PERCEPTION FOR ROBOTICS: PART II

Localization & 3D reconstruction

Cédric Demonceaux – ImViA VIBOT ERL CNRS 6000
Université Bourgogne Franche-Comté

Robotics Winter School

January 2019

cedric.demonceaux@u-bourgogne.fr

- «What is around me?»
- In this part, we will explain some basic tools of computer vision for helping the robot to perceive its world.
- We will try to understand how the robot can localize itself with a vision system (mono-camera, multi-camera, LiDAR, RGB-D cameras…)

Introduction

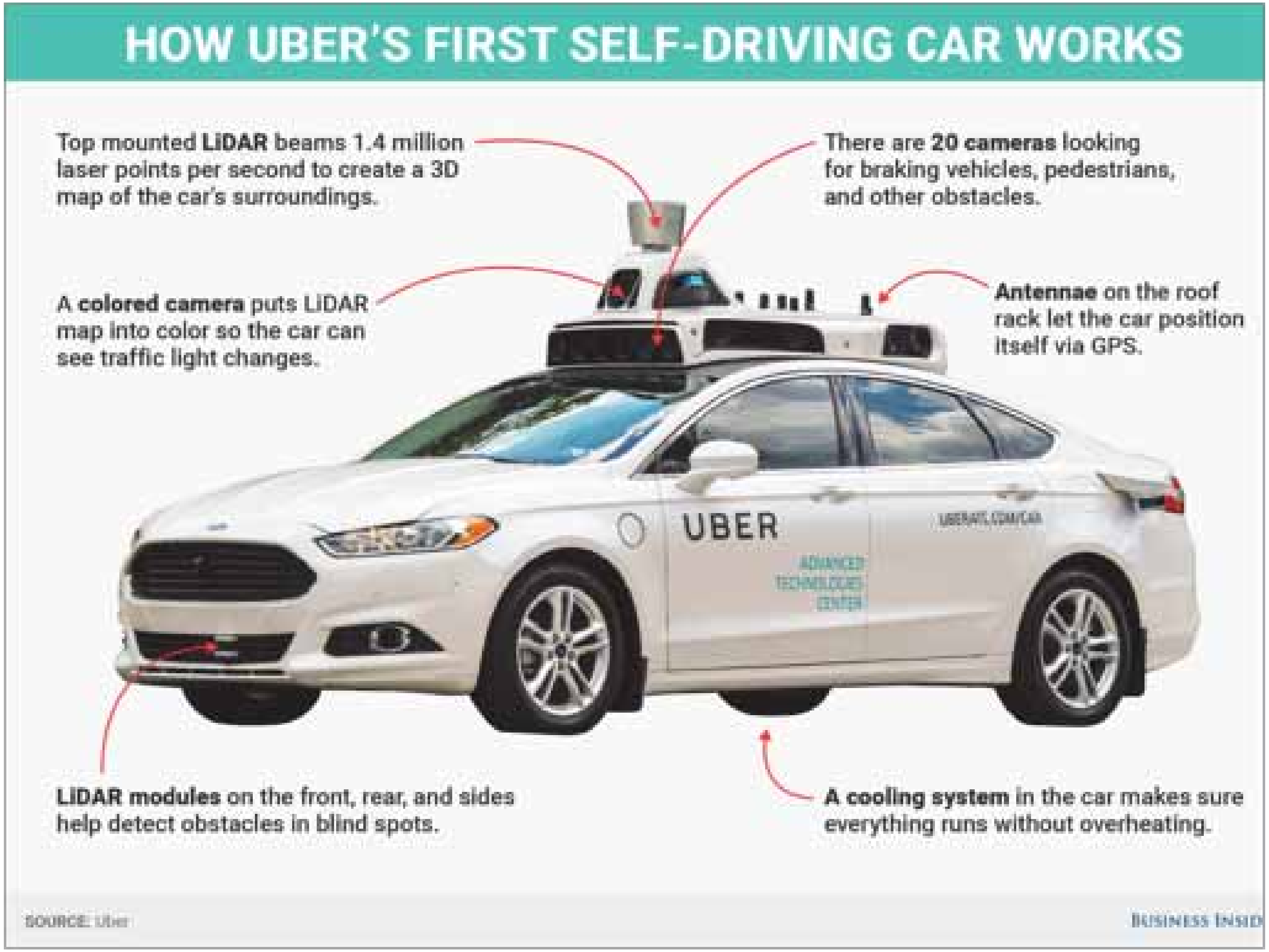


Table of contents

ACTIVE SENSORS

- Use external devices that emit light wavelength, signal or patterns to interact with the scene.
- The data generated by this external source are gathered by the sensor to deduce information on the environment around the robot.
- This conversion can be carried out in many ways depending upon the type of sensors.
- 4 main technologies: RADAR, LiDAR, Structured-Light and Time-of-Flight.

RADAR

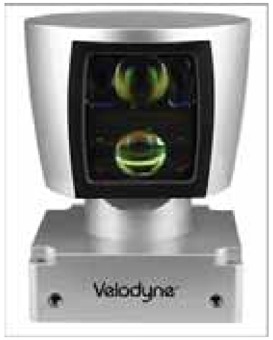
- Radio Detection and Ranging uses radio wave to compute velocity and/or range to an object.
- Large wavelenght
  - ✓works with large distance
  - ✗low resolution

Principle

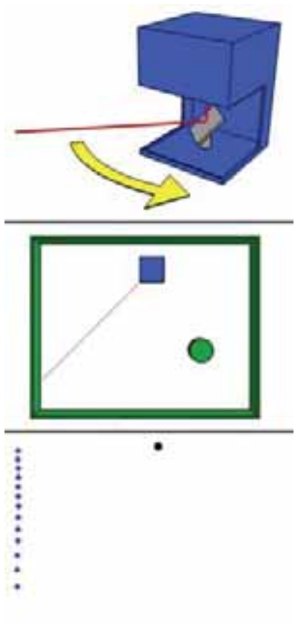
Radar sensor

LiDAR

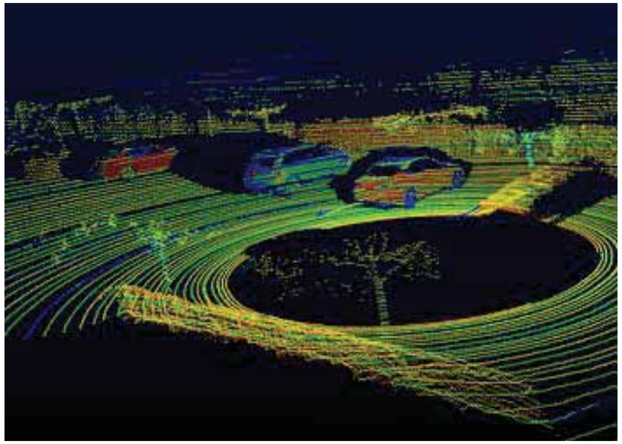
- Light Detection and Ranging uses a laser that is emitted and received back
- Small wavelength
  - ✗ works with small distance
  - ✓ high resolution



LiDAR



Principle



LiDAR Data

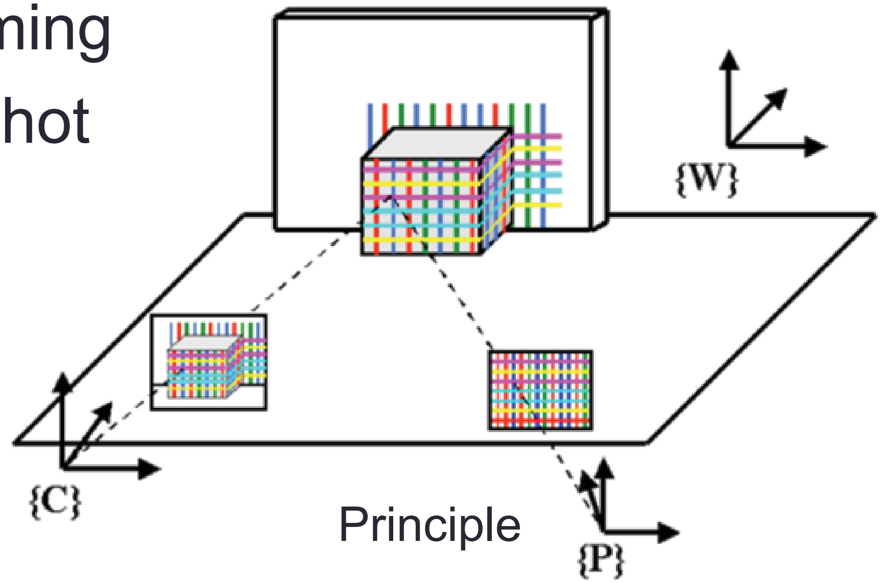
Structured light

- Project bi-dimensional patterns to estimate the dense depth information of the object surface points.
- The main role of the projected patterns is to establish correspondences between the known pattern

✓Light, small, low energy consuming

✓Color + 3D information in one shot

✗Sensitive to the light condition.



Principle

Structured light

Kinect 1
RGB : 640*480
Depth : 320*240

Time-of-Flight Cameras

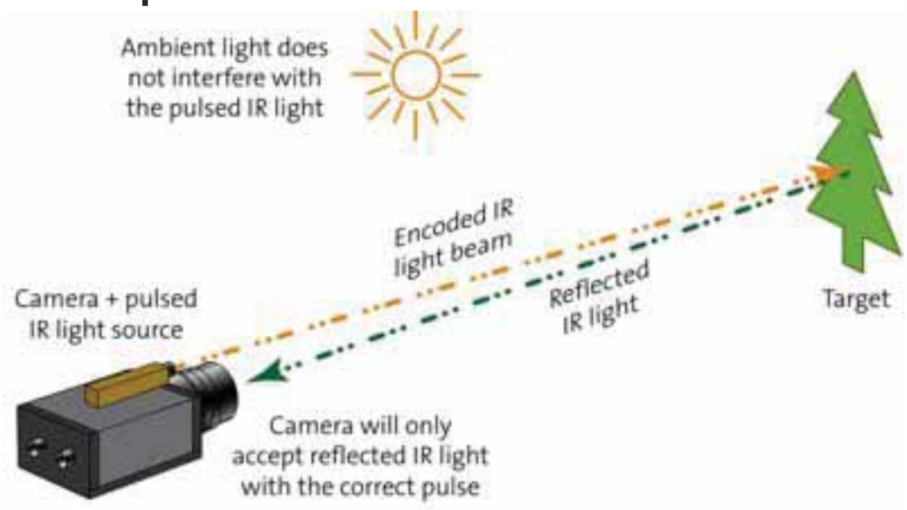
- Range imaging that measure the time of flight of a signal between camera and the object
- The artificial illumination may be provided by laser or LED

✓Can provide Color + 3D data in one shot

✓No sensitive to the light condition

✗Low resolution compared to 2D cameras

✗Expensive



Principle

SwissRanger 4000
176*144

Time-of-Flight Cameras

Kinect 2
RGB : 1920*1080
Depth : 512*424

PASSIVE SENSORS

- Gather data through the detection of vibrations, light, radiation, heat or other phenomena occurring in the environment without external devices.
- Commonly, the passive sensors used in robotics are cameras.

CAMERA

1826 : « Point de vue du gras » (Nicéphore Niepce)

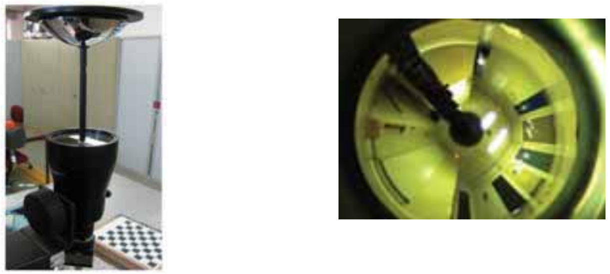
CAMERA

2019 : Photo of Shanghai, 195 billion pixels

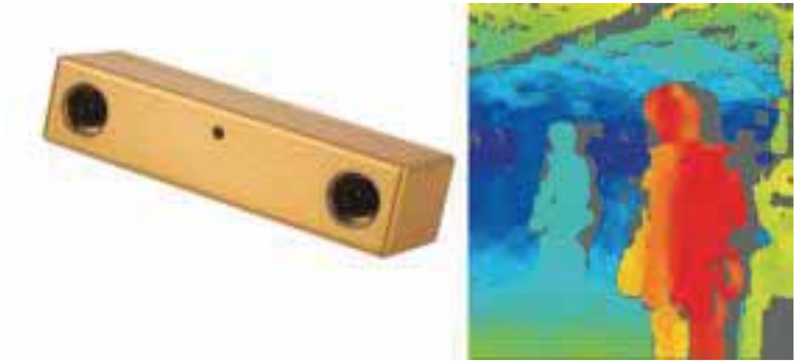
CAMERAS : multifocal

Perspective camera

Fisheye camera



Catadioptric camera

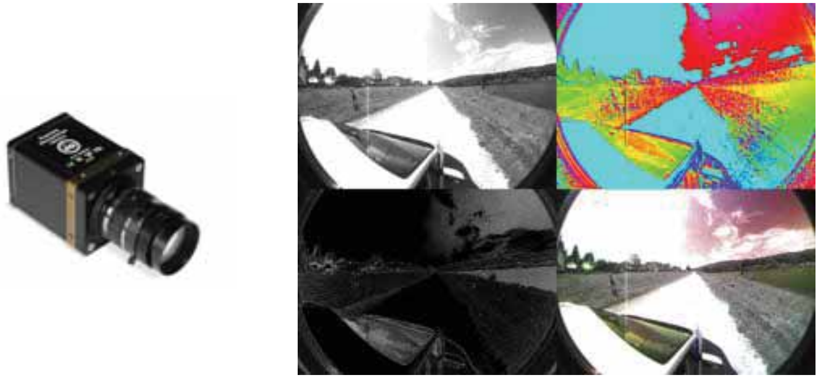


Stereoscopic cameras

Spherical camera

CAMERAS : multimodal

Thermal camera



Polarimetric camera

CAMERAS

- ✓This sensor gives a rich information about the scene
- ✓High Resolution
- ✗Some computer vision tools are required to obtain 3D data.
- Main steps :
  - Feature detection and matching
  - Calibration
  - Pose estimation / Visual Odometry / Bundle adjustment

SOME BASIC IMAGE PROCESSING TOOLS

Features detection

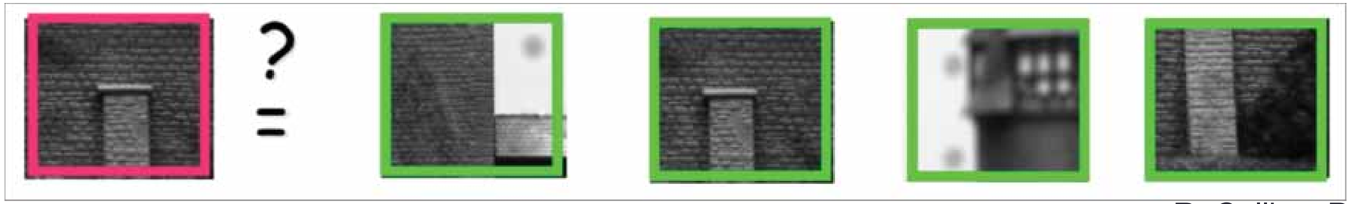
Optical Flow

Feature detection

HARRIS detector

- Principle : detect points based on intensity variation in a local neighborhood

A patch is a good candidate for matching if it is very distinctive



R. Collins, Penne State

Feature detection

HARRIS detector

pixel $p = (u, v)$

$$E(p) = \sum_{(x,y)\in W} w(x,y)[I(x+u, y+v) - I(x,y)]^2$$

neighborhood

Weighted function

By a Taylor Expansion:

$$E(p) \simeq [u,v]M[u,v]^T$$

Image derivatives matrix:

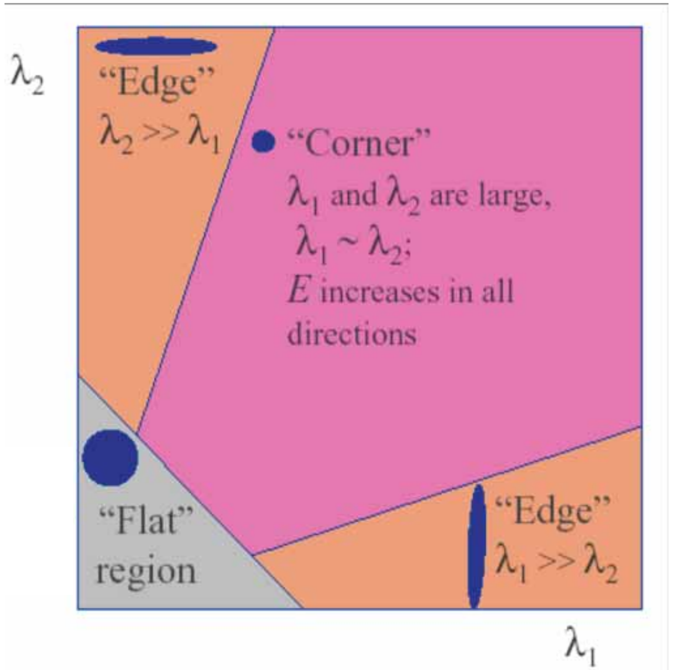
$$M = \sum_{(x,y)\in W} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Feature detection

HARRIS detector

This matrix plays an important role in image processing because it characterizes the homogeneity of a patch.

This disparity can be estimated by its eigenvalues :



Corner response descriptor

$$R = \det(M) - k(trace M)^2$$

$$k \in [0.04, 0.06]$$

R is "large" for a corner

- ✓ Fast
- ✓ Rotation invariant
- ✗ Not scale invariant

R. Collins, Penne State

Feature detection

SIFT

Scale-invariant Feature Transform is inspired by Harris detector by making a detector/descriptor scale invariant.

Difference of Gaussian (DoG) :

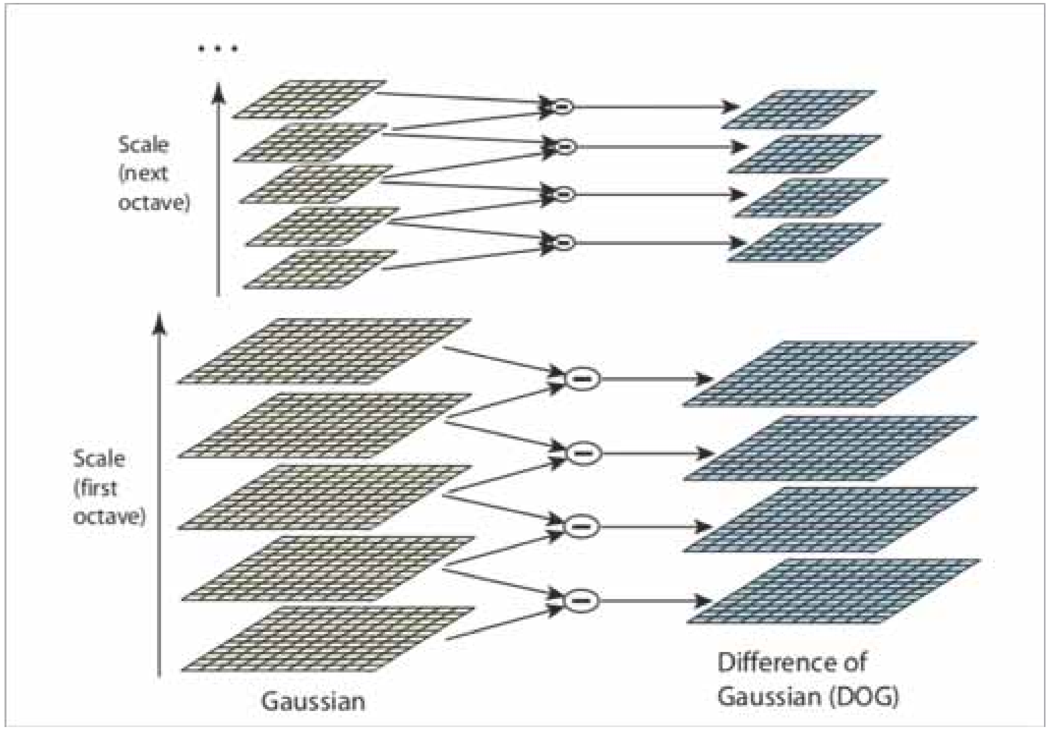
$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$

$G(x, y, \sigma)$ Gaussian at scale σ

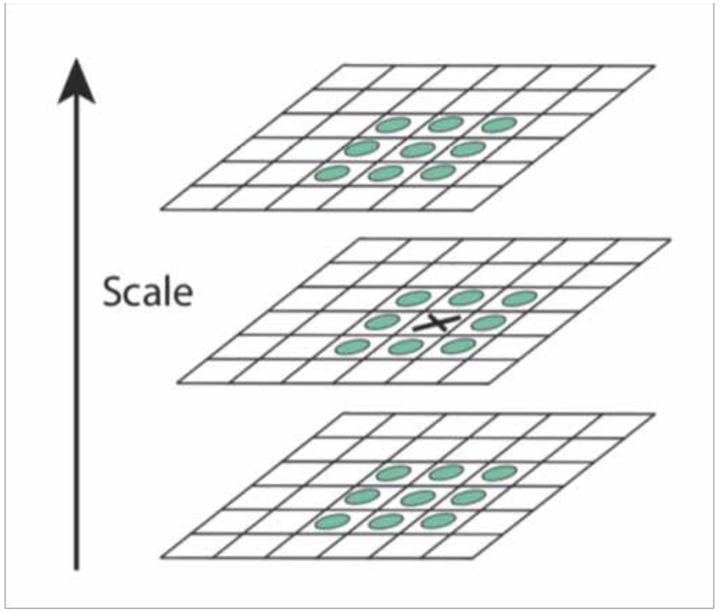
R. Collins, Penne State

Feature detection

SIFT - detector



DoG computation

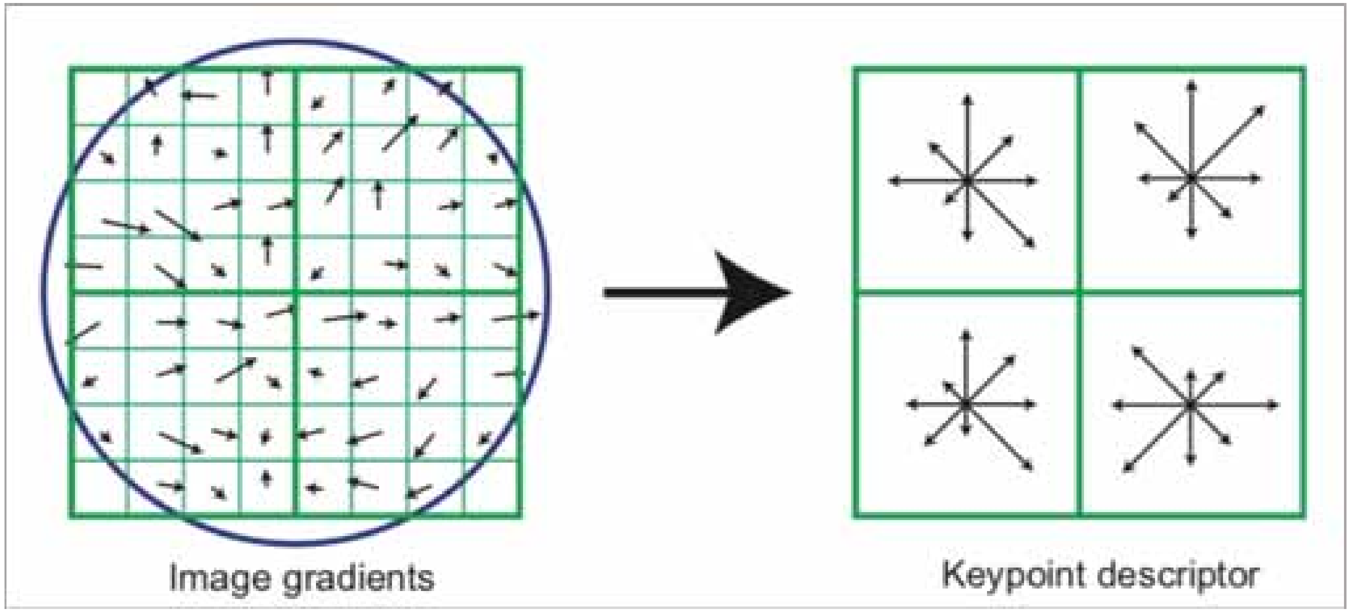


Feature extraction based on M matrix

D. Lowe – IJCV 2004

Feature detection

SIFT - descriptor



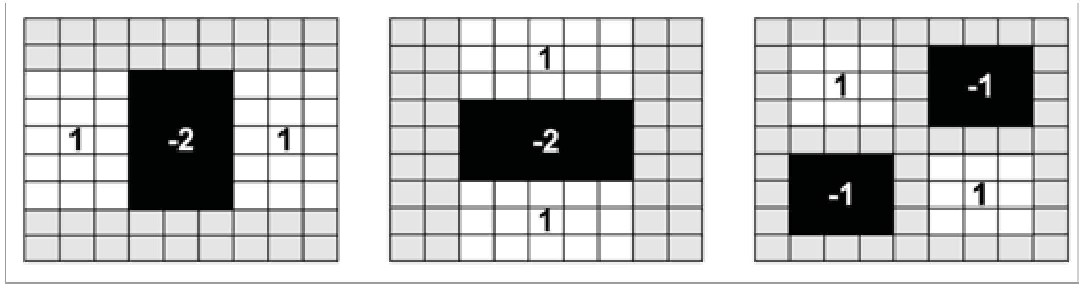
✓ Robust to rotation and scale, to change in illumination, camera view point
✗ Time consuming

D. Lowe – IJCV 2004

Feature detection

SURF

Speeded Up Robust Features approximate DoG by 2D-Haar wavelets and use the integral images to speed up the computation.



Haar kernels to approximate DoG

✓ Robust to rotation and scale, to change in illumination
✓ 3-7 times faster than SIFT
✗ Less efficient than SIFT

Feature detection

FAST

Features from Accelerated Segment Test uses a circle of 16 pixels stored and analyzed as a vector.

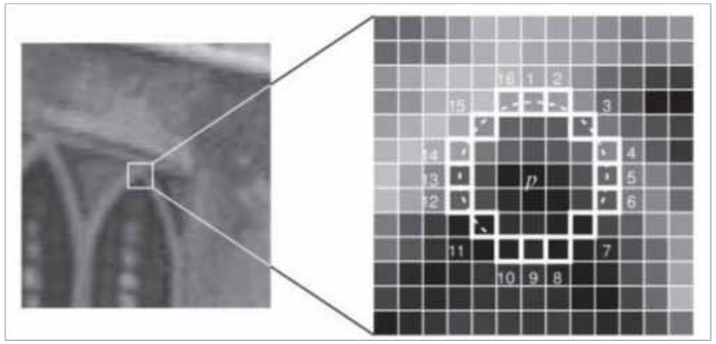
p is a corner if :

For a set S of N contiguous pixel :

$$I(x) > I(p) + t$$

or

$$I(x) < I(p) - t$$

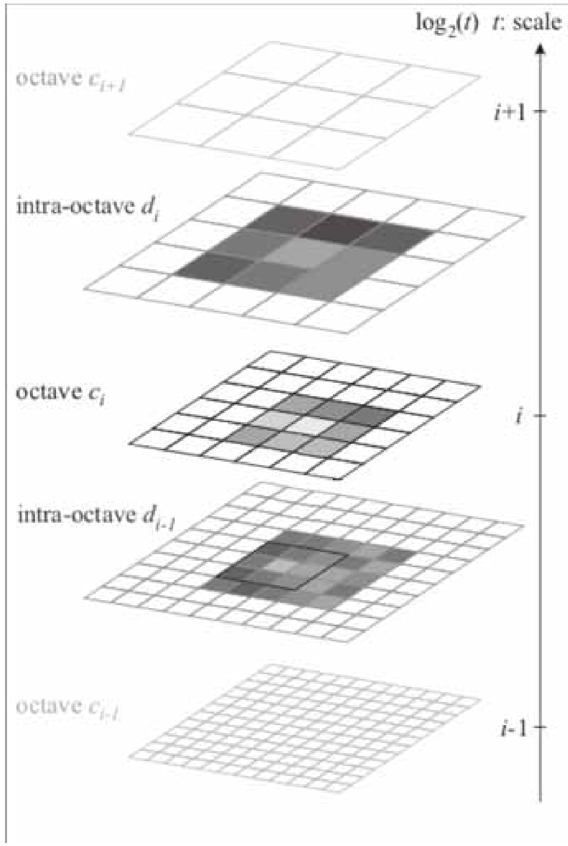


- ✓ Robust to rotation
- ✓ Fast
- ✗ Less robust to change illumination

Feature detection

BRISK

Binary Robust Invariant Scalable Keypoints use FAST detector in a multiresolution scheme for scaling invariance.



- ✓ Robust to rotation and scale
- ✓ Fast
- ✗ Less robust to change illumination

Feature detection

ORB

Oriented Fast and Rotated BRIEF is a combination of FAST detector and BRIEF descriptor (Binary Robust Independent Elementary Features)

- ✓ Robust to rotation and scale
- ✓ Fast
- ✓ Good alternative to SIFT and SURF
- ✓ Use in many SLAM methods

Feature detection

	Rotation Invariant	Scale Invariant	Repeatability	Localization accuracy	Robustness	Efficiency
HARRIS	X		+++	+++	++	++
SIFT	X	X	+++	+++	+++	+
SURF	X	X	++	++	++	++
FAST	X		++	++	++	+++
BRISK	X	X	+++	++	++	+++
ORB	X	X	+++	++	++	+++

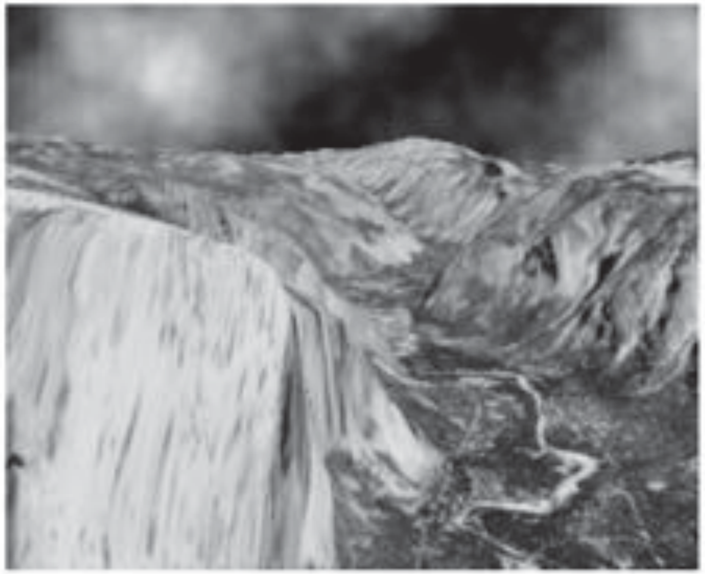
Performance comparison

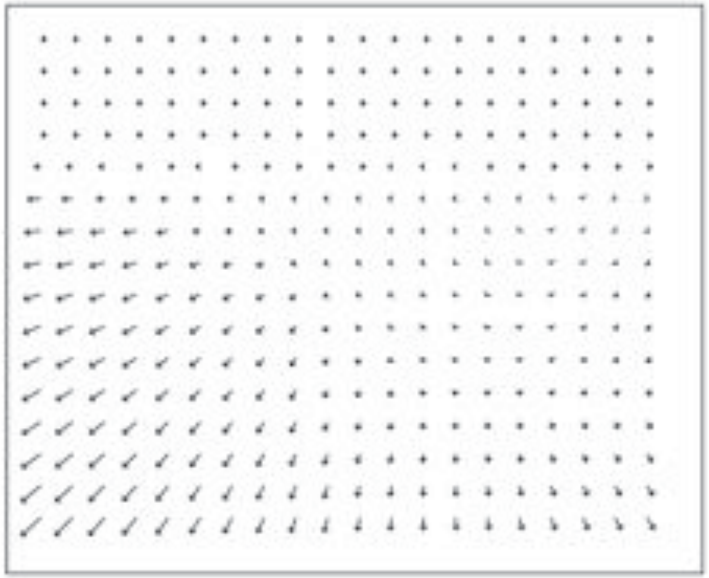
Optical flow

$I((x(t), y(t)), t)$ an image sequence

Determine the Optical Flow means to compute the 2D motion field:

$$\vec{v}((x(t), y(t), t) = (\frac{dx}{dt}(t), \frac{dy}{dt}(t))$$





Yosemite sequence

Optical flow

Main hypothesis : **The brightness constancy**.

The brightness of a physical point in the image does not change over the time.

$$I((x(t), y(t)), t) = I((x(t_0), y(t_0)), t_0)$$

By derivation, we obtain the Optical Flow Constraint Equation :

$$\vec{\nabla} I . \vec{v} + \frac{\partial I}{\partial t} = 0,$$

2 unknowns, 1 equation => Aperture problem

2 solutions to overcome this problem :

Dense & Sparse approaches

Optical flow

Horn-Schunk methods (1981):

Hyp: the optical flow is smooth

Optical flow constraint :

$$H_1(\overrightarrow{v}) = \int\int \left(\overrightarrow{\nabla} I.\overrightarrow{v} + \frac{\partial I}{\partial t}\right)^2 dxdy.$$

Regularization term

$$H_2(\overrightarrow{v}) = \int\int \|\overrightarrow{\nabla} v\|^2 dxdy.$$

Horn and Schunck estimate $\overrightarrow{v}$ which minimizes :

$$E(\overrightarrow{v}) = H_1(\overrightarrow{v}) + \alpha^2 H_2(\overrightarrow{v})$$

Optical flow

Lucas-Kanade methods (1981):

Hyp: the optical flow is constant on the neighborhood W

$$E(\vec{v}(p,q)) = min_{\vec{v}} \sum_{(p,q)\in W_{(p,q)}} w^2(p,q) \left[\vec{\nabla} I(p,q).\vec{v}(p,q) + \frac{\partial I}{\partial t}(p,q) \right]^2$$

$\vec{v}$ is computed by:

$$\vec{v}(p,q) = -M^{-1} \begin{bmatrix} \sum_{(p,q)\in W_{(p,q)}} w(x,y) I_x I_t \\ \sum_{(p,q)\in W_{(p,q)}} w(x,y) I_x I_t \end{bmatrix}$$

Where M is the image derivatives matrix

ROBUST TECHNIQUES

RANSAC

M-Estimator

Robust estimation

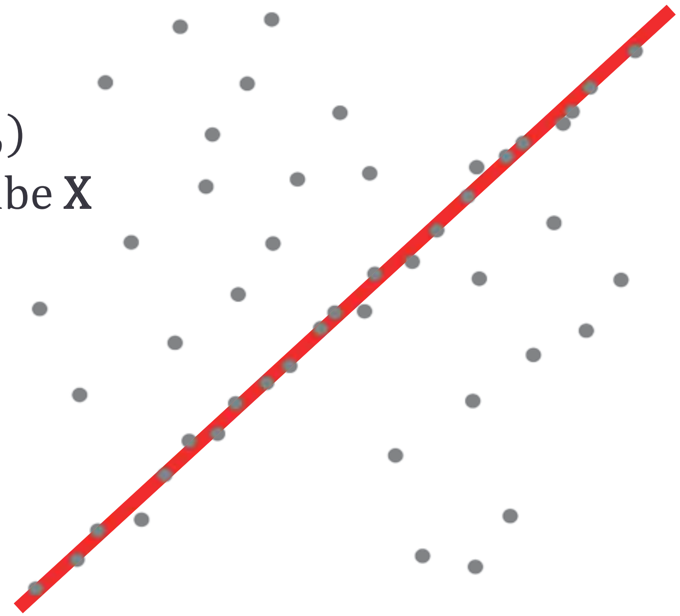
- In practice, a lot of data are noisy
  - Noise in the image
  - Bad matching
  - Bad motion estimation
  - Occultation
  - Dynamic objects in the 3D scene
  - ….
- Thus, computer vision tools require robust estimation

Toy example

$\boldsymbol{X} = (X_1, X_2 \ldots X_M)$
M data

$\boldsymbol{\Theta} = (\Theta_1, \Theta_2 \ldots \Theta_p)$
p variables to describe **X**

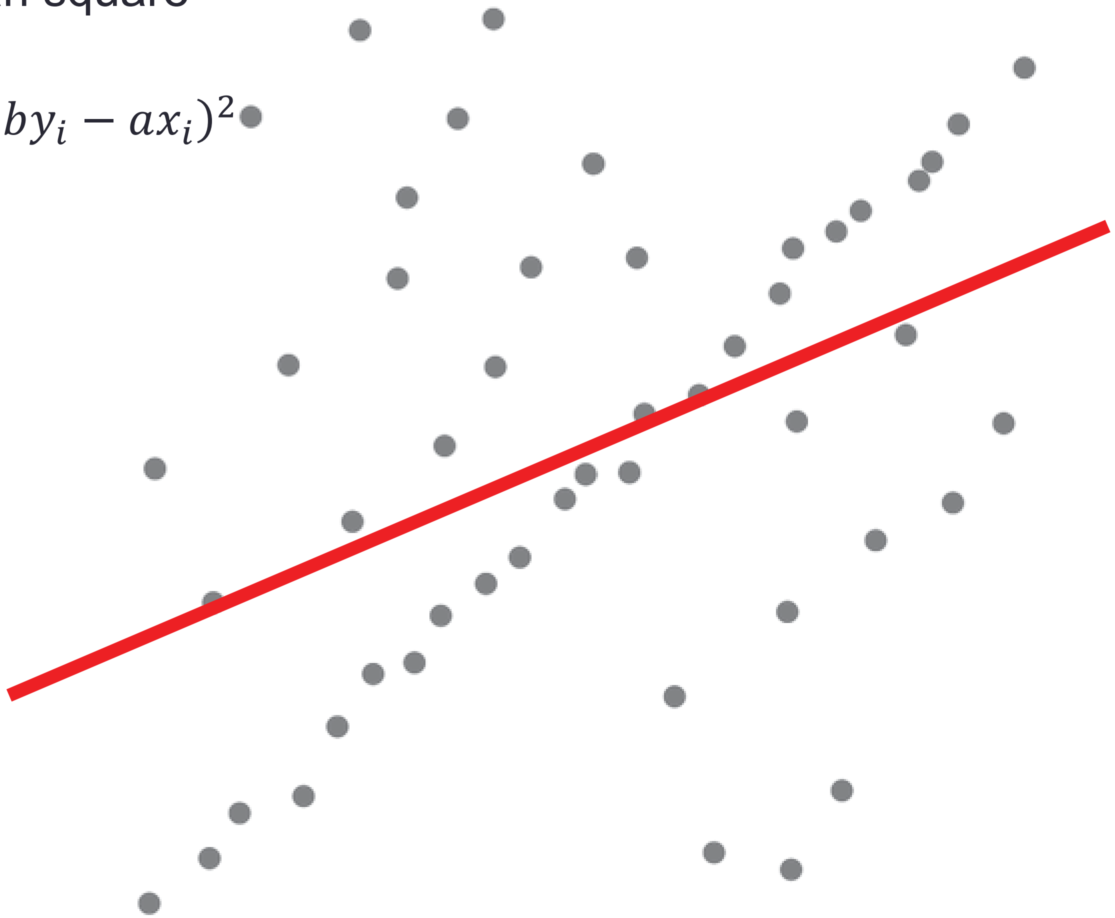
$\boldsymbol{\Theta} \leftarrow f\left(X_{i_1}, X_{i_2} \ldots X_{i_n}\right)$
n points are needed to compute $\boldsymbol{\Theta}$



Toy example

Least mean square

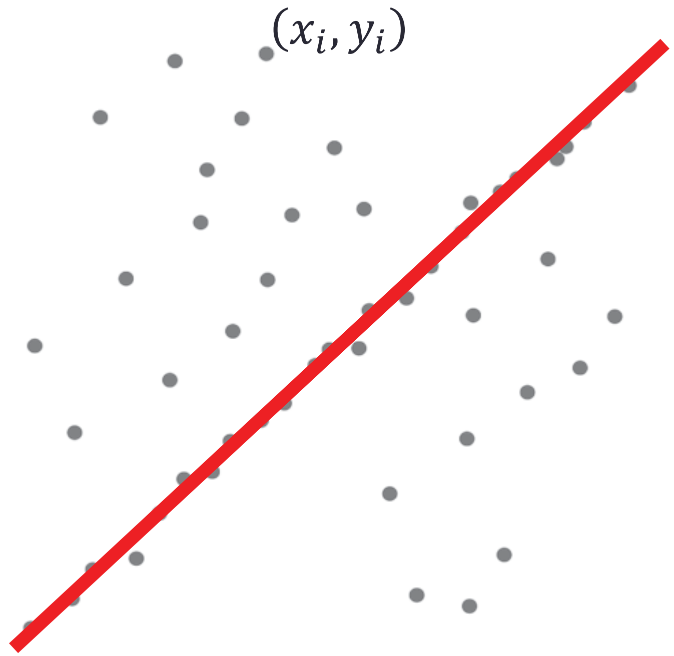
$$\Theta = \min_{(a,b)} \sum_{i=1}^{M} (1 - by_i - ax_i)^2$$



RANSAC (RANdom Sample Consensus)

<u>Example :</u> line fitting

$\boldsymbol{X} = (X_i = (x_i, y_i))_{i=1\ldots M}$



$$a x_i + b y_i = c \quad \forall\, (x_i, y_i)$$

$$\boldsymbol{\Theta} = (a, b, c)$$

How many points of $\boldsymbol{X}$ do we need in order to estimate $\boldsymbol{\Theta}$?

RANSAC (RANdom Sample Consensus)

A lot of problems in CV can be modelized by :

$$\Theta = f(x_{i1}, x_{i2}, \cdots x_{in})$$

with $X = (x_1 \cdots x_M)$, M observations

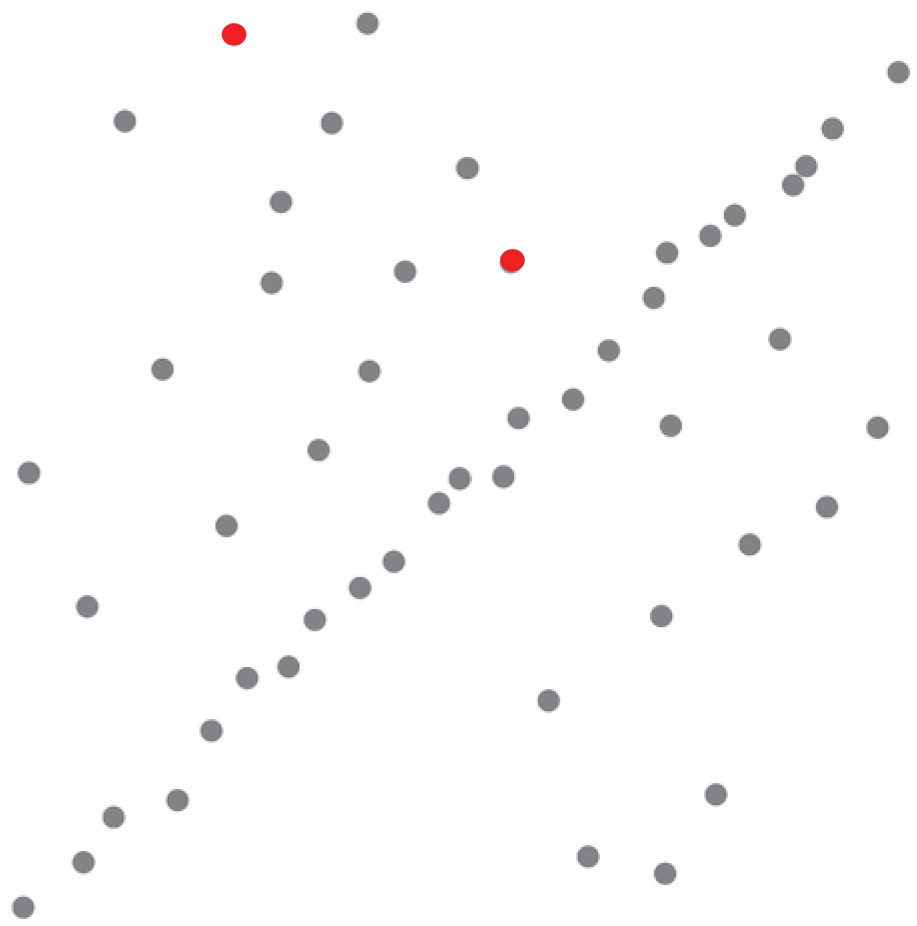
n : number of data needed for estimating Θ

If all the data are inliers, Θ can be evaluated whatever the n observations we choose.

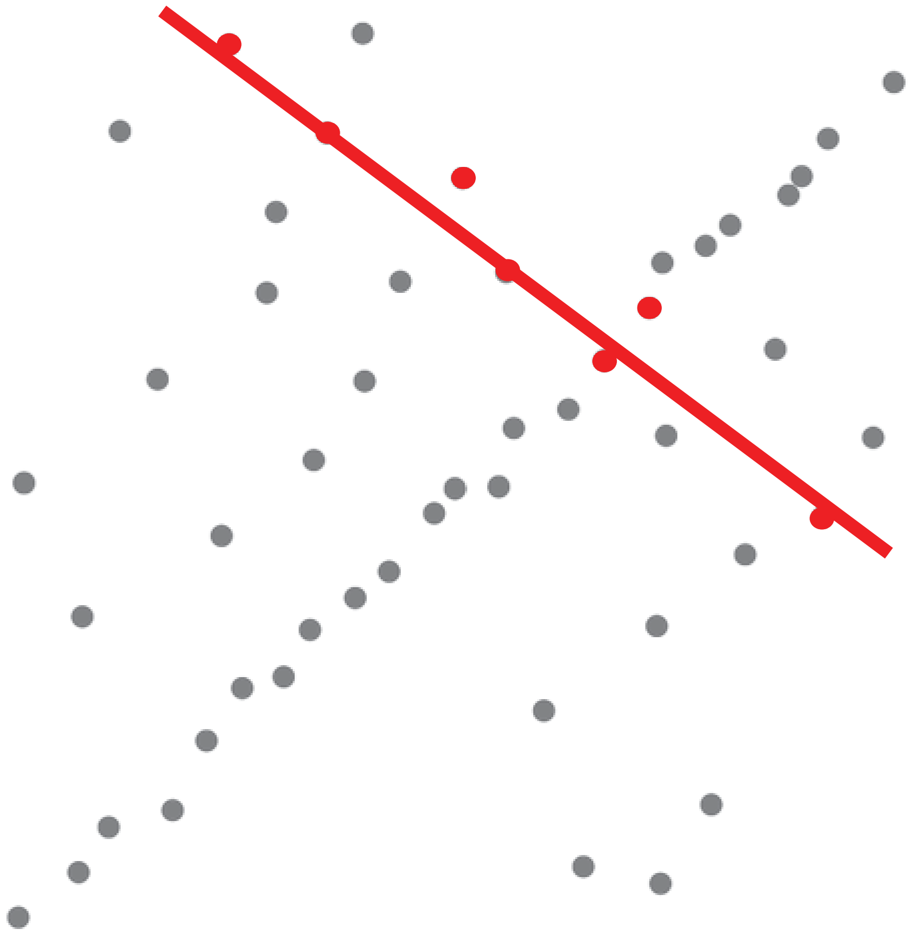
In practice, some data are often corrupted !

How to ensure the robustness of the approach?

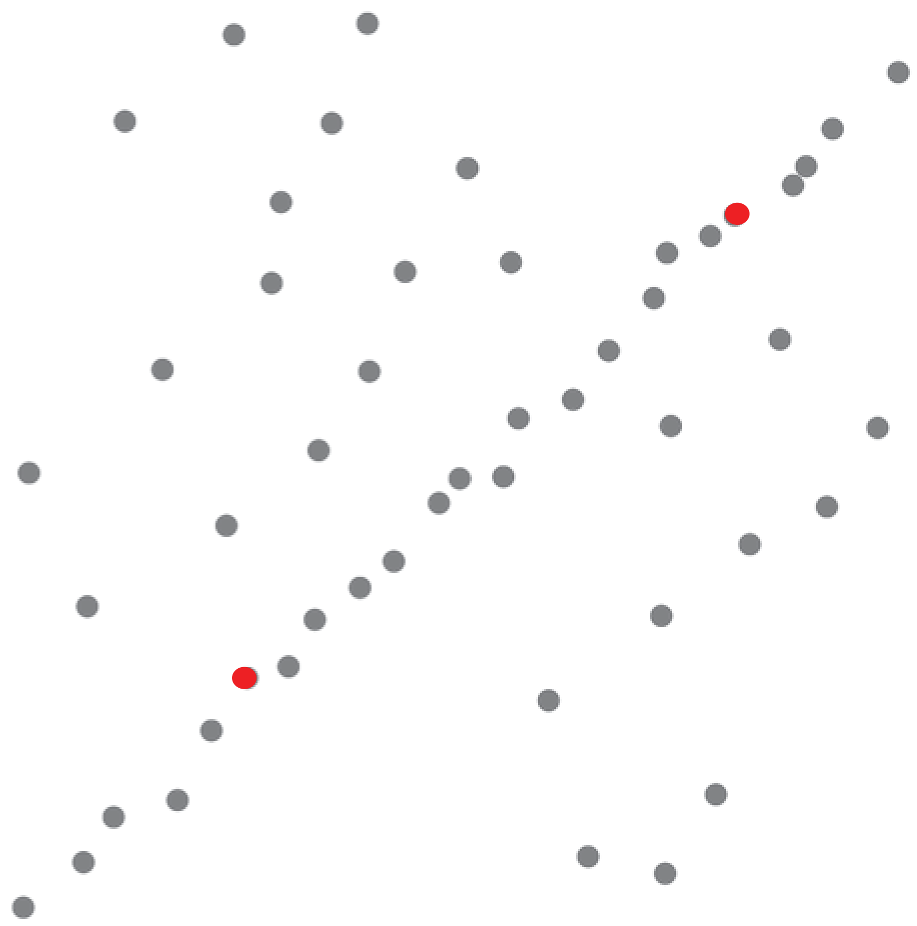
RANSAC (RANdom Sample Consensus)



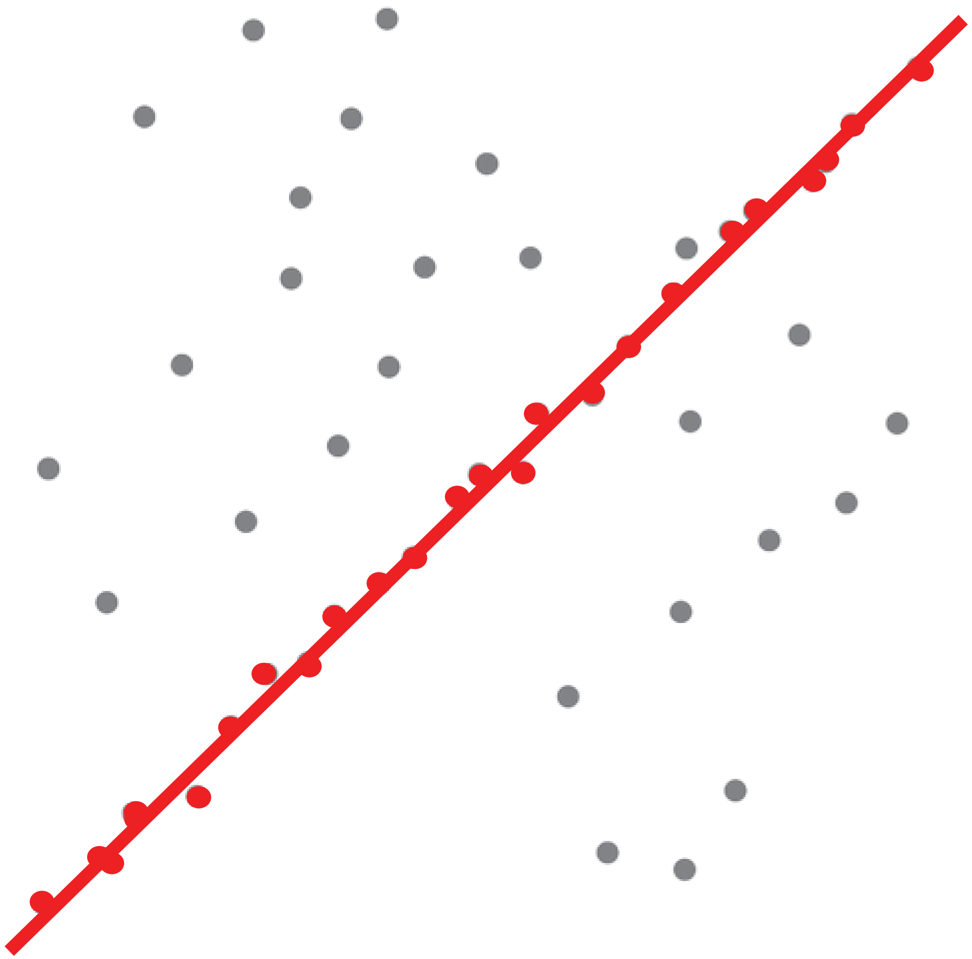
RANSAC (RANdom Sample Consensus)



RANSAC (RANdom Sample Consensus)



RANSAC (RANdom Sample Consensus)



RANSAC (RANdom Sample Consensus)

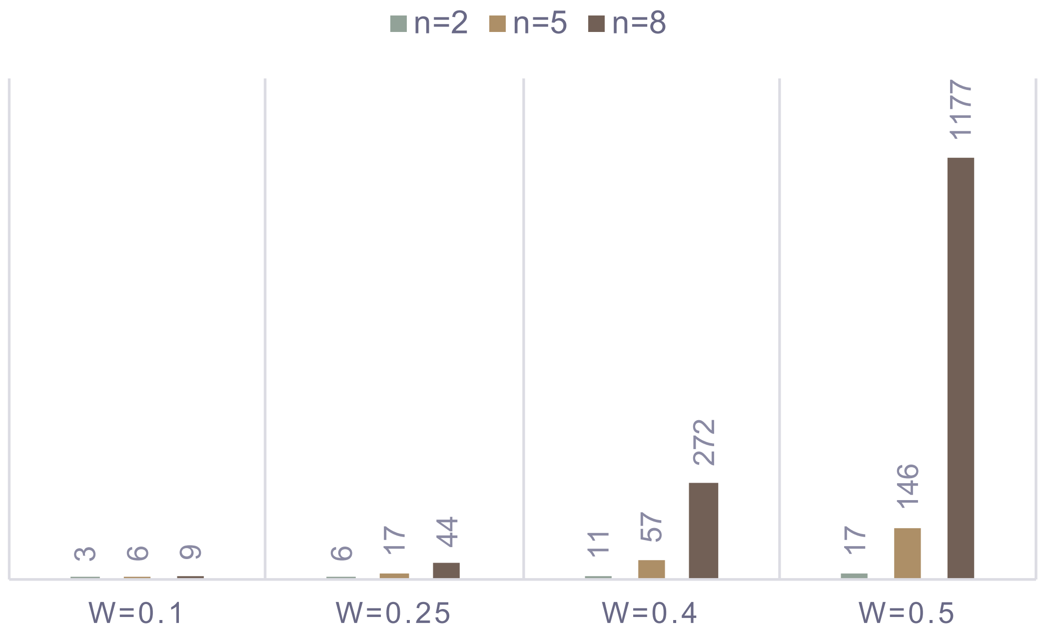
To ensure the convergence of the algorithm, we have to iterate L times where :

$$\mathrm{L} = \frac{\log(1 - p_r)}{\log(1 - w^n)}$$

p_r = success probability
w = ratio of outliers
n = number of observations needed for estimating $\boldsymbol{\Theta}$

RANSAC (RANdom Sample Consensus)

With $p_r = 0.99\%$,



-> it's really important to reduce the point we need to estimate the model in order to reduce the number of iteration

M-estimator

<u>Example :</u> line fitting

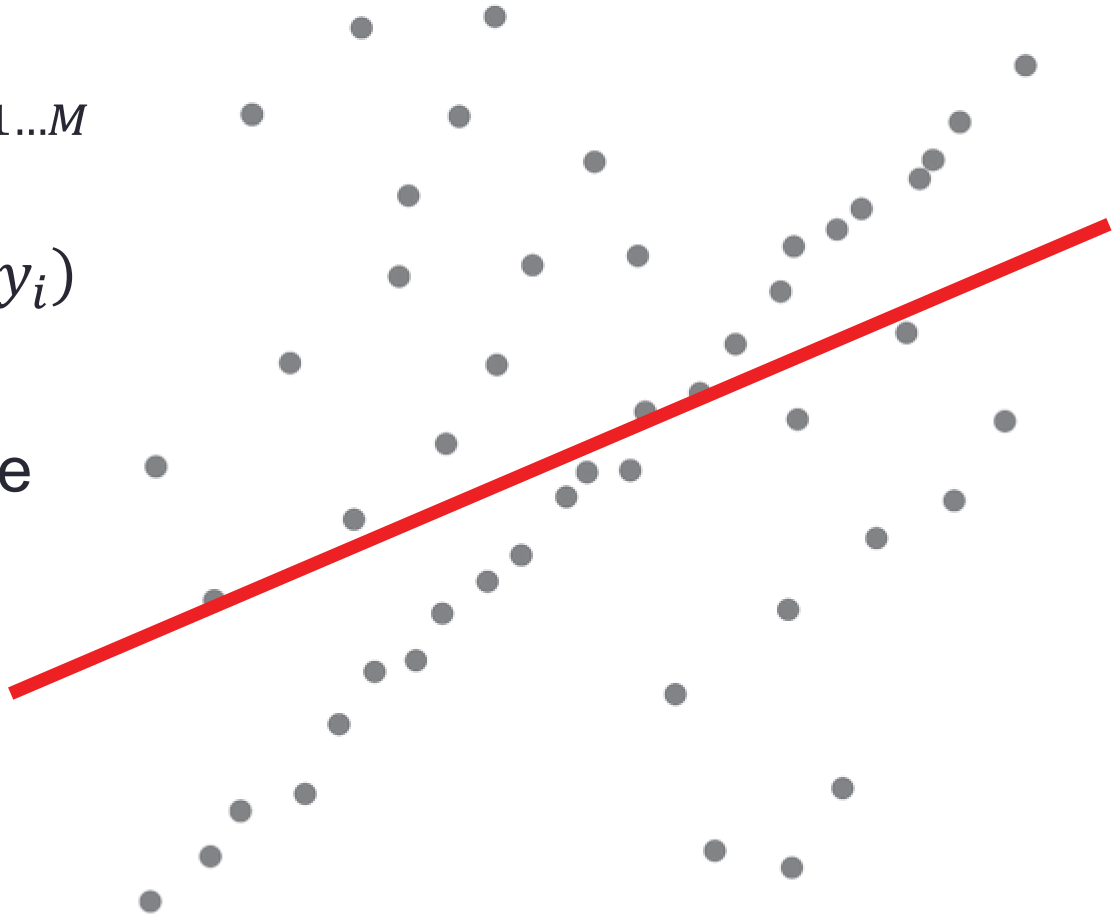
$\boldsymbol{X} = (X_i = (x_i, y_i))_{i=1...M}$

$$ax_i + by_i = c \quad \forall\, (x_i, y_i)$$

$$\boldsymbol{\Theta} = (a, b, c)$$

Least mean square

$$\Theta = \min_{(a,b)} \sum_{i=1}^{M} (1 - by_i - ax_i)^2$$



M-estimator

Idea : Replace the quadratic error

$$\Theta = \min_{(a,b)} \sum_{i=1}^{M} (1 - by_i - ax_i)^2$$

By $$\Theta = \min_{(a,b)} \sum_{i=1}^{M} \rho(1 - by_i - ax_i)$$

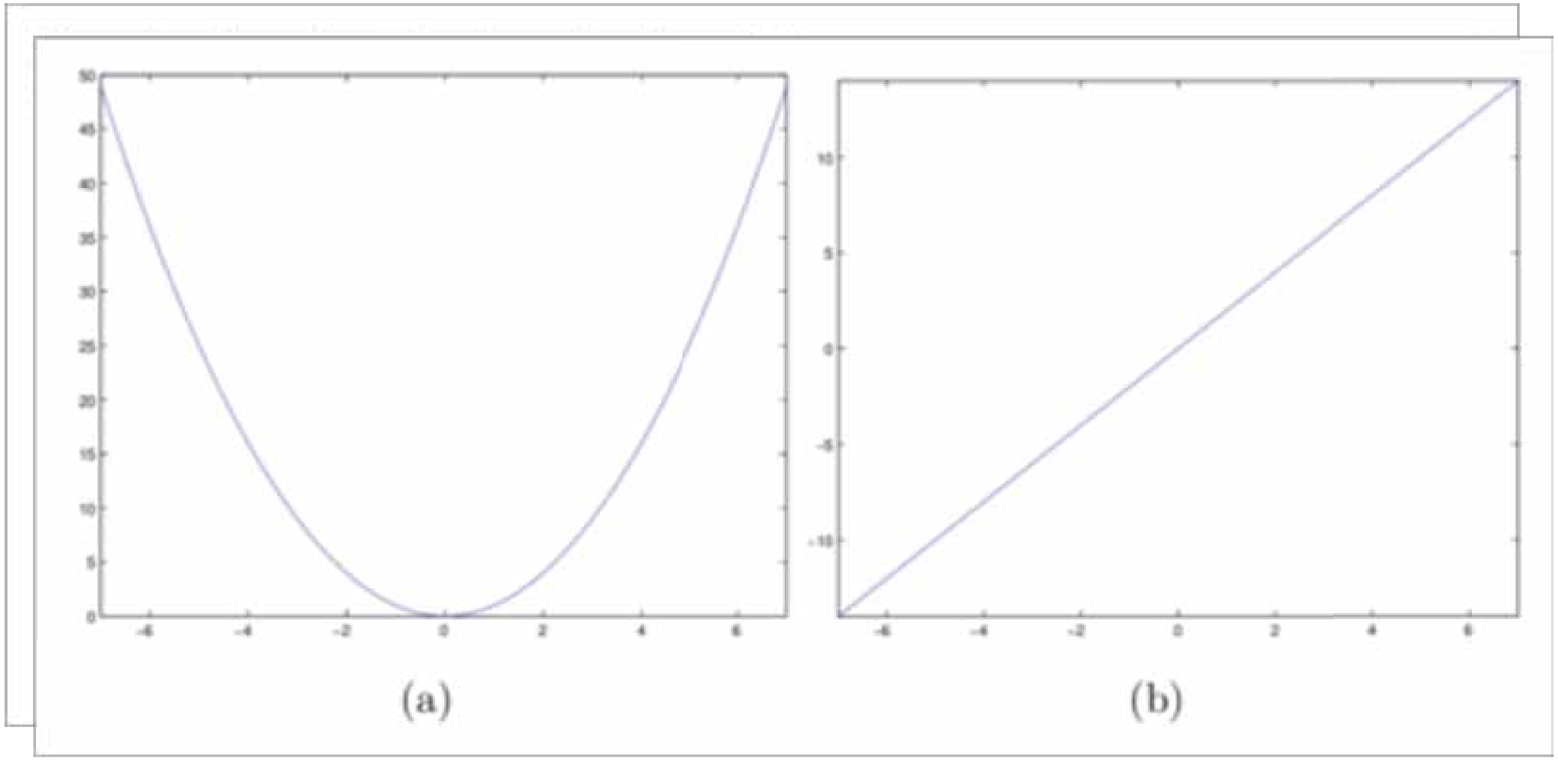
Where ρ is a function called M-estimator.

It minimizes by an iteratively re-weighted least squares

$$\hat{\Theta} = \arg\min_{\Theta} \sum_{i=1}^{n} \frac{1}{2} w_i r_i^2 \quad \text{with} \quad w_i(r,\sigma) = \frac{\Psi(r,\sigma)}{r}$$

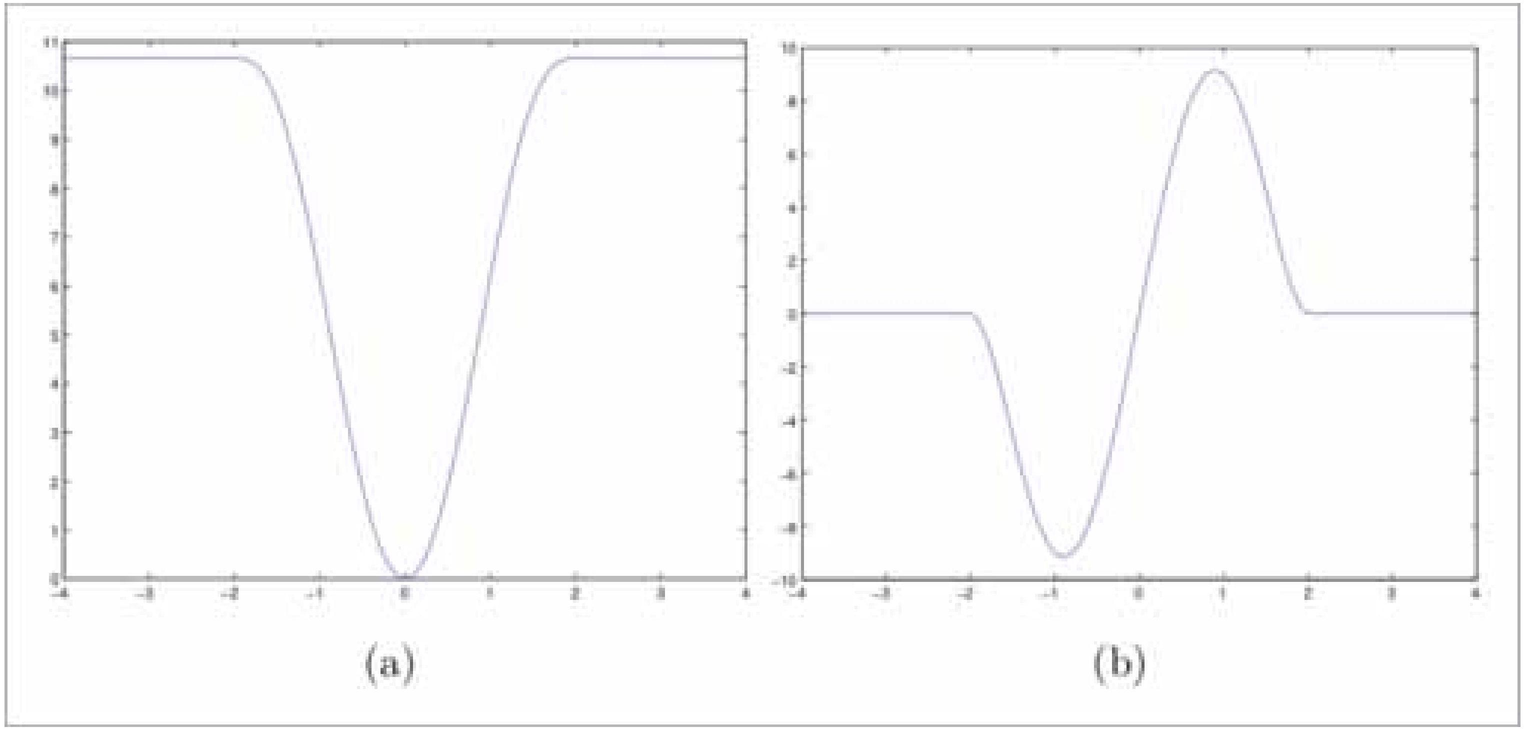
Influence function
Derivative of ρ

M-estimator



(a) $\rho(x) = x^2$ (b) its influence function $\psi = 2x$

M-estimator



(a) Tukey M-estimator (b) its influence function

$$\rho(x,\sigma) = \begin{cases} \frac{x^6}{6} - \frac{\sigma^2 x^4}{2} + \frac{\sigma^4 x^2}{2} & \text{if} \quad |x| < \sigma \\ \frac{\sigma^6}{6} & \text{elseif,} \end{cases}$$

$$\Psi(x,\sigma) = \begin{cases} x(x^2 - \sigma^2)^2 & \text{if} \quad |x| < \sigma \\ 0 & \text{elseif.} \end{cases}$$

M-estimator

Example : line fitting

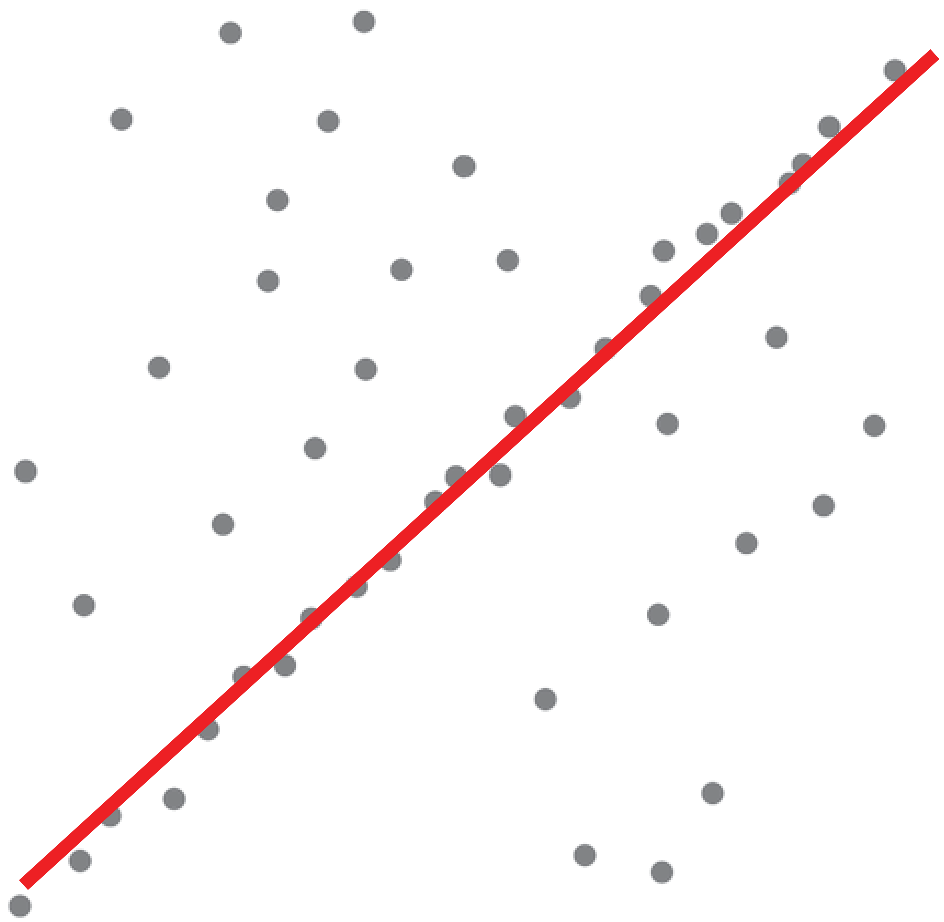
$\boldsymbol{X} = (X_i = (x_i, y_i))_{i=1...M}$

$$ax_i + by_i = c \quad \forall\, (x_i, y_i)$$

$$\boldsymbol{\Theta} = (a, b, c)$$

M-estimator

$$\Theta = \min_{(a,b)} \sum_{i=1}^{M} \rho(1 - by_i - ax_i)$$



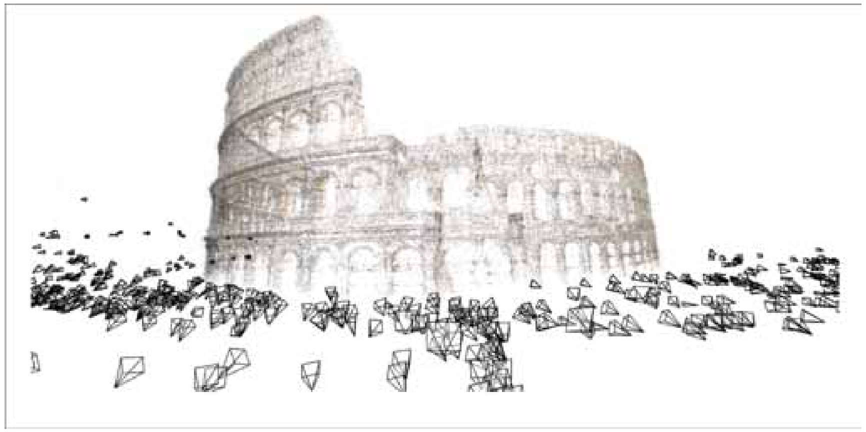
FROM 2D CAMERAS TO 3D RECONSTRUCTION AND MOTION

Camera modeling and calibration

Epipolar geometry

Multiple view geometry

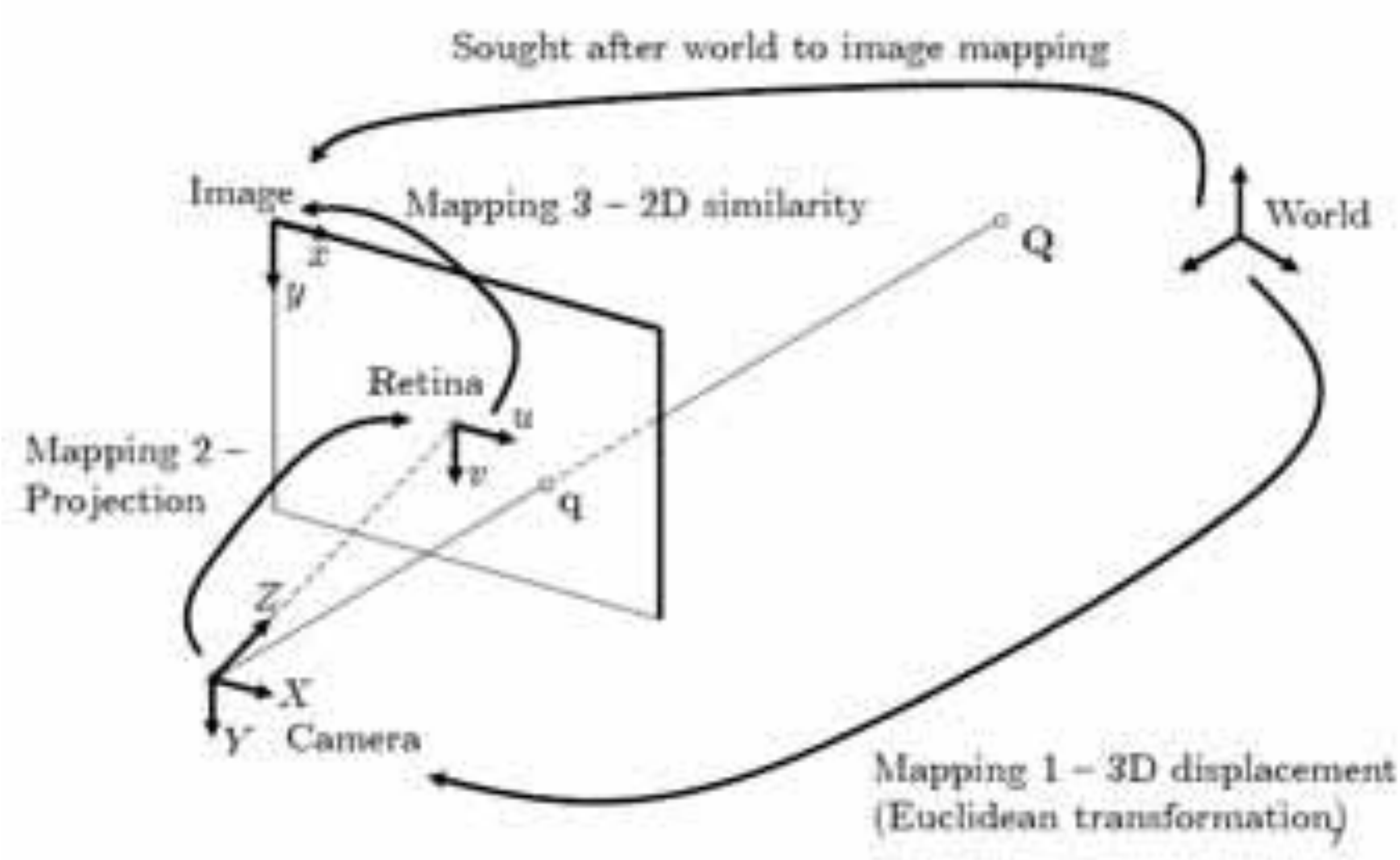
Bundle adjustment



Agarwal et al., Building Rome in one day, ICCV 2009

Camera Modeling

Mapping a 3D point to a 2D image point : 3 mappings



Camera Modeling

First mapping : From 3D world to Camera

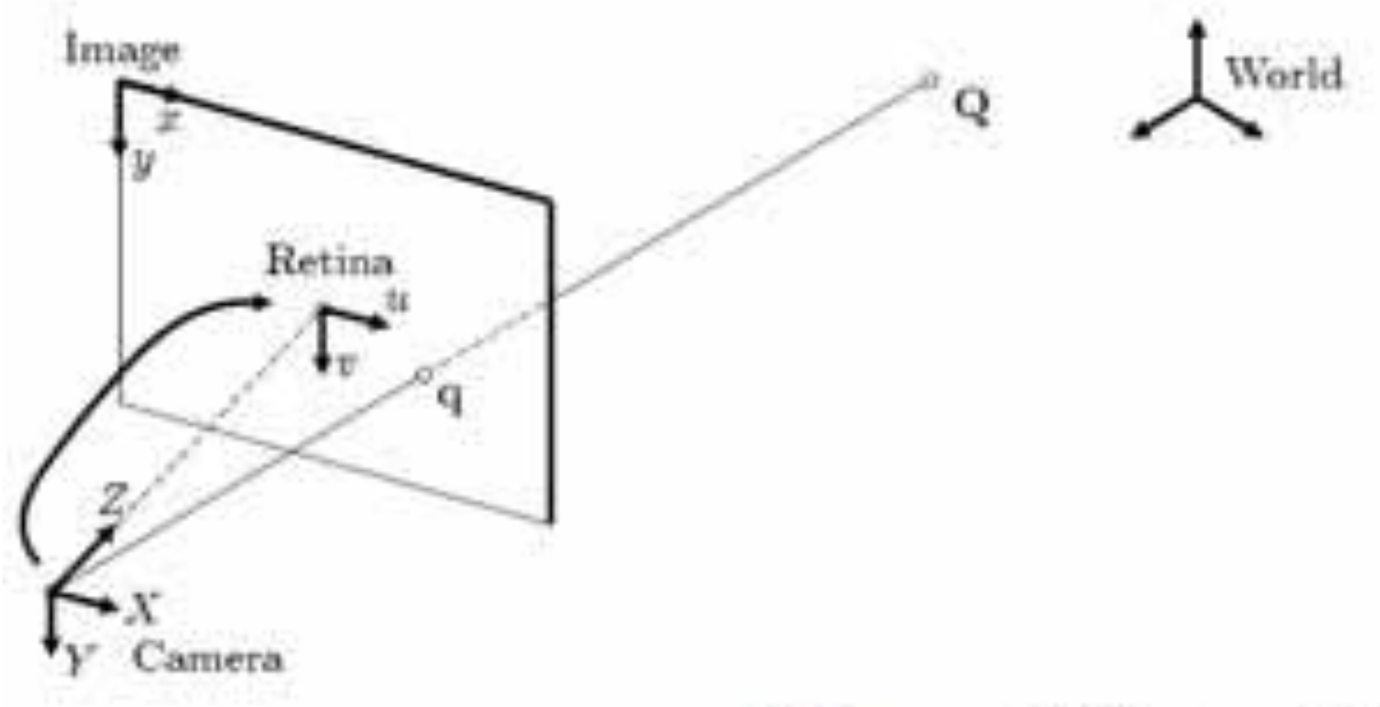
- Models camera displacements : position and orientation
- In homogeneous coordinates:

$$Q_c = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} Q$$

- **R** is a (3 x 3) rotation matrix : $R^T R = I$ and det(R)= 1
- **t** is a (3 x 1) translation vector
- **Q** is the world homogeneous coordinates of the 3D point $Q \sim \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$

Camera Modeling

Second mapping : From Camera to Retina



Maps $\mathbb{R}^3$ to $\mathbb{R}^2$

Camera Modeling

Second mapping : From Camera to Retina

- Camera-centered 3D point coordinates : $Q_c \sim (X_c \ Y_c \ Z_c \ 1)^T$
- Retina-centered coordinates:

$$u = f\frac{X_c}{Z_c} \quad \text{and} \quad v = f\frac{Y_c}{Z_c}$$

in homogeneous coordinates

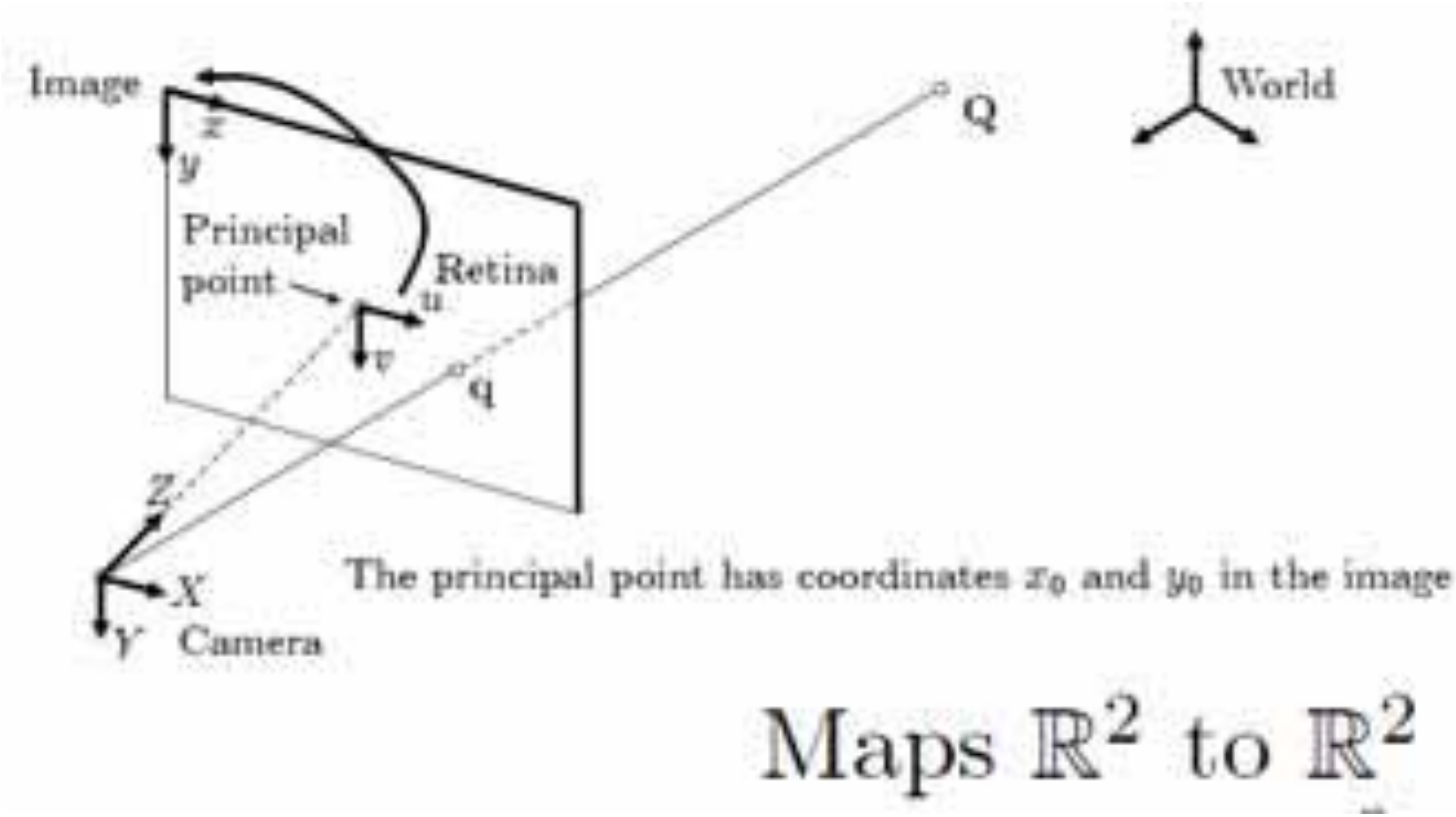
$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} fX_c \\ fY_c \\ Z_c \end{pmatrix}$$

- In matrix form, the projection can be written as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}$$

Camera Modeling

Third mapping : From Retina to Image



Camera Modeling

Third mapping : From Retina to Image

- k_x, k_y: are the density of pixels along *u* and *v*, e.g. in number of pixels per mm
- We have:

$$x = k_x u + x_0 \quad \text{and} \quad y = k_y v + y_0$$

which in matrix form gives:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} k_x & 0 & x_0 \\ 0 & k_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$$

Camera Modeling

Mapping 2+3: Camera to Image

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} k_x & 0 & x_0 \\ 0 & k_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} fk_x & 0 & x_0 \\ 0 & fk_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}}_{K} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}$$

- ❑ K is the **camera calibration matrix**
- ❑ K contains the ‘internal’ or ‘intrinsic’ camera parameters

Camera Modeling

Mapping 1+2+3: World to Image

$$q \sim \underbrace{\begin{pmatrix} fk_x & 0 & x_0 \\ 0 & fk_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}}_{K} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & & & t \\ 0 & 0 & 0 & 1 \end{pmatrix} Q$$

or

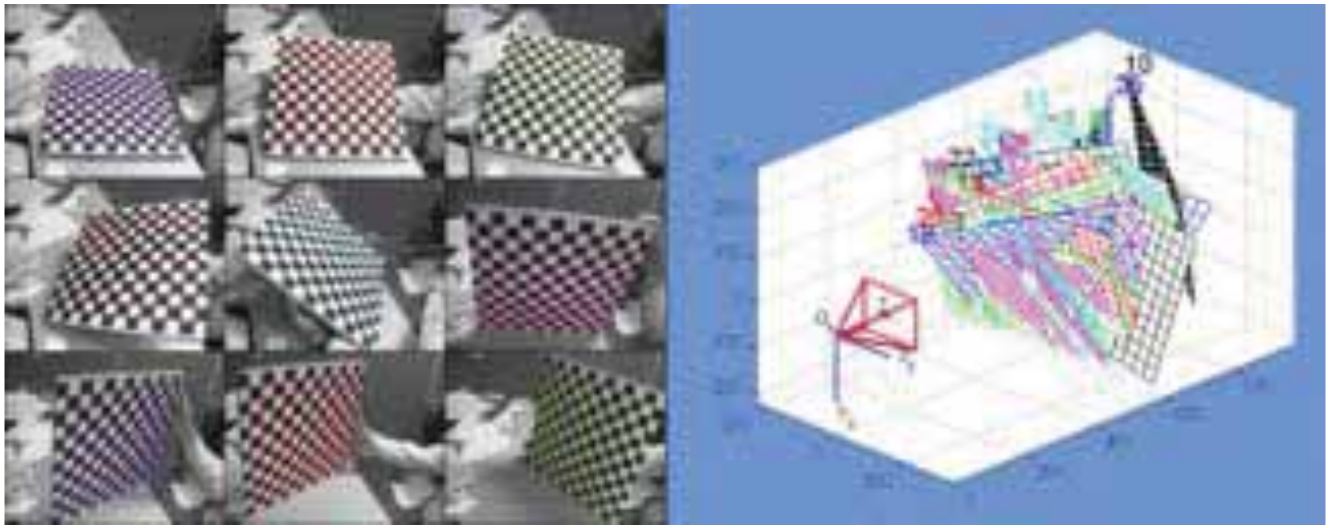
$$q \sim \underbrace{(KR \;\; Kt)}_{P} Q$$

- **P** is the **perspective projection matrix**: P is a (3 x 4) matrix
- **K** contains the ‘internal’ or ‘intrinsic’ camera parameters
- **R** and **t** are the ‘extinsic’ or ‘external’ camera parameters, also called the pose of the camera

Calibration

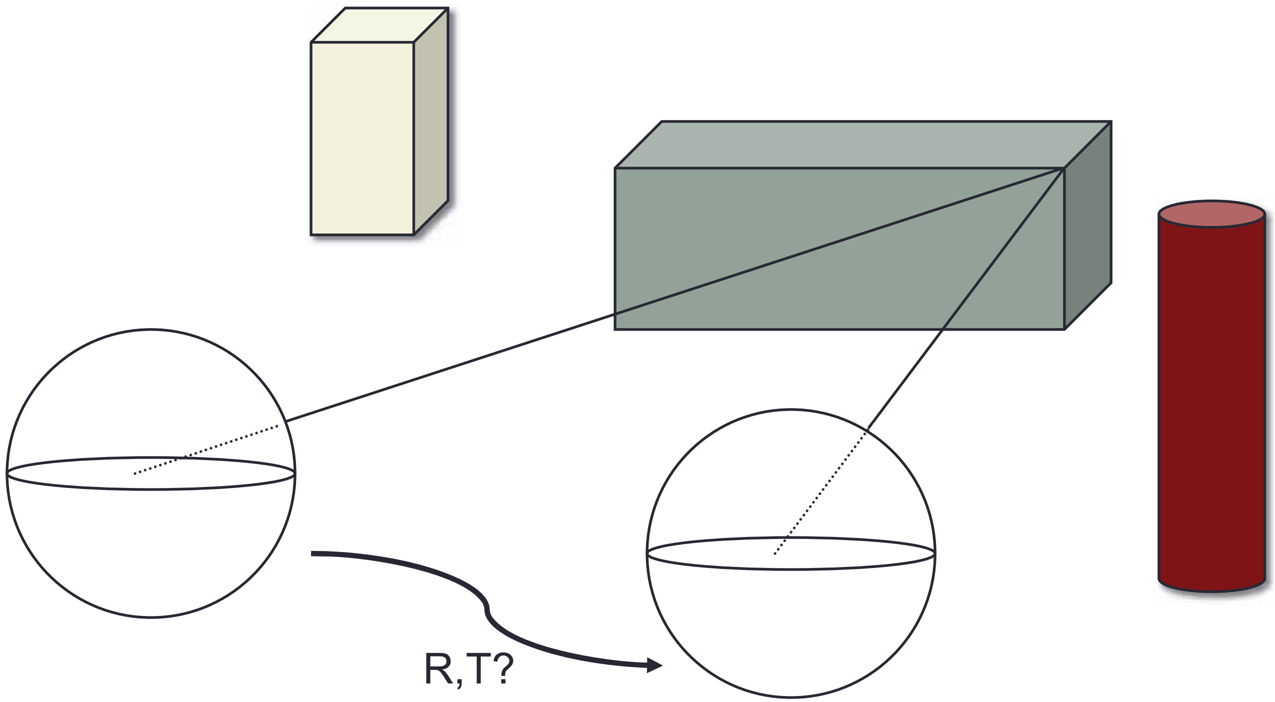
The goal is to estimate P

$$q \sim \underbrace{(KR \;\; Kt)}_{P} Q$$



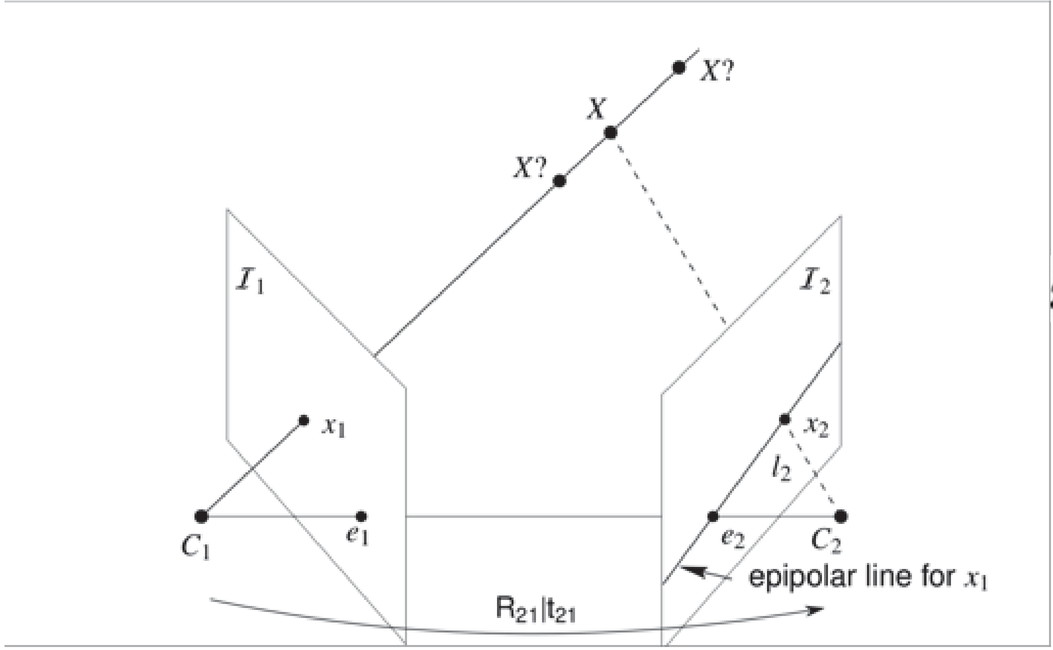
Camera calibration Toolbox for Matlab (Bouguet)

Structure from motion



Epipolar Geometry

The **epipolar geometry** is the intrinsic projective geometry between two views.
It is independent of scene, and only depends on the cameras' internal parameters and relative pose.



$$x_2^T [t_{21}]_\times R_{21} x_1 = x_2^T E_{21} x_1 = 0$$

Epipolar Geometry

$E = [T]_X R$

E is of rank 2

$$\det(E) = 0$$

$$E = U \begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$$

$$EE^T E - \frac{1}{2}\mathrm{Trace}(EE^T)E = 0$$

Epipolar Geometry: 8 pts algorithm

E can be estimated thanks the linear 8 points algorithm

$$p'Ep = 0 \qquad E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}$$

$$p = (x, y, 1) \quad p' = (x', y', 1)$$

$$x'xe_{11} + x'ye_{12} + x'e_{13} + y'xe_{21} + y'ye_{22} + y'e_{23} + xe_{31} + ye_{12} + e_{33} = 0$$

For n correspondences:

$$Ae = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} e = 0$$

Epipolar Geometry: 8 pts algorithm

We have an homogeneous system Ae=0 which can be estimated up to scale by least mean square

$$\min_e \sum_{i=1}^{n} \|Ae\|^2 \quad \text{such as} \quad \|e\| = 1$$

Epipolar Geometry: 5 pts algorithm

$$Ae = \begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} e = 0$$

With n = 5, the problem is over determined (matrix 5 *9) :

$$E = xX + yY + zZ + wW$$

but

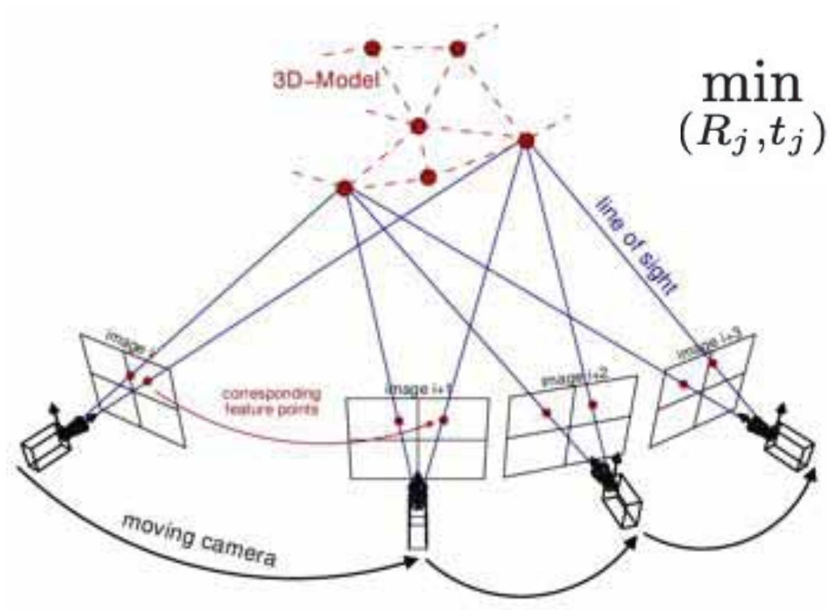
$$\det(E) = 0$$

$$EE^T E - \frac{1}{2}\mathrm{Trace}(EE^T)E = 0$$

D. Nister, « An efficient solution to the five-point relative pose problem », PAMI 2004

Bundle Adjustment

- In multiple view geometry, we can conjointly refine the 3D position of the cameras (R,t) and the 3D reconstruction



$$\min_{(R_j, t_j)} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij}\, d(K_j R_j [I| - t_j]\, X_i, x_{ij})^2$$

$v_{ij} = 0$ If point i is invisible in image j

$v_{ij} = 1$ If point i is visible in image j

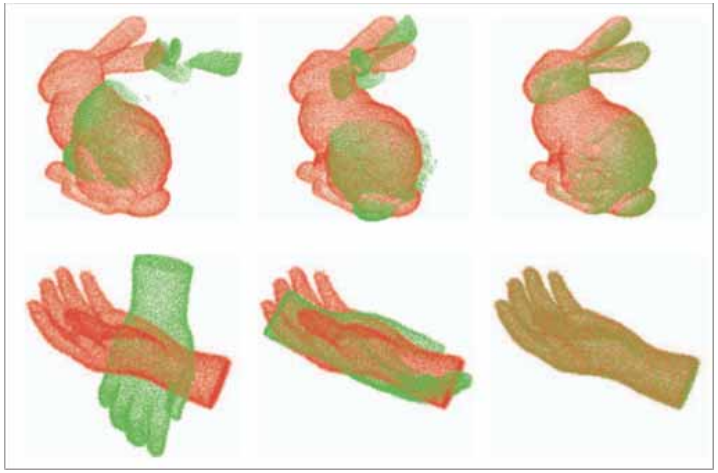
nonlinear least-squares algorithms such as Levenberg Marquardt

Y. Furukawa, J. Ponce, Accurate, Dense, and Robust Multi-View Stereopsis, PAMI 2010

FROM 3D POINT CLOUD TO 3D MOTION

ICP

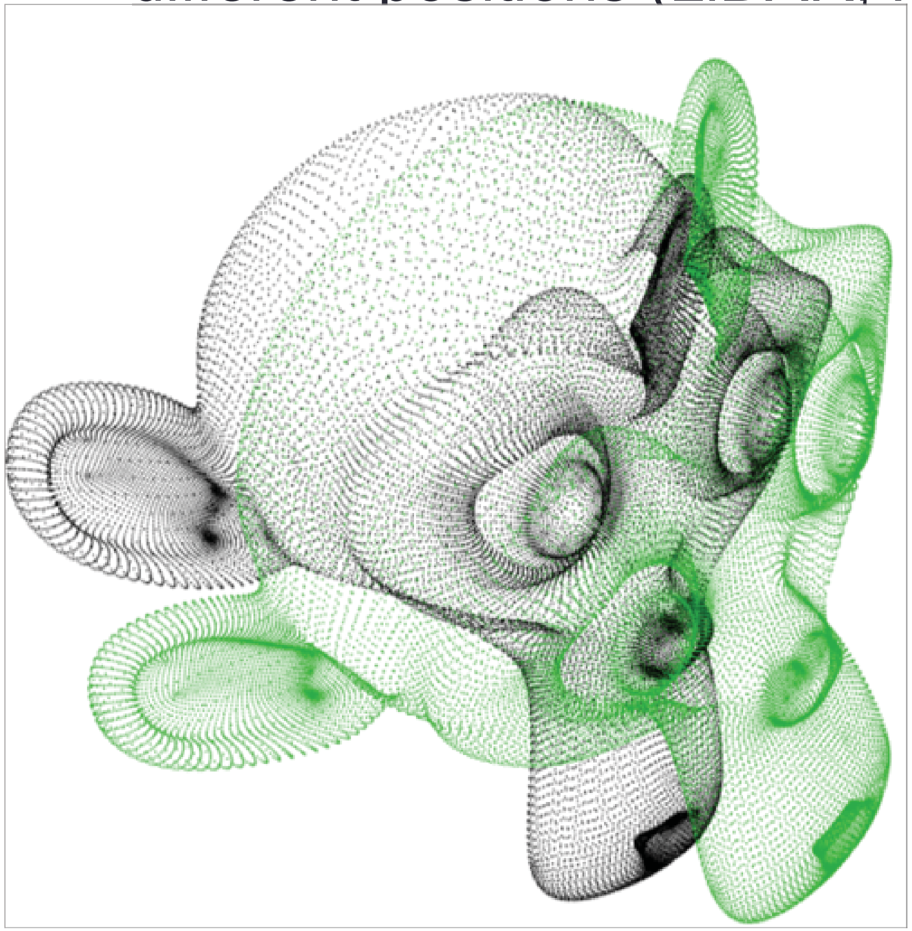
Dense RGB-D registration



Yang et al., Go-ICP: Solving 3D registration efficiently and globally optimally, ICCV 2013

3D sensors

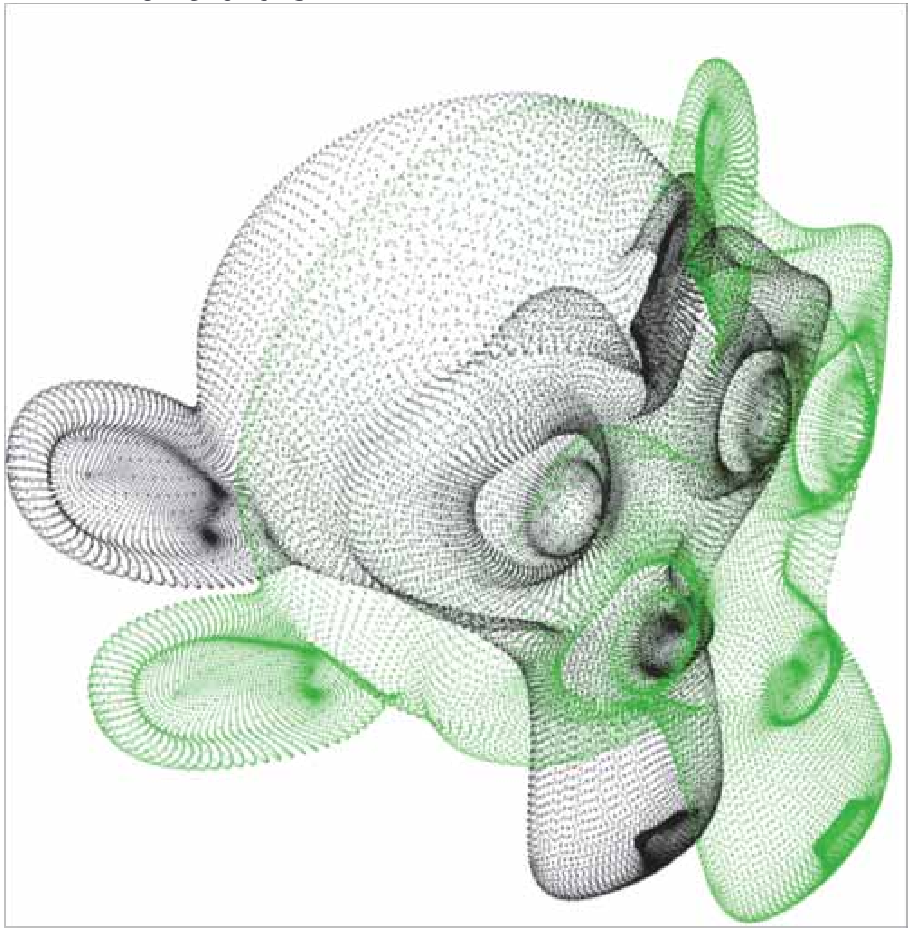
- Let's suppose that we directly have 3D data taken at 2 different positions (LiDAR, RGBD cameras…)



How to register green data on black data?
This registration is related to the 3D sensor motion (R,T)

ICP : Iterative Closest Point

- $\mathcal{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_m\}$ and $\mathcal{Y} = \{\mathbf{Y}_1, \ldots, \mathbf{Y}_m\}$,2 3D point clouds



If $\mathcal{X}$ and $\mathcal{Y}$ are matched, the problem consists in finding R and t solution of :

$$\sum_{i=1}^{n} \|R\mathbf{X}_i + t - \mathbf{Y}_i\|^2$$

ICP : Iterative Closest Point

$$\sum_{i=1}^{n} \|R\mathbf{X}_i + t - \mathbf{Y}_i\|^2$$

The solution is computed by SVD decomposition of the matrix

$$W = UDV^T = \sum_{i=1}^{n} \mathbf{X'}_i * \mathbf{Y'}_i^T$$

with $\mathcal{X}' = \{\mathbf{X}_1 - \bar{\mathcal{X}}, \ldots, \mathbf{X}_m - \bar{\mathcal{X}}\}$

$\mathcal{Y}' = \{\mathbf{Y}_1 - \bar{\mathcal{Y}}, \ldots, \mathbf{Y}_m - \bar{\mathcal{Y}}\}$

$$R = UV^T$$

$$t = \bar{\mathcal{X}} - R\bar{\mathcal{Y}}$$

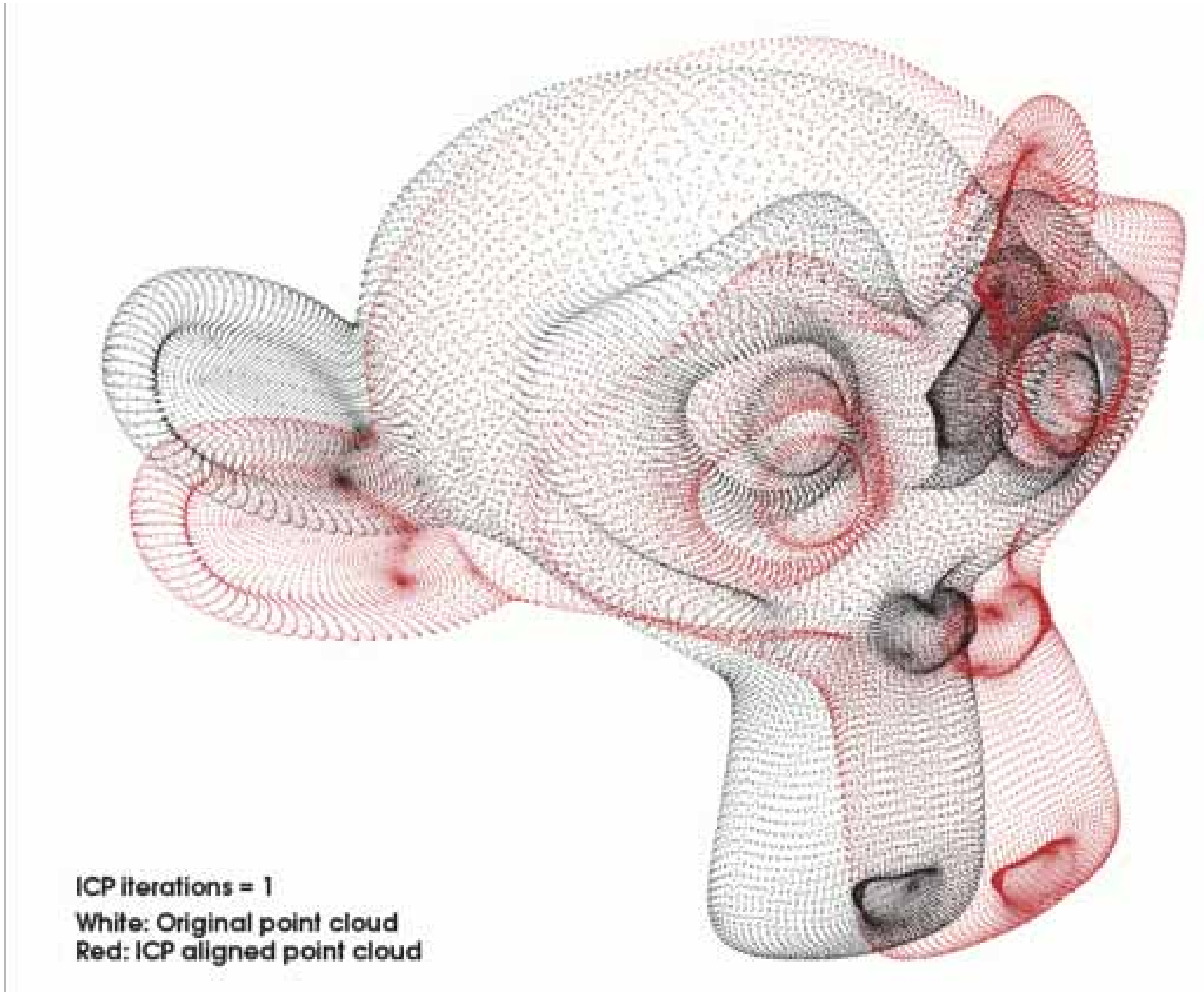
ICP : Iterative Closest Point

- When $\mathcal{X}$ and $\mathcal{Y}$ are not matched, the problem becomes more difficult. It's solved iteratively by the following algorithm:

Algorithm 1 Compute the rigid transformation R and t between two point clouds $\mathcal{X}$ and $\mathcal{Y}$

input : two point clouds $\mathcal{X}$ and $\mathcal{Y}$, an initialization (R_0, t_0), d_{max} threshold.
output : the rigid transformation R and t
$(R, t) \leftarrow (R_0, t_0)$
while not converged **do**
  for $i \leftarrow 1 \text{to} \quad n$ **do**
    $m_i \leftarrow FindClosestPointInY(R\mathbf{X}_i + T)$
    if $\|R\mathbf{X}_i + T - \mathbf{Y}_i\| \leqslant d_{max}$ **then**
      $\omega_i \leftarrow 1$
    else
      $\omega_i \leftarrow 0$
    end if
  end for
  $(R, T) \leftarrow \arg\min \sum_{i=1}^{n} \omega_i \|R\mathbf{X}_i + T - \mathbf{Y}_i\|^2$
end while

ICP : Iterative Closest Point



Dense RGB-D registration

- Let us suppose that we have conjointly depth and color data (RGB-D cameras)

Photometric error

$$e_I(p, X, t) = I(\omega(p, \mathcal{X}, T), t) - I(p, t-1)$$

Warping function

Depth error

$$e_D(p, X, t) = D(\omega(p, \mathcal{X}, T), t) - D(p, t-1)$$

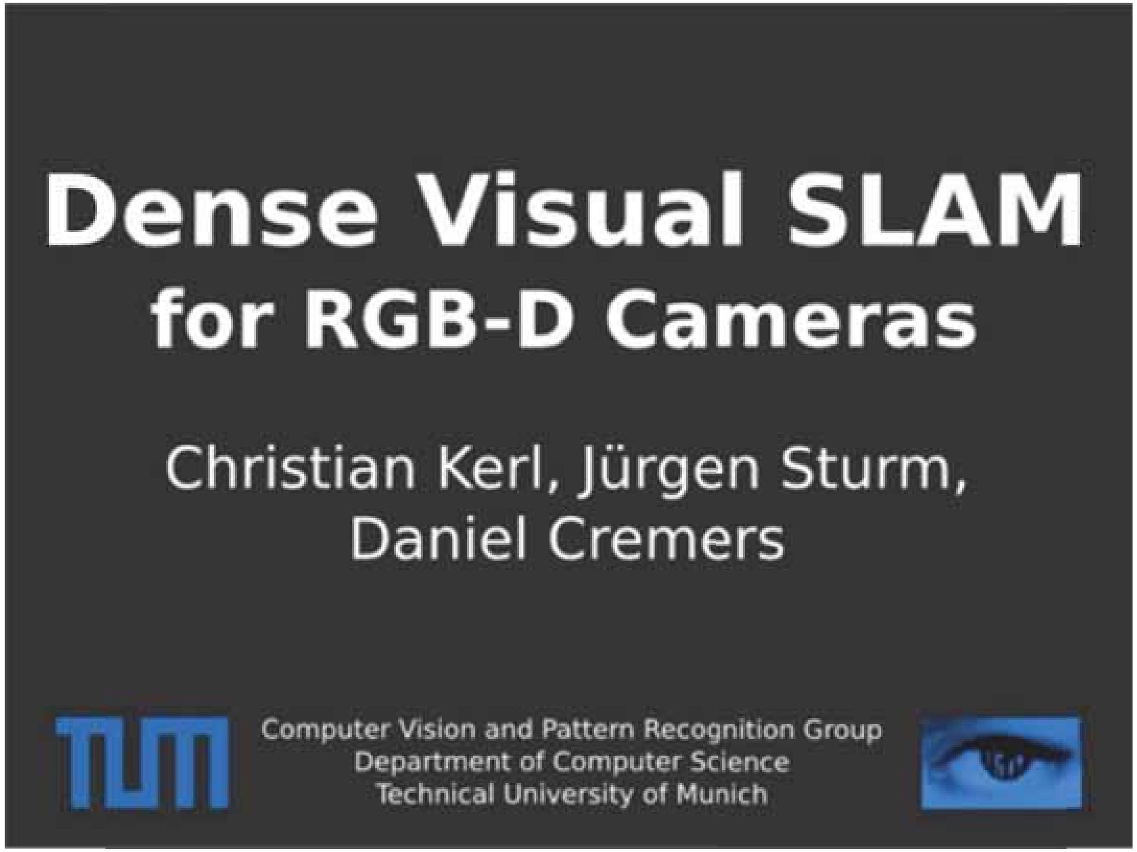
Conjoint minimization

$$\widehat{T} = \arg\min_T \sum_p \rho_I(e_I(p, X, t)) + \lambda \rho_D(e_D(p, X, t))$$

~Optical flow

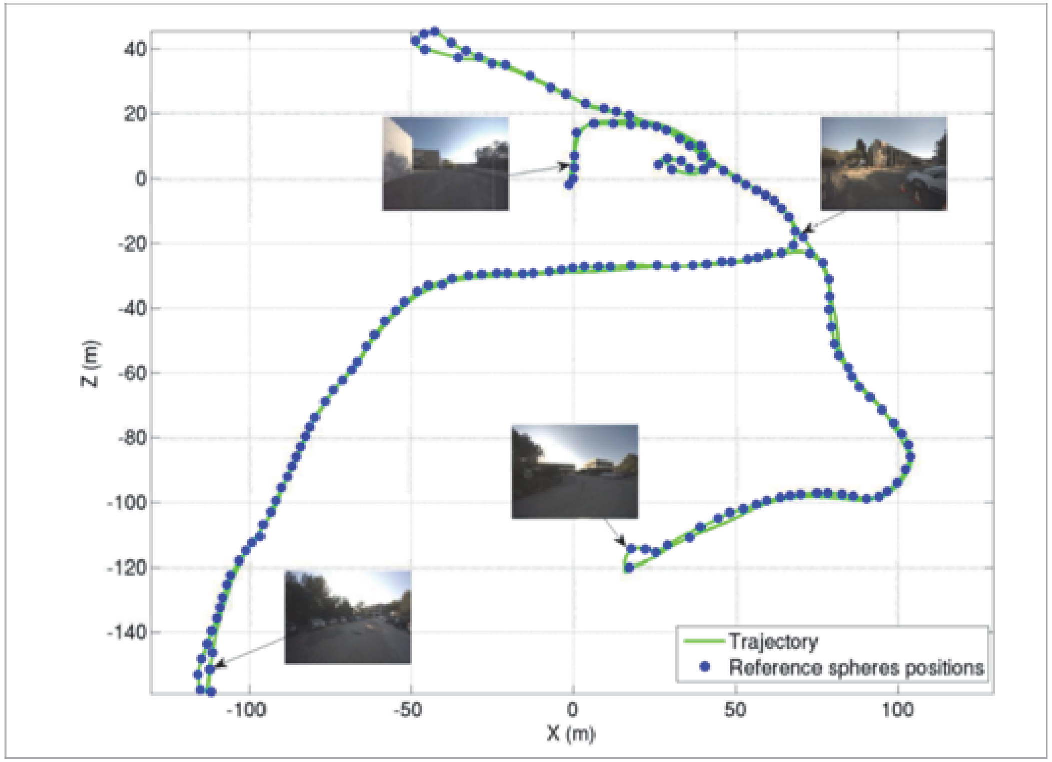
~ ICP point to plane ICP

Dense RGB-D registration



Kerl et al., Dense visual slam for rgbd cameras, IROS 2013

Dense RGB-D registration

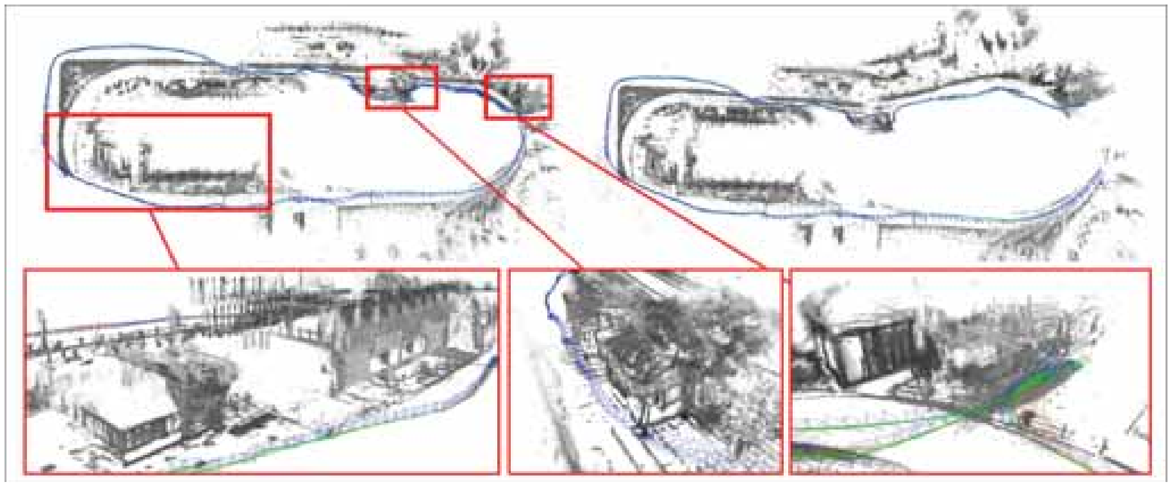


Meilland et al., Dense omnidirectional RGB-D mapping of large scale outdoor environments for real-time localisation and autonomous navigation. Journal of Field Robotics

SLAM

Dense SLAM

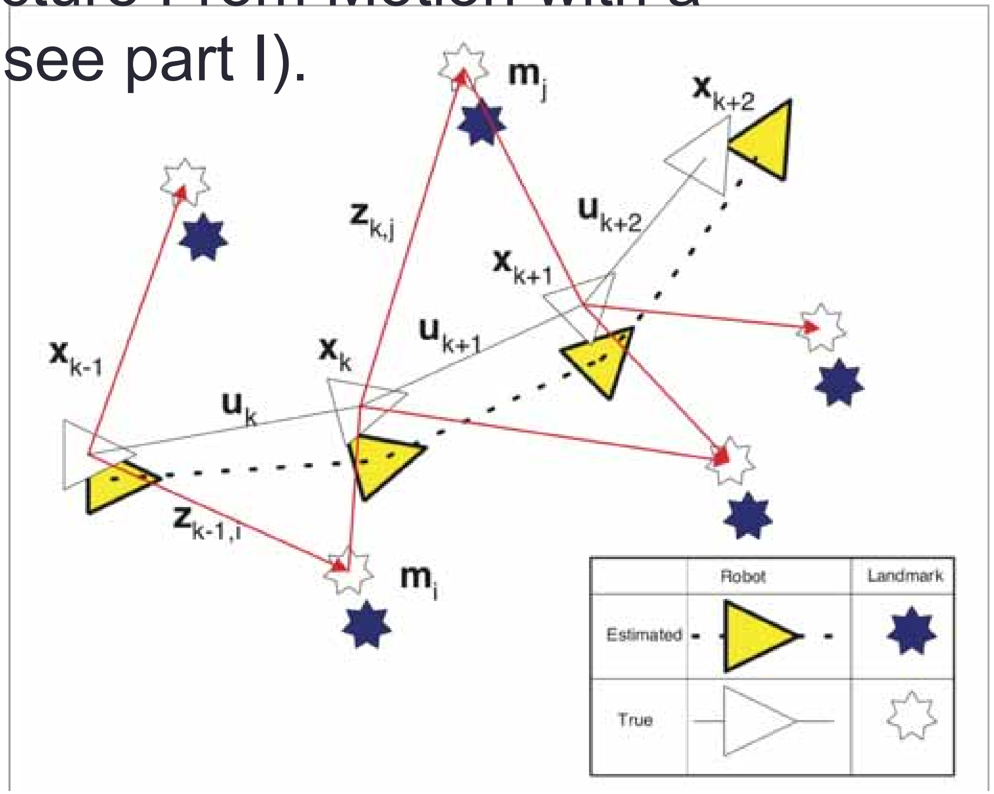
Sparse SLAM



Engel et al. LSD-SLAM, ECCV14

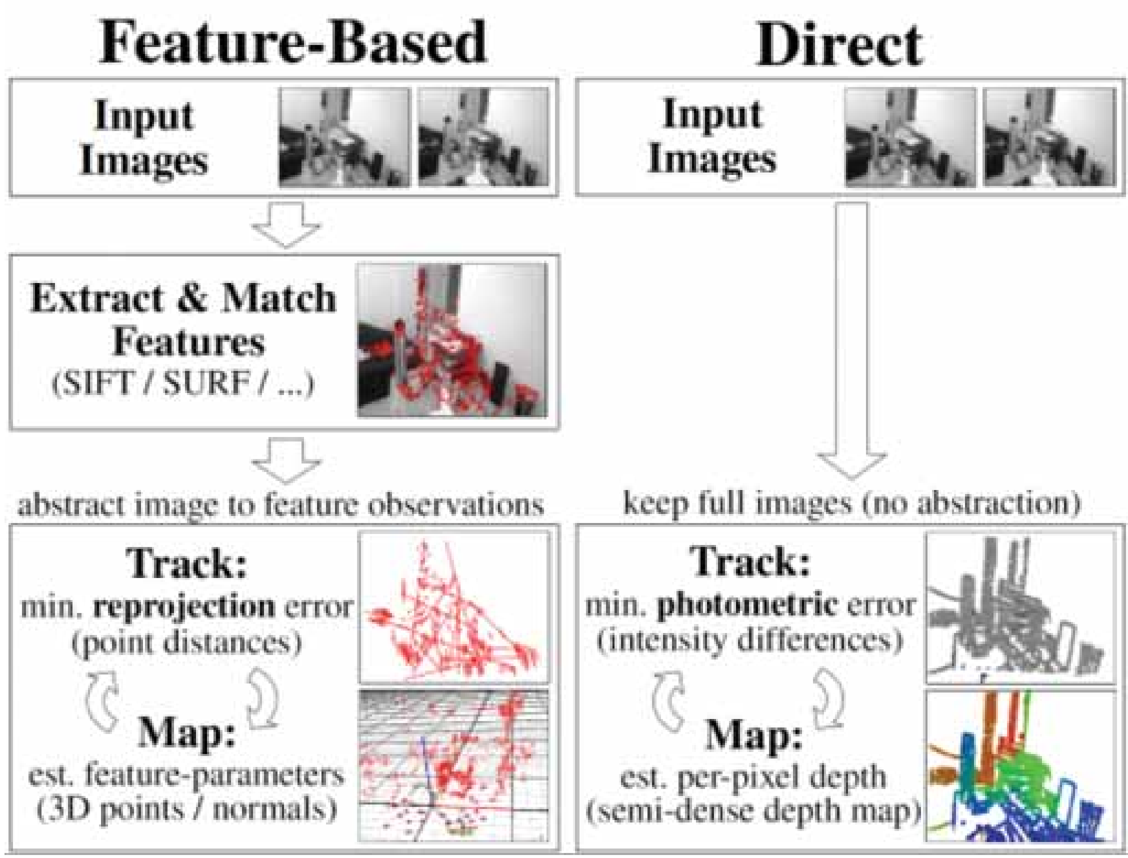
SLAM

- Simultaneous Localization and Mapping methods consist in estimating conjointly the position of the robot and the map of the environment
- Quite similar to SFM : Structure From Motion with a probabilistic modelization (see part I).



SLAM : Sparse vs Direct

Sparse:
- ✓ Large baseline
- ✓ Sparse 3D reconstruction
- ✓ Less time consuming

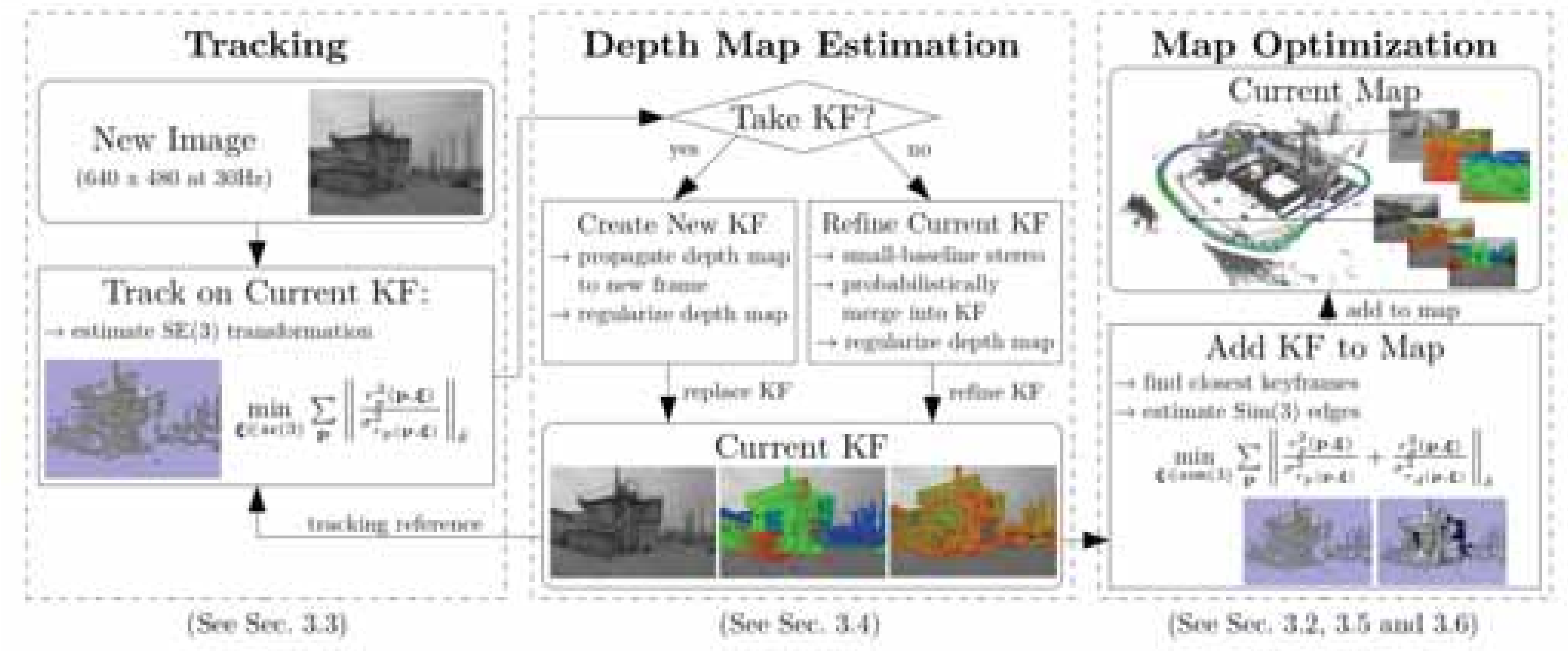


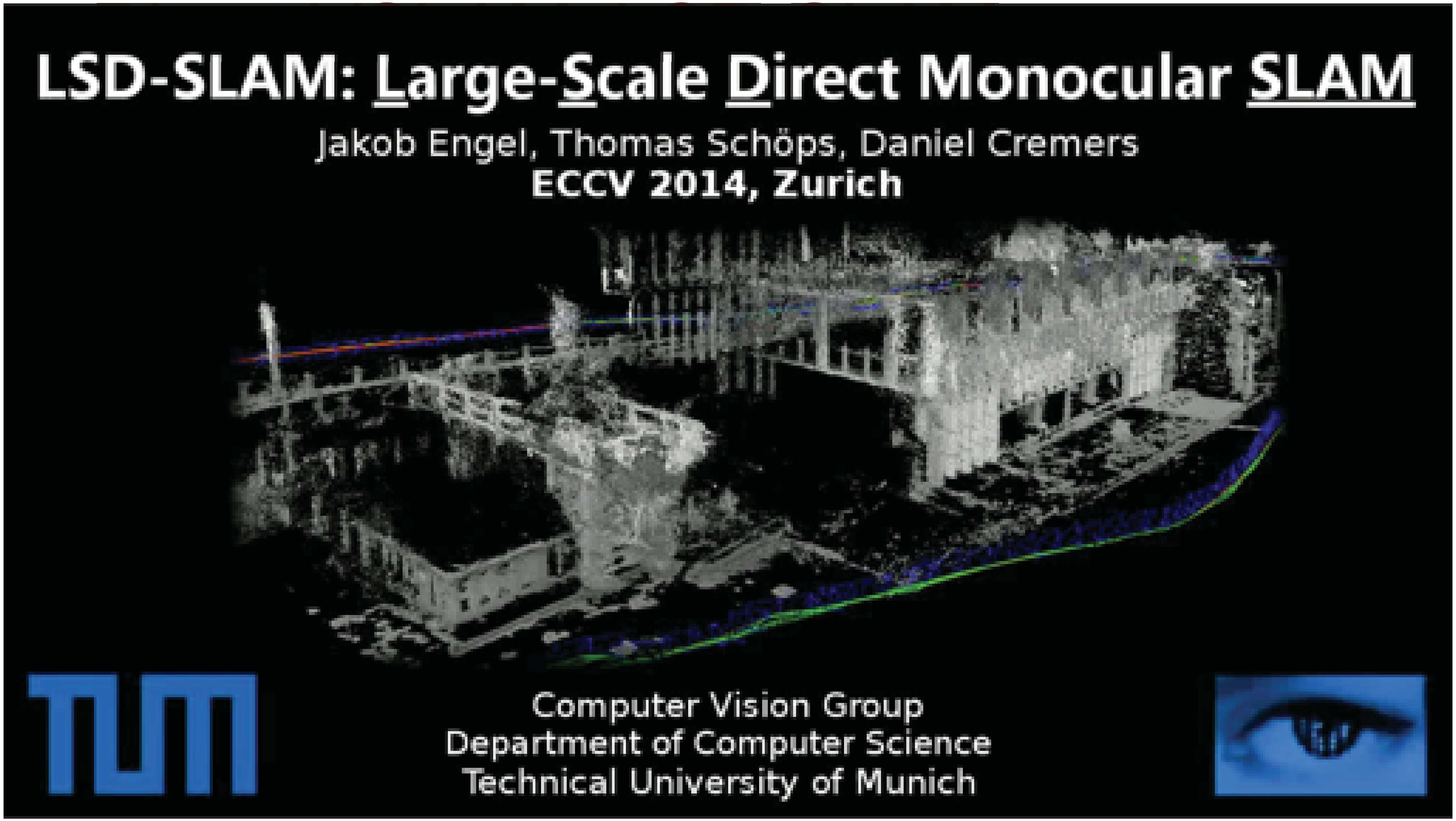
Direct:
- ✓ No feature extraction
- ✓ Robust & Higher accuracy
- ✓ Dense 3D reconstruction

Visual Slam, Computer Vision Group, TUM

Direct VSLAM: LSD-SLAM

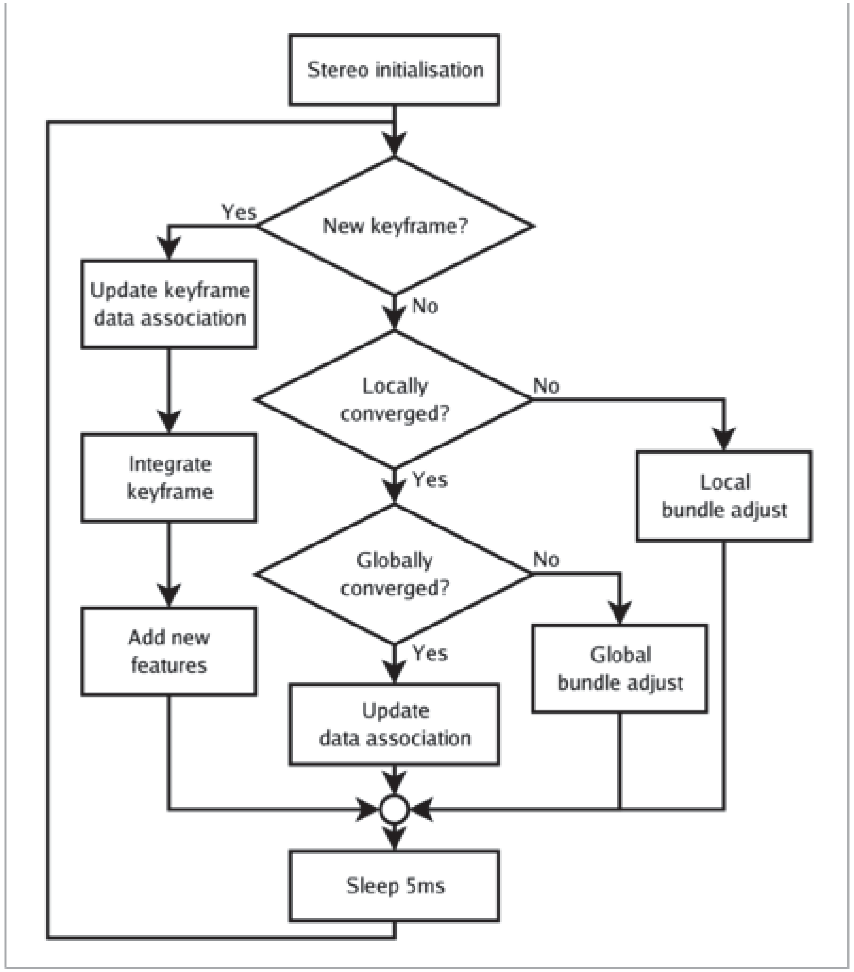
- Large Scale Direct SLAM (Engel 2014)




LSD-SLAM: Large-Scale Direct Monocular SLAM
Jakob Engel, Thomas Schöps, Daniel Cremers
ECCV 2014, Zurich
Computer Vision Group
Department of Computer Science
Technical University of Munich

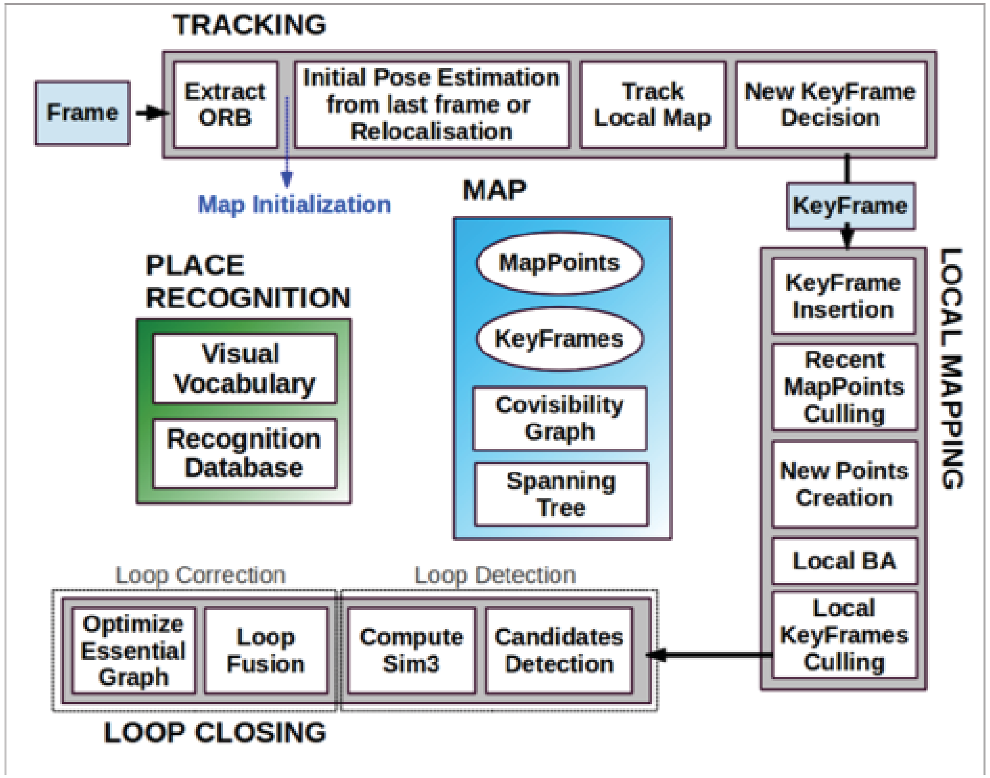
Sparse VSLAM: PTAM

- Parallel Tracking and Mapping (Klein 2016) is a method of Monocular SLAM that runs in real time
- Tracking and mapping are run in parallel on different threads of a multi-core processor



Sparse VSLAM: ORB-SLAM

- ORB-SLAM (Mur-Artal 2015)
- 3 threads : Tracking, local mapping and loop closing



Universidad Zaragoza

ROBOTICS Universidad de Zaragoza

Instituto Universitario de Investigación en Ingeniería de Aragón Universidad Zaragoza

ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

Raúl Mur-Artal and Juan D. Tardós

raulmur@unizar.es tardos@unizar.es

SOME OPEN PROBLEMS

Multifocalities

Multimodalities

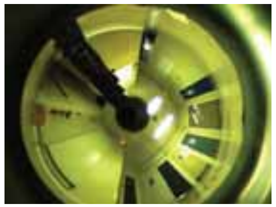
Visual Odometry with external sensors

Dynamic scenes

Multifocalities

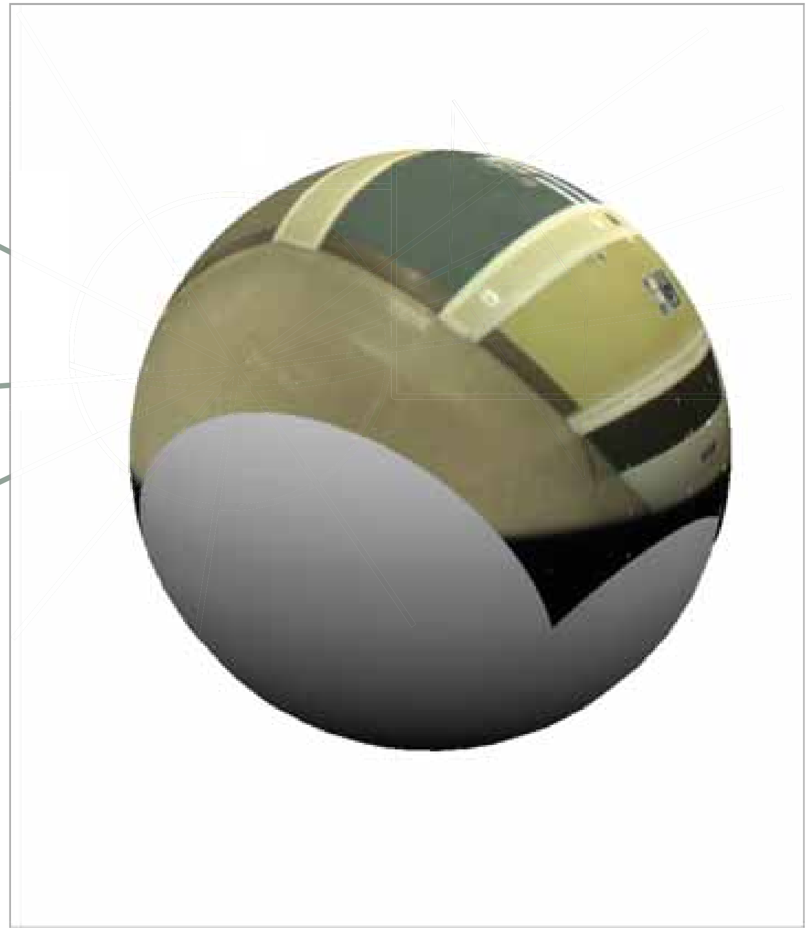




1,2,3

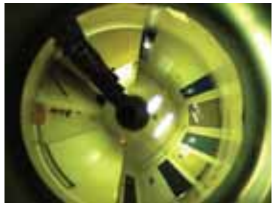
$\Rightarrow$



[1] C. Geyer and K. Daniilidis. Catadioptric projective geometry. IJCV 2001.
[2] J. Courbon, Y. Mezouar, L. Eck, and P. Martinet. A generic fisheye camera model for robotic applications. IROS 2007.
[3] X. Ying and Z. Hu. Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model. ECCV 2004.

Multifocalities

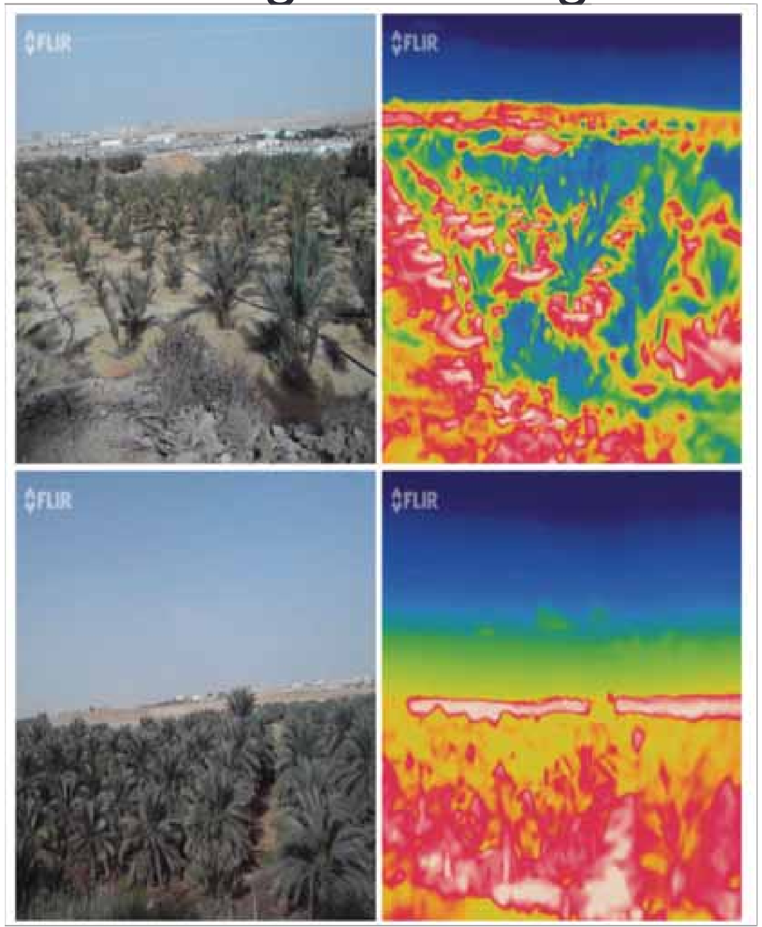
- For calibrated central cameras, epipolar geometry is still valid (Essential Matrix, Homography…)->SFM pipeline developed for perspective cameras can be used.
- Feature detection :
    - HARRIS, SIFT, SURF, FAST…are not valid!

How to adapt them?
How to compare two images coming from two different camera sensor?

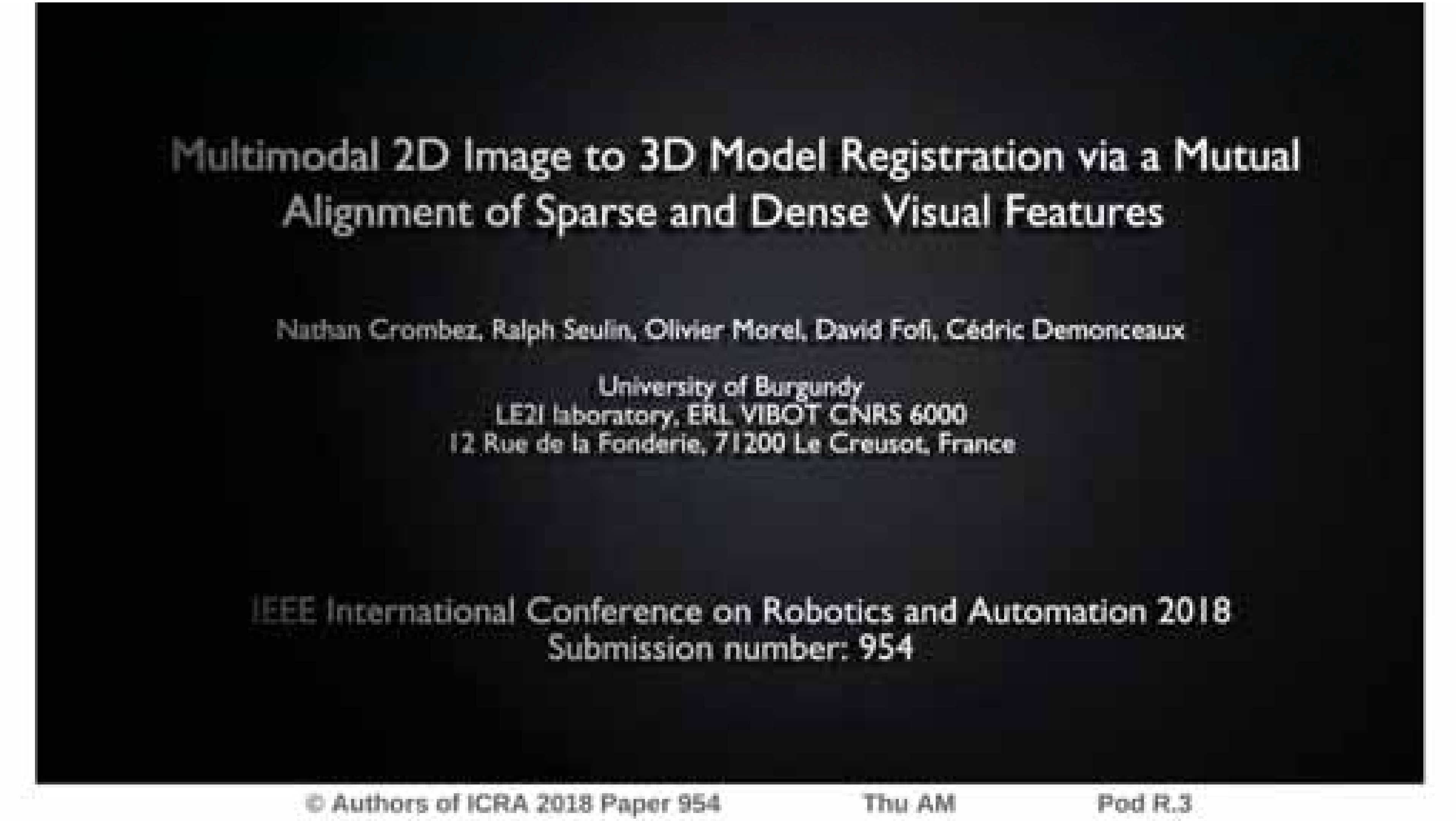

Multimodalities

- How to compare two images coming from two modalities?



RGB vs. Thermal images

Multimodalities



Visual odometry using external knowledge

- Epipolar geometry:
  - Non calibrated : 8 pts algorithm (Fundamental Matrix)
  - Calibrated : 5 pts algorithm (Essential Matrix)
  - Can we reduce the number of points (Important in RANSAC)?

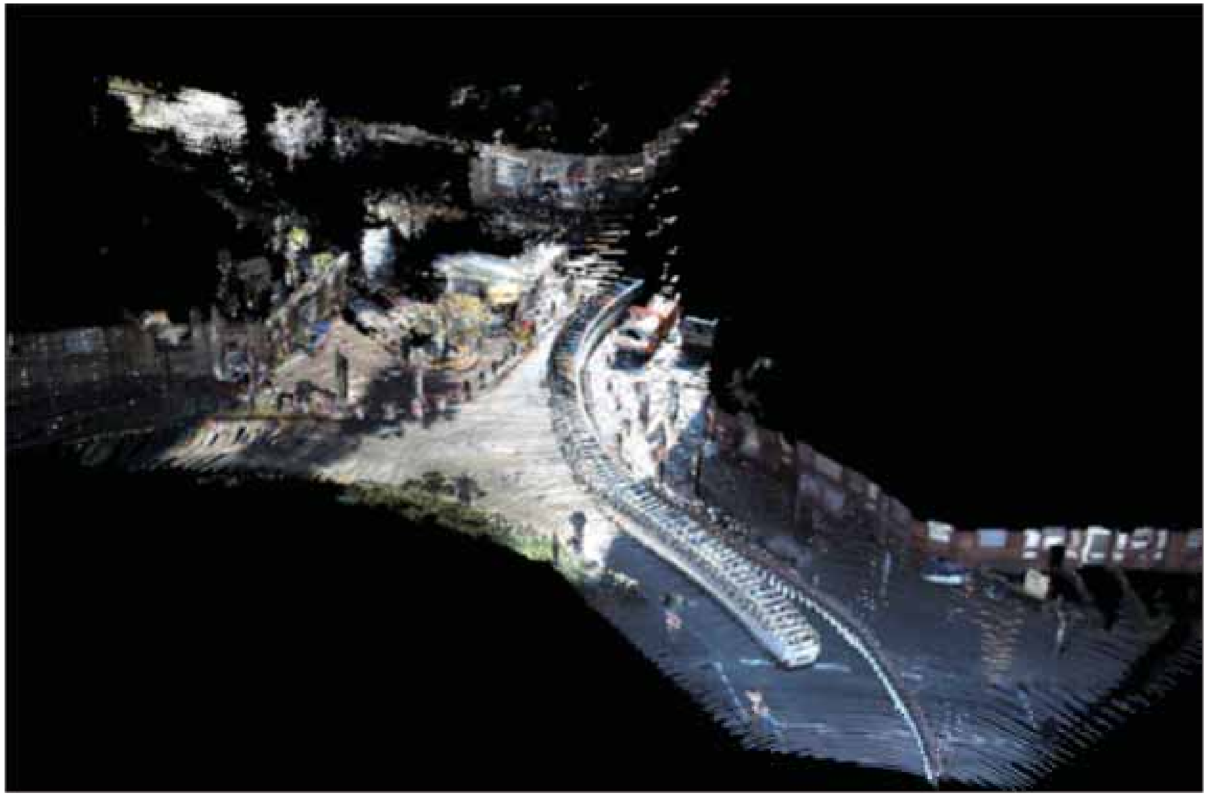
Yes : if IMU information are provided
3 pts algorithm (*Fraundorfer, ECCV 2010*)

Yes : if information related to the scene structure
4 pts algorithm (Homography)

Yes : if information related to the scene structure + IMU
2 pts algorithm (Homography)(*Saurer, PAMI 2018*)

Dynamic Scene

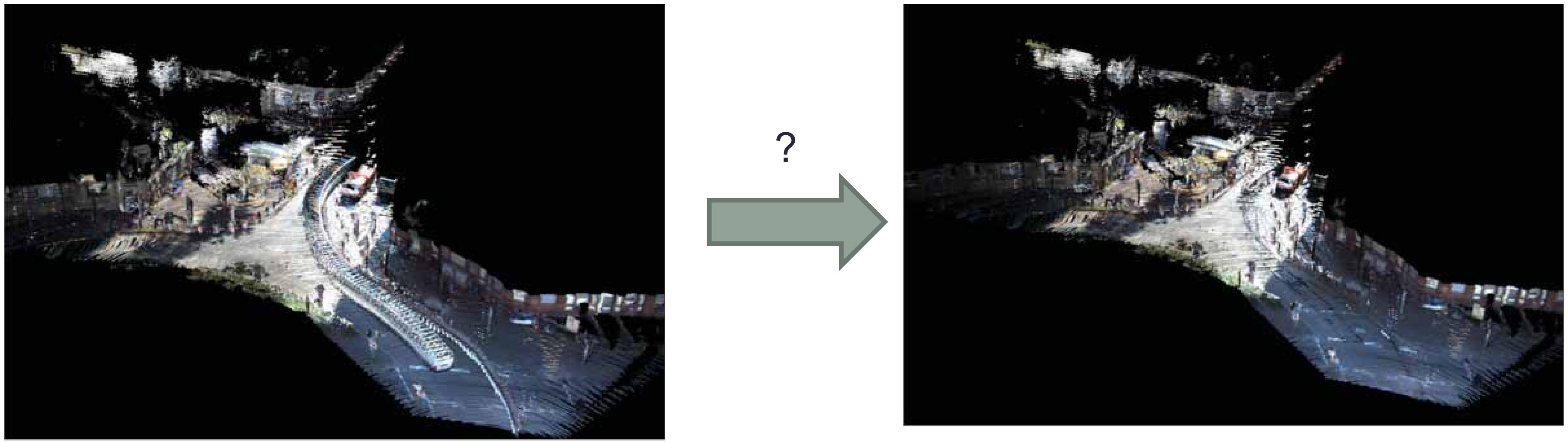
- In a dynamic world, previous methods do not work



D.P. Paudel, C. Demonceaux, A. Habed, P. Vasseur, I.S. Kweon. 2D-3D Camera fusion for Visual Odometry in outdoor environments. IROS 2014

Dynamic Scene

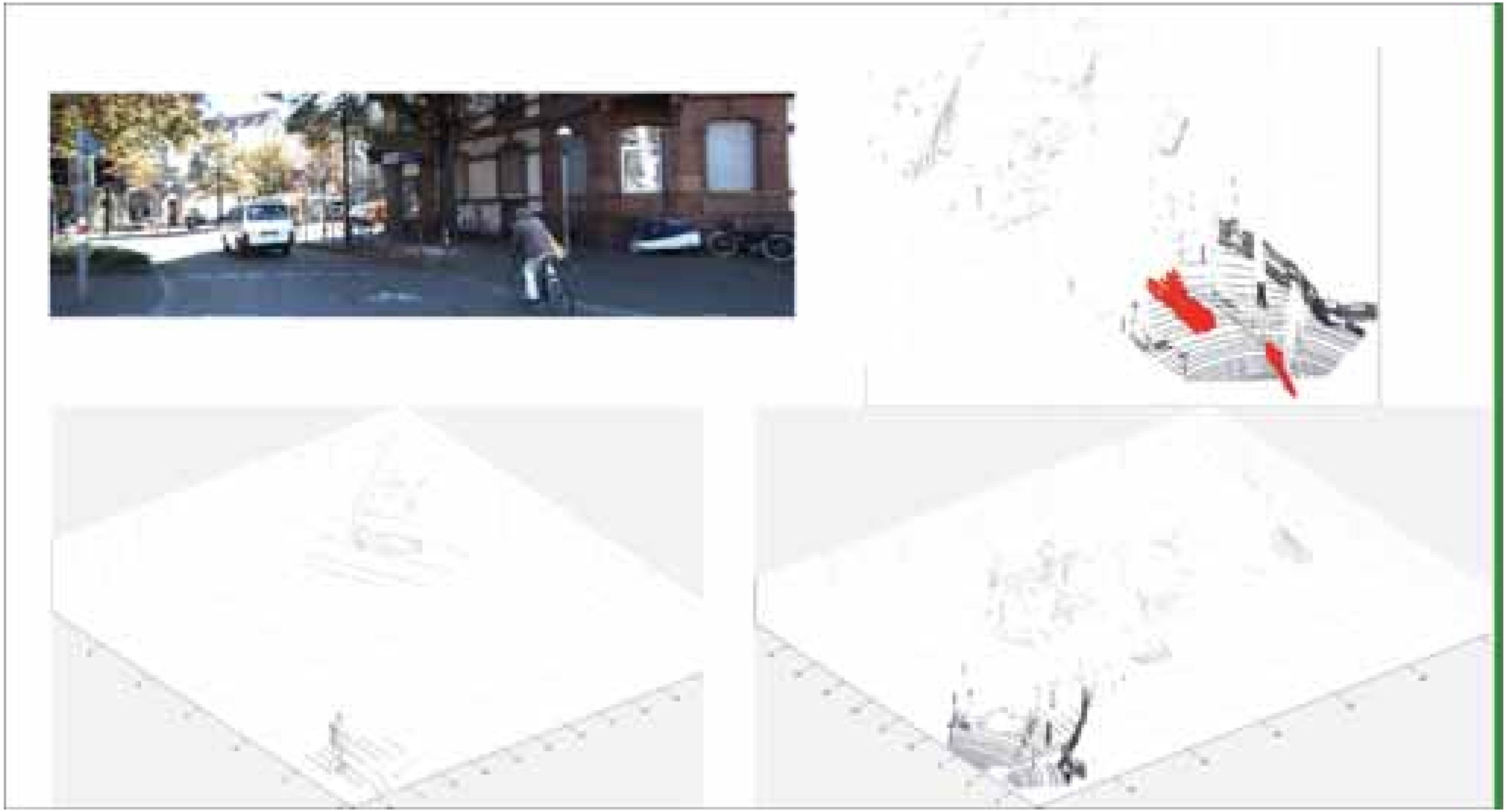
- In a dynamic world, previous methods do not work



Dynamic Scene

C. Jiang, D. P. Paudel, Y. Fougerolle, D. Fofi, C. Demonceaux. Static and Dynamic Objects Analysis as a 3D Vector Field. 3DV 2017

Dynamic Scene



Dynamic Scene

High Quality Reconstruction of Dynamic Objects using 2D-3D Camera Fusion

Multimedia Attachment for IEEE International Conference on Image Processing (ICIP'17)

Cansen Jiang, Dennis Christie, Danda Pani Paudel, and Cedric Demonceaux

SLAM?

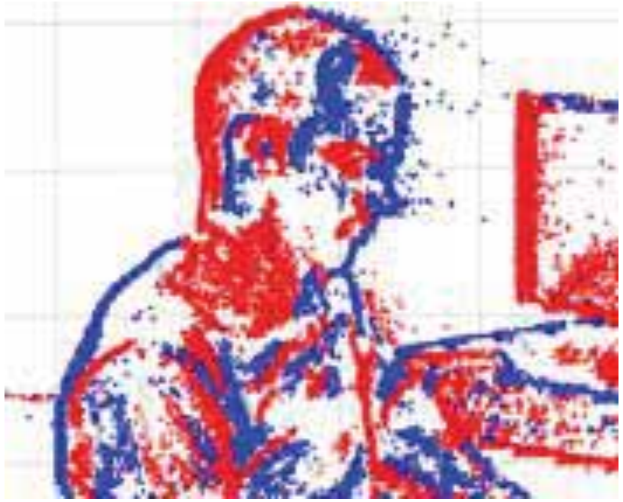
Some challenges :

- **Photometric calibration**. Pixels corresponding to the same 3D point may have different intensities across images
- **Motion bias**. Running a VO method on the same sequence forward and backward sometimes can result in significantly different performances.
- **Rolling shutter effect**. Exposing pixels within one image at different timestamps can produce distortions that may introduce non-trivial errors into VO systems.

N. Yang, R. Wang, W. Goa, D. Cremers, **Challenges in Monocular Visual Odometry: Photometric Calibration, Motion Bias, and Rolling Shutter Effect,** IROS 2018

New cameras

Event cameras:



Zhou et al. Semi-Dense 3D Reconstruction with a Stereo Event Camera. ECCV 2018

New cameras

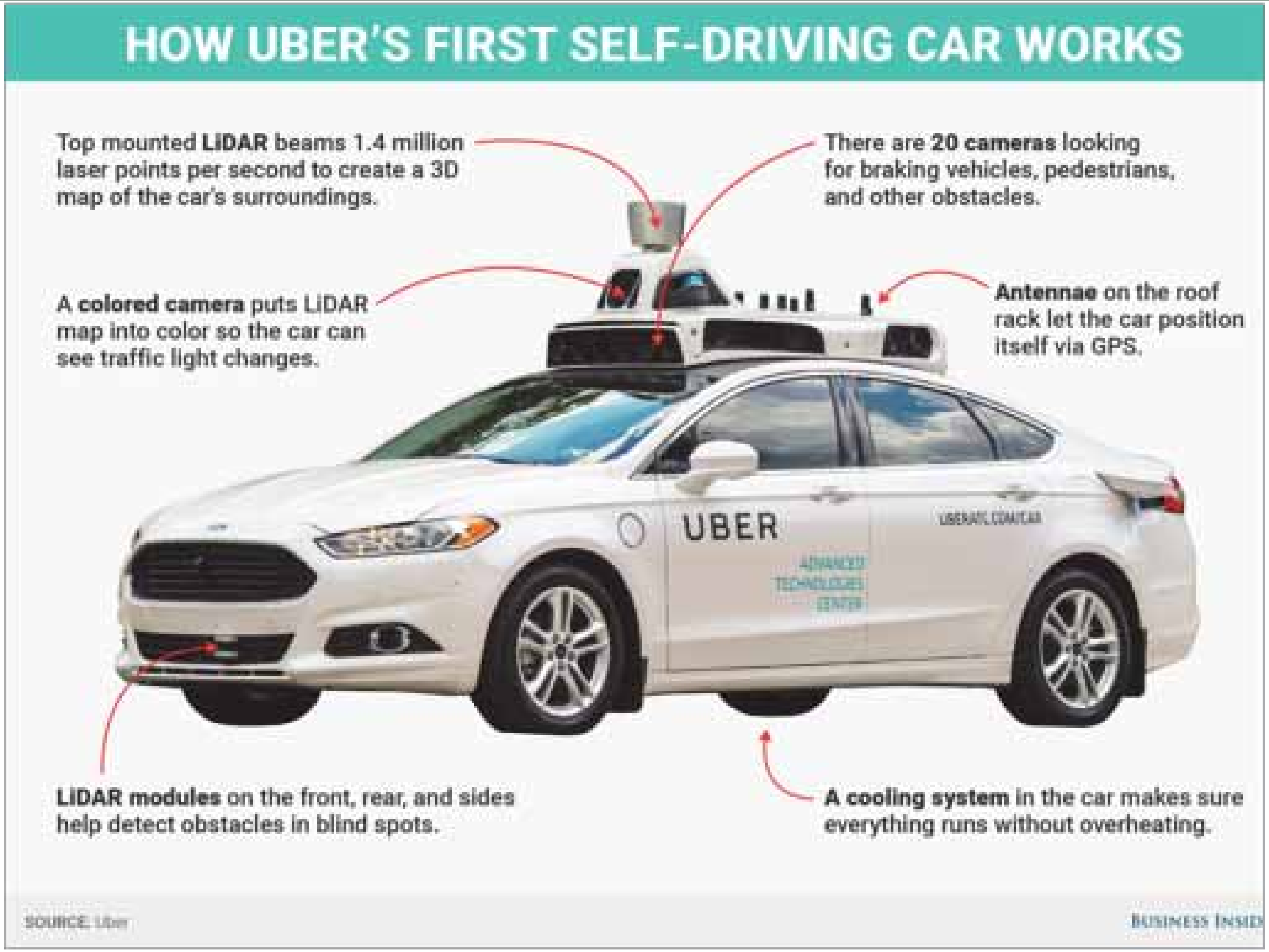
Plenoptic cameras:

array of microlenses which captures small image from different viewpoints -> 3D Reconstruction

Raytrix Plenoptic camera

Crombez et al. Reliable Planar Object Pose Estimation in Light Fields From Best Subaperture Camera Pairs. *RAL 2018*

Conclusion & discussions



"In my view, (LiDAR) is a crutch that will drive companies to a local maximum that they will find very hard to get out of. Perhaps I am wrong, and I will look like a fool. But I am quite certain that I am not." Elon Musk

?

cedric.demonceaux@u-bourgogne.fr