基于 Level Set 方法的曲线演化*

杨 猛, 汪国平, 董士海

(中国科学院 计算技术研究所 智能信息处理开放实验室,北京 100080);
(北京大学 计算机科学技术系,北京 100871)
E-mail: yang@graphics.pku.edu.cn
http://graphics.pku.edu.cn

摘要: Level Set 方法是一种描述曲线以曲率相关的速度演化的有力工具,最近几年在医学图像处理、自然现象的模拟以及计算机视觉等领域得到了广泛的应用.其中,曲线演化后的平滑算法和轮廓跟踪方法是 Level Set 方法实际应用中的两个关键算法.给出了一种平滑 Level Set 距离函数的简单方法.该方法只采用内插值的方式,消除平面上的全部孤立点以及部分可能产生歧义的点,在允许存在部分冗余点的情况下,利用曲线轮廓跟踪算法,得到平面上所有曲线的轮廓.经过实验验证,该方法简单、高效,适应范围广.

关 键 词: Level Set 方法;曲线演化;医学图像处理;轮廓跟踪;基于图像的建模
中图法分类号: TP391 文献标识码: A

Level Set 方法是 Sethian 在研究曲线以曲率相关的速度演化时提出来的[1],用于描述曲线(或曲面,以下对曲线的讨论均适用于曲面,故略去曲面)的演化过程.Level Set 方法的基本数学思想是,将当前正在演化的曲线看做是一个更高维函数的 Level Set,利用曲线演化与 Hamilton-Jacobi 方程的相似性[2],给出了一种曲线演化的强鲁棒性的计算方法.

自从 Osher 和 Setian[1]在 1988 年提出 Level Set 方法以来,该方法已在图像处理(如图像噪音消除、医学图像中静脉血管的形状检测等)[3,4]、晶体生长[5]、流体的模拟[6]以及计算机视觉[7,8]等领域得到了广泛的应用.尤其是在基于图像的建模方面,Faugeras 提出了一种利用曲面演化恢复物体几何模型的方法[7],可以很好地恢复漫反射物体的三维模型.采用与 Faugeras 类似的方法,我们在将 Level Set 方法应用于 IBM(image based modeling)时,发现了一些与 Level Set 相关的问题.本文讨论了其中的两个核心问题——曲线轮廓跟踪和曲线平滑.

Level Set 方法与其他曲线演化方法相比,最大的优势在于它的稳定性以及拓扑无关性.曲线在演化过程中可能会产生尖点、断裂为多条曲线或者两条或多条曲线融合为一条.Level Set 方法可有效地处理这些情况.但正如 Sethian 在文献[1]中所指出的,Level Set 方法有一个缺点,即计算量太大,因为它将二维的问题扩展到三维、三维的问题(曲面演化问题)扩展到了四维,维数的扩展增加了计算复杂度,平面曲线演化算法的计算复杂度为 $O(N^2)$,三维曲面的计算复杂度为 $O(N^3)$,其中 N 是将平面(或空间)均匀离散成网格点后,水平方向的网格点数目.

为此,Sethian 提出了窄带(narrow band)Level Set 方法[9].这种方法的基本思想是,在曲线轮廓的周围建立一个自适应的窄带,每次演化时,只更新窄带内的网格点的函数值.在演化过程中,为防止曲线的点跨越窄带,需要存储窄带的内外边界,当曲线上的点接近内外边界时,再重新建立一条以当前曲线为中心的宽度为 k 的窄

* 收稿日期: 2002-02-28; 修改日期: 2002-06-18
基金项目: 国家自然科学基金资助项目(60173016);国家高技术研究发展计划资助项目(2001AA115126);国家教育部骨干教师基金资助项目;北京市自然科学基金资助项目(4012008)

作者简介: 杨猛(1975－),男,山东淄博人,博士,讲师,主要研究领域为虚拟现实;汪国平(1964－),男,浙江嵊州人,博士,副教授,主要研究领域为计算机图形学,CAD,虚拟现实,多媒体;董士海(1939－),男,浙江镇海人,教授,博士生导师,主要研究领域为计算机软件,图形学,人机交互.

带,Sethian 将这一步称为重新初始化(reinitialize).通过这种方式,可以将计算复杂度降至 $O(Nk)$,其中 k 是窄带的宽度.

在窄带方法中,核心内容是窄带的动态更新以及距离函数的重新初始化过程.Sethian 的重新初始化过程需要先取得曲线的轮廓,而由于无论采用何种方式更新距离函数都会带来误差,所以在寻找曲线轮廓之前,必须对离散网格点的距离函数进行平滑.Sethian 采用内、外插值的方法进行平滑,能够消除所有的歧义点(距离值不精确的点),但耗时较长.Peng[10]提出了一种基于 PDE 的快速局部 Level Set 方法(PDE based fast local level set method).这种方法不需要得到曲线的轮廓,只需利用局部网格点的距离函数值就可以重新估算各点距曲线的最短距离,所以,该方法得到了广泛的应用.但是,这种方法对曲线演化的速度有一定的限制,要求每次演化时,曲线上点的运动速度不会超过一个网格点.

Level Set 方法要求平面上所有的点(或窄带内的点)都有运动速度,但在一些实际应用中[7,8],只有曲线上的点才有演化速度,并且运动速度可能并不仅仅与曲线的曲率有关,有时必须取得曲线的轮廓才能计算点的运动速度,然后将曲线上点的速度扩展到整个平面(或窄带内).所以,平滑算法与曲线轮廓跟踪算法的好坏直接影响到曲线演化的效率.

本文的平滑算法与 Sethian 的方法不同之处在于,只采用内插值的方式,消除所有的孤立点(关于孤立点的定义,见下文)和引起歧义的外部点,在保留一些可能引起歧义的边界点的情况下,利用第 3 节将要介绍的曲线轮廓跟踪算法,得到曲线演化后的轮廓.由于平滑的过程并不要求消除所有的歧义点,所以,算法相对简单,速度快而且易于实现.

本文第 1 节是对 Level Set 方法的简介.第 2 节重点介绍距离函数的平滑算法.曲线轮廓的跟踪算法在第 3 节中加以论述.最后是实验结果和结论.

1 Level Set 方法

给定平面上的一条封闭曲线,以曲线为边界,把整个平面划分为曲线的外部区域和内部区域.在平面上定义一个距离函数 $Z=\pm\phi(x,y,t)$,其中 Z 值是点(x,y)到曲线的最短距离,函数的符号取决于该点在曲线的内部还是外部,一般地,定义在曲线内部点的距离为负值;t 表示时间.这样,在任意时刻,曲线上的点就是距离函数值为 0 的点(即距离函数的 Zero Level Set).图 1 是平面上两个圆的 Level Set 表示,平面上所有点的距离值构成两个相交的空心锥体,xy 平面与两个锥体的交线就表示两个圆的轮廓.因为在演化的过程中,曲线上的点始终都满足下面的方程:

$$\phi(x,y,t)=0\,. \tag{1}$$

方程(1)的两边对时间 t 求导,则可以得到

$$\phi_t+\phi_x\frac{\mathrm{d}x}{\mathrm{d}t}+\phi_y\frac{\mathrm{d}y}{\mathrm{d}t}=0\,. \tag{2}$$

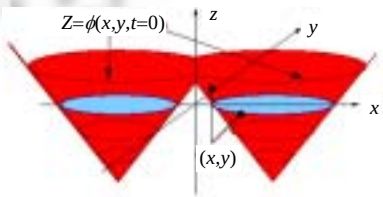

Fig.1 Level Set representation for two circles (from Ref.[9])
图 1 两个圆的 Level Set 表示(来自文献[9])

假设曲线上各点的运动速度为 F,因为我们限制曲线上各个点的运动方向是沿着曲线的法线方向,也就是曲线上各点的梯度方向,所以,整个曲线的演化可以用下面的方程[1,9]表示:

$$\phi_t-F|\nabla\varphi|=0\,, \tag{3}$$

$$\phi(x,y,t=0)=\pm d\ . \tag{4}$$

式(3)和式(4)构成了带初值的偏微分方程,比初始问题多了一维(时间),这样,曲线演化的问题就转化为一个微分方程求解的过程.尽管问题看似复杂了,但通过引入与时间相关的距离函数,曲线演化中的拓扑改变问题却迎刃而解了.此方程的数值解法有很多种[11],通常可以采用下面一种相对简单的方案.

$$\phi^{n+1}=\phi^{n}-\Delta t(\max(F_{x,y},0)\nabla^{+}+\min(F_{x,y},0)\nabla^{-})\ , \tag{5}$$

其中:

$$\nabla^{+}=\max(D_{x,y}^{-x},0)^2+\min(D_{x,y}^{+x},0)^2+\max(D_{x,y}^{-y},0)^2+\min(D_{x,y}^{+y},0)^2\ , \tag{6}$$

$$\nabla^{-}=\min(D_{x,y}^{-x},0)^2+\max(D_{x,y}^{+x},0)^2+\min(D_{x,y}^{-y},0)^2+\max(D_{x,y}^{+y},0)^2\ , \tag{7}$$

$$D_{x,y}^{-x}=\frac{\phi_{x,y}-\phi_{x-1,y}}{h},\ D_{x,y}^{+x}=\frac{\phi_{x+1,y}-\phi_{x,y}}{h},\ D_{x,y}^{-y}=\frac{\phi_{x,y}-\phi_{x,y-1}}{h},\ D_{x,y}^{+y}=\frac{\phi_{x,y+1}-\phi_{x,y}}{h}\ . \tag{8}$$

这样,曲线演化的问题就转化为函数迭代的过程.采用这个过程可用以下 4 个步骤简单地进行描述:

(1) 构造曲线窄带,初始化窄带内各点到此曲线的最短距离,即 t=0 时刻的函数值.

(2) 根据上面的函数更新机制计算下一时刻的函数值.根据实际问题的需要,此步骤可以重复运行 m(m 值的选择取决于窄带的宽度和曲线演化的速度)次,即从 $t=k$ 一直演化到 $t=k+m$.

(3) 因为数值方法会带来误差,所以需要对演化后平面中各点的函数值进行平滑,以消除平面上的歧义点或冗余点.

(4) 跟踪曲线的轮廓,得到曲线的离散点表示,计算曲线上各点的演化速度,重新构造窄带,初始化窄带内各点到曲线的最短距离(即函数值)及演化速度,然后回到步骤(2).

本文就是针对步骤(3)和(4)展开详细讨论的.关于 Level Set 的详细介绍,可以参阅文献[12].

2 平滑算法

由于存在误差,无论采用何种数值解决方案,每次更新距离函数之后都会存在一些歧义点,这些点的距离值或者不准确,或者不合理,由于存在这些点,可能使轮廓跟踪算法失效,以至于无法得到演化后的曲线轮廓,所以,需要对距离函数进行平滑.为了便于说明,首先定义各种符号的含义,见表 1.

Table 1 Symbol meanings

表 **1** 符号含义

Symbol①	Meaning②
★	Inner point of curve③
■	Outer point of curve④
○	Point in curve⑤
☆	Inner point needing smoothing⑥
□	Outer point needing smoothing⑦
◎	Border point needing smoothing⑧
※	Any type of point⑨

①符号,②含义,③曲线内部的点,④曲线外部的点,⑤曲线上的点,⑥需要平滑的内部点,⑦需要平滑的外部点,⑧需要平滑的边界点,⑨任意类型的点.

图 2 是歧义点的示例,其中的点◎是一个含意模糊的点,也就是说,既可以将它作为曲线上的点,也可以认为它是一个内部点.从曲线平滑的角度来看,将它作为内部点更为合理.曲线平滑的工作主要是消除各种可能引起歧义的点.通过对平面上的点进行分类,曲线平滑的流程可用以下 4 个步骤来描述:

○○○○○○○○
○★★★★★★○
○★◎★★★★○
○○○○○○○○

Fig.2 Ambiguous point example

图 2 歧义点示例

(1) 消除孤立点;

(2) 循环进行消除特殊模式的外部点(内部点)、消除孤立点的操作,直到不再发生变化为止;

(3) 添加边界点;

(4) 循环进行消除特殊模式的边界点、消除孤立点的操作,直到不再发生变化为止.

2.1 消除孤立点

平滑算法中的第 1 步是消除孤立点(在后面的步骤中,需要多次调用消除孤立点的操作).所谓孤立点,实际上可以看做是曲线演化过程中的噪音点,由于这些噪音的存在,导致曲线不平滑或断裂.为方便下面的讨论,定义一个点的“邻居”为与它相临近的 8 个点.定义一个点的“近邻”为它的上、下、左、右 4 个点.

定义下面 5 种类型的点为孤立点:

(1) 如果一个边界点的近邻都是外部点(内部点),则称此边界点为孤立边界点(如图 3(a)所示).需要将边界点变为外部点(内部点).

(2) 如果一个外部点的近邻都是内部点(边界点),则称此外部点为孤立外部点(如图 3(b)所示).此时将外部点变为内部点(边界点).

(3) 如果一个外部点的邻居都不是外部点,则称此外部点为半孤立外部点(如图 3(c)所示).此时根据邻居的平均距离值,将半孤立外部点设为边界点或内部点.

(4) 如果一个内部点的近邻都是外部点(边界点),则称此内部点为孤立内部点(如图 3(d)所示).此时将内部点变为外部点(或边界点).

(5) 如果一个内部点的邻居都不是内部点,则称此内部点为半孤立内部点(如图 3(e)所示).此时根据邻居的平均距离值,将半孤立内部点变为外部点或边界点.

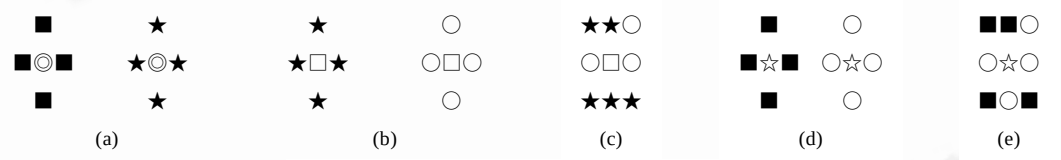

Fig.3 Various kinds of isolated points

图 3 各种类型的孤立点

2.2 消除特殊模式的内部点(外部点)

经过消除孤立点的操作之后,并不能消除所有的歧义点,接下来的步骤是去掉一些特殊模式的外部点和内部点.这些模式的内部点(外部点),实际上也是一些噪音点,这里之所以没有将它们列为孤立点,是因为消除孤立点的操作将在后面多次调用,也就是说,后面的某些操作将会再次导致孤立点的产生,但是不会产生这些特殊模式的内部点(外部点),所以此操作只需进行一次.

如果一个内部点(外部点)的 4 个近邻中有 3 个点均为边界点或者 8 个邻居中有连续 5 个(必须包含某个斜对角线上的两个点)都是边界点,则同样通过内插值的方式将内部点(外部点)变为边界点.

2.3 添加边界点

经过一次演化之后可能会失去一些边界点,即在一些地方,曲线外部点与曲线内部点直接相邻,也就是说,曲线可能在此处发生了断裂,此时需要在外部点和内部点之间添加一个边界点.实际上,在从上面消除孤立点、消除特殊模式的外部点或内部点的操作中,我们已经添加了一些边界点,但是仍然存在外部点与内部点直接相邻的情况.处理这些点的方法很简单,只要比较两个点的距离的绝对值,选择具有较小距离值的点作为边界点即可.需要注意的是,添加边界点的操作必须在上面两个步骤之后,否则会将孤立点或特殊类型点的影响扩大,从而导致错误的结果.图 4 给出的是一种可能导致错误的孤立点的分布情况.图 4(a)中的两个外部点是两个孤立外部点,合理的平滑结果应该是将这两个孤立点变为曲线内部点,但是如果先进行添加边界点操作,则一种可能的结果即如图 4(b)所示,很多内部点被错误地平滑为边界点.如果此后再进行消除孤立点及特殊模式的内部点

的操作,则结果如图 4(c)所示.

(a) Before adding border point (b) After adding border point (c) Delete isolate and special inner points
(a) 添加边界点之前 (b) 添加边界点之后 (c) 消除孤立点及特殊内部点

Fig.4 Isolate points possibly resulting in error
图 4 可能导致错误的孤立点

2.3.1 消除特殊模式的边界点

经过上面 3 种操作之后,不但消除了各种孤立点,而且也增加了一些边界点.为了能够得到曲线的轮廓,还必须消除一些冗余的边界点.在实际应用中,根据轮廓跟踪算法的需要,并不一定消除全部冗余的边界点,只需消除两种特殊模式的边界点即可.为了方便讨论,我们将此边界点称为中心边界点,并对边界点的 8 个邻居进行编号,编号的方式如图 5 所示.

2.3.2 被特定模式外部或内部点包围的边界点

如果从偶数编号开始的 5 个连续邻近点都是外部点(内部点),而另外一个偶数编号的邻近点又是边界点,则此中心边界点可以看做是一个冗余的边界点,此时可以将它变为外部点(内部点).例如,如果 0,1,2,3,4 都是外部点,而 6 是边界点,则可以将当前的边界点变为外部点.事实上,共有 8 种可能的模式(外部点、内部点各 4 种).图 6 给出了 4 种外部点的模式.

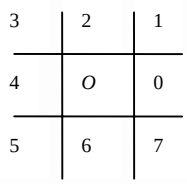

Fig.5 Code of adjacent points
图 5 邻近点的编码

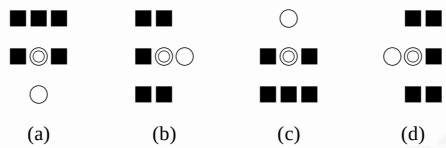

Fig.6 Four special kinds of outer points
图 6 4 种特殊模式的外部点

2.3.3 被特定模式边界点及内部点包围的边界点

如果边界点的某一侧都是边界点,而与该侧平行且包含中心边界点的两个邻近点都是内部点,则可以将中心边界点变为内部点.这样做的目的是使曲线的轮廓相对平滑.如图 7 所示为 4 种可能的模式.

Fig.7 Four possible kinds of border points surrounded by border and inner points of special modes
图 7 4 种被特殊模式边界点和内部点包围的边界点

同样,在消除特殊的边界点之后又可能产生孤立边界点,此时需要再次调用消除孤立点的操作,循环进行这两种操作,直到不再发生变化为止.

3 轮廓跟踪算法

曲线平滑的目的是为了得到曲线的轮廓.当定义曲线的正向为沿此方向行走时,曲线的内部始终都在曲线的左侧.因为在前面的平滑过程中并没有消除所有的冗余边界点,所以在进行曲线轮廓跟踪时,必须正确地判断哪些点属于有效边界点,哪些是冗余点.这种判断是在轮廓跟踪过程中,根据当前曲线的状态,动态地进行的.

首先将平面上的边界点进行分类,如果一个边界点的 8 个邻居中有一个内部点,就把这个边界点称为“种子点”.种子点是可能成为轮廓点的有效边界点.将所有的种子点收集起来,构成种子点集,以后的操作就只在种子点集上进行.我们之所以如此定义种子点,是为了保证曲线始终都紧贴着曲线的内部前进.

为了构造曲线的轮廓,先从队列中抽取一个种子点,然后以此种子点为起点,考虑它的 8 个邻居,从中挑选一个最可能的轮廓点作为曲线上的下一个点,依次循环,直至回到曲线的起点.下一个轮廓点的选择方法是算法的核心.

3.1 轮廓点的选择

在选择轮廓点时,8 个邻居的地位并不是平等的,根据曲线上前一个点以及当前点的位置,先将 8 个邻居进行排序,然后再依据次序进行选择.轮廓点的选择原则是:

(1) 方向优先性:让曲线沿着既有的方向继续前进;

(2) 紧贴内部点:曲线在前进过程中始终紧贴内部点.

本着原则(1),根据曲线前进过程中的 8 种不同情况,如果用 C 表示当前轮廓点,用 P 表示曲线上前一个点,则排序的结果如图 8 所示.

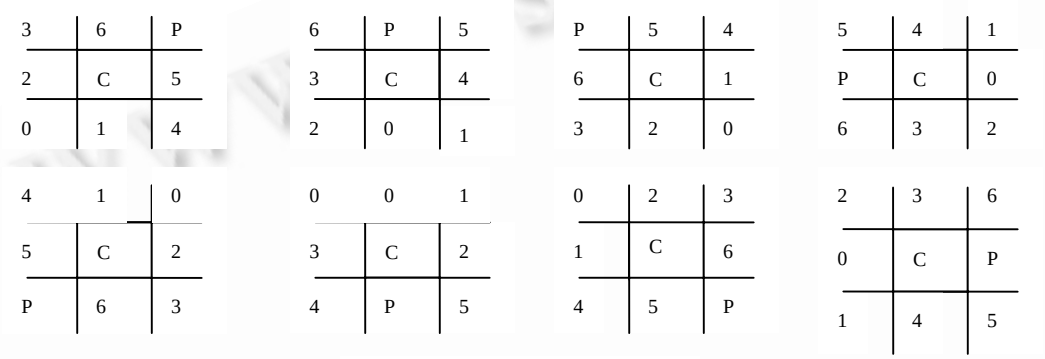

Fig.8 Eight kinds of sort results

图 8 8 种排序结果

经过排序之后,再以表 2 的方式定义各个点的左邻右舍.

Table 2 Adjacent codes define table

表 2 左邻右舍定义表

Current point code①	Left point code②	Right point code③
0	1	2
1	4	1
2	0	3
3	2	6
4	5	1
5	Undefined④	4
6	3	Undefined

①当前点的编号,②左邻的编号,③右舍的编号,④无定义.

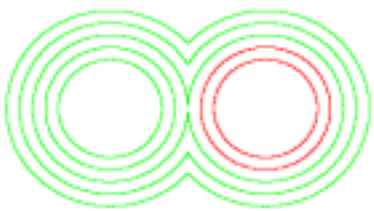

Fig.9 Evolution process of two circles

图 9 两个圆的演化过程

所谓一个点的左邻(右舍),是指如果下一个轮廓点选择该点,则根据曲线的前进方向,位于该点左侧(右侧)的点.为了避免曲线过早地回到起点,所以,如果一个点的左邻(右舍)的标号为 P,则标志为无定义.

需要注意的一点是曲线的起点,因为该点是曲线上的第 1 个点,所以不存在 P 点,此时可以任意选择一个邻点作为 P 点,然后按照如图 9 所示的方式进行编号,不同的是在它的左邻右舍定义表中,需要将“无定义”改为 P 点的编号.

在经过编码和定义左邻右舍之后,依据原则(2),根据各个邻居点的性质,依次判断是否属于下列 4 个等级之一,然后分别存储到 4 个队列中:

(1) 如果一个点的左邻是内部点,则将该点添加到等级 1 的队列中;

(2) 如果一个点的左邻是边界点,则将该点添加到等级 2 的队列中;

(3) 如果一个点的右舍是外部点,则将该点添加到等级 3 的队列中;

(4) 如果一个点的右舍是边界点,则将该点添加到等级 4 的队列中.

在经过挑选种子点、编码、分级等一系列操作之后,就可以选取曲线轮廓点了.选取轮廓的方法是先判断等级 1 的队列是否为空,如果存在等级为 1 的点,则选取该点为下一个轮廓点,否则依次检查等级为 2、3、4 的队列.一旦选定一个种子点为曲线轮廓上的点,就要把它从种子点集中删除.

在构造出曲线的轮廓之后,就可以利用传统的方法对窄带内的点重新进行初始化,并且可以灵活地控制曲线上各个点的运动速度,从而开始新一轮的演化过程.

4 实验结果与结论

利用文中介绍的距离函数平滑方法和轮廓跟踪算法,我们对不同类型的曲线的演化过程进行了模拟,下面是两个实验结果的说明.可以看出,尽管我们的曲线平滑方法相对于 Sethian 的要简单,但是结合后面的轮廓跟踪算法,仍然可以得到与 Sethian 所得到的相近的结果.

图 9 是两个圆(图中左右对称的最里边的两个圆)的演化过程.在初始阶段,两条曲线分别以匀速进行膨胀,但在演化到一定阶段之后,两条曲线便融合为一条曲线

图 10 是按照文中的方法,让一个行楷体的“马”字以速度 $F=-AVE(k)$进行演化的过程,$AVE(k)$表示平均曲率.

$$AVE(k)=\frac{\int k(s)\mathrm{d}s}{\int \mathrm{d}s},\quad s \text{ 是曲线的弧长.} \tag{9}$$

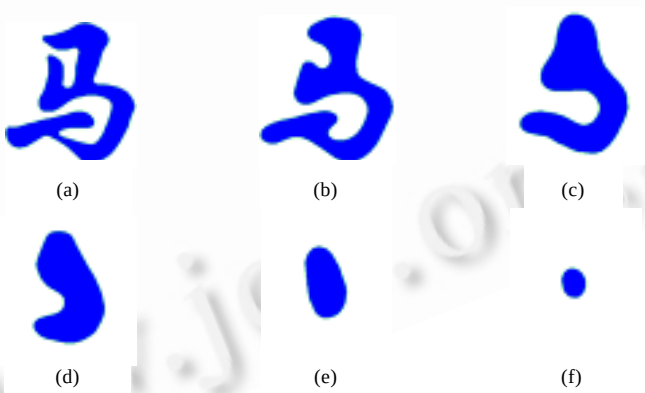

Fig.10 Evolution process of the Chinese word “horse”

图 10 “马”字的演化过程

Grayson 在文献[13]中证明,当平面上任何一条简单封闭曲线,以速度 $F=-k(k$ 是曲率或平均曲率)演化时,最终都将收缩为一个点.从图中可以清楚地看到,“马”字在慢慢地收缩,到图 10(f)时已经接近圆的形状,而我们知道,圆形以上述速度运动时,必将收缩为一个点.

本文从 Level Set 方法的一些具体应用领域出发,提出了一种平滑 Level Set 距离函数以及得到演化后曲线轮廓的方法.该方法对曲线演化的速度没有太多限制,适用于大多数 Level Set 曲线演化过程.经过实验证明,该方法简单、有效,结合 Sethian 的窄带方法,可以应用于图像处理、计算机视觉等领域.至于如何把这种平滑方法和轮廓跟踪方法扩展到三维曲面的演化,则是需要进一步研究的问题.

References:

[1] Osher, S., Sethian, J.A. Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulation. Journal of Computer Physics, 1988,79(1):12~49.

[2] Sethian, J.A. Numerical algorithms for propagating interfaces: Hamilton-Jacobi equations and conservation laws. Journal of Differential Geometry, 1990,31(1):131~161.

[3] Malladi, R., Sethian, J.A., Vemuri, B.C. Evolving fronts for topology-independent shape modeling and recover. In: Proceedings of the 3rd European Conference on Computer Vision. LNCS 800, Stockholm, 1994. 3~13.

[4] Malladi, R., Sethian, J.A. A unified approach to noise removal, image enhancement, and shape recovery. IEEE Transactions on Image Processing, 1996,5(11):1554~1568.

[5] Sethian, J.A, Strain, J.D. Crystal growth and dendritic solidification. Journal of Computer Physics, 1992,98(2):231~253.

[6] Sethian, J.A. Level Set Methods: Evolving Interfaces in Geometry Fluid Mechanics, Computer Vision, and Material Science. Cambridge: Cambridge University Press, 1996.

[7] Faugeras, O., Kerivan, R. Variational principles, surface evolution, PDE's, level set methods and the stereo problem. Technical Report, 3021, INRIA, 1996.

[8] Malladi, R., Sethian, J.A. Shape modeling with front propagation: a level set approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995,17(2):158~175.

[9] Adalsteinsson, D., Sethian, J.A. A fast level set method for propagating interfaces. Journal of Computer Physics, 1995,118(2): 269~277.

[10] Peng, Dan-ping, Merriman, Barry. A PDE-based fast local level set method. Journal of Computational Physics, 1999,155(2): 410~438.

[11] Jiang, Guang-shan, Peng, Dan-ping. Weighted ENO schemes for Hamilton Jacobi equations. Technical Report, 97-29, Department of Mathematics, University of California, 1997.

[12] Sethian, J.A. Level Set Methods and Fast Marching Methods. Cambridge: Cambridge University Press, 1996.

[13] Grayson, M. The heat equation shrinks embedded plane curves to round points. Journal of Differential Geometry, 1987,26(1):285.

Curves Evolving Based on Level Set Method*

YANG Meng, WANG Guo-ping, DONG Shi-hai

(*Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, The Chinese Academy of Sciences, Beijing* 100080, *China*);

(*Department of Computer Science and Technology, Beijing University, Beijing* 100871, *China*)

E-mail: yang@graphics.pku.edu.cn

http://graphics.pku.edu.cn

Abstract: Level Set method is a power tool for tracking the evolution of fronts propagating with curvature dependent speed. Since its introduction, Level Set method has been used in a wide collection of problems such as medical image processing, the simulation of natural phenomenon and computer vision. In practical applications, two of the most important algorithms are how to smooth the grid value and track the fronts contour after some time steps. In this paper, a simple method is introduced to smooth the value of Level Set function. It uses only intra-interpolation to clear out all of the single points and some of the ambiguous points. Use the contour line tracking method introduced in this paper, people can get the contour lines of all of the curves in a plane, even if there exists some redundant points. The experimental results show that this method is simple and useful, and can be used to wide fields concerning with fronts propagating.

Key words: Level Set method; curves evolving; medical image processing; contour tracking; image based modeling

* Received February 28, 2002; accepted June 18, 2002

Supported by the National Natural Science Foundation of China under Grant No.60173016; the National High-Tech Research and Development Plan of China under Grant No.2001AA115126; the Foundation of the Ministry of Education of China for the Backbone Teachers; the Natural Science Foundation of Beijing of China under Grant No.4012008