NAME

pmdblur - generate views for combined camera motion and depth blurring

SYNOPSIS

pmdblur speed aperture nsamp v0file v1file

DESCRIPTION

Pmdblur takes two viewfiles and generates *nsamp* views starting from *v0file* and moving towards *v1file,* simulating an aperture of diameter *aperture* in world coordinate units. When rendered and averaged together, these views will result in a picture with motion and depth-of-field blur due to a camera changing from v0 to v1 in a relative time unit of 1, whose shutter is open starting at v0 for *speed* of these time units. Either *pinterp(1)* or *rpict(1)* may be called to do the actual work. (The given *v0file* must also be passed on the command line to the chosen renderer, since *pmdblur* provides supplemental view specifications only.)

For *pinterp,* feed the output of *pmdblur* to the standard input of *pinterp* and apply the *−B* option to blur views together. In most cases, two pictures with z-buffers at v0 and v1 will get a satisfactory result, though the perfectionist may wish to apply the *−ff* option together with the *−fr* option of *pinterp*.

To use *pmdblur* with *rpict,* apply the *−S* option to indicate a rendering sequence, and set the *−o* option with a formatted file name to save multiple output pictures. When all the renderings are finished, combine them with the *pcomb(1)* program, using appropriate scalefactors to achieve an average. Note that using *rpict* is MUCH more expensive than using *pinterp,* and it is only recommended if the scene and application absolutely demand it (e.g. there is prominent refraction that must be modeled accurately).

For both *pinterp* and *rpict,* the computation time will be proportional to the number of views from *pmdblur.* We have found a *nsamp* setting somewhere between 7 and 15 to be adequate for most images. Relatively larger values are appropriate for faster camera motion.

The *−pm* and/or *−pd* options of *rpict* may be used instead or in combination to blur animated frames, with the added advantage of blurring reflections and refractions according to their proper motion. However, this option will result in more noise and expense than using *pmdblur* with *pinterp* as a post-process. If both blurring methods are used, a smaller value should be given to the *rpict −pm* option equal to the shutter speed divided by the number of samples, and the *−pd* option equal to the aperture divided by the number of samples. This will be just enough to blur the boundaries of the ghosts which may appear using *pmdblur* with a small number of time samples.

To simulate a particular camera's aperture, divide the focal length of the lens by the f-number, then convert to the corresponding world coordinate units. For example, if you wish to simulate a 50mm lens at f/2.0 in a scene modeled in meters, then you divide 50mm by 2.0 to get 25mm, which corresponds to an effective aperture of 0.025 meters.

EXAMPLES

To use *pinterp* to simulate motion blur between two frames of a walk-through animation, where the camera shutter is open for 1/4 of the interframe distance with an aperture of 0.1 world units:

```
pmdblur .25 .1 8 fr1023.hdr fr1024.hdr | pinterp −B −vf fr1023.hdr −x 640 −y 480 fr1023.hdr fr1023.zbf fr1024.hdr fr1024.zbf > fr1023b.hdr
```

AUTHOR

Greg Ward

SEE ALSO

pcomb(1), pdfblur(1), pinterp(1), pmblur(1), pmblur2(1), rcalc(1), rpict(1), vwright(1)