Subject: Re: Azimuth and Offset XYZ position correction
Posted by James Kuyper on Thu, 23 Mar 2006 23:42:46 GMT
View Forum Message <> Reply to Message

Dave wrote:
> Hi All,
>
> Quick question for anyone out there. I am looking to write a script and
> I'm wondering if anyone out there has done it already. I have a known
> precise position, say X and Y (UTM coordinate) in meters. From that
> precise position I am using a laser range finder to calculate a
> distance measurement from my original XY location. Along with this
> distance measure I obtain the exact azimuth.
>
> What I would like to do is calculate the precise XY location at the end
> of the laser measurement.

You don't mention the elevation angle of the laser finder. Without that information, the best we can do is assume that it's 0. I would imagine that the distances you measure are much smaller than the radius of the earth, which would allow the use of certain approximations, but I'm going to use a method that will give accurate results for any distance measured along a great circle on the surface of the earth all the way up to half the circumference of the Earth. First, set up the utm map projection:

```
utm = MAP_PROJ_INIT('UTM', CENTER_LONGITUDE=clon, $
   CENTER_LATITUDE=clat, ZONE=z);

;If you don't know what the zone is for your UTM projection, you
probably aren't using
;one, and you can drop that argument. Next, convert your known position
to longitude
;and latitude:

lonlat = MAP_PROJ_INVERSE(known, MAP_STRUCTURE=utm);

;Next, set up an azimuthal equidistant projection centered at your
known position:

azeq = MAP_PROJ_INIT('Azimuthal Equidistant',
CENTER_LONGITUDE=lonlat[0], $
   CENTER_LATITUDE=lonlat[1]);

;The next step depends upon how you define azimuth - there is a
standard convention; ;unfortunately, there are many different mutually
incompatible conventions. 0 degrees
;can represent either due North, or due East. A positive azimuth can
```

represent either a
;clockwise or a counterclockwise rotation, as seen from above. I will
assume that due
;North is 0 degrees, and that due East is +90 degrees. Then calculate
appropriate
;offsets:

dx = distance*sin(azimuth*!DTOR)
dy = distance*cos(azimuth*!DTOR)

;Now convert those offsets to a latitude and a longitude.

newlonlat = MAP_PROJ_INVERSE(dx, dy, MAP_STRUCTURE=azeq);

;And finally, back to UTM XY coordinates:

newxy = MAP_PROJ_FORWARD(newlonlat, MAP_STRUCTURE=utm)

There are other ways to do this, and some are more efficient, but this
one hides all of the spherical trig inside the map projection code,
allowing you to concentrate on other issues.

There should be a standard function to do this; it's sort-of the
inverse of map_2points(). However, I couldn't find one using the
built-in help.
