Model-based 3D tracking

Prof. Didier Stricker
Doz. Gabriele Bleser

Two different levels

Recognition (2D)

Tracking (2D)

Where in the image … ?

Pose estimation,
Tracking in 3D

Where in the world … ?

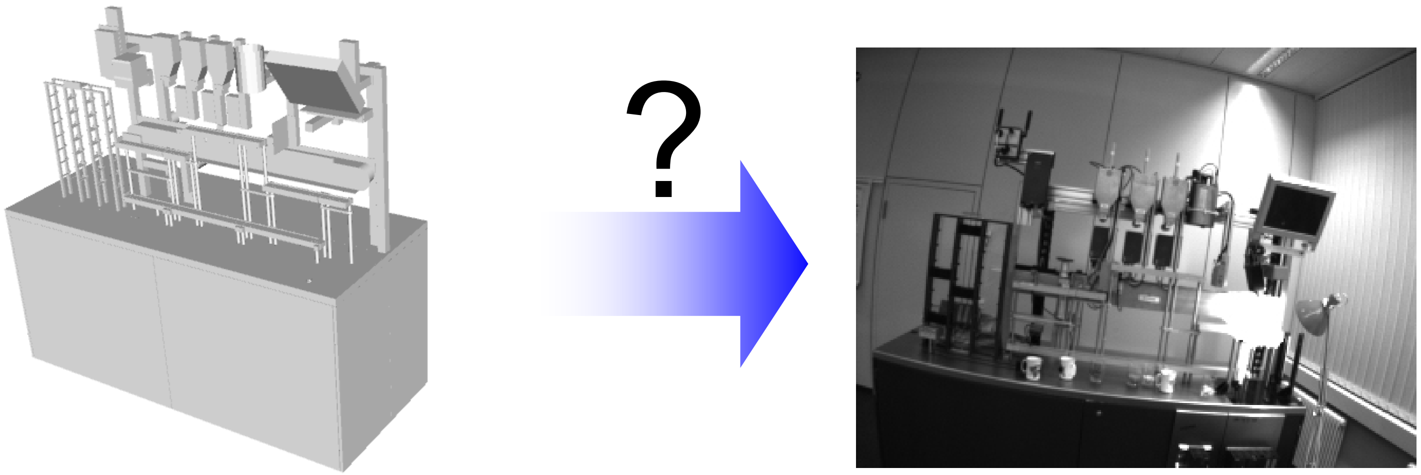
Initial Pose Estimation

Recognition/Tracking (x,y)

Pose estimation (X,Y,Z,ϕ,ψ,γ)

Problem statement

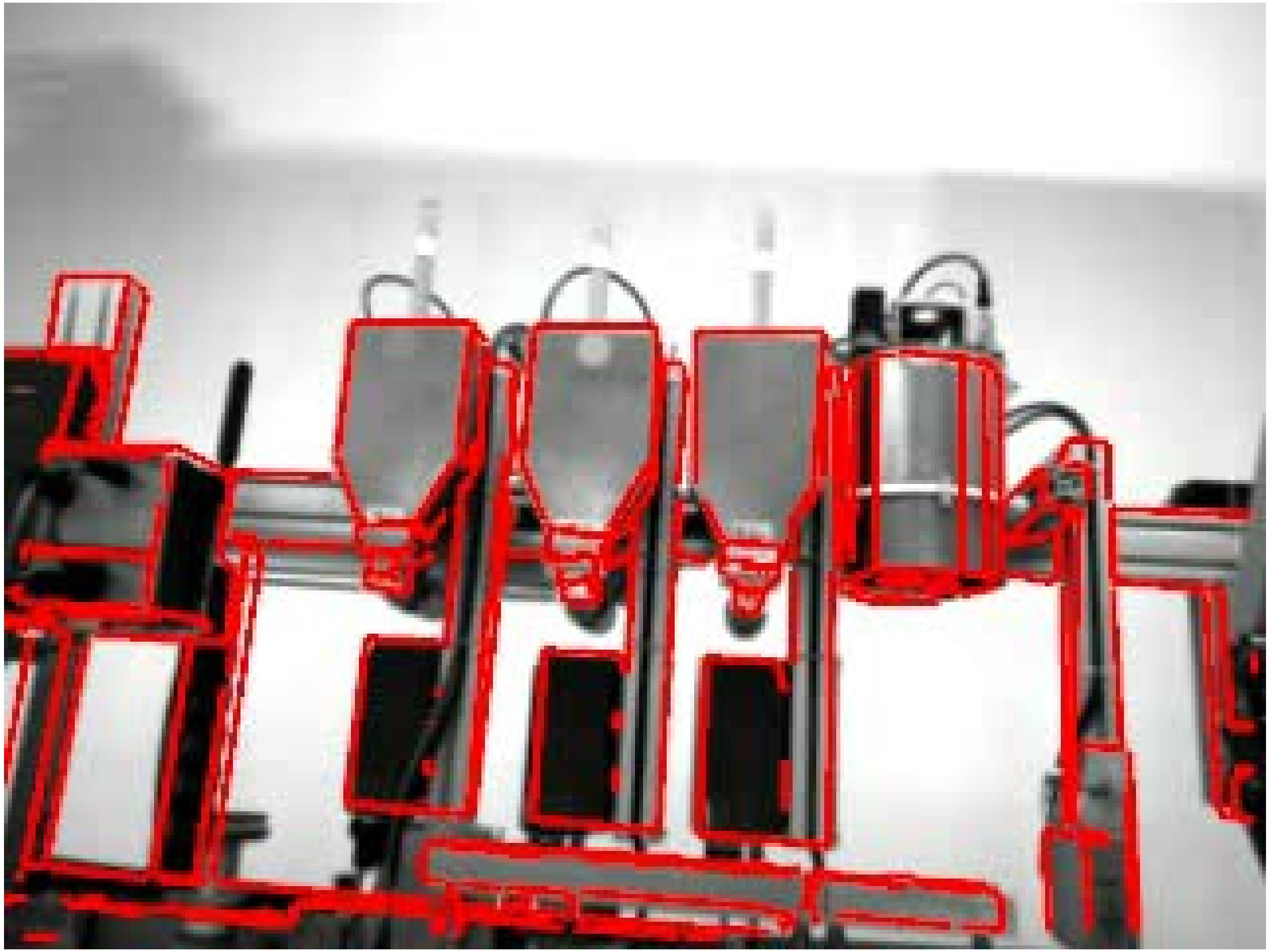
To compute the position and orientation of the object by tracking it through the video sequence



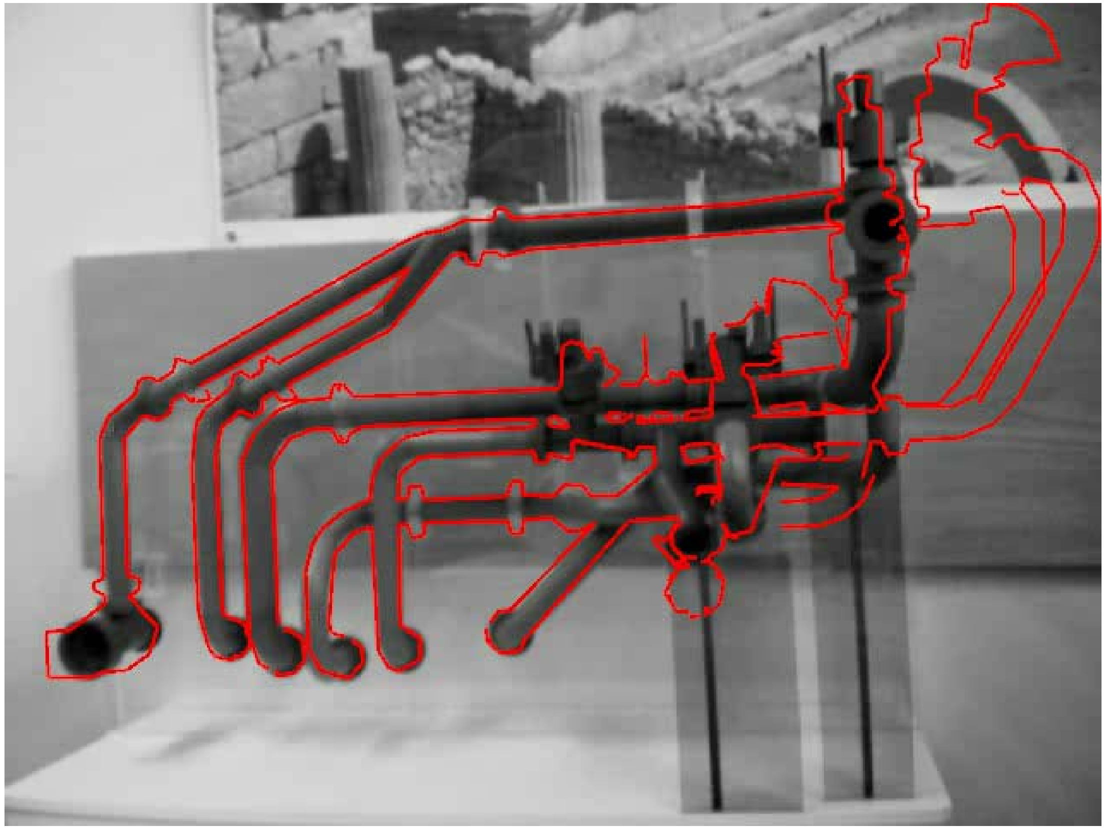
- The description of the object consists of a 3D CAD model
- Objects can be complex and have little structures (texture)

→ use contours / edges

Tracking with a 3D countour model



Tracking with a 3D countour model



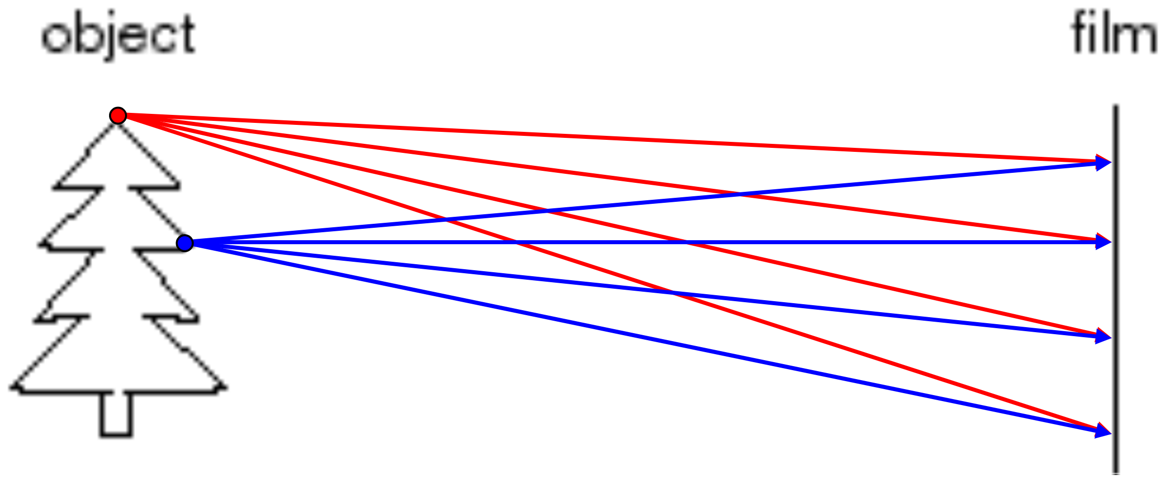
- Tracking using the silhouette
- The contour has to be recomputed online
- The CAD and model does not coincinde completelty – this will create outliers.

(Howaldtwerke-Deutsche Werft)

Outline

- Part I: Camera model and pose computation
- Part II: 3D countour tracking (RAPID)

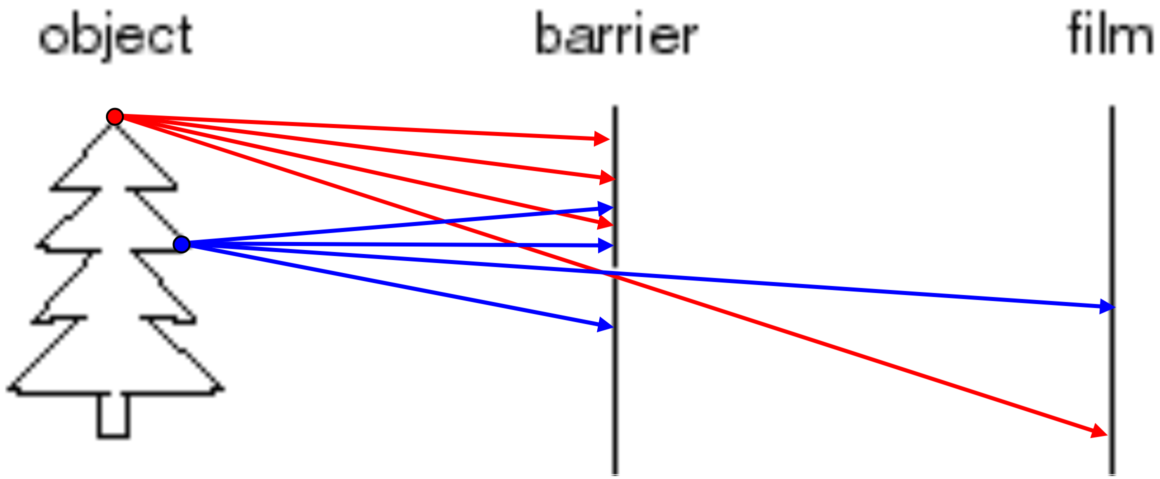
Image formation



Let’s design a camera

- Idea 1: put a piece of film in front of an object
- Do we get a reasonable image?

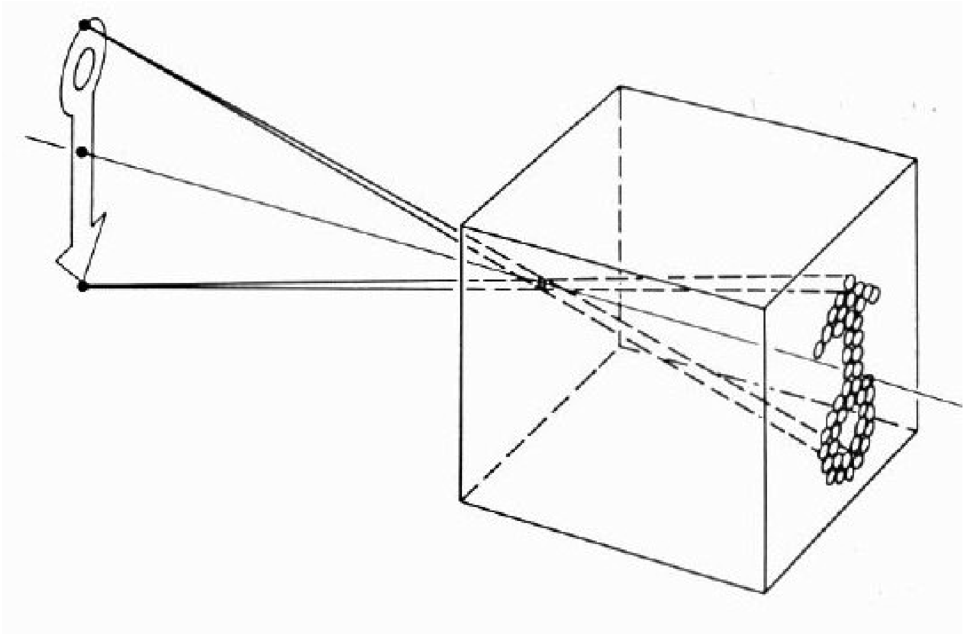
Pinhole camera



Add a barrier to block off most of the rays

- This reduces blurring
- The opening known as the **aperture**
- How does this transform the image?

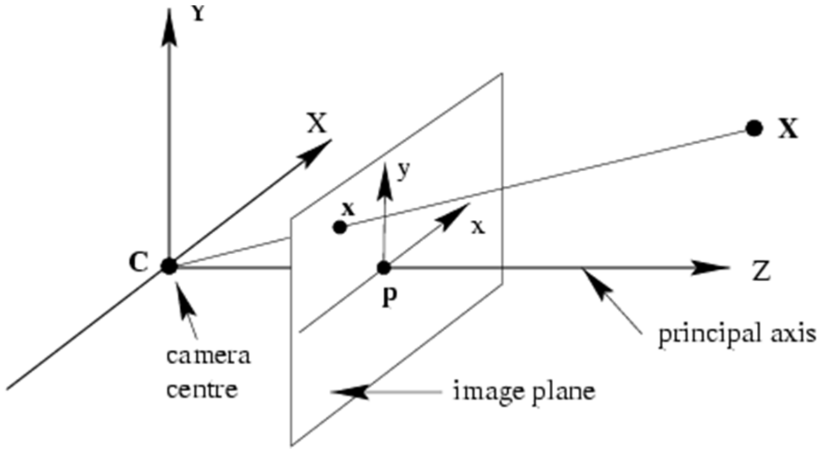
Camera Obscura



The first camera

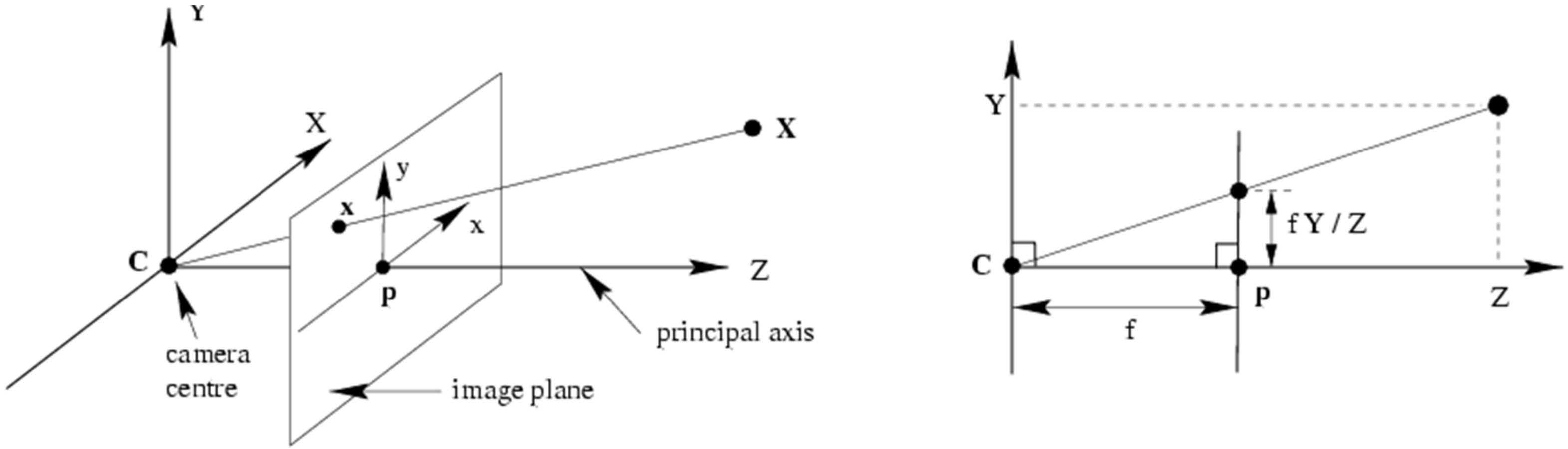
- Known to Aristotle

Camera coordinate system



- **Principal axis:** line from the camera center perpendicular to the image plane
- **Normalized (camera) coordinate system:** camera center is at the origin and the principal axis is the z-axis
- **Principal point (p):** point where principal axis intersects the image plane (origin of normalized coordinate system)

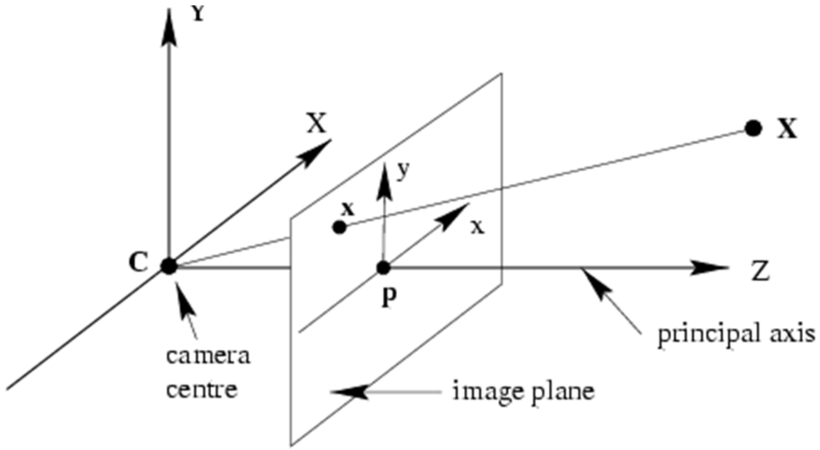
Modeling projection

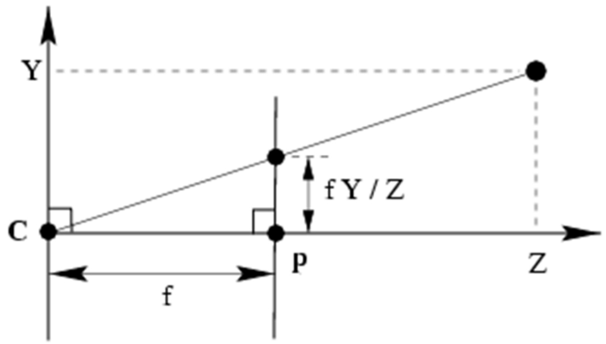


The coordinate system

- We will use the pin-hole model as an approximation
- Put the optical center (**C**enter **O**f **P**rojection) at the origin
- Put the image plane (**P**rojection **P**lane) *in front* of the COP
- The camera looks down the *negative* z axis
  - we need this if we want right-handed-coordinates

Modeling projection





$$(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

Homogeneous coordinates

Is this a linear transformation?

- no—division by z is nonlinear

Trick: add one more coordinate:

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

homogeneous image coordinates

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

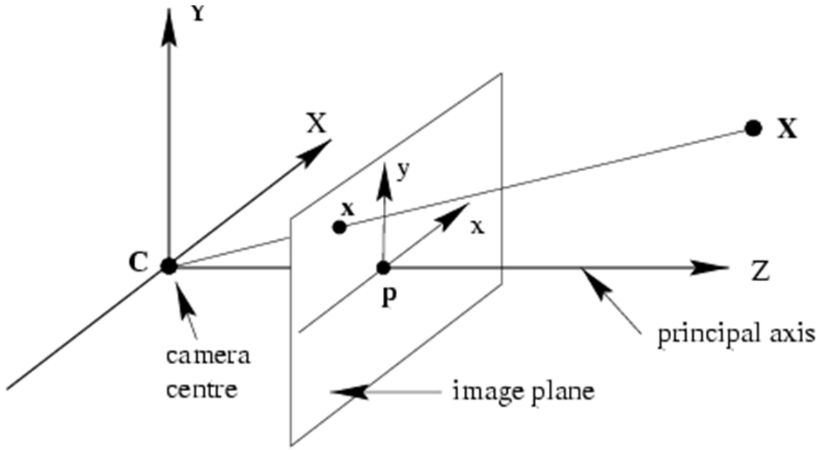
homogeneous scene coordinates

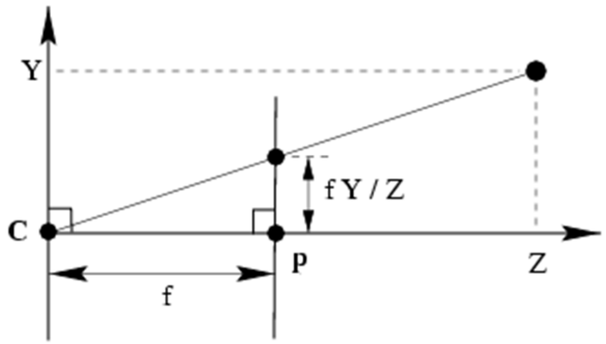
Converting *from* homogeneous coordinates

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

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

Pinhole camera model



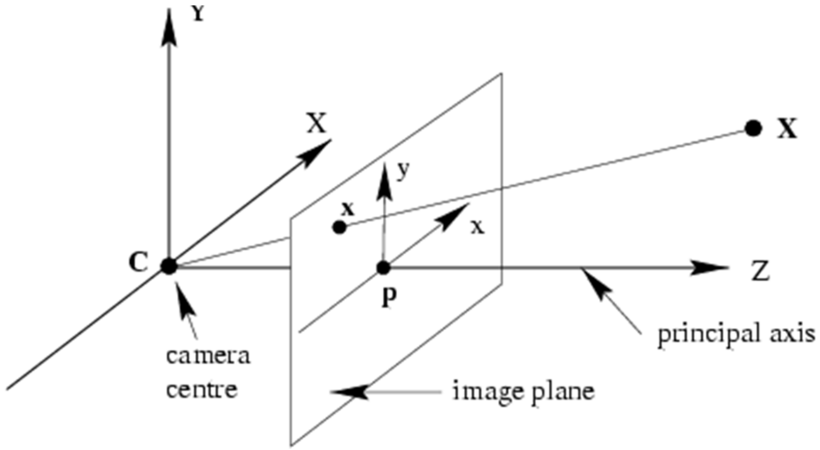


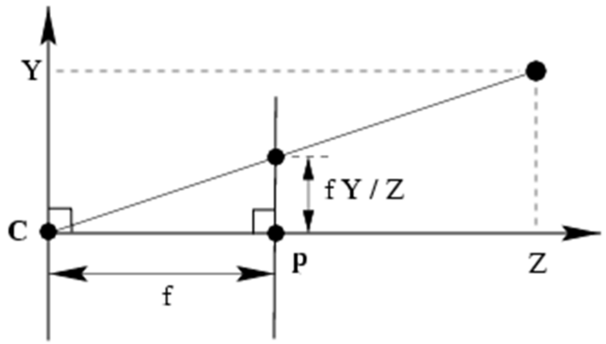
$$(X, Y, Z) \mapsto (f X / Z, f Y / Z)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f X \\ f Y \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\mathrm{x} = \mathrm{P}\mathrm{X}$$

Pinhole camera model

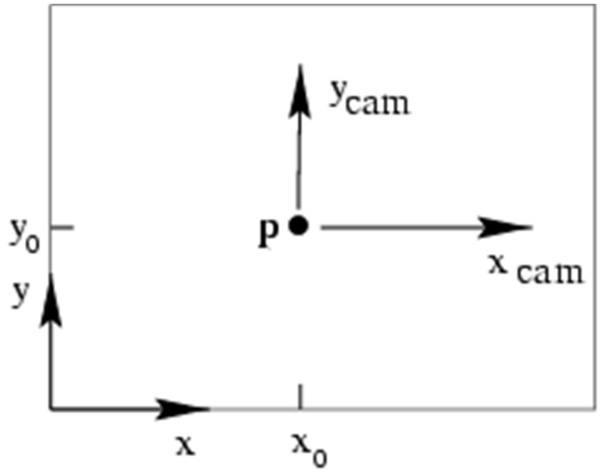




$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\mathrm{x} = \mathrm{PX} \qquad \mathrm{P} = \mathrm{diag}(f, f, 1)\left[\mathrm{I} \mid 0\right]$$

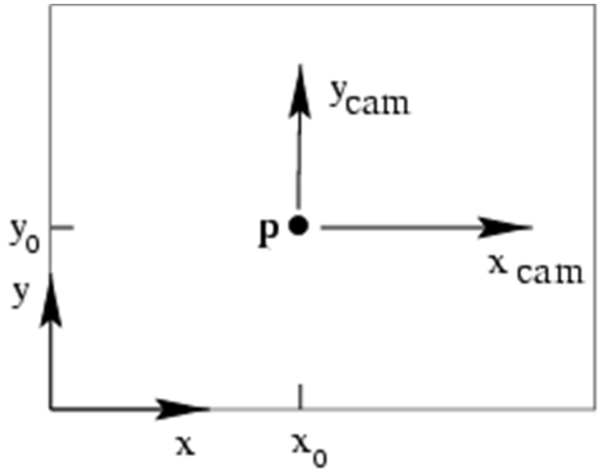
Principal point offset



principal point: (p_x, p_y)

- Camera coordinate system: origin is at the prinicipal point
- Image coordinate system: origin is in the corner

Principal point offset

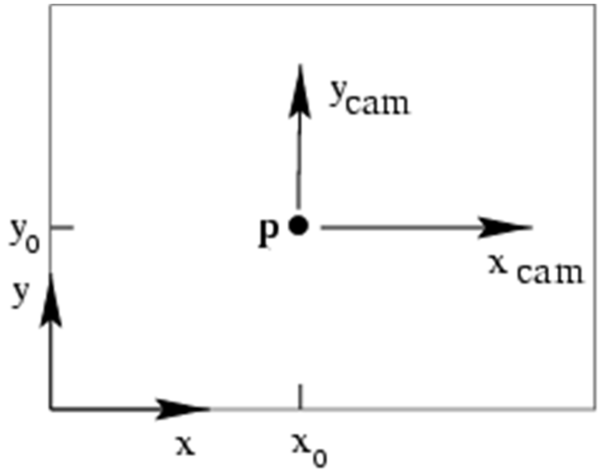


principal point: (p_x, p_y)

$$(X, Y, Z) \mapsto (f\,X/Z + p_x, f\,Y/Z + p_y)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f\,X + Z\,p_x \\ f\,Y + Z\,p_x \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Principal point offset

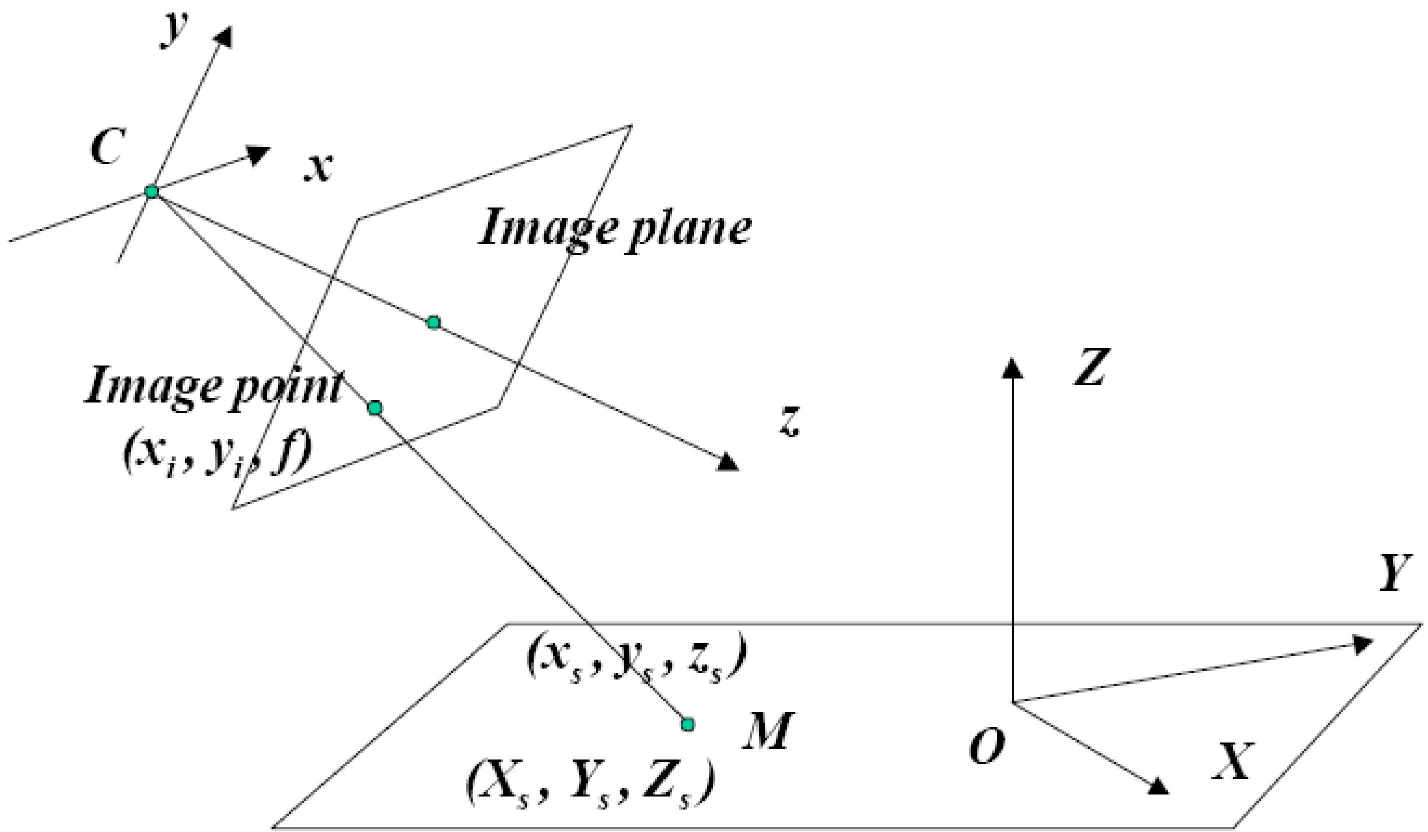


principal point: (p_x, p_y)

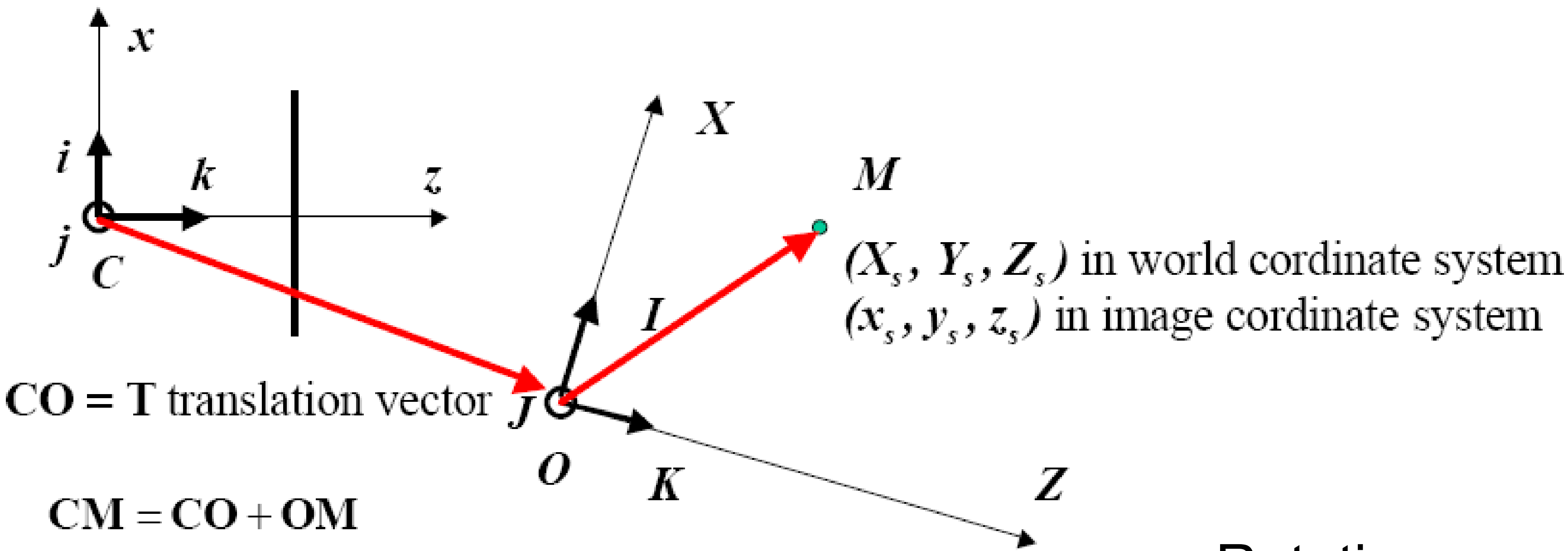
$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_x \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\boxed{K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}$$ calibration matrix $$\boxed{\mathrm{P} = \mathrm{K}[\mathrm{I} \mid 0]}$$

World Coordinates and Camera Coordinates



Camera Frame to World Frame



$$\mathbf{CM} = \mathbf{CO} + \mathbf{OM}$$

$$x_S\mathbf{i} + y_S\mathbf{j} + z_S\mathbf{k} = T_x\mathbf{i} + T_y\mathbf{j} + T_z\mathbf{k} + X_S\mathbf{I} + Y_S\mathbf{J} + Z_S\mathbf{K}$$

$$x_S = T_x + X_S\mathbf{I.i} + Y_S\mathbf{J.i} + Z_S\mathbf{K.i}$$

$$\begin{bmatrix} x_S \\ y_S \\ z_S \end{bmatrix} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \mathbf{I.i} & \mathbf{J.i} & \mathbf{K.i} \\ \mathbf{I.j} & \mathbf{J.j} & \mathbf{K.j} \\ \mathbf{I.k} & \mathbf{J.k} & \mathbf{K.k} \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \end{bmatrix}$$

Rotation matrix

Definition: 3D-Rotation

Linear Algebra

Definition: a matrix R is a rotation matrix if and only if it is a orthogonal matrix with determinant +1

Orthogonal Matrix: a square matrix with real entries whose columns and rows are orthogonal vectors with length 1.

That means: $RR^T = I$

Or: $R^{-1} = R^T$

3D Transformations - Rotation

- Euler Angles for Rotation $R=R_zR_yR_x$:

rotation by ψ about the z axis: $R_z = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$

rotation by θ about the y axis: $R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$

rotation by ϕ about the x axis: $R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$

Sequence not standardized! Many different conventions!

Homogeneous Coordinates

$$\begin{bmatrix} x_S \\ y_S \\ z_S \end{bmatrix} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \mathbf{I.i} & \mathbf{J.i} & \mathbf{K.i} \\ \mathbf{I.j} & \mathbf{J.j} & \mathbf{K.j} \\ \mathbf{I.k} & \mathbf{J.k} & \mathbf{K.k} \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \end{bmatrix}$$

$$\begin{bmatrix} x_S \\ y_S \\ z_S \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{I.i} & \mathbf{J.i} & \mathbf{K.i} & T_x \\ \mathbf{I.j} & \mathbf{J.j} & \mathbf{K.j} & T_y \\ \mathbf{I.k} & \mathbf{J.k} & \mathbf{K.k} & T_z \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_S \\ y_S \\ z_S \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}_3^{\mathbf{T}} & 1 \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

Let $\mathbf{C} = -\mathbf{R}^t\,\mathbf{T}$

$$\begin{bmatrix} x_S \\ y_S \\ z_S \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\,\widetilde{\mathbf{C}} \\ \mathbf{0}_3^{\mathbf{T}} & 1 \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

Putting Everything Together

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \mathbf{K}\left[\mathbf{I}_3 \quad | \quad \mathbf{0}_3\right] \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_S \\ y_S \\ z_S \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\widetilde{\mathbf{C}} \\ \mathbf{0}_3^{\mathbf{T}} & 1 \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

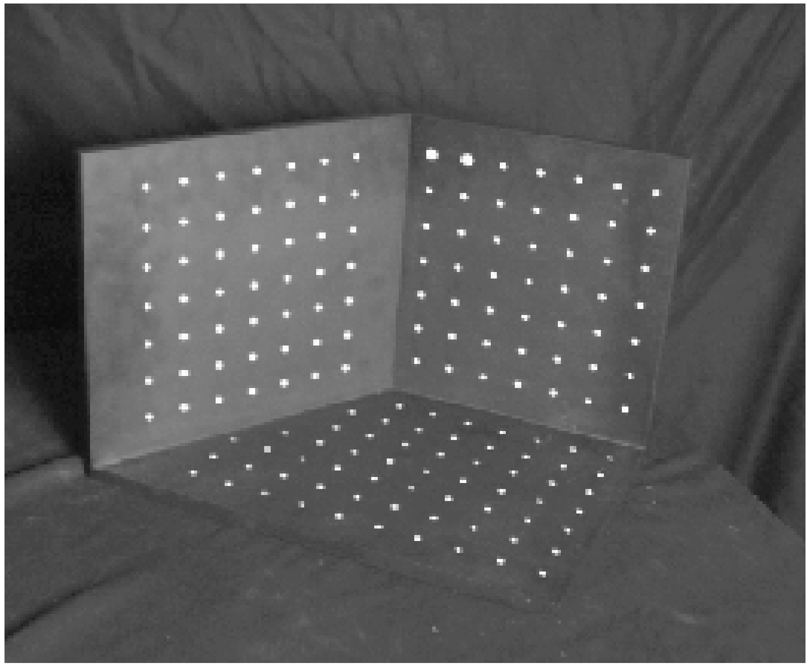
$$\Rightarrow \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \mathbf{K}\left[\mathbf{I}_3 \quad | \quad \mathbf{0}_3\right] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\widetilde{\mathbf{C}} \\ \mathbf{0}_3^{\mathbf{T}} & 1 \end{bmatrix} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

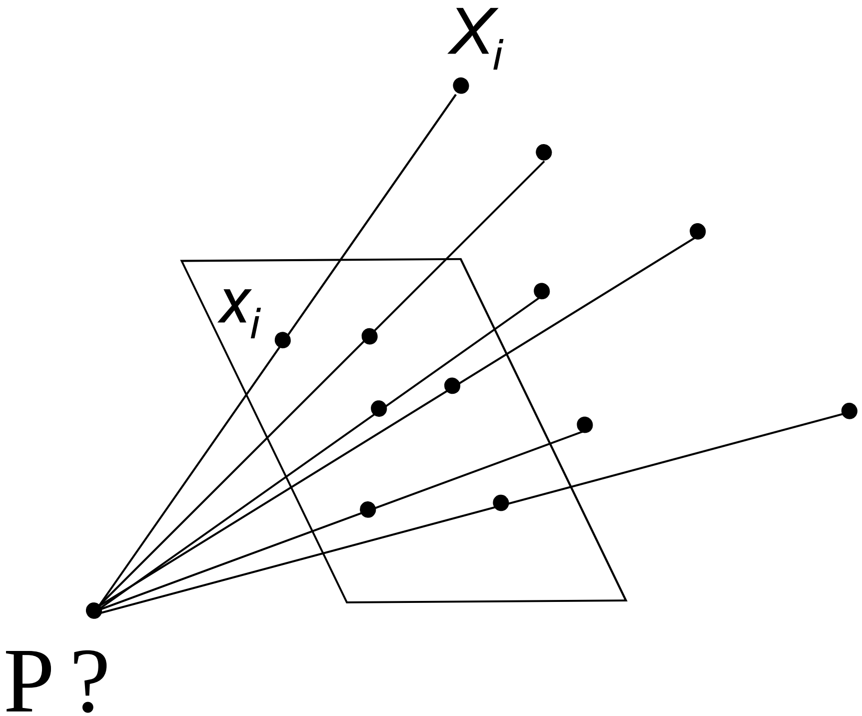
$$\Rightarrow \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \mathbf{P} \begin{bmatrix} X_S \\ Y_S \\ Z_S \\ 1 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

Camera calibration

- Given n points with known 3D coordinates X_i and known image projections x_i, estimate the camera parameters





Calibration

1. Estimate matrix **P** using scene points and their images

2. Estimate the intrinsic and extrinsic parameters

$$\mathbf{P} = \mathbf{K}\,\mathbf{R}\left[\mathbf{I}_3 \quad | \quad -\tilde{\mathbf{C}}\right]$$

Left 3x3 sub-matrix is the product of an upper triangular matrix and an orthogonal matrix.

Computing the Matrix P

- Use corresponding image and scene points
  - 3D points $\mathbf{X_i}$ in world coordinate system
  - Images $\mathbf{x_i}$ of $\mathbf{X_i}$ in image
- Write $\mathbf{x_i} = \mathbf{P}\,\mathbf{X_i}$ for all i

Computing the Matrix P

- $\mathbf{x_i} = \mathbf{P}\,\mathbf{X_i}$ involves homogeneous coordinates, thus $\mathbf{x_i}$ and $\mathbf{P}\,\mathbf{X_i}$ just have to be proportional: $\mathbf{x}_\mathrm{i} \times \mathbf{P}\,\mathbf{X}_\mathrm{i} = 0$
- Let $\mathbf{p}_1{}^\mathrm{T}$, $\mathbf{p}_2{}^\mathrm{T}$, $\mathbf{p}_3{}^\mathrm{T}$ be the 3 row vectors of $\mathbf{P}$

$$\mathbf{P}\,\mathbf{X}_\mathrm{i} = \begin{bmatrix} \mathbf{p}_1^\mathbf{T}\mathbf{X}_\mathrm{i} \\ \mathbf{p}_2^\mathbf{T}\mathbf{X}_\mathrm{i} \\ \mathbf{p}_3^\mathbf{T}\mathbf{X}_\mathrm{i} \end{bmatrix} \qquad \mathbf{x}_\mathrm{i} \times \mathbf{P}\,\mathbf{X}_\mathrm{i} = \begin{bmatrix} v'_\mathrm{i}\,\mathbf{p}_3^\mathbf{T}\mathbf{X}_\mathrm{i} - w'_\mathrm{i}\,\mathbf{p}_2^\mathbf{T}\mathbf{X}_\mathrm{i} \\ w'_\mathrm{i}\,\mathbf{p}_1^\mathbf{T}\mathbf{X}_\mathrm{i} - u'_\mathrm{i}\,\mathbf{p}_3^\mathbf{T}\mathbf{X}_\mathrm{i} \\ u'_\mathrm{i}\,\mathbf{p}_2^\mathbf{T}\mathbf{X}_\mathrm{i} - v'_\mathrm{i}\,\mathbf{p}_1^\mathbf{T}\mathbf{X}_\mathrm{i} \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \mathbf{0}_4^\mathbf{T} & -w'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} & v'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} \\ w'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} & \mathbf{0}_4^\mathbf{T} & -u'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} \\ -v'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} & u'_\mathrm{i}\,\mathbf{X}_\mathrm{i}^\mathbf{T} & \mathbf{0}_4^\mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = 0 \qquad \begin{bmatrix} \mathrm{p}_1 \\ \mathrm{p}_2 \\ \mathrm{p}_3 \end{bmatrix} \text{is a } 12 \times 1 \text{ vector}$$

Computing the Matrix P

- Third row can be obtained from sum of u'_i times first row - v'_i times second row

Rank 2

$$\begin{bmatrix} \mathbf{0}_4^T & -w'_i \mathbf{X}_i^T & v'_i \mathbf{X}_i^T \\ w'_i \mathbf{X}_i^T & \mathbf{0}_4^T & -u'_i \mathbf{X}_i^T \\ -v'_i \mathbf{X}_i^T & u'_i \mathbf{X}_i^T & \mathbf{0}_4^T \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = 0$$

- So we get 2 independent equations in 11 unknowns (ignoring scale)
- With 6 point correspondences, we get enough equations to compute matrix **P**

$$\mathbf{A}\,\mathbf{p} = 0$$

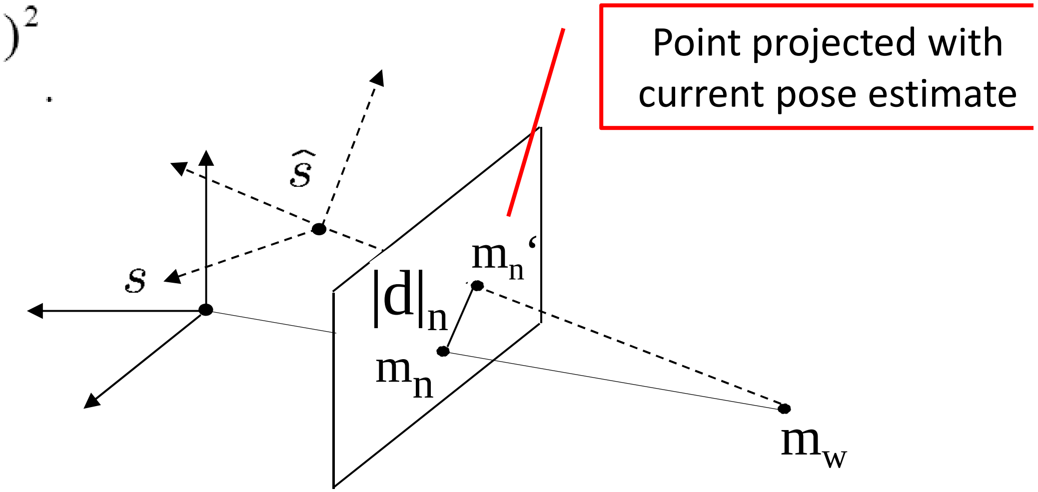
Computing the Matrix P

- Linear system $\mathbf{A}\,\mathbf{p} = 0$
- When possible, have at least 5 times as many equations as unknowns (28 points)
- Minimize $\| \mathbf{A}\,\mathbf{p} \|$ with the constraint $\| \mathbf{p} \| = 1$
  - P is the unit singular vector of **A** corresponding to the smallest singular value (the last column of **V**, where $\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^{\mathbf{T}}$ is the SVD of **A**)

Computing the Matrix P

Further Improvement

Use as initialization for nonlinear minimization of $\sum_{\mathbf{i}} d(\mathbf{x}_{\mathbf{i}}, \mathbf{P}\mathbf{X}_{\mathbf{i}})^2$



Most popular non-linear minimization algorithm is the **Levenberg-Marquart minimization**

LM is more robust to local minima than e.g. the **Gauss–Newton algorithm** and the method of **gradient descent**)

Iterative minimization

General nonlinear problem formulation for n measurements:

$$\hat{x} = \arg\min_{x} \sum_{i=1}^{n} r^{(i)}$$

Estimate

Parameter vector

Residual/reprojection error

Concrete problem of camera <u>pose estimation</u> from 2D/3D correspondences

- Parameters to estimate: camera pose (rotation and translation)

$$(R,t) = \arg\min_{\delta(R,t)} = \sum_{i} dist(proj(m_{wi};(R,t)), m_{ni})$$

Least squares (LS) estimation

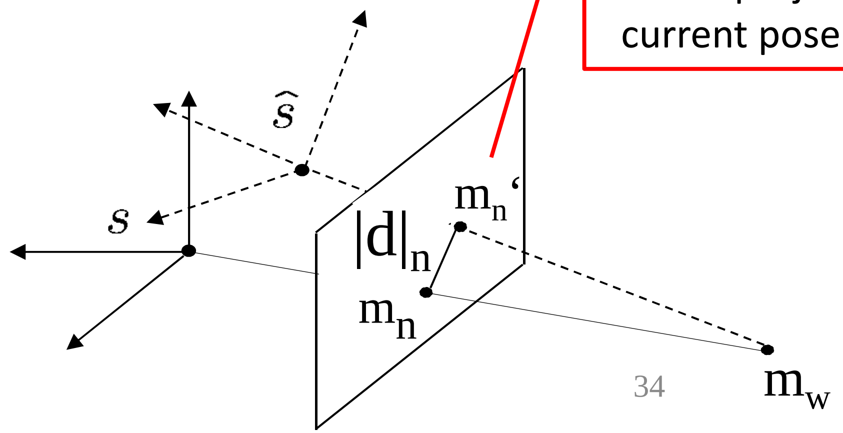
Typical problem formulation (error in normalized image space):

$$\hat{\mathrm{s}} = \arg\min_{\mathrm{s}} \sum_{i=1}^{n} r_n^{(i)LS} = \arg\min_{\mathrm{s}} \sum_{i=1}^{n} \| \, d_n(\mathrm{m}_n^{(i)}, \mathrm{m}_w^{(i)}, \mathrm{s}) \|^2$$

All measurements are incorporated with equal weight

If information about the quality of measurements is given,
how to incorporate this?

Point projected with current pose estimate



Robust estimation (M-estimators)

Minimize the sum of a function of residuals:

$$\hat{\mathrm{s}} = \arg\min_{\mathrm{s}} \sum_{i=1}^{n} \rho(r^{(i)})$$

Function appears as weight

How could a simple robust error function be constructed?

$$\rho(r) = \begin{cases} |r|, & \text{if } |r| < r_{max} \\ r_{max}, & \text{if } |r| \geq r_{max} \end{cases}$$

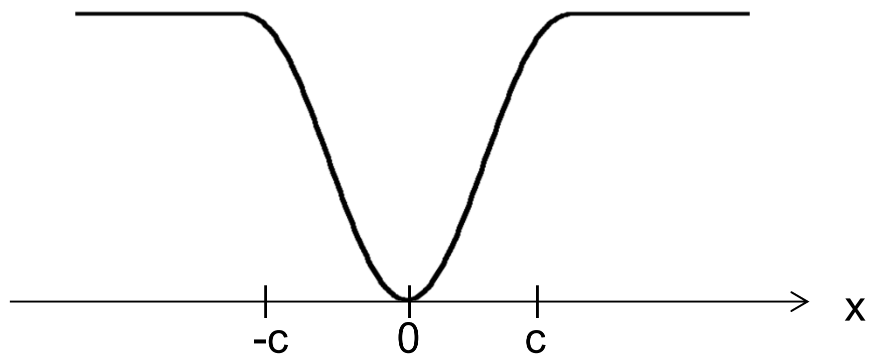
Assumptions:

- Only few outliers
- Initialization close to solution

Typical M-estimators: TUKEY, Cauchy, Huber, …

Der Tukey- Estimator

$$\rho_{Tuk}(x) = \begin{cases} \dfrac{c^2}{6}\left[1 - \left(1 - \left(\dfrac{x}{c}\right)^2\right)^3\right] & |x| \leq c \\ \dfrac{c^2}{6} & |x| > c \end{cases}$$

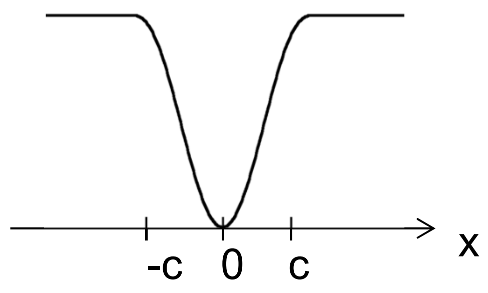


The Tukey Estimator

The threshold c depends of the standard deviation ov the error function, e.g. $c=4\sigma$

Correspondences with $|x| \leq c$ influences the minimization results.

Correspondences with $|x|>c$ are outliers and have no influence.



Outline

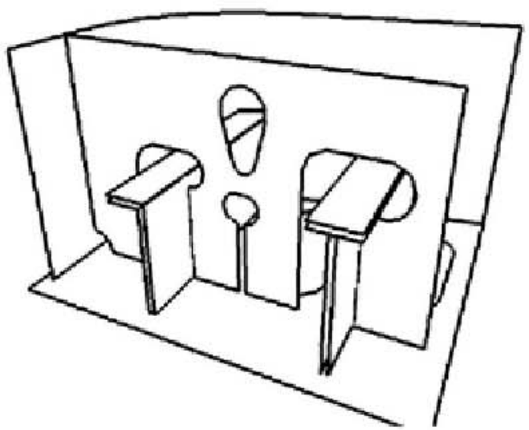
- Part I: Camera model and pose computation
- Part II: 3D countour tracking (RAPID)

Contour-based tracking

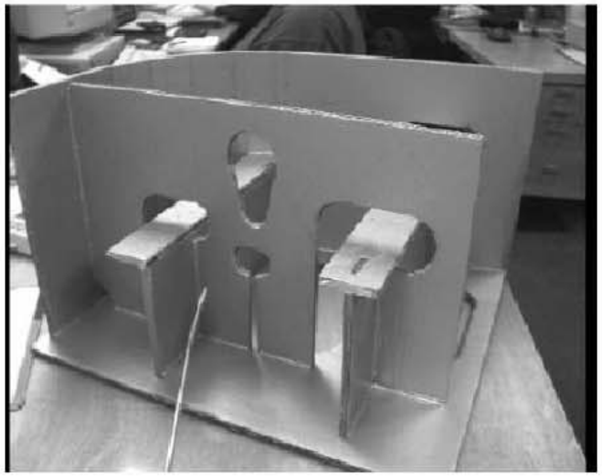
Definition of contour-based tracking

A procedure to estimate the pose of an object by using contours, selected from a full wire-frame (CAD) model

Model contours = boundaries along the object surface, that we expect to see in the image, from a given viewpoint



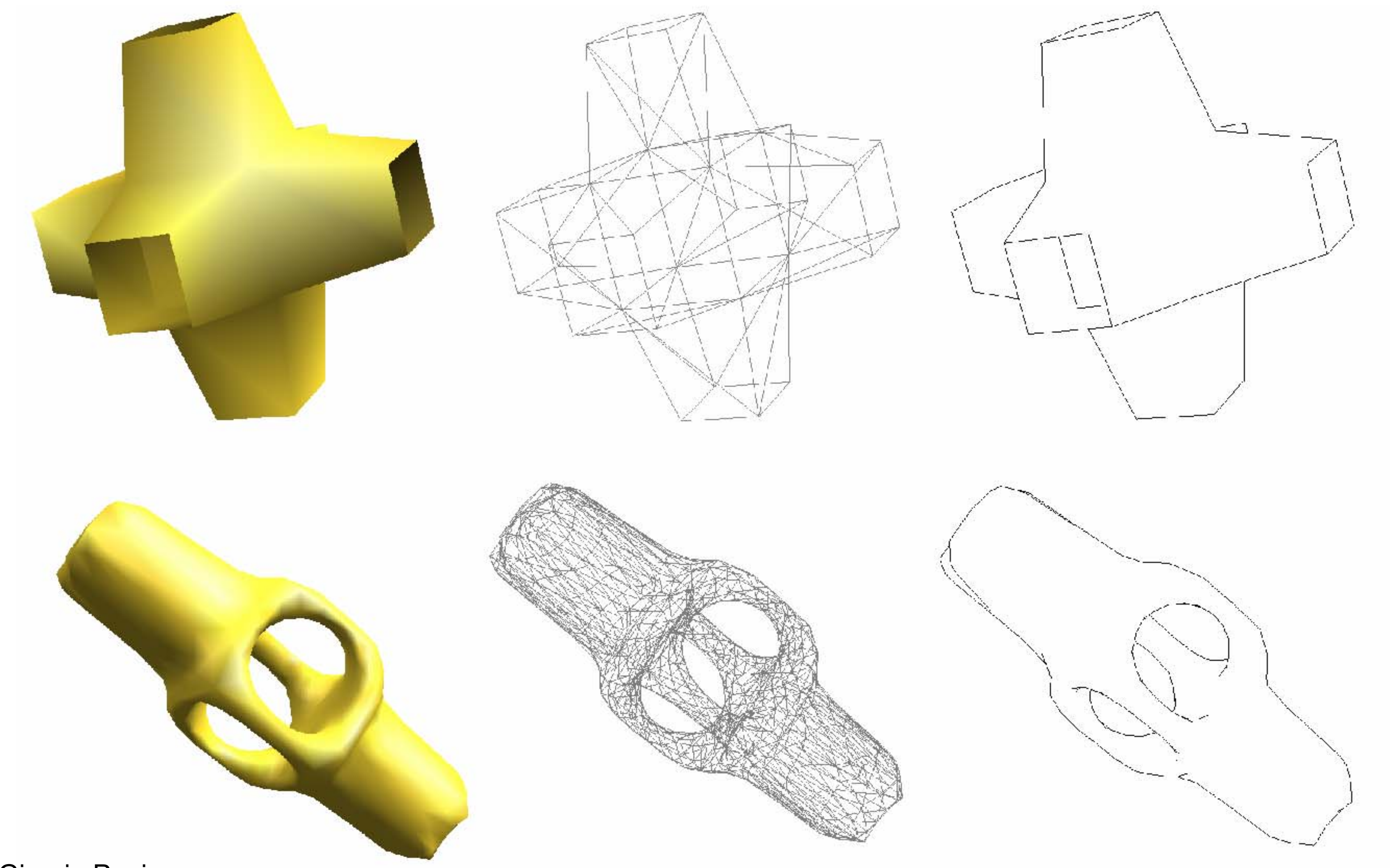
Model contours



Camera view

Examples of model contours

Rendered view | CAD model | Model contours

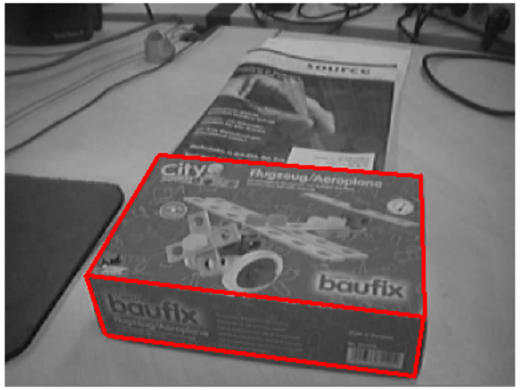


Giorgio Panin

Contours vs. keypoints

Advantages

1. Edges are usually well-defined (sharp transitions), for many different poses, light conditions, etc.



Advantage: more robustness and precision w.r.t. pose and light

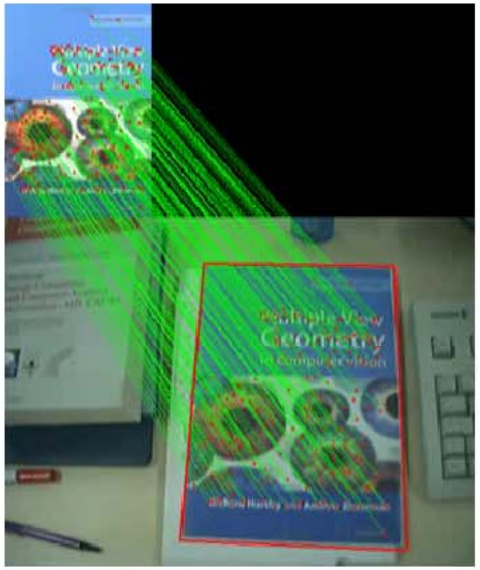
2. We can obtain good tracking with low computational time, because contours are rather easy to detect and to track (while point features require complex selection-description-matching)

Advantage: faster detection and tracking (real-time guaranteed)

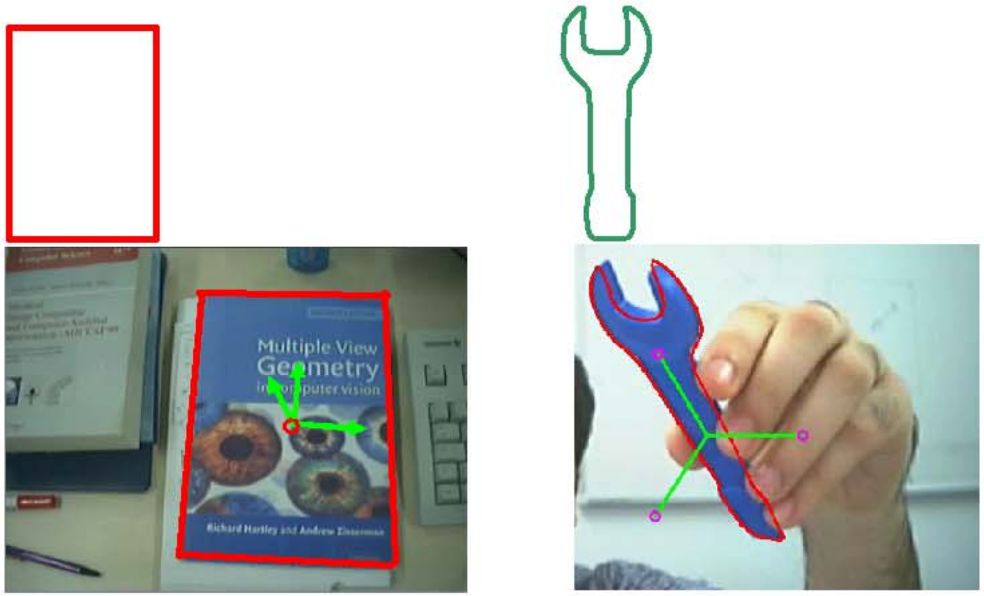
Contour vs. Point-based Tracking

Advantages

3. Contours depend only on the object shape (geometry), while feature points require a distinctive surface appearance (not good for uniform colors or textures)



Keypoints



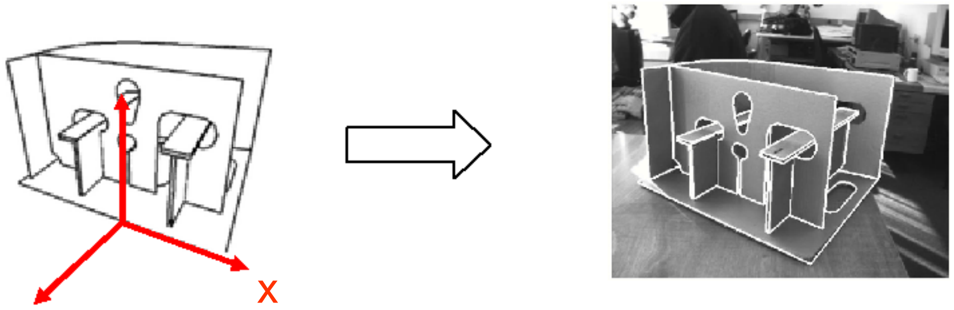
Contours

Advantage: we can track a large class of objects or environment items, that have no feature points on the surface

Contour vs. Point-based Tracking

Disadvantages

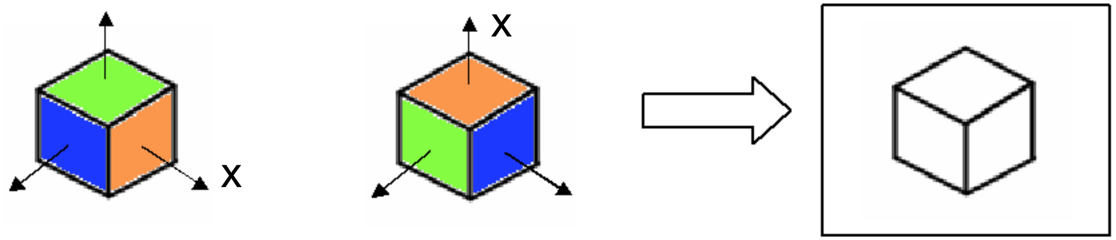
1. It works only if the 3D object has a sufficiently asymmetric shape, in order to identify the pose without ambiguity



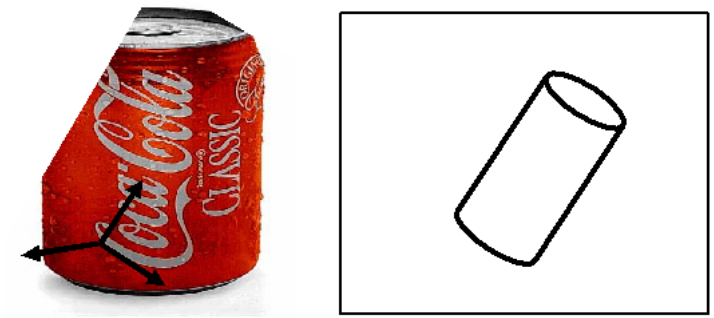
OK

(only 1 solution)

Cube: ambiguous, because of symmetries (multiple solutions)



Revolving surface: cannot estimate axial rotation (infinite solutions)



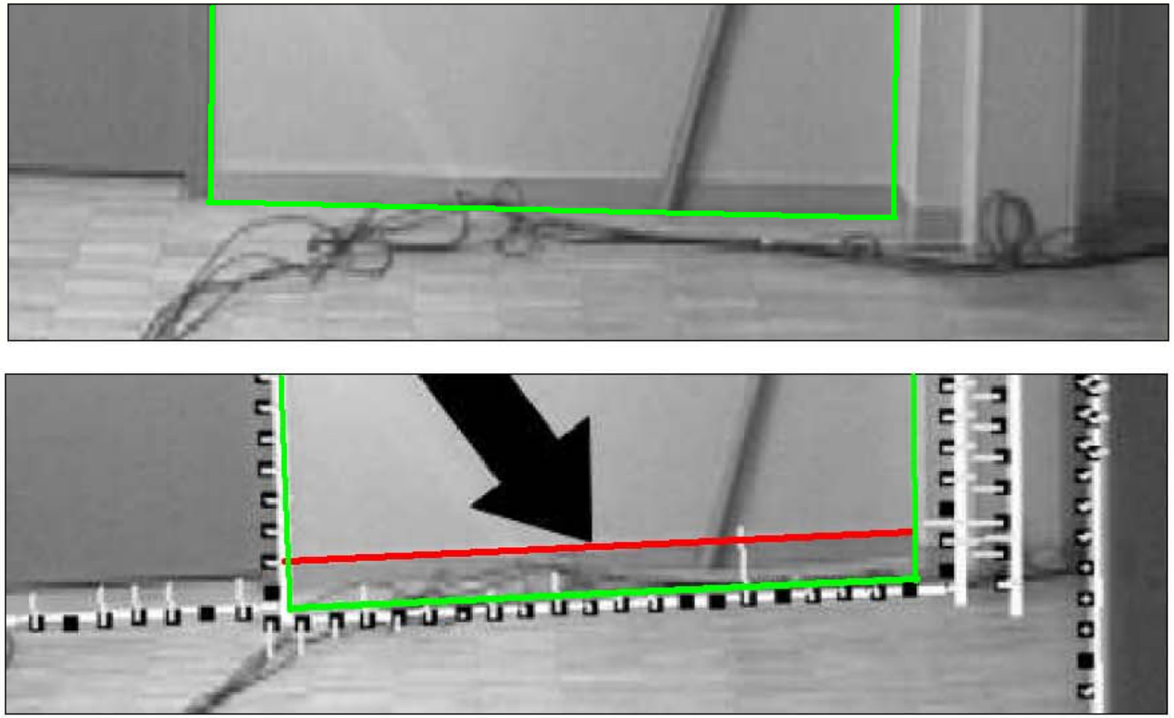
Contour vs. Point-based Tracking

Disadvantages

2. Edges are less distinctive features than key-points

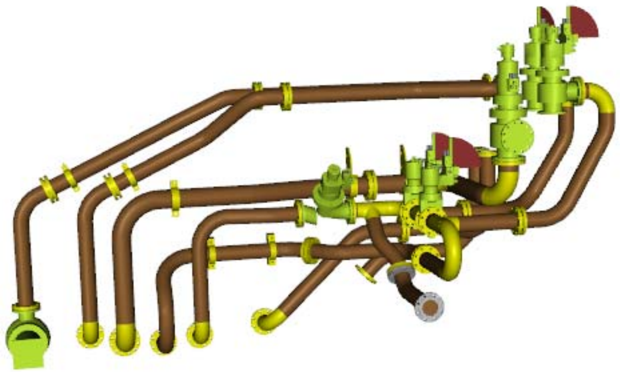
They have a weak identity

Less robust matching: we may easily follow the wrong edge!

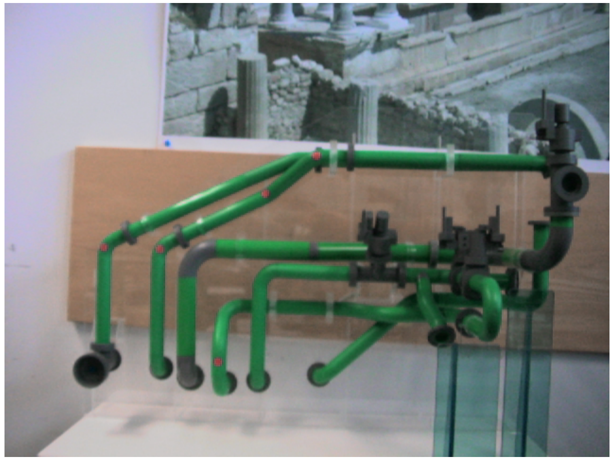


Example: two edges parallel and close together – which one is from the model?

Goal

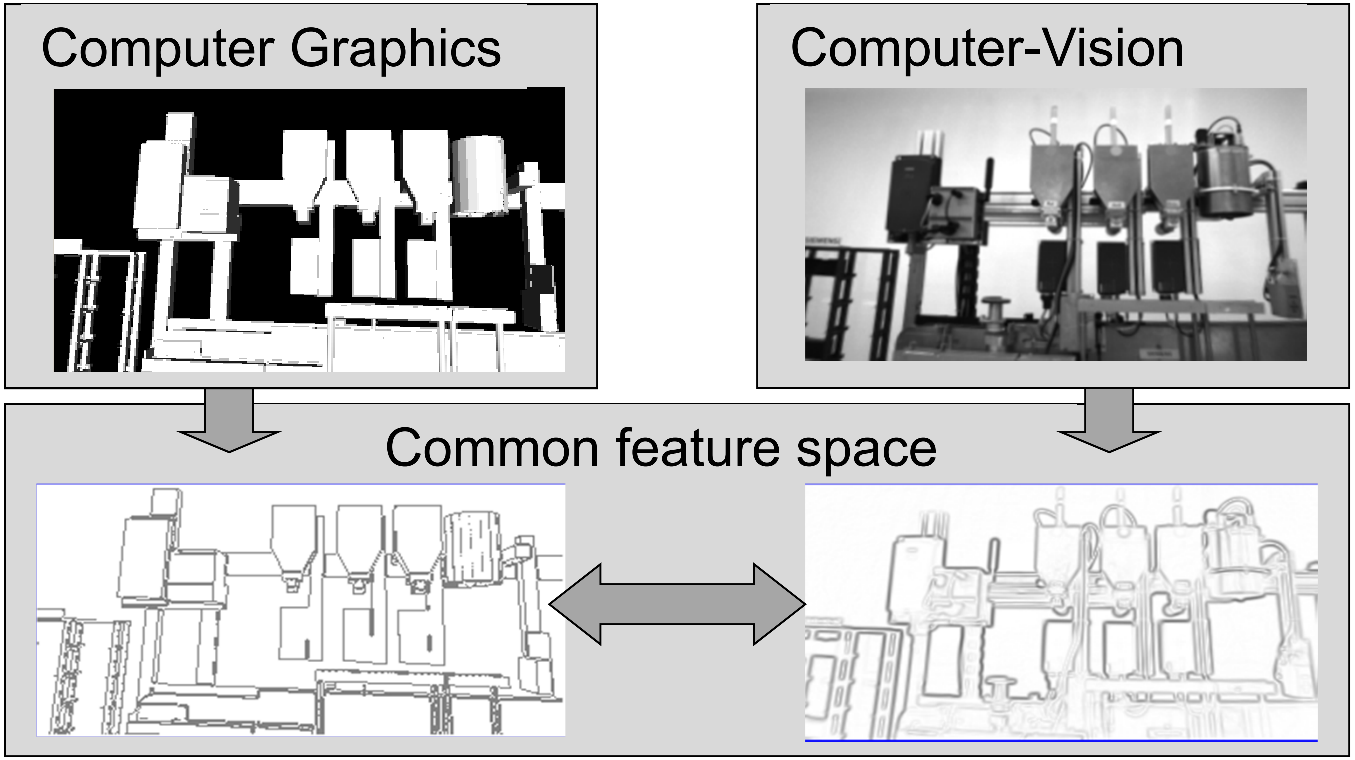


?



Computer-graphics (CG) and computer vision CV

Goal: to work in a and feature space common to CV and CG representation of the object

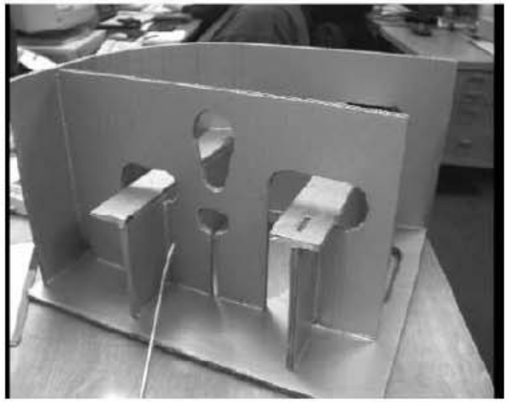


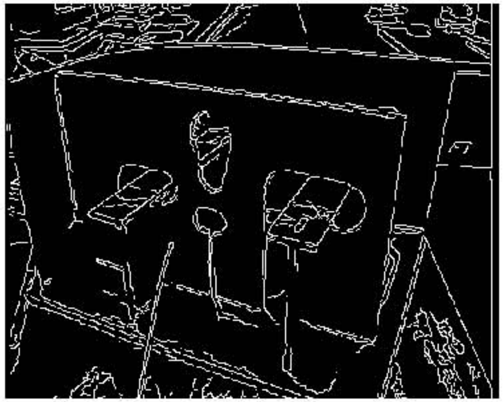
The RAPiD algorithm

Edge-based tracking = the „RAPiD“ algorithm [Harris, 1992]

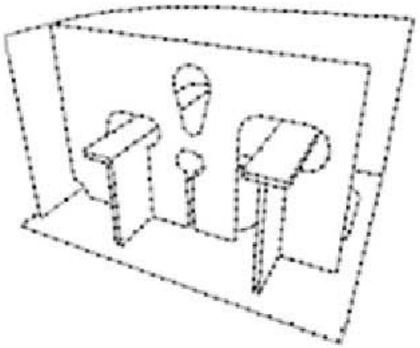
RAPiD = Real-time Attitude and Position Determination

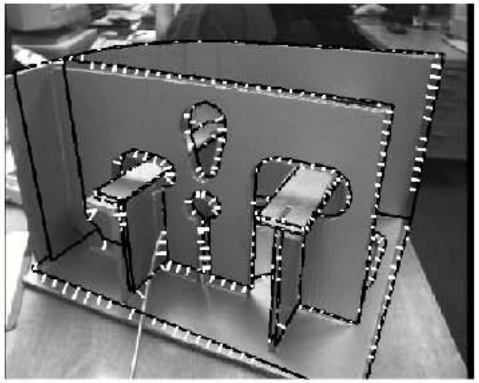
A. Pre-processing: compute the edge map



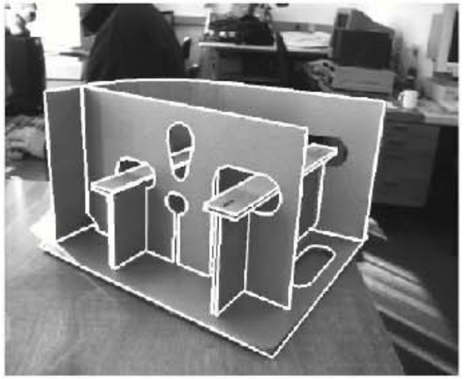


B. Iterate two steps: 1. data association, 2. pose update





Data association



Pose update

A. Computing the edge map: Contour features

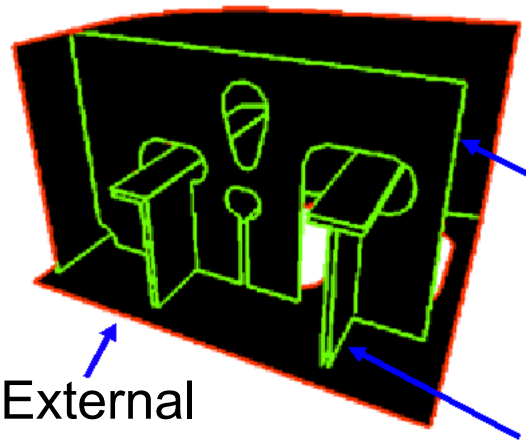
Existing type of edges – which ones are good to track?

Crease edges: have a sharp angle (e.g. polyhedral faces)

Boundary edges: define the border of a planar part of the surface

Silhouette edges: define the horizon of a round part of the surface

Texture edges: separate different texture parts (reflectance properties)

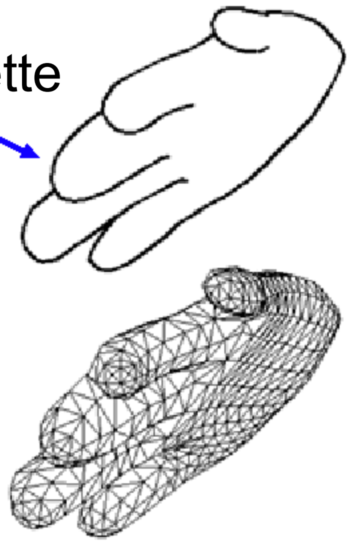


Internal boundary

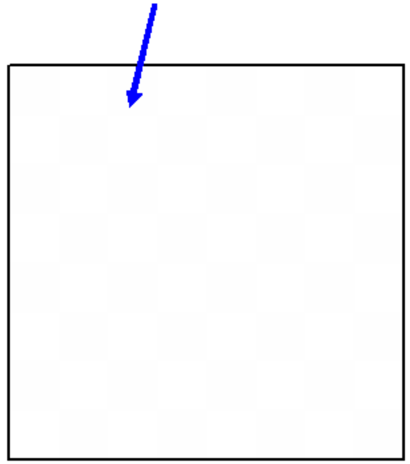
External boundary

Crease

Silhouette



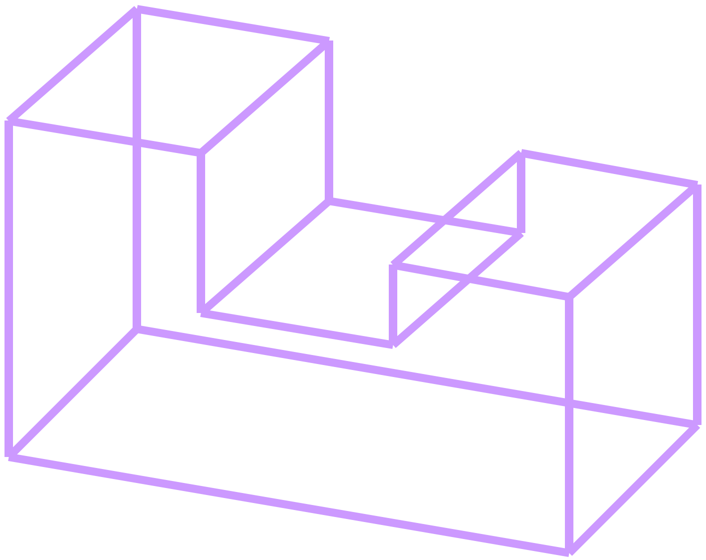
Texture edge

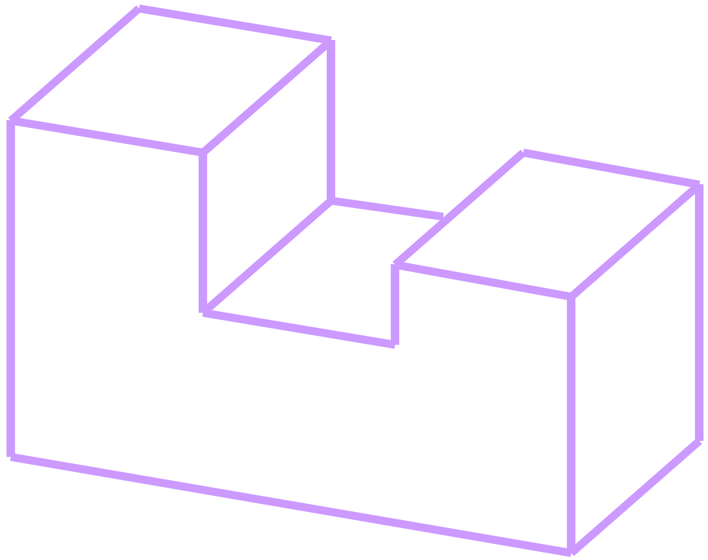


Giorgio Panin

Contour model: visible contours

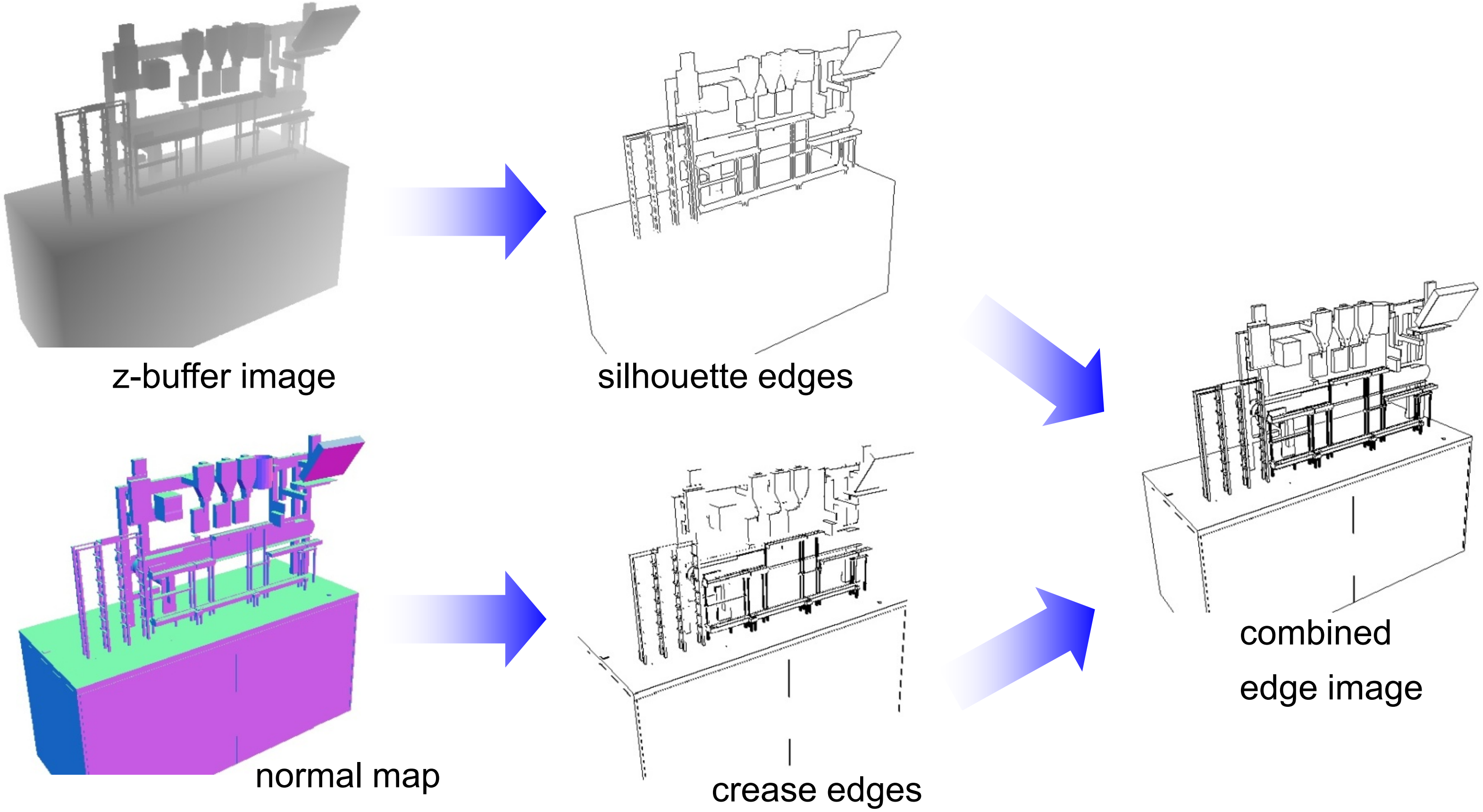
Use computer graphics techniques





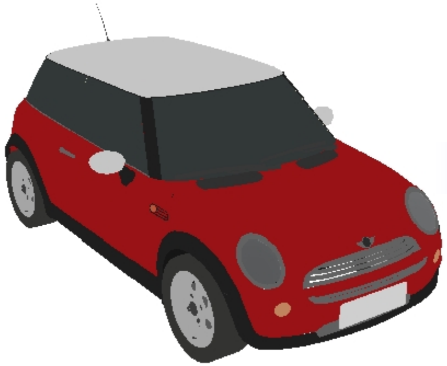
Generating a 3D contour model

The edges can be computed in once at initialization but also in real-time during tracking using the graphics card

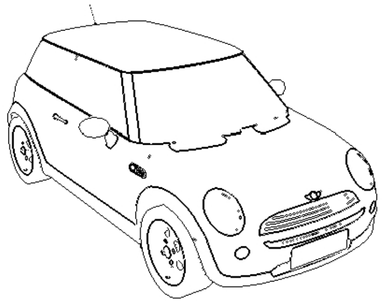


Using computer graphics

Texture / Material Edges:

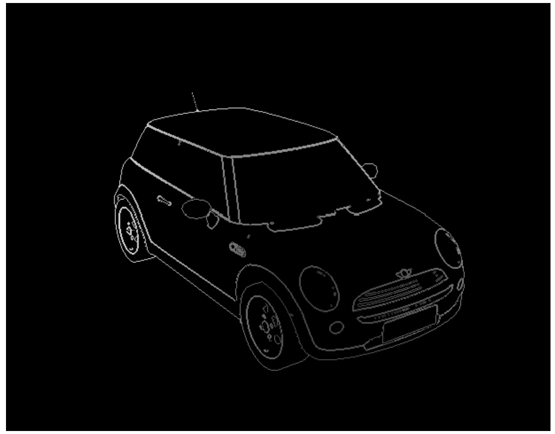


render into texture with ambient lighting



edge image

store z-value in alpha-channel

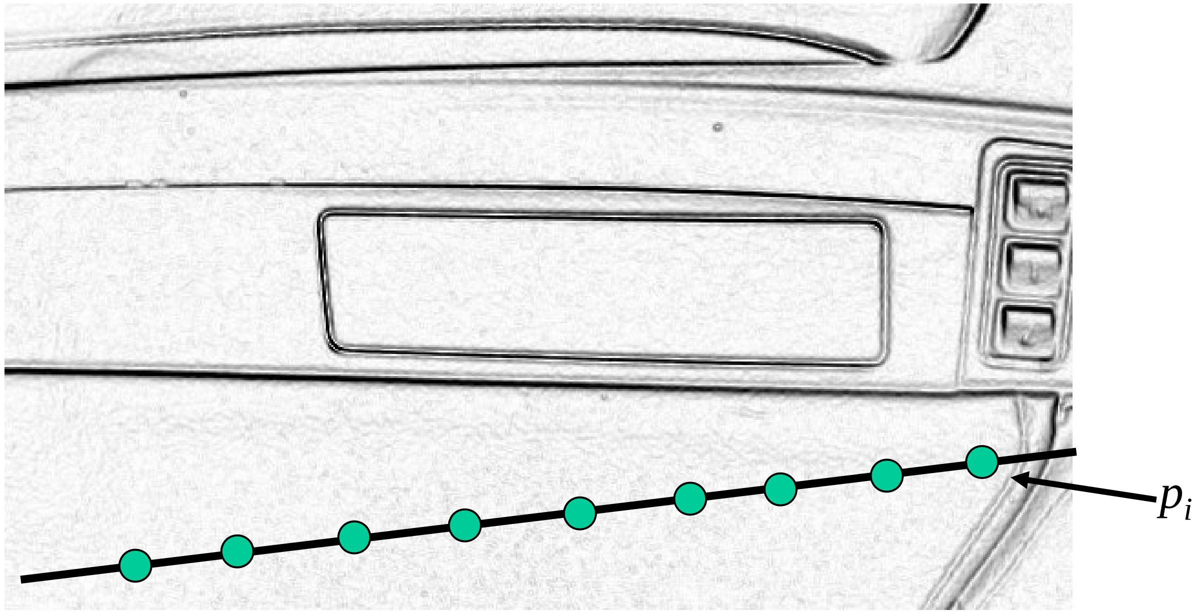


2½D edge image

B. Association and pose computation

The 3D edge model is projected onto the image, using the last pose.

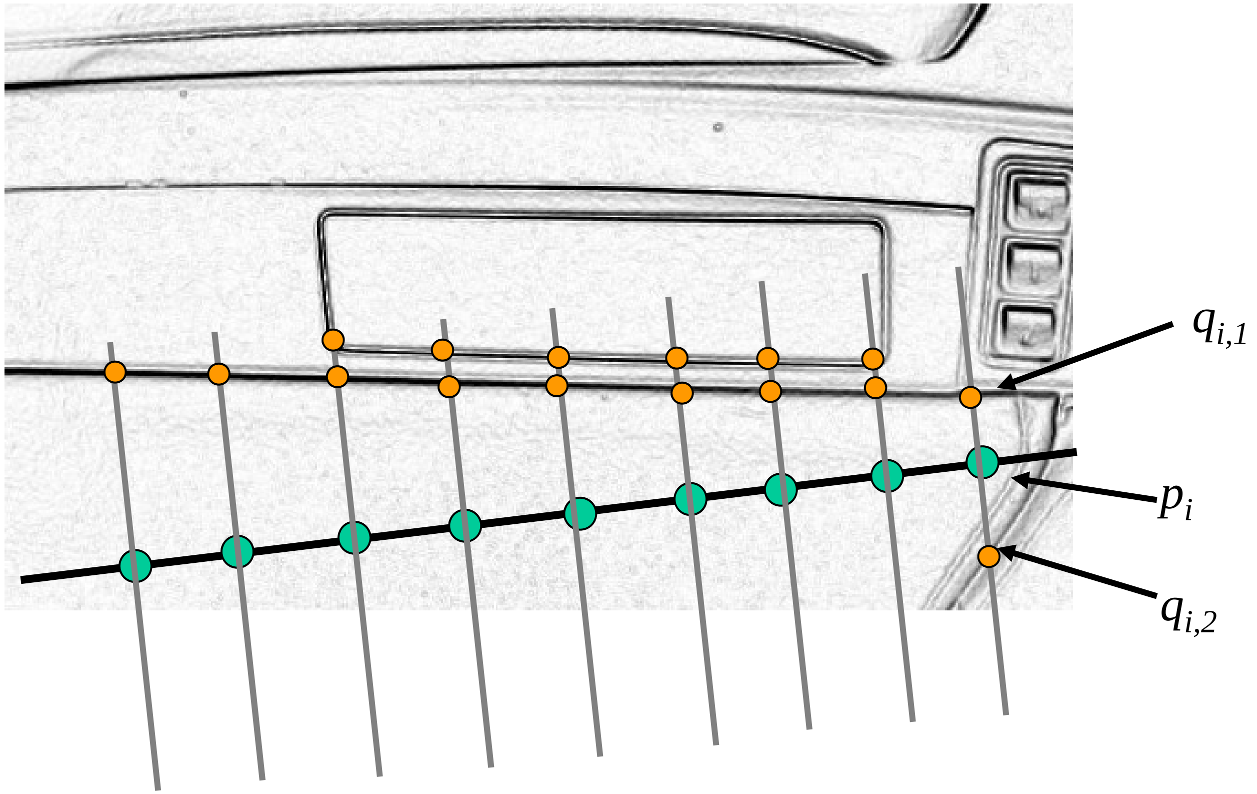
Control points p_i are defined at regular disatnce for each edge E of the model.



B. Association and pose computation

An orthogonal search segment is define at each p_i

Candiates $q_{i,j}$ are defined along this segment.



B. Association and pose computation

Single hypothesis solution:

p_i the controll poin and q_i the detected point in the image

The following error function is minimized:

$$err = \sum_i \rho_{Tuk}(\Delta(p_i, q_i))$$

With ρ_{Tuk} the Tukey Estimator Function.

The distance function Δ is defined as:

$$\Delta(p_i, q_i) = \left|(q_i - p_i) \cdot n_i\right|$$

With n_i die normal vector of the projected line.

B. Association and pose computation

Problem with the single hypothese solution

The „first“ gradient in the image is not necessary the right one.
Ambiguity for complex objects and cluttered background.

-> object edges are stuck on the wrong image edges.

Idea: Multiple hypotheses

- Take into account several hypothesis
- For a given controll point pi several q_{ij} are considered.
- The choice of the correct candidate happens implicitely during the minimization

(Vacchetti & Lepetit’04)

B. Association and pose computation

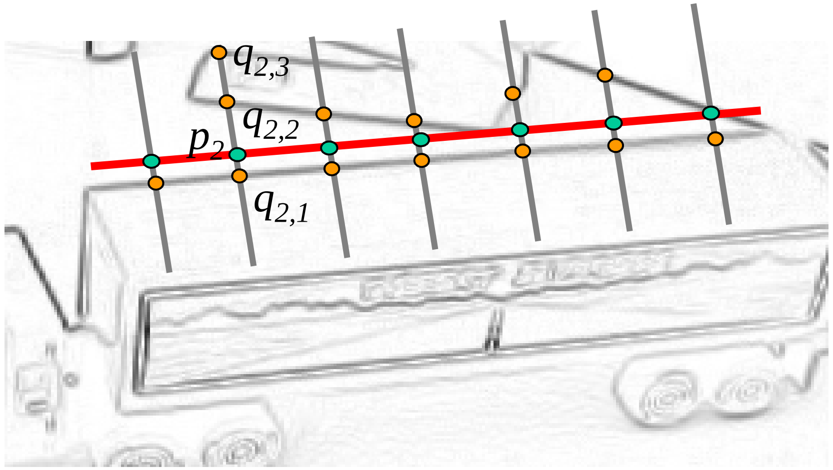
For multiple hypothsis, we have:

p_i the control points of the edge and $q_{i,j}$ seien multiple potential candidates.

The following error function is minimzed:

$$err = \sum_i \rho_{Tuk}(\min_j \Delta(p_i, q_{i,j}))$$

The Estimator-Function is computed for the hypothesis, which has the smallest distance to the projected point.

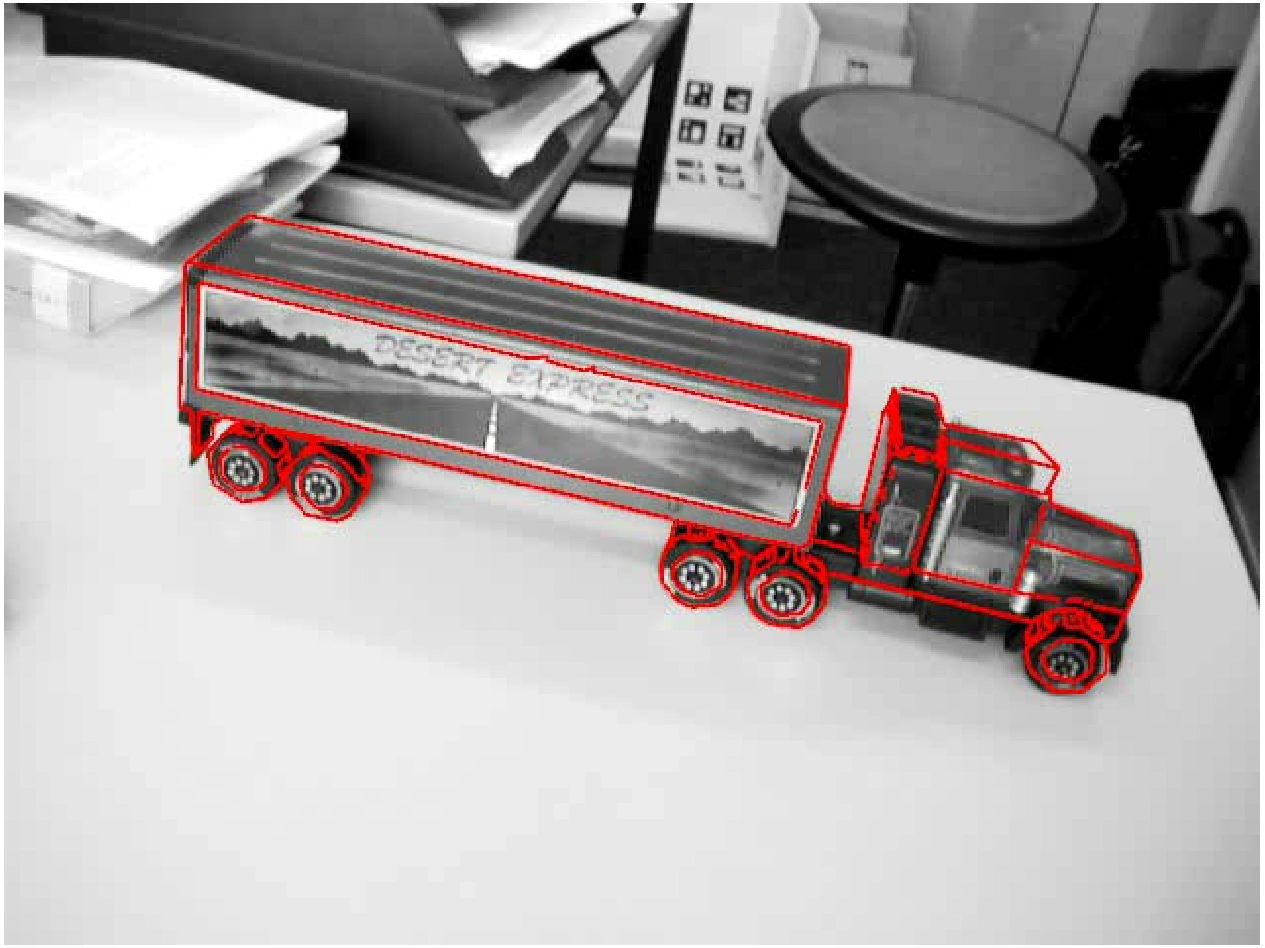


B. Association and pose computation

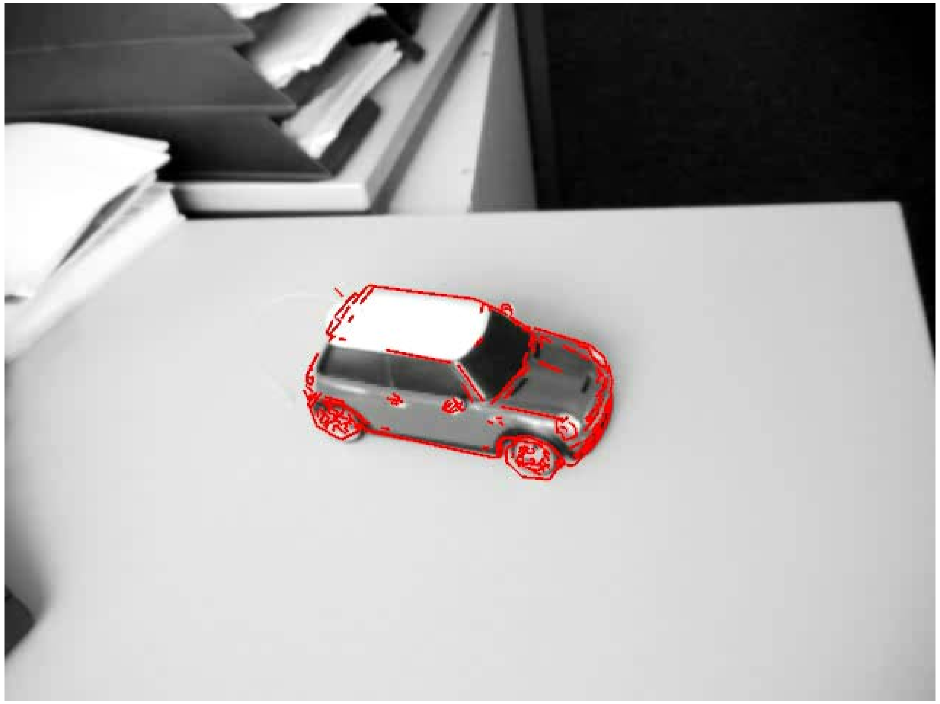
Summry multiple hypothesis:

- More computation expensive, but still real-time
- More robust!
- Ambiguity are solved through consideration of multiple candidates.

Results



Results



Thank you!