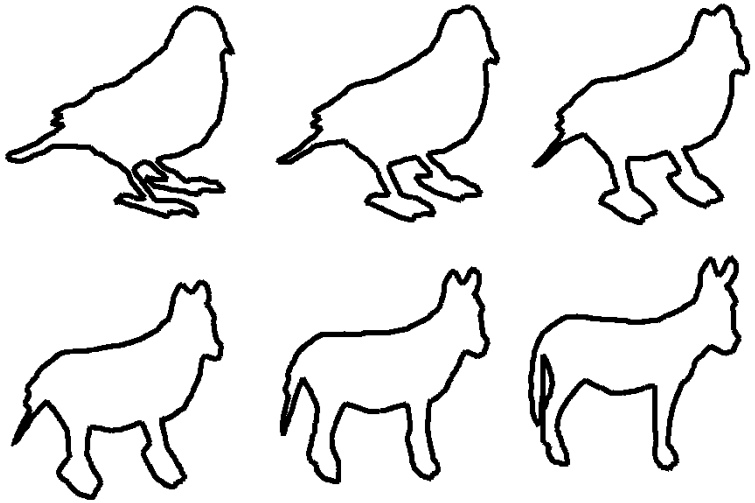
二维多边形形状渐变
2D Shape Blending



金小刚　Email: jin@cad.zju.edu.cn

2D Shape Blending演示

问题的引出

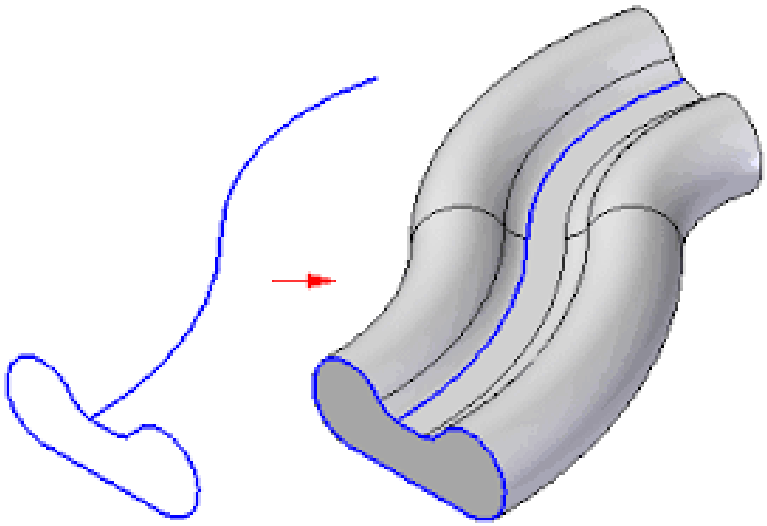
- 在二维角色动画(Character Animation)中，经常会碰到这样的问题：给定一个初始和最终的形状(shape)，我们称它们为关键帧形状，求从初始形状光滑过渡到最终形状的中间形状。

- 这个问题称为二维形状的自然渐变（Shape Blending或Shape Morphing）。

- 该问题实际上分为两个子问题
  - 顶点的对应关系问题;
  - 顶点的插值问题。

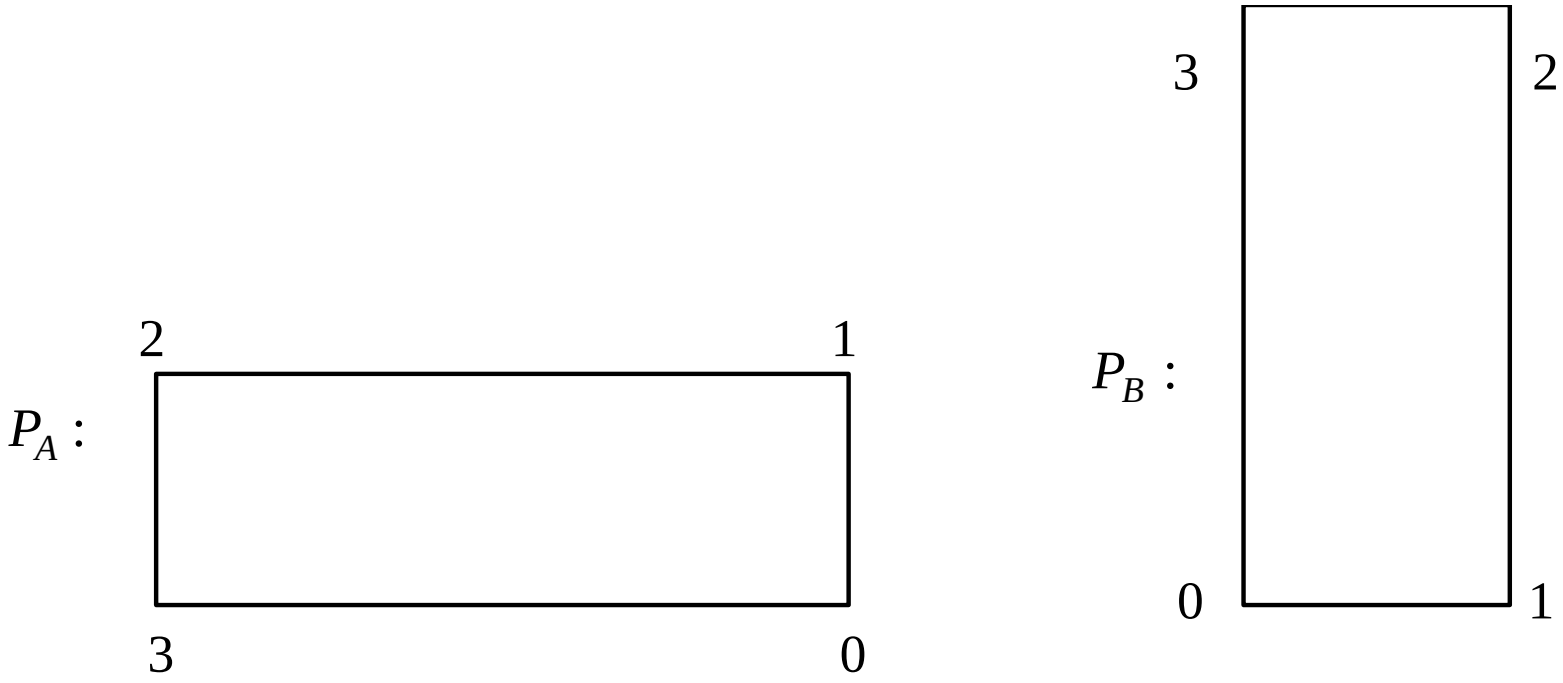
问题的引出

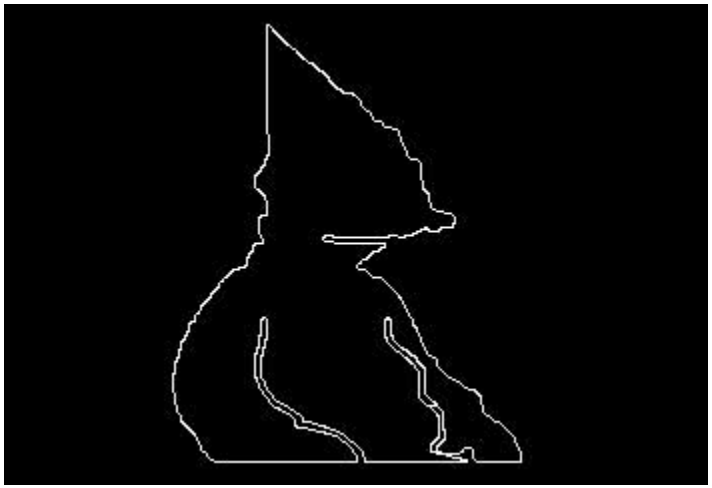
- 二维形状的自然渐变不仅在计算机动画，而且在模式识别、曲面重建和三维造型中也具有重要意义。



Sweeping曲面

Skin曲面


P_A :
2
1
3
0
P_B :
3
2
0
1

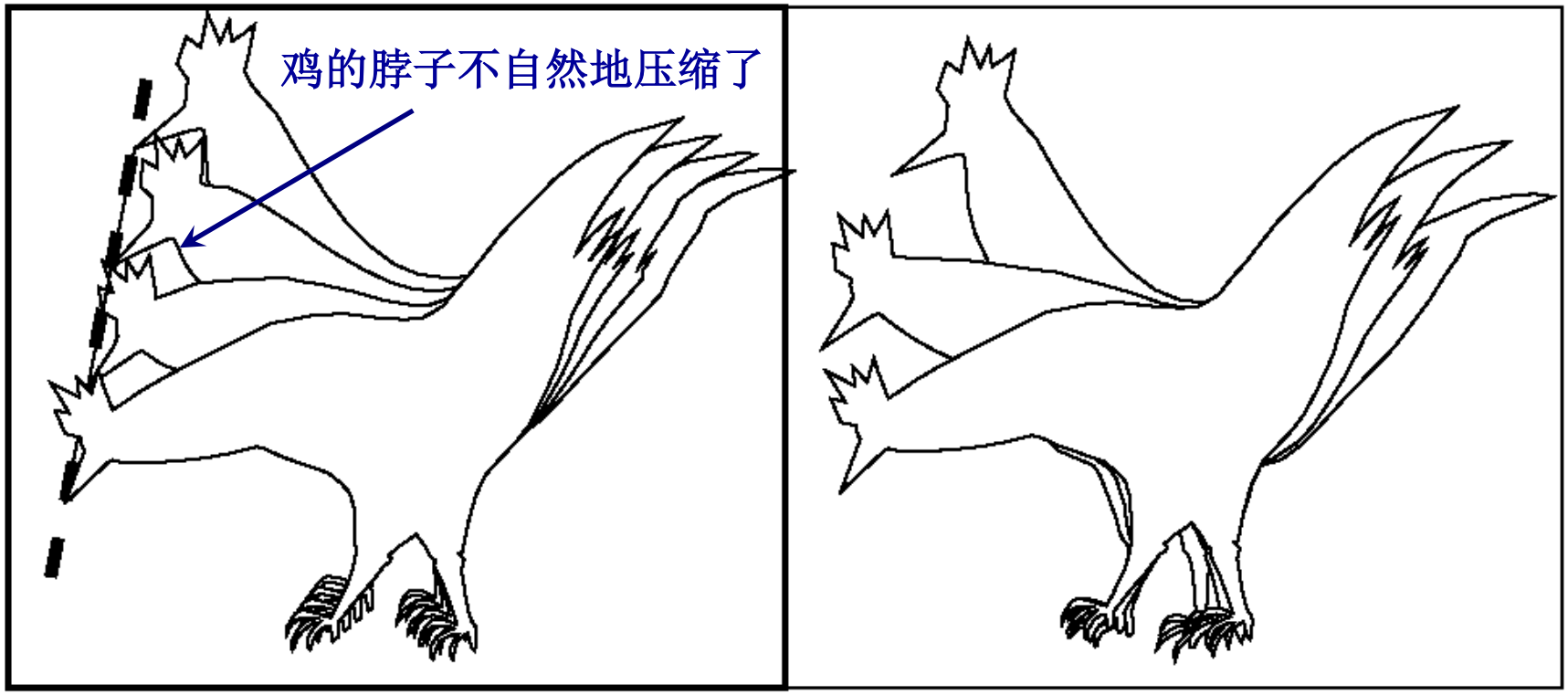


线性插值法

- 假设两个关键帧多边形的顶点为$\mathbf{P}_{A_i}$和$\mathbf{P}_{B_i}$ (i=0, 1, 2, 3, …, n)，顶点的数目都为n个，则我们要解决的是：
  - 顶点的对应关系问题，即多边形$\mathbf{P}_A$中的一个顶点与$\mathbf{P}_B$中的哪个顶点对应?
  - 顶点的路径问题，即$\mathbf{P}_A$以何种方式运动到$\mathbf{P}_B$？
- 顶点路径问题一个简单的方法为采用线性插值：

$$P_i = (1-t)P_{A_i} + tP_{B_i}, \qquad (i = 0,1,2,\cdots,n-1)$$

- 但线性插值会带来收缩(shrinkage)和纽结现象(Kink)，这在刚体旋转时表现得尤为明显。



(a) 鸡的脖子压缩了　　**(b)** 更自然的结果

鸡的形状渐变。

基于内在形状插值的多边形渐变方法

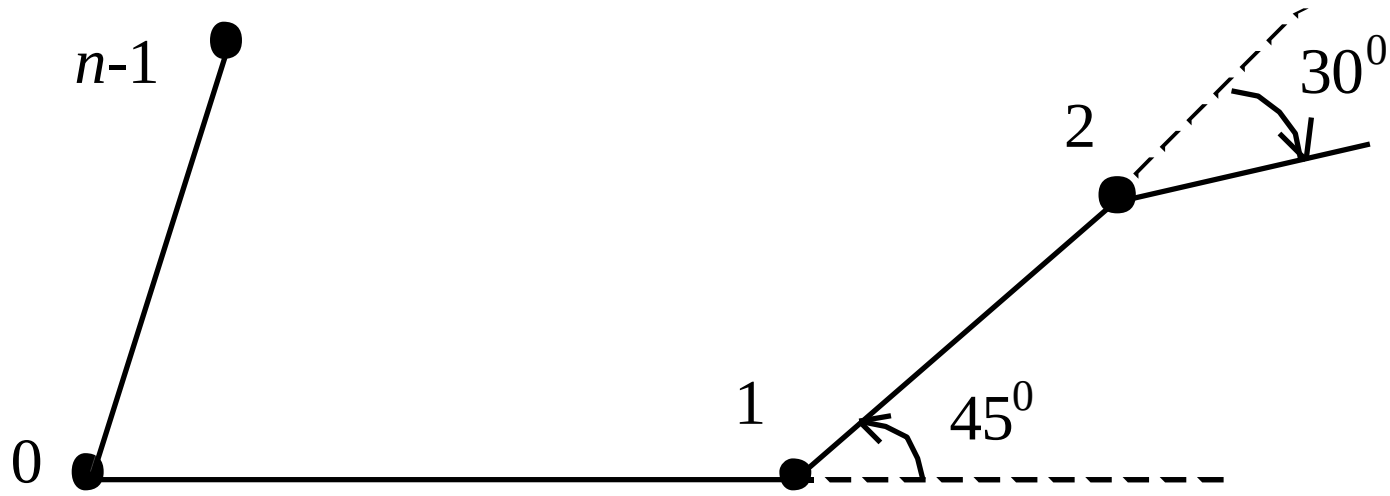
- 参考：**Sederberg T W, Gao P S, Wang G J, Mu H. 2D shape blending: an intrinsic solution to the vertex path problem. Computer Graphics, 1993, 27(3):15~18**

- **Intrinsic Shape Interpolation的数学原理：乌龟几何+Lagrange乘数法优化**

乌龟几何

- 在笛卡尔坐标系中，多边形是通过顶点的坐标显式给出的。
- 但多边形也可以通过乌龟几何来定义，即通过顶点处的边长和有向角来定义。例如，以某一点为起点，向东往前走10米，往左拐45^0，继续往前走6米，往右拐30^0，向前走5米，…，最后得到一多边形。
- 因而一个自然的想法是，能否对关键帧多边形的边长和顶点角进行插值来产生比线性插值更好的效果？
- 回答是肯定的，因为这种插值具有更好的几何意义。

乌龟几何



用乌龟几何定义多边形

多边形的内在定义

- 设$m=n-1$，逆时针方向的角为正的，P_0为形状的平移锚点(anchor point)，我们的目标是计算中间多边形的顶点P_i ($i=1, 2, \ldots, m$), $0 \le t \le 1$。
- 通过计算多边形P_A和P_B的边长和有向顶点角，我们得到多边形的内在定义$\{\alpha_0, L_0, (\theta_i, L_i)_{i=1}^{m}\}$ ：

$$\alpha_{A_0} = \theta(x, P_{A_0}, P_{A_1})$$

$$\alpha_{B_0} = \theta(x, P_{B_0}, P_{B_1})$$

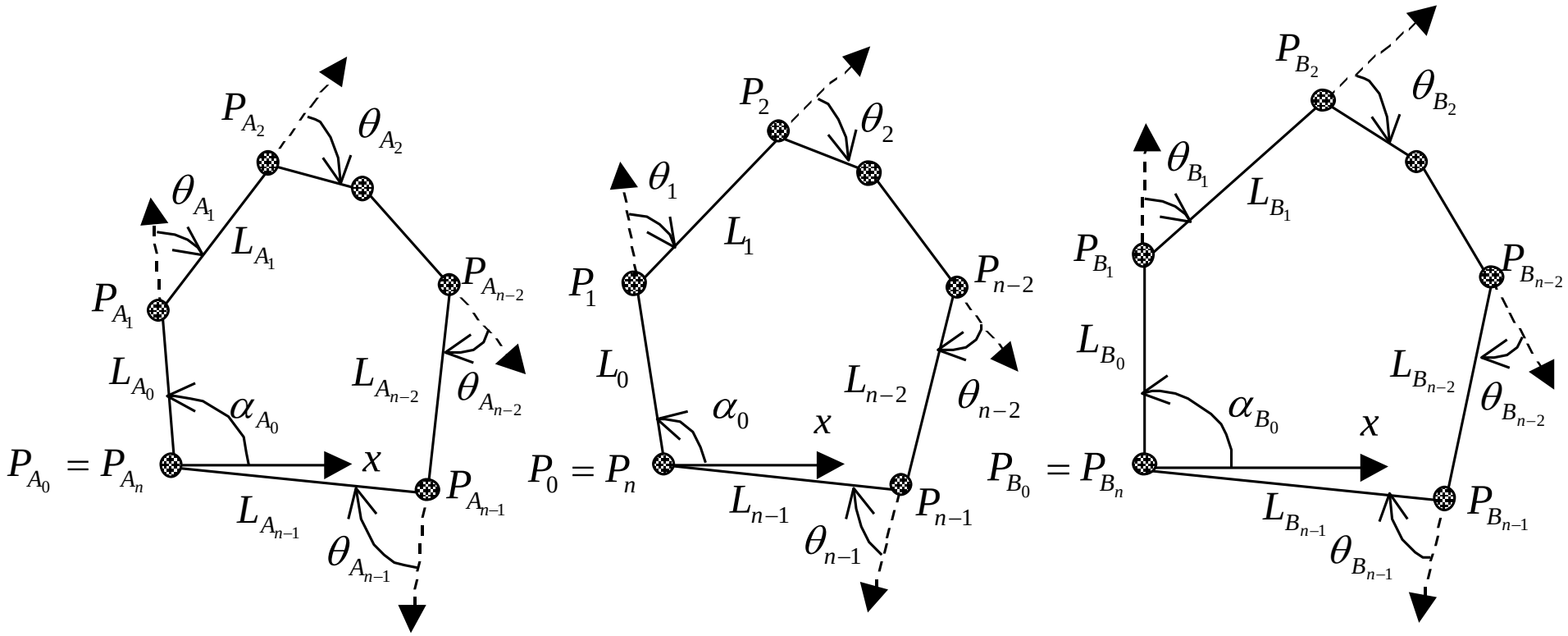
$$\theta_{A_i} = \theta(P_{A_{i-1}}, P_{A_i}, P_{A_{i+1}})$$

$$\theta_{B_i} = \theta(P_{B_{i-1}}, P_{B_i}, P_{B_{i+1}})$$

$$L_{A_i} = |P_{A_{i+1}} - P_{A_i}|$$

$$L_{B_i} = |P_{B_{i+1}} - P_{B_i}|$$

其中：$\theta(P_{A_{i-1}}, P_{A_i}, P_{A_{i+1}})$表示边$P_{A_{i-1}}P_{A_i}$和边$P_{A_i}P_{A_{i+1}}$之间的有向夹角


P_{A_2}
θ_{A_2}
θ_{A_1}
L_{A_1}
P_{A_1}
$P_{A_{n-2}}$
L_{A_0}
$L_{A_{n-2}}$
$\theta_{A_{n-2}}$
α_{A_0}
$P_{A_0} = P_{A_n}$
x
$P_{A_{n-1}}$
$L_{A_{n-1}}$
$\theta_{A_{n-1}}$
P_2
θ_2
θ_1
L_1
P_1
P_{n-2}
L_0
L_{n-2}
θ_{n-2}
α_0
x
$P_0 = P_n$
L_{n-1}
P_{n-1}
θ_{n-1}
P_{B_2}
θ_{B_2}
θ_{B_1}
L_{B_1}
P_{B_1}
$P_{B_{n-2}}$
L_{B_0}
$L_{B_{n-2}}$
$\theta_{B_{n-2}}$
α_{B_0}
x
$P_{B_0} = P_{B_n}$
$L_{B_{n-1}}$
$P_{B_{n-1}}$
$\theta_{B_{n-1}}$

多边形的内在定义

中间多边形生成

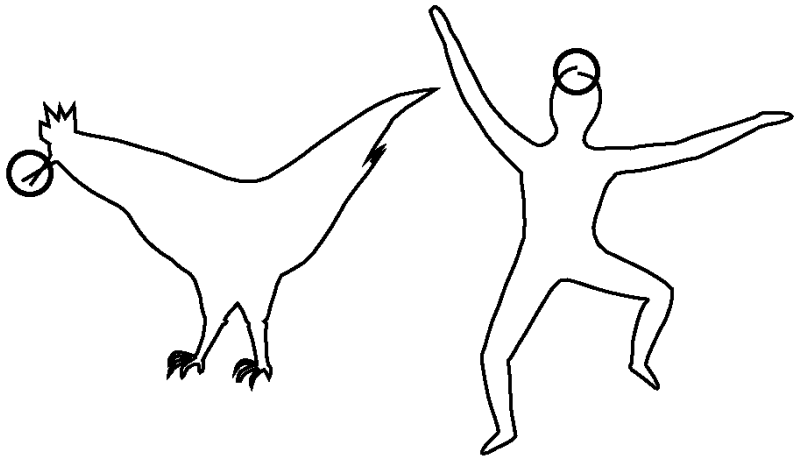
- 那么中间多边形可由插值相应的边长和顶点角得到：

$$\alpha_0 = (1-t)\alpha_{A_0} + t\alpha_{B_0}$$

$$\theta_i = (1-t)\theta_{A_i} + t\theta_{B_i} \qquad (i = 1,2,\cdots,m)$$

$$L_i = (1-t)L_{A_i} + tL_{B_i} \qquad (i = 1,2,\cdots,m)$$

- 遗憾的是，这样得到的多边形通常是不封闭的。但实验发现，得到多边形的起始点和终止点非常接近。



不封闭的多边形

中间多边形生成

- 一个解决方法为保持插值的顶点角不变，适当调整插值得到的边长**(Edge Tweaking)**：

$$L_i = (1-t)L_{A_i} + tL_{B_i} + S_i,\ \ (i = 0,1,2,\cdots,m)$$

- 如果关键多边形的某一条对应的边具有相同的长度L，那么我们可认为中间多边形的边长也为L，因此我们可认为 $|S_i| \propto |L_{A_i} - L_{B_i}|$ 。为了防止除以零，我们定义：

$$L_{AB_i} = \max\left\{\left|L_{A_i} - L_{B_i}\right|, L_{tol}\right\}, \qquad (i = 0,1,2,\cdots,m)$$

其中 $L_{tol} = 0.0001 \times (\max_{i\in[0,m]}\left|L_{A_i} - L_{B_i}\right|)$

中间多边形生成

- 我们的目标是为了求得 $S_0, S_1, \cdots, S_m$，使目标函数：

$$f(S_0, S_1, \cdots, S_m) = \sum_{i=0}^{m} \frac{S_i^2}{L_{AB_i}^2}$$

最小，并且 $S_0, S_1, \cdots, S_m$ 应满足约束条件(强迫多边形封闭)：

$$\varphi_1(S_0, S_1, \cdots, S_m) = \sum_{i=0}^{m} \left[(1-t)L_{A_i} + tL_{B_i} + S_i\right]\cos\alpha_i = 0$$

$$\varphi_2(S_0, S_1, \cdots, S_m) = \sum_{i=0}^{m} \left[(1-t)L_{A_i} + tL_{B_i} + S_i\right]\sin\alpha_i = 0$$

其中α_i是由矢量P_iP_{i+1}和x轴构成的有向角，

$$\alpha_i = \alpha_{i-1} + \theta_i, (i = 1, 2, \cdots, m)$$

中间多边形生成

- 这是个具有约束条件的极值问题，可用**Lagrange**乘数法来求解。引进**Lagrange**函数:

$$\Phi(\lambda_1,\lambda_2,S_1,S_2,\cdots,S_m)=f+\lambda_1\varphi_1+\lambda_2\varphi_2$$

其中λ_1，λ_2为**Lagrange**乘数。对Φ求偏导得:

$$\frac{\partial\Phi}{\partial S_i}=\frac{2S_i}{L^2_{AB_i}}+\lambda_1\cos\alpha_i+\lambda_2\sin\alpha_i=0 \qquad (i=0,1,\cdots,m)$$

$$\sum_{i=0}^{m}[(1-t)L_{A_i}+tL_{B_i}+S_i]\cos\alpha_i=0$$

$$\sum_{i=0}^{m}[(1-t)L_{A_i}+tL_{B_i}+S_i]\sin\alpha_i=0$$

中间多边形生成

- 由第一式求得S_i代入后二式得：

$$\begin{cases} E\lambda_1 + F\lambda_2 = U \\ F\lambda_1 + G\lambda_2 = V \end{cases}$$

其中

$$E = \sum_{i=0}^{m} L_{AB_i}^2 \cos^2 \alpha_i, F = \sum_{i=0}^{m} L_{AB_i}^2 \sin \alpha_i \cos \alpha_i, G = \sum_{i=0}^{m} L_{AB_i}^2 \sin^2 \alpha_i$$

$$U = 2\left\{\sum_{i=0}^{m}\left[(1-t)L_{A_i} + tL_{B_i}\right]\cos \alpha_i\right\}$$

$$V = 2\left\{\sum_{i=0}^{m}\left[(1-t)L_{A_i} + tL_{B_i}\right]\sin \alpha_i\right\}$$

中间多边形生成

- 如果 $EG - F^2 \neq 0$ ，则

$$\lambda_1 = \begin{vmatrix} U & F \\ V & G \end{vmatrix} \Big/ \begin{vmatrix} E & F \\ F & G \end{vmatrix}$$

$$\lambda_2 = \begin{vmatrix} E & U \\ F & V \end{vmatrix} \Big/ \begin{vmatrix} E & F \\ F & G \end{vmatrix}$$

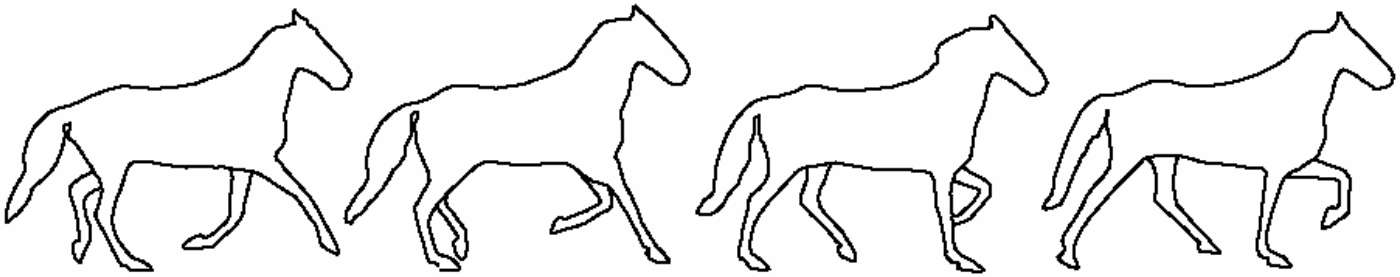
$$S_i = -\frac{1}{2} L_{AB_i}^2 (\lambda_1 \cos\alpha_i + \lambda_2 \sin\alpha_i),\ (i = 0,1,2,\cdots,m)$$

- 因而可得到中间多边形顶点的坐标：

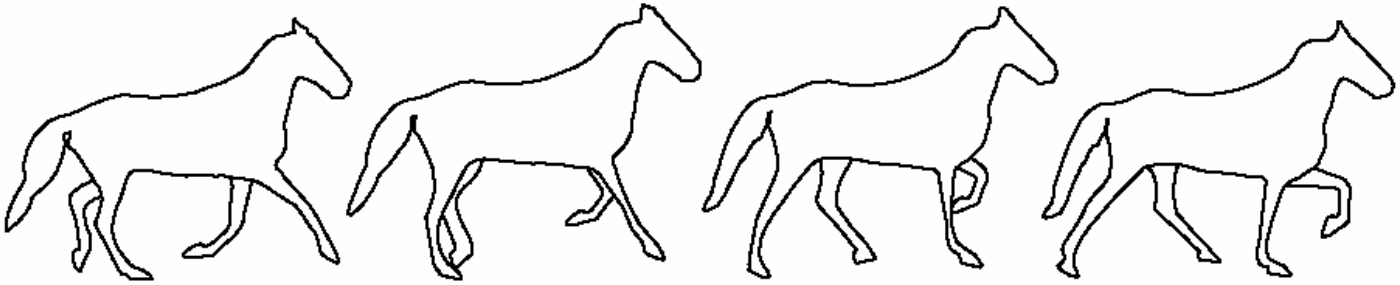
$$\begin{cases} x_i = x_{i-1} + L_{i-1} \cos\alpha_{i-1} \\ y_i = y_{i-1} + L_{i-1} \sin\alpha_{i-1} \end{cases}$$

结果

- 实验表明，对于角色动画中的许多应用，内在插值的形状渐变效果良好。



数字化得到的形状(形状取自经典摄影图书《Animals in Motion》)



给定的　　内在插值得到的　　内在插值得到的　　给定的

算法实现

输入：两个关键帧多边形的顶点

- 计算两个关键帧多边形的内在表示；
- 解线性方程组，计算λ_1, λ_2；
- 计算S_i；
- 中间多边形顶点的坐标；

输出：中间帧多边形的顶点

DEMO

内在形状插值法的优缺点

- 优点：
  - 简单；
  - 能在一定程度上避免形状插值中出现的收缩和纽结现象，这在二维角色动画中尤其有用。
- 缺点：
  - 不能处理曲线形状；
  - 当源和目标多边形中包含短边时，由于计算短边的方向不稳定，中间帧多边形有可能产生较严重的畸变。

基于模糊数学的二维多边形形状渐变

- **1996年，Zhang提出了一种基于模糊数学的二维多边形形状渐变方法。**
- 它采用基于模糊数学的方法来建立源多边形和目标多边形顶点的对应关系以及顶点之间的插值。
- 优点：
  - 能处理具有不同位置、方向、大小或形状的多边形。该方法不仅稳定，而且易于推广到曲线形状。

- **参考：Zhang Yuefeng. A fuzzy approach to digital image warping. IEEE Computer Graphics and Applications, 1996, 16(6):34~41**

基于模糊数学的顶点对应关系的建立

- 顶点对应关系在多边形形状渐变中占居非常重要的地位。
  - 一个好的对应关系可以指导插值的过程，并生成光滑流畅的中间帧；
  - 而一个不好的对应关系则会生成不自然的中间帧。
- 如果源多边形和目标多边形具有不同的位置、方向、大小或形状，在没有人工干预的情况下，要找到一个好的对应关系通常是件困难的事。当两个多边形的顶点个数不同时，问题就更加复杂。
- 基于模糊数学的顶点对应关系建立方法能够较好地处理多边形具有不同的位置、方向、大小、顶点数目的情形。
- 思想：用目标多边形建立一个与给定源多边形相似的模糊多边形集，然后在模糊集中查找多边形相似函数最大的多边形。

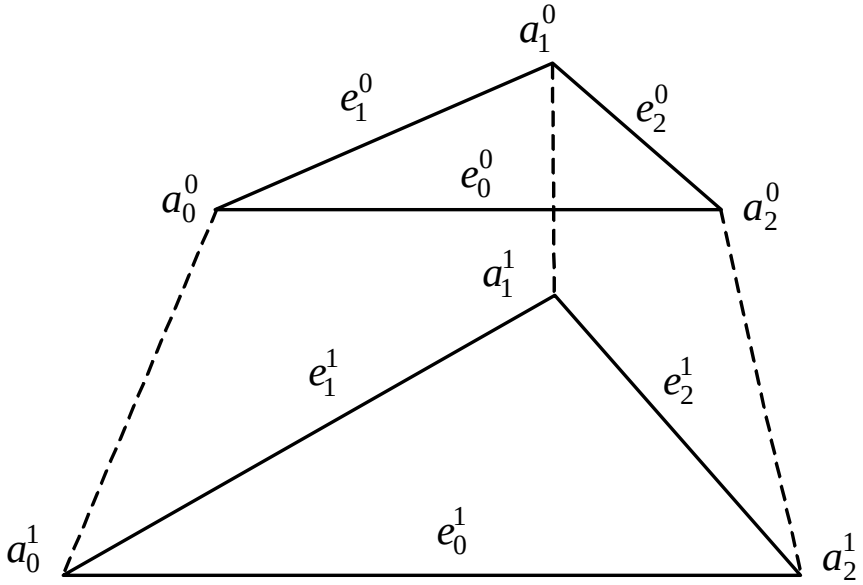
多边形相似函数

- 对于两个多边形 $P^0=\{a_0^0,a_1^0,\cdots,a_n^0\}$ 和 $P^1=\{a_0^1,a_1^1,\cdots,a_n^1\}$，如果它们*对应的角相等*且*对应的边成比例*，则称它们是相似的。

- 在一个模糊多边形集中，一个多边形与其余多边形的相似程度由多边形相似函数来决定。
  - 若多边形相似函数的值很高，则这两个多边形的相似程度很高；
  - 若多边形相似函数的值很低，则这两个多边形的相似程度很低。

多边形相似函数

- 多边形相似函数：对应角相同和对应边成比例的程度。
- 对于两个多边形，注意到如果由对应的角和形成该角的两条边组成的三角形相似，则这两个多边形相似。因此我们可以用这些三角形来定义相似函数。



两个相似三角形

多边形相似函数

- 若这两个三角形相似，则 $e_1^0 \times e_2^1 = e_1^1 \times e_2^0$,且 $a_1^0 = a_1^1$
- 若它们不完全相似，　则 $e_1^0 \times e_2^1 \neq e_1^1 \times e_2^0$, 且 $a_1^0 \neq a_1^1$
- 多边形的相似程度依赖于这种不同的程度。因此我们可根据这种不同来定义三角形的相似函数：

$$\mathrm{sim}_t = w_1 \times (1 - \frac{\left| e_1^0 \times e_2^1 - e_1^1 \times e_2^0 \right|}{e_1^0 \times e_2^1 + e_1^1 \times e_2^0}) + w_2 \times (1 - \frac{\left| a_1^0 - a_1^1 \right|}{360})$$

其中 $0 < w_1, w_2 < 1$ ， 且 $w_1 + w_2 = 1$ 。通常，我们可设 $w_1 = w_2 = 0.5$

多边形相似函数

- 上述三角形相似函数具有以下性质：
  1. $0 \le \mathrm{sim}_t \le 1$；
  2. 如果两个三角形相似，则 $\mathbf{sim}_t\mathbf{=1}$;
  3. 如果两个三角形较相似，则$\mathbf{sim}_t$较大;
  4. 如果两个三角形有很大不同，则$\mathbf{sim}_t$较小；

多边形相似函数

- 给定一顶点对应关系映射$P^0 \rightarrow P^1$，我们可以利用三角形的相似函数来定义多边形P^0和P^1的相似函数：

$$\mathrm{sim}_p(P^0, P^1) = \frac{1}{n+1}\sum_{i=0}^{n}\mathrm{sim}_t(T_i^0, T_j^1)$$

其中$j=\mathrm{map}(i)$,T_i^0和T_j^1为由多边形对应角a_i^0和a_j^1以及它们相邻的边e_i^0, e_{i+1}^0和e_j^1, e_{j+1}^1组成的三角形，$\mathrm{sim}_t(T_i^0, T_j^1)$为相应三角形的相似函数。为了方便起见，我们把由多边形的角和其相邻边组成的三角形称为多边形角。

多边形相似函数

- 函数Sim_p具有以下性质：

1. $0 \le \mathrm{sim}_p(P^0, P^1) \le 1$；
2. 如果两个多边形相似， 则Sim_p=1;
3. 如果如果两个多边形较相似，则Sim_p较大;
4. 如果两个多边形有很大不同，则Sim_p较小；

模糊集

- 若把一个多边形看成一系列有序的多边形顶点，则对于相同的多边形，取不同的顶点为起始点，我们可以得到多个多边形。不妨设多边形的顶点按顺时针方向排列。

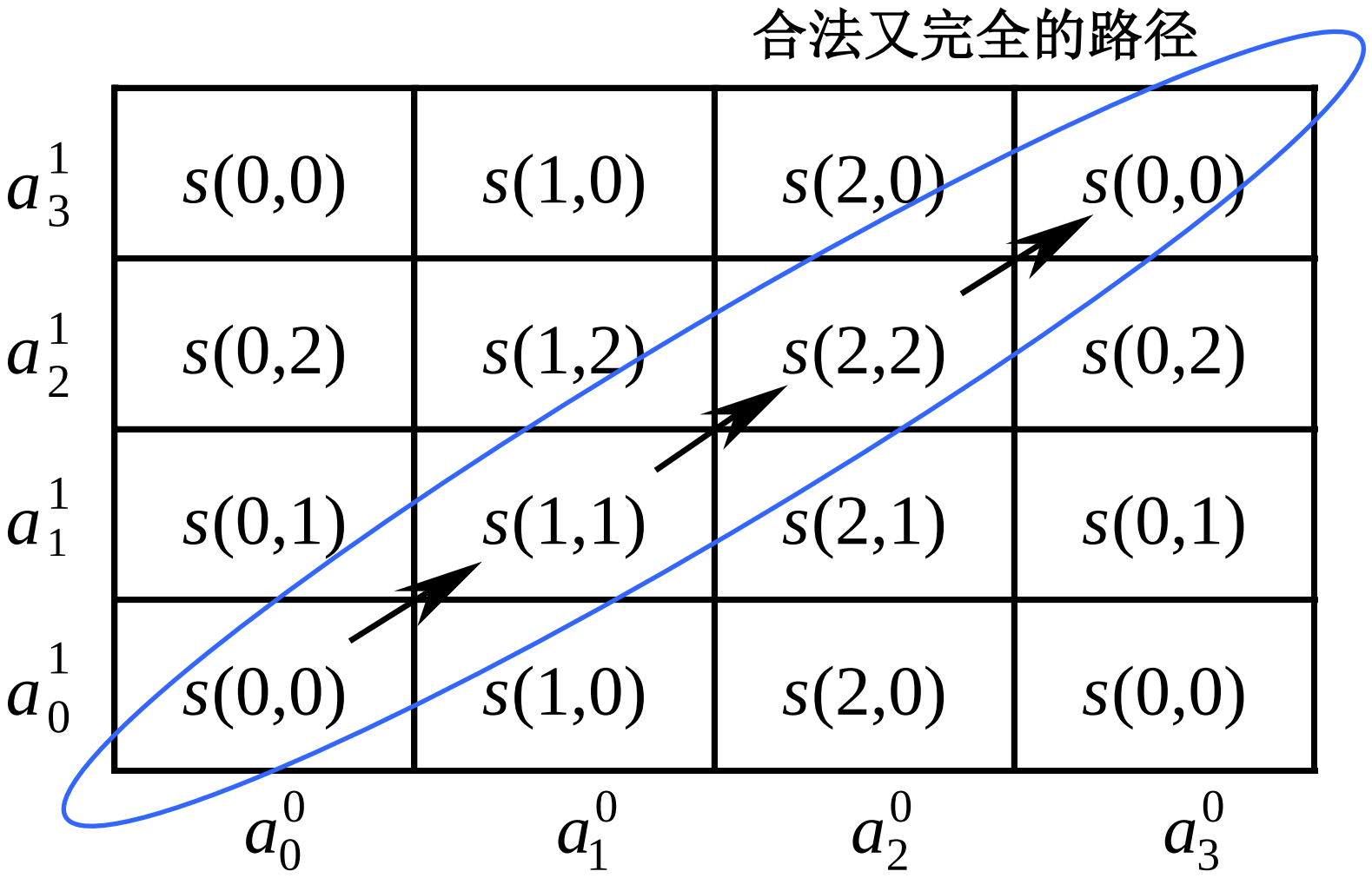
- 对于***n***个顶点的多边形，可得到***n***个多边形。例如，对于目标多边形P^1，我们可以得到***n*+1**个多边形 $P^{1,i}, i = 0,1,\cdots,n$，其中 $P^{1,i} = \{a_i^1, a_{(i+1) \bmod (n+1)}^1, \cdots, a_{(i+n) \bmod (n+1)}^1\}$ 。给定源多边形P^0，多边形集$P^{1,i}$和多边形相似函数Sim_p构成了模糊集$P=<x, Sim_p>$。属于该集合的多边形X和P^0的相似程度为$Sim_p(P^0, X)$。

图论求解方法

- 应用模糊数学技术，建立多边形P^0和P^1的对应关系问题转化成为在模糊集中寻找多边形P，使得相应的*隶属函数值最大*。
- **Zhang**采用了一种基于图论的方法来求解。
- 给定多边形 $P^0=\{a_0^0,a_1^0,\cdots,a_m^0\}$ 和 $P^1=\{a_0^1,a_1^1,\cdots,a_n^1\}$，$m\geq n$ ；每对对应多边形角的模糊相似度可由节点数为$(m+2)\times(n+2)$的图$G[(m+2),(n+2)]$来表示，其中 $G[i,j]$表示多边形角a_i^0和a_j^1的模糊相似程度，$0\leq i\leq m+1, 0\leq j\leq n+1$ ，

$$G[i,n+1]=G[i,0]\ ,\qquad G[m+1,j]=G[0,j]$$

- 我们称这样的图为模糊相似图。



多边形的模糊相似图，$s(i,j)=\mathrm{sim}_t(a_i^0,a_j^1)$

路径

- 在相似图中，一条*路径*的代价(cost)为该路径中每个节点相似度的和。
- **Zhang**发现，如果P^0中的每个顶点只对应于P^1中的一个顶点时，得到的结果较好（ P^0 的顶点数$m>=n$）。
- 这意味着在相似图中，不允许有向“北”、“南”的跨步，只允许“东”或“东北”的跨步。这样得到的路径称为是一条*合法的路径*。如果路径上的节点表示了多边形的顶点对应关系，则称该路径是*完全的路径*。
- 因此，使相似函数最大的两多边形顶点对应关系问题转化成为在模糊相似图中寻找一条最长的合法、完全的路径问题。

图论求解方法

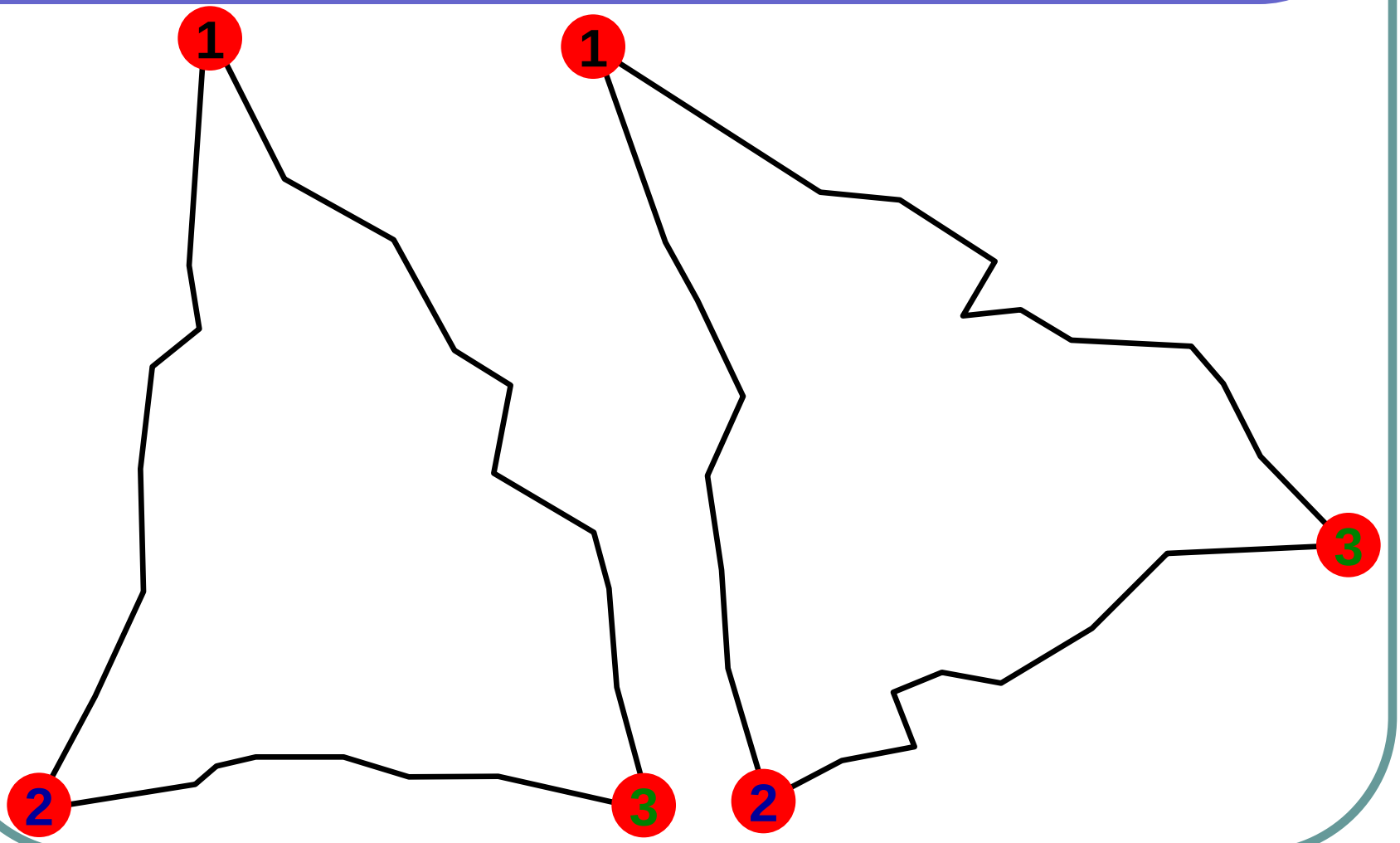
- 为了利用标准的图论算法，我们考虑图G的余图G^{-1}，其中图G^{-1}中的元素为$G^{-1}[i, j]=1- G[i, j]$;
- 显然，在图G中寻找最长合法路径问题变成了在图G^{-1}中寻找最短的合法路径问题。
- 给定图G^{-1}中初始节点$[i_1, j_1]$和终止节点$[i_2, j_2]$，我们可用标准的图论算法来决定连接这两个节点的最短合法路径。
- 初始和终止节点可由一对对应多边形角来给出。例如，如果我们已知a_0^0对应于a_0^1，则初始和终止节点为$[0, 0]$和$[m+1, n+1]$；
- 如果没有一对多边形角的对应关系是已知的，则我们可以让P^0中的a_0^0对应于P^1中的a_i^1，$i=0,1,2, \ldots, n$

多边形的对应关系由其中最短的一条来建立!

仿射变换的选取

- **Zhang的方法首先需求得一把源多边形变换到目标多边形的*好的仿射变换*。**
- **我们知道三对顶点决定了一个唯一的仿射变换。给定两个多边形，所有三对对应多边形顶点决定了一个仿射变换集。**
- **为了选择一个好的仿射变换，考虑由上述仿射变换构成的模糊仿射变换集$T=<y, \text{smooth}_t>$，其中仿射变换函数smooth_t表示把源多边形光滑过渡到目标多边形的“好坏”程度。**
- **我们的策略: 基于一对对应多边形角的好坏函数smooth_a来决定smooth_t**

仿射变换的选取



仿射变换的选取

- 给定一对对应角a_i^0和a_j^1，我们考虑对应三角形的如下因素来决定$smooth_a$：

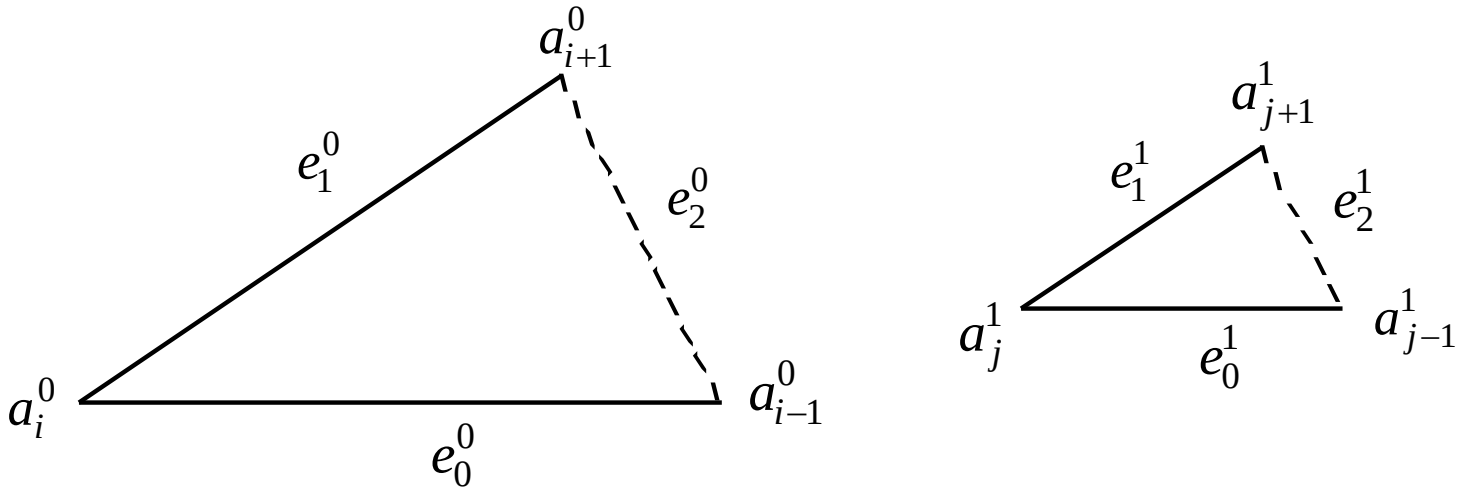
1. 三角形$T_i^0 = a_{i-1}^0 a_i^0 a_{i+1}^0$ 和 $T_j^1 = a_{j-1}^1 a_j^1 a_{j+1}^1$ 在形状和大小上的相似程度S；

$$S = w_1 \times (1 - \frac{\sum_{i=0}^{2} \left| e_i^0 - e_i^1 \right|}{\sum_{i=0}^{2} \left| e_i^0 + e_i^1 \right|}) + w_2 \times (1 - \frac{\sum_{i=0}^{2} \left| a_i^0 - a_i^1 \right|}{180})$$

$0 < w_1, w_2 < 1, w_1 + w_2 = 1$, 它们的缺省值都是0.5.

仿射变换的选取

2. 把T_i^0变换为T_j^1所需的最小旋转角R。它可由三角形的顶点来求得。
3. 两个三角形的面积之和与两个多边形面积之和之比A



一对对应的多边形角

仿射变换的选取

- 因为顶点 $a_{i-1}^0, a_i^0, a_{i+1}^0$和 $a_{i-1}^1, a_i^1, a_{i+1}^1$用来构成下面我们要讨论的插值过程中所需的局部坐标系，所以 a_i^0, a_i^1 的大小必须落于**[0, 180]**范围之内。

- 考虑到这个因素，当上述条件不满足时，我们把**smooth$_a$**置为**0**；否则，我们用***S***, ***R***和***A***来决定**smooth$_a$**

$$\text{smooth}_a = a \times S + b \times (1 - \frac{R}{180}) + c \times A$$

仿射变换的选取

- 给定三对对应角(a_i^0, a_i^1)，(a_j^0, a_j^1) 和 (a_k^0, a_k^1)，我们定义对应仿射变换的“好坏”函数为：

$$\text{smooth}_t(T(i,j,k)) = \text{smooth}_a(a_i^0, a_i^1) \times \text{smooth}_a(a_j^0, a_j^1) \times \text{smooth}_a(a_k^0, a_k^1)$$

- 在模糊仿射变换集中“好坏”函数最大的仿射变换，即为我们所选取的好的仿射变换（三对对应顶点）。

- 注意，这样决定的三对对应顶点并不需要是相邻的。

仿射变换的插值

- 现在讨论仿射变换的插值问题。采用齐次矩阵表示，一个仿射变换可写成如下形式：

$$(u',v')=(u,v)\begin{pmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{pmatrix}+(a_{31},a_{32})=(u,v)\mathbf{A}+\mathbf{T}$$

在几何变换中，**T**为平移向量，而**A**为表示旋转、比例缩放和剪切变换的复合变换。

- 插值仿射变换的的一个方法是线性插值复合旋转矩阵和平移向量，即

$$(1-t)\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}+t\begin{pmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{pmatrix}$$

和$t(a_{31},a_{32})$，其中$\mathbf{0}\le t\le\mathbf{1}$

仿射变换的插值

- 但线性插值并不能得到光滑的变换。为了得到更流畅的结果，我们把矩阵A分解成如下形式：

$$\mathbf{A} = \mathbf{BC} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

其中B为一纯粹的旋转矩阵。

- 给定矩阵A，矩阵B和C的计算方法如下：

$$\mathbf{B} = \mathbf{A} + sign(\det(\mathbf{A})) \begin{pmatrix} a_{22} & -a_{21} \\ -a_{12} & a_{11} \end{pmatrix}$$

$$\mathbf{C} = \mathbf{B}^{-1}\mathbf{A}$$

仿射变换的插值

- 由于**B**为一纯粹的旋转矩阵，可把它写成

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

的形式。从而我们得到一更流畅的复合矩阵插值公式：

$$(1-t)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} \cos(t\theta) & -\sin(t\theta) \\ \sin(t\theta) & \cos(t\theta) \end{pmatrix} \begin{pmatrix} tc_{11} & tc_{12} \\ tc_{21} & tc_{22} \end{pmatrix}$$

- 该方法使显式插值多边形的方向成为可能。

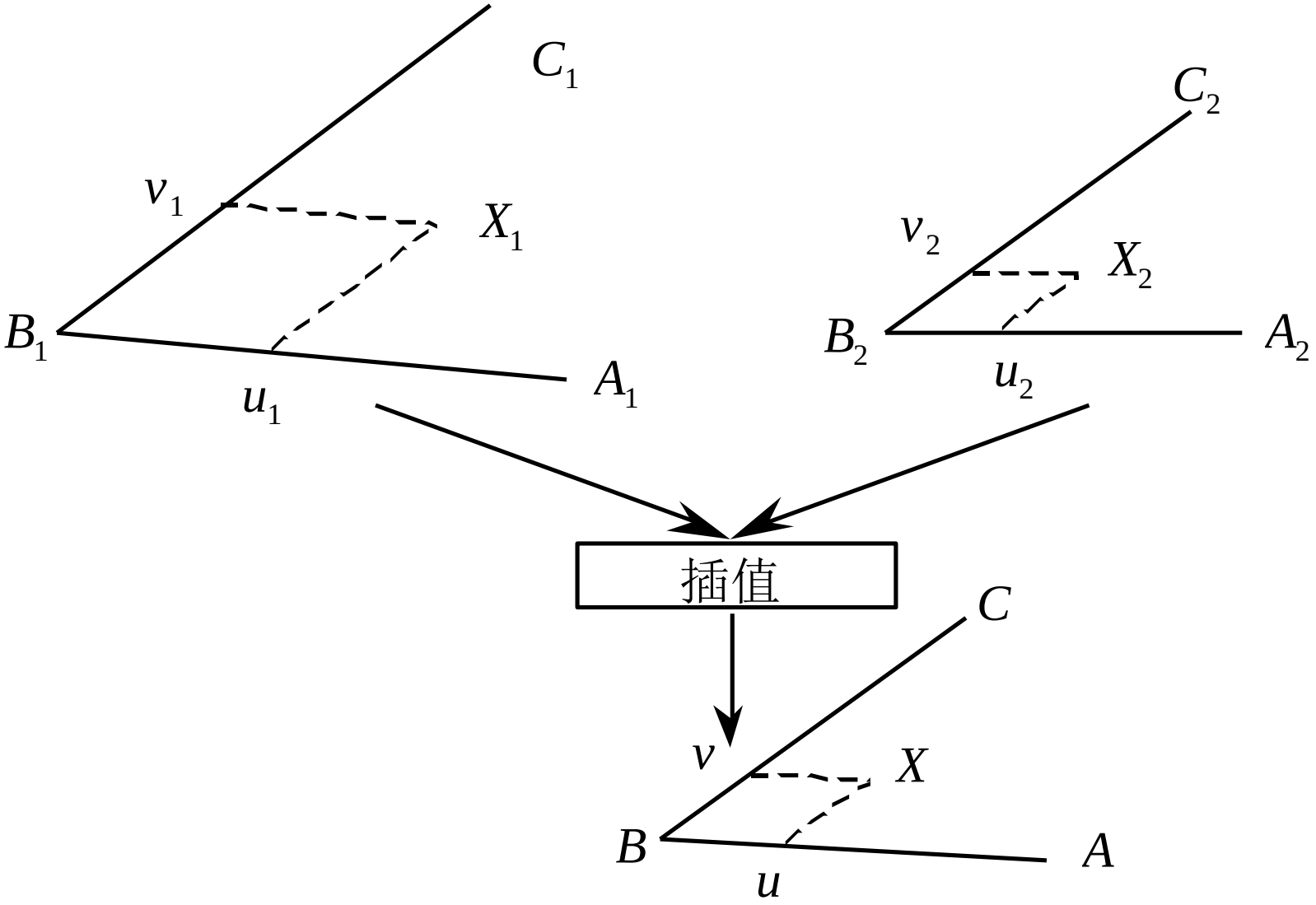
计算中间帧多边形

- 现在我们讨论怎样把中间帧的仿射变换应用于源多边形P^0，从而得到中间帧的多边形。
- 在仿射变换的选取中，我们得到了源多边形和目标多边形的三对对应顶点$(A_1,A_2),(B_1,B_2),(C_1,C_2)$，它们构成了这两个多边形的局部坐标系。
- 设X_1,X_2为源多边形和目标多边形上的点，它们在相应局部坐标系上的规范化局部坐标分别为(u_1,v_1)和(u_2,v_2)，则

$$X_1 = B_1 + u_1 \times (A_1 - B_1) + v_1 \times (C_1 - B_1)$$

$$X_2 = B_2 + u_2 \times (A_2 - B_2) + v_2 \times (C_2 - B_2)$$

- 给定顶点$X_1,X_2,(u_1,v_1)$和(u_2,v_2)很容易根据上述方程组求得。



规范化的局部坐标系

计算中间帧多边形

- 为了计算顶点X_1和X_2的插值点X，我们根据前面所述的方法插值得到中间帧的仿射变换，并把它作用于顶点A_1，B_1和C_1来生成三个中间帧顶点A，B和C 。
- 取X点的规范化局部坐标 (u,v)为(u_1,v_1)和(u_2,v_2) 的线性插值，即

$$u=(1-t)u_1+tu_2,\qquad v=(1-t)v_1+tv_2$$

- 从而得到插值点X的位置，

$$X=B+u\times(A-B)+v\times(C-B)$$

- 把上述方法应用到多边形的所有顶点对，我们得到给定多边形的中间帧多边形。这种插值方法可处理包括平移、旋转、比例缩放和剪切变换在内的所有的仿射变换。

DEMO

More Examples

DEMO

Morphing in After Effects

The End