Visual tracking

Eric Marchand

Lagadic group

Inria Rennes Bretagne Atlantique & Irisa

http://www.irisa.fr/lagadic

Visual tracking

Definition

- **Visual tracking** is the process of locating a moving object (or multiple objects) over time using a camera.
- Tracking then refers to a **localization** problem

Tracking can be done:

- In the image plane (2D)
- In the real world (3D)

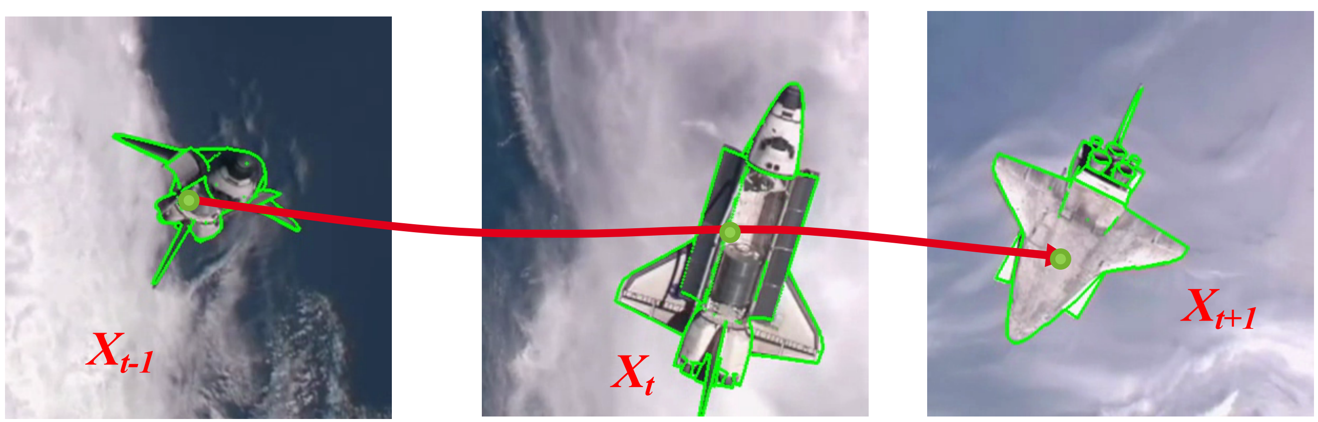
Objects

- Keypoints, geometrical features
- Image regions
- 3D objects

What is tracking ?

Tracking: A state estimation issue from image measurements

Using image measurements consistently estimate the state(s) $\mathbf{x}_t$ of one or more object(s) over the discrete time steps in a video



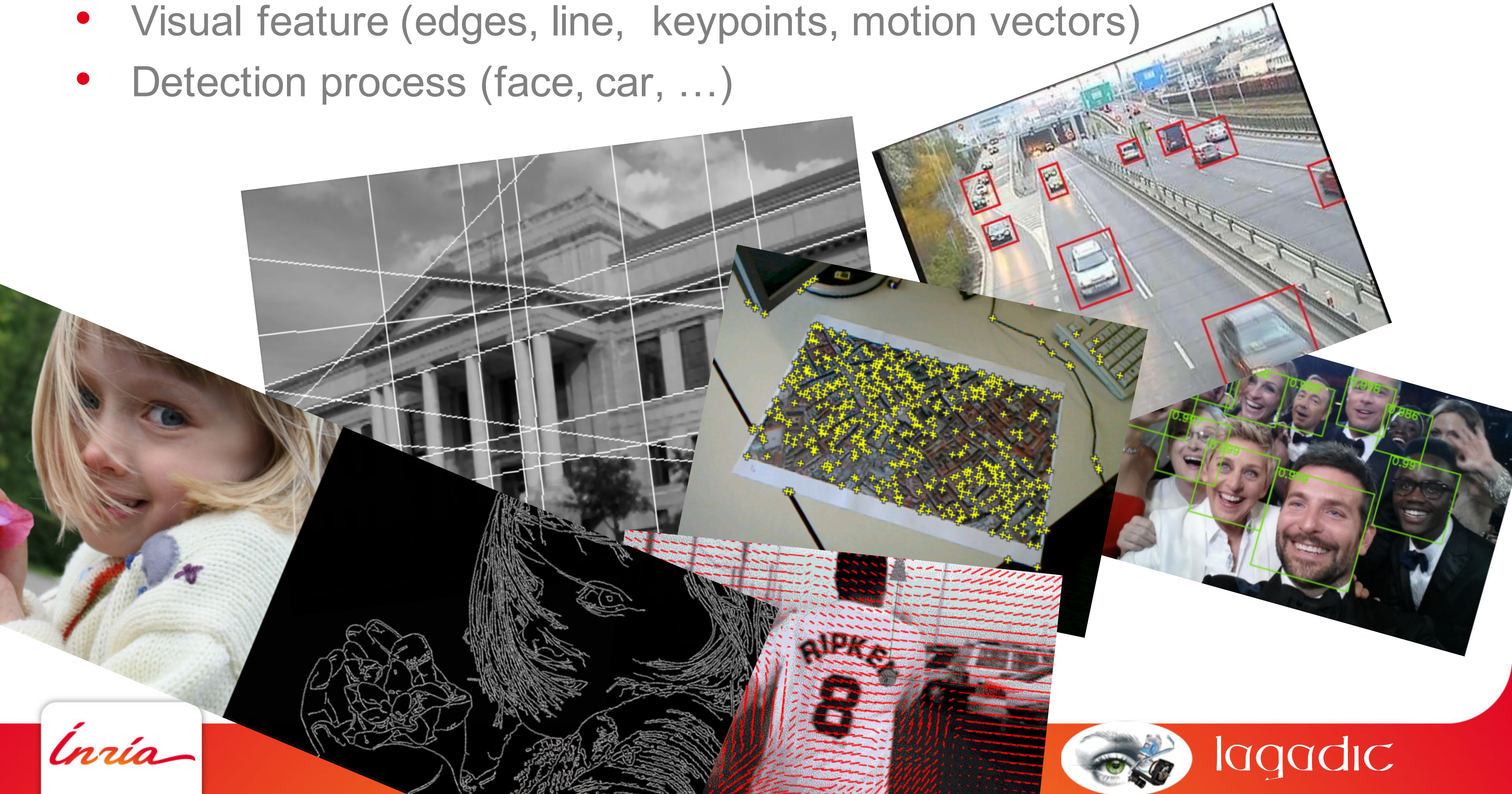
t-1　　　　*t*　　　　*t+1*

Measurements? state?

From image measurements to state estimation

Measurements:

- Pixel intensity (raw data), color
- Visual feature (edges, line, keypoints, motion vectors)
- Detection process (face, car, …)



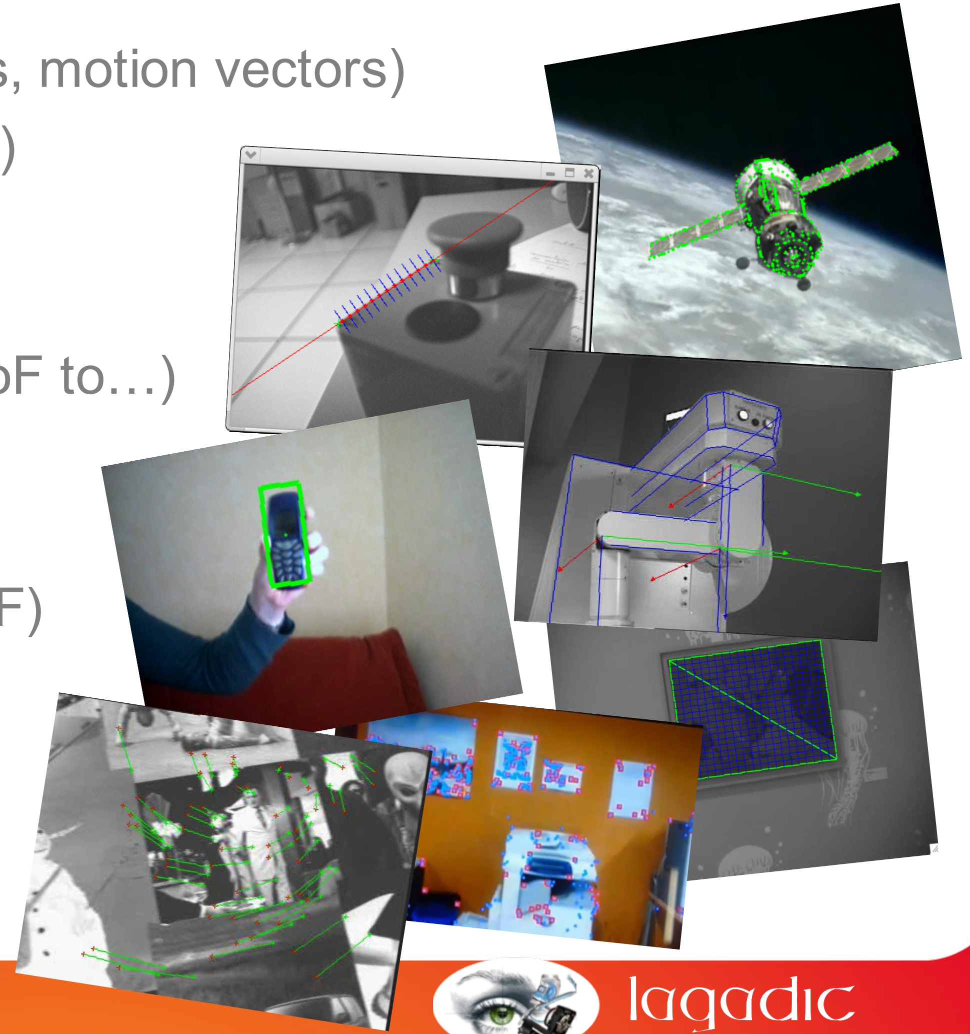
From image measurements to state estimation

Measurements:

- Pixel intensity (raw data), color
- Visual feature (edges, keypoints, motion vectors)
- Detection process (face, car, …)

State:

- Coordinates(2 DoF)
- Geometrical features (from 2 DoF to…)
- Bounding box (4-6 DoF)
- 3D rigid pose (6 DoF)
- 3D pose + deformation (6+k DoF)
- Homography (8 DoF)
- Visual SLAM (6N + M DoF)



Formalizing tracking

[Patrick Perez 2015]

Given past and current measurements $\mathbf{z}_{1:t} = (\mathbf{z}_1 \ldots \mathbf{z}_t)$

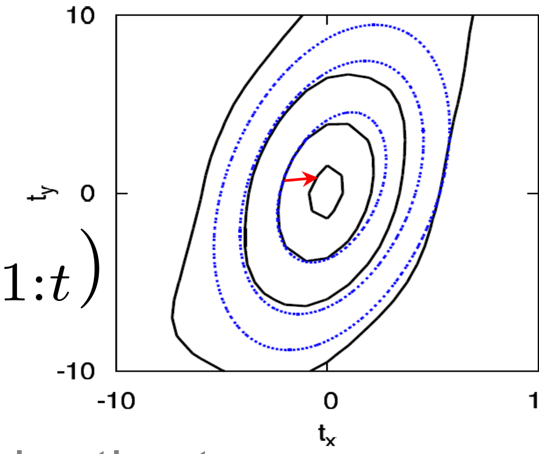
Output an estimate of current state

$$\widehat{\mathbf{x}}_t = f(\mathbf{z}_{1:t})$$

Deterministic tracking

- Optimization of ad-hoc objective function

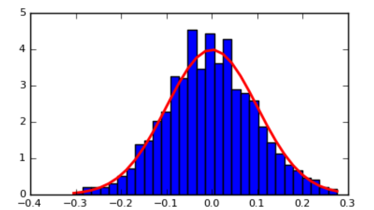
$$\widehat{\mathbf{x}}_t = \operatorname{argmin} \; E(\mathbf{x}_t, \widehat{\mathbf{x}}_{t-1}, \mathbf{z}_{1:t})$$



Probabilistic tracking

- Computation of the filtering pdf $p(\mathbf{x}_t \mid \mathbf{z}_{1:t})$ and estimate:

$$\widehat{\mathbf{x}}_t = \operatorname{argmax} \; p(\mathbf{x}_t \mid \mathbf{z}_{1:t})$$



I will not talk about… (but a few words)

vSLAM and RGB-D mapping (and then tracking)

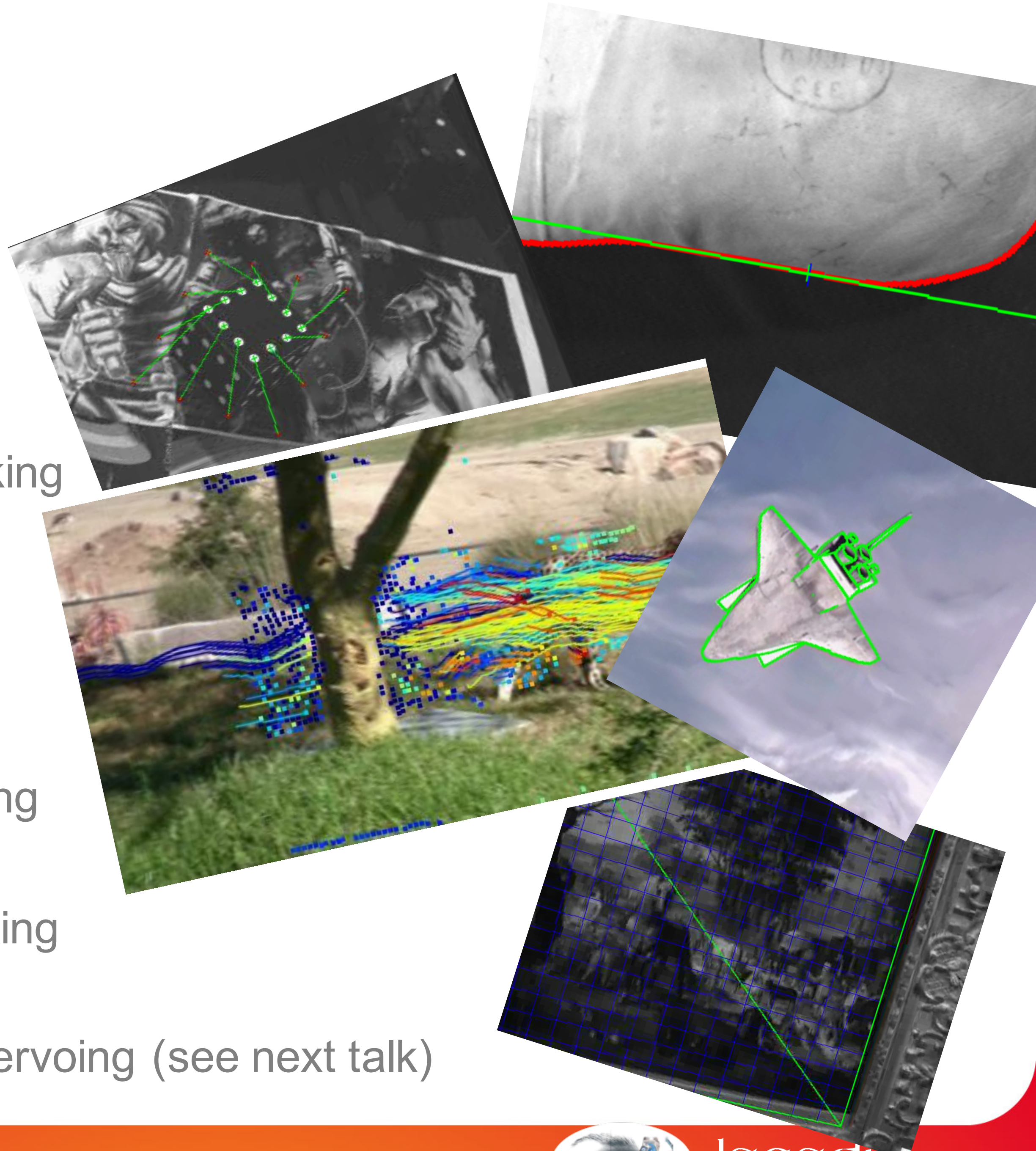
- It is clearly in the scope of a talk on visual tracking !
- Tutorial this afternoon 13:30 – 17:30 room A1

Probabilistic tracking

- Kalman filter, EKF, Particle filter
- All the presented approaches could take advantage of probabilistic filter

Overview

- 2D tracking
  - Fiducial markers
  - Contour-based tracking
  - Keypoints, KLT
  - Color tracking
- Motion estimation
  - Region based tracking
- 3D model-based tracking
- Application in visual servoing (see next talk)



Detection vs tracking

Is tracking only a detection and matching problem ?

Detection/matching:

- Estimate $\mathbf{x}_t$ for a given frame regardless of past frames
- Search over the whole image (may computationally be inefficient)

Tracking:

- Spatio-temporal issue maintain the estimate of $\mathbf{x}_t$ over time
- Restrict search space (may consider prediction process)
- Dynamic evolution model

Tracking may be achieved thanks to a detection/matching algorithm

Fiducial markers : still useful ?

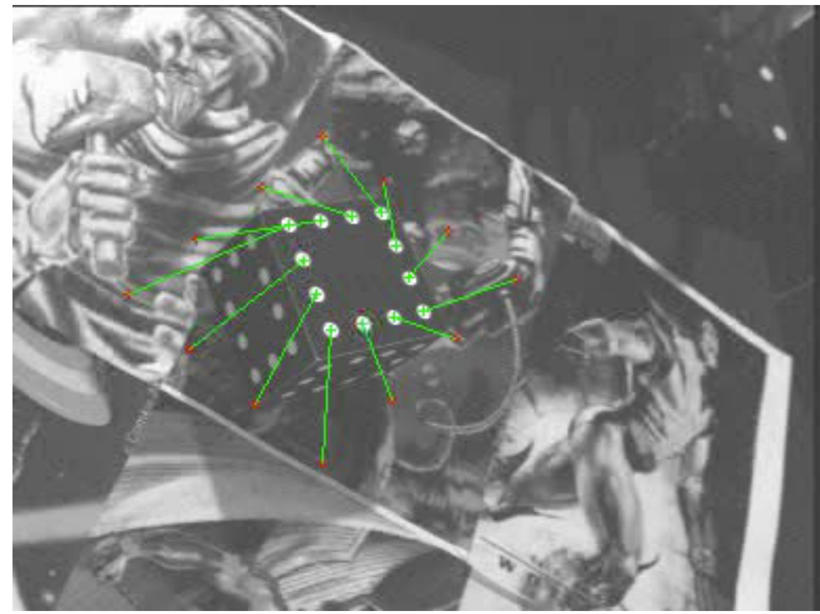
White dots on black background

So simple, yet efficient,…

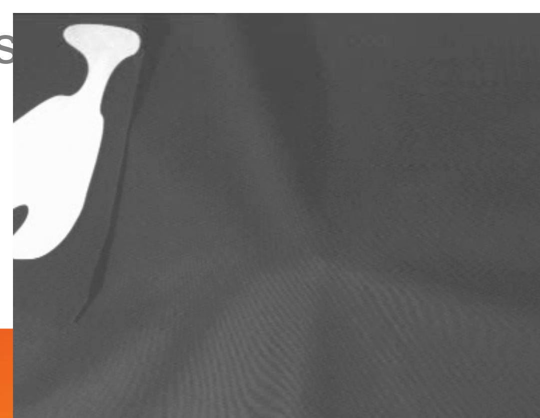
Just a connected component labeling

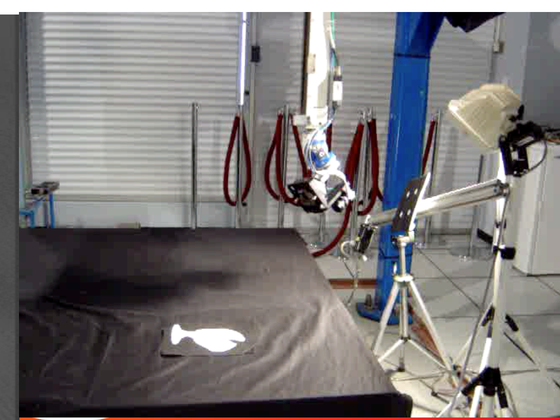
Still considered in research in visual servoing to test

- modeling aspects
- design of new control laws



Ready for industrial applications





Tracking contour-based 2D features

Local tracking of edge points

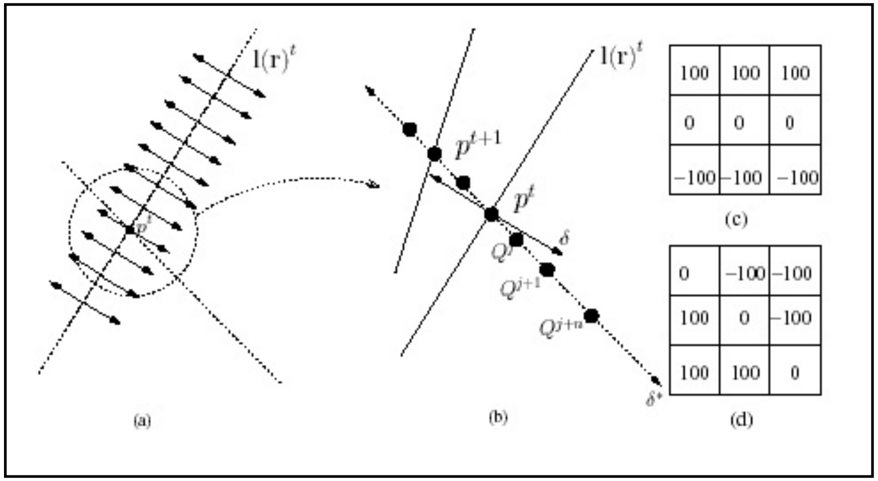
- Eg, ECM algorithm [Bouthemy PAMI 89]
- 1D search algorithm
- Convolution with oriented mask

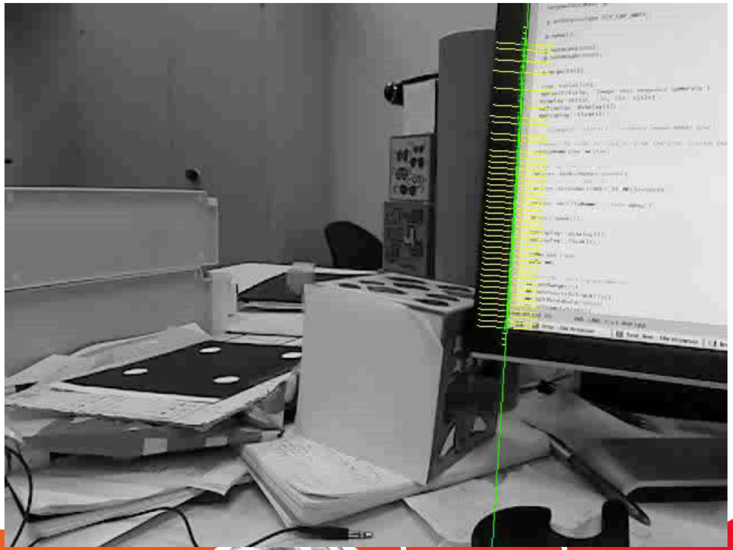
Robust estimation of feature parameters

- Lines, circles, splines, etc.
- Least square
- IRLS (M-estimation)

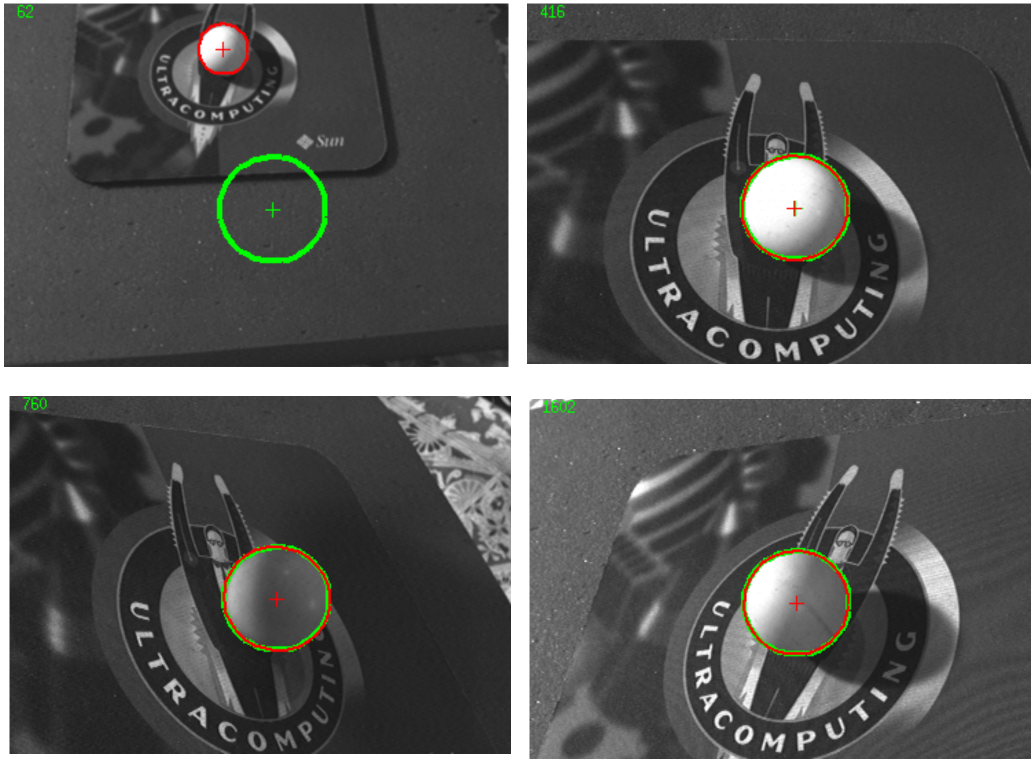
Frame rate performance

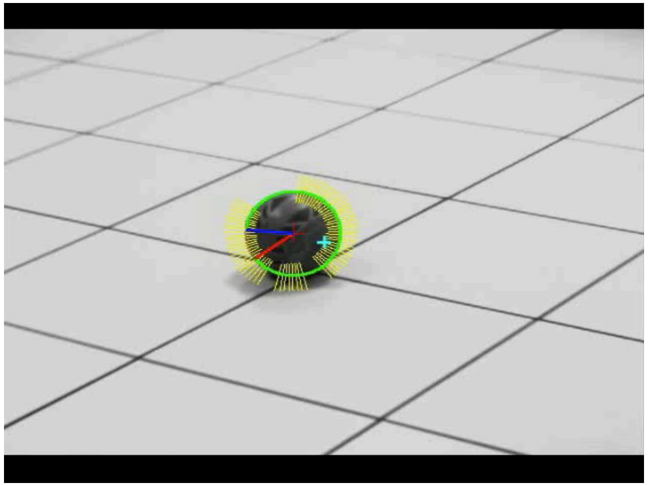
Source code in ViSP [Marchand IEEE RAM05]





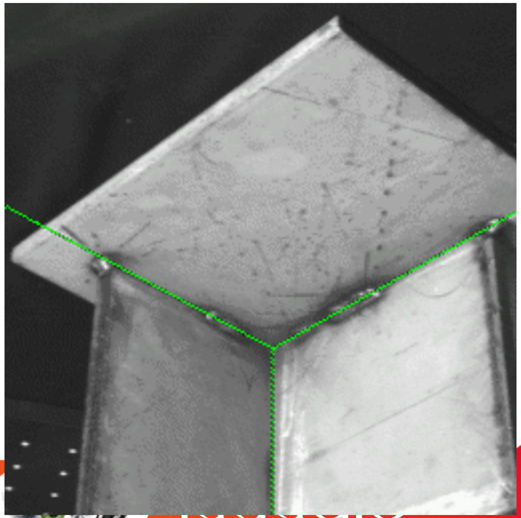
Tracking contour-based 2D features





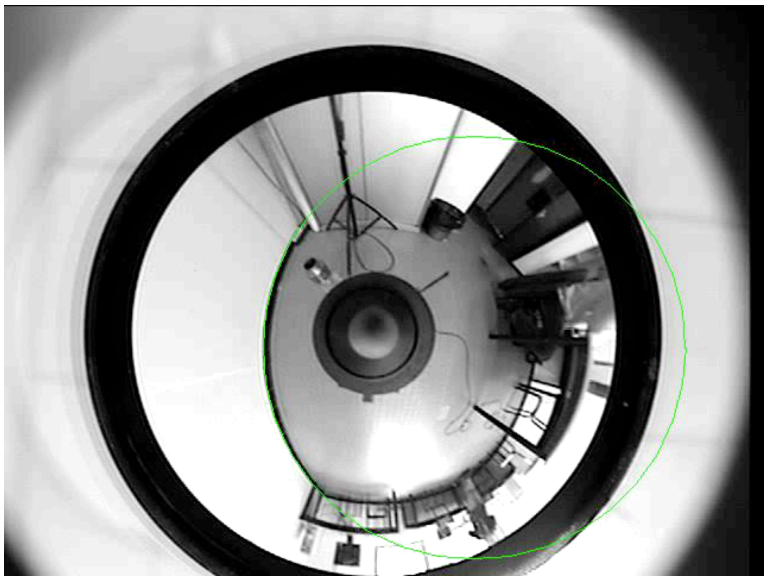
Tracking a set of features

[Andreff IJRR01]

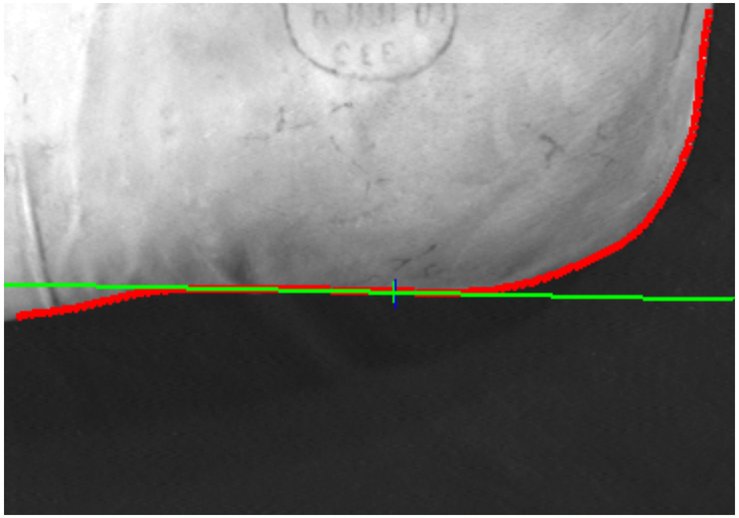


Tracking contour-based 2D features

Tracking in catadioptric images [Hadj-Abdelkader, LASMEA]



Contour following



Tracking points of interest: KLT

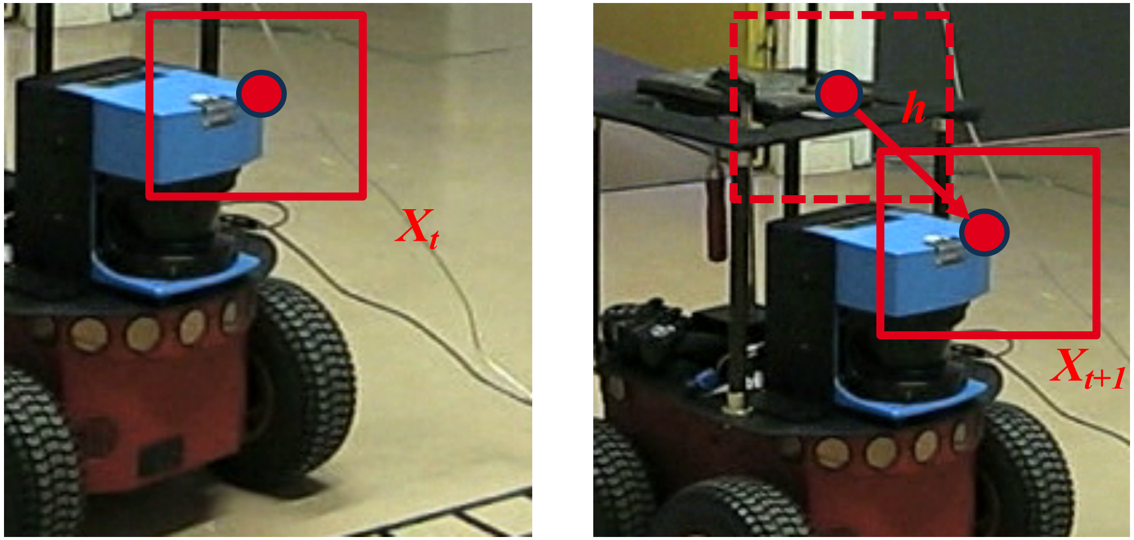
[Lucas Kanade 1981]

Point detection using Harris and Stephen detector

- Maximum of the signal autocorrelation matrix

Tracking using KLT algorithm

- Based on brightness consistency hypothesis $I_0(\mathbf{x}) = I(\mathbf{x} + \mathbf{h})$



Tracking points of interest: KLT

[Lucas Kanade 1981]

Point detection using Harris and Stephen detector

- Maximum of the signal autocorrelation matrix

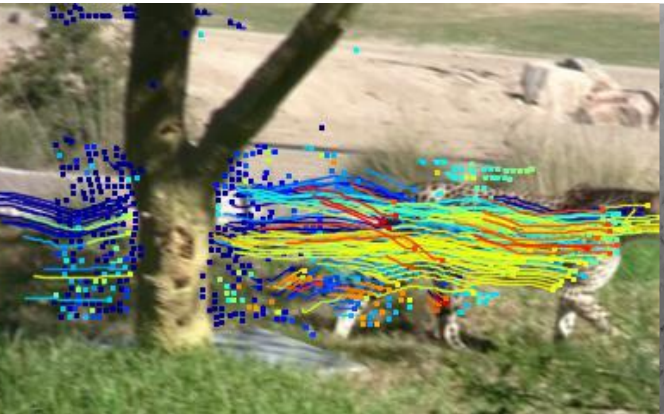
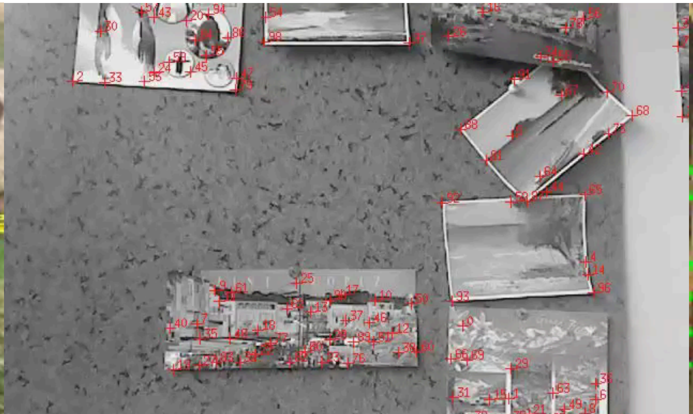
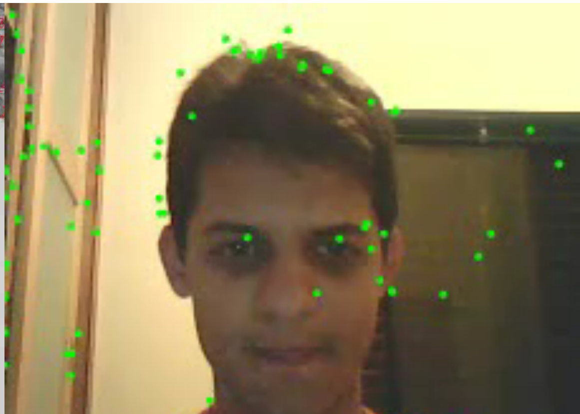
Tracking using KLT algorithm

- Based on brightness consistency hypothesis $I_0(\mathbf{x}) = I(\mathbf{x})$

SSD

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x}\in W} (I_0(\mathbf{x}) - I(\mathbf{x}+\mathbf{h}))^2$$

- **h** is the translation motion for a given patch
- Extended to more complex motion (see later) [Shi, CVPR 1994] [Baker IJCV 2004]

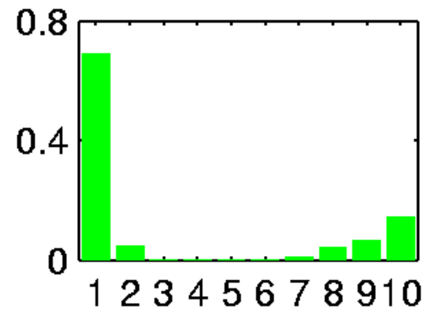
Color-based tracking

[Comaniciu PAMI 2003]

Global description of tracked region: color histogram

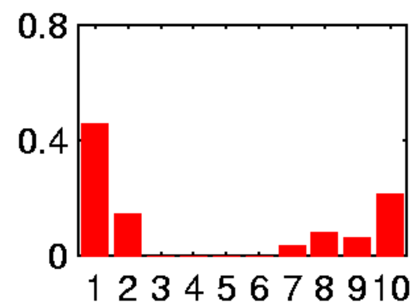
- Reference histogram with B bins

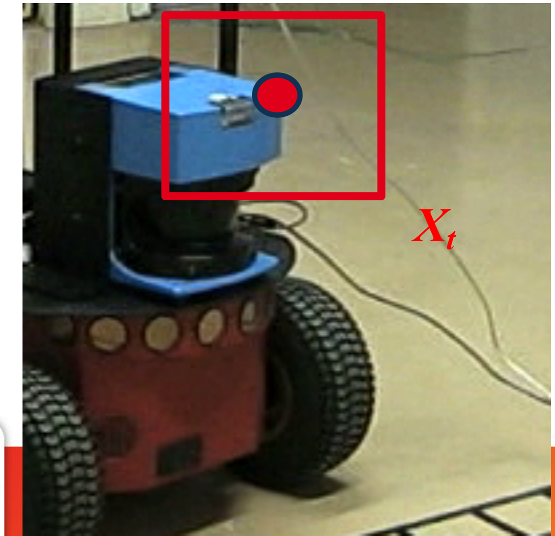
$$\mathbf{q}^* = \{q_i^*\}_{i=1..N}$$

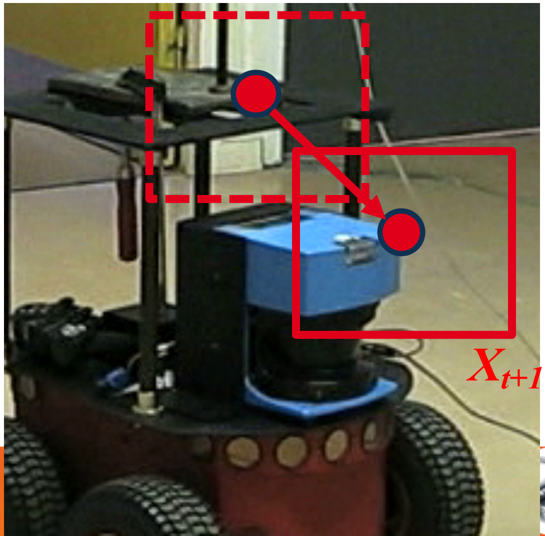


Candidate histogram at current instant

$$\mathbf{q}(\mathbf{x}) = \{q_i(\mathbf{x})\}_{i=1..N}$$







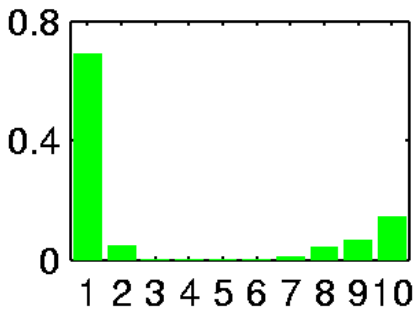
Color-based tracking

[Comaniciu PAMI 2003]

Global description of tracked region: color histogram

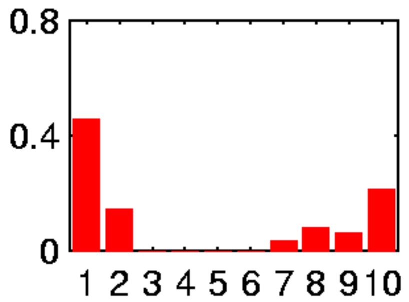
- Reference histogram with B bins

$$\mathbf{q}^* = \{q_i^*\}_{i=1..N}$$



Candidate histogram at current instant

$$\mathbf{q}(\mathbf{x}) = \{q_i(\mathbf{x})\}_{i=1..N}$$



At each instant

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} \quad dist(\mathbf{q}^*(\mathbf{x}), \mathbf{q}(\mathbf{x}+\mathbf{h}))$$

- iterative search: meanshift-like iteration
- Battacharyya measure

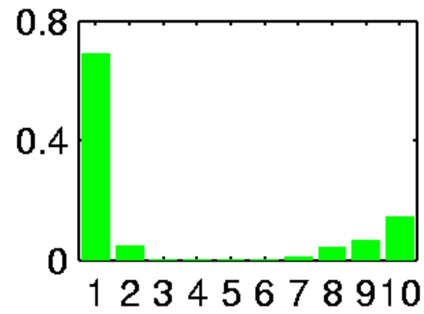
Color-based tracking

[Comaniciu PAMI 2003]

Global description of tracked region: color histogram

- Reference histogram with B bins

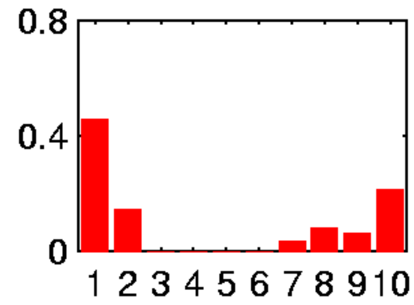
$$\mathbf{q}^* = \{q_i^*\}_{i=1..N}$$



set at track initialization

Candidate histogram at current instant

$$\mathbf{q}(\mathbf{x}) = \{q_i(\mathbf{x})\}_{i=1..N}$$

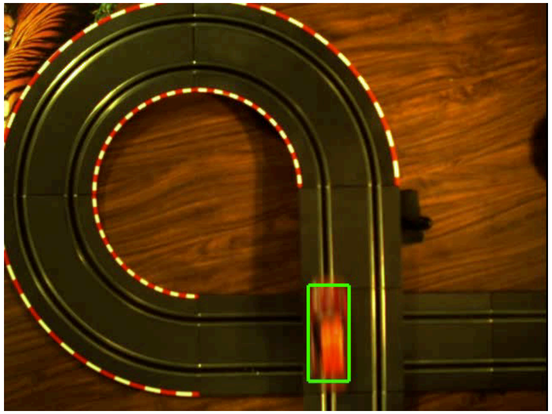


At each instant

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} \quad 1 - \sum_i \sqrt{\mathbf{q}_\mathbf{i}^*(\mathbf{x})\mathbf{q}_\mathbf{i}(\mathbf{x}+\mathbf{h})}$$

- iterative search: meanshift-like iteration
- Battacharyya measure

Color-based tracking



Remark

Up to now

- Estimated state is composed of 2D information
- Restricted number of DoF
- Need many features/tracker and further estimation to provide 3D displacement or 3D object position

An issue

- There does not exist a 2D transformation that can account for 3D object motion

Two alternatives (among others)

- Homography estimation / planar constraints
- Model-based tracking / 3D shape prior
- SLAM (later this afternoon)

From basic image features to 3D information

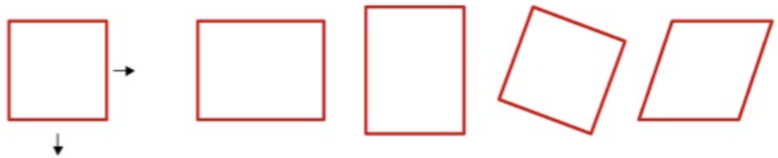
A partial solution to:

"Need many features/trackers and further estimation to provide 3D displacement or 3D object position"

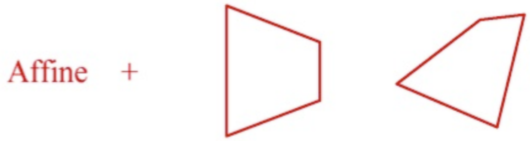
From basic image features to motion

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x}\in W} (\mathbf{x}_0 - w(\mathbf{x},h))^2$$

- ***w*(x,h)** is a 2D motion model (warp function)
  - Translation, Rt, sRt
  - Affine motion model



  - Homography



Short reminder: homography

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

$$\left.\begin{array}{l} {}^1\tilde{\mathbf{X}} \in \mathcal{P}({}^1\mathbf{n}, {}^1d) \Leftrightarrow {}^1\mathbf{n}^\top {}^1\tilde{\mathbf{X}} = {}^1d \\ {}^2\tilde{\mathbf{X}} = {}^2\mathbf{R_1} {}^1\tilde{\mathbf{X}} + {}^2\mathbf{t_1} \end{array}\right\} \Rightarrow {}^2\tilde{\mathbf{X}} = {}^2\mathbf{R_1} {}^1\tilde{\mathbf{X}} + {}^1\mathbf{t_2} \frac{{}^1\mathbf{n}^\mathbf{T}}{{}^1d} {}^1\tilde{\mathbf{X}}$$

and

$$Z_1 \mathbf{x_1} = {}^1\tilde{\mathbf{X}} \qquad \text{and} \qquad Z_2 \mathbf{x_2} = {}^2\tilde{\mathbf{X}}$$

leading to

$$\lambda \mathbf{x_2} = {}^2\mathbf{H}_1 \mathbf{x_1}$$

with

$${}^2\mathbf{H_1} = {}^2\mathbf{R_1} + \frac{{}^1\mathbf{n}^\top}{{}^1d} {}^2\mathbf{t}_1 \qquad \text{and} \quad \lambda = \frac{Z_2}{Z_1}$$

Note that a homography integrates information about the camera displacement

Homography estimation is a linear problem

For each point we have (in homogeneous coordinates) :

$$\mathbf{x_2} = {}^{\mathbf{2}}\mathbf{H_1}\mathbf{x_1}$$

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

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

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

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

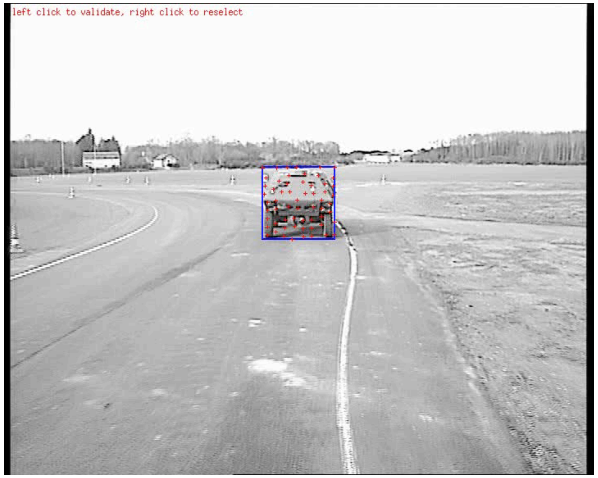
it can be solved using an SVD decomposition

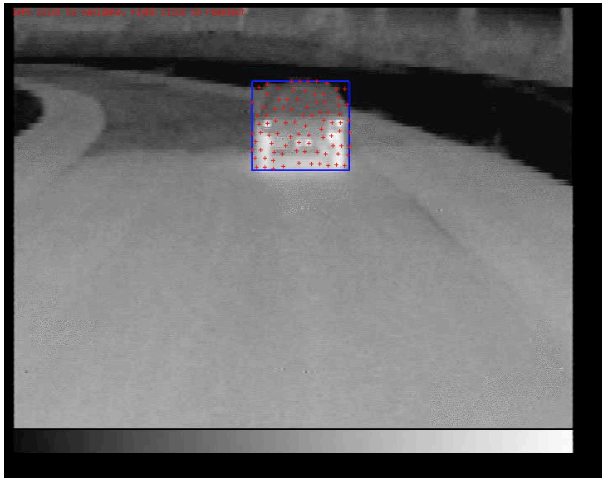
$$\Gamma = \mathbf{UDV}^\top$$

h is the vector of **V** associated with the smallest singular value of Γ

Motion estimation from keypoints

- Harris points extracted on a selected template
- points tracked using a KLT-like method
- statistically robust method to get a coherent global motion model






From keypoints tracking/matching to 3D tracking

[Berger, Simon IEEE CGA02]

[Lepetit]

A remark:

Always use robust estimation

RANSAC is perfect here

Appearance based tracking

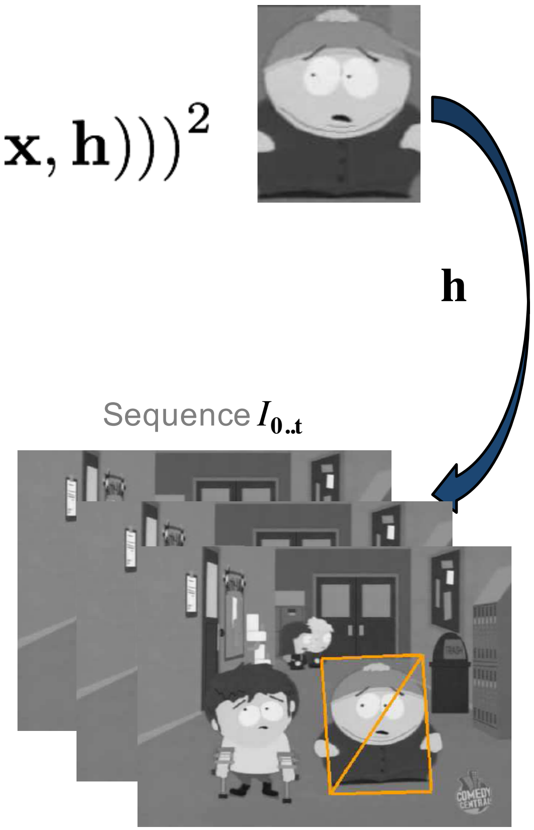
Appearance based tracking

Extention of the KLT approach

- Based on brightness consistency hypothesis

$$\hat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x} \in W} \left(I_0(\mathbf{x}) - I(w(\mathbf{x}, \mathbf{h})) \right)^2$$

- **w(x,h)** is a 2D motion model (warp function)



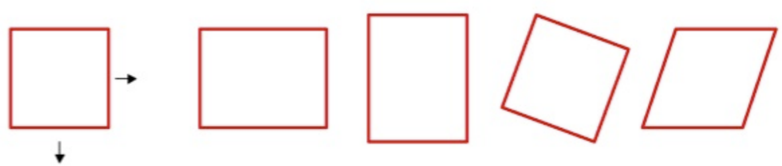
Appearance based tracking

Extention of the KLT approach

- Based on brightness consistency hypothesis

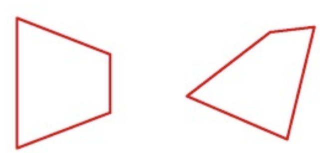
$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x}\in W} \left(I_0(\mathbf{x}) - I(w(\mathbf{x},\mathbf{h}))\right)^2$$

- ***w*(x,h)** is a 2D motion model (warp function)
  - Translation (KLT), Rt, sRt
  - Affine motion model [Shi CVPR 1994,Baker IJCV 2004, Hager PAMI 1998]



  - Homography [Malis IROS 2004] (ESM minimization process)

Affine +



Algorithm Overview

- Principe :

  Small dispalcement between 2 successive images

- Effect:

  $\mathbf{h}_{t-1}$ is close from $\mathbf{h}_t$.

- Algorithm:

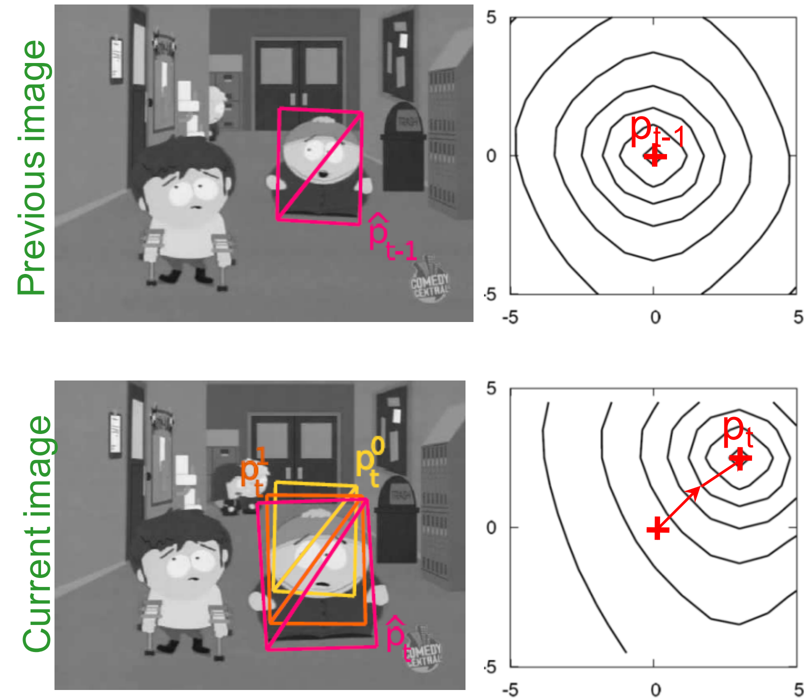
  $\mathbf{h}_{t-1}$ is iteratively adapted to estimate $\mathbf{h}_t$.

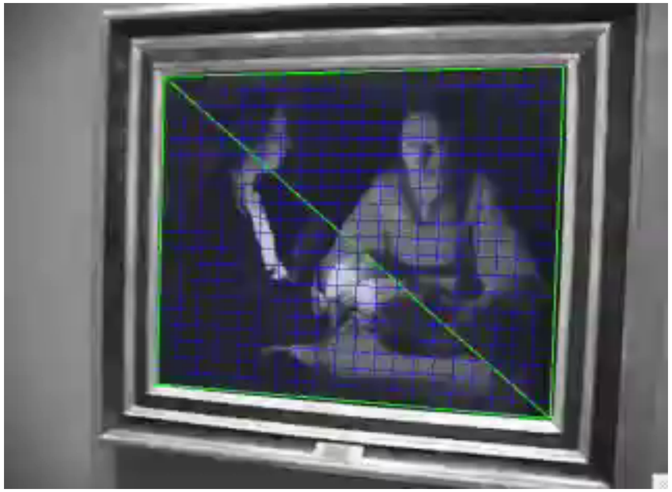
  Iterative processitératif:
  - Initialization :
  - loop:

$$\mathbf{p}_t^0 = \widehat{\mathbf{p}_{t-1}}$$

$$\Delta\mathbf{p}^k = \arg\max_{\Delta\mathbf{p}} f\left(I^*(\mathbf{x}), I(w(w(\mathbf{x}, \Delta\mathbf{p}), \mathbf{p}^k))\right)$$

$$w(\mathbf{x}, \mathbf{p}^{k+1}) \leftarrow w(w(\mathbf{x}, \Delta\mathbf{p}^k), \mathbf{p}^k)$$





KLT and template tracking [Lucas Kanade 1981, Baker IJCV 04]

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} \; C(\mathbf{h}) \text{ with } \quad C(\mathbf{h}) = \sum_{\mathbf{x} \in W} \left(I_0(\mathbf{x}) - I(w(\mathbf{x}, \mathbf{h}))\right)^2$$

For each pixel a first order Taylor extension of $c(\mathbf{h}) = I(w(\mathbf{x}, \mathbf{h})) - I_0(\mathbf{x})$

$$\begin{aligned} c(\mathbf{x}, \mathbf{h} + \delta\mathbf{h}) &= I(w(\mathbf{x}, \mathbf{h})) + \frac{\partial I(w(\mathbf{x}, \mathbf{h}))}{\partial \mathbf{h}} \delta\mathbf{h} - I_0(\mathbf{x}) + O(\delta\mathbf{h}) \\ &\approx I(w(\mathbf{x}, \mathbf{h})) + \nabla I \frac{\partial w(\mathbf{x}, \mathbf{h})}{\partial \mathbf{h}} \delta\mathbf{h} - I_0(\mathbf{x}) \end{aligned}$$

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) \text{ with } \quad C(\mathbf{h}) = \sum_{\mathbf{x}\in W} \left(I_0(\mathbf{x}) - I(w(\mathbf{x},\mathbf{h}))\right)^2$$

If we now consider all the pixel (vector notation)

$$C(\mathbf{h}) = \mathbf{c}(\mathbf{h})^\top \mathbf{c}(\mathbf{h}), \text{ with } \quad \mathbf{c}(\mathbf{h}) = \mathbf{I}_t(w(\mathbf{h})) - \mathbf{I}_0$$

With

$$\begin{aligned} \mathbf{I}_0 &= (\ldots, I_0(\mathbf{x}), \ldots)^\top \\ \mathbf{I}_t(w(\mathbf{h})) &= (\ldots, I_t(w(\mathbf{x},\mathbf{h})), \ldots)^\top \end{aligned}$$

In this case the linearization of **c(h)** is

$$\mathbf{c}(\mathbf{h} + \delta\mathbf{h}) = \mathbf{I}_t(w(\mathbf{h})) + \mathbf{J}(\mathbf{h})\delta\mathbf{h} - \mathbf{I}_0$$

With the Jacobian given by $\mathbf{J}(\mathbf{h}) = \left(\ldots, \nabla I \frac{\partial w(\mathbf{x},\mathbf{h})}{\partial \mathbf{h}}, \ldots\right)^\top$

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) \text{ with } \quad C(\mathbf{h}) = \sum_{\mathbf{x}\in W} (I_0(\mathbf{x}) - I(w(\mathbf{x},\mathbf{h})))^2$$

If we now consider all the pixel (vector notation)

$$C(\mathbf{h}) = \mathbf{c}(\mathbf{h})^\top \mathbf{c}(\mathbf{h}), \text{ with } \quad \mathbf{c}(\mathbf{h}) = \mathbf{I}_t(w(\mathbf{h})) - \mathbf{I}_0$$

With the Gauss-Newton method the solution consists in minimizing $C(\mathbf{h}+\delta\mathbf{h})$ where:

$$C(\mathbf{h}+\delta\mathbf{h}) = \|\mathbf{c}(\mathbf{h}+\delta\mathbf{h})\| \approx \|\mathbf{c}(\mathbf{h}) + \mathbf{J}(\mathbf{h})\delta\mathbf{h}\|$$

This minimization problem can be solved by an iterative least square approach and we have

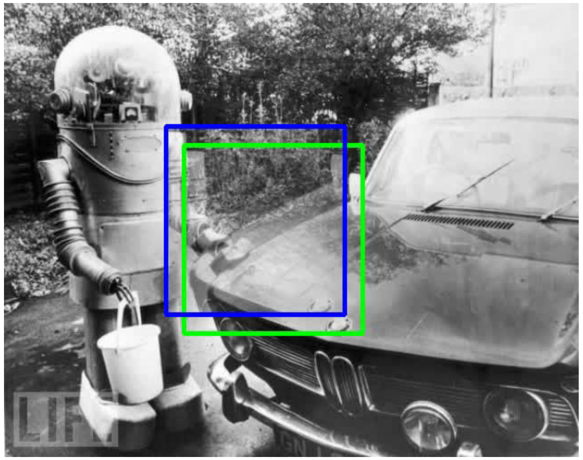
$$\delta\mathbf{h} = -\mathbf{J}(\mathbf{h})^+ (\mathbf{I}_t(w(\mathbf{h})) - \mathbf{I}_0)$$

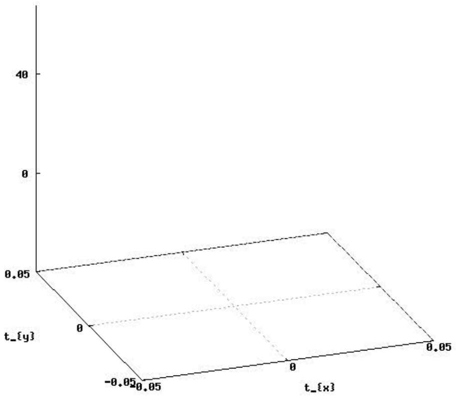
Appearance based tracking

Large patch tracking

- Homography suitable for planar object, rotating camera

Is SSD suitable for large pacth





- Mostly yes…
- But, it is not robust to illumination variations, blur (and then to fast motion), multi-modality

Appearance based tracking

Large patch tracking

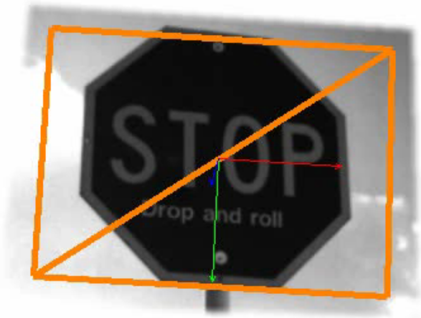
- Suitable for planar object, rotating camera

Is SSD suitable for large pacth

- Mainly yes… but…
- No robust to illumination variations, blur (and then to fast motion), multi-modality

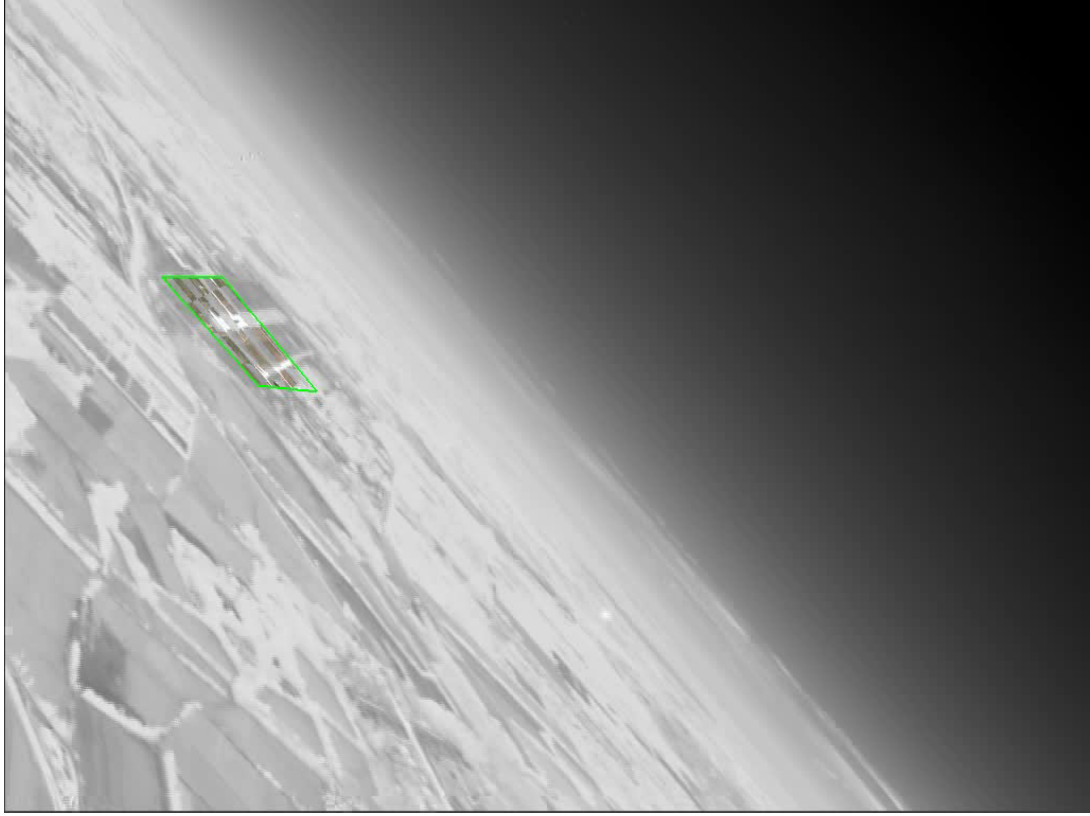
Other registration function

- ZNCC
- SCV [Richa Hager IROS 2011]
- Mutual information [Dame IEEE IP 2010]
- …

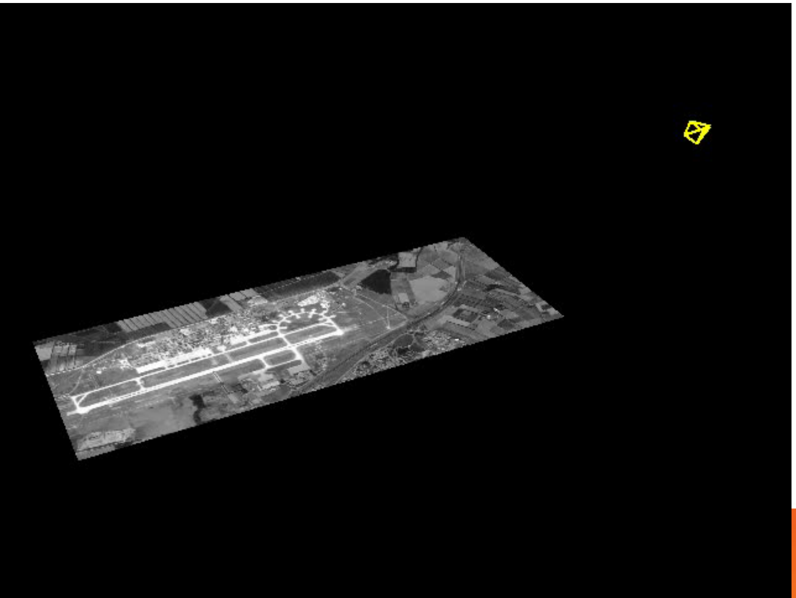


Reference image

Current image



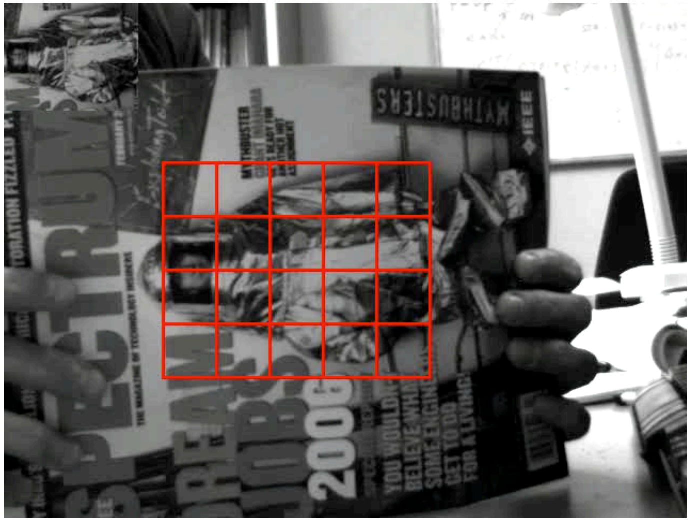
Localization



Generalization

Other parameterized transformation, eg, [Malis IROS 2007]

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x}\in W} \left(I_0(\mathbf{x}) - I(w(\mathbf{x},\mathbf{h}))\right)^2$$



Generalization

May include light variation [Silveira-Malis CVPR07]

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x} \in W} \left(I_0(\mathbf{x}) - aI(w(\mathbf{x}, \mathbf{h}) + b)\right)^2$$

Optimization: inverse compositional

Direct formulation

$$\begin{aligned}\mathbf{\Delta p}^k &= \arg\max_{\mathbf{\Delta p}} f\left(I^*(\mathbf{x}), I(w(w(\mathbf{x}, \mathbf{\Delta p}), \mathbf{p}^k))\right)\\ w(\mathbf{x}, \mathbf{p}^{k+1}) &\leftarrow w(w(\mathbf{x}, \mathbf{\Delta p}^k), \mathbf{p}^k)\end{aligned}$$

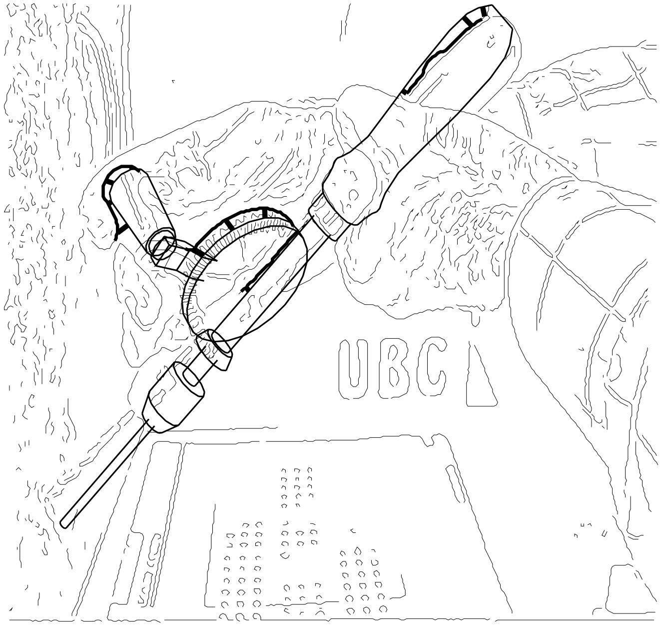
Idea: inverse current and template

$$\begin{aligned}\mathbf{\Delta p}^k &= \arg\max_{\mathbf{\Delta p}} \mathrm{MI}\left(I^*(w(\mathbf{x}, \mathbf{\Delta p})), I(w(\mathbf{x}, \mathbf{p}^k))\right)\\ w(\mathbf{x}, \mathbf{p}^{k+1}) &\leftarrow w(w^{-1}(\mathbf{x}, \mathbf{\Delta p}^k), \mathbf{p}^k)\end{aligned}$$

Main advantage:

- Many terms are precomputed (Jacobian which is huge)
- Almost equivalent convergence properties

Model-based trackers



[Lowe PAMI 91]

3D model-based tracking

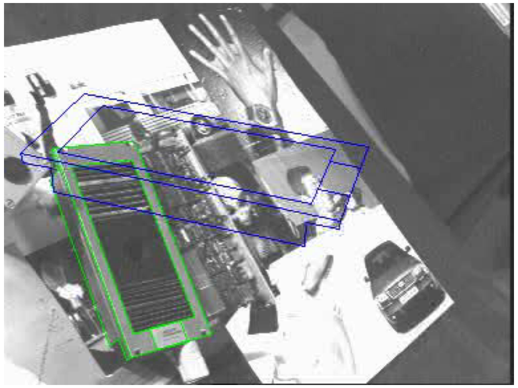
Tracking is handled through pose estimation

- Small object/camera displacement between two frames

Pose computation by minimizing the error between the projection of the CAD model and the image contours

Efficient tracking

- even with occlusions
- video rate (50Hz).

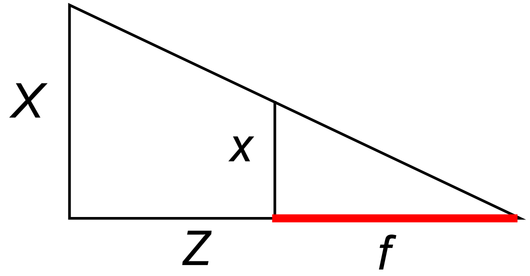


Pose estimation: basic problem

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

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

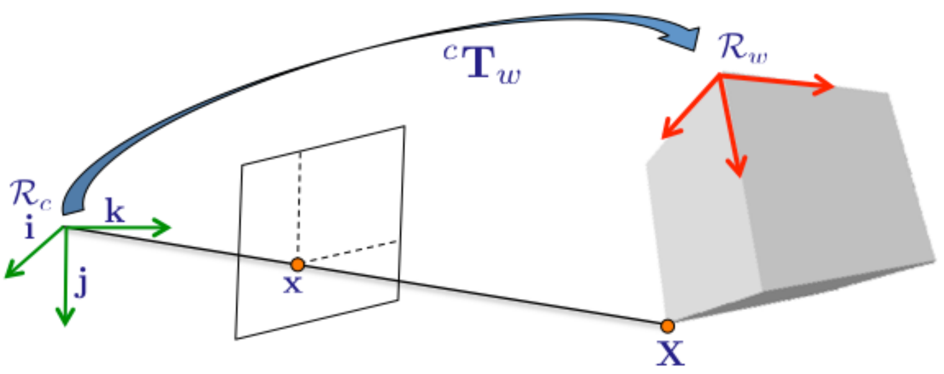
Solution is quite simple : change frame first



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

Then project

$$
\left\{ \begin{array}{l} x = \dfrac{(\mathbf{r}_1 \;\; 0)^w\mathbf{X} + t_x}{(\mathbf{r}_3 \;\; 0)^w\mathbf{X} + t_z} \\ \\ y = \dfrac{(\mathbf{r}_2 \;\; 0)^w\mathbf{X} + t_y}{(\mathbf{r}_3 \;\; 0)^w\mathbf{X} + t_z} \end{array} \right.
$$



Pose estimation

This problem is known as PnP

Many solution exists

- P3P [Fishler CACM 83] (introducing RANSAC) to [Kneip CVPR11]
- PnP (Direct Linear Transform, POSIT [Dementhon IJCV 95], EPnP [Lepetit]

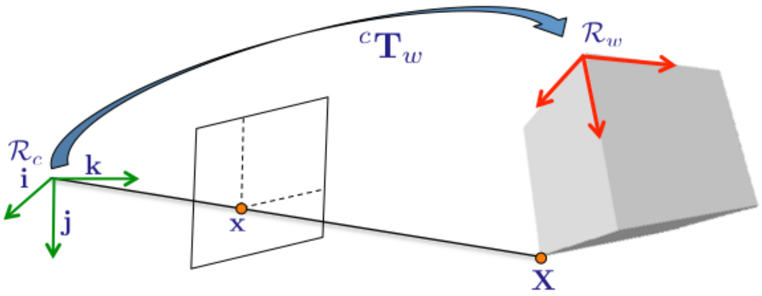
Most of them can be found in openCV, ViSP, openGV, etc…

A gold standard solution: non-linear minimization of the reprojection error

Pose estimation: the “gold-standard” solution

Goal

- Estimate the pose ${}^{c}\mathbf{T}_{w}$ 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

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

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

- $\mathbf{q}$ is a minimal representation of ${}^{c}\mathbf{T}_{w}$

Pose: non linear minimization

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

consists in minimizing the error **e**(**q**) defined by:

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

with

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

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

Linearization of the non-linear system

Problem: no general method to solve **e**(**q**) = 0

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

First order Taylor expansion around q

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

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

Computation if the Jacobian can be find in, eg, [Marchand IEEE TVCG 2016]

Solving the linearized system

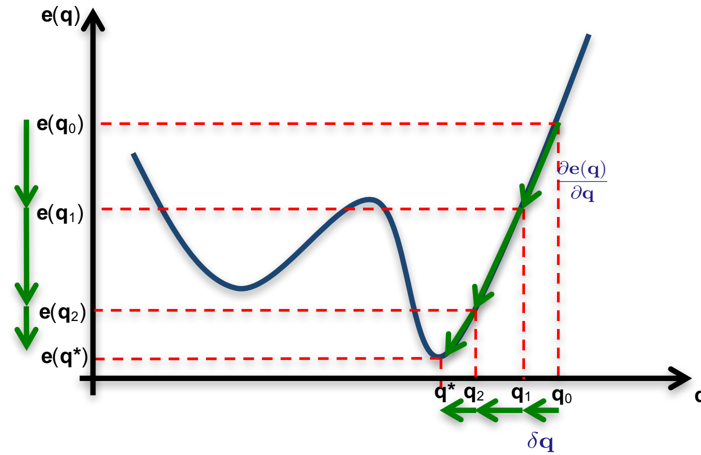
With the Gauss-Newton method, we do no want to determine the value of **q** that ensures **e**(**q**)=0 but the value that minimizes the cost function:

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

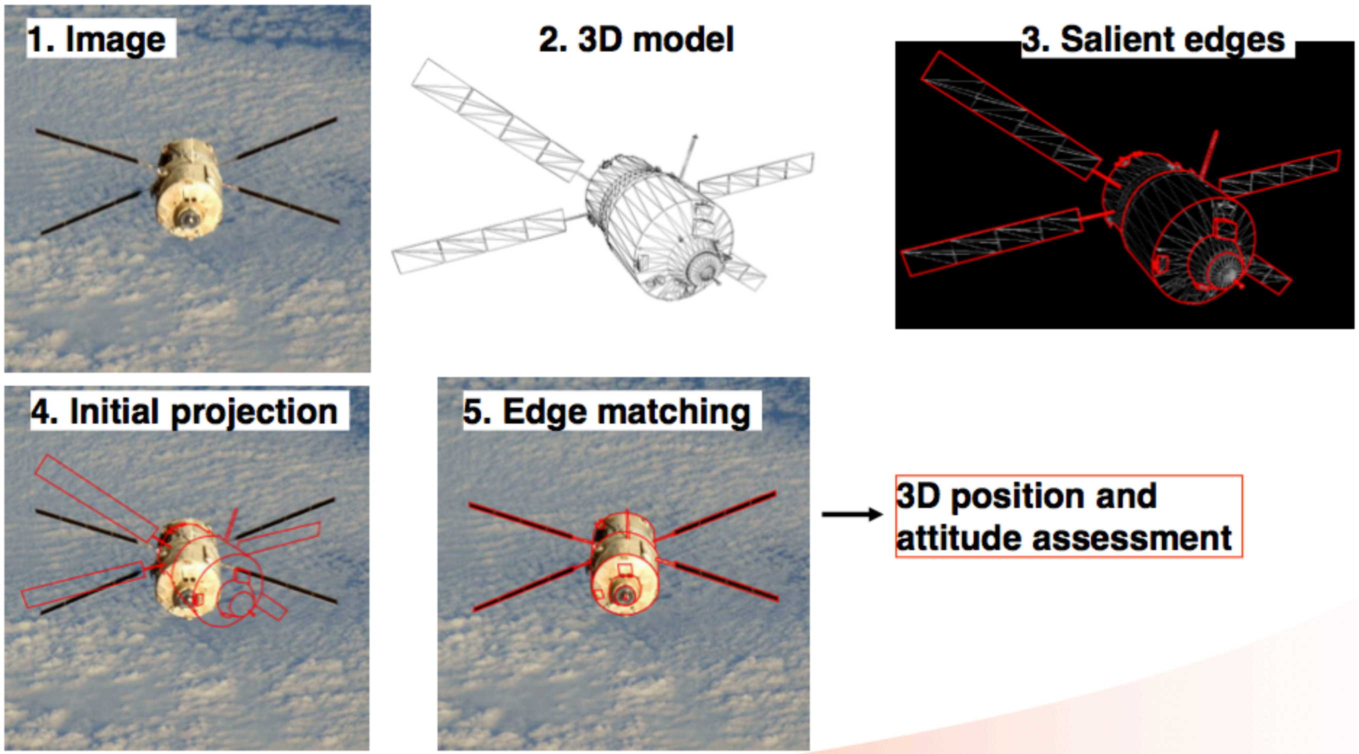
This is a linear minimization problem (solved by a least-square approach) and we have: $\delta\mathbf{q} = -\mathbf{J}(\mathbf{q})^{+}\mathbf{e}(\mathbf{q})$

Solved by an iterative least square method

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



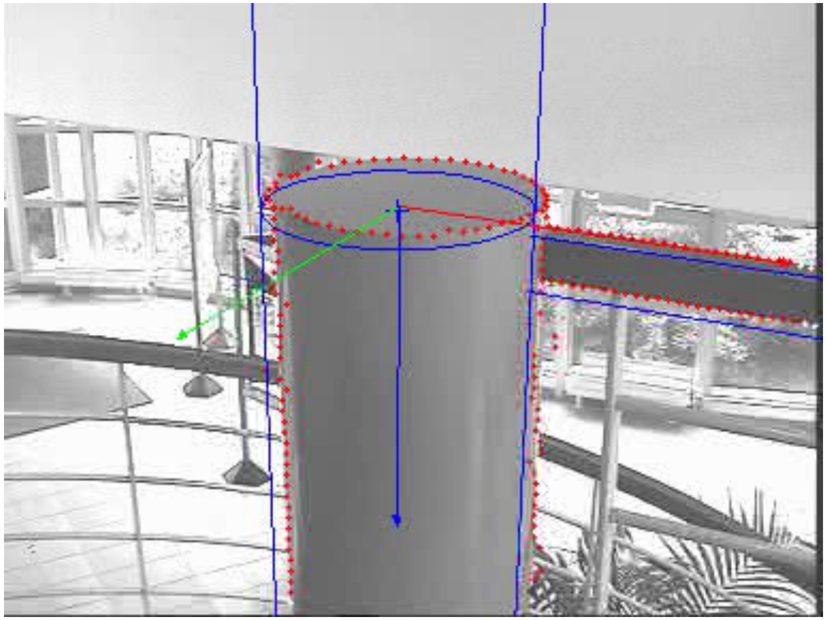
Beyond points



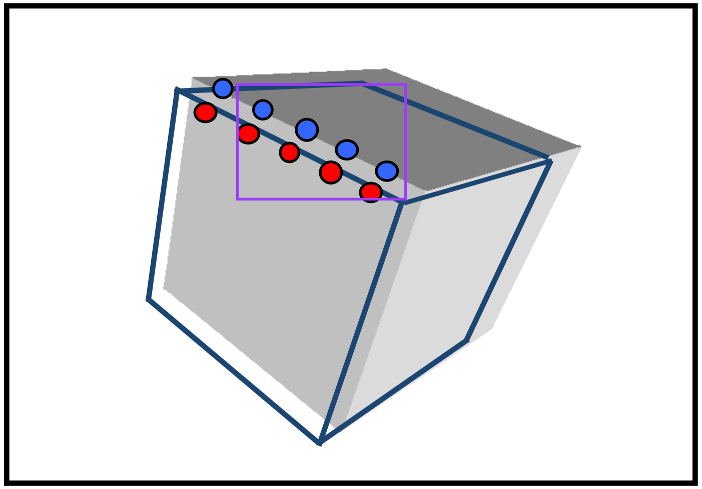
Beyond points

Distance to a moving line

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



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

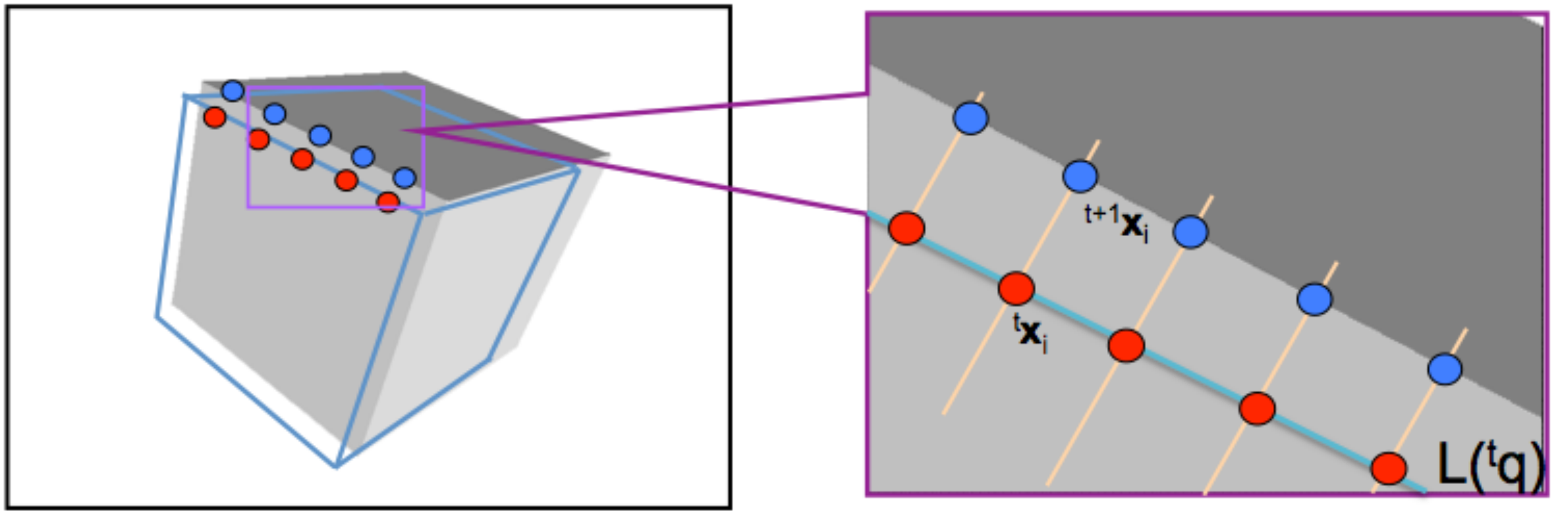


Beyond points

Distance to a moving line

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

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

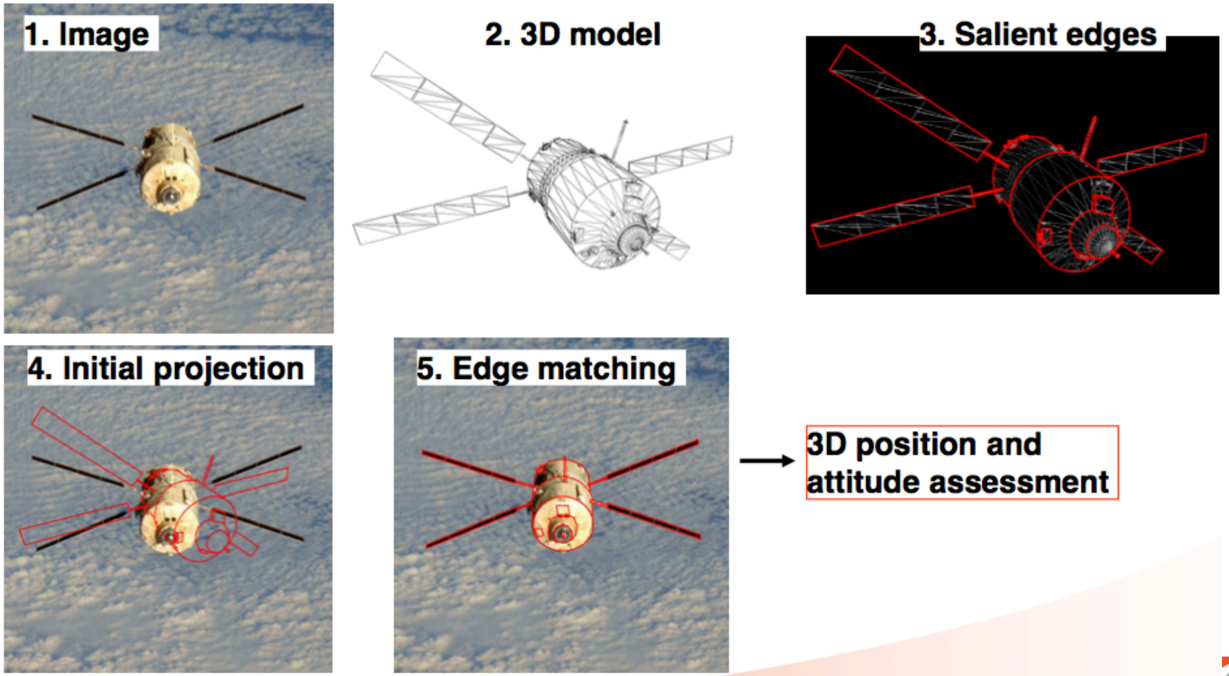


Markerless tracking

Similar approach but point to contour distance

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

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



Pose estimation: robustness to outliers

The residue is given by:

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

- where ×is a robust function (M-estimation)
- Minimize

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

Tukey's M-estimator

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

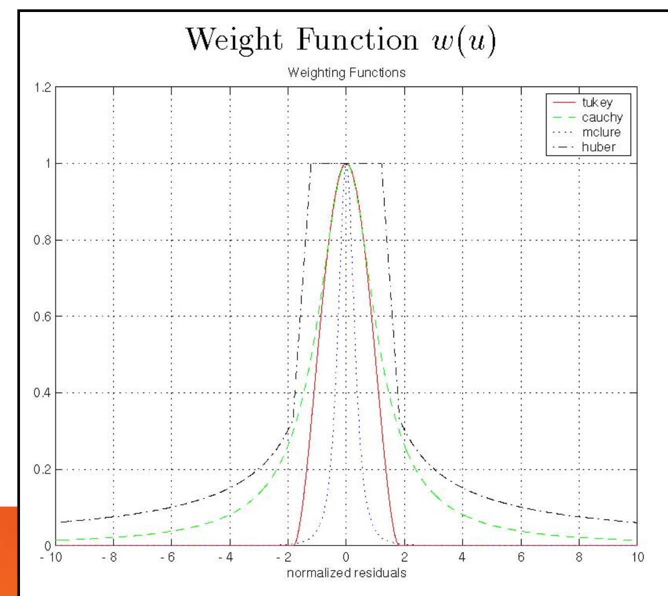
$$\psi(u) = \begin{cases} u(C^2 - u^2)^2 & |u| \leq C \\ 0 & \text{else} \end{cases}$$

The control law, similar to an IRLS, which minimizes x-x(q) is given by

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

where

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

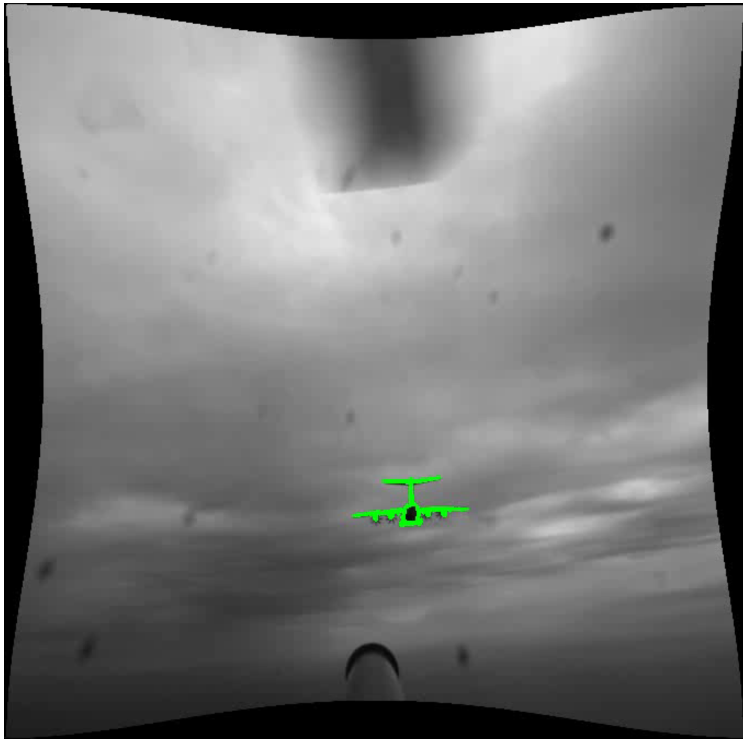


Model-based trackers

Such approach is quite efficient

It did the job pretty well

At video rate [Petit ICRA 2014]



Spatial applications

Manoeuvre Atlantis /ISS

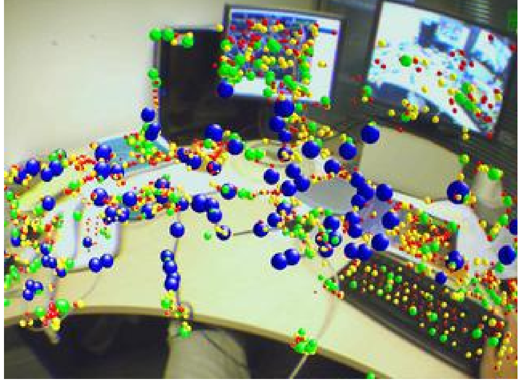
Rendez-vous Soyuz/ISS

EADS astrium

Inria

lagadic

Trends: vSLAM



See Tutorial on SLAM this afternoon

But the idea is to estimate both the pose (current and past) along with scene 3D structure

$$([\widehat{\mathbf{q}}]_t, \quad [\widehat{{}^w\mathbf{X}}]_N) = \arg\min_{([\mathbf{q}]_t, [{}^w\mathbf{X}]_N)} \sum_{j=1}^{t} \sum_{i=1}^{N} d\left(\mathbf{x}_{j_i}, \Pi\, {}^j\mathbf{T}_w {}^w\mathbf{X}_i)\right)^2$$

This is related to the structure from motion problem

Use to be done using Kalman filter [Monoslam Davison 03] but current state of the art approaches rely on bundle adjustment methods using fature or photometric data [PTAM ISMAR 07][DTAM ICCV2011][LSD-SLAM ECCV 2014]…

Large vSLAM

Issue with scale and drift solved thanks to loop closure detection

[Lim ICRA 2014]

An example: LSD-Slam [Engel ECCV 2014]

LSD Slam minimize photometric error

Trends: RGB-D tracking

Use RGD-D camera provides point cloud

Clearly related to SLAM

Registration and localization is the done in the 3D space (ICP)

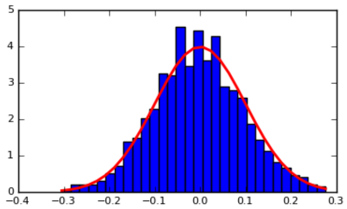
$$\widehat{\mathbf{q}} = \operatorname*{argmin}_{\mathbf{q}} \sum_{i=1}^{N} ({}^{c}\mathbf{X}_i - {}^{c}\mathbf{T}_w {}^{w}\mathbf{X}_i)^2$$

Featured, eg, in Kinect Fusion (with a signed distance function error)

[Newcombe, ISMAR 11]

Tracking with dynamics

$$\widehat{\mathbf{x}}_t = \operatorname{argmax} \; p(\mathbf{x}_t \mid \mathbf{z}_{1:t})$$



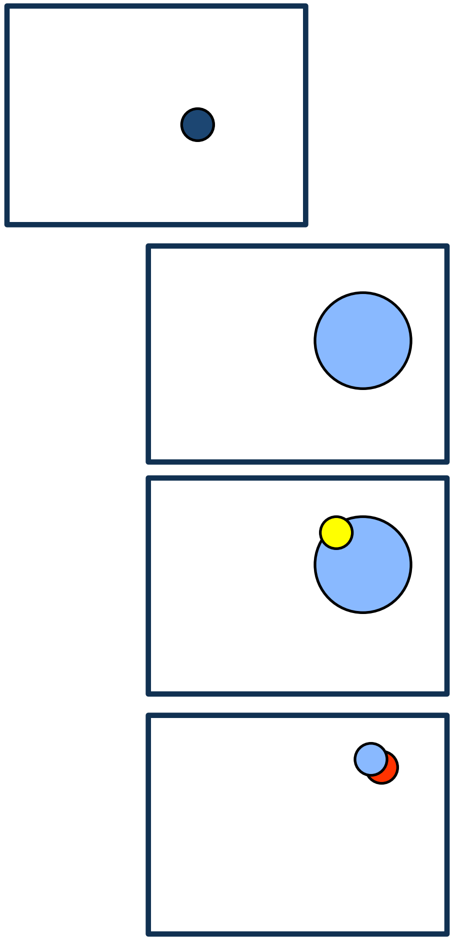
Tracking with dynamics

image measurements is used to estimate position of object, but also incorporate position predicted by dynamics, i.e., the expectation of object's motion pattern

Use a dynamic motion model

- Constant velocity, constant acceleration….

Kalman filter VS particule filter

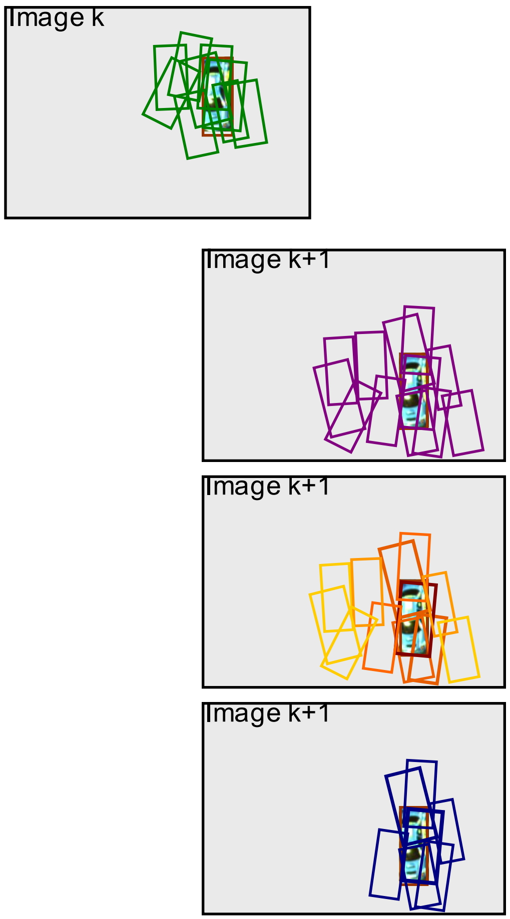


initial position

prediction

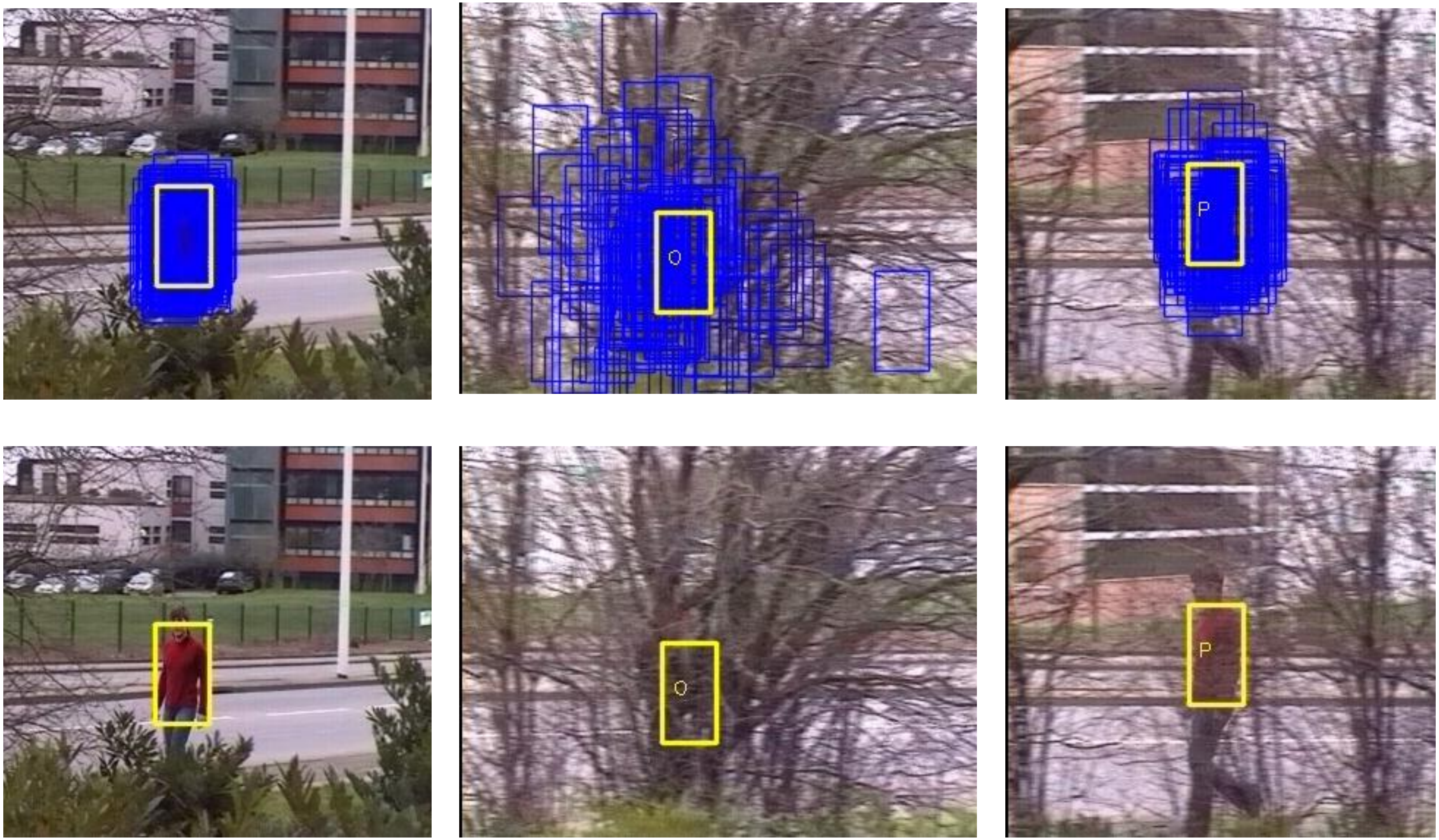
measurement

update



Particle filtering

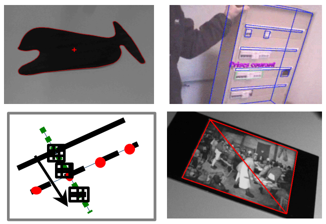
[Isard and Blake, ECCV 1996] [Pérez et al. ECCV'02]



ViSP

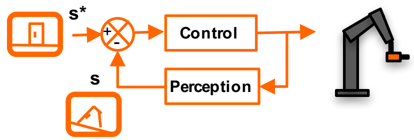
http://visp.inria.fr

Visual tracking core



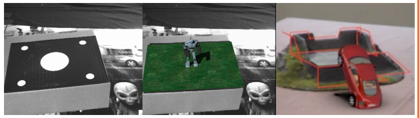
Blob trackers, moving edges, model-based trackers, keypoint trackers, template tracker.

Visual servoing core



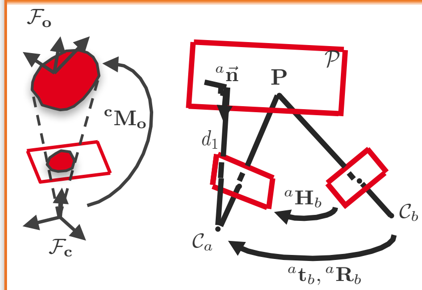
IBVS, PBVS, 2D ½ and many other control laws for eye-in-hand and eye-to-hand systems.

AR core



Provides a wrapper over Ogre 3D engine for augmented reality applications.

Computer vision

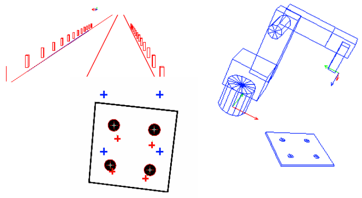


Pose and homography estimation.

Hardware abstraction

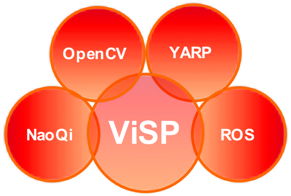
Provides generic interfaces over robot drivers, framegrabbers and display devices.

Simulation



Includes wireframe and robot viewers, planar textures generator.

Bridges



Bridges with Naoqi, OpenCV and YARP, ROS nodes for camera calibration and tracking.

Cross platform

MinGW
Microsoft Visual C++
GCC
eclipse

Support multi OS (Fedora, Ubuntu, Debian, Linux Mint, OSX, Windows), but also compilers (g++, MinGW, msvc…) and IDE.

Inria

lagadic

Image manipulation

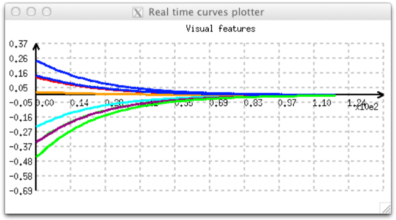
Read-write pgm, png, jpeg, tiff images, filtering, mathematical morphology.

Mathematics core

$$\mathbf{v} = -\lambda \mathbf{L}^{+}(\mathbf{s} - \mathbf{s}^{*})$$

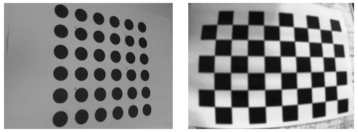
Operations on vectors, matrices, homogeneous transformations, pseudo-inverse or SVD computation.

Real-time data plotter



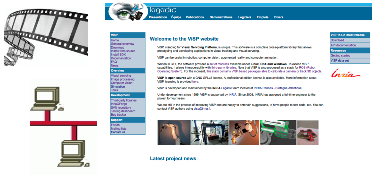
Display in real-time and record time-graphs, x/y-graphs or 3D curves.

End user tools



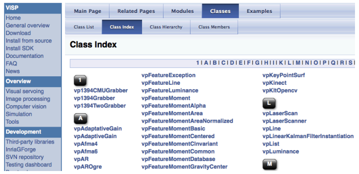
Camera calibration, hand-eye calibration.

Many more features



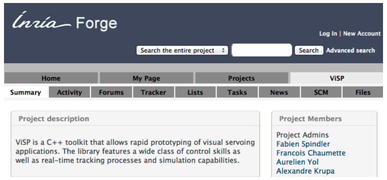
Movie reader and recorder, XML I/O, data transmission over the network, Kalman filter...

Powerful API



C/C++, 310 000 lines, more than 250 classes fully documented, 200 examples 200 sample codes, and 27 tutorials.

Forge



Hosted on GForge, under Subversion control. Mailing lists, forum, bug tackers, continuous integration.

Open source license

open source

Released under the terms of the open source GPLv2 license. Also available as a professional edition.

Inria

lagadic

Conclusions

Old problem but still open

Nevertheless, efficient solutions now exists !

Open (difficult) problems

- Initialization is still an issue (especially when 6+ DoF are concerned)
- Robustness vs Efficiency
- High number of DoF

Foreseen solutions

- Learning (aspects, shape, …)
- On-line modification learning/estimation