Appendix A:
The Equations of Homogeneous Stereographics (e.g. qVP's)

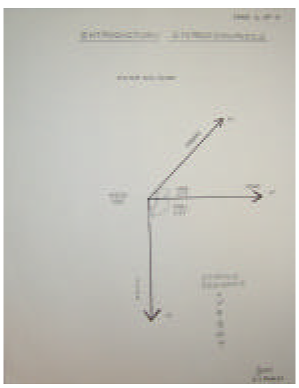
Jeff Setterholm **2009.12.11 posted as "qVPMath12-AppendixA-20091211.pdf"**
at www.setterholm.com / Geodesy

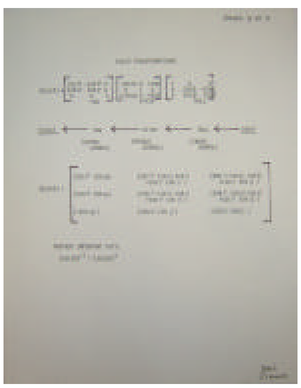
Version 1.2

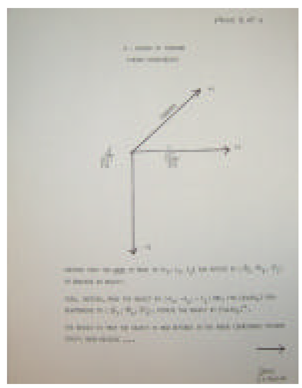
Verson 1.3 2015.10.16 save as: "qVPMath13-AppendixA.pdf"

Sign errors are corrected in red on pages 6 & 7.
& pages 10 is added.

In the early 1980's I was making extensive and productive use of computer-generated stereographic (3-D) images in my engineering work, and taught about 20 of my fellow engineers the core equations. My handout was four pages of notes; viewed as thumbnails images, the originals looked like this:


1. coordinate frame


2. direction cosines


3.eye locations

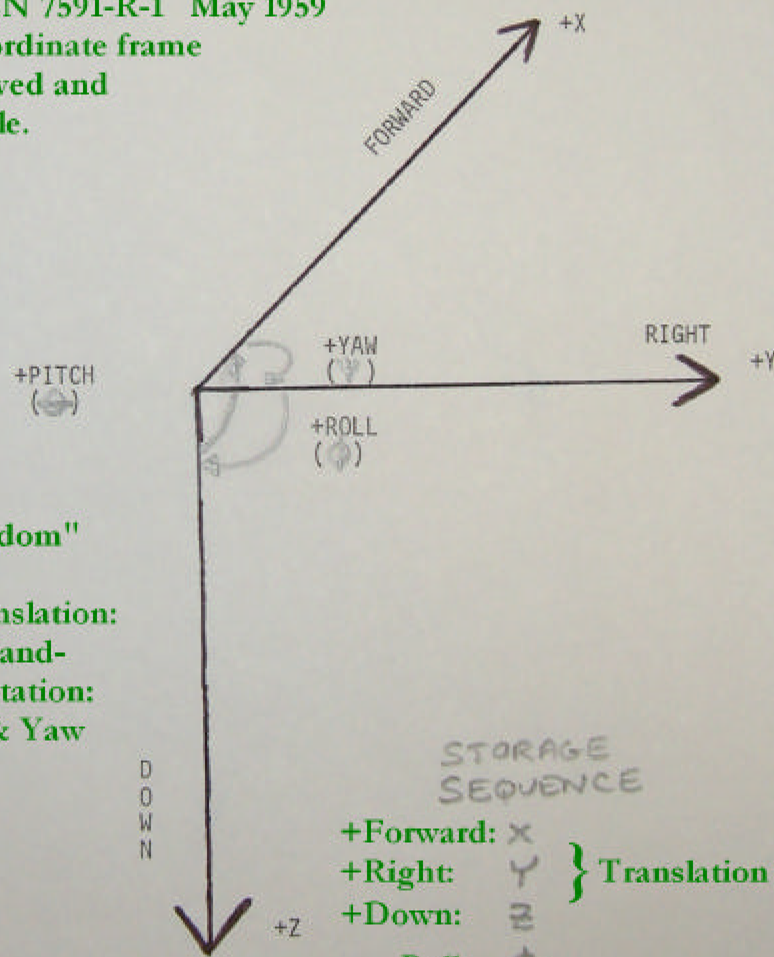

4. stereo projection

This appendix expands upon my original handout:

INTRODUCTORY STEREOGRAPHICS

Aircraft Axis System

I refer to the coordinates described on pages one & two as:
"Standard Flight Simulation Coordinates"
Ref. e.g.: "Simulation of Aircraft" page 6
NAVTRADEVCEN 7591-R-1 May 1959
This right-handed coordinate frame
was wisely conceived and
is widely applicable.



A rigid body has
"Six Degrees-Of-Freedom"
= "6-DOF"
three degrees of translation:
e.g. X,Y, &Z -and-
three degrees of rotation:
e.g. Roll, Pitch, & Yaw

STORAGE SEQUENCE

+Forward: x
+Right: y **} Translation**
+Down: z

Roll: ϕ
Pitch: θ **} Rotation**
Yaw: ψ

JMS
27 AUG 81

"Angles" make rotations easy to communicate verbally. "Direction cosines" are an equivalent [3x3] matrix of numbers that is very useful for computation. The numbers are the cosines of the angles between each input axis and each output axis.

Direction cosines = EULER TRANSFORMATIONS

$$[\text{EULER}] = \underbrace{\begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1. \end{bmatrix}}_{\text{YAW}} \underbrace{\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}}_{\text{PITCH}} \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}}_{\text{ROLL}}$$

OUTPUT ← YAW ← PITCH ← ROLL ← INPUT

(OUTER GIMBAL) (MIDDLE GIMBAL) (INNER GIMBAL)

$$[\text{EULER}] = \begin{bmatrix} (\cos\psi\cos\theta) & (\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi) & (\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi) \\ (\sin\psi\cos\theta) & (\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi) & (\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi) \\ (-\sin\theta) & (\cos\theta\sin\phi) & (\cos\theta\cos\phi) \end{bmatrix}$$

ANOTHER IMPORTANT FACT:

$$[\text{EULER}]^{-1} = [\text{EULER}]^{T}$$

An example matrix multiply [2x2]:

$$\begin{bmatrix} a*e+c*f, a*g+c*h \\ b*e+d*f, a*g+d*h \end{bmatrix} = \begin{bmatrix} a, c \\ b, d \end{bmatrix} * \begin{bmatrix} e, g \\ f, h \end{bmatrix}$$

A "homogeneous transform" is a [4x4] matrix that multiplies like a regular matrix, but wherein the bottom row of numbers are used in various and often amazingly useful ways. The 6-DOF of a rigid body can be described as the following [4x4]:

Direction			, X
Cosines			, Y
[3x3] above			, Z
0. ,	0.,	0.,	, 1.

The upper-left 3x3 is the angular information, the right upper 3x1 is the translation (X,Y,Z) information, and the lower right 1. makes the matrix invertible (hence bi-directional). Wow!

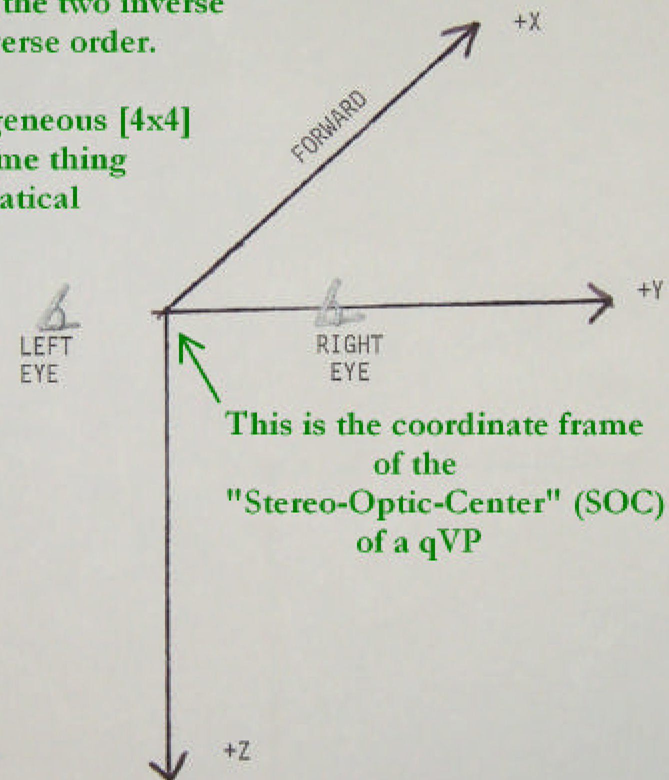
JMS
27 AUG 81

6 - DEGREE OF FREEDOM

STEREO COORDINATES

When translating and direction cosine rotating are two different operations, reversing the sequence involves performing the two inverse operations in the reverse order.

Inverting the homogeneous [4x4] accomplishes the same thing via a purely mathematical operation.



This is the coordinate frame of the "Stereo-Optic-Center" (SOC) of a qVP

SUPPOSE THAT YOU <u>WANT</u> TO MOVE TO (E_x, E_y, E_z) AND ROTATE BY (ϕ_E, θ_E, ψ_E) TO OBSERVE AN OBJECT.

THEN, INSTEAD, MOVE THE OBJECT BY ($-E_x$, $-E_y$, $-E_z$) AND, FOR [$EULER_E$] CORRESPONDING TO (ϕ_E, θ_E, ψ_E), ROTATE THE OBJECT BY $[EULER_E]^{-1}$.

THE RESULT IS THAT THE OBJECT IS NOW DEFINED IN THE ABOVE COORDINATE SYSTEM; THAT'S GOOD BECAUSE ...

→

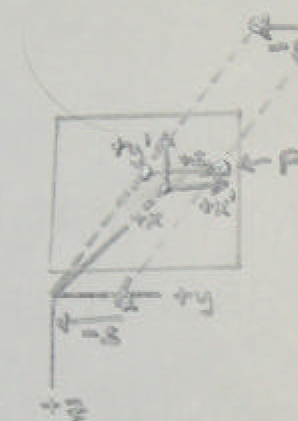
JMS
27 AUG 81

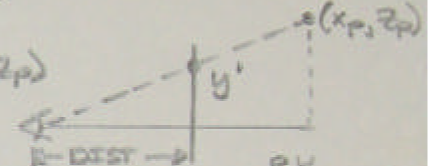
The upper 3 components component of a homogeneous vector are divided by the 4th component. Thus, the 4th row of the H-matrix manages the division factor, and hence -by design- supports projection. (Silicon Graphics' "OpenGL" uses many, many H-matrices.)

ELEMENTARY STEREO

PROJECTION

- IS SIMPLE IF DONE IN HEAD COORDINATES





PLACE IMAGE PLANE ⊥ TO X-AXIS AT A FIXED DIST.

BY SIMILAR TRIANGLES

$$\frac{P'y}{Dist.} = \frac{-Zp}{Xp}$$

$$P'y = \frac{-Zp \times Dist.}{Xp}$$

LIKEWISE

SET EYE SPACING TO ± S

RIGHT EYE: $$\frac{P'x}{Dist} = \frac{(Yp-s)}{Xp}$$

$$P'x = \frac{(Yp-s) \times Dist.}{Xp}$$

CHANGE SIGNS FOR LEFT EYE

THEN--FOR EACH POINT

1. SHIFT ± S IN Y — $P'y = P'y$
2. PROJECT - BY SIM. TRIANGLES — $P'y = \frac{-Zp \times Dist.}{Xp}$
3. SHIFT ± S IN x'

PROJECTION COMPLETE

$P'x = P'x + S$

$$P'x = \frac{(Yp - S) \times Dist.}{Xp} + S$$

In the present coordinates, rotation and translation have been removed. Folding "Xp" into the H-matrix 4th row (where the division factor goes) will accomplish the projection.

To form the qVP - we want to map (X,Y,Z) to (P'xleft,P'xright,P'Y)

To use the qVP - we want to map (P'xleft,P'xright,P'Y) to (X,Y,Z).

(P'xleft,P'xright,P'Y) will need to be re-expressed in pixels, which will also be done with H-matrices.

JMS
27 AUG 81

For simplicity of notation, let the 2-D P' (X,Y) coordinates be re-expressed as coordinates (U,V), and where: P'xleft becomes UL, and
P'xright becomes UR, and
P'y becomes V .

In homogeneous form use the lower-case symbols (uL,uR,v,w1), where w1 is the *homogeneous division factor*.

Let P coordinate components (Px,Py,Pz) be expressed as (X,Y,Z)
In homogeneous form use the lower-case symbols (x,y,z,w2), where w2 is again the *homogeneous division factor*.

Homogenizing (UL,UR,V) and (X,Y,Z) the challenge is to define [H] such that

```
| uL/w1 |   | uL |         | X |   | a, e, i, m|   | x |
| uR/w1 | = | uR | = [H]*  | Y | = | b, f, j, n| * | y |
| v /w1 |   | v  |         | Z |   | c, g, k, o|   | z |
            | w1 |         | 1.|   | d, h, l, p|   | w2|
```

Note that w2=1., because that's how vectors are initially homogenized.

Explaining the solution, starting at the bottom row…

x is going to be the division factor:
set `d=1., h=0., l=0.   , p=0.`

v comes from –z*Dist:
set `c=0., g=0., k=-Dist, o=0.`

uR comes from `(y-S)*Dist+S*x`
set `b=+S, f=Dist, and j=0., n=-Dist*S`
Roughly speaking, offsets ^ are managed
by the upper right 3x1 on page 3 of 9.

uL comes from `(y+S)*Dist-S*x`
set `a=-S, e=Dist, and i=0., m=+Dist*S`

```
      | -S , Dist,   0. , +Dist*S |
So H= |  S , Dist,   0. , -Dist*S |
      |  0.,  0. , -Dist,   0.    |
      |  1.,  0. ,   0. ,   0.    |
```

Performing the homogeneous multiplication H*(**x**,**y**,**z**,w2) transpose yields:

```
uL = -S*x + Dist*y          +(+Dist*S)*w2
uR =  S*x + Dist*y          +(-Dist*S)*w2
V  =               -Dist*z
W1 =    x
```

Note that `x=X*w2, y=Y*w2, and z=Z*w2.`

```
So:
uL = -S*X*w2 +Dist*Y*w2           +(+Dist*S)*w2
uR =  S*X*w2 +Dist*Y*w2           +(-Dist*S)*w2
V  =                    -Dist*Z*w2.
W1 =    X*w2
```

Hence, since the w2's cancel during division:

```
UL= uL/w1 =-S + Dist*(Y+S)/X   =P'xleft
UR= uR/w1 = S + Dist*(Y-S)/X   =P'xright
V = v /w1 =   - Dist* Z   /X   =P'y
```

…which proves that the desired result
has been achieved
by the homogeneous matrix!

The matrix is invertible (= bi-directional) because the **depth information is not lost** in stereo projection, but rather stored in the stereo image coordinates. In a qualitative way, the human stereo visual system reconstructs that depth information continuously *in the twinkling of a pair of eyes*.

Consider deriving the reverse mapping yourself, or invert the H-matrix computed above to find the reverse map.

Conversion between U,V space and pixels (for undistorted images):

Homogeneous matrices do a great job of performing 1,2,OR 3 **Y=m*X+b operations** simultaneously. So, knowing how to compute m and b directly is useful:

Suppose that a straight line graph passes through the point (X1,Y1)=(A,C) and also the point (X2,Y2)=(B,D) then

m =(D-C)/(B-A) and
b =C- A*(D-C)/(B-A)
=(B*C-A*C-A*D+A*C)/(B-A)
=(B*C-A*D)/(B-A)

Horizontally, for the left eye (and in a like manner the right eye, and vertically):
Set A= the left limit of U (e.g. the minimum value) and
B= the right limit of U (e.g. the maximum value)
Set C= the left-eye's-pixel value at the left limit (e.g. 0.) and
D= the left-eye's-pixel value at the right limit
(e.g. the total number of horizontal pixels)
Then solve for **m**(left &horizontal) and **b**(left &horizontal) and likewise
Solve for **m**(right&horizontal) and **b**(right&horizontal) and likewise
Solve for **m**(vertical) and **b**(vertical)

The homogeneous transform becomes:

```
|  mlefth,   0.  ,   0. , blefth  |
|    0.  ,mrighth,   0. , brighth |
|    0.  ,   0.  , mvert, bvert   |
0.,  0.  ,   0.  ,   0. , 1.      |
```

Multiplying a series of matrix operations together to form a single matrix is called **"concatenation"**. The product of a series of invertible matrices remains itself invertible. The matrices described in this write-up are all invertible… which accounts for how qVP's can have a single [4x4] matrix that bi-directionally maps pixels to GPS (X,Y,Z).

Once H-matrices involving non-6-DOF transformations are concatenated, the coefficients tend to cease to be intuitively recognizable; that transformational information can be thus compressed and bi-directionalized -*beyond & outside human intuition with no human intervention*- strikes me as a bit of a miracle.

Removal of distortions from images before turning them into qVP's is necessary,
but is beyond the scope of this appendix.

qVP's can be formed under other geometric assumptions from the simple assumptions that I have demonstrated here. This write-up is intended as an introduction to the subject.

Unnecessary complexity interferes with understanding:
Simplifying : elementary stereographics years ago
made it easy to consider: homogeneous stereographics in recent times.

Disclaimers:

1. This is my first try at communicating the analytics of qVP's. &

2. 6-DOF spatial analytics are not trivially easy to "get right"
(properly define & sequence), particularly the first time,
whether or not homogeneous transforms are the tools employed. &

3. Although I mean well, I sometimes make mistakes. &
In fact, the price of pursuing cutting-edge R&D is being wrong almost all the time;
physical reality (~"Mother Nature") acts as an impartial but relentless judge.

4. This write-up is positively and certainly not fool-proof:

This concludes corrections of version 1.2

Version 1.3 Additions to Appendix A: 2015.10.16

The following equations compute the H matrix described at the bottom of page three.

H describing a rigid body's Six Degree-Of-Freedom (6DOF) position & attitude:

```
Within a "standard flight simulation coordinate system":
  Let the position = (x,y,z), as in drawing #1 and
  let the attitude = (roll,pitch,yaw)  where: +yaw   takes +x toward +y
                                              +roll  takes +y toward +z
   sr=  sine(roll); sp=  sine(pitch); sy=  sine(yaw)  +pitch takes +z toward +x
   cr=cosine(roll); cp=cosine(pitch); cy=cosine(yaw)
                        with gimbal (concatenation) sequence: yaw <-Pitch <-Roll
   H(1,1)= cy*cp; H(1,2)=-sy*cr+cy*sp*sr; H(1,3)= sy*sr+cy*sp*cr;  H(1,4)= x
   H(2,1)= sy*cp; H(2,2)= cy*cr+sy*sp*sr; H(2,3)=-cy*sr+sy*sp*cr;  H(2,4)= y
   H(3,1)=   -sp; H(3,2)= cp*sr;          H(3,3)= cp*cr;           H(3,4)= z
   H(4,1)=    0.; H(4,2)=    0.;          H(4,3)=    0.;           H(4,1)= 1.
```

Numerical example:

```
Position=(  x ,   y , z ,w) =( 1. ,2., 3.,1.)e.g. in meters
Attitude=(roll,pitch,yaw,w) =(10.,20.,30.,1.)degrees - become direction cosines
                             ...which are unitless and independent of "angle" definition!
```

H_{6DOF} =

0.813797681	-0.440969611	0.378522306	1.000000
0.469846310	0.882564119	0.018028311	2.000000
-0.342020143	0.163175911	0.925416578	3.000000
0.000000000	0.000000000	0.000000000	1.000000

A recent advance:

www.setterholm.com /Images includes: "**StPaul-LookingDown-RedGB3D.jpg**"
Use Red/Cyan glasses for 3D viewing.
see: "RedGB3D.pdf" for further information.

The corresponding qVP data structure (which includes a numerical example) applies:
http://ftp.setterholm.com/Geodesy/StPaul/StPaul-LookingDown-qVP-RB.txt