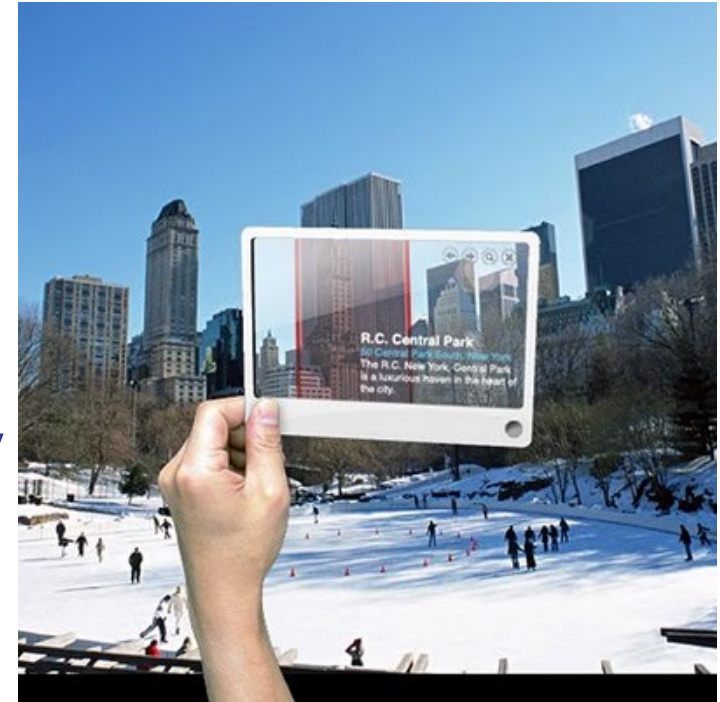


Real-time Real-Time Augmented reality

Éric Marchand



The application: augmented reality

Definition : [Azuma 97]

- Add virtual object in the video stream
- In real-time

Theoretical problem to be solved

- Find the camera position

Extensions

- post-production (we will see that later)



Application to augmented reality

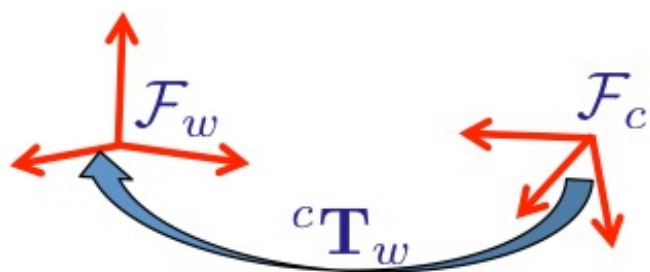
Augmented reality

- Coherent insertion of virtual objects within real images stream

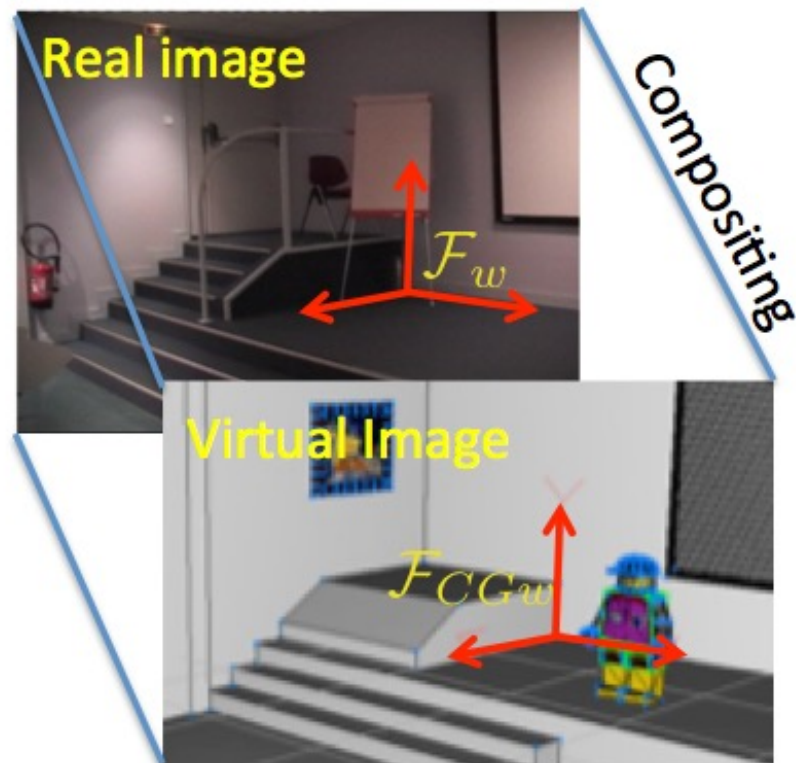
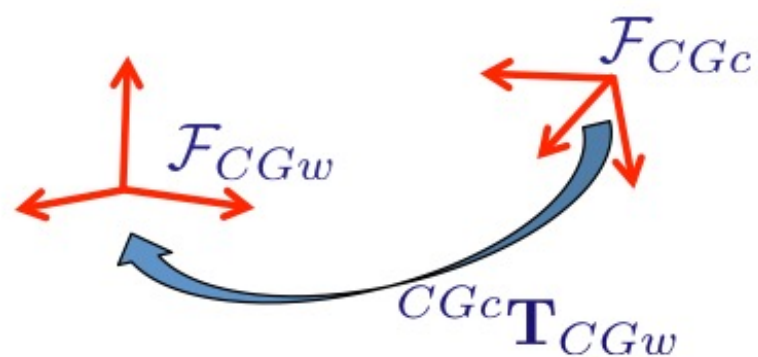
Augmented reality is handled as a 2D-3D registration issue

- Post production (match moving)
 - Full knowledge of the video sequence
 - Localization of the camera and structure of the scene
 - Bundle adjustment techniques (Realviz, 2D3)
- On-line augmented reality
 - Real-time requirements
 - No knowledge on the future
 - [Navab][Lepetit-Fua][Berger][Kutulakos],...





Align CG camera
with real camera



So...

Augmented reality is... viewpoint computation

Goal :

- Tracking the camera
 - in a sequential way (video streams, real time)
 - for getting stable and accurate augmentation results



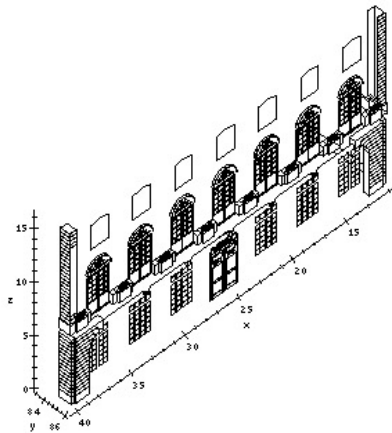
Model-based tracking

Pro

- fast and sequential (real-time)
- no drift

Cons

- the scene has to be partially known (markers, natural features)
 - tedious task of reconstructing or measuring scene features
- difficult to define a lot of features
 - jittering effect



Loria



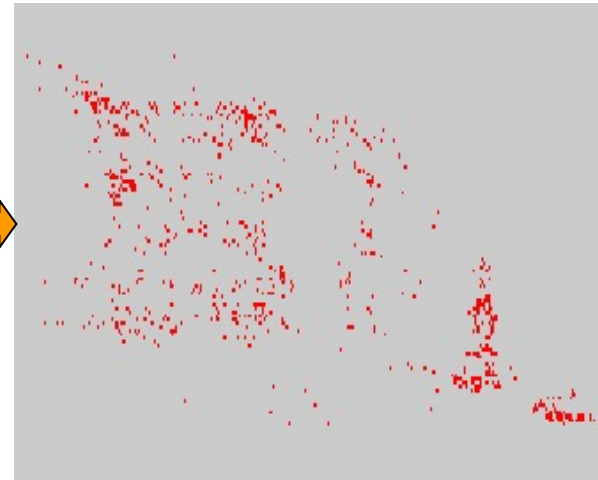
Motion computation

Pro

- do not require any model of the scene
- easy to track a lot of features + bundle adjustment
 - very accurate registration, negligible jitter

Cons

- slow and not sequential
- The world and the virtual coordinate system must be aligned manually



Tracking by matching

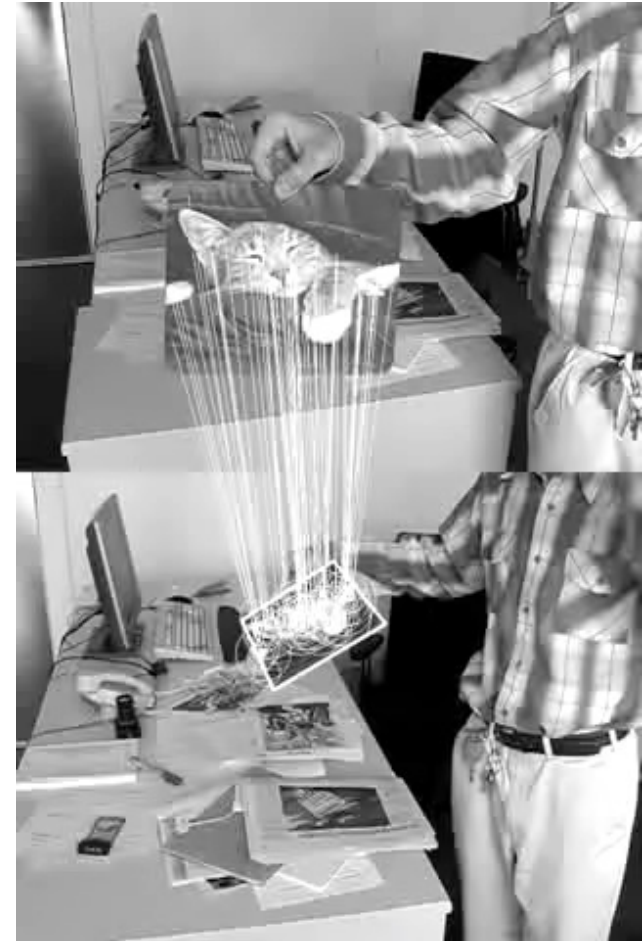
Establish the correspondences between primitives extracted from 2 images

Pro

- No tracking and then no real (re)initialization

Cons

- Viewpoint change (3D change)
- Image transformation (translation, rotation, scale change)
- Illumination change, Occlusion
- Low frame rate ?



What is not in the scope of this talk

Visualization devices

- HMD
- Video see through, optical see through



Augmented reality for post production



AR toolkit

Rendering

- Image synthesis
- Lighting considerations

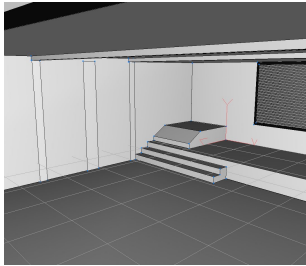


First,... Model-based tracking



Camera alignment for augmented reality

Virtual object



Real scene



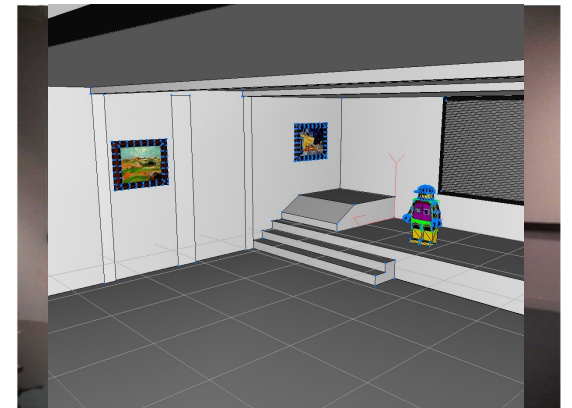
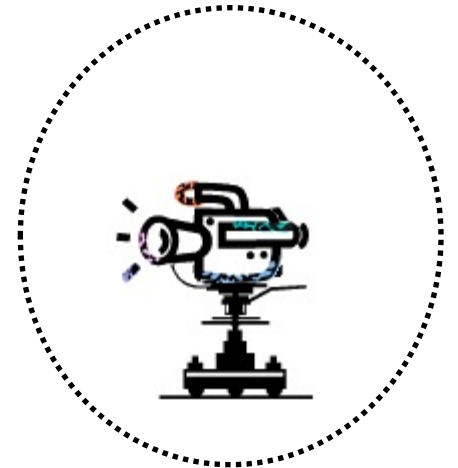
Graphic open GL



Align graphics camera
to real camera



Real camera



3D localization, pose estimation

Goal

- Determine 3D camera location wrt. an object using only one image of this object

Then

- With no a priori knowledge, localization is impossible
- Position of specific features have to be known in an object related frame

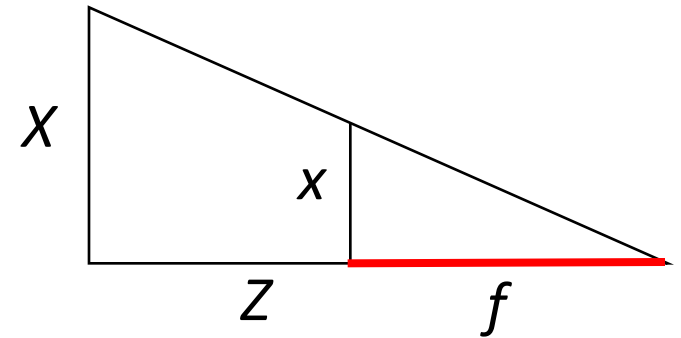
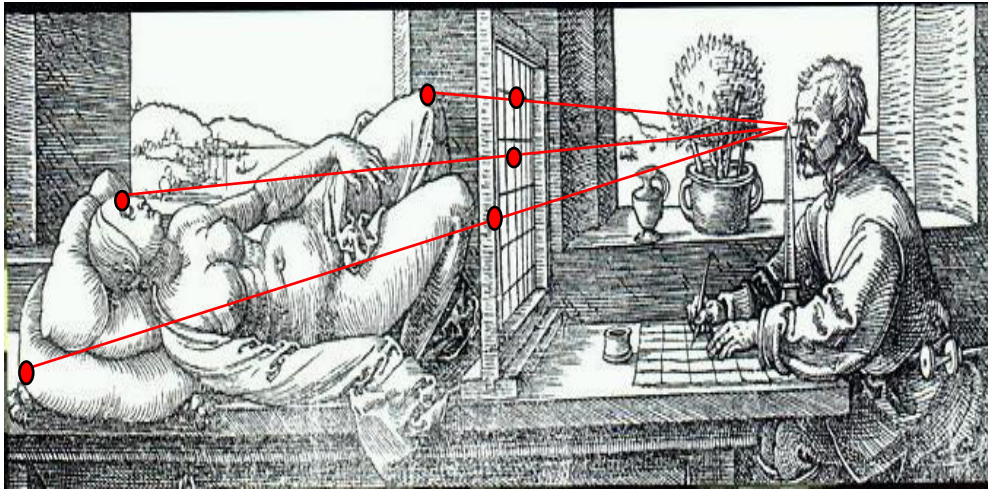
Approach: similar to calibration

- Simplification of the Toscanis-Faugeras method
- Dementhon-Davis method
- Non-linear minimization



Middle school geometry

Théorème de Thalès



Back to basic... projection



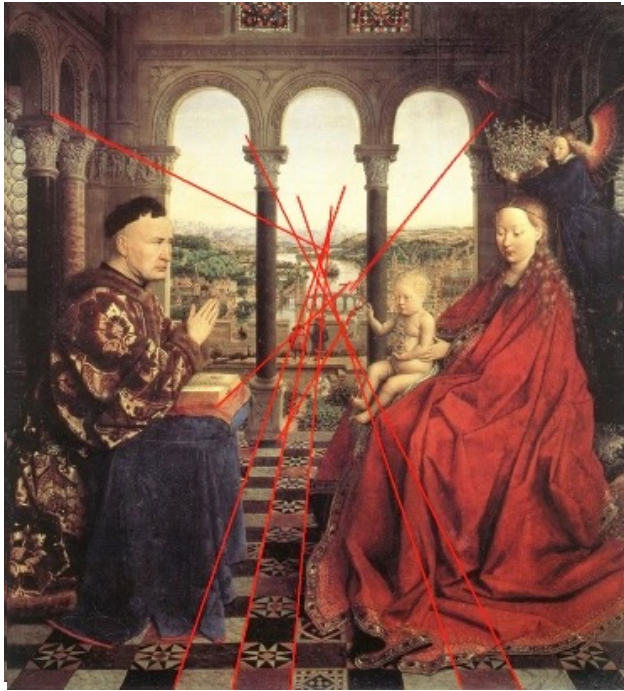
Perspective discovery



Ancient egypt, Bayeux tapestry (11e), Lorenzetti, les effets du bon gouvernement, 1340



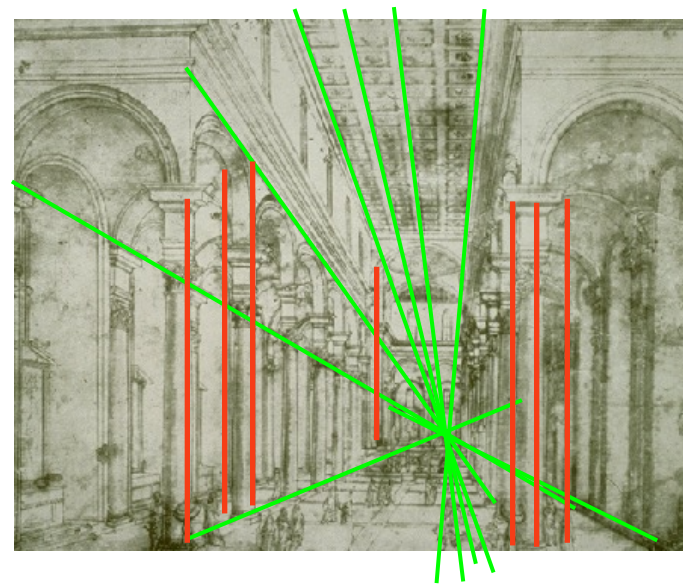
Perspective discovery



Van Eyck, 1435 (flandre)



Brunelleschi: Santa Maria della Fiore, 1435

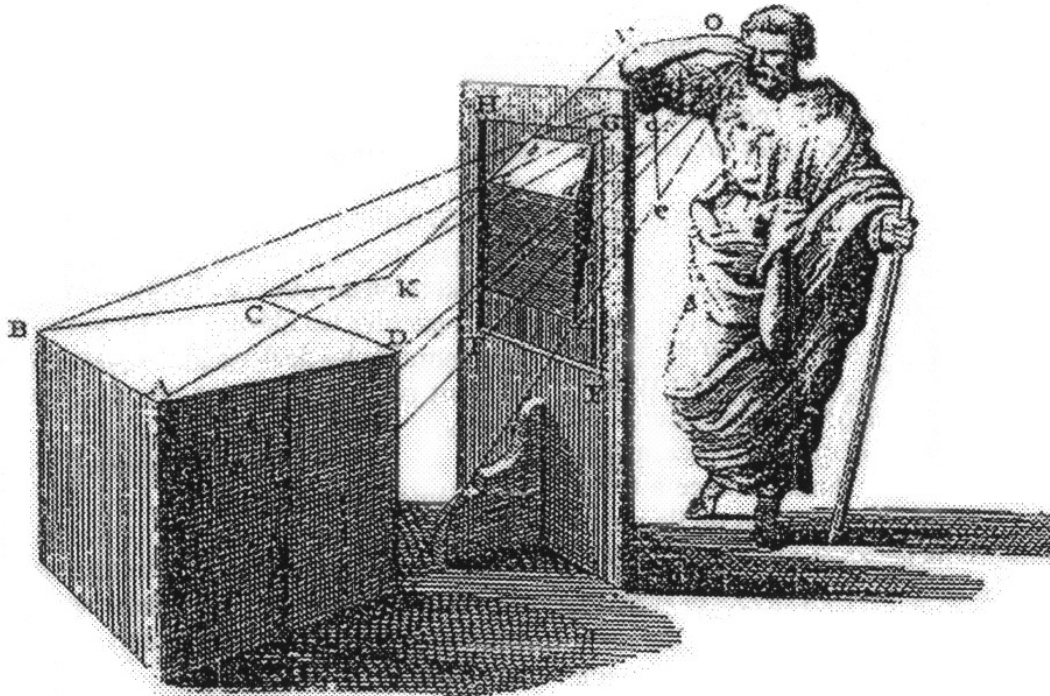


Brunelleschi, début 15ème

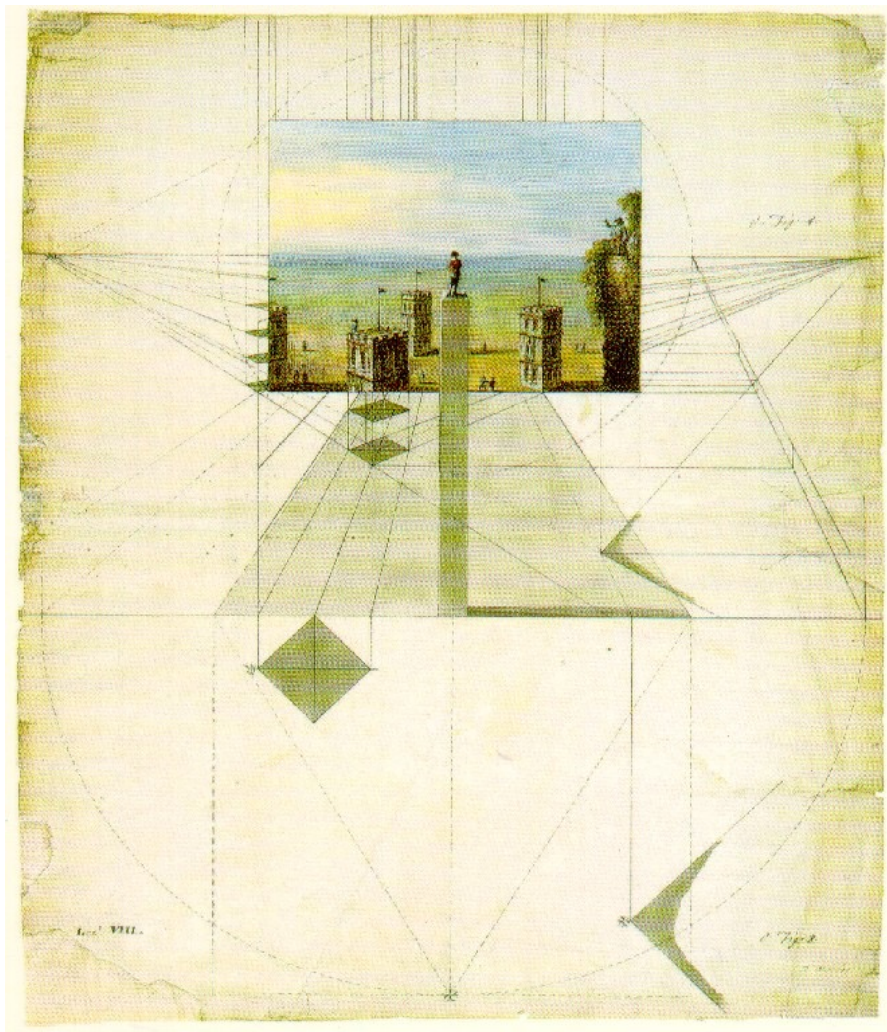


Alberti, *della pittura* 1435

As Brunelleschi made no written record of his perspective findings, it remained for Alberti to be the first to put the theory into writing, in his treatise on painting, *Della pittura* (1435). There, Alberti gave practical information for painters and advice on how to paint istoria or history paintings.



Perspective projection

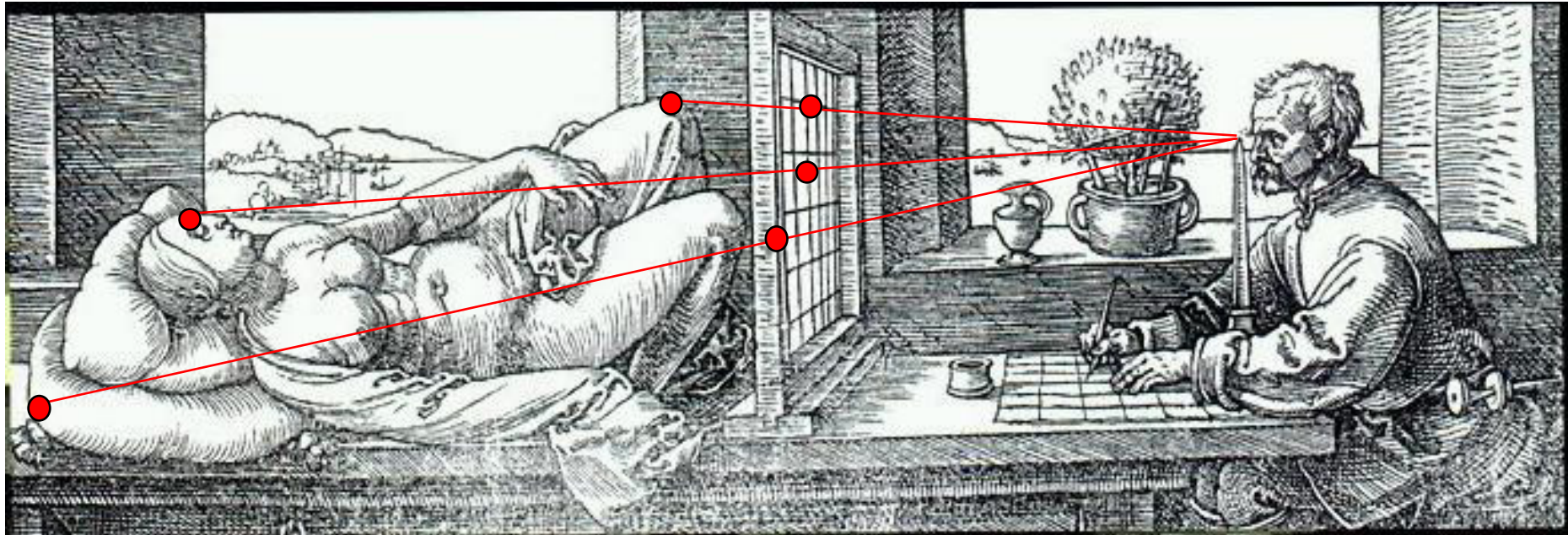


Dürer perspective device

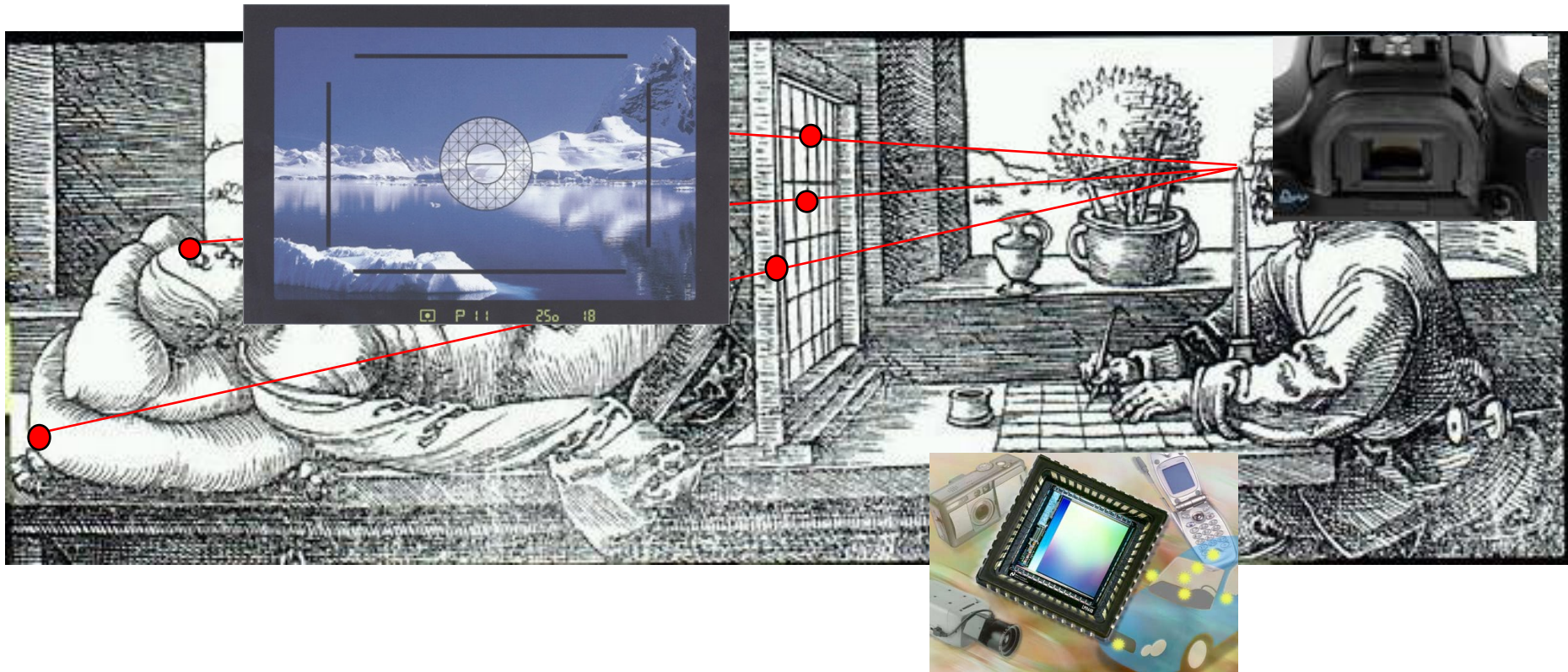
Albrecht Dürer: Artist Drawing a Nude with Perspective Device 1525



Dürer perspective device

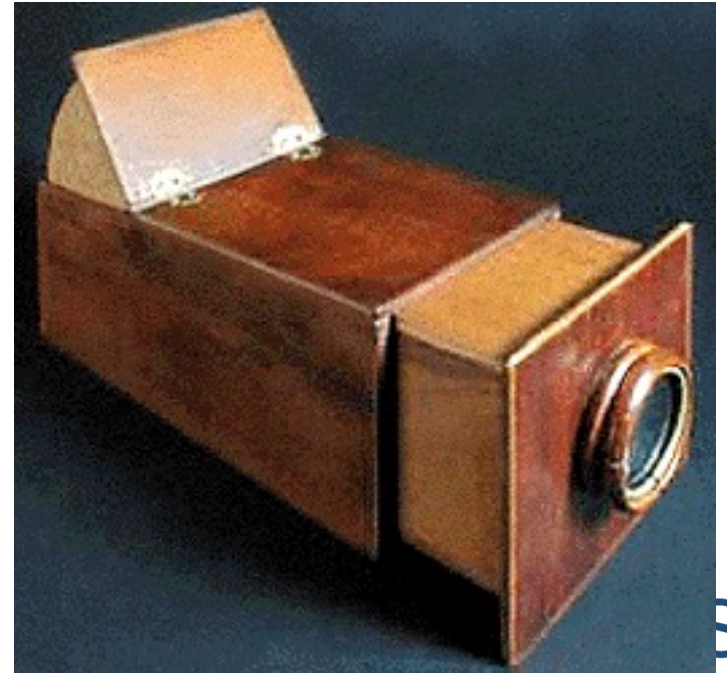
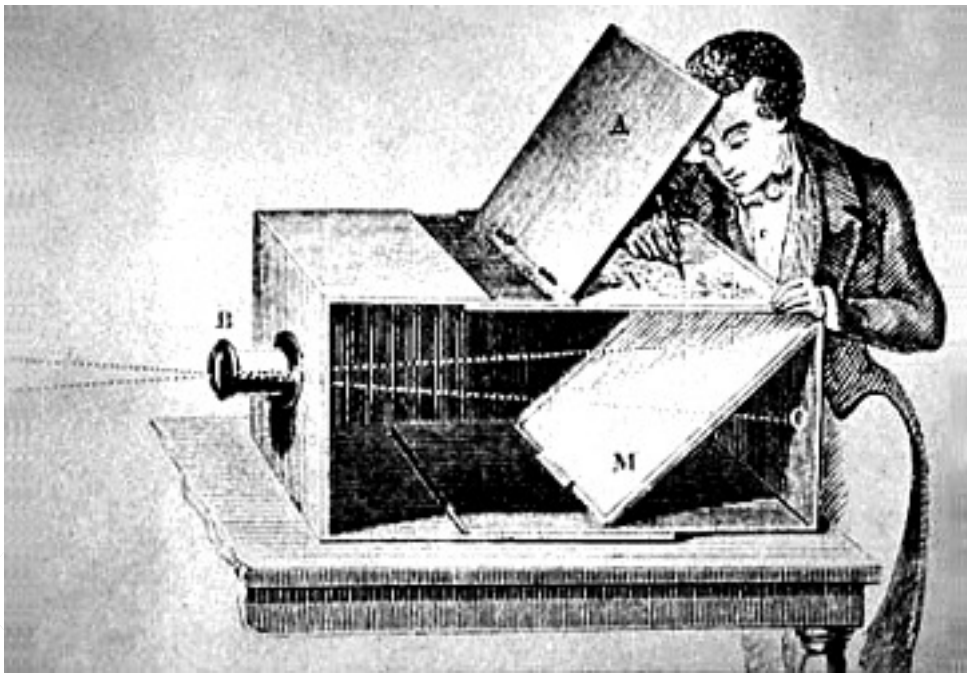


Dürer perspective device



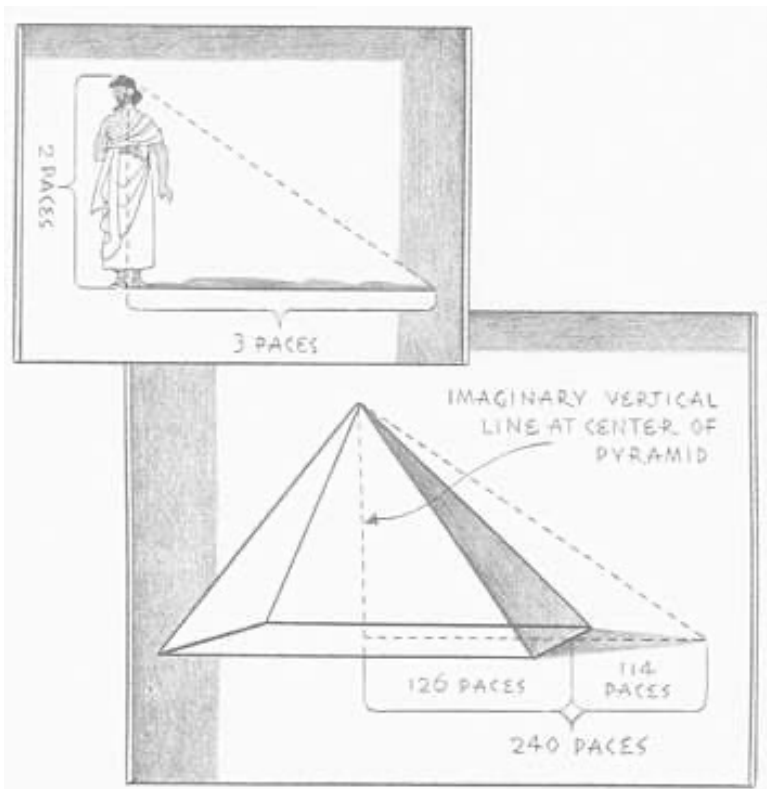
Camera obscura

- Invented in the sixteenth century, the camera obscura is made out of an arrangement of lenses and mirrors in a box that is darkened. The machine permits accuracy in a drawing, often of topographical detail. When looking through the lens of a camera obscura, the view presented is actually reflected through the mirrors onto the paper or cloth and allows the artist to draw by tracing the outline.



Perspective projection

Thales Theorem (625-546 Av JC)



Height of Pyramid (Imaginary Post) =

$$\text{SHADOW OF IMAGINARY POST} \times \frac{\text{Thales' Height}}{\text{Thales' Shadow}}$$

HEIGHT OF PYRAMID =

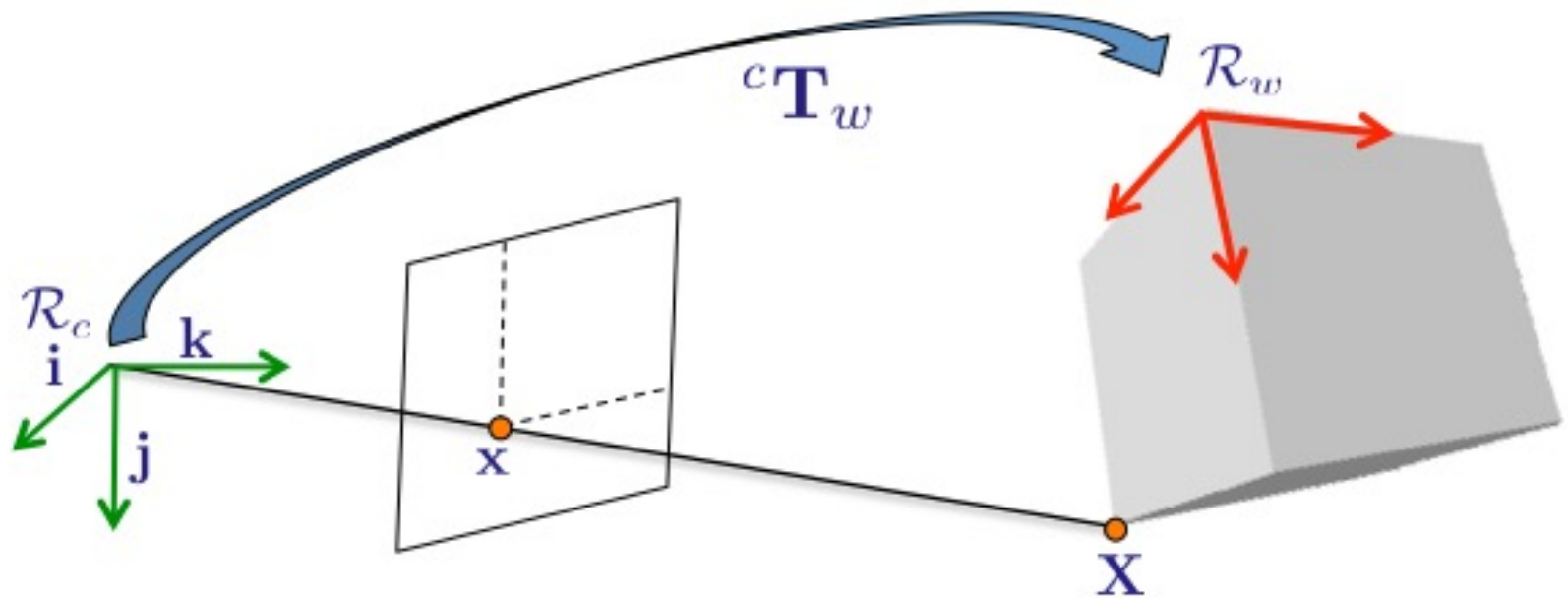
$$(\frac{1}{2} \text{ its Base} + \text{its Shadow}) \times \frac{\text{Thales' Height}}{\text{Thales' Shadow}}$$

$$H_p = (126 \text{ paces} + 114 \text{ paces}) \times \frac{2 \text{ paces}}{3 \text{ paces}}$$

$$H_p = 240 \times \frac{2}{3} = 160 \text{ paces!}$$



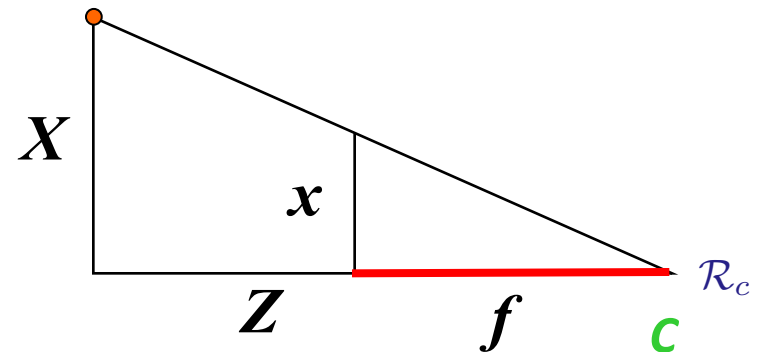
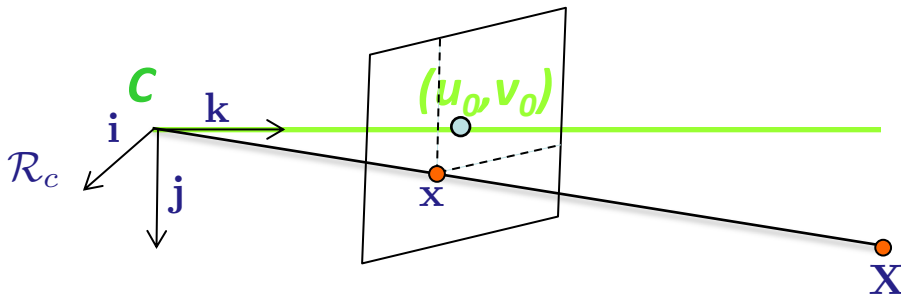
Perspective projection



Perspective projection

Definitions

- The origin of the camera frame \mathcal{R}_c is the center of projection C
- i (x) axes is parallel to image lines and j (y) axes is parallel to image columns
- Intersection of z axes with image plane is the principal point u_0, v_0
- focal distance $f = d(C, \pi)$



Perspective projection

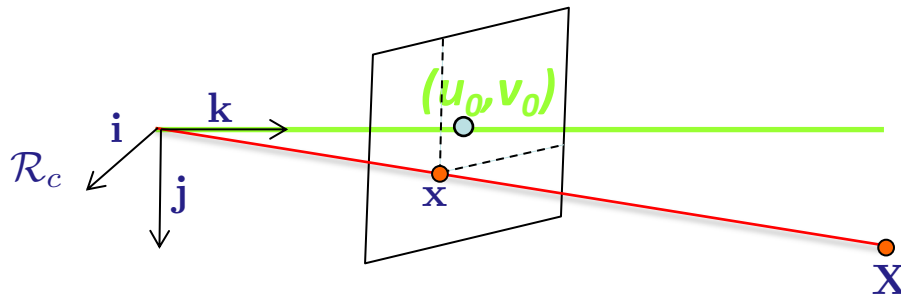
In \mathcal{R}_c , the perspective projection of a point $\mathbf{X} = (X, Y, Z)$ on the image point $\mathbf{x} = (x, y)$ with:

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

All the points on the CM straight line given by:

$$\begin{cases} fX - Zx = 0 \\ fY - Zy = 0 \end{cases}$$

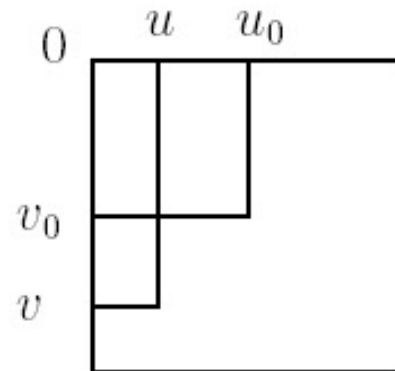
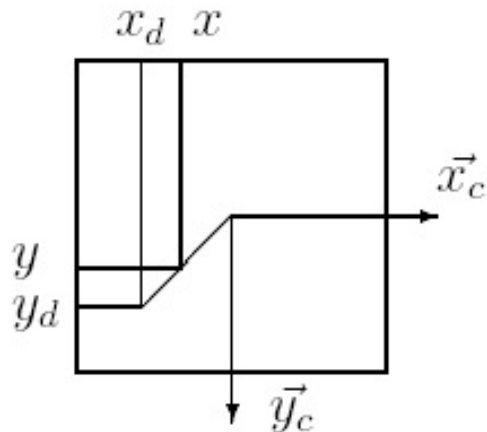
are projected on the same point \mathbf{x} . It is impossible to determine the position of \mathbf{X} without *a priori* knowledge with one camera.



Sampling

Let us define x_d by its position (u, v) in the digitized image (u and v expressed in pixel):

where l_x and l_y are the pixel size and u_0, v_0 are the translation of the center of the coordinate system (principal point coordinates)



Complete camera model

The model is given by

$$\begin{cases} u = u_0 + p_x \frac{X}{Z} \\ v = v_0 + p_y \frac{Y}{Z} \end{cases}$$

where

$$p_x = f/l_x, \quad p_y = f/l_y$$

The unknown parameters ξ , called intrinsic parameters are:

$$\xi = (u_0, v_0, p_x, p_y)$$



Camera model

When camera distortion can be neglected, the camera model is simply given by:

$$\begin{cases} u = u_0 + p_x \frac{X}{Z} \\ v = v_0 + p_y \frac{Y}{Z} \end{cases}$$



Perspective model: linear notation

$$\begin{cases} u = u_0 + p_x \frac{X}{Z} \\ v = v_0 + p_y \frac{Y}{Z} \end{cases}$$

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{K}} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\mathbf{\Pi}} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$



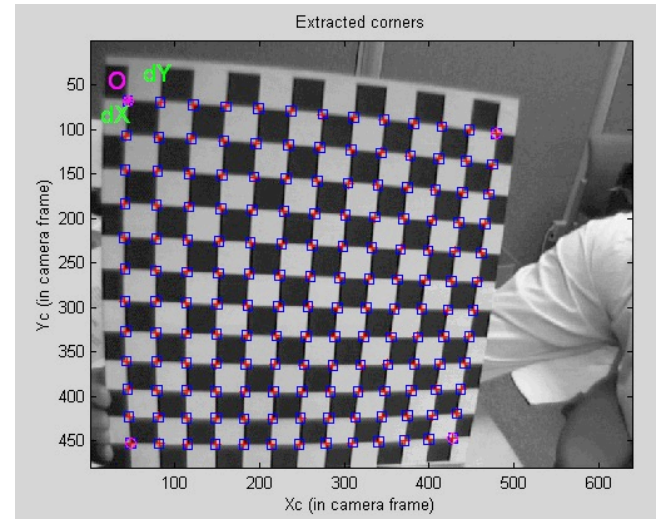
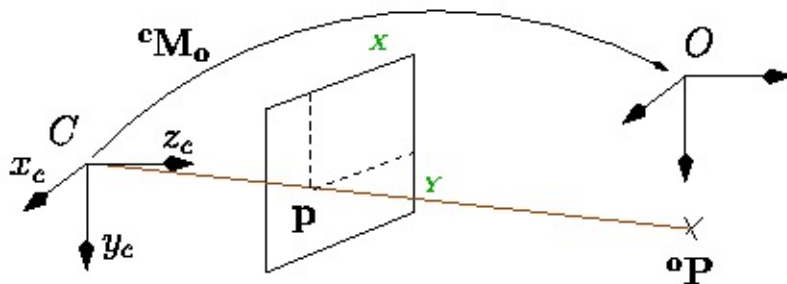
Distorsion



Pose

Let us now determine the pose...

Let us consider point coordinates $(X,Y,Z,1)$ and their projections $(x,y,1)$ in the image plane, can we invert the model ?



Frame transformation

Let us consider the world frame R_w . Points coordinates in the world frame (pattern frame):

If we want ${}^c\mathbf{X}$, we have to change frame

$${}^w\mathbf{X} = ({}^wX, {}^wY, {}^wZ, 1)^\top$$

$${}^c\mathbf{X} = {}^c\mathbf{T}_w {}^w\mathbf{X} \quad \text{with} \quad {}^c\mathbf{T}_w = \begin{pmatrix} {}^c\mathbf{R}_w & {}^c\mathbf{t}_w \\ \mathbf{0}_{3 \times 1} & 1 \end{pmatrix}$$

${}^c\mathbf{T}_w$ is an homogeneous matrix

Parameters in the homogeneous matrix ${}^c\mathbf{T}_w$ are the intrinsic parameters



Frame transformation

Frame transformation

$${}^c\mathbf{X} = {}^c\mathbf{T}_w {}^w\mathbf{X} \quad \text{with} \quad {}^c\mathbf{T}_w = \begin{pmatrix} {}^c\mathbf{R}_w & {}^c\mathbf{t}_w \\ \mathbf{0}_{3 \times 1} & 1 \end{pmatrix}$$

${}^c\mathbf{R}_w$ is a 3x3 rotation matrix and ${}^c\mathbf{t}_w$ is a translation vector.

${}^c\mathbf{T}_w$ expresses the position frame R_w in frame R_c



Pose

Frame transformation

$${}^c\mathbf{X} = {}^c\mathbf{T}_w {}^w\mathbf{X} \quad \text{with} \quad {}^c\mathbf{T}_w = \begin{pmatrix} {}^c\mathbf{R}_w & {}^c\mathbf{t}_w \\ \mathbf{0}_{3 \times 1} & 1 \end{pmatrix}$$

You can expand this equation:

$$\begin{pmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{22} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} {}^wX \\ {}^wY \\ {}^wZ \\ 1 \end{pmatrix}$$

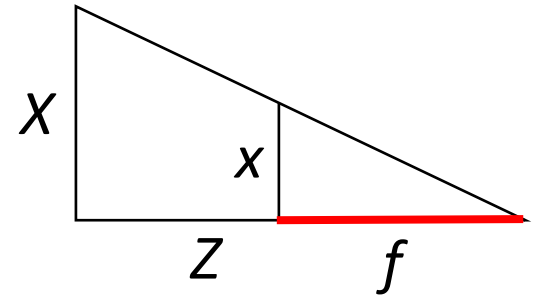
Unknown parameters in the homogeneous matrix ${}^c\mathbf{T}_w$ are the pose



Back to Thales

We know (x,y) and the object model wX

We seek the pose cT_w



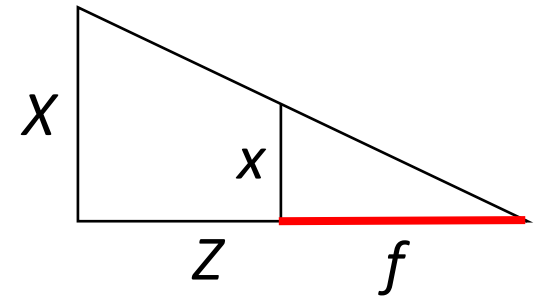
Generalization

We know (x,y) and the object model ${}^w\mathbf{X}$

We seek the pose ${}^c\mathbf{T}_w$

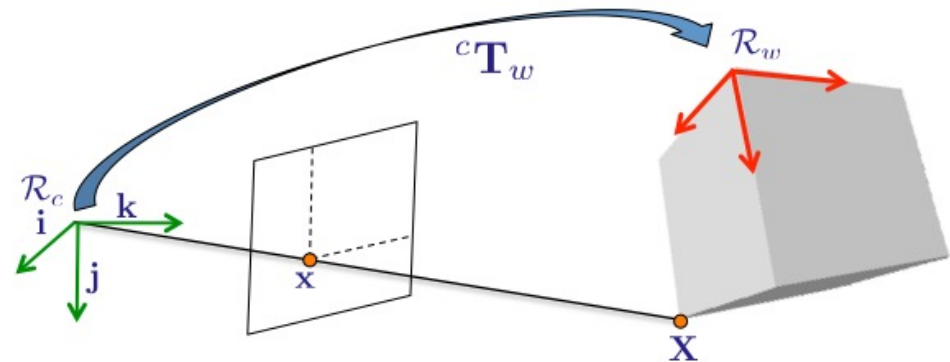
Solution is quite simple : change frame first

$${}^c\mathbf{X} = {}^c\mathbf{T}_w {}^w\mathbf{X} \quad \Leftrightarrow \quad \begin{cases} {}^cX = (\mathbf{r}_1 \ 0) {}^w\mathbf{X} + t_x \\ {}^cY = (\mathbf{r}_2 \ 0) {}^w\mathbf{X} + t_y \\ {}^cZ = (\mathbf{r}_3 \ 0) {}^w\mathbf{X} + t_z \end{cases}$$



Then project

$$\begin{cases} x = \frac{(\mathbf{r}_1 \ 0) {}^w\mathbf{X} + t_x}{(\mathbf{r}_3 \ 0) {}^w\mathbf{X} + t_z} \\ y = \frac{(\mathbf{r}_2 \ 0) {}^w\mathbf{X} + t_y}{(\mathbf{r}_3 \ 0) {}^w\mathbf{X} + t_z} \end{cases}$$



Pose estimation “Direct Linear Transform”

Let's note the pose ${}^c\mathbf{T}_w \Leftrightarrow$

$$\begin{cases} {}^cX = (\mathbf{r}_1 \ 0)^w \mathbf{X} + t_x \\ {}^cY = (\mathbf{r}_2 \ 0)^w \mathbf{X} + t_y \\ {}^cZ = (\mathbf{r}_3 \ 0)^w \mathbf{X} + t_z \end{cases}$$

$$\begin{cases} x = \frac{(\mathbf{r}_1 \ 0)^w \mathbf{X} + t_x}{(\mathbf{r}_3 \ 0)^w \mathbf{X} + t_z} \\ y = \frac{(\mathbf{r}_2 \ 0)^w \mathbf{X} + t_y}{(\mathbf{r}_3 \ 0)^w \mathbf{X} + t_z} \end{cases}$$

The perspective projection equation gives:

$$\begin{cases} r_{31}^w X x + r_{32}^w Y x + r_{33}^w Z x + x t_z - r_{11}^w X - r_{12}^w Y - r_{13}^w Z - t_x = 0 \\ r_{31}^w X y + r_{32}^w Y y + r_{33}^w Z y + y t_z - r_{21}^w X - r_{22}^w Y - r_{23}^w Z - t_y = 0 \end{cases}$$

Which with simple developments leads to:



“Linear” solution

We obtain an homogeneous system with 12 unknowns parameters:

$$\Gamma \mathbf{X} = \begin{pmatrix} \vdots \\ \Gamma_i \\ \vdots \end{pmatrix} \mathbf{X} = \mathbf{0}$$

Where

- Γ depends on the data extracted from the image
- \mathbf{X} is function of the parameters to be estimated

$$\Gamma_i \mathbf{X} = [0 \ 0]^T$$

Each point of the object gives 2 equations

$$\Gamma_i = \begin{pmatrix} -{}^wX_i & -{}^wY_i & -{}^wZ_i & 0 & 0 & 0 & x_i {}^wX_i & x_i {}^wY_i & x_i {}^wZ_i & -1 & 0 & x_i \\ 0 & 0 & 0 & -{}^wX_i & -{}^wY_i & -{}^wZ_i & y_i {}^wX_i & y_i {}^wY_i & y_i {}^wZ_i & 0 & -1 & -y_i \end{pmatrix}$$

$$\mathbf{X} = (r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}, t_x, t_y, t_z)^T$$



Solution of the linear system

System to be solved $\Gamma \mathbf{X} = \mathbf{0}$

- where \mathbf{X} is a non null vector of size $2n$

Solution 1

- Compute \mathbf{X} a vector of the null space of Γ (SVD)

Solution 2

- Considering that r_{ij} is a rotation matrix
- Solved the system under the constraint that $[r_{31}, r_{32}, r_{33}]$ is a unitary vector



POSIT or Dementhon-Davis method

International Journal of Computer Vision, 15, 123–141 (1995)
© 1995 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Model-Based Object Pose in 25 Lines of Code

DANIEL F. DEMENTHON AND LARRY S. DAVIS

Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742

Received October 1992; Revised March 1994; Accepted May 1994

Abstract. In this paper, we describe a method for finding the pose of an object from a single image. We assume that we can detect and match in the image four or more noncoplanar feature points of the object, and that we know their relative geometry on the object. The method combines two algorithms; the first algorithm, *POS* (Pose from Orthography and Scaling) approximates the perspective projection with a scaled orthographic projection and finds

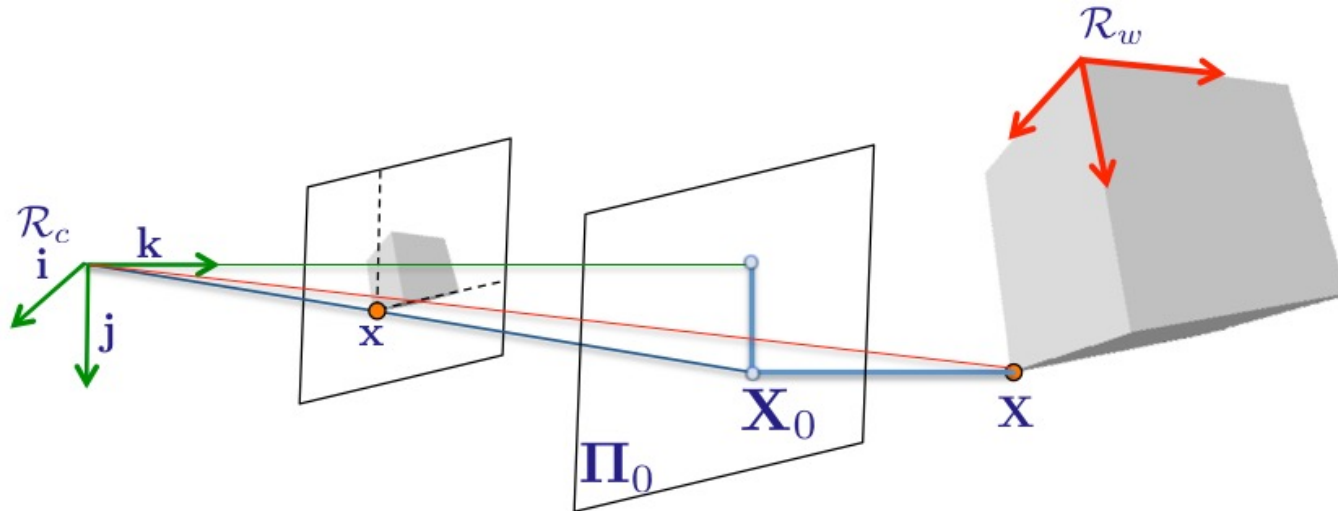
And... that is true...



POSIT

An alternative and very elegant solution

Pose estimation problem is linear under the scaled orthographic projection



Dementhon proposed to iteratively go back from the scaled

POS

Considering the projection equations, for each point (x,y)

$$\begin{cases} x = \frac{(\mathbf{r}_1 \ 0)^w \mathbf{X} + t_x}{(\mathbf{r}_3 \ 0)^w \mathbf{X} + t_z} \\ y = \frac{(\mathbf{r}_2 \ 0)^w \mathbf{X} + t_y}{(\mathbf{r}_3 \ 0)^w \mathbf{X} + t_z} \end{cases}$$

Or

$$\begin{cases} x_i (\mathbf{r}_3 \ t_z)^w \mathbf{X}_i = t_z^w \mathbf{X}_i^\top \mathbf{I} \\ y_i (\mathbf{r}_3 \ t_z)^w \mathbf{X}_i = t_z^w \mathbf{X}_i^\top \mathbf{J} \end{cases}$$

with

$$\mathbf{I}^\top = \frac{1}{t_z} (\mathbf{r}_1 \ t_x) \quad \text{and} \quad \mathbf{J}^\top = \frac{1}{t_z} (\mathbf{r}_2 \ t_z)$$



POS

Equations

$$\begin{cases} x_i (\mathbf{r}_3 \ t_z)^w \mathbf{X}_i = t_z {}^w \mathbf{X}_i^\top \mathbf{I} \\ y_i (\mathbf{r}_3 \ t_z)^w \mathbf{X}_i = t_z {}^w \mathbf{X}_i^\top \mathbf{J} \end{cases}$$

Can be rewritten as $\begin{cases} x_i (\varepsilon_i + 1) = {}^w \mathbf{X}_i^\top \mathbf{I} \\ y_i (\varepsilon_i + 1) = {}^w \mathbf{X}_i^\top \mathbf{J} \end{cases}$

$$\varepsilon_i = \frac{1}{t_z} (\mathbf{r}_3 \ 0)^w \mathbf{X}_i$$

With



If $\varepsilon_i = 0$ we have a scaled orthographic projection

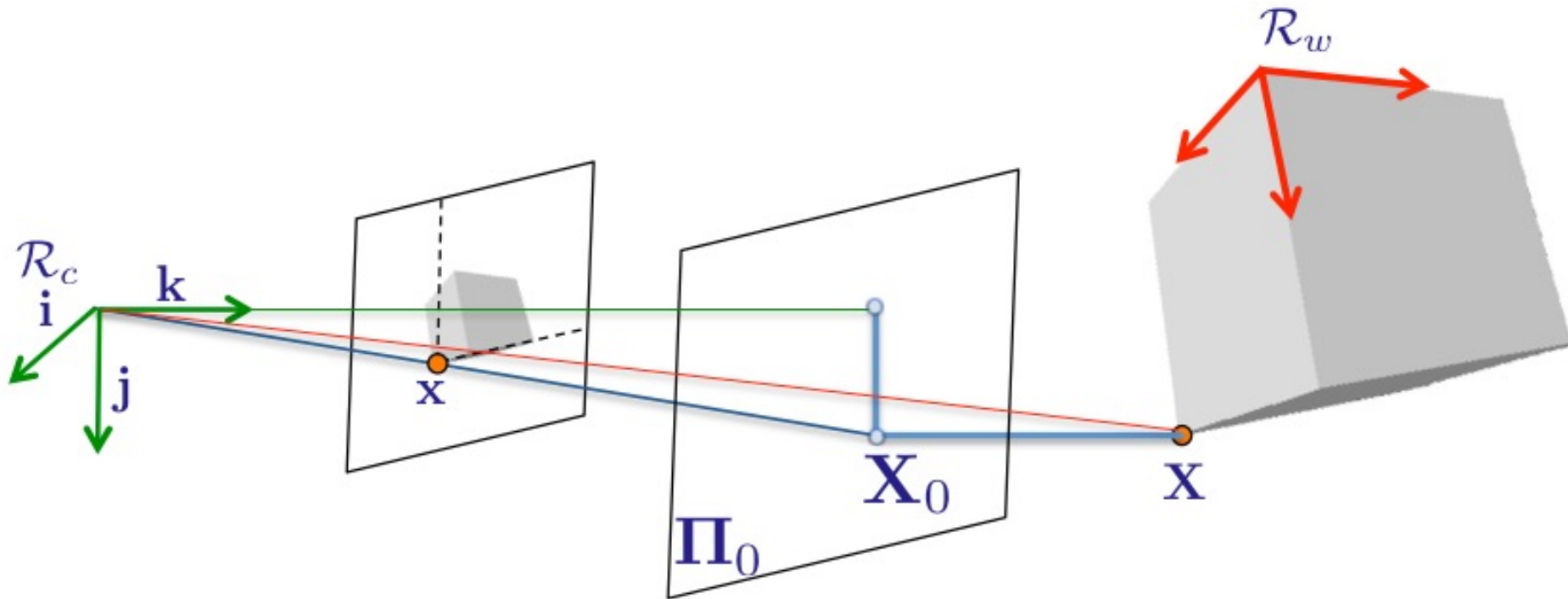
POS: error analysis

The error between the two projection models is

$$\mathbf{x}_i \in \varepsilon_i \quad \mathbf{y}_i \in \varepsilon_i$$

and

with
$$\varepsilon_i = \frac{c \mathbf{X}_i}{t_z}$$



Dementhon-Davis

We obtain $\mathbf{A}\mathbf{I} = \mathbf{B}_x$ and $\mathbf{A}\mathbf{J} = \mathbf{B}_y$

with

$$\mathbf{A} = \begin{pmatrix} \dots & w\mathbf{X}_i^\top & \dots \end{pmatrix}^\top$$
$$\mathbf{B}_x = \begin{pmatrix} \dots & x_i(\varepsilon_i + 1) & \dots \end{pmatrix}^\top$$
$$\mathbf{B}_y = \begin{pmatrix} \dots & y_i(\varepsilon_i + 1) & \dots \end{pmatrix}^\top$$

That is two linear systems with N equations and 4 unknowns

- If ε_i is known

4 non coplanar points are necessary to solve such system

Just one more thing...

- ε_i is unknown



POSIT: iterative POS

1. Initialization $\mathbf{A}\mathbf{I} = \mathbf{B}_x$ and $\mathbf{A}\mathbf{J} = \mathbf{B}_y$

2. Solve the linear systems

- $\mathbf{I} = \mathbf{A}^+ \mathbf{B}_x$ and $\mathbf{J} = \mathbf{A}^+ \mathbf{B}_y$
- \mathbf{A}^+ has to be computed only once

3. Let \mathbf{I}^* (resp \mathbf{J}^*) be the three first rows of vector \mathbf{I} (resp \mathbf{J}), we have

$$\mathbf{I}^* = \mathbf{r}_1/t_z \text{ and } \mathbf{J}^* = \mathbf{r}_2/t_z.$$

considering the properties of a rotation matrix

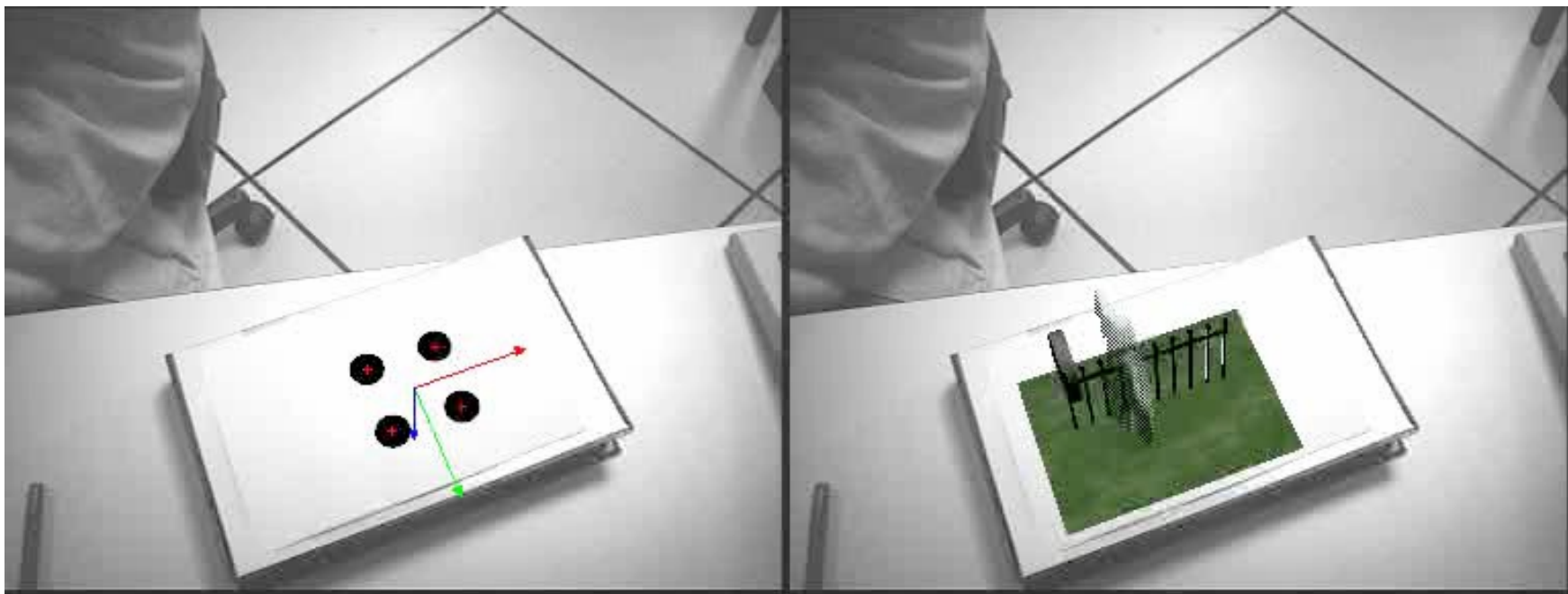
$$\mathbf{r}_1 = \mathbf{I}^* / \|\mathbf{I}^*\| \text{ and } \mathbf{r}_2 = \mathbf{J}^* / \|\mathbf{J}^*\|. \quad \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \quad t_z = 1/\|\mathbf{I}^*\|$$

4. Recompute $\varepsilon_i = \frac{1}{t_z} (\mathbf{r}_3 \ 0)^w \mathbf{X}_i$

5. Iterate



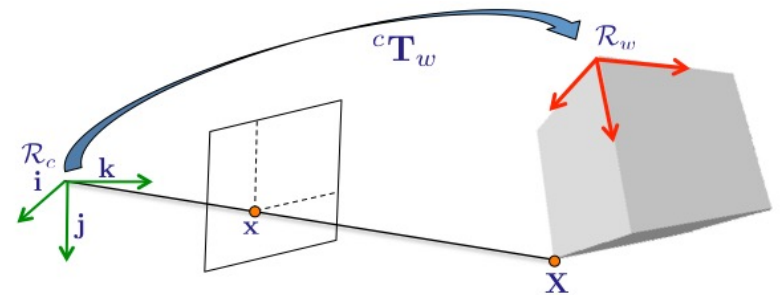
Dementhon-Davis



Pose estimation: the “gold-standard” solution

Goal

- Estimate the pose cT_w of an object with respect to the camera frame



Example for point features

- Minimizing the error between the observation \mathbf{x}_i and the projection of the model in the image

$$\hat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^N (\mathbf{x}_i - \Pi {}^cT_w {}^w\mathbf{X}_i)^2$$

where ${}^w\mathbf{X}$ are the coordinates of the same points in the object frame

- \mathbf{q} is a minimal representation of cT_w



Pose: non linear minimization

Solving
$$\hat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^N (\mathbf{x}_i - \Pi^c \mathbf{T}_w^w \mathbf{X}_i)^2$$

consists in minimizing the error $\mathbf{e}(\mathbf{q})$ defined by:

$$\mathbf{e}(\mathbf{q}) = (\mathbf{x} - \mathbf{x}(\mathbf{q}))^\top (\mathbf{x} - \mathbf{x}(\mathbf{q}))$$

with

$$\mathbf{x} = (\dots, \mathbf{x}_i, \dots)^\top$$

$$\mathbf{x}(\mathbf{q}) = (\dots, \Pi^c \mathbf{T}_w^w \mathbf{X}_i, \dots)^\top$$



Linearization of the non-linear system

Problem: no general method to solve $\mathbf{e}(\mathbf{q}) = 0$

There exists iterative method that linearize the problem in order to find an adequate solution

First order Taylor expansion around \mathbf{q}

$$\begin{aligned}\mathbf{e}_i(\mathbf{q} + \delta\mathbf{q}) &= \mathbf{e}_i(\mathbf{q}) + \delta\mathbf{q}_1 \frac{\partial \mathbf{e}_i(\mathbf{q})}{\partial \mathbf{q}_1} + \dots + \delta\mathbf{q}_n \frac{\partial \mathbf{e}_i(\mathbf{q})}{\partial \mathbf{q}_n} + O(|\delta\mathbf{q}|^2) \\ &\approx \mathbf{e}_i(\mathbf{q}) + \mathbf{J}(\mathbf{q})\delta\mathbf{q}\end{aligned}$$

where $\mathbf{J}(\mathbf{q}) = \left(\frac{\partial \mathbf{e}_i(\mathbf{q})}{\partial \mathbf{q}_1}, \dots, \frac{\partial \mathbf{e}_i(\mathbf{q})}{\partial \mathbf{q}_n} \right)^\top$ is the gradient of \mathbf{e}_i in \mathbf{q} and where second order terms are neglected that the Jacobian



Solving the linearized system

With the Gauss-Newton method, we do not want to determine the value of \mathbf{q} that ensures $\mathbf{e}(\mathbf{q})=0$ but the value that minimizes the cost function:

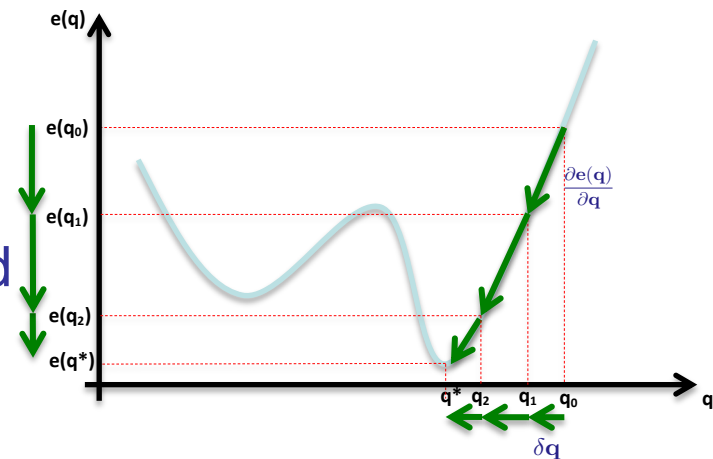
$$E(\mathbf{q} + \delta\mathbf{q}) = \|\mathbf{e}(\mathbf{q} + \delta\mathbf{q})\| \approx \|\mathbf{e}(\mathbf{q}) + \mathbf{J}(\mathbf{q})\delta\mathbf{q}\|$$

$$\delta\mathbf{q} = -\mathbf{J}(\mathbf{q})^+ \mathbf{e}(\mathbf{q})$$

This is a linear minimization problem (solved by a least-square approach) and we have:

$$\mathbf{q}_{k+1} = \mathbf{q}_k \oplus \delta\mathbf{q} = \exp^{\delta\mathbf{q}} \mathbf{q}$$

Solved by an iterative least square method



Issue: computing the Jacobian

We have to compute the Jacobian that links the variation of the measurements $\mathbf{x} = (x,y)$ to the variation of the pose.

That is :

$$\frac{\partial \mathbf{x}}{\partial t} = \frac{\partial \mathbf{x}}{\partial \mathbf{q}} \frac{d\mathbf{q}}{dt}$$

or

$$\dot{\mathbf{x}} = \mathbf{L}_x \mathbf{v}$$

*\mathbf{L}_x links the motion of a point in the image to the velocity of a camera
is the desired Jacobian*



Jacobian: case of the point

Some definitions

- Let (O, x, y, z) be the camera frame
- Let $\mathbf{X}=(X, Y, Z)$ be the 3D position the point
- Let the camera velocity be $\mathbf{v} = (\mathbf{v}, \boldsymbol{\omega})^\top = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)^\top$

The relation that link the point velocity $\dot{\mathbf{X}}$ to the camera velocity is given by:

$$\dot{\mathbf{X}} = -\mathbf{v} - \boldsymbol{\omega} \times \mathbf{X}$$



Jacobian: case of the point

$$\dot{\mathbf{X}} = -\mathbf{v} - \boldsymbol{\omega} \times \mathbf{X} \quad \Leftrightarrow \quad \begin{cases} \dot{X} &= -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} &= -v_y - \omega_z X + \omega_x Z \\ \dot{Z} &= -v_z - \omega_x Y + \omega_y X \end{cases}$$

On the other hand, the perspective equation gives

$$\begin{cases} x &= \frac{X}{Z} \\ y &= \frac{Y}{Z} \end{cases}$$

Which can be derived

$$\begin{cases} \frac{\partial x}{\partial t} &= \frac{\partial x}{\partial X} \frac{\partial X}{\partial t} + \frac{\partial x}{\partial Z} \frac{\partial Z}{\partial t} \\ \frac{\partial y}{\partial t} &= \frac{\partial y}{\partial Y} \frac{\partial Y}{\partial t} + \frac{\partial y}{\partial Z} \frac{\partial Z}{\partial t} \end{cases} \Leftrightarrow \begin{cases} \dot{x} &= \frac{\dot{X}}{Z} - \frac{X}{Z^2} \dot{Z} \\ \dot{y} &= \frac{\dot{Y}}{Z} - \frac{Y}{Z^2} \dot{Z} \end{cases} \quad (1)$$



Jacobian: case of the point

Considering $(\dot{X}, \dot{Y}, \dot{Z})$ obtained in (1)

$$\begin{cases} \dot{x} &= \frac{\dot{X}}{Z} - \frac{X}{Z^2} \dot{Z} &= \frac{-v_x - \omega_y Z + \omega_z Y}{Z} - \frac{Z}{Z^2} (-v_z - \omega_x Y + \omega_y X) \\ \dot{y} &= \frac{\dot{Y}}{Z} - \frac{Y}{Z^2} \dot{Z} &= \frac{-v_y - \omega_z X + \omega_x Z}{X} - \frac{Y}{Z^2} (-v_z - \omega_x Y + \omega_y X) \end{cases}$$

or

$$\begin{cases} \dot{x} &= -\frac{1}{Z} v_x & + \frac{X}{Z^2} v_z & + \frac{XY}{Z^2} \omega_x & - (1 + \frac{X^2}{Z^2}) \omega_y & + \frac{Y}{Z} \omega_z \\ \dot{y} &= & -\frac{1}{Z} v_y & + \frac{Y}{Z^2} v_z & + (1 + \frac{Y^2}{Z^2}) \omega_x & - \frac{XY}{Z^2} \omega_y & - \frac{X}{Z} \omega_z \end{cases}$$



Jacobian: case of the point

We finally have

$$\begin{cases} \dot{x} &= -\frac{1}{Z}v_x & +\frac{x}{Z}v_z & +xy\omega_x & -(1+x^2)\omega_y & +y\omega_z \\ \dot{y} &= & -\frac{1}{Z}v_y & +\frac{y}{Z}v_z & +(1+y^2)\omega_x & -xy\omega_y & -x\omega_z \end{cases}$$

or

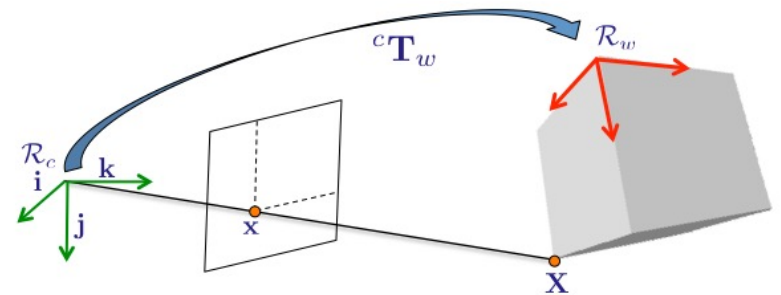
$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{pmatrix} \begin{pmatrix} v \\ \omega \end{pmatrix}$$



Pose estimation: the “gold-standard” solution

Goal

- Estimate the pose cT_w of an object with respect to the camera frame



Example for point features

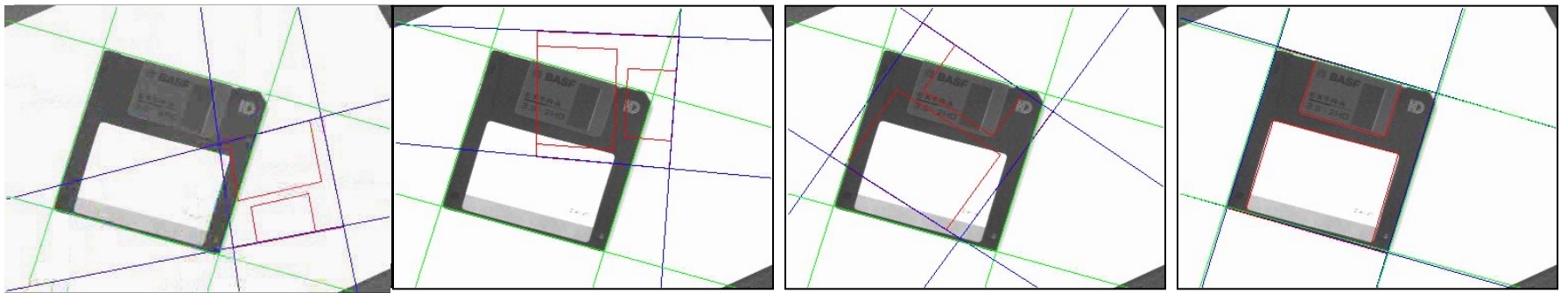
- Minimizing the error between the observation \mathbf{x}_i and the projection of the model in the image

$$\hat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^N (\mathbf{x}_i - \Pi {}^cT_w {}^w\mathbf{X}_i)^2$$

where ${}^w\mathbf{X}$ are the coordinates of the same points in the object frame

- \mathbf{q} is a minimal representation of cT_w

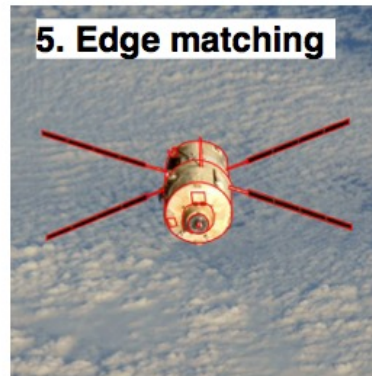
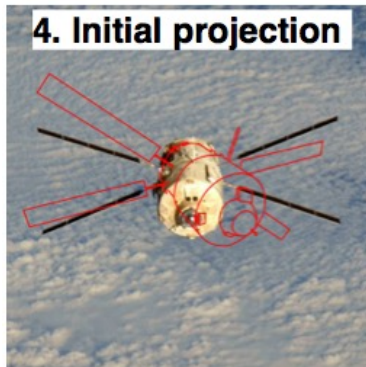
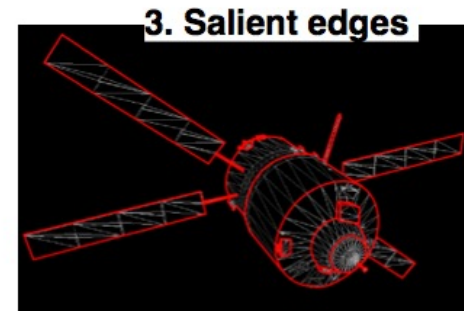
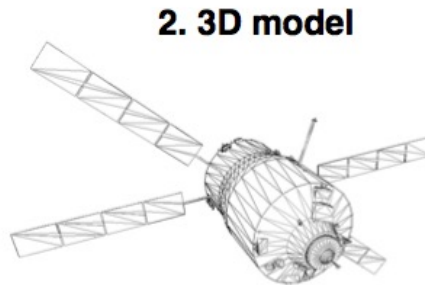
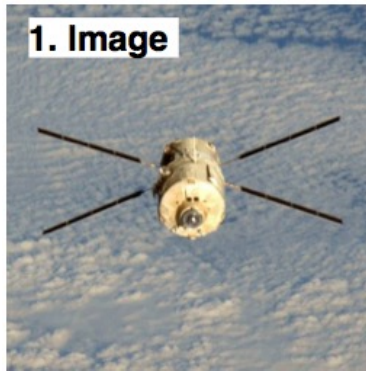




Pose calculation via non-linear methods



Beyond points



→ **3D position and attitude assessment**

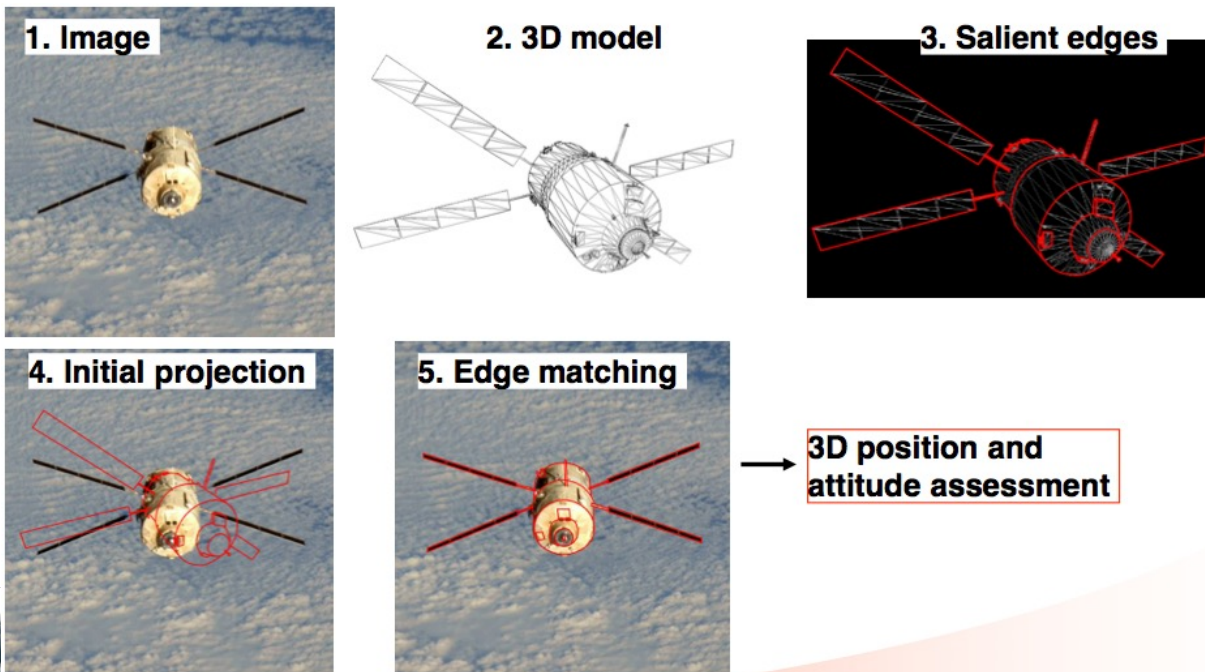


Markerless tracking

Similar approach but point to contour distance

$${}^{t+1}\hat{\mathbf{q}} = \arg \min_{\mathbf{q}} \sum_i d_{\perp}(L(\mathbf{q}), {}^{t+1}\mathbf{x}_i)$$

$d_{\perp}(L_i(\mathbf{q}), {}^{t+1}\mathbf{x}_i)$ is the squared distance between the point \mathbf{x}_i and

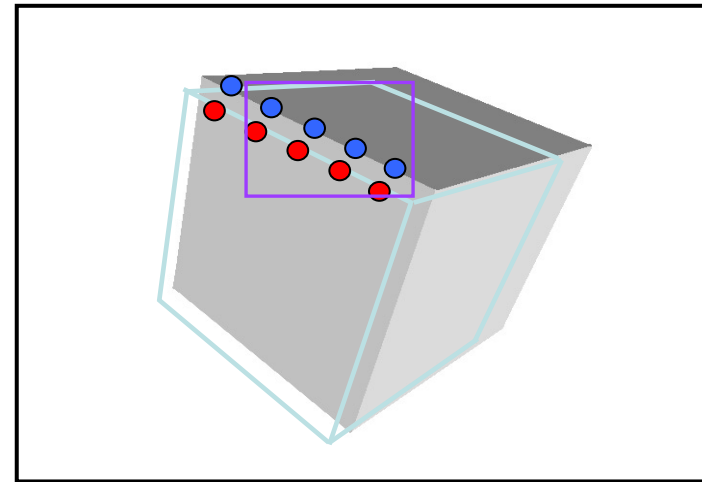
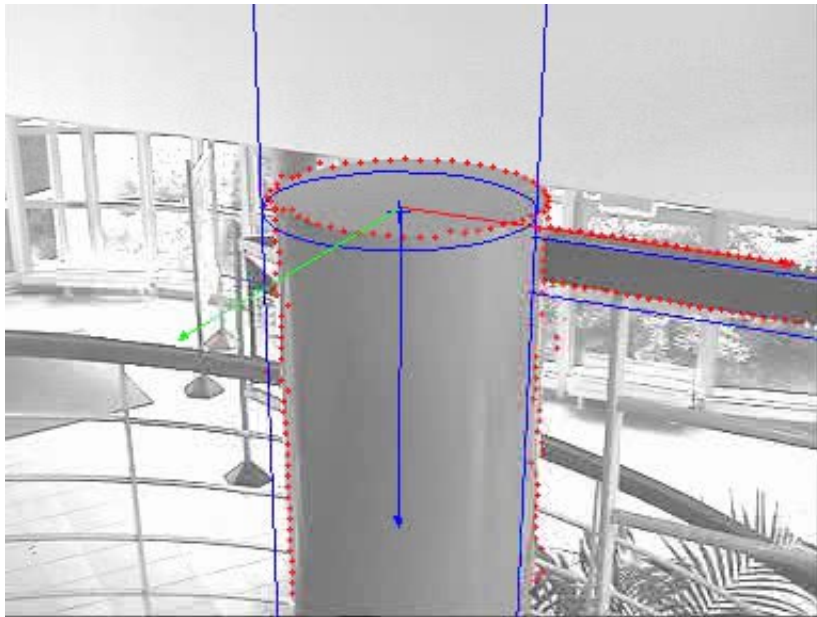


Beyond points

Distance to a moving line

- \mathbf{x}_i : point extracted in the image using, eg, the ECM algorithm
- $L(\mathbf{q})$: projection of the object model for pose \mathbf{r}

$$d_{\perp}(L(\mathbf{q}), {}^{t+1}\mathbf{x}_i)$$

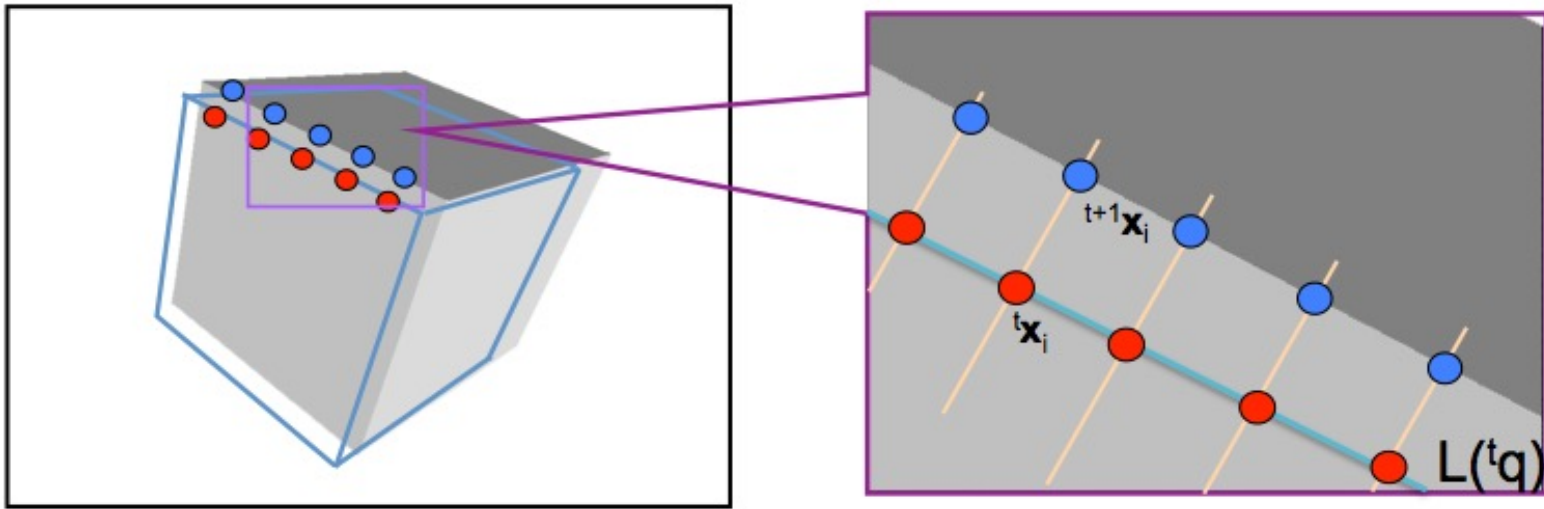


Beyond points

Distance to a moving line

- \mathbf{x}_i : point extracted in the image using, eg, the ECM algorithm
- $L(\mathbf{q})$: projection of the object model for pose \mathbf{r}

$$d_{\perp}(L(\mathbf{q}), {}^{t+1}\mathbf{x}_i)$$



Pose estimation: robustness to outliers

The residue is given by:

$$\mathbf{e}_\rho(\mathbf{q}) = \rho(\mathbf{x} - \mathbf{x}(\mathbf{q}))$$

- where ρ is a robust function (M-estimation)

- Minimize
$$\mathbf{e}_\rho(\mathbf{q}) = \mathbf{D}(\mathbf{x} - \mathbf{x}(\mathbf{q}))$$

The control law, similar to an IRLS, which minimizes $\mathbf{x} - \mathbf{x}(\mathbf{q})$ is given by

$$\delta \mathbf{q} = -(\mathbf{D}\mathbf{J}(\mathbf{q}))^+ \mathbf{D}\mathbf{e}_\rho(\mathbf{q})$$

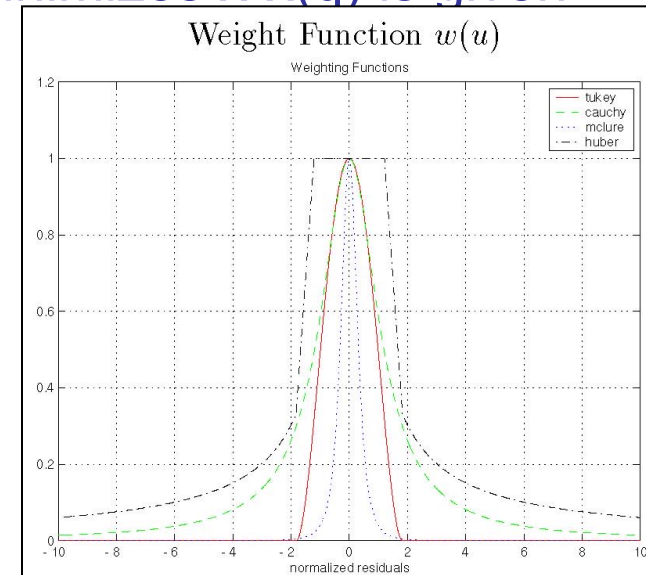
where

$$\mathbf{D} = \begin{pmatrix} w_1 & & 0 \\ & \ddots & \\ 0 & & w_n \end{pmatrix}$$

Tukey's M-estimator

$$w_i = \frac{\psi(\delta_i/\sigma)}{\delta_i/\sigma}$$

$$\psi(u) = \begin{cases} u(C^2 - u^2)^2 & |u| \leq C \\ 0 & \text{else} \end{cases}$$

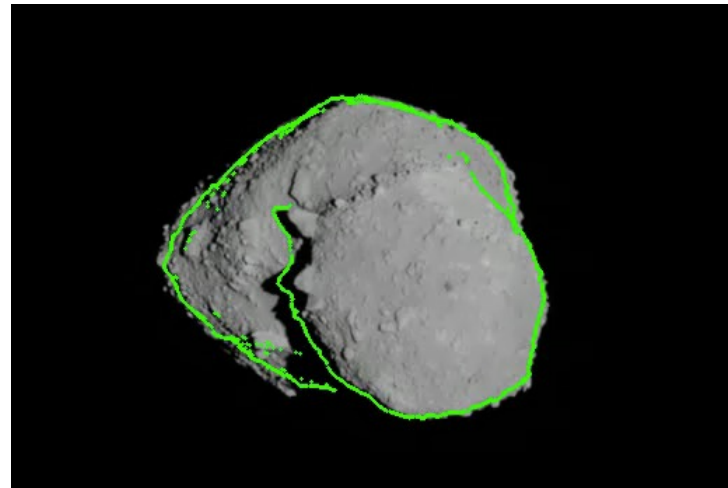
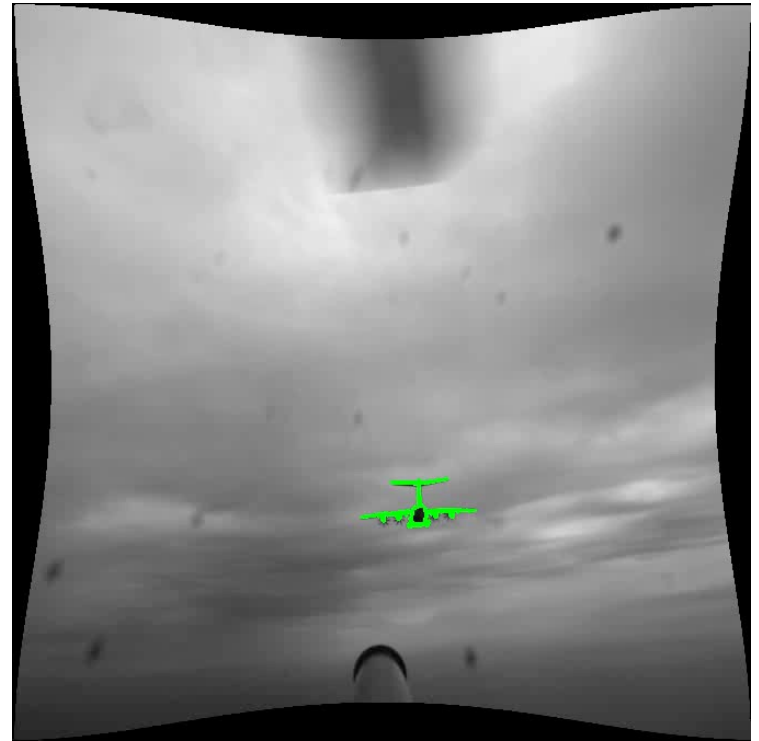


Model-based trackers

Such approach is quite efficient

It did the job pretty well

At video rate [Petit ICRA 2014]



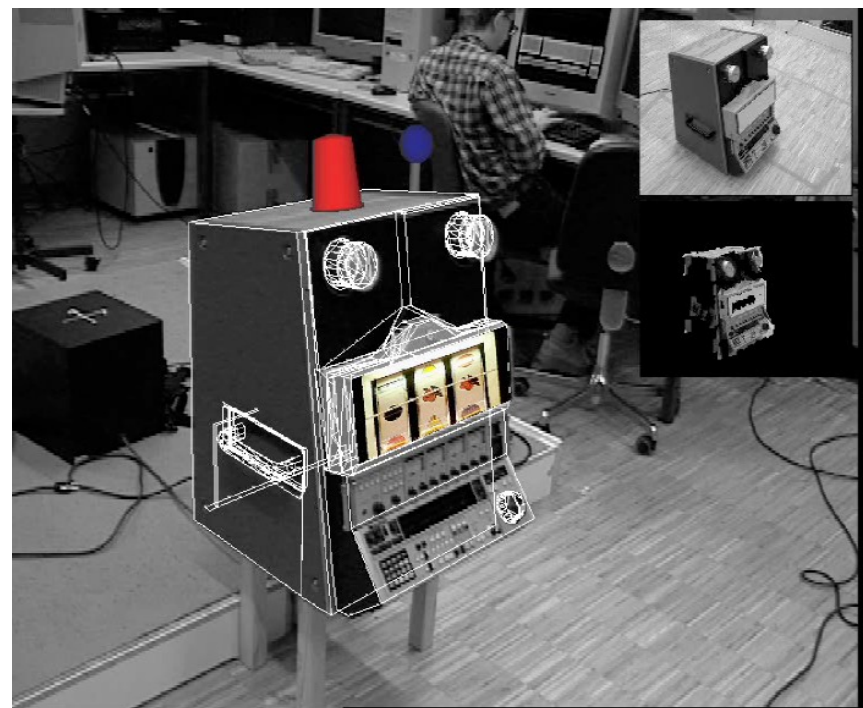
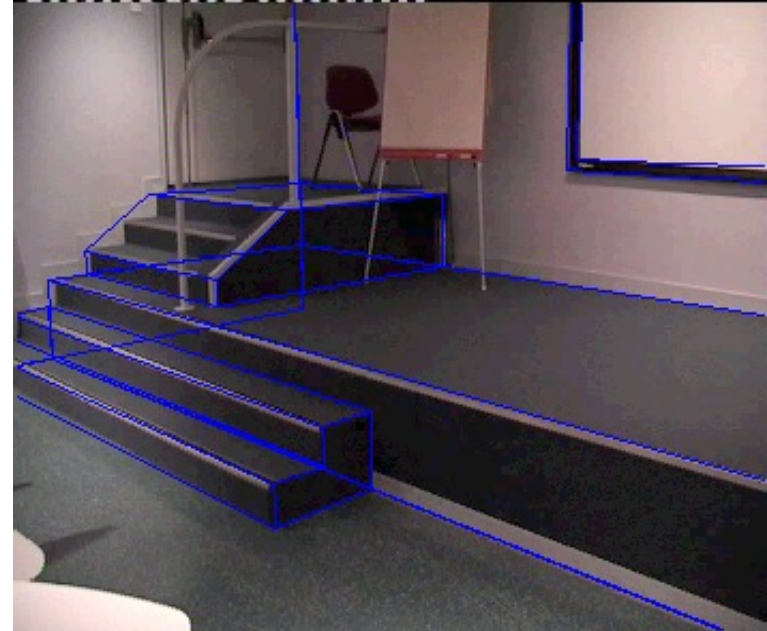
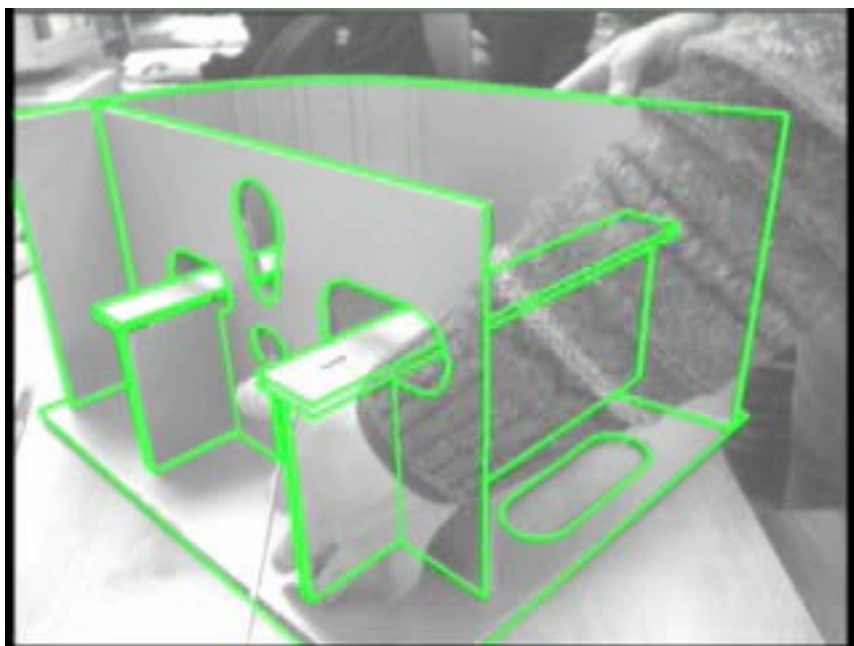
Spatial applications

Manoeuvre Atlantis /ISS

Rendez-vous Soyuz/ISS

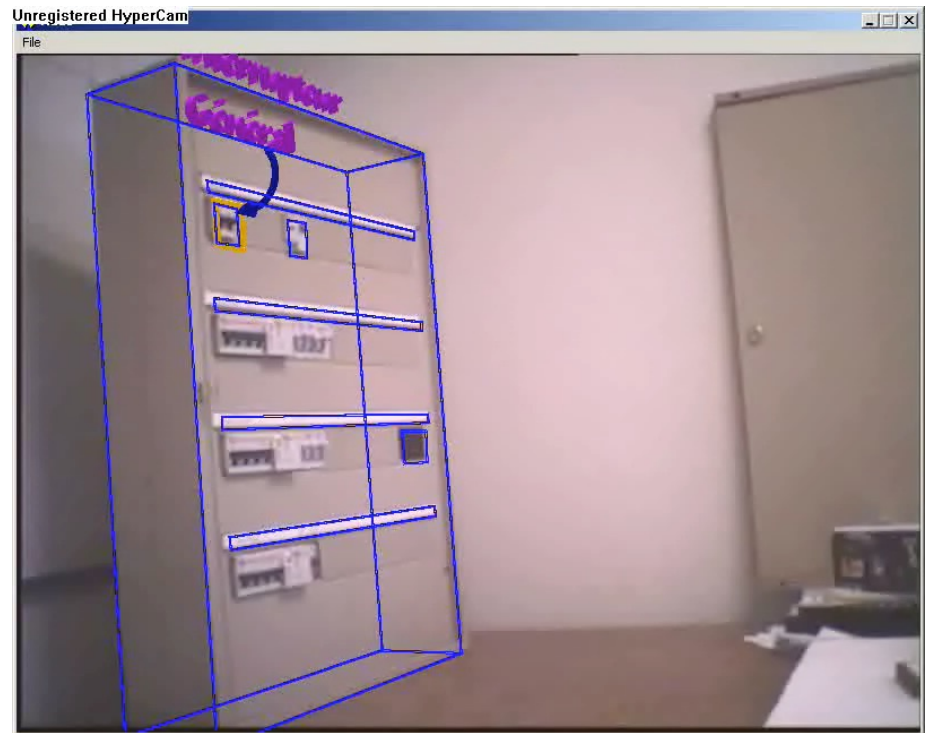
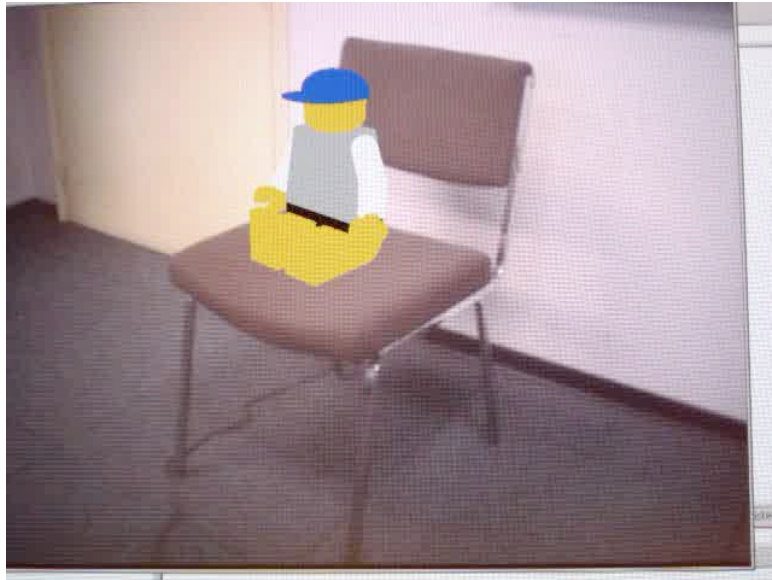


3D model-based tracking



3D model-based tracking: augmented reality

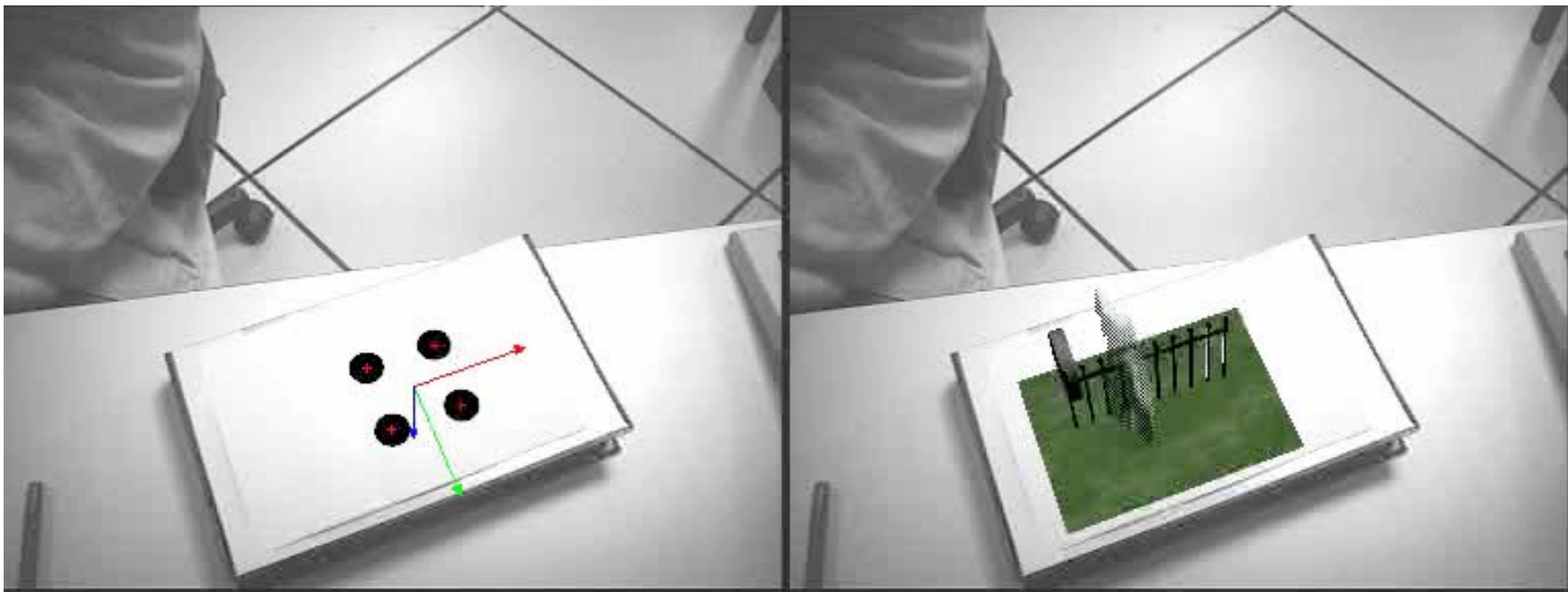
Application to augmented reality



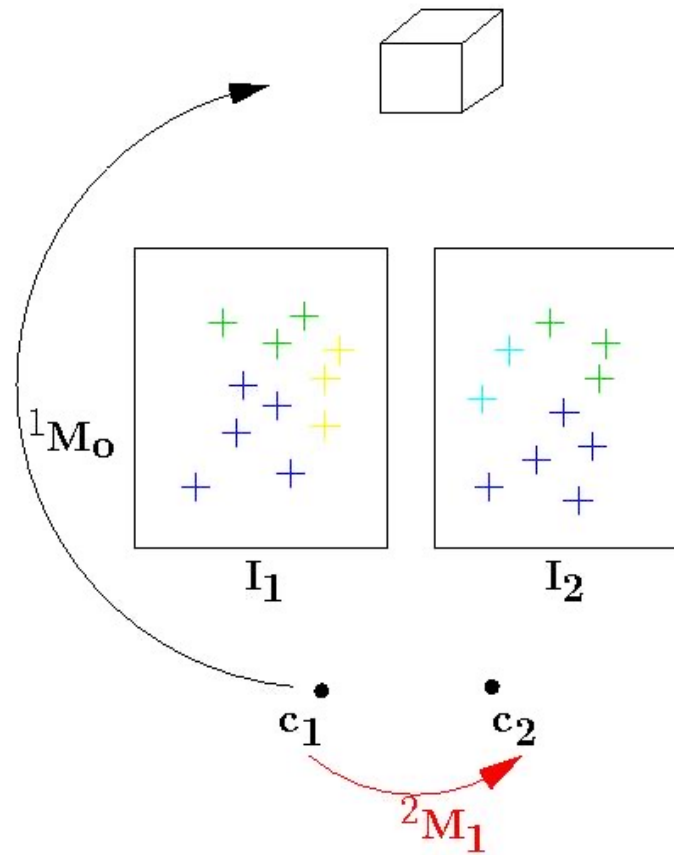
3D model-based tracking: augmented reality

Application to augmented reality





Second : displacement estimation



Motion estimation

$\text{tr}(\mathbf{x}_2)$ is the coordinates of a point *transferred* in a reference image according to the camera displacement

$$\mathbf{e}({}^2\hat{\mathbf{q}}_1) = \underset{\mathbf{q}}{\text{argmin}} \sum_i (\mathbf{x}_2 - {}^2tr_1(\mathbf{x}_1, {}^2\mathbf{q}_1))^2$$

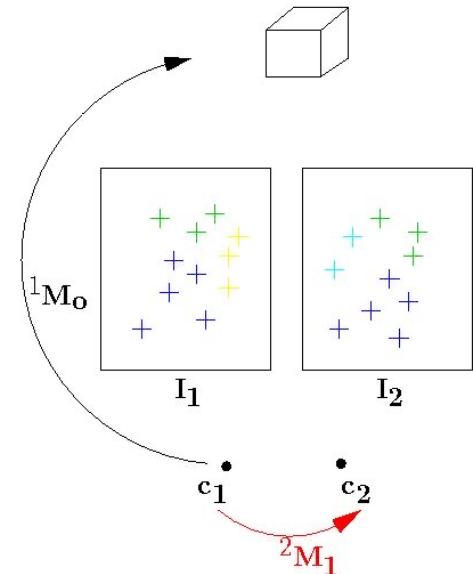
Allows to estimate the camera displacement

Advantages

- Implicit spatio-temporal constraints
- Less jitter

Issue

- Prone to drift if reference image is modified



Motion estimation

Point transfert

- For planar object the transfert is function of the camera displacement ${}^2\mathbf{T}_1$ and of the homography ${}^2\mathbf{H}_1$ $\mathbf{x}_2 = {}^2tr_1(\mathbf{x}_1, {}^2\mathbf{q}_1) = {}^2\mathbf{H}_1\mathbf{x}_1$

$$\text{with } {}^2\mathbf{H}_1 = {}^2\mathbf{R}_1 + \frac{{}^1\mathbf{n}^\top}{{}^1d} {}^2\mathbf{t}_1$$

- If ${}^1\mathbf{T}_0$ is known, estimating the displacement is equivalent to a pose estimation

Solutions

- Estimation of H [Simon, Berger IEEE CGA 02][Benhimane Malis IJRR 07]
- Estimation of R, t [Pressigout Marchand ICPR 04]

Illumination

- May also considered more complex illumination model in Δ



Case of planar scenes (in normalized coordinates)

Let us assume that points belong to a plane $\mathcal{P}({}^1\mathbf{n}, {}^1d)$

$$\left. \begin{array}{l} {}^1\tilde{\mathbf{X}} \in \mathcal{P}({}^1\mathbf{n}, {}^1d) \Leftrightarrow {}^1\mathbf{n}^\top {}^1\tilde{\mathbf{X}} = {}^1d \\ {}^2\tilde{\mathbf{X}} = {}^2\mathbf{R}_1 {}^1\tilde{\mathbf{X}} + {}^2\mathbf{t}_1 \end{array} \right\} \Rightarrow {}^2\tilde{\mathbf{X}} = {}^2\mathbf{R}_1 {}^1\tilde{\mathbf{X}} + {}^1\mathbf{t}_2 \frac{{}^1\mathbf{n}^\top}{{}^1d} {}^1\tilde{\mathbf{X}}$$

and $Z_1 \mathbf{x}_1 = {}^1\tilde{\mathbf{X}}$ and $Z_2 \mathbf{x}_2 = {}^2\tilde{\mathbf{X}}$

leading to $\lambda \mathbf{x}_2 = {}^2\mathbf{H}_1 \mathbf{x}_1$

with

$${}^2\mathbf{H}_1 = {}^2\mathbf{R}_1 + \frac{{}^1\mathbf{n}^\top}{{}^1d} {}^2\mathbf{t}_1 \quad \text{and} \quad \lambda = \frac{Z_2}{Z_1}$$



Homography estimation is a linear problem

For each point we have (in homogeneous coordinates) :

$$\mathbf{x}_2 = {}^2\mathbf{H}_1 \mathbf{x}_1$$

$$\mathbf{x}_2 \times {}^2\mathbf{H}_1 \mathbf{x}_1 = 0$$

which is equivalent to:

$$\underbrace{\begin{pmatrix} \mathbf{0}^\top & -w_{i_2} \mathbf{x}_1^\top & y_{i_2} \mathbf{x}_1^\top \\ w_{i_2} \mathbf{x}_1^\top & \mathbf{0}^\top & -x_{i_2} \mathbf{x}_1^\top \\ -y_{i_2} \mathbf{x}_1^\top & x_{i_2} \mathbf{x}_1^\top & \mathbf{0}^\top \end{pmatrix}}_{\mathbf{A}_i (3 \times 9)} \underbrace{\begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix}}_{\mathbf{h} (9 \times 1)} = 0$$



Homography estimation is a linear problem

For each point we have (in homogeneous coordinates) :

$$\mathbf{x}_2 = {}^2\mathbf{H}_1 \mathbf{x}_1$$

which is equivalent to: $\mathbf{x}_2 \times {}^2\mathbf{H}_1 \mathbf{x}_1 = 0$

$$\underbrace{\begin{pmatrix} \mathbf{0}^\top & -w_2 \mathbf{x}_1^\top & y_2 \mathbf{x}_1^\top \\ w_2 \mathbf{x}_1^\top & \mathbf{0}^\top & -x_2 \mathbf{x}_1^\top \end{pmatrix}}_{\mathbf{A} (2 \times 9)} \underbrace{\begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix}}_{\mathbf{h} (9 \times 1)} = 0$$

it can be solved using an SVD decomposition

$$\Gamma = \mathbf{U} \mathbf{D} \mathbf{V}^\top$$

\mathbf{h} is the vector of \mathbf{V} associated with the smallest singular value of Γ



From homography to Pose [Simon et al 99]

For planar scenes, one can directly and easily compute the pose when the 3D position of some points is known on the reference plane.

All the point lies in the plane ${}^wZ = 0$

Thus ${}^w\mathbf{X} = ({}^wX, {}^wY, 0, 1)^\top$

The projection is then given by

$$\mathbf{x}_0 = \Pi {}^0\mathbf{T}_w {}^w\mathbf{X} = \Pi \begin{pmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & {}^0\mathbf{t}_w \end{pmatrix} \begin{pmatrix} {}^wX \\ {}^wY \\ 0 \\ 1 \end{pmatrix}$$

$$\begin{aligned} \text{Or } \mathbf{x}_0 &= \Pi \begin{pmatrix} \mathbf{c}_1 & \mathbf{c}_2 & {}^0\mathbf{t}_w \end{pmatrix} ({}^wX, {}^wY, 1)^\top \\ &= {}^0\mathbf{H}_w ({}^wX, {}^wY, 1)^\top \end{aligned}$$



From homography to Pose

$$\mathbf{x}_0 = {}^0\mathbf{H}_w ({}^wX, {}^wY, 1)^\top$$

${}^0\mathbf{H}_w$ is a homography that maps the plane of the object (${}^wZ=0$) on the image plane.

It can be computed using the DLT algorithm

Knowing ${}^0\mathbf{H}_w$, the pose ${}^0\mathbf{T}_w$ can be easily computed noting that

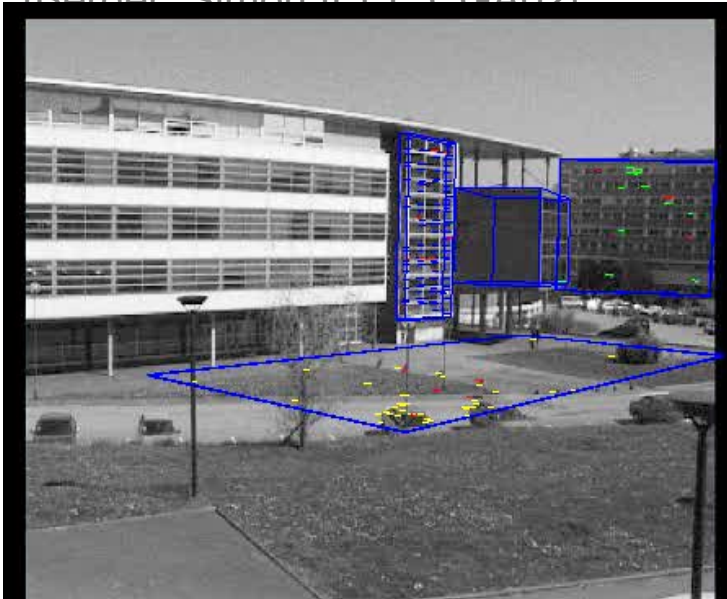
$$(\mathbf{c}_1, \mathbf{c}_2, {}^0\mathbf{t}_w) = \Pi^{-1} {}^0\mathbf{H}_w$$

Considering that the rotation matrix is orthogonal, the third column of the rotation matrix is computed such that $\mathbf{c}_3 = \mathbf{c}_1 \times \mathbf{c}_2$



From keypoints tracking/matching to 3D tracking

[Borger, Simon, IEEE CGA02]



[Lepetit]



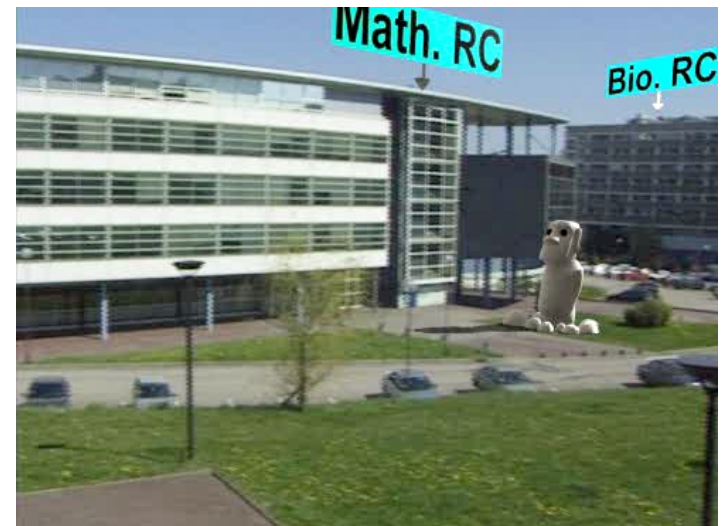
A remark:

Multi-plane tracking

Pose from planar structures

Constraints in the homography estimation

[Simon, Berger IEEE CGA 02]



Virtual visual servoing

Estimation of R and t [Pressigout Marchand ICPR'04]



AR Book Application



KLT Like approach

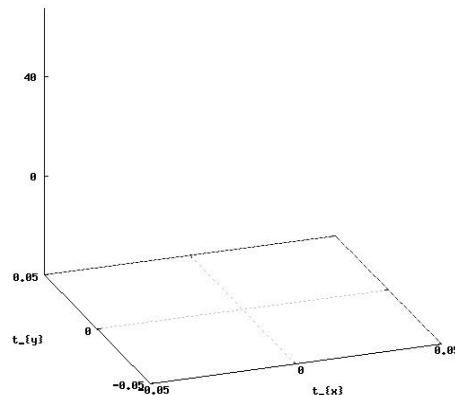
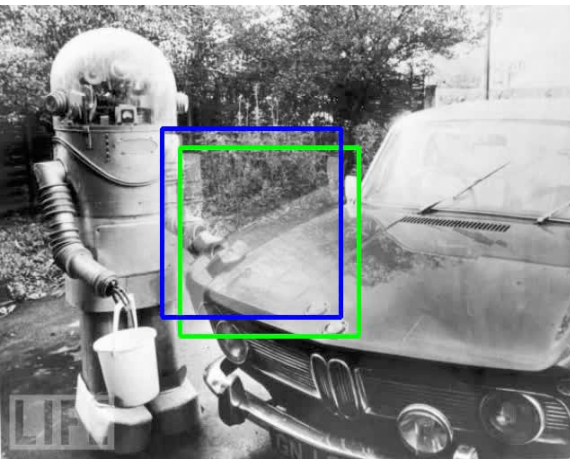
Use all the pixel in the tracked patch

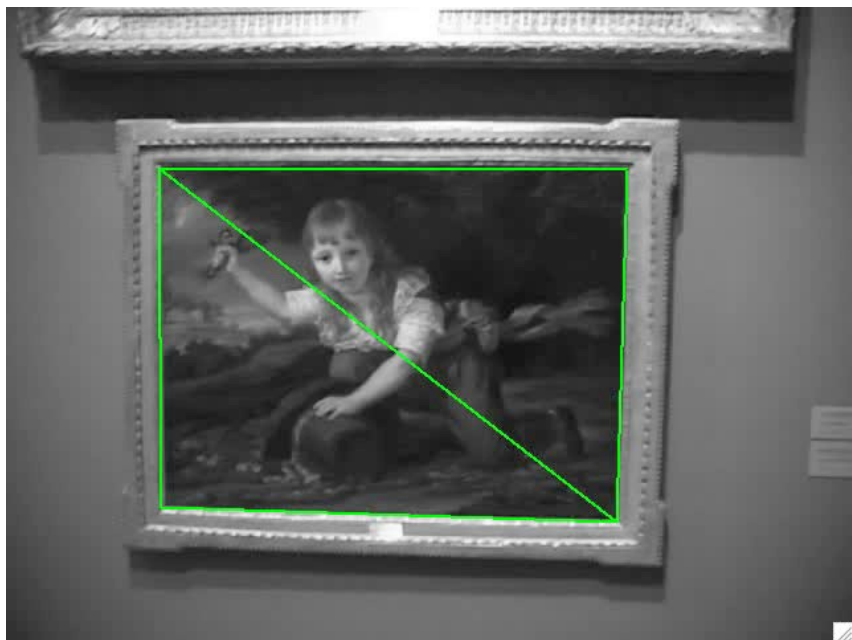
$$\Delta = \sum_i (I_1(p_1^i) - I_2({}^2tr_1(p_1^i)))^2$$

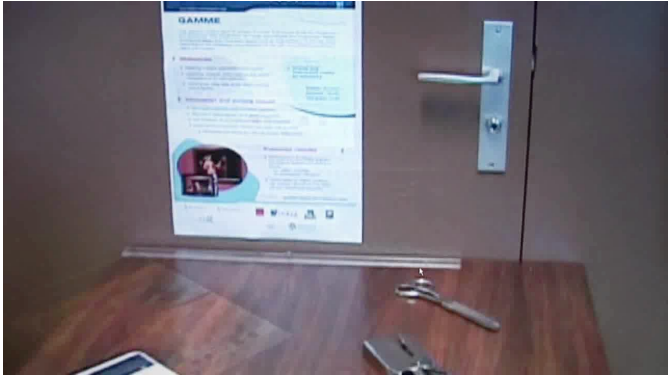
$$\mathbf{p}_2 = {}^2tr_1(\mathbf{p}_1) = {}^2\mathbf{H}_1\mathbf{p}_1$$

Direct estimation of the homography

[Benhimane Malis IJRR 07]







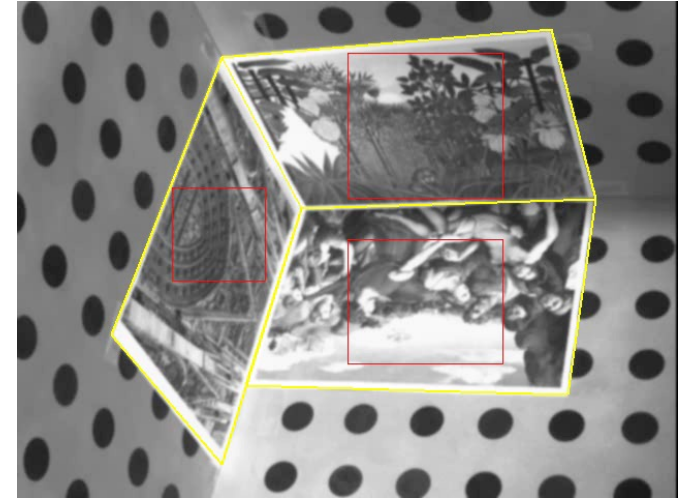
SLAM

Simultaneous Localization and Mapping

- On-line model construction
- 3D localization
- Recursive estimation process (EKF)

[Servant 07]

~ 1.5x original speed

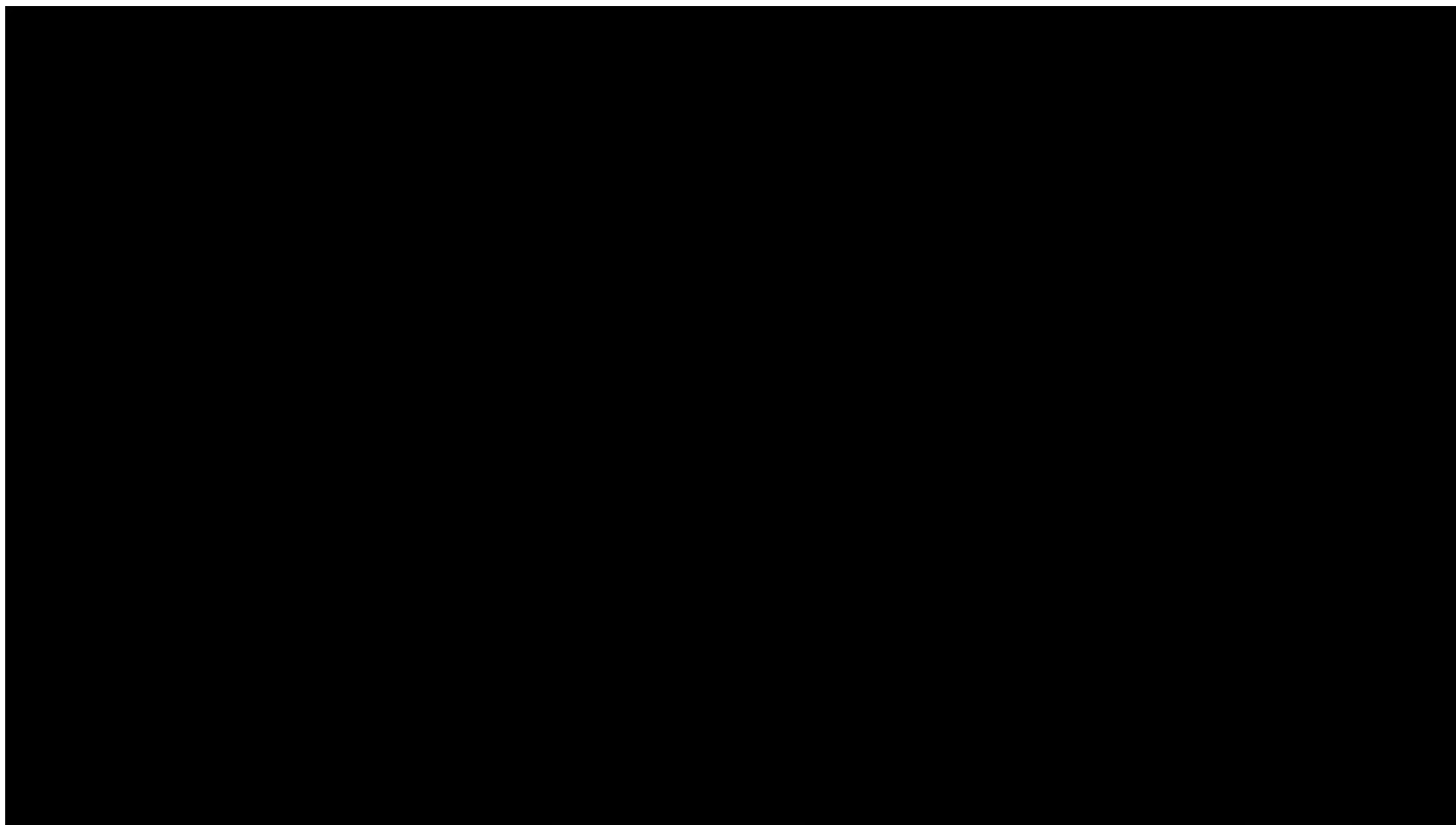


Parallel Tracking and Mapping for Small AR Workspaces

Extra video results made for
ISMAR 2007 conference

Georg Klein and David Murray
Active Vision Laboratory
University of Oxford





SLAM

This leads to matchmoving...

