

Polynomial Texture Maps

Tom Malzbender
Dan Gelb
Hans Wolters
Hewlett-Packard Laboratories

Texture Mapping

[Catmull '74]

Pro:

- Photographic input
- Simplicity
- Hardware Support

Con:

- Unrealistic Silhouettes
- Static Lighting

Bump Mapping

[Blinn '78]

Pro:

- Lighting Variations

Con:

- Per-pixel lighting computation
- Filtering is Problematic
- Procedural Synthesis (Not image-based.) [Rushmeier '97]

PTM Demonstration:

Top: Polynomial Texture Map

Bottom: Conventional Texture Map

Advantages:

- Image based unlike Bump Mapping
- Simpler to evaluate than Bump Mapping
- Can leverage Mip Mapping

Acquiring PTM's Photographically

• Fixed object, fixed camera.
• Limited to Diffuse Objects.

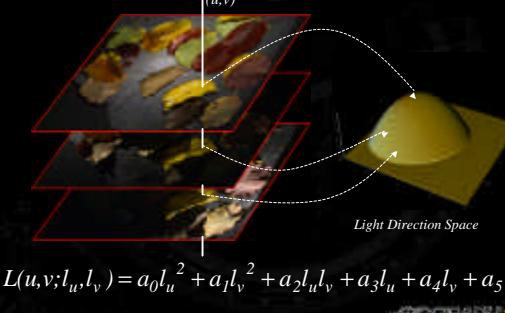
Acquiring Lighting Models

Debevec Goerghades '99

Modeling Pixel Color Changes Directly

$$L(u,v; l_u, l_v) = a_0 l_u^2 + a_1 l_v^2 + a_2 l_u l_v + a_3 l_u + a_4 l_v + a_5$$

Modeling Pixel Color Changes Directly



Polynomial Texture Mapping

PTM: Store RGB per pixel and
Store polynomial coefficients
(a_0-a_5) per texel:

$$L(u, v; l_u, l_v) = a_0 l_u^2 + a_1 l_v^2 + a_2 l_u l_v + a_3 l_u + a_4 l_v + a_5$$

$$\begin{aligned} R &= L \cdot R' \\ G &= L \cdot G' \\ B &= L \cdot B' \end{aligned}$$

Why Polynomials?

- Compact Representation
- Consist solely of multiplies and adds.
- Cheap to evaluate on both modern CPUs and VLSI

What PTMs Capture



Light Direction Parameterization

$$L(u, v; l_u, l_v) = a_0 l_u^2 + a_1 l_v^2 + a_2 l_u l_v + a_3 l_u + a_4 l_v + a_5$$

u, v - texture coordinates

a_0-a_5 - fitted coefficients stored in texture map

l_u, l_v - projection of light direction into texture plane

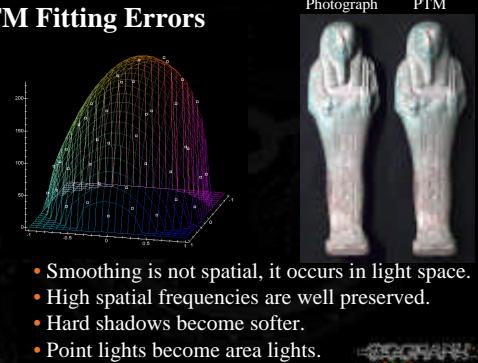


Fitting PTMs to Image Data

- Given N light sources we compute the best fit for (a_0-a_6) in the L_2 norm using S.V.D.
- SVD computed once for a given lighting arrangement.

$$\begin{bmatrix} l_{u0}^2 & l_{v0}^2 & l_{uv} l_{v0} & l_{u0} & l_{v0} & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} L_0 \\ L_1 \\ \vdots \\ L_{N-1} \end{bmatrix}$$

PTM Fitting Errors



PTM Formats

Format	Per Pixel Storage
• LRGB	
• RGB	
• ENC	Index to L.U.T. storing Polynomial Coefficients
...	

Scale and Bias

$$a_i = I_i(a_i - \Omega_i)$$

- Allows polynomial coefficients to be stored as 8 bit values.
- Handles large dynamic range among coefficients.
- 12 Global values stored per texture map (LRGB).



Mip Mapping PTM's vs Bump Maps

- Mip-mapping bump maps effectively smoothes geometry.
[Schilling 97]
- PTMs are linear in polynomial coefficients so mip-mapping PTMs is accurate.

$$L(u,v; l_u, l_v) = a_0 l_u^2 + a_1 l_v^2 + a_2 l_u l_v + a_3 l_u + a_4 l_v + a_5$$

$$\frac{1}{n} \sum_{i,j \in O} L_{r,g,b}(a_{0-5}(u_i, v_j)) = L_{r,g,b}\left(\frac{1}{n} \sum_{i,j \in O} a_{0-5}(u_i, v_j)\right)$$

Evaluation – MMX / SSD Implementation

Parallel computation

- Fixed point arithmetic
- Pack 4 PTM coefficients in 64 bit integer MMX register
- Parallel multiply/adds

Yields 6.5M pixels/sec on a 1 Ghz CPU (software only).



Evaluation – Programmable Hardware

Vertex Processing

- Store precomputed tangent and binormal per vertex
- Vertex code projects light vector onto tangent and binormal



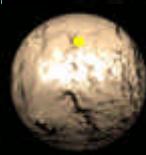
Pixel Processing

- l_u, l_v passed from vertex stage
- PTM coefficients stored in 2 textures
- Calculate using dot products / multiplies / adds
- Single pass on current hardware

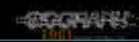
Bump Maps as PTM's

- If PTM rendering methods are implemented, they can be used for rendering bump maps.
- Provides specular and diffuse effects.

$$I = I_a k_a + I_d k_d (N \cdot L) + I_s k_s (N \cdot H)^n$$



- Precompute $N \cdot V$ PTM L.U.T.
 - Convert Normals to PTM using L.U.T.
 - Render PTM
- Diffuse - evaluate $N \cdot L$
Specular - evaluate $(N \cdot H)^n$



Complex Shading Effects

Combine with hardware lighting

- Use with existing Phong lighting
- PTM models more complex reflectance effects

• Examples:

Anisotropic
Fresnel
Off-specular



2D Applications

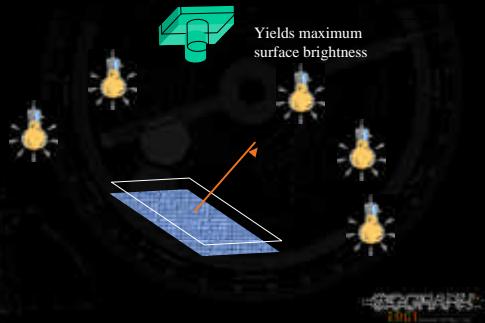
- Enhancement of Cuneiform Tablets w/ Zuckerman USC

- PTMs for Short Image Sequences

- PTMs for Depth of Focus Effects.



Surface Normal Extraction



Surface Normal Extraction

For a diffuse object, coordinates of (l_u, l_v) that maximize luminance yield local surface normals.

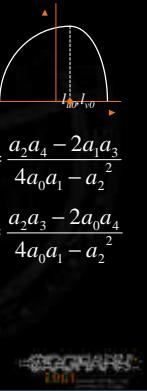
$$\text{Setting } \frac{\partial L}{\partial u} = \frac{\partial L}{\partial v} = 0 \text{ yields:}$$

$$l_{u0} = \frac{a_2 a_4 - 2 a_3 a_5}{4 a_0 a_1 - a_2^2}$$

$$l_{v0} = \frac{a_2 a_3 - 2 a_0 a_4}{4 a_0 a_1 - a_2^2}$$

Providing a surface normal per texel:

$$\bar{N} = (l_{u0}, l_{v0}, \sqrt{1 - l_{u0}^2 - l_{v0}^2})$$



Specular Enhancement



Cuneiform tablet courtesy of Dr. Bruce Zuckerman at U.S.C.

Diffuse Gain - a reflection transformation that:

- Keeps the surface normal fixed.
- Increases the curvature (second derivative) of the reflectance function by g .

$$a_0' = g a_0$$

$$a_1' = g a_1$$

$$a_2' = g a_2$$

$$a_3' = (1-g)(2a_0 l_{u0} + a_2 l_{v0}) + a_3$$

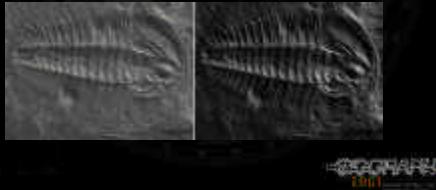
$$a_4' = (1-g)(2a_1 l_{v0} + a_2 l_{u0}) + a_4$$

$$a_5' = (1-g)(a_0 l_{u0}^2 + a_1 l_{v0}^2 + a_2 l_{u0} l_{v0}) + (a_3 - a_3') l_{u0} + (a_4 - a_4') l_{v0} + a_5$$



Light Direction Extrapolation

- Input images are collected across a hemisphere of light directions, i.e. $-1 \leq l_u, l_v \leq 1$
- PTM's can be evaluated outside of the hemisphere, ($l_u, l_v < -1$ or $l_u, l_v > 1$)



PTMs as Parametric Images

For each (u, v) we have:

$$a_0 l_u^2 + a_1 l_v^2 + a_2 l_u l_v + a_3 l_u + a_4 l_v + a_5$$

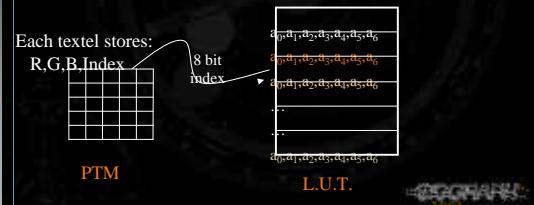


Depth of Focus



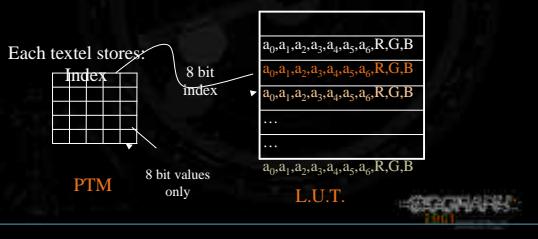
Palletization

- Random Access – Each pixel treated independently.
- Light Space Lookup table – contains polynomials.
- LUT constructed by K-means clustering.
- RGB values can be stored in L.U.T. or image space.



Palletization

- Random Access – Each pixel treated independently.
- Light Space Lookup table – contains polynomials.
- LUT constructed by K-means clustering.
- RGB values can be stored in L.U.T. or image space.



Compression

- Allows better rate/distortion tradeoff than palletization.
- Similar to compression of multispectral images.
- Removes correlations within and between byte planes.
- Sacrifices pixel independence.
- Visible artifacts don't appear until ~10 bits/pixel.

Perceptually lossless results:				
Original Size	Lossless	Loss = 1	Loss = 2	Loss = 4
72 - 144 bits	27.4 bits	17.1 bits	13.5 bits	10.1 bits

Future Work, Conclusions + Web Tools

- PTM's are fast, compact, effective representations.
- PTM's encoding opacity channels?
- Full BRDF's can be modeled using PTMs by trading off spatial variation with viewing angle.
- Applications in Medicine, Forensics, Paleontology

Tools available at hpl.hp.com/ptm

- sample PTMs
- PTM viewer
- Polynomial Fitter
- PTM format document



The End

hpl.hp.com/ptm

