Simulation et Applications Interactives

Deformation of Complex Systems

Caroline Larboulette

Fall 2018

Overview

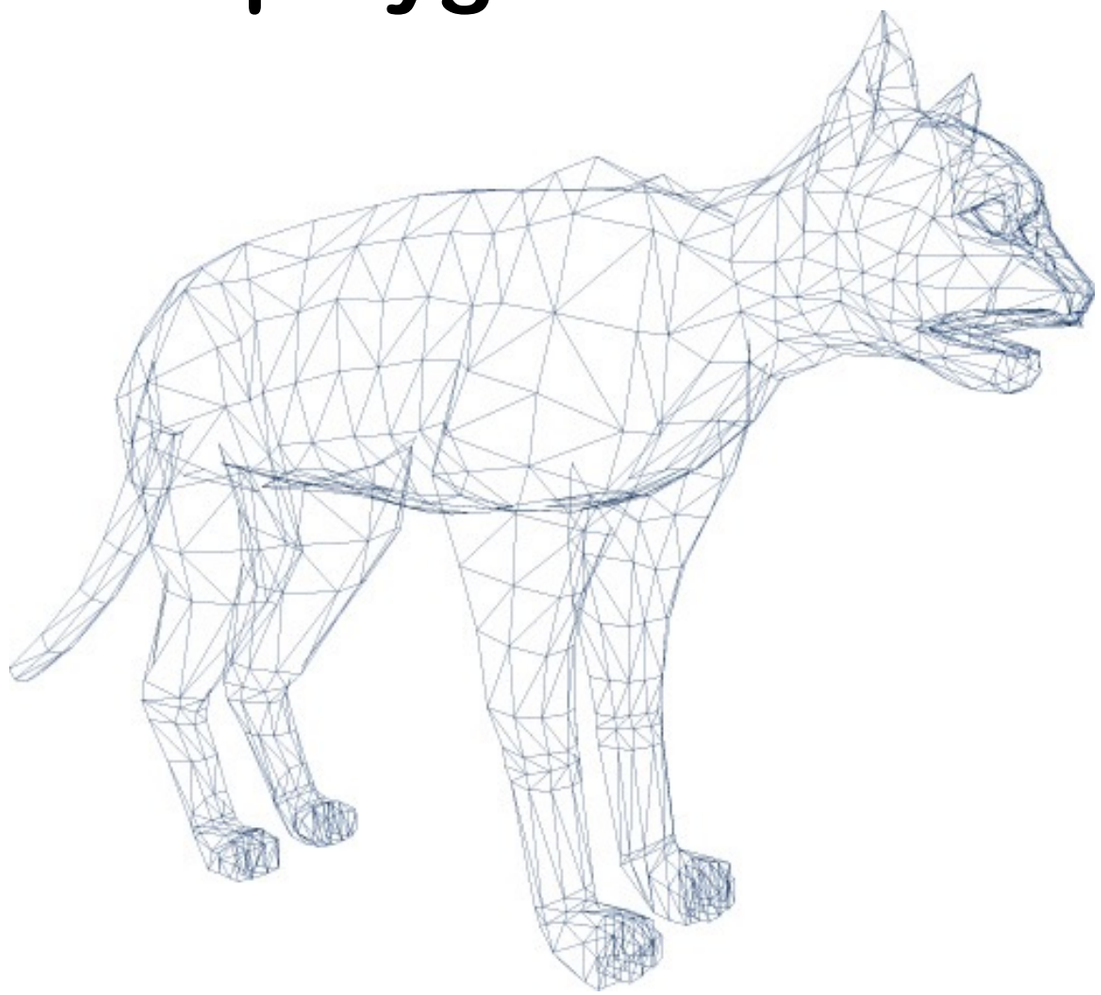
- Articulated Models
- Skinning
- Anatomically Based Techniques
- Facial Animation
- Hair Animation
- Grass / Tree Animation

Articulated Models

Articulated Models

Traditional Framework

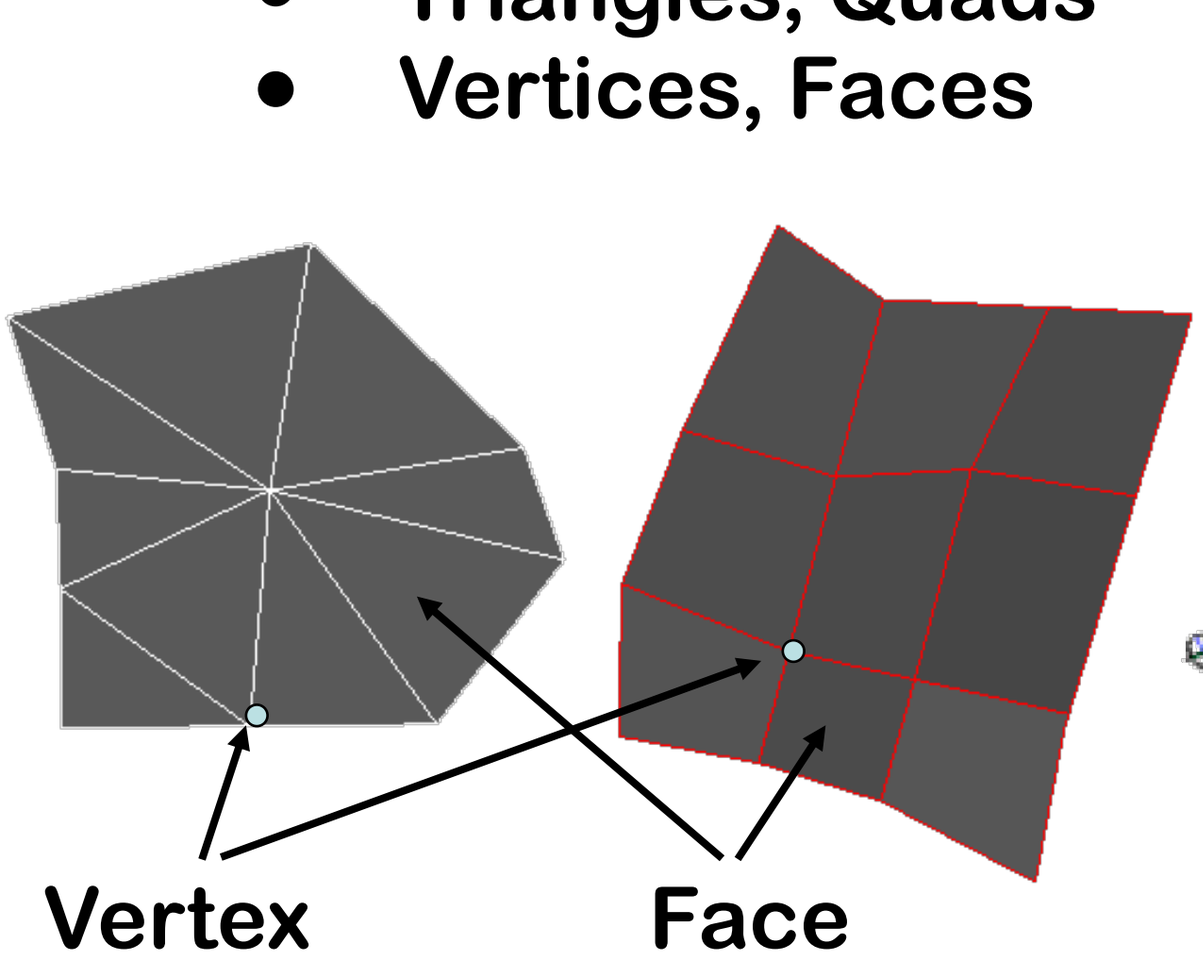
- An animation skeleton
- A skin as a 3D polygonal mesh

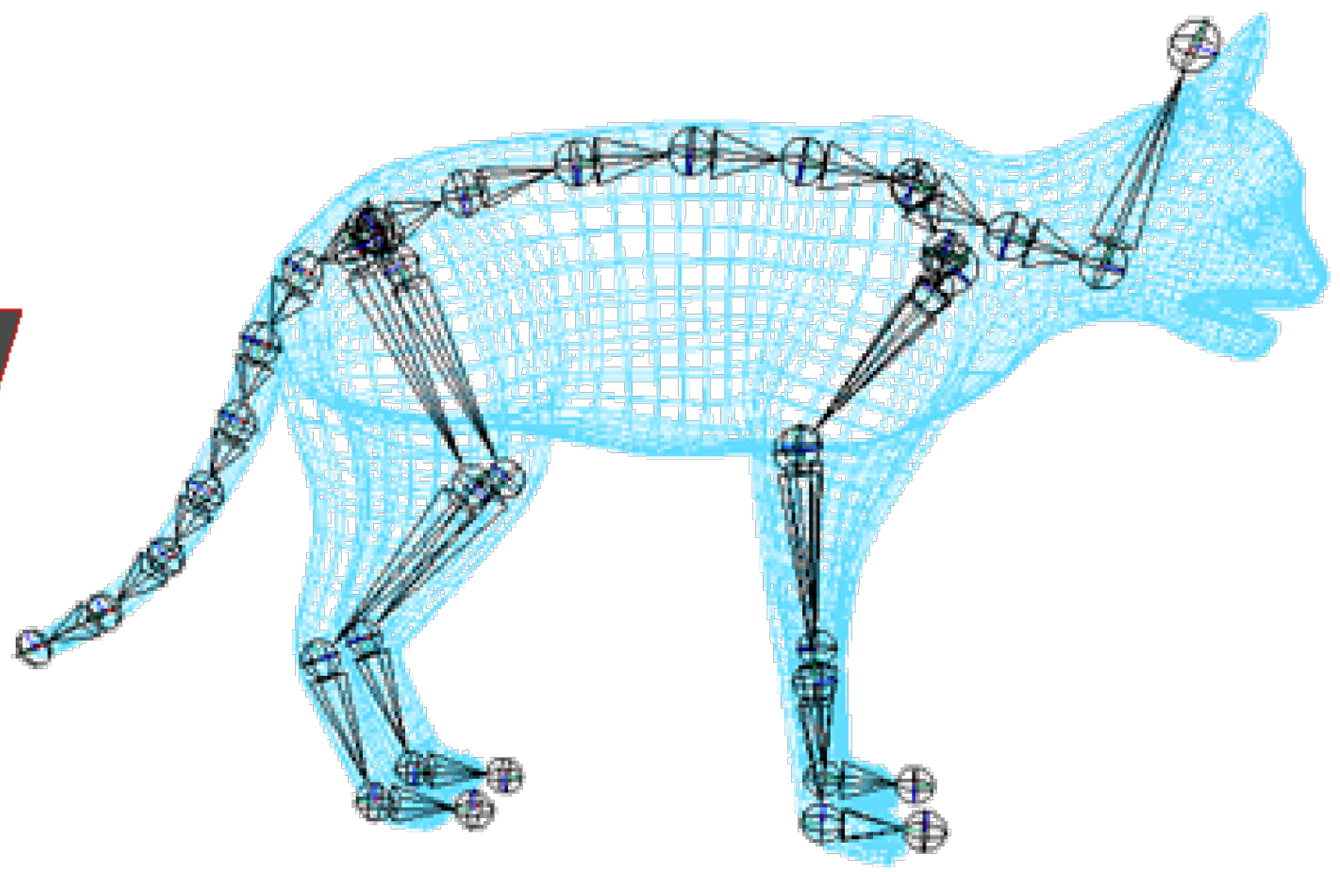


Articulated Models

Traditional Framework

1. Skin: polygonal mesh
   - Triangles, Quads
   - Vertices, Faces

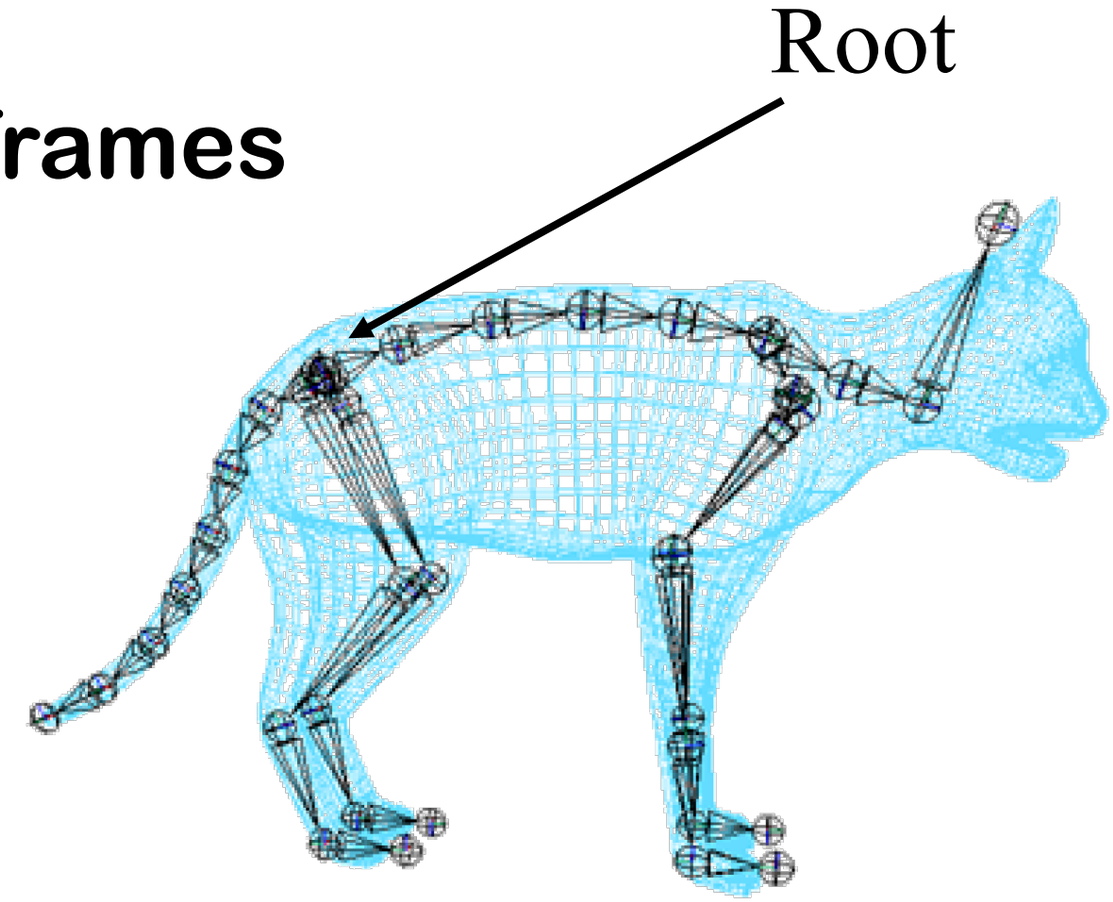




Articulated Models

Traditional Framework

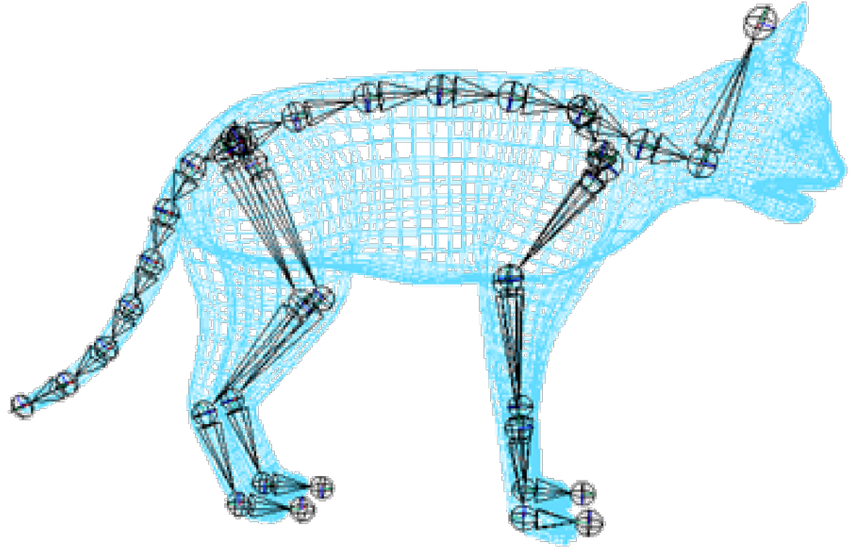
1. Skin: polygonal mesh
   - Triangles, Quads
   - Vertices, Faces
2. Skeleton: hierarchy of frames
   - The Root frame
   - Local transformations
     Rotation + Translation



Articulated Models

Traditional Framework

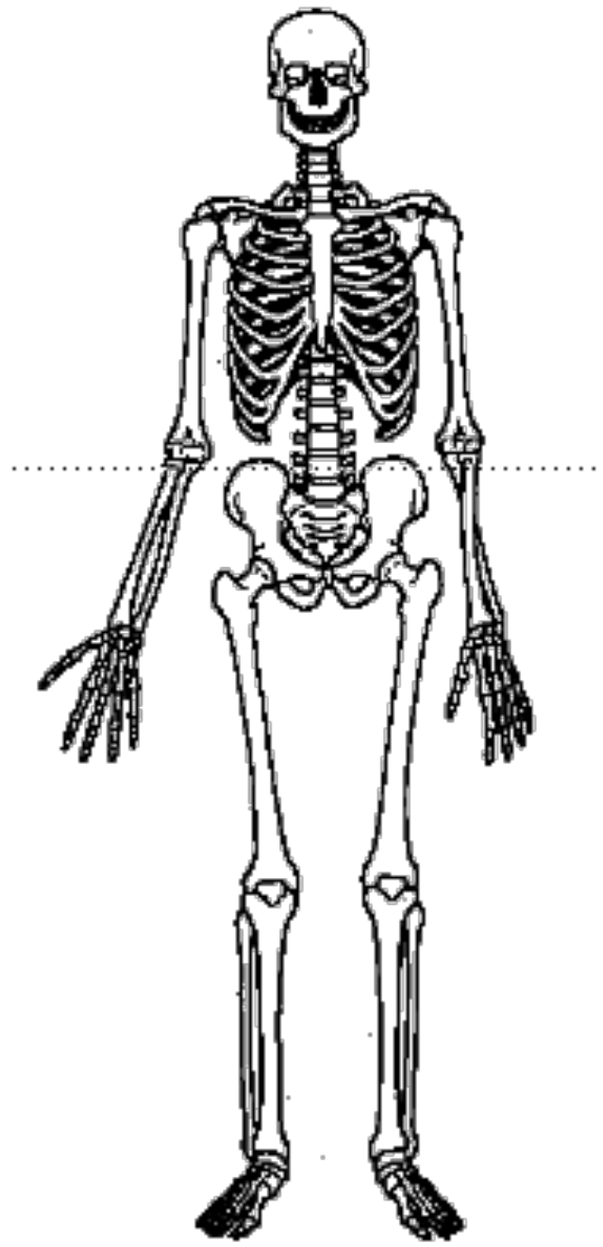
1. Skin: polygonal mesh
   - Triangles, Quads
   - Vertices, Faces
2. Skeleton: hierarchy of frames
   - The Root frame
   - Local transformations
     Rotation + Translation
3. Rig / Skinning
   - How mesh vertices are attached to the skeleton's frames / How mesh vertices positions are computed



Articulated Models

Skeleton

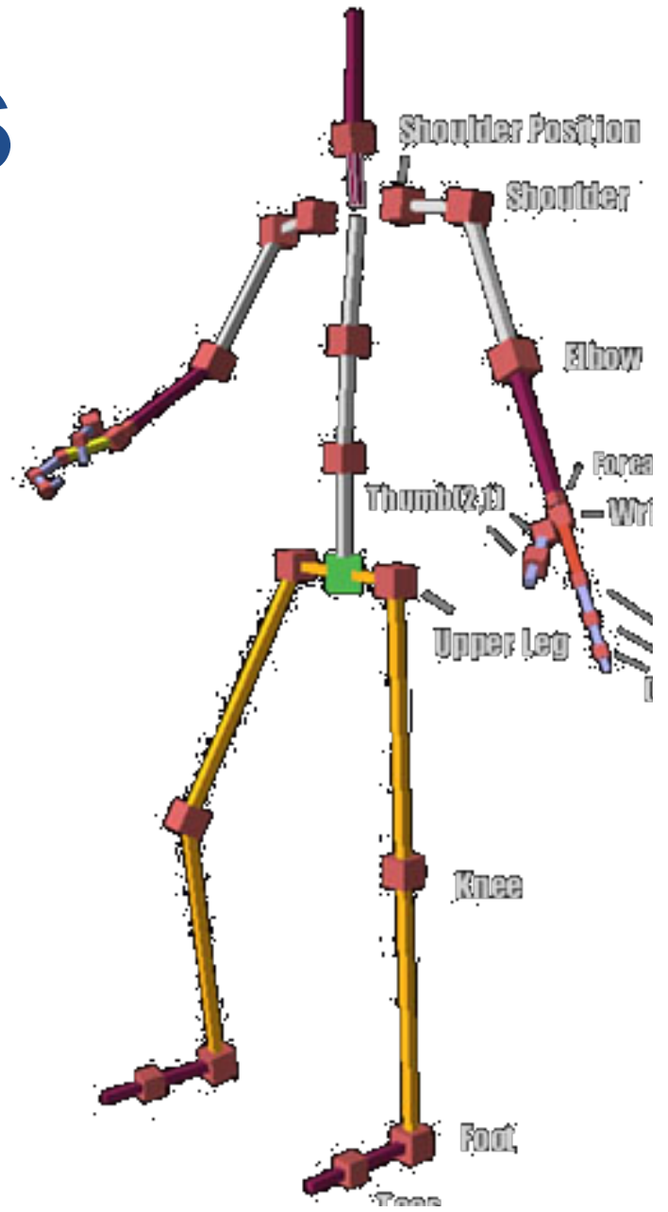
- Real Life
  - Bones
  - Joints



Articulated Models

Skeleton

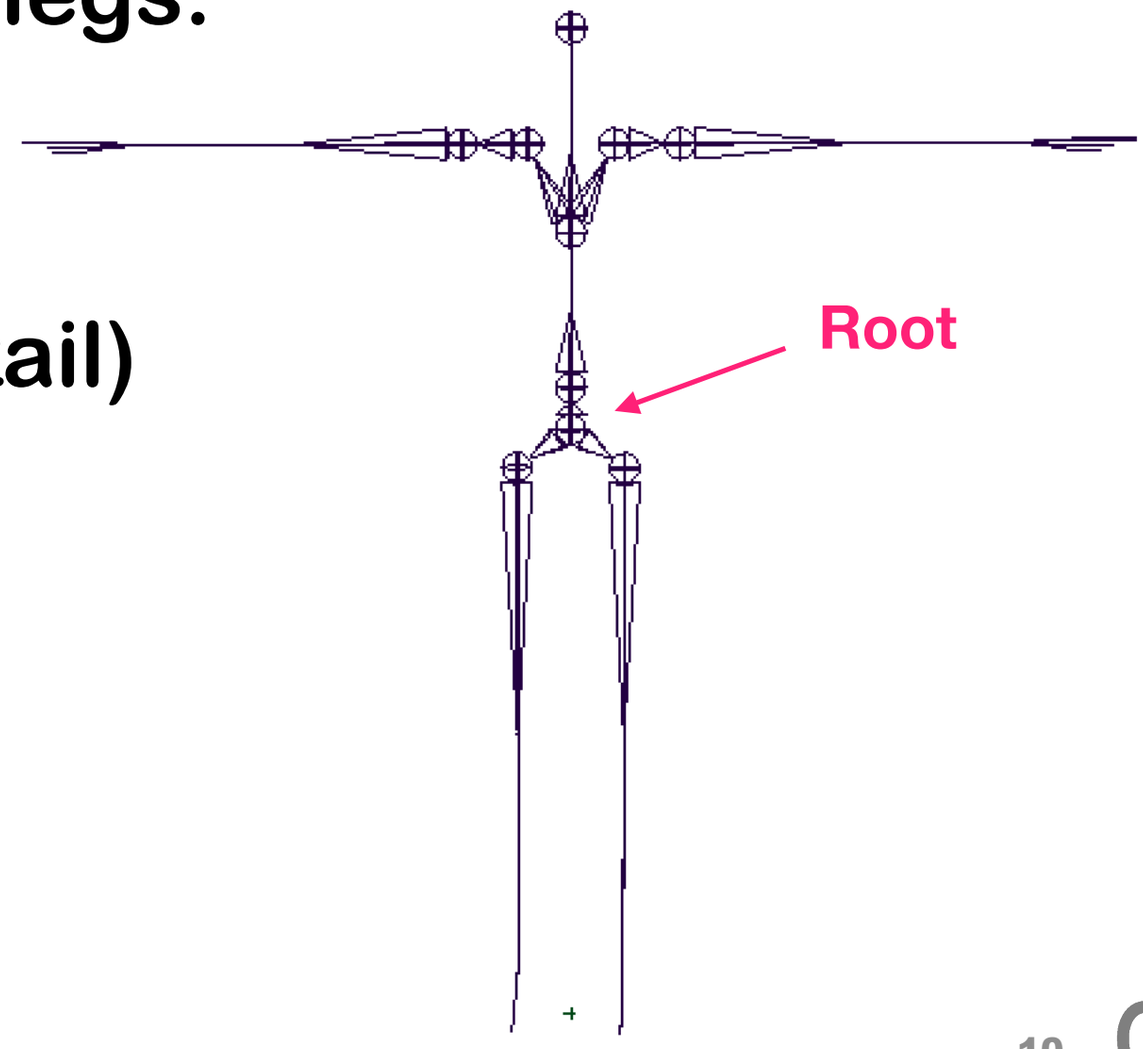
- Real Life
  - Bones
  - Joints
- Animation
  - Bone: a segment
  - Joint : a single point (called node)
  - The segments (bones) and points (joints) are connected in a hierarchy of frames
    - local coordinates
    - root, leaf, parent, child nodes



Articulated Models

Skeleton

- From root node to leaf nodes
- Humans or animals with legs:
  - Root : basin
  - Leaves : hands, feet, (tail)



Articulated Models

Skeletal Animation

- Keyframes:
  - Key poses of the character at given times
  - In-between poses are interpolated
- Forward and Inverse kinematics
  - FK : each joint position / orientation is set by hand
  - IK : the position / orientation of each joint in a chain is automatically computed from end positions

Articulated Models

Rigging

- Problem: the mesh does not “follow” the skeleton
- Solution: rigging by skinning also called
  - Smooth skinning,
  - Linear blend skinning or
  - SSD (Skeleton Subspace Deformations)

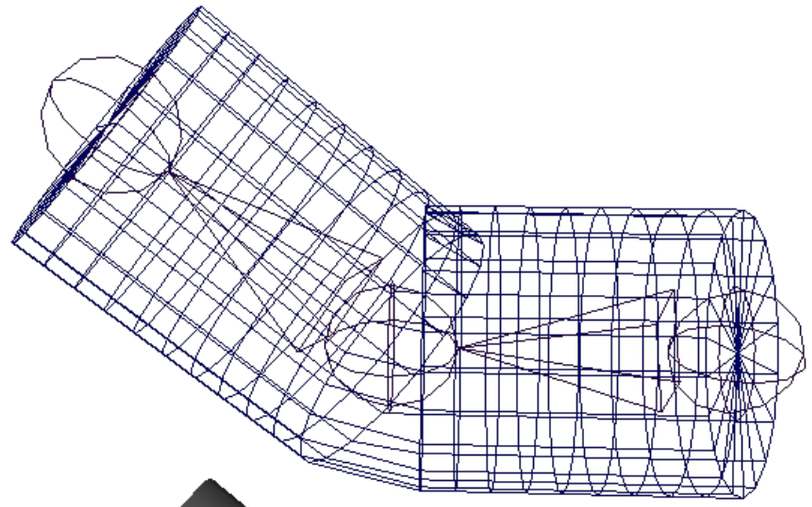
Skinning

Definition

- Skinning is the name given to any technique that deforms the skin of a character
- By extension, the term skinning is commonly used to describe skeleton subspace deformations
  - Static (depends only on the pose)
  - Skeleton driven

Rigid Skinning

- Rigid Skinning: position of a mesh vertex only depends on 1 joint

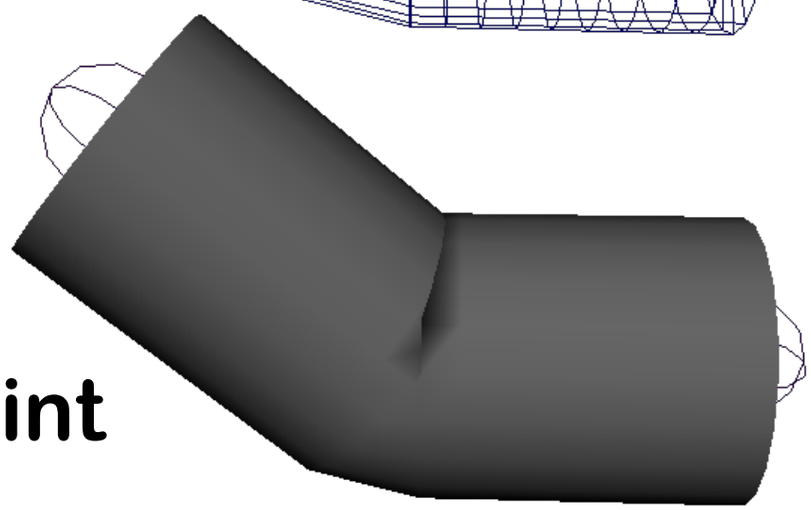


$$p_{v_i} = T_f \cdot p_{0v_i}$$

p_{v_i} = position of vertex i

p_{0v_i} = initial position of vertex i

T_f = transformation matrix of the joint



Rigid Skinning

- Rigid Skinning: position of a mesh vertex only depends on 1 joint

$$p_{v_i} = T_f \cdot p_{0v_i}$$

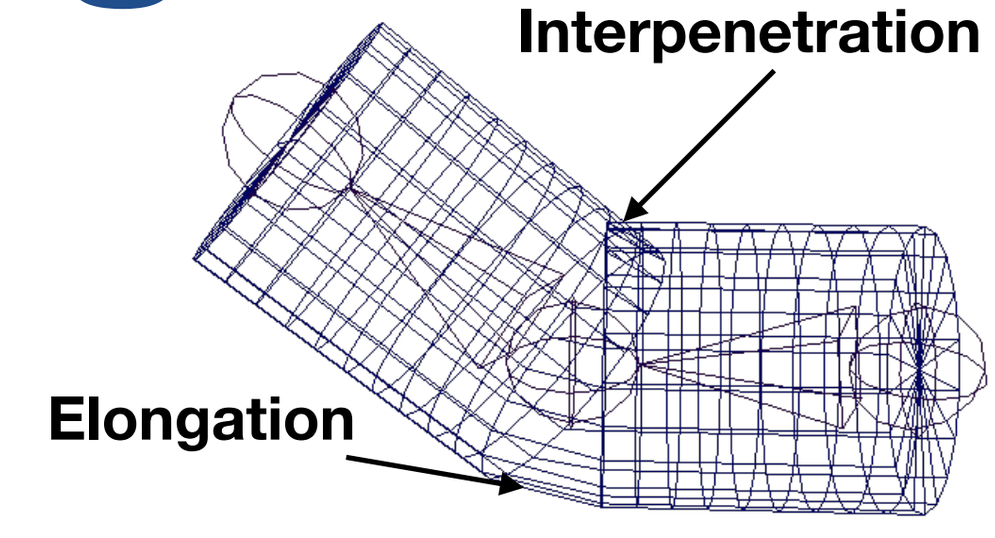
p_{v_i} = position of vertex i

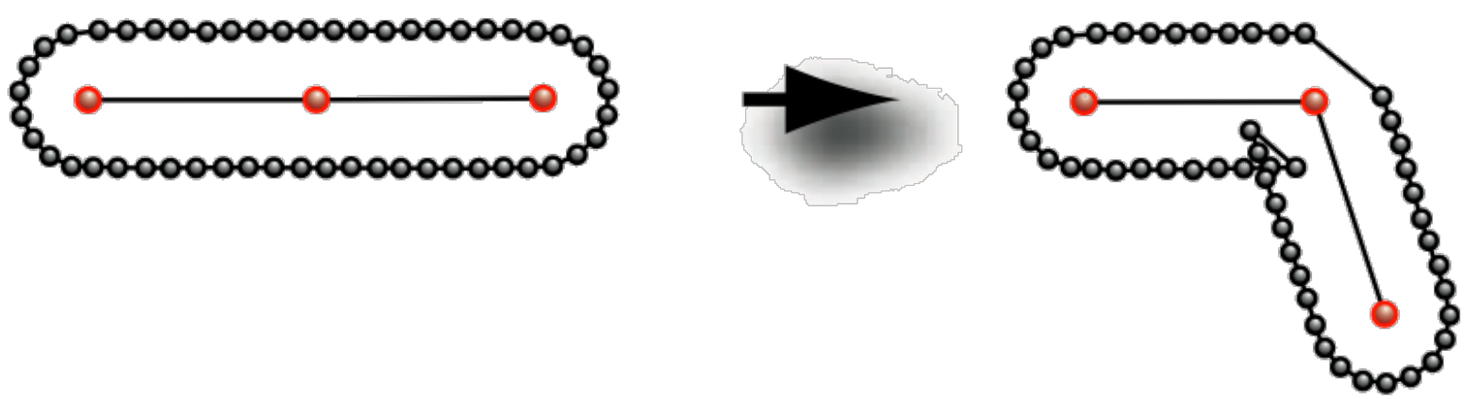
p_{0v_i} = initial position of vertex i

T_f = transformation matrix of the joint

Rigid Skinning

- Drawbacks
  - Holes (if separate meshes)
  - Mesh elongation (if only one mesh)
  - Mesh interpenetrations (if only one mesh)





Smooth Skinning

- Each vertex is rigidly attached to several bones, its final position is interpolated
- Common technique used in animation software (Maya, 3DS, Poser …) and video games, movies, etc …

Smooth Skinning

Classic Linear

- Rigid Skinning: position of a mesh vertex only depends on 1 joint

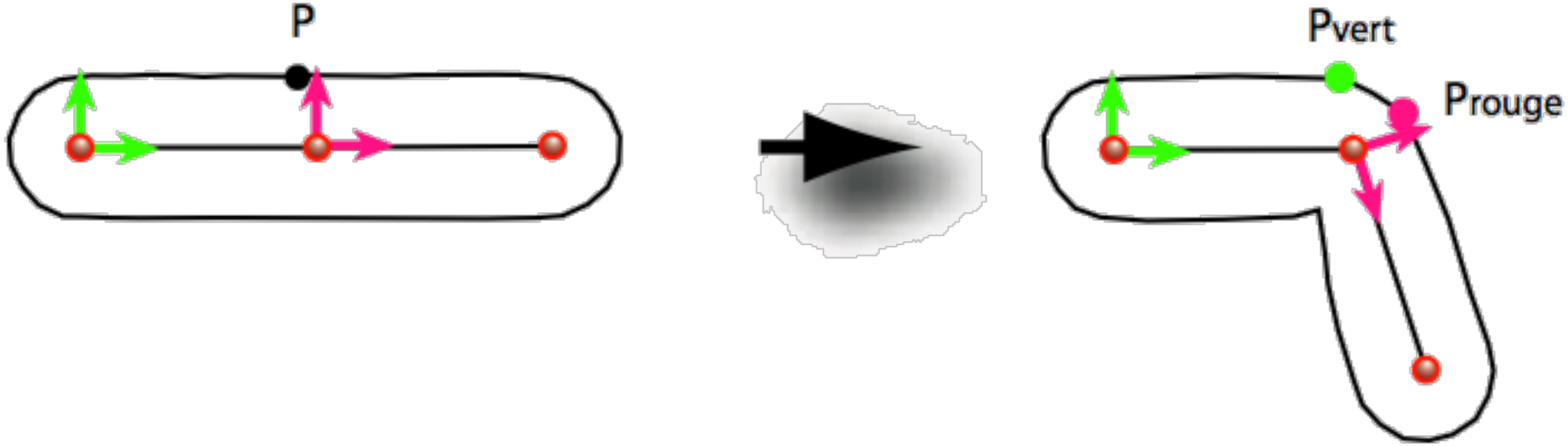
$$p_{v_i} = T_f \cdot p_{0v_i}$$

- Linear Skinning: position of a mesh vertex depends on n joints

$$p_{v_i} = \left(\sum_{f=0}^{n} w_{if} . T_f\right) . p_{0v_i}$$

Smooth Skinning

Classic Linear

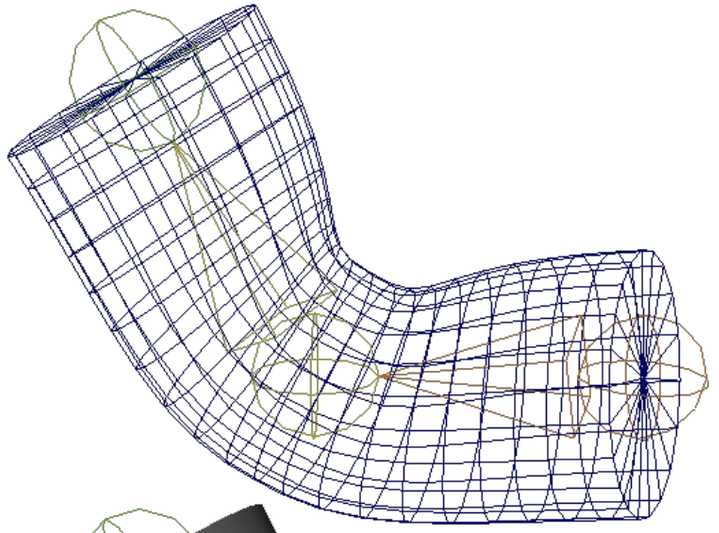


Smooth Skinning

Classic Linear

- Smooth Skinning: position of a mesh vertex depends on n joints : linear interpolation

$$p_{v_i} = (\sum_{f=0}^{n} w_{if} . T_f) . p_{0v_i}$$

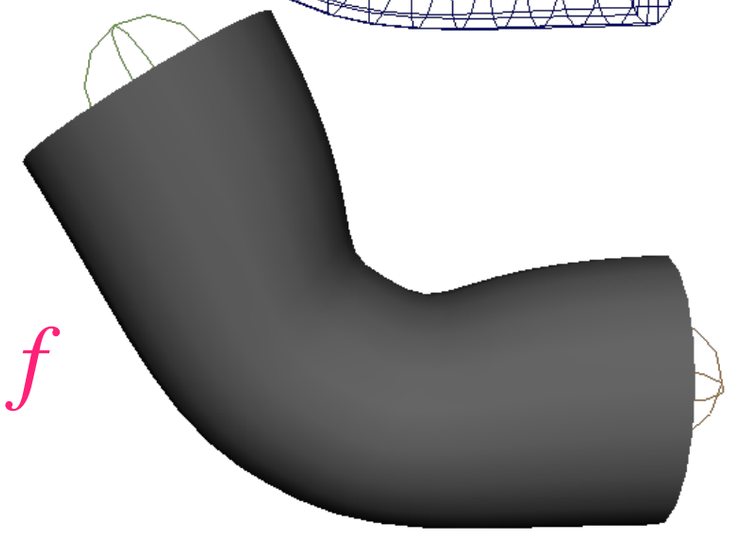


p_{v_i} = position of vertex i

p_{0v_i} = initial position of vertex i

T_f = transformation matrix of the bone f

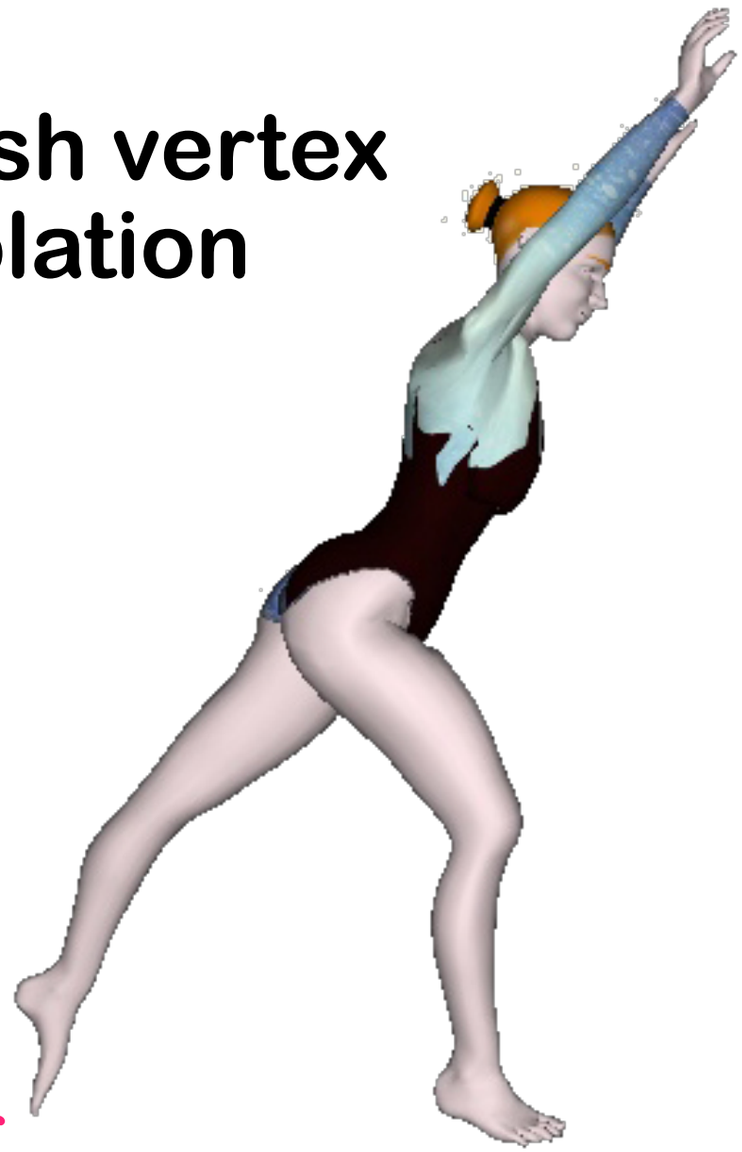
w_{if} = weight of vertex i for joint f



Smooth Skinning

Classic Linear

- Smooth Skinning: position of a mesh vertex depends on n joints : linear interpolation



$$p_{v_i} = (\sum_{f=0}^{n} w_{if} . T_f) . p_{0v_i}$$

p_{v_i} = position of vertex i

p_{0v_i} = initial position of vertex i

T_f = transformation matrix of the bone f

w_{if} = weight of vertex i for joint f

Rigid vs Smooth Skinning

Weights Computation

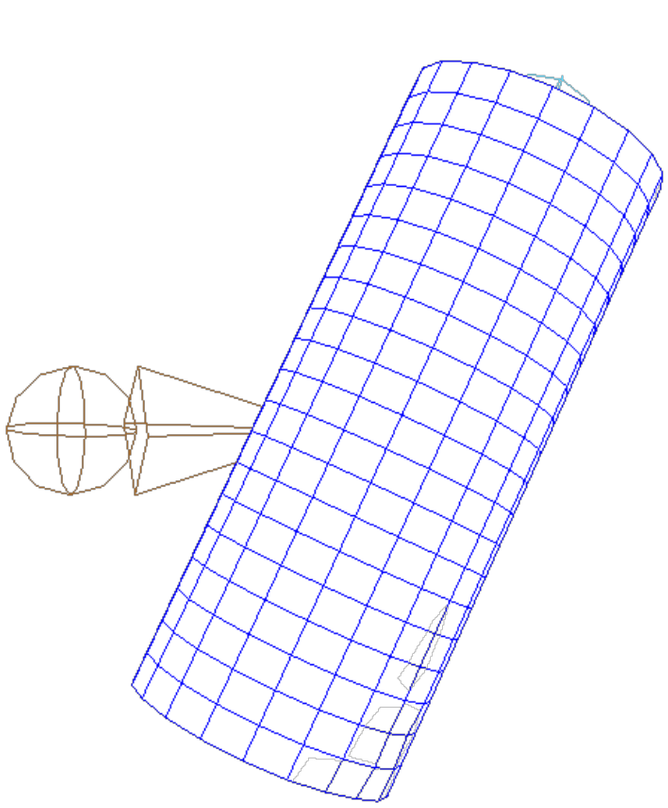
- No good automatic technique
  - Usually done by hand
- Approximation can be computed by

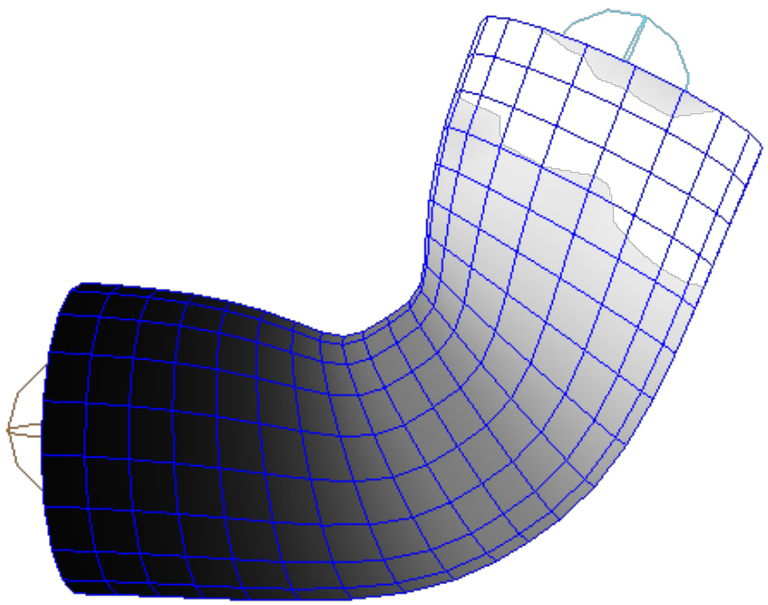
$$w_{if} = \frac{1}{d_i^2}$$

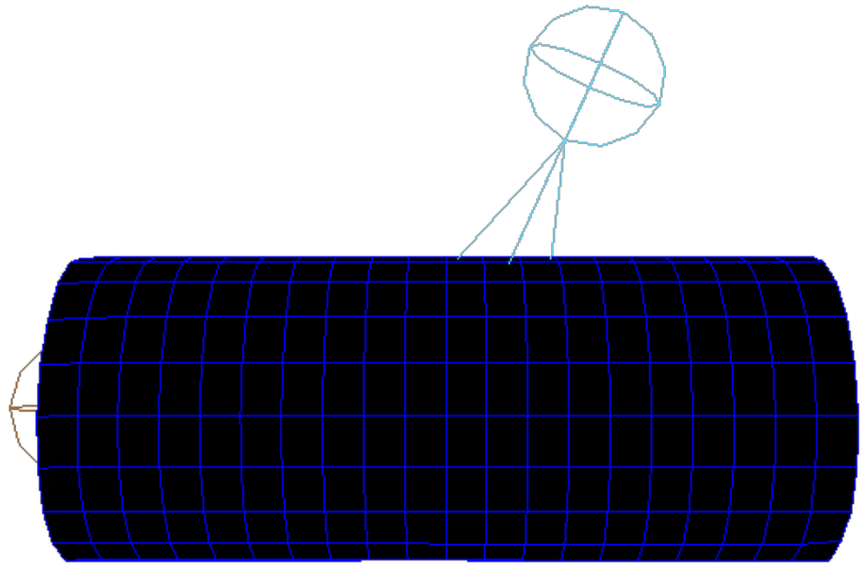
- In any case, weights should normalized: $\sum w_{if} = 1$
- w_{if} painted by hand in Maya

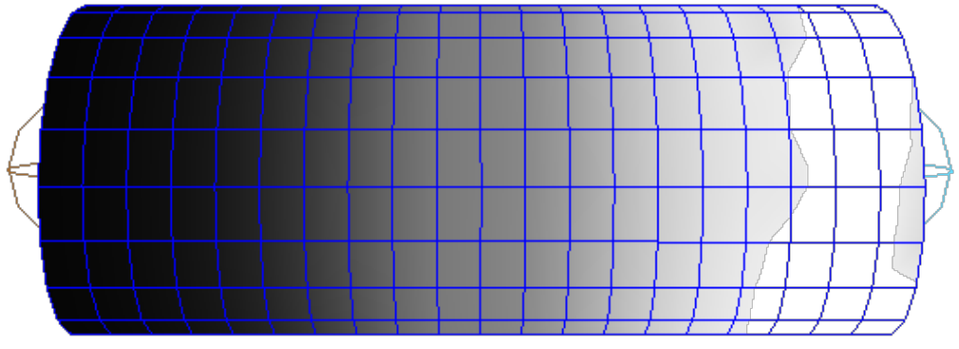
Weights Computation

Example



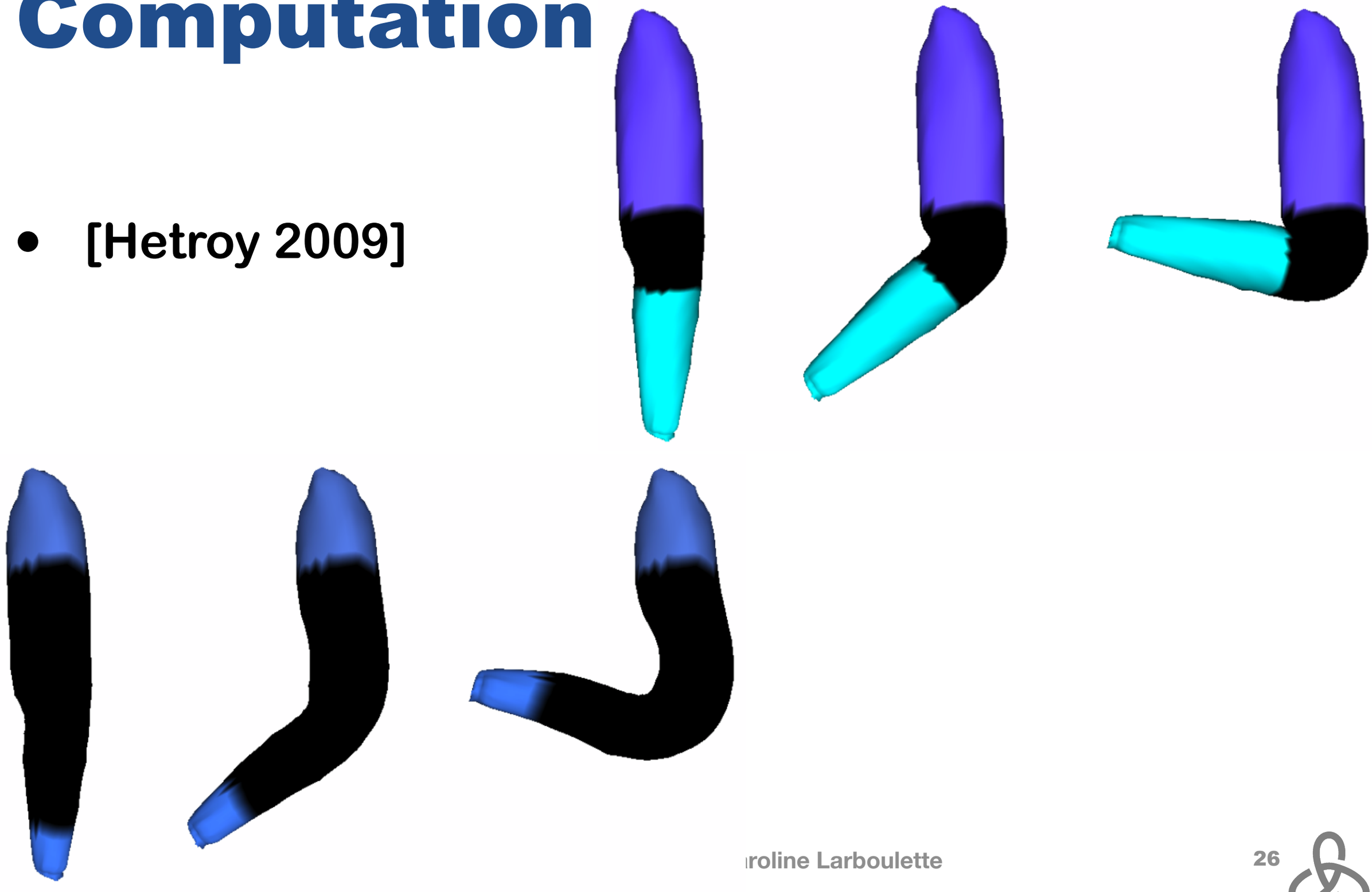






Automatic Weights Computation

- [Hetroy 2009]



Skinning Weights Computation

1. Mesh segmentation (using the skeleton joints)
2. Overlap generation
3. A distance to the boundary is computed using
   - Euclidean / Geodesic / Harmonic distance
4. Weights are computed using the normalized distance multiplied by a Hermite function to get non-linear behavior

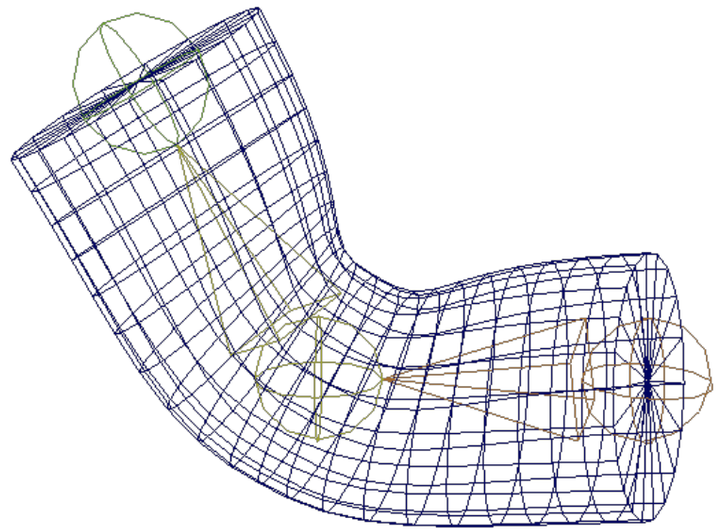
Skinning Weights Computation

- VIDEO

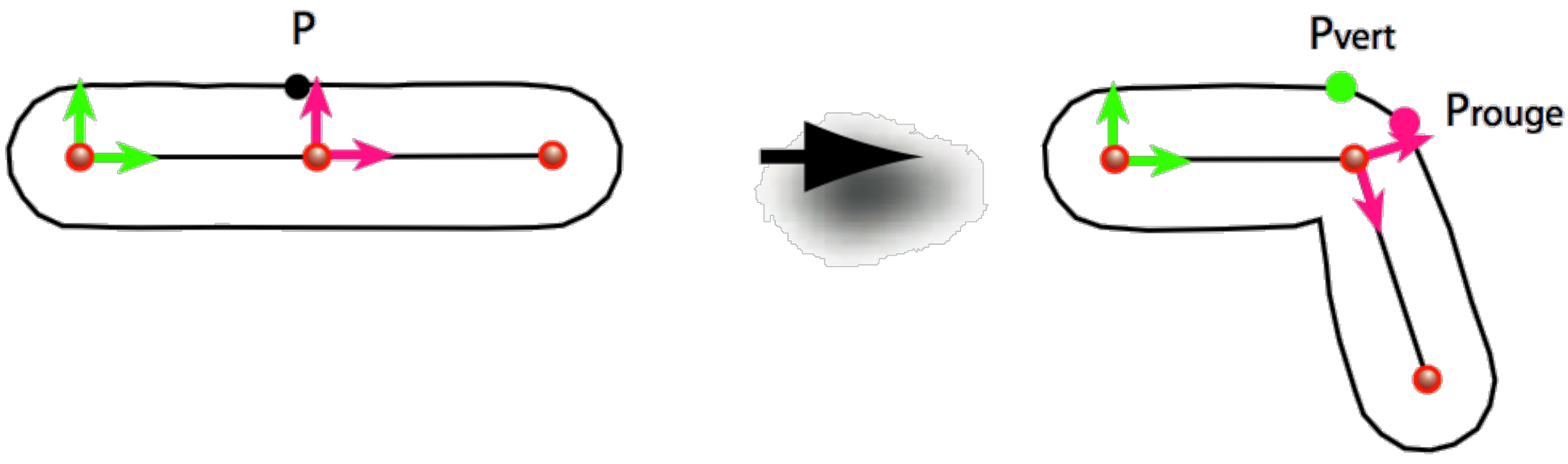
Classic Linear Skinning

Advantages

- Advantages
  - No holes
  - No triangle elongation



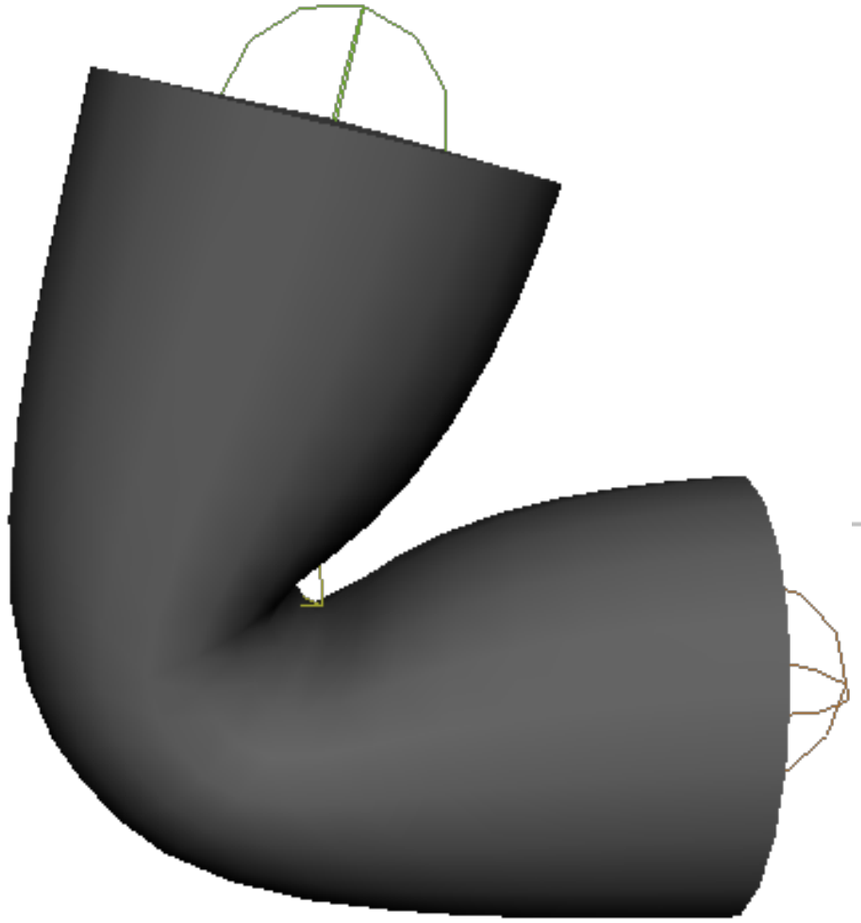
- No mesh interpenetration

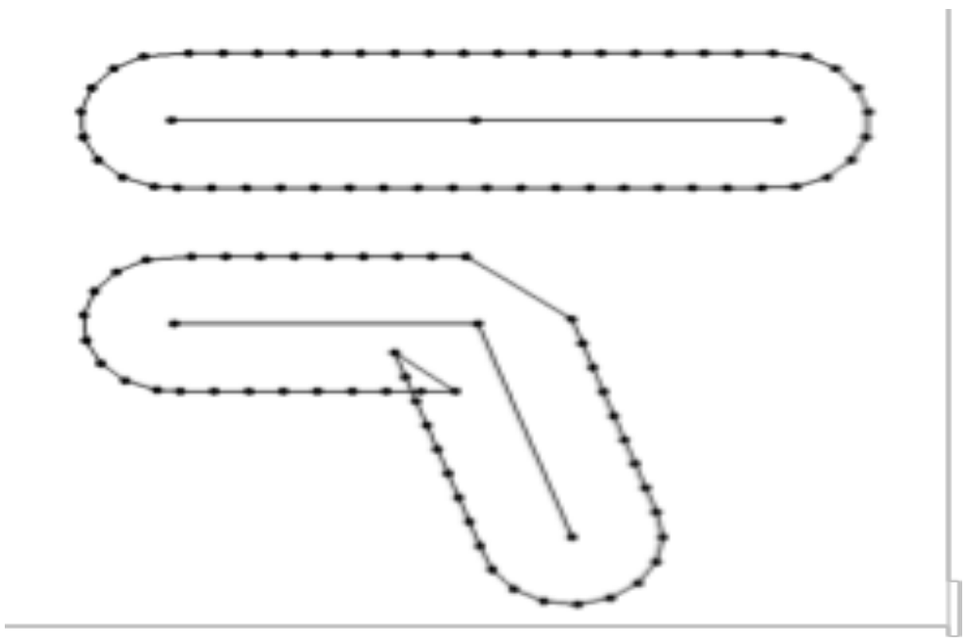


Classic Linear Skinning

Drawbacks

- Not perfect : 2 important problems
  - Collapsing joint

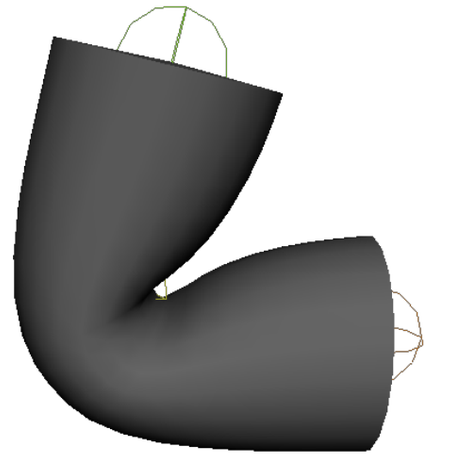


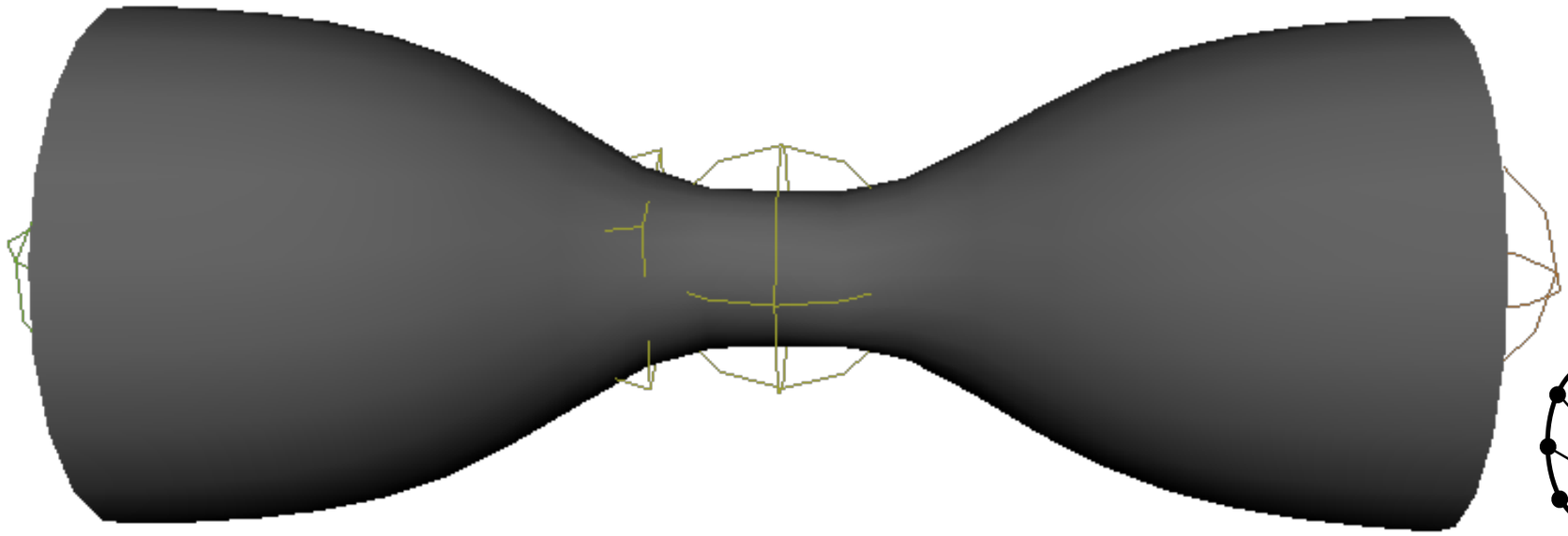


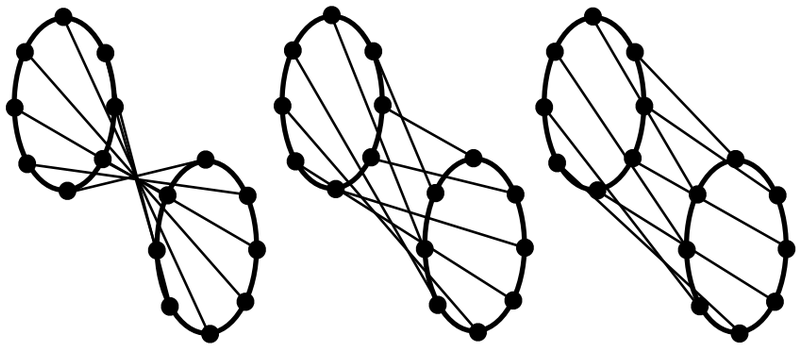
Classic Linear Skinning

Drawbacks

- Not perfect : 2 important problems
  - Collapsing joint
  - Candy-wrapper effect



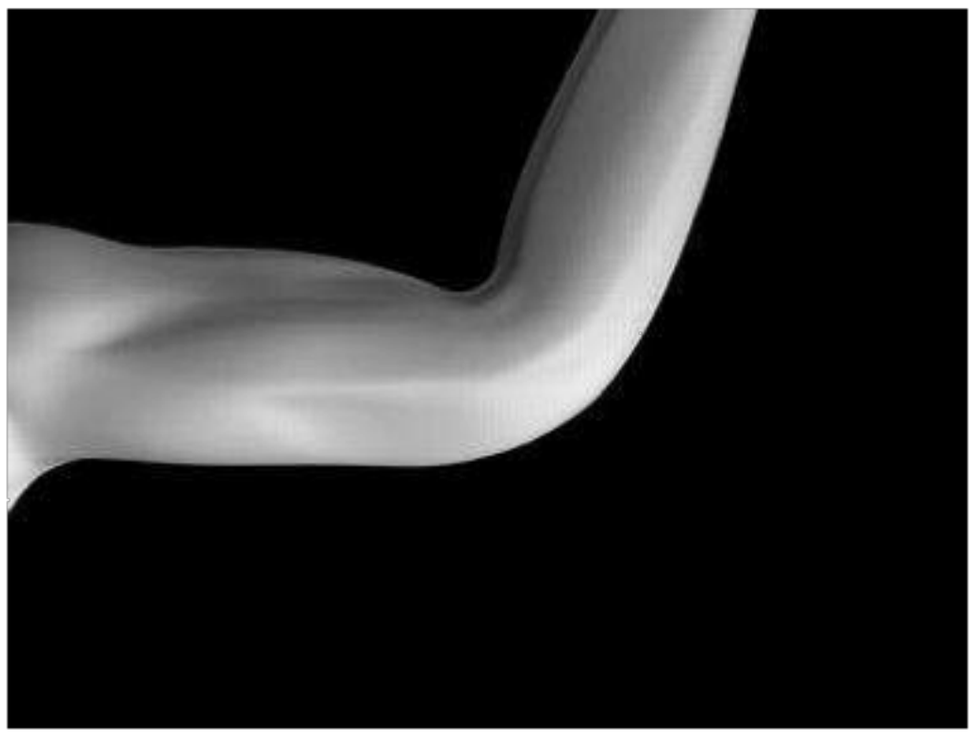


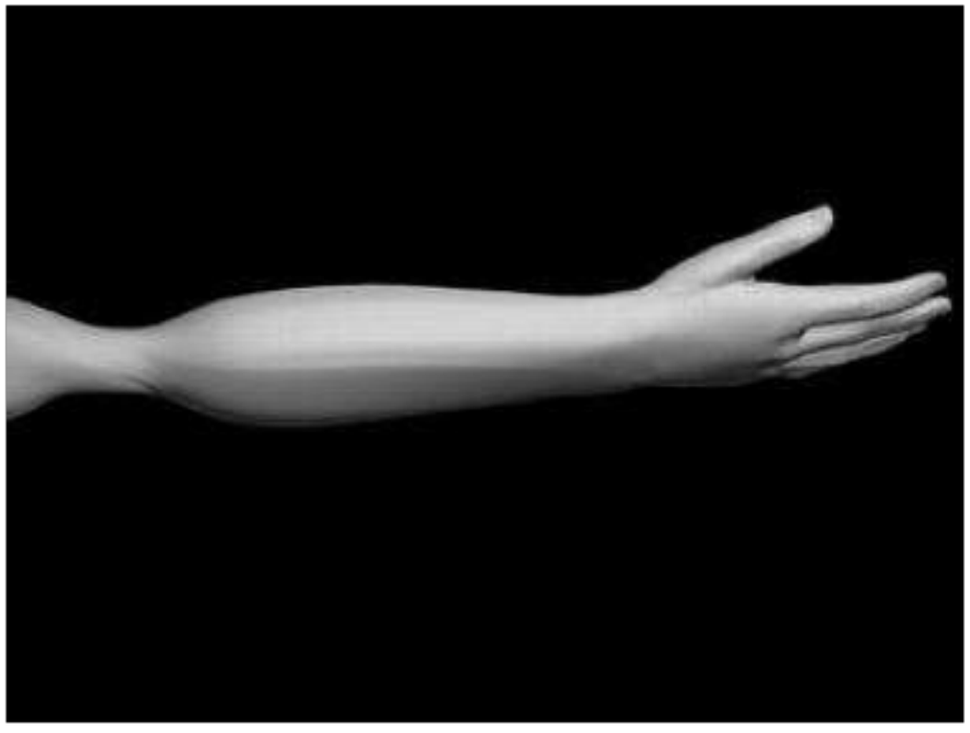


Classic Linear Skinning

Drawbacks

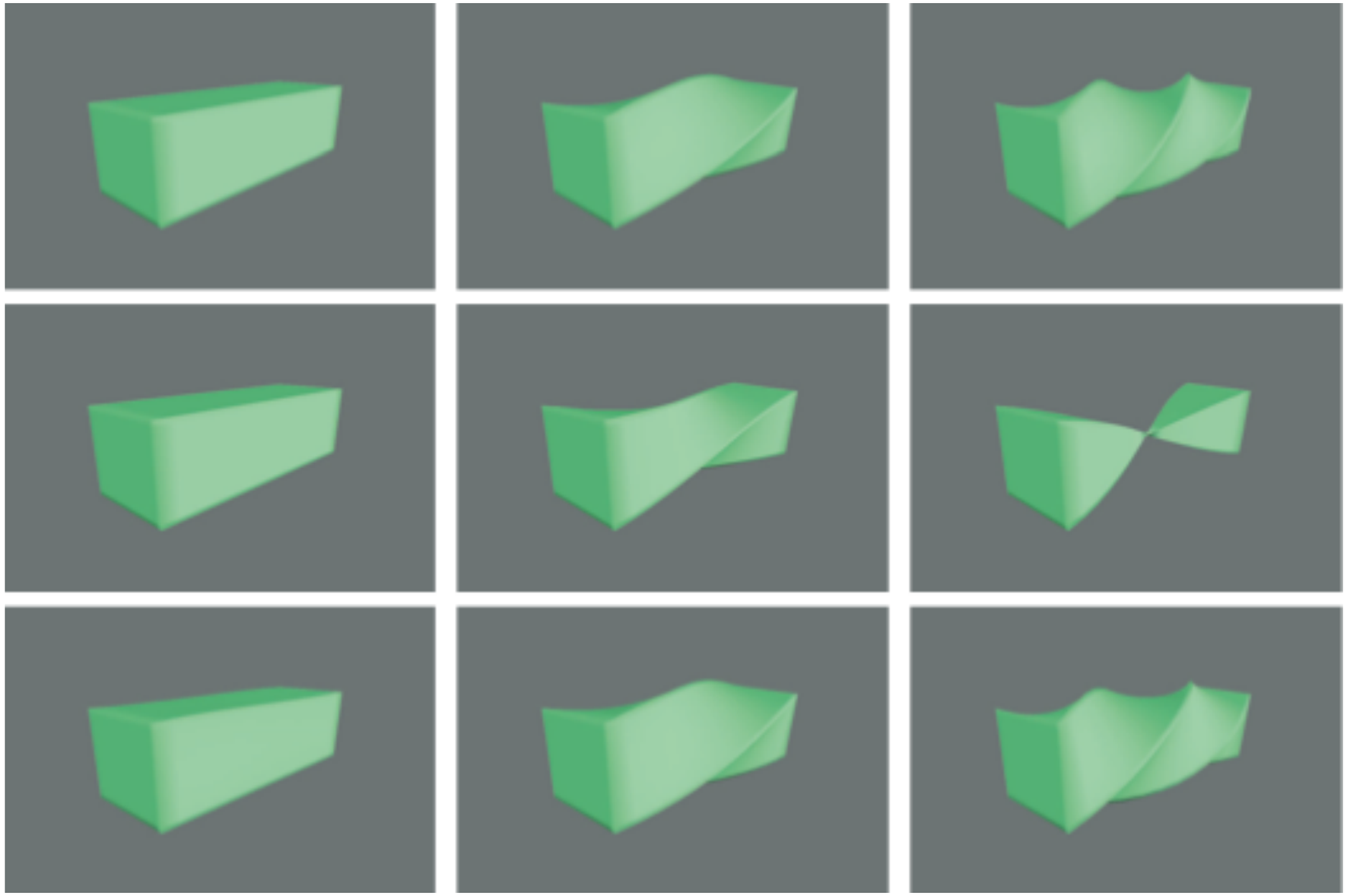
- Drawbacks mainly due to the linear interpolation
- No dynamics





Classic Linear Skinning

Drawbacks



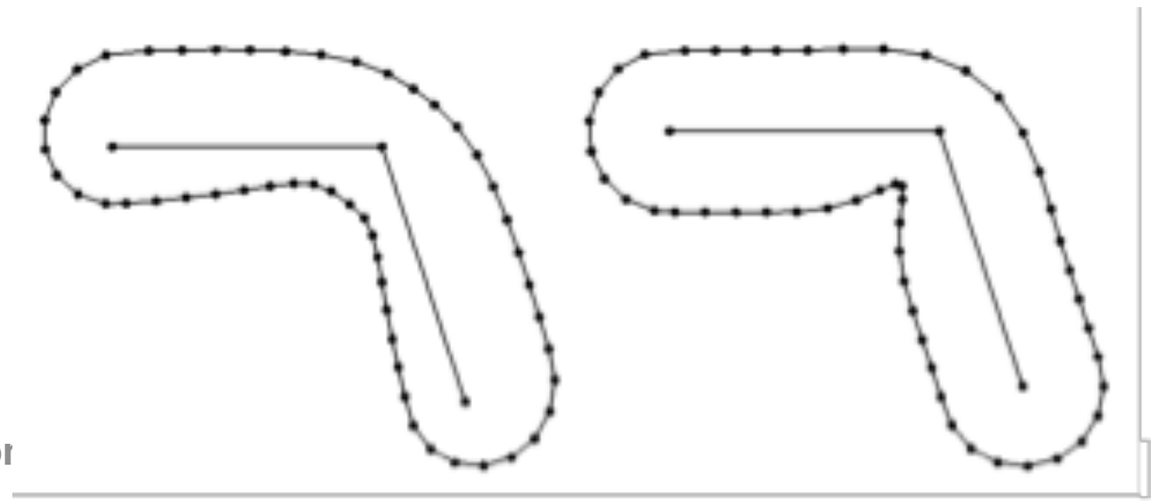
Classic Linear Skinning

Solutions

- Do not create poses with big variation angles
- Move the camera in such a way that the problematic areas cannot be seen
- Use a more sophisticated linear blend skinning solution
  - By adding frames [Bloomenthal, Mohr]
  - By adding weights [Wang]
- Use a non-linear blend skinning solution

Bloomenthal technique

- Key Idea: add additional frames
  - First the medial axis of the surface is computed
  - A frame f and its associated weights w_{if} is added for each primitive (segment, triangle) of the medial

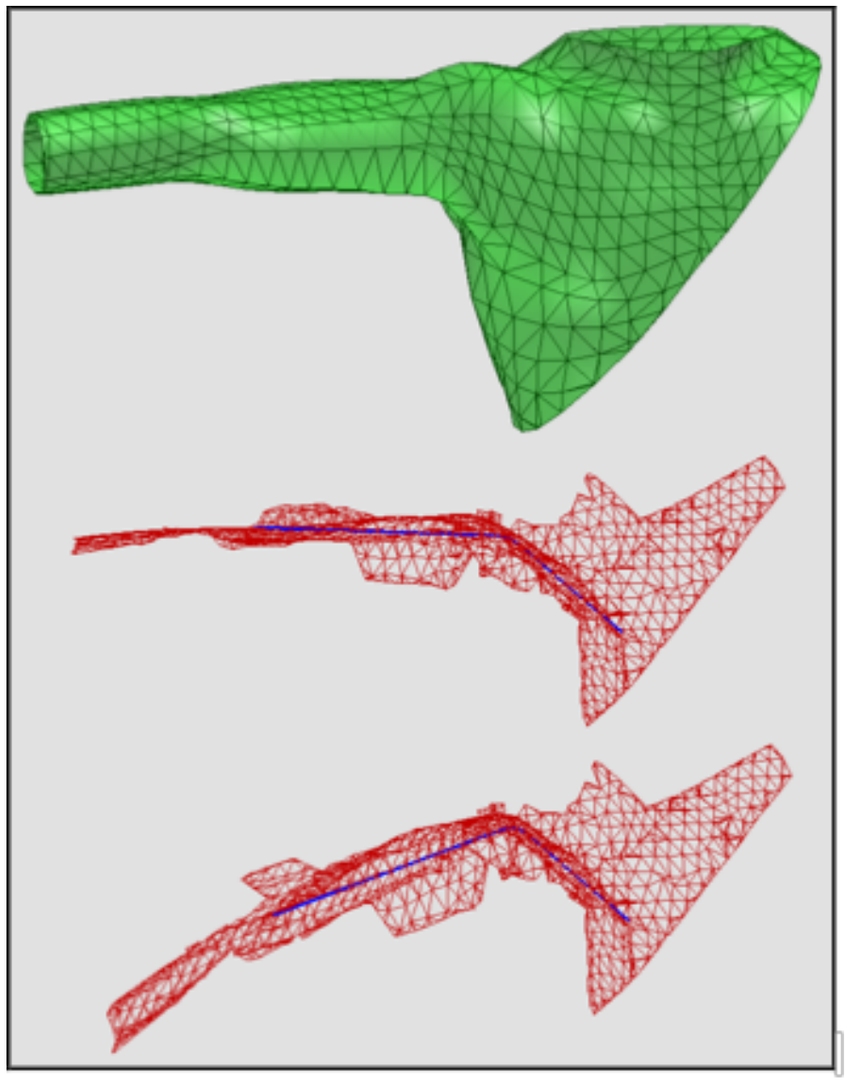


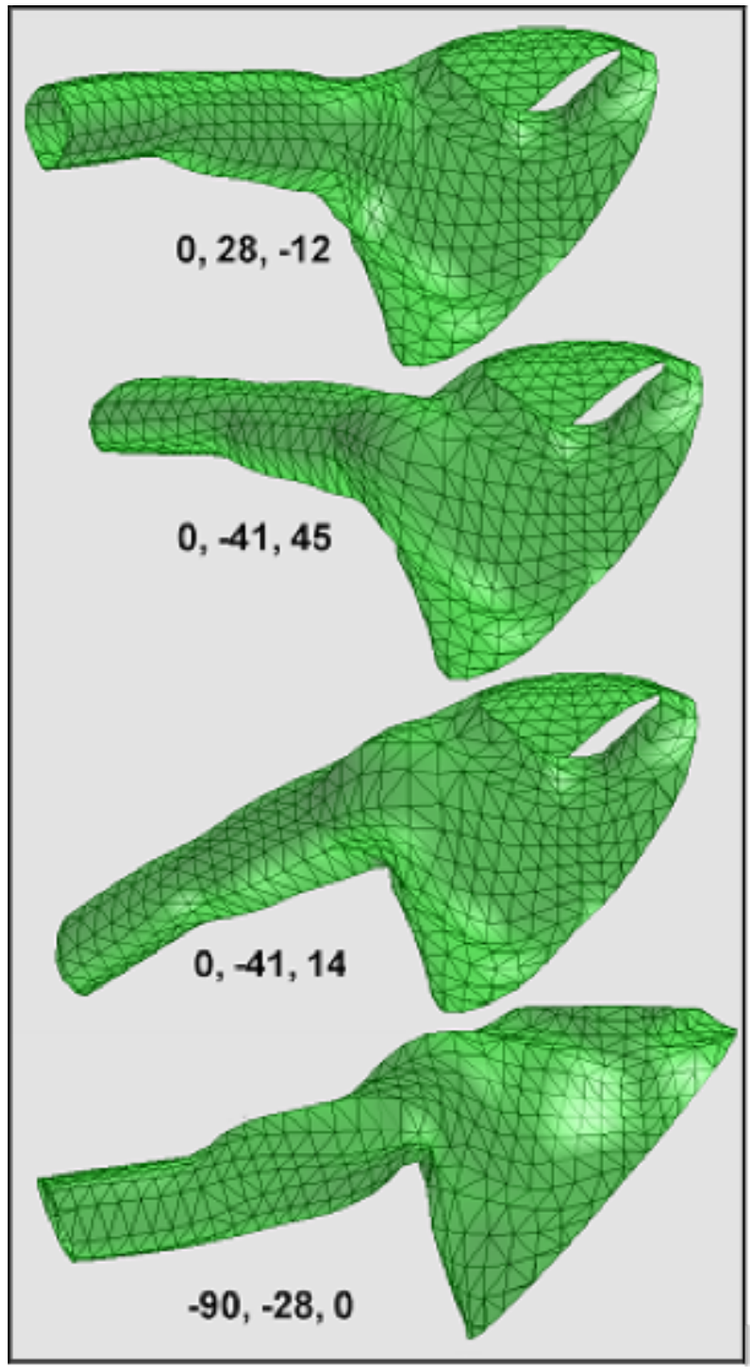
Bloomenthal Technique

- The weight depends on the distance to the medial instead of the distance to the skeleton -> better approximation of the thickness

- During animation, the medial frames are transformed by skinning. They then deform the skin by a second skinning operation

Bloomenthal Technique



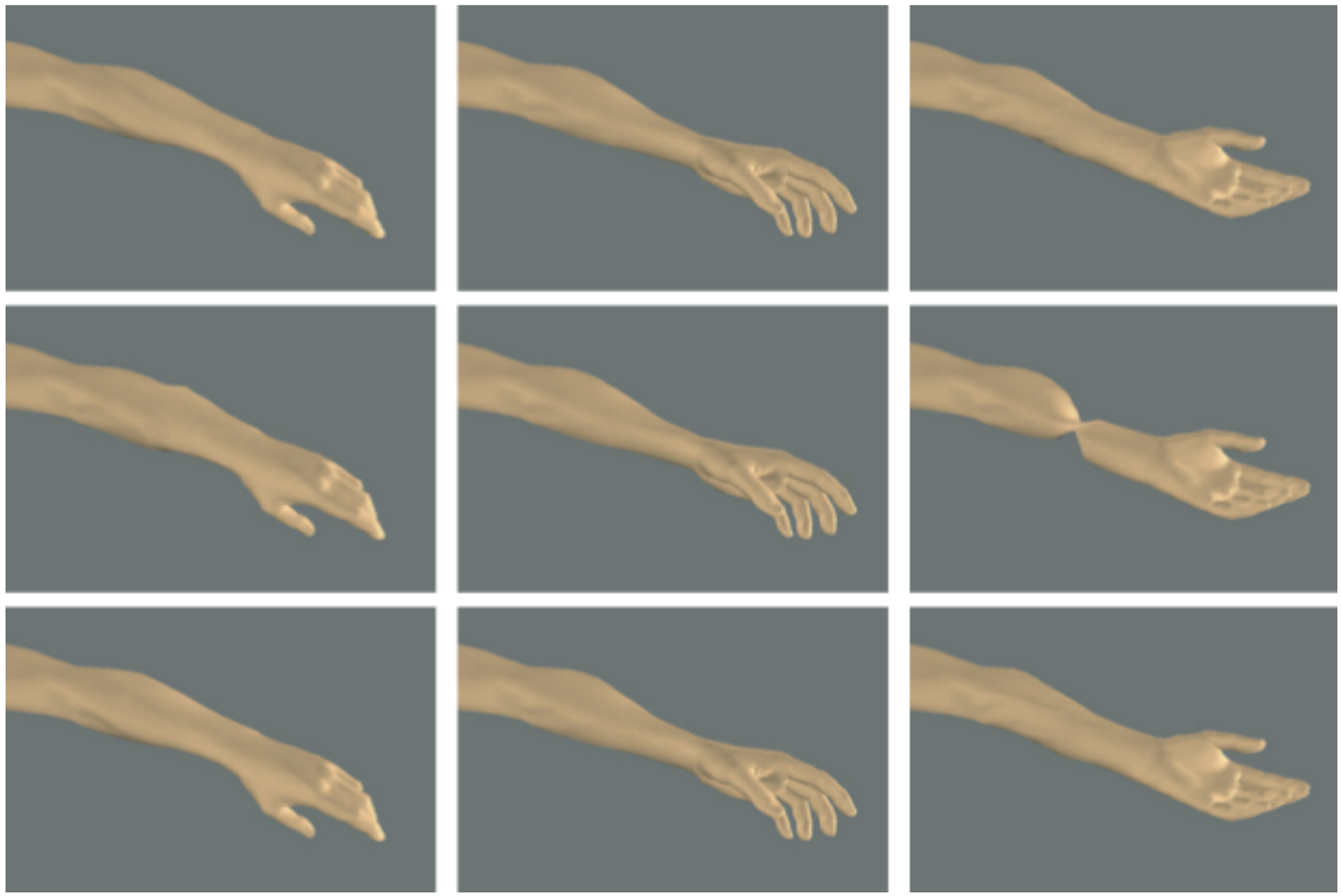


Mohr Technique

- Key Idea: add more joints
  - New rotation joints
  - New bulging joints (by groups of 4)

Mohr Technique - Results

- Original (captured)
- Linear Blend
- Mohr



Mohr Technique - Results

Original Examples

Linear Blend Skin

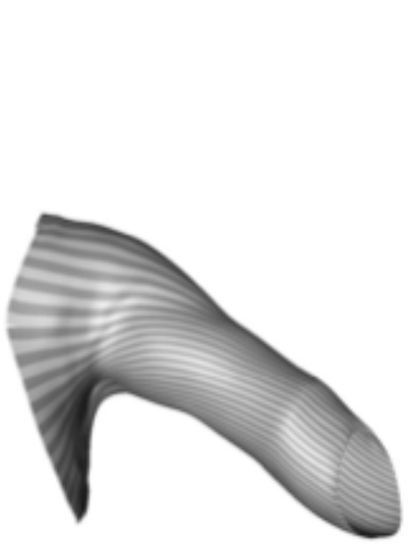
Our Method

Wang Technique

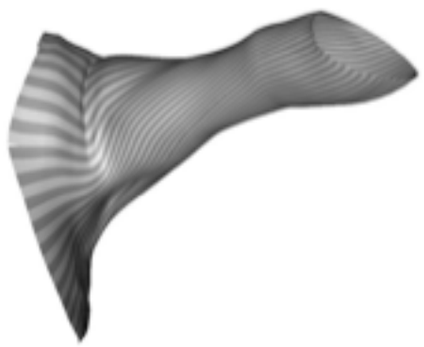
- Key Idea: add more weights
  - 12 weights per vertex per joint instead of 1
  - One weight for each dimension of the Tf matrix
  - > enables scaling (muscle bulging)

- Pb: examples must be created by hand or skin mocap

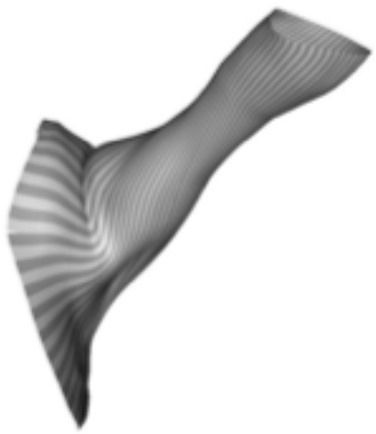
Wang Technique - Results



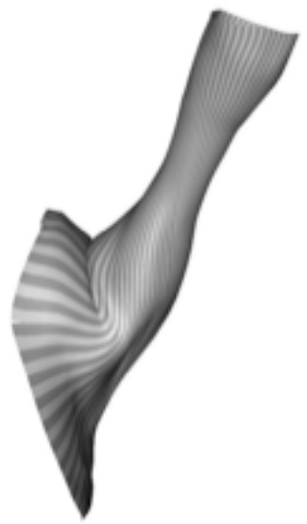
3-a: SWE frame 1



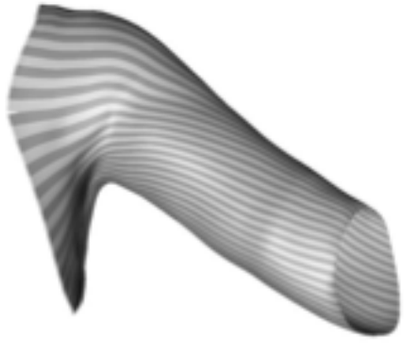
3-b: SWE frame 2



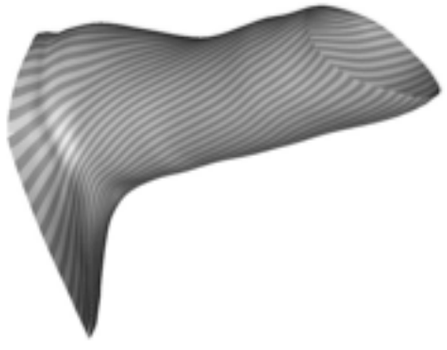
3-c: SWE frame 3



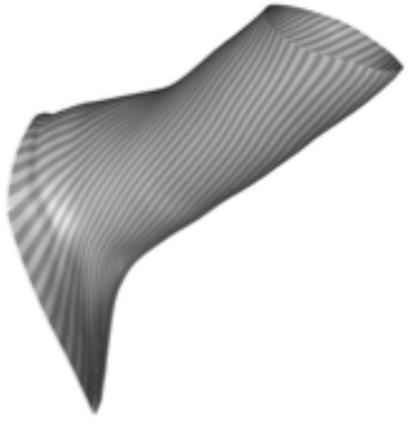
3-d: SWE frame 4



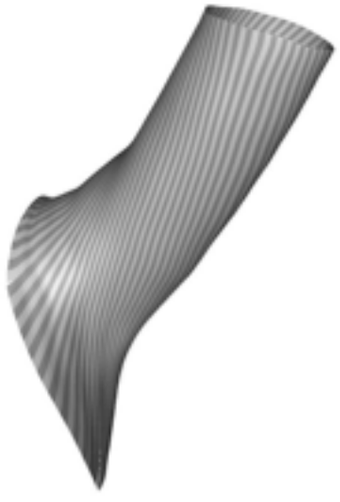
3-e: MWE frame 1



3-f: MWE frame 2



3-g: MWE frame 3



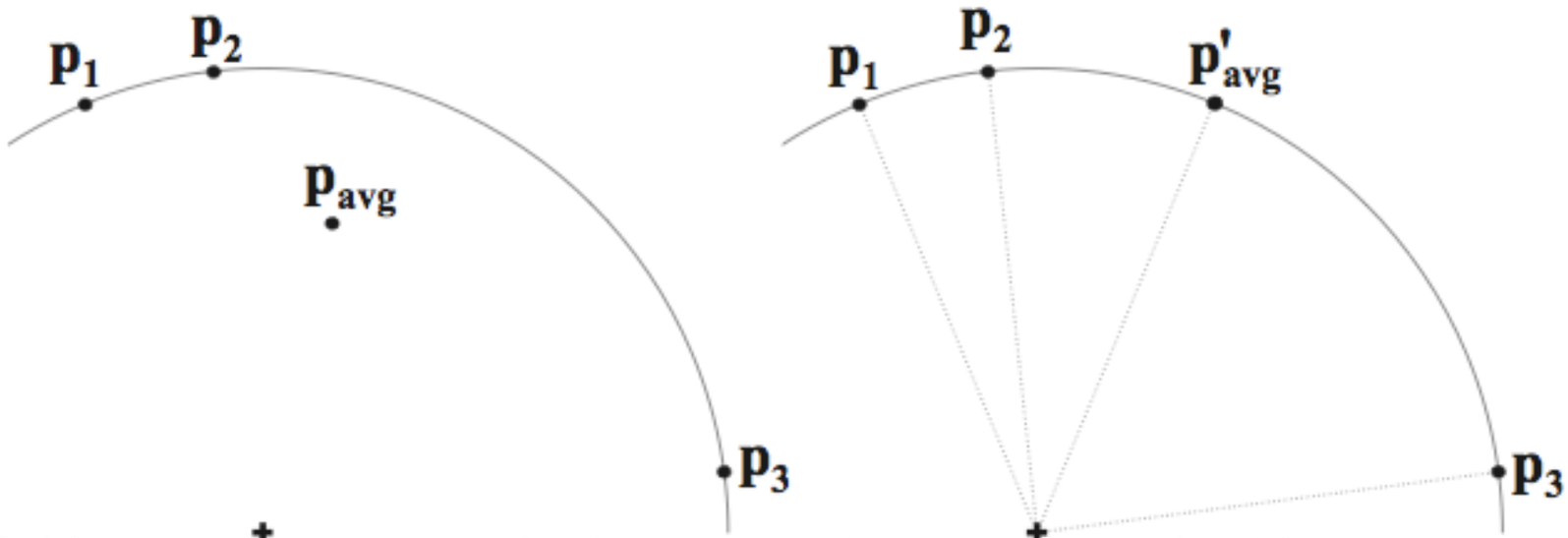
3-h: MWE frame 4

Non-Linear Blend Skinning

- Radial interpolation [Kavan 05]
- Dual quaternions [Kavan 08]

Spherical Blend / Dual Quaternion Skinning

- Idea: blend the rotations and translations independently



- Difficulties: find the appropriate center of rotation
  - Spherical Blend (slow & bad center of rotation)
  - Log Matrix (rotations are scaled -> bad interpolation)
  - DLB (Dual Quaternion Linear Blend)

Dual Quaternion

- Non-linear interpolation
- Solves the loss of volume problem
- May create too much volume
- Uses the same skinning weights as for classic linear skinning

Results

- Videos

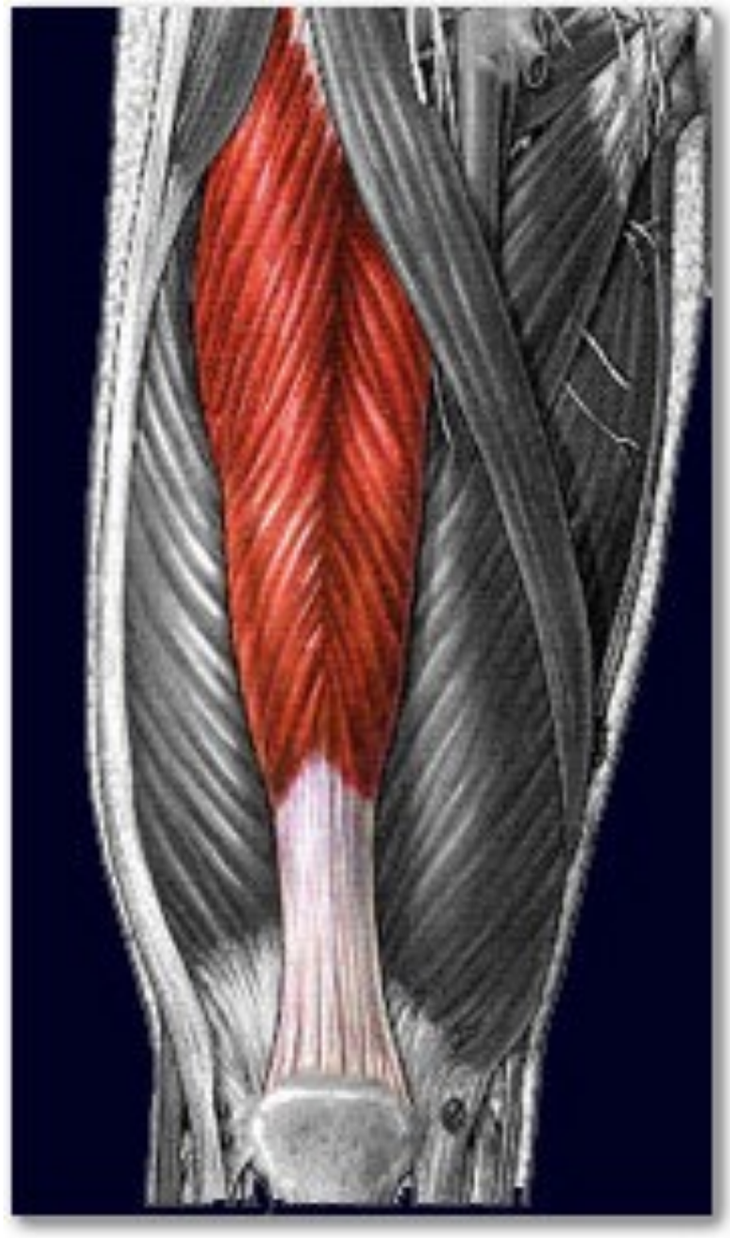
Dynamic Skinning

Larboulette et al. Dynamic Skinning: Adding Real-time Dynamic Effects to an Existing Character Animation, 2005.

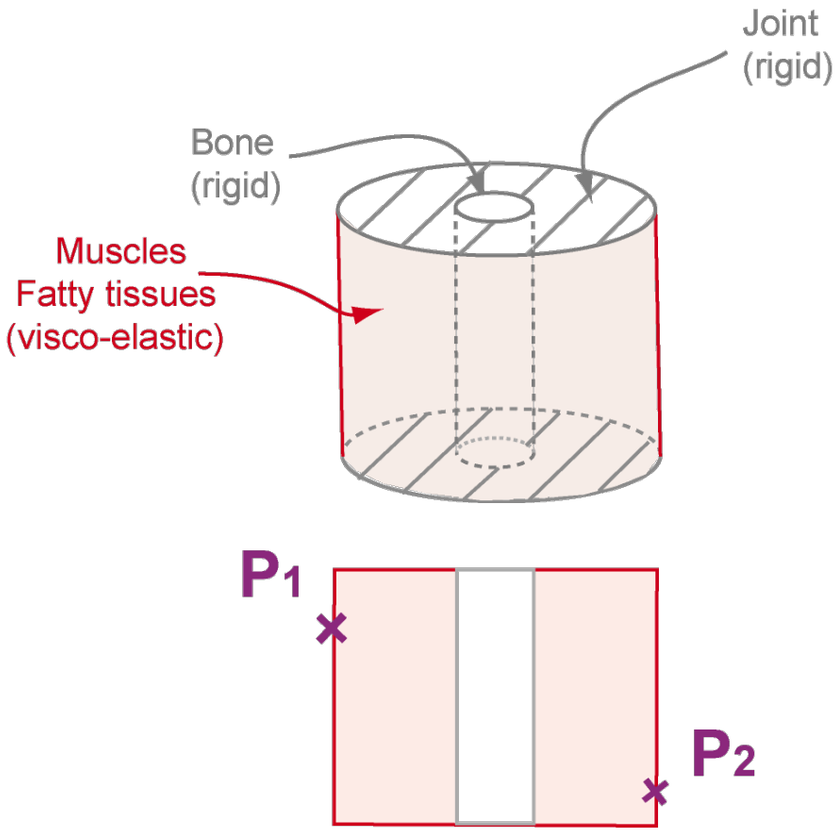
- Principle
  - Approximation of the deformation with one spring
  - Uses the traditional animation pipeline (skeleton + layers)
  - Based on skinning and FEM theory

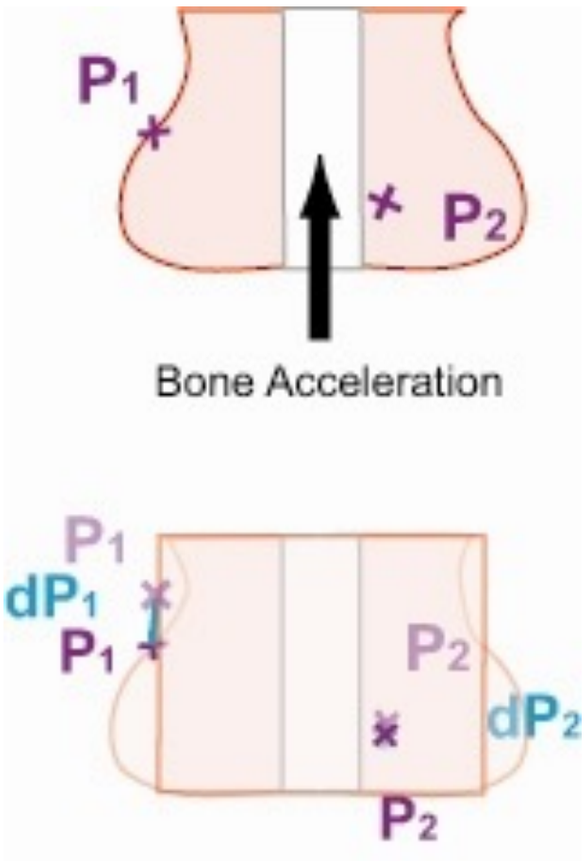
Inertia of tissues : constraints

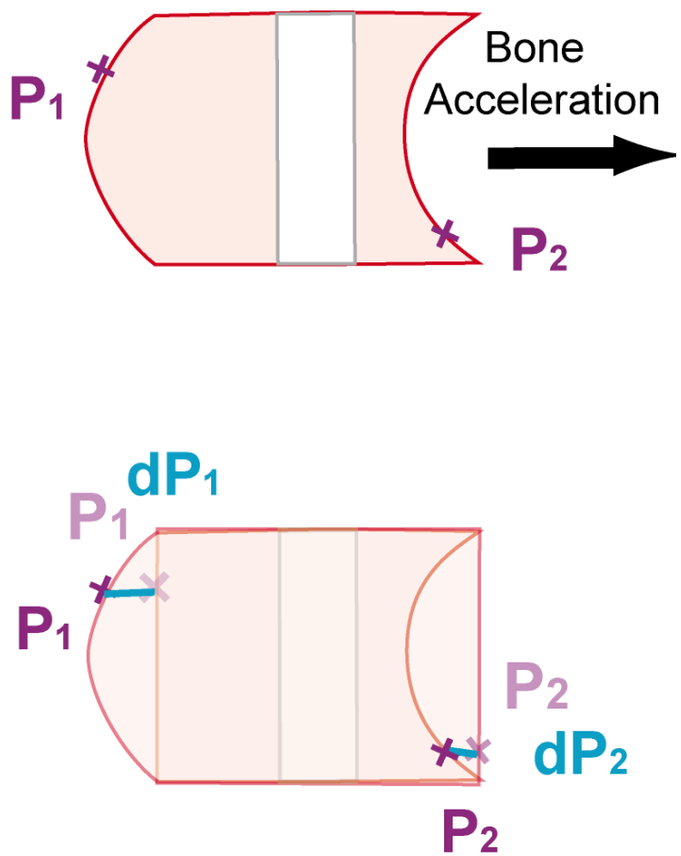
- Bone surrounded by
  - Muscles
  - Fatty tissues
  - Skin
- Elastic Vibrations
  - Initiated by bones
  - Local effect with local properties
- Need for a physical model
- Real-time deformation
- Need for a simplified model



Approximation of the deformations

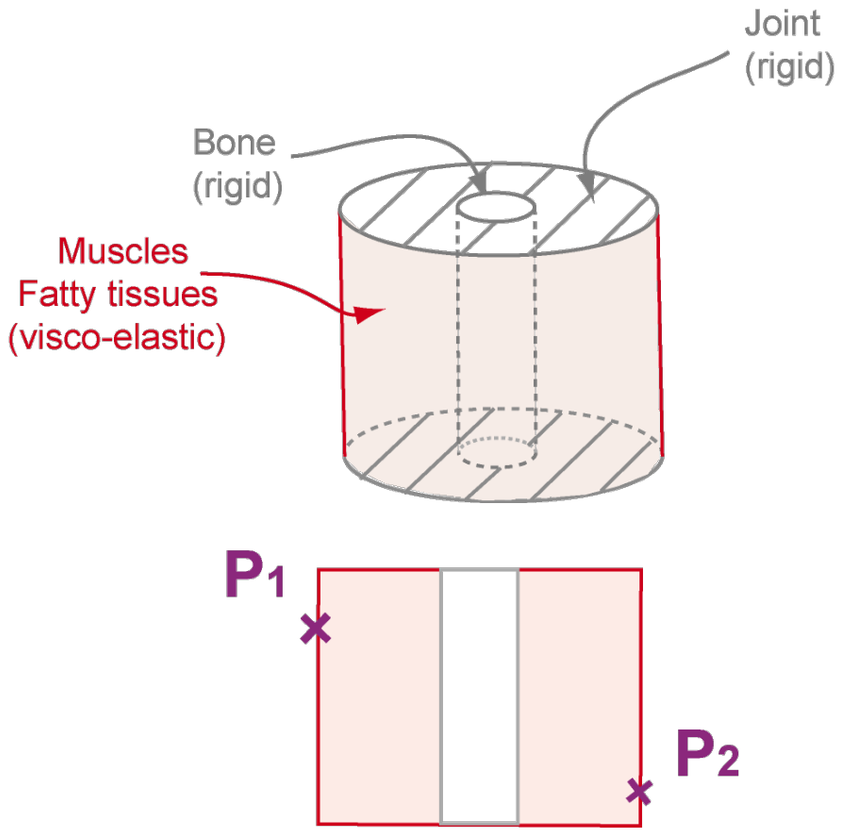


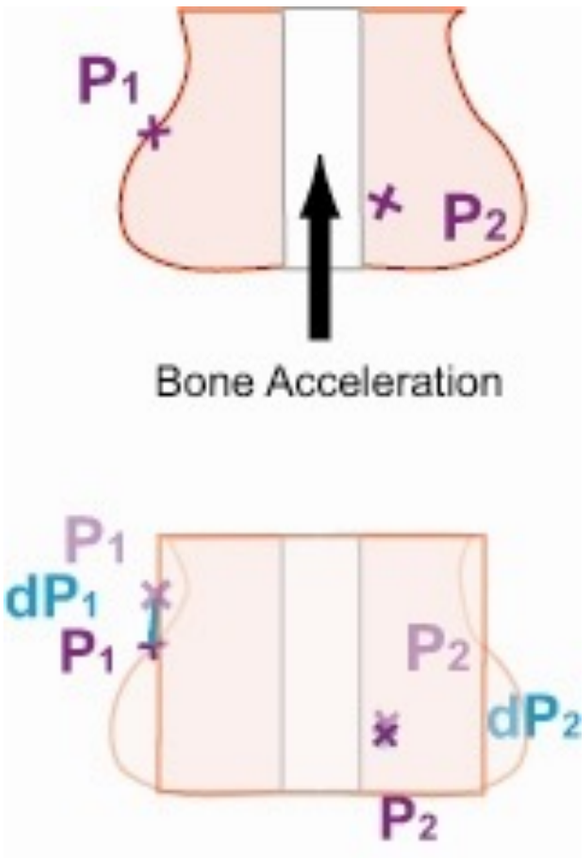


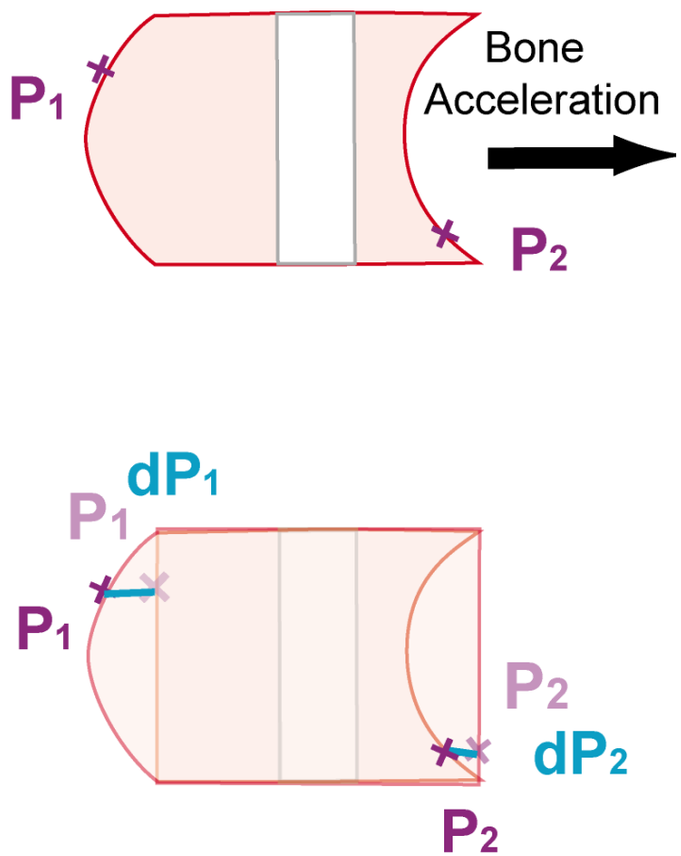


- The further the point the stronger the deformation

Approximation of the deformations

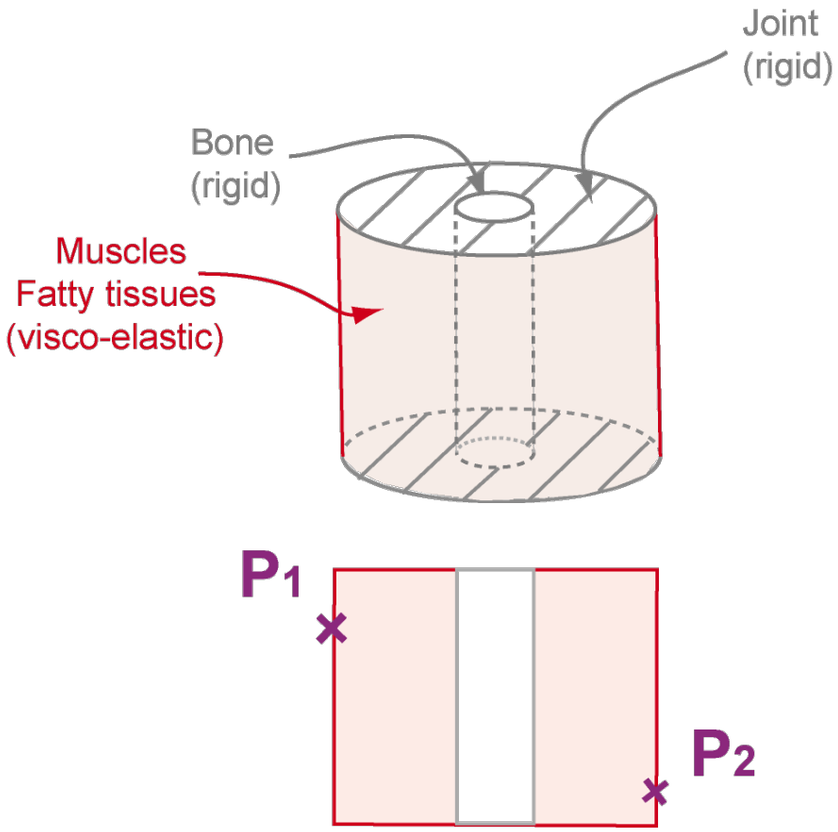


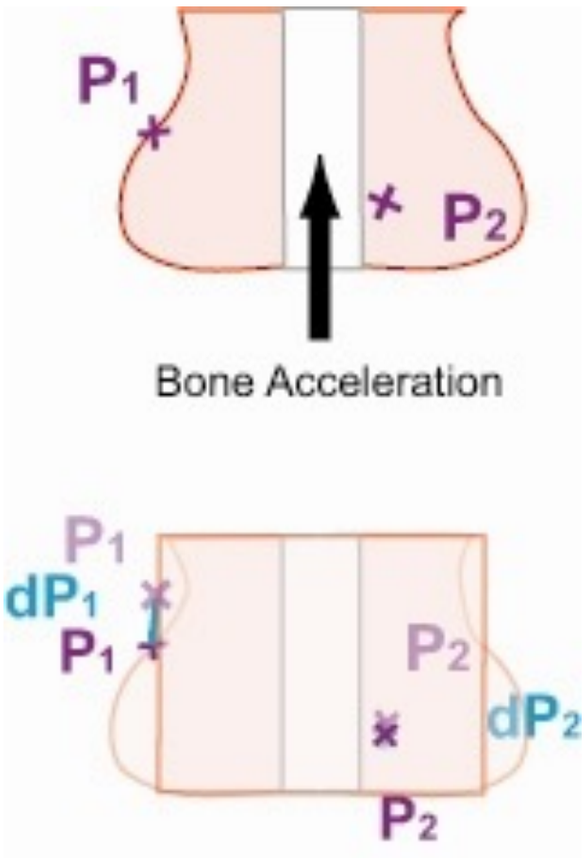


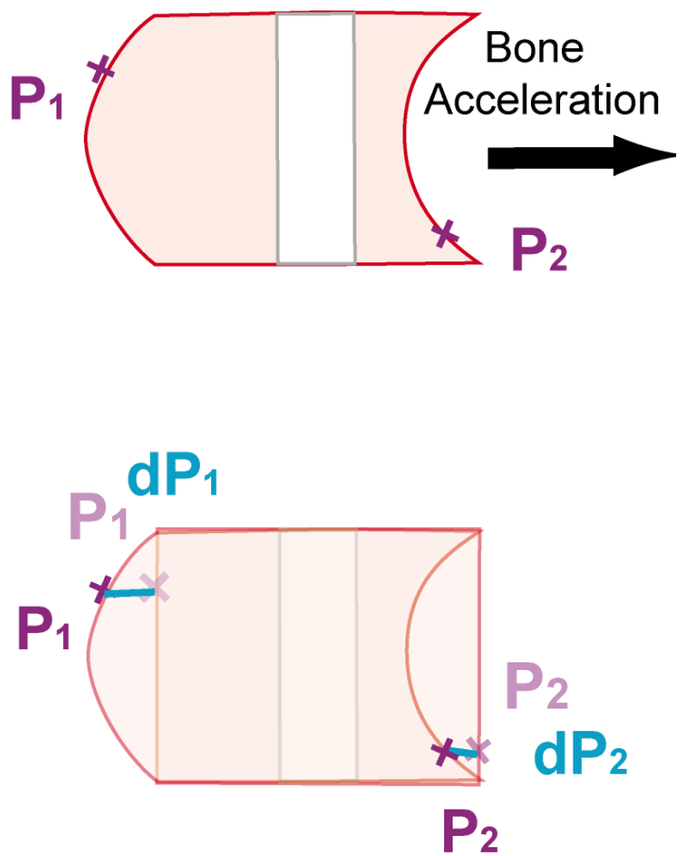


- No deformation at both ends

Approximation of the deformations







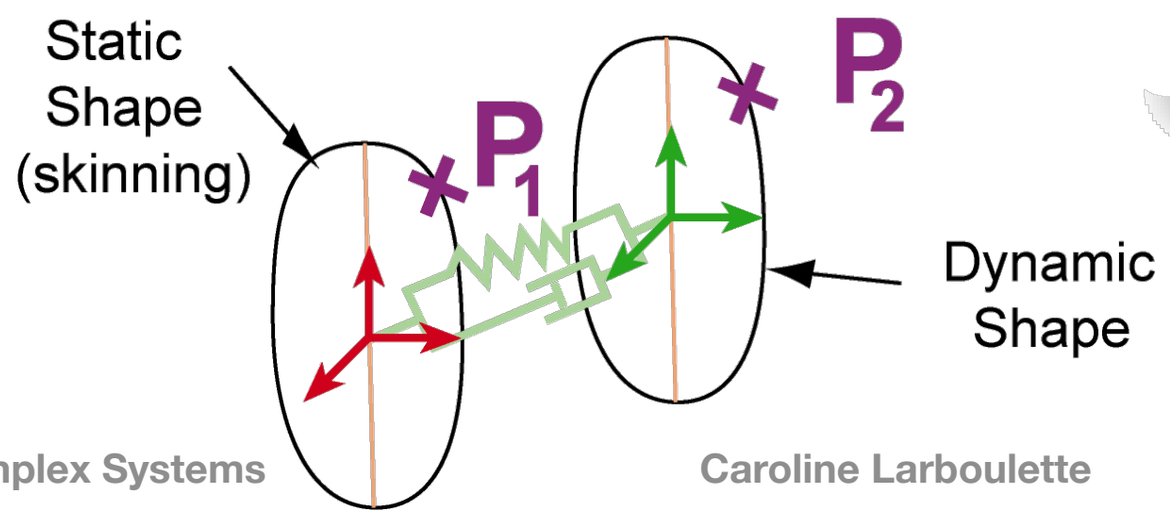
- Points are not allowed to cross bone areas

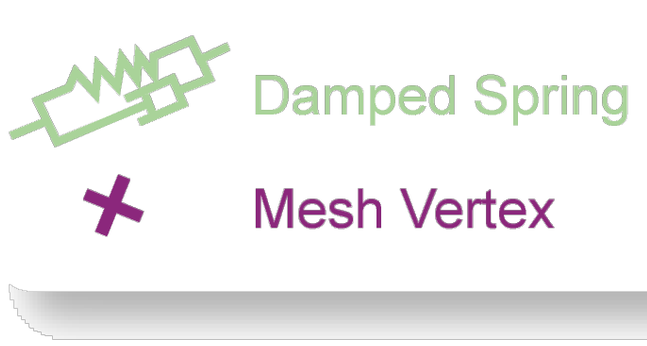
Dynamic Skinning

Principle

- Flesh element
  - A set a mesh vertices defining a volume with homogeneous behavior
  - One damped spring
  - A unique weight for each mesh vertex

●=> Blending operation

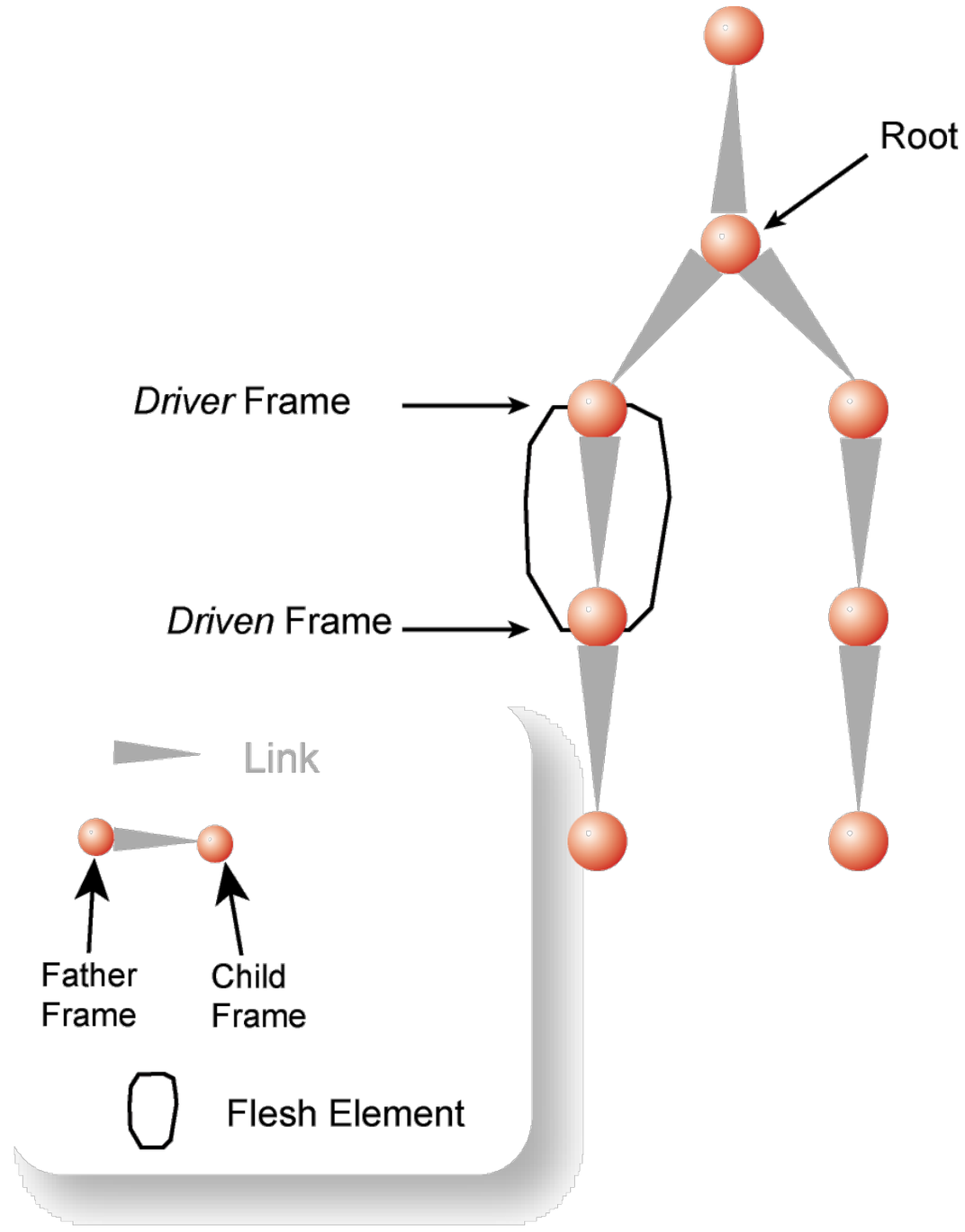




Flesh Element: Specification

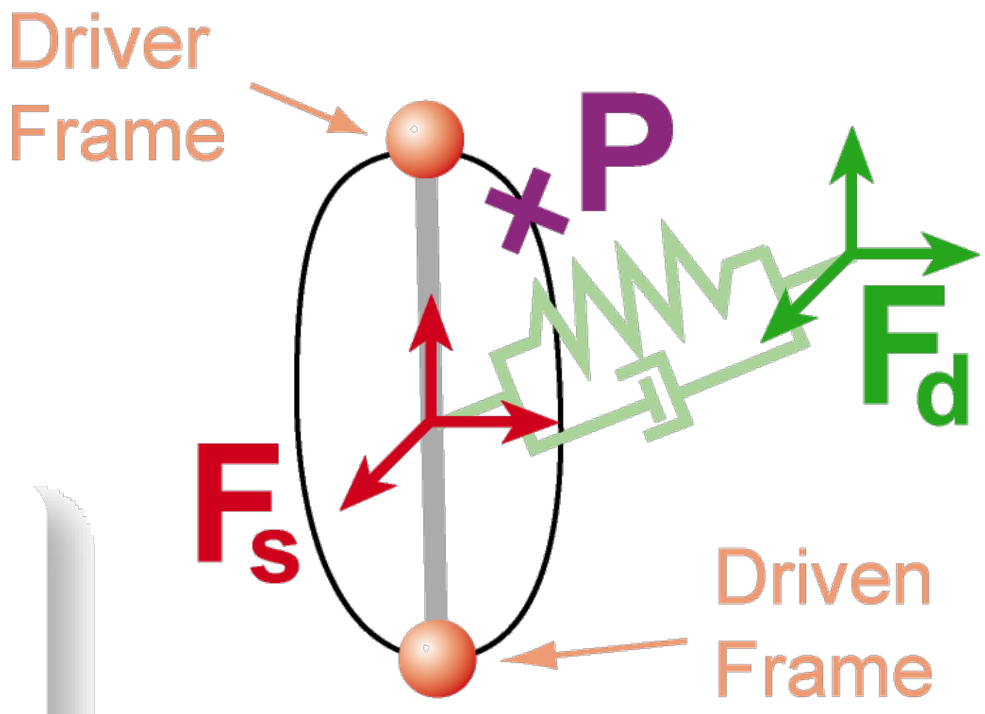
- A set of vertices
- 2 frames:
  - Driver Frame (father)
  - Driven Frame (child)

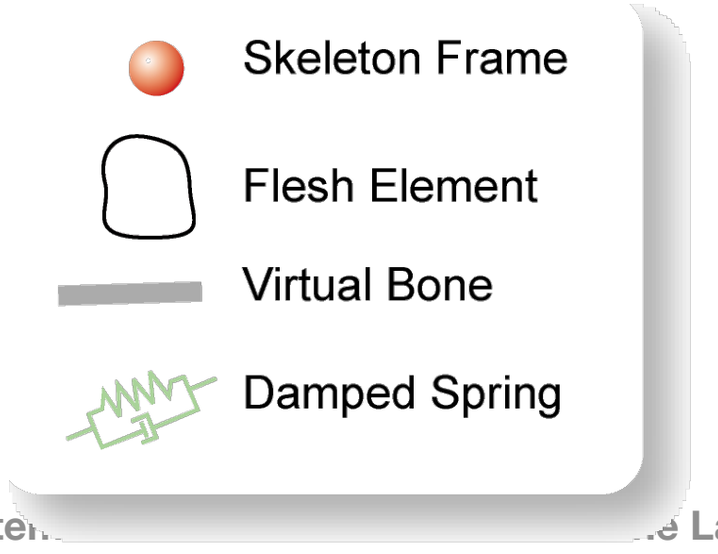
- Define
  - A virtual bone
  - A local frame Fs



Dynamics of the Flesh Element

- Dynamic Frame Fd
  - Attached to local frame Fs through a spring
- Damped spring
  - $L_0 = 0$
  - Mass concentrated on Fd
  - Stiffness & damping coefficients





Numerical Integration

- Euler explicit
- Forces taken into account
  - Gravity
  - Air friction ?

- Gives a new position for Fd
- Maximal elongation
- (to prevent crossing of bone areas)
  - If $L > L_{max}$ then $L = L_{max}$

Deformation of the mesh

1. Skinning -> static deformations
2. Dynamic Skinning
   - Compute Fd position
   - Make a linear interpolation

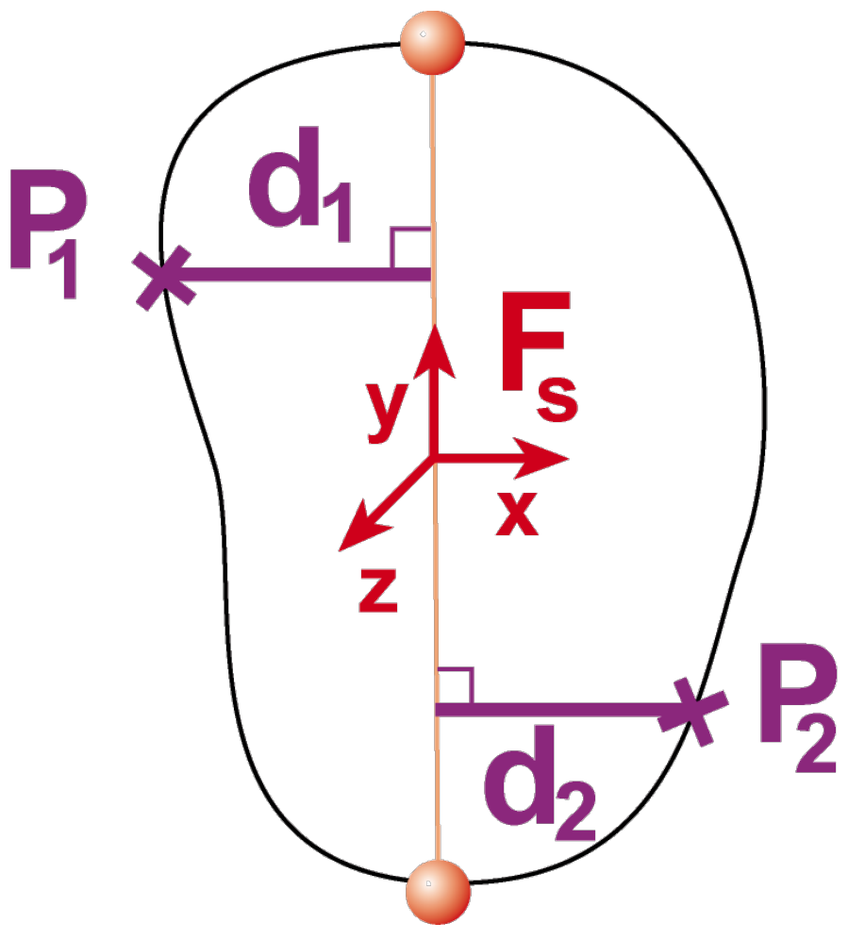
$$P_{new} = \alpha_1(P) \cdot P_1 + \alpha_2(P) \cdot P_2$$

=> Equals to applying a translation t to each vertex

$$\vec{t} = \alpha_2(P) \cdot \overrightarrow{F_s F_d}$$

Weights (1) – Shape

- "The further the point, the stronger the deformation"

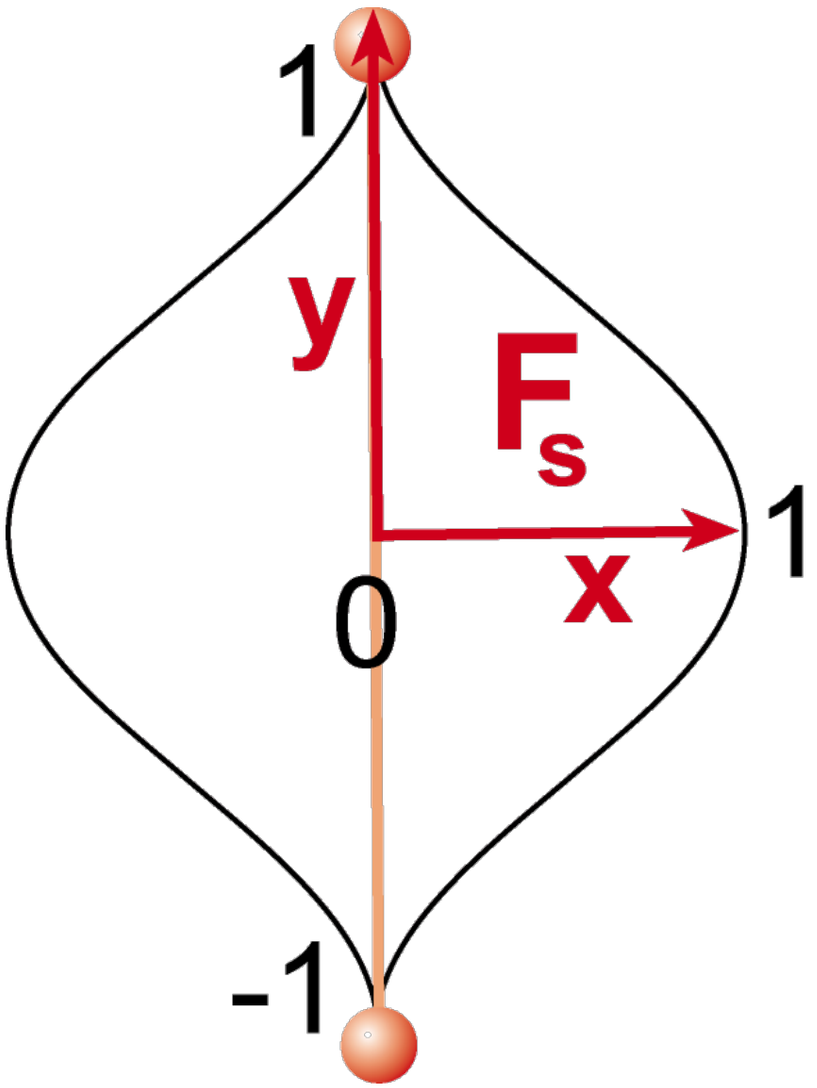


$$shape(x,z) = d(P, Bone)$$

- d = Euclidean distance
- Linear function

Weights (2) – Attenuation

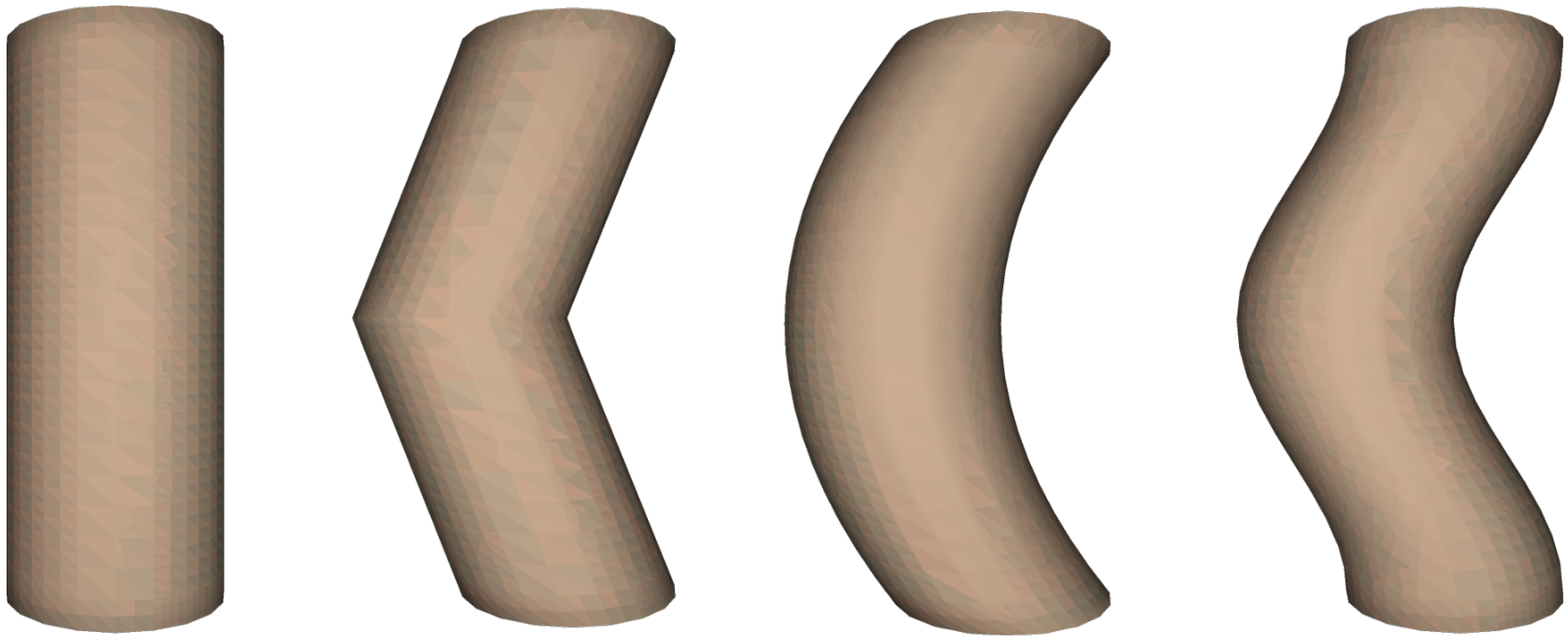
- "No deformation at both ends"



- attenuation(y) = 0 for y = -1
- attenuation(y) = 1 for y = 0
- attenuation(y) = 0 for y = 1

$$\text{attenuation}(y) = 1 + \frac{-4y^6 + 17y^4 - 22y}{9}$$

Different Attenuation Functions



Linear Parabola Wyvill

- **Requirements:**
  - **C1 continuity**
  - **Tangents equal to 0 at both ends**

Normalization & Maximal Elongation

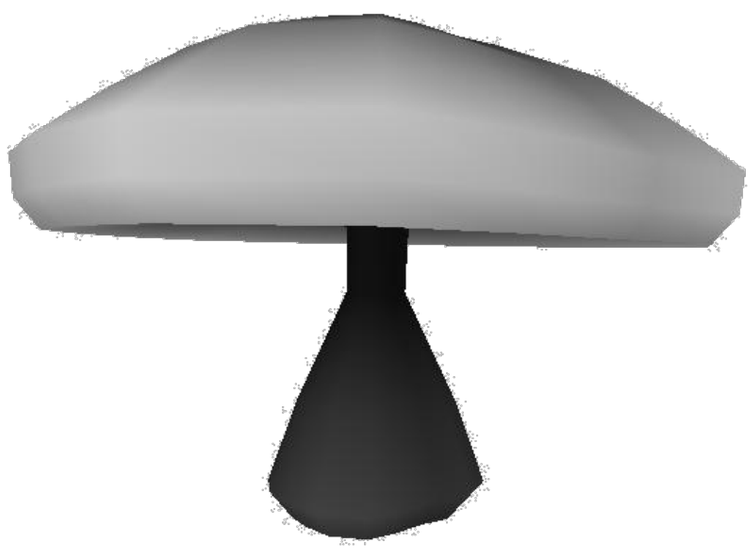
- "Weights between 0 and 1"

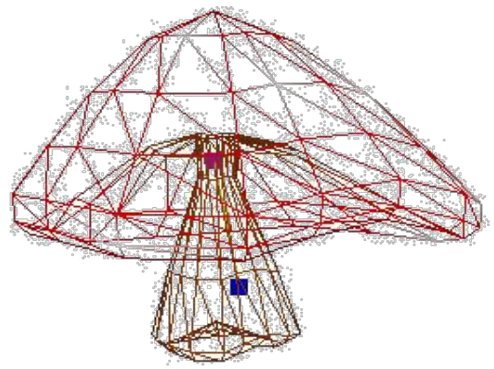
$$\alpha_2(x, y, z) = \frac{shape(x,z).attenuation(y)}{\max(shape.attenuation)}$$

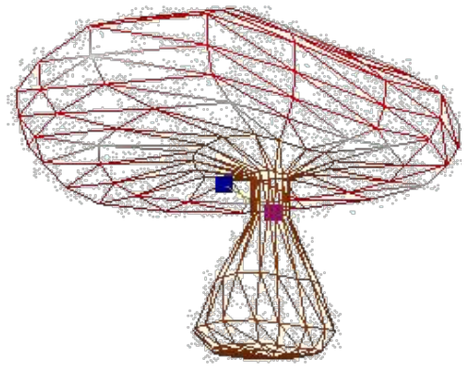
- "Flesh cannot cross bone"

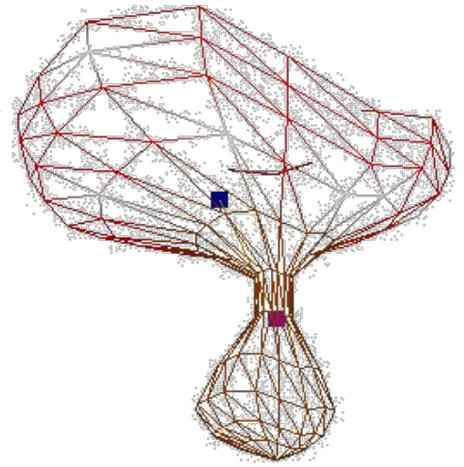
- Max_elongation = max (shape.attenuation)

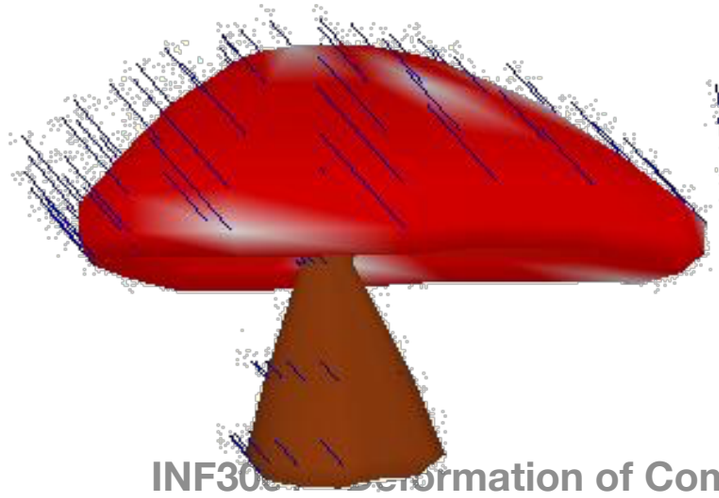
Results: the mushroom

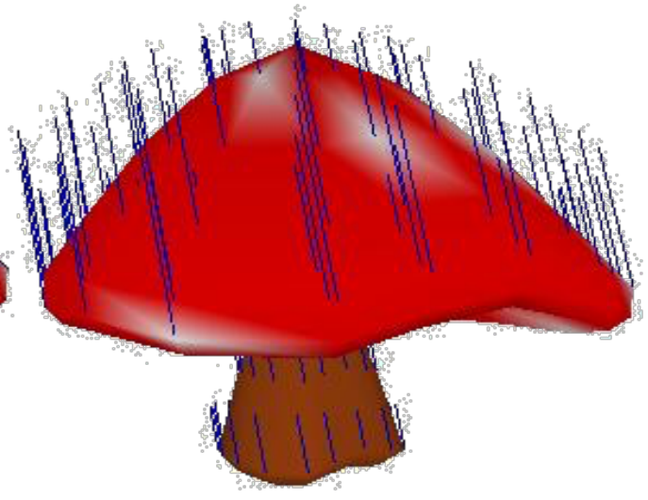


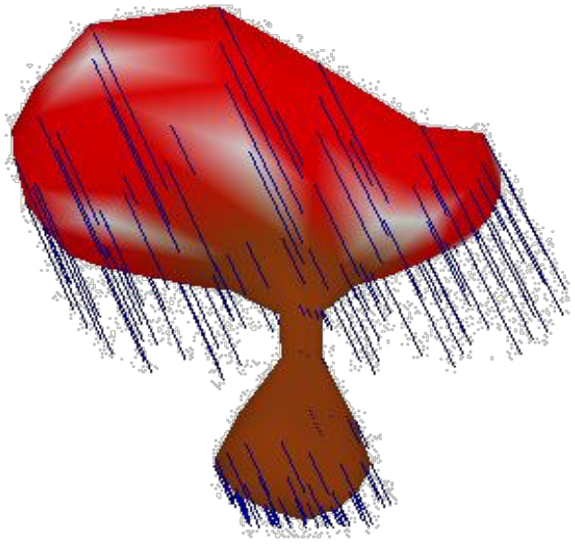




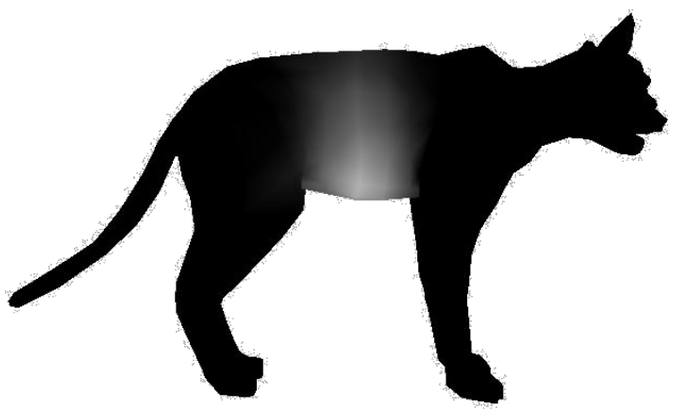


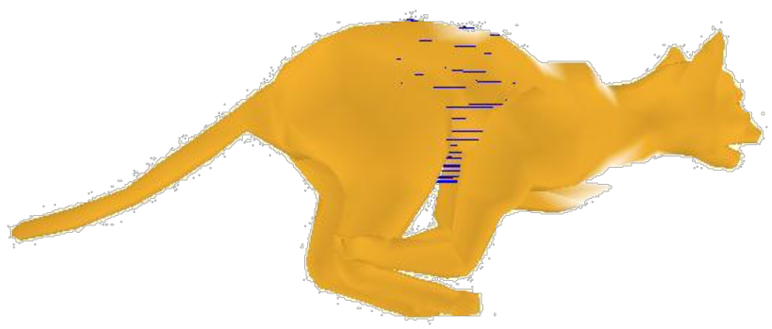


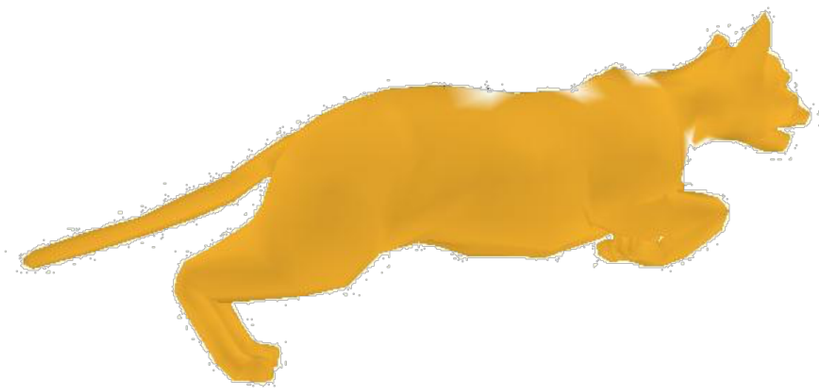


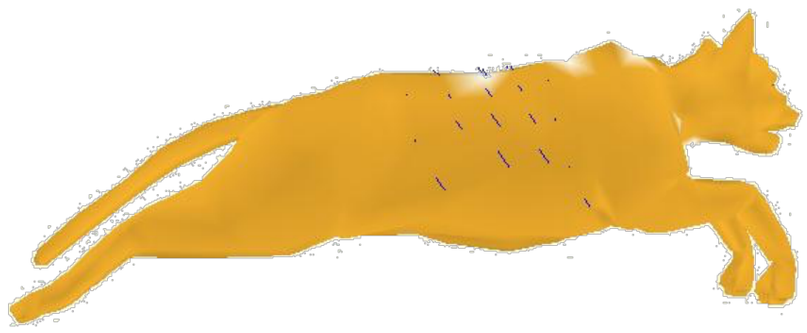


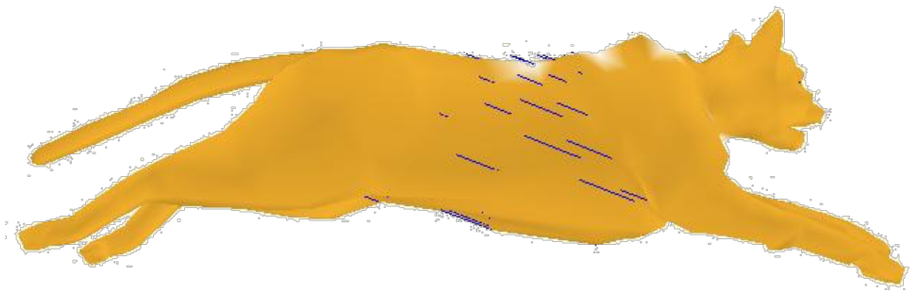
Results: a running cat

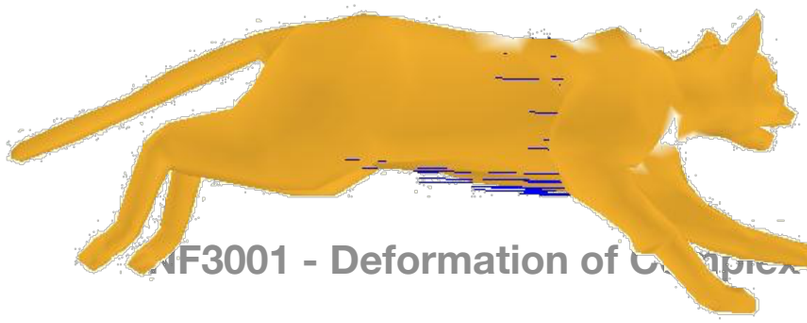


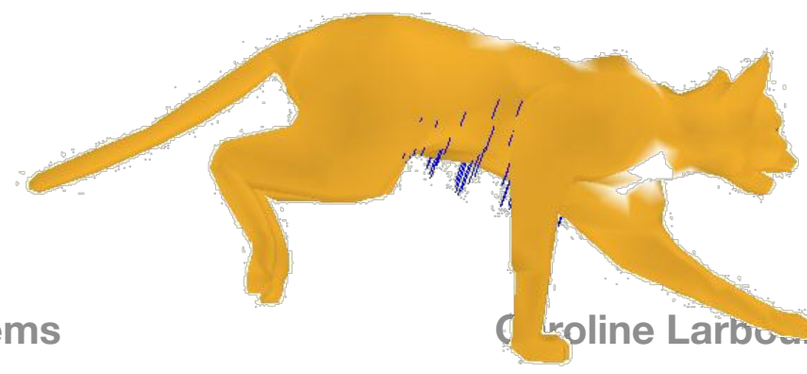


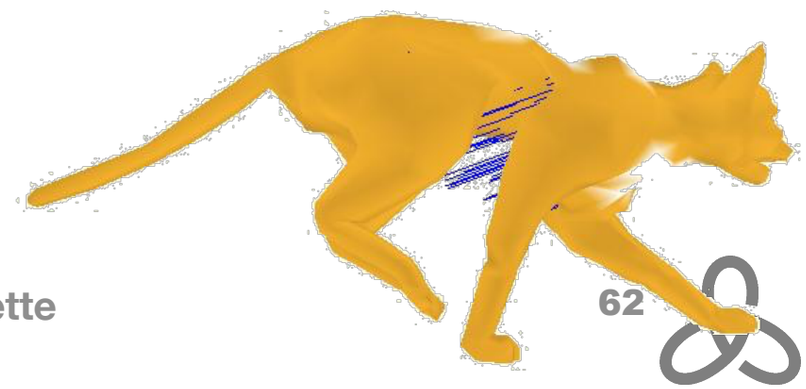












Videos

Dynamic Skinning

- Advantages
  - Add dynamics
  - Real-time (only one spring per element)
  - GPU compatible
- Drawbacks
  - Only translations

Anatomically Based Animation

Aim

- Getting more realism by being more accurate (biologically speaking)
- Modeling of individual
  - Bones
  - Muscles
- Covered by an elastic skin

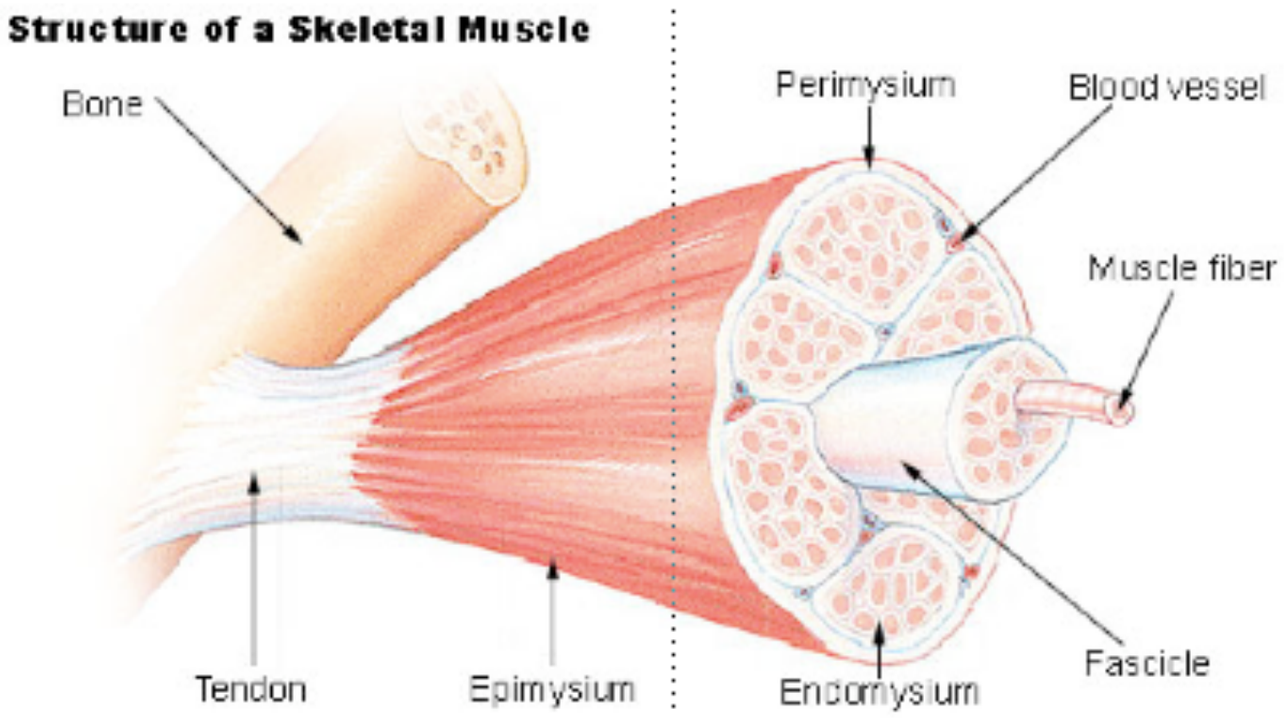
- Not all models include all layers

Real Life / Animation

- In real life: muscles contract to make the bones move
- In animation: the bones move, the muscles are deformed to make the skin bulge

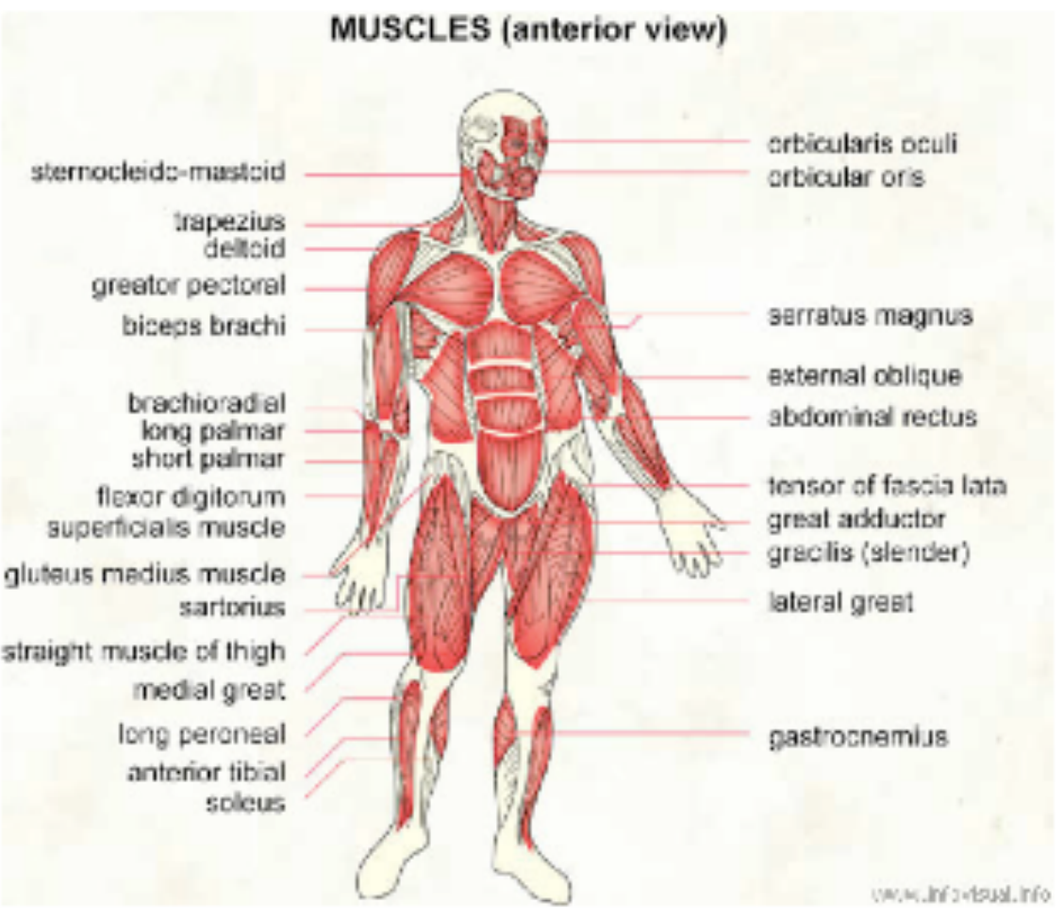
Human Anatomy - Muscles

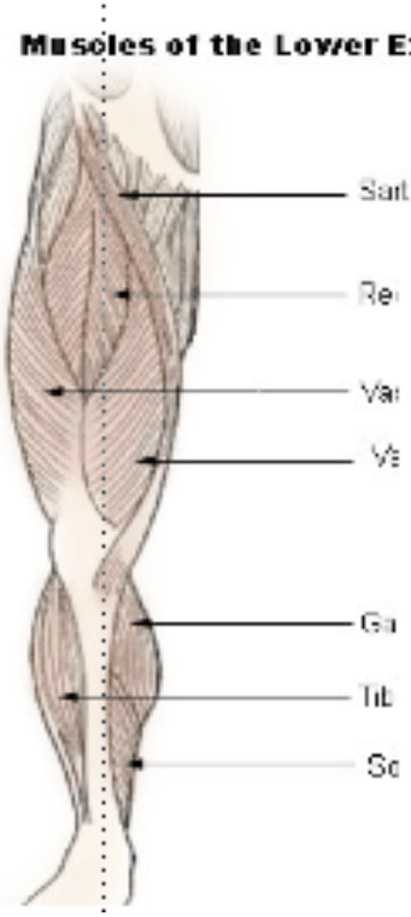
- Skeletal muscles: move the bones
- Fusiform muscles are composed of
  - A belly
  - An origin or proximal insertion (static)
  - A distal insertion or insertion (moving part)

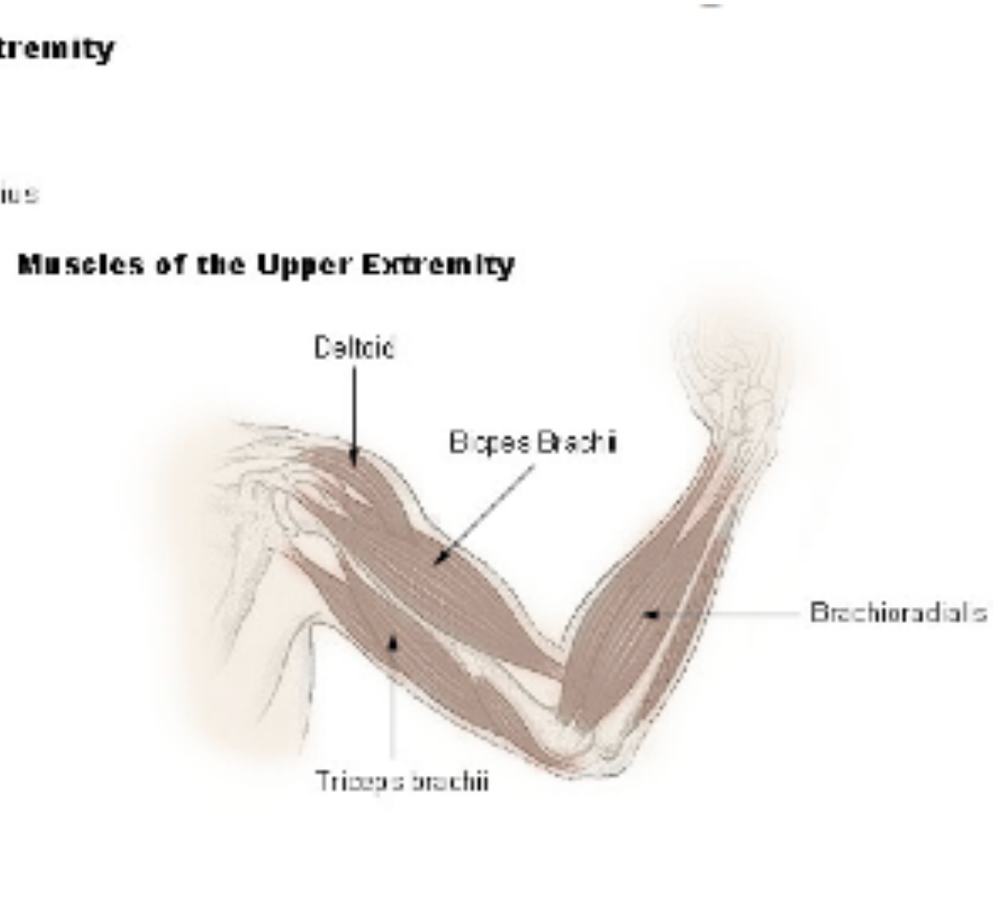


Human Anatomy - Muscles

- Tendons may attach the muscle to the bone
- Human body: 639 muscles
- 200 muscles are activated when walking







Muscles - Contraction

- Two types of contraction
  - Isometric contraction (same length): the muscle contracts, the belly changes but not the length - no movement
  - Isotonic contraction (same tonicity): the belly changes, the length changes - movement of the bones
- In real life, both types of contraction happen at the same time
- Isotonic more visible, more important in animation

Physical Properties

- Elasticity

$$E = (F \times L \times k) / A$$

E: elongation of the muscle
F: force applied
L: initial length of the muscle
k: elasticity parameter (const)
A: section of the muscle

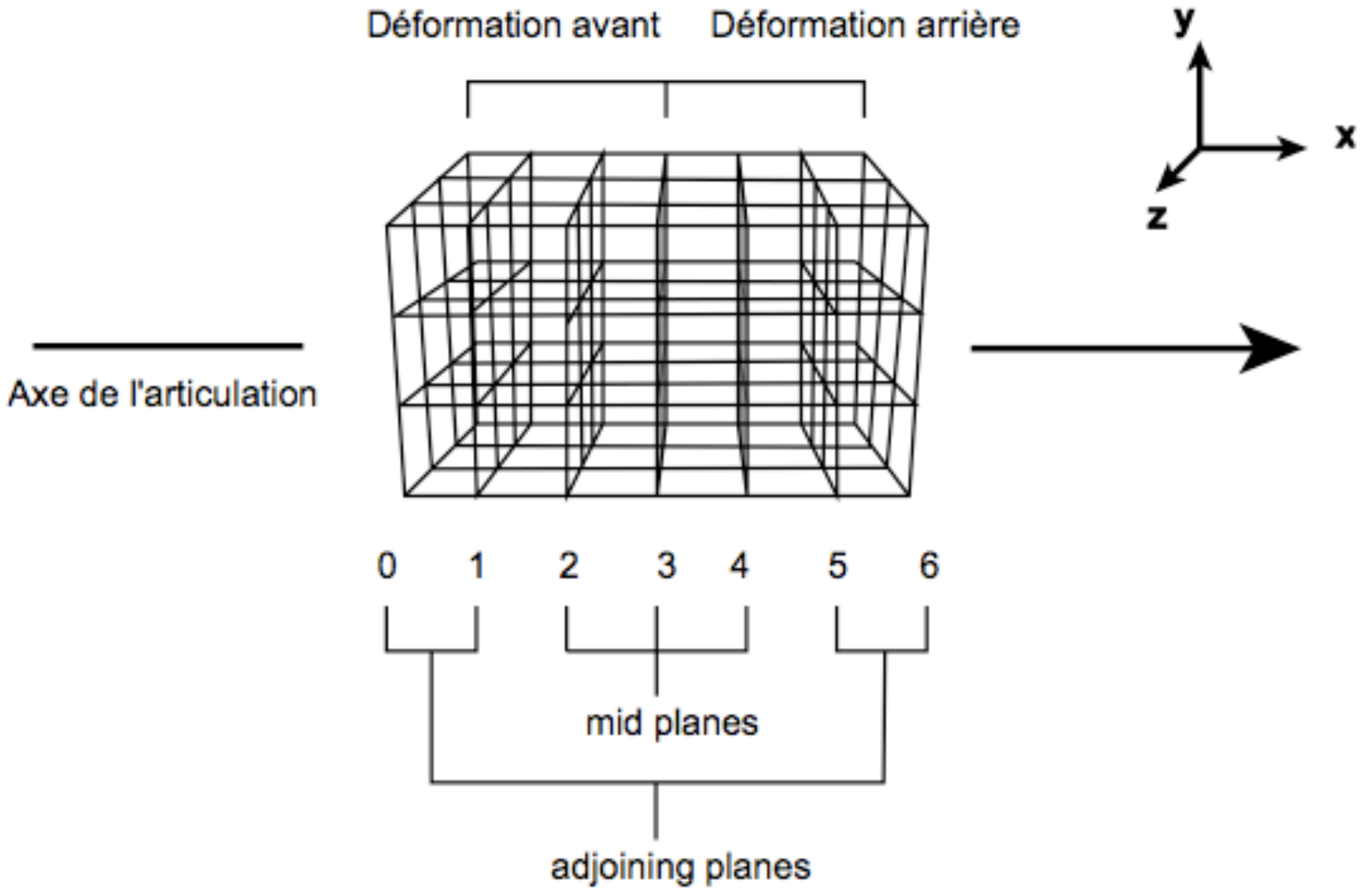
- Contractibility

$$R = \theta \times k$$

R: shortening of the muscle
θ: rotation of the joint

FFD Model

- Model: polygonal mesh inside 2 FFD grids



FFD: Belly Deformation

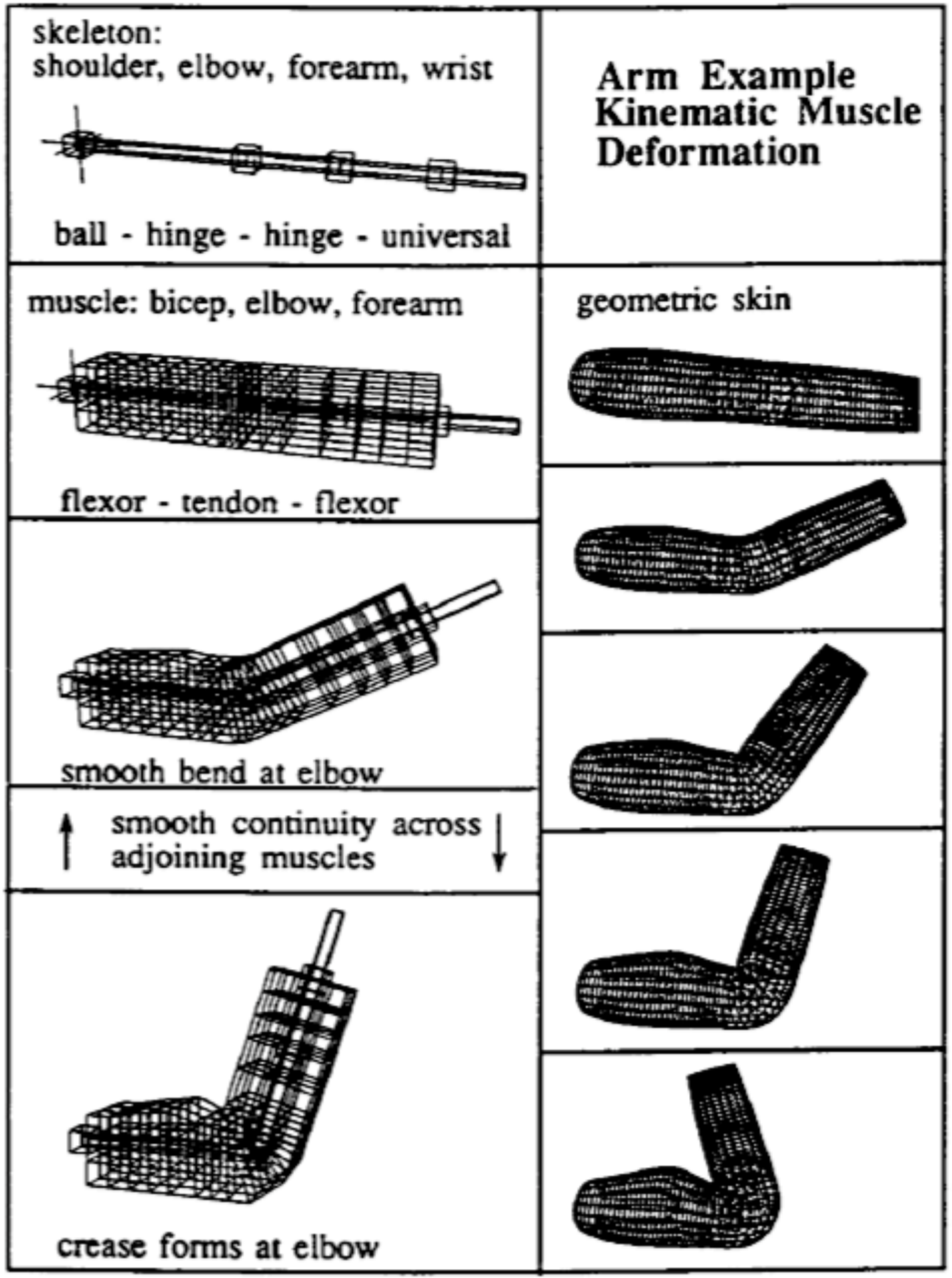
Muscle contraction (uses contractibility equation)

1. **Computation of the shortening R (depends on θ)**
2. **Propagation of the shortening to the mid-planes (if x↘, y↗ & z↗)**
3. **Deformation of the adjoining planes to ensure continuity**

FFD: Tendons Deformation

1. Compute angle = rotation angle / 2
2. Move mid-planes 2 and 3 by angle around z and mid-plane 4 by -angle
3. Slide adjoining planes away from mid-pl.
4. Rotate trailing deform by joint angle
5. Check for intersections and C1 continuity
6. Scale if needed to maintain volume

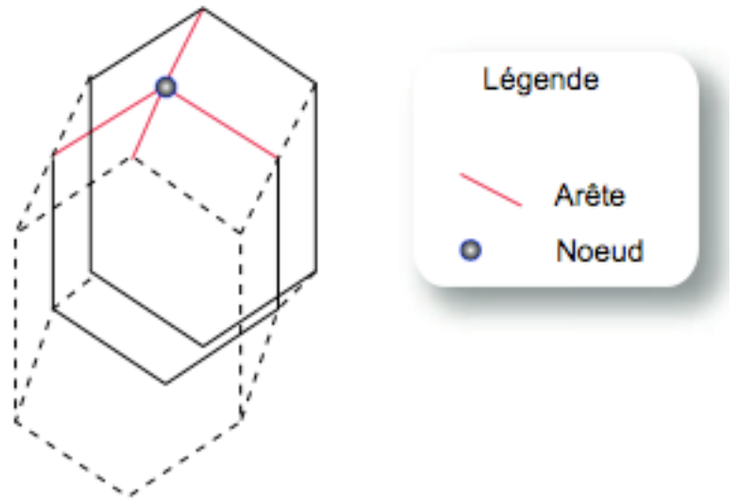
Results

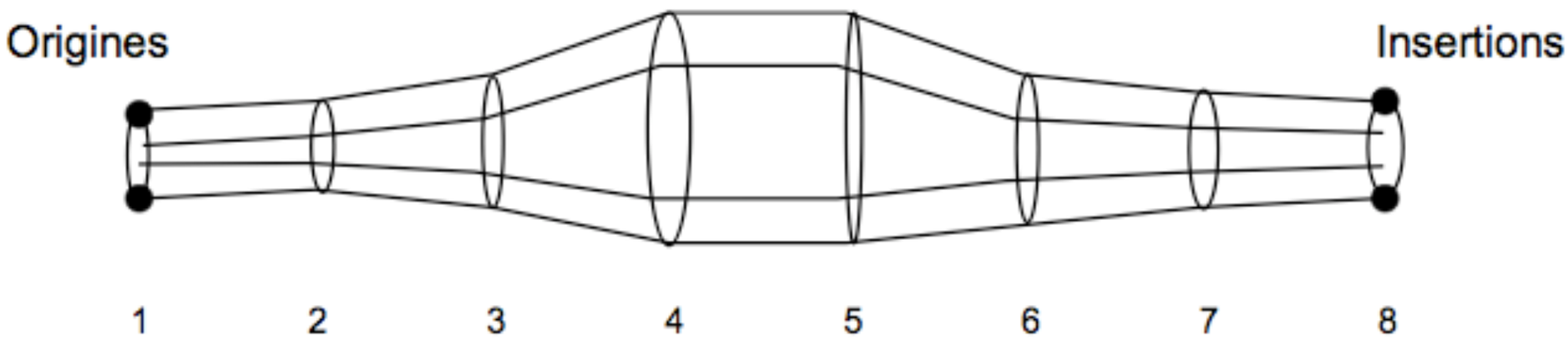

skeleton:
shoulder, elbow, forearm, wrist
ball - hinge - hinge - universal
Arm Example
Kinematic Muscle
Deformation
muscle: bicep, elbow, forearm
flexor - tendon - flexor
geometric skin
smooth bend at elbow
smooth continuity across
adjoining muscles
crease forms at elbow

Deformed Cylinder Model

Wilhelms at al. Anatomically Based Modeling, 1997

- One deformed (generalized) cylinder
  - Two origins
  - Two insertions
  - 7 sections (8 ellipses)
  - No tendons
  - Discretized for rendering





Deformed Cylinder Model

- Parameters controllable by the user
  - Origins and Insertions
  - Pivot
  - Size, position and translation of each section
  - Total length
- Parametrization makes it easy to retarget
  - Same muscles location
  - New parameters (length, origins…)

Deformed Cylinder Model

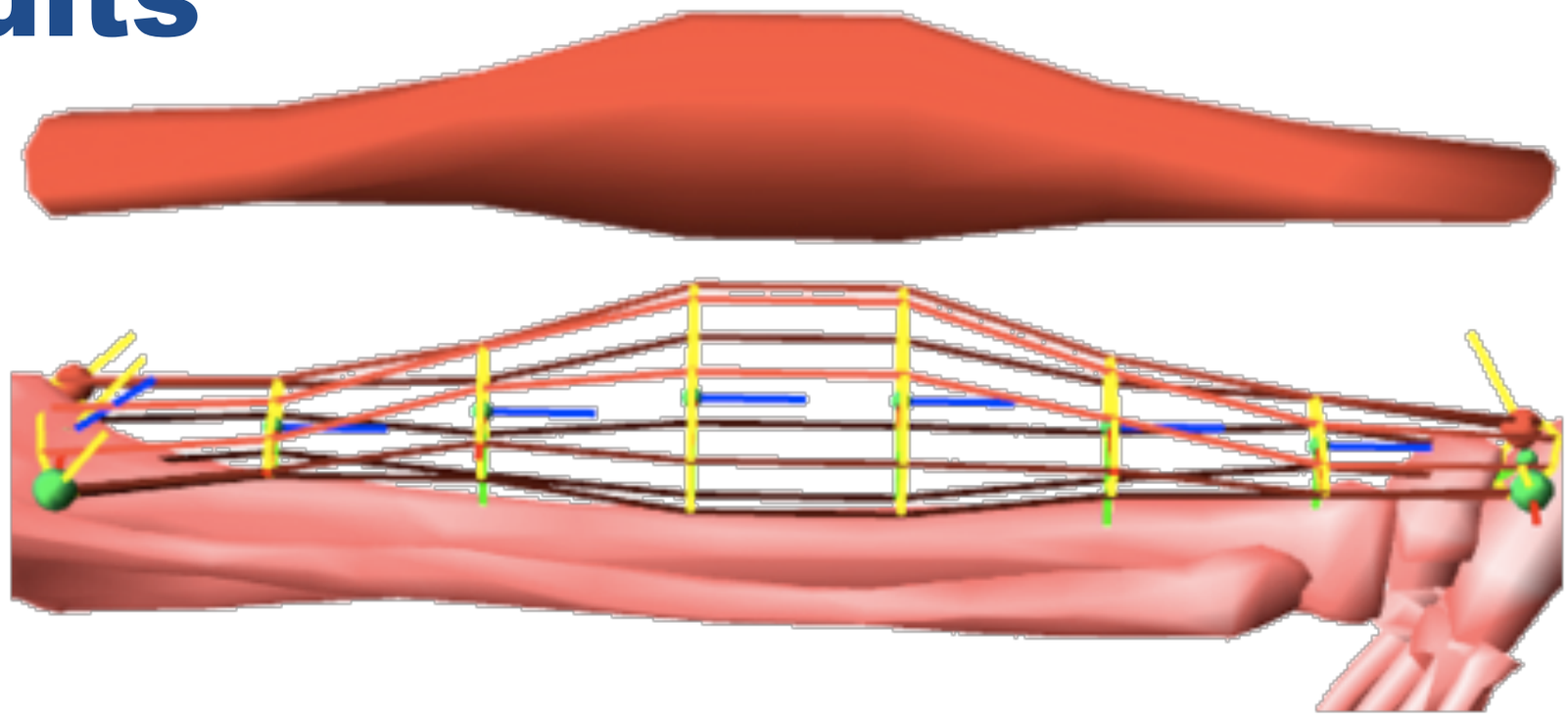
Animation

Rest length is the length of the muscle at rest (origins (to pivot) to insertions)

1. **Joints are deformed**
2. **Ellipses are scaled by sqrt(current length / rest length)**
   - **Volume exactly conserved between parallel sections**
   - **Volume changed a little otherwise (isometric ?)**

Deformed Cylinder Model

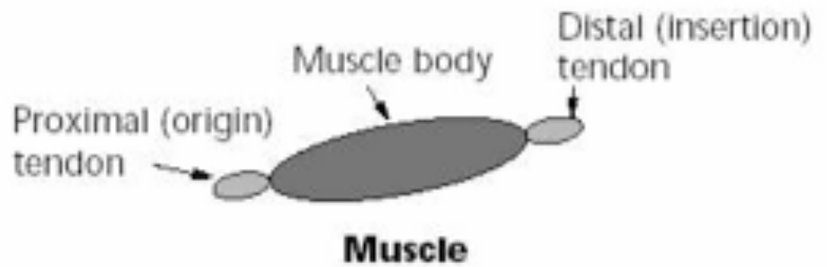
Results

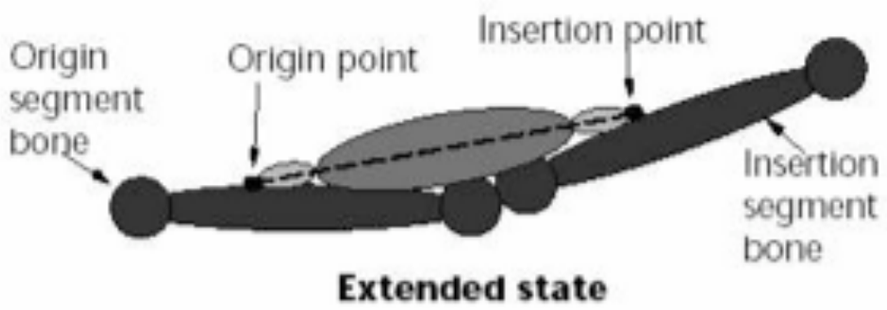


Ellipsoid Model

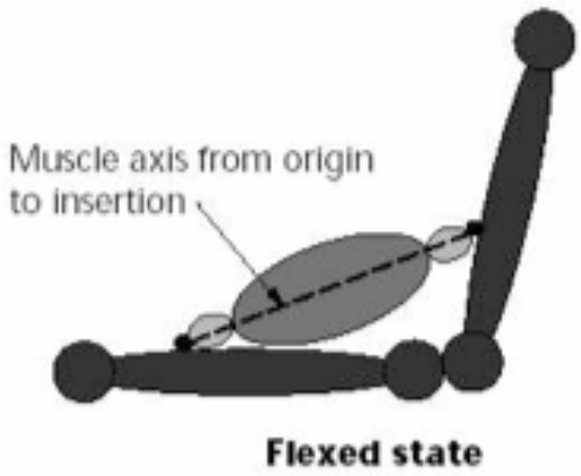
- **Fusiform muscle model**
  - **One ellipsoid with one origin and one insertion**

$$f(x,y,z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} + cste.$$



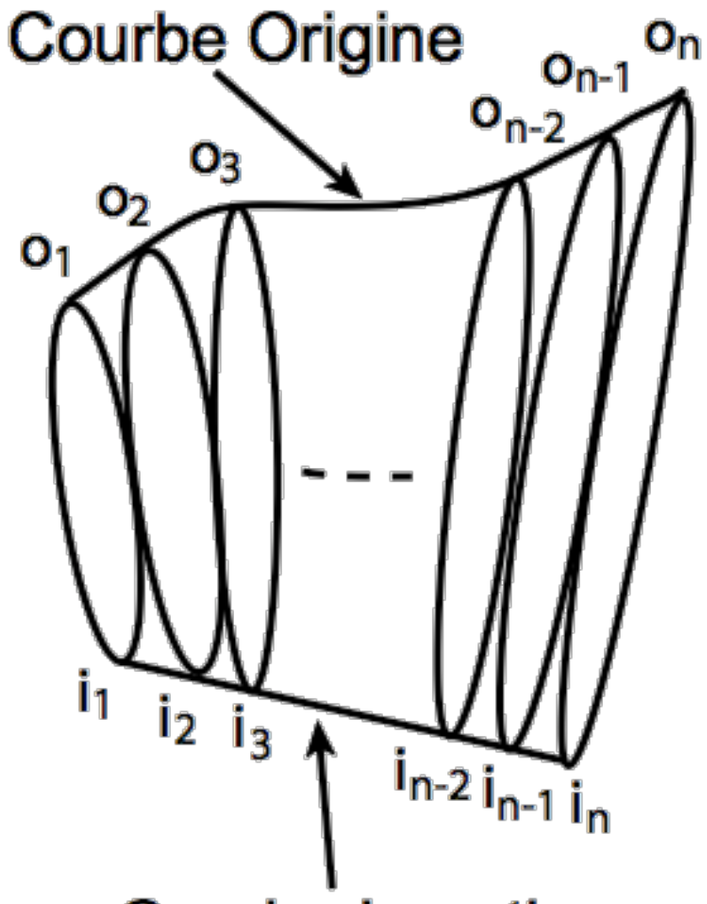


  - **Tendons can be modeled by**
    - **A new origin and a new insertion**
    - **Two small ellipsoids**



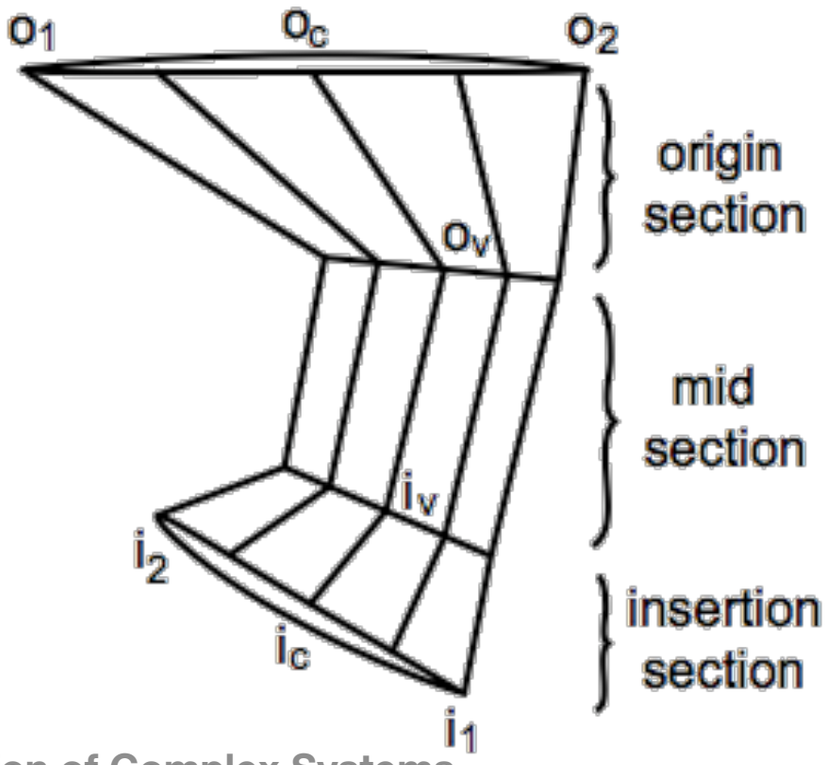
Ellipsoid Model

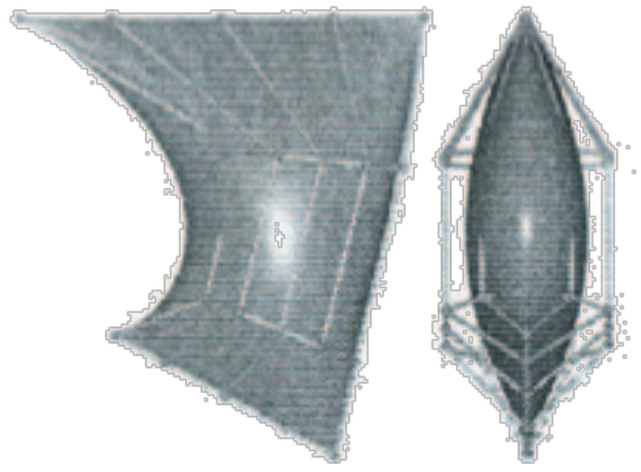
- Multi-belly muscle model
  - n origins and n insertions : two spline curves
  - Each ellipsoid is defined by
    - (o_j, i_j)
    - the normal to the plane defined by i_j, o_{j-1}, o_{j+1}



Ellipsoid Model

- General Muscle model is divided into 3 zones
  - Origin, Mid and Insertion sections
  - o_v and i_v are control points at the limit
  - Surface: Bezier patches + 2 half ellipsoids

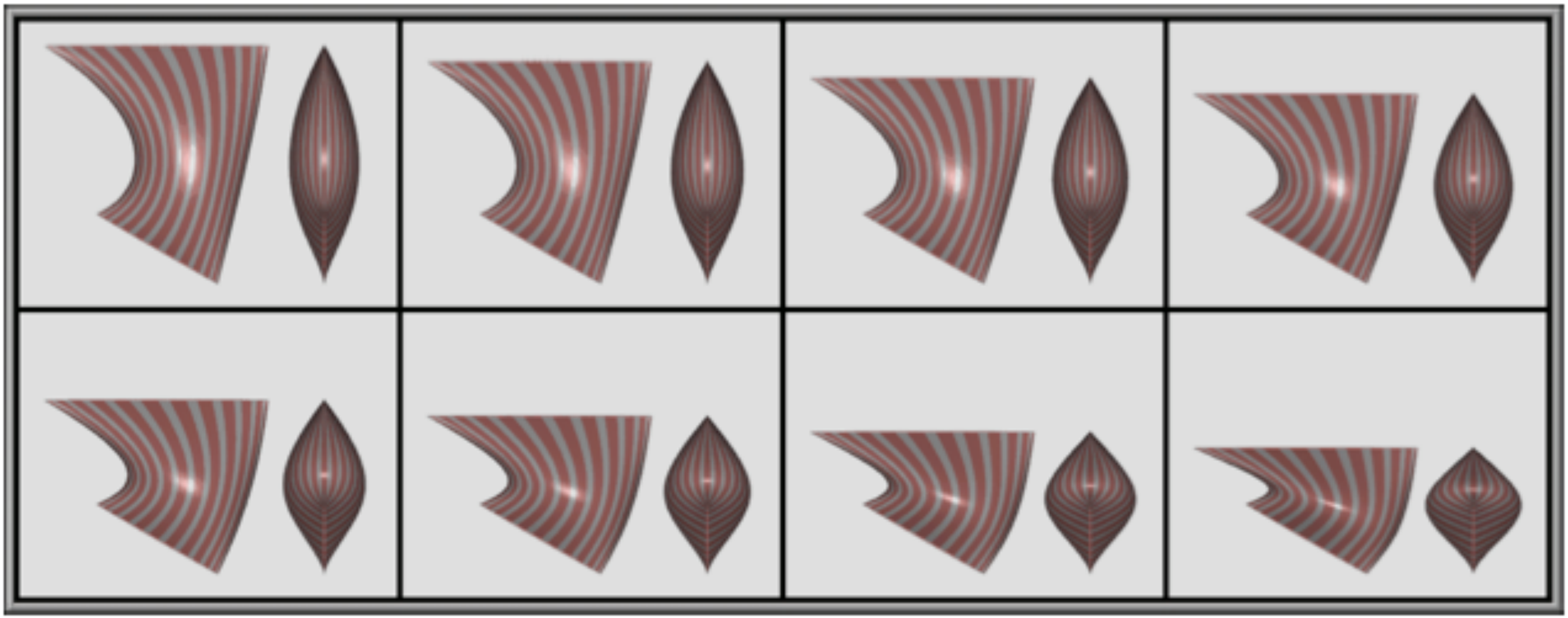




Vue de face

Vue de côté

Ellipsoid General Model

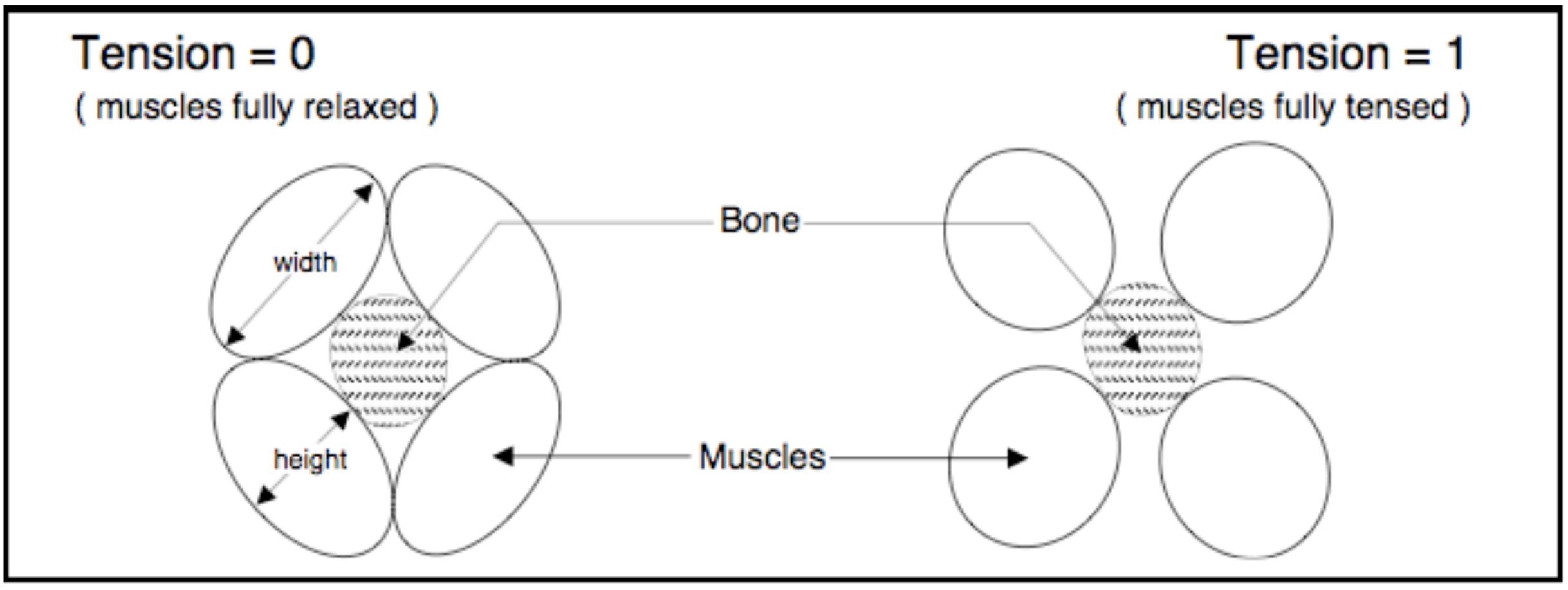


Ellipsoid Model

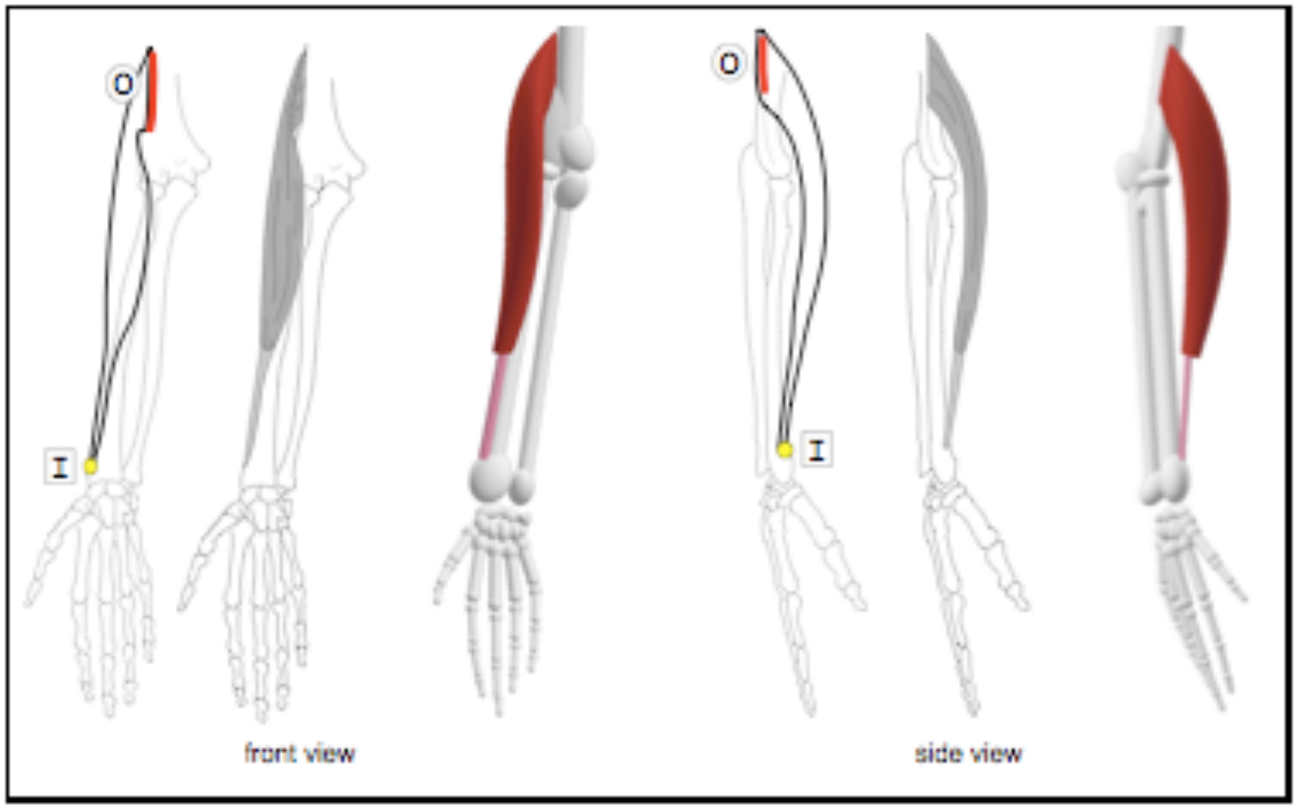
Animation

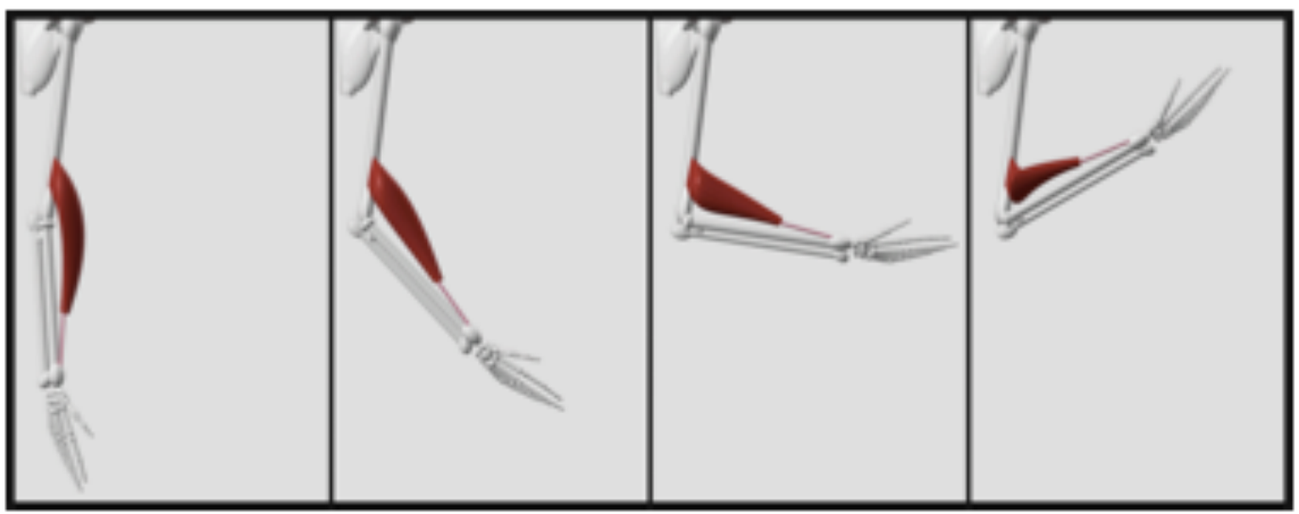
- Isotonic contraction
  - a, b, c: axes of the ellipsoid with l = 2c
  - v = (4 Pi *a*b*c) / 3
  - v = cst & r = a/b = cst (ratio)
  - When l changes, $b^2 = (3v)/(4*Pi*r*c)$
- Isometric contraction
  - Tension t
  - r has to be adjusted to reflect the tension
    - $r = (1 - t).r_n + k.t.r_n$ (r_n: ratio in the relaxed state)

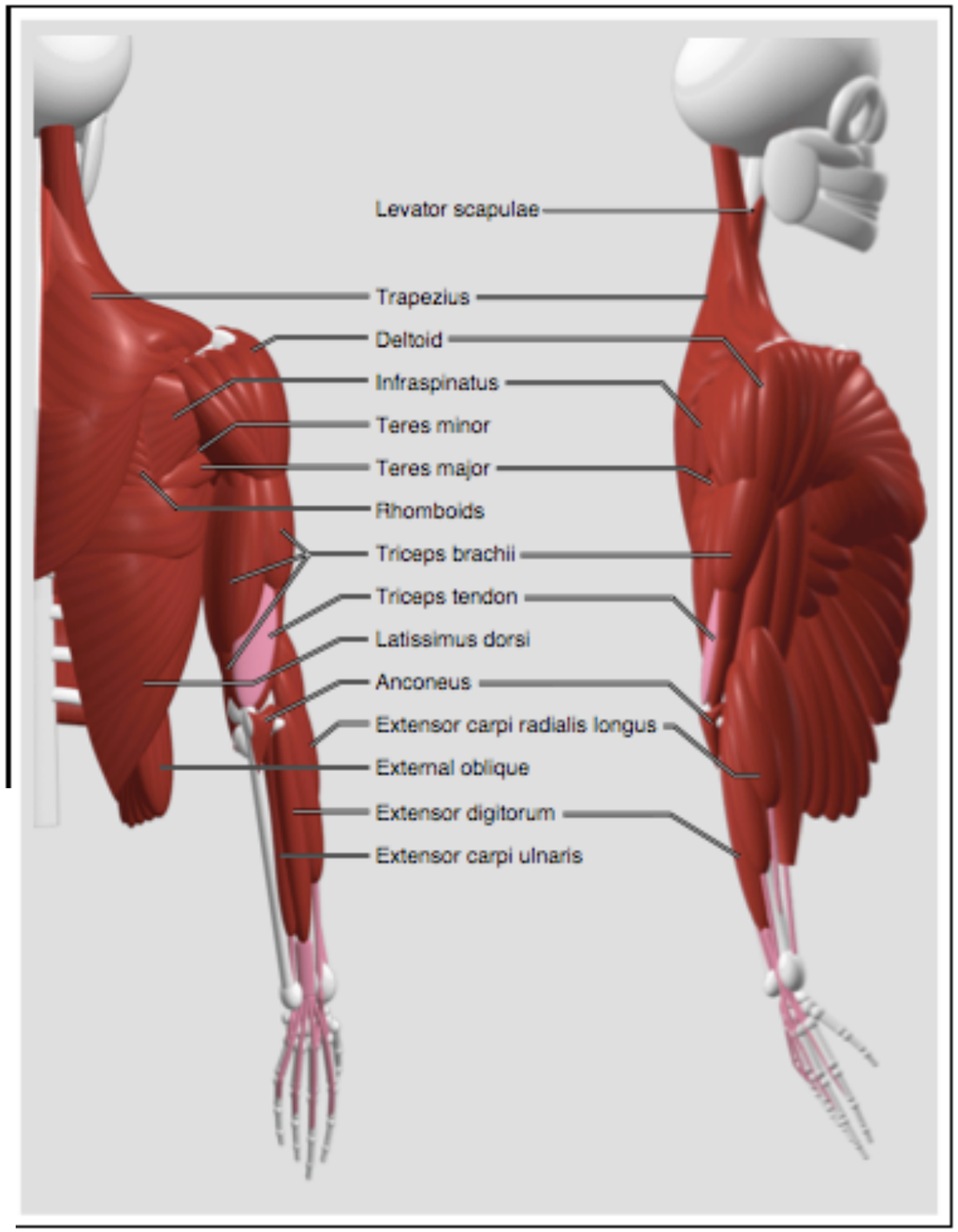
Ellipsoid - Isometric



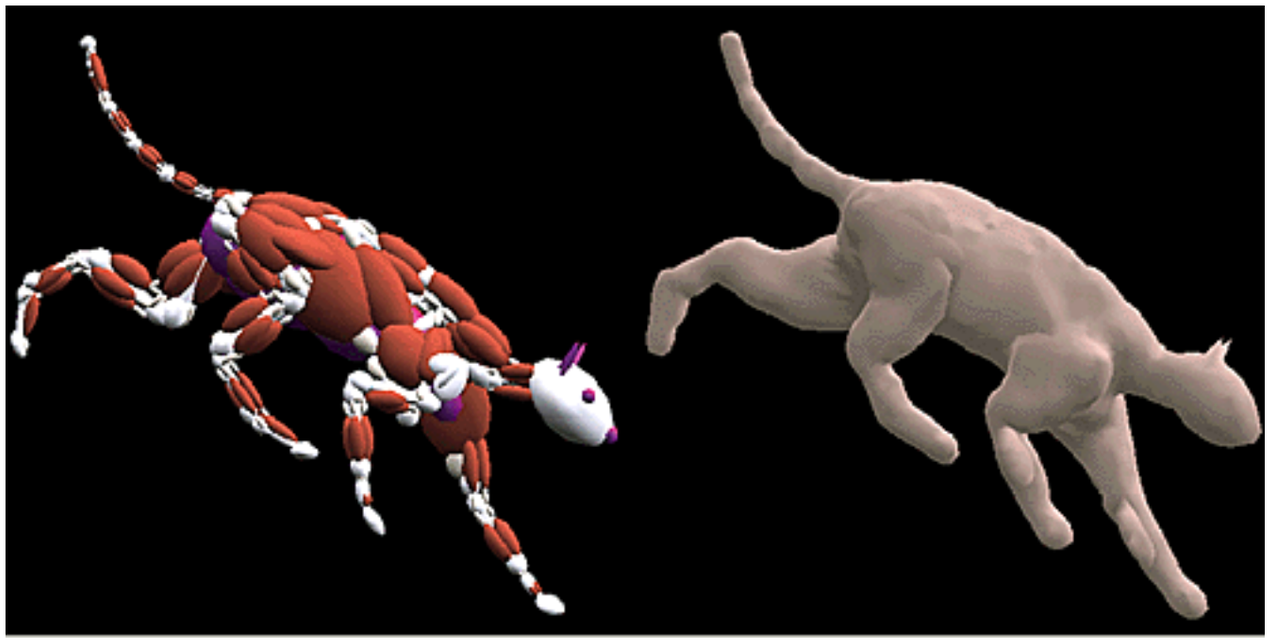
Ellipsoid - Results







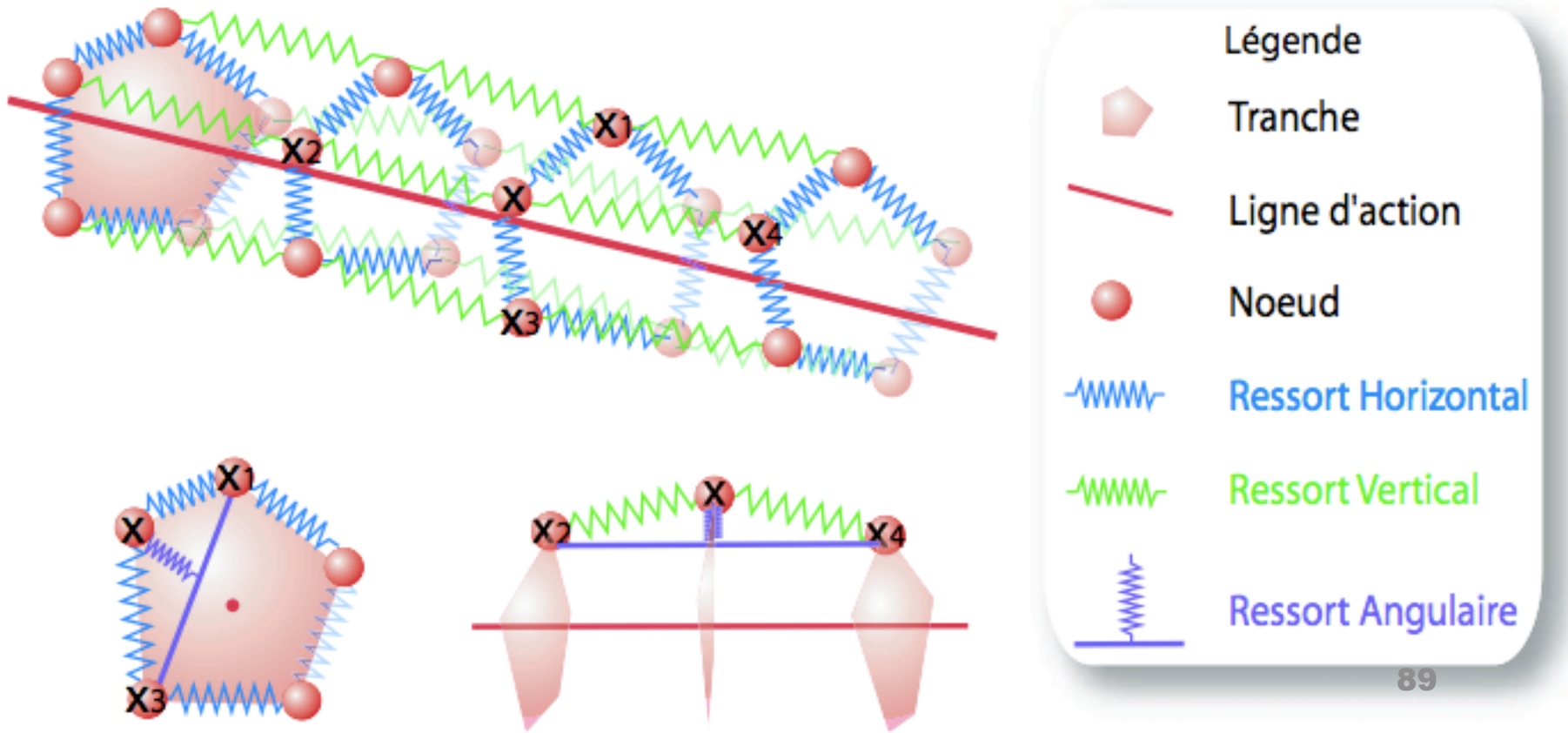
Cats - 1996



Horses

Physically Based Model

- Action line: central curve of the muscle where the muscular forces apply
- Envelope, shape of the muscle, made of masses and springs



Muscle Creation

1. Shape created by
   - Medical data
   - Ellipsoid
   - By hand by an artist
2. Shape discretized in n sections, each section's contour discretized in y nodes

   -> n*y nodes in the mesh
3. Linear and Angular springs added
4. Constraints added to move the insertions (solved by inverse dynamics)

Springs

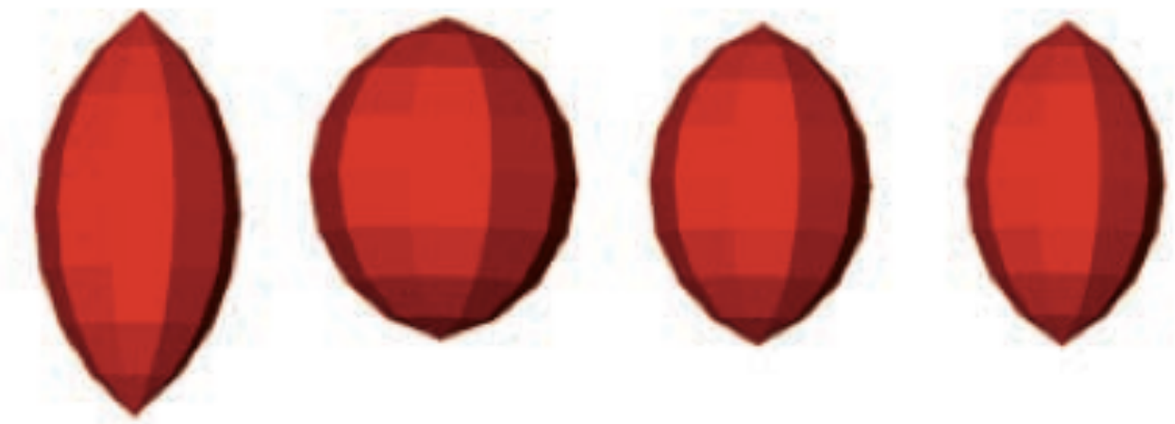
- Parameters
  - Horizontal and vertical stiffness for linear springs
  - Horizontal and vertical bending for angular springs
  - Damping coefficient
  - Mass of the muscle
- Forces computation

$$fresult_i = felasticity_i + fcurvature_i$$

$$m\ddot{x_i} + \gamma\dot{x_i} + fresult_i = fextern_i$$ With $fextern_i$ = 0

Model Limitations

- Equations solved until equilibrium
  - only kinematics
  - long computation time
  - numerical instabilities (small time step)
- Only works for fusiform muscles

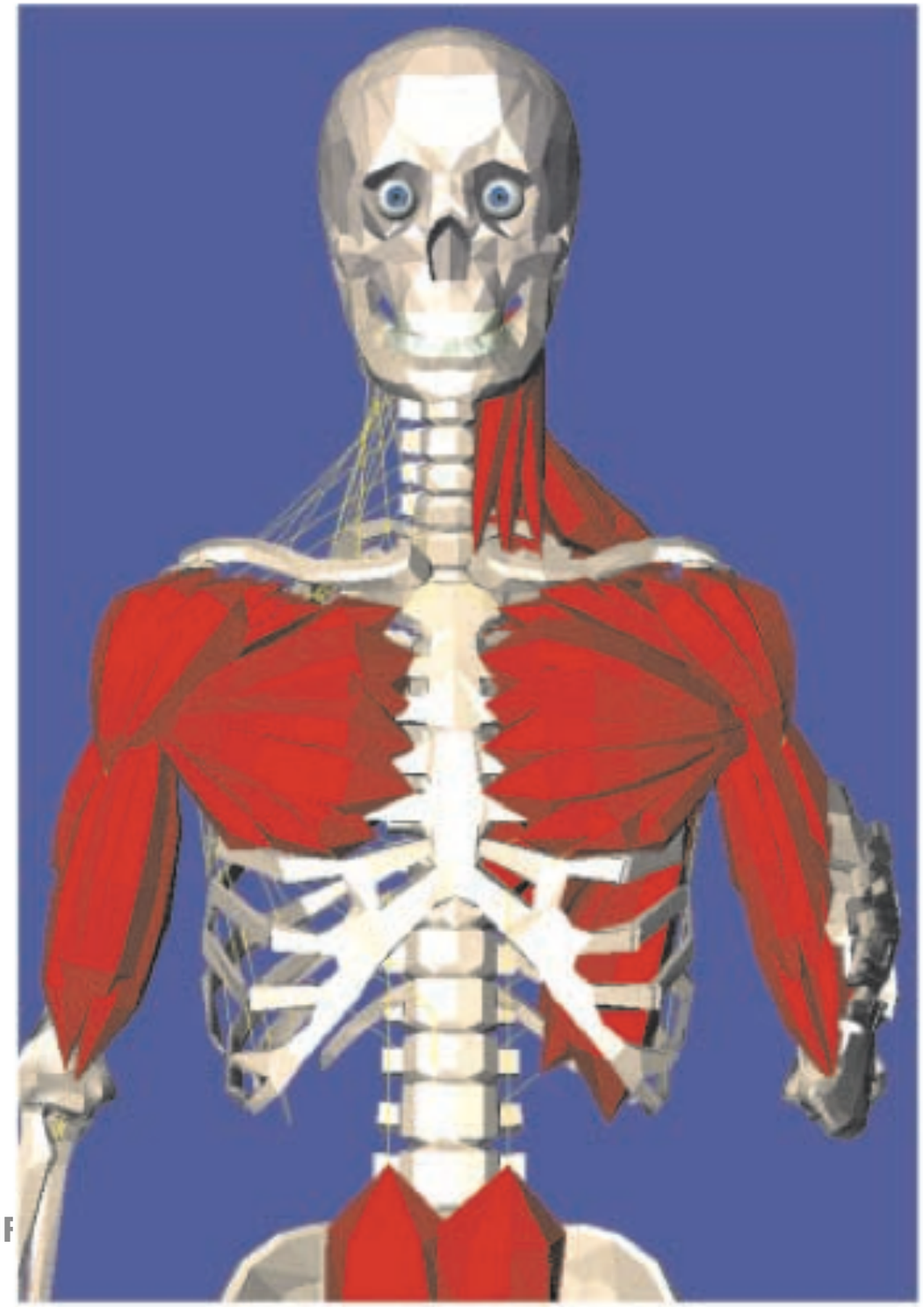


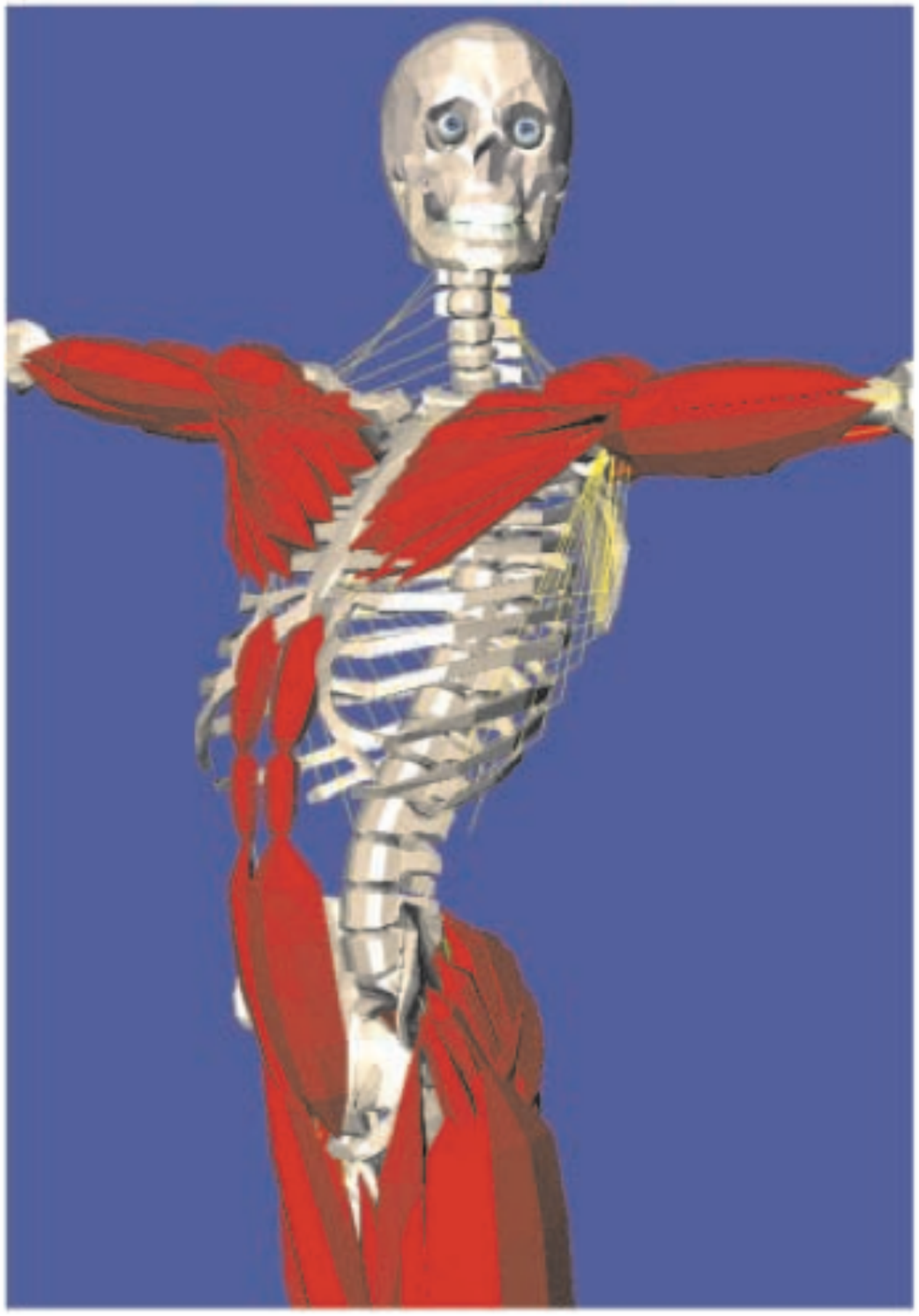
Variants of the Model

[Aubel 00]

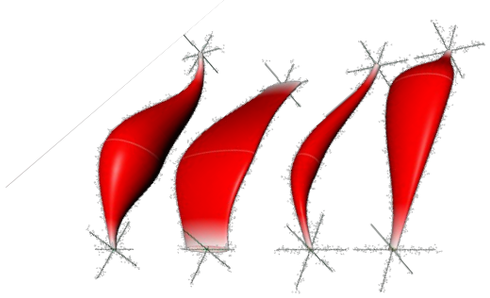
- Action line is made of connected linear springs
- The shape is a classical polygonal model
- Repulsive and attractive forces are added to “guide” the deformation of the action line
- Another variant is to have a geometric action line (less precision)
- When the action line deforms, the position of the muscle vertices are computed by SSD (linear blend)

Results

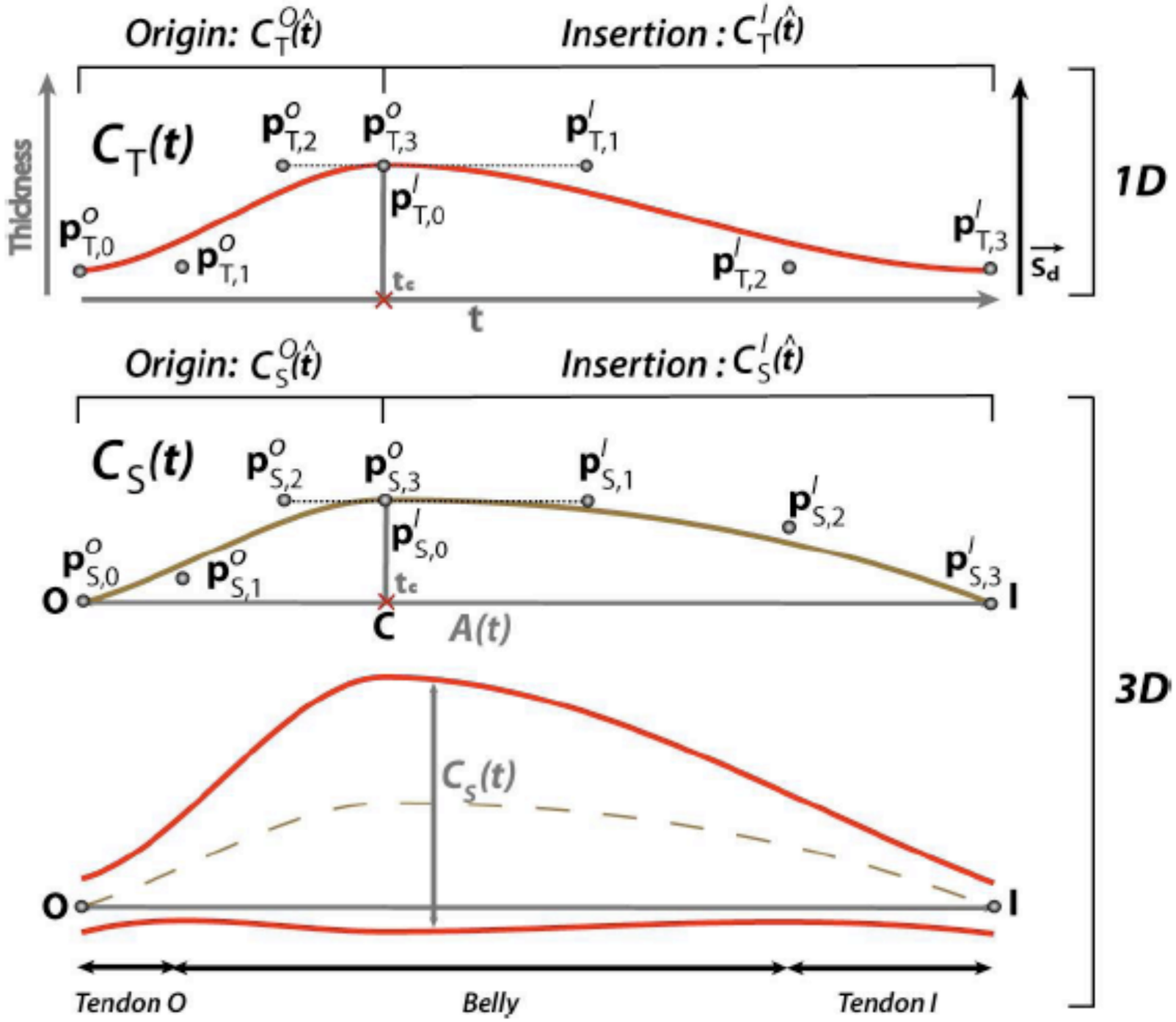




Parametric Model



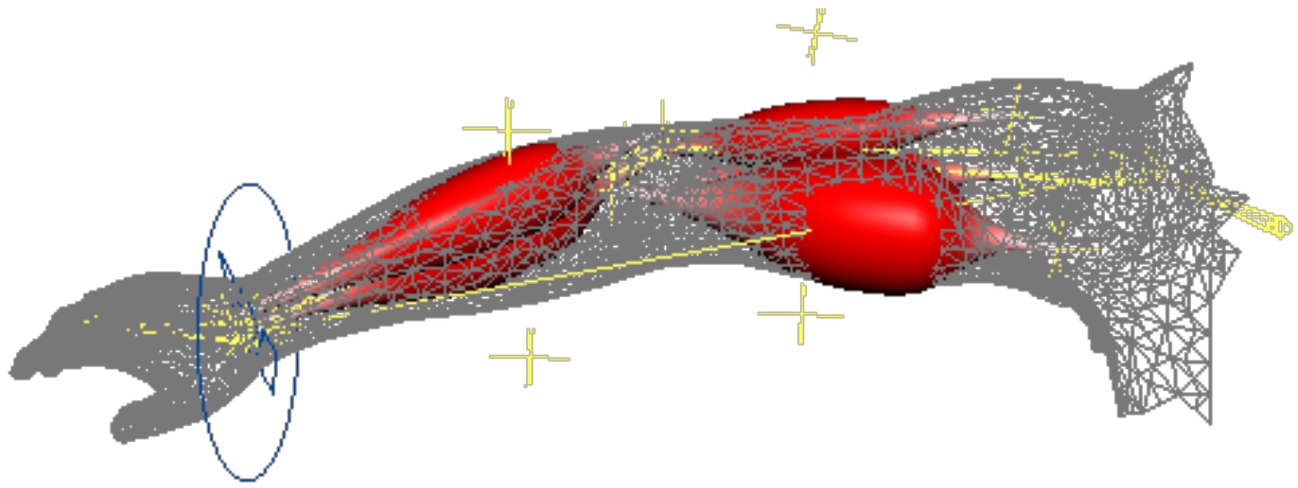
Ramos et al. A Muscle Model for Enhanced Character Skinning, 2013



Parametric Model

Properties

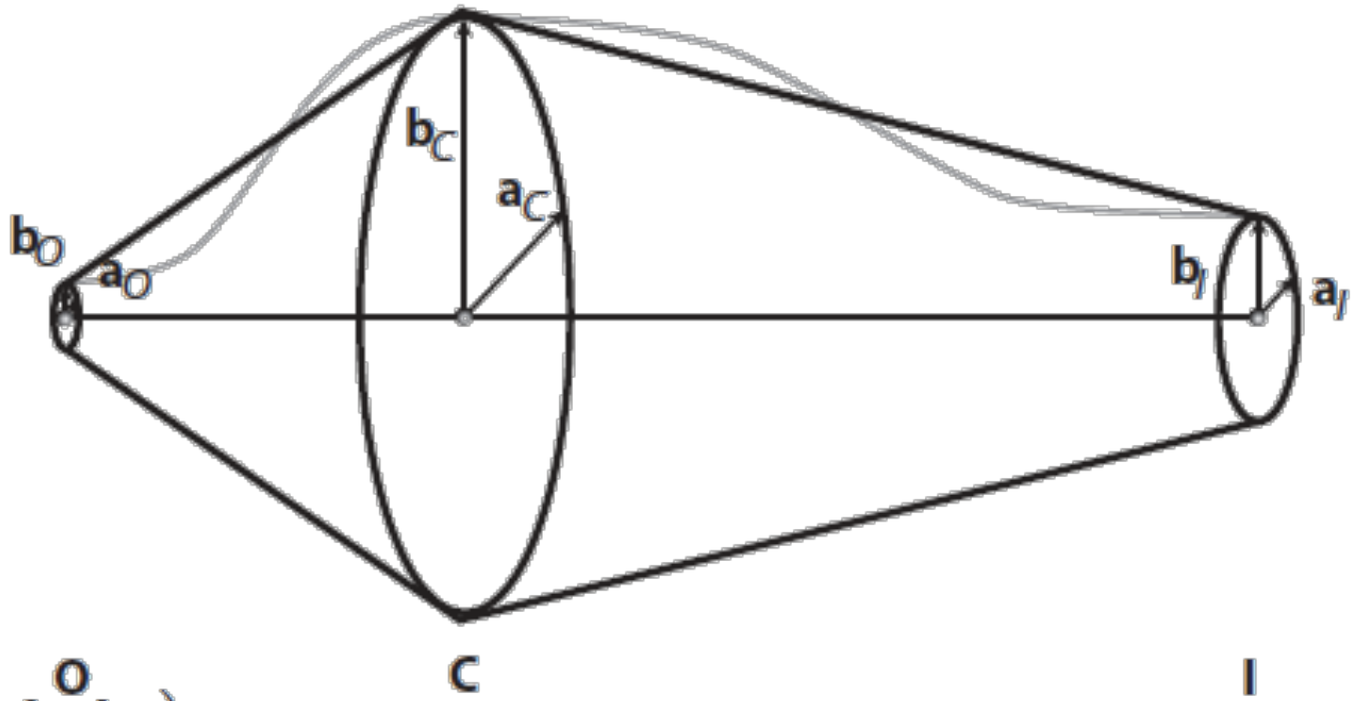
- Individual muscle model with Action Line
- Volume conservation
- Isometric and isotonic contractions
- Sliding skin behavior
- GPU implementation



Parametric Model

Volume Approximation

- 2 elliptic cones
- Variation < 1%



$$
\begin{aligned}
V &= \|\overrightarrow{OC}\|\pi\left(\frac{2a_Cb_C+a_Cb_O+a_Ob_C+2a_Ob_O}{6}\right) + \\
&\quad \|\overrightarrow{CI}\|\pi\left(\frac{2a_Cb_C+a_Cb_I+a_Ib_C+2a_Ib_I}{6}\right) \\
&= \|\overrightarrow{OC}\|\pi\left(\frac{b_C{}^2s_f+b_Ob_Cs_f+b_O{}^2s_f}{3}\right) + \\
&\quad \|\overrightarrow{CI}\|\pi\left(\frac{b_C{}^2s_f+b_Ib_Cs_f+b_I{}^2s_f}{3}\right)
\end{aligned}
\tag{6}
$$

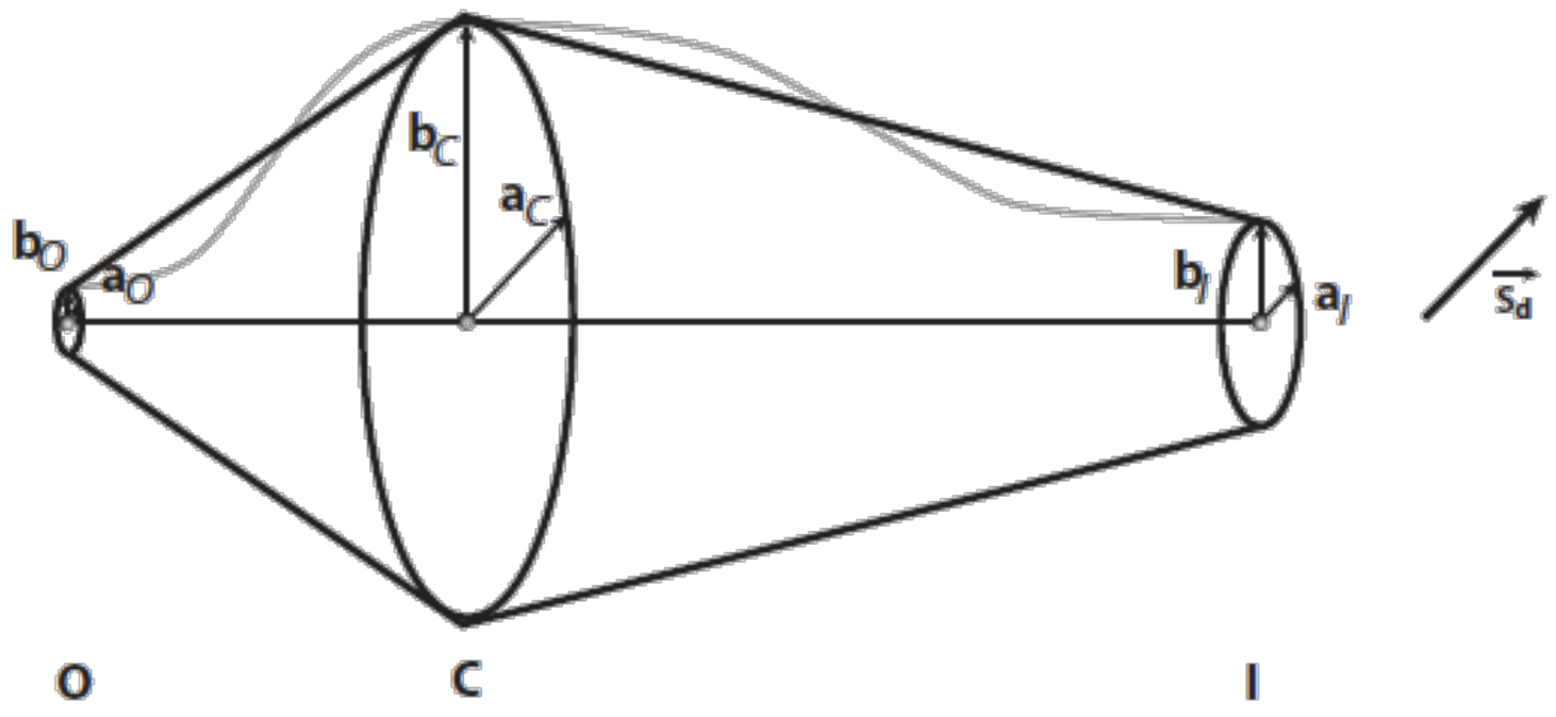
$$
V = \frac{\pi s_f}{3}(\|\overrightarrow{OC}\|(b_C{}^2+b_O{}^2+b_Cb_O)+\|\overrightarrow{CI}\|(b_C{}^2+b_I{}^2+b_Cb_I))
$$

Parametric Model

Isotonic Contraction

- Volume is constant, OC & CI shorten
- New b_C is computed (b_O and b_I const)

$$V = \frac{\pi s_f}{3}(\|\overrightarrow{OC}\|(b_C^2 + b_O^2 + b_C b_O) + \|\overrightarrow{CI}\|(b_C^2 + b_I^2 + b_C b_I))$$



Parametric Model

Isometric Contraction

- Tension Parameter
- $S_f = a_x / b_x = const$ (hence V = const)
- $a_x = a_x * tension$; $b_x = b_x / tension$

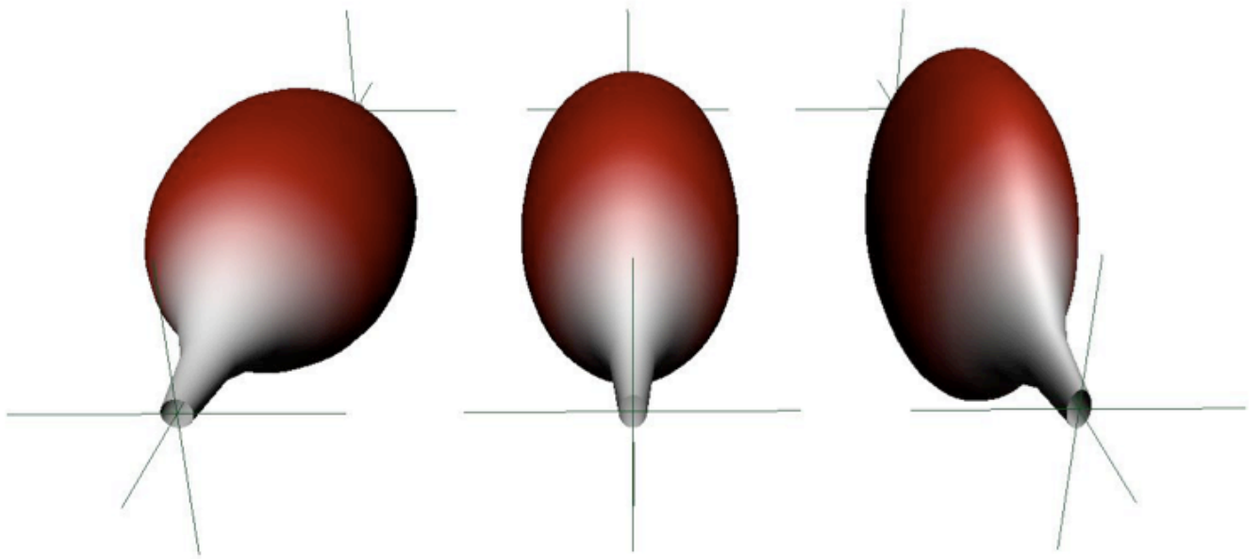


Figure 5: *Varying tension parameter. From left to right:* $\mathbf{tension} = 1$, $\mathbf{tension} = 1.15$ *and* $\mathbf{tension} = 1.3$.

Parametric Model

Muscle Deformation

- At runtime, the two types of contractions are involved
- New muscle shape is computed:

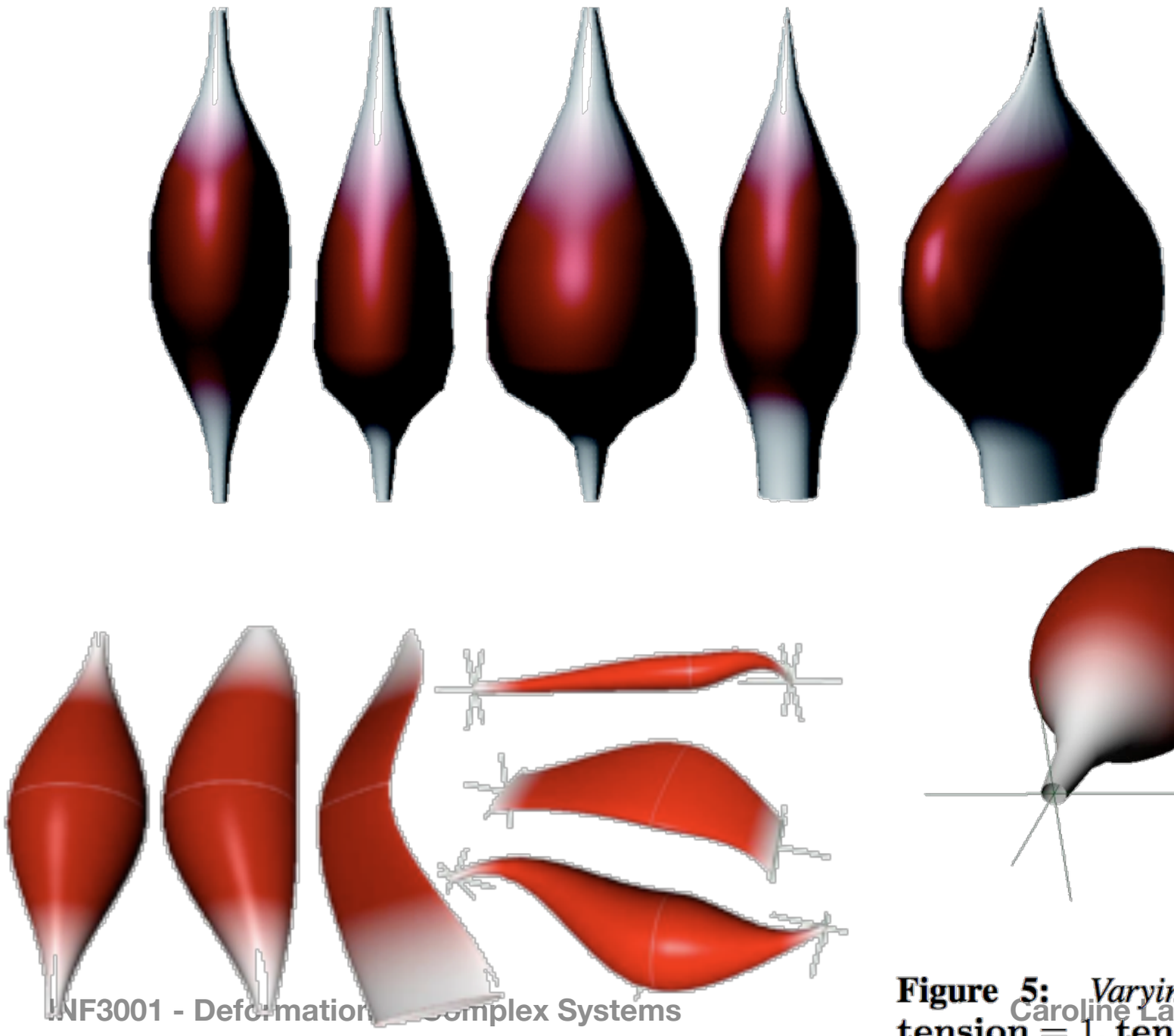
1. Isotonic contraction

$$V = \frac{\pi s_f}{3}(\|\overrightarrow{OC}\|(b_C{}^2 + b_O{}^2 + b_C b_O) + \|\overrightarrow{CI}\|(b_C{}^2 + b_I{}^2 + b_C b_I))$$

2. Isometric contraction

$$\tilde{a}_X = a_X * \text{tension}$$
$$\tilde{b}_X = b_X / \text{tension}$$

Parametric Model



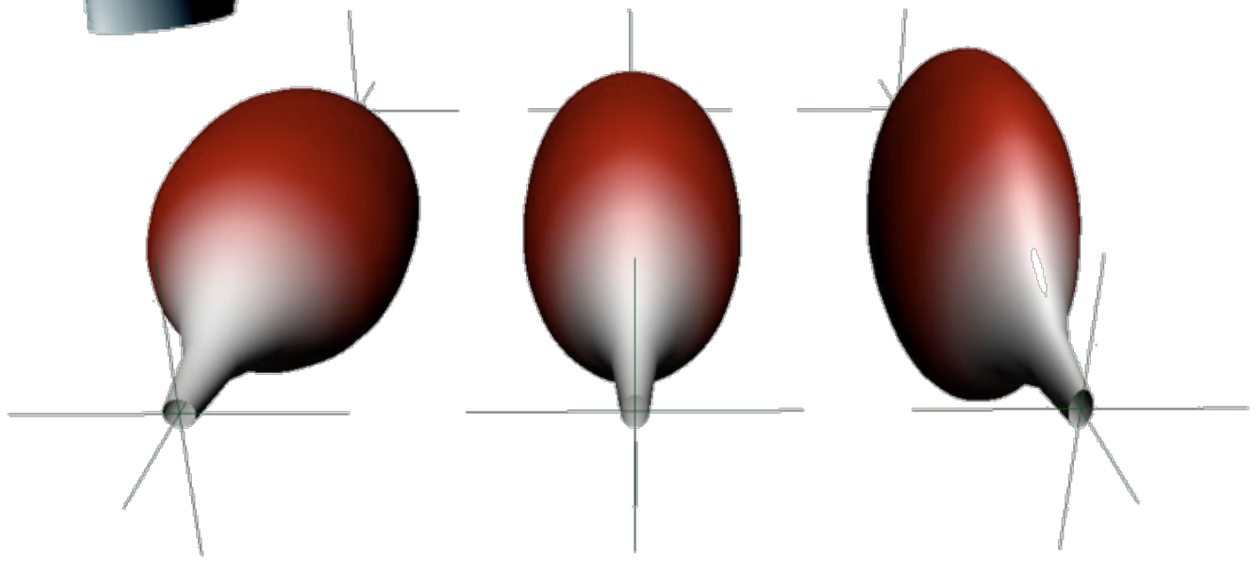
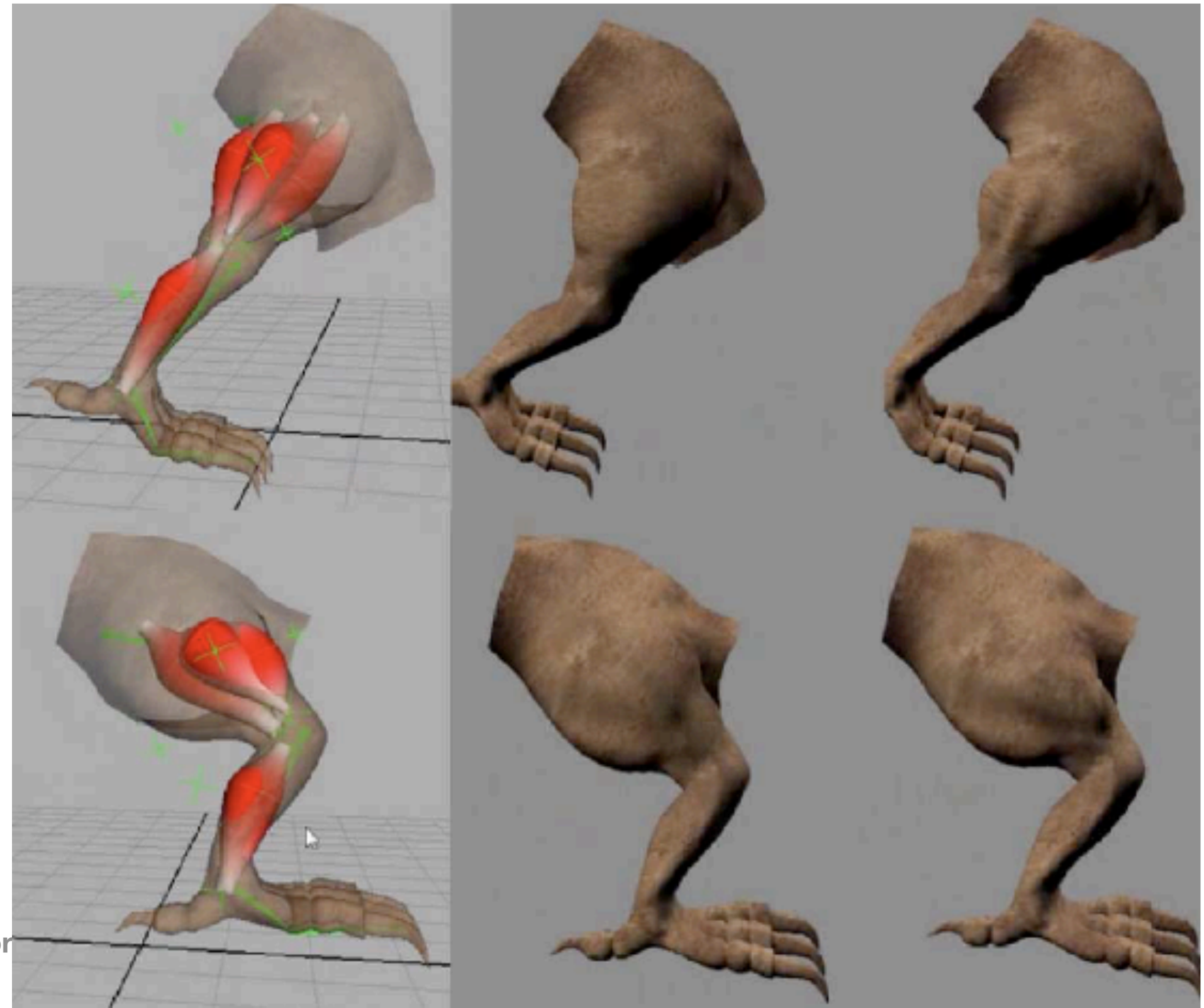


Figure 5: *Varying tension parameter. From left to right:* **tension** $= 1$, **tension** $= 1.15$ *and* **tension** $= 1.3$.

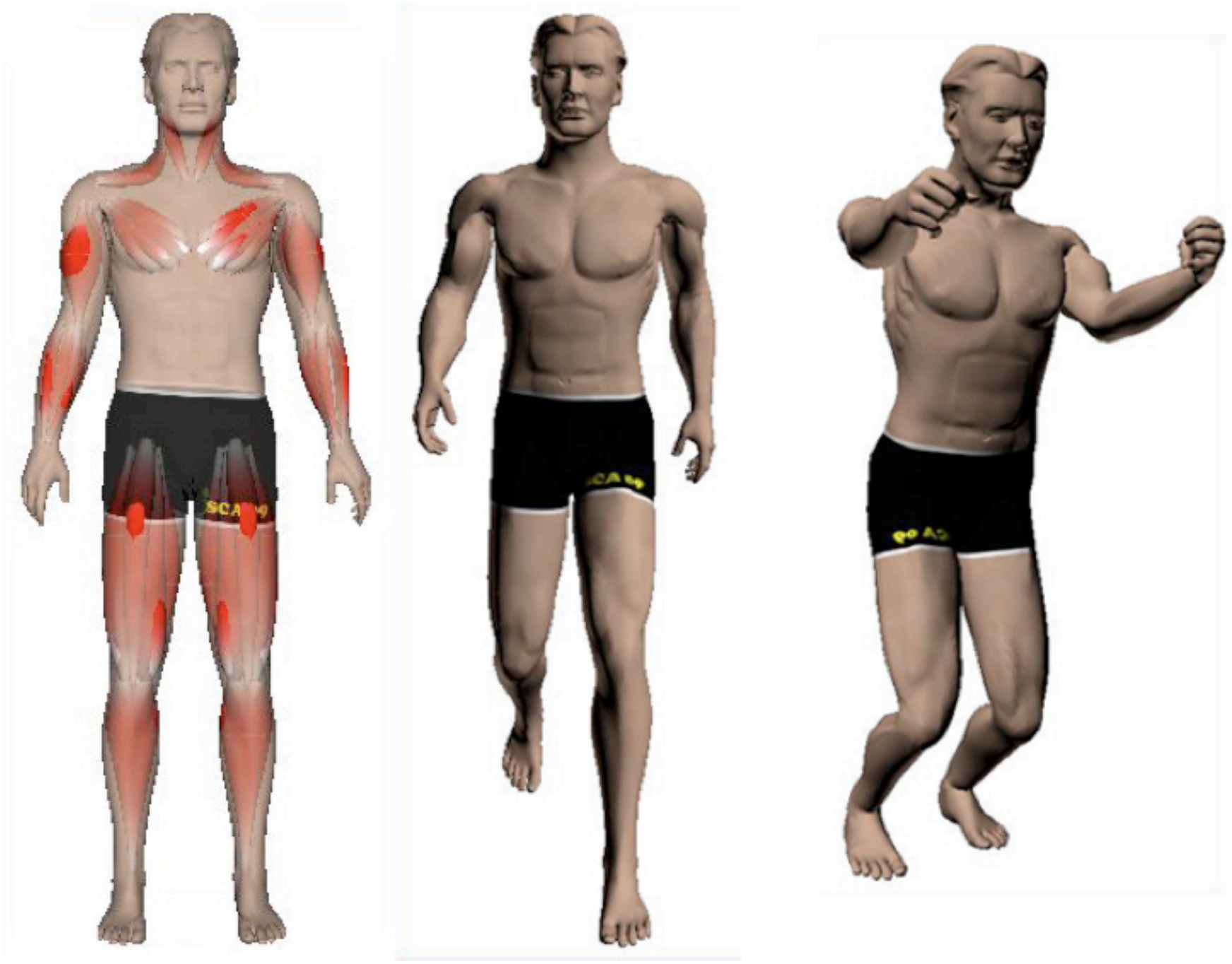
Skin Deformation

- Each skin vertex is influenced by one or more muscles
- An offset is applied in the bulging direction $\overrightarrow{s_d}$
  - Influence weights
  - Fat layer of constant thickness

Results



Results



Parametric Model

Results

Juan Ramos, Caroline Larboulette

A Muscle Model for Enhanced Character Skinning

Universidad Rey Juan Carlos
University of South Brittany

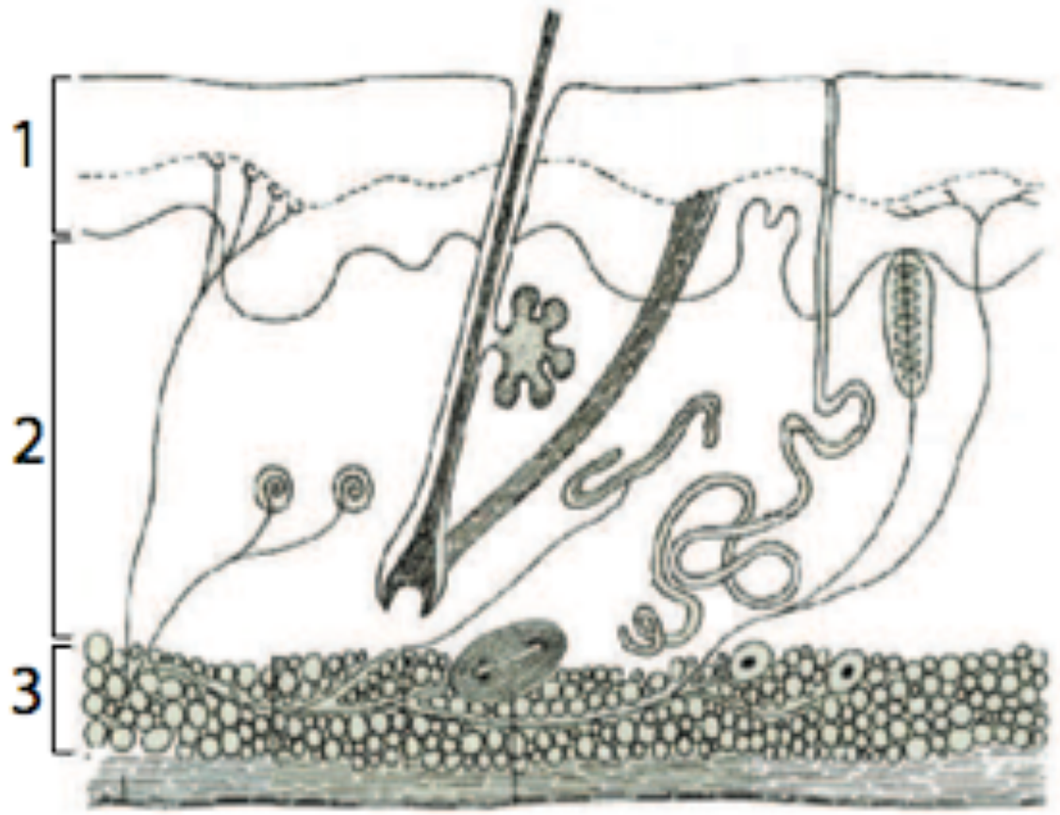
Contolling Muscles

- Like in real life, muscles control the movement

Heads Up

Human Anatomy - Skin

- Composed of 3 layers
  1. Epidermus
  2. Derme
  3. Hypodermus
- Deformations
  - Elastic (wrinkles, quasi-incompressible)
  - Plastic (aging)



Continuous Elastic Model

- Surface : polygonal model
- Nodes are assigned
  - Position, velocity, acceleration
  - Mass
  - Stiffness coefficient
- Forces are applied (attractive or repulsive)
  - From the skeleton
  - From the muscles (can simulate a fat layer)
  - Springs can be added to “anchor” the skin
- Equations solved by numerical methods

Mass-Spring model

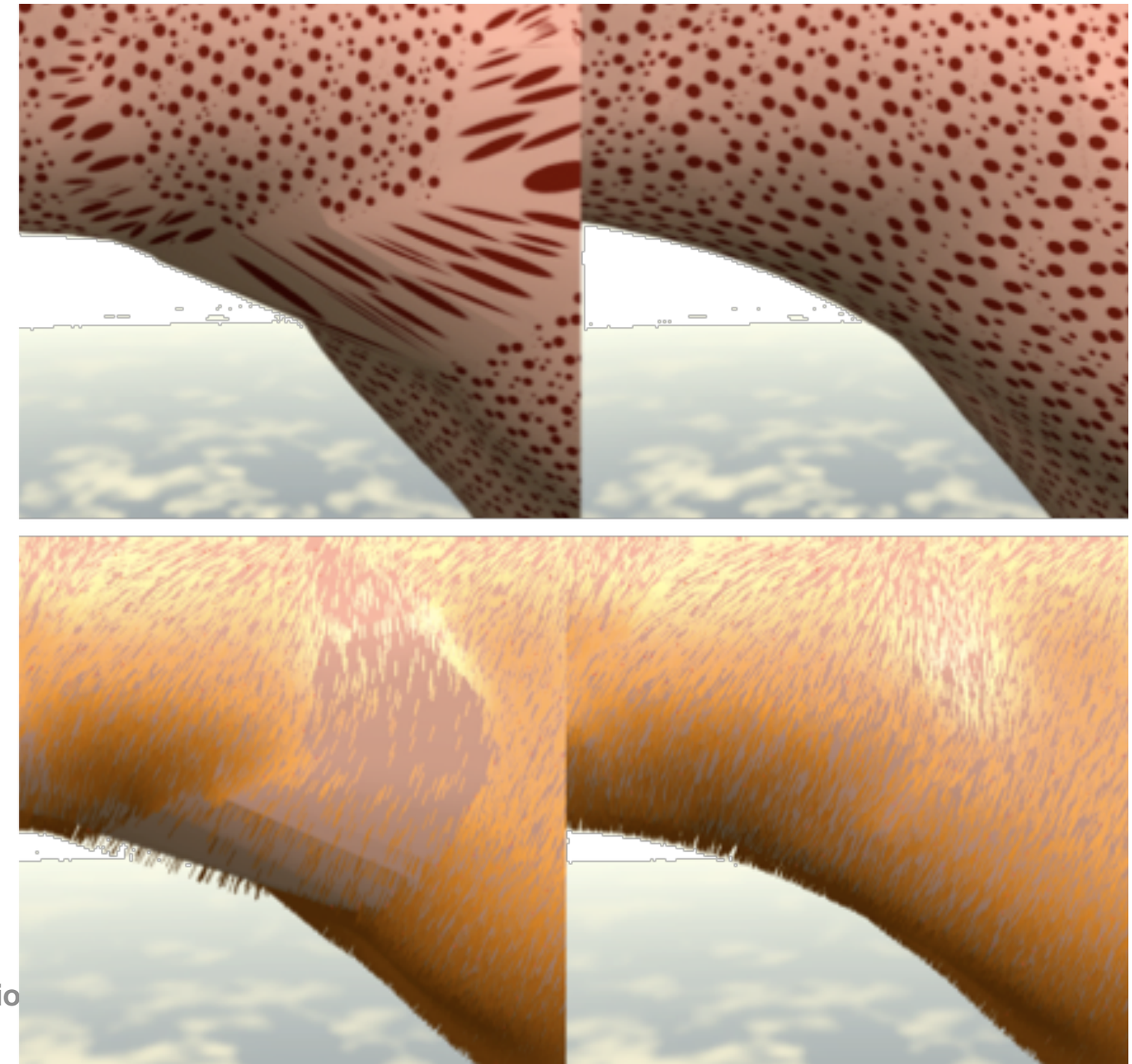
- Each edge is a spring

$$k_e(v_1, v_2) = (a_1 + a_2)/len^2$$

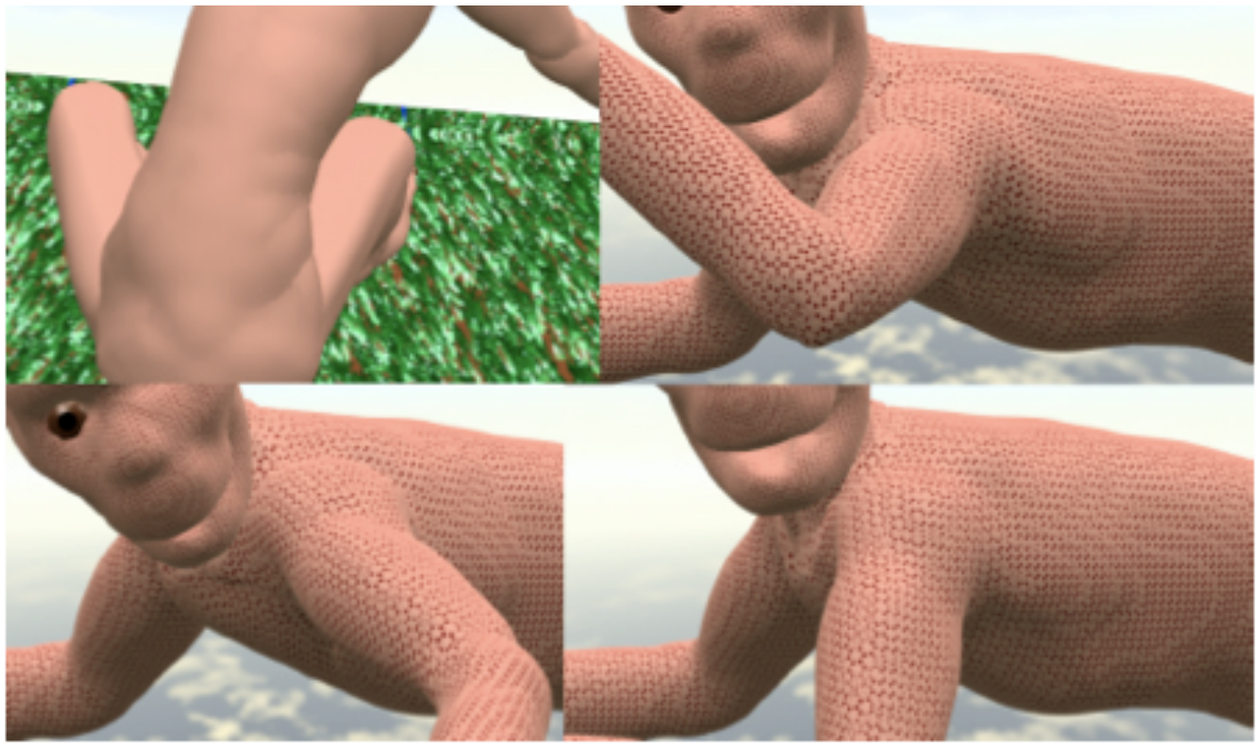
  - ke is the stiffness
  - Len is the length of the edge at rest
  - v_1 and v_2 are the vertices
  - a_1 and a_2 are the areas of the connected triangles
- Each node is attached to its “anchor” on the underlying surface (muscle, bone, other) by a spring
  - ka is the stiffness
  - Ca is a cst that helps control the amount of sliding
  - Rest length = 0

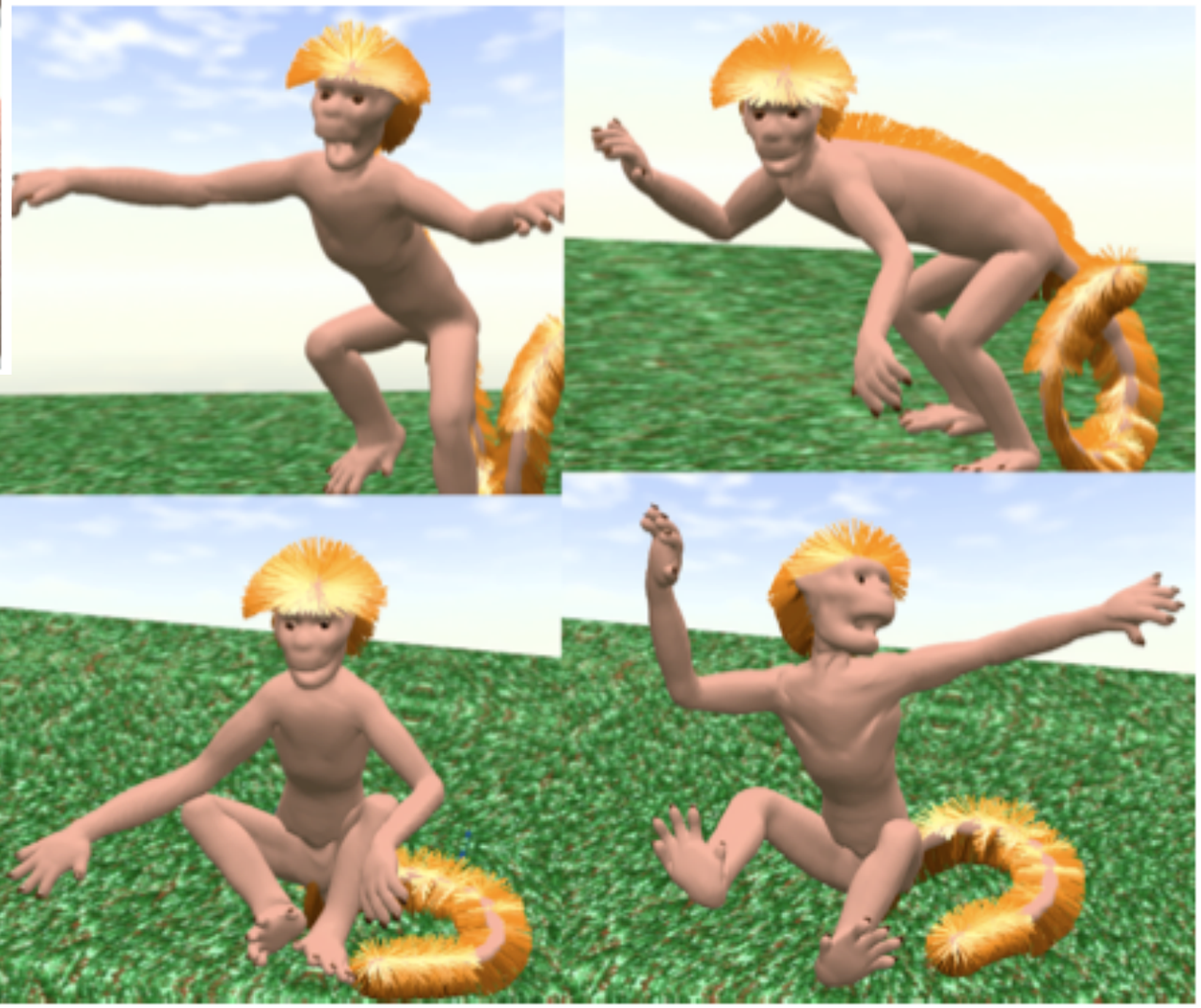
$$k_a = C_a . \sum_i (\frac{a_i}{3})$$

Skin Relaxation



Results





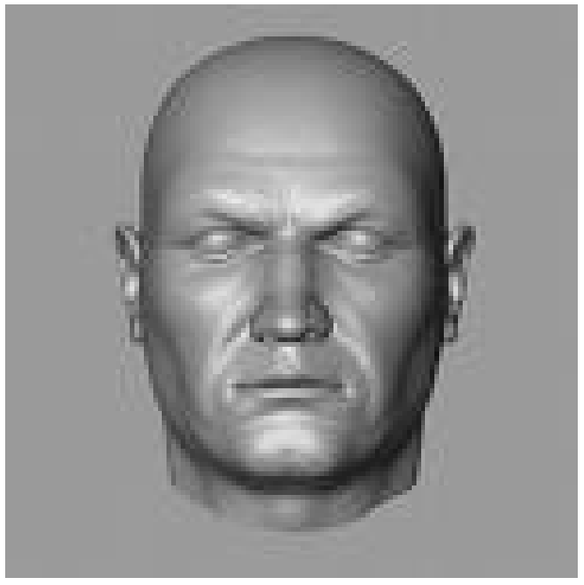
Facial Animation

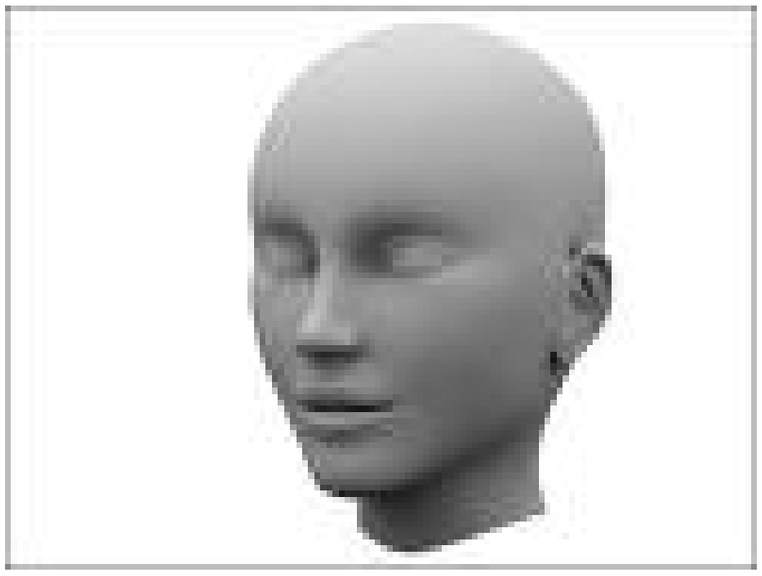
Definitions

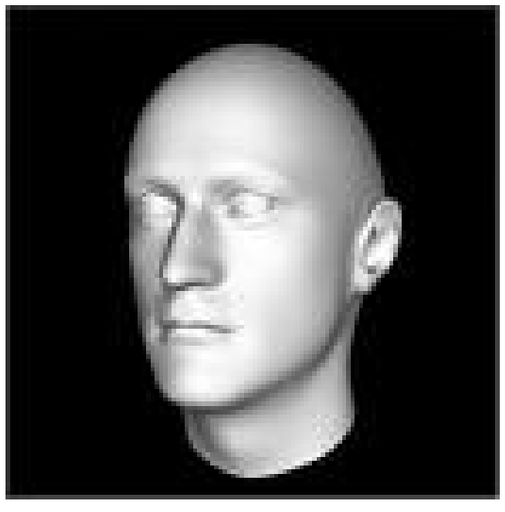
Facial Deformation Parameters

- Conformational parameters: parameters that distinguish a person's head and face from another one
  - Taken into account during modeling
- Expressive parameters: parameters that change during animation
  - What we are interested in in this class

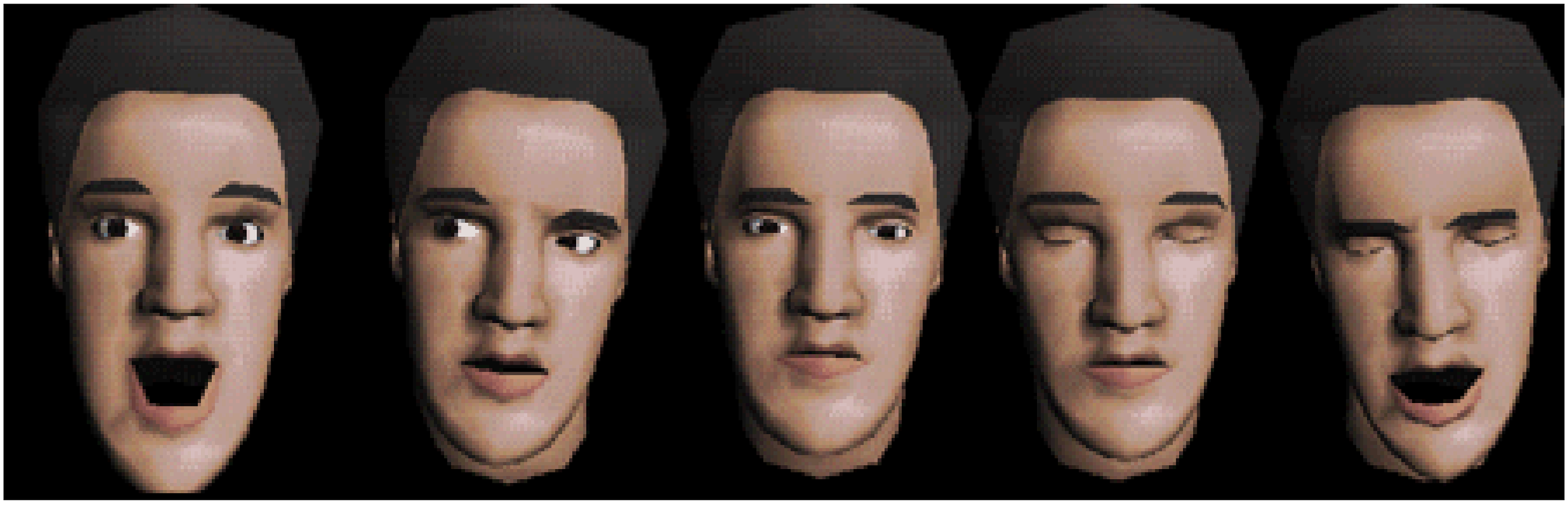
Conformational Param.







Expressive Parameters



Mesh-based Techniques

1. Face Deformation
   - Texture Mapping
   - Key-shape interpolation / Blendshapes
   - FACS
   - Parameterizations
   - Muscle-based modeling
2. Details (wrinkles)

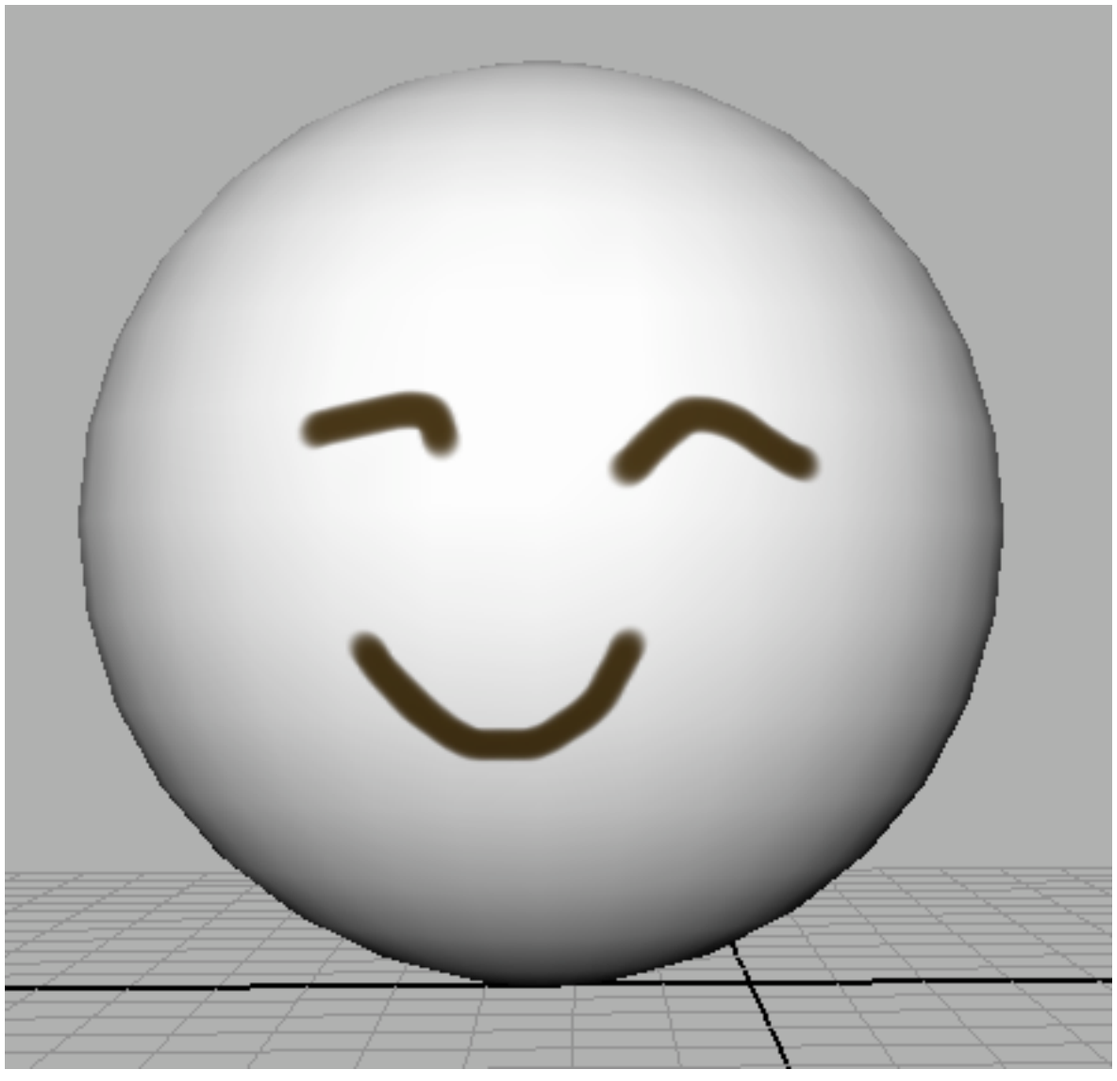
Texture Mapping

- Uses a simple primitive (sphere)
- Maps a different texture (with different position of the eyebrows, eyes, lips … over time
- A bone to control the jaw can be added
- Similar to 2D cartoon hand-drawn animation

Texture Mapping

+ Suitable for cartoon animation
+ Possibility of changing lighting conditions or skin color (based on emotional state)
- Unrealistic
- No 3D deformation of the nose, ears, …

Texture Mapping



Keyshape Interpolation / Blendshapes

- Linear interpolation between 2 KF or weighted sum of two blend shapes

neutral face | interpolated image | smiling face

- Cosine interpolation
  - Creates acceleration/deceleration effects at beginning and end of the interpolation

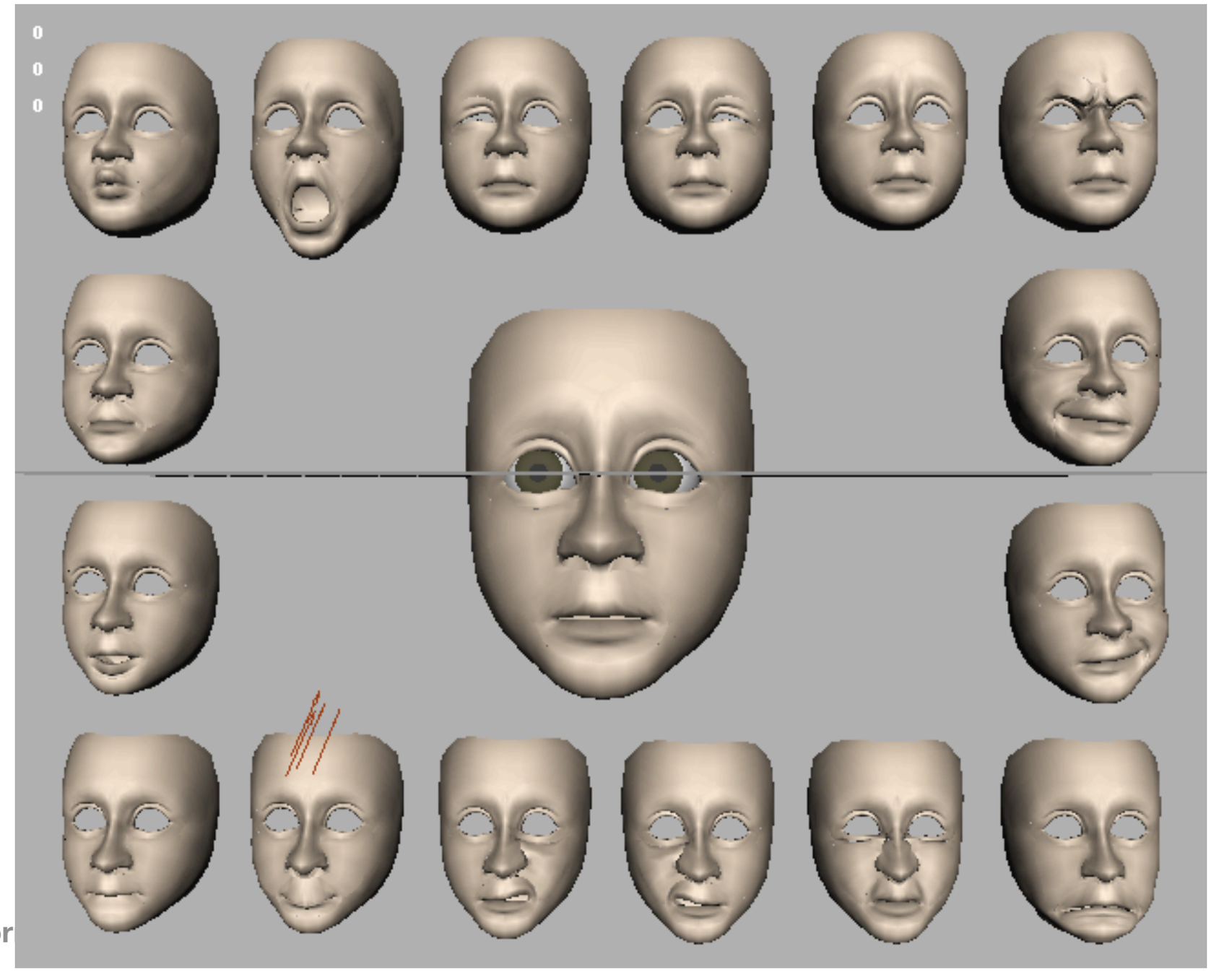
Keyshape Interpolation / Blendshapes

- The interpolation is done:
  - Directly on the vertices (needs the same amount of vertices in all meshes)
  - Indirectly on functions that control the vertices

Keyshape Interpolation / Blendshapes

+ Fast to compute (sum)
+ Good for creating a small set of animations from a few key shapes
- Small range of realistic configurations
  - For a wide range, the number of key shapes increases quickly
- Combination of independent motion difficult

Keyshape Interpolation / Blendshapes



FACS

- Means: Facial Action Coding System
- All facial expressions are decomposed into a set of 46 Action Units (AUs) basic movements
- AUs are recombined to describe a particular expression

FACS examples

- Brow lowerer
- Inner brow raiser
- Wink
- Cheek raiser
- Upper lip raiser
- Jaw drop …

FACS parameterization

- Face model is parameterized according to the motion of the AUs (using one parameter to control one AU)
- For animation, the values of parameters are interpolated
  - Not intended for animation but for the creation of specific expressions
  - Not designed for speech (mvt that depends on phonemes)

FACS : example for basic expressions

AU	FACS Name	AU	FACS Name
1	Inner Brow Raiser	12	Lid Corner Puller
2	Outer Bow Raiser	14	Dimpler
4	Brow Lower	15	Lip Corner Depressor
5	Upper Lid Raiser	16	Lower Lip Depressor
6	Check Raiser	17	Chin Raiser
7	Lid Tightener	20	Lip Stretcher
9	Nose Wrinkler	23	Lip Tightener
10	Upper Lid Raiser	26	Jaw Drop

The 6 universal emotions:
(used in MPEG-4)

Basic Expressions	Involved Action Units
Surprise	AU1, 2, 5, 15, 16, 20, 26
Fear	AU1, 2, 4, 5, 15, 20, 26
Disgust	AU2, 4, 9, 15, 17
Anger	AU2, 4, 7, 9, 10, 20, 26
Happiness	AU1, 6, 12, 14
Sadness	AU1, 4, 15, 23

Parameterization

- Extension of FACS: a given face pose is considered as a point in the n-dimensional space of all possible poses (usually 3D or 4D)
- The parameterization should be
  - Complete (covers all possible/necessary poses)
  - Easy to use (set as small as possible)

Parameterization

- Expression parameters (lip position, eye gaze, jaw rotation…) similar to FACS AUs
- Parameters need to be independent
  - No systematic way to solve conflicts
  - Done by hand … and experience ;-)
- [different from conformational parameters (things that distinguish the face of a human from another)]

Parameterization

+ Explicit control of specific facial regions
  - Introduces noticeable motion boundaries
+ Large range of facial expressions
+ Fast to compute
- Depends on mesh topology
  - No generic parameterization
- Usually, manual tuning is necessary

Using muscles

- Three types of muscles in the face
  - Linear (pulls the insertion point when contracted towards origin = fusiform)
  - Sheet (same but acts like there is a parallel array of muscles = multi-belly model)
  - Sphincter (contracts radially toward an imaginary center -> mouth)
- What varies from a model to another
  - Geometry, position of the muscles
  - Model of the skin

Muscles representation

- 3D muscles attached to bones
- 2D muscles modeled on the surface
  - Simpler than the anatomy based solution
  - But less accurate

Vector Muscle (Water)

- Consists of
  - A vector field direction + cosine functions and fall off factors (warping)
  - An origin and an insertion

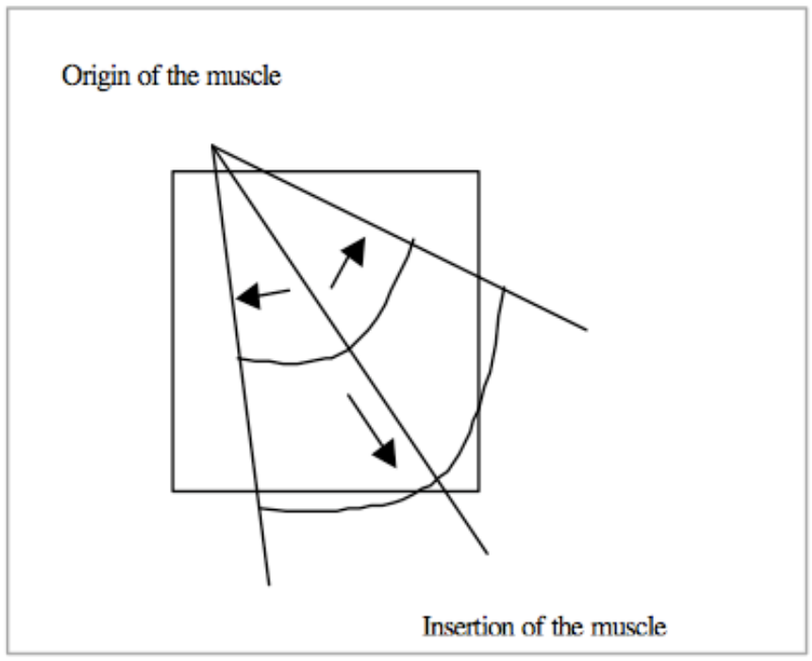


Fig. 3 - Zone of influence of Waters' linear muscle model. Deformation decreases in the directions of the arrows.

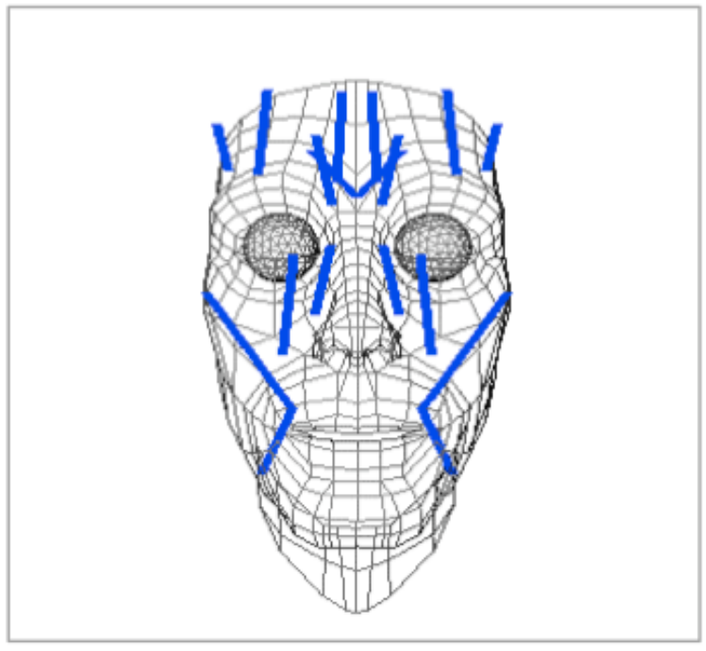


Fig. 4 - Waters' linear muscles

Vector Muscle

- Skin possibly attached by mass-springs

Lip-synch

Suwajanakorn et al., Synthesizing Obama: Learning Lip Sync from Audio, SIGGRAPH 2017

- Necessary to synchronize lips movement with sound

https://youtu.be/9Yq67CjDqvw

Wrinkles

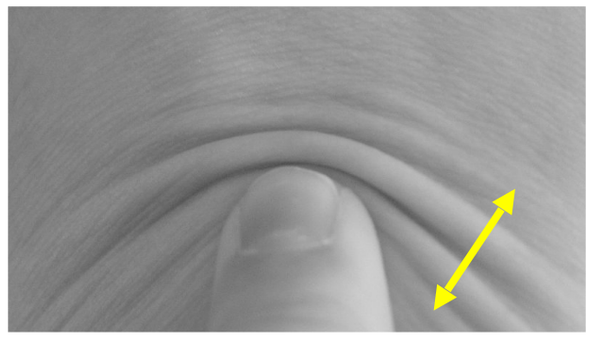
- Add realism
- Can be done with
  - Bump mapping
    - Problem with the silhouette
  - Displacement mapping
    - Problem if not the last layer
  - Mesh deformation
    - Blendshapes, driven keys … (manual)

Geometric Wrinkles

Basic Idea :

skin/cloth wrinkle because they tend to maintain area

- Physically-based simulation[Choi02,Baraff03]
- geometric constraints [sauvage04]



Aims

- Fast approximation of constant surface
- No manual design of the wrinkles shapes
- Provide control of location and main direction of wrinkles

Geometric Wrinkles

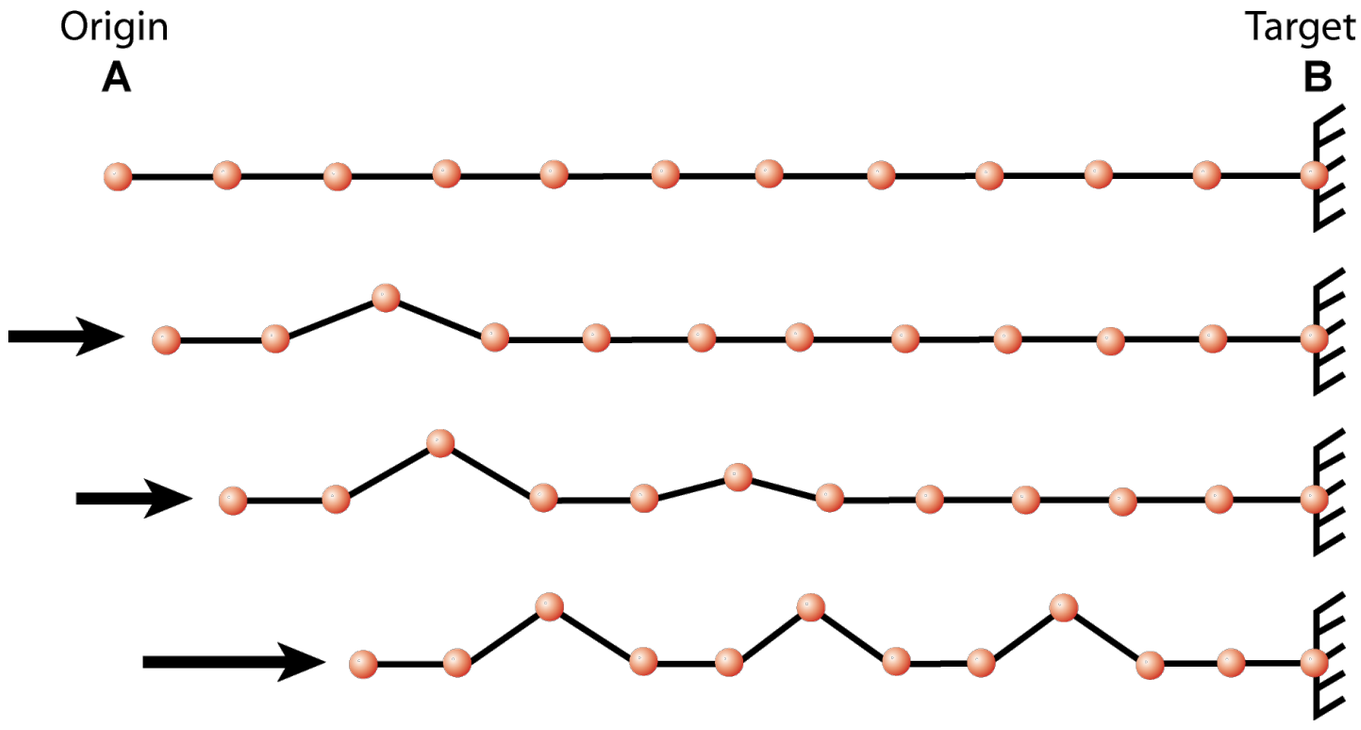
Solution

- A constant length wrinkling curve
- Controls the mesh deformation in a region of influence

- Published in 2004. Idea re-used in 2006 to model shoes and in 2009 to make car seat wrinkles.

Wrinkling Curve

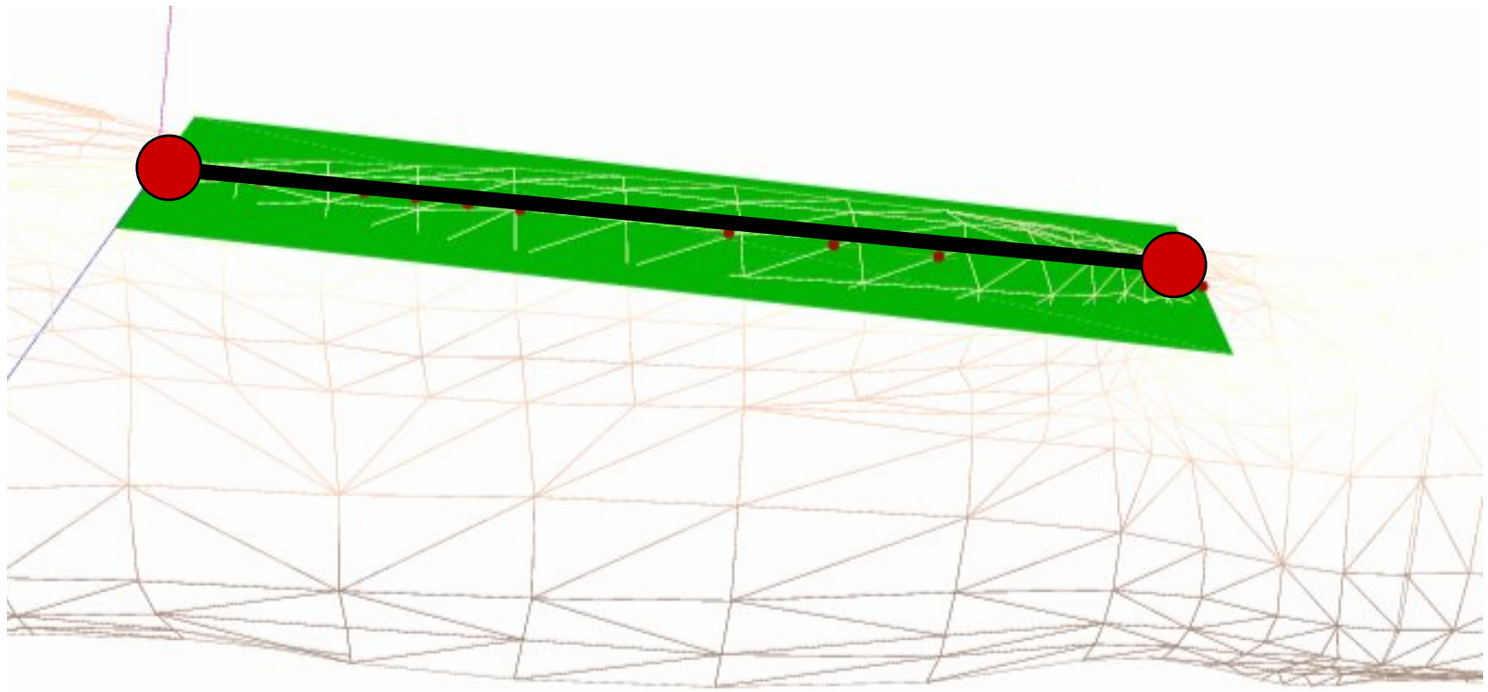
- 2D discrete curve of constant length
- Different propagation schemes



- Control of frequency, width of the bumps

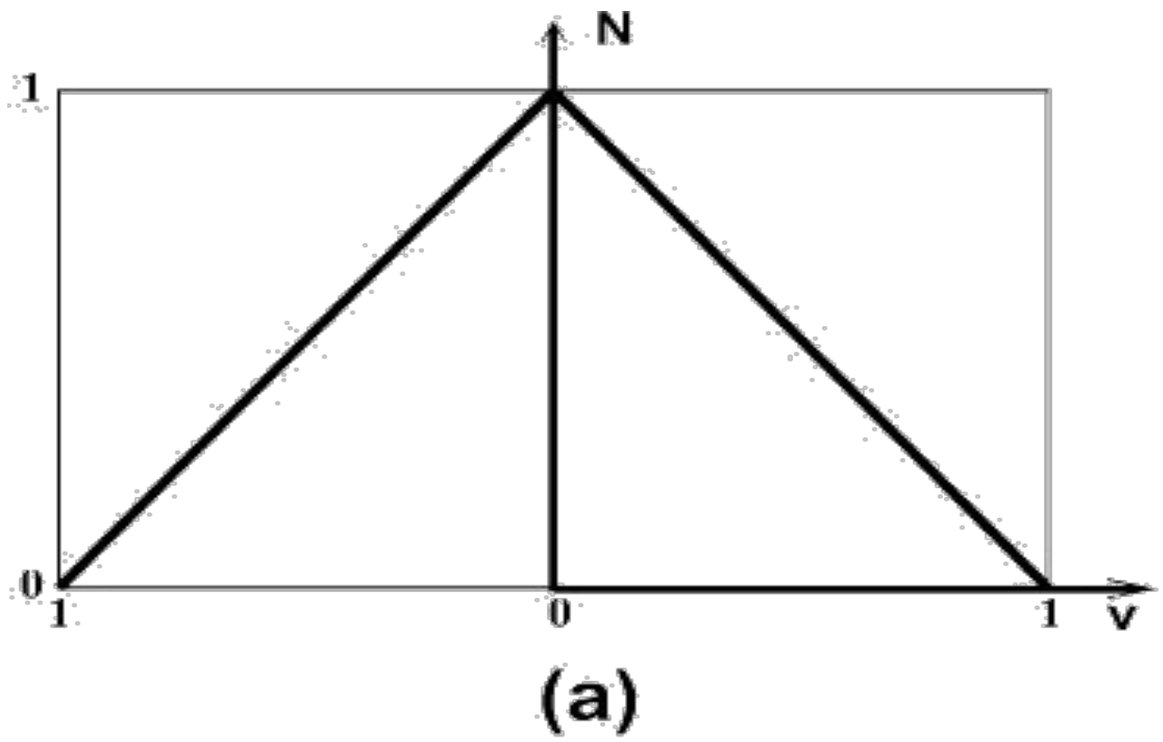
The Wrinkling Tool

1. Draw a line segment
   - End points are anchored to the mesh
2. Specify a region of influence
   - Points under the green rectangle

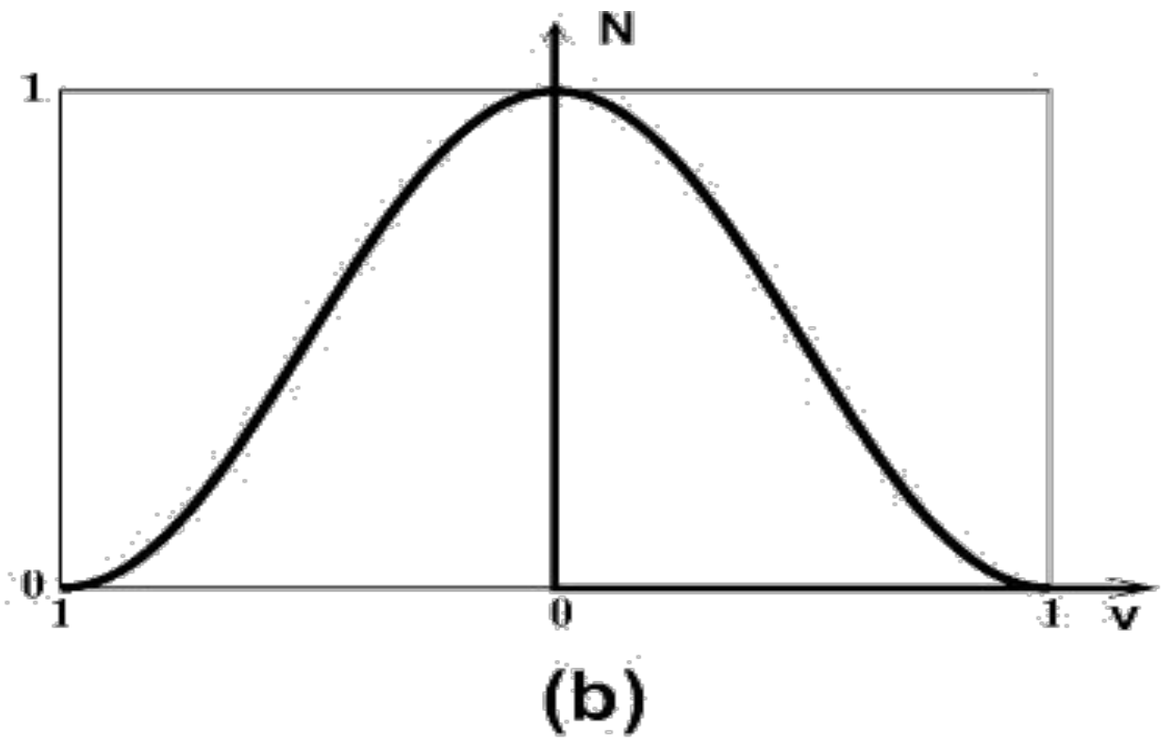


The Wrinkling Tool

3. Choose an attenuation profile (to describe how it vanishes on the sides)



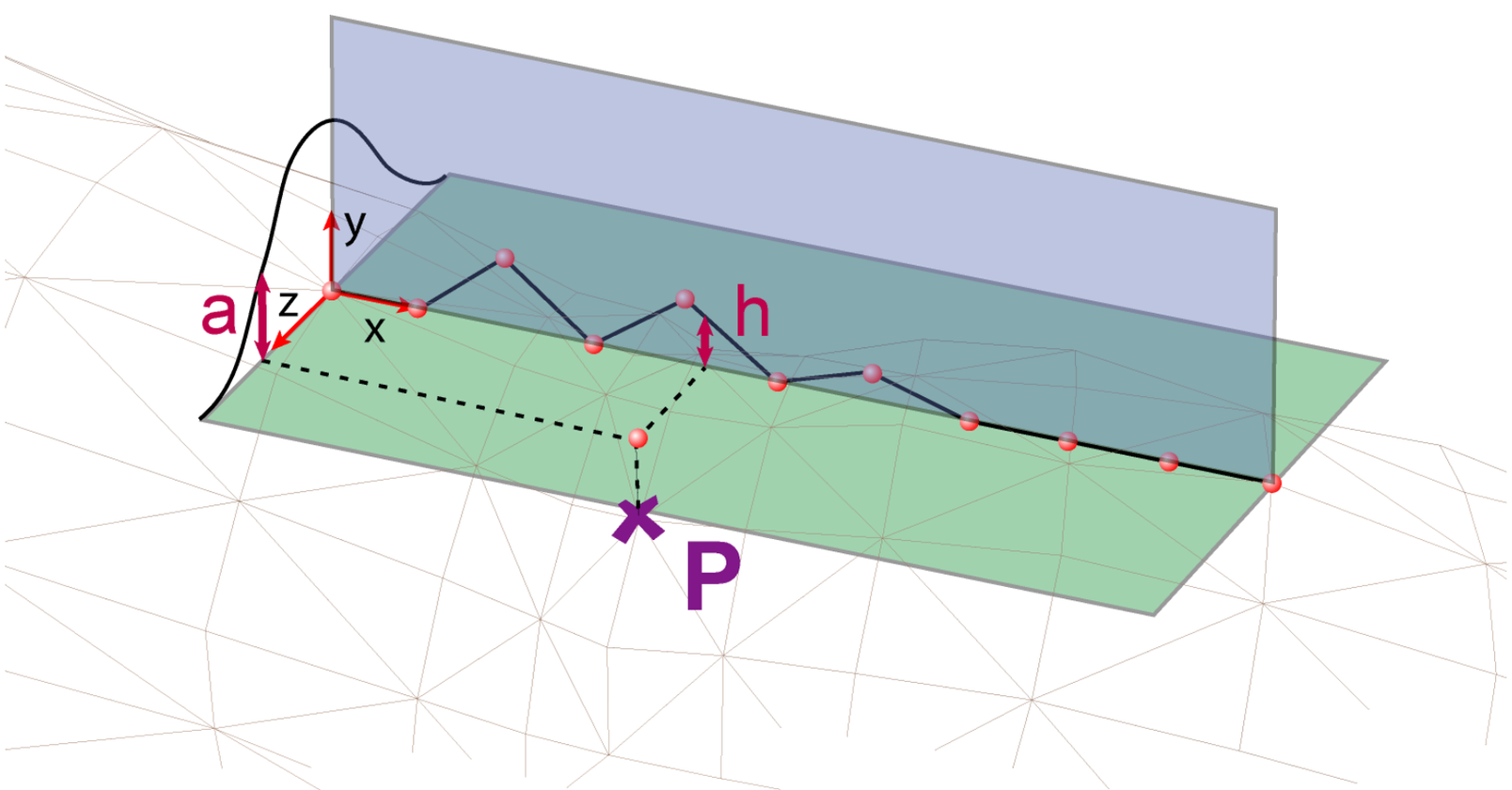
(a)



(b)

Mesh Deformation

- Mesh animated by standard skinning
- Wrinkles updated just before rendering

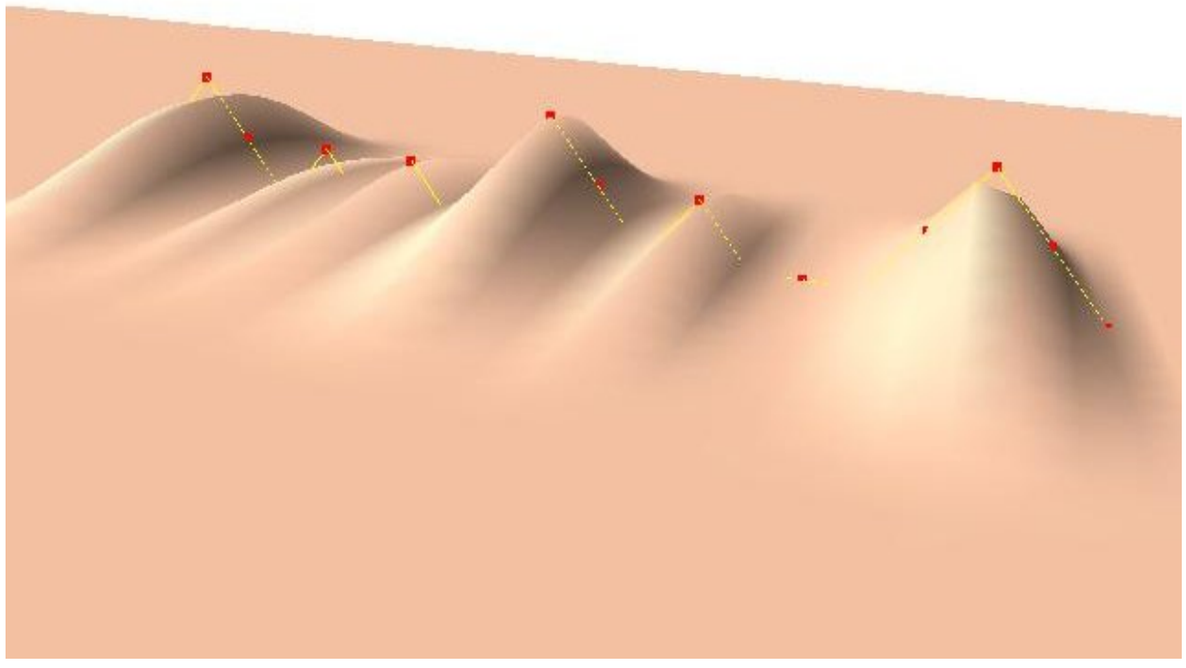


Deformation Algorithm

1. Update 2D curve due to length changes
2. Refine mesh if needed
3. Compute mesh vertices displacements
   - Height given by the x coordinate along the curve
   - Weighted according to the attenuation profile (y coordinate)

Levels of Details

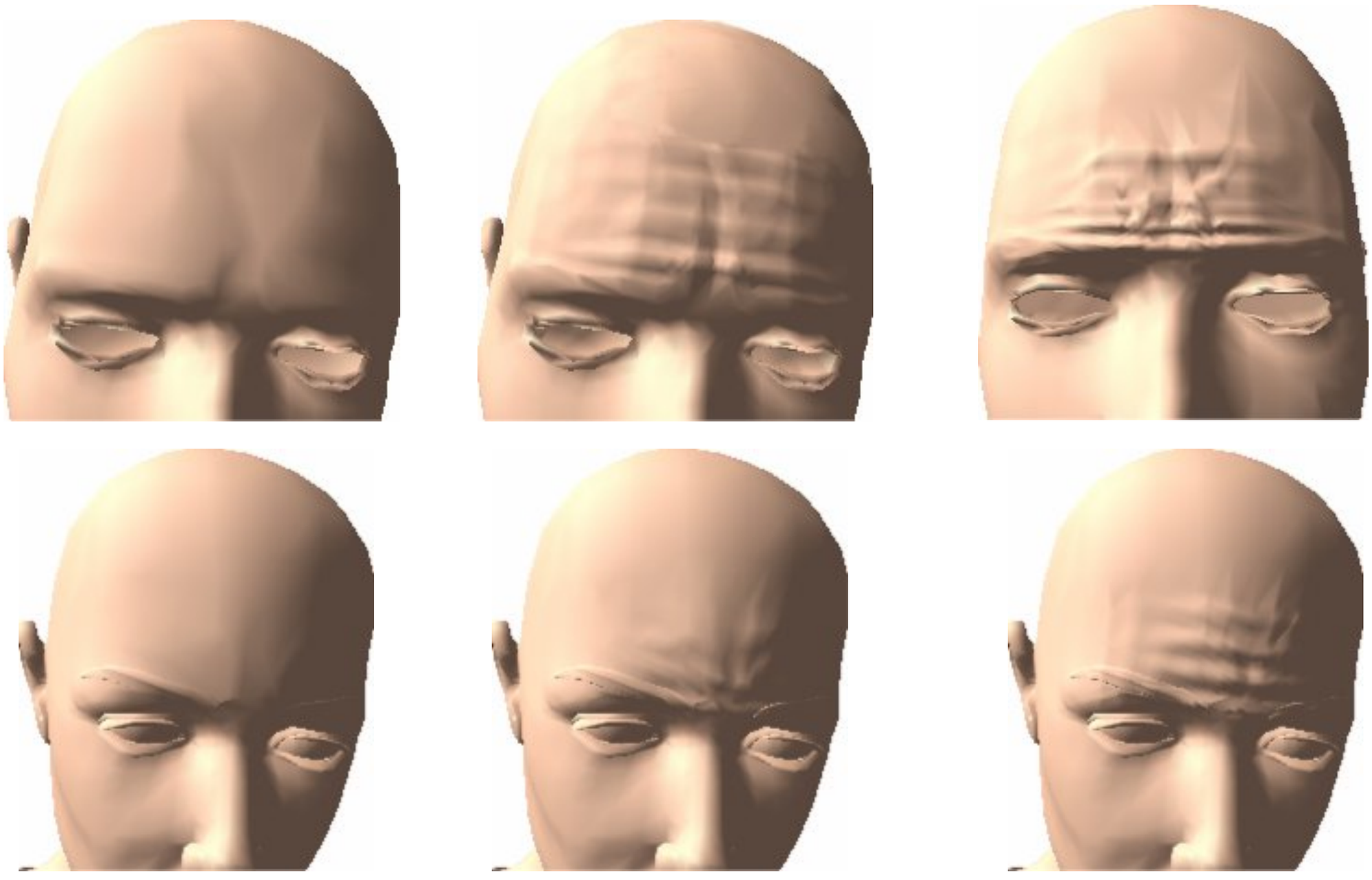
- Combination of bumps at different scales
- Absorbs compression at different resolutions



Multi-Wrinkles

- Combination of several wrinkling tools

Forehead wrinkles



Videos

This is not complete

- There are other ways to facial animation
  - Blend shape animation using mocap data (work of Clément Reverdy)
  - Animation using tracking
  - Tongue animation…
- Lip-synch is very important for realism

Hair / Rods

Some Definitions ...

- Hair strand: one single piece of hair
- Wisp: a group of strands that move together
- Scalp: the part of the head where the strands are attached

Hair in Real Life …

- Hair appearance varies depending on people and ethnicity
  - Curliness
  - Size of the cross-section (45 to 100 microns)
  - Shape of the cross-section
  - Direction in which the hair comes out of the scalp

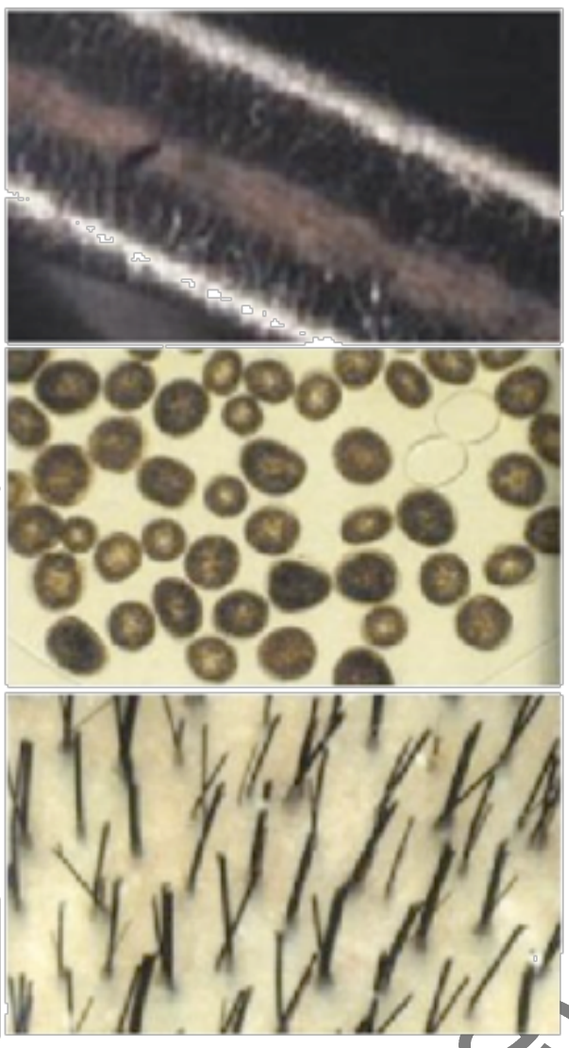
Hair Types

- Three main hair types in the world
  - Asian
  - Caucasian
  - African

Hair Types

Asian

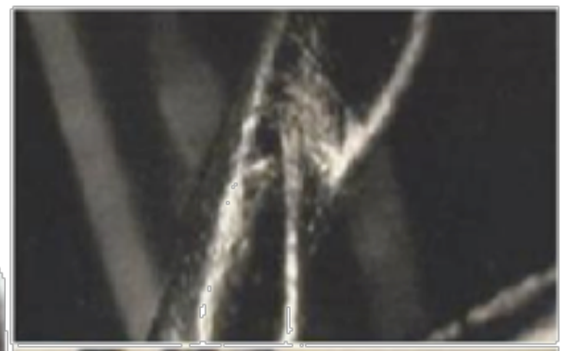
- Circular section, regular, large cross-section, straight, perpendicular implantation



Hair Types

African

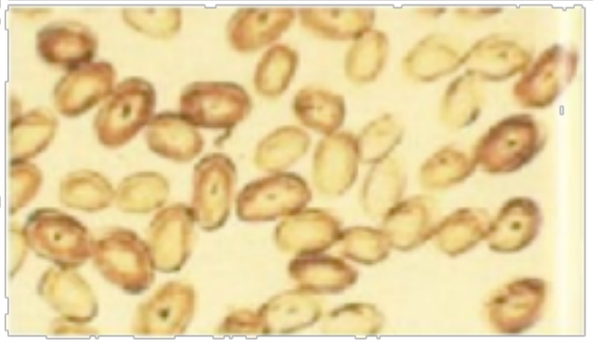
- Irregular, high curliness, twisted, elliptic cross-section, implantation parallel to the scalp



Hair Types

Caucasian

- In-between the two other types. Varying curliness, cross-section, implantation, etc …

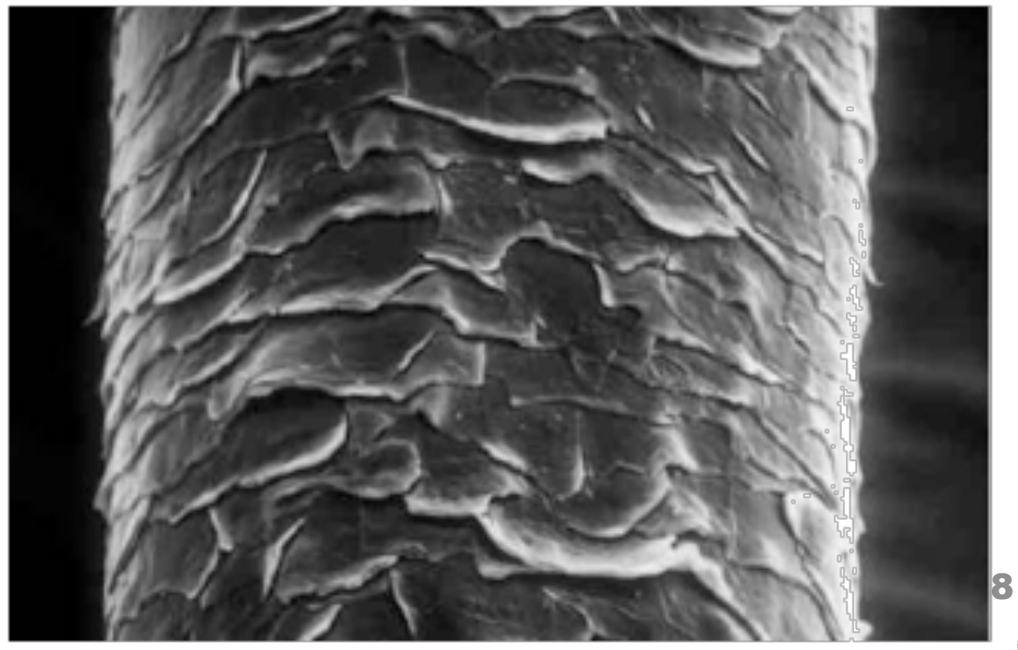


Hair strand properties (1)

- Anisotropic deformable object
- Can bend, twist but resists shearing and stretching
- Elastic properties (recovers its original shape after stress has been removed)

Hair strand properties (2)

- Interactions between hair strands are very complex
  - Surface composed of tilted scales
  - Anisotropic friction inside hair with an amplitude that depends on the
  - orientation of the scales and the
  - direction of motion



Creation Pipeline

1. Modeling
    1. Attach hair to the scalp (points)
    2. Create a global shape (wisps)
    3. Add finer details and properties (strands)
2. Animation

Attaching hair to the scalp

- High number of individual strands (250 per cm2)
  — > cannot be placed by hand (150 000 total)
- 2D placement: the hair are painted on a map that is then projected onto the head
  — Pbs of distorsions
- 3D placement: the user select the triangles that are part of the scalp, a wisp is generated for each triangle
- Create a uniform distribution of hair on the scalp and then assign each strand to a wisp
- Paint local density of hair on the scalp

Example of 2D placement

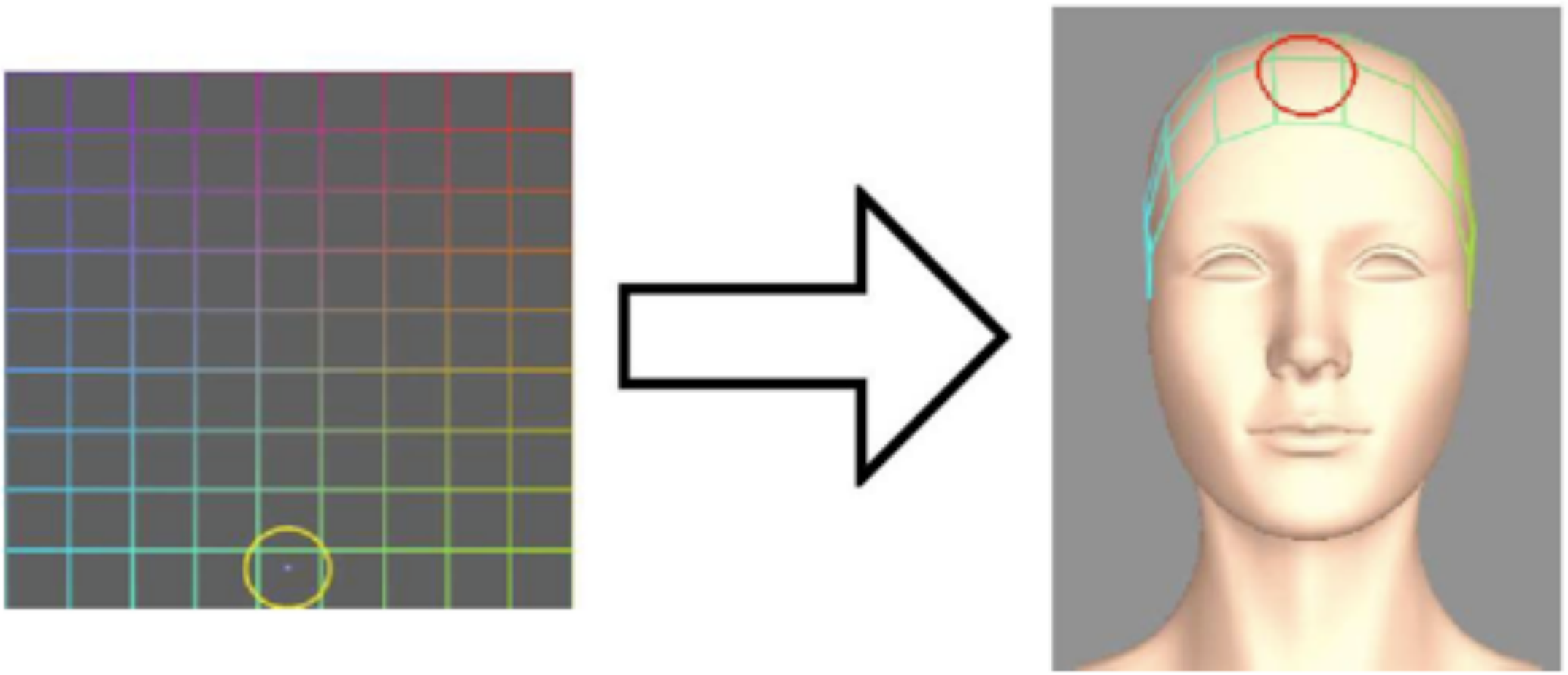


Fig. 1. 2D square patch wrapped onto the 3D model by the method of Kim *et al.* [9].

Generating Global Shape

- A desired global shape must be generated from the wisps
- 3 common approaches
  - Geometry based techniques
  - Physically based techniques
  - Image-based techniques

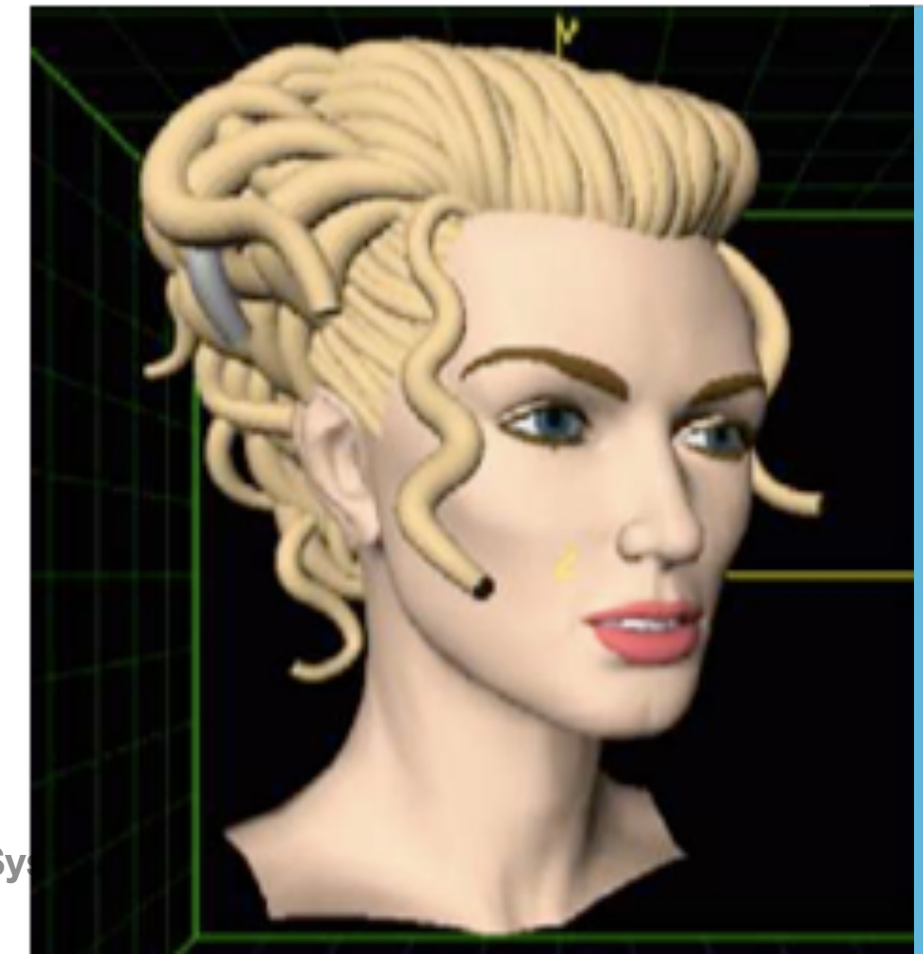
Generating Global Shape
Using NURBS

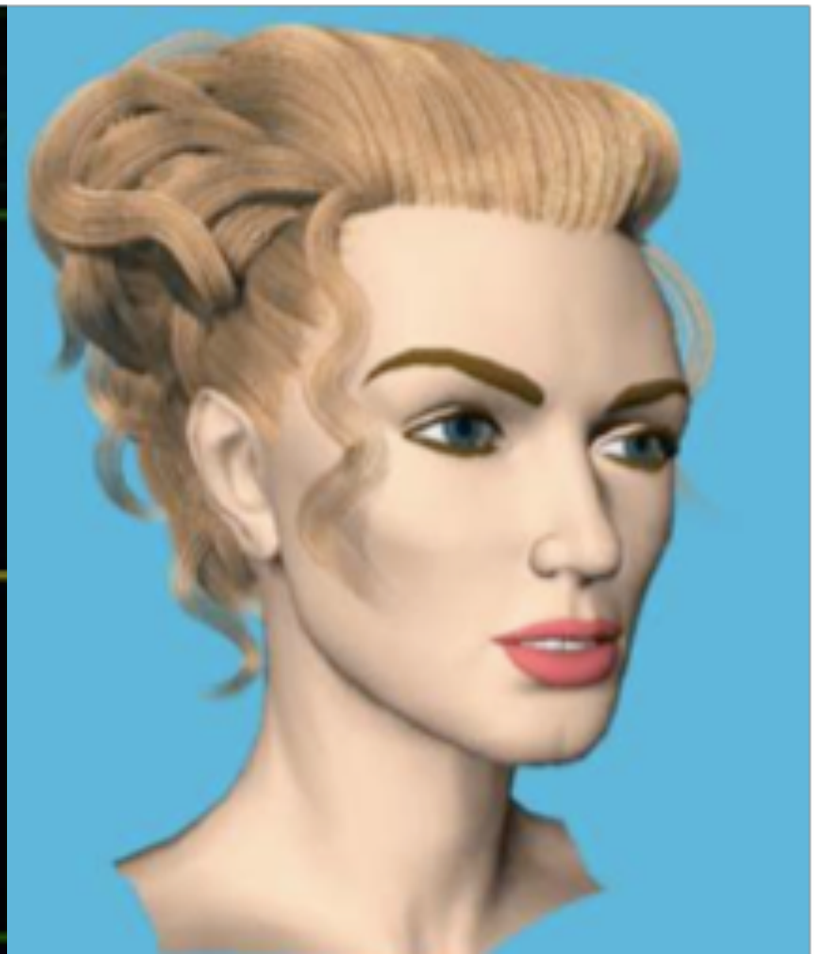
- Also called hair strips
- Strips are attached to the scalp
- Thickness is added by offsetting the surface along its normal direction
- Individual hair strands are then created inside

Generating Global Shape

Using Generalized Cylinders

- For each wisp, a curve is placed
- A cylinder is generated from the curve
- Hair strands are distributed inside the cylinder





Generating Global Shape

Adding Finer Details

- Local properties (curls, waves, noise)
  - To create more realistic appearance
  - To capture additional features (effect of water)
- Using trigonometric offset functions
  - Magnitude, frequency, phase + randomness

[Yu01]

[Choe05]

Generating Global Shape
Multi-Resolution Techniques

- Using a hierarchy of cylinders
  - Easier to model
  - Can be done up to the hair strand level
  - Can use copy/paste

Generating Global Shape

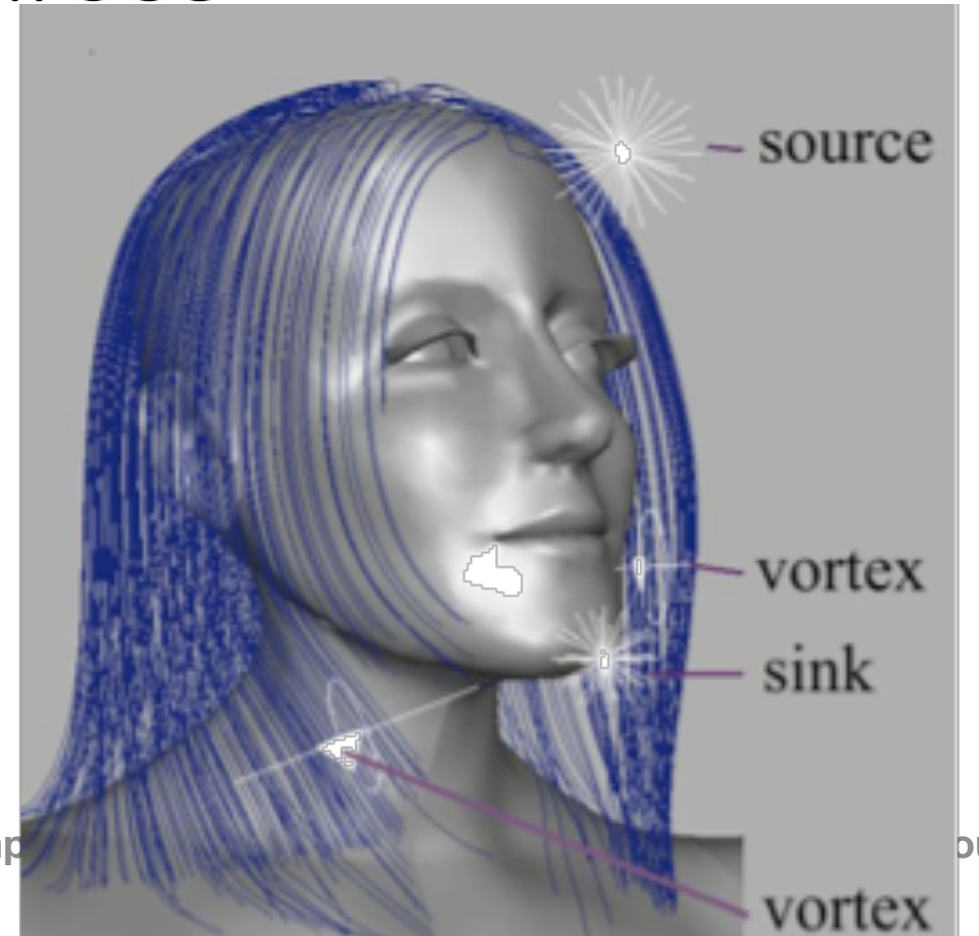
Physics Based

- Cantilever beam (straight beam embedded in a fixed support at one end only)
- A cantilever is attached to the scalp and a simulation under gravity is performed
- Need extra forces to get a proper final shape

Generating Global Shape

Fluid Flow Method

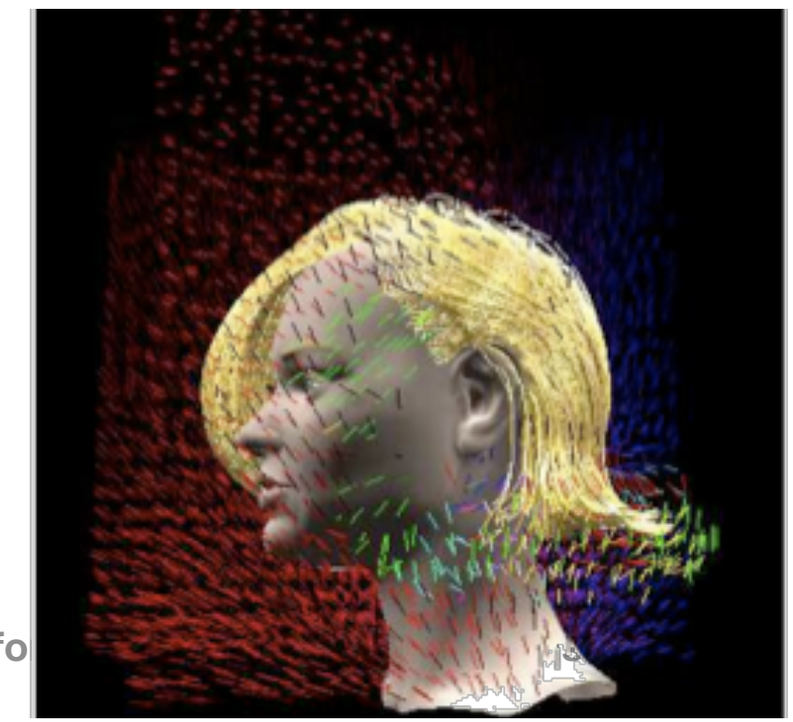
- Idea: static hair shapes resemble snapshots of fluid flow around obstacles
- User places streams, vortices (for curls) and sinks / sources



Generating Global Shape

Using Motion Fields

- Procedurally generated field using predefined vector field primitives and constraints (point, trajectory, direction)
- Hair strands are generated by tracing the field lines of the vector field



Generating Global Shape

Image-based techniques

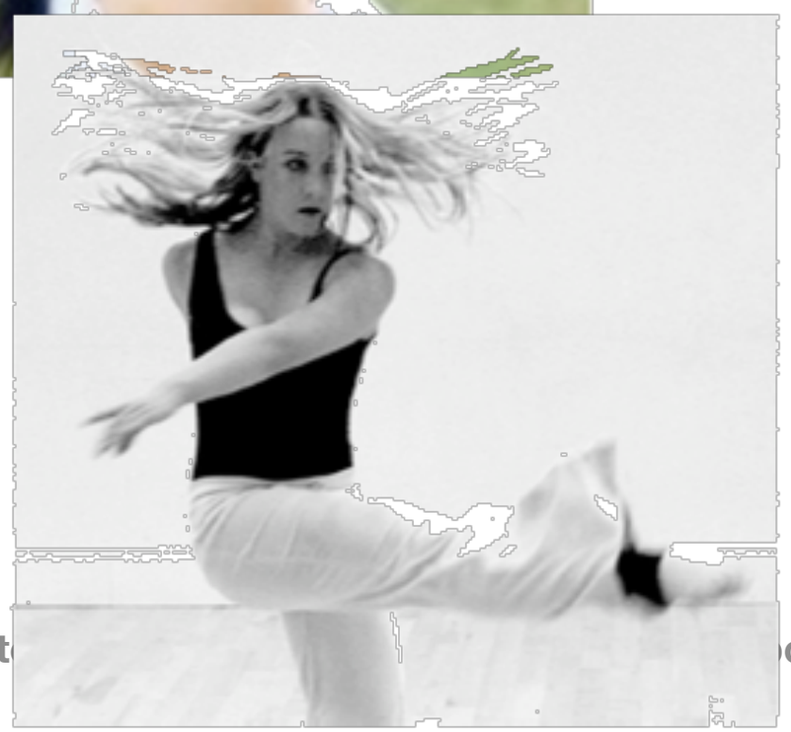
- From photographs, sketches
- Idea: capture the local hair orientation by studying the reflectance of the hair under various illumination conditions

— > captures local orientation of the visible part of hair

Animation

- Over 100 000 hair strands on a human head
  - Individual animation of strands (almost) not possible
  - Strands tend to move in a similar way as their neighbors
  - > Often, wisps are animated

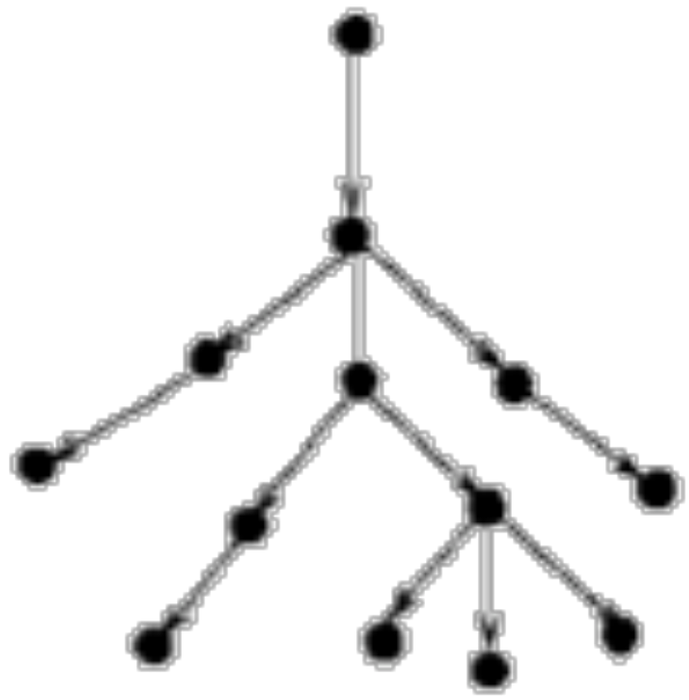
Animation ... in real life



Animation

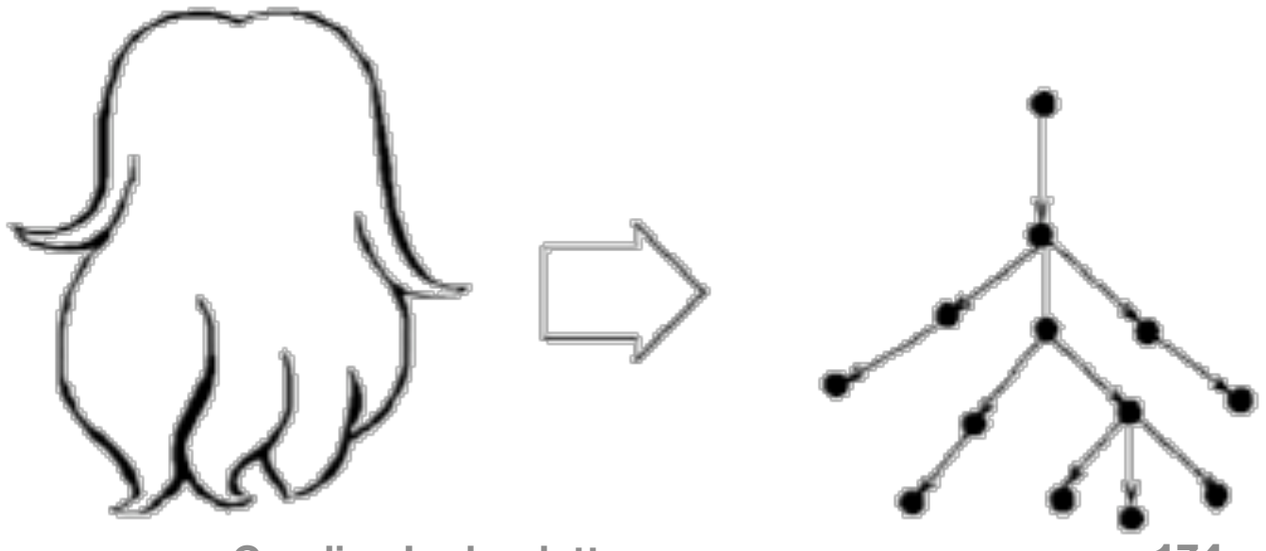
Mass-springs model (1)

- Uses a multi-resolution approach of the hair strands
- Stored in a tree structure



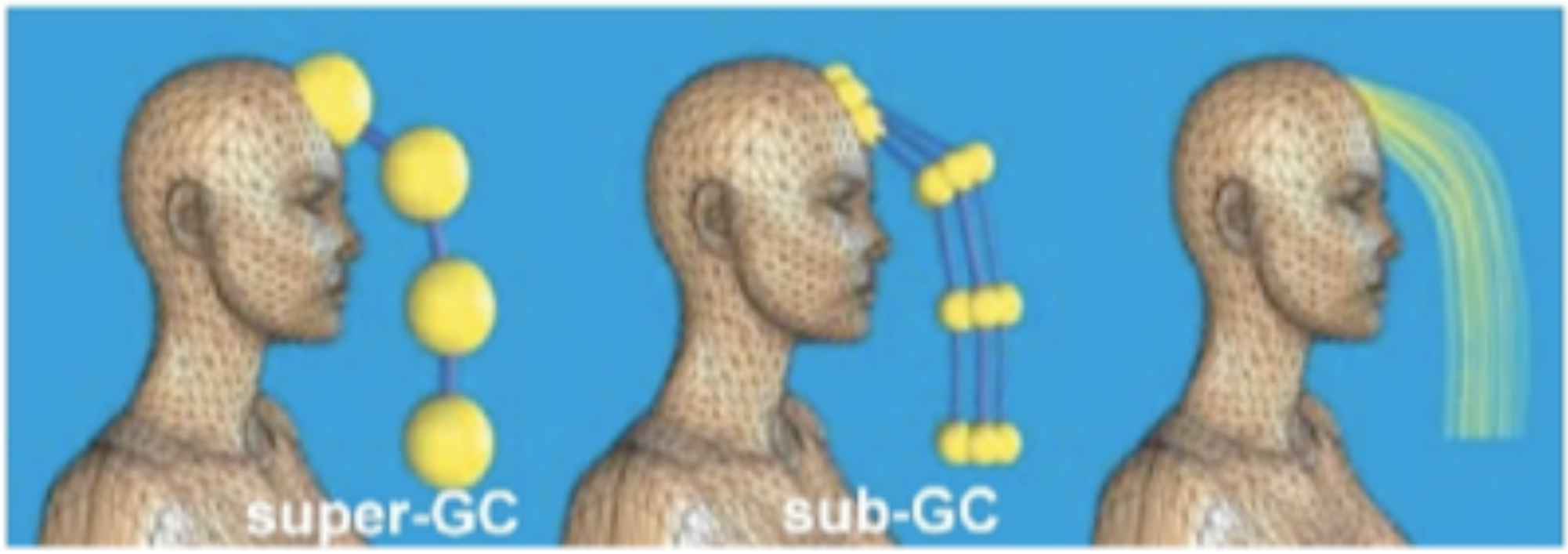
Mass-springs model (1)
Wisp-Tree Technique

- Masses (depends on the number of hair strands) are stored in the nodes and Springs connect nodes
- This “tree” is used to predict how wisps are gonna separate during animation



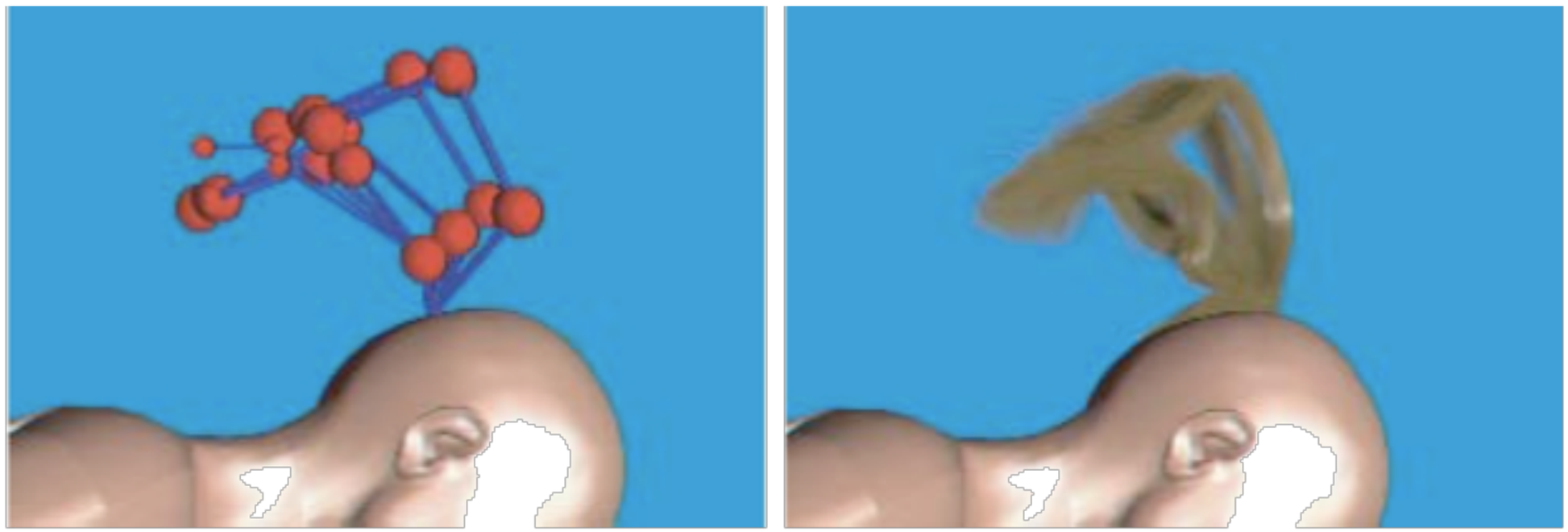
Levels of Details

- The hair model exists at different levels of details
- At a given time t, only one LOD is active



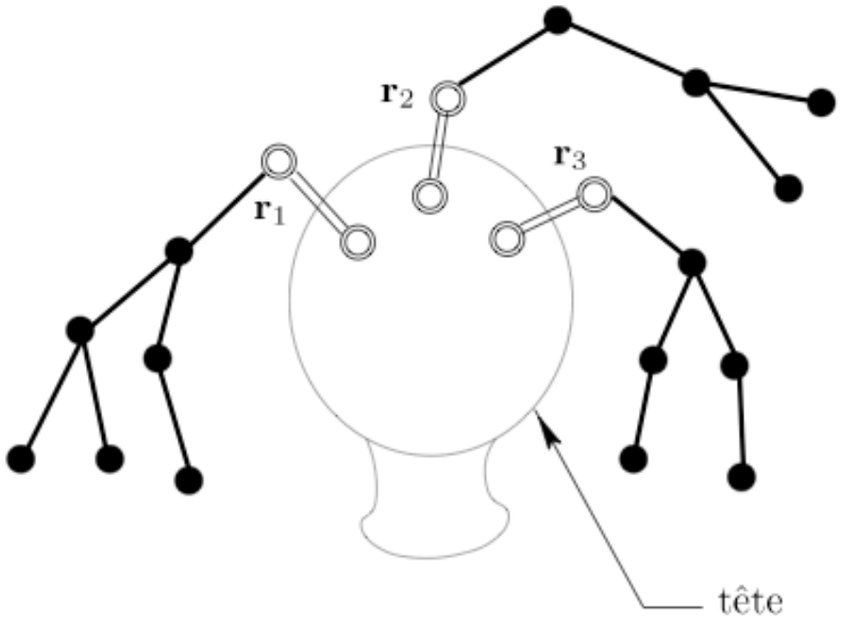
Segments

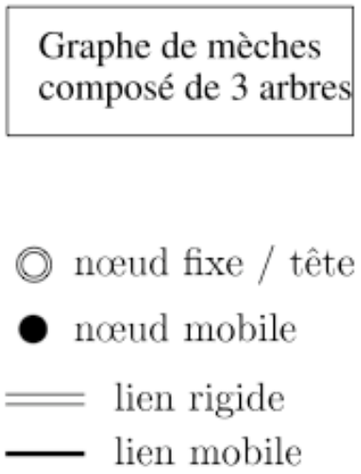
- The segments between two spheres (blue cylinder) are modeled by springs
  - High stiffness for straight hair
  - Low stiffness for curly hair

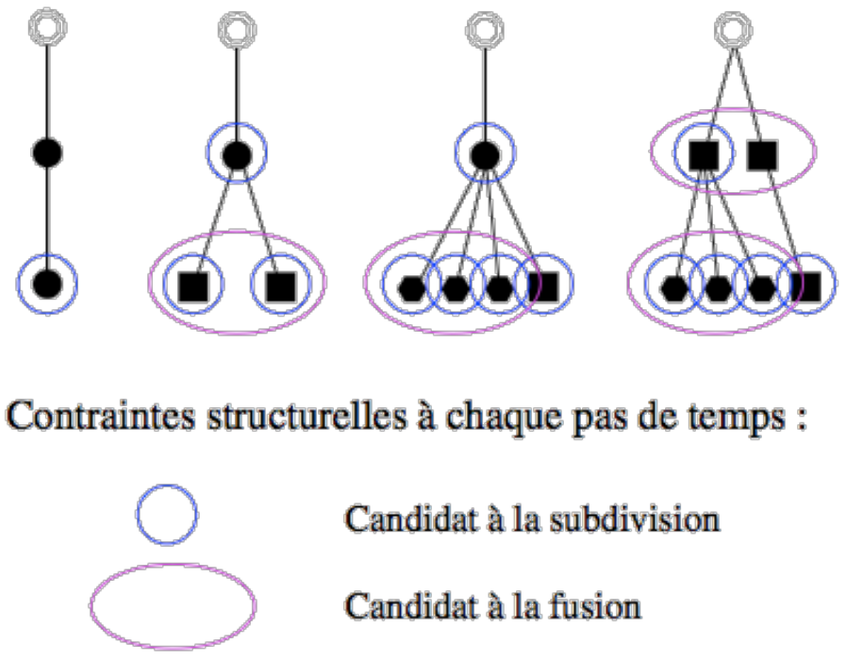


Grouping/Degouping

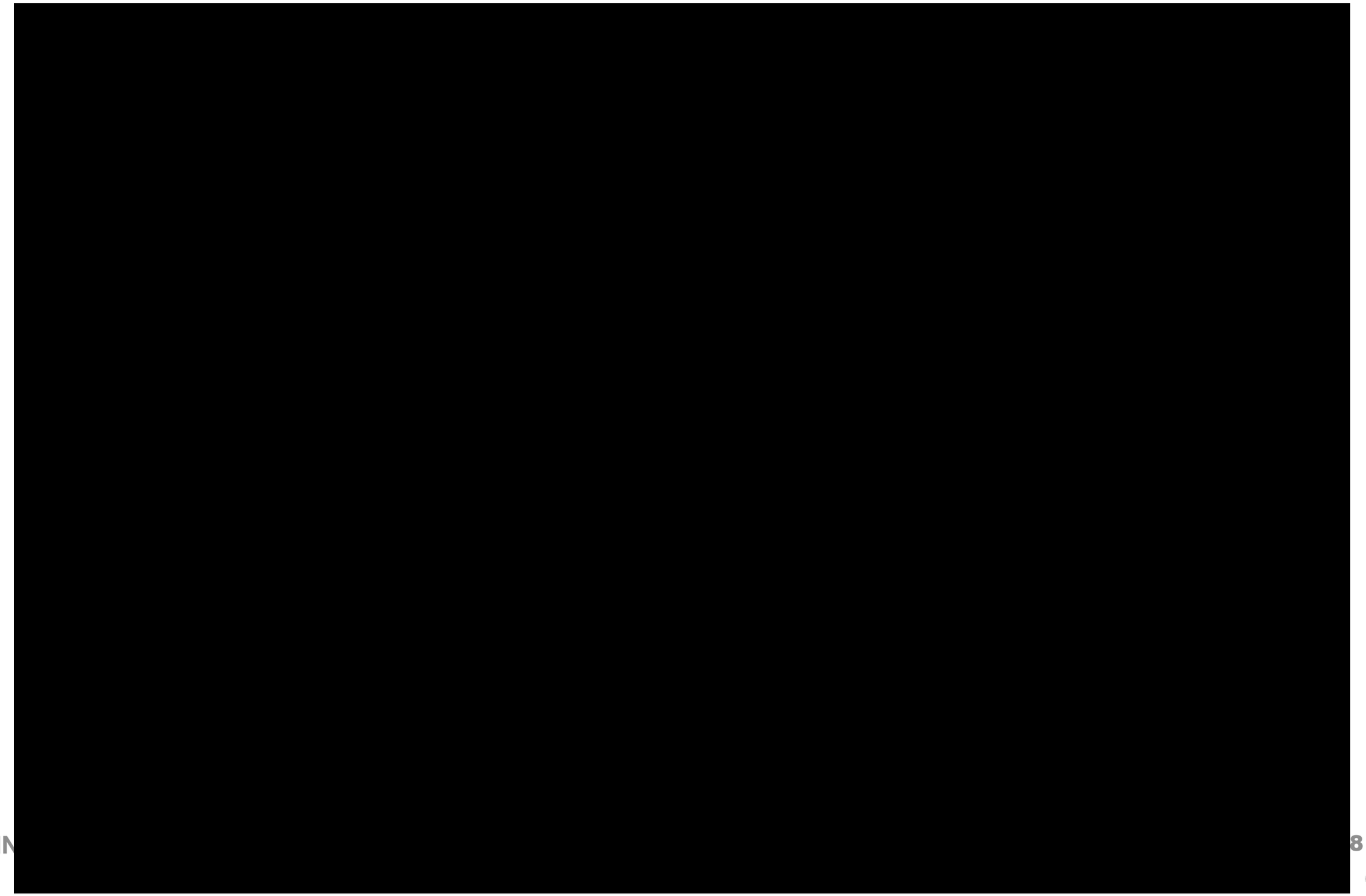
- During animation
  - If node acceleration > threshold, then ungroup (ie, use lower LOD)
  - If “brother” nodes are close (distance) and have close velocities, then group







Results : single AWT



Results

Dynamic Super-Helices (Rods)

- Model based on the Cosserat Curves

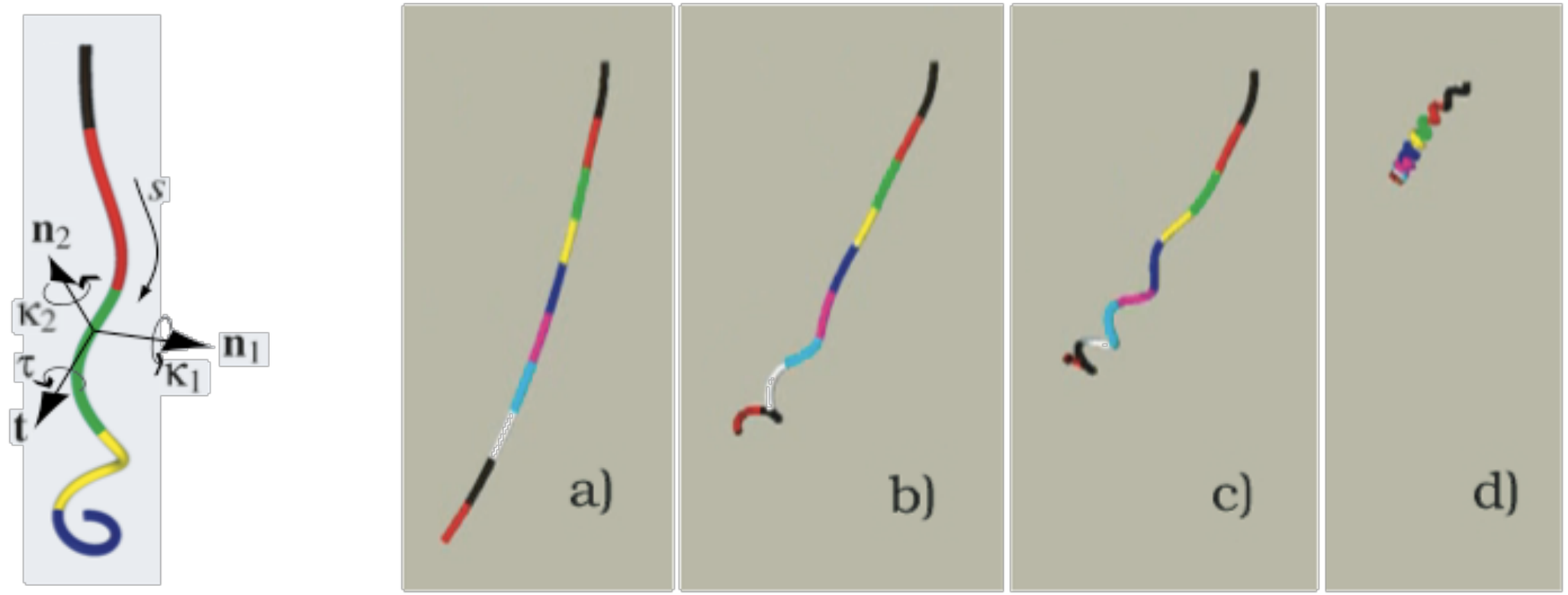
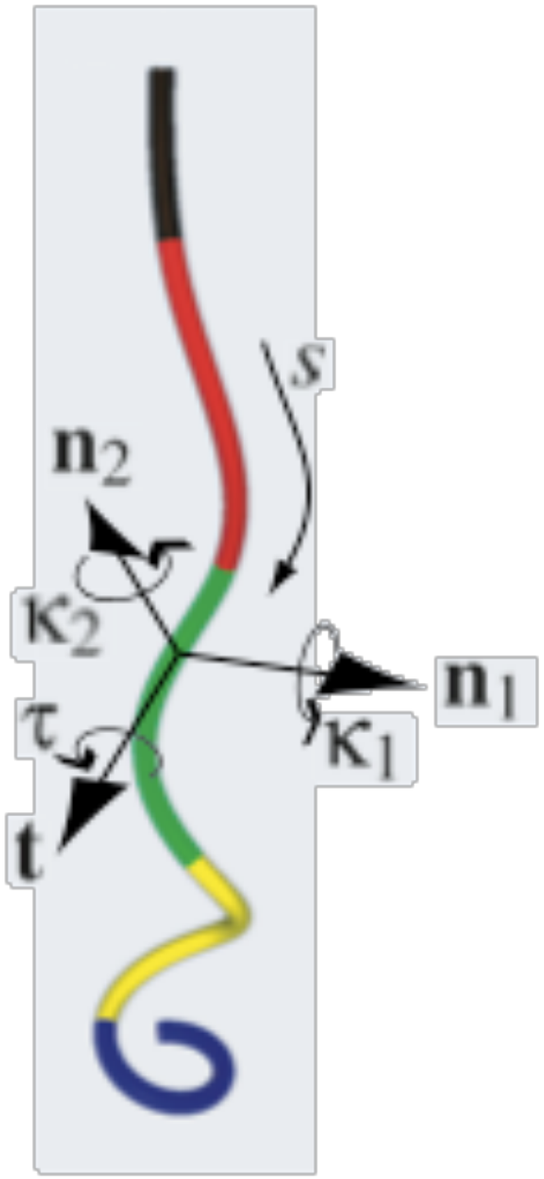


Figure 3: Left, geometry of Super-Helix. Right, animating Super-Helices with different natural curvatures and twist: a) straight, b) wavy, c) curly, d) strongly curly. In this example, each Super-Helix is composed of 10 helical elements.

Cosserat Model



- Centerline r(s,t) = curve passing through the center of mass of each cross section
- n0 = tangent + Tau describe the twist around the centerline
- n1, n2 + K1, K2 describe the material curvatures
- r(s=0,t) is attached to the scalp
- r(s=L,t) is the end point with constant length

Cosserat Model

Parameters

- Hair mass and stiffness
- Twist and curvature (curliness of hair)
- Ellipticity of cross section and length
- Friction (internal + with air)

Results

Results

Results

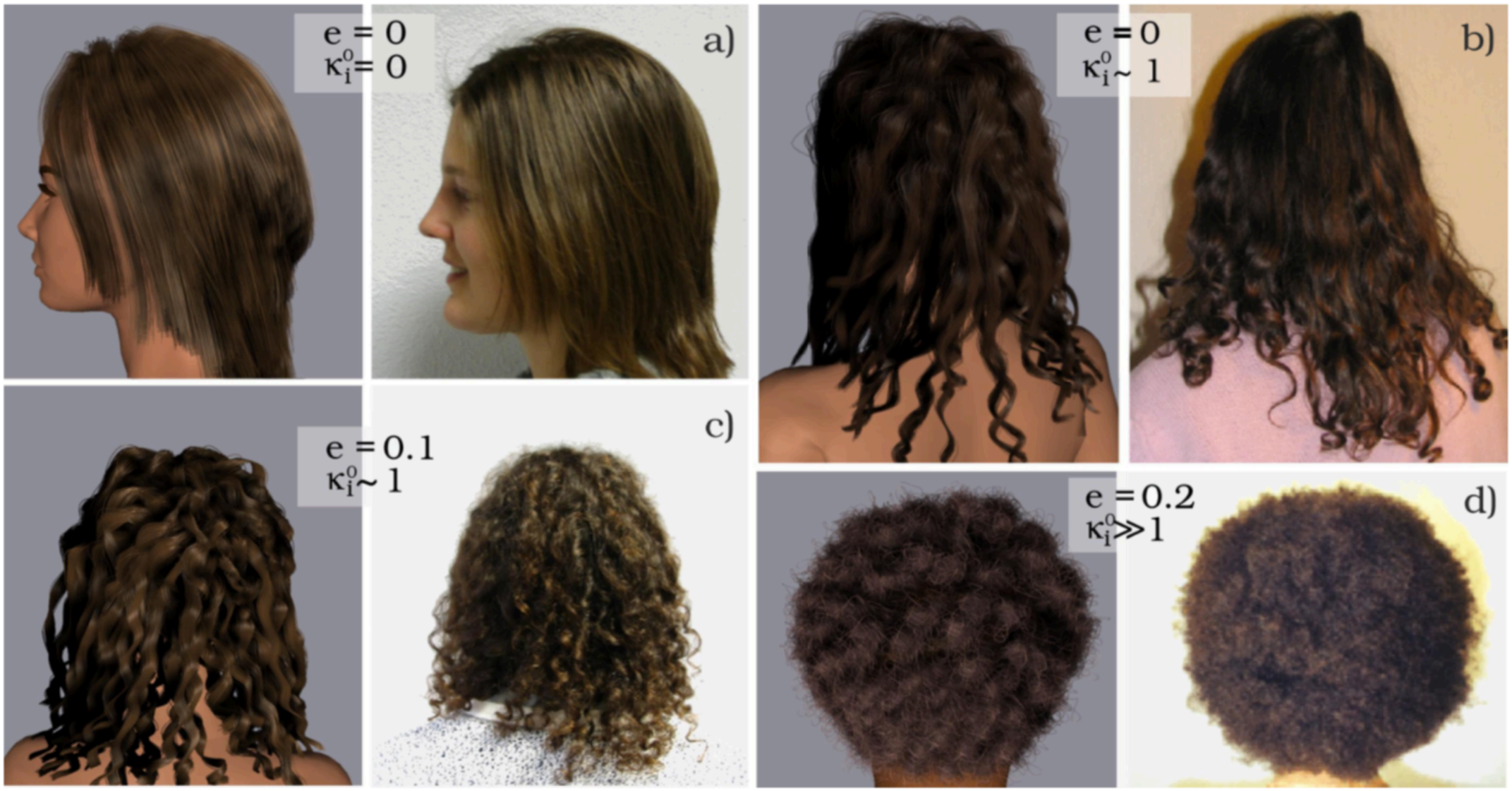
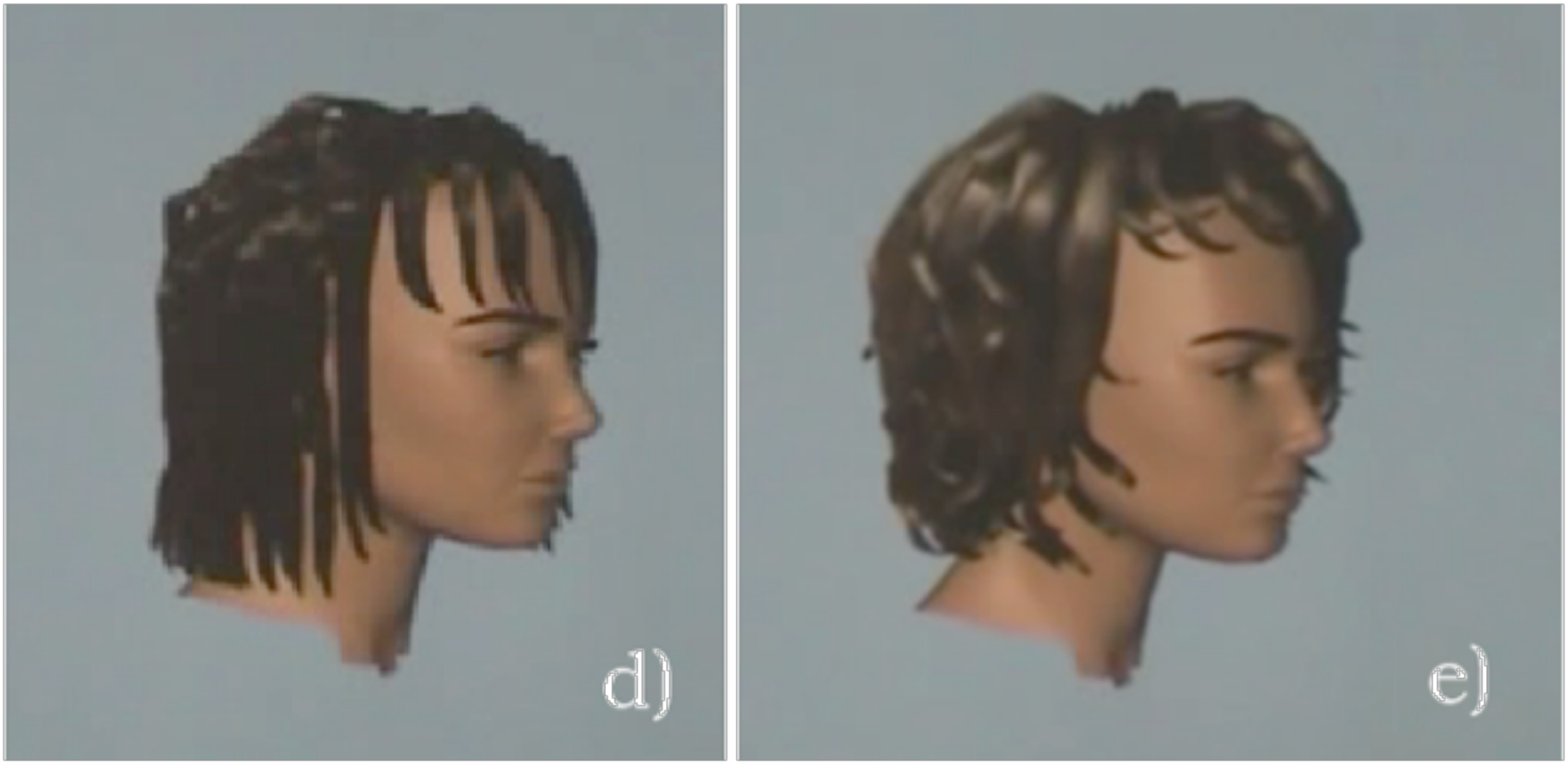


FIG. 6.12: *Comparaison entre des coiffures générées par notre logiciel et des coiffures réelles. À droite de chaque couple d'images : images réelles de cheveux (a) raides, (b et c) bouclés et (d) crépus. À gauche : résultats synthétiques correspondants, générés à partir de valeurs adéquates pour l'excentricité de la section du cheveu e et pour les courbures naturelles κ_i^0. Chacune des coiffures de synthèse a été réalisée à l'aide de notre logiciel en moins de 30 minutes.*

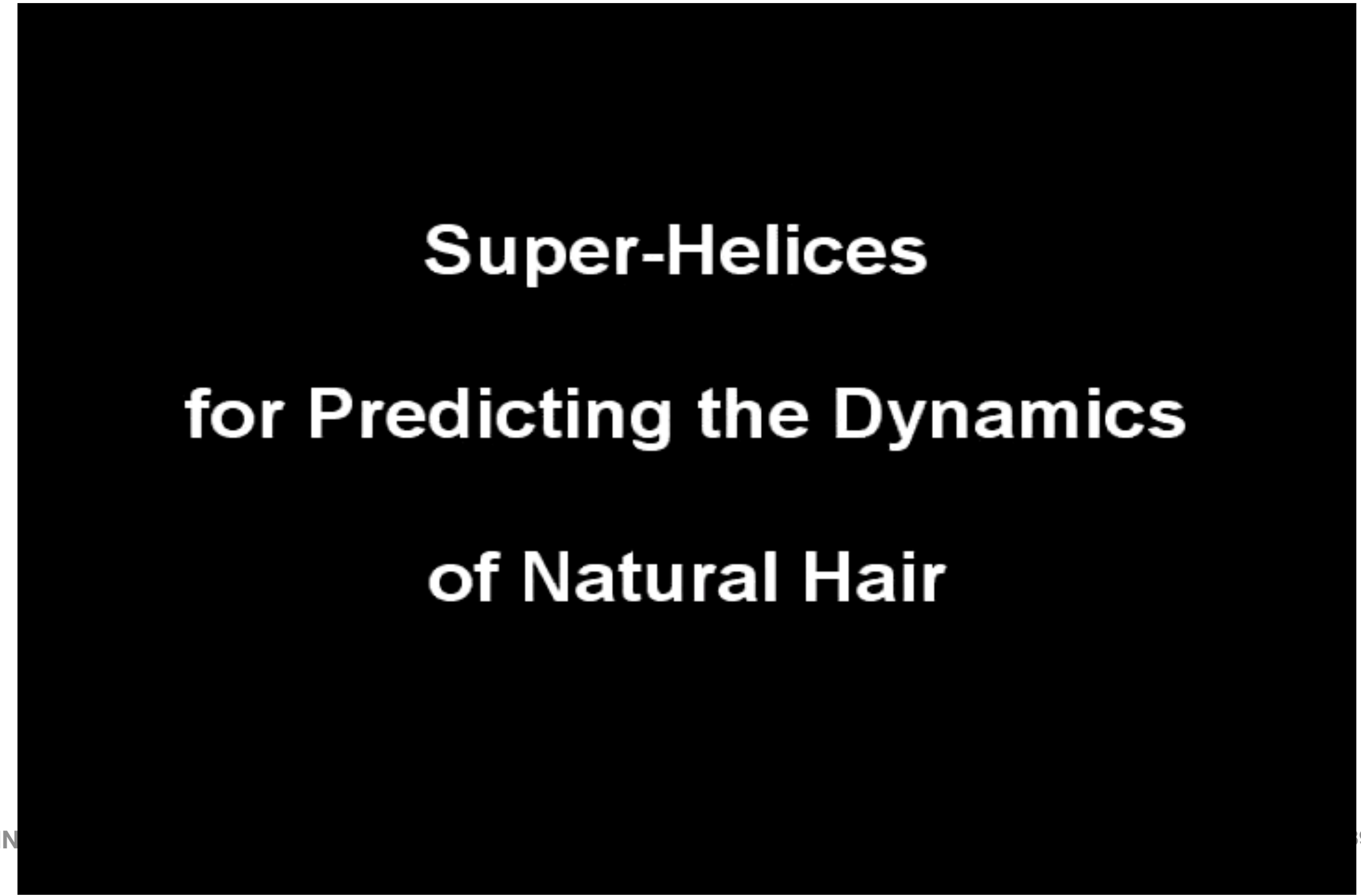
Wet hair

Cut hair

Dry hair



Results



Beard of Merlin - Shrek the 3rd

- One dynamic hair strand
- 300 guide hair strands passed to the hair shader

- Deformation system divided in 3 regions
  - Hair near the face are deformed by the motion of the face
  - Hair further down are deformed by the dynamic strand
  - Hair inbetween are blended

Merlin - Shrek

Mass-Spring Model for Strands

Selle A. et al., A Mass Spring Model for Hair Simulation, SIGGRAPH 2008

- Idea: no wisp simulated, but individual hair strands
- The hair is sampled into n discrete points equally spaced in arclength
  - Additional subdivision + perturbation to prevent colinearity
- Springs are connected into tetrahedra

Tetrahedra

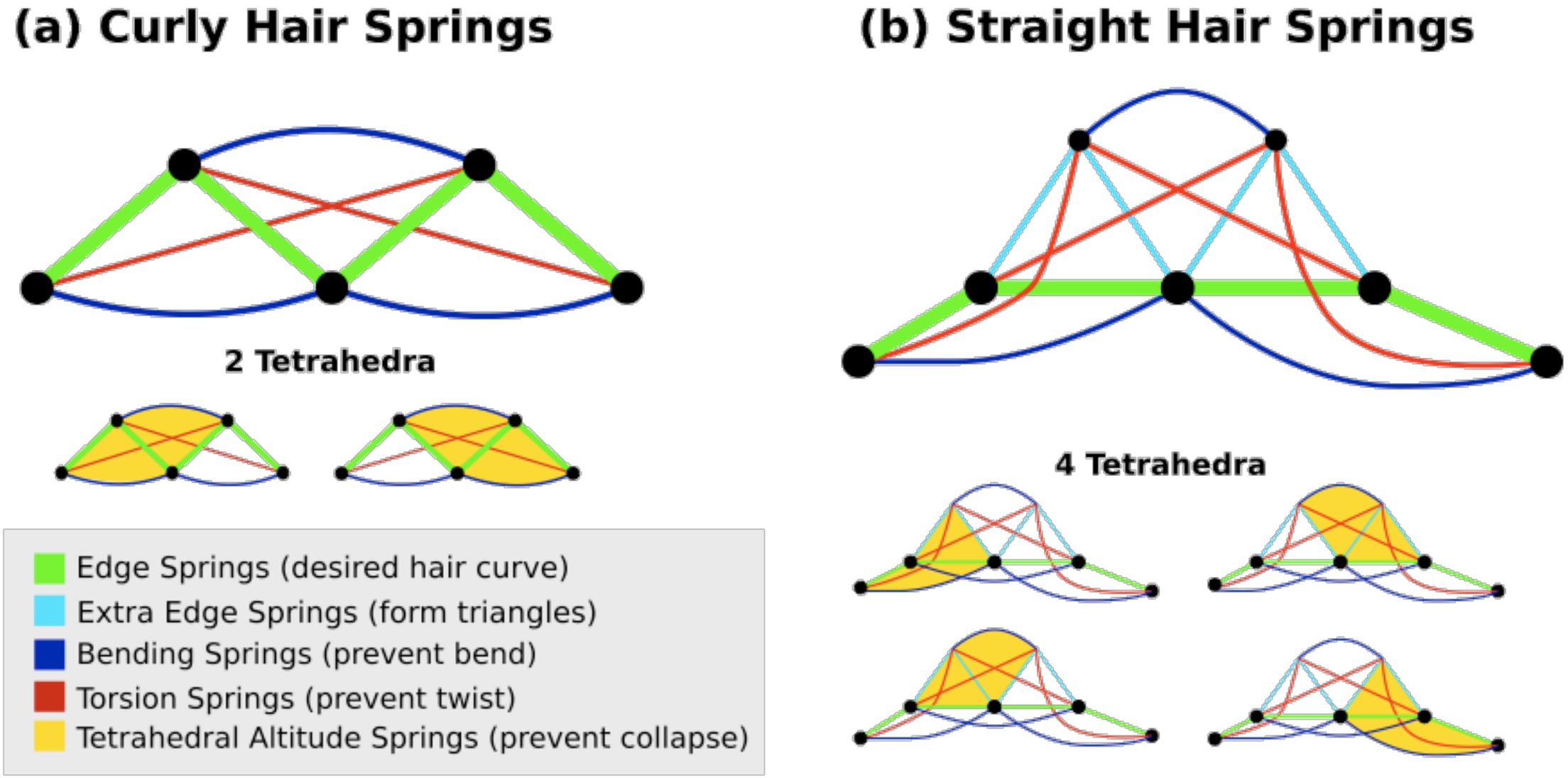
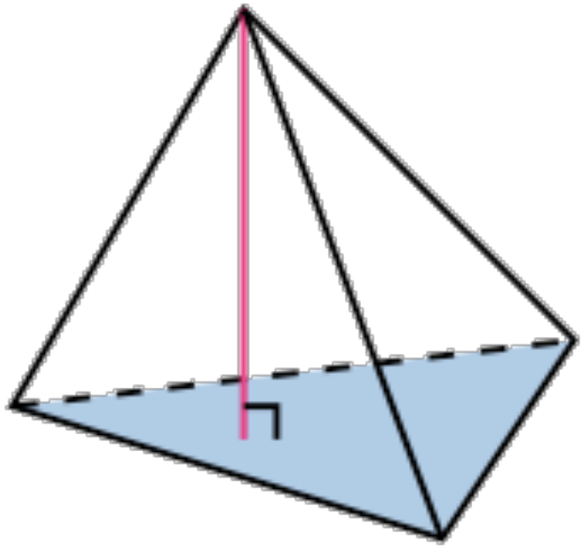


Figure 7: *Straight and curly hair models using edge, bending, torsion, and altitude springs preserving the implied tetrahedra.*

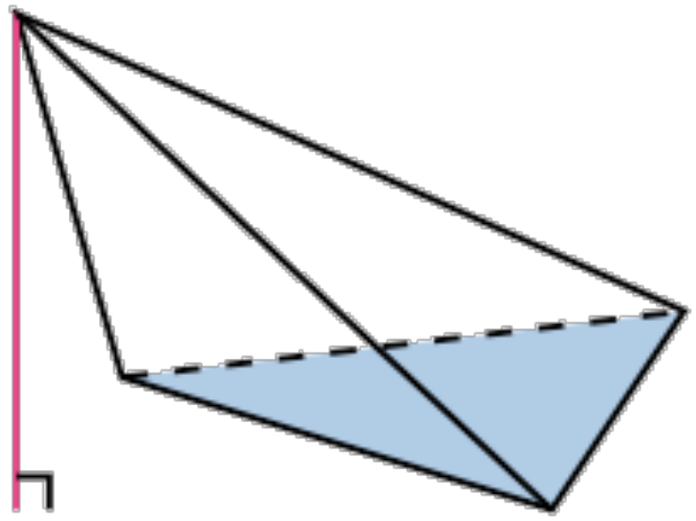
Altitude springs

- Point/Face and Edge/Edge Altitude springs are used to prevent collapse

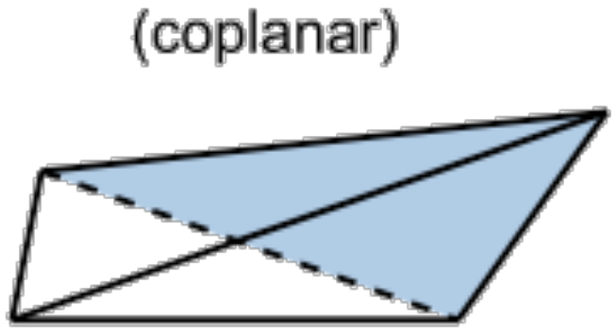
Point/Face Altitude Springs



(a) Spring has all non-negative barycentric weights



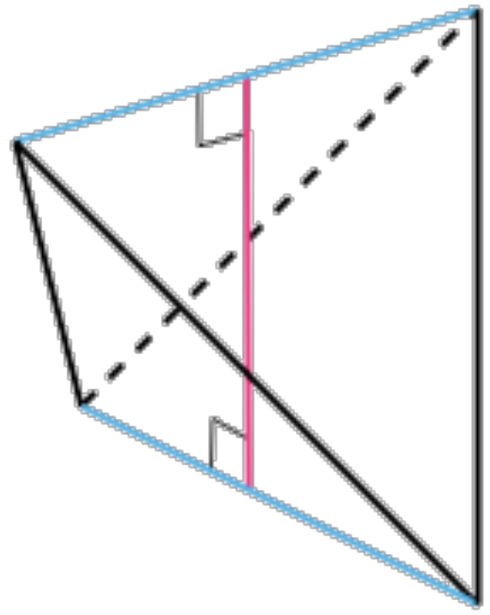
(b) Spring has negative barycentric weights



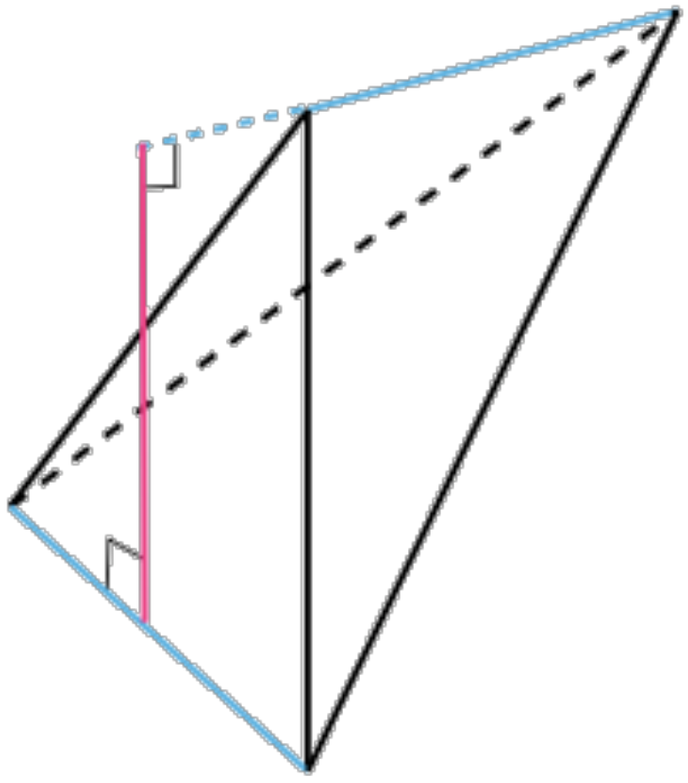
(c) Degenerate: all point/face springs have negative barycentric weights

Altitude springs

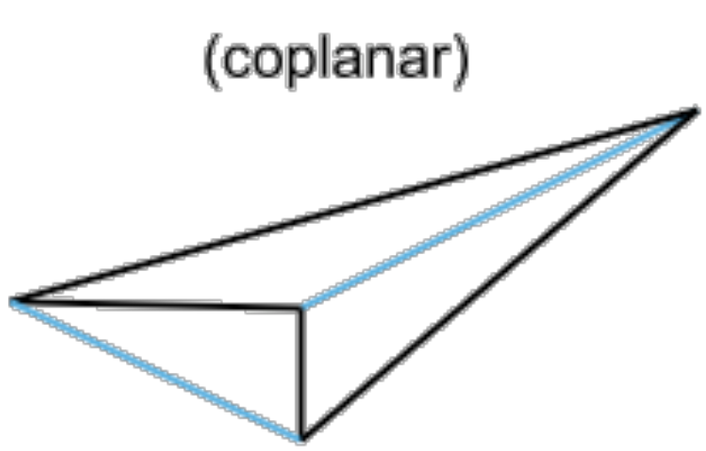
- Point/Face and Edge/Edge Altitude springs are used to prevent collapse



(d) Spring has all non-negative barycentric weights

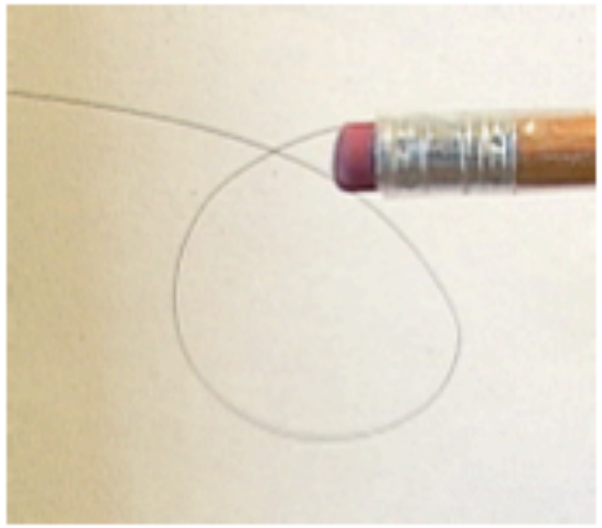
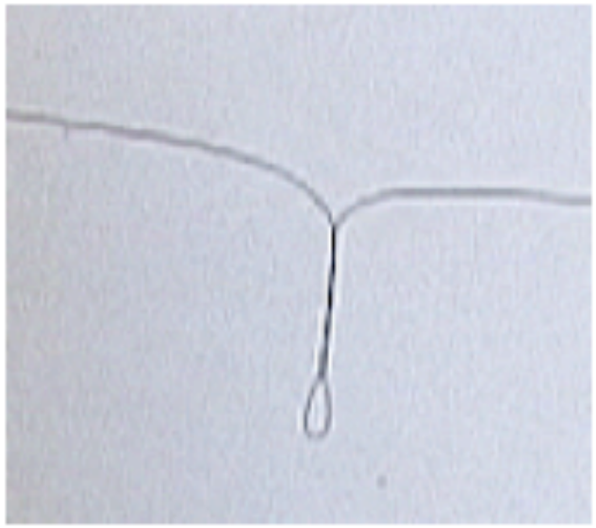
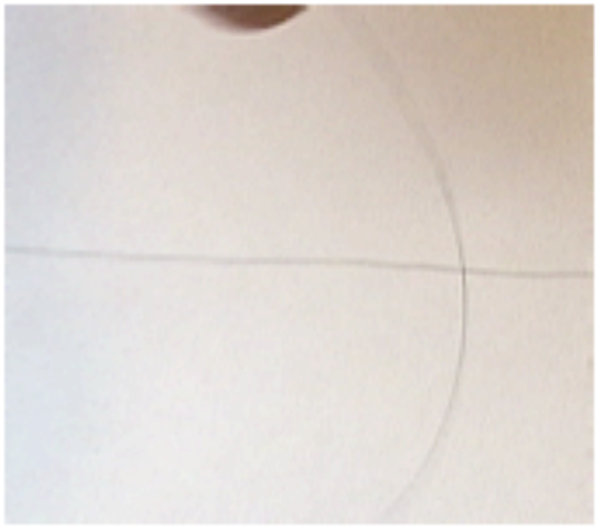


(e) Spring has negative barycentric weights



(f) Degenerate: all edge/edge springs have negative barycentric weights

Results on one Hair Strand




Collisions Detection/ Response

- Collision detection with the body is done using level-sets (implicit function)
- Hair-Hair collision is done on an edge-edge collision detection
  - If two edges are closed, they are identified
  - A spring of rest length epsilon is added
  - Maximal number of springs per edge

Results

- Hairs: 5 000 to 10 000
- Particles 250 000 to 1 million (ie 100 particles per hair)
- Time: a few minutes to a few hours per frame
- Videos

Grass / Tree

Botany ...

- Stem
- Node
  - Internode
- Bud
- Leaf
- Flower

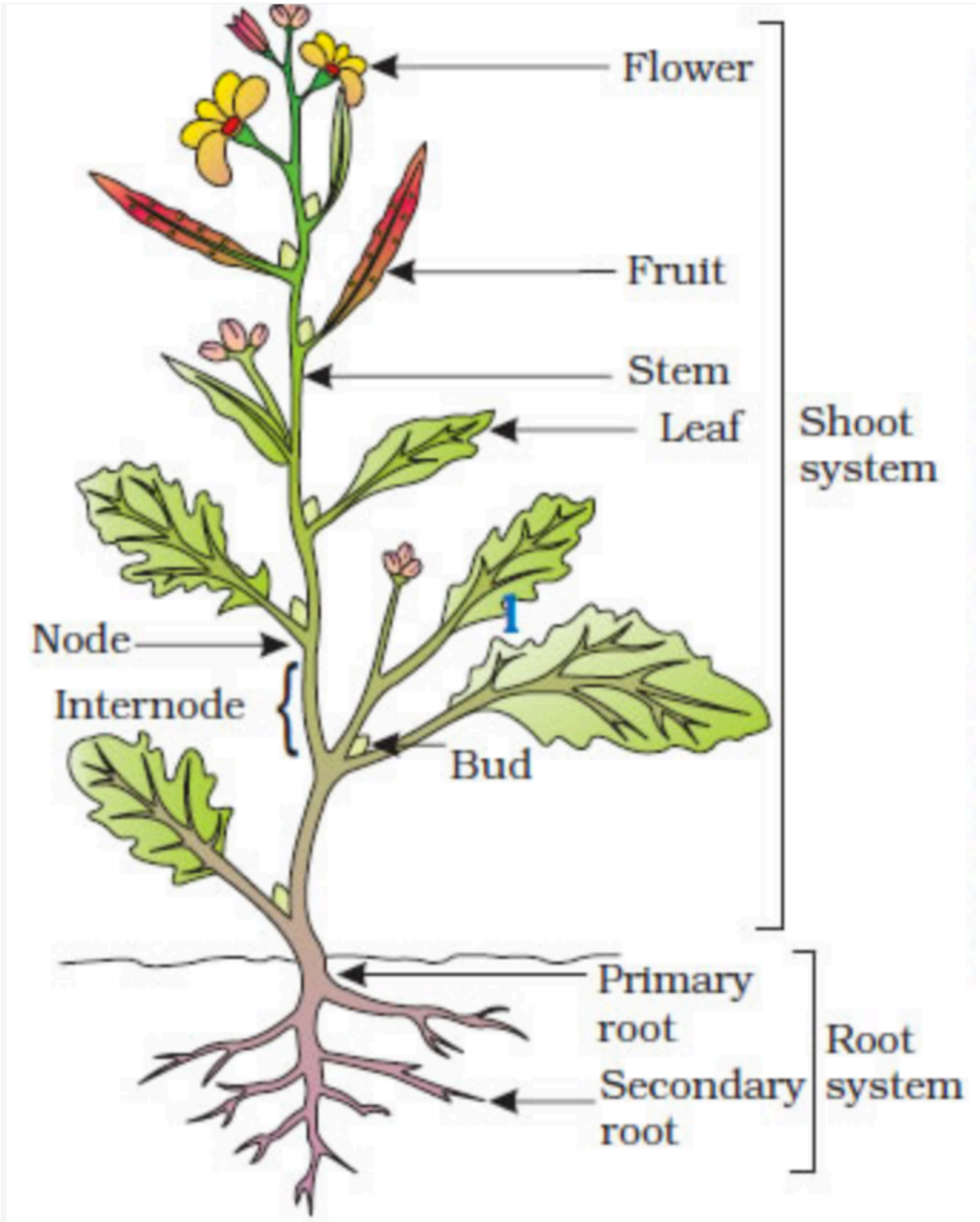


Figure 5.1 Parts of a flowering plant

Plants

- **Part I : Modeling and Growth**
- **Part II : Animation (swaying in fluid)**

Modeling of Plants

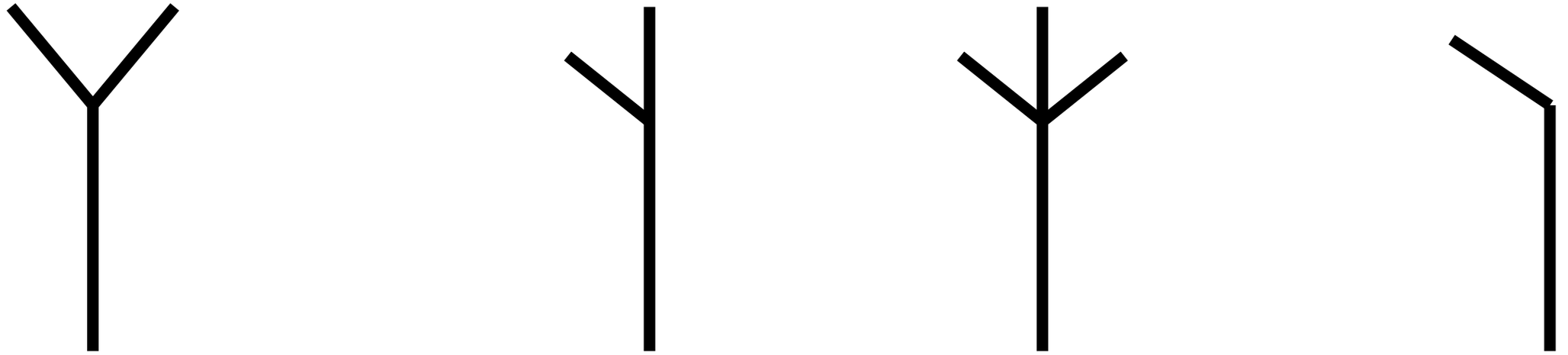
- Topology: recursive branching structure
- Can be modeled by
  - Fractals
  - Particle systems
  - L-systems (can handle growth too)

Modeling of Plants

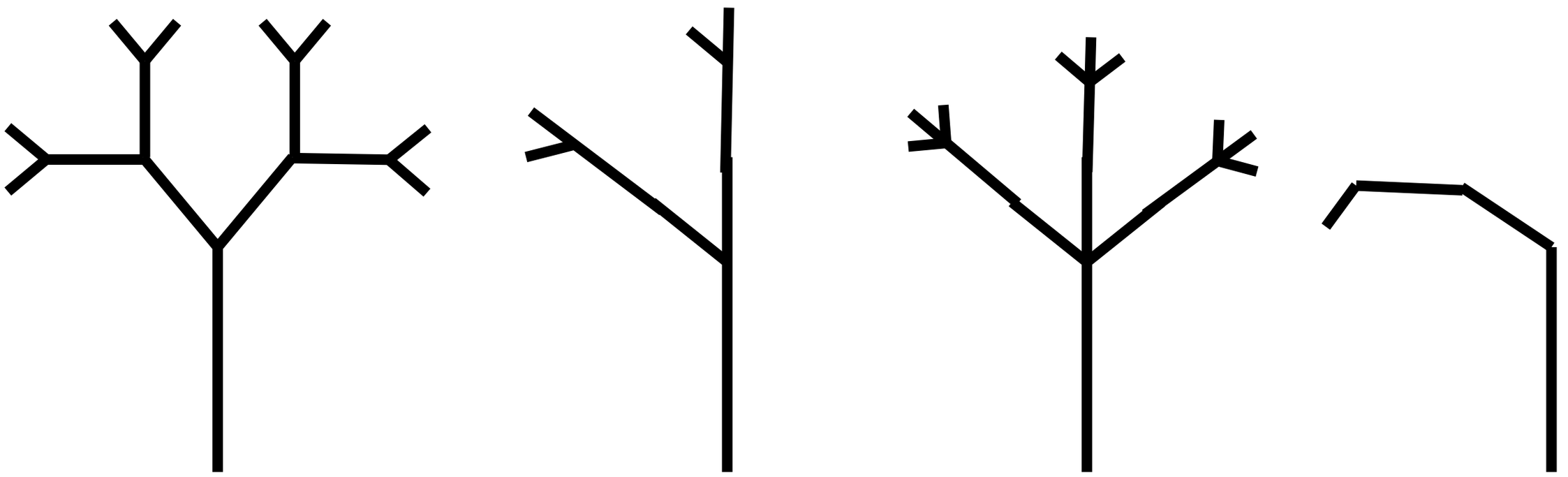
L-Systems and animation

- Growth process
  - Create branching
  - Grow stems
  - Grow flowers from buds
  - React to obstacles
- Effects of external events (wind …)

Branching Structures (2D)



Branching Structures (2D)



L-Systems

- Conceived to model plants by Lindenmayer
- Works with production rules of the form

$\alpha i \rightarrow \beta i$

αi is the predecessor (single symbol)

βi is the successor (many symbols)

- All symbols are processed in parallel
- Identity production if the predecessor isn't in the set of rules ($\alpha i \rightarrow \alpha i$)
- Stops when no symbols are predecessors

L-Systems

Example

- S -> ABA
- A -> XX
- B -> TT
- S is the axiom (the starting point)
- XXTTXX is the string obtained

Geometrical Interpretation

- Each symbol of the string is replaced by a geometric element
— Geometric replacement
(X = straight line, T = V-segment)
— Turtle graphics
(X = move d units, T = turn by angle Ω)

L-Systems

Other refinements

- Bracketed L-systems (to create sub-strings)
- Non-deterministic L-systems (n times the same preprocessor then chosen randomly)
- Stochastic L-systems (using several times the same preprocessor with different probabilities summing to 1)
- With context (to be chosen a preprocessor must have some precise symbols before or after)

Animated L-Systems

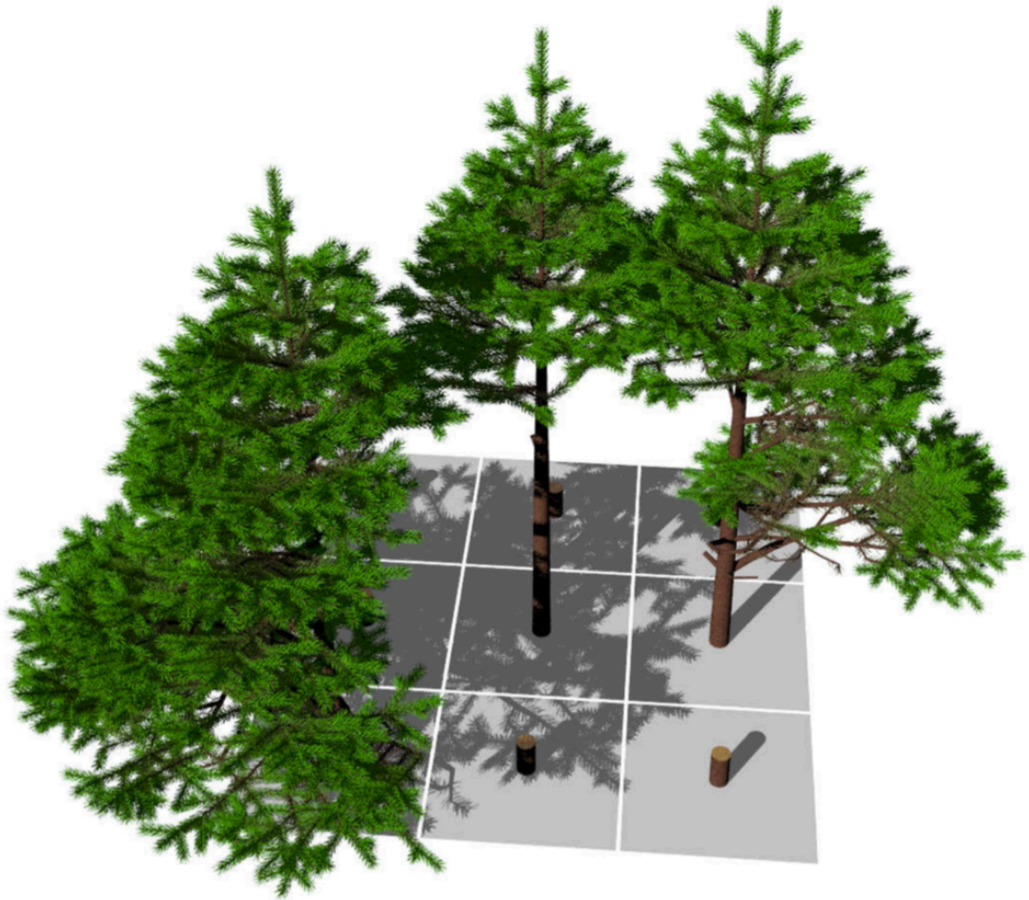
- For growth animation, the length need to be increased over time
  - Parametric L-systems (uses a parameter t that varies from 0 to 1. The symbol is changed only when reaching the 1)
- The shape might evolve too
  - Timed L-systems (using a local age value that changes the shape of the geometry -- flower vs bud)

Animated L-Systems

- To interact with the environment, variables can be added such as the amount of sunlight, length of day, gravity, obstacles …
- In Turtle, the chain of symbols is scanned left to right. The variables are then transported along the chain.

L-Systems

Some Examples



Animation of Plants

- Many different techniques
  - Textures
  - Skinning
  - Geometrically based
  - Physically based
  - …
- Different types of plants
  - Grass, trees, anemones …

Animation of Grass

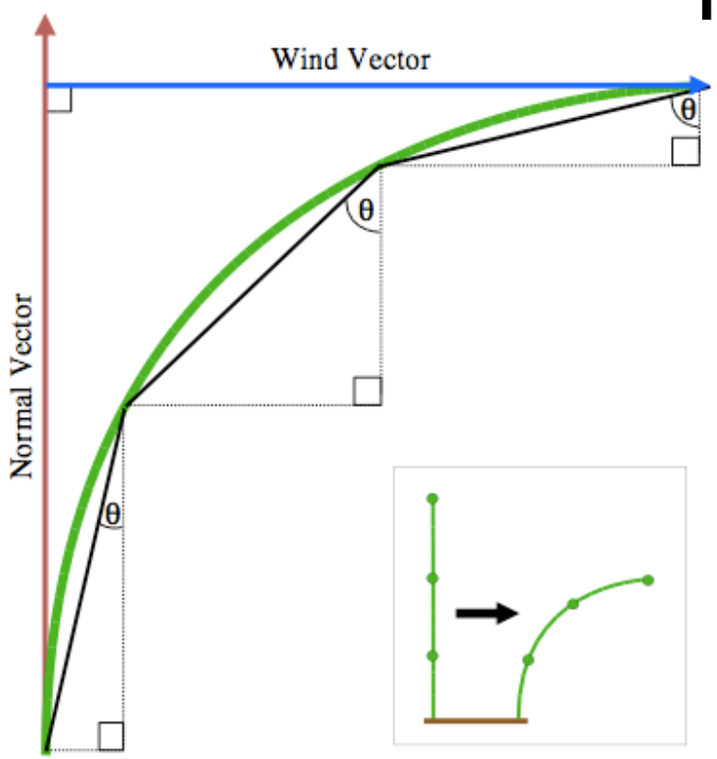
- In addition to growth, plants are animated when they interact with the wind, walking characters …
- Problem of grass: many blades are necessary
- Combination of different animation algorithms
  - Animation of volumetric textures
  - Animation of the geometry

Animation of Grass

Using a length preserving curve

Bakay et al., Real Time Animated Grass, EUROGRAPHICS 2002

- Each blade of grass is made of N segments
- The wind is horizontal

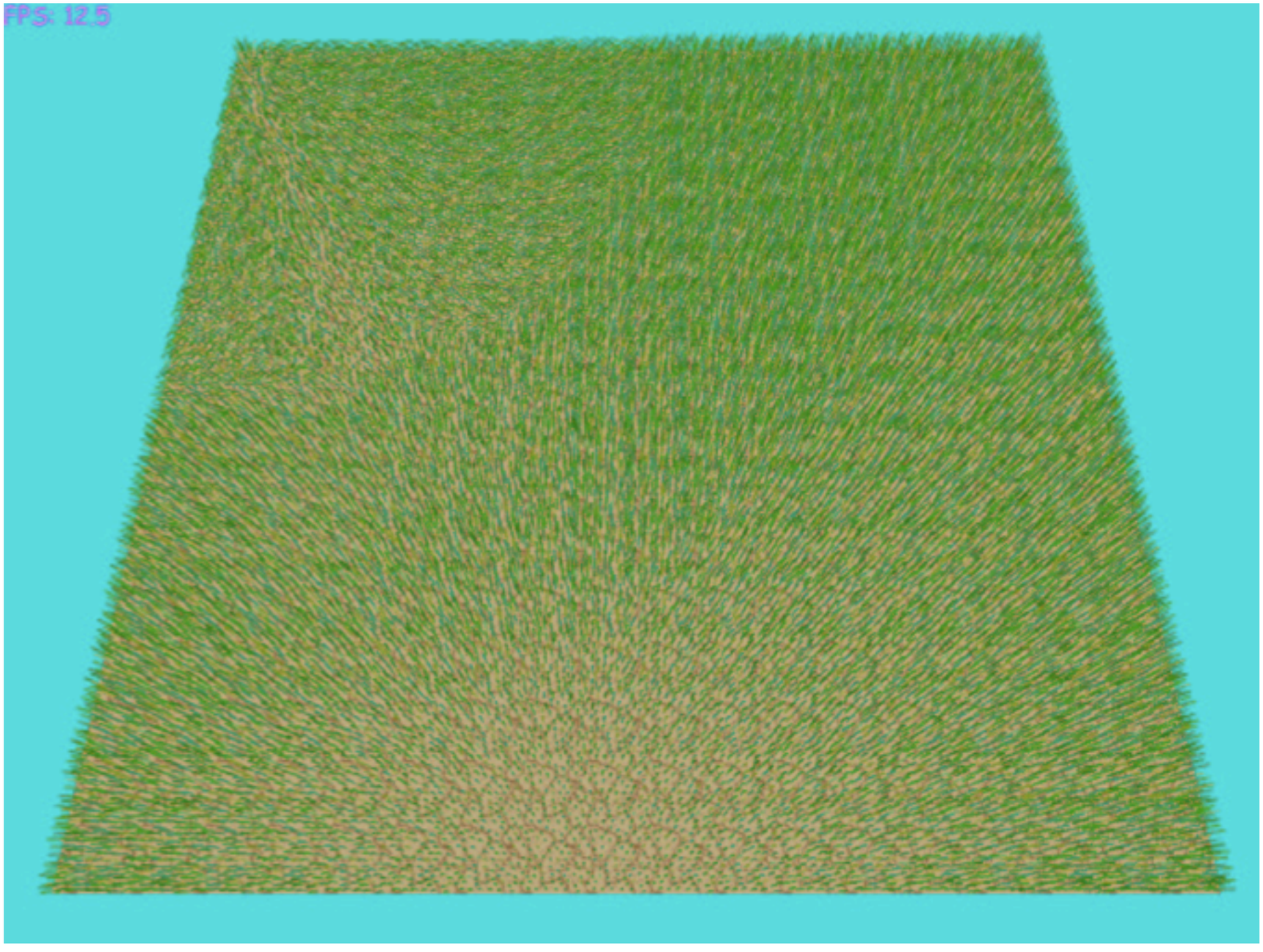


Normal Displacement $\sum_{i=0}^{N} \cos\left(\frac{i \cdot I \cdot \pi \cdot S}{2 \cdot N}\right)$

Wind Displacement $\sum_{i=0}^{N} \sin\left(\frac{i \cdot I \cdot \pi \cdot S}{2 \cdot N}\right)$

where i is the current shell, I is the current wind intensity at this vertex, S is the inter-shell distance and N is the total number of shells being rendered.

Results

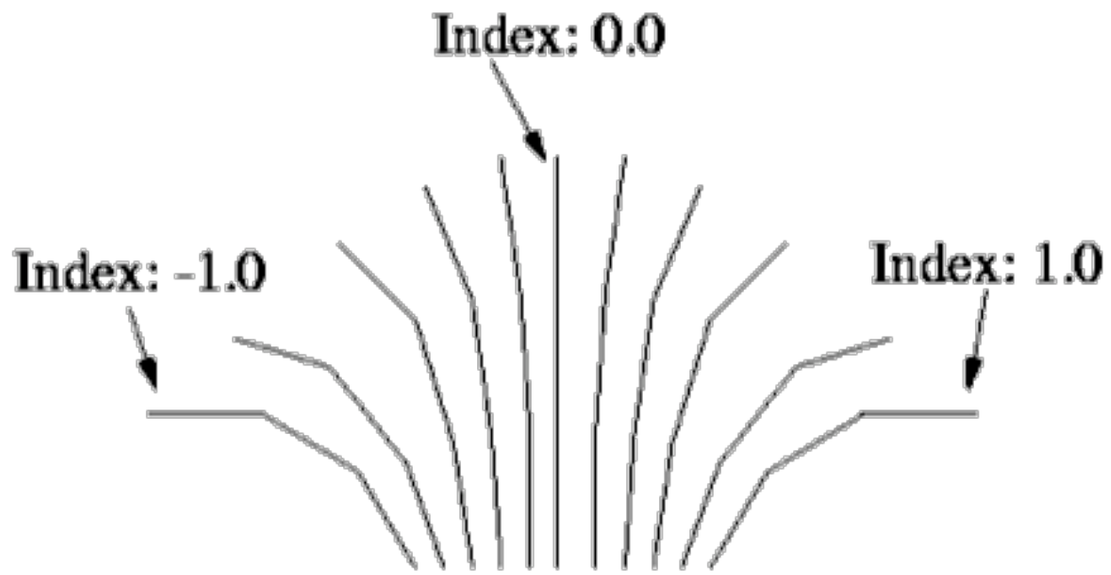


Using pre-computed bending

Perbet et al., Animating Prairies in Real-Time, 2001

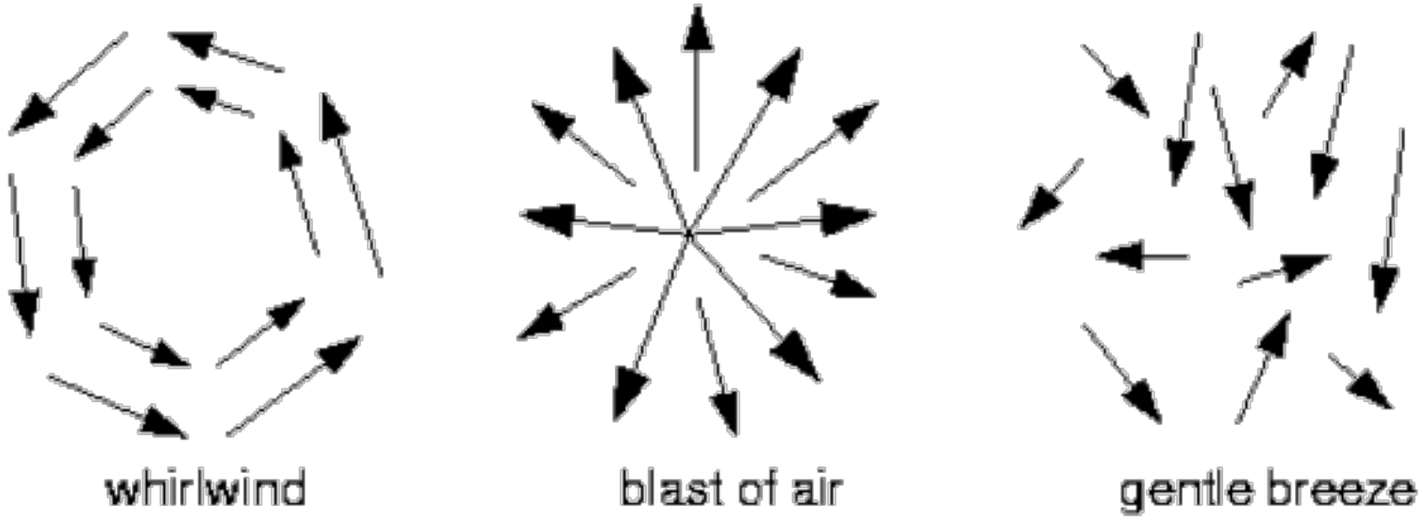
- Bending precomputed by taking into account different grass parameters
  - Size, stiffness …

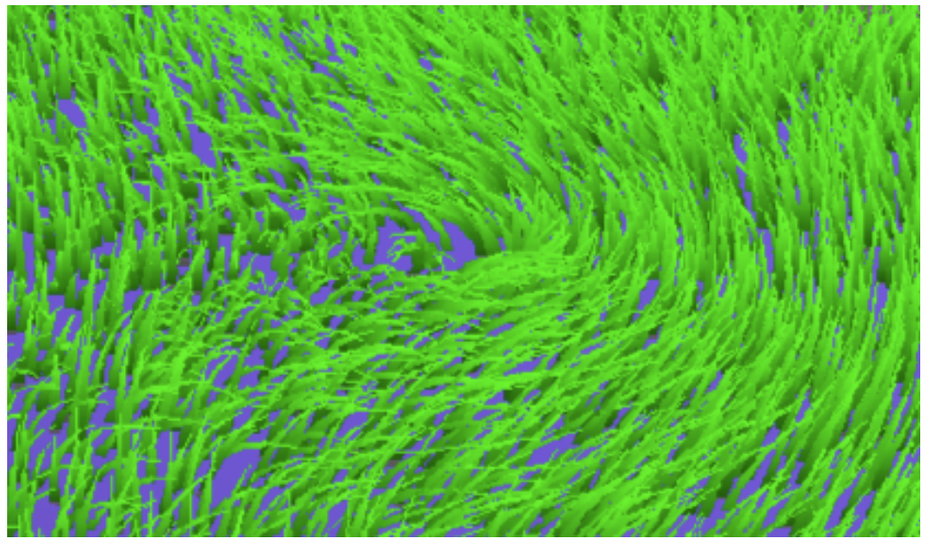
- The bending index and direction is chosen procedurally



Using precomputed bending

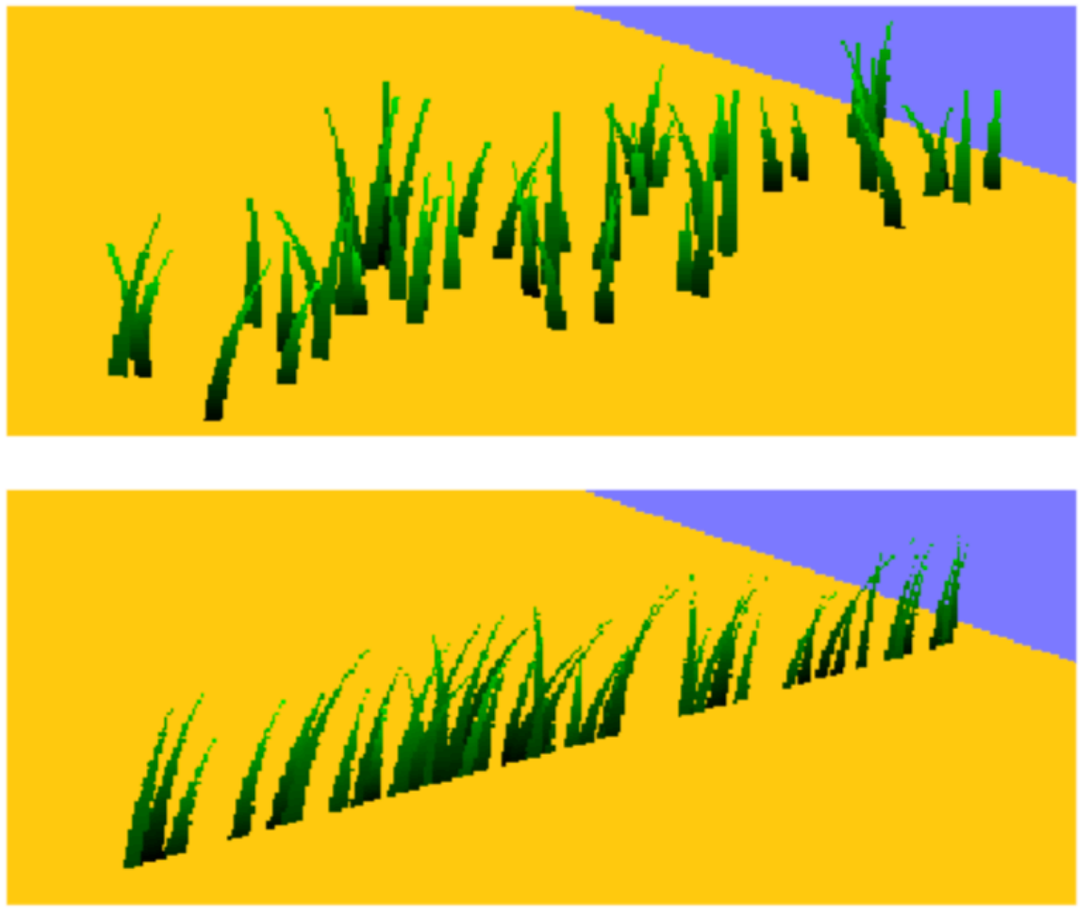
- The bending index and direction is chosen procedurally
  - According to the wind direction and intensity
  - According to other objects (characters, stone) through a field





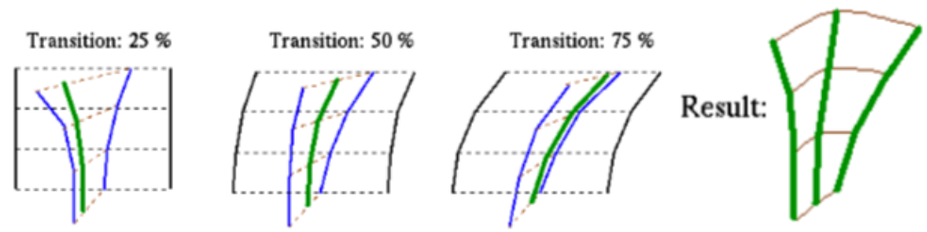
Level of Details

- Front: 3D (model)
- Middle: 2.5 D
- Back: 2D (texture)
- Transition from one level to the next according to distance from the camera
- Morphing between 2.5D and 3D



Level of Details

- Front: 3D (model)
- Middle: 2.5 D
- Back: 2D (texture)



(a) (b)

Figure 15: (a): Computing a smooth transition between the 3D and the 2.5D representations is done through a progressive linear interpolation, during motion, between a 3D blade of grass and its target blade in the 2.5D texture. (b): The resulting motion of the displayed blade is non-planar

- Transition from one level to the next according to distance from the camera
- Morphing between 2.5D and 3D

Results

Perbet et al., Animating Prairies in Real-Time, 2001

Results (Interaction)

Perbet et al., Animating Prairies in Real-Time, 2001

Animating trees with skinning

Singh et al., Tree Animation for a 3D Interactive Visualization System for Hurricane Impacts, 2005

- Bones in the trunk and branches
- Rotation of bones depends on wind-bone angle (use of heuristics)

Animating hedge with mass-springs

- Used in the movie “Over The Hedge”
- Hedge created by an L-system
- Nodes are connected with springs
  - Can elongate
  - Can break if forces too strong
  - Forces can be tweaked by hand

Over The Hedge

Sea Anemones

- Sea Anemone tentacles or seaweeds are similar to grass

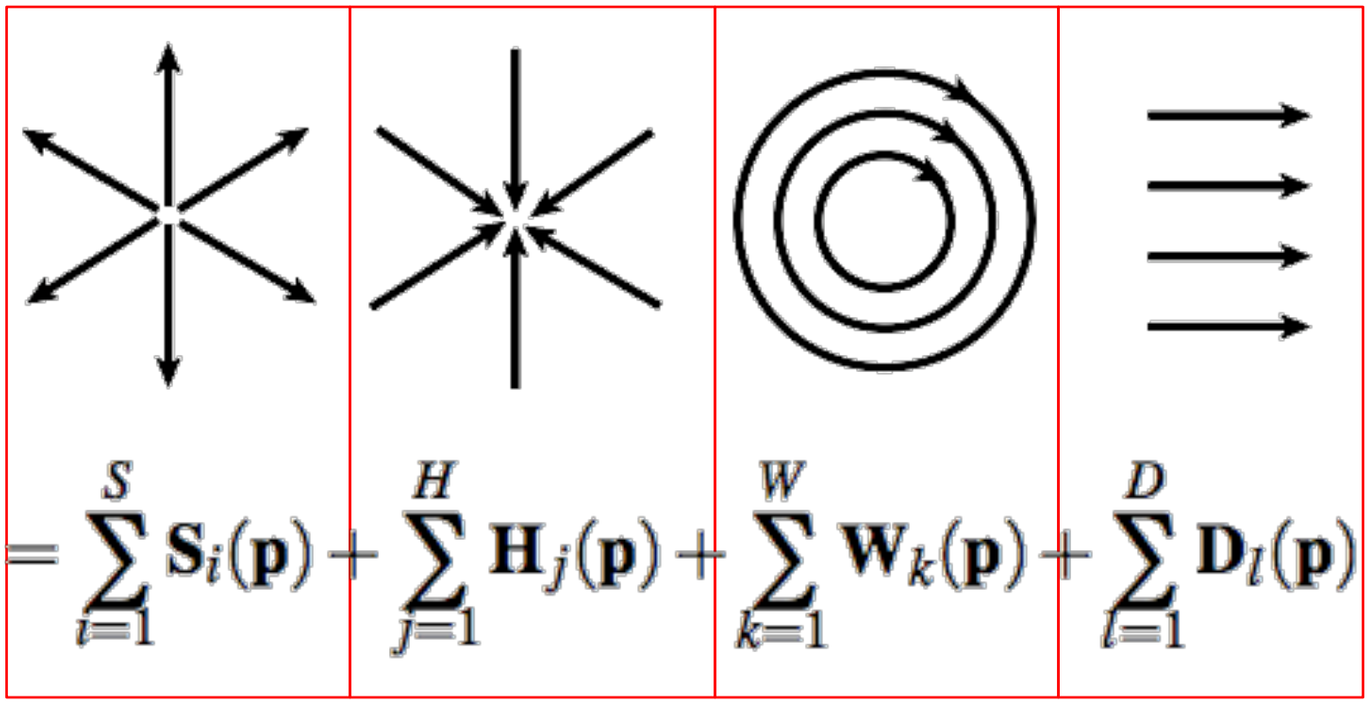
  [Aliaga & Larboulette 09]

- Fluid: air replaced by water
- Continuous description of the fluid using singularities
- Tentacle represented by a chain of nodes

Fluid Description

- 3D vector field composed of 4 types of singularities

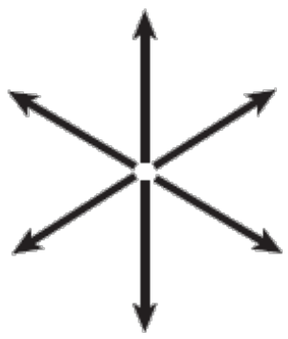
Source Hole Vortex Directional



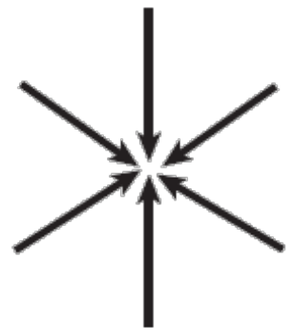
$$\mathbf{V}(\mathbf{p}) = \sum_{i=1}^{S} \mathbf{S}_i(\mathbf{p}) + \sum_{j=1}^{H} \mathbf{H}_j(\mathbf{p}) + \sum_{k=1}^{W} \mathbf{W}_k(\mathbf{p}) + \sum_{l=1}^{D} \mathbf{D}_l(\mathbf{p})$$

Singularities

- Source and Sink (Hole)
  - Intensity depends on the distance (local -- ϕ_{max} limits influence)
  - Opposite
  - To model water and fish



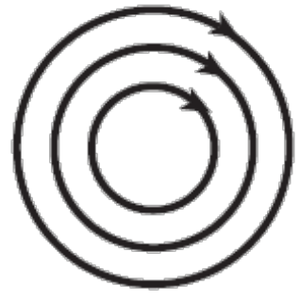
$$\mathbf{S}(\mathbf{p}) = \frac{\Phi}{\frac{\Phi}{\Phi_{max}} + d(\mathbf{p},\mathbf{C})^2} \frac{\vec{\mathbf{Cp}}}{||\vec{\mathbf{Cp}}||}$$



$$\mathbf{H}(\mathbf{p}) = -\mathbf{S}(\mathbf{p}) = -\frac{\Phi}{\frac{\Phi}{\Phi_{max}} + d(\mathbf{p},\mathbf{C})^2} \frac{\vec{\mathbf{Cp}}}{||\vec{\mathbf{Cp}}||}$$

Singularities

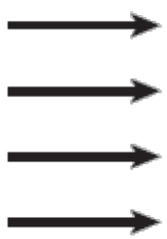
- Whirlwind (Vortex)
  — Local + rotation



$$\mathbf{W}(\mathbf{p}) = \pm \frac{\Phi}{\frac{\Phi}{\Phi_{max}} + d(\mathbf{p},\mathbf{C})^2} \frac{\vec{\mathbf{C}}\mathbf{p} \times \vec{R}}{||\vec{\mathbf{C}}\mathbf{p} \times \vec{R}||}$$

- Directional Field
  — Global
  — Intensity can vary over time: sine or cosine function to obtain waves

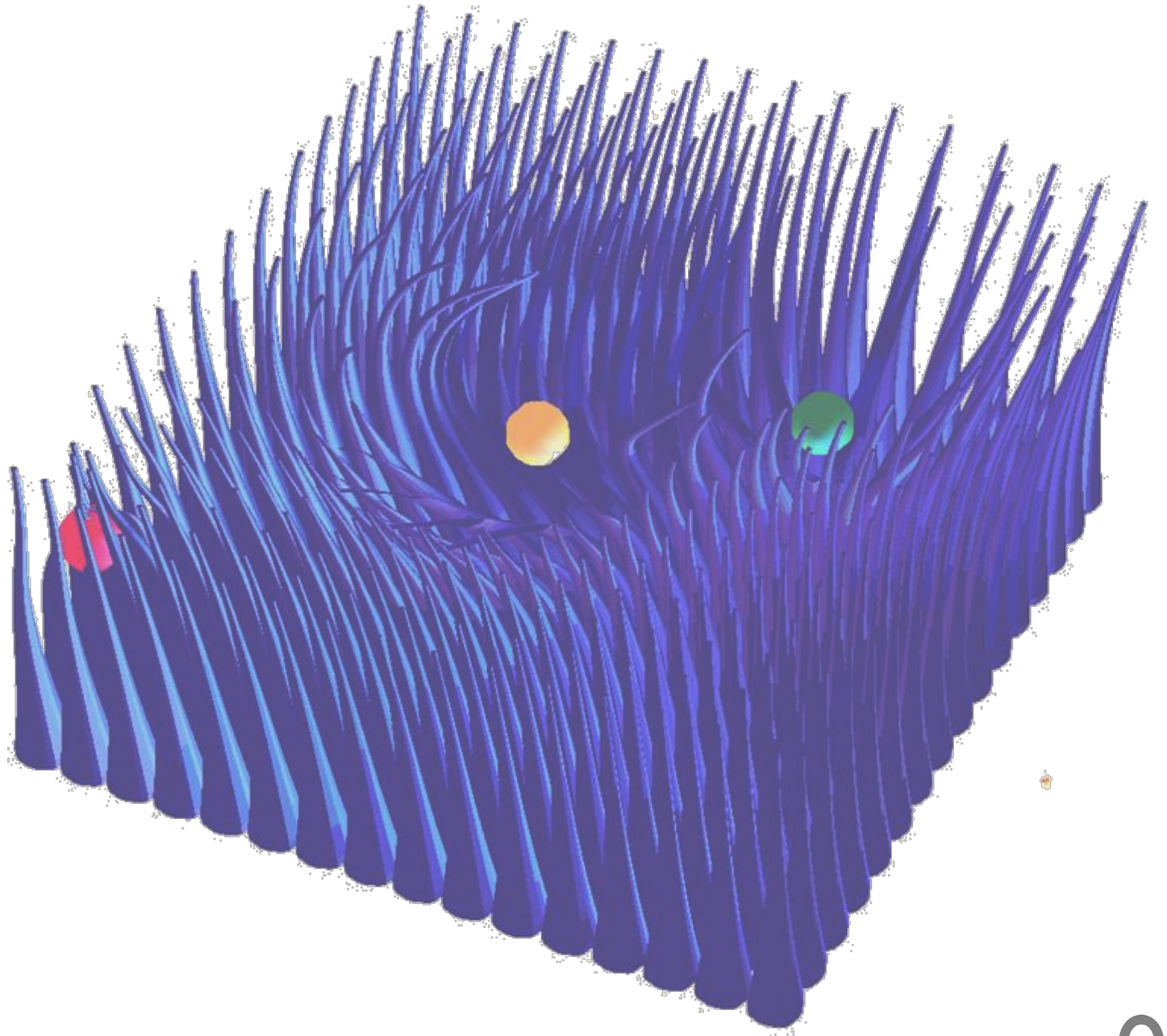
- To model currents



$$\mathbf{D}(\mathbf{p}) = \Phi(\mathbf{p},t).\vec{v}$$

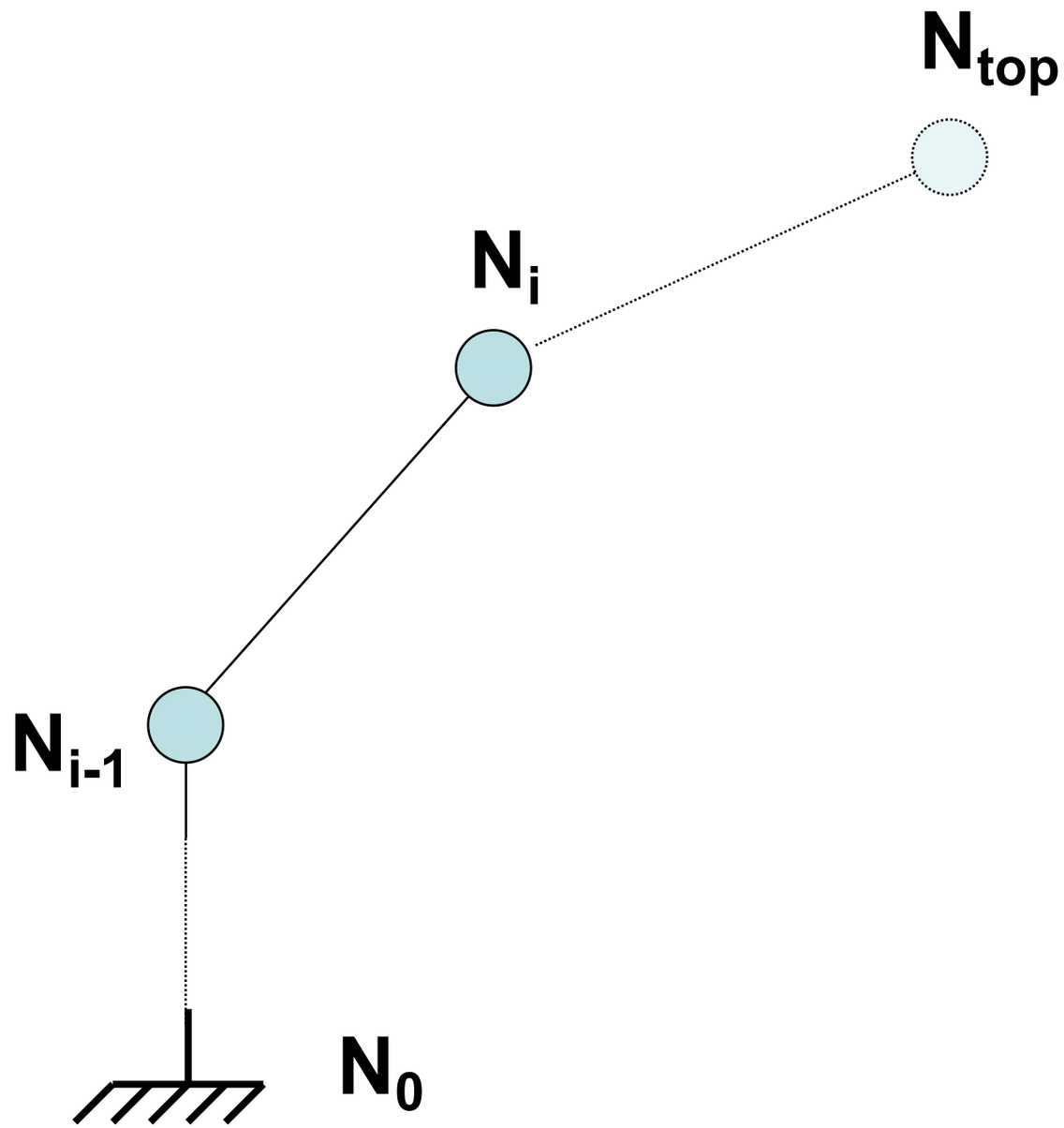
Singularities Effect

- Green: Source
- Red: Sink
- Yellow: Vortex



Node Chain

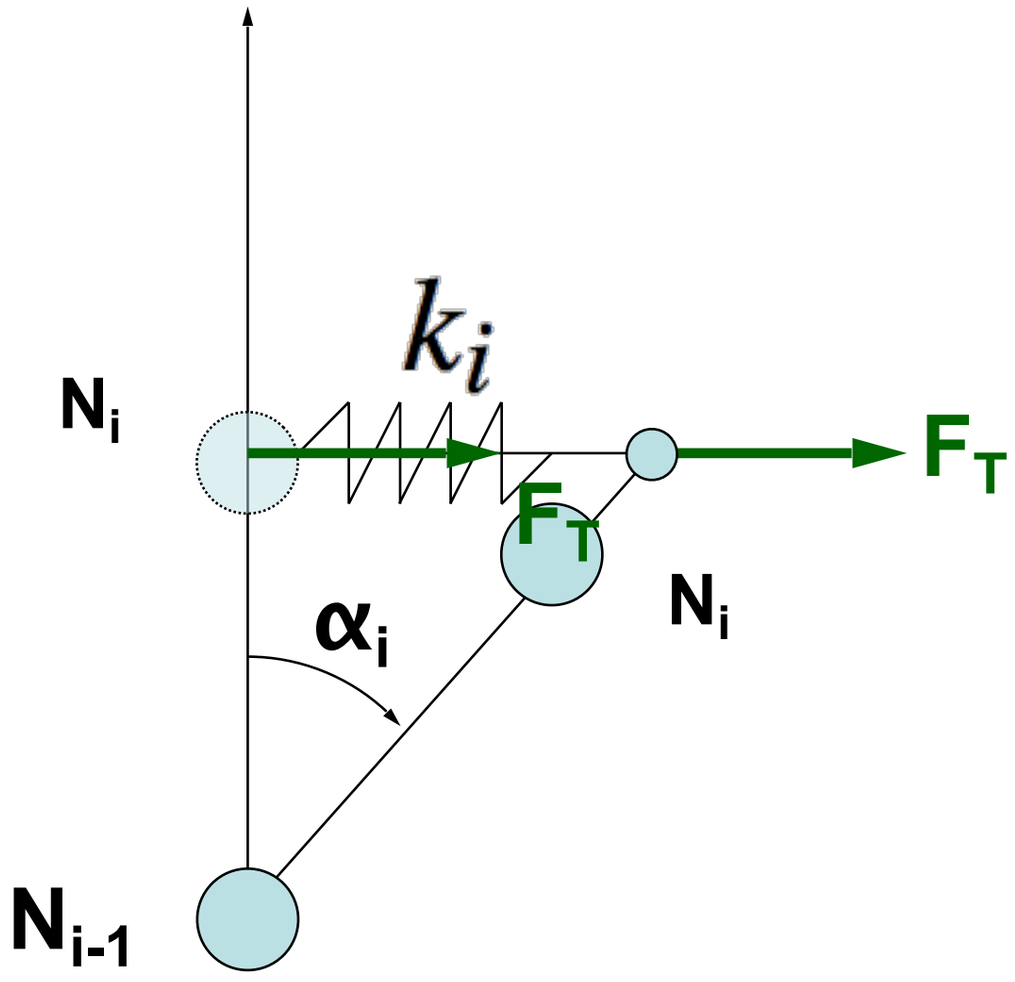
- N_0 to N_{top}
- N_0 is anchored to the foot



Deformation of the chain

- For each node
  1. Get net force from singularities
  2. Compute node displacement
  3. Transmit moment down the chain

2. Node Displacement



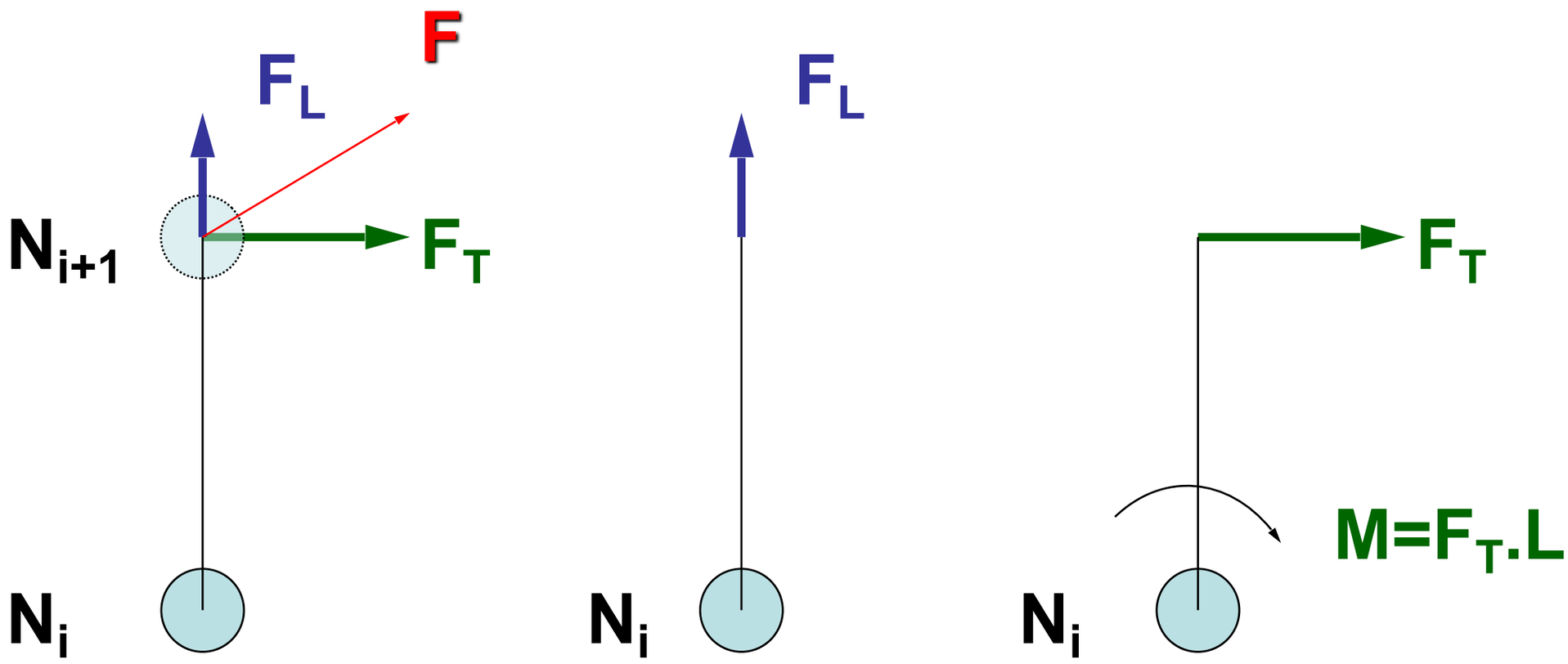
- Elastic force gives a bending angle
- k_i : stiffness at node Ni

$$k_i = r_i^n . K_{material}$$

- Moment generated:

$$\mathbf{F_M}(N_{i-1}) = \frac{M(N_i)}{L_{i-1}} \cdot \frac{\vec{N_{i-1}N_{i-2}} \times \mathbf{M}(N_i)}{||\vec{N_{i-1}N_{i-2}} \times \mathbf{M}(N_i)||}$$

3. Propagation Moments and Forces



$$\mathbf{F}(N_i) = A.\mathbf{V}(N_i) + \mathbf{F_L}(N_{i+1}) + \mathbf{F_M}(N_i)$$

Singularity Keyframing

Seascape video

Seegrasses videos