CPSC 4040/6040 Computer Graphics Images

Joshua Levine
levinej@clemson.edu

Lecture 07
Green Screening

Sept. 10, 2015

Agenda

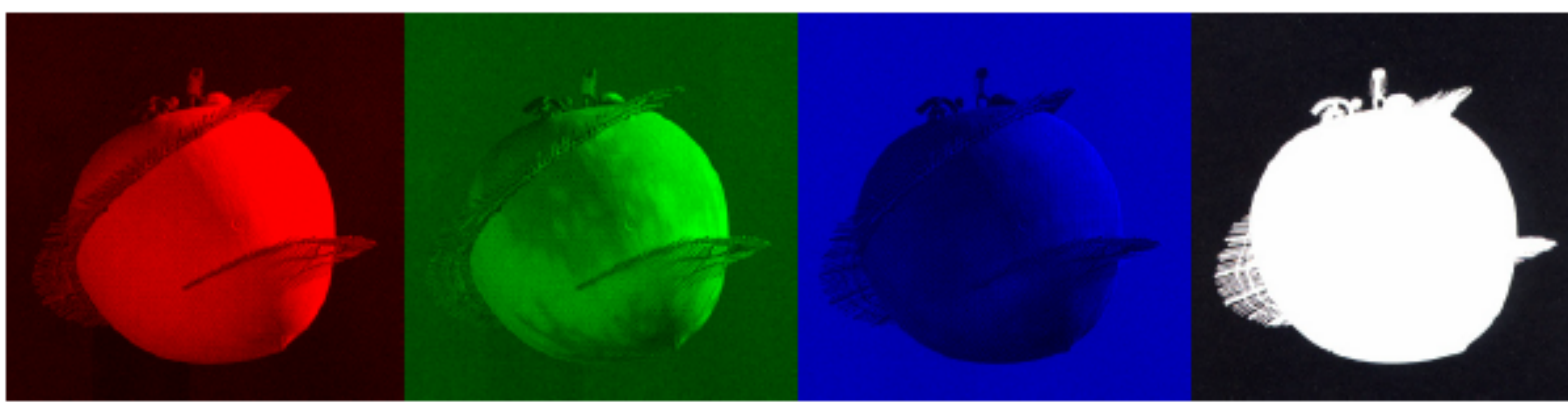
- Quiz01 DUE
- Lab02 Questions?

Recall: Compositing with Alpha

What is $\boldsymbol{\alpha}$ (alpha)?

- $\boldsymbol{\alpha}$ is a measurement of the opacity.
- $\boldsymbol{\alpha}$=1 means fully opaque.
- $\boldsymbol{\alpha}$=0 means fully transparent.

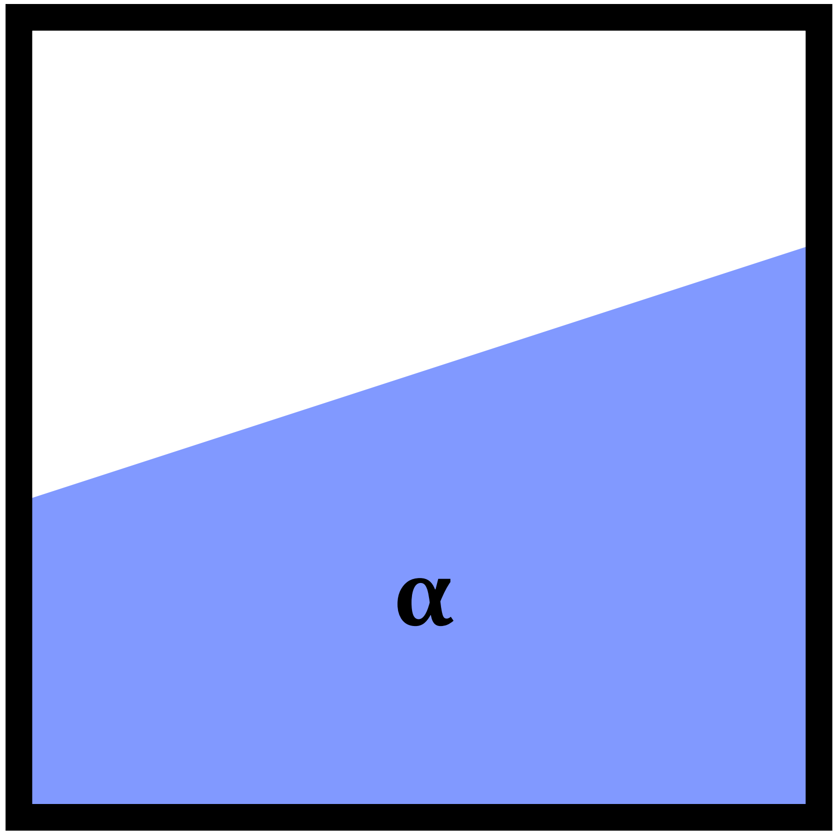
Red Green Blue $\boldsymbol{\alpha}$



The Art and Science of Digital Compositing, Brinkmann

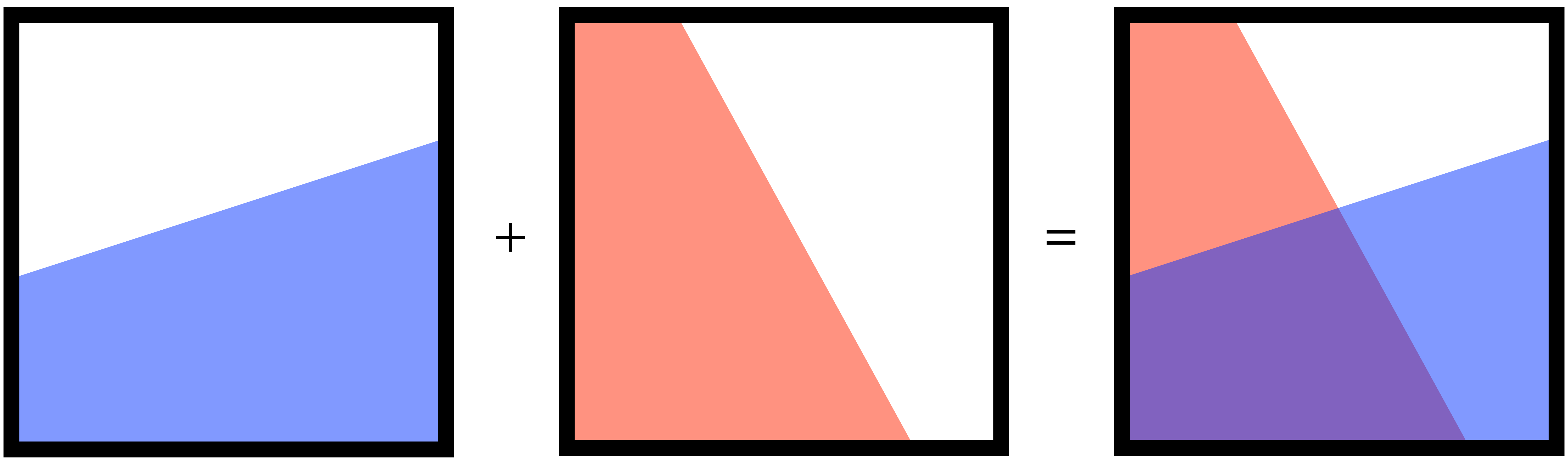
Premultiplying Color

- Problem: have to compute C and $\boldsymbol{\alpha}$ separately
- Instead, use a **premultiplied** color, $c = \boldsymbol{\alpha}C$
- Idea: alpha is some percentage of (subpixel) color
- $\boldsymbol{\alpha}$ = area of blue = pixel coverage
- Convention: use lowercase letters for premultiplied colors and UPPERCASE for normal



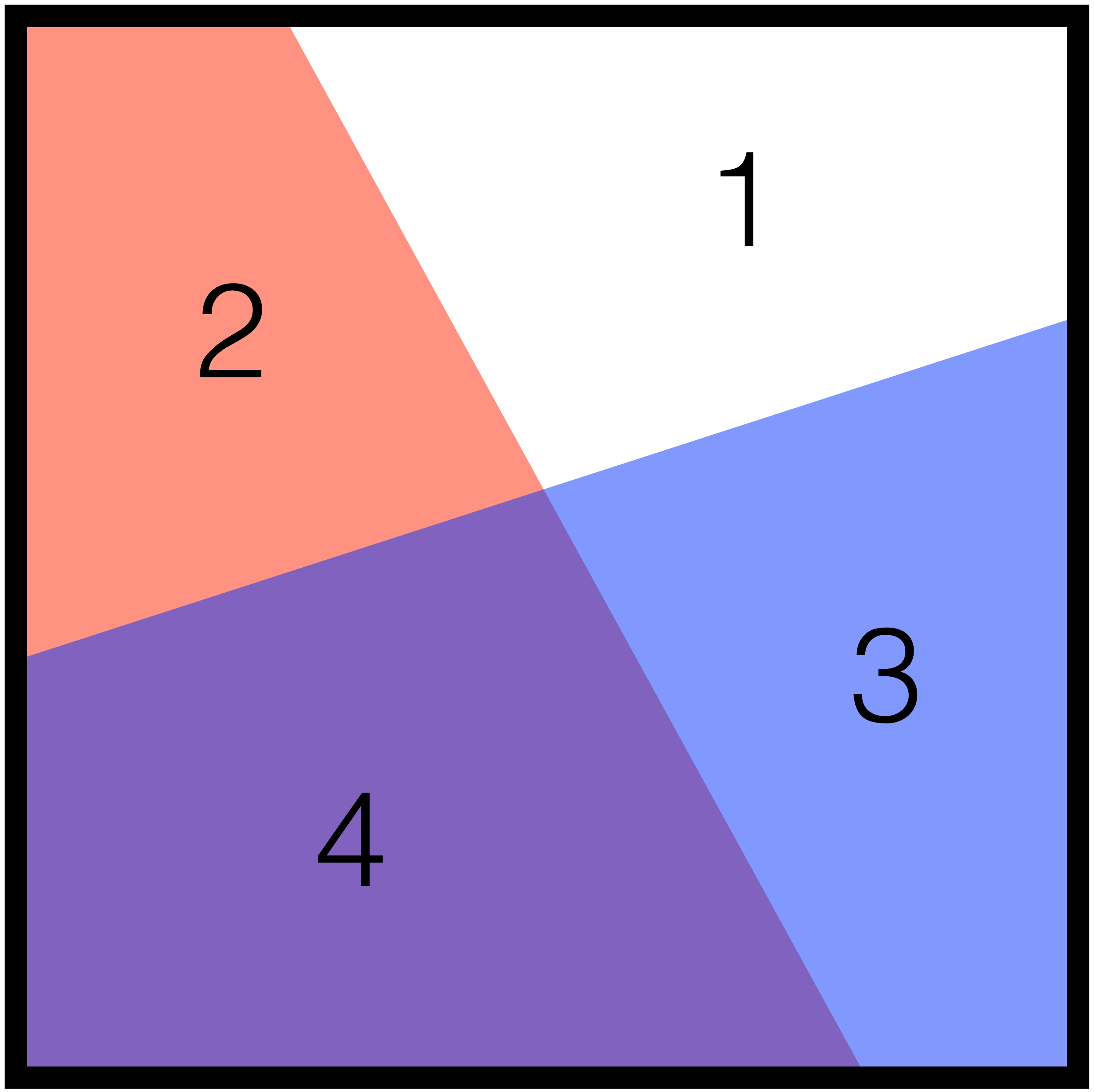
A Pixel

Operations on Associated Colors



A B 4 regions

Porter-Duff Composition

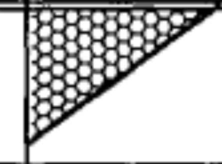
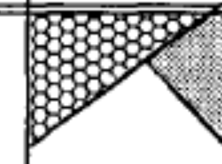
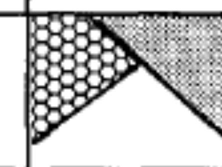
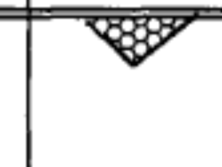
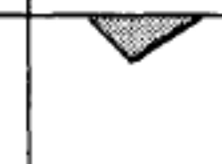
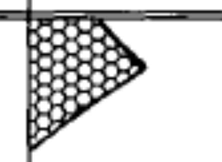
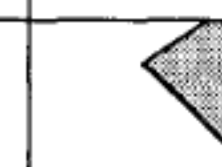
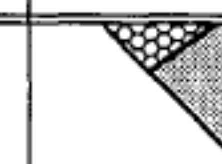
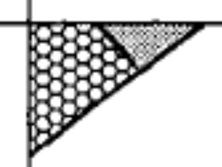
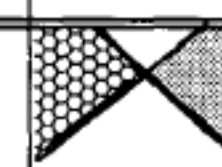


4 regions

- Region 1: 1 possibility - 0
- Region 2: 2 possibilities - A or 0
- Region 3: 2 possibilities - B or 0
- Region 4: 3 possibilities - A, B or 0
- Operators: 12 total possibilities

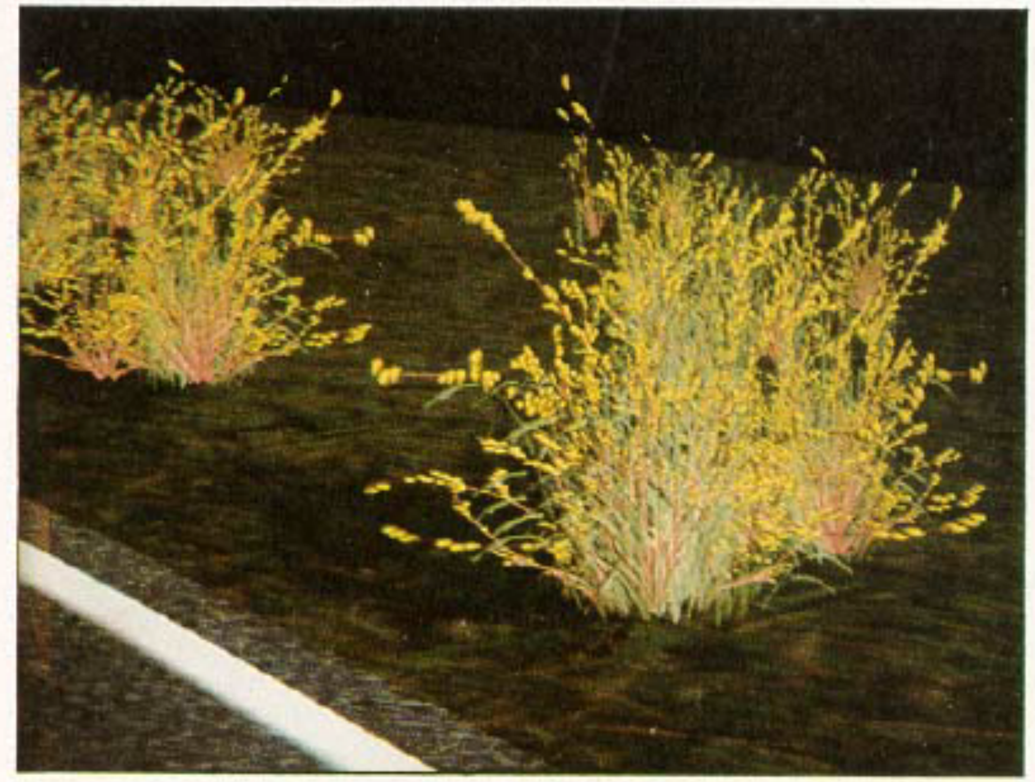
12 Operators

- $c_P = F_A c_A + F_B c_B$
- Various F functions dictate how to blend
- Always assumes color-associated values
- What commonly used operation does **in** do?

operation	quadruple	diagram	F_A	F_B
clear	(0,0,0,0)		0	0
A	(0,A,0,A)		1	0
B	(0,0,B,B)		0	1
A **over** *B*	(0,A,B,A)		1	$1-\alpha_A$
B **over** *A*	(0,A,B,B)		$1-\alpha_B$	1
A **in** *B*	(0,0,0,A)		α_B	0
B **in** *A*	(0,0,0,B)		0	α_A
A **out** *B*	(0,A,0,0)		$1-\alpha_B$	0
B **out** *A*	(0,0,B,0)		0	$1-\alpha_A$
A **atop** *B*	(0,0,B,A)		α_B	$1-\alpha_A$
B **atop** *A*	(0,A,0,B)		$1-\alpha_B$	α_A
A **xor** *B*	(0,A,B,0)		$1-\alpha_B$	$1-\alpha_A$

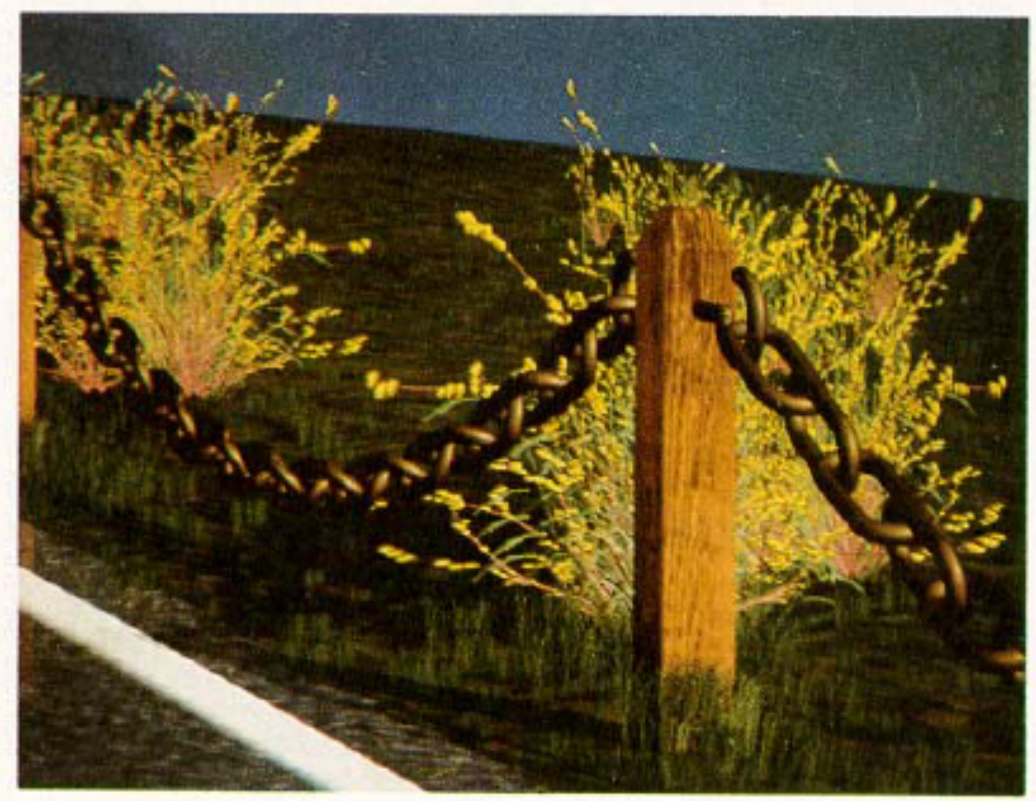
Porter & Duff, 1984

Foreground = *FrgdGrass* **over** *Rock* **over** *Fence* **over** *Shadow* **over** *BkgdGrass*;



Hillside = *Plant* **over** *GlossyRoad* **over** *Hill*;

Background = *Rainbow* **plus** *Darkbow* **over** *Mountains* **over** *Sky*;



Pt.Reyes = *Foreground* **over** *Hillside* **over** *Background*.

Porter & Duff, 1984

Limitations of Relying on α for Compositing

- Hard to represent certain types of transparency, like stained glass
    - Focus only on the subpixel occlusion (of all colors)
- Does not model more complex optical effects (magnifying glasses, lighting, etc.)
- Size of individual pixels

Cinefex

Green Screening

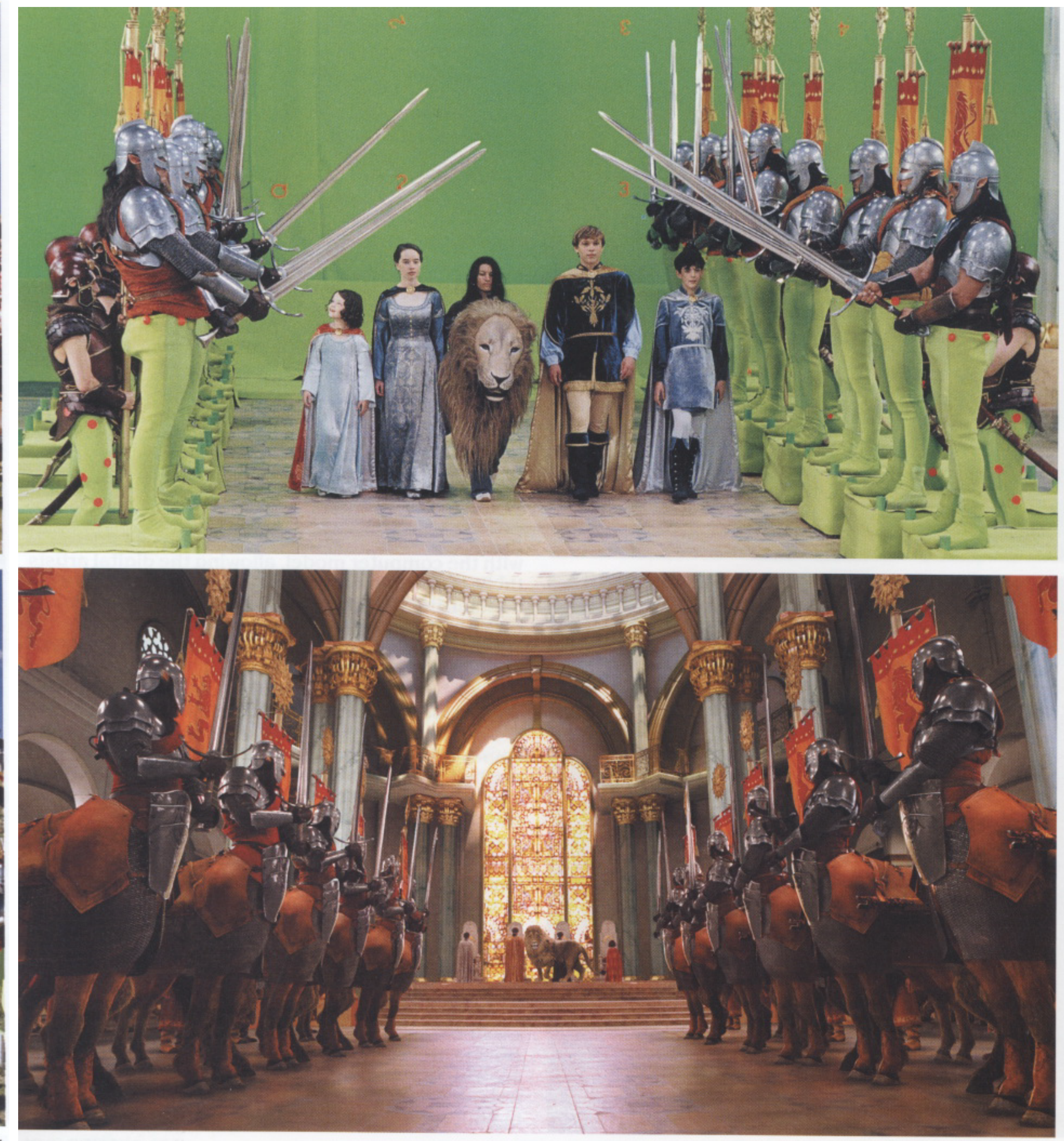
Example: Education/TV/Weather

Example: Film

Cinefex

Example: Film



Cinefex

Example: Colbert Challenge

http://en.wikipedia.org/wiki/The_Colbert_Report#Green_Screen_Challenge

Problem: “Pulling a Matte”

- Problem definition, Separate an image C into:
  1. A foreground object image C_F,
  2. a background image C_B,
  3. and an alpha matte $\boldsymbol{\alpha}$
  - C_F and $\boldsymbol{\alpha}$ can then be used to composite the foreground object into a different image
- This is a HARD problem
  - Even if you only require $\boldsymbol{\alpha}$ to be binary, this is hard to do automatically (why?)
  - For movies/TV, manual segmentation of each frame is infeasible
  - Need to do something procedural (or at least semi-automatic)

Some Simple Solutions

Idea #1: Luma Keying

- If the object is significantly brighter/darker, could look for high luminance (e.g. Y in xyY color space, V in HSV, etc.)



The Art and Science of Digital Compositing, Brinkmann

Idea #1: Luma Keying

- Usually, one needs to modify $\boldsymbol{\alpha}$ to be a function of luminance, it cannot be used directly

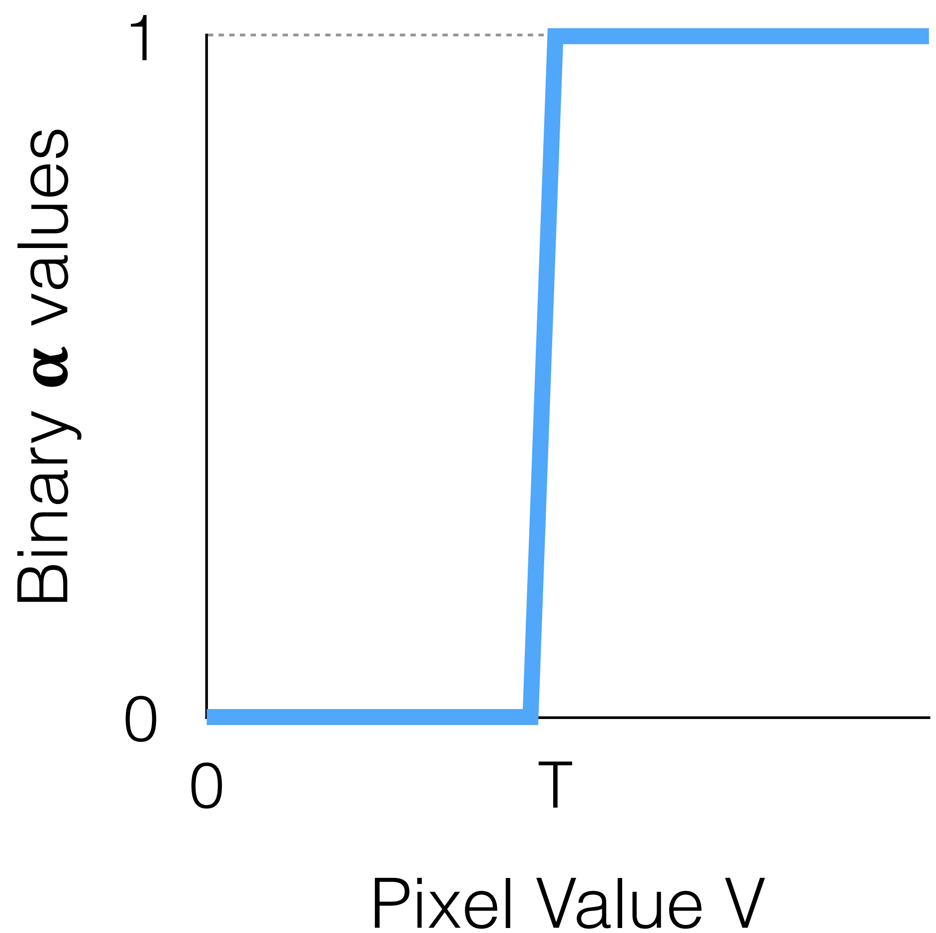
The Art and Science of Digital Compositing, Brinkmann

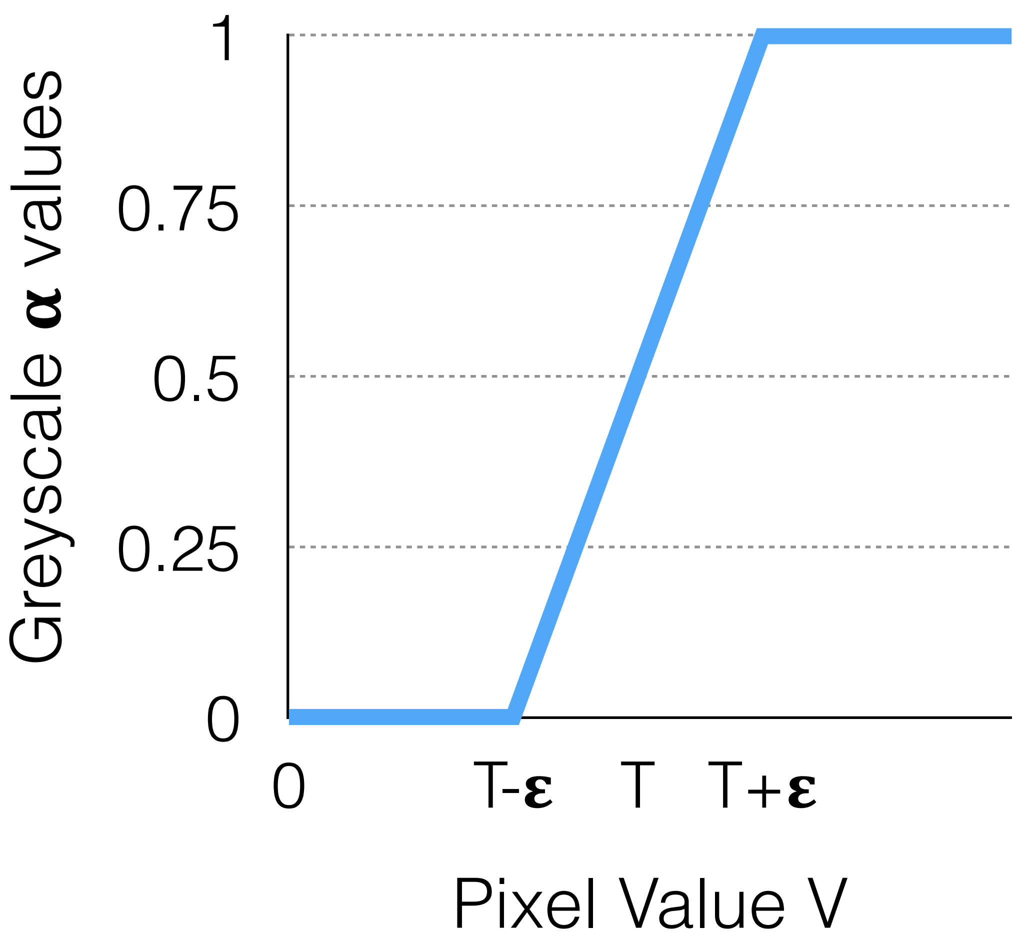
Modifying Luma-Keying

- Instead of a binary matte, can use a smoother function to define **α** based on a threshold T and a window size **ε**:

$$\alpha = \begin{cases} 1, & V > T \\ 0, & \text{otherwise} \end{cases}$$

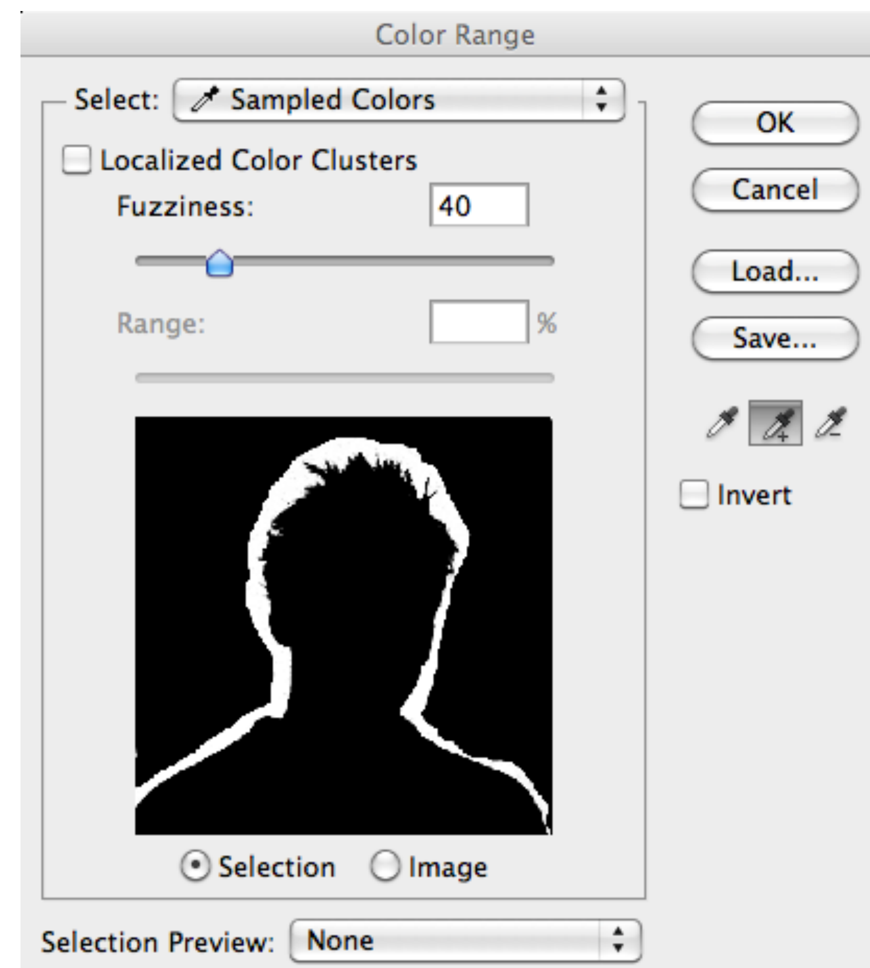
$$\alpha = \begin{cases} 1, & V > T + \varepsilon \\ \frac{V-(T-\varepsilon)}{2\varepsilon}, & T - \varepsilon < V \leq T + \varepsilon \\ 0, & \text{otherwise} \end{cases}$$





Garbage Matting

- Quickly generated, approximate mattes can help distinguish what to include/exclude spatially.



Garbage Matting

- Quickly generated, approximate mattes can help distinguish what to include/exclude spatially.

Lumakey

Garbage Matte

Lumakey + Garbage

Composite

The Art and Science of Digital Compositing, Brinkmann

Comparison

Lumakey

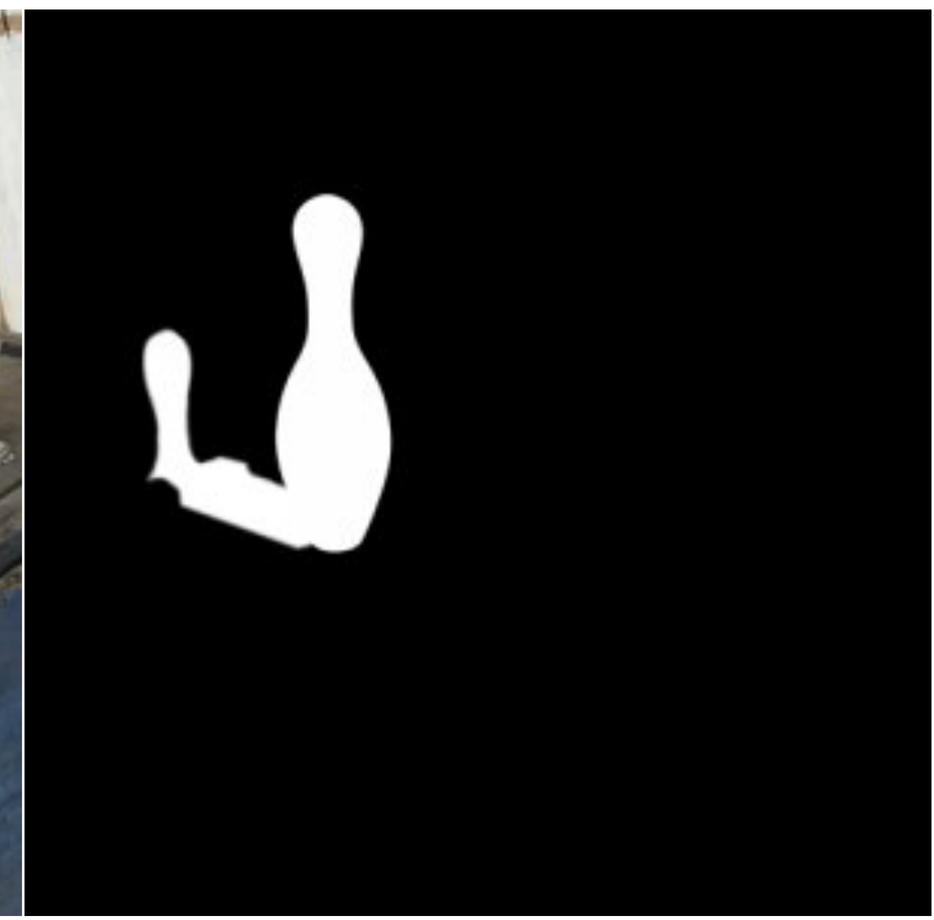
Lumakey + Garbage

The Art and Science of Digital Compositing, Brinkmann

Idea #2: Difference Matte

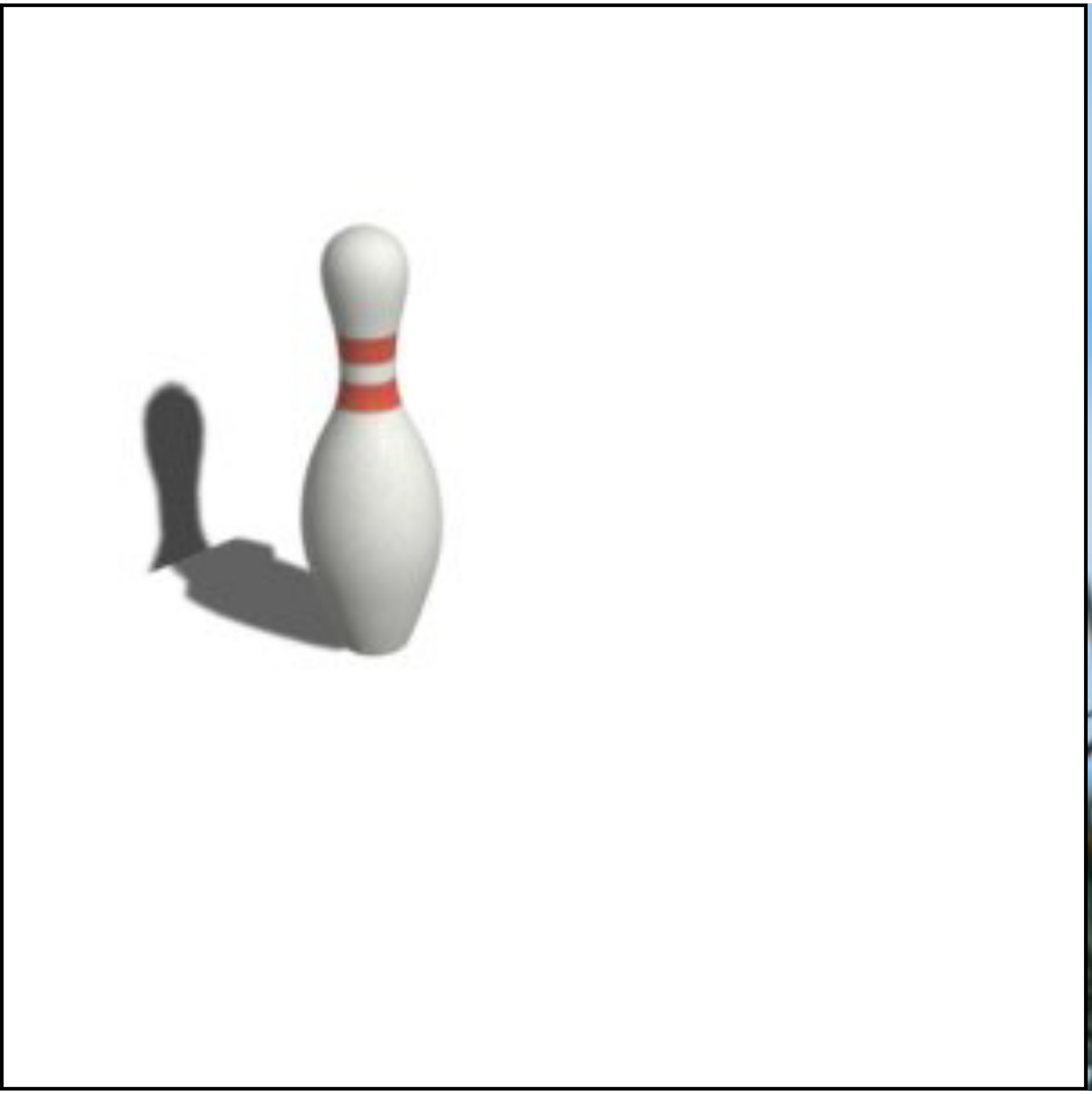
- Shoot the scene twice, with and without the object, then subtract





Difference Matte

Difference Matte: Example

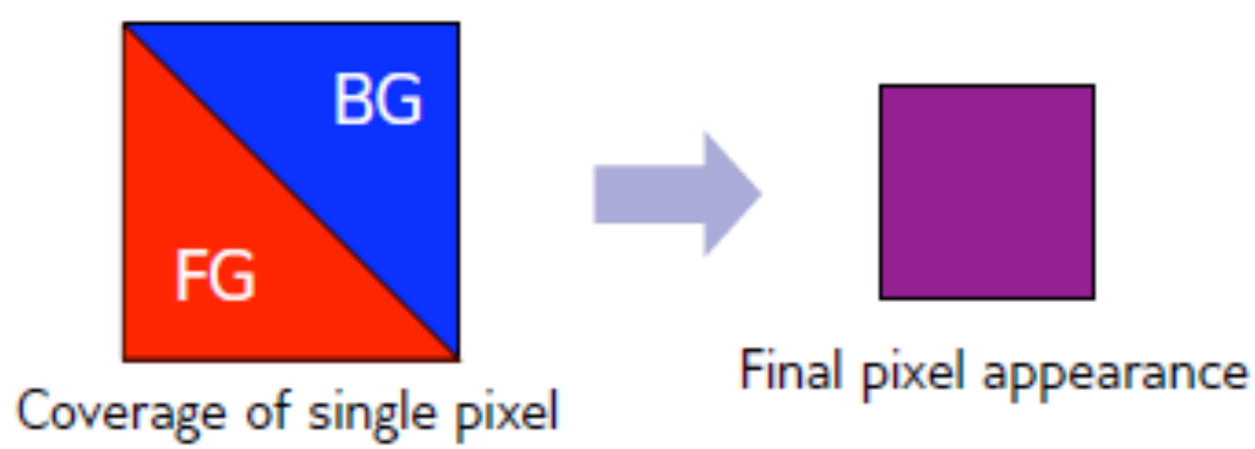


Matte Applied to Original

Composite

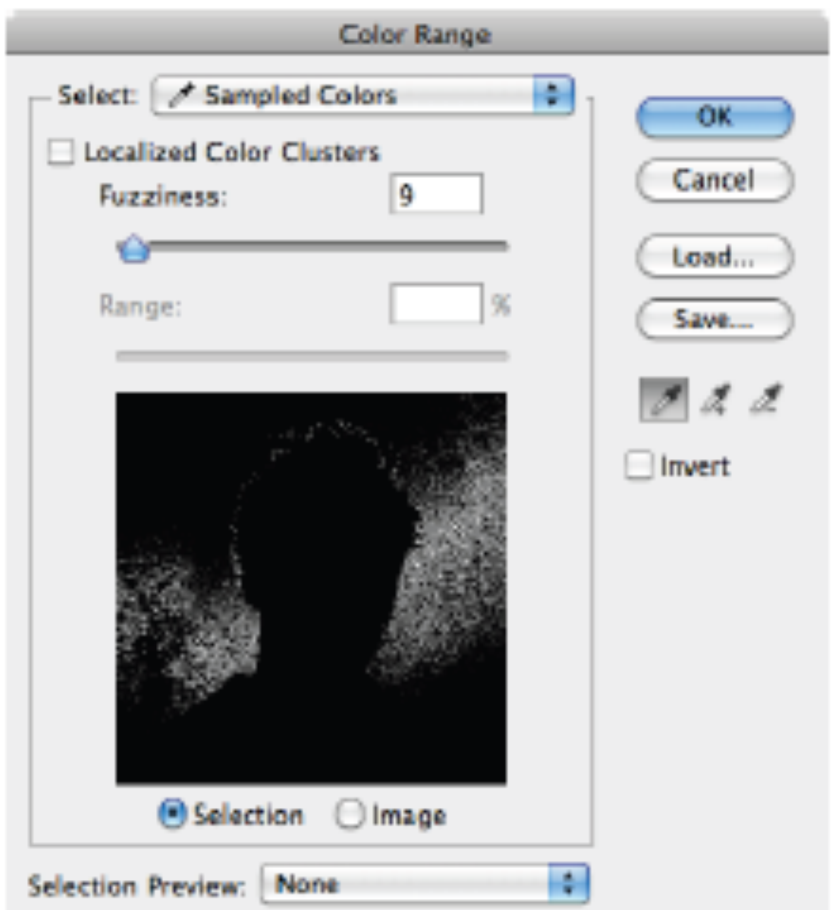
Idea #3: Chroma-Keying

- If the object has a different spectrum of color (H in HSV color space) then background.
- Ideally, using a range of H or H+S, etc. (important for Lab03!)
- Basis of Blue/Green Screening
  - why blue?, why green?, why not red?
- What about partial coverage in pixels?

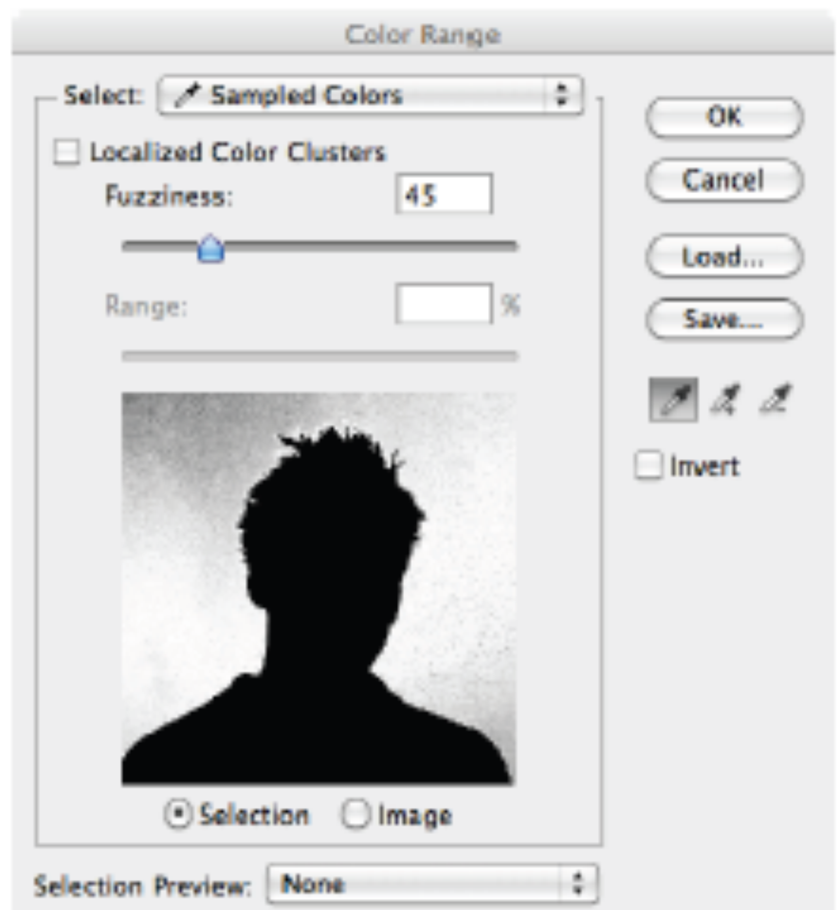


Chroma-Keying is sensitive to the range of colors

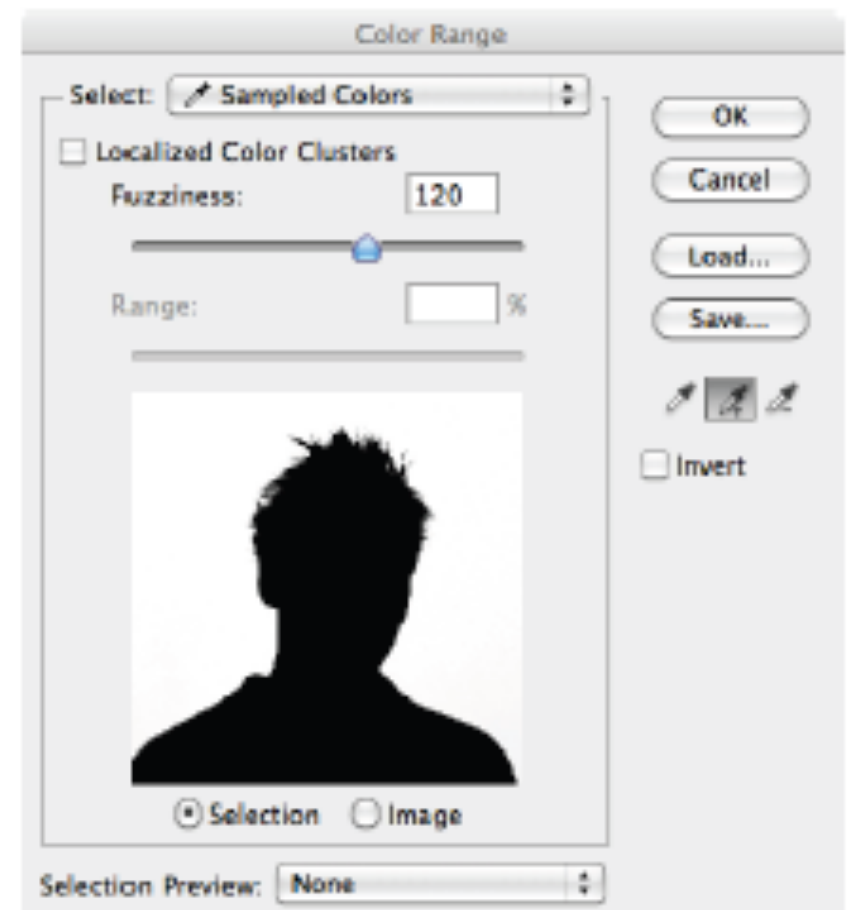
Fuzziness = 9



Fuzziness = 45



Fuzziness = 120



Chroma keying – color match too wide

Petro Vlahos Method

Math of Matte Extraction

- Given an image C, assume it is a composite of a foreground C_F and background $C_{B.}$

- Recall the compositing **over** equation (assuming background has $\boldsymbol{\alpha}_B$=1): C = C_F **over** C_B = $\boldsymbol{\alpha}_F C_F + (1-\boldsymbol{\alpha}_F)C_B$

$$R = (\boldsymbol{\alpha}_F R_F) + (1-\boldsymbol{\alpha}_F)R_B$$

$$G = (\boldsymbol{\alpha}_F G_F) + (1-\boldsymbol{\alpha}_F)G_B$$

$$B = (\boldsymbol{\alpha}_F B_F) + (1-\boldsymbol{\alpha}_F)B_B$$

- Knowns (R,G,B) and (R_B, G_B, B_B) Unknowns (R_F, G_F, B_F, $\boldsymbol{\alpha}_F$)

- 3 equations in 4 unknowns!

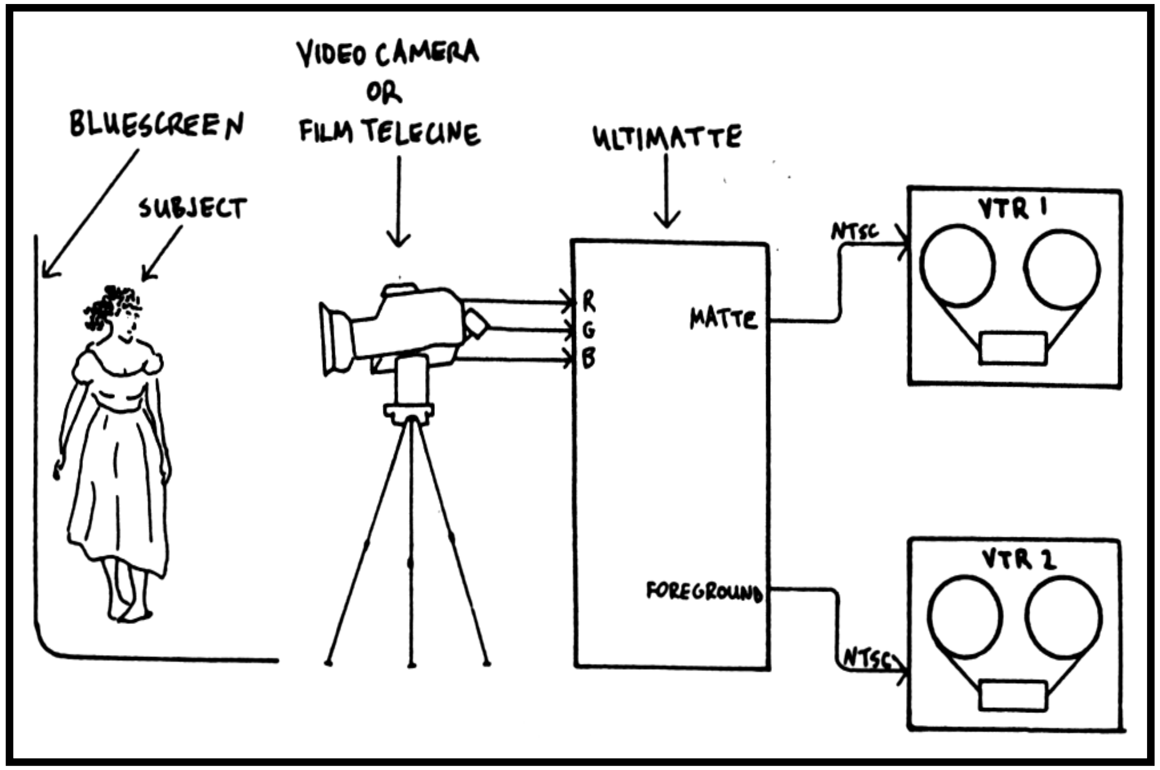
Traditional Blue Screen Matting

- Invented by Petro Vlahos in the 50s (Won Oscar in 1964)
- Initially for film, then video, then digital
- Assume that the foreground has no blue
- Note that computation of $\boldsymbol{\alpha}$ was originally done in analog, required a simple solution.

Petro Vlahos
GORDON E. SAWYER AWARD
66TH ACADEMY AWARDS
1993

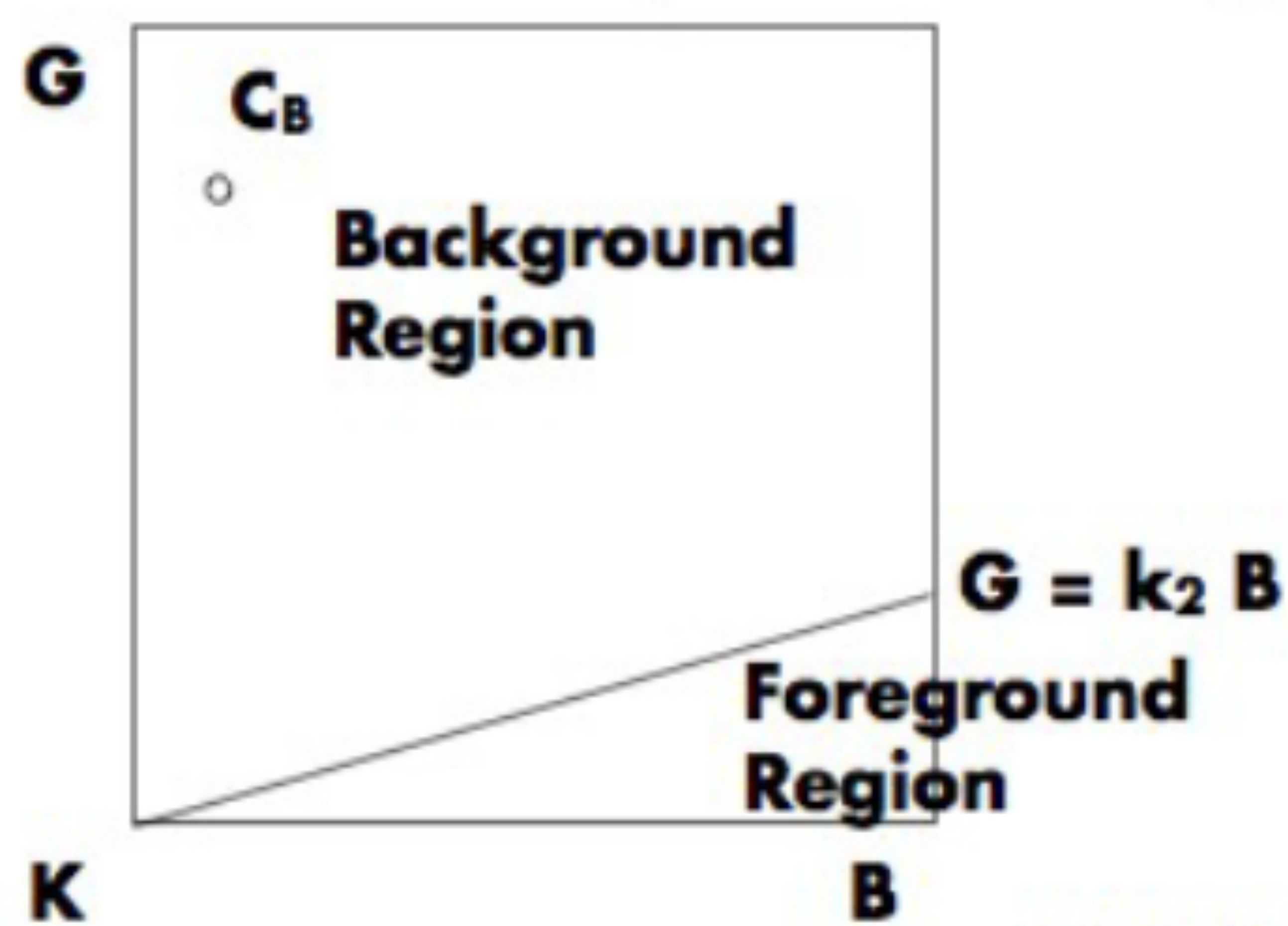
Cinefex

The Ultimatte
(Technical Academy Award 1995)



Petro Vlahos Algorithm

- Idea: Divide the color space into two regions
- Separating line at $G = k_2 B$ (for greenscreen)
- Colors with high green are background
- $0.5 < k_2 < 1.5$ (why?)

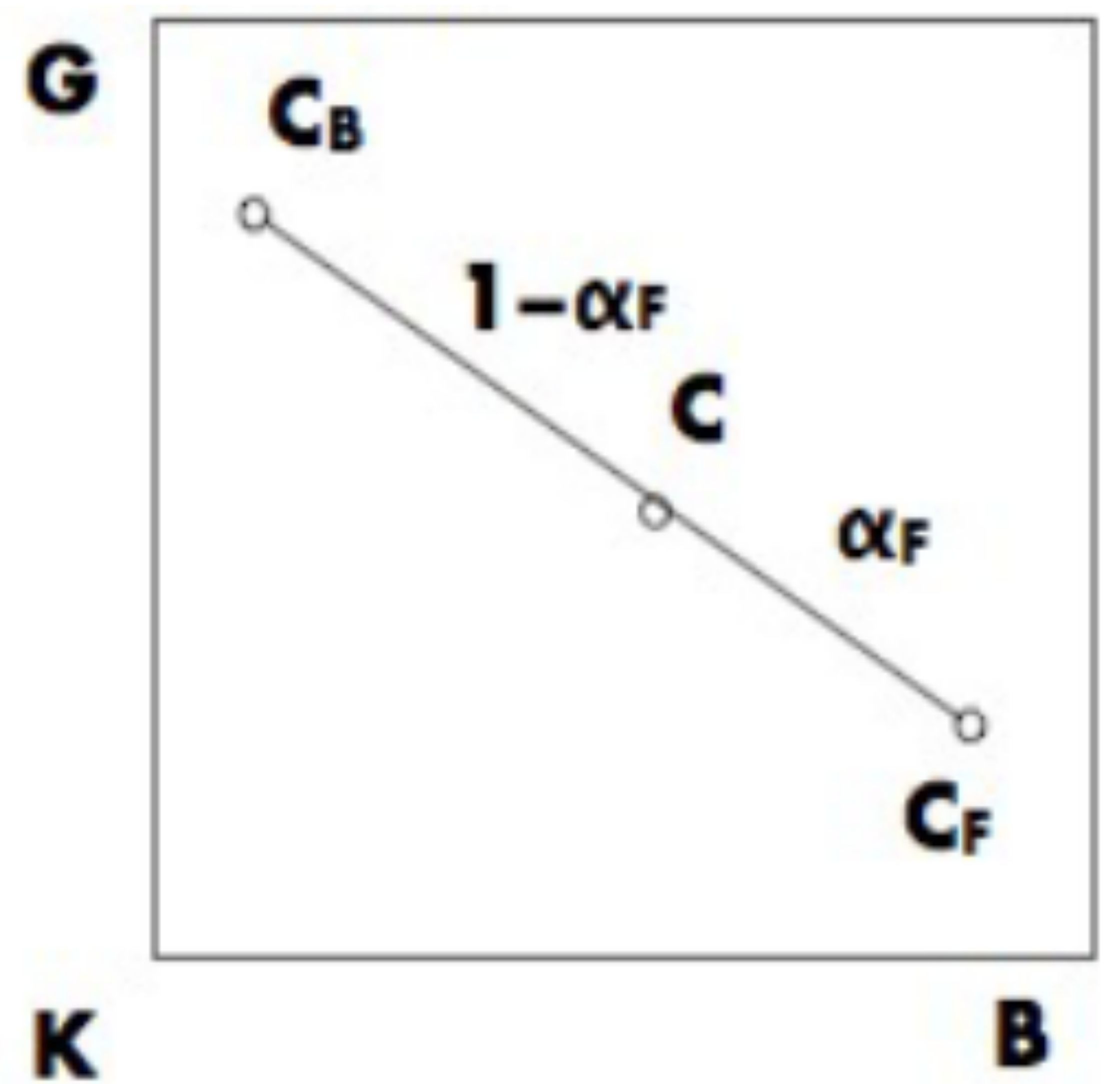


Petro Vlahos Algorithm

- Any background color C has been composited with C_B, this means C is on some line between C_B and C_F

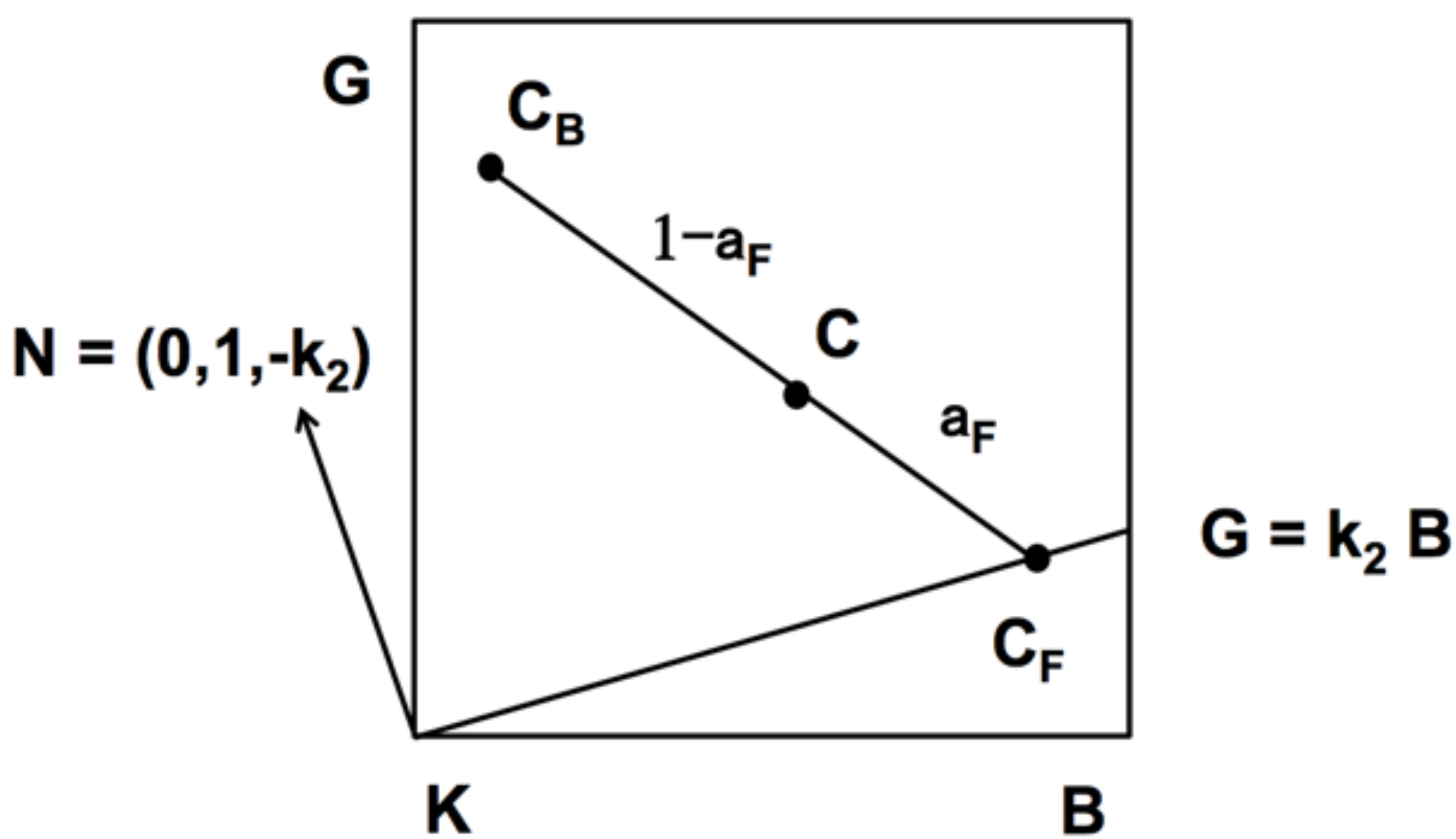
  $C = \boldsymbol{\alpha}_F C_F + (1- \boldsymbol{\alpha}_F)C_B \Rightarrow c_F = C - (1- \boldsymbol{\alpha}_F)C_B$

- But we don't know $\boldsymbol{\alpha}_F$!



Petro Vlahos Algorithm

- Pure background colors are assumed to be above the plane $G = k_2B$
- This fourth equation allows us to solve for the four unknowns.



Blue/Green Screen Matting Issues

- Color limitations
  - Annoying for blue-eyed people
  - So adapt screen color (in particular green)
- Blue/Green spilling and reflection
  - The background illuminates the foreground, blue/green at silhouettes
  - Modify blue/green channel, e.g. set to min(b, $\boldsymbol{\alpha}_{G}$)
- Shadows
  - How to extract shadows cast on background?

Blue/Green Screen Matting

The Art and Science of Digital Compositing, Brinkmann

Blue/Green Screen Matting Issues

Figure 13.13c A composite where the foreground element has had spill-suppression applied before compositing.

Figure 13.14 The foreground object photographed in the actual scene (not a composite).

The Art and Science of Digital Compositing, Brinkmann

Blue/Green Screen Matting

Figure 3. *Firefox Blue Spill Matte Series 1, original shot. Note blue reflected on wing surfaces from bluescreen -- undesirable but unavoidable on such surfaces.*

http://www.digitalgreenscreen.com/figure3.html

Suppressing Blue Spill

B = min(R, G, B)

Suppressing Blue Spill

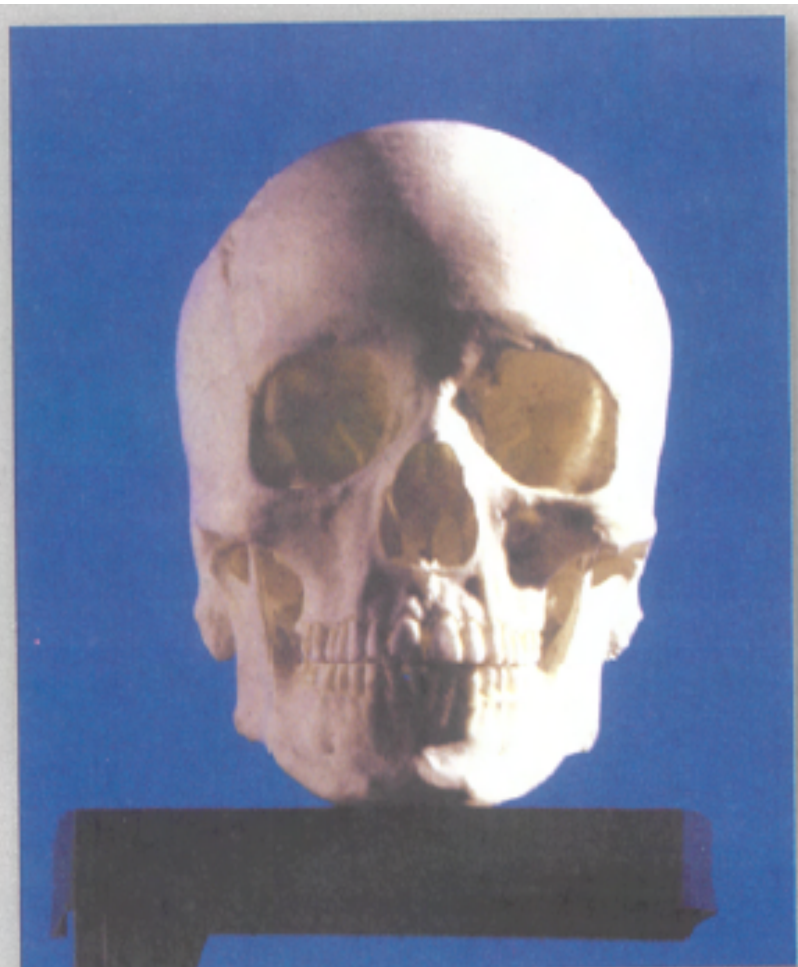


Plate 34 *Original bluescreen image.*

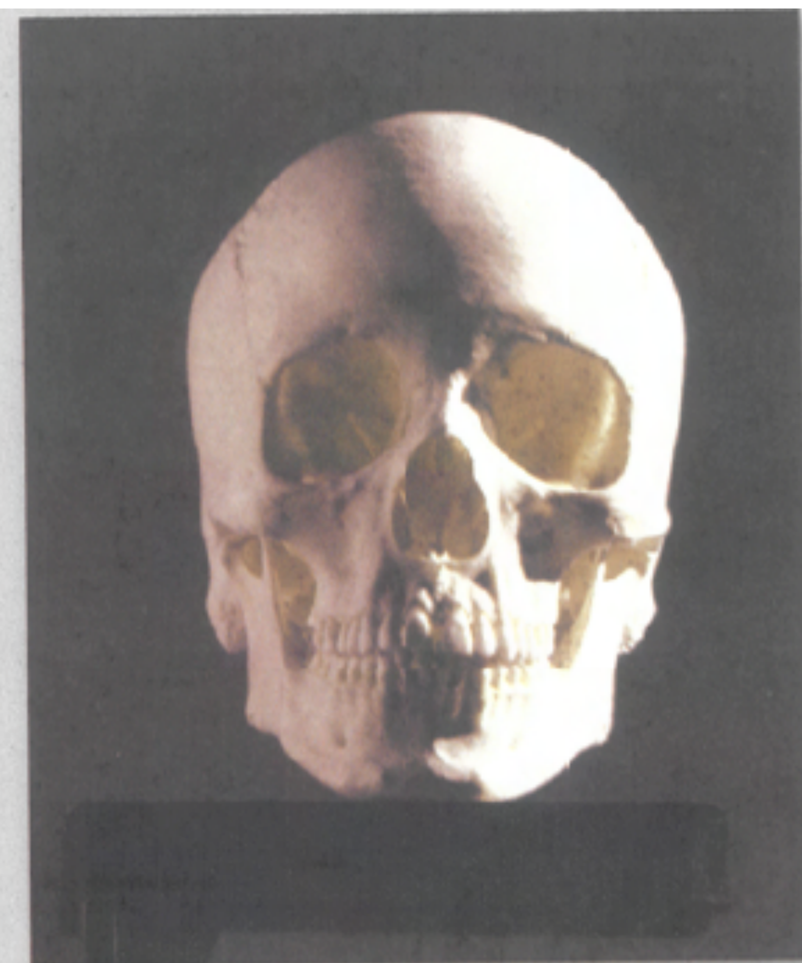


Plate 35 *The image from Plate 34 after applying spill suppression.*

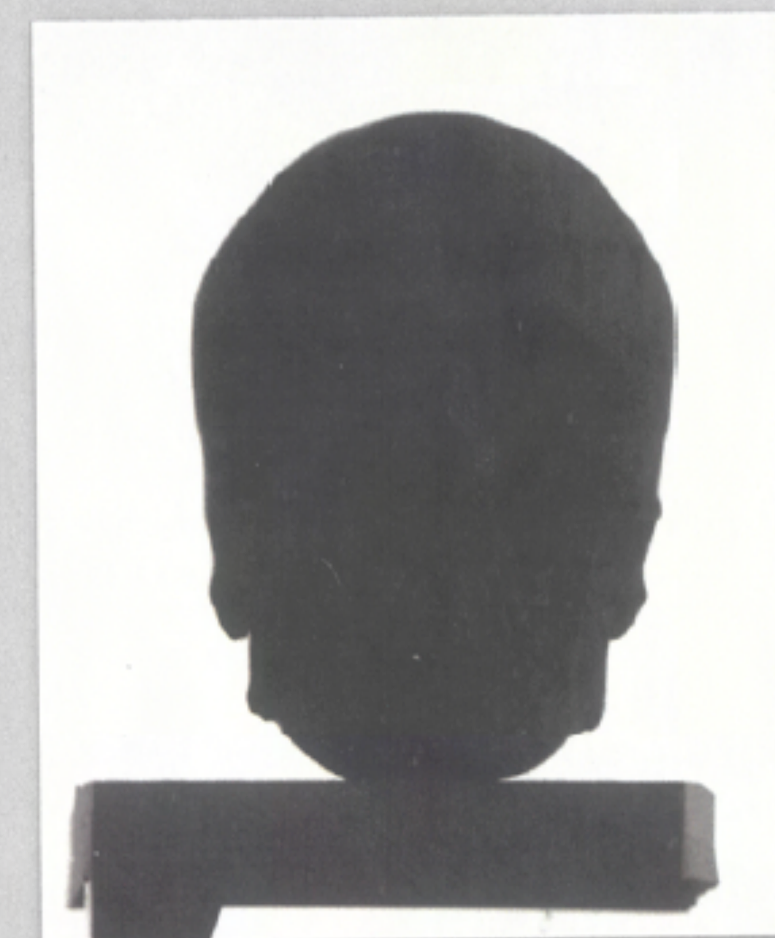


Plate 36 *Plate 34 after matte extraction (reverse matte).*

If (B > kG) B = G

Matte = B - max(R,G)

α = 1 - Matte

The Art and Science of Digital Compositing (1st Edition), Brinkmann

Smith-Blinn Formalization

Blue Screen Matting

Alvy Ray Smith and James F. Blinn[1]
Microsoft Corporation

ABSTRACT

A classical problem of imaging—the *matting problem*—is separation of a non-rectangular foreground image from a (usually) rectangular background image—for example, in a film frame, extraction of an actor from a background scene to allow substitution of a different background. Of the several attacks on this difficult and persistent problem, we discuss here only the special case of separating a desired foreground image from a background of a constant, or almost constant, *backing* color. This backing color has often been blue, so the problem, and its solution, have been called *blue screen matting*. However, other backing colors, such as yellow or (increasingly) green, have also been used, so we often generalize to *constant color matting*. The mathematics of constant color matting is presented and proven to be unsolvable as generally practiced. This, of course, flies in the face of the fact that the technique is commonly used in film and video, so we demonstrate constraints on the general problem that lead to solutions, or at least significantly prune the search space of solutions. We shall also demonstrate that an algorithmic solution is possible

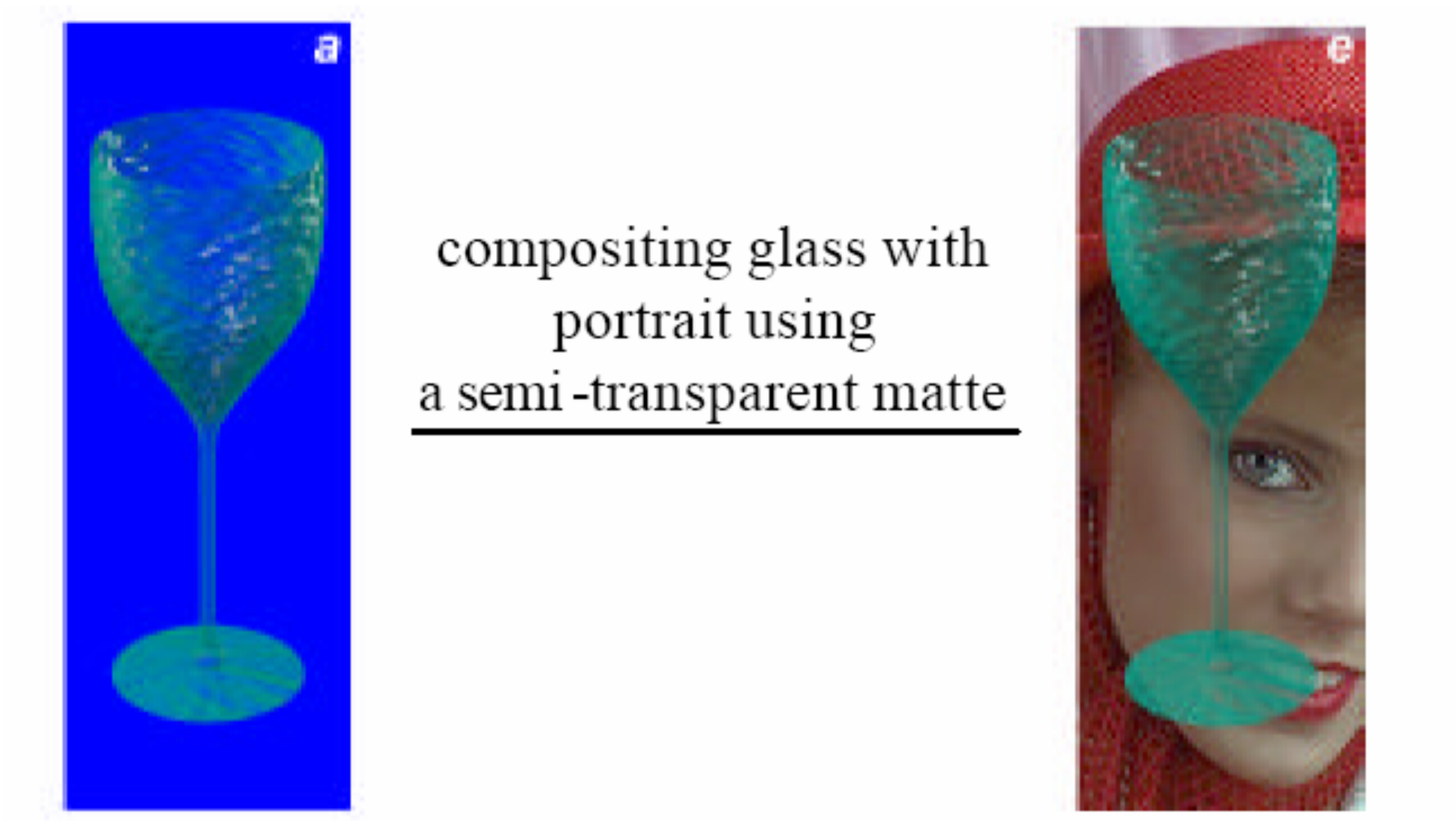
the color film image that is being matted is partially illuminated.

The use of an *alpha channel* to form arbitrary compositions of images is well-known in computer graphics [9]. An alpha channel gives shape and transparency to a color image. It is the digital equivalent of a holdout matte—a grayscale channel that has full value pixels (for opaque) at corresponding pixels in the color image that are to be seen, and zero valued pixels (for transparent) at corresponding color pixels not to be seen. We shall use 1 and 0 to represent these two alpha values, respectively, although a typical 8-bit implementation of an alpha channel would use 255 and 0. Fractional alphas represent pixels in the color image with partial transparency.

We shall use "alpha channel" and "matte" interchangeably, it being understood that it is really the holdout matte that is the analog of the alpha channel.

The video industry often uses the terms "key" and "keying"—as in "chromakeying"—rather than the "matte" and "matting" of the film industry. We shall consistently use the film terminology, after first pointing out that "chromakey" has now taken on a more sophisticated meaning (e.g., [8]) than it originally had (e.g., [19]).

Semi-Transparent Mattes



- What we really want is to obtain a true α matte, which involves semi-transparency
  - α between 0 and 1

Big Picture Idea: Matting is Ill-posed

- An infinite number of solutions exist to the matting equations:

$$R = (\boldsymbol{\alpha}_F R_F) + (1-\boldsymbol{\alpha}_F)R_B$$

$$G = (\boldsymbol{\alpha}_F G_F) + (1-\boldsymbol{\alpha}_F)G_B$$

$$B = (\boldsymbol{\alpha}_F B_F) + (1-\boldsymbol{\alpha}_F)B_B$$

- Four unknowns (R_F, G_F, B_F, $\boldsymbol{\alpha}_F$), but 3 equations
- Need to the problem somehow:
  - e.g. Petro Vlahos approach: $G = kB$, but could we constrain the system otherwise?

Simple Constraint: Assume Foreground has no Green

- In this case, $G_F = 0$, so the equations change to:

  $R = (\boldsymbol{\alpha}_F R_F) + (1-\boldsymbol{\alpha}_F)R_B$

  $G = G_B - \boldsymbol{\alpha}_F G_B$

  $B = (\boldsymbol{\alpha}_F B_F) + (1-\boldsymbol{\alpha}_F)B_B$

- In total, 3 unknowns (R_F, B_F, $\boldsymbol{\alpha}_F$), and 3 equations

- What's wrong with this solution?

Simple Constraint: Assume Foreground has no Green

- In this case, $G_F = 0$, so the equations change to:

$$R = (\boldsymbol{\alpha}_F R_F) + (1-\boldsymbol{\alpha}_F)R_B$$

$$G = G_B - \boldsymbol{\alpha}_F G_B$$

First, solve for $\boldsymbol{\alpha}_F$

$$B = (\boldsymbol{\alpha}_F B_F) + (1-\boldsymbol{\alpha}_F)B_B$$

- In total, 3 unknowns (R_F, B_F, $\boldsymbol{\alpha}_F$), and 3 equations
- What's wrong with this solution?

Simple Constraint: Assume Foreground has no Green

- In this case, $G_F = 0$, so the equations change to:

  $R = (\boldsymbol{\alpha}_F R_F) + (1-\boldsymbol{\alpha}_F)R_B$

  $G = G_B - \boldsymbol{\alpha}_F G_B$ ⟵ First, solve for $\boldsymbol{\alpha}_F$

  $B = (\boldsymbol{\alpha}_F B_F) + (1-\boldsymbol{\alpha}_F)B_B$

- In total, 3 unknowns (R_F, B_F, $\boldsymbol{\alpha}_F$), and 3 equations

- What's wrong with this solution?

  - Excludes all greys except black, and a huge collection of other colors because white contains green.

Lec08 Required Reading

The Art of Deep Compositing

By Mike Seymour
February 27, 2014

At this year's SciTech Oscars there were three separate awards given for the invention, development and implementation of deep compositing.

> *"It is definitely a great honour and the Academy along with the reviewers did an amazing and thorough job," commented Colin Doncaster of Peregrine Labs, one of those honored at this SciTech awards. "The recognition is a huge validation for everyone involved, from the initial use cases to ODZ and EXR 2.0, it's great to see everyone working towards a standard to embrace acceptance and innovation. This very well could have been another case of a studio patenting technology and we'd all be without it. It was great to see Peter's (Hillman) and Weta's involvement in this be acknowledged by the Academy, and of course the initial concepts from Pixar. Jon (Wadelton) and others at The Foundry also had the insight to work with the various parties to make it available to a wider audience."*

Since our first coverage of this new and robust form of compositing in 2010, (click here for our fxguidetv ep where we originally explained Deep Comp) – fxguide has been following the development and now wide scale adoption of deep comp. Today it is a permanent feature at high-end effects companies such as Weta

Related Content

articles & podcasts

OpenEXR 2.0 goes Deep

fxguide at the SciTechs

NUKE 8 is coming: here's what you need to know

Hiero 1.5, Hieroplayer and Nuke 7 from The Foundry

fxpodcast #281: Milking the best for Doctor Who

Foundry At Siggraph with V-Ray + Speed tests

Foundry releases NUKE 6.3

Deep image compositing in a modern visual effects pipeline

Berlin, March 18th 2013

Bachelorthesis in the course audiovisual media
Stuttgart Media University

by
Patrick Heinen