Summer School in Surgical Robotics

Dynamic visual servoing for surgical robotics

Control II, Thursday, Sept. 10, 14h30-16h

jacques.gangloff@unistra.fr
http://eavr.u-strasbg.fr/

Outline

- Introduction
  - Definitions
  - History & current challenges
  - Classification
  - Examples of tasks
- Measurement estimation
  - Dementhon method
  - Image Jacobian method

Outline

- Control
  - Kinematic control
    - 2D
    - 3D
  - Dynamic control
    - Image acquisition model
    - Manipulator model
    - Typical dynamic visual loop
- Conclusion

1.1 Definitions

- *Exteroceptive sensor* :

  Sensor that provides measurements on the environment of the system.

  - Examples : camera, ultrasound range sensor, medical imaging system, ...

- *Visual servoing* :

  Control of a mechanical actuated system using exteroceptive visual informations.

1.2 History

Date	Authors	Task	Sampling period
1973	Shirai & Inoue (Tokyo, JP)	Grasping	10s
1979	Prajoux (CNRS, FR)	Tracking	1s
1984	Weiss (CMU, USA)	Tracking	33ms
1992	Chaumette (INRIA, FR)	Tracking	40ms
1996	Corke (CSIRO, AU)	Tracking	16ms
1998	Gangloff (LSIIT, FR)	Tracking	8ms
1999	Nakabo (U. Tokyo, JP)	Tracking	1ms

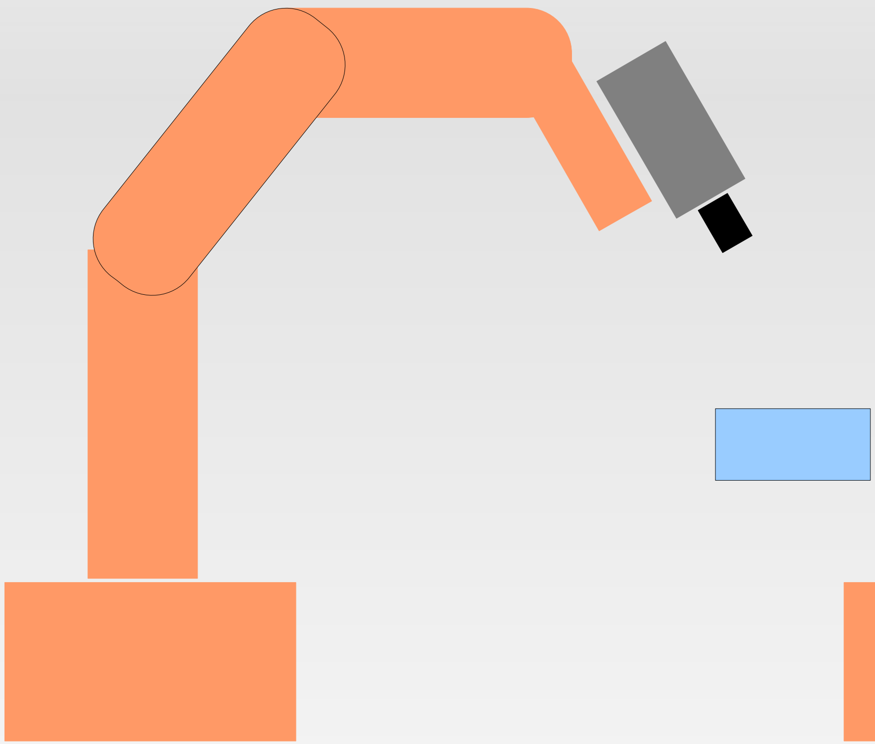
1.2 Some current challenges (situation in France)

- Robust visual servoing (INRIA Rennes, INRIA Sophia, ISIR Paris, LSIIT Strasbourg)
- Visual servoing of parallel robots (LASMEA, Clermont Ferrand)
- Physiological motion compensation (LSIIT Strasbourg, LIRMM Montpellier)
- Ulrasound imaging visual servoing (INRIA Rennes, LIRMM Montpellier, ISIR Paris)
- CT-scan and MRI visual servoing (LSIIT Strasbourg)

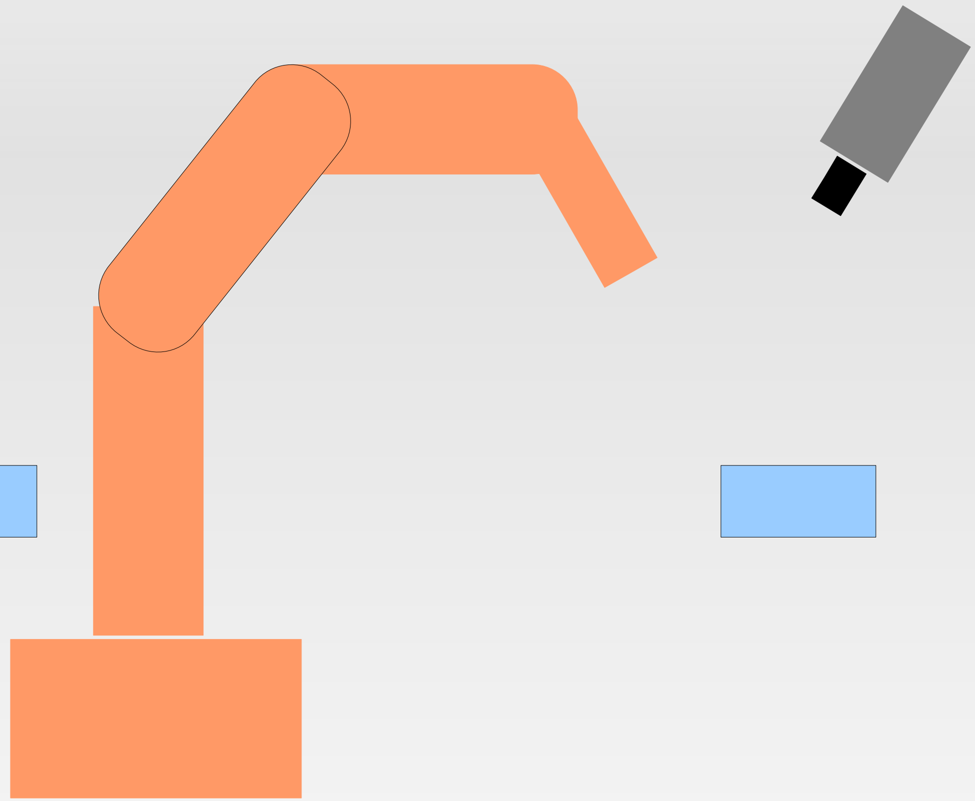
1.3 Classification

- Criteria :
  - Position of the camera(s)
  - Type of control signal
  - Type of measurement
  - Bandwidth

1.3.1 Position of the camera(s)

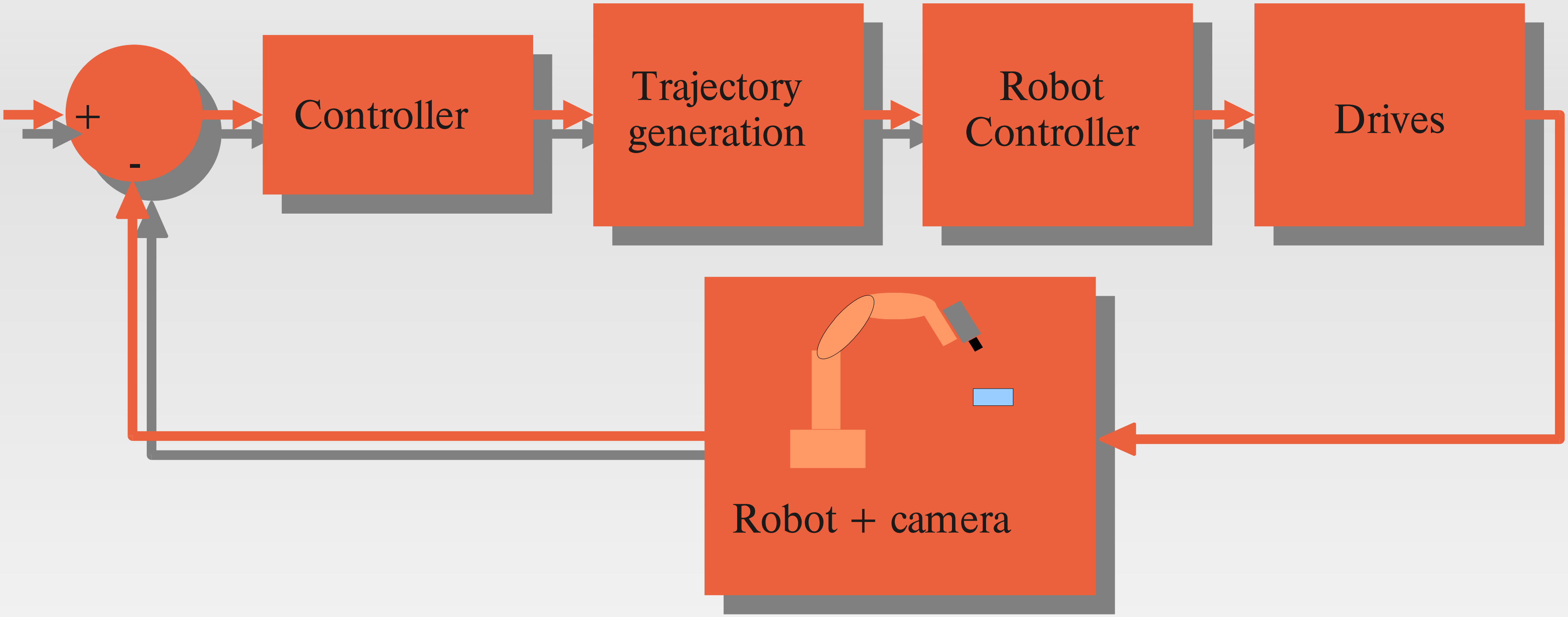


Eye in hand



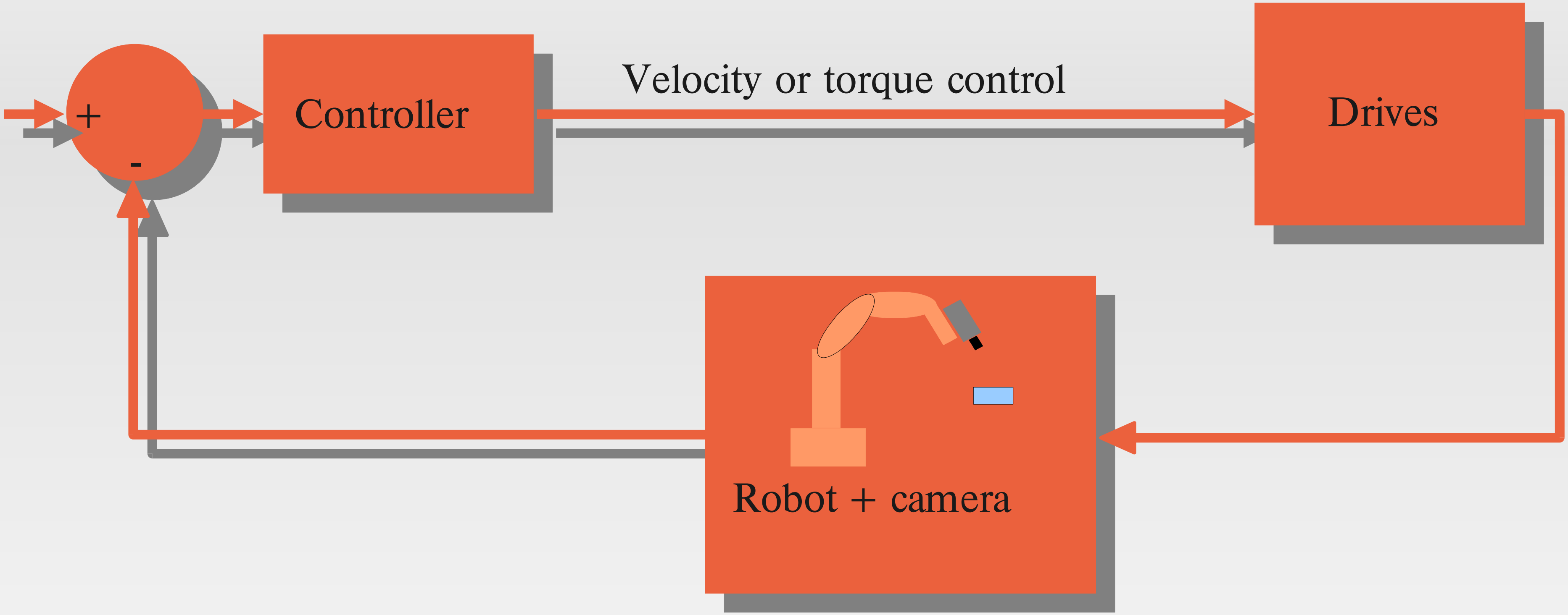
Eye to hand

1.3.2 Type of control signal



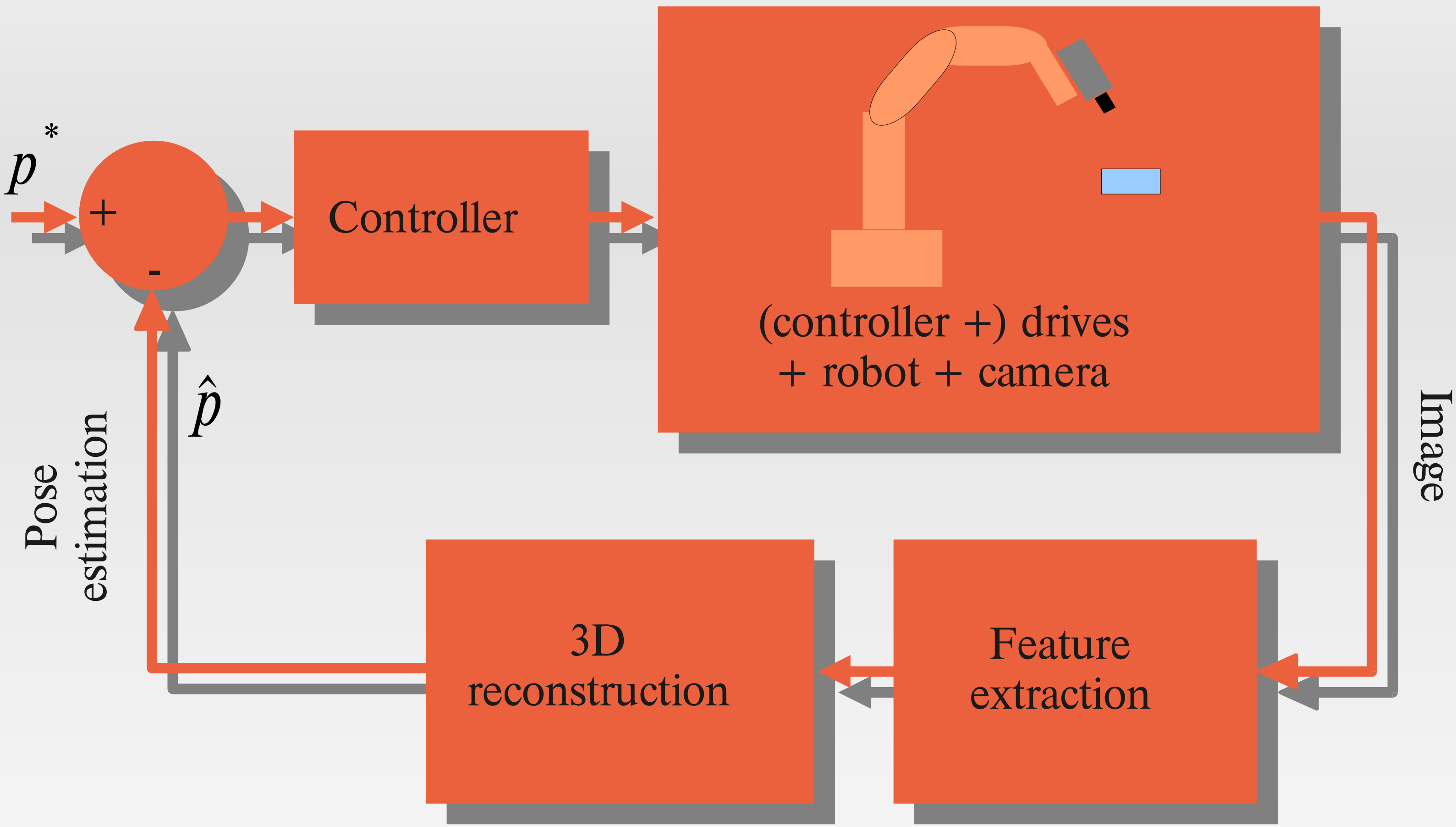
Indirect visual servoing : robot joint positions are controlled by the controller provided by the robot manufacturer. The vision loop controller provides velocity control signals which are converted in position setpoints by the robot controller. This type of visual servoing is used when the sampling period is slow thus corresponding to a slow response time.

1.3.2 Type of control signal



Direct visual servoing : robot joints torques or velocities are controlled. Torque or velocity control signals are issued by the controller of the visual loop. They are sent directly to the drives. This architecture is used for high bandwidth visual servoing with a high sampling rate.

1.3.3 Type of measurement



Position based (3D) visual servoing : reference and measurement are poses. To reconstruct the measurement it is necessary to have a model of the object of interest.

1.3.3 Type of measurement

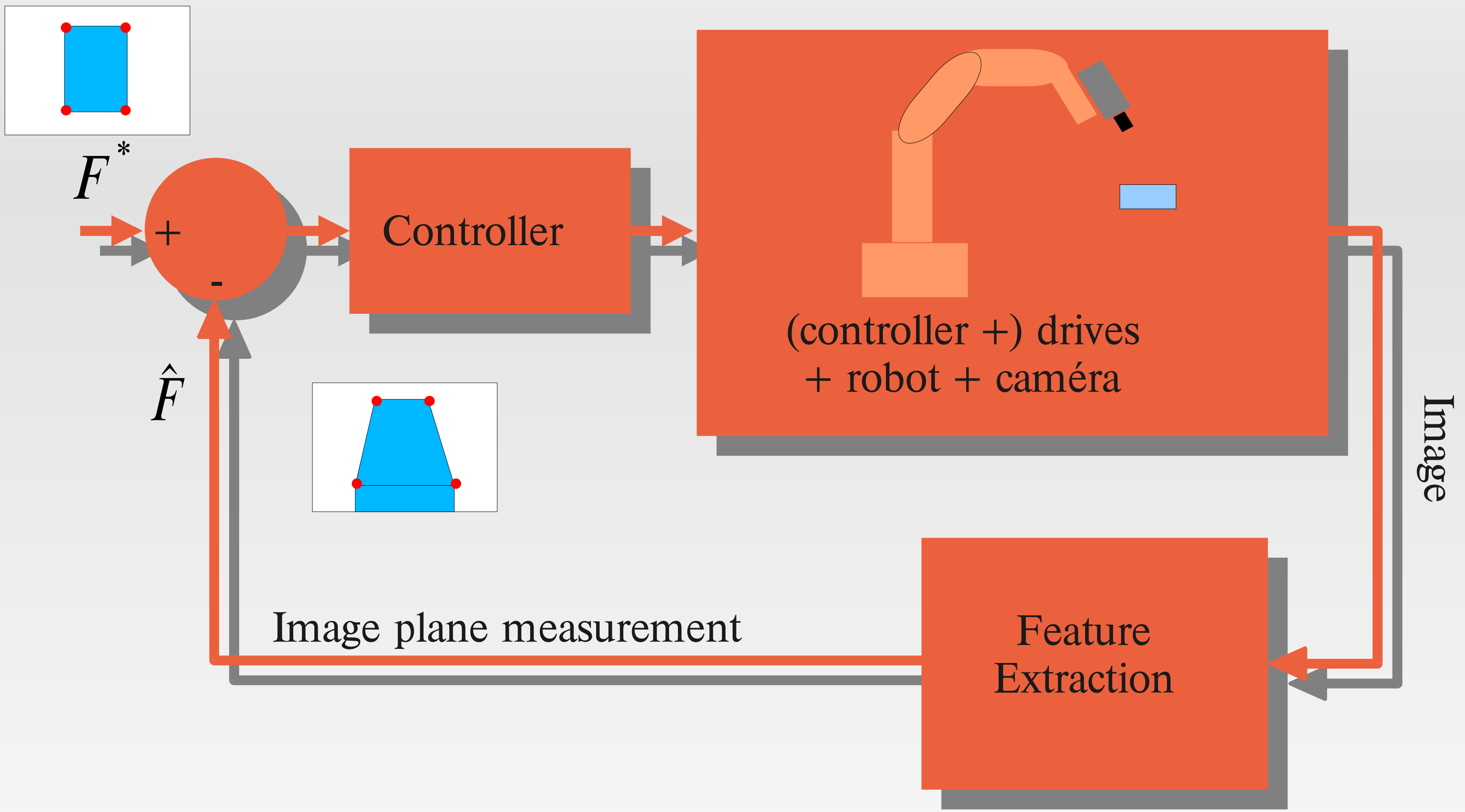
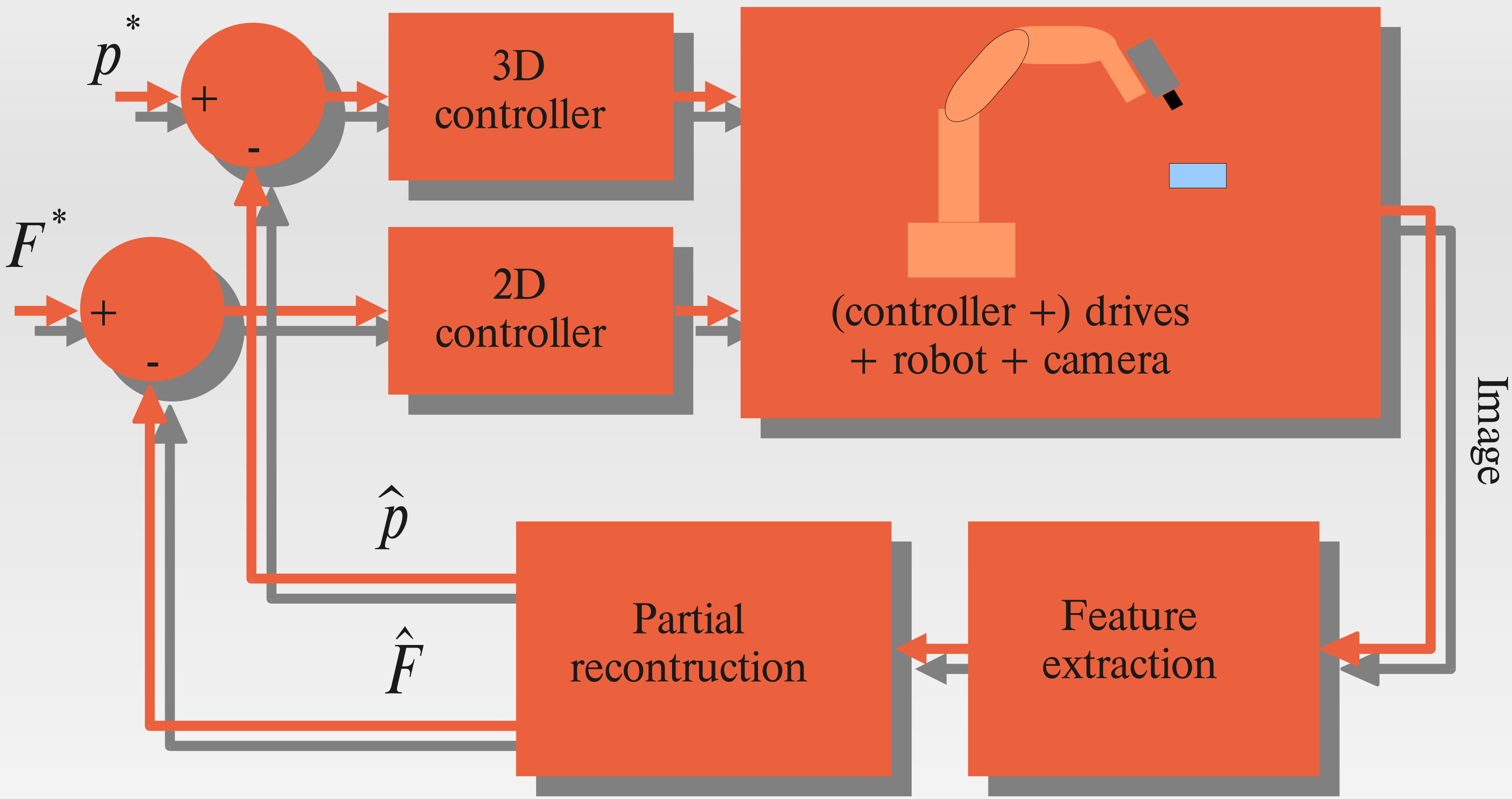


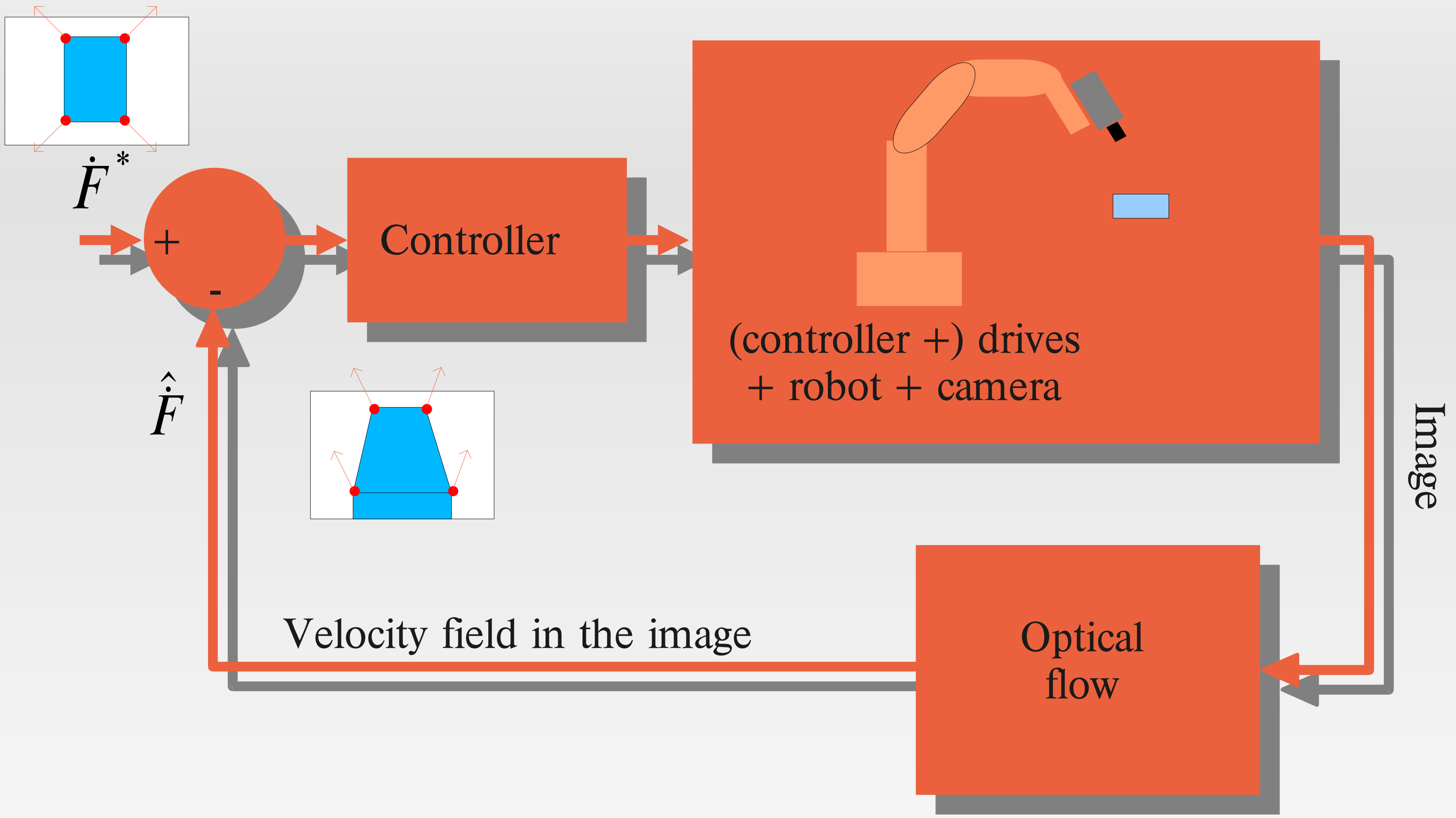
Image based (2D) visual servoing : reference and measurement are feature coordinates. No 3D reconstruction. Not necessary to have a model of the object of interest.

1.3.3 Type of measurement



Hybrid (2,5D) visual servoing : some DoFs are controlled in the 3D space and other are controlled in the image plane. The goal is to obtain the best decoupling between the 2 loops.

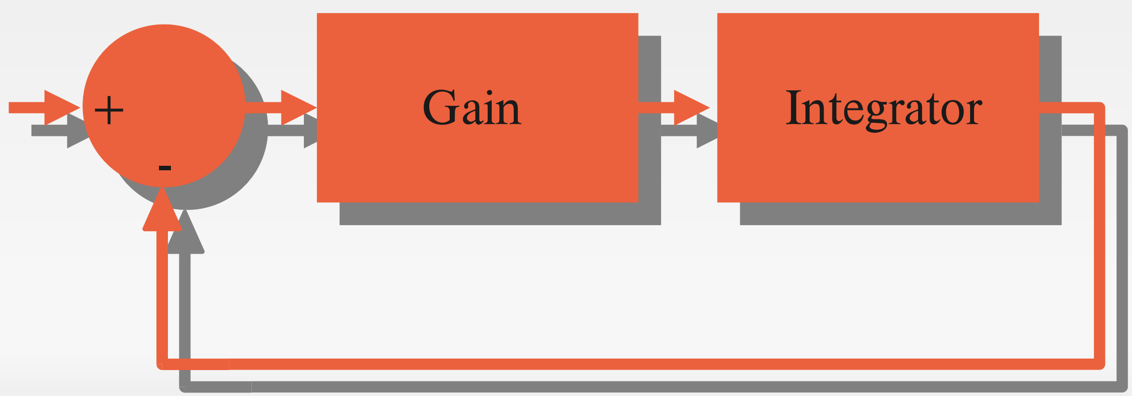
1.3.3 Type of measurement



d2D/dt visual servoing : the reference and the measurement are velocities in the image.

1.3.4 Bandwidth

- Kinematic visual servoing (slow)
  - Continuous time model
  - Robot = pure integrator
  - Controller is a gain
  - Dynamics of vision are neglected
  - Indirect control scheme



1.3.4 Bandwidth

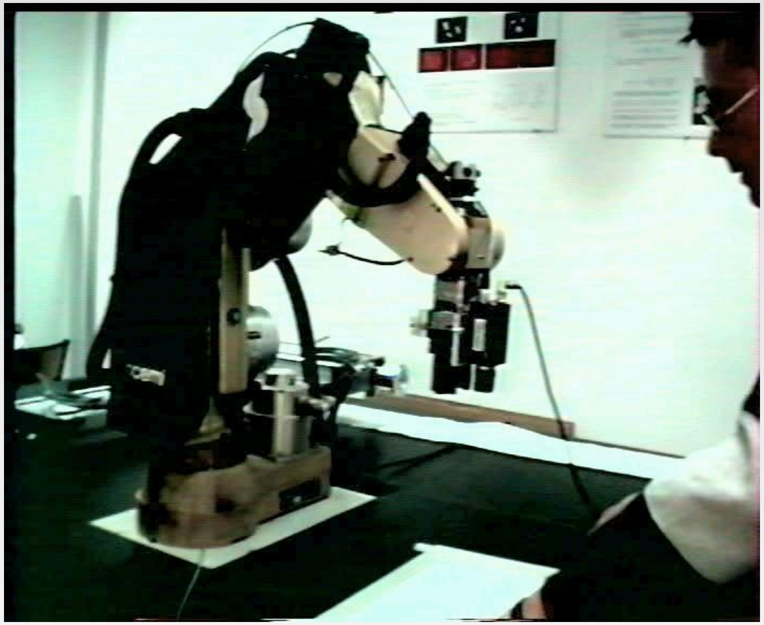
- Dynamic visual servoing (fast)
  - Discrete time model
  - Detailed electromecanical modeling of the robot
  - Detailed modeling of the camera (delays, sampling)
  - Advanced control (predictive, robust, ...)
  - Direct control scheme (velocity or torque)

1.4 Examples

- Tasks :
  - Tracking
  - Grasping
  - Insertion
  - Positioning

1.4.1 Tracking

- 6 DoF fast target tracking (Gangloff et al., Strasbourg University) :
  - Eye in hand, 2D, direct
  - 120 Hz
  - Predictive control
  - Velocity control signal

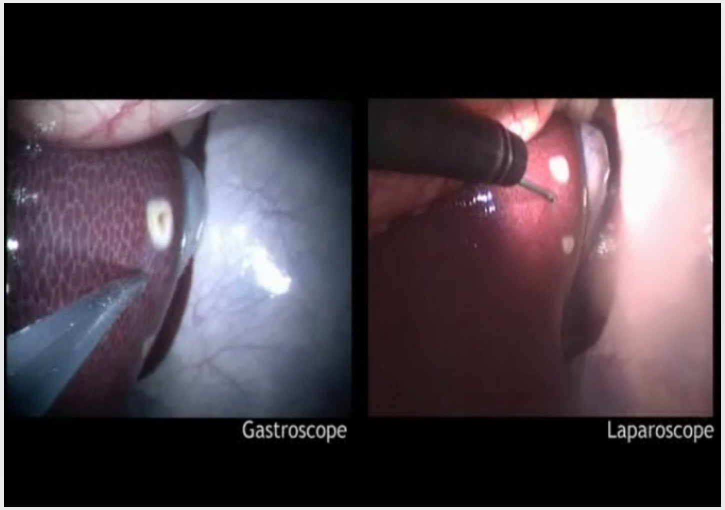


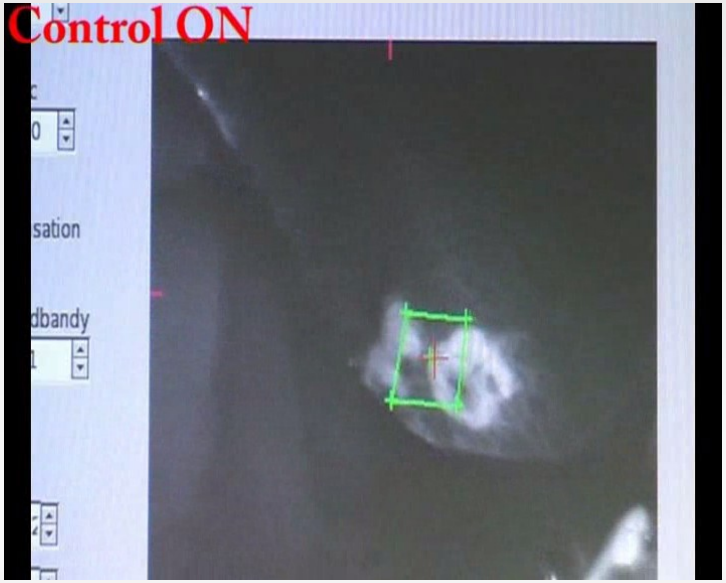
1.4.1 Tracking

- 2 DoF fast target tracking (Nakabo et al., Tokyo University) :
  - Eye in hand, 2D, direct
  - 1 kHz
  - Vision chip
  - Torque control signal

1.4.1 Tracking

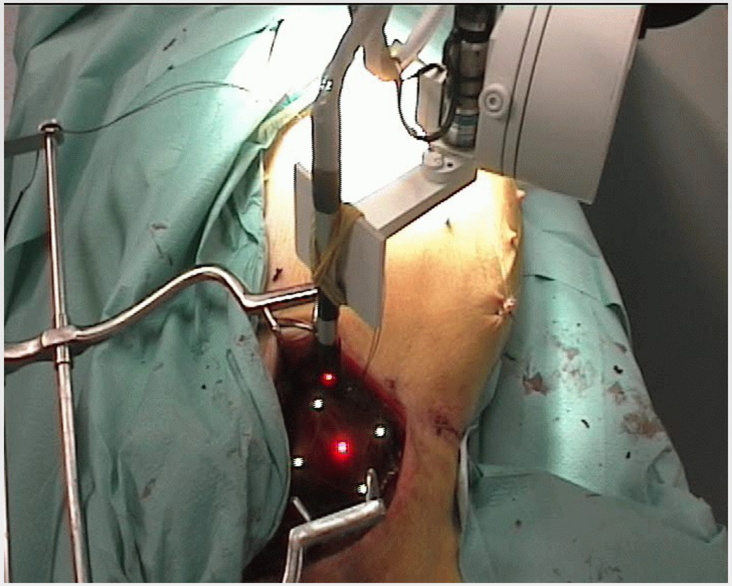
- 2 DoF tracking of liver (Ott et al., Strasbourg University):
  - Eye in hand, 2D, direct
  - 25 Hz
  - Repetitive control
  - Velocity control signal
  - The robot is an actuated flexible endoscope





1.4.1 Tracking

- 3 DoF tracking of beating heart (Ginhoux et al., Strasbourg University):
  - Eye to hand, 2D, direct
  - Predictive control
  - Velocity control signal
  - 500 Hz



1.4.2 Grasping

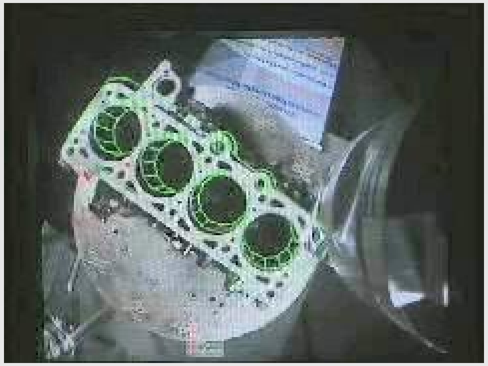
- 6 DoF grasping of a cube (Allen et al., Columbia University):
  - Eye in hand, 3D
  - Camera is at the wrist center

1.4.2 Grasping

- 5 DoF satellite grasping (DLR):
  - Eye in hand, 3D
  - Use of a special end-effector to capture the satellite using its nozzles
  - A second robot simulates micro-gravity
  - The insertion phase is guided using LASER rangefinders.

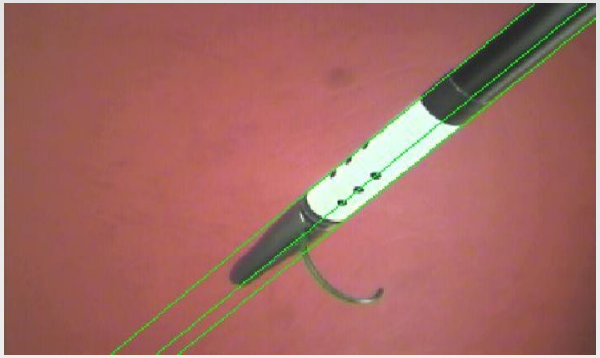
1.4.3 Insertion

- 5 DoF piston insertion (DLR):
  - Eye in hand, 3D
  - The engine block is put on a rotating table
  - The approach of the cylinder is controlled by vision
  - The insertion is force controlled



1.4.3 Insertion

- 4 DoF needle insertion (Nageotte et al., University of Strasbourg):
  - Eye to hand, 2D
  - The vision system is an endoscope

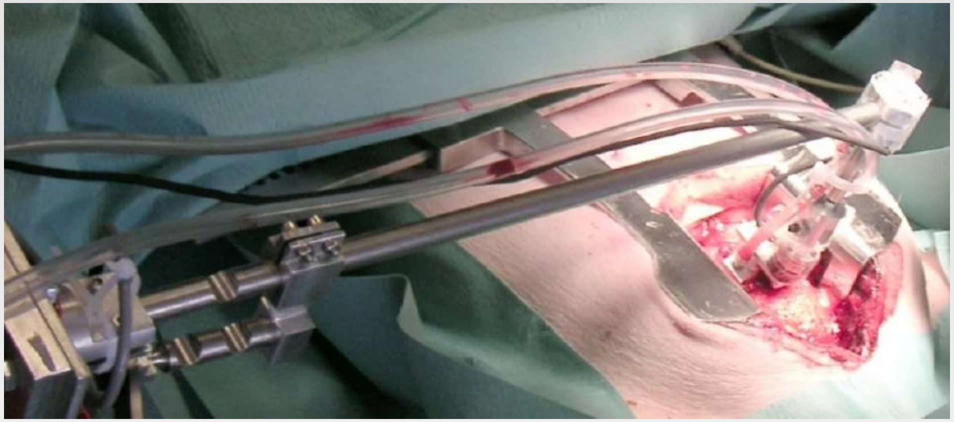


1.4.4 Positioning

- 5 DoF needle positioning (Maurin et al., Strasbourg University):
  - Eye to hand, 3D, indirect (look then move)
  - The vision system is a CT-scan
  - Sequence : acquisition – 3D reconstruction – motion

1.4.4 Positioning

- 1 DoF myocardium stabilization (Bachta et al., Strasbourg University):
  - Eye to hand, 2D, direct
  - 300 Hz
  - Piezoelectric actuation



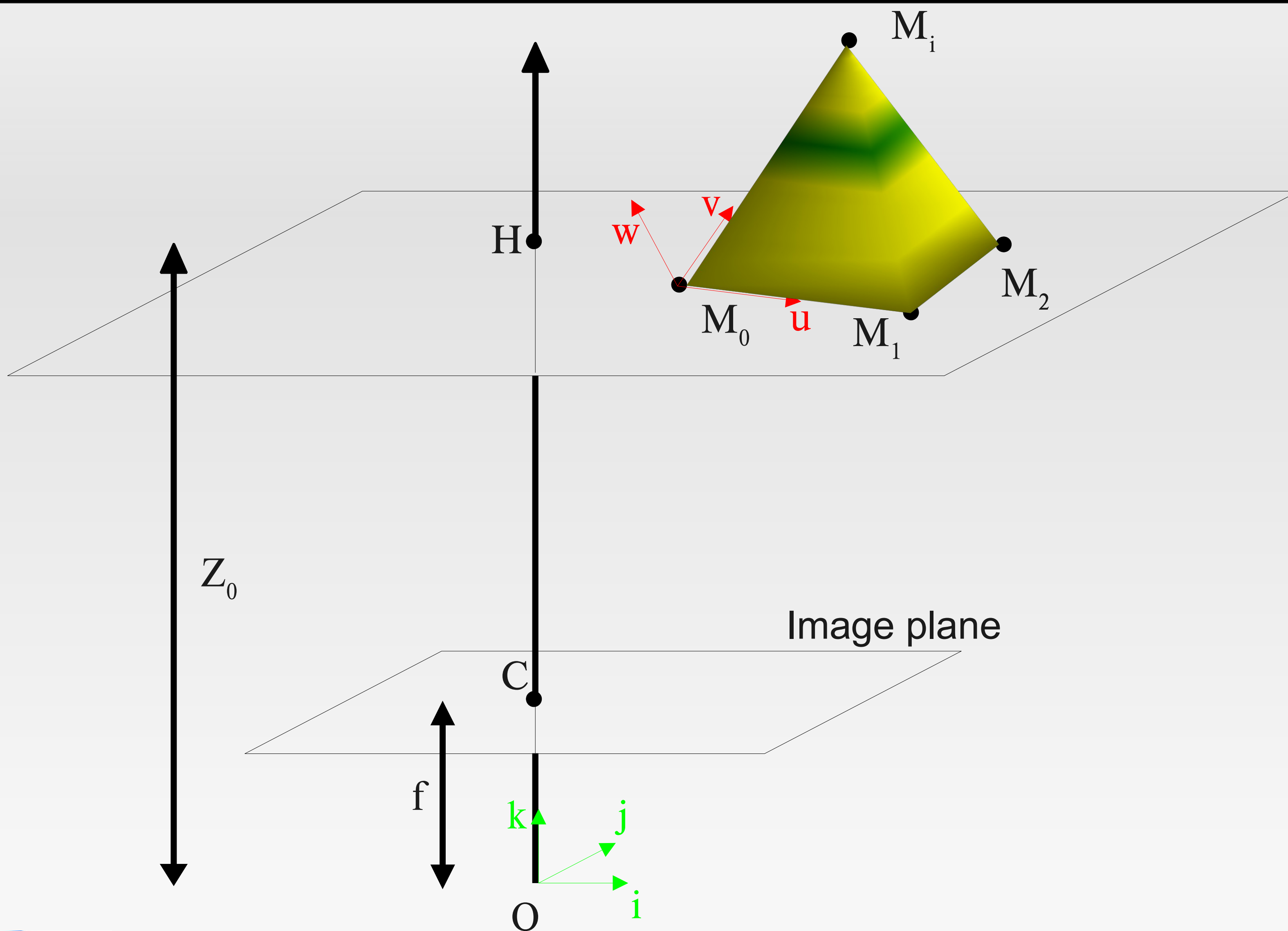
2. Measurement estimation

- Image acquisition
- Image processing (noise removal, filtering)
- Feature detection (detection of points, lines, ellipses, ...)
- For 3D visual servoing : 3D reconstruction
  - Dementhon method
  - Image Jacobian method

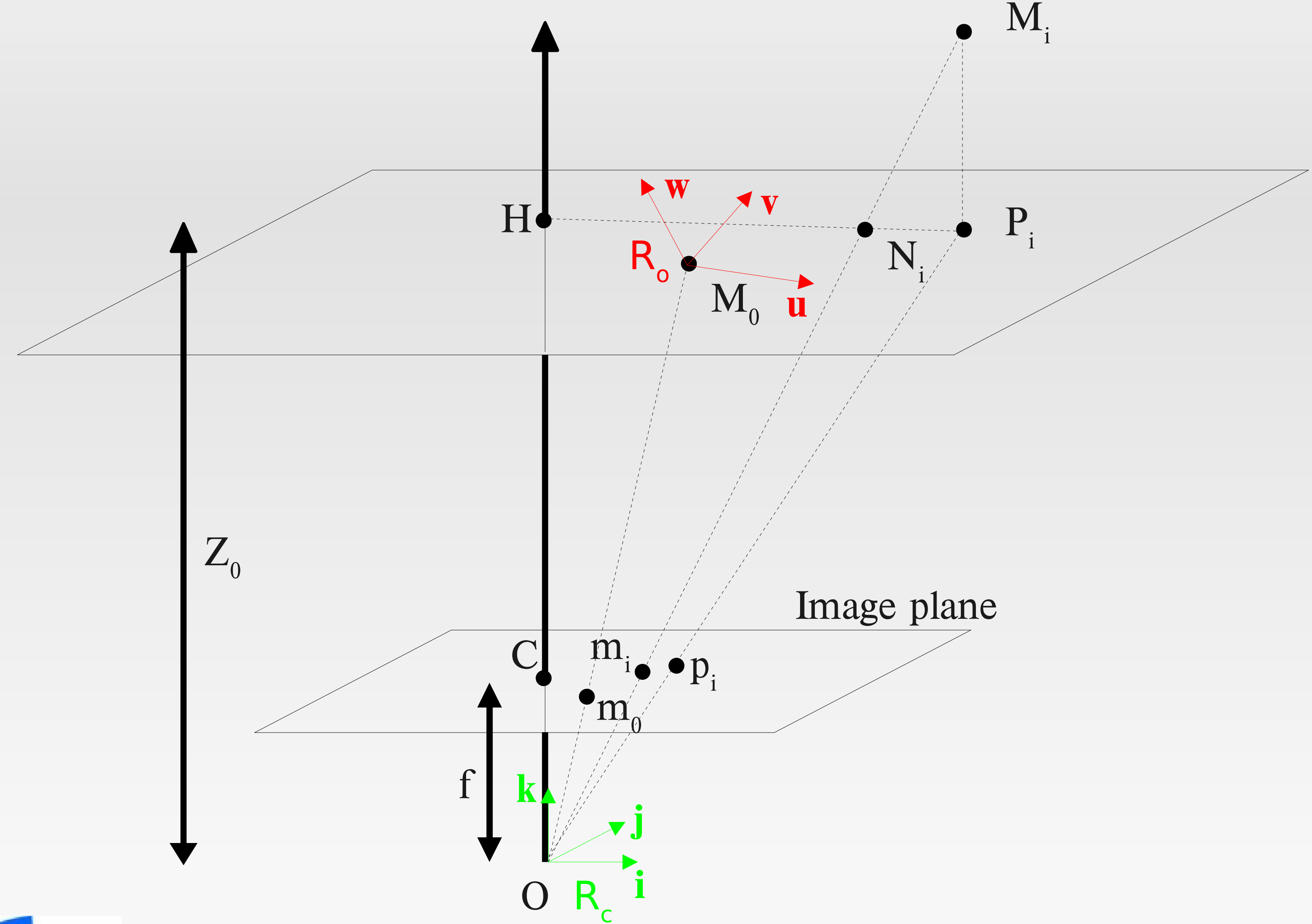
2.1 Dementhon method

- Computes the pose of an object defined by points (not necessarily coplanar)
- Minimum of 4 points
- Iterative method : non deterministic execution time
- References :
  - D. DeMenthon et L. S. Davis, Model-Based Object Pose in 25 Lines of Code, International Journal of Computer Vision, volume 15, pages 123 à 141, 1995.
  - http://www.cfar.umd.edu/~daniel/#pose

2.1.1 Definitions



2.1.1 Definitions



2.1.2 Objectives

Let M_{co}, the homogeneous transformation between the camera frame R$_c$ $(O, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$ and the frame linked to the object R$_o$ $(M_o, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$:

$$M_{co} = \begin{bmatrix} i_u & i_v & i_w & T_x \\ j_u & j_v & j_w & T_y \\ k_u & k_v & k_w & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Coordinates of $\boldsymbol{i}$ in R$_o$ denoted ${}^o\boldsymbol{i}$

Coordinates of $\boldsymbol{j}$ in R$_o$ denoted ${}^o\boldsymbol{j}$

${}^o\boldsymbol{i} \times {}^o\boldsymbol{j}$

$T_x = x_0 Z_0 / f$, $T_y = y_0 Z_0 / f$ et $T_z = Z_0$

To estimate M_{co} , it is sufficient to estimate the coordinates of $\boldsymbol{i}$ and $\boldsymbol{j}$ in R$_o$ and the coordinates of M_0 in R$_c$. The coordinates of M_0 in R$_c$, is obtained from the depth Z_0 : if x_0 and y_0 are the coordinates of m_0 in R$_c$ then $T_x = x_0 Z_0 / f$ and $T_y = y_0 Z_0 / f$

2.1.3 Orthographic projection

Let $m_i\ (x_i y_i f)_{Rc}$ the perspective projection of $M_i\ (X_i\ Y_i\ Z_i)_{Rc}$ in the image plane and $p_i\ (x'_i y'_i f)_{Rc}$ its othographic projection.
Let :

$$x'_i = f\frac{X_i}{Z_0} \qquad y'_i = f\frac{Y_i}{Z_0}$$

And :

$$x_i = f\frac{X_i}{Z_i} \qquad y_i = f\frac{Y_i}{Z_i}$$

Let $s = f/Z_0$ be the magnification factor of the orthographic projection.
Let :

$$x'_i = f\frac{X_0}{Z_0} + f\frac{(X_i - X_0)}{Z_0} = x_0 + s(X_i - X_0)$$

$$y'_i = y_0 + s(Y_i - Y_0)$$

2.1.4 Fundamental relationship

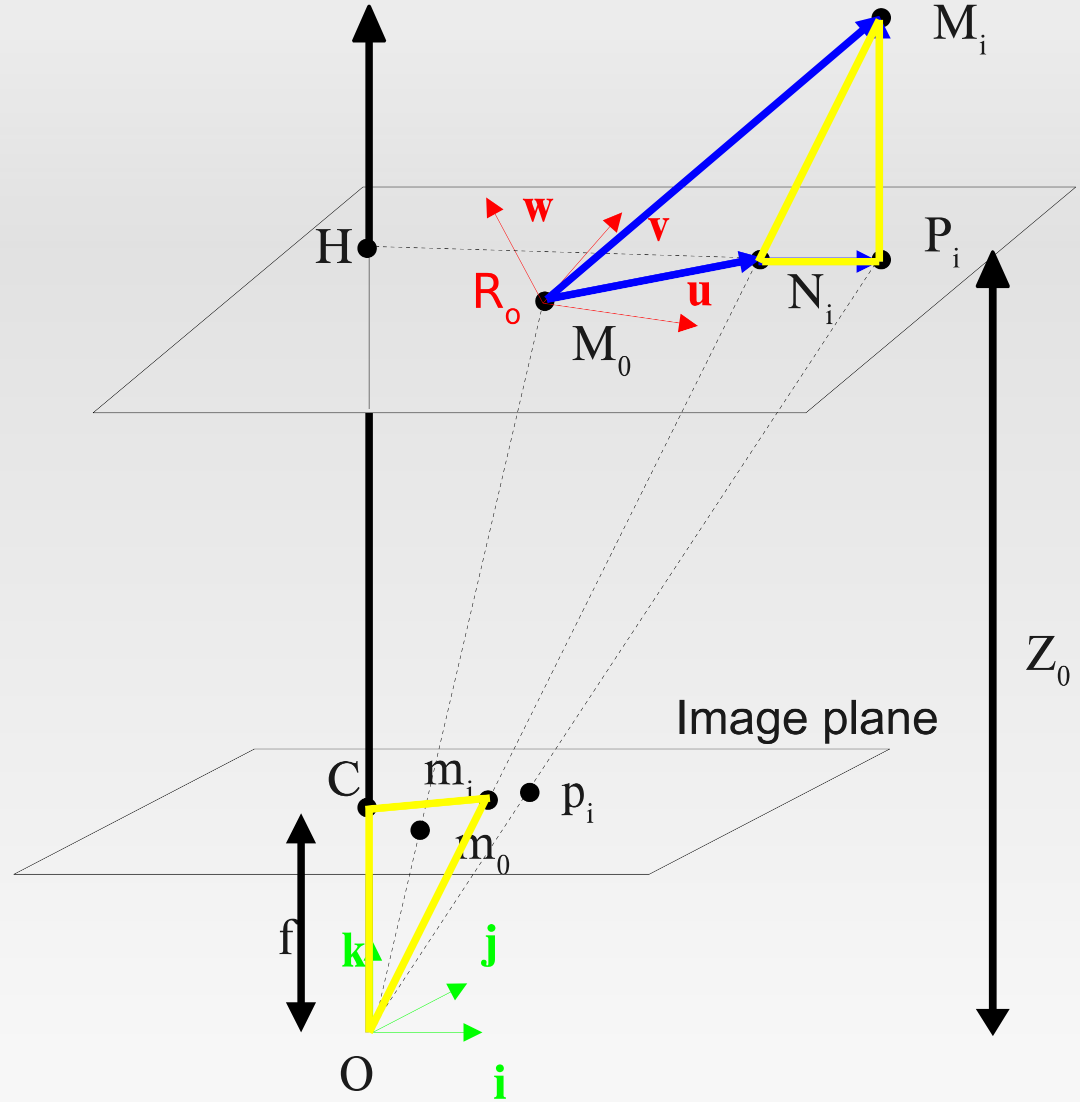
Equations :

$$\boldsymbol{M_0 M_i}\cdot\frac{f}{Z_0}\boldsymbol{i}=x_i(1+\varepsilon_i)-x_0 \qquad (1)$$

$$\boldsymbol{M_0 M_i}\cdot\frac{f}{Z_0}\boldsymbol{j}=y_i(1+\varepsilon_i)-y_0 \qquad (2)$$

with : $\varepsilon_i=\frac{1}{Z_0}\boldsymbol{M_0 M_i}\cdot\boldsymbol{k}$

Demonstration : see paper.



2.1.4 Fundamental relationship

Equations (1) et (2) can be put in the compact form :

$$\boldsymbol{M}_0\boldsymbol{M}_i \cdot \boldsymbol{I} = \xi_i$$
$$\boldsymbol{M}_0\boldsymbol{M}_i \cdot \boldsymbol{J} = \eta_i$$

With :

$$\boldsymbol{I} = \frac{f}{Z_0}\boldsymbol{i} \qquad \boldsymbol{J} = \frac{f}{Z_0}\boldsymbol{j} \qquad \xi_i = x_i(1+\varepsilon_i) - x_0 \qquad \eta_i = y_i(1+\varepsilon_i) - y_0$$

Let $[U_i V_i W_i]$, ${}^o\boldsymbol{I} = [I_u I_v I_w]$ and ${}^o\boldsymbol{J} = [J_u J_v J_w]$ respectively the coordinates of $\boldsymbol{M}_o\boldsymbol{M}_i$, $\boldsymbol{I}$ and $\boldsymbol{J}$ in frame R_o.

Let A the matrix of the coordinates $[U_i\ V_i\ W_i]$ of points M_i , $\boldsymbol{x}$ ‘, is the vector of the ξ_i and $\boldsymbol{y}$ ‘ is the vector of the η_i . It yields the following equations :

$$A\,{}^o\boldsymbol{I} = \boldsymbol{x}' \qquad A\,{}^o\boldsymbol{J} = \boldsymbol{y}' \tag{3}$$

UNIVERSITÉ DE STRASBOURG

2.1.5 Resolution

With at least 3 points M_i other than M_0 , the rank of matrix A is 3. So $^o\boldsymbol{I}$ and $^o\boldsymbol{J}$ are obtained from the inverse or the pseudo-inverse B of A :

$$^o\boldsymbol{I}=B\,\boldsymbol{x}' \qquad ^o\boldsymbol{J}=B\,\boldsymbol{y}' \qquad \text{avec } B=[A^T A]^{-1}A^T$$

This yields :

$$^o\boldsymbol{i}=\frac{^o\boldsymbol{I}}{\|I\|} \qquad ^o\boldsymbol{j}=\frac{^o\boldsymbol{J}}{\|J\|} \qquad ^o\boldsymbol{k}={}^o\boldsymbol{i}\times{}^o\boldsymbol{j}$$

The depth Z_0 is derived from the norm of $\boldsymbol{I}$ or $\boldsymbol{J}$. It yields :

$$T_x=x_0\frac{Z_0}{f} \qquad T_y=y_0\frac{Z_0}{f} \qquad T_z=Z_0$$

2.1.6 Algorithm

Equations (1) and (2) put into matrix form (3) can be solved by fixing a value for ε_i. The initial value of ε_i in the algorithm is 0, then, at each new iteration ε_i is reevaluated thanks to the following equation :

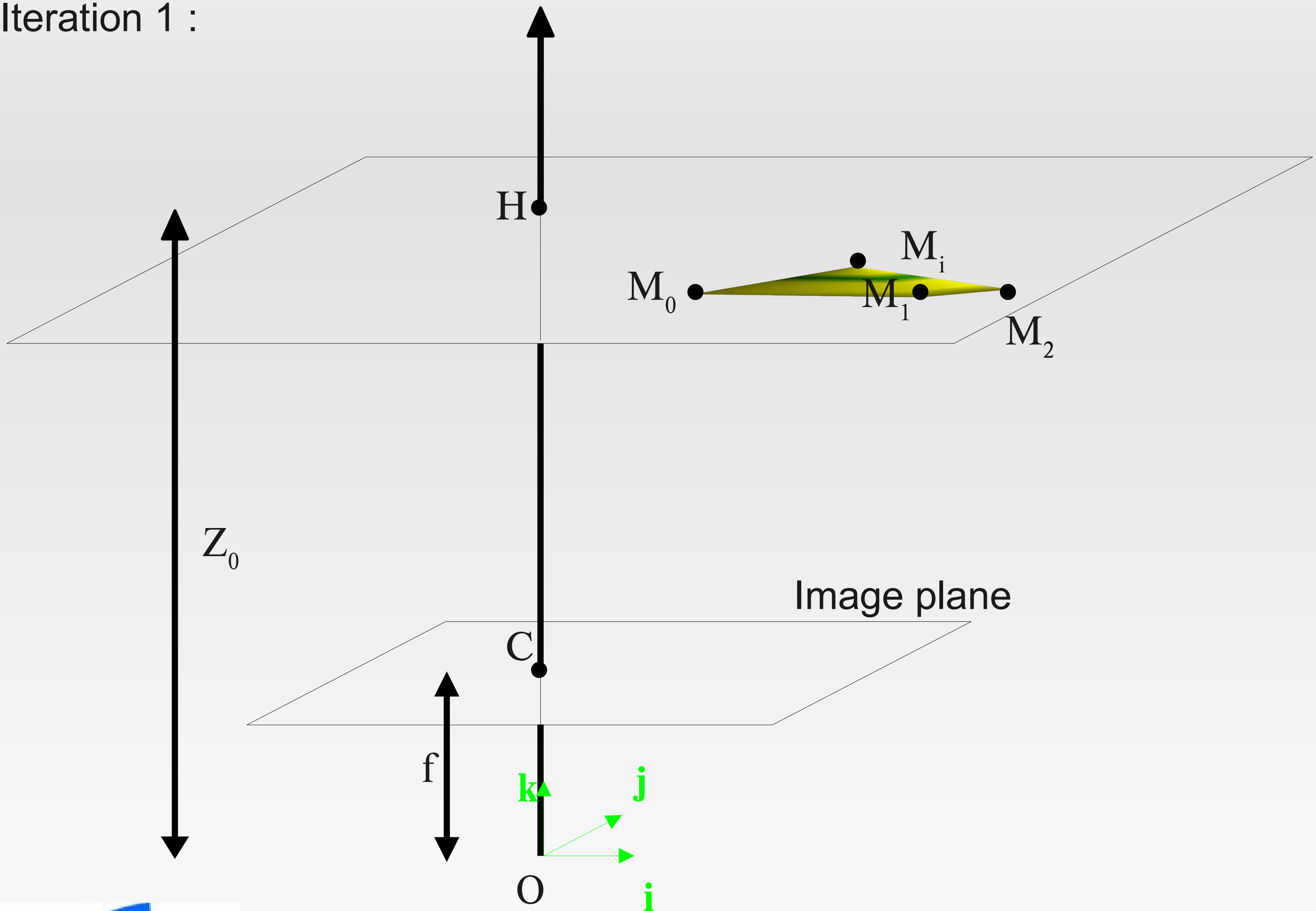
$$\varepsilon_i = \frac{1}{Z_0} \boldsymbol{M}_0 \boldsymbol{M}_i \cdot \boldsymbol{k}$$

The iterations stop when the difference between two successive values of ε_i falls below a defined threshold. Usually, this falls needs less than 10 iterations to converge.

The assumption made by arbitrary fixing $\varepsilon_i = 0$ is equivalent to the assumption of a planar object parallel to the image plane. Of course, in the general case, this is false but not false enough to make the algorithm diverge.

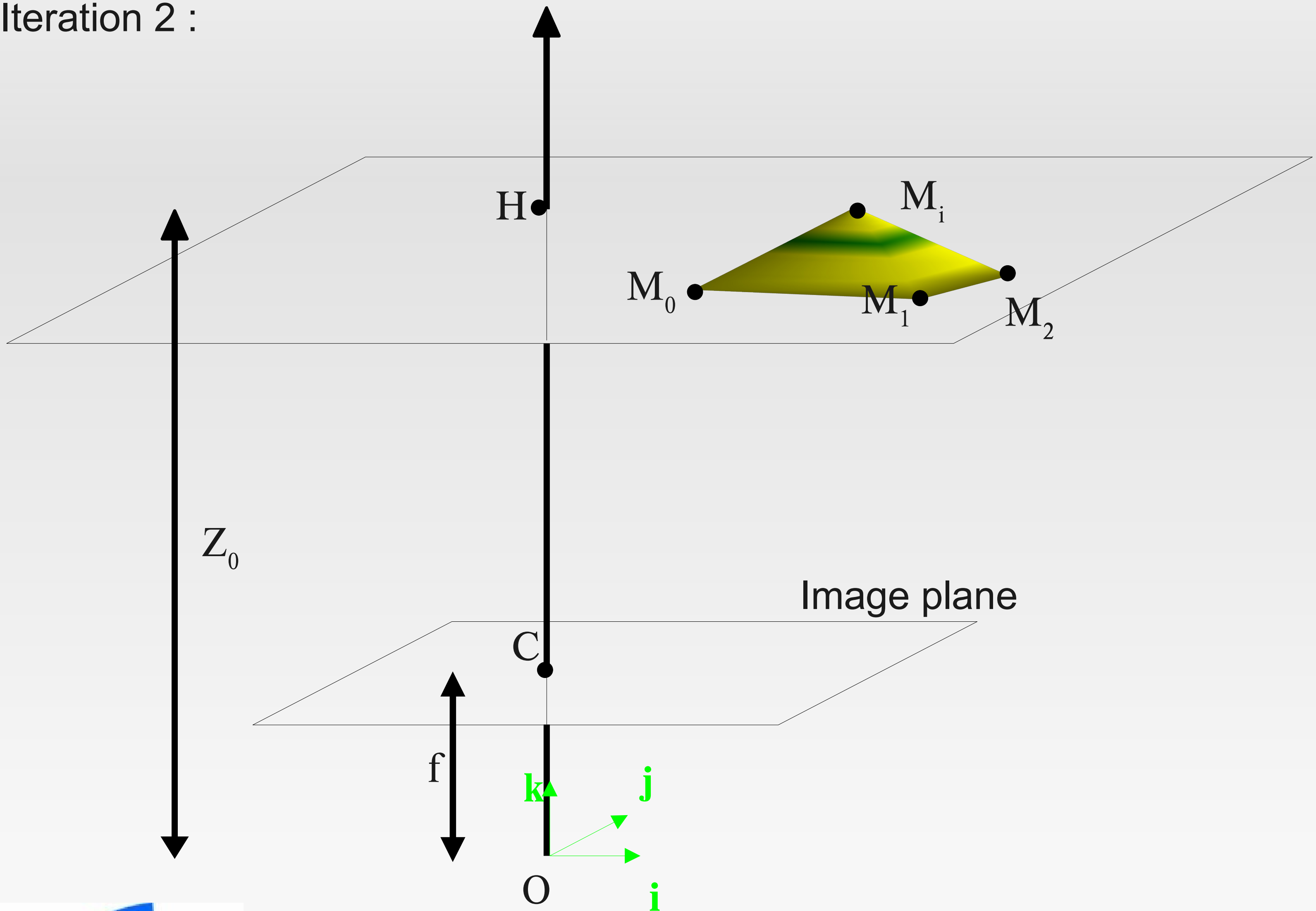
2.1.6 Algorithm

Iteration 1 :



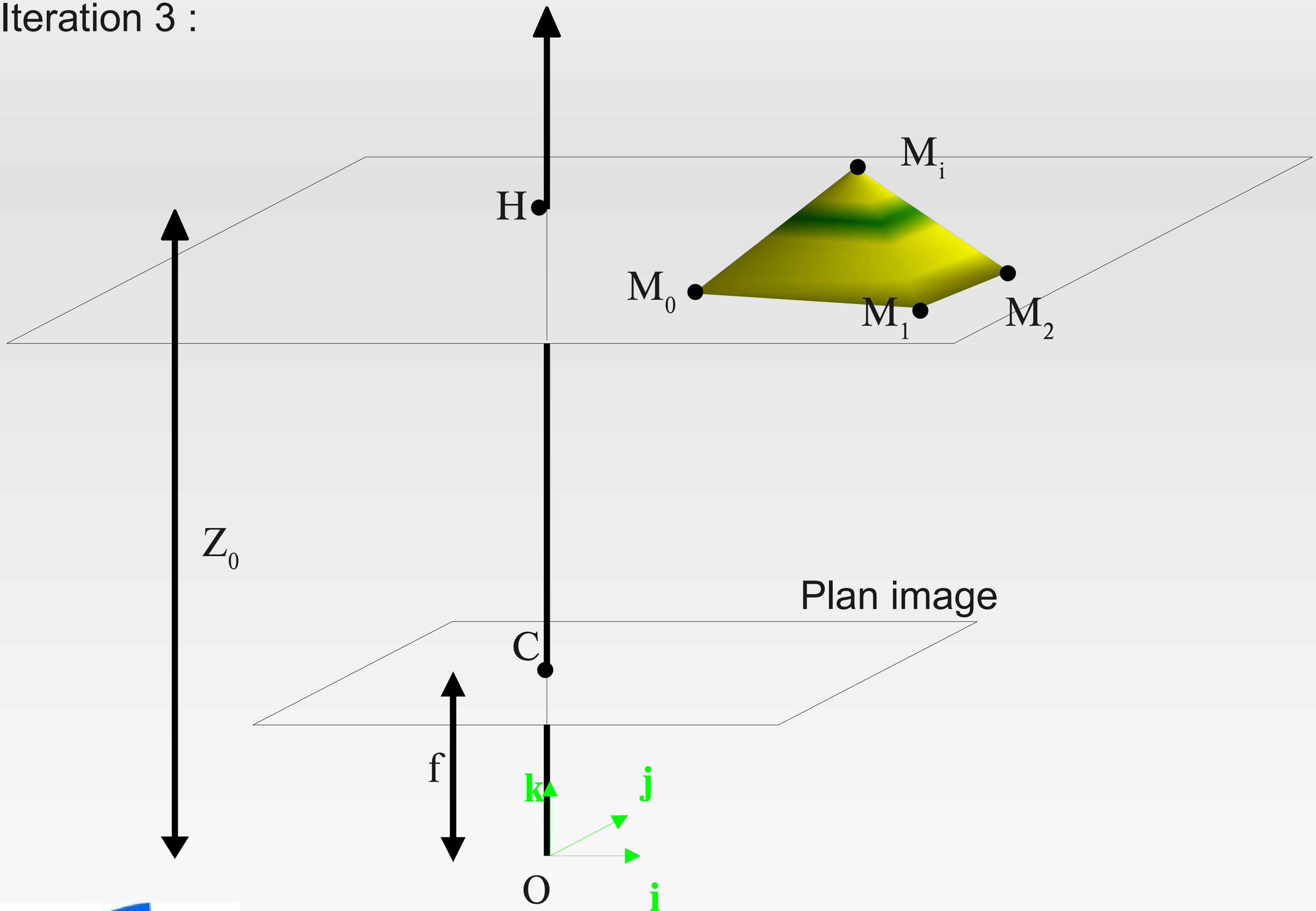
2.1.6 Algorithm

Iteration 2 :



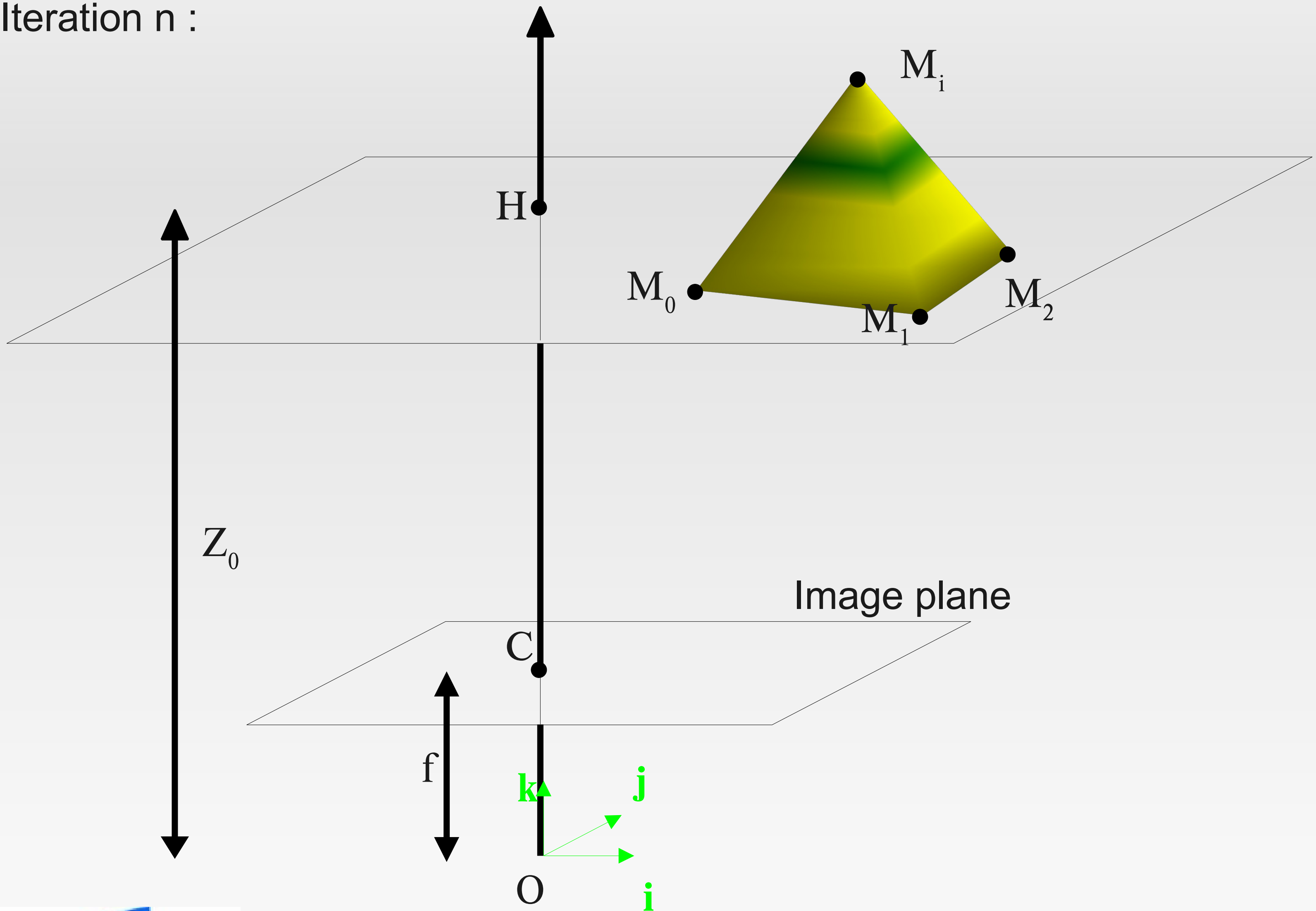
2.1.6 Algorithm

Iteration 3 :



2.1.6 Algorithm

Iteration n :



2.1.6 Algorithm

Initialization : $n=0$, N = number of points. Initialization of matrix A $(N-1)$x3
Compute B 3x$(N-1)$, pseudo-inverse of A. $\varepsilon_{i(0)} = 0$, $i = 1...N-1$

Compute $^o\boldsymbol{i}$, $^o\boldsymbol{j}$, Z_0 : Compute $\boldsymbol{x}'$ et $\boldsymbol{y}'$ of dimension $N-1$ which coordinates are : $x_i(1+\varepsilon_{i(n-1)})-x_0$ et $y_i(1+\varepsilon_{i(n-1)})-y_0$

Compute $^o\boldsymbol{I}$ and $^o\boldsymbol{J}$: $^o\boldsymbol{I}=B\,\boldsymbol{x}'$ $\quad {}^o\boldsymbol{J}=B\,\boldsymbol{y}'$

Compute $^o\boldsymbol{i}$, $^o\boldsymbol{j}$ and Z_0 :

$$s_1=(\boldsymbol{I}\cdot\boldsymbol{I})^{1/2} \quad s_2=(\boldsymbol{J}\cdot\boldsymbol{J})^{1/2} \quad {}^o\boldsymbol{i}=\frac{{}^o\boldsymbol{I}}{s_1} \quad {}^o\boldsymbol{j}=\frac{{}^o\boldsymbol{J}}{s_2} \quad Z_0=\frac{2f}{s_1+s_2}$$

Compute $\varepsilon_{i(n)}$: $^o\boldsymbol{k}={}^o\boldsymbol{i}\times{}^o\boldsymbol{j} \quad \varepsilon_{i(n)}=\frac{1}{Z_0}\boldsymbol{M}_0\boldsymbol{M}_i\cdot\boldsymbol{k} \quad i=1...N-1$

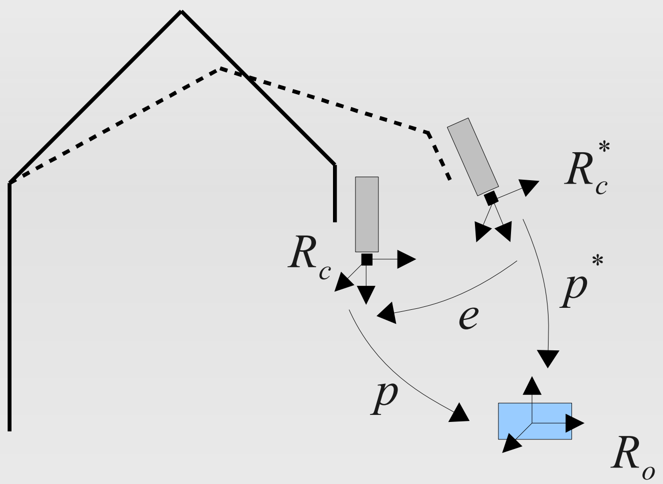
$|\varepsilon_{i(n)}-\varepsilon_{i(n-1)}|>\text{thrd}$

yes: $n=n+1$

no: Construction of M_{co} from the last values of $^o\boldsymbol{i}$, $^o\boldsymbol{j}$ and Z_0

2.2 Image Jacobian method

- p^* : desired pose
- p : current pose
- Find $e = p - p^*$
- Task : target tracking
- Maintain e as small as possible



UNIVERSITÉ DE STRASBOURG

2.2.1 Formulation

Let : $\dot{F} = L_F T_c$ with :

$\dot{F}$ the velocities of the points in the image,

L_F the image Jacobian,

T_c the velocity skrew of the camera frame.

$\dot{e} = L_e T_c$ with :

$$L_e = \begin{bmatrix} R_{c*c} & 0 \\ 0 & L_{\vartheta u} \end{bmatrix} \qquad e = [t_{c*c} \quad \vartheta u]^T$$

t_{c*c} and R_{c*c} : resp. translation vector and rotation matrix between R_c^* and R_c

u is the unit vector defining the rotation axis and ϑ the rotation angle

$$L_{\vartheta u} = I_3 - \frac{\vartheta}{2}[u]_\times + \left(1 - \frac{\operatorname{sinc} \vartheta}{\operatorname{sinc}^2 \frac{\vartheta}{2}}\right)[u]_\times^2 \qquad [u]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix}$$

2.2.2 First order approximation

Since $\dot{F}=L_F T_c$ and $\dot{e}=L_e T_c$

it yields : $\dot{e}=L_e L_F^+ \dot{F}$ with $L_F^+=(L_F^T L_F)^{-1} L_F^T$

First order approximation : $e \approx L_e L_F^+ (F-F^*)$ for $F-F^*$ and e small

with : F and F^* resp. the current and desired feature vector

Considering that $\underset{e=0}{L_e}=I$ it yields :

$$\boxed{e \approx L_F^+ (F-F^*)}$$

- Suited for tracking since *e* is supposed to be small
- L_F can be computed beforehand
- Very efficient algorithm

2.2.3 Case of point features

The scene is defined by n points :

$F=[x_1\, y_1\, x_2\, y_2 \ldots x_n\, y_n]^T$ with $(x_i\, y_i)$ the coordinates in the image plane of point i

In this case :

$$L_F=\begin{bmatrix} \frac{-G_x}{Z_1} & 0 & \frac{x_1}{Z_1} & \frac{x_1 y_1}{G_y} & -\frac{G_x^2+x_1^2}{G_x} & y_1\frac{G_x}{G_y} \\ 0 & \frac{-G_y}{Z_1} & \frac{y_1}{Z_1} & \frac{G_y^2+y_1^2}{G_y} & \frac{-x_1 y_1}{G_x} & -x_1\frac{G_y}{G_x} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{-G_x}{Z_n} & 0 & \frac{x_n}{Z_n} & \frac{x_n y_n}{G_y} & -\frac{G_x^2+x_n^2}{G_x} & y_n\frac{G_x}{G_y} \\ 0 & \frac{-G_y}{Z_n} & \frac{y_n}{Z_n} & \frac{G_y^2+y_n^2}{G_y} & \frac{-x_n y_n}{G_x} & -x_n\frac{G_y}{G_x} \end{bmatrix}$$

with G_x and G_y the lens magnification along x and y (pixel unit)

Z_i is the depth of point i in 3D space

UNIVERSITÉ DE STRASBOURG

2.2.4 Example

$M_0 \ (0 \ \ 0 \ \ 0) \quad M_1 \ (0.1 \ \ 0 \ \ 0) \quad M_2 \ (0 \ \ 0.1 \ \ 0) \quad M_3 \ (0 \ \ 0 \ \ 0.1)$

$$M_{c*o} = \begin{bmatrix} 1 & 0 & 0 & 0.15 \\ 0 & 0 & -1 & 0.1 \\ 0 & 1 & 0 & 0.3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

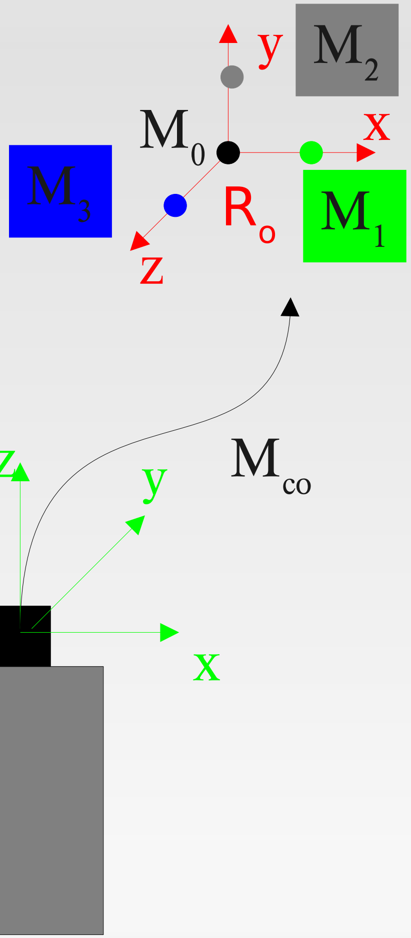
$F_d = [500 \ \ 333.33 \ \ 833.33 \ \ 333.33 \ \ 375 \ \ 250 \ \ 500 \ \ 0]^T$

$t_{c*c} = [0.1 \ \ 0 \ \ 0]^T \qquad \vartheta = 0.1 \text{ rad} \qquad u = [1 \ \ 0 \ \ 0]^T$

$F = [173.3 \ \ 448.7 \ \ 519.9 \ \ 448.7 \ \ 128.9 \ \ 359.3 \ \ 167.5 \ \ 100.3 \ ]^T$

$$L_F(F - F_d) = \begin{bmatrix} 0.0990 \\ 0.0021 \\ 0.0016 \\ 0.1016 \begin{pmatrix} 0.9861 \\ 0.0594 \\ -0.1549 \end{pmatrix} \end{bmatrix}$$

$G_x = G_y = 1000$

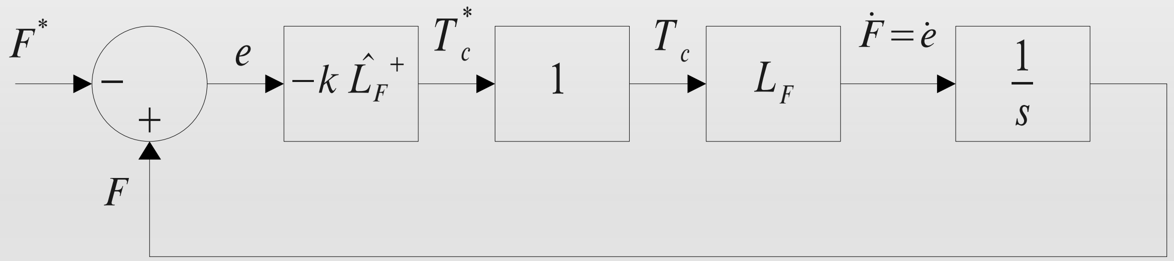


Matlab script : jacod3D.m

3. Control

- Kinematic control
  - 2D (IBVS)
  - 3D (PBVS)
- Dynamic control

3.1.1 2D kinematic vis. servoing



F^* : desired image $\quad$ F : measured current image

Control law : $T_c^* = -k\,\hat{L}_F^+ e$ with $\hat{L}_F^+$ estimated pseudo-inverse of image Jacobian

$T_c = T_c^*$: the robot is supposed to be perfect

$\dot{e} = \frac{d}{dt}(F - F^*) = \dot{F}$ with F^* supposed to be constant

According to the definition of the image Jacobian : $\dot{F} = \dot{e} = L_F T_c$

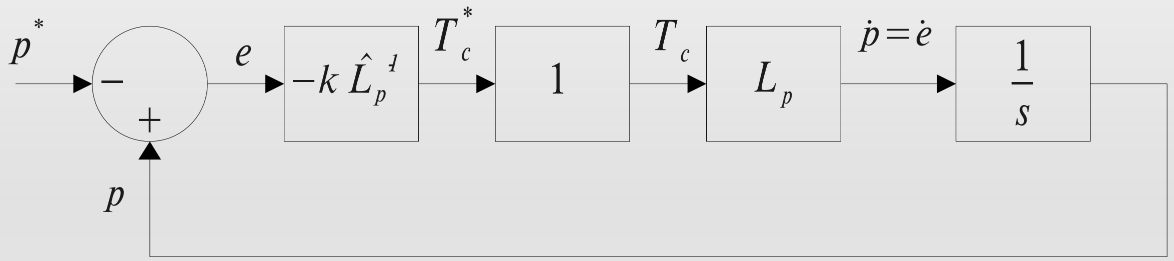
Since $T_c = T_c^* = -k\,\hat{L}_F^+ e$ it yields : $\dot{e} = -kL_F \hat{L}_F^+ e$

For 3 points and if we suppose $\hat{L}_F = L_F$ the solution is :

$e(t) = e(0)\exp(-kt)$ where $e(0)$ is the error at $t=0$

See simulink simulation 2Dservo.mdl

3.1.2 3D kinematic vis. servoing



p^* : desired pose $\quad$ p : reconstructed current pose

Control law : $T_c^* = -k\,\hat{L}_p^{-1} e$ with $\hat{L}_p^{-1}$ estimated inverse of L_p

$T_c = T_c^*$: the robot is supposed to be perfect

$\dot{e} = \dfrac{d}{dt}(p - p^*) = \dot{p}$ with p^* supposed to be constant

According to the definition of L_p : $\dot{p} = \dot{e} = L_p T_c$

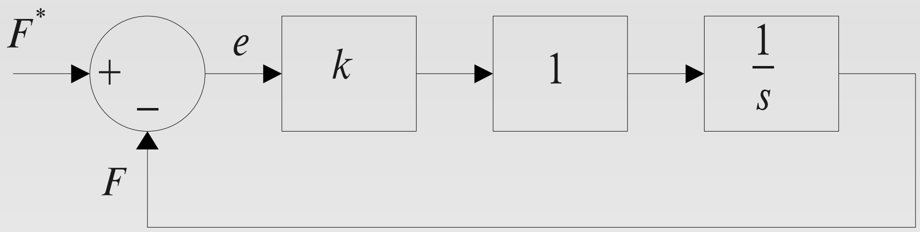
Since $T_c = T_c^* = -k\,\hat{L}_p^{-1} e$ it yields : $\dot{e} = -k L_p \hat{L}_F^{-1} e$

If we suppose $\hat{L}_p = L_p$ the solution is :

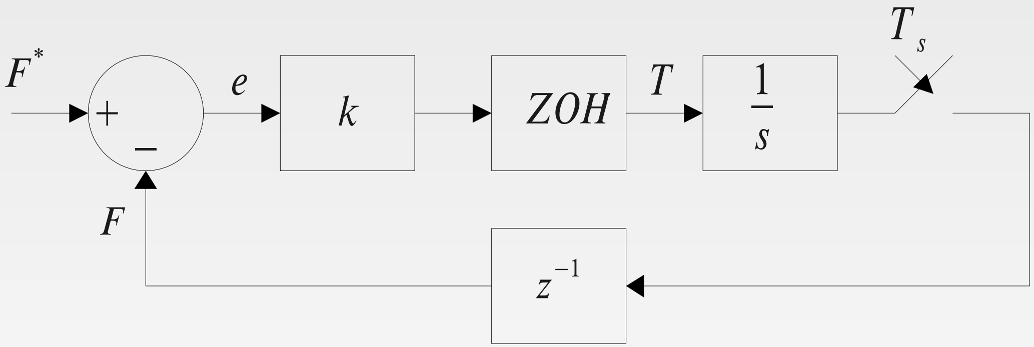
$e(t) = e(0)\exp(-kt)$ where $e(0)$ is the error at $t = 0$

3.1.3 Limitations

- Simplified continuous-time kinematic 2D VS :

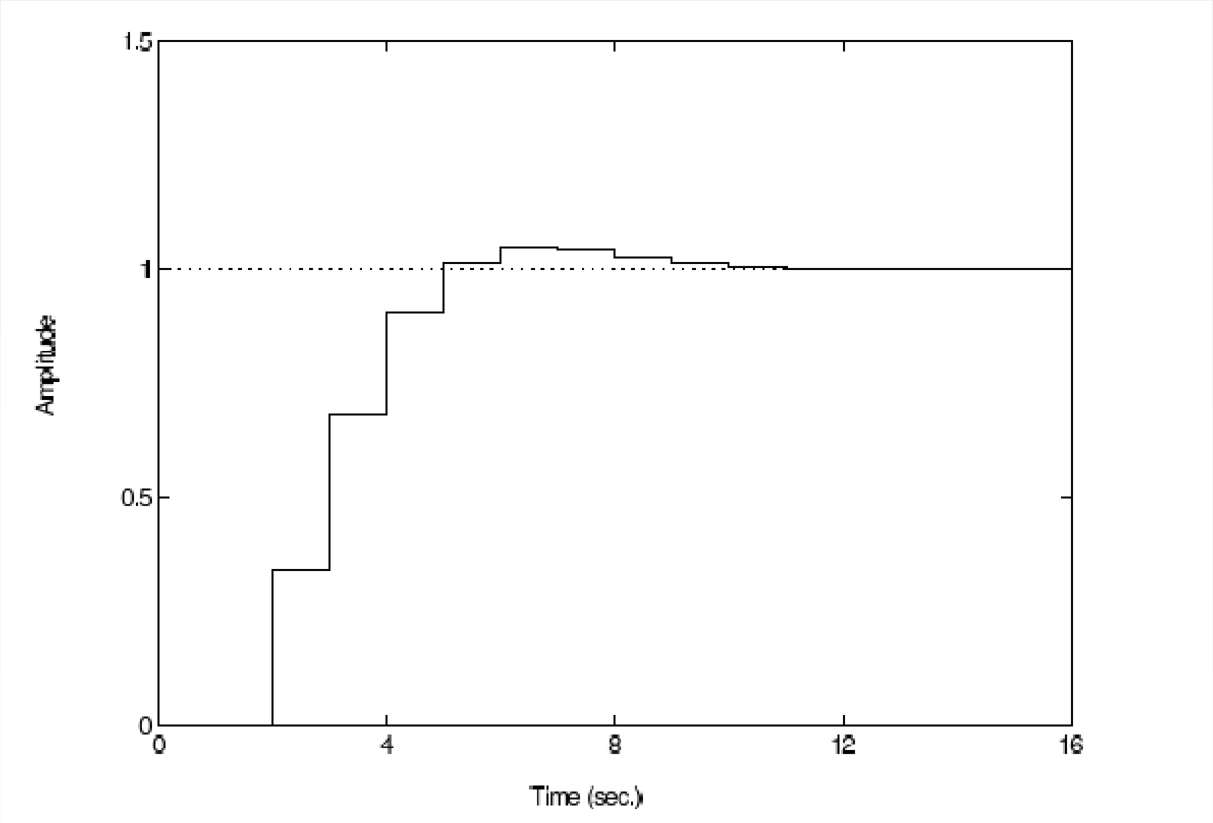


- Simplified discrete-time kinematic 2D VS :



3.1.3 Limitations

- Response of the discrete-time 2D VS with k = 0.34 :



3.2 Dynamic control

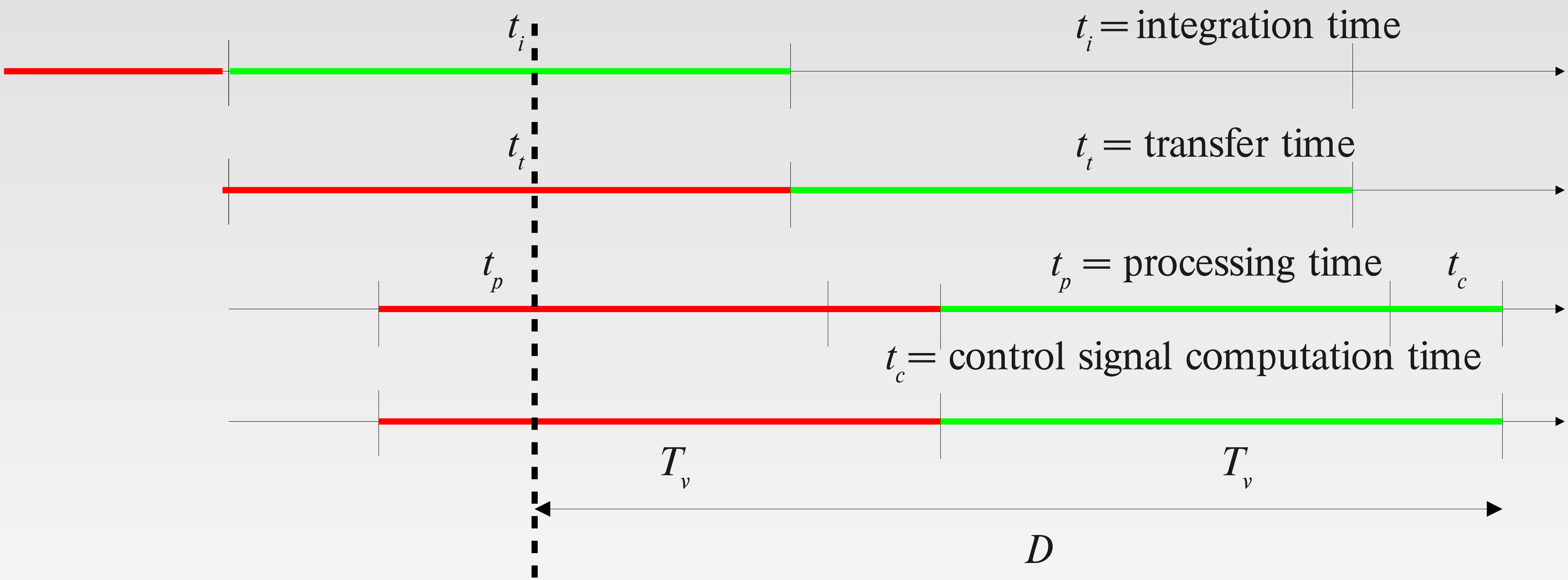
- Image acquisition model
- Manipulator model
- Typical dynamic visual loop

3.2.1 Image acquisition model

- Acquisition sequence :
  - Integration of light-generated charges in pixels
  - Readout of the charges :
    - Global shutter : all charges are read together
    - Rolling shutter : each charge is read sequentially
    - Progressive scan : the whole image is scanned at once
    - Interlaced scan : even and odd lines are scanned alternatively
  - Image transfer to the acquisition board :
    - Analog transfer : coaxial cable (50 Mpix/s max)
    - Digital transfer : USB2 (480 Mbits/s), Firewire (800 Mbits/s), LDVS (4 Gbits/s), Gigethernet (1 Gbits/s), Camlink (7.14 Gbits/s)

3.2.1 Image acquisition model

- Typical image acquisition time diagram :

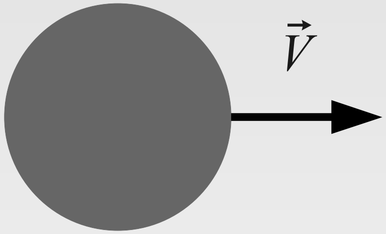


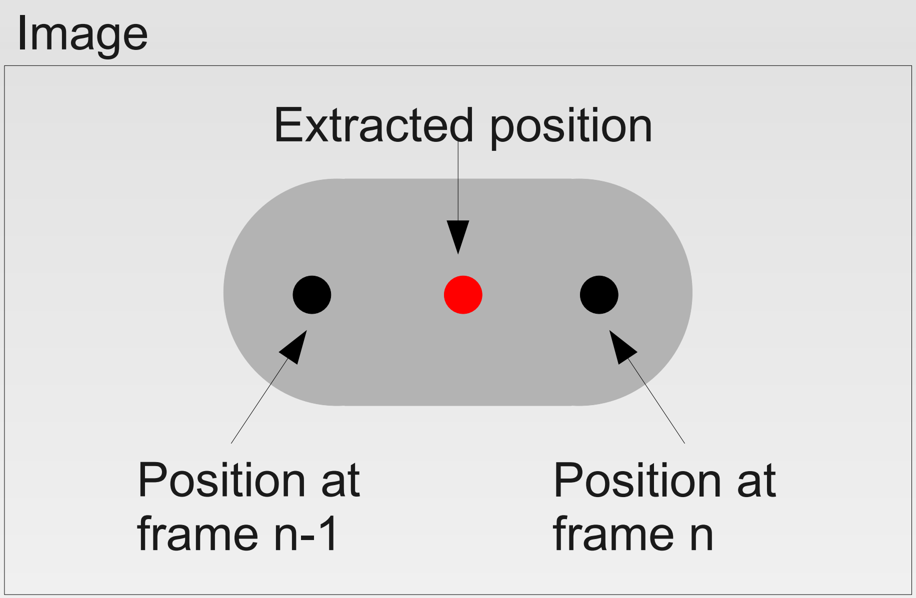
Equivalent image sampling time

Model = Delay D $\Leftrightarrow z^{\mathrm{int}\left(\frac{D}{T_v}+0.5\right)}$

3.2.1 Image acquisition model

- Motion blur :

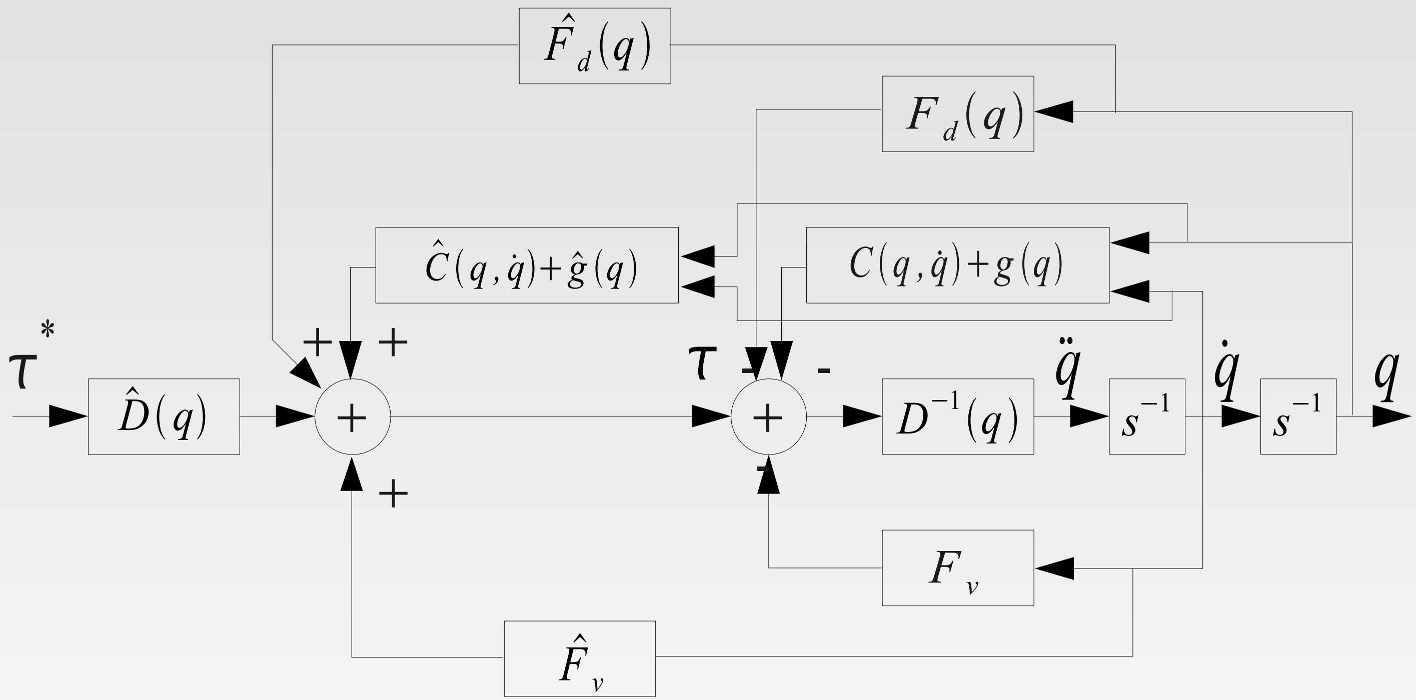




$$\text{Model = averaging filter} = \frac{1+z^{-1}}{2}$$

3.2.2 Manipulator model

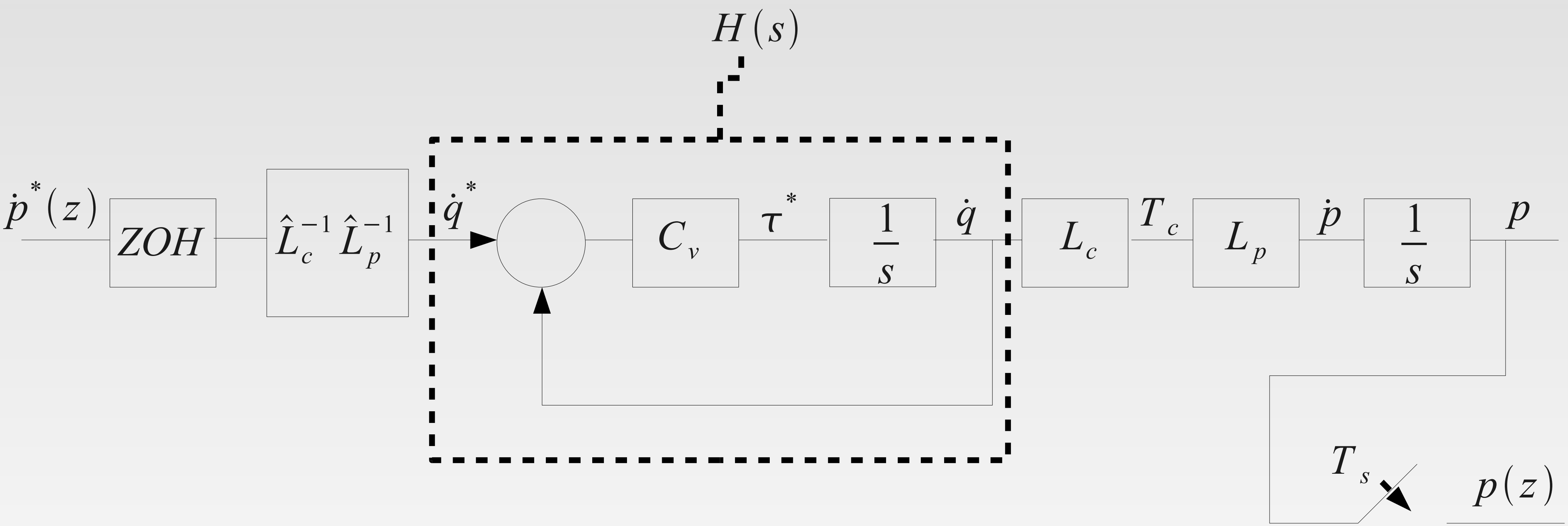
- Computed torque :



Equivalent model : $\dfrac{\dot{q}(s)}{\tau^*(s)}=\dfrac{1}{s}$

3.2.2 Manipulator model

- Virtual Cartesian motion device :



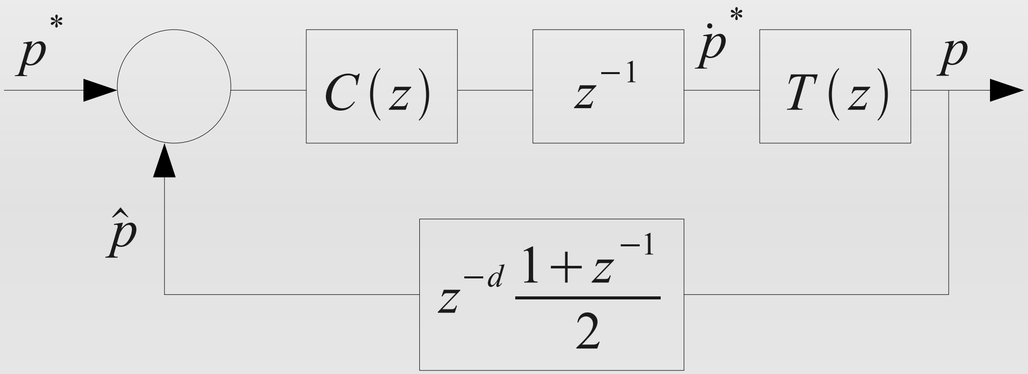
$$T(z) = \frac{p(z)}{\dot{p}^*(z)} = (1 - z^{-1}) Z \left\{ \frac{L_p L_c H(s) \hat{L}_c^{-1} \hat{L}_p^{-1}}{s^2} \right\} \approx (1 - z^{-1}) Z \left\{ \frac{H(s)}{s^2} \right\}$$

3.2.2 Manipulator model

- Notations :

  $\dot{p}^*$: reference velocity
  ZOH : Zero Order Hold (model of the DACs)
  $\hat{L}_p$: estimated Jacobian of the representation s.t. : $\dot{p}=\hat{L}_p T_c$
  $\hat{L}_c$: estimated Jacobian of the robot s.t. : $T_c=\hat{L}_c \dot{q}$
  $\dot{q}^*$: joint velocity reference
  C_v : velocity controller of the decoupled robot
  τ^* : torque reference for the computed torque algorithm
  $\dot{q}$: joint velocity
  $H(s)$: model of the velocity controlled decoupled robot
  T_c : velocity skrew of the camera
  p : output (can be a pose (2D) or a feature vector(3D))
  T_s : sampling period

3.2.3 Typical dynamic visual loop



p^* : reference
$\hat{p}$: visual measurement
$C(z)$ visual loop controller

- Notes :
  - Controllability : the dimension of *p* should be less or equal to the number of degrees of freedom of the manipulator in order for the system to be controllable. So in case of IBVS, the number of features coordinates should be restricted to 6. When using more feature coordinates, PBVS should be preferred.
  - The additional delay in the direct transfer is due to the time needed to compute the control signal (applied at the next iteration).

4. Conclusion

- Kinematic visual servoing :
  - Slow, simple to implement
  - Suited when robustness and accuracy are more important than speed
  - Applications : positioning of surgical instruments with respect of static structures (brain surgery, orthopedic surgery)
- Dynamic visual servoing :
  - Fast, can by tricky to tune
  - Suited when advanced control techniques are relevant (predictive control, repetitive control)
  - Applications : physiological motion compensation

4. Conclusion

- Perspectives :
  - High bandwidth visual servoing with medical imaging feedback (X-ray, CT, MRI, Ultrasound)
  - Issue : high bandwidth and low latency access to the low level data

Thank you !