3D SSD Tracking with Estimated 3D Planes

CRV Special Issue

Dana Cobzas [a] Martin Jagersand [a] Peter Sturm [b]

[a] *Computing Science, University of Alberta, Edmonton, T6G2E8, Canada*

[b] *INRIA Rhône-Alpes, Montbonnot 38334, France*

Abstract

We present a tracking method where full camera position and orientation is tracked from intensity differences in a video sequence. The camera pose is calculated based on 3D planes, and hence does not depend on point correspondences. The plane based formulation also allows additional constraints to be naturally added, e.g. perpendicularity between walls, floor and ceiling surfaces, co-planarity of wall surfaces etc. A particular feature of our method is that the full 3D pose change is directly computed from temporal image differences without making a commitment to a particular intermediate (e.g. 2D feature) representation. We experimentally compared our method with regular 2D SSD tracking and found it more robust and stable. This is due to 3D consistency being enforced even in the low level registration of image regions. This yields better results than first computing (and hence committing to) 2D image features and then from these compute 3D pose.

Key words: visual tracking, SSD tracking, image registration, plane tracking, 3D model

1 Introduction

In visual tracking the pose of an object or the camera motion is estimated over time based on image motion information. Some applications such as video surveillance only require that the target object is tracked in image space. For other applications such as augmented reality and robotics full 3D camera motion is needed. In this paper we concentrate on tracking full 3D pose.

One way to classify tracking methods is into feature-based and registration based. In feature-based approaches features in a (usually a priori) 3D model are matched with features in the current image. Commonly a feature detector is used to detect either special markers or natural image features. Pose estimation techniques can then be used to compute the camera position from the 2D-3D correspondences. Many approaches use image contours (edges or curves) that are matched with an a priori CAD model of the object [15,18,8]. Most systems compute pose parameters by linearizing with respect to object motion. A characteristic of these algorithms is that the feature detection is relatively decoupled from the pose computation, but sometimes past pose is used to limit search ranges, and the global model can be used to exclude feature mismatches [15,2].

In registration based tracking the pose computation is based on directly aligning a reference intensity patch with the current image to match each pixel intensity as closely as possible. These methods assume that the change in

Email addresses: dana@cs.ualberta.ca (Dana Cobzas), jag@cs.ualberta.ca (Martin Jagersand), Peter.Sturm@inrialpes.fr (Peter Sturm).

location and appearance of the target in consecutive frames is small. Image constancy can be exploited to derive efficient gradient based schemes using normalized correlation, or a sum-of-squared differences (e.g. L_2 norm) criterion, giving the technique its popular name SSD tracking. Unlike the feature-based approaches which build the definition of what is to be tracked into the low level routine (e.g. a line feature tracker tracks just lines), in registration based tracking any distinct pattern of intensity variation can be tracked. The first such methods required spatial image derivatives to be recomputed for each frame when "forward" warping the reference patch to fit the current image [16], while more recently, efficient "inverse" algorithms have been developed, which allow the real time tracking for the 6D affine [10] and 8D projective warp [3]. A more complicated appearance model can be used to compensate changes in intensity [10] or can be learned as a mixture of stable image structure and motion information [13]. A related approach [14,9], where instead of using spatial image derivatives, a linear basis of test image movements are used to explain the current frame, has proved equally efficient as the inverse methods during the tracking, but suffers from longer initialization times to compute the basis, and a heuristic choice of the particular test movements.

In this paper we extend the registration-based techniques by constraining the tracked regions to 3D planes. This will allow tracking full 3D camera position like in the featue-based approaches but eliminates the need for explicit feature matching. The update is based on the same SSD criterion as the classical registration-based methods with the difference that the update is done directly on the 3D parameters and not on the 2D warp parameters. The approach is thus different from previous approaches that first estimate the homography warp from salient points and then the 3D motion parameters from

the homography [19]. We do not assume any apriori model. Instead 3D plane parameters are estimated and optimized during the first ≈ 100 frames using structure-from-motion techniques. The algorithm does not require complete scene decomposition in planar facets, but works with few planar patches identified in the scene. Man-made environments usually contain planar structures (e.g. walls, doors). Some advantages of using a global 3D model and local surface patches are that only surfaces with salient intensity variations need to be processed, while the 3D model connects these together in a physically correct way. We show experimentally that this approach yields more stable and robust tracking than previous approaches, in which each surface patch motion is computed individually.

Previously [7] we have shown how 3D points estimated using structure-from-motion techniques can be used to constrain the SSD tracking. The current work focuses on planar regions constrained to lie on estimated 3D planes. We investigated two ways of defining the 3D planes, one using plane equation parameters (normal and distance) and the other using 4 control points. Related work of incorporating a 3D model into SSD tracking [22] calculates a 3D model from a 2D active appearance model (AMM) and use it to improve the tracking. Baker et al. [4] presented another related extension of the original Lucas-Kanade tracking algorithm applied to either 3D volumetric data (e.g CT, MRI data) or projection of 3D data in images.

The rest of the paper is organized as follows: we start with a short remonder of traditional 2D SSD tracking in Section 2 followed by the general description of our 3D SSD tracking approach in Section 3, followed by a presentation of the particular parametrization of the homography warp induced by a 3D plane in Section 4. Then Section 5 presents existing methods for estimating 3D planes

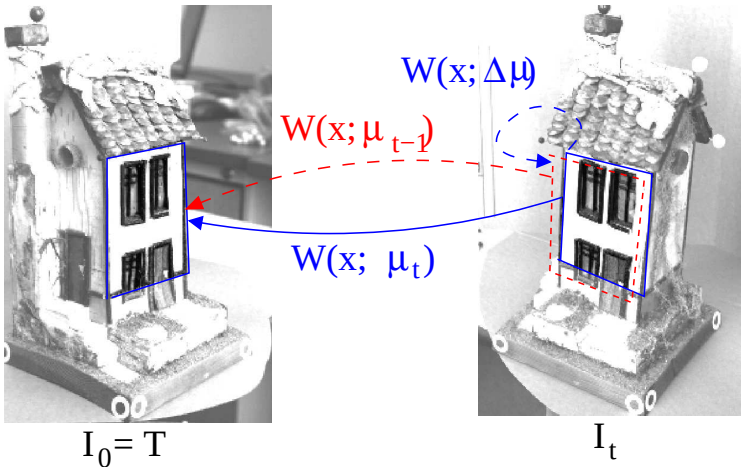

Fig. 1. Overview of the 2D SSD tracking. A 2D surface is related through a warp W between template space T and image space I_t. An incremental update in parameter space $\Delta\mu$ is computed at every step and added to the current warp.

from images. The complete tracking system is presented in Section 6 and its qualitative and quantitative evaluation in Section 7 followed by conclusions and a discussion in Section 8.

2 Background: 2D SSD Tracking

The goal of the SSD tracking algorithm, as originally formulated by Lucas-Kanade [16], is to find an image warp $W(\mathbf{x};\mu)$ that aligns a 2D template region $T(\mathbf{x})$ with the current image region $I(\mathbf{x})$ (see Figure 1). The warp is parametrized by a set of parameters μ and defines how a pixel $\mathbf{x}$ in template space maps to the location $W(\mathbf{x};\mu)$ in the space of image I. Commonly used warps include the translational (2 parameters) that is used in modeling optic flow, the affine warp (6 parameters) or the more complex homography warp (8 parameters).

Formally, under image constancy assumption (e.g. no illumination variation,

no occlusion) used in motion detection and tracking [12], the goal of the Lucas-Kanade algorithm is to find the warp (parameters μ) that minimize the error between the template and the image warped in the space of the template:

$$T(\mathbf{x}) = I_t(W(\mathbf{x}; \mu_t))) \tag{1}$$

The problem is efficiently solved iteratively by computing an incremental update $\Delta\mu$ for the parameters of the warp from frame I_{t-1} to I_t that is added to the current warp. The advantage of this formulation is that if the change $\Delta\mu$ is small the problem can be linearized. We write the update through function composition ("$\circ$") instead of say simple addition to allow a more general set of transforms. Mathematically $\mu_t = \mu_{t-1} \circ \Delta\mu$ can be obtained by minimizing the following objective function with respect to $\Delta\mu$:

$$\sum_x [T(\mathbf{x}) - I_t(W(\mathbf{x}; \mu_{t-1} \circ \Delta\mu)))]^2 \tag{2}$$

3 3D SSD Tracking problem formulation

We modified the 2D SSD tracking algorithm by constraining the motion of a set of Q patches in a sequence of images through a 3D rigid model $\mathcal{M}$ (refer to Figure 2). As a consequence the 2D motions of the image patches are defined as 2D warps $W(\mathbf{x}_k; \mu(P_t, \mathcal{M}))$ induced by the 3D motion P_t of the model $\mathcal{M}$.

The main differences in our approach compared to the 2D SSD tracking [16,3,10] are:

- We track full 3D camera position P_t instead of 2D warp parameters μ_t.
- The warp parameters $\mu(P_t, \mathcal{M})$ are defined by the model $\mathcal{M}$ and its 3D pose

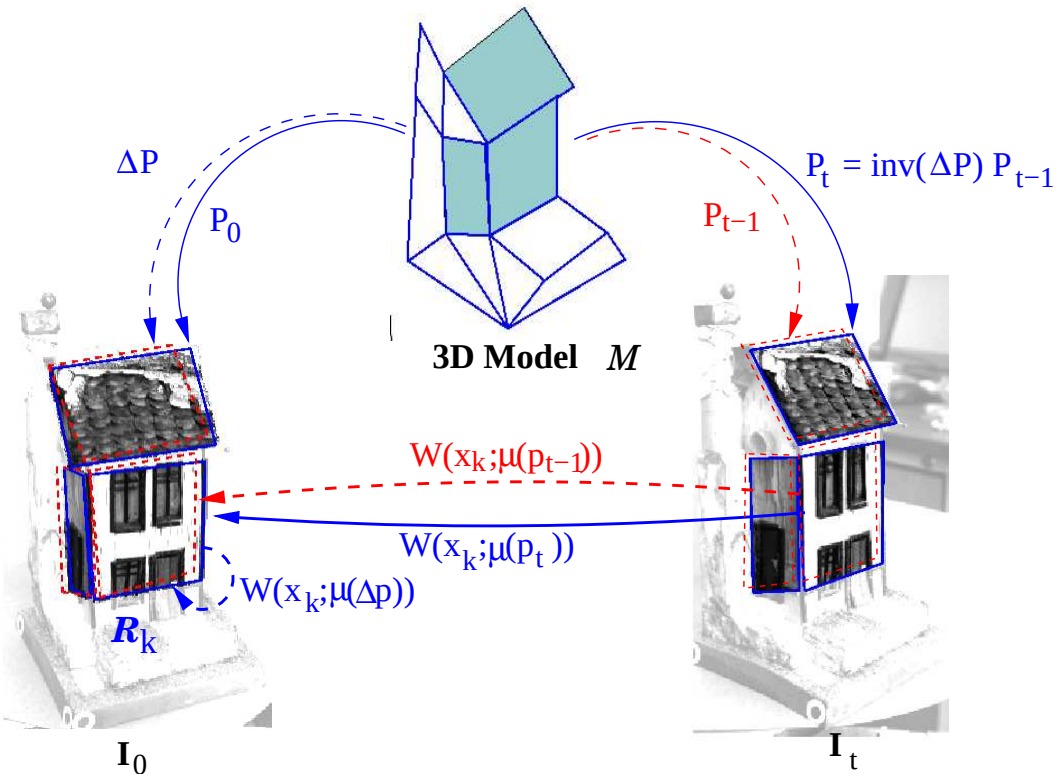

Fig. 2. Overview of the 2D-3D tracking system. In standard SSD tracking 2D surface patches are related through a warp W between frames. In our system a 3D model is estimated (from video alone), and a global 3D pose change ΔP is computed, and used to enforce a consistent update of all the surface warps.

P_t. Hence 2D warp parameters for the patches are no longer independent but a function of the 3D pose P_t. The model $\mathcal{M}$ is also estimated from video in a bootstrapping phase as described in Section 5.

- The algorithm tracks several regions unified by the same rigid motion through the model.

Compared to feature based tracking main difference is that our model directly influences (constrains) the parameters $\mu(P_t, \mathcal{M})$ of each 2D patch. In feature based approaches the positions of patches are first computed independently from local 2D image information, and only after these 2D alignments are fixed is the 3D pose computed.

- It computes the globally optimal 3D alignment P_t with respect to the chosen measure (normally sum of squared differences SSD, but other norms would be possible). This is not the case for feature based methods. Even if 2D SSD tracking is used to find the locally optimal parameters μ_k for each feature patch k, these parameters are not normally the same as those giving the globally optimal 3D alignment of all patches.
- By restricting movement of 2D patches to those consistent with a rigid 3D scene transform, individual feature trackers don't lose track even if an individual image signature is weak.

We first develop the general theory without committing to a particular 3D model or 2D warp. Later, in the next section, we present the algorithm for tracking planar patches constrained to lie on estimated 3D planes.

A model of the scene $\mathcal{M}$ is projected into the image space I_t through a projective transformation P_t (defined by its parameters $\mathbf{p}_t$). In the calibrated case, P_t is an Euclidean transformation $P_t = [R_t, \mathbf{t}_t]$, where $R = R_x(\alpha_x)R_y(\alpha_y)R_z(\alpha_z)$ represents the rotation matrix and $\mathbf{t} = [t_x, t_y, t_z]^T$ is the translation vector. Therefore $\mathbf{p}_t$ contains the rotation angles and the components of the translation $\mathbf{p} = [\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z]^T$.

Having defined a set of Q regions on the model, the goal of the 3D SSD tracking algorithm is to find the (camera) motion P_t (motion parameters $\mathbf{p}_t$ that best align the regions in the template space $\cup_k T(\mathbf{x}_k)$ with the regions in the current image $\cup_k I_t(\mathbf{x}_k)$. As we track 3D Euclidean motion we assume that the image pixels $\mathbf{x}_k$ have been normalized. As mentioned before, the 3D motion P_t of $\mathcal{M}$ induces a 2D warp for each patch denoted for convenience with $W(\mathbf{x}_k; \mu(\mathbf{p}_t))$. As an example, we look at the case of the motion of a planar patch. It is known

that the motion in image space induced by the 3D motion of a planar patch is perfectly modeled by a homography (2D projective transformation). We show in Section 4 that the 8 parameters of the homography that define the warp between two images can be calculated from the 3D parameters of the plane and the relative motion between the images. Note that the 3D model motion is global but each individual local region has a different 2D motion warp W_k.

Mathematically, we are looking for the set of 3D parameters $\mathbf{p}_t$ such as the image constancy assumption holds:

$$\cup_k T(\mathbf{x}_k) = \cup_k I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_t))) \tag{3}$$

As in the case of the 2D algorithm, the motion is computed as an incremental update $\Delta\mathbf{p}$ from frame I_{t-1} to I_t that is composed to the current motion $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}$ and can be obtained by minimizing the following objective function with respect to $\Delta\mathbf{p}$:

$$\sum_k \sum_x [T(\mathbf{x}_k) - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1} \circ \Delta\mathbf{p})))]^2 \tag{4}$$

The update in position $\Delta\mathbf{p}$ is based on the image difference between the template image and the current image warped in the space of the template, the update in position taking place on the side of the current image. As a consequence, the computations are performed in the space of the current image.

For efficiency, we solve the problem by an inverse compositional algorithm [3] that minimize the error between the template image and the current image warped in the space of the template image, with the update on the template image (see Equation 6). As shown below, working in the space of the template image, speeds up the tracking as more computations can be done only once

at the initialization. The goal then is to find $\Delta\mathbf{p}$ that minimizes:

$$\sum_k \sum_x [T(W(\mathbf{x}_k; \mu(\Delta\mathbf{p}))) - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})))]^2 \tag{5}$$

where in this case the 3D motion parameters are updated as:

$$P_t = \text{inv}(\Delta P) \circ P_{t-1} \tag{6}$$

where $\text{inv}(P) = [R^T | -R^T\mathbf{t}]$ for $P = [R|\mathbf{t}]$. As a consequence, if the 2D warp W is invertible, the individual warp update is (see Figure 2):

$$W(\mathbf{x}_k; \mu(\mathbf{p}_t)) = W(\mathbf{x}_k; \mu(\Delta\mathbf{p}))^{-1} \circ W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})) \tag{7}$$

For the numerical computation of $\Delta\mathbf{p}$ Equation 5 is linearizing through a Taylor expansion ($\Delta\mathbf{p}_t$ is represented as a column vector of motion parameters):

$$\sum_k \sum_x [T(W(\mathbf{x}_k; \mu(\mathbf{0}))) + \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}} \Delta\mathbf{p} - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})))] \tag{8}$$

As the motion of the template image is zero (the model is aligned with the template frame) $T = T(W(\mathbf{x}_k; \mu(\mathbf{0})))$. Denote the image derivatives by M

$$M = \sum_k \sum_x \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}} \tag{9}$$

and the error between the template regions and the corresponding current warped image regions by $\mathbf{e}_t$, flattened as a column vector on intensities :

$$\mathbf{e}_t = T - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1}))) \tag{10}$$

Equation 8 can be rewritten as an overdetermined linear equation system in

matrix form :

$$M\Delta\mathbf{p} \simeq \mathbf{e}_t \tag{11}$$

Where each column in M represents the spatial derivatives of a particular parameter p_i (i.e. the derivative images shown in Fig. 3 flattened into column vectors) and the motion $\Delta\mathbf{p}$ is computed as the least squares solution to Equation 11.

The image derivatives M are evaluated at the reference pose $\mathbf{p} = \mathbf{0}$ and they are constant across iterations and can be precomputed, resulting in an efficient tracking algorithm that can run in real time (see Section 6). Figure 3 shows examples for the 6 derivative images corresponding to the three rotation angles and three components of the translation. Note that several of the derivative images for one patch look perceptually similar. While they are linearly independent, they are not very well separated. For instance image plane translation and translation along the optic axis will be similar if the image patch used is wholly on one side of the optic axis – see the rightmost two images. Hence tracking of several DOF from one planar patch as in 2D SSD tracking is relatively ill-conditioned (more on this in the experiments section). However in 3D tracking the combination of several planar patches gives a well conditioned problem.

4 Homography induced by a plane

The proposed tracking algorithm is based on the assumption that the motion in image space induced by the motion of the model regions can be expressed

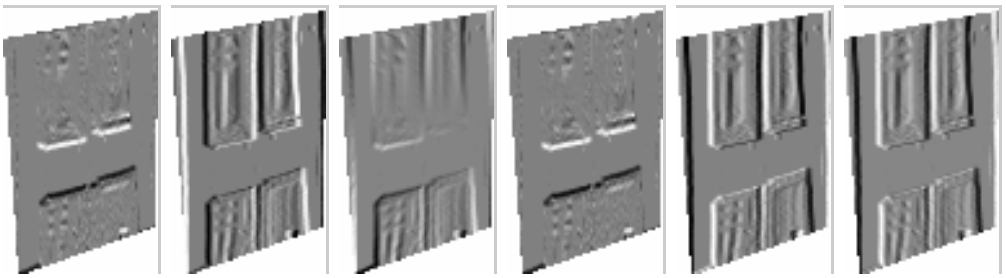

Fig. 3. Examples of derivative images M for the image patch on the front of the house in Fig. 2. The images from left to right corresponding to the three rotation angles and three components of the translation.

as a 2D warp $W(\mathbf{x}_k; \mu(\mathbf{p}_t))$ that depends on the model $\mathcal{M}$ and the current motion parameters $\mathbf{p}_t$. We now show, for the case of a planar patch, the explicit formulation of the homography warp function of plane model and its relative position to the camera. We consider two cases, one where the plane is defined using the normal and distance to origin and the other when the plane is defined using four control points. The second case can be reduced to the first one but numerically different as it enforces stronger constraints.

4.1 Parametrized plane

It is well known that images of points on a plane in two views are related by a homography [11]. In general a homography warp H has 8 independent parameters represented as a vector μ. For Euclidean planes in general positions H is uniquely determined by the plane equation and thus there are only 6DOF in H. A 3D plane is represented as $\pi = [\mathbf{n}^T, d]$, where $\mathbf{n}$ is the unit normal and d is the signed distance from the origin to the plane. For points $\mathbf{X}$ on the plane, $\mathbf{n}^T\mathbf{X} + d = 0$. If the world coordinate system is aligned with the first

camera coordinate system, the calibrated projection matrices have the form:

$$P_0 = K[I|\mathbf{0}] \quad P_t = K[R|\mathbf{t}] \tag{12}$$

where K is the camera matrix (internal parameters) and $R, \mathbf{t}$ represents the 3D motion of the second camera with respect to the first one. As mentioned before, there are six motion parameters in $\mathbf{p}$ that determine R and $\mathbf{t}$: the three angles of the general rotation and the three components of the translation.

The homography induced by the plane π has the form:

$$H = K(R - \frac{\mathbf{t}}{d}\mathbf{n}^T)K^{-1} \tag{13}$$

Image points in the two views I_1, I_2 are then related by $\mathbf{u}_2 = H\mathbf{u}_1$. If the image points are normalized with respect to camera internal parameters $\mathbf{x} = K^{-1}\mathbf{u} = [R|\mathbf{t}]\mathbf{X}$ the homography becomes:

$$H = R - \frac{\mathbf{t}}{d}\mathbf{n}^T \tag{14}$$

Using the notation from the previous section, the 2D warp W induced by the motion $\mathbf{p}$ then has the form:

$$W(\mathbf{x}; \mu(\mathbf{p})) = H\mathbf{x} = (R - \frac{\mathbf{t}}{d}\mathbf{n}^T)\mathbf{x} \tag{15}$$

The explicit dependency for the warp parameters μ function of $\mathbf{p}$ and $\pi = [\mathbf{n}^T, d]$ can be calculated from Equation 14, but this is not necessary as the image derivatives M can be calculated directly using the warp derivatives $\frac{\partial W}{\partial \mathbf{p}}$ obtained from the derivation of Equation 14.

4.2 Plane defined through four control points

In a projective space, a plane can be defined using four control points $\mathbf{Y}_j,\ j = 1 \cdots 4$. Denote the projection of the control points in the current image by $\mathbf{y}_j = K[R|\mathbf{t}]Y_j$, where R and $\mathbf{t}$ represents the motion of the camera for the current image relative to the template frame. The parameters μ of the homography H between the reference view and the current view are determined by the projection of the control points in the two views. This explicit dependency of the 2D warp parameters μ as function of 3D motion parameters $\mathbf{p}$ and model 3D points $\mathbf{Y}_j$ is difficult to obtain analytically. Instead we calculate the $\frac{\partial \mu}{\partial \mathbf{p}}$ terms required by the image derivatives M using the implicit function theorem in the warp equation

$$\mathbf{y}_{0j} = H\mathbf{y}_j = HK[R|\mathbf{t}]Y_j \quad j = 1, N \tag{16}$$

where $\mathbf{y}_{0j} = KY_j$ represents the projection of the control points in the first frame and H is parametrized as [1]

$$H = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 \\ \mu_4 & \mu_5 & \mu_6 \\ \mu_7 & \mu_8 & 1 \end{bmatrix} \tag{17}$$

[1] In the current parametrization of the homography warp we set the last value from the 3×3 matrix $\mu_9 = 1$ (fixing the scale) which does not allow this value to be 0. In the present case this is not a limitation since all points on the tracked patch remain finite ($\mu_9 = 0$ when the image center is mapped to infinity).

Equation 16 can be rewritten in the form

$$A(\mathbf{p})\mu = B(\mathbf{p}) \tag{18}$$

with

$$A(\mathbf{p}) = \begin{bmatrix} y_1^1 & y_1^2 & 1 & 0 & 0 & 0 & -y_1^1 y_{01}^1 - y_1^2 y_{01}^1 \\ 0 & 0 & 0 & y_1^1 & y_1^2 & 1 & -y_1^1 y_{01}^2 - y_1^2 y_{01}^2 \\ \vdots & & & & & & \\ y_N^1 & y_N^2 & 1 & 0 & 0 & 0 & -y_N^1 y_{0N}^1 - y_N^2 y_{0N}^1 \\ 0 & 0 & 0 & y_N^1 & y_N^2 & 1 & -y_N^1 y_{0N}^2 - y_N^2 y_{0N}^2 \end{bmatrix} \tag{19}$$

$$B(\mathbf{p}) = [y_{01}^1, y_{01}^2, \ldots, y_{0N}^1, y_{0N}^2]^T \tag{20}$$

where $[y_j^1, y_j^2, 1]^T$ are the normalized homogeneous coordinates for $\mathbf{y}_j$.

Taking the derivatives in Equation 18 with respect to each component p of $\mathbf{p}$ we obtain:

$$\frac{\partial A}{\partial p}\mu + A\frac{\partial \mu}{\partial p} = \frac{\partial B}{\partial p} \tag{21}$$

For a given $\mathbf{p}$ value μ can now be linearly computed from Equation 18 and then $\frac{\partial \mu}{\partial p}$ is computed from Equation 21.

5 Estimating the structure

The 3D model used to constrain the tracking is estimated from images in a bootstrapping phase using structure from motion (SFM). SFM methods use corresponding features in several images to estimate both their structure and the camera positions. For getting correspondences, we track salient feature points using standard (2D image-plane) SSD trackers as in [3,10]. We next present a method for estimating parameters of 3D planes followed by a method that reconstructs 3D points. The resulting models are matching the models required for tracking planar regions using homographies as described in Sections 4.1 and 4.2.

5.1 Estimating plane equations from images

The plane equations are estimated from tracked feature points on each plane. The grouping of the points into planes is done in the initialization phase by having the user mark planar regions in the first frame. (Since SSD tracking doesn't depend on a particular identifiable feature type it is commonly initialized manually to particular image regions.)

We first present the algorithm that computes a plane equation from two images. It is a special case of the structure from motion problem where the camera is internally calibrated and the feature points belong to a physical plane. The homography H induced by the plane is robustly computed using RANSAC from 4 or more corresponding points. Having H of the form $H = R - \mathbf{t}\mathbf{n}^T/d$ it can be decomposed into the motion and structure parameters $\{R, \frac{1}{d}\mathbf{t}, \mathbf{n}\}$ [17]. There are in general four solutions but only at most two are physically valid

by imposing the positive depth constraint (model points are in front of the camera). We disambiguate between the two remaining solutions using multiple frames.

In a more general case, when multiple planes are viewed in multiple images, a reference view is chosen and the corresponding plane homographies that relate the reference view with additional views are computed. To reduce the solution to the decomposition of H to one, we assume a smooth position change between adjacent views and only the solution that corresponds to the motion closest to the previous one is chosen. For the first pair one of the two physically valid solutions is chosen. The scale of the scene is also disambiguated by fixing the distance to one plane. The global motion $R, \mathbf{t}$ for each frame is averaged over the individual motions estimated from each plane homography and the plane parameters are averaged over the ones computed from several views. At the end a nonlinear optimization using Levenberg-Marquardt algorithm over all the frames is performed which locally minimizes the symmetric transfer error for points related through a homography:

$$\begin{aligned} &\{R_2, \mathbf{t}_2, \ldots R_m, \mathbf{t}_m; \mathbf{n}_1, d_1, \ldots, \mathbf{n}_k, d_k\} = \\ &\text{argmin} \sum_t \sum_k \sum_{\mathbf{x}_{tk}} d^2(\mathbf{x}_{tk}, H_{tk}\mathbf{x}_{1k}) + d^2(\mathbf{x}_{1k}, H_{tk}^{-1}\mathbf{x}_{tk}) \end{aligned} \tag{22}$$

This is close to but not exactly the maximum likelihood estimator under the Gaussian noise assumption. It is nevertheless more practical in our case as it will give the best motion and plane structure without explicitly computing the 3D point coordinates.

5.2 Incorporating constraints between planes

Known constraints between planes such as perpendicularity or parallelism of walls that naturally appear in man-made environments result in better structure and can potentially further stabilize the tracking. We impose constraints by a minimum parametrization of the plane parameters as in [5].

Consider two planes $\pi_1 = [\mathbf{n}_1^T, d_1]$,$\pi_2 = [\mathbf{n}_2^T, d_2]$. A perpendicularity constraint can be algebraically expressed by a vanishing dot product between the plane normals:

$$n_{11}n_{21} + n_{12}n_{22} + n_{13}n_{23} = 0 \tag{23}$$

This bilinear constraint is enforced by eliminating one plane parameter. We chose to eliminate the parameter n_{ik} such that the absolute value of the corresponding parameter on the second plane n_{jk} is maximal over all the parameters.

For the other type of constraint when the planes are parallel we impose that the normals of the two planes are the same. This eliminates all parameters that represent the unit normal of one plane.

$$n_{1k} = n_{2k}, k = 1, 2, 3 \tag{24}$$

The resulting plane parameters and the originally recovered motions are then optimized as before using Equation 22. A full parametrization of the planes is recovered for every plane from Equations 23,24. A potentially somewhat more accurate approach would involve obtaining a minimal parameterization of 3D points on constrained planes and estimating the structure of those points and

the camera motion from feature correspondences. This would allow defining a maximum likelihood estimator under Gaussian image noise. The plane parameters are then computed from the estimated 3D points.

5.2.1 Estimating 3D points

For estimating the 3D positions of the control points that define planar regions (as described in Section 4.2) we use standard SFM techniques and the stratified uncalibrated approach [11] (projective reconstruction that is upgraded to a Euclidean structure using automatic self-calibration). There are several well known estimation algorithms to recover the projective structure and motion of a scene using the fundamental matrix (2 views), the trilinear tensor (3 views) or multi view tensors for more than 3 views. In our system we used the method developed by Werner et al. [21] that estimates the trilinear tensors for triplets of views and then recovers epipoles from adjoining tensors. The projection matrices are computed at once using the recovered epipoles. New views are integrated through the trilinear tensor between the new and two previous views. Assuming that the cameras have zero skew and aspect ratio ($a_u = a_v$ and $s = 0$) and the principal point (u_c,v_c) is approximately known, the Euclidean projections are recovered using self-calibration [20]. There is still an absolute scale ambiguity that cannot be recovered without additional metric scene measurements, but since this scale remains fixed over a video sequence, we can use a 6DOF Euclidean motion model for tracking the motion between frames.

An important consideration for the 3D modeling used to bootstrap the 3D tracking is that the model is accurate enough to make the 3D tracking con-

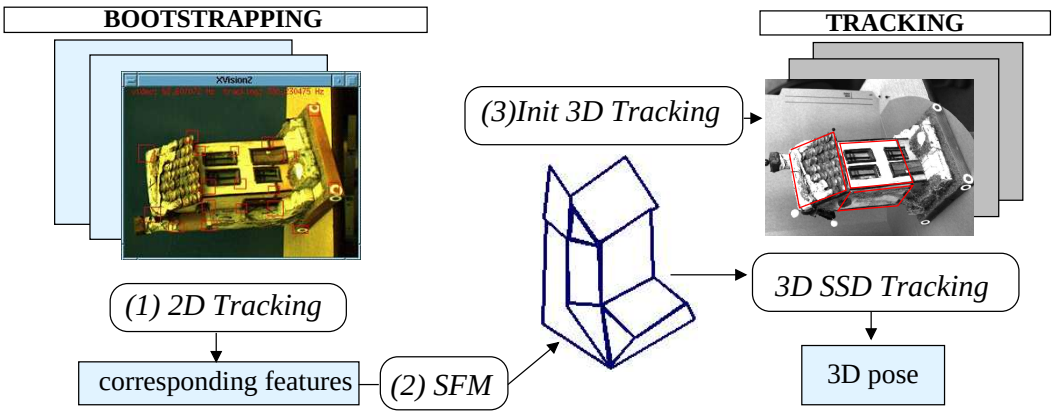

Fig. 4. Overview of the tracking system and phases. To bootstrap 3D tracking, first regular 2D SSD tracking is used for an initial 100 frames to calculate a 3D model. Then the 2D patches are integrated into the 3D tracking which directly computes 3D pose updates from temporal intensity variation.

verge. In another paper [6] where we compared the accuracy of the SFM algorithms for different geometries (affine, projective, Euclidean), we found that the model obtained from a scene can be reprojected into new (different from the training) views with a reprojection accuracy of about 1-3 pixels (if bundle adjusted). This accuracy of the reconstructed model is in the convergence range for the tracking algorithm.

6 Tracking system overview

We incorporated the proposed plane tracking algorithm into a system that first initializes the model (plane equations or 3D points) from 2D image tracking over a limited motion and then switches to track points on the estimated 3D planes. Refer to Figure 4.

6.0.2 Bootstrapping phase

(1) *2D SSD tracking:* The user marks planar regions in the first frame and specifies plane constraints (parallelism, perpendicularity) as applicable. Feature points inside these regions are tracked using standard SSD 2D trackers.

(2) *Model computation:* For the first model described in Section 4.1 plane parameters $\pi = [\mathbf{n}^T, d]$ are first initialized by averaging close form solutions from homographies H that relate points on planes from a frame to the reference (first) frame. Then a minimal parametrization is optimized together with the estimated motion over all the training frames as described in Section 5.1.

For the model described in Section 4.2 the 3D coordinates of plane's control points $\mathbf{Y}_j$ are estimated using projective structure from motion and self-calibration (Section 5.2.1).

(3) *Initialize 3D tracking:* The 3D planes are related to the current frame using the 2D tracked points. This will align the origin of the world coordinate system with the current frame (initial camera is $[I|\mathbf{0}]$). Then the 3D plane based tracking is initialized by computing the derivative images M (Equation 9).

6.1 Tracking phase

The tracking now continues in 3D with the 2D surface patches integrated into the 3D model of the planes. This enforces a globally consistent motion for all surface patches as described in Section 3.

(1) An incremental position update $\Delta\mathbf{p}$ is computed based on image differences between the regions in the reference template and the warped regions from the current image (Equation 11).

(2) The global camera position P_t is updated based on Equation 6.

7 Experimental results

Two important properties of tracking methods are convergence and accuracy. Tracking algorithms based on optimization and spatio-temporal derivatives (Equation 8) can fail to converge because the image difference between consecutive frames I_{t-1}, I_t is too large (more than just few pixels), and the first order Taylor expansion (Equation 8) around $\mathbf{p}_{t-1}$is no longer valid, or some disturbance causes the image constancy assumption to be invalid.

In the numerical optimization the pose update $\Delta\mathbf{p}$ is computed by solving an overdetermined equation system, Equation 11. Each pixel in a tracking patch provides one equation and each model freedom (DOF) one variable. The condition number of the linearized motion model M affects how measurement errors propagate into $\Delta\mathbf{p}$, and ultimately if the computation converges or not. In general, it is more difficult to track many DOF. In particular, the homography warp (that incorporates scaling and out-of-plane rotations) causes less apparent image change compared to a 2D translational warp. By tracking a connected 3D model, the tracking convergence is no longer solely dependent on one surface patch alone, and the combination of differently located and oriented patches can give an accurate 3D pose estimate even when each patch would be difficult to track individually.

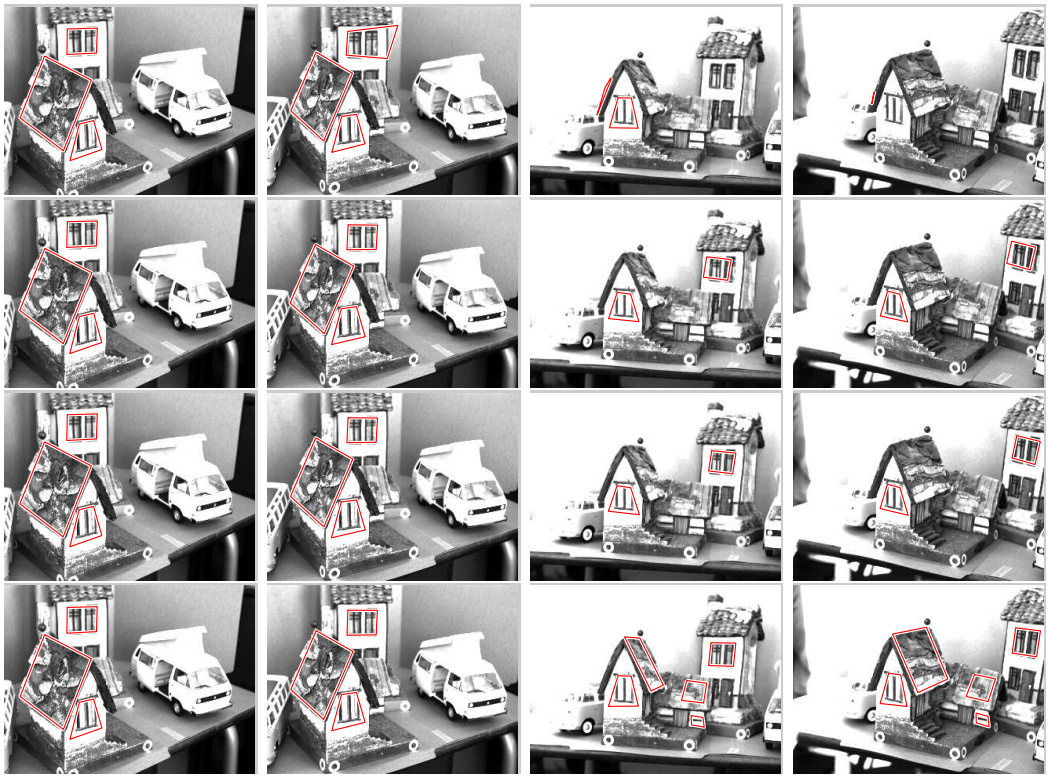

Fig. 5. *EXP1*: **First row** Tracking individual patches using conventional 2D image SSD [3]. The individual patch trackers lose track of several regions during the test sequence. **Second row** Through the 3D plane equations (model from Section 4.1) each region motion is rigidly related to the model, and tracking succeeds through the whole sequence but one of the trackers starts drifting (slightly turning within the constraint plane) in the second half of the sequence. **Third row** A better model obtained by imposing constraints between planes improves the tracking. **Forth row** The best results are obtained by controlling the planar regions with four model points (model from Section 4.2). This prevents the drifting observed in row two. In this case, a visibility test was performed to introduce/remove patches as they go in/out of view. (Results are also shown in on-line `video1` and `video5` left [1])

In the first experiment (*EXP1*) we compared the tracking stability for the 6DOF plane based tracker (with constraints through plane equations: `SSD+PL`, planes through 3D points: `SSD+3DPT` proposed models from Section 4) and the most general form of traditional 2D image-plane SSD tracking based on an

8DOF homography warp (`SSD`) [3]. We also tested the influence of plane constraints on tracking `SSD+PLCT` (e.g. roof planes perpendicular to front plane) introduced as described in Section 5.2. The results are shown in Figure 5 (above) and summarized in Table 1 (also on-line `video1` and `video5` left [1]). We found that all of the 3D model based algorithms that track 6DOF global pose perform better than the original 2D image plane SSD tracking that individually tracks an 8DOF homography for each region. The better stability is indicated also by the difference in condition numbers of M that vary between $4 * 10^6$ and $1 * 10^7$ for the 2D image-plane SSD tracker (indicating an ill conditioned problem) and drops to an order of 10^3 for the 3D model based trackers. Qualitatively, we observe that the tracker on the tall house is lost at about frame 70 for the `SSD` while the first tracker starts drifting only in frame 340 for the `SSD+PL`. The constraints have generally small influence but nevertheless we noticed a better performance for `SSD+PLCT` vs. `SSD+PL` (drift appears later). The last method `SSD+3DPT` performs the best and is able to track the whole 512 frames without problems. This is due to the fact that stronger constraints are imposed to the region when it is defined through four points compared to when it is defined as part of a plane. In the first case all degrees of freedom are defined whereas in the last case the region still has 2DOF as it can drift on the defined plane. The last model also allows detection and removal of occluded regions and introduction of new regions using a Z-buffer algorithm. In the case when the regions are constrained by parametrized planes, the removal/addition of regions cannot be performed as the corresponding 3D region on the planes is unknown.

One of the main advantages of the proposed approach over traditional SSD tracking is that true Euclidean 3D camera pose can be directly tracked. This

is useful for example in robotics or augmented reality applications. In the next experiment we evaluate the accuracy of 3D tracking in an indoor lab scene tracked by a moving camera. Ground truth was obtained by measuring the camera path and performing a Euclidean calibration of the model. Figure 6 shows two tracked frames of the sequence and the sequence can be seen in `video4` [1].

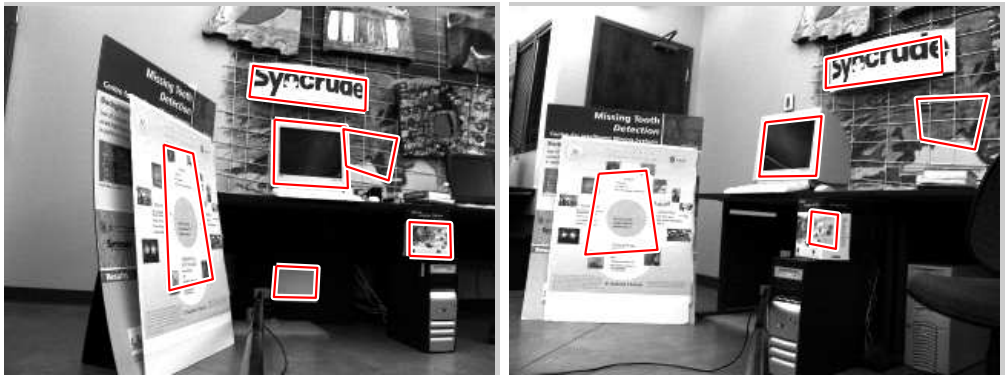

Fig. 6. Tracking 3D planes. Pose accuracy experiment (*EXP2a* and *EXP2b*). `video4` [1]

The first test trajectory (*EXP2a*) is a straight line in the horizontal plane of 1m. Figure 7 (left) illustrates the recovered trajectory. To measure the accuracy of the tracking algorithm we calibrated the 3D model for the planes assuming given real dimensions (distance from camera to one plane) so we could get the translation in meters. Here the parallelism constraints imposed between planes (e.g. back wall and Syncrude sign) had a very small influence on the pose accuracy. The results are displayed in Table 1. The motion in the second trajectory (*EXP2b*) was along two perpendicular lines in the horizontal plane. In this experiment, the real physical motion was not particularly smooth and the recorded image data therefore also somewhat jumpy.

We measured two types of errors, ones that are relative measurements between projective geometric entities (e.g. deviation from lines, planes) and others

Model	*STABILITY EXP1*		*ACCURACY EXP2a*			*ACCURACY EXP2b*		
	Failure frame	*Cond. num.*	*Dev. plane (cm)*	*Dev. line (cm)*	*Err. length*	*Dev. plane (cm)*	*Dev. line (cm)*	*Err. angle*
`SSD` [3]	70	10^7						
`SSD+PL`	240	10^3	0.025	2.42	15%	0.79	1.47	15%
`SSD+PLCT`	340	10^3	0.023	2.36	13%	0.66	1.35	15%
`SSD+3DPT`	-	10^3	0.02	0.95	8%	0.56	0.99	9%

Table 1

Comparison for the stability (*EXP1*) and accuracy (*EXP2a* and *EXP2b*) of our 3D SSD tracking algorithm: `SSD+PL` uses estimated plane parameters; `SSD+PLCT` uses estimated constrained planes; `SSD+3DPT` uses planes defined through 4 control points. For the stability of the trackers we compared our approach with the conventional 2D image-plane SSD as in [3]

that are related to calibrated Euclidean measurements (e.g. error in distance). The relative measurements were quite good (on the average less than 0.5 cm deviation from plane and about 1.5 cm deviation from straight line). But the error in measure length was $8 - 15\%$ and the error in the angle between the two lines fitted to the recovered positions (in the second experiment) was also relatively large.

The experiments show that the accuracy of the measurements connected to properties that are not directly related to calibrated properties of the structure is higher than the accuracy in measured distances. This is due to the difficulty in making calibrated (Euclidean) measurements from an initially uncalibrated

(projective) camera.

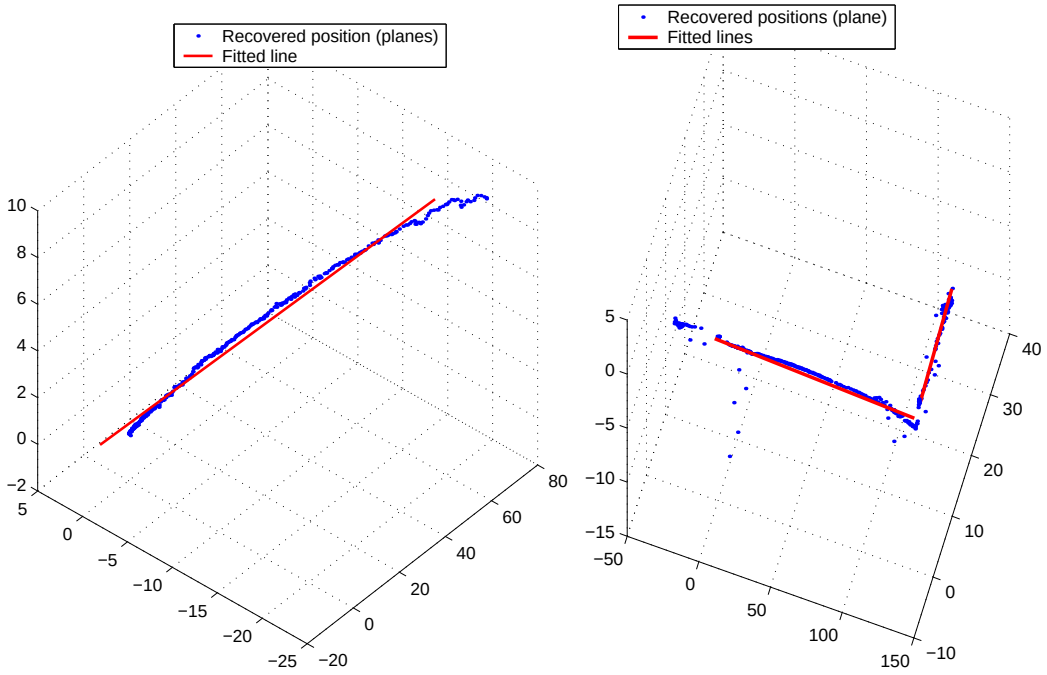

Fig. 7. Recovered positions (in 3D space) for the straight line trajectory *EXP2a*(left) and the 2 perpendicular lines trajectory *EXP 2b*(left). The red line are the fitted 3D lines to each line segment.

8 Discussion

We have presented a tracking algorithm that extends the existing SSD homography tracking by computing a global 3D position based on precomputed plane equations. The parameters of the 3D planes are estimated from an initial sequence (about 100 frames) where feature points on the planes are tracked using regular SSD translational trackers. Constraints between planes are also incorporated using a minimal parametrization of the planes. We showed that the proposed tracking algorithm is more stable due to the reduced DOF compared to tracking individual homographies and can handle a large range of motion.

A main advantage of the method is that it tracks full 3D camera position that might be required in applications like robotics or augmented reality. The pose is computed directly from image derivatives with respect to pose parameters that guarantees the best 3D pose update from the linearized model. This is unlike the other model based approaches where 3D pose is estimated from tracked 2D image correspondences.

We like to clarify the difference between our approach and a related method publish by Baker et. al [4] that is also tracking 3D data through its projection in images. The main difference is that they track 3D points whereas in our case the tracking is performed in 2D but the warp is implicitly constrained by a 3D model. The consequence is that in their case the inverse compositional algorithm is not valid but in our case it is.

The present version of the algorithm does not handle partial occlusions and illumination variation. This problem can be solved by using a robust norm like in [10].

References

[1] On-line mpeg movies of the experiments are available. See `videoX` at http://www.cs.ualberta.ca/∼dana/Movies/tracking/.

[2] M. Armstrong and A. Zisserman. Robust object tracking. In *Second Asian Conference on Computer Vision*, pages 58–62, 1995.

[3] S. Baker and I. Matthews. *Lucas-Kanade 20 Years On: A Unifying Framework*. Technical Report CMU-RITR02-16, 2002.

[4] S. Baker, R. Patil, K. Cheung, and I. Matthews. *Lucas-Kanade 20 Years On:*

Part 5. Technical Report CMU-RI-TR-04-64, 2004.

[5] A. Bartoli and P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *IJCV - International Journal of Computer Vision*, 52(1):45–64, 2003.

[6] D. Cobzas and M. Jagersand. A comparison of viewing geometries for augmented reality. In *Proc. of Scandinavian Conference on Image Analysis (SCIA 2003)*, 2003.

[7] D. Cobzas and M. Jagersand. 3d ssd tracking from uncalibrated video. In *ECCV 2004 Workshop on Spatial Coherence for Visual Motion Analysis (SCVMA)*, 2004.

[8] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *PAMI*, 24(7):932–946, July 2002.

[9] M. Gleicher. Projective registration with difference decomposition. In *CVPR97*, pages 331–337, 1997.

[10] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *PAMI*, 20(10):1025–1039, October 1998.

[11] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[12] B. Horn. *Computer Vision*. MIT Press, Cambridge, Mass., 1986.

[13] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *PAMI*, 25(10):1296–1311, 2003.

[14] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *PAMI*, 24(7):996–1000, July 2002.

[15] D. Lowe. Fitting parameterized three-dimensional models to images. *PAMI*, 13(5):441–450, May 1991.

[16] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Int. Joint Conf. on Artificial Intelligence*, 1981.

[17] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3D Vision*. Springer, 2004.

[18] E. Marchand, P. Bouthemy, and F. Chaumette. A 2d-3d model-based approach to real-time visual tracking. *IVC*, 19(13):941–955, November 2001.

[19] G. Simon, A. W. Fitzgibbon, and A. Zisserman. Markerless tracking using planar structures in the scene. In *IEEE and ACM International Symposium on Augmented Reality (ISAR)*, 2000.

[20] W. Triggs. Auto-calibration and the absolute quadric. In *CVRP*, pages 609–614, 1997.

[21] T.Werner, T.Pajdla, and M.Urban. Practice of 3d reconstruction from multiple uncalibrated unorganized images. In *Czech Pattern Recognition Workshop*, 2000.

[22] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2d+3d active appearance models. In *Proc. of International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.