3D SSD Tracking from Uncalibrated Video

Dana Cobzas and Martin Jagersand

Computer Science, University of Alberta, Canada

Abstract

In registration-based motion tracking precise pose between a reference template and the current images is determined by warping image patches into the template coordinates and matching pixel-wise intensities. Efficient such algorithms are based on relating spatial and temporal derivatives using numerical optimization algorithms. We extend this approach from planar patches into a formulation where the 3D geometry of a scene is both estimated from uncalibrated video and used in the tracking of the same video sequence. Our tracking algorithm is different than traditional SSD tracking as it trackers a 3D pose global to all patches and not individual 2D image warps. Experimentally we compare convergence and accuracy of our uncalibrated 3D tracking to previous approaches. Notably, the 3D algorithm can successfully track over significantly larger pose changes than ones using only 2D planar regions. It also allows for the detection of occlusions and removal/insertion of tracking regions as appropriate in response.

Index Terms

visual tracking, image registration, 3D model

I. INTRODUCTION

In visual tracking motion information from a video sequence is distilled and unified to determine pose parameters of a moving camera or object. One way of classifying tracking methods is into feature based, segmentation based and registration based.

In feature based tracking a feature detector is used to locate the image projection of either special markers or natural image features. Then a 3D pose computation can be done by relating 2D image feature positions with their 3D model. Many approaches use image contours (edges or curves) that are matched with an a-priori given CAD model of the object [7], [15], [18]. 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, other than sometimes past pose is used to limit search ranges, and the global model can be used to exclude feature mismatches [2], [15].

In segmentation based tracking some pixel or area based property (e.g. color, texture) is used to binarize an image. Then the centroid and possibly higher moments of connected regions are computed. While the centroid and moments are sufficient to measure 2D image properties, it

is typically not used for precise 3D tracking alone, but can be used to initialize more precise tracking modalities [22].

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. Often a sum-of-squared differences (e.g. L_2 norm) error is minimized, giving the technique its popular name SSD tracking. Unlike the two previous approaches which builds 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. Since the type of target (or feature) is not pre-defined, typically the user points to desired patches in the first frame. This technique can also be used in image alignment to create mosaics [21]. Early approaches used brute force search by correlating a reference image patch with the current image. While this works reasonably well for 2D translational models, it would be unpractical for planar affine and projective (homography) image transforms. Instead, modern methods are based on numerical optimization, where a search direction is obtained from image derivatives. 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 for real time tracking of the above mentioned 6D affine [9] and 8D projective warp [3]. An appearance model can be used to compensate changes in intensity [9] or can be learned as a mixture of stable image structure and motion information [5], [12]. In a recent paper, Lee and Kriegman [14] extend Black and Jepson's idea 'eigentracking' by incrementally learning an appearance manifold that is approximated with a collection of submanifolds and the connectivity between them. Benhimane and Malis [4] proposed a SSD tracking algorithm based on a second order minimization method (the ESM method [17]) that has a high converge rate like the Newton method but does not require the computation of the Hessian. A related approach [8], [13], where instead of using spatial image derivatives, a linear basis of test image movements are used to explain the current frame, has proven equally efficient as the inverse methods during the tracking, but suffer from much longer initialization times to compute the basis, and a heuristic choice of the particular test movements.

The extension of 2D SSD tracking to 3D has been recently addressed. Baker et. al [25] calculates a 3D model from a 2D active appearance model (AMM) and use it to improve the tracking. The main difference in our approach is that we use an estimated rigid Euclidean model

so we can track 3D camera position while their approach tracks a deformable model and does not estimate the camera pose. Baker et al. [3] 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 difference compared to our approach is that they track points on a 3D surface and therefore the 3D-2D inverse compositional algorithm is not valid. In our case the tracking is performed in 2D but the warp is implicitly constrained by a 3D model (using control points). So the inversion is valid as the warp is ultimately performed on 2D image points.

In this paper we extend the registration based technique by constraining the tracked regions with a 3D rigid scene model, estimated from the same uncalibrated video. The algorithm is thus able to track full 3D camera position like in the model-based approaches, but eliminates the need for explicit feature matching. The update is based on the same SSD error as the classical registration-based techniques with the difference that the update is done directly on the 3D parameters and not on the 2D warp parameters.

Our method starts by tracking the image motion of several surface patches using conventional SSD tracking. In this initialization phase the motion is relatively restrictive to ensure convergence for the 2D trackers. After some time (typically ≈ 100 frames) a 3D model is computed using uncalibrated structure-from-motion (SFM), and from this point the system switches to full 3D tracking of camera rotation and translation using the estimated 3D model. The improved stability of the 3D tracking allows larger motions to be performed for the rest of the sequence. The algorithm does not require complete scene decomposition in planar facets, but works with few planar patches identified in the scene. Experiments prove furthermore that the algorithm is also quite robust to patches not being perfectly planar. One advantage of using a global 3D model imposed on 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 traditional SSD tracking, where each surface patch motion is computed individually.

To summarize, our main contributions presented in contrast with previous work are:

- **SSD tracking:** We propose an image registration algorithm that track full 3D camera position using an estimated 3D model, instead of tracking 2D warp parameters. In our case the 2D warp parameters are defined as a function of control points on the model and

the current camera pose. Hence the 2D warp parameters are no longer independent but unified by the same rigid motion through the model.

- **Model-based tracking:** Compared to feature-based tracking approaches, our method computes an optimal 3D alignment with respect to the chosen measure (sum of square differences SSD) in image space. This is not the case for feature-based methods (or in the so-called structure from motion - SFM - algorithms) where the feature matching (tracking) is decoupled from the camera position estimation. Therefore, even though in feature-based methods individual patches are tracked using traditional 2D SSD tracking, the locally estimated 2D parameters are in general not the same as the ones giving one optimal global 3D alignment. One other advantage of the global registration algorithm is that individual features don't loose track even if one image signature is weak. This happens because the restricted motions of the 2D patches are constrained to motions that are consistent with the 3D model.
- **Structure-from-motion:** The 3D model that constrains the tracking is estimating using SFM from individually tracked features in a bootstrapping phase. For the rest of the sequence the model is used to constrain the registration-based tracking but the 3D pose is computed in one unified step, optimal w.r. to the SSD error.

The rest of the paper is organized as follows: we start with a presentation of the general tracking algorithm in Section II, and then present the details for useful combinations of motions (3D models and 2D planar image warps) in Section III. A complete model acquisition and tracking system algorithm is describeed in Section IV. The qualitative and quantitative evaluation of the algorithm is presented in Section V followed by conclusions and a discussion in Section VI.

II. General tracking problem formulation

We consider the problem of determining the motion of a rigid structure in a video sequence using image registration. We assume that an initial sparse 3D structure is calculated from images using structure-from-motion (SFM) (see Section IV). A sparse 3D structure represented by a set of 3D points $\mathbf{Y}_i$, $i = 1, N$ is calculated in a training stage using uncalibrated SFM techniques (see Section IV). The structure points define Q image regions that are tracked in the sequence. Each region $\mathcal{R}_k$ is determined by a number of control points $\mathbf{Y}_{kj}$ that define its geometry. For example, a planar surface region can be specified by 4 corner points. The model points are projected onto the image plane using a projective transformation. First we develop the general theory without

committing to a particular projection model and denote the 3×4 projection matrix for image I_t by P_t. Hence, the model points are projected in image I_t using:

$$\mathbf{y}_{ti} = P_t \mathbf{Y}_i, \quad i = 1, N \tag{1}$$

where $\mathbf{y}_{ti}$ denotes the projection in image t of control point i. Throughout the paper we use i as an index for the control points, t for indexing frames (time) and k for indexing regions. Let $\mathbf{x}_k = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_{K_k}\}$ denote all the (interior) image pixels that define the projection of region $\mathcal{R}_k$ in image I. We refer to $I_0 = T$ as the *reference image* and to the union of the projections of the model regions in T, $\sum_k T(\mathbf{x}_k)$ as the *reference template*. The goal of the tracking algorithm is to find the (camera) motion P_t that best aligns the image template with the current image I_t. A more precise formulation follows next. Refer to Figure 1 for an illustration of the tracking approach.

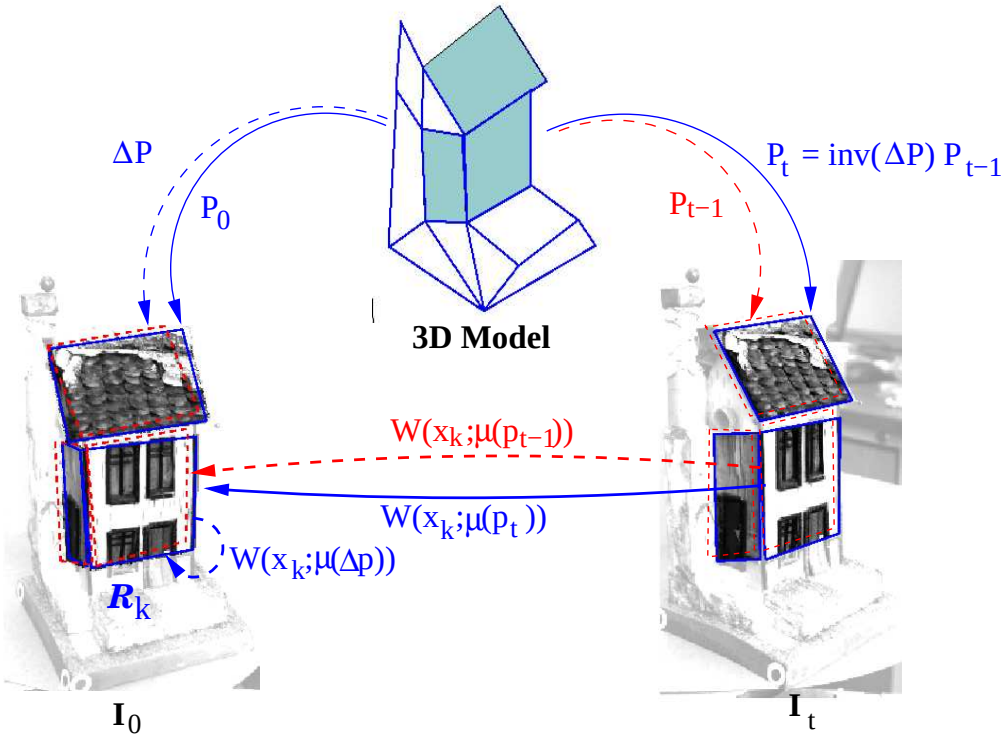


Fig. 1. Overview of the 2D-3D tracking system. In standard SSD tracking 2D surface patches are related through a warp W between frames. In our system a 3D model is estimated (from video alone), and a global 3D pose change ΔP is computed, and used to enforce a consistent update of all the surface warps.

A corresponding physically correct 2D-to-2D image warp is associated with a particular camera

model. Assume that the image motion in frame t for each individual model region k can be perfectly modeled by a parametric motion warp model $W(\mathbf{x}_k; \mu(P_t, \mathbf{Y}_k))$ where μ are 2D motion parameters that are determined by the projection of the region control points $\mathbf{y}_{tkj} = P_t \mathbf{Y}_{kj}$. As an example, for a planar region the corresponding 4 control points in the template image and target image t define a homography (2D projective transformation) that will correctly model all the interior region points from the template image to the target image t. Note that the motion of the 3D model is global but each individual local region has a different 2D motion warp W_k. The coupling of all the different 2D motion warps W_k through a single global motion model P constrains the feasible parameters for each individual warp W_k such that its 2D motion is consistent with the rigid 3D motion. This is unlike approaches where independent 2D trackers [3], [9] or feature detectors [19], that can move arbitrarily in the image plane, are first used to determine corresponding 2D-3D points that are then used to compute the 3D camera pose [20]. For convenience, the 2D warp is denoted by $W(\mathbf{x}_k; \mu(\mathbf{p}_t))$ where $\mathbf{p}_t$ are column vectors of the 3D motion parameters that define the camera projection matrix P_t. Note that $W(\mathbf{x}_k; \mu(\mathbf{p}_t))$ refers to the warped points so $\mathbf{x}_w = W(\mathbf{x}; \mu(\mathbf{p}))$. Later in the paper we also denote by $W(\mu(\mathbf{p}))$ the warp transformation (matrix) with the interpretation that $\mathbf{x}_w = W(\mu(\mathbf{p}))\mathbf{x}$.

We next relate the above geometric description to image intensity variation. Under the common image constancy assumption used in motion detection and tracking [11], the tracking problem can be formulated as finding (tracking) the current state $\mathbf{p}_t$ such that:

$$T(\mathbf{x}_k) = I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_t))), k = 1 \dots Q \tag{2}$$

Note the difference from the traditional 2D SSD tracking algorithms [3], [16] where the goal is to find for each region the 2D warp parameters μ_{kt} such that the image constancy assumption is valid:

$$T(\mathbf{x}_k) = I_t(W(\mathbf{x}_k; \mu_{kt})) \tag{3}$$

Therefore the traditional algorithm estimates, at each time step t, K independent 2D motions μ_{kt} while we estimate one set of global 3D motion parameters $\mathbf{p}_t$ global for all Q regions. The new aspect is that here the parameters are directly computed from the global motion and conversely global motion is computed directly from image derivatives. The consequences of this for different global models is further elaborated in Section III.

The tracking problem is simplified by expressing the current pose $\mathbf{p}_t$ in terms of the previous frame pose $\mathbf{p}_{t-1}$ subjected to a small change $\Delta\mathbf{p}$, and then linearized. Here the change is modeled through function composition ("$\circ$") instead of simple addition to allow a more general set of transforms. Mathematically $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}$ can be obtained by minimizing the following objective function with respect to $\Delta\mathbf{p}$:

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

For efficiency, we switch the role of the target and template image, to express the geometric variation $\Delta\mathbf{p}$ on the constant template image T instead of the current frame I_t. In the terminology of [3] this gives an inverse compositional algorithm. The goal then is to find $\Delta\mathbf{p}$ that minimizes:

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

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

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

The notation $\mathrm{inv}(\Delta P)$ means inverting the 3D motion parameters in a geometrically valid way, i.e. in the calibrated camera case when $\Delta P = K[R|\mathbf{t}]$, the inverse motion is given by $\mathrm{inv}(\Delta P) = K[R'| - R'\mathbf{t}]$ (see Section III). As a consequence, if the 2D warp W is invertible, the individual warp update is (see Figure 1):

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

Next we linearize Equation 5 through a Taylor expansion:

$$\sum_k \sum_{\mathbf{x}_k} [T(W(\mathbf{x}_k; \mu(\mathbf{0}))) + \nabla T \left.\frac{\partial W}{\partial \mu}\right|_{\mu=\mu(\mathbf{0})} \left.\frac{\partial \mu}{\partial \mathbf{p}}\right|_{\mathbf{p}=\mathbf{0}} \Delta\mathbf{p} - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})))]^2 \tag{8}$$

where ∇T represents the gradient of the template, $\left.\frac{\partial W}{\partial \mu}\right|_{\mu=\mu(\mathbf{0})}$ is the derivative of the warp with respect to the 2D parameters μ at position $\mathbf{0}$ (template image motion is assumed $\mathbf{0}$) and $\left.\frac{\partial \mu}{\partial \mathbf{p}}\right|_{\mathbf{p}=\mathbf{0}}$ denotes the derivatives of the dependency between 2D and 3D motion parameters $\mu(\mathbf{p})$ also evaluated at position $\mathbf{0}$. We assume that the 3D motion for the template image is zero $T = T(W(\mathbf{x}_k; \mu(\mathbf{0})))$. This assumption is not a limitation as the constraint can be easily achieved by rotating the model in order to aligned it with the first frame at the beginning of tracking. We denote the derivative images with $M = \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}}$ (matrix notation) and the column vector

image error with $\mathbf{e}_t = T - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})))$. Under Gauss-Newton approximation the problem is equivalent to solving the linear least square:

$$M\Delta\mathbf{p} \simeq \mathbf{e}_t \tag{9}$$

which can be solved for in the least square sense using the normal equations or better QR factorization. The derivative images are evaluated on the constant template image and hence they are constant across iterations and can be precomputed. (Unlike the case where variability Δp is expressed on the time varying video images and hence derivatives have to be recomputed at every frame [16].

This results in an efficient tracking algorithm that can be implemented in real time (see Section IV).

Computing derivatives images

We compute the derivative images from spatial derivatives of template intensities and the inner derivatives Jacobian of the warp. In conventional 2D image-plane tracking this only involves taking derivatives w.r.t. the six (affine [9]) or eight (projective [3]) warp parameters. But as mentioned before, in our 3D tracking the 2D motion parameters μ for a region k are functions of the 3D motion parameters $\mathbf{p}$, the 3D control points $\mathbf{Y}_j$ and the position of the control points in the template image $\mathbf{y}_{0j}$. The warp $W(\mu(\mathbf{p}))$ maps the projected control points in the current image $\mathbf{y}_j = P\mathbf{Y}_j$ to the template image by:

$$\mathbf{y}_{0j} = W(\mu(\mathbf{p}))\mathbf{y}_j = W(\mu(\mathbf{p}))P\mathbf{Y}_j, \quad j = 1, N \tag{10}$$

Note that here we used the notation $W(\mu(\mathbf{p}))$ for the warp as a 2D-2D homography transformation (3×3 matrix). Hence the warp W is a composed function, and its derivatives can be calculated as:

$$\frac{\partial W}{\partial \mathbf{p}} = \frac{\partial W}{\partial \mu}\frac{\partial \mu}{\partial \mathbf{p}}$$

First the warp derivatives with respect to the 2D motion parameters are directly computed from the chosen warp expression (see Section III for some examples). However, the explicit dependency between the 2D parameters μ and the 3D motion parameters $\mathbf{p}$ is not always obtainable (see Section III-B), but Equation 10 represents their implicit dependency, so the

$\frac{\partial \mu}{\partial \mathbf{p}}$ terms are computed using the implicit function theorem. Assume that Equation 10 can be written in the form:

$$A(\mathbf{p})\mu(\mathbf{p}) = B(\mathbf{p}) \tag{11}$$

Taking the derivatives with respect to each component p of $\mathbf{p}$ we get:

$$\frac{\partial A}{\partial p}\mu + A\frac{\partial \mu}{\partial p} = \frac{\partial B}{\partial p} \tag{12}$$

Therefore, for a given $\mathbf{p}$ value, we can linearly compute μ from the Equation 11. Then $\frac{\partial \mu}{\partial p}$ is computed from Equation 12.

III. Practically useful motion models

Different levels of 3D reconstruction - projective, affine, metric Euclidean - can be obtained from an uncalibrated video sequence [10]. A projective reconstruction gives more degrees of freedom (15 DOF) so it might fit the data better under some conditions (e.g. poorly calibrated camera). On the other hand, fitting an Euclidean structure will result in a stronger constraint on the 3D structure, and fewer parameters can represent the model motion (6DOF). For our tracking algorithm we investigated two levels of geometric models reconstructed under perspective camera assumption - projective and Euclidean.

As mentioned before, the 2D image warp motion is determined by the region control points. Different motion approximations are common for the 2D-2D image warps. Warps with few parameters (e.g 2D translation) are in general stable for small regions or simple motion. To better capture the deformation of a region, more general warp should be considered. But, typically tracking with these warps need a large planar surface area or, as we will explore here, stabilization from a 3D model. A natural parametrization, which also correctly captures motion of planar regions, would be a homography warp for a perspective camera model (projective or Euclidean) and an affine warp for a linear camera model (orthographic, weak perspective, para-perspective). The next subsections give concrete examples of how the tracking algorithm can be applied to three types of useful combinations of motions: an Euclidean model with either small translational patches, or larger homography patches, and a projective model with small translational patches.

A. Euclidean model with translational warps

A perspective calibrated camera has the following form in Euclidean geometry:

$$P = K[R|\mathbf{t}] \tag{13}$$

where K is the camera calibration matrix (internal parameters), $R = R_x(\alpha_x)R_y(\alpha_y)R_z(\alpha_z)$ represents the rotation matrix and $\mathbf{t} = [t_x, t_y, t_z]^T$ is the translation vector. So the 3D motion parameters are $\mathbf{p} = [\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z]$. A translational warp is controlled by one model point for each region and has the form:

$$W(\mathbf{x}_k; \mu) = \mathbf{x}_k + \mu \tag{14}$$

where $\mu = [\mu_x, \mu_y]^T$ is the 2D image translation vector and is computed from the motion of the control point $\mathbf{Y}_k$ using:

$$\mu(\mathbf{p}) = \mathbf{y}_{0k} - K[R|\mathbf{t}]\mathbf{Y}_k \tag{15}$$

The inner derivatives $\frac{\partial W}{\partial \mu}$ and $\frac{\partial \mu}{\partial \mathbf{p}}$ can be directly computed from Equation 14,15 without the need of the implicit function formulation.

B. Euclidean model with homography warps

The image motion of a planar patch can be modeled projectively using a homography warp that is determined by at least 4 control points $\mathbf{Y}_{kj}$. Denote the projection of the control points in the current image by $\mathbf{y}_{tj}$. Note that k is dropped as here we show all the calculations for only one region. With the Euclidean camera model, $\mathbf{y}_j = K[R|\mathbf{t}]Y_j$. A homography can be represented using 8 independent parameters $\mu = [\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8]^T$ [1]:

$$W(\mathbf{x}; \mu) = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 \\ \mu_4 & \mu_5 & \mu_6 \\ \mu_7 & \mu_8 & 1 \end{bmatrix} \mathbf{x} = H\mathbf{x} \tag{16}$$

[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.

The explicit dependency of the 2D warp parameters as function of 3D motion parameters is difficult to obtain analytically in this case, but we can apply the method described in Section II and compute the inner derivatives $\frac{\partial \mu}{\partial \mathbf{p}}$ using the implicit dependency from Equation 10:

$$\mathbf{y}_{0j} = H\mathbf{y}_j \quad j = 1, N \quad (N \geq 4) \tag{17}$$

which can be put in the form of Equation 11 $A(\mathbf{p})\mu = B(\mathbf{p})$ with

$$A(\mathbf{p}) = \begin{bmatrix} y_1^1 & y_1^2 & 1 & 0 & 0 & 0 & -y_1^1 y_{01}^1 - y_1^2 y_{01}^1 \\ 0 & 0 & 0 & y_1^1 & y_1^2 & 1 & -y_1^1 y_{01}^2 - y_1^2 y_{01}^2 \\ \vdots & & & & & & \\ y_N^1 & y_N^2 & 1 & 0 & 0 & 0 & -y_N^1 y_{0N}^1 - y_N^2 y_{0N}^1 \\ 0 & 0 & 0 & y_N^1 & y_N^2 & 1 & -y_N^1 y_{0N}^2 - y_N^2 y_{0N}^2 \end{bmatrix} \tag{18}$$

$$B(\mathbf{p}) = [y_{01}^1, y_{01}^2, \ldots, y_{0N}^1, y_{0N}^2]^T \tag{19}$$

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

C. Projective model with translational warp

This final example is very similar to the first one except that the 3D motion is represented by a projective 3×4 camera matrix P with 11 independent parameters $\mathbf{p} = [p_1 p_2 \ldots p_{11}]^T$. The 2D warp parameters μ are related to $\mathbf{p}$ by:

$$\mu(\mathbf{p}) = \mathbf{y}_{0k} - P\mathbf{Y}_k \tag{20}$$

The translational warp is given by Equation 14.

This model presents difficulties in calculating a unique and numerically stable inverse of the 3D motion, as required in Equation 6. To avoid this problem, while we still compute a global motion update $\Delta\mathbf{p}$ we instead update each warp independently as in Equation 7. This solution is closer to the original SSD tracking algorithm [3], [9] and, as demonstrated by the experimental results, performs worse than our new algorithm described in Section II, but still better than the simple unconstrained image plane SSD tracker.

IV. Tracking system and model acquisition

We incorporated the proposed 3D tracking method in a system that first initializes the 3D model from 2D image tracking over a limited motion in an initial video segment and then switches to track and refine the model using 3D model based tracking.

A. Bootstrapping phase

1) *2D SSD Tracking* Several salient surface patches are manually selected in a non-planar configuration from a scene image and tracked in about 100 frames using standard (image-plane) SSD trackers as in [3], [9].
2) *Model computation* From the tracked points a 3D model is computed using structure-from-motion and stratified reconstruction [10] (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 [24] 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 projection is recovered using self-calibration [23]. 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 still use a 6DOF Euclidean motion model for tracking the motion between frames.

   In a previous paper [6] we compared the accuracy of the SFM algorithms for different geometries (affine, projective, Euclidean) and we show 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 is in the convergence range for the tracking algorithm.
3) *Initialize 3D tracking* The 3D model is related to the start frame of 3D tracking using the 2D tracked points $\mathbf{y}_i$. The camera matrix is computed using camera resection from $\mathbf{y}_i \leftrightarrow \mathbf{Y}_i$ 2D-3D correspondences (we used the non-linear algorithm for accuracy [10]). Then the model based tracking algorithm is initialized by computing the gradient images M at that position (Section II).

B. Tracking phase

The tracking is now continued with the 2D surface patches integrated in the 3D model that enforces a globally consistent motion for all surface patches.

1) *Position update* 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 images (Equation 9). The global camera pose P_t is updated using Equation 6.
2) *Add new patches* New patches visible can be added by first tracking their image projection using 2D tracking, then computing their 3D coordinates through camera intersection in $n \geq 2$ views. In the current implementation the user specifies (clicks on) the image control points $\mathbf{y}_i$ that will characterize the new surfaces but in the future we plan to automatically select salient regions.
3) *Delete occluded patches* During tracking, we calculate the visibility of each patch using a Z-buffer algorithm. Patches that become occluded are eliminated and re-activated only when they become visible.

V. Experimental results

Two important properties of tracking methods are *convergence*, *stability* 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, 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 step pose updates $\Delta\mathbf{p}$ are computed by solving an overdetermined equation system, Equation 9. 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, warp models W which cause very apparent image change, such as image plane translations are easy to track, while ones with less apparent image change such as scaling and out-of-plane rotations are more difficult. A general plane-to-plane transform such as the homography contains all of these and tends to have a relatively large condition number. By tracking a 3D model, the 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.

One of the main advantages of the proposed method when using a 3D Euclidean models is that actual 3D camera pose can be tracked. This is not the case with traditional SSD tracking where image patches are tracked in a 2D image space. Therefore, one additional aspect that characterize our model-based trackers is the recovered *3D position accuracy*.

We have tested the three proposed trackers (described in Sections III-A,III-B and III-C) and compare them with the corresponding 2D SSD trackers (translational and homography) and described in [3] and mathematically equivalent with [9], [16]. Here are the notations that will be used in the experiments section:

- `T` : 2D SSD translational (as in [3], [9], [16])
- `TE` : 3D Euclid. model + translation (Section III-A)
- `TP` : 3D projective model + translation (Section III-C)
- `H` : 2D SSD homography (as in [3], [9], [16])
- `HE` : 3D Euclid. model + homohgraphy (Section III-B)

We divided the experimental results in four parts, the first two are performed on artificial data and the last two on real images. The first experiment presented in Subsection V-A measures the robustness and convergence of the trackers for different magnitude of motions (in image space). The second experiment from Subsection V-B measures the sensitivity of the model-based trackers (`TE, TP, HE`) to noise in the model. In Subsection V-C we test the stability of the trackers in two real sequences with quite large motion, and the last experiment from Subsection V-D measures the accuracy of the recovered 3D position for one of the model-based trackers (`HE`).

A. Convergence and stability: synthetic motion perturbations

We used a frame of the first real sequence from the third experiment along with the 3D model (estimated as described in Section IV). The used homography trackers are shown in Figure 2 left and the translational trackers in Figure 2 right. We perturbed the model 1000 times with 10 different magnitudes of 3D pose changes $\Delta\mathbf{p}$ (corresponding to $0.1 \ldots 10$ pixels projected in the image space), and render an image containing the trackers warped with the corresponding 2D warp $\mu(\mathbf{p})$.

Figure 3 (a) plots the frequency of convergence for the 5 tested trackers (ratio over 100 tests

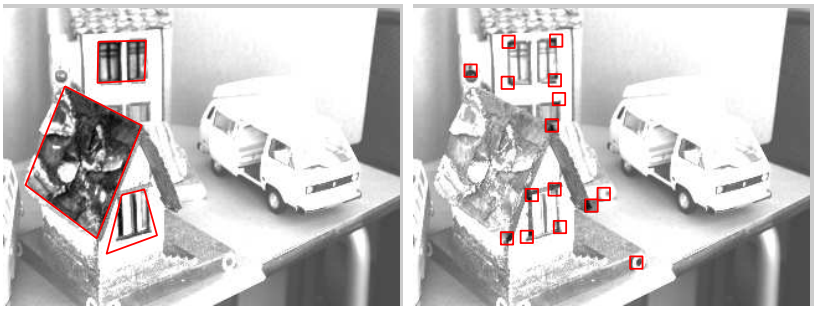
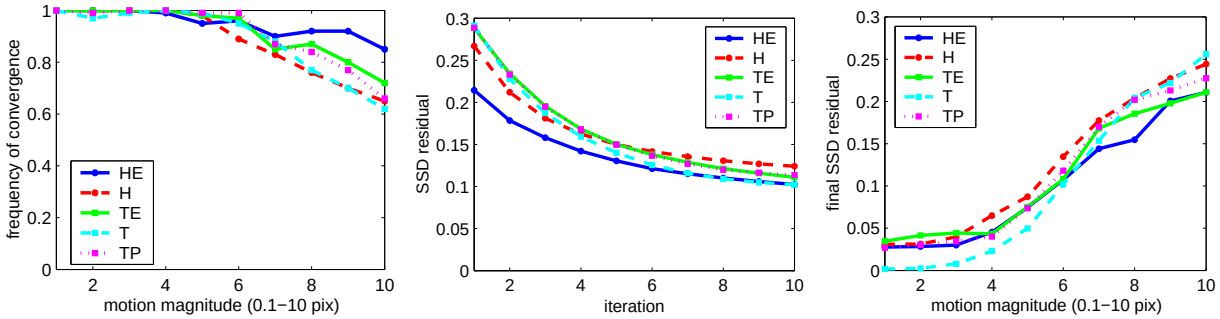


Fig. 2. Templates used for homography trackers (left) and translational trackers (right). Test images are produced by perturbing the 3D pose of the model and rendering templates with the corresponding 2D homography.



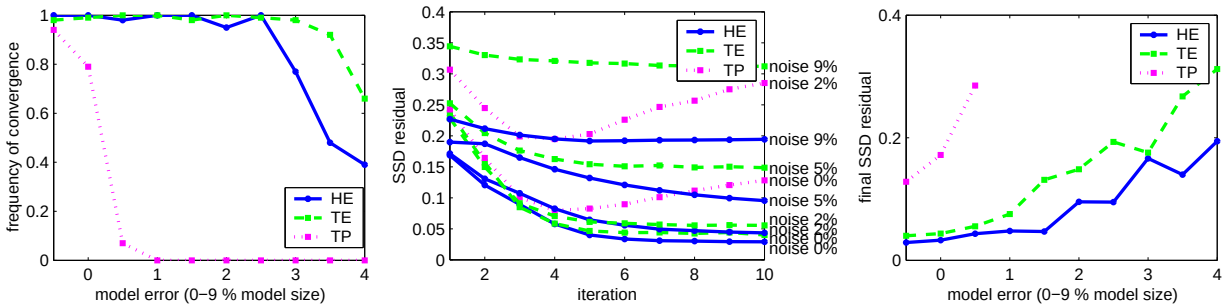
(a) Frequency of convergence (b) Convergence rate (c) Average residuals

Fig. 3. Comparing the stability and convergence of different trackers. We tested 10 levels of motion magnitude. The SSD error is in pixel intensity space [0,1]

in each motion level). For small $\Delta\mathbf{p}$ there is not a huge difference in convergence rate among different trackers but we notice a better convergence for the trackers that use the Euclidean model (TE, HE). For large perturbations the model-based homography tracker HE fails only half as often compared to the others. Figure 3 (b) shows the average convergence rate (mean over the convergent tests). We notice again that the HE, TE and TP algorithms have somewhat a better speed of convergence. The 2D translational tracker also converges quite well due to the limited DOF (2) that it estimates. The last graph (Figure 3 (c)) shows average residuals over all convergent tests depending on the motion magnitude. The HE and TE can tolerate larger motions, while the 2D translational T gives the lowest residuals for low levels of motion. This is again due to the reduced number of estimated parameters.

B. *Sensitivity to model accuracy : synthetic images*

As our model-based tracking system uses a model estimated with structure-from-motion (SFM) and therefore the model is not perfect, we tested the sensitivity of the model-based trackers TE, TP, HE to model deviations. We introduced two types of deviations, first we perturbed the model control points (involved in the computation of the warps). In a second experiment we tested the sensitivity of the homography based trackers to patch planarity.

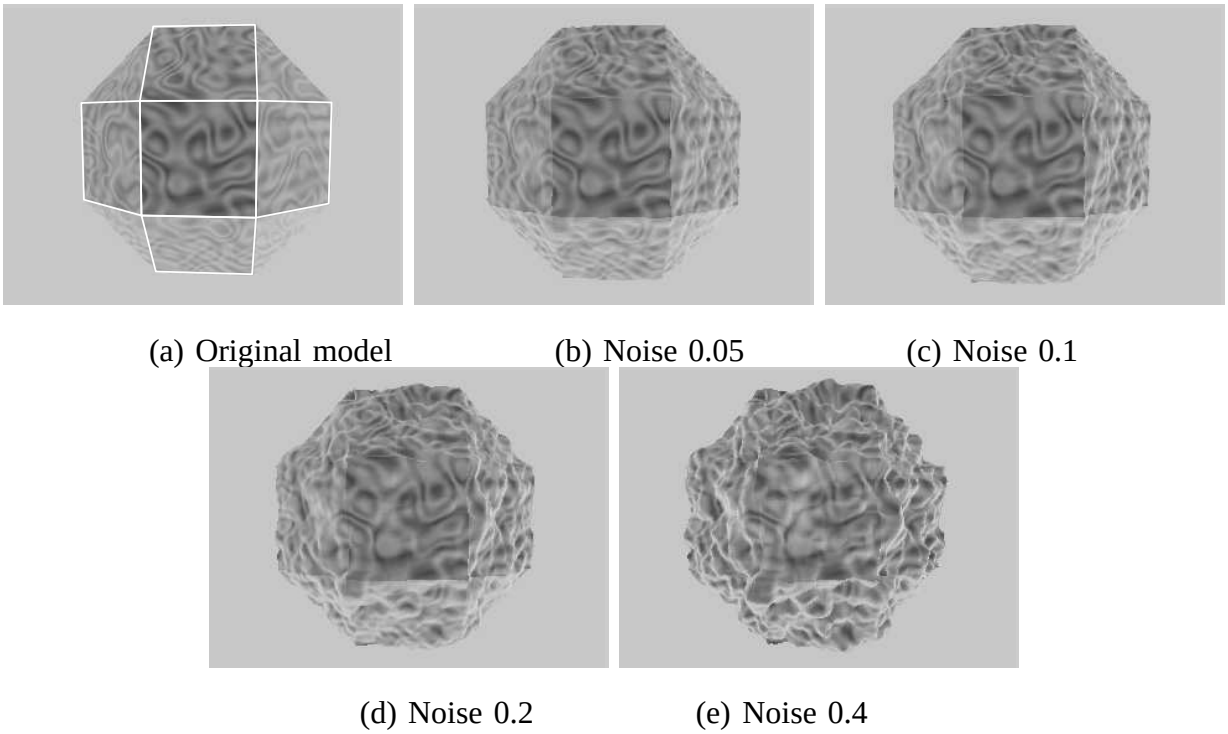


(a) Frequency of convergence (b) Convergence rate (c) Average residuals

Fig. 4. Comparing the stability and convergence of different trackers when adding 10 levels of noise in the geometric model.

We performed the same experiment as in Section V-A with a mean perturbation (such as all trackers have a high frequency of convergence) and added 10 levels of noise in the model points. Note that this time the nature of the perturbation is different as in the first case (Section V-A) we had a motion that we expect to be corrected by the tracking while now we introduce noise in the model that will be used in the mode-based tracking. The results are displayed in Figure 4. The convergence for the trackers that use a projective model (TP) deteriorates rapidly and the trackers stop converging for model noise bigger than 2% (see Figure 4 (a)). Among the trackers that use the Euclidean model ((TE, HE), the translational one (TE) has better convergence but bigger residual (see Figure 4 (b) (c)). Overall both TE and HE can tolerate quite a lot of noise as they have a frequency of convergence close to 1 for noise level up to 6%. In practice, for an estimated model (see Section IV) the average reprojection error is less then one pixel and therefore does not cause any problem.

In a second type of experiment we tested the sensitivity of the homography trackers (H, HE) to patch planarity. We generated an artificial model (see Figure 5 (a)) and we added



(a) Original model (b) Noise 0.05 (c) Noise 0.1

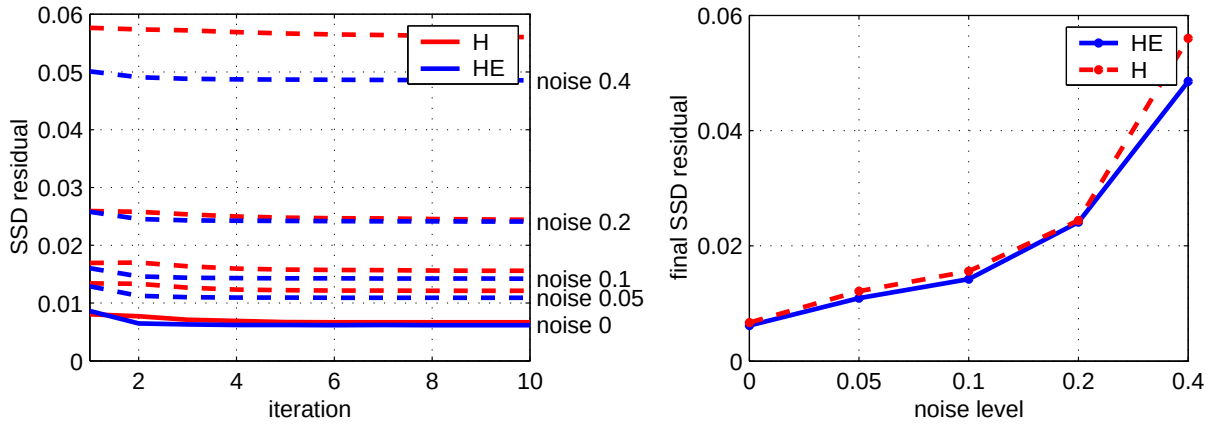
(d) Noise 0.2 (e) Noise 0.4

Fig. 5. (a) Synthesized model with four levels of plane perturbations. The numbers represent percentage of plane width. (b)(c)(d)(e)

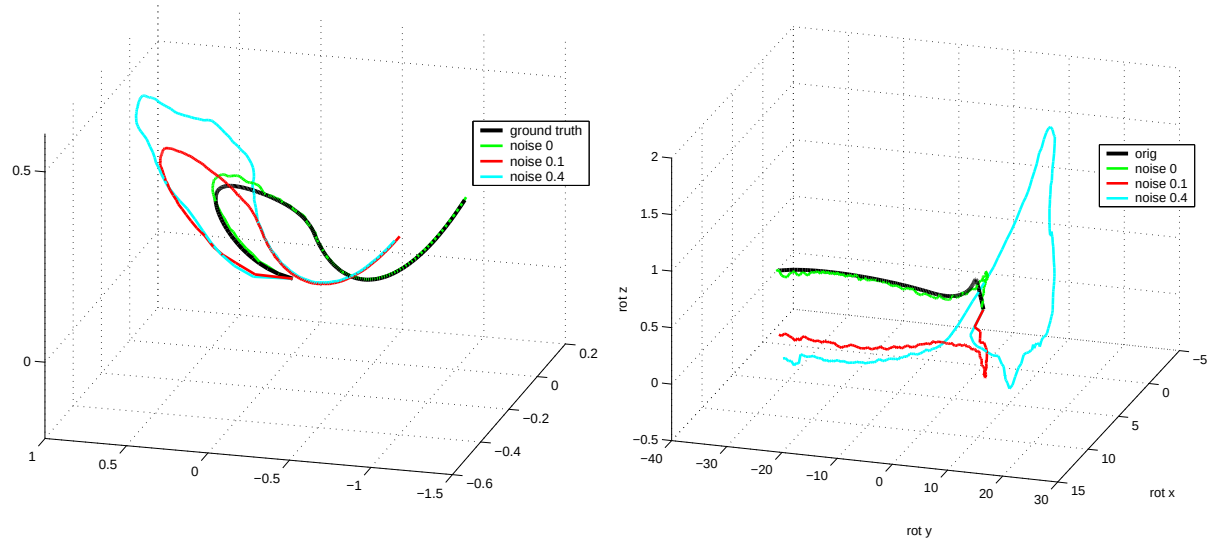
four levels of perturbations to the planes (using displacement maps with different magnitudes $5\%, 10\%, 20\%, 40\%$ of planes width- see Figure 5 (b), (c), (d), (e)).

For each perturbed model we measured the convergence rate and final residuals (see Figure 6 (a), (b)). The trackers did not get lost in any of the sequences, showing a good tolerance to patch planarity. As expected, the residuals increase for higher levels of noise as the template does not anymore quite fit the current image (because of parallax and occlusions caused by the displacement map).

For the model-based tracker (HE) we also measured the accuracy of the recovered position as compared to the ground truth. Figure 6 (c), (d) plots the recovered trajectory (translations) and rotations for each level of perturbations. We noticed that while the recovered trajectory with no noise (Figure 5 (a)) is very close to the ground truth (original trajectory), the error increases while adding noise. The mean error in translation at the last level of noise ((Figure 5 (e)) is about 0.4 matching the magnitude of the perturbation while the rotation error is about 6°.



(a) Convergence rate (b) Average residuals



(d) Recovered position (e) Recovered orientation

Fig. 6. Sensitivity of homography based trackers patch planarity. We tested four levels of perturbations. (top) Measured convergence (bottom) recovered position for tracker HE.

C. *Convergence and stability: real sequence*

We now tested the stability of the trackers for two real sequences.

In Figure 7 planar regions in the image sequence are tracked using an 8DOF homography. When each patch is tracked individually as in [3] (top images) the first region is lost already after 77 frames and all lost after 390 frames. (See `video1` left [1]). The condition numbers for M varies between $5 * 10^5$ and $2 * 10^7$, indicating a numerically ill conditioned situation. When

instead the regions are related by the global 3D model using our algorithm, pose is successfully tracked through the whole sequence of 512 frames (`video1` right [1]). Additionally the model allows the detection and removal of the region on the left roof side when it becomes occluded and the introduction of three new regions on the right roof side and the smaller outhouse when they come into view. The condition number of the 6DOF (3 rot, 3 trans) using a rigid 3D model is 900, which is significantly better than that of the 8DOF planar 2D homography (above).

The next experiment uses a simpler 2DOF translation model to relate regions as described in Sections III-A and III-C, through either an Euclidean or Projective global 3D model. In Figure 8 three cases are compared. In the first, (figure top and `video2` left [1]) no model is is used standard individual 2D trackers as in [9], [16]), and almost half of the region trackers are lost starting already from frame 80. Because only 2D spatial x and y derivatives are used in M the condition number is very low at an average 1.3. In the middle sequence, a projective model is used to relate the regions. This stabilizes the tracking until about frame 400, where one tracker is slightly off target and further at about frame 430 some are lost due to occlusion. The projective model has 11 DOF and the condition number is quite high at $2 * 10^4$. In the final (figure bottom) sequence a Euclidean model relates the trackers, and provides handling of occlusions. The condition number is a reasonable 600, and the whole 512 frame sequence is successfully tracked.

The second sequence has a much larger overall motion showing the ability to maintain track without concurrent visibility. The results for four trackers are displayed in Figure V-C (see also `video6,7` [1]). Like in the previous experiment new patches are introduced as they become visible and the ones that are occluded are deleted. We can automatically detect occlusions using Z-buffering for the model based tracking but not for the simple SSD one. To be consistent we manually add/delete patches in all four cases. We see that the SSD homography tracker (`H`) looses track quite early in the sequence while the model-based homography tracker (`HE`) trackers all patches until the end. The translational tackers (`T,TE`) perform similar in this case and are able to track all patches for the whole sequence.

D. Recovered pose accuracy

One of the main advantages of the proposed method when using a 3D Euclidean model is that actual 3D camera pose can be tracked (as opposed to traditional SSD tracking where image

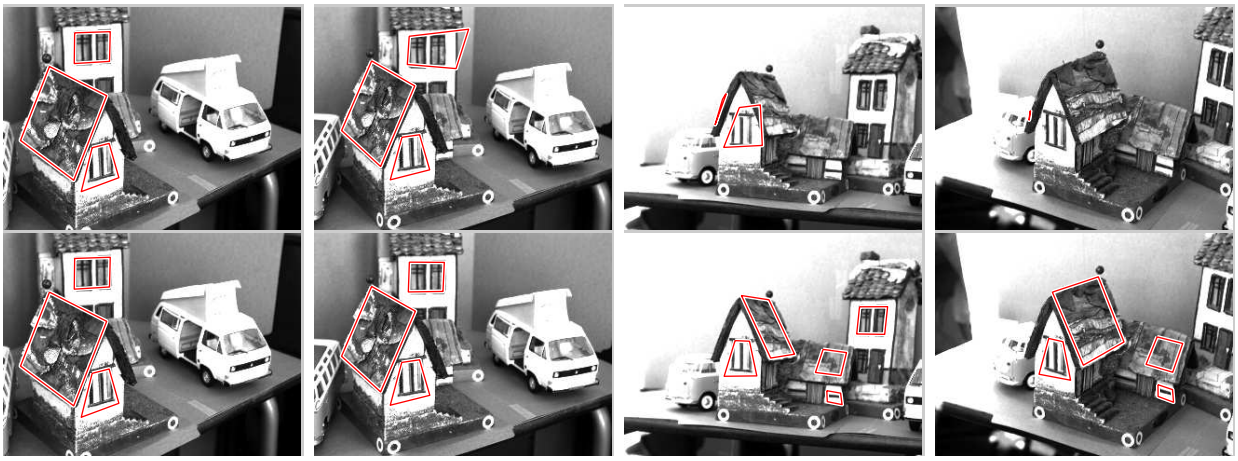


Fig. 7. **Top** Tracking individual patches using a homography based on conventional SSD tracking as in [3]. Not all regions can be tracked through the whole sequence and occlusion is not handled. **Bottom** Through the 3D model each region motion is rigidly related to the model, and tracking succeeds through the whole sequence. The model also allows detection and removal of occluded regions and introduction of new regions. See `video1` [1]

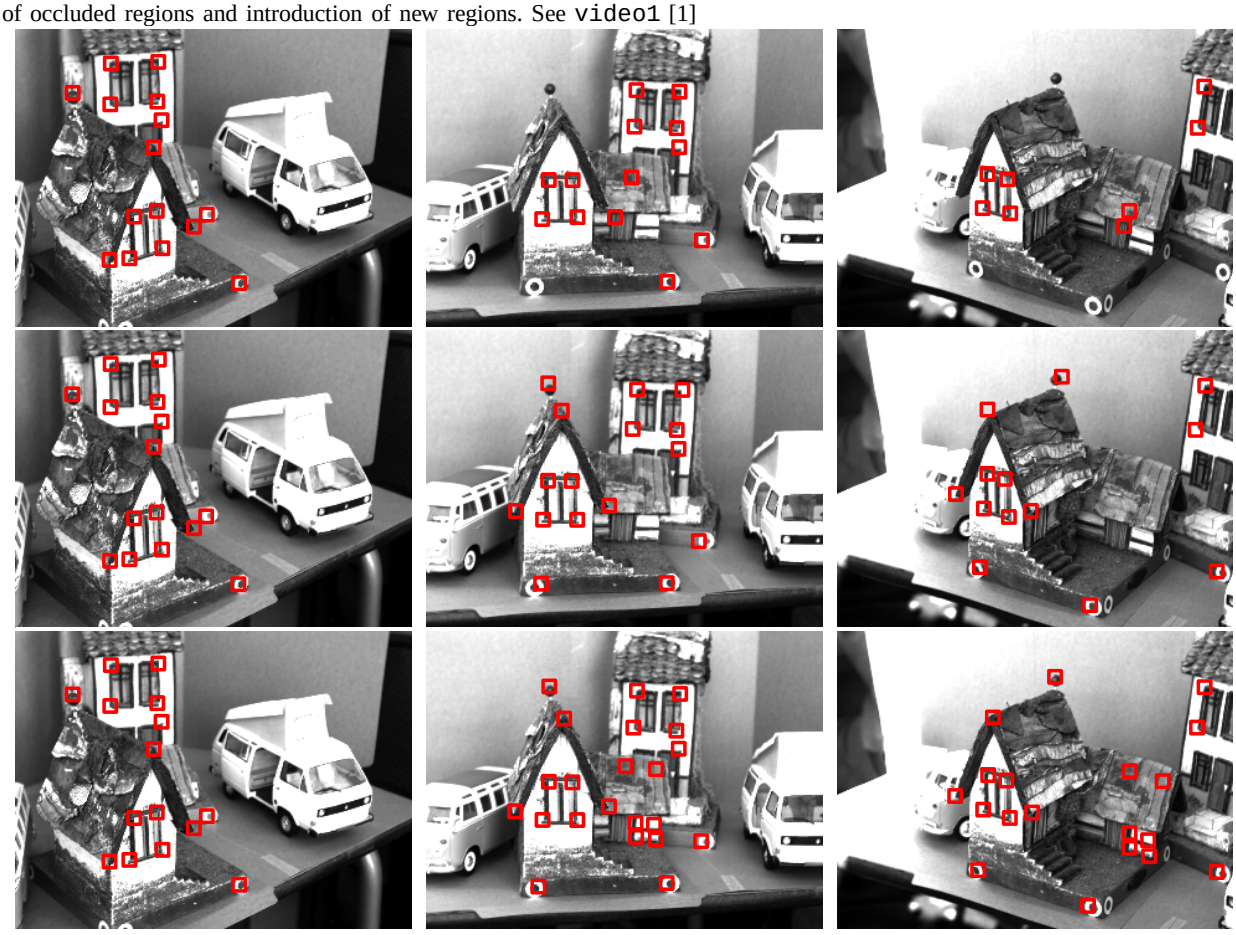
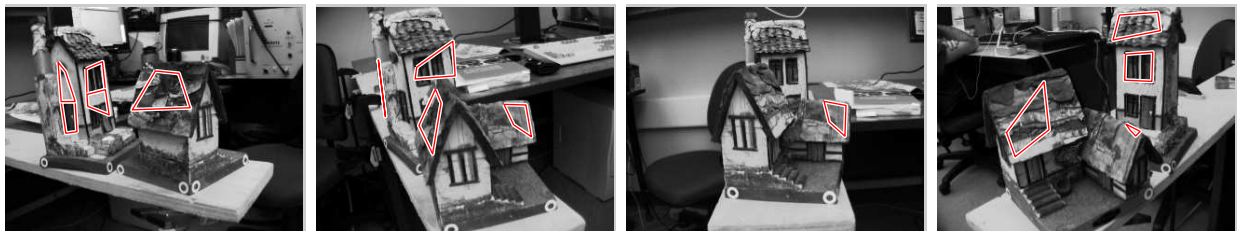
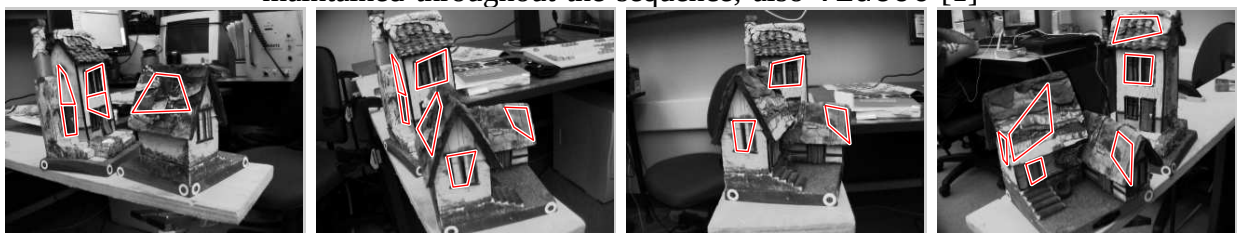


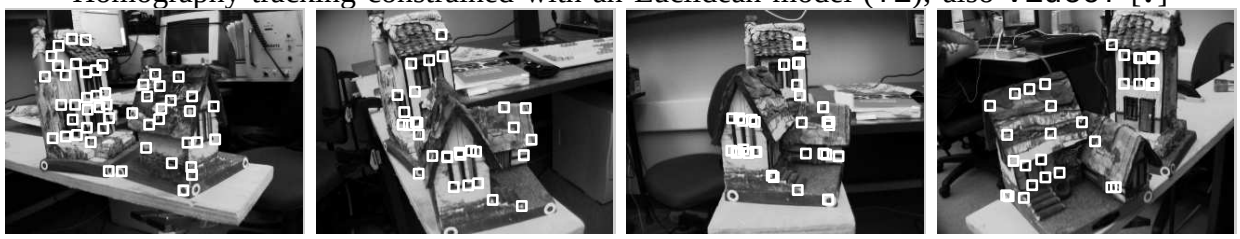
Fig. 8. **Top** Translation tracking of individual regions based on conventional SSD tracking as in [9], [16]. Through the video sequence many patches are lost. **Middle** A projective 3D model is used to relate the regions, and provide more stable tracking through the whole sequence. **Bottom** An Euclidean model relates the regions, and also allows the deletion of occluded and introduction of new regions. `video2` [1]



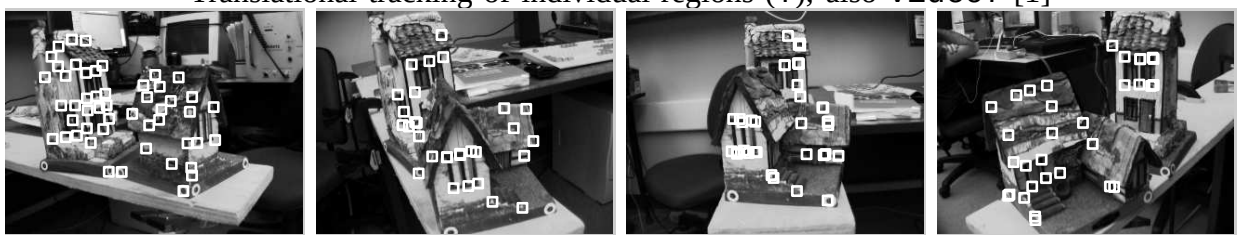
Homography tracking of individual regions (H); Sufficient number of tracked regions cannot be maintained throughout the sequence; also `video6` [1]



Homography tracking constrained with an Euclidean model (TE); also `video7` [?]



Translational tracking of individual regions (T); also `video7` [1]



Translational tracking constrained with an Euclidean model (HE); also `video7` [1]

patches are tracked in a 2D image space and pose would have to be computed later). This is useful for example in robotics or augmented reality applications. To test the accuracy of the recovered camera position we tracked two trajectories composed from straight lines using the homography model-based tracker (HE). Figure 9 shows two image frames of the scene that is tracked, and `video4` [1] shows a whole motion sequence.

The first trajectory was a straight line in the horizontal plane of about 1m. Figure 10 (left) illustrates the recovered trajectory. To measure the accuracy of the tracking algorithm we cali-

brated the 3D model for the planes assuming some given real dimensions (distance from camera to one plane) so we could get the translation in meters. We found that the tracked trajectory had about 0.95 cm mean deviation from a straight line and 5.1 cm mean deviation from the horizontal plane. The recovered line length was about 1.08 m, resulting in an error of 8% with respect to the measured ground truth. There was no camera rotation along the first trajectory. Corresponding measured rotation was less than 1 degree on average.

The motion in the second trajectory was along two perpendicular lines in the horizontal plane. In this experiment, the real physical motion did not turn out particularly smooth and the recorded data therefore also somewhat jumpy. We measured the angle between the two lines fitted to the recovered positions (see Figure 10) as $82°$. Hence it had an error of about $8°$ with respect to the ground truth.

The experiments show that the accuracy of the measurements connected to properties that are not directly related to calibrated properties of the structure (e.g. deviation from lines, planes) is higher that the accuracy in measured distances. This is due to the difficulty in making calibrated (Euclidean) measurements from an initially uncalibrated (projective) camera.

VI. Discussion

We have shown how conventional 2D SSD tracking can be extended to 3D using a scene model estimated from images alone. The method makes tracking of a-priori unknown scenes more stable and handles occlusions by removing and introducing tracking regions as appropriate when new views become available. A main feature of our method is that 3D pose change ΔP is computed directly from image intensity derivatives w.r.t. P. Note that this guarantees the best 3D pose update available from the linearized model (here using L_2 norm, but other e.g. robust norms are also possible [9]). This is unlike the more common approach of first tracking 2D image correspondences, and then computing a 3D pose from points, where first each 2D point location is committed to based on a locally optimal image fit but without regards to the global 3D constraints.

In combining different types of 3D global models and 2D region warps we found that:

- Tracking planar regions using an 8DOF homography without a 3D model is unstable due to the many DOF estimated, but limited image signature available from geometric change of only one planar patch.

- Using the estimated 3D model we constrain multiple individual patches to move in a consistent way and achieve very robust and stable tracking of full 3D pose over long sequences.
- With some loss in generality and magnitude of maximum trackable pose change, the image-plane 8DOF homography can be replaced by simple and faster 2DOF translational trackers. Each individual such tracker has to use only a small image region since it doesn't deform projectively, but instead many regions can be used. Using 2DOF regions with either an Euclidean or projective 3D model this gives almost as good tracking as the homography + 3D model, and makes execution somewhat faster.
- The model-based trackers that use an Euclidean model and recover full 3D camera pose are tolerant to noise in the model that is larger than the usual noise resulting from a SFM estimation. The homography-based tracker performs better and can also accommodate patches that are not perfect planes. But the accuracy of the recovered position degrades when the noise increases.

Convergence for the translational only warp over large angular changes in camera viewpoint can be improved by using a few view-dependent templates each associated with a smaller angular range, and switch these in and out depending on the current angular pose computed from the 3D model. While this introduces a risk for drifts and errors from the templates being slightly offset, in practice we have found it works well using 5-10 different templates over the visible range of a patch.

Visual tracking has many applications in e.g. robotics, HCI, surveillance and model building. Tracking and modeling are interrelated in that (as we have shown) a model improves tracking, and tracking can also be used to obtain the image correspondences needed for a model. In unstructured environments this used to represent a chicken-and-egg like problem: Without a model it was difficult to track, and without tracking one couldn't obtain a model. Our method integrates both into a system which begins by defining regions to track in only a 2D image. First 2D tracking is used over an initial video segment with moderate pose change to obtain point correspondences and build a 3D model from image data. After the model is built, the system switches to 3D tracking and is now ready to handle large pose changes and provide full 3D pose (rotation, translation) tracking.

REFERENCES

[1] On-line mpeg movies of the experiments are available. See `video` 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. *IJCV International Journal of Computer Vision*, 56(3):221–255, 2004.

[4] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IROS*, 2004.

[5] M. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *ECCV96*, pages I:329–342, 1996.

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

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

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

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

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

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

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

[14] K.-C. Lee and D. Kriegman. Online learning of probabilistic appearance manifolds for video-based recognition and tracking. In *CVPR*, 2005.

[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] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *ICRA*, 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] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8):735–758, 2002.

[20] 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 (IS AR)*, 2000.

[21] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, pages 22–30, March 1996.

[22] K. Toyama and G. Hager. Incremental focus of attention for robust vision-based tracking. *IJCV*, 35(1):45–63, November 1999.

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

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

[25] 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.

Fig. 9. Tracking 3D planes. Pose accuracy experiment. `video4` [1]

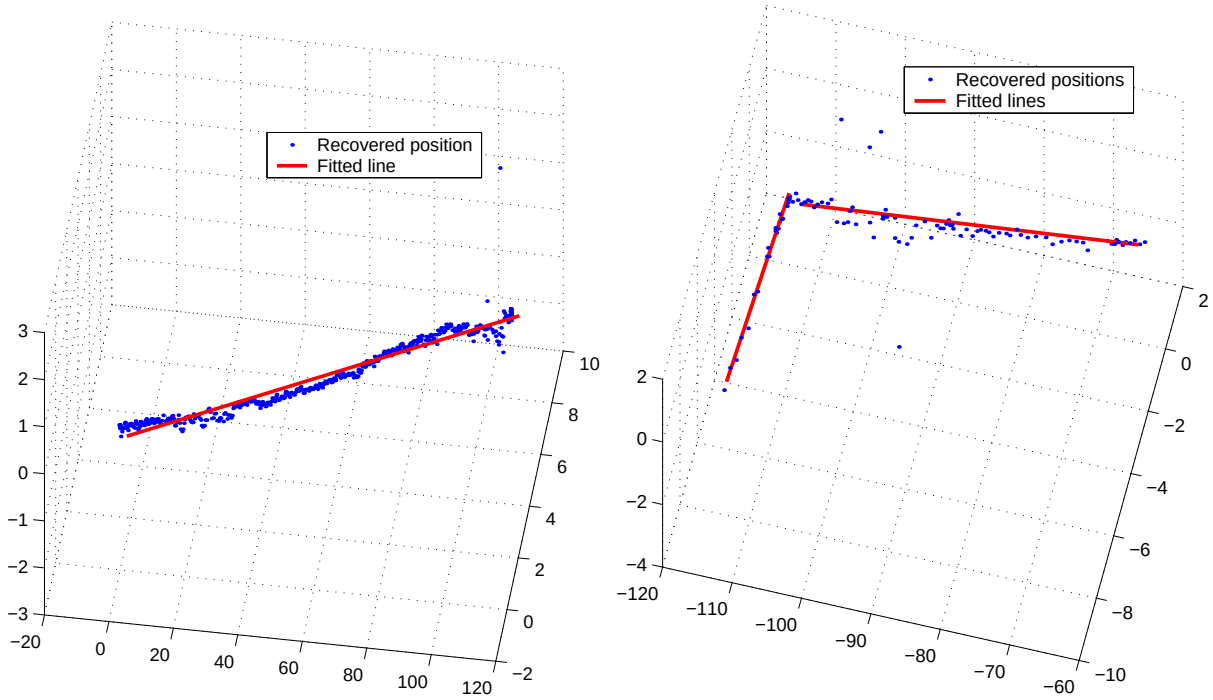


Fig. 10. Recovered positions for the straight line trajectory (left) and the 2 perpendicular lines trajectory (left). The red line are the fitted 3D lines to each line segment.