Data-driven Facial Animation

Computer Animation and Visualisation
Lecture 7

Taku Komura

Outline

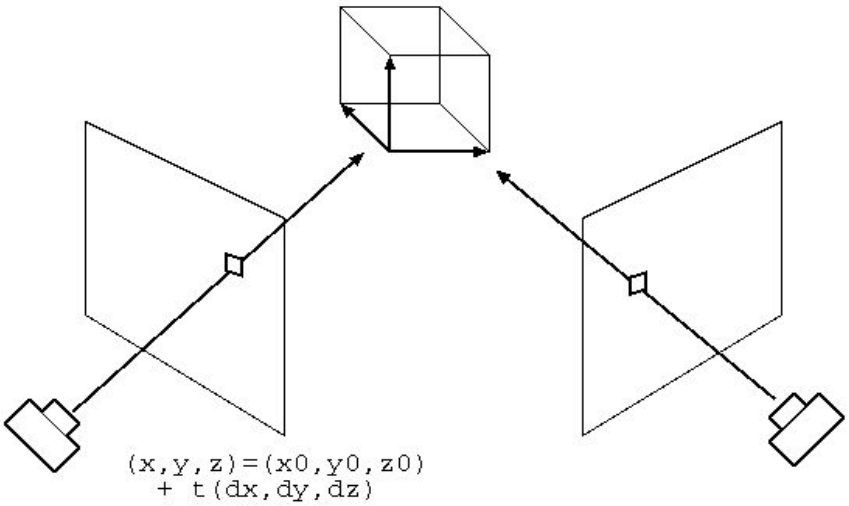
- Photometric Capturing
- Blendshapes
- Facial Rigging
- Digital Emily
  - Capturing
  - Reflectance model
  - Polarization for obtaining a reflectance model
  - Animation synthesis by an interface

Photometric capturing

- Nowadays, we can capture the surface details of the facial movements in real-time using multiple cameras
- Can capture the surface shape as well as the textures
- Sometimes project a texture pattern onto the face (Performance capture, KINECT)

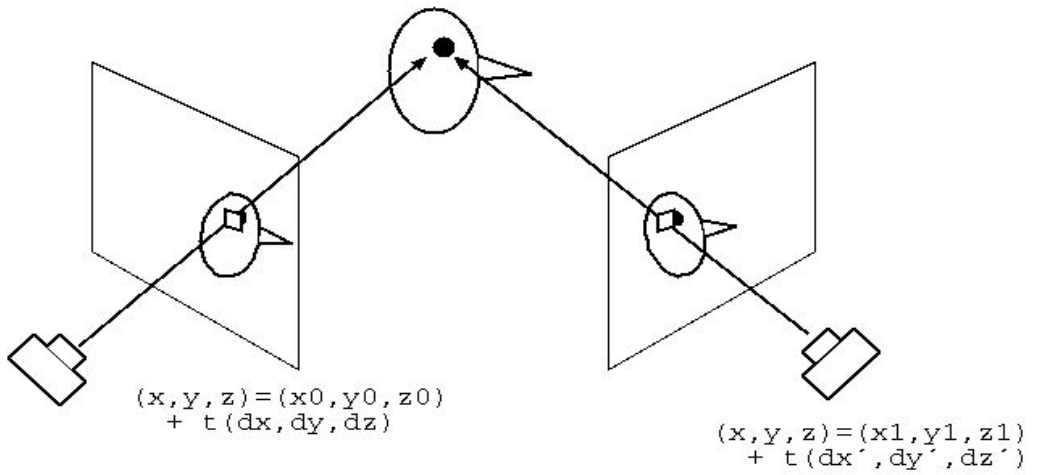
Stereo Vision (1)

- For each pixel in the left and right images, you can find out the equations of the rays that pass through them (by calibration)



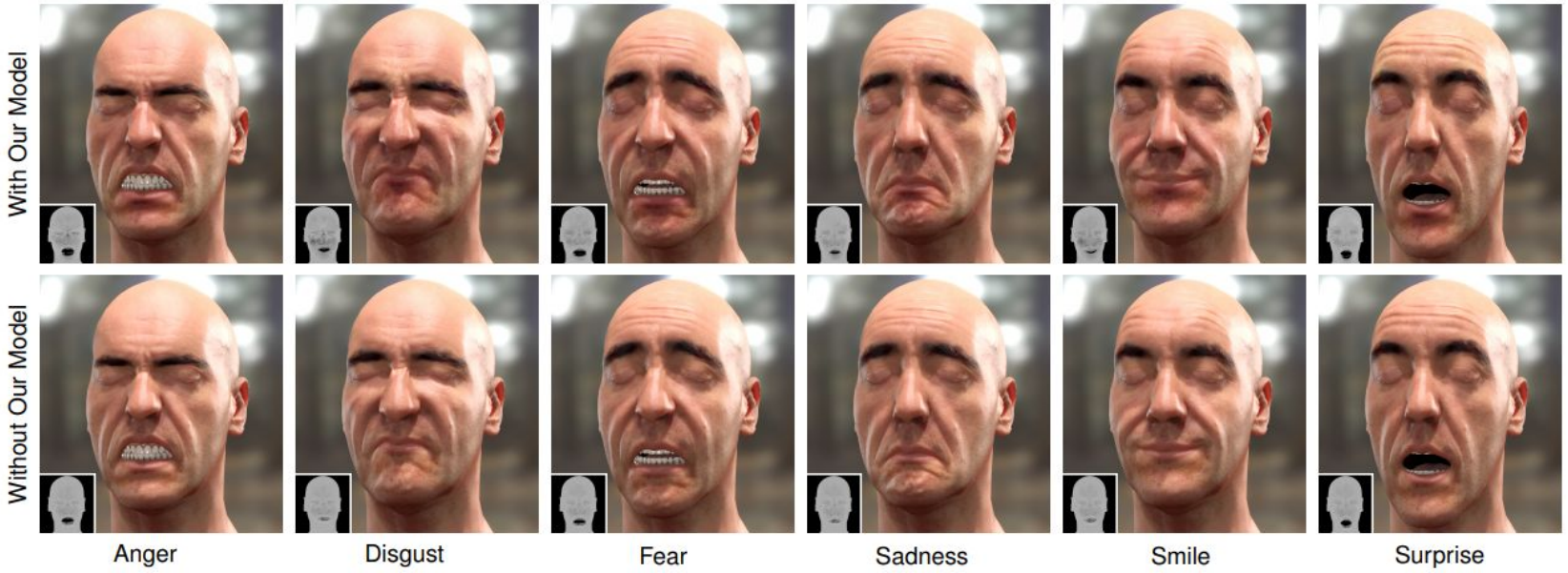
Stereo Vision (2)

- You find the pixels in the left and right images where the same point appears
- You can compute the 3D location by computing their intersection (or the closest point)



http://www.iryoku.com/skincolor/downloads/A-Practical-Appearance-Model-for-Dynamic-Facial-Color.mp4

Photorealistic Facial Animation



Jimenez et. al., ACM Transactions on Graphics, Vol. 29 (5), 2010

Paper:
http://www.iryoku.com/skincolor/downloads/A-Practical-Appearance-Model-for-Dynamic-Facial-Color.pdf

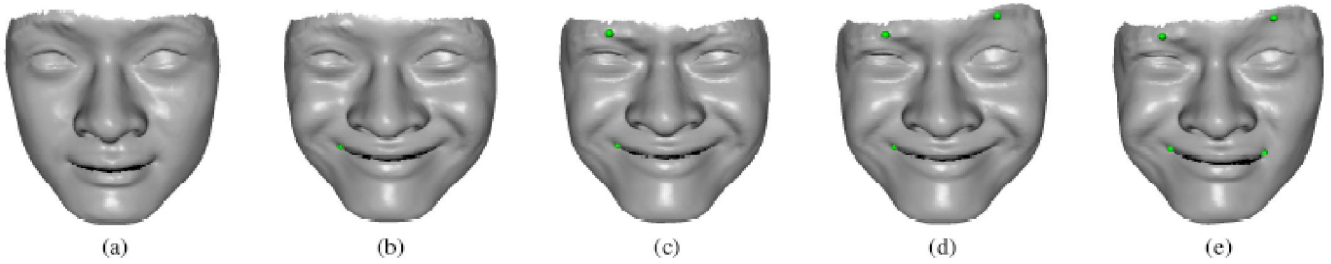
Video:
http://www.iryoku.com/skincolor/downloads/A-Practical-Appearance-Model-for-Dynamic-Facial-Color.mp4

Outline

- Photometric Capturing
- **Blendshapes**
- Facial Rigging
- Digital Emily
  - Capturing
  - Reflectance model
  - Polarization for obtaining a reflectance model
  - Animation synthesis by an interface

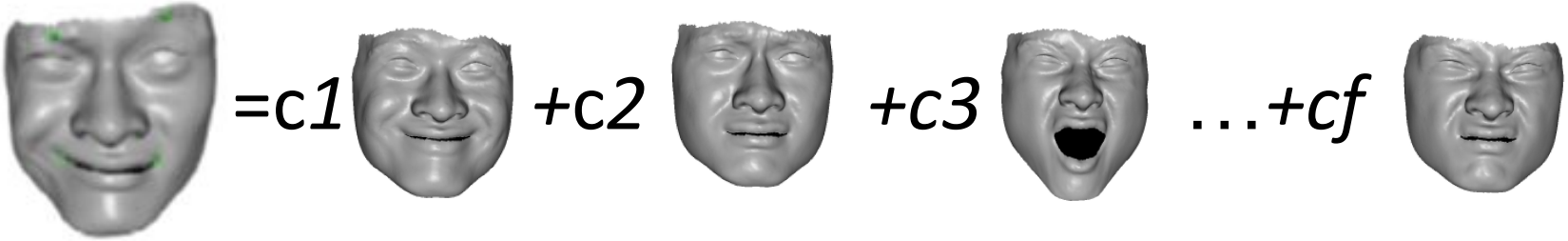
Blendshapes

- Using photometric stereo, we can capture realistic 3D shapes with textures
- Once a generic face mesh is fitted to the data, we can interpolate these shapes to synthesize arbitrary facial expressions

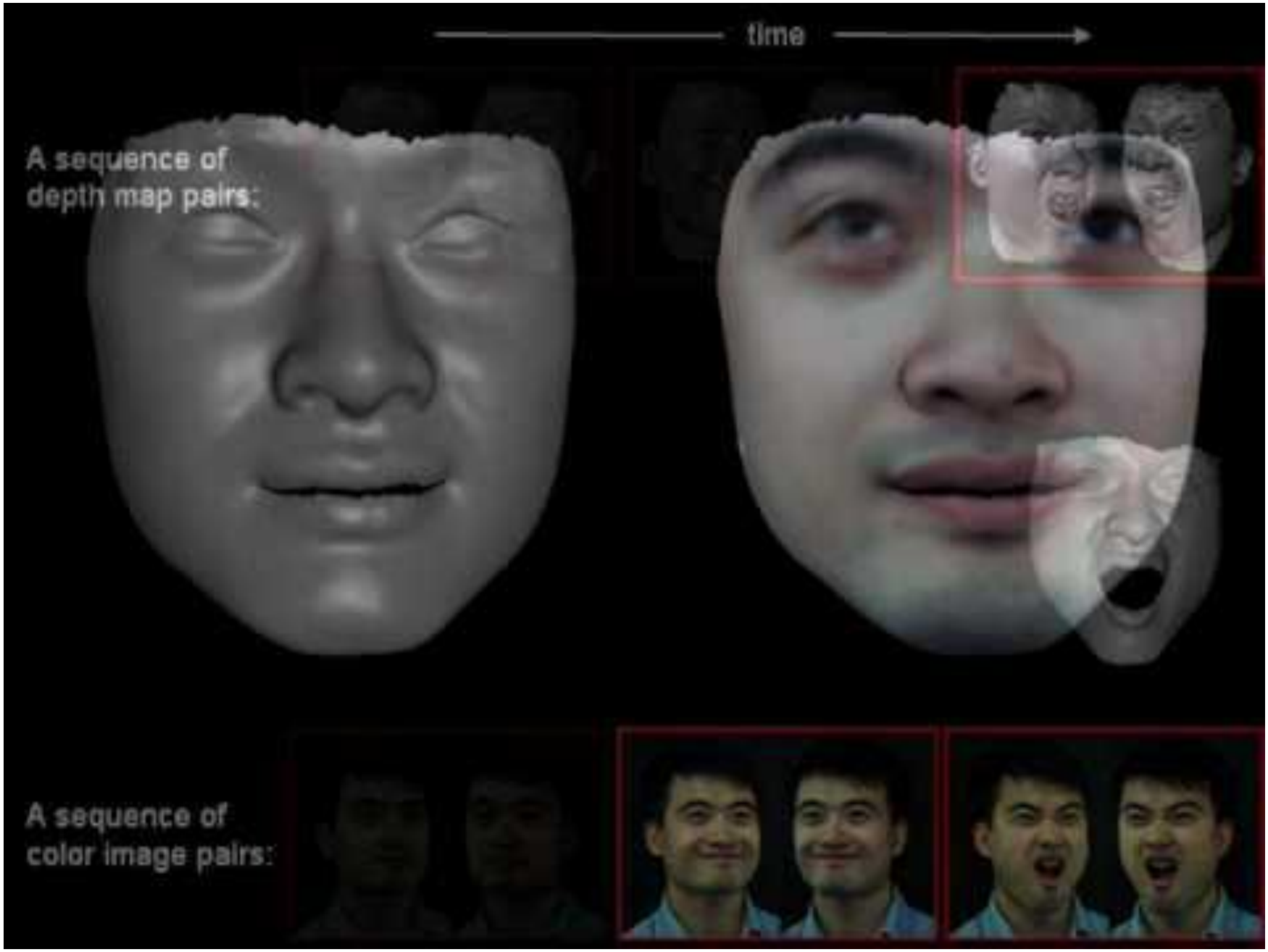


Blendshapes

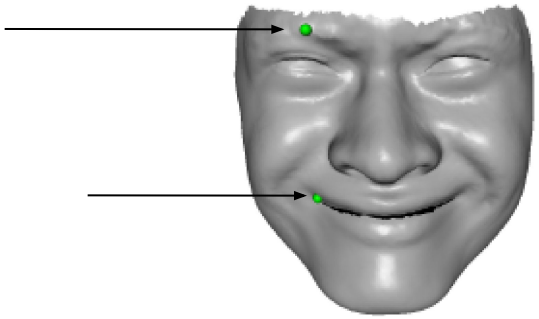
- Assume we have *F* meshes of the face
- We are going to blend these *F* meshes to produce a new face mesh
- Blend coefficients c_i



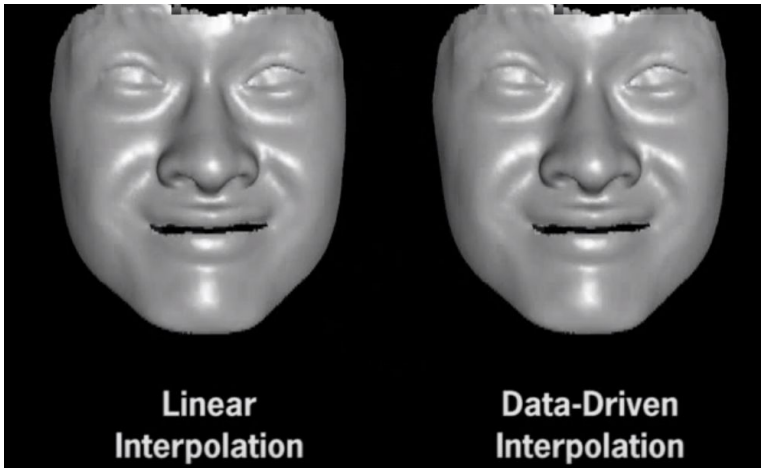
$$\sum_{f=1}^{F} c_f = 1$$


time
A sequence of
depth map pairs:
A sequence of
color image pairs:

Face IK

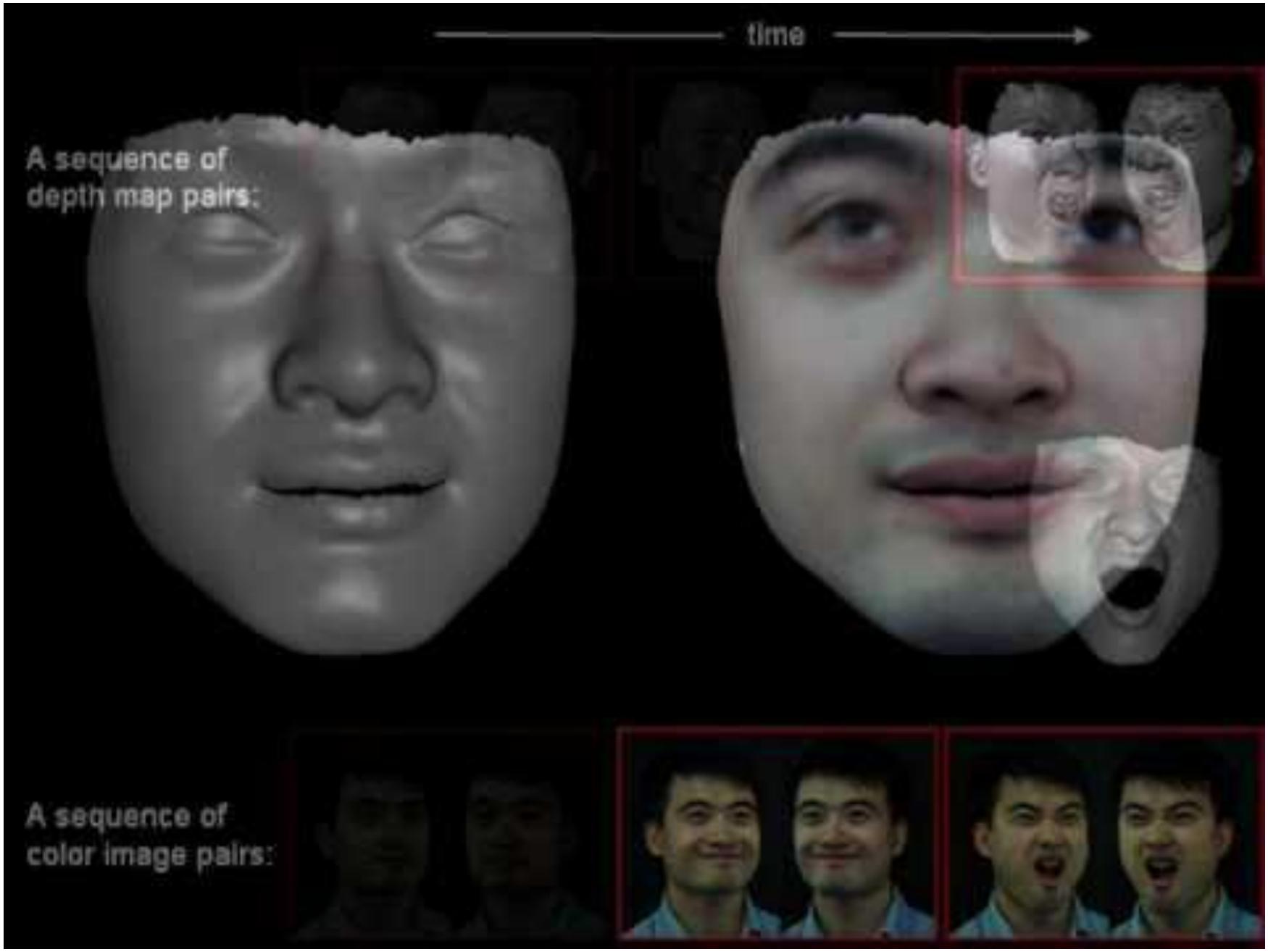


- The animators will prefer to control some points on the face in an IK fashion rather than the weights

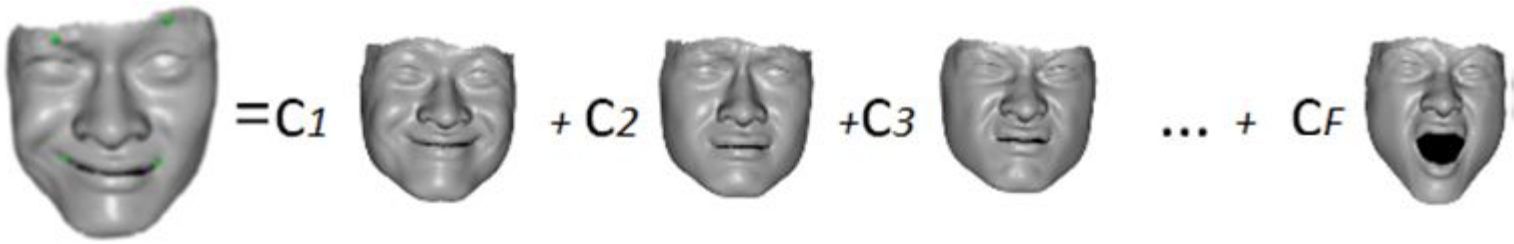


https://www.youtube.com/watch?v=6lSrgPeOZsA

Around 3:45


time
A sequence of
depth map pairs:
A sequence of
color image pairs:

Face IK

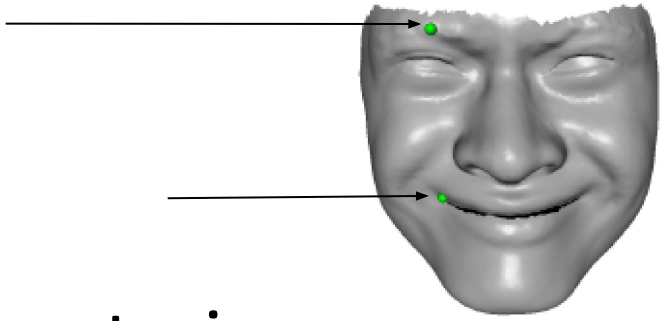


- **S***n,f* : n-th vertex in mesh f
- **S***n* : n-th vertex in the final face

$$\mathbf{S}n = c_1 \mathbf{S}_{n,1} + c_2 \mathbf{S}_{n,2} + c_3 \mathbf{S}_{n,3} + \ldots + c_F \mathbf{S}_{n,F}$$

$$\sum_{f=1}^{F} c_f = 1$$

Face IK - 2



- We want to constrain some vertices (controlled by the animator)
- {**p**l}l=1,L : user specified constraints, vertex *l* should be at position **p**l

$$\sum_{f=1}^{F} c_f \mathbf{s}_{l,f} = \mathbf{p}_l \quad \text{and} \quad \sum_{f=1}^{F} c_f = 1$$

How to solve this?

- $\min \frac{1}{2}\mathbf{x}^{\mathbf{T}}\mathbf{A}\mathbf{x}$ s.t. $\mathbf{J}\mathbf{x} = \mathbf{d}$

Equaivalent to solve $\min \frac{1}{2}\mathbf{x}^{\mathbf{T}}\mathbf{A}\mathbf{x} + \boldsymbol{\lambda}(\mathbf{J}\mathbf{x} - \mathbf{d})$

Lagrange multipliers

Can be solved by $\begin{bmatrix} \mathbf{A} & \mathbf{J}^{\mathbf{T}} \\ \mathbf{J} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{d} \end{bmatrix}$

→ J is the matrix for the linear constraints
→ A is the positive definite matrix of the objective function
→ d are the constants of linear constraints

Problem 1: Hard constraints are not always possible to satisfy

Resulting in unstable results

Face IK – Solution

Using soft constraints to allow slight violation of the constraints

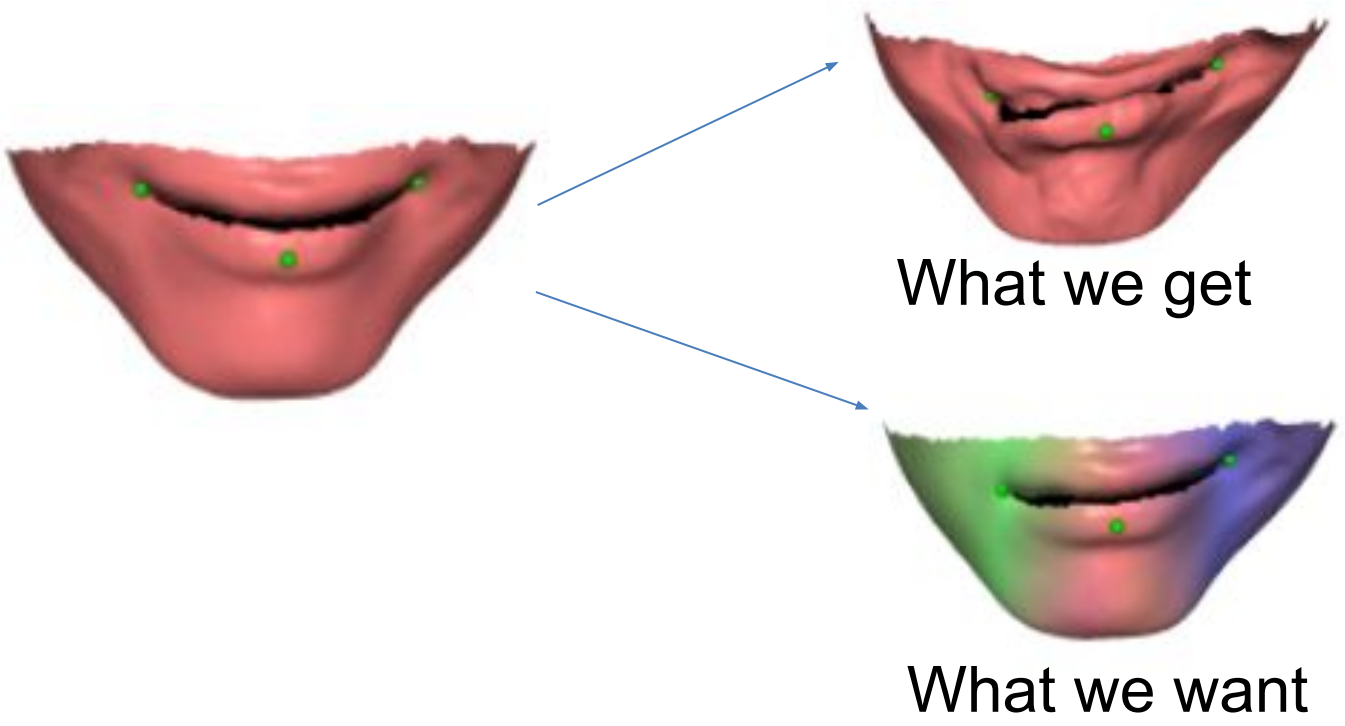
$$c_f = \arg\min_{c_f} \sum_{f=1}^{F} (c_f \mathbf{s}_{l,f} - \mathbf{p}_l)^2 \qquad \text{s.t} \qquad \sum_{f=1}^{F} c_f = 1$$

- Instead of hard constraints

$$\sum_{f=1}^{F} c_f \mathbf{s}_{l,f} = \mathbf{p}_l$$

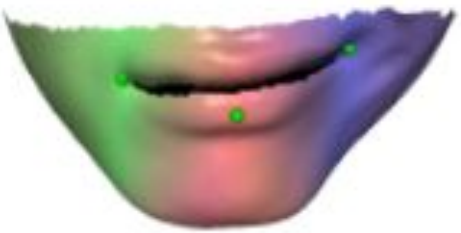
Problem 2

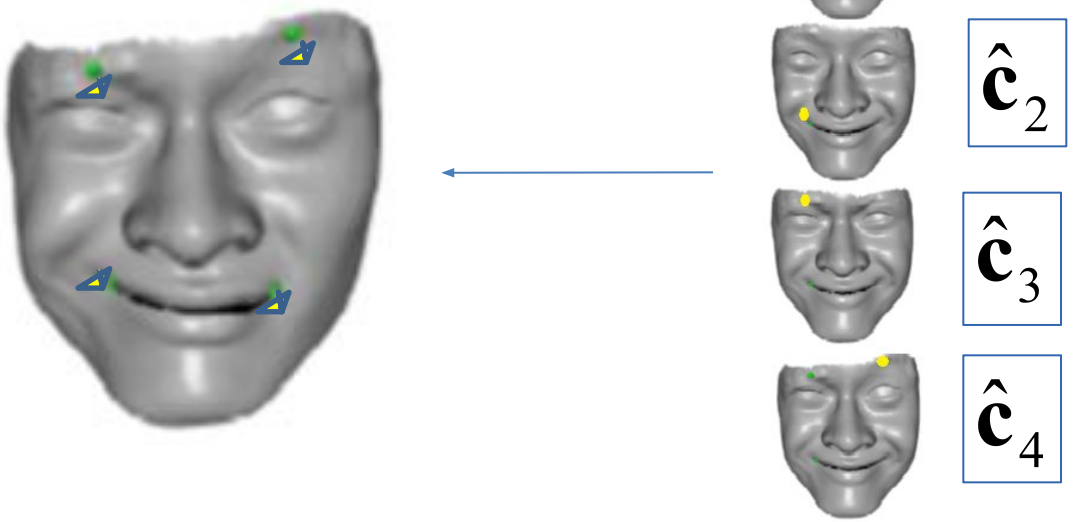
- Just linearly combining the captured motions cannot create enough variations
- For example, cannot produce asymmetric expressions from symmetric expressions
  - Local control is needed



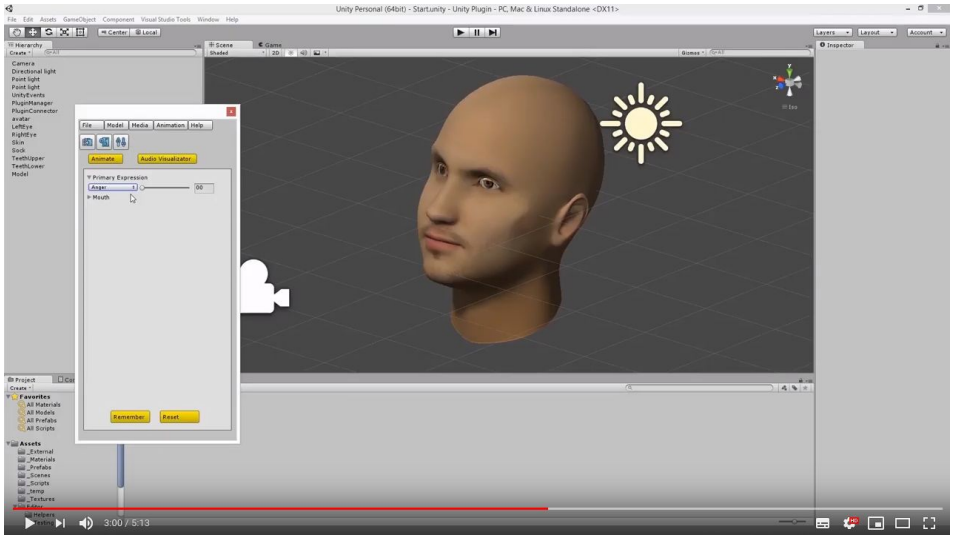
Local Control

- Divide the face into different regions
- Solve the optimization within the region for each constraints
- Blend each region to the entire face

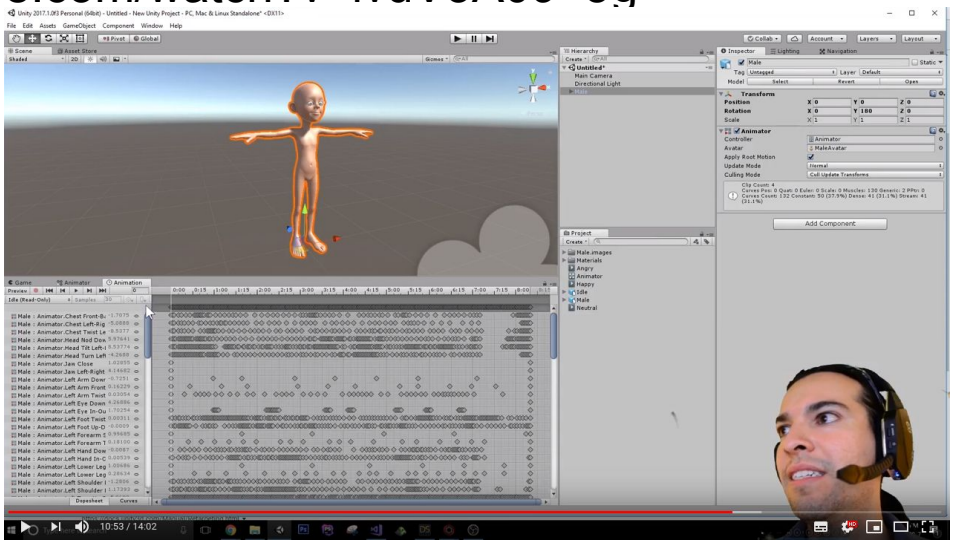




Face Animation in Unity



https://www.youtube.com/watch?v=lvuV3A09_5g



https://www.youtube.com/watch?v=pqf4wxgl938

Outline

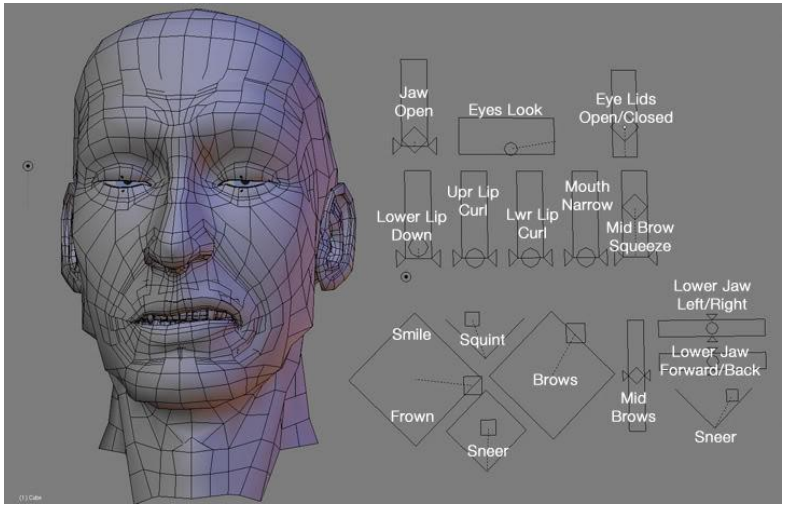
- Photometric Capturing
- Blendshapes
- **Facial Rigging**
- Digital Emily
  - Capturing
  - Reflectance model
  - Polarization for obtaining a reflectance model
  - Animation synthesis by an interface

Facial Rig

– A parameter space and user interface for editing the character face

–Animators produce facial animation by keyframe animation

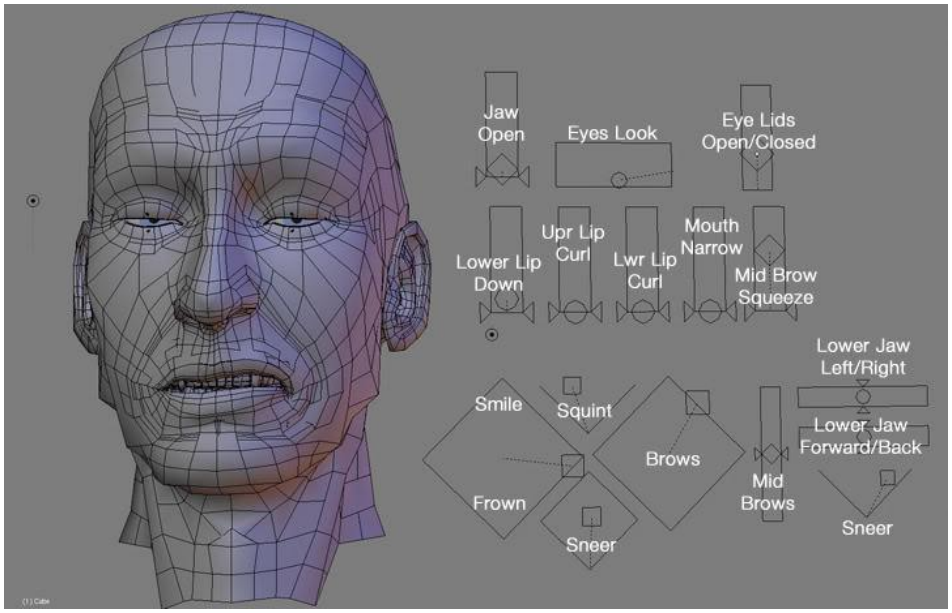
–Keyframes are designed by the facial rig

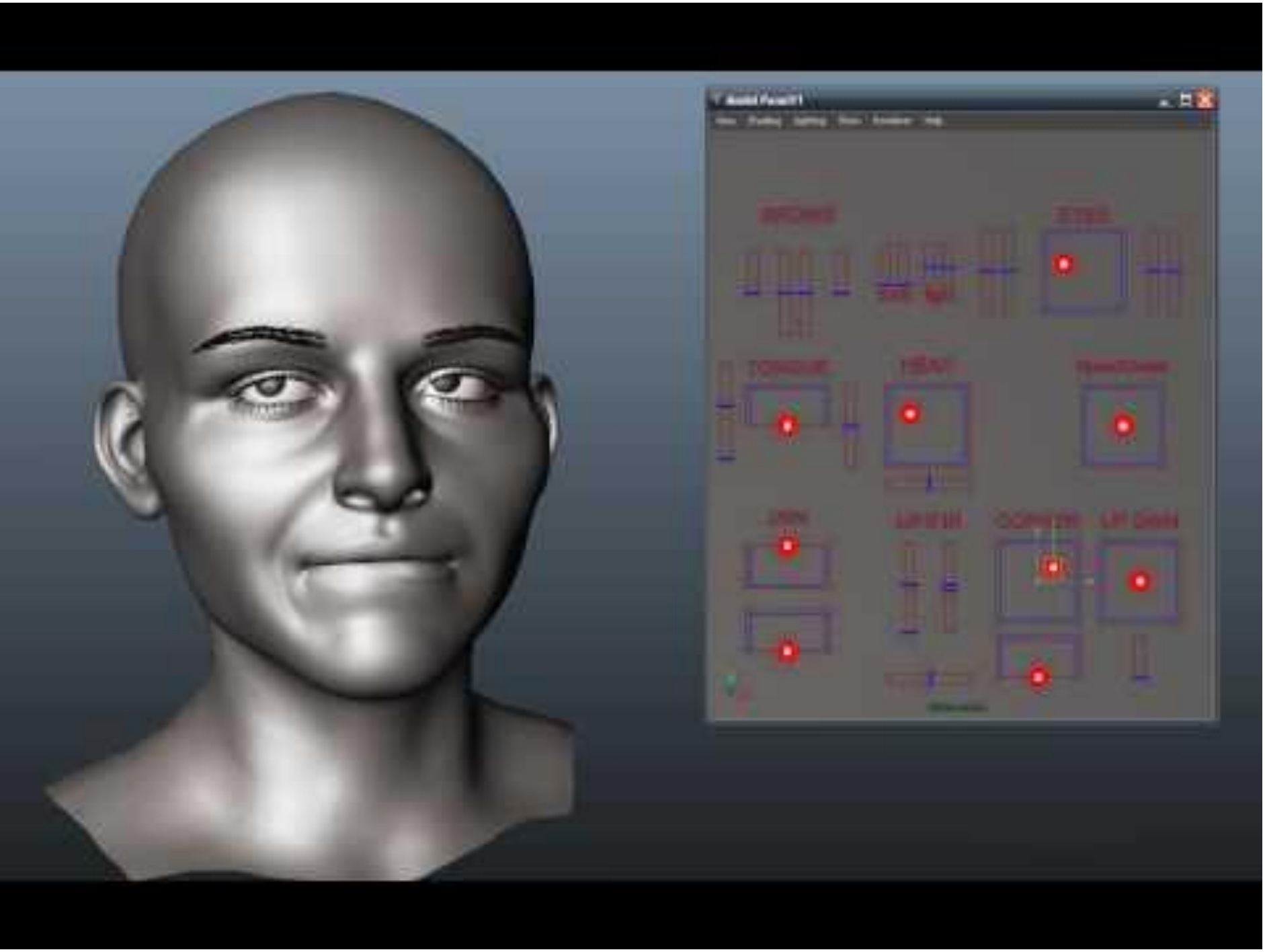


https://www.youtube.com/watch?v=KPDfMpuK2fQ

Facial Rig (2)

– The rig parameters can be FK parameters, blendshape weights, muscle contraction etc.





Outline

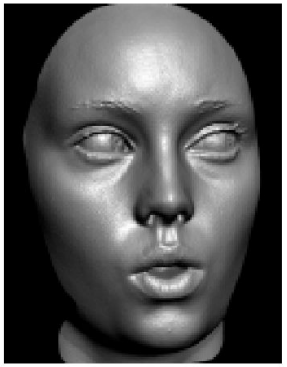
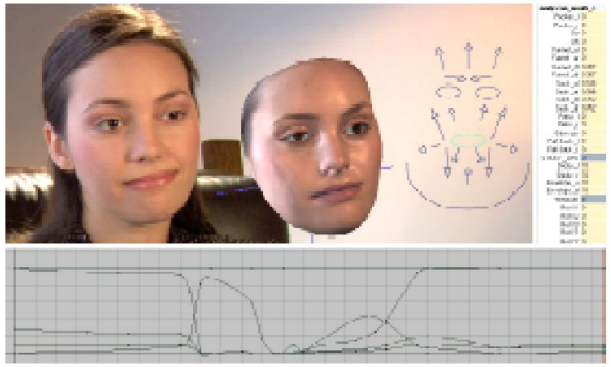
- Photometric Capturing
- Blendshapes
- Facial Rigging
- **Digital Emily**
  - Capturing
  - Reflectance model
  - Polarization for obtaining a reflectance model
  - Animation synthesis by an interface

Digital Emily Project

By USC ICT and Image Metrics 2008

- Synthesizing realistic 3D animation of a human
- Reflectance parameters of the skin are obtained
- 3D shapes reconstructed
- Manually designing the keyframes

http://www.youtube.com/watch?v=GBgURIUQ700

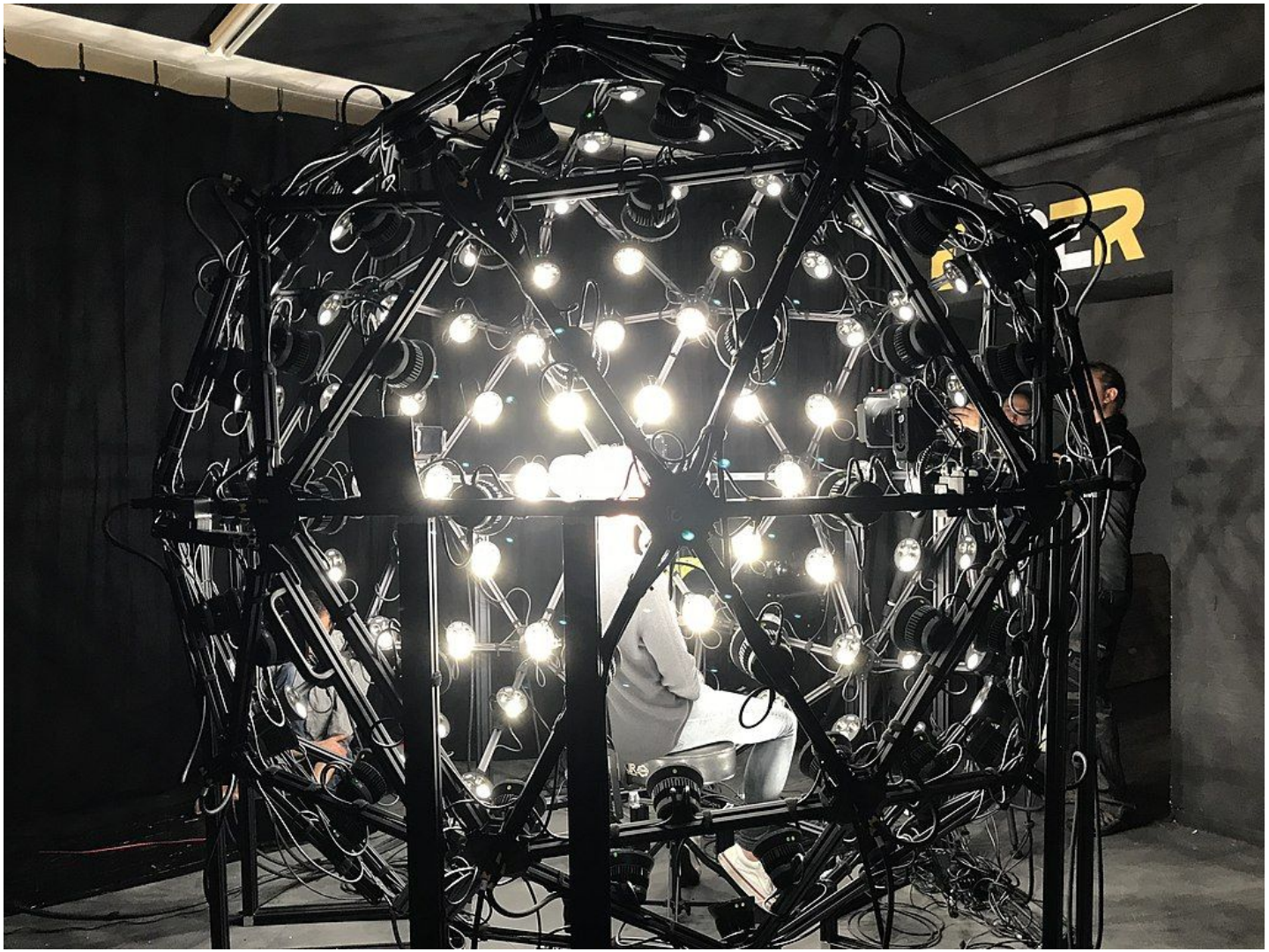


diffuse shading

Digital Emily Project : steps

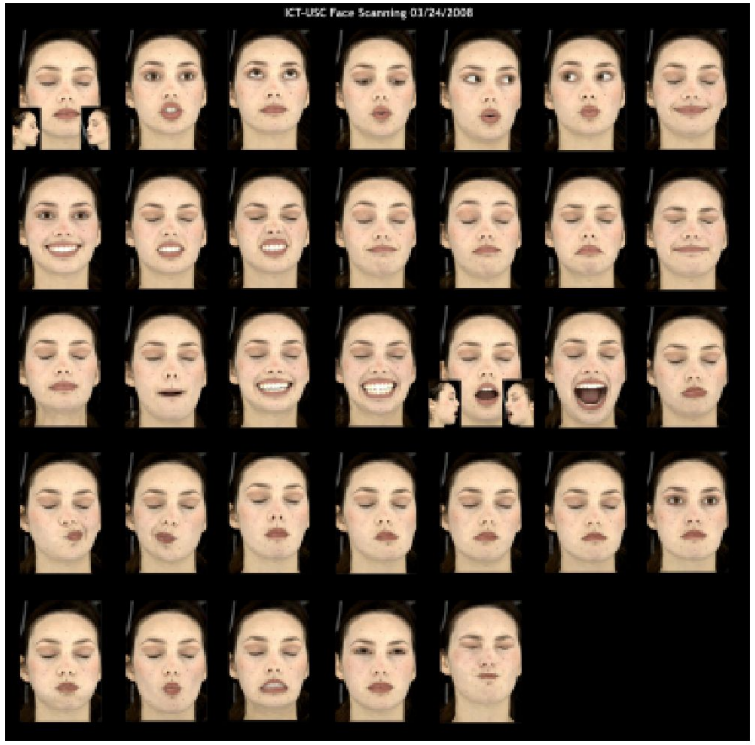
1. Taking images of the face under different lighting conditions
2. Creating reflectance model of the face
   → polarization
3. Reconstructing the 3D geometry
   → stereo vision, photometric stereo
4. Producing the facial rig
5. Creating keyframe animation

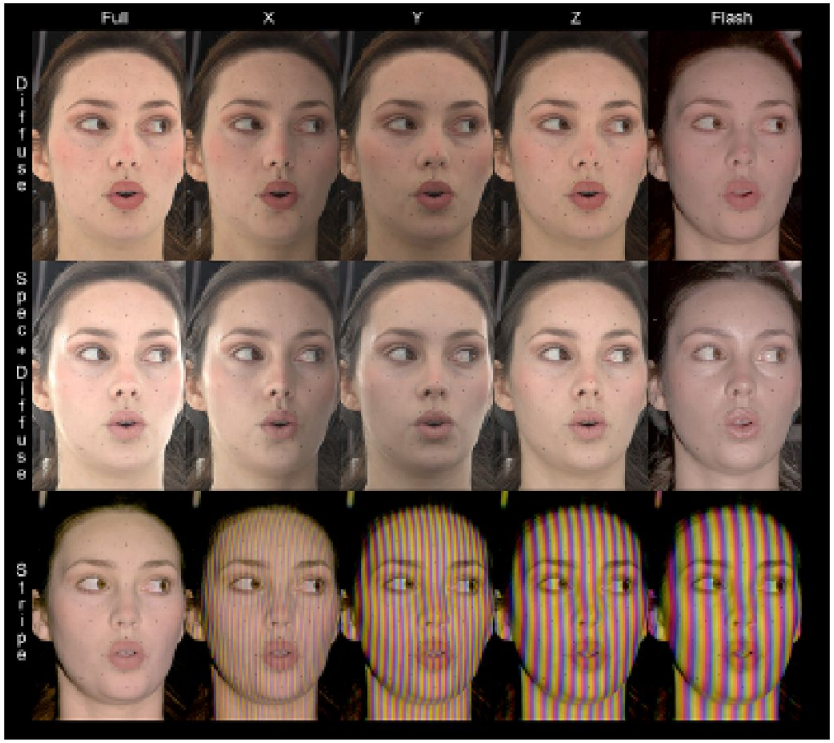
Recording the faces



Recording the faces

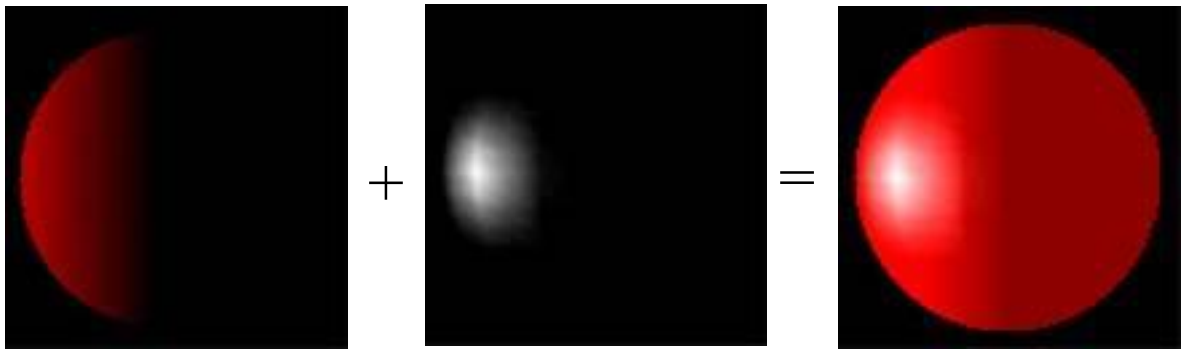
- Shooting 33 different expressions captured
- 15 shots taken for each expression



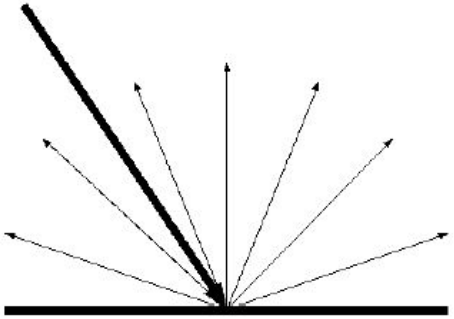


Reflectance model

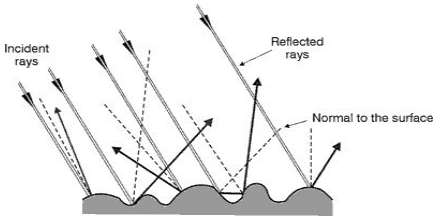
- Realistic reflection can be simulated by combining diffuse and specular reflection



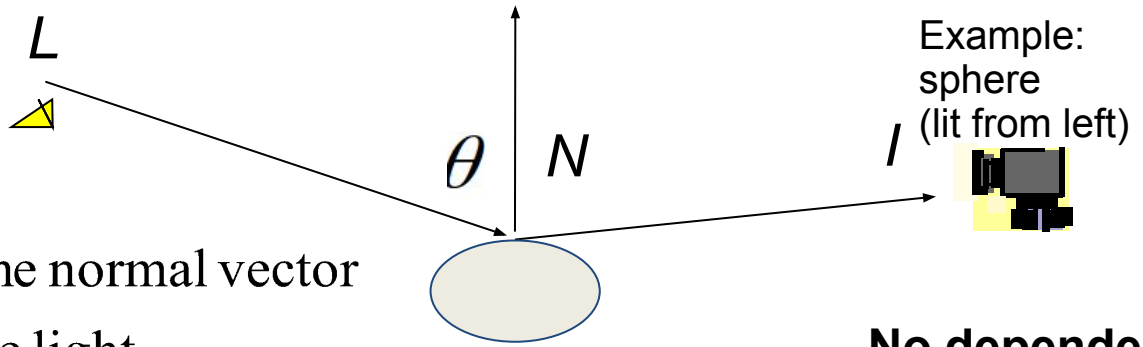
Diffuse Reflection



- Light reflected to all directions
  - considers the angle of incidence of light on surface
  - The light is subscattered under the skin, before emerging at the surface again
  - Reflected to a random direction



$$I = I_p k_d \cos\theta$$



Example:
sphere
(lit from left)

I_p : Light Intensity

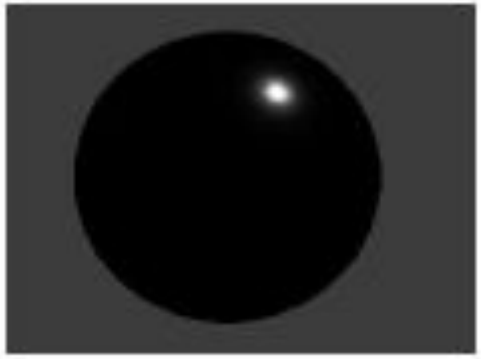
θ : the angle between the normal vector direction towards the light

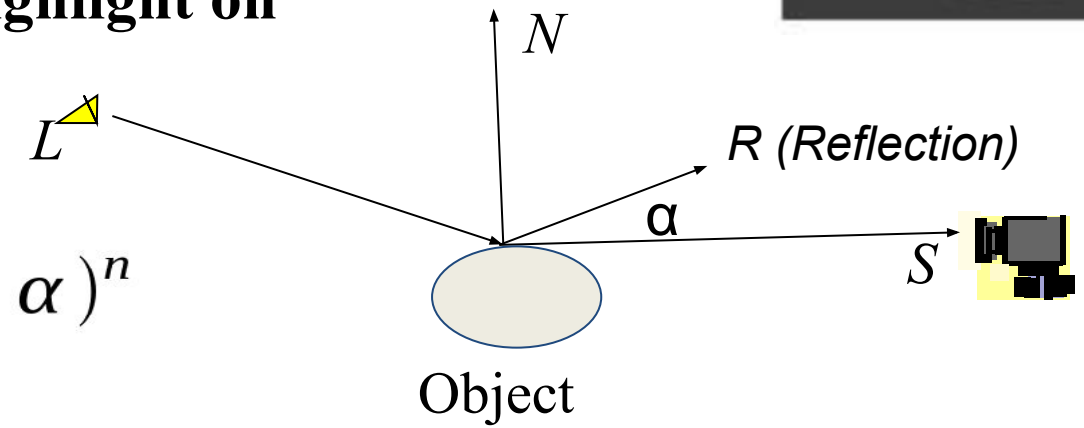
k_d : diffuse reflectivity

No dependence on camera angle!

Specular Reflection

- Direct reflections of light source off shiny object
  - specular intensity n = shiny reflectance of object
  - Result: **specular highlight on object**





$$I = I_p k_s (\cos \alpha)^n$$

I_p: light intensity
I : output color
R: reflection vector
S : direction towards the camera
α : angle between R and S

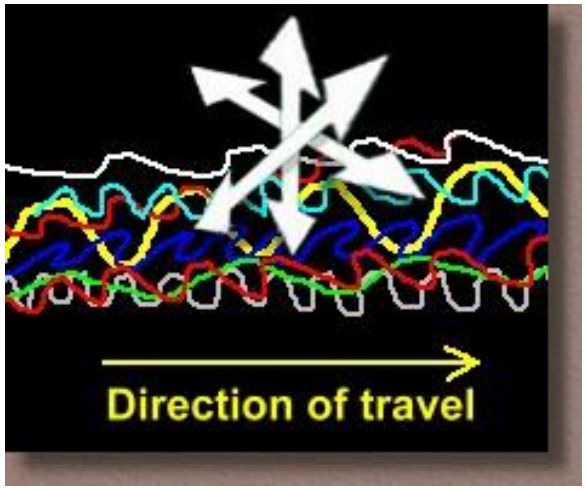
No dependence on object colour.

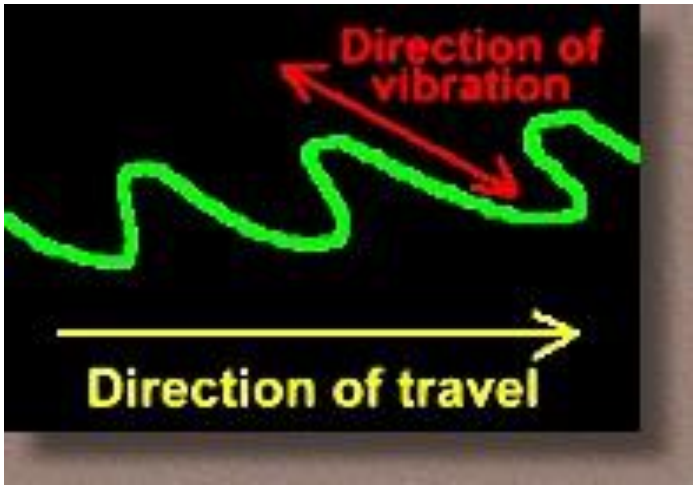
Creating a reflectance model of the face

- Simple texture mapping cannot make the face appear realistic
- Need to produce a precise face reflectance model
  - how much light is specularly reflected, and how much is diffusely reflected over the face
    - Need to compute the specular/diffuse reflectance parameters
- This is done by polarization

Light Polarization

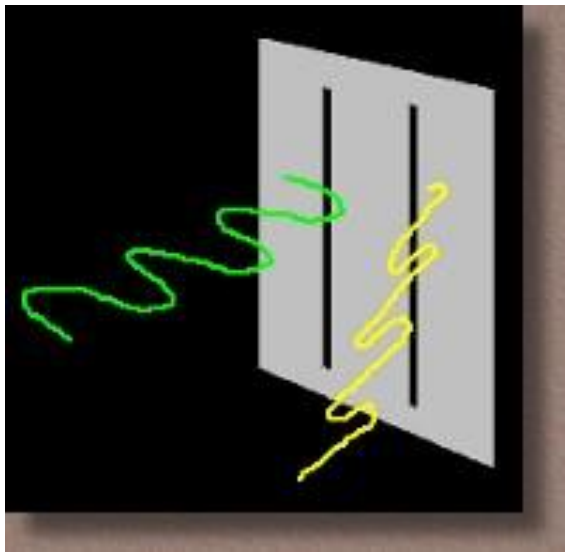
- Lights can be considered as waves vibrating in all directions.
- Polarization: Making them vibrate in one direction

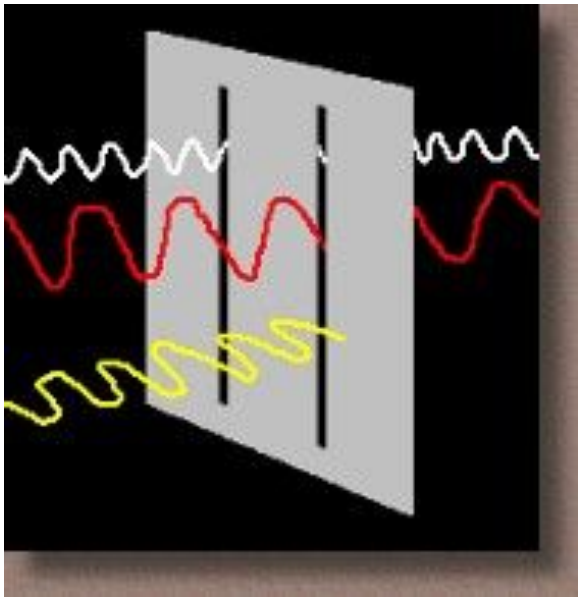




How to polarize light?

- Pass them through a barrier that contains many very narrow slits
- Those light only parallel to the slit pass through
- The rest are blocked



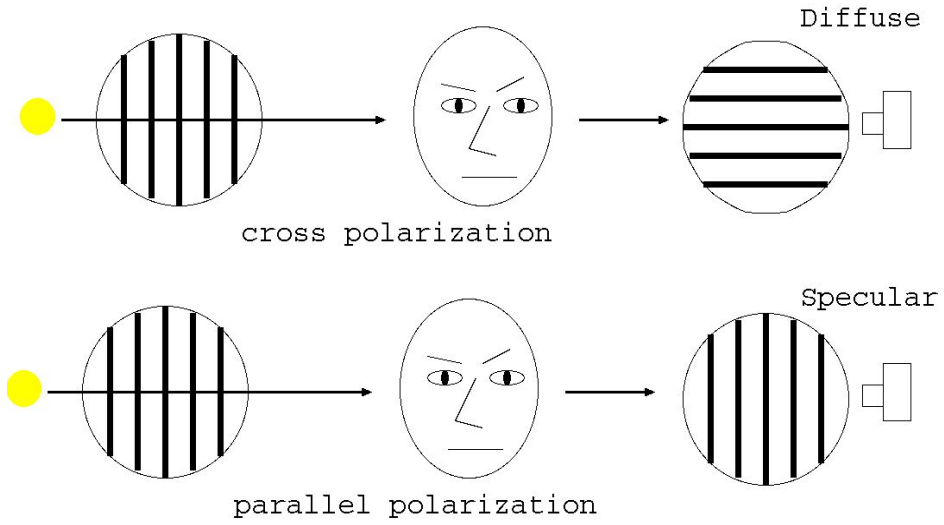


Reflection and Polarization

- When the light reflects over the surface, the direction of the light rotates
- The light is still polarized in almost the same direction after specular reflection
- Diffuse reflection depolarize light waves to all directions
- By only passing the polarized light through the camera lenses, we can know how much light is specularly reflected

Polarizing the light

- Every casted light is filtered by a polarizer
- The light coming into the camera are also filtered, either by cross-polarizer or parallel polarizer
- Cross-polarizer passes part of the diffusely reflected light
- Parallel polarizer passes specularly reflected light plus part of the diffusely reflected light



D

D+S

S

- (top) cross polarized image (middle) parallel polarized image (bottom) Subtracting

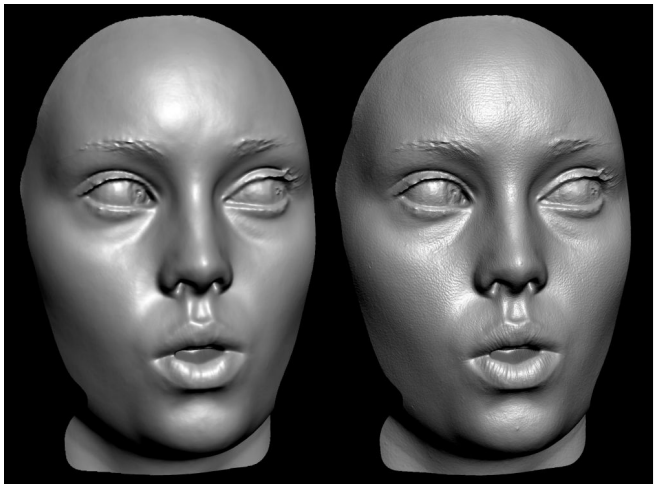
$$I = I_p k_d \cos\theta$$

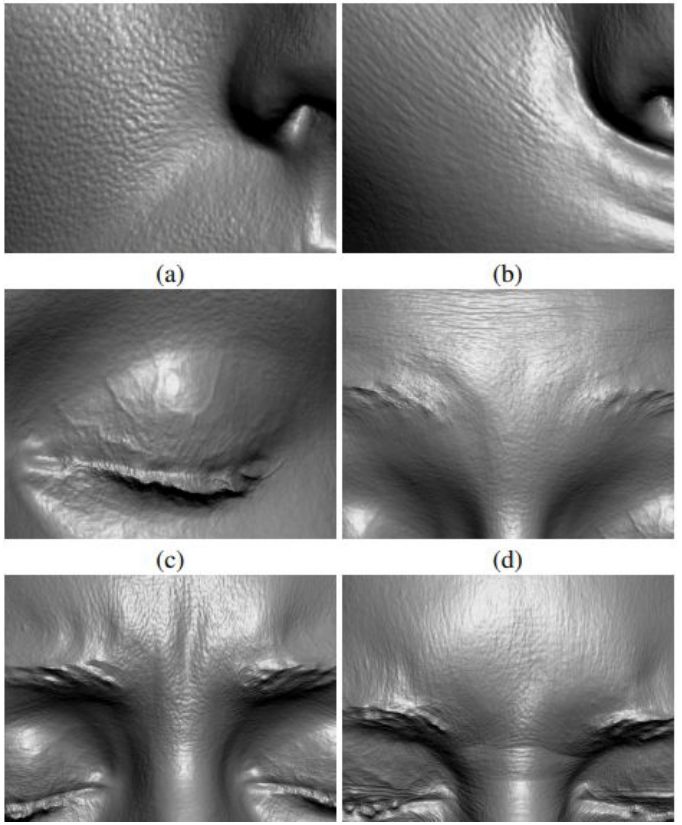
$$I = I_p k_s (\cos\ \alpha)^n$$

As we already know all the other values (geometry, camera direction, light direction), we can compute the reflection parameters for each pixel region from the shading equations

Reconstructing the 3D Geometry

- The geometry is reconstructed in two steps
- First the rough geometry is reconstructed with stereo vision
- The normal vectors are computed by photometric stereo and added as the details



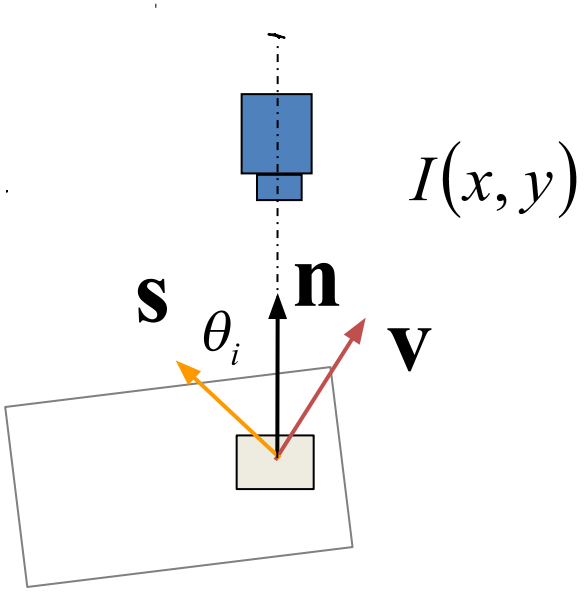


Photometric Stereo

- Estimating the surface normals of an object by observing it under different lighting conditions
- Relates pixel brightness *I(x,y)* to surface orientation for given source direction and surface reflectance
- A simple diffuse case with point light:

I_p : source brightness

k_d : surface albedo (reflectance)

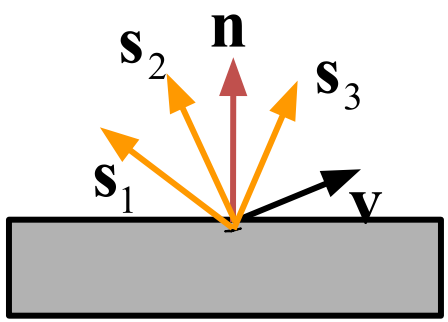


Pixel brightness

$$I = I_p k_d \cos\theta = I_p k_d \mathbf{n} \cdot \mathbf{s}$$

So $I \propto \mathbf{n} \cdot \mathbf{s}$

Photometric Stereo



Three light source:

$$I_1 = \rho \mathbf{n} \cdot \mathbf{s}_1$$
$$I_2 = \rho \mathbf{n} \cdot \mathbf{s}_2$$
$$I_3 = \rho \mathbf{n} \cdot \mathbf{s}_3$$

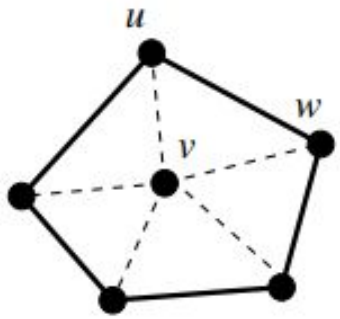
- We can write this in matrix form:

$$\begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix} = \rho \begin{pmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \mathbf{s}_3^T \end{pmatrix} \mathbf{n}$$

- Solve for **n (computing the normal vector)**

Combining the Normal Vectors and the Geometry

- The normal maps computed by specular reflection contain lots of fine details
- We wish to adjust the geometry such that the normals become close to the normal maps.
- Optimizing the position of the vertices such that the computed normals are close to those of the normal maps



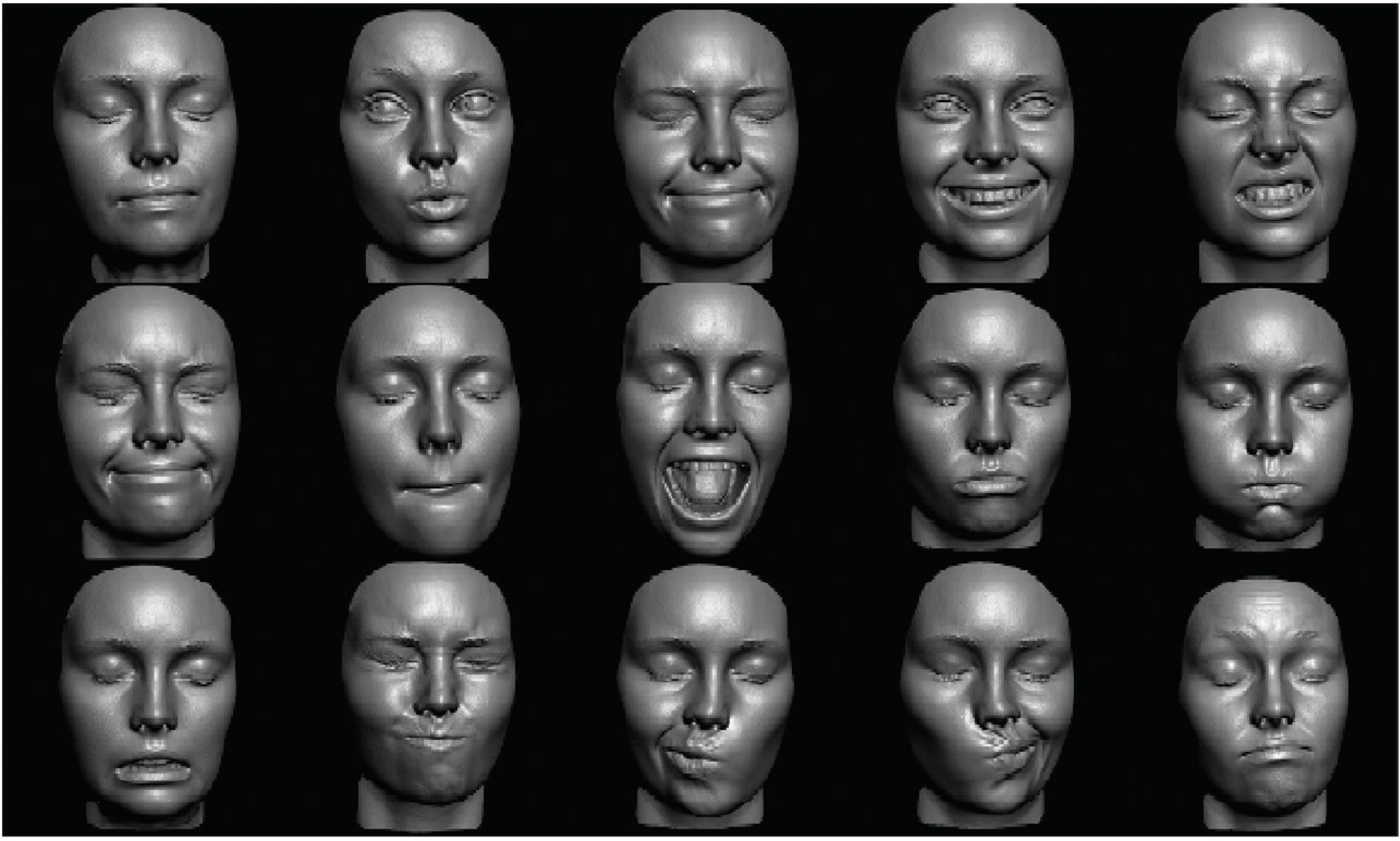
$$\hat{E}^n = \sum_v \sum_{u,w} [N_v^m \cdot (P_u - P_w)]^2$$

$$\hat{E}^p = \sum_v ||M_v(P_v - P_v^m)||^2$$

$$\arg\min_P \lambda \hat{E}^p + (1-\lambda)\hat{E}^n$$

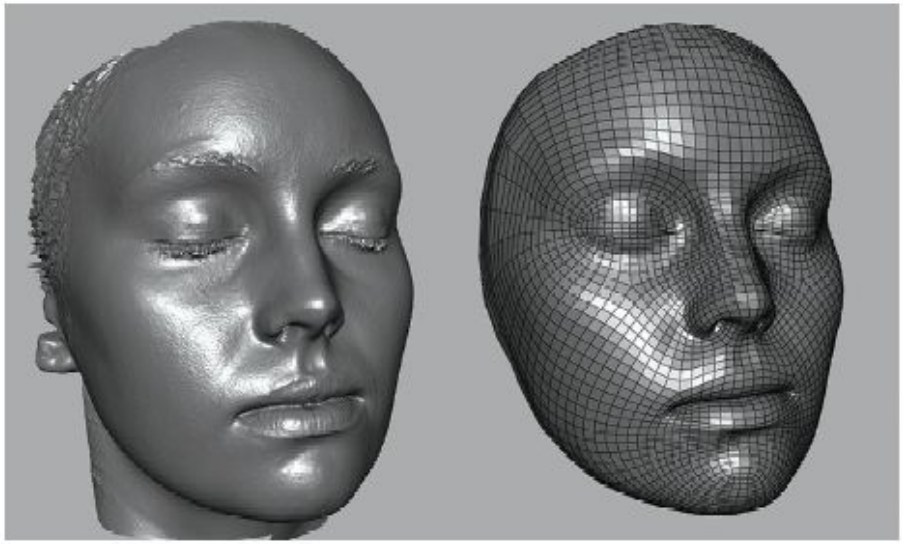
Nehab et al. 2005

Reconstructed 3D shapes



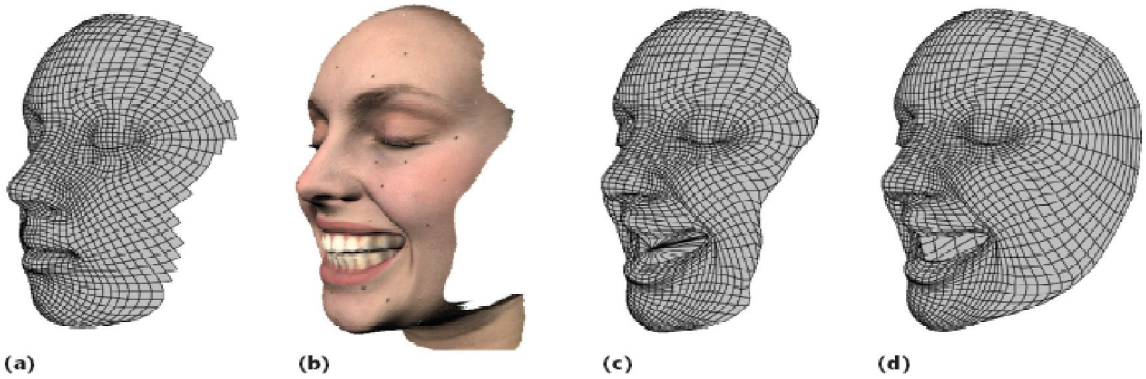
Fitting a Generic Face Mesh

- A face mesh model of 4000 polygons is fitted to the data
- The high resolution details are saved as displacement maps and added later when synthesizing the animation



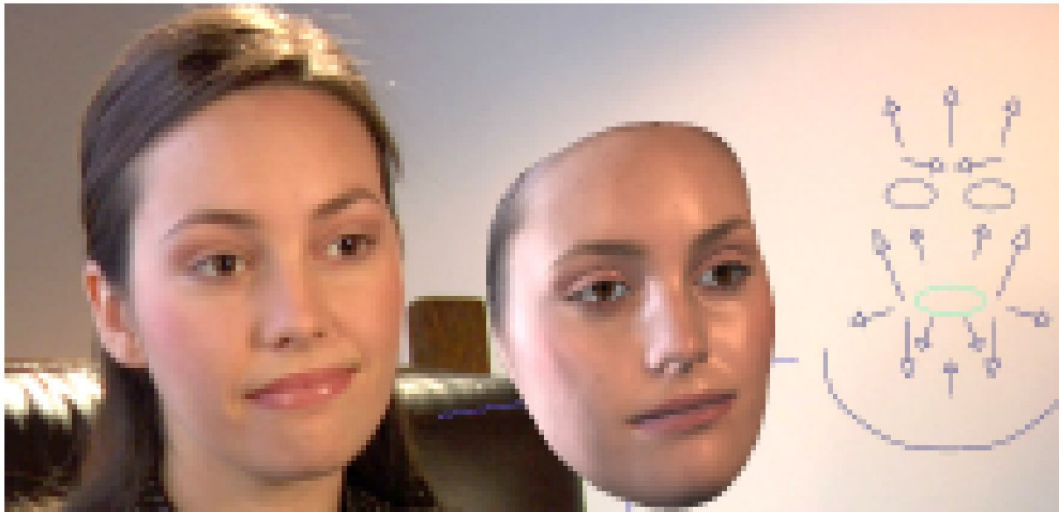
Fitting a Generic Face Mesh (2)

- We need to fit the reduced mesh to 32 expressions
- The dots drawn on the face of Emily were used to find the rough correspondence.
- The internal area were fitted based on the 3D location and normal vectors



Keyframe Animation

- Shown an image of the real Emily on the left, the user produce keyframes using the facial rig

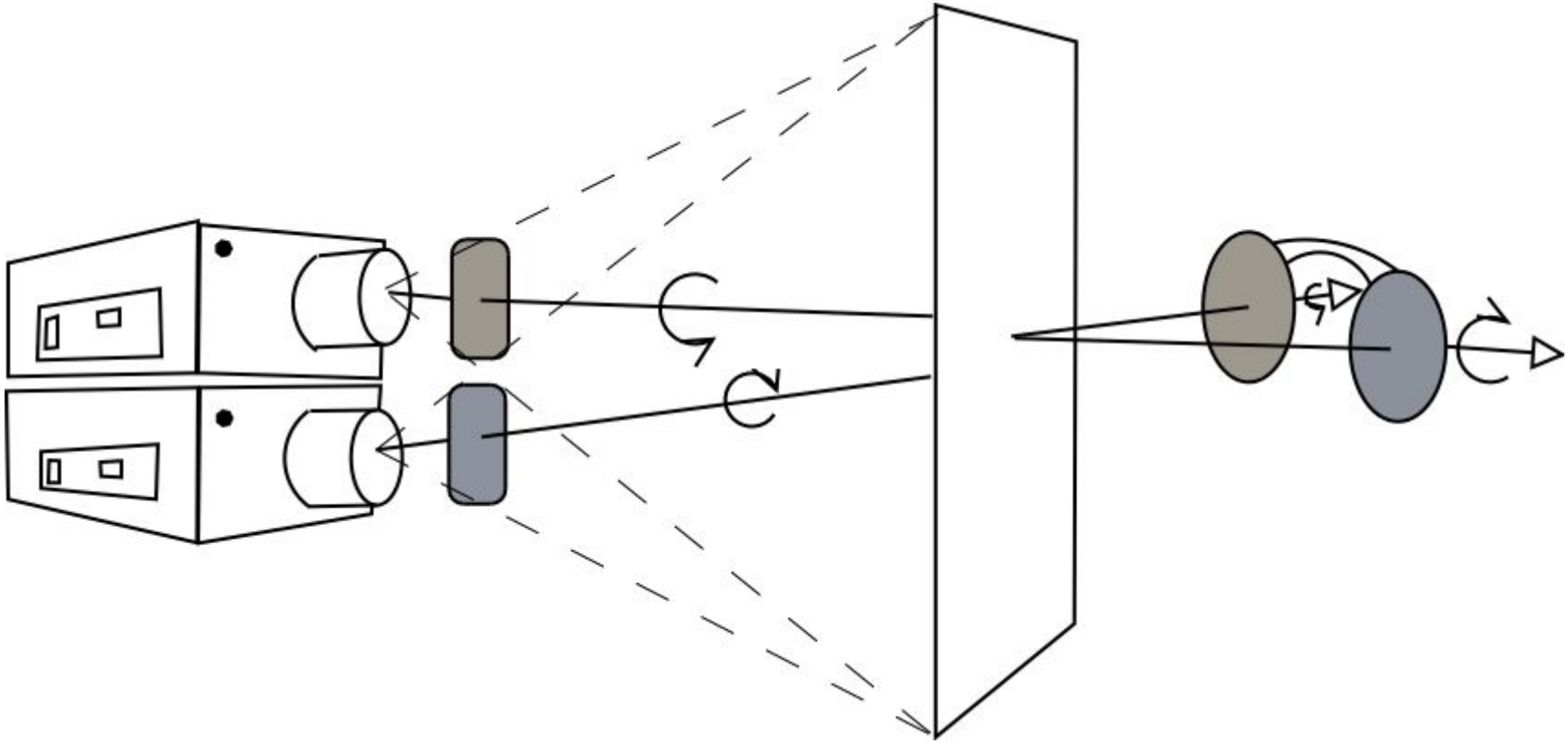


Extra Bits

- Teeth models are produced separately

- Sticky lips : subtle effect of the lips peeling apart during speech – the lips sticking to each other when pulled apart

Another application of polarization



Use pairs of polarized projectors and provide each viewer with inexpensive polarized glasses
These glasses have lenses polarized differently for the right and left eyes
Each eye sees the image component of the stereo pair with matching polarization

Summary

- Photometric Capturing
- Blendshapes, Face IK
- Facial Rigging
- Digital Emily
  - Capturing
  - Reflectance model
  - Polarization for obtaining a reflectance model
  - Animation synthesis by an interface

Readings

- Spacetime faces
  - http://grail.cs.washington.edu/projects/stfaces/

- Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination, EGSR 2007
- Nehab et al. Efficiently Combining Positions and Normals for Precise 3D Geometry", SIGGRAPH 2005
- Creating a Photoreal Digital Actor: The Digital Emily Project, SIGGRAPH 2009 Course

- http://www.worsleyschool.net/science/files/polarized/light.html
- http://gl.ict.usc.edu/Research/DigitalEmily/
- http://gl.ict.usc.edu/Research/DigitalEmily/DigitalEmily-IEEECGA-2010.pdf