Shaders

- **GLSL** Pixel shaders
- Determines color of each pixel in an image
- Data parallel
  - Does *nearly* the same thing at each pixel
Shaders Parameters

- **Varying parameters**: surface normal, distance to camera.
  - These are *vary* for each pixel.
  - User has no effect on these.
- **Uniform parameters**: Written by programmer in shader code.
  - Artist alters these, they *don't* vary per pixel.
The Problem

- Artists' search of parameter space can be time consuming
  - High dimensionality
  - Non-intuitive parameter names

- Solution: Allow artists to reparameterize shaders
Our Solution

- **System** for allowing artist's to reparameterize GLSL shaders
- Novel algorithms for reparameterization
- Implemented
  - Linear reparameterization
- Discussed
  - Non-linear reparameterization
  - Perceptually uniform reparameterization
System Example

- RGB shader, want to reparameterize as a grayscale shader
  - User inputs new parameter names
  - User sets high and low points for new parameters
  - Use new shader
Linear Reparameterization

- Consider a shader with $N$ uniform parameters
- User wants to reparameterize with $M$ parameters
  - Choose $M$ vectors $v_1, \ldots, v_M$ in $\mathbb{R}^N$
  - Give each vector a parameter name
- Change of basis to map from new parameter space to old parameter space
  - Matrix multiply
Problems with Linear Reparamaterization

● What if no linear map exists that maps new parameters to old parameters?
  – Consider reparameterizing RGB as HSV
  – Nonlinear reparameterization

● How do we scale the sliders for each new parameter
  – Perceptualy uniform reparameterization
Thanks!

- Paper + code available at:
  http://www.cs.virginia.edu/~cab6fh/PL/final_colin.zip

- Questions?