

Re: using a pixel shader for undistorting an image?

Source: <http://sci.tech-archive.net/Archive/sci.image.processing/2007-04/msg00083.html>

- *From:* "pixel.to.life" <pixel.to.life@xxxxxxxx>
 - *Date:* 12 Apr 2007 23:25:55 -0700
-

Giff,

Sounds like you already got the difficult part out of the way: calibration. How did you get the mappings between distorted and corrected image-pixels? This is what you plan to use as a lookup table on the GPU right?

I wonder if just resampling a 640x480 image interactively when the pixel transforms are given is too computationally intensive to move to GPU. If it is a just-for-fun project, cool. But GPUs would be best at handling large streams of data and apply the same operation on to a bunch all at once. In this case, the only such operation I see is to resample based on a lookup table. Sounds easy enough. I have come across using lookup tables for interactive manipulation of optical properties of primitives for volume rendering, and this one sounds along the lines. Good luck!

P.

On Apr 12, 3:19 am, "Giff" <Giffn...@xxxxxxxx> wrote:

On 12 Apr, 11:46, Harris <xgeorg...@xxxxxxxxxxxxxxxx> wrote:

The problem you are describing is generally known as "camera calibration"

Not exactly, maybe I was not clear enough. I already did the calibration, now I need to use the results of it to undistort the images coming from the camera, I know how to do this in C++, what I want to do is to move this computation onto the GPU, using a pixel shader.

I think I have a clue on how to do this: in the initialization process

Re: using a pixel shader for undistorting an image?

I will create a texture containing, for each pixel, the destination of that pixel in the undistorted image (or probably just the offsets in the x and y directions). I will then pass this texture to the pixel shader and use it to sample the right pixel.

Thanks for your reply anyway.