Computer Animation

Lesson 3 - Animation curves

Remi Ronfard, Nov 2019

Animation curves

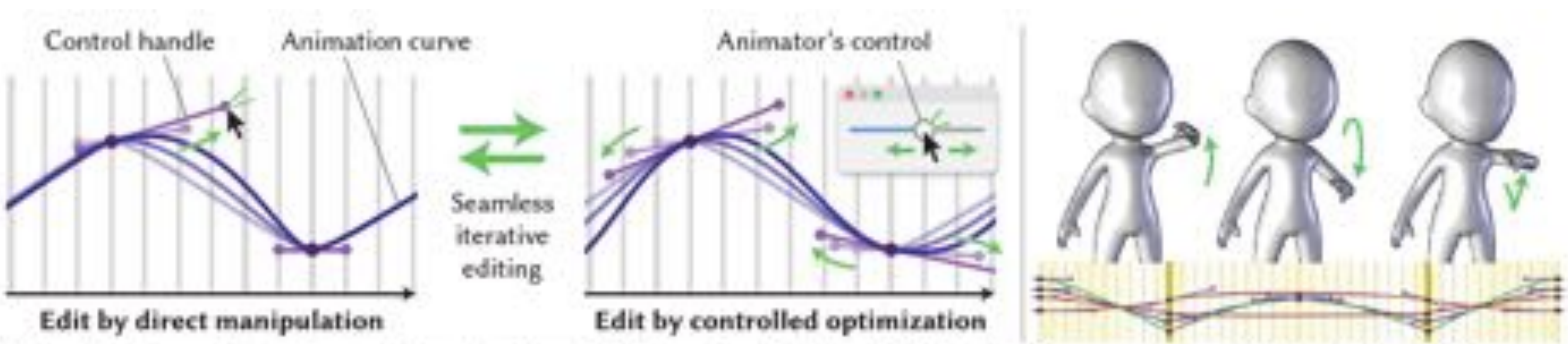
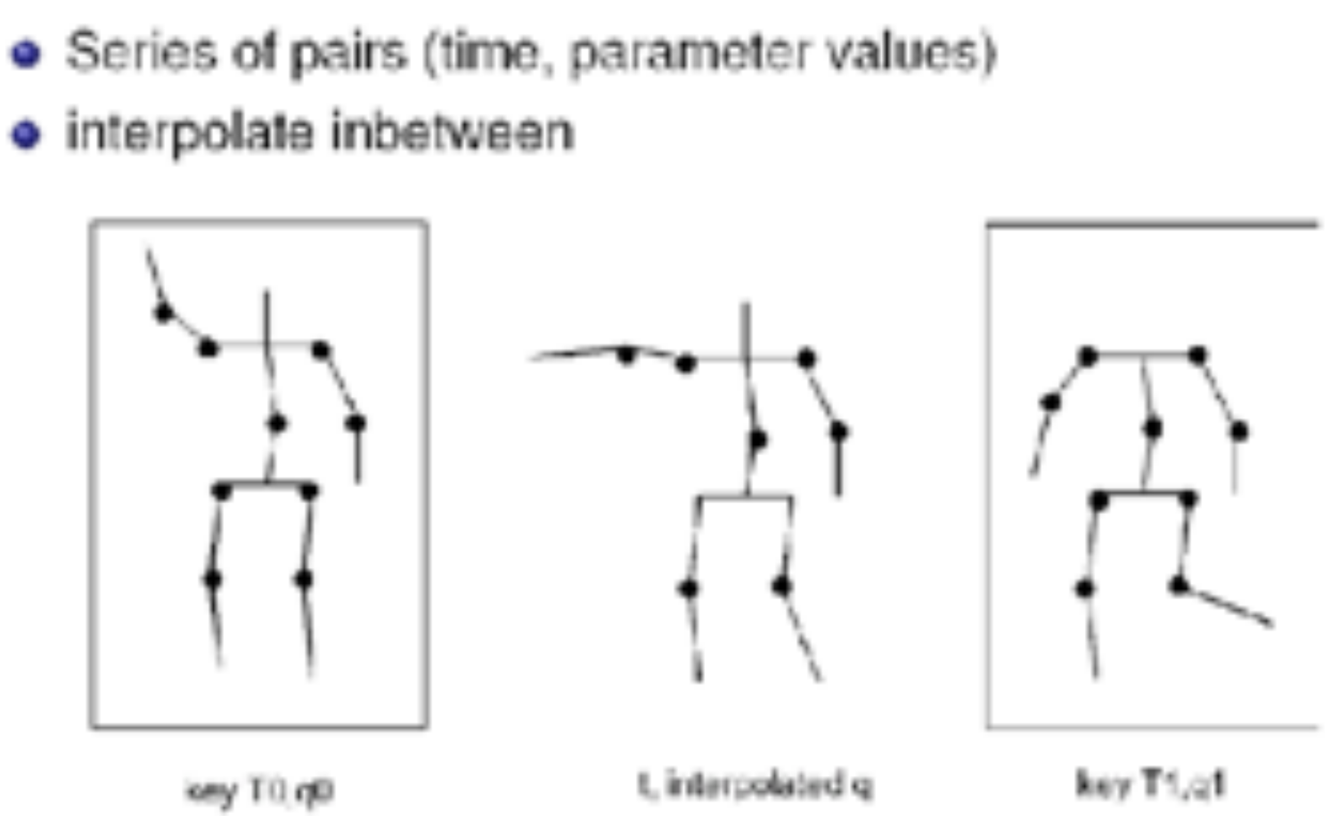


Figure 1. We present a new framework for animators to edit character motions by effectively using the power of numerical optimization. (Left) Concept of the framework. As well as direct manipulation, animators can use controlled optimization to efficiently edit animation curves in the iterative editing process. (Right) Example of edited motions using our proof-of-concept system, named *OptiMo*.

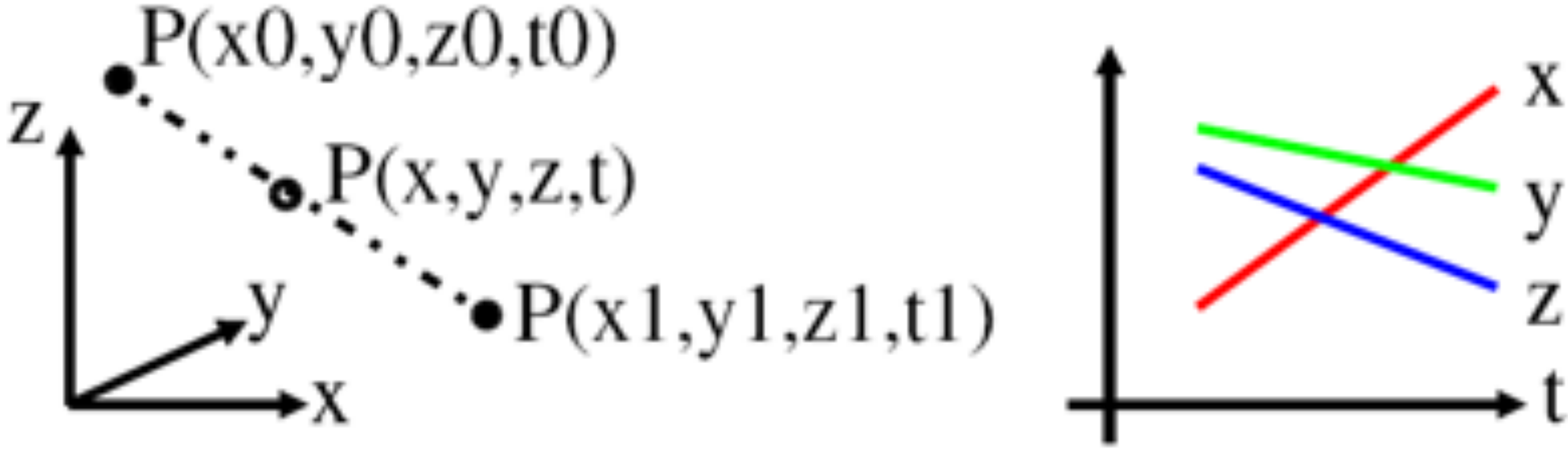
Source: Koyama and Goto. OptiMo: Optimization-Guided Motion Editing for Keyframe Character Animation. CHI 2018.

Animation and interpolation

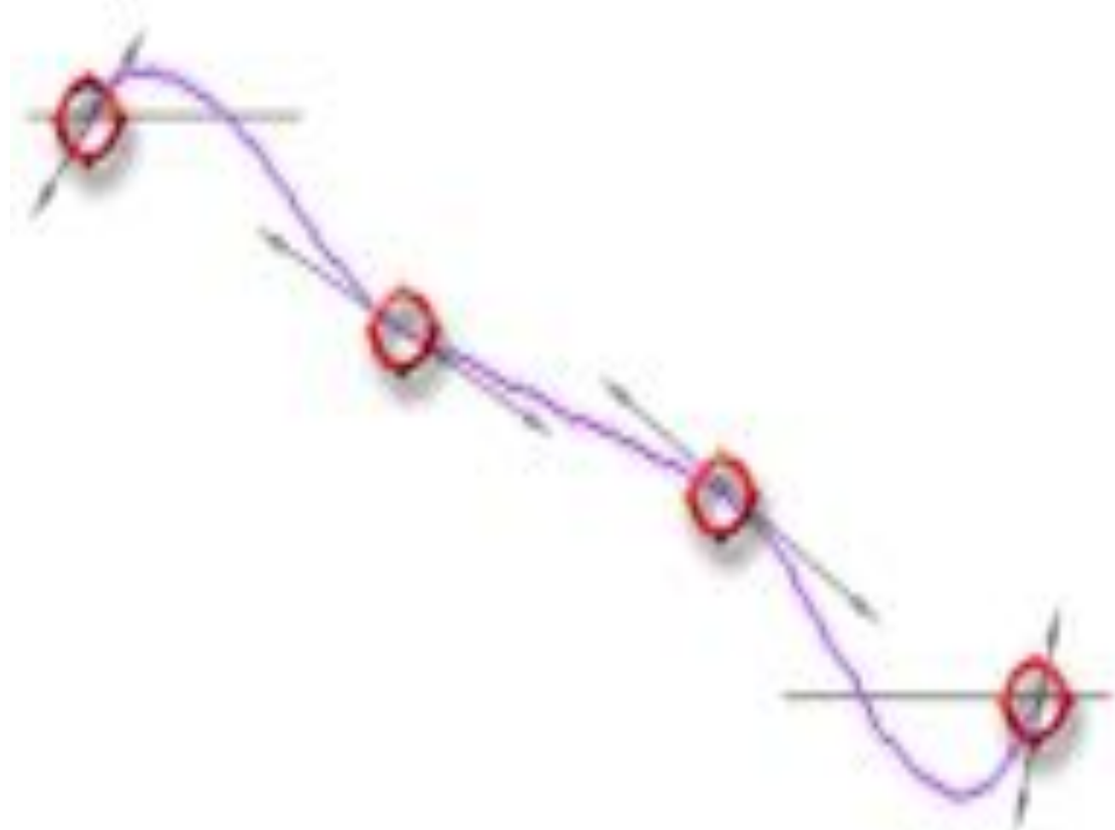


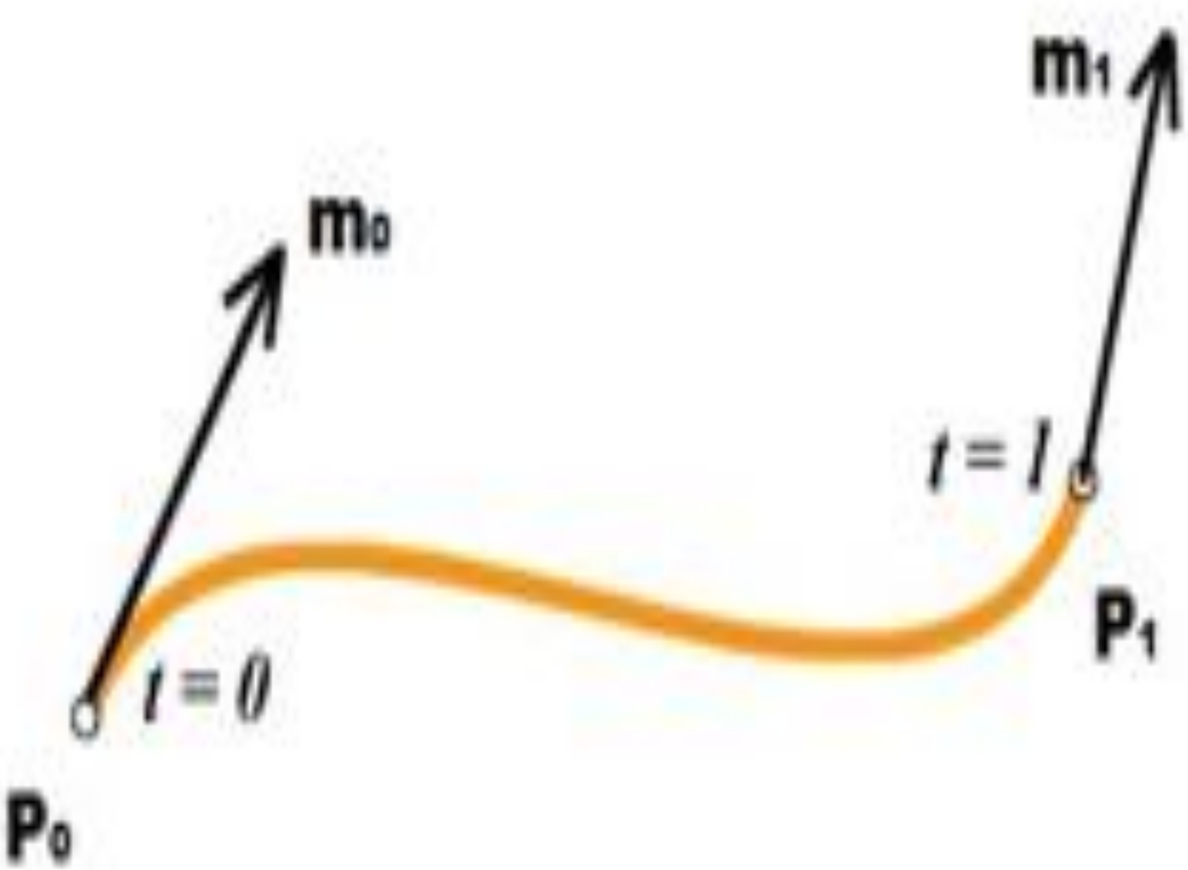
- Keyframe animation
- How to interpolate motion between key-frames

Interpolation of translation



Hermite splines





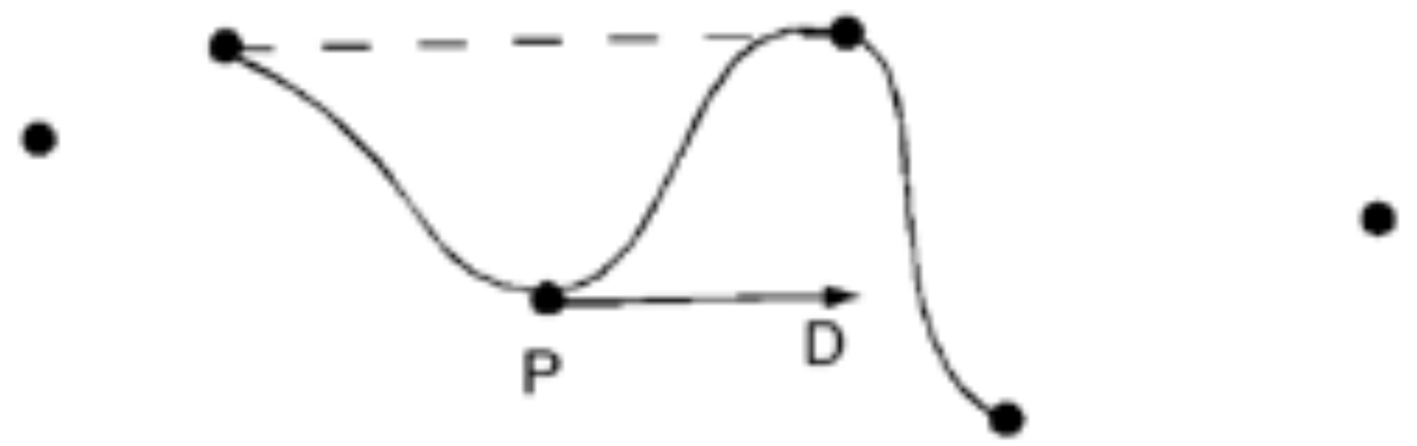
Hermite polynomials

$$\begin{cases} x_0 &= a_0 \\ \dot{x}_0 &= a_1 \\ x_1 &= a_0 + a_1 + a_2 + a_3 \\ \dot{x}_1 &= a_1 + 2a_2 + 3a_3 \end{cases} \Longrightarrow \begin{cases} a_0 &= x_0 \\ a_1 &= \dot{x}_0 \\ a_2 &= 3(x_1 - x_0) - 2\dot{x}_0 - \dot{x}_1 \\ a_3 &= 2(x_0 - x_1) + \dot{x}_0 + \dot{x}_1 \end{cases}$$

$$x(t) = \begin{pmatrix} & (1 - 3t^2 + 2t^3) & x_0 \\ + & (3t^2 - 2t^3) & x_1 \\ + & (t - 2t^2 + t^3) & \dot{x}_0 \\ + & (-t^2 + t^3) & \dot{x}_1 \end{pmatrix} = \begin{pmatrix} & h_0(t) & x_0 \\ + & h_1(t) & x_1 \\ + & h_0'(t) & \dot{x}_0 \\ + & h_1'(t) & \dot{x}_1 \end{pmatrix}$$

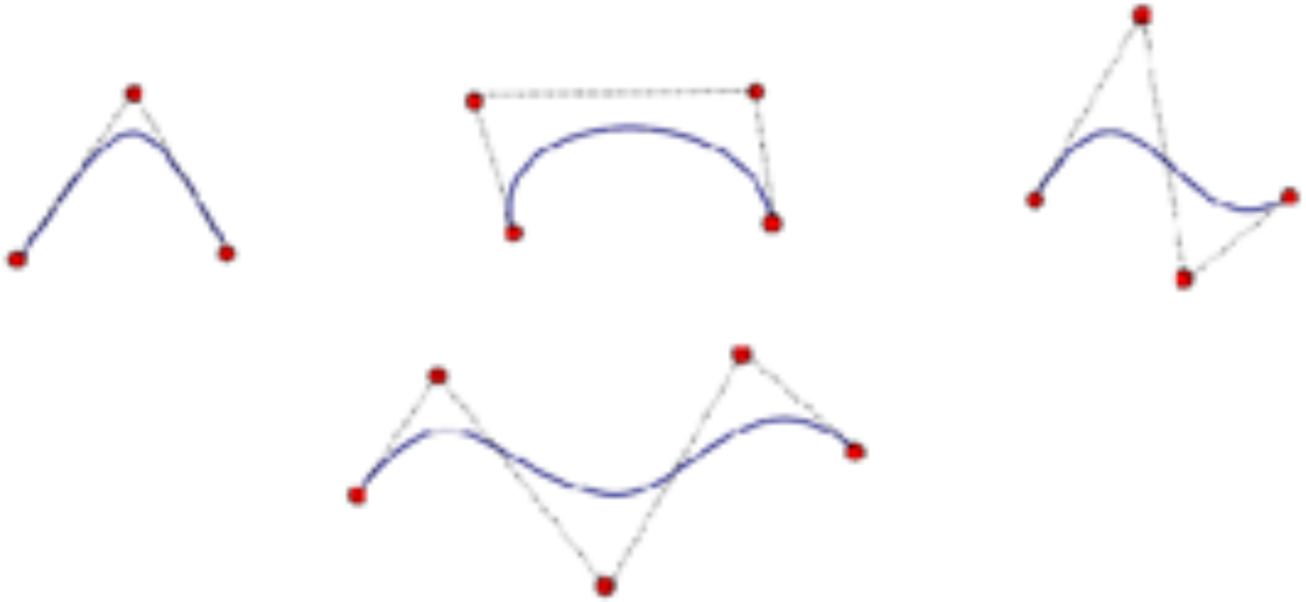
Catmull-Rom splines

- Derived from Hermite splines
- Approximate tangents using control points
  $D_i = \frac{1}{2}(P_{i+1} - P_{i-1})$
- Arbitrary first and last points



Bezier curves, B-splines and NURBS

- Bezier: Piece-wise polynomials with tangent continuity
- B-splines: control points, arcs and curves
- NURBS: piece-wise rational curves, i.e. projective splines in projective coordinates



Bezier curves, B-splines and NURBS

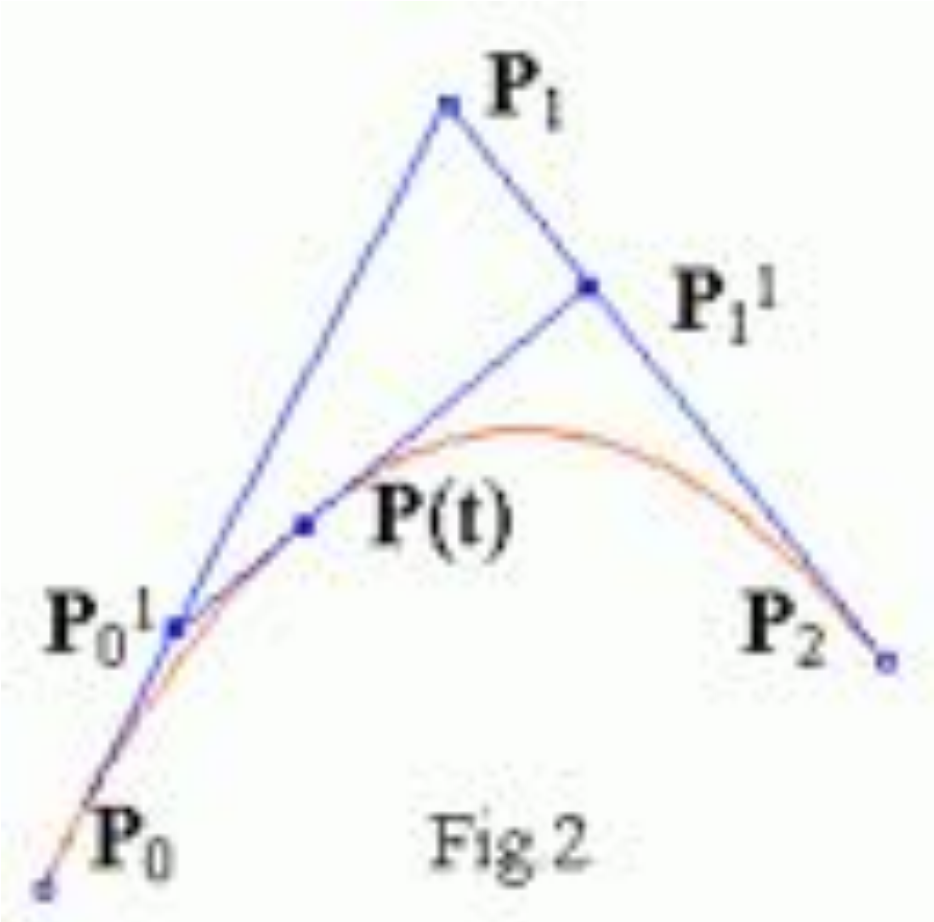


Fig.2

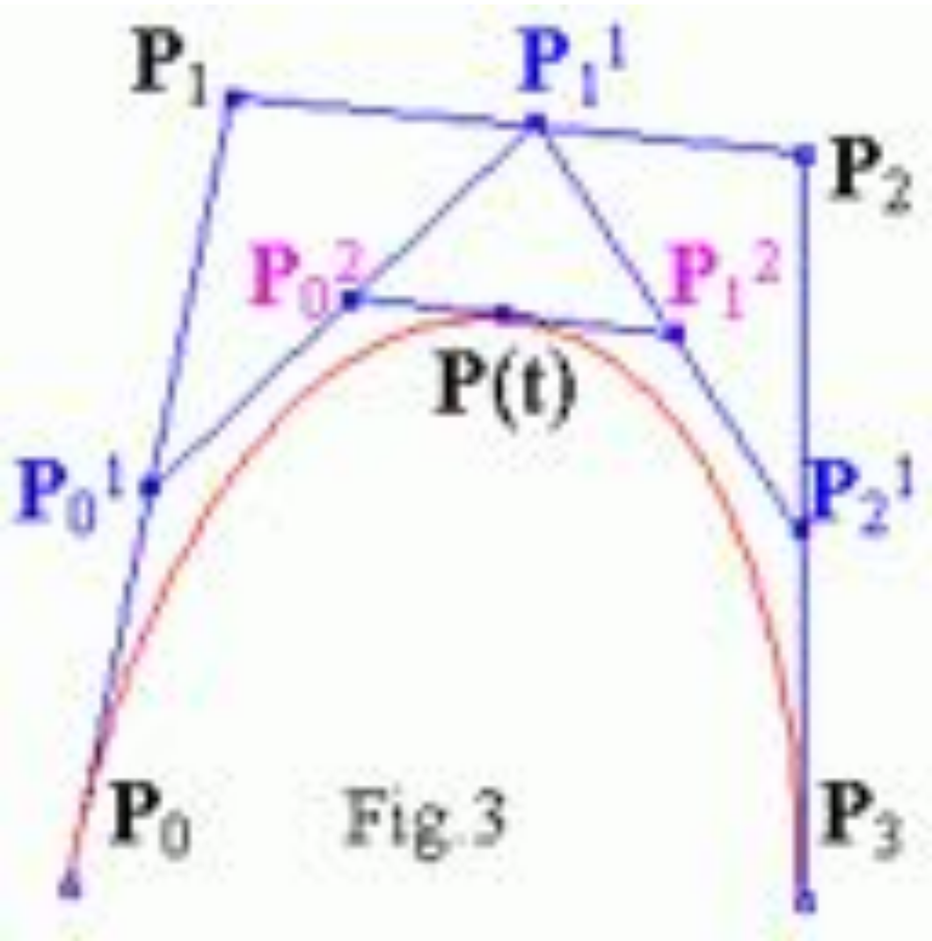
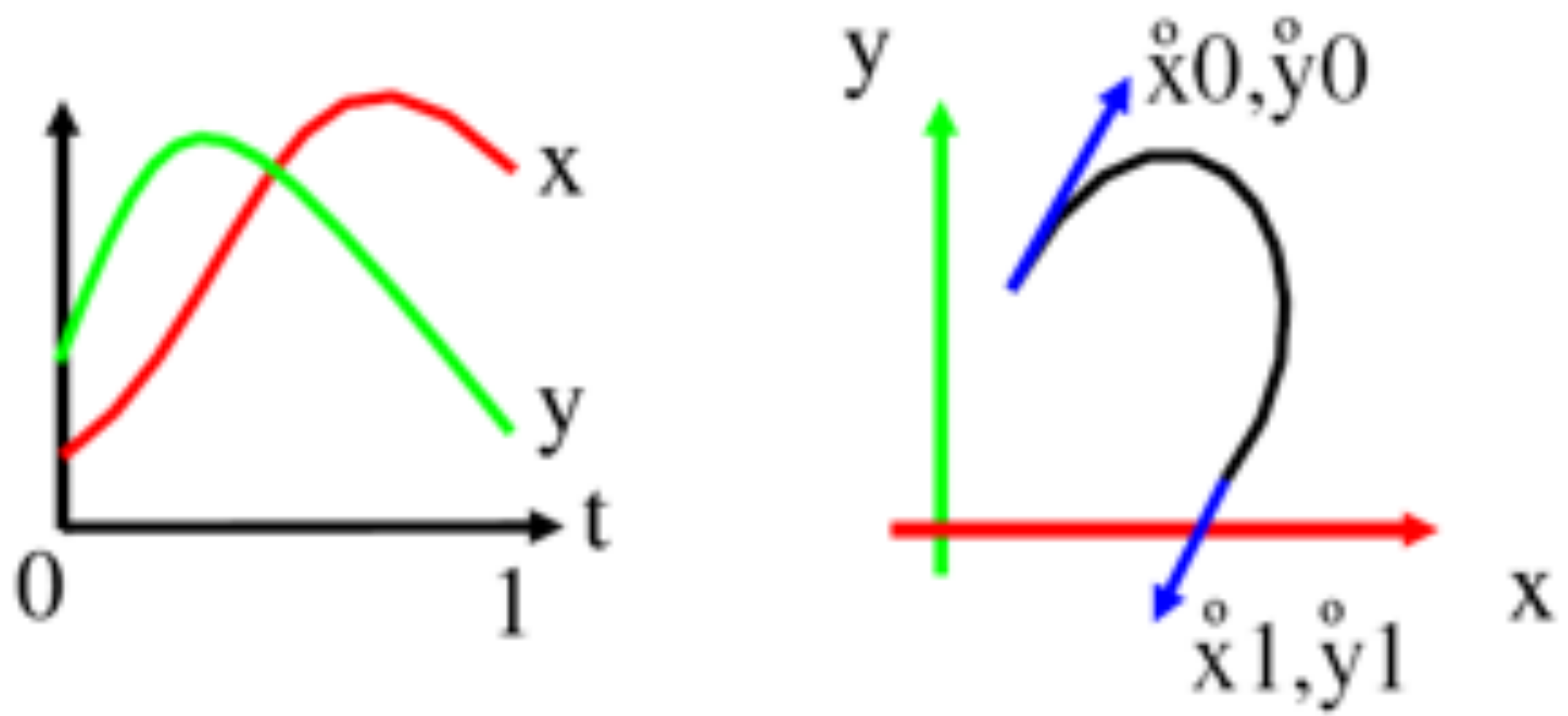


Fig.3

Three-dimensional interpolation



$$\frac{1}{2}(P_{i+1} - P_{i-1})$$

Interpolation of rotations

- Cannot interpolate matrices
- Compute $R = R_1^T R_2$
- Compute axis and angle
- Interpolate angles

$$
\begin{aligned}
\theta &= \arccos\left(\frac{R_{11} + R_{22} + R_{33} - 1}{2}\right) \\
s &= 2\frac{\sin(\theta)}{\theta} \\
n_x &= \frac{R_{32} - R_{23}}{s} \\
n_y &= \frac{R_{13} - R_{31}}{s} \\
n_z &= \frac{R_{21} - R_{12}}{s}
\end{aligned}
$$

Quaternion interpolation

$$q = [w, v], v = (x, y, z), w = scalar$$

Arbitrary axis   Angle of rotation

q form a sphere of unit length in the 4D space

q1 * q2 = (w1.w2 - v1.v2, w1.v2 + w2.v1 + v1 X v2)

Quaternion interpolation

1. Use a quaternion to represent the rotation.
2. Generate a temporary quaternion for the change from the current orientation to the new orientation.
3. PostMultiply the temp quaternion with the original quaternion. This results in a new orientation that combines both rotations.
4. Convert the quaternion to a matrix and use matrix multiplication as normal.

Spherical interpolation

$$\mathrm{Slerp}(p_0, p_1; t) = \frac{\sin[(1-t)\Omega]}{\sin\Omega} p_0 + \frac{\sin[t\Omega]}{\sin\Omega} p_1.$$

- SLERP : Interpolation on the sphere of unit quaternions
- LERP : Linear interpolation then normalization
- US patent by Budge (2007): Fast approximation to the spherical linear interpolation function

Equivalence between Euler angles and quaternions

if you have three Euler angles (a, b, c),

then you can form three independent quaternions:

Qx = [cos(a/2), (sin(a/2), 0, 0)]
Qy = [cos(b/2), (0, sin(b/2), 0)]
Qz = [cos(c/2), (0, 0, sin(c/2))]

And the final quaternion is obtained by **Qx * Qy * Qz**.

Rigid motion interpolation

- **Screw Theory :** we can represent any movement of a solid body by a single operation which combines both the rotation and the translation.
  - As Plucker coordinates.
  - As Dual Quaternions.
  - Using Motor Theory based on Clifford Algebra.

- More about this in Lessons 11 and 12

Interpolation of matrices

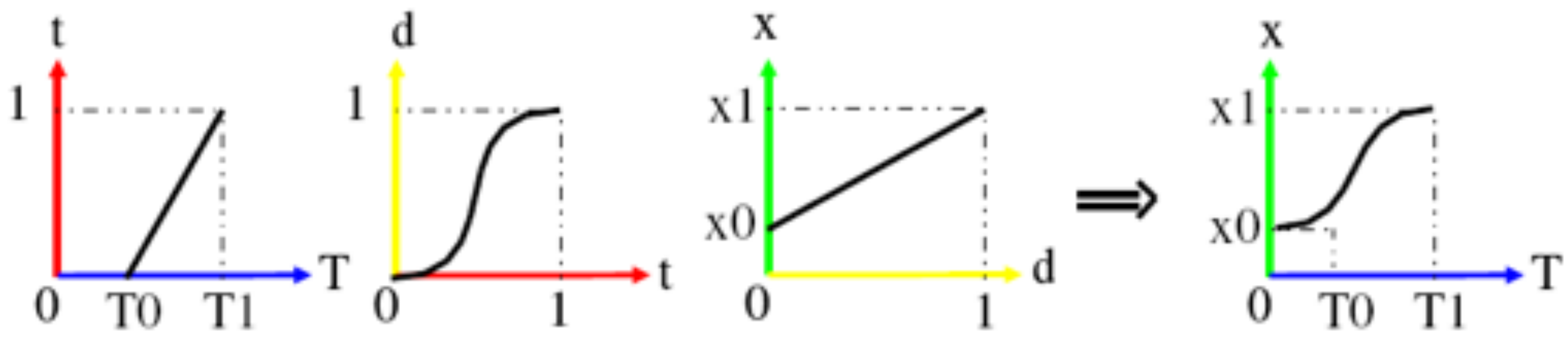
- Transformation matrix $T_f = [SR|t]$ with 12 parameters
- Non independent
- 3 translations, 3 rotations, 3 re-scalings
- Better to control them separately
- Automatic weight computation
- Wang et Philips, Multi-weight enveloping: least-squares approximation techniques for skin animation, SCA 2002

Dual quaternion interpolation

- [Skinning with Dual Quaternions](#)
  Ladislav Kavan, Steven Collins, Jiri Zara, Carol O'Sullivan.
  Symposium on Interactive 3D Graphics and Games, 2007.

Dual quaternion interpolation

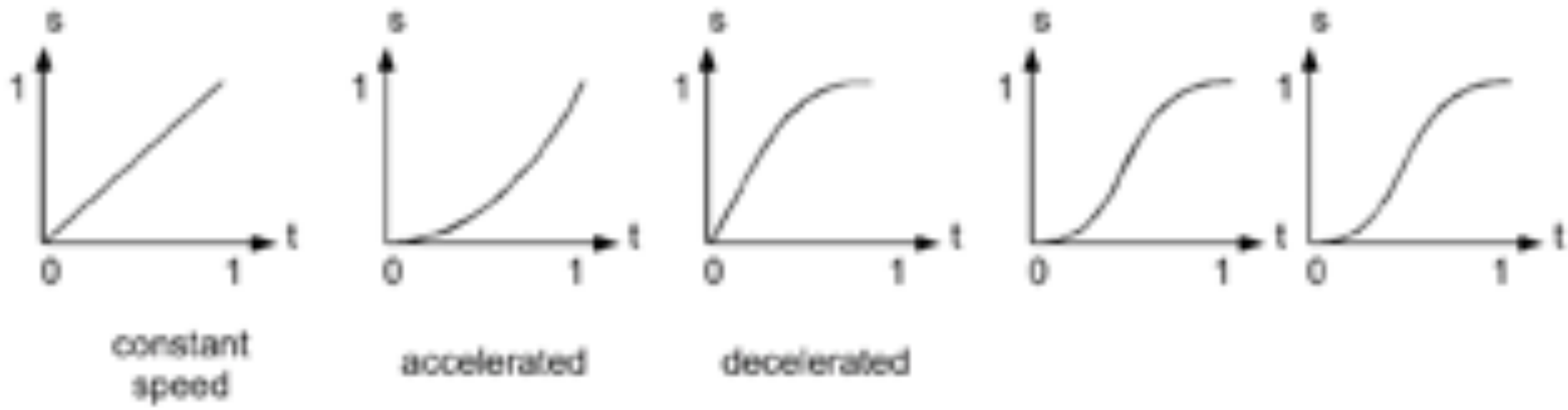
Timing curves



$$x(T) = x(d(t(T)))$$

Slow-in and slow-out

- We want to control the velocity of a moving object along a given path (spline)
- Use arclength parameterization
- Apply velocity control as $s(t)$ with $s = 0$ at starting point and $s = 1$ at end point



Principle 3 - Timing

9. Timing

Principle 3 - Timing

Timing, or the speed of an action, is an important principle because it gives meaning to movement-- the speed of an action defines how well the idea behind the action will read to an audience. It reflects the weight and size of an object, and can even carry emotional meaning.

Paper 3. Motion doodles (2004)

To appear in the ACM SIGGRAPH conference proceedings

Motion Doodles: An Interface for Sketching Character Motion

Matthew Thorne David Burke Michiel van de Panne

University of British Columbia*

Paper 3. Motion doodles (2004)

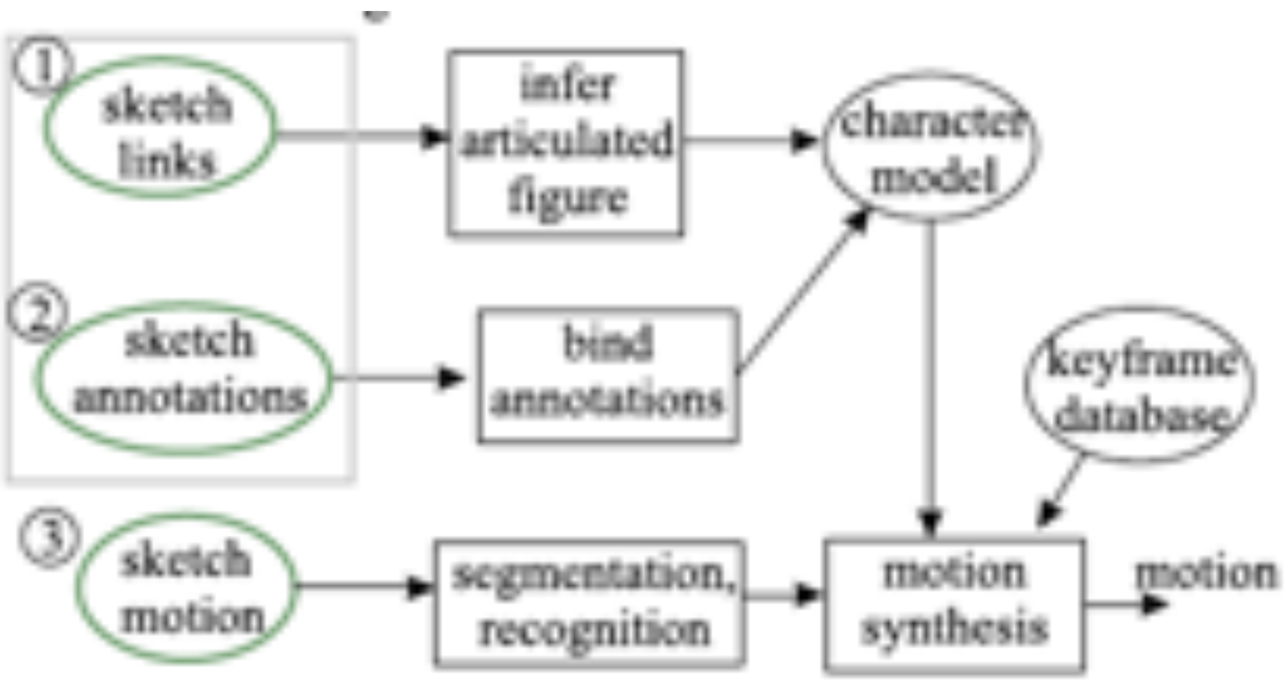


Figure 2: The sketching system.

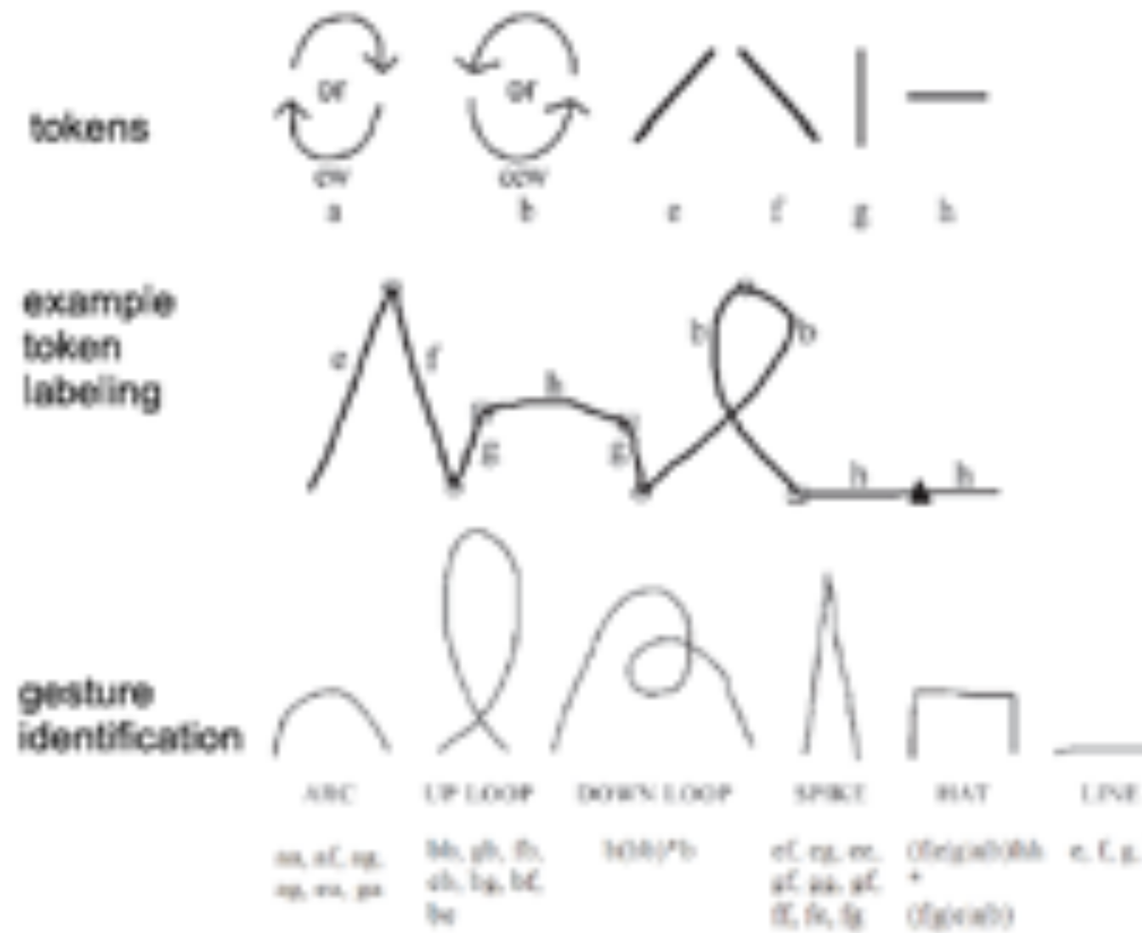


Figure 7: Segmenting the motion sketch input.

Paper 3. Motion doodles (2004)

1) Description
2) Clarity of Exposition
3) Quality of References
4) Reproducibility
5) Strengths and weaknesses
6) Rating (1-5)

Paper 3. Motion doodles (2004)

1) Problem statement
2) Scientific contributions
3) Experimental validation
4) Limitations
5) Impact