第 34 卷 第 1 期 计 算 机 工 程 2008 年 1 月
Vol.34 No.1 Computer Engineering January 2008
·人工智能及识别技术· 文章编号：1000—3428(2008)01—0198—03 文献标识码：A 中图分类号：TP18

噪声对基于不同梯度算子的 ICG 算法的影响

张 培，吴亚锋
(西北工业大学数据处理中心，西安 710072)

摘 要：反向合成梯度算法是一种基于局部指向性的反向合成图像对齐算法，能有效克服光照变化对匹配结果准确性的影响。局部指向性的计算在本质上是梯度的计算，可以用不同的梯度算子求解。该文采用 4 种梯度算子计算局部指向性，通过给输入图像和模板图像中加入噪声模拟实际图像，研究了噪声对基于 4 种梯度算子的反向合成梯度算法的影响。

关键词：反向合成；图像对齐；局部指向性；反向合成梯度算法；梯度算子

Influence of Noise on ICG Algorithm Based on Different Gradient Operators

ZHANG Pei, WU Ya-feng
(Data Processing Center, Northwestern Polytechnical University, Xi'an 710072)

【Abstract】 The Inverse Compositional Gradient(ICG) algorithm is a novel Inverse Compositional Image Alignment (ICIA) algorithm, which is based on the local orientation. Compared with the ICIA algorithm, the ICG algorithm gives more accurate and reliable matching results under different lighting conditions. The evaluation of the local orientation and gradient is essentially equivalent. Hence four different gradient operators are used to compute the local orientation. White Gaussian noise is added to the input or the template images to imitate those in the real world. Performances of the ICG algorithm using the above four gradient operators under different noise, are compared.

【Key words】 inverse compositional; image alignment; local orientation; inverse compositional gradient algorithm; gradient operators

计算机视觉的许多领域，如图像拼嵌、目标跟踪、运动估计等，都需要用到图像对齐(image alignment)技术。图像对齐是指通过移动、变形模板图像使模板图像与输入图像之间的差值最小化。目前图像对齐的常用方法是比较模板图像与输入图像的像素值并使其最小化，如Lucas-Kanade算法[1]及其改进算法[2-3]。但是实际中的图像往往包含有光照的变化，由此会引起图像中像素值的剧烈变化，从而容易导致Lucas-Kanade算法及其改进算法给出错误的匹配结果。而反向合成梯度算法[4]，通过比较模板图像和输入图像之间局部指向性的差值并使其最小化，很好地解决了光照变化对匹配结果准确性的影响。反向合成梯度算法可以采用不同的梯度算子计算局部指向性，而实际图像中又通常含有噪声，因此，本文主要研究了噪声对基于不同梯度算子的反向合成梯度算法的影响。

1 反向合成梯度算法

反向合成梯度算法是一种基于局部指向性的反向合成图像对齐算法。局部指向性是图像边缘结构的一种非线性表示。它不仅可以给出任意像素点上结构的指向性而且还可以对指向性的可靠性进行评估。

1.1 局部指向性的表示

文献[5]中用

$$G(x)=f(g)\left(g_x/|g|,g_y/|g|\right)^{\mathrm{T}} \tag{1}$$

表示任意像素点 x 的局部指向性，其中，$g=(g_x,g_y)^{\mathrm{T}}$ 为该点的梯度；$f(g)$是规范化函数，对于所有 g 有 $0\leq f(g)\leq 1$。由式(1)可知，局部指向性的计算本质上是图像梯度的计算，因此，在实际中可以根据情况采用不同的梯度算子来计算局部指向性。$f(g)$的作用是突出图像中可能的边缘结构，抑制图像中可能的噪声部分。典型的规范化函数为

$$f(g)=|g|\Big/\left(|g|+\overline{|g|}\right) \tag{2}$$

其中，$\overline{|g|}$是图像中所有像素点处$|g|$的平均值。当$|g|$小于$\overline{|g|}$时，则该像素点可能为图像中的噪声，$f(g)$趋近于零抑制该点；当$|g|>\overline{|g|}$时，则该像素点可能为图像中的边缘结构，$f(g)$趋近于 1 突出该点。文献[5]中还给出了其他形式的 $f(g)$，本文采用式(2)，则 $G(\boldsymbol{x})=\left(g_x,g_y\right)^{\mathrm{T}}\Big/\left(|g|+\overline{|g|}\right)$。

1.2 基于局部指向性的反向合成算法

反向合成图像对齐算法使用如下的更新策略：

$$W(x;p)\leftarrow W(x;p)W(x;\Delta p)^{-1} \tag{3}$$

使得

$$\sum_x\left[I(W(x;p))-T(W(x;\Delta p))\right]^2 \tag{4}$$

最小化。其中，$T(x)$ 为模板图像；$I(x)$为输入图像；$W(x;\ p)$为定义在模板图像与输入图像上的具有参数 p 的扭曲变换。$W(x;\ p)$将模板图像中的像素点 x 映射到输入图像中，即对模板图像中的像素点 x，在输入图像中与之对应的像素点为 $W(x;\ p)$。用像素点的局部指向性 $G(x)$代替像素值，则式(4)变为

$$\sum_x\left[G(I(W(x;p)))-G(T(W(x;\Delta p)))\right]^2 \tag{5}$$

令 $G_I(W(x;p))-G(I(W(x;p)))$，$G_{\mathrm{T}}(W(x;p))-G(T(W(x;p)))$，则式(5)可写为

作者简介：张 培(1982－)，男，硕士，主研方向：数字图像处理，计算机视觉；吴亚锋，教授、博士生导师

收稿日期：2007-01-17 **E-mail：**zhangpeioo@163.com

$$\sum_x \left| G_I(W(x;p)) - G_T(W(x;\Delta p)) \right|^2 \tag{6}$$

即

$$\sum_x \left[(G_{Ix}(W(x;p)) - G_{Tx}(W(x;\Delta p)))^2 + (G_{Iy}(W(x;p)) - G_{Ty}(W(x;\Delta p)))^2 \right] \tag{7}$$

其中，G_{Ix}, G_{Iy} 和 G_{Tx}, G_{Ty} 分别表示 G_I 和 G_T 在 x 和 y 方向上的分量。对式(7)进行一阶泰勒展开得

$$\sum_x \left[(G_{Ix}(W(x;p)) - G_{Tx}(W(x;0)) - \nabla G_{Tx} \frac{\partial W}{\partial p} \Delta p)^2 + \right.$$
$$\left. \sum_x \left[(G_{Iy}(W(x;p)) - G_{Ty}(W(x;0)) - \nabla G_{Ty} \frac{\partial W}{\partial p} \Delta p)^2 \right] \right. \tag{8}$$

假定当 p=0 时，$W(x;p)$将 $\boldsymbol{x}$ 映射至自身，即 $W(x;0)$=x。则式(8)变为

$$\sum_x \left[(G_{Ix}(W(x;p)) - G_{Ix}(x) - \nabla G_{Ix} \frac{\partial W}{\partial p} \Delta p)^2 + (G_{Iy}(W(x;p)) - G_{Iy}(x) - \nabla G_{Iy} \frac{\partial W}{\partial p} \Delta p)^2 \right] \tag{9}$$

其中，∇G_T 为 $G_T(\boldsymbol{x})$ 的梯度；∇G_{Tx} 和 ∇G_{Ty} 为其在 x 和 y 方向上的分量。对 Δp 求导，并令式(9)等于 0 得

$$\Delta p = H^{-1} \sum_x \left[\left(\nabla G_{Tx} \frac{\partial W}{\partial p} \right)^{\mathrm{T}} (G_{Ix}(W(x;p)) - G_{Tx}(x)) + \right.$$
$$\left. H^{-1} \sum_x \left[\left(\nabla G_{Ty} \frac{\partial W}{\partial p} \right)^{\mathrm{T}} (G_{Iy}(W(x;p)) - G_{Ty}(x)) \right] \right. \tag{10}$$

其中，

$$H = \sum_x \left[\left(\nabla G_{Tx} \frac{\partial W}{\partial p} \right)^{\mathrm{T}} \left(\nabla G_{Tx} \frac{\partial W}{\partial p} \right) + \left(\nabla G_{Ty} \frac{\partial W}{\partial p} \right)^{\mathrm{T}} \left(\nabla G_{Ty} \frac{\partial W}{\partial p} \right) \right] \tag{11}$$

2　算法的实验比较及分析

人为给定 $T(x)$的大小，然后在 $I(x)$中任选与 $T(x)$大小相同的区域，分别在该区域和 $T(x)$中的相同位置各选定 3 个标准点。这 6 个点就构成了一个仿射扭曲 $W(x;p)$。用具有一定方差 σ 的高斯白噪声对区域内的 3 个标准点的位置进行扰动($T(x)$中的保持不变)，可以得到不同的 $W(x;p)$。用生成的 $W(x;p)$对 $I(x)$进行扭曲得到 $T(x)$。由于实际图像中的噪声通常是高斯白噪声，因此给图像中加入不同Σ的高斯白噪声来模拟实际图像。考虑如下 3 种情况：(1)仅给 $I(x)$中加噪声；(2)仅给 $T(x)$中加噪声；(3)同时给 $I(x)$和 $T(x)$加噪声。加入噪声后，分别用基于不同梯度算子的反向合成梯度算法求出 $T(x)$在 $I(x)$中的位置，并计算 3 个标准点当前位置与实际位置的 RMS 点分布误差：

$$E_{RMS} = \sqrt{\frac{\sum_{i=1}^{3} \| P_i - P_i' \|_2^2}{2 \times 3}} \tag{12}$$

其中，P_i'表示标准点当前位置的坐标；P_i表示标准点初始位置的坐标。如果经过若干次迭代最终的RMS点分布误差小于 ε，则认为算法是收敛的，反之，则认为是发散的。为了比较 4 种算法在不同噪声下的平均收敛速度，先进行若干次图像对齐的实验，然后对同一 σ，从中选取N组 4 种算法均收敛的实验数据，对其相同迭代步骤上的N组RMS点分布误差求均值。如果其中一种或多种算法在某次实验中发散，则该组数据不能用于算法的比较，应该舍弃。

本文选用如图 1 所示的两幅图像作为输入图像。具体实验参数如下：$T(x)$大小为 100×100 像素，3 个标准点的位置分别为该区域的左上角，右上角和底部的中心点，ε = 1.0 像素，分别采用一阶差分算子、Roberts 算子、Sobel 算子和 Prewitt 算子来计算局部指向性，σ =2, 6, 10，Σ = 4, 8, 16, 32，N=10，迭代次数为 30。由于篇幅所限，在第 2.1 节和第 2.2 节中只给出图 1(1)的实验结果，图 1(2)的结果与图 1(1)类似。


(1)

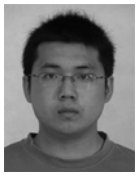
(2)

图 1　输入图像

2.1　仅给 $I(x)$中加噪声

由图 2～图 5 可知，对同一 σ，随着 $I(x)$中噪声程度的加剧，4 种算法达到收敛时所需的迭代次数和 RMS 点分布误差均有所增加。

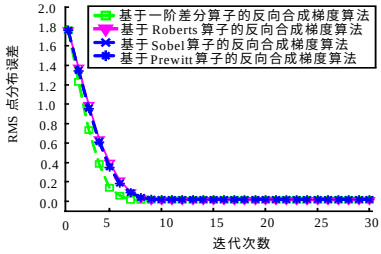

图 2　σ=2,∑$I(x)$=4, ∑$T(x)$=0 时图 1(1)的平均收敛速率结果

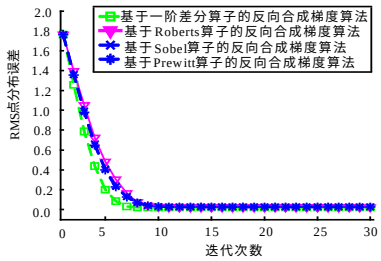

图 3　σ=2,∑I(x)=8, ∑$T(x)$=0 时图 1(1)的平均收敛速率结果

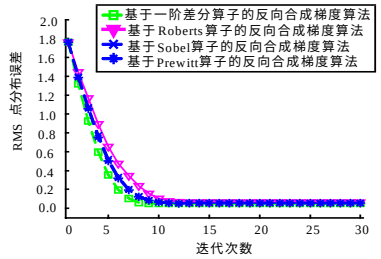

图 4　σ=2,∑$I(x)$=16,∑$T(x)$=0 时图 1(1)的平均收敛速率结果

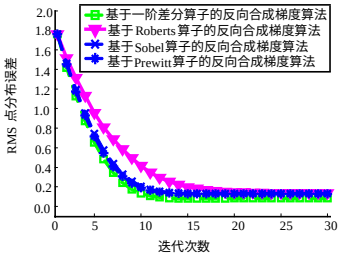

图 5　σ=2,∑$I(x)$=32,∑$T(x)$=0 时图 1(1)的平均收敛速率结果

在图 2～图 5 中，基于 Roberts 算子的反向合成梯度算法的增幅最大，基于 Sobel 和 Prewitt 算子的反向合成梯度算法次之；基于一阶差分算子的反向合成梯度算法的增幅最小。因此，基于一阶差分算子的反向合成梯度算法的收敛性能要优于其他 3 种算法。同时可注意到基于 Sobel 和 Prewitt 算子的反向合成梯度算法有近似的收敛性能(第 2.2 节和第 2.3 节中也有类似的结果)。当 σ =6,10 时，虽然在个别情况下(如

图 1(1)，当 σ =6,10，Σ=32 时)，基于 Sobel 和 Prewitt 算子的反向合成梯度算法的收敛性能接近甚至优于基于一阶差分算子的反向合成梯度算法，但总而言之，后者的收敛性能仍然优于其他 3 种算法。

2.2 给 $T(x)$中加噪声

由图 6～图 9 可知，对同一 σ，随着 $T(x)$中噪声程度的加剧，4 种算法达到收敛时所需的迭代次数和 RMS 点分布误差均有所增加，其中，基于一阶差分算子的反向合成梯度算法的增幅最大；基于 Roberts 算子的反向合成梯度算法次之；基于 Sobel 和 Prewitt 算子的反向合成梯度算法的增幅最小。与 $I(x)$中的噪声相比，在相同情况下 4 种算法对 $T(x)$中的噪声更敏感。尤其是基于一阶差分算子的反向合成梯度算法，在有的情况下还出现了发散的趋势(如图 1(2)，当 σ =2，Σ=32 时)。当 σ =6, 10 时，结果与上面的类似。

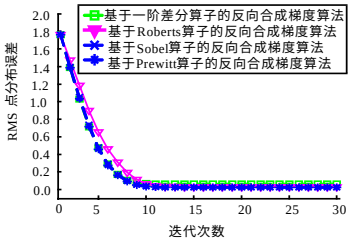

图 6　σ=2, ∑$I(x)$=0, ∑$T(x)$=4 时图 1(1)的平均收敛速率结果

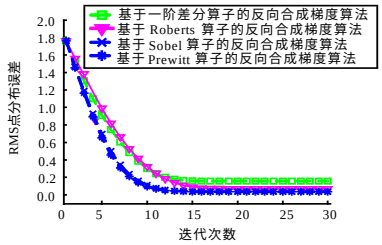

图 7　σ=2, ∑$I(x)$=0, ∑$T(x)$=8 时图 1(1)的平均收敛速率结果

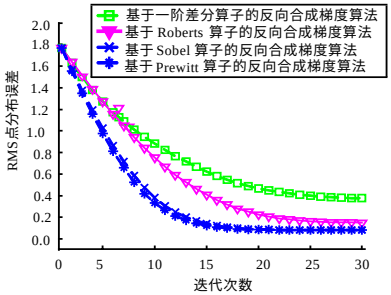

图 8　σ=2, ∑$I(x)$=0, ∑$T(x)$=16 时图 1(1)的平均收敛速率结果

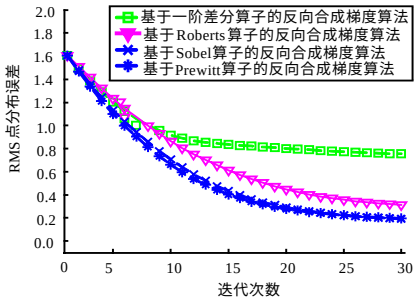

图 9　σ=2, ∑$I(x)$=0, ∑$T(x)$=32 时图 1(1)的平均收敛速率结果

2.3 同时给 $I(x)$和 $T(x)$加噪声

第 2.1 节和第 2.2 节的结论如表 1 表示。由表 1 可知，无论是 $I(x)$中含有噪声还是 $T(x)$中含有噪声，基于 Sobel 和 Prewitt 算子的反向合成梯度算法的收敛性能都要优于基于 Roberts 算子的反向合成梯度算法。尽管仅 $I(x)$中含有噪声时，基于一阶差分算子的反向合成梯度算法的收敛性能要优于基于 Sobel 和 Prewitt 算子的反向合成梯度算法，但是由第 2.2 节可知，前者对 $T(x)$中的噪声很敏感。因此，当 $I(x)$和 $T(x)$都含有噪声时，基于一阶差分算子的反向合成梯度算法的收敛性能将会急剧下降。由此可得如下结论：当 $I(x)$和 $T(x)$都含有噪声时，基于 Sobel 和 Prewitt 算子的反向合成梯度算法的收敛性能最好。算法实验验证了本文的推论，实验结果如图 10～图 13 所示。

表 1　噪声对基于 4 种梯度算子的反向合成梯度算法的影响(强至弱)

仅 $I(x)$中含有噪声	仅 $T(x)$中含有噪声
Roberts 算子	一阶差分算子
Sobel 和 Prewitt 算子	Roberts 算子
一阶差分算子	Sobel 和 Prewitt 算子

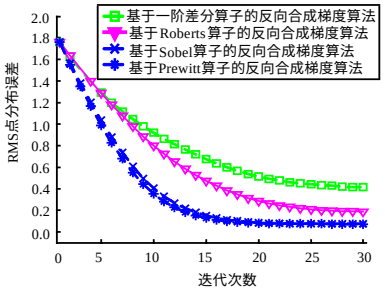

图 10　σ=2, ∑$I(x)$=8, ∑$T(x)$=16 时图 1(1)的平均收敛速率结果

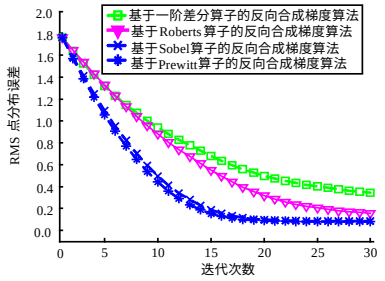

图 11　σ=2, ∑$I(x)$=4, ∑($T(x)$=8 时图 1(1)的平均收敛速率结果

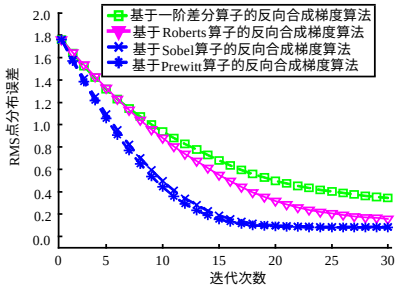

图 12　σ=2, ∑$I(x)$=4, ∑$T(x)$=8 时图 1(2)的平均收敛速率结果

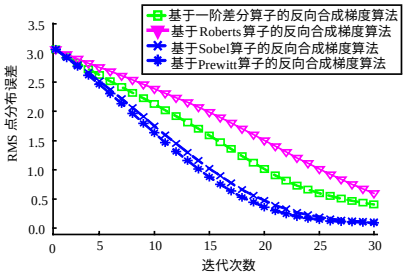

图 13　σ=2, ∑$I(x)$=4, ∑$T(x)$=8 时图 1(2)的平均收敛速率结果

（下转第 203 页）