# Algorithm to blend gradient filled corners in image

Asked 16 years, 3 months ago Modified 15 years, 6 months ago

Viewed 2k times

5

I need to put an alpha blended gradient border around an image. My problem is in blending the corners so they are smooth where the horizontal and vertical gradients meet. I believe there is a standard algorithm that solves this problem. I think I even encountered it in school many years ago. But I have been unsuccessful in finding any reference to one in several web searches.

(I have implemented a radial fill pattern in the corner, but the transition is still not smooth enough.)

**My questions**:

1. If there is a standard algorithm for this problem, what is the name of it, and even better, how is it implemented?
2. Forgoing any standard algorithm, what's the best way to determine the desired pixel value to produce a smooth gradient in the corners? (Make a smooth transition from the vertical gradient to the horizontal gradient.)

**EDIT**: So imagine I have an image I will insert on top of a larger image. The larger image is solid black and the smaller image is solid white. Before I insert it, I want to blend the smaller image into the larger one by setting the alpha value on the smaller image to create a transparent "border" around it so it "fades" into the larger image. Done correctly, I should have a smooth gradient from black to white, and I do everywhere except the corners and the inside edge.

At the edge of the gradient border near the center of the image, the value would be 255 (not transparent). As the border approaches the outside edge, the alpha value approaches 0. In the corners of the image where the vert & horiz borders meet, you end up with what amounts to a diagonal line. I want to eliminate that line and have a smooth transition.

What I need is an algorithm that determines the alpha value (0 - 255) for each pixel that overlaps in the corner of an image as the horizontal and vertical edges meet.

Share

Improve this question

Follow

edited Jun 2, 2009 at 16:21

asked Sep 20, 2008 at 0:46

TMarshall

825 ● 6 ● 15

## 2 Answers

Sorted by: Highest score (default)

2

Presumably you're multiplying the two gradients where they overlap, right?

Dunno about a standard algorithm. But if you use a signoid shaped gradient instead of a linear one, that should eliminate the visible edge where the two overlap.

A simple sigmoid function is smoothstep(t) = t*t*(3 - 2*t) where 0 <= t <= 1

Share Improve this answer

Follow

answered Sep 20, 2008 at 1:15



Mike F

This looks promising. I'm not sure I want to mess with XNA (This is a C# application and I think XNA is the best (only?) way to get smoothstep) but I'm working up a test now with it and will decide if I want to use it or roll my own based on the results. – TMarshall Sep 20, 2008 at 4:35

Smoothstep is just the (very simple) function I described; it's been around for yonks and is not exclusive to XNA. – Mike F

Sep 20, 2008 at 10:12

I just did an implementation - while it makes the gradient edges nicer, the corners have the same issue. The real problem is the optical illusion created by the gradients. There's no 'error' for either linear or smoothstep, but you still 'see' a line unless you block the reset of the gradient from your vision. – Lilith River Sep 7, 2012 at 19:03

0

If you don't need it to be resizable, then you can just use a simple alpha map.

However, I once used a simple Gaussian fade, with the mean at the location where I wanted it to be the last fully-opaque pixels to be. If that makes sense.

Share Improve this answer

Follow

answered Sep 20, 2008 at 0:57

TraumaPony

**10.8k** ● 13 ● 57 ● 75