Style Transfer Functions for Illustrative Volume Rendering

Alessandro Febretti

CS526 – Scientific Visualization Spring 09

Authors

M.E. Gröller

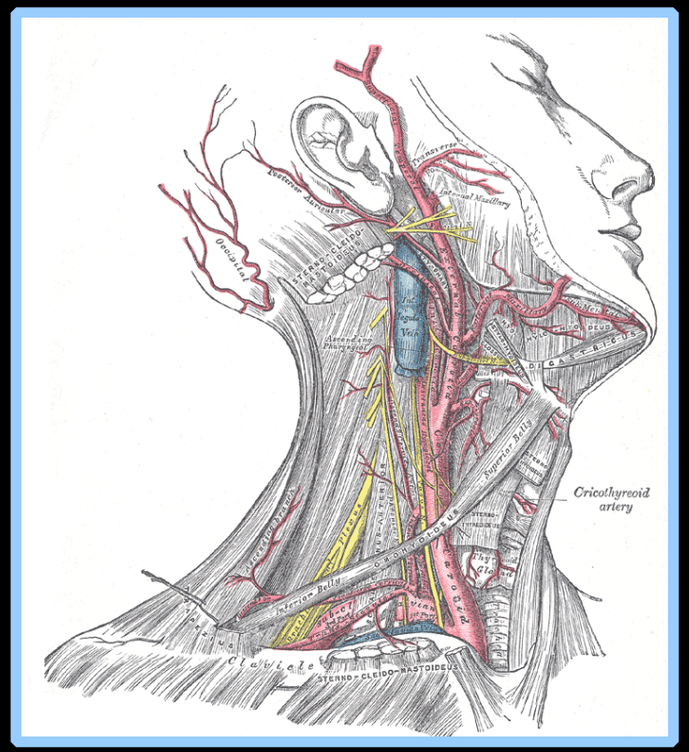
- Associate Professor, Institute of Computer Graphics and Algorithms, Vienna University

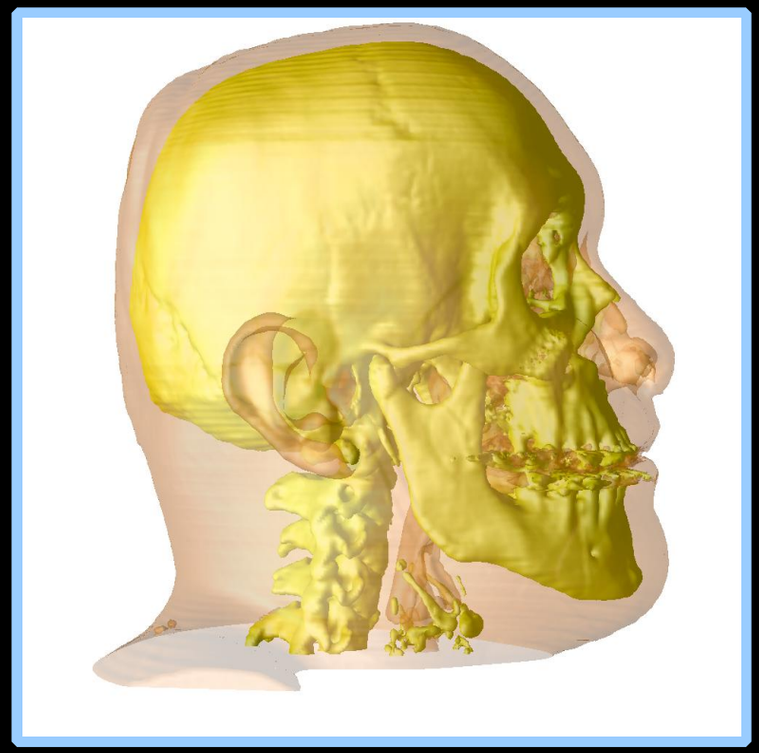
S. Bruckner

- Assistant Professor, Institute of Computer Graphics and Algorithms, Vienna University

Non-Photorealistic Volume Rendering

- Volume rendering usually based on approximate physical representations.





- Traditional illustration use non-photorealistic techniques to focus user on important features.

Style Transfer Functions

- Color Transfer Functions
  - Difficult to tweak to achieve desired results
  - Lighting model pretty much 'fixed' and must be configured separately
- **Style Transfer Functions**
  - Allow to easily specify an illustrative *style* to be applied to volume data
  - Color, opacity, **shading** & other features expressed in a consistent way.

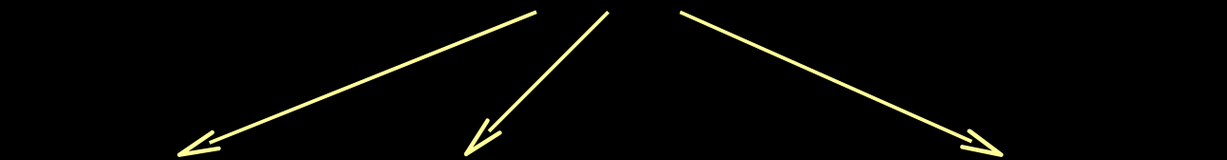
Style Transfer Functions, cont'd

- Color Transfer Functions

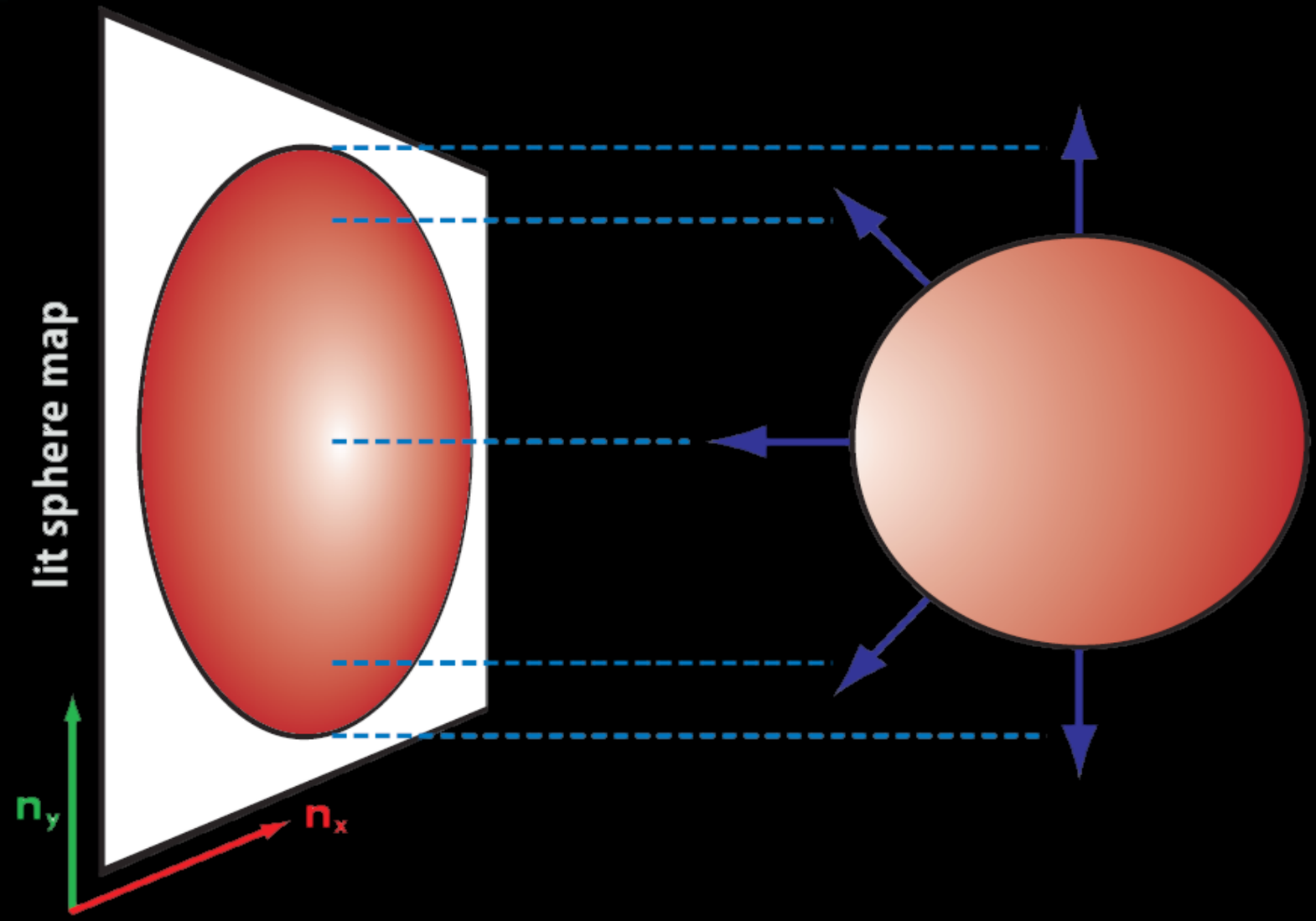
sample → Color xfer function → <color, alpha>

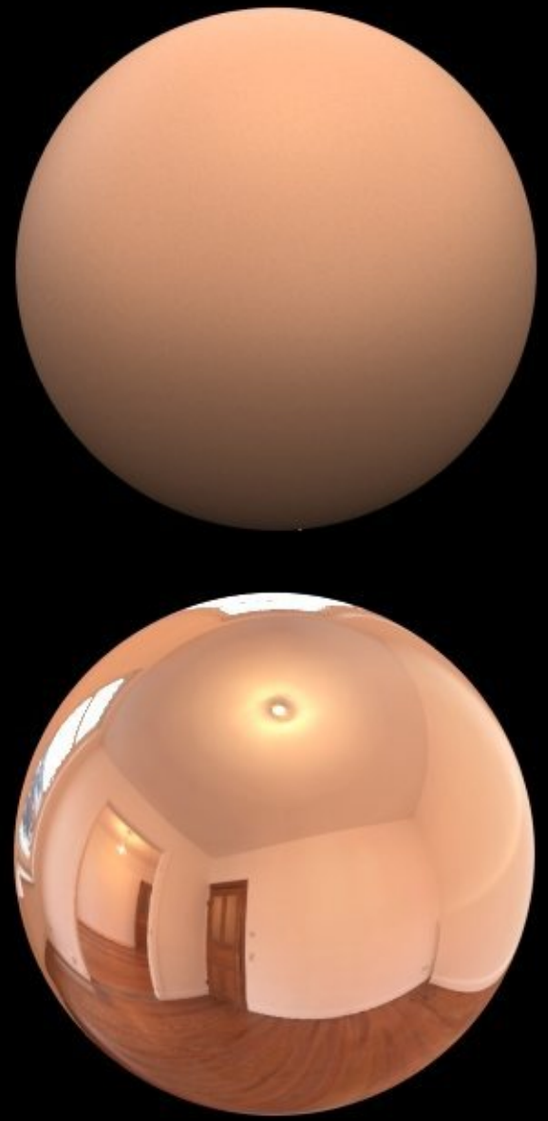
- Style Transfer Functions

sample → style xfer function → ***style***

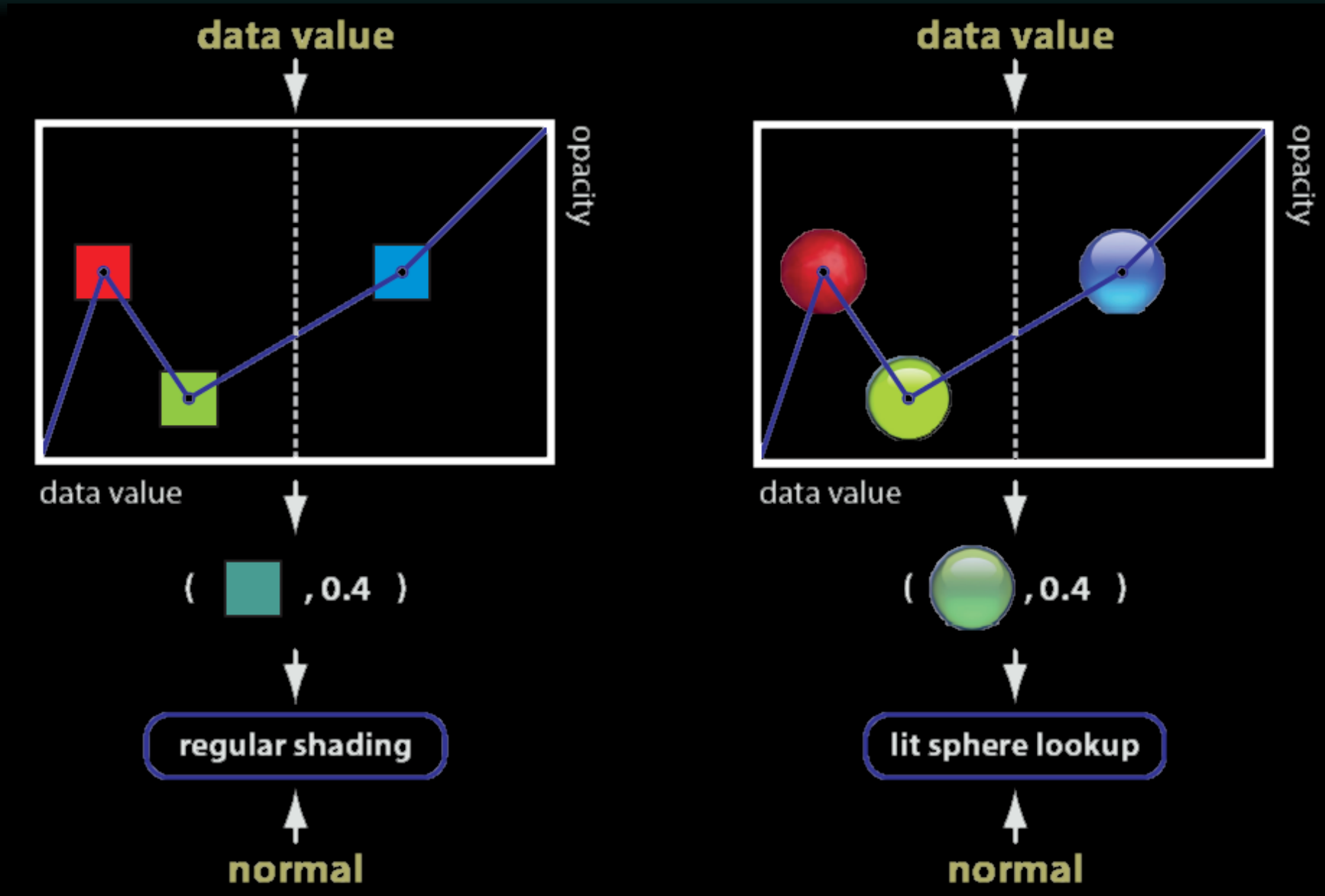


Style texture: Sphere map Shading

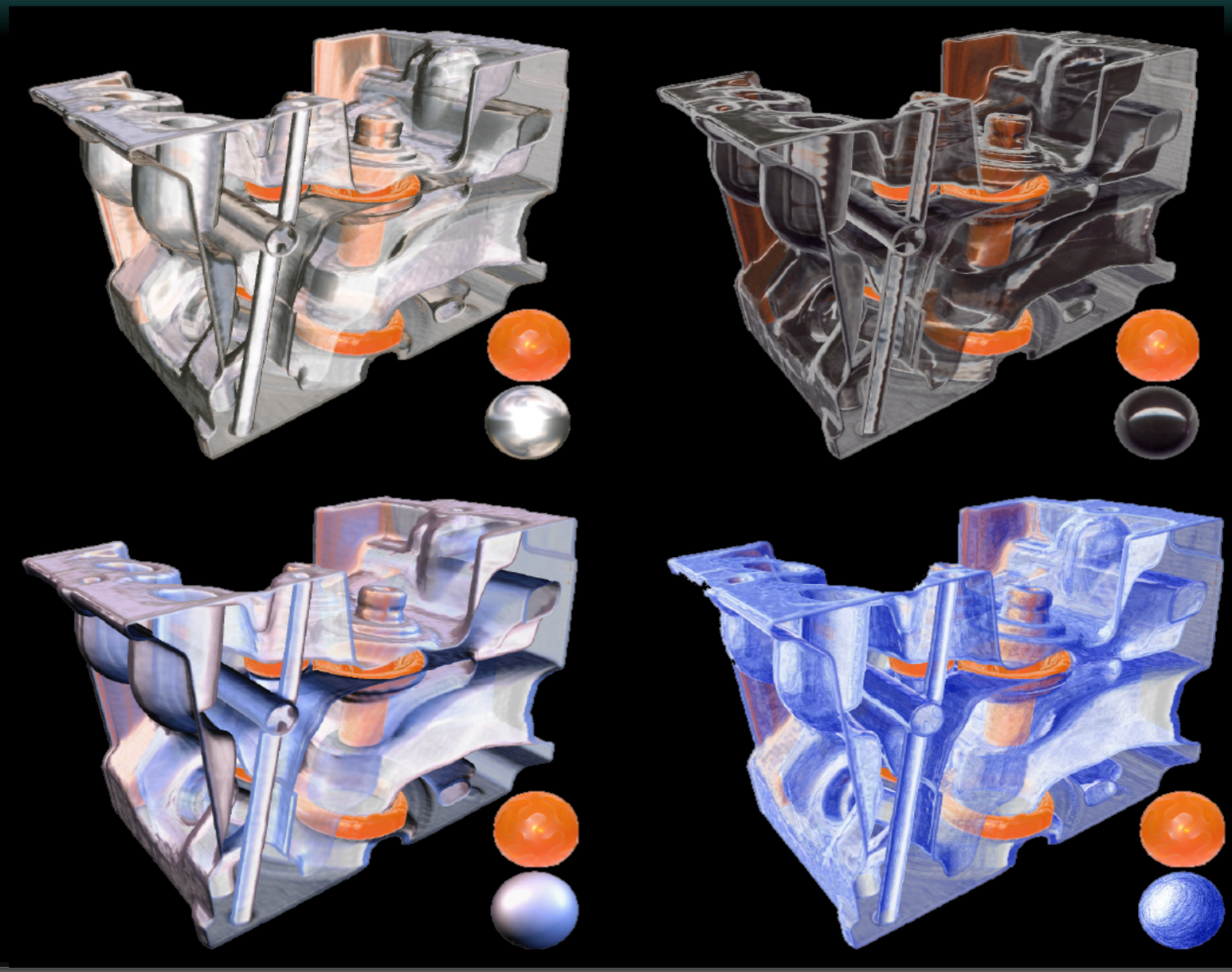




Standard vs Sphere Map Shading



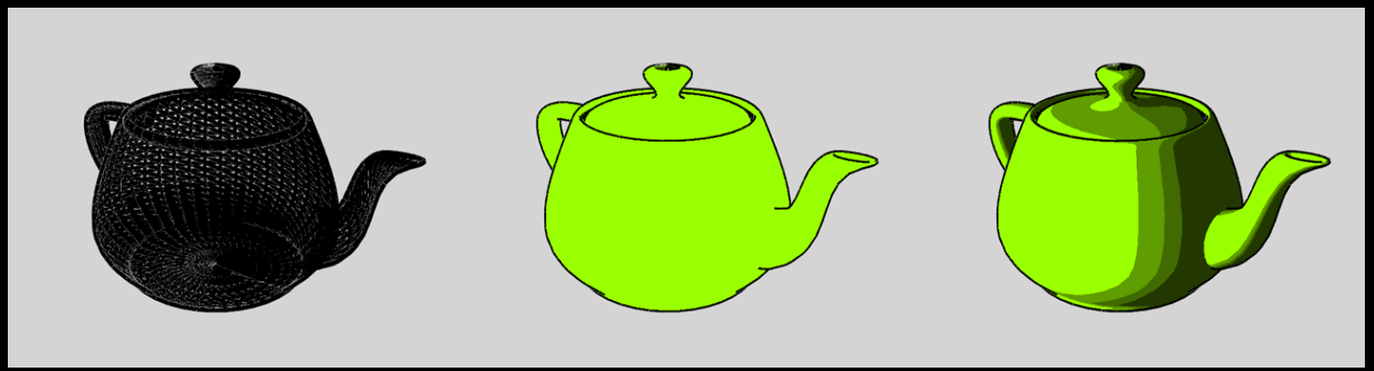
Example: Two styles



Style Contours

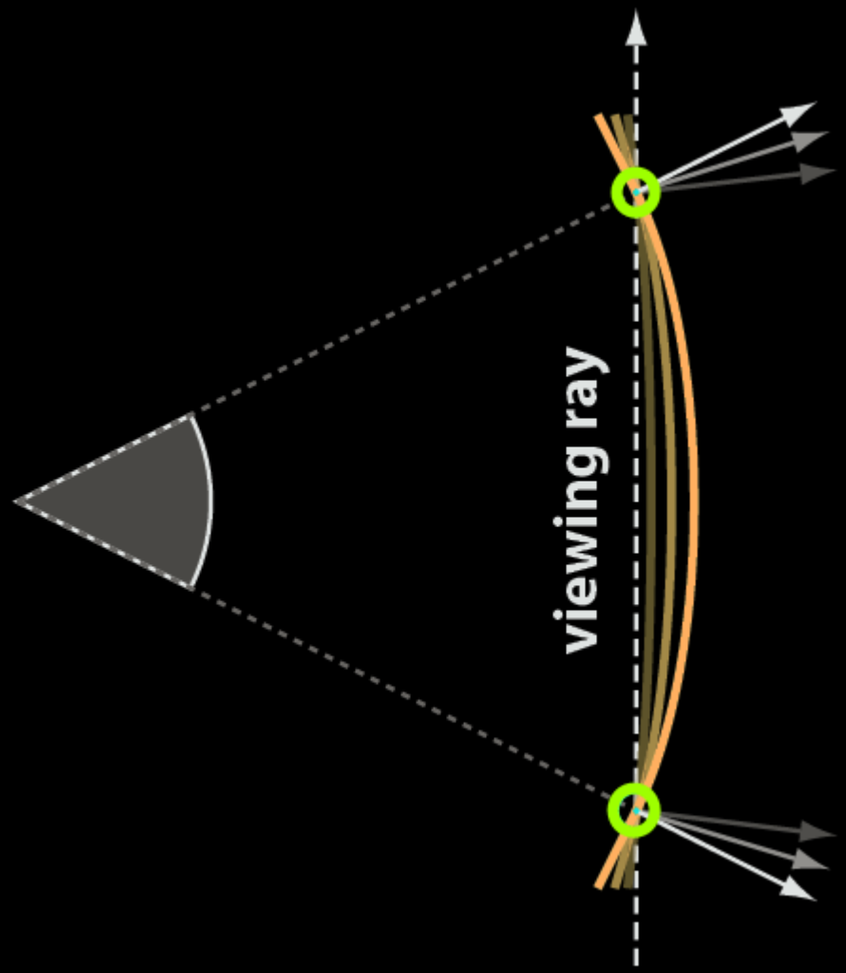
Contours help delineating object shapes

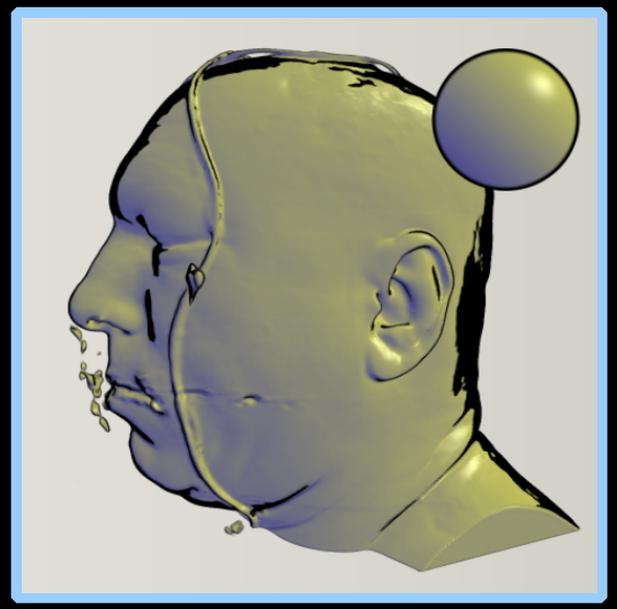
- Simple contour generation: viewVector • normal
  - Drawback: no control over thickness.
  - Refine using curvature info → requires 2nd order derivatives

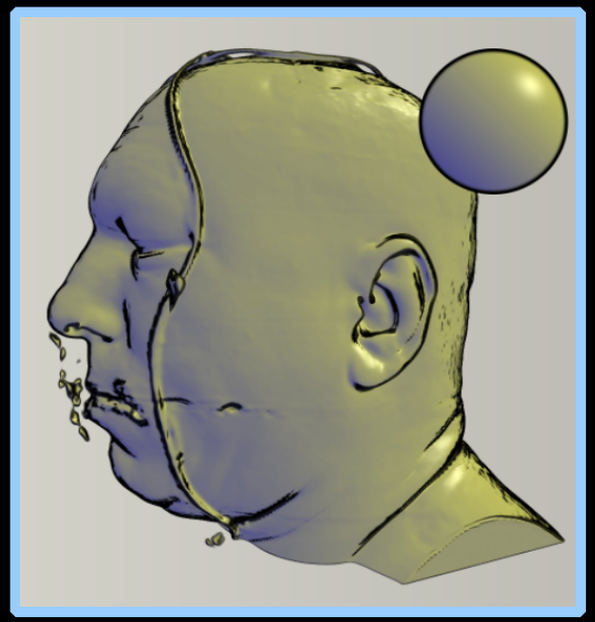


Style Contours, cont'd

- Solution: approx curvature sampling normal along view ray





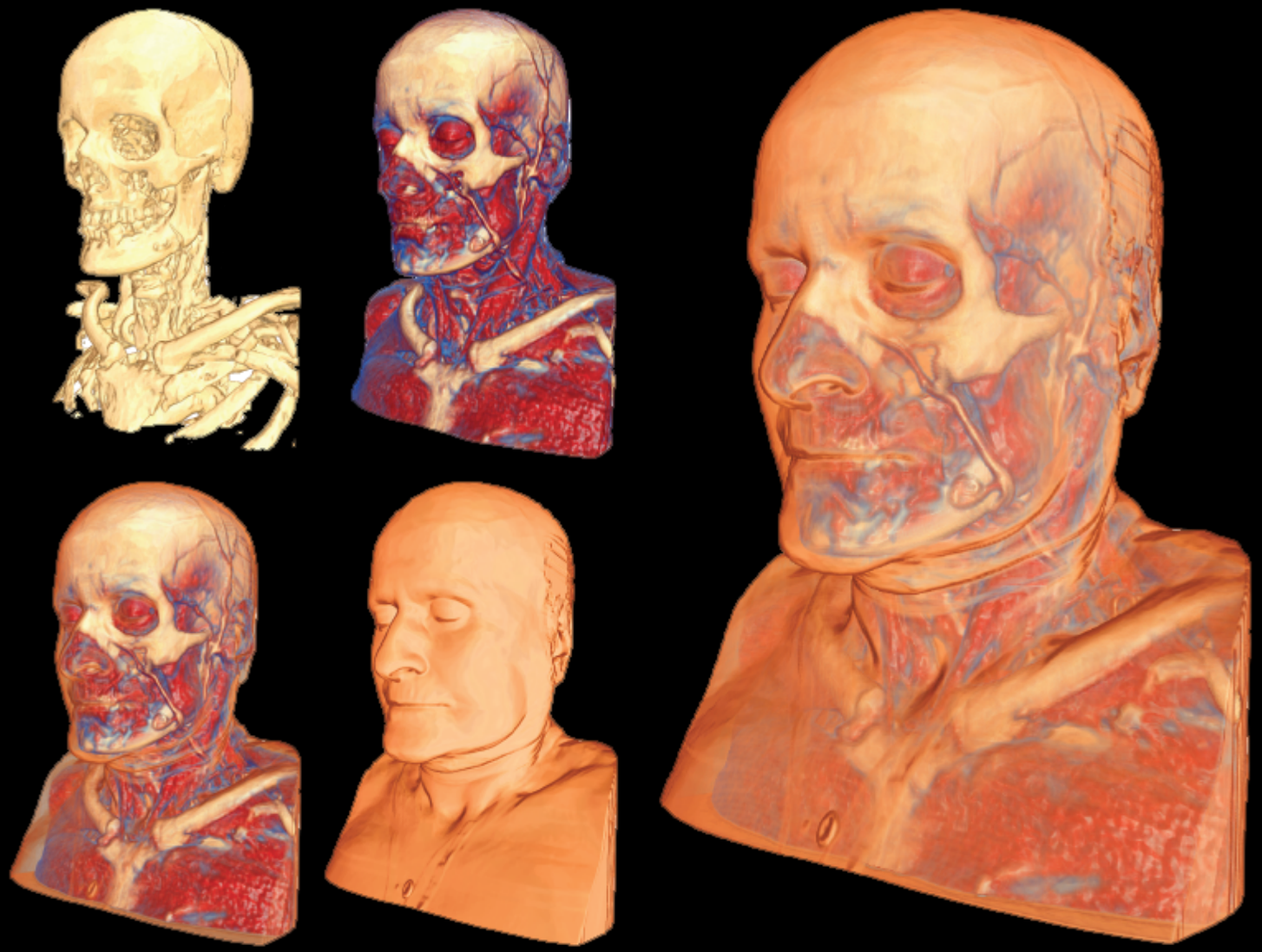


Transparency

- Two kinds: uniform & normal based
  - Uniform: constant over the style
  - Normal-based: each sphere map point has color & alpha

Normal based + curvature info: transparency falloff around transparent objects.

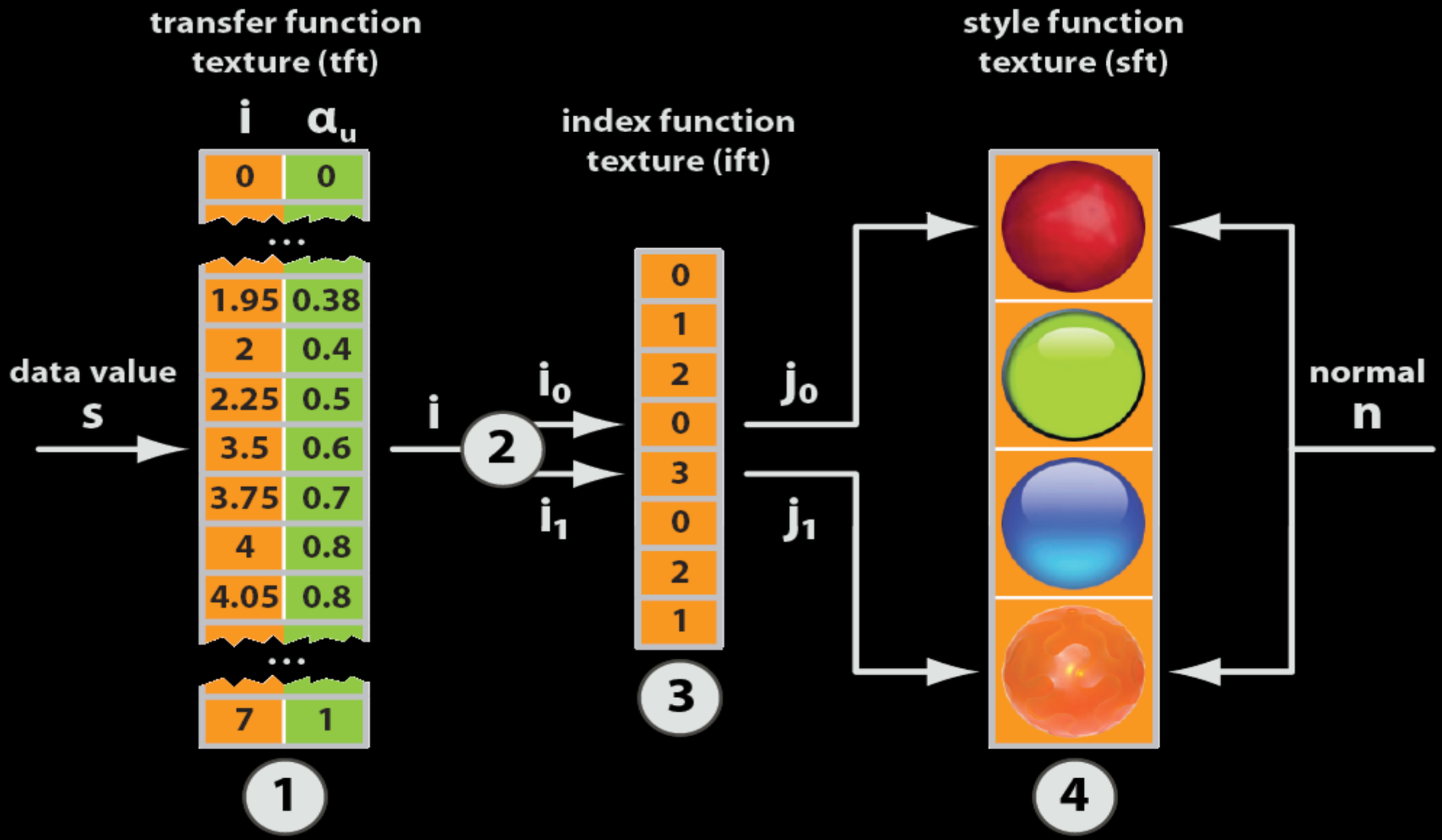
Transparency cont'd



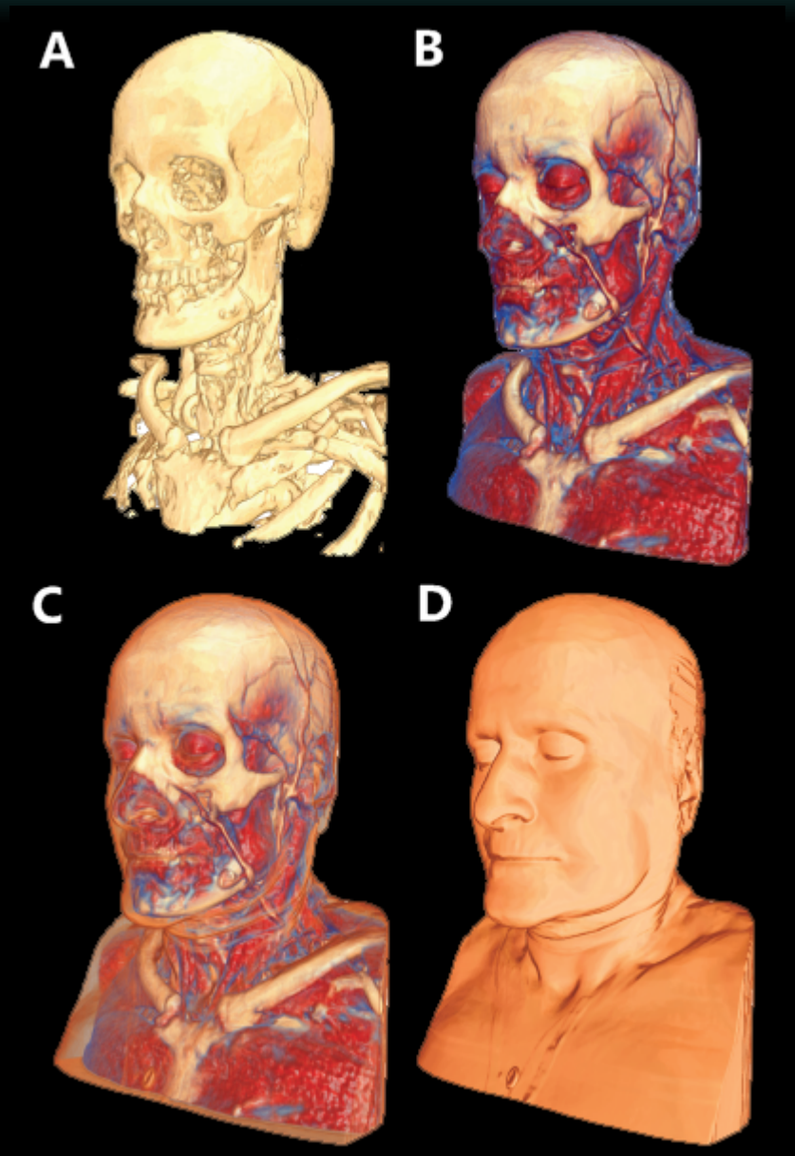
Implementation

- GPU-based implementation
  - Needs Shader Model 3.0
  - C++ / GLSL
  - Simple to integrate inside existing pipelines

Implementation



Performance

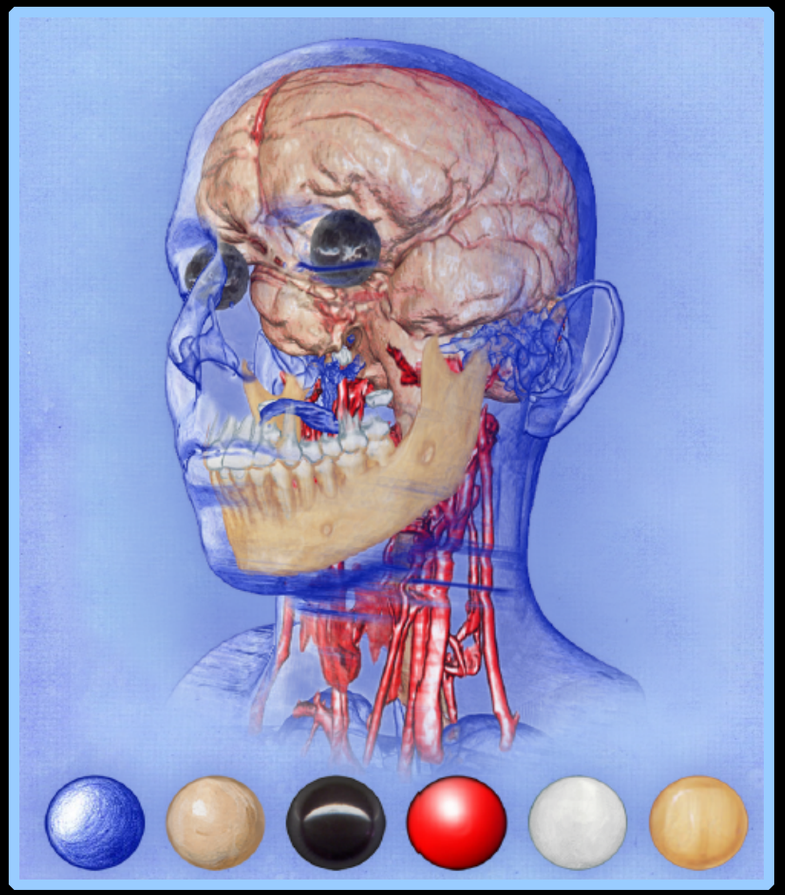


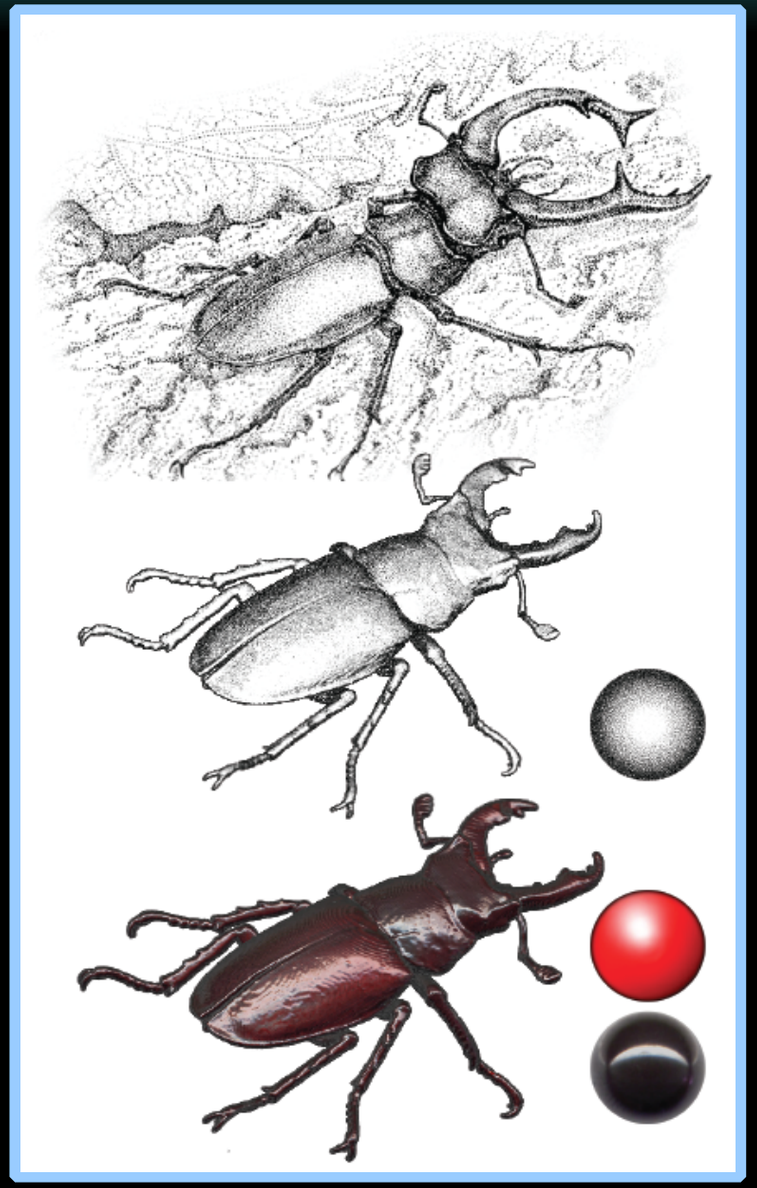
- Dataset size: 256 x 256 x 230
- Viewport size: 512 x 512

Figure	Regular TF	Style TF
A	11.7 fps	11.9 fps
B	10.5 fps	9.6 fps
C	10.1 fps	8.1 fps
D	12.5 fps	12.8 fps

Conclusion

- Style transfer functions work well in reproducing effects classically used in illustrations





- 'Style databases'
- Easy style switching
- Good real time performance
- Styles **cannot** be used as textures

That's All, Folks

Questions?