3D SSD Tracking from Uncalibrated Video

Dana Cobzas[1] and Martin Jagersand[2]

[1] INRIA Rhone-Aples, Montbonnot 38334, France
`Dana.Cobzas@inriaples.fr`
`http://www.cs.ualberta.ca/ dana`
[2] Computing Science, University of Alberta,
Edmonton, T6G2E8, Canada
`jag@cs.ualberta.ca`

Abstract. In registration-based motion tracking precise pose between a reference template and the current image 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. 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 planar regions. It also detects occlusions and removes/inserts tracking regions as appropriate in response.

1 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 [2–4]. 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 [2, 5].

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 motion, it is typically not used for precise 3D tracking alone, but can be used to initialize more precise tracking modalities [6].

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 it's not pre-defined, typically the user points to desired patches in the first frame. This technique can also be used in image alignment to e.g. create mosaics [7]. Early approaches used brute force search by correlating a reference image patch with the current image. While this worked 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 [8], while most recently, efficient "inverse" algorithms have been developed, which allow the real time tracking for the above mentioned 6D affine [9] and 8D projective warp [10]. A related approach [11, 12], where instead of using spatial image derivatives, a linear basis of test image movements are used to explain the current frame, have proved 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.

In this paper we extend the registration based technique from 2D image plane tracking by involving a full 3D scene model, estimated from the same uncalibrated video, and used directly in the computation of the motion update between frames. Our method starts out tracking the image motion of several surface patches using conventional SSD tracking. After some time (typically $\approx$ 100 frames) when the scene-camera pose has undergone sufficient motion 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 algorithm does not require complete scene decomposition in planar facets, but works with few planar patches identified in the scene. 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 connect these together in a physically correct way. We show experimentally that this approach yields more stable and robust tracking than previous approaches, where each surface patch motion is computed individually.

The rest of the paper is organized as follows: we start with a presentation of the general tracking algorithm in Section 2, and then present the details for useful combinations of motions (3D models and 2D planar image warps) in Section 3. A complete model acquisition and tracking system algorithm is descried in Section 4. The qualitative and quantitative evaluation of the algorithm is presented in Section 5 followed by conclusions and a discussion in Section 6.

2 General tracking problem formulation

We consider the problem of determining the motion of a rigid structure through a sequence of images using image registration. A sparse 3D structure for the model described by 3D points $\mathbf{Y}_i,\ i = 1, N$ is calculated in a training stage using uncalibrated structure from motion techniques (see Section 4). 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 general 3×4 projection matrix for image I_t by P_t. Hence, the model points are projected in image I_t using:

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

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, $\cup_k T(\mathbf{x}_k)$ 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 Fig. 1 for an illustration of the tracking approach.

Assume that the image motion in image t for each individual model region k can be perfectly modeled by a parametric motion model $W(\mathbf{x}_k; \mu(P_t, \mathbf{Y}_k))$ where μ are 2D motion parameters that are determined by the projection of

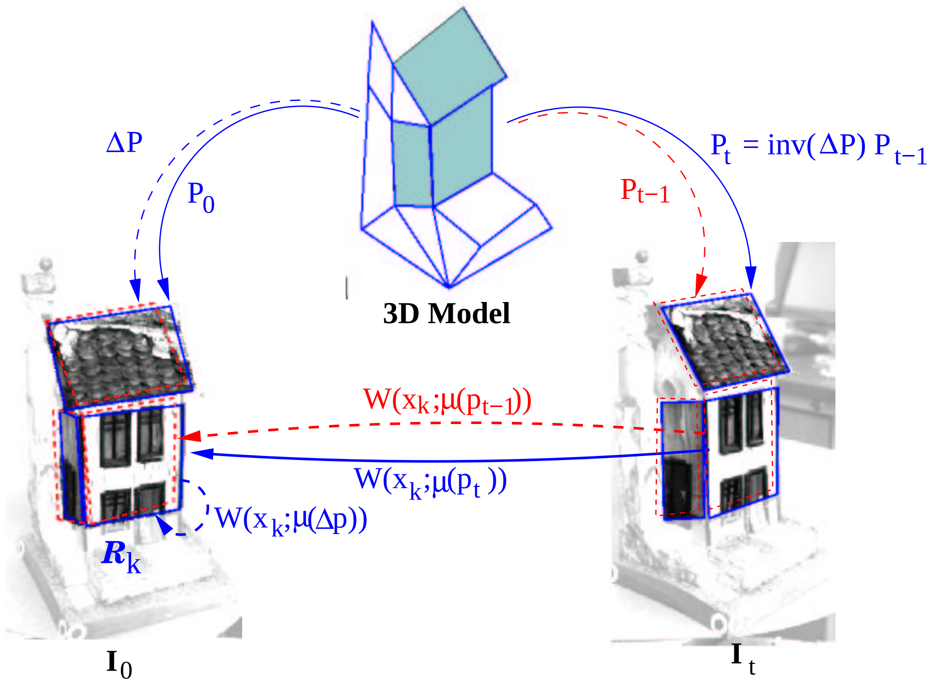

Fig. 1. Overview of the 2D-3D tracking system. In standard SSD tracking 2D surface patches are related through a warp W between frames. In our system a 3D model is estimated (from video alone), and a global 3D pose change ΔP is computed, and used to enforce a consistent update of all the surface warps.

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 3D model motion is global but each individual local region has a different 2D motion warp W_k. For simplicity, 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.

Under the common image constancy assumption used in motion detection and tracking [13] the tracking problem can be formulated as finding $\mathbf{p}_t$ such as:

$$\cup_k T(\mathbf{x}_k) = \cup_k I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_t))) \tag{2}$$

$\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}$ can be obtained by minimizing the following objective function with respect to $\Delta\mathbf{p}$:

$$\sum_k \sum_x [T(\mathbf{x}_k) - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1} \circ \Delta\mathbf{p})))]^2 \tag{3}$$

For efficiency, we solve the problem by an inverse compositional algorithm [10] that switches the role of the target and template image. The goal is to find $\Delta\mathbf{p}$ that minimizes:

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

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

$$P_t = \mathrm{inv}(\Delta P) \circ P_{t-1} \tag{5}$$

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

$$W(\mathbf{x}_k; \mu(\mathbf{p}_t)) = W(\mathbf{x}_k; \mu(\Delta\mathbf{p}))^{-1} \circ W(\mathbf{x}_k; \mu(\mathbf{p}_{t-1})) \tag{6}$$

Performing a Taylor extension of equation 4 gives:

$$\sum_k \sum_x [T(W(\mathbf{x}_k; \mu(\mathbf{0}))) + \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}} \Delta\mathbf{p} - I_t(W(\mathbf{x}_k; \mu(\mathbf{p}_t)))] \tag{7}$$

Assuming that the 3D motion for the template image is zero (which can be easily achieved by rotating the model in order to be aligned with the first frame at the beginning of tracking), $T = T(W(\mathbf{x}_k; \mu(\mathbf{0})))$. Denoting $M = \sum_k \sum_x \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}}$, equation 7 can be rewritten as:

$$M\Delta\mathbf{p} \simeq \mathbf{e}_t \tag{8}$$

where $\mathbf{e}_t$ represents the image difference between the template regions and warped image regions, and the motion $\Delta\mathbf{p}$ is computed as the least squares solution to Eq. 8.

The derivative images $M = \sum_k \sum_x \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}}$ are evaluated at $\mathbf{p} = \mathbf{0}$ and they are constant across iterations and can be precomputed, resulting in an efficient tracking algorithm that can be implemented in real time (see Section 4).

Computing derivatives images

We compute the derivative images from spatial derivatives of template intensities and inner derivatives of the warp. As mentioned before, the 2D motion parameters μ for a region k are functions of the 3D 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 projection of the points in the current image $\mathbf{y}_j = P\mathbf{Y}_j$ are mapped to the template image control points through the 2D warp $W(\mu(\mathbf{p}))$ using:

$$\mathbf{y}_{0j} = W(\mu(\mathbf{p}))(P\mathbf{Y}_j), \quad j = 1, N \tag{9}$$

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 3 for some examples). However, the explicit dependency between the 2D parameters μ and the 3D motion parameters $\mathbf{p}$ is in general difficult to obtain, but equation 9 represents their implicit dependency, so $\frac{\partial \mu}{\partial \mathbf{p}}$ are computed using the implicit function theorem. Assume that equation 9 can be written in the form:

$$A(\mathbf{p})\mu(\mathbf{p}) = B(\mathbf{p}) \tag{10}$$

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

$$\frac{\partial A}{\partial p}\mu + A\frac{\partial \mu}{\partial p} = \frac{\partial B}{\partial p} \tag{11}$$

For a given $\mathbf{p}$ value μ can be linearly computed from equation 10 and then $\frac{\partial \mu}{\partial p}$ is computed from equation 11.

3 Practically useful motion models

Different levels of 3D reconstruction - projective, affine, metric Euclidean - can be obtained from an uncalibrated video sequence [14]. A projective reconstruction gives more degrees of freedom (15 DOF) so it might fit the data better under different noise conditions. On the other hand, fitting a metric structure will result in a stronger constraint, and fewer parameters can represent the model motion (6DOF). For our tracking algorithm we will investigate two levels of geometric models reconstructed under perspective camera assumption - projective and metric. The 3D motion of the model results in 2D motion of the regions $\mathcal{R}_k$ on the image plane.

As mentioned before, the 2D motion is determined through the regions' 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 parameters should be considered but, in general, tracking with these warps need large surface area or 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, paraperspective). 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 small translational patches, or larger homography patches, and a projective model with small translational patches.

3.1 Euclidean model with translational warps

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

$$P = K[R|\mathbf{t}] \tag{12}$$

where the internal parameters are:

$$K = \begin{bmatrix} a_u & s & u_c \\ 0 & a_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

$R = R_x(\alpha_x)R_y(\alpha_y)R_z(\alpha_z)$ 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 points 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 needing the implicit function formulation.

3.2 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 all the calculations are done for 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 $\mu = [\mu_1\mu_2\mu_3\mu_4\mu_5\mu_6\mu_7\mu_8]^T$:

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

The explicitly 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 2 to compute the inner derivatives $\frac{\partial \mu}{\partial \mathbf{p}}$ using the implicit dependency from equation 9:

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

which can be put in the form of equation 10 $A(\mathbf{p})\mu = B(\mathbf{p})$ 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 $\mathbf{y}_j$.

3.3 Projective model with translational warp

This last 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 $\mathbf{p}$,:

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

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 5. To avoid this problem, while we still compute a global motion update $\Delta\mathbf{p}$ we instead update each warp independently as in equation 6. This solution is closer to the original SSD tracking algorithm [10, 9] and, as demonstrated by the experimental results, performs worse than our new algorithm described in Section 2, but still better than the simple unconstrained image plane SSD tacker.

4 Tracking system and model acquisition

We incorporated the proposed 3D tracking algorithm is 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. The main steps in the implemented method are:

1. Several salient surface patches are selected in a non-planar configuration from a scene image and tracked in about 100 frames using standard (image-plane) SSD trackers as in [10, 9].
2. From the tracked points a 3D model is computed using structure from motion and the stratified uncalibrated approach [14] (projective reconstruction that is upgraded to a metric 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 Urban et all [15] 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 is recovered using self-calibration [16]. 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 between frames.
In a recent paper [17] 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 can be obtained (if bundle adjusted). This accuracy is in the convergence range for the tracking algorithm.

3. The 3D model is related to the start frame of 3D tracking using the 2D tracked points $\mathbf{y}_i$ and camera matrix computed using camera resection (non-linear for accuracy [14]) from $\mathbf{y}_i \leftrightarrow \mathbf{Y}_i$ 2D-3D correspondences. Then the model based tracking algorithm is initialized by computing the derivatives images M at that position. 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.
4. New patches visible only in new views are added by first tracking their image projection using 2D tracking then computing their 3D coordinates through camera intersection in $n \geq 2$ views then incorporate them in the 3D model. 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.

5 Experimental results

Two important properties of tracking methods are convergence and accuracy. Tracking algorithms based on a optimization and spatio-temporal derivatives (Eq. 7) 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 7) 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, Eq. 8. 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 tend to have a relatively large condition numbers. 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.

In Fig. 2 planar regions in the image sequence are tracked using an 8DOF homography. When each patch is tracked individually as in [10] (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) model is 900, which is significantly better than the 8DOF homography.

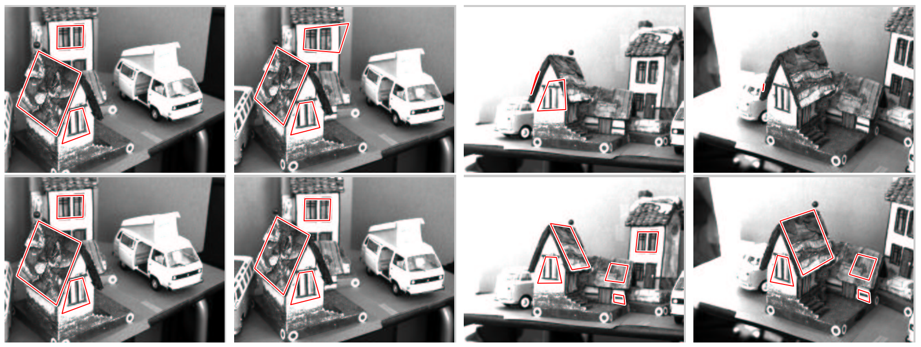

Fig. 2. Top Tracking individual patches using a homography [10]. Not all regions can be tracked through the whole sequence and occlusion is not handled. **Bottom** Through the 3D model each region motion is rigidly related to the model, and tracking succeeds through the whole sequence. The model also allows detection and removal of occluded regions and introduction of new regions. See `video1` [1]

The next experiment uses a simpler 2DOF translation model to relate regions as described in Sections 3.1 and 3.3, through either an Euclidean or Projective global 3D model. In Fig. 3 three cases are compared. In the first, (figure top and `video2` left [1]) no model is is used [8, 9], 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.

In the final experiment we evaluate the accuracy by tracking four sides of a cube textured with a calibration pattern, see Fig. 4. For each frame we superimpose the warped image onto the template, displaying each using separate

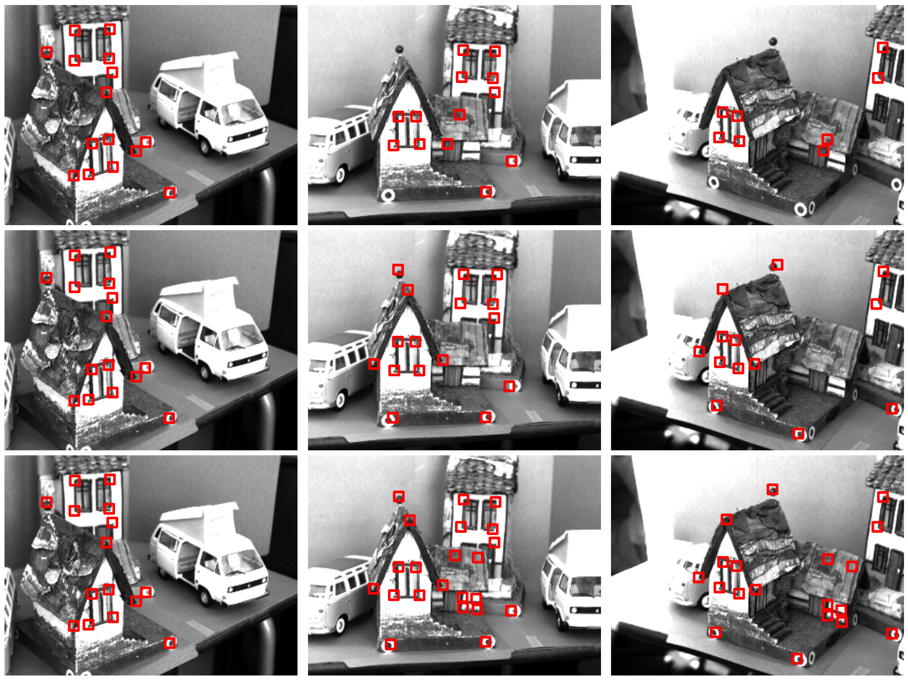

Fig. 3. Top Translation tracking of individual regions [8, 9]. Though the video sequence many patches are lost. **Middle** A projective 3D model is used to relate the regions, and provide more stable tracking through the whole sequence. **Bottom** An Euclidean model relates the region, and also allow the introduction of new regions. `video2` [1]

color channels. Hence any misalignment can be seen as color bands between the white and black squares. We measured the width of these bands and found that for the Euclidean model with homography warps on average misalignments were less than 1 pixel, and worst case over several hundred frames was 2 pixels. The 3D model will constrain the relative location of the regions, and inaccuracies in the estimated model causes the (small) misalignments we observed. In the case of no 3D model and tracking with homography warps alone the regions would eventually lose track (as in the previous house sequence). But, for the frames when they converged, the alignment was very good, with an error significantly less than a pixel. This is to be expected since the homography parameters allow exactly the freedoms needed to warp planar surfaces into any camera projection.

6 Discussion

We have shown how a 3D scene model, estimated from images alone, can be integrated into SSD region tracking. 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.

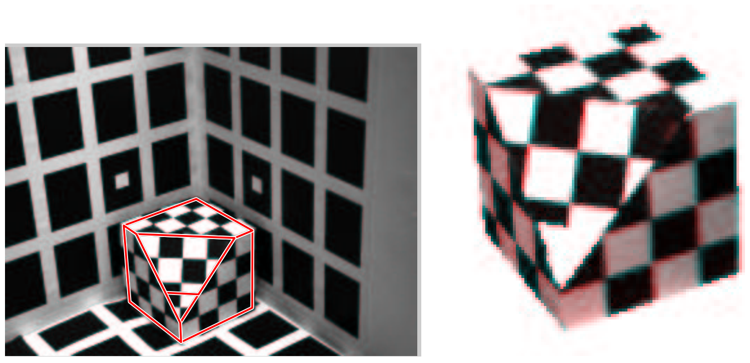

Fig. 4. Accuracy experiment. **Left** Scene image. Right current image and template superimposed indicate accurate alignment `video3` [1]

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.
- On the other hand, 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 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 and either an Euclidean or projective 3D model this gives almost as good tracking as the homography + 3D model, and makes execution somewhat faster.

Convergence in the last case (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 present 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 is started 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.

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.

References

1. On-line mpeg movies of the experiments are available. See `videoX` at http://www.cs.ualberta.ca/ dana/SCVMA04.
2. Lowe, D.: Fitting parameterized three-dimensional models to images. PAMI **13** (1991) 441–450
3. Marchand, E., Bouthemy, P., Chaumette, F.: A 2d-3d model-based approach to real-time visual tracking. IVC **19** (2001) 941–955
4. Drummond, T., Cipolla, R.: Real-time visual tracking of complex structures. PAMI **24** (2002) 932–946
5. Armstrong, M., Zisserman, A.: Robust object tracking. In: Second Asian Conference on Computer Vision. (1995) 58–62
6. Toyama, K., Hager, G.: Incremental focus of attention for robust vision-based tracking. IJCV **35** (1999) 45–63
7. Szeliski, R.: Video mosaics for virtual environments. IEEE Computer Graphics and Applications (1996) 22–30
8. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Int. Joint Conf. on Artificial Intelligence. (1981)
9. Hager, G., Belhumeur, P.: Efficient region tracking with parametric models of geometry and illumination. PAMI **20** (1998) 1025–1039
10. Baker, S., Matthews, I.: Lucas-Kanade 20 Years On: A Unifying Framework. Technical Report CMU-RITR02-16 (2002)
11. Jurie, F., Dhome, M.: Hyperplane approximation for template matching. PAMI **24** (2002) 996–1000
12. Gleicher, M.: Projective registration with difference decomposition. In: CVPR97. (1997) 331–337
13. Horn, B.: Computer Vision. MIT Press, Cambridge, Mass. (1986)
14. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2000)
15. T.Werner, T.Pajdla, M.Urban: Practice of 3d reconstruction from multiple uncalibrated unorganized images. In: Czech Pattern Recognition Workshop. (2000)
16. Triggs, W.: Auto-calibration and the absolute quadric. In: CVRP. (1997) 609–614
17. Cobzas, D., Jagersand, M.: A comparison of viewing geometries for augmented reality. In: Proc. of Scandinavian Conference on Image Analysis (SCIA 2003). (2003)