Camera Transforms in Computer Vision

Ricardo Carrillo – Applied Computing / Physics

Advisors: Dr. Laurie Anderson, Dr. Warren Buck, Dr. Tyler Folsom

University of Washington - Bothell: School of STEM

What is a Camera Transform?

A camera transform is a set of mathematical procedures needed in order to transform from one coordinate system to another. Computer vision uses multiple coordinate systems, but need a way to communicate spatial data between them.

Coordinate Systems and Transforms

- GPS (Spherical)
- World Coordinate System (3D Cartesian, aligned with NS/WE)
- Camera Coordinate System (3D Cartesian, aligned with camera)
- Camera Plane Coordinate System (2D Cartesian, projected to screen)

- Haversine Transform (GPS to World)
- Camera Matrix (World to Camera Transform)
- Camera Projection (Camera to Camera Plane Transform)

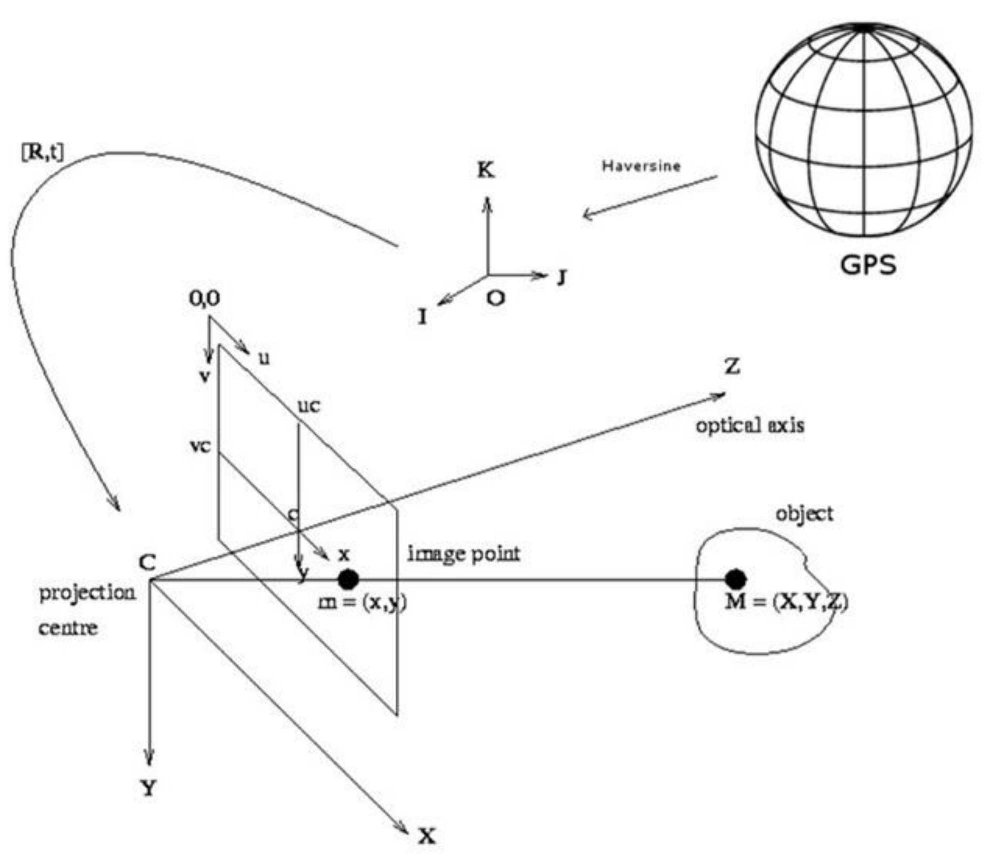


Figure 1. Diagram of Coordinate system and transforms

The Pinhole Camera Model

Computer vision systems operate under the pinhole camera model. Light coming in from a 3D source is projected on to a 2D plane, called the image plane.

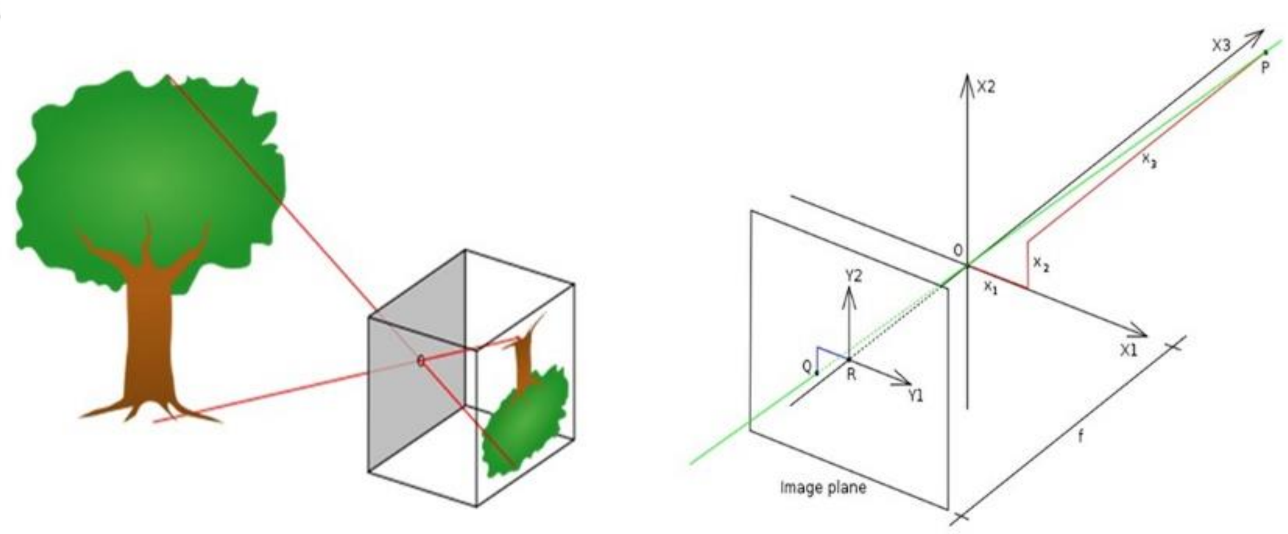
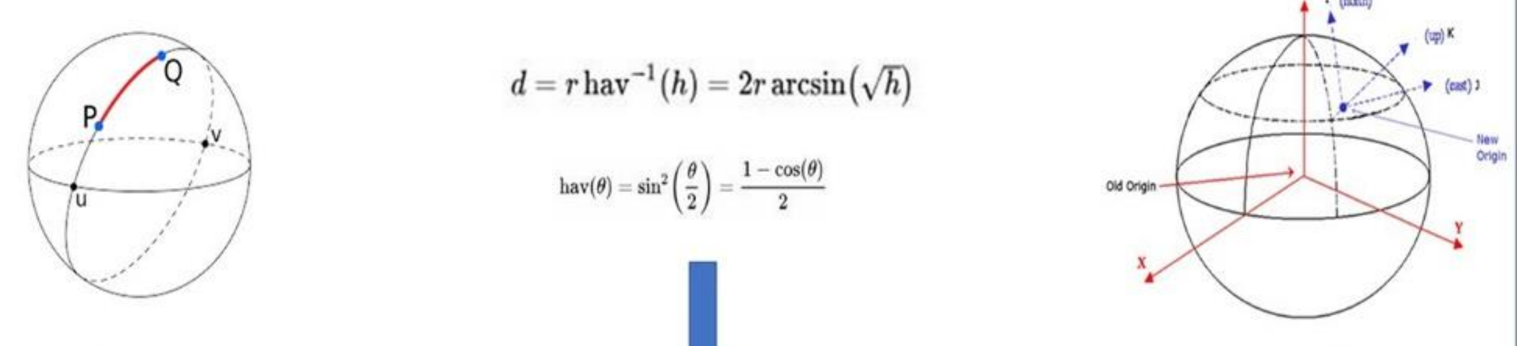


Figure 2. Example and Coordinate setup of pinhole camera model

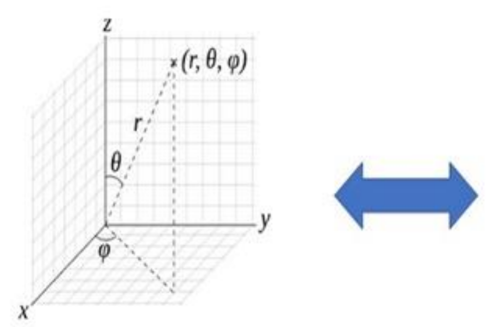
Haversine: GPS to World Coordinate Transform

The Haversine transform maps spherical GPS Coordinates to flat cartesian coordinates centered at Earth's core.



$$d = r\,\mathrm{hav}^{-1}(h) = 2r\arcsin\left(\sqrt{h}\right)$$

$$\mathrm{hav}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1-\cos(\theta)}{2}$$



$$x = r * cos(lat) * cos(lon)$$

$$y = r * cos(lat) * sin(lon)$$

$$z = r * sin(lat)$$

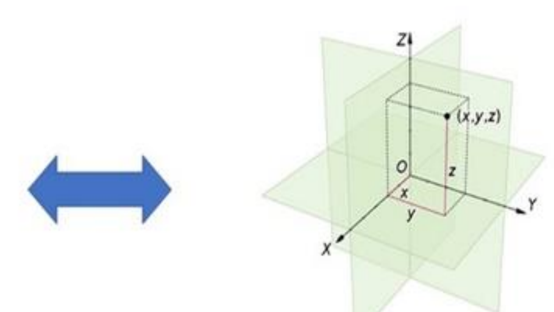


Figure 3. Equations for lat/long derived from haversines.

World to Camera Transform: Camera Matrices [R|T]

The Camera Matrix is a 4x4 matrix that contains all information necessary to move from the World Coordinate System to the Camera Coordinate system. It is a composition of rotation and translation matrices .

a = pitch b = roll c = yaw

$$\begin{pmatrix} Oi \\ Oj \\ Ok \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & h \\ 0 & 1 & 0 & k \\ 0 & 0 & 1 & l \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} cos(a) & -sin(a) & 0 & 0 \\ -sin(a) & cos(a) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -cos(b) & sin(b) & 0 \\ 0 & -sin(b) & -cos(b) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} cos(c) & 0 & sin(c) & 0 \\ 0 & 1 & 0 & 0 \\ sin(c) & 0 & -cos(c) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Cx \\ Cy \\ Cz \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} Cx \\ Cy \\ Cz \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{(cos(2c)+1)}{2cos(c)} & 0 & sin(c) & 0 \\ 0 & 1 & 0 & 0 \\ sin(c) & 0 & -cos(c) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{-cos(2b)-1}{2cos(b)} & -sin(b) & 0 \\ 0 & sin(b) & -cos(b) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{cos(2a)+1}{cos(a)+cos(3a)} & \frac{sin(a)}{cos(2a)} & 0 & 0 \\ \frac{sin(a)}{cos(2a)} & \frac{cos(a)}{cos(2a)} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -h \\ 0 & 1 & 0 & -k \\ 0 & 0 & 1 & -l \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Oi \\ Oj \\ Ok \\ 1 \end{pmatrix}$$

Figure 4. The Camera Matrix and Inverse Camera Matrix

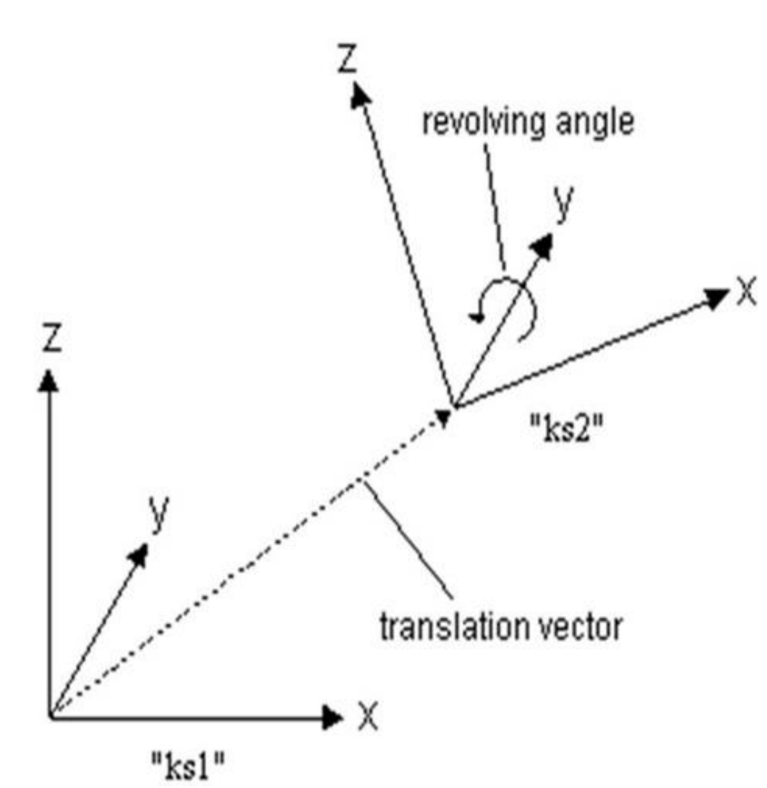


Figure 5. Example Camera Matrix Transform

Projection to Camera Plane

The coordinate on the camera plane is the location where the ray connecting camera center and a point in camera coordinates intersects the camera plane.

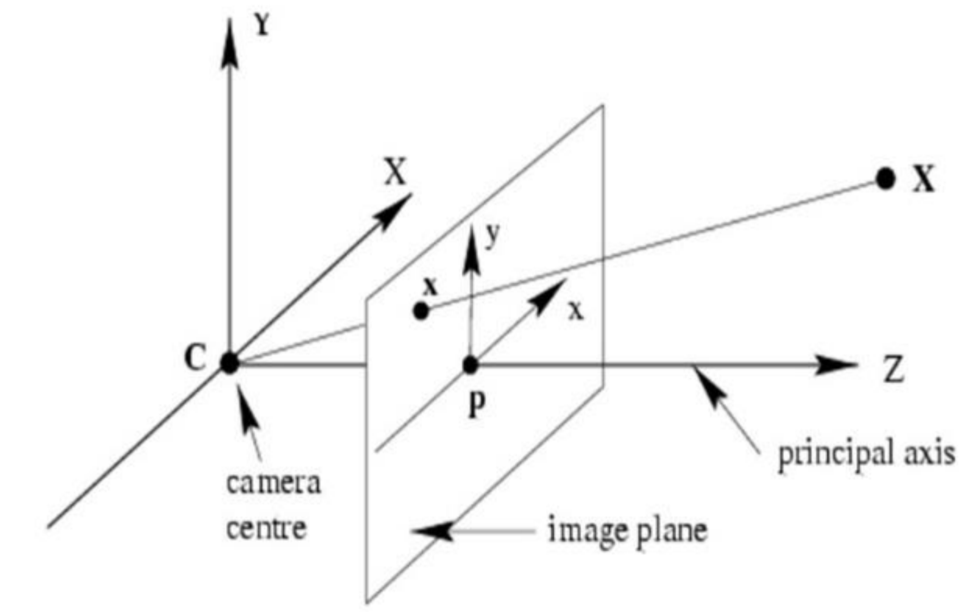


Figure 6. Projecting a 3D point on to 2D camera Plane

Correcting Camera Distortion

No camera is perfect, all have measurable tangential, radial distortion and optical aberration which throws off image plane projection. Software calibration is necessary to produce an undistorted image. Calibration is done through reference object.

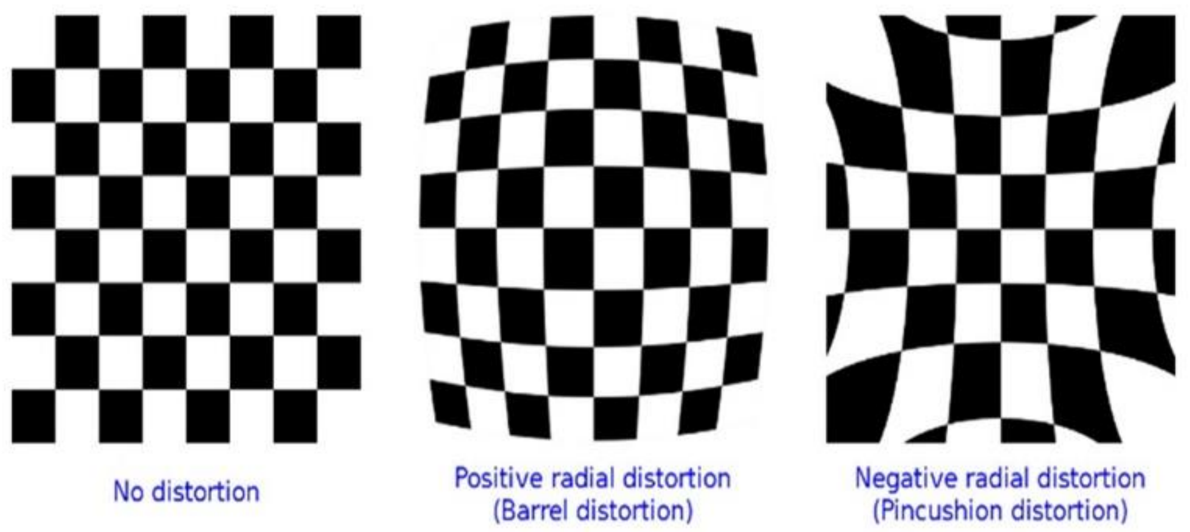


Figure 7. Example Camera Distortion and Corrected Image

Issues Encountered/Continuing Research

- Distance information is lost when projecting to the 2D Camera Plane.
- LIDAR or binocular vision needed for reprojection to 3D.
- Camera calibration is computationally expensive in real time.
- Coordinate systems are defined differently between vision software.
- Current implementation does not use camera calibration.