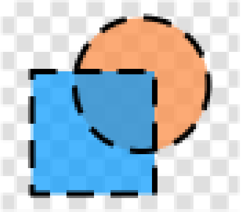
Deep Compositing Using Lie Algebras

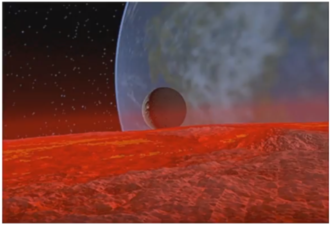
Minliang LIN
SIGGRAPH 2017
Paper Author: Tom Duff

May 8, 2018

Author Info

- Tom Duff
- Classic papaer of **Compositing** *Digital Images*, 1984
- Fully develop the concept of **Alpha Compositing** and **Alpha Channel**



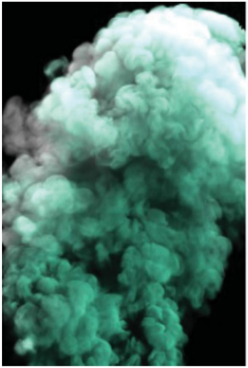


Motivation

- Nowadays, industrial film scene is a **compositing** of many parts.
  - Dividing and conquer
  - Reusing

Deep Compositing

- Considering special objects, e.g. cloud, fog or shadow
- Storing different color for different depth, i.e. **Volumetric Image**

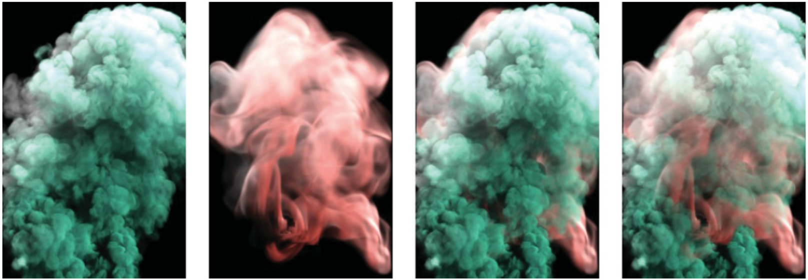


Deep Compositing

- Considering special objects, e.g. cloud, fog or shadow
- Storing different color for different depth, i.e. **Volumetric Image**
- Very useful in industry
    - Three awards from *Oscars SciTech*, 2014
    - Used in OpenEXR file format

Deep Compositing

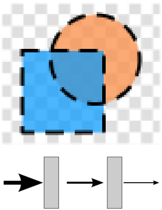
- Considering special objects, e.g. cloud, fog or shadow
- Storing different color for different depth, i.e. **Volumetric Image**



(a) A bluish-green cloud. (b) A reddish-orange cloud. (c) Incorrectly mixed (d) Correctly mixed.

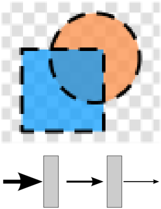
Alpha Compositing Review

- Pixel Value: $A = (a, \alpha), B = (b, \beta)$
  - $a = (R, G, B)$
  - $\alpha = 0$ for transparent
  - $\alpha = 1$ for opaque
- Compositing: A **Over** $B = A + (1 - \alpha)B$



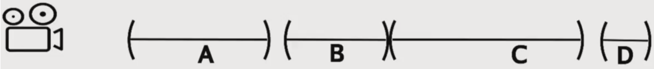
Alpha Compositing Review

- Pixel Value: $A = (a, \alpha), B = (b, \beta)$
  - $a = (R, G, B)$
  - $\alpha = 0$ for transparent
  - $\alpha = 1$ for opaque
- Compositing: A **Over** $B = A + (1 - \alpha)B$
- Object color premultiplied by alpha
- $a = \alpha a_{origin}, b = \beta b_{origin}$
- Not confused with $\cancel{c = \alpha a + (1 - \alpha)\beta b}$.



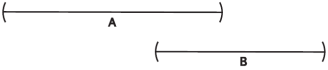
Representing Volumetric Image

- Voxelization is expensive.
- Interval representation
- Assume piecewise constant pixel value in interval

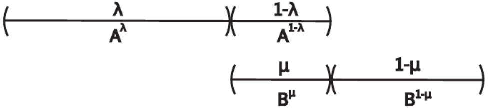


Split And Merge

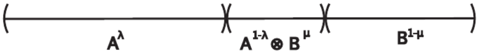
- **Assume** constant optical density
- Split to length $\lambda, 1-\lambda, \mu, 1-\mu$ (relative length)



- Reassign pixel value $A^\lambda, A^{1-\lambda}, B^\mu, B^{1-\mu}$



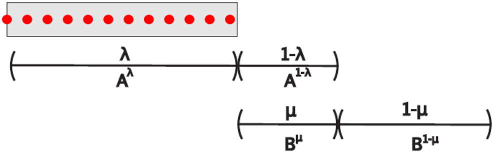
- Merge pixel value on common interval $A^{1-\lambda} \otimes B^\mu$



- Finally, A^λ **Over** $((A^{1-\lambda} \otimes B^\mu)$ **Over** $B^{1-\mu})$

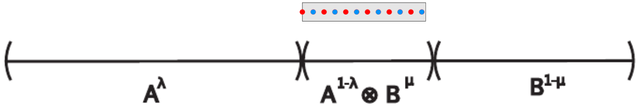
Split Function

- **Assume** constant optical density
- In other words, interval consists of n tranparent unit
- $A^n = \underbrace{A \textbf{ Over } (A \textbf{ Over } \cdots \textbf{ Over } A) \cdots)}_{\text{n times}} = (\sum_{i=0}^{n-1} (1-\alpha)^i)A = \begin{cases} \frac{1-(1-\alpha)^n}{\alpha} A, & \text{if } \alpha \neq 0 \\ nA, & \text{if } \alpha = 0 \end{cases}$
- easily generalized to A^λ



Merging Function

- **Assume** A, B unit occupy the common interval alternately.
- $A \otimes B = \lim_{n\to\infty} \underbrace{(A^{1/n} \textbf{ Over } B^{1/n} \textbf{ Over } \ldots A^{1/n} \textbf{ Over } B^{1/n})}_{\text{n times}} =$
  $\lim_{n\to\infty} (A^{1/n} \textbf{ Over } B^{1/n})^n$
- SymPy: $A \otimes B = \frac{1-(1-\alpha)(1-\beta)}{\log(1-\alpha)+\log(1-\beta)}(\frac{\log(1-\alpha)}{\alpha}A + \frac{\log(1-\beta)}{\beta}B)$



So Far...

- All formulas are known 6 years ago for industry.
- $A^{\lambda} = \frac{1-(1-\alpha)^n}{\alpha} A$, if $\alpha \neq 0$...
- $A \otimes B = \frac{1-(1-\alpha)(1-\beta)}{\log(1-\alpha)+\log(1-\beta)} \left(\frac{\log(1-\alpha)}{\alpha} A + \frac{\log(1-\beta)}{\beta} B \right)$

So Far...

- All formulas are known 6 years ago for industry.
- $A^\lambda = \frac{1-(1-\alpha)^n}{\alpha} A$, if $\alpha \neq 0 \ldots$
- $A \otimes B = \frac{1-(1-\alpha)(1-\beta)}{\log(1-\alpha)+\log(1-\beta)} (\frac{\log(1-\alpha)}{\alpha} A + \frac{\log(1-\beta)}{\beta} B)$
- Using monochrome channel $A = (a, \alpha)$, $a = (R)$ for convenience

$$\exp[\![q,\ p]\!] = \begin{cases} (-\frac{\overline{\exp p}}{p} q,\ \overline{\exp p}), & \text{if } p \neq 0 \\ (q,\ 0), & \text{if } p = 0 \end{cases}$$

$$\log(a,\ \alpha) = \begin{cases} [\![-\frac{\log \overline{\alpha}}{\alpha} a,\ \log \overline{\alpha}]\!], & \text{if } \alpha \neq 0 \\ [\![a,\ 0]\!], & \text{if } \alpha = 0. \end{cases}$$

Correspondence

- A **Over** B is hard to analyze, not commute
- Over operator cannot give a vector space
- $A \otimes B = exp(logA + logB)$
- $A^{\lambda} = exp(\lambda logA)$
- $logA, logB$ are in a **vector space** with addition and scalar multiplication, easy to analyze.

Interpolation

- $interpolation(A, B, t) = exp((1 - t)logA + tlogB)$
- $interpolation(A^n, B^n, t) = interpolation(A, B, t)^n$
- Split invariant

Interpolation

- $interpolation(A, B, t) = exp((1 - t)logA + tlogB)$
- $interpolation(A^n, B^n, t) = interpolation(A, B, t)^n$
- Split invariant

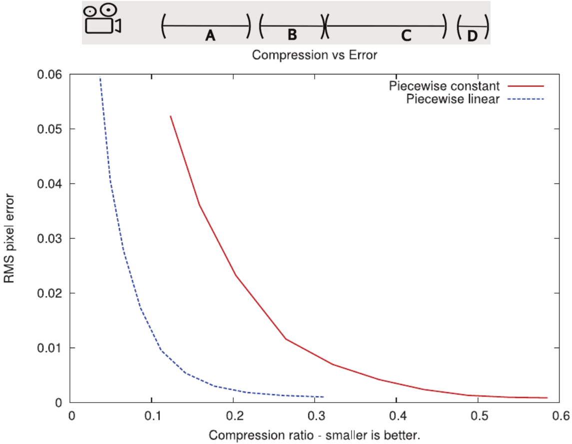
(a) Componentwise linear interpolation

(b) Interpolation using equation (21)

Fig. 3. Pixel value interpolation. Each image is composed of four stripes, subdivided in depth 1, 2, 4, or 8 times (top to bottom) before interpolation. In

Compression

Compare OpenEXR and new approach.



Thanks!