TensorTextures: Multilinear Image-Based Rendering

Computer Graphics

Motivation

- Goal: Generation of photorealistic virtual environments
- Classical Computer Graphics: Model – based Rendering
  - From object models to images
  - Model specifies geometry of a scene and surface properties
  - Images are generated by projecting 3D model onto an image plane and computing surface shading
- Photorealism requires complex models
  - Difficult
  - Time consuming

Image-Based Rendering

[Gortler et al. 1996, Levoy & Hanrahan 1996, Debevec, Taylor & Malik 1996, …]

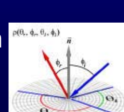
- World is modeled by a collection of images (and possibly some coarse geometry)
- These images are used to synthesize novel images representing the scene from arbitrary viewpoints and illuminations
- Advantages:
  - Rendering is decoupled from the scene complexity
  - Photorealism is improved

Our Contribution

- We introduce a tensor framework for image-based rendering (IBR)
  - Specifically, rendering of 3D textured surfaces
- Surface appearance is determined by the complex interaction of multiple factors:
  - Scene geometry
  - Illumination
  - Imaging

Bidirectional Texture Function

- BTF: Captures the appearance of extended textured surfaces with
  - Spatially varying reflectance
  - Surface mesostructure (3D texture)
  - Subsurface scattering
  - Etc.
- Generalization of BRDF, which accounts only for surface microstructure at a point



BTF Texture Mapping

[Dana et al. 1999]

	Concrete	Pebbles	Plaster
Standard Texture Mapping			
BTF Texture Mapping			

BTF

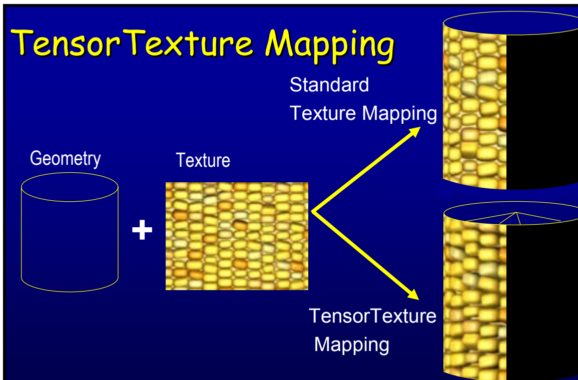
- Reflectance as a function of position on surface, view direction, and illumination direction

$$f_{BTF}(\underbrace{x,y}_{\text{position on surface}},\overbrace{\underbrace{\theta_v,\phi_v}_{\text{view direction}},\underbrace{\theta_i,\phi_i}_{\text{illumination direction}}}^{})$$

photometric angles

- The BTF captures shading and mesostructural self-shadowing, self-occlusion, interreflection

TensorTexture Mapping



TensorTextures: Learns BTFs from ensembles of sample images

Nonlinear generative BTF model

Background

- BTF introduced by Dana et al. [1999]
- BTF acquisition devices
  - [Debevec et al. 2000]
  - [Dana 2001]
  - [Furukawa et al. 2002]
  - [Han & Perlin 2003] (BTF Kaleidoscope)
- BTF based rendering methods
  - Polynomial texture maps [Malzbender et al. 2001]
  - Synthesis of BTFs for curved surfaces [Liu et al. 2001] [Tong et al. 2002]

TensorTextures Overview

1. Mathematical foundations: Eigentextures
   - Linear Analysis / Principal Components Analysis
     - fixed viewpoint, changing illumination
     - changing viewpoint and illumination
2. TensorTextures
   - Nonlinear (multilinear) Analysis / Tensor decomposition
3. Experiments and results

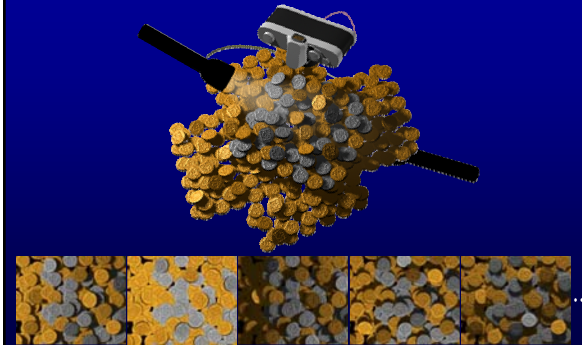
THE FOLLOWING **PREVIEW** HAS BEEN APPROVED FOR

ALL AUDIENCES

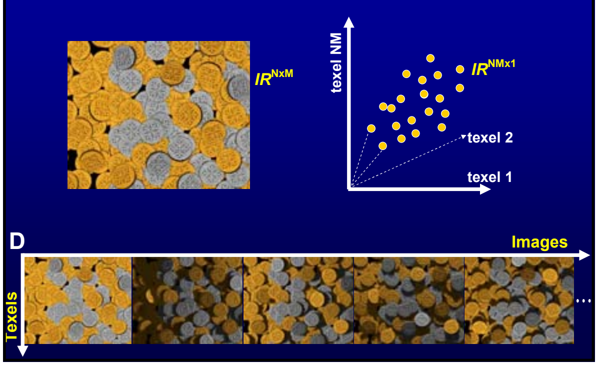
BY THE MOTION PICTURE DISASSOCIATION OF AMERICA

“Eigentextures” – PCA (Matrix Algebra)

Simple Data Acquisition: Fixed Viewpoint, Varying Illumination



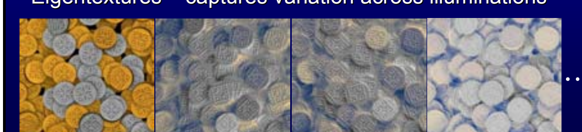
Sample images are points in “pixel space”



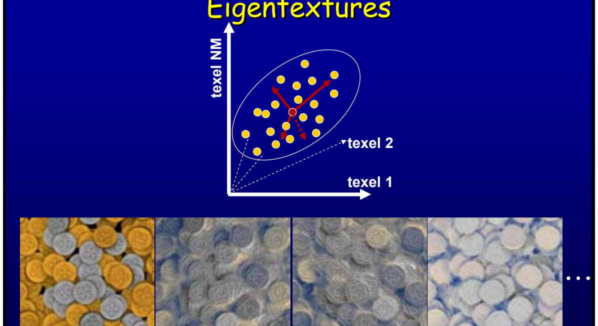
The 1-Mode Case

(fixed viewpoint, varying illumination)

- Eigentextures – captures variation across illuminations



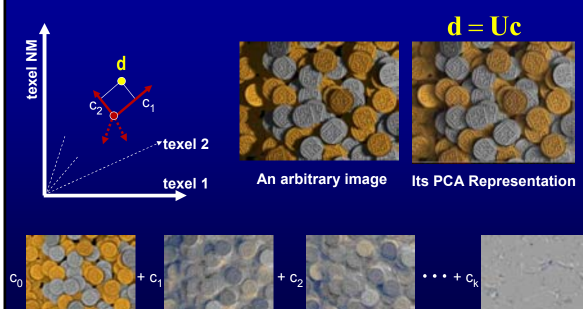
Principal Components Analysis (PCA) - Eigentextures



- Eigentextures – captures variation across illuminations

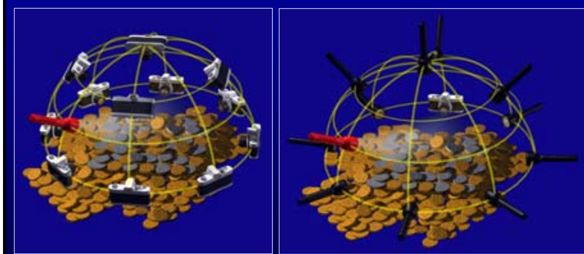
Image Representation using PCA

$\mathbf{d} = \mathbf{U}\mathbf{c}$



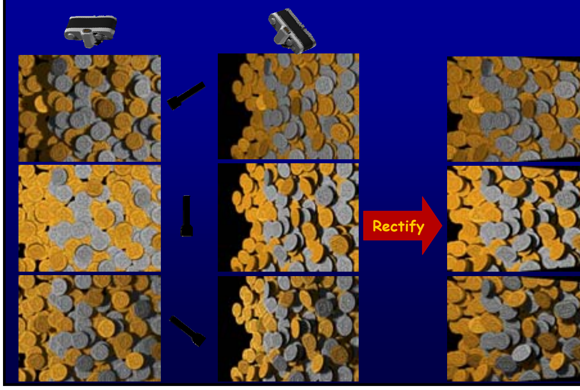
- Note: This is a linear representation

Sampling Multiple Viewpoints and Illuminations



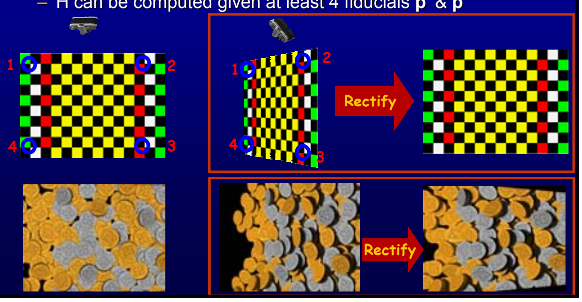
- This poses a <u>3-mode</u> BTF estimation problem
  - Viewpoint, illumination, and pixel modes

Image Rectification



Rectifying Homography

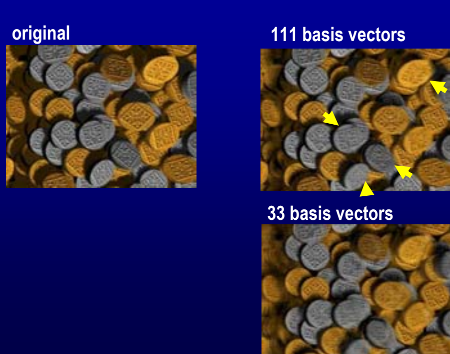
- Image unwarping $\mathbf{p}' = \mathbf{H}\mathbf{p}$
  - H can be computed given at least 4 fiducials **p'** & **p**



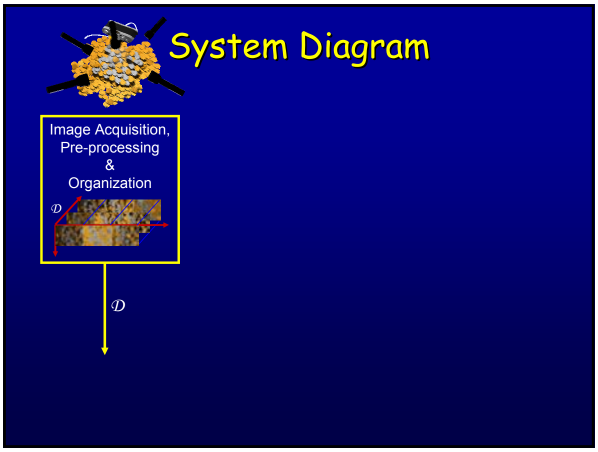
Applying PCA

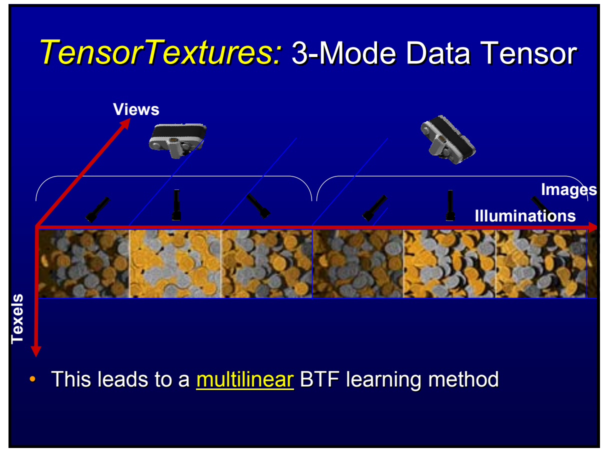
- Eigentextures – variation across views <u>and</u> illuminations

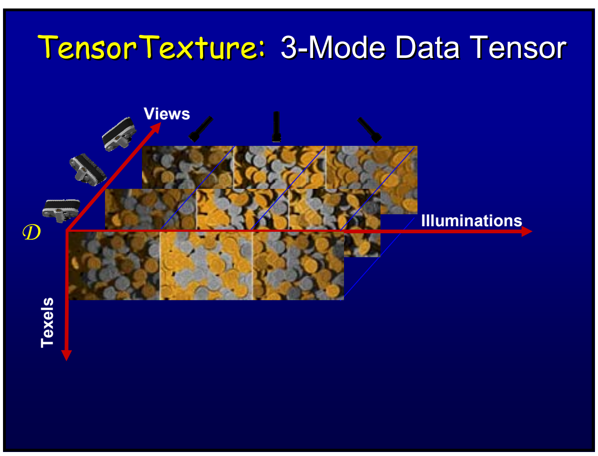
PCA Reconstruction

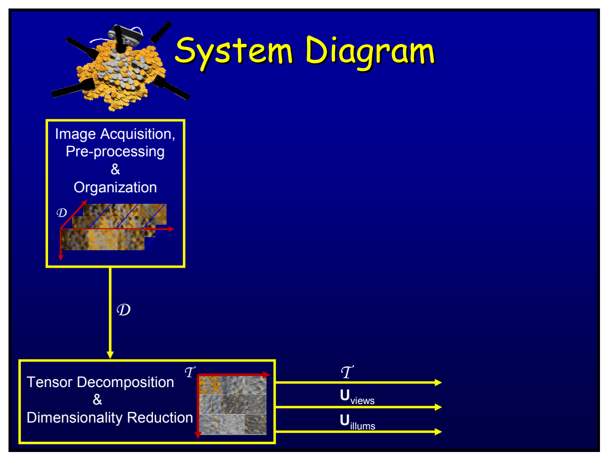


TensorTextures
(Tensor Algebra)









Background on Tensor Decomposition

- Factor Analysis:
  - Psychometrics, Econometrics, Chemometrics,…
- SVD:
  - [Beltrani, 1873] (Giornalle di Matematiche 11) "Sulle funzioni bilineari"
  - [Eckart and Young, 1936] (*Psychometrika*) "The approximation of one matrix by another of lower rank"
- 3-Way Factor Analysis:
  - [Tucker,1966] (*Psychometrika*) "Some mathematical notes on three mode factor analysis"
  - [Kroonenberg and De Leeuw, 1980] – 3-mode ALS
- N-Way Factor Analysis:
  - [Kapteyn, Neudecker, and Wansbeek, 1986] – N-way ALS factor analysis
  - [Franc, 1992] – tensor algebra
  - [Denis & Dhorne, 1989]
  - [de Lathauwer, 1997]

Matrix Decomposition - SVD

- A matrix $\mathbf{D} \in IR^{I_1 \times I_2}$ has a column and row space
- SVD orthogonalizes these spaces and decomposes $\mathbf{D}$

$$\mathbf{D} = \mathbf{U}_1 \mathbf{S} \mathbf{U}_2^T \qquad (\mathbf{U}_1 \text{ contains the "eigentextures"})$$

- Rewrite in terms of *mode-n products*

$$\boxed{\mathbf{D} = \mathbf{S} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2}$$

Tensor Decomposition

- $\mathcal{D}$ is a n-dimensional matrix, comprising N-spaces
- N-mode SVD is the natural generalization of SVD

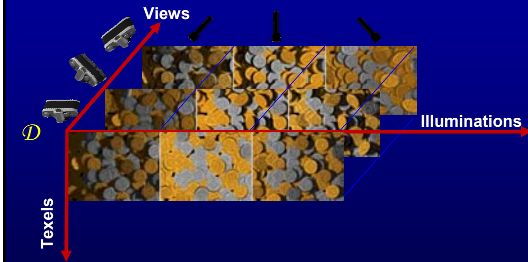
$$\mathbf{D} = \mathbf{U}_1 \mathbf{S} \mathbf{U}_2^T \quad \Longrightarrow \quad \mathbf{D} = \mathbf{S} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2$$

- N-mode SVD orthogonalizes these spaces & decomposes $\mathcal{D}$ as the mode-n product of N-orthogonal spaces

$$\boxed{\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3 \times_4 \cdots \times_N \mathbf{U}_N}$$

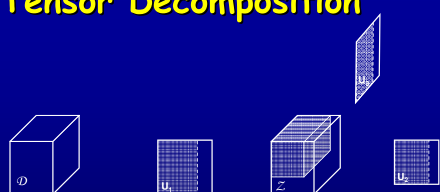
- $\mathcal{Z}$ core tensor; governs interaction between mode matrices
- $\mathbf{U}_n$ mode-n matrix, is the column space of $\mathbf{D}_{(n)}$

TensorTexture Decomposition



$$\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_{\text{texels}} \times_2 \mathbf{U}_{\text{illums}} \times_3 \mathbf{U}_{\text{views}}$$

Tensor Decomposition



$$\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_{\text{texels}} \times_2 \mathbf{U}_{\text{illums.}} \times_3 \mathbf{U}_{\text{views}}$$

$$= \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \sigma_{r_1 r_2 r_3} \mathbf{u}_{texels, r_1} \circ \mathbf{u}_{illums., r_2} \circ \mathbf{u}_{views, r_3}$$

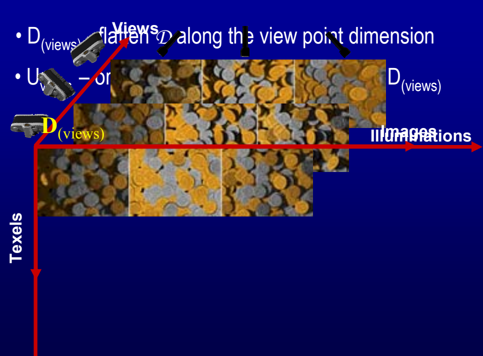
$$vec(\mathcal{D}) = (\mathbf{U}_{views} \otimes \mathbf{U}_{illums} \otimes \mathbf{U}_{texels}) \, vec(\mathcal{Z})$$

N-Mode SVD Algorithm

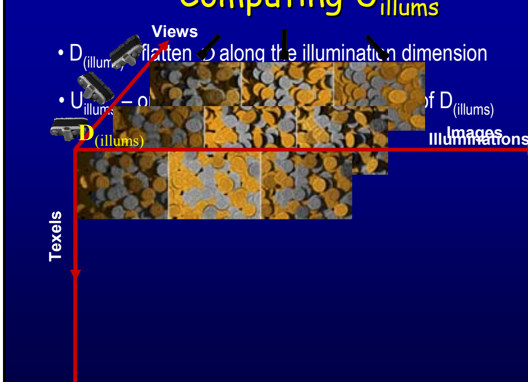
1. For n=1,…, N, compute matrix $\mathbf{U}_n$ by computing the SVD of the flattened matrix $\mathbf{D}_{(n)}$ and setting $\mathbf{U}_n$ to be the left matrix of the SVD.
2. Solve for the core tensor as follows

$$\mathcal{Z} = \mathcal{D} \times_1 \mathbf{U}_1^T \times_2 \mathbf{U}_2^T \cdots \times_N \mathbf{U}_N^T$$

Computing U_{views}



Computing U_{illums}



Computing U_{texels}

- $D_{(texels)}$ - flatten $\mathcal{D}$ along the pixel dimension
- U_{texels} – orthogonal column space of $D_{(texels)}$
  – eigenimages

N-Mode SVD Algorithm

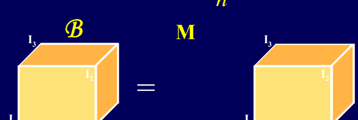
1. For n=1,…,N, compute matrix $\mathbf{U}_n$ by computing the SVD of the flattened matrix $\mathbf{D}_{(n)}$ and setting $\mathbf{U}_n$ to be the left matrix of the SVD.
2. Solve for the core tensor as follows

$$\mathcal{Z} = \mathcal{D} \times_1 \mathbf{U}_1^T \times_2 \mathbf{U}_2^T \cdots \times_N \mathbf{U}_N^T$$

Mode-N Product

- Mode-n product is a generalization of the product of two matrices
- It is the product of a tensor with a matrix
- Mode-n product of $\mathcal{A} \in IR^{I_1 x..x I_n x..x I_N}$ and $\mathbf{M} = IR^{J_n x I_n}$
  $\mathcal{B} \in IR^{I_1 x..x I_{n-1} x J_n x I_{n+1} x..x I_N}$

$$\mathcal{B} = \mathcal{A} \times_n \mathbf{M}$$



$$(\mathcal{A} \times_n \mathbf{M})_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} m_{j_n i_n}$$

Mode-N Product

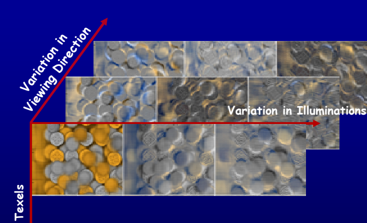
N-th order tensor $\quad \mathcal{A} \in \Re^{I_1 \times I_2 \times \cdots I_N}$

matrix (2-nd order tensor) $\quad \mathbf{M} \in \Re^{J_n \times I_n}$

mode-n product:

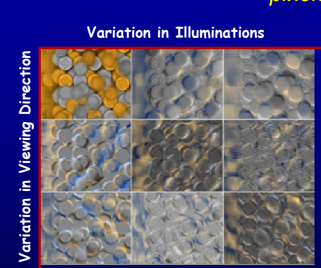
$$\mathcal{B} = \mathcal{A} \times_n \mathbf{M} \quad \text{where} \quad \mathbf{B}_{(n)} = \mathbf{M}\,\mathbf{A}_{(n)}$$

TensorTextures: $\mathcal{T} = \mathcal{Z} \times_3 \mathbf{U}_{pixels}$



TensorTextures:
explicitly represent covariance across factors

TensorTextures: $\mathcal{T} = \mathcal{Z} \times_3 \mathbf{U}_{pixels}$



TensorTextures:
explicitly represent covariance across factors

TensorTextures vs. PCA

- Multilinear Analysis / TensorTextures:

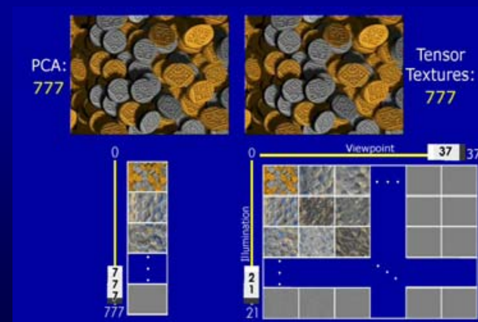
$$\mathcal{D} = \mathcal{Z} \times_1 \mathbf{U}_{\text{texels}} \times_2 \mathbf{U}_{\text{illums.}} \times_3 \mathbf{U}_{\text{views}}$$

- Linear Analysis :

$$\underbrace{\mathbf{D}_{(\text{texels})}}_{\text{data matrix}} = \underbrace{\mathbf{U}_{\text{texels}}}_{\text{basis matrix}} \underbrace{\mathbf{Z}_{(\text{texels})} \left(\mathbf{U}_{\text{views}} \otimes \mathbf{U}_{\text{illums}}\right)^T}_{\text{coefficient matrix}}$$

- *TensorTextures subsumes PCA / Eigentextures*

Strategic Dimensionality Reduction

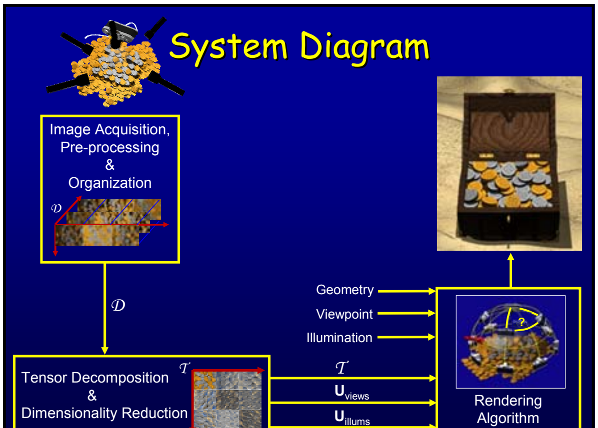


Strategic Dimensionality Reduction

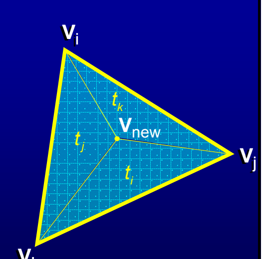
TensorTextures
Dimensionality
Reduction

#basis	PCA - Eigentextures	TensorTextures
111		
37		

System Diagram



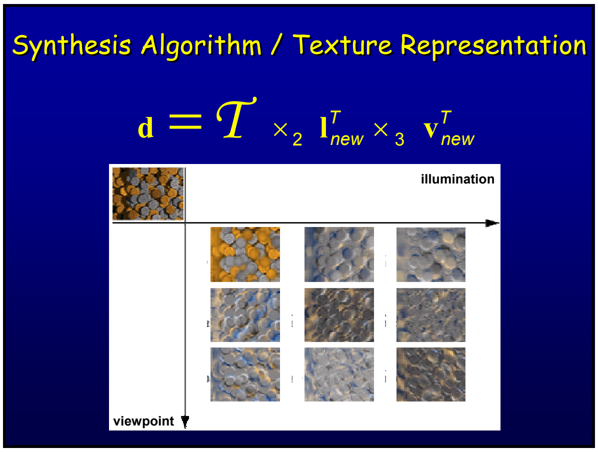
Computing $\mathbf{v}_{new}$: Homogeneous Barycentric Blend



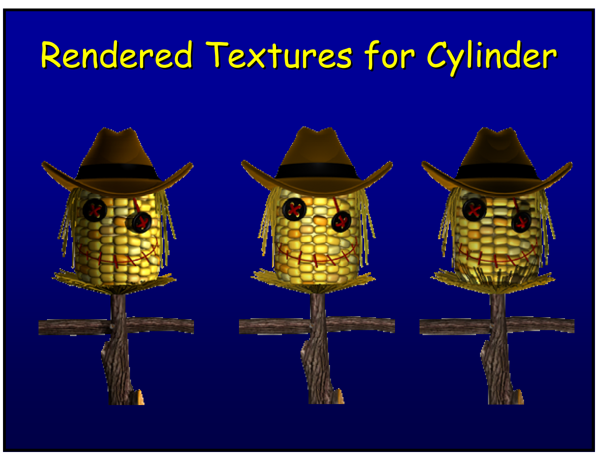
$$\mathbf{v}_{new} = \mathbf{v}_i\,\Delta t_i + \mathbf{v}_j\,\Delta t_j + \mathbf{v}_k\,\Delta t_k$$

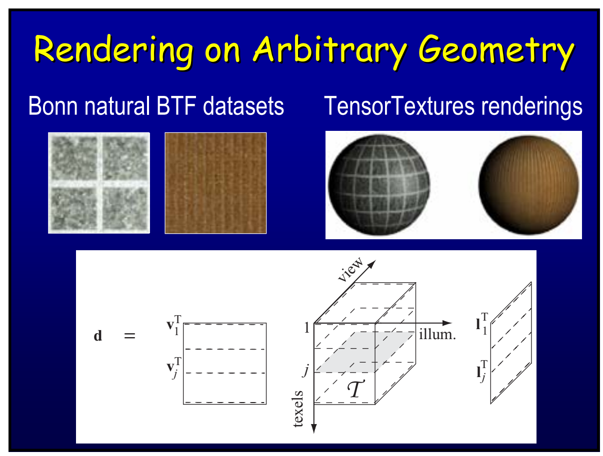
$$\Delta t_i + \Delta t_j + \Delta t_k = 1$$

$$\Delta t_i = \frac{\left((\mathbf{v}_j - \mathbf{v}_{new}) \times (\mathbf{v}_k - \mathbf{v}_{new})\right)}{\left((\mathbf{v}_j - \mathbf{v}_i) \times (\mathbf{v}_k - \mathbf{v}_i)\right)}$$


Synthesis Algorithm / Texture Representation
$\mathbf{d} = \mathcal{T} \times_2 \mathbf{l}^T_{new} \times_3 \mathbf{v}^T_{new}$
illumination
viewpoint


Rendered Texture for a Planar Surface


Rendered Textures for Cylinder


Rendering on Arbitrary Geometry
Bonn natural BTF datasets
TensorTextures renderings
$\mathbf{d} =$
$\mathbf{v}_1^T$
$\mathbf{v}_j^T$
view
illum.
texels
1
j
$\mathcal{T}$
$\mathbf{l}_1^T$
$\mathbf{l}_j^T$


Video
Scarecrows' Quarterly
Treasure Chest
Flintstones Bird


TensorTextures