N^o d'ordre : 4247 ANNÉE 2010

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale MATISSE

présentée par

Amaury DAME

préparée à l'unité mixte de recherche 6074 IRISA
Institut de Recherche en Informatique et Systèmes Aléatoires

A unified direct approach for visual servoing and visual tracking using mutual information

Thèse soutenue à l'IRISA
le 23/12/2010

devant le jury composé de :

Adrien BARTOLI
Professeur Université d'Auvergne/ rapporteur

Gregory D. HAGER
Professor The John Hopkins University/ rapporteur

Patrick BOUTHEMY
Directeur de recherche INRIA / examinateur

Vincent LEPETIT
Senior Researcher EPFL / examinateur

Ezio MALIS
Chargé de recherche INRIA / examinateur

Éric MARCHAND
Professeur Université de Rennes 1 / directeur de thèse

Acknowledgments

First of all, I thank Patrick Bouthemy for having accepted to be president of the jury, Professor Adrien Bartoli and Professor Gregory D. Hager for being the reviewer and senior researcher Vincent Lepetit for his examination of the thesis.

Special thanks go to my supervisor, Eric Marchand. He advised me to do this thesis with him and if I had to do the choice again I won't hesitate to do this again. He chose a good solution to supervise me, acting as a guide and as a friend. It was just perfect!

I also want to thank all the members of the Lagadic team. I will begin with François Chaumette that hosted me in his team and with whom I had good time talking about different research topics. I thank Céline who had the nerve to share her desk with me. I had really good times with you Céline! My thanks go to the ones that help the team work so nicely, Céline Ammoniaux, Fabien Spindler, and the engineers Anthony, Nicolas and Romain. Thanks also to all the other members of the team that make such a pleasant working environment.

My thanks also go to Robert Mahony that hosted me in the Australian National University in Canberra for one month. He helped me have a clear look upon my researches and opened me new perspectives.

Finally thanks for encouragement and support go to my family. Thanks to all the nice persons I met in my numerous trips that shared their culture and happiness with me and gave me the urge to share more and more. And of course special thanks to my friends and flatmates that made my life during the PhD thesis even happier and exciting.

Contents

Introduction

Vision is a *in* very powerful tool that allows humans to interact with our environment with simplicity and accuracy. In this process, the eyes only perform a very small part of the job. The efficiency to analyze our visual environment is indeed mainly due to our brain. The parallelism with the computer vision systems (computer equipped with visual sensors) is immediate: the camera acquires the appearance of the environment as an image that is then analyzed by the computer. The main difference is that the human has a large *a priori* knowledge of its environment and has many years of evolution behind him. Since the low-level human interpretation of an image is mostly unconscious, many people are still wondering why there are still researches in the vision domain to achieve a task that seems so simple. But the huge literature on the subject proves the problem to be even more complex than it seems.

Work in computer vision is motivated by the wide applications that it offers. Indeed, the partial reproduction of the vision sens applied to computers would have significant repercussions in many domains: health care, surveillance, indexation, robot control and many others. Although a human being has a good interpretation of his environment, it lacks the accuracy, repeatability and strength of a robot. While many researches are focused on machine learning or artificial intelligence to perform some complex tasks, some more "basic" researches on low-level detection, tracking, matching or segmentation tasks are still under progress. In this thesis, we address two of these tasks: the tracking problem, where the goal is to estimate the position of an object in an image sequence or video and the visual servoing problem, that consists of positioning a robot using the information provided by a visual sensor.

To perform this visual tracking and visual servoing tasks, many approaches propose to extract visual features such as key-points or contours from the images and estimate this displacement or control the robot using the geometrical information provided by the features. Nevertheless, most of these approaches do not use the whole information provided by the images, but only some geometrical features that are suffering from measurement errors which may affect the accuracy of the tasks. In this thesis, we focus on direct approaches: instead of using local features, the object is directly represented by the whole set of pixel intensities of the image where it appears. Therefore, all the information provided by the image is used. In this approach a measure of similarity between two patches of intensities has to be considered to realize the tracking and visual servoing tasks. Depending on the external conditions when the image is acquired, the appearance of the object can undergo many variations. Therefore, the similarity measure, or alignment function, has to be robust to such appearance variations.

After having evaluated many alignment functions, our interest has been caught by the mutual information function that had significant repercussions in the medical field, due to its robustness in the case of occlusions and multimodality of the images. Its latter appearance in tracking applications showed that it is also robust to illumination variations. One of the first ideas of this thesis was then to use the mutual information as a new metric for visual servoing tasks. To do so the direct visual servoing approach has been considered: the image acquired at the desired position is known, and, the goal is to make the camera move to the position that maximizes the mutual information between the current image and desired image. The visual servoing problem becomes, therefore, an optimization problem, where the mutual information has to be differentiated with respect to the degrees of freedom of the camera/robot. First, the classical optimization methods, that were common to the mutual information maximization, were used. However, our research has shown that some very significant improvements of the performances in terms of accuracy and robustness could be obtained using a new optimization approach. Since there is a strong duality between the visual tracking and visual servoing problem, a similar registration approach has been considered to create a new mutual information-based tracking method yielding the same improvements of its performance.

This thesis provides the knowledge required to perform a visual tracking and visual servoing task based on the mutual information. We present a comparative analysis of many tracking and visual servoing approaches to show the capability of the proposed approach. We present the researches that are significant in our context and present the work that has been performed to improve the existing approaches and create new tracking and visual servoing techniques. Many experiments demonstrate the improvements that have been achieved. Several applications illustrate the interest of such approaches in augmented reality, mosaicing or face tracking. Finally the visual servoing approach has been validated on a 6 degrees of freedom gantry robot and on a navigation task using a non-holonomic autonomous vehicle.

How is organized this thesis

This thesis has three major parts. The first part presents the basic knowledge on which our contributions are defined. Its first chapter provides the background in computer vision that is required to solve most of the visual tracking and visual servoing problems. Its second chapter provides the background in statistics and information theory that is necessary to define the mutual information function.

The second part presents the visual tracking part. A state of the art is presented where we show the limitation of the existing approaches. To the previous issues, we propose as an answer a mutual information-based tracking approach. The tracking method is first defined to track objects in the image using parametric motion models. Then the approach is extended to the estimation of the pose of the object in the 3D space.

In the last part, this optimization approach is applied to the pose of a real camera to solve a visual servoing problem. We will note that the visual servoing problem is dual with the previous pose computation problem. Therefore, the previous MI approach is simply adapted to the visual servoing tasks. The performances of the new visual servoing method are then evaluated in many experiments on a 6 degrees of freedom robot that shows the approach to be very accurate and robust. The convergence is analyzed and compared to classical visual servoing approaches. Finally an application of this new visual servoing metric is applied to the navigation of a non-holonomic autonomous vehicle.

Contributions

This work has led several contributions in both the computer vision and robotic domain. Following chronological order, here are the contributions with the corresponding publications:

- We propose a new tracking approach based on mutual information that allies robustness and efficiency [C4]. Many applications have demonstrated the efficiency of the method with augmented reality application and multimodal registration of images.

- The new tracking approach has been proposed as an alternative to the classical KLT approach that uses the difference between the pixel intensities and we show the advantages on the robustness of the estimation [C2]. The detection process that is optimal for the proposed tracking approach is proposed.

- A mutual information-based approach is also proposed to define a new control law for 2D visual servoing tasks [C1][C3][R1]. This positioning task is robust to partial occlusions and illumination variations compared to the existing direct approaches. Since it uses the whole information provided by the camera, it is also very accurate.

- The visual servoing approach has been adapted to propose a navigation approach for non-holonomic vehicles equipped with a camera.
- A solution that unifies both the visual tracking and direct visual servoing problems is proposed in [CN1].

Publications

Journal articles

[R1] A. Dame, E. Marchand. – Mutual information-based visual servoing. – In *IEEE Trans. on Robotics*, Conditionally accepted.

National conference papers

[CN1] A. Dame, E. Marchand. – Une approche unifiée reposant sur l'information mutuelle pour l'asservissement visuel et le suivi différentiel. – In *18e congrès francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle, RFIA 2010*, Caen, France, January 2010.

International conference papers

[C1] A. Dame, E. Marchand. – Entropy Based Visual Servoing. – In *IEEE Int. Conf. on Robotics and Automation, ICRA'09*, Kobe, Japan, May 2009.

[C2] A. Dame, E. Marchand. – Optimal Detection and Tracking of Feature Points using Mutual. – In *IEEE Int. Conf. on Image Processing, ICIP'09*, Cairo, Egypt, November 2009.

[C3] A. Dame, E. Marchand. – Improving mutual information based visual servoing. – In *IEEE Int. Conf. on Robotics and Automation, ICRA'10*, Anchorage, Alaska, May 2010.

[C4] A. Dame, E. Marchand. – Accurate real-time tracking using mutual information. – In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'10*, Seoul, Korea, October 2010 (ISMAR2010 Best paper runner-up award).

Notations

General

- Some detailed information are given in separate Frames. We decided to separate the content of these frames when the information which they provide is not required to understand the remainder of the document. These frames are usually defining some particular applications of the more general theory that is defined beside.

- the superscript * denotes a reference. For instance, if I is an image, then I^* is a reference image.

General mathematics

- $\mathbf{M} = (\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_n)$: vertical concatenation of matrices, vectors or scalars.
- $\mathbf{M} = [\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_n]$: horizontal concatenation of matrices, vectors or scalars.
- $\mathbb{R}^n$: Real space with n dimensions.
- $[\boldsymbol{\omega}]_\times$: skew-symmetric matrix of $\boldsymbol{\omega}$.

Euclidean and projective geometry

- $\mathcal{F}$: Cartesian frame.
- $\mathcal{X}$: point in the Euclidean space.
- $\mathbf{X} = (X, Y, Z)$: vector representing the coordinates of a point in the Euclidean space.
- $\tilde{\mathbf{X}} = (\mathbf{X}, 1) = (\lambda\mathbf{X}, \lambda)$: vector representing the homogeneous coordinates of a point in the Euclidean space.
- $\mathbf{m} = (x_m, y_m)$: image point coordinates in normalized form.
- $\tilde{\mathbf{m}} = (\mathbf{m}, 1)$: image point coordinates in homogeneous normalized form.
- $\mathbf{x} = (x, y)$: image point coordinates in pixels.
- $\tilde{\mathbf{x}} = (\mathbf{x}, 1)$: image point coordinates in homogeneous pixel formulation.
- ${}^j\mathbf{M}_i = \begin{bmatrix} {}^j\mathbf{R}_i & {}^j\mathbf{t}_i \\ \mathbf{0} & 1 \end{bmatrix}$: homogeneous transformation matrix (from the frame $\mathcal{F}_i$ to the frame $\mathcal{F}_j$).
- ${}^j\mathbf{R}_i = \exp([\boldsymbol{\omega}]_\times)$: rotation matrix
  - $\boldsymbol{\omega} = \theta\mathbf{u}$: rotation vector.
  - $\mathbf{u}$: rotation axis.
  - $\theta = ||\boldsymbol{\omega}||$: rotation angle.
- ${}^j\mathbf{t}_i = (t_x, t_y, t_z)$: translation vector.

- $\mathbf{r} = (\theta\mathbf{u}, \mathbf{t})$: pose parameters (parametrization of an homogeneous transformation matrix).
- $\mathbf{K} = \begin{bmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$: intrinsic matrix of a camera ($\tilde{\mathbf{x}} = \mathbf{K}\tilde{\mathbf{m}}$).
  - (p_x, p_y): ratio between the focal and the pixel size.
  - (u_0, v_0): coordinates of the principal point in pixels.

Tracking

- $w(\mathbf{x}, \mathbf{p})$: warping function or parametric motion model that transforms the position of the point $\mathbf{x}$ into a new position.
- $\mathbf{p}$: parameter associated to a parametric motion model.
- $\mathbf{G}$: Gradient matrix.
- $\mathbf{H}$: Hessian matrix.

Visual servoing

- $\mathbf{v} = (\boldsymbol{\nu}, \boldsymbol{\omega})$: velocity vector.
  - $\boldsymbol{\nu} = (\nu_x, \nu_y, \nu_z)$: translation velocity vector.
  - $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$: rotation velocity vector.
- $\mathbf{s}$: visual feature.
- $\mathbf{L_s}$: interaction matrix related to $\mathbf{s}$.

Part I

Background knowledge

Chapter 1

Background on computer vision

This chapter presents the geometric tools required to build our tracking and visual servoing approaches. The definition of the Euclidean geometry allows for modeling our 3D space in which our camera is moving, and, from which the projection into the image space is performed. We explain how the real world projects itself into the digital images. From the definition of the 3D geometry and its projection, the effect in the image of the object or camera motions is presented. Several models of transformations, or warping functions, are then defined to introduce their use in the visual tracking problem.

1.1 Modeling the world

Euclidean geometry is a mathematical system that originally describes the two and three-dimensional planar spaces. It is the ideal tool to give a simple and practical model of the 3D world in which the object and camera are moving. In this section, we suppose that the reader knows the basics of linear algebra. If the reader needs more details on this subject, he can find them, for example, in [Ma 2004].

1.1.1 Euclidean geometry

To define the coordinates of an object in the space, first, it is required to define a frame in which the coordinates will be expressed. Any object can be expressed relatively to any frame. Nevertheless, if the tracked object is known, it is more likely in a frame $\mathcal{F}_o$ attached to the object. Then, when we want to study the position of the object with respect to the camera, the goal is to express the coordinates of the object in a frame attached to the camera noted $\mathcal{F}_c$. Therefore, we will choose the object and camera frames as an example to illustrate the transformation of the coordinates from one frame to a new one. In the remainder of this section, we suppose that the object is fixed and that the camera frame is changing but we have to keep in mind that this problem is dual with the one considering a moving object and a fixed frame.

As the pose of an object is defined by a 3D position (for instance its center) and by its orientation, we define the object frame (resp. the camera frame) by its position using its origin $\mathcal{X}_o$ (resp. $\mathcal{X}_c$) and its orientation using a basis of 3 orthonormal unitary vectors $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ as illustrated in Figure 1.1.

An interesting property of the Euclidean geometry is that any geometric feature can be defined using points (a line can for instance be defined by two points). Let us then limit ourselves to the study of points in the 3D space. To locate each point $\mathcal{X}$ in the euclidean space, three coordinates are required. The three coordinates are typically defined in the object frame as ${}^o\mathbf{X} = ({}^oX, {}^oY, {}^oZ)$ so that:

$$\mathcal{X} = \mathcal{X}_o +^o X\mathbf{i}_o +^o Y\mathbf{j}_o +^o Z\mathbf{k}_o \tag{1.1}$$

To change the coordinates of the point from the object frame to the camera frame, a transformation that models both the change in position an orientation is defined. The position transformation is defined by a 3D translation ${}^c\mathbf{t}_o$ that transforms the center of the object frame into the center of the camera frame, and the orientation by a 3D rotation ${}^{c}\mathbf{R}_{o}$ that defines the transformation from the axes of the object frame to the axis of the camera frame. The coordinates of the point $\mathcal{X}$ in the object frame ${}^{o}\mathbf{X}$ are then given by the affine transformation:

$$ {}^{c}\mathbf{X} = {}^{c}\mathbf{R}_{o}\, {}^{o}\mathbf{X} + {}^{c}\mathbf{t}_{o} \tag{1.2} $$

To put this affine formulation into a simpler form, a linearization using the homogeneous formulation is usually preferred. A point $\mathcal{X}$ defined by its coordinates $\mathbf{X} = (X, Y, Z)$ in the Euclidean space can also be defined in a projective space by its homogeneous coordinates $\tilde{\mathbf{X}} = (\mathbf{X}, 1) = (\lambda\mathbf{X}, \lambda)$ where λ is a non-null scalar. Using the homogeneous coordinates of $\mathcal{X}$ in both the object and camera frame we can write the equivalent homogeneous formulation of the equation (1.2) as:

$$ {}^{c}\tilde{\mathbf{X}} \;=\; {}^{c}\mathbf{M}_{o}\, {}^{o}\tilde{\mathbf{X}} \qquad \text{with:} \qquad {}^{c}\mathbf{M}_{o} = \begin{bmatrix} {}^{c}\mathbf{R}_{o} & {}^{c}\mathbf{t}_{o} \\ \mathbf{0} & 1 \end{bmatrix} \tag{1.3} $$

where the homogeneous matrix ${}^{c}\mathbf{M}_{o}$ contains the whole transformation from the object frame to the camera frame that typically defines the camera pose. ${}^{c}\mathbf{R}_{o}$ is a 3×3 matrix and ${}^{c}\mathbf{t}_{o}$ is a 3×1 vector, therefore, the homogeneous matrix ${}^{c}\mathbf{M}_{o}$ originally contains 12 values. Since these values respect several constraints, it is in fact possible to represent the whole transformation with fewer parameters.

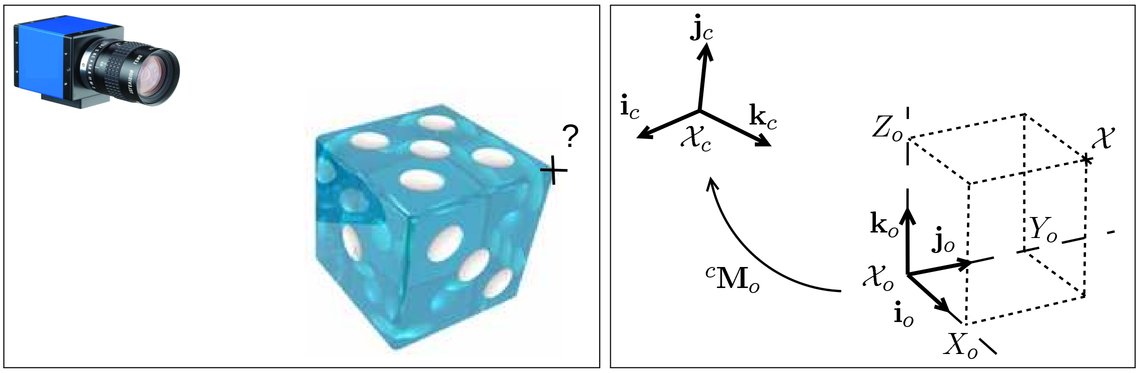


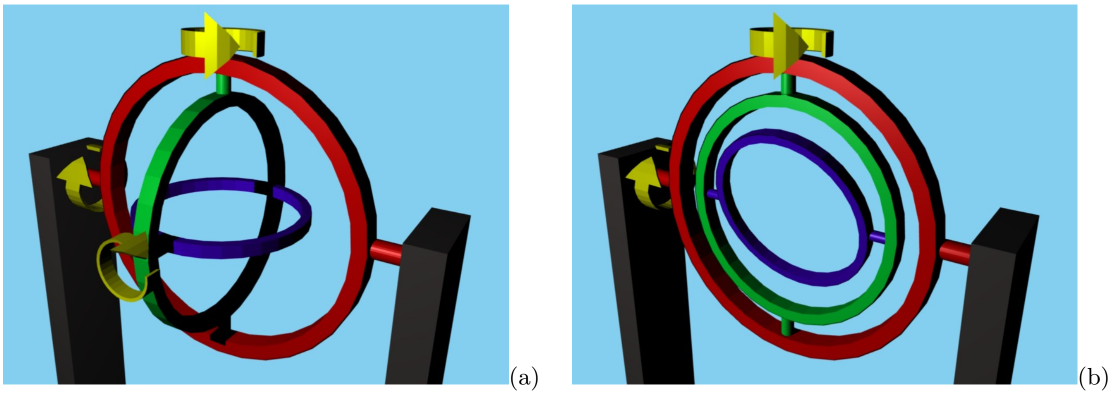
Figure 1.1: Representation of the 3D space using Euclidean geometry.

1.1.2 Homogeneous matrix parametrization

Several solutions exist to parametrize the 3D transformation generated by an homogeneous matrix. In this section, we focus only briefly on the most intuitive solutions in our point of view. There is no trouble defining the translational part of the pose. Indeed the translation part of the homogeneous matrix that changes the coordinates of a 3D point from the object frame to the camera frame is already defined as a set of 3 parameters, that are the translations along the three axis of the camera frame.

The problem is different for the rotational part. Indeed, the rotation is defined by a 3×3 matrix. If no constraint on the matrix is considered, it corresponds to 9 values. However, since the displacement of an object must not affect the object size or its orientation, the rotation matrix has to be a special orthogonal matrix that means that its columns define 3 orthogonal unit vectors (typically the axis of the object frame defined in the camera frame).

Several parametrization of the rotation matrix are possible to respect these constraints. A first solution, the Euler representation, is to consider the 3D rotation as a combination of three consecutive rotations around the axes of the space. The rotation matrix is, therefore, represented as the product of the 1D rotations around the X, Y and Z axes. Every 1D rotation is defined by one angle of rotation (typically r_x,r_y and r_z) and the parametrization of the 3D rotation is then simply defined by a 3D vector (r_x, r_y, r_z). This parametrization is very satisfying in terms of simplicity, nevertheless, it presents one drawback. Several configurations of the rotation angles cause one degree of freedom to be lost. This singularity problem can be illustrated with a mechanical system of the Euler representation that is shown in Figure 1.2: each gimbal can rotate around one axis (3 gimbals = 3×1D rotation). If two rotation axes are aligned (Figure (b)) then one degree of freedom is lost (the singularity is usually called gimbal lock).


(a)

(b)

Figure 1.2: Three axis gimbal set: the blue gimbal can rotate in 3 dimensions. (a) Position with three degree of freedom, and (b) one singularity position known as gimbal lock: since the first and last rotation axis are equal, one degree of freedom is lost.

Another representation is the Exponential map. This parametrization assumes that all the rotations in the 3D space can be parametrized by a rotation around one axis of the 3D space. Therefore, the parameters contain two information: the axis of the rotation and its angle. The Exponential map parametrizes this rotation as a vector $\boldsymbol{\omega} = (\boldsymbol{\omega}_x, \boldsymbol{\omega}_y, \boldsymbol{\omega}_z)$ where the rotation axis is defined by the orientation of $\boldsymbol{\omega}$ and the angle of rotation by its norm $\theta = ||\boldsymbol{\omega}||$.

Using this parametrization and the Lie algebra, the rotation matrix can be written as:

$$\begin{aligned} \mathbf{R} &= \exp(\boldsymbol{\Omega}) \\ &= \mathbf{I} + \boldsymbol{\Omega} + \frac{1}{2!}\boldsymbol{\Omega}^2 + \frac{1}{3!}\boldsymbol{\Omega}^3 + \ldots \end{aligned} \tag{1.4}$$

where $\boldsymbol{\Omega}$ is the skew-symmetric matrix of $\boldsymbol{\omega}$:

$$\boldsymbol{\Omega} = [\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\boldsymbol{\omega}_z & \boldsymbol{\omega}_y \\ \boldsymbol{\omega}_z & 0 & -\boldsymbol{\omega}_z \\ -\boldsymbol{\omega}_y & \boldsymbol{\omega}_z & \boldsymbol{\omega}_x \end{bmatrix}. \tag{1.5}$$

We see that the name exponential Map comes from the fact that the equation (1.4) is very similar to the Taylor expansion of a classical exponential. This representation presents singularities for $\theta = 2\pi n$ for $n \geq 1$, nevertheless, these singularities are easily avoided: when the parameters are near a singularity, its values are set to a new equivalent rotation far from the singularity.

1.2 Projection in the image space

As the previous section shows, the Euclidean model is a nice way to model our environment. However, our problem does not, in general, induce a direct knowledge of the world. Our perception of the environment is usually limited to its 2D projection in an image. Depending on the camera, the projection from the 3D world to the image space can have many forms. In this thesis we will only focus on the pin-hole (or perspective) model that is the most widespread type of camera.

1.2.1 Pin-hole camera model and digital images

To model the displacement of an object in an image, we must know how the image is created beforehand. We are not particularly interested in the mechanic or photometric definition of a camera, but more in the geometrical projection from the 3D space into the 2D image space and to the transformation that changes the euclidean coordinates of the projection into the pixel coordinates in the image.

1.2.1.1 Projection into the image plane

The optical system in a perspective camera is constructed to capture the incoming light rays that are directed to a single point. The perspective cameras thus belong to the single-viewpoint camera classes. This point of the camera is called the center of projection.

To have a clear vision of the perspective projection, a simple way is to have a look at the history of photography. Indeed, we can assume that photography is born from a device called *camera obscura*. This optical device is similar to the first cameras, as the difference the resulting image was manually drawn instead of being acquired by chemistry. *Camera obscura* is composed by two essential elements: one box with a hole that separates the outside bright environment from the acquisition "room" and a plane in the unlit room (see Figure 1.3(b)). The rays from the environment are directed into the hole and intersect the plane where an inverted image of the environment appears.

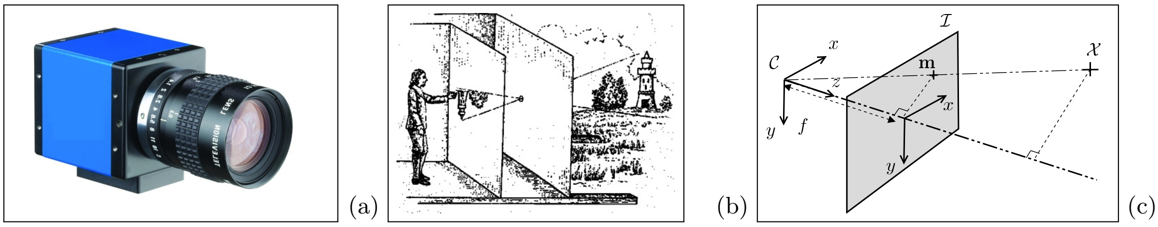


Figure 1.3: A camera and its perspective projection model: (a) usual camera, (b) camera obscura and (c) perspective projection model.

The current pin-hole cameras respect the same principle. The hole is called the projection center and the plane, where the light rays are projected, is called image plane or projection plane. To simplify the problem, we consider a frontal pinhole imaging model meaning that the image plane is located between the center of projection and the scene. Then, the geometrical model corresponds to the projection represented in Figure 1.3(c). The camera frame is chosen such that the z axis (called the focal axis) is orthogonal to the projection plane. The coordinates $\mathbf{m} \in \mathbb{R}^2$ of the projection of $\mathcal{X}$ in the image are then given by the Thales theorem:

$$\mathbf{m} = \begin{bmatrix} x_m \\ y_m \end{bmatrix} = \frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{1.6}$$

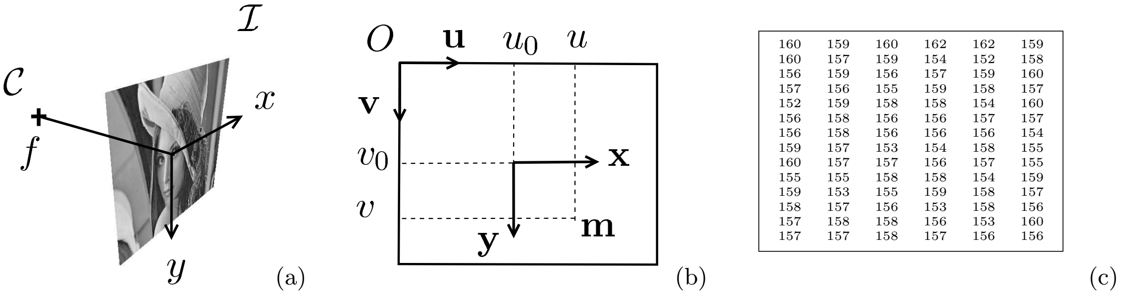
where ${}^{c}\mathbf{X} = (X, Y, Z)$ are the coordinates of the point in the camera frame and f is the focal length, *i.e.* the distance between the center of projection and the image plane. To write the projection operation in a linear form, we express in homogeneous formulation both the coordinates of the point in the image $\tilde{\mathbf{m}} = (\lambda\mathbf{m}, \lambda)$ and in the 3D space $\tilde{\mathbf{X}} = (\lambda\mathbf{X}, \lambda)$:

$$\tilde{\mathbf{m}} = \mathbf{A}\tilde{\mathbf{X}} \qquad \text{with} \qquad \mathbf{A} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.7}$$

The resulting coordinates, called normalized coordinates, define the location of the projected point in the 2D euclidean space corresponding to the projection plane. This coordinates are, therefore, typically expressed in the metric space. One supplementary step is then necessary to convert them into the coordinates of the point in the digital image.

1.2.1.2 From the metric space to the pixel space

A digital image is a set of pixel intensities organized on a regular grid. An image I with a width of w pixels and height of h pixels is thus basically defined as a $w \times h$ matrix where each element of the matrix defines a pixel intensity. Physically, the pixel intensities are acquired using a grid of electronic image sensors placed on the image plane. Each pixel encodes the value of the perceived intensity. The matrix form of the image induces to have only integer coordinates in the pixel space. But first, let us consider that both the metric space and pixel space are defined on $\mathbb{R}$, then, the next section will show how to estimate the intensity of the image at a non integer position.



160	159	160	162	162	159
160	157	159	154	152	158
156	159	156	157	159	160
157	156	155	159	158	157
152	159	158	158	154	160
156	158	156	156	157	157
156	158	156	156	156	154
159	157	153	154	158	155
160	157	157	156	157	155
155	155	158	158	154	159
159	153	155	159	158	157
158	157	156	153	158	156
157	158	158	156	153	160
157	157	158	157	156	156

Figure 1.4: Formation of the digital image. (a) Projection of the image, (b) conversion from the metric space of the projection plane to the pixel space of the digital image and (c) part of the resulting digital image represented as a matrix.

To convert the coordinates from meters to pixels, the position of the grid in the image plane is usually parametrized as a four degrees of freedom (DOF) transformation. This transformation, illustrated in Figure 1.4, is defined by the coordinates (u_0, v_0) of the principal point[1] in pixel and also by (l_x, l_y) the pixel size. The function that changes the coordinates of a point from meters (x, y) to pixels $\mathbf{x} = (u, v)$ is then defined by:

$$\left\{ \begin{array}{lcl} u & = & u_0 + \frac{1}{l_x} x \\ v & = & v_0 + \frac{1}{l_y} y \end{array} \right. \tag{1.8}$$

This transformation can be written in a linear form using the homogeneous coordinates. If the pixel is written in its homogeneous form $\tilde{\mathbf{x}} = (\lambda\mathbf{x}, \lambda)$ then an equivalent formulation of the

[1] where the z axis of the camera frame intersects the image plane, i.e. the point with coordinates $(0, 0)$ in the 2D meter space or $(0, 0, f)$ in the 3D space

transformation is:

$$\tilde{\mathbf{x}} = \mathbf{K}'\tilde{\mathbf{m}} \qquad \text{with} \qquad \mathbf{K}' = \begin{bmatrix} \frac{1}{l_x} & 0 & u_0 \\ 0 & \frac{1}{l_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.9}$$

If we couple this equation with the equation (1.7) of the projection and put the focal term of $\mathbf{A}$ into the matrix $\mathbf{K}'$, then, the whole projection of the 3D point into the pixel space is formulated as:

$$\tilde{\mathbf{x}} = \mathbf{K}\mathbf{\Pi}^c\tilde{\mathbf{X}} \qquad \text{with} \qquad \mathbf{\Pi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.10}$$

$$\text{and} \qquad \mathbf{K} = \begin{bmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.11}$$

or equivalently:

$$\tilde{\mathbf{x}} = \mathbf{K}^c\mathbf{X} \tag{1.12}$$

where p_x (resp. p_y) is the ratio between the focal length and the pixel width (resp. height): $p_x = f/l_x$ (resp. $p_y = f/l_y$). $\mathbf{K}$ contains all the parameters peculiar to the camera, and is called the intrinsic matrix. To have the projection of a point $\mathcal{X}$ defined in the object frame, the homogeneous matrix is introduced in the expression and the whole projection becomes:

$$\tilde{\mathbf{x}} = \mathbf{K}\mathbf{\Pi}^c\mathbf{M}_o\,{}^o\tilde{\mathbf{X}} \tag{1.13}$$

where ${}^c\mathbf{M}_o$ contains the parameters that define the position and orientation of the camera called extrinsic parameters. An estimation of the intrinsic parameters is usually provided by the camera constructor. These values are general estimations and the parameters of each camera can strongly vary due to the fabrication process. A better estimation of the parameters is therefore required. This estimation called camera calibration is generally performed using a set of images where some known 3D points ${}^o\tilde{\mathbf{X}}$ are projected at some known positions $\tilde{\mathbf{x}}$. This defines a system of equations from which the intrinsic parameters (u_0, v_0, p_x, p_y) can be estimated (during the process the extrinsic parameters are also estimated, nevertheless, it is usually not the purpose of the calibration).

1.2.1.3 Subpixel approach

The classical definition of the image only provides the intensity of a pixel at an integer position (u, v). Since we require a high accuracy in our tracking algorithm, it is sometimes necessary to get the intensity of the image at a non integer position, a subpixel approach is then needed (see Figure 1.5(a)). The evaluation of the intensity of an image at a non integer position is called image interpolation. The most simple approach is to consider that the intensity of the image at a real position is given by the intensity of the closest neighboring pixel. In Figure 1.5(a) it corresponds to choose $I(\mathbf{x}) = I(\mathbf{x}_{00})$. However, the resulting intensities of the image are not smooth and it usually causes the computer vision algorithms to lack accuracy. When an high accuracy is needed, an interpolation using the intensities of several neighboring pixels is generally preferred.

A common approach is the bilinear interpolation that computes the intensity of a pixel $\mathbf{x}$ at a non integer position with respect to the intensities of its four nearest neighbors. If the position of $\mathbf{x}$ is expressed with respect to the position of its neighbors as (see Figure 1.5):

$$\mathbf{x} = \mathbf{x}_{00} + \alpha_u(\mathbf{x}_{01} - \mathbf{x}_{00}) + \alpha_v(\mathbf{x}_{10} - \mathbf{x}_{00}) \tag{1.14}$$

then the interpolated intensity of $\mathbf{x}$ is:

$$\begin{aligned} I(\mathbf{x}) &= \alpha_v((1-\alpha_u)I(\mathbf{x}_{00}) + \alpha_u I(\mathbf{x}_{01})) \\ &+ (1-\alpha_v)((1-\alpha_u)I(\mathbf{x}_{10}) + \alpha_u I(\mathbf{x}_{11})) \end{aligned} \tag{1.15}$$

This is typically the expression of a first linear interpolation of the intensities along the u axis and then one along the v axis (see Figure 1.5(b)).

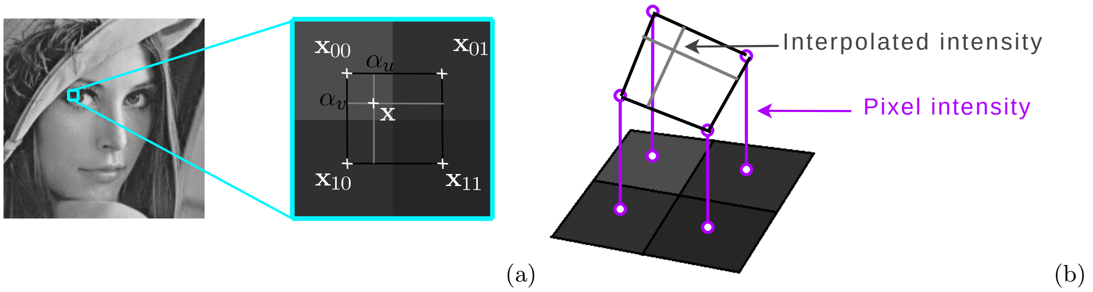


Figure 1.5: Numerical image: subpixel approach. (a) Part of lena, a classical digital image and zoom in and (b) bilinear interpolation illustration.

1.2.2 Parametrization of the image motion: the warp functions

In the second part of this thesis, our goal will be to track a rigid object projected onto the image plane. No prior information on the object will be known. The tracking process will therefore only define the displacement of the object in the image space.

The goal is then to find the transformation function, called warp function or parametric motion model, that best maps the points $\mathbf{x}^*$ of a first image I_1 into the points $\mathbf{x}$ of another image of the same object. Depending on the object itself and its motion in the 3D space, many warp functions can be defined. A general notation of the warp function is written as:

$$\begin{aligned} w : \quad \mathbb{P}^2 \times \mathbb{R}^n &\rightarrow \mathbb{P}^2 \\ (\mathbf{x}^*, \mathbf{p}) &\mapsto \mathbf{x} \end{aligned}$$

where n is the number of parameters of the considered motion model which corresponds to the number of DOF that defines the image transformation. To illustrate the following warping functions and one of their applications, we use the mosaicing application presented in Frame 1.

1.2.2.1 Translation model

The simplest model is to consider that the object moves in the image with a translational motion (see Figure 1.6). A 2D translation vector $\mathbf{t}$ that defines the horizontal and vertical translations is applied to the coordinates of the point $\mathbf{x}$ as:

$$w(\mathbf{x}, \mathbf{t}) = \mathbf{x} + \mathbf{t} \tag{1.16}$$

Frame 1 Application of various warping functions in a mosaicing problem.

Image mosaics are a collection of overlapping images. The goal of the mosaicing problem is to find the transformations that relate the different image coordinates. Once the transformation between all the images is known, an image of the whole scene can be constructed. This problem requires to find a warping function that maps the coordinates of one image into the coordinates of another image. The presented warping functions have been evaluated on a simple set of two overlapping images.

The images are matched using M couples of points that are manually selected. Depending on the warping function, the displacement parameters $\mathbf{p}$ are then retrieved from the set of points $\mathbf{x}$ and $\mathbf{x}^*$ using a linear operation or a non-linear optimization. The displacement parameters are typically estimated by searching the parameters that minimize the error between the positions $\mathbf{x}^*$ of the point in second image and the transformation of the points from the first image to the second image $w(\mathbf{x}, \mathbf{p})$:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{m=1}^{M} \|\mathbf{x}_m^* - w(\mathbf{x}_m, \mathbf{p})\|$$

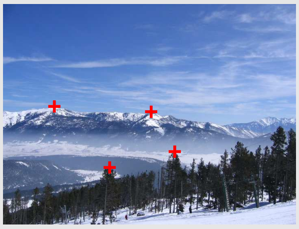
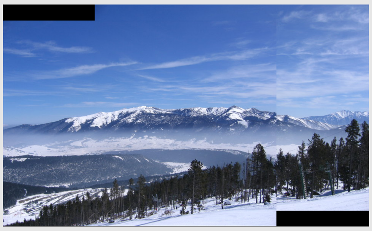
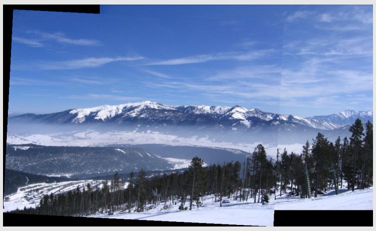
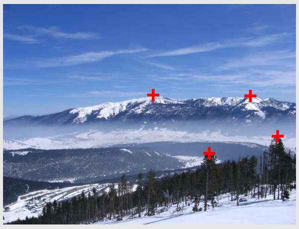
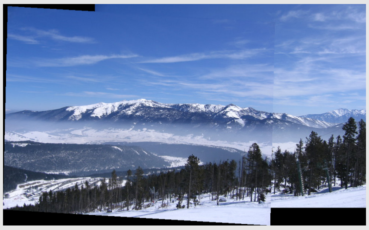
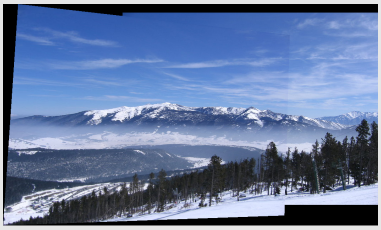
For instance if we want to estimate a translation then the equation becomes:

$$\widehat{\mathbf{t}} = \arg\min_{\mathbf{t}} \sum_{m=1}^{M} \|\mathbf{x}_m^* - (\mathbf{x}_m + \mathbf{t})\|$$

Since the function is minimal when its derivative is null, the problem is equivalent to:

$$\sum_{m=1}^{M} \left(\mathbf{x}_m^* - (\mathbf{x}_m + \widehat{\mathbf{t}})\right) = 0 \qquad \Longleftrightarrow \qquad \widehat{\mathbf{t}} = \frac{1}{M} \sum_{m=1}^{M} (\mathbf{x}_m^* - \mathbf{x}_m)$$

The estimation has been performed on the bottom images. When the warp function becomes more complex, the quality of the alignment increases.

Input images	*Mosaicing results*	
First image 	*Translation* 	*Similitude* 
Second image 	*Affine* 	*Homography* 

Using the projection model, it can be proved that this model depicts only the motion of a planar object that is parallel to the image plane and relatively moves parallely to it in a translational motion. At first sight, this model is thus of limited interest. In practice, it can be very useful. Let us first notice that both the displacement of the camera and the object will have an effect on the motion in the image. These effects are very different: a rotation of the object yields a large perspective effect in the image while a rotation of the camera will only cause a large motion in the image. In the second case, the translation warp function is then a good approximation if there is no rotation around the focal axis.

The advantage of this warp function is that the estimation of the displacement is very simple. Indeed only one couple of point locations $(\mathbf{x}^*, \mathbf{x})$ is required to evaluate the translation from one image to another, since one couple of points gives two equations, one for the horizontal displacement and one for the vertical displacement.

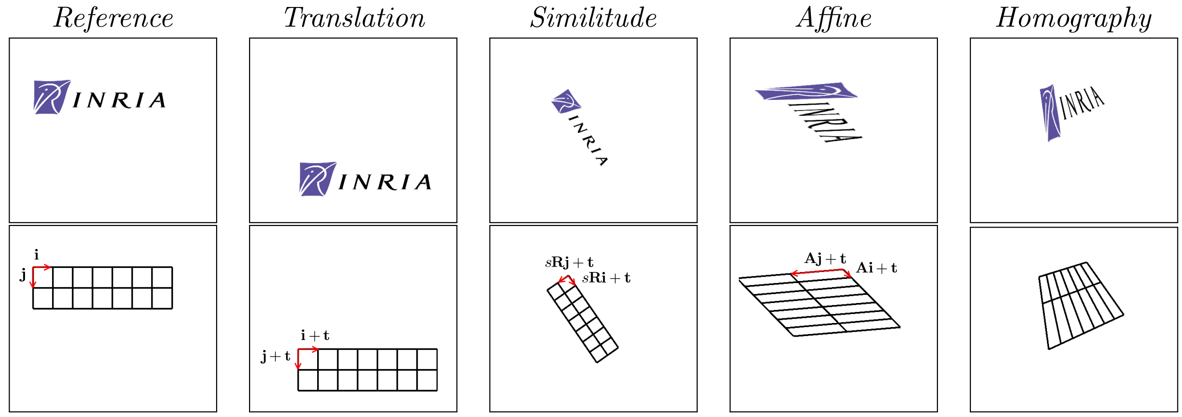


Figure 1.6: Classical warp functions. The homography defines the 3D motion of a planar object projected in the image, therefore it can not be represented as 2D change of basis.

1.2.2.2 Similitude transformation

Another frequently used warping function is the similitude transformation [Goodall 1991], also called Procrustes model or "shape-preserving mapping" model. This model considers the translation and adds the rotation φ and a scaling factor s in the image plane (see Figure 1.6). We can therefore express the function as:

$$w(\mathbf{x}, \mathbf{p}) = (1 + s)\mathbf{R}(\varphi)\mathbf{x} + \mathbf{t} \tag{1.17}$$

where $\mathbf{p} = (s, \varphi, \mathbf{t}) \in \mathbb{R}^4$ and $\mathbf{R}(\varphi) \in \mathbb{R}^{2\times 2}$ is the rotation matrix whose expression depends on the angle φ:

$$\mathbf{R}(\varphi) = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

Theoretically, this model only depicts the projection of a planar object that remains parallel to the image plane. In practice, if we consider that the camera is moving and not the object, then the further the object is from the camera, the less the depth of the object has an effect in the transformation, so the object's shape can be approximated as a plane. Such a situation is present, for instance, when a camera is mounted on the bottom of a quadrotor. Applications are then various, from localization to mosaicing.

1.2.2.3 Affine model

The affine model [Bookstein 1978] is a generalization of the similitude model that adds the possibility to follow an object whose projection is stretched along the two directions from one frame to another. Although it has no specific meaning in the 3D Euclidean space, it is a good approximation of a lot of motions in the image plane. As its name indicates, this warping function links the coordinates of a point in the two frames using an affine transformation, that is a linear transformation $\mathbf{A}$ followed by a translation $\mathbf{t}$:

$$w(\mathbf{x}, \mathbf{p}) = \mathbf{A}\mathbf{x} + \mathbf{t} \qquad \text{with} \qquad \mathbf{A} = \begin{bmatrix} 1 + a_{00} & a_{01} \\ a_{10} & 1 + a_{11} \end{bmatrix}$$
$$\text{and} \qquad \mathbf{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{1.18}$$

where $\mathbf{p} = (a_{00}, a_{10}, a_{01}, a_{11}, t_x, t_y) \in \mathbb{R}^6$. This transformation allows to approximate the displacement in the image of a planar object as soon as the perspective effects are not too important. For instance, in the mosaicing application of the Frame 1, the affine model is already giving a correct estimation of the transformation.

1.2.2.4 Homography

The last transformation that we consider is the homography transformation. In our domain, where an accurate estimation of the displacement is generally required, the homography is a suitable model. Contrary to the previous warping functions, the homography models the perspective effects in the image. Indeed, the homography, also called projective transformation, maps the displacement of a planar object from one frame to another. Therefore once it is estimated, many 3D properties of the object can be retrieved [Faugeras 1988].

Let us consider a planar object in the first camera frame $\mathcal{F}_1$. Each physical point $\mathcal{X}$ of the plane satisfies the equation:

$${}^1\mathbf{X}.\mathbf{n} = d \qquad \equiv \qquad \frac{\mathbf{n}^\top\, {}^1\mathbf{X}}{d} = 1 \tag{1.19}$$

where $\mathbf{n}$ is the unit vector normal to the plane expressed in $\mathcal{F}_1$ and d is the distance between the center of projection of $\mathcal{F}_1$ and the plane. Let us consider a second camera frame $\mathcal{F}_2$. The transformation from $\mathcal{F}_1$ to $\mathcal{F}_2$ can be written:

$${}^2\mathbf{X} = {}^2\mathbf{R}_1\, {}^1\mathbf{X} + {}^2\mathbf{t}_1 \tag{1.20}$$

where ${}^2\mathbf{R}_1$ (resp. ${}^2\mathbf{t}_1$) is the rotation matrix (resp. translation vector) from the frame $\mathcal{F}_1$ to the frame $\mathcal{F}_2$. If the translation is factorized by the expression of equation (1.19), the equation becomes:

$${}^2\mathbf{X} = \mathbf{H}{}^1\mathbf{X} \qquad \text{with} \qquad \mathbf{H} = {}^2\mathbf{R}_1 + \frac{{}^2\mathbf{t}_1\mathbf{n}^\top}{d} \tag{1.21}$$

where $\mathbf{H}$ is a 3×3 homography matrix that links the 3D coordinates of the point from the first to the second frame. Since the projections $\mathbf{x}_1$ and $\mathbf{x}_2$ of the point in both images respect the projection equation (1.12), we have:

$$\tilde{\mathbf{x}}_1 = \mathbf{K}{}^1\mathbf{X} \qquad \text{so} \qquad {}^1\mathbf{X} = \mathbf{K}^{-1}\,\tilde{\mathbf{x}}_1 \tag{1.22}$$

Using both the projection equation (1.12) on the second image and equation (1.21) the projection $\tilde{\mathbf{x}}_2$ can be rewritten as:

$$\begin{aligned}\tilde{\mathbf{x}}_2 &= \mathbf{K}\ {}^2\mathbf{X} \\ &= \mathbf{KH}\ {}^1\mathbf{X} \\ &= \mathbf{KH}\ \mathbf{K}^{-1}\tilde{\mathbf{x}}_1 \\ &= \mathbf{G}\ \tilde{\mathbf{x}}_1 \qquad \text{with} \qquad \mathbf{G} = \mathbf{KHK}^{-1}\end{aligned} \tag{1.23}$$

The homography matrix $\mathbf{G}$ is thus a 3×3 matrix, but, due to the homogeneous formulation, it is defined up to a scale factor and has in fact only 8 DOF. The degrees of freedom that we can retrieve from the factorized formulation in equation (1.21) correspond to the normal of the planar object (2 DOF) and to the rigid motion between the two camera poses (6 DOF). Using a special decomposition, it is possible to find back these parameters from the homography matrix [Faugeras 1988].

Considering that the homography matrix has 8 DOF, a first parametrization [Baker 2004] is to choose the displacement parameters $\mathbf{p}$ as a vector in $\mathbb{R}^8$ corresponding to eight elements of the matrix $\mathbf{G}$, the last element of the matrix is then set to 1:

$$\tilde{\mathbf{x}}_2 = \begin{bmatrix} 1+p_0 & p_2 & p_4 \\ p_1 & 1+p_3 & p_5 \\ p_6 & p_7 & 1 \end{bmatrix} \tilde{\mathbf{x}}_1 \tag{1.24}$$

Using the classical coordinates $\tilde{\mathbf{x}}_1 = ({}^1x, {}^1y, 1)$ and $\tilde{\mathbf{x}}_2 = ({}^2x, {}^2y, 1)$, it is equivalent to write:

$${}^2x = \frac{(1+p_0)\ {}^1x + p_2\ {}^1y + p_4}{p_6\ {}^1x + p_7\ {}^1y + 1} \qquad \text{and} \qquad {}^2y = \frac{p_1\ {}^1x + (1+p_3)\ {}^1y + p_5}{p_6\ {}^1x + p_7\ {}^1y + 1} \tag{1.25}$$

This parametrization is usefull since it is very simple to compute and obtain its analytical derivatives with respect to $\mathbf{p}$. Therefore, in this thesis, we will frequently use this model.

One of the drawbacks of this parametrization is that the resulting matrices are not necessarily corresponding to homography matrix as they are defined in equation (1.21). Both have 8 DOF, but the homography matrices $\mathbf{G}$ that describe a real displacement of a planar object in an image are part of a group called the Special Lie group $\mathbb{SL}(3)$. To limit the estimated homographies to this group, another parametrization is sometimes preferred as in [Benhimane 2004] where the determinant of $\mathbf{G}$ is fixed to 1 and the matrix is computed using the Lie algebra. Altough this formulation shows some advantages in the problem discussed in [Benhimane 2004], in our problem, it only limits our optimization formulations and causes problems due to the computation of a matrix exponential.

Since it has 8 DOF, a minimum of four couples of points are required to estimate the homography matrix. In the mosaicing application, the homography transformation between two images has been estimated using points that have been manually selected. We can see in Frame 1 that the reconstructed image shows a nice continuity.

1.3 Conclusion

In this chapter, we define the necessary tools to model a 3D environment as well as its projection in the 2D image plane and into digital images. Several classical motion models are defined. In the next chapter, we will define a similarity measure between two images that will allow an accurate estimation of the considered motion parameters.

Chapter 2

Background in information theory

One main idea of this thesis is to create robust algorithms for visual tracking and visual servoing. As we will see, both domains are interconnected since both can be performed as a registration task by combining an alignment function between two images and a non-linear optimization process. Whereas the optimization approach is specific to the application, a common definition of the alignment function is possible. This alignment function is nothing but a similarity measure between two images. Since the robustness of the tasks, *i.e.* its capability to deal with any conditions such as occlusions or illumination variations, is mainly depending on the robustness of the alignment function, the goal will be to choose an adapted similarity measure.

This chapter provides the basics in information theory that are required to define mutual information, the similarity measure on which this work is based. We present the original definition of mutual information and how to adapt it to our computer vision problem as an alignment function. Finally we evaluate the robustness of the resulting function and compare it with other well-known measures such as Sum of Squared Differences (SSD) or Zero-mean Normalized Cross Correlation (ZNCC).

2.1 Statistics basics

This section provides some elements of statistics that are required to understand the information theory. Since our final goal is to work with discrete variables (digital images), we limit this introduction to discrete random variables and their properties. Once the definition of mutual information is given, no more statistics are used. The readers that already have the necessary knowledge in statistics can directly refer to the next section where the definition of the alignment function is given. Since our work associates both information theory and computer vision, we chose the notations in order to best respect the standards of each field.

2.1.1 Random variables

Every gambler is indirectly studying statistics. When he throws a dice or when he draws a card, he probably does not know that random variables could model it. But random variables are not only hiding in casinos, they are also useful to model many physical entities. Indeed, a random variable (RV) is a variable whose value is unpredictable.

What is interesting is that, although the value of a RV is unpredictable, statistical laws provide some properties of the RV that help us to know its most likely value. Let X be a random variable and x one of its values. If we consider the RV X as the resulting number of the throw of a six-sided dice, then one obvious fact is that the possible values of x are in the set of values $\Omega_X = \{1, 2, 3, 4, 5, 6\}$. For simplicity reasons, let us consider that all our random variables can only have bound positive integer values, so that $\Omega_X \subset \mathbb{N}$. This condition is typically satisfyed if we consider the pixel intensities of a gray-level image: its values can be any of the 256 possible gray-levels, so that $\Omega_X = \{0, ..., 255\}$.

2.1.2 Probability distribution

The probability distribution $p_X(x) = \Pr(X = x)$ is the proportion of times the variable X should take the value x. For instance, considering the case of a fair dice, one could expect that each value has an equal probability to occur one time every six throws. The probability distribution is then called uniform, since it is constant on Ω_X (see Figure 2.1(a)).

Two axioms of the probability distribution are the following: first, the probability distribution function has a compact support. If a value is not in the set of possible values Ω_X, then its probability is not defined[1]. And, since the value x is always in the set Ω_X, the sum of the probabilities of the values over the set is one:

$$\sum_{x\in\Omega_X} p_X(x) = 1 \tag{2.1}$$

Considering these two properties, a probability distribution function stands in a wide range of functions. Simply imagine that the six-sided dice is loaded, the probability distribution function has an infinity of possible definitions since you can load whichever part of the dice you want and the way you want. If for instance the 6^{th} side is loaded, we can expect a probability distribution function as in Figure 2.1(b). Face 6 has a larger occurrence probability than the other faces, and face 1, *i.e.* opposite to face 6, has the smallest probability.

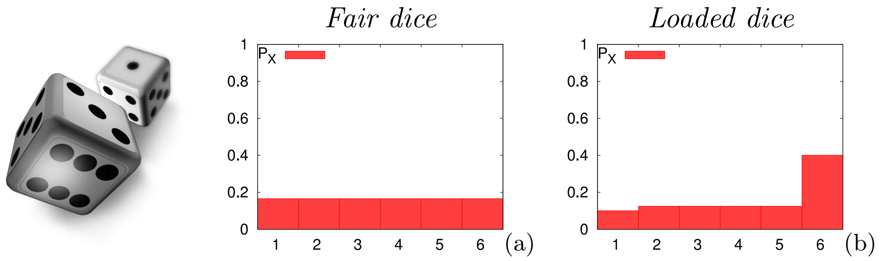


Figure 2.1: Probability distribution function of the throw of a six-sided dice.

2.1.3 Joint probability and conditional probability

Given two random variables X and Y, we can similarly express the probability $p_{XY}(x, y)$ of a couple of value (x, y) to occur, called joint probability.

$$p_{XY}(x, y) = \Pr(X = x \text{ and } Y = y) \tag{2.2}$$

Its properties are similar to the ones of the probability distribution function. The probability is not defined for each samples x and y out of the set of possible values Ω_X and Ω_Y. And the sum over the two sets of values equals one:

$$\sum_{x\in\Omega_X} \sum_{y\in\Omega_Y} p_{XY}(x, y) = 1. \tag{2.3}$$

To find the link between this joint probability and the probabilities of the two variables X and Y, two cases have to be considered. The simplest one is when the two variables X and Y are independent, *i.e.* knowing that $X = x$ has no effect on the probability of Y, and conversely, knowing that $Y = y$ has no effect on the probability of X. One simple example is to consider two throws of a dice: it is clear that knowing the result of the first throw has no effect on the

[1] Who would bet on the number 7 with a six-sided dice?

second throw. The joint probability or proportion of time the couple of values (x, y) should occur is then simply given by the product of the two probabilities of the variables X and Y as:

$$p_{XY}(x, y) = p_X(x)p_Y(y) \tag{2.4}$$

It is classical to refer to $p_X(x)$ and $p_Y(y)$ as the marginal probabilities to differentiate them from the joint probability $p_{XY}(x, y)$.

Let us now consider the example of two following draws from the same deck containing N cards. If all the cards are unique, then the probability $p_X(x)$ of one card to appear in the first draw is $1/N$. If we do not put back the first card in the deck, then, there are only $N - 1$ cards remaining. The probability of the remaining cards to appear in a second draw is then of $1/(N-1)$ and the probability of the card x to appear a second time is 0. This function is called conditional probability and is noted $\Pr(Y = y \mid X = x)$, the probability of a variable Y to have a value y knowing that $X = x$. The joint probability of (x, y) is then given by the following equations:

$$\begin{aligned} p_{XY}(x, y) &= \Pr(Y = y \mid X = x)p_X(x) \\ &= \Pr(X = x \mid Y = y)p_Y(y) \end{aligned} \tag{2.5}$$

If X and Y are independent, then $\Pr(Y = y \mid X = x) = \Pr(Y = y)$ and this equation is equivalent with the Equation (2.4). This probability distribution provides already a measure to compare the link between two variables. It is not sufficient to directly compare the correlation between these two variables but it gives the necessary tools to define the information theory and the mutual information.

2.2 Information theory

Information theory is one basis of computer science. Its first use in the mathematical theory of communication by Claude Shannon [Shannon 1948] is now generalized to every numerical systems, from internet to image processing problems. This section defines how, from the probability distribution functions of random variables, we can measure the quantity of information they contain or they share.

2.2.1 Entropy

Shannon defines the quantity of information of a variable as a measure of its variability. Indeed, if a variable is constant then it contains no information, and the more it is variable, the more we can consider that it is containing information. Shannon's entropy $\mathrm{H}(X)$ measures the variability, *i.e.* the quantity of information, of a discrete random variable X as:

$$\mathrm{H}(X) = -\sum_{x \in \Omega_X} p_X(x) \log_\alpha p_X(x). \tag{2.6}$$

with $0 \log_\alpha(0) = 0$. The formulation can be interpreted as follows: the higher $p_X(x)$, the more evident $X = x$, and the lower $p_X(x)$, the more uncertain $X = x$. At the same time, the higher $p_X(x)$, the lower $-\log_\alpha p_X(x)$, and the lower $p_X(x)$, the higher $-\log_\alpha p_X(x)$. Thus $-\log_\alpha p_X(x)$ is a measure of the uncertainty of the value x. $\mathrm{H}(X)$ is then a weighted mean of the uncertainties, which finally represents the variability of X.

The logarithm basis α is only changing the unit of the entropy. For instance, the logarithm basis is set to $\alpha = 2$ to compute the entropy of a digital signal in bits. This measure is typically related to the theoretical number of bits required for a noiseless compression of a signal [Shannon 1948]. In practice, the basis only changes the entropy value with a scale factor. Therefore, it has no interest in our context, since we only seek the maximum of a cost function and not a particular value. The only thing is to keep it constant. In the remainder of this thesis, the logarithm basis is set to $\alpha = 2$, but the units of the resulting values (in bits) will not be precised since they are not significant.

If the variable has a uniform probability distribution function, it contains a large amount of information and its entropy is high. On the contrary, if for instance, one value has a high probability, the variable is quasi constant, it contains a small amount of information and the entropy is low.

Example: the previous interpretations are easily illustrated in the case of a variable with two possible values. Figure 2.2(a) represents the uncertainty added by a value according to its probability. we can notice that the smaller the probability, the stronger the effect it has on the entropy value. The entropy has been computed with respect to the probability of the first event $P_X(x_1)$ [2]. As expected, Figure 2.2(b) shows that the entropy is maximal when the probability distribution is uniform (*i.e.* $P_X(x_1) = P_X(x_2) = 0.5$) and null when the variable is constant (ie $P_X(x_1) = 1$ or $P_X(x_2) = 1$).

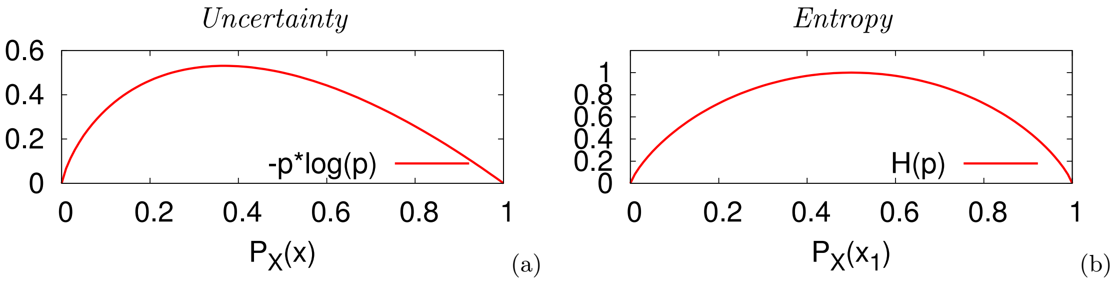


Figure 2.2: Uncertainty added by a value with respect to its probability (a) and entropy of a binary variable with respect to the probability of its first value x_1 (b).

Several other entropy measures have been proposed to solve some particular problems. We can cite for instance Rényi's entropy [Rényi 1956] that includes a new parameter to control the relative effect of the values depending on their probabilities. It is for example possible to limit the effect of the values that have a small probability. Nevertheless, in this thesis, we prefer to focus only on Shannon's entropy [Shannon 1948], since it does not require any additional parameter.

2.2.2 Joint entropy

Following the same principle, the joint entropy $\mathrm{H}(X,Y)$ of two random variables X and Y is defined as the variability of the couple of variables (X,Y). Shannon's joint entropy is computed as:

$$\mathrm{H}(X,Y) = -\sum_{x\in\Omega_X}\sum_{y\in\Omega_Y} p_{XY}(x,y)\log_\alpha p_{XY}(x,y) \tag{2.7}$$

This equation typically defines a weighted mean of the joint uncertainties $-\log p_{XY}(x,y)$. Similarly to the probabilities function, it is classical to refer to $\mathrm{H}(X)$ and $\mathrm{H}(Y)$ as the marginal entropies to differentiate them from the joint entropy $\mathrm{H}(X,Y)$. Since the joint entropy measures the global variability of the system of variables, it also provides some information about the correlation between the two variables. Indeed the global variability can be also interpreted as

[2] The second event probability is simply given by $P_X(x_2) = 1 - P_X(x_1)$.

the variability of X to which we add the new information provided by Y. At first sight, the joint entropy could be considered as a good similarity measure, since, if the variability of Y is mostly part of the variability of X (the variables are strongly correlated), the joint entropy is relatively small. Nevertheless, the joint entropy is also linked with the marginal entropies of the two variables and this dependence makes it not suited as a similarity measure.

Example: let us consider a similarity measure between two signals X and Y illustrated in Figure 2.3. If the two variables are equal then $p_{XY}(x, y) = p_X(x) = p_Y(y)$ and the joint entropy is minimal with $\mathrm{H}(X, Y) = \mathrm{H}(X) = \mathrm{H}(Y)$. If now we consider that the variable X is constant, the uncertainty of the system is then the uncertainty of the variable Y. The entropy is therefore also minimal with $\mathrm{H}(X, Y) = \mathrm{H}(Y)$, despite the fact that X and Y are not similar. That shows that joint entropy is not a good similarity measure.

To overcome this dependency with respect to the variable entropies, the conditional entropy of the variable Y given knowledge of X is considered. The definition of the conditional entropy is the uncertainty of the system of variables (X, Y) when X is known. Its expression is given by:

$$\mathrm{H}(Y \mid X) = \mathrm{H}(X, Y) - \mathrm{H}(X) \tag{2.8}$$

If the two variables are completely correlated with respect to each other, the conditional entropy is null. If the two variables are independent: $\mathrm{H}(Y \mid X) = \mathrm{H}(Y)$, a dependence with the entropy of Y is remaining. Thus it can still not be used as a similarity measure.

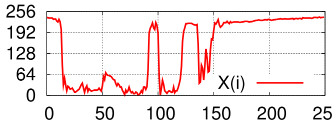
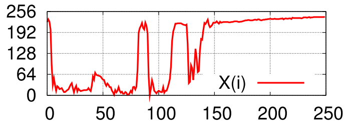
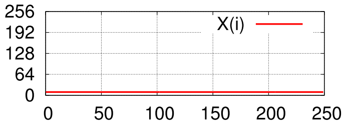
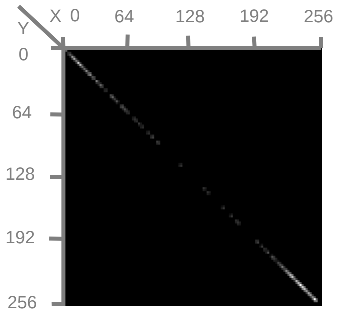
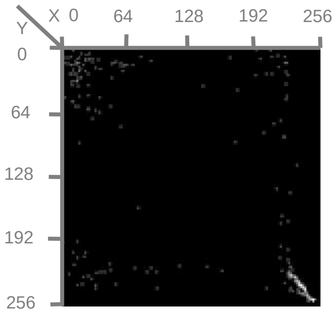
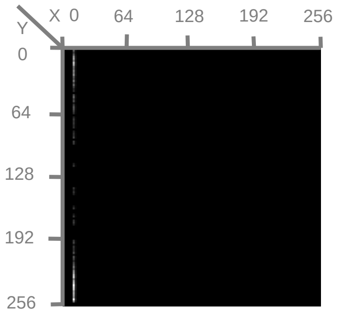
	$X = Y$	$X(i) = Y(i + 10)$	$X = \text{Constant}$
Variables			
Joint probability			
H	4.62	5.56	3.98
MI	3.14	2.19	$4.44e - 16$

Figure 2.3: The joint entropy is a dispersion measure of joint probability. It is not adapted to measure the alignment between two variables since if one RV is constant then the dispersion is small (gray cell). This problem is solved when using the mutual information that is maximal for the perfect alignment $X = Y$ (gray cell).

2.2.3 Mutual information

Mutual information has been proposed to define a similarity measure that solves the above mentioned problem. A solution to remove the dependency of conditional entropy with respect to the entropy of Y is to consider the difference between these two values. This formulation yields to the following definition of mutual information:

$$\begin{aligned} \mathrm{MI}(X,Y) &= \mathrm{H}(Y) - \mathrm{H}(Y \mid X) \\ &= \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y) \end{aligned} \tag{2.9}$$

This expression defines the difference of variability between the variable Y and the variable Y knowing X. The more dependent the two variables, the larger this difference is. Thus this function simply defines the amount of information that is shared by the two variables.

If mutual information is computed using the previous examples presented in Figure 2.3, we see that it is not depending on the marginal entropies of the variables as joint entropy was. If the two variables are equal then mutual information is maximal. If one of the variables is constant then it shares no information with the other variable, so mutual information is null.

2.3 Application to image processing

As we previously stated, the goal is to use mutual information to compare images. The previous definition of mutual information is generic and was defined to be used in the theory of communication. In this section we show the limits of this original formulation in the image processing problem and review the most significant work that have been performed to adapt mutual information to our problem.

2.3.1 Information theory on digital images

The two previous variables X and Y are now referring to the image intensities, so let us change the notations and use instead I and I^*. The pixel intensities of both images I and I^* are respectively noted i and j. Since we use classical 256 gray-level images, the possible values of i and j are $\Omega_I = \Omega_{I^*} = [0, 255] \subset \mathbb{N}$. The probability $p_I(i)$ of a gray-level value i is the proportion of occurrence of value i in the image I. This is classically estimated using the histogram of the image normalized by the total number of pixels:

$$p_I(i) = \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \delta\left(i - I(\mathbf{x})\right) \tag{2.10}$$

where $\mathbf{x}$ are the pixel positions of the image and $N_{\mathbf{x}}$ is the number of pixels. In the classical formulation of the histogram computation, each time $I(\mathbf{x}) = i$ the i^{th} histogram bin value is incremented. This is typically performed using a Kronecker's function defined as:

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

The entropy, that depicts the variability of the image, can be simply considered as a measure of the dispersion of the image histogram. Using the probability distribution function of the image, the entropy expression becomes:

$$\mathrm{H}(I) = -\sum_{i \in \Omega_I} p_I(i) \log p_I(i). \tag{2.12}$$

As for the probability $p_I(i)$, the joint probability $p_{II^*}(i,j)$ of the couple (i,j) is the proportion of occurrence of the couple of values (i,j) in the couple of images (I, I^*). Its computation is thus simply obtained by computing the joint histogram of the two images normalized by the number of pixels $N_{\mathbf{x}}$:

$$p_{II^*}(i,j) \;=\; \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \delta\left(i - I(\mathbf{x})\right) \delta\left(j - I^*(\mathbf{x})\right) \tag{2.13}$$

The expression of the joint entropy is obtained from equation (2.7) and gives:

$$\mathrm{H}(I, I^*) = -\sum_{i\in\Omega_I} \sum_{j\in\Omega_{I^*}} p_{II^*}(i,j) \log p_{II^*}(i,j). \tag{2.14}$$

Developing the mutual information function using the equations (2.12) and (2.14) yields the following expression:

$$\begin{aligned}
\mathrm{MI}(I, I^*) &= \mathrm{H}(I) + \mathrm{H}(I^*) - \mathrm{H}(I, I^*) \\
&= -\sum_i p_I(i) \log p_I(i) - \sum_j p_{I^*}(j) \log p_{I^*}(j) + \sum_{i,j} p_{II^*}(i,j) \log p_{II^*}(i,j) \\
&= \sum_{i,j} -p_{II^*}(i,j) \log p_I(i) - p_{II^*}(i,j) \log p_{I^*}(j) + p_{II^*}(i,j) \log p_{II^*}(i,j) \\
&= \sum_{i,j} p_{II^*}(i,j) \log\left(\frac{p_{II^*}(i,j)}{p_I(i) p_{I^*}(j)}\right) \qquad (2.15)
\end{aligned}$$

where the set of possible values has been omitted for purpose of clarity.

2.3.1.1 Experimental evaluation of mutual information

In this section, we propose a process to evaluate the quality of mutual information as an alignment function. A first image I^* is considered and a set of images I are created that represents the same scene than I^*, as the difference that they are warped using the horizontal and vertical translations[3] (see section 1.2.2 page 15). The translational motions are applied following a regular grid of positions centered on the ground truth position of I^*. Since MI is maximal when the two images are aligned, the goal is typically to have an alignment function that decreases when the similarity between the images decreases, *i.e.* when the error of alignment increases.

To create the images I^* and I, we use a virtual environment that allows for the control of the external conditions of acquisition. To create the translational motion between the images we use the environment presented in Figure 2.4: a camera is oriented perpendicularly to a planar scene and we move the camera with a translation parallel to the scene (perpendicular to its focal axis). We note $\mathbf{\Delta r} = (t_x, t_y)$ the horizontal and vertical displacement between the positions of acquisition of I^* and I.

The values of MI have been computed with respect to $\mathbf{\Delta r}$ and represented in Figure 2.4. The global maximum of the cost function corresponds to the correct alignment position $\mathbf{\Delta r} = (0,0) = \mathbf{0}$. Mutual information provides a correct measure of the dependence between the information of the two images. Nevertheless we can observe that the global maximum is very sharp and that the function shape shows many local maxima. Therefore, once the alignment error is large, it is impossible to judge the quality of the alignment using this original formulation of MI. To improve the definition of MI, some approaches have been proposed to smooth the function and remove this local maxima that are interpolation artifacts.

[3] We limit our alignment task to two DOF (two parameters) for a purpose of clarity, since the results using more than two DOF would be difficult to illustrate.

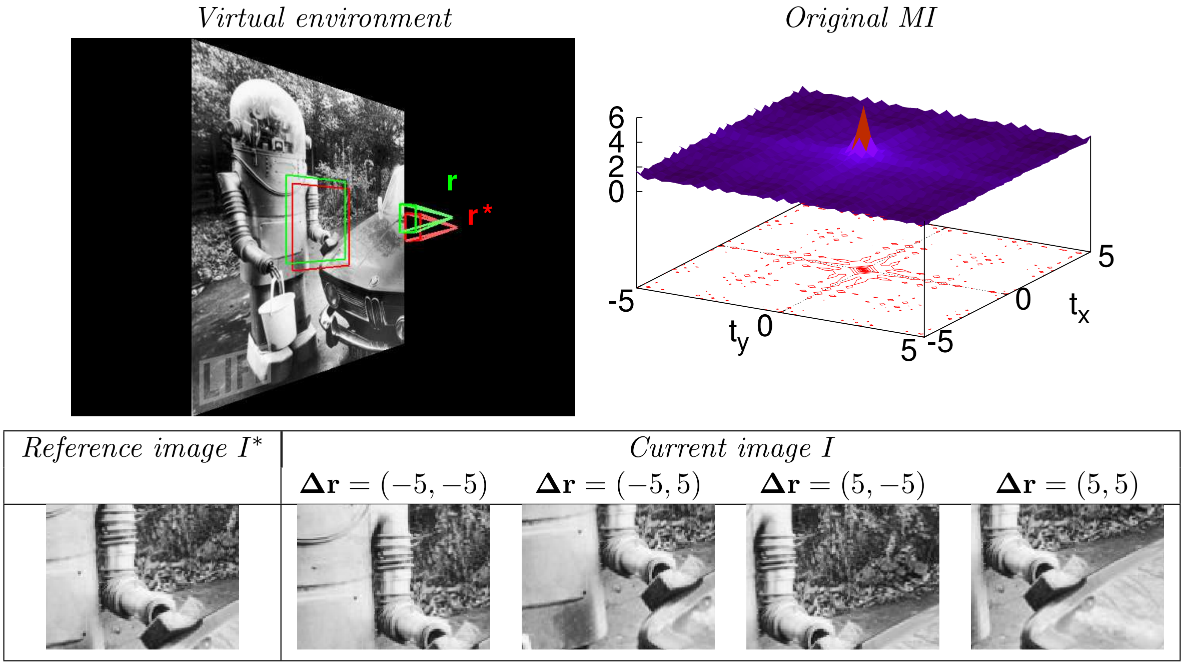


Figure 2.4: Mutual information between two images with respect to the translation between them. $\mathbf{\Delta r}$ is the translation between the current and desired camera poses expressed in centimeters.

2.3.1.2 Interpolation artifacts

To explain the formation of the interpolation artifacts, we simply use a short example. For a more detailed definition of them, the reader can refer to [Pluim 1999]. Let us consider a very small translation, if we focus on an edge of the image then the resulting changes in intensity will probably yield to the apparition of new pixel intensities that were not present in the original image. Thus a small translation can change several bins of the histograms from a null value to a small value (as well as changing all the other ones).

The definition of entropy that was previously given in 2.2.1 precised that, when the probability of an event is small, then the variation of entropy due to its variation is large. One can therefore consider that a small translation will have a strong effect on the marginal and joint entropies as well as on mutual information, and produce the interpolation artifacts.

The apparition of new intensities is not only due to the displacement of the camera. New intensities can also be due to noise in the image acquisition process. Such noise also strongly affects the value of mutual information. Therefore, the effects of new pixel intensities have to be avoided to smooth and robustify the cost function.

2.3.2 Smoothing MI

The previous section shows that mutual information provides a measure of the correlation between two images that suffers from interpolation artifacts and has a sharp maximum. To smooth and robustify the mutual information, the solution is simply to remove the empty bins from the histograms. To our knowledge, two methods can then be applied [Tsao 2003]: binning the histograms and working on the interpolation of the images.

2.3.2.1 Histogram binning and partial volume interpolation

The first observation that can be made is that the larger the number of histogram bins, the more bins will be empty. It is therefore not surprising that an histogram with 256 entries has a large number of empty bins. Moreover, such large histograms are very expensive to build in terms of both memory and time.

Starting from these observations, a first solution is to decrease the number of histogram bins. In the original formulation proposed in equation (2.10), the number of bins was originally equal to the number of possible gray-level intensities $N_{c_I} = 256$ of the image pixels. This number is typically linked with the maximum gray-level value $I_{max} = N_{c_I} - 1$. To reduce the number of bins, we simply change I_{max} by scaling the image intensities as follows:

$$\overline{I^*}(\mathbf{x}) = I^*(\mathbf{x}) \frac{N_c - 1}{N_{c_{I^*}} - 1} \tag{2.16}$$

where N_c is the new number of histogram bins. The resulting intensities are no longer integer values. Using a Kronecker's function δ, as it was previously defined in equation (2.10), will require to keep only the integer part of the intensity and thus lose a large part of the information contained in the images. This function has then to be modified to keep the information of these real values. Several solutions have been proposed to simultaneously smooth the mutual information function and keep its accuracy [Maes 1997, Viola 1997]. In [Viola 1997] the histogram is computed using Gaussian functions, while, in [Maes 1997], an approximation of these Gaussian functions are performed using B-spline functions. The advantages of the B-splines come from the simple computation of their values and derivatives. Since these properties are required in our context, our approach is based on the use of centered cardinal B-spline functions. Classical B-spline functions and their properties have been represented in Frame 2, where B_n refers to the B-spline of order n. We can see that, the more the order of the B-spline is high, the more it is smooth, differentiable and the approximation of the Gaussian function is good. Nevertheless, as the order increases, the complexity of its computation also increases.

To respect the differentiability conditions that will be explained in the next section we chose to use the cubic B-spline functions B_4. For a purpose of clarity, let us note ϕ these B-splines. The final analytical formulation of the normalized histogram becomes:

$$p_I(i) \quad = \quad \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(\mathbf{x})\right) \tag{2.17}$$

where the possible gray-level values are now $\overline{I}(\mathbf{x}) \in [0, N_c - 1]$.

Remark: Since cubic B-splines are used, N_c does not correspond anymore to the number of histogram bins. Indeed, let us consider equation (2.17): if a pixel has an intensity $\overline{I}(\mathbf{x}) = 0$, the bins $i = \{-1, 0, 1\}$ are incremented. Similarly if $\overline{I}(\mathbf{x}) = N_c - 1$ then the bins $i = \{N_c - 2, N_c - 1, N_c\}$ are incremented. Therefore, the cubic B-spline interpolation has side-effects that add two bins (for $i = -1$ and $i = N_c$) to the marginal histograms.

The probability distribution function of I^* and the joint probability of (I, I^*) are modified using the same approach that yields to:

$$p_{I^*}(j) \quad = \quad \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(j - \overline{I^*}(\mathbf{x})\right) \tag{2.18}$$

$$p_{II^*}(i,j) \quad = \quad \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(\mathbf{x})\right) \phi\left(j - \overline{I^*}(\mathbf{x})\right) \tag{2.19}$$

Frame 2 Centered cardinal B-Splines.

B-Splines are very usefull functions for interpolation problems. Let us note B_n the centered cardinal B-Spline of order n. The general definition of the functions B_n can be expressed using the recursive expression:

$$B_n(i) = (B_{n-1} * B_1)(i) \qquad \text{with:} \qquad B_1(i) = \begin{cases} 1 & \text{if } i \in [-0.5, -0.5[\\ 0 & \text{otherwise} \end{cases}$$

where $*$ denotes the convolution operation. This function shows many interesting properties that are well suited in our problem:

- their unit summation:

$$\sum_{m=-\infty}^{+\infty} B_n(m+i) = 1 \qquad \forall (i,m) \in \mathbb{R} \times \mathbb{Z}$$

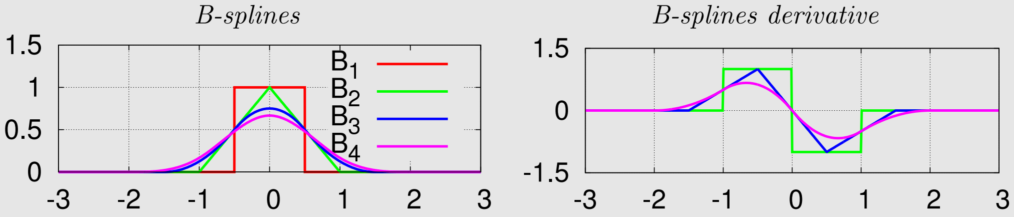
- B_n is $n-1$ times differentiable function. The derivatives are easily computed using the B-splines of lower order:

$$\frac{\partial B_n(i)}{\partial i} = B_{n-1}\left(i + \frac{1}{2}\right) - B_{n-1}\left(i - \frac{1}{2}\right)$$

For instance, the B-splines from order 2 to 4 are defined by (see the bottom figure):

$$B_2(i) = \begin{cases} 1+i & \text{if } i \in [-1, 0[\\ 1-i & \text{if } i \in [0, 1[\\ 0 & \text{otherwise} \end{cases} \quad B_3(i) = \begin{cases} (1.5+i)^2/2 & \text{if } i \in [-1.5, -0.5[\\ 1+i-(0.5+i)^2 & \text{if } i \in [-0.5, 0[\\ 1-i-(0.5-i)^2 & \text{if } i \in [0, 0.5[\\ (1.5-i)^2/2 & \text{if } i \in [0.5, 1.5[\\ 0 & \text{otherwise} \end{cases}$$

$$B_4(i) = \begin{cases} ((2+i)^3)/6 & \text{if } i \in [-2, -1[\\ (1+3(1+i)+3(1+i)^2-3(1+i)^3)/6 & \text{if } i \in [-1, 0[\\ (1+3(1-i)+3(1-i)^2-3(1-i)^3)/6 & \text{if } i \in [0, 1[\\ ((2-i)^3)/6 & \text{if } i \in [1, 2[\\ 0 & \text{otherwise} \end{cases}$$



Several solutions have been proposed to estimate an optimal number of histogram bins using for instance Sturges' rule [Sturges 1926] or Scott's rule [Scott 1979]. Nevertheless, a constant number of bins, for instance $N_c = 8$, which keeps a small value and avoids losing too much information, has always given satisfying results in our experiments.

If we compare the mutual information values between the basic formulation and the new one, the benefits of histogram binning are obvious. As Figure 2.5 shows, the mutual information function is not subject to interpolation artifacts anymore. Nevertheless, MI keeps a very sharp maximum.

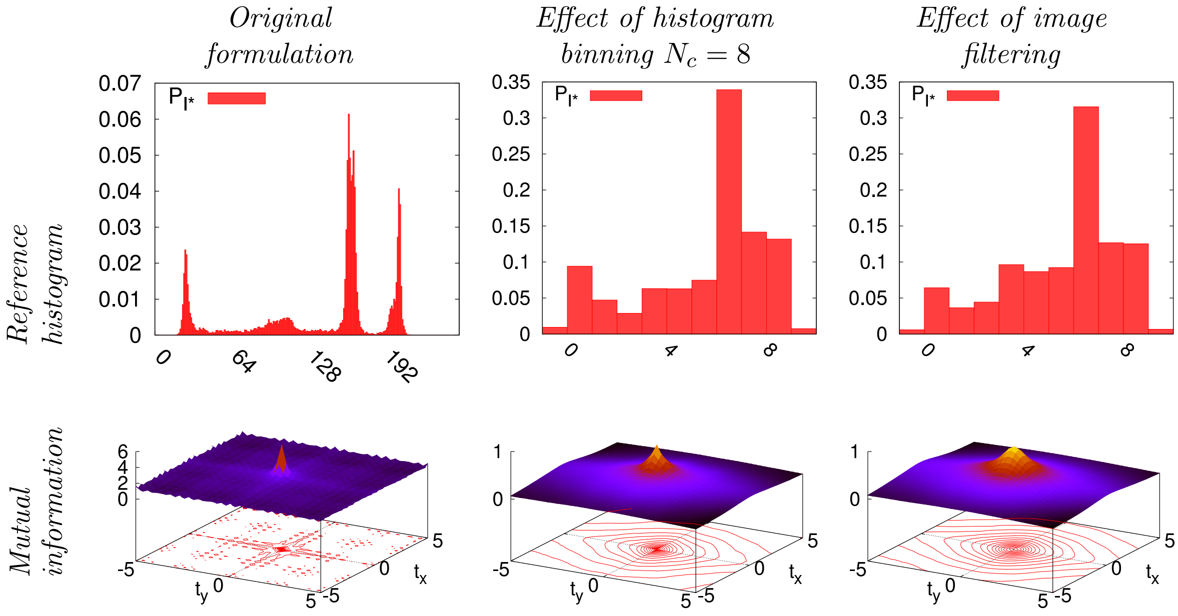


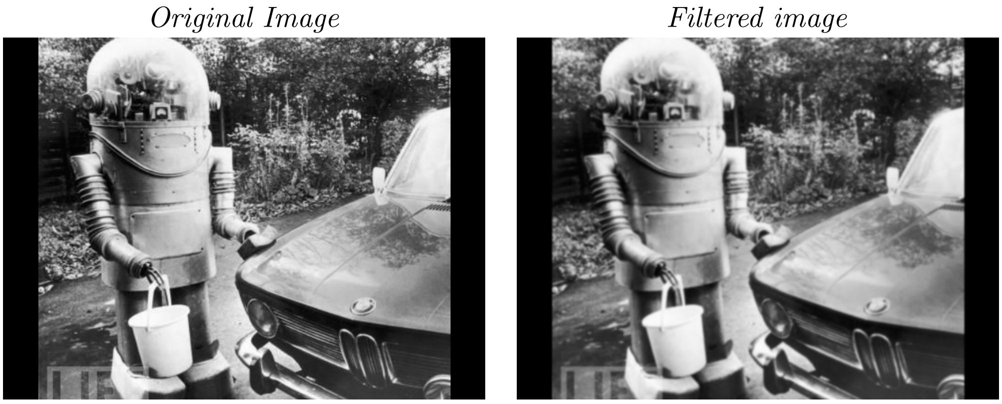
Figure 2.5: Smoothing mutual information. The input images are the ones presented in Figure 2.4.

2.3.2.2 Image interpolation or image smoothing?

The sharpness of the maximum is due to the quality of the alignment measure. Nevertheless, we will see in the next chapters that a too sharp maximum is not adapted for most of the optimization methods. To reduce the sharpness of the maximum, a simple method is to reduce the quality of the alignment at the maximum by "smoothing the information".

A way to smooth the information in the image is to use a more complex interpolation of the image or simply to filter the image [Tsao 2003]. In most of our work, we decided to filter the input images that is the less time consuming operation. The chosen filter is a 5×5 Gaussian filter (see Figure 2.6). This filter smoothes the information in the sens that the intensity of a pixel is smoothed using the intensities of its neighbors. For instance the edges that were previously really strong and defining a precise information are smoother after the image filtering, but they keep the information were it is.

As we show in Figure 2.5, the use of both histogram binning and image filtering with a 5×5 Gaussian filter makes the mutual information function's shape perfectly smooth and accurate. And we will see that these properties are necessary to solve the tracking and visual servoing problems that will be presented in the next chapters.


Original Image

Filtered image

Figure 2.6: Smoothing the input images: original and filtered image. The filtered image has smoother edges.

2.3.3 Comparison with classical alignment functions

MI is known as one of the more robust alignment functions [Pluim 1999]. To validate its robustness and accuracy compared to other well known appearance-based alignment functions, we perform several experiments similar to the one presented in Figure 2.4 with, this time, appearance variations. The alignment functions we consider are the ones that follow:

- The sum of the squared difference (SSD) of each pixel intensities that is the alignment function of the well known KLT algorithm [Tomasi 1991, Hager 1998a]:

$$SSD(I, I^*) \quad = \quad \sum_{\mathbf{x}} \left(I(\mathbf{x}) - I^*(\mathbf{x})\right)^2$$

- the Kernel density alignment function that is typically a distance between the weighted histograms of both images that can be computed either using a Batthacharrya distance as in [Comaniciu 2003] or a simple Matuzita distance [Kallem 2007] between the histograms:

$$\rho(I, I^*) \quad = \quad \sum_{i} \left(\sqrt{p_I(i)} - \sqrt{p_{I^*}(i)}\right)^2.$$

  This is typically the alignment function that is considered in the mean shift optimization. Here a Gaussian Kernel K centered around $\mathbf{x}_0$ has been used so that the histograms are computed with:

$$p_I(i) \quad = \quad C^{-1} \sum_{\mathbf{x}} K(\mathbf{x}) \phi(\overline{I}(\mathbf{x}) - i)$$

  with $K(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{x}_0\|)$ and $C = \sum_{\mathbf{x}} K(\mathbf{x})$. The number of bins used here are the same as the one in the MI computation. The formulation of the histograms remains therefore very similar to those computed in the MI. The only difference comes from the fact that they are weighted by the Kernel K;

- the zero-mean normalized cross correlation or ZNCC (or NCC in [Irani 1992]) defined as:

$$ZNCC(I, I^*) \quad = \quad \frac{\sum_{\mathbf{x}} \left(I(\mathbf{x}) - \widehat{I}\right) \left(I^*(\mathbf{x}) - \widehat{I^*}\right)}{\sigma_I \sigma_{I^*}}$$

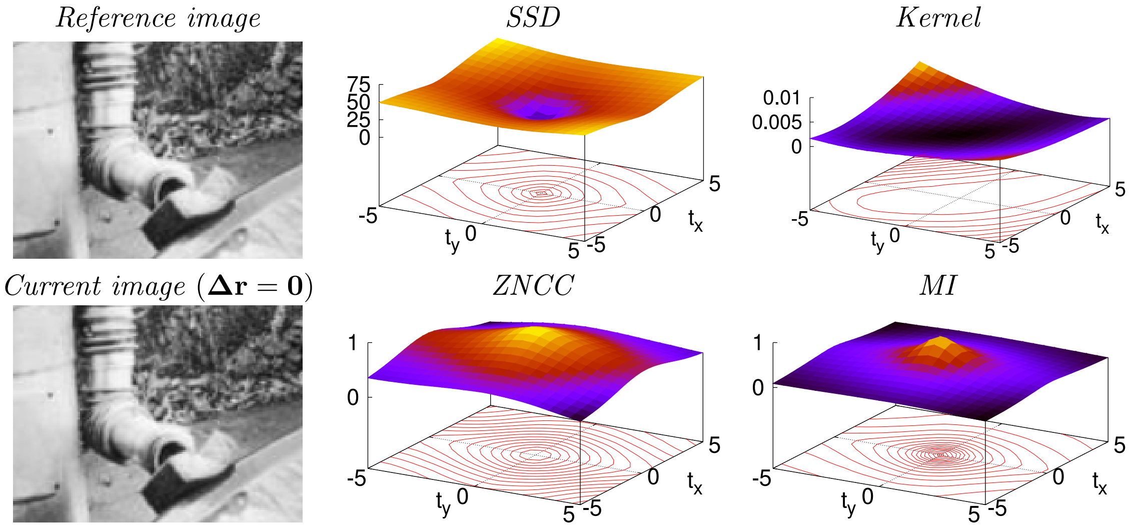
  where $\widehat{I}$ and $\widehat{I^*}$ are respectively the average intensities of the images I and I^*: and σ_I and σ_{I^*} are the standard deviations of the two images.

First, we focus on the alignment task in nominal conditions, then we focus on some variations that are commonly met in the usual image sequences or in the visual servoing experiments: white noise, occlusions and illumination variations. Finally, we also evaluate the robustness of these different approaches with respect to non-linear variations in the image intensity by comparing images acquired using several modalities.

2.3.3.1 Nominal conditions

The first step to evaluate the alignment function is to consider them in the nominal conditions. The evaluation that we propose is based on the same approach that we used in the previous section (see Figure 2.4 page 28) to analyze the mutual information function.

Figure 2.7 shows the values of the alignment measures with respect to the translations between the reference and current images. As expected, the four measures have their optimum located at the correct alignment position with a pose error $\mathbf{\Delta r} = \mathbf{0}$. The SSD and the Kernel approach, which are both dissimilarity functions, have a minimum at the alignment position, whereas the ZNCC and MI functions, which are similarity functions, present a maximum. The four measures are very smooth and present no local optimum in the domain of interest. They are therefore all suited for optimization problems, as a consequence they can all be used for tracking applications in nominal conditions.


Reference image

SSD

Kernel

Current image ($\mathbf{\Delta r} = \mathbf{0}$)

ZNCC

MI

Figure 2.7: Evaluation of the alignment measures in nominal conditions (the reference and current image at the alignment position are identical).

2.3.3.2 Robustness with respect to noise

Image noise is a random change in the intensity of the pixels. It can be due to the sensors of the camera as well as to the communication or coding process of the images. In any case, noise changes the images from their ideal form and, thus, can affect the performance of the alignment measures.

Figure 2.8 shows the values of the alignment functions with respect to the translations of the camera when the reference and current images are subject to a Gaussian white noise with a covariance of 5. We can see that all the functions are really robust to white noise. The optima of the functions are always correctly located at the alignment position $\mathbf{\Delta r} = \mathbf{0}$ and there is still no local optimum.

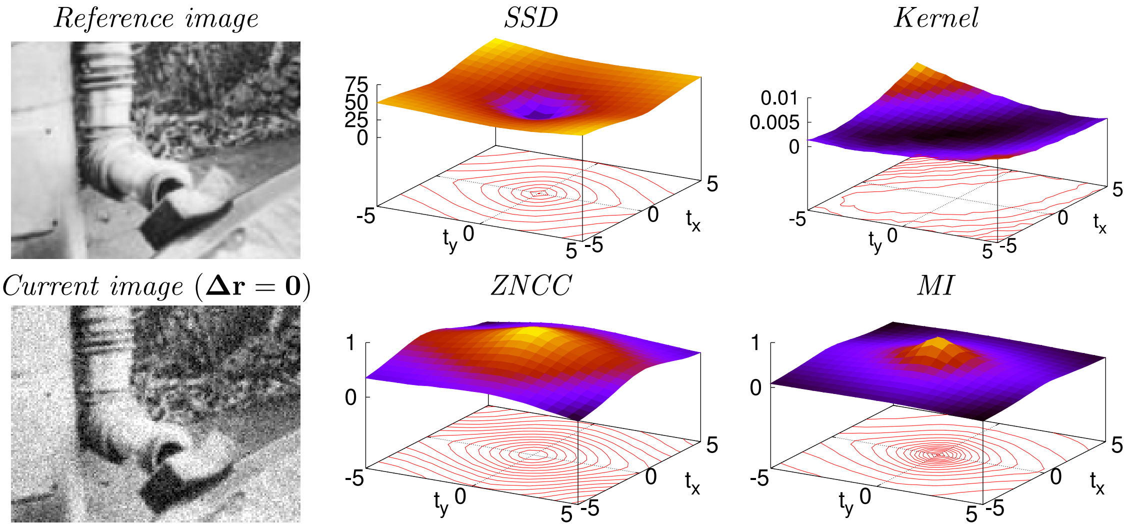


Figure 2.8: Robustness of the cost functions with respect to white noise. The white noise is visible on the current image at the alignment position.

2.3.3.3 Robustness with respect to occlusions

In image sequences, there can be objects moving in the foreground. These objects can thus move in front of our region of interest and temporarily modify the appearance of the considered scene. These modifications can have a strong impact on the shape of the alignment functions since they are built on the appearance of the scene. Figure 2.9 shows the results of an experiment where the scene has been modified using such occlusions.

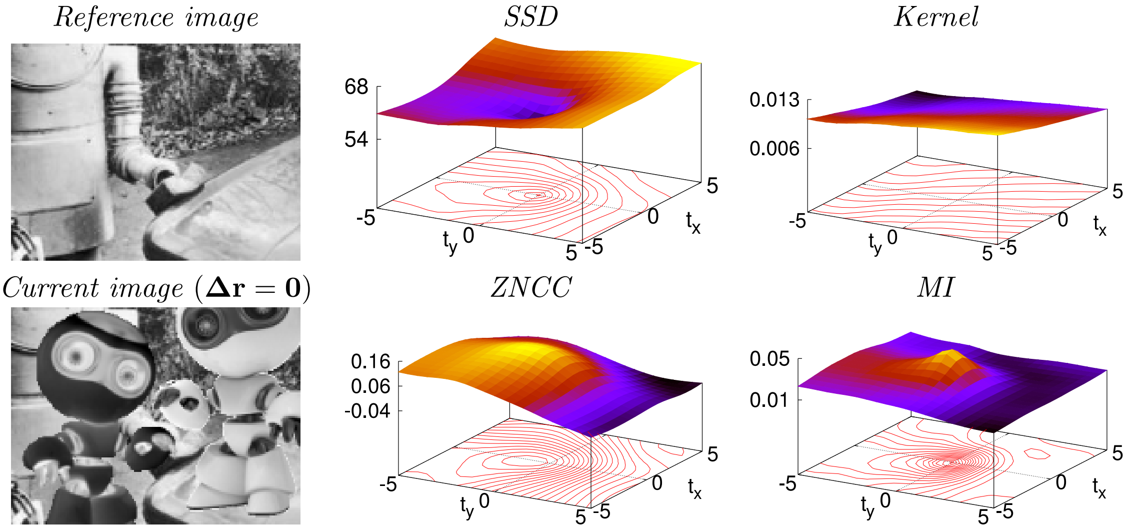


Figure 2.9: Robustness of the cost functions with respect to occlusions.

This experiment, that corresponds to moderately large occlusions, shows that the Kernel based function is not robust to occlusions. There is no optimum in the searched window. Indeed, the occlusions are completely changing the histogram of the scene at the desired position and thus the distance between histograms cannot provide satisfying results.

The ZNCC function has a very large maximum near the alignment position, nevertheless the accuracy suffers from the occlusion. The estimated pose of the camera is located 1 cm away from the ground truth alignment pose.

Surprisingly, despite the large difference of appearance, the difference between the images (SSD) keeps its minimum at the correct alignment pose. Nevertheless, we see that the occlusions have strongly modified the SSD cost function shape and that a valley begins to appear near the optimum.

As for mutual information, since there is still some information shared by the two images, and since the occlusions do not share information with the desired image at any position, the mutual information remains robust to the occlusions with an accurate maximum at the ground truth alignment position.

To show that the results obtained with MI can be very surprising, we also evaluate the alignment measures in the case of a very large occlusion in Figure 2.10. Despite the large occlusions, we see that there is still enough information shared between the two images to align them. Moreover, we can observe that there is still no local maxima in the searched window. This time the SSD and ZNCC functions show their limits: both have no optimum at the alignment position.

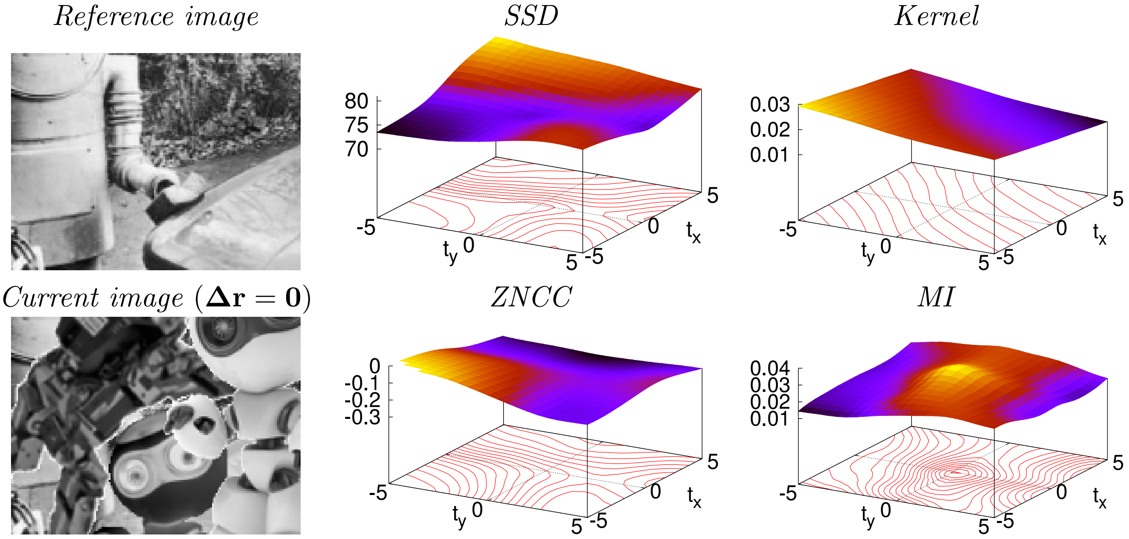


Figure 2.10: Robustness of the cost functions with respect to very large occlusions.

2.3.3.4 Robustness with respect to illumination variations

Illumination variations can have many forms. Nevertheless, independently of their nature, they can either modify the appearance of the whole scene, called global illumination variation, or modify only a part of the scene, called local illumination variations [Phong 1975, Blinn 1977]. Since the alignment measures are global appearance measures, we will see that the effects of global and local variations are very different.

Global illumination variations:

Global illuminations affect every pixel of the image in a consistent way. Many variations can be applied to the pixels, depending on the surface of the object, its orientation and the nature of the illumination (light source). Nevertheless, we limit our study to variations that affect the intensities with a simple affine relationship. This assumption remains a good approximation of the existing global illumination variations [Blinn 1977].

We compute the cost function using the same reference image and a current image acquired under a more intense light. The results are presented in Figure 2.11. MI is robust and keeps exactly the same shape as in the nominal case (or in the noise case). ZNCC, which is by definition robust to linear relationships between the intensities of the images pixels keeps also the same shape and thus gives the correct alignment position.

All the pixel intensities of the current image are modified compared to the reference image so that SSD, which is based on the pixel intensity differences, is also modified. Nevertheless, the modification is mainly a shift of the values of SSD. Indeed, the optimum of the SSD function corresponding to the position where the smallest (respectively the highest) intensities of the current image are matched with the smallest (respectively the highest) intensities of the reference image, and this is typically enough to be robust with respect to a linear change of the pixel intensities.

The only alignment function that suffers from the global illumination variation is the Kernel density function. Indeed, the shift in the histogram of the current image causes the distance between the histograms to provide no information about the alignment position.

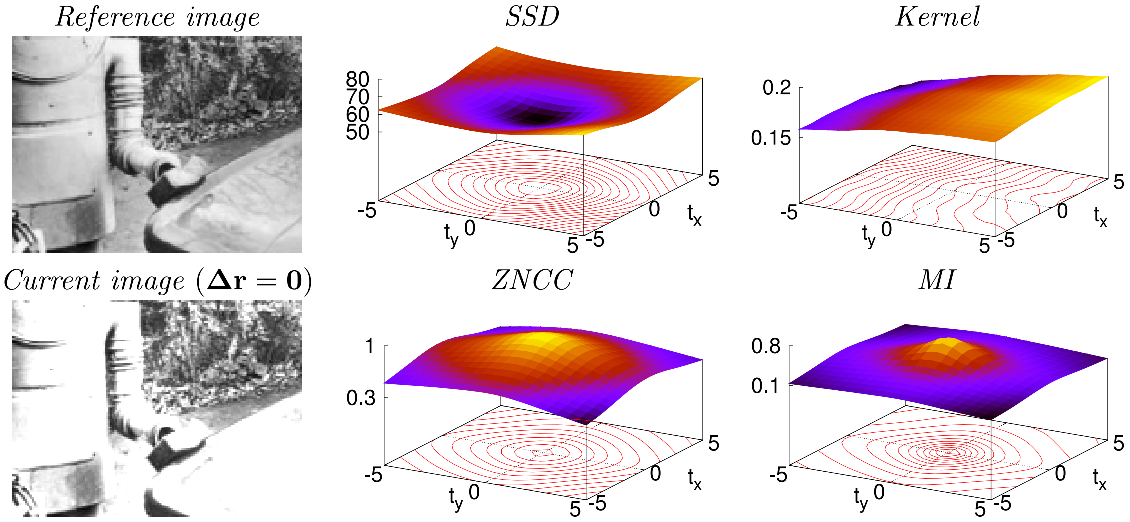


Figure 2.11: Robustness of the cost functions with respect to global illumination variations.

Local illumination variations:

Local illumination variations partially change the illumination of the scene. They appear in two major conditions: when surrounding objects partially increase the illumination (as a focused light would do) or decrease the illumination (as an object occluding the light directed to the scene) or when the object is non Lambertian, which does not diffuse an isotropic light (same light in all directions).

For instance, let us consider a laptop screen: some light can be reflected on its surface, the resulting brightness of the object will thus be strongly increased on a narrow spot. These modifications in the appearance of the object are more difficult to manage than the global variations since they are difficily modeled. The only way to predict it is to know the position of the light, the position and the surface properties of the object and the position of the observer.

We evaluate the robustness of the cost functions with respect to local illumination variations using the example in Figure 2.12. For the same reasons than for global illumination variations, the Kernel approach is not adapted to match the two images. This time, the relationship between the pixel intensities is different from one part of the image to another part, so the minimum of SSD is affected by the illumination variation and gives a wrong estimation of the correct alignment position.

Since the relation between the pixels intensities is different from one part of the image to another, ZNCC and MI are affected by the variation. The modification is visible on the shapes of both cost function, nevertheless the current and learned images still share a lot of common information, so the maxima are still located on the correct alignment position. ZNCC and MI can therefore be considered as strongly robust to global and local illumination variations.

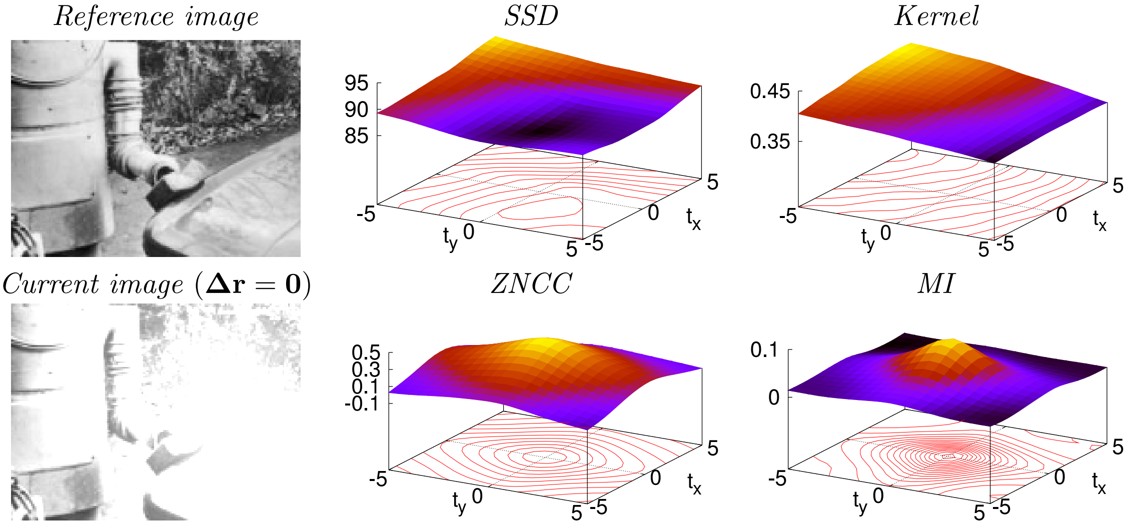


Figure 2.12: Robustness of the cost functions with respect to global illumination variations.

2.3.3.5 Robustness with respect to multimodality

Researches on the use of mutual information in computer vision is highly focused on its use in the medical field to align images acquired under different modalities. Indeed, in the medical field, the sensors are not limited to the classical cameras that we usually consider in robotics. Scanners of many kind are used and the resulting images have to be matched from one system to another. This multimodal alignment task is nevertheless not limited to the medical field and we show in this thesis that other applications are also possible.

Multimodal medical images:

To illustrate the performance of the alignment measures in the medical domain, we use two magnetic resonance images. The ground truth translation between the two images is known. As Figure 2.13 shows, the two images represent the same part of a head. Nevertheless, it is clear that no linear relationship can express the pixel intensities of one image with respect to the intensities of the other. The use of the mutual information to compare the two images is thus perfectly adapted.

As expected, the Kernel, SSD and ZNCC approaches are not able to measure the alignment between the two multimodal images. On the contrary, MI remains robust and allows for the estimation of the correct alignment position. Near the alignment position, the more the images are correctly aligned, the higher the mutual information is. Far from the alignment position, the cost function is quasi planar since the two images do not share information anymore.

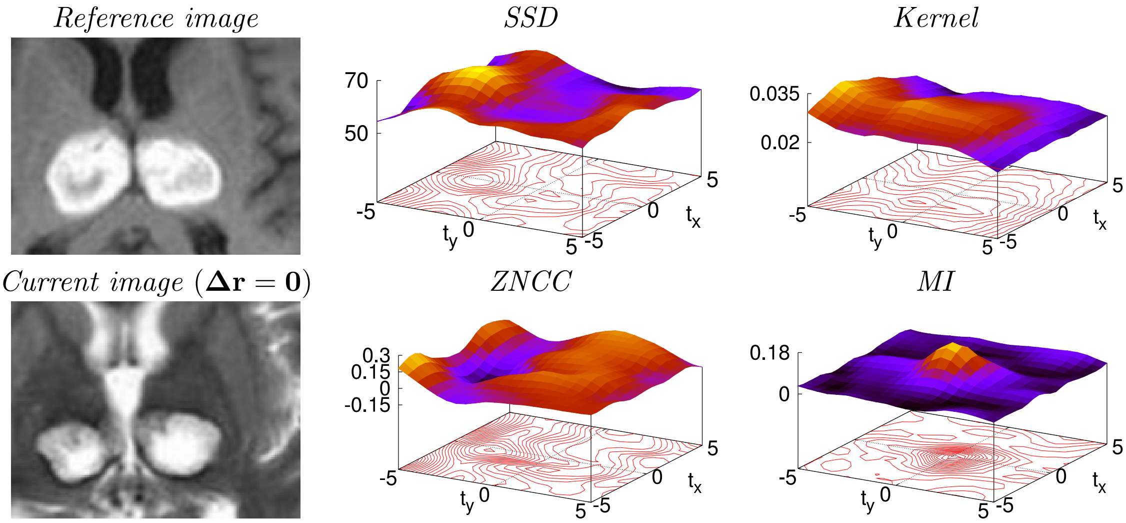


Figure 2.13: Robustness of the cost functions with respect to multimodality: application to the medical field with two different magnetic resonance images.

Airborne vs map images:

In this thesis, we also propose a new application of the MI-based multimodal matching to estimate the displacement between airborne images and map images. The two images used are provided by the IGN (Institut Géographique National) Geoportail website a tool similar to Google earth.

Figure 2.14 shows the values of the four alignment measures in the case of a multimodal alignment task. As it was the case in the previous multimodal example, only the mutual information effectively copes with the transformation of the pixel intensities between the airborne image and the map image.

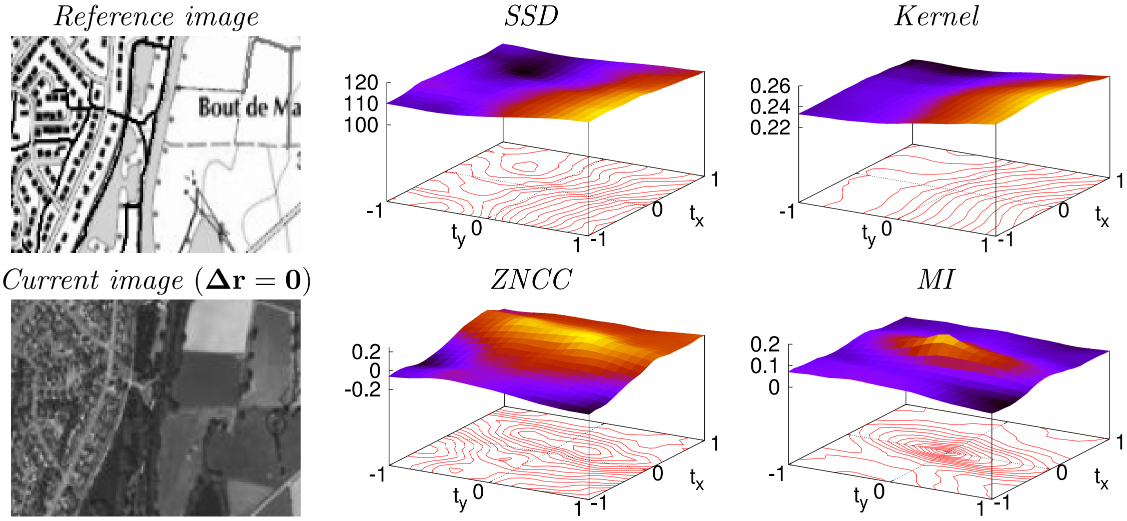


Figure 2.14: Robustness of the cost functions with respect to multimodality: alignment between one airborne image and one map.

2.4 Conclusion

In this chapter, we show that using some basics on statistics and information theory a very robust alignment function, mutual information, can be defined. Among many applications, we will show in the next chapters that MI can be used in optimization problems to track some objects in image sequences or to control a robot using visual servoing tasks.

Part II

Visual tracking

Introduction

Visual tracking is one of the main research areas in computer vision. The overall objective is to estimate the displacement of an object in an image sequence. Depending on the application, a coarse estimation of the displacement can be sufficient (for example in surveillance). In this thesis, we are interested in this last application, but also on the development of approaches that allows the complete localization of an object in the 3D space. Prospective applications range from image based-robot control to augmented reality or mosaicing.

Augmented reality and robotic applications illustrate how challenging defining a visual tracking method is [Marchand 2005a]. First, the algorithm must be robust: it has to keep on tracking even in the presence of illumination changes, occlusions and many other variations of the object aspect. Second, it must be accurate: if the goal is, for instance, to control a gripper mounted on a 6 degrees of freedom robot arm, then the positioning error between the effector and the object has to be small. Finally, the tracking algorithm must usually provide this localization information in real-time (at video rate).

In computer vision or robot vision literature, the history of visual tracking is really significant. We begin this part with a quick state of the art on the existing visual tracking approaches. In this context, we define our new tracking algorithm based on the mutual information registration function. Since MI was previously defined in the Chapter 2, this part focuses on the actual way to use MI in tracking applications within a differential approach. We detail our contribution on the optimization method in the 2D tracking problem. This new approach is validated through many experiments where we show the improvements in terms of accuracy, robustness and computational efficiency. A generalization to the 3D localization (or pose estimation problem) follows and ends with several experiments that show significant results.

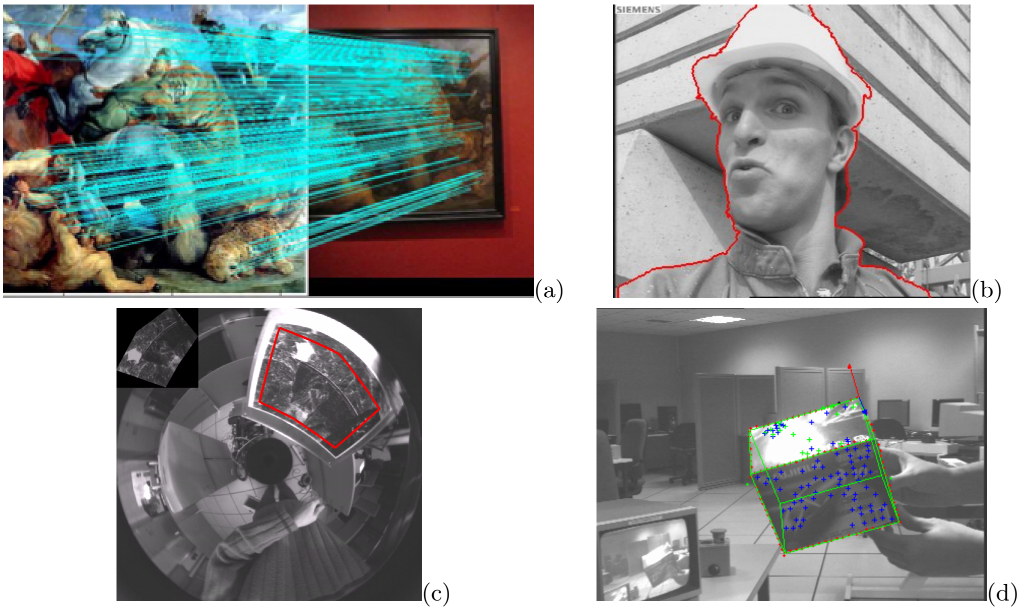


Figure 2.15: Some illustrations of classical tracking approaches over the time: (a) tracking points of interest [Lowe 2004], (b) contours tracking [Blake 1998], (c) SSD tracking with omnidirectional camera [Mei 2006], (d) hybrid pose estimation approach based on both the contours and appearance of the object [Pressigout 2005].

Chapter 3

Tracking as a registration issue

The literature related to visual tracking is very rich. In this chapter, we will mainly focus on the approaches that are related to our context of real-time monocular tracking of rigid objects. We organize this chapter following the classical steps of a tracker's creation: first, we need to know what information is relevant in the image. This information depends on the considered object to be tracked as well as its displacement to be estimated. Once the visual features are chosen, we need to define the matching or registration process, and, define a suitable parametrization to estimate its 2D position or 3D position, called its pose.

Most of the presented algorithms can be incorporated into filtering processes such as Kalman filters [Bar-Shalom 1993] or particle filters [Li 2004, Ababsa 2007] to increase their robustness. Nevertheless, our goal is not to propose contributions to the filtering processes, therefore, we will not focus on these aspects.

3.1 Modeling the object's appearance

The properties of the object appearance in the image can be divided in two classes, some that are local, meaning basically geometrical features such as points or contours, and the ones that are more general, representing for instance its color, its texture or its information.

3.1.1 Local features

A way to describe the tracked objects can be performed using simple geometric features such as dots, points of interest [Harris 1988], angles, contours [Berger 1994, Blake 1998], straight lines, segments [Hager 1998b, Boukir 1998, Marchand 1999], ellipses [Vincze 2001, Marchand 1999], etc. Some good examples of tracking systems using such features are the XVision [Hager 1998b] framework or the ViSP library [Marchand 2005b], that were, for instance, used to extract and track the features in the experiments presented in Figure 3.1.

These geometrical features can be enriched by a local appearance model as in the tracking by matching approaches that mainly consider the notion of keypoint characterized by a descriptor. Several approaches have been proposed, the well-known SIFT [Lowe 2004] and SURF [Bay 2006] approaches use histograms of gradients to define the local models, while the FERNS approach [Ozuysal 2007] defines it using the surrounding pixel intensities.

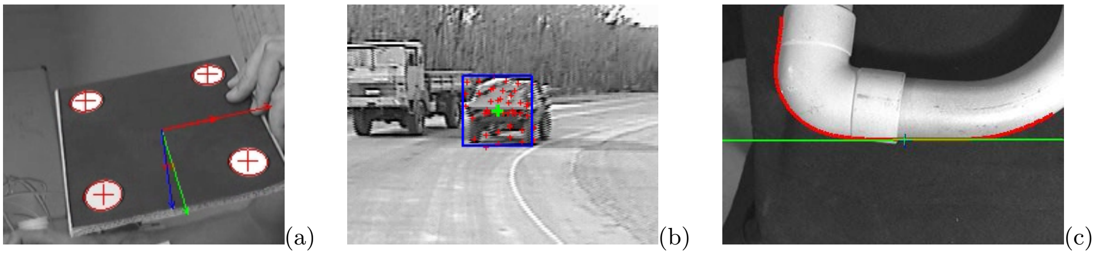


Figure 3.1: Local features are mainly defined by geometrical properties in the image as (a) dots, (b) points of interest and (c) contours.

The main advantages of this class of approaches are their simplicity and, now, their low computation cost. Moreover, these features represent geometrical properties of the object, which are, in general, invariant to illumination variations. On the other side, they totally depend on the presence of geometrical features on the object. Furthermore, the quality of the results (precision and robustness),that typically depends on the number of features, can not always be guaranteed and depends on the scene (for example, these approaches are usually very dependent on the features density and on the occlusion phenomenons).

3.1.2 Global features

The previous approaches rely mainly on the analysis of intensity gradients in the images. Another possibility is to directly consider the image intensity and to model the object using its colors, defining the appearance as an histogram [Comaniciu 2000], or using its general appearance in order to store the whole texture of the object [Lucas 1981, Irani 1998, Jurie 2001, Benhimane 2004].

These global features depict the appearance of the object on a large set of positions, while each geometrical feature depicts one particular position. To evaluate the effect of the displacement on one global feature, it is required to transform the whole set of positions. The resulting tracking approaches are therefore usually more complex than the local features methods and also more time consuming. Moreover, since the appearance of the object depends on the illumination variations, most of the resulting tracking approaches are sensitive to the illumination conditions. Nevertheless, these approaches do not require any particular features extraction process in the image. They remain relatively robust to partial occlusions. Finally, since most of the information of the image is used to model the object, the resulting estimation of the object motion is very accurate thanks to the redundancy of the available information.

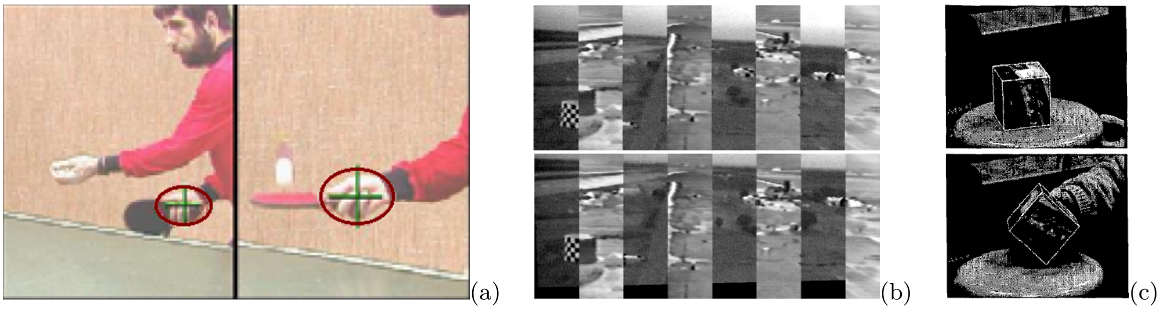


Figure 3.2: Global features tracking: (a) using histograms with the Mean-shift approach [Comaniciu 2000], (b) texture based registration with the NCC [Irani 1998] and (c) tracking from differences (SSD) [Jurie 2001].

3.2 The registration formulation

Once the object's visual features are chosen, the goal is to match these features from the model (2D or 3D) of the object with the current image features. In this section, we differentiate two registration approaches: one that estimates the transformation in the image plane, while the other is performed between one image and 3D model of the tracked object (the pose or viewpoint estimation problem).

3.2.1 2D registration or motion estimation

The tracking problem can usually be formulated as the optimization of a similarity function between the current input image I_t of an image sequence $(I_0, ..., I_N)$ and a model $\mathcal{M}$:

$$\widehat{\mathbf{p}}_t = \arg\max_{\mathbf{p}} f(I_t, \mathcal{M}(\mathbf{p})) \tag{3.1}$$

where $\mathbf{p}$ are parameters representing the position of the model in the 2D image space and $\widehat{\mathbf{p}}_t$ are the parameters that best map the model with the current image in the sens of f. The problem is different whether the features of the model are local or global: the local features have generally their low-level peculiar tracking or matching process and the information provided by the displacement of all the features allows the estimation of the object's displacement, while one global feature is enough to estimate this displacement.

3.2.1.1 Local features-based registration

The use of local features classically induces a first low level tracking or matching approach. A feature point can, for instance, be defined as a corner in the appearance of the object. But the feature can also be enriched using a local descriptor [Lowe 2004, Bay 2006] of appearance and define a keypoint. The descriptor will allow for matching the feature points between several images (see Figure 3.3(a)). Similarly a line can be tracked along an image sequence with the moving edges algorithm (see Figure 3.3(b)) [Bouthemy 1989]. This low level tracking tasks provide an estimation of the displacement for each independent feature. Since the goal is to find the motion of a rigid object, a global motion constraint has to be applied.

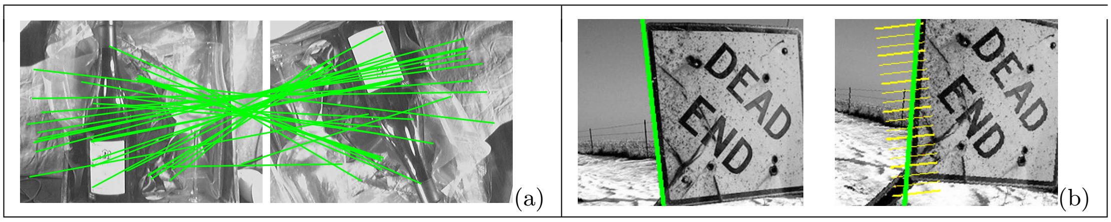


Figure 3.3: Some low-level matching and tracking processes. (a) Key-points matching between two images using local descriptors constructed with the image gradients [Bay 2006] and (b) moving edge algorithm [Bouthemy 1989]: the new contour line is searched as a strong edge along the normal of the previous line position.

If the geometrical features in a first image are defined as vectors $\mathbf{y}^*$ and matched with the features $\mathbf{y}$ in another image, the registration problem can then be defined as the minimization of their reprojection error from one frame to another. Using a warping function w, the displacement parameters $\widehat{\mathbf{p}}$ that best map $\mathbf{y}$ into $\mathbf{y}^*$ are found using:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{m=1}^{M} \|\mathbf{y}_m^* - w(\mathbf{y}_m, \mathbf{p})\|$$

where M is the number of visual features. If, for instance, the features are interest points $\mathbf{x}$ and the displacement is an homography [Berger 1998, Simon 2000, Pressigout 2005], then, the problem is:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{m=1}^{M} \|\tilde{\mathbf{x}}_m^* - \mathbf{H}\tilde{\mathbf{x}}_m\|$$

where $\mathbf{H}$ is the homography matrix and $\tilde{\mathbf{x}}_m$ are the homogeneous coordinates of the point $\mathbf{x}_m$ (see chapter 1). In some cases, this problem can be solved linearly as we showed in the mosaicing example in Frame 1 page 16. Nevertheless, most of the feature models and their motion are non-linear. Then the resolution of the problem requires non-linear optimization methods, such as a Gradient descent method (see Frame 3). Since it is classical to have some failure in the low-level tracking or matching process, robust methods can be performed [Stewart 1999, Malis 2006] to decrease the effect of outliers.

3.2.1.2 Appearance-based registration

The appearance-based approaches, also known as template-based approaches, are different in the way that there is no low-level tracking or matching processes. Indeed, let us consider that the appearance of the object is learned from a model defined as a reference image I^* at some pixel locations $\mathbf{x} \in \mathcal{W}$ and that we seek its new location $w(\mathbf{x}, \mathbf{p})$ in an image I. Then we can directly define the registration problem as minimizing the dissimilarity (or maximizing the similarity) between the appearance in I^* at the positions $\mathbf{x}$ in a region $\mathcal{W}$ and in I at the positions $w(\mathbf{x}, \mathbf{p})$. An analytic formulation of the tracking problem can then be written as:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} f\left(I^*(\mathbf{x}), I(w(\mathbf{x}, \mathbf{p}))\right)$$

where f is a dissimilarity function. Considering that the appearance is defined as the set of pixel intensities of the patch and that the dissimilarity function is the SSD, leads typically to the KLT algorithm [Lucas 1981, Shi 1994] for small patch and translational model or to the algorithm of [Hager 1998a, Belhumeur 1999, Baker 2004] for large template and affine motion:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{x} \in \mathcal{W}} \left(I^*(\mathbf{x}) - I(w(\mathbf{x}, \mathbf{p}))\right)^2$$

This algorithm is described in details in Frame 4. Since the problem is defined with the images difference, this formulation is very sensitive to the perturbations of the object's appearance. Therefore, several approaches have been proposed to robustify it. To add robustness with respect to occlusions or specularities, one can for instance use a robust estimation process. A proposed solution is to consider M-estimators [Odobez 1995, Hager 1998a] and modify the previous formulation as:

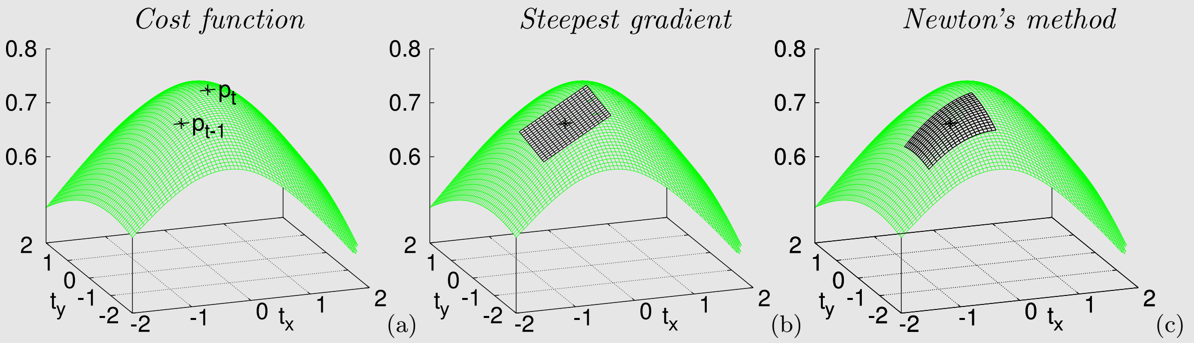
$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{x} \in \mathcal{W}} \rho\left(I^*(\mathbf{x}) - I(w(\mathbf{x}, \mathbf{p}))\right)$$

where ρ is a function that decreases the sensitivity of the cost function to outliers. The problem can also be modified to model the illumination variations of the object [Hager 1998a, Silveira 2007]. In [Hager 1998a], a learning step is performed to model the possible illumination variations of the object using a principal component analysis. In [Silveira 2007], no learning step is required: several parameters are added to the optimization problem, with one parameter β to model the global illumination variation and a set of parameters to model a patch of local variations $\mathcal{S}$ (an image of illumination variations). The optimization problem becomes:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}, \beta, \mathcal{S}} \sum_{\mathbf{x} \in \mathcal{W}} \rho\left(I^*(\mathbf{x}) - (\mathcal{S}(\mathbf{x}) I(w(\mathbf{x}, \mathbf{p})) + \beta)\right)$$

Nevertheless, the additional parameters are slowing the convergence of the optimization. To keep the same number of parameters and change the robustness of the tracking problem, another solution is simply to consider another alignment function f or other global features. The

Frame 3 Non-linear optimization.



The goal of an optimization problem is to find the best solution for one problem. Typically an analytical formulation defines it using parameters $\mathbf{p}$ and a cost function f that is maximal for the best parameters $\widehat{\mathbf{p}}$:

$$\widehat{\mathbf{p}} = \arg\max_{\mathbf{p}} \ f(\mathbf{p})$$

In some cases, this problem can be solved linearly. In our tracking problem it is rarely the case. The problem is hence solved with a non-linear optimization: a first estimation of the parameters $\mathbf{p}^0$ is iteratively refined using an update $\boldsymbol{\Delta}\mathbf{p}$ until it reaches the maximum of f (let us note the iteration number by the superscript k). Two widespread approaches are the steepest gradient descent and the Newton's method. If the update step is defined as an addition ($\mathbf{p}^{k+1} = \mathbf{p}^k + \boldsymbol{\Delta}\mathbf{p}^k$), these approaches can be defined as follows.

Steepest gradient descent

Using a parameter vector $\mathbf{p} \in \mathbb{R}^2$, the steepest gradient descent (ascent in our example) can be illustrated as a climbing problem: suppose you are on a hill (the cost function), searching for the top of it (the optimum) but you can not see the relief of the place (only know the function locally), a good solution is to approximate the local relief as a plane (the function derivatives: see the top figure (b)) and follow the steepest ascent. To find the optimum, the first guess of the parameter $\mathbf{p}^0$ is iteratively updated using:

$$\boldsymbol{\Delta}\mathbf{p}^k = \alpha\mathbf{G} \qquad \text{with:} \qquad \mathbf{G} = \left.\frac{\partial f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}^k}$$

where α is a scale factor applied to the Gradient $\mathbf{G}$ to avoid large steps that can cause the optimization to diverge. This process is repeated until convergence.

Newton's method

The Newton's method also uses a local approximation of the cost function in order to update the current displacement parameters. While the steepest descent locally approximates the function by its first-order derivatives (by a plane if $\mathbf{p} \in \mathbb{R}^2$), the Newton's method uses its second order derivatives (f is locally approximated by a parabola, see top figure (c)). This approximation is performed using the Taylor expansion of $\partial f / \partial \mathbf{p}$ at $\mathbf{p}^{k+1}$:

$$\left.\frac{\partial f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}^{k+1}} = \left.\frac{\partial f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}^k} + \left.\frac{\partial^2 f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}^2}\right|_{\mathbf{p}=\mathbf{p}^k} \boldsymbol{\Delta}\mathbf{p}^k$$

Since the goal is to find $\mathbf{p}^{k+1} = \widehat{\mathbf{p}}$ for which the derivative of f are zero, the update is obtained using:

$$\boldsymbol{\Delta}\mathbf{p}^k = -\left.\frac{\partial^2 f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}^2}\right|_{\mathbf{p}=\mathbf{p}^k}^{-1} \left.\frac{\partial f(\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\mathbf{p}=\mathbf{p}^k}$$

The process is repeated until $\mathbf{p}$ converges, *i.e.* until $\boldsymbol{\Delta}\mathbf{p}$ is small enough.

object can indeed be characterized using histograms (of colors, for instance) [Comaniciu 2000], where f defines the distance between two histograms as the Batthacharrya [Bhattacharyya 1943] or Matuzita distance [Cha 2008]. Several approaches have also been proposed to directly define a robust alignment function between two set of pixel intensities. For instance, some approaches define f as the Normalized Cross Correlation [Irani 1992] or the mutual information [Collignon 1995, Viola 1995]. While MI cost function has a shape adapted to the tracking problem (see section 2.3.3), there is still one drawback: no approach such as Gauss-Newton (for the SSD function) exists to perform the optimization in a simple and efficient way. Therefore complex optimization approaches, such as Line-search methods, are used [Thévenaz 2000, Dowson 2006, Dowson 2008], that are time-consuming and limit the accuracy of the estimation.

Warp update rule

Since all these alignment problems can not be solved linearly, a non-linear optimization approach is usually required. The basic idea is that a first guess of the displacement parameters is available and the goal is to refine it. Several approaches are then possible. All the possibilities depend on how the refinement process (or update) is applied to the current parameters, *i.e.* the warp update rule [Baker 2001]. In the following table, we reported several of these update rules and their corresponding optimization formulations.

Method	Optimization	Update
Forward additional	$\mathbf{\Delta p}^k = \arg\min_{\mathbf{\Delta p}} f(I^*(\mathbf{x}), I(w(\mathbf{x}, \mathbf{p}^k + \mathbf{\Delta p})))$	$\mathbf{p}^{k+1} = \mathbf{p}^k + \mathbf{\Delta p}^k$
Forward compositional	$\mathbf{\Delta p}^k = \arg\min_{\mathbf{\Delta p}} f(I^*(\mathbf{x}), I(w(w(\mathbf{x}, \mathbf{\Delta p}), \mathbf{p}^k)))$	$w(\mathbf{x}, \mathbf{p}^{k+1}) \leftarrow w(w(\mathbf{x}, \mathbf{\Delta p}^k), \mathbf{p}^k)$
Inverse compositional	$\mathbf{\Delta p}^k = \arg\min_{\mathbf{\Delta p}} f(I^*(w(\mathbf{x}, \mathbf{\Delta p})), I(w(\mathbf{x}, \mathbf{p}^k)))$	$w(\mathbf{x}, \mathbf{p}^{k+1}) \leftarrow w(w^{-1}(\mathbf{x}, \mathbf{\Delta p}^k), \mathbf{p}^k)$

Other approaches are possible such as in [Odobez 1995, Hager 1998a], we do not report them in details since their formulation can not be simplified as the previous ones. The forward additional approach [Lucas 1981] has the advantage of its simplicity, it is very intuitive and suitable to any warp, but it is very expensive to compute. The forward compositional approach [Shum 2000] has the advantage of its relatively good efficiency compared to the forward additional formulation. Since the forward compositional formulation is both intuitive and relatively efficient, this will be the approach used to introduce the mutual information optimization in the next chapter. Nevertheless, the inverse approaches [Odobez 1995, Hager 1998a, Baker 2001] allow a large increase of the efficiency of the tracking problem. In the remainder of this thesis, we preferred to focus on the inverse compositional approach [Baker 2001] for its simplicity. A combination of both the forward and inverse formulation was also proposed in [Benhimane 2004] to improve the convergence rate of the tracking approach.

3.2.2 Pose estimation

The goal of the pose estimation problem is to find the relative pose between the camera and the object corresponding to an image. This is also a registration problem, but in that case, the 3D model of the object is required. This model can be known [Drummond 2002, Vacchetti 2004, Comport 2006] or learned from a sequence of images [Lim 1988], or both the tracking and reconstruction can be performed simultaneously [Davison 2007].

We focus here on applications where the 3D model and camera calibration are known. In this case, the problem can be formulated as an optimization of a similarity function between the input images I^* and the projection of the model $\mathcal{M}$ using a camera with its intrinsic parameters

Frame 4 KLT algorithm [Lucas 1981, Tomasi 1991, Shi 1994].

The KLT (for Kanade Lucas Tomasi) algorithm is an approach that estimates the displacement of a template between two images. If we consider two frames I_0 and I_t of an image sequence, the goal is to estimate the transformation of a small patch of pixels $\mathcal{W}$ between the two frames. If we search the transformation $w(\mathbf{x}, \mathbf{p}_t)$ that maps the pixel $\mathbf{x}$ of I_0 into the pixels $\mathbf{x}_t$ of I_t, then the brightness constancy constraint gives:

$$I_t(\mathbf{x}_t) = I_t(w(\mathbf{x}, \mathbf{p}_t)) = I_0(\mathbf{x})$$

for all the pixels $\mathbf{x}$ in the patch $\mathcal{W}$. The transformation that best respects this equation over the whole patch is the one that minimizes the difference:

$$\widehat{\mathbf{p}_t} = \arg\min_{\mathbf{p}} \ \mathcal{C}(\mathbf{p}) \qquad \text{with:} \qquad \mathcal{C}(\mathbf{p}) = \sum_{\mathbf{x}\in\mathcal{W}} (I_t(w(\mathbf{x}, \mathbf{p})) - I_0(\mathbf{x}))^2$$

This problem is reformulated as searching $\mathbf{p}_t$ for which the derivative of $\mathcal{C}$ is zero. It is solved using a non-linear approach (see Frame 3), where a first guess $\mathbf{p}_t^0 = \widehat{\mathbf{p}_{t-1}}$ is iteratively updated in order to find $\widehat{\mathbf{p}_t}$ ($\lim_{k\to\infty} \mathbf{p}_t^k = \widehat{\mathbf{p}_t}$). The update $\mathbf{\Delta p}^k$ of the parameters $\mathbf{p}_t^k$ is searched to approximate:

$$\mathbf{\Delta p}^k = \arg\min_{\mathbf{\Delta p}} \mathcal{C}(\mathbf{\Delta p}) = \arg\min_{\mathbf{\Delta p}} \sum_{\mathbf{x}\in\mathcal{W}} (I_t(w(\mathbf{x}, \mathbf{p}_t^k + \mathbf{\Delta p})) - I_0(\mathbf{x}))^2 \tag{3.2}$$

Using a first order Taylor expansion, this yields:

$$\mathbf{\Delta p}^k = \arg\min_{\mathbf{\Delta p}} \sum_{\mathbf{x}\in\mathcal{W}} \left(I_t(w(\mathbf{x}, \mathbf{p}_t^k)) + \nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \mathbf{\Delta p} - I_0(\mathbf{x}) \right)^2 \tag{3.3}$$

where ∇I_t is the gradient of the image at the pixel $w(\mathbf{x}, \mathbf{p}_t^k)$. At the optimum of $\mathcal{C}$, the derivative of $\mathcal{C}$ with respect to $\mathbf{\Delta p}$ is zero:

$$\frac{\partial \mathcal{C}}{\partial \mathbf{\Delta p}} = \sum_{\mathbf{x}\in\mathcal{W}} \left(\nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \right)^\top \left(I_t(w(\mathbf{x}, \mathbf{p}_t^k)) + \nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \mathbf{\Delta p} - I_0(\mathbf{x}) \right) = 0 \tag{3.4}$$

That yields the final update expression:

$$\mathbf{\Delta p}^k = \left(\sum_{\mathbf{x}\in\mathcal{W}} \left(\nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \right)^\top \left(\nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \right) \right)^{-1} \sum_{\mathbf{x}\in\mathcal{W}} \left(\nabla I_t \frac{\partial w}{\partial \mathbf{\Delta p}} \right)^\top (I_0(\mathbf{x}) - I_t(w(\mathbf{x}, \mathbf{p}_t^k))) \tag{3.5}$$

The update is computed and applied to the current parameter $\mathbf{p}_t^k$ until convergence. We can observe the similitude with the Newton's approach presented in Frame 3, where the second derivatives of the Hessian matrix are neglected (Gauss-Newton method).

$\boldsymbol{\gamma}$ and pose $\mathbf{r}$. The formulation can be written:

$$\widehat{\mathbf{r}} \;=\; \arg\max_{\mathbf{r}} f\left(I^*, pr_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r})\right). \tag{3.6}$$

The similarity function f can represent the distance between the geometrical features of I^* and the model or represent the similarity between the image and the projection of textured 3D model of the object.

The first approach that uses geometrical features is the most commonly used approach. Indeed, only the geometrical properties of the object are required and their projection into the image space is easy to perform. For instance, the model can be defined by 3D points ${}^o\mathbf{P}_m$ in the object frame and matched with 2D points $\tilde{\mathbf{x}}_m$ in the image. The problem is then to find the homogeneous matrix ${}^c\mathbf{M}_o$ (equivalent of the pose $\mathbf{r}$) that minimizes the reprojection error:

$$\widehat{\mathbf{r}} \;=\; \arg\min_{\mathbf{r}} \sum_{m=1}^{M} \|\tilde{\mathbf{x}}_m - \mathbf{K}{}^c\mathbf{M}_o\,{}^o\tilde{\mathbf{P}}_m\| \tag{3.7}$$

one classical approach is for instance to use the Dementhon approach where only four points are sufficient [Dementhon 1995] to compute $\mathbf{r}$.

Although geometric 3D models are usually considered, some methods use global model of appearance (textured models). We can, for instance, cite one approach using directly the pixel intensities coupled with a robust estimation in [Comport 2007] and one using mutual information in [Panin 2008], with the same drawback as the MI based 2D tracking approaches. Since these approaches use the whole information of the image, they yield an accurate estimation of the pose. Finally some approaches that mix both local and global features have been proposed in [Masson 2003, Kyrki 2005, Pressigout 2007]. These approaches take advantages of both local and global methods in terms of accuracy and robustness.

The maximization defined in the pose computation problem remains very similar to the tracking problem that was previously defined. A first guess of the parameters, that is the current camera pose, is known and the goal is to iteratively refine it to reach the optimum of the alignment function corresponding to the desired camera pose. Since the problem is not linear, the optimization of the cost function is in general performed using a Gradient descent approach, where the variations of the cost function have to be computed with respect to the variations of the parameters.

Several approaches are possible to perform the refinement. The pose can be updated using the additional or compositional approaches. In the additional approach, the goal at each iteration of the optimization is to find the update $\boldsymbol{\Delta}\mathbf{r}$ that brings to the maximum of the cost function f using:

$$\boldsymbol{\Delta}\mathbf{r}^k \;=\; \arg\max_{\boldsymbol{\Delta}\mathbf{r}} f\left(I^*, pr_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r}^k + \boldsymbol{\Delta}\mathbf{r})\right). \tag{3.8}$$

And the update $\mathbf{r}^{k+1} = \mathbf{r}^k + \boldsymbol{\Delta}\mathbf{r}$ is performed until we reach the convergence.

Another solution is to use the similitude with the visual servoing problem where the camera pose is updated using a velocity [Sundareswaran 1998, Marchand 2002]. The virtual visual servoing (VVS) approach follows the same update rule: the current camera pose $\mathbf{r}^k$ is updated using the velocity $\mathbf{v}$ of the virtual camera as:

$${}^{c^{k+1}}\mathbf{M}_o = e^{[\mathbf{v}]}\;{}^{c^k}\mathbf{M}_o \tag{3.9}$$

where $e^{[\mathbf{v}]}$ is the exponential map of $\mathbf{v}$. Since the velocity depicts the variation of the pose with respect to the time, the “period of time” between two iterations is considered as a gain.

Whatever the update solution is, the optimization problem usually requires to estimate the variation of the visual features (*i.e.* the variation of the points, edges, SSD, MI) with respect to the variation of the pose. Since visual features are build on points (a line is defined by two points and a patch by a set of points), the basis is to know the variation of a point position with respect to the variation of the pose. The advantage of the VVS approach comes from its simplicity to compute these variations compared to the other approaches (see the Frame 9 versus the Appendix B.2). However, we show in the Appendix that some simplifications on the compositional approach lead to an equation equivalent to the virtual visual servoing approach. An equivalent approach was also proposed in [Drummond 2002] using the Lie algebra (equivalence proved in [Comport 2005]).

3.3 Conclusion

In this chapter, we have seen that the tracking (2D or 3D) problem can be solved using many approaches mainly depending on the visual features used to represent the object. Geometrical features, computed using the strong gradients in the images, have the advantages of their robustness and simplicity. Nevertheless they require a first low-level tracking or matching step that remains their main difficulty.

To solve this problem, global visual features can be used to model the appearance of the object in terms of intensity. Since these intensities vary with occlusions and illumination variations, most of this method have a lack of robustness. To improve the robustness of the tracker, the optimization can be build on a robust alignment function, mutual information. Nevertheless, although MI based registration techniques were already largely studied, the existing approaches still show many limits that will be discussed and solved in the following chapter.

Chapter 4

Mutual information-based tracking

In this chapter, we present the mutual information-based tracking approach. We detail the existing approaches and exhibit some drawbacks of these methods and present our solutions to solve these issues.

The first section of this chapter recall the principle of the template based tracking approach. To begin with an intuitive formulation, the problem is first defined using a forward compositional formulation. We apply the classical optimization methods to MI, and discuss them to present our new optimization approach. This section ends with an empirical evaluation that compares the existing optimization approaches with the proposed one. The following section presents some possible improvements of the optimization method to make it more efficient: we define the inverse compositional formulation. Then, we study the choice of the reference pixels to improve the computation time, and finally, we present coarse-to-fine estimation approaches that improve both the robustness and efficiency of the tracker. To validate the performance of the proposed MI-based tracker, we end this chapter with several tracking and registration experiments in various contexts.

4.1 MI based registration: forward compositional formulation

In this section, we recall the definition of our tracking problem and develop it using a compositional formulation. We detail the classical approaches that are possible to solve the underlying optimization problem. In the same time, we detail the derivation of MI with respect to the motion parameters, that is necessary to perform the optimization methods. We highlight the drawbacks of the existing approaches and present our new optimization method as a response. Finally an empirical evaluation of the approaches efficiency is presented.

4.1.1 Mutual information optimization

The goal of our tracking problem is to align an image template I^* with an input image I. If we assume that the reference template appears in I, the goal is to search for the transformation that aligns the pixels $\mathbf{x}$ of the reference image I^* to the corresponding pixels $\mathbf{x}'$ of I. Assuming that the transformation from the reference points to the input image can be modeled by a warp function, as defined in the previous section, the problem can be formulated as:

$$\begin{aligned} \widehat{\mathbf{p}} &= \arg\max_{\mathbf{p}} f\left(I^*(\mathbf{x}), I(\mathbf{x}')\right) \\ &= \arg\max_{\mathbf{p}} f\left(I^*(\mathbf{x}), I(w(\mathbf{x},\mathbf{p}))\right) \qquad (4.1) \end{aligned}$$

where f is the cost function that measures the alignment between the two images and thus $\widehat{\mathbf{p}}$ are the displacement parameters that best align the reference image with the input image I in the sens of f. The more the alignment function f is robust to the appearance variations of the object, the more the tracking problem is itself robust. Therefore, following the evaluation of the alignment functions that were performed in section 2.3.3, we define f as the mutual information function because of its robustness to occlusions and illumination variations, and we rewrite the problem as:

$$\widehat{\mathbf{p}} \;=\; \arg\max_{\mathbf{p}} \mathrm{MI}\left(I^*(\mathbf{x}), I(w(\mathbf{x},\mathbf{p}))\right) \tag{4.2}$$

Since this problem is impossible to solve linearly, a non-linear optimization is performed.To initialize the optimization, a first guess of the displacement parameters is required. Since we suppose that the displacement of the object between two consecutive frames is small, a good approximation is to approximate the parameter $\mathbf{p}_t$ of the input image I_t at a time t using the parameters estimated for the previous frame: $\mathbf{p}_t = \widehat{\mathbf{p}_{t-1}}$ (see the Figure 4.1).

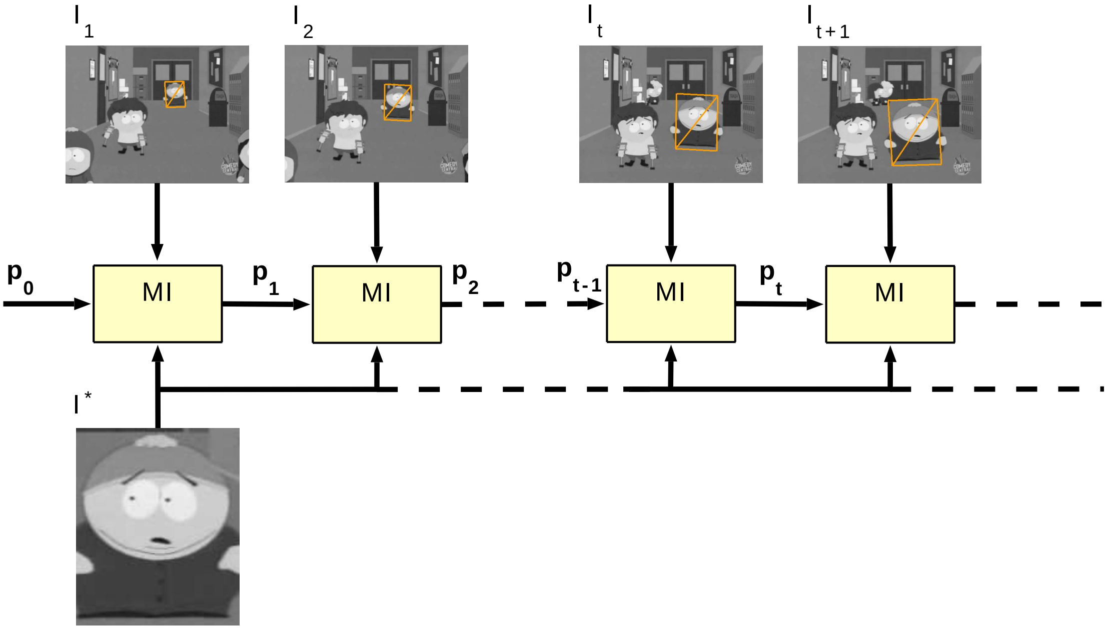


Figure 4.1: Mutual information-based tracking.

To initialize the whole tracking approach, the position of the template in the first image I_0 has to be known. Since the first image of the sequence is usually the one that defines the template I^*, the first displacement parameters $\mathbf{p}_0$ between the template and the first image simply correspond to an identity transformation (considering the warp functions defined in the first chapter it yields: $\mathbf{p}_0 = \mathbf{0}$). Otherwise, the first estimation can be performed using some matching process, such as a keypoints matching approach [Lowe 2004, Lepetit 2006].

The first approximation of the displacement $\mathbf{p}_t^0 = \widehat{\mathbf{p}_{t-1}}$ is then refined using the numerical resolution of the equation (4.2). To solve the maximization, an iterative optimization method is used that successively goes closer and closer to the optimum of the cost function $\widehat{\mathbf{p}_t}$ (see Figure 4.2). For a clarity purpose, let us now consider the maximization peculiar to one image I (we drop the subscript t) and focus on the iteration number noted using the superscript k. In the forward compositional approach [Shum 2000], the goal is then formulated as finding the update $\mathbf{\Delta p}$ that leads to the optimum, so that, at each iteration k, we seek:

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

with:

$$w(\mathbf{x},\mathbf{p}^{k+1}) \leftarrow w(w(\mathbf{x},\mathbf{\Delta p}^k),\mathbf{p}^k) \tag{4.3}$$

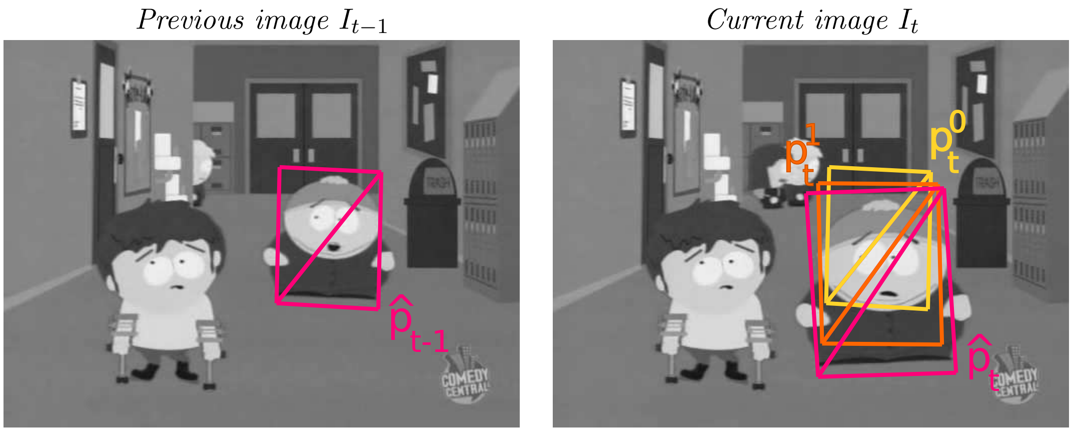
Remark that the warp function should correspond to an identity transformation if the update is null. Therefore it must satisfy: $w(\mathbf{x}, \mathbf{0}) = \mathbf{x}$ (that is the case for all the transformation functions defined in the previous chapters). For a clarity purpose, let us note $\mathbf{p}^{k+1} = \mathbf{p}^k \circ \mathbf{\Delta p}^k$ where $\circ$ is the composition operator. As an example, we provide the detailed computation of $\mathbf{p}^{k+1}$ with respect to $\mathbf{p}^k$ and $\mathbf{\Delta p}^k$ in the Appendix A.1 for the affine model. If the previous expression is approximately respected at each iteration, the displacement parameters should converge to the optimum of MI:

$$\widehat{\mathbf{p}} = \lim_{k \to +\infty} \mathbf{p}^k \tag{4.4}$$

Of course, the number of iterations k has to be limited for the efficiency of the approach. To do so, a convergence test is performed to evaluate the significance of the update. Since the elements of $\mathbf{\Delta p}^k$ have not the same units, its Euclidean norm does not give a proper measure of its significance. To evaluate it, we compute its effect in terms of displacement in the image. We set M reference points $\mathbf{x}_m$ (in our experiments, we choose the corner of the region of interest in I^*) and compute the effect as:

$$d(\mathbf{\Delta p}^k) = \frac{1}{M} \sum_{m=1}^{M} \| w(\mathbf{x}_m, \mathbf{p}^{k+1}) - w(\mathbf{x}_m, \mathbf{p}^k) \| \tag{4.5}$$

where $d(\mathbf{\Delta p}^k)$ represents the mean of the displacement of the reference points in the image caused by $\mathbf{\Delta p}^k$ in pixels. If this displacement is below a threshold ε, *i.e.* $d(\mathbf{\Delta p}^k) < \epsilon$, we consider that the algorithm has converged ($\widehat{\mathbf{p}} = \mathbf{p}^k$) and we stop the iterative process.


Previous image I_{t-1}

Current image I_t

Figure 4.2: The first approximation of the position $\mathbf{p}_t^0$, given by the previous position $\widehat{\mathbf{p}_{t-1}}$, is iteratively refined to find the optimal parameters $\widehat{\mathbf{p}_t}$ (images from Comedy Central).

Let us now focus on the classical optimization approaches that were used in the context of the mutual information maximization: the method of steepest descent [Viola 1995] (where the gradient was not computed analytically) and the Newton's method [Dowson 2008]. In the same time, the derivatives of MI are defined and we show the characteristics of MI that cause the existing optimization approaches to have a small convergence domain. Finally, we propose a solution to increase the convergence domain and efficiency of the registration problem. In the remainder of this section, we focus on the computation of one update step in one particular iteration. Therefore, the superscript k, that indicates the iteration number, will be omitted.

4.1.1.1 Steepest gradient-descent and MI gradient

Steepest gradient-descent is the basis of the non-linear optimization methods (see Frame 3 page 49). If the goal is to optimize a function with a few parameters (2D or 3D), this solution is practical and efficient. Nevertheless, in our tracking problem, where our goal is to estimate homographies using 8 parameters that are strongly correlated, the steepest gradient descent approach becomes slow and subject to oscillations in the update estimation. Although it was used in [Viola 1995], where the derivative is approximated with a stochastic approach, it is presented here only as an introduction to the other optimization methods and to the computation of the mutual information derivatives.

Optimization approach

As it was defined in Frame 3 page 49, the principle of the steepest gradient-descent is really simple: if we are near the maximum of the cost function and we “climb” it following the steepest ascent, then, after a few iterations, we should reach the maximum. The derivatives of the cost function are by definition the steepest descent direction. The goal of the steepest gradient-descent is thus to iteratively update the displacement parameters using the direction provided by the derivatives. One remaining problem is that there is no information about the amplitude needed to reach the optimum. A scalar α, the step-size, has thus to be applied to the derivative of the cost function to control this amplitude. The update parameters are then given by:

$$\boldsymbol{\Delta}\mathbf{p} \quad = \quad \alpha\mathbf{G} \qquad \text{with} \qquad \mathbf{G} = \left.\frac{\partial \mathrm{MI}\left(I^*(\mathbf{x}), I(w(\mathbf{x}, \mathbf{p}\circ\boldsymbol{\Delta}\mathbf{p}))\right)}{\partial\boldsymbol{\Delta}\mathbf{p}}\right|_{\boldsymbol{\Delta}\mathbf{p}=\mathbf{0}} \tag{4.6}$$

where $\mathbf{G}$ is the first order derivative of mutual information with respect to the displacement update, also called gradient. This gradient is computed for the current parameter $\mathbf{p}$, *i.e.* for $\boldsymbol{\Delta}\mathbf{p} = \mathbf{0}$. The definition of MI given through the partial volume interpolation allows the analytical computation of its derivatives. We recall that MI is obtained with:

$$\mathrm{MI}(I, I^*) = \sum_{i,j} p_{II^*}(i,j) \log\left(\frac{p_{II^*}(i,j)}{p_I(i)p_{I^*}(j)}\right)$$

where $p_I(i)$ and $p_{I^*}(j)$ are respectively the marginal probabilities of I and I^*, and $p_{II^*}(i,j)$ is the joint probability of I and I^*.

Mutual information gradient

Applying the derivative chain rules to equation (2.15) page 27 yields the following expression of the mutual information gradient [Dowson 2006]:

$$\mathbf{G} \quad = \quad \sum_{i,j} \frac{\partial p_{II^*}}{\partial\boldsymbol{\Delta}\mathbf{p}}\left(1 + \log\frac{p_{II^*}}{p_{I^*}}\right) \tag{4.7}$$

where we drop the i and j parameters for a clarity purpose. The details of the computation can be found in the Appendix B.1. As we can see in this equation, the gradient of mutual information depends on the derivatives of the joint probability. Applying the forward compositional formulation to MI yields the following joint probability expression:

$$p_{II^*} \quad = \quad \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \phi\left(i - \overline{I}(w(\mathbf{x}, \mathbf{p}\circ\boldsymbol{\Delta}\mathbf{p}))\right)\phi\left(j - \overline{I^*}(\mathbf{x})\right) \tag{4.8}$$

We recall that ϕ is a B-spline function that is chosen to be twice differentiable. Passing the derivative through the summation yields the following expression:

$$\frac{\partial p_{II^*}}{\partial \boldsymbol{\Delta}\mathbf{p}} = \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \frac{\partial \phi\left(i - \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))\right)}{\partial \boldsymbol{\Delta}\mathbf{p}} \phi\left(j - \overline{I^*}(\mathbf{x})\right) \tag{4.9}$$

The remaining expression to evaluate is the variation of the function ϕ with respect to the update. Its derivative is obtained using the chain rules and gives:

$$\frac{\partial \phi(i - \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p})))}{\partial \boldsymbol{\Delta}\mathbf{p}} = -\frac{\partial \phi}{\partial i} \frac{\partial \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}} \tag{4.10}$$

The way to compute the B-spline derivative is detailed in the Chapter 2 among the MI definition. Finally the derivative of the current image intensity with respect to the update parameters $\boldsymbol{\Delta}\mathbf{p}$ is given by the following expressions:

$$\begin{aligned}\frac{\partial \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}} &= \nabla \overline{I} \left.\frac{\partial w(w(\mathbf{x}, \boldsymbol{\Delta}\mathbf{p}), \mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\boldsymbol{\Delta}\mathbf{p}=\mathbf{0}} \\ &= \nabla \overline{I} \left.\frac{\partial w(\mathbf{x}, \mathbf{p})}{\partial \mathbf{x}} \frac{\partial w(\mathbf{x}, \boldsymbol{\Delta}\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\boldsymbol{\Delta}\mathbf{p}=\mathbf{0}} \\ &= \nabla \overline{I}(w(\mathbf{x}, \mathbf{p})) \left.\frac{\partial w(\mathbf{x}, \boldsymbol{\Delta}\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}}\right|_{\boldsymbol{\Delta}\mathbf{p}=\mathbf{0}} \end{aligned} \tag{4.11}$$

where $\nabla \overline{I} = (\nabla_x \overline{I}, \nabla_y \overline{I})$ is the gradient of the image $\overline{I}$ (computed at the position $w(\mathbf{x}, \mathbf{p})$), while $\nabla \overline{I}(w(\mathbf{x}, \mathbf{p}))$ is the gradient of the warped image $\overline{I}(w(\mathbf{x}, \mathbf{p}))$, *i.e.* their derivatives with respect to the horizontal and vertical axes. The second term of the expression is the Jacobian of the warp function that links the displacement of each point $w(\mathbf{x}, \boldsymbol{\Delta}\mathbf{p})$ with the update $\boldsymbol{\Delta}\mathbf{p}$ computed for $\boldsymbol{\Delta}\mathbf{p} = \mathbf{0}$. This Jacobian matrix is computed as follows:

$$\frac{\partial w(\mathbf{x}, \boldsymbol{\Delta}\mathbf{p})}{\partial \boldsymbol{\Delta}\mathbf{p}} = \begin{bmatrix} \frac{\partial w_x}{\partial \boldsymbol{\Delta}\mathbf{p}} \\ \frac{\partial w_y}{\partial \boldsymbol{\Delta}\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \frac{\partial w_x}{\partial p_0} & \frac{\partial w_x}{\partial p_1} & \cdots & \frac{\partial w_x}{\partial p_n} \\ \frac{\partial w_y}{\partial p_0} & \frac{\partial w_y}{\partial p_1} & \cdots & \frac{\partial w_y}{\partial p_n} \end{bmatrix} \tag{4.12}$$

where n is the number of elements of the displacement update $\boldsymbol{\Delta}\mathbf{p}$. As an example, we provide the computation of the derivation of the affine and homography warps in Frame 5. Since this matrix is computed for $\boldsymbol{\Delta}\mathbf{p} = \mathbf{0}$, it is constant and can be precomputed for each point.

To illustrate the resulting gradient in a numerical example, we simply compute the mutual information of two identical images with respect to the horizontal translation t_x between them, and compute the corresponding gradient analytically (from the equation (4.7)) and numerically using the MI values: $\partial \mathrm{MI}(t_x)/\partial t_x \simeq (\mathrm{MI}(t_x + \delta t_x) - \mathrm{MI}(t_x - \delta t_x))/(2\delta t_x)$ with $\delta t_x = 0.1$px that we consider as ground truth. The resulting values are presented in Figure 4.3. The analytical and numerical values are similar. This formulation using B-splines results in a really smooth gradient suited to optimization problems.

To focus back on our tracking problem, we propose in Figure 4.4 a 2D translation estimation example. In this experiment, we compute the MI between two images $I^*(\mathbf{x})$ and $I(w(\mathbf{x}, \mathbf{t}))$ with respect to a translational motion $\mathbf{t} = (t_x, t_y)$, knowing that their exact alignment position is $\mathbf{t} = (0, 0)$. For every parameter $\mathbf{t}$, the gradient of MI is computed to verify if the steepest ascent provides a good estimation of the update $\boldsymbol{\Delta}\mathbf{p}$. As we represent in Figure 4.4(a), the steepest gradient method approximates the local shape of the function by a plane. Since the resulting updates correspond to 2D motions, we can represent them in the image space as a vector flow (see Figure 4.4(b)). In this figure, the mutual information value is represented in

Frame 5 Jacobian of the affine and homography warp functions.

Jacobian of the affine warp

If the formulation of the affine warp function, presented in section 1.2.2.3 page 18, is developed, it yields the following formulation:

$$w(\mathbf{x}, \mathbf{\Delta p}) = \begin{bmatrix} (1+\Delta p_0)x + \Delta p_2 y + \Delta p_4 \\ \Delta p_1 x + (1+\Delta p_3)y + \Delta p_5 \end{bmatrix}$$

If the two components of the resulting vector is differentiate with respect to the elements of the displacement parameter $\mathbf{\Delta p}$ using the equation (4.12), it yields:

$$\frac{\partial w(\mathbf{x}, \mathbf{\Delta p})}{\partial \mathbf{\Delta p}} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 \\ 0 & x & 0 & y & 0 & 1 \end{bmatrix}$$

For the affine warp, we can observe that there is no difference if the Jacobian of the warp is computed for $\mathbf{\Delta p} = \mathbf{0}$. In any case, this Jacobian matrix is only depending on the position of the point $\mathbf{x}$.

Jacobian of the homography warp

Similarly, if the expression of the homography warp function, defined in section 1.2.2.3 is developed, we obtain:

$$\begin{aligned} w(\mathbf{x}, \mathbf{\Delta p}) &= \frac{1}{\Delta p_6 x + \Delta p_7 y + 1} \begin{bmatrix} (1+\Delta p_0)x + \Delta p_2 y + \Delta p_4 \\ \Delta p_1 x + (1+\Delta p_3)y + \Delta p_5 \end{bmatrix} \\ &= \frac{1}{D} \begin{bmatrix} x'_a \\ y'_a \end{bmatrix} \end{aligned} \tag{4.13}$$

with:

$$x'_a = (1+\Delta p_0)x + \Delta p_2 y + \Delta p_4$$

$$y'_a = \Delta p_1 x + (1+\Delta p_3)y + \Delta p_5 \tag{4.14}$$

$$D = \Delta p_6 x + \Delta p_7 y + 1. \tag{4.15}$$

Developing the general Jacobian expression defined in equation (4.12), the Jacobian of the homography warp becomes:

$$\frac{\partial w(\mathbf{x}, \mathbf{\Delta p})}{\partial \mathbf{\Delta p}} = \frac{1}{D} \begin{bmatrix} x & 0 & y & 0 & 1 & 0 & -xx'_a & -yx'_a \\ 0 & x & 0 & y & 0 & 1 & -xy'_a & -yy'_a \end{bmatrix}$$

This matrix is originally depending on the value of the parameters $\mathbf{\Delta p}$. One of the advantages of the compositional approach is that this Jacobian is always computed for $\mathbf{\Delta p} = \mathbf{0}$. Therefore, it becomes only dependent on the position of the pixel $\mathbf{x}$:

$$\left.\frac{\partial w(\mathbf{x}, \mathbf{\Delta p})}{\partial \mathbf{\Delta p}}\right|_{\mathbf{\Delta p}=\mathbf{0}} = \begin{bmatrix} x & 0 & y & 0 & 1 & 0 & -x^2 & -xy \\ 0 & x & 0 & y & 0 & 1 & -xy & -y^2 \end{bmatrix}$$

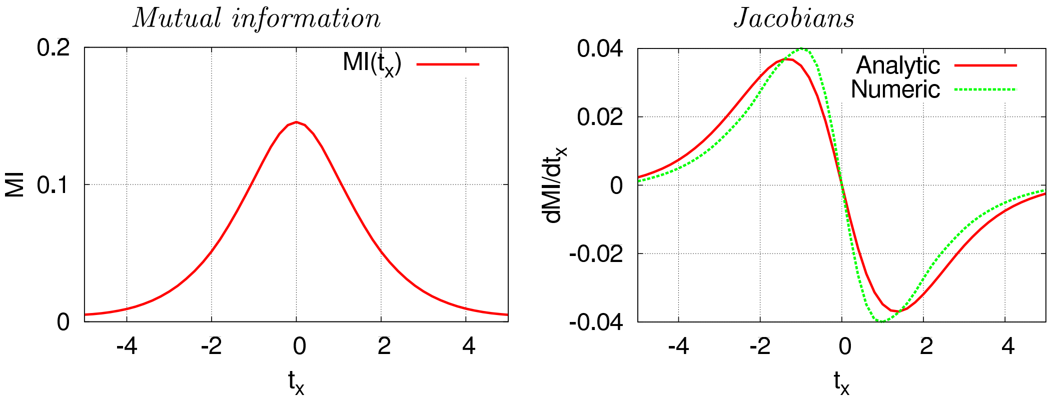


Figure 4.3: Mutual information and its gradient with respect to the horizontal translation.

the image space by isocontours, or contour lines, which are contours that join points with an equal function value. As expected, since the isocontour direction is the one with a null descent (the smallest descent), the resulting directions of the steepest descent are orthogonal to the isocontours (Figure 4.4(c)).

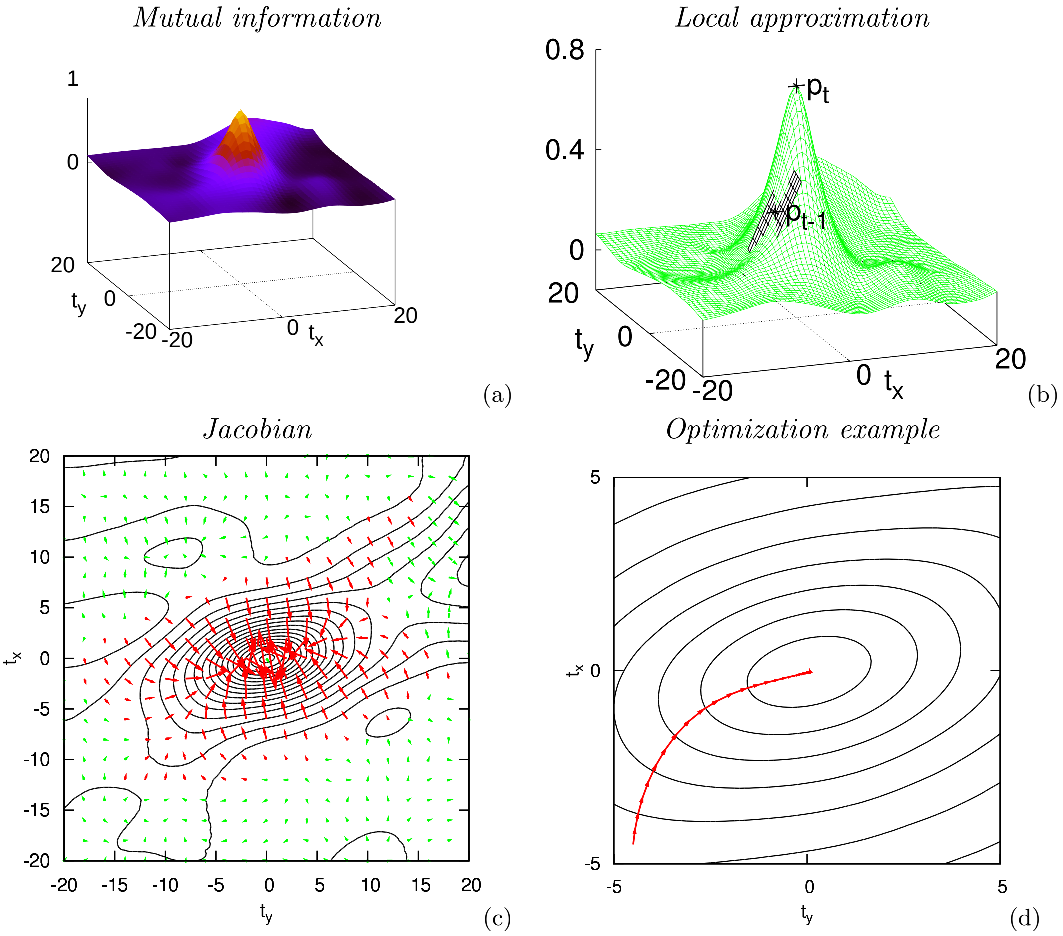


Figure 4.4: Steepest gradient descent. (a) Mutual information, (b) its local approximation (in black) at $(t_x, t_y) = (4.5, 4.5)$, (c) its gradient and (d) one example of optimization of mutual information. The gradients that make the optimization converge are represented in red arrows and the others in green.

In Figure 4.4(d), we represent one steepest descent optimization using a first guess $\mathbf{t}_0 = (4.5, 4.5)$. To best avoid the divergence of the optimization, we fix the step-size α to a small value. After 25 iterations, the optimization reaches an estimation of $\mathbf{t}$ within an error of 0.1 px. Since the gain α depends on the function to optimize a more clever computation of α is strongly recommended.

Gain computation: line-search method

The line-search approaches are dedicated to the estimate an ideal step-size α. The estimation is, in general, performed using an iterative method. Some of them, as the Wolfe's Rule [Press 1992], require to compute the derivative of the cost function for each temporary step-size α in the iterative process. In our context, the computation of the MI derivatives is time consuming and we prefer to consider the Brent's method [Press 1992]: a successive parabolic interpolation method, that only uses the cost function values.

Let us focus on one iteration of the maximization problem: once the direction is computed, the goal is to maximize the cost function with respect to the step-size α:

$$\widehat{\alpha} = \arg\max_{\alpha} \mathrm{MI}\left(I^*(\mathbf{x}), I(w(w(\mathbf{x}, \alpha\mathbf{G}), \mathbf{p}^k))\right)$$

The usual assumption of this method is that we can approximate the function shape along the direction of the update by a parabola. If we compute three values of the cost function using three different step-sizes, then, we can estimate this parabola and compute the step-size that maximizes it. Since the first estimation of the parabola is coarse, the first estimation of the step-size is, in general, not optimal and the method has to be iterated. Keeping the three best estimations of the minimum (the step-sizes that maximize the MI) among the four available step-sizes, the process is performed again until it reaches the convergence.

The successive iterations of the step-size computation method are represented in Figure 4.5(a) for the first iteration of the maximization. The estimation of only 4 parabolas are sufficient to reach an accurate estimation of the maximum along the direction given by the steepest gradient-descent.

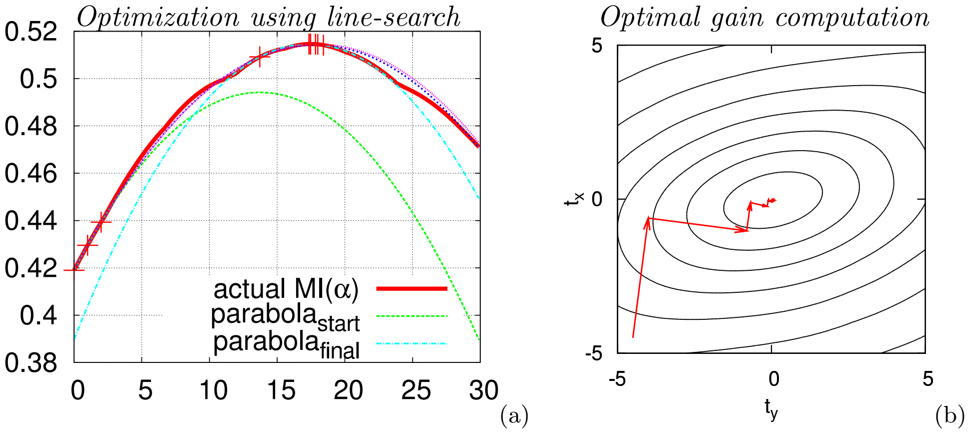


Figure 4.5: First order optimization of mutual information using a line-search algorithm. Brent's method is performed to compute the optimal gain of the first step.

Figure 4.5(b) shows the result of the steepest gradient-descent optimization method coupled with the Brent's method. The optimization with the line-search method gives good results on the translation estimation. After 8 iterations, the precision of the estimation is within 0.1 pixels. Nevertheless, at each step, the generated descent direction is orthogonal to the previous one. Indeed, if the line-search method is successfully performed, the update must reach the minimum of the function along its direction that is then tangent to the isocontour. The new steepest descent, which is defined as orthogonal to the isocontour, is therefore also orthogonal to the previous direction. Even with an efficient line-search method the steepest gradient-descent keeps a slow convergence rate.

4.1.1.2 Newton's method and MI Hessian matrix

Newton's method is one of the most popular optimization methods. It locally approximates the function by a parabola (as we presented in Frame 3). Most of the time, this approximation provides a good estimation of the function shape and computes an update that has both a good direction and a good amplitude. It is a second order optimization method, *i.e.* it is based on the second-order derivatives of the function called its Hessian matrix $\mathbf{H}$.

Optimization approach

Considering that, at the optimum of the cost function, its gradient is null, the Newton's method reformulates the problem of searching the optimum of the function, as finding the zero of its gradient. To find it, this method makes the assumption that the function's gradient is linear (it considers the local function's shape to be parabolic) around the current parameters $\mathbf{p}$. The equation of the cost function's derivative is approximated using a Taylor's expansion:

$$\begin{aligned}\frac{\partial \mathrm{MI}(\mathbf{p}\circ\mathbf{\Delta p})}{\partial \mathbf{\Delta p}} &= \left.\frac{\partial \mathrm{MI}(\mathbf{p}\circ\mathbf{\Delta p})}{\partial \mathbf{\Delta p}}\right|_{\mathbf{\Delta p}=\mathbf{0}} + \left.\frac{\partial^2 \mathrm{MI}(\mathbf{p}\circ\mathbf{\Delta p})}{\partial \mathbf{\Delta p}^2}\right|_{\mathbf{\Delta p}=\mathbf{0}} \mathbf{\Delta p} + \xi \\ &\simeq \mathbf{G} + \mathbf{H}\mathbf{\Delta p}\end{aligned}$$

where ξ represents the terms of higher order that are neglected. Since the goal is to reach a position where the gradient is null, the equation to solve is:

$$\frac{\partial \mathrm{MI}(\mathbf{p}\circ\mathbf{\Delta p})}{\partial \mathbf{\Delta p}} = \mathbf{0} \qquad \Longrightarrow \qquad \mathbf{G} + \mathbf{H}\mathbf{\Delta p} = \mathbf{0}$$

Then, the update can be estimated as:

$$\mathbf{\Delta p} = -\mathbf{H}^{-1}\mathbf{G}. \tag{4.16}$$

This approach typically corresponds to approximate the local shape of the function with a parabola and estimate the update $\mathbf{\Delta p}$ that leads to its optimum. More commonly, the update is regulated by a stepsize $\alpha \in [0,1]$ and lead to $\mathbf{\Delta p} = -\alpha\mathbf{H}^{-1}\mathbf{G}$.

Mutual information Hessian computation

The Hessian matrix of mutual information is defined as the derivative of the gradient matrix. Recalling that the gradient $\mathbf{G}$ is:

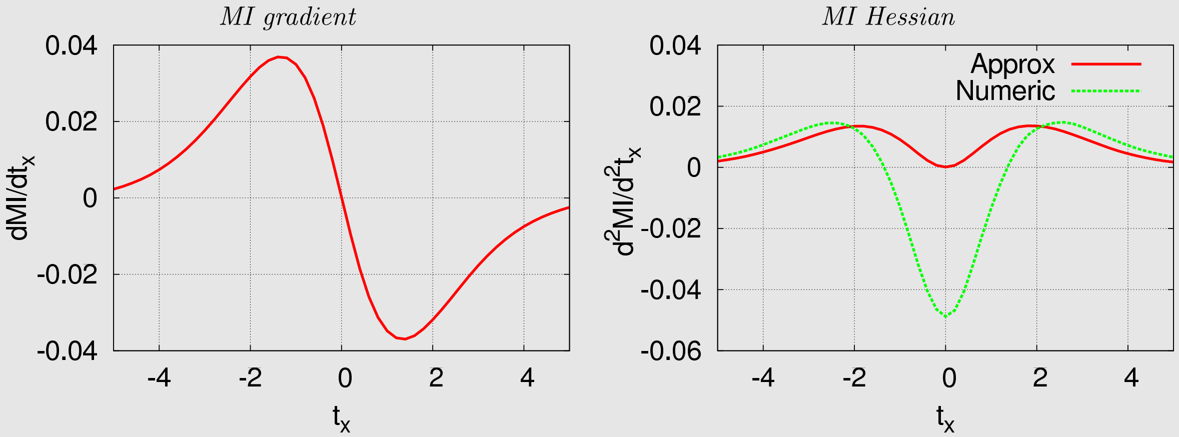
$$\mathbf{G} = \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_{I^*}}\right)$$

If we apply the derivative chain rules, we find the following expression:

$$\mathbf{H} = \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}}^{\top} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_{I^*}}\right) + \frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} \left(1 + \log \frac{p_{II^*}}{p_{I^*}}\right) \tag{4.17}$$

By analogy with the Gauss-Newton approach that will be detailed later, the second derivatives are usually neglected in the Hessian matrix computation [Thévenaz 2000, Dowson 2006,

Frame 6 Why the Hessian matrix must not be approximated?



It is common to find the Hessian matrix of MI given in equation (4.17) approximated by the following expression [Thévenaz 2000][Dowson 2006]:

$$\mathbf{H} \quad \simeq \quad \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}}^\top \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_{I^*}} \right) \tag{4.18}$$

where the second order derivative of the joint probability has been neglected (see equation (4.17)). The approximation is inspired from the one that is made in the Gauss-Newton's method for the resolution of the SSD minimization, where we assume that the neglected terms are null at the convergence. Considering the properties of the marginal probability and joint probability in the MI computation, we have:

$$p_{I^*}(j) = \sum_i p_{II^*}(i,j) \qquad \text{with:} \qquad p_{II^*}(i,j) > 0 \qquad \text{and} \qquad p_{I^*}(j) > 0$$

Therefore, it is clear that:

$$\forall (i,j) \in [0, N_c]^2, \; p_{I^*}(j) > p_{II^*}(i,j) \qquad \Longleftrightarrow \qquad 1/p_{II^*}(i,j) - 1/p_{I^*}(j) > 0$$

Since $\frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}}^\top \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}}$ is a positive matrix then the final Hessian matrix given by equation (4.18) is positive. The goal is to maximize MI, therefore the Hessian matrix at the optimum must be negative, so that the approximation is wrong at the optimum.

As an example, the evaluation of the approximation has been performed on the horizontal translation example previously used to illustrate the gradient computation. The top figures show the MI gradient and its Hessian using the approximation overlaid with the Hessian numerically computed with the gradient using: $\mathbf{H}(t_x) \simeq (\mathbf{G}(t_x + \delta t_x) - \mathbf{G}(t_x - \delta t_x))/(2\delta t_x)$. The comparison illustrates the above mentioned problem: the approximated Hessian is positive although it should be negative near the optimum, where MI is concave. The common approximation of equation (4.18) is thus not suited for the optimization of MI.

Dowson 2008]. Let us call this method, the Hessian Approximation (HA) method. In our approach we compute the Hessian matrix using the full expression, which is, in our point of view, required to obtain an accurate and robust optimization method. We give a detailed explanation of why $\mathbf{H}$ must not be approximated in Frame 6.

The expression of the second order derivative of the joint probability and its computation are similar to those of the first order derivative:

$$\frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} = \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \frac{\partial^2 \phi\left(i - \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2} \phi\left(j - \overline{I^*}(\mathbf{x})\right). \tag{4.19}$$

Since ϕ has been chosen to be differentiable twice, the derivative of the ϕ function is given by:

$$\begin{aligned} \frac{\partial^2 \phi\left(i - \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2} &= \frac{\partial}{\partial \mathbf{\Delta p}} \left(- \frac{\partial \phi}{\partial i} \frac{\partial \overline{I}(w(\mathbf{x}, \mathbf{p} \circ \mathbf{\Delta p}))}{\partial \mathbf{\Delta p}} \right) \\ &= \frac{\partial^2 \phi}{\partial i^2} \frac{\partial \overline{I}}{\partial \mathbf{\Delta p}}^{\top} \frac{\partial \overline{I}}{\partial \mathbf{\Delta p}} - \frac{\partial \phi}{\partial i} \frac{\partial^2 \overline{I}}{\partial \mathbf{\Delta p}^2}. \end{aligned} \tag{4.20}$$

with:

$$\begin{aligned} \frac{\partial^2 \overline{I}}{\partial \mathbf{\Delta p}^2} &= \frac{\partial}{\partial \mathbf{\Delta p}} \left(\nabla \overline{I}(w(\mathbf{x}, \mathbf{p})) \, \frac{\partial w(\mathbf{x}, \mathbf{\Delta p})}{\partial \mathbf{\Delta p}} \right) \\ &= \frac{\partial w}{\partial \mathbf{\Delta p}}^{\top} \nabla^2 \overline{I}(w(\mathbf{x}, \mathbf{p})) \frac{\partial w}{\partial \mathbf{\Delta p}} + \nabla_x \overline{I}(w(\mathbf{x}, \mathbf{p})) \frac{\partial^2 w_x}{\partial \mathbf{\Delta p}^2} + \nabla_y \overline{I}(w(\mathbf{x}, \mathbf{p})) \frac{\partial^2 w_y}{\partial \mathbf{\Delta p}^2} \end{aligned} \tag{4.21}$$

where $\nabla^2 \overline{I}(w(\mathbf{x}, \mathbf{p}))$ is the Hessian of the warped image $\overline{I}(w(\mathbf{x}, \mathbf{p}))$ with respect to the horizontal and vertical axes and $\frac{\partial^2 w}{\partial \mathbf{\Delta p}^2}$ is the Hessian of the warp function. For instance, $\frac{\partial^2 w_x}{\partial \mathbf{\Delta p}^2}$ is generally written:

$$\frac{\partial^2 w_x}{\partial \mathbf{\Delta p}^2} = \begin{bmatrix} \frac{\partial^2 w_x}{\partial p_0^2} & \frac{\partial^2 w_x}{\partial p_0 \partial p_1} & \cdots & \frac{\partial^2 w_x}{\partial p_0 \partial p_n} \\ \frac{\partial^2 w_x}{\partial p_0 \partial p_1} & \frac{\partial^2 w_x}{\partial p_1 \partial p_1} & \cdots & \frac{\partial^2 w_x}{\partial p_0 \partial p_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 w_x}{\partial p_0 \partial n} & \frac{\partial^2 w_x}{\partial p_0 \partial p_n} & \cdots & \frac{\partial^2 w_x}{\partial p_n^2} \end{bmatrix} \tag{4.22}$$

In Frame 7, we developed the computation of the Hessian matrix $\frac{\partial^2 w}{\partial \mathbf{\Delta p}^2}$ of the homography warp. To evaluate the quality of the Hessian matrix formulation, it has been computed on the simple horizontal translation (the one that we presented in the previous section). In Figure 4.6, we compare the Hessian computed with the analytical formulation and the numerical values obtained with $\mathbf{H}(t_x) \simeq (\mathbf{G}(t_x + \delta t_x) - \mathbf{G}(t_x - \delta t_x))/(2\delta t_x)$. The values of the Hessian computed with this analytical formulation are close to the numerical value and second they are smooth.

One geometrical interpretation of the Newton's method is represented in Figure 4.7(b) for the translation estimation already considered in the previous paragraph. Given a first coarse estimation of the translation, the tangent line of the gradient at the current guess is estimated using the value of the Hessian. Newton's method assumes that a better estimation of the zero is then found at the intersection between the tangent line and the horizontal axis. The corresponding interpretation can be made at the cost function level: given a coarse guess of the optimum of the cost function, if the function is approximated by a parabola using a Taylor's expansion, then a better estimation of the optimum can be found at the optimum of the parabola (see Figure 4.7(a)). In this 1D example, the Newton's method seems ideal when the first guess is near convergence, *i.e.* when it is situated in the parabolic shape of the maximum.

Frame 7 Hessian of the affine and homography warp functions.

Affine warp

As we showed in the Frame 5, the Jacobian matrix of the affine warp function does not depend on the displacement parameter $\mathbf{\Delta p}$. Its Hessian (the derivative of its Jacobian) with respect to $\mathbf{\Delta p}$ is therefore null. Remark that this does not cause the approximation of the second terms of the Hessian matrix to be justified, indeed many terms of the equation (4.20) are still not null.

Homography warp

Using the same notation as in the Frame 5 and the general equation of the Hessian matrix defined in equation (4.22), the Hessian matrix of the homography warp is computed as:

$$\frac{\partial^2 w_x}{\partial \mathbf{\Delta p}^2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & xy & y^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x & y \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x^2 & 0 & xy & 0 & x & 0 & -x^2x'_a & -xyx'_a \\ xy & 0 & y^2 & 0 & y & 0 & -xyx'_a & -y^2x'_a \end{bmatrix} \text{ and } \frac{\partial^2 w_y}{\partial \mathbf{\Delta p}^2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & xy & y^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x & y \\ 0 & x^2 & 0 & xy & 0 & x & -x^2y'_a & -xyy'_a \\ 0 & xy & 0 & y^2 & 0 & y & -xyy'_a & -y^2y'_a \end{bmatrix}$$

As it was the case for the computation of the Jacobian, we can observe that in the compositional formulation, since the Hessian matrix is computed for $\mathbf{\Delta p} = \mathbf{0}$, it becomes constant for each point and it can be precomputed.

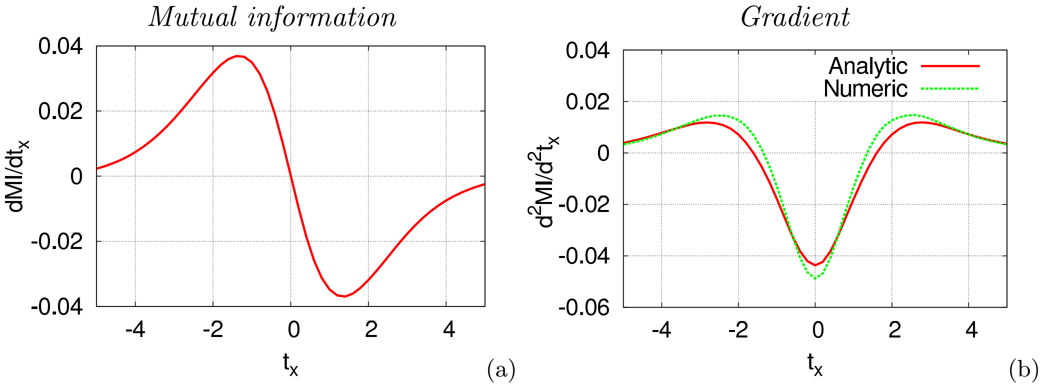


Figure 4.6: Mutual information derivatives: the analytic computation of $\mathbf{H}$ is smooth and is similar to the numeric values obtained using the gradient as: $\mathbf{H}(t_x) \simeq (\mathbf{G}(t_x + \delta t_x) - \mathbf{G}(t_x - \delta t_x))/(2\delta t_x)$.

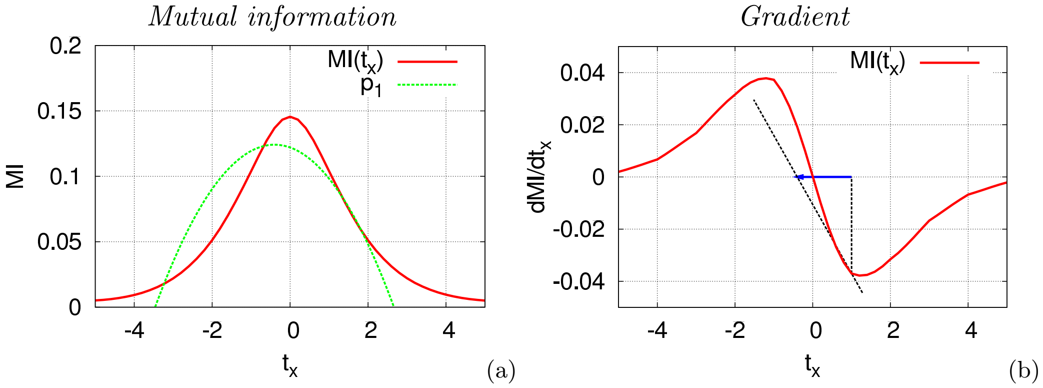


Figure 4.7: Mutual information and its gradient with respect to the horizontal translation and geometrical interpretation of Newton's method: with a first guess $t_x^0 = 1$ near the optimum $\widehat{t_x} = 0$, the optimization converges.

We evaluate the Newton's method with the 2D translation estimation that was previously used in the steepest gradient-descent method. As in the 1D problem, the Newton's method remains suited to the estimation of the displacement parameters when a first guess, close to the real optimum position, is available. The estimation of the parabola is close to the parabolic shape of the function near the optimum (see figure 4.8(a)) and, therefore, the resulting updates and optimization are effective (see figure 4.8(c)). In 3 iterations, the algorithm reaches an estimation within a precision of 0.1 px.

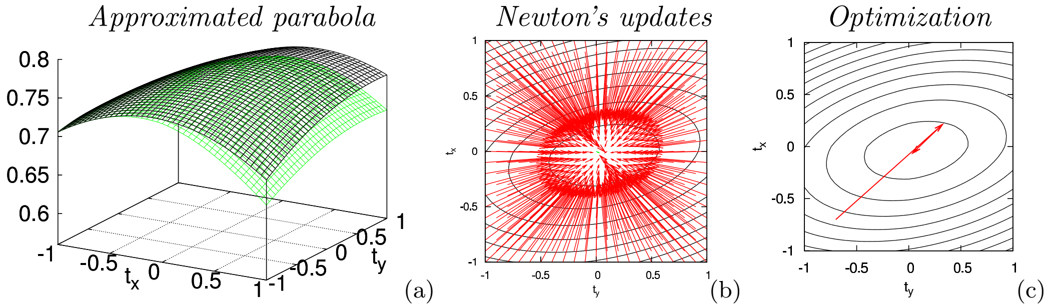


Figure 4.8: Optimization of MI using the Newton's method near the convergence. (a) shows the local approximation (in black) of MI at $(t_x, t_y) = (0.7, 0.7)$, (b) shows all the resulting updates converging to the optimum. (c) Since the first guess is near the optimum of MI, the optimization converges.

Quasi-concave problem

The Newton's method approximates the cost function's shape with a parabola to find a new estimation. This method is highly efficient to evaluate the displacement parameters, if a first guess, close to the optimum, is available. Nevertheless, if this guess is far from it, the estimated parabola is not a good approximation of the function at convergence and it causes the method to diverge.

A geometrical interpretation is given in the case of a 1D translation estimation problem. In the previous case, when the first guess was near the optimum (see Figure 4.7), the Hessian was negative (MI is concave, see Figure 4.6 for the Hessian values), the tangent line was directed toward the optimum and the algorithm converged. Now if the first guess is far from it, the second-order derivative is positive (MI is convex), so the tangent line is directed toward a translation further from the optimum and the method diverges as it is shown in Figure 4.9.

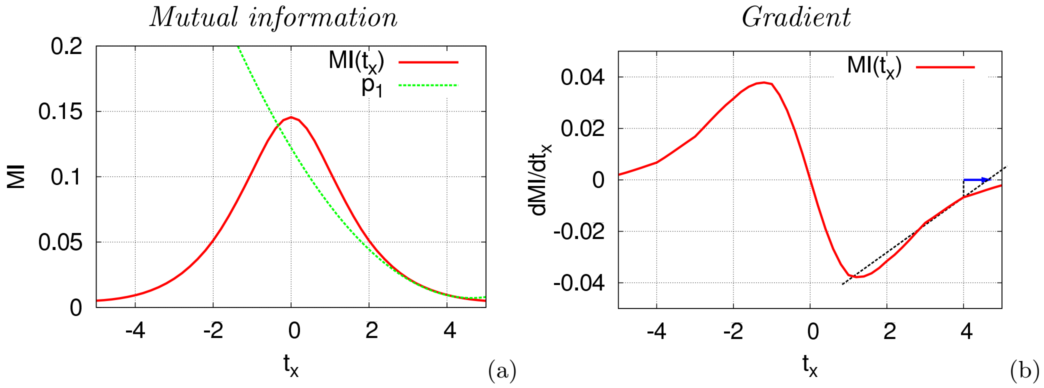


Figure 4.9: Mutual information and its gradient with respect to the horizontal translation and the geometrical interpretation of Newton's method: with a first guess $t_x^0 = 4$ far from the optimum $\widehat{t_x} = 0$, the optimization diverges.

If we evaluate the approach with the 2D translation estimation that was previously used we see on figure 4.10 that it is exactly the same situation as in the 1D problem. If the first guess is far from the convergence then the local approximation of MI is not a concave parabola and thus, the updates make the optimization diverge.

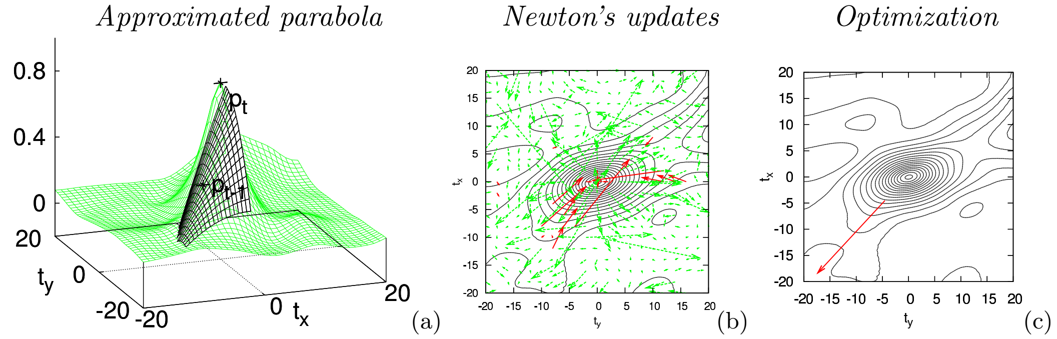

Approximated parabola *Newton's updates* *Optimization*

Figure 4.10: Optimization of the MI using the Newton's method far from the convergence. (a) shows the local approximation of MI at $(t_x, t_y) = (4.5, 4.5)$, this approximation is right but its optimum is further from the optimum of MI. (b) shows the resulting updates, red vectors represents the converging updates, the convergence domain is in fact small. (c) since the first guess of the optimization is out of the concave domain, the algorithm diverges.

4.1.1.3 Toward an efficient optimization approach

The convergence domain of the optimization using the Newton's method with MI is limited compared to the quality of its function shape. For a cost function that has a similar shape, the convergence domain using the SSD and the Gauss-Newton optimization is much larger! In this section, we propose a new Newton's like approach that solves the quasi-concavity problem and has a large convergence domain.

Analogy with the SSD optimization

To introduce our solution, let us first study the optimization of the SSD. This problem is similar to the problem of maximizing the MI. The SSD function is quasi-convex, as MI is quasi-concave. If we consider the opposite of SSD, the two problems become equivalent. If we seek to minimize the SSD using the Newton's method with a first guess out of its convex domain, we will face exactly the same problem that we faced in section 4.1.1.2 with MI: the Hessian matrix will not be positive and it will make the optimization diverge.

However this problem is rarely discussed. Due to some second derivative terms, the exact Hessian matrix of SSD is difficult and expensive to compute (see the Frame 8 for the details). The optimization is then always performed using the Gauss-Newton approach where these terms are neglected. This approximation is not only good for its simplicity, indeed the resulting approximated Hessian matrix is (in this minimization problem) always a positive matrix. The Gauss-Newton gives therefore the solution to avoid the problem due to the quasi-convexity of the function.

In Figure 4.11, we represent the values of the opposite SSD with respect to one translation t_x between two images I and I^*, and compute the resulting gradient and Hessian matrices using the Newton and Gauss-Newton approaches. We represented the opposite of the function to show the similitude with MI. Using a Newton's like approach, the update will be directed toward the optimum only where the gradient is also oriented toward the optimum and where the Hessian is negative. If the exact Hessian matrix is used (Newton's method), the convergence domain is small (purple area), while, if the Gauss-Newton method is used, the convergence domain is large (blue domain).

As we can see, the MI and SSD functions and its derivatives are really similar. Therefore, a Gauss-Newton's like approach for MI would improve its robustness as Gauss-Newton improves the robustness of SSD. But as we have seen previously, neglecting the second derivatives of the Hessian matrix has a completely different effect whether the cost function is MI or SSD.

Frame 8 Newton and Gauss-Newton optimization of the SSD function.

To introduce our solution, let us first study the optimization of the SSD. In Figure 4.11, we represent the values of the SSD with respect to one translation $\mathbf{p}$ between two images I and I^*:

$$SSD(\mathbf{p}) = \sum_{\mathbf{x}} (I(w(\mathbf{x}, \mathbf{p})) - I^*(\mathbf{x}))^2 \tag{4.23}$$

The gradient of the SSD is computed using the compositional formulation as follows:

$$\mathbf{G} = \sum_{\mathbf{x}} \frac{\partial I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}}^\top (I(w(\mathbf{x}, \mathbf{p})) - I^*(\mathbf{x})) \tag{4.24}$$

And the Hessian matrix have been computed using the exact formulation $\mathbf{H}_N$ (the matrix used in the Newton's method) and the one computed as in the Gauss-Newton approach $\mathbf{H}_{GN}$, where the second order derivative are neglected:

$$\begin{aligned} \mathbf{H}_N &= \sum_{\mathbf{x}} \frac{\partial I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}}^\top \frac{\partial I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}} \qquad (4.25) \\ &+ \sum_{\mathbf{x}} \frac{\partial^2 I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}^2} (I(w(\mathbf{x}, \mathbf{p})) - I^*(\mathbf{x})) \\ \mathbf{H}_{GN} &\simeq \sum_{\mathbf{x}} \frac{\partial I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}}^\top \frac{\partial I(w(\mathbf{x}, \mathbf{p} \circ \boldsymbol{\Delta}\mathbf{p}))}{\partial \boldsymbol{\Delta}\mathbf{p}} \qquad (4.26) \end{aligned}$$

We can see on the resulting values, that dropping the second order derivatives makes the approximated Hessian matrix quasi-constant. This observation is easily verified on the equation (4.26) in the 1D problem: since the derivation of the translational motion is constant for every pixel $\mathbf{x}$, the resulting Hessian is only depending on the summation of the image gradient over the region of interest. A small motion of the region of interest will only change this summation due to the pixels that enter or leave the region, therefore, the variation of $\mathbf{H}_{GN}$ remains small.

In our compositional formulation, since the Jacobian of the warp remains constant for each pixel and is only slightly different from one pixel to another one, this observation can be generalized to our more complex parametric motions. Therefore, neglecting the second-order terms of the Hessian matrix is similar to estimate the Hessian matrix after convergence that remains by definition convex. In [Baker 2004], the authors interpreted the improvements of the Gauss-Newton methods by the fact that the noise increasement in estimating the second derivatives of the image outweigh the increase of sophistication in the algorithm. Comparing the analytical Hessian and its numerical values (computed directly using the gradient values) in figure 4.11 that are almost equal, makes us search a new interpretation. In our point of view, the approximation allows to avoid the problem caused by the concavity of the function, when the first guess is out of the convex domain of the SSD. In this case, the approximated Hessian is still positive and the update is directed toward the optimum, while the exact Hessian matrix is not positive and makes the Newton's method diverge.

MI optimization

To solve the problem concerning the MI optimization, our goal is to follow the same principle than the Gauss-Newton approach. As we presented in Frame 6, we recall that, simply neglecting the second derivatives of the Hessian of MI does not provide the solution of our problem. We build our approach on two observations: first, the Newton's method is converging when the first guess is near the optimum, second, the Gauss-Newton approach gives an approximation of the Hessian matrix of the SSD at the optimum (see the Frame 8).

Our solution is therefore simply to use the Hessian matrix $\mathbf{H}^*$ of MI computed at the optimum in a Newton's like approach. Let us note this approach HC for Hessian after convergence. Since this matrix is by definition negative (it corresponds to the maximum of MI), the quasi-concavity of MI is not a problem anymore. And since this Hessian matrix depicts the exact variability of MI at the optimum, it allows an optimal convergence as soon as the current guess is near the optimum, and should provide an accurate estimation of the displacement parameters.

At convergence, the expression of the Hessian matrix is unchanged. To compute it, we need to estimate the joint probability and its derivatives at the optimum $\mathbf{p}^*$:

$$\begin{aligned} p_{II^*} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}^*))\right)\phi\left(j - \overline{I^*}(\mathbf{x})\right) & (4.27)\\ \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \frac{\partial \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}^*\circ\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}}\phi\left(j - \overline{I^*}(\mathbf{x})\right) & (4.28)\\ \frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \frac{\partial^2 \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}^*\circ\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2}\phi\left(j - \overline{I^*}(\mathbf{x})\right). & (4.29)\end{aligned}$$

To compute these expressions, we have to estimate the value of the current image I at the pixel locations $\mathbf{x}$ warped using the optimal parameters $\mathbf{p}^*$. To do so, we can assume that the intensities of the reference and current warped images remain similar and write:

$$\overline{I}(w(\mathbf{x},\mathbf{p})) \simeq \overline{I^*}(\mathbf{x}) \qquad (4.30)$$

This is typically right in nominal conditions, in the case of a perfect alignment. Therefore, we can estimate the joint probability and its derivatives as:

$$\begin{aligned} p_{II^*} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \phi\left(i - \overline{I^*}(\mathbf{x})\right)\phi\left(j - \overline{I^*}(\mathbf{x})\right) & (4.31)\\ \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \frac{\partial \phi\left(i - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}}\phi\left(j - \overline{I^*}(\mathbf{x})\right) & (4.32)\\ \frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} &= \frac{1}{N_{\mathbf{x}}}\sum_{\mathbf{x}} \frac{\partial^2 \phi\left(i - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2}\phi\left(j - \overline{I^*}(\mathbf{x})\right). & (4.33)\end{aligned}$$

Since the reference image I^* is constant and the gradient and Hessian matrices are computed for $\mathbf{\Delta p} = \mathbf{0}$, these three expressions are constant. The Hessian matrix $\mathbf{H}^*$ is therefore also constant and can be precomputed. We recall that the gradient and Hessian matrices are computed for $\mathbf{\Delta p} = \mathbf{0}$ due to the compositional formulation. In an additional formulation, the derivatives would have to be computed for $\mathbf{p} = \mathbf{p}^*$ that is unknown. This approach is, therefore, not suited for an additional formulation.

In Figure 4.12, we show the estimation of a 2D motion estimated using this approach (with the same example that was used with the steepest gradient-descent method and the Newton's method). We can observe that the local approximation of MI is given by the curvature of MI

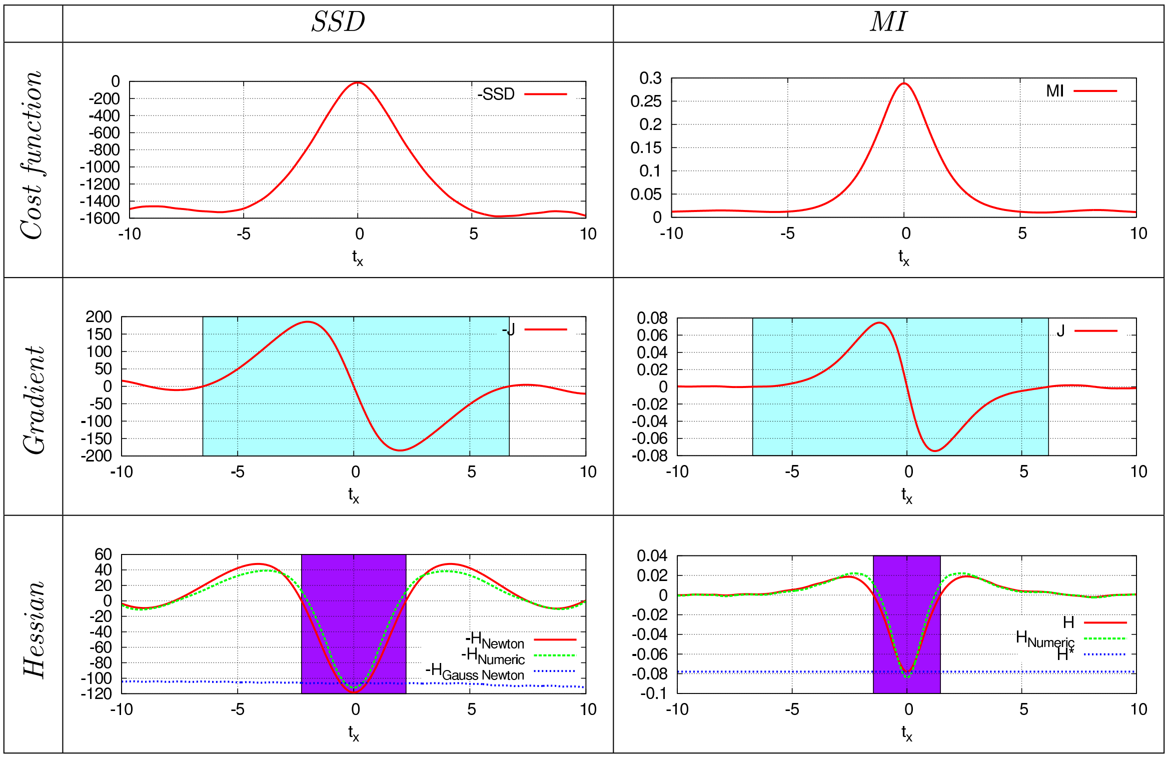
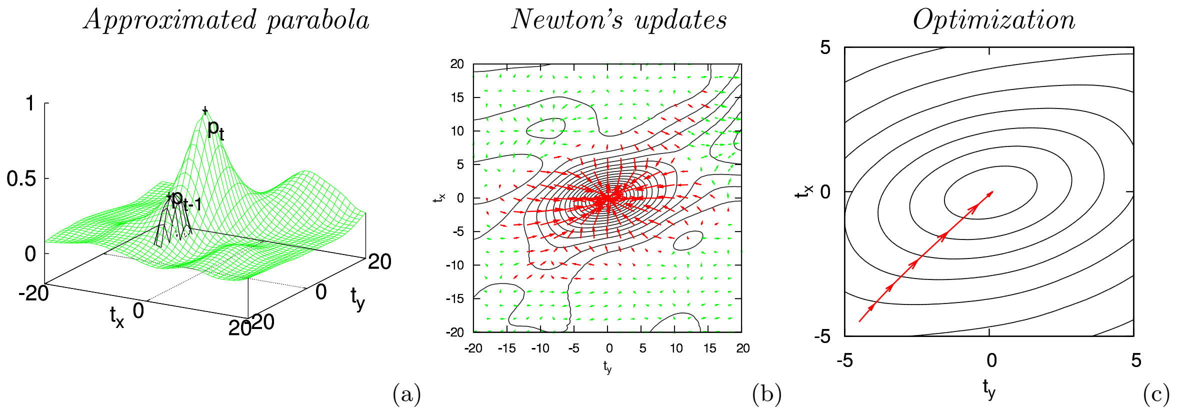


Figure 4.11: SSD, MI and their derivatives with respect to one translation (px). The purple area is the convergence domain using a classical Newton's method, the blue one is the convergence domain of a Gradient descent method. The proposed method keeps the wider convergence domain of the gradient's method in blue.

at the optimum. The estimation of the resulting update is performed for a large set of initial guesses. There is no problem due to the quasi-concave shape of the function, as it was the case with the Newton's method in Figure 4.10. The convergence domain becomes as large as the one obtained with the steepest gradient-descent method, but the direction is better and there is no problem of choosing an update step-size. The optimization presented in Figure 4.12(c) with a first guess $(t_x, t_y) = (4.5, 4.5)$ is now converging and after 6 iterations it reaches an error within 0.1 px. We can see that all the steps of the optimization are directed toward the optimum.



(a) (b) (c)

Figure 4.12: Optimization of MI using the Hessian $\mathbf{H}^*$ far from the convergence. Most of the resulting updates are converging to the optimum since the approximated parabola remains always concave.

4.1.2 Empirical convergence analysis

The previous section has presented several optimization approaches applied to MI and we presented the theoretical advantages of the proposed HC approach that uses the estimation of the Hessian matrix computed at the optimum. In this section, we realize a set of experiments to compare the performances of the optimization methods in a practical homography estimation problem.

The goal of this set of experiments is to estimate the position $\mathbf{p}^*$ of a template of 100×100 pixels in an image knowing a first guess $\mathbf{p}^0$. We study the effect of the initial positioning error between $\mathbf{p}^*$ and $\mathbf{p}^0$ on the different registration approaches. We define this positioning error $err(\mathbf{p})$ by the RMS distance in pixels between the ground truth position of M reference points $\mathbf{x}^*_m = w(\mathbf{x}_m, \mathbf{p}^*)$ and their positions computed with the current parameters $\mathbf{p}$. We simple choose the reference points as the 4 corners of the template. The analytic formulation of the error is given by:

$$err(\mathbf{p}) = \sum_{m=1}^{M} \|\mathbf{x}^*_m - w(\mathbf{x}_m, \mathbf{p})\| \tag{4.34}$$

In Figure 4.13, we show the ground truth position of the template and some initial positions that were used to begin the optimization, depending on the initial error $err(\mathbf{p}^0)$. For each initial error $err(\mathbf{p}^0)$ from 1 to 30, 500 initial parameters $\mathbf{p}^0$ are randomly generated. For every initial parameters $\mathbf{p}^0$, all the optimization approaches are performed. We consider that an optimization converges, if, after 250 iterations, the positioning error $err(\mathbf{p}^{250})$ is below 0.5 pixels (px).

To compare the behavior of the optimization, we used the following performance metrics:

- The frequency of convergence: for each initial positioning error and method, we compute over 500 experiments the proportion of them that converge.

- The average convergence rate: we compute the average positioning error against the number of iterations. Since the goal is to have a fair comparison between the approaches, only the experiments that converge for all the optimization approaches are considered.

- The average number of iterations required to reach the optimum, that is also computed considering the experiments where all the approaches converge.

- The average residue, *i.e.* the average remaining positioning error of the converging experiments after the 250 iterations.

- The average computation time per iteration of the optimization methods.

We recall that, since the displacement parameters are strongly correlated, the steepest gradient descent approach is not suited for the estimation of an homography. Therefore, it converges so rarely on the set of experiments that plotting the results does not make sense and it would bias the results of the other approaches. We consider the following optimization approaches:

- the Newton's method: the Hessian matrix is computed without approximation.

- the HA method: the second derivatives of the Hessian matrix are neglected [Dowson 2006].

- the HC method: the update is computed using the exact Hessian matrix $\mathbf{H}^*$ estimated at the optimum.

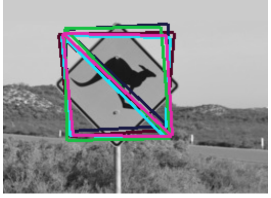
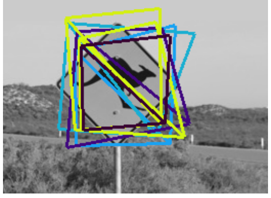
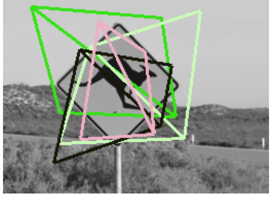
Ground truth	*Initial positions*		
	$err = 5$px	$err = 15$px	$err = 25$px
			

Figure 4.13: Empirical convergence analysis of the optimization methods. Illustration of some initial parameters guesses compared to the ground truth.

4.1.2.1 Evaluation without line-search

First we begin to evaluate the performance of the different optimization approaches without line-search method. As presented in Frame 6, the Hessian matrix computed using the approximation proposed in [Thévenaz 2000, Dowson 2006, Dowson 2008] causes the optimization to diverge if a line-search method is not considered. Therefore, no experiment using this optimization approach converged, so we do not consider or plot the corresponding results in this section.

The results concerning the Newton's approach and the Newton's like approach using $\mathbf{H}^*$ are reported in the Figure 4.14, where we plot the convergence frequency, (average) convergence rate and (average) number of iterations. In the Figure 4.15, we give the computation time and remaining positioning error. The main differences are in terms of convergence frequency and computation time. We can observe that the frequency of convergence of the Newton's method decreases as soon as the initial error becomes larger than 3 px, while the frequency of convergence of the HC approach begins to decrease for $err(\mathbf{p}_0) > 15$. For a computation time that is half the computation time of the Newton's method, the advantages of the HC approach are evident.

4.1.2.2 Evaluation with a line-search method

In this section, the optimization was performed using a Brent method, as defined in section 4.1.1.1. Since the issue raised by the approximation of the second derivative of the Hessian matrix (the HA method) can be solved by a line-search method, this optimization approach converges and we can compare its performances with the other optimization methods.

We can observe in the Figure 4.14 that the convergence frequency of the HA method becomes as high as the HC approach. The convergence domain of the exact Newton's method remains small. Concerning the number of iterations, we see that the Newton's method requires as many iterations as the HC approach, while the HA approach requires a larger number of iterations. Since both the HC and HA methods only require to compute the gradient matrix of the joint histogram, they have also a similar computation time. For a lesser amount of iterations and a smaller residue, the HC approach is, in general, more adapted to the estimation of the homography.

4.1.3 Conclusion

The approach that is proposed in this section is intuitive, it has a large convergence domain and provides an accurate estimation of the optimum. Since the Hessian matrix is computed only one time, it is not necessary to improve the efficiency of its computation. Nevertheless the computation of the gradient matrix still requires a large computation time to evaluate the current image gradients and fill the joint histogram and joint histogram derivatives. In the next

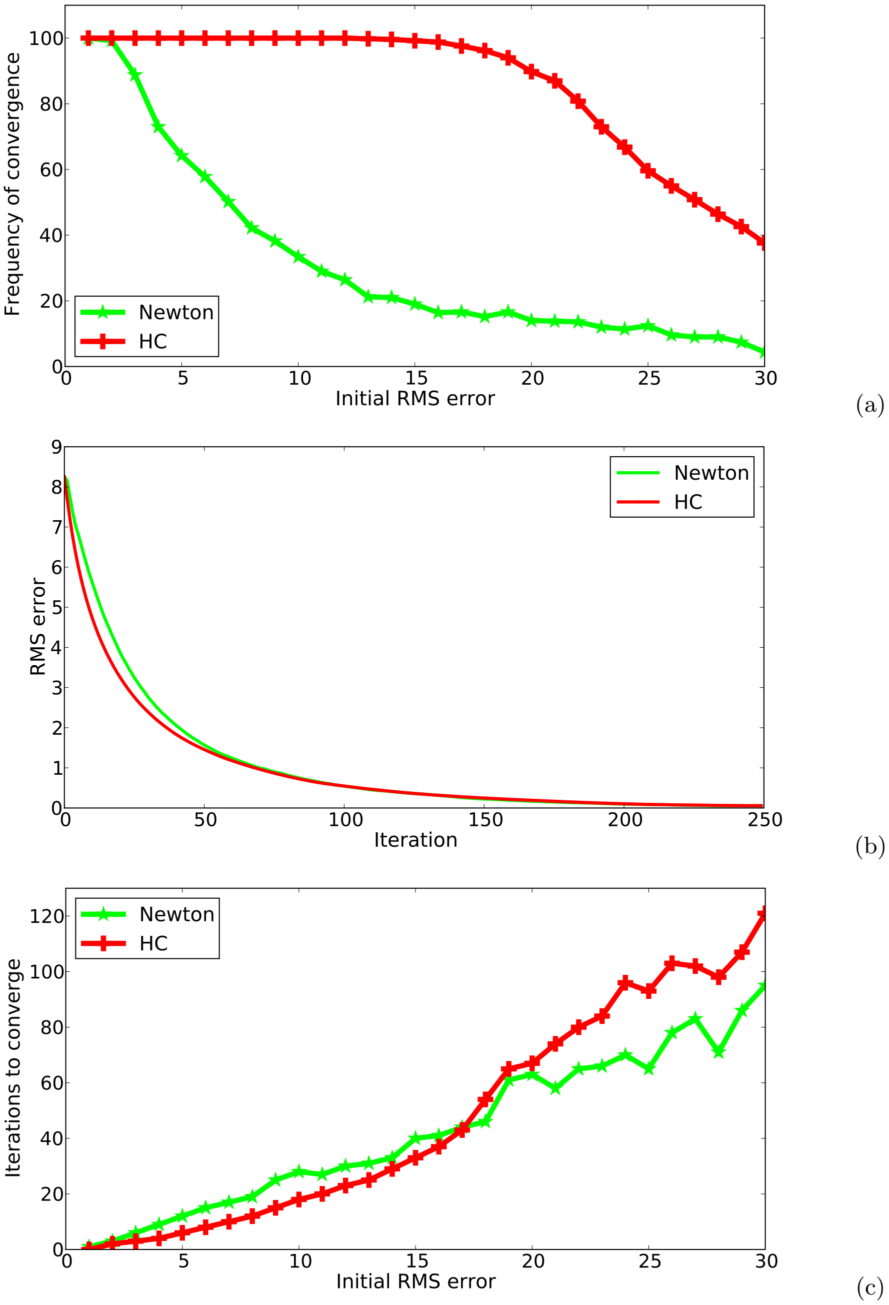


Figure 4.14: Empirical convergence analysis of the optimization methods. The convergence frequency of the method using $\mathbf{H}^*$ (in red) is far better than the real Newton's approach, while the convergence rate and number of iterations are similar.

	Newton	HA	HC
Time/iteration (ms)	26.4	10	10
Residue (px)	0.06	—	0.06

Figure 4.15: Average time in ms required to compute one update $\mathbf{\Delta p}$ without line-search and remaining positioning error after 250 iterations.

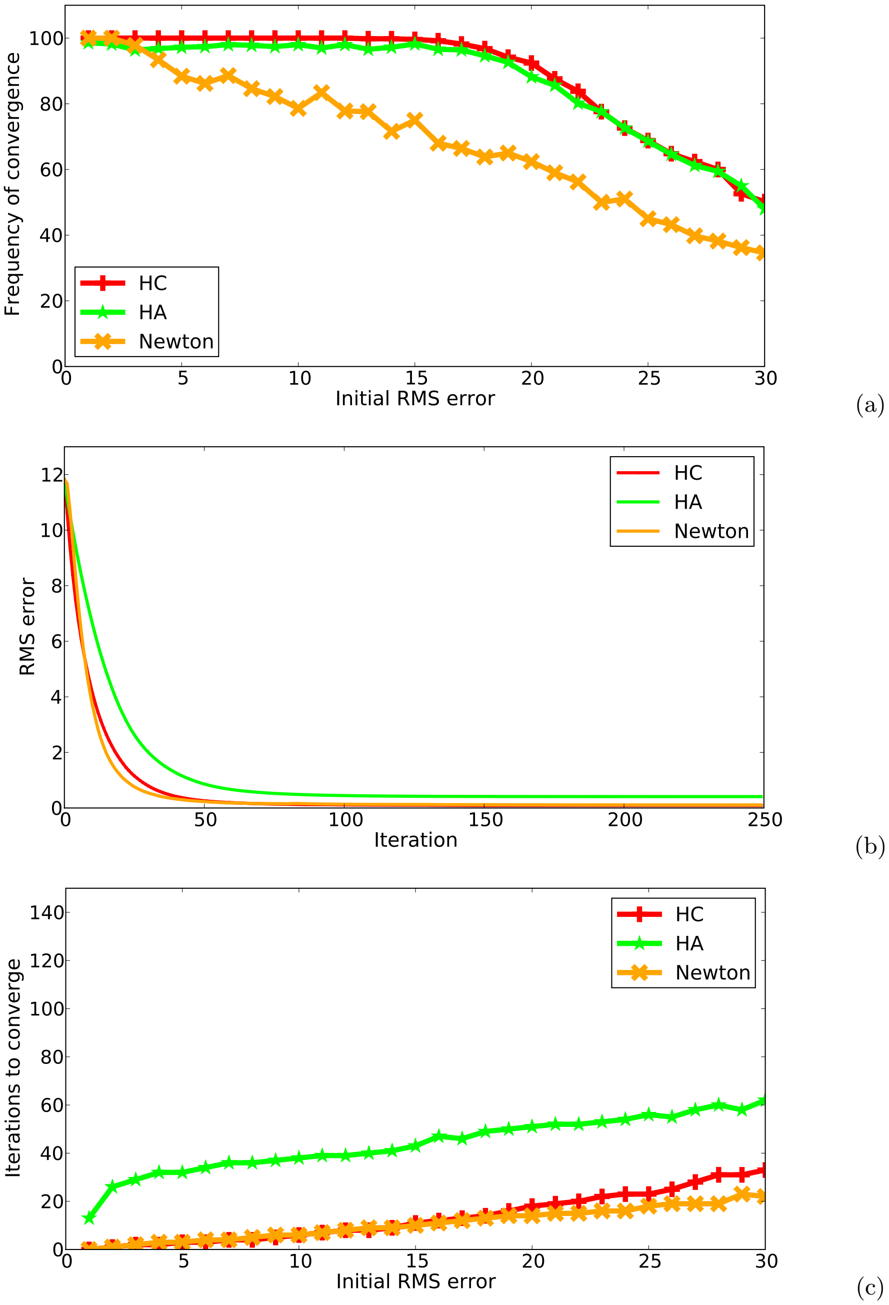


Figure 4.16: Empirical convergence analysis of the optimization methods with the Brent line-search method. The convergence frequency of the method using $\mathbf{H}^*$ (in red) is far better than the other approaches. The efficiency of the Newton's method is limited in terms of convergence rate and number of iterations.

	Newton	HA	HC
Time/iteration (ms)	42.2	24.2	24.2
Residue (px)	0.045	0.052	0.06

Figure 4.17: Average time in ms required to compute one update $\mathbf{\Delta p}$ with a Brent line-search method and remaining positioning error after 250 iterations.

section, we will see how the formulation of the problem can be changed to improve the efficiency of the optimization.

4.2 Improving the tracker efficiency and robustness

This section is intended to present some methods that can be used to improve both the efficiency and robustness of the tracker. To evaluate the proposed methods, we will keep the same empirical validation presented in the previous section.

4.2.1 Optimization approaches

In this section, we present some possible improvements of the MI-based tracking approach in terms of efficiency and convergence domain. We begin this section by two propositions that intend to reduce the computation time. First, we apply the inverse compositional formulation to our MI optimization method. Then we present a new approach to limit the number of pixels used in the computation of the MI derivatives. Finally, we apply two coarse-to-fine schemes to increase the robustness of the tracker: one hierarchical warping method and one pyramidal tracking approach.

4.2.1.1 Inverse compositional formulation

A first step to make the optimization faster is to change the formulation of the problem. As we saw in the previous section, one drawback of the forward compositional approach is that the derivatives of the cost function are depending on the current image gradients that are obviously changing for each input image. To avoid it, we switch the roles of the current and reference images, and consider that the update parameters are applied to the reference image. This approach is called the inverse compositional formulation, that is commonly used to solve the SSD minimization problem for its efficiency [Baker 2001].

We recall that the tracking problem remains formulated as the following optimization problem:

$$\widehat{\mathbf{p}} \quad = \quad \arg\max_{\mathbf{p}} \mathrm{MI}\left(I^*(\mathbf{x}), I(w(\mathbf{x},\mathbf{p}))\right)$$

The difference with the forward compositional approach comes from the optimization process where the updating steps from the current guess to the optimal displacement parameters are modified. Instead of searching the update parameters that will bring the warped points of the current image into the points of the template image, the formulation of the problem is inverted so that we search the "inverse" update that brings the points of the template image into the warped points of the current image. The goal can thus be formulated as finding the update $\boldsymbol{\Delta}\mathbf{p}$ so that:

$$\boldsymbol{\Delta}\mathbf{p}^k = \arg\max_{\boldsymbol{\Delta}\mathbf{p}} \mathrm{MI}\left(I^*(w(\mathbf{x},\boldsymbol{\Delta}\mathbf{p})), I(w(\mathbf{x},\mathbf{p}^k))\right) \tag{4.35}$$

The updating step can be found by changing variables. If we note $\mathbf{y} = w(\mathbf{x},\boldsymbol{\Delta}\mathbf{p}^k)$ that induces $\mathbf{x} = w^{-1}(\mathbf{y},\boldsymbol{\Delta}\mathbf{p}^k)$ then equation (4.35) can be rewritten:

$$\boldsymbol{\Delta}\mathbf{p}^k = \arg\max_{\boldsymbol{\Delta}\mathbf{p}} \mathrm{MI}\left(I^*(\mathbf{y}), I(w(w^{-1}(\mathbf{y},\boldsymbol{\Delta}\mathbf{p}),\mathbf{p}^k))\right). \tag{4.36}$$

By analogy with the tracking problem definition, it becomes clear that the update satisfies:

$$w(\mathbf{x},\mathbf{p}^{k+1}) \leftarrow w(w^{-1}(\mathbf{x},\boldsymbol{\Delta}\mathbf{p}^k),\mathbf{p}^k) \tag{4.37}$$

that justifies the "inverse compositional" term: the new parameters are obtained with a composition of the transformation using the current displacement parameters $\mathbf{p}^k$ and the inverse of the transformation produced by the update $\mathbf{\Delta p}^k$.

The optimization using this formulation is similar to the optimization using the forward compositional approach. Nevertheless, since the update is considered to affect the reference image, we will see that more elements of the mutual information derivatives with respect to the update can be precomputed.

The formulation has changed, nevertheless, the function to optimize remains the same. The Newton's, HA and HC methods have, therefore, the same effect on the optimization of MI in the forward and inverse compositional approaches. Thus, we directly consider the optimization of the problem using the proposed HC approach with the Hessian matrix computed at the optimum:

$$\mathbf{\Delta p} = \mathbf{H}^{*-1}\mathbf{G} \tag{4.38}$$

As we focus on the computation of the update in one iteration, we simplify the notations by dropping the superscript k that indicates the iteration number (as we did in the previous section). The gradient matrix $\mathbf{G}$ and Hessian matrix $\mathbf{H}$ are this time computed from the equation (4.35) as:

$$\begin{aligned}
\mathbf{G} &= \left.\frac{\partial \mathrm{MI}\left(I^*(w(\mathbf{x},\mathbf{\Delta p})), I(w(\mathbf{x},\mathbf{p}))\right)}{\partial \mathbf{\Delta p}}\right|_{\mathbf{\Delta p}=\mathbf{0}} && (4.39)\\
\mathbf{H} &= \left.\frac{\partial^2 \mathrm{MI}\left(I^*(w(\mathbf{x},\mathbf{\Delta p})), I(w(\mathbf{x},\mathbf{p}))\right)}{\partial \mathbf{\Delta p}^2}\right|_{\mathbf{\Delta p}=\mathbf{0}} && (4.40)
\end{aligned}$$

The equation of the gradient and Hessian matrices are modified considering the fact that now p_{I^*} is depending on $\mathbf{\Delta p}$ and p_I is not. The developed expression becomes [Dowson 2008]:

$$\begin{aligned}
\mathbf{G} &= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I}\right) && (4.41)\\
\mathbf{H} &= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}}^{\top} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_I}\right) + \frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} \left(1 + \log \frac{p_{II^*}}{p_I}\right) && (4.42)
\end{aligned}$$

These expressions depend on the joint probability and its derivatives that are now equal to:

$$\begin{aligned}
p_{II^*} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}))\right) \phi\left(j - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right) && (4.43)\\
\frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}))\right) \frac{\partial \phi\left(j - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}} && (4.44)\\
\frac{\partial^2 p_{II^*}}{\partial \mathbf{\Delta p}^2} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(w(\mathbf{x},\mathbf{p}))\right) \frac{\partial^2 \phi\left(j - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2} && (4.45)
\end{aligned}$$

The derivatives of the ϕ function can be computed using the same decomposition that were performed in the previous section:

$$\begin{aligned}
\frac{\partial \phi(j - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p})))}{\partial \mathbf{\Delta p}} &= -\frac{\partial \phi}{\partial j} \frac{\partial \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))}{\partial \mathbf{\Delta p}} && (4.46)\\
\frac{\partial^2 \phi\left(j - \overline{I^*}(w(\mathbf{x},\mathbf{\Delta p}))\right)}{\partial \mathbf{\Delta p}^2} &= \frac{\partial^2 \phi}{\partial i^2} \frac{\partial \overline{I^*}}{\partial \mathbf{\Delta p}}^{\top} \frac{\partial \overline{I^*}}{\partial \mathbf{\Delta p}} - \frac{\partial \phi}{\partial j} \frac{\partial^2 \overline{I^*}}{\partial \mathbf{\Delta p}^2} && (4.47)
\end{aligned}$$

with:

$$\frac{\partial \overline{I^*}(w(\mathbf{x}, \mathbf{\Delta p}))}{\partial \mathbf{\Delta p}} = \nabla \overline{I^*} \left. \frac{\partial w(\mathbf{x}, \mathbf{\Delta p})}{\partial \mathbf{\Delta p}} \right|_{\mathbf{\Delta p} = \mathbf{0}} \tag{4.48}$$

$$\frac{\partial^2 \overline{I^*}(w(\mathbf{x}, \mathbf{\Delta p}))}{\partial \mathbf{\Delta p}^2} = \frac{\partial w}{\partial \mathbf{\Delta p}}^{\top} \nabla^2 \overline{I^*} \frac{\partial w}{\partial \mathbf{\Delta p}} + \nabla_x \overline{I^*} \frac{\partial^2 w_x}{\partial \mathbf{\Delta p}^2} + \nabla_y \overline{I^*} \frac{\partial^2 w_y}{\partial \mathbf{\Delta p}^2} \tag{4.49}$$

One of the main interests of the inverse formulation becomes clear. Indeed, since the reference image is constant, its gradient $\nabla \overline{I^*}$ is also constant, as well as $\nabla^2 \overline{I^*}$. The derivative of the ϕ function is computed for $I^*(w(\mathbf{x}, \mathbf{\Delta p}))$ for a null update $\mathbf{\Delta p}$, thus $I^*(w(\mathbf{x}, \mathbf{\Delta p})) = I^*(\mathbf{x})$ is also constant. Finally, since the derivatives of the warp function are also computed for $\mathbf{\Delta p} = \mathbf{0}$, they are also constant. The whole expressions defined by equations (4.46) and(4.47) are finally constant and can be precomputed for each pixel $\mathbf{x}$.

Considering the computation of the Hessian matrix $\mathbf{H}^*$, since we assume that at the maximum the current image is perfectly aligned with the reference image with: $I(w(\mathbf{x}, \mathbf{p}^*)) = I^*(\mathbf{x})$, the whole expression of the joint probability second derivatives in the equation (4.45) is constant. Therefore the Hessian matrix $\mathbf{H}^*$ remains constant, as it was previously the case in the forward compositional formulation.

In this section, we presented the inverse compositional formulation applied to the proposed HC method. It has been shown in [Baker 2001] that the inverse and forward compositional approaches are equivalent in nominal conditions. Since the derivatives are computed using the reference image (resp. current image) gradients in the inverse (resp. forward) formulation, the optimization will be more sensitive to image noise in the reference (resp. current) image. In our approaches, where, most of the time, the reference image is the first image of the sequence, the current and reference frames are equally noisy and both optimization methods are then equivalent. Considering the gain in efficiency that will be evaluated in the empirical validation, the inverse compositional approach is best suited for our tracking problem.

4.2.1.2 Template pixels selection

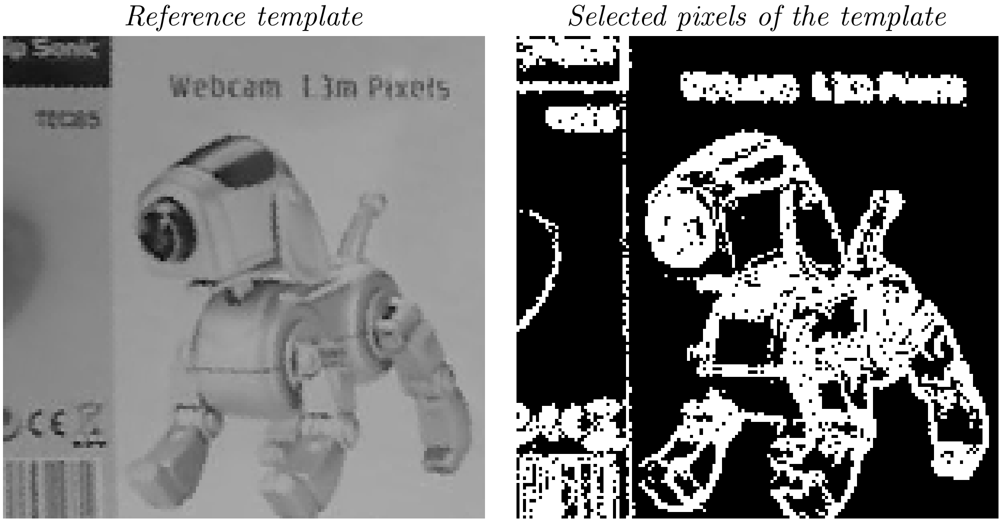
Compared to a simple SSD minimization problem, mutual information can still be considered as a complex function to compute. The inverse approach coupled with the proposed optimization method offers already a practical solution. Nevertheless, a faster computation allows to achieve more optimization iterations within the same constant amount of time (for instance between two incoming frames) and is therefore always an advantage. In this section, we will see that a simple selection of pixels of the template in the derivatives computation yields to a really faster optimization that keeps its robustness and accuracy.

To compute the MI between the two images, all the information is required, so all the reference pixels must be used to compute the marginal and joint probabilities. Nevertheless, to compute the mutual information derivatives all pixels are not necessary. Indeed, if we consider the equations (4.46) and (4.47), we can see that the effect of a pixel on the MI derivatives is proportional to the value of the gradient of the reference image at the pixel location. If a pixel is located in a uniform region of the image (with small gradients), its motion will not cause any change in the image entropy or in its mutual information with an other image. One simple modification is then to perform the computation of the gradient and Hessian matrices using only the reference pixels that are not in uniform regions.

A simple measure to consider if a pixel is in an uniform region of the template is given by the norm of the reference image gradients. Therefore, the selection condition can be written as:

$$\|\nabla I^*(\mathbf{x})\| > \alpha \tag{4.50}$$

where α is a given threshold. The computation of the joint probability derivatives in the equations (4.44) and (4.45) is therefore only performed on the reference pixels that respect this condition (see the Figure 4.18). Since the computation of the joint probability gradient remains a time consuming operation, this selection significantly improves the efficiency of the registration.


Reference template

Selected pixels of the template

Figure 4.18: To improve the computation time, only the pixels of the template that are not in a uniform region are used to compute the gradient and Hessian of MI (the pixels in white).

4.2.1.3 Hierarchical motion model

This section is intended to propose a practical solution for the estimation of complex parametric motion models such as homographies. This approach is based on a coarse-to-fine motion estimation approach that allows to improve the registration in terms of efficiency and robustness.

The transformations that we defined in the first chapter form a group with particular properties. It is indeed possible to show that translation motions are a subset of the affine transformations, that are as well a subset of the homography transformations [Hartley 2001]. Since the estimation of a translation is easier and more efficient to estimate, it is much more adapted to search the homography between two images, by first estimating the translation between them. Following the same principle, once the translation displacement is estimated, an affine displacement can be estimated to end with the homography estimation.

This optimization approach is possible with the compositional formulation. Let us note w the homography warp function and w' one transformation that belongs to the homography transformation subset. The tracking problem can be formulated as finding the update $\mathbf{\Delta p}^k$ that satisfies:

$$\mathbf{\Delta p}^k = \arg\max_{\mathbf{\Delta p}} \mathrm{MI}\left(I^*(w(\mathbf{x}, \mathbf{\Delta p})), I(w(\mathbf{x}, \mathbf{p}^k))\right) \tag{4.51}$$

where $\mathbf{\Delta p}^k$ is the displacement parameter of the transformation w'. The warp update rule is modified to project the update $\mathbf{\Delta p}$ into an update on the homography parameter $\mathbf{p}^k$ as:

$$w(\mathbf{x}, \mathbf{p}_{k+1}) \leftarrow w(w'^{-1}(\mathbf{x}, \mathbf{\Delta p}^k), \mathbf{p}^k) \tag{4.52}$$

If $\mathbf{\Delta p}$ represents a translation, then then gradient and Hessian matrices of MI are computed with respect to only two parameters. Moreover, since the displacement model is simple, the convergence is reached in a few iterations. The solution that we propose is thus to first estimate the translational motion. As soon as the convergence is reached for the current motion model, a more complex displacement model w' is considered to end with the homography estimation. In our experiments, we choose to use the translational model, then the Similitude transformation, then the homography.

The estimation of the homography $\mathbf{p}_t$ between a reference image I^* and the image I_t with a first guess $\mathbf{p}_{t-1}$ has been illustrated in Figure 4.19. The figure shows some steps of the optimization with and without the hybrid warping with respect to the number of iteration. It is clear that the convergence of the hybrid approach is fast compared to the classical approach.

On top of this faster convergence, since the motion model is simple far from the convergence, this formulation is less sensible to local maxima. The robustness of the tracker is therefore really improved by this approach. Figure 4.20 represents a registration task where the classical approach falls in a local maxima and fails, whereas the hybrid approach estimates the large translational motion first and thus, it avoids the local maxima and converges to the optimum of MI.

4.2.1.4 Pyramidal implementation

One last point that can increase the convergence rate and the robustness of the optimization is the use a spatial multi resolution approach. This method is inspired from the KLT implementation of [Bouguet 2000]. Both the reference and current images are sub-sampled to build a Gaussian pyramid. The estimation of the displacement parameters is performed from the smaller resolution to the higher resolution, that is the actual resolution of the input images. This approach allows to have a coarse to fine estimation of the displacement parameters that enlarges the convergence domain and, since the optimization is mainly performed at a lower resolution, it also speeds up the convergence and computation time.

A diagram that represents the pyramidal approach for the MI estimation is given in Figure 4.21. First a Gaussian pyramid of the reference image I^* is created. Let us call I^{*r} the reference image at a resolution level r, let $r = 0$ be the original resolution $I^{*0} = I^*$ and M the number of pyramid levels.

For each input image I a Gaussian pyramid $(I^0...I^M)$ is computed. The displacement parameters $\mathbf{p}^0_{t-1}$ estimated for the previous image for the lower level of the pyramid are converted into the corresponding displacement parameters at the M^{th} level of the pyramid:

$$\mathbf{p}^M_{t-1} = Pyr_{M,0}(\mathbf{p}^0_{t-1}) \tag{4.53}$$

where $Pyr_{n,m}(\mathbf{p}^m)$ is the function that converts the displacement parameters from the m^{th} pyramid level to the n^{th} that is detailed in the Appendix A.2. A first estimation of the displacement $\mathbf{p}^M_t$ is then performed by the MI optimization between the two images I^M and I^{*M}. Since the optimization uses the sub-sampled input images that give less information than the original images, the resulting estimation can be considered as "coarse". The estimated parameter is then converted into the corresponding displacement in the lower level of the pyramid:

$$\mathbf{p}^{M-1}_{t-1} = Pyr_{M-1,M}(\mathbf{p}^M_t) \tag{4.54}$$

then the estimation of the displacement is refined with the images I^{M-1} and I^{*M-1} of this lower level of the pyramid. The process is repeated until the lower level of the pyramid is reached, and finally, the finer parameter $\mathbf{p}^0_t$ is obtained using the original input images.

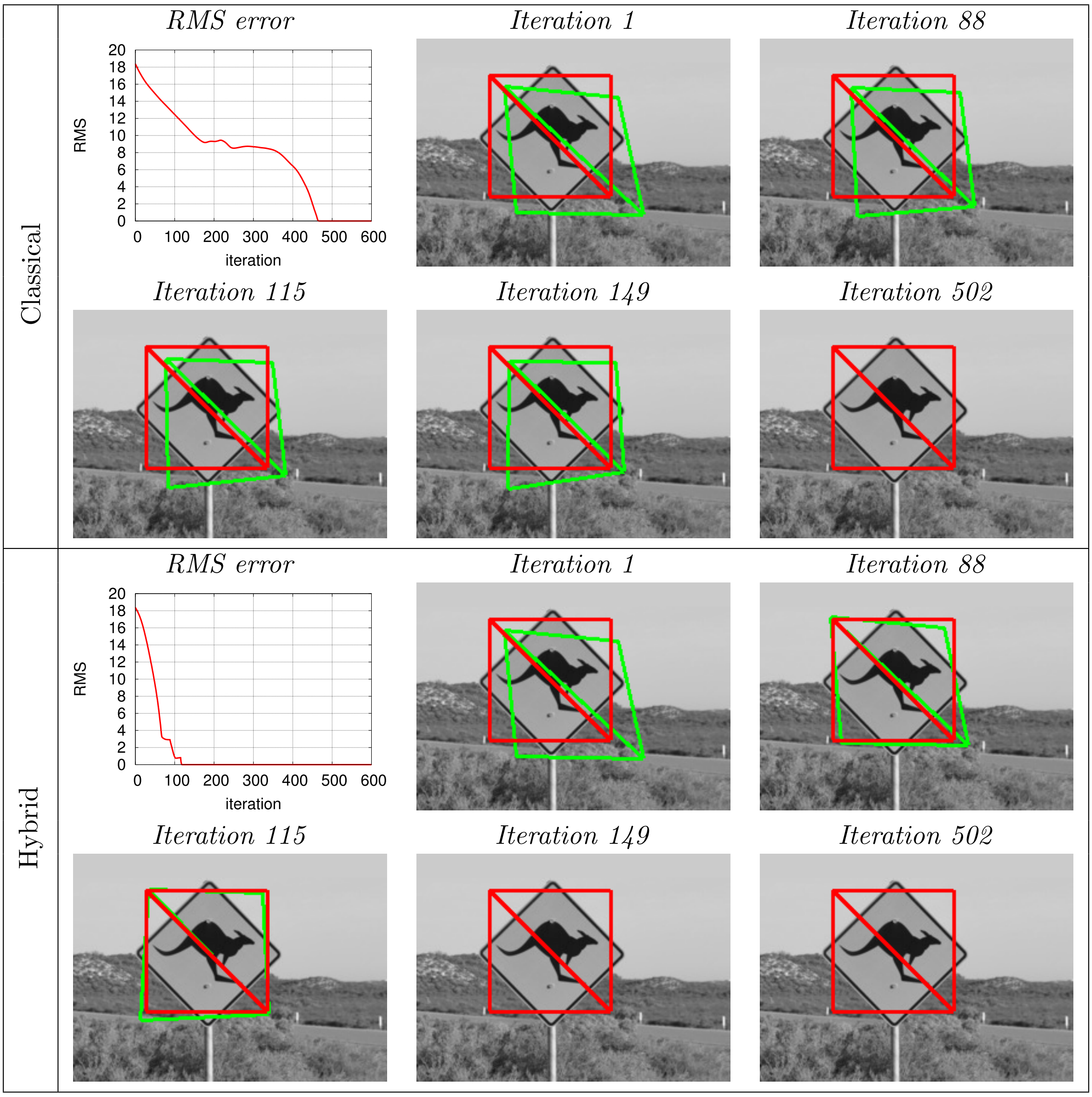


Figure 4.19: Efficiency of the hybrid approach: comparison between the convergence in the classical optimization and the hybrid warping optimization. The red rectangle represents the ground truth of the position of the template in the image and the green one represents the current estimated position. The convergence is reached in 149 iterations using the hybrid warping process and 502 iterations with the classical one. We represent the iterations corresponding to the moment when the warping function w' changed in the hierarchical approach: translation estimation [0..88], affine estimation [88..115], homography estimation [115..149].

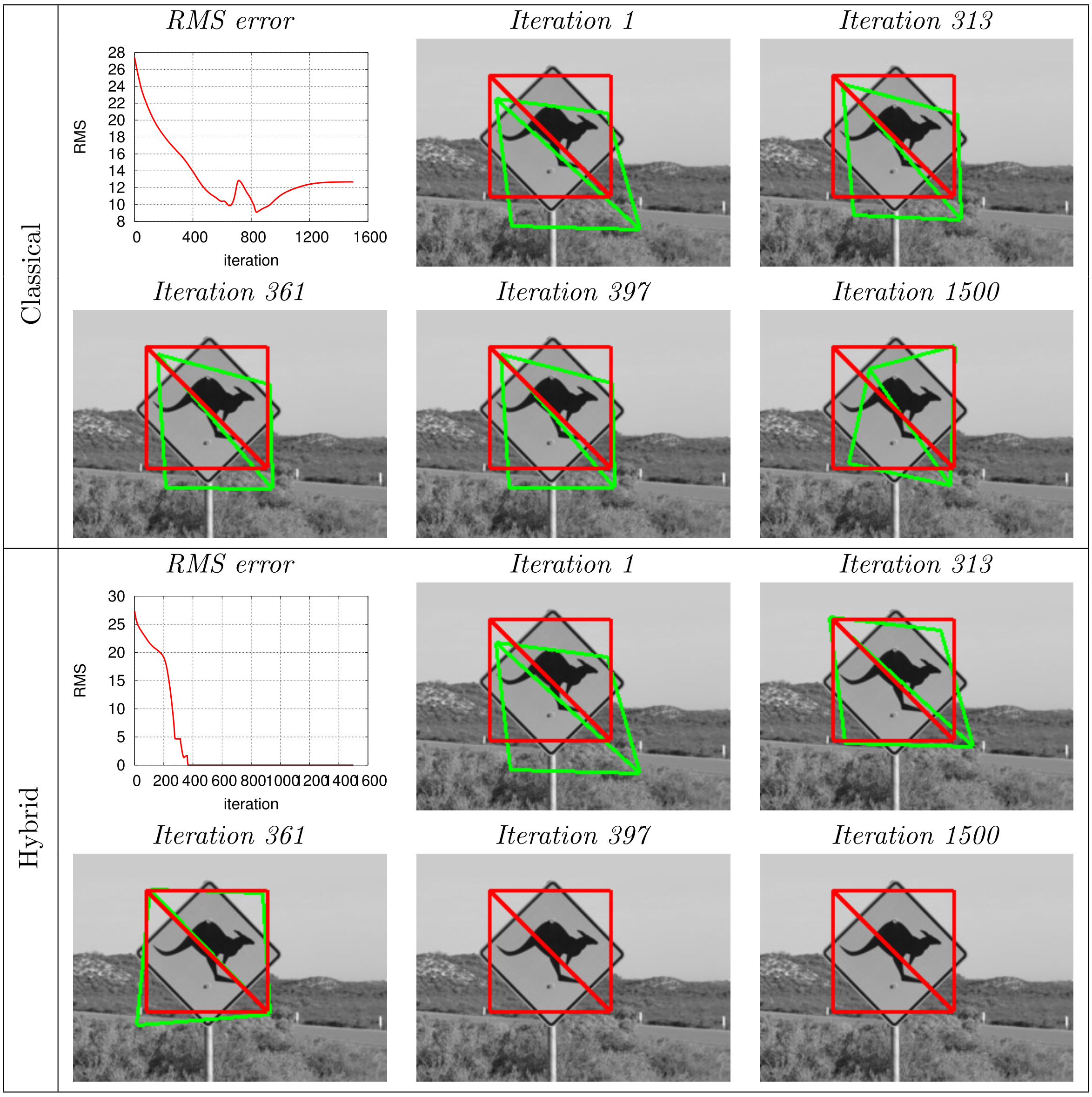


Figure 4.20: Robustness of the hybrid approach: comparison between the classical and hybrid approaches as in Figure 4.19. The classical approach falls in a local maxima while the hybrid approach converges.

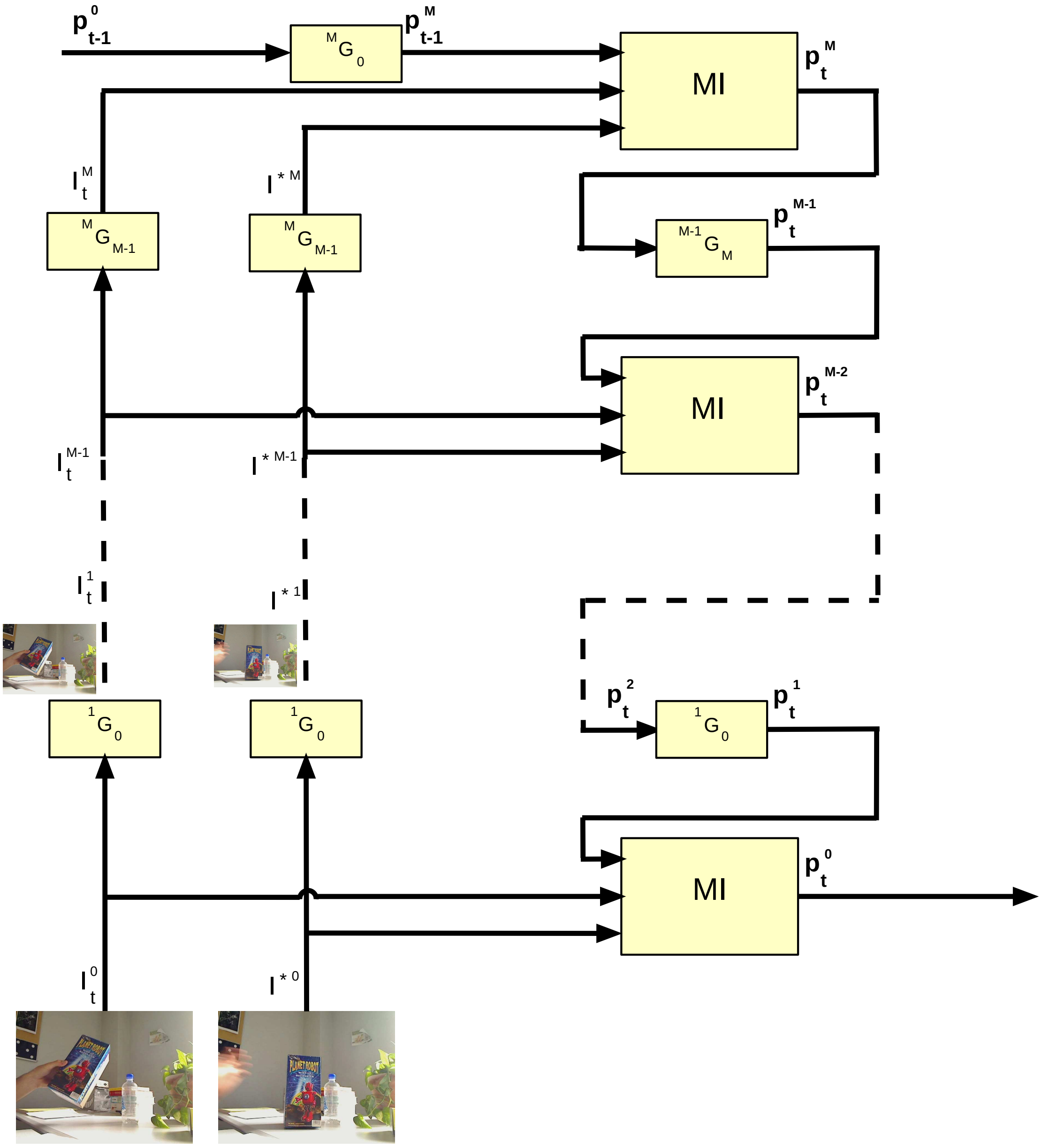


Figure 4.21: Mutual information-based tracking.

4.2.2 Empirical convergence analysis

In this section, we use the empirical evaluation defined in the previous section to measure the performances of the considered registration approaches, and validate their advantages. We reported the results in the Figure 4.22. The original inverse approach using the Hessian computed at convergence is referred as "HC", the one using the selection approach as "HC fast", the hierarchical approach as "HC warp", and the approach using both the hierarchical warp model and the pyramidal approach as "HC pyramidal + warp".

First, let us compare the results obtained using the inverse and compositional approaches using the HC method. The frequency of convergence and the number of iterations to converge are similar between the two approaches. As expected, the difference comes from the time per iterations reported in the Figure 4.23: the computation time of the inverse approach is half the computation time of the forward approach.

Now, if we compare the classical inverse approach with the one that selects the reference pixels for the MI derivatives computation, we can see that the performances are also similar. In this experiment, only 30% of the pixels were used to compute the gradient and Hessian matrix of MI. This selection improves the computation time of about 33% although the convergence frequency and required number of iterations remain similar.

Finally, as it is generally the case, the coarse-to-fine approaches allow a large increase of the convergence frequency for a smaller number of iterations. The average computation time is not significantly modified. Indeed, the convergence is rapidly reached and most of the 250 iterations used to compute the computation time are therefore performed using the finest estimation. However, in practice, the computation time is greatly reduced.

4.2.2.1 Conclusion

In this section, from MI based registration methods that had a very small convergence domain or a low efficiency, we developed a really efficient registration approach. A new optimization approach that allows for an fast and accurate estimation of the optimum is proposed. This optimization is generic to the forward and inverse compositional approaches. We show that additional improvements can be applied to the inverse formulation using a selection of the reference pixels to obtain a faster computation of the MI derivatives. However, we still have to evaluate the robustness of the proposed approach to appearance variations that is the topics of the following section.

4.3 Experimental results

The visual tracking method that is presented in this chapter has been implemented on a laptop with an Intel core 2 duo 2.4GHz processor. The evaluation of the displacement parameters has been performed using the presented inverse compositional scheme combined with the coarse-to-fine approaches. We limit our experiments to the displacement estimation of planar objects (or almost planar objects in the mosaicing application).

4.3.1 Tracking experiments

The robustness and accuracy of the proposed mutual information tracker have been evaluated on various image sequences. This first experiments show the performance of the displacement estimation in classical tracking problems.

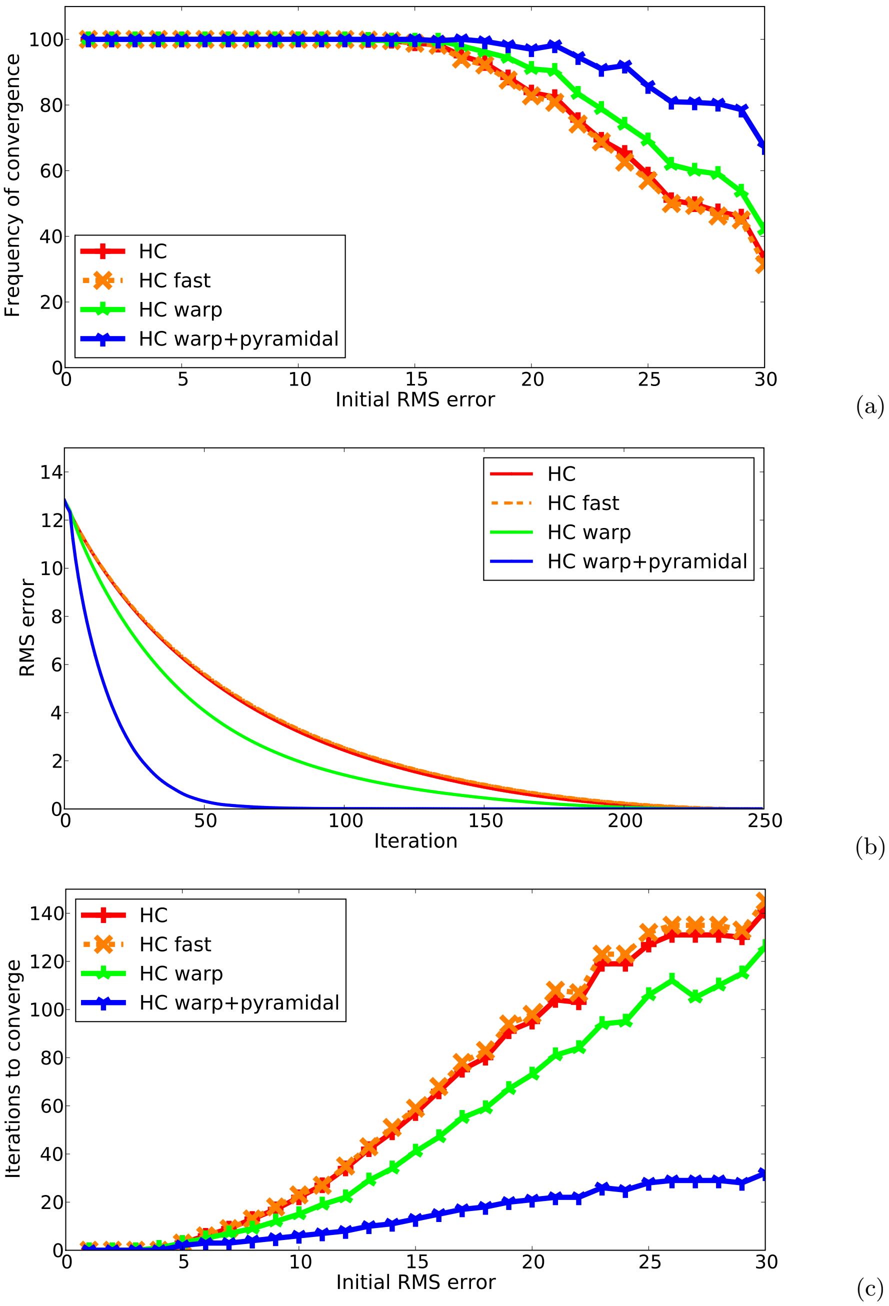


Figure 4.22: Empirical convergence analysis of the improved optimization methods. The efficiency of the fast approach is similar to the classical proposed approach. The hybrid warping and pyramidal approaches increase the efficiency of the optimization in terms of both convergence domain and speed.

	Inverse	Inverse fast	Inverse fast hierarchical	Inverse fast hierarchical and pyramidal
Time/iteration (ms)	5.1	3.4	3.1	3.0
Residue (px)	0.06	0.06	0.06	0.06

Figure 4.23: Average time in ms required to compute one update $\mathbf{\Delta p}$ with the improved methods.

4.3.1.1 Indoor sequences

These experiments concern two indoor sequences acquired at video rate (25Hz). The initialization of the tracker has been performed by learning the reference image from the first image of the sequence and setting the initial homography to the identity. No ground truth of the position of the object is known, so the comparison in this section is only qualitative.

Comparison between the SSD, NCC and MI trackers

This first experiment shows the advantages of using the MI tracker compared to the SSD tracker using a simple Gauss-Newton method [Lucas 1981, Shi 1994] and the NCC tracker using the HC method.

The figure 4.24 illustrates some images of a sequence showing a planar object. The parametric motion is an homography. We represent the estimated position of the reference template by its projection in the image. The reference template includes 5000 pixels. When there is no strong appearance variations, the SSD tracker performs a good estimation of the object motion. The registration is very efficient. Indeed, the estimation is performed in an average of 8 miliseconds per image. Nevertheless, once the object is subject to illumination variations, the SSD tracker diverges. The NCC and MI trackers still converge even in the presence of occlusions and illumination variations. But the estimation of the displacement using the NCC is not as robust to occlusions as the MI tracker, we can see in the Figure 4.24 that the optimization falls into a local maxima or the local maxima of NCC is no more corresponding to the correct position of the object. The computation time is however higher for this two approaches: the NCC tracker requires about 20 ms per image and the optimization of MI requires about 35 ms per image.

In figure 4.25, we compare the result of the different tracking approaches on a sequence where the object is not planar. In this sequence, obtained from Youtube, the tracked object is the face of a speaker. The speaker moves his hands in front of his face, creating large occlusions. Since the object is not planar, we simply want to estimate a general displacement of the face in the image and use an affine motion. All the approaches keep on tracking tracking the face despite the large occlusions. Nevertheless, we can observe that during the first occlusion, the estimation given by the MI-based tracker is far more accurate than the other approaches.

Testing the tracker robustness

This second sequence is chosen to illustrate the robustness of the motion estimation through many perturbations. Some images of the sequence are shown in figure 4.26. The template includes 16000 reference pixels. Firstly, the object is subject to several illumination variations: the artificial light produced an oscillation on the global illumination of the captured sequence. Moreover the object is not Lambertian, thus the sequence is subject to saturation and specularities. The object is moved from its initial position using wide angle and wide range motions. And finally the object is subject to fast motion causing a significant blur in many images.

The frames of the sequence are presented with the corresponding estimated positions of the reference image. No ground truth of the object position is known, however, the projection of the tracked image on the reference image has been performed and qualitatively attests the accuracy of the tracker. Indeed the reconstructed templates show strong variations in terms of appearance but not in terms of position. The augmented reality application based on the homography estimation and presented in Figure 4.27 also illustrates the quality of the computed displacement. We can conclude that the estimation of the motion is robust and accurate despite the strong illumination variations and blurring effects.

Concerning the processing time, using inverse approach with the Hessian computed at the optimum with no selection of the reference pixels (section 4.2.1.1) the images are processed at

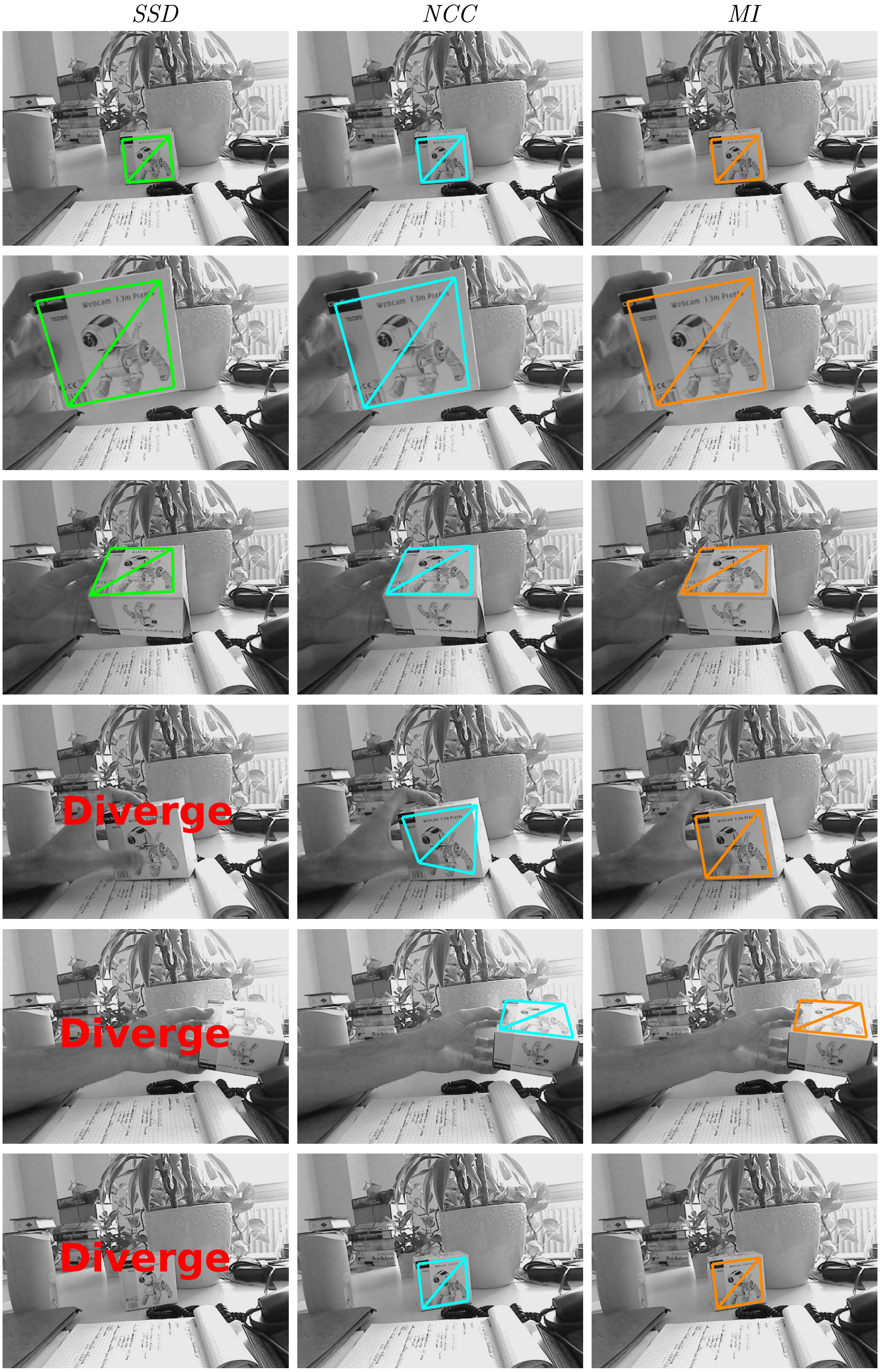


Figure 4.24: Tracking a planar object in an indoor sequence. Comparison between the SSD (in green), NCC (blue) and MI tracker (orange): the SSD tracker fails when the object is subject to small local illumination variations, while both the NCC and MI trackers converge. However the NCC tracker is more sensible to occlusions.

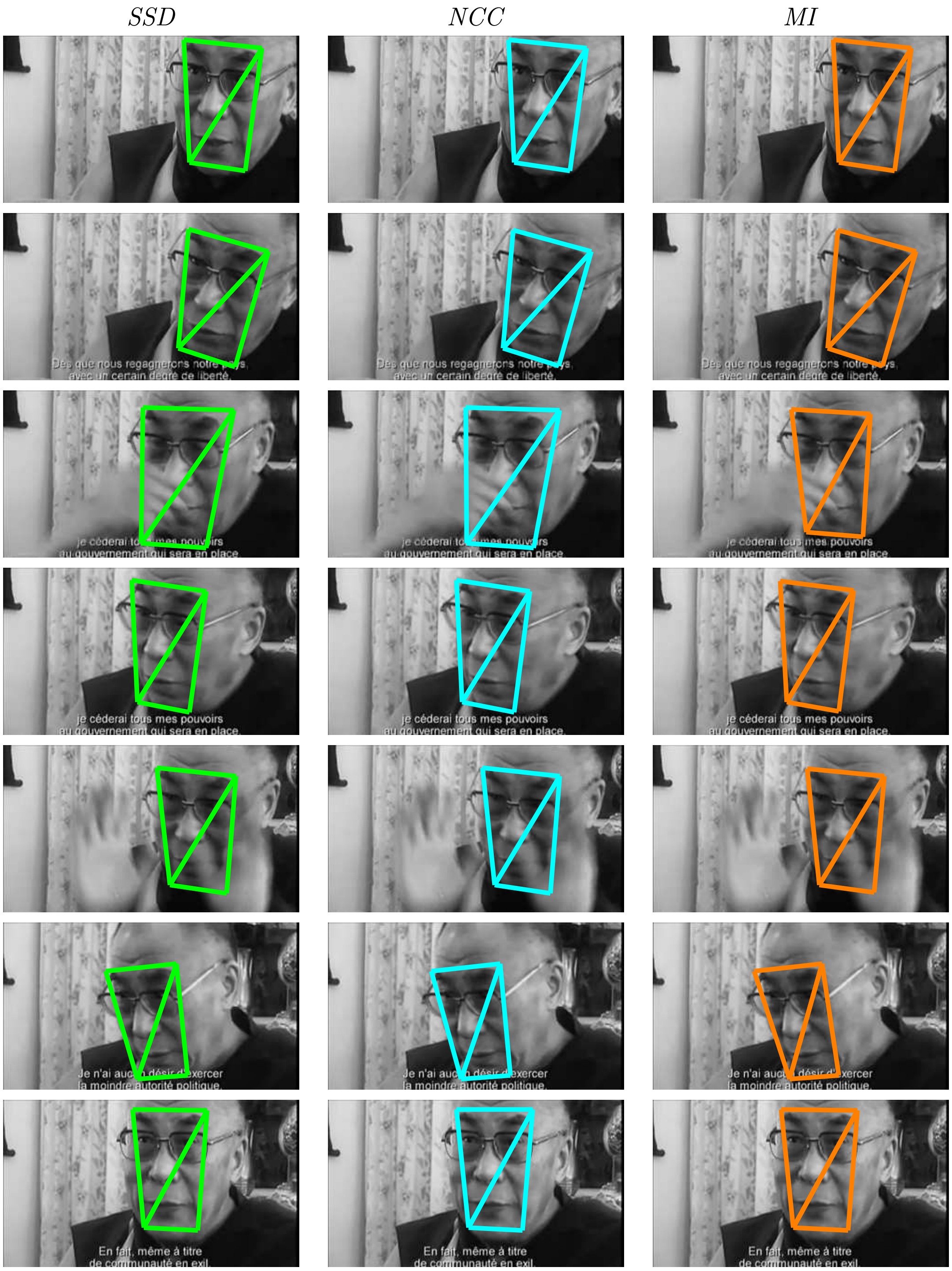

SSD
NCC
MI
Dès que nous regagnerons notre pays
avec un certain degré de liberté,
je céderai tous mes pouvoirs
au gouvernement qui sera en place.
Je n'ai aucun désir d'exercer
la moindre autorité politique.
En fait, même à titre
de communauté en exil

Figure 4.25: Tracking a non-planar object using affine motion. Comparison between the SSD (in green), NCC (blue) and MI tracker (orange): the SSD and NCC trackers are more sensible to local occlusions than the MI-based tracker.

video rate (25Hz). Using the fast computation (section 4.2.1.2) it is about 40Hz with the same robustness and precision.

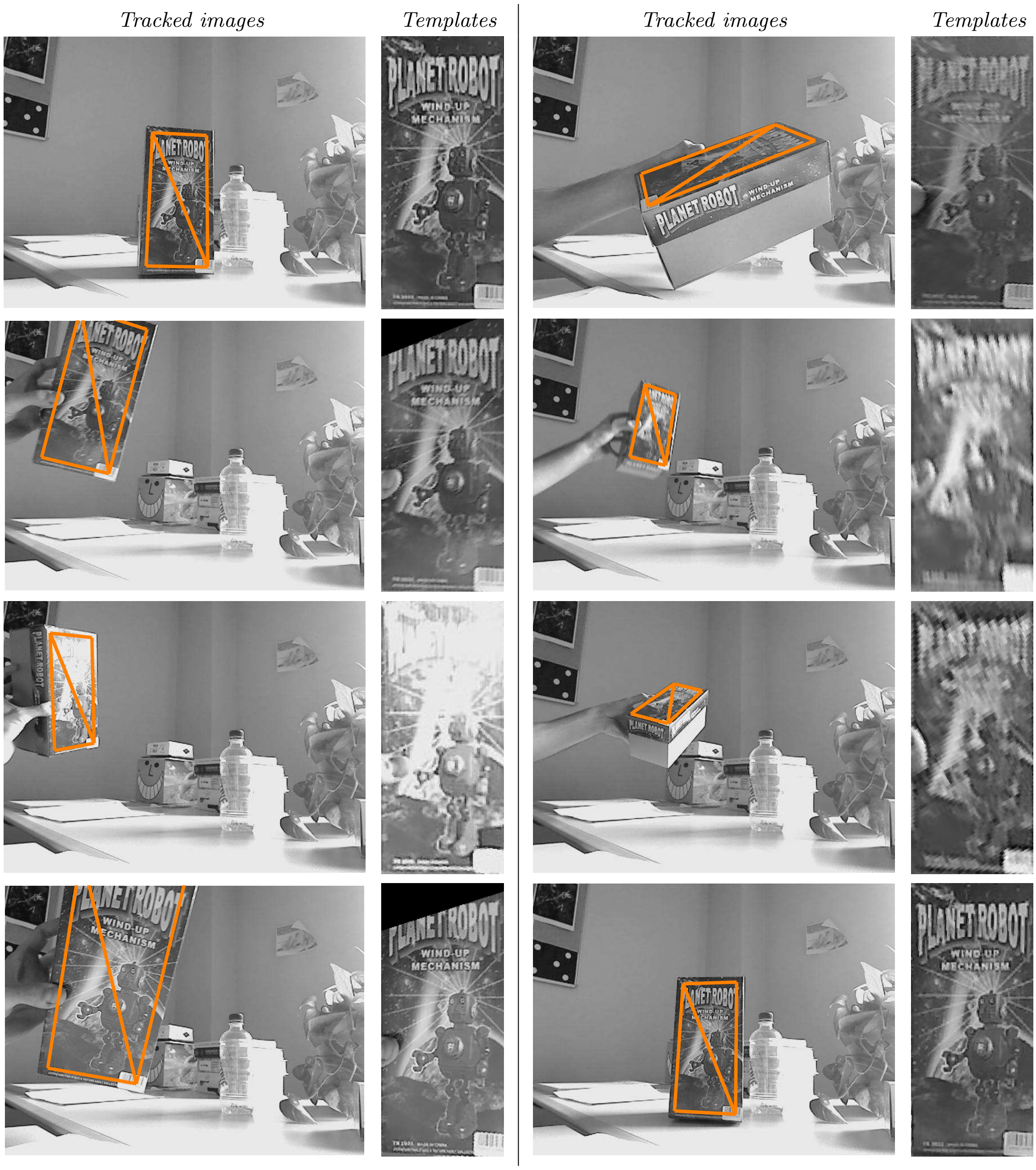


Figure 4.26: Tracking a planar object through illumination variations. Tracked frames: the orange rectangle represents the rectangle from the template image transformed using the estimated homography. Templates: projection of the templates for the same iterations in the reference image.

4.3.1.2 Evaluation on benchmark datasets

To have a quantitative measure of its accuracy and robustness, the tracker has been evaluated on some very demanding reference datasets proposed by Metaio GmbH [Lieberknecht 2009]. Those datasets include a large set of sequences with the typical motions that we are supposed to face in augmented reality applications. Indeed it includes sequences using the eight reference

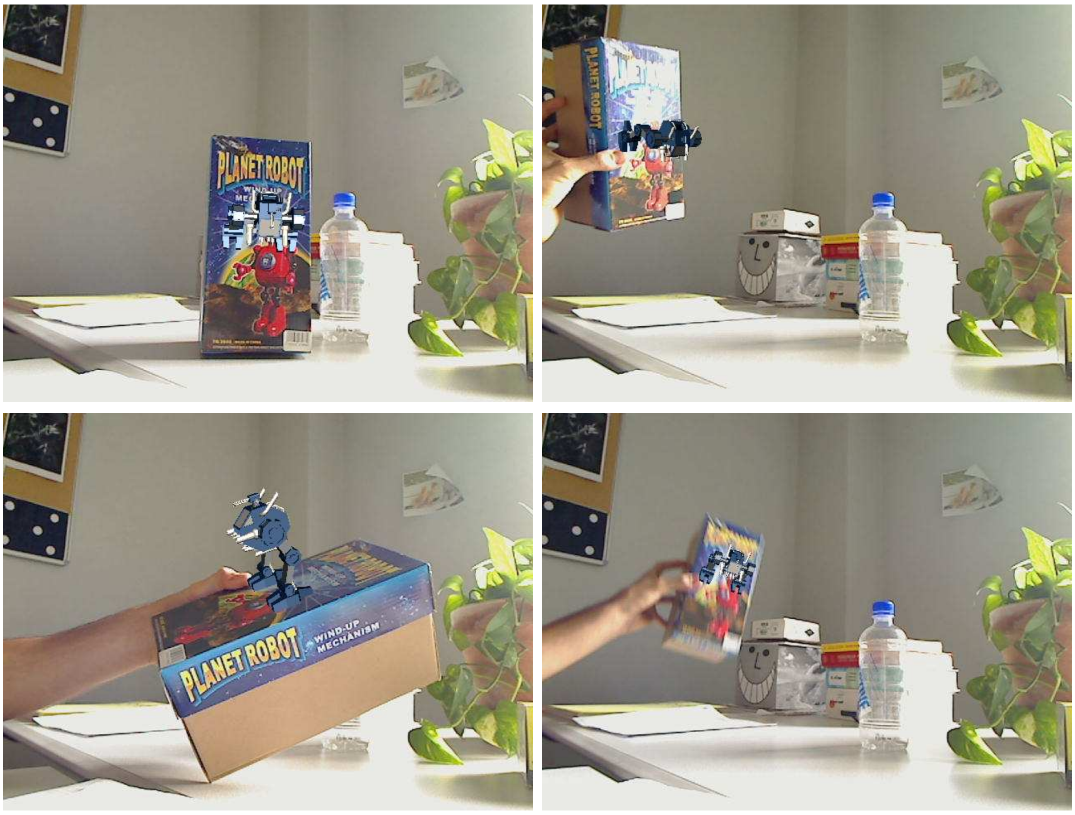
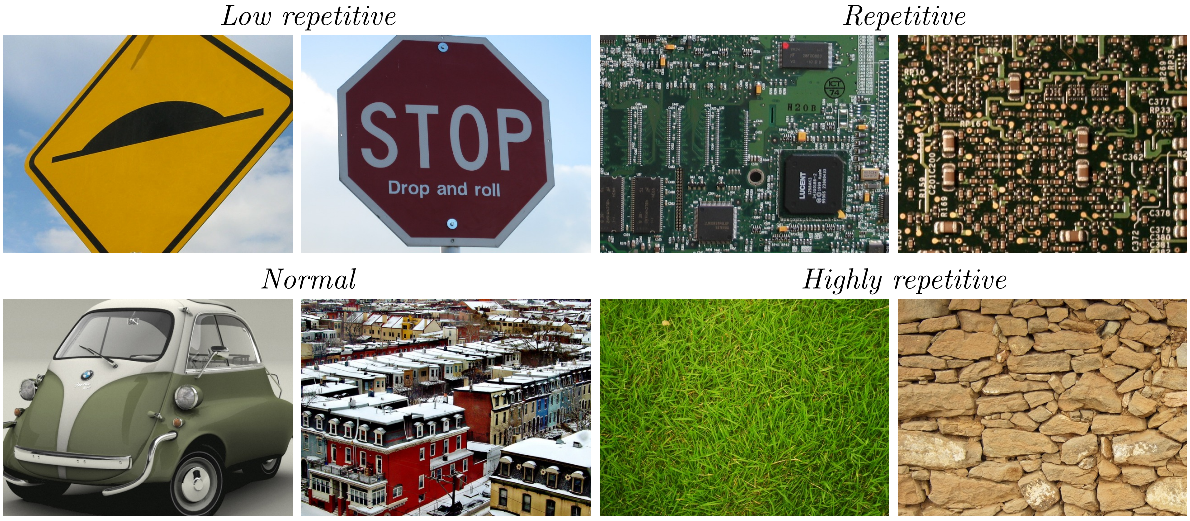


Figure 4.27: Augmenting the input images with a virtual robot placed on the top of the box.

images presented in Figure 4.28, from low repetitive texture to highly repetitive texture. And for each reference image is a set of four sequences depicting wide angle, high range, fast far and fast close motion and one sequence with illumination variations.



Low repetitive

Repetitive

Normal

Highly repetitive

Figure 4.28: Reference images used in the tracking experiments from low repetitive texture to highly repetitive texture.

The estimated motion has been compared with the ground truth for each sequences. The percentages given in the tables have been computed by Metaio relative to their ground truth.

The upper table on figure 4.31 shows the results that have been obtained using the proposed approach. The tracker is considered in convergence if the error between the estimation and the ground truth is below a given threshold. The error measure represents the reprojection error of the corner of the reference in pixels. Its equation is the same as the one defined in equation (4.34). Some images of the sequences are shown in Figure 4.29 and Figure 4.30 with the estimated position of the reference template. The mutual information-based tracker proves its robustness and accuracy on most of the sequences.

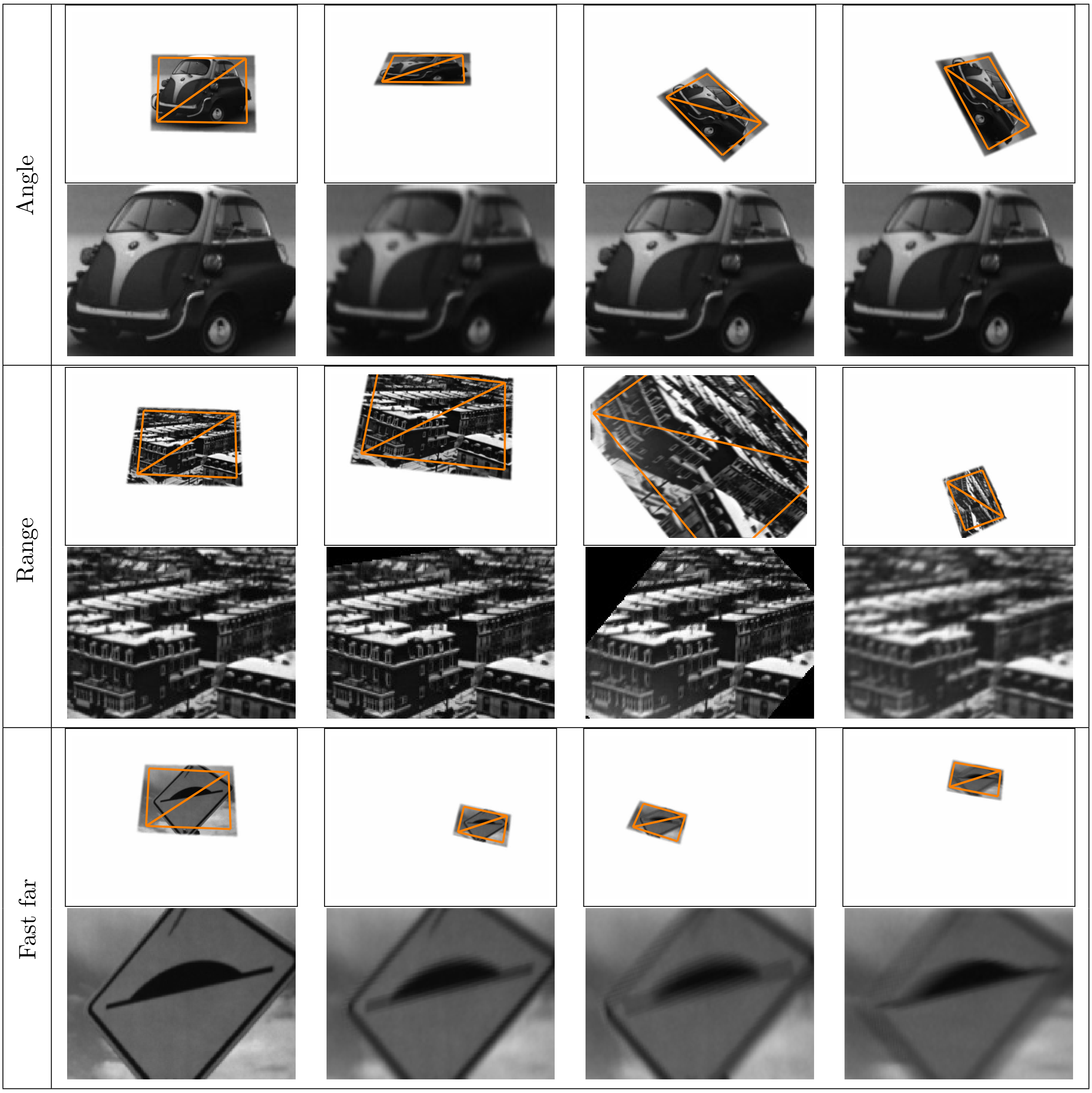


Figure 4.29: Various image sequences from the dataset: angle, range and fast far. The first line represents the image with the estimated position of the reference (in green). The second line represents inverse projection from the image to the reference image.

The results obtained using the ESM approach [Benhimane 2007] are also represented in

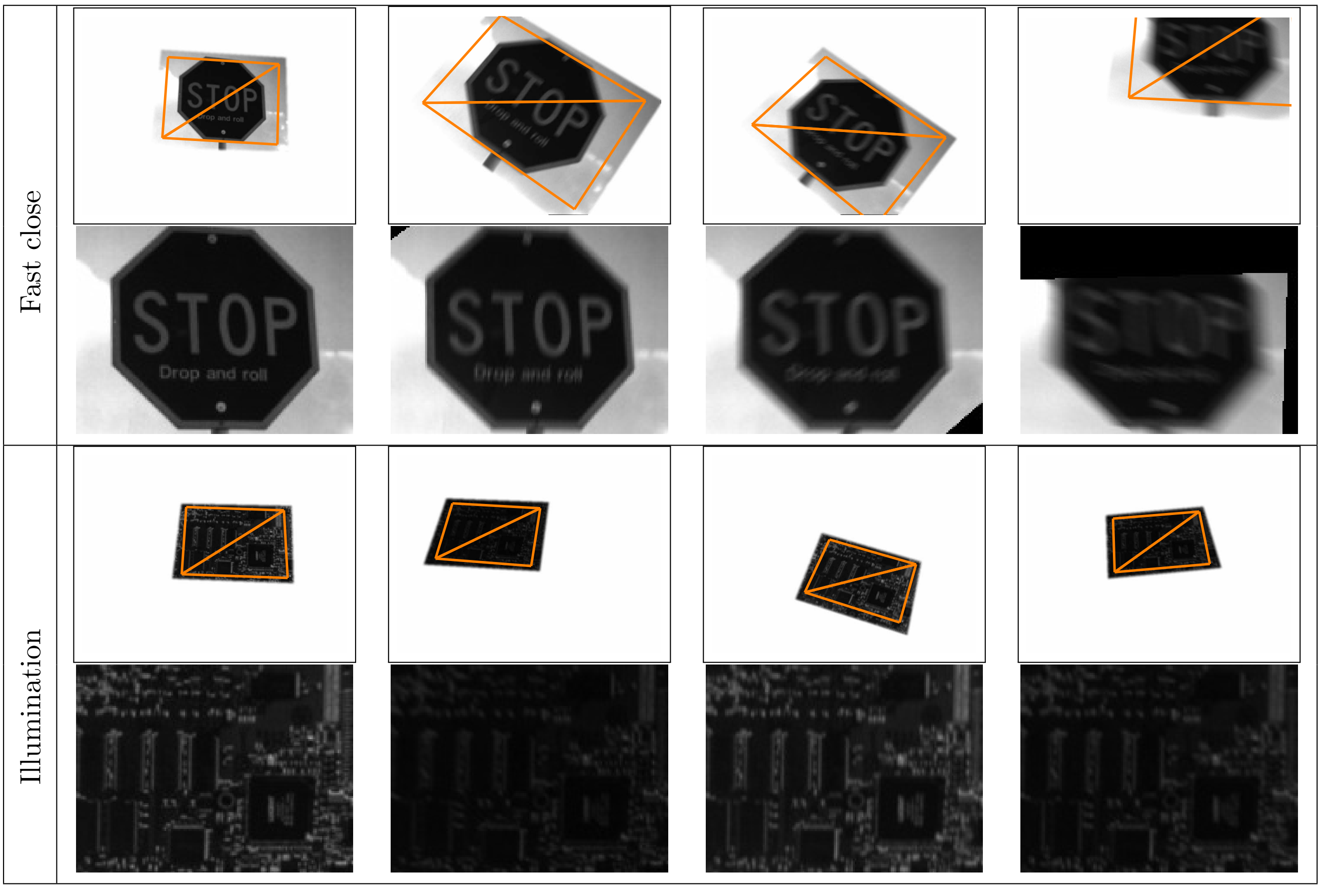


Figure 4.30: Various image sequences from the dataset: fast close and illumination variations.

the lower table of figure 4.31 where better convergence results are in bold font (reported from [Lieberknecht 2009]). If we compare the results of the two methods we can see that both have similar convergence rates in most cases. But MI has an undeniable advantage in the cases of illumination variations experiments.

We can conclude that the proposed MI computation has a large convergence domain (at least as large as the one using the SSD function) and that the proposed registration scheme is adapted to use the potential of the MI function leading to a very efficient tracker well suited for the augmented reality problem.

MI	Angle	Range	Fast Far	Fast Close	Illumination
Low	100.0 %	**94.1 %**	**75.2 %**	**56.5 %**	**99.5 %**
	100.0 %	**98.1 %**	**69.9 %**	**43.7 %**	**93.0 %**
Repetitive	**76.9 %**	**67.9 %**	**22.8 %**	**63.6 %**	**100.0 %**
	91.3 %	**67.1 %**	**10.4 %**	**70.5 %**	**96.2 %**
Normal	**99.2 %**	**99.3 %**	**43.9 %**	**86.7 %**	**99.6 %**
	100.0 %	**100.0 %**	14.8 %	84.5 %	**100.0 %**
High	**47.1 %**	**23.2 %**	**7.2 %**	**10.0 %**	**50.6 %**
	100.0 %	**69.8** %	**20.8 %**	**83.8 %**	**100.0 %**

ESM	Angle	Range	Fast Far	Fast Close	Illumination
Low	100.0 %	92.3 %	35.0 %	21.6 %	71.1 %
	100.0 %	64.2 %	10.6 %	26.8 %	56.3 %
Repetitive	61.9 %	50.4 %	22.5 %	50.2 %	34.5 %
	2.9 %	11.3 %	6.8 %	35.8 %	11.3 %
Normal	95.4 %	77.8 %	7.5 %	67.1 %	76.8 %
	99.6 %	99.0 %	**15.7 %**	**86.8 %**	90.7 %
High	0.0 %	0.0 %	0.0 %	0.0 %	0.0 %
	100.0 %	61.4 %	22.8 %	45.5 %	79.7 %

Figure 4.31: Ratio of successfully tracked images for our approach compared to the ESM [Lieberknecht 2009] (these results are provided by Metaio after the evaluation of our results).

4.3.2 Multimodal tracking

In this section, we show the application of the proposed approach in multimodal tracking or registration problems. We call multimodal images, the images that have been acquired using different types of sensors. In this chapter, we will present applications registering a map with airborne images and a satellite image with infrared images.

4.3.2.1 Satellite images versus map

This experiment illustrates the capabilities of the mutual information-based tracker in alignment applications between map and aerial images (see Figure 4.32). The reference image is a map template provided by IGN (Institut Géographique National) that can easily be linked to Geographic Information System (GIS) and the sequence has been acquired using a moving USB camera focusing on a poster representing the satellite image corresponding to the map.

As it has been previously noticed in the second chapter on mutual information, a non-linear relationship exists between the intensities of the map and aerial image and this link can be evaluated by the MI functions. Mutual information can therefore allow for tracking the satellite image using the map template. Figure 4.33 shows the reference image and some image of the sequence with the corresponding overlaid results. There is no available ground truth for this experiment, nevertheless the overlaid results give a good overview of the alignment accuracy. To validate the accuracy, we also used the estimated homography in an augmented reality

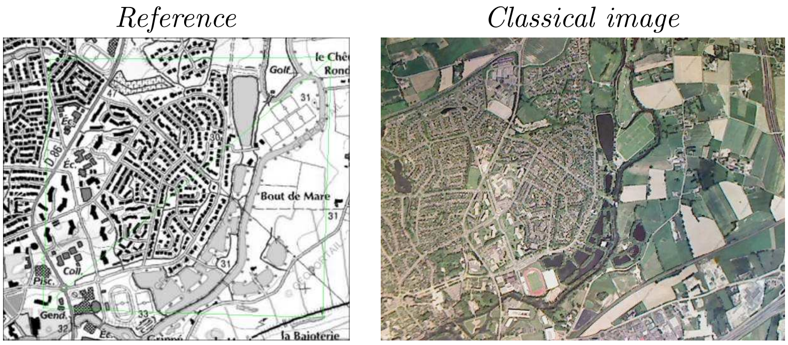


Figure 4.32: Input images: map vs airborne images.

application. Since the IGN map is linked with a GIS, some virtual information such as road, hydrographic network, or house footprint can be overlaid on the original satellite image in a consistent way.

4.3.2.2 Airborne infrared image versus satellite images

The same method has been evaluated with another current modality. This time the reference is a satellite image and the sequence is an airborne infrared sequence provided by Thales Optronic (see Figure 4.34). The initial homography is manually defined.

As we can expect, although very different, the two images shown in figure 4.35 are sharing a lot of information and thus MI can handle the tracking on the infrared sequence. The warp function is still a homography. The satellite scene is then supposed to be planar leading to an approximation. Nevertheless the proposed method remains robust. No ground truth is available, but the overlaid images as well as the augmented reality application qualitatively validates the accuracy of the tracker. As figure 4.35 shows, the satellite image of the airport is well tracked on the sequence.

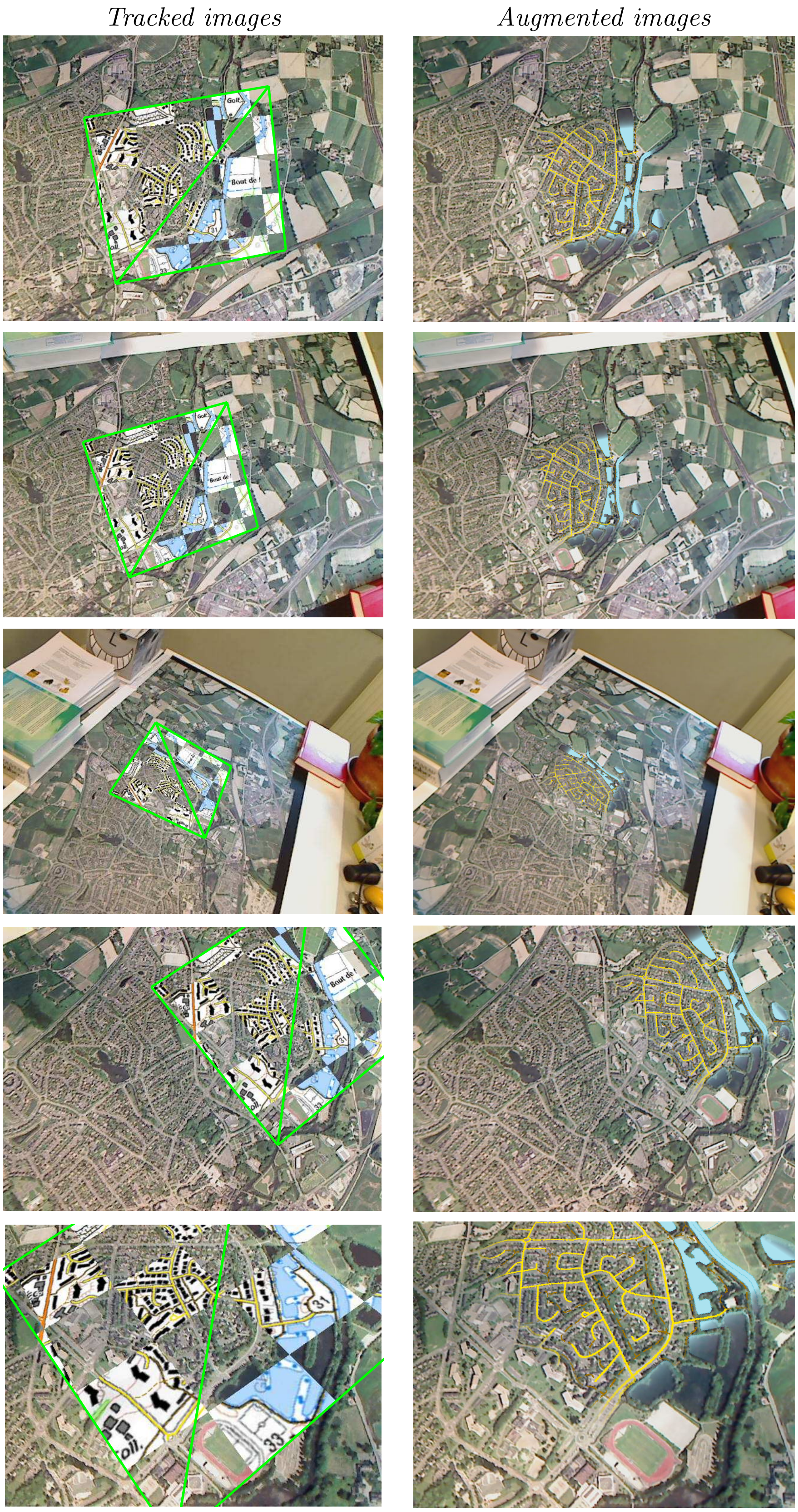
The homographies have been decomposed to estimate the position of the plane with respect to the airport. The resulting 3D trajectory of the camera is represented in figure 4.36, as we can see the trajectory is smooth and has the expected behavior that shows the approach of a plane with respect to the runway. Figures 4.35 and 4.37 show some tracked images and some augmented images that validate the accuracy of the motion estimation.

4.3.2.3 Potential AR applications of multimodal registration

Registering a map and an aerial image sequence is an extreme case, but registration between aerial and satellite (or any combination of such modalities), acquired at different time (and thus different) can be considered. Potential applications include visual odometry, aircraft or drone localization, pilot assistance, etc.

Infrared cameras (although still expensive) are widely used by civilians and, obviously, military aircraft. Such a registration process with a simple satellite image may prove to be very helpful for the pilots especially when landing (night or day) on a small and ILS free airport. Considering that aircraft position is fully known, additional information about runway, other aircraft positions or military targets may thus be easily displayed in the pilot helmet.

Although we mentioned here applications in the aeronautic area, it is clear that other domains may be targeted such as energy monitoring, robotics, urbanization, architecture or defense.



Tracked images

Augmented images

Figure 4.33: Tracking an aerial sequence using a map template image by MI: the frames are represented with the overimposed satellite reference (inside the green rectangle) projected using the estimated homography (image and map source: IGN) and augmented with roads and aquatic areas.

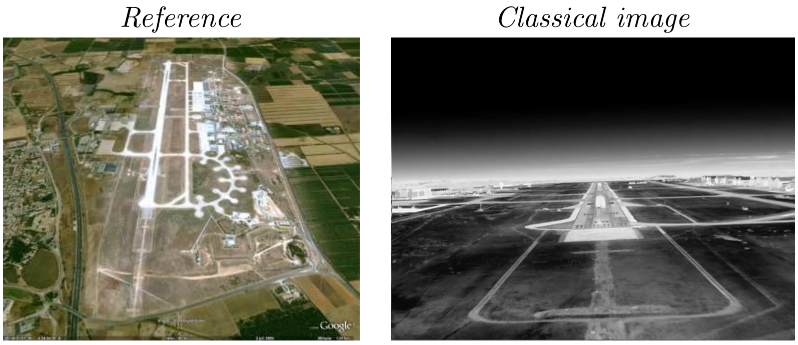


Figure 4.34: Input images: satellite vs infrared airborne images.

4.3.3 Mosaicing application

This experiment shows the application of the MI tracker to the mosaicing problem that was defined in Frame 1 page 16. We recall that the goal of the mosaicing task is to create one image from several overlapping images. In this experiment, the overlapping images are simply a compressed sequence of 3600 images obtained from Youtube. The aerial scene is acquired from a camera embedded on a flying UAV and shows the ground that is approximately 1 kilometer away from the camera. Since this distance is very large, the scene can be approximated as a plane and tracked using homographies. During the acquisition of the sequence, the camera is moving forward and is rotating around the vertical axis.

Since there is no object visible in the whole sequence, it is necessary to define multiple reference images. The approach is build as follows:

- Initialization: the first image is chosen as reference image, *i.e.* $I_0^* = I_0$.
- Tracking: for every frame, we compute the displacement $\mathbf{p}_k$ between I_t and $I^*{}_k$.
- Reference Update: every 30 images, the reference image I_k^* is changed and defined as the current image, *i.e.* $I_k^* = I_t$ for $t = 30k$.

Using the homography from the current image to the current reference image and the homographies between the references, we retrieve the homography between the current image and the first image. Using this homography, we can project all the images of the sequence into the mosaic image and construct the global image of the whole scene. In Figure 4.38 we show some images from the sequence. This sequence has been downloaded on Youtube and is affected by the H264 coding artifacts. We can also note the poor quality of the images. As we can see in the resulting mosaic image. Despite this poor quality, the resulting mosaic presented in Figure 4.39 shows the accuracy of the MI tacker. Since the camera is making an entire revolution, the first and last images are overlapping. A small shift occurs between the first and last estimated positions: we highlight two corresponding patterns that should have been at the same location on the mosaic. Let us note that nothing has been performed to reduce the drift (such as the bundle adjustment approach proposed by [Brown, Matthew 2007]). Considering the template update problem and the planar approximation result, we can assume that the estimated homographies are accurate. The same experiment was performed using the SSD tracker. In this case, due to the noise and illumination variations, this registration approach diverges after a few iterations.

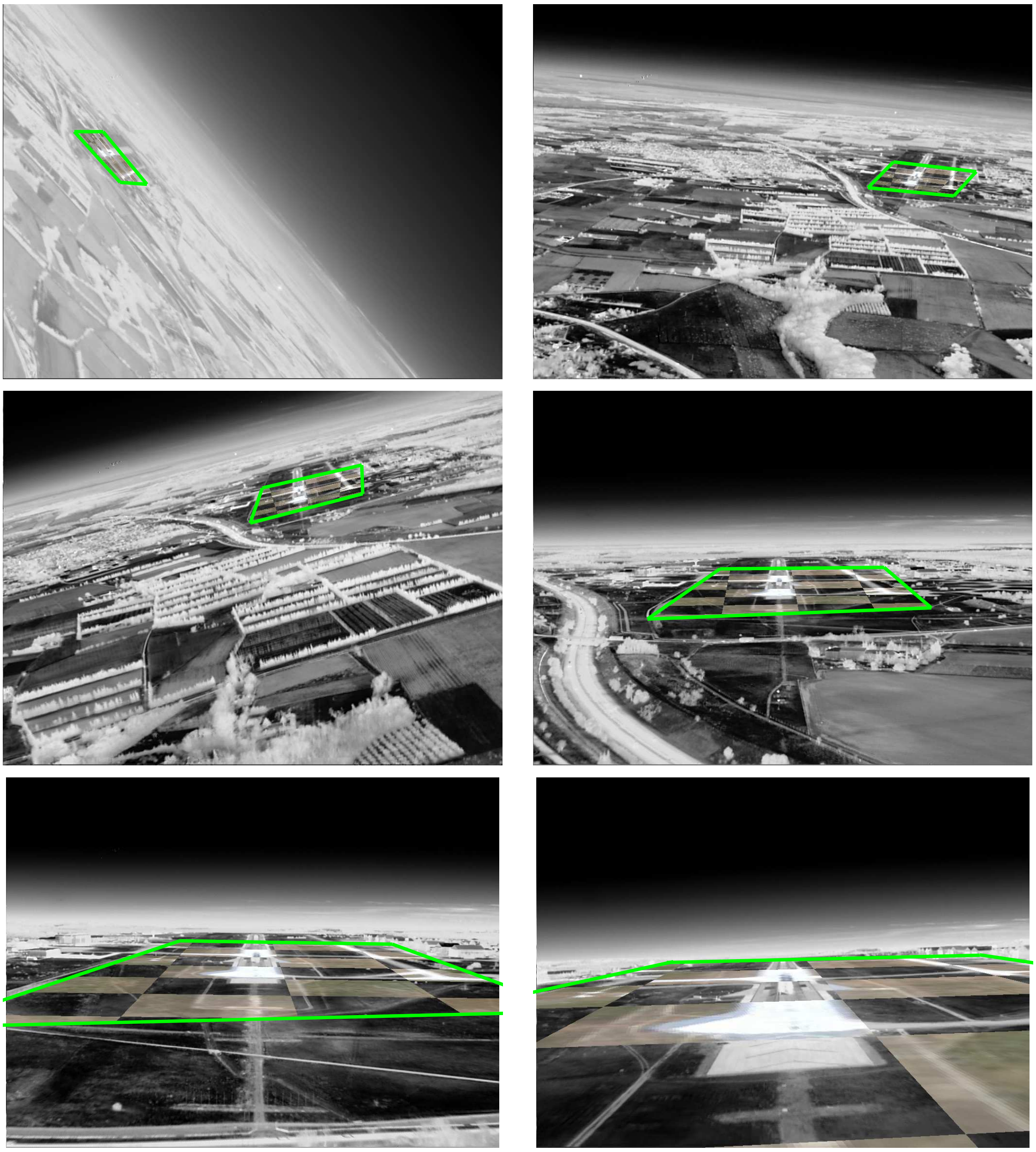


Figure 4.35: Tracking of a satellite template image using MI on an airborne FLIR sequence. 6 frames are represented with the overimposed aerial reference (inside the green rectangle) projected using the estimated homography (Infrared images courtesy of Thales Optronic, optical image is obtained from google earth).

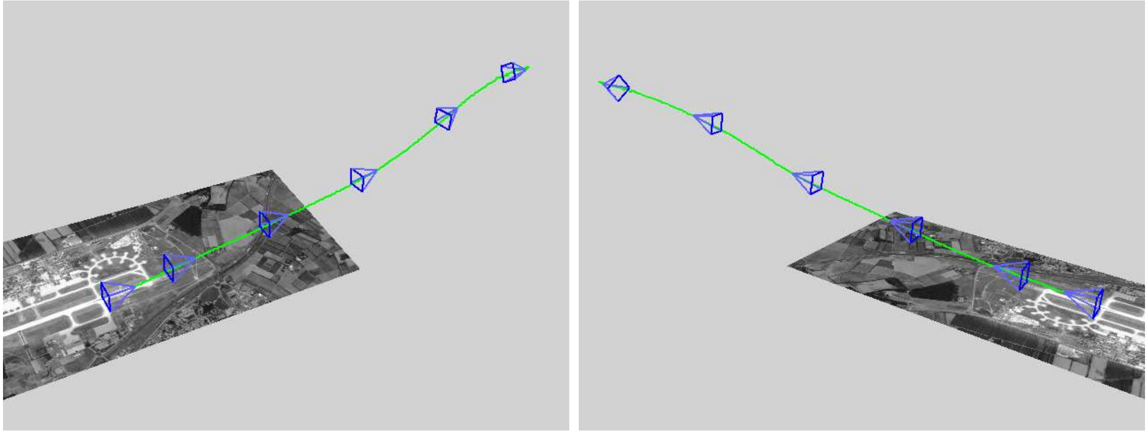


Figure 4.36: From the homography to the estimation of the camera position. Green curve: estimated camera trajectory in the 3D space, blue: the 6 estimated camera positions corresponding to the frames represented in figure 4.35.

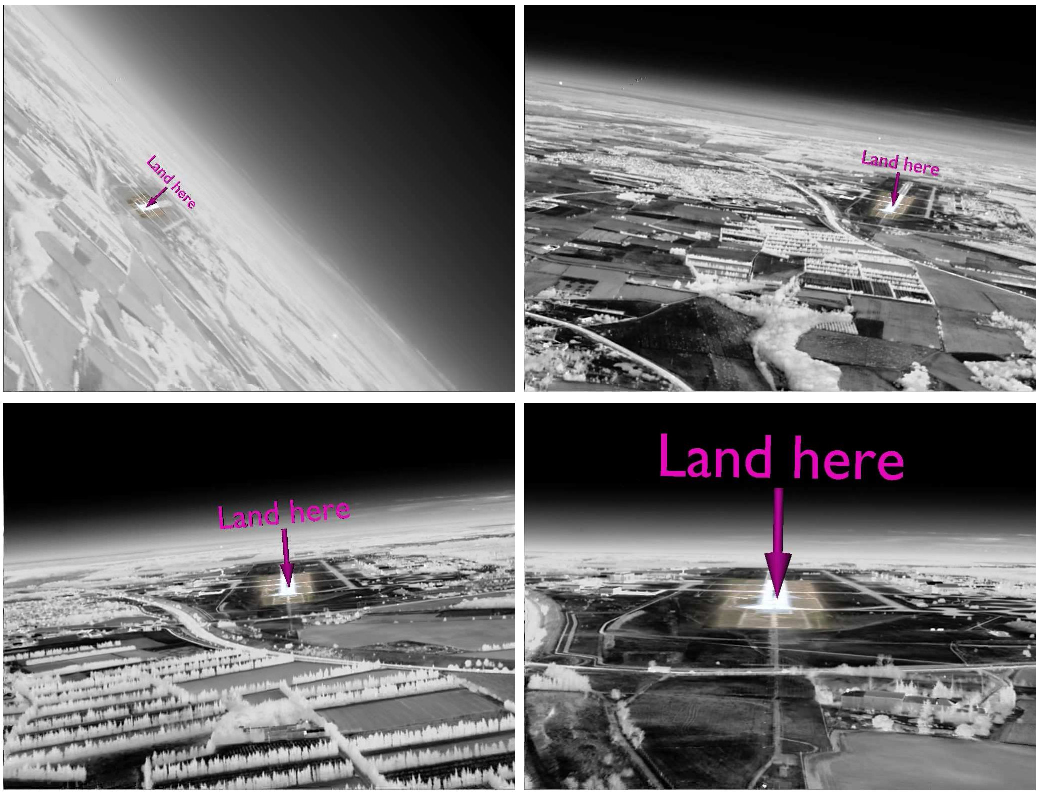


Figure 4.37: Augmenting the infrared images with the satellite appearance of the runway and an additional "land here" sign.

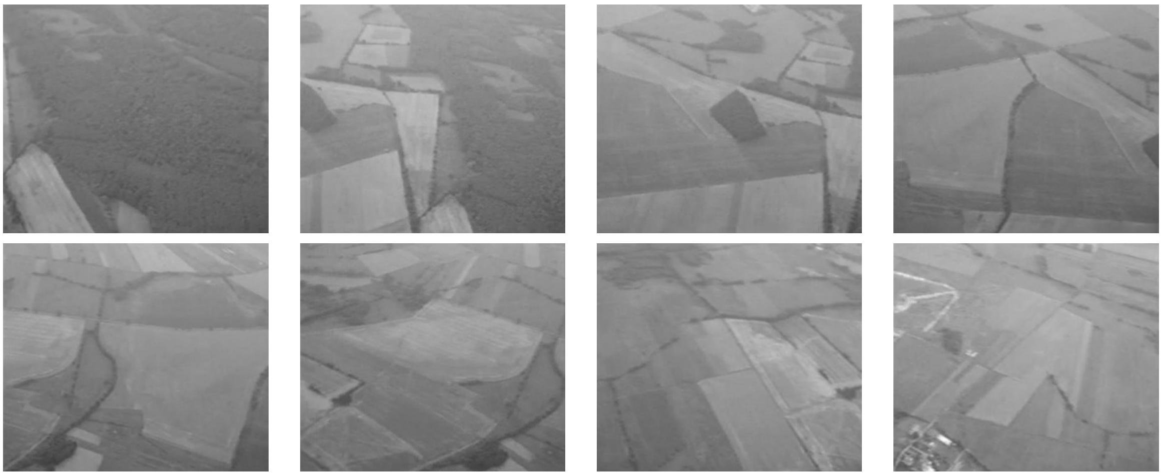


Figure 4.38: Some overlapping key images used for the mosaicing application.

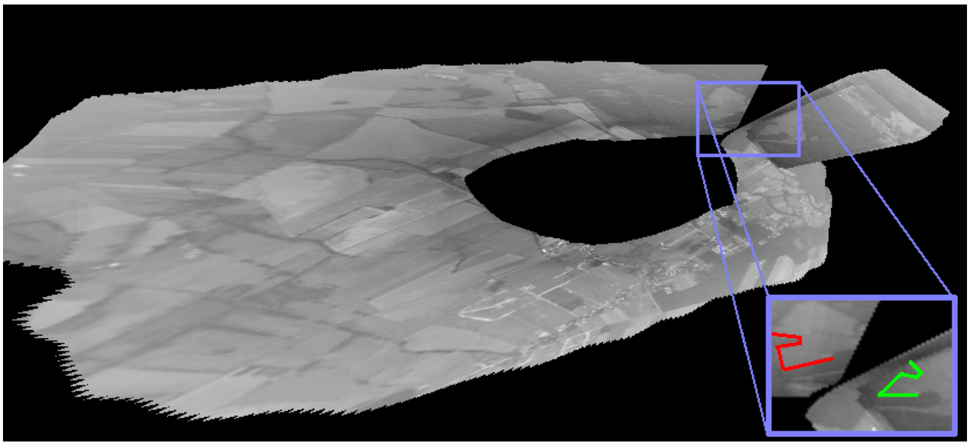


Figure 4.39: Resulting mosaic image: despite the poor quality of the sequence and the approximation that the scene is planar, the final displacement between the first and last image is accurate. The red and green contours show the position of one physical pattern in the first and last images of the sequence.

4.4 Conclusion

This chapter has presented a robust and accurate template based-tracker that was defined using a new approach based on the mutual information function. A reference image representing the object is used to track the object along a sequence of images. At each input image, the algorithm estimates the position of the object that maximizes the mutual information between the current image and the warped reference image. Many experiments including reference benchmarks shows the proposed approach to be very efficient:

- The proposed definition uses the advantages of MI with respect to its robustness toward occlusions, illumination variations and images from different modalities.
- No geometrical features are required, the only condition is to have a textured object. Using this texture in our approach provides a very accurate estimation of the object or camera displacement. This accuracy is demonstrated in several augmented reality applications.
- Our contributions include the creation of a new optimization approach that is defined to deal with the quasi-concave shape of MI. This approach is taking advantage of both the wide convergence domain of MI and its accurate maximum and, besides, the approach is computationally efficient.
- We also propose algorithm improvements that greatly reduce the computation time of the optimization using a selection on the reference pixels that yields to an accurate, fast and robust tracker.

More generally, our tracking approach is here only applied to planar rigid objects, future works include tracking approaches of deformable objects. In this chapter we limit the optimization method to the mutual information function. Nevertheless, this approach can also be applied to every quasi-concave or quasi-convex cost function. Such method is not necessary in the SSD minimization problem for which the Gauss-Newton method has proved to be efficient. However, considering the NCC or ZNCC similarity functions, the proposed approach would be well suited.

Chapter 5

Mutual information-based pose estimation

The previous chapter presented how to determine the position of an object in the image. We have seen that some parametric motion models allow the estimation of the pose of the object in the Cartesian space. Nevertheless, this model is limited to planar objects. To extend the pose estimation to the other rigid objects, instead of considering warping functions, the projection of a 3D model is directly considered with respect to the relative pose between the camera and the model. The goal is then to find the pose that best register the model with the input images.

In this chapter, we present our new pose estimation method that is based on a virtual visual servoing approach. The optimization approach that was proposed in the previous chapter is modified to fit the present problem. Finally, the last section shows some applications that show the robustness and accuracy of the proposed pose estimation approach.

5.1 Pose estimation as a registration problem

In this section, we focus on a model-based pose estimation problem that assumes the model of the object (or the scene) to be known. Since the goal is to perform the registration of the model with respect to an image, it can be formulated as the optimization of a similarity function between the input image I^* and the projection of the model $\mathcal{M}$. If $\boldsymbol{\gamma}$ are the intrinsic camera parameters and $\mathbf{r}$ its pose (extrinsic parameters), the registration problem can be written as:

$$\widehat{\mathbf{r}} = \arg\max_{\mathbf{r}} f\left(I^*, pr_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r})\right). \tag{5.1}$$

We assume that the intrinsic parameters are already estimated using the camera calibration. The camera pose $\mathbf{r}$ contains the parameters that define the homogeneous matrix involved in the projection of the model. This general formulation yields to many approaches. The classical one is to extract some features from the input image I^* and minimize the reprojection error between these features and the one of the model projected using $\mathbf{r}$ [Dementhon 1995, Drummond 2002, Comport 2006, Lim 1988, Lu 2000].

The considered approach differs from those classical approaches in the sens that there is no more features extraction. The mutual information function directly measures the similarity of appearance between the image I^* and the image $I_{\boldsymbol{\gamma}}$ resulting from the projection of the model [Panin 2008]:

$$\begin{aligned}\widehat{\mathbf{r}} &= \arg\max_{\mathbf{r}} \mathrm{MI}(I^*, I_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r})) \qquad (5.2)\\ &= \arg\max_{\mathbf{r}} \mathrm{MI}(I^*, I(\mathbf{r})).\end{aligned}$$

In this approach, it is therefore necessary to have the textured 3D model of the object and this model has to be projected for each pose of the camera $\mathbf{r}$ used in the iterative optimization process. Using the projection equations that have been defined in Section 1.1.1, it is possible to implement this projection. An optimized implementation of this projection process can be performed using any 3D renderer as OpenGL or DirectX.

To perform this maximization problem, several approaches are possible. We recall that the optimization on the pose parameters can be performed using either an additional or a compositional approach. In this part, we consider that the current parameters $\mathbf{r}$ define the estimated camera pose, and we update it using an increment $\mathbf{v}$ that is nothing but the velocity $\mathbf{v}$ of this virtual camera. This is exactly the problem that will be considered in the visual servoing problem, as the difference that the camera will be real. Since in this method the camera is virtual, it is generally called the virtual visual servoing approach [Sundareswaran 1998, Marchand 2002].

5.2 Mutual information on SE(3)

Since we know the variation of the point coordinates with respect to the camera pose, it is possible to study the resulting variation of MI and solve the registration problem based on the MI maximization.

The pose of the camera is defined by 6 parameters: 3 translations and 3 rotations. Independently, the rotations around the 3 axes of the camera frame have a very different effect on the image, as well as the 3 translations. However it is not the case if the rotations and translations are simultaneously considered. For instance we computed the mutual information with respect to the positioning error on the rotation around y and translation along x in Figure 5.1. The distance between the desired camera pose and the scene is 1 meter. We can see that the mutual information similarity function has a valley shape that follows $t_x = -r_y$. Indeed the two degree of freedom are strongly correlated. As we can see the image I^* acquired at the desired position is very similar to the image acquired with a positioning error of $(t_x, r_y) = (-0.1, 0.1)$. For the reasons of symmetry, it is similar if we consider the rotation around x with a translation along y.

This correlation can be analytically observed in the expression of the interaction matrix. Indeed if we focus only on the translation velocity ν_x and the rotation velocity ω_y of the interaction matrix, the position variation in the image gives:

$$\dot{\mathbf{x}} = \begin{bmatrix} -1/Z \\ 0 \end{bmatrix} \nu_x + \begin{bmatrix} -(1+x^2) \\ -xy \end{bmatrix} \omega_y \tag{5.3}$$

If x is small then the variation $\dot{\mathbf{x}}$ tends to be null for $\omega_y = -Z\nu_x$. If the camera is moving with a positioning error that satisfies $r_y = -Zt_x$, the variation in the image is very small, as well as the variation of the mutual information or any alignment function (the alignment functions have a valley shape).

Since the rotations and translations in $\mathbf{r}$ are correlated, a simple steepest descent optimization approach using the direction given by the interaction matrix related to MI would not provide an accurate estimation of the optimum of MI. Therefore, a second order optimization approach such as a Newton's like method is necessary.

5.2.1 MI based optimization

Since the link between the variation of the point position and the pose variation is known as well as the MI shape with respect to the camera pose, the goal is now to find the increment that changes the current image $I(\mathbf{r})$ to optimally increase its MI with the input image I^*.

Let f be any function depending on the camera pose $\mathbf{r}$. Using a first order Taylor expansion f at the current pose $\mathbf{r}^k$ in the visual servoing problem gives:

$$f(\mathbf{r}^{k+1}) \simeq f(\mathbf{r}^k) + \mathbf{L}_f^\top \dot{\mathbf{r}} \Delta t \tag{5.4}$$

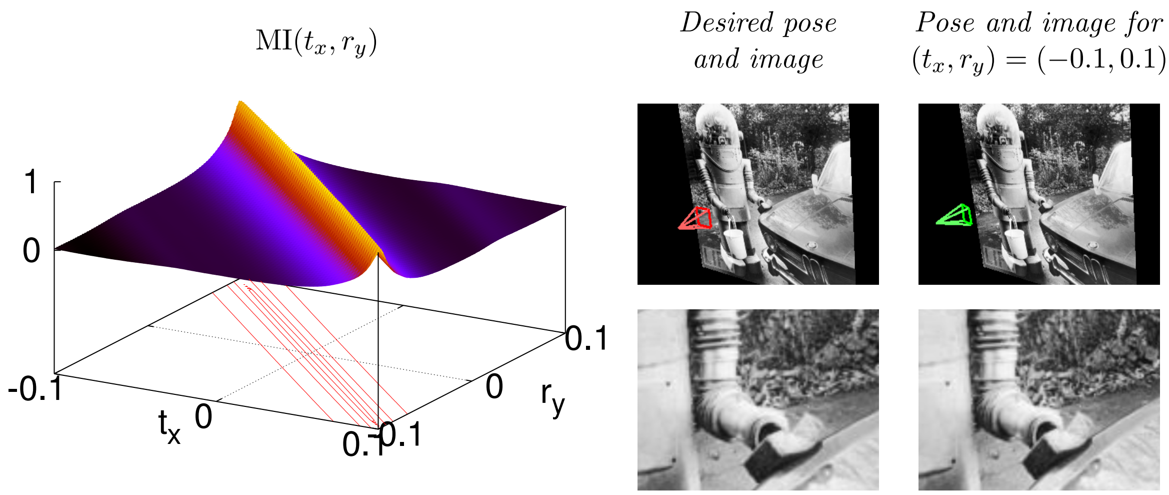


Figure 5.1: Correlation between the camera pose parameters. MI function has a valley shape. Indeed, the rotation r_y is strongly correlated with the translation t_x, as well as r_x and t_y. On the right we can observe its cause: the two transformations have almost the same effect on the image, so that two different camera poses can yield to similar images.

where δt is the lapse of time necessary to transform $\mathbf{r}^k$ into $\mathbf{r}^{k+1}$ using the pose variation $\dot{\mathbf{r}}$ (which can be seen as a virtual camera pose $\mathbf{v}$ with $\mathbf{v} = \dot{\mathbf{r}}$). The update rule of the pose is computed as follows:

$$\mathbf{r}_{k+1} \quad = \quad e^{[\mathbf{v}]}\,\mathbf{r}_k \tag{5.5}$$

where $e^{[\mathbf{v}]}$ is the exponential map of $\mathbf{v}$. $\mathbf{L}_g$ is the interaction matrix related to f, *i.e.* the matrix that links the variation of f with the pose variation [Chaumette 2006]. If f is replaced by $\mathbf{L}_{\mathrm{MI}}^{\top}$ the interaction matrix of MI, it yields:

$$\mathbf{L}_{\mathrm{MI}}^{\top}(\mathbf{r}^{k+1}) \simeq \mathbf{L}_{\mathrm{MI}}^{\top}(\mathbf{r}^{k}) + \mathbf{H}_{\mathrm{MI}}(\mathbf{r}^{k})\mathbf{v}\Delta t \tag{5.6}$$

where $\mathbf{H}_{\mathrm{MI}}$ is the interaction matrix of the interaction matrix of MI, that we call the MI Hessian matrix in an abuse of notation. Since the goal is to maximize the mutual information, we want to reach the pose $\mathbf{r}^{k+1}$ where the variation of MI with respect to the pose variation is zero ($\mathbf{L}_{\mathrm{MI}}(\mathbf{r}^{k+1}) = \mathbf{0}$). Let us simply set Δt to 1 second and drop it. Using the equation (5.6), the increment that leads to a null MI variation approximately respects:

$$\mathbf{v} = -\alpha \mathbf{H}_{\mathrm{MI}}^{-1}(\mathbf{r}^{k})\mathbf{L}_{\mathrm{MI}}^{\top}(\mathbf{r}^{k}) \tag{5.7}$$

where $\alpha \in [0, 1]$ is a scalar gain that control the convergence speed. Since the expression of the mutual information keeps the same quasi-concave shape as in the tracking problem, with the same coupling problem between the pose parameters, as it was the case between the parameters of the homography, we can assume that the study of the optimization approach used in the tracking problem is applicable to the pose estimation problem. Therefore, the increment of the pose is computed using the proposed HC approach:

$$\mathbf{v} = -\alpha {\mathbf{H}_{\mathrm{MI}}^{*}}^{-1}\mathbf{L}_{\mathrm{MI}}^{\top} \tag{5.8}$$

where $\mathbf{H}_{\mathrm{MI}}^{*}$ is the Hessian matrix estimated at the optimal position $\mathbf{r}^*$ ($\mathbf{H}_{\mathrm{MI}}^{*} = \mathbf{H}_{\mathrm{MI}}(\mathbf{r}^*)$) and $\mathbf{L}_{\mathrm{MI}}$ refers to the interation matrix related to MI computed at the current position $\mathbf{r}^k$. Of course the optimal pose $\mathbf{r}^*$ is unknown, nevertheless, we will see that, as in the tracking problem, the Hessian matrix at the optimum $\mathbf{H}_{\mathrm{MI}}^{*}$ can be estimated without knowing $\mathbf{r}^*$.

5.2.2 MI derivatives

Given the MI equation (2.15) page 27 and the chain rules simplifications [Dowson 2006] detailed in Appendix B.1, the expression of the Gradient and Hessian are:

$$\begin{aligned} \mathbf{L}_{\mathrm{MI}} &= \sum_{i,j} \mathbf{L}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) && (5.9) \\ \mathbf{H}_{\mathrm{MI}} &= \sum_{i,j} \mathbf{L}_{p_{II^*}}^{\top} \mathbf{L}_{p_{II^*}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_{I^*}}\right) + \mathbf{H}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) && (5.10) \end{aligned}$$

For the same reasons detailed in Section 4.1.1.2, we compute the full expression of the Hessian matrix. Considering the expressions of equations (5.9) and (5.10), all the required variables are known apart from the joint probability derivatives. From the equation (2.19), we deduce the joint probability expression and its variations with respect to the camera pose:

$$\begin{aligned} p_{II^*}(i,j,\mathbf{r}) &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \overline{I}(\mathbf{x},\mathbf{r})\right) \phi\left(j - \overline{I^*}(\mathbf{x})\right) && (5.11) \\ \mathbf{L}_{p_{II^*}(i,j,\mathbf{r})} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \mathbf{L}_{\phi\left(i - \overline{I}(\mathbf{x},\mathbf{r})\right)} \phi\left(j - \overline{I^*}(\mathbf{x})\right) \\ \mathbf{H}_{p_{II^*}(i,j,\mathbf{r})} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \mathbf{H}_{\phi\left(i - \overline{I}(\mathbf{x},\mathbf{r})\right)} \phi\left(j - \overline{I^*}(\mathbf{x})\right) \end{aligned}$$

The variation of ϕ can be obtained using the chain rule:

$$\begin{aligned} \mathbf{L}_{\phi\left(i - \overline{I}(\mathbf{x},\mathbf{r})\right)} &= -\frac{\partial \phi}{\partial i} \mathbf{L}_{\overline{I}} && (5.12) \\ \mathbf{H}_{\phi\left(i - \overline{I}(\mathbf{x},\mathbf{r})\right)} &= \frac{\partial^2 \phi}{\partial i^2} \mathbf{L}_{\overline{I}}^{\top} \mathbf{L}_{\overline{I}} - \frac{\partial \phi}{\partial i} \mathbf{H}_{\overline{I}} && (5.13) \end{aligned}$$

If we make the assumption that the scene is Lambertian that is correct for a very small displacement, then the interaction matrix of the intensity of a point $\mathbf{L}_{\overline{I}}$ and its Hessian $\mathbf{H}_{\overline{I}}$ are found using [Collewet 2008b]:

$$\begin{aligned} \mathbf{L}_{\overline{I}} &= \nabla \overline{I}\, \mathbf{L}_{\mathbf{x}} && (5.14) \\ \mathbf{H}_{\overline{I}} &= \mathbf{L}_{\mathbf{x}}^{\top}\, \nabla^2 \overline{I}\, \mathbf{L}_{\mathbf{x}} + \nabla_x \overline{I} \mathbf{H}_x + \nabla_y \overline{I} \mathbf{H}_y && (5.15) \end{aligned}$$

where $\nabla \overline{I} = (\nabla_x \overline{I}_m, \nabla_x \overline{I}_m)$ are the image gradients, $\nabla^2 \overline{I} \in \mathbb{R}^{2\times 2}$ are the gradients of the image gradients and $\mathbf{L}_{\mathbf{x}}$ is the interaction matrix of a point that links its displacement in the image plan to the camera velocity. $\mathbf{H}_x$ and $\mathbf{H}_y$ are the Hessians of the two coordinates of the point with respect to the camera velocity. Since the variations of position of the point provided by the interaction matrix and Hessian matrix are, in general, given in meters, the image gradients must also be expressed in meters. This convertion from the pixel to the metric space of the image gradients is performed using the camera intrinsic parameters p_x and p_y using $\nabla \overline{I} = (p_x \nabla_x \overline{I}, p_y \nabla_y \overline{I})$. The interaction matrix is given by (see the details in Frame 9):

$$\mathbf{L}_{\mathbf{x}} = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix}.$$

This interaction matrix depends on both the position (x, y) of the point in the image plane and its depth Z in the camera frame. In the pose estimation problem, since the image is the projection of our 3D model, it is possible to get the depth image or depth map using the object's

Frame 9 Interaction matrix computation.

Let us note $\dot{x}$ the derivative of the variable x with respect to the time. We recall that a point ${}^{c}\mathbf{X} = (X, Y, Z)$ expressed in the camera frame is projected in the image plan using $\mathbf{x} = (X/Z, Y/Z)$ ($\mathbf{x}$ is expressed in the metric space). To obtain the derivative of the position of a pixel, we differentiate this expression [Chaumette 2006]:

$$\left\{ \begin{array}{l} \dot{x} = \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z \\ \dot{y} = \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{Y} - y\dot{Z})/Z \end{array} \right. \tag{5.16}$$

Where we see that the derivative of the pixel location is depending on the derivative of the 3D coordinates of the physical point $\mathcal{X}$ in the camera frame. Since the physical point is supposed motionless in the world frame, the derivative of its coordinates are only depending on the camera velocity $\mathbf{v} = (\boldsymbol{\nu}, \boldsymbol{\omega})$ and are given by:

$$\dot{\mathbf{X}} = -\boldsymbol{\nu} - \boldsymbol{\omega} \times \mathbf{X} \Longleftrightarrow \left\{ \begin{array}{l} \dot{X} = -\nu_x - \omega_y Z - \omega_z Y \\ \dot{Y} = -\nu_y - \omega_z X - \omega_x Z \\ \dot{Z} = -\nu_z - \omega_x Y - \omega_y X \end{array} \right. \tag{5.17}$$

where × denotes the cross product operation. Combining the two equations (5.16) and (5.17) the displacement of a point in the image can be expressed with respect to the camera velocity:

$$\left\{ \begin{array}{l} \dot{x} = -\nu_x/Z + x\nu_z/Z + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\ \dot{y} = -\nu_y/Z + y\nu_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \end{array} \right. \tag{5.18}$$

This equation is linear and can therefore be rewritten as:

$$\dot{\mathbf{x}} = \mathbf{L_x}\mathbf{v} \tag{5.19}$$

with

$$\mathbf{L_x} = \left[\begin{array}{cccccc} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{array} \right]. \tag{5.20}$$

$\mathbf{L_x}$ is called the interaction matrix that links the displacement of a point in the image to the velocity of the camera. We see that it only depends on the coordinates of the point (x, y) in meters and the depth Z of the point in the camera frame.

model and the perspective geometry, or directly using the Z-buffer that is usually available in the 3D renderers.

The Hessian matrix of the point is given by [Lapresté 2004]:

$$
\begin{aligned}
\mathbf{H}_x &= \begin{bmatrix}
0 & 0 & -V^2 & 0 & 2xV & -yV \\
0 & 0 & 0 & -V & 0 & -xV \\
-V^2 & 0 & 2xV^2 & yV & -2x^2V & 2xyV \\
0 & -V & yV & -x & -2xy & y^2-x^2 \\
2xV & 0 & -2x^2V & -2xy & 2x(1+x^2) & -y(1+2x^2) \\
-yV & -xV & 2xyV & y^2-x^2 & -y(1+2x^2) & x(2y^2+1)
\end{bmatrix} \\
\mathbf{H}_y &= \begin{bmatrix}
0 & 0 & 0 & V & yV & 0 \\
0 & 0 & -V^2 & 0 & xV & -2yV \\
0 & -V^2 & 2yV^2 & -xV & -2xyV & 2y^2V \\
V & 0 & -xV & -y & x^2-y^2 & -2xy \\
yV & xV & -2xyV & x^2-y^2 & y(2x^2+1) & -2xy^2 \\
0 & -2yV & 2y^2V & -2xy & -2xy^2 & 2y(1+y^2)
\end{bmatrix}
\end{aligned}
$$

where $V = 1/Z$. During the iterative process, the virtual camera moves, causing the depth of each point to change. The interaction matrix and Hessian matrix are therefore changing at every iteration. The HC optimization method, that consists in using the Hessian matrix at the optimum $\mathbf{r}^*$ of MI, needs the depth of each point Z^* at the optimum that is unknown. To perform the estimation, we assume that the depth of the points between the current and the desired pose is slighly changing and we set the depth Z^* equal to the current depth Z. To improve the convergence of the optimization, the depths of each point Z^* and the Hessian matrix $\mathbf{H}^*$ are updated while the algorithm is converging. Therefore, at convergence, the estimation of $\mathbf{H}^*$ will be accurate.

5.3 Experimental results

This section presents two pose estimation experiments using the proposed MI-based approach. To show its capability to estimate the pose of a known object, one first pose computation experiment is performed in simulation so that the ground truth is known and that the quality of the pose estimation can be evaluated. A second experiment presents the proposed approach in real conditions.

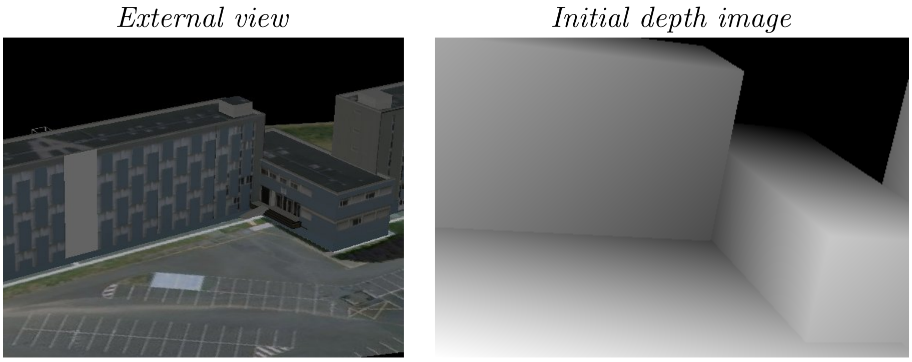
5.3.1 Simulation results

In this experiment, the virtual outdoor environment presented in Figure 5.2 is used to acquire a reference image shown in Figure 5.3. The desired pose of the camera is learned as ground truth. The virtual camera is then moved to a new pose. Then the virtual visual servoing task is performed begining from this initial pose.

As we can see in Figure 5.3, the positioning error is decreasing to reach a very accurate estimation of the pose of the camera with a final image error that depicts a null error (uniform gray image). The final positioning error is about 0.02 meters in translation and 0.01^o. Considering that the distance between the camera and the scene is about 50 meters, the accuracy is very significant.

5.3.2 Real environment

To show the application of the approach in real conditions a pose computation experiment is performed on a real indoor image. The input image has been acquired with a calibrated camera.



External view *Initial depth image*

Figure 5.2: Localization experiment in an urban environment.

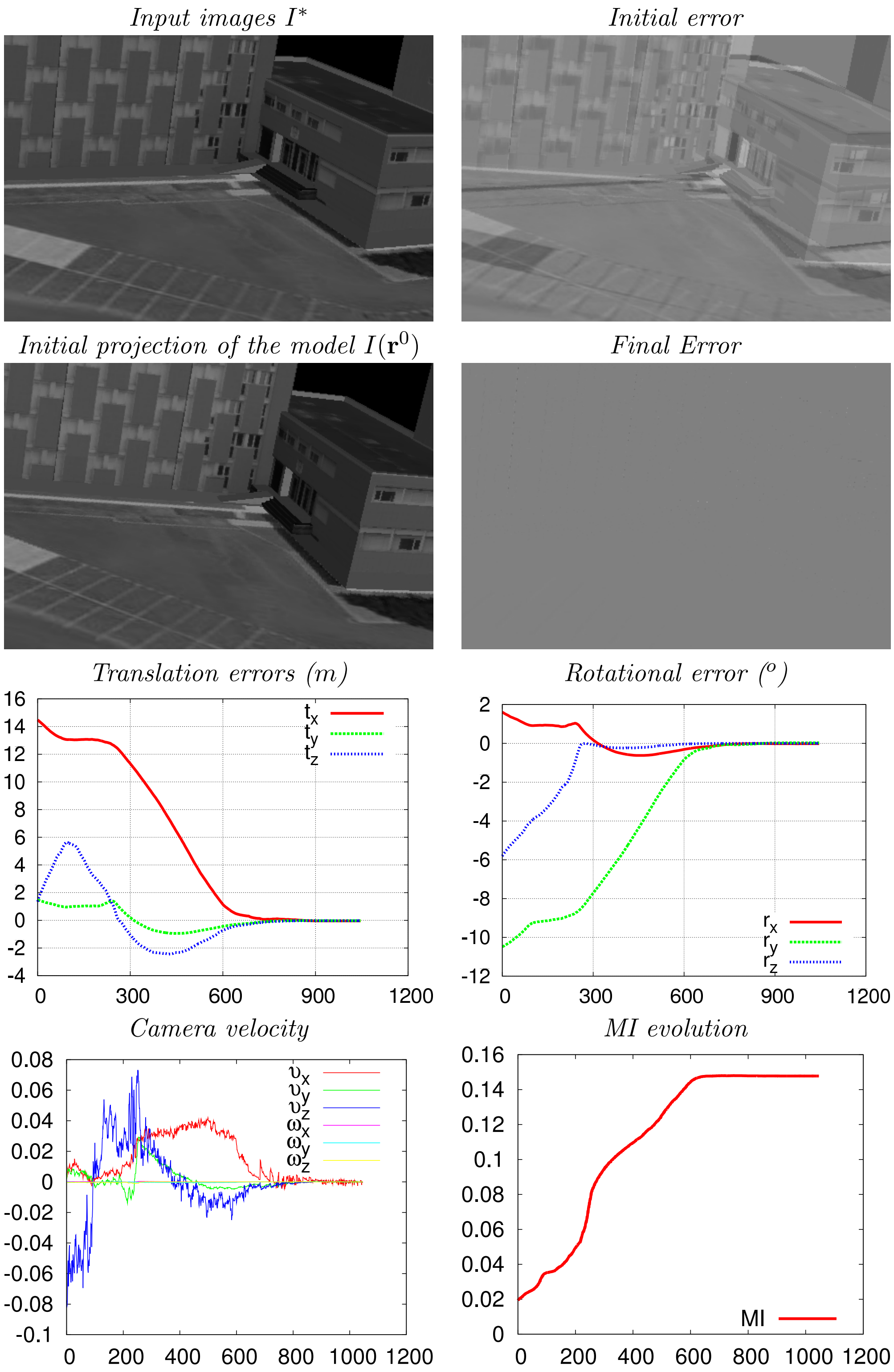
A coarse model of the object has been created. No ground truth is available so that the results are only judged using the final image error.

Figure 5.4 shows the reference image (the acquired image) and the initial image resulting from the model projected at the initial virtual camera pose. We can see that the initial pose is far from the desired pose. Figure 5.5 shows the evolution of pose estimation in the registration process. Since the illumination conditions of the real scene are unknown and different from the conditions in the simulator, the final image error does not show a null error, nevertheless it is clear that the two images are well matched since the edges are in the same position in the error image. In spite of the large initial positioning error and the different illumination conditions from the reference to the rendered image the pose estimation task converges and provides a good estimation of the camera pose (or object pose).

5.4 Conclusion

In this chapter, our contribution is to propose a new way to estimate the pose of an object in an image using the mutual information between the image and the projection of a 3D model. The advantages of this approach are its robustness to appearance variations and its accurate estimation of the camera pose that it provides. We observe that the approach requires to render the current image for each pose of the camera during the optimization. The process is currently time consuming due to rendering process and, therefore, the whole approach can for the moment not be performed at video rate.

Nevertheless, also more experiments have to be performed, the experimental results presented here are really promising for many applications. Since 3D models of most of the cities are created, a very interesting application will be to use the proposed approach for vehicle localization and visual odometry. The approach would be perfectly suited due to the robustness of MI with respect to illumination variations and occlusions.



Input images I^*

Initial error

Initial projection of the model $I(\mathbf{r}^0)$

Final Error

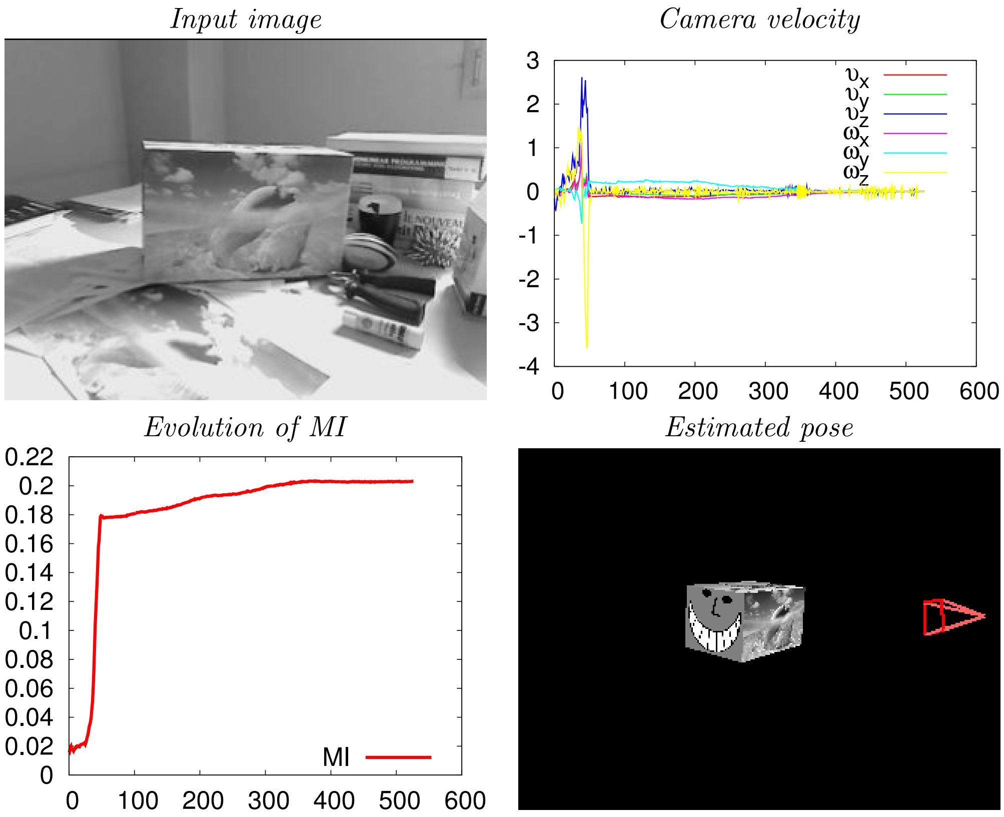
Translation errors (m)

Rotational error (^{o})

Camera velocity

MI evolution

Figure 5.3: Virtual visual servoing using mutual information with a simulation. The image error and positioning error show that the estimation of pose is very accurate.



Input image

Camera velocity

Evolution of MI

Estimated pose

Figure 5.4: Pose computation experiments on a real indoor image: input image, camera velocity (in $m.s^{-1}$ and $rad.s^{-1}$), MI with respect to the iteration number and final estimated pose relative to the 3D model.

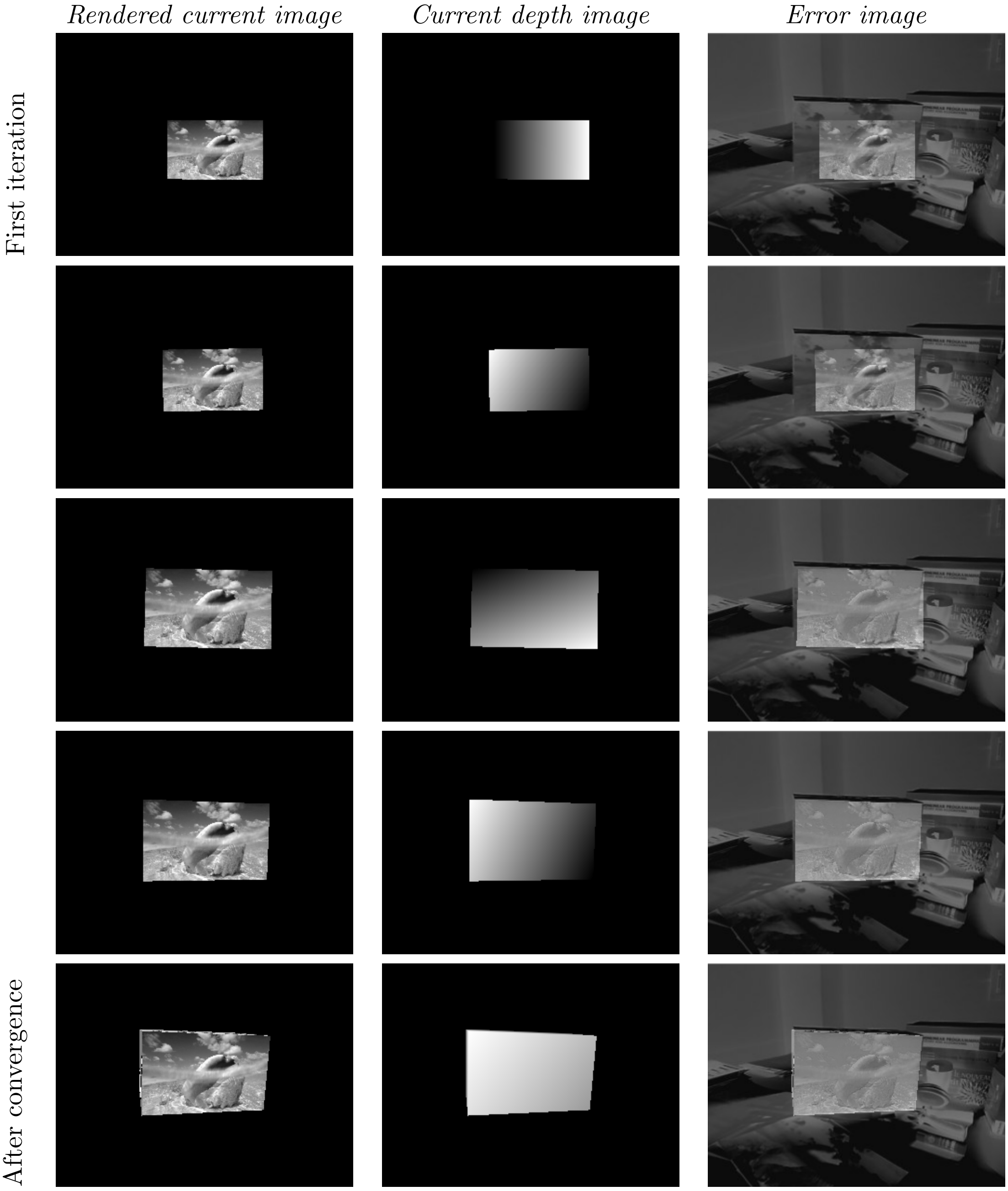


Figure 5.5: Evolution of the registration process.

Part III

Visual Servoing

Overview

The integration of sensors in the area of robotics has created many new possibilities. As the sens of sight is very important in the human perception, from all robots sensors (ultrasonic, infrared, accelerometers, force sensors), visions-sensors greatly improve the adaptability of robotic systems. Visual servoing consists in controlling a robot using the information provided by one or multiple cameras that can either be mounted on the robot or that observes it [Hashimoto 1993, Hutchinson 1996, Chaumette 2006]. From the classical positioning task to more complex navigation tasks, the applications are widespread in the industrial, medical, space, defense domains and many others.

Most of these applications require a positioning task that is accurate, robust to the appearance variations of the scene and that is responsive (see the Figure 5.6). In this thesis, we focus on the case when one camera is mounted on the robot (also called monocular eye-in-hand configuration). We begin this part with a partial state of the art on the different visual servoing scheme. Along with the classical approaches that use geometric features, we present the direct approaches (that does not require any matching nor tracking steps) and show the interest of our new method based on mutual information. Since visual servoing and pose computation (defined in the previous chapter) are two dual problems, we will briefly define the mutual information-based visual servoing task and will focus on the experimental validation of the proposed approach. The new approach is evaluated on many experiments using a 6 dof gantry robot and compared with other visual servoing methods. The classical visual servoing task is finally adapted to a navigation problem. We will show its efficiency on outdoor experiments carried out using a non-holonomic robot.

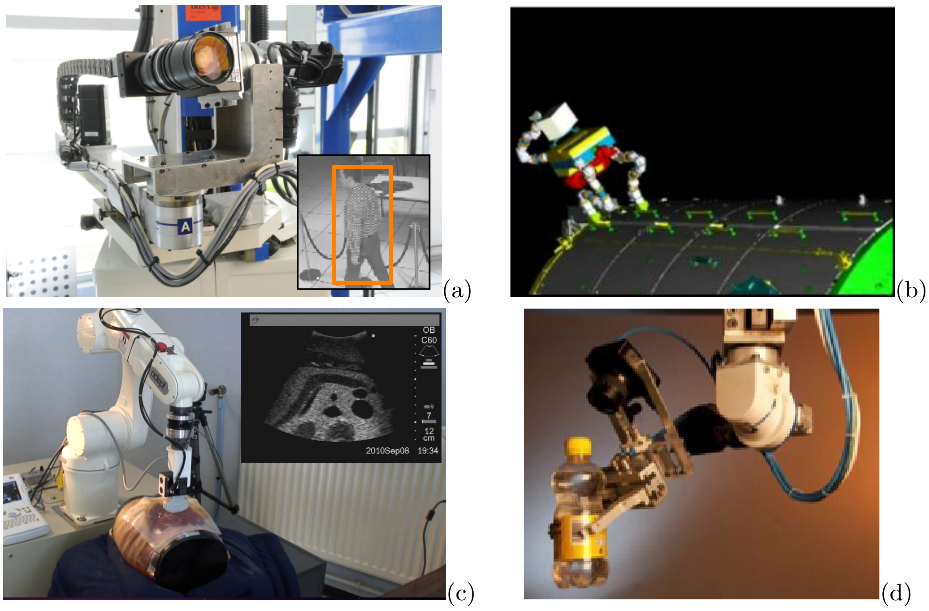


Figure 5.6: Some illustrations of visual servoing applications: (a) people tracking [Crétual 1998], (b) Eurobot walking on ISS (image courtesy of ESA) [Dionnet 2007], (c) ultrasound visual servoing [Nadeau 2010], (d) bottle grasping.

Chapter 6

From geometric to direct visual servoing

In this section, we recall the general principle of a visual servoing task and some of the solutions proposed in the literature. The goal is not to provide an exhaustive state of the art, but is simply to present the main visual servoing approaches to introduce and show what is original in our own method.

6.1 Positioning tasks

A visual servoing task involves using the information provided by a camera to control the displacement of a dynamic system. This positioning problem can always be written as the optimization of a cost function. From an initial arbitrary pose $\mathbf{r}_0$, the camera has to reach the desired pose $\mathbf{r}^*$ that better satisfies some properties measured in the images. If we note f, the function that measures the positioning error, then the visual servoing task can be written as:

$$\widehat{\mathbf{r}} = \arg\min_{\mathbf{r}} f(\mathbf{r}, \mathbf{r}^*). \tag{6.1}$$

The visual servoing problem can, therefore, be considered as an optimization of the function f where $\mathbf{r}$ is incrementally updated to reach an optimum of f at $\widehat{\mathbf{r}}$ (if f is correctly chosen at the end of the minimization we should have $\widehat{\mathbf{r}} = \mathbf{r}^*$). The pose update is performed by applying a velocity $\mathbf{v}$ to the camera that is mounted on a robot. Depending on the way to build the cost function f, we separate the visual servoing approaches in two categories: the ones that require a matching or/and tracking step between visual features to define the measurement error and the ones that do not.

6.1.1 Visual servoing using geometric features

Most of the visual servoing approaches use 2D, 2D1/2 or 3D information extracted from the image to control the position of the robot $\mathbf{r}$. The information is parametrized into a vector of visual features $\mathbf{s}(\mathbf{r})$ that depends on the pose $\mathbf{r}$. The goal of the task is then to minimize the error between the desired features $\mathbf{s}^*$ and the current features $\mathbf{s}(\mathbf{r})$ extracted from the current image. The error is usually defined as the difference between the two vectors $\mathbf{s}^*$ and $\mathbf{s}(\mathbf{r})$:

$$\widehat{\mathbf{r}} = \arg\min_{\mathbf{r}} \mathbf{e}(\mathbf{r}) \qquad \text{with:} \qquad \mathbf{e} = \mathbf{s}(\mathbf{r}) - \mathbf{s}^* \tag{6.2}$$

Since the desired features are constant, the variation of the error is equal to the variation of the current features ($\dot{\mathbf{s}} = \dot{\mathbf{e}}$). To evaluate the evolution of the error with respect to time, the time variation of the features is then considered. This variation is defined by:

$$\dot{\mathbf{s}} = \mathbf{L_s}\mathbf{v} \tag{6.3}$$

where $\mathbf{v} = (\boldsymbol{\nu}, \boldsymbol{\omega})$ is the velocity of the camera and $\mathbf{L_s}$ is the interaction matrix that links the variation of the features to the velocity of the camera. Several control laws can be considered.

The classical solution is to require an exponential decrease of the error, so that when the camera is far from the desired position it moves faster. On the contrary, when it reaches the desired pose, it converges to a null velocity. To have an exponential decrease, the error must satisfy

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}$$

where λ is a scalar factor. The velocity has then to satisfy:

$$\mathbf{L_s v} = -\lambda \mathbf{e} \tag{6.4}$$

where the resolution of the problem is performed using an iterative least square approach:

$$\mathbf{v} = -\lambda \mathbf{L_s^+ e} \qquad \text{with:} \qquad \mathbf{L_s^+} = (\mathbf{L_s^\top L_s})^{-1} \mathbf{L_s^\top} \tag{6.5}$$

where the velocity $\mathbf{v}$ is recomputed for each input image. $\mathbf{L_s^+}$ is called pseudo-inverse of the matrix $\mathbf{L_s}$. Several choices of visual features $\mathbf{s}$ have been proposed.

6.1.1.1 3D visual servoing

A first solution is to define the error directly in the Cartesian space and the features as the pose of the object (environment) with respect to the camera (or with respect to a reference frame). $\mathbf{s}$ is then defined using the parametrization of the transformation between the current and desired pose. Since the interaction matrix is a full rank 6×6 matrix of rank 6, the pseudo-inverse of the interaction matrix is equivalent with its inverse. The velocity is then directly obtained using [Wilson 1996, Chaumette 2006]:

$$\mathbf{v} = -\lambda \mathbf{L_s^{-1} e} \tag{6.6}$$

The main drawback of this approach is that it requires to estimate the pose of the object (see the pose computation problem in the previous chapter). This estimation is a difficult task and requires to have the intrinsic parameters of the camera and a 3D model of the object. An other issue is that the resulting control law defines an optimal trajectory in the Cartesian space but gives no guaranty to keep the visibility of the visual features in the image (necessary to solve the pose estimation problem). To keep the visibility of the features some approaches have been proposed to couple the visual servoing task with a trajectory planification task [Mezouar 2001]. Nevertheless, the combination of the two tasks becomes complex.

6.1.1.2 2D visual servoing

Feature-based approaches

The 2D visual servoing approaches minimize the positioning error in the image space. A widespread approach is to define the features as geometrical patterns in the images such as points, lines or moments. The error is then defined as the distance between these current features and their desired positions (see the 2D feature-based visual servoing in Figure 6.1: the error to minimize is represented by the green segments between the desired features in red and current features in blue). Since a large number of visual features allows a more accurate positioning task, it is common to consider several visual features.

A typical visual servoing approach based on 4 dots is detailed in the Frame 10. At the beginning of the task, the current and desired features have to be matched. Then, at each iteration of the control law, the current features are tracked over the frames. The control law is defined to minimize the visual error between the current and desired features. If the interaction matrix is correctly estimated, the current features are supposed to move from their initial position to their desired position following a straight line in the image. The current features should, therefore, remain in the image and, contrary to the 3D approaches, the risk to have them outside the image decreases. Nevertheless, this implies that the trajectory of the robot in the Cartesian space is not optimal. For example, a large rotational positioning error around the z axis induces a large translational motion of the camera along the z axis [Chaumette 1997]. This is why several researches have been proposed to best decouple the degrees of freedom of the camera using, for example, image moments [Tahri 2005], second order control laws [Corke 2001, Lapresté 2004, Malis 2004] or introducing some 3D information in the features as in the 2D 1/2 visual servoing approach [Chaumette 2000].

The main problem of these approaches is their dependence to particular geometrical features that have to be chosen for each object and extracted from the image. Moreover, the robustness of the task depends on the matching and tracking steps. And finally, the positioning error is computed using the features that are subject to measurement errors which may limit the accuracy of the positioning task.

Toward direct approaches

To solve some of these problems, more global visual features can be considered. Indeed the desired feature can be simply defined as the image acquired at the desired pose of the camera. The goal of the visual servoing task simply becomes to move the camera in order to align the current image with the reference image using the motion of the camera.

Several solutions have been proposed that use the homography between the current image acquired by the camera and the reference image. One first solution proposed in [Vargas 2005] was to decompose the homography matrix to evaluate the 3D displacement between the current and desired pose. The estimated transformation was then used in a position-based visual servoing task. Later, the solution presented in [Benhimane 2007] proposed to use the homography, this time, without any decomposition. In these approaches, a first matching step is manually or automatically performed, then a tracking algorithm [Benhimane 2004] estimates the homography that is used as input of the control law. In this case, the positioning error is no longer defined as a difference between the desired and current features, but similarly as a distance between the current estimated homography and the identity matrix. An illustration of this approach is proposed in Figure 6.1, where the current position of the template is represented in blue and its desired position is represented in red.

Following a similar principle, [Crétual 1998] proposes a dynamic visual servoing approach where the feature is defined as a global motion model (parametrized optical flow). The current optical flow is computed using a tracking step [Odobez 1995], and the goal of the control law is to minimize the error between the current motion parameters and a desired one.

The main drawback of these approaches is that a tracking step is always necessary to estimate either the current homography or the optical flow at each iteration of the visual servoing task.

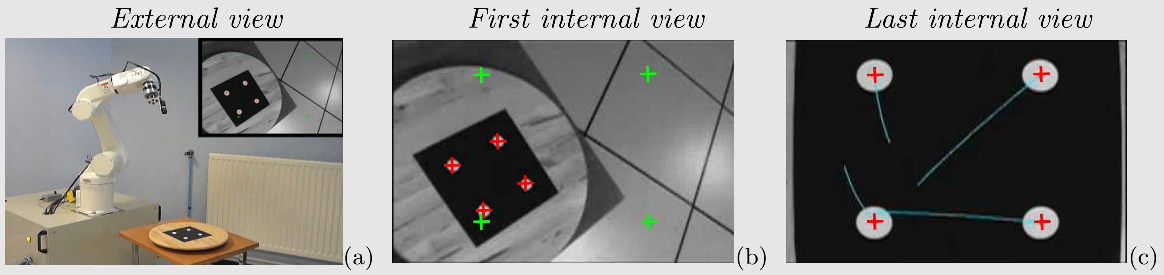
6.1.2 Direct visual servoing

Recently, it has been shown that no information other than the image intensity can be considered to control the robot motion. Instead of trying to reach some desired geometric features, the goal can be formulated as finding the velocity $\mathbf{v}$ that makes the current image $I(\mathbf{r})$ reach a desired image I^*. The optimization problem of the visual servoing task can simply be rewritten:

$$\widehat{\mathbf{r}} \;=\; \arg\min_{\mathbf{r}} \mathcal{C}(I(\mathbf{r}), I^*) \tag{6.7}$$

where $\mathcal{C}$ directly measures the dissimilarity between the images $I(\mathbf{r})$ and I^*. [Deguchi 2000, Nayar 1996] consider the full image, but in order to reduce the dimension of image data, they

Frame 10 Example of image-based visual servoing with 4 dots [Chaumette 2006].


External view (a) *First internal view* (b) *Last internal view* (c)

The goal of the visual servoing problem using 4 points is to make the current position of the points $\mathbf{x}_i = (x_i, y_i)$ (represented in red) reach the desired positions $\mathbf{x}_i^* = (x_i^*, y_i^*)$ (in green) with $i \in [0..3]$. The features vectors are then simply defined using the coordinates of the dots:

$$\begin{aligned}\mathbf{s} &= (\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)\\ \mathbf{s}^* &= (\mathbf{x}_0^*, \mathbf{x}_1^*, \mathbf{x}_2^*, \mathbf{x}_3^*)\end{aligned}$$

The 6 dof positioning problem of the robot in the Cartesian space is formulated as the minimization of the 2D position error $\mathbf{e}$ between the current features $\mathbf{s}(\mathbf{r})$ and the desired features $\mathbf{s}^*$:

$$\mathbf{e} = \mathbf{s}(\mathbf{r}) - \mathbf{s}^*$$

To have an exponential decrease of the error, the velocity of the camera is computed as:

$$\mathbf{v} = -\lambda \mathbf{L}_{\mathbf{s}}^{+} \mathbf{e}$$

where $\mathbf{L}_{\mathbf{s}}$ is the interaction matrix that links the variation of the feature $\mathbf{s}$ with the camera velocity $\mathbf{v}$. Since $\mathbf{s}$ is obtained with a concatenation of the point coordinates, its interaction matrix is computed with a concatenation of the interaction matrix of the points:

$$\mathbf{L}_{\mathbf{s}} = \begin{bmatrix} \mathbf{L}_{\mathbf{x}_0} \\ \mathbf{L}_{\mathbf{x}_1} \\ \mathbf{L}_{\mathbf{x}_2} \\ \mathbf{L}_{\mathbf{x}_3} \end{bmatrix}$$

with:

$$\mathbf{L}_{\mathbf{x}} = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix}.$$

The details of the computation of $\mathbf{L}_{\mathbf{x}}$ can be found in the Frame 9 page 105. Since the current depth of the points Z are unknown, it is common to use the interaction matrix of the point computed at the desired position $\mathbf{L}_{\mathbf{x}}^*$ instead of $\mathbf{L}_{\mathbf{x}}$ in the control law. Using this approach, the current positions of the points are reaching the desired positions following the blue trajectories represented in the top right figure (since the interaction matrix is approximated by $\mathbf{L}_{\mathbf{x}}^*$, the trajectories are not perfectly straight).

consider an eigenspace decomposition of it. The control is then performed directly in the eigenspace which requires the off-line computation of this eigenspace (using a principal component analysis) and then, for each new frame, the projection of the image on this subspace. Moreover, the interaction matrix related to the eigenspace is not computed analytically but learned. To cope with these issues a way to compute the interaction matrix related to the luminance under temporal luminance constancy case has been proposed in [Collewet 2008b] (see Frame 11 for more details). In that case, the function $\mathcal{C}$ to be regulated is nothing but the SSD between the current and a reference image. [Kallem 2007] also consider the pixels intensity. This approach is based on the use of kernel methods that lead to a high decoupled control law. However, only the 3 translations and the rotation around the z axis are considered. Another approach that does not require tracking nor matching has been proposed in [Abdul Hafez 2008]. It models collectively feature points extracted from the image as a mixture of Gaussian and attempts to minimize the distance function between the Gaussian mixture at the current and desired poses. Simulation results show that this approach is able to control a 3 degrees of freedom robot. However, note that an image processing step is still required to extract the current feature points.

A representation of these approaches is given in Figure 6.1 (3^{rd} row). This figure shows the simplicity of these methods where the control law is simply defined using the comparison between the patch of intensities of the current and desired images at the same pixel locations. Such approaches have the advantage of their accuracy in the nominal conditions. Nevertheless, the existing approaches are all very sensitive to non global illumination variations. Although using an illumination model is possible as in [Collewet 2008a], this approach remains limited to some specific conditions.

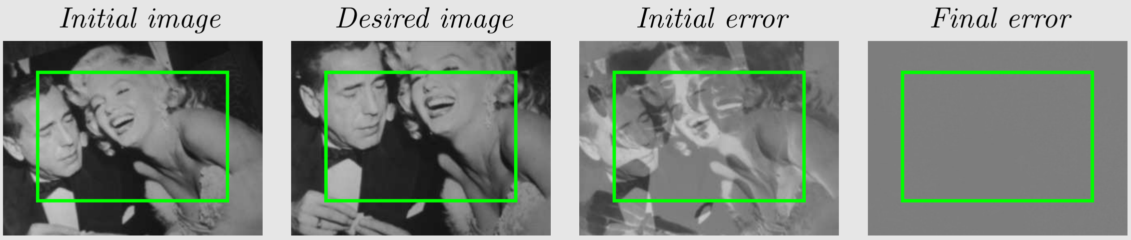
6.2 Navigation tasks

In recent years, robot localization and navigation have made considerable progress. Navigation can be seen as the ability for a robot to move autonomously from an initial position to a desired one (which may be far away from the initial one). Thanks to sensor-based navigation, we have seen autonomous robots in various challenging areas (from highways to deserts and even on Mars). Nevertheless, the design of these autonomous robots usually relies on more than one sensor (camera, stereo sensors, lidar, GPS,...). In this section, we discuss few researches related to non-holonomic vehicle navigation using a monocular camera. As in the visual servoing problem, the different approaches can be divided into two classes: the position-based visual navigation methods, where the control law is directly defined using the Cartesian space and where a 3D model of the environment has to be known or reconstructed, and the image-based methods, that are conceptually simpler since the task is directly defined in the image space.

Reconstruction-based navigation

Most navigation approaches consider a partial 3D reconstruction of the environment, leading to SLAM-like techniques (Simultaneous Localization And Mapping). Such solutions are attractive, since the navigation task will be achieved using a classical pose-based control of the robot in the metric space. Within this context, during a learning step the environment is reconstructed using bundle adjustment approaches [Royer 2007] or Kalman/particle filters based approaches [Clemente 2007]. Despite the complexity of the underlying problem, SLAM has proved to be a viable solution to create accurate maps of the environment [Clemente 2007, Se 2002] even in large ones [Frese 2006]. In this context, the control of the robot during the navigation task is a well known problem and the main difficulties here are the complexity of the initial

Frame 11 Photometric visual servoing [Collewet 2008b].



The photometric visual servoing approach can be considered as a feature-based approach. The difference with the classical approaches comes from the fact that the features are no longer defined using geometric information but directly using the pixel intensities of the images. The desired features are the intensities of the reference image I^* for all the points $\mathbf{x}$ in the region of interest $\mathcal{W}$ (usually all the image) and the current features are the intensities of the current image $I(\mathbf{r})$ at the same locations:

$$\begin{array}{rcl} \mathbf{s} & = & (I(\mathbf{x}_0,\mathbf{r}),\ldots,I(\mathbf{x}_N,\mathbf{r})) \\ \mathbf{s}^* & = & (I^*(\mathbf{x}_0),\ldots,I^*(\mathbf{x}_N)) \end{array} \qquad \text{with:} \qquad \mathcal{W} = \{\mathbf{x}_0,\ldots,\mathbf{x}_N\}$$

The goal remains to minimize the error between the current and desired features, *i.e.* to minimize the difference between the intensities of the current and desired images:

$$\mathbf{e} = \mathbf{s}(\mathbf{r}) - \mathbf{s}^*$$

To solve this problem, the velocity is computed as in the classical feature-based approach:

$$\mathbf{v} = -\lambda \mathbf{L}_{\mathbf{s}}^{+}\mathbf{e}$$

where $\mathbf{L}_{\mathbf{s}}$ is the interaction matrix that links the variation of the feature $\mathbf{s}$ with the camera velocity $\mathbf{v}$. Since $\mathbf{s}$ is now obtained with a concatenation of the pixel intensities, its interaction matrix is computed with a concatenation of the interaction matrix of the intensities:

$$\mathbf{L}_{\mathbf{s}} = \begin{bmatrix} \mathbf{L}_{I(\mathbf{x}_0)} \\ \vdots \\ \mathbf{L}_{I(\mathbf{x}_N)} \end{bmatrix}$$

Using the temporal luminance constancy hypothesis [Horn 1981], the interaction matrix related to a pixel intensity is given by [Collewet 2008b]:

$$\mathbf{L}_{I(\mathbf{x})} = \nabla I \; \mathbf{L}_{\mathbf{x}}$$

where $\nabla I = (\partial I(\mathbf{x})/\partial x, \partial I(\mathbf{x})/\partial y)$ is the gradient of the image at the point $\mathbf{x}$ and $\mathbf{L}_{\mathbf{x}}$ is the interaction matrix related to the point $\mathbf{x}$. Using this approach in nominal conditions shows very good results. Since there is no feature extraction and that the whole information of the image is used, the final position of the robot is very accurate. Problems come from its robustness with respect to illumination variations.

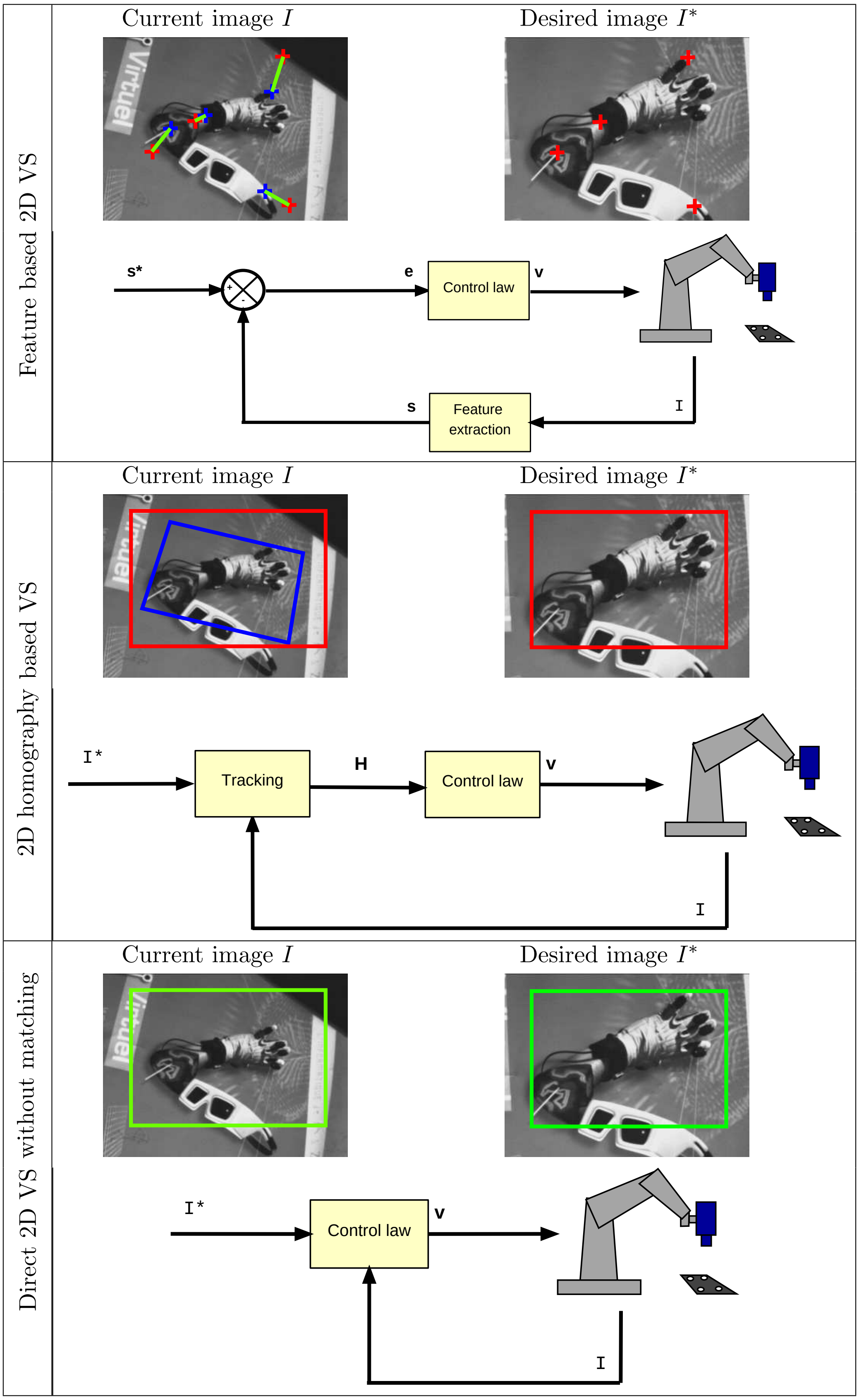


Figure 6.1: From the feature-based 2D visual servoing approach to the direct approaches.

reconstruction step and the matching of visual features observed during the learning step with current observations. With a monocular camera as unique sensor, these are mainly complex computer vision issues.

Model-free based navigation

Another class of techniques relies on the definition of a visual path: the appearance-based approaches [Burschka 2001, Remazeilles 2007, Segvic 2009, Courbon 2009, Chen 2009]. The trajectory is no longer described in the metric space but as a set of reference images. The applications are therefore more limited and most of them concern path following applications. A 2D visual servoing step allows the robot to navigate from its current position to the next key images. When the robot gets close to this image, a new key image is selected. In this context, the environment can be modeled by a graph whose nodes are the key images. A visual path in the environment is nothing but a path in the graph [Remazeilles 2007]. Working directly in the sensor space, such approaches do not require prior 3D reconstruction step. In some cases, partial reconstruction has to be considered. In [Segvic 2009, Diosi 2007] a part of the epipolar geometry that links the current and key images is considered in order to predict the location of currently not visible features and ensure a robust tracking. In [Courbon 2009], an homography estimation between the current image and the reference images allows to precisely localize the robot. In any case, the learning step of these appearance-based approaches is far less complex since it does not require any prior 3D reconstruction.

Nevertheless, at navigation level, for both pose-based or image-based visual navigation, features have to be extracted or tracked in the image stream and matched with either the 3D database or key images to design the control law. This robust extraction and real-time spatio-temporal tracking and matching of the visual cues [Marchand 2005a] are non trivial tasks. Some more direct navigation tasks exist that use template-based approaches [Matsumoto 1996], but a tracking step is still required at each iteration of the process.

6.3 Conclusion

In this chapter, we gave a general overview of the existing visual servoing approaches. The approaches based on a low level tracking or matching process have the advantages of their robustness and convergence domain. But these advantages are entirely depending on the robustness of the tracker. To avoid this tracking (and sometimes matching) step, new direct approaches have been proposed. Some of this direct visual servoing approaches demonstrate to be well suited to reach an accurate position. Nevertheless, the existing approaches exhibit major issues of robustness with respect to illumination variations.

Chapter 7

Mutual information-based visual servoing

In this chapter, we propose a new way to achieve visual servoing task. Rather than minimizing the error between the position of two set of geometric features, we consider the direct visual servoing approach. The positioning task is defined as computing the camera velocity $\mathbf{v}$ that will maximize the similarity between the current image $I(\mathbf{r})$ and the desired image I^*. Since the appearance of the image $I(\mathbf{r})$ can change depending on the illumination variations or occlusions, the similarity measure between the two images has to be robust for the visual servoing task to be robust. To satisfy this condition, we propose to find the desired pose by maximizing the mutual information shared by the current image and a reference image:

$$\widehat{\mathbf{r}} \;=\; \arg\max_{\mathbf{r}} \mathrm{MI}\left(I^*, I(\mathbf{r})\right). \tag{7.1}$$

This leads to a new information theoretic approach to visual servoing. Since mutual information is robust to illumination variations, occlusions and non-linear changes in the intensities of the image, its use in the visual servoing problem allies both robustness and efficiency. Furthermore as for [Collewet 2008b], it does not require any tracking or matching process. Experiments on a 6 dof robot validate the advantages of the proposed visual servoing task.

7.1 MI based control law

In this section, we present the control law obtained using the mutual information function. We show that both the MI-based visual servoing and MI-based pose computation are sensibly equivalent. Therefore, the computation of the velocity is then nothing but the increment computation in the pose computation task.

We recall that the goal of the pose computation problem was to find the object (or camera) pose $\mathbf{r}$ in an input image I^*. This problem can be formulated as finding the camera pose $\mathbf{r}$ that projects a model of the object $\mathcal{M}$ into an image I that is the most similar to the image I^*:

$$\widehat{\mathbf{r}} \;=\; \arg\max_{\mathbf{r}} \mathrm{MI}(I^*, I_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r})) \tag{7.2}$$

where $\boldsymbol{\gamma}$ are the intrinsic camera parameters and I_{ξ} is the image resulting from the projection of the model into the image plane of the virtual camera. This equation can be solved using a non-linear optimization. We consider that $\mathbf{r}$ represents the position of a virtual camera. The update of $\mathbf{r}$ can, therefore, be performed using the velocity $\mathbf{v}$ of the virtual camera.

This pose estimation process can be seen as a simulation of a visual servoing task or virtual visual servoing task [Sundareswaran 1998, Marchand 2002]. $\mathcal{M}$ models the environment where the camera is moving and the projection into $I_{\boldsymbol{\gamma}}(\mathcal{M}, \mathbf{r})$ simply corresponds to the projection that takes place in the real camera $I(\mathbf{r})$. Finally, the update of the camera pose that was performed using the exponential map of the velocity simply corresponds to the velocity control of the robot. Therefore, pose computation and visual servoing are two dual problems.

Since all the inputs of the MI optimization are equivalent, the velocity necessary to maximize the MI in the visual servoing problem can be obtained using almost the same approach as in the pose estimation problem. A steepest gradient descent approach [Viola 1995] is inefficient when the parameters are strongly correlated: the optimization is then subject to oscillation, and the resulting velocity of the robot would be unadapted. We have also seen that a Newton's method using a the approximation of the Hessian matrix [Thévenaz 2000, Dowson 2006, Dowson 2008, Panin 2008] is satisfying only coupled to a line-search algorithm. But a back-tracking approach is impractical in the visual servoing problem: we could hardly move continually the robot forward and backward to reach an accurate pose. On the contrary, the optimization approach that we proposed in the chapter 4 is perfectly suited to compute the velocity of the robot. This method (called HC method) is a Newton's like approach that uses the Hessian of the cost function estimated at its optimum to compute the velocity as:

$$\mathbf{v} = -\alpha {\mathbf{H}^*_{\mathrm{MI}}}^{-1} \mathbf{L}_{\mathrm{MI}}^{\top} \tag{7.3}$$

where $\alpha \in [0, 1]$ is the stepsize that limit the convergence speed. $\mathbf{L}_{\mathrm{MI}}$ and $\mathbf{H}_{\mathrm{MI}}$ are respectively the interaction matrix of MI and its Hessian matrix. We recall that $\mathbf{H}^*_{\mathrm{MI}}$ is the estimation of the Hessian matrix computed at the optimum of the mutual information function. Using the developed formulation of MI, we recall that its derivatives yield:

$$\begin{aligned}
\mathbf{L}_{\mathrm{MI}} &= \sum_{i,j} \mathbf{L}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) && (7.4)\\
\mathbf{H}_{\mathrm{MI}} &= \sum_{i,j} \mathbf{L}_{p_{II^*}}^{\top} \mathbf{L}_{p_{II^*}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_{I^*}}\right) + \mathbf{H}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) && (7.5)
\end{aligned}$$

We recall that the joint probability p_{II^*} between the key image I^* and current image I and its variations are given by:

$$\begin{aligned}
p_{II^*}(i, j, \mathbf{r}) &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \phi\left(i - \bar{I}(\mathbf{x}, \mathbf{r})\right) \phi\left(j - \overline{I^*}(\mathbf{x})\right)\\
\mathbf{L}_{p_{II^*}(i,j,\mathbf{r})} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \mathbf{L}_{\phi\left(i - \bar{I}(\mathbf{x}, \mathbf{r})\right)} \phi\left(j - \overline{I^*}(\mathbf{x})\right)\\
\mathbf{H}_{p_{II^*}(i,j,\mathbf{r})} &= \frac{1}{N_{\mathbf{x}}} \sum_{\mathbf{x}} \mathbf{H}_{\phi\left(i - \bar{I}(\mathbf{x}, \mathbf{r})\right)} \phi\left(j - \overline{I^*}(\mathbf{x})\right)
\end{aligned}$$

where $N_{\mathbf{x}}$ is the number of pixels considered in the images I and I^*. ϕ is a B-spline function differentiable twice (see the MI definition in section 2.3.2.1). The interaction matrix and Hessian of ϕ are given by:

$$\begin{aligned}
\mathbf{L}_{\phi\left(i - \bar{I}(\mathbf{x}, \mathbf{r})\right)} &= -\frac{\partial \phi}{\partial i} \mathbf{L}_{\bar{I}} && (7.6)\\
\mathbf{H}_{\phi\left(i - \bar{I}(\mathbf{x}, \mathbf{r})\right)} &= \frac{\partial^2 \phi}{\partial i^2} \mathbf{L}_{\bar{I}}^{\top} \mathbf{L}_{\bar{I}} - \frac{\partial \phi}{\partial i} \mathbf{H}_{\bar{I}} && (7.7)
\end{aligned}$$

and:

$$\begin{aligned}
\mathbf{L}_{\bar{I}} &= \nabla \bar{I}\, \mathbf{L}_{\mathbf{x}} && (7.8)\\
\mathbf{H}_{\bar{I}} &= \mathbf{L}_{\mathbf{x}}{}^{\top}\, \nabla^2 \bar{I}\, \mathbf{L}_{\mathbf{x}} + \nabla \bar{I} \mathbf{H}_{\mathbf{x}} && (7.9)
\end{aligned}$$

where the two expressions are depending on the interaction matrix $\mathbf{L}_{\mathbf{x}}$ and Hessian matrix $\mathbf{H}_{\mathbf{x}}$ of each pixels. To compute the analytical formulation of these matrices in a 6 dof task, the depth of each pixel at the current position is required (see equation (5.16)). In the real visual servoing problem, this depth is unknown. In our approach, we assume that the depth of the pixels can be approximated by a constant.

7.2 Efficient control law

As we have seen previously the optimization of the mutual information is based on the Hessian matrix at convergence that is kept constant. Using a classical Newton's method will give a velocity that is directly linked to the gradient of mutual information. Since mutual information cost function is flat at convergence and far from the convergence, the velocity will be small at convergence what is required, but it may also be small far from the convergence.

The usual goal of a visual servoing task is to have an error that decreases exponentially. This is easily implemented using the classical geometric feature-based visual servoing methods. In a direct visual servoing task, there is no measure of the distance to reach the optimum. We seek the maximum of the mutual information, but there is no available estimation of the MI value after the convergence. In the geometric approach, we know that the final error must be null. This section shows how the proposed visual servoing approach can be improved by increasing the velocity far from the convergence.

7.2.1 Sequencing the task

To improve the behavior of the visual servoing task far from the convergence, we propose to separate the convergence process in two steps. When the robot is far from the convergence (the first step of the task), a post treatment is applied to the resulting velocity $\mathbf{v}$ of the optimization approach. The final velocity $\mathbf{v}_{post}$ is constrained to have a constant norm. The modification is then simply applied as a gain factor to the velocity $\mathbf{v}$:

$$\mathbf{v}_{post} = \mathbf{v}\frac{\|\mathbf{v}_c\|}{\|\mathbf{v}\|} \tag{7.10}$$

When the robot is close to the convergence (the second step), this post treatment stops and the velocity sent to the robot is simply the one computed with the control law that is properly decreasing to reach a null velocity at convergence as well as the MI gradient's norm is decreasing.

7.2.2 Estimation of the remaining positioning error

One problem to divide the visual servoing task in two steps is that we cannot estimate directly the remaining error between the current and the desired pose. To determine in which state the robot is, the knowledge on the evolution of the Hessian matrix is essential. As it has been seen in previous sections mutual information is quasi concave at convergence. It means that the concavity of mutual information at the current pose of the camera reflects somehow the distance with the desired pose.

A study of the current Hessian matrix could seem to be a solution of the problem. However, since the mutual information function is highly depending on the scene, it is not possible to simply consider the Hessian matrix.

On the other hand, we know that the current Hessian matrix is adapted to the optimization problem as soon as the camera pose is in the concave domain of MI, that is near the convergence. In this case, the velocities $\mathbf{v}_c$ and $\mathbf{v}_d$ obtained respectively using the current Hessian matrix and the Hessian matrix at convergence are both adapted and similar when the camera reaches the concave domain. To estimate if the camera is next to the convergence, a simple similarity measure is computed between the two velocities $\mathbf{v}_c$ and $\mathbf{v}_d$ as follows:

$$\epsilon(\mathbf{v}_c, \mathbf{v}_d) = \frac{1}{\|\mathbf{v}_c\|^2 \|\mathbf{v}_d\|^2}\mathbf{v}_c^\top \mathbf{v}_d \tag{7.11}$$

that has a score bound to $[-1,1]$. If $\epsilon = -1$ then $\mathbf{v}_c$ is pointing to the opposite direction of $\mathbf{v}_d$, and if $\epsilon = 1$ then the two vectors are equals up to a factor. First, the normalized velocity is sent to the robot. And, as soon as the two velocity vectors are similar that is ϵ above a given threshold (0.5 in our experiments), the visual servoing task is considered to be near the convergence, and the normalization of the velocity takes end.

7.3 Experimental results

To validate the proposed approach several experiments have been performed using a camera mounted on a 6 dof gantry robot. The computation time remains low, since the control law is computed at video rate. A velocity is computed in about 10ms for a 320×240 input image using a 2.4 GHz laptop and sent to the robot at 50Hz, the video-rate. In the following experiments, we will use, in several occasions, the difference between the desired and current image $I^* - I$ to illustrate first the precision of the positioning task in nominal conditions, then to illustrate the perturbations. Nevertheless, we recall that we do absolutely not rely on this error to perform the visual servoing task.

7.3.1 Positioning tasks

A first set of experiments is realized to validate the robustness of the proposed approach using classical images in nominal conditions. We also consider occlusions and illumination variations to validate the robustness of mutual information with respect to perturbations. The camera is first moved to the desired pose $\mathbf{r}^*$ where the reference image is acquired. The camera pose is then set to an initial pose $\mathbf{r}$ to satisfy that the reference image is partially represented in the current image.

During the task, the positioning error $\mathbf{\Delta r}$, that is the transformation between $\mathbf{r}$ and $\mathbf{r}^*$, is computed to evaluate the behavior of the task and assess its precision. For a clarity purpose, the error will be referred in the text as $\mathbf{\Delta r_{trans}}$, the norm of the translation of the camera between the current pose and the desired pose in meters, and as $\mathbf{\Delta r_{rot}}$, the norm of rotation error in degrees.

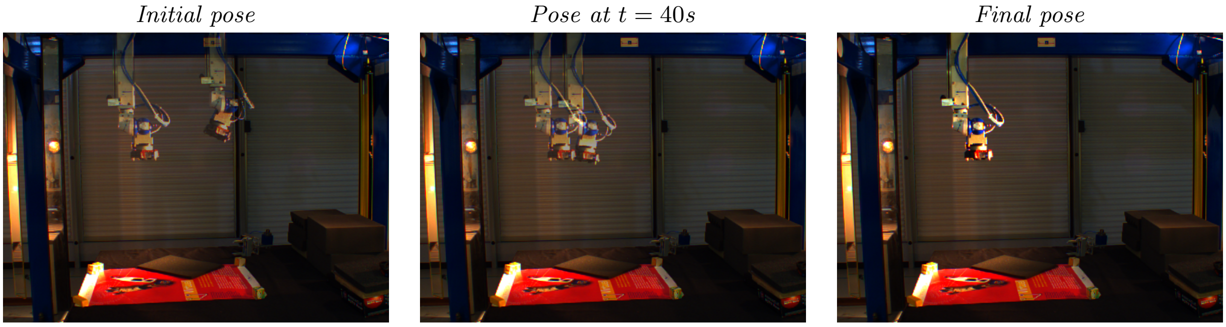
7.3.1.1 Nominal conditions

In this experiment, the illumination conditions remain constant during the realization of the positioning task. Figure 7.2 shows the desired and initial images acquired by the camera, the initial and final error image, and the positioning error using the Cartesian coordinates for the translation part of $\mathbf{\Delta r}$ and the error on the rotational part.

From an initial positioning error of $\mathbf{\Delta r_{trans}} = 0.37\ m$ and $\mathbf{\Delta r_{rot}} = 25.0^o$, the camera is smoothly converging to the desired pose to reach a final positioning error of $\mathbf{\Delta r_{trans}} = 3.10^{-4}$ m and $\mathbf{\Delta r_{rot}} = 0.06^o$. Considering that the distance from the camera to the scene is about one meter, the proposed visual servoing task proves to be very accurate compared to the accuracy that feature-based approaches would provide.

Comparison between the original and sequenced tasks

To show the advantages of the sequencing method proposed in section 7.2, we repeat exactly the same experiment and perform the visual servoing task without the sequencing method. Figure 7.3 shows the results of both the original and improved visual servoing tasks. Far from the optimum (far from the desired pose), the gradient of MI has small values. Since the HC method provides a velocity that is directly linked to the gradient of MI, the resulting velocity

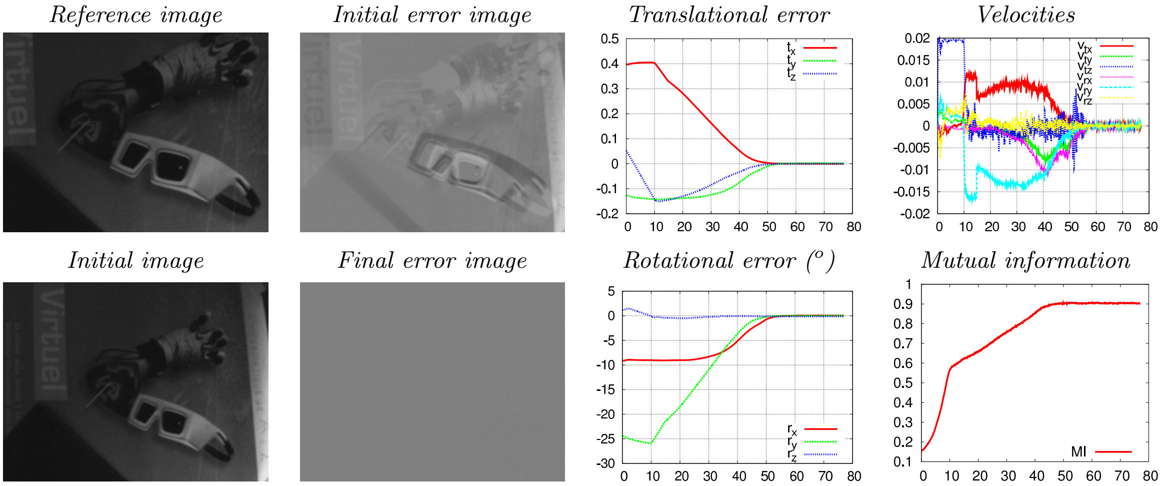


Initial pose

Pose at $t = 40s$

Final pose

Figure 7.1: External view of one MI-based visual servoing task. The current and desired poses of the robot are overlaid. On the final pose, there is no difference between the two poses.



Reference image

Initial error image

Translational error

Velocities

Initial image

Final error image

Rotational error (^{o})

Mutual information

Figure 7.2: Visual servoing using mutual information. The positioning error, velocities (in $m.s^{-1}$ and $rad.s^{-1}$) and mutual information are represented with respect to the time in seconds.

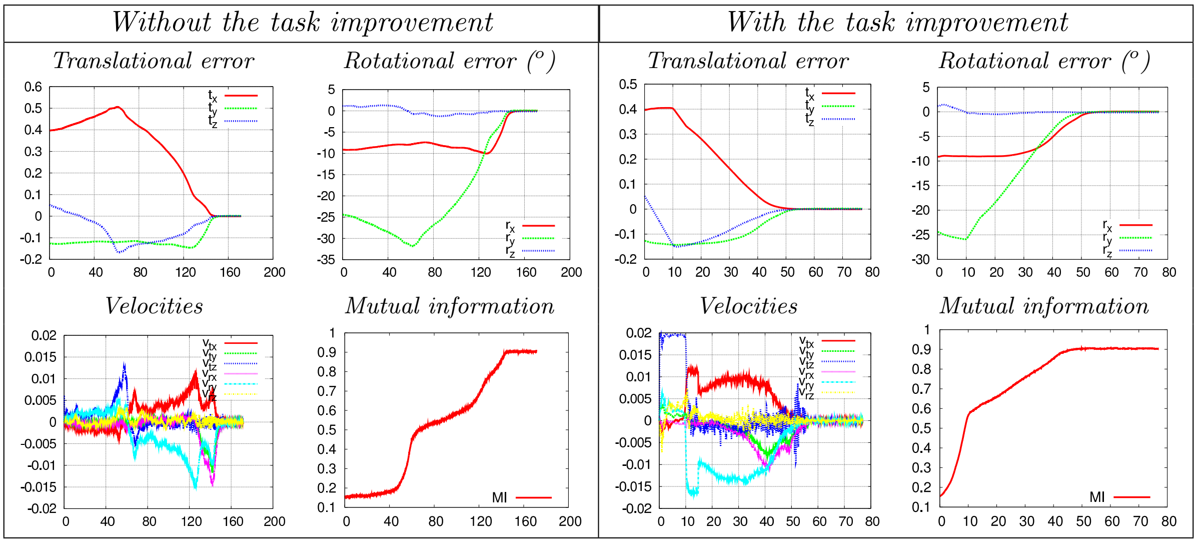


Figure 7.3: Comparison between the original and improved visual servoing tasks: far from the desired pose, the original task converges slowly while the improved task is fast.

far from the desired pose is small. The convergence rate of the original visual servoing task is therefore very slow. The method proposed in the previous section allows the improvement of the convergence rate. When the camera is far from its desired pose, the velocity is normalized using the equation (7.10) and the desired pose is rapidly reached (in about 60 seconds while it took 140 seconds without the method).

7.3.1.2 Robustness with respect to calibration errors

Since the MI gradient matrix depends on the interaction matrix related to a point, it also depends on the camera calibration. In most of the visual servoing applications, the camera and its calibration are known. However, this calibration can be subject to estimation error. In this section, we show the effects of calibration errors on the previous positioning task.

We plot the convergence rates of the positioning tasks using an accurate calibration of the camera and using coarse calibrations in Figure 7.4. We note (p_x, p_y, u_0, v_0) the intrinsic camera parameters where (p_x, p_y) are related to the pixel size and (u_0, v_0) represent the position of the central point in the image. $(\widehat{p_x}, \widehat{p_y}, \widehat{u_0}, \widehat{v_0})$ are the accurate intrinsic camera parameters that were used in the previous experiment. The visual servoing task keeps converging even with a percentage error on the intrinsic parameters wider than 50 percents. The accuracy of the final positioning error it does not change with the p_x and p_y parameters but is affected by the u_0 and v_0 values. For instance, the experiment that uses $(u_0, v_0) = (0.1\widehat{u_0}, 0.1\widehat{v_0})$ reaches a final positioning error of $\mathbf{\Delta r_{trans}} = 9.10^{-4}\ m$ and $\mathbf{\Delta r_{rot}} = 0.15^o$, while, we recall that the one using $(u_0, v_0) = (\widehat{u_0}, \widehat{v_0})$ reaches an error of $\mathbf{\Delta r_{trans}} = 3.10^{-4}\ m$ and $\mathbf{\Delta r_{rot}} = 0.06^o$. Finally, if this error is too large (last experiment of the figure with 20 percents of error on p_x and p_y, and 150 percents on u_0 and v_0), the task diverges. However, the calibration errors are not supposed to be this large.

7.3.1.3 Robustness with respect to occlusions and illumination variations

In these experiments the appearance of the scene is modified during the positioning task using occlusions or illumination variations. As we can see in Figure 7.5, despite the large initial positioning error and the change in appearance, the visual servoing task converges and we can see

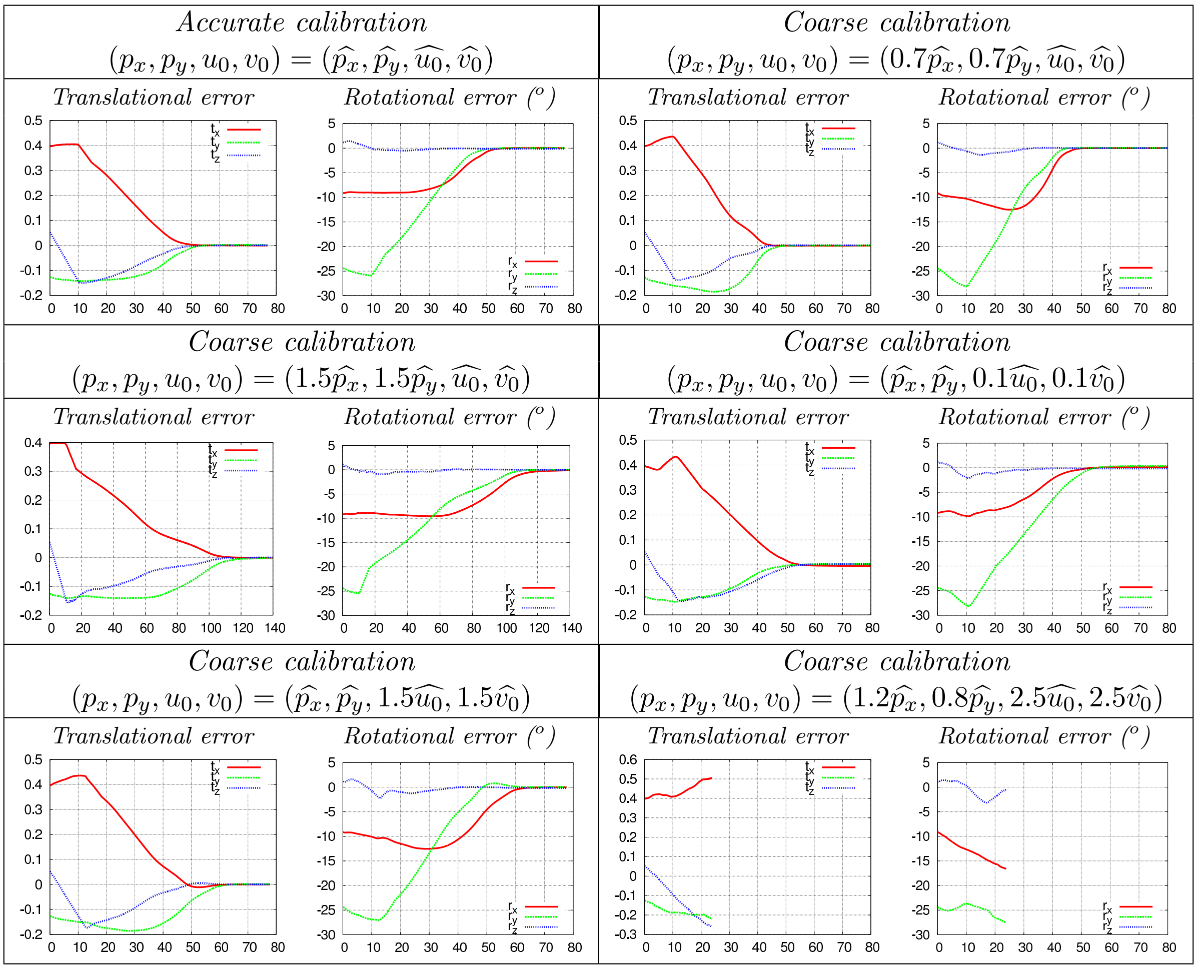


Figure 7.4: Effect of the camera calibration on the visual servoing task. The MI-based visual servoing task is robust to camera calibration errors. Nevertheless, if the error is too large, the task diverges (bottom right experiment).

that the positioning error decreases with respect to the time to become very small. As expected, the proposed MI-based visual servoing scheme is naturally robust to large perturbations and remains very accurate. Indeed, in the illumination variation experiment, the initial positioning error is $\mathbf{\Delta r_{trans}} = 0.17\ m$ and $\mathbf{\Delta r_{rot}} = 16.3^o$ and the visual servoing task reaches a final positioning error of $\mathbf{\Delta r_{trans}} = 9.10^{-4}\ m$ and $\mathbf{\Delta r_{rot}} = 0.08^o$. In the occlusion experiment the initial error is $\mathbf{\Delta r_{trans}} = 0.35\ m$ and $\mathbf{\Delta r_{rot}} = 25.2^o$, after the visual servoing task it is $\mathbf{\Delta r_{trans}} = 1.10^{-3}\ m$ and $\mathbf{\Delta r_{rot}} = 0.1^o$.

7.3.1.4 Grasping, robustness with respect to depth approximation

As previously discussed, the proposed approach is very accurate. This section shows that this accuracy can be useful to position a gripper with respect to an object and grasp it. Since the object is non planar it also validates the robustness of the proposed approach with respect to non-constant depth (see the approximation of the scene's depth in section 5.2.2).

Since the appearances of the object from the initial pose to the final pose are very different, several sequential positioning tasks are required to reach the final grasping pose. In this experiment five following visual servoing tasks have been performed using five different reference images. Figure 7.6 shows the scene with the initial and final poses and 3 reference images used to reach the final pose. The experiment converges and the final positioning error is $\mathbf{\Delta r_{trans}} = 4.10^{-4}\ m$ and $\mathbf{\Delta r_{rot}} = 4.10^{-2\ o}$ which is sufficient to position the gripper and grab the object. Note also the lack of texture in the scene. Even with a small amount of information, the method remains accurate enough to positionate the gripper.

7.3.2 Multimodal image-based navigation

In the definition of the mutual information, we stated that a linear dependency between the intensities of the desired and current images is not required. Mutual information is thus able to align two images even if they are acquired from different modalities as soon as they share enough information. In this experiment, we show the robustness of MI as a multimodal similarity measure by servoing the camera on an aerial scene using a map as a reference image.

Here, we consider a different task: following a visual path. A more complete analysis of such task will be presented in the next chapter. This experiment is similar to the grasping experiment described in the previous paragraph except that more images are considered to define the visual path. A second important difference is that the visual path has been learnt when observing the map (provided by *IGN (Institut géographique nationale) geoportail*). Once the visual path (a set of reference images) has been learnt, the map is replaced by a satellite image (at the same scale) and the camera is positioned near the camera pose corresponding to the image of the learned visual path (in order to be in the convergence domain of the task). Then the navigation task can be performed.

Several navigation tasks have been performed with success. Figure 7.7 illustrates one of these navigation tasks using only 3 dof: the rotation around the z axis and the translations along the x and y axis. We can see that the current and reference images are correctly aligned with the displacement of the camera despite their large differences of appearance (that would cause every feature-based or photometric-based techniques to fail). And thus, we can see on the projection of the trajectories on the xOy plane that the resulting camera trajectory of the navigation task properly replays the learned trajectory.

Figure 7.8 shows the images and trajectory in a visual servoing task using 6 degrees of freedom. To show the accuracy of the method, we define a learnt trajectory that implies large motions of the camera for small transformations in the image. The defined trajectory is about

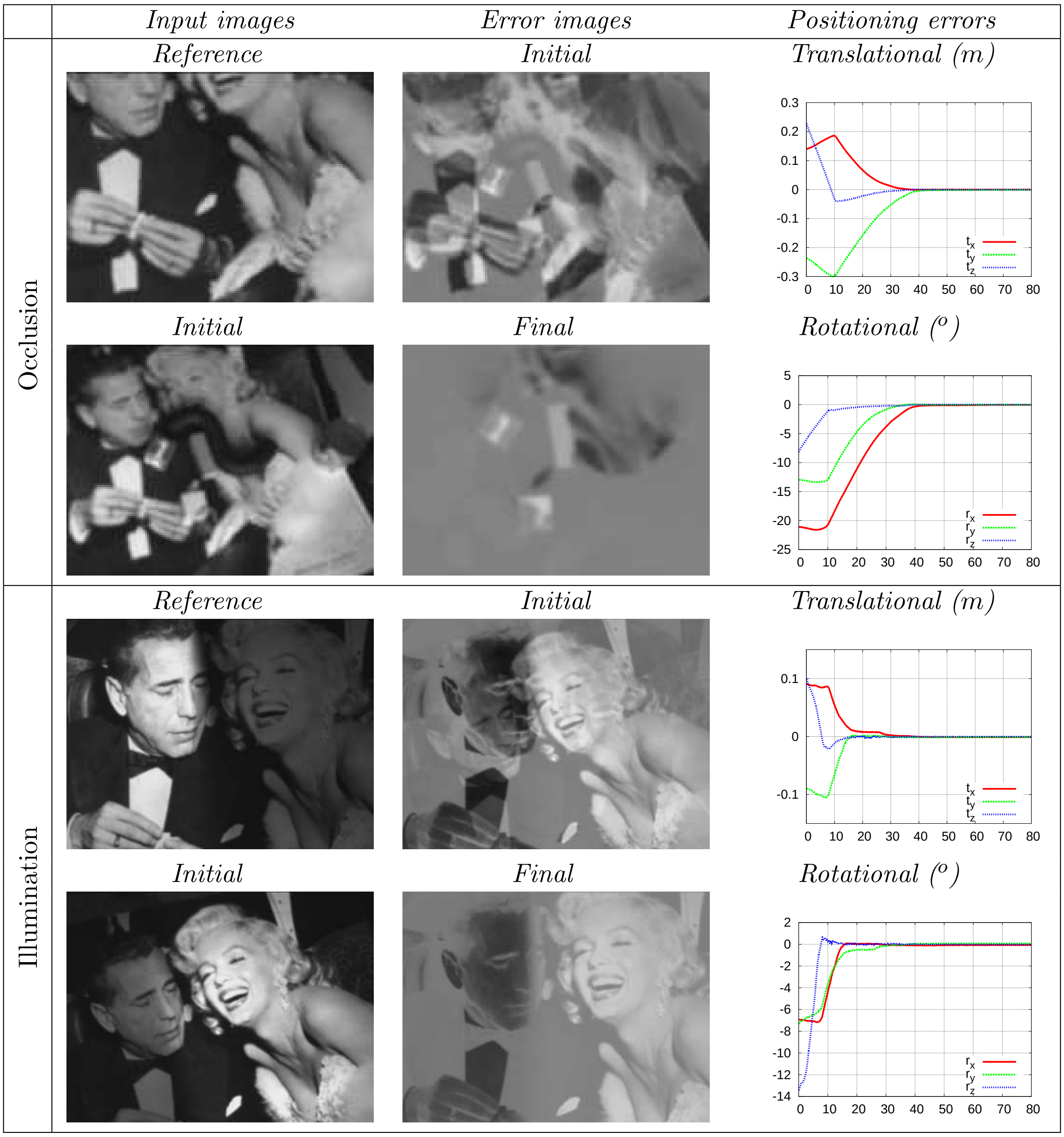
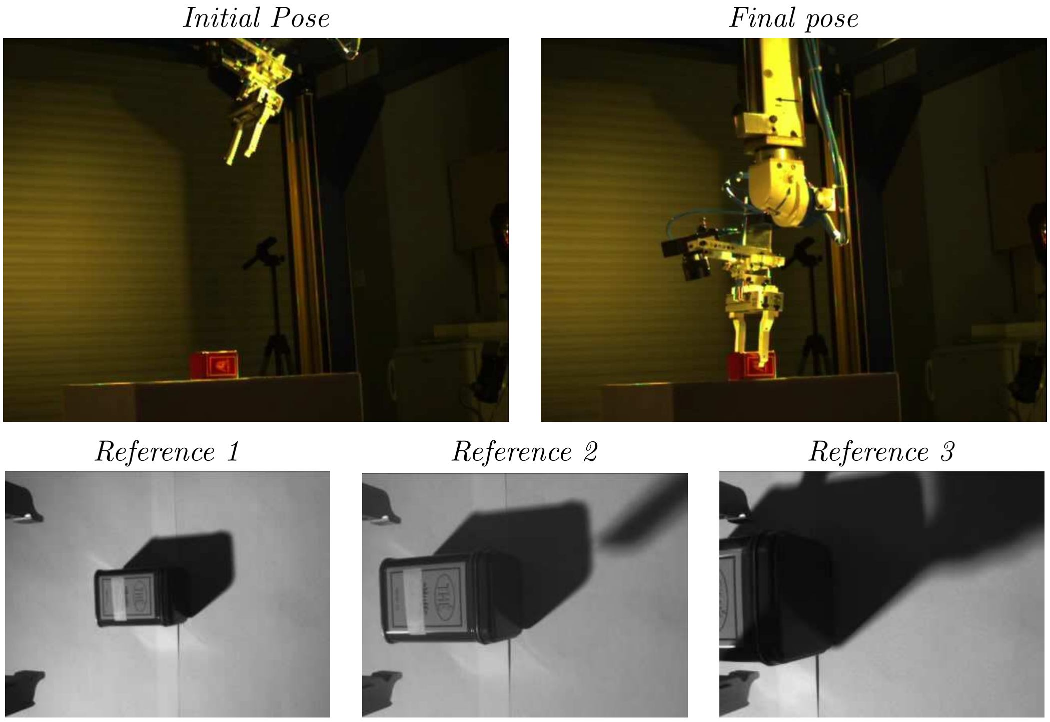


Figure 7.5: Visual servoing using mutual information. The robustness of the visual servoing task with respect to occlusions and illumination variations is verified by the evolution of positioning error over time (in seconds) and by appearance of the final error image.



Initial Pose

Final pose

Reference 1

Reference 2

Reference 3

Figure 7.6: Grasping using multiple visual servoing tasks.

60 centimeters long. Despite the strong correlation between the DOF of the robot, the visual servoing task converges and the resulting trajectory is similar to the desired trajectory.

7.4 Empirical convergence analysis

When considering IBVS with redundant features, only local stability can be considered and this is also the case for our MI-based visual servoing scheme [Chaumette 2006]. Nevertheless, it is always possible to evaluate the convergence area of this method from an empirical point of view. This section evaluates the performances of the proposed visual servoing approach on a large set of simulated experiments (considering simulation allows to perform exhaustive tests with hundreds of positioning tasks).

7.4.1 Convergence domain and performance metrics

A set of initial poses has been chosen to best evaluate the robustness of the task with respect to the six degrees of freedom of the robot in nominal conditions (no lighting variations are considered).

7.4.1.1 Convergence area

The initial poses are set so that the center of the camera is placed on a regular three-dimensional grid centered on the desired pose and its direction is defined so that initial and desired images overlap (this implies large variation around the r_x and r_y axes). In this case, each initial pose ensures that the current image shares some information with the desired image. We also consider several initial rotation errors around the camera z axis r_z, since these rotations are usually difficult to handle in visual servoing.

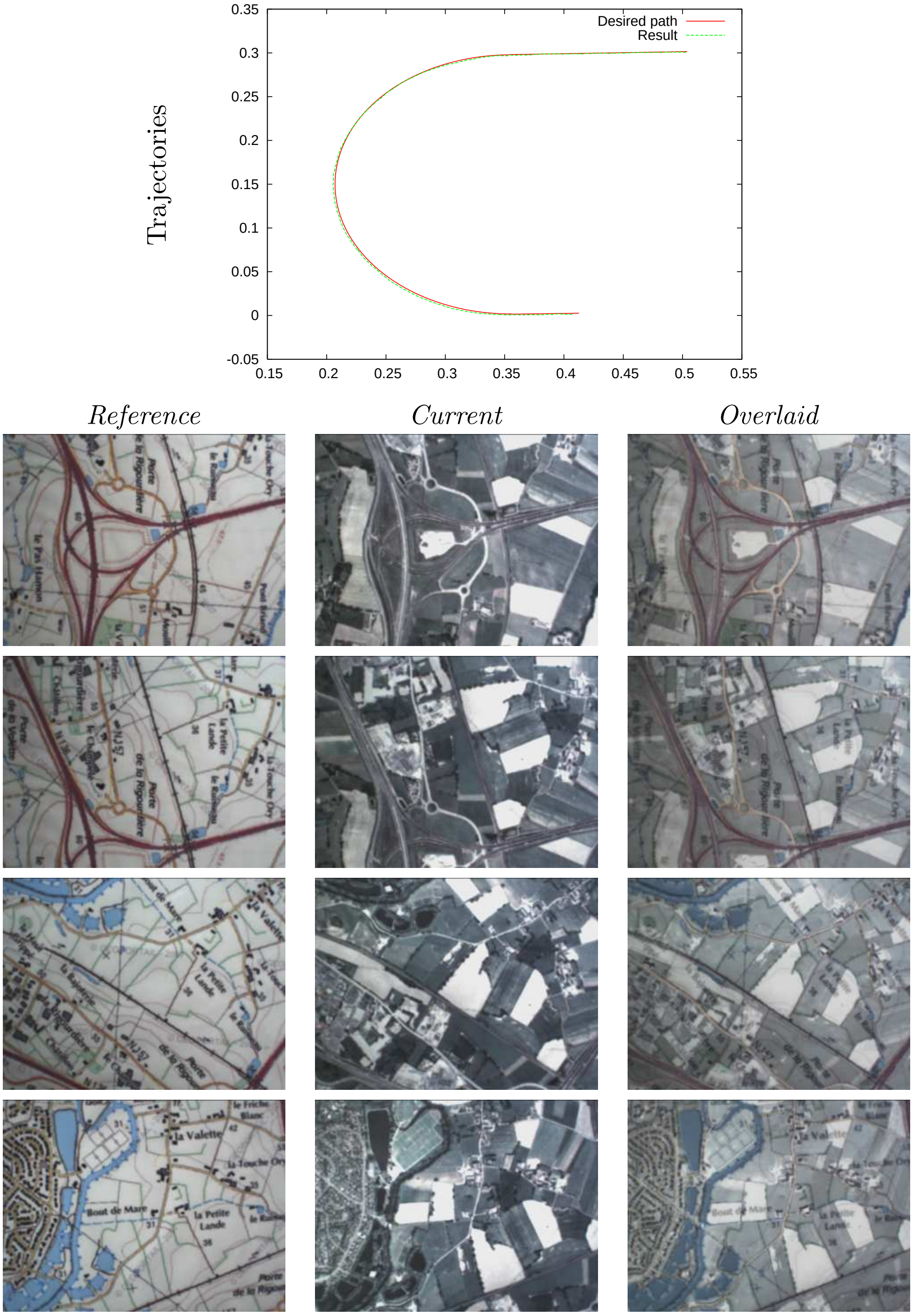


Figure 7.7: 3D Multi-modal MI-based visual servoing in a navigation task. An image path is learnt on the map scene and the visual servoing task performs the navigation on the satellite scene. The reference and resulting trajectories are very close. The correct alignment is also visible in the image with the reference and current images overlaid.

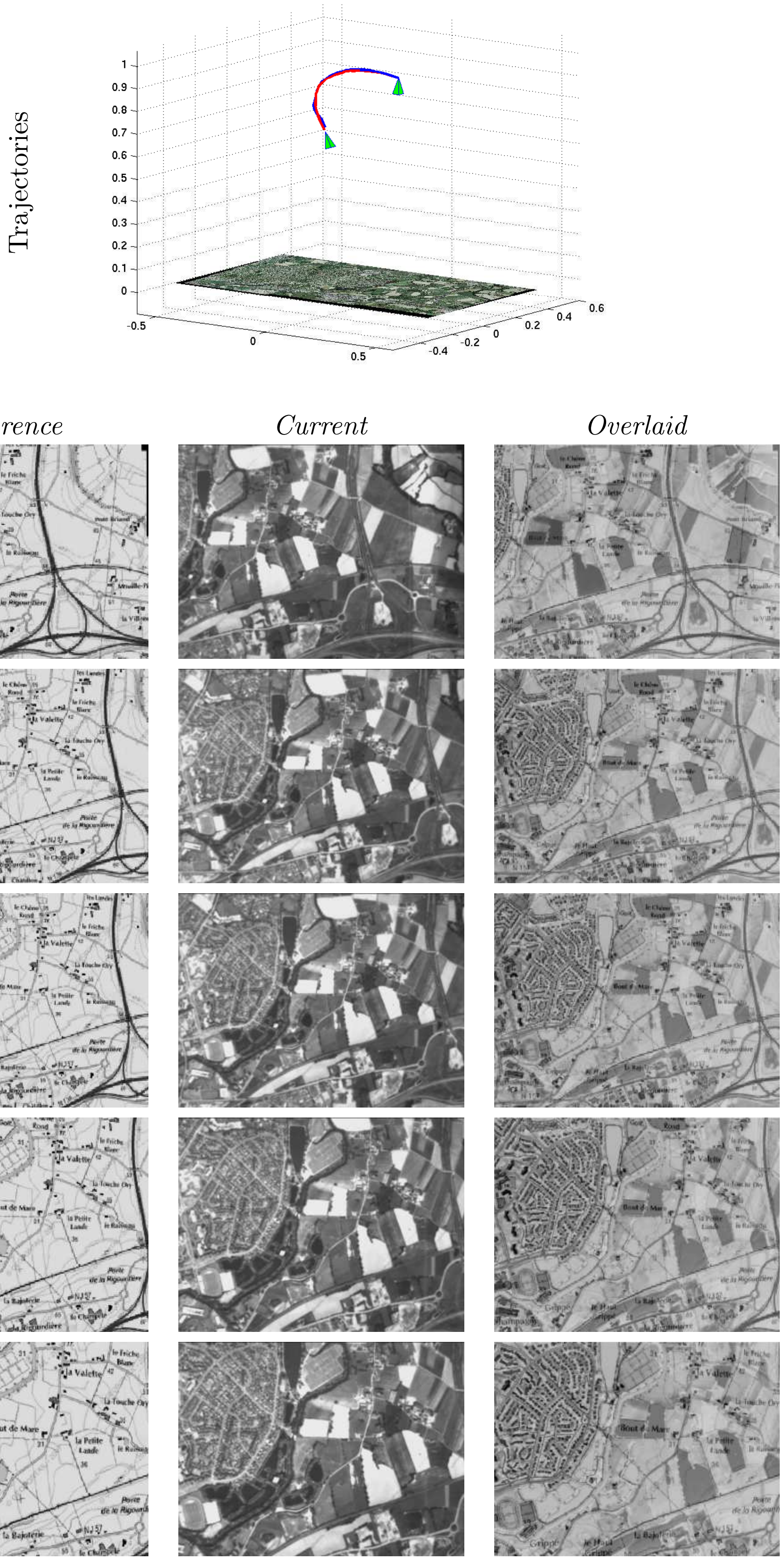


Figure 7.8: 6D Multi-modal MI-based visual servoing in a navigation task. The reference and resulting trajectories remain similar.

Figure 7.9 shows the convergence results obtained on a 3D grid of $21 \times 21 \times 21$ initial poses varying from -2 to 2 meters on the translations along the x and y axes (producing rotations from -60 to 60^o around the same axis) and from -1 to 1 meter in translation along the camera z axis with an initial rotation r_z of 0^o and 20^o. The convergence domain is large. We can notice on the slices along the three plans defining the grid that the convergence domain is convex and that its hull has a spherical shape approximately centered on the desired pose with a radius of about 1 meter. The size of this sphere decreases while the initial rotation error around z increases.

7.4.1.2 Camera trajectory

Whereas the previous experiments described the convergence area, it is also of interest to analyze the camera trajectory during the positioning task. A set of four quantitative metrics [Gans 2003] has been measured on this set of experiments to perform this evaluation. The first is the convergence ratio that gives the proportion of converging visual servoing tasks on the whole set of experiments. The second and third are the average distance covered by the center of the camera and the average integral between the camera center and the geodesic (considering only successful experiments). Both measures are illustrated in Figure 7.10 on the resulting trajectory of the camera in one of the experiments with an initial positioning error of $\mathbf{\Delta r_{trans}} = 1\ m$ and $\mathbf{\Delta r_{rot}} = 49.0^o$. Finally the last metric is the final positioning error, that is the transformation between the final pose of the camera and its desired pose.

The results that have been obtained are represented in Table 7.1 with respect to the initial rotational error around the focal axis r_{z_0}. Despite the small convergence frequencies that are reported in the table, the proposed visual servoing task has a large convergence domain. The frequency is in fact small due to the large domain of evaluation. The integral between the geodesic and the camera trajectory is relatively small compared to the distance covered by the camera, we can, therefore, consider that the camera trajectory is close from the geodesic and is good. As expected, the greater the initial rotational error, the more the convergence rate decreases. Indeed, if this rotation is too large then the initial amount of shared information between the images is too small and the control law reaches a local optimum. The final positioning error has not been reported in the table since it is remaining constant with a final translation error of 3.10^{-5}mm and 0.003^o in every converging experiment.

r_{z_0}	0^o	10^o	20^o	30^o	40^o	50^o
Convergence (%)	26.7	27.6	25.2	21.0	12.4	1.8
Distance (m)	1.06	1.08	1.11	1.16	1.27	1.65
Integral (m^2)	0.11	0.12	0.12	0.13	0.14	0.25

Table 7.1: Performance of the proposed visual servoing task on the set of initial poses represented in Figure 7.9.

7.4.2 Comparison with existing solutions

Let us now compare our approach with other visual servoing schemes in one typical visual servoing task with and without illumination variations. Two visual servoing approaches, adapted to the current problem, have been considered. The first one is the photometric based visual servoing [Collewet 2008b]. The second one is a classical feature based visual servoing where the features are points extracted and matched using the SIFT algorithm [Lowe 2004].

The obtained results have been summarized in Figure 7.11 where SSD refers to the photometric approach. Without illumination variations the proposed approach has a similar behavior as the photometric one. In term of trajectory the direct approaches (MI and photometric) are further from the geodesic than the SIFT approach. Considering the final positioning error, the direct approaches are more accurate than the feature-based approach. A good alternative is then to use the feature-based approach and switch the control law at convergence to finally use the MI based visual servoing approach to take advantage of both a trajectory near the geodesic and a very accurate final pose. This approach, that we call hybrid approach, has been implemented and gives indeed the most adapted behavior with both advantages of trajectory and accuracy.

To evaluate the performance of the approaches with respect to illumination variations, the following variation has been applied to the scene: from the acquisition of the desired image to the visual servoing tasks, the left part of the scene has been illuminated and the right part is put in the shadow. The consequence of such a modification in the intensities makes the SSD function to have a minimum at the wrong camera pose and thus the photometric approach diverges. The illumination variation has also a slight effect on the matching step of the SIFT approach, the visual servoing task is then converging but has with a larger final positioning error. As for MI based approach, the trajectory of the camera is slightly affected but the final positioning task remains very accurate.

7.5 Conclusion

In this chapter, we presented a new visual servoing approach based on mutual information. The algorithm uses directly the image that should be visible at the desired pose of the camera. Many experiments demonstrate the validity of our approach on a 6 dof robot. This approach shows many advantages:

- The new control law does not use any geometrical features, so that it also does not require any extraction, matching or tracking steps that are usually the bottleneck of classical approaches. Only the desired image has to be known to reach the desired camera pose.

- Since the mutual information is naturally robust to partial occlusions and illumination variations, the control law, that is nothing but the optimization of the mutual information, is also robust to the appearance variations of the scene.

- The whole content of the image is directly considered in the computation of the mutual information. Mutual information is very accurate. Since the control law is directly defined by the MI optimization, without any tracking and matching processes, the resulting positioning task is very accurate.

- As it is well known in the medical field, mutual information is also robust to multimodal alignment. Some new visual servoing application are therefore possible including for instance aerial drones navigation. Although our experiments were limited to map and aerial images, other modalities can be easily considered such as infrared images.

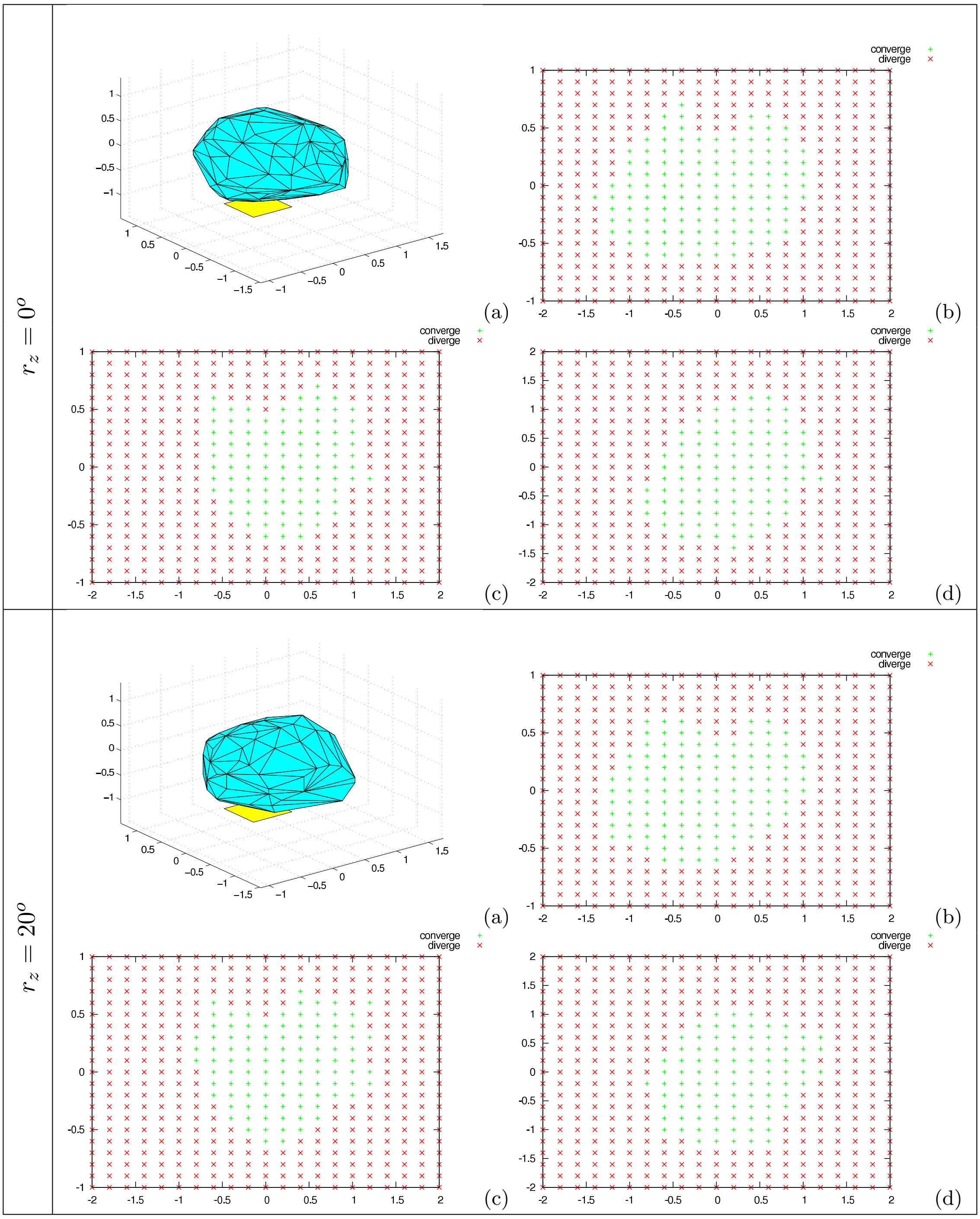


Figure 7.9: Domain of attraction in the 3D space with respect to the rotation around the focal axis. Axes are in meters. The blue volume in (a) represents the convex hull of the initial poses that converged. The yellow plan represents the target (a). Slices of the domain of attraction through the x(b), y(c) and z(d) plans are represented with the poses that converged in green and that diverged in red.

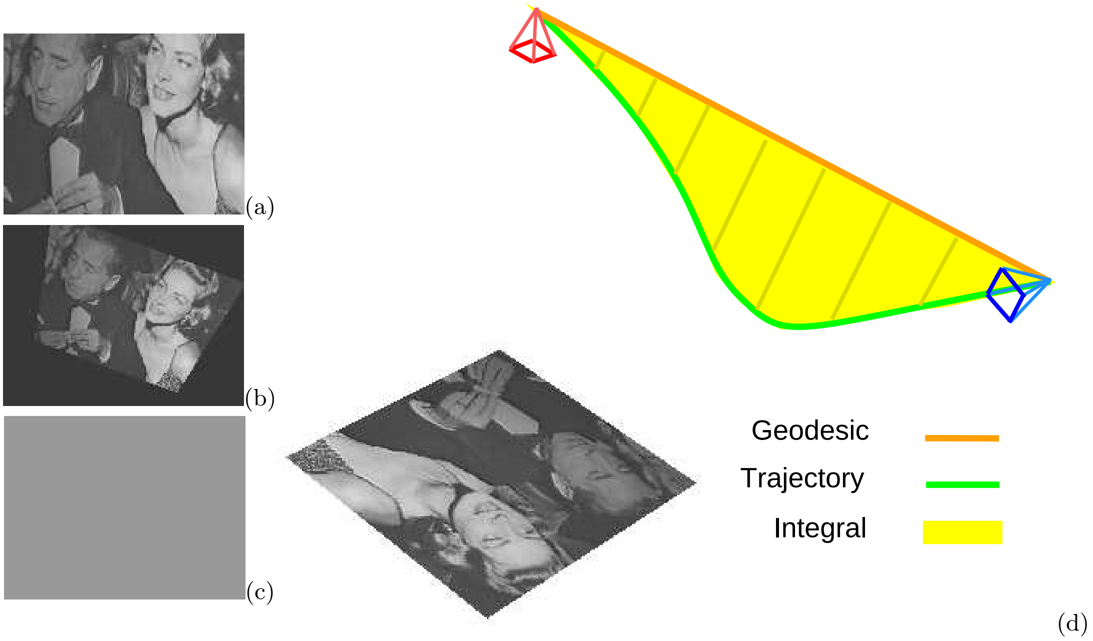
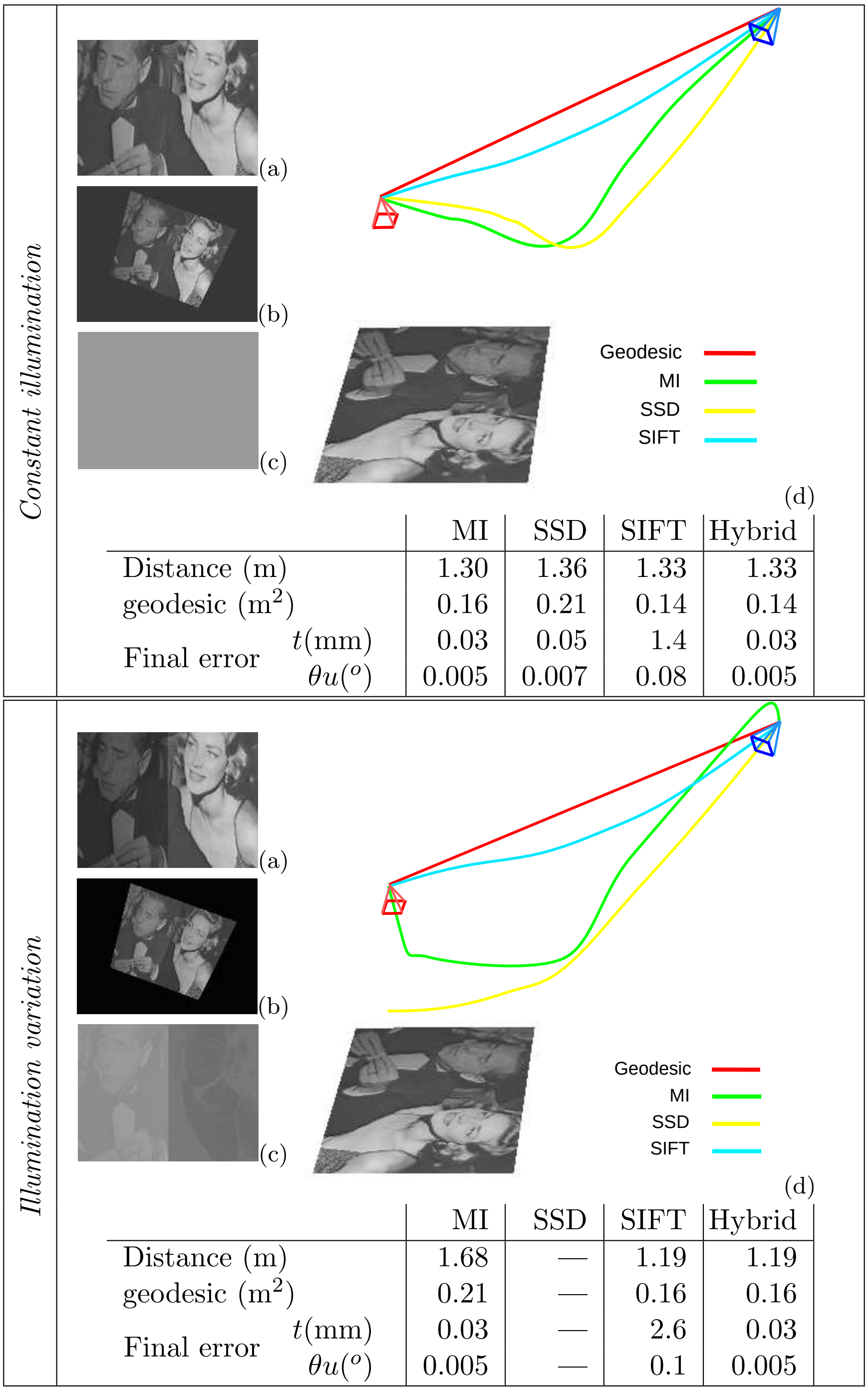


Figure 7.10: Illustration of the performance metrics on one task. (a) Reference image, (b) image at the initial pose, (c) final image error and (d) resulting trajectory (green) of the camera from the initial pose (blue) to the desired pose (red).



		MI	SSD	SIFT	Hybrid
Distance (m)		1.30	1.36	1.33	1.33
geodesic (m^2)		0.16	0.21	0.14	0.14
Final error	t(mm)	0.03	0.05	1.4	0.03
	$\theta u(^o)$	0.005	0.007	0.08	0.005

		MI	SSD	SIFT	Hybrid
Distance (m)		1.68	—	1.19	1.19
geodesic (m^2)		0.21	—	0.16	0.16
Final error	t(mm)	0.03	—	2.6	0.03
	$\theta u(^o)$	0.005	—	0.1	0.005

Figure 7.11: Comparison between our MI based VS, the photometric VS (SSD), a SIFT based VS and an hybrid VS with and without illumination variations. (a) Reference image, (b) Image acquired at the initial pose, (c) final error image, (d) trajectories in the 3D space. The tables show the corresponding values of the performance metrics.

Chapter 8

Mutual information-based visual non holonomic navigation

In this thesis we propose a new approach based on the mutual information based visual servoing, that no longer relies on geometrical features nor on pixels intensity but uses directly the information contained in the images. We propose an approach that requires only a sequence of key images that are acquired from a camera mounted on the vehicle in a learning step. Using the current image acquired from the vehicle and one of the key images, we show that it is possible to compute the interaction matrix that relates the variation of the mutual information to the vehicle velocity leading to the definition of the control law. The variation of the mutual information related to the camera rotation velocity is computed as in the classical visual servoing problem. The link between the translational velocity, rotational velocity and the steering angle is then obtained using the model of the vehicle. Once the steering angle is computed the vehicle can navigate and the key images are selected. A simultaneous process also based on the MI based VS approach is proposed to switch from one key frame to another.

As for the visual servoing approach, the navigation task does not require any features extraction, tracking, matching step or 3D reconstruction. The approach remains very robust to illumination variations, to large occlusions, to the wind in the trees and several other appearance variations.

8.1 Navigation process overview

This section describes briefly the implemented visual navigation framework. In this work, we consider a non-holonomic robot with a camera mounted on the front. Our goal is not to localize the robot within its environment (visual odometry) but only to ensure that it is able to reproduce a visual path defined as a set of images previously acquired by the camera.

Learning step: definition of the visual path

With respect to the approaches that rely on 3D reconstruction (eg. [Royer 2007]) or even on appearance-based approaches [Segvic 2009], the learning step of our method is simple. It does not require any features extraction nor scene reconstruction: no image processing is done, only raw images are stored. The vehicle is driven manually along a desired path. While the vehicle is moving, the images acquired by the camera are stored chronologically thus defining a trajectory in the image space. Let us call $I_0^*, \ldots, I_N^*$ the key images that define this visual path (see Figure 8.1).

Navigation step: following the visual path

The vehicle is initially placed close to the initial position of the learned visual path (defined by the image $I^*{}_0$). The navigation is performed using a visual servoing task. Figure 8.2 shows the general control scheme used for the navigation. In [Royer 2007, Segvic 2009, Chen 2009] the considered control scheme is either pose-based control law or consider classical visual servoing

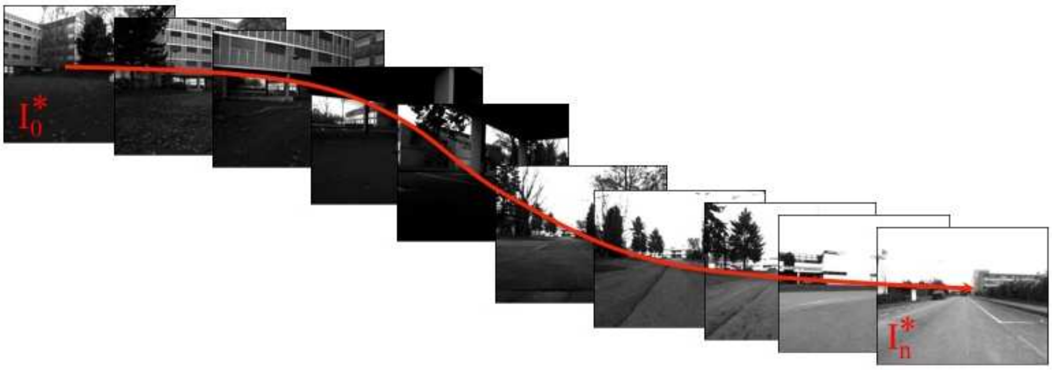


Figure 8.1: Key images that define the visual path. This visual path is learned prior to the navigation step.

process based on the use of visual features extracted from the current and key images (I and I_k^*).

In this work the definition of a new control law is proposed. One of the originality of this work is that, rather than relying on features extraction and tracking, we build the control law directly from the information shared by I and I_k^*, their mutual information. When the mutual information between two images is maximized, the two images are similar. We then control the robot in order to maximize the mutual information between I and I_k^*. As for any visual servoing scheme, it is then necessary to exhibit the gradient that links the variation of the mutual information to the control input of the robot (that is the steering angle ψ or the camera rotational velocity $\dot{\rho}$) needed to follow the path with a constant translational velocity v_z. This process, already presented in the previous chapter for a generic 6 dof robot, is presented in the next section in the specific case of a non holonomic vehicle. In the same time, when the vehicle reaches the key image I_k^*, a new one I_{k+1}^* is selected in the visual path. To achieve a seamless switching between key images, a specific process is proposed.

8.2 Mutual information based navigation

Considering that the translational velocity is constant, we consider in this section that the navigation consists of controlling the rotational velocity of the robot. The rotational velocity is obtained with a visual servoing approach. The model of the vehicle is then used to retrieve the steering angle with respect to the velocity.

8.2.1 Navigation as a visual servoing task

In the previous chapter, it has been shown that it is possible to achieve a 6 dof visual servoing task using only the information contained in the images acquired from a camera mounted on a robot and one reference image. The robot reached its desired position by maximizing the mutual information between the current and reference images. Our navigation approach remains very similar in the sens that it consists of maximizing the mutual information using only the rotational velocity around the y axis of the camera (the vertical axis). If we assume that the translational velocity is known (for example set to a constant value or depending on the steering angle), the basic approach remains therefore to compute the rotational velocity that corresponds to a simplification of the 6 dof visual servoing problem to one degree of freedom. The problem becomes to find the camera orientation ρ that maximizes the mutual information as:

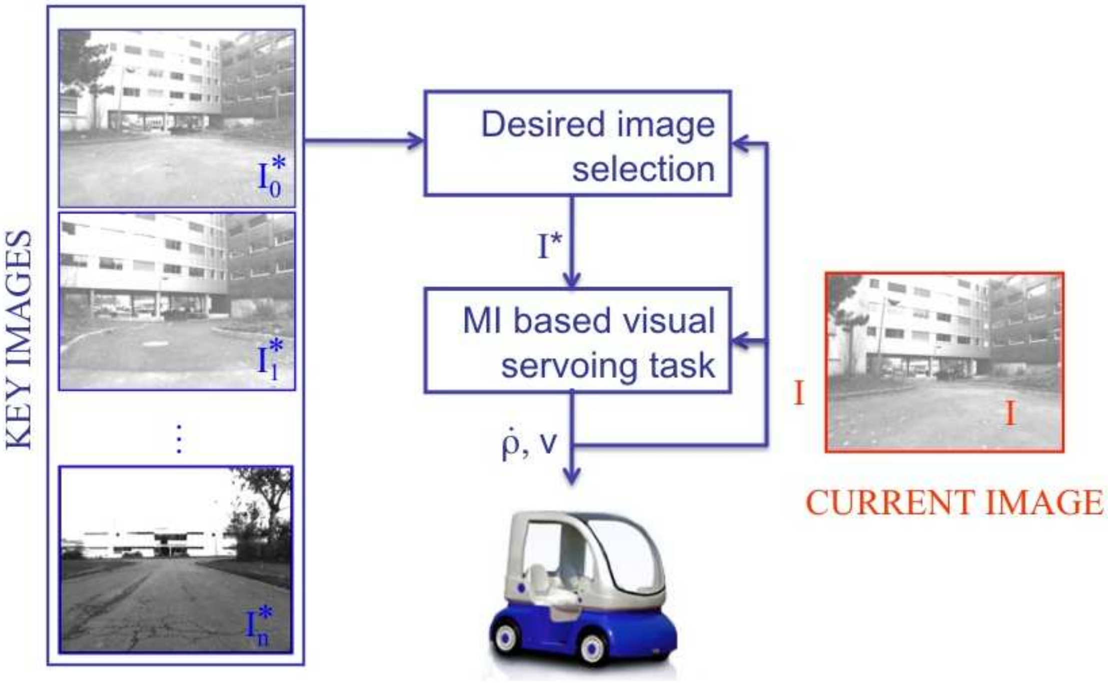


Figure 8.2: Navigation based on multiple visual servoing tasks.

$$\widehat{\rho^*} = \arg\max_{\rho} \mathrm{MI}(I^*, I(\rho)). \tag{8.1}$$

One difference is that the more the rotational error between the current and key images is large the more the steering angle has to be large. The use of a constant Hessian (the HC method) as it was advised in the previous chapters is not suited in the navigation context. Indeed as it was explained in the visual servoing problem, the use of the constant Hessian causes the update to be small far from the convergence.

To avoid this problem, the classical Newton's method is preferred. Since the Newton's method fails when the function is convex (in a maximization problem), the images are smoothed and a small number of histogram bins are chosen to have the largest possible concave domain (see section 2.2.3). Using the classical Newton's method, the rotational velocity $\dot{\rho} = \omega_y$ can be computed using:

$$\dot{\rho} = -\alpha \mathbf{H}_{\mathrm{MI}}^{-1} \mathbf{L}_{\mathrm{MI}}^{\top} \tag{8.2}$$

where $\mathbf{L}_{\mathrm{MI}}$ and $\mathbf{H}_{\mathrm{MI}}$ are respectively the interaction matrix and Hessian that links the variation of MI with the rotation $\dot{\rho} = \omega_y$. In this problem, since ω_y has one dimension, $\mathbf{L}_{\mathrm{MI}}$ and $\mathbf{H}_{\mathrm{MI}}$ are in fact just scalar values. $\alpha \in [0, 1]$ is a scalar factor that allows to set the speed of the convergence (in our navigation task the goal is not to get to the maximum in one iteration). The divergence problem caused by the quasi-concavity of the function is easily avoided in our 1D optimization problem: a simple verification on the Hessian sign is enough to verify if the local approximation of the function is concave, if it is not, the rotation is very large so that the steering angle is set to its maximal value. Nevertheless, this case had never occurred in our experiments since the MI function was always concave. The computation of $\mathbf{L}_{\mathrm{MI}}$ and $\mathbf{H}_{\mathrm{MI}}$ is

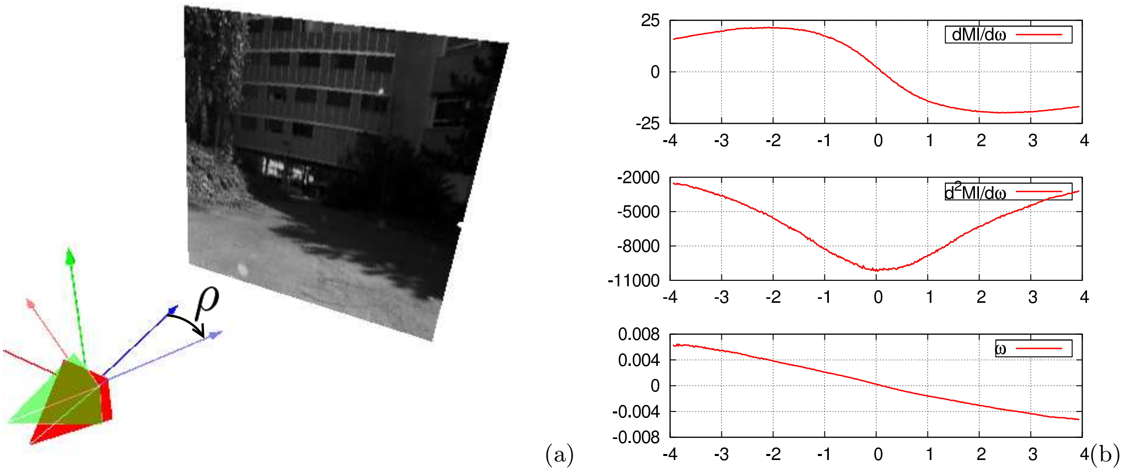


Figure 8.3: Compute of mutual information and its derivatives with respect to the rotation around the vertical axis ρ. (a) External view of the camera at the desired position in red (used to acquire the desired image I^*) and at one measurement position in green. (b) Derivatives of mutual information and corresponding rotational update. First row: derivative of mutual information with respect to the rotational error ρ (°), second row: second derivative and third row: computed velocity ω_y.

similar to the classical visual servoing problem:

$$\mathbf{L}_{\mathrm{MI}} = \sum_{i,j} \mathbf{L}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) \tag{8.3}$$

$$\mathbf{H}_{\mathrm{MI}} = \sum_{i,j} \mathbf{L}_{p_{II^*}}^\top \mathbf{L}_{p_{II^*}} \left(\frac{1}{p_{II^*}} - \frac{1}{p_{I^*}}\right) + \mathbf{H}_{p_{II^*}} \left(1 + \log\left(\frac{p_{II^*}}{p_{I^*}}\right)\right) \tag{8.4}$$

where the detailled computation is exactly the same as the one in the classical visual servoing problem. The only difference with the 6 dof visual servoing problem of the previous part is that this time, the interaction matrix $\mathbf{L_x}$ and Hessian $\mathbf{H_x}$ of a point are simply expressed with respect to the rotation velocity ω_y:

$$\mathbf{L_x} = \begin{bmatrix} -(1+x^2) \\ -xy \end{bmatrix} \qquad \text{and} \qquad \mathbf{H_x} = \begin{bmatrix} 2x(1+x^2) \\ y(1+2x^2)] \end{bmatrix} \tag{8.5}$$

The resulting velocities have been computed on the example illustrated in Figure 8.3. The figure shows the value of the derivatives of mutual information and the resulting computed camera velocity depending on the rotation between the desired and the current positions. The relation between the real rotation and the computed update is quasi linear. The result of this proposed update will then cause a quasi exponential decreasing of the error, that is the ideal goal of typical visual servoing tasks.

8.2.2 Application to the non-holonomic vehicle

For every acquisition of an image I, a velocity ω_y is computed in order to move the camera and increase mutual information between I and I^*. Using the model of the vehicle, we can find back the steering angle that will produce the required rotational velocity.

The velocity of the camera is directly linked to the steering angle ψ of the wheels. Using the model of the non-holonomic vehicle used in our experiments (see the car-like model on Figure 8.4), the general steering angle is computed as follows:

$$\psi = \arctan\left(\frac{L\,\omega_y}{v_z}\right) \tag{8.6}$$

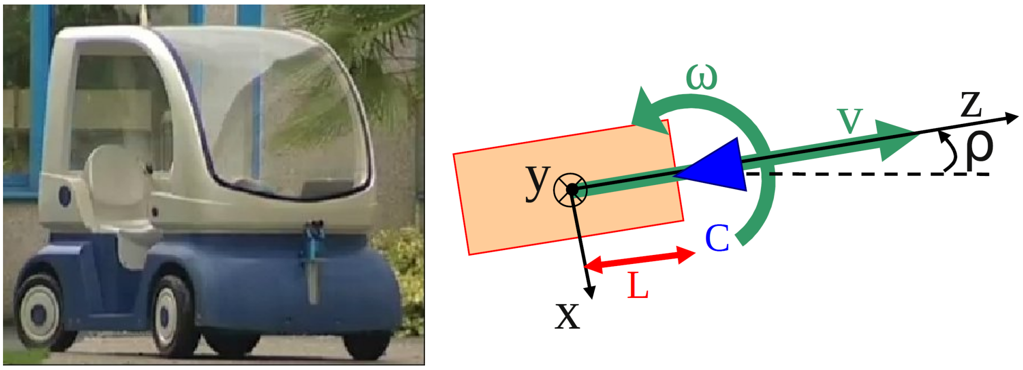


Figure 8.4: Considered vehicle and its model.

where v_z is the translational velocity of the vehicle (along the z axis) and L is the distance between the front and rear wheels.

8.3 Switching between key images in the visual path

The previous section shows how to control a vehicle toward one key image using the information shared by the current and key images. To be able to follow the learned trajectory a switching process between the key images of the visual path has to be defined.

8.3.1 Various solutions

At first sight this problem shows several solutions. Since mutual information is supposed to increase in the visual servoing task, one could propose to simply analyze the cost function evolution and check if the function (equation (8.1)) is increasing. If it is no longer the case, it could mean that the key image is outdated. This solution is unfortunately limited to nominal or simple cases. In an outdoor environment, such conditions are unpractical: if the illumination is changing then the cost function value will be affected and this simple selection process of the key image may fail.

Another solution could be to consider the rotation required to reach the alignment position. We can assume that this rotation is provided by the resulting camera velocity ω_y (obtained using the visual servoing approach). In this case, if the computed rotation is smaller than a given threshold it could mean that the vehicle is next to the desired position, then the next key image can be loaded. But such a simplistic solution will obviously fail when the tracked trajectory is a straight line.

8.3.2 Proposed key image selection process

The proposed approach uses the last described solution coupled with a translation estimation along the z axis between the camera current position and the current key image. Two conditions are then verified: first, the remaining rotational error between the current and desired image must be low and, second, the translational error between the two images is small.

The rotation required to reach I^* is linked with the velocity ω_y. However no estimation is given a priori on the translational error (that is the remaining distance between the current position and the position corresponding to the key image).

Our approach is to consider the variation of the mutual information between the key image and the current image depending on the variation of translational position t_z of the vehicle. If the variation of MI is null, it means that the maximum of MI is reached and that the robot is at the desired position. If the variation is positive (respectively negative) it means that mutual information is increasing (respectively decreasing) and that the robot is getting closer to (respectively moving away from) the desired position.

This variation is simply computed as the derivation of mutual information with respect to the translational velocity v_z of the camera. The formulation of the problem is similar to the one proposed to estimate the rotational velocity ω_y, as the difference that the current image is now depending on v_z. The derivative of the mutual information is now expressed with the interaction matrix corresponding to the translational degree of freedom that is $\mathbf{L}_{t_z} = [x/Z \quad y/Z]^t$ where Z is the depth of each point. Since an accurate estimation of the translation is not needed, Z is approximated to be constant with 20 meters.

To validate the proposed approach, some simulations have been performed using an environment with a non constant depth. Figure 8.5 illustrates the performed experiment. The value of the derivative of mutual information is shown depending on the translation between the current and the key position along the z axis. We can see that the choice of the depth value is not critical (in fact using the previous equations, it can be seen that changing Z is only modifying the derivative up to a scale factor). Considering a strongly non flat scene, mutual information derivative with respect to the translation remains accurate with a null value when the robot reaches the desired translation.

Using two given thresholds on both the parameter update and the translation estimation allows to update the key image each time the robot is close to the current desired position.

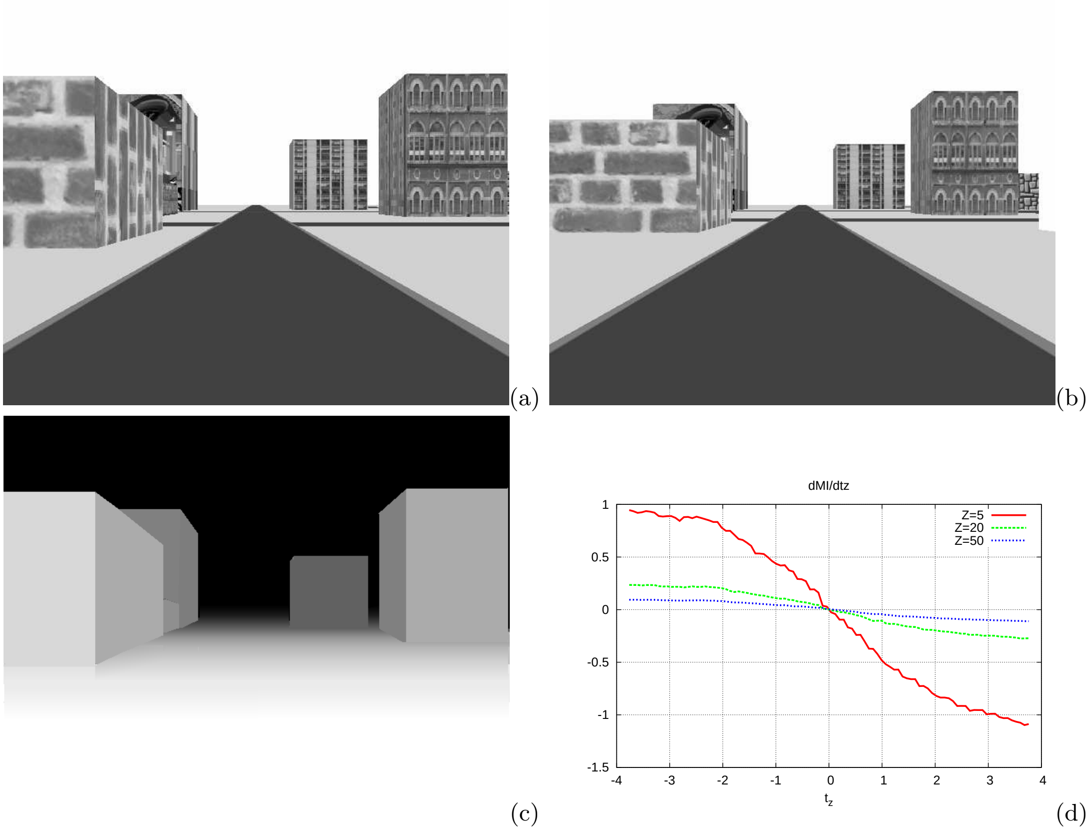


Figure 8.5: Translation estimation between the current and desired image. (a) Desired image, (b) acquired image with a 4 meter translation, (c) scene depth and (d) derivative of mutual information with respect to the translation along the z axis (in meters) with various fixed scene's depth Z.

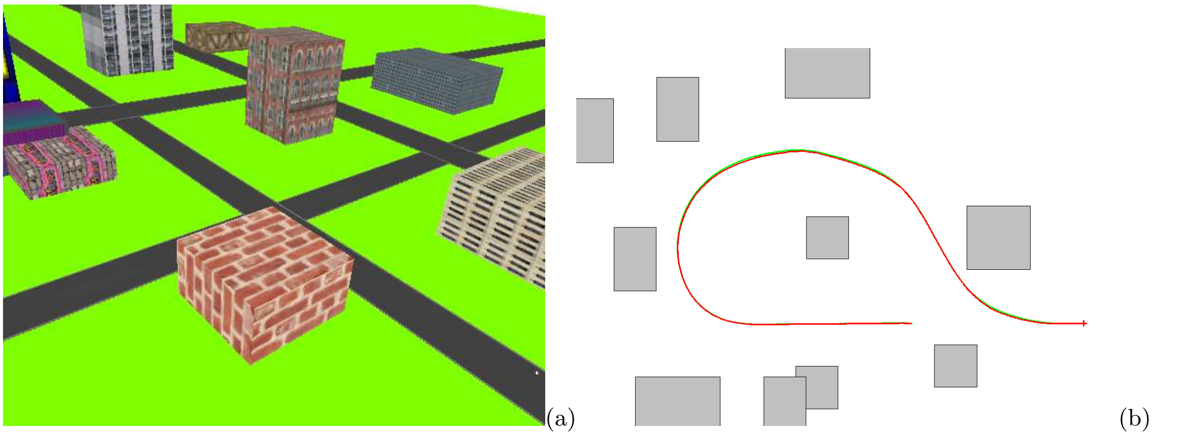


Figure 8.6: Simulation experiment. (a) Aerial view of the environment, (b) 2D representation of map with the learned trajectory in green and the resulting path in red (gray rectangles are buildings).

8.4 Experimental results

This section presents navigation experiments performed with the vehicle represented in Figure 8.4 using the mutual information-based navigation process.

8.4.1 Simulation

The first experiment is a simulation that describes the behavior of the proposed navigation task in nominal conditions. Since this is a simulation, the acquired trajectory and the resulting one are perfectly known. The simulation is performed in an urban-like environment that is shown in Figure 8.6(a). The ground is flat and there are no occlusions nor illumination variations between the environment in the learning and in the navigation phases. The buildings of the environment have various type of textures with low, high and repetitive textures. The experiment has been done using 320 × 240 images. Figure 8.6(b) shows the trajectory used to acquire the learned sequence. The learned sequence contains 400 images on the whole trajectory and the navigation task is performed using 2500 images.

The results have been obtained using a number of histogram bins set to 8. The standard deviation of the Gaussian filter applied on the images is set to 11. The resulting trajectory is represented on Figure 8.6(b) overlaid with the learned trajectory.

Considering the steering angle sent to the vehicle (red plot on Figure 8.7), we can see that the control law is obviously not continuous. Each time the key image is changed, the computed rotation is abruptly changed and then decreases exponentially. The effect on a real vehicle may be hard to endure. To solve this issue, we propose to filter the previous result to have smoother changes of the steering angle. A simple Kalman filter with a constant velocity model has been applied to the computed angle. The result of the Kalman filter is shown on previous computed steering angles (see Figure 8.7). This result is adapted for the control of the vehicle that keeps on following properly the path with smoother changes in its direction.

8.4.2 Navigation in natural environment

The mutual information control scheme has been tested on a non-holonomic vehicle (see Figure 8.4) in an outdoor environment. The final approach presented in the previous paragraph has been used. Let us emphasize that the vehicle is equipped with a monocular camera and that no other sensor such as GPS, radar or odometry are considered in these experiments. Furthermore,

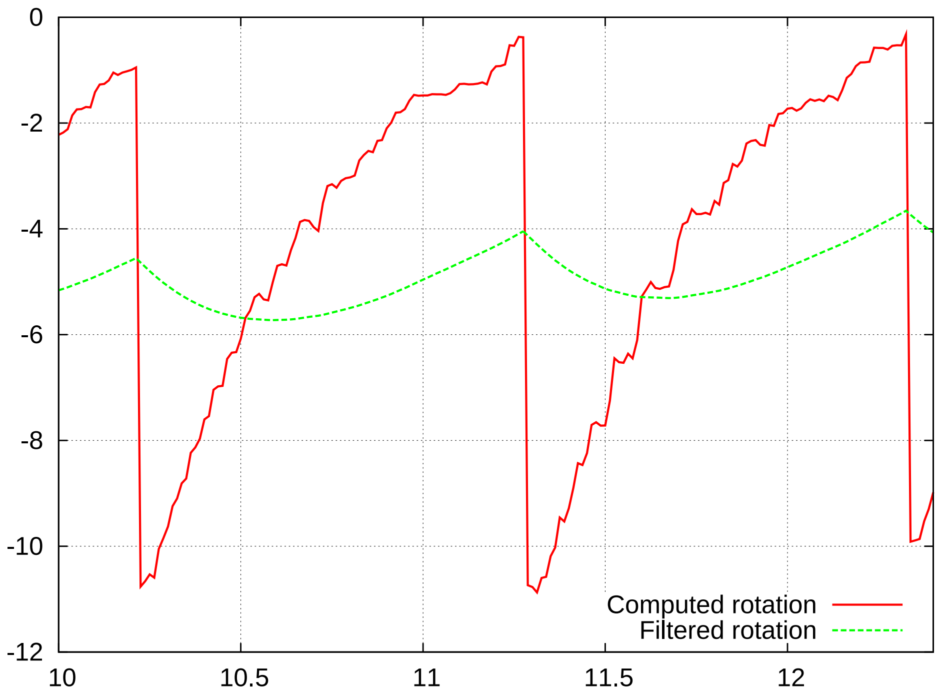


Figure 8.7: Simulation experiment: steering angle (in degrees) in the first turn of the path with respect to the time (in second). The computed steering angle in red shows two exponential decreasement corresponding to two visual servoing tasks, the filtered steering angle is in green.

the 3D structure of the scene remains fully unknown during the learning and navigation steps. Considering the vehicle speed during the learning step, a key-frame has been acquired each meter.

For information, aerial views of the environment, where the navigation task takes place, are shown in Figure 8.8 along with the considered trajectory (about 400 meters). As seen on the pictures, the environment is semi-urban with both trees and buildings (with windows acting as repetitive textures). Let us note that the vehicle crosses a covered parking lot (green part of the trajectory in Figure 8.8) and that the ground is not flat (mainly in the first 100 meters of the trajectory).

When learning the path the vehicle is manually driven at a roughly constant velocity. For this experiment we consider 1200 key images (that is around three key images per meter). The navigation task itself is carried out at 0.5 m/s. Images are acquired at 30Hz (nearly 25000 images are acquired and processed in real-time during this navigation task).

Some pictures of one navigation task are shown in Figure 8.9 and 8.10. By comparing the current and key images (and the image error on the third row), we can see that the robot is qualitatively (as defined in [Remazeilles 2007]) following the same path. The navigation task has been tested with both cloudy and sunny weather using the same learned visual path). Since time had passed between the acquisition of the visual path and the navigation task, there have been very large illumination changes between the current and the key images as it is highlighted in Figure 8.11(a). The task has even been tested with a ground recovered with snow still using the same initial visual path (See Figure 8.11(b)). Despite those illumination and scene variations the navigation task was still converging. This highlights the robustness of the proposed control law to illumination variations and the efficiency of the mutual information similarity criterion to perturbation. Although no ground truth was available, the average metric deviation from the

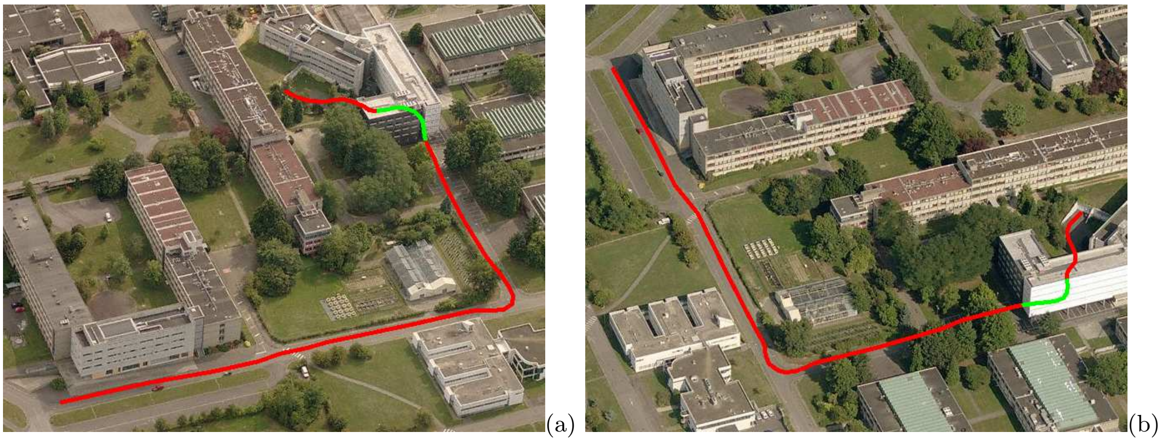


Figure 8.8: Outdoor environment with an approximation of the learned trajectory (the red part is the trajectory executed under a covered parking lot). (a) First aerial view, (b) second aerial view.

original path was less than 1 meter. Since the vehicle is moving in the visual path, we preferred to evaluate the navigation using a visual evaluation (image differences).

Since no obstacle avoidance process is considered, the navigation task has been performed in quiet conditions. Nevertheless several vehicles have overtaken or crossed our experimental vehicle and appeared in the camera view. Despite this perturbation, and thanks to the robustness of the similarity criterion, the navigation task has never failed showing the robustness of mutual information to occlusions (see in Figure 8.11(c) where the van in the current image was not present in the key image).

8.5 Conclusion

In this chapter, we have presented a new way to achieve image based navigation task. We show that our vehicle is able to track a previously learned trajectory using only the information provided by a monocular camera. The navigation task can be achieved despite important variation in the lighting condition and possible perturbations. This can be achieved thanks to various elements related to the use of mutual information that are new in vision-based navigation:

- Our approach does not rely on features extracted from the image. Therefore we do not need to track or match features which has proved to be a difficult and not always reliable process. Furthermore no 3D information related to the scene structure is required.

- Again, as in the classical visual-servoing task, to avoid this tracking and matching processes, the vision-based control law of the non-holonomic vehicle is directly linked to the MI optimization.

- The derivation of the control law requires the derivation (up to the second order) of the interaction that links the vehicle motion to mutual information. The proposed approach that considers a derivation, up to the second order, allows a large convergence domain along with fast (video rate) computation.

- Considering information contained in the images and not features extracted from the image or the image intensities induces a natural robustness to perturbation that is essential in our navigation context.

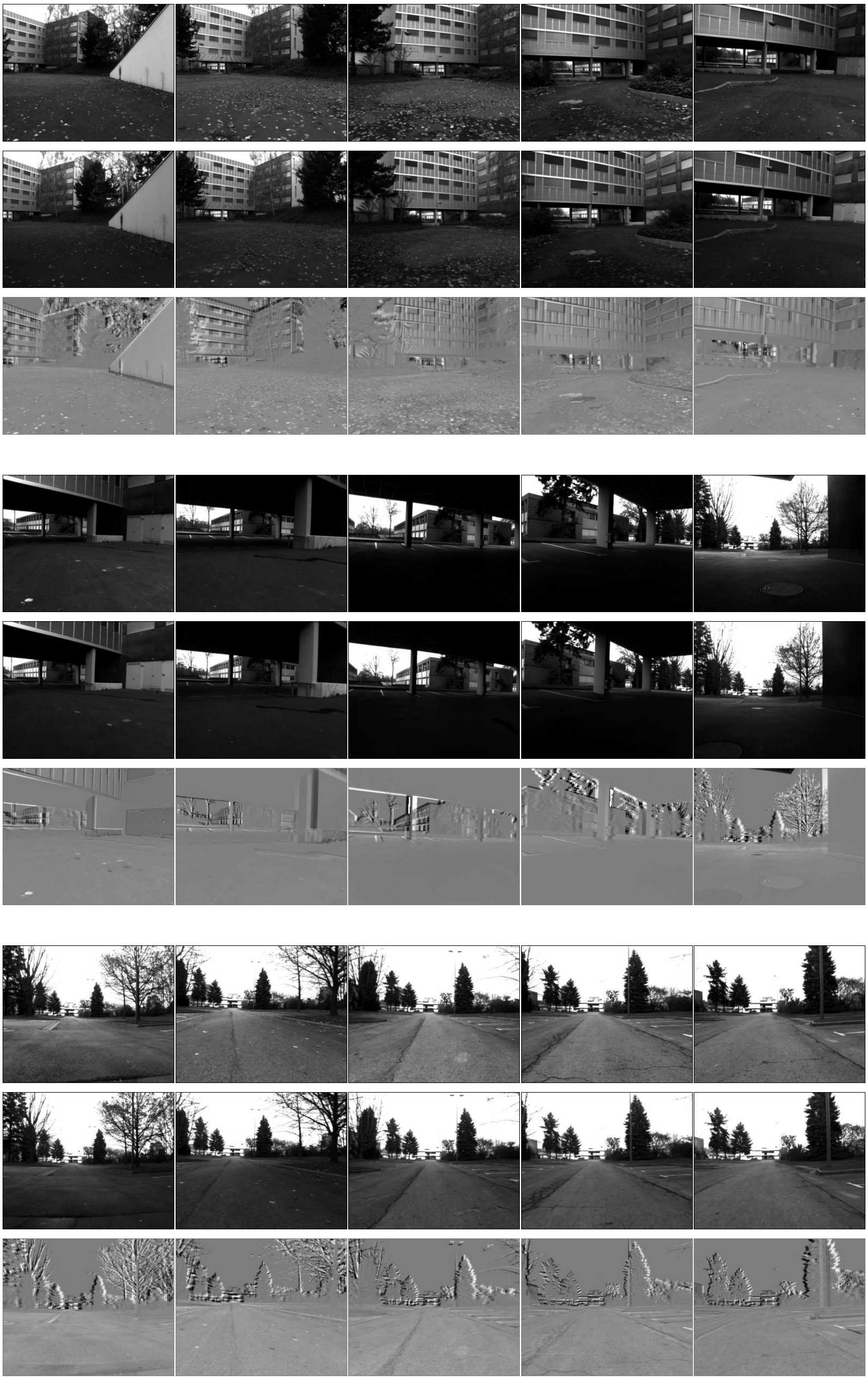


Figure 8.9: Outdoor navigation experiment, begining of the experiment. First row: current image acquired by the vehicle, second row: desired image and third row: difference between the current and desired image. The first top column and the last bottom column show respectively the first images and the last images used in the navigation task.

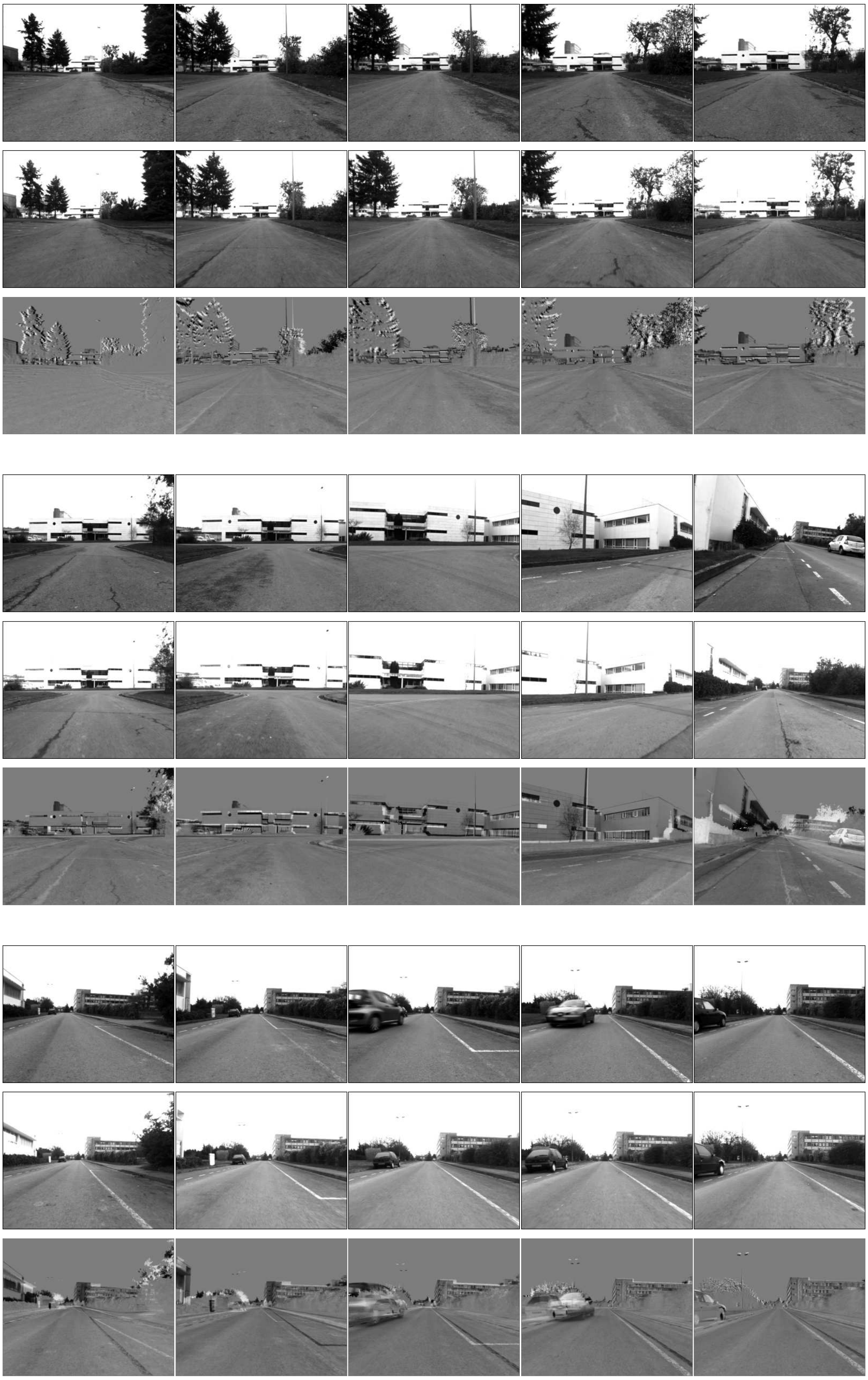


Figure 8.10: Outdoor navigation experiment, end of the experiment.

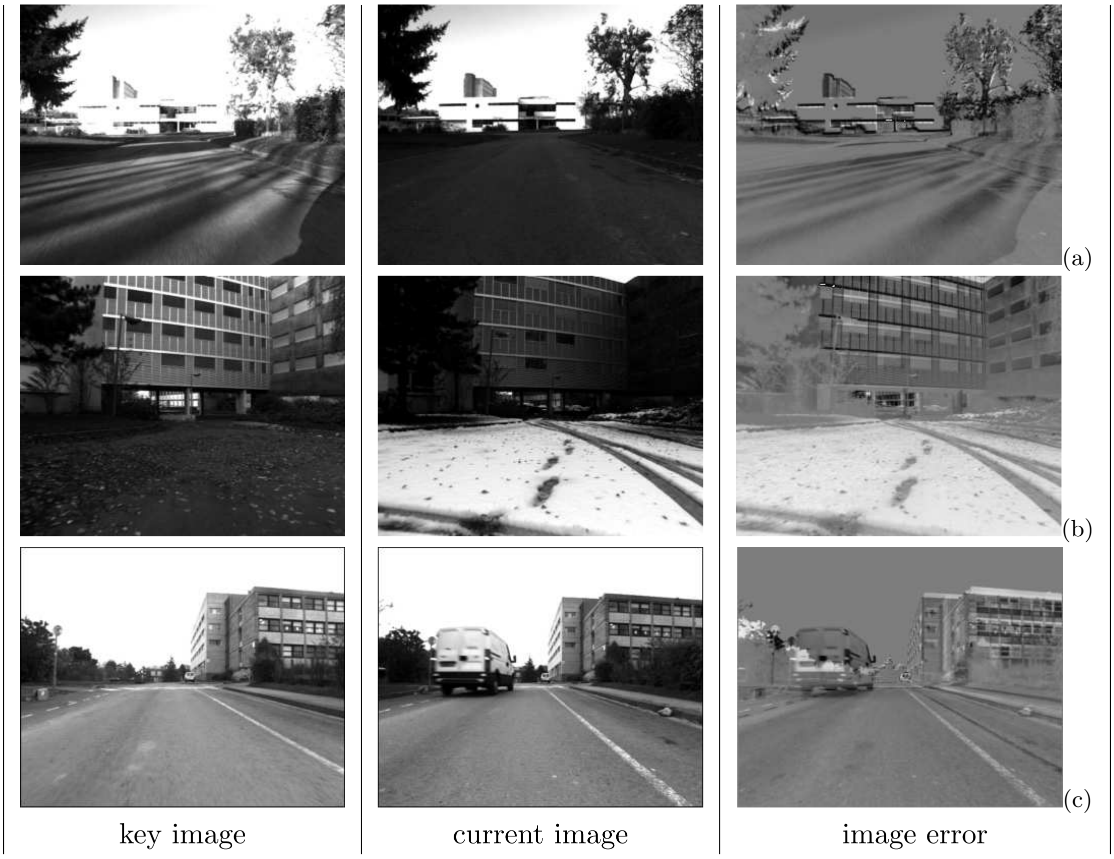


Figure 8.11: Mutual information robustness. (a) robustness to illumination variations, (b) illumination variations and snow on the ground, (c): robustness to occlusions. First column: desired image, second column: current image and third column: difference of the current and desired images.

Future works include improving the key images selection process and the definition of navigation tasks that require more degrees of freedom and more complex control models such as aerial drones.

Conclusion and perspectives

The development of new technologies have led new needs in the computer vision domain. Applications such as augmented reality or special effects require some very robust and accurate visual tracking methods. While others such as autonomous navigation or manipulation require efficient visual servoing techniques. One widespread technique, in both the visual tracking and servoing problems, has been to use some geometrical features of the objects in the image. While they demonstrated to be very efficient in many situations, one remaining problem is that they miss some information of the image that is provided by the pixel intensities and the accuracy of the corresponding tasks is, therefore, not maximal. Considering only the pixel intensities but not the geometrical information has been proposed in the density based approaches, such as the mean-shift. In these methods, the object is only modeled by its global appearance in terms of colors (or grayscale levels). The problem is that no spatial information is included in the model and, therefore, a coarse task (such as a translation estimation) can be performed, but problems such as positioning a robot using 6 degrees of freedom or estimating the pose of an object are impossible to solve.

To deal with these problems, direct approaches have been proposed where the object or scene is directly defined by its whole appearance. These techniques are based on a similarity function that measures the alignment between two sets of pixel intensities. For instance, the classical approach directly considers the difference between the two sets of intensities. Nevertheless, since the appearance of the object is changing with the environmental conditions, we show that the simple difference between the intensities is not robust enough to accurately solve the tracking and servoing problems. To improve the robustness of these approaches in our method, the intensities of the image are not directly used. Instead, we consider the information that is shared between the two sets of pixel intensities by measuring the mutual information similarity function. While the intensities of the image are strongly depending on the illumination conditions, the information that it contains is mainly maintained.

In this document, we presented the researches that we performed during the last three years on this subject. The contributions include a visual tracking approach based on mutual information that uses a new optimization method. This work has shown that the MI based registration is a practical solution to build a robust, accurate and efficient tracking approach. Many applications have been tested with face tracking, mosaicing and augmented reality experiments. We show that this approach can be extended to the model-based pose estimation problem to perform a very accurate estimation of the pose of a camera with respect to an object. Since the information is also maintained between images from different modalities, new multimodal registration applications have also been demonstrated.

Since the visual servoing problem is dual to the pose estimation one, we show that MI provides also an efficient metric for the visual servoing problem. In the continuity with the existing direct visual servoing approaches, we developed a new method that does not require any feature extraction or matching steps. A first advantage comes from the robustness of MI that visual servoing tasks in case of many conditions, such as occlusions and illumination variations. Secondly, since there is no intermediary measure and since the control law is directly computed with respect to the whole image, the positioning task is very accurate. We demonstrated the performances of the approach through many experiments including the positioning task of a 6 dof robot and the navigation of a non-holonomic autonomous vehicle.

Perspectives

Many topics of this research have still to be studied. First, it will be very beneficial for the tracking and visual servoing approaches to have a measure of the confidence of the estimated position. For the moment, we can estimate if the parameters correspond to a maximum of the mutual information. However, there is for the moment no information if this maximum is local or global. Due to the robustness of MI, the estimated parameters are "usually" corresponding to the correct position (*i.e.* the global maximum), but sometimes the process can fail. Working on the cost function, it would be also very interesting to be able to estimate, at the beginning of the tracking or visual servoing process, if the reference image provides enough information to perform the optimization and achieve the task.

Our MI-based tracking approach has shown its efficiency on many applications. This study was limited to rigid objects, but its application to deformable objects will be very favorable for many applications, for instance, in the medical field. We also limit the tracking approach to pin-hole cameras, the extension to other projection models such as central catadioptric cameras would provide new solutions, for instance, in the navigation field.

In this work, several contributions have been proposed on the optimization aspects of the visual tracking and visual servoing approach. The optimization approaches are similar from one cost function to another. An extension to other similarity measures than MI could therefore be possible. For instance, when the problem remains to optimize a quasi-concave (or convex) function that requires a large computation time such as the Zero-mean Normalized Cross Correlation, the proposed approach would be perfectly adapted.

Finishing with the theoretic perspectives, a principal subject of interest in the visual servoing community is to decouple the degrees of freedom of the camera. Many approaches have been proposed in the classical geometric features based methods, but work remains to be done concerning the direct visual servoing methods.

Concerning more the applications, one first objective could be to improve the implementation of the tracking method. Putting the algorithm on dedicated hardware would be very interesting to include the method in embedded systems. Since the accuracy depends on the number of optimization iterations per image, a faster implementation, using for instance graphics processing unit (GPU) would be also an advantage.

Finally, one of our projects is to use our pose estimation approach in an autonomous vehicle navigation application. The project is held in Paris where a partial textured model of the city is already existing. An image of the environment is given as input of the system. Using both this image and the 3D model of the city, it is possible to localize the vehicle and perform the navigation. In this application, the MI based approach would be particularly well suited, since many occlusions can occur from the model to the image, as well as modifications of small parts of the architecture or illumination variations.

Part IV

Appendices

Appendix A

Parametric motion models and differential tracking

In this appendix, we provide some specific knowledge required to use parametric motion models in a differential tracking problem. The first section defines the update rule using an affine motion. This step is required to modify the displacement parameters and make them converge to the optimum of the function to optimize. The second section is dedicated to a pyramidal implementation of the tracking approaches and provides the conversion of the displacement parameters from one level of the pyramid to an other.

A.1 Update rule of the affine warp function

We recall the definition of the affine warp function given in section 1.2.2.3

$$w(\mathbf{x},\mathbf{p}) = \mathbf{A}\mathbf{x} + \mathbf{t} = \begin{bmatrix} p_0 & p_1 \\ p_2 & p_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} p_4 \\ p_5 \end{bmatrix}$$

The goal of this appendix is to find the displacement parameters $\mathbf{p}'$ resulting from the composition of $\mathbf{p}$ and $\boldsymbol{\Delta}\mathbf{p}$. Following the update rule defined in equation 4.3 we have:

$$\mathbf{x}' = w(w(\mathbf{x},\boldsymbol{\Delta}\mathbf{p}),\mathbf{p}) = w(\mathbf{x},\mathbf{p}')$$

Developing using the affine function, it yields:

$$\begin{aligned}
\mathbf{x}' &= \begin{bmatrix} 1+p_0 & p_2 \\ p_1 & 1+p_3 \end{bmatrix} \left(\begin{bmatrix} 1+\Delta p_0 & \Delta p_2 \\ \Delta p_1 & 1+\Delta p_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta p_4 \\ \Delta p_5 \end{bmatrix} \right) \begin{bmatrix} p_4 \\ p_5 \end{bmatrix} \\
&= \begin{bmatrix} 1+p_0+\Delta p_0+p_0\Delta p_0+p_2\Delta p_1 & p_2+\Delta p_2+p_0\Delta p_2+p_2\Delta p_3 \\ p_1+\Delta p_1+p_1\Delta p_0+p_3\Delta p_1 & 1+p_3+\Delta p_3+p_1\Delta p_2+p_3\Delta p_3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \\
&\quad + \begin{bmatrix} p_4+\Delta p_4+p_0\Delta p_4+p_2\Delta p_5 \\ p_5+\Delta p_5+p_1\Delta p_4+p_3\Delta p_5 \end{bmatrix} \\
&= \begin{bmatrix} p_0' & p_2' \\ p_1' & p_3' \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} p_4' \\ p_5' \end{bmatrix}
\end{aligned}$$

By identification, the resulting parameters $\mathbf{p}'$ are obtained using:

$$\mathbf{p}' = \begin{bmatrix} p_0+\Delta p_0+p_0\Delta p_0+p_2\Delta p_1 \\ p_1+\Delta p_1+p_1\Delta p_0+p_3\Delta p_1 \\ p_2+\Delta p_2+p_0\Delta p_2+p_2\Delta p_3 \\ p_3+\Delta p_3+p_1\Delta p_2+p_3\Delta p_3 \\ p_4+\Delta p_4+p_0\Delta p_4+p_2\Delta p_5 \\ p_5+\Delta p_5+p_1\Delta p_4+p_3\Delta p_5 \end{bmatrix}$$

A.2 Pyramidal conversion of the displacement parameters

Let us consider that we know the displacement parameter $\mathbf{p}^k$ of the k^{th} level of the pyramid and that we seek the parameter $\mathbf{p}^{k-1}$. A simple way to solve the problem is to consider the displacement of a point $\mathbf{x}^k$ into the point $\mathbf{x}_2^k$ at the k^{th} level with:

$$\mathbf{x}_2^k = w(\mathbf{x}^k, \mathbf{p}^k) \tag{A.1}$$

The corresponding displacement parameters $\mathbf{p}^{k-1}$ should as well modify the corresponding points of the lower level of the pyramid following:

$$\mathbf{x}_2^{k-1} = w(\mathbf{x}^{k-1}, \mathbf{p}^{k-1}) \tag{A.2}$$

Knowing that, from one level of the pyramid to another, we have:

$$\begin{align} \mathbf{x}^k &= \frac{1}{2}\mathbf{x}^{k-1} \tag{A.3}\\ \mathbf{x}_2^k &= \frac{1}{2}\mathbf{x}_2^{k-1} \tag{A.4}\\ & \tag{A.5} \end{align}$$

Thus:

$$\begin{align} w(\mathbf{x}^{k-1}, \mathbf{p}^{k-1}) &= 2\mathbf{x}_2^k \tag{A.6}\\ &= 2w(\mathbf{x}^k, \mathbf{p}^k) \tag{A.7} \end{align}$$

For every $\mathbf{x}$ we thus have:

$$w(\mathbf{x}, \mathbf{p}^{k-1}) = 2w\left(\frac{1}{2}\mathbf{x}, \mathbf{p}^k\right) \tag{A.8}$$

that is sufficient to estimate the parameters $\mathbf{p}^{k-1}$.

Example Let us consider the homography example. For simplicity, we note $\mathbf{p}' = \mathbf{p}^{k-1}$ and $\mathbf{p} = \mathbf{p}^k$. The warp function is given by:

$$\mathbf{x}_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = w(\mathbf{x}, \mathbf{p}) = \frac{1}{p_6 x + p_7 y + 1}\begin{bmatrix} p_0 x + p_2 y + p_4 \\ p_1 x + p_3 y + p_5 \end{bmatrix} \tag{A.9}$$

Therefore the equation (A.8) yields:

$$\begin{align} \frac{1}{p_6 x + p_7 y + 1}\begin{bmatrix} p_0 x + p_2 y + p_4 \\ p_1 x + p_3 y + p_5 \end{bmatrix} &= \frac{2}{p_6' \frac{x}{2} + p_7' \frac{y}{2} + 1}\begin{bmatrix} p_0' \frac{x}{2} + p_2' \frac{y}{2} + p_4' \\ p_1' \frac{x}{2} + p_3' \frac{y}{2} + p_5' \end{bmatrix} \tag{A.10}\\ &= \frac{1}{\frac{p_6'}{2} x + \frac{p_7'}{2} y + 1}\begin{bmatrix} p_0' x + p_2' y + 2p_4' \\ p_1' x + p_3' y + 2p_5' \end{bmatrix} \tag{A.11} \end{align}$$

By identification, the projection in the k^{th} level of the pyramid of $\mathbf{p}^{k-1}$ is given by $\mathbf{p}^k = (p_0^{k-1}, p_1^{k-1}, p_2^{k-1}, p_3^{k-1}, 2p_4^{k-1}, 2p_5^{k-1}, p_6^{k-1}/2, p_7^{k-1}/2)^\top$.

Appendix B

Derivative computation

In this appendix, we detail some of the derivative computation that are required in the tracking and visual servoing approaches. First, the mutual information derivation is detailed. This derivation is corresponding to both the tracking and visual servoing problems. Then, the Jacobian of a pixel position is computed with respect to the camera pose, to show different approaches that can be chosen to achieve a pose estimation task.

B.1 Mutual information derivatives

We recall that the mutual information between two images I and I^* is given with respect to the normalized histograms and joint histograms of the images by (see equation 2.15):

$$\mathrm{MI}(I, I^*) = \sum_{i,j} p_{II^*}(i,j) \log\left(\frac{p_{II^*}(i,j)}{p_I(i)p_{I^*}(j)}\right) \tag{B.1}$$

Since only the current image I depends on the parameters $\mathbf{r}$, the resulting mutual information derivative is:

$$\mathbf{G} = \sum_{i,j} \frac{\partial p_{II^*}}{\partial \boldsymbol{\Delta}\mathbf{p}} \log \frac{p_{II^*}}{p_I p_{I^*}} + \frac{\partial p_{II^*}}{\partial \boldsymbol{\Delta}\mathbf{p}} - \frac{\partial p_I}{\partial \boldsymbol{\Delta}\mathbf{p}} \frac{p_{II^*}}{p_I} \tag{B.2}$$

For the purpose of clarity, the marginal probabilities and joint probability that are actually depending on i, j, $\mathbf{p}$ and $\boldsymbol{\Delta}\mathbf{p}$ are simply denoted as p_I, p_{I^*} and p_{II^*}. Since the summation of the probability function is always constant with $\sum p = 1$, the summation of its derivative is always null. Therefore, the last term of the expression can be decomposed and simplified as:

$$\begin{aligned}
\sum_{i,j} \frac{\partial p_I}{\partial \boldsymbol{\Delta}\mathbf{p}} \frac{p_{II^*}}{p_I} &= \sum_i \sum_j \frac{\partial p_I}{\partial \boldsymbol{\Delta}\mathbf{p}} \frac{p_{II^*}}{p_I} \\
&= \sum_i \frac{\partial p_I}{\partial \boldsymbol{\Delta}\mathbf{p}} \frac{1}{p_I} \sum_j p_{II^*} \\
&= \sum_i \frac{\partial p_I}{\partial \boldsymbol{\Delta}\mathbf{p}} \\
&= 0
\end{aligned}$$

The expression of the gradient is then:

$$\begin{aligned}
\mathbf{G} &= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I p_{I^*}}\right) \\
&= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I} - \log(p_{I^*})\right) \\
&= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I}\right) - \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \log p_{I^*} \\
&= \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I}\right) - \sum_{j} \frac{\partial I^*}{\partial \mathbf{\Delta p}} \log p_{I^*}
\end{aligned}$$

Since the reference image is constant, the derivative of its histogram is null and the last term of this expression is also null. Finally the expression of the gradient of the mutual information is given by the expression:

$$\mathbf{G} = \sum_{i,j} \frac{\partial p_{II^*}}{\partial \mathbf{\Delta p}} \left(1 + \log \frac{p_{II^*}}{p_I}\right) \tag{B.3}$$

B.2 Optimization on SE(3) using the Rodrigues formula

A very straightforward solution to study the position variation of a point with respect to the pose of the camera is to directly use the projection model defined in section 1.1.1 and write the non-linear relation between the projection of the point and the camera pose parameters $\mathbf{r}$. To focus on the extrinsic parameters let us first consider the projection of the point in meters with a focal distance of 1 meter so that the intrinsic parameters are not involved [1]. If the projection function is noted h, the projection $\mathbf{x}$ of a 3D point $\mathbf{X}_o$ defined in the object frame is given by:

$$\mathbf{x}(\mathbf{r}) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix} = h\left(\mathbf{X}_c\right) = h(\mathbf{R}(\boldsymbol{\omega})\mathbf{X}_o + \mathbf{T}(\boldsymbol{\nu})) \tag{B.4}$$

where $\mathbf{R}(\boldsymbol{\omega})$ and $\mathbf{T}(\boldsymbol{\nu})$ are the rotation and translation matrices resulting from the parameters $\mathbf{r} = (\boldsymbol{\nu}, \boldsymbol{\omega})$.

B.2.1 Forward additional formulation

If the parameters update is directly applied by an addition using a forward additional formulation, the goal becomes to compute the variation of the function:

$$\mathbf{x}(\mathbf{r} + \mathbf{\Delta r}) = h(\mathbf{R}(\boldsymbol{\omega} + \mathbf{\Delta}\omega)\mathbf{X}_o + \mathbf{T}(\boldsymbol{\nu} + \mathbf{\Delta}\nu)) \tag{B.5}$$

with respect to $\mathbf{\Delta r}$. The derivative of this expression is computed for $\mathbf{\Delta}\omega = \mathbf{0}$ and it can thus be written:

$$\frac{\partial \mathbf{x}(\mathbf{r} + \mathbf{\Delta r})}{\partial \mathbf{\Delta r}} = \frac{\partial h(\mathbf{X}_c)}{\partial X_c} \begin{bmatrix} \frac{\partial \mathbf{T}(\boldsymbol{\nu}+\mathbf{\Delta}\nu)}{\partial \mathbf{\Delta r}} & \frac{\partial \mathbf{R}(\boldsymbol{\omega}+\mathbf{\Delta}\omega)\mathbf{X}_o}{\partial \mathbf{\Delta r}} \end{bmatrix} \tag{B.6}$$

From the definition of h the projection into the image plan given in equation (B.4), its derivative with respect to the 3D coordinates of the object in the camera frame is:

$$\frac{\partial h(\mathbf{X}_c)}{\partial X_c} = \begin{bmatrix} X_c/Z_c & 0 & -X_c/Z_c^2 \\ 0 & Y_c/Z_c & -Y_c/Z_c^2 \end{bmatrix}. \tag{B.7}$$

[1] The focal approximation is only changing the expression up to a scale factor.

In equation (B.6) the derivative of the translational part is evident and is the identity matrix. As for the rotational part, the expression of the rotation matrix using the exponential map defined in equation (1.4) gives no way to compute the derivative easily. Nevertheless it has been proven that the exponential map can be equivalently rewritten using the Rodrigues formula:

$$\mathbf{R}(\boldsymbol{\omega}) = \cos\theta \mathbf{I} + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\omega}\boldsymbol{\omega}^\top + \frac{\sin\theta}{\theta}\boldsymbol{\Omega}. \tag{B.8}$$

Using this expression it becomes possible to compute the derivative of the rotational part. But, even if the derivative can be obtained, its expression becomes quickly unreadable and difficult to implement.

Nevertheless one can observe that if $\boldsymbol{\omega}$ is very small, then the rotation matrix and its derivative computed using the Rodrigues formula becomes very simple. As it has been shown in the tracking problem, it is possible to change the general formulation to satisfy this condition using a compositional approach.

B.2.2 Forward compositional formulation

A solution to simplify the problem is to consider that the new position resulting from the update is a composition of a first transformation using $\mathbf{r}$ and a second one using $\boldsymbol{\Delta}\mathbf{r}$. The new homogeneous matrix ${}^{c'}M_o(\mathbf{r} \circ \boldsymbol{\Delta}\mathbf{r})$ is therefore given by:

$${}^{c'}\mathbf{M}_o(\mathbf{r} \circ \boldsymbol{\Delta}\mathbf{r}) = {}^{c'}\mathbf{M}_c(\boldsymbol{\Delta}\mathbf{r}){}^{c}\mathbf{M}_o(\mathbf{r}).. \tag{B.9}$$

The position of the point in the image plan with respect to $\boldsymbol{\Delta}\mathbf{r}$ becomes:

$$\mathbf{x}(\mathbf{r} \circ \boldsymbol{\Delta}\mathbf{r}) = h(\mathbf{R}(\boldsymbol{\Delta}\omega)(\mathbf{X}_c) + \mathbf{T}(\boldsymbol{\Delta}\nu)). \tag{B.10}$$

This expression can be written using only the 3D point transformed with the current estimated camera position and the transformation resulting from the update. The derivative of this expression with respect to $\boldsymbol{\Delta}\mathbf{r}$ is computed for $\boldsymbol{\Delta}\omega = \mathbf{0}$ and thus, it is given by:

$$\frac{\partial \mathbf{x}(\mathbf{r} \circ \boldsymbol{\Delta}\mathbf{r})}{\partial \boldsymbol{\Delta}\mathbf{r}} = \frac{\partial h(\mathbf{X}_c)}{\partial X_c} \begin{bmatrix} \frac{\partial \mathbf{T}(\boldsymbol{\Delta}\nu)}{\partial \boldsymbol{\Delta}\mathbf{r}} & \frac{\partial \mathbf{R}(\boldsymbol{\Delta}\omega)\mathbf{X}_c}{\partial \boldsymbol{\Delta}\mathbf{r}} \end{bmatrix} \tag{B.11}$$

Again the expression of the rotation matrix $\mathbf{R}(\boldsymbol{\Delta}\omega)$ is given by the Rodrigues formula. Nevertheless, since $\boldsymbol{\Delta}\omega$ is the update and is assumed to be very small, θ can be considered null and the approximation of the Rodrigues formula becomes:

$$\mathbf{R}(\boldsymbol{\Delta}\omega) = \mathbf{I} + \boldsymbol{\Omega}. \tag{B.12}$$

This approximation is very helpful to compute the derivative that simply becomes:

$$\begin{aligned} \frac{\partial \mathbf{x}(\mathbf{r} \circ \boldsymbol{\Delta}\mathbf{r})}{\partial \boldsymbol{\Delta}\mathbf{r}} &= \frac{\partial h(\mathbf{X}_c)}{\partial X_c} \begin{bmatrix} \mathbf{I} & \mathbf{X}_{c_\times} \end{bmatrix} \\ &= \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix} \end{aligned} \tag{B.13}$$

To keep an orthogonal rotation matrix, the rotation matrix corresponding to the update in equation (B.9) is preferably computed using the exact Rodrigues formula.

The readers that are already familiar with the visual servoing problem typically recognize in equation (B.9) the equivalence between the two approaches (see the interaction matrix computation in Frame 9 page 105).

Bibliography

[Ababsa 2007] F.-E. Ababsa and M. Mallem. *Hybrid three-dimensional camera pose estimation using particle filter sensor fusion.* Advanced Robotics, vol. 21, no. 1, pages 165–181, January 2007.

[Abdul Hafez 2008] A.H. Abdul Hafez, S. Achar and C.V. Jawahar. *Visual Servoing Based on Gaussian Mixture Models.* In IEEE Int. Conf. on Robotics and Automation, ICRA'08, pages 3225–3230, Pasadena, California, May 2008.

[Baker 2001] S. Baker and I. Matthews. *Equivalence and Efficiency of Image Alignment Algorithms.* In IEEE conf. on Computer Vision and Pattern Recognition, volume 1, pages 1090 – 1097, December 2001.

[Baker 2004] S. Baker and I. Matthews. *Lucas-kanade 20 years on: A unifying framework.* Int. Journal of Computer Vision, vol. 56, no. 3, pages 221–255, 2004.

[Bar-Shalom 1993] Y. Bar-Shalom and X.-R. Li. Estimation and tracking, principles, techniques, and software. Artech House, Boston, 1993.

[Bay 2006] H. Bay, T. Tuytelaars and L. Van Gool. *SURF: Speeded Up Robust Features.* In 9th European Conference on Computer Vision, Graz Austria, May 2006.

[Belhumeur 1999] P. Belhumeur and G. Hager. *Tracking in 3D: Image Variability Decomposition for Recovering Object Pose and Illumination.* Pattern Analysis & Applications, vol. 2, no. 1, pages 82–91, April 1999.

[Benhimane 2004] S. Benhimane and E. Malis. *Real-time image-based tracking of planes using efficient second-order minimization.* In IEEE/RSJ Int. Conf. on Intelligent Robots Systems, volume 943-948, page 1, Sendai, Japan, October 2004.

[Benhimane 2007] S. Benhimane and E. Malis. *Homography-based 2D Visual Tracking and Servoing.* International Journal of Robotic Research, vol. 26, no. 7, pages 661–676, July 2007.

[Berger 1994] M.-O. Berger. *How to track efficiently piecewise curved contours with a view to reconstructing 3D objects.* In Int. Conf on Pattern Recognition, ICPR'94, pages 32–36, Jerusalem, October 1994.

[Berger 1998] M.-O. Berger and G. Simon. *Robust image composition algorithms for augmented reality.* In Asian Conf. on Computer Vision, ACCV'98, volume 2, pages 360–367, Hong-Kong, 1998.

[Bhattacharyya 1943] A. Bhattacharyya. *On a measure of divergence between two statistical populations defined by their probability distributions.* Bull. Calcutta Math Soc., vol. 35, pages 99–109, 1943.

[Blake 1998] A. Blake and M. Isard. Active contours. Springer Verlag, April 1998.

[Blinn 1977] J. Blinn. *Models of light reflection for computer synthesized pictures.* In ACM Conf. on Computer Graphics and Interactive Techniques, SIGGRAPH'77, pages 192–198, San Jose, California, 1977.

[Bookstein 1978] F.L. Bookstein. The measurement of biological shape and shape change. Springer, 1978.

[Bouguet 2000] J.-Y. Bouguet. *Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm*, 2000. `http://robots.stanford.edu/cs223b04/algo_tracking.pdf`.

[Boukir 1998] S. Boukir, P. Bouthemy, F. Chaumette and D. Juvin. *A local method for contour matching and its parallel implementation.* Machine Vision and Application, vol. 10, no. 5/6, pages 321–330, April 1998.

[Bouthemy 1989] P. Bouthemy. *A Maximum Likelihood Framework for Determining Moving Edges.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 11, no. 5, pages 499–511, May 1989.

[Brown, Matthew 2007] Brown, Matthew and Lowe, David G. *Automatic Panoramic Image Stitching using Invariant Features.* Int. J. Comput. Vision, vol. 74, no. 1, pages 59–73, August 2007.

[Burschka 2001] D. Burschka and G. Hager. *Vision-based control of mobile robots.* In IEEE Int. Conf. on Robotics and Automation, ICRA'2001, volume 2, pages 1707–1713, 2001.

[Cha 2008] S. Cha. *Taxonomy of nominal type histogram distance measures.* In MATH'08: American Conference on Applied Mathematics, pages 325–330, 2008.

[Chaumette 1997] F. Chaumette. *Potential problems of stability and convergence in image-based and position-based visual servoing.* In D.J. Kriegman, G. Hager and A.S. Morse, editeurs, The confluence of vision and control, Lecture Notes in control and information sciences, No 237, pages 67–78. Springer, June 1997.

[Chaumette 2000] F. Chaumette and E. Malis. *2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings.* In IEEE Int. Conf. on Robotics and Automation, volume 1, pages 630–635, San Francisco, CA, April 2000.

[Chaumette 2006] F. Chaumette and S. Hutchinson. *Visual Servo Control, Part I: Basic Approaches.* IEEE Robotics and Automation Magazine, vol. 13, no. 4, pages 82–90, December 2006.

[Chen 2009] Z. Chen and S.T. Birchfield. *Qualitative vision-based path following.* IEEE Trans. on Robotics, vol. 25, no. 3, pages 749–754, 2009.

[Clemente 2007] L.A. Clemente, A.J. Davison, I.D. Reid, J. Neira and J.D. Tardós. *Mapping large loops with a single hand-held camera.* In Robotics: Science and Systems, Atlanta, Georgia, June 2007.

[Collewet 2008a] C. Collewet and E. Marchand. *Modeling complex luminance variations for target tracking.* In IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'08, Anchorage, Alaska, June 2008.

[Collewet 2008b] C. Collewet, E. Marchand and F. Chaumette. *Visual servoing set free from image processing.* In IEEE Int. Conf. on Robotics and Automation, ICRA'08, Pasadena, CA, May 2008.

[Collignon 1995] A. Collignon, D. Vandermeulen, P. Suetens and G. Marchal. *3D Multi-Modality Medical Image Registration Using Feature Space Clustering.* In Int. Conf. on Computer Vision, Virtual Reality and Robotics in Medicine, CVRMed '95, pages 195–204, London, UK, 1995.

[Comaniciu 2000] D. Comaniciu, V. Ramesh and P. Meer. *Real-Time Tracking of Non-Rigid Objects using Mean Shift.* In IEEE Int. Conf. on Computer Vision and Pattern Recognition, volume 2, pages 142–149, 2000.

[Comaniciu 2003] D. Comaniciu, V. Ramesh and P. Meer. *Kernel-based object tracking.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 25, no. 5, pages 564–577, May 2003.

[Comport 2005] A.I. Comport, D. Kragic, E. Marchand and F. Chaumette. *Robust Real-Time Visual Tracking: Comparison, Theoretical Analysis and Performance Evaluation.* In IEEE Int. Conf. on Robotics and Automation, ICRA'05, pages 2852–2857, Barcelona, Spain, April 2005.

[Comport 2006] A.I. Comport, E. Marchand, M. Pressigout and F. Chaumette. *Real-time markerless tracking for augmented reality: the virtual visual servoing framework.* IEEE Trans. on Visualization and Computer Graphics, vol. 12, no. 4, pages 615–628, July 2006.

[Comport 2007] A.I. Comport, E. Malis and P. Rives. *Accurate Quadrifocal Tracking for Robust 3D Visual Odometry.* In IEEE Int. Conf. on Robotics and Automation, ICRA'07, pages 40 –45, Roma, Italia, apr. 2007.

[Corke 2001] P.I. Corke and S. Hutchinson. *A new partitioned approach to image-based visual servo control.* IEEE Trans on Robotics and Automation, vol. 17, no. 4, pages 507–515, August 2001.

[Courbon 2009] J. Courbon, Y. Mezouar and P. Martinet. *Autonomous navigation of vehicles from a visual memory using a generic camera model.* IEEE Trans. on Intelligent Transportation Systems, vol. 10, no. 3, pages 392–402, 2009.

[Crétual 1998] A. Crétual and F. Chaumette. *Image-based visual servoing by integration of dynamic measurements.* In IEEE Int. Conf. on Robotics and Automation, ICRA'98, volume 3, pages 1994–2001, Leuven, Belgium, May 1998.

[Davison 2007] A. Davison, I. Reid, N. Molton and O. Stasse. *MonoSLAM: Real-Time Single Camera SLAM.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pages 1052–1067, 2007.

[Deguchi 2000] K. Deguchi. *A Direct Interpretation of Dynamic Images with Camera and Object Motions for Vision Guided Robot Control.* Int. Journal of Computer Vision, vol. 37, no. 1, pages 7–20, June 2000.

[Dementhon 1995] D. Dementhon and L. Davis. *Model-Based Object Pose in 25 Lines of Codes.* Int. J. of Computer Vision, vol. 15, no. 1-2, pages 123–141, 1995.

[Dionnet 2007] F. Dionnet and E. Marchand. *Robust stereo tracking for space robotic applications.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07, pages 3373–3378, San Diego, CA, October 2007.

[Diosi 2007] A. Diosi, A. Remazeilles, S. Segvic and F. Chaumette. *Outdoor Visual Path Following Experiments.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07, pages 4265–4270, San Diego, CA, October 2007.

[Dowson 2006] N.D.H. Dowson and R. Bowden. *A Unifying Framework for Mutual Information Methods for Use in Non-linear Optimisation.* In European Conference on Computer Vision, ECCV'06, volume 1, pages 365–378, June 2006.

[Dowson 2008] N. Dowson and R. Bowden. *Mutual Information for Lucas-Kanade Tracking (MILK): An Inverse Compositional Formulation.* In IEEE Trans. on Pattern Analysis and Machine Intelligence, volume 30, pages 180–185, January 2008.

[Drummond 2002] T. Drummond and R. Cipolla. *Real-Time Visual Tracking of Complex Structures.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pages 932–946, July 2002.

[Faugeras 1988] O. Faugeras and F. Lustman. *Motion and structure from motion in a piecewise planar environment.* Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 2, no. 3, pages 485–508, 1988.

[Frese 2006] U. Frese and L. Schröder. *Closing a million-landmarks loop.* In IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS'06, pages 5032–5039, Beijing, China, October 2006.

[Gans 2003] N. R. Gans, S. Hutchinson and P. Corke. *Performance Tests for Visual Servo Control Systems, with Application to Partitioned Approaches to Visual Servo Control.* Int. Journal of Robotics Research, vol. 22, no. 10-11, pages 955–984, 2003.

[Goodall 1991] C. Goodall. *Procrustes Methods in the Statistical Analysis of Shape.* Journal of the Royal Statistical Society. Series B (Methodological), vol. 53, no. 2, pages 285–339, 1991.

[Hager 1998a] G. Hager and P. Belhumeur. *Efficient Region Tracking With Parametric Models of Geometry and Illumination.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 20, no. 10, pages 1025–1039, October 1998.

[Hager 1998b] G. Hager and K. Toyama. *The XVision System: A General-Purpose Substrate for Portable Real-Time Vision Applications.* Computer Vision and Image Understanding, vol. 69, no. 1, pages 23–37, January 1998.

[Harris 1988] C. Harris and M. Stephens. *A combined corner and edge detector.* In Alvey Conference, pages 147–151, Manchester, 1988.

[Hartley 2001] R. Hartley and A. Zisserman. Multiple view geometry in computer vision. Cambridge University Press, 2001.

[Hashimoto 1993] K. Hashimoto, editeur. Visual servoing : Real time control of robot manipulators based on visual sensory feedback. World Scientific Series in Robotics and Automated Systems, Vol 7, World Scientific Press, Singapor, 1993.

[Horn 1981] B.K.P. Horn and B.G. Schunck. *Determining Optical Flow.* Artificial Intelligence, vol. 17, no. 1-3, pages 185–203, August 1981.

[Hutchinson 1996] S. Hutchinson, G. Hager and P. Corke. *A tutorial on Visual Servo Control.* IEEE Trans. on Robotics and Automation, vol. 12, no. 5, pages 651–670, October 1996.

[Irani 1992] M. Irani, B. Rousso and S. Peleg. *Detecting and Tracking Multiple Moving Objects Using Temporal Integration.* In ECCV'92, pages 282–287, 1992.

[Irani 1998] M. Irani and P. Anandan. *Robust multi-sensor image alignment.* In IEEE Int. Conf. on Computer Vision, ICCV'98, pages 959–966, Bombay, India, 1998.

[Jurie 2001] F. Jurie and M. Dhome. *Real Time 3D Template Matching.* In Int. Conf. on Computer Vision and Pattern Recognition, volume 1, pages 791–796, Hawai, December 2001.

[Kallem 2007] V. Kallem, M. Dewan, J.P. Swensen, G.D. Hager and N.J. Cowan. *Kernel-Based Visual Servoing.* In IEEE/RSJ Int. Conf. on Intelligent Robots and System, IROS'07, pages 1975–1980, San Diego, USA, October 2007.

[Kyrki 2005] V. Kyrki and D. Kragic. *Integration of Model-based and Model-free Cues for Visual Object Tracking in 3D.* In IEEE Int. Conf. on Robotics and Automation, ICRA'05, pages 1566–1572, Barcelona, Spain, April 2005.

[Lapresté 2004] J.T. Lapresté and Y. Mezouar. *A Hessian approach to visual servoing.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'04, Sendai, Japan, 2004.

[Lepetit 2006] V. Lepetit and P. Fua. *Keypoint Recognition Using Randomized Trees.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 28, no. 9, pages 1465–1479, September 2006.

[Li 2004] P. Li and F. Chaumette. *Image Cues Fusion for Contour Tracking Based on Particle Filter.* In Int. Workshop on articulated motion and deformable objects, AMDO'04, LNCS, Palma de Mallorca, Spain, September 2004.

[Lieberknecht 2009] S. Lieberknecht, S. Benhimane, P. Georg Meier and N. Navab. *A dataset and evaluation methodology for template-based tracking algorithms.* In IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'09, pages 145–151, Orlanda, USA, October 2009.

[Lim 1988] H.S. Lim and T. Binford. *Curved Surface Reconstruction Using Stereo Correspondence.* In Image Understanding Workshop, pages 809–819, 1988.

[Lowe 2004] D. Lowe. *Distinctive image features from scale-invariant keypoints.* Int. Journal of Computer Vision, vol. 60, no. 2, pages 91–110, 2004.

[Lu 2000] C.P. Lu, G.D. Hager and E. Mjolsness. *Fast and Globally Convergent Pose Estimation from Video Images.* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 22, no. 6, pages 610–622, June 2000.

[Lucas 1981] B.D. Lucas and T. Kanade. *An Iterative Image Registration Technique with an Application to Stereo Vision.* In Int. Joint Conf. on Artificial Intelligence, IJCAI'81, pages 674–679, 1981.

[Ma 2004] Y. Ma, S. Soatto, J. Košecká and S. Sastry. An invitation to 3-d vision. Springer, 2004.

[Maes 1997] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal and P. Suetens. *Multimodality image registration by maximization of mutual information.* IEEE Trans. on Medical Imaging, vol. 16, no. 2, pages 187–198, 1997.

[Malis 2004] E. Malis. *Improving vision-based control using efficient second-order minimization techniques.* In IEEE Int. Conf. on Robotics and Automation, ICRA'04, volume 2, pages 1843–1848, New Orleans, April 2004.

[Malis 2006] E. Malis and E. Marchand. *Experiments with robust estimation techniques in real-time robot vision.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06, pages 223–228, Beijing, China, October 2006.

[Marchand 1999] E. Marchand. *ViSP: A Software Environment for Eye-in-Hand Visual Servoing.* In IEEE Int. Conf. on Robotics and Automation, ICRA'99, volume 4, pages 3224–3229, Detroit, Michigan, Mai 1999.

[Marchand 2002] E. Marchand and F. Chaumette. *Virtual Visual Servoing: a framework for real-time augmented reality.* In G. Drettakis and H.-P. Seidel, editeurs, EUROGRAPHICS'02 Conf. Proceeding, volume 21(3) of *Computer Graphics Forum*, pages 289–298, Saarebrücken, Germany, September 2002.

[Marchand 2005a] E. Marchand and F. Chaumette. *Feature tracking for visual servoing purposes.* Robotics and Autonomous Systems, vol. 52, no. 1, pages 53–70, June 2005. special issue on "Advances in Robot Vision", D. Kragic, H. Christensen (Eds.).

[Marchand 2005b] E. Marchand, F. Spindler and F. Chaumette. *ViSP for visual servoing: a generic software platform with a wide class of robot control skills.* IEEE Robotics and Automation Magazine, vol. 12, no. 4, pages 40–52, December 2005. Special Issue on "Software Packages for Vision-Based Control of Motion", P. Oh, D. Burschka (Eds.).

[Masson 2003] L. Masson, F. Jurie and M. Dhome. *Contour/Texture Approach for Visual Tracking.* In 13th Scandinavian Conf. on Image Analysis, SCIA 2003, volume 2749 of *Lecture Notes in Computer Science*, pages 661–668. Springer, 2003.

[Matsumoto 1996] Y. Matsumoto, M. Inaba and H. Inoue. *Visual navigation using view-sequenced route representation.* In IEEE Int. Conf. on Robotics and Automation, ICRA'96, volume 1, pages 83 –88 vol.1, apr. 1996.

[Mei 2006] C. Mei, S. Benhimane, E. Malis and P. Rives. *Homography-based Tracking for Central Catadioptric Cameras.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06, Beijing, China, October 2006.

[Mezouar 2001] Y. Mezouar and F. Chaumette. *Path Planning for Robust Image-based Visual Servoing.* Rapport technique 1377, IRISA, January 2001.

[Nadeau 2010] C. Nadeau and A. Krupa. *A multi-plane approach for ultrasound visual servoing: application to a registration task.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'10, Taipei, Taiwan, October 2010.

[Nayar 1996] S.K. Nayar, S.A. Nene and H. Murase. *Subspace methods for robot vision.* IEEE Trans. on Robotics, vol. 12, no. 5, pages 750 – 758, October 1996.

[Odobez 1995] J.-M. Odobez and P. Bouthemy. *Robust multiresolution estimation of parametric motion models.* Journal of Visual Communication and Image Representation, vol. 6, no. 4, pages 348–365, December 1995.

[Ozuysal 2007] M. Ozuysal, P. Fua and V. Lepetit. *Fast Keypoint Recognition in Ten Lines of Code.* In IEEE Int. Conf. on Computer Vision and Pattern Recognition, 2007.

[Panin 2008] G. Panin and A. Knoll. *Mutual Information-Based 3D Object Tracking.* Int. Journal of Computer Vision, vol. 78, no. 1, pages 107–118, 2008.

[Phong 1975] B.T. Phong. *Illumination for Computer Generated Pictures.* Communication of the ACM, vol. 18, no. 6, pages 311–317, June 1975.

[Pluim 1999] J.P.W. Pluim, J.B.A. Maintz and M.A. Viergever. *Mutual information matching and interpolation artefacts.* In K.M. Hanson, editeur, SPIE Medical Imaging, volume 3661, pages 56–65. SPIE Press, 1999.

[Press 1992] W. Press, S. Teukolsky, W. Vetterling and B. Flannery. Numerical recipes in C. Cambridge University Press, Cambridge, UK, 2nd édition, 1992.

[Pressigout 2005] M. Pressigout and E. Marchand. *Real-time planar structure tracking for visual servoing: a contour and texture approach.* In IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'05, volume 2, pages 1701–1706, Edmonton, Canada, aug 2005.

[Pressigout 2007] M. Pressigout and E. Marchand. *Real-Time Hybrid Tracking using Edge and Texture Information.* Int. Journal of Robotics Research, IJRR, vol. 26, no. 7, pages 689–713, July 2007.

[Remazeilles 2007] A. Remazeilles and F. Chaumette. *Image-based robot navigation from an image memory.* Robotics and Autonomous Systems, vol. 55, no. 4, pages 345–356, April 2007.

[Rényi 1956] A. Rényi. *On Conditional Probability Spaces Generated by a Dimensionally Ordered Set of Measures.* Theory of Probability and its Applications, vol. 1, no. 1, pages 55–64, 1956.

[Royer 2007] E. Royer, M. Lhuillier, M. Dhome and J.M. Lavest. *Monocular vision for mobile robot localization and autonomous navigation.* International Journal of Computer Vision, vol. 74, no. 3, pages 237–260, 2007.

[Scott 1979] David W. Scott. *On optimal and data-based histograms.* Biometrika, vol. 66, no. 3, pages 605–610, December 1979.

[Se 2002] S. Se, D. Lowe and J. Little. *Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks.* The Int. Journal of Robotics Research, vol. 21, no. 8, page 735, 2002.

[Segvic 2009] S. Segvic, A. Remazeilles, A. Diosi and F. Chaumette. *A mapping and localization framework for scalable appearance-based navigation.* Computer Vision and Image Understanding, vol. 113, no. 2, pages 172–187, February 2009.

[Shannon 1948] C. E. Shannon. *A mathematical theory of communication.* Bell system technical journal, vol. 27, 1948.

[Shi 1994] J. Shi and C. Tomasi. *Good Features to Track.* In IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'94, pages 593–600, Seattle, Washington, June 1994.

[Shum 2000] H.-Y. Shum and R. Szeliski. *Systems and Experiment Paper: Construction of Panoramic Image Mosaics with Global and Local Alignment.* International Journal of Computer Vision, vol. 36, no. 2, pages 101–130, February 2000.

[Silveira 2007] G. Silveira and E. Malis. *Real-time visual tracking under arbitrary illumination changes.* In IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'07, Minneapolis, USA, June 2007.

[Simon 2000] G. Simon, A. Fitzgibbon and A. Zisserman. *Markerless Tracking using Planar Structures in the Scene.* In IEEE/ACM Int. Symp. on Augmented Reality, pages 120–128, Munich, Germany, October 2000.

[Stewart 1999] C.-V. Stewart. *Robust parameter estimation in computer vision.* SIAM Review, vol. 41, no. 3, pages 513–537, September 1999.

[Sturges 1926] H. A. Sturges. *The choice of a class interval.* American Statistical Association, vol. 21, pages 65–66, 1926.

[Sundareswaran 1998] V. Sundareswaran and R. Behringer. *Visual Servoing-based Augmented Reality.* In IEEE Int. Workshop on Augmented Reality, San Francisco, November 1998.

[Tahri 2005] O. Tahri and F. Chaumette. *Point-based and region-based image moments for visual servoing of planar objects.* IEEE Trans. on Robotics, vol. 21, no. 6, pages 1116–1127, 2005.

[Thévenaz 2000] P. Thévenaz and M. Unser. *Optimization of Mutual Information for Multiresolution Image Registration.* IEEE trans. on Image Processing, vol. 9, no. 12, pages 2083–2099, 2000.

[Tomasi 1991] C. Tomasi and T. Kanade. *Detection and Tracking of Point Features.* Rapport technique CMU-CS-91-132, Carnegie Mellon University Technical Report, April 1991.

[Tsao 2003] J. Tsao. *Interpolation artifacts in multimodality image registration based on maximization of mutual information.* IEEE Trans. on Medical Imaging, vol. 22, no. 7, pages 854–864, July 2003.

[Vacchetti 2004] L. Vacchetti, V. Lepetit and P. Fua. *Combining Edge and Texture Information for Real-Time Accurate 3D Camera Tracking.* In ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'2004, volume 2, pages 48–57, Arlington, Va, November 2004.

[Vargas 2005] M. Vargas and E. Malis. *Visual servoing based on an analytical homography decomposition.* In IEEE Conf. on Decision and Control and European Control Conf, pages 5379–5384, Sevilla, Spain, December 2005.

[Vincze 2001] M. Vincze. *Robust Tracking of Ellipses at Frame Rate.* Pattern Recognition, vol. 34, no. 2, pages 487 – 498, February 2001.

[Viola 1995] P. Viola and III W. M. Wells. *Alignment by maximization of mutual information.* In ICCV '95: the Fifth Int. Conf. on Computer Vision, page 16, Washington, DC, USA, 1995. IEEE Computer Society.

[Viola 1997] P. Viola and W. Wells. *Alignment by maximization of mutual information.* Int. Journal of Computer Vision, vol. 24, no. 2, pages 137–154, 1997.

[Wilson 1996] W. Wilson, C. Hulls and G. Bell. *Relative end-effector control using cartesian position-based visual servoing.* IEEE Trans. on Robotics and Automation, vol. 12, no. 5, pages 684–696, October 1996.

Abstract

In this thesis, we address the visual tracking and visual servoing problems, that are crucial in the robot vision domain. With the expansion of potential applications (robot control, augmented reality, ...), the robustness and accuracy of the algorithms become major issues. The classical tracking and servoing methods remain mainly built on the observation of particular geometrical features in the image. Since this solution does not take advantage of the full information provided by the image, new techniques called direct approaches have been proposed. Nevertheless, the existing direct methods are most of the time built directly on the pixel intensities of the images that are sensitive to appearance variations such as when the illumination conditions are changing. As a result, their robustness is limited to nominal conditions.

To overcome this problem, we propose a solution that is no longer built directly on the intensities but on the information contained in the images. An approach is proposed to create a unified solution, practical for both the tracking and servoing problems. Several tracking experiments validate the robustness and accuracy of the proposed method compared to the existing ones. The illustrated applications are various with localization, face tracking, mosaicing and augmented reality. Similarly, the mutual information-based visual servoing approach is validated though many experiments on a six dof gantry robot and also on a non-holonomic autonomous vehicle.

Keywords : Visual tracking, visual servoing, mutual information, entropy, computer vision, image registration.

Résumé

Dans cette thèse, nous traitons les problèmes d'asservissement et de suivi visuel, qui sont essentiels dans le domaine de la vision robotique. Leur robustesse ainsi que leur précision deviennent des enjeux majeurs. Les techniques classiques sont principalement basées sur l'observation de primitives géométriques dans l'image. Ces primitives ne prennent néanmoins pas compte de toute l'information présente dans les images. C'est pour cette raison que de nouvelles approches, dites approches directes, ont vu le jour. Un inconvénient des méthodes directes actuelles vient du fait qu'elles sont centrées sur l'observation des intensités lumineuses des images qui sont fortement sensibles aux changements d'apparence qui peuvent survenir, par exemple, lors de modification de l'illumination ou occultation. Ceci a pour effet de rendre l'application de ces techniques limitée à des conditions nominales d'utilisation.

Pour régler ce problème, nous proposons une solution qui n'est plus directement basée sur les intensités lumineuses mais sur l'information contenue dans les images. Nous montrons que la maximisation de cette information permet de créer une solution unifiée pour résoudre des tâches de suivi et d'asservissement visuel. De nombreuses expériences de suivi valident la robustesse et la précision de la technique proposée dans des applications variées en passant par le suivi de visages, la localisation, la construction de mosaïques et la réalité augmentée. La méthode d'asservissement visuel reposant sur l'information mutuelle est également validée à l'aide d'une plateforme contenant un robot cartésien à six degrés de liberté ainsi qu'un véhicule autonome non-holonome.

Most-clefs : Suivi visuel, asservissement visuel, information mutuelle, entropie, vision par ordinateur, recalage d'images.