Subject: Re: arbitrary rotation of 3-d arrays
Posted by morisset on Sat, 12 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Foster wrote:

> The previous posts discussing the use of T3D were all assuming that
> once the transformation of coordinates was performed, one would have
> to then use interpolation to actually transform the data. At least in
> my post I simply forgot to mention this important step. (oops!)

And perhaps the original question was 'How do interpolate in a 3D cube?' !!

> My main concern with your method is that it is using 2D techniques
> to perform a 3D task, and I believe will invariably suffer from poorer
> performance. Both methods will require large amounts of memory, but
> in your method you make entire copies of the original data-set, and
> three times! Often the data for such an operation will be huge, as
> in the case of medical images. The performance penalties this copying
> will incur, as well as the use of for loops to process each set of
> 2D images through the data, will make this method much slower than
> the use of T3D and interpolation.

Once more, the point is the interpolation, not to get the coordinate matrix. I'm not sure that a 3D interpolation will be faster than N 2D interpolation. Since the poly_2d function used in the rsi ROT function is not available, it's not possible to 'have a look and generalize'!

> I would also argue that it would be less accurate, since you are
> performing interpolation three times in succession basically, once
> within each 2D plane, whereas the T3D method will transform the
> coordinates mathematically and then interpolate the original data
> once from those computed coordinates. Quite a different animal.

I agree with you and at the time I was needing this turn_3d, I tried to make the 3D interpolation after doing the t3d transformation. As you see, I didn't succeed (well, I didn't tried a lot of time, 'cause my datas are 'just' 100^3)! And the use of 2D slide by slide was better, 'cause I finally just had to make one rotation ;-)

Anyway, the question remains: where is a 3D interpolation function???

Best regards.
Chris.