Curves and surfaces & modeling case studies

Karan Singh

Dynamic Graphics Project
University of Toronto
www.dgp.toronto.edu

Curves and Surfaces

- Polynomial curves from constraints: Hermites.
- Basis functions: Beziers, BSplines.
- Coons Interpolation: Coons patches.
- Desirable curve properties.
- Other curve formulations: clothoids.

Parametric Polynomial Curves

Recall a linear curve (line) is: $p(t) = a_1 t + a_0$

A cubic curve is similarly:

$$
\begin{aligned}
x(t) &= a_3 t^3 + a_2 t^2 + a_1 t + a_0 \\
y(t) &= b_3 t^3 + b_2 t^2 + b_1 t + b_0 \\
z(t) &= c_3 t^3 + c_2 t^2 + c_1 t + c_0
\end{aligned}
$$

…or $p(t) = d_3 t^3 + d_2 t^2 + d_1 t + d_0$, where $d_i = [a_i, b_i, c_i,]^T$

Cubics are commonly used in graphics because:

- curves of lower order have too little flexibility (only planar, no curvature control).
- curves of higher order are unnecessarily complex and easily wiggle.

Polynomial curves from constraints

p(t) = TA , where T is powers of t. for a cubic $T=[t^3 \; t^2 \; t^1 \; 1]$.

Written with geometric constraints p(t) = TMG,
where M is the **Basis matrix** of the curve, G the design constraints.

An example of constraints for a cubic Hermite for eg. are
end points and end tangents. i.e. P_1,R_1 at t=0 and P_4,R_4 at t=1.
Plugging these constraints into p(t) = TA we get.

$$
\begin{array}{llcl}
 & & \mathrm{H} & \\
p(0) = P_1 & = & [\,0\;0\;0\;1\,] & A \\
p(1) = P_4 & = & [\,1\;1\;1\;1\,] & A \\
p'(0) = R_1 & = & [\,0\;0\;1\;0\,] & A \\
p'(1) = R_4 & = & [\,3\;2\;1\;0\,] & A
\end{array}
\quad \Rightarrow \quad G=HA \qquad \text{TA=TMG \&} \quad \Rightarrow H=M^{-1}
$$

Bezier Basis Matrix

A cubic Bezier can be defined with four points where:
P_1,R_1 at t=0 and P_4,R_4 at t=1 for a Hermite.
$R_1 = 3(P_2-P_1)$ and $R_4 = 3(P_4-P_3)$.

We can thus compute the Bezier Basis Matrix by finding the matrix that transforms $[P_1\ P_2\ P_3\ P_4\ ]^T$ into $[P_1\ P_4\ R_1\ R_4\ ]^T$ i.e.

B_H = [1 0 0 0]
[0 0 0 1]
[-3 3 0 0]
[0 0 -3 3]

$M_{bezier}=M_{hermite} * B\_H$

Bezier Basis Functions

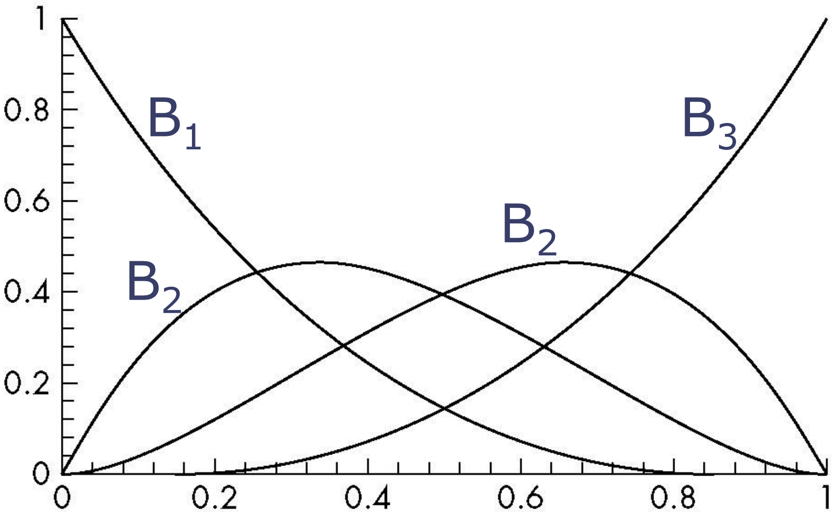
[-1 3 -3 1]
[3 -6 3 0]
[-3 3 0 0]
[1 0 0 0]

The columns of the Basis Matrix form Basis Functions such that:
$p(t) = B_1(t)P_1 + B_2(t)P_2 + B_3(t)P_3 + B_4(t)P_4.$

From the matrix:

$B_{i+1}(t) = \binom{n}{i} * (1-t)^{(n-i)} * t^i$

…also called Bernstein polynomials.



Basis Functions

Basis functions can be thought of as an influence weight that each constraint has as *t* varies.

Note: actual interpolation of any constraint only happens if its Basis function is 1 and all others are zero at some *t*.

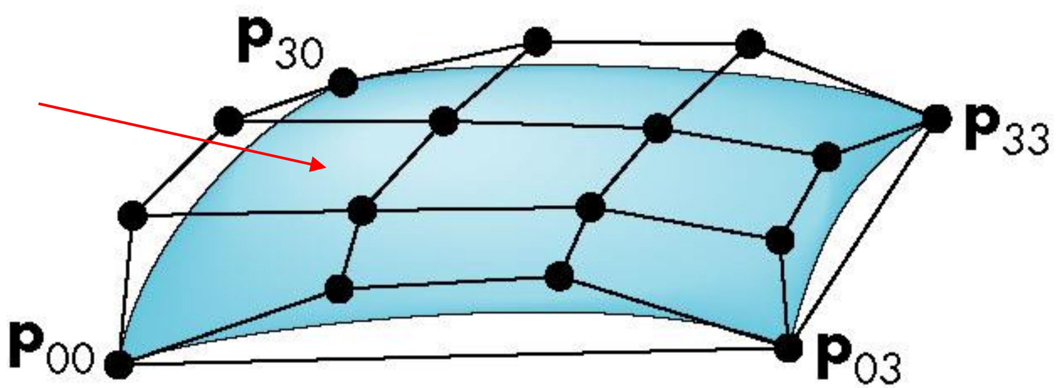
Often Basis functions for design curves sum to 1 for all *t*.
This gives the curve some nice properties like affine invariance and the convex hull property when the functions are additionally non-negative.

Bezier Patches

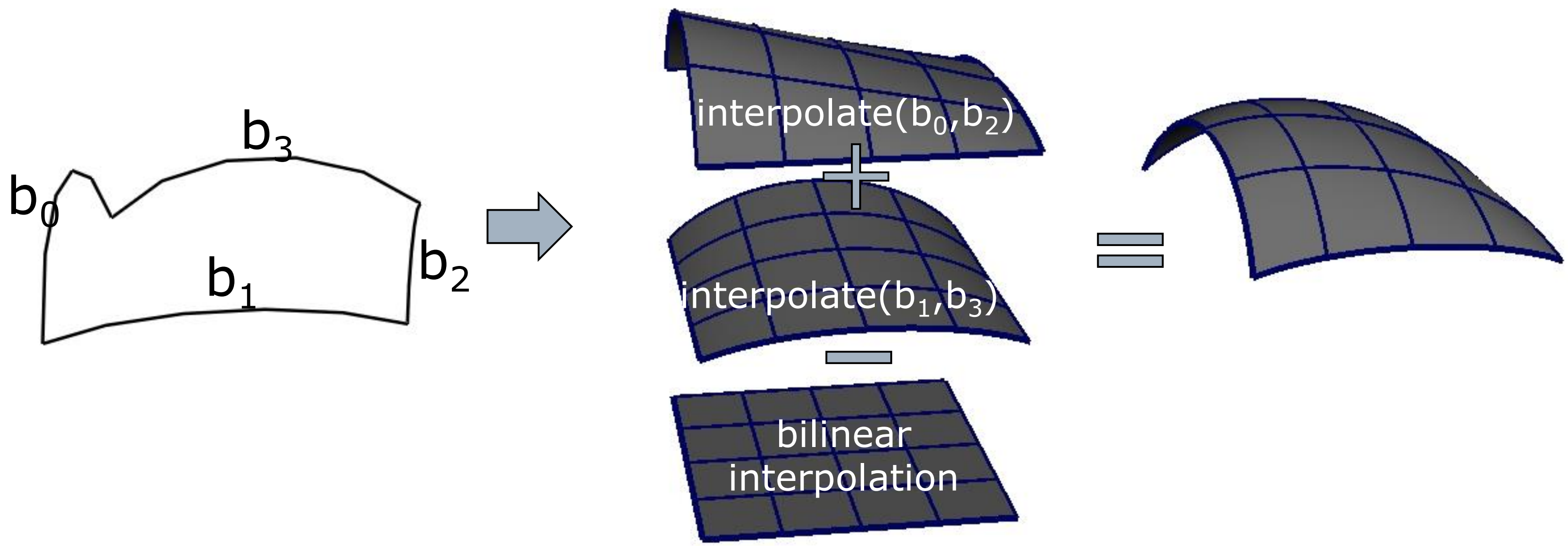
Using same data array $\mathbf{P}=[p_{ij}]$ as with interpolating form:

$$p(u,v) = \Sigma_i \Sigma_j B_i(u) B_j(v) p_{ij} = u^T \mathbf{B} \mathbf{P} \mathbf{B}^T v$$

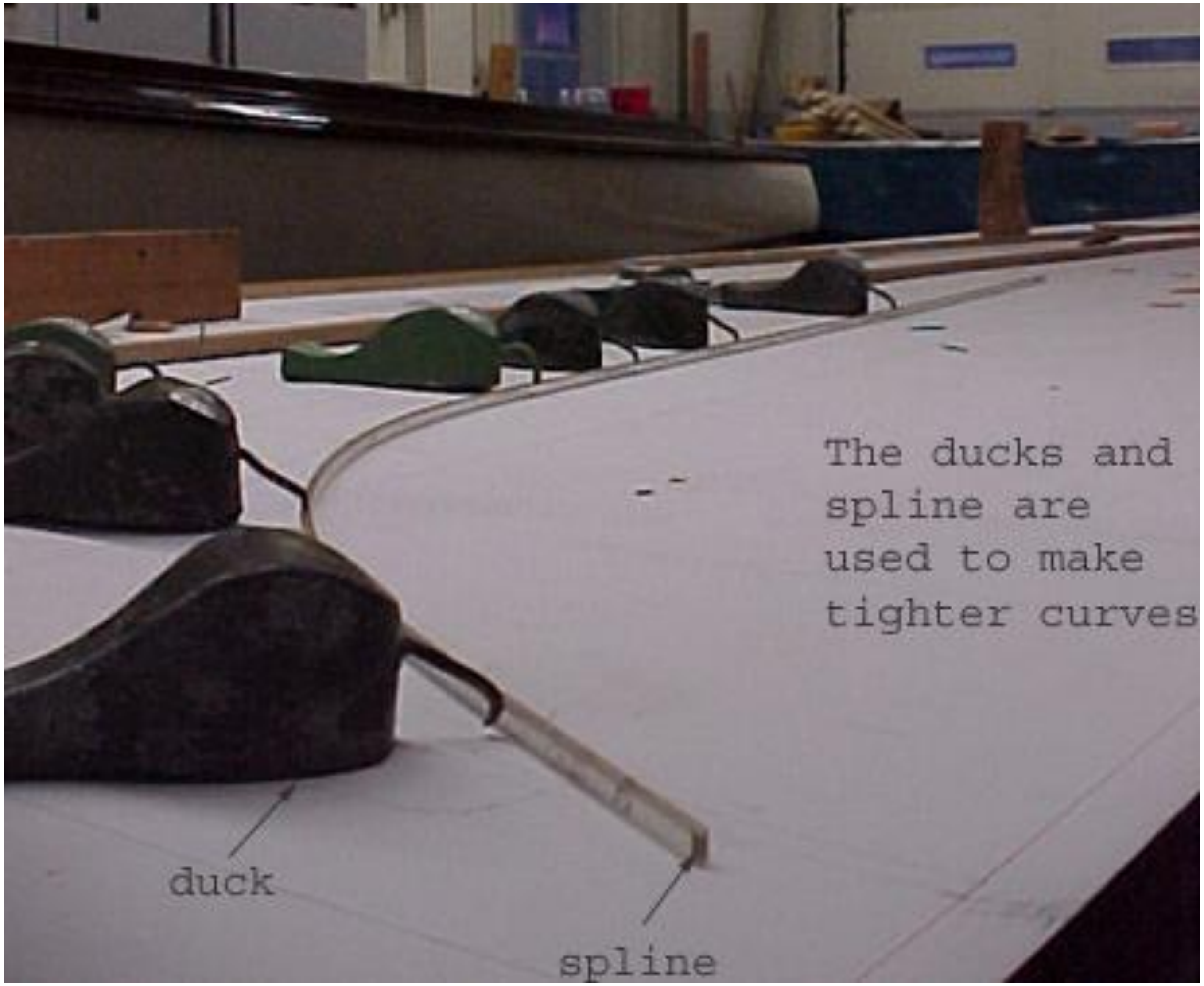
Patch lies in convex hull



Coons Patches: only boundary curves

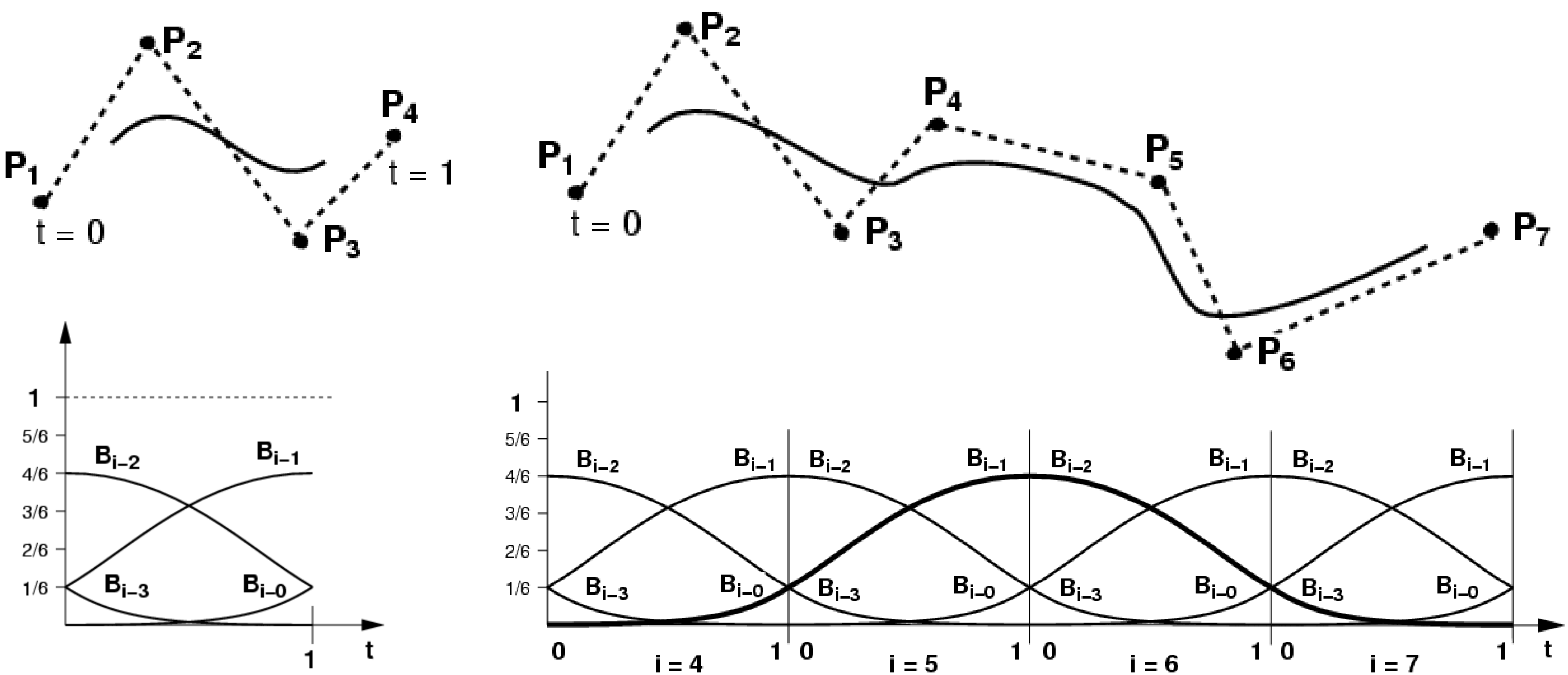


Traditional Splines



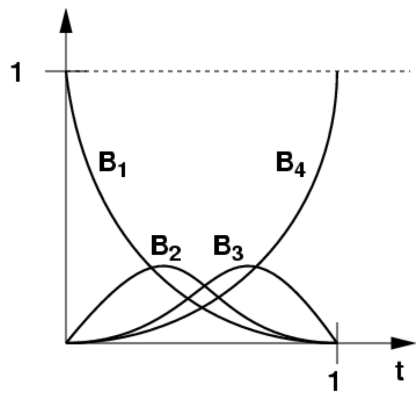
BSpline Basis Functions

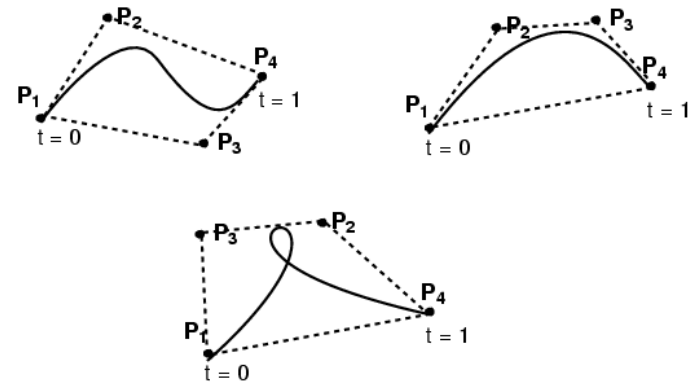
- Can be chained together.
- Local control (windowing).



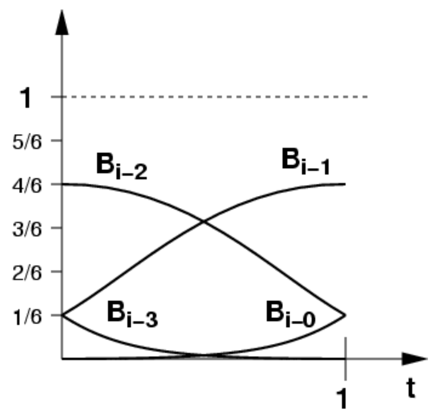
Bezier vs. BSpline

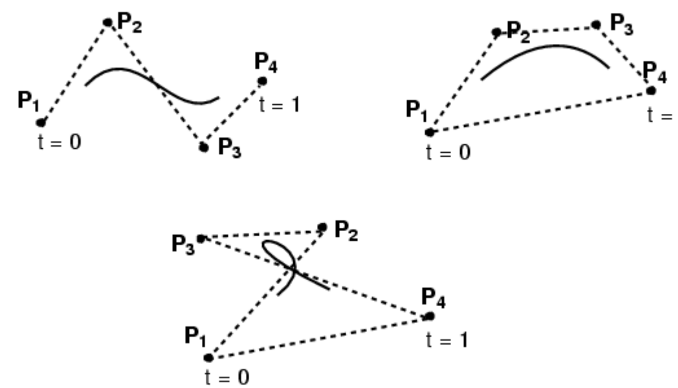
Bezier





BSpline





Representing a conic as a polynomial

- $<x(t),y(t)> = < \cos(t), \sin(t) >$
- Taylor series for $\sin(t)= t - t^3/3! + t^5/5! \ldots$
- $u=\tan(t/2)$
- $<x(u),y(u)> = < (1-u^2)/(1+u^2) , 2u/(1+u^2) >$
- Rational Bspline's are defined with homogeneous coordinates using $w(t)$.
- NURBS additionally adds non-uniform knots.

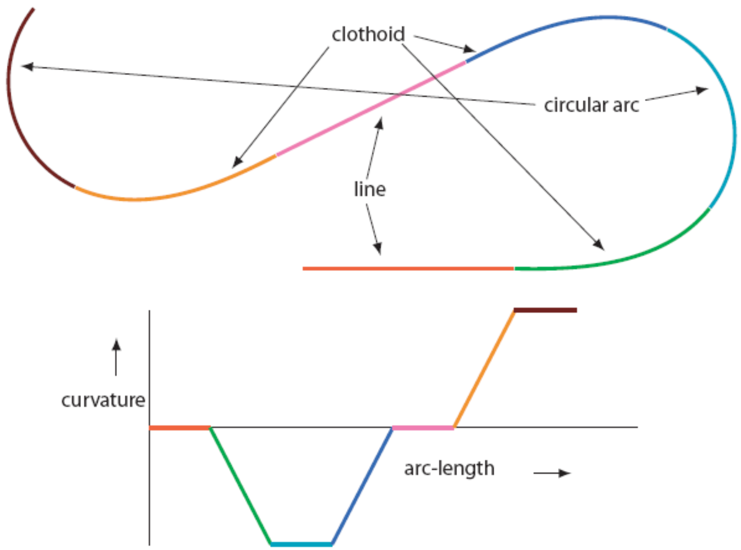
Curve Design Issues

- Continuity (smoothness, fairness and neatness).
- Control (local vs. global).
- Interpolation vs. approximation of constraints.
- Other geometric properties (planarity, invariance).
- Efficient analytic representation.

Smooth curves

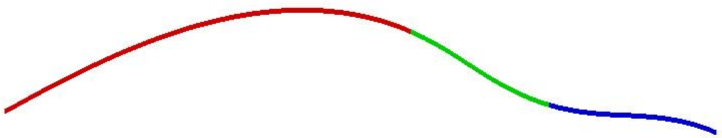
- **Fairness**: *"curvature continuous curves with a small number of segments of almost piecewise linear curvature"* [Farin et al. 87].

- Lines, circles and clothoids are the simplest primitives in curvature space.



Clothoids

Sketch curves often represent gestural information or capture design intent where the overall stroke appearance (*fairness*) is more important than the precise input.



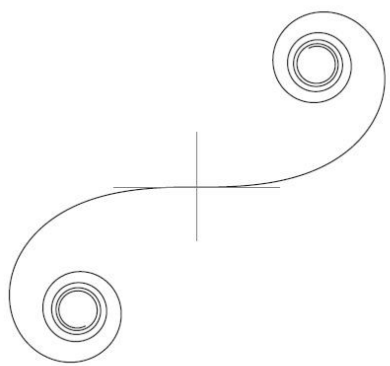
[**McCrae & Singh**, Sketching Piecewise Clothoid Curves, *SBIM 2008*]
http://www.dgp.toronto.edu/~mccrae/clothoid/

What are Clothoids?

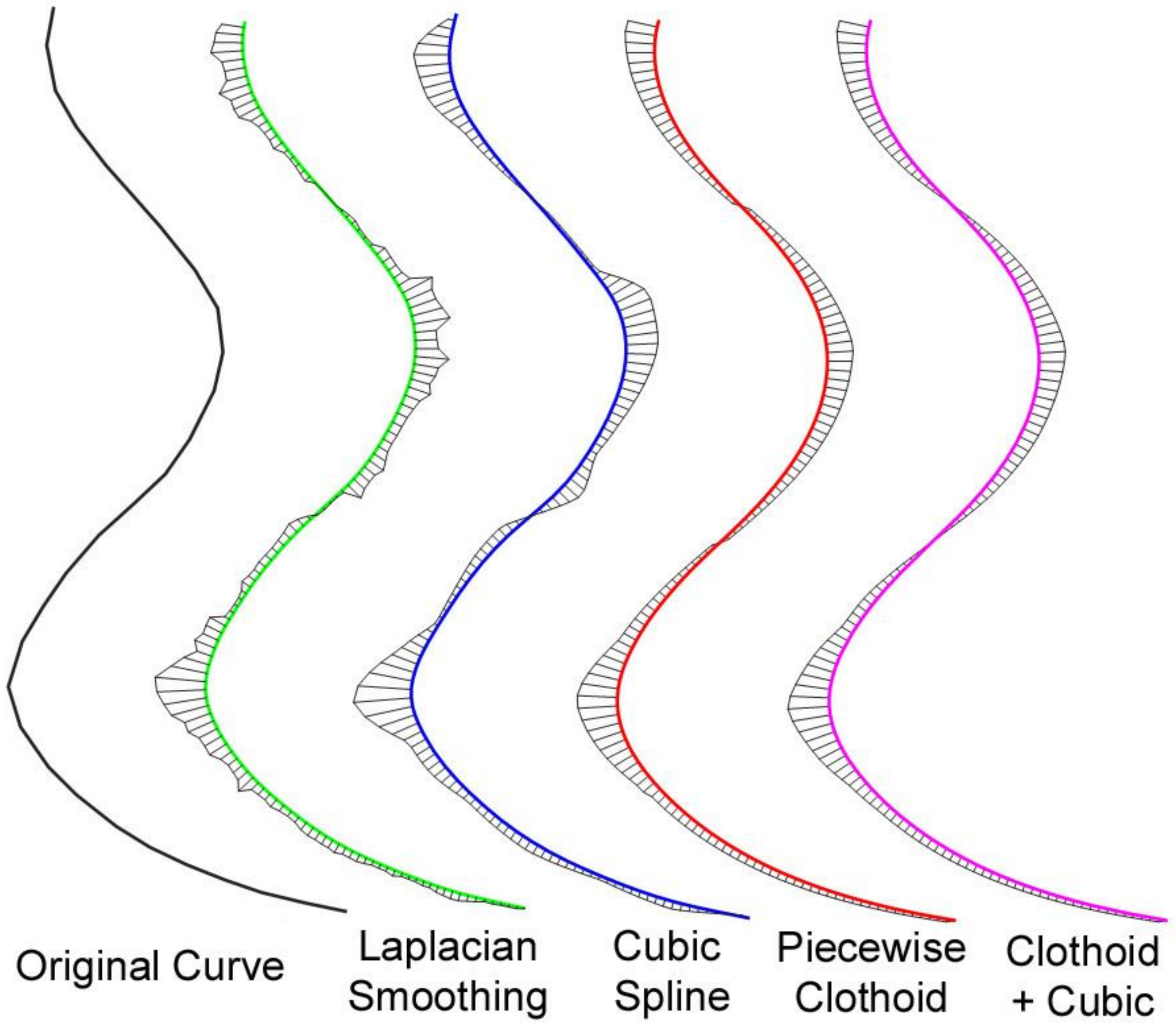
- Curves whose curvature changes linearly with arc-length.
- Described by Euler in 1774, a.k.a. Euler spiral.
- Studied in diffraction physics, transportation engineering (constant lateral acceleration) and robot vehicle design (linear steering).

$$C(t) = \int_0^t \cos \frac{\pi}{2} u^2 du$$

$$S(t) = \int_0^t \sin \frac{\pi}{2} u^2 du$$



Comparative approaches to fairing

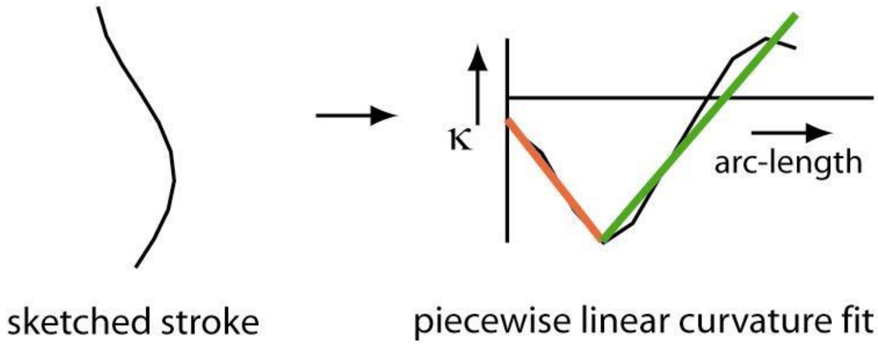


Approach

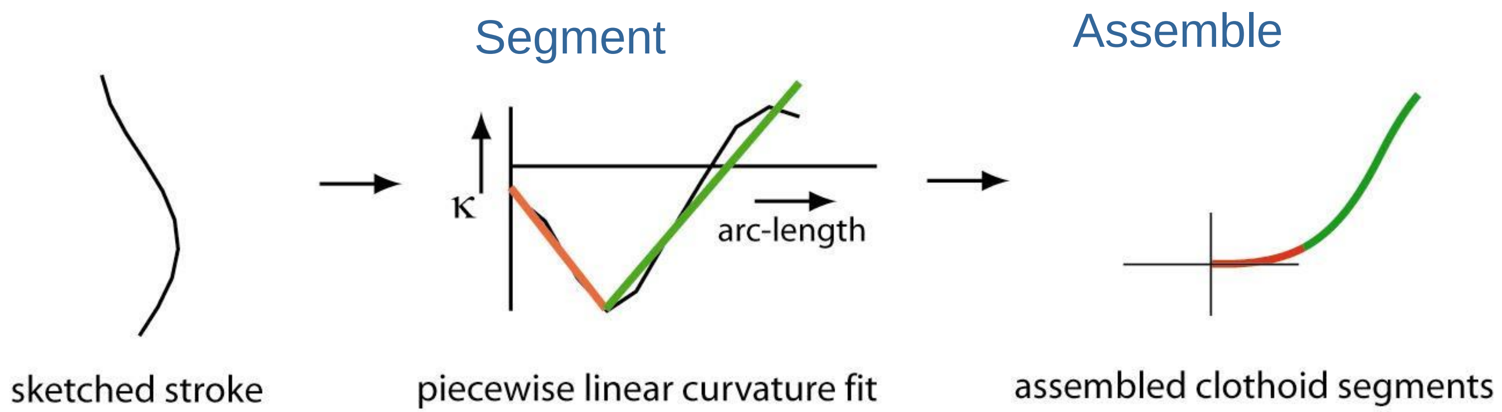
sketched stroke

Approach

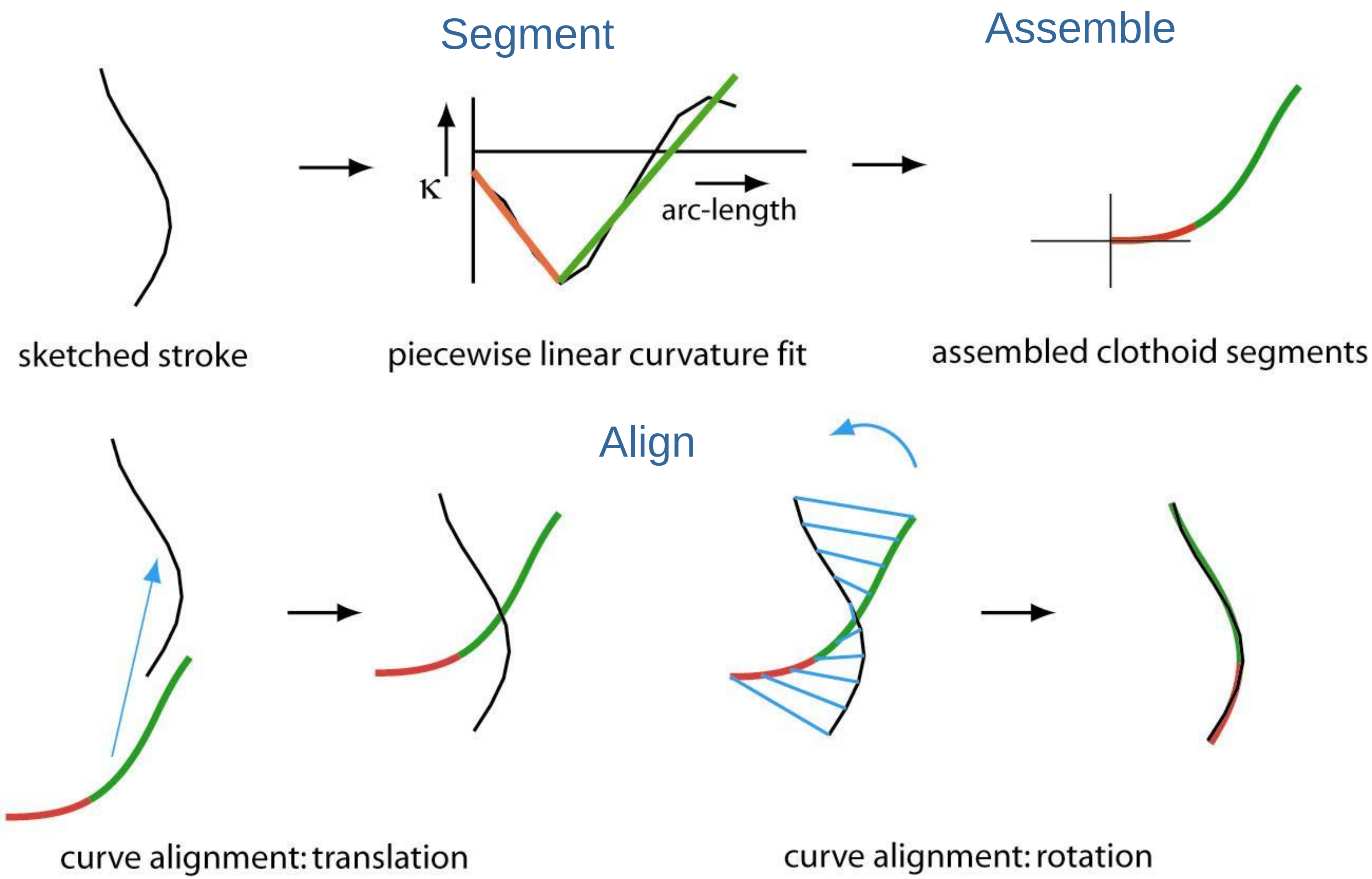
Segment



Approach

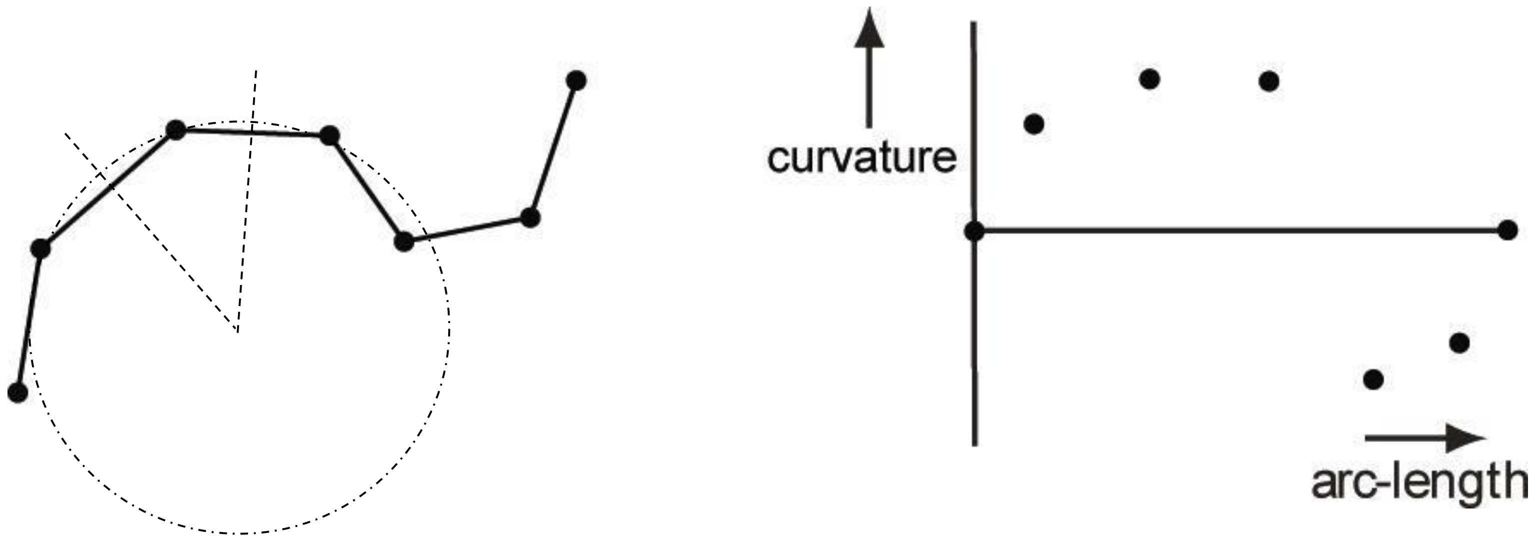

Segment
Assemble
κ
arc-length
sketched stroke
piecewise linear curvature fit
assembled clothoid segments

Approach


Segment
Assemble
κ
arc-length
sketched stroke
piecewise linear curvature fit
assembled clothoid segments
Align
curve alignment: translation
curve alignment: rotation

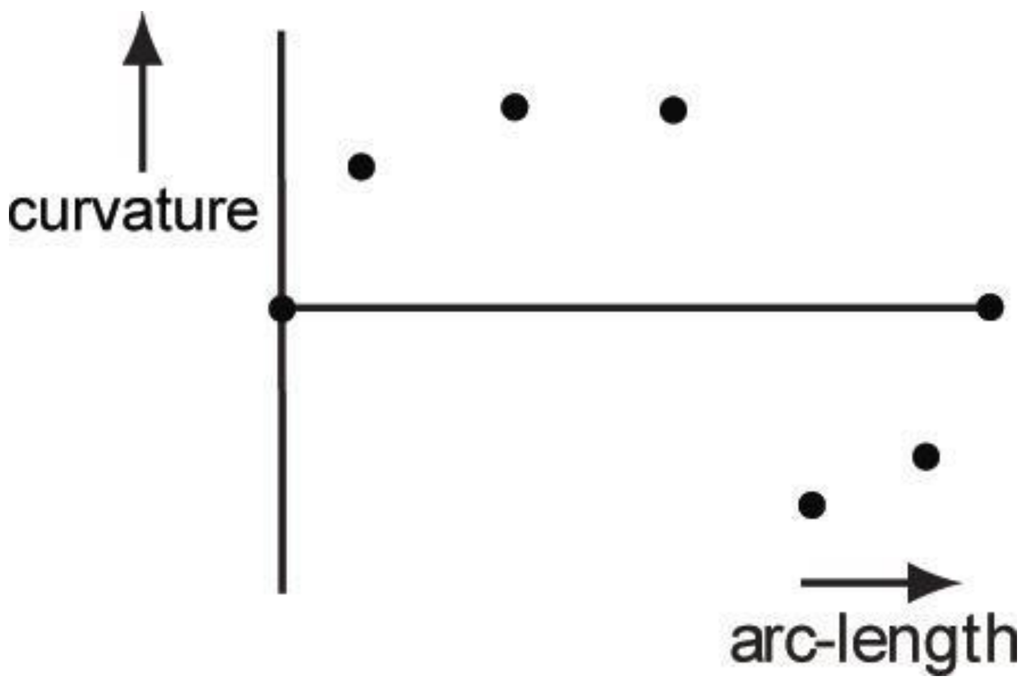
Piecewise Linear Curvature Fit

- Any discrete curvature estimator can be used to obtain curvature space points



Piecewise Linear Curvature Fit

- Find a ***small number*** of connected line segments that ***minimize fit error.***



Piecewise Linear Curvature Fit

- **Dynamic programming** (cost of fit matrix *M)*:

$$M(a,b) = \min_{a<k<b} \{M(a,k)+M(k,b), E_{fit}(a,b)+E_{cost}\}$$

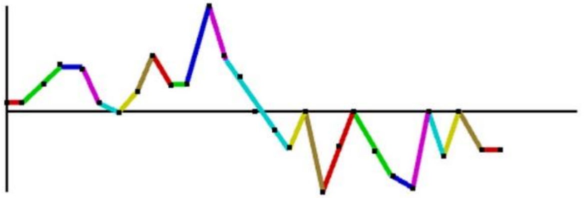
$E_{fit}(a,b)$ is the fitting error of a line to points *a..b.*

E_{cost} is the penalty incurred to increment the number of line segments.

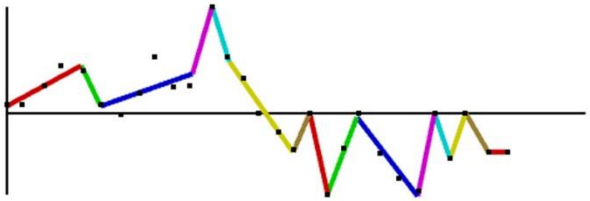
Can be used to fit other primitives like circle involute.

Piecewise Linear Curvature Fit

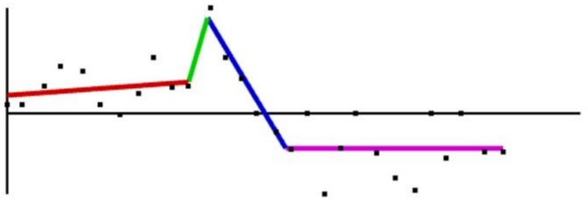
$E_{cost} = 0.01$



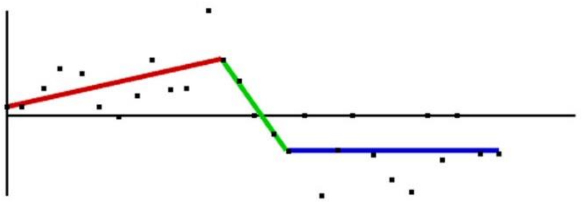
$E_{cost} = 0.02$



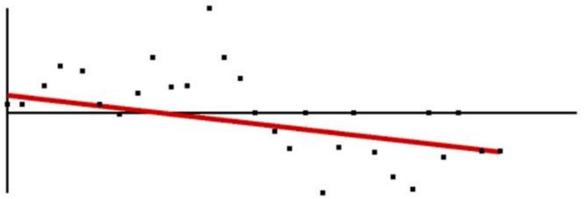
$E_{cost} = 0.035$



$E_{cost} = 0.05$

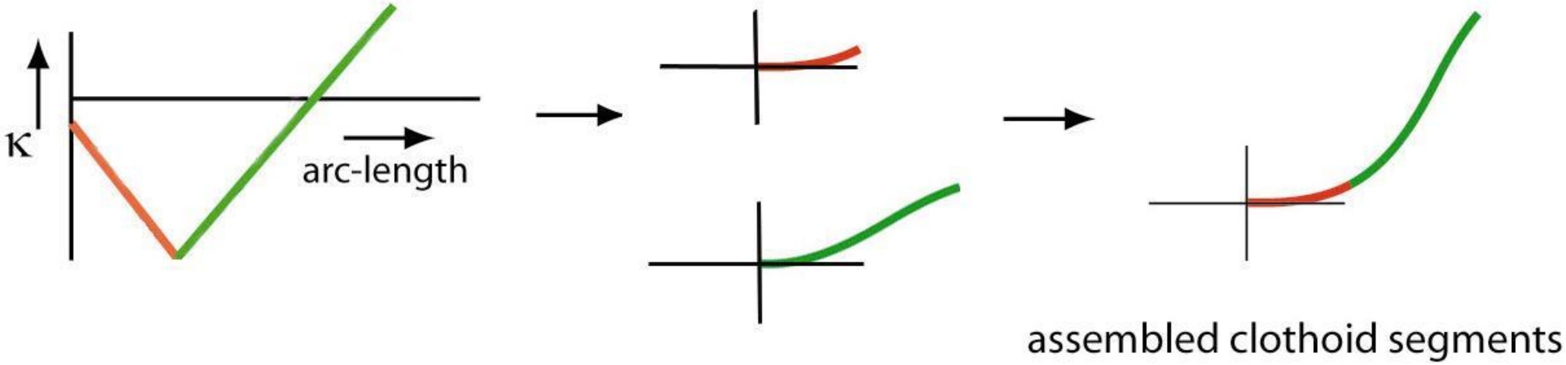


$E_{cost} = 0.08$



Assembly

- To assemble piecewise clothoid curve:
  - Map each curvature space line segment to a unique line, circle or clothoid curve segment.
  - Attach segments so they are position/tangent continuous
- Resulting curve has G^2 continuity

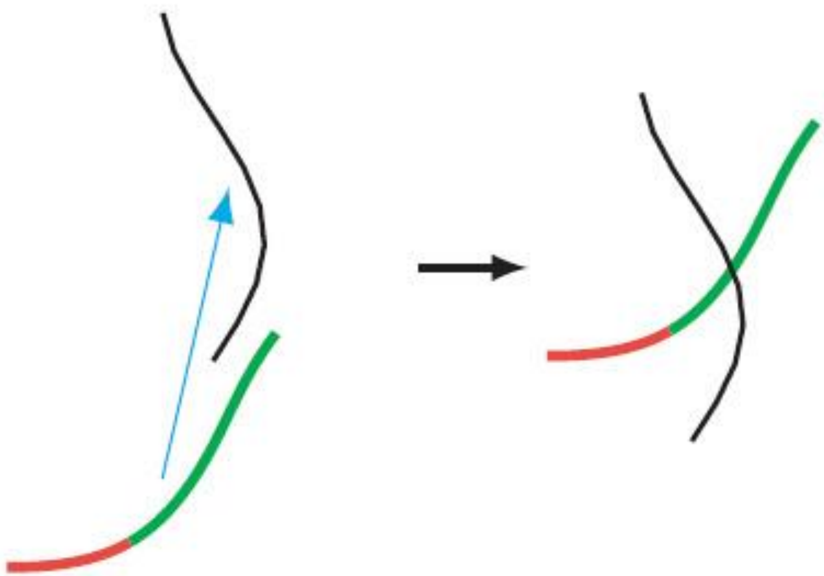


Align

Find a rigid transform that minimizes the sum of squared distance between arc-length corresponding points on the input polyline and piecewise clothoid curve. [Horn 1987]

Curve Alignment: Translation

- Translation is difference between the centroids of the points along both curves



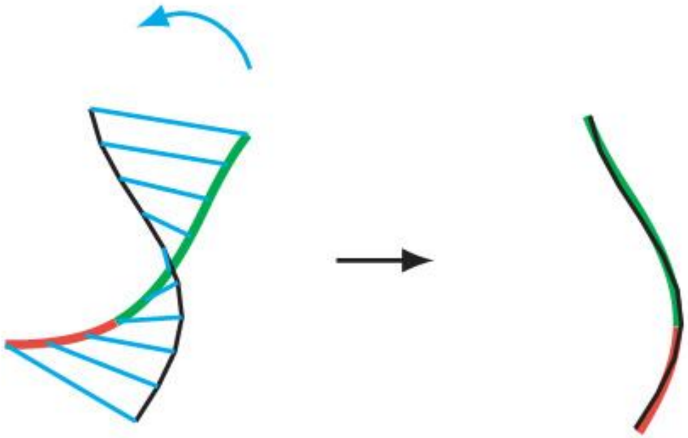
curve alignment: translation

Curve Alignment: Rotation

- Rotation minimizes weighted squared distances:
- Optimal A given by: $\sum_{i=0}^{n-1} w_i(Aq_i - p_i)^2$
- Rotation R extracted as

$$A = (\sum_{i=0}^{n-1} w_i p_i q_i^T)(\sum_{i=0}^{n-1} w_i q_i q_i^T)^{-1} = A_{pq}A_{qq}$$,where

$$R = A_{pq}S^{-1} \qquad S = \sqrt{A_{pq}^T A_{pq}}$$



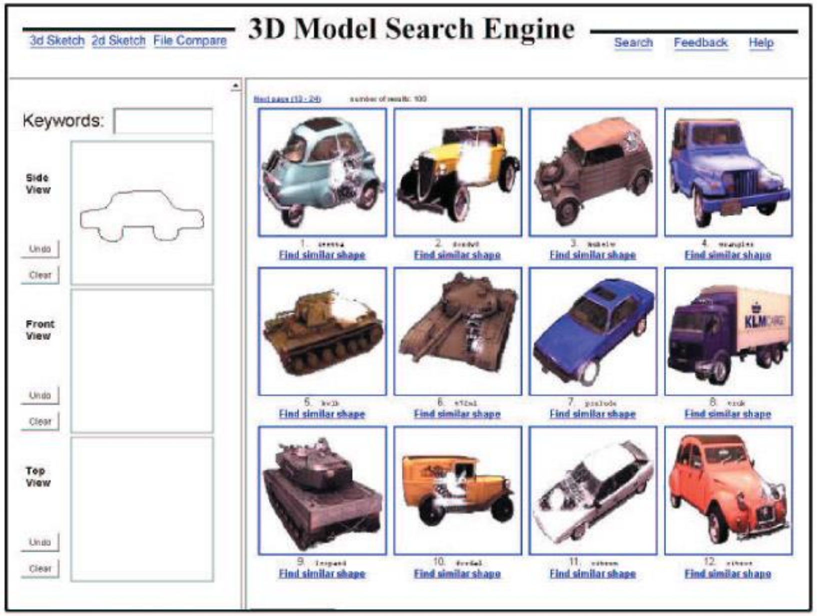
curve alignment: rotation

Model creation categories

- Suggestive systems
  - Input compared to template objects
  - *symbolic* or *visual memory*
- Constructive systems
  - Input directly used to create object
  - *perceptual* or *visual rules*

Suggestive systems

- User draws complete or gestural sketch.
- Sketch matched against object database or known primitives.



Funkhouser et al., *A Search Engine for 3D Models*, Proc. of SIGGRAPH'03, 2003.

Suggestive systems (matching 2D to 3D)

- Extract several contours for each object.
- Create feature vector
  - Direct comparison, eg. Euclidean distance.

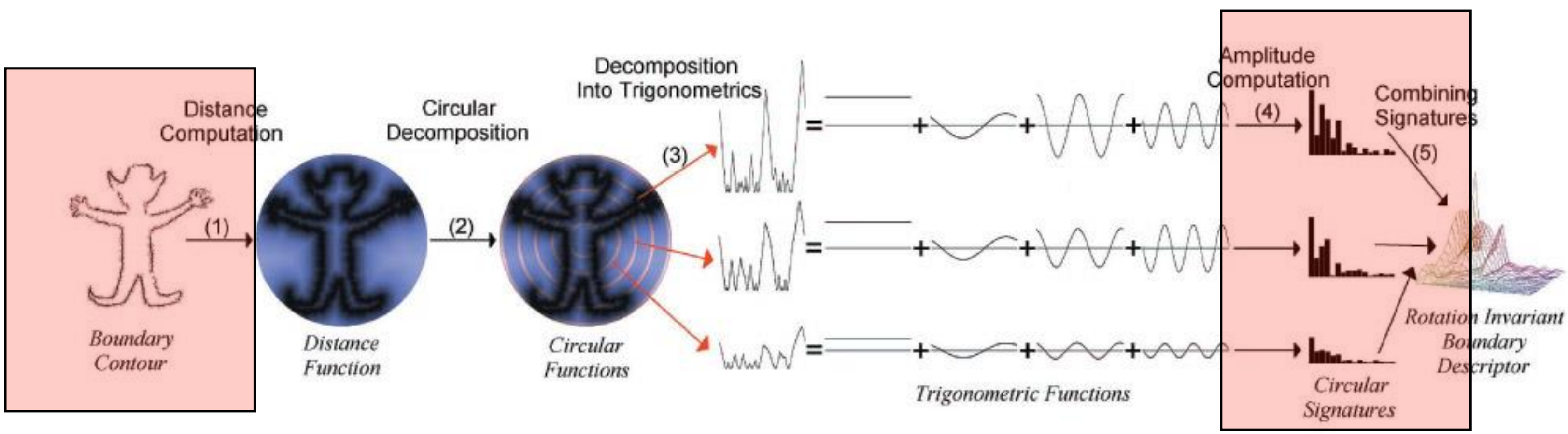
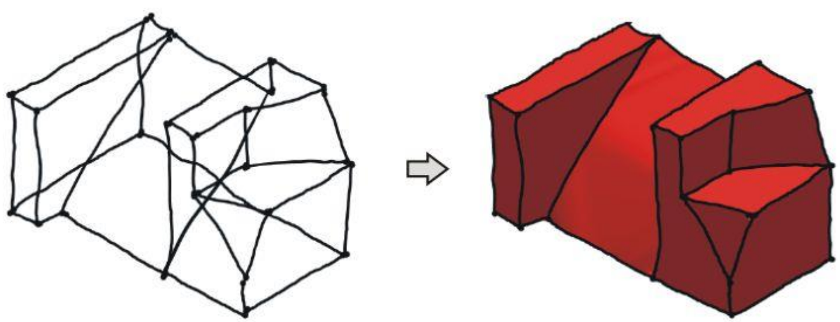


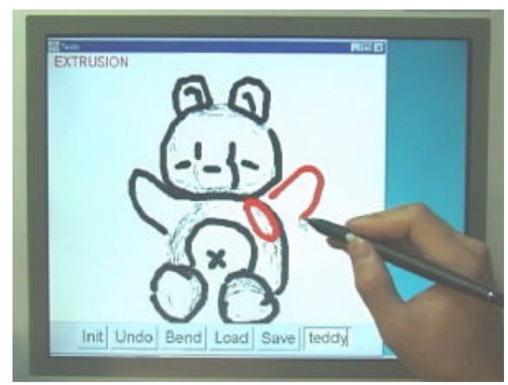
Fig. 9. Computing our shape descriptor for boundary contours.

Funkhouser et al., *A Search Engine for 3D Models*, Proc. of SIGGRAPH'03, 2003.

Constructive systems

- Rules and constraints rather than templates:
  - Restricting application domain (eg. sketching roads).
  - Restricting object type (eg. mechanical or organic).
  - Restricting task (eg. smoothing, cutting or joining).





M. Masry and H. Lipson, *A Sketch-Based Interface for Iterative Design and Analysis of 3D Objects*, EG SBIM'05, 2005.
T. Igarashi et al., ***Teddy**: A Sketching Interface for 3D Freeform Design*, Proc. of SIGGRAPH'99, 1999.

Case Studies

- Drive.
- Teddy, Fibermesh.
- ILoveSketch.
- Analytic drawing.
- True2Form.
- SecondSkin.

Drive: single-view sketching

A sketch-based system to create conceptual layouts of 3D path networks.

Drive features

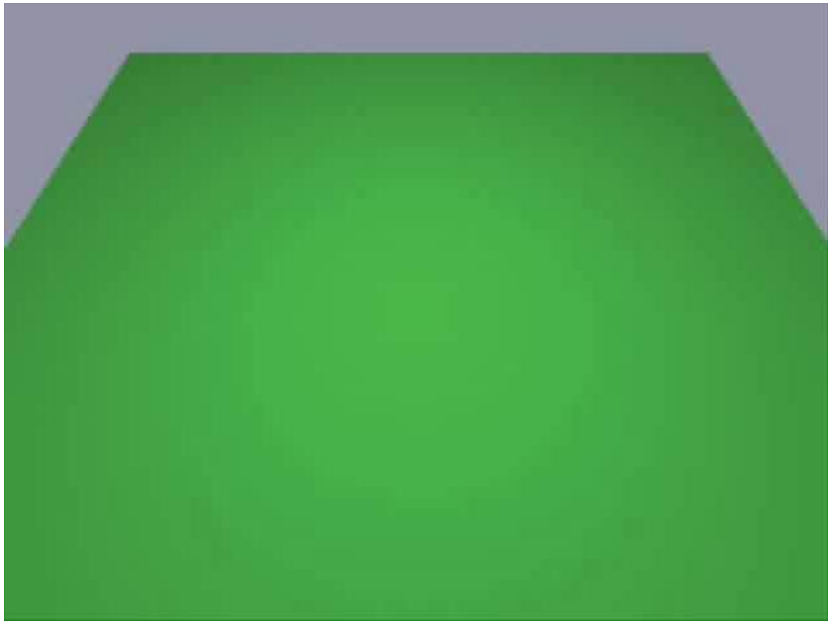
- Elegant interface:

  open stroke = path

  closed stroke = selection-action menu.
- Piecewise clothoid path construction.
- Crossing paths.
- Break-out lens. (single-view context)
- Terrain sensitive sketching.

[**McCrae & Singh**, Sketching based Path Design, *Graphics Interface 2009*]

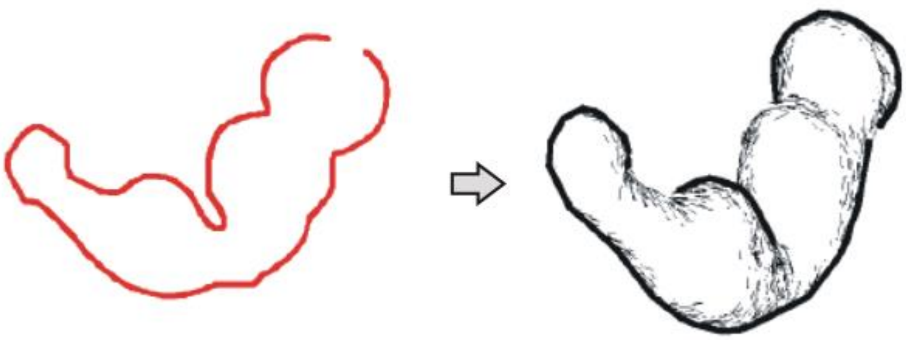
Drive



[**McCrae & Singh**, Sketching based Path Design, *Graphics Interface 2009*]

Teddy

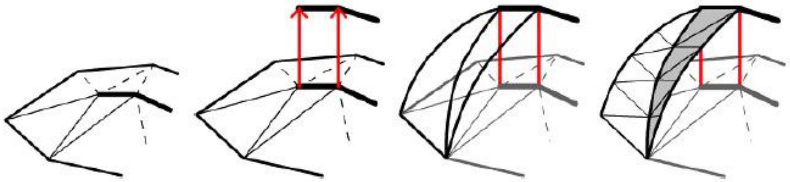
- Teddy inflates a closed 2D stroke like blowing up a balloon.



T. Igarashi et al., ***Teddy**: A Sketching Interface for 3D Freeform Design*, Proc. of SIGGRAPH'99, 1999.

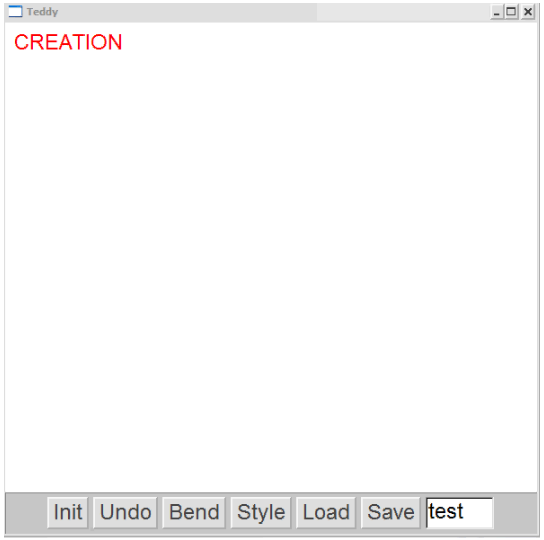
Inflation

- Offset surface proportionally to distance from spine of the contour
- Produces smooth blobby objects



a) before b) elevate spines c) elevate edges d) sew elevated edges

Figure 15: Polygonal mesh construction.



Igarashi et al., ***Teddy****: A Sketching Interface for 3D Freeform Design*, SIGGRAPH’99, 1999.

Skeleton extraction

- Delaunay triangulation
- Chordal axis transform

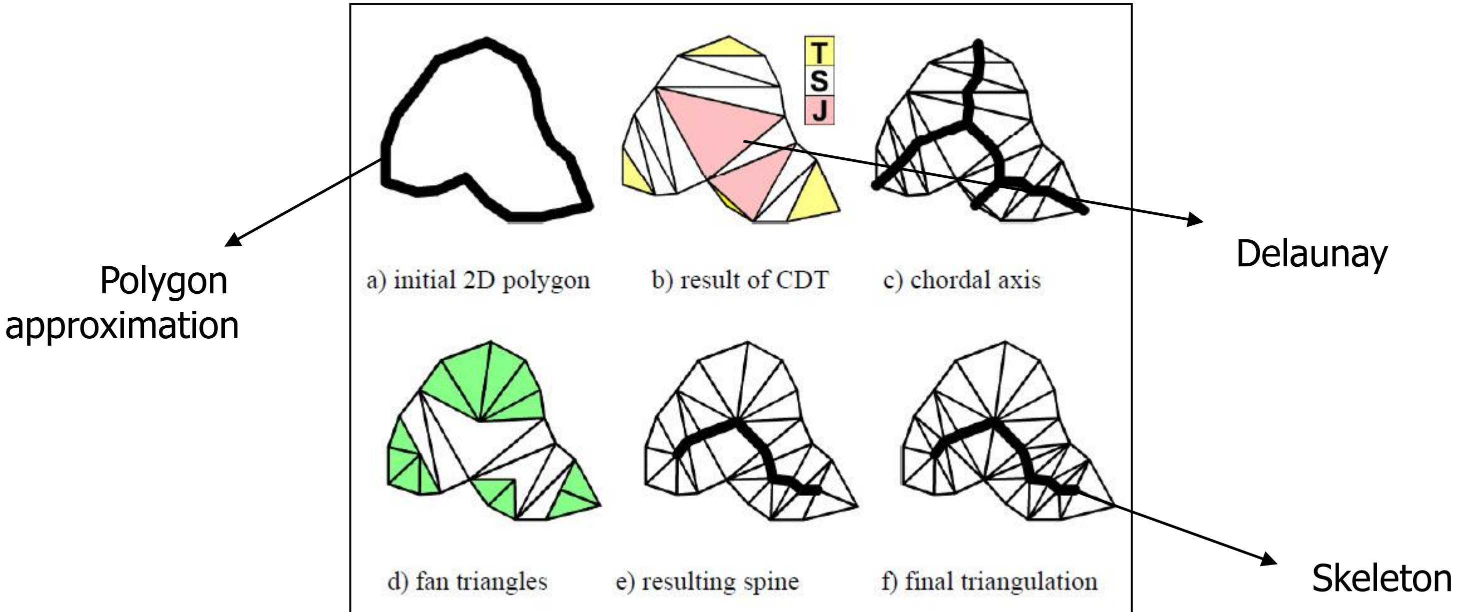
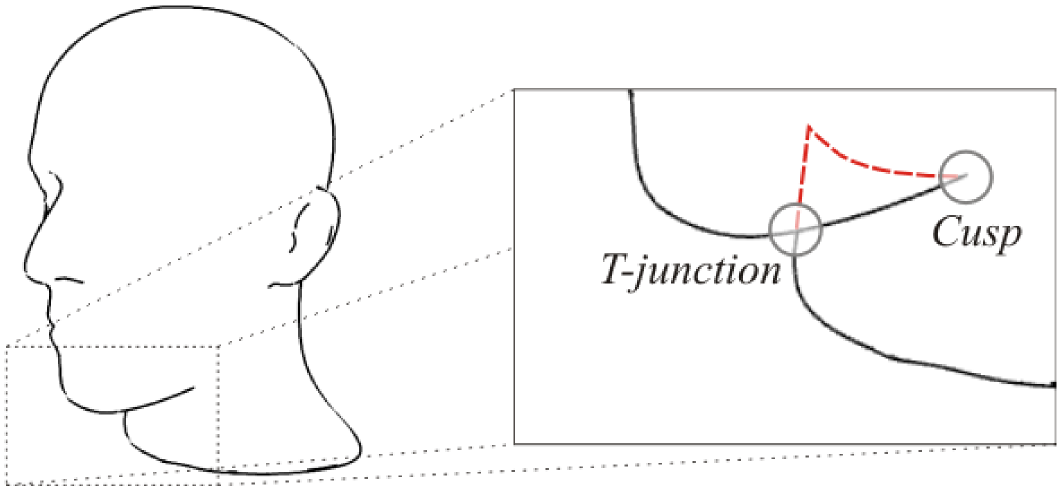


Figure 13: Finding the spine.

Igarashi et al., ***Teddy****: A Sketching Interface for 3D Freeform Design*, SIGGRAPH'99, 1999.

Trouble with contours and silhouettes

- Rarely planar.
- Can contain T-junctions and cusps.
- Occlusion.



3D Curve networks: surface optimization

- Surface results from solving non-linear system
  - 3D curves defines geometric constraints
  - Smoothness constraints

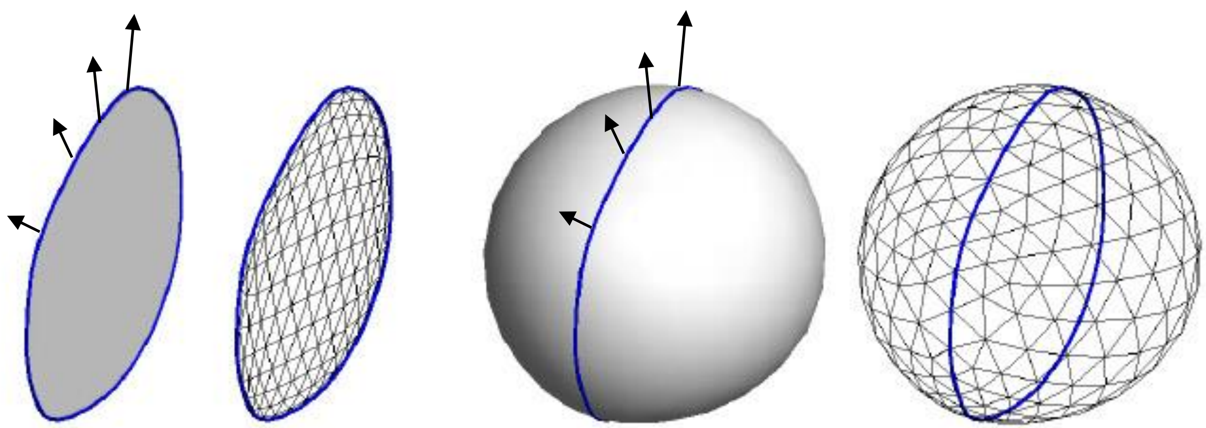
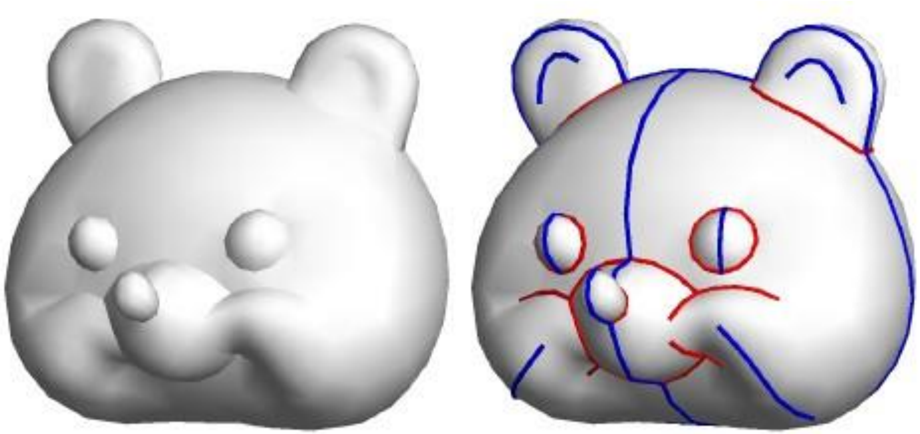


Figure 11: *The results of least-squares meshes (left) and our non-linear solution (right) for a planar curve.*

A. Nealen et al., ***FiberMesh****: Designing Freeform Surfaces with 3D Curves*, Proc. of SIGGRAPH'07, 2007.

FiberMesh

- User can specify additional curves on the surface
  - Further constraints that define the surface
  - Sharp features



A. Nealen et al., ***FiberMesh****: Designing Freeform Surfaces with 3D Curves*, Proc. of SIGGRAPH'07, 2007.

I ♥ SKETCH: multi-view sketching

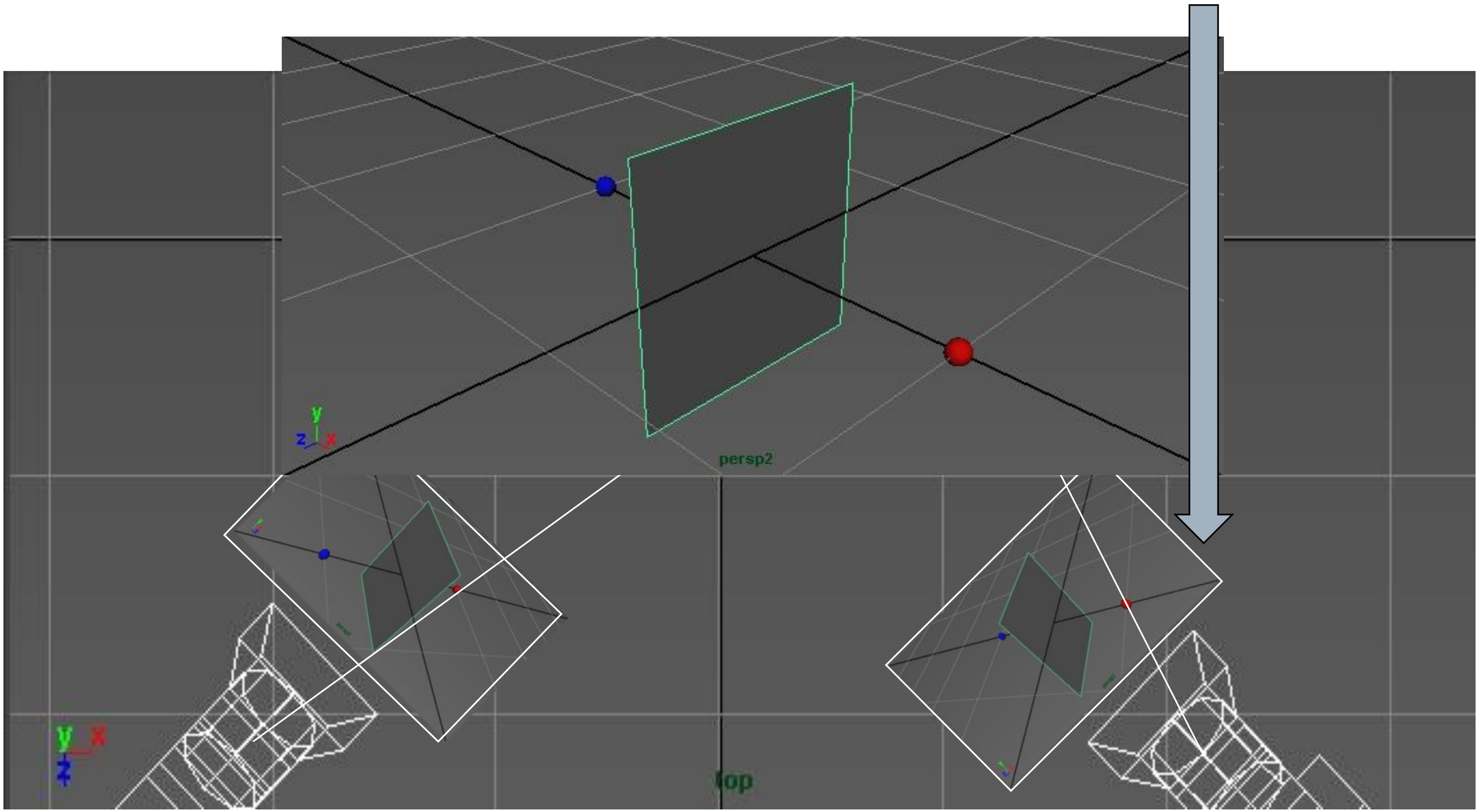
I ♥ SKETCH: multi-view sketching

A judicious leap from 2D to 3D.

- Presents a virtual 2D sketchbook with simple paper navigation and automatic rotation for ergonomic *pentimenti* style 2D sketching.
- Seamless transition to 3D with a suite of *multi-view curve sketching* tools with context switching based on *sketchability*.

[**Bae, Balakrishnan & Singh**, ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. *UIST 2008*]
www.ilovesketch.com

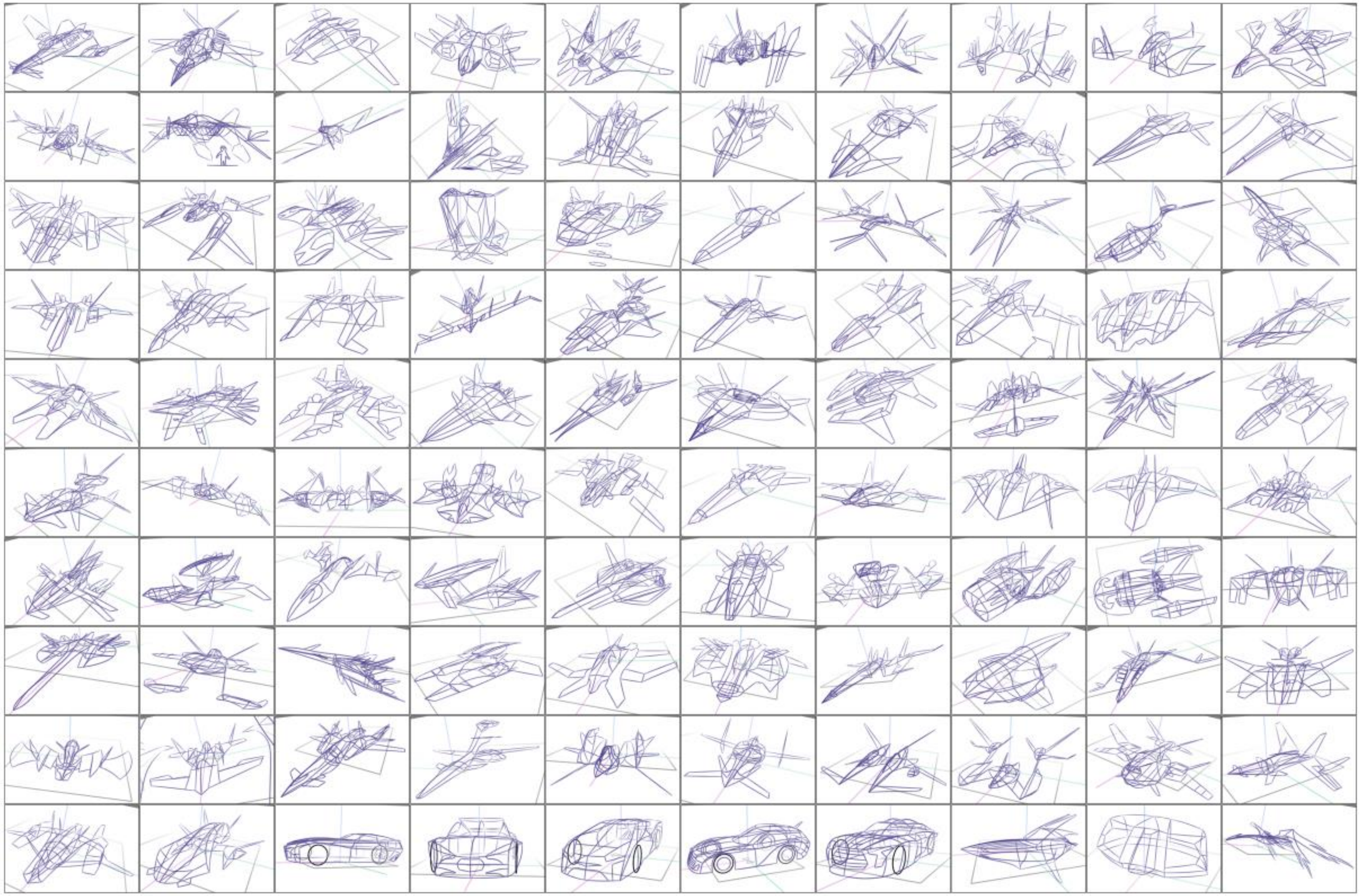
I ♥ SKETCH : epi-polar symmetry



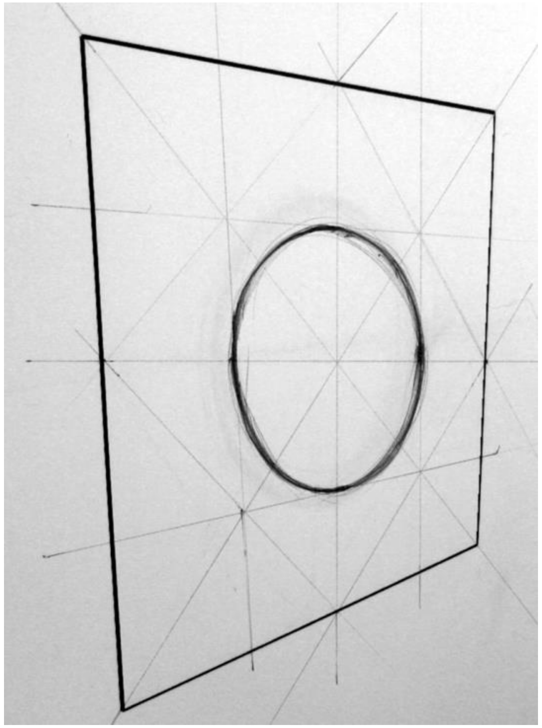
I ♥ SKETCH (at SIGGRAPH 09 eTech)

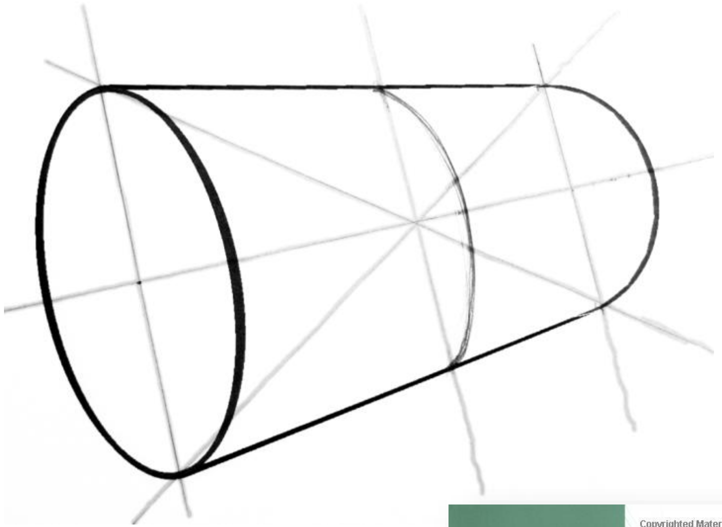
100 models created over 4 days (made public for research)

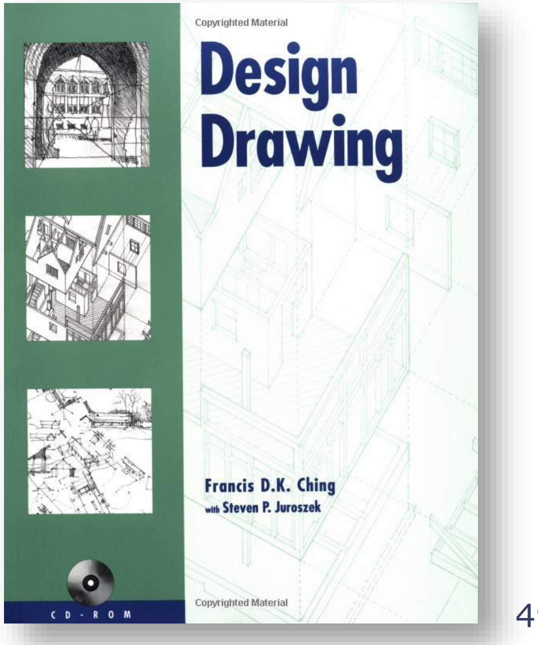
http://www.dgp.toronto.edu/~shbae/ilovesketch_siggraph2009.htm



Experts and drawing systems

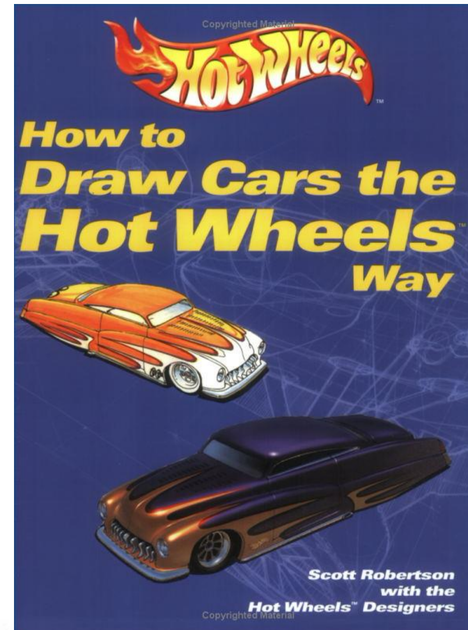


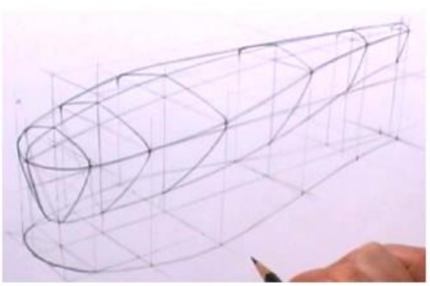


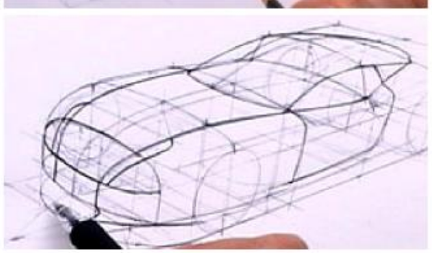


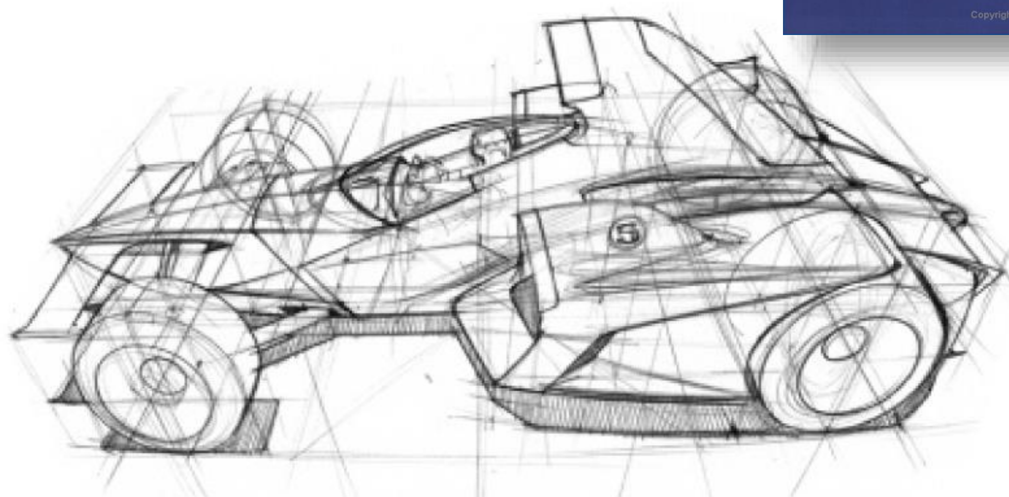
Analytic Drawing: single-view sketching

1. Pick a drawing system
   - 2-point perspective, isometric,…
   - Rules for how to interpret lines
2. Construct a 3D scaffold
3. Draw curves within the scaffold

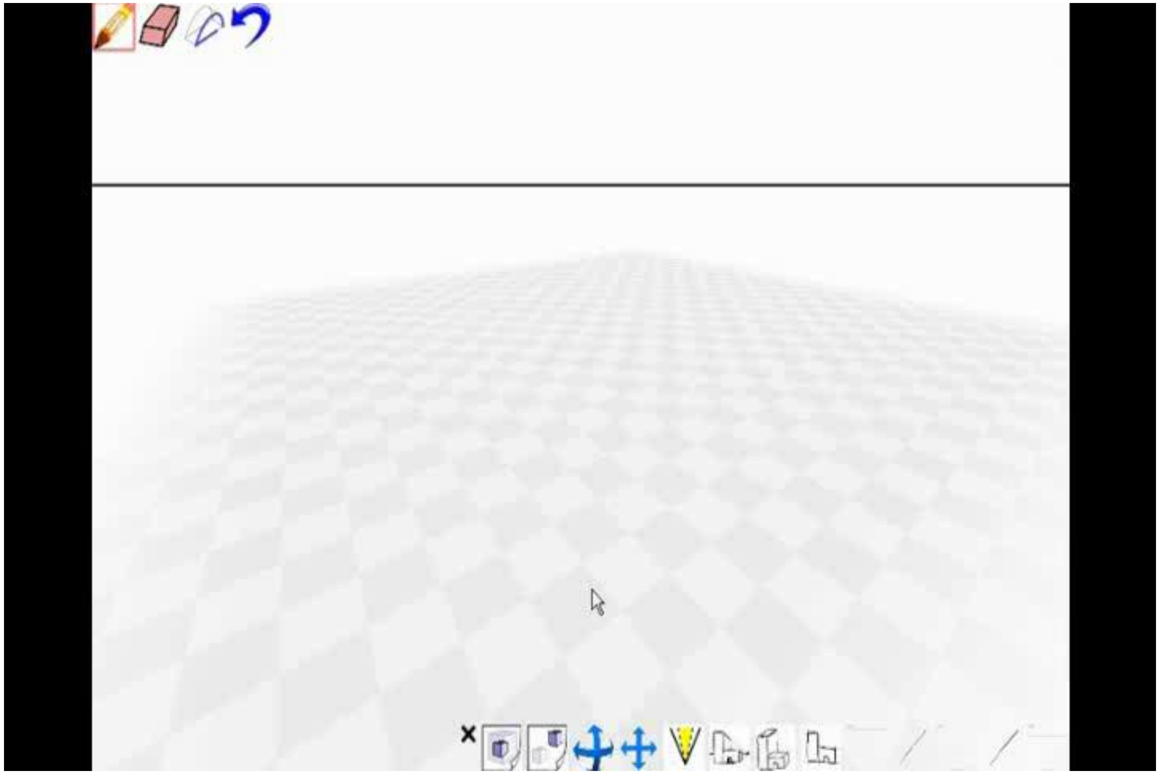






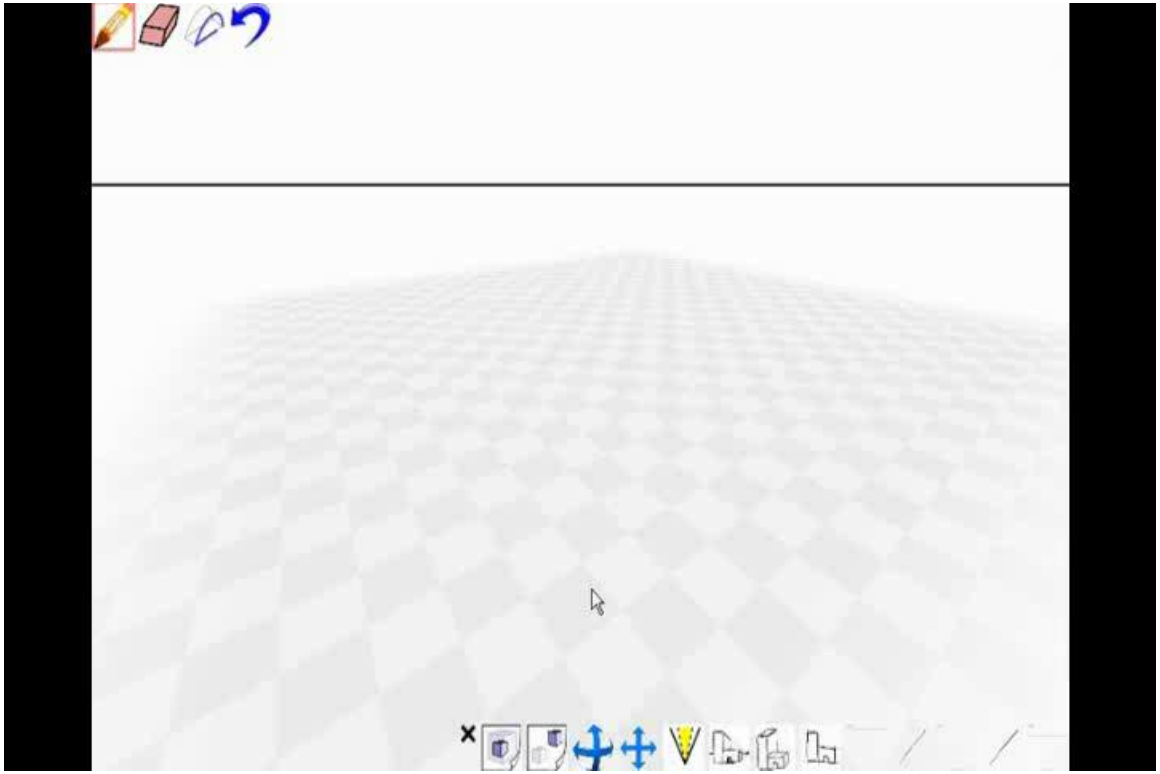


Analytic Drawing: single-view sketching



[**Schmidt, Khan, Singh, Kurtenbach,** Analytic drawing of 3D scaffolds. *SIGGRAPH Asia 2009*]
www.dgp.toronto.edu/~rms/pubs/DrawingSGA09.html

Analytic Drawing: single-view sketching

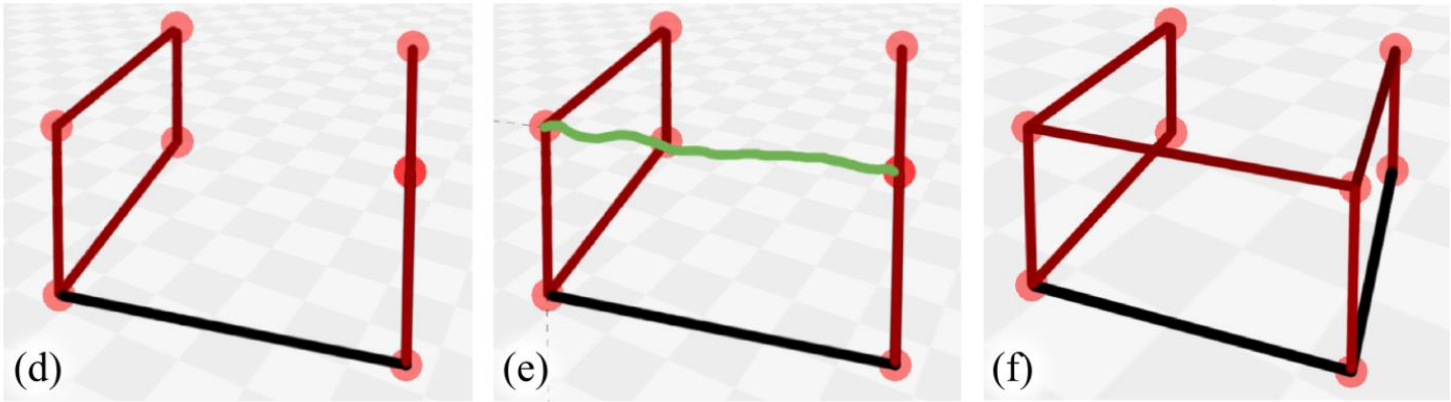


[**Schmidt, Khan, Singh, Kurtenbach,** Analytic drawing of 3D scaffolds. *SIGGRAPH Asia 2009*]
www.dgp.toronto.edu/~rms/pubs/DrawingSGA09.html

Analytic Drawing: inference

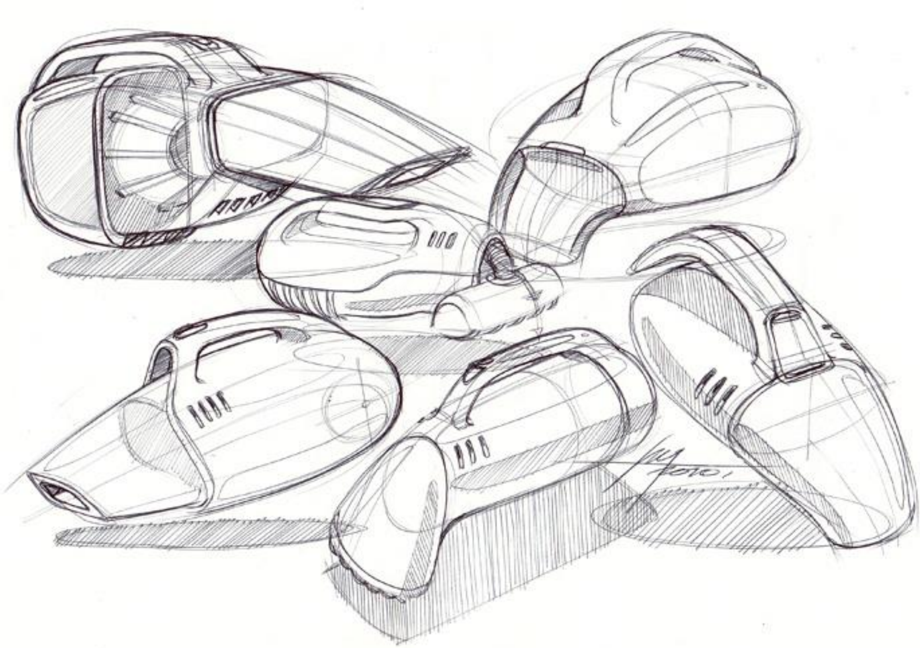
Scaffold constraints: position, direction, length.
Geometric priors: lines, circular-arcs.

Probabilistic model: **redundancy** resolves ambiguity.



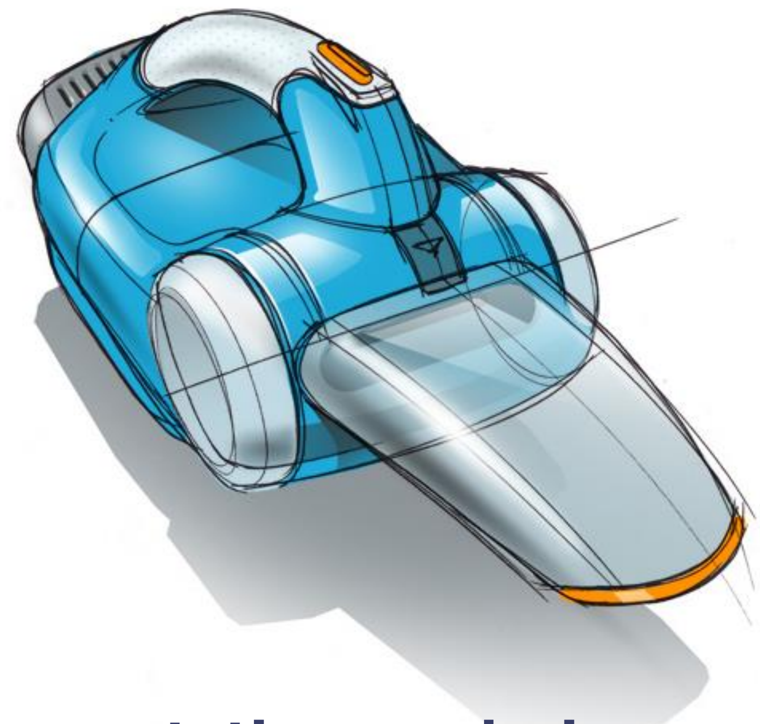
CrossShade/True2Form: design sketches

©www.sketch-a-day.com



concept sketch

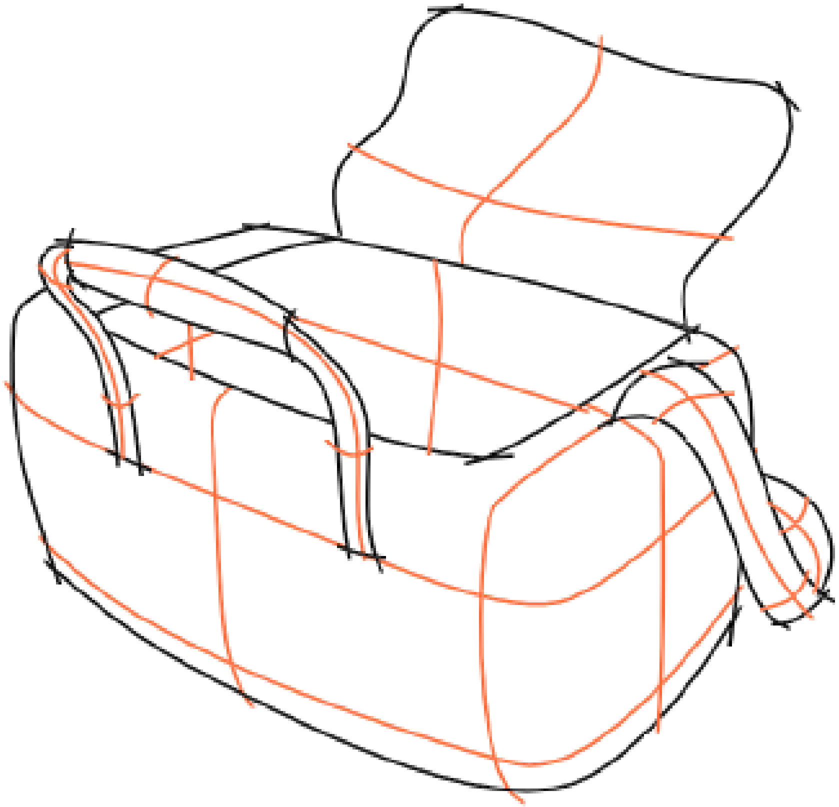
- Explore form.
- Fast to draw.



presentation rendering

- Communicate with clients.
- Tedious, repetitive, painting.

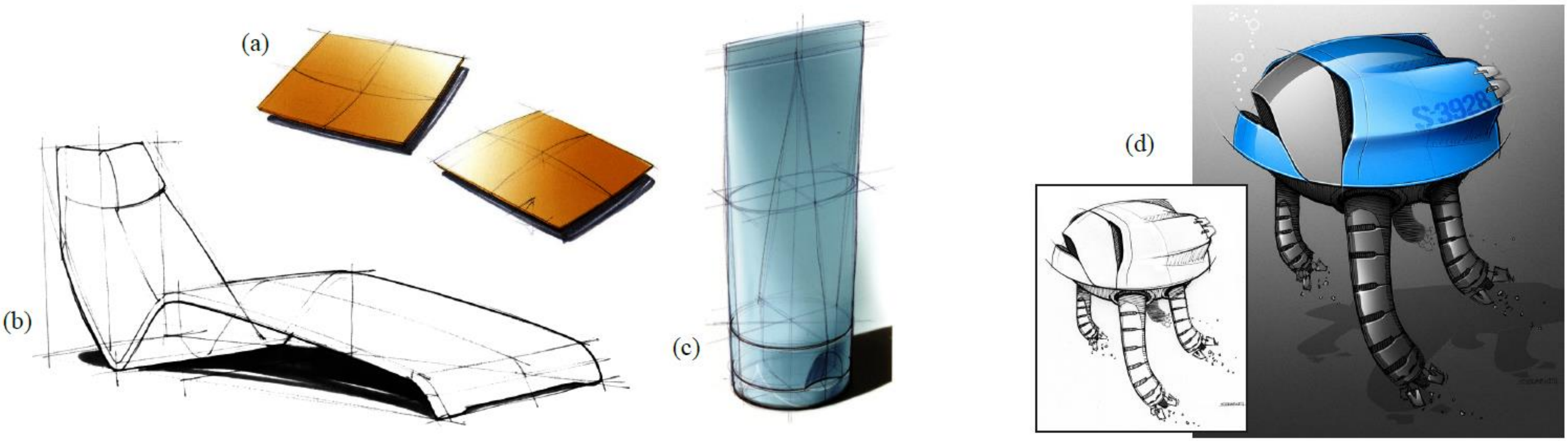
CrossShade: 3D presentation rendering



CrossShade: 3D presentation rendering

[**Shao, Bousseau, Sheffer, Singh,** CrossShade: Shading Concept Sketches Using Cross-Section Curves *SIGGRAPH 2012*] http://www.crossshade.com/

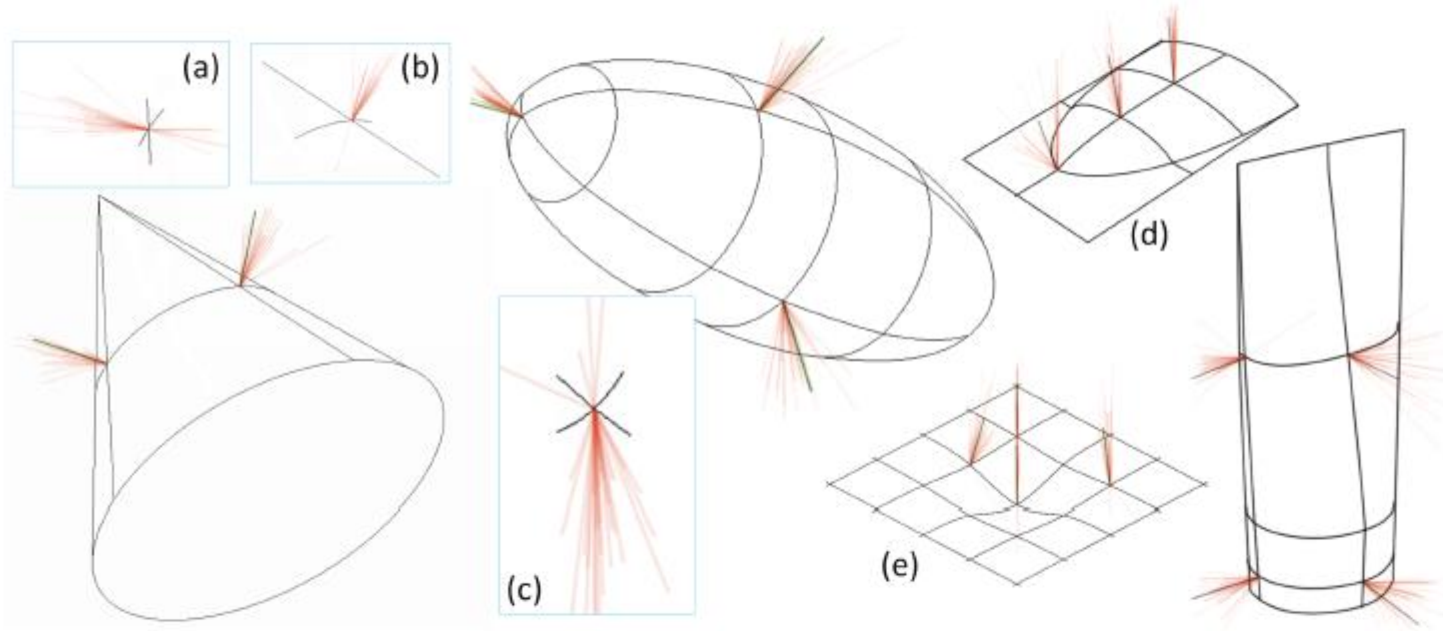
CrossShade: design analysis



"Cross-sections on a surface explain or emphasize its curvature."

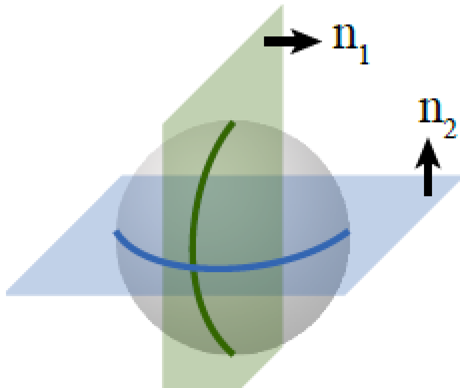
"…bend or transform the object's surface."

CrossShade: perceptual study

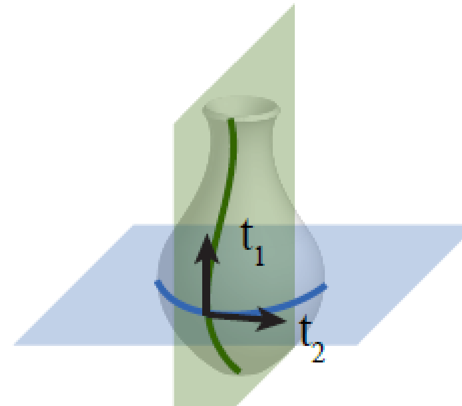


Viewers are *persistent*, *consistent* and *accurate* in X-hair perception.

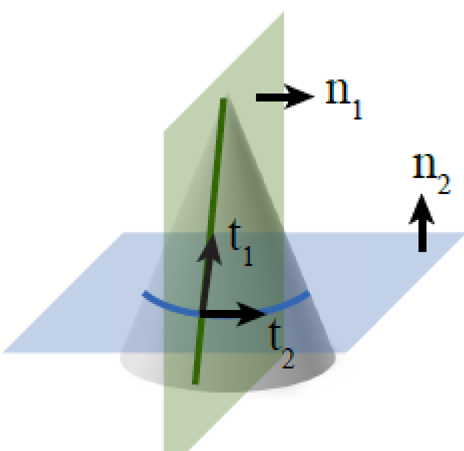
CrossShade: defining cross-hairs



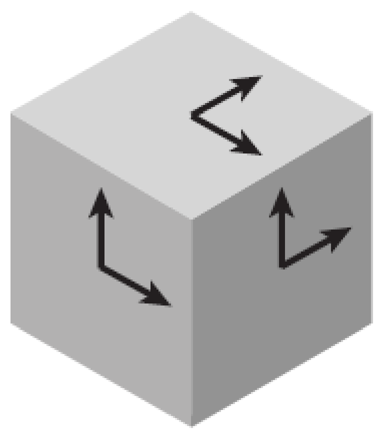
Plane Orthogonality



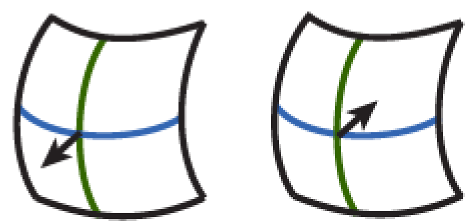
Local curvature lines



Local Geodesics

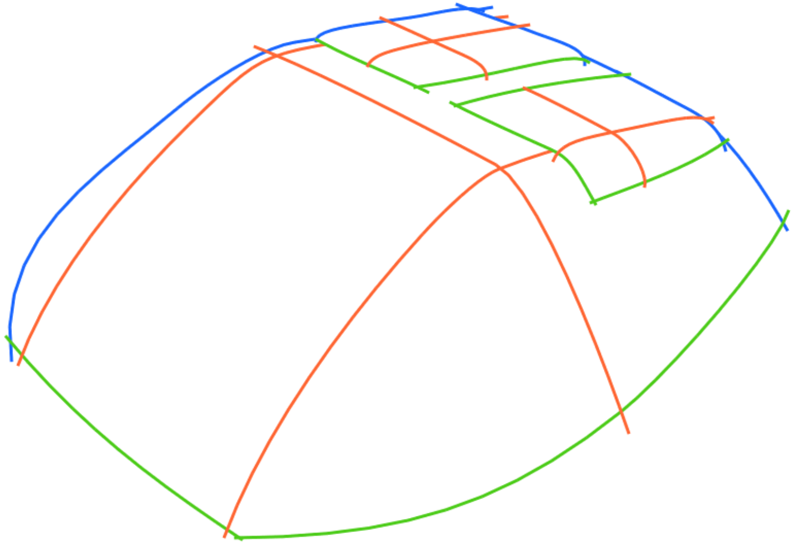


Minimal Foreshortening



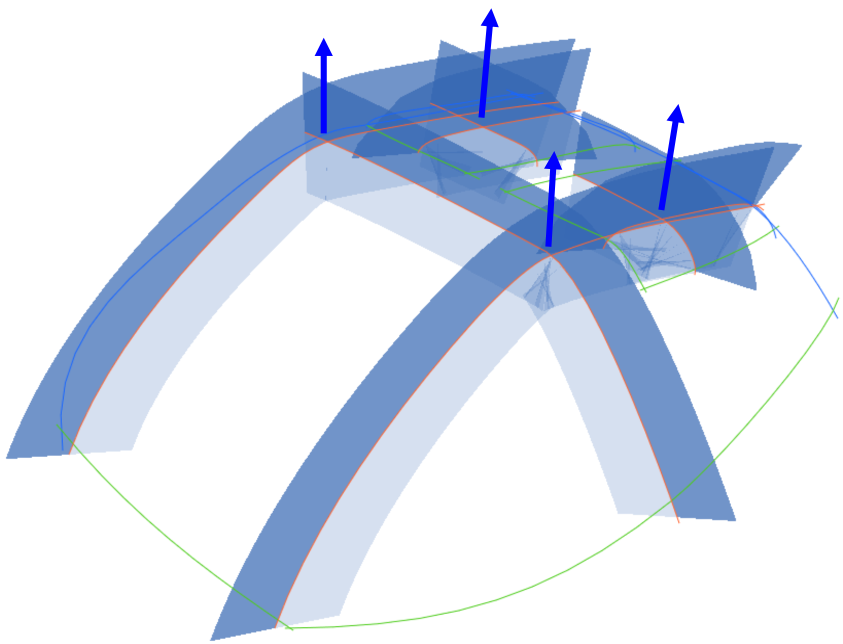
Orientation

CrossShade: Algorithm



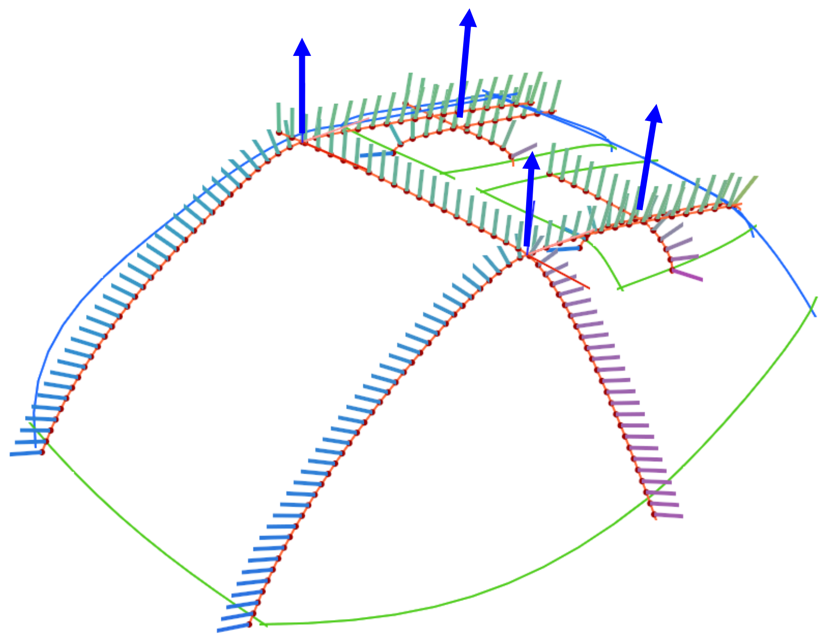
CrossShade: Algorithm

- Compute X-section planes, X-hair normals: *use 5 properties.*



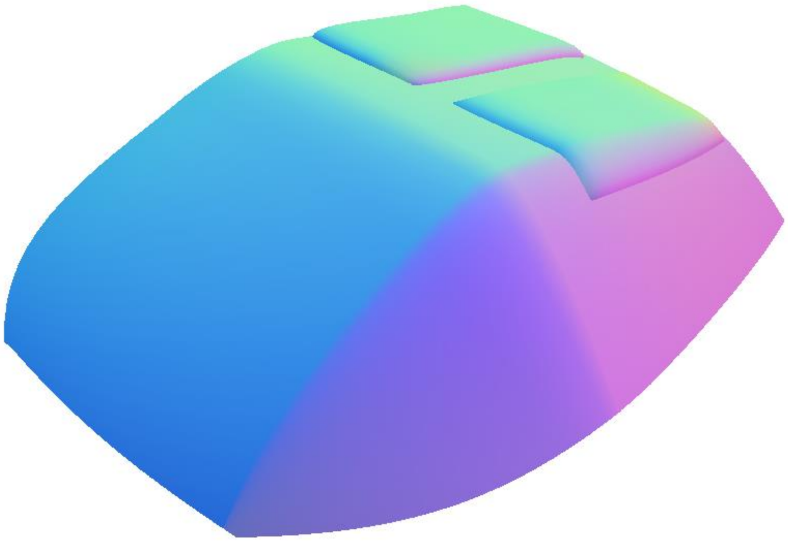
CrossShade: Algorithm

- Compute X-section planes, X-hair normals: *use 5 properties.*
- Propagate normals along X-section curves: *minimize twist.*



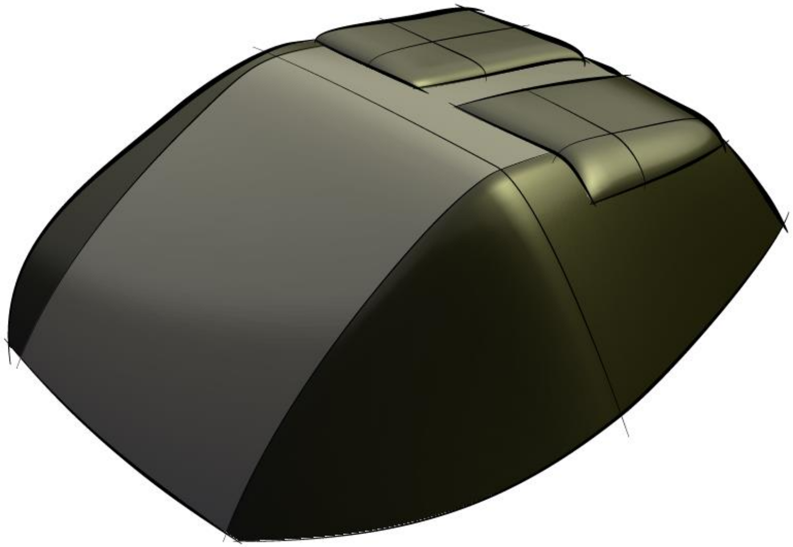
CrossShade: Algorithm

- Compute X-section planes, X-hair normals: *use 5 properties.*
- Propagate normals along X-section curves: *minimize twist.*
- Propagate normals into interior regions: *Coons interpolation.*

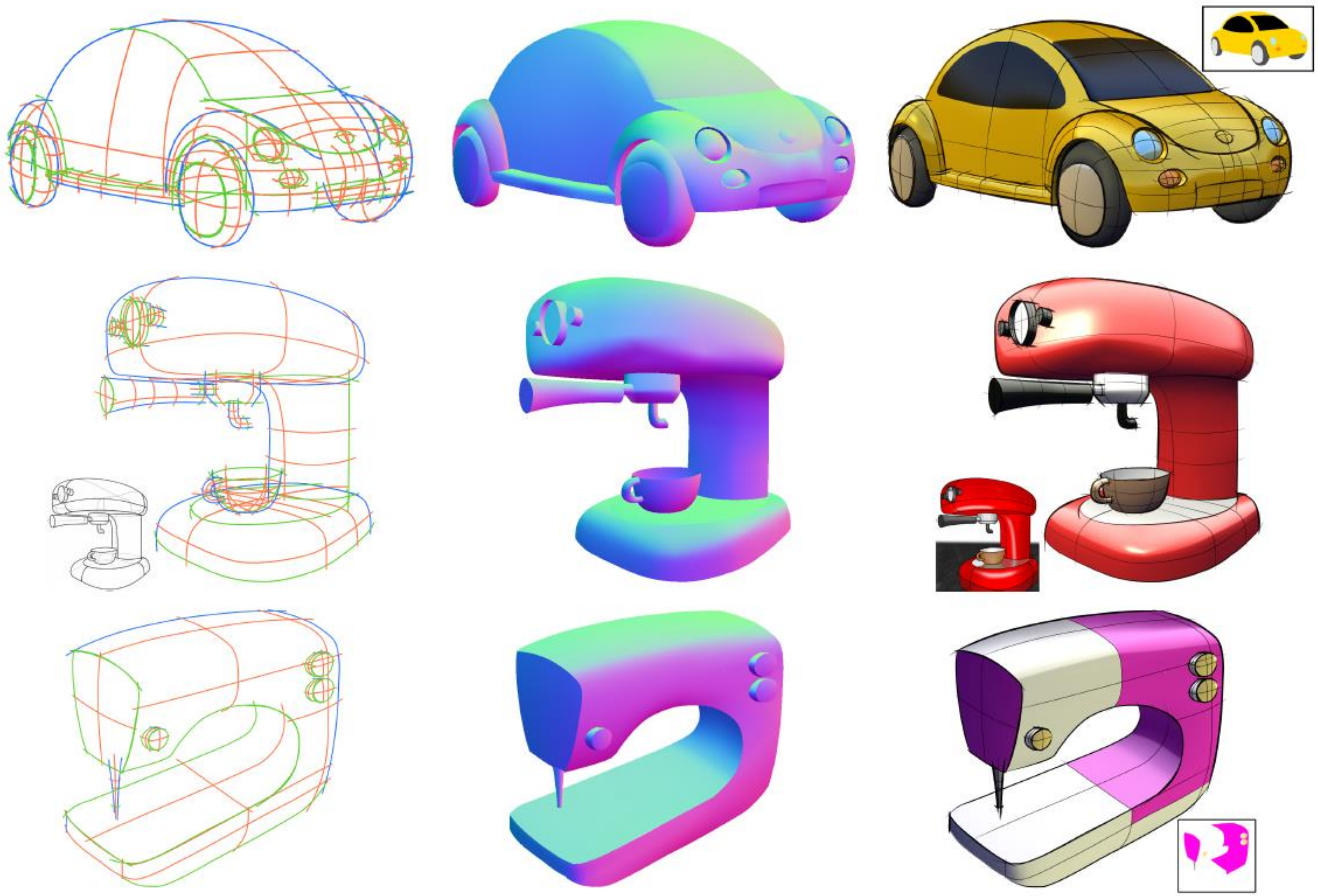


CrossShade: Algorithm

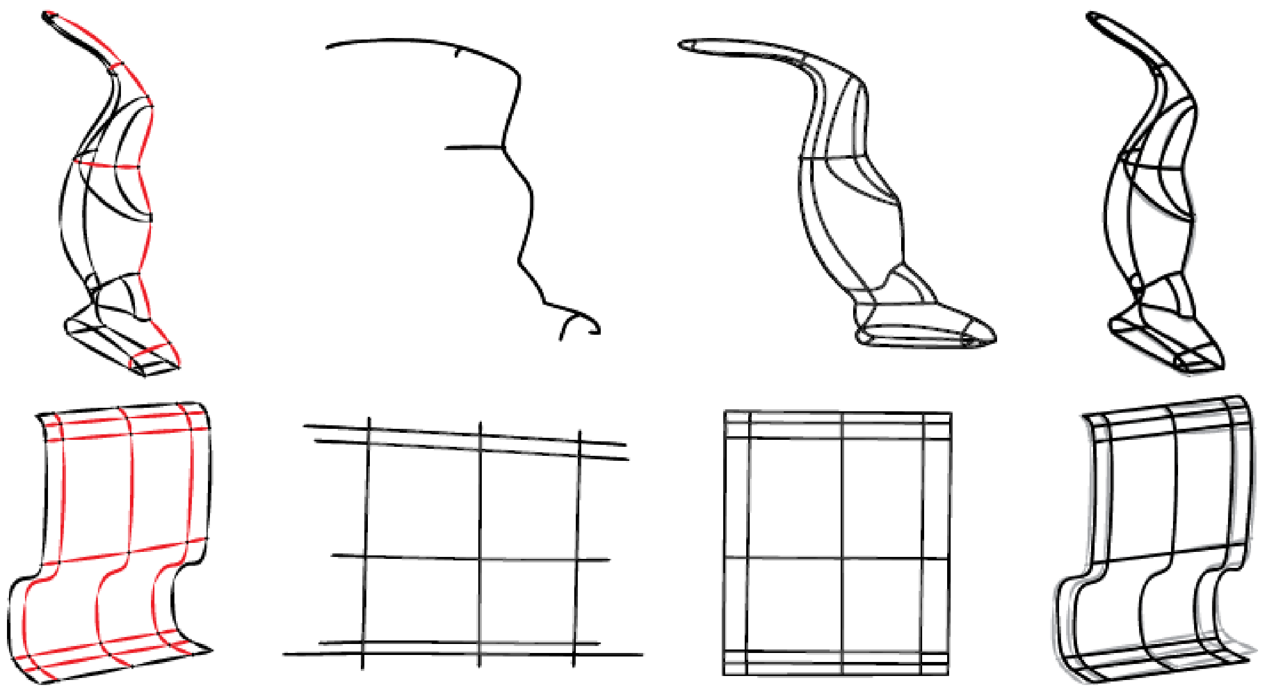
- Compute X-section planes, X-hair normals: *use 5 properties.*
- Propagate normals along X-section curves: *minimize twist.*
- Propagate normals into interior regions: *Coons interpolation.*
- Shade!



CrossShade: Results



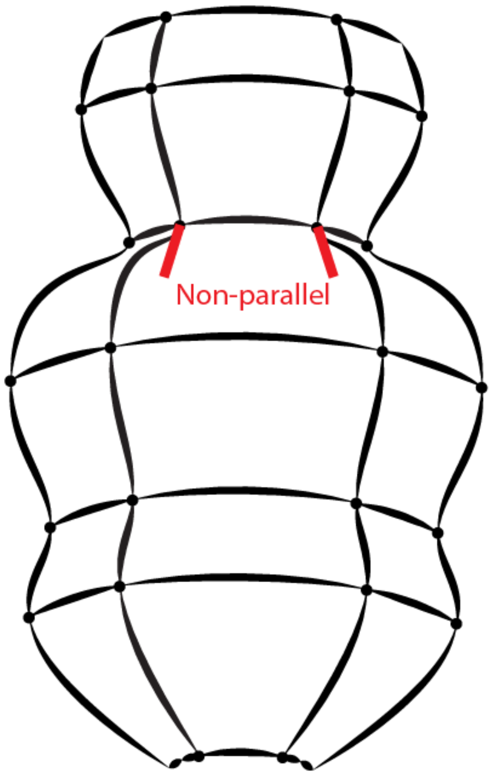
True2Form: 3D ~~rendering~~ modeling



2D sketch | Crossshade *side view* 3D | True2Form *side view* 3D | True2Form *input overlay*

[**Xu, Chang, Bousseau, McCrae, Sheffer, Singh,** True2Form: 3D curve networks from 2D sketches via selective regularization. *SIGGRAPH 2014*]

T2F: *descriptive* curve properties => 3D



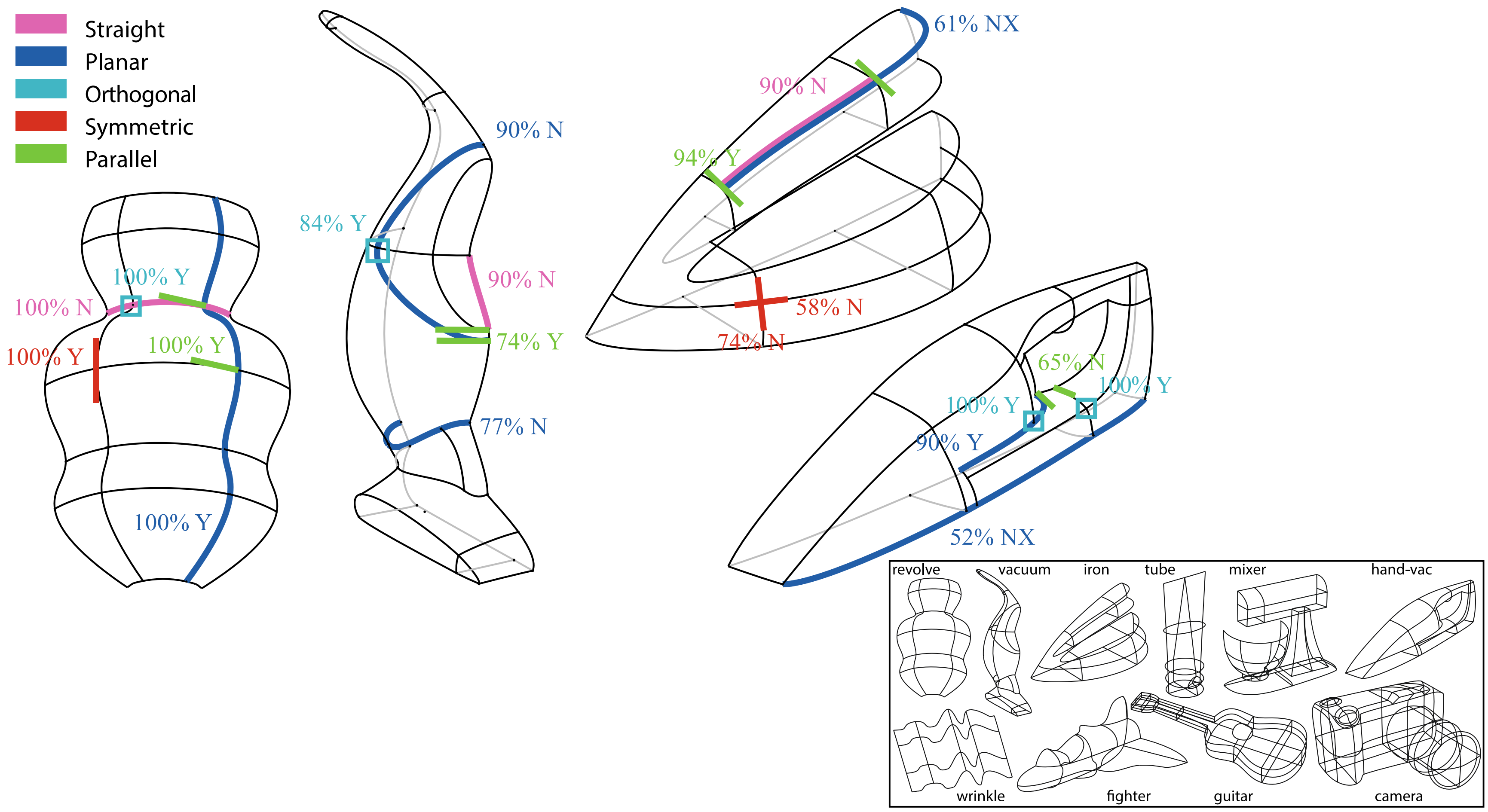
Fidelity: sketches reflect 3D geometry.

- Satisfied by flat interpretation.

Regularity: curves often capture 3D constraints.

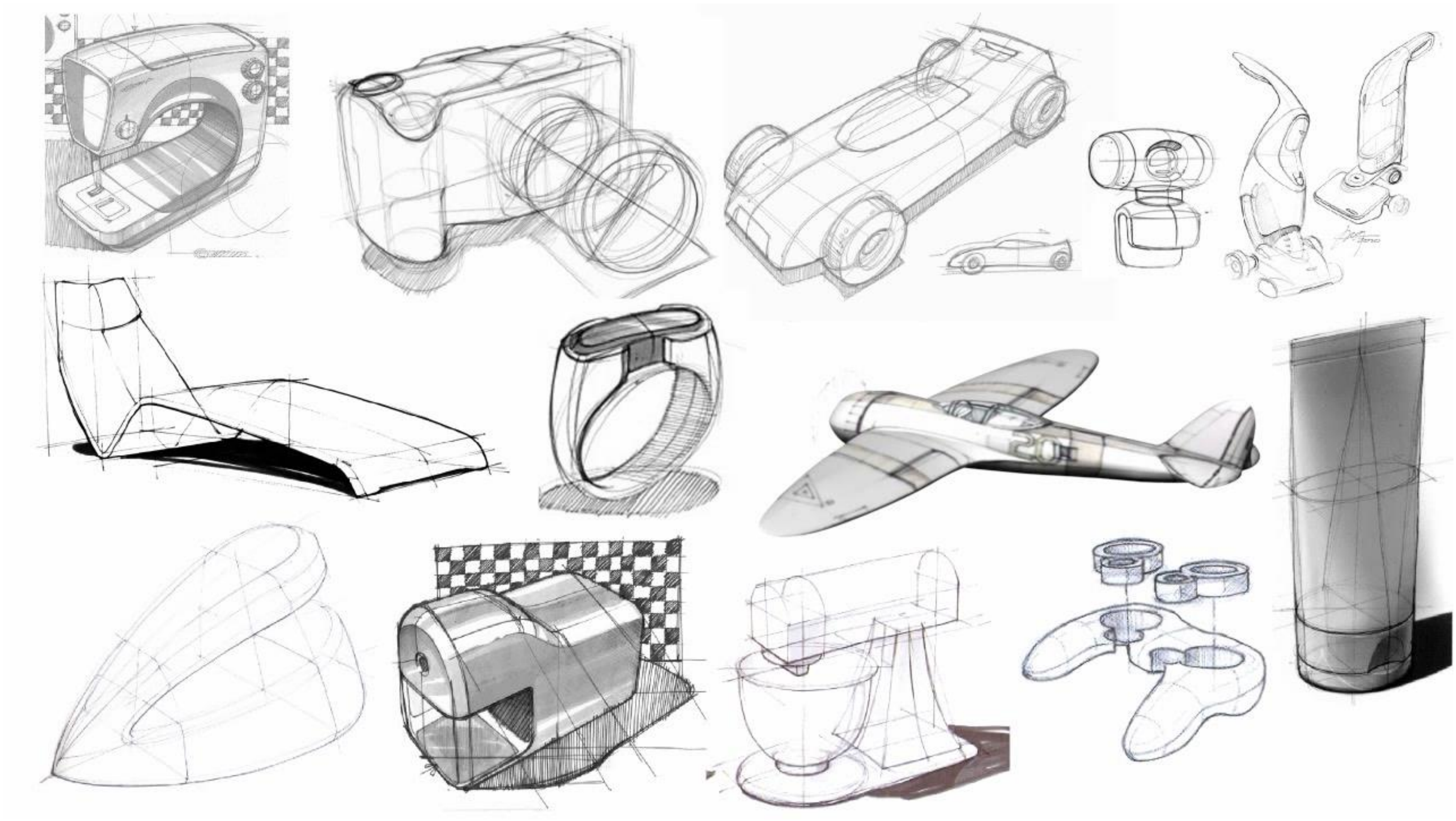
- This lifts curves to 3D.
- Regularity is **context-based.**

True2Form: perceptual validation

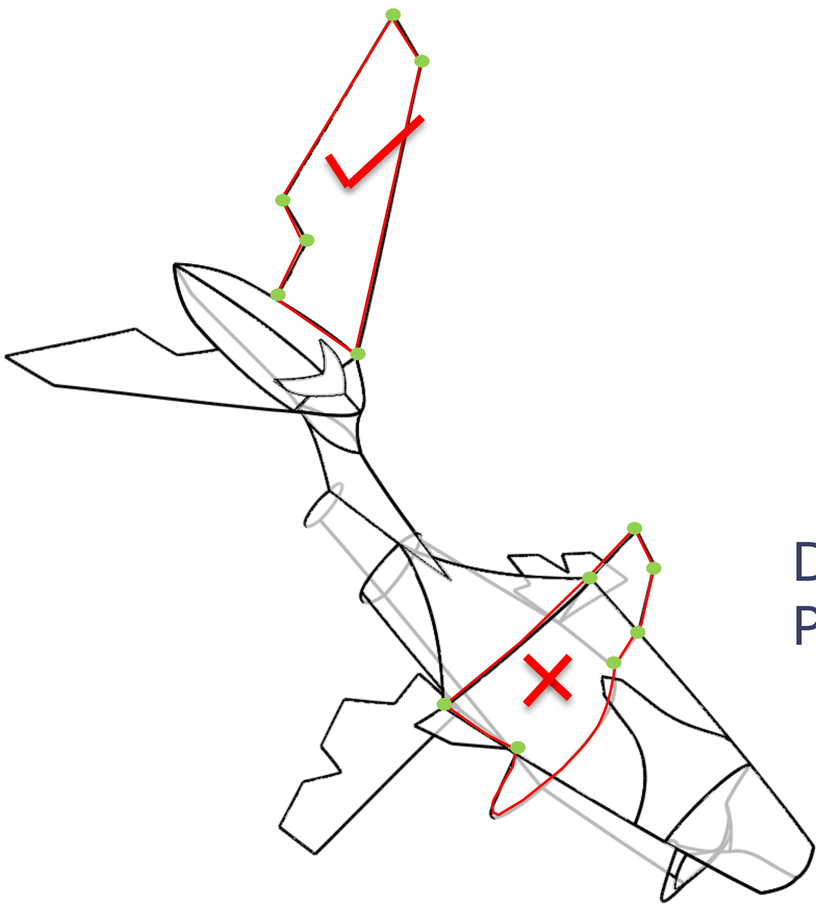


Viewers *consistently* perceive 3D parallelism, symmetry, orthogonality, linearity and planarity cues in 2D sketches.

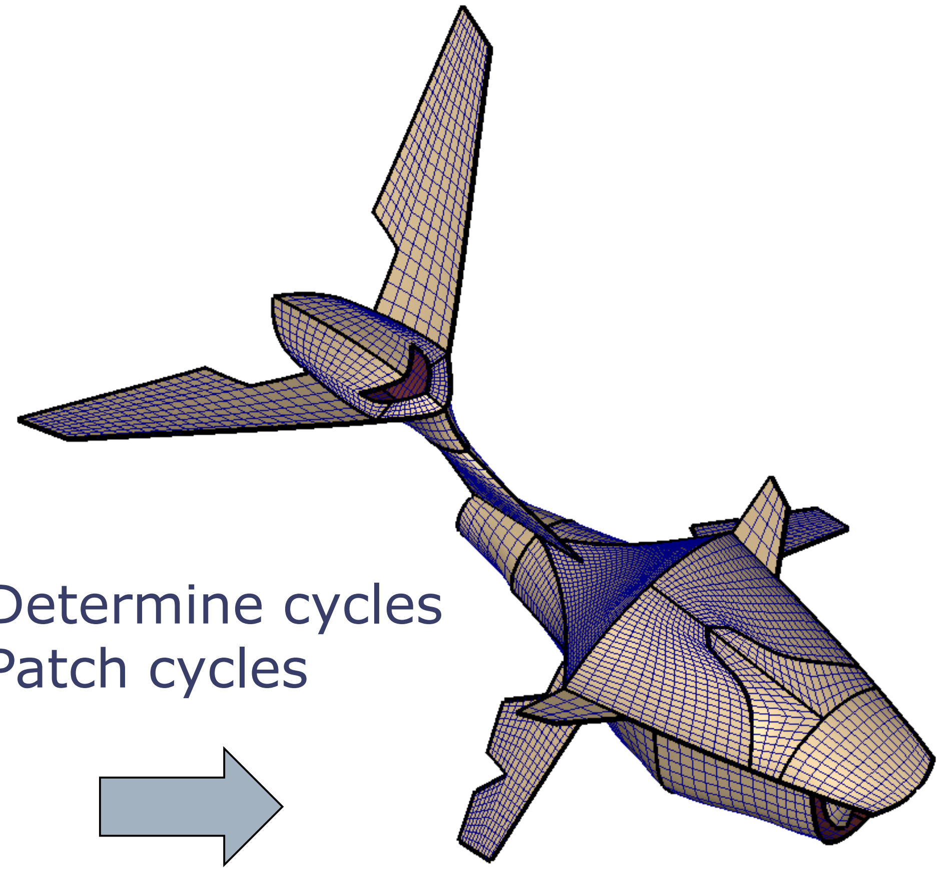
True2Form: fidelity and regularity



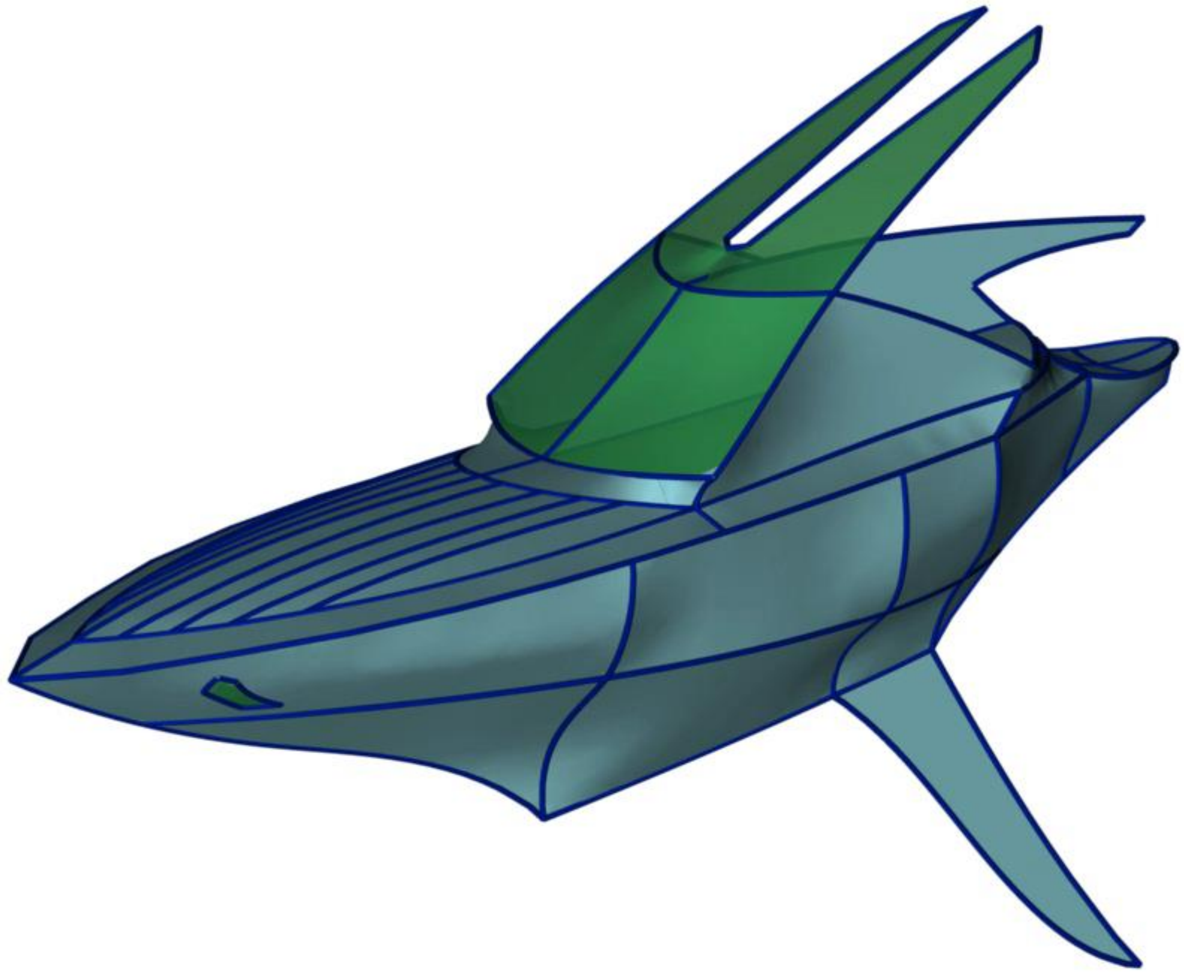
Curve network surfacing



Determine cycles
Patch cycles

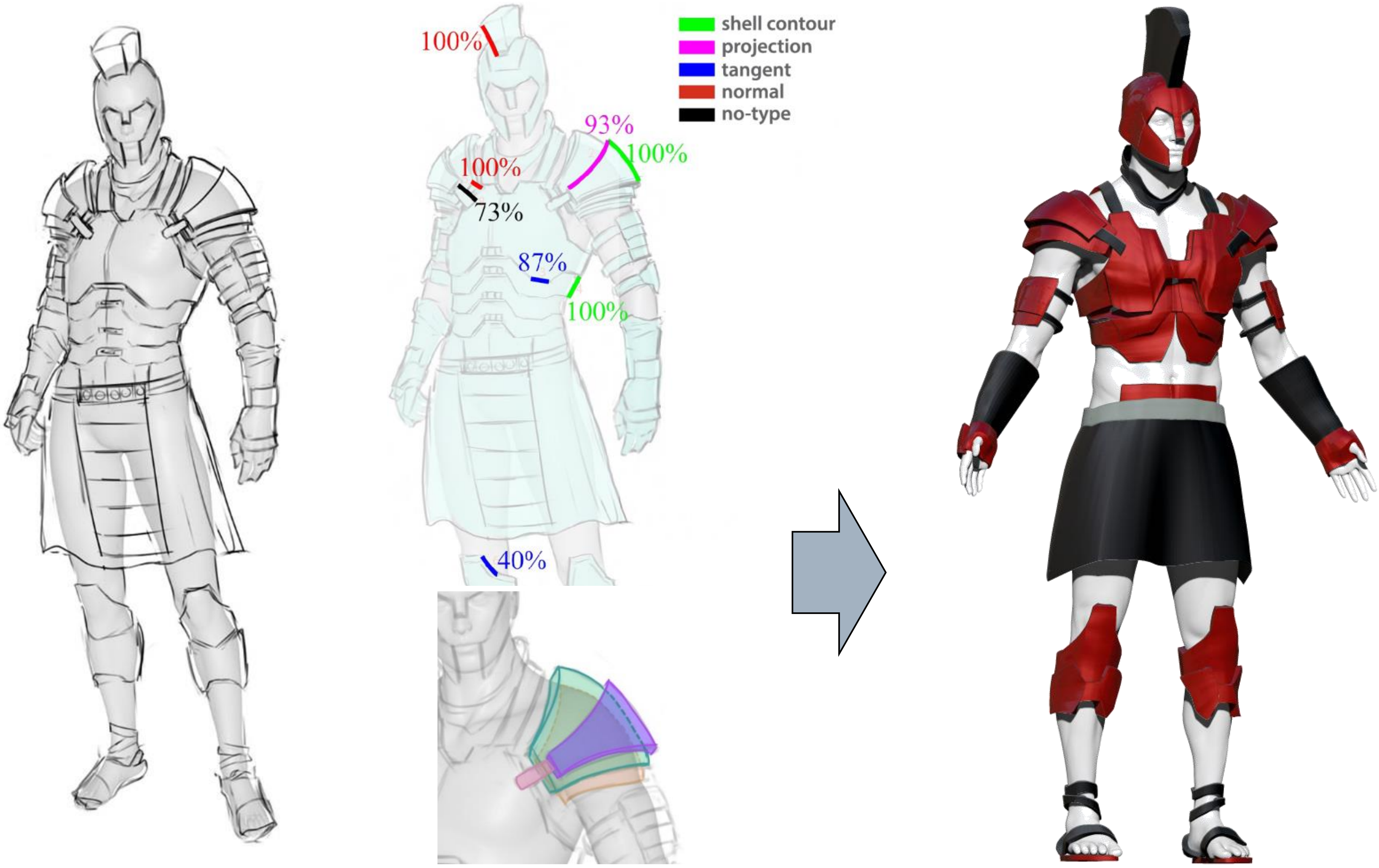


[**Bessmeltsev, Wang, Sheffer, Singh**, Design-Driven Quadrangulation of Closed 3D Curves. *SIGGRAPH Asia 2012*]



[**Sadri & Singh**, Flow Complex based shape reconstruction from 3D curves. ACM TOG, *2014*]

SecondSkin: layered 3D modeling



[**DePaoli & Singh**, SecondSkin: Sketch-based Construction of Layered 3D Models. *SIGGRAPH, 2015*]

SecondSkin: layered 3D modeling

Sketch-based modeling ingredients

System	Fidelity	Geom. priors	**Perceived Regularity**		View
Teddy/FM	Planarity	Fixed depth	Project on mesh		Multi-view
ILoveSketch		Cubic Bezier	Symmetry		Multi-view
Analytic 3D	Proj. accuracy	Length,direction, circular arcs	Scaffold		Single-view
True2Form	Proj. accuracy, Min. variation		Orthogonal X-hairs etc.		Single-view
SecondSkin	Foreshortening	Cubic Bezier	Underlying geom. & curves		Multi-view

Key Messages

- Centuries of visual experience captured in artistic practice.
- Perceived regularity.
- Techniques based on artistic and perceptual insights, and leaving user ultimate creative control.
- Better tools = Better VIDEO OUT
  Better tools != Better content