第 33 卷 第 2 期 计 算 机 工 程 2007 年 1 月
Vol.33 No.2 Computer Engineering January 2007
·多媒体技术及应用· 文章编号：1000—3428(2007)02—0186—03 文献标识码：A 中图分类号：TP39

基于时间延迟的矢量场可视化方法的应用研究

顾耀林，吉晓娟
(江南大学信息工程学院，无锡 214122)

摘 要：矢量场可视化是科学计算可视化中最具有挑战性的研究课题之一。该文提出了一种基于时间延迟逐步生成、绘制和显示流线的方法，从而把稳定的二维矢量场的拓扑结构、方向、速度等特征以动画的形式显示出来。实验证明此方法简单、直观、形象。
关键词：矢量场可视化；动画；流线；时间延迟

Study on Application for Visualization Based on Time Delay Vector Fields

GU Yaolin, JI Xiaojuan
(School of Information and Engineering, Southern Yangtze University, Wuxi 214122)

【Abstract】 Vector fields visualization is one of the most challenge researches of visualization in scientific computing. A new method about create, protract and display streamlines step by step based on time delay is presented. It can accurately depict topology, direction and velocity at any points of 2D steady vector fields as a form of animation. It can be proved that the method presented in this paper is simple, visual and imaginable.
【Key words】 Vector fields visualization; Animation; Streamline; Time delay

1 矢量场数据的处理流程及其相关的研究

矢量场可视化是科学计算可视化中最具挑战性的课题之一，它以直观的图形图像显示场的运动。透过抽像数据有效洞察其内涵本质和变化规律，广泛应用于计算流体力学、航空动力学、大气物理和气像分析等领域。

矢量场可视化至少要包括 3 个主要步骤[1]：矢量数据的预处理、矢量数据的映射以及矢量场的绘制和显示。其数据流模型如图 1 所示。传统的矢量场数据的映射方法有两种：一种是基于几何形状的，有箭头、流线、流面和流体；另一种是基于纹理生成的，有点噪声方法和线积分卷积法。点噪声方法[2]是由J.J.Van.Wijk于 1991 年提出的。线积分卷积法[3]是由B.Cabral, C.Leedom于 1993 年提出的。浙江大学的陈莉于 1996 年提出了可变形参数域卷积法[4]。W.Lefer提出了一种基于色表板动画技术[5]的矢量场数据映射技术。线积分卷积法还被应用于不稳定二维矢量场[6]和三维矢量场[7]的研究。但由于此方法只考虑了沿速度方向那条折线线段上的像素点，生成的图像高频噪声较大，矢量的大小很难通过其纹理反映出来。矢量场可视化的绘制与显示是利用计算机图形学的理论与算法将映射后的几何数据和属性数据以人们容易理解的图形或图像方式显示并绘制出来。

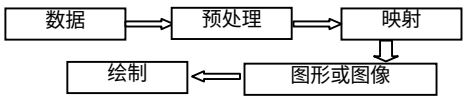


图 1 矢量场可视化的数据流模型

本文提出的方法是把矢量场数据映射到流线上，通过时间延迟以动画的形式把矢量场的特征显示出来。

2 基于时间延迟的可视化方法

2.1 种子点的生成

根据定义，一个矢量场的场线具有如下的性质：在任意一点，场线的方向与该点的矢量方向一致。速度矢量场的场线，称为流线。单个的流线只能表现流场的局部信息，多条流线才能反映整个流场的全局特征。这就需要合理布局流线。流线太少，可能使可视化图像缺失流场空间中某一部分信息，流线太多，又可能造成视觉的混淆。要想获得好的可视化效果，必须选择恰当的种子点，控制种子点之间的距离和流线之间的距离。本文的种子点选取算法能从当前流线中提取出尽可能多的种子点，算法如下：

```
选取矢量场的中心点为第一个种子点
计算初始流线并存入队列和链表中
//存入队列是为了选取种子点方便
//存入链表是为了数据映射方便
 记这条初始流线为当前流线
 Finished ：= False
  Repeat
    Repeat
      在距离当前流线距离为 d 处选取候选种子点
    //d 的大小根据实验的需要设定
    Until 候选种子点有效 or 没有可用的候选种子点
    If 一个有效的候选种子点被选取 Then
        计算新的流线并放入队列和链表中
    Else
        If 队列里没有可用的流线 Then
            Finished : = True
        Else
            记队列中的下一条流线为当前流线      EndIf
                        EndIf
Until Finished=True
```

作者简介：顾耀林(1948－)，男，教授，主研方向：计算机图形学与虚拟现实；吉晓娟，硕士
收稿日期：2006-02-13 **E-mail：**jixiaojuan1234@163.com

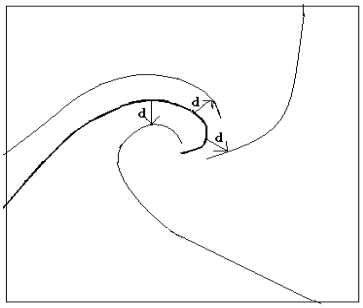


图 2 从与第 1 条流线(粗)距离为 d 处选择种子点生成流线(细)

2.2 流线的生成

一般将流线方程定义为如下形式：

$$\frac{d\vec{\gamma}(t)}{dt}=\vec{f}(\vec{\gamma}(t)) \tag{1}$$

即空间中流线上任一点的方向与该点的切线方向一致，其中 $\vec{\gamma}$ 代表空间点的位置，$\vec{f}(\vec{\gamma})$ 为流线上任一点的切线方向，$\vec{\gamma}$ 为 t 的函数，t可以是时间、弧长等参量。

本文流线方程的建立则严格按照流线的定义给出，即

$$\Delta\vec{l}\times\vec{v}=0 \tag{2}$$

其中：$\Delta\vec{l}$ 为矢量线上的矢量元，$\vec{v}$ 为速度。

$$\Delta\vec{l}=\Delta X\vec{i}+\Delta Y\vec{j} \tag{3}$$

$$\vec{v}=u\vec{i}+v\vec{j} \tag{4}$$

且 u,v 为 X,Y 的函数，按矢量叉乘的物理含义可知，式(2)的物理含义为经过流线上的任一点的切线方向与该点的速度方向相平行。将式(3)、式(4)带入式(2)进一步推导可得出

$$\frac{\Delta X}{u}=\frac{\Delta Y}{v}=k \tag{5}$$

则式(5)为生成二维流线的原始方程。流线方程中参数 k 的确定，由式(5)得

$$(\Delta X)^2+(\Delta Y)^2=k^2\left(u^2+v^2\right)$$

在 u,v 不同时为 0 的情况下：

$$k=\pm\frac{\sqrt{(\Delta X)^2+(\Delta Y)^2}}{\sqrt{u^2+v^2}} \tag{6}$$

将 k 取为正值，则式(5)变为如下形式：

$$\begin{cases}\dfrac{\Delta X}{\sqrt{(\Delta X)^2+(\Delta Y)^2}}=\dfrac{u}{\sqrt{u^2+v^2}}\\[2ex]\dfrac{\Delta Y}{\sqrt{(\Delta X)^2+(\Delta Y)^2}}=\dfrac{v}{\sqrt{u^2+v^2}}\end{cases} \tag{7}$$

令 $(\Delta X)^2+(\Delta Y)^2=(\Delta R)^2$，则 ΔR 为流线上相邻两点间距离，即为求取流线的步长。因为 $u^2+v^2=\left|\overline{V}\right|^2$，则式(7)转化为

$$\begin{cases}\dfrac{\Delta X}{\Delta R}=\dfrac{u}{\left|\overline{V}\right|}\\[2ex]\dfrac{\Delta Y}{\Delta R}=\dfrac{v}{\left|\overline{V}\right|}\end{cases} \tag{8}$$

式(8)表示的物理含义是 x,y 两个方向的方向余弦，对与式(8)的两个初始条件为

$$\begin{cases}X(\Delta R_i)=X_0\\Y(\Delta R_i)=Y_0\end{cases}$$

即 (X_0,Y_0) 为流线初始点坐标，即每条流线种子点的坐标。上面的流线方程可表示成如下形式：

$$\begin{cases}\dfrac{\Delta X}{\Delta R}=\dfrac{u}{\left|\overline{V}\right|}=f(\Delta R,X,Y)\\[2ex]\dfrac{\Delta Y}{\Delta R}=\dfrac{v}{\left|\overline{V}\right|}=f(\Delta R,X,Y)\\X(\Delta R_i)=X_0\\Y(\Delta R_i)=Y\end{cases} \tag{9}$$

已知 (X,Y) 的初值 (X_0,Y_0)，即初始点坐标，也就是种子点的坐标，为求出流线上下一点坐标位置，用 4 阶龙格一库塔方法。对于 k 取为负值用类似的方法讨论。设起始点与下一点的距离为 h，即积分步长为 h，根据 4 阶龙格一库塔方法有：

$$\begin{cases}K_1=f(\Delta R_i,X_i,Y_i)\\K_2=f\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_1,Y_i+\dfrac{1}{2}hL_1\right)\\K_3=f\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_2,Y_i+\dfrac{1}{2}hL_2\right)\\K_4=f\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_3,Y_i+\dfrac{1}{2}hL_3\right)\\X_{i+1}=X_i+(h/6)\times(K_1+2K_2+2K_3+K_4)\end{cases} \tag{10}$$

$$\begin{cases}L_1=g(\Delta R_i,X_i,Y_i)\\L_2=g\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_1,Y_i+\dfrac{1}{2}hL_1\right)\\L_3=g\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_2,Y_i+\dfrac{1}{2}hL_2\right)\\L_4=g\left(\Delta R_i+\dfrac{h}{2},X_i+\dfrac{1}{2}hK_3,Y_i+\dfrac{1}{2}hL_3\right)\\Y_{i+1}=Y_i+(h/6)\times(L_1+2L_2+2L_3+L_4)\end{cases} \tag{11}$$

这样通过求解式(10)，式(11)即可求出流线上下一点的坐标 (X_{i+1},Y_{i+1})。

对于生成的流线从种子点开始把该流线上计算出来的点称为样本点依次存储到链表里，一条流线一个链表。另设一个头结点链表，存放每个链表的头结点。在生成流线的过程中要设一个计数器记录流线的样本点个数，当样本点个数小于 3 时，删除该流线的链表，因为至少要 3 个点才能确定一条流线的走向。通常有 3 种原因使流线的生成过程中断，如遇到关键点[9]、遇到边界点、遇到处理过的点。链表结点的数据结构定义如下：

```
struct node{
    float x;//该结点的 x 坐标
    float y;//该结点的 y 坐标
    float u;//该结点 i 方向的矢量
    float v;//该结点 j 方向的矢量
    float velocity;//该结点的速度
    float time;//绘制该结点需要延迟的时间
      float …//该结点所要显示的向量场的一些变量
        …
}
```

当整个矢量场所有流线计算完毕存入链表即完成了数据映射这一步骤，接着就涉及到流线的绘制和显示。

2.3 流线的绘制和显示

对于上面数据结构中的变量 time 用下面公式计算：

$$time=1/velocity\times k \tag{12}$$

式(12)中 k 为参量，可以根据计算机 CPU 的处理速度设定，

当 CPU 速度快时可以把 k 设定得比较大，当 CPU 速度慢时可以把 k 设定得比较小。Velocity 为结点的速度。使用 OPENGL 语言中的 GL_LINES 语句连接相邻的两个点，程序如下：

```
Do{
  for (t=0; t<time; t++)  ;
//用空循环表示时间延迟
  glBegin(GL_LINES);
  glVertex2f(x, y);
  glVertex2f(x1, y1);
//连接相邻的两点(x,y)和（x1,y1）
  glEnd();
  x1=x;
  y1=y;
  p=p->next;
  x1=p->x;
  y1=p->y;
  time=(int)p->time;
  } while (p->next !=NULL);
```

通过上面的程序就可以把矢量场中一条流线画出来，流线生成的方向就显示了矢量的方向，划线速度的快慢就显示了矢量的大小。这样就以动画的形式把一条流线上各点矢量的大小及方向显示出来。

矢量场有许多条流线，而计算机在某一时刻只能处理其中一条流线上的一个点。在实验中通过头结点链表取出每条流线待绘制的结点的延迟时间 time 放入一个排序对列中进行排序，当 time 最小的那个点的时间延迟结束时就绘制该点，并从该点所在的流线链表中取出下一个点放入排序队列中进行新一轮排序。只要链表中某个结点的延迟时间一到就处理该结点，这样就可以同时绘制和显示多条流线。

在矢量场中可以把温度、压力量化后，通过流线粗细和不同的颜色来显示这些标量数据所表示的特征。

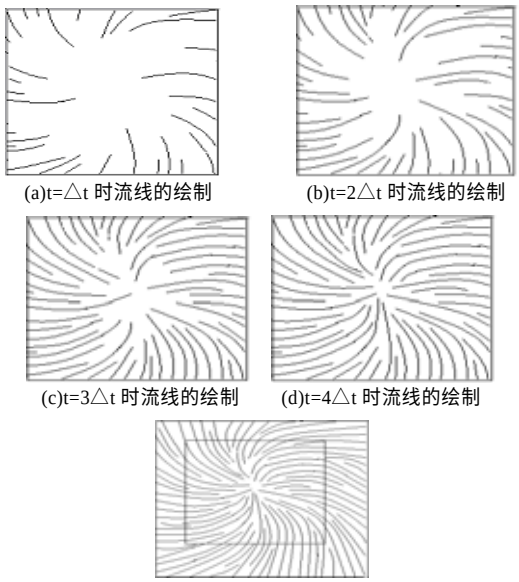
2.4 算法的分析与对比

参考文献[8]中提出的种子点生成算法如下：先计算关键点的位置并判断类型，接着根据关键点计算 Voronio 分区，然后套用模板分布种子点生成流线，最后在空白处随机地补充一些种子点生成流线。与本文的算法相比文献[8]中的算法复杂，考虑的问题更多，并可能在可视化图像的空白处丢失一部分种子点，从而缺失流场空间中某一部分的信息。本文的种子点生成算法可以提取尽可能多的种子点，这样就不会遗漏流场中的关键点和特征数据，并通过控制种子点之间的距离和流线之间的距离使流线均匀分布。

3 实验结果

为了验证算法的可行性，在实验中用 C 语言完成矢量场数据的映射，用 OPENGL 语言完成矢量场数据的绘制与显示。实验所需的数据源是由法国 Pau 大学的 Wilfrid Lefer 教授提供的.VEC 格式的数据源。在配置为 Intel® Celeron® CPU 2.10 GHz, Intel® 82845G/GL/GE/PE/GV Graphics Controller，256MB 内存的工作站上运行该程序。所得的实验结果如图 3 所示。图 3(a)~图 3(d)是每隔时间△t 从整个动画中截取的 4 幅图像，每幅图像也只截取了整幅图像中的一部分，这部分也体现了矢量场拓扑结构关键点中的重要一类吸引点。

由于生成的流线有长有短，每个样本点的速度有快有慢，因此绘制和显示每条流线所需的时间不同，但必须等矢量场中所有的流线全部绘制完后，整幅图像才消隐[10]，然后从头开始重新绘制和显示矢量场的流线。这样就以循环动画的形式把矢量场的流线显示出来。



(a)t=△t 时流线的绘制　(b)t=2△t 时流线的绘制

(c)t=3△t 时流线的绘制　(d)t=4△t 时流线的绘制

(e)矢量场绘制完毕的整幅图像

图 3 矢量场流线绘制和显示的整个动画过程

4 结束语

本文提出的基于时间延迟的矢量场可视化方法以动画的形式显示了矢量场的流线及流线绘制的快慢。只是显示每个样本点速度的相对速度快慢并不是绝对的速度。以流线的粗细和颜色表示向量场数据的内在关系也只是一种相对的比较并不是绝对的。在动画的循环过程中由于图像的消隐会出动画的跳跃现象，需要进一步改进。本方法比较简单、形象、直观，现在只把它应用于稳定的二维矢量场，同样可以对三维矢量场进行类似的研究，这也是我们下一步要做的工作！

参考文献

1 石教英, 蔡文立. 科学计算可视化算法与系统[M]. 北京: 科学出版社, 1996.

2 Wijk J J V. Spot Noise: Texture Synthesis for Data Visualization[J]. Computer Graphics, 1991, 25(4): 309-318.

3 Cabral B, Leedom C. Imaging Vector Fields Using Line Integral Convolution[C]. Proceedings of SIGGRAPH'93, 1993: 263-270.

4 陈　莉. 三维矢量场可视化的基础算法研究[D]. 杭州: 浙江大学, 1996.

5 Lefer W, Jobard B, Leduc C. High-quality Animation of 2D Steady Vector Fields[J]. Visualization and Computer Graphics, 2004, 10(1): 2-14.

6 Liu Z, Moorhead II R J. AUFLIC: An Accelerated Algorithm for Unsteady Flow Line Integral Convolution[C]. Proc. of IEEE TCVG/ EuroGraphics, 2002.

7 Liu Z, Moorhead II R J. Visualizing Time-varying Three-dimensional Flow Fields Using Accelerated UFLIC[C]. Proc. of the 11th Int'l Symp. on Flow Visualization, 2004: 43-52.

8 Verma V, Kao D, Pang A. A Flow-guided Streamline Seeding Strategy[C]. Proceedings of the Conference on Visualization, 2000: 163-170.

9 Helman J L, Hesselink L. Visualizing Vector Field Topology in Fluid Flows[J]. IEEE Computer Graphics and Applications, 1991, 11(3): 36-46.

10 Rogers D F. 计算机图形学的算法基础[M]. 北京: 机械工业出版社, 2002.