Geometric Modeling

Summer Semester 2010

Rational Spline Curves

Projective Geometry · Rational Bezier Curves · NURBS

UNIVERSITÄT DES SAARLANDES

M2CI CLUSTER OF EXCELLENCE

max planck institut informatik

Overview...

Topics:

- Polynomial Spline Curves
- Blossoming and Polars
- Rational Spline Curves
  - Some projective geometry
  - Conics and quadrics
  - Rational Bezier Curves
  - Rational B-Splines: NURBS

Some Projective Geometry

Projective Geometry

A very short overview of projective geometry:

- The computer graphics perspective
- Formal definition

Homogeneous Coordinates

Problem:

- Linear maps (matrix multiplication in $\mathbb{R}^d$) can represent...
  - Rotations
  - Scaling
  - Sheering
  - Orthogonal projections
- ...but not:
  - Translations
  - Perspective projections
- This is a problem in computer graphics:
  - We would like to represent compound operations in a single, closed representation

Translations

"Quick Hack" #1: Translations

- Linear maps cannot represent translations:
  - Every linear map maps the zero vector to zero $\mathbf{M0} = \mathbf{0}$
  - Thus, non-trivial translations are non-linear
- Solution:
  - Add one dimension to each vector
  - Fill in a one
  - Now we can do translations by adding multiples of the one:

$$\mathbf{Mx} = \begin{pmatrix} r_{11} & r_{21} & t_x \\ r_{12} & r_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} r_{11} & r_{21} \\ r_{12} & r_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \\ 1 \end{pmatrix}$$

Normalization

Problem: What if the last entry is not 1?

- It's not a bug, it's a feature...
- If the last component is not 1, divide everything by it before using the result

$$\mathbf{x} \rightarrow \begin{pmatrix} \omega\mathbf{x} \\ \omega \end{pmatrix}$$

Cartesian coordinates
(Euclidian space)

homogenous coordinates
(projective space)

$$\frac{1}{\omega}\mathbf{x} \leftarrow \begin{pmatrix} \mathbf{x} \\ \omega \end{pmatrix}$$

Notation

Notation:

- The extra component is called the *homogenous component* of the vector.
- It is usually denoted by ω:
  - 2D case:

    $$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} \omega x \\ \omega y \\ \omega \end{pmatrix}$$

  - 3D case:

    $$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \rightarrow \begin{pmatrix} \omega x \\ \omega y \\ \omega z \\ \omega \end{pmatrix}$$

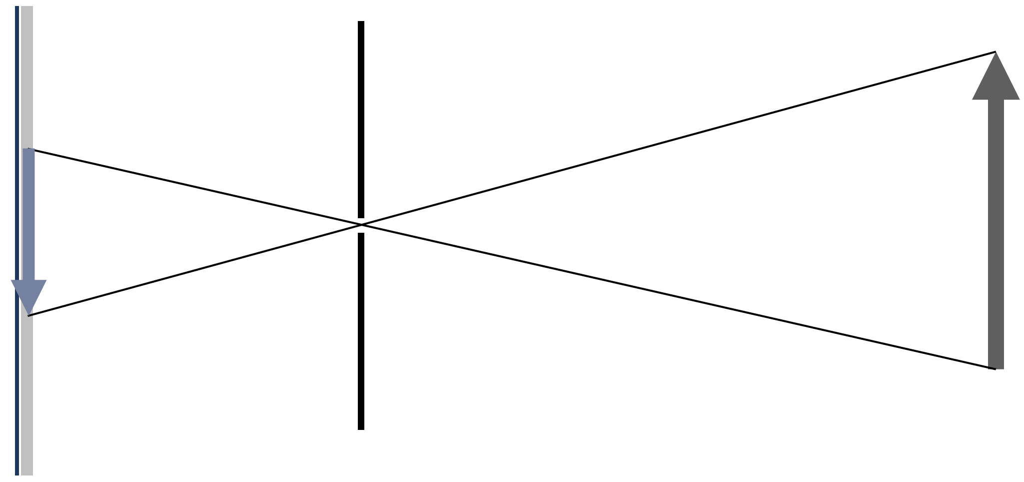
  - General case:

    $$\mathbf{x} \rightarrow \begin{pmatrix} \omega \mathbf{x} \\ \omega \end{pmatrix}$$

Perspective Projections

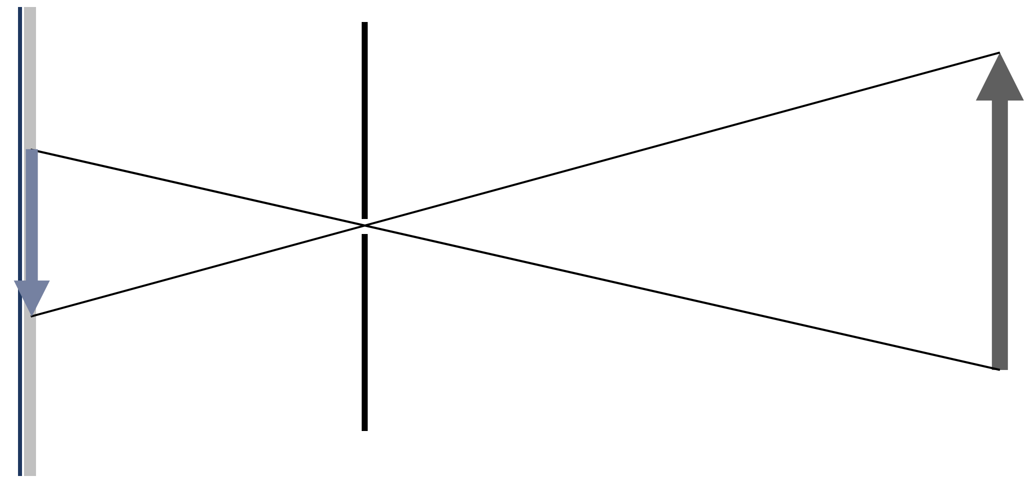
New Feature: We can perform perspective projections

- Very useful for 3D computer graphics
- Perspective projection (central projection) involves divisions that can be packaged into the homogeneous component:

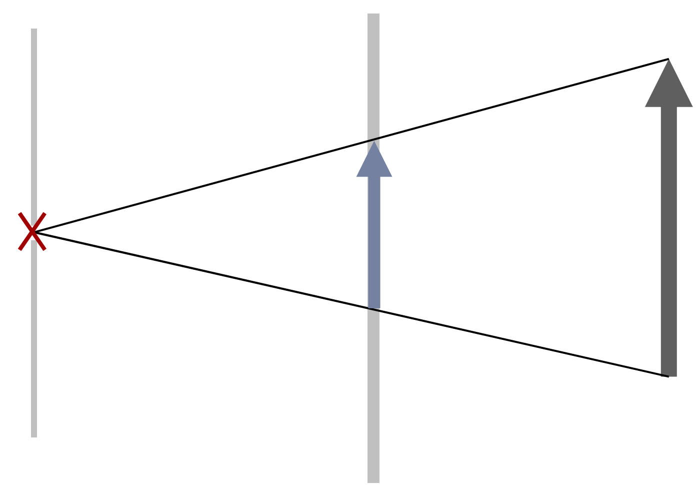


Perspective Projection

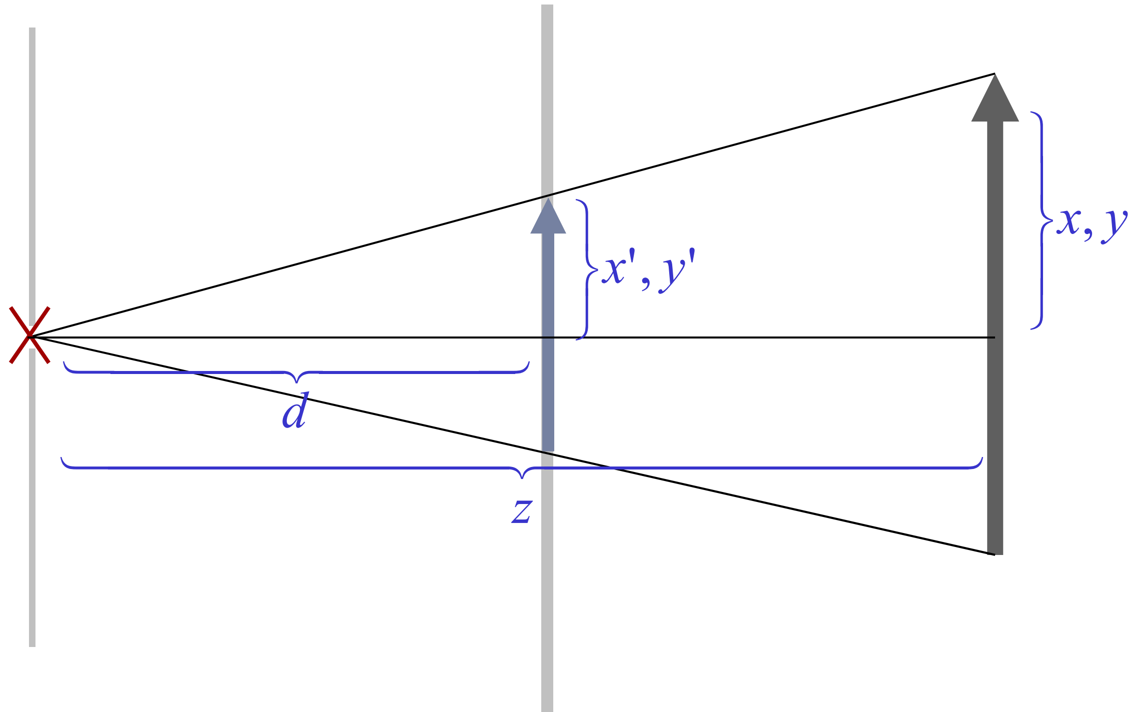
Physical camera:



Virtual camera:



Perspective Projection



center of projection image plane object

Perpective projection: $x' = d\frac{x}{z}, \quad y' = d\frac{y}{z}$

Homogenous Transformation

Projection can be expressed as linear transformation in homogenous coordinates:

- Trick: Put the denominator into the ω component.

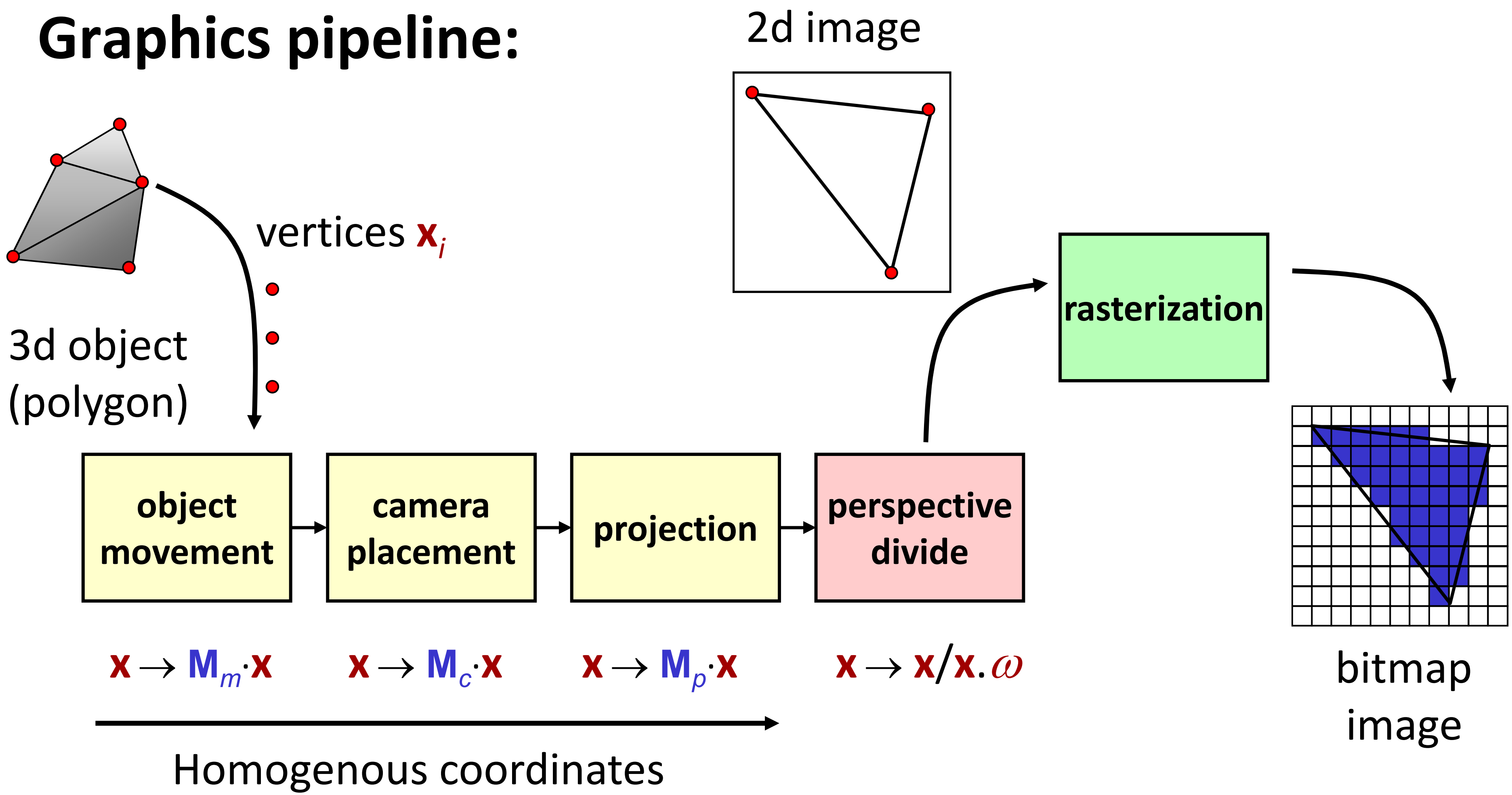
$$x' = d\frac{x}{z}, \quad y' = d\frac{y}{z}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ \omega' \end{pmatrix} = \begin{pmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ \omega \end{pmatrix}$$

- Camera placement: move scene in opposite direction

Graphics Pipeline

Graphics pipeline:



OpenGL Graphics Pipeline

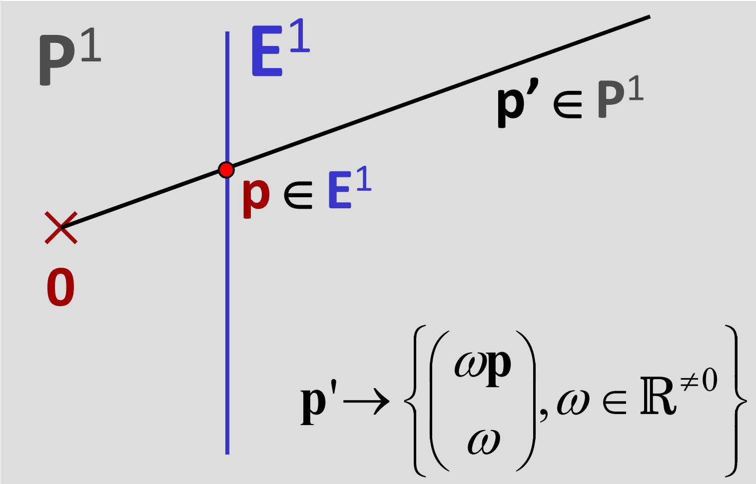
Example: OpenGL Pipeline

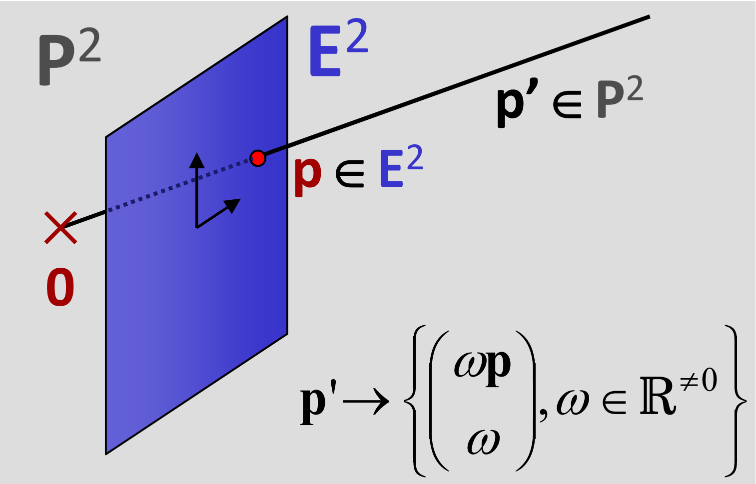
- Polygon primitives (triangles)
- Vertices specified by homogenous coordinates (4 floats)
- Transformation pipeline (basically) implemented by a 4x4 matrix transformation
- Hardware accelerated: Special purpose hardware that supports rapid 4D vector operations ("vertex shader")

Formal Definition

Projective Space $\mathbf{P}^d$:

- Obtained by embedding the Euclidian space $\mathbf{E}^d$ into a d+1 dimensional Euclidian space at $\omega = 1$.
  (The additional dimension is usually named ω.)
- Identify all points on lines through the origin to *represent* the same Euclidian points.





Properties

Properties:

- Points are represented by lines through the origin
- Consequence: scaling by common constant does not change the point ($euclidian(\lambda \mathbf{x}) = euclidian(\mathbf{x}),\ \lambda \neq 0$)
- We can scale the points arbitrarily
  (scaling does not matter)
- Means: Division by ω can be done at any time when multiple projective operations are performed on the projective points.
  (projective operation: map lines through the origin to lines through the origin)

Properties

Projective Maps:

- Represented by linear maps in the higher dimensional space
- Scale at any time: Linear maps are projective operations in the sense of mapping lines through the origin to lines through the origin again.
- Therefore:

$$\mathbf{y} = \mathbf{M}\mathbf{x} \mathrel{\hat{=}} \frac{\mathbf{M}\mathbf{x}}{\mathbf{y}.\omega} \mathrel{\hat{=}} \mathbf{M}\frac{\mathbf{x}}{\mathbf{x}.\omega} \quad (\text{for } \omega \neq 0)$$

Important: We have $\mathbf{x} \mathrel{\hat{=}} \alpha\,\mathbf{x}$, but in general: $\mathbf{x} + \mathbf{y} \mathrel{\hat{\neq}} \mathbf{x} + \alpha\mathbf{y}$

Directions

Problem: What if $\omega = 0$?

- Again – it's not a bug, it's a feature
- Projective points with $\omega = 0$ do not correspond to Euclidian points
- They represent *directions*, or *points at infinity*.
- This gives a natural distinction:
  - Euclidian points: $\omega \neq 0$ in homogenous coordinates.
  - Euclidian vectors: $\omega = 0$ in homogenous coordinates.
- The difference of points yields a vector.
- Vectors can be added to points, but not (not really) points to points.

Quadrics and Conics

Modeling Wish List

We want to model:

- Circles (Surfaces: Spheres)
- Ellipses (Surfaces: Ellipsoids)
- And segments of those
- Surfaces: Objects with circular cross section
  - Cylinders
  - Cones
  - Surfaces of revolution (lathing)

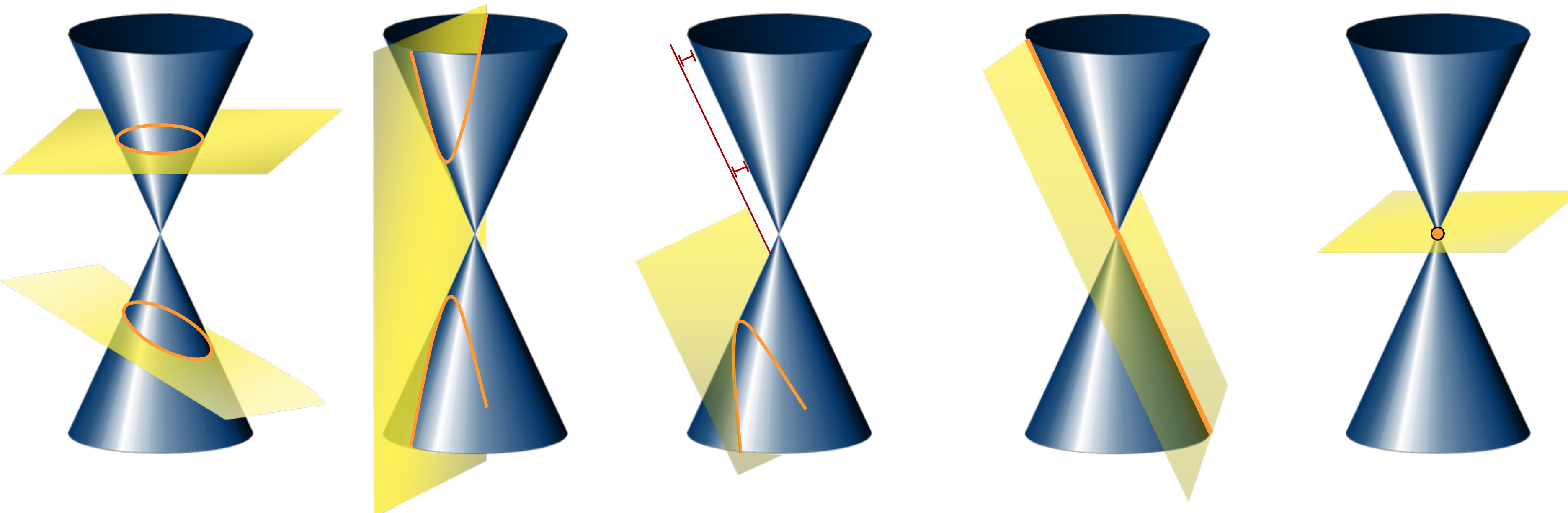
These objects cannot be represented exactly (only approximated) by piecewise polynomials

Conical Sections

Classic description of such objects:

- Conical sections (conics)
- Intersections of a cone and a plane
- Resulting objects:
  - Circles
  - Ellipses
  - Hyperbolas
  - Parabolas
  - Points
  - Lines

Conic Sections



Circle, Ellipse	Hyperbola	Parabola	Line (degenerate case)	Point (degenerate case)

Implicit Form

Implicit quadrics:

- Conic sections can be expressed as zero set of a quadratic function:

$$a x^2 + b xy + c y^2 + d x + e y + f = 0$$

$$\Leftrightarrow \mathbf{x}^{\mathrm{T}} \begin{pmatrix} a & 1/2 \cdot b \\ 1/2 \cdot b & c \end{pmatrix} \mathbf{x} + \begin{bmatrix} d & e \end{bmatrix} \mathbf{x} + f = 0$$

- Easy to see why:

Implicit eq. for a cone: $Ax^2 + By^2 = z^2$

Explicit eq. for a plane: $z = Dx + Ey + F$

Conical Section: $Ax^2 + By^2 = (Dx + Ey + F)^2$

Quadrics & Conics

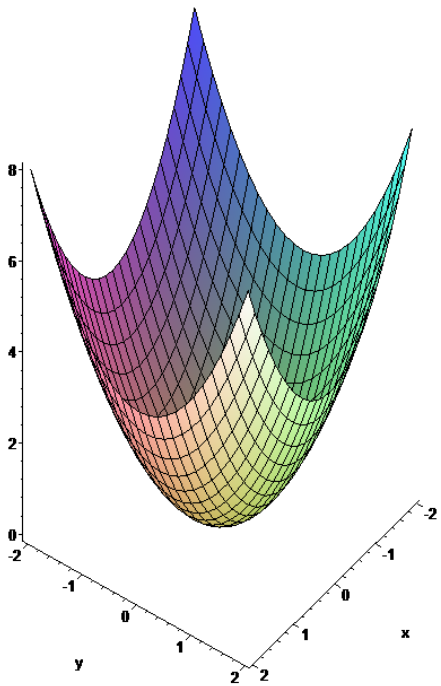
Quadrics:

- The general zero sets of quadratic functions (any dimension) are called *quadrics*:

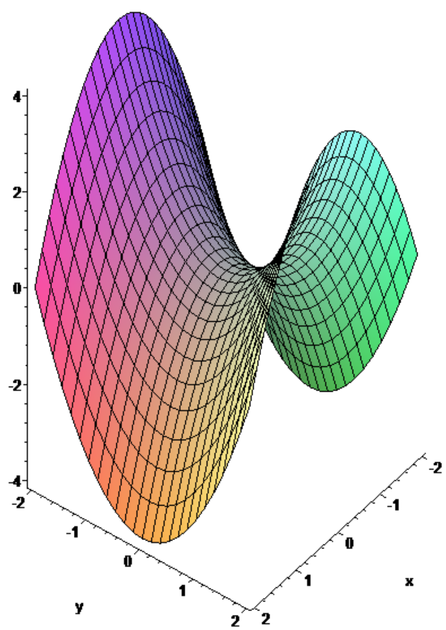
$$\left\{\mathbf{x} \in \mathbb{R}^{\mathrm{d}} \mid \mathbf{x}^{\mathrm{T}}\mathbf{M}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + \mathbf{c} = 0\right\}$$

- *Conics* are the special case for d = 2.

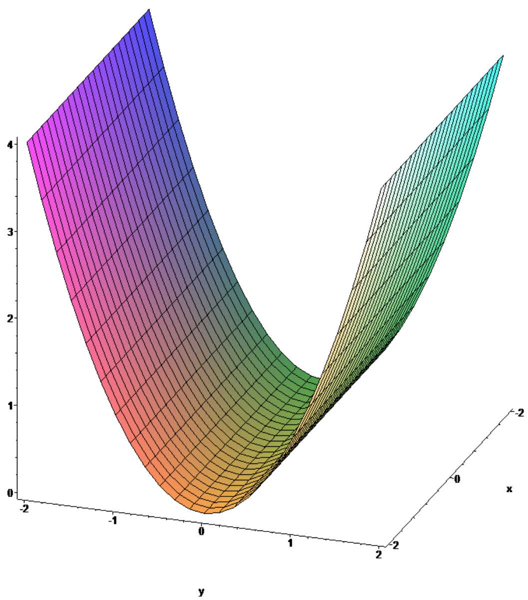
Shapes of Quadratic Polynomials



$\lambda_1 = 1, \lambda_2 = 1$

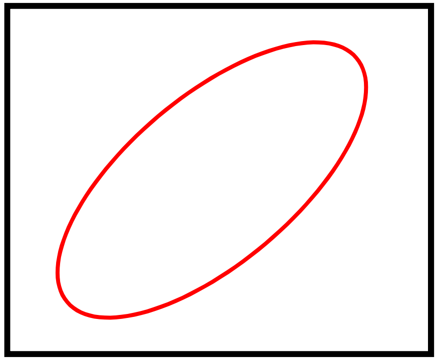
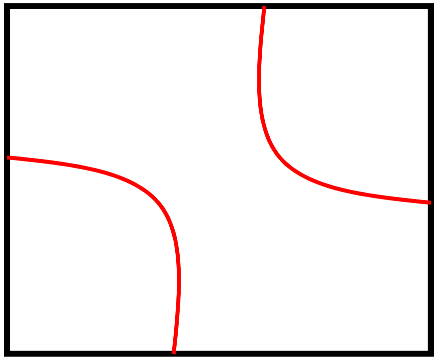
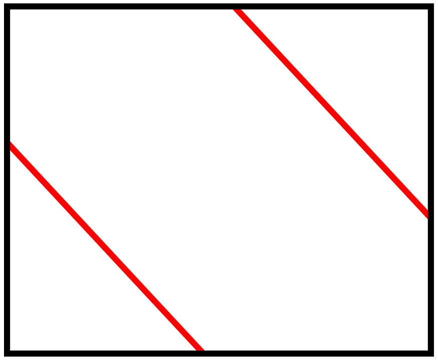


$\lambda_1 = 1, \lambda_2 = -1$



$\lambda_1 = 1, \lambda_2 = 0$

The Iso-Lines: Quadrics

elliptic	hyperbolic	degenerate case
		
$\lambda_1 > 0, \lambda_2 > 0$	$\lambda_1 < 0, \lambda_2 > 0$	$\lambda_1 = 0, \lambda_2 \neq 0$

Characterization

Determining the type of Conic from the implicit form:

- Implicit function: quadratic polynomial

$$a x^2 + b xy + c y^2 + d x + e y + f = 0$$

$$\Leftrightarrow \mathbf{x}^{\mathrm{T}} \underbrace{\begin{pmatrix} a & 1/2 \cdot b \\ 1/2 \cdot b & c \end{pmatrix}}_{\mathbf{M}} \mathbf{x} + \begin{bmatrix} d & e \end{bmatrix} \mathbf{x} + f = 0$$

- Eigenvalues of **M**:

$$\lambda_{1|2} = \frac{a+c}{2} \pm \frac{1}{2} \sqrt{(a-c)^2 + b^2}$$

Cases

implicit function:

$$a x^2 + b xy + c y^2 + d x + e y + f = 0$$

We obtain the following cases:

- Ellipse: $b^2 < 4ac$
  - Circle: $b = 0, a = c$
  - Otherwise: general ellipse
- Hyperbola: $b^2 > 4ac$
- Parabola: $b^2 = 4ac$ (border case)

Cases

Explanation:

implicit function:

$$a x^2 + b xy + c y^2 + d x + e y + f = 0$$

$$
\begin{aligned}
b^2 = 4ac \;\Rightarrow\; \lambda_{1|2} &= \frac{a+c}{2} \pm \frac{1}{2}\sqrt{(a-c)^2 + 4ac} \\
&= \frac{a+c}{2} \pm \sqrt{a^2 - 2ac + c^2 + 4ac} \\
&= \frac{a+c}{2} \pm \frac{1}{2}\sqrt{a^2 + 2ac + c^2} \\
&= \frac{a+c}{2} \pm \frac{1}{2}\sqrt{(a+c)^2} \\
&= \frac{a+c}{2} \pm \frac{a+c}{2} \\
&= \{0, a+c\}
\end{aligned}
$$

Parametrization

We want to represent conics with parametric curves:

- How can we represent (pieces) of conics as parametric curves?
- How can we generalize our framework of piecewise polynomial curves to include conical sections?

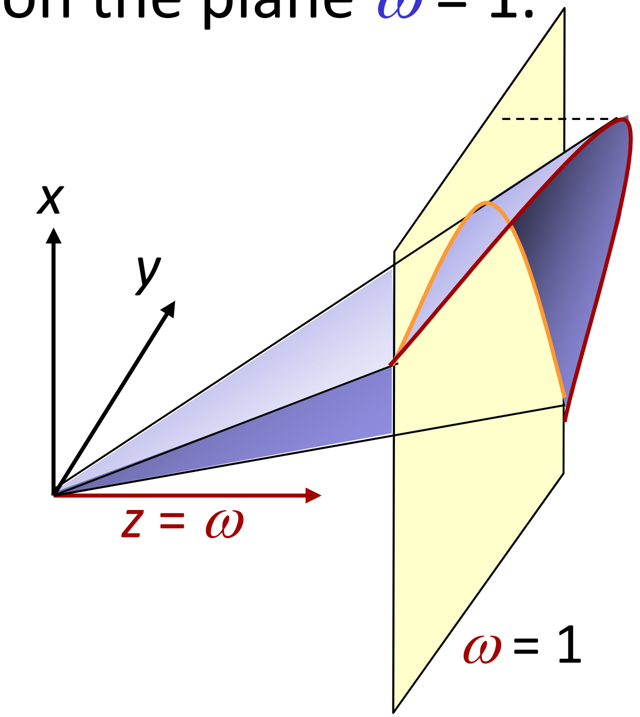
Projections of Parabolas:

- We will look at a certain class of parametric functions – projections of parabolas.
- This class turns out to be general enough,
- and can be expressed easily with the tools we know.

Projections of Parabolas

Definition: Projection of a Parabola

- We start with a quadratic space curve.
- Interpret the z-coordinate as homogenous component ω.
- Project the curve on the plane $\omega = 1$.



Projected Parabola

Formal Definition:

- Quadratic polynomial curve in three space
- Project by dividing by third coordinate

$$\mathbf{f}^{(hom)}(t) = \mathbf{p}_0 + t\,\mathbf{p}_1 + t^2\,\mathbf{p}_2 = \begin{pmatrix} \mathbf{p}_0.x \\ \mathbf{p}_0.y \\ \mathbf{p}_0.\omega \end{pmatrix} + t \begin{pmatrix} \mathbf{p}_1.x \\ \mathbf{p}_1.y \\ \mathbf{p}_1.\omega \end{pmatrix} + t^2 \begin{pmatrix} \mathbf{p}_2.x \\ \mathbf{p}_2.y \\ \mathbf{p}_2.\omega \end{pmatrix}$$

$$\mathbf{f}^{(eucl)}(t) = \frac{\begin{pmatrix} \mathbf{p}_0.x \\ \mathbf{p}_0.y \end{pmatrix} + t \begin{pmatrix} \mathbf{p}_1.x \\ \mathbf{p}_1.y \end{pmatrix} + t^2 \begin{pmatrix} \mathbf{p}_2.x \\ \mathbf{p}_2.y \end{pmatrix}}{\mathbf{p}_0.\omega + t\,\mathbf{p}_1.\omega + t^2\,\mathbf{p}_2.\omega}$$

Bernstein Basis

Alternatively: Represent in Bernstein basis

- Rational quadratic Bezier curves:

$$\mathbf{f}^{(hom)}(t) = B_0^{(2)}(t)\mathbf{p}_0 + B_1^{(2)}(t)\mathbf{p}_1 + B_2^{(2)}(t)\mathbf{p}_2$$

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(t)\begin{pmatrix}\mathbf{p}_0.x \\ \mathbf{p}_0.y\end{pmatrix} + B_1^{(2)}(t)\begin{pmatrix}\mathbf{p}_1.x \\ \mathbf{p}_1.y\end{pmatrix} + B_2^{(2)}(t)\begin{pmatrix}\mathbf{p}_2.x \\ \mathbf{p}_2.y\end{pmatrix}}{B_0^{(2)}(t)\mathbf{p}_0.\omega + B_1^{(2)}(t)\mathbf{p}_1.\omega + B_2^{(2)}(t)\mathbf{p}_2.\omega}$$

Properties

Projective invariance:

- Quadratic Bezier curves are invariant under projective maps
- The following operations yield the same result
  - Applying a projective map to the control points, then evaluate the curve
  - Applying the same projective map to the curve
- Proof:
  - 3D curve is invariant under linear maps
  - Scaling does not matter for projections (divide by ω before or after applying a projection matrix does not matter)

Parametrizing Conics

Conics can be parameterized using projected parabolas:

- We show that we can represent (piecewise):
  - Points and lines (obvious ✓)
  - A unit parabola
  - A unit circle
  - A unit hyperbola
- General cases (ellipses etc.) can be obtained by affine mappings of the control points (which leads to affine maps of the curve)

Parametrizing Parabolas

Parabolas as rational parametric curves:

$$\mathbf{f}^{(eucl)}(t) = \frac{\begin{pmatrix}0\\0\end{pmatrix} + t\begin{pmatrix}1\\0\end{pmatrix} + t^2\begin{pmatrix}0\\1\end{pmatrix}}{1 + 0t + 0t^2} \qquad \begin{pmatrix} x(t) = t \\ y(t) = t^2 \end{pmatrix} \quad \checkmark$$

(pretty obvious as well)

Circle

Let's try to find a rational parametrization of a (piece of a) unit circle:

$$\mathbf{f}^{(eucl)}(\varphi) = \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix}$$

$$\cos\varphi = \frac{1-\tan^2\frac{\varphi}{2}}{1+\tan^2\frac{\varphi}{2}}, \quad \sin\varphi = \frac{2\tan\frac{\varphi}{2}}{1+\tan^2\frac{\varphi}{2}}$$ (tangent half-angle formula)

$$t := \tan\frac{\varphi}{2} \Rightarrow \mathbf{f}^{(eucl)}(\varphi) = \begin{pmatrix} \frac{1-t^2}{1+t^2} \\ \frac{2t}{1+t^2} \end{pmatrix}$$

Circle

Let's try to find a rational parametrization of a (piece of a) unit circle:

$$\mathbf{f}^{(eucl)}(\varphi) = \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix} = \begin{pmatrix} \dfrac{1-t^2}{1+t^2} \\ \dfrac{2t}{1+t^2} \end{pmatrix} \text{ with } t := \tan\frac{\varphi}{2}$$

$$\Rightarrow \mathbf{f}^{(hom)}(t) = \begin{pmatrix} 1-t^2 \\ 2t \\ 1+t^2 \end{pmatrix}$$

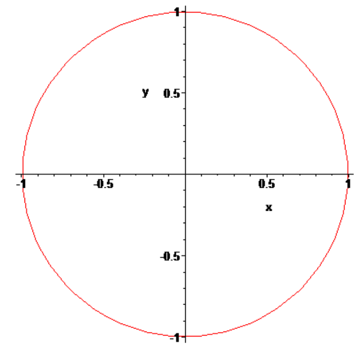
parametrization for $\varphi \in$ (-90°..90°)

$\Rightarrow$ we need at least three segments to parametrize a full circle

Hyperbolas

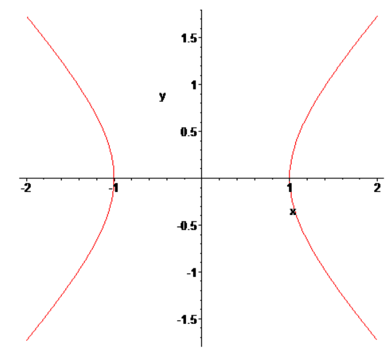
Unit Circle: $x^2 + y^2 = 1$

$$\Rightarrow \; x(t) = \frac{1-t^2}{1+t^2}, y(t) = \frac{2t}{1+t^2} \quad (t \in \mathbb{R})$$



Unit Hyperbola: $x^2 - y^2 = 1$

$$\Rightarrow \; x(t) = \frac{1+t^2}{1-t^2}, y(t) = \frac{2t}{1-t^2} \quad (t \in [0..1)\,)$$



Rational Bezier Curves

Rational Bezier Curves

Rational Bezier curves in $\mathbb{R}^n$ of degree d:

- Form a Bezier curve of degree d in n+1-dimensional space
- Interpret last coordinate as homogenous component
- Euclidian coordinates are obtained by projection.

$$\mathbf{f}^{(hom)}(t) = \sum_{i=0}^{n} B_i^{(d)}(t)\mathbf{p}_i, \quad \mathbf{p}_i \in \mathbb{R}^{n+1}$$

$$\mathbf{f}^{(eucl)}(t) = \frac{\sum_{i=0}^{n} B_i^{(d)}(t) \begin{pmatrix} p_i^{(1)} \\ \vdots \\ p_i^{(n)} \end{pmatrix}}{\sum_{i=0}^{n} B_i^{(d)}(t) p_i^{(n+1)}}$$

More Convenient Notation

The curve can be written in "weighted points" form:

$$\mathbf{f}^{(eucl)}(t) = \frac{\sum_{i=0}^{n} B_i^{(d)}(t)\,\omega_i \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}}{\sum_{i=0}^{n} B_i^{(d)}(t)\,\omega_i}$$

Interpretation:

- Points are weighted by weights ω_i
- Normalized by interpolated weights in the denominator
- Larger weights $\rightarrow$ more influence of that point

Properties

What about affine invariance, convex hull prop.?

$$\mathbf{f}^{(eucl)}(t) = \frac{\sum_{i=0}^{n} B_i^{(d)}(t)\omega_i \mathbf{p}_i}{\sum_{i=0}^{n} B_i^{(d)}(t)\omega_i} = \sum_{i=0}^{n} q_i(t)\mathbf{p}_i \quad \text{with} \quad \sum_{i=0}^{n} q_i(t) = 1$$

Consequence:

- Affine invariance still holds
- For strictly positive weights:
  - Convex hull property still holds
  - This is not a big restriction (potential singularities otherwise)
- Projective invariance (projective maps, hom. coord's)

Quadratic Bezier Curves

Quadratic curves:

- Necessary and sufficient to represent conics
- Therefore, we will examine them closer...

Quadratic rational Bezier curve:

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(t)\omega_0\mathbf{p}_0 + B_1^{(2)}(t)\omega_1\mathbf{p}_1 + B_2^{(2)}(t)\omega_2\mathbf{p}_2}{B_0^{(2)}(t)\omega_0 + B_1^{(2)}(t)\omega_1 + B_2^{(2)}(t)\omega_2}, \quad \mathbf{p}_i \in \mathbb{R}^n, \omega_i \in \mathbb{R}$$

Standard Form

How many degrees of freedom are in the weights?

- Quadratic rational Bezier curve:

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(t)\omega_0\mathbf{p}_0 + B_1^{(2)}(t)\omega_1\mathbf{p}_1 + B_2^{(2)}(t)\omega_2\mathbf{p}_2}{B_0^{(2)}(t)\omega_0 + B_1^{(2)}(t)\omega_1 + B_2^{(2)}(t)\omega_2}$$

- If one of the weights is $\neq 0$ (which must be the case), we can divide numerator and denominator by this weight and thus remove one degree of freedom.
- If we are only interested in the *shape of the curve*, we can remove one more degree of freedom by a *reparametrization*...

Standard Form

How many degrees of freedom are in the weights?

- Concerning the shape of the curve, the parametrization does not matter.
- We have:

$$\mathbf{f}^{(eucl)}(t) = \frac{(1-t)^2\,\omega_0\mathbf{p}_0 + 2t(1-t)\,\omega_1\mathbf{p}_1 + t^2\,\omega_2\mathbf{p}_2}{(1-t)^2\,\omega_0 + 2t(1-t)\,\omega_1 + t^2\,\omega_2}$$

- We set: (with α to be determined later)

$$t \leftarrow \frac{\tilde{t}}{\alpha(1-\tilde{t}) + \tilde{t}}, \quad i.e., (1-t) \leftarrow \frac{\alpha\left(1-\tilde{t}\right)}{\alpha(1-\tilde{t}) + \tilde{t}}$$

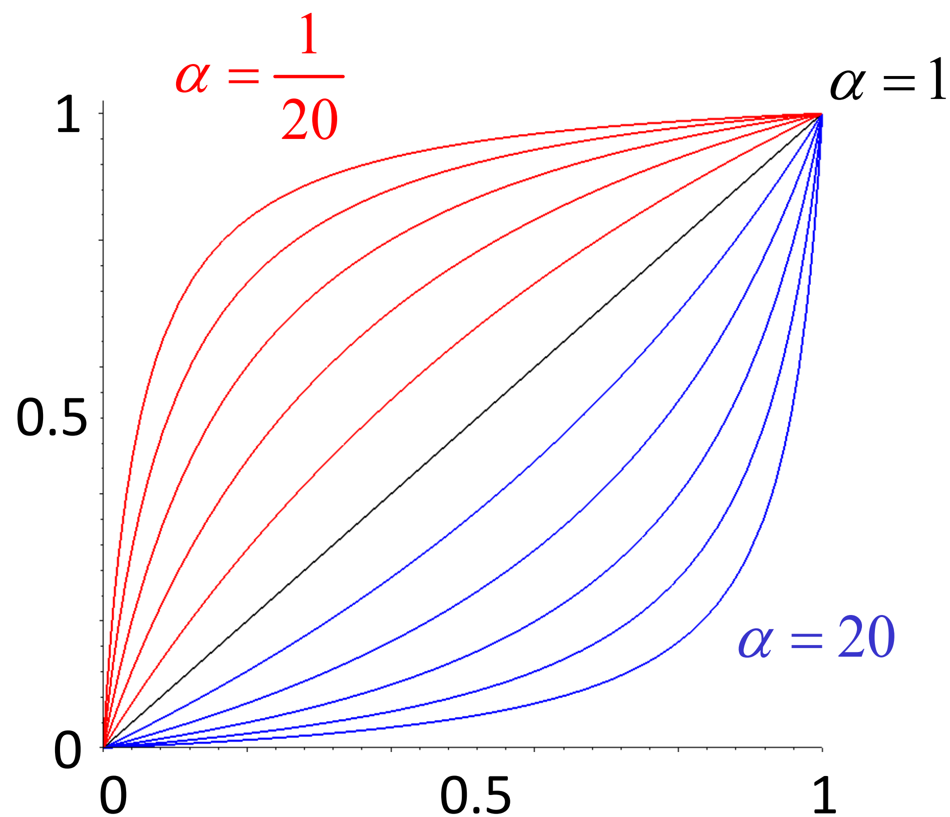
Remark: Why this reparametrization?

Reparametrization:

$$t \leftarrow \frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}$$

Properties:

- $0 \rightarrow 0$,
  $1 \rightarrow 1$,
  monotonic in between
- Shape determined by parameter α.



Standard Form

$$t \leftarrow \frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}, \; i.e., (1-t) \leftarrow \frac{\alpha\left(1-\tilde{t}\right)}{\alpha(1-\tilde{t})+\tilde{t}}$$

Standard Form

$$t \leftarrow \frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}},\ i.e., (1-t) \leftarrow \frac{\alpha(1-\tilde{t})}{\alpha(1-\tilde{t})+\tilde{t}}$$

$$\mathbf{f}^{(eucl)}(t) = \frac{\left(\frac{\alpha(1-\tilde{t})}{\alpha(1-\tilde{t})+\tilde{t}}\right)^2 \omega_0\mathbf{p}_0 + 2\left(\frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}\right)\frac{\alpha(1-\tilde{t})}{\alpha(1-\tilde{t})+\tilde{t}}\omega_1\mathbf{p}_1 + \left(\frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}\right)^2 \omega_2\mathbf{p}_2}{\left(\frac{\alpha(1-\tilde{t})}{\alpha(1-\tilde{t})+\tilde{t}}\right)^2 \omega_0 + 2\left(\frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}\right)\frac{\alpha(1-\tilde{t})}{\alpha(1-\tilde{t})+\tilde{t}}\omega_1 + \left(\frac{\tilde{t}}{\alpha(1-\tilde{t})+\tilde{t}}\right)^2 \omega_2}$$

$$= \frac{\alpha^2(1-\tilde{t})^2\omega_0\mathbf{p}_0 + 2\alpha\tilde{t}(1-\tilde{t})\omega_1\mathbf{p}_1 + \tilde{t}^2\omega_2\mathbf{p}_2}{\alpha^2(1-\tilde{t})^2\omega_0 + 2\alpha\tilde{t}(1-\tilde{t})\omega_1 + \tilde{t}^2\omega_2}$$

$$= \frac{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0\mathbf{p}_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1\mathbf{p}_1 + B_2^{(2)}(\tilde{t})\omega_2\mathbf{p}_2}{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1 + B_2^{(2)}(\tilde{t})\omega_2}$$

Standard Form

$$\mathbf{f}^{(eucl)}(t) = \frac{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0 \mathbf{p}_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1 \mathbf{p}_1 + B_2^{(2)}(\tilde{t})\omega_2 \mathbf{p}_2}{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1 + B_2^{(2)}(\tilde{t})\omega_2}$$

$$\text{let } \alpha = \sqrt{\frac{\omega_2}{\omega_0}} \quad (\text{assume } 0 \leq \frac{\omega_2}{\omega_0} < \infty)$$

Standard Form

$$\mathbf{f}^{(eucl)}(t) = \frac{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0\mathbf{p}_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1\mathbf{p}_1 + B_2^{(2)}(\tilde{t})\omega_2\mathbf{p}_2}{\alpha^2 B_0^{(2)}(\tilde{t})\omega_0 + \alpha B_1^{(2)}(\tilde{t})\omega_1 + B_2^{(2)}(\tilde{t})\omega_2}$$

$$\text{let } \alpha = \sqrt{\frac{\omega_2}{\omega_0}} \quad \left(\text{assume } 0 \le \frac{\omega_2}{\omega_0} < \infty\right)$$

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}^2 \omega_0\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1\mathbf{p}_1 + \omega_2 B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}^2 \omega_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1 + \omega_2 B_2^{(2)}(\tilde{t})}$$

$$= \frac{B_0^{(2)}(\tilde{t})\omega_2\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1\mathbf{p}_1 + \omega_2 B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t})\omega_2 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1 + \omega_2 B_2^{(2)}(\tilde{t})}$$

Standard Form

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(\tilde{t})\omega_2\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1\mathbf{p}_1 + \omega_2 B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t})\omega_2 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1 + \omega_2 B_2^{(2)}(\tilde{t})}$$

Standard Form

$$\mathbf{f}^{(eucl)}(t) = \frac{B_0^{(2)}(\tilde{t})\omega_2\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1\mathbf{p}_1 + \omega_2 B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t})\omega_2 + B_1^{(2)}(\tilde{t})\sqrt{\frac{\omega_2}{\omega_0}}\omega_1 + \omega_2 B_2^{(2)}(\tilde{t})}$$

$$= \frac{B_0^{(2)}(\tilde{t})\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\sqrt{\frac{1}{\omega_0\omega_2}}\omega_1\mathbf{p}_1 + B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t}) + B_1^{(2)}(\tilde{t})\sqrt{\frac{1}{\omega_0\omega_2}}\omega_1 + B_2^{(2)}(\tilde{t})}$$

$$= \frac{B_0^{(2)}(\tilde{t})\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\omega\mathbf{p}_1 + B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t}) + B_1^{(2)}(\tilde{t})\omega + B_2^{(2)}(\tilde{t})} \qquad \text{with: } \omega := \sqrt{\frac{1}{\omega_0\omega_2}}\omega_1$$

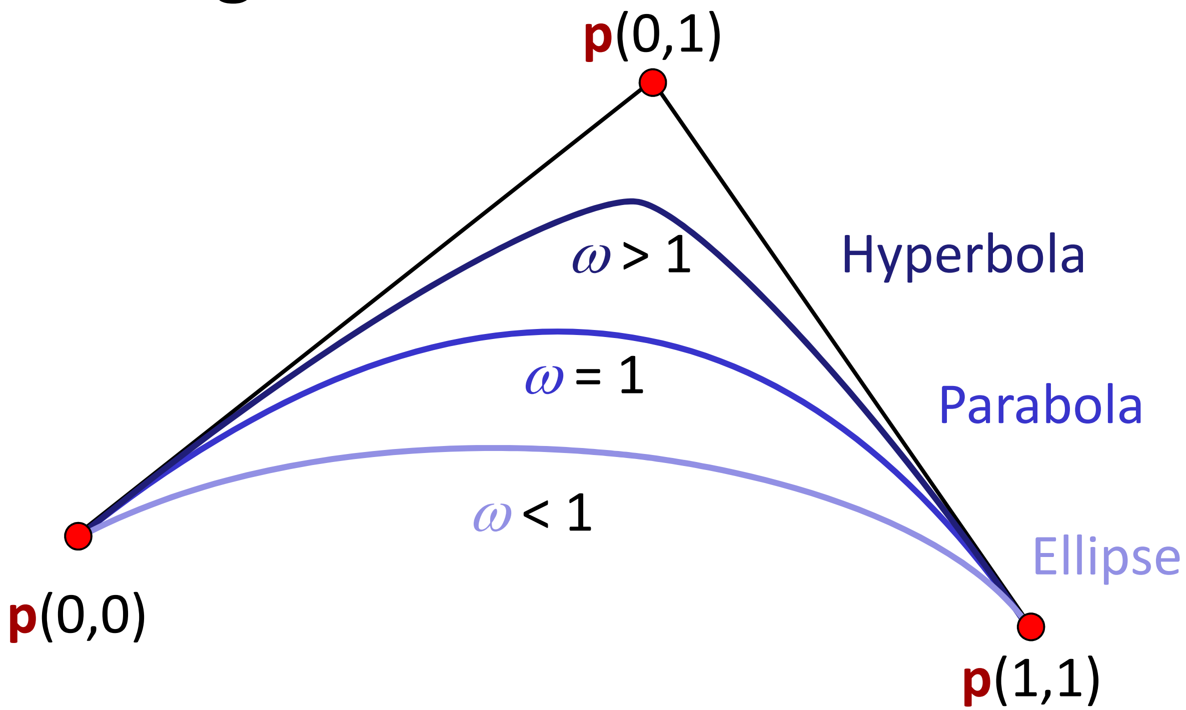
Standard Form

Consequence:

- It is sufficient to specify the weight of the inner point
- We can w.l.o.g. set $\omega_0 = \omega_2 = 1$, $\omega_1 = \omega$
- This form of a quadratic Bezier curve is called the *standard form*.
- Choices:
  - $\omega < 1$: ellipse segment
  - $\omega = 1$: parabola segment (non-rational curve)
  - $\omega > 1$: hyperbola segment

Illustration

Changing the weight:



Conversion to Implicit Form

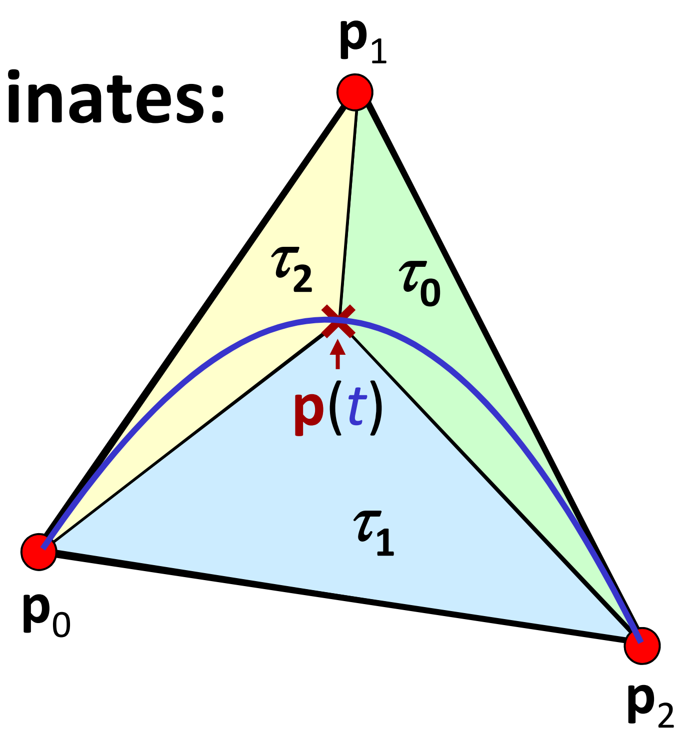
Convert parametric to implicit form:

- In order to show the shape conditions
- For distance computations / inside-outside tests

Express curve in barycentric coordinates:

- Curve can be expressed in barycentric coordinates (linear transform):

$$\mathbf{f}(t) = \tau_0(t)\mathbf{p}_0 + \tau_1(t)\mathbf{p}_1 + \tau_2(t)\mathbf{p}_2$$



Conversion to Implicit Form

Comparison of coefficients yields:

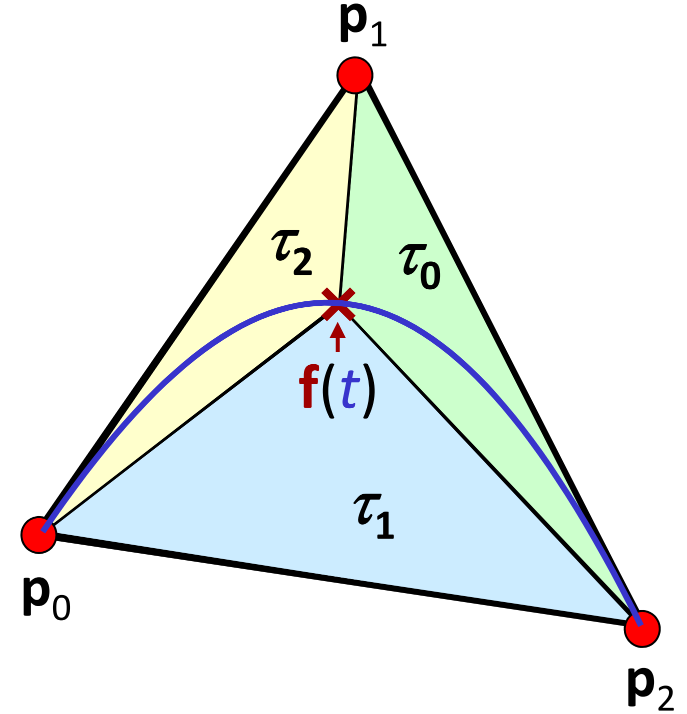
$$\mathbf{f}(t) = \tau_0(t)\mathbf{p}_0 + \tau_1(t)\mathbf{p}_1 + \tau_2(t)\mathbf{p}_2$$

$$\mathbf{f}^{(eucl)}(t) = \frac{(1-t)^2\omega_0\mathbf{p}_0 + 2t(1-t)\omega_1\mathbf{p}_1 + t^2\omega_2\mathbf{p}_2}{(1-t)^2\omega_0 + 2t(1-t)\omega_1 + t^2\omega_2}$$

$$\tau_0(t) = \frac{\omega_0 B_0^{(2)}(t)}{\underbrace{\sum_{i=0}^{2}\omega_i B_i^{(2)}(t)}_{=:D}} = \frac{\omega_0(1-t)^2}{D(t)}$$

$$\tau_1(t) = \frac{\omega_1 B_1^{(2)}(t)}{\sum_{i=0}^{2}\omega_i B_i^{(2)}(t)} = \frac{2\omega_1 t(1-t)}{D(t)}$$

$$\tau_2(t) = \frac{\omega_0 B_0^{(2)}(t)}{\sum_{i=0}^{2}\omega_i B_i^{(2)}(t)} = \frac{\omega_0 t^2}{D(t)}$$



Conversion to Implicit Form

Solving for t, (1-t):

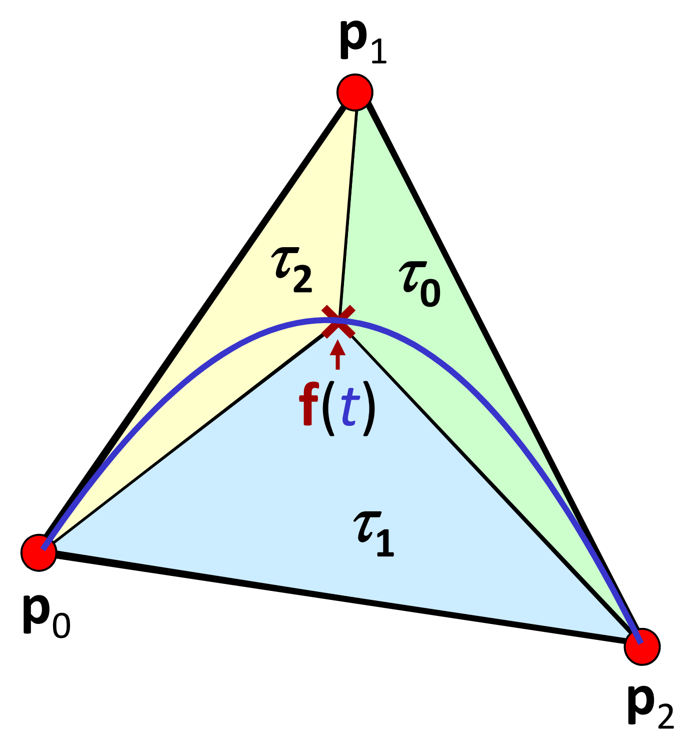
$$\tau_0(t) = \frac{\omega_0(1-t)^2}{D(t)} \Rightarrow (1-t) = \sqrt{\frac{\tau_0(t)D(t)}{\omega_0}}$$

$$\tau_1(t) = \frac{2\omega_1 t(1-t)}{D(t)}$$

$$\tau_2(t) = \frac{\omega_2 t^2}{D(t)} \Rightarrow t = \sqrt{\frac{\tau_2(t)D(t)}{\omega_2}}$$

$$\tau_1(t) = \frac{2\omega_1\sqrt{\frac{\tau_2(t)D(t)}{\omega_2}\frac{\tau_0(t)D(t)}{\omega_0}}}{D(t)} = 2\omega_1\sqrt{\frac{\tau_2(t)\tau_0(t)}{\omega_0\omega_2}}$$

$$\Rightarrow \frac{\tau_1(t)^2}{\tau_2(t)\tau_0(t)} = 4\frac{\omega_1^2}{\omega_0\omega_2}$$



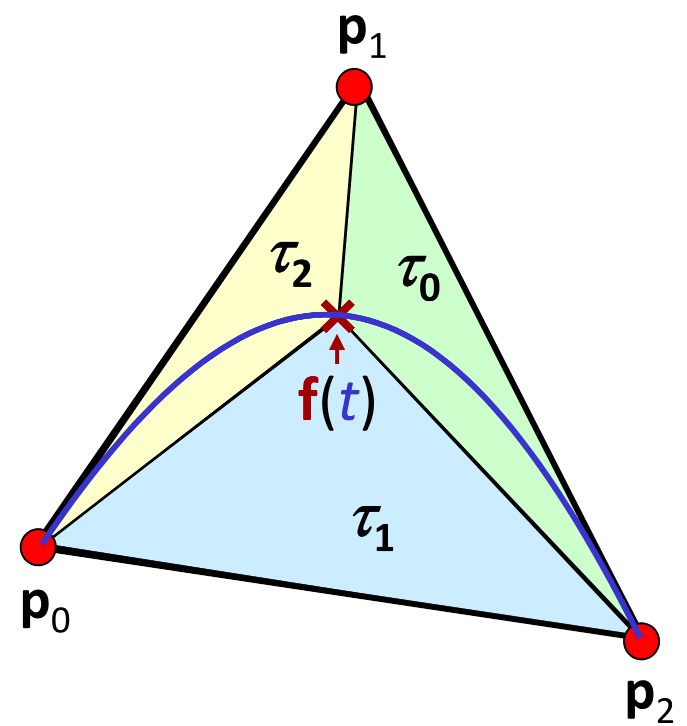
Conversion to Implicit Form

Some more algebra...:

$$\frac{\tau_1(t)^2}{\tau_2(t)\tau_0(t)} = 4\frac{\omega_1^{\,2}}{\omega_0\omega_2}$$

Using $\tau_2(t) = \left(1-\tau_0(t)-\tau_1(t)\right)$ we get:

$$\begin{aligned}
[\omega_0\omega_2]\tau_1(t)^2 &= \left[4\omega_1^{\,2}\right]\tau_2(t)\tau_0(t) \\
&= \left[4\omega_1^{\,2}\right]\tau_0(t)\left(1-\tau_0(t)-\tau_1(t)\right) \\
&= \left[4\omega_1^{\,2}\right]\left(\tau_0(t)-\tau_0(t)^2-\tau_1(t)\tau_0(t)\right)
\end{aligned}$$



$$\Rightarrow \boxed{[\omega_0\omega_2]}\tau_1(t)^2 + \boxed{\left[4\omega_1^{\,2}\right]}\tau_1(t)\tau_0(t) + \boxed{\left[4\omega_1^{\,2}\right]}\tau_0(t)^2 - \boxed{\left[4\omega_1^{\,2}\right]}\tau_0(t) = 0$$

$$\boxed{a}\, x^2 \quad + \quad \boxed{b}\, xy \quad + \quad \boxed{c}\, y^2 \quad + \quad \boxed{e}\, x \quad + 0y \; + \; 0 \; = 0$$

Classification

Eigenvalue argument led to:

- Parabola requires $b^2 = 4ac$ in $a\,x^2 + b\,xy + c\,y^2 + d\,x + e\,y + f = 0$
- In our case:

$$[\omega_0\omega_2]\tau_1(t)^2 + [4\omega_1^{\;2}]\tau_1(t)\tau_0(t) + [4\omega_1^{\;2}]\tau_0(t)^2 - [4\omega_1^{\;2}]\tau_0(t) = 0$$

i.e.:

$$4[\omega_0\omega_2][4\omega_1^{\;2}] = [4\omega_1^{\;2}]^2$$
$$\Leftrightarrow 16\omega_0\omega_2\omega_1^{\;2} = 16\omega_1^{\;4}$$
$$\Leftrightarrow \omega_0\omega_2 = \omega_1^{\;2}$$

Standard form: $\omega_0 = \omega_2 = 1$

$$\Rightarrow \omega_1 = 1$$

Classification

Similarly, it follows that:

$\omega_1 < 1 \rightarrow$ Ellipse

$\omega_1 = 1 \rightarrow$ Parabola $\quad (\omega_0 = \omega_2 = 1)$

$\omega_1 > 1 \rightarrow$ Hyperbola

Circle in Bezier Form

Quadratic rational polynomial:

$$\mathbf{f}(t) = \frac{1}{1+t^2}\begin{pmatrix} 1-t^2 \\ 2t \end{pmatrix}, \mathrm{t} = \tan\frac{\varphi}{2}, \varphi \in (-90°..90°)$$

Conversion to Bezier basis:

$$B_0^{(2)} = (1-t)^2 = 1 - 2t + t^2 \mathrel{\hat{=}} \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}^{\mathrm{T}}$$

$$B_1^{(2)} = 2t(1-t) = 2t - 2t^2 \mathrel{\hat{=}} \begin{bmatrix} 0 & 2 & -2 \end{bmatrix}^{\mathrm{T}}$$

$$B_2^{(2)} = t^2 \mathrel{\hat{=}} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}}$$

$$1-t^2 \mathrel{\hat{=}} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}^{\mathrm{T}}$$

$$2t \mathrel{\hat{=}} \begin{bmatrix} 0 & 2 & 0 \end{bmatrix}^{\mathrm{T}}$$

$$1+t^2 \mathrel{\hat{=}} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}^{\mathrm{T}}$$

Circle in Bezier Form

Conversion to Bezier basis:

$$B_0^{(2)} = (1-t)^2 = 1-2t+t^2 \mathrel{\hat{=}} \begin{bmatrix}1 & -2 & 1\end{bmatrix}^{\mathrm{T}}$$
$$B_1^{(2)} = 2t(1-t) = 2t-2t^2 \mathrel{\hat{=}} \begin{bmatrix}0 & 2 & -2\end{bmatrix}^{\mathrm{T}}$$
$$B_2^{(2)} = t^2 \mathrel{\hat{=}} \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}}$$

$$1-t^2 \mathrel{\hat{=}} \begin{bmatrix}1 & 0 & -1\end{bmatrix}^{\mathrm{T}}$$
$$2t \mathrel{\hat{=}} \begin{bmatrix}0 & 2 & 0\end{bmatrix}^{\mathrm{T}}$$
$$1+t^2 \mathrel{\hat{=}} \begin{bmatrix}1 & 0 & 1\end{bmatrix}^{\mathrm{T}}$$

Comparison yields:

$$\begin{aligned} 1-t^2 &= B_0^{(2)} + B_1^{(2)} \\ 2t &= B_1^{(2)} + 2B_2^{(2)} \\ 1+t^2 &= B_0^{(2)} + B_1^{(2)} + 2B_2^{(2)} \end{aligned}$$

$$\mathbf{f}^{(\mathrm{hom})}(t) = \begin{pmatrix}1\\0\\1\end{pmatrix} B_0^{(2)} + \begin{pmatrix}1\\1\\1\end{pmatrix} B_1^{(2)} + \begin{pmatrix}0\\2\\2\end{pmatrix} B_2^{(2)}$$

Circle in Bezier Form

Result:

$$\mathbf{f}(t) = \frac{B_0^{(2)}(t)\begin{pmatrix}1\\0\end{pmatrix} + B_1^{(2)}(t)\begin{pmatrix}1\\1\end{pmatrix} + 2B_2^{(2)}(t)\begin{pmatrix}0\\1\end{pmatrix}}{B_0^{(2)}(t) + B_1^{(2)}(t) + 2B_2^{(2)}(t)}$$

Parameters:

$$\mathrm{t} = \tan\frac{\varphi}{2} \Rightarrow \varphi = 2\arctan t$$

$$t \in [0,1] \rightarrow \varphi \in [0°..90°]$$

Circle in Bezier Form

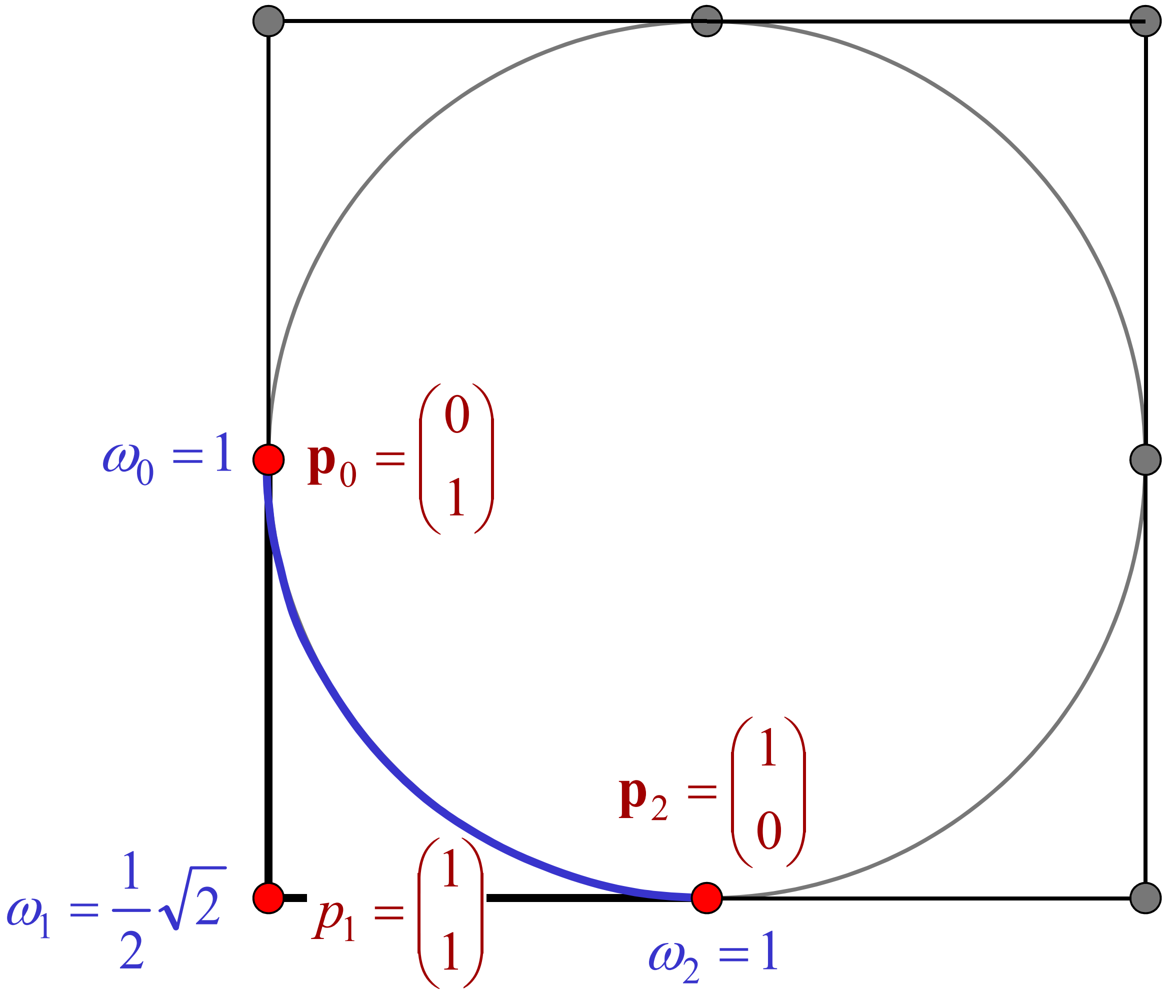
Standard Form:

$$\mathbf{f}(t) = \frac{B_0^{(2)}(\tilde{t})\mathbf{p}_0 + B_1^{(2)}(\tilde{t})\omega\mathbf{p}_1 + B_2^{(2)}(\tilde{t})\mathbf{p}_2}{B_0^{(2)}(\tilde{t}) + B_1^{(2)}(\tilde{t})\omega + B_2^{(2)}(\tilde{t})} \qquad \text{with: } \omega := \sqrt{\frac{1}{\omega_0\omega_2}}\omega_1$$

$$\mathbf{f}(t) = \frac{B_0^{(2)}\begin{pmatrix}1\\0\end{pmatrix} + \frac{1}{2}\sqrt{2}B_1^{(2)}\begin{pmatrix}1\\1\end{pmatrix} + B_2^{(2)}\begin{pmatrix}0\\1\end{pmatrix}}{B_0^{(2)} + \frac{1}{2}\sqrt{2}B_1^{(2)} + B_2^{(2)}}$$

Result: Circle in Bezier Form

Final Result:



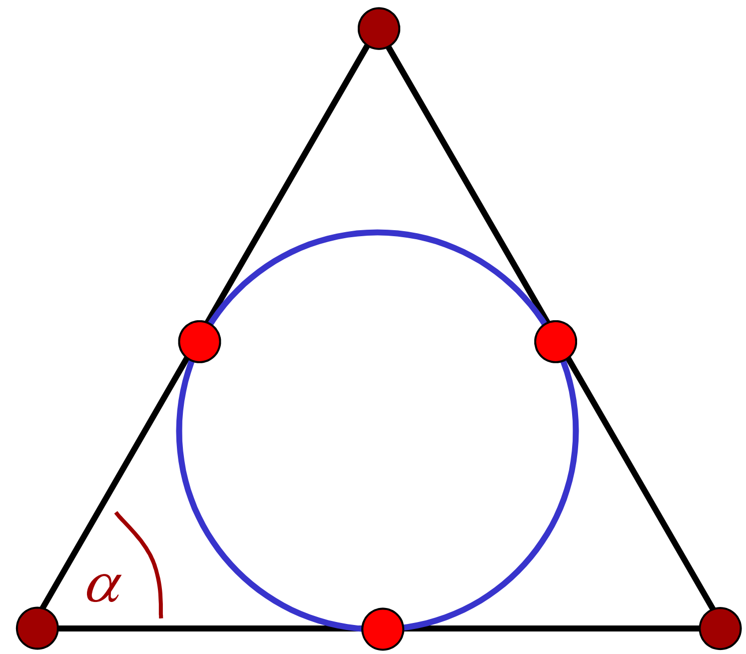
General Circle Segments

In general:

for $\omega_0 = \omega_2 = 1$:

$\omega_1 = \cos\alpha$

angle interval < 180°



α = 60°

$\rightarrow \omega_1$ = 0.5

Properties, Remarks

Continuity:

- The parametrization is only C^1, but G^∞
- No arc length parametrization possible
- *Even stronger:* No rational curve other than a straight line can have an arc-length parametrization.

Circles in in general degree Bezier splines:

- Simplest solution:
  - Form quadratic circle (segments)
  - Apply degree elevation to obtain the desired degree

Rational De Casteljau Algorithm

Evaluation with De Casteljau Algorithm

- Two Variants:
  - Compute numerator and denominator separately, then divide
  - Divide in each intermediate step (“rational de Casteljau”)
- Non-rational de Casteljau algorithm:

$$\mathbf{b}_i^{(r)}(t) = (1-t)\mathbf{b}_i^{(r-1)}(t) + t\mathbf{b}_{i+1}^{(r-1)}(t)$$

- Rational de Casteljau algorithm:

$$\mathbf{b}_i^{(r)}(t) = (1-t)\frac{\omega_i^{(r-1)}(t)}{\omega_i^{(r)}(t)}\mathbf{b}_i^{(r-1)}(t) + t\frac{\omega_{i+1}^{(r-1)}(t)}{\omega_i^{(r)}(t)}\mathbf{b}_{i+1}^{(r-1)}(t)$$

with

$$\omega_i^{(r)}(t) = (1-t)\omega_i^{(r-1)}(t) + t\omega_{i+1}^{(r-1)}(t)$$

Rational De Casteljau Algorithm

Advantages:

- More intuitive (repeated weighted linear interpolation of points and weights)
- Numerically more stable (only convex combinations for the standard case of positive weights, $t \in [0,1]$)

Weight Points

Alternative technique to specify weights:

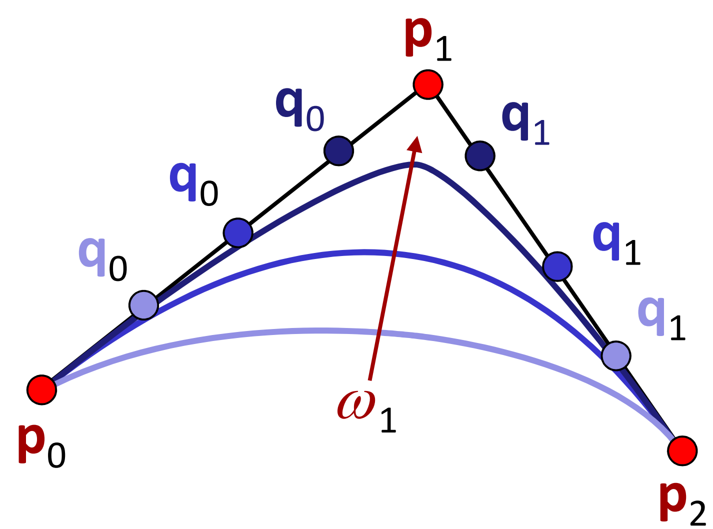
- Weight points
- User interface: More intuitive in interactive design

Weight Points:

$$\mathbf{q}_0 = \frac{\omega_0 \mathbf{p}_0 + \omega_1 \mathbf{p}_1}{\omega_0 + \omega_1}, \quad \mathbf{q}_1 = \frac{\omega_1 \mathbf{p}_1 + \omega_2 \mathbf{p}_2}{\omega_1 + \omega_2}$$

Standard Form:

$$\mathbf{q}_0 = \frac{\mathbf{p}_0 + \omega_1 \mathbf{p}_1}{1 + \omega_1}, \quad \mathbf{q}_1 = \frac{\mathbf{p}_1 + \omega_1 \mathbf{p}_2}{1 + \omega_1}$$



Derivatives

Computing derivatives of rational Bezier curves:

- Straightforward: Apply quotient rule
- A simpler expression can be derived using an algebraic trick:

$$\mathbf{f}(t) = \frac{\sum_{i=0}^{d} B_i^{(d)}(t)\omega_i \mathbf{p}_i}{\sum_{i=0}^{d} B_i^{(d)}(t)\omega_i} =: \frac{\mathbf{p}(t)}{\omega(t)}$$

$$\mathbf{f}(t) = \frac{\mathbf{p}(t)}{\omega(t)} \;\Rightarrow\; \mathbf{p}(t) = \mathbf{f}(t)\omega(t) \;\Rightarrow\; \mathbf{p}'(t) = \mathbf{f}'(t)\omega(t) + \mathbf{f}(t)\omega'(t)$$

$$\Rightarrow\; \mathbf{f}'(t)\omega(t) = \mathbf{p}'(t) - \mathbf{f}(t)\omega'(t) \;\Rightarrow\; \mathbf{f}'(t) = \frac{\mathbf{p}'(t) - \mathbf{f}(t)\omega'(t)}{\omega(t)}$$

Derivatives

At the endpoints:

$$\mathbf{f}'(t) = \frac{\mathbf{p}'(t) - \omega'(t)\mathbf{f}(t)}{\omega(t)}$$

$$\mathbf{f}'(0) = \frac{\mathbf{p}'(0) - \omega'(0)\mathbf{f}(0)}{\omega(0)}$$

$$= \frac{d(\omega_1\mathbf{p}_1 - \omega_0\mathbf{p}_0) - d(\omega_1 - \omega_0)\mathbf{p}_0}{\omega_0} = \frac{d}{\omega_0}(\omega_1\mathbf{p}_1 - \omega_0\mathbf{p}_0 - \omega_1\mathbf{p}_0 + \omega_0\mathbf{p}_0)$$

$$= d\frac{\omega_1}{\omega_0}(\mathbf{p}_1 - \mathbf{p}_0)$$

$$\mathbf{f}'(1) = d\frac{\omega_{d-1}}{\omega_d}(\mathbf{p}_d - \mathbf{p}_{d-1})$$

NURBS:

Rational B-Splines

NURBS

NURBS: Rational B-Splines

- Same idea:
  - Control points in homogenous coordinates
  - Evaluate curve in (d+1)-dimensional space (same as before)
  - For display, divide by ω-component
    - (we can divide anytime)

NURBS

NURBS: Rational B-Splines

- Formally: ($N_i^{(d)}$: B-spline basis function *i* of degree *d*)

$$\mathbf{f}(t) = \frac{\sum_{i=1}^{n} N_i^{(d)}(t)\omega_i \mathbf{p}_i}{\sum_{i=1}^{n} N_i^{(d)}(t)\omega_i}$$

- Knot sequences etc. all remain the same
- De Boor algorithm – similar to rational de Casteljau alg.
  - 1. option – apply separately to numerator, denominator
  - 2. option – normalize weights in each intermediate result
    - The second option is numerically more stable

Some Issues

Interpolation problems:

- Finding a B-Spline curve that *interpolates* a set of *homogeneous* points is easy
- Just solve a linear system
- Note: The problem is easy when the weights are *given*.

What if no weights are given (only Euclidian points)?

- More degrees of freedom than constraints
- If we reduce the number of points:
  - Non-linear system of equations
  - Issues: How to find a solution? Does it exist? Is it unique?

Related Problem

Approximation with rational curves:

- **Scenario 1:** Homogeneous data points given, with weights
  - Easy problem – linear system
- **Scenario 2:** Euclidian data points are given, but weights are fixed for each control point (e.g. manually)
  - Easy problem again – linear system
  - Weights just change the shape of the basis functions
- **Scenario 3:** Euclidian data points, want to compute weights as well
  - Non-linear optimization problem

General Rational Data Approximation

Scenerio 3: Euclidian data points, want to compute weights as well

- Non-linear optimization problem
- Issues:
  - No direct solution possible
  - Numerical optimization might get stuck in local minima
- Constraints:
  - We have to avoid poles
  - E.g. by demanding $\omega_i > 0$
  - Constrained optimization problem (even nastier)

General Rational Data Approximation

Simple idea for a numerical approach:

- First solve non-rational problem (all weights = 1)
- Then start constrained non-linear gradient descend (or Newton) solver from there