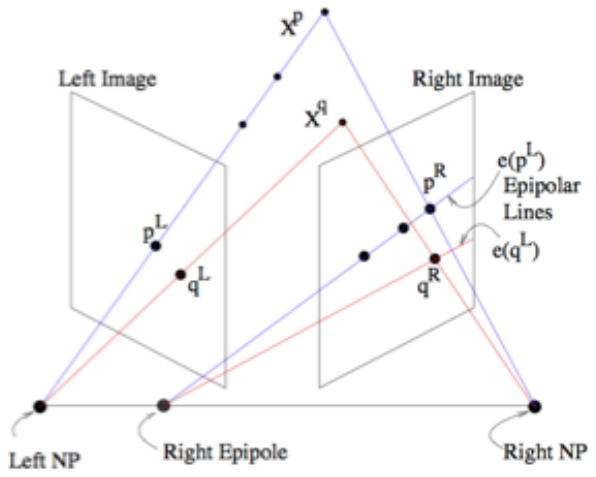
Projection, stereo and panoramic images

X^p and X^q are the physical location of objects.



http://www.cs.toronto.edu/~jepson/csc420/notes/epiPolarGeom.pdf
(note, the image is posted on an educational site and copied here without following up on permissions. Any further use of the image should follow up on the origins and permissions.)

$(X_L)^T * F * X_R = 0$ for any pair or points in the images.

Note: the "Essential matrix" is a matrix used if the camera details are known. The "bifocal tensor", a.k.a. "fundamental matrix" does not need camera details.

NP are the nadir points of the cameras.

The line connecting the left and right NP is the baseline.

The projection of the left nadir, NP is seen as a the left epipole w.r.t. right image. The epipoles may or may not be within the border of the images.

The projection of X^p to left nadir, NP is seen as an epipole line in the right image. If more than one epipole line is present in the right image, they converge at the right epipole (which might not be within the boundaries of the image).

Once the points in the left image, X_L are matched with points in the right image, X_R, if there are at least 7 points, one can determine the "bifocal tensor",
a.k.a. "fundamental matrix, relating the points in the 2 images using a 3x3 matrix of rank 2.

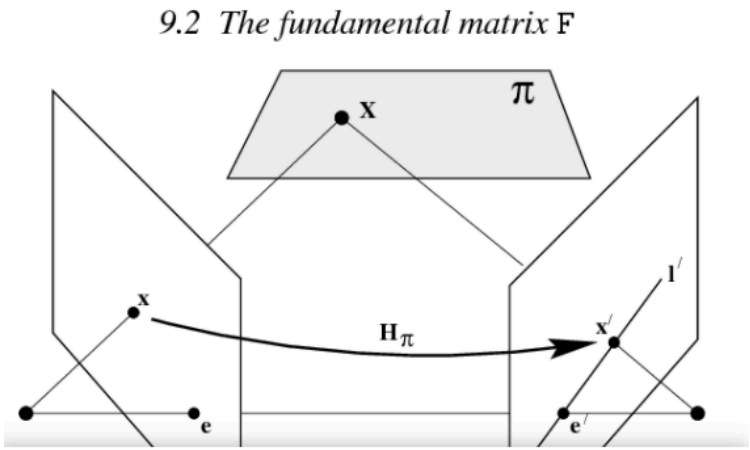


Fig. 9.5. A point x in one image is transferred via the plane π to a matching point x' in the second image. The epipolar line through x' is obtained by joining x' to the

Multiple View Geometry in Computer Vision by Wrobel, 2001

2D transformations

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

2D Translation. $x' = x + t$

$$x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

$$\bar{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \bar{x}$$

whereI is the (2×2) identity matrix

2D Affine: $x' = A\bar{x}$

$$x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \bar{x}.$$

preserves parallel property of lines.

2D Rotation + translation (a.k.a. rigid body motion, a.k.a. 2D euclidian): $x' = Rx+t$

$$x' = \begin{bmatrix} R & t \end{bmatrix} \bar{x} \qquad R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

2D Scaled rotation (a.k.a. similarity transform): $x' = sRx + t$

$$x' = \begin{bmatrix} sR & t \end{bmatrix} \bar{x} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \end{bmatrix} \bar{x},$$

preserves angles between lines.

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} I \mid t \end{bmatrix}_{2\times3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} R \mid t \end{bmatrix}_{2\times3}$	3	lengths	
similarity	$\begin{bmatrix} sR \mid t \end{bmatrix}_{2\times3}$	4	angles	
affine	$\begin{bmatrix} A \end{bmatrix}_{2\times3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times3}$	8	straight lines	

Table 2.1 Hierarchy of 2D coordinate transformations. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The 2×3 matrices are extended with a third $[0^T\ 1]$ row to form a full 3×3 matrix for homogeneous coordinate transformations.

2D Projective (a.k.a. perspective transformation, homography): $\tilde{x}' = \hat{H}\tilde{x}$

Note that $\tilde{H}$ is homogeneous, i.e., it is only defined up to a scale, Two H matrices that differ only by scale are equivalent.

The resulting homogeneous coordinate $\tilde{x}'$ must be normalized in order to obtain an inhomogeneous result:

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}} \quad \text{and} \quad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}}.$$

preserves straightness property of lines.

Planar surface flow: Bilinear interpolant:

$x' = a0 + a1x + a2y + a6x^2 + a7xy$

$y' = a3 + a4x + a5y + a7x^2 + a6xy$

planar surface with a small 3D motion

Bilinear interpolant:

$x' = a0 + a1x + a2y + a6xy$

$y' = a3 + a4x + a5y + a7xy$

useful for interpolation of sparse grids using splines , but does not preserve straight lines

3D transformations

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

3D Translation. $x' = x + t$

$$x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

whereI is the (3×3) identity matrix

3D Affine: $x' = Ax^{-}$

$$x' = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{bmatrix} \bar{x}$$

preserves parallel property of lines and planes.

3D Rotation + translation (a.k.a. rigid body motion, a.k.a. 3D euclidian): $x' = Rx+t$

$$x' = \begin{bmatrix} R & t \end{bmatrix} \bar{x}$$

where R is a 3 × 3 orthonormal rotation matrix with $RR^T = I$ and $|R| = 1$.

can describe a rigid motion using
$x' = R(x-c) = Rx - Rc$

where c is the center of rotation (often the camera center).

3D Scaled rotation (a.k.a. similarity transform): $x' = sRx + t$

$$x' = \begin{bmatrix} sR & t \end{bmatrix} \bar{x} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \end{bmatrix} \bar{x},$$

preserves angles between lines and planes.

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} I \mid t \end{bmatrix}_{3\times4}$	3	orientation	
rigid (Euclidean)	$\begin{bmatrix} R \mid t \end{bmatrix}_{3\times4}$	6	lengths	
similarity	$\begin{bmatrix} sR \mid t \end{bmatrix}_{3\times4}$	7	angles	
affine	$\begin{bmatrix} A \end{bmatrix}_{3\times4}$	12	parallelism	
projective	$\begin{bmatrix} \tilde{H} \end{bmatrix}_{4\times4}$	15	straight lines	

Table 2.2 Hierarchy of 3D coordinate transformations. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The 3 × 4 matrices are extended with a fourth $[0^T\ 1]$ row to form a full 4 × 4 matrix for homogeneous coordinate transformations. The mnemonic icons are drawn in 2D but are meant to suggest transformations occurring in a full 3D cube.

3D Projective (a.k.a. perspective transformation, homography): $\tilde{x}' = \hat{H}\tilde{x}$

Note that $\tilde{H}$ is homogeneous, i.e., it is only defined up to a scale, Two H matrices that differ only by scale are equivalent.

The resulting homogeneous coordinate $\tilde{x}'$ must be normalized in order to obtain an inhomogeneous result:

preserves straightness property of lines.

3D transformations

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

3D Rotation represented by a rotation axis and angle (a.k.a. Rodriguez’s formula).

rotation axis and angle as a vector $\omega = \theta \hat{n}$

let $[\hat{n}]$ be the matrix form of the cross product operator with the vector $\hat{n} = (\hat{n}_x , \hat{n}_y , \hat{n}_z)$,

$$[\hat{\boldsymbol{n}}]_\times = \begin{bmatrix} 0 & -\hat{n}_z & \hat{n}_y \\ \hat{n}_z & 0 & -\hat{n}_x \\ -\hat{n}_y & \hat{n}_x & 0 \end{bmatrix}$$

and

$$\boldsymbol{R}(\hat{\boldsymbol{n}}, \theta) = \boldsymbol{I} + \sin\theta[\hat{\boldsymbol{n}}]_\times + (1 - \cos\theta)[\hat{\boldsymbol{n}}]_\times^2$$

good for small rotations

For infinitesimal or instantaneous) rotations and θ expressed in radians, Rodriguez’s formula simplifies to:

$$\boldsymbol{R}(\boldsymbol{\omega}) \approx \boldsymbol{I} + \sin\theta[\hat{\boldsymbol{n}}]_\times \approx \boldsymbol{I} + [\theta\hat{\boldsymbol{n}}]_\times = \begin{bmatrix} 1 & -\omega_z & \omega_y \\ \omega_z & 1 & -\omega_x \\ -\omega_y & \omega_x & 1 \end{bmatrix}$$

from”Bundle Adjustment — A Modern Synthesis” by Triggs et al.:
Rotations should be parametrized using either quaternions subject to $\|q\|^2 = 1$, or local perturbations $R\delta R$ or $\delta R\, R$ of an existing rotation R, where δR can be any well-behaved 3 parameter small rotation approximation, e.g. $\delta R = (I + [\delta r]_\times)$, the Rodriguez formula, local Euler angles, etc.
State updates: Just as state vectors x represent points in some nonlinear space, state updates $x \rightarrow x + \delta x$ represent displacements in this nonlinear space that often can not be represented exactly by vector addition.

3D transformations

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

3D Rotation represented Unit quaternions:

(Shoemake 1985)

a unit length 4-vector whose components can be written as

$q = (q_x, q_y, q_z, q_w)$ or $q = (x, y, z, w)$

Unit quaternions live on the unit sphere $\|q\| = 1$ and *antipodal* (opposite sign) quaternions, q and –q, represent the same rotation

“origin” $q_o = (0, 0, 0, 1)$.

$$q = (v, w) = (\sin\frac{\theta}{2}\hat{n}, \cos\frac{\theta}{2}).$$

to express Rodriguez’s formula:

$$\begin{aligned} R(\hat{n}, \theta) &= I + \sin\theta[\hat{n}]_\times + (1 - \cos\theta)[\hat{n}]_\times^2 \\ &= I + 2w[v]_\times + 2[v]_\times^2. \end{aligned}$$

$$R(q) = \begin{bmatrix} 1 - 2(y^2 + z^2) & 2(xy - zw) & 2(xz + yw) \\ 2(xy + zw) & 1 - 2(x^2 + z^2) & 2(yz - xw) \\ 2(xz - yw) & 2(yz + xw) & 1 - 2(x^2 + y^2) \end{bmatrix}. \quad (2.41)$$

The diagonal terms can be made more symmetrical by replacing $1 - 2(y^2 + z^2)$ with $(x^2 + w^2 - y^2 - z^2)$, etc.

Given two quaternions $q_0 = (v_0, w_0)$ and $q_1 = (v_1, w_1)$,
the *quaternion multiply* operator is defined as:

$$q_2 = q_0 q_1 = (v_0 \times v_1 + w_0 v_1 + w_1 v_0,\ w_0 w_1 - v_0 \cdot v_1).$$

quaternion division:

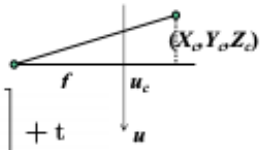
$$q_2 = q_0 / q_1 = q_0 q_1^{-1} = (v_0 \times v_1 + w_0 v_1 - w_1 v_0,\ -w_0 w_1 - v_0 \cdot v_1).$$

good for tracking of a smoothly moving camera, since there are no discontinuities in the representation. It is also easier to interpolate between rotations and to chain rigid transformations (Murray, Li, and Sastry 1994; Bregler and Malik 1998).

can use quaternions, and update estimates using an incremental rotation, as described in Section 6.2.2

https://courses.cs.washington.edu/courses/csep576/11sp/pdf/

3D → 2D Perspective Projection



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} \mathrm{R} \end{bmatrix}_{3\times 3} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathrm{t}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Homogeneous coordinates

Is this a linear transformation?

- no—division by z is nonlinear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous image coordinates — homogeneous scene coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \qquad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Perspective Projection

Projection is a matrix multiply using homogeneous coordinates:

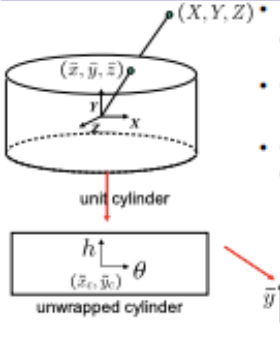
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow (-d\frac{x}{z}, \; -d\frac{y}{z})$$

divide by third coordinate

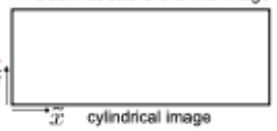
This is known as **perspective projection**

- The matrix is the **projection matrix**

Cylindrical projection



- Map 3D point (X,Y,Z) onto cylinder
  $(\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Z^2}}(X, Y, Z)$
- Convert to cylindrical coordinates
  $(sin\theta, h, cos\theta) = (\hat{x}, \hat{y}, \hat{z})$
- Convert to cylindrical image coordinates
  $(\bar{x}, \bar{y}) = (s\theta, sh) + (\bar{x}_c, \bar{y}_c)$
  – s defines size of the final image



Global alignment

- Register *all* pairwise overlapping images
- Use a 3D rotation model (one R per image)
- Use direct alignment (patch centers) or feature based
- *Infer* overlaps based on previous matches (incremental)
- Optionally *discover* which images overlap other images using feature selection (RANSAC)

Degrees of Freedom (DOF)

Similarity Transformation in 2D space:
4 DOF: 2 for translation, 1 for rotation, 1 for scale, 1 for aspect ration, 1 for shear

Affine Transformation in 2D space:
6 DOF: 2 for translation, 1 for rotation, 1 for scale

Cartesian 3D space:
6 DOF (three for position and three for orientation)

Rigid Body in D-dimensional space:
translational: d
rotational: d*(d-1)/2

Projective Transformation (homogenous coordinates):
8 DOF: 2 for translation, 1 for rotation, 1 for scale, 1 distortion of x for depth, 1 distortion of y for depth

Rotation in 3D space:

$$R_{z,\theta} = \begin{vmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{vmatrix} \quad R_{x,\theta} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{vmatrix} \quad R_{y,\theta} = \begin{vmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{vmatrix}$$

The Camera Matrix

from “Refinement in 3D Reconstruction using Cheirality Constraints” by Sengupta and Das” (https://pdfs.semanticscholar.org/b603/af27e5e72352ab75782e4b07f5f2aa669a57.pdf)

The Camera matrix contains the information of the camera; both the internal parameters, namely the zoom, focus, skew and external parameters like rotation and the translation with respect to the world coordinate frame. The camera matrix is given as P.

$$P = K[R|\mathbf{t}]$$

where, K contains the camera internal parameters, R and t are rotation and translation parameters.. The internal parameters are arranged in an upper-triangular matrix,

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$

where α_x and α_y represent the focal length in terms of pixel dimension in the x and y direction respectively, s is the camera skew and $(x0, y0)^T$ are the coordinates of the principal point.

Cheirality constraints arise from the fact that any point that lies in an image must lie in front of the camera producing that image. The depth of a point X = [X, Y, Z, T], with respect to the camera P = [MIm], is given as

$$depth(\mathbf{X}; P) = \frac{sign(detM)w}{\mathrm{T}||\mathbf{m}^3||}$$

sign(depth(X; P)) is known as the cheirality of point X with respect to camera P.

Perspective Projection

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

points are projected unto an image plane by dividing them by their z-components, sometimes scaled between near and far clipping planes.

for inhomogeneous and homogeneous coordinates, respectively:

$$\bar{\boldsymbol{x}} = \mathcal{P}_z(\boldsymbol{p}) = \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix} \quad \tilde{\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\boldsymbol{p}} \quad \tilde{\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -z_{\text{far}}/z_{\text{range}} & z_{\text{near}} z_{\text{far}}/z_{\text{range}} \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\boldsymbol{p}},$$

Note, cannot recover the distance to the 3D point after projection.
In comp graphics, projection is usually in 2-steps:
1) project 3D coordinates into *normalized device coordinates* in the range (x, y, z) ∈ [−1, −1] × [−1, 1] × [0, 1]
2) rescale these coordinates to integer pixel coordinates using a *viewport* transformation

disparity, the inverse depth, is defined using $z_{near} = 1$, $z_{far} \rightarrow \infty$, and switching the sign of the third row.

distance to a surface point can be measured using range sensors and stereo matching algorithms.

using that distance or the disparity, the 3D location can be estimated using inverse of a 4X4 matrix.

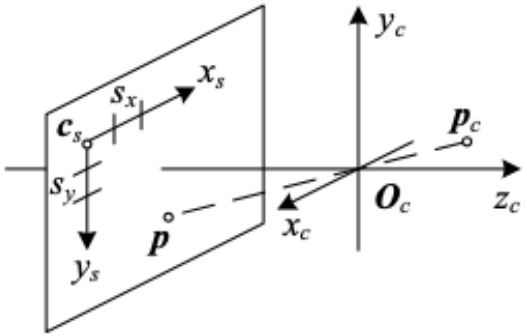


Fig 2.8 Projection of a 3D camera-centered point p_c onto the sensor planes at location p. O_c is the camera center (nodal point), c_s is the 3D origin of the sensor plane coordinate system, and s_x and s_y are the pixel spacings.

Perspective Projection, camera matrices

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

points are projected unto an image plane by dividing them by their z-components, sometimes scaled between near and far clipping planes.

for inhomogeneous and homogeneous coordinates, respectively:

$$\boldsymbol{p} = \left[\begin{array}{c|c} \boldsymbol{R}_s & \boldsymbol{c}_s \end{array} \right] \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \boldsymbol{M}_s \bar{\boldsymbol{x}}_s$$

$$\tilde{\boldsymbol{x}}_s = \alpha \boldsymbol{M}_s^{-1} \boldsymbol{p}_c = \boldsymbol{K} \boldsymbol{p}_c$$

where M_s is sensor homography. parameterized by 8 unknowns: the 3 parameters describing the rotation R_S, the 3 parameters describing the translation c_S, and the two scale factors (s_X , s_Y). often a 3X3 matrix is used
(x_s, y_s) are the pixel coordinates from the image sensor
(s_x, s_y) are the pixel spacings (e.g. in units of microns)
O_c is the camera projection center
c_s is the original in Oc
Rs is the 3D rotation

K is the camera intrinsic matrix (a.k.a. the calibration matrix), which is used to map 3D camera-centered points p_c to 2D pixel coordinates $\tilde{x}$. It has 7 degrees of freedom, but 8 in practice.

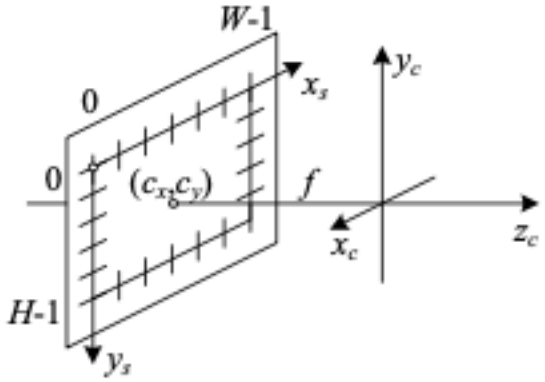
The camera’s orientation in space is called extrinsics (e.g. (R,t))

NOTE that multi-view geometry in practice treats K as an upper-left triangular matrix with 5 degrees of freedom.

Perspective Projection, camera matrices

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any of this. use it’s presented here for educational purposes only.)

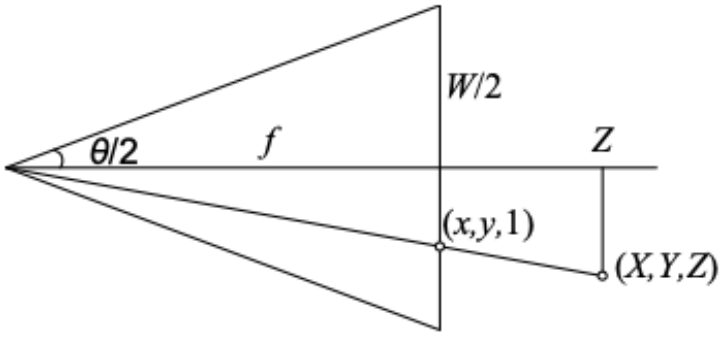
Fig 2.9 Simplified camera intrinsics showing the focal length f and the optical center (c_x,c_y). The image width and height are W and H.



$$\boldsymbol{K} = \begin{bmatrix} f & s & c_x \\ 0 & af & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

(c_x,c_y) is the *optical center* in pixel coordinates

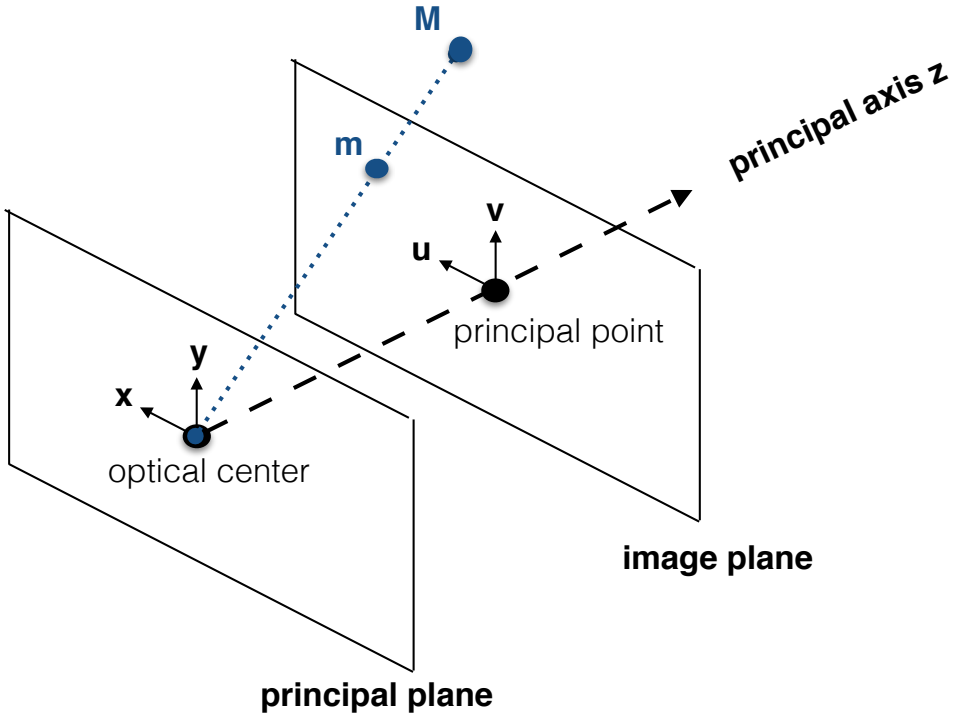
if set the origin at roughly the center of the image, e.g., (cx,cy) = (W/2,H/2), where W and H are the image height and width, the camera model has a single unknown, i.e., the focal length f. θ is the FOV. $f = (W/2)(\tan(\theta/2))^{-1}$

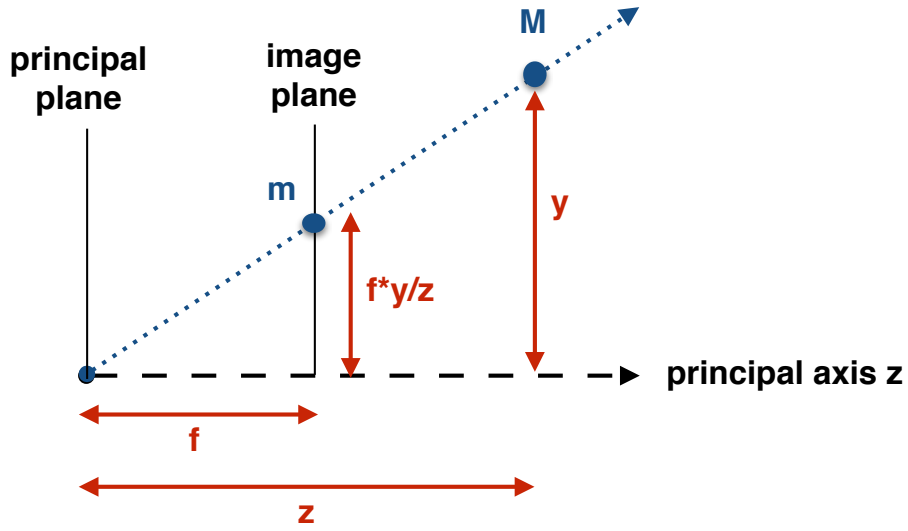


camera matrix

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}]$$

Pinhole camera geometry





camera projection matrix P is defined in z***m** = P***M**
where M = $(x,y,z,1)^T$ and m = $(f*x/z, f*y/z, 1)^T$

The right side of the projection eqn can be modified by rotation matrix **R** and translation vector **t** to put the coordinates in the (external) world coordinate system. The matrix is | **R t** |
The six parameters are called *external parameters*. | 0 1 |
rows of **R** and the *optical center* describe the *camera reference frame* in world coordinates.

The left side of the projection eqn can be modified to change coordinates in the image frame.

$$K = \begin{vmatrix} f/s_x & f/s_x*\cot\theta & o_x \\ 0 & f/s_y & o_y \\ 0 & 0 & 1 \end{vmatrix}$$

K is the *camera calibration matrix,*
result is pixel coords in image plane.

Pinhole camera geometry (cont.)

intrinsic parameters:
f is the focal distance in mm
o_x, o_y is the principal point in pixel coordinates
s_x is the width of the pixel in mm
s_y is the height of the pixel in mm
ø is the angle between the axes, usually 90 degrees
The aspect ratio s_x/s_y is usually 1
The *extrinsic parameters* depend upon Rotation and Translation of the camera.
P can be rewritten as P = **K***[I | 0]***G** = **K***[**R** | **t**]

A scale factor **λ** applied is P = **λ*****K***[**R** | **t**]
This is a 3x4 full rank matrix and can be factorized by QR factorization.

P can be rewritten as P = [$\mathbf{P_{3x3}}$ | $\mathbf{p_4}$] where $\mathbf{P_{3x3}}$ is the first 3 rows of P and $\mathbf{p_4}$ is the 4th column.

If **λ**=1, the matrix is normalized and the distance of **M** from the focal plane of the camera is *depth*.

For the image plane at infinity, the image of points doesn't depend on camera position.
The angle between 2 rays is cos θ = $\mathbf{m1^T * \omega * m2 / (}$sqrt$\mathbf{(m1^T * \omega * m1) * (}$sqrt$\mathbf{(m2^T * \omega * m2)}$

extrinsic parameters are the position and orientation of the camera w.r.t. a coord frame.

(Notes on pinhole camera followed by a few notes on multi-view geometry are from http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FUSIELLO4/tutorial.html)

projection and the principal and image planes

remembering that the center of projection is in the principal plane
one can define the position of m with respect to the 0,0 coordinates
in the image plane in which it is in.
u_c and v_c are the coordinates of the principal point.
m is at x,y in that image plane.

u = u_c + (x/pixelWidth) and v = v_c + (y/pixelHeight)

Similarly, the coordinates in the world coordinate system can be transformed to the image plane coordinate system.
Z*u = Z*u_c + (X * f)/pixelWidth
Z*v = Z*v_c + (Y * f)/pixelHeight

using scaling factor:

$$\begin{vmatrix} scale * u \\ scale * v \\ scale \end{vmatrix} = \begin{vmatrix} (f/pixelWidth) & 0 & u_c & 0 \\ 0 & (f/pixeHeight) & v_c & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} X \\ Y \\ Z \\ 1 \end{vmatrix}$$

camera calibration matrix:

$$C = \begin{vmatrix} r_1 * (f/pixelWidth) + u_c * r_3 & t_x * (f/pixelWidth) + u_c * t_z \\ r_2 * (f/pixelHeight) + v_c * r_3 & t_y * (f/pixelHeight) + v_c * t_z \\ r_3 & t_z \end{vmatrix}$$

3D transformations

from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any use of this. it’s presented here for educational purposes only.)

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8	straight lines	

Table 2.1 Hierarchy of 2D coordinate transformations. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The 2×3 matrices are extended with a third $[\mathbf{0}^T\ 1]$ row to form a full 3×3 matrix for homogeneous coordinate transformations.

Transform	Matrix	Parameters $\boldsymbol{p}$	Jacobian $\boldsymbol{J}$
translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$	(t_x, t_y)	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Euclidean	$\begin{bmatrix} c_\theta & -s_\theta & t_x \\ s_\theta & c_\theta & t_y \end{bmatrix}$	(t_x, t_y, θ)	$\begin{bmatrix} 1 & 0 & -s_\theta x - c_\theta y \\ 0 & 1 & c_\theta x - s_\theta y \end{bmatrix}$
similarity	$\begin{bmatrix} 1+a & -b & t_x \\ b & 1+a & t_y \end{bmatrix}$	(t_x, t_y, a, b)	$\begin{bmatrix} 1 & 0 & x & -y \\ 0 & 1 & y & x \end{bmatrix}$
affine	$\begin{bmatrix} 1+a_{00} & a_{01} & t_x \\ a_{10} & 1+a_{11} & t_y \end{bmatrix}$	$(t_x, t_y, a_{00}, a_{01}, a_{10}, a_{11})$	$\begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0 & x & y \end{bmatrix}$
projective	$\begin{bmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix}$	$(h_{00}, h_{01}, \ldots, h_{21})$	(see Section 6.1.3)

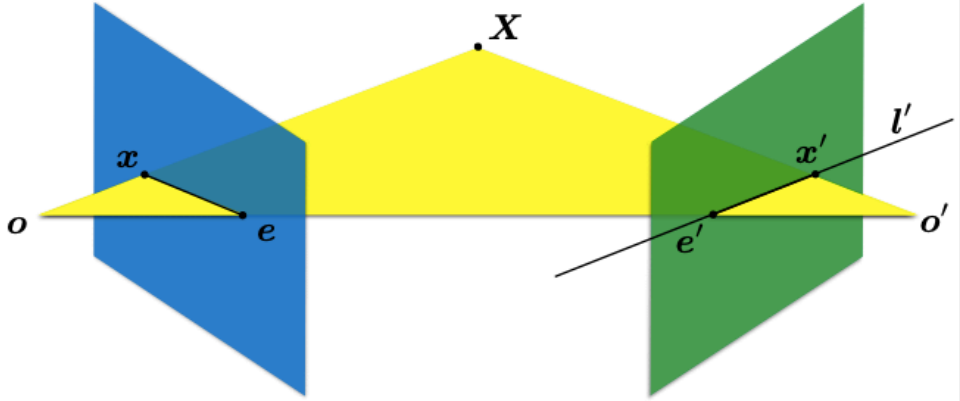
Table 6.1 Jacobians of the 2D coordinate transformations $\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p})$ shown in Table 2.1, where we have re-parameterized the motions so that they are identity for $\boldsymbol{p} = 0$.

Points and Epipolar geometry:Essential Matrix

http://www.cs.cmu.edu/~16385/s17/Slides/12.2_Essential_Matrix.pdf

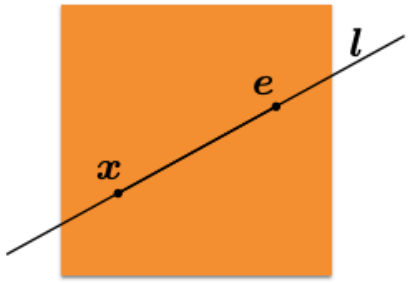
Given a point in one image,
multiplying by the **essential matrix** will tell us
the **epipolar line** in the second view.

$$\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$$



Epipolar Line

$ax + by + c = 0$ in vector form $\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$

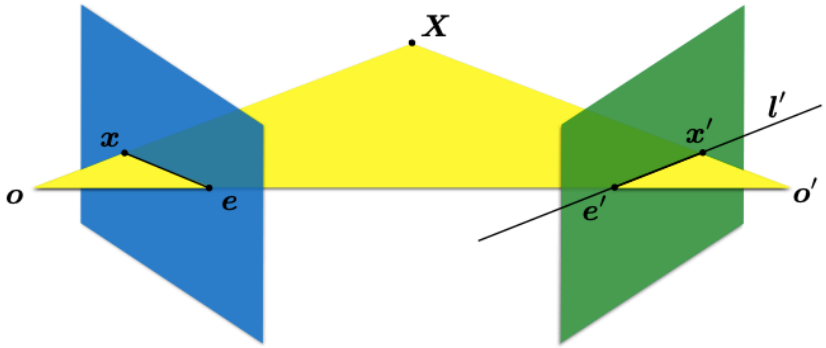


If the point $\boldsymbol{x}$ is on the epipolar line $\boldsymbol{l}$ then

$$\boldsymbol{x}^\top \boldsymbol{l} = 0$$

So if $\boldsymbol{x}^\top \boldsymbol{l} = 0$ and $\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$ then

$$\boldsymbol{x}'^\top \mathbf{E}\boldsymbol{x} = 0$$



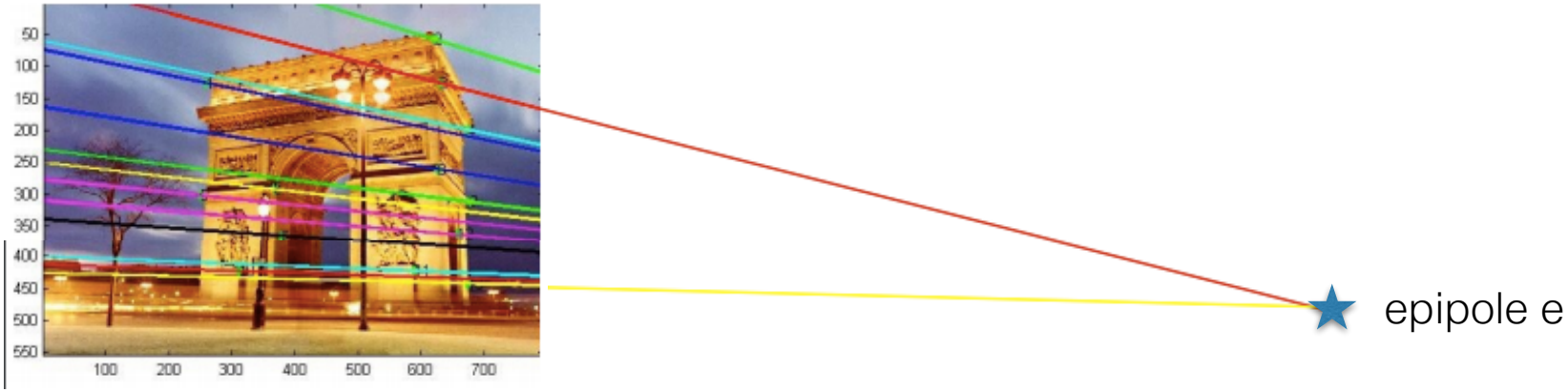
https://www.robots.ox.ac.uk/~vgg/hzbook/hzbook1/
HZepipolar.pdf
e' is the **left null-space of F**.
Similarly Fe = 0, i.e.
e is the **right null-space of F**.

coords of epipole e' w.r.t. left image coords is where the right camera is w.r.t. left image coords=
svd(F^T*F).u[2]/svd(F^T*F).u[2][2]

Points and Epipolar geometry:Essential Matrix

http://www.cs.cmu.edu/~16385/s17/Slides/12.2_Essential_Matrix.pdf

coords of right camera center in left image coords is epipole e =
svd(F^T * F).u[2]/svd(F^T * F).u[2][2]




```
>> [u,d] = eigs(F' * F)
```

eigenvectors

```
u =
    -0.0013    0.2586   -0.9660
     0.0029   -0.9660   -0.2586
     1.0000    0.0032   -0.0005
```

eigenvalue

```
d = 1.0e8*
   -1.0000         0         0
         0   -0.0000         0
         0         0   -0.0000
```

Eigenvector associated with smallest eigenvalue

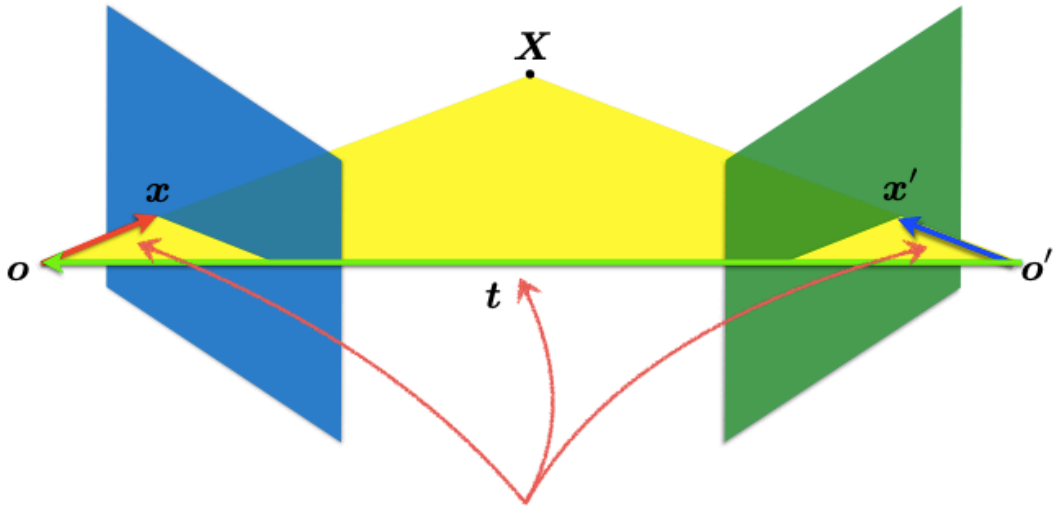
```
>> uu = u(:,3)
(  -0.9660    -0.2586    -0.0005)
```

Epipole projected to image coordinates

```
>> uu / uu(3)
(1861.02     498.21     1.0)
```

Points and Epipolar geometry:Essential Matrix

http://www.cs.cmu.edu/~16385/s17/Slides/12.2_Essential_Matrix.pdf



these three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$ so $\boldsymbol{x}^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$

also $(\boldsymbol{x} - \boldsymbol{t})^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$

rigid motion	coplanarity
$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t})$	$(\boldsymbol{x} - \boldsymbol{t})^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$

$$(\boldsymbol{x}'^\top \mathbf{R})(\boldsymbol{t} \times \boldsymbol{x}) = 0$$
$$(\boldsymbol{x}'^\top \mathbf{R})([\mathbf{t}_\times]\boldsymbol{x}) = 0$$
$$\boldsymbol{x}'^\top(\mathbf{R}[\mathbf{t}_\times])\boldsymbol{x} = 0$$

$$\boxed{\boldsymbol{x}'^\top \mathbf{E}\boldsymbol{x} = 0}$$

Essential Matrix
[Longuet-Higgins 1981]

Longuet-Higgins equation	$\boldsymbol{x}'^\top \mathbf{E}\boldsymbol{x} = 0$	
Epipolar lines	$\boldsymbol{x}^\top \boldsymbol{l} = 0$	$\boldsymbol{x}'^\top \boldsymbol{l}' = 0$
	$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x}$	$\boldsymbol{l} = \mathbf{E}^T \boldsymbol{x}'$
Epipoles	$\boldsymbol{e}'^\top \mathbf{E} = \mathbf{0}$	$\mathbf{E}\boldsymbol{e} = \mathbf{0}$

(points in normalized camera coordinates)

Points and Epipolar geometry:Fundamental Matrix

http://www.cs.cmu.edu/~16385/s17/Slides/12.3_Fundamental_Matrix.pdf

$$\hat{x}'^{\top} \mathbf{E} \hat{x} = 0$$

The Essential matrix operates on image points expressed in **normalized coordinates**
(points have been aligned (normalized) to camera coordinates)

$$\hat{x}' = \mathbf{K}^{-1} x' \qquad \hat{x} = \mathbf{K}^{-1} x$$

camera point — image point

Writing out the epipolar constraint in terms of image coordinates

$$x'^{\top} \mathbf{K}'^{-\top} \mathbf{E} \mathbf{K}^{-1} x = 0$$

$$x'^{\top} (\mathbf{K}'^{-\top} [\mathbf{t}_{\times}] \mathbf{R} \mathbf{K}^{-1}) \mathbf{x} = 0$$

$$x'^{\top} \mathbf{F} x = 0$$

properties of the F matrix

Longuet-Higgins equation $x'^{\top} \mathbf{F} x = 0$

Epipolar lines $x^{\top} l = 0$ $\quad x'^{\top} l' = 0$

$l' = \mathbf{F} x$ $\quad l = \mathbf{F}^{T} x'$

Epipoles $e'^{\top} \mathbf{F} = \mathbf{0}$ $\quad \mathbf{F} e = \mathbf{0}$

(points in **image** coordinates)

Points and Epipolar geometry:Fundamental Matrix

http://www.cs.cmu.edu/~16385/s17/Slides/12.4_8Point_Algorithm.pdf

$$\begin{bmatrix} x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x_m \\ y_m \\ 1 \end{bmatrix} = 0$$

Set up a homogeneous linear system with 9 unknowns

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_Mx'_M & x_My'_M & x_M & y_Mx'_M & y_My'_M & y_M & x'_M & y'_M & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = \mathbf{0}$$

Faugeras 1992

In the case where at least eight point correspondences have been obtained between two images of an uncalibrated stereo rig, if we arbitrarily choose five of those correspondences and consider that they are the images of five points in general positions (i.e not four of them are coplanar), then it is possible to reconstruct the other three points and any other point arising from a correspondence between the two images in the projective coordinate system defined by the five points. This reconstruction is uniquely defined up to an unknown projective transformation of the environment.

In the case where at least eight point correspondences have been obtained between two images of an uncalibrated stereo rig, if we arbitrarily choose four of these correspondences and consider that they are the images of four points in general positions (i.e not coplanar), then it is possible to reconstruct the other four points and any other point arising from a correspondence between the two images in the affine coordinate system defined by the four points. This reconstruction is uniquely defined up to an unknown affine transformation of the environment.

Torr & Murray, 1997, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix"

Consider the movement of a set of point image projections from an object which undergoes a rotation and non-zero translation between views. After the motion, the set of homogeneous image points $\{\underline{\mathbf{x}}_i\}, i = 1, \ldots n$, as viewed in the first image is transformed to the set $\{\underline{\mathbf{x}}_i'\}$ in the second image, positions related by

$$\underline{\mathbf{x}}_i'^{\top} \mathbf{F} \underline{\mathbf{x}}_i = 0 \qquad (1)$$

where $\underline{\mathbf{x}} = (x, y, \zeta)^{\top}$ is a homogeneous image coordinate and

$$\mathbf{F} = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix}$$

is the fundamental Matrix (Faugeras 1992). Although 3×3, the matrix has only seven degrees of freedom because only the ratio of parameters is significant and because $\det \mathbf{F} = 0$. Throughout, underlining a symbol $\underline{x}$ indicates the perfect or noise-free quantity, distinguishing it from $x = \underline{x} + \Delta x$, the value corrupted by noise (assumed Gaussian).

Statistics of matching correspondence: Fundamental Matrix

from “Robust Parameter Estimation in Computer Vision”, 1999 Stewart
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3914&rep=rep1&type=pdf
from “Computer Vision: Algorithms and Applications” by Szeliski
(Note, please consult author and book for any of this. use it’s presented here for educational purposes only.)

see equations (1) and (2) of Zhang, Z. 1996, RR-2927, INRIA, ffinria-00073771
”Determining the Epipolar Geometry and its Uncertainty: A Review".
And:
https://courses.cs.washington.edu/courses/csep576/11sp/pdf/SfM.pdf

Essential matrix

Co-planarity constraint:

$$x' \approx R x + t$$
$$[t]_\times x' \approx [t]_\times R x$$
$$x'^T [t]_\times x' \approx x'^T [t]_\times R x$$
$$x'^T E x = 0 \text{ with } E = [t]_\times R$$

- Solve for E using least squares (SVD)
- t is the least singular vector of E
- R obtained from the other two sing. vectors

CSE 576, Spring 2008 Structure from Motion 44

Fundamental matrix

Camera calibrations are unknown

$x' F x = 0$ with $F = [e]_\times H = K'[t]_\times R K^{-1}$

- Solve for F using least squares (SVD)
  - re-scale (x_i, x_i') so that $|x_i| \approx 1/2$ [Hartley]
- e (epipole) is *still* the least singular vector of F
- H obtained from the other two s.v.s
- “plane + parallax” (projective) reconstruction
- use self-calibration to determine K [Pollefeys]

CSE 576, Spring 2008 Structure from Motion 45

F has 7 degrees of freedom so can be estimated from at least 7 correspondence pairs.
For any point, $\tilde{x}'$, in image 2, $F\tilde{x}'$ defines a line in image 1 through which the point corresponding to $\tilde{x}'$ must pass. Similarly, for any point, $\tilde{x}$, in image 1, $F^T \tilde{x}$ defines a line in image 2 through which the point corresponding to $\tilde{x}$ must pass. All such lines, known as epipolar lines, pass through the epipoles, which are the null space of F^T and F, respectively.

Points and Epipolar geometry:Fundamental Matrix

from Strang's "Introduction to Linear Algebra",
Subspaces of matrix A which is mxn:

- row space:
  - is denoted C(A^T) and is a subspace of R^n.
  - all A^T*y.
  - dimension is nxr.
- column space:
  - is denoted C(A) and is a subspace of R^m.
  - all A*x. (the system A*x=b is solvable iff b is in column space).
  - dimension is mxr.
- Null space:
  - denoted as N(A) and is a subspace of R^n. a.k.a. the kernel of A.
  - Ax=0.
  - is orthogonal complement to ROW SPACE of A.
  - dimension is nx(n-r).
- left null space:
  - denoted as N(A^T) and is a subspace of R^m. a.k.a. as the kernel of A^T.
  - all A^T*y = 0. all column vectors x such that x^T*A = 0^T.
  - is the orthogonal complement to COLUMN SPACE of A.
  - dimension is mx(m-r).

Ax = λx, where λ is an eigenvalue of A.
λ tells whether the special vector x is stretched or shrunk or reversed or unchanged when it is multiplied by A.
nullspace entries: for eigenvalue lambda = 0 , P*x=0*x
Singular matrices have λ = 0 (and their determinants are 0).

A*f = 0 (in terms of the subspaces of matrix A, we determine the null space by the A's row space transposed times x = 0 to find what's orthogonal to the rows)

f orthogonal to A and eigenvalue=0 is smallest eigenvector of A which can be found with
SVD(A).V or SVD(A^TA).V or SVD(A).U or SVD(AA^T).U then dimension reduction to rank=2 for F

Points and Epipolar geometry:Fundamental Matrix and ***smallest eigenvalue***

from Wirtz and Guhr 2014,
"Distribution of the Smallest Eigenvalue in Complex and Real Correlated Wishart Ensembles"
https://arxiv.org/pdf/1310.2467.pdf

In linear discriminant analysis it ***[smallest eigenvalue]*** gives the leading contribution for the threshold estimate [21]. It is most sensitive to noise in the data [18]. In linear principal component analysis, ***the smallest eigenvalue determines the plane of closest fit [18]***. It is also crucial for the identification of single statistical outliers [17]. In numerical studies involving large random matrices, the condition number is used, which depends on the smallest eigenvalue [22, 23]. In wireless communication the Multi–Input–Multi– Output (MIMO) channel matrix of an antenna system is modeled by a random matrix [24]. The smallest eigenvalue of C yields an estimate for the error of a received signal [25, 26, 27]. In finance, the optimal portfolio is associated with the eigenvector to the smallest eigenvalue of the covariance matrix, which is directly related to the correlation matrix [28]. This incomplete list of examples shows the influence of the smallest eigenvalue in applications. ***Further information on the role of the smallest eigenvalue is given in Appendix A***.

[17] Barnett V, Lewis T. Outliers in Statistical Data. first edition ed. John Wiley & Sons; 1980.
[18] Gnanadesikan R. Methods for Statistical Data Analysis of Multivariate Oberservations. Second edition ed. John Wiley & Sons; 1997.

Points and Epipolar geometry:Fundamental Matrix: evaluating best fit

Torr & Murray, 1997, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix"

https://www.robots.ox.ac.uk/ActiveVision/Publications/torr_murray_ijcv1997/torr_murray_ijcv1997.pdf

2.2. *Orthogonal Least Squares Regression: Method OR*

Ignoring for the moment the problem of enforcing the rank 2 constraint, given $n \geq 8$ correspondences this system appears an archetype for solution by linear least squares regression. Linear here refers to linearity in the parameters f_i — equation (1) written out is just

$$f_1 \underline{x}'_i \underline{x}_i + f_2 \underline{x}'_i \underline{y}_i + f_3 \underline{x}'_i \zeta + f_4 \underline{y}'_i \underline{x}_i + f_5 \underline{y}'_i \underline{y}_i + f_6 \underline{y}'_i \zeta + f_7 \underline{x}_i \zeta + f_8 \underline{y}_i \zeta + f_9 \zeta^2 = 0 \ .$$

Because errors exist in all the measured coordinates x, y, x', y', orthogonal least squares (Pearson 1901) rather than ordinary least squares should be used, minimizing the sum of the squares of the distances shown in part (a) rather than (b) of Figure 2.

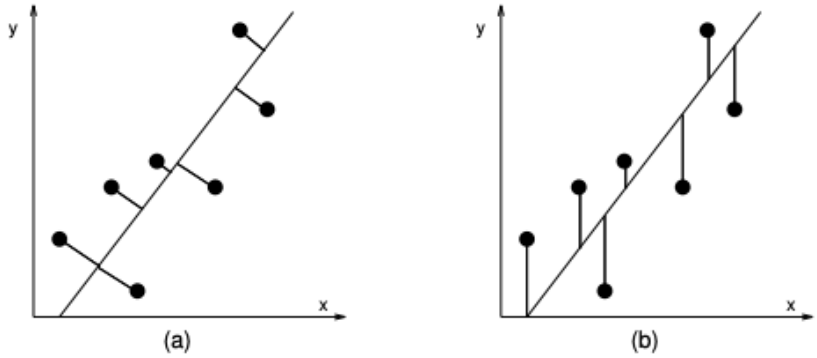


Fig. 2. The (a) orthogonal and (b) ordinary least squares distances.

Consider fitting a hyperplane $\mathbf{f} = (f_1, f_2, \ldots, f_p)$ through a set of n points in $\mathcal{R}^p$ with coordinates $\mathbf{z}_i = (z_{i_1}, z_{i_2}, \ldots, z_{i_p})$, taking the centroid of the data as origin. (Centring is a standard statistical technique that involves shifting the coordinate system of the data points so that the centroid lies at the origin. The best fitting hyperplane passes through the centroid of the data (Pearson 1901).) Assuming that the noise is Gaussian and that the elements of $\mathbf{z}$ have equal variance[1], the hyperplane $\mathbf{f}$ with maximum likelihood is estimated by minimizing the perpendicular sum of Euclidean distances from the points to the plane (Pearson 1901; Kendall & Stuart 1983)

$$\min_{\mathbf{f}} \sum_{i=1}^{n} (\mathbf{f}^\top \mathbf{z}_i)^2$$

. . .

Method S1.

The orthogonal least squares method (OR) will in fact produce a sub-optimal estimate of $\mathbf{F}$ because the residuals in the minimization

$$r_i = f_1 x'_i x_i + f_2 x'_i y_i + f_3 x'_i \zeta + f_4 y'_i x_i + f_5 y'_i y_i + f_6 y'_i \zeta + f_7 x_i \zeta + f_8 y_i \zeta + f_9 \zeta^2 \qquad (2)$$

are not Gaussianly distributed. The cause of this is

Points and Epipolar geometry:Fundamental Matrix: evaluating best fit

Torr & Murray, 1997, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix"

https://www.robots.ox.ac.uk/ActiveVision/Publications/torr_murray_ijcv1997/torr_murray_ijcv1997.pdf

Method S2. Luong and Faugeras (1993)

the epipolar line $\mathbf{F}\underline{\mathbf{x}}$ in image two. Noisy measurements will however not lie on their associated epipolar lines exactly. The perpendicular distance of a point $\mathbf{x}$ to the predicted epipolar line $\mathbf{x}'^{\top}\mathbf{F}$ in the first image is

$$e_1 = \frac{xr_x + yr_y + r_\zeta}{(r_x^2 + r_y^2)^{1/2}} = \frac{r}{(r_x^2 + r_y^2)^{1/2}}.$$

and the distance of a point $\mathbf{x}'$ to the epipolar line $\mathbf{F}\mathbf{x}$ in the second image is

$$e_2 = \frac{r}{(r_{x'}^2 + r_{y'}^2)^{1/2}}.$$

. . .

its epipolar line is used. This ensures that each image receives equal consideration. The distance $e = (e_1^2 + e_2^2)^{1/2}$ is referred to as the epipolar distance. It is assumed that the errors in the measured location of each point are Gaussian with variance σ^2. If the co-

w_E, so that $e = rw_E$. Effectively then the epipolar weighting is

$$w_E = \left(\frac{1}{r_x^2 + r_y^2} + \frac{1}{r_{x'}^2 + r_{y'}^2}\right)^{1/2},$$

which is not dissimilar to the Sampson weighting

$$w_S = \left(\frac{1}{r_x^2 + r_y^2 + r_{x'}^2 + r_{y'}^2}\right)^{1/2}.$$

In summary, three least squares methods with different error terms have been discussed:

1. Method OR uses the sum of squares of *algebraic distances*:
$$R^2 = \sum r_i^2, \text{ where } r_i = \mathbf{x}_i'^{\top}\mathbf{F}\mathbf{x}_i.$$
2. Method S1 uses the sum of squares of the Sampson distances:
$$D^2 = \sum (w_{Si} r_i)^2 = \sum (r_i / \nabla r_i)^2 = \sum d_i^2.$$
3. Method S2 uses the sum of squares of the epipolar distances:
$$E^2 = \sum (w_{Ei} r_i)^2 = \sum (e_{1i}^2 + e_{2i}^2) = \sum e_i^2.$$

In terms of probability distributions, the first corresponds to something intractable, the second is a first order approximation to a χ^2 distribution, and the third is a χ^2 distribution.

Statistics of matching correspondence

from “Robust Parameter Estimation in Computer Vision”, 1999 Stewart
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3914&rep=rep1&type=pdf

Robust Estimation

breakdown point: the minimum fraction of outlying data that can cause an estimate to diverge arbitrarily far from the true estimate. (e.g. for least squares, 1 outlier can harm the estimate, so the breakdown point is 0). a breakdown point of 0.5 is traditionally used.

influence function*:* the change in an estimate caused by insertion of outlying data as a function of the distance of the data from the (uncorrupted) estimate. (e.g. for least squares, the influence function is simply proportional to the distance of the point from the estimate.) influence function should approach 0 with increasing distance.

statistical efficiency: ratio of minimum possible variance to the actual variance of an estimate. the minimum possible variance is determined by a distribution such as a gaussian. the upper bound of efficiency is then 1.

Linear Regression or estimation of univariate or multivariate location and scatter are used.

Let **X** = {**x_i**} be a set of data points (vectors)

Let **a** be a k-dimensional parameter vector to be estimated.

r_i_a = r(**x_i, a**) is the objective function is defined in terms of an error distance or residual function, a true geometric distance is ideal. euclidean, mahalanobis of the covariance matrix, etc.

(Mahalanobis distance is a measure of the distance between a point and a distribution in terms of standard deviations.)

M-estimators and least-median of squares

Statistics of matching correspondence

from “Robust Parameter Estimation in Computer Vision”, 1999 Stewart
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3914&rep=rep1&type=pdf

M-estimators: generalizations of maximum likelihood and least squares.

$\boldsymbol{\sigma}_i^2$ = variance (scale) assoc. w/ scalar r_i_a

ρ(u) = a robust loss function

â is the M-estimator of **a**.

$$\hat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in X} \rho(r_{i,\mathbf{a}}/\sigma_i)$$

ψ(u) = ***ρ’(u)*** is essentially proportional to the influence function.

w(u) · u = ψ(u) is a weight function.

iteratively re-weighted least-squares (IRLS) alternates between calculating weights $w_i = w(r\_i\_a/\boldsymbol{\sigma}_i)$ using the current estimate of **a** then solving following eqn. to estimate **a** with fixed weights**.**

$$\sum_{\mathbf{x}_i \in X} \psi(r_{i,\mathbf{a}}/\sigma_i) \frac{dr_{i,\mathbf{a}}}{d\mathbf{a}} \frac{1}{\sigma_i} = 0.$$

Beaton and Tukey [4]	$\rho(u) = \begin{cases} \frac{a^2}{6}\left[1-\left(1-\left(\frac{u}{a}\right)^2\right)^3\right] & \lvert u\rvert \le a \\ \frac{a^2}{6} & \lvert u\rvert > a \end{cases}$	$\psi(u) = \begin{cases} u\left[1-\left(\frac{u}{a}\right)^2\right]^2 & \lvert u\rvert \le a \\ 0 & \lvert u\rvert > a \end{cases}$
Cauchy [32]	$\rho(u) = \frac{b^2}{2}\log\left[1+\left(\frac{u}{b}\right)^2\right]$	$\psi(u) = \frac{u}{1+(u/b)^2}$
Huber [34, Chapter 7]	$\rho(u) = \begin{cases} \frac{1}{2}u^2 & \lvert u\rvert \le c \\ \frac{1}{2}c(2\lvert u\rvert - c) & c < \lvert u\rvert \end{cases}$	$\psi(u) = \begin{cases} 1 & \lvert u\rvert \le c \\ c\frac{u}{\lvert u\rvert} & \lvert u\rvert > c \end{cases}$

Table 1: Three different robust loss functions, $\rho(u)$, and associated ψ functions. The “tuning parameters” a, b and c are often tuned to obtain 95% efficiency [32].

Statistics of matching correspondence

from “Robust Parameter Estimation in Computer Vision”, 1999 Stewart
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3914&rep=rep1&type=pdf

Least-Median of Squares: has a breakdown point of 0.5.
â is the LMS estimate of **a**. $\hat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmin}} \underset{\mathbf{x}_i \in X}{\operatorname{median}} r_{i,\mathbf{a}}^2.$

because the median is not differentiable, search techniques are needed.
e.g. for regression lines, can computationally solve for median exactly and efficiently.

a random sampling technique: randomly select a number of *k*-point subsets of the *N* data points. A parameter vector, $\mathbf{a_s}$, is fit to the points in each subset, *s*. Each $\mathbf{a_s}$ is tested as a hypothesized fit by calculating the squared residual distance $(r_{i\mathbf{a_s}})^2$ of each of the *N-k* points in X-s and finding the median. The $\mathbf{a_s}$ corresponding to the smallest median over S subsets is chosen as the estimate, **â**. Overall, this computation requires O(S N) time using linear-time median finding techniques.

Determining the number of subsets **S** is important. S needs to be large enough to have high probability of containing al “good” points, that is, no outliers.
p is the minimum fraction of good points.
p^k is the probability that a set contains all good points.

$$P_g = 1 - (1 - p^k)^S.$$

choose P_g and solve for S:

P_g	k	p	S
0.99	3	0.5	35
0.99	6	0.6	97
0.99	6	0.5	293

Statistics of matching correspondence

from “Robust Parameter Estimation in Computer Vision”, 1999 Stewart
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3914&rep=rep1&type=pdf

Least-Median of Squares (cont.):

â can be used to refine the fit.

s* is the subset used to form **â**.

Φ^{-1} is the inverse of the cumulative normal distribution.

$$\hat{\sigma} = 1/\Phi^{-1}(0.75)\left(1 + \frac{5}{N-k}\right)\sqrt{\underset{\mathbf{x}_i \in X - s^*}{\text{median}}\, r^2_{i,\hat{\mathbf{a}}}},$$

if all N points are sampled from a normal distribution with variance σ^2 , then $\hat{\sigma} \to \sigma$ as $N \to \infty$.
can choose data points such that $(r_{ias})^2 < (\theta\, \hat{\sigma})^2$ where θ is constant, typically about 2.5 then re-estimate **â**.

Random Sample Consensus (RANSAC):

a minimal subset random sampling search technique, the objective function to be maximized is the number of data points (inliers) having absolute residuals smaller than a predefined value. Equivalently, this may be viewed as minimizing the number of outliers, which may then be viewed as a binary robust loss function that is 0 for small (absolute) residuals, 1 for large absolute residuals, and has a discontinuous transition at θ.

Torr & Murray, 1997, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix"

https://www.robots.ox.ac.uk/ActiveVision/Publications/torr_murray_ijcv1997/torr_murray_ijcv1997.pdf

Using RANSAC w/ 8-point or 7-point methods:

RANSAC uses iterations over random subsets of size 7 or 8 of the total point set (which is usually over-determined, having more than 8 points), solves for the fundamental matrix (F), evaluates the fit using epipolar projections of all points, and keeps the best F.

The number of iterations is pre-calculated for the probability that all points are not outliers. Then upon each improved F in the iterations, the number of iterations is readjusted downwards.

Table 3. The number m of subsamples required to ensure $\Upsilon \geq 0.95$ for given p and ϵ, where Υ is the probability that all the data points selected in one subsample are non-outliers.

Features	Fraction of Contaminated Data, ϵ						
p	5%	10%	20%	25%	30%	40%	50%
2	2	2	3	4	5	7	11
3	2	3	5	6	8	13	23
4	2	3	6	8	11	22	47
5	3	4	8	12	17	38	95
6	3	4	10	16	24	63	191
7	3	5	13	21	35	106	382
8	3	6	17	29	51	177	766

Fundamental Matrix: 7 point algorithm

(see src/…/EpipolarTransformer.java)
(see https://www.robots.ox.ac.uk/~vgg/hzbook/code/vgg_multiview/vgg_F_from_7pts_2img.m from "Multiple View Geometry in Computer Vision
Second Edition, by Hartley and Zisserman)

7-Point algorithm (for use with uncalibrated cameras)**:**

```
(1)Transform the coordinates
(2)build matrix A with the normalized x,y points
(3) compute linear least square solution to the least eigenvector of f.
     solve A = U * D * V^T   for A*f = [..x...]*f = 0
     A has rank 7.  f has rank 2.  calculate [U,D,V] from svd(A)
(4) make the fundamental matrix have a rank of 2 by performing a svd and then reconstructing with the two largest
     singular values. [U,D,V] = svd(F,0);
     The homogeneous system AX = 0 : X is the null space of matrix A.
     The system is nullable because rank 7 < number of columns, 9.
     The nullable system must have a solution other than trivial where|A| = 0.
     There should be 9-7=2 linearly independent vectors u1, u2, ... , un-r that span the null space of A.
     The right null space of A reduced by SVD is then 2D and the last
        2 columns of V can be extracted and reshaped to [3x3] as F1 and F2.
     A linear convex combination of F1 and F2 form the estimate of F.
     F = α*F1 + (1 - α)*F2  where α is between 0 and 1
     The eigenvalues of F are possible only if the determinant of F is 0.
           det A = 0 ==> det(α*F1 + (1 - α)*F2) = 0
           because det(F1 + F2) != det(F1) + det(F2), have to step through the
             determinant of the sums, and group the terms by a^3, a^2, a^1, and a^0
             and then solve for the cubic roots as the values of 'a'.
     The determinant of F is a polynomial function, the characteristic
        polynomial whose degree is the order of the matrix, which is 3 in this
        case. Therefore, the answer(s) to determinant(F) = 0 requires the cubic
        roots of the equation.
     After the cubic root(s) are solved, they are back substituted into :
        Fi = a(i) * FF{1} + (1-a(i)) * FF{2};
        to get the solutions Fi which may be one or 3 solutions
(5) denormalize the fundamental matrix
     The related part of the normalization equation: inv(T_2) * F * inv(T_1)
     so denormalizing is:
         F = (T_1)^T * F * T_2
(6) validate the 1 to 3 solutions:
(7) estimate the error in the fundamental matrix by calculating epipolar
     lines for points in image 1 and find their nearest points in image 2
     and measure the perpendicular distance from the epipolar line for
     those nearest points.
```

Torr, Zisserman, & Maybank 1996, "Robust Detection of Degenerate Configurations whilst Estimating the Fundamental Matrix

https://www.robots.ox.ac.uk/~phst/Papers/CVIU97/m.ps.gz

multi-view geometry

With point correspondences $\mathbf{m_i} \longleftrightarrow \mathbf{M_i}$, , can solve for camera projection matrix P. Using the Kroenecker delta product and vec operator, coefficients are rewritten to their linearly independent forms of a matrix of size $2n \times 12$ called the coefficient matrix A

From a set of n point correspondences, we obtain a $2n \times 12$ coefficient matrix A, where n must be > 6.

In general A will have rank 11 (provided that the points are not all coplanar) and the solution is the *1-dimensional right null-space of A*.

The linear system of equations for inexact data is solved using least squares.
The least-squares solution for vec P^T is the singular vector corresponding to the smallest *singular value* decomposition of A and the direct linear transform.
The equation is usually written in form A * h = 0 where h is vec P^T (its the one dimensional reshaping of the homograph matrix).

multi-view geometry (cont.)

In general:
— dot product of a point and a line is zero if the point lies on the line
— if the *intrinsic parameters* are known, the relationship between corresponding points is given by the *essential matrix*: $\mathbf{m_r^T * E * m_l^T}$
— when no camera details are available, one can use the *fundamental matrix*.
It's a 3x3 rank 2 homogeneous matrix. It has 7 degrees of freedom.
For any point $\mathbf{m_l}$ in the left image, the corresponding epipolar line $\mathbf{l_r}$ in the right image can be expressed as $\mathbf{l_r = F * m_l}$ and vice versa $\mathbf{l_r = F^T * m_r}$
— the *essential and fundamental matrices* are related through $\mathbf{F} = \mathbf{K_r^{-T}} * \mathbf{E} * \mathbf{K_l^{-T}}$ where Kr and Kl are the left and right camera matrices

Regarding 3D reconstruction:
— when both *intrinsic* and *extrinsic* camera parameters are known, the reconstruction is solved unambiguously by triangulation.
— when only *intrinsic* parameters are known, extrinsic parameters can be estimated and the reconstruction can be solved up to an unknown scale factor, that is **R** can be estimated, but **t** can be only up to a scale factor. the epipolar geometry is the essential matrix and the solvable projection is euclidean (rigid+ uniform scale)
— when neither *intrinsic* nor *extrinsic* parameters are known, i.e., the only information available are pixel correspondences, the reconstruction can be solved up to an unknown, global projective transformation of the world.

In Trifocal geometry, a trifocal tensor is used.

Statistics of matching correspondence: Fundamental Matrix

from “IVPV Handbook of Optics”, Vol II
edited by Michael Bass
http://photonics.intec.ugent.be/education/IVPV/res_handbook/v2ch17.pdf

from “Multiple View Geometry in Computer Vision”
Second Edition by Hartley & Zisserman
https://www.robots.ox.ac.uk/~vgg/hzbook/

(Note, consult authors and books for any use.)

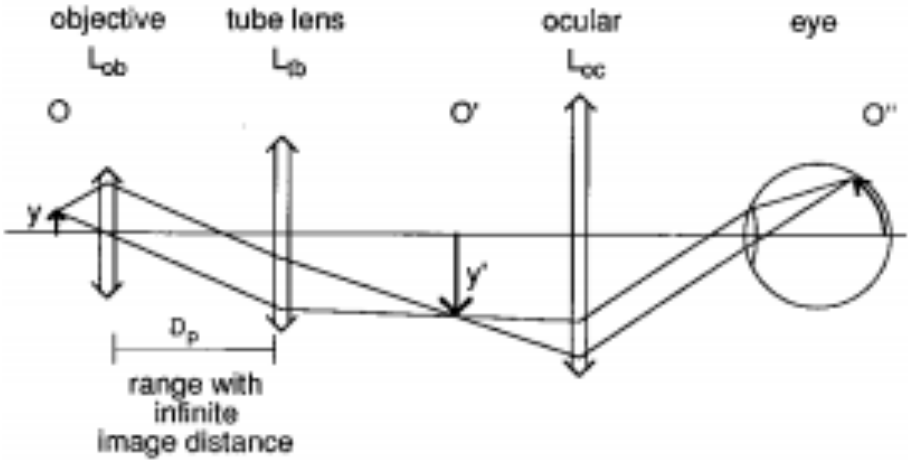


FIGURE 4 Ray path in microscope with infinity-corrected objective and tube lens.

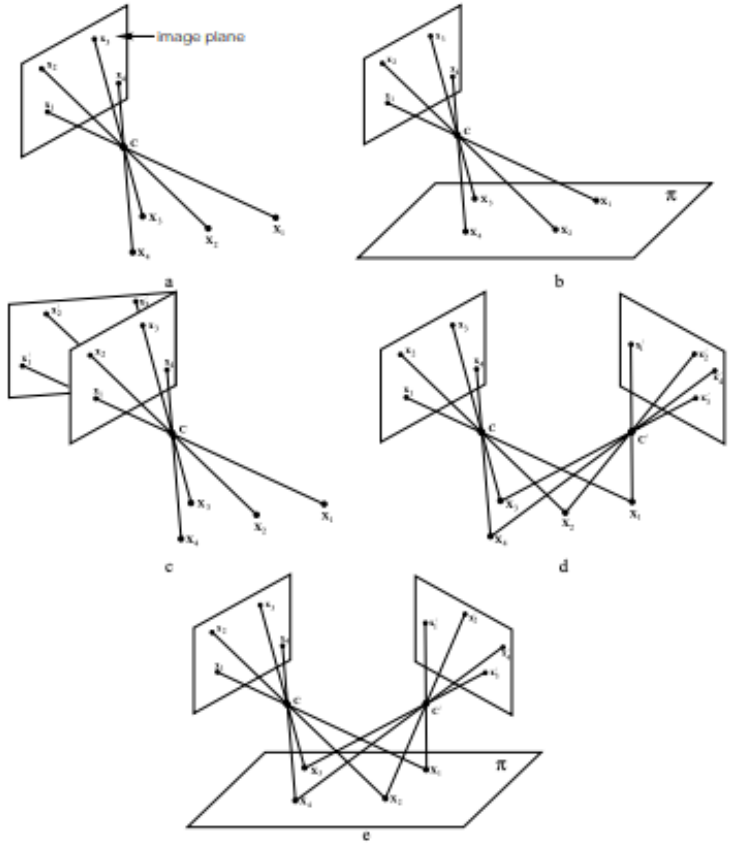


Fig. 1.1. **The camera centre is the essence.** *(a) Image formation: the image points* $\mathbf{x}_i$ *are the intersection of a plane with rays from the space points* $\mathbf{X}_i$ *through the camera centre* $\mathbf{C}$. *(b) If the space points are coplanar then there is a projective transformation between the world and image planes,* $\mathbf{x}_i = \mathrm{H}_{3\times3}\mathbf{X}_i$. *(c) All images with the same camera centre are related by a projective transformation,* $\mathbf{x}'_i = \mathrm{H}'_{3\times3}\mathbf{x}_i$. *Compare (b) and (c) – in both cases planes are mapped to one another by rays through a centre. In (b) the mapping is between a scene and image plane, in (c) between two image planes. (d) If the camera centre moves, then the images are in general not related by a projective transformation, unless (e) all the space points are coplanar.*

Statistics of matching correspondence: Fundamental Matrix

from “Multiple View Geometry in Computer Vision”
Second Edition by Hartley & Zisserman
https://www.robots.ox.ac.uk/~vgg/hzbook/
(Note, consult author and book for any of this. use it’s presented here for educational purposes only.)

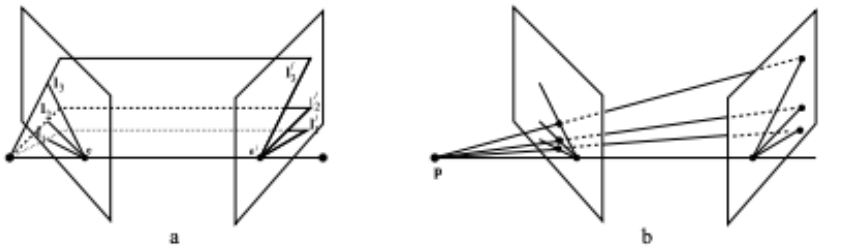


Fig. 9.6. **Epipolar line homography.** *(a) There is a pencil of epipolar lines in each image centred on the epipole. The correspondence between epipolar lines, $\mathbf{l}_i \leftrightarrow \mathbf{l}'_i$, is defined by the pencil of planes with axis the baseline. (b) The corresponding lines are related by a perspectivity with centre any point* $\mathbf{p}$ *on the baseline. It follows that the correspondence between epipolar lines in the pencils is a 1D homography.*

- F is a rank 2 homogeneous matrix with 7 degrees of freedom.
- **Point correspondence**: If $\mathbf{x}$ and $\mathbf{x}'$ are corresponding image points, then $\mathbf{x}'^{\mathsf{T}}\mathbf{F}\mathbf{x} = 0$.
- **Epipolar lines**:
  - $\mathbf{l}' = \mathbf{F}\mathbf{x}$ is the epipolar line corresponding to $\mathbf{x}$.
  - $\mathbf{l} = \mathbf{F}^{\mathsf{T}}\mathbf{x}'$ is the epipolar line corresponding to $\mathbf{x}'$.
- **Epipoles**:
  - $\mathbf{F}\mathbf{e} = \mathbf{0}$.
  - $\mathbf{F}^{\mathsf{T}}\mathbf{e}' = \mathbf{0}$.
- **Computation from camera matrices** $\mathtt{P}, \mathtt{P}'$:
  - General cameras, $\mathtt{F} = [\mathbf{e}']_\times \mathtt{P}'\mathtt{P}^+$, where $\mathtt{P}^+$ is the pseudo-inverse of $\mathtt{P}$, and $\mathbf{e}' = \mathtt{P}'\mathbf{C}$, with $\mathtt{P}\mathbf{C} = \mathbf{0}$.
  - Canonical cameras, $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{M} \mid \mathbf{m}]$, $\mathtt{F} = [\mathbf{e}']_\times \mathtt{M} = \mathtt{M}^{-\mathsf{T}}[\mathbf{e}]_\times$, where $\mathbf{e}' = \mathbf{m}$ and $\mathbf{e} = \mathtt{M}^{-1}\mathbf{m}$.
  - Cameras not at infinity $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$, $\mathtt{F} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1} = [\mathtt{K}'\mathbf{t}]_\times \mathtt{K}'\mathtt{R}\mathtt{K}^{-1} = \mathtt{K}'^{-\mathsf{T}}\mathtt{R}\mathtt{K}^{\mathsf{T}}[\mathtt{K}\mathtt{R}^{\mathsf{T}}\mathbf{t}]_\times$.

Table 9.1. *Summary of fundamental matrix properties.*

Bundle Adjustment

from “Bundle Adjustment — A Modern Synthesis”
by Triggs, McLauchlan, Hartley, and Fitzgibbon
Vision Algorithms’99, LNCS 1883, pp. 298–372, 2000
http://www.cs.jhu.edu/~misha/ReadingSeminar/Papers/Triggs00.pdf
(Note, consult author for any use of figures and text - it’s presented here for educational purposes only.)

(not implemented in this project currently)

Bundle adjustment: refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates, by minimizing a cost function that quantifies the model fitting error, and jointly that the solution is simultaneously optimal with respect to both structure and camera variations… a large sparse geometric parameter estimation problem, the parameters being the combined 3D feature coordinates, camera poses and calibrations.

cost function: often use non-quadratic M-estimator-like distributional models to handle outliers more integrally, and many include additional penalties related to overfitting, model selection and system performance (priors, MDL).
modelled as a sum of opaque contributions from the independent information sources (individual observations, prior distributions, overfitting penalties ...). The functional forms of these contributions and their dependence on fixed quantities such as observations will usually be left implicit

main conclusion will be that robust statistically-based error metrics based on total (inlier + outlier) log likelihoods should be used, to correctly allow for the presence of outliers… A typical ML cost function would be the *summed negative log likelihoods* of the prediction errors of all the observed image features. For Gaussian error distributions, this reduces to the *sum of squared covariance-weighted prediction errors* (§3.2). A MAP estimator (is Bayesian) would typically add cost terms giving certain structure or camera calibration parameters a bias towards their expected values.

first realize that geometric fitting is really a special case of parametric probability density estimation.
Maximizing the likelihood corresponds to fitting this predicted observation density to the observed data. The geometry and camera model only enter indirectly, via their influence on the predicted distributions.
ML estimation is naturally robust to outliers.

Bundle Adjustment

from “Bundle Adjustment — A Modern Synthesis”
by Triggs, McLauchlan, Hartley, and Fitzgibbon
Vision Algorithms’99, LNCS 1883, pp. 298–372, 2000
http://www.cs.jhu.edu/~misha/ReadingSeminar/Papers/Triggs00.pdf
(Note, consult author for any use of figures and text - it’s presented here for educational purposes only.)

Pattern Matrices and Factorization:
The most common ***top-down*** method is called nested dissection or recursive partitioning: recursively split the factorization problem into smaller sub-problems, solve these independently, and then glue the solutions together along their common boundaries. Splitting involves choosing a separating set of variables, whose deletion will separate the remaining variables into two or more independent subsets. This corresponds to finding a (vertex) graph cut of the elimination graph, i.e. a set of vertices whose deletion will split it into two or more disconnected components.
Given such a partitioning, the variables are reordered into ***connected components***, with the separating set ones last. This produces a pattern matrix such as ‘block tridiagonal matrix’, ’arrowhead matrix’, or ‘bundle hessian’, etc.
**Finding a globally minimal partition sequence is NP complete but several effective heuristics exist.

Many ***bottom-up*** variable ordering heuristics exist.

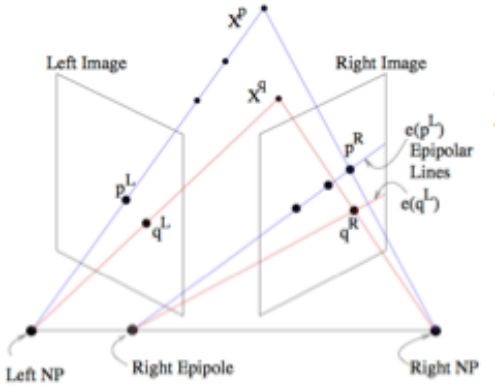
Deciding the reconstructed coordinate system is in the realm of **Gauge Freedom (**a gauge is a reference frame**)**:
constraining the coordinate values of certain aspects of the reconstructed structure — features, cameras or combinations of these — whatever the rest of the structure might be.

Bundle Adjustment

from “Bundle Adjustment in the Large”, 2010
by Agarwal, Snavely, Seitz, and Szeliski
https://homes.cs.washington.edu/~sagarwal/bal.pdf
(Note, consult author for any use of figures and text - it’s presented here for educational purposes only.)

treated as re-weighted non-linear least squares problem .

Projection, stereo and panoramic images



http://www.cs.toronto.edu/~jepson/csc420/notes/epiPolarGeom.pdf
(note, the image is posted on an educational site and copied here without following up on permissions. Any further use of the image should follow up on the origins and permissions.)

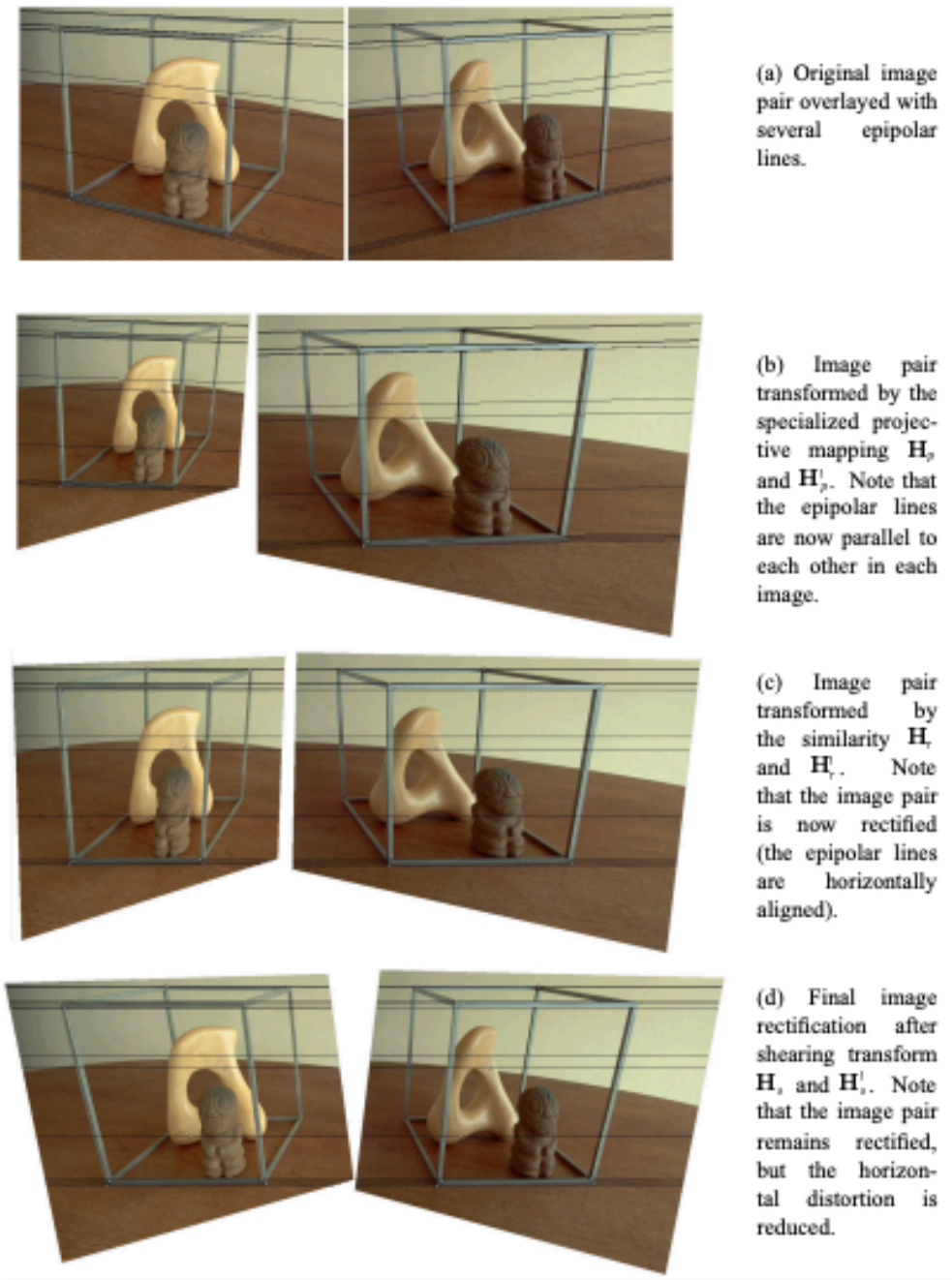
panoramic images here are from Brown & Lowe 2003

rectification and registration

from “Computing Rectifying Homographies for **Stereo Vision**”
by Loop & Zhang, 1999
https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr99-21.pdf

the transforms Hp and H0p were found that map the epipoles e and e0 to points at 1. In this section we define a pair of similarity transforms Hr and H0r that rotate these points at inf. into alignment with the direction $i = [1\ 0\ 0]^T$ as required for rectification. Additionally, a translation in the v-direction on one of the images is found to exactly align the scan-lines in both images.

… reduce the distortion with a shearing transform.



(a) Original image pair overlayed with several epipolar lines.

(b) Image pair transformed by the specialized projective mapping $\mathbf{H}_p$ and $\mathbf{H}'_p$. Note that the epipolar lines are now parallel to each other in each image.

(c) Image pair transformed by the similarity $\mathbf{H}_r$ and $\mathbf{H}'_r$. Note that the image pair is now rectified (the epipolar lines are horizontally aligned).

(d) Final image rectification after shearing transform $\mathbf{H}_s$ and $\mathbf{H}'_s$. Note that the image pair remains rectified, but the horizontal distortion is reduced.

Figure 3: An example showing various stages of the proposed rectification algorithm.

Projection, stereo and panoramic images

A review on bundle adjustment and 3D rectification and tracking in Section 1 and 2 of this paper:
(not implemented in this project)

from “Flight Dynamics-based Recovery of a UAV Trajectory using Ground Cameras”
by Rozantsev et al. 2017
http://openaccess.thecvf.com/content_cvpr_2017/papers/Rozantsev_Flight_Dynamics-Based_Recovery_CVPR_2017_paper.pdf

We propose a new method to estimate the 6-dof trajectory of a flying object such as a quadrotor UAV within a 3D airspace monitored using multiple fixed ground cameras…Our method requires neither perfect single-view tracking nor appearance matching across views. For robustness, we allow the tracker to generate multiple detections per frame in each video. The true detections and the data association across videos is estimated using robust multi-view triangulation and subsequently refined during our bundle adjustment procedure.

…

existing multi-camera tracking methods are designed to track people, vehicles to address indoor and outdoor surveillance tasks, where the targets are often on the ground. In contrast, small drones must be tracked within a 3D volume that is orders of magnitude larger

…

Most existing multi-camera systems also rely on accurate camera calibration that requires someone to collect calibration data by walking around in the scene.

…

3D from structured light

active sensing method called phase shifting method (PSM)
(not implemented in this project)

Projection, stereo and panoramic images

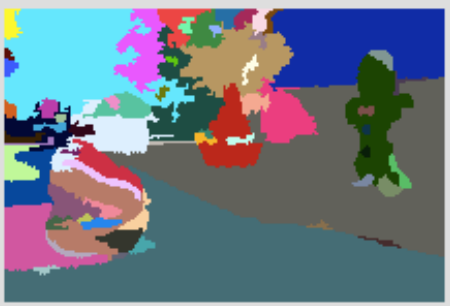
single point perspective and two and three point perspectives sometimes can have "foreshortened" object dimensions if the axes of the object are not parallel to the camera plane.

For example, when creating correspondence for the gingerbread man in the image below using partial shape matching and euclidean transformation for evaluation, half of the object is unmatched within a tolerance.

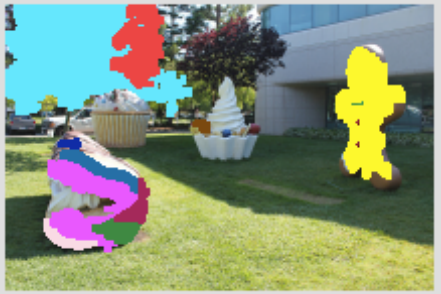
So, a rough calculation of the angle of inclination is needed to find the other matching points consistent with a single-point perspective projection.

The "foreshortening" along the x-axis is corrected using cosine of the angle, that is the true dimension is the measured divided by cosine of angle.
The foreshortening along the y axis can be approximated by the ratio of the far sides of two triangles, one embedded in the other.

image for the search for gingerbread man

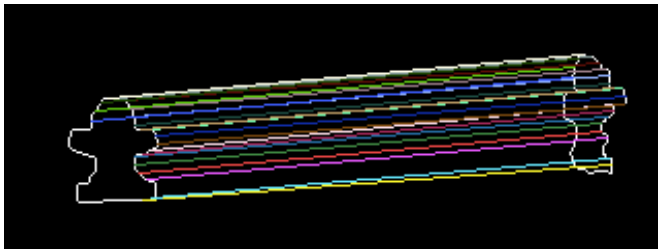


segmentation



segmentation filtered to colors similar to the template object gingerbread man (not shown)

euclidean transformed matches …need a perspective projection search for the other half



The remaining notes and snapshots on corners, stereo matching and matching under conditions of different lighting, pose and location are in file **doc/colorSegmentation3.pdf**

Note: tried the use of a segmentation filter based upon the atrous wavelet to limit the points to be matched. (later methods of MSER + HOGs proved more successful).