R&D Topics in Computer Film

j.p.lewis

R&D Topics in Computer Film

(About the author)

- Computer Graphics and Immersive Technology Lab, U. Southern California; Digimax
- previously: ILM, ESC, Disney TSL, NYIT
- Film credits on *Forest Gump, Godzilla, Matrix Reloaded*
- publications (Siggraph, TOG, CHI, GI)
- algorithms adopted by Softimage XSI, Houdini, Shake, Commotion, RenderDotC

Topics

- computer graphics research - in Hollywood?
- scattered interpolation, + applications (Pose Space, 101 Dalmations, Matrix Reloaded)
- virtual actors and face tracking (Gemini Man, Matrix Reloaded, Digimax, USC)

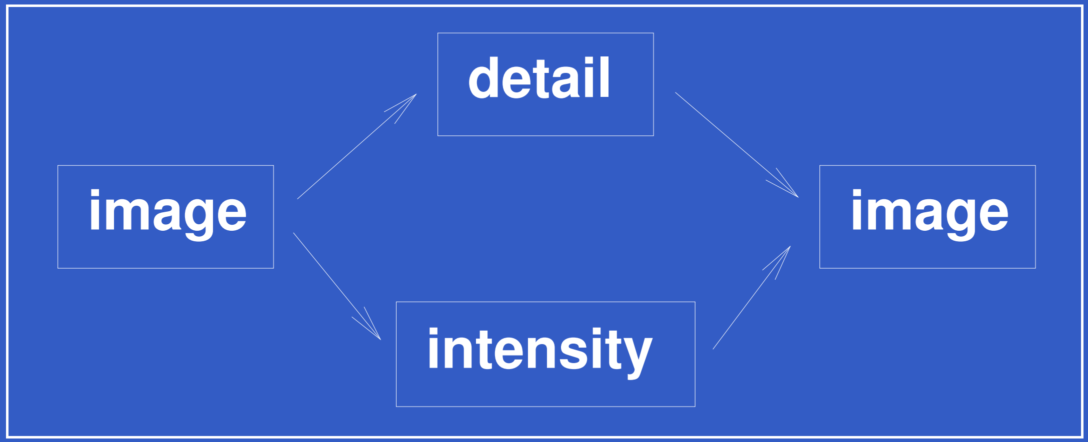
R&D in Computer Film

- most algorithms are invented in academic labs
- some algorithms later adopted in commercial software
- programmers in Hollywood: usually doing "uninteresting" projects, such as file format translators
- but there are exceptions!

102 Dalmations

J.P.Lewis, Lifting Detail from Darkness, SIGGRAPH 2001

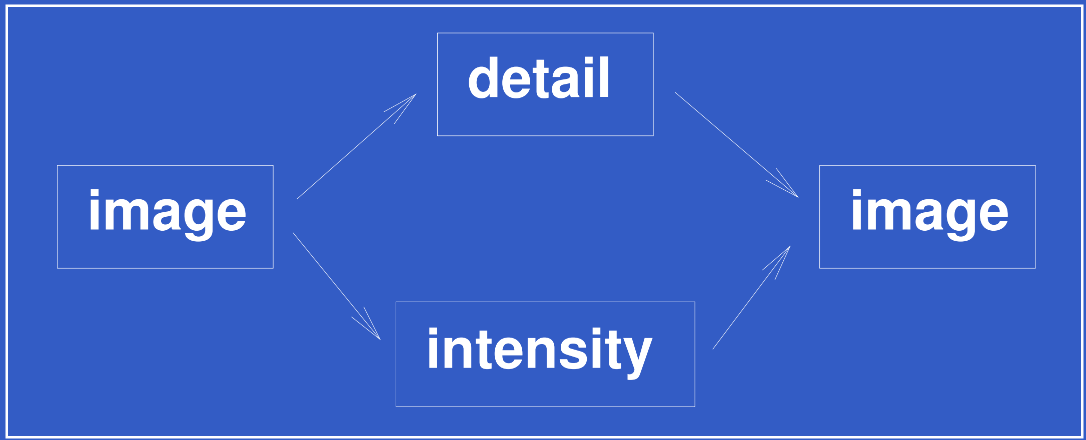
Brightness-Detail Decomposition



- Separate detail by Wiener filter
- Represent brightness by membrane PDE
- “Unsharp masking 2.0”

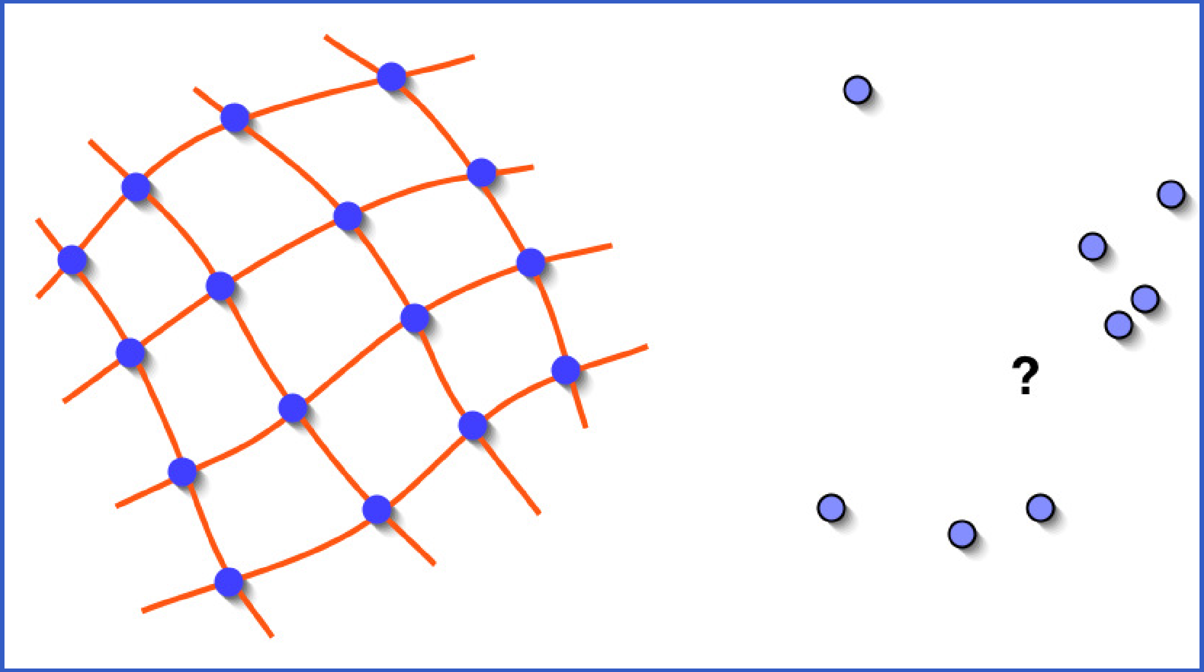
Topics: Wiener filter, Laplace PDE, multigrid

Brightness-Detail Decomposition



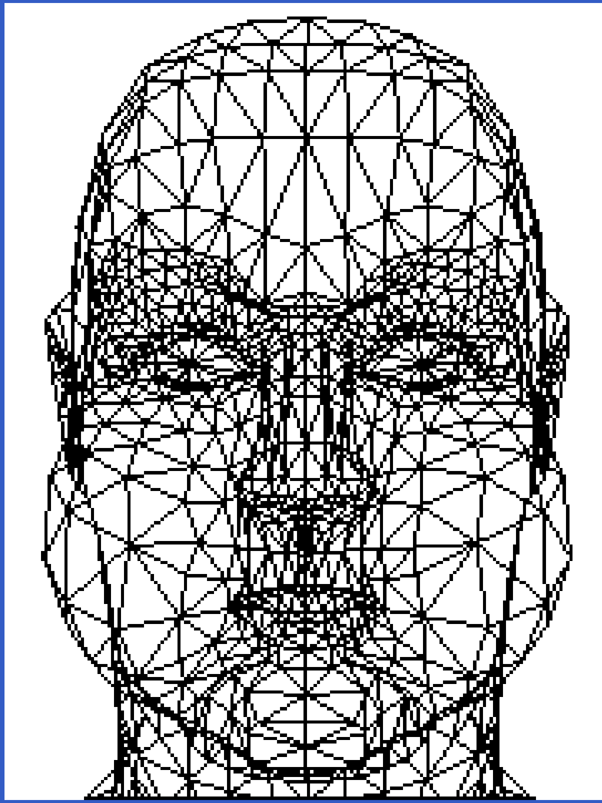
- Modify detail, keep brightness. Example: texture replacement (subject to limitations of 2D technique)
- Modify brightness, keep detail. Example: 102 Dalmatians spot removal

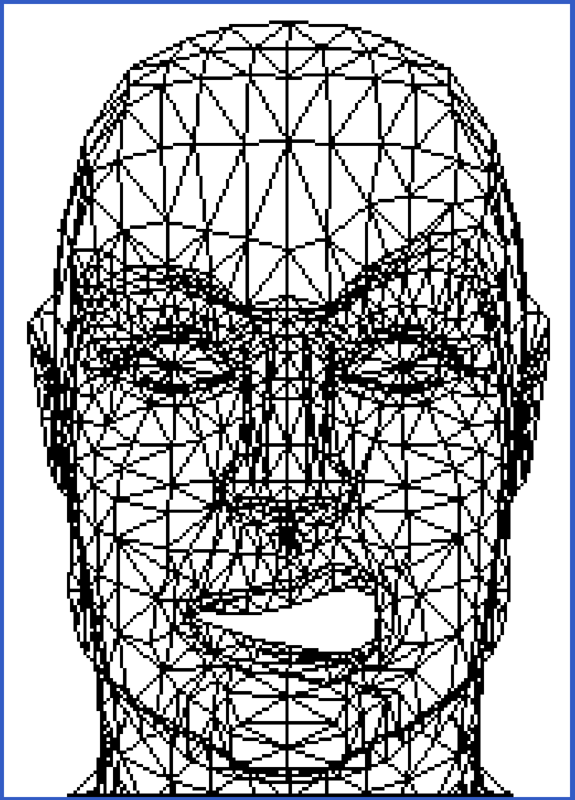
Scattered interpolation



Modeling

Deforming a face mesh





Images: Jun-Yong Noh and Ulrich Neumann, CGIT lab

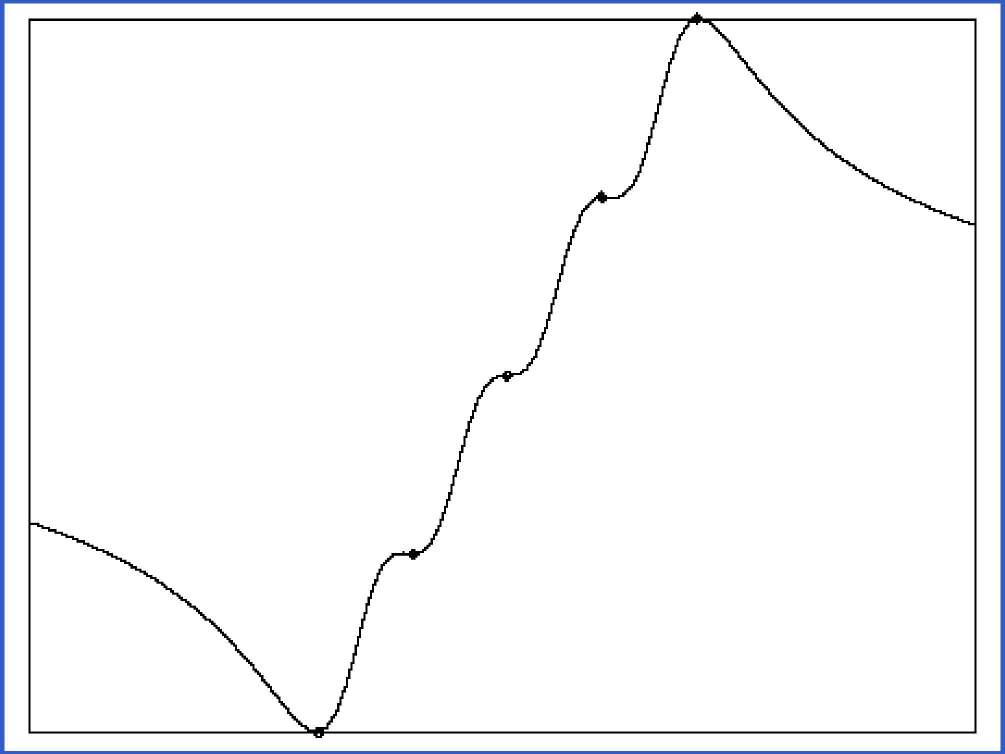
Shepard Interpolation

$$\hat{d}(\mathbf{x}) = \frac{\sum w_k(\mathbf{x}) d_k}{\sum w_k(\mathbf{x})}$$

weights set to an inverse power of the distance:
$w_k(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_k\|^{-p}$.

Note: singular at the data points $\mathbf{x}_k$.

Shepard Interpolation



Laplace/Poisson interpolation

minimize $\int (f'(x))^2 dx$

should come out like $\frac{d^2 f}{dx^2} = \nabla^2 = 0$

$F(y, y', x) = y'^2$

$\delta F = \frac{\partial F}{dy'} \frac{dy'}{d\epsilon} \delta\epsilon$

$= \frac{\partial F}{dy'} q' \delta\epsilon$

$\frac{\partial F}{dy'} = 2y' = 2\frac{df}{dx}$

$\frac{dE}{d\epsilon} = \int \frac{\partial F}{dy'} q' dx$

now change q' to q

$\int \frac{\partial F}{dy'} q' dx = \frac{\partial F}{dy'} q - \int \frac{d}{dx} \frac{\partial F}{dy'} q dx$

integration by parts

$\frac{\partial F}{dy'} q \Big|_a^b = 0$

because q is zero at both ends

$\frac{dE}{d\epsilon} = -\int \frac{d}{dx} \frac{\partial F}{dy'} q dx = 0$

variation of functional is zero at minimum

$= -2 \frac{d}{dx} \frac{df}{dx} = -2 \frac{d^2 f}{dx^2} = -2\nabla^2 f = 0$

Laplace/Poisson Interpolation

Objective: Minimize a roughness measure, the integrated derivative (or gradient) squared:

$$\int \frac{df}{dx}^2 dx$$

$$\int\int |\nabla f|^2 ds$$

Laplace/Poisson Interpolation

Discrete/matrix viewpoint: Encode derivative operator in a matrix D

$$D = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & -1 & 1 & \\ & & & \cdots \end{bmatrix}$$

$$\min_f f^T D^T D f$$

Laplace/Poisson Interpolation

$$\min_f f^T D^T D f$$

$$2 D^T D f = 0$$

i.e.

$$\frac{d^2 f}{dx^2} = 0 \text{ or } \nabla^2 = 0$$

$f = 0$ is a solution; last eigenvalue is zero, corresponds to a constant solution.

Laplace/Poisson: alternate derivation

Local viewpoint:

roughness $$R = \int |\nabla u|^2 du \approx \sum (u_{k+1} - u_k)^2$$

for a particular k:

$$\frac{dR}{du_k} = \frac{d}{du_k}[(u_k - u_{k-1})^2 + (u_{k+1} - u_k)^2]$$

$$= 2(u_k - u_{k-1}) - 2(u_{k+1} - u_k) = 0$$

$$u_{k+1} - 2u_k + u_{k-1} = 0 \rightarrow \nabla^2 u = 0$$

Notice: $D^T D = \ldots 1, -2, 1$

Laplace/Poisson: solution approaches

- direct matrix inverse (better: Choleski)
- Jacobi (because matrix is quite sparse)
- Jacobi variants (SOR)
- Multigrid

Jacobi iteration

matrix viewpoint

$$Ax = b$$

$$(D + E)x = b$$ split into diagonal D, non-diagonal E

$$Dx = -Ex + b$$

$$x = -D^{-1}Ex + D^{-1}b$$ call $B = D^{-1}E, z = D^{-1}b$

$$x \leftarrow Bx + z$$ D^{-1} is easy

hope that largest eigenvalue of B is less than 1

Jacobi iteration

Local viewpont

Jacobi iteration sets each f_k to the solution of its row of the matrix equation, independent of all other rows:

$$\begin{aligned} & \sum A_{rc} f_c = b_r \\ \rightarrow \qquad & A_{rk} f_k = b_k - \sum_{j \neq k} A_{rj} f_j \\ & f_k \leftarrow \frac{b_k}{A_{kk}} - \sum_{j \neq k} A_{kj} / A_{kk} f_j \end{aligned}$$

Jacobi iteration

apply to Laplace eqn

Jacobi iteration sets each f_k to the solution of its row of the matrix equation, independent of all other rows:

$$\ldots f_{t-1} - 2f_t + f_{t+1} = 0$$

$$2f_t = f_{t-1} + f_{t+1}$$

$$f_k \leftarrow 0.5 * (f[k-1] + f[k+1])$$

In 2D,

```
 f[y][x] = 0.25 * ( f[y+1][x] + f[y-1][x] +
f[y][x-1] + f[y][x+1] )
```

But now let's interpolate

1D case, say f_3 is known. Three eqns involve f_3. Subtract (a multiple of) f_3 from both sides of these equations:

$$\begin{array}{lcl} f_1 - 2f_2 + f_3 = 0 & \rightarrow & f_1 - 2f_2 + 0 = -f_3 \\ f_2 - 2f_3 + f_4 = 0 & \rightarrow & f_2 + 0 + f_4 = 2f_3 \\ f_3 - 2f_4 + f_5 = 0 & \rightarrow & 0 - 2f_4 + f_5 = -f_3 \end{array}$$

$$L = \begin{bmatrix} 1 & -2 & 0 & \\ & 1 & 0 & 1 \\ & & 0 & -2 \\ & & \dots & \end{bmatrix}$$ one column is zeroed

Multigrid

r is known, e is not

$$
\begin{aligned}
Ax &= b \\
\tilde{x} &= x + e \\
r &= A\tilde{x} - b \\
r &= Ax + Ae - b \\
r &= Ae
\end{aligned}
$$

For Laplace/Poisson, r is smooth. So decimate, solve for e, interpolate. And recurse...

demo

Applications: Pose Space Deformation

Lewis/Cordner/Fong, SIGGRAPH 2000

incorporated in Softimage

Figure 8a. Comparison of PSD and SSD on an animating shoulder – PSD using only two sculpted poses.

Figure 8b. SSD on an animating shoulder. The shoulder area is especially problematic for SSD due to the large range of rotational movement.

(video)

Applications: Matrix virtual city

Radial Basis Functions

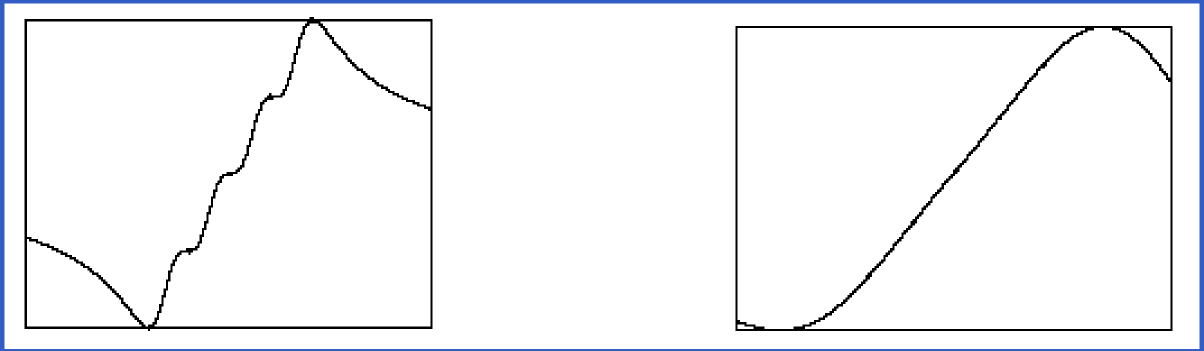
$$\hat{d}(\mathbf{x}) = \sum_{k}^{N} w_k \phi(\|\mathbf{x} - \mathbf{x}_k\|)$$

Radial Basis Functions (RBFs)

- *any* function other than constant can be used!
- common choices:
  - Gaussian $\phi(r) = \exp(-r^2/\sigma^2)$
  - Thin plate spline $\phi(r) = r^2 \log r$
  - Hardy multiquadratic
    $\phi(r) = \sqrt{(r^2 + c^2)}, c > 0$

Notice: the last two *increase* as a function of radius

RBF versus Shepard's



Solving RBF interpolation

- if few known points: linear system
- if few unknown points: multigrid

Radial Basis Functions

$$\hat{d}(\mathbf{x}) = \sum_{k}^{N} w_k \phi(\|\mathbf{x} - \mathbf{x}_k\|)$$

$$\begin{aligned} e &= \sum_{j} (d(\mathbf{x}_j) - \hat{d}(\mathbf{x}_j))^2 \\ &= \sum_{j} (d(\mathbf{x}_j) - \sum_{k}^{N} w_k \phi(\|\mathbf{x}_j - \mathbf{x}_k\|))^2 \\ &= (d(\mathbf{x}_1) - \sum_{k}^{N} w_k \phi(\|\mathbf{x}_1 - \mathbf{x}_k\|))^2 + (d(\mathbf{x}_2) - \sum_{k}^{N} w_k \phi(\|\mathbf{x}_2 - \mathbf{x}_k\|))^2 + \cdots \end{aligned}$$

Radial Basis Functions

$$\text{define } R_{jk} = \phi(\|\mathbf{x}_j - \mathbf{x}_k\|)$$

$$\begin{aligned}
&= (d(\mathbf{x}_1) - (w_1 R_{11} + w_2 R_{12} + w_3 R_{13} + \cdots))^2 \\
&\quad + (d(\mathbf{x}_2) - (w_1 R_{21} + w_2 R_{22} + w_3 R_{23} + \cdots))^2 + \cdots \\
&\quad + (d(\mathbf{x}_m) - (w_1 R_{m1} + w_2 R_{m2} + w_3 R_{m3} + \cdots))^2 + \cdots \\
\frac{d}{dw_m} &= 2(d(\mathbf{x}_1) - (w_1 R_{11} + w_2 R_{12} + w_3 R_{13} + \cdots))R_{1m} \\
&\quad + 2(d(\mathbf{x}_2) - (w_1 R_{21} + w_2 R_{22} + w_3 R_{23} + \cdots))R_{2m} \\
&\quad + \cdots \\
&\quad + 2(d(\mathbf{x}_m) - (w_1 R_{m1} + w_2 R_{m2} + w_3 R_{m3} + \cdots)) + \cdots = 0
\end{aligned}$$

put $R_{k1}, R_{k2}, R_{k3}, \cdots$ in row m of matrix.

Radial Basis Functions

$$\hat{d}(\mathbf{x}) = \sum_{k}^{N} w_k \phi(\|\mathbf{x} - \mathbf{x}_k\|)$$

$$e = ||(\mathbf{d} - \mathbf{\Phi w})||^2$$

$$e = (\mathbf{d} - \mathbf{\Phi w})^T(\mathbf{d} - \mathbf{\Phi w})$$

$$\frac{de}{d\mathbf{w}} = 0 = -\mathbf{\Phi}^T(\mathbf{d} - \mathbf{\Phi w})$$

$$\mathbf{\Phi}^T\mathbf{d} = \mathbf{\Phi}^T\mathbf{\Phi w}$$

$$\mathbf{w} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{d}$$

Thin plate spline

Minimize the integrated second derivative squared (approximate curvature)

$$\min_f \int \left(\frac{d^2 f}{dx^2} \right)^2 dx$$

Where does TPS kernel come from

Fit an unknown function f to the data y_k, regularized by minimizing a smoothness term.

$$E[f] = \sum (f_k - y_k)^2 + \lambda \int ||Pf||^2$$

$$\text{e.g.} \quad ||Pf||^2 = \int \left(\frac{d^2 f}{dx^2} \right)^2 dx$$

A similar discrete version.

$$E[f] = (f - y)' S' S (f - y) + \lambda 2 P' P f$$

Where does TPS kernel come from

(continued) A similar discrete version.

$$E[f] = (f-y)'S'S(f-y) + \lambda 2P'Pf$$

- To simplify things, here the data points to interpolate are required to be at discrete sample locations in the vector y, so the length of this vector defines a "sample rate" (reasonable).
- S is a "selection matrix" with 1s and 0s on the diagonal (zeros elsewhere). It has 1s corresponding to the locations of data in y. y can be zero (or any other value) where there is no data.
- P is a diagonal-constant matrix that encodes the discrete form of the regularization operator. E.g. to minimize the integrated curvature, rows of P will contain:

$$\begin{bmatrix} -2, 1, 0, 0, \ldots \\ 1, -2, 1, 0, \ldots \\ 0, 1, -2, 1, \ldots \end{bmatrix}$$

Where does TPS kernel come from

Take the derivative of E with respect to the vector f,

$$2S(f - y) + \lambda 2P'Pf = 0$$

$$P'Pf = -\frac{1}{\lambda}S(f - y)$$

Multiply by G, being the inverse of $P'P$:

$$f = GP'Pf = -\frac{1}{\lambda}GS(f - y)$$

So the RBF kernel is $G = (P'P)^{-1}$.

Matrix regularization

Find w to minimize $(Rw - b)^T(Rw - b)$. If the training points are very close together, the corresponding columns of R are nearly parallel. Difficult to control if points are chosen by a user.

Add a term to keep the weights small: $w^T w$.

$$\text{minimize} \qquad (Rw - b)^T(Rw - b) + \lambda w^T w$$

$$R^T(Rw - b) + 2\lambda w = 0$$

$$R^T Rw + 2\lambda w = R^T b$$

$$(R^T R + 2\lambda I)w = R^T b$$

$$w = (R^T R + 2\lambda I)^{-1} R^T b$$

Virtual Actors - Disney

Disney's *Gemini Man* test, 2001

(video)

Virtual Actors - Disney

Disney's *Gemini Man* test, 2001

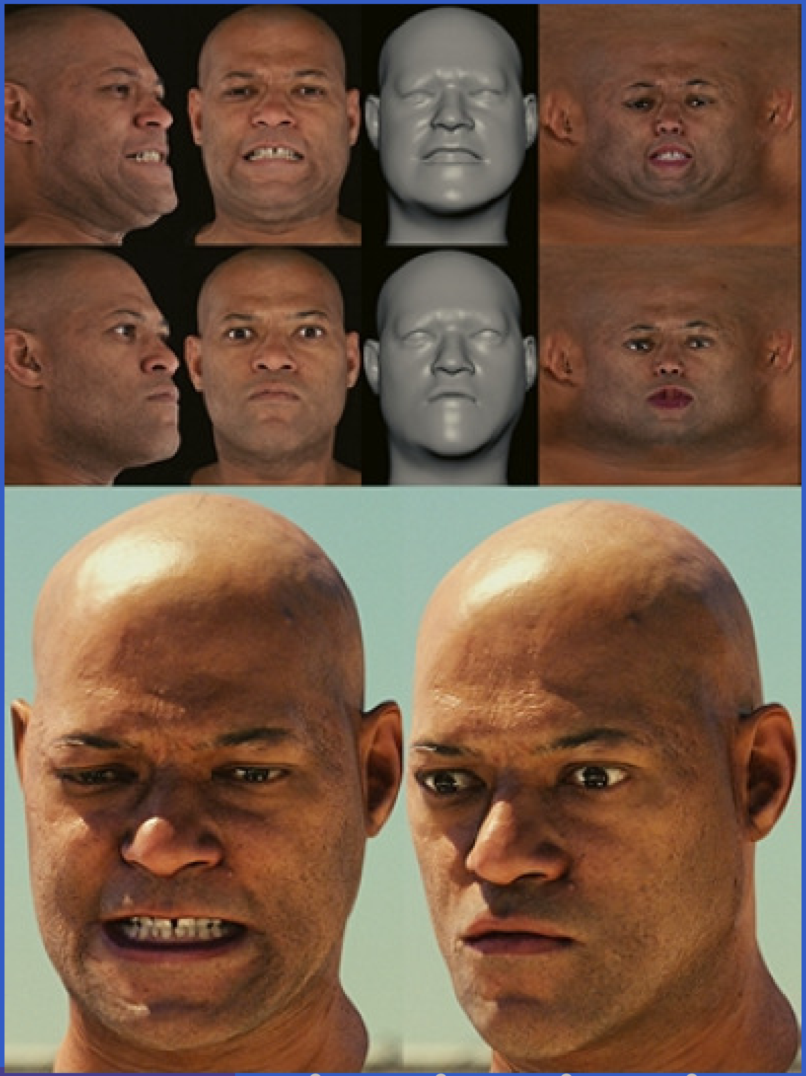
Optic flow, constrained by markers

Virtual Actors - Matrix Reloaded

- George Borshukov and J.P.Lewis Realistic Human Face Rendering for "The Matrix Reloaded" Siggraph 2003 Technical Sketch
- George Borshukov, Dan Piponi, Oystein Larsen, J. P. Lewis, Christina Tempelaar-Lietz Universal Capture: Image-BasedFacial Animation for "The Matrix Reloaded" Siggraph 2003 Technical Sketch

(pdfs)

Virtual Actors - Matrix Reloaded

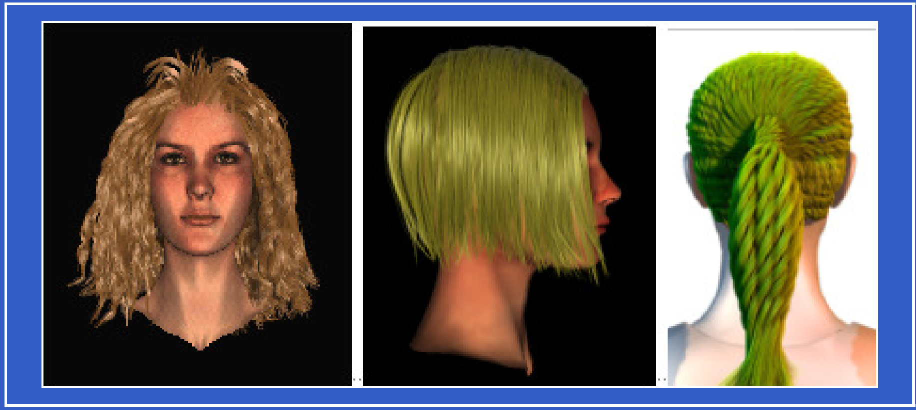


Face tracking - Digimax

(live demo)

Virtual Actors - USC research

- T.Y.Kim and U. Neumann, Interactive Multiresolution Hair Modeling and Editing, SIGGRAPH 2002 paper (video)
- Z. Deng, JP Lewis, U. Neumann, Practical Eye Movement Model using Texture Synthesis, SIGGRAPH 03 sketch



Conclusion

- learn the math
- numerical methods do matter