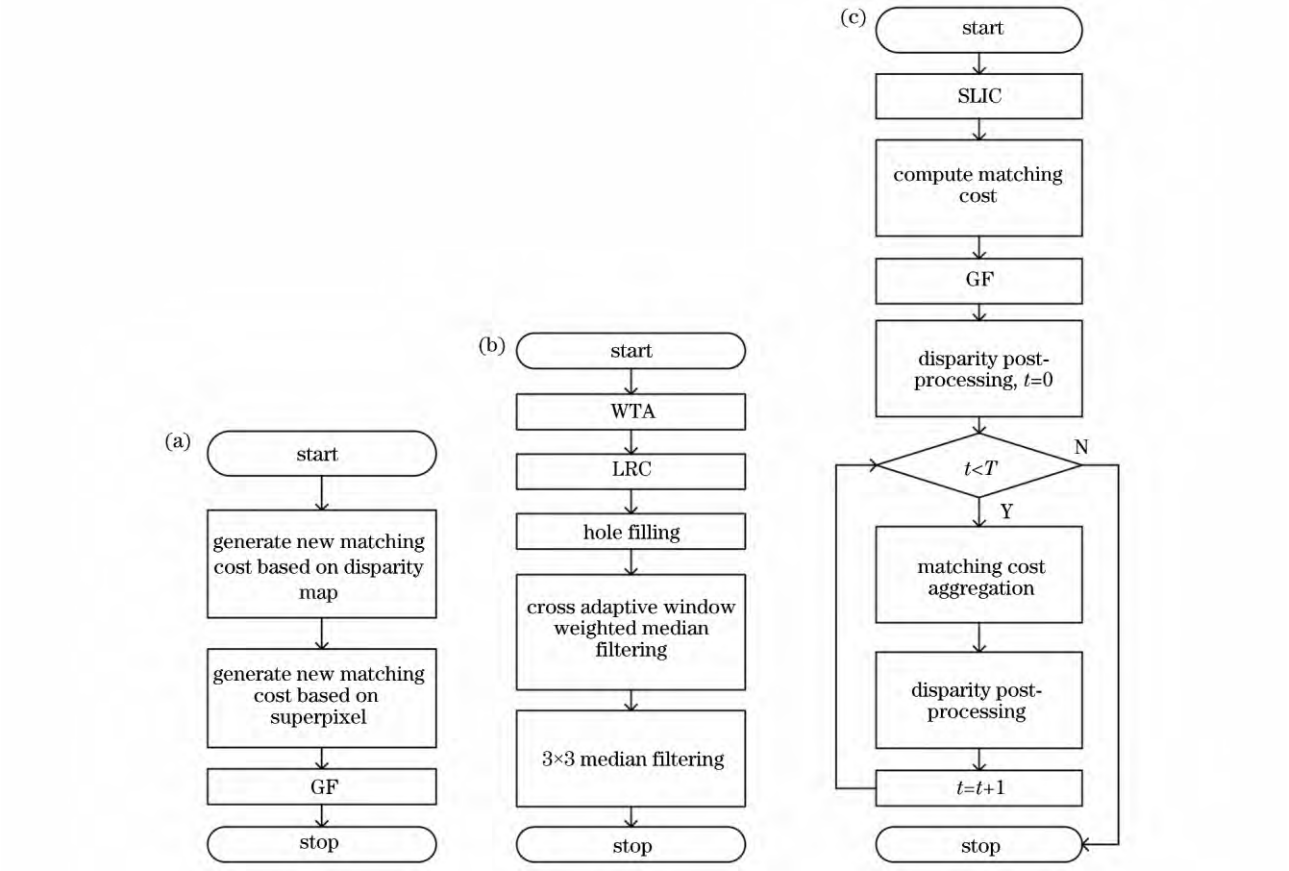

图1　算法流程图。(a)匹配代价聚合；(b)视差后处理；(c)综合流程图

Fig. 1　Flow chart of algorithm. (a) Matching cost aggregation; (b) disparity post-processing; (c) integrated flow chart

2.1　匹配代价计算

匹配代价计算基于立体校正过的图像，即同一物体在左右两幅图像的同一行。匹配代价计算就是计算左右图像中各个点之间的相似性程度。分别计算图像中每点的灰度-梯度算法[11]和零均值归一化互相关(ZNCC)算法的匹配代价。灰度-梯度算法的输入为RGB图像，而ZNCC算法的输入是由RGB图像转换成的灰度图像。

灰度-梯度算法的计算公式为

$$C_{\text{gray-gradient}}=(1-\alpha)\min[\|I_{\text{L}}(x,y)-I_{\text{R}}(x+d,y)\|,\tau_1]+\alpha\min[\|\nabla_x I_{\text{L}}(x,y)-\nabla_x I_{\text{R}}(x+d,y)\|,\tau_2], \tag{1}$$

式中：$C_{\text{gray-gradient}}$为灰度-梯度算法匹配代价；I_{L}为左图像；$I_{\text{R}}(x+d,y)$为右图像；x和y表示像素在图像中的坐标；d表示视差值；$\nabla_x I_{\text{L}}$和$\nabla_x I_{\text{R}}$为图像水平方向的梯度图像；α为梯度图像匹配代价的权重；τ_1、τ_2为常量。

ZNCC算法的计算公式为

$$C_{\text{ZNCC}}=\frac{\sum\limits_{i,j\in W_{\text{in}}}[I_{\text{L}}(x+i,y+j)-\overline{I}_{\text{L}}(x,y)]\times[I_{\text{R}}(x+i+d,y+j)-\overline{I}_{\text{R}}(x+d,y)]}{\sqrt{\sum\limits_{i,j\in W_{\text{in}}}[I_{\text{L}}(x+i,y+j)-\overline{I}_{\text{L}}(x,y)]^2}\times\sqrt{\sum\limits_{i,j\in W_{\text{in}}}[I_{\text{R}}(x+i+d,y+j)-\overline{I}_{\text{R}}(x+d,y)]^2}}, \tag{2}$$

式中：C_{ZNCC}为ZNCC算法匹配代价；W_{in}是匹配窗口；i和j表示窗口中以中心元素为原点时，窗口内元素的坐标值，窗口大小为$N\times N$；$\overline{I}_{\text{L}}(x,y)$和$\overline{I}_{\text{R}}(x+d,y)$分别是左图像和右图像窗口$W_{\text{in}}$的平均值。

一般在物体边缘处容易发生遮挡，遮挡处容易发生误匹配。遮挡就是在相机的公共视场内，物体在左图中可以看到而在右图中看不到，或物体在左图中看不到而在右图中能看到。物体边缘处一般为图像边缘处。图像边缘和非边缘处具有不同的特征，因此在边缘和非边缘处对两种匹配代价计算方法分配不同的权值。由于灰度-梯度算法对灰度变化敏感，而物体边缘处一般灰度发生变化较大，因此在物体边缘处，给灰度-梯度算法分配的权值较大，给ZNCC算法分配的权值较小。反之，在非边缘区域，ZNCC算法对噪声具有较强的稳健性，因此给ZNCC算法分配的权值较大，给灰度-梯度算法分配的权值较小。

匹配代价计算公式为

$$C=\begin{cases}(1-|C_{\text{ZNCC}}|)(2-\beta)+C_{\text{gray-gradient}}(2-\gamma)/2, & \text{edge}\\(1-|C_{\text{ZNCC}}|)\beta+C_{\text{gray-gradient}}\gamma/2, & \text{others}\end{cases}, \tag{3}$$

式中：β和γ分别为ZNCC算法和灰度-梯度算法在非边缘区域的权值，都为常量。

图像的边缘可以利用SLIC超像素分割算法得到。SLIC算法将图像从RGB颜色空间转换到CIE-Lab颜色空间，每个像素的颜色(l,a,b)和坐标(x,y)组成一个5维向量$\boldsymbol{V}=[l,a,b,x,y]$，两个像素的相似性由对应向量$\boldsymbol{V}$的欧式距离来测量。然后在图像上生成$K$个种子点，在每个种子点的周围空间搜索距离该种子最近的若干个像素，将它们与该种子归为同一类，直到所有像素点都归类完毕。然后计算K个超像素里所有像素点的平均向量值，再以K个平均向量值为中心去搜索其周围与其最相似的若干像素，所有像素归类完毕后重新得到K个超像素。更新聚类中心，再次迭代，如此反复直到收敛。种子点个数K可表示为

$$K=\frac{r_{\text{ow}}c_{\text{ol}}}{\rho}, \tag{4}$$

式中：r_{ow}和c_{ol}表示图像的行和列；ρ为超像素内初始的像素个数，为常量。

对Tsukuba进行SLIC超像素分割，根据分割结果提取边缘，其结果如图2所示。

2.2　匹配代价聚合

匹配代价聚合分3个步骤：1)通过视差图更新匹配代价；2)利用超像素更新匹配代价；3)利用GF

图 2　SLIC 算法结果。(a) Tsukuba 图像;(b)超像素分割图;(c)边缘图像

Fig. 2　Result of SLIC algorithm. (a) Tsukuba image; (b) superpixel segmentation image; (c) edge image

对匹配代价进行滤波。

首先利用视差图对匹配代价进行更新,视差图的更新公式为

$$C_{\text{new}}(x,y,d)=|\,d-D(x,y)\,|, \tag{5}$$

式中:$C_{\text{new}}(x,y,d)$为更新后的匹配代价;$D(x,y)$表示视差图;视差 $d\in[0,d_{\max}]$,$d_{\max}$为视差最大值。

利用超像素更新匹配代价,超像素内大多数像素拥有正确的视差值,用这些正确的视差更新超像素内像素的匹配代价。匹配代价更新公式为

$$C_{\text{new}}=C_{\text{old}}\times\exp\left(-\frac{n_{d,s}}{n_s}\right), \tag{6}$$

式中:n_s 表示第 s 块超像素中像素数目;$n_{d,s}$ 表示第 s 块超像素中视差值为 d 的像素数目;C_{old}为更新之前的匹配代价;C_{new}为更新之后的匹配代价。

为了解决边缘处的误匹配问题,使用 GF 对匹配代价进行滤波,GF 滤波器定义为

$$\begin{cases} g_p=\sum\limits_{q\in w_p}W_{pq}(I)f_q \\ W_{pq}(I)=\dfrac{1}{|\,w_k\,|^2}\sum\limits_{k:(p,q)\in w_k}\left[1+(\boldsymbol{I}_p-\boldsymbol{\mu}_k)^{\mathrm{T}}(\boldsymbol{\Sigma}_k+\varepsilon\boldsymbol{E})^{-1}(\boldsymbol{I}_q-\boldsymbol{\mu}_k)\right] \end{cases}, \tag{7}$$

式中:I 为 RGB 引导图像;f_q 为待滤波图像,即匹配代价图像;g_q 为滤波后的匹配代价图像;w_p 为以像素 p 为中心的窗口;$W_{pq}(I)$为像素点 p 和 q 之间的权值;w_k 为以像素 k 为中心的窗口,公式中窗口大小都为 $r\times r$;$\boldsymbol{I}_p$ 和 $\boldsymbol{I}_q$ 为窗口 w_k 内像素的 RGB 颜色值,为三维向量;$|w_k|$为窗口内像素的数目;$\boldsymbol{\mu}_k$ 为窗口 w_k 的均值,为三维向量;$\boldsymbol{\Sigma}_k$ 为 3×3 的协方差矩阵;ε 为惩罚系数;$\boldsymbol{E}$ 为3×3 的单位矩阵。

2.3　视差后处理

完成匹配代价聚合后,使用 WTA 算法得到视差图。但是视差图上存在许多误匹配点,视差后处理在一定程度上可以消除这些误匹配点。视差后处理包括 LRC 算法,孔洞填充,十字交叉自适应窗口加权中值滤波和 3×3 的中值滤波。

LRC 算法是检测误匹配点的重要且有效的方法,其主要思想是:当以左图像为参考图时,生成左视差图,反之生成右视差图。左视差图和右视差图中匹配点的视差值相同,视差值不同的点为误匹配点。LRC 算法的计算公式为

$$|\,D_{\text{L}}(x-d,y)-D_{\text{R}}(x,y)\,|\leqslant 1, \tag{8}$$

式中:$D_{\text{L}}(x-d,y)$表示左视差图;$D_{\text{R}}(x,y)$表示右视差图。

通过 LRC 算法标记出误匹配点,在孔洞填充阶段对误匹配点进行填充。孔洞填充可以看作是一个图像修复问题,经典的图像修复算法[17-19]常利用偏微分方程、纹理合成、结构特征等方法修复图像,虽然能取得很好的结果,但是计算效率低,一般在孔洞填充阶段很少采用。传统的视差图孔洞填充算法是分别寻找误匹配点所在行的左侧和右侧距离误匹配点最近的有效匹配点,用这 2 个有效匹配点视差值中较小的视差值填充误匹配点,虽然效果一般,但是速度快。为了兼顾视差图修复过程的速度和精度,改进传统的视差图孔洞填充算法过程,不仅在误匹配点所在行寻找有效匹配点,还在误匹配点所在行的上一行和下一行寻找有效匹配点,此时一共有 6 个有效匹配点视差值,取 6 个视差值中的最小值填充误匹配点。但当选择同一个视差值作为大量无效点的填充值时,会在视差图上产生条纹效应,如图 3(a)所示。

通过十字交叉自适应窗口加权中值滤波来减弱

条纹效应，其效果图如图3(b)所示。首先确定加权中值滤波的支持域，其支持域示意图如图4所示。图4中 p 点为误匹配点，q 是其邻域点。支持域的确定过程如下：首先在 p 点的竖直方向进行扩展，然后在扩展竖直方向的基础上，扩展水平方向，得到加权中值滤波的支持域。为了加快计算速度，只计算误匹配点的自适应窗口。用于判断 q 点是否在 p 点的支持域内[20]的计算公式为

$$\begin{cases} D_c(p,q)<\theta_1 \text{ and } D_c(p,q^+)<\theta_1 \\ q^+=\begin{cases} q(x,y+1), & \text{vertical} \\ q(x+1,y), & \text{horizonal} \end{cases} \\ D_s(p,q)<L_1 \\ D_c(p,q)<\theta_2 \text{ if } L_2<D_s(p,q)<L_1 \end{cases}, \tag{9}$$

式中：D_c 表示点 p 与点 q 的RGB颜色差值；D_s 表示两点之间的距离差值；L_1、L_2、θ_1、θ_2 为常数。

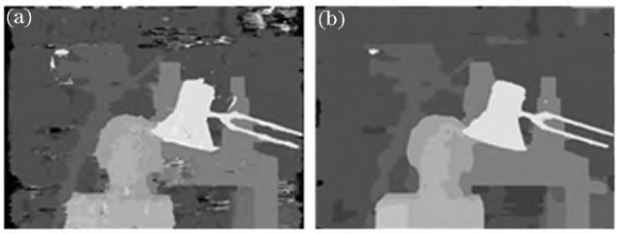

图3　不同算法所得结果。(a)孔洞填充；(b)十字交叉自适应窗口加权中值滤波

Fig. 3　Results of different algorithms. (a) Hole filling; (b) cross adaptive window weighted median filtering

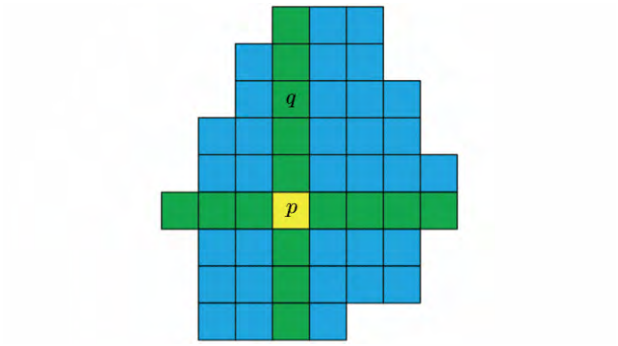

图4　支持域

Fig. 4　Support region

在支持域内进行加权中值滤波，权值通过RGB颜色值计算[21]：

$$w_{pq}=\exp\left[-\frac{(I_p-I_q)^2}{\sigma_c^2}\right], \tag{10}$$

式中：I_p 和 I_q 分别为 p 点和 q 点的RGB颜色值；σ_c 为常量。

求出权值后，再取中值。构造视差直方图：

$$h(x,d)=\sum_{x'\in N_s(x)} w(x,x')\delta[V(x_{x'})-d], \tag{11}$$

式中：$h(x,d)$ 为 p 点的直方图；x 为 p 点的坐标；x' 为支持域内其他点的坐标；$N_s(x)$ 表示点 x 的支持域；V 表示取点的视差值操作；$\delta(\cdot)$ 为狄拉克函数；$w(x,x')$ 表示权值。

将权值按照视差值从小到大的顺序相加，当权值之和大于等于总权值的一半时，此时对应的视差为加权中值滤波的结果，即

$$\begin{cases} d^*=\min d \\ \text{s.t } \sum_{j_d=d_{\min}}^{d} h(x,j_d)\geqslant\frac{1}{2}\sum_{j_d=d_{\min}}^{d_{\max}} h(x,j_d) \end{cases}, \tag{12}$$

式中：d^* 为加权中值滤波的结果；$d_{\min}$、$d_{\max}$ 分别表示视差的最小值和最大值。

3　实验结果与分析

实验环境为：Windows7 64位系统，Intel(R) Core(TM)i7-6700CPU主频3.4 GHz，4核，8 GB内存。使用Middlebury[22]平台提供的Tsukuba、Venus、Teddy和Cones 4组彩色图像进行测试。利用这4组图像确定算法参数，并与其他算法进行比较。除这4组图像外，再加上平台提供的27组图像，共利用31组图像对所提算法进行测试。

3.1　参数设置

所提算法参数设置如下：灰度-梯度算法和GF的参数采用文献[11]的参数 $\alpha=0.9$，$\tau_1=10$，$\tau_2=2$，$r=9$，$\varepsilon=0.0001$；十字交叉自适应窗口的参数采用文献[20]的参数 $L_1=62$，$L_2=32$，$\theta_1=32$，$\theta_2=16$；加权中值滤波的参数采用文献[21]的参数 $\sigma_c=5$；剩下的参数ZNCC窗口大小 N，匹配代价融合系数 β、γ，超像素分割系数 ρ 和迭代次数 T 通过实验确定。

首先，确定ZNCC窗口大小 N 的值，具体方法如下：1)计算ZNCC的匹配代价，GF滤波得到新的匹配代价；2)采用WTA算法得到视差图；3)计算AvgPBM。AvgPBM表示4组图像在非遮挡区域、所有区域和深度不连续区域(遮挡处)三个区域误匹配率的平均误匹配率。选择误匹配极限误差 $E_{\text{rror}}>1$，即得到的视差图与标准视差图的像素差值大于1时认为是误匹配。N 与AvgPBM的关系如图5(a)所示，当 $N=5$ 时，AvgPBM值最小。

下一步确定 β 和 γ 的值，与确定 N 的方法相同。固定 $N=5$、$\rho=1000$。β、γ 取不同的值时与AvgPBM的关系如图5(b)所示。当 $\beta=0.9$、$\gamma=0.3$

时，AvgPBM 值最小。

确定 N、β、γ 的值之后，选择迭代次数 $T=2$，固定这几个值。按照第 2 节描述的算法得到视差图。超像素分割系数 ρ 与 AvgPBM 的关系如图 5(c)所示。当 $\rho=2300$ 时，AvgPBM 值最小。

固定 $N=5$、$\beta=0.9$、$\gamma=0.3$ 和 $\rho=2300$，确定迭代次数 T 的值。迭代次数 T 与 AvgPBM 的关系如图 5(d)所示。两次迭代后，AvgPBM 大幅度减小，第三次迭代后 AvgPBM 减小幅度较小，所以迭代两次后停止迭代，即 $T=2$。

通过实验，最终确定：$N=5$，$\beta=0.9$，$\gamma=0.3$，$\rho=2300$，$T=2$。

3.2　算法对比分析

确定参数之后，在 Middlebury 提供的 Tsukuba、Venus、Teddy 和 Cones 4 组图像上进行测试，并将所提算法与 MST[13]、GF[12] 和 GA-DP (Gradient Adaptive-Dynamic Programming)[5] 算法进行对比，如图 6 所示。从图 6 中可以看出所提算法在 Tsukuba 上的视差图效果一般，但是在其他三张视差图上效果较好。

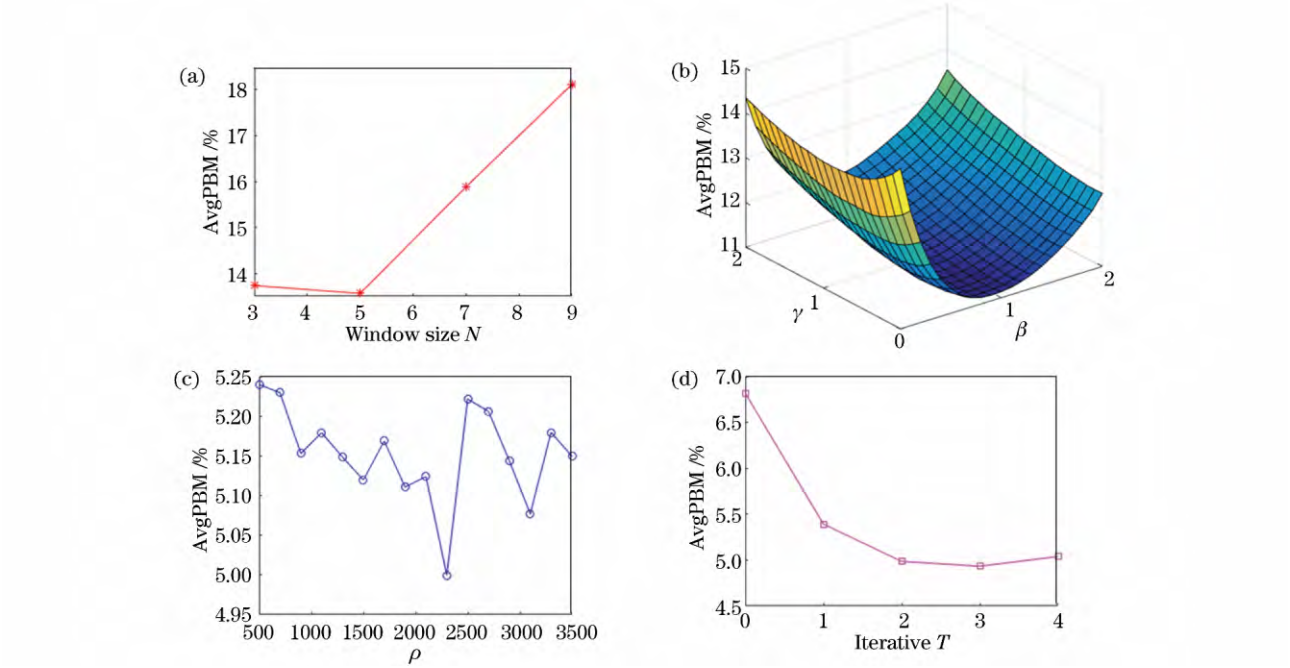

图 5　不同参数对平均误差的影响。(a) N；(b) β 和 γ；(c) ρ；(d) T

Fig. 5　Influences of parameters on AvgPBM. (a) N; (b) β and γ; (c) ρ; (d) T

所提算法、MST、GF 和 GA-DP 算法的误匹配率如表 1 所示。表 1 中“n-occ”表示非遮挡区域，“all”表示所有区域，“disc”表示深度不连续区域(遮挡处)，AvgDisc 表示 4 组图像在遮挡区域的平均误匹配率。表 1 中，Gray 表示采用所提算法，但是输入图像为灰度图像。从表 1 中可以看出，当灰度图像作为输入时，比 RGB 图像作为输入在 AvgPBM 上高 1.47%，在 AvgDisc 上高 2.85%。因此采用灰度图像作为输入时，效果比 RGB 图像作为输入要差。所提算法的 AvgPBM 比 MST、GF 和 GA-DP 算法的分别低 0.49%、0.56%和 1.11%。但是所提算法在 Tsukuba 上的 n-occ 和 all 区域的误匹配率较高，Tsukuba 图像匹配错误的地方集中在无纹理和弱纹理区域，因此所提算法在无纹理和弱纹理区域容易出现误匹配。所提算法的 AvgDisc 比 MST、GF 和 GA-DP 算法的分别低 0.72%、1.06%和 1.44%。可见所提算法的 AvgPBM 和 AvgDisc 较低。

为全面地测试算法性能，对 Middlebury 提供的 31 组图像上进行测试，测试结果如表 2 所示。表 2 中数字下标表示各个算法匹配精度的排名，AvgErr 表示算法在 n-occ 区域的平均误匹配率，AvgRank 表示算法的平均排名。在表 2 中，灰度图像作为输入比直接输入 RGB 图像的 AvgErr 高1.32%，但是比 MST 算法低 0.49%，因此得出所提算法可以用灰度图像作为输入的结论。所提算法在 Midd1、Midd2、Monopoly、Plastic 和 Reindeer 的误匹配率比较高，是因为这 5 幅图像中有大量无纹理和弱纹理区域。但是所提算法的 AvgErr 分别比 GF、CS-MST 和 MST 低0.83%、1.02%和1.81%。在4种

表 1　不同算法下的平均误匹配率

Table 1　AvgPBM for different algorithms　%

Algorithm	Tsukuba			Venus			Teddy			Cones			Avg PBM	Avg Disc
	n-occ	all	disc	n-occ	all	disc	n-occ	all	disc	n-occ	all	disc		
Proposed	1.50	1.95	**6.71**	**0.11**	**0.33**	**1.25**	**5.27**	**10.8**	14.5	**2.38**	**8.02**	**7.01**	**4.99**	**7.36**
MST	**1.47**	**1.85**	7.88	0.25	0.42	2.60	6.01	11.6	**14.3**	2.87	8.45	8.10	5.48	8.08
GF	1.51	**1.85**	7.61	0.20	0.93	2.42	6.16	11.8	16.0	2.71	8.24	7.66	5.55	8.42
GA-DP	1.57	2.00	7.32	0.89	1.00	3.18	7.20	12.4	16.1	3.68	9.18	8.62	6.10	8.80
Gray	1.91	2.74	9.70	0.32	0.68	4.25	5.99	11.7	16.2	3.70	9.64	10.7	6.46	10.21

算法中，所提算法的平均排名为 1.87，排名高于其他三种算法。

4　结　论

提出一种基于图像分割的稠密立体匹配算法。首先提出基于超像素边缘的匹配代价融合计算方法，为后面迭代提供良好的初始视差图。经过基于视差图和超像素的匹配代价更新、GF 滤波和视差后处理循环迭代后，视差图的误匹配率有所降低。实验结果表明所提算法具有较低的平均误匹配率和较低的遮挡处误匹配率，但是在无纹理和弱纹理区域的匹配精度有待提高。

后续研究方向包括：1)继续研究匹配代价计算方法的融合；2)研究图像分割算法与立体匹配算法相结合的方法；3)对于无纹理区域，所提算法的错误率较高，后续工作着重解决无纹理区域的误匹配问题。

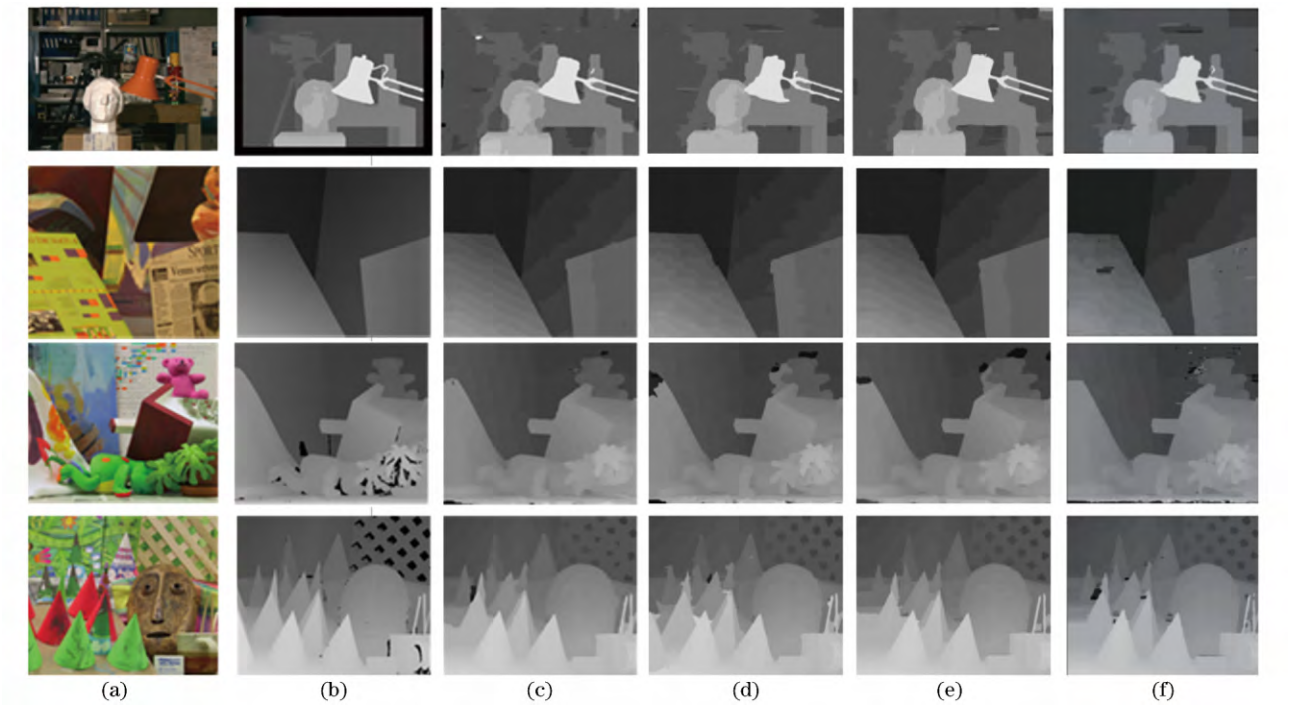

图 6　实验结果。(a)左图像；(b)真实视差图；(c)所提算法结果；(d) MST 算法结果；(e) GF 算法结果；(f) GA-DP 算法结果

Fig. 6　Experimental results. (a) Left image; (b) ground-truth disparity; (c) result of proposed algorithm; (d) result of MST algorithm; (e) result of GF algorithm; (f) result of GA-DP algorithm

表 2 不同算法下 n-occ 的平均误匹配率

Table 2 AvgPBM for different algorithms on n-occ region %

Stereo pair	Proposed	GF	CS-MST	Gray	MST
Tsukuba	1.50_2	1.51_3	2.12_5	1.91_4	1.47_1
Venus	$\mathbf{0.11}_1$	0.20_2	0.84_5	0.32_4	0.25_3
Teddy	$\mathbf{5.27}_1$	6.16_5	7.61_4	5.99_3	5.53_2
Cones	$\mathbf{2.38}_1$	2.71_2	4.10_4	3.70_3	6.01_5
Alone	4.58_2	5.53_4	$\mathbf{4.14}_1$	7.13_5	4.63_3
Art	$\mathbf{7.08}_1$	9.03_2	9.79_3	9.88_4	10.79_5
Baby1	$\mathbf{2.62}_1$	4.69_3	7.37_4	3.24_2	8.39_5
Baby2	$\mathbf{3.30}_1$	6.08_3	11.95_4	4.91_2	13.37_5
Baby3	$\mathbf{3.46}_1$	5.79_4	5.64_3	4.52_2	7.25_5
Books	$\mathbf{8.29}_1$	10.22_3	9.56_2	10.64_5	10.26_4
Bowling1	$\mathbf{6.48}_1$	14.52_3	16.81_4	9.77_2	20.89_5
Bowling2	$\mathbf{4.87}_1$	7.08_3	9.31_4	6.82_2	10.15_5
Cloth1	1.01_3	1.08_4	$\mathbf{0.51}_1$	1.12_5	0.61_2
Cloth2	$\mathbf{2.31}_1$	3.46_3	2.85_2	3.57_4	4.13_5
Cloth3	$\mathbf{1.46}_1$	2.15_3	1.77_2	2.20_4	2.66_5
Cloth4	$\mathbf{3.20}_4$	1.62_2	1.30_1	3.74_5	1.87_3
Dolls	$\mathbf{4.08}_1$	5.04_3	5.00_2	6.57_5	5.95_4
Flowerpots	$\mathbf{9.80}_1$	12.79_2	16.67_4	12.88_3	19.41_5
Lampshade1	$\mathbf{5.59}_1$	11.57_4	10.43_3	6.74_2	11.99_5
Lampshade2	$\mathbf{13.88}_1$	21.13_5	20.88_4	15.04_2	18.20_3
Laundry	15.65_3	16.40_4	13.69_2	18.50_5	$\mathbf{12.94}_1$
Midd1	40.10_4	40.11_5	32.32_2	36.67_3	$\mathbf{27.85}_1$
Midd2	39.24_5	35.85_3	34.50_2	36.93_4	$\mathbf{32.09}_1$
Moebius	$\mathbf{7.44}_1$	9.25_4	7.67_2	9.33_5	8.69_3
Monopoly	25.40_3	27.99_5	$\mathbf{22.51}_1$	27.71_4	24.21_2
Plastic	$\mathbf{33.62}_1$	39.29_2	42.53_4	40.21_3	47.03_5
Reindeer	27.57_4	$\mathbf{7.23}_1$	9.15_2	28.36_5	9.87_3
Rocks1	3.68_4	2.70_2	$\mathbf{2.23}_1$	4.41_5	2.83_3
Rocks2	2.04_3	1.61_2	$\mathbf{1.57}_1$	2.58_5	2.08_4
Wood1	3.77_2	$\mathbf{2.83}_1$	8.68_4	4.73_3	11.06_5
Wood2	2.27_2	2.34_3	$\mathbf{0.99}_1$	3.00_4	5.61_5
AvgErr	**9.42**	10.25	10.44	10.74	11.23
AvgRank	**1.87**	3.06	2.61	3.67	3.65

参 考 文 献

[1] Fan H R, Yang F, Pan X R, *et al*. Stereo matching algorithm for improved Census transform and gradient fusion[J]. Acta Optica Sinica, 2018, 38(2): 0215006.
范海瑞, 杨帆, 潘旭冉, 等. 一种改进 Census 变换与梯度融合的立体匹配算法[J]. 光学学报, 2018, 38(2): 0215006.

[2] Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms [J]. International Journal of Computer Vision, 2002, 47(1/2/3): 7-42.

[3] Taniai T, Matsushita Y, Sato Y, *et al*. Continuous 3D label stereo matching using local expansion moves [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018, 40(11): 2725-2739.

[4] Li J J, Ma L, Wang A X, *et al*. Stereo matching algorithm based on improved patchmatch and slice sampling particle belief propagation[J]. Journal of Northeastern University (Natural Science), 2016, 37(5): 609-613.
李晶皎, 马利, 王爱侠, 等. 基于改进 Patchmatch 及切片采样粒子置信度传播的立体匹配算法[J]. 东北大学学报(自然科学版), 2016, 37(5): 609-613.

[5] Zhu S P, Li Z. A stereo matching algorithm using improved gradient and adaptive window[J]. Acta Optica Sinica, 2015, 35(1): 0110003.

祝世平，李政. 基于改进梯度和自适应窗口的立体匹配算法[J]. 光学学报，2015，35(1)：0110003.

[6] Zhu S P，Yan L N，Li Z. Stereo matching algorithm based on improved Census transform and dynamic programming[J]. Acta Optica Sinica，2016，36(4)：0415001.

祝世平，闫利那，李政. 基于改进 Census 变换和动态规划的立体匹配算法[J]. 光学学报，2016，36(4)：0415001.

[7] Liang Z F，Feng Y L，Guo Y L，*et al*. Learning deep correspondence through prior and posterior feature constancy[C]. IEEE Conference on Computer Vision and Pattern Recognition，2017：2403-2411.

[8] Xiao J P，Tian H，Zou W T，*et al*. Stereo matching based on convolutional neural network[J]. Acta Optica Sinica. 2018，38(8)：0815017.

肖进胜，田红，邹文涛，等. 基于深度卷积神经网络的双目立体视觉匹配算法[J]. 光学学报，2018，38(8)：0815017.

[9] Mayer N，Ilg E，Häusser P，*et al*. A large dataset to train convolutional networks for disparity，optical flow，and scene flow estimation[C]. IEEE Conference on Computer Vision and Pattern Recognition，2016：4040-4048.

[10] Pang J H，Sun W X，Ren J S，*et al*. Cascade residual learning：a two-stage convolutional neural network for stereo matching[C]. IEEE International Conference on Computer Vision Workshops，2017：878-886.

[11] Rhemann C，Hosni A，Bleyer M，*et al*. Fast cost-volume filtering for visual correspondence and beyond[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2013，35(2)：504-511.

[12] Yang Q X. A non-local cost aggregation method for stereo matching[C]. IEEE Conference on Computer Vision and Pattern Recognition，2012：1402-1409.

[13] Zhang K，Fang Y Q，Min D B，*et al*. Cross-scale cost aggregation for stereo matching[C]. IEEE Conference on Computer Vision and Pattern Recognition，2014：1590-1597.

[14] Liu Y，Li Q W，Huo G Y，*et al*. Local binary description combined with superpixel segmentation refinement for stereo matching[J]. Acta Optica Sinica，2018，38(6)：0615003.

刘艳，李庆武，霍冠英，等. 结合局部二进制表示和超像素分割求精的立体匹配[J]. 光学学报，2018，38(6)：0615003.

[15] Gong W B，Gu G H，Qian W X，*et al*. Stereo matching algorithm based on image segmentation and adaptive support weight[J]. Acta Optica Sinica，2015，35(s2)：s210002.

龚文彪，顾国华，钱惟贤，等. 基于图像分割和自适应支撑权重的立体匹配算法[J]. 光学学报，2015，35(s2)：s210002.

[16] Achanta R，Shaji A，Smith K，*et al*. SLIC superpixels compared to state-of-the-art superpixel methods[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2012，34(11)：2274-2282.

[17] Criminisi A，Perez P，Toyama K. Region filling and object removal by exemplar-based image inpainting[J]. IEEE Transactions on Image Processing，2004，13(9)：1200-1212.

[18] Guo H，Ono N，Sagayama S. A structure-synthesis image inpainting algorithm based on morphological erosion operation[C]. Congress on Image and Signal Processing，2008：530-535.

[19] Jiao A S M，Tsang P W M，Poon T C. Restoration of digital off-axis Fresnel hologram by exemplar and search based image inpainting with enhanced computing speed[J]. Computer Physics Communications，2015，193：30-37.

[20] Mei X，Sun X，Zhou M C，*et al*. On building an accurate stereo matching system on graphics hardware[C]. IEEE International Conference on Computer Vision Workshops，2011：467-474.

[21] Yoon K J，Kweon I S. Adaptive support-weight approach for correspondence search[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2006，28(4)：650-656.

[22] Scharstein D，Szeliski R. Middlebury stereo vision page[EB/OL]. (2017-11-15)[2018-12-03]. http://vision.middlebury.edu/stereo/.