GlitterNet

Computer Science Independent Study

Professor Robert Pless, Undergraduate Student Ray Su
WASHINGTON UNIVERSITY IN ST. LOUIS MISSOURI

Astrometry paper: https://arxiv.org/pdf/0910.2233v1.pdf
Database: http://www.astronexus.com/hyg
GitHub: https://github.com/suziray/LAB-astro-net

Part 1: Introduction

- how this database is organized
- what data is in it
- how to sort for very bright stars
- how to make a picture of the sky that we might a little bit recognize

The database is a subset of data in 3 major catalogs (Hipparcos Catalog, Yale Bright Star Catalog, and Gliese Catalog of Nearby Stars). These 3 catalogs have their distinct advantages:

- Hipparcos is the largest collection of high-accuracy stellar positional data, particularly parallaxes, which makes it useful as a starting point for stellar distance data
- Yale Bright Star Catalog contains basic data on essentially all naked-eye stars, including much information (such as the traditional Bayer Greek letters and Flamsteed numbers) missing from many other catalogs
- The Gliese catalog is the most comprehensive catalog of nearby stars (those within 75 light years of the Sun). It contains many fainter stars not found in Hipparcos

We used the 3rd version of this database, which contained 119,614 stars and could be downloaded from HYG 3.0 as a csv file. While there were 37 fields in this database, we used 22 of them that have no data missing:

- **id**: The database primary key
- **ra**, **dec**: The star's right ascension and declination, for epoch and equinox 2000.0
- **dist**: The star's distance in parsecs, the most common unit in astrometry. To convert parsecs to light years, multiply by 3.262. A value >= 10000000 indicates missing or dubious (e.g., negative) parallax data in Hipparcos
- **pmra**, **pmdec**: The star's proper motion in right ascension and declination, in milliarcseconds per year
- **rv**: The star's radial velocity in km/sec, where known
- **mag**: The star's apparent visual magnitude
- **absmag**: The star's absolute visual magnitude (its apparent magnitude from a distance of 10 parsecs)
- **x**,**y**,**z**: The Cartesian coordinates of the star, in a system based on the equatorial coordinates as seen from Earth. +X is in the direction of the vernal equinox (at epoch 2000), +Z towards the north celestial pole, and +Y in the direction of R.A. 6 hours, declination 0 degrees
- **vx**,**vy**,**vz**: The Cartesian velocity components of the star, in the same coordinate system described immediately above. They are determined from the proper motion and the radial velocity (when known). The velocity unit is parsecs per year; these are small

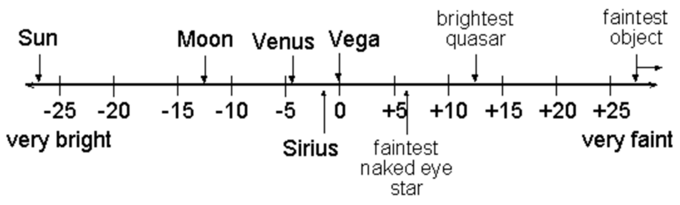
values (around 1 millionth of a parsec per year), but they enormously simplify calculations using parsecs as base units for celestial mapping

- **rarad**, **decrad**, **pmrarad**, **prdecrad**: The positions in radians, and proper motions in radians per year
- **comp**, **comp_primary**: Identifies a star in a multiple star system. comp = ID of companion star, comp_primary = ID of primary star for this component. Currently only used for Gliese stars
- **lum**: Star's luminosity as a multiple of Solar luminosity

Computing environment: we used **MATLAB_R2016a** to access and analyze this database.

Brightness of starts: in astronomy, there were several measurements for the brightness of a star. Our database has three: apparent magnitude, absolute magnitude, and luminosity. Absolute magnitude is the magnitude of a star from 10 parsecs, and luminosity is just a multiple of Solar luminosity. Therefore, we used apparent magnitude, which would give us the brightness of a star if we are observing from Earth. (http://www.astronomynotes.com/starprop/s4.htm)

The dimmest object visible with naked eye has apparent magnitude of 6.5 under normal condition. As shown below, the lower the magnitude, the brighter a star is:

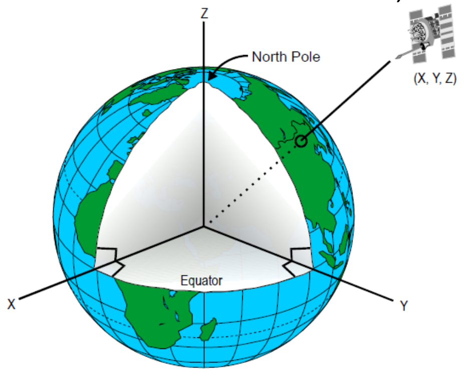


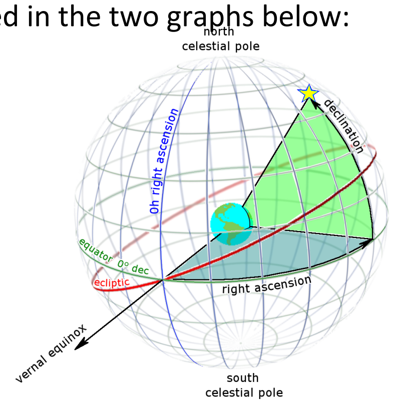
Apparent brightnesses of some objects in the magnitude system.

In the database, there are 8913 stars with magnitude 6.5 or lower. Let's draw them.

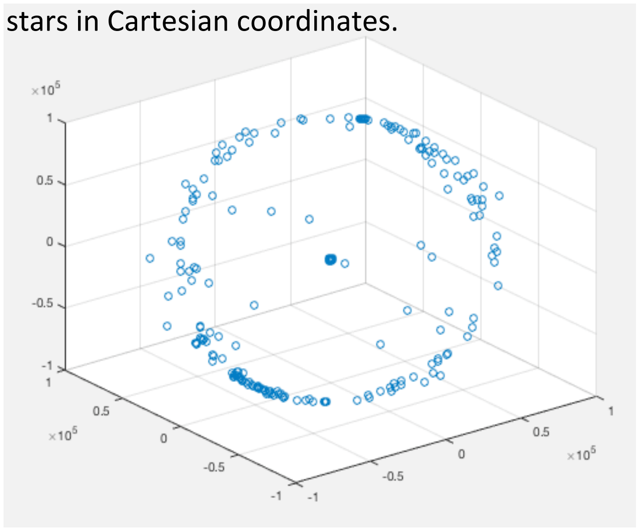
Part 2: Plotting the Universe

The (x, y, z) coordinates we used for stars were based on a system called Earth-centered inertial (ECI) coordinate frame, whose +X points towards the direction of vernal equinox (direction to Sun on March 21st), +Y towards Right Ascension 6 hours (90 degrees), declination 0 degrees, and +Z towards North Celestial Pole, as demonstrated in the two graphs below:

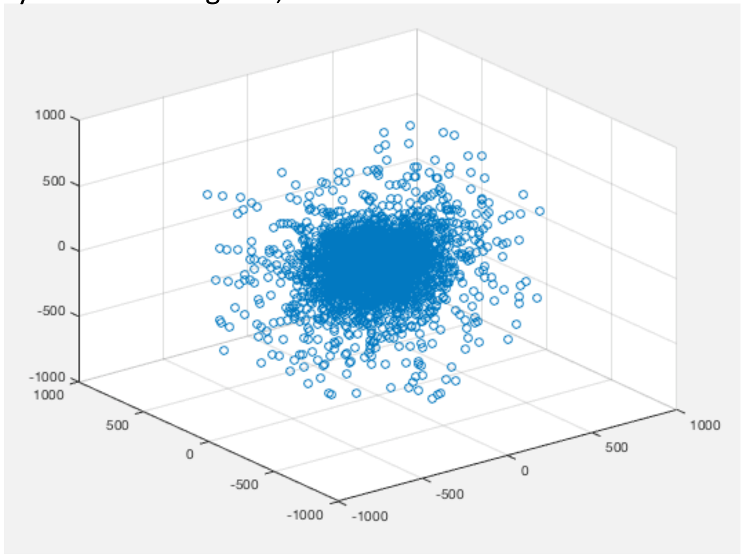




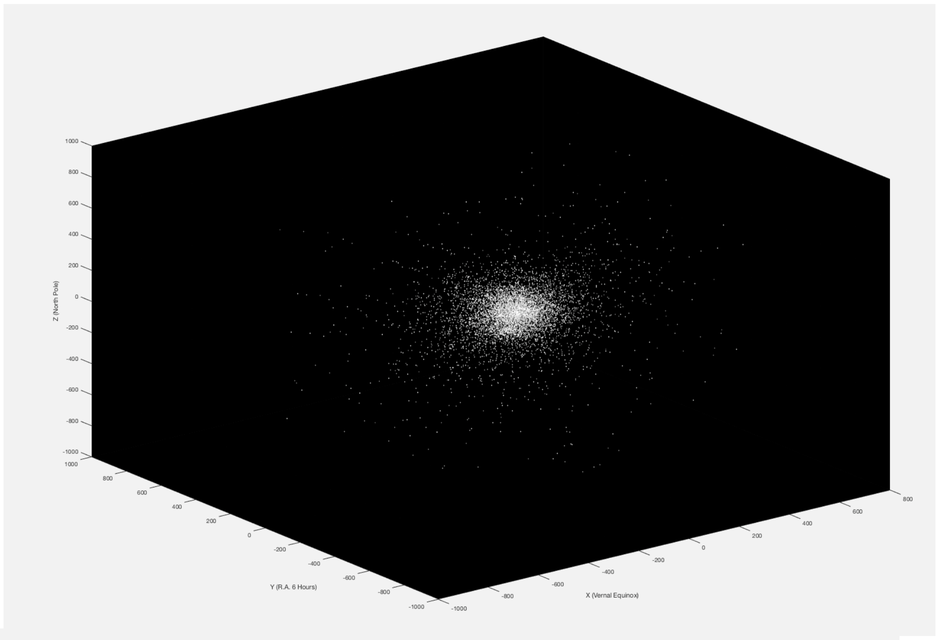
We chose this system because ECI does not rotate with Earth, and it is useful for specifying location of and direction towards celestial objects. For example, Sun's coordinates are (0.000005, 0, 0) in this system with unit length = 1 parsec. We used scatter3(x, y, z) function in MATLAB to draw the stars in Cartesian coordinates.

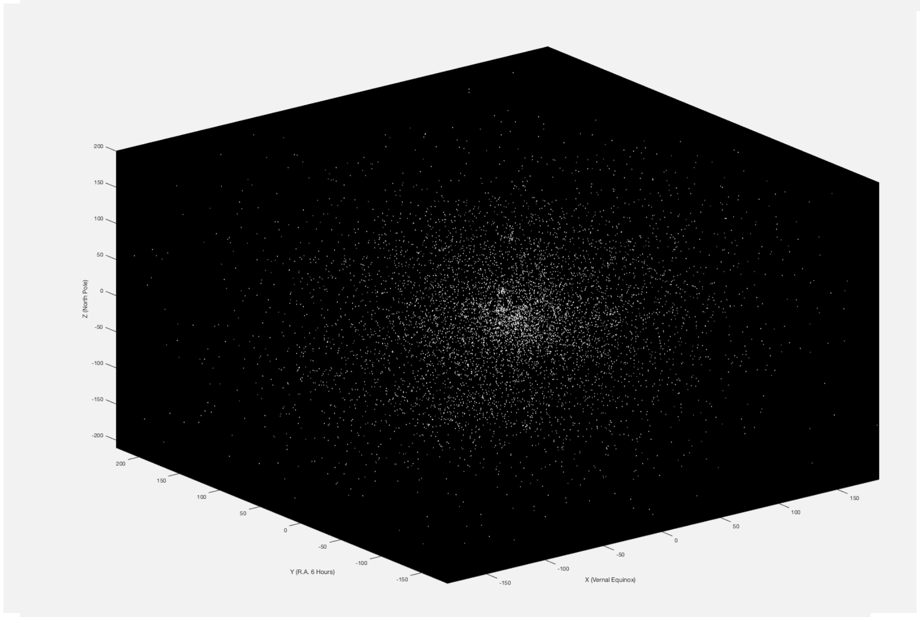


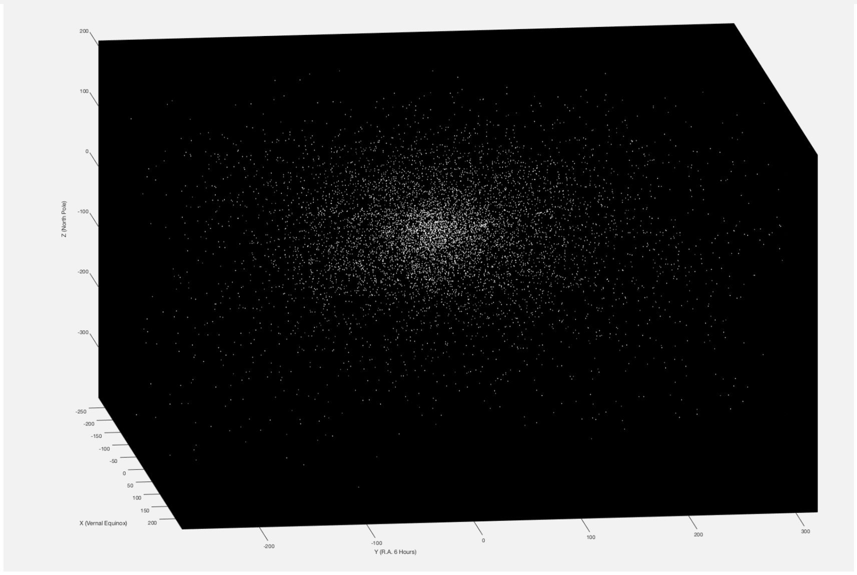
We noticed that there are 199 stars with distance of 100000 parsecs, which is the maximum distance value in the whole database and greatly distorted the plot. Although the website mentioned a value >= 10000000 indicates missing or dubious parallax data in Hipparcos, we believed that removing these 199 stars would be helpful in plotting. Thus, we obtained a data set of 8713 stars by also removing Sun, which are drawn below:



Now the graph makes much more sense, let's improve our plotting and include more information. We added axes labels, changed background color, and set each data point's color (brightness) using a linear distribution between RGB triplet [.2, .2, .2] and [1, 1, 1] based on the stars' apparent magnitude ranking (e.g. the brightest star, Sun, with the lowest magnitude value got pure white color) to simulate the real universe. We included several graphs with different angles and zooming to illustrate the result:


Z (North Pole)
Y (R.A. 6 Hours)
X (Vernal Equinox)

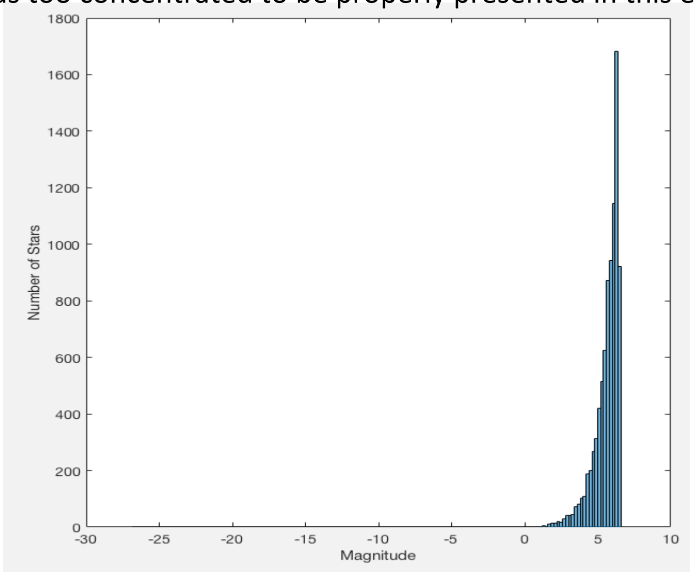

Z (North Pole)
Y (R.A. 6 Hours)
X (Vernal Equinox)


Z (North Pole)
X (Vernal Equinox)
Y (R.A. 6 Hours)

Full MATLAB script for Universe Plotting:

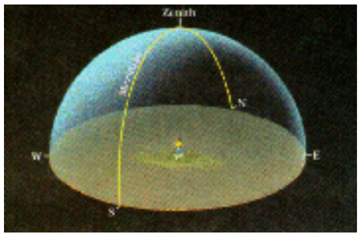
```
stars = readtable('hyg3.csv');
vstars = stars(stars.mag <= 6.5, :);
cvstars = vstars(vstars.dist < 100000, :);
cvstars = sortrows(cvstars, 'mag');
cvstars = cvstars(2:end,:);
c = linspace(1, .2, length(cvstars.id));
c = [c' c' c'];
scatter3(cvstars.x,cvstars.y,cvstars.z, 5, .2+c, 'filled');
xlabel('X (Vernal Equinox)');
ylabel('Y (R.A. 6 Hours)');
zlabel('Z (North Pole)');
set(gca,'Color',[0 0 0]);
```

The reason that we used a linear distribution of brightness was because the distribution of magnitude value was too concentrated to be properly presented in this enormous space:



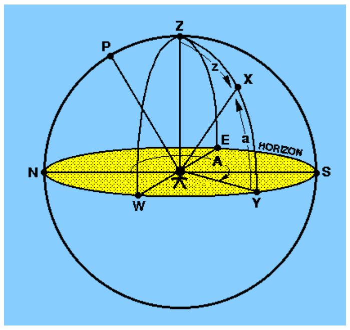
Part 3: Observing from Earth

Although a visual model of our universe was beautiful, we were more interested in how the stars look like when we stood on Earth. Therefore, we would need to build a star chart, which demonstrates stars' locations on a celestial (hemi)sphere relative to observer's location and time. Note that distance does not matter here because humans perceive in Euclidean space and when distances are much larger than apparent shape, the mind draws a sphere. Additionally, the apparent magnitude of a star incorporates distance (we assumed a sphere with radius 10 parsecs here, which is the benchmark distance for absolute magnitude).

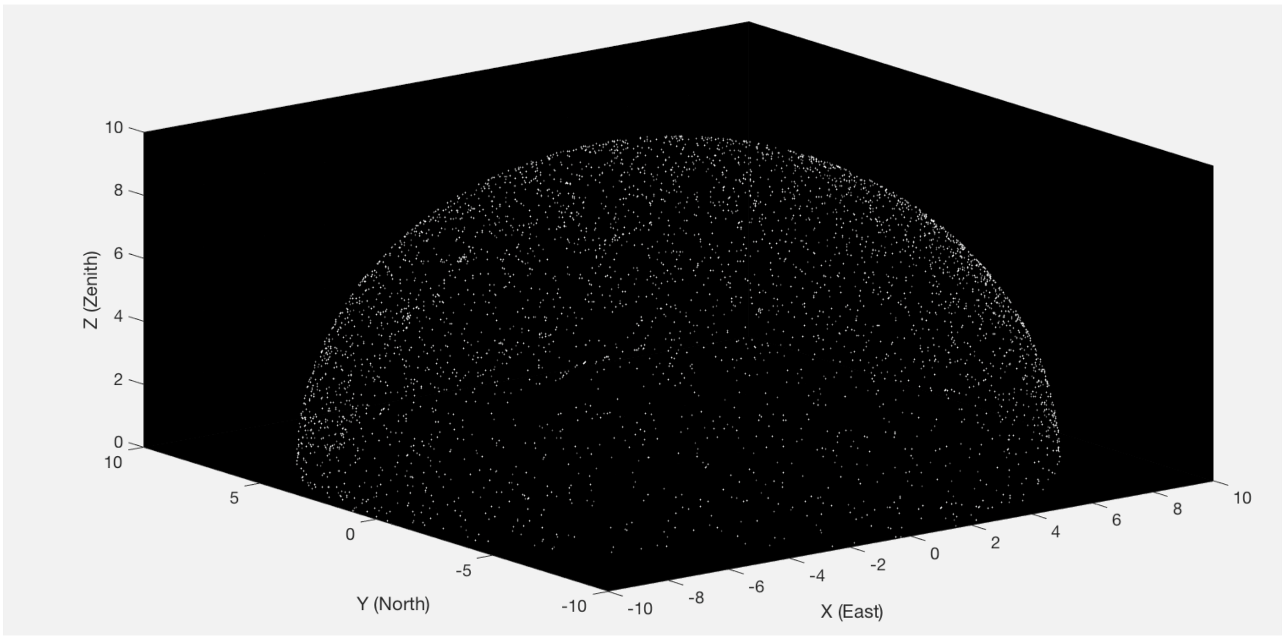


To achieve our goal, we needed to build a horizontal coordinate system with 2 coordinates **azimuth** (A) and **altitude** (a): azimuth is the angular distance measured clockwise along the observer's horizon from a specified reference point (true north) to the intersection with a great circle drawn from the observer's zenith through the celestial body of interest; altitude is the angular distance of a celestial body above or below the observer's horizon, measured along a great circle passing through the body and the observer's zenith. To compute these 2 numbers, we need a star's right **ascension** and **declination**, which come from the database, **Julian date**, a continuous count of days and fractions since noon Universal Time on January 1st, 4713 BCE (on the Julian calendar), and the observer's **longitude** and **latitude**.

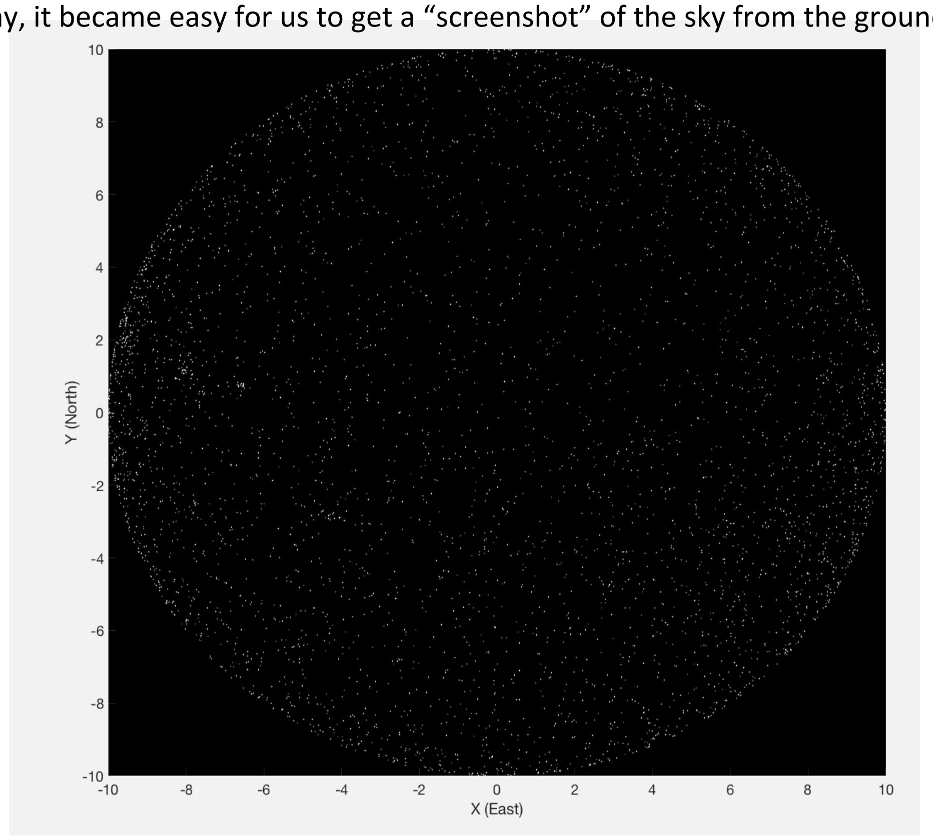
One could obtain Julian date using this online Julian data converter, and we chose 00:00:00 UT1 on January 1st, 2017 as our testing date, which is 2457754.5 in Julian date. We could then obtain Greenwich Apparent Sidereal Time and Local Hour Angle. We also chose Olin Library's geographical location as the observer's longitude and latitude (38.6486°N 90.3078°W). Combining all data together, we could compute coordinates azimuth and altitude for each star. The system is demonstrated in the graph below, where Z represents the zenith and P the north celestial pole:



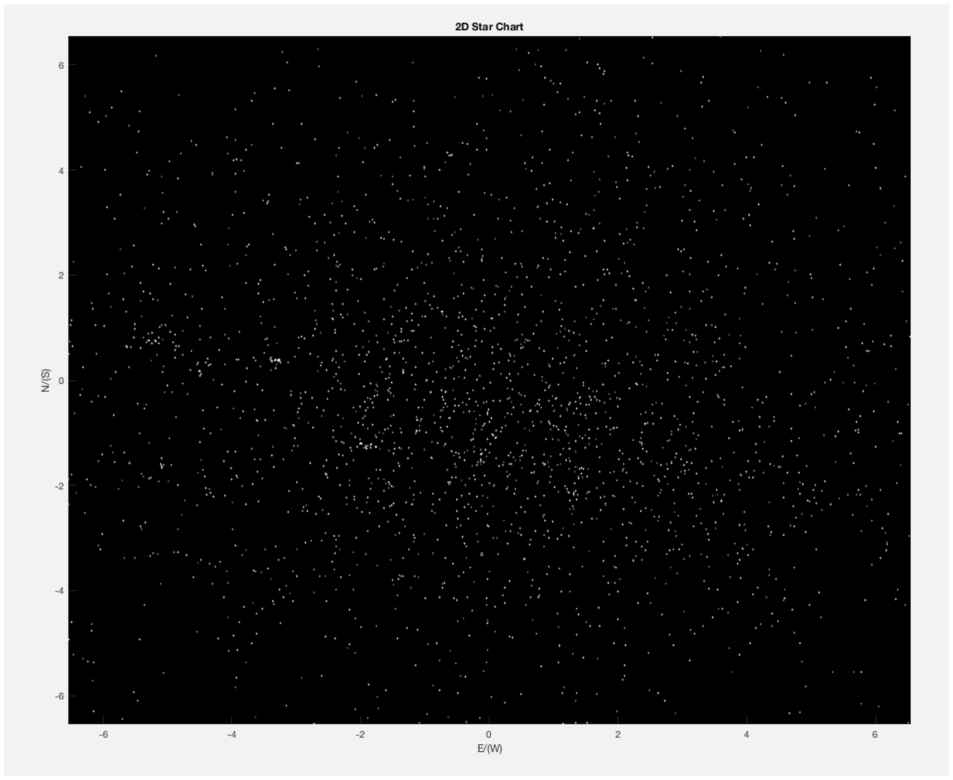
Using these data, we plotted the stars on a sphere, and we hided the ones with negative altitude:



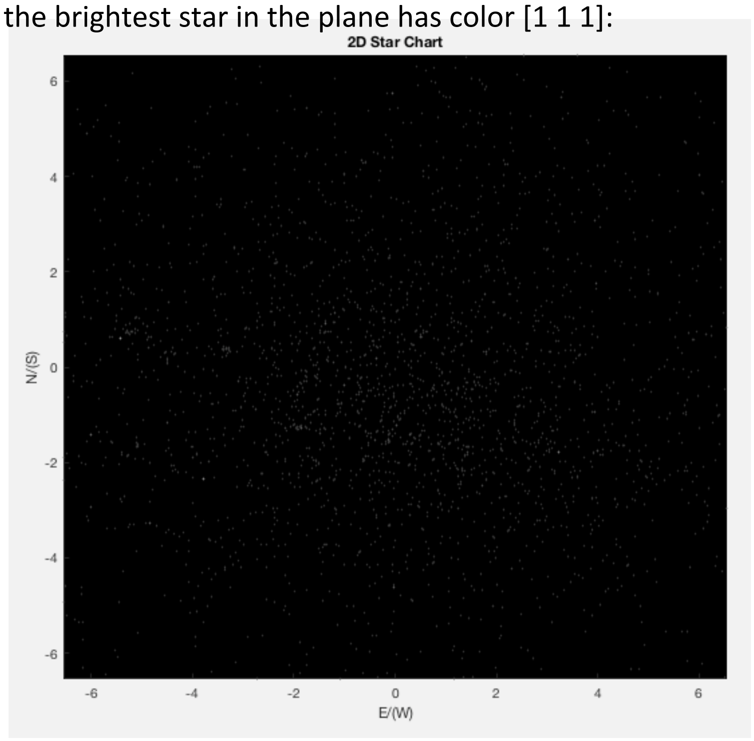
In this way, it became easy for us to get a “screenshot” of the sky from the ground:



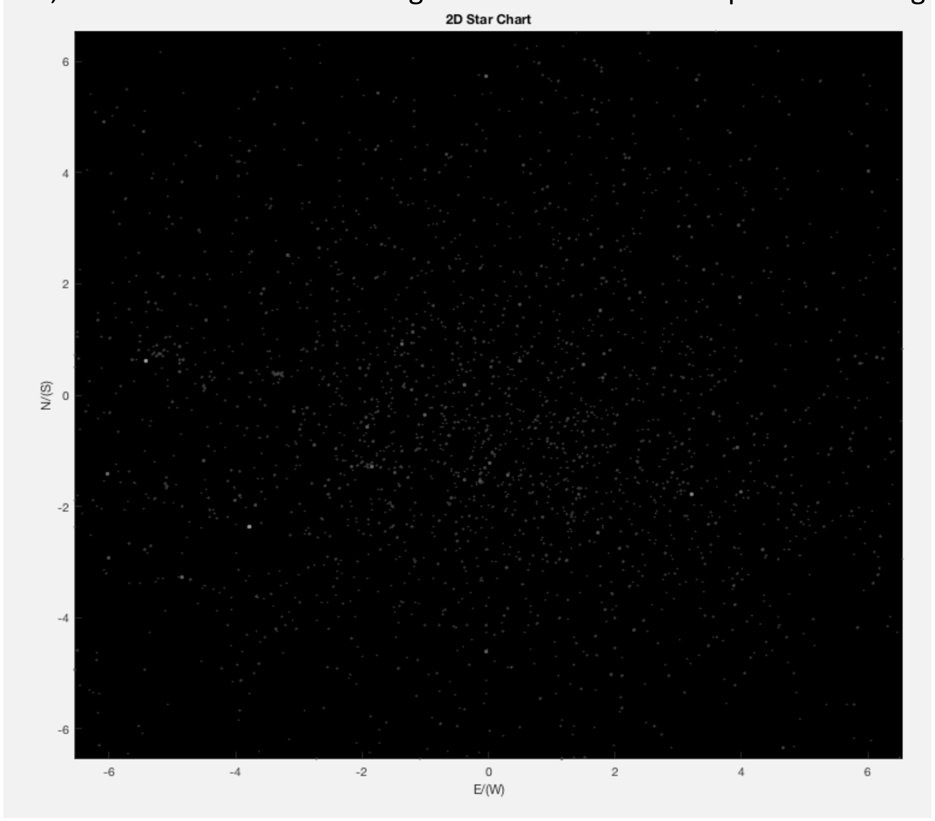
Since the observer would be standing at the center of this graph, we assumed a field of view of 120° (horizontal) * 120° (vertical) pointing at the Zenith and could therefore project the stars onto a rectangular plane:



To fully simulate the appearance of stars, we need to calculate the relative brightness among them to replace the linearly distributed colors. A star is $\sqrt[5]{100}$ brighter than another star whose apparent magnitude is 1 larger. We therefore assumed all visible stars have color no dimmer than [.2 .2 .2] and the brightest star in the plane has color [1 1 1]:



To further highlight the brighter stars, we made sizes of the scatter points to be proportional to their magnitude, with the smallest size being 5 – this much better replicates our night sky:



Full MATLAB script for 3D and 2D Star Charts Plotting:

```
% filtering
stars = readtable('hyg3.csv');
vstars = stars(stars.mag <= 6.5, :);
cvstars = vstars(vstars.dist < 100000, :);
cvstars = sortrows(cvstars, 'mag');
cvstars = cvstars(2:end,:);

% inputs
JD = 2457754.5;
la = 38.6486; lo = -90.3078;
r = 10;
ah = 2*pi/3; av = 2*pi/3;

% compute azimuth A and altitude a
D = JD - 2451545.0;
GMST = 18.697374558 + 24.06570982441908*D;
L = 280.47 + 0.98565*D;
omega = 125.04 - 0.052954*D;
obliquity = 23.4393 - 0.0000004*D;
nutation = -0.000319*sind(omega) - 0.000024*sind(2*L);
GAST = GMST + nutation * cosd(obliquity);
LHA = (GAST - cvstars.ra) * 15 + lo;
a = asin(cosd(LHA).*cosd(cvstars.dec)*cosd(la) + sind(cvstars.dec)*sind(la));
A = atan2(-sind(LHA), (tand(cvstars.dec)*cosd(la)-sind(la)*cosd(LHA))) + pi;

% compute Cartesian coordinates
x = r*cos(a).*sin(A); x = x(a > 0);
y = r*cos(a).*cos(A); y = y(a > 0);
z = r*sin(a);         z = z(a > 0);
mag = cvstars.mag(a > 0);

% determine colors for 3D chart
c = linspace(1, .2, length(x));
c = [c' c' c'] + .2;

% 3D plotting
scatter3(x, y, z, 5, c, 'filled');
xlabel('X (East)');
ylabel('Y (North)');
zlabel('Z (Zenith)');
set(gca,'Color',[0 0 0]);
axis([-r,r,-r,r,0,r]);
title('3D Star Chart');

% compute projected 2D coordinates
% h*h + xl*xl + yl*yl = r*r;
h = r / sqrt(1 + tan(ah/2).^2 + tan(av/2).^2);
xl = h * tan(ah/2);
yl = h * tan(av/2);
x = x(z >= h); y = y(z >= h); mag = mag(z >= h); z = z(z >= h);
x = x*h./z; y = y*h./z;
y = y(x >= -xl & x <= xl); z = z(x >= -xl & x <= xl); mag = mag(x >= -xl & x <= xl); x = x(x >=
-xl & x <= xl);
x = x(y >= -yl & y <= yl); z = z(y >= -yl & y <= yl); mag = mag(y >= -yl & y <= yl); y = y(y >=
-yl & y <= yl);

% determine colors and dot sizes for 2D chart
mrange = max(mag) - min(mag);
mag = mag - min(mag);
sz = 5 + 10 * (1 - mag / mrange);
mag = 100.^.2.^(mag);
c = .8./mag+.2;
c = [c c c];

% 2D plotting
figure
scatter(x, y, sz, c, 'filled');
xlabel('E/(W)');
ylabel('N/(S)');
set(gca,'Color',[0 0 0]);
axis([-xl,xl,-yl,yl]);
title('2D Star Chart');
```

This research log, all scripts, and data files are available on GitHub.