Subject: Re: Finding all angles within a range of directions; an algorithm question
Posted by Mark Zimmerman on Thu, 11 Apr 2002 21:29:17 GMT
View Forum Message <> Reply to Message

In article <fd12a3f3.0204111301.41ce7b31@posting.google.com>,
tbowers0@yahoo.com wrote:
> Say I have a 3D array of observation data of 'Brightness' over the
> hemisphere of theta,phi angles (theta=0 is straight up, phi=0 is
> North, and 3rd dim is timesteps).
>
> data = findgen(3,4,10) ;simple example with 3 thata angles, 4 phi
> angles, 10 timesteps
> theta = [0,30,60] ;a *very* simple example
> phi = [0,90,180,270]
> timestep = indgen(10)
>
> If I have a flat plate facing an arbitrary angle, how do I find all
> brightnesses that fall on its surface. In other words, all
> brightness[*,*,*] that come from angles within +- 90 degree hemisphere
> of plate's surface (of course, all angles are really in radians, but
> listed here as degrees for clarity). The only solution I come up with
> requires:
>
> 1) reforming the brightness data to a list of
> theta,phi,timestep,brightness quadruplets (now a 2D array
> [4,n_thetas*n_phis*n_timesteps]; about 7 or 8 lines of code
> reform()ing and transpose()ing). Doing this cause I'll use where() in
> a moment
>
> 2) convert the theta,phi polar coordinates to x,y,z cartesian
> coordinates by:
> brightnessAnglesCartesian = sin(brightness[0,*]) *
> cos(brightness[1,*]), sin(brightness[0,*]) * sin(brightness[1,*]),
> cos(brightness[0,*])]
>
> 3) convert the plate's theta,phi polar coordinate facing direction
> (its 'normal') to x,y,z cartesian coordinate by same formula to create
> plateAngleCartesian, a 3-element vector
>
> 4) compute all angles psi that plate normal (plateAngleCartesian)
> makes with all brightness angles (brightnessAnglesCartesian) by:
> psi = acos(plateAngleCartesian # brightnessAnglesCartesian) ;acos(dot
> product of 3x1 plate angle vector and 3xN brightness angle vectors)
>
> 5) indices = where(psi le !pi/2.0)
>
> 6) Now I can work with these angles: brightness[3,indices]) = 0.0
> ;brightnesses in 4th column

>
> This seems awfully circuitous. I'm using an interactive interface to
> rotate the plate with the mouse so calculation speed is critical and I
> think my method is way too slow (and not very elegant either). Could
> anyone please advise on a better way to do this? One thing I've
> learned is that when dealing with angles and rotations, there are
> usually very quick and clever alternatives than my usual brutish
> approach.
>
> Many thanks in advance

One simplification: in (4), skip taking the acos of the dot products.
All of the angles you want will have positive dot products; just throw
out the negative ones.

-- Mark
