# Sightations ← A Computer Vision Blog

- Home
- About
- Contact
- Code
- Archive

# Calibrated Cameras and gluPerspective

June 18, 2013

After posting my last article relating glFrustum to the intrinsic camera matrix, I receieved some emails asking how the (now deprecated) gluPerspective function relates to the intrinsic matrix. We can show a similar result with `gluPerspective` as we did with `glFrustum`, namely that it is the product of a `glOrtho` matrix and a (modified) intrinsic camera matrix, but in this case the intrinsic matrix has different constraints. I'll be re-using notation and concepts from the previous article, so if you aren't familiar with them, I recommend reading it first.

## Decomposing gluPerspective

The matrix generated by `gluPerspective` is

$$\begin{pmatrix} \frac{f}{\text{aspect}} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & C' & D' \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

where

$$\begin{aligned} f &= \cot(fovy/2) \\ C' &= -\frac{far + near}{far - near} \\ D' &= -\frac{2\ far\ near}{far - near} \end{aligned}$$

Like with `glFrustum`, `gluPerspective` permits no axis skew, but it also restricts the viewing volume to be centered around the camera's principal (viewing) axis. This means that the principal point offsets $x_0$ and $y_0$ must be zero, *and* the matrix generated by `glOrtho` must be centered, i.e. `bottom = -top` and `left = -right`. The *Persp* matrix corresponding to the intrinsic matrix is:

$$Persp = \begin{pmatrix} \alpha & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & A & B \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

where

$$A = near + far$$
$$B = near * far$$

and the $NDC$ matrix is

$$NDC = \begin{pmatrix} \frac{2}{right-left} & 0 & 0 & t_x \\ 0 & \frac{2}{top-bottom} & 0 & t_y \\ 0 & 0 & -\frac{2}{far-near} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \frac{2}{width} & 0 & 0 & 0 \\ 0 & \frac{2}{height} & 0 & 0 \\ 0 & 0 & -\frac{2}{far-near} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$t_x = -\frac{right + left}{right - left}$$
$$t_y = -\frac{top + bottom}{top - bottom}$$
$$t_z = -\frac{far + near}{far - near}$$

It is easy to show that the product $(NDC \times Persp)$ is equivalent to the matrix generated by `gluPerspective(fovy, aspect, near, far)` with

$$\text{fovy} = 2\text{arctan}\left(\frac{\text{height}}{2\beta}\right)$$
$$\text{aspect} = \frac{\beta}{\alpha}\frac{\text{width}}{\text{height}}.$$

## glFrustum vs. gluPerpsective

In my experience, the zero-skew assumption is usually reasonable, so `glFrustum` can provide a decent approximation to the full intrinsic matrix. However there is quite often a non-negligible principal point offset (~ 2% of the image size), even in high-quality cameras. For this reason, `gluPerspective` might be a good choice for quick-and-dirty demos, but for the most accurate simulation, you should use the full camera matrix like I described previously.

*Posted by Kyle Simek*
Dissecting the Camera Matrix, Part 3: The Intrinsic Matrix → ← Calibrated Cameras in OpenGL without glFrustum

0 points Tweet

0 Comments Sightations 1 Login

♡ Recommend Tweet f Share Sort by Best

Start the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Be the first to comment.

ALSO ON SIGHTATIONS

**Q & A: Recovering pose of a calibrated camera - Algebraic vs. Geometric method?**

1 comment • 4 years ago

Avatar dissertation writing service — Learning this 3D points projections is ideal in math subject. It gives the easy way to determine the …

**Compiling ELSD (Ellipse and Line Segment Detector) on OS X**

4 comments • 5 years ago

Avatar philip andrew — Do you know anything better than elsd now?

**Dissecting the Camera Matrix, Part 2: The Extrinsic Matrix**

16 comments • 6 years ago

Avatar Rajesh — Hi, I am looking for a method to compute homography matrix between images(left & right) of stereo camera using …

**Calibrated Cameras in OpenGL without glFrustum**

21 comments • 6 years ago

Avatar ksimek — Thanks for the clarifying photos. Sorry, I misunderstood the scenario -- I thought your calibration image was larger …

✉ Subscribe Ⓓ Add Disqus to your siteAdd DisqusAdd 🔒 Disqus' Privacy PolicyPrivacy PolicyPrivacy

*Content by Kyle Simek. Original design by Mark Reid*
*(Some rights reserved) Powered by Jekyll*