Advances in Applied Mathematics 应用数学进展, 2018, 7(1), 118-127
Published Online January 2018 in Hans. http://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2018.71015

A Centroidal Voronoi Tessellation Based Visualization Algorithm for Time-Dependent Flow Field

Tiancheng Gao, Liyong Zhu

School of Mathematics and Systems Science, Beihang University, Beijing
Email: gtczz@sina.com

Received: Dec. 22nd, 2017; accepted: Jan. 18th, 2018; published: Jan. 26th, 2018

Abstract

In this work, we present a visualization algorithm based on Centroidal Voronoi tessellation (CVT) for time-dependent flow field datum. The proposed method is based on the CVT-based visualization algorithm for steady flow field datum. The proposed method has good computation cost of obtaining the CVT by inheriting the time layer generator. And by introducing short streamline arrows to represent flow field datum, this method can avoid the reciprocal chiasma of straight arrow as well as can capture the features of flow field datum. Numerical examples demonstrate effectiveness and robustness of the centroidal Voronoi tessellation based visualization algorithm for typical flow field datum with eddies.

Keywords

Time-Dependent Flow Field Datum, Visualization, Centroidal Voronoi Tessellation, Short Streamline Arrows, Inheriting Generator

一种基于CVT的动态流场可视化方法

高天成，朱立永

北京航空航天大学，数学与系统科学学院，北京
Email: gtczz@sina.com

收稿日期：2017年12月22日；录用日期：2018年1月18日；发布日期：2018年1月26日

摘　要

本文给出了一种基于CVT (Centroidal Voronoi Tessellation)的动态流场可视化方法。这种方法基于稳态

文章引用: 高天成, 朱立永. 一种基于 CVT 的动态流场可视化方法[J]. 应用数学进展, 2018, 7(1): 118-127.
DOI: 10.12677/aam.2018.71015

流场的可视化方法，借助于不同时间层生成元的继承性，缩短了可视化的计算时间；同时，通过引入短流线箭头，不但避免了直线箭头表示向量时相互交叉的问题，而且又更清楚地刻画了流场的特征。数值算例表明了所给出的基于CVT动态流场可视化方法的有效性。

关键词

动态流场，可视化，质心Voronoi剖分，短流线箭头，生成元的继承性

1. 引言

科学计算可视化，是指将科学计算和实验获得的数据，转化为人们可以直接识别的图像或图形的行为。通过数据可视化，人们可以直观地看出数据的特点、理解数据背后的本质，为后续的精确分析和处理提供方向上的指导。通常的可视化过程可分为以下四个步骤：第一，数据过滤：主要包括预处理原始数据、转换数据形式、滤掉噪声、抽取特征数据等；第二，映射：将过滤后的数据映射为几何元素，常见的几何元素有点、线、面、图元、三维体图元等；第三，绘制：绘制几何元素，得到结果图像；第四，反馈：显示图像并分析得到的可视化结果。这四步中尤以数据过滤最为关键，它不仅决定了过滤后的数据是否能抓住问题的特征，而且过滤后数据量的大小直接影响着后面几步的处理时间。寻找最能代表数据特征的数据是数据过滤的关键。

在文献[1]中，Du 和 Wang 提出了一种基于 CVT (Centroidal Voronoi Tessellation)的稳态向量场可视化方法。这种方法通过引入空间和向量场的距离函数来度量空间分布向量场的相似性。基于这样的距离，向量场可以被自然地分类并找出其中典型代表。这种方法既有简单几何上的直观，又建立在 CVT 的最优性质基础之上。这种方法描述简单，容易理解和实施。文献[2]基于计算 CVT 的多水平算法，发展了一种快速的 CVT 可视化方法，这种方法要优于经典的 Lloyd 方法，可以做到对不同尺寸问题的一致收敛。在文献[3]中，Liu 和 Lu 等人提出了流线 CVT 的概念，并给出了一种数值计算流线 CVT 的方法，这种方法可以给出流线的最优位置来代表流场的几何特征。

流场可视化是科学计算可视化的一个分支方向[4]，同时也是流体力学的重要部分。流场可视化推动了计算流力学的快速发展。流场可视化用箭头、流线和粒子跟踪技术研究二维流场。在本文中，我们将文献[1]中发展的基于 CVT 的可视化方法推广到动态流场的情形。借助于相邻时间层生成元的继承性，缩短了可视化的计算时间。同时，通过引入短流线箭头，不但避免了直线箭头表示向量时相互交叉的问题，而且又更清楚地刻画了流场的特征。数值算例表明了所给出的基于 CVT 的动态流场可视化方法的有效性。

本文的结构安排如下，在第二节，我们将介绍文献[1]中发展的基于 CVT 的稳态流场数据可视化方法；在第三节，基于生成元的继承性和引入短流线的表示方法，我们给出了针对动态流场数据的基于 CVT 的可视化方法；在第四节，我们给出了几个动态流场的可视化结果；最后，我们给出了简单的总结。

2. Delaunay 三角化和高维嵌入

这里首先给出文献[1]中发展的基于 CVT 稳态流场的数据可视化方法，这是后面发展针对动态流场数据可视化方法的基础。

2.1. CVT 的定义及其计算方法

我们从 CVT 的定义[5]开始：

定义 1：对一个点集 $\Omega \subseteq \mathbf{R}^n$，记 $d(x_1, x_2) \in \mathbf{R}$ 为 x_1, x_2 两点间的距离。一个划分可以由一组子集 $\{V_i\}_{i=1}^k$ 表示。这个划分称为 CVT，如果满足：

1) 对所有的子集 V_i，$V_i \cap V_j = \varnothing \Leftrightarrow i \neq j$，$\bigcup_{i=1}^{k} \overline{V_i} = \overline{\Omega}$ 其中 $\overline{V_i}$，$\overline{\Omega}$ 是 V_i，Ω 的闭包。

2) 每个子集的质心，记为 $\{v_i\}_{i=1}^k$，恰好是整个 Voronoi 划分的生成元，即为：

$$V_i = \left\{ x \in \Omega \mid d(v_i, x) < d(v_j, x), j \neq i \right\}.$$

文献[5]中还以定理的形式提出了一种等价的基于能量的 CVT 的定义。定义能量积分为：

$$E(x) = \int_{V_i} \rho(s) d^2(x, s) \mathrm{d}s.$$

若 $\{v_i\}_{i=1}^k$ 是对应于 $\{V_i\}_{i=1}^k$ 的 CVT 的质心即生成元 v_i，同时也是最小化能量积分的点。其中 $\rho(s)$ 为密度函数。

如果生成元给定，就可以通过定义 1 中的等式生成 Voronoi 划分。同时，通过最小化能量积分函数，可以直接得到这个划分的全部质心。尽管如此，直接生成 CVT 依然很困难，因为质心和生成元通常不重合。在文献[5]中给出了如下的 CVT 生成算法：

算法 1: 生成 CVT 的 Lloyd 算法

输入:

一个点集 Ω;

一个正整数 k 用来将 Ω 分成 k 个区域;

一个阈值 ϵ;

输出:

k 个点作为 CVT 的生成元;

在 Ω 中选取任意 k 个不重合的点, 记为 $\{v_i\}_{i=1}^k$.
while *true* **do**
  对每一个 $0 < i \leq k$, 生成它的区域 $\{v_i\}_{i=1}^k$. 记为 $\{V_i\}_{i=1}^k$.
  对每一个 $0 < i \leq k$, 通过最小化能量函数计算区域质心 $\{V_i\}_{i=1}^k$. 记质心点为 $\{\overline{v}_i\}_{i=1}^k$.
  计算变动距离：$\delta = \sum_{i=1}^k d(v_i, \overline{v}_i)$.
  if $\delta \leq \epsilon$ **then**
    输出 k 个点 $\{\overline{v}_i\}_{i=1}^k$ 并终止循环;
  else
    替换 $v_i = \overline{v}_i, i = 1, \ldots, k$ 并继续循环;
  end
end

该算法的优点是简单，有效。其在一维空间中对 CVT 的收敛性由 Du Q.，Emelianenko M.，Ju L.等在文献[6]中给出，对高维空间有一些弱收敛性[7] [8]。但是，通常 Lloyd 算法收敛很慢，在一维中证明了其为线性收敛。因此，为了加快 Lloyd 算法的计算，可以使用过松弛移动[9]，拟牛顿方法[10]等方法来提升计算效率。

2.2. 基于 CVT 的稳态流场可视化方法

对一个简单点集，由算法\ref{al1}可以生成其对应的 CVT。在文献[5]中提出基于 CVT 的可视化应该分成两个步骤：第一步由原始数据计算出对应的 CVT，第二步对生成可 CVT 通过合适的可视化策略绘制成普通人可以直接理解的图片。但是对像流场一样相对复杂一点的集合，如果希望质心点能够表现更多的信息，就应当将这些信息加入计算之中。一个直接的想法就是修改能量函数。

对一个点集 $\Omega \subset \mathbf{R}^n, \forall x \in \Omega$，加上其上的向量函数 $y = F(x) \in \mathbf{R}^m$ 成为一个流场。为了最小化其上的能量函数，可以使用稳态流场 CVT 生成算法。

算法 2: 稳态流场 CVT 生成算法

输入:

一个流场中的点的集合 Ω 和其上的向量函数 $F(x)$；

一个正整数 k 用来将 Ω 分成 k 个区域;

一个阈值 ϵ;

输出:

k 个点作为 CVT 的生成元;

每个生成元上一个向量;

选取 Ω 中 k 个不重合的点, 记为 $\{x_i\}_{i=1}^k$

while *true* **do**

  由 k 个点 $\{x_i\}_{i=1}^k$ 生成 VT：

  $V_i = \{x \in \Omega | d(x, x_i) < d(x, x_j), j \neq i\}$;

  计算每个区域的质心：$\overline{x}_i = \frac{\int_{V_i} |F(x)|^2 x dx}{\int_{V_i} |F(x)|^2 dx}$ $\overline{y}_i = \frac{\int_{V_i} |F(x)| F(x) dx}{|\int_{V_i} |F(x)| F(x) dx|}$

  $i = 1, \ldots, k$;

  计算位移量：$\delta = \sum_{i=1}^k d(x_i, \overline{x}_i)$;

  if $\delta < \epsilon$ **then**

    输出 $\{\overline{x}_i, \overline{y}_i\}_{i=1}^k$ 并终止循环;

  else

    替换 $\{x_i, y_i\}\{\overline{x}_i, \overline{y}_i\}_{i=1}^k$ 并继续循环;

  end

end

在该算法中假定密度是常数，即整个场是均匀的。这等价于 $\rho(x) \equiv 1$。对于场密度不为常数的情况，可以简单地将算法中的质心计算式改为：

$$\overline{x}_i = \frac{\int_{V_i} \rho(x) |F(x)|^2 x \mathrm{d}x}{\int_{V_i} \rho(x) |F(x)|^2 dx}, \quad \overline{y}_i = \frac{\int_{V_i} \rho(x) |F(x)| F(x) \mathrm{d}x}{\left| \int_{V_i} \rho(x) |F(x)| F(x) \mathrm{d}x \right|}$$

其中 $i = 1, \cdots, k$。

这种算法是基于 Lloyd 的方法，具有方法简单，收敛稳定的优势。

之所以在可视化之前先进行 CVT 的处理而不直接对所有点进行描绘，其原因是希望通过 CVT 提取流场的特征，去除不必要的冗余，避免关注点被掩盖。相比于其它的特征提取方法，CVT 是通过极小化能量函数获得的，因而可以将能量函数相对应的距离函数导出的特征表现的最突出。

在图 1 中我们对一个稳定的绕柱流场进行了可视化。图 1(左)中，将所有的数据点进行了绘制，而在

图 1(右)中仅绘制了 CVT 的质心点。通过图 1(右)可以很明显的看出圆柱右侧的两个涡这一明显的特征，而在图 1(左)中这一特征被过多的数据点掩盖了。这即是基于 CVT 的可视化的典型优势：消除冗余，保留特征。

通过 Lloyd 算法，可以得到一个误差足够小的 CVT。至此，得到了一组质心点和其上的单位向量。如果只关心整个流场的流向，那么这就已经足够了，但如果同时对流速也有兴趣，那么就需要对这个单位向量的模长进行修正。因为质心代表了整个区域的信息，所以在修正模长时一般选择一种有代表性的长度，典型的有区域最大模长、最小模长，或者简单平均模长。为了与质心公式的加权方法匹配，文献[1]中给出了加权平均模长的例子：

$$L_i = \frac{1}{|V_i|}\int_{V_i} |F(x)| \mathrm{d}x,$$

或者

$$L_i = \frac{1}{V_i}\left(\int_{V_i} |F(x)|^2 \mathrm{d}x\right)^{1/2},$$

对于非均匀场：

$$L_i = \frac{\left(\int_{V_i} \rho(x)|F(x)|^2 \mathrm{d}x\right)^{1/2}}{\int_{V_i} \rho(x)\mathrm{d}x}.$$

这样，通过将之前的单位向量 $\overline{y}_i$ 修正为 $\overline{y}_i * L_i$。每个 $\overline{y}_i * L_i$ 的长度表征了整个区域内的向量的平均大小，即可以更直观地表现出不同区域向量大小的区别。区域内向量的平均模长越大，则代表向量的长度越大，反之平均模长越小，代表向量则越短。

3. 基于 CVT 的动态流场数据可视化

有时需要可视化一段时间的流场，从而观察其变化。但对于多个时间步，获得 CVT 的计算时间将会是个很大的问题。为此，基于两个连续时间步的连续性，提出一种起始点 $\{v_i\}_{i=1}^k$ 的选择方法来加速对动态流场数据 CVT 的计算。

考虑流场 F 发生变化时，能量函数 $E(x)$ 可以写成 $E(x,F)$ 即为流场和一组点的连续泛函。限制流场

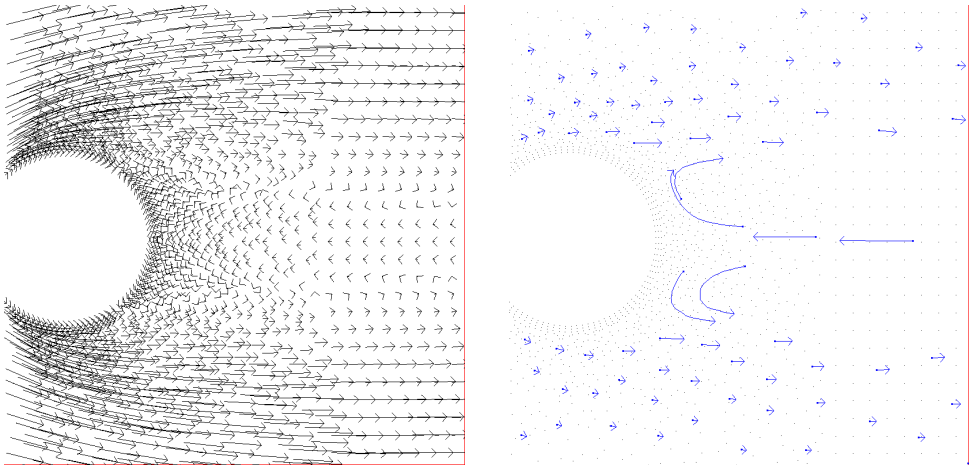

Figure 1. Flow around a pile
图 1. 圆柱绕流

的区域不动，且由流场函数本身的连续性，可以得出流场函数空间为$C(\Omega)$的连续函数空间，故为内积空间，于是可以定义流场函数空间中的距离。记一个连续变化的流场的场函数为$F(t)$，t为时间参数。由流场变化性质可得$F(t)$关于t连续。至此，对于两个连续时间步，即Δt不大，则F变化不大，对于最小化能量泛函的x来说亦不会有太大的移动。具体对应算法中质心的计算公式是区域内所有点关于向量模和密度函数的加权平均，换言之，只要向量模，和密度函数变化不大，那么这个加权平均值也不会有太大的改变。

基于这一想法，提出了针对动态流场，快速生成一系列 CVT 的方法，从而进一步实现对动态流流场的数据可视化。

3.1. 动态流场 CVT 生成算法

该算法在简单计算一系列稳态流场的基础上，改进了每个时间步初始 CVT 生成元的选取。通过这个改进便可以将整体计算时间减少一半以上。表 1 和图 2 中展示了对同一动态流场相同终止条件下两种选点方法的时间对比。

算法 3: 动态流场 CVT 生成算法

输入:

一个动态流场中每个时间步的点的集合 Ω_i 和其上的向量函数 $F_i(x)$ $i=1,2,...,n$;

一个正整数 k 用来将 Ω 分成 k 个区域;

一个阈值 ϵ;

输出:

每个时间步 k 个点作为 CVT 的生成元;

每个生成元上一个向量;

选取 Ω_1 中 k 个不重合的点, 记为 $\{x_i^1\}_{i=1}^k$

for $a=1; a\le n; a++$ **do**

  while *true* **do**

    由 k 个点 $\{x_i^a\}_{i=1}^k$ 生成 VT:

    $V_i^a=\{x\in\Omega_a | d(x,x_i^a)<d(x,x_j^a), j\neq i\}$;

    计算每个区域的质心: $\overline{x}_i^a=\frac{\int_{V_i^a}|F_a(x)|^2 x dx}{\int_{V_i^a}|F_a(x)|^2 dx}$

    $\overline{y}_i^a=\frac{\int_{V_i^a}|F_a(x)|F_a(x)dx}{|\int_{V_i^a}|F_a(x)|F_a(x)dx|}$ $i=1,\ldots,k$;

    计算位移量: $\delta=\sum_{i=1}^k d(x_i^a,\overline{x}_i^a)$;

    if $\delta<\epsilon$ **then**

      输出 $\{\overline{x}_i^a,\overline{y}_i^a\}_{i=1}^k$ 并跳出 While 循环;

    else

      替换 $\{x_i^a,y_i^a\}$ 为 $\{\overline{x}_i^a,\overline{y}_i^a\}_{i=1}^k$ 并继续循环;

    end

  end

  if $a\neq n$ **then**

    取 Ω_{a+1} 中 k 个点 $\{x_i^{a+1}\}_{i=1}^k$ 与 $\{x_i^a\}_{i=1}^k$ 重合;

  end

end

Table 1. Comparison on time cost
表 1. 时间花费对比

时间步	0	5	10	15	20	25	30	35	40	45	50	55
非随机(s)	17	61	90	126	163	209	266	325	372	422	454	460
随机(s)	17	111	197	294	387	479	572	692	808	920	1024	1162

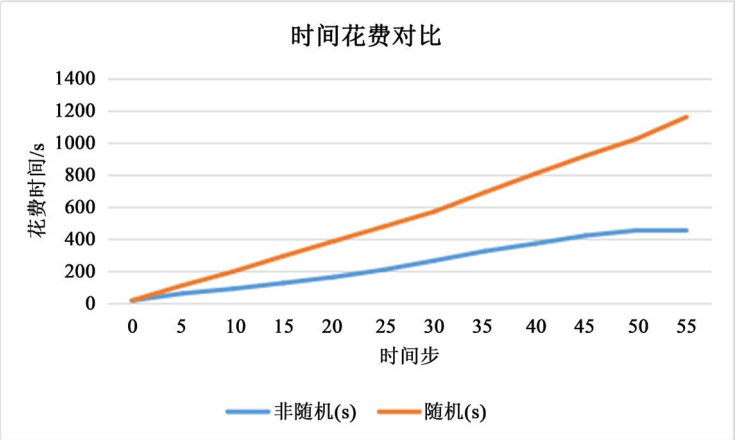

Figure 2. Comparison on time cost with different generators choosen strategy
图 2. 不同生成元选择时间花费对比

同时，由于对相邻时间步的 CVT 生成元进行了继承，在后一时间步中每一对 CVT 生成元的相对移动可以很大程度上反映出这一时间步中流场的变化，即特征位置的移动。对于动态流场，通过继承上一个时间步的生成元，可以在高效的同时更好的表现流场的特征变化。因为 CVT 的性质，生成元“倾向”于聚集在对能量函数贡献较大的点附近。因此，生成元的移动能在一定程度上反映能量函数极值点的移动。但是对于每个时间步随机选取初始生成元的情况，两个时间步之间的生成元没有关联性，最终很难将两组生成元进行对应，因而割裂的整个动态流场的连续性关系，使得连续时间步时生成元跳跃极大，十分不利于观察。

3.2. 可视化策略——短流线

对于动态流场的可视化，依然可以对每一个时间步沿用稳态流场的可视化策略：对向量长度进行恢复，添加颜色表示额外信息。但在实际问题中，注意到在选点较多时，经常会出现向量的表示箭头的交

算法 4: 短流线生成算法

输入:

初始点 z_0 和向量函数 $F(z)$;

迭代次数 k;

输出:

一条流线;

for $i = 0; i < k; i++$ **do**

  计算 $w_i = F(z_i)$ 作为这个循环的向量;

  将 z_i 移动到 $z_{i+1} = z_i + \frac{w_i}{k*|w_i|}$;

  在 z_i 和 z_{i+1} 之间绘制一条直线;

end

叉，导致可视化效果变差。对此，基于流场流线不相交的特性，提出向量的局部短流线箭头表示，从而避免直接交叉，影响可视化效果。

对于流场函数没有定义在连续空间的情况，可以通过插值得到每一个位置的向量函数近似值，从而通过上述算法得到短流线。与此同时，短流线在整个区域的流向上有更好的表现，因为其沿着流线移动，在流线变化剧烈的区域，典型的如涡流，能够更加清楚地表现出这些特征。

4. 应用实例

本节主要展示一些具体的基于 CVT 的流场数据可视化实例。其中的流场均为非稳态可压缩流场，由非稳态可压缩 Navier-Stokes 方程控制[11]：

$$\frac{\partial U(x,t)}{\partial t}+\frac{\partial F_k\left(U(x,t)\right)}{\partial x_k}=\frac{\partial G_k\left(U(x,t),\Delta U(x,t)\right)}{\partial x_k},$$

其中 U 代表守恒量， F 代表对流通量， G 代表粘性通量。

4.1. 翼型周围流场可视化

在图 3 中展示了一个无攻角匀速运动的二维翼形周边的流场。可以看出随着时间演化，机翼上侧速度逐渐增大，而下侧则逐渐减小。这与文献[12]中的结果一致，表现出了翼型周围流场的变化情况。

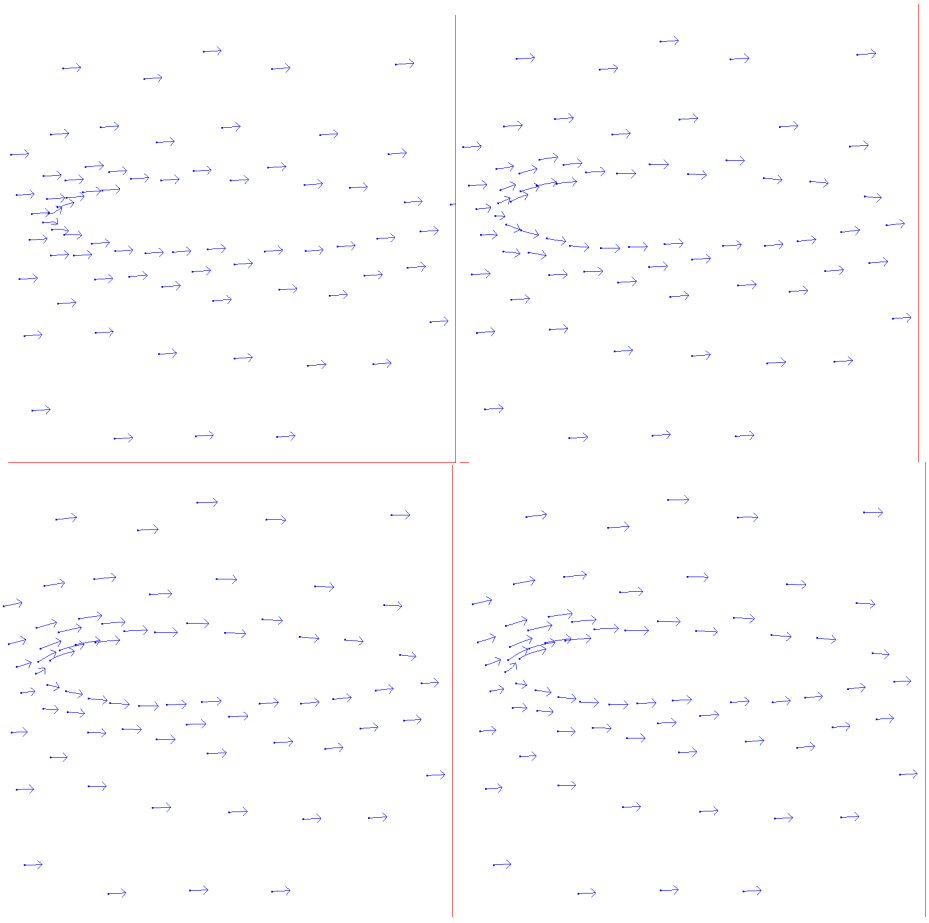

Figure 3. Variation of the flow field around a wing
图 3. 翼型周围的流速变化

需要注意的是，不考虑具体问题地使用相同的距离函数和相同的可视化策略是十分危险的。很显然，应该针对不同的特征选择合适的距离函数和可视化策略。在下一节中，图 4 左侧三幅图与右侧三幅图展示了对同一个流场，不同距离函数和可视化策略所产生的巨大差别。

4.2. 绕柱流场可视化

图 4 描绘了一个绕柱流场的演化过程。流体在绕过刚体柱后先是平流，之后逐渐演变最终形成两个涡，使用了 $E(x)=\int_{V_i}\frac{d^2(x,s)}{F(s)}\mathrm{d}s$ 作为能量函数。使用这样的能量函数，CVT 生成元自动地聚集在了速度较低的位置。左侧三张图中不对向量大小进行修正，直接绘制短流线，其结果是两个涡并不明显。而在右侧三张图中，使用上文算法中提到的能量函数，同时取向量模长的倒数，并以此为基础步长绘制短流线。可以看出，此时两个涡是十分明显的。

5. 总结

本文通过应用短流线和继承生成元的方法发展了一种针对动态流场的基于 CVT 的可视化方法。短流线可以表达更多流场中的细节信息，继承生成元在缩短计算时间的同时，也能将流场的变化刻画的更加明显。通过这两种方法，我们可以将整个流场的特征表现的更加清楚。

相比于其他的可视化方法，基于 CVT 的可视化在全局可视化，算法可靠性，计算效率，适应性上有一定的优势。许多的可视化方法倾向于通过选择一些特定的原始数据点来避免冗余信息，CVT 方法通过生成新的代表点，即质心，来表征区域信息。这使得它可以不受原始数据点的束缚，来生成更加有代表性的质心，从而更好的可视化整个流场。

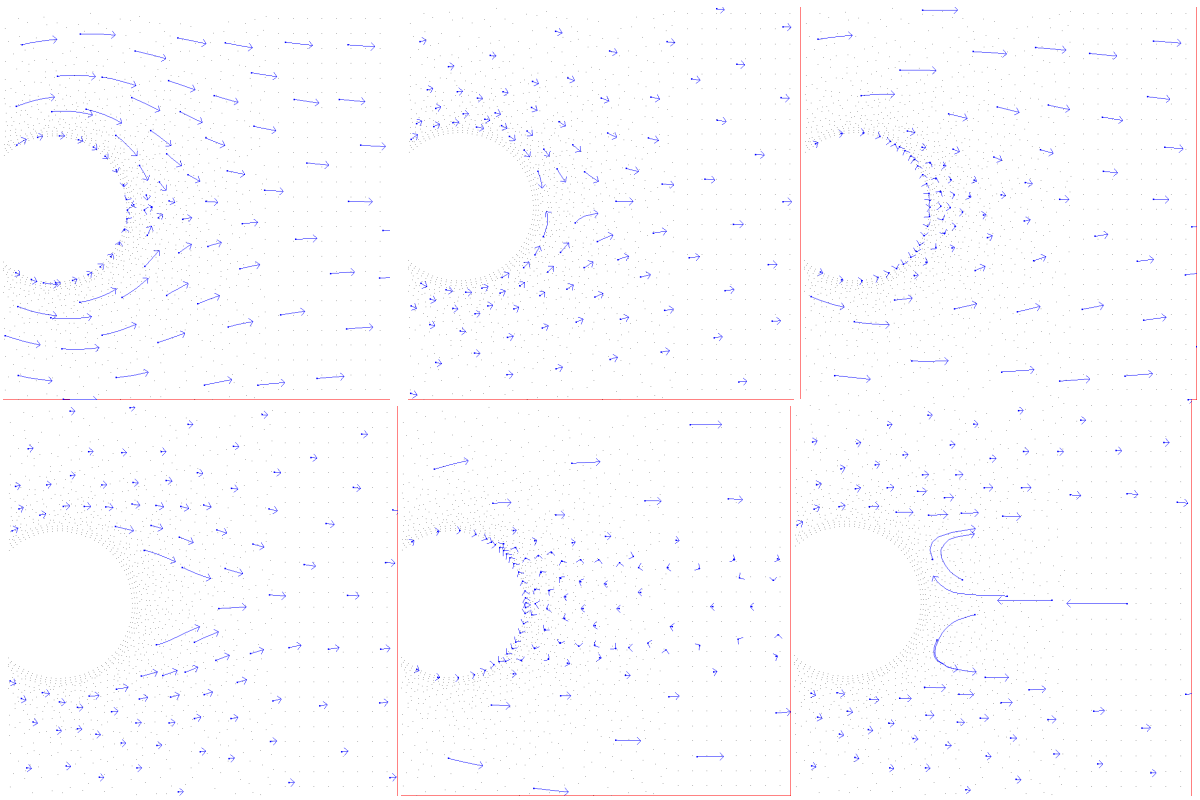

Figure 4. Flow around a pile
图 4. 绕柱流

尽管如此，还有一些问题依然存在。基于 Lloyd 的 CVT 生成算法计算量大，收敛缓慢，需要大量的时间迭代出最终结果，稳定快速的 CVT 生成算法仍然需要更多更深入的研究。同时，对不同的情况如何选择可视化策略，目前只是基于经验地尝试不同的策略然后选择较好的。对此，如何针对性的选择合适的可视化策略，将是一个值得深入研究的问题。

基金项目

国家民用客机基础研究项目(项目号：MJ-F-2012-04)资助项目。

参考文献 (References)

[1] Du, Q. and Wang, X. (2004) Centroidal Voronoi Tessellation Based Algorithms for Vector Fields Visualization and Segmentation. *Proceedings of the* 2004 *Conference on Visualization*, Washington DC, 10-15 October 2004, 43-50.

[2] Emelianenko, M. (2010) Fast Multilevel CVT-Based Adaptive Data Visualization Algorithm. *Numerical Mathematics*: *A Journal of Chinese Universities*, **3**, 195-211. https://doi.org/10.4208/nmtma.2010.32s.5

[3] Liu, W., Lu, L., Levy, B., *et al.* (2013) Centroidal Voronoi Tessellation of Streamlines for Flow Visualization. International Symposium on Voronoi Diagrams in Science and Engineering. *Proceedings of the* 2013 10*th International Symposium on Voronoi Diagrams in Science and Engineering*, Washington DC, 8-10 July 2013, 75-81. https://doi.org/10.1109/ISVD.2013.8

[4] 刘晓波，华祖林，何国建. 计算流体力学的科学计算可视化研究进展[J]. 水动力学研究与进展，2004(01): 120-125.

[5] Du, Q., Faber, V. and Gunzburger, M. (1999) Centroidal Voronoi Tessellations: Applications and Algorithms. *Siam Review*, **41**, 637-676. https://doi.org/10.1137/S0036144599352836

[6] Du, Q., Emelianenko, M. and Ju, L. (2006) Convergence of the Lloyd Algorithm for Computing Centroidal Voronoi Tessellations. *Siam Journal on Numerical Analysis*, **44**, 102-119. https://doi.org/10.1137/040617364

[7] Sabin, M.J. and Gray, R.M. (1986) Global Convergence and Empirical Consistency of the Generalized Lloyd Algorithm. *IEEE Transactions on Information Theory*, **32**, 148-155. https://doi.org/10.1109/TIT.1986.1057168

[8] Emelianenko, M., Ju, L. and Rand, A. (2009) Nondegeneracy and Weak Global Convergence of the Lloyd Algorithm in Rd. *SIAM Journal on Numerical Analysis*, **46**, 1423-1441. https://doi.org/10.1137/070691334

[9] Xiao, X. (2010) Over-Relaxation Lloyd Method For Computing Centroidal Voronoi Tessellations. Master's Thesis, University of South Carolina, Columbia.

[10] Hateley, J.C., Wei, H. and Chen, L. (2014) Fast Methods for Computing Centroidal Voronoi Tessellations. *Journal of Scientific Computing*, **63**, 185-212. https://doi.org/10.1007/s10915-014-9894-1

[11] Cheng, J., Yang, X., Liu, T., *et al.* (2015) A Direct Discontinuous Galerkin Method for the Compressible Navier-Stokes Equations on Arbitrary Grids. *Journal of Computational Physics*, **327**, 484-502. https://doi.org/10.1016/j.jcp.2016.09.049

[12] 叶建，林国华，邹正平，等. 低雷诺数下二维翼型绕流的流场特性分析[J]. 航空动力学报, 2003, 18(01): 38-45.