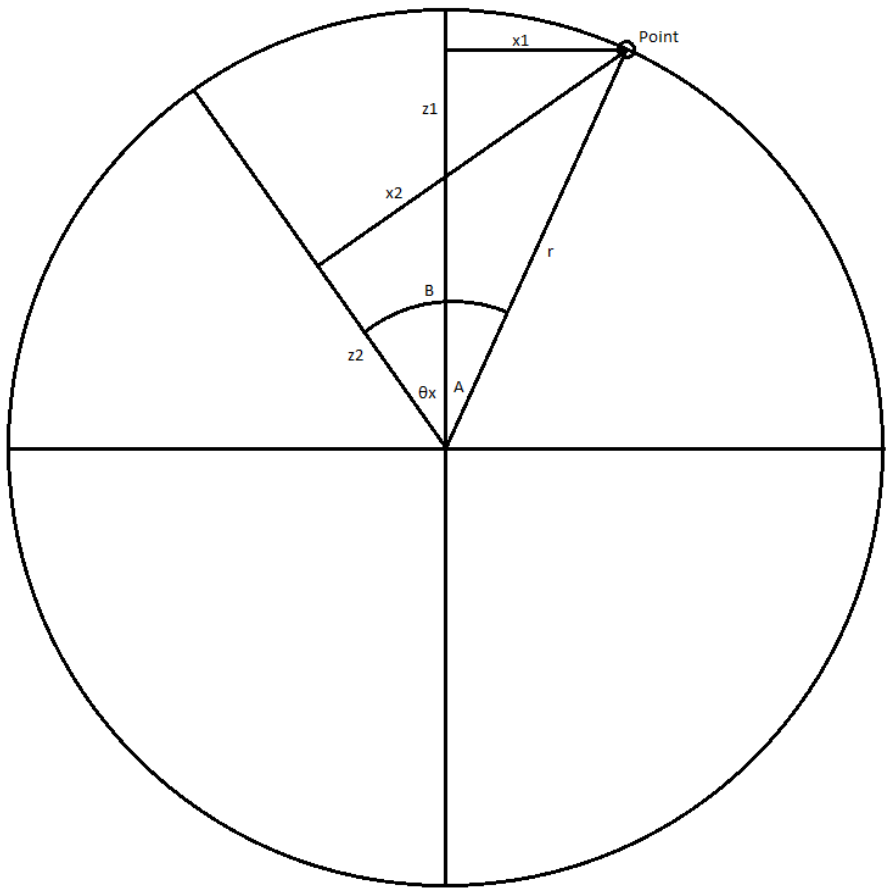
Rotation of 3D Points on Both Axis

Figure 1:



$B=A+\theta x$

$\sin A=x1/r$

$\cos A=z1/r$

$\sin B=x2/r$

$x2=r\sin(A+\theta)$

$x2=r(\sin A\cos\theta+\cos A\sin\theta)$

$x2=r(x1\cos\theta/r+z1\sin\theta/r)$

$x2=x1\cos\theta+z1\sin\theta$

$\cos B=z2/r$

$z2=r\cos(A+\theta)$

$z2=r(\cos A\cos\theta x-\sin A\sin\theta)$
$z2=r(z1\cos\theta/r-x1\sin\theta)$
$z2=z1\cos\theta-x1\sin\theta$

X rotation is always along the same axis, regardless of Y rotation. The axis of y rotation is based off of the X rotation. This means the X rotation must be performed first. If neither rotation is given a set axis, there could be multiple places that the user would end up looking depending on how their mouse gets to a certain point.

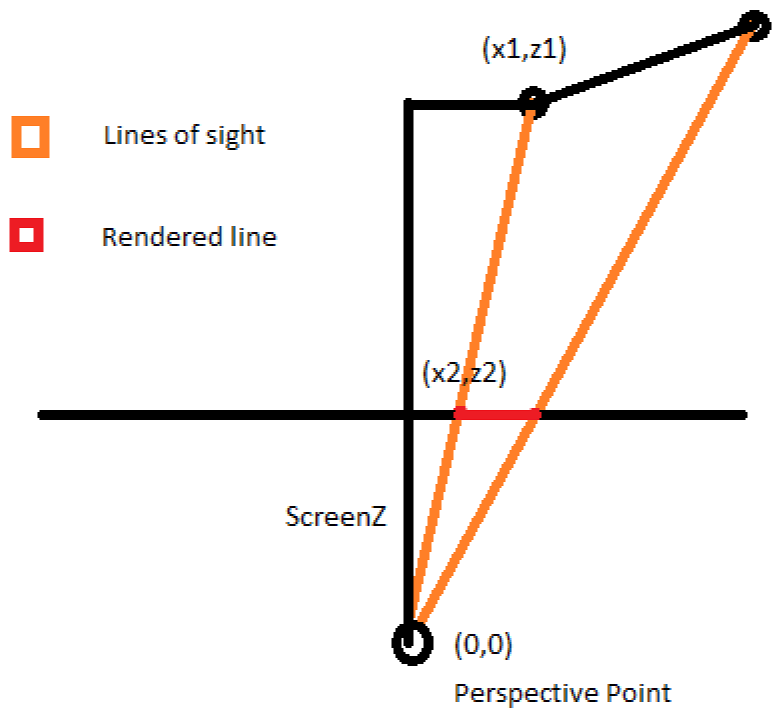
Final Equations:
θ=rotation on the X axis
φ=rotation on the Y axis

Applying X Rotation:
x2=x*Math.cos(θ)+z*Math.sin(θ);
z2=z*Math.cos(θ)-x*Math.sin(θ);

Applying Y Rotation:
y2=y*Math.cos(φ)+z2*Math.sin(φ);
z3=z2*Math.cos(φ)-y*Math.sin(φ);

Rendering of 3D points to 2D plane:

Figure 2:



In order to render points in 3-space to a screen, the place where the lines of sight intersect the arbitrarily positioned plane representing the screen must be found. These points of intersection become the 3d-points 2d representation, and lines between points connected in 3-space are drawn between their 2d counterparts.

z2=screenZ (arbitrary value, changing this changes what is known as Frame of Vision)
x1/z1=x2/z2

Final Equation:
x2=x1*screenZ/z1;