基于水平线插值的图像修复算法

汪 潇,汪继文
(安徽大学 计算机学院,安徽 合肥 230039)

摘 要:介绍用水平线插值的方法进行图像修复。讨论了基于 Euler 弹性的图像水平线模型,并用这个模型作为图像水平线的局部插值核。其次用星型插值的方法,从破损区域边界处出发,由外向内、一环一环地对水平线进行插值,把水平线逐步地延伸进去,让修复区域逐步缝合起来。该文的算法速度快,能保持强边缘,而且结构更合理,重构的水平线不再只是直线,而是光滑曲线。

关键词:图像修复;水平线;Euler 弹性;星型插值
中图分类号:TN911.73;TP18 **文献标识码**:A **文章编号**:1673-629X(2007)09-0061-04

Digital Image Impainting Using Level Line Interpolation

WANG Xiao, WANG Ji-wen
(School of Computer, Anhui University, Hefei 230039, China)

Abstract: Discusses the impainting method using level line interpolation. Firstly it discusses the level line model based on the Euler's elastic, which will be used as the local interpolation kernel to extend the level lines. Then it uses the star-shaped interpolation method to interpolate the level lines layer by layer, from the boundary of the impainting region into it. Thus the level lines can be smoothly extended into the corrupted region, and the impainting domain sews up step by step. The algorithm is very fast and can preserve strong edges, and the reconstructed structure is more natural.

Key words: image impainting; level line; Euler's elastic; star-shaped interpolation

0 引 言

图像修复(image impainting)是指对一个数字图像的指定区域的缺损数据进行填补修复,其内容包括从图像中剔除指定物体、恢复图像作品中的受损部分等。目标是使得修正的区域和整个图像结合得天衣无缝,或修正后的图像是原图像的真实还原。图像修复在文物的修复与保护、影视特技制作、虚拟现实等方面有着广泛需求,是当前图像处理和计算机视觉研究的热点之一。

图像修复是一个典型的病态问题,先验模型在求解中起关键作用。对单幅图像修复问题,根据视觉心理学中的 Gestalt 效应,自然图像的连续性和光滑性是建立先验模型的基础。文中只讨论单幅非纹理图像的修复。图像修复的应用包括:旧图画的修复,有颜色剥落的旧照片的修复,有划伤的旧电影的修复,旧书报上的笔迹、图片上印的标题文字等的去除,压缩图像和视频的编码纠错(error concealment)等。基于偏微分方程或扩散的修复方法,如文献[1,2]等,速度太慢,不能产生强边缘。基于水平线的方法[3]只能用直线连接水平线。文中用水平线插值的方法进行图像修复。主要思想是:对破损区域由外向内、一环一环地进行插值,把水平线逐步地延伸进去,让修复区域逐步缝合。首先将讨论基于 Euler 弹性[4]的图像水平线模型:

$$E_1(u) = \int |\nabla u| (\alpha + \beta(\mathrm{div}(\frac{\nabla u}{|\nabla u|}))^2)\mathrm{d}x$$

并以它为依据建立局部水平线插值核。接着介绍了星型插值的方法,并对它稍作改进,用它把局部插值算法组织起来,就可以实现对破损区域逐步地、一环一环地进行插值。试验表明,文中的算法鲁棒性好,能保持强边缘,产生的结构更合理、重构的水平线都是光滑曲线。而且该算法速度很快,修复一个一般区域耗时一般在0.5s左右。

1 水平线模型

记 $u(x)$ 为图像的灰度函数,则图像的水平线定义为水平集 $X_\lambda u = \{x, u(x) \geqslant \lambda\}$ 的边界。总变差模

收稿日期:2006-10-23
基金项目:安徽省高校自然科学研究项目(2006KJ028B)
作者简介:汪 潇(1983-),男,安徽歙县人,硕士研究生,研究方向为计算机模拟与仿真;汪继文,教授,博士后,博士生导师,研究方向为智能计算。

型[2]假设图像可以表示成一个有界变差函数 u，使得 $\int_{R^2}|u|\mathrm{d}x<\infty$，并把 u 的总变差定义为：

$$TV_{R^2}(u)=\int_{R^2}|\nabla u|=\sup\{\int_{R^2}u\mathrm{div}\phi\mathrm{d}x:\phi\in C_c^1(R^2,R^2),|\phi|\leqslant 1\}<\infty \tag{1}$$

设 $E\subset R^2$ 是一个 Lebesgue 可测集，如果 $P(E):=TV_{R^2}(1^E)<\infty$，则称它是边长有限的，其中 1^E 表示 E 的特征函数。而且，如果 E 具有 Lipschitz 边界，则 $P(E)=H^1(\partial E)$，其中 H^1 为 Hausdorff 测度。根据联合面积公式(Coarea Formula)：

$$TV_{R^2}(u)=\int_{-\infty}^{+\infty}P(X_\lambda u)\mathrm{d}\lambda \tag{2}$$

就是说，图像 u 的总变差就等价于它的所有水平线的长度之和。既然两点间的直线段距离最短，使总全变差模型最小就要用直线来连接水平线。所以基于总变差模型的图像修复算法[2]的几何意义就是用直线连接水平线。Euler 弹性模型[4]是一个高阶曲线模型：

$$e(\Gamma)=\int_\Gamma(\alpha+\beta\kappa^2)\mathrm{d}s=\alpha\mathrm{length}(\Gamma)+\beta\int_\Gamma\kappa^2\mathrm{d}s \tag{3}$$

其中 κ 为曲线的曲率，$\mathrm{d}s$ 为弧长单位，α 和 β 为权重参数。现在用 Euler 弹性模型来刻画图像的水平线。记 λ 为图像 u 的灰度水平，Γ_λ 为对应灰度 λ 的水平线，则图像中所有水平线的 Euler 弹性能量之和为：

$$E(u)=\int_\lambda\int_{\Gamma_\lambda}1\mathrm{d}s\mathrm{d}\lambda$$

由式(1)：

$$\int|\nabla u|\mathrm{d}x=\int_\lambda\int_{\Gamma_\lambda}1\mathrm{d}s\mathrm{d}\lambda$$

将其代入式(3)，并注意到：

$$\kappa=\mathrm{div}(\frac{\nabla u}{|\nabla u|})$$

得到：

$$E_1(u)=\int|\nabla u|(\alpha+\beta(\mathrm{div}(\frac{\nabla u}{|\nabla u|}))^2)\mathrm{d}x \tag{4}$$

这就是基于 Euler 弹性的图像水平线模型。

2 水平线插值

那么如何具体地在每一个像素点处延伸水平线，怎样估计它的像素值。这里借助局部 Markov 随机场的方法进行描述。设图像区域为 S，$P(u)$ 为图像 u 分布函数。记 u_i 为图像在 $i\in S$ 上的取值，而 $u_{s\setminus i}$ 为 u 在除 i 之外的所有点上的取值。另外，记 $u_{\partial i}$ 为 u 在 i 一个空心邻域上的取值。如果考虑条件分步 $P(u_i|u_{s\setminus i})$，$\{P(u)\}$ 就是一个 Markov 随机场。这里只需要考虑局部 Markov 随机场，即每个像素只依赖于它的相邻的几个像素，故令：

$$P(u_i|u_{s\setminus i})\equiv P(u_i|u_{\partial i})$$

上式中需要考虑的有两个要素：其一是合适的局部邻域 $u_{\partial i}$，其二是分布函数 P。设 Ω 为图像的待修复区域，$\partial\Omega$ 为其边界。对修复区域内一点 i，定义 $u(i)$ 为以 i 为中心的一个 7×7 邻域。这个邻域中可能有的点灰度值是已知的，有的是未知的。记已知的那一部分为 $u_{\mathrm{valid}}(i)$，并取 $\partial i=U_{\mathrm{valid}}(i)$。如图1所示，$i$ 是中心像素，灰色区域是灰度值已知的区域，i 和白色区域像素都是未知的，箭头则代表水平线方向。要在这个邻域用水平线的方法估计出 i 的像素值。

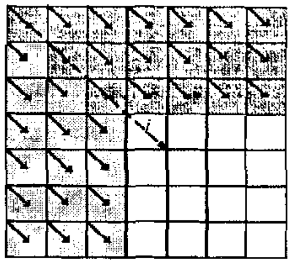


图1 水平线插值示意图

定义 $P(u_i|u_{\partial i})\propto\exp\{-E_1(u|_{U_{\mathrm{valid}}(i)\cup\{i\}})\}$，文中将通过最大 $P(u_i|u_{\partial i})$ 来估计 $u(i)$。为了最大化 $P(u_i|u_{\partial i})$，只要最小化 $E_1(u|_{U_{\mathrm{valid}}(i)\cup\{i\}})$ 就行了。几何意义就是把水平线尽可能直、尽可能光滑地延伸到 i，如图1所示。估计 u_i 的算法步骤如下：

(1) 计算 $u(i)$ 内每一个已知像素点的水平线方向，可由梯度方向经过 $90°$ 旋转得到。

(2) 计算 $u(i)$ 内这些已知像素点的水平线方向的平均值，并把这个方向当作 i 的水平线方向。

(3) 沿这个方向(或反方向)在 $u(i)$ 内查找离 i 最近的几个已知的像素点，计算这几个像素的平均值，作为对 u_i 的估计值。如果计算得到水平线方向与已知区域和未知区域之间的边界平行，这时沿这个方向将查找不到已知的像素点，i 的像素值也无法确定，只好将这样的点暂时搁置起来，待到下一轮再对它进行插值(星型插值的下一轮，后文将论述)。另外，如果计算得到的水平线方向为零(或接近于零)，说明 i 处在一个像素值比较一致的平滑区，这时可以让取其局部已知像素的平均值，也可以暂时搁置 i，待到下一轮再对它进行插值。

之所以计算局部邻域内水平线方向的平均值，作为对 i 的水平线方向的估计，原因有二：(1) 为了提高算法的鲁棒性，在自然图像中，只在一点用梯度旋转的方法求水平线方向将会很不准确；(2) 为了使其解在一个邻域内是最优的，而不是仅仅对一个点最优。如果

只对一个点是最优的,那么水平线就会只是直线延伸。而如果像这样在一个邻域计算水平线方向的均值,作为对 i 的水平线方向的估计,这个方向在这个邻域内就是调和的,沿这个方向作水平线延伸也不再是简单的直线延伸。

3 星型插值

这一部分介绍星型插值[5]的方法,并将其作一些改进,应用到文中的算法中。对区域 D,如果其中存在一点 q,使得连接 D 中每一点 z 和 q 的线段都包含在 D 内,则称 D 为一个星型区域,且称 q 为其可视点。设 S 是图像区域,u 是图像的灰度函数。假设 $D \subset S$ 是一个星型区域,q 是它的可视点。以 q 为原点建立坐标系。取 $0 < \alpha < 1$,定义 $D_k = \alpha^k D$,并记 $u_k := u|_{\partial D_k}$,其中 ∂D_k 为 D_k 的边界,而 $\partial D_0 = \partial D$。显然 ∂D_{k+1} 是包含在 D_k 中的。只要 ∂D_{k+1} 离 ∂D_k 足够近($k = 0,1,2,\cdots$)就可以从 u_0 插值 u_1,从 u_1 插值 u_2,继续一直到整个 D 都被插值完毕。而插值核是依赖于先验模型的。这个插值方法就叫星型插值,如图 2 所示。

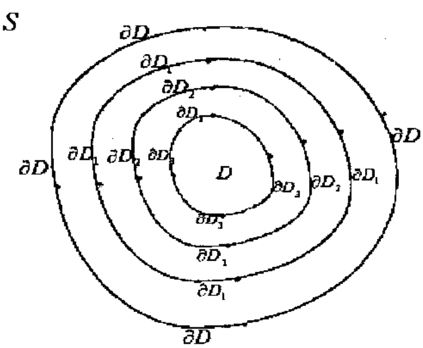


图 2　星型插值示意图

通常图像中的待修复区域不一定是星型区域,所以有必要把这个插值方法推广到一般的情形,放松星型区域这个条件,对一个一般区域也实行这样从外向里一环一环地进行插值。这样就不能再通过以可视点为极点的极坐标来确定环线 $\partial D_k, \partial D_{k+1}\cdots$ 了。文中采用从区域边界往里逐步收缩的方法。

设 Ω 是图像中的待修复区域,$\partial\Omega$ 为其边界。由于待修复区域是已知的,因此可以用一个 mask 图像来记录待修复区域,它在待修复区域灰度值为 1,而其它地方的灰度值为 0。由这个 mask 图像膨胀一个像素后减去它自己,就可以得到关于修复区域的一个像素宽的外边界线 $\partial\Omega_0 = \partial\Omega$。图像在这条外边界上的数据是已知的。推广后的星型插值算法步骤如下:

(1) 用 mask 减去对它腐蚀一个像素后的结果,得到 $\partial\Omega_1$。它是紧贴 $\partial\Omega_0$ 内侧的一条环线,一个像素宽,原图像在这条闭合曲线上的数据已丢失,各个像素值都未知。

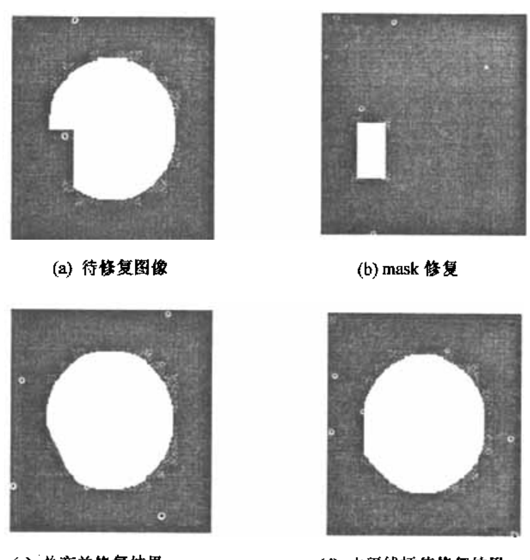
(2) 用第 2 节介绍的局部插值方法计算 $u|_{\partial\Omega_1}$。$\partial\Omega_1$ 上的各个点有的用这个方法可以估计出像素值,有的则不能(下文解释)。能求出像素值的点,将其在 mask 中对应位置上的点置为零,表明这个点的值已求出,下一轮插值时,这个点就成了已知点。而不能估计出灰度值的点,则暂时搁置,mask 也不变。

(3) 对新的 mask 用(1)中的方法计算 $\partial\Omega_2$,再用与(2)同样的方法计算 $u|_{\partial\Omega_2}$,这样一直下去,直到整个区域被插值完毕。

对 $\partial\Omega_k$ 上的像素 i,根据第 2 节中的局部插值算法,如果计算得到水平线方向与 $\partial\Omega_k$ 在这个局部的方向平行,则 i 的像素值无法确定,这样的点暂时留着。如果计算得到的水平线方向为零(或接近于零),说明 i 处在一个像素值较一致的平滑区,这时可以让 i 取局部已知像素的平均值,也可以暂时搁置 i。

4 试验结果与算法分析

图 3 显示的是在圆盘图像上的修复试验结果。由于采用的是二值图像,图像中除了边缘就是平滑区,边缘就是水平线。这样可以把水平线延伸的效果看得更清楚。图 3(a)是待修复图像;图 3(b)是 mask 图像,用以标出待修复区域;图 3(c)是用总变差方法修复的结果,圆盘的边缘被破坏的部分经过修复后用直线连接起来了;图 3(d)是用文中的算法修复的结果,由于缺少区域内其它水平线之间的相互制约,使得水平线被



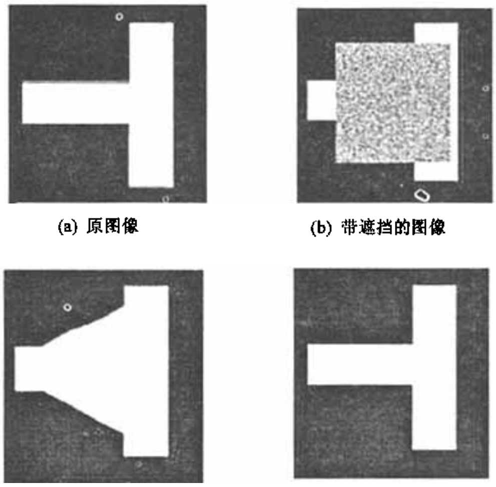
(a) 待修复图像　(b) mask 修复

(c) 总变差修复结果　(d) 水平线插值修复结果

图 3　圆盘图像上的修复试验结果

直线延伸。但由于是从四周一环一环地往里延伸的，修复结果失真并不大，已经比较接近圆盘了。跟全变差扩散修复方法比较，文中算法的效果明显好多了，而且修复速度上是前者无法比拟的。

图 4 显示的是在“T”形图像上的修复试验结果。图 4(a)是原图像；图 4(b)是带遮挡的图像；图 4(c)是用 S. Masnou[3]中的方法去除遮挡的结果，由于用直线连接水平线，结果跟原图比有很大的失真；图 4(d)是用文中的算法修复的结果，这个结果已经和原始图像一模一样了。



(a) 原图像　　(b) 带遮挡的图像

(c) 用 S.Masnou[3]中的方法去除遮挡的结果　　(d) 文中的算法修复的结果

图 4　在“T”形图像上的修复试验结果

图 5 显示了对自然图像的一个较复杂的区域的修复结果。图 5(a)是待修复图像。图 5(b)是用文中算法修复的结果，基本上看不出来修复过的痕迹。用连接水平线的方法根本无法修复这样的区域，因为白色的条子到中间突然断了，使图像变得在区域边界上根本找不到配对的水平线。用扩散的方法将会很模糊，根本得不到这样的效果。如图 5(c)所示为总变差扩散方法修复的结果。

(a) 待修复图像　　(b) 文中算法修复结果　　(c)总变差修复结果

图 5　对自然图像的一个较复杂的区域的修复结果

图 6 显示了 lady 图像上的修复试验。这些试验结果充分地显示了文中的算法是一个行之有效的修复算法。跟扩散的方法比，文中的算法有明显的速度上的优势，上述用该算法进行的修复试验耗时一般在 0.5s 左右，这跟扩散的方法往往需要几分钟比，几乎可以忽略。文中的算法对较大的区域也能修复，而扩散的方法往往只能修复较小的区域。较复杂的区域，扩散的算法就达不到好的效果，文中的算法对较复杂的区域也有好的效果。

(a) 损坏的图像　　(b) 修复过后还原的图像

图 6　对 lady 图像的修复试验

跟分块重构的方法比，文中的算法不需要检测边缘。自然图像并不一定都是分块平滑的，还有一些拓扑结构较复杂的区域不能看作分块平滑函数，分块重构的方法对这样的图像不一定合适。文中的算法对修复区域的拓扑结构没有什么限制。

文中的算法属于水平线的方法，它比 S. Masnou[3]的基于水平线的算法鲁棒性更好，而且重构的结构更合理。无需逐个计算到达修复区域的边界的所有的水平线，这使算法更简单、更快，鲁棒性更好。S. Masnou 只考虑了水平线的端点，而没有考虑其方向，而且用直线连接水平线，这样重构有时会产生较大的失真。文中考虑了水平线到达修复区域边界的方向，并将水平线沿这个方向光滑地延伸进去，重构的结构更合理。

参考文献：

[1] Bertalmo M, Sapiro G, Caselles V, et al. Image Impainting [C]//Computer Graphics, SIGGRAPH '00. [s.l.]: [s.n.], 2000: 417 – 424.

[2] Chan T, Shen J. Mathematical Models for Local Non – texture Impainting[J]. SIAM J Appl Math, 2001, 62(3): 1019 – 1043.

[3] Masnou S. Disocclusion: A Variational Approach Using Level Lines[J]. IEEE Trans Image Processing, 2002, 11(2): 68 – 76.

[4] Mumford D. Elastic and Computer Vision[M]. New York: Springer – Verlag, 1994: 491 – 506.

[5] Rudin L, Osher S, Fatemi E. Nonlinear Total Variation Based Noise Removal Algorithms[J]. Nonlinear Phenomena, 1992, 60(1/4): 259 – 264.