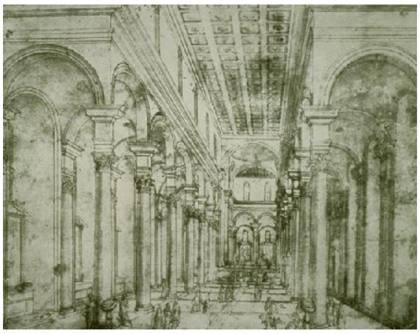
camera model

image acquisition

600 years ago



church of the Holy Spirit, Brunelleschi, 1436

today

the camera projects 3D points into an image plane

image formation: historical view

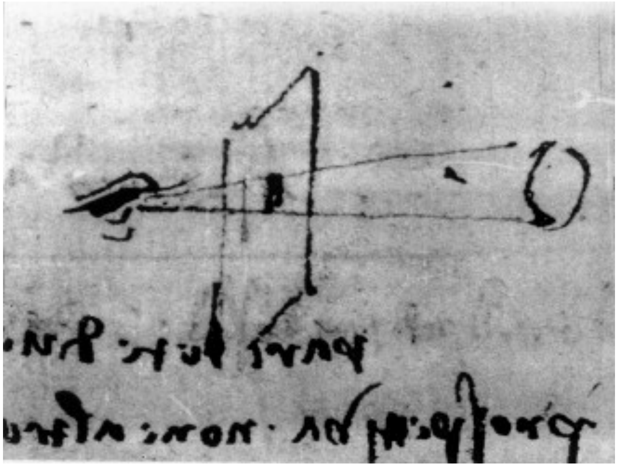
a) egiptian art

image formation: historical view

b) Chinese art

image formation: historical view

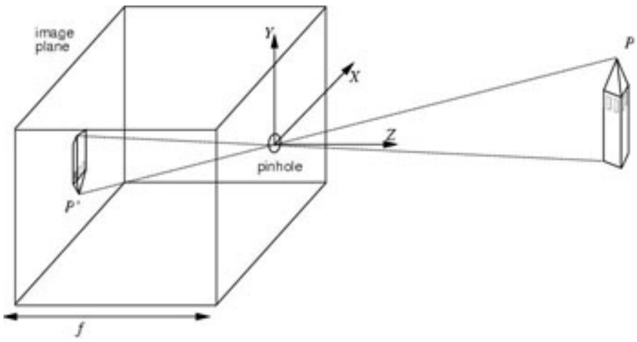
c) Renaissance



perspective projection

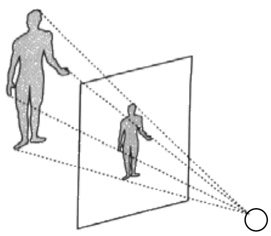
Ideal case:

Pin-Hole Camera



- Only light passing through the pin-hole reaches the image plane.
- Each image point corresponds to a unique 3D point

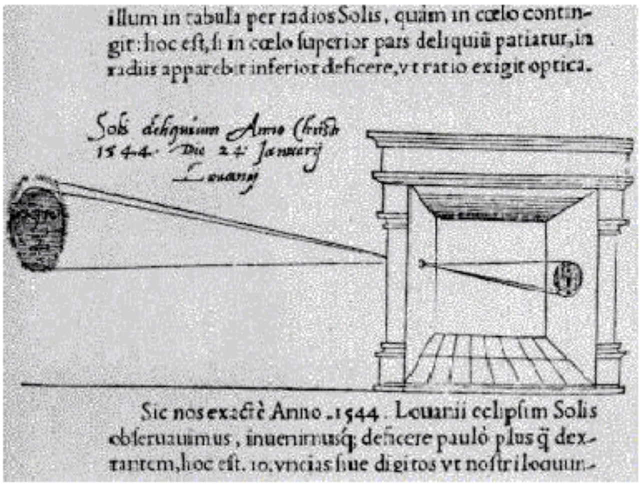
<u>Alternative Representation:</u>



Projection/Optical Centre

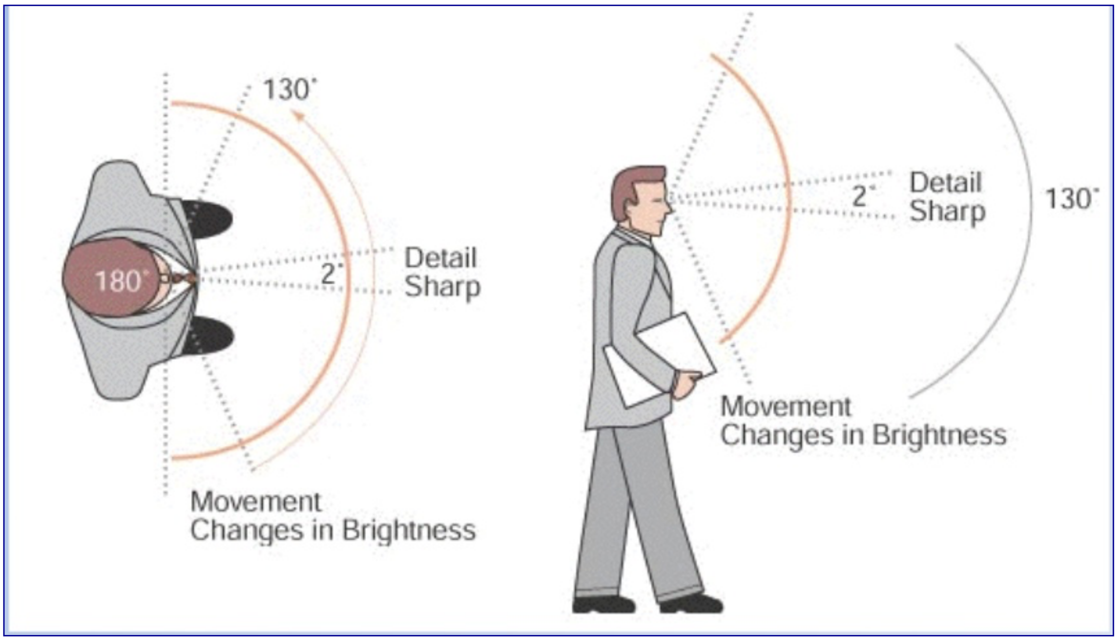
perspective projection

The observation of a Solar Eclipse



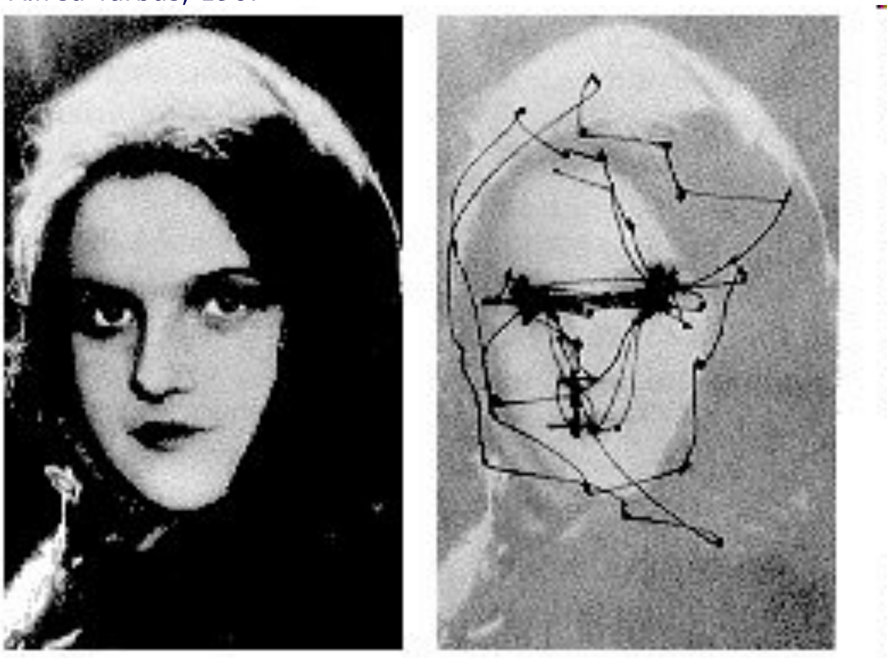
Gemma Frisius, De Radio Astronomica et Geometrica, 1545

Why do we move the eyes?



Human eye movements (eye-tracking)

Alfred Yarbus, 1967



Human eye movements

Q: Why?

•A: Limitations (?) of the eye. Only fovea is high-res enough for many tasks

Reflexive – gaze stabilization

•VOR: Stabilize for head movements

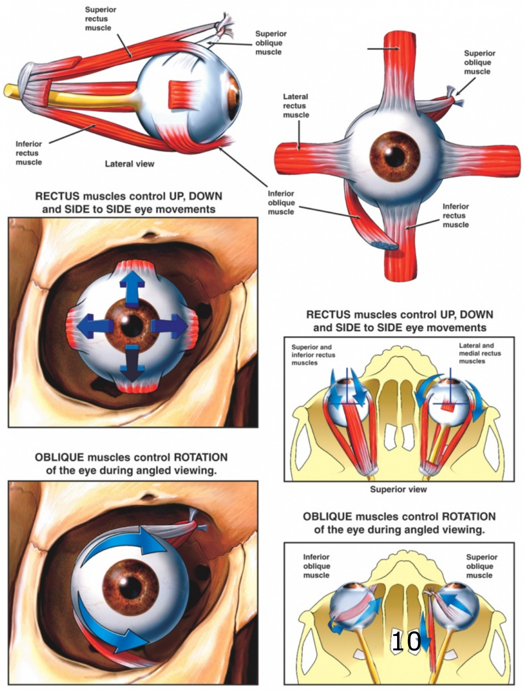
•Optokinetic: Stabilize for image motion

Voluntary – gaze shifting

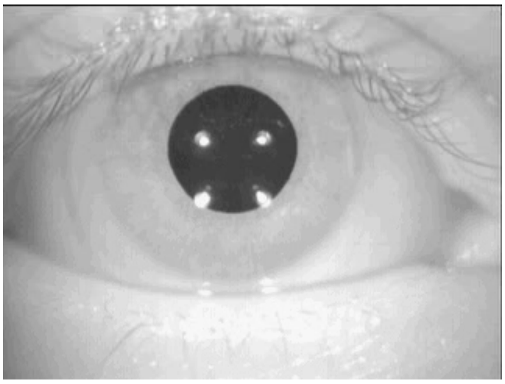
•Saccades: Acquire stationary target
  Rapid motion (25-30 ms) between
    fixations, up to 900º/s
      Saccades occur every 250-300 ms
      Also evidence for "micro-saccades"

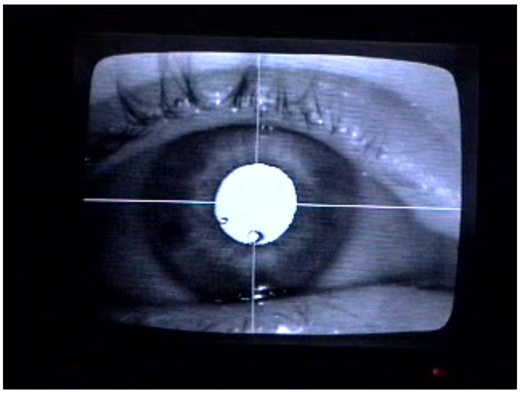
•Smooth pursuit: Acquire moving target
      Requires feedback

•Vergence: Acquire target in depth



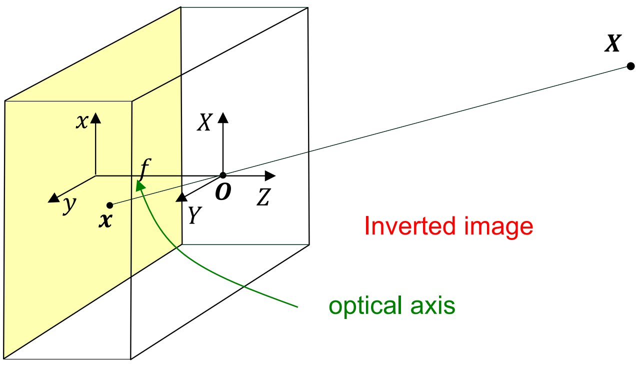
Human eye movements





Saccades (~3 / second)! Smooth pursuit

pin-hole camera model



Inverted image

optical axis

O – optical center (lenses)

coordinates systems attached to the camera

point in space: $\boldsymbol{X} = [X \quad Y \quad Z]^T$

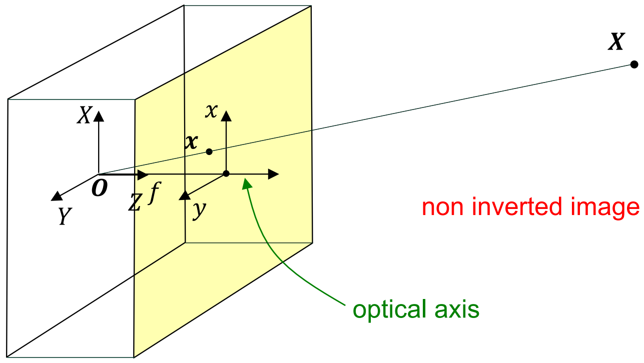
point in the image: $\boldsymbol{x} = [x \quad y]^T$

perspective projection

$$x = -f\frac{X}{Z} \qquad y = -f\frac{Y}{Z}$$

current cameras automatically compensate the image inversion.

perspective projection with frontal plane



O – optical center (lenses)

coordinates systems attached to the camera

point in space: $\boldsymbol{X} = [X \quad Y \quad Z]^T$

point in the image: $\boldsymbol{x} = [x \quad y]^T$

perspective projection

$$x = f\frac{X}{Z} \qquad y = f\frac{Y}{Z}$$

perspective projection (ideal case)

Normalized camera *f=1*.

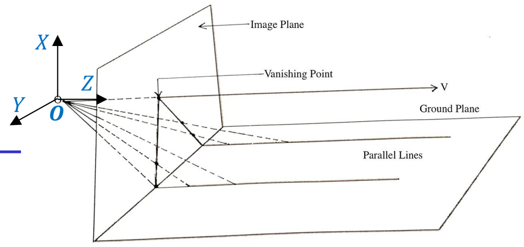
$$x = \frac{X}{Z} \qquad y = \frac{Y}{Z}$$

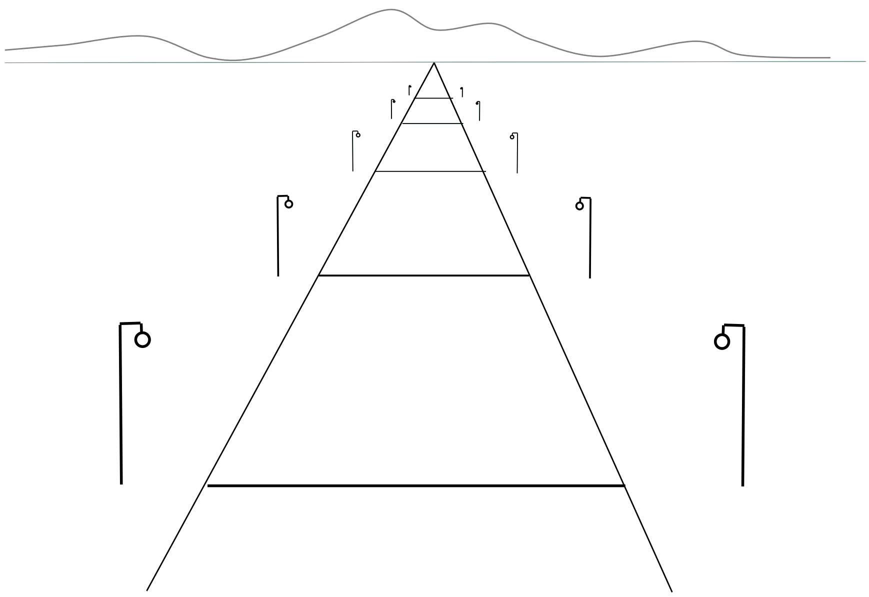
Using homogeneous coordinates,

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

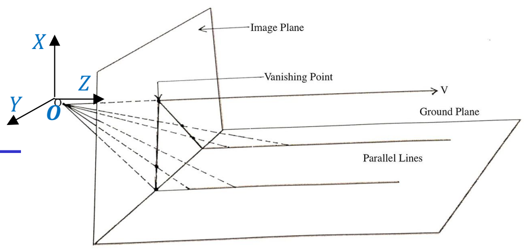
$$\lambda \tilde{\boldsymbol{x}} = [I \ 0]\tilde{\boldsymbol{X}} = \boldsymbol{X}$$

example: static case





example: static case (cont)



camera model: $x = f\frac{X}{Z} \qquad y = f\frac{Y}{Z}$

road points in 3D: $\boldsymbol{X} = (-h, a, Z), a \in [-10,10], Z > 0$

camera h m above the road, road width is 20m.

road points in 2D: $x = -f\frac{h}{Z} \qquad y = f\frac{a}{Z}$

conclusions:

i) vanishing point: $\lim_{Z\to\infty}(x, y) = (0,0)$

ii) the height of road lights decrease with the distance.

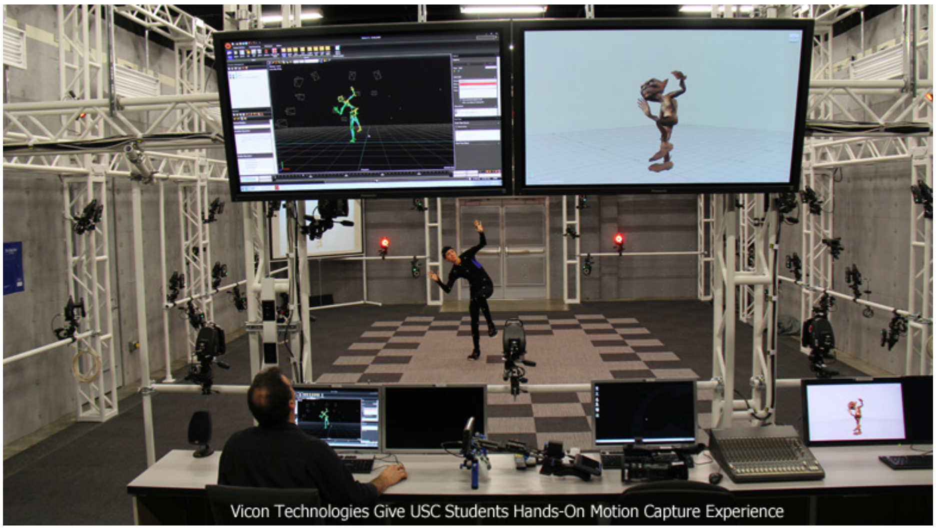
$$x = f\frac{-h}{Z} - f\frac{-h+l}{Z} = f\frac{l}{Z}$$

l – light height

iii) the ground position of road lights increases with the distance.

$$x = -f\frac{h}{Z}$$

full camera model

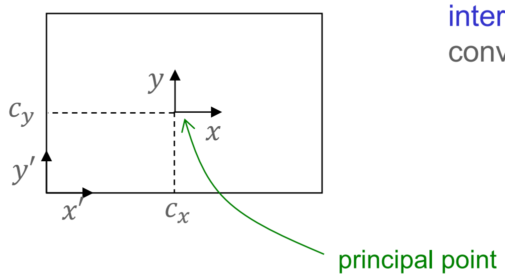


http://www.cgarena.com/newsworld/vicon-usc-motion-capture.php

the previous model was derived under simplifying hypotheses:

- extrinsic parameters: world frame located at an arbitrary position
- internal parameters: focal length, scale factors, principal point

intrinsic parameters



internal model:
conversion from metric coordinates to pixels

$$x' = f s_x x + c_x \qquad (x,y) \text{ in metric units}$$
$$y' = f s_y y + c_y \qquad (x',y') \text{ in pixels}$$

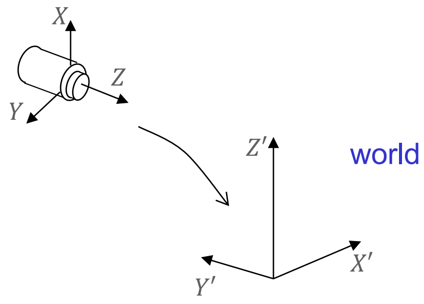
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} f s_x & 0 & c_x \\ 0 & f s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\tilde{x}' \sim K \tilde{x}$$

K upper triangular matrix

extrinsic parameters

camera in arbitrary position



$$\boldsymbol{X} = R\boldsymbol{X}' + \mathbf{T}$$

conversion:
world → camera coordinate frames

T - origin of the world coordinate frame expressed in camera coordinates
R – expresses the rotation between the world and camera frames

$$\tilde{\boldsymbol{X}} = \begin{bmatrix} R & \boldsymbol{T} \\ \mathbf{0}^T & 1 \end{bmatrix} \tilde{\boldsymbol{X}}'$$

homogeneous coordinates

Combining with the projection we obtain

$$\tilde{\boldsymbol{x}} \sim [R \quad \boldsymbol{T}]\tilde{\boldsymbol{X}}'$$

$\boldsymbol{R}, \boldsymbol{T}$ are camera extrinsic parameters

full perspective model

Camera model:

$$\tilde{\boldsymbol{x}} \sim K[R|\boldsymbol{T}]\tilde{\boldsymbol{X}}$$

$$\tilde{\boldsymbol{x}} \sim P\tilde{\boldsymbol{X}}$$

Notation:
for simplicity, $\boldsymbol{x}'$ and $\boldsymbol{X}'$
were replaced by $\boldsymbol{x}, \boldsymbol{X}$

P is a *3×4* matrix, denoted as camera matrix.

the camera model is linear in homogeneous coordinates!

camera model in Cartesian coordinates

Homogeneous coordinates camera matrix (3x4)

$$\lambda \tilde{\boldsymbol{x}} = P\tilde{\boldsymbol{X}} \ , \qquad \lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix} \tilde{\boldsymbol{X}}$$

Cartesian coordinates

$$x = \frac{p_1^T \tilde{\boldsymbol{X}}}{p_3^T \tilde{\boldsymbol{X}}} \qquad y = \frac{p_2^T \tilde{\boldsymbol{X}}}{p_3^T \tilde{\boldsymbol{X}}}$$

11 degrees of freedom

Matrix P is specified apart from a scale factor!

exercises

vanishing point

straight line in homogeneous coordinates

$$\tilde{\boldsymbol{X}} = \tilde{\boldsymbol{X}}_0 + \alpha\tilde{\boldsymbol{T}} \qquad \tilde{\boldsymbol{X}} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tilde{\boldsymbol{X}}_0 = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}, \tilde{\boldsymbol{T}} = \begin{bmatrix} t_x \\ t_y \\ t_z \\ 0 \end{bmatrix}$$

$$\lambda\tilde{\boldsymbol{x}} = P\tilde{\boldsymbol{X}} = P(\tilde{\boldsymbol{X}}_0 + \alpha\tilde{\boldsymbol{T}}) = P\tilde{\boldsymbol{X}}_0 + \alpha P\tilde{\boldsymbol{T}}$$

point at infinity

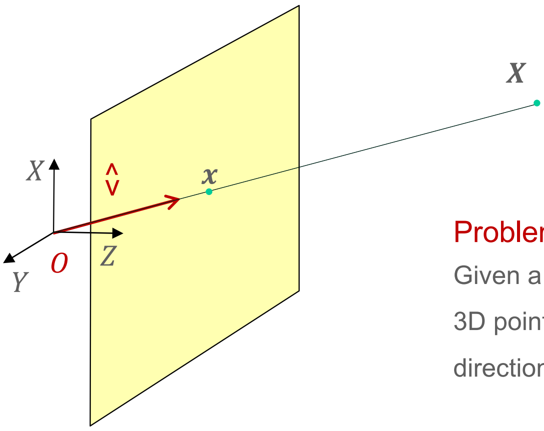
When α goes to infinity

$$\tilde{\boldsymbol{x}} \sim P\tilde{\boldsymbol{T}} \qquad x = \frac{p_1^T\tilde{T}}{p_3^T\tilde{T}} \qquad y = \frac{p_2^T\tilde{T}}{p_3^T\tilde{T}}$$

The vanishing point does not depend on p_0.
Each set of parallel lines has its own vanishing point.

Problem: solve this problem using Cartesian coordinates.

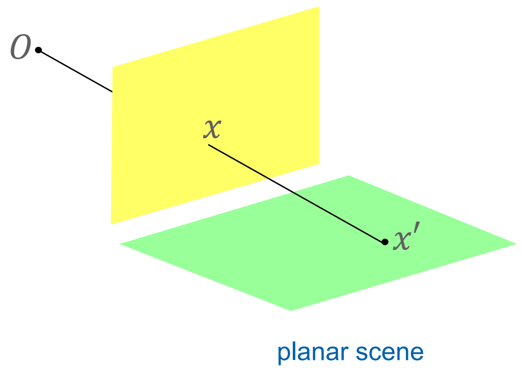
optical center and optical ray



Problem:

Given a camera model $(K, R, \boldsymbol{T})$ and the projection $\boldsymbol{x}$ of a 3D point $\boldsymbol{X}$, determine the camera optical center and direction of the optical ray (in world coordinates).

planar scene (1 camera)



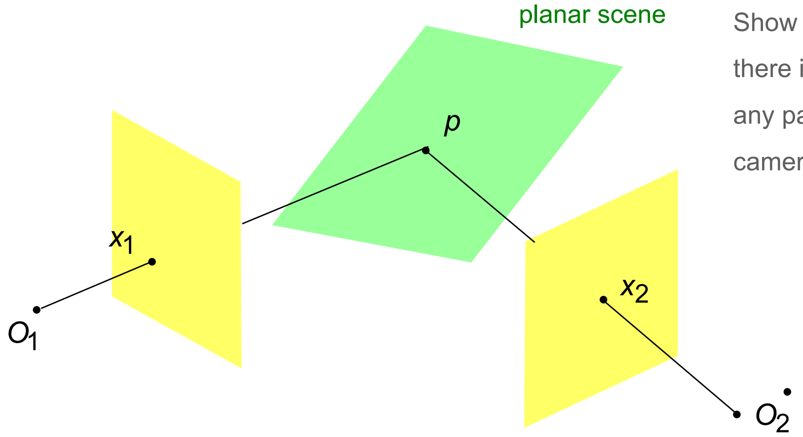
Problem:
Show that there is a projective transformation between the local coordinates $\boldsymbol{x}'$ of a point in the plane and projection $\boldsymbol{x}$.

$$\boxed{\tilde{\boldsymbol{x}} \sim H\tilde{\boldsymbol{x}}'} \qquad \tilde{\boldsymbol{x}}' = [\tilde{x} \;\; \tilde{y} \;\; \tilde{w}]^T$$

The converse is also true $\tilde{\boldsymbol{x}}' \sim H^{-1}\tilde{\boldsymbol{x}}$

Problem:
Homographies, or 2D projective transformations, map points between two planes. How do homographies map **lines**, $\boldsymbol{u}$ and $\boldsymbol{u}'$, between those planes?

planar scene (2 cameras)



Problem:
Show that if a point $\boldsymbol{p}$ lies on a 3D plane, there is a projective transformation between any pair of projections of $\boldsymbol{p}$ in any two cameras. $\boxed{\tilde{x}_1 \sim H\ \tilde{x}_2}$

homographies and lines in the projective plane

Homographies represent projective transformations between points in planes (either between the world plane and the image plane, or between the images of two cameras observing the same plane in 3D.
An interesting question is to understand how they transform lines.

Line equation: $u_1\lambda x + u_2\lambda y + u_3\lambda = 0$ or $\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{m}} = \boldsymbol{0}$

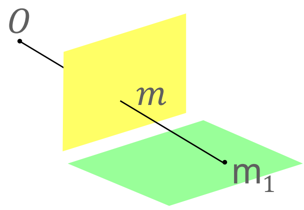
Given a homography $\boldsymbol{H}$ between two planes: $\tilde{\boldsymbol{m}} = \boldsymbol{H}\tilde{\boldsymbol{m}}_1$

We have: $\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{m}} = \boldsymbol{0} \Leftrightarrow \tilde{\boldsymbol{u}}^T\boldsymbol{H}\tilde{\boldsymbol{m}}_1 = 0$

$\Leftrightarrow (\tilde{\boldsymbol{u}}^T\boldsymbol{H})\tilde{\boldsymbol{m}}_1 = 0$

$\Leftrightarrow (\boldsymbol{H}^T\tilde{\boldsymbol{u}})^T\tilde{\boldsymbol{m}}_1 = 0$

$\Leftrightarrow \tilde{\boldsymbol{u}}_1{}^T\tilde{\boldsymbol{m}}_1 = 0$ with $\tilde{\boldsymbol{u}}_1 = \boldsymbol{H}^T\tilde{\boldsymbol{u}}$



planar scene

Conclusion: while H maps points between the two planes, $\tilde{\boldsymbol{m}} = \boldsymbol{H}\tilde{\boldsymbol{m}}_1$,

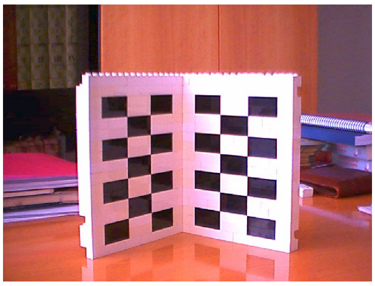
the mapping between lines in those planes is given by: $\tilde{\boldsymbol{u}} = \boldsymbol{H}^{-T}\tilde{\boldsymbol{u}}_1$

camera calibration

camera calibration

problem

Given a set of 3D points and their projections on a camera, we wish to estimate the camera pose and intrinsic parameters.



data: $\{(\boldsymbol{x}_i, \boldsymbol{X}_i), i = 1, \cdots, N\}$

camera model (homogeneous coordinates)

$$\lambda\tilde{\boldsymbol{x}} = P\tilde{\boldsymbol{X}}$$

$$\lambda\tilde{\boldsymbol{x}} = K[R \mid \boldsymbol{T}]\tilde{\boldsymbol{X}}$$

$\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{w}),\ \tilde{\boldsymbol{X}} = (\tilde{X}, \tilde{Y}, \tilde{Z}, \tilde{W})$

(Cartesian coordinates)

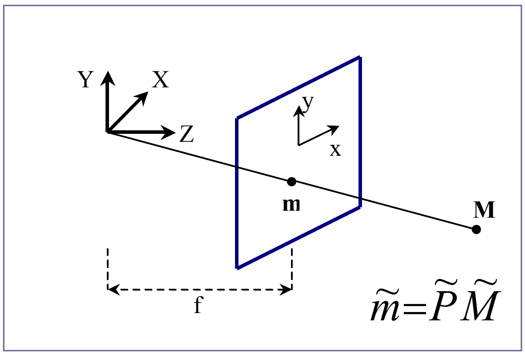
$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{p_3^T\tilde{\boldsymbol{X}}}\begin{bmatrix} p_1^T\tilde{\boldsymbol{X}} \\ p_2^T\tilde{\boldsymbol{X}} \end{bmatrix},$$

$P \epsilon \mathbb{R}^{3\times4}, K, R \epsilon \mathbb{R}^{3\times3}, \boldsymbol{T} \epsilon \mathbb{R}^3,$

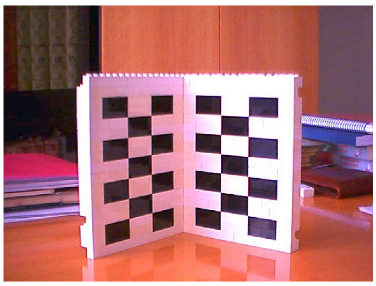
K is upper triangular

R is a rotation matrix (orthogonal, $\det R = 1$)

camera calibration



Matrix $\tilde{P}$ contains information about the camera's intrinsic and extrinsic parameters, often unknown, as well as the perspective projection. Camera calibration is the process of estimating matrix $\tilde{P}$.

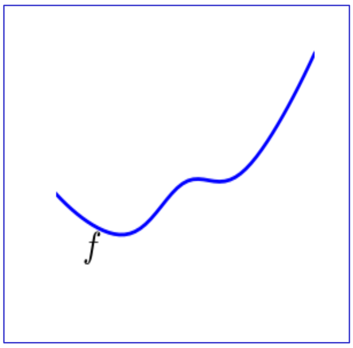


We have :

$$\begin{bmatrix} \lambda x_p \\ \lambda y_p \\ \lambda \end{bmatrix} = \underbrace{\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}}_{\tilde{P}}{}^{W}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$\tilde{P}$ is defined up to a scale factor. We can set : $p_{34} = 1$

non linear least squares



Camera (non linear) model

$$\boldsymbol{x} = f(\boldsymbol{X}, \boldsymbol{p})$$

Cost function to minimize

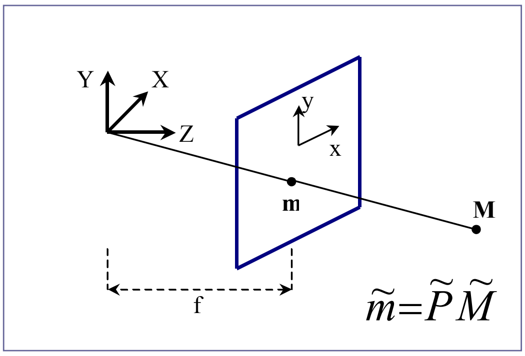
$$\boldsymbol{E}(\boldsymbol{p}) = \sum_i \|\boldsymbol{x}_i - f(\boldsymbol{X}_i, \boldsymbol{p})\|^2$$

this is a non linear cost function with many local minima.

minimization can be done using numerical algorithms.

drawbacks: with many local minima, we need a good initial estimate.

linear camera calibration



Expressions can be written as:

$$x_p = \frac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + 1}$$

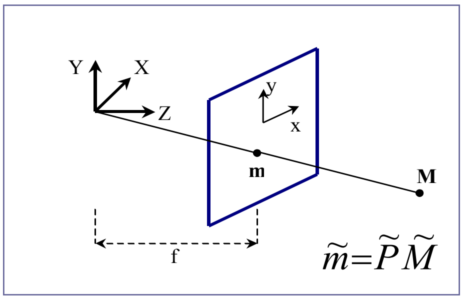
$$y_p = \frac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + 1}$$

In matrix form :

$$\begin{bmatrix} X & Y & Z & 1 & 0 & 0 & 0 & 0 & -Xx_p & -Yx_p & -Zx_p \\ 0 & 0 & 0 & 0 & X & Y & Z & 1 & -Xy_p & -Yy_p & -Zy_p \end{bmatrix} \underbrace{\begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ \vdots \\ p_{33} \end{bmatrix}}_{\theta} = \begin{bmatrix} x_p \\ y_p \end{bmatrix}$$

11 unknowns
2 eqs./point => minimum 6 points for calibration

linear camera calibration



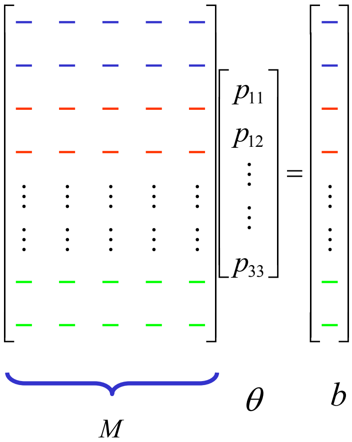
For N pairs of corresponding points we have the following system of linear equations:

Direct linear transform:
When the system is overdetermined (2N>11), we can use least-squares:

$$\hat{\theta}_{LS} = \underset{\theta}{Min} \| M \cdot \theta - b \|^2$$

$$\hat{\theta}_{LS} = \left(M^T M\right)^{-1} M^T b$$

2N eqs.



11 unknowns

Problem: outliers → Robust estimation of $\tilde{P}$

linear estimation of camera matrix

$$\begin{cases} x = \dfrac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}} = \dfrac{\boldsymbol{X}^T\boldsymbol{p}_1}{\boldsymbol{X}^T\boldsymbol{p}_3} \\ y = \dfrac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}} = \dfrac{\boldsymbol{X}^T\boldsymbol{p}_2}{\boldsymbol{X}^T\boldsymbol{p}_3} \end{cases}$$

multiplying both members by the denominator...

$$\underbrace{\begin{bmatrix} \boldsymbol{X}_1^T & 0^T & -x_1\boldsymbol{X}_1^T \\ 0^T & \boldsymbol{X}_1^T & -y_1\boldsymbol{X}_1^T \\ \cdots & \cdots & \cdots \\ \boldsymbol{X}_N^T & 0^T & -x_N\boldsymbol{X}_N^T \\ 0^T & \boldsymbol{X}_N^T & -y_N\boldsymbol{X}_N^T \end{bmatrix}}_{M} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Direct Linear transform (DLT)

optimization

$$\pi = \arg\min_{p} \|M\boldsymbol{p}\|^2 = \arg\min_{p} \boldsymbol{p}^T M^T M \boldsymbol{p} \qquad \|\boldsymbol{p}\|^2 = \boldsymbol{p}^T\boldsymbol{p} = 1$$

prove!

Solution: π is the eigenvector associated with the smallest eigenvalue of $M^T M$.

estimation of pose and intrinsic parameters

If we assume that the camera matrix is known, and we wish to obtain the pose and intrinsic parameters.

$$P = [P_0 \,|\boldsymbol{q}] = K[R \,|T] \qquad P_0 = KR, \quad \boldsymbol{q} = K\boldsymbol{T}$$

How can we decompose a square matrix P_0 into the product of a upper triangular and a rotation matrix?

QR decomposition performs something similar

$$M = Q'R'$$

upper triangular

unitary matrix

$$P_0^{-1} = R^{-1}K^{-1} = Q'R'$$

upper triangular

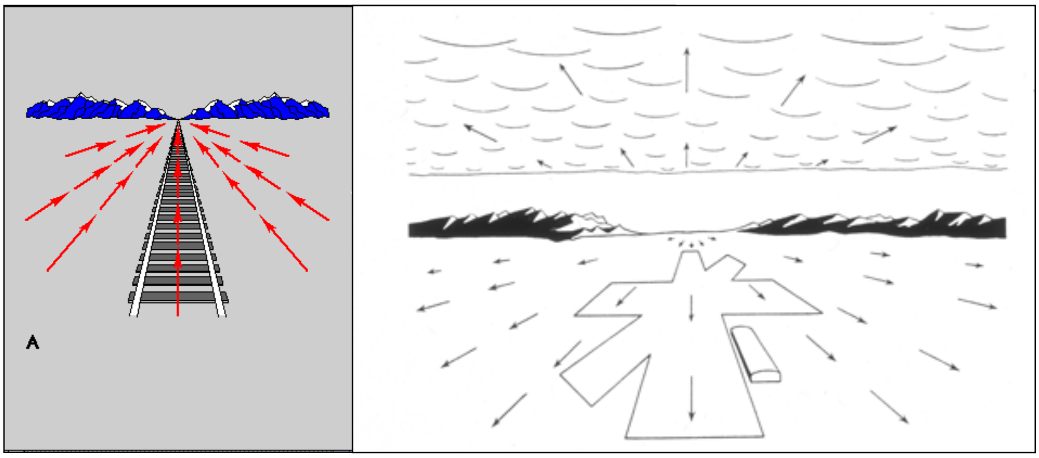
unitary matrix

$$(P_0)^{-1} = Q'R'$$

$$R = Q'^{-1}, K = R'^{-1}$$
$$\boldsymbol{T} = K^{-1}\boldsymbol{q}$$

Motion perception

camera motion



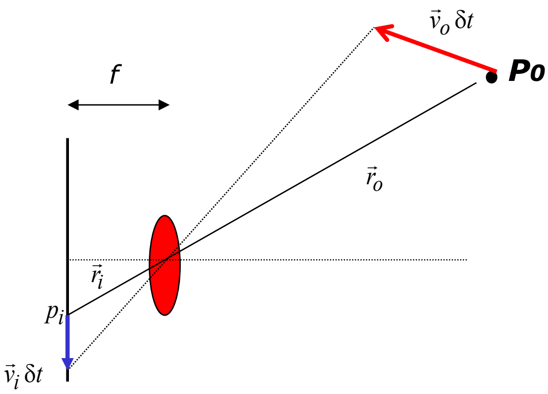
http://www.cns.nyu.edu/~david/ example from J. J. Gibson

Motion Field

Objects moving in front of static camera or

Moving camera in a static environment

⟹ Motion on the image plane



The velocity, $\vec{v}_0$ of Po induces a velocity $\vec{v}_i$ in the image.

$$\vec{v}_0 = \frac{\partial \vec{r}_o}{\partial t} \qquad \vec{v}_i = \frac{\partial \vec{r}_i}{\partial t}$$

Vectors $\vec{r}_o$ and $\vec{r}_i$ are related by the perspective equation:

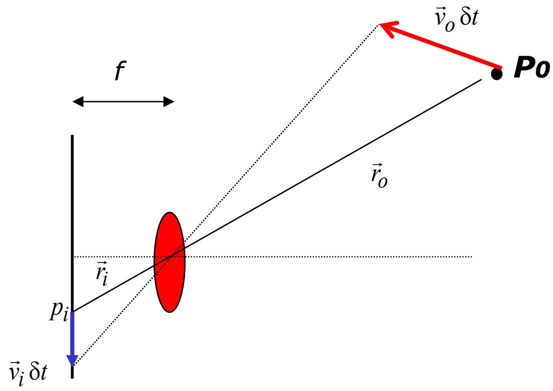
$$\frac{\vec{r}_i}{f} = \frac{\vec{r}_o}{\vec{r}_o \cdot \hat{Z}}$$

Motion Field

Differenciating, it yields:

$$\vec{v}_i = f\,\frac{(\vec{r}_o\cdot\hat{Z})\vec{v}_o - (\vec{v}_o\cdot\hat{Z})\vec{r}_o}{(\vec{r}_o\cdot\hat{Z})^2}$$

$$= f\,\frac{(\vec{r}_o\times\vec{v}_o)\times\hat{Z}}{(\vec{r}_o\cdot\hat{Z})^2}$$



Relationship between a 3D velocities and the corresponding image velocity.

This movement on the image plane is referred to as the motion field

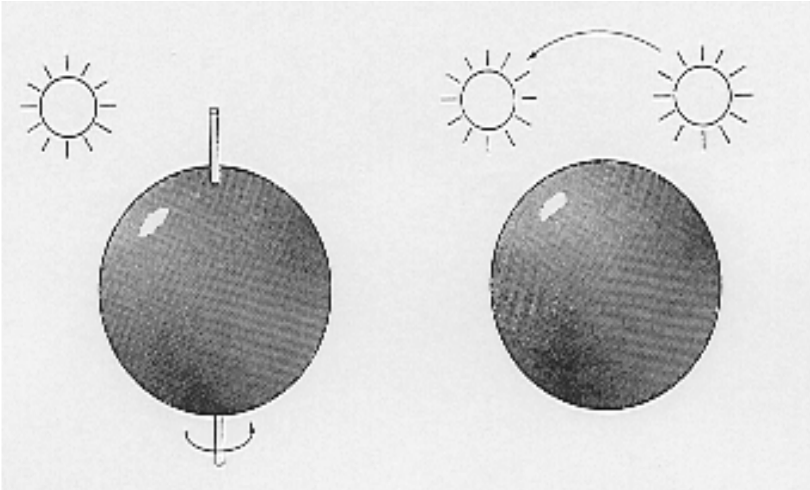
Optic flow

- The apparent movement of brightness patterns in the image is called the **optical flow**

Optic flow

- The apparent movement of brightness patterns in the image is called the **optical flow**

- The optical flow is not always coincident with the motion field:



43

"optical flow"

The motion of the projected points in the image is called motion field.

position velocity

$$(x, y) \rightarrow (u, v)$$

motion of the camera $\dot{\boldsymbol{X}} = \boldsymbol{T} + \boldsymbol{\omega} \times \boldsymbol{X}$

motion of the point in the camera frame $\dot{\boldsymbol{X}} = -\boldsymbol{T} - \boldsymbol{\omega} \times \boldsymbol{X}$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = -\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} - \begin{bmatrix} \omega_y Z - \omega_z Y \\ \omega_z X - \omega_x Z \\ \omega_x Y - \omega_y X \end{bmatrix}$$

camera model

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} X \\ Y \end{bmatrix}$$

Motion field

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \frac{1}{Z^2} \begin{bmatrix} \dot{X}Z - \dot{Z}X \\ \dot{Y}Z - \dot{Z}Y \end{bmatrix}$$

44

“optical flow”

After algebraic manipulations, we obtain

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{Z}\underbrace{\begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \end{bmatrix}\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}}_{\text{translation component}} + \underbrace{\begin{bmatrix} xy & -(1+x^2) & y \\ (1+y^2) & -xy & -x \end{bmatrix}\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}}_{\text{rotation component}}$$

The translation component carries information about the scene Z(x,y), in a non linear way.

The rotation component is linear and does not provide any information about the scene.

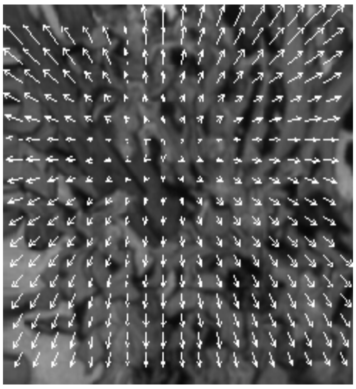
There are methods that attempt to separate both components.

“optical flow”: translation motion

If there is translation motion only (no rotation)

$$u = \frac{t_z}{Z}\left(x - \frac{t_x}{t_z}\right), \quad v = \frac{t_z}{Z}\left(y - \frac{t_y}{t_z}\right)$$

The optical flow has a radial structure, with focus of expansion $\left(\frac{t_x}{t_z}, \frac{t_y}{t_z}\right)$.



The scale of the scene is ambiguous. If we multiply the velocity T and the depth Z by the same scale factor we obtain the same optical flow.

We cannot distinguish a camera moving slowly in a small scene from a camera moving fast in a large scene.

“optical flow”: rotation motion

If there is no translation motion,

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} xy & -(1+x^2) & y \\ (1+y^2) & -xy & -x \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

The optical motion does not depend on the scene. The motion parameters can be retrieved from the observed data easily, since the model is linear.