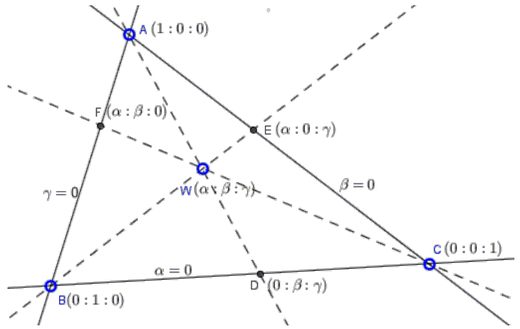
Bremen

CG
VR

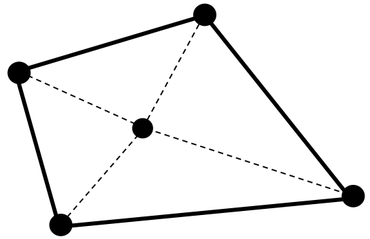
Advanced Computer Graphics
Generalized Barycentric Coordinates



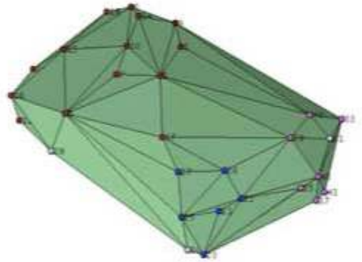
G. Zachmann
University of Bremen, Germany
cgvr.informatik.uni-bremen.de

Different Kinds of Generalizations of Barycentric Coordinates

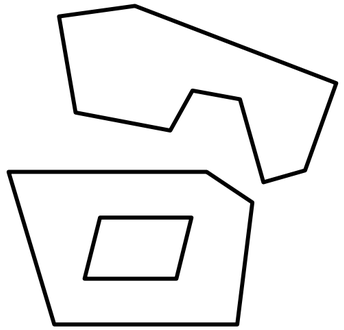
1. What about barycentric coords of a point inside a (convex) 2D polygon with $k > 3$ sides?



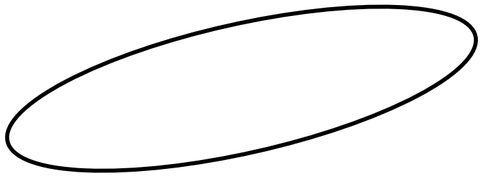
2. Similarly, what does one do with polyhedra in d dimensions with $k > d+1$ vertices? (i.e., non-simplices)



3. How to handle non-convex (concave) areas?



4. What does one do when the area is not limited by a polyline (piecewise linear curve), but rather by a smooth, closed convex curve?



Generalized Barycentric Coords with $k>3$ Vertices in 2D

- Definition:

  Let Ω be a convex polygon in $\mathbb{R}^2$ with n vertices $P_1, \ldots, P_n$, $n \geq 3$ in CCW order ("*counter-clockwise*").

  A set of functions $\lambda_i : \Omega \to \mathbb{R}$ are called barycentric coordinates, if for all $X \in \Omega$ the following conditions are met:

  1. Partition of unity: $\sum_{i=1}^{n} \lambda_i(X) = 1$

  2. Linear precision: $\sum_{i=1}^{n} \lambda_i(X) P_i = X$

  3. Convex combination: $\forall i = 1 \ldots n : \ \lambda_i(X) \geq 0$

- Other desirable characteristics:
  - "Good behavior" with respect to the area outside of Ω
  - Smoothness: λ_i should be in C^∞
  - Affine invariance

The Interpolation Property

- Theorem:

  Any such (generalized) barycentric coordinates have the following property: let data values f_i be given at the vertices P_i ; then, the function

$$f(X) = \sum_{i=1}^{n} \lambda_i(X) \cdot f_i$$

  is actually an *interpolating* (and not simply approximating) function, in other words

$$\forall i : \; f(P_i) = f_i \; .$$

  This is called the interpolation property.

Proof

- We will show: $\lambda_i(P_j) = \delta_{ij}$ (Kronecker Delta)

1. Because of properties 1 and 2, one can reproduce all linear functions using barycentric combinations. Let f be just such a linear function. Then

$$f(Q) = \begin{pmatrix} a \\ b \end{pmatrix} Q + c$$

Prop. 2 (lin. precision)

Prop. 1 (partition of unity)

$$= \begin{pmatrix} a \\ b \end{pmatrix} \sum \lambda_i(Q) P_i + c \sum \lambda_i(Q)$$

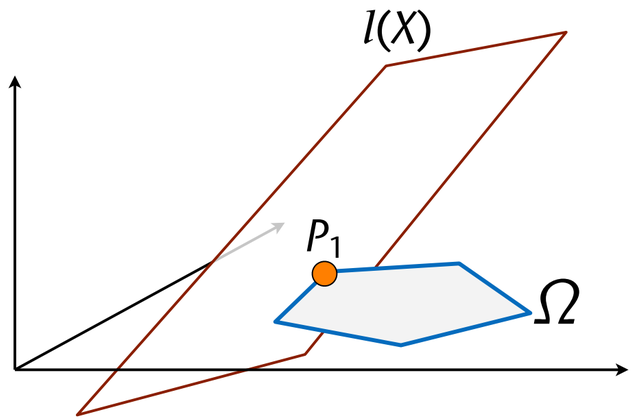
$$= \sum \lambda_i(Q) \left(\begin{pmatrix} a \\ b \end{pmatrix} P_i + c \right)$$

$$= \sum \lambda_i(Q) f(P_i)$$

2. We define a plane (= linear fct) $l(X)$ so that

$$l(P_1) = 0 \quad \text{and} \quad \forall i \geq 2: \; l(P_i) > 0$$

This is possible because Ω is a convex polygon



3. From step 1, it follows that:

$$l(X) = \sum_{i=1}^{n} \lambda_i(X) \cdot l(P_i) = \sum_{i=2}^{n} \lambda_i(X) \cdot l(P_i)$$

$l(P_1) = 0$

4. l was specially chosen so that

$$0 = l(P_1) = \sum_{i=2}^{n} \lambda_i(P_1) l(P_i) \;\Rightarrow\; \forall i \geq 2: \; \lambda_i(P_1) = 0$$

$l(P_i) \gneqq 0$, for $i \geq 2$

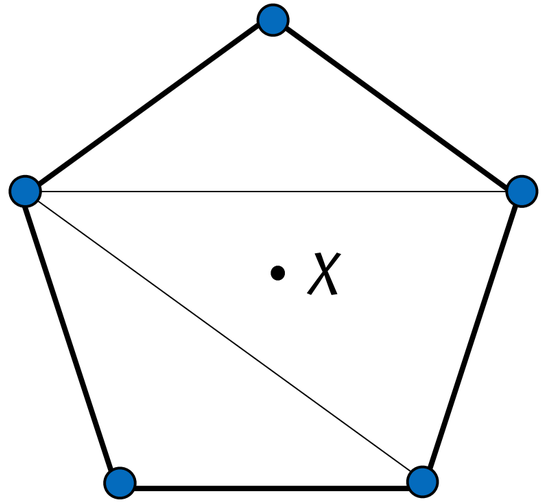
5. Because of property 1:

$$1 = \sum_{i=1}^{n} \lambda_i(P_1) = \lambda_1(P_1)$$

(4)

Trivial Construction of Generalized Barycentric Coords

- Triangulate the polygon (in any way possible)
- Determine the barycentric coordinates with respect to the triangle in which X is located



- Problems:
  - The triangulation is not unique

➢Hence, these generalized barycentric coordinates are *not unique*!

  - These barycentric coordinates are only $\mathcal{C}^0$-continuous
  - The extension for points outside of the polygon is unclear

Construction of Generalized Barycentric Coords

- Goal: construct barycentric coordinates for any convex polygon in 2D
- Observation: since $\sum \lambda_i = 1$, we rewrite

$$\sum \lambda_i P_i = X \Leftrightarrow \sum \lambda_i \cdot (P_i - X) = 0$$

- If one has functions $w_i = w_i(X)$ for which the following two properties hold,

$$\sum w_i \cdot (P_i - X) = 0 \qquad (1)$$

$$\text{and} \qquad \forall i : \; w_i \geq 0 \qquad (2)$$

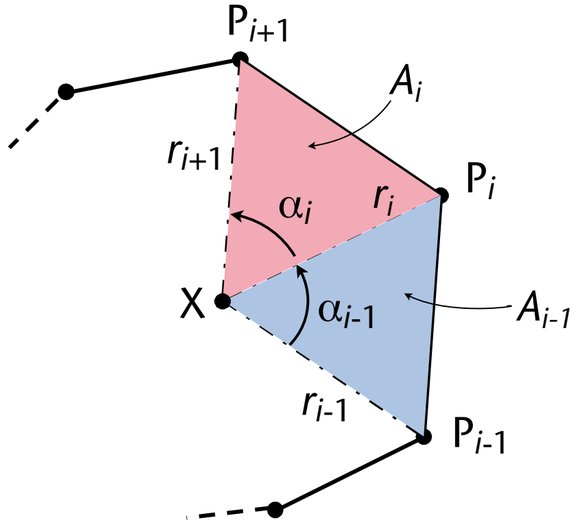
then one can easily make "real" barycentric coordinates by just letting

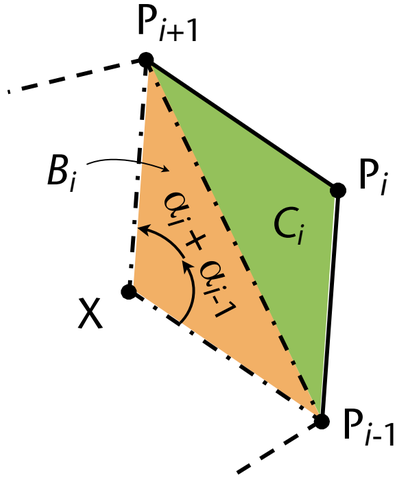
$$\lambda_i = \frac{w_i}{\sum_{i=1}^{n} w_i}$$

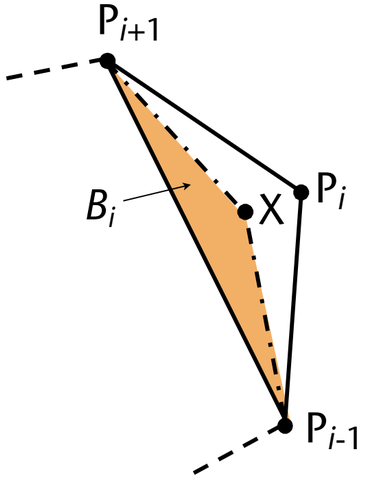
➢ Look for functions w_i that fulfill conditions (1) & (2)

Notations

- Define the following lengths and areas:
  - $r_i := \|P_i - X\|$
  - $A_i := A_i(X) = \mathcal{A}(\Delta X P_i P_{i+1}) = \frac{1}{2} \sin \alpha_i \cdot r_i r_{i+1}$
  - $B_i := B_i(X) = \mathcal{A}(\Delta X P_{i+1} P_{i-1}) = -\frac{1}{2} r_{i-1} r_{i+1} \sin(\alpha_{i-1} + \alpha_i)$
    - Note: the sign of B_i is *negative*, if X is *outside* of $\Delta P_{i-1} P_i P_{i+1}$!
  - $C_i := C_i(X) = \mathcal{A}(\Delta P_{i-1} P_i P_{i+1}) = A_i + A_{i-1} + B_i$
- In the following, all indices are meant to be "modulo n", i.e. $P_i := P_{i \bmod n}$ and so $P_{n+1} = P_1$, $P_{-1} = P_{n-1}$

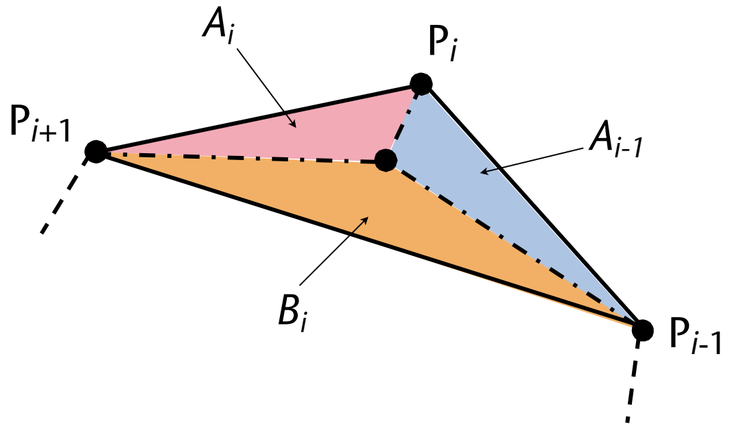






- Reminder: in a triangle (in the case where X is inside $\Delta P_{i-1}P_iP_{i+1}$),
  $\frac{A_i}{C_i}, \frac{B_i}{C_i}, \frac{A_{i-1}}{C_i}$ are the barycentric coordinates

- So, $\frac{A_i}{C_i}(P_{i-1} - X) + \frac{B_i}{C_i}(P_i - X) + \frac{A_{i-1}}{C_i}(P_{i+1} - X) = 0$



- Therefore, A_i, B_i, A_{i-1}, applied in the correct order, fulfill conditions (1) and (2):

$$A_i(P_{i-1} - X) + B_i(P_i - X) + A_{i-1}(P_{i+1} - X) = 0$$

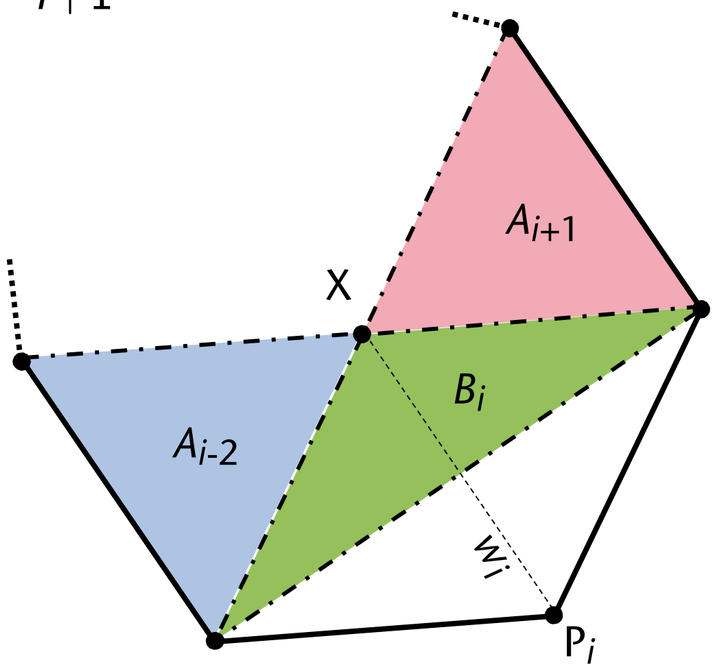
"Barycentric coordinates"

- Idea: consider the series of triangles $\Delta\, P_{i-1}\, P_i\, P_{i+1}$
- Approach: compute the weighted average of the (non-normalized) barycentric coordinates wrt. each of these triangles:

$$w_i := w_i(X) = \sigma_{i-1} A_{i-2} + \sigma_i B_i + \sigma_{i+1} A_{i+1}$$

where $\sigma_i := \sigma(X)$ can be any function (for the time being)

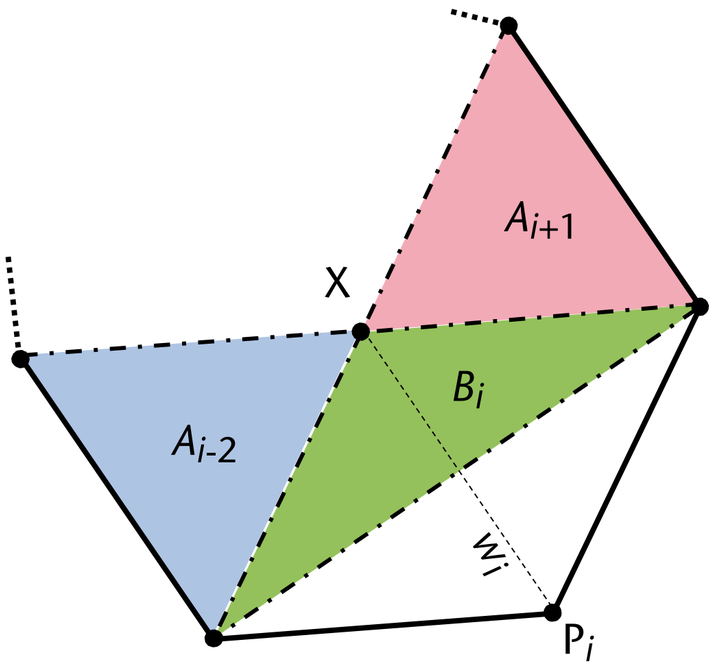
- So, every vertex is involved in 4 or 5 barycentric coordinates, respectively



- Proposition 1:

  These $w_i = \sigma_{i-1} A_{i-2} + \sigma_i B_i + \sigma_{i+1} A_{i+1}$

  fulfill condition (1)



- Proof: $\sum_{i=1}^{n} w_i (P_i - X) = \sum_{i=1}^{n} \sigma_i \left(A_i (P_{i-1} - X) + B_i (P_i - X) + A_{i-1} (P_{i+1} - X) \right) = 0$

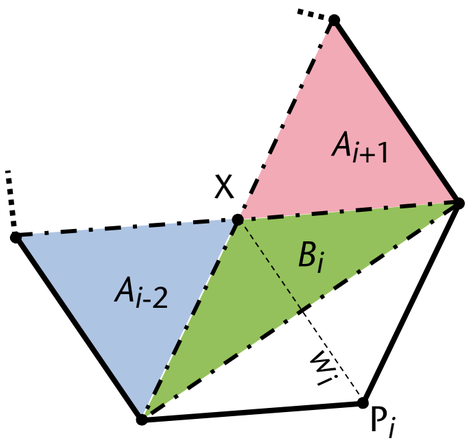
  (remember: index arithmetic "wraps around")

- Proposition 2:

  If the polygon is convex and $\forall i : \sigma_i(X) > 0$

  then $\sum w_i(X) > 0$

  for all values of X in the interior of the polygon.



- Proof:

$$\sum_{i=1}^{n} w_i(X) = \ldots = \sum_{i=1}^{n} \sigma_i(X) \cdot C_i > 0 \quad , \quad \text{since} \quad \forall i : \; C_i > 0$$

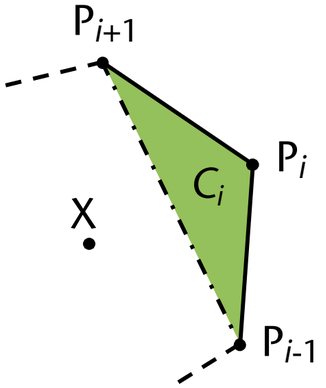
Plug in definition of the w_i, change summation indices appropriately, remember indices are mod n

- Note: $\sigma_i > 0$ alone does not guarantee that $\sum w_i(X) > 0$!
  - Also, the convexity of the polygon is crucial...

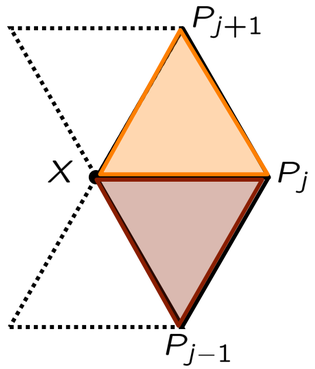
- Note: with $\sum w_i > 0$, the normalization of the w_i's to get the λ_i's always works
  - Reminder: $w_i > 0$ was a requirement of condition (2) of the definition

- Next goal: look for appropriate σ_i , such that $w_i > 0$ and $\sigma_i > 0$

Some Candidates



- Naive approach: choose $\sigma_i = \frac{1}{C_i}$
  - Thus $\sum w_i(X) \equiv n$
  - Unfortunately, $w_i(X) > 0$ is not guaranteed (why?)
  - Consequence: the interpolation property doesn't hold ☹

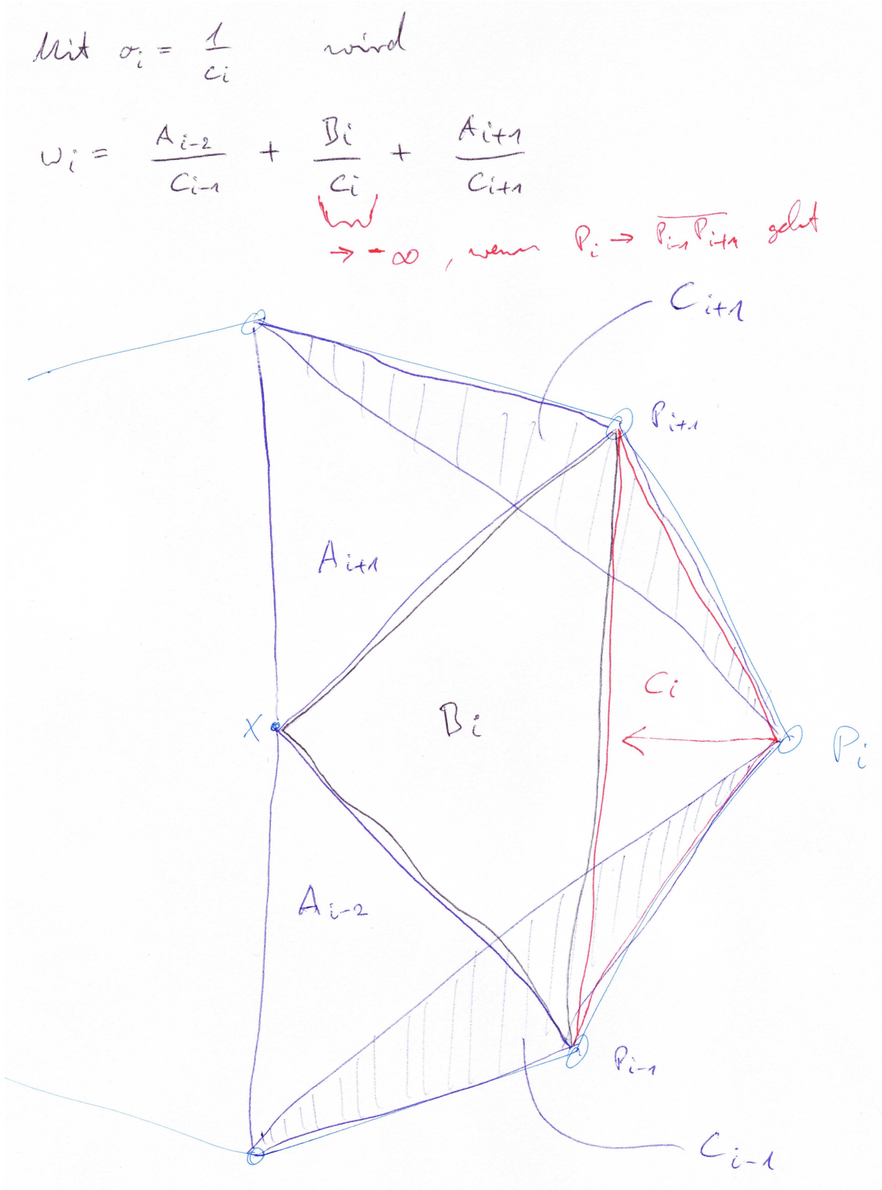


- Wachspress coordinates: choose $\sigma_i(X) = \dfrac{1}{A_{i-1}A_i}$
  - Thus

$$w_i := \frac{\mathcal{F}(\Delta P_{i-1}P_iP_{i+1})}{\mathcal{F}(\Delta XP_{i-1}P_i)\cdot\mathcal{F}(\Delta XP_iP_{i+1})}$$

  - Disadvantage: they behave badly in a non-convex polygon, since $\sum w_i(X) = 0$ is possible, which means that the λ_i's have a pole there

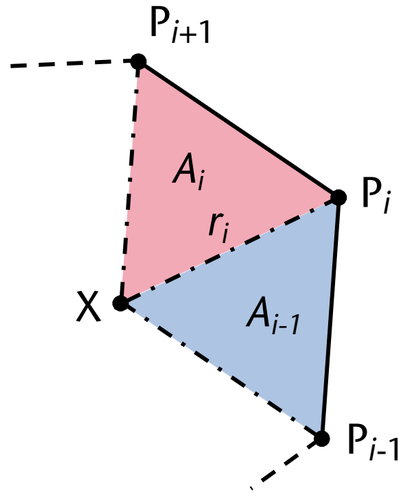
- Explanation why $w_i < 0$ is possible with the naïve choice

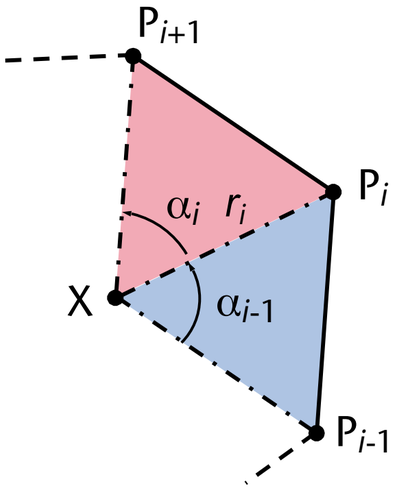


The Best Candidate (as of Today)

- Mean value coordinates (MVCs):
  - Choose $\sigma_i = \dfrac{r_i}{A_{i-1}A_i}$
  - Thus, $w_i(X) = \dfrac{r_{i-1}A_i + r_i B_i + r_{i+1}A_{i-1}}{A_{i-1}A_i}$
  - With some trigonometric substitutions, we get:

$$w_i = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{r_i/2}$$





- Claim: the MVCs are barycentric coordinates for all X in the interior of the polygon
- Proof: if X is in the interior, then all $\sigma_i > 0$ and all $w_i > 0$

- Beginning of a proof of the trigonometric equation for w_i :

$$
\begin{aligned}
w_i &= \sigma_{i-1} A_{i-2} + \sigma_i B_i + \sigma_{i+1} A_{i+1} \\
&= \frac{r_{i-1}}{A_{i-2} A_{i-1}} A_{i-2} + \frac{r_i}{A_{i-1} A_i} B_i + \frac{r_{i+1}}{A_i A_{i+1}} A_{i+1} \\
&= \frac{r_{i-1}}{A_{i-1}} + \frac{r_i}{A_{i-1} A_i} B_i + \frac{r_{i+1}}{A_i} = \cdots
\end{aligned}
$$

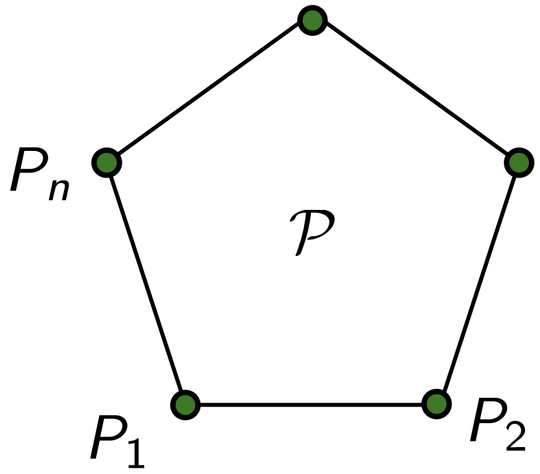
- Then, for the A'sand B_i use the sin-formula for the surface area and use trigonometric identities

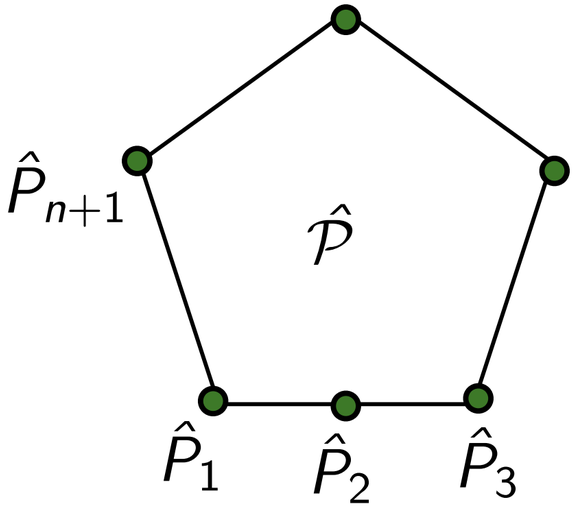
Extension to Non-Convex Polygons

- Lemma (w/o proof):
  Let $\mathcal{P}$ be a given convex polygon. Denote the MVCs of a point X w.r.t. $\mathcal{P}$ with w_i, i=1…n.
  Now refine $\mathcal{P}$ with the insertion of a point. Denote this refined polygon by $\hat{\mathcal{P}}$.
  Denote the MVCs of X w.r.t. $\hat{\mathcal{P}}$ with $\hat{w}_i$, i=1…n+1.

  Then

$$\sum_{i=1}^{n+1} \hat{w}_i = \sum_{i=1}^{n} w_i$$

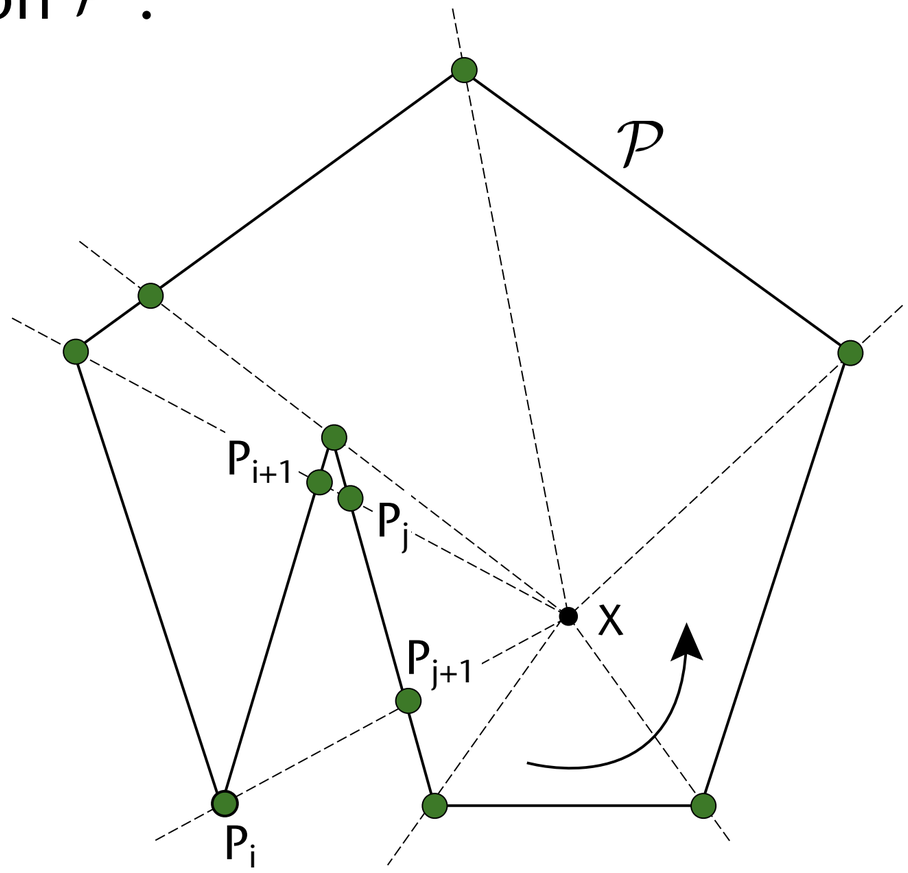




- Corollary: the $\lambda's$ are also well-defined for $\hat{\mathcal{P}}$

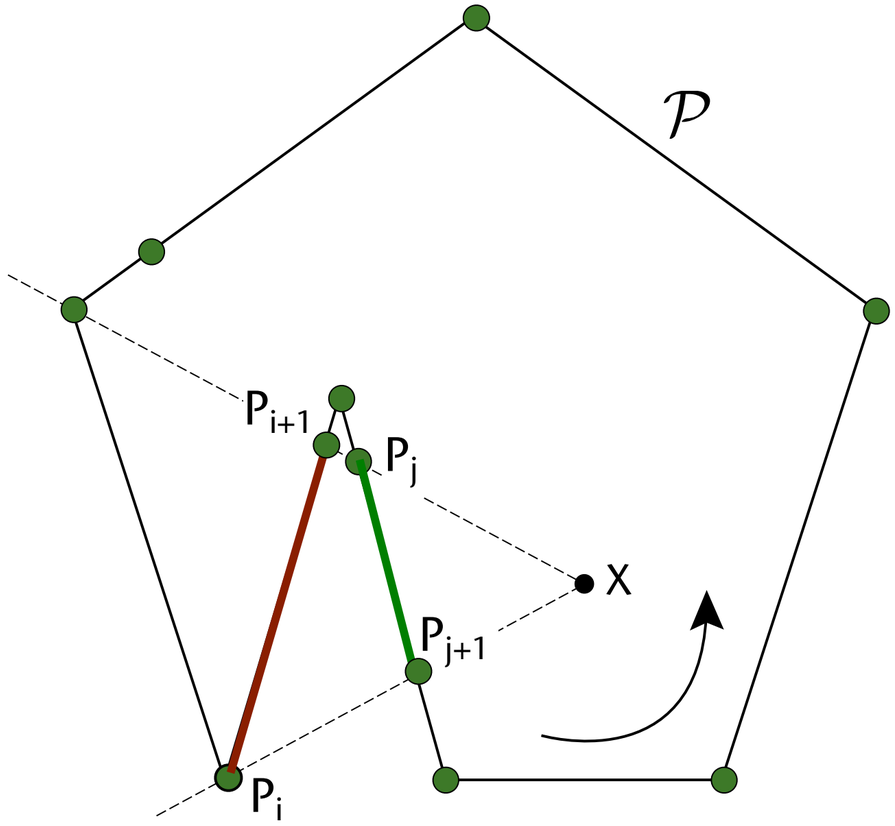
- Theorem:
  Let $\mathcal{P}$ be any simple polygon (possibly non-convex).
  For all X not located on the edge of the polygon $\mathcal{P}$:

$$\sum w_i(X) \neq 0$$



- Proof:
  - Assumption: X is in the interior of $\mathcal{P}$
  - Draw rays from X through the vertices of $\mathcal{P}$ $\longrightarrow$ refinement of $\mathcal{P}$
  - Name the refinement $\mathcal{P}$ again with vertices $P_1,\ldots,P_n$.

- Classify edges into "entry edge" (red) or "exit edge" (green)
- Can be done easily either by checking the orientation of the edge w.r.t. *X*, or while following the ray from *X* outward
- Observation: for every entry-edge there is an exit-edge *closer* to X
- For every edge P_iP_{i+1}, define the following value



$$k_i = \left(\frac{1}{r_i} + \frac{1}{r_{i+1}}\right) \tan\frac{\alpha_i}{2}$$

where the *signs* of the angles α_i are determined by the *orientation* of the respective edges

- One sees immediately that: $\sum k_i = \frac{1}{2} \sum w_i$

  (summands are combined just a little differently in the trig. equation)

- For an edge P_iP_{i+1} , our sign-choosing rule yields:

  if exit-edge $\longrightarrow k_i > 0$

  if entry-edge $\longrightarrow k_i < 0$

- Let P_iP_{i+1} be an entry-edge; then a corresponding exit-edge P_jP_{j+1} must exist, and it is closer to X

- The following holds for their angles:

$$\alpha_i = -\alpha_j$$

- The following holds for their distances:

$$r_j \leq r_{i+1} \ \wedge \ r_{j+1} < r_i \quad \text{or} \quad r_j < r_{i+1} \ \wedge \ r_{j+1} \leq r_i$$

- With that, we have

$$k_j = \left(\frac{1}{r_j} + \frac{1}{r_{j+1}}\right) \tan\frac{\alpha_j}{2} \quad > \quad \left(\frac{1}{r_i} + \frac{1}{r_{i+1}}\right) \tan\frac{-\alpha_i}{2} = -k_i$$

- In other words: for every k_i of an entry-edge, there is a k_j of an exit-edge so that $k_i + k_j > 0$
- Thus $\sum k_i > 0$

and with that $\sum w_i > 0$

for all X in the interior of $\mathcal{P}$

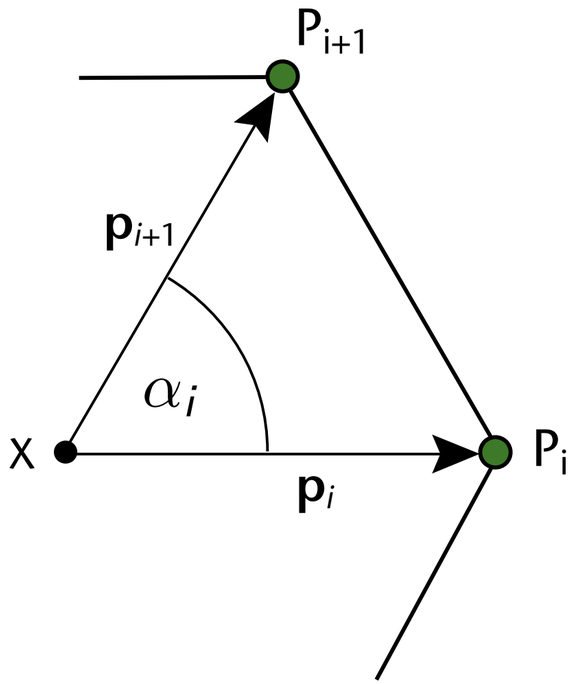
- Furthermore, we can show that for non-convex polygons the *mean value coordinates* have the following properties:
  - λ_i are well-defined for X on the edge of the polygon
  - $\lambda_i(P_j) = \delta_{ij}$
  - $\lambda_i \in \mathcal{C}^\infty$ with the exception of those at P_j; there they are only $\mathcal{C}^0$

Implementation

- Practical calculation of $\tan\left(\frac{\alpha_i}{2}\right)$:

$$\tan\frac{\alpha_i}{2} = \frac{\sin\alpha_i}{1+\cos\alpha_i}$$

$$\cos\alpha_i = \frac{\mathbf{p}_i \cdot \mathbf{p}_{i+1}}{|\mathbf{p}_i| \cdot |\mathbf{p}_{i+1}|} \qquad \sin\alpha_i = \frac{|\mathbf{p}_i \times \mathbf{p}_{i+1}|}{|\mathbf{p}_i| \cdot |\mathbf{p}_{i+1}|}$$

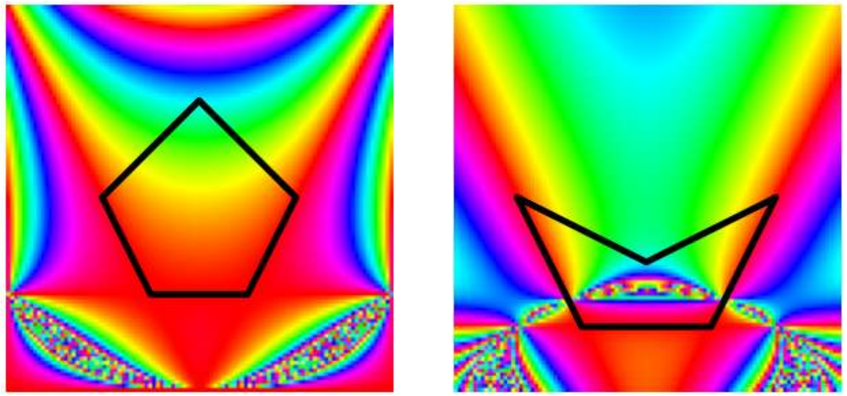


Thus: $\tan\frac{\alpha_i}{2} = \frac{|\mathbf{p}_i \times \mathbf{p}_{i+1}|}{|\mathbf{p}_i| \cdot |\mathbf{p}_{i+1}| + \mathbf{p}_i \cdot \mathbf{p}_{i+1}}$

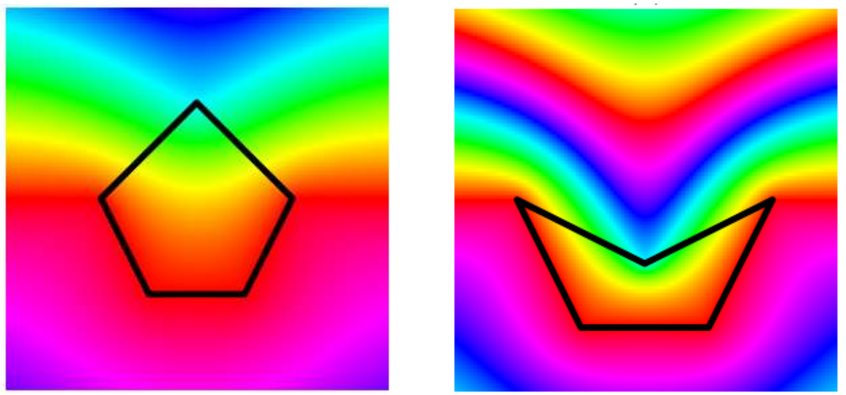
- If $|\mathbf{p}_i \times \mathbf{p}_{i+1}| = 0$, then *X* is located on an edge $\longrightarrow$ special treatment:
  1. X = P_i or X = P_{i+1}
  2. Otherwise: use linear interpolation between P_i und P_{i+1}

Application: Interpolation of Colors

- Given:
  - A simple polygon (not necessarily convex)
  - A color at every corner
- Task: color the interior of the polygon with "nice" color gradients (a common task in drawing software, for example)
- Solution:
  - Calculate barycentric coordinates for every pixel in the interior of the given polygon
  - Interpolate the colors of the vertices using these barycentric coords



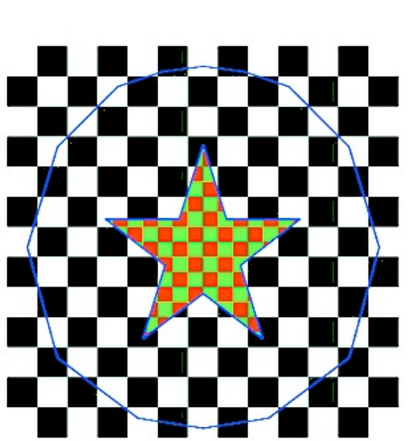
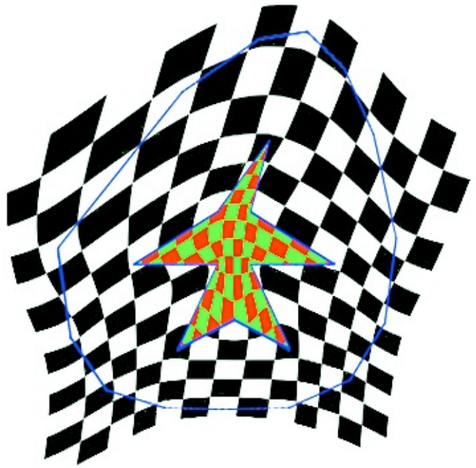
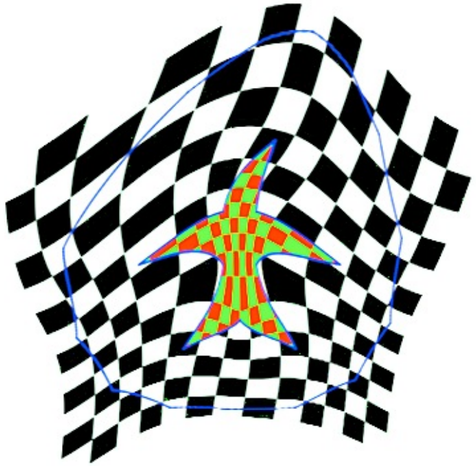
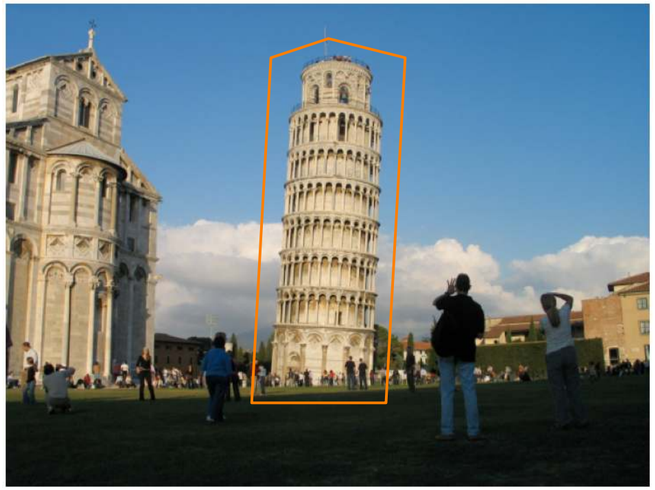
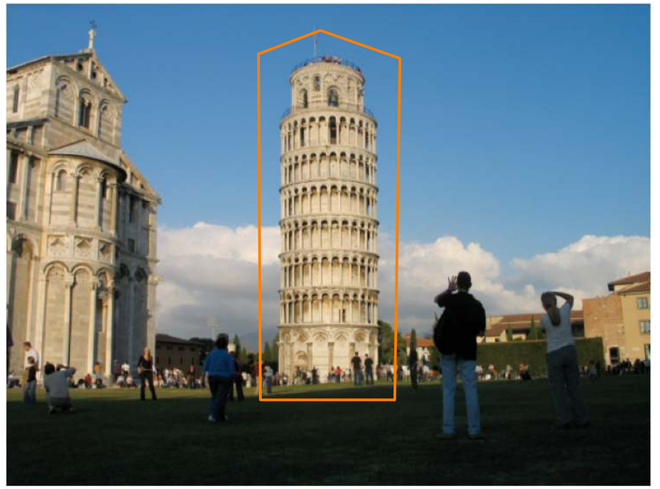
Wachspress



Mean Value Coordinates

Application: Image Warping

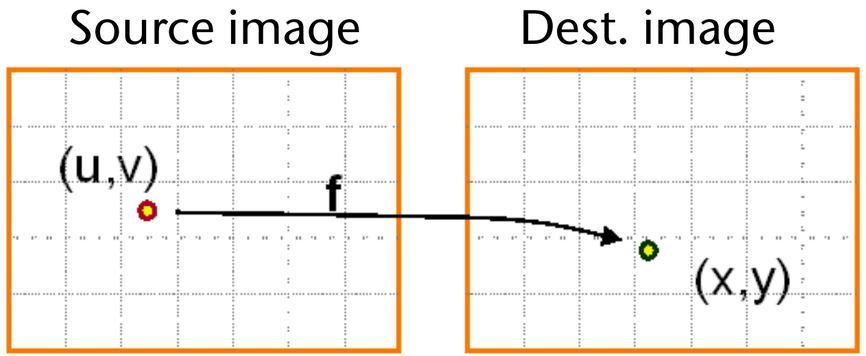
- Task: warp the given image by transforming a "control polygon"
- Examples:

Algorithm for Resampling Images

- First idea: *"forward mapping"*

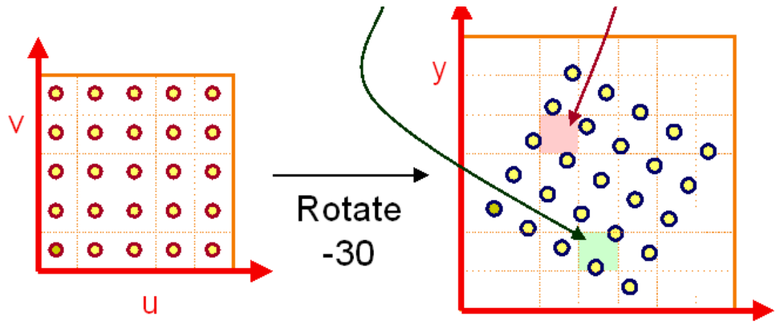
```
for u = 0 .. umax:
  for v = 0 .. vmax
    x,y = f(u,v)
    dst(x,y) ← src(u,v)
```



- Construction of *f* :
  - Determine barycentric coordinates of a point *X* w.r.t. the control polygon in the source image
  - Interpolate the *positions* of the vertices of the control polygon in the destination image ⟶ *X'*
- Problems:

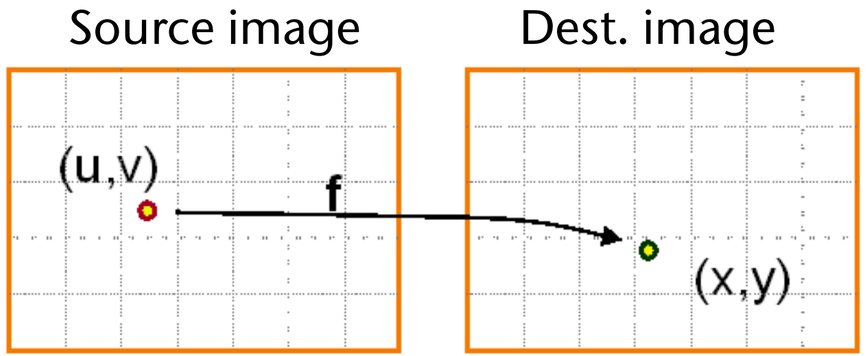
Several source pixels might map to same dest. pixel

Some dest. pixels might not get covered by any source pixel

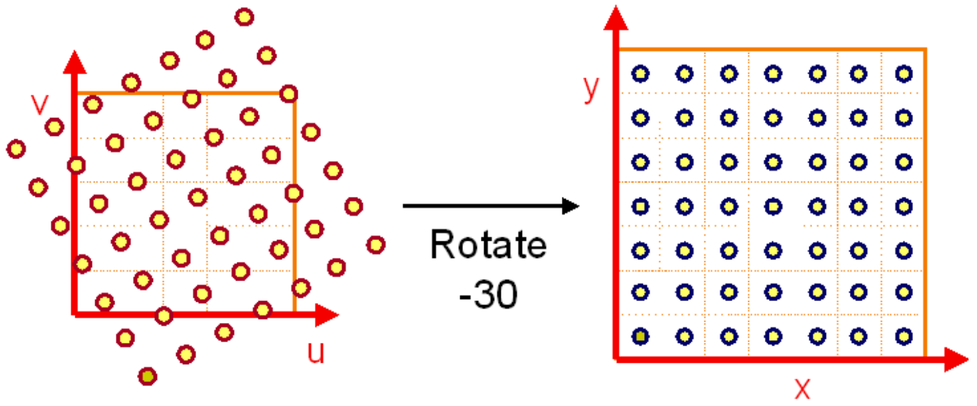


- Better idea: *"reverse mapping"*

```
for x = 0 .. xmax:
  for y = 0 .. ymax
    u,v = f^-1(x,y)
    dst(x,y) ← src(u,v)
```

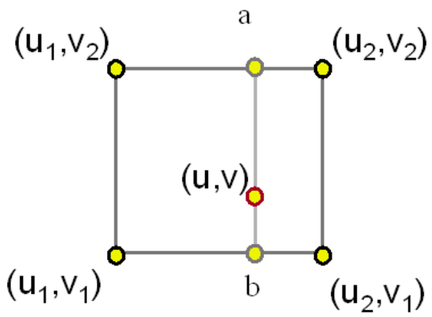


- Use barycentric interpolation again to construct f^{-1}
  - Just swap the roles
- Minor problem: (u,v) are not pixel coords.; rather, they are located "between" pixels
  - One has to do *"resampling"* or interpolation in the source image



Resampling

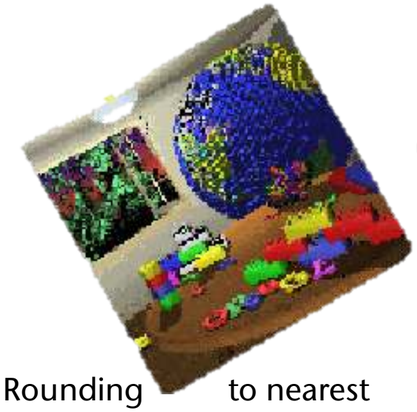
- Simplest solution: choose nearest pixel (i.e., rounding coords)
  - Produces big artifacts (aliasing, "jaggies", Moirée effects)
- The second-simplest solution: bi-linear interpolation



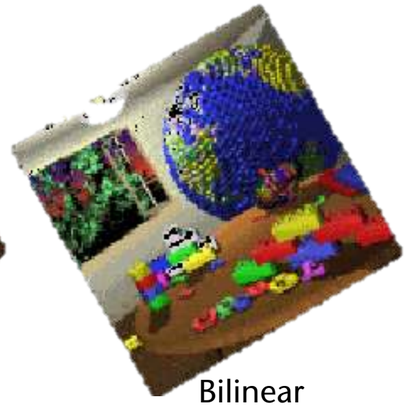

```
for x = 0 .. xmax:
  for y = 0 .. ymax
    u, v = f^-1(x,y)
    a = lin.interp. between src(u1,v2) and src(u2,v2)
    b = lin.interp. between src(u1,v1) and src(u2,v1)
    c = lin.interp. between a and b
    dest(x,y) ← c
```

- Examples:

Original



Rounding to nearest



Bilinear

- Better yet: Gaussian convolution

- Additional examples:

- Today fully-integrated in software:

Application: Morphing

- Given: two isomorphic triangle meshes M_1 and M_2 , i.e. ...
  1. Exactly the same number of vertices and triangles; and
  2. A correspondence $\Phi : V_1 \rightarrow V_2$ such that

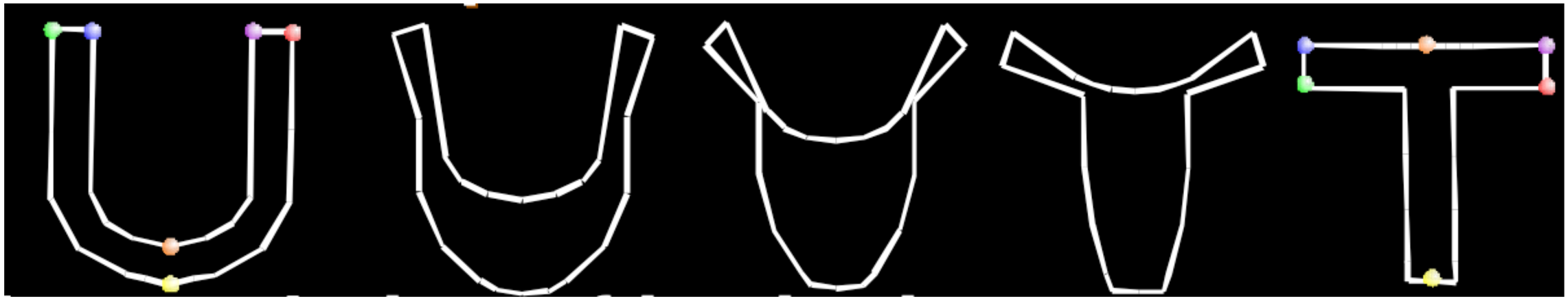
$$P, Q, R \text{ is triangle in } M_1 \Leftrightarrow$$
$$\Phi(P), \Phi(Q), \Phi(R) \text{ is triangle in } M_2$$

- Task: a uniform "deformation" of mesh M_1 in M_2
  - Because of the correspondence, it is sufficient to manipulate the coordinates of the vertices from V_1 continuously (for example, across 1000 time steps), so that in the end V_2 is reached
- Terminology: M_1 and M_2 are also called *"morph targets,"* or *"source"* and *"target"*

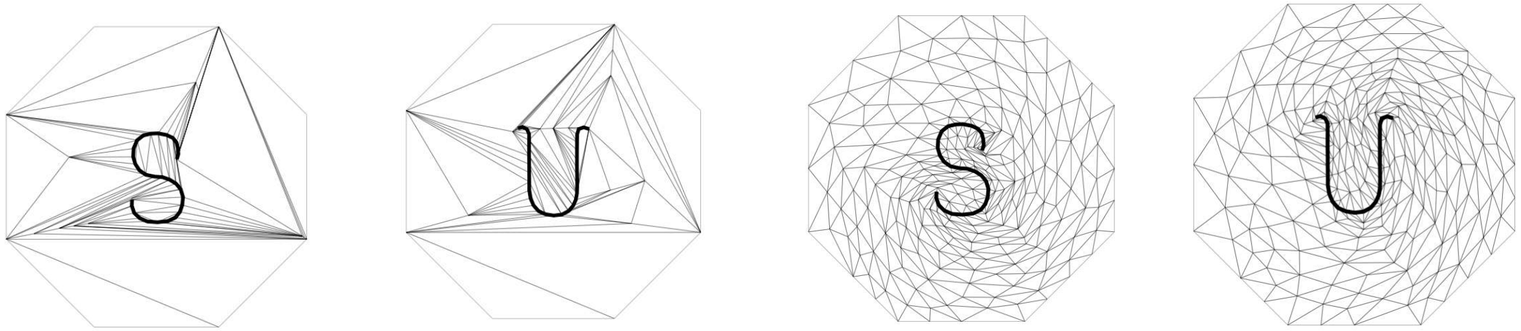
- Let $t \in [0,1]$ be the "morph parameter"
- Naive solution: linear interpolation of the vertices

$$P(t) = (1-t)P + t\Phi(P)$$

- 2D example:



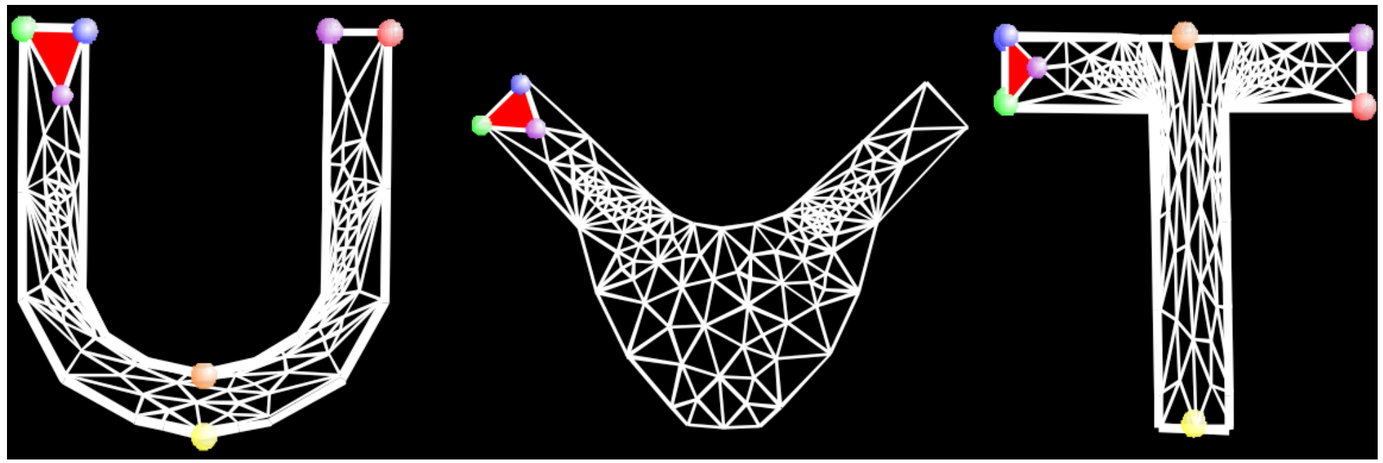
- Enclose both morph targets in a common, fixed polyline:



With none or few additional points

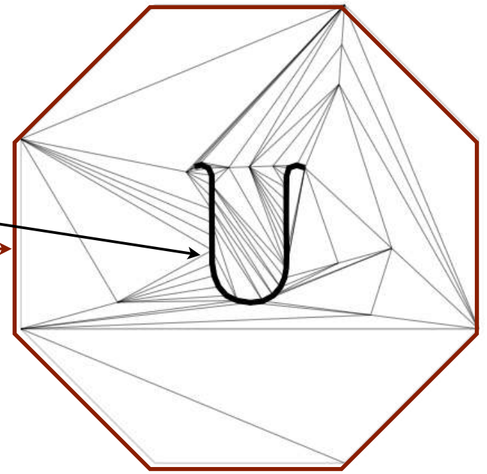
Many additional (Steiner) points

- You can also add inner points:



- Assumption for now: both meshes M_1 and M_2 are 2D in the plane

- Terms:
  - Inner vertices $V_I = \{P_i \mid i = 1 \ldots n\}$
  - Boundary points $V_B = \{P_i \mid i = n+1 \ldots n+k\}$
  - Let $N = n + k$, E = set of edges



- Using generalized barycentric coordinates, set up a system of linear equations for all vertices (for both M_1 and M_2):
  - For every $P_i \in V_I$, $i = 1 \ldots n$

    calculate $\lambda_{ij} > 0 \; \forall (i,j) \in E$, i.e. the barycentric coords of P_i

    wrt. its neighbors P_j , and set $\lambda_{ij} = 0 \; \forall (i,j) \notin E$

- Therefore, each P_i is the barycentric sum of its neighbors:

$$P_i = \sum_{j=1}^{N} \lambda_{ij} P_j \ , \quad i = 1 \ldots n$$

- Writing this slightly differently gives:

$$P_i - \sum_{j=1}^{n} \lambda_{ij} P_j = \sum_{j=n+1}^{n+k} \lambda_{ij} P_j \ , \quad i = 1 \ldots n$$

- Since $P_i = (x_i, y_i, z_i)$, this yields the 3 systems:

$$\underbrace{\begin{pmatrix} 1 & -\lambda_{12} & \cdots & -\lambda_{1n} \\ -\lambda_{21} & 1 & \cdots & \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} \lambda_{1,n+1} x_{n+1} + \cdots + \lambda_{1,n+k} x_{n+k} \\ \vdots \\ \vdots \end{pmatrix}}_{\mathbf{b}}$$

  - Analogously for y and z coordinates

- The (simple) idea:
  1. Interpolate the λ's: $\lambda_{ij}^{(t)} = (1-t)\lambda_{ij}^{(1)} + t\lambda_{ij}^{(2)}$
  2. Solve the 3 systems for every $t \longrightarrow$ yields $P_i(t)$, $i = 1,\ldots,n$
- A more sophisticated idea ("intrinsic morphing"):
  1. Interpolate the α's and r's (= angles & distances in the mesh):

$$\alpha_{ij}^{(t)} = (1-t)\alpha_{ij}^{(1)} + t\alpha_{ij}^{(2)} \quad , \quad r_{ij}^{(t)} = (1-t)r_{ij}^{(1)} + tr_{ij}^{(2)}$$

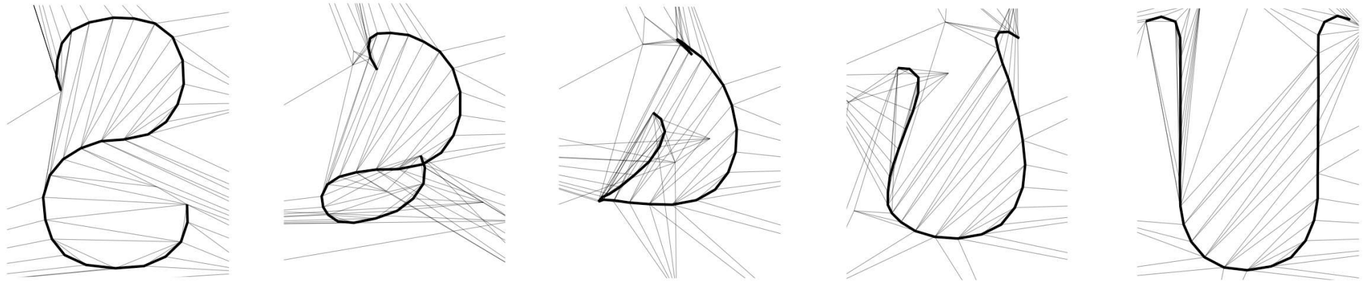
  2. From that, calculate $\lambda(t)$'s according to the definition of MVC's
  3. Solve the 3 linear systems
- Q: how many variables are interpolated in each variant (for a specific t)?

Notes Towards an Implementation

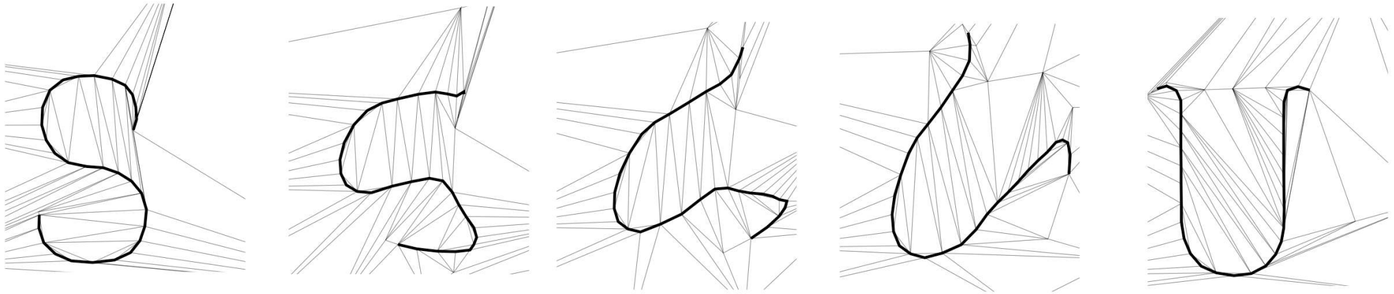
- The matrix *A* is not necessarily symmetric
- It is sparsely populated
- It is not a band matrix
- Use an iterative solver
  - Initialize it with the solution of step *t*-1

Results

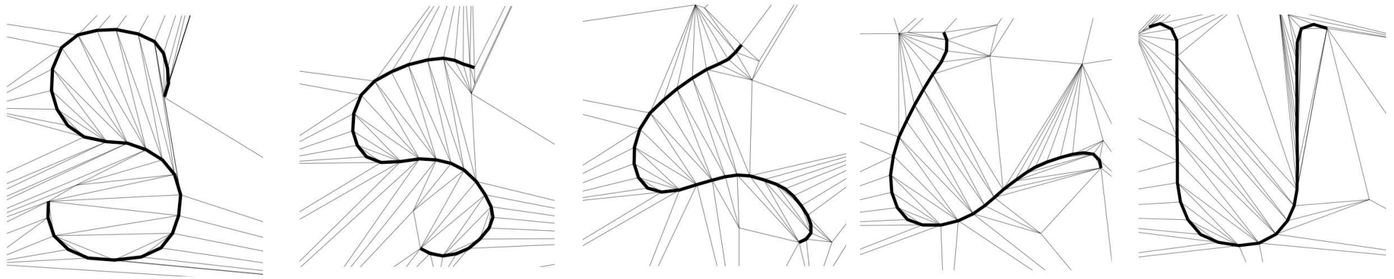
- Linear interpolation of vertices:



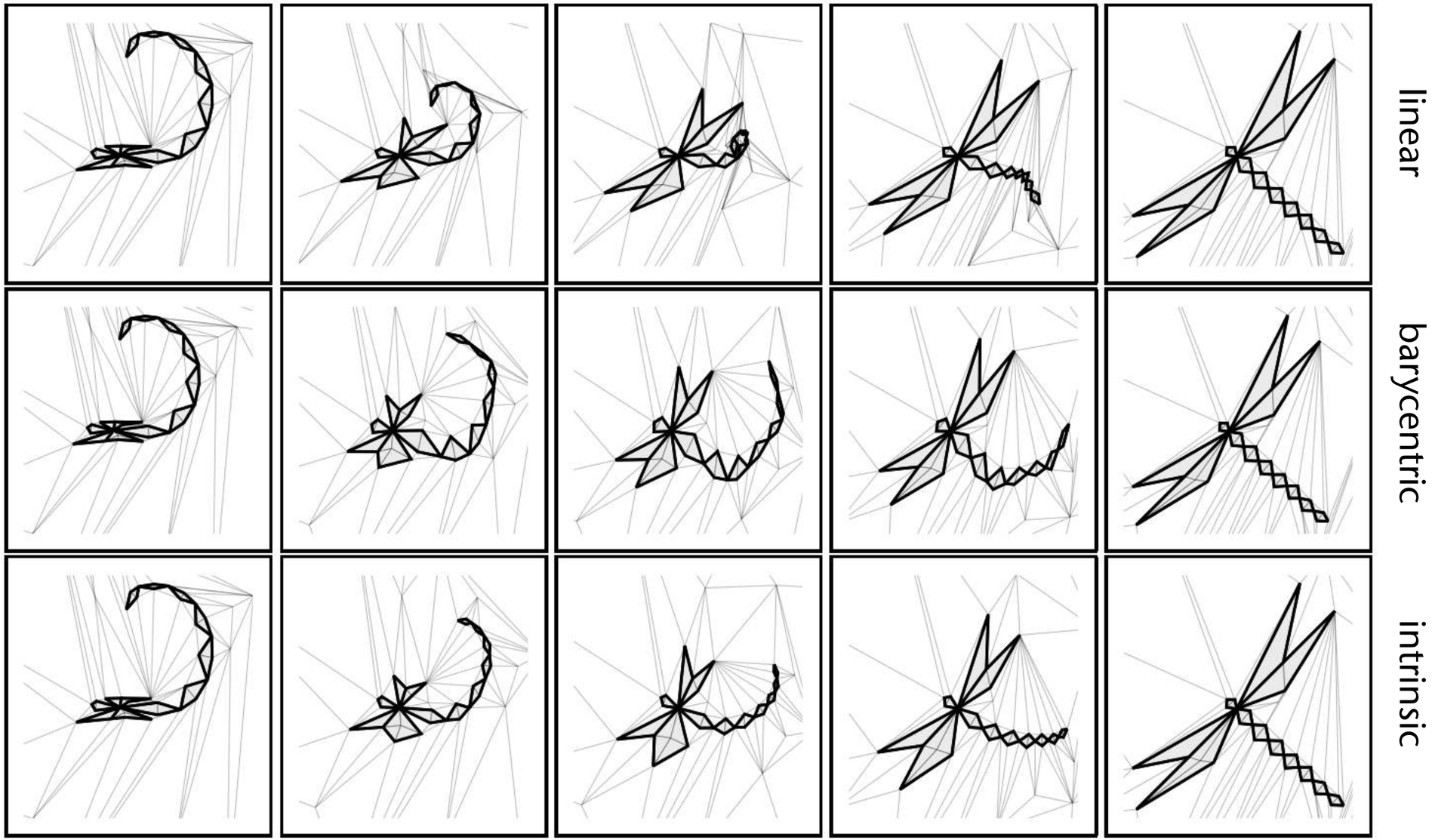
- Linear interpolation of barycentric coordinates:



- Intrinsic interpolation:



Additional Examples

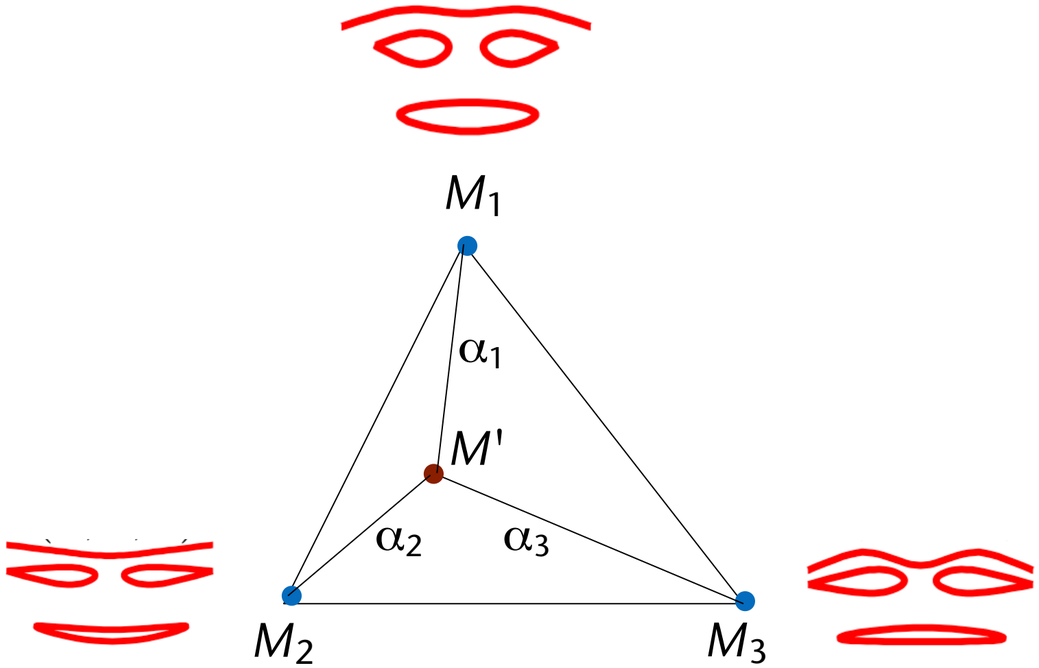


Simultaneous Morphing of Multiple Targets

- Given *n* morph targets M_k (meshes)
- Task: determine an "in-between"

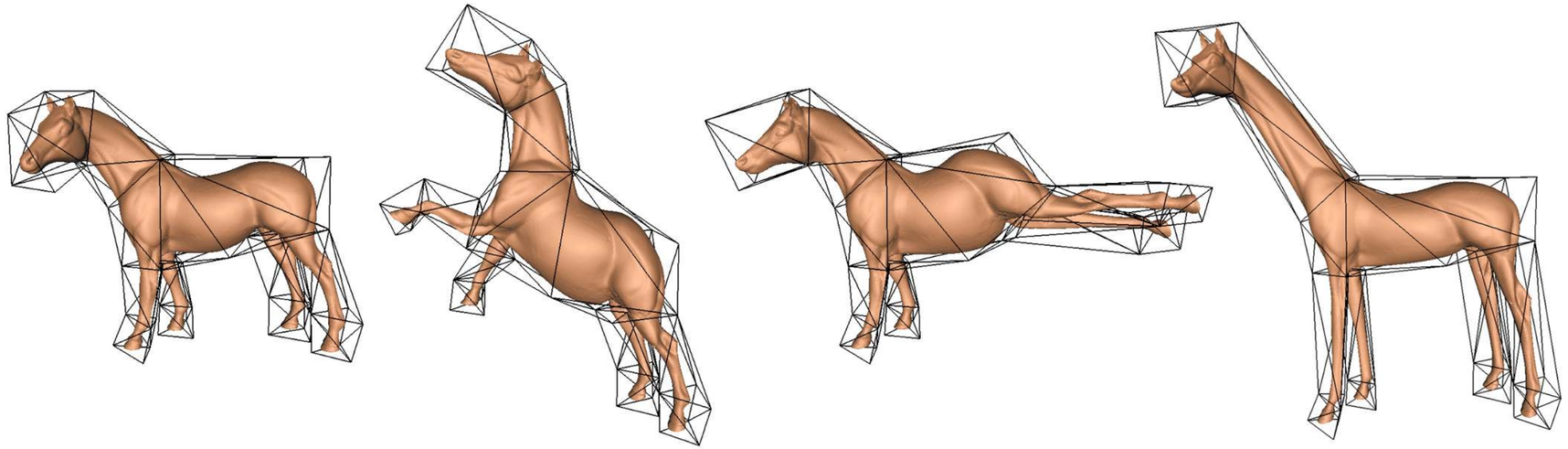
$$M' = \sum \alpha_k M_k$$

- General idea:
  1. Determine the barycentric coordinates, $\lambda_{ij}^{(k)}$, of all vertices of all M_k w.r.t. a fixed control polygon (or control polyhedron in 3D)
  2. Interpolate the λ's: $\lambda'_{ij} = \sum \alpha_k \lambda_{ij}^{(k)}$
  3. Set up and solve the linear systems



Application: 3D Shape Deformation

- Given: a mesh
- Task: targeted deformation of individual parts of the surface
- Example:



- The "cages" (a.k.a. "control mesh") determines the deformation (and is set, for example, by the animator)
- Solution: ...

Literature

- Take a look at papers on the class's homepage!
  - Under "Online Literature"