Subject: Re: [Q]: How to calculate distance from GPS measurements
Posted by Liam Gumley on Fri, 22 Mar 1996 08:00:00 GMT
View Forum Message <> Reply to Message

Vince Scullin wrote:

>
> I am working on a project where I will be receiving measurements from the Global
> Positioning System, presumably latitude and longitude measurements, and I will
> need to calculate the distances between the measurement points. The measurements
> will all be taken over a region of only a few miles so I guess I could assume
> the earth is flat over this region and just calculate the straight line
> distance. But I was wondering if anyone could help me with a more mathematically
> rigorous method for calculating distance from pairs of latitude/longitude
> measurements?

I converted this from FORTRAN to IDL. It was sent to me by someone on the net whose name I don't recall. It has some sort of reference you could try and look up. Note that I did not try to optimize the loop.

```
PRO eqldaz, eth, eph, th, phi, n, xdeg, dis, az
;+
; Purpose:
; EQLDAZ calculates the distance (degrees and km) and azimuth
; between a position in decimal degrees (ETH latitude, EPH longitude),
; and an array of N decimal lat (TH) and long (PHI) positions.
; All input positions are in geographic coordinates which are
; converted to equidistant latitude coordinates for all internal
; calculations using the method of Brown (1984).
; The first-order great-ellipse correction is applied (equation (36)).
; Author: DRHO (using the results of Brown (1984) GJRAS, 445.)
; input:
; double     eth        initial latitude (degrees, N+)
; double     eph        initial longitude (degrees, E+)
; double     th(n)      array of latitudes (degrees, N+)
; double     phi(n)     array of longitudes (degrees, E+)
; integer    n          number of points array
; output:
; double     xdeg       distance between points (degrees)
; double     dist       distance between points (kilometers)
; double     az         azimuth from initial point (degrees)
;-

xdeg = dblarr(n)
dis = dblarr(n)
az = dblarr(n)

RADCO = 1.745329251994329D-02
DEGCO = 57.29577951308232D0
```

```
FLATTN = 1.D0/298.257D0
ONFLAT = 1.D0 - FLATTN
ELLIP0 = 1.001119D0
ELLIP1 = 0.001687D0
DEGKM = 111.19504D0
PIO2 = RADCO*90.D0

; Calculate equidistant latitude factor

FLTFAC = ONFLAT*SQRT(ONFLAT)
THK = ETH*RADCO
PHK = EPH*RADCO

; Convert source geographic latitude to equidistant latitude

THG = THK
IF(ABS(ETH) NE 90.D0) THEN THG = ATAN(FLTFAC*TAN(THK))

; Convert to colatitude

THG = PIO2 - THG

; Calculate spherical trig quantities

D = SIN(PHK)
E = COS(PHK)
F = SIN(THG)
A = E*F
B = D*F
C = COS(THG)
G = C*E
H = C*D
E = -E
F = -F

; Calculate distance and azimuth from each position to source

FOR I = 0, N-1 DO BEGIN

THC = TH(I)*RADCO
PHC = PHI(I)*RADCO
THG = THC
IF(ABS(TH(I)) NE 90.D0) THEN THG = ATAN(FLTFAC*TAN(THC))
THG = PIO2 - THG
D1 = SIN(PHC)
E1 = COS(PHC)
F1 = SIN(THG)
A1 = F1*E1
```

```
B1 = D1*F1
C1 = COS(THG)

; Calculate distance in radians (Bullen p 155 (4))

XDEG(I) = ACOS(A*A1+B*B1+C*C1)
AD = A1 - D
BE = B1 - E
AG = A1 - G
BH = B1 - H
CK = C1 - F

; Calculate azimuth in radians (Bullen p 155 (7)&(8))

AZ(I) = ATAN(AD*AD+BE*BE+C1*C1-2.D0,AG*AG+BH*BH+CK*CK-2.D0)

; Calculate quantities for great-ellipse correction (Brown (36))

BH = SIN(ACOS(-E*SIN(AZ(I))))

; Make great-ellipse correction and convert to degrees

XDEG(I) = XDEG(I) * DEGCO * (ELLIP0 - ELLIP1*BH*BH)
AZ(I) = DEGCO * AZ(I)

; Calculate arc distance in km (Brown (33) & (36))

DIS(I) = DEGKM * XDEG(I)

ENDFOR

END
```