A Direct Texture Placement and Editing Interface

Yotam I. Gingold

Philip L. Davidson

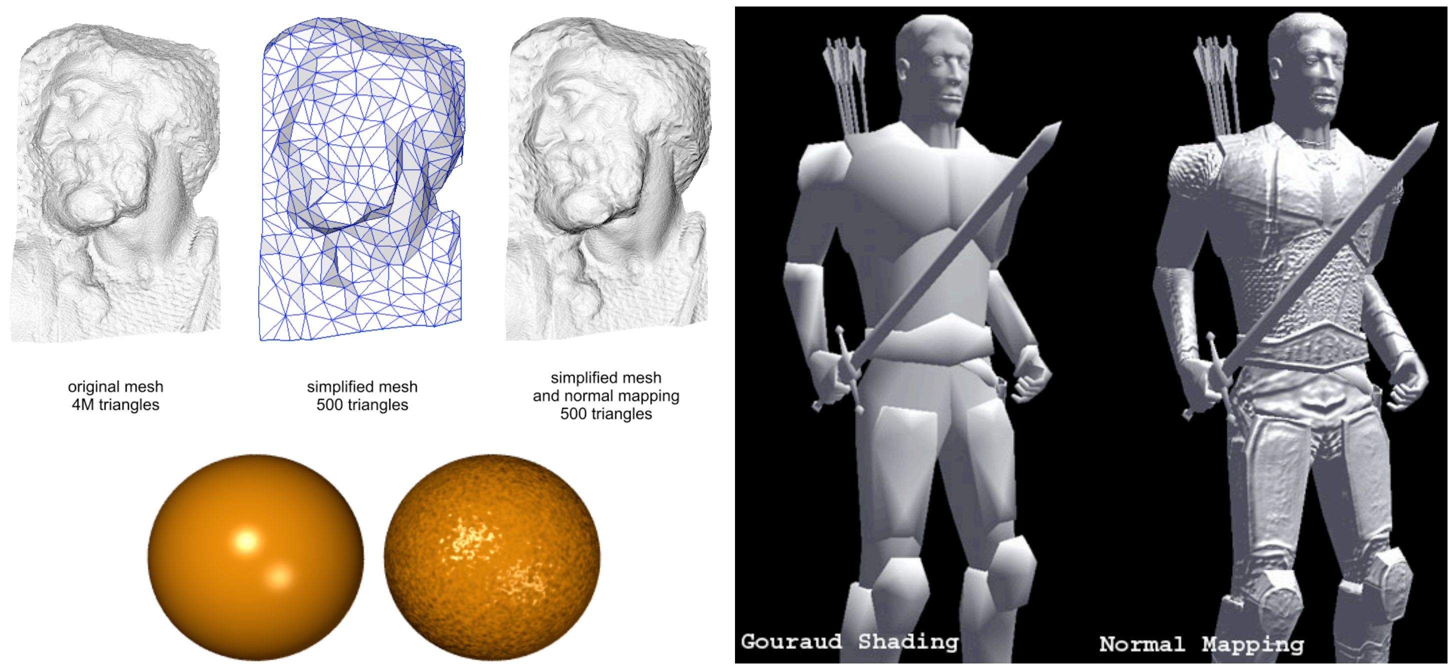
Jefferson Y. Han

Denis Zorin

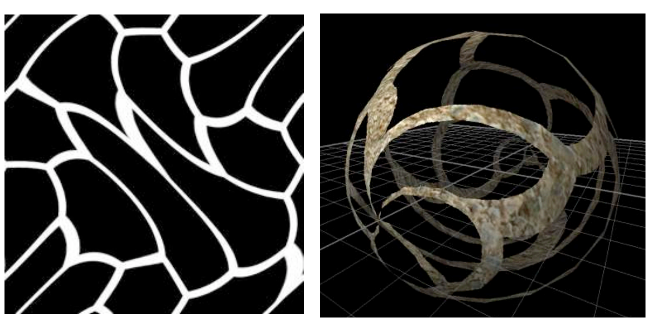
Courant Institute of Mathematical Sciences
New York University

Textures

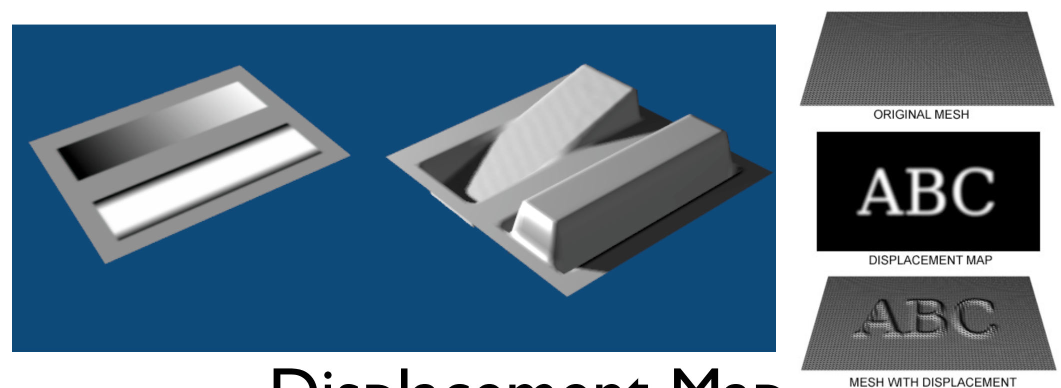
Color Map



Normal Map



Alpha Map



Displacement Map

Texture mapping is a general technique used throughout computer graphics.

Texture maps, in general, are planar domains (usually rectanges) mapped onto surfaces via some parameterization (mapping function). They are used to add information to the surface, such as color, alpha, and geometric detail.

[Q: Should I add a picture of parameterization: [0..1]x[0..1] -> mesh pts ?]

Overview

2 approaches to texturing

1 technical digression

7 operations

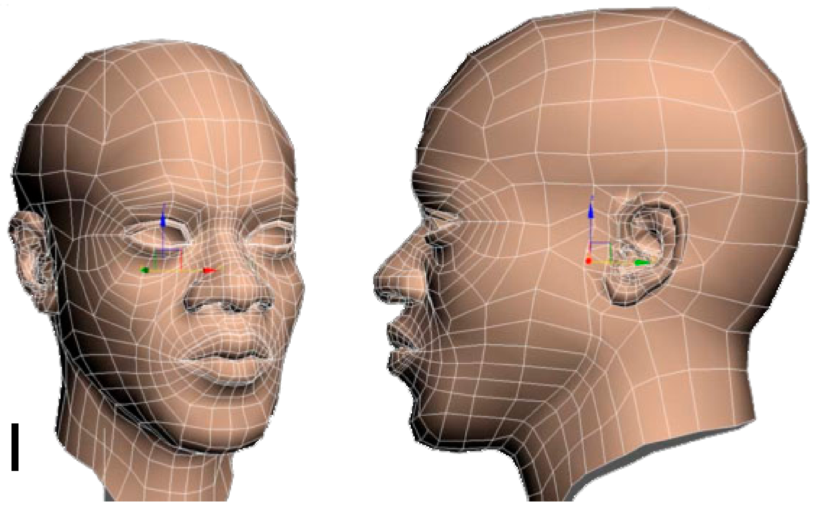
3 formulae

1 technical comparison

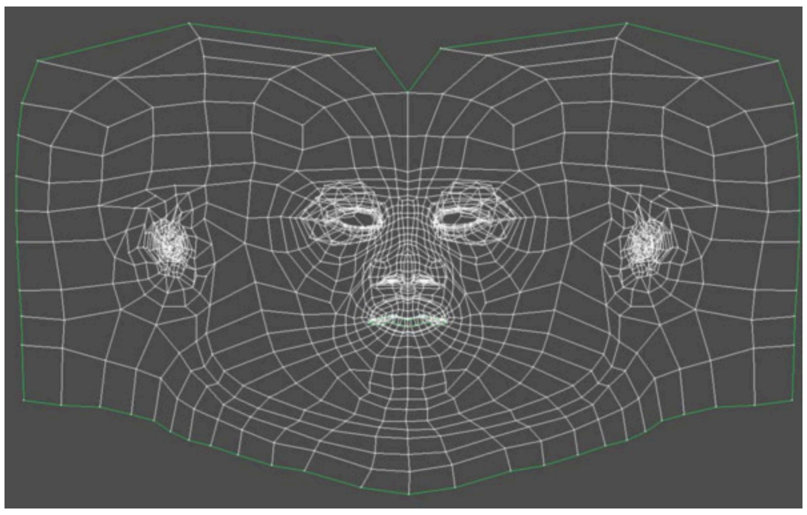
First Approach to Texturing

Jiri's Texturing Tutorial

[Jiri Adamec]



1



2

3

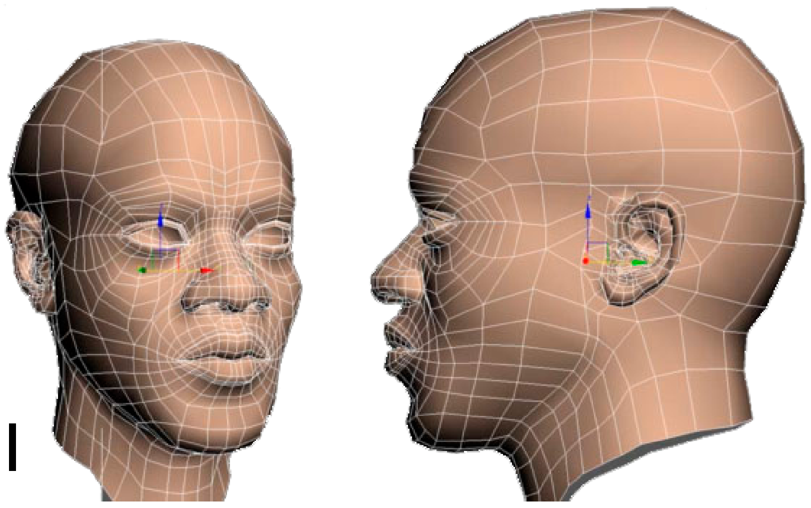
4

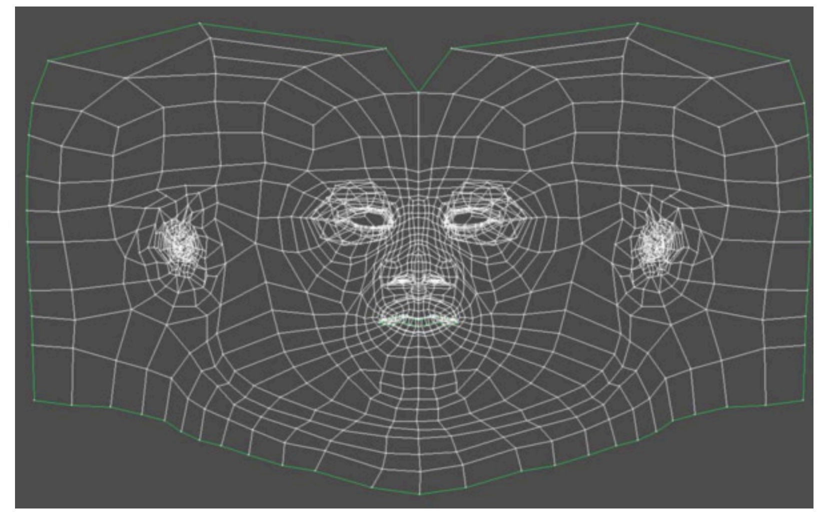
Top left is the 3d mesh, top right is the flattened mesh...

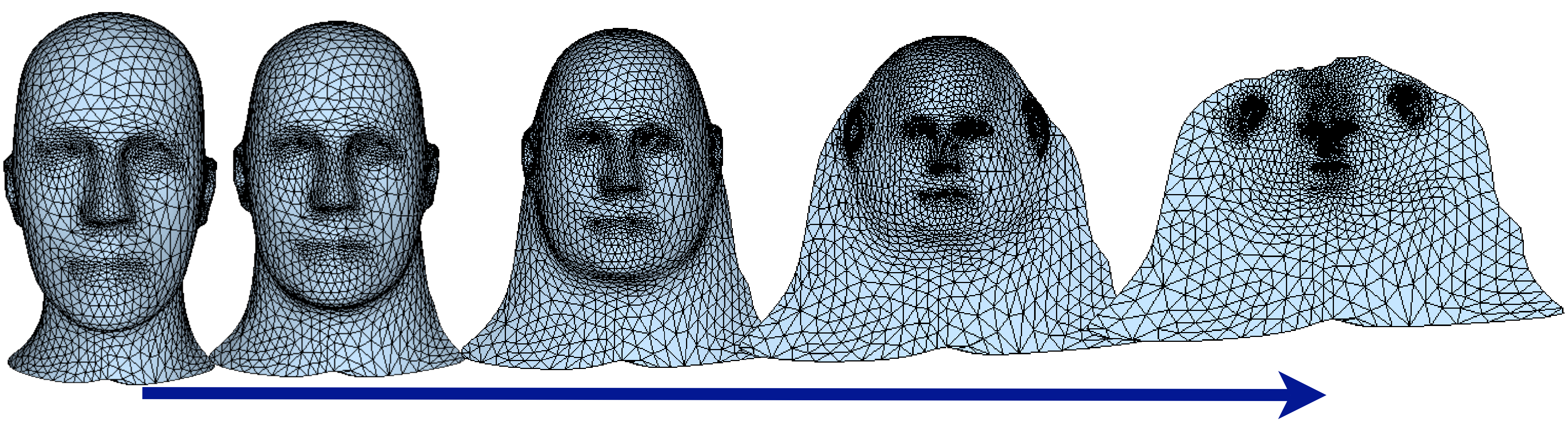
flattened?

Technical Digression

Flattening

1


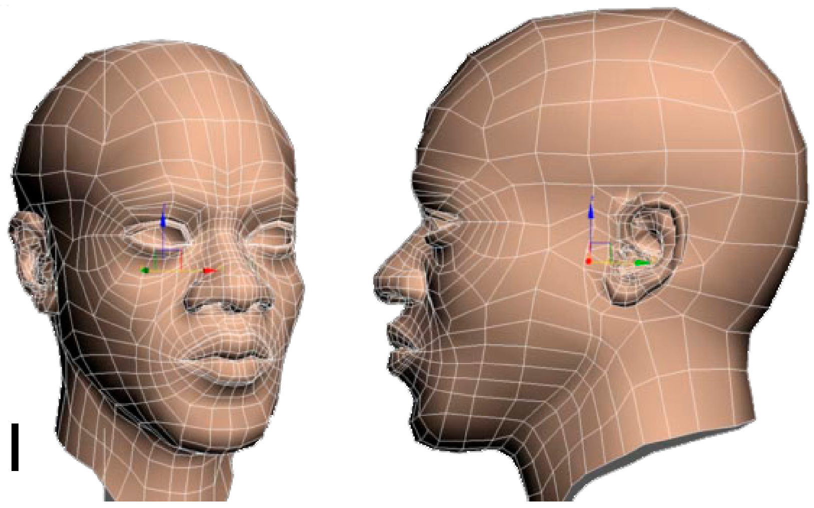
2




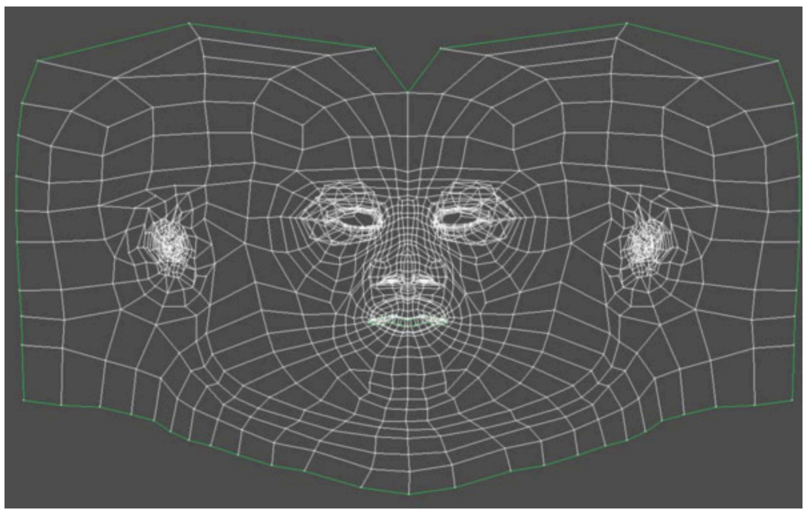
The 3d triangle mesh has to be flattened out into 2D triangles on a plane
- called a parameterization
- obviously distorts the mesh, but should make as much sense as possible to the artist (who still has to invert it mentally)
    - preserve angles between triangles
    - preserve each triangle's shape

Jiri's Texturing Tutorial

[Jiri Adamec]



1



2

3

4

The texture must also be created, typically by painting or placing found materials into the planar domain.

- **Photoshop**

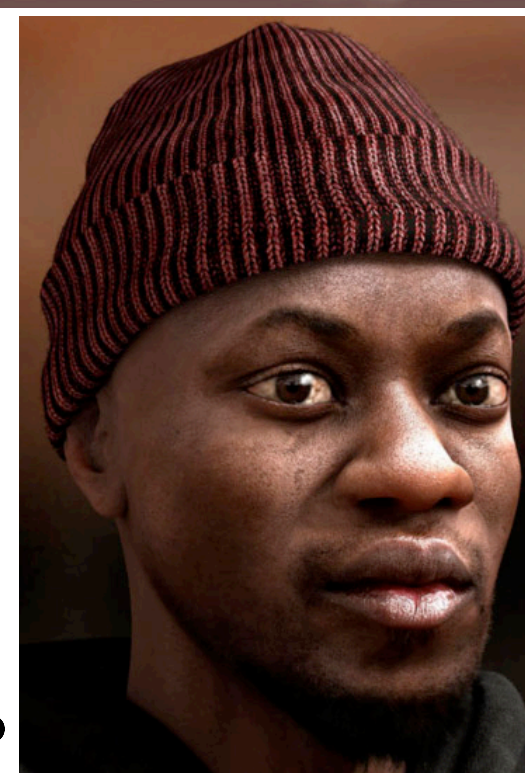
Jiri's Texturing Tutorial

[Jiri Adamec]

3

4

5



6

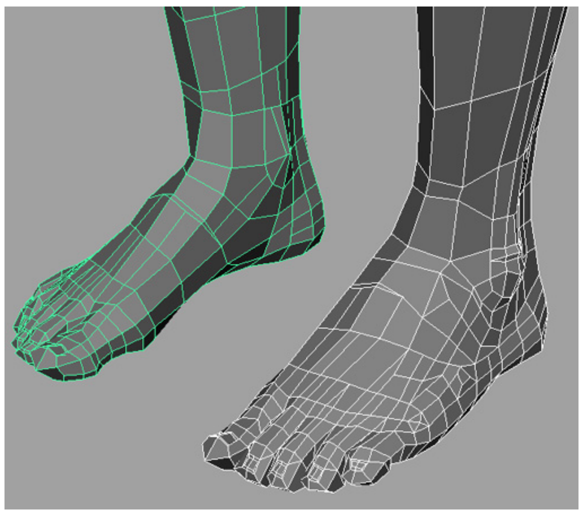
Artist has to invert this flattening (parameterization) in their mind.

Have to switch between 3 views (3D mesh, flattened mesh, photoshop).

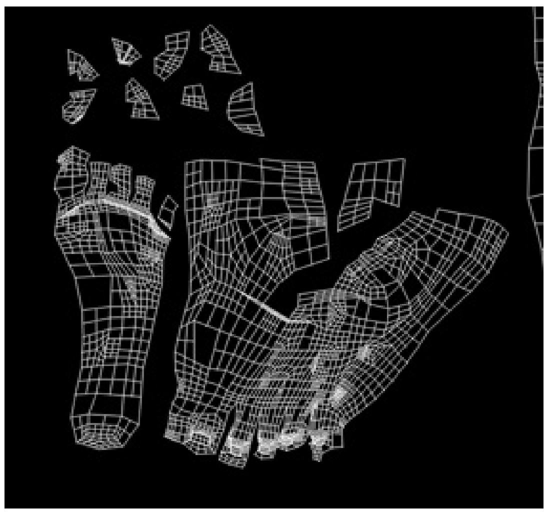
Iterative editing of the coordinate assignment and texture.

Feet Texturing Tutorial

[Steven Stahlberg]



1



2

3

4

Creating a texture, the planar domain, and its map onto the 3D geometry, the parameterization, is hard.

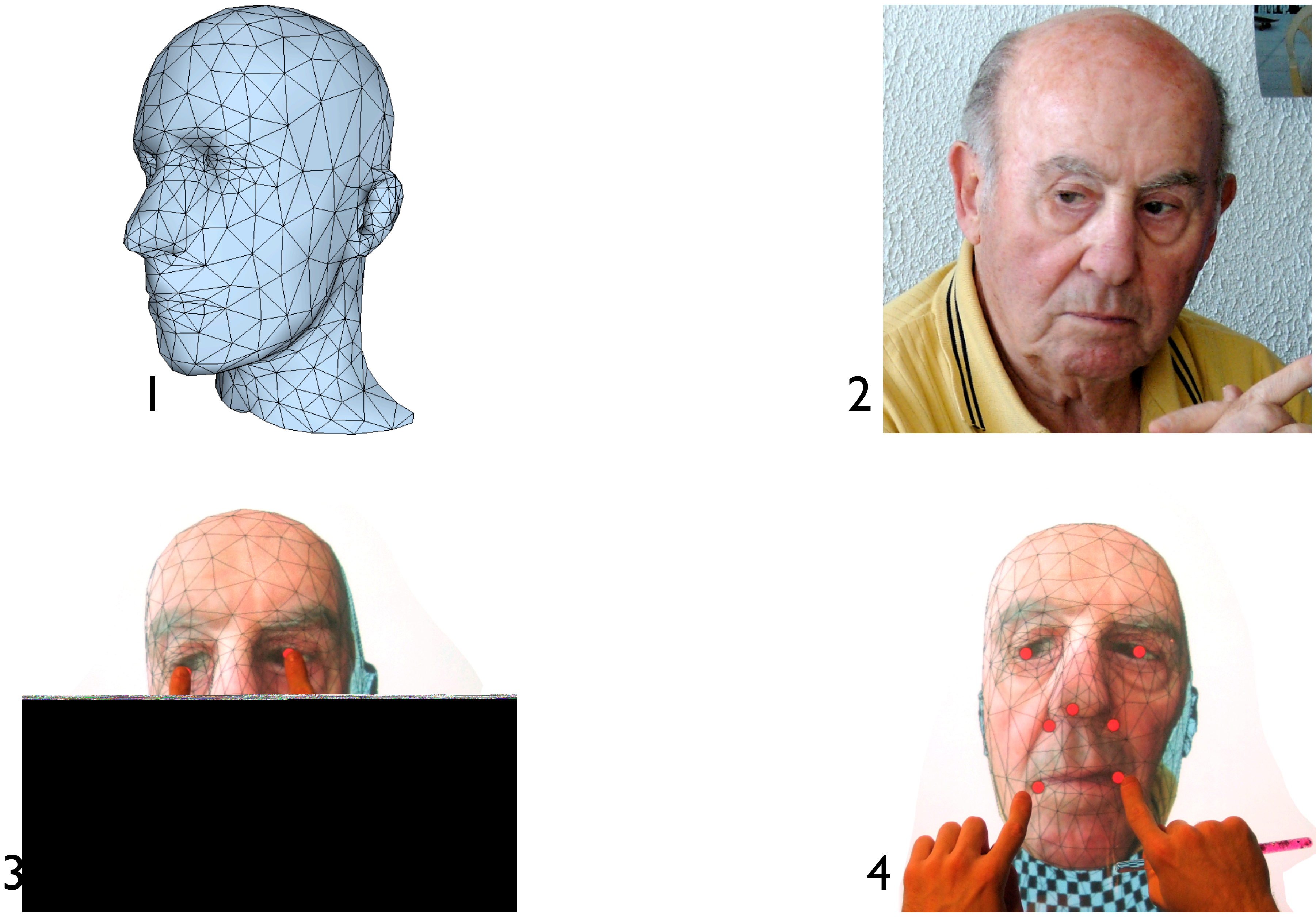
Experts can create nice results!

Interviewed artists about this process.

Most unpleasant parts of process (at the risk of sounding repetitive):

- **mentally inverting the flattening (parameterization)**
- **editing the texture in a 2d view**
- **mode switching to 3d to see the result**
- **tweaking the parameterization, also**
- **difficult to re-use textures**

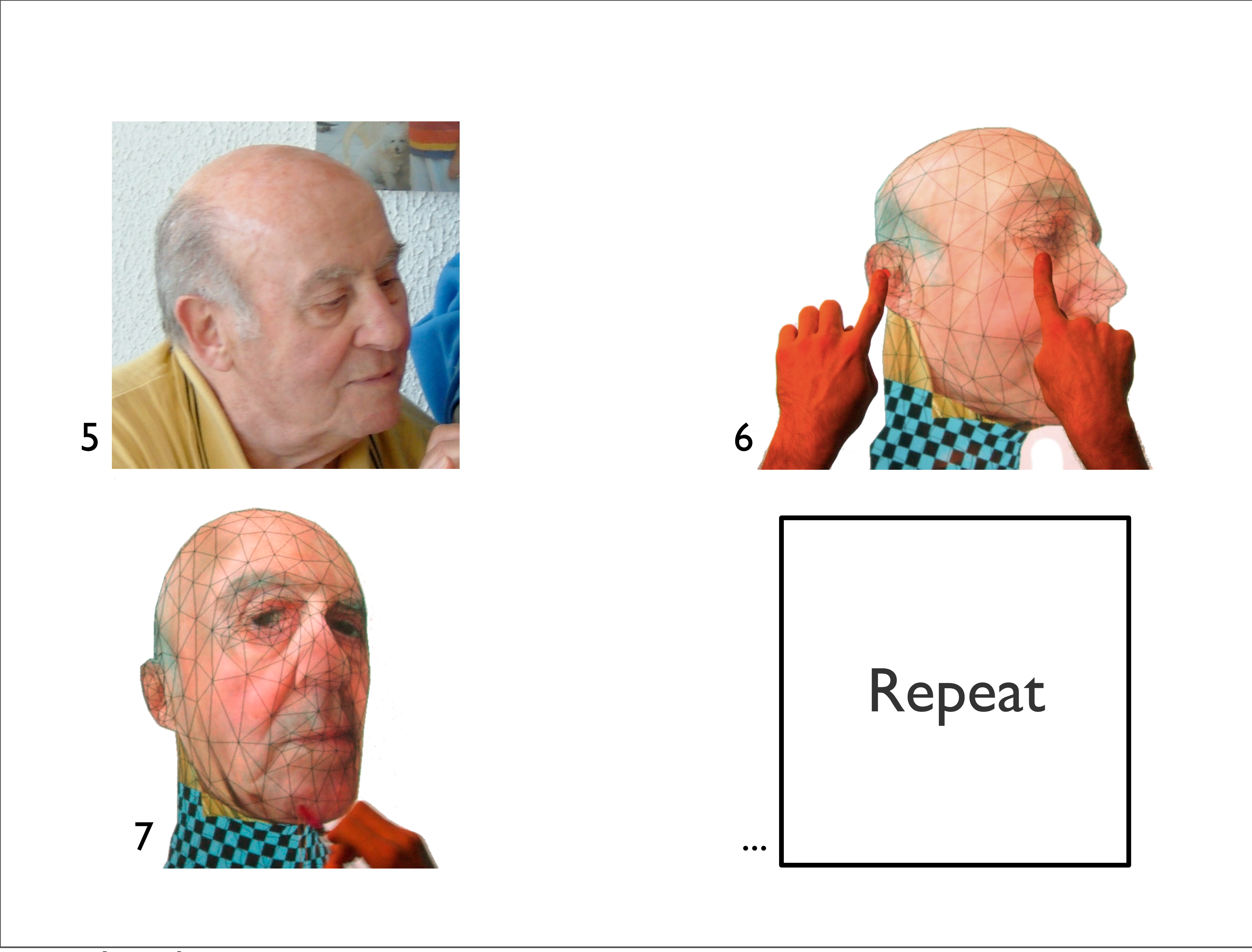
Our Approach to Texturing



Invert the process.
Perform texture placement & feature alignment directly on the model.

1 Load the mesh
2 Load picture
3 Place the picture (roughly: traslante, rotate, scale)
4 Align features in photo with mesh.

Texture is a rubber sheet deformed over the model.



5 Load another picture
6 Place and align it.
7 Blend between the two textures
8 Repeat until satisfied

- take found or existing texture (or draw naturally)
- spend entire time in 3D deforming it to fit geometry

Artists can use found textures or draw them naturally in 2D, and blend together multiple textures.

Related Work

2D Image Warping, etc

[Beier and Neely 1992]
[Igarashi et al. 2005]
[Schaefer et al. 2006]
[James and Pai 1999]

3D Texture Painting

[Hanrahan and Haeberli 1990]
[Agrawala et al. 1995]
[Igarashi and Cosgrove 2001]
[Igarashi and Hughes 2002]
[Carr and Hart 2004]
[Schmidt et al. 2006]

2-Handed Manipulation

[Guiard 1987]
[Hinckley et al. 1994]
[Zeleznik et al. 1997]
[Kurtenbach et al. 1997]
[Balakrishnan and Kurtenbach 1999]
[Balakrishnan and Hinckley 2000]
[Llamas et al. 2003]
[Wu and Balakrishnan 2003]

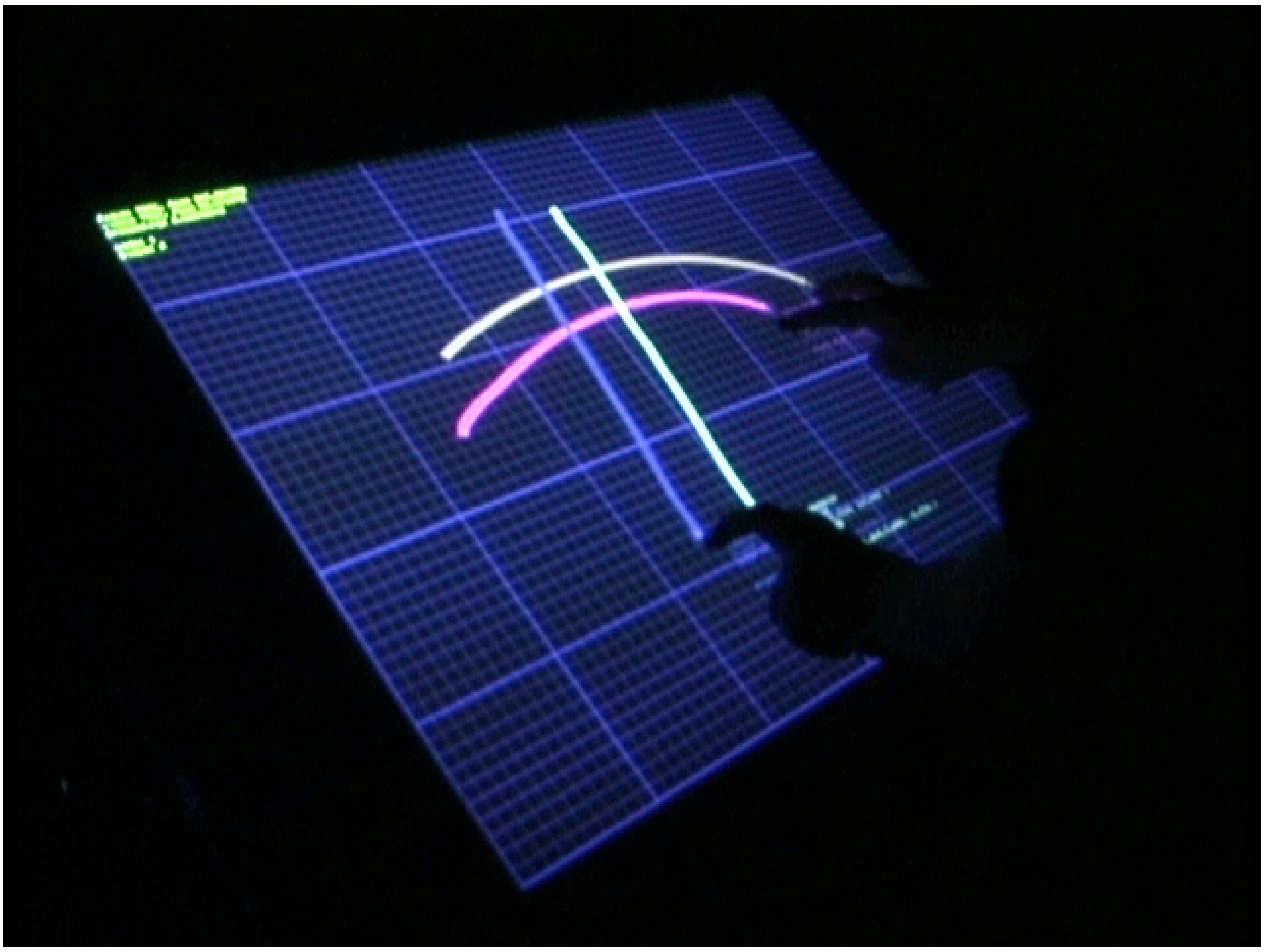
Related Work

Parameterization

[Beier and Neely 1992]
[Maillot et al. 1993]
[Floater 1997]
[Piponi and Borshukov 2000]
[Lévy 2001]
[Sander et al. 2001]
[Sheffer and de Sturler 2001]
[Lévy et al. 2002]
[DeBry et al. 2002]
[Desbrun et al. 2002]
[Kraevoy et al. 2003]
[Yoshizawa et al. 2004]
[Yoshizawa et al. 2005]
[Lee et al. 2005]
[Sheffer et al. 2005]
[Zayer et al. 2005]
[Yamauchi et al. 2005]

7 Operations

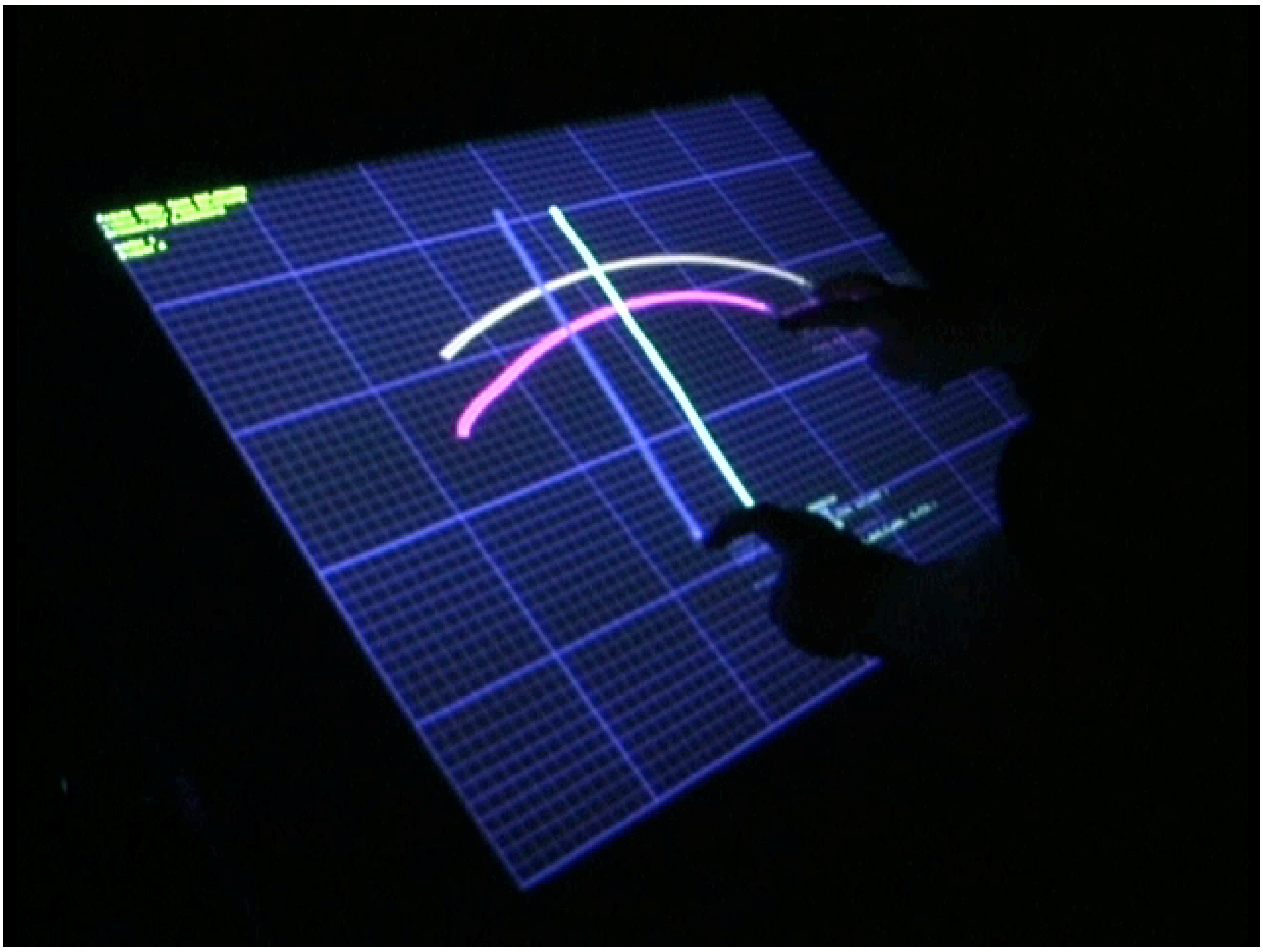
Multi-touch



We use the multi-touch system introduced by [Han 2005].
In the form of a 36” drafting table.
Pressure sensitive, so passive styluses or fingers -- whatever the artist is comfortable with.
Rear projected, so hands don’t occlude projection.

Precise, unambiguous discrimination between points of contact.
High res and high frequency updates.

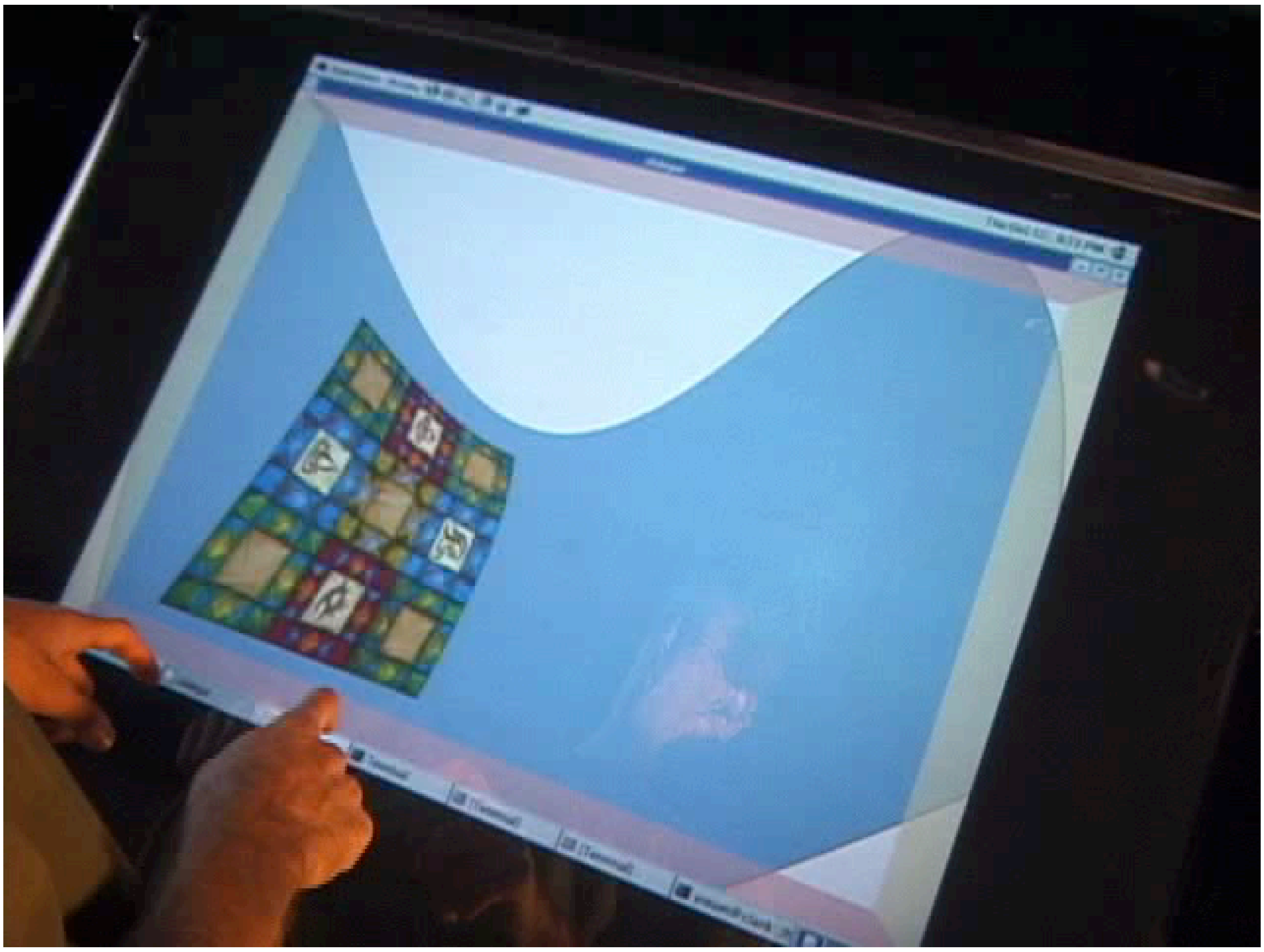
Multi-touch



We use the multi-touch system introduced by [Han 2005].
In the form of a 36" drafting table.
Pressure sensitive, so passive styluses or fingers -- whatever the artist is comfortable with.
Rear projected, so hands don't occlude projection.

Precise, unambiguous discrimination between points of contact.
High res and high frequency updates.

Texture Placement

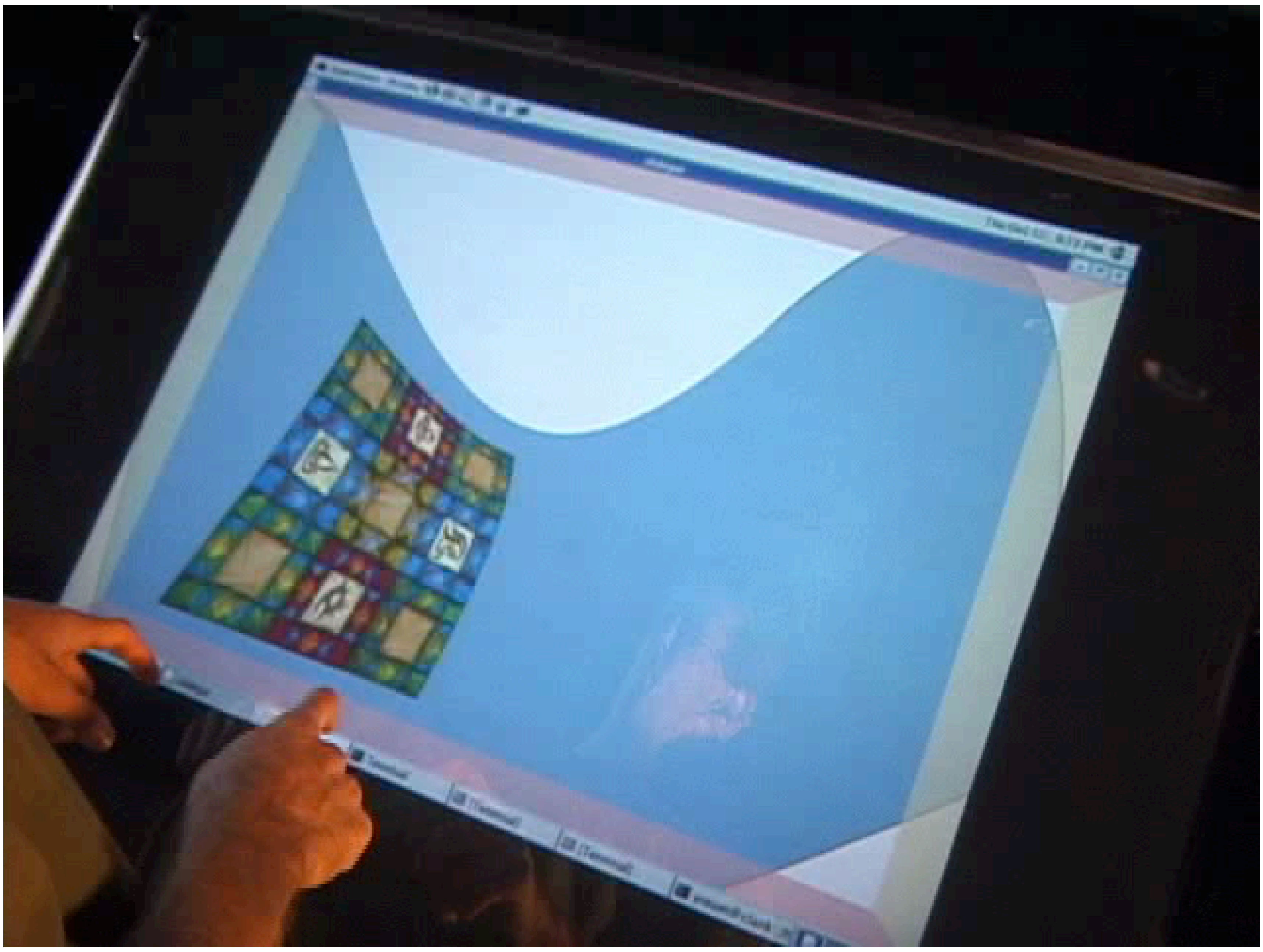


2 finger similarity transform (translate, rotate, uniform scale)
- used at least as early as [Kurtenbach et al. 1997]

Motivation:

Very fast and easy to roughly align features.

Texture Placement



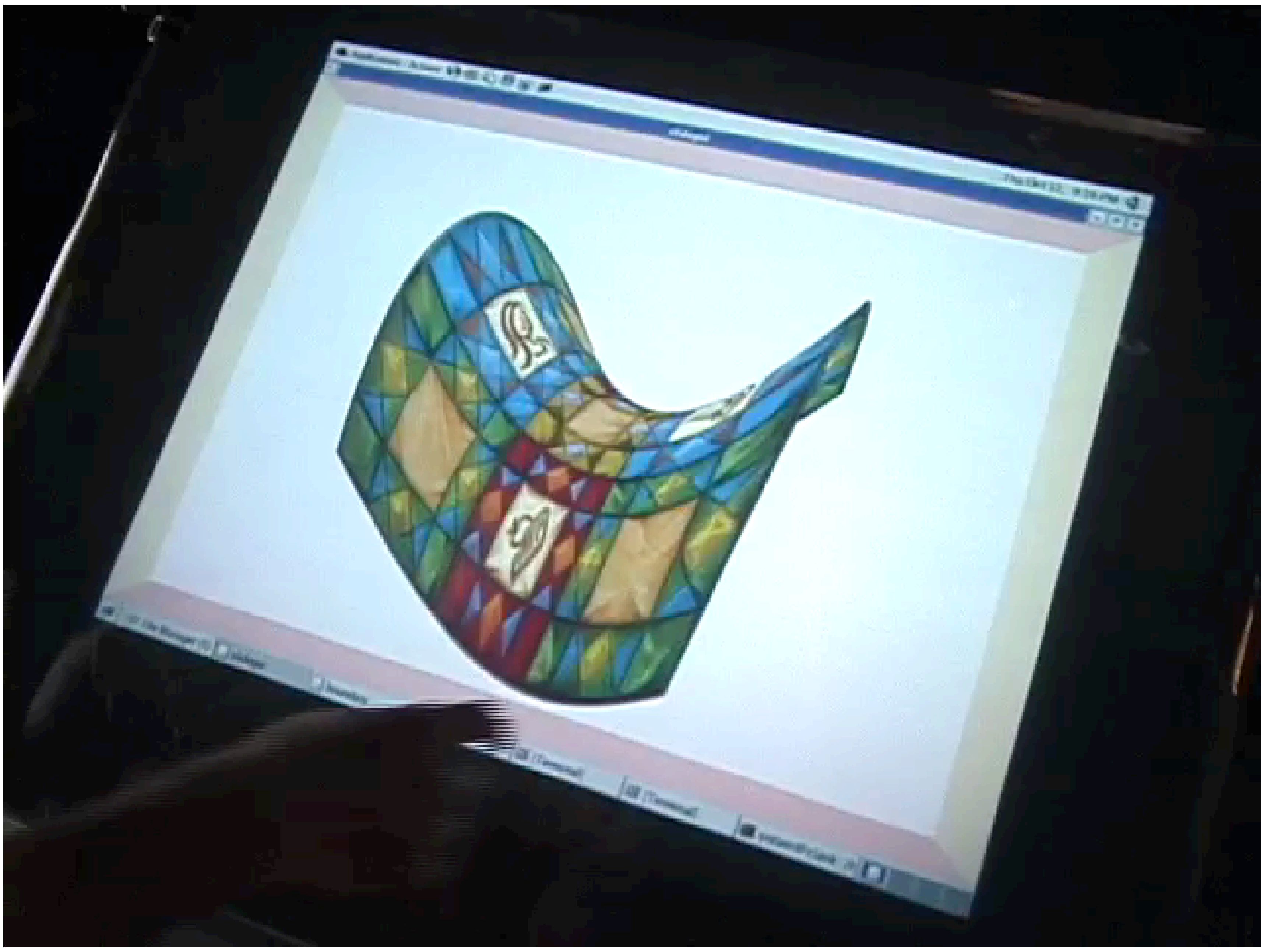
2 finger similarity transform (translate, rotate, uniform scale)

- used at least as early as [Kurtenbach et al. 1997]

Motivation:

Very fast and easy to roughly align features.

Feature Alignment

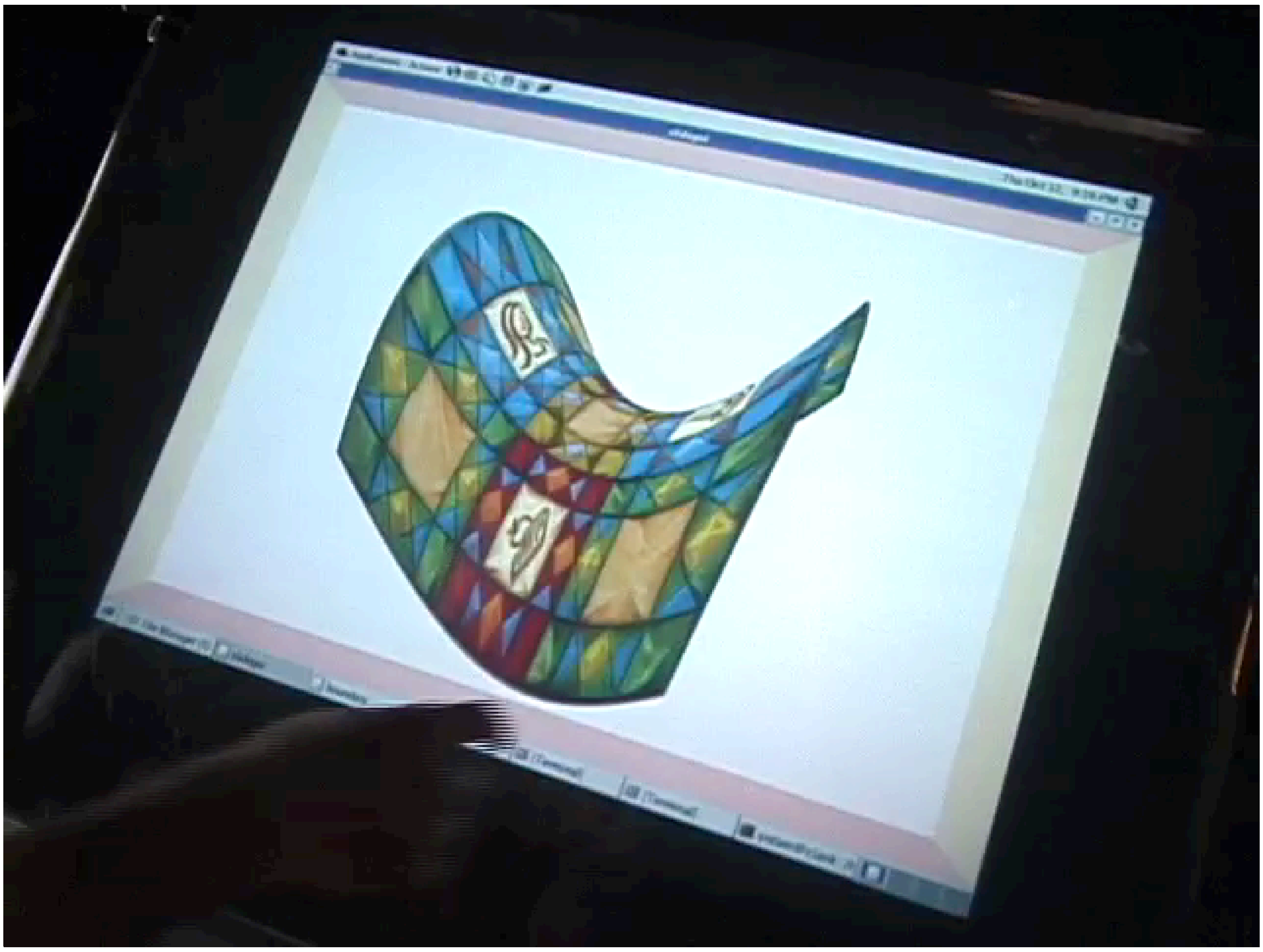


Point constraints
- assign points in texture to points on 3d mesh
- rest of mesh deforms smoothly (texture behaves like a rubber sheet)

Motivation:

most common deformation.
align features in texture to corresponding features on mesh.

Feature Alignment

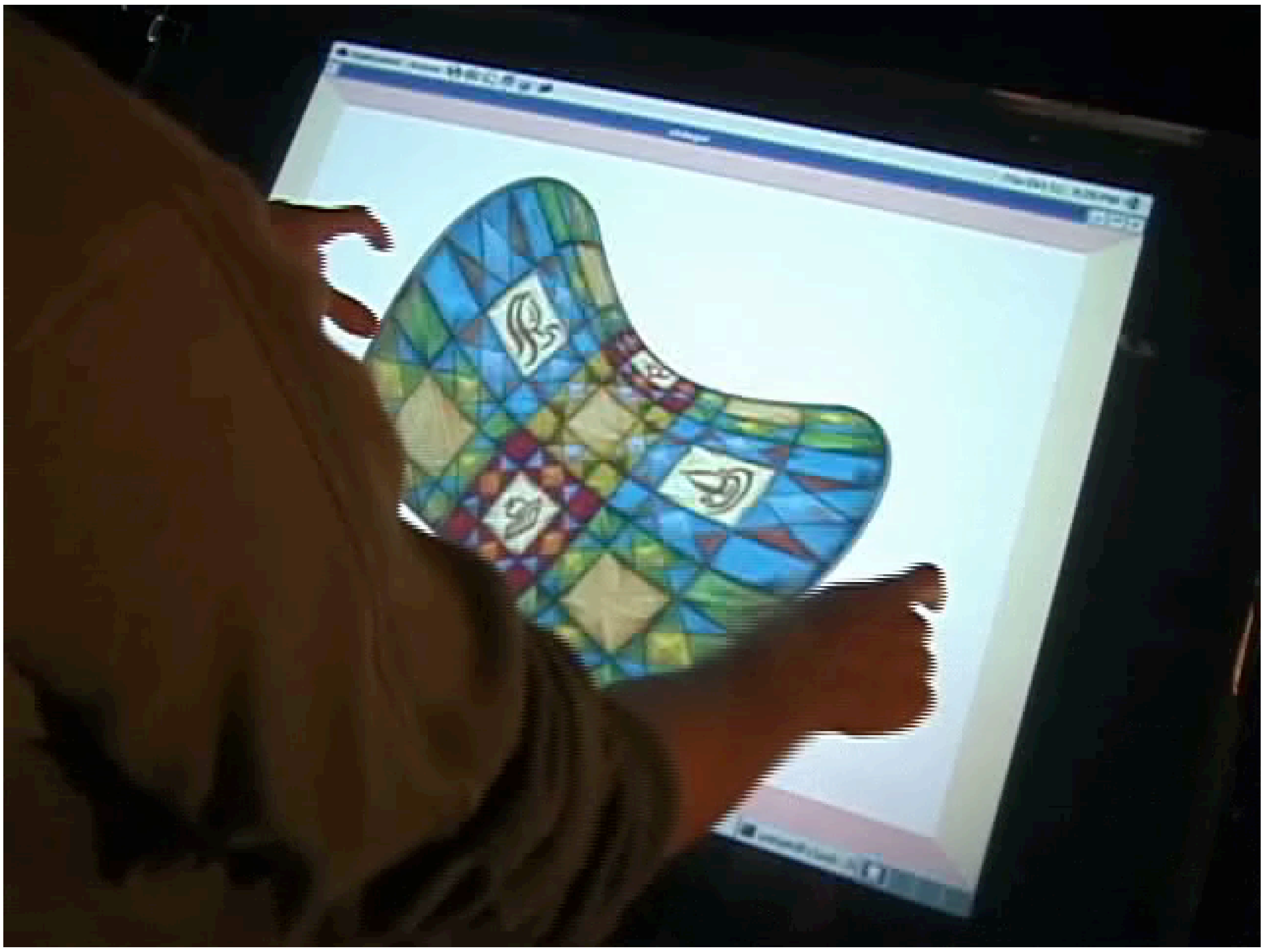


Point constraints
- assign points in texture to points on 3d mesh
- rest of mesh deforms smoothly (texture behaves like a rubber sheet)

Motivation:

most common deformation.
align features in texture to corresponding features on mesh.

Pushpin Constraints



Point constraints remain fixed like pushpins.

motivation:

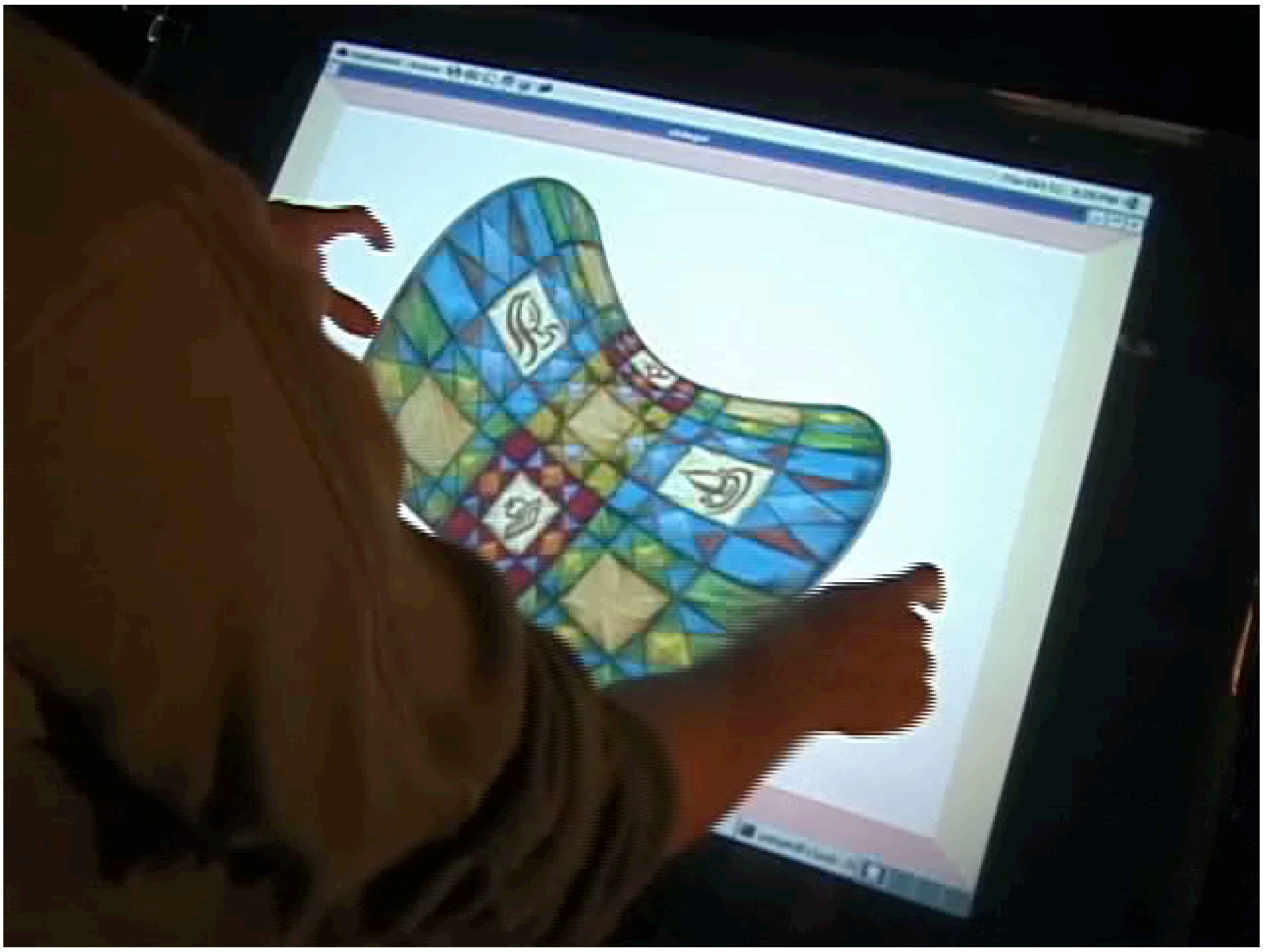
like a tailor, can return to and move or remove the push-pins.

- surface bounces back elastically.

- encourages experimentation.

- (more flexible than undo because they can be done and undone in any order.)

Pushpin Constraints



Point constraints remain fixed like pushpins.

motivation:

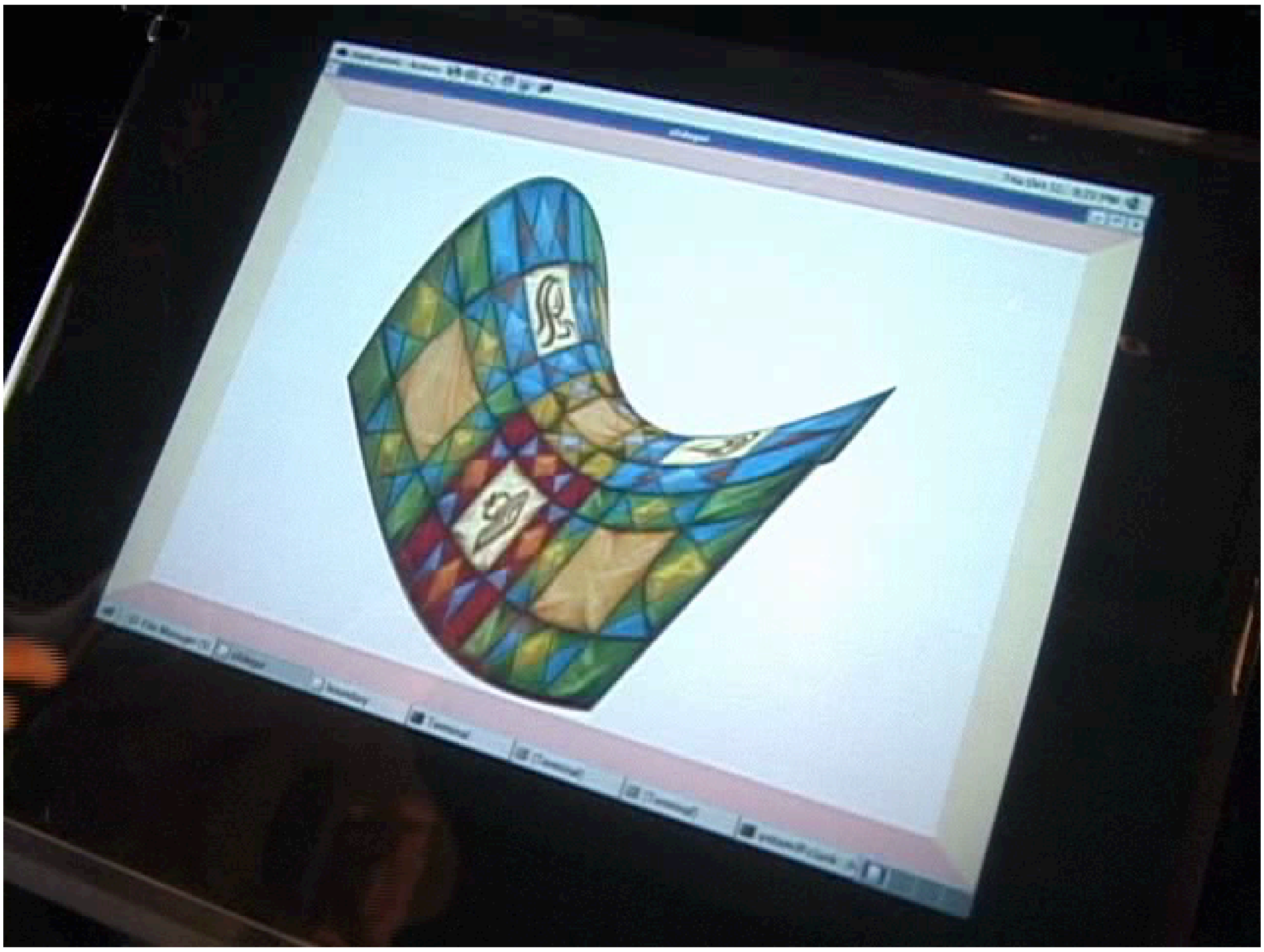
like a tailor, can return to and move or remove the push-pins.

- surface bounces back elastically.

- encourages experimentation.

- (more flexible than undo because they can be done and undone in any order.)

Plastic Update



Bake point constraints into the rest state of the texture

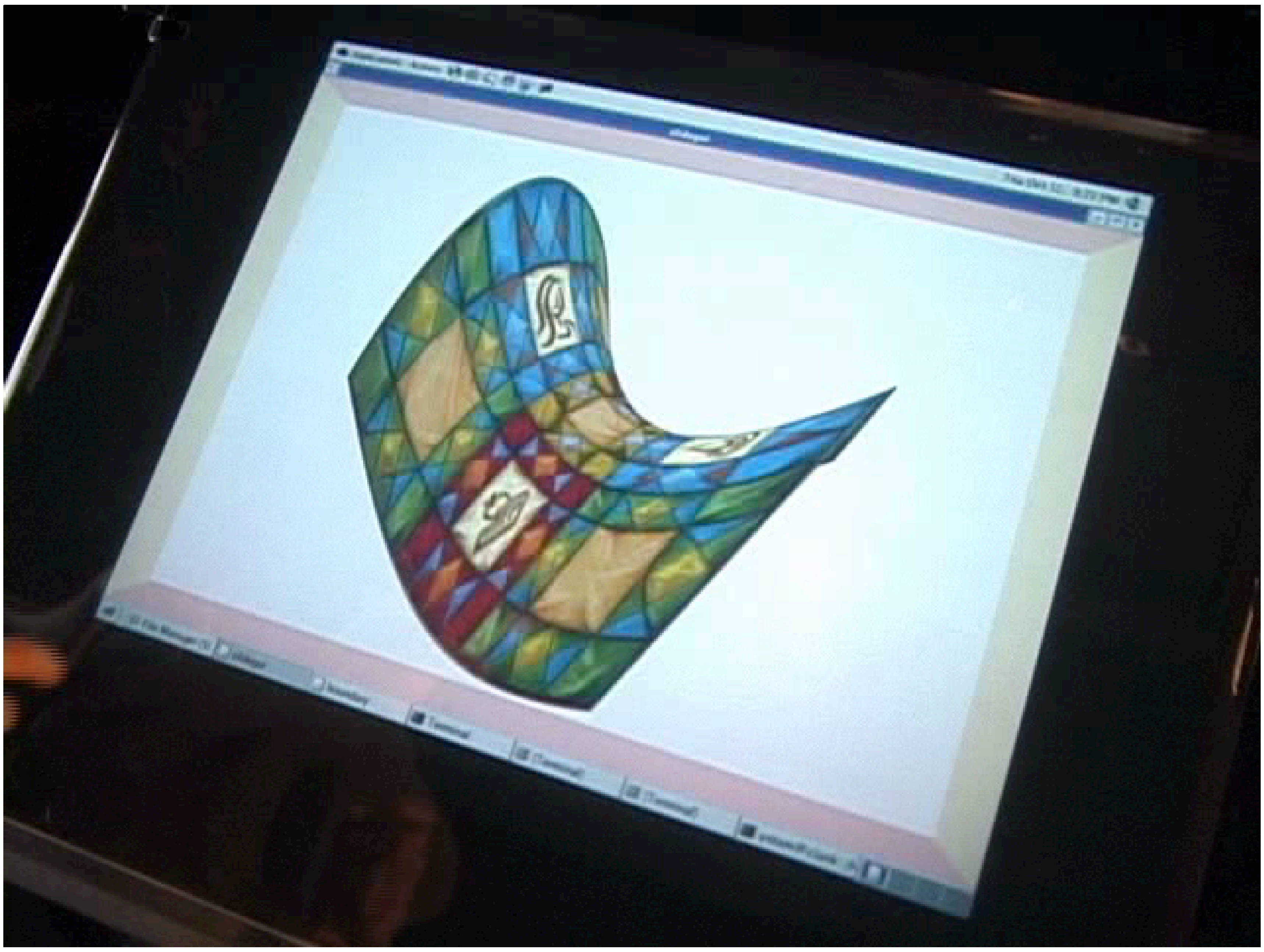
- makes distortion permanent
- exactly like a material behaving “plastically”

motivation:

- Excessive pushpin constraints can prevent smooth texture deformations.

old: the current state of the rubber sheet becomes the neutral state; now deforms elastically from the new configuration

Plastic Update



Bake point constraints into the rest state of the texture

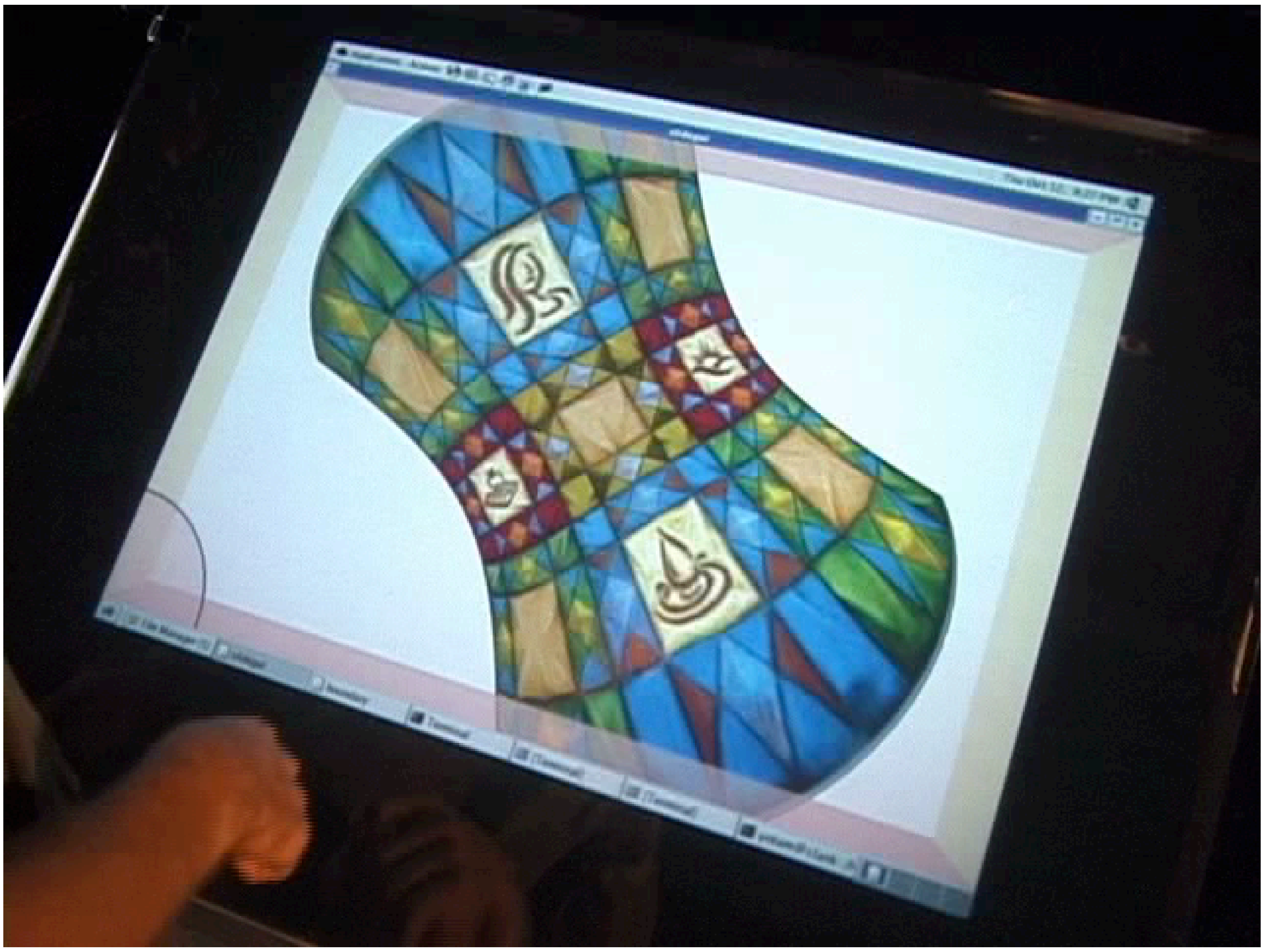
- makes distortion permanent
- exactly like a material behaving “plastically”

motivation:

- Excessive pushpin constraints can prevent smooth texture deformations.

old: the current state of the rubber sheet becomes the neutral state; now deforms elastically from the new configuration

Local Deformations

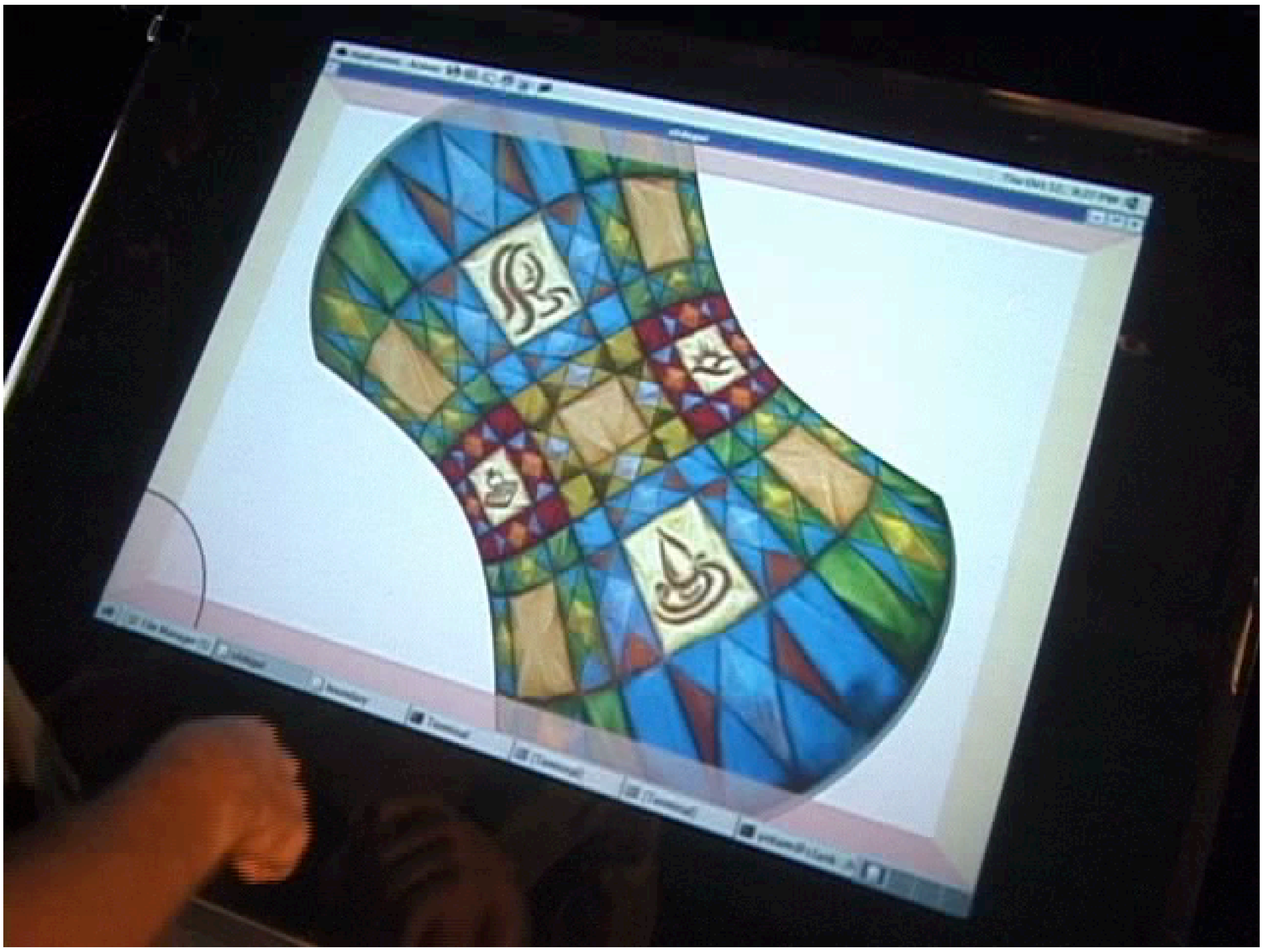


Isolate deformations within a circle

Motivation: Important to have a tool for making fine adjustments (((or for protecting local features)))

old: modifications inside the circle (or outside) don't cross to the other side

Local Deformations

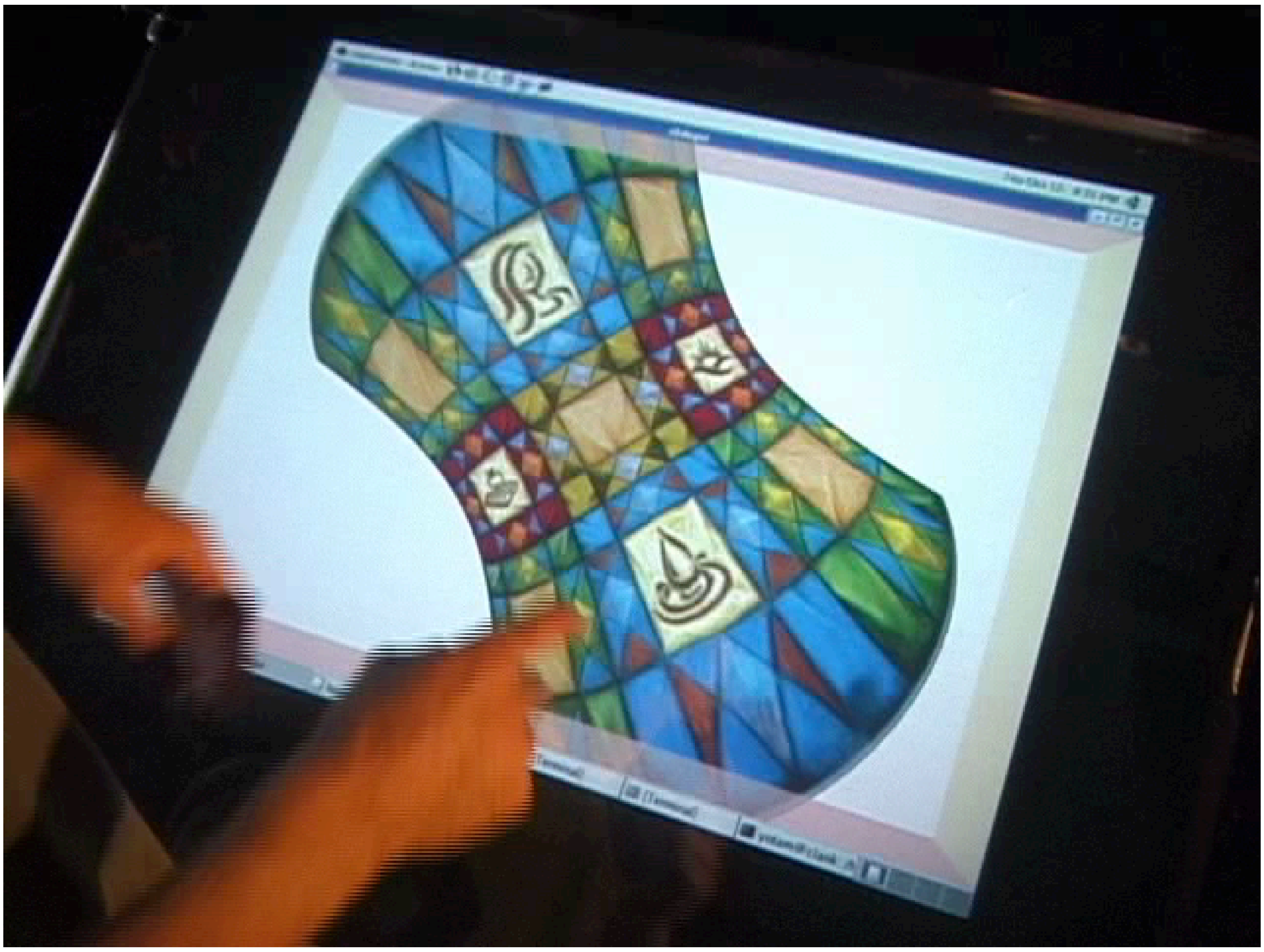


Isolate deformations within a circle

Motivation: Important to have a tool for making fine adjustments (((or for protecting local features)))

old: modifications inside the circle (or outside) don't cross to the other side

Glue



Barriers to change; deformations don't cross glue boundaries.

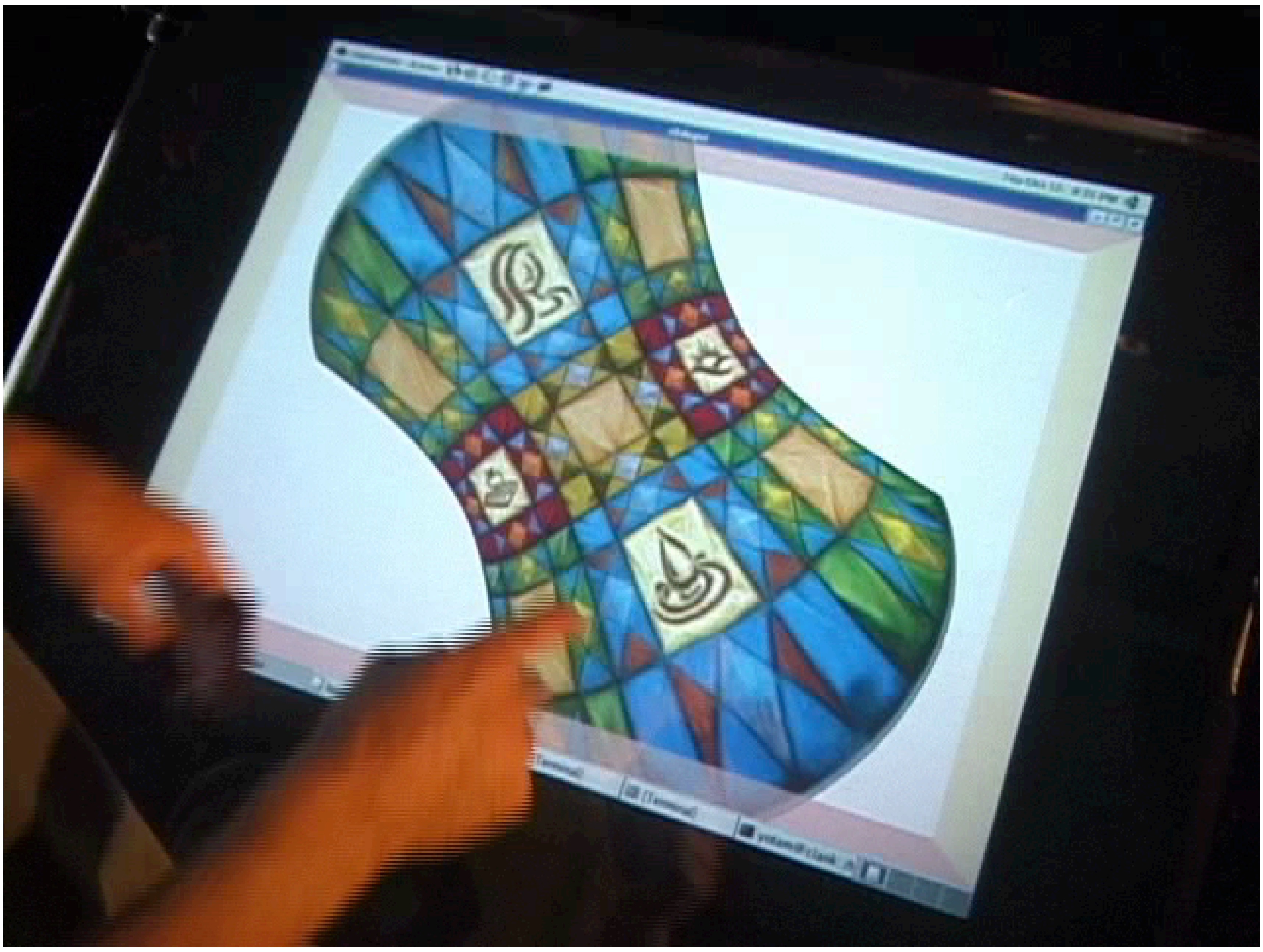
Can be drawn free-form on the mesh

Motivation:

localizing deformations within circles is convenient, but the same idea is generally useful for fine control.

- isolate finished regions of the mesh from in-progress areas

Glue



Barriers to change; deformations don't cross glue boundaries.

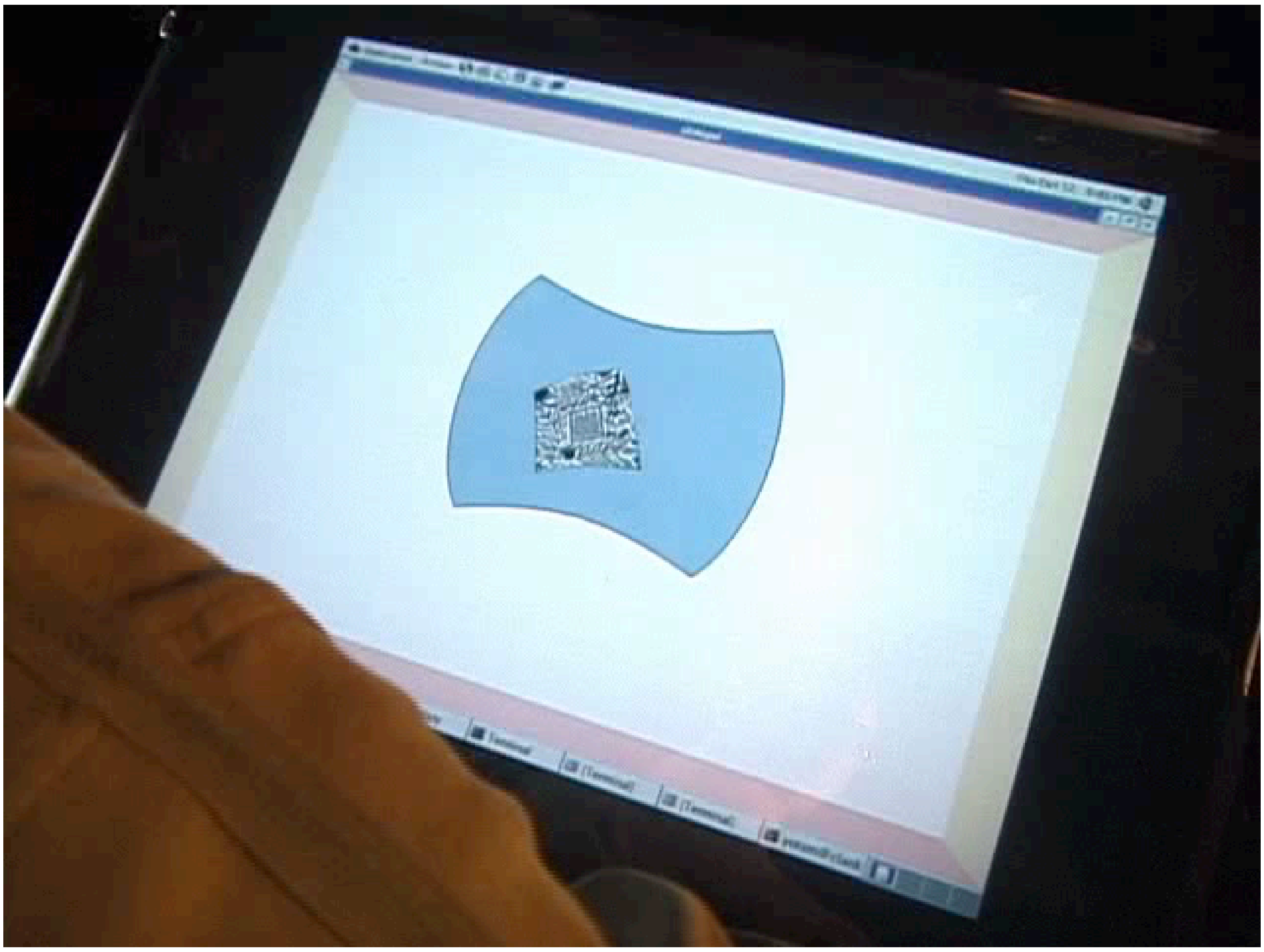
Can be drawn free-form on the mesh

Motivation:

localizing deformations within circles is convenient, but the same idea is generally useful for fine control.

- isolate finished regions of the mesh from in-progress areas

Texture Layers



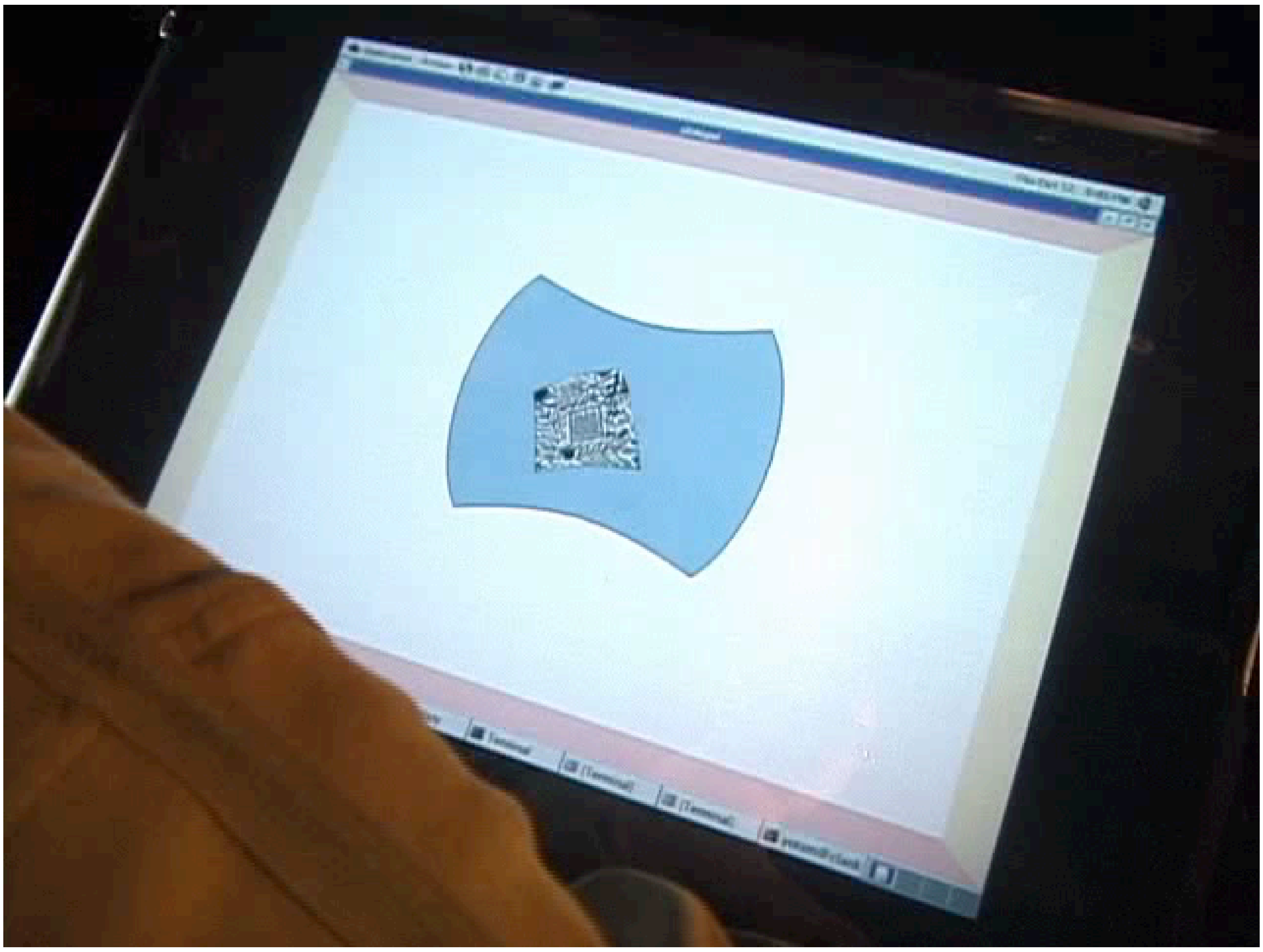
This is the placement step for multiple texture compositing.

Motivation:

- Want to use multiple photos or drawings from different angles.
  - photo taken from one angle (or close-up of, say, an eye) useful for part of the mesh, not the whole thing
  - hard to find (or draw) a single texture that depicts the surface detail from every angle
    - that is a flattening! there will be lots of distortion
- Allows artists to obtain (photograph or draw) each part of the geometry in a fashion convenient for them, and then layer them together.

Example use: photos from different angles

Texture Layers



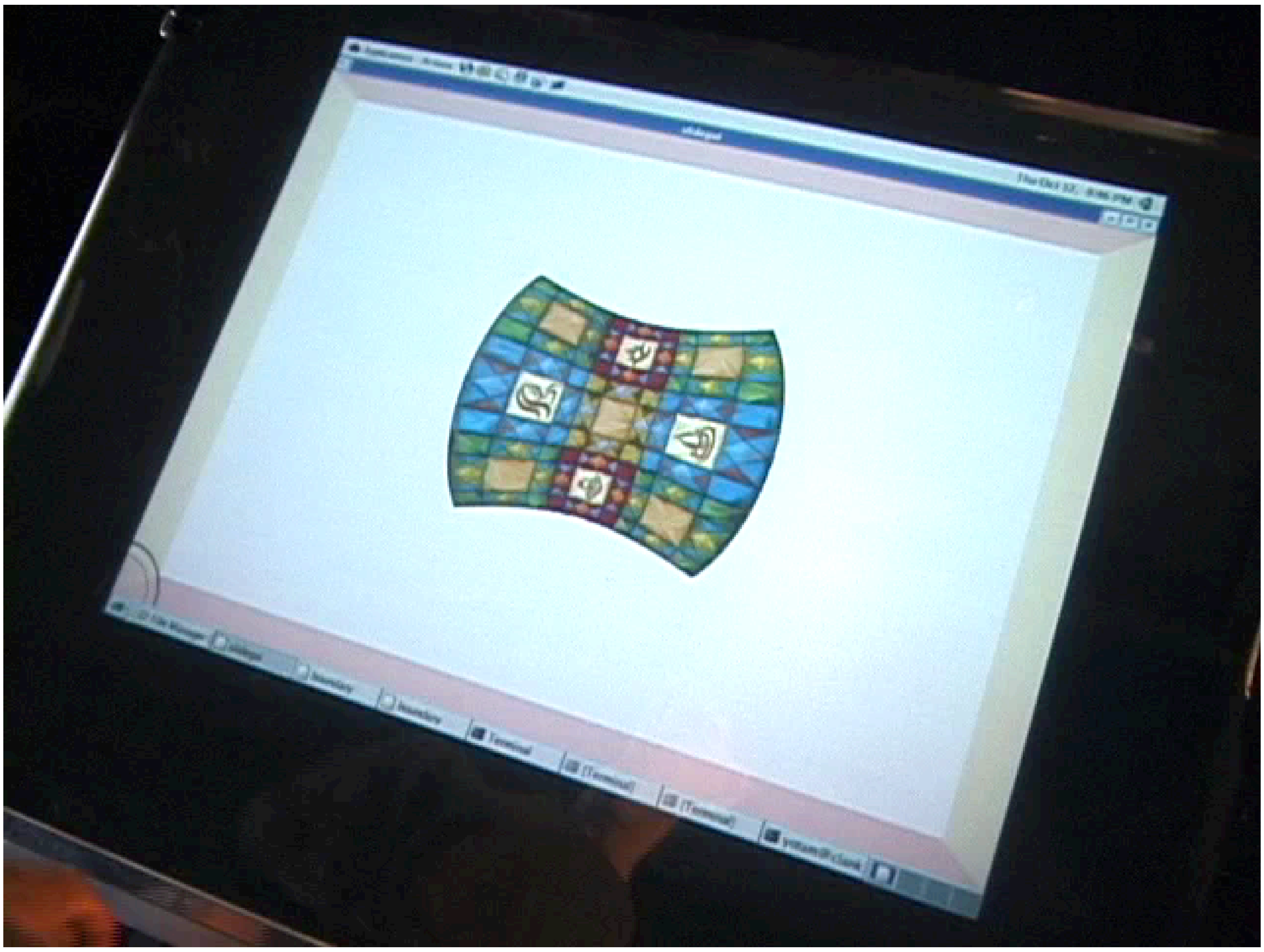
This is the placement step for multiple texture compositing.

Motivation:

- Want to use multiple photos or drawings from different angles.
  - photo taken from one angle (or close-up of, say, an eye) useful for part of the mesh, not the whole thing
  - hard to find (or draw) a single texture that depicts the surface detail from every angle
    - that is a flattening! there will be lots of distortion
- Allows artists to obtain (photograph or draw) each part of the geometry in a fashion convenient for them, and then layer them together.

Example use: photos from different angles

Alpha Airbrush



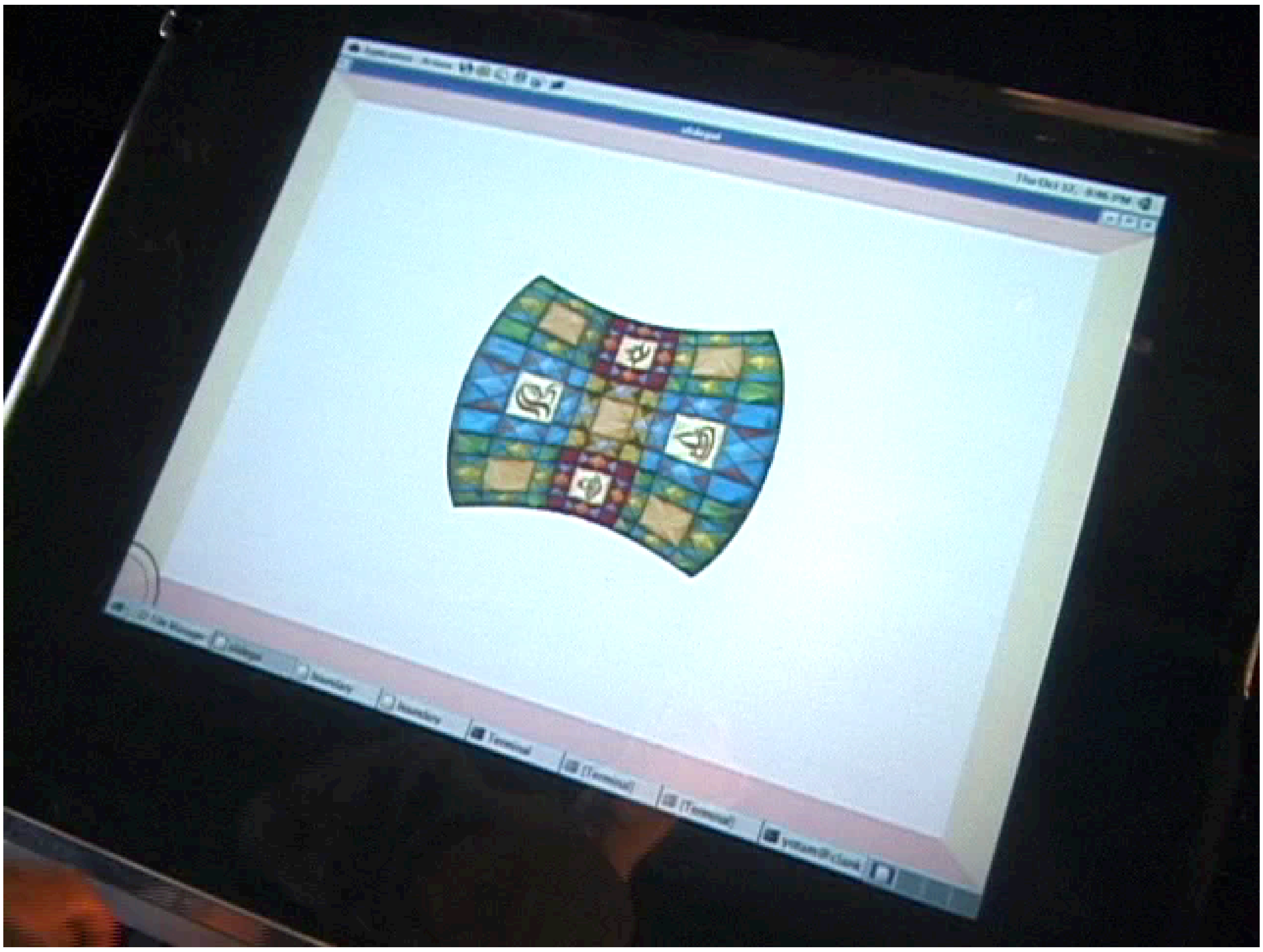
Finally, we have the alpha airbrush;
this is the tool for blending between texture layers so images fit together nicely.

Sprays transparency or opacity.

Motivation:

- different layers will have overlap
  - makes artist's job easier, he or she doesn't have to worry about seams

Alpha Airbrush



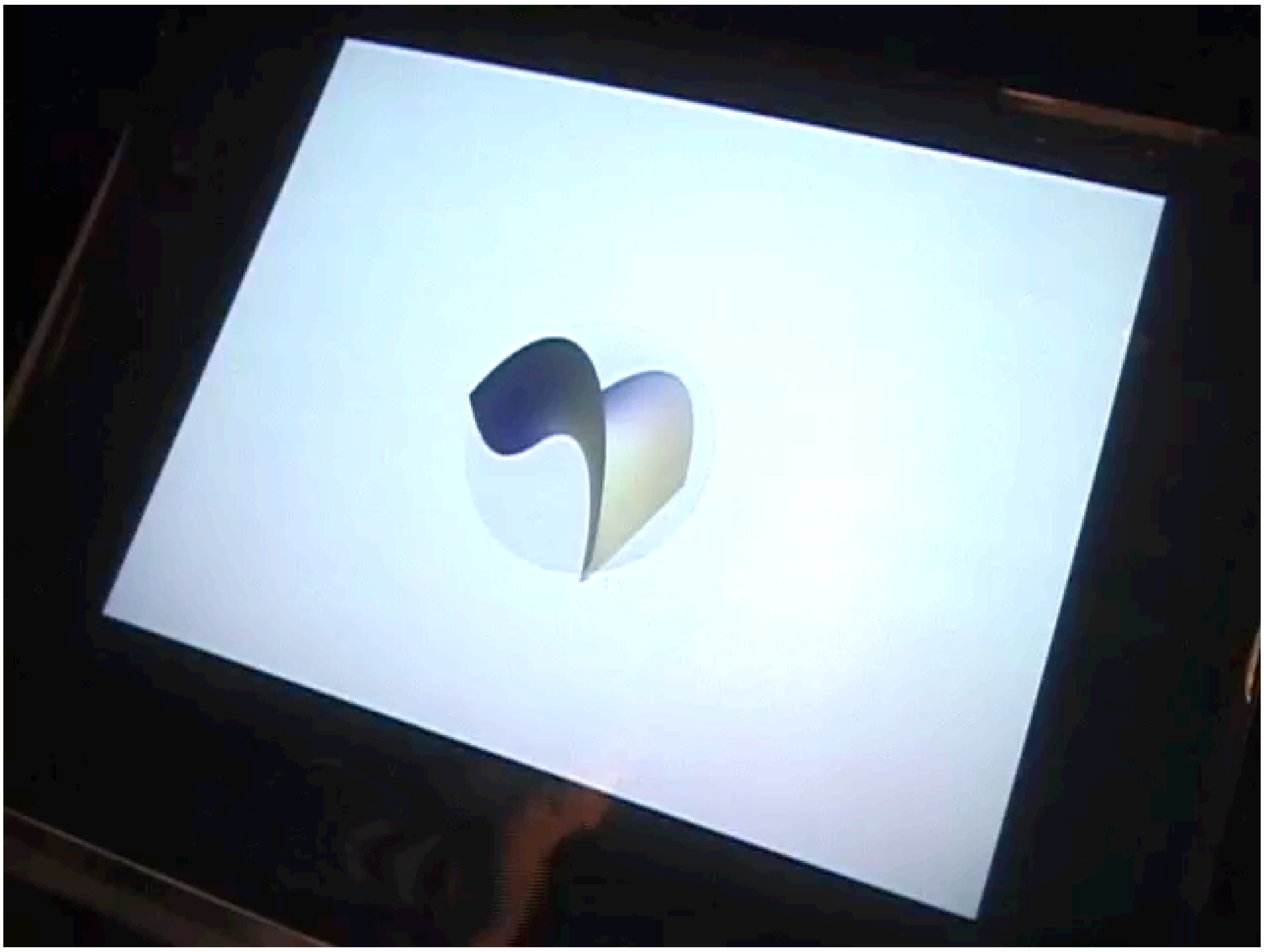
Finally, we have the alpha airbrush;
this is the tool for blending between texture layers so images fit together nicely.

Sprays transparency or opacity.

Motivation:

- different layers will have overlap
    - makes artist’s job easier, he or she doesn’t have to worry about seams

Object Positioning



For object positioning, we have the ‘tiltpad,’ a disk-shaped pan-zoom-rotate & tilt control.

In-plane similarity transform performed from 2-finger manipulation of the disc.

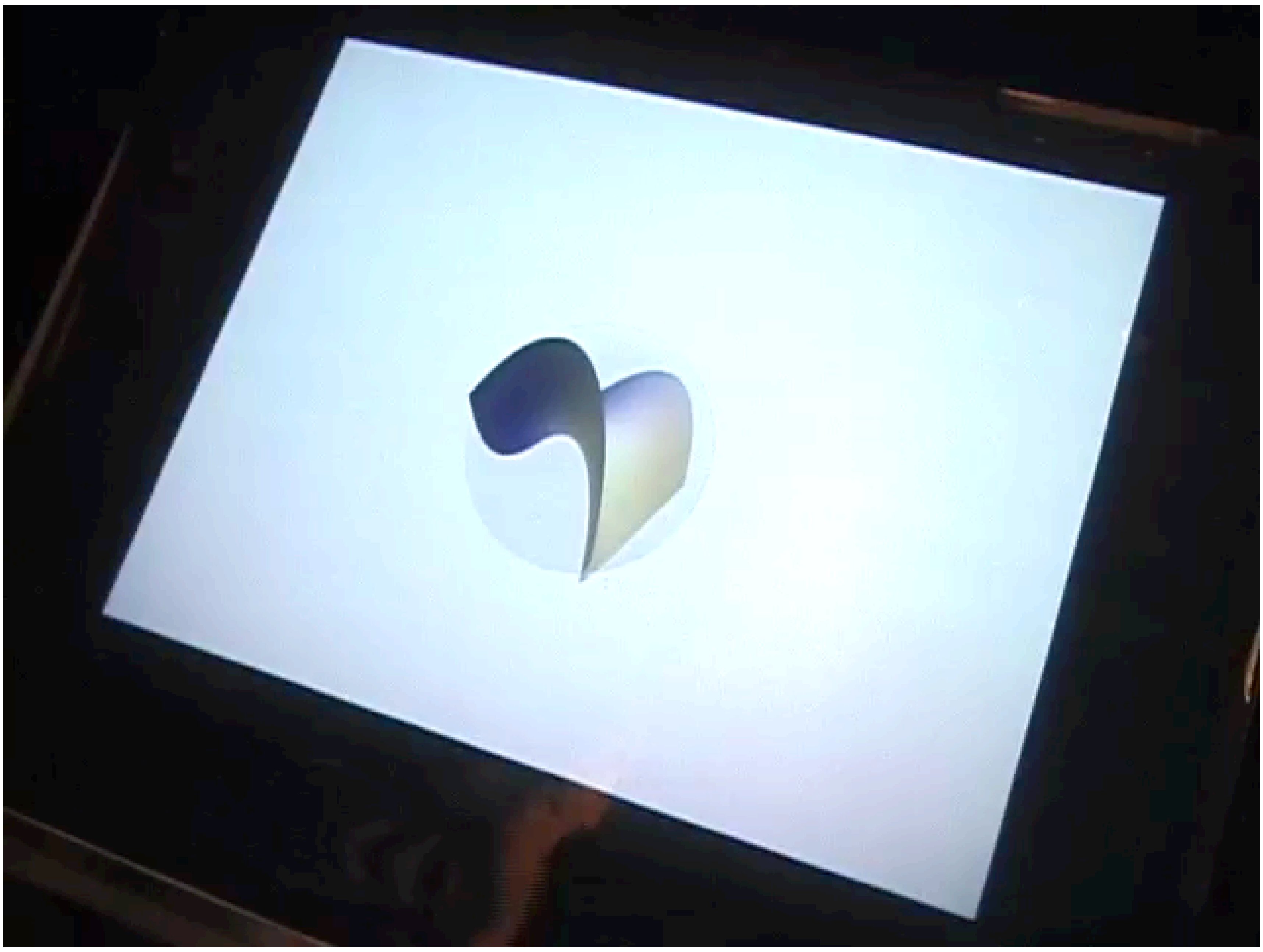
In addition, the disc “tilts” with pressure, tilting into the screen;

- the rotation from flat to tilted, determines an out-of-plane rotation for the object.

Motivation:

need a scheme for manipulating the 3D geometry, especially a natural one that can be used by the non-dominant hand

Object Positioning

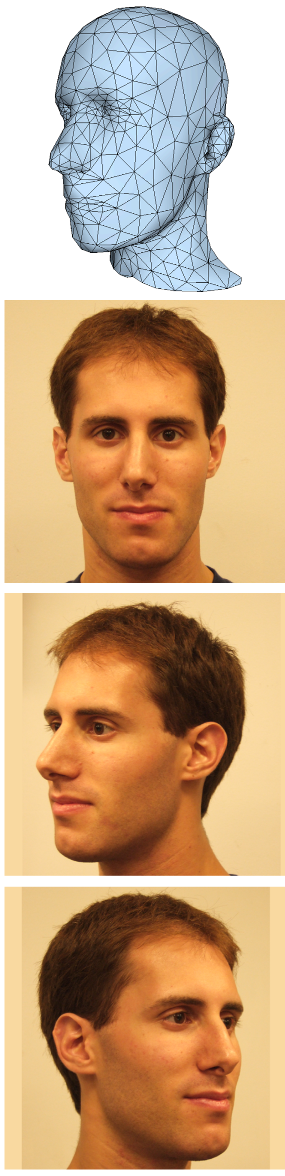


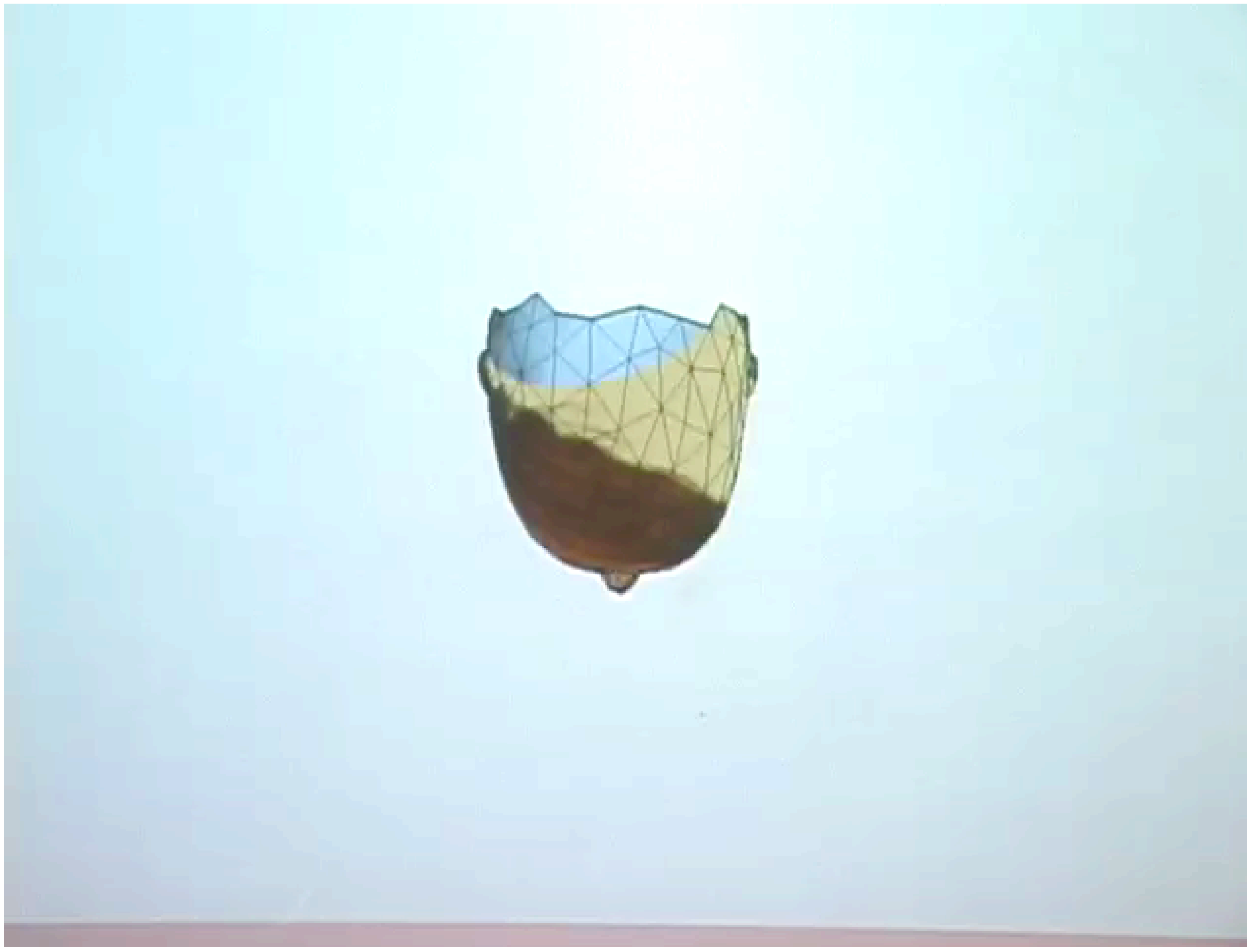
For object positioning, we have the ‘tiltpad,’ a disk-shaped pan-zoom-rotate & tilt control.
In-plane similarity transform performed from 2-finger manipulation of the disc.
In addition, the disc “tilts” with pressure, tilting into the screen;
- the rotation from flat to tilted, determines an out-of-plane rotation for the object.

Motivation:

need a scheme for manipulating the 3D geometry, especially a natural one that can be used by the non-dominant hand

Results



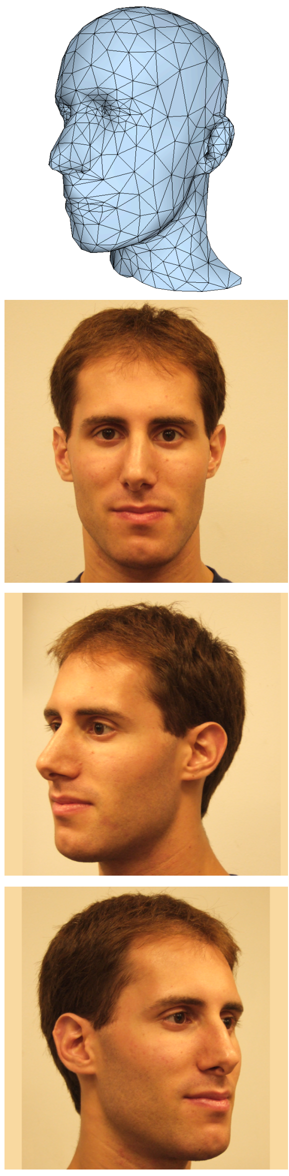


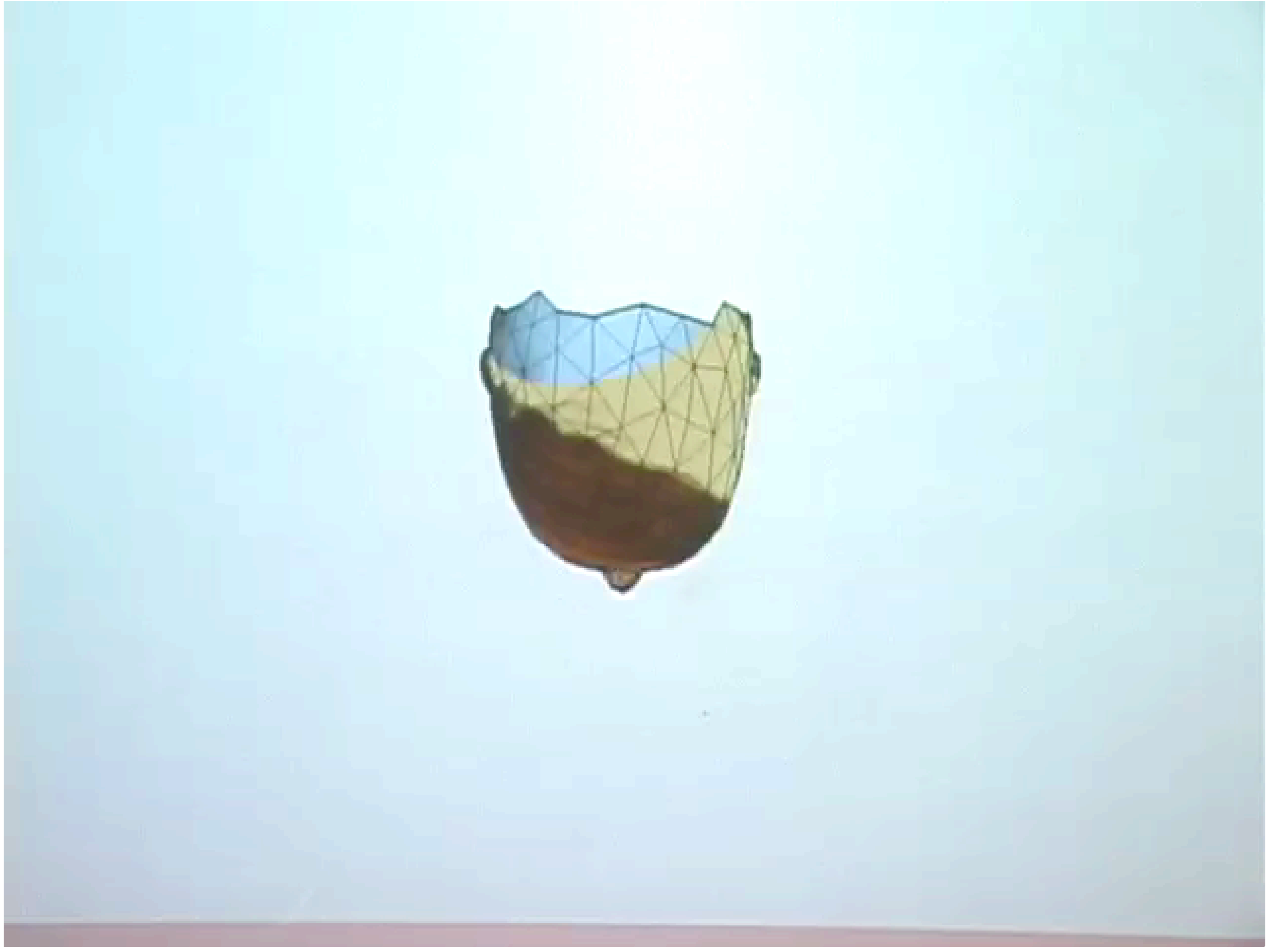
Here is a video of our system in use, sped up 4x.
This video shows a non-artist user familiar with our system.

Entire process took 4.5 minutes (about 1 minute here)
- good results even without fine adjustment tools

We have created a textured model very quickly, using found source data (photographs), working entirely in the final 3d view.

Results





Here is a video of our system in use, sped up 4x.
This video shows a non-artist user familiar with our system.

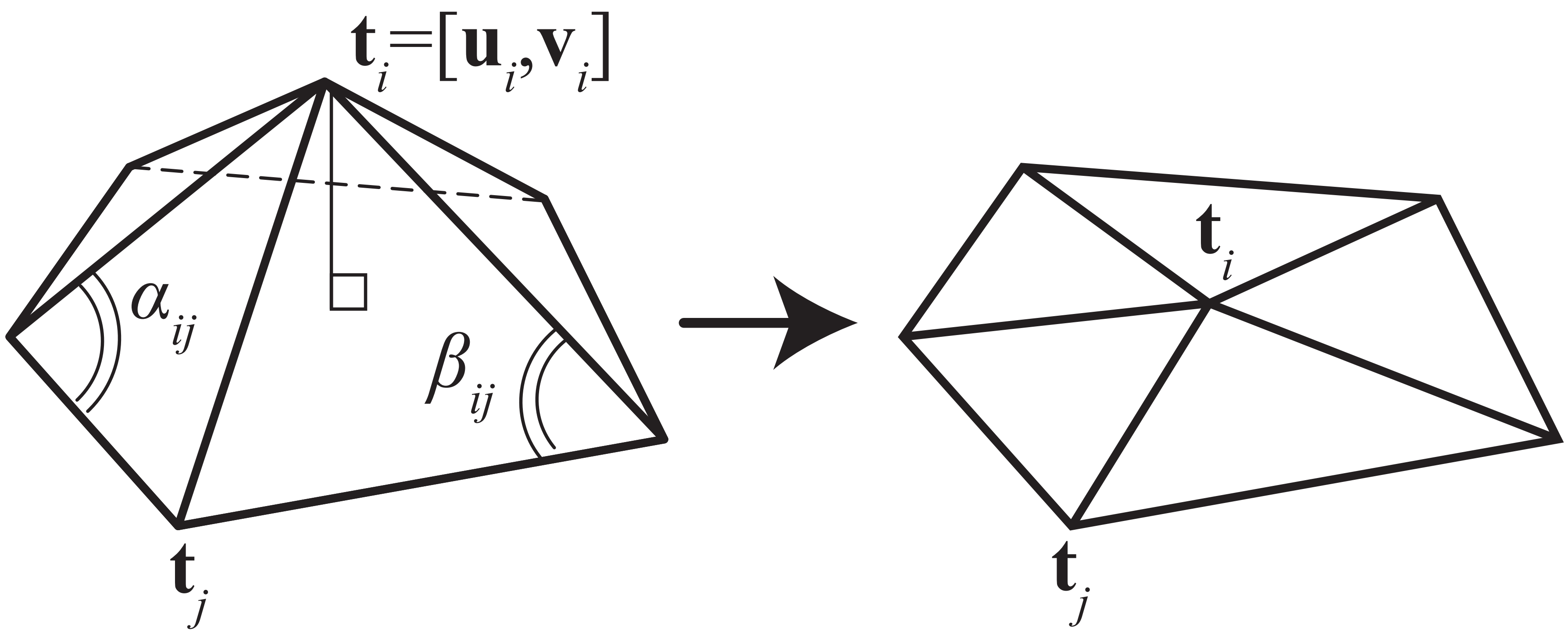
Entire process took 4.5 minutes (about 1 minute here)
- good results even without fine adjustment tools

We have created a textured model very quickly, using found source data (photographs), working entirely in the final 3d view.

3 Formulae

Parameterization Algorithm

Linearized Bending Energy



$$t^T A t = E = \sum_i \frac{1}{8\, area_i} \left(\sum_{j \in N(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(\mathbf{t}_i - \mathbf{t}_j) \right)^2$$

- recall: Parameterization is the flattening of the mesh onto a plane

- We flatten the mesh in a way that minimizes bending the mesh.
- The flattening we get out of linearized bending energy handles constraints robustly, as we'll see in our 1 technical comparison.

- Linearized, so most terms are computed from the undeformed 3d configuration.
- Quadratic, so the minimum is a linear system ("A" above)
- Closed form
- Each operation is fast, interactive, and stable.
- Cotangent discretization [Pinkall & Polthier 1993] is stable

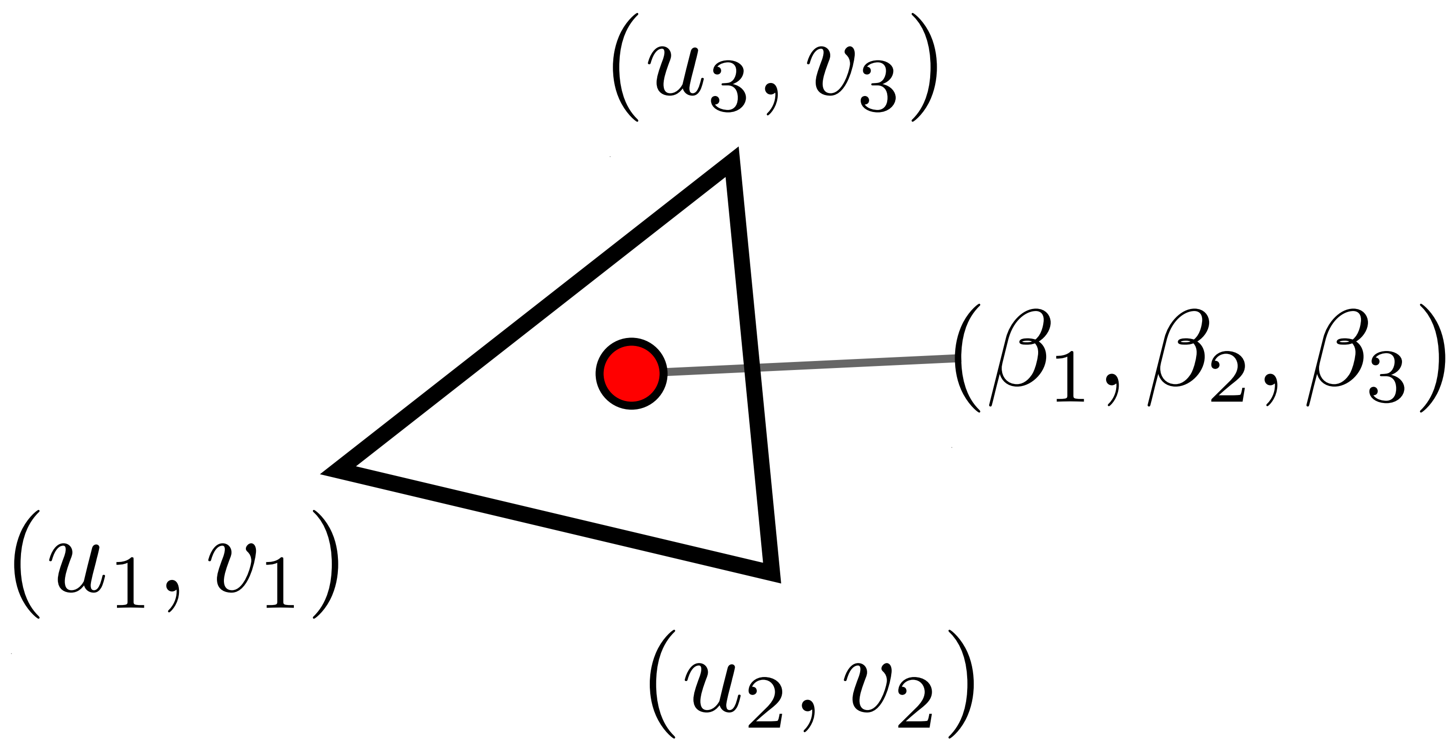
[]
- most parameterizations minimize stretching.
- these handle boundaries fine, but don't handle interior constraints (this will be our technical comparison)

Old:
- all the fixed terms get updated in a plastic update
- (sets the undeformed positions to the current deformation)

Free boundaries (for placement step) from
Intrinsic parameterizations of surface meshes.
M. Desbrun, M. Meyer, and P. Alliez.
Computer Graphics Forum, 21(3):209-218,
2002

Constraints

Linear on triangles



$$\beta_1 u_1 + \beta_2 u_2 + \beta_3 u_3 = u_{fixed}$$
$$\beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 = v_{fixed}$$

Point constraints are linear on triangles

- expressed barycentrically in terms of the triangle's three texture (u,v) coordinates.

Constraints

Modify system

bending energy Hessian

$$A^{ext} = \begin{pmatrix} A & C^T \\ C & 0 \end{pmatrix}$$

constraints $\begin{pmatrix} \beta_1 u_1 + \beta_2 u_2 + \beta_3 u_3 = u_{fixed} \\ \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 = v_{fixed} \end{pmatrix}$

Need a scheme for quickly updating inverse

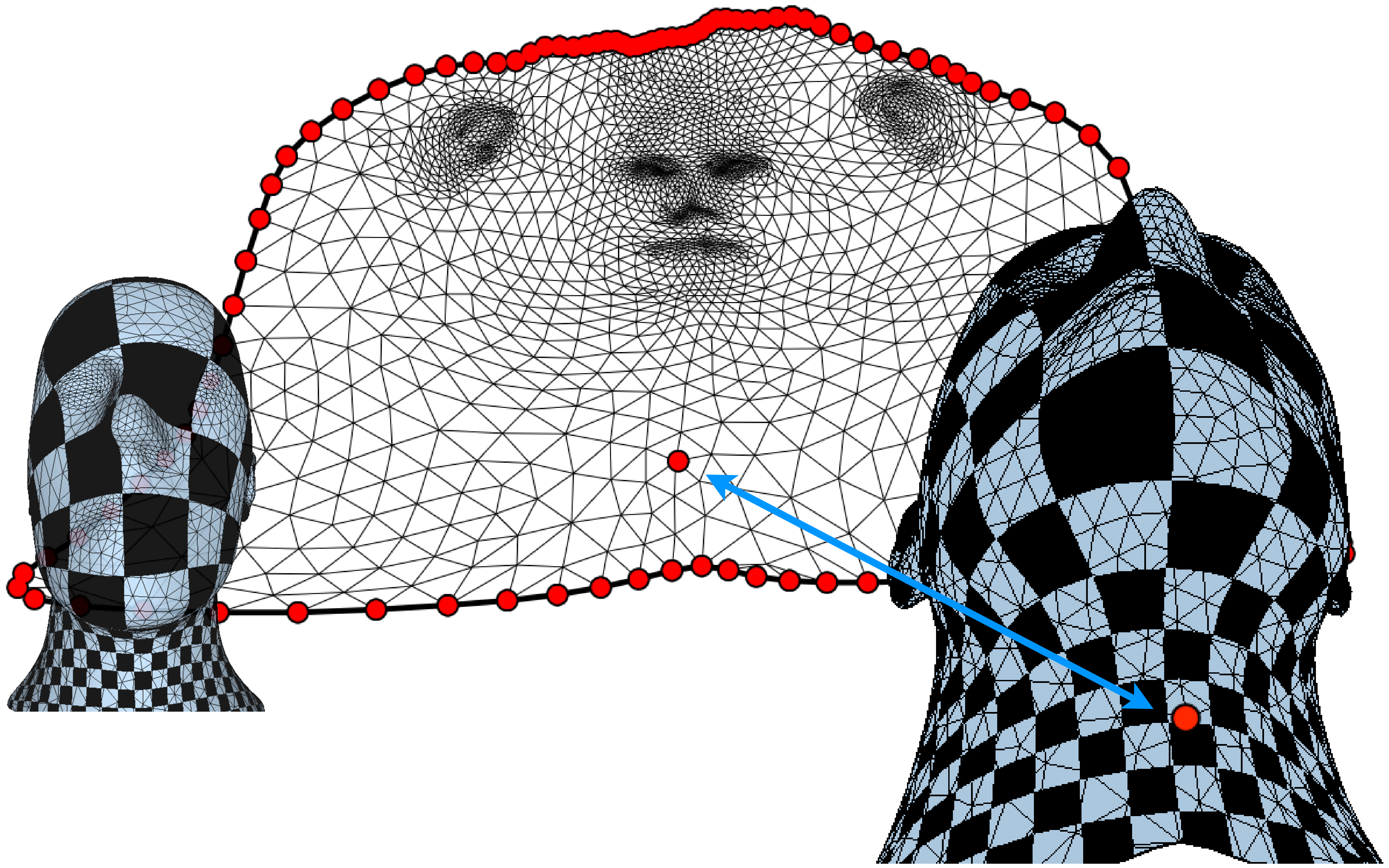
A^{ext} is the entire system matrix.
- Handle arbitrary linear constraints with lagrange multipliers (matrix ‘C’, which is small compared to ‘A’)
    - Glue & localized deformations are also linear constraints (on all vertices of a triangle)
        - (energy doesn’t cross a fixed two-ring of vertices)

- But constraints modify the system matrix A^{ext}!
    - for live constraints, we need a fast way of updating the inverse.
        - the Sherman-Morrison-Woodbury formula:
            - pre-compute (solve one system)
            - then we can find inverse with M live constraints by solving for 2M right hand sides
                - (fast so long as M is small -- limited by, say 10 fingers (7-9x speedup))

- At mouse/finger-up time: update pre-computed data (fold live constraints in)
- plastic deformations update matrix A (also at mouse/finger up-time)

I Technical Comparison

Constraint Matching

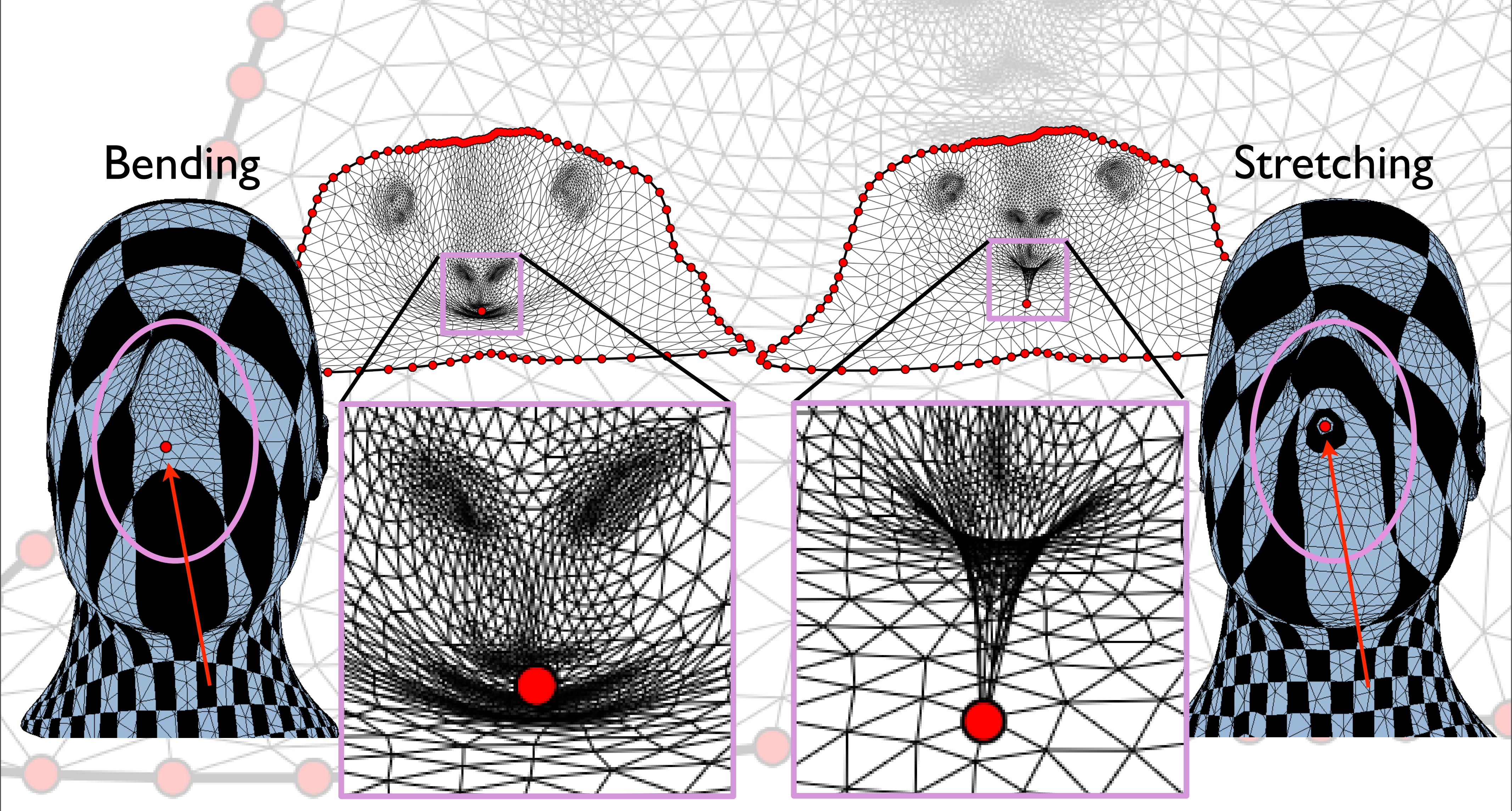


Compare parameterizations obtained from stretching energy, which is commonly used, to bending energy, what we use.

Bending and stretching, with no constraints, generate a similar flattening (shown here).

We introduce a point constraint (still at its rest state), shown in the interior in red.

Comparison



This is the parameterization after the constraint has moved along the red arrow.

- Bending energy matches the constraint smoothly & spreads out the distortion.
- Stretching matches constraint discontinuously, causing a fold-over, where multiple triangles map to the same piece of texture.
    - unusable for our purposes

- Bending energy can produce fold-overs, but it's much more robust than stretching.
- There are other parameterization approaches, but ones that guarantee no fold-overs are non-linear; bending energy is simple and fast, and interactivity is key.

Contributions

System for direct manipulation of textures in 3D

- Re-use: can use found textures & same texture on multiple models (with a different deformation).

Several types of constraints

- If we want artists to use our system, it needs to be refinable.
- Finer refinements should be possible, and they should modify the result smoothly

Contributions

System for direct manipulation of textures in 3D

Create textured models more easily & with less skill

- Re-use: can use found textures & same texture on multiple models (with a different deformation).

Several types of constraints

- If we want artists to use our system, it needs to be refinable.
- Finer refinements should be possible, and they should modify the result smoothly

Contributions

System for direct manipulation of textures in 3D

Create textured models more easily & with less skill

Provides a variety of tools for the user

- Re-use: can use found textures & same texture on multiple models (with a different deformation).

Several types of constraints

- If we want artists to use our system, it needs to be refinable.
- Finer refinements should be possible, and they should modify the result smoothly

Contributions

System for direct manipulation of textures in 3D

Create textured models more easily & with less skill

Provides a variety of tools for the user

Exploits multi-touch input

- Re-use: can use found textures & same texture on multiple models (with a different deformation).

Several types of constraints

- If we want artists to use our system, it needs to be refinable.
- Finer refinements should be possible, and they should modify the result smoothly

Contributions

System for direct manipulation of textures in 3D

- Create textured models more easily & with less skill
- Provides a variety of tools for the user
- Exploits multi-touch input
- Bending Energy for parameterization

- Re-use: can use found textures & same texture on multiple models (with a different deformation).

Several types of constraints

- If we want artists to use our system, it needs to be refinable.
- Finer refinements should be possible, and they should modify the result smoothly

Future Work

Image editing operations

Parameterization robustness

User evaluations

Image editing

- PhotoMontage texture blending

Parameterization robustness

- Multiple constraints per triangle

User evaluations

Acknowledgments

NYU Computer Science colleagues

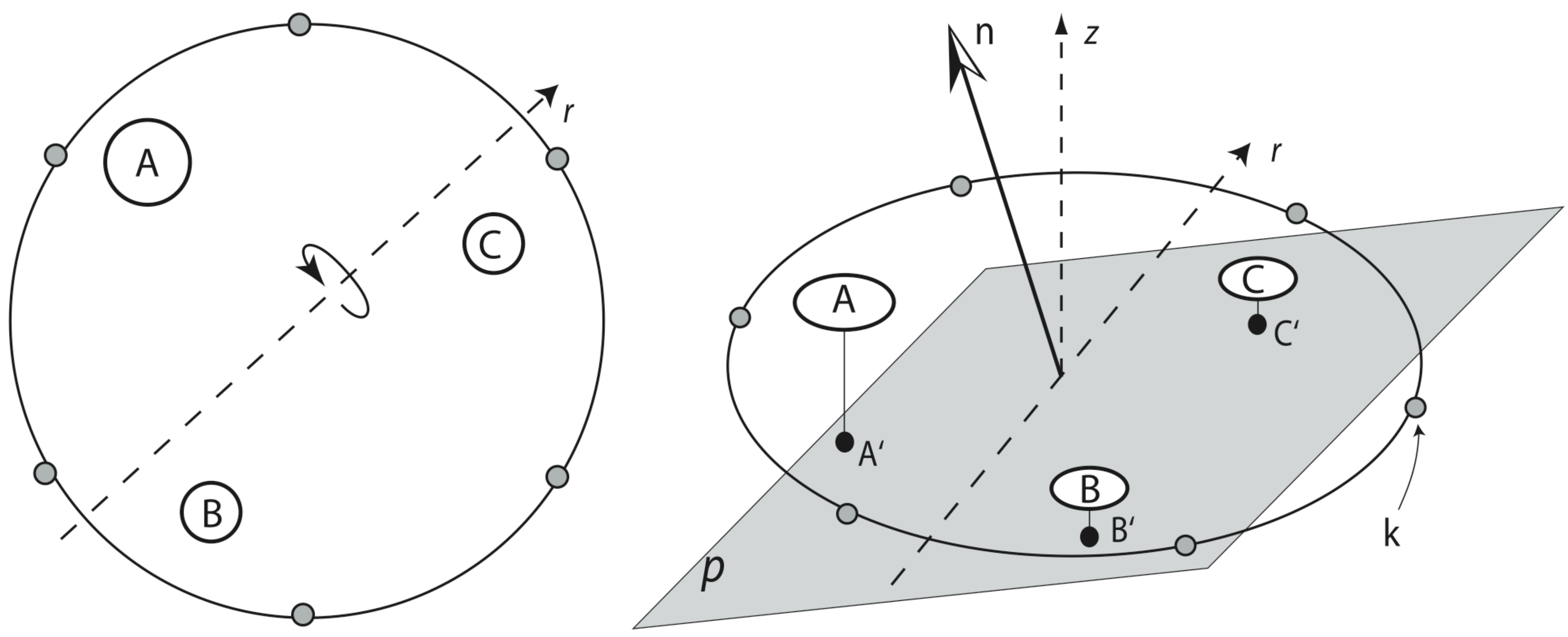
Anonymous reviewers

Mike Khoury

Yotam Gingold <gingold@cs.nyu.edu>

End

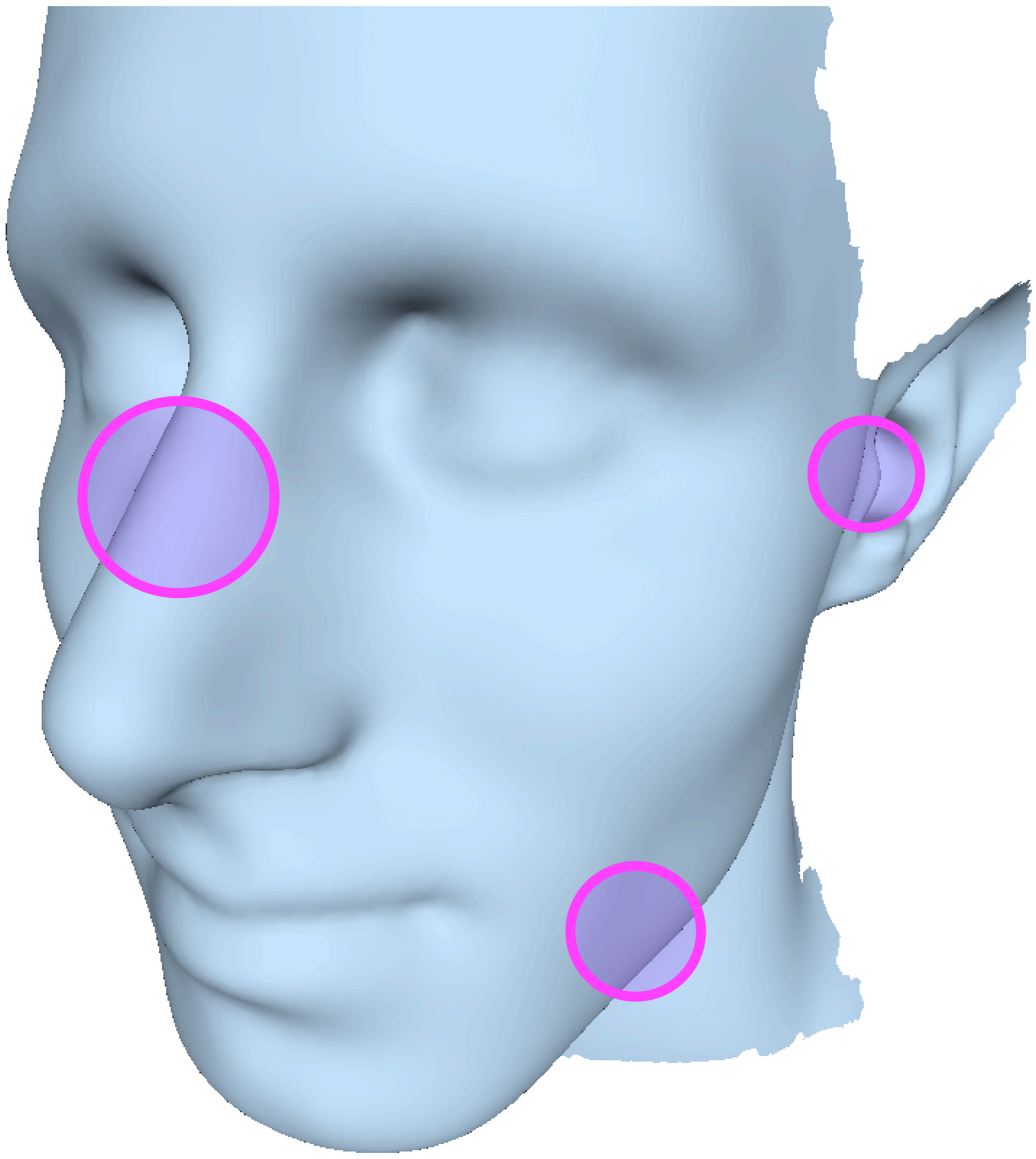
Object Positioning



The tiltpad is a disk-shaped isotonic pan-zoom-rotate control augmented by
an isometric `tilt' measurement, shown in Figure~\ref{fig:tiltpad} (left).
We first determine orientation in the plane, using the least-squares solution
for translation, uniform scaling, and rotation. Then, we apply a `tilt' rotation
about an axis r in the view plane, calculated using terms indicated in Figure~\ref{fig:tiltpad} (right).
For the three contact points A,B,C (white) falling in the control area,
we interpret pressure as `depth' below the surface, and find the normal n of the
best-fit plane p to the projected points A',B',C' (black). The `tilt' transform is
applied as an incremental rotation around the in-plane axis r that maps the z-axis to n.
To ensure that n is well-defined, we add a set of weak constraint points k (grey) around the circular
boundary of the control
%at zero depth zero.
with depth equal to zero.

For consistency, we adjust solution parameters when points are
added or removed in order to maintain a constant transformation.
Pressure values are thresholded using a `deadband' model, providing
a transition from purely planar motion to tilt-sensitive manipulation.

Alpha Airbrush



Wraps around mesh at center-point

This prevents discontinuities along silhouette edges.
(like “parameter space brushes” in [Hanrahan & Haeberli 1990])

[TODO: radius widget]

With this mapping (and its inverse), we can transform the airbrush circle to texture coordinates and iterate over every texel inside, transforming back for the radius fall-off.

Screen-space radius
Compute matrix converting picked triangle’s screen coordinates to texture coordinates
Project airbrush bounding box into texture space
Iterate over every texel in projected bounding box
  project texel back into screen-space
  spray alpha with screen-space radius falloff

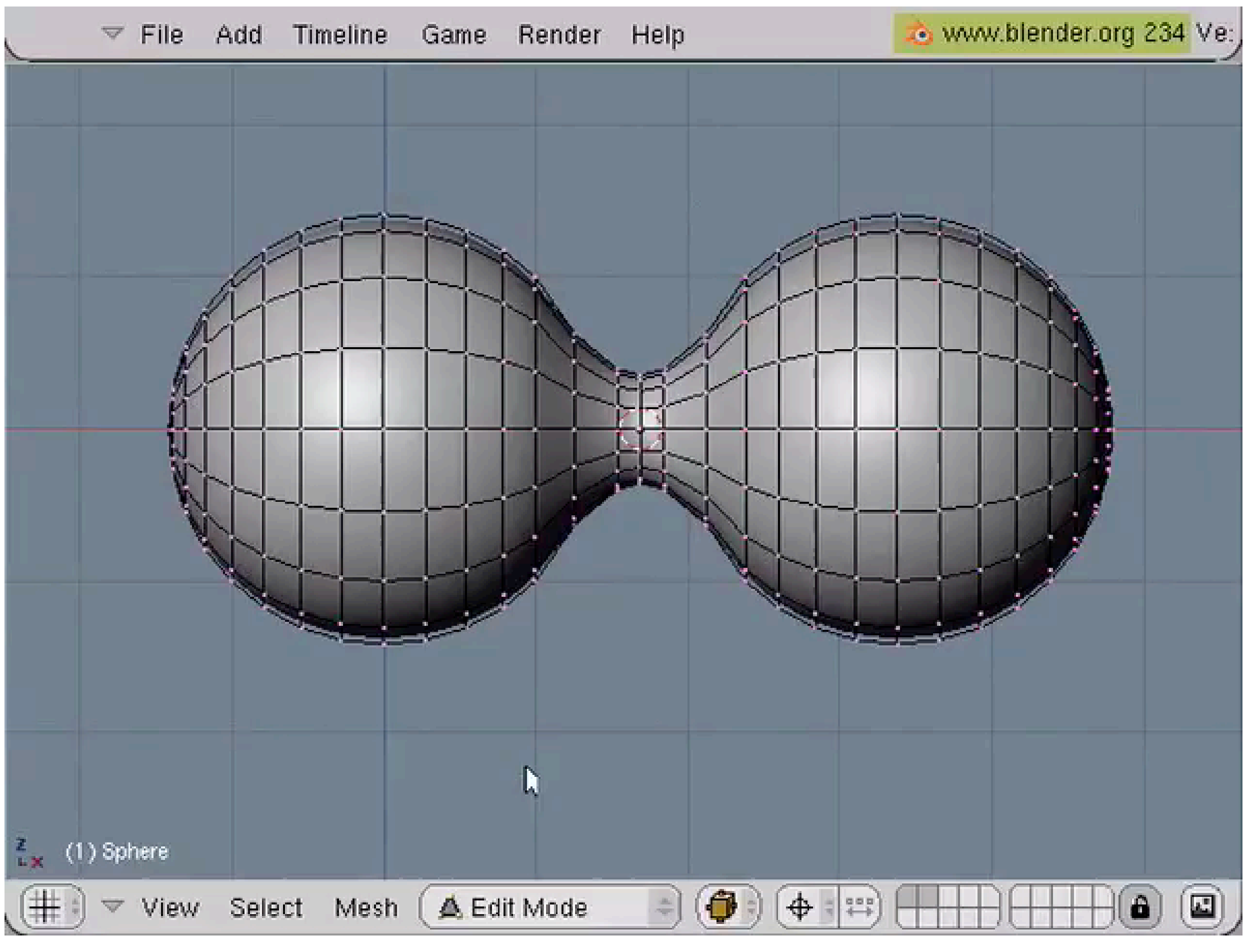
Results

Another example, using photographs from family album.

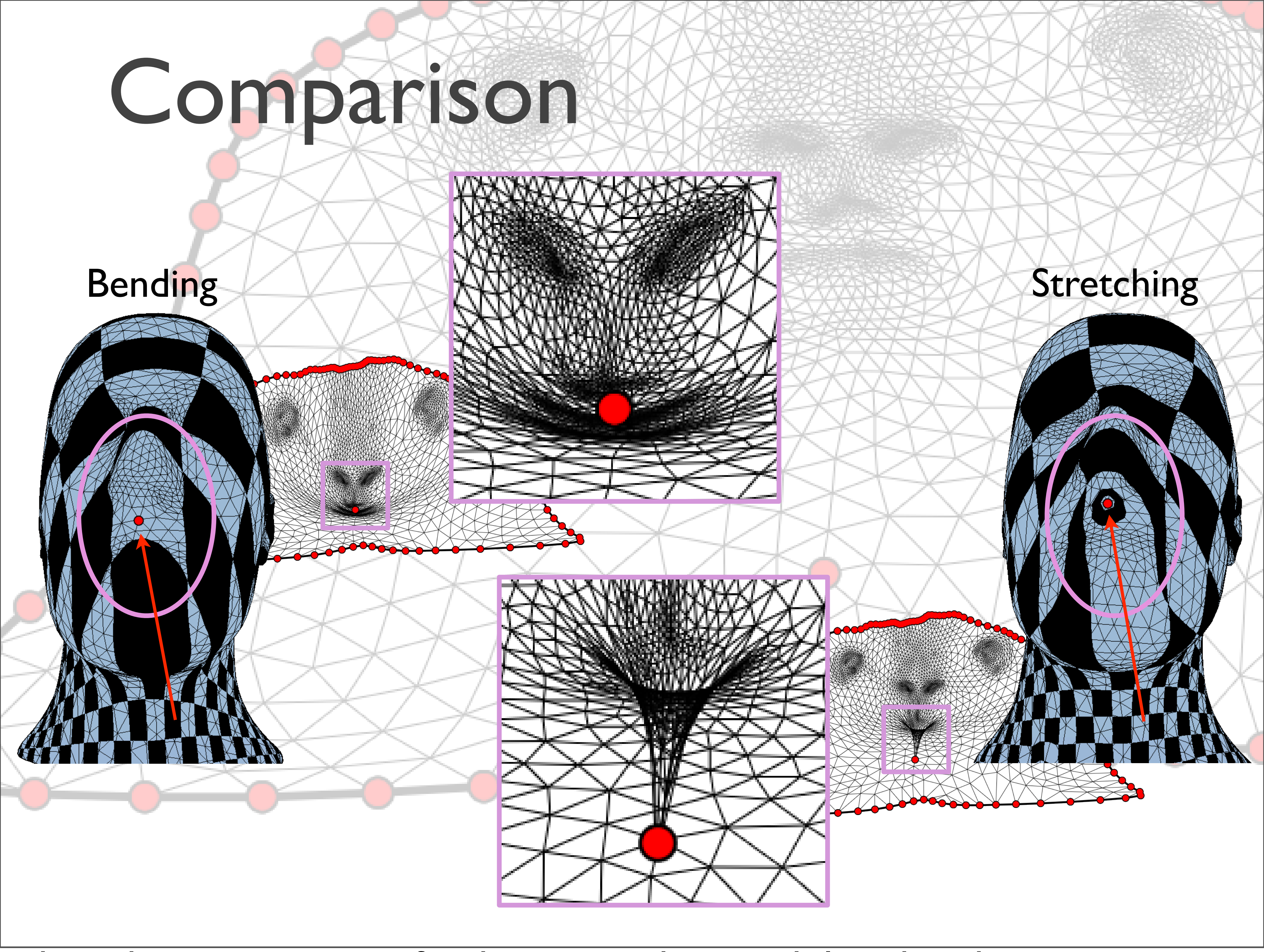
Also ~5 minutes, and by an amateur.

Traditional Approach

Assigning texture coordinates



- tutorial video: how to parameterize
  - Choose seams like a tailor
  - nice, automatic parameterization algorithm: Least Squares Conformal Maps [Lévy 2002]
  - still requires manual tweaking
  - now, at the very end, have the parameterization exported for Photoshop editing

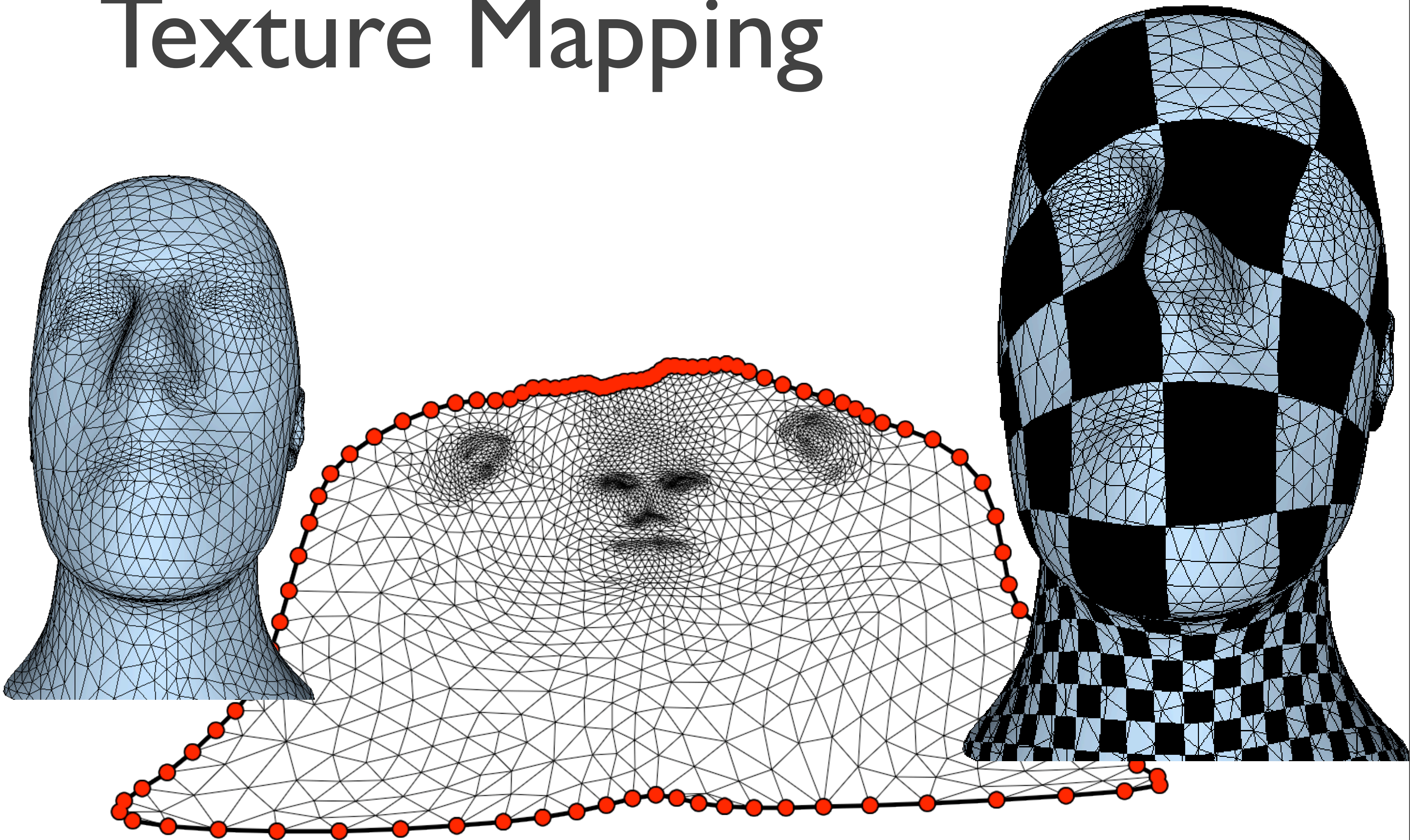


This is the parameterization after the constraint has moved along the red arrow.

- Bending energy matches the constraint smoothly & spreads out the distortion.
- Stretching matches constraint discontinuously, causing a fold-over, where multiple triangles map to the same piece of texture.
  - unusable for our purposes

- Bending energy can produce fold-overs, but it's much more robust than stretching.
- There are other parameterization approaches, but ones that guarantee no fold-overs are non-linear; bending energy is simple and fast, and interactivity is key.

Texture Mapping



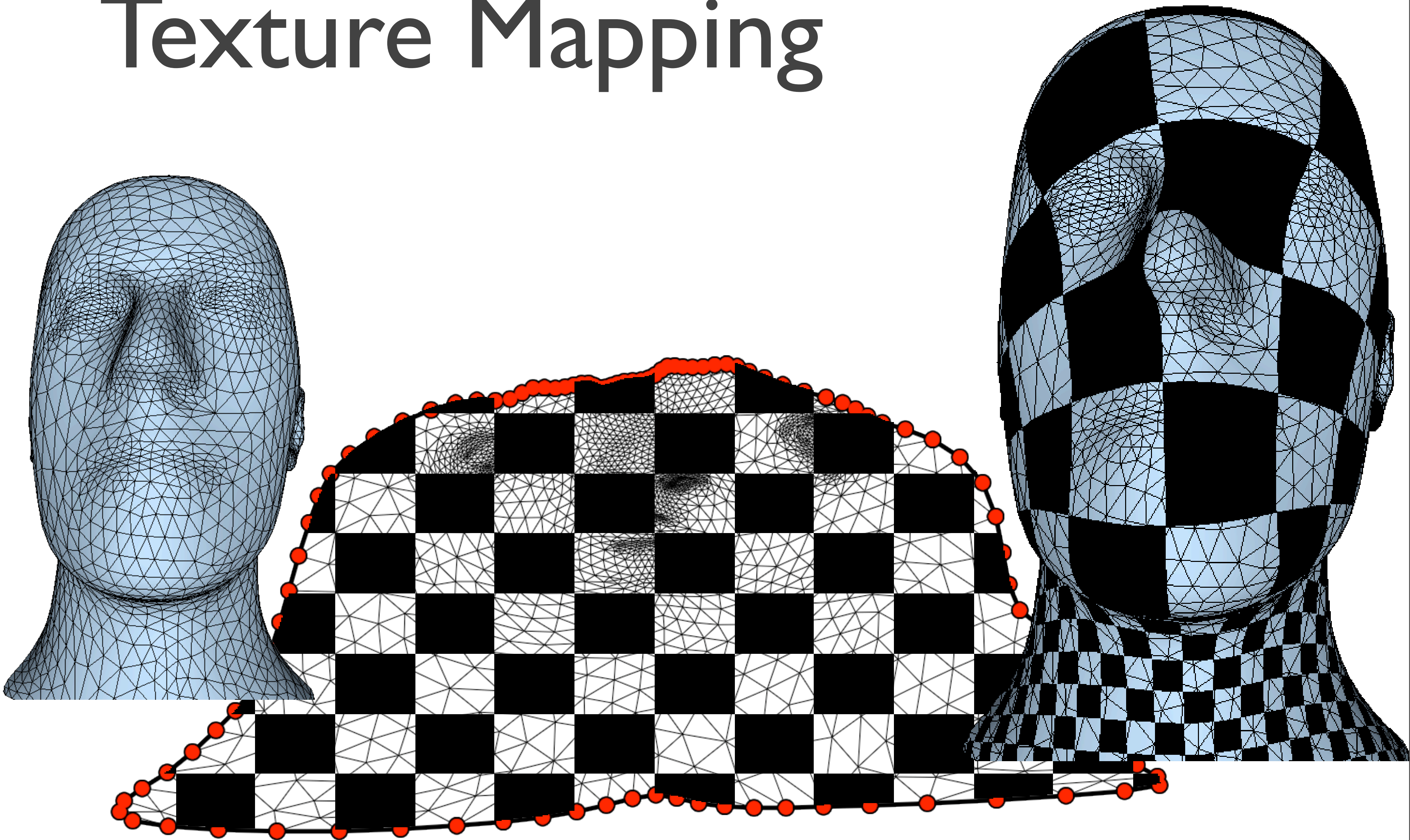
TODO: close-up of pink rectangles

- This is the unwrapped face, the parameterization.
- reminder: in traditional approach, this is what artists paint into
- Stretching is the traditional parameterization approach, like stretching a soap film over a boundary.
    - doesn't handle constraints, internal boundaries, well
    - localize the deformation too much and cause fold-overs, where multiple triangles map to the same piece of texture.
    - unusable for our purposes
- Bending energy does what we want, and distributes the deformation over a greater area.
    - matches constraint exactly and smoothly
        - [Levy 2001] had a trade-off between exactness and smoothness
    - can still cause fold-overs, but much, much less often

- There are other parameterization approaches, but bending energy is simple and fast, and interactivity is key.
    - ones that guarantee no fold-overs are non-linear

Intrinsic Parameterization = some linear combination of dirichlet (area) energy + euler characteristic (curvature)

Texture Mapping



TODO: close-up of pink rectangles

- This is the unwrapped face, the parameterization.
- reminder: in traditional approach, this is what artists paint into
- Stretching is the traditional parameterization approach, like stretching a soap film over a boundary.
    - doesn't handle constraints, internal boundaries, well
    - localize the deformation too much and cause fold-overs, where multiple triangles map to the same piece of texture.
    - unusable for our purposes
- Bending energy does what we want, and distributes the deformation over a greater area.
    - matches constraint exactly and smoothly
        - [Levy 2001] had a trade-off between exactness and smoothness
    - can still cause fold-overs, but much, much less often

- There are other parameterization approaches, but bending energy is simple and fast, and interactivity is key.
    - ones that guarantee no fold-overs are non-linear

Intrinsic Parameterization = some linear combination of dirichlet (area) energy + euler characteristic (curvature)

Related Work

UI

[Guiard 1987] - theoretical framework
[Balakrishnan and Kurtenbach 1999] - 3d camera and object
[Zeleznik et al. 1997] - 3d camera and object
[Hinckley et al. 1994] - 3d camera and object
[Kurtenbach et al. 1997] - 2 hand pan,zoom,rotate camera
[Balakrishnan and Hinckley 2000] - 2 hand alignment
Twister [Llamas et al. 2003] - 3d modeling
[Dietz and Leigh 2001] - other multitouch systems (DiamondTouch)
[Rekimoto 2002] - other multitouch systems (SmartSkin)
[Wilson 2004] - other multitouch systems (TouchLight)
[Han 2005] - other multitouch systems (FTIR)
[Wu and Balakrishnan 2003] - 2 finger rotation and scaling
[Igarashi et al. 2005] - ARAP
[Hanrahan and Haeberli 1990] - 3D painting
[Agrawala et al. 1995] - 3D painting with a tracker
[Carr and Hart 2004] - increasing texture resolution when painting
[Igarashi and Cosgrove 2001] - adaptive parameterization when painting
[Schmidt et al. 2006] - placing textures on the mesh
[Igarashi and Hughes 2002] - clothing manipulation

Parameterization

[Maillot et al. 1993] - elasticity deformation
[Piponi and Borshukov 2000] - stretching techniques
[Sander et al. 2001] - stretching techniques
[Yoshizawa et al. 2004] - stretching techniques
[Yoshizawa et al. 2005] - elasticity with 2 linear solves
[Floater 1997] - solve 1 linear system
[Desbrun et al. 2002] - geometric weights for floater
[Lévy et al. 2002] - geometric weights for floater
[Sheffer and de Sturler 2001] - angle distortion
[Sheffer et al. 2005] - angle distortion (more efficient)
Zayer et al. [45] - boundary free by solving several linear systems
[Lee et al. 2005] - based on tracing geodesics
[Lévy 2001] - allows constraints
Desbrun and Alliez [7] - lagrange multiplier constraints
[Kraevoy et al. 2003] - observations on meeting constraints
skipping consistently parameterizing several surfaces
skipping partitioning the surface into patches
[Yamauchi et al. 2005] - automated pipeling for partitioning the surface
[DeBry et al. 2002] - avoid parameterization
[Beier and Neely 1992] - image warping on a mesh
[Schaefer et al. 2006] - interactive image warping
[James and Pai 1999] - incremental matrix update for interactivity

[don't go into any of them, just put them on-screen and say it exists (UI, parameterization)]

TODO: cull this, or remove descriptions and enlarge, or break it up onto multiple slides