Pose Estimation for Augmented Reality: A Hands-On Survey

Eric Marchand, Hideaki Uchiyama, Fabien Spindler

Pose estimation for augmented reality: a hands-on survey

Eric Marchand, Hideaki Uchiyama and Fabien Spindler

Abstract—Augmented reality (AR) allows to seamlessly insert virtual objects in an image sequence. In order to accomplish this goal, it is important that synthetic elements are rendered and aligned in the scene in an accurate and visually acceptable way. The solution of this problem can be related to a pose estimation or, equivalently, a camera localization process. This paper aims at presenting a brief but almost self-contented introduction to the most important approaches dedicated to vision-based camera localization along with a survey of several extension proposed in the recent years. For most of the presented approaches, we also provide links to code of short examples. This should allow readers to easily bridge the gap between theoretical aspects and practical implementations.

Index Terms—Survey, augmented reality, vision-based camera localization, pose estimation, PnP, SLAM, motion estimation, homography, keypoint matching, code examples.

1 Introduction

Augmented reality (AR) allows to seamlessly insert virtual objects in an image sequence. A widely acknowledged definition of augmented reality is due to Azuma in the first survey dedicated to the subject [7]. An AR system should *combine real and virtual objects, be interactive in real time, register real and virtual objects*. It has to be noted that this definition does not focus on specific technologies for localization and visualization. Back in 1997, registration was considered as *"one of the most basic problems currently limiting augmented reality [7]"*.

Pose estimation: a "basic problem" for augmented reality.

AR has been intrinsically a multidisciplinary and old research area. It is clear that real and virtual world registration issues received a large amount of interest. From a broader point of view, this is a motion tracking issue. To achieve this task, many sensors have been considered: mechanical devices, ultrasonic devices, magnetic sensors, inertial devices, GPS, compass, and obviously, optical sensors [146]. To paraphrase [146], there was *no silver bullet* to solve this problem but vision-based techniques rapidly emerged.

Indeed, with respect to other sensors, a camera combined with a display is an appealing configuration. As pointed out in [9], such a setup provides vision-based feedback that allows to effectively close the loop between the localization process and the display. This also reduces the need for heavy calibration procedure. Nevertheless, when Azuma's survey [7] was published, only few vision-based techniques meeting his definition existed.

Until the early 2000s, almost all the vision-based registration techniques relied on markers. Then various markerless approaches quickly emerged in the literature. On one hand, markerless model-based tracking techniques improve clearly (but are in line with) marker-based methods. On the other hand, with the ability to easily match keypoints like SIFT, and the perfect knowledge of multi-view geometry, new approaches based on an image model and on the estimation of the displacement of the camera [122] arose. Finally, the late 2000s saw the introduction of keyframe-based Simultaneous Localization and Mapping (SLAM) [57] that, as a sequel of structure from motion approaches (widely used in off-line compositing for the movie industry), allows to get rid of a model of the scene.

Although vision-based registration is still a difficult problem, mature solutions may now be proposed to the end-users and real-world or industrial applications can be foreseen (if not already seen). Meanwhile, many open source software libraries (OpenCV, ViSP, Vuforia,...) and commercial SDK (Metaio (now with Apple), Wikitude, AugmentedPro, Diotasoft,...) have been released providing developers with easy-to-use interfaces and efficient registration processes. It therefore allows fast prototyping of AR systems.

Rationale.

Unfortunately, using such libraries, end-users may widely consider the underlying technologies and methodological aspects as black boxes. Our goal is then to present, in the reminder of the paper, a brief but almost self-contained introduction to the most important approaches dedicated to camera localization along with a survey of the extensions that have been proposed in the recent years. We also try to link these methodological concepts to the main libraries and SDK available on the market.

The aim of this paper is then to provide researchers and practitioners with an almost comprehensive and consolidated introduction to effective tools for facilitating research in augmented reality. It is also dedicated to academics involved in teaching augmented reality at the undergraduate and graduate level. For most of the presented approaches, we also provide links to code of short examples. This should allow readers to easily bridge the gap between theoretical aspects and practice. These examples have been written using both OpenCV and the ViSP library [79] developed at Inria.

- *E. Marchand is with Université de Rennes 1, IRISA, Inria Rennes-Bretagne Atlantique, Rennes, France .*
  E-mail: Eric.Marchand@irisa.fr
- *H. Uchiyama is with Kyushu University, Japan*
- *F. Spindler is with Inria Rennes-Bretagne Atlantique, Rennes, France*

Choices have to be made.

A comprehensive description of all the existing vision-based localization techniques used in AR is, at least in a journal paper, out of reach and choices have to be made. For example, we disregard Bayesian frameworks (Extended Kalman Filter). Although such methods were widely used in the early 2000s, it appears that EKF is less and less used nowadays for the profit of deterministic approaches (to mitigate this assertion, it is acknowledged that they are still useful when considering sensor fusion). Not considering display technologies (e.g., optical see-through HMD), we also disregard eyes/head/display calibration issues. As pointed out in [146], many other sensors exist and can be jointly used with cameras. We acknowledge that this provides robustness to the localization process. Nevertheless, as stated, we clearly focus in this paper, only on the image-based pose estimation process.

Related work.

In the past, two surveys related to AR (in general) have been published in 1997 [7] and 2001 [8]. These surveys have been completed in 2008 by an analysis of 10 years of publications in ISMAR [151]. Demonstrating the interest for vision-based localization, it appears that more than 20% of the papers are related to "*tracking*" and then to vision-based registration (and they are also among the most cited papers). In [146] the use of other sensors and hybrid systems is explored. Dealing more precisely with 3D tracking, a short monograph was proposed in [65].

To help the students, engineers, or researchers pursue further research and development in this very active research area, we explain and discuss the various classes of approaches that have been considered in the literature and that we found important for vision-based AR. We hope this article will be accessible and interesting to experts and students alike.

2 Overview of the Problem

The goal of augmented reality is to insert virtual information in the real world providing the end-user with additional knowledge about the scene. The added information, usually virtual objects, must be precisely aligned with the real world. Figure 1 shows how these two worlds can be combined into a single and coherent image.

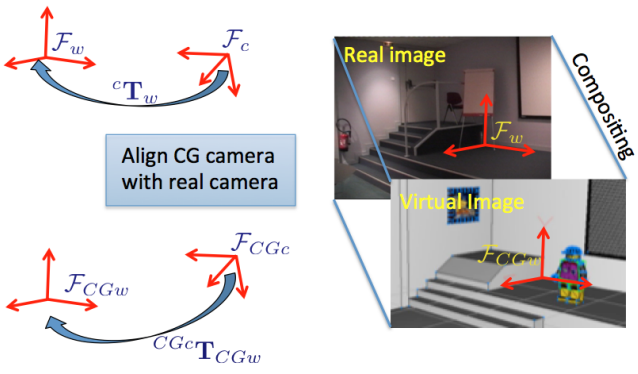

Fig. 1. AR Principle and considered coordinate systems: to achieve a coherent compositiong, computer graphics (CG) camera and real one should be located at the very same position and have the same parameters.

From the real world side, we have the scene and the camera. Let us denote $\mathcal{F}_c$ the camera frame, $\mathcal{F}_w$ the scene frame (or world frame). On the virtual side, we have a virtual world with various virtual objects whose position are expressed in the virtual world frame $\mathcal{F}_{CGw}$ (computer graphics (CG) frame). To render the virtual scene, a virtual (CG) camera is added to the system. Let us denote $\mathcal{F}_{CGc}$ the virtual camera frame. For simplicity and without loss of generality, let us assume that the world frame and the virtual world are the same ($\mathcal{F}_{CGw} = \mathcal{F}_w$). To create an image of the virtual world that is consistent with the real camera current view, CG camera and real one should be located at the very same position and have the same parameters (focal, viewing angle, etc). Once the real and CG cameras are perfectly aligned, a compositing step simply provides the resulting augmented image.

Within this process, the only unknown is the real camera position in the world frame (we denote ${}^c\mathbf{T}_w$ the transformation that fully defines the position of $\mathcal{F}_w$ wrt. $\mathcal{F}_c$). Vision-based AR is thus restricted to a camera pose estimation problem. Any error in the estimation of the camera position in the world reference frame appears to the user as inconsistencies.

Pose estimation is a problem which found its origin in photogrammetry where it is known as *space resection*. A simple definition could be: "*given a set of correspondences between 3D features and their projections in the images plane, pose estimation consists in computing the position and orientation of the camera*". There are many ways to present the solutions to this inverse problem. We made the choice to divide the paper according to available data: do we have 3D models (or can we acquire them?) or do we restrict to planar scenes? The paper is then organized as follow:

- In Section 3, we chose to consider first the general case where 3D models are available or can be built on-line. We first review in Section 3.1 the solutions based on classical pose estimation methods (known as P*n*P). We then show in Section 3.2 a generalization of the previous method to handle far more complex 3D model. When 3D models are not a priori available, they can be estimated on-line thanks to Simultaneous Localization and Mapping (SLAM) techniques (see Section 3.3). Finally when 3D data can be directly measured, registration with the 3D model can be done directly in the 3D space. This is the objective of Section 3.4.
- It appears that the problem could be easily simplified when the scene is planar. This is the subject of Section4. In that case, the pose estimation could be handled as a camera motion estimation process.
- From a practical point of view, the development of actual AR applications rises the question of the features extraction and of the matching issues between image features. This issue will be discussed in Section 5.

Overall, whatever the method chosen, it will be seen that pose estimation is an optimization problem. The quality of the estimated pose is highly dependent on the quality of the measurements. We therefore also introduce in Section 3.1.3 robust estimation process able to deal with spurious data (outliers) which is fundamental for real-life applications.

3 Pose Estimation Relying on a 3D Model

In this section we assume that a 3D model of the scene is available or can be estimated on-line. As stated in the previous section, the pose should be estimated knowing the correspondences between

2D measurements in the images and 3D features of the model. It is first necessary to properly state the problem. We will consider here that these features are 3D points and their 2D projections (as a pixel) in the image.

Let us denote $\mathcal{F}_c$ the camera frame and ${}^c\mathbf{T}_w$ the transformation that fully defines the position of $\mathcal{F}_w$ wrt. $\mathcal{F}_c$ (see Figure 2). ${}^c\mathbf{T}_w$, is a homogeneous matrix defined such that:

$$ {}^c\mathbf{T}_w = \begin{pmatrix} {}^c\mathbf{R}_w & {}^c\mathbf{t}_w \\ \mathbf{0}_{3\times 1} & 1 \end{pmatrix} \tag{1} $$

where ${}^c\mathbf{R}_w$ and ${}^c\mathbf{t}_w$ are the rotation matrix and translation vector that define the position of the camera in the world frame (note that ${}^c\mathbf{R}_w$ being a rotation matrix, it should respect the orthogonality constraints).

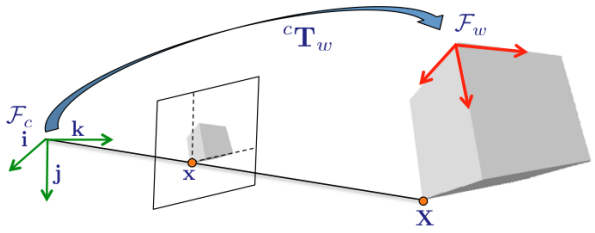

Fig. 2. Rigid transformation ${}^c\mathbf{T}_w$ between world frame $\mathcal{F}_w$ and camera frame $\mathcal{F}_c$ and perspective projection

The perspective projection $\bar{\mathbf{x}} = (u, v, 1)^\top$ of a point ${}^w\mathbf{X} = ({}^wX, {}^wY, {}^wZ, 1)^\top$ will be given by (see Figure 2):

$$ \bar{\mathbf{x}} = \mathbf{K}\,\Pi\,{}^c\mathbf{T}_w\,{}^w\mathbf{X} \tag{2} $$

where $\bar{\mathbf{x}}$ are the coordinates, expressed in pixel, of the point in the image; $\mathbf{K}$ is the camera intrinsic parameters matrix and is defined by:

$$ \mathbf{K} = \begin{pmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} $$

where $(u_0, v_0, 1)^\top$ are the coordinates of the principal point (the intersection of the optical axes with the image plane) and p_x (resp p_y) is the ratio between the focal length of the lens f and the size of the pixel l_x: $p_x = f/l_x$ (resp, l_y being the height of a pixel, $p_y = f/l_y$). Π the projection matrix is given, in the case of a perspective projection model, by:

$$ \Pi = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} $$

The intrinsic parameters can be easily obtained through an off-line calibration step (e.g. [20], [149]). Therefore, when considering the AR problem, we shall consider image coordinates expressed in the normalized metric space $\mathbf{x} = \mathbf{K}^{-1}\bar{\mathbf{x}}$. Let us note that we consider here only a pure perspective projection model but it is clear that any model with distortion can be easily considered and handled. From now, we will always consider that the camera is calibrated and that the coordinates are expressed in the normalized space.

If we have N points ${}^w\mathbf{X}_i, i = 1..N$ whose coordinates expressed in $\mathcal{F}_w$ are given by ${}^w\mathbf{X}_i = ({}^wX_i, {}^wY_i, {}^wZ_i, 1)^\top$, the projection $\mathbf{x}_i = (x_i, y_i, 1)^\top$ of these points in the image plane is then given by:

$$ \mathbf{x}_i = \Pi\,{}^c\mathbf{T}_w\,{}^w\mathbf{X}_i. \tag{3} $$

Knowing 2D-3D point correspondences, $\mathbf{x}_i$ and ${}^w\mathbf{X}_i$, pose estimation consists in solving the system given by the set of equations (3) for ${}^c\mathbf{T}_w$. This is an inverse problem that is known as the Perspective from N Points problem or PnP (Perspective-n-point).

3.1 Pose estimation from a known 3D model

In this paragraph, we review methods allowing to solve the set of equations (3) for the pose ${}^c\mathbf{T}_w$. Among various solutions, we will explain more deeply two classical algorithms widely considered in augmented reality: one method that does not require any initialization of the pose (Direct Linear Transform) and a method based on a gradient approach that needs an initial pose but which can be consider as the "gold standard" solution [48]. We will also discuss more complex, but also more efficient, solutions to the pose estimation issue. Optimization procedure in the presence of spurious data (outliers) is also considered. In each case, a comprehensive description of each methods will be given.

3.1.1 P3P: solving pose estimation with the smallest subset of correspondences

P3P is an important and old problem for which many solutions have been proposed. Theoretically, since the pose can be represented by six independent parameters, three points should be sufficient to solve this problem.

Most of the P3P approaches rely on a 2 steps solution. First an estimation of the unknown depth ${}^cZ_i$ of each point (in the camera frame) is done thanks to constraints (law of cosines) given by the triangle $C\mathbf{X}_i\mathbf{X}_j$ for which the distance between $\mathbf{X}_i$ and $\mathbf{X}_j$ and the angle between the two directions $C\mathbf{X}_i$ and $C\mathbf{X}_j$ are known and measured. The estimation of the points depth is usually done by solving a fourth order polynomial equation [39] [105] [41] [5]. Once the three points coordinates are known in the camera frame, the second step consists in estimating the rigid transformation ${}^c\mathbf{T}_w$ that maps the coordinates expressed in the camera frame to the coordinates expressed in the world frame (3D-3D registration, see Section 3.4). The rotation represented by quaternions can be obtained using a close form solution [49]. Alternatively least squares solution that use the Singular Value Decomposition (SVD) [5] can also be considered. Since a fourth order polynomial equation as to be solved, the problem features up to four possible solutions. It is then necessary to have at least a fourth point to disambiguate the obtained results [39] [48].

More recently, Kneip et al. [62] propose a novel closed-form solution that directly computes the rigid transformation between the camera and world frames ${}^c\mathbf{T}_w$. This is made possible by introducing first a new intermediate camera frame centered in C whose x axes is aligned with the direction of the first point $\mathbf{X}_1$ and secondly a new world frame centered in $\mathbf{X}_1$ and whose x axes is aligned with the direction of the first point $\mathbf{X}_2$. Their relative position and orientation can be represented using only two parameters. These parameters can then be computed by solving a fourth order polynomial equation. A final substitution allows computing ${}^c\mathbf{T}_w$. The proposed algorithm is much faster than the other solutions since it avoids the estimation of the 3D points depth in the camera frame and the estimation of the 3D-3D registration step. Kneip's P3P implementation is available in OpenGV [59].

Although P3P is a well-known solution to the pose estimation problem, other PnP approaches that use more points ($n > 3$) were usually preferred. Indeed pose accuracy usually increases with the number of points. Nevertheless within an outliers rejection process

such as RANSAC, being fast to compute and requiring only three points correspondences, fast P3P such as [59] is the solution to chose (see Section 3.1.3). P3P is also an interesting solution to bootstrap a non-linear optimization process that minimizes the reprojection error as will be seen in Section 3.1.2.

3.1.2 PnP: pose estimation from N point correspondences

P*n*P considered an over-constrained and generic solution to the pose estimation problem from 2D-3D point correspondences. Here again, as for the P3P, one can consider multi-stage methods that estimate the coordinates of the points [105] or of virtual points [67] in the camera frame and then achieve a 3D-3D registration process [105]. On the other side, direct or one stage minimization approaches have been proposed.

Among the former, [105] extended their P3P algorithm to P4P, P5P and finally to P*n*P. In the EP*n*P approach [67] the 3D point coordinates are expressed as a weighted sum of four virtual control points. The pose problem is then reduced to the estimation of the coordinates of these control points in the camera frame. The main advantage of this latter approach is its reduced computational complexity, which is linear wrt. the number of points.

Within the latter one step approaches, the Direct Linear Transform (DLT) is certainly the oldest one [48], [129]. Although not very accurate, this solution and its sequels have historically widely been considered in AR application. P*n*P is intrinsically a non-linear problem; nevertheless a solution relying on the resolution of a linear system can be considered. It consists in solving the homogeneous linear system built from equations (3), for the 12 parameters of the matrix ${}^c\mathbf{T}_w$. Indeed, considering that the homogeneous matrix to be estimated is defined by:

$$ {}^c\mathbf{T}_w = \begin{pmatrix} \mathbf{r}_1 & t_x \\ \mathbf{r}_2 & t_y \\ \mathbf{r}_3 & t_z \\ \mathbf{0}_{3\times 1} & 1 \end{pmatrix} $$

where $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ are the rows of the rotation matrix ${}^c\mathbf{R}_w$ and ${}^c\mathbf{t}_w = (t_x, t_y, t_z)$. Developing (3) yields to solve the system:

$$ \mathbf{A}\mathbf{h} = \begin{pmatrix} \vdots \\ \mathbf{A}_i \\ \vdots \end{pmatrix} \mathbf{h} = 0 \tag{4} $$

with $\mathbf{A}_i$ given by [129]:

$$ \mathbf{A}_i = \begin{pmatrix} {}^wX_i & {}^wY_i & {}^wZ_i & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & {}^wX_i & {}^wY_i & {}^wZ_i & 1 \\ & & & -x_i{}^wX_i & -x_i{}^wY_i & -x_i{}^wZ_i & -x_i & \\ & & & -y_i{}^wX_i & -y_i{}^wY_i & -y_i{}^wZ_i & -y_i & \end{pmatrix} \tag{5} $$

and

$$ \mathbf{h} = \begin{pmatrix} \mathbf{r}_1, & t_x, & \mathbf{r}_2, & t_y, & \mathbf{r}_3, & t_z \end{pmatrix}^\top $$

is a vector representation of ${}^c\mathbf{T}_w$. The solution of this homogeneous system is the eigenvector of $\mathbf{A}$ corresponding to its minimal eigenvalue (computed through a Singular Value Decomposition of $\mathbf{A}$). An orthonormalization of the obtained rotation matrix is then necessary[1].

Obviously and unfortunately, being over-parameterized, this solution is very sensitive to noise and a solution that explicitly considers the non-linear constraints of the system should be preferred.

An alternative and very elegant solution, which takes these non-linear constraints into account, has been proposed in [28] [93]. Considering that the pose estimation problem is linear under the scaled orthographic projection model (weak perspective projection) [48] [28], Dementhon proposed to iteratively go back from the scaled orthographic projection model to the perspective one. POSIT is a standard approach used to solve the P*n*P problem. An advantage of this approach is that it does not require any initialization. It inherently enforces the non-linear constraints and is computationally cheap. A drawback is that POSIT is not directly suited for coplanar points. Nevertheless an extension of POSIT has been proposed in [93]. Its implementation is available in OpenCV [20] or in ViSP [79] and it has widely been used in AR application (see Section 3.1.4).

In our opinion, the "gold-standard" solution to the P*n*P consists in estimating the six parameters of the transformation ${}^c\mathbf{T}_w$ by minimizing the norm of the reprojection error using a non-linear minimization approach such as a Gauss-Newton of a Levenberg-Marquardt technique. Minimizing this reprojection error provides the Maximum Likelihood estimate when a Gaussian noise is assumed on measurements (ie, on point coordinates $\mathbf{x}_i$). Another advantage of this approach is that it allows easily integrating the non-linear correlations induced by the P*n*P problem and provides an optimal solution to the problem. The results corresponding to this example is shown on Figure 4. Denoting $\mathbf{q} \in se(3)$ a minimal representation of ${}^c\mathbf{T}_w$ ($\mathbf{q} = ({}^c\mathbf{t}_w, \theta\mathbf{u})^\top$ where θ and $\mathbf{u}$ are the angle and the axis of the rotation ${}^c\mathbf{R}_w$), the problem can be formulated as:

$$ \widehat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^{N} d\left(\mathbf{x}_i, \Pi {}^c\mathbf{T}_w {}^w\mathbf{X}_i\right)^2 \tag{6} $$

where $d(\mathbf{x}, \mathbf{x}')$ is the Euclidian distance between two points $\mathbf{x}$ and $\mathbf{x}'$. The solution of this problem relies on an iterative minimization process such as a Gauss-Newton method.

Solving equation (6) consists in minimizing the cost function $E(\mathbf{q}) = \|\mathbf{e}(\mathbf{q})\|$ defined by:

$$ E(\mathbf{q}) = \mathbf{e}(\mathbf{q})^\top \mathbf{e}(\mathbf{q}), \quad \text{with} \quad \mathbf{e}(\mathbf{q}) = \mathbf{x}(\mathbf{q}) - \mathbf{x} \tag{7} $$

where $\mathbf{x}(\mathbf{q}) = (..., \pi({}^c\mathbf{T}_w {}^w\mathbf{X}_i), ...)^\top$ and $\mathbf{x} = (..., \tilde{\mathbf{x}}_i, ...)^\top$ where $\tilde{\mathbf{x}}_i = (x_i, y_i)$ is a Euclidian 2D point and $\pi(\mathbf{X})$ is the projection function that project a 3D point $\mathbf{X}$ into $\tilde{\mathbf{x}}$. . The solution consists in linearizing $\mathbf{e}(\mathbf{q}) = 0$. A first order Taylor expansion of the error is given by:

$$ \mathbf{e}(\mathbf{q} + \delta\mathbf{q}) \approx \mathbf{e}(\mathbf{q}) + \mathbf{J}(\mathbf{q})\delta\mathbf{q} \tag{8} $$

where $\mathbf{J}(\mathbf{q})$ is the Jacobian of $\mathbf{e}(\mathbf{q})$ in $\mathbf{q}$. With the Gauss-Newton method the solution consists in minimizing $E(\mathbf{q} + \delta\mathbf{q})$ where:

$$ E(\mathbf{q} + \delta\mathbf{q}) = \|\mathbf{e}(\mathbf{q} + \delta\mathbf{q})\| \approx \|\mathbf{e}(\mathbf{q}) + \mathbf{J}(\mathbf{q})\delta\mathbf{q}\| \tag{9} $$

This minimization problem can be solved by an iterative least square approach (ILS), see Figure 3, and we have:

$$ \delta\mathbf{q} = -\mathbf{J}(\mathbf{q})^+ \mathbf{e}(\mathbf{q}) \tag{10} $$

1. *The source code of the DLT algorithm is proposed as a supplementary material of this paper and is available here.*

where $\mathbf{J}^+$ is the pseudo inverse[2] of the $2N \times 6$ Jacobian $\mathbf{J}$ given by [78]:

$$\mathbf{J} = \begin{pmatrix} \vdots & & & & & \\ -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1+x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & 1+y_i^2 & -x_i y_i & -x_i \\ \vdots & & & & & \end{pmatrix}. \quad (11)$$

Since we have an iterative method, the pose is updated at each iteration:

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

where $\oplus$ denotes the composition operation over se(3) obtained via the exponential map [76]. A complete derivation of this problem, including the derivation of the Jacobian, is given in [22][3].

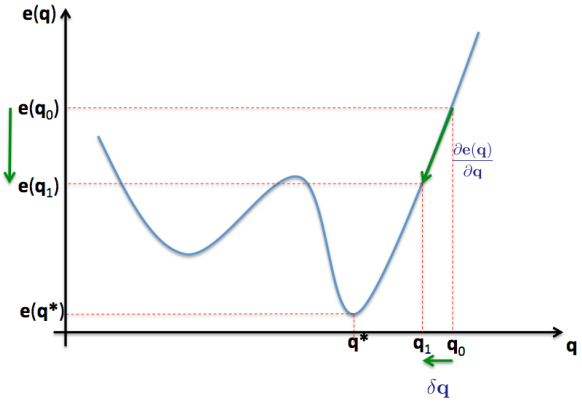

Fig. 3. Iterative minimization of the pose: overview of the non-linear least squares problem (here in 1D).

However, the algorithm requires a good initial guess ${}^c\mathbf{T}_w$ in order to converge to the globally optimal solution. If this is not the case only a local minima is attained. Olsson et al. [94] propose a Branch-and-Bound algorithm that allows retrieving a global minimum but this drastically increases the computational cost. Another iterative method have been proposed in [74] where the authors proposed to minimize an algebraic error which is faster to converge but that remains unfortunately sensitive to local minima.

When complexity is of interest (i.e., when N increases), non-iterative PnP algorithms with a linear complexity have been proposed. A first accurate $O(N)$ solution to the PnP was EPnP [67]. Later, other $O(N)$ solutions such as OPnP [150], GPnP [60], UPnP [61] were proposed and are of interest when the number of point correspondences increases.

As can be seen, many approaches have been proposed to solve the pose estimation from point correspondences. In our opinion, the choice of the "best" method widely depends on the number N of points, the noise level, the number of correspondence errors, etc. Indeed, in real life applications such as AR, pose estimation is plagued by spurious data and embedding PnP in dedicated algorithms has then to be considered. This is the purpose of Section 3.1.3. A discussion about possible choices in an AR context is proposed in Section 3.1.4. Let us finally note that the specific (and simpler) case of coplanar points will be reviewed in Section 4.

3.1.3 Dealing with spurious data

Whatever the method chosen to solve the PnP, the solution must deal with the problem of robustness so as to account for noise in real video acquisition, occlusion phenomena, changes in illumination, miss-tracking or errors in the correspondences and, more generally, for any perturbation that may be found in the video. Using a robust low-level feature extraction is certainly useful but usually not sufficient since it is not possible to model all possible perturbations.

As a solution, a robust estimation process is usually incorporated into pose estimation. Voting techniques, Random Sample Consensus (RANSAC) [39], M-Estimators [50], Least-Median of Squares (LMedS) [109] have been widely used to solve this issue. How to consider robust parameters estimation in computer vision algorithm has been reviewed in [126].

Random Sample Consensus (RANSAC).

RANSAC is an iterative method proposed in [39] to solve the P3P problem. Since then, it has been applied to many computer vision problems such as PnP, visual SLAM, homography estimation, fundamental or essential matrix estimation, etc. The goal is to divide the data in two sets: the inliers and the outliers (spurious data). We present this algorithm in the case of a PnP problem but it is worth keeping in mind that it applies to most estimation problems (especially those presented in the reminder of this paper in Section 3.3, 3.4 and 4).

Let us assume that we have a set of pairs of matched 2D-3D points (correspondences): $(\mathbf{x}_i, {}^w\mathbf{X}_i)$. Among these data let us assume that some matches are wrong. RANSAC uses the smallest set of possible correspondences and proceeds iteratively to enlarge this set with consistent data. At iteration k of the algorithm, it:

1) draws a minimal number (e.g., 3 for a P3P, 4 for a P4P) of randomly selected correspondences S_k (a *random sample*).
2) computes the pose ${}^c\widehat{\mathbf{T}}_w$ from these minimal set of point correspondences using the P3P, DLT, POSIT or EPnP (or any other approach that does not require an initialization).
3) determines the number C_k of points from the whole set of all correspondences that are consistent with the estimated parameters ${}^c\widehat{\mathbf{T}}_w$ with a predefined tolerance ε (that is for which $d(\mathbf{x}, \Pi\, {}^c\widehat{\mathbf{T}}_w {}^w\mathbf{X})^2 \leq \varepsilon$). If $C_k > C^*$ then we retain the randomly selected set of correspondences S_k as the best one (to date) : $S^* = S_k$ and $C^* = C_k$.
4) repeats steps 1 to 3

The C^* correspondences that participate to the *consensus* obtained from S^* are the inliers. The others are the outliers. A more accurate PnP approach considering all the determined inliers can then be considered to estimate the final pose. It has to be noted that the number of iterations, which ensures a probability p that at least one sample with only inliers is drawn, can be determined automatically. It is given by [39]:

$$N = \frac{\log(1-p)}{\log(1-(1-\eta)^n)}$$

where η is the probability that a correspondence is an outlier and s is the size of the sample. For the P4P problem ($n = 4$) when data

2. An alternative to the pseudo-inverse to solve this system is to consider the QR decomposition of $\mathbf{J}(\mathbf{q})$.

3. *The source code of the pose estimation using a non-linear minimisation technique is also proposed as a supplementary material of this paper and is available here*

is contaminated with 10% of outliers, 5 iterations are required to ensure that $p = 0.99$ and with 50% of outliers 72 iterations are necessary.

IRLS : using M-estimator.

M-estimators are a generalization of maximum likelihood estimation and least squares. Therefore they are well suited to detect and reject outliers in a least square or iterative least square approach. With respect to RANSAC, which aggregates a set of inliers from a minimal number of correspondences, M-estimators use as many data as possible to obtain an initial solution and then iterate to reject outliers.

M-estimators are more general than least squares because they permit the use of different minimization functions not necessarily corresponding to normally distributed data. Many functions have been proposed in the literature that allow uncertain measures to have less influence on the final result and in some cases to completely reject the measures. In other words, the objective function is modified to reduce the sensitivity to outliers. The robust optimization problem is then given by:

$$\widehat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^{N} \rho\left(d(\mathbf{x}_i, \Pi^c \mathbf{T}_w {}^w\mathbf{X}_i)\right) \tag{12}$$

where $\rho(u)$ is a M-estimator [50] that grows sub-quadratically and is monotonically non-decreasing with increasing $|u|$. Iteratively Reweighted Least Squares (IRLS) is a common method of applying the M-estimator [50], [126]. It converts the M-estimation problem into an equivalent weighted least-squares problem.

The basic idea is no longer to minimize the error $\mathbf{e}(\mathbf{q}) = \mathbf{x}(\mathbf{q}) - \mathbf{x}$ as defined in (7) but the error $\mathbf{e}(\mathbf{q}) = \mathbf{W}(\mathbf{x}(\mathbf{q}) - \mathbf{x})$ where $\mathbf{W}$ is a diagonal weighting matrix where each element of the diagonal w_i reflects the confidence in the i-th feature (when $w_i = 0$, its influence in the least square problem is null, when equal to 1, its influence is maximal). This minimization problem can be solved by an IRLS approach. Equation (10) is then replaced by:

$$\delta\mathbf{q} = -(\mathbf{W}\mathbf{J}(\mathbf{q}))^{+}\mathbf{W}\,\mathbf{e}(\mathbf{q}). \tag{13}$$

Weights are recomputed at each iteration according to the current estimate of the position $\mathbf{q}$. Many M-estimator $\rho(u)$ (Beaton Tuckey, Cauchy, Huber,...) can be considered leading to various ways to compute the confidence. A comprehensive way to compute the weights is given in [126] or in [24] using the Tukey loss function (which allows to completely rejects outliers and gives them a zero weight).

RANSAC or M-estimatiors are two classical ways to ensure robust estimation. They can be considered for pose estimation but as will be shown in the reminder of this survey, these are generic tools that allow treating the fundamental problem of the outliers. Almost all the approaches presented in this paper can take advantage of these methods that must be considered for real-life applications.

3.1.4 Example of PnP in AR applications and discussion

All the PnP approaches presented in Sections 3.1.1 and 3.1.2 can now run in real-time even when a large number of point correspondences are considered. For AR application, rather than computational efficiency (as soon as real-time requirement are met), accuracy is the key criterion in order to avoid jitter effects.

POSIT has been widely used in AR contexts with artificial landmarks such as in [30], [113] or in [18], [68], [78], [117] for pose initialization. A result of these PnP methods is reported in Figure 4. Four points are considered to compute the pose. A planar version of POSIT [93] is considered in the very first image of the sequence while a non-linear estimation technique [78] is then considered (a video is available here).

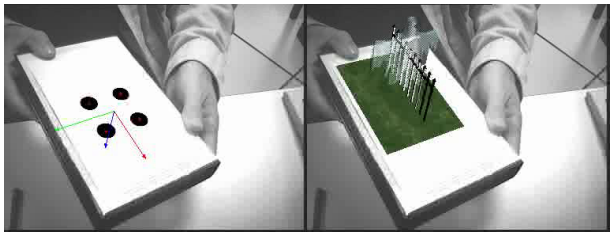

Fig. 4. Pose estimation using planar version of POSIT [93] followed by a non-linear estimation process [78] to improve the registration.

Alternatively other approaches to pose estimation can be considered. ARToolkit [54] uses an iterative search for the pose. A very efficient solution for planar target pose estimation is considered in [115] and has been used in ARToolkit+ [145].

Although markers were used in the previous examples, keypoints (see Section 5) have also been widely considered in the literature (see, for example, Figure 5). A non-linear minimization technique is for example considered in [144] [96] using SIFT and FERNS. In any case, robust process using RANSAC or IRLS is usually considered [113]. Also considering keypoints, these methods are also used for fast re-localization issue [6] [81] in environments that have been previously reconstructed using a SLAM approach (see Section 3.3).

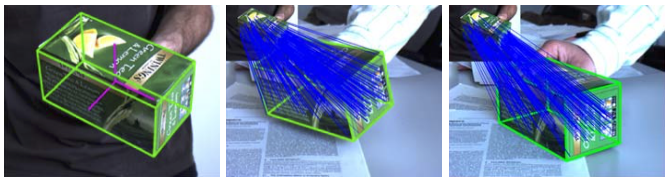

Fig. 5. Pose estimation using the EPnP algorithm [67]: reference image on the left ; projection of the model of the box after pose estimation computed using EPnP using correspondences shown by blue lines.

As can be seen, users tend to favor a PnP that does not require any initialization (such as EPnP) along with a RANSAC. An iterative estimation process based on a non-linear minimization approach improve the obtain results. Although P3P was not, to date, the most popular approach, but this tends to change. Indeed since the size of the environment increases, the need for faster algorithms (e.g., [62]) now become prevalent (especially in a SLAM context, see Section 3.3). Time computation of various PnP approaches with respect to N is reported in e.g. [150] [61].

3.2 Extension to markerless model-based tracking

Various authors have proposed different formulations of the pose estimation problem, which do not require the need of markers or keypoints matching process [23], [24], [31], [73], [99], [120], [140]. Although one can find some differences in these various solutions, the main idea is the following: as for equation (6) which is based on the distance between two points, the idea here is to

define a distance between a contour point in the image and the projected 3D line underlying the corresponding 3D model.

Assuming an estimate of the pose is known, the 3D model is first projected into the image according to that pose. Contour $\mathbf{L}(\mathbf{q})$ is sampled (black points in Figure 6) and a search is performed along the edge normal to the contour (dashed lines) to find strong gradients in the next frame. Usually the point of maximum likelihood with respect to the initial sampled point $\mathbf{x}_i$ is selected from this exploration step. It is denoted by $\mathbf{x}_i$ i in the following (white points in Figure 6).

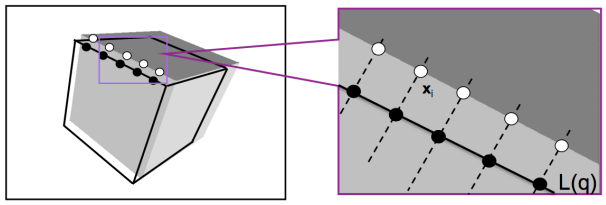

Fig. 6. Markerless model-based tracking: search for point correspondences between two frames and distance to be minimized.

A non linear optimization approach is then used to estimate the camera pose which minimizes the errors between the selected points and the projected edges [24], [31], that is:

$$\widehat{\mathbf{q}} = arg\min_{\mathbf{q}} \sum_i d_\perp(L(\mathbf{q}), \mathbf{x}_i) \tag{14}$$

where $d_\perp(L(\mathbf{q}), \mathbf{x}_i)$ is the squared distance between the point $\mathbf{x}_i$ and the projection of the contour of the model for the pose $\mathbf{q}$. This minimization process is usually handled thanks to a Gauss-Newton or a Levenberg-Marquardt minimization approach as presented in Section 3.1.2. The main difference with respect to Section 3.1.2 is that a point-to-contour distance is considered rather than a point-to-point distance. The earliest approaches that consider these markerless model based tracking algorithms mainly consider models composed with segments (see Figure 7).

Fig. 7. Markerless model-based tracking [31] [140] [24].

Weighted numerical nonlinear optimization techniques like Newton-Raphson or Levenberg-Marquardt are usually considered. To reject outliers, methods like RANSAC [18] or the use of M-Estimators such as the Tukey estimator [24], [31], [140] are common trends to make the algorithm robust to occlusions and illumination variations. But the robustness deteriorates when ambiguities between different edges occur, especially between geometrical and texture edges of the scene. One way to address this issue has been to fuse the information of edge features with information given by particular keypoints [23], [98], [104] or by other sensors [56]. Other solutions have considered multiple hypotheses for potential edge-locations in the image [99], [133], [140].

One of the drawbacks of these methods is that the 3D model is usually made of segments, which implies dealing with simple objects or manually pre-processing the CAD model. This is why more recent approaches proposed to render the 3D model (which can be arbitrarily complex) using a 3D rendering engine and a GPU [147] [99] [23]. This allows automatically managing the projection of the model and determining visible and prominent edges from the rendered scene. An advantage of these techniques is to automatically handle the hidden faces removal process and to implicitly handle self-occlusions (see Figure 8).

Fig. 8. Markerless model-based tracking [147] [100] [23]: GPU is used to render complex models and to ensure hidden faces removal.

Open source code for markerless model-based tracking exists in ViSP [24] from Inria[4], or in openTL from DLR. A commercial library was also available from Metaio (see Figure 9).

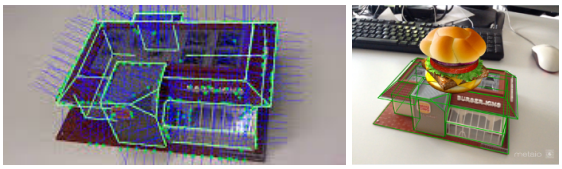

Fig. 9. Metaio model-based tracker

3.3 Pose from an a priori unknown model: Simultaneous Localization and Mapping

The previous approaches require a 3D model of the object or of the environment. Since a comprehensive or even a sparse 3D knowledge is not always easily available, the development of pose estimation methods that involve less constraining knowledge about the observed scene has been considered. The idea is then to perform the estimation of the scene structure and the camera localization within the same framework. This problem originally known as the *structure from motion* issue was handled off-line due to the high computational complexity of the solution. For real-time AR, although the theoretical problem is similar, solutions have evolved in the recent years and are now very efficient. This leads to vision-based SLAM (vision-based Simultaneous Localization And Mapping or vSLAM) that received much attention in both the robotics and AR community.

Considering monocular SLAM, two methodologies have been widely considered. The former is based on Bayesian filtering approaches. In [27], it is proposed to integrate data thanks to an Extended Kalman Filter whereas in [32] (inspired from FastSLAM) a particle filter is considered. Within these approaches, measurements are sequentially integrated within the filter, updating the probability density associated with the state of the system (the camera position, its velocity and the scene structure). All past poses being marginalized, the number of parameters to be estimated only grows with the size of the map. The latter approach is based on the minimization of reprojection errors

4. *We propose as a supplementary material of this paper (here) an example of how to deal with such a such model-based tracker. The interested reader could easily access the full source code of the tracker in ViSP [79].*

(as in Section 3.1.2). It is known as a bundle adjustment (BA) method [134] [84] [57], which had proved to be very efficient and accurate in off-line applications. In [127], it has been shown that, once the "past" poses sequence has been sparsified (choosing adequately a reduced set of keyframes), the problem becomes tractable and BA proved to be superior to filter-based SLAM.

Thus, denoting $[\mathbf{q}]_M = (\mathbf{q}_1, \ldots, \mathbf{q}_t)$ a sequence of t camera positions (keyframes) and $[{}^w\mathbf{X}]_N = ({}^w\mathbf{X}_1, \ldots, {}^w\mathbf{X}_N)$ a set of N 3D points, the goal is, as for the PnP problem to minimize the error between the observations and the reprojection of 3D points. The error to be minimized is then given by:

$$([\widehat{\mathbf{q}}]_t, \quad [\widehat{{}^w\mathbf{X}}]_N) = \arg\min_{([\mathbf{q}]_t, [{}^w\mathbf{X}]_N)} \sum_{j=1}^{t} \sum_{i=1}^{N} d\left(\mathbf{x}_{j_i}, \Pi^j \mathbf{T}_w {}^w\mathbf{X}_i\right)\Big)^2 \quad (15)$$

It is obvious that the complexity of the problem increases with the number of keyframes.

Initialization being an important issue, camera motion between a given keyframe and the current one is estimated using e.g. [91] and points are triangulated. [84] and [92] proposed to perform the BA only on a sliding window (which may lead to a camera drift) while Parallel Tracking and Mapping (PTAM) [57] considers in parallel a local BA with a tracking method that involves only a localization process as in 3.1.2 with points that have been already reconstructed (see Figure 10).

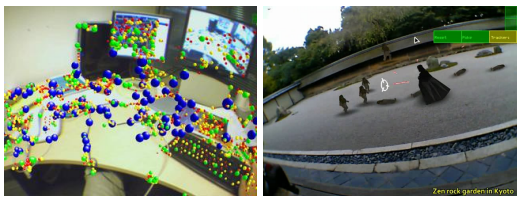

Fig. 10. Parallel Tracking and Mapping (PTAM) [57] (a video is available here)

[84] [92] and [57], have clearly demonstrated the feasibility of a deterministic SLAM system for augmented reality on a PC [57] and on mobile devices [58]. Companies such as Metaio, 13th Lab (now with Oculus) or Qualcomm provide industrial and cost effective frameworks[5].

Nevertheless, such SLAM based approaches lack absolute localization and are computationally expensive in large environments. To achieve real-time requirement and to cope with scale factor and the lack of absolute positioning issues, it has been proposed to decouple the localization and the mapping step. Mapping is handled by a full scale BA or a keyframe based BA. It is processed to fix scale factor and define the reference frame. Then, only a tracking (PnP) is performed on-line providing an absolute and reliable pose to the end-user. Such an approach has been successfully used for vehicle localization [110] and augmented reality [144] [81] [143] (see Figure 11). Another interesting approach that merges model-based tracking (Section 3.2) with SLAM has been proposed in [116] for piecewise planar scene and in [19] [131] for more complex 3D models. The approach proposed in [131] has been adopted in the Diotasoft product (see Figure 12).

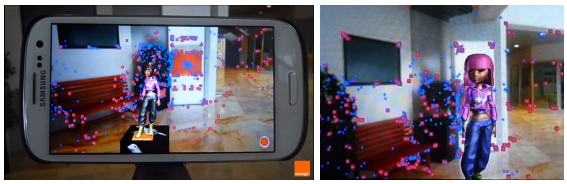

Fig. 11. AR system that considers first an off-line SLAM approach followed by an on-line PnP [81]. The reduced computational complexity allows an implementation on a smartphone (a video is available here).

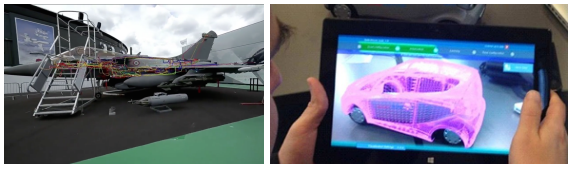

Fig. 12. Merging model-based tracking and SLAM [131] as proposed in Diotasoft tools.

In vSLAM approaches like PTAM, only few pixels contribute to the pose and structure estimation process. As in Section 4.2, dense or direct approaches such as DTAM [90], [34] or [137] allow each pixel contributing to the registration process (optimization is performed directly over image pixel intensities). This is also the case for LSD-SLAM [33]. This latter approach is a keyframe method that builds a semi-dense map, which provides far more information about the scene than feature-based approaches. Another interesting feature of LSD-SLAM is that it does not estimate a rigid transformation between two camera positions but a similarity transform which allows solving the scale estimation issue thanks to a scale-drift aware image alignment process. It demonstrated very impressive results showing that scene reconstruction and camera localization can be achieved in real-time without GPU acceleration [114]. A sequel of this work demonstrated that it could run in real-time on a smartphone. It can also be noted that the code has been released to the community [33]. Considering only pixel intensities, these approaches do not need feature extraction and matching process and provide a dense or semi-dense map of the environment. Nevertheless, the underlying illumination model assumes photometric consistency (mainly valid for Lambertian surfaces) which is not always realistic in real scenes and imposes small baselines between frames.

Over the years, EKF based vSLAM has been progressively replaced by keyframe and BA-based methods. This was certainly due to [84] and PTAM [57] which demonstrated that a real-time implementation of BA was possible. Now, real-time bundle adjustments can operate on large-scale environment [33]. For AR applications, with respect to sparse SLAM approaches, such dense or semi-dense map, obtained thanks to direct methods, can be considered to build meshes of the environment and ease interaction between real and virtual worlds.

3.4 Registration in the 3D space

So far we considered a 2D-3D registration process. With some devices (e.g., multiple cameras systems) it is possible to get

5. *Remark:* It has to be noted that for post-production scenario, since real-time constraints are not relevant, all the image of the sequence can be considered (no sparsification of the sequence by keyframe selection is done) within BA methods. Commercial systems such as Boujou from 2D3 (now from Vicon) or MatchMover from Realviz (now in Maya) exploit these very efficient techniques and are widely used in the cinema industry for special effects production. Along with camera localization and scene structure, these softwares are also able to estimate the camera intrinsic parameters and subsequently also handled non-calibrated image sequences.

directly the 3D coordinates of the observed points. In this case, the registration can be done directly in the 3D space. The observed point ${}^c\mathbf{X}$ has to be registered with the model point ${}^w\mathbf{X}$ up to the transformation ${}^c\mathbf{T}_w$ that needs to be estimated.

Denoting $\mathbf{q} \in se(3)$ a minimal representation of ${}^c\mathbf{T}_w$ (as in Section 3.1.2), the problem can be formulated as:

$$\widehat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^{N} ({}^c\mathbf{X}_i - {}^c\mathbf{T}_w {}^w\mathbf{X}_i)^2 \tag{16}$$

and solved using closed form solutions (e.g., [49]) or robust Gauss-Newton or Levenberg-Marquardt approaches. This is a trivial problem when the matching between ${}^c\mathbf{X}_i$ and ${}^w\mathbf{X}_i$ is known (even with some outliers). When this matching is unknown, Iterative Closest Point ICP [16] is a simple and attractive solution to this problem. More efficient solutions than ICP were proposed in [40] [112]. These approaches are used in rigid body target localization used both in augmented and virtual reality [101] (see Figure 13) and proposed in commercial products such as in iotracker or in ART Advanced Realtime Tracking.

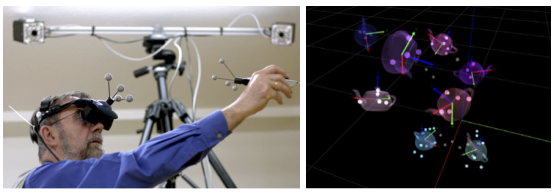

Fig. 13. Pose from rigid body targets and multiple cameras [101]. Position of each element of the constellation are first estimated using triangulation techniques. A rigid transformation is then computed from 3D measurements.

In late 2010 a new sensor, the *kinect*, has been introduced by Primesense and Microsoft. The originality of this sensor is that it is able to provide in real time a dense 3D representation of the environment. Prior to the introduction of this cheap sensor, only expensive time-of-flight camera, heavy structured light systems and stereovision cameras existed. Kinect integrates a structured light (infra-red) depth sensor able to provide depth map at 30Hz. KinectFusion [89] was one of the first systems that enables scene reconstruction and consequently camera localization in real-time and in a way compatible with interactive applications [53] (see Figure 14). The idea is to simultaneously localize the camera and fuse live dense depth data building a global model of the scene. Indeed, estimation of the 6dof camera pose is equivalent to finding the pose that aligns the depth map data onto the current model [89]. This can be done by a modified version of the ICP [16], where the expensive closest point computation is replaced by a projective data-association [17] that allows obtaining fast dense correspondences using closest point approximation. A fast point-to-plane ICP (based on a Gauss-Newton minimization approach) is finally used to register the current dense 3D map with the global model. The camera is then localized and the global model improved[6].

It remains that these methods consider specific sensors. Recent vSLAM approaches, such as [90] [33], that consider only monocular cameras now provide similar results and may be considered as more generic choices.

6. Note that such approaches are also widely used for scene reconstruction. They are able to provide precise models, which can later be used in markerless model-based localization methods (see Section 3.2).

Fig. 14. Particles that interact with the reconstructed scene while camera motion is estimated thanks to KinectFusion [89] [53]

4 Pose Estimation for Planar Scenes

The previous approaches require a 3D model of the tracked object. Since such 3D knowledge is not always easily available (although we have seen that it can be computed on-line), it is also possible to overcome the pose computation considering less constraining knowledge about the viewed scene. In this section, the proposed method copes with this problem by using, at most, the 2D information extracted from the images and the geometrical constraints inherent to a moving vision system. The objective is therefore to estimate the camera displacement between the acquisitions of two images instead of the camera pose. The 3D model is then replaced by a reference (localized) image.

For augmented reality applications, the pose between the camera and the world coordinates system is required. If an initial pose ${}^0\widehat{\mathbf{T}}_w$ is known[7], computing the current pose from the estimated displacement is straightforward and is given by:

$${}^n\widehat{\mathbf{T}}_w = \prod_{n=1}^{M} {}^n\widehat{\mathbf{T}}_{n-1} {}^0\widehat{\mathbf{T}}_w. \tag{17}$$

Usually the current image is registered with an image I_0 in a database for which the pose ${}^0\mathbf{T}_w$ has been computed off-line. Computing ${}^0\widehat{\mathbf{T}}_w$ may require the introduction of 3D information and solutions have been presented in Section 3 and in 4.1.3.

Let us note that drift, due to error accumulation, is inherent to this kind of approach since estimated camera displacements are successively integrated. To limit the drift, it is possible to compute the motion no longer between two successive frames as in (17), but between the current frame and a reference frame (say frame 0) [44]:

$${}^n\mathbf{T}_w = {}^n\mathbf{T}_0 {}^0\mathbf{T}_w \tag{18}$$

Other solutions to limit drift have been proposed in e.g. [140].

4.1 Motion estimation through points correspondences

As stated our goal will be to estimate the 3D motion undergone by the camera between the acquisitions of two images using only 2D image information. An efficient solution to motion estimation through points correspondences relies on the estimation of a homography.

4.1.1 *Overview: the homography*

In [122], it has been proposed to restrict the general case to a simple yet common special case: planar scene. This widely simplifies the pose estimation process. If we now consider a 2D motion

7. To simplify the notation we note ${}^k\mathbf{T}_w$ the position of the camera which acquires frame k and subsequently ${}^k\mathbf{T}_h$ the displacement of the camera between frames k and h.

model noted w that transfers a point $\mathbf{x}_1$ in image I_1 to a point $\mathbf{x}_2$ in image I_2 according to a set $\mathbf{h}$ of parameters ($\mathbf{h}$ can account for a simple translation, an affine motion model, a homography, etc.): $\mathbf{x}_2 = w(\mathbf{x}_1, \mathbf{h})$. From a general point of view, there does not exist a 2D motion model or transfer function $w(.)$ that account for any 3D scene and any camera motion. Nevertheless, it can be demonstrated that, *when the scene is planar*, the coordinates of these two points are linked thanks to a homography ${}^2\mathbf{H}_1$ such that

$$\mathbf{x}_2 = w(\mathbf{x}_1, \mathbf{h}) = {}^2\mathbf{H}_1 \mathbf{x}_1 \tag{19}$$

with

$${}^2\mathbf{H}_1 = ({}^2\mathbf{R}_1 + \frac{{}^2\mathbf{t}_1}{{}^1d} {}^1\mathbf{n}^\top) \tag{20}$$

where ${}^1\mathbf{n}$ and ${}^1d$ are the normal and distance to the origin of the reference plane expressed in camera frame 1 (${}^1\mathbf{n}^1\mathbf{X} = {}^1d$). Let us note that when the camera undergoes a pure rotation motion, ${}^2\mathbf{t}_1 = 0$ and ${}^2\mathbf{H}_1 = {}^2\mathbf{R}_1$. In this special case, equation (19) is then valid regardless the scene structure which has no longer to be planar.

Note that, as for the pose, we can chain the homographies between consecutive frames. We have:

$${}^n\mathbf{H}_w = \prod_{n=1}^{M} {}^n\mathbf{H}_{n-1} {}^0\mathbf{H}_w \quad \text{and then} \quad {}^n\mathbf{H}_w = {}^n\mathbf{H}_0 {}^0\mathbf{H}_w \tag{21}$$

where ${}^0\mathbf{H}_w$ is a homography that map points in frame I_0 to planar 3D points expressed in world coordinates $\mathcal{F}_w$.

4.1.2 Homography estimation

The estimation of ${}^1\mathbf{H}_2$ can be easily and precisely retrieved using a Direct Linear Transform (DLT) algorithm[8], see [48]. Equation (19) can be rewritten as $\mathbf{x}_2 \times {}^2\mathbf{H}_1\mathbf{x}_1 = 0$. If the j-th row of ${}^2\mathbf{H}_1$ is denoted $\mathbf{h}_j^\top$, we have:

$$\mathbf{x}_2 \times {}^2\mathbf{H}_1\mathbf{x}_1 = \begin{pmatrix} y_2\mathbf{h}_3^\top\mathbf{x}_1 - \mathbf{h}_2^\top\mathbf{x}_1 \\ \mathbf{h}_1^\top\mathbf{x}_1 - x_2\mathbf{h}_3^\top\mathbf{x}_1 \\ x_2\mathbf{h}_2^\top\mathbf{x}_1 - y_2\mathbf{h}_1^\top\mathbf{x}_1 \end{pmatrix} \tag{22}$$

with $\mathbf{x}_2 = (x_2, y_2, 1)$. Finally, we have a homogeneous linear system $\mathbf{A}_i\mathbf{h} = 0$ for each corresponding points:

$$\underbrace{\begin{pmatrix} \mathbf{0}^\top & -\mathbf{x}_{1_i}^\top & y_{2_i}\mathbf{x}_{1_i}^\top \\ \mathbf{x}_{1_i}^\top & \mathbf{0}^\top & -x_{2_i}\mathbf{x}_{1_i}^\top \\ -y_{2_i}\mathbf{x}_{1_i}^\top & x_{2_i}\mathbf{x}_{1_i}^\top & \mathbf{0}^\top \end{pmatrix}}_{\mathbf{A}_i(3\times 9)} \underbrace{\begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix}}_{\mathbf{h}(9\times 1)} = 0 \tag{23}$$

where 2 equations are linearly independent. For N matched points we have a system $\mathbf{A}\mathbf{h} = 0$ with $\mathbf{A} = (\mathbf{A}_1 \dots \mathbf{A}_N)^\top$ (see Section 3.1.2 on how to minimize the algebraic distance defined by the norm of $\|\mathbf{A}\mathbf{h}\|$).

Another solution to estimate the homography is to consider the minimization of a cost function, the geometric distance, defined by:

$$\widehat{\mathbf{h}} = \underset{\mathbf{h}}{\operatorname{argmin}} \sum_{i=1}^{N} d(\mathbf{x}_{1i}, {}^1\mathbf{H}_2\mathbf{x}_{2i})^2 \tag{24}$$

which can be solved directly for $\mathbf{h}$ which represents the 8 independent parameters $h_k, k = 1 \dots 8$ of the homography ${}^1\mathbf{H}_2$ using a gradient approach such as a Gauss-Newton. Usually the symmetric error distance $\sum_{i=1}^{N} d(\mathbf{x}_{1i}, {}^1\mathbf{H}_2\mathbf{x}_{2i})^2 + d(\mathbf{x}_{2i}, {}^1\mathbf{H}_2^{-1}\mathbf{x}_{1i})^2$ could also be considered to improve precision. Note that considering the geometric distance as in equation (24) or the algebraic one as for the DLT is, here, equivalent [48].

Rather than solving (24) to estimate the parameters of $\mathbf{H}$ it is also possible to directly perform the optimization over the displacement parameters ${}^1\mathbf{T}_2$. In that case, thanks to (20), we have [102]

$$\widehat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{i=1}^{N} d(\mathbf{x}_{1i}, ({}^2\mathbf{R}_1 + \frac{{}^2\mathbf{t}_1}{{}^1d} {}^1\mathbf{n}^\top)\mathbf{x}_{2i})^2 \tag{25}$$

where $\mathbf{q}$ is a minimal representation of ${}^1\mathbf{T}_2$. This latter method does not require the homography decomposition.

4.1.3 From homography to pose computation

In the case of AR applications, one has to compute the pose ${}^n\mathbf{T}_w$ with respect to a reference frame $\mathcal{F}_w$. The homography can be decomposed to retrieve the pose [48] [36]. Alternatively for planar scenes, one can directly and easily compute the pose when the 3D position of some points is known on the reference plane.

Thus to compute the initial pose ${}^0\mathbf{T}_w$, we assume that all the points lie in the plane ${}^wZ = 0$. In that case each 3D point coordinates is given by ${}^w\mathbf{X} = ({}^wX, {}^wY, 0, 1)^\top$. Their projections in the image plane is then given by:

$$\mathbf{x}_0 = \Pi\, {}^0\mathbf{T}_w {}^w\mathbf{X} = \Pi \begin{pmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & {}^0\mathbf{t}_w \end{pmatrix} \begin{pmatrix} {}^wX \\ {}^wY \\ 0 \\ 1 \end{pmatrix} \tag{26}$$

where $\mathbf{c}_i$ is the i-th column of the rotation matrix ${}^0\mathbf{R}_w$ which can be rewritten as:

$$\begin{aligned} \mathbf{x}_0 &= \Pi \begin{pmatrix} \mathbf{c}_1 & \mathbf{c}_2 & {}^0\mathbf{t}_w \end{pmatrix} ({}^wX, {}^wY, 1)^\top \\ &= {}^0\mathbf{H}_w ({}^wX, {}^wY, 1)^\top \end{aligned} \tag{27}$$

${}^0\mathbf{H}_w$ is a homography that maps the plane of the object (${}^wZ = 0$) on the image plane. It can be easily computed using the DLT algorithm presented in the previous paragraph. Knowing ${}^0\mathbf{H}_w$, the pose ${}^0\mathbf{T}_w$ can be easily computed noting that $(\mathbf{c}_1, \mathbf{c}_2, {}^0\mathbf{t}_w) = \Pi^{-1}\, {}^0\mathbf{H}_w$. Considering that the rotation matrix is orthogonal, the third column of the rotation matrix is computed such that $\mathbf{c}_3 = \mathbf{c}_1 \times \mathbf{c}_2$. This is an easy way to estimate pose when the scene is planar[9].

Ultimately one wants to compute ${}^n\mathbf{T}_w$. Like ${}^0\mathbf{T}_w$ that can be retrieved from ${}^0\mathbf{H}_w$, ${}^n\mathbf{T}_w$ can be retrieved from the homography ${}^n\mathbf{H}_w$. Indeed, similar to equation (27), we have $(\mathbf{c}_1, \mathbf{c}_2, {}^n\mathbf{t}_w) = \Pi^{-1}\, {}^n\mathbf{H}_w$ (where $\mathbf{c}_i$ is, here, the i-th column of the rotation matrix ${}^n\mathbf{R}_w$) and $\mathbf{c}_3 = \mathbf{c}_1 \times \mathbf{c}_2$. This gives the complete pose ${}^n\mathbf{T}_w$.

4.1.4 Discussion

These motion estimation processes through point correspondences have been widely studied e.g. [122] (see Figure 15, left). This is one of the standard approaches for augmenting planar scene. It can be extended to the case of multiple planes [121]. Non-planar scene can be considered when pure rotational camera motions are considered [122] [102]. Alternatively, multiple planar structures can be considered [121] (see Figure 15, right).

As stated the current image is often registered with a localized image in a database. This is the case for an augmented museum application as shown on Figure 23 or for an augmented book application as shown on Figure 16. For each reference image in

8. *An example of the DLT code for homography estimation is proposed as a supplementary material of this paper and is available here*

9. *The code of this pose estimation method based on the DLT for homography estimation is proposed as a supplementary material of this paper and is available here.*

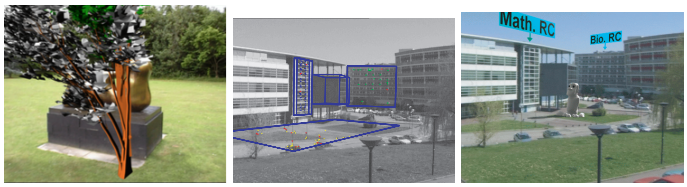

Fig. 15. Pose estimation through homography estimation for a single plane (left, [122]) and for multiple planes (right, [121]).

the database, a pose ${}^0\mathbf{T}_w$ is estimated off-line using, for example, the planar pose estimation method presented in Section 4.1.3. A homography ${}^n\mathbf{H}_0$ that links the current image I_n and the reference one is estimated which allows to deduce the pose ${}^n\mathbf{T}_0$ and finally, according to equation (18), ${}^n\mathbf{T}_w$.

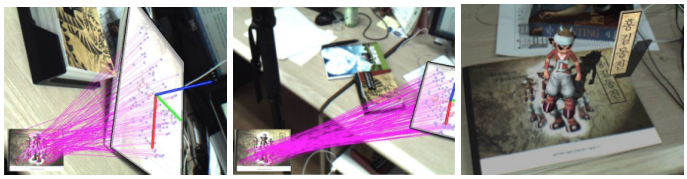

Fig. 16. Augmented book [55]: Sift matching followed by a RANSAC based homography estimation and augmented contents (see full video).

We quickly discuss in Section 5 the 2D point matching issue which is fundamental in the development of such approaches. In any case, AR based on homography estimation from point correspondences has become a standard in the industry and many commercial libraries providing such capabilities are now available (Metaio, Vuforia from PTC Inc., Total Immersion, etc.)

4.2 Motion estimation using direct image registration

All the previous approaches consider pure geometric methods. An alternative is to fully embed the motion estimation process in an image processing process. 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. It is also possible to consider that the 2D model is a reference image (or a template). In this case, the goal is to estimate the motion (or warp) between the current image and a reference template at the pixel intensity level.

4.2.1 Template registration

Let us consider that the appearance of the object is learned from a model defined as a reference image I_0 at some pixel locations $\mathbf{x} \in W$ (W is a set of pixels that defines the template to be tracked) and that we seek its new location $w(\mathbf{x},\mathbf{h})$ in an image I. As seen in Section 4.1.1, $\mathbf{h}$ are parameters of a motion model. In AR applications it is usually modeled by a homography and is then defined by equation (19). It is then possible to directly define this alignement or registration problem as a minimization of the dissimilarity (or maximization of the similarity) between the appearance in I_0 at the positions $\mathbf{x}$ in a region W and in I at the positions $w(\mathbf{x},\mathbf{h})$. An analytic formulation of the registration problem can then be written as:

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} \sum_{\mathbf{x}\in W} f(I_0(\mathbf{x}), I(w(\mathbf{x},\mathbf{h}))) \tag{28}$$

where f is, here, a dissimilarity function. The choice of the similarity function is important. An obvious choice originates in the brightness constancy constraint stating that:

$$I(\mathbf{x}) = I(w(\mathbf{x},\mathbf{h})) = I_0(\mathbf{x})$$

is to consider the sum of squared differences (SSD). In this case, when the appearance is defined as the set of pixel intensities of the patch and the dissimilarity function is the SSD, it leads typically to the KLT (for Kanade-Lucas-Tomasi algorithm) [75], [118] for small patches and translational model or to [10], [46] for large template and affine motion. For augmented reality applications, homography has to be preferred [13] as it allows inferring the camera pose. The problem can be rewritten as:

$$\widehat{\mathbf{h}} = \arg\min_{\mathbf{h}} C(\mathbf{h}) = \sum_{\mathbf{x}\in W} (I_0(\mathbf{x}) - I(w(\mathbf{x},\mathbf{h})))^2 \tag{29}$$

This is a non-linear optimization problem which, as for the pose problem defined in Section 3.1.2, can be efficiently solved by a Gauss-Newton method[10].

Remark: the KLT.

The method presented in this paragraph considers a homography estimation. When a small patch and a translation motion model is considered, this leads to the KLT algorithm [75] used to track points over frames. From these tracked points one can easily compute the homography between two frames, using for example the DLT approach presented in Section 4.1.2.

4.2.2 Extensions and improvements

The formulation presented in the previous section is the most simple and intuitive. It is usually referred as the forward additional approach and has been initially proposed in [75] (KLT). Other approaches can be considered such as the forward compositional approach [119], the inverse compositional approach [10] (for which the pseudo-inverse of the Jacobian has to be computed only once beforehand) or the Efficient Second Order Minimization (ESM) method [13] (see Figure 19).

Considering a planar hypothesis, these methods are well suited to augment planar targets such as painting in museum (see Figure 17) or books (see Figure 19).

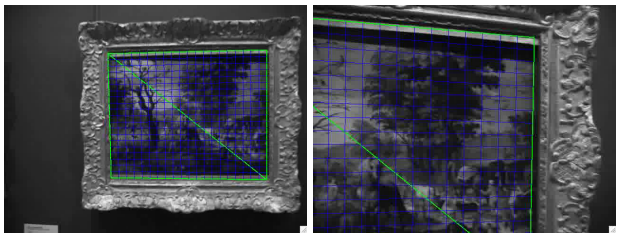

Fig. 17. Direct image registration and camera localization in a museum [11] (see full video).

Extensions of template trackers have been proposed to handle efficiently blur [97] (see Figure 18) or degradation in image resolution [52]. Extension to planar model can also be considered by adding a parallax term to the definition of a homography [103]. Rather than a homography, motion models that include deformations (modeled using radial basis functions or free form deformation) have also been proposed [43].

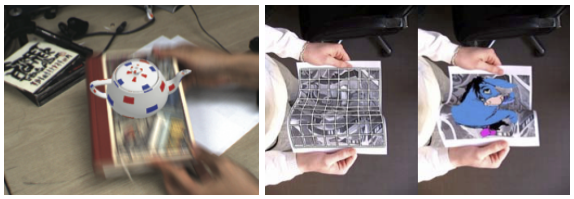

Fig. 18. Extension of template registration process to handle blur [97] or to consider large template deformation [43].

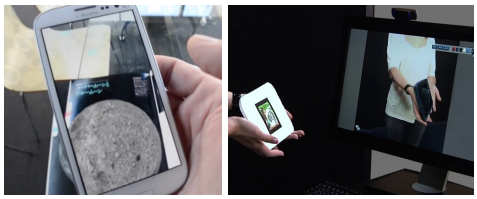

Fig. 19. Companies such as Robocortex propose a SDK based on template tracking methods (included in AugmentedPro) which can be integrated in third party products such as xloudia (left) or Dassault Systemes 3DVIA (right, see video).

However, the SSD is not effective in the case of illumination changes and occlusions. Several solutions have been proposed to add robustness toward these variations. The former solution is to consider an M-Estimator (see Section 3.1.3) as proposed in, e.g., [46]. The later deals with the choice of the (dis-)similarity function that is also important. Along with the SSD, one can consider local zero-mean normalized cross correlation (ZNCC) [51], the Sum of Conditional Variances (SCV) [107] or the mutual information (MI) [25]. The later criterion, the mutual information, proposes to maximize the information shared between the reference image and the current one. MI has proved to be robust to occlusions and illumination variations and can therefore be considered as a good alignment measure for tracking and localization [25], [26]. In [25], it has been demonstrated that multimodal registration (using e.g. infrared and visible image) can be handled using mutual information as a similarity function (see Figure 20).

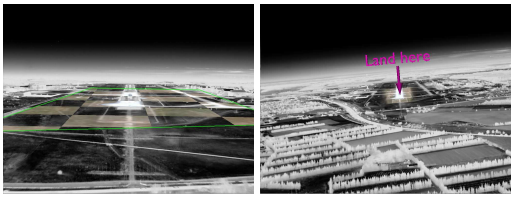

Fig. 20. Registration and homography estimation between infrared and visible image (from Google Earth) for camera localization (and augmentation) [25].

4.3 Merging various cues to improve localization

It has been noted that it could be interesting to merge 2D-3D registration methods along with 2D-2D ones. Indeed, approaches which directly compute the pose (Section 3) are intrinsically mono-image processes and can be subject to jitter, whereas motion-based methods (Section 4) consist in multi-view processes that are subject to drift. Therefore, merging multiple cues from markerless model-tracking (Section 3.2) and motion-estimation has received some interest in the AR community.

Most of the current approaches that integrate multiple cues in a tracking process are probabilistic techniques. Most of these approaches rely on the well-known Extended Kalman filter or particle filter [132] [64] [45] but non-linear optimization techniques have also been considered (see Figure 21). In [141] the proposed localization approach considers both 2D-3D matching against a key-frame and 2D-2D temporal matching (which introduces multiple view spatio-temporal constraints in the tracking process). An extension is proposed in [140] to integrate contribution of a model-based tracker similar to [24], [31]. In [104], it is proposed to fuse a classical model-based approach based on the edge extraction and a temporal matching (motion estimation) relying on texture analysis into a single non-linear objective function that has then to be minimized. In [98], color cues along with keypoints matching and edge-based model tracking are combined to provide a very robust tracker.

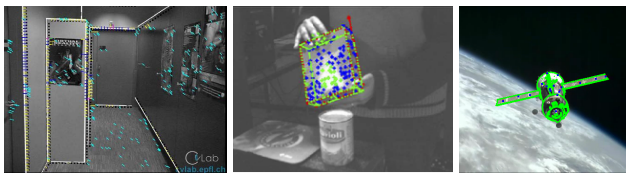

Fig. 21. Merging keypoints and model-based tracking in a single minimization process [141] (left), [104] (middle), [98] (right). This allows to introduce a spatio-temporal constraints in a model-based tracking approach.

5 Matching Low-Level Features

At this point, the geometry that underlies the camera localization problem has been reviewed. Formulation of the problem, along with resolution techniques, has been exposed. Although an initialization is always required (and can be quite complex), edge-based model tracking (section 3.2) and template tracking algorithms (section 4.2) act as tracking methods and can be considered as self-contained. For other methods, P*n*P, SLAM, homography from point correspondences, low level features extraction and matching processes are required. A comprehensive review of all the approaches proposed in the literature seems out of reach [136]. In this section, we review the main solutions that have been considered in actual AR systems.

5.1 Fiducial marker detection and localization

Fiducial markers have been a basic tool in developing AR applications. Indeed, they allow achieving simultaneously both target identification and camera localization. Such markers were first introduced in [106] and extended to ARtoolkit [54], ARToolkit plus [145], Studierstube Tracker, and ARTag [37] [38]. To simplify the detection process and the underlying image processing algorithm, their design is ultimately simplified. Square shape and binary color combination are usually considered (see Figure 22). More precisely, rectangle shape is first searched in a binarized image, and then camera pose with respect to the rectangle is computed from the known 3D coordinates of four corners of the marker using approaches similar to those presented in section 3.1 or 4.1.3. The texture inside the marker is uniquely designed for marker identification. Circular shape is often selected as an alternative to square shape [86]. Since single circle is not sufficient

10. *We propose as a supplementary material of this paper (here) an example of how to use such tracker. The interested reader could easily access the full source code of the tracker in ViSP [79].*

for camera localization, multiple circles are randomly [139], circularly [14] or squarely [15] distributed on a plane.

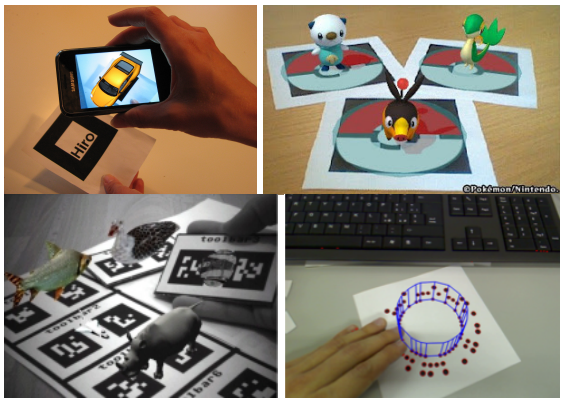

Fig. 22. ARToolkit [54], Pokedex 3D (Nintendo), ARTag in the Magic Lens system [37] (see video), circular Rune-Tag [14] (see video).

Although research related to markers is still active, the development of keypoints matching methodologies in the late 1990s allows augmented reality reaching a new maturity level.

5.2 Keypoints matching

In the previous sections, we mentioned that point correspondences should be available beforehand. These correspondences are established between 2D points in the image and points of a 3D reference model for P*n*P (section 3.1) and between two 2D points located on a plane for homography estimation (section 4.1).

In the literature, SIFT [72], which has been considered a breakthrough for 2D points matching, was proposed in 1999 and then various types of keypoint detectors and descriptors have been considered. The common framework for 2D matching usually consider three steps: keypoints extraction, description and matching. First, among all the pixels in the image, a subset of pixels of interest is selected according to a criterion of "cornerness". For each selected pixel, its local texture is then converted into a descriptor (a vector that intends to encode, in a unique way, the keypoint and its local neighborhood). Finally, these descriptors extracted in two images are matched to find correspondences.

As far as pose or homography estimation is concerned, keypoint descriptors on a reference model (3D or image model) are first computed offline and stored in a descriptor database. Then, on-line, keypoints are extracted from each image and matched, in the descriptor space, with those in the database. Finally, camera pose or displacement can be computed from these correspondences (see Figure 16 and 23).

Feature extraction

From a captured image, local features are extracted according to image properties computed from texture such as "cornerness". Ideally, since a camera can freely move in AR applications, such features should be extracted from perspectively transformed images. This process should be highly repeatable and performed in real time. Therefore, existing keypoint detectors are designed to feature invariance properties with respect to geometric transformation such as translation, rotation, affine transformation and scale change.

Historically, Harris detector [47] is a widely used corner detector that computes the cornerness score of each pixel from gradients

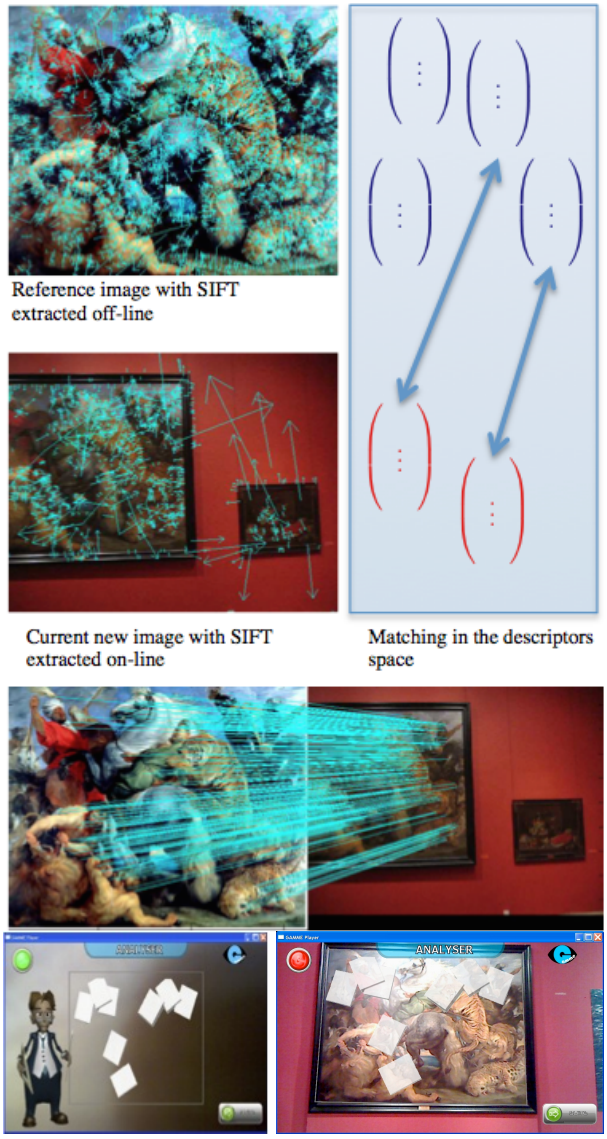

Fig. 23. Keypoints matching framework. From a reference image (top), a set of keypoints is extracted and the corresponding descriptor vectors are computed off-line. In the current image, another set of keypoints are extracted and their corresponding descriptor vectors are computed on-line and matched with those of the reference image. Here SIFT were considered [72]. If the reference image is localized (ie, ${}^0\mathbf{T}_w$ or ${}^0\mathbf{H}_w$ has been computed off-line, see section 4.1.3), camera localization can then be computed thanks to a homography estimation). CG image to be inserted and final augmentation (bottom). Image from [116] [11].

of an image patch. The cornerness score is then classified into flat, edge and corner according to the intensity structure of the patch. SUSAN is an alternative approach that selects a pixel as a corner if it is not self-similar within a local image patch. The similarity is computed between a pixel and its surrounding pixels in the patch instead of computing gradients [125]. FAST [108] follows SUSAN's approach and considers only pixels on a circle for fast extraction. FAST is computationally fast because it only computes similarity with pixels selected with a machine learning technique. AGAST [77] further improved computational cost against FAST by dynamically changing the optimal configuration of pixels for similarity measurement.

Since the keypoints mentioned above are not scale-invariant, an image pyramid can be considered so that keypoints can be detected under scale changes. But to deal with scale issue, several

scale-invariant detectors based on scale space theory have been proposed [70]. Generally, a linear Gaussian scale space is built and local extrema on this space is selected as a keypoint. One of the first scale-invariant keypoint detector used Laplacian of Gaussian (LoG) [71]. But for efficiency issue, LoG is approximated by a difference of Gaussian in SIFT [72], and is further accelerated with GPU [123] so that it can be used in AR applications. In SURF [12], the determinant of the Hessian is used as another scale-space operator and is computed efficiently with integral images. Recently, KAZE [2] employs a non-liner diffusion filtering as a non-linear scale space so that object boundaries can be retained and keypoints extraction be more robust, and accelerated for real-time detection [3]. Note that non-maximum suppression that selects only local extrema of cornerness scores within a region is normally used after extraction because redundant keypoints can be extracted and may cause false correspondences [88].

Feature description

The next step usually consists in computing a feature vector that fully describes the keypoint and its local neighborhood. For robustness issue, the resulting descriptor should be made invariant to geometric and photometric variations. Rotation invariance is normally achieved by assigning orientation to extracted keypoints for orientation normalization. Orientation is computed by several ways as the peak of histogram of gradient in an image patch [72] and center of mass [42]. For each oriented keypoint, a feature vector (a descriptor) is then computed. Roughly, a local keypoint descriptor can be mainly classified into two approaches: histogram of oriented gradients or intensity comparisons.

Histogram of oriented gradients used in SIFT [72], [124] is computed such that an image patch is segmented into small regions, and histogram of oriented gradients in each region is computed and finally concatenated (see Figure 24). This well preserves the shape and intensity information of an image patch. A similar framework is used in SURF [12] and CARD [4]. Since feature descriptors from the methods above have floating-point values, they can be compacted into a binary string with machine learning techniques [128], [135]. Memory consumption in a descriptor database and computational cost for matching is then reduced.

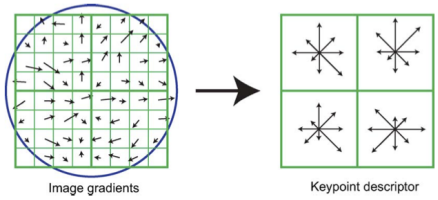

Fig. 24. Histogram of oriented gradients (image from [72]): gradient is computed in the neighborhood of the keypoint. 8 bins histogram of gradient are then computed in each 4x4 region and concatenated to build the descriptor.

Intensity comparisons based approach has recently been considered. In BRIEF [21], a descriptor is composed of a binary string in which each binary digit is computed from intensity comparison between pairwise pixels (see different pattern in 25). A binary value is described by 0 if a pixel is brighter and 1 if darker in the comparison. The descriptor is then composed of a binary string concatenating the result of a set of binary tests. This means that a binary descriptor is directly computed from the image patch while gradients based approaches need additional computations. They are far more computationally efficient. To increase the discriminative property of descriptors, different designs of intensity comparisons have been proposed in ORB [111] (rotation invariance), BRISK [69] (scale and rotation invariance), and FREAK [1], LDB [148].

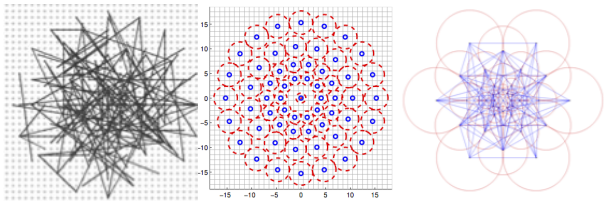

Fig. 25. Binary descriptors: a pattern is used for sampling the neighborhood of the keypoint. Pattern for BRIEF [21] (left), BRISK [69] (center), Freak [1] (right)

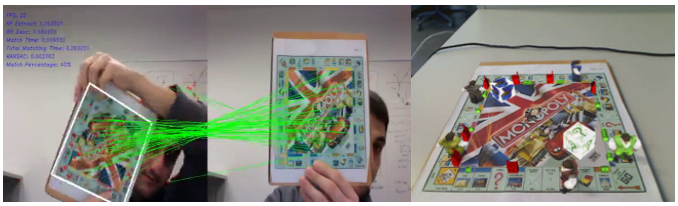

Fig. 26. Game AR apps [83]. Matching using BRIEF descriptors [21] (see video1 and video2)

All the methods above need correct orientation assignment to match before computing descriptors. This means that keypoints are never matched if the orientation assignment failed. To avoid computing orientation, rotation invariant descriptors have also been proposed in [35], [130].

Since inertial sensors are now available in mobile phones, gravity direction may be incorporated in keypoint descriptor [63]. According to gravity direction, a captured image is first rectified and orientations from both texture and gravity are used to enhance the distinctiveness of descriptors.

Matching

For AR applications, keypoints matching usually consider a nearest neighbor searching approach. The idea is basically to find the closest descriptor in the reference image in the descriptor space. Since this is not a generic problem, various efficient solutions for this problem have been already proposed [85]. If a feature descriptor is binary, brute-force matching with hamming distance (XOR) is used because it can be efficiently implemented with common CPUs.

5.3 Alternative frameworks

Recently, keypoint matching has been formulated as a classification problem [66], [95]. Compared to the classical framework presented above, the view set of a keypoint under affine transformations is compactly described and treated as one class. At run-time, a classification technique is used for deciding to which class an extracted keypoint belongs. In this approach, statistical classification tools such as randomized trees [66] and random ferns [95] are applied.

Enforcing geometrical constraints between keypoints can also be used to ease matching. In [138], it is proposed to match keypoint thanks to geometric features instead of using local image

patches. Geometrical relationship between neighbor keypoints is used as a feature so that various kinds of rich and binary textures can be detected.

Another interesting approach is to consider a contour-based approach for non-textured objects [29] [80]. In [29], contours of the objects are extracted with MSER [82] and cross ratios are computed from bitangent of each contour as a descriptor.

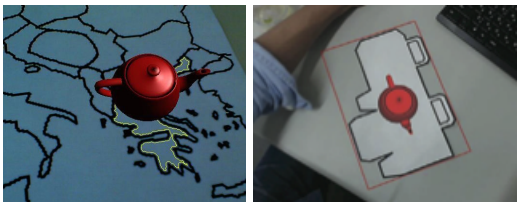

Fig. 27. Pose from a single, closed object contour (MSER) [29] (see video), [80]

5.4 Chosing the "best" matching techniques.

It is difficult to state that one approach is better than the other. This is usually a trade-off between stability, number of extracted keypoints, recall, percentage of outliers, computational cost, etc. It has to be noted that most of these low-level matching methods are proposed in OpenCV or VLFeat [142]. This is the case for SIFT, SURF, FAST, BRIEF, ORB, MSER, etc. It is then easy to test each methods in a specific context and chose the most efficient one. SIFT, which is patented in the US, have proved for year [72] to be very efficient and a good choice (although quite heavy to compute). From a practical point of view, it seems that FAST is often used in augmented reality libraries; it is for example used in Vuforia from Qualcomm or Zappar.

6 Conclusion

This survey is an attempt to cover the camera localization problem for real-time augmented reality purposes. We mainly focus on the geometrical aspects of the pose estimation seen here as an alignment problem. We also provide hints to the low level image processing techniques inherent to this process. Our goal in writing this survey was to produce a guide for researchers, practitioners and students involved in the development of AR applications. We hope that the presented material and the accompanying examples fulfill the initial objective.

We focused on one of the basic tools required in AR applications. Despite the tremendous progress in the area, much work remain to be done. Five years ago, tracking reliability and robustness (to occlusions, fast camera motions, cluttered scene,...) was clearly an issue. This now has been clearly improved (thank also to the joint use of computer vision techniques and other sensors such as IMU). Beyond the camera localization one can also consider occlusions detection and handling, dynamic scenes, light source direction. For a precise rendering process, such issues have to be considered.

The last decade has also seen the development of many companies and start-ups involved in AR. Nevertheless few "killer apps" emerged [87] in the industry. Still, most of the proposed systems are prototypes. Scalability of the solutions, end-users and market acceptance are clearly potential improvement areas that must be considered by both academics and industries.

References

[1] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'12*, pages 510–517, 2012.

[2] P. Alcantarilla, A. Bartoli, and A. Davison. KAZE features. In *European Conf. on Computer Vision*, pages 214–227, 2012.

[3] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conf., BMVC*, 2013.

[4] M. Ambai and Y. Yoshida. CARD: Compact and real-time descriptors. In *Int. Conf. on Computer Vision*, pages 97–104, 2011.

[5] M.-A. Ameller, B. Triggs, and L. Quan. Camera pose revisited–new linear algorithms. In *European Conf. on Computer Vision*, 2000.

[6] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg. Wide area localization on mobile phones. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'09*, pages 73–82, 2009.

[7] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, August 1997.

[8] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Application*, 21(6):34–47, November 2001.

[9] M. Bajura and U. Neumann. Dynamic registration correction in video-based augmented reality systems. *IEEE Computer Graphics and Applications*, 15(5):52–60, 1995.

[10] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. Journal of Computer Vision*, 56(3):221–255, 2004.

[11] A. Bationo-Tillon, J. Laneurit, E. Marchand, F. Servant, I. Marchal, and P. Houlier. A day at the museum: an augmented fine-art exhibit. In *IEEE Int. Symp. on Mixed and Augmented Reality (Art, Media and Humanities), ISMAR'10-AMH*, pages 69–70, Seoul, Korea, October 2010.

[12] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[13] 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.

[14] F. Bergamasco, A. Albarelli, E. Rodola, and A. Torsello. RUNE-Tag: A high accuracy fiducial marker with strong occlusion resilience. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 113–120, 2011.

[15] F. Bergamasco, A. Albarelli, and A. Torsello. Pi-tag: a fast image-space marker design based on projective invariants. *Machine vision and applications*, 24(6):1295–1310, 2013.

[16] P.J. Besl and N.D. McKay. A method for registration of 3-d shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[17] G. Blais and M.D. Levine. Registering multiview range data to create 3d computer objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(8):820–824, August 1995.

[18] G. Bleser, Y. Pastarmov, and D. Stricker. Real-time 3d camera tracking for industrial augmented reality applications. *Journal of WSCG*, pages 47–54, 2005.

[19] G. Bleser, H. Wuest, and D. Stricker. Online camera pose estimation in partially known and dynamic scenes. In *IEEE/ACM Int. Symp. on Mixed and Augmented Reality, ISMAR'06.*, pages 56–65, 2006.

[20] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[21] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. BRIEF: Computing a local binary descriptor very fast. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(7):1281–1298, July 2012.

[22] F. Chaumette and S. Hutchinson. Visual servo control, Part I: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[23] C. Choi and H.I. Christensen. Robust 3d visual tracking using particle filtering on the special euclidean group: A combined approach of keypoint and edge features. *Int. Journal of Robotics Research*, 31(4):498–519, April 2012.

[24] 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*, 12(4):615–628, July 2006.

[25] A. Dame and E. Marchand. Accurate real-time tracking using mutual information. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'10*, Seoul, Korea, October 2010.

[26] A. Dame and E. Marchand. Second order optimization of mutual information for real-time image registration. *IEEE Trans. on Image Processing*, 21(9):4190–4203, September 2012.

[27] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *IEEE Int. Conf. on Computer Vision*, pages 1403–1410, 2003.

[28] D. Dementhon and L. Davis. Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[29] M. Donoser, P. Kontschieder, and H. Bischof. Robust planar target tracking and pose estimation from a single concavity. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'11*, pages 9–15, 2011.

[30] K. Dorfmüller. Robust tracking for augmented reality using retroreflective markers. *Computers & Graphics*, 23(6):795–800, 1999.

[31] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):932–946, July 2002.

[32] E. Eade and T. Drummond. Scalable monocular slam. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'2006*, volume 1, pages 469–476, June 2006.

[33] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision, ECCV'14*, September 2014.

[34] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *IEEE Int. Conf. on Computer Vision, ICCV'13*, pages 1449–1456, December 2013.

[35] B. Fan, F. Wu, and Z. Hu. Rotationally invariant descriptors using intensity order pooling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(10):2031–2045, October 2012.

[36] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, 1988.

[37] M. Fiala. Artag, a fiducial marker system using digital techniques. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'2005*, pages 590–596, June 2005.

[38] M. Fiala. Designing highly reliable fiducial markers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(7):1317–1324, July 2010.

[39] N. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communication of the ACM*, 24(6):381–395, June 1981.

[40] A.W. Fitzgibbon. Robust registration of 2d and 3d point sets. *Image and Vision Computing*, 21(12-13):1145–1153, December 2003.

[41] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(8):930–943, August 2003.

[42] S. Gauglitz, M. Turk, and T. Höllerer. Improving keypoint orientation assignment. In *British Machine Vision Conference*, 2011.

[43] V. Gay-Bellile, A. Bartoli, and P. Sayd. Deformable surface augmentation in spite of self-occlusions. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'07*, Nara, Japan, November 2007.

[44] Y. Genc, S. Riedel, F. Souvannavong, C. Akinlar, and N. Navab. Marker-less tracking for AR: A learning-based approach. In *IEEE Int. Symp. on Mixed and Augmented Reality (ISMAR'02)*, pages 3–6, Darmstadt, Germany, September 2002.

[45] M. Haag and H.H. Nagel. Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences. *Int. Journal of Computer Vision*, 35(3):295–319, December 1999.

[46] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, October 1998.

[47] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Conference*, pages 147–151, Manchester, 1988.

[48] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2001.

[49] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. of the Optical Society of America, JOSA A*, 4(4):629–642, 1987.

[50] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.

[51] 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.

[52] E. Ito, T. Okatani, and K. Deguchi. Accurate and robust planar tracking based on a model of image sampling and reconstruction process. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'11*, pages 1–8, Oct 2011.

[53] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *ACM Symp. on User Interface Software and Technology, UIST'11*, October 2011.

[54] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *ACM/IEEE Int. Workshop on Augmented Reality, IWAR'99*, pages 85–95, San Francisco, CA, October 1999.

[55] K. Kim, V. Lepetit, and W. Woo. Scalable real-time planar targets tracking for digilog books. *The Visual Computer*, 26(6-8):1145–1154, 2010.

[56] G. Klein and T. Drummond. Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing*, 22(10):769–776, September 2004.

[57] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[58] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, Orlando, October 2009.

[59] L. Kneip and P. Furgale. Opengv: A unified and generalized approach to real-time calibrated geometric vision. In *IEEE INt. Conf. on Robotics and Automation, ICRA'14*, May 2014.

[60] L. Kneip, P. Furgale, and R. Siegwart. Using multi-camera systems in robotics: Efficient solutions to the npnp problem. In *IEEE Int. Conf. on Robotics and Automation, ICRA'13*, pages 3770–3776, May 2013.

[61] L. Kneip, H. Li, and Y. Seo. Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In *European Conf. on Computer Vision–ECCV 2014*, pages 127–142. Springer, 2014.

[62] L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2011*, pages 2969–2976, June 2011.

[63] D. Kurz and S. Benhimane. Handheld augmented reality involving gravity measurements. *Computers & Graphics*, 36(7):866–883, July 2012.

[64] 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.

[65] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, October 2005.

[66] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, September 2006.

[67] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate O(n) solution to the PnP problem. *Int. Journal of Computer Vision*, 81(2):155–166, 2009.

[68] V. Lepetit, L. Vacchetti, T. Thalmann, and P. Fua. Fully automated and stable registration for augmented reality applications. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03*, pages 93–102, Tokyo, Japan, October 2003.

[69] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *International Conference on Computer Vision*, pages 2548–2555, 2011.

[70] T. Lindeberg. *Scale-space theory in computer vision*. Springer, 1993.

[71] T. Lindeberg. Feature detection with automatic scale selection. *Int. Journal of Computer Vision*, 30(2):79–116, 1998.

[72] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004.

[73] D.G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):441–450, May 1991.

[74] 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*, 22(6):610–622, June 2000.

[75] 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.

[76] Y. Ma, S. Soatto, J. Košecká, and S. Sastry. *An invitation to 3-D vision*. Springer, 2004.

[77] E. Mair, G. Hager, D. Burschka, M. Suppa, and G. Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *European Conf. on Computer Vision*, pages 183–196, 2010.

[78] E. Marchand and F. Chaumette. Virtual visual servoing: a framework for real-time augmented reality. In G. Drettakis and H.-P. Seidel, editors, *EUROGRAPHICS'02 Conf. Proceeding*, volume 21(3) of *Computer Graphics Forum*, pages 289–298, Saarebrücken, Germany, September 2002.

[79] 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*, 12(4):40–52, December 2005. Special Issue on "Software Packages for Vision-Based Control of Motion", P. Oh, D. Burschka (Eds.).

[80] S. Martedi, B. Thomas, and H. Saito. Region-based tracking using sequences of relevance measures. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'13*, pages 1–6, 2013.

[81] P. Martin, E. Marchand, P. Houlier, and I. Marchal. Mapping and re-localization for mobile augmented reality. In *IEEE Int. Conf. on Image Processing, ICIP'14*, Paris, France, October 2014.

[82] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, October 2004.

[83] E. Molla and V. Lepetit. Augmented reality for board games. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'10*, pages 253–254, 2010.

[84] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *IEEE Int. Conf. on Computer Vision*, volume 1, pages 363–370, 2006.

[85] M. Muja and D. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36, 2014.

[86] L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Int. Symp. on Mixed and Augmented Reality*, page 27, 2002.

[87] N. Navab. Developing killer apps for industrial augmented reality. *IEEE Computer Graphics and Applications*, 24(3):16–20, May 2004.

[88] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *IAPR Int. Conf. on Pattern Recognition*, volume 3, pages 850–855, 2006.

[89] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE/ACM Int. Symp. on Mixed and Augmented Reality, ISMAR'11*, pages 127–136, Basel, October 2011.

[90] R. Newcombe, S. Lovegrove, and A. Davison. Dtam: Dense tracking and mapping in real-time. In *IEEE Int. Conf. on Computer Vision*, pages 2320–2327, 2011.

[91] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(6):756–770, June 2004.

[92] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, page 652, 2004.

[93] D. Oberkampf, D.F. Dementhon, and L.S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, May 1996.

[94] C. Olsson, F. Kahl, and M. Oskarsson. Branch-and-bound methods for euclidean registration problems. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(5):783–794, May 2009.

[95] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(3):448–461, March 2010.

[96] Y. Park, V. Lepetit, and W. Woo. Multiple 3d object tracking for augmented reality. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 117–120, Washington, USA, 2008.

[97] Y. Park, V. Lepetit, and W. Woo. Handling motion-blur in 3d tracking and rendering for augmented reality. *IEEE Trans. on Visualization and Computer Graphics*, 18(9):1449–1459, Sept 2012.

[98] A. Petit, E. Marchand, and A. Kanani. Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes. In *IEEE Int. Conf. on Robotics and Automation, ICRA'14*, pages 4115–4120, Hong Kong, China, June 2014.

[99] A. Petit, E. Marchand, and K. Kanani. Tracking complex targets for space rendezvous and debris removal applications. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'12*, pages 4483–4488, Vilamoura, Portugal, October 2012.

[100] A. Petit, E. Marchand, and K. Kanani. Augmenting markerless complex 3d objects by combining geometrical and color edge information. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR 2013*, pages 287–288, Adelaide, Australia, October 2013.

[101] T. Pintaric and H. Kaufmann. Affordable infrared-optical pose-tracking for virtual and augmented reality. In *Proc. of Trends and Issues in Tracking for Virtual Environments Workshop, IEEE VR*, pages 44–51, 2007.

[102] M. Pressigout and E. Marchand. Model-free augmented reality by virtual visual servoing. In *IAPR Int. Conf. on Pattern Recognition, ICPR'04*, volume 2, pages 887–891, Cambridge, UK, August 2004.

[103] M. Pressigout and E. Marchand. Hybrid tracking algorithms for planar and non-planar structures subject to illumination changes. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'06*, pages 52–55, Santa Barbara, CA, October 2006.

[104] M. Pressigout and E. Marchand. Real-time hybrid tracking using edge and texture information. *Int. Journal of Robotics Research, IJRR*, 26(7):689–713, July 2007.

[105] L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(8):774–780, August 1999.

[106] J. Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *Computer Human Interaction, 3rd Asia Pacific*, pages 63–68, 1998.

[107] R. Richa, R. Sznitman, R. Taylor, and G. Hager. Visual tracking using the sum of conditional variance. In *IEEE Conference on Intelligent Robots and Systems, IROS'11*, pages 2953–2958, San Francisco, September 2011.

[108] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(1):105–119, January 2010.

[109] P.J. Rousseeuw. Least median of squares regression. *Journal American Statistic Association*, 79:871–880, 1984.

[110] E. Royer, M. Lhuillier, M. Dhome, and J.M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *Int. Journal of Computer Vision*, 74(3):237–260, 2007.

[111] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. In *Int. Conf. on Computer Vision*, pages 2564–2571, 2011.

[112] R. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *IEEE Int. Conf. on Robotics and Automation, ICRA'09.*, pages 3212–3217, 2009.

[113] G. Schall, H. Grabner, M. Grabner, P. Wohlhart, D. Schmalstieg, and H. Bischof. 3d tracking in unknown environments using on-line keypoint learning for mobile augmented reality. In *IEEE Conf. on Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08.*, pages 1–8, June 2008.

[114] T. Schöps, J. Engel, and D. Cremers. Semi-dense visual odometry for AR on a smartphone. In *International Symposium on Mixed and Augmented Reality, ISMAR'14*, September 2014.

[115] G. Schweighofer and A. Pinz. Robust pose estimation from a planar target. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(12):2024–2030, December 2006.

[116] F. Servant, E. Marchand, P. Houlier, and I. Marchal. Visual planes-based simultaneous localization and model refinement for augmented reality. In *IAPR, Int. Conf. on Pattern Recognition, ICPR'08*, Tampa, Florida, dec 2008.

[117] A. Shahrokni, L. Vacchetti, V. Lepetit, and P. Fua. Polyhedral object detection and pose estimation for augmented reality applications. In *Proc. of Computer Animation*, pages 65–69, 2002.

[118] 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.

[119] H.-Y. Shum and R. Szeliski. Construction of panoramic image mosaics with global and local alignment. *Int. Journal of Computer Vision*, 36(2):101–130, 2000.

[120] G. Simon and M-O Berger. A two-stage robust statistical method for temporal registration from features of various type. In *IEEE Int. Conf. on Computer Vision*, pages 261–266, Bombay, India, January 1998.

[121] G. Simon and M.-O. Berger. Reconstructing while registering: A novel approach for markerless augmented reality. In *IEEE Int. Symp. on Mixed and Augmented Reality (ISMAR'02)*, pages 285–294, Darmstadt, Germany, September 2002.

[122] 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.

[123] S. Sinha, J.-M. Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. In *EDGE, Workshop on Edge Computing Using New Commodity Architectures*, volume 278, page 4321, 2006.

[124] I. Skrypnyk and D.G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *ACM/IEEE Int. Symp. on Mixed*

and Augmented Reality, ISMAR'04, pages 110– 119, Arlington, VA, November 2004.
[125] S. Smith and J.M. Brady. SUSAN - A new approach to low level image processing. *Int. Journal of Computer Vision*, 23(1):45–78, 1997.
[126] C. Stewart. Robust parameter estimation in computer vision. *SIAM Rev.*, 41:513–537, 1999.
[127] H. Strasdat, J.M.M. Montiel, and A. Davison. Real-time monocular slam: Why filter? In *Int. Conf. on Robotics and Automation, ICRA'10*, pages 2657–2664, Anchorage, USA, 2010.
[128] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. LDAHash: Improved matching with smaller descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(1):66–78, January 2012.
[129] I.E Sutherland. Three-dimensional data input by tablet. *Proceedings of the IEEE*, 62(4):453–461, April 1974.
[130] G. Takacs, V. Chandrasekhar, S. Tsai, D. Chen, R. Grzeszczuk, and B. Girod. Unified real-time tracking and recognition with rotation-invariant fast features. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 934–941, 2010.
[131] M. Tamaazousti, V. Gay-Bellile, S. Collette, S. Bourgeois, and M. Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2011*, pages 3073–3080, 2011.
[132] G. Taylor and L. Kleeman. Fusion of multimodal visual cues for model-based object tracking. In *Australasian Conference on Robotics and Automation (ACRA2003)*, Brisbane,Australia, December 2003.
[133] C. Teulière, L. Eck, E. Marchand, and N. Guenard. 3d model-based tracking for uav position control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'10*, pages 1084–1089, Taipei, Taiwan, October 2010.
[134] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment: A modern synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer Berlin Heidelberg, 2000.
[135] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit. Boosting binary keypoint descriptors. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2874–2881, 2013.
[136] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.
[137] T. Tykkälä, H. Hartikainen, A. I Comport, and J. Kämäräinen. Rgb-d tracking and reconstruction for tv broadcasts. In *VISAPP (2)*, pages 247–252, 2013.
[138] H. Uchiyama and E. Marchand. Toward augmenting everything: Detecting and tracking geometrical features on planar objects. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'11*, pages 17–25, Basel, Switzeland, October 2011.
[139] H. Uchiyama and H. Saito. Random dot markers. In *IEEE Virtual Reality Conference*, pages 35–38, 2011.
[140] 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.
[141] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offline information. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, October 2004.
[142] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.
[143] J. Ventura and T. Hollerer. Wide-area scene mapping for mobile visual tracking. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'12*, pages 3–12, 2012.
[144] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *IEEE/ACM Int. Symp. on Mixed and Augmented Reality*, pages 125–134, 2008.
[145] D. Wagner and D. Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Proc. of the 12th Computer Vision Winter Workshop*, St. Lambrecht, Austria, February 2007.
[146] G. Welch and E. Foxlin. Motion tracking: No silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications*, 22(6):24–38, 2002.
[147] H. Wuest and D. Stricker. Tracking of industrial objects by using cad models. *Journal of Virtual Reality and Broadcasting*, 4(1), April 2007.
[148] X. Yang and K.-T. Cheng. LDB: an ultra-fast feature for scalable augmented reality on mobile devices. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'12*, pages 49–57, 2012.
[149] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, November 2000.
[150] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *IEEE Int. Conf. on Computer Vision, ICCV'13*, pages 2344–2351, December 2013.
[151] F. Zhou, H.B.-L. Duh, and M. Billinghurst. Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. In *IEEE/ACM Int. Symp. on Mixed and Augmented Reality*, pages 193–202, 2008.

Eric Marchand is professor of computer science at Université de Rennes 1 in France and a member of the Inria/IRISA Lagadic team. He received a Ph.D degree and a "Habilitation Diriger des Recherches" in Computer Science from the Université de Rennes 1 in 1996 and 2004 respectively. He spent one year as a Postdoctoral Associates in the AI lab of the Dpt of Computer Science at Yale University. He has been an INRIA research scientist at INRIA Rennes-Bretagne Atlantique from 1997 to 2009. His research interests include robotics, visual servoing, real-time object tracking and augmented reality. He received a Best application paper award at IEEE IROS 2007 and Best paper runner-up award at IEEE ISMAR 2010 and IEEE 3DUI 2011. He has been an associate editor for IEEE Trans. on Robotics (2010-2014) and guest co-editor of The Int. J. of Robotics Research (IJRR) special issue on "Robot Vision" (april 2015). From june 2015, he is an associate editor for IEEE Robotics and Automation Letters (RA-L).

Hideaki Uchiyama received B.S., M.S. and Ph.D. degrees from Keio University, Japan, in 2006, 2007, and 2010, respectively. He was a postdoctoral fellow at INRIA Rennes, France, from 2010 to 2012, and a researcher at Toshiba Corporation, Japan, from 2012 to 2014. He is currently an assistant professor at Kyushu University, Japan. His research interests include augmented reality, computer vision and image processing. He was Local Arrangement Chair for IEEE ISMAR 2015.

Fabien Spindler graduated from ENI Brest engineer school (specialisation in electronics) in 1992, and received the Master of Science degree in Electronics and Aerospace Telecommunication from Supaero in Toulouse in 1993. Since 1994, he has been with Inria in Rennes as research engineer. He is in charge of the material and software management of several robotic experimentation platforms dedicated to researches in visual servoing. His interests include software engineering for the design of real-time computer vision and robotics applications. He is the software architect of the open-source ViSP (Visual Servoing Platform) library and is involved in the software transfer to industrial or academic partners.