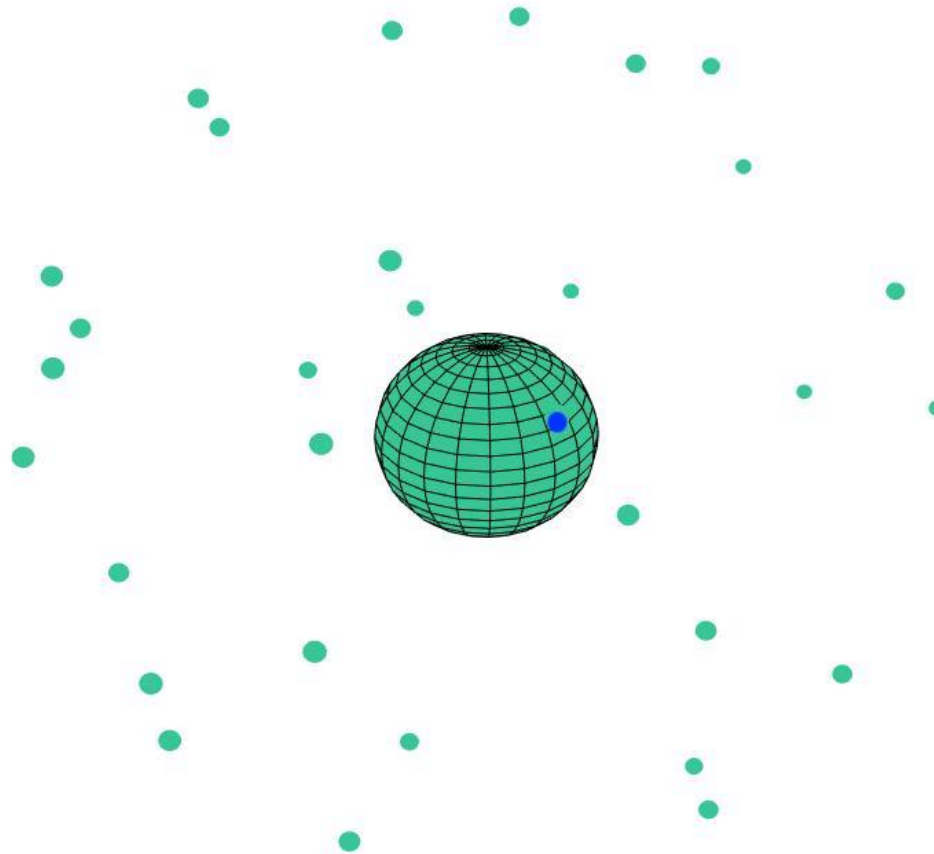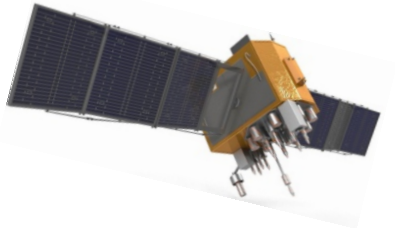# Simplified GPS
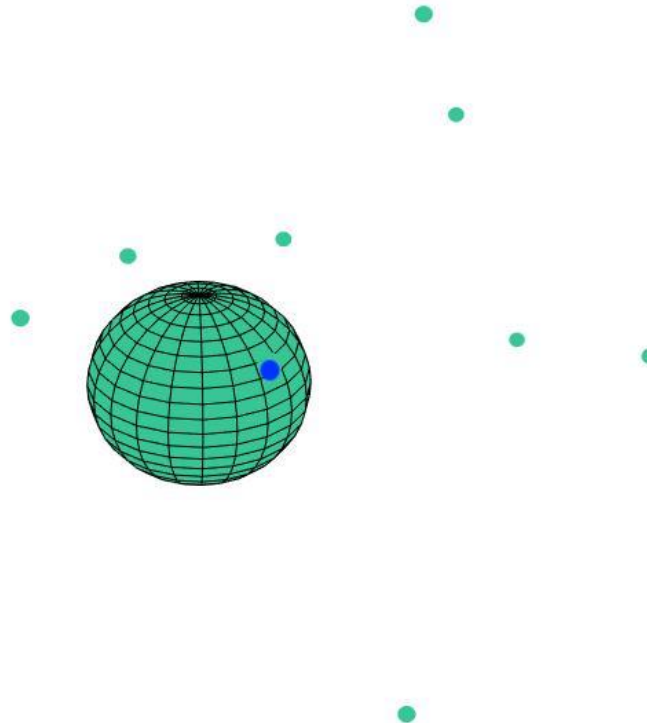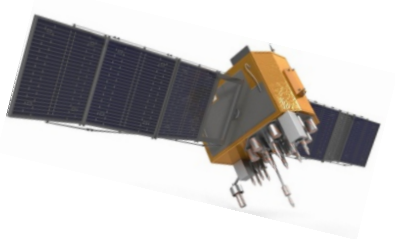## spherical Earth, receiver sync'd with satellite clocks

31 U.S. GPS satellites active at 1:30 pm, June 12, 2019

https://in-the-sky.org/satmap_worldmap.php

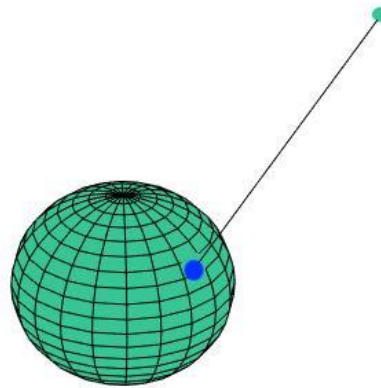**Russia, China and the EU also have GPS systems**

satellite sizes are exaggerated

Simplified GPS

**8 satellites are 10° or more above horizon at San Diego, CA
32.7° latitude, -117° longitude
GPS receiver at blue dot does NOT know it is there yet,
only that it's somewhere on Earth**

Simplified GPS
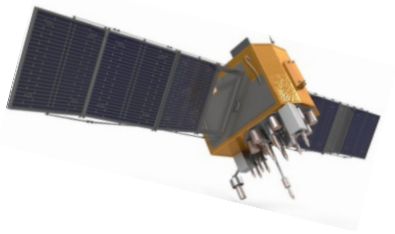
**Each satellite transmits signals with time sent and satellite location,
the time-sync'd GPS receiver computes distance to satellite
from the time difference between broadcast and reception,
using the speed of light**

Simplified GPS

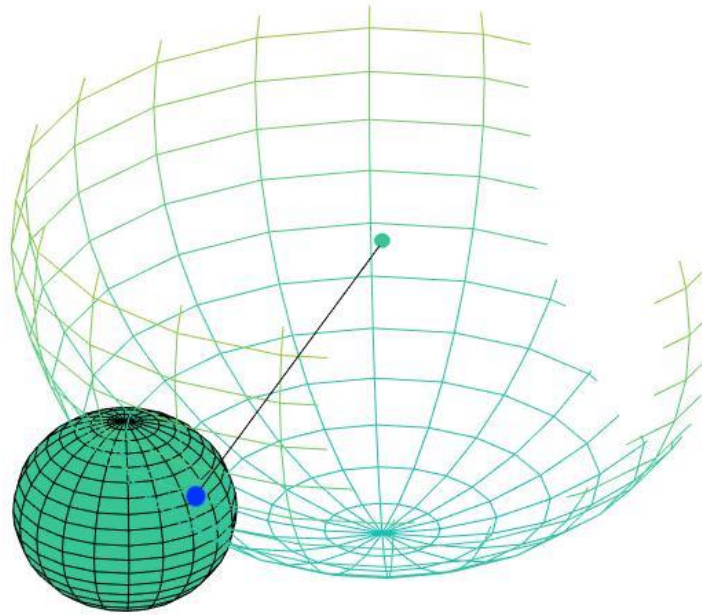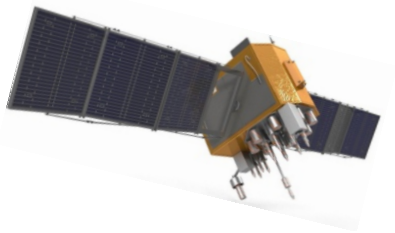**Distance to satellite gives equation for sphere of that
radius around known satellite location - receiver only knows
it is located somewhere on that sphere and on Earth's surface**

Simplified GPS
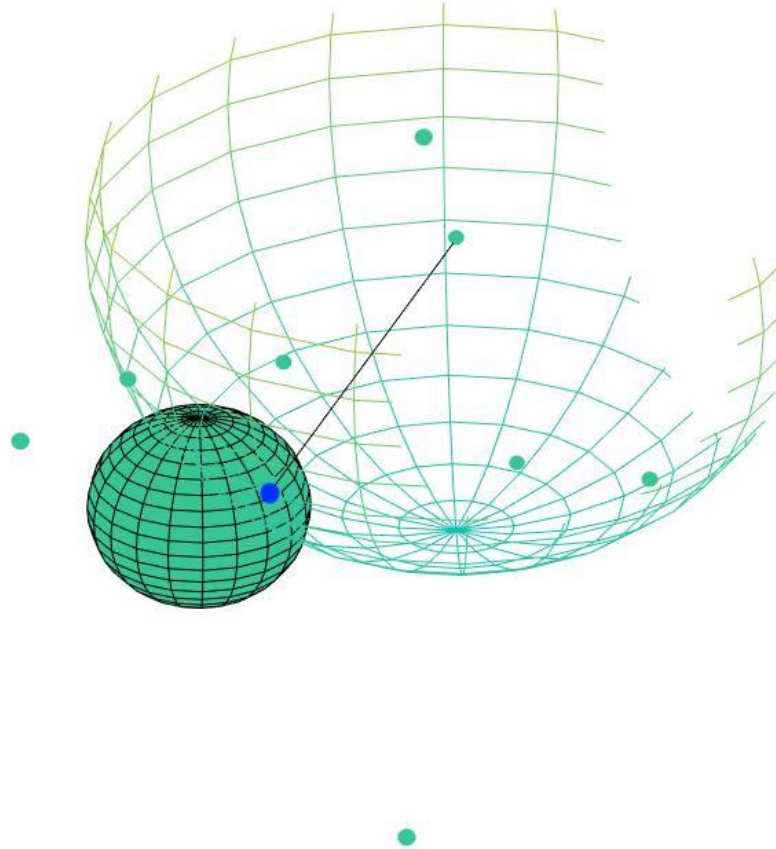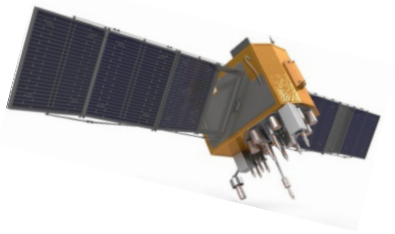
**The GPS receiver then computes its location from the intersection of
3 or more satellite spheres with Earth's surface, where
intersection of 2 spheres is a circle, 3 is 2 points, 4 is one point,
a problem in Linear Algebra, solving multiple, coupled algebraic equations**

Improved GPS
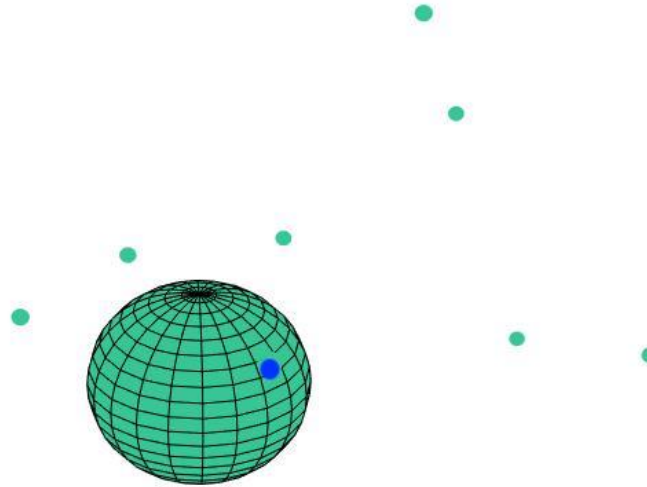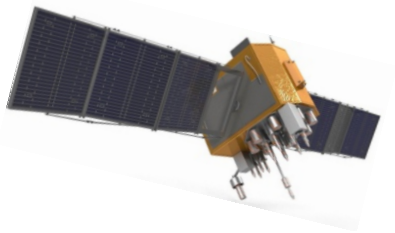
**GPS receivers are not usually sync'd with the satellites, so
receivers can only determine the time and distance difference between
satellite pairs, which defines 2-sheet hyperboloids of revolution,
then solve for intersection of hyperboloids & oblate spheroid Earth.**

B. Fang, "Simple Solutions for Hyperbolic and Related Position Fixes," IEEE Transactions on Aerospace and Electronic Systems, vol. 26, no. 5, pp. 748–753, 1990
https://ieeexplore.ieee.org/abstract/document/102710

Cell phones use additional info: last know location, cell & wifi tower locations…

Simplified GPS in MATLAB

```matlab
% simplified GPS in MATLAB - receiver clock sync'd with satellites
re = 6370; % (km), spherical earth radius

% specify GPS receiver latitude, longitude and altitude (altitude == 0)
rec = [32.7,-117,0]; % San Diego, CA, USA is [32.7,-117,0]
[x,y,z] = fLatLongToXYZ(rec, re);
xyzRec = [x,y,z]; % xyz coordinates of receiver, earth center is origin

% specify >= 3 satellite latitude (deg), longitude (deg), altitude (km)
% 31 listed in file sat.txt taken 1:30 pm, June 12, 2019 from data at
% https://in-the-sky.org/satmap_worldmap.php
load sat.txt

% get xyz coordinates of satellites
[x, y, z] = fLatLongToXYZ(sat,re);
xyz = [x, y, z];

% get satellites above horizon and in view of receiver
degdel = 10; % min degree above horizon for sat in view
rView = fReturnSatViewRows(sat,xyz,xyzRec,re,degdel);
xyz = xyz(rView,:);
r = fDistance(xyz,xyzRec); % sats to receiver

% END SETUP

% GIVEN:
% radius of spherical earth, re
% lat, long and altitude of >= 3 satellites
% distance of each satellite from receiver

% FIND:
% lat and long of receiver on earth's surface

% matrix eqn for sphere intersections is A * xyz = c
A = xyz;
c = fCcoef(xyz,r,re);

% xyzCalc = inv(A) * c; % OK only for A and c rows == 3
xyzCalc = A \ c; % OK for A and c rows >= 3

% compute receiver lat and long
[latCalc, longCalc, altCalc] = fXYZtoLatLong(xyzCalc', re);

fprintf('rec loc: lat, long, alt, %6.3f, %6.3f, %4.3e \n', rec)
fprintf('rec cal: lat, long, alt, %6.3f, %6.3f, %4.3e \n', ...
    latCalc, longCalc, altCalc)
```

```matlab
function rowView = fReturnSatViewRows(sat,xyz,xyzRec,re,degdel)
    % returns row numbers of satellites > degdel above horizon

    dRec = fDistance(xyz,xyzRec); % distances from sats to receiver
    dOrig = re + sat(:,3); % distances from sats to earth center

    % we know 3 sides of triangle between sat, rec, earth center
    % use law of cosines
    num = re^2 + dRec.^2 - dOrig.^2;
    denom = 2 * re * dRec;
    gamma = -90 + acosd(num ./ denom);

    % find and return satellite row numbers where gamma >= degdel
    rowView = find(gamma >= degdel);
```

```matlab
function c = fCcoef(xyz,r,re)
    % input xyz are locations of satellites (each row is satellite)
    % input r are distances from satellites to receiver
    % input re is radius of spherical earth
    % returns vector of coefficients for matrix solution
    % option 2 for sum( ,2) sums each row

    c = ( (re^2 + sum(xyz.^2, 2) - r.^2) / 2 );
```

```
>> gps3_
rec loc: lat, long, alt, 32.700, -117.000, 0.000e+00
rec cal: lat, long, alt, 32.700, -117.000, -9.095e-13
```