Tracking and Predictive Display for a Remote Operated Robot using Uncalibrated Video

Dana Cobzas
INRIA Rhone-Aples
Montbonnot 38334, France
Dana.Cobzas@inriaples.fr

Martin Jagersand
Computing Science,University of Alberta
Edmonton, T6G2E8, Canada
jag@cs.ualberta.ca

***Abstract*— Delays in the visual feedback can seriously impact operator performance in telerobotics. In predictive display synthesized visual feedback is rendered immediately in response to operator motions. In this paper we present a system using a geometric and appearance model that is captured using structure-from-motion by an uncalibrated camera. The geometric model is integrated into a registration-based tracking algorithm that allows stable tracking of full 3D pose of the robot. Experimentally we show that predictive scene views can be rendered with both high visual fidelity and metric accuracy.**

I. Introduction

A main challenge in telerobotics is to accurately convey the situation at a remote worksite to the operator. The efficient solution of many telerobotics tasks demand both quantitative and qualitative information. Human operators can quickly judge situations and perform actions based on qualitative information in video streams. However, performance degrades with delays as short as 0.4 seconds [10]. Additionally, human judgment of distance made from watching monitors or using head mounted displays can be distorted [19]. This suggests that the augmentation of the operator environment with metric maps of the robot movements is beneficial.

Developments in predictive display through the latest decade have seen systems going from augmenting delayed video with simple wireframe drawings [16], through ones based on displaying an appropriate 2D area of a larger image plane, or panoramic view [3], to systems based on various forms of 3D models. Recently Barth *et al.* [4] used a calibrated stereo rig on a mobile robot to acquire and use a 3D model for predictive display, and Yerex *et al.* [20] did predictive display from uncalibrated video using an affine linear camera approximation. In this paper we present a method and system to model, track and render predictive display from uncalibrated video under a full (non-linear) perspective model.

In traditional model based tracking systems a 3D pose computation is done by relating 2D image feature positions with an a-priori 3D model [13], [15], [7]. However, this is impractical in unstructured environments often encountered in mobile robotics. Additionally, the feature detection is relatively decoupled from the pose computation that makes the correspondence between model and current image features challenging. A different approach named registration based tracking is to align 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. The alignment problem is solved using numerical optimization, where a search direction is obtained from image derivatives [14], [8], [2]. One of the disadvantages of the SSD tracking methods with respect to robotics applications is that the position is tracked in a 2D space (image plane). In this paper we extended the traditional SSD tracking by imposing a global 3D model that will allow tracking full 3D position of the robot required for the predictive view generation. This method gives tracking more stability as compared to the traditional 2D SSD tracking [5]. We also show how the same idea of image variability can be used to generate a view-dependent texture that will correctly render the model from a new viewpoint in the predictive display. The main contributions of the paper include:

- We use a composite model with a sparse acquired geometry coarsely representing the scene, and an appearance based *dynamic texture* representing fine scale detail, with the property to modulate a time varying view dependent texture to correct the sparse geometric model.
- 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.
- We integrate the tracking and dynamic texture rendering into a real-time predictive display system.

II. System Overview and Model

Consider the tele-robotics setup depicted in Figure 1 where an operator controls a remote robot. The remote scene is viewed by an uncalibrated camera mounted on the robot. Our system is designed to provide the operator with both immediate synthesized video feedback (predictive display) and accurate metric information (tracking and localization). The basis of our system is a model with

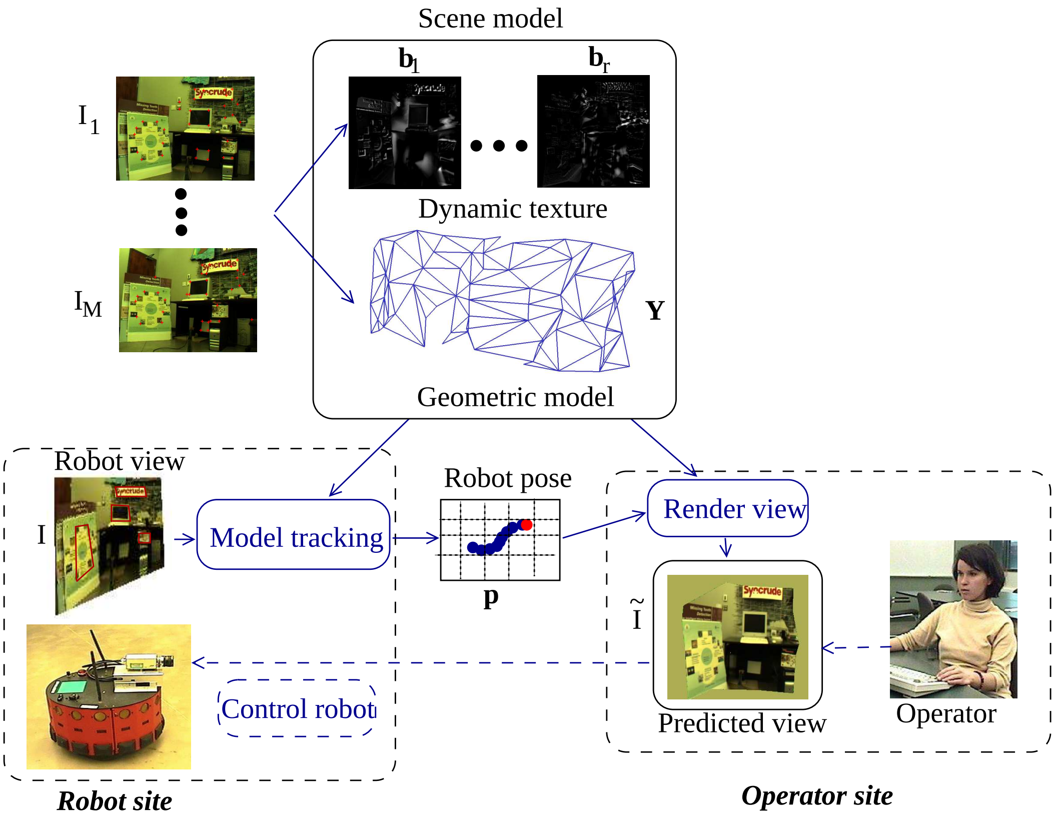

Fig. 1. Overview of the tracking with predictive display system. The robot pose is tracked using the model-based tracking algorithm from Section III. The 3D position is transmitted to a remote operator, composed with the current (desired) operator motion and an instant rendered view from robot's perspective is generated using the dynamic texture model

two types of scene information. On a macroscopic level a geometric model represents coarse scene structure, and on a microscopic level an differential appearance basis represents the time/view variability of both texture and fine-scale geometry. This basis is used both to drive the tracking pose update at the robot site and provide view-dependent texturing at the operator site.

The model is acquired from the robot uncalibrated video. A standard Euclidean camera model (in homogeneous coordinates) relates $i = 1 \ldots n$ 2D image points $\mathbf{y}_{ti} = [u, v, 1]^T$ to the corresponding 3D scene points $\mathbf{Y}_{ti} = [x, y, z, 1]^T$ so that for each image I_t, at time t the reprojection property holds:

$$\lambda \mathbf{y}_{ti} = P_t \mathbf{Y}_i = K[R|\mathbf{t}]\mathbf{Y}_i, \quad i = 1 \ldots n \tag{1}$$

where K is the camera matrix, $R = R_x(\alpha_x)R_y(\alpha_y)R_z(\alpha_z)$ the rotation matrix, $\mathbf{t} = [t_x, t_y, t_z]^T$ the translation vector, and λ the homogeneous scale. The area between the points is divided into planar facets. For tracking a few salient quadrilaterals over planar scene surfaces are selected, while for rendering a triangulation of the complete scene is sent to a standard graphics card. Hence the tracking and pose estimation is driven by only the most salient parts of the video stream.

Geometric and image intensity change are related differentially in the appearance basis M. For example, in the simple case of 2D image plane translation this relationship is expressed in the well-known optic flow constraint $M\Delta\mathbf{y} = \Delta I$, where for 2D $M = [\frac{\partial I}{\partial u}, \frac{\partial I}{\partial u}]$ In 2D tracking given temporal image differences ΔI the incremental geometric differences are accumulated $\mathbf{y}_{t,i} = \mathbf{y}_{t-1,i} + \Delta\mathbf{y}$ to follow how an image point moves over the scene. While 2D tracking works for pure image translations[1] it fails in the case of general 3D camera movement. In the next section we develop the mathematics for a 3D tracking, which using a different (higher dimensional) basis M computes the 6D camera pose change $\Delta\mathbf{p}$ from temporal intensity differences. Section IV describes how the 3D geometric model is computed from the tracked points using uncalibrated Structure-From-Motion (SFM).

At the operator site the geometry and appearance model is used to render synthesized scene views immediately in response to motion commands. Since generally SFM only provides a coarse geometric model, texturing with a conventional single image texture produces incorrect views with strong artifacts. Instead of a single texture we use

[1] Indeed we use 2D tracking for the first frames to bootstrap 3D tracking, see Section VI.

an appearance basis $\hat{M}$ extended (compared to M) to also capture non-planar (parallax) variation. A new time-varying *dynamic texture* is formed by modulating $T = \hat{M}\mathbf{z}$, and then warped onto the reprojected geometry to render predicted scene views, see Section V. Section VI describes how the parts are integrated into a distributed software system. At the remote site tracking and SFM is first used to acquire a model of geometry Y and appearance $\hat{M}$. The model is transmitted to the operator site. Subsequently only new camera poses $\mathbf{p}$ are transmitted, and the model is used to render operator visual feedback. In the last two sections we describe experimental results from a mobile robotics application and conclude with a discussion and outlook.

III. SSD Model-based tracking

The goal of the tracking algorithm is to compute/track how the 3D camera pose $\mathbf{p}$ changes over time. We developed an image-based formulation where the pose change $\Delta\mathbf{p}$ is computed directly from the intensity variation in time of a set of salient quadrilateral regions. For a mathematical intuition, let $T = I_0$ be the template image, and I_t the current image. We seek to find M such that:

$$\Delta\mathbf{p} = \Delta[\alpha_x, \alpha_y, \alpha_z, t_x, t_y, t_z] = M^{-1}(T - I(W_{t-1}) \quad (2)$$

Here W_{t-1} is a warp that registers the current image with the template frame based on the previous frame pose, $\mathbf{p}_{t-1}$. The $\Delta\mathbf{p}$ serves to update the pose estimate from frame I_{t-1} to I_t. These types of algorithms are referred as registration based tracking or SSD tracking in the computer vision literature [2]. The original formulation tracks a 2D position in image space and it is not applicable for a mobile robot that is controlled in 3D Euclidean space. We developed a new algorithm [5] that extends the regular 2D SSD tracking by involving the full 3D model. Next follows a detailed description of the algorithm. Refer to Figure 2 for an illustration of the tracking approach.

Each quadrilateral region $\mathcal{R}_k$ is defined by 4 control points $Y_k = [\mathbf{Y}_{k1}, \mathbf{Y}_{k2}, \mathbf{Y}_{k3}, \mathbf{Y}_{k4}]$. 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. For each quadrilateral a plane-to-plane warp W registers it's location in the image I_t with the template T. This warp function is composed of a 3D to 2D geometric projection of the control points $\mathbf{y}_{tkj} = P(\mathbf{p}_t)\mathbf{Y}_{kj} = K[R|\mathbf{t}]\mathbf{Y}_{kj}, j = 1, 4,$ and a planar warp function for the interior points. Here a projective homography, (Appendix, Equation 13) is the geometrically correct plane-to-plane transform. Hence the composed warp function is $W(\mathbf{x}_k; \mu(\mathbf{p}_t, Y_k))$, where μ are the 2D warp parameters determined by the projection of the region control points. Note that the 3D model motion is global but each individual local region has a different 2D motion warp W_k. To simplify notation in the following the 2D warp is written $W(\mathbf{x}_k; \mu(\mathbf{p}_t))$.

Under the common image constancy assumption used in motion detection and tracking [11] the tracking problem can be formulated as finding $\mathbf{p}_t$ such that:

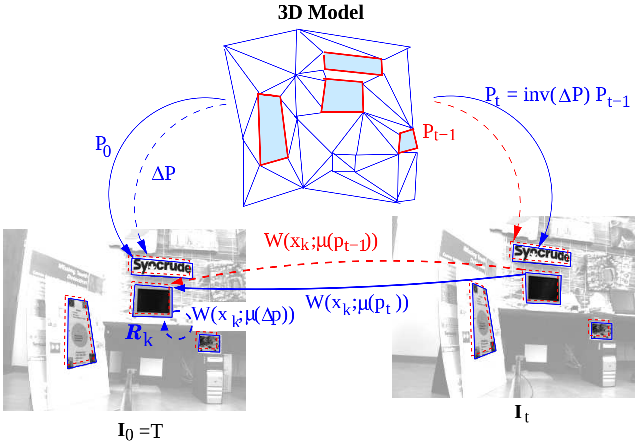

Fig. 2. Overview of the 2D-3D tracking system. In standard SSD tracking 2D surface patches are related through a warp W between frames. In our system a 3D model is estimated (from video alone), and a global 3D pose change ΔP is computed, and used to enforce a consistent update of all the surface warps.

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

The new motion parameters are obtained by function composition, (since adding projection matrices is not geometrically meaningful) $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}$, and are computed by minimizing the image residual ΔT with respect to $\Delta\mathbf{p}$:

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

Computationally it is more efficient to compute the derivatives once on the static template rather than the time varying image. This yields an algorithm in the so called inverse compositional class[2]. The goal is to find $\Delta\mathbf{p}$ that minimizes:

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

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

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

where $\text{inv}(\Delta P) = K[R'| - R'\mathbf{t}]$ inverts the 3D motion parameters in a geometrically valid way. As a consequence, if the 2D warp W is invertible, the individual warp update is (see Figure 2):

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

Performing a Taylor expansion of Equation 5 gives:

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

Define the 3D pose of the template image as zero, $T = T(W(\mathbf{x}_k; \mu(\mathbf{0})))$. Denoting $M = \sum_k \sum_x \nabla T \frac{\partial W}{\partial \mu} \frac{\partial \mu}{\partial \mathbf{p}}$, Equation 8 can be rewritten as:

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

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 Equation 9.

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. A detailed derivation of M is presented in Appendix.

IV. Geometric model from uncalibrated video

Several techniques for extracting scene structure and camera motion from uncalibrated video have been developed in the past decade [9]. Most of these methods assume a static scene and estimate the structure from a set of corresponding points. Depending on the camera model and calibration data the estimated model can vary from a projective, affine, or to a metric or 3D Euclidean model. A mobile robot is controlled in a metric world and therefore the tracking and model used for tracking have to be upgraded to Euclidean. We chose a stratified approach to recover the metric model (projective reconstruction that is upgraded to a metric structure using automatic self-calibration).

Recall projection Equation 1, $y_{ti} = P_t\mathbf{Y}_i,\ \ i = 1, n\ \ t = 1, m$. In an uncalibrated setup P_t is the 3×4 projection matrix that has 11 DOF. A calibrated projection matrix can be decomposed as $Pe_t = K[R|\mathbf{t}]$. There are in general 15 DOF between the projective reconstruction and its corresponding Euclidean one that are encoded by a projective transformation H s.t. $Pe_t = P_t H$.

Several well known estimation algorithms have been developed 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. al* [18] 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. The global structure $\mathbf{Y}_i$ and the rest of the projection matrices P_t are recovered by integrating new views through the trilinear tensor between the new and two previous views. Assuming that the cameras have zero skew and aspect ratio and the principal point is approximately known, the Euclidean projections $Pe_t = P_t H$ are recovered using self-calibration [17]. The resulting metric structure is $H^{-1}\mathbf{Y}_i$. 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.

V. Dynamic texture

In convectional graphics textures are represented as an image on a 3D mesh. This assumes that the geometry accurately models the true underlying scene. As mentioned before, the SFM geometric model only approximates the scene geometry so the texture changes from view to view. We model this appearance change with a time-varying *dynamic texture*.

From the training set of m images I_t we obtain a set of corresponding texture T_t by warping the image model points y_{ti} to canonical (here mean) position $w_t = \sum_{t=1,m} y_{ti}$. If the SFM geometry would be accurate the texture images would be constant. In our case they vary and we can smoothly parametrize this variation using a basis $\hat{M}$ such as:

$$\mathbf{T}_t = \hat{M}\mathbf{z}_t,\ \ t = 1, m. \tag{10}$$

It has been shown [6], [12] that the modulation coefficients vary smoothly with viewing angle and that the basis $\hat{M}$ captures the geometric and non-geometric (e.g. photometric) texture variability up to a first order model of true intensity variation.

The analytical derivation of $\hat{M}$ starts with the same principle of image constancy under a warp $W(\mathbf{x}, \mu)$ as in the tracking section (Equation 3). The error introduced by the geometry can be viewed as a pixel rearrangement due to a perturbed warp $W(\mathbf{x}, \hat{\mu})$, We study the intensity variation introduced by this warp $\Delta T = T(\mathbf{x}_k) - I_t(W(\mathbf{x}_k, \hat{\mu}))$. Denoting $\hat{\mu} = \mu + \Delta\mu$ and approximating the image residual with its first order Taylor approximation (as in the tracking) we get (dropping t and k):

$$\begin{aligned} \Delta T &= T(\mathbf{x}; \Delta\mu) - I(\mathcal{W}(\mathbf{x}; \mu)) \\ &= T + \nabla T \tfrac{\partial \mathcal{W}}{\partial \mu}\Delta\mu - I(\mathcal{W}(\mathbf{x}; \mu)) \\ &\approx \nabla T \tfrac{\partial \mathcal{W}}{\partial \mu}\Delta\mu \end{aligned} \tag{11}$$

Residual errors due to imperfect tracking or SFM cause a planar shift in the texture coordinates. This is modeled by the same linear subspace as used in tracking, and for the analytic form of the warp derivatives see appendix, Equation 16. Here it is sufficient to note that they are spanning an 8-dimensional subspace in which (part of) our texture variability will lie.

$$\Delta T = \left[\frac{\partial \mathbf{T}}{\partial u}, \frac{\partial \mathbf{T}}{\partial v}\right] \frac{\partial \mathcal{W}}{\partial \mu}\Delta\mu = [\mathbf{b}_1 \ldots \mathbf{b}_8] \begin{bmatrix} z_1 \\ \vdots \\ z_8 \end{bmatrix} = \hat{M}_h \mathbf{z}_h \tag{12}$$

For the tracking, quadrilaterals were selected specifically to be planar in the scene. In rendering we need to render all model facets, planar or not. This introduces a further 2 dimensions of parallax variability to $\hat{M}$. Finally, a 9-dimensional linear subspace will cover light variation. Analytical derivation of these are similar and details can be found in [12].

In real world scenes it is unpractical to calculate the texture basis analytically as some information is incomplete or missing (eg. for parallax variability we need a dense depth map, and for light the surface normals). We instead estimate the texture from image statistics in the original set of textures. Knowing that texture variability can be compactly approximated by a basis $\hat{M}$ of dimension $8+2+9=19$ we extract this linear subspace (or a slightly larger one) $\tilde{M}=[\tilde{z}_1 \dots \tilde{z}_r]$ using PCA from the original texture images.

Rendering a new view

New views are rendered by modulating the texture basis $\tilde{M}$ and warping it to the projected geometry. The modulation coefficients $\mathbf{z}$ are calculated by interpolating the texture coefficients $\mathbf{z}_t$ from the training set for the new camera pose. For achieving real time rendering we implemented the texture blending in hardware using nVidia register combiners [6].

VI. Predictive display system

The model based tracking algorithm has been incorporated into a predictive display system that is using the geometric model and dynamic texture to generate synthetic images of the current robot view for a remote operator (Figure 1). In the *bootstrapping phase* the geometric model and dynamic texture basis are generated from a set of training images and in the *tracking phase* the geometric model is tracked and a predictive view is generated.

Bootstrapping phase

1) Several salient surface patches are selected in a non-planar configuration from a scene image and tracked in about $m \approx 100$ frames using standard (2D image-plane) SSD trackers as in [2], [8].
2) From the tracked points $\mathbf{y}_{it}$, a 3D model points $Y_i,\ i=1,m$ are computed and tessellated into quadrilateral regions. The dynamic texture basis $\tilde{M}$ is estimated from the training images warped to a standard shape. The geometric model is stored and texture basis is transmitted to the operator site.
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 [9]) 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.

Tracking and predictive display phase

For each time step:

robot site:

4) track robot pose $\mathbf{p}=(\alpha_x,\alpha_y,\alpha_z,t_x,t_y,t_z)$
5) send position to operator site

user remote site:

6) add current operator motion command
7) project geometric model in new location
$\mathbf{y}_i = K[R(\alpha_x,\alpha_y,\alpha_z)|\mathbf{t}(t_x,t_y,t_z)]\mathbf{Y}_i$
8) compute the dynamic texture T for the new location
$T=\tilde{M}\mathbf{z}(\mathbf{p})$
9) warp T onto the projected structure and display
10) send motion command to robot site

During tracking patches that become occluded are detected and removed. Similarly new patches visible only in new views are added and incorporated in the model 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.

VII. Experimental results

To evaluate the tracking and predictive display system, we captured a model of a research lab. We used the model based tracking algorithm from Section III to recover camera location along two motion trajectories. Figure 3 shows the planar patches that are being tracked in two positions along the trajectories.

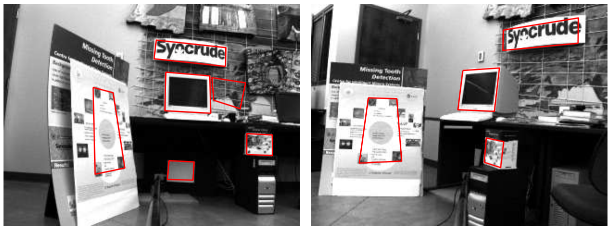

Fig. 3. Tracking planar patches. The model also allows detection and removal of occluded regions.

The first trajectory was a straight line in the horizontal plane of about 1m. Figure 5 (left) illustrates the recovered trajectory. For measuring the accuracy of the tracking algorithm we calibrated the 3D room model assuming some given real dimensions (here the size of the monitor screen) so we could get the translation in meters. We found that the 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, that result in an error of 0.08 m with respect to the measured ground truth. There was no camera rotation along the first trajectory, that corresponded to the measured rotation (error was less than 1 degree on average).

We tracked the second trajectory along two perpendicular lines in the horizontal plane. In this experiment, the physical motion was not 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 5) 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 projective properties e.g. deviation

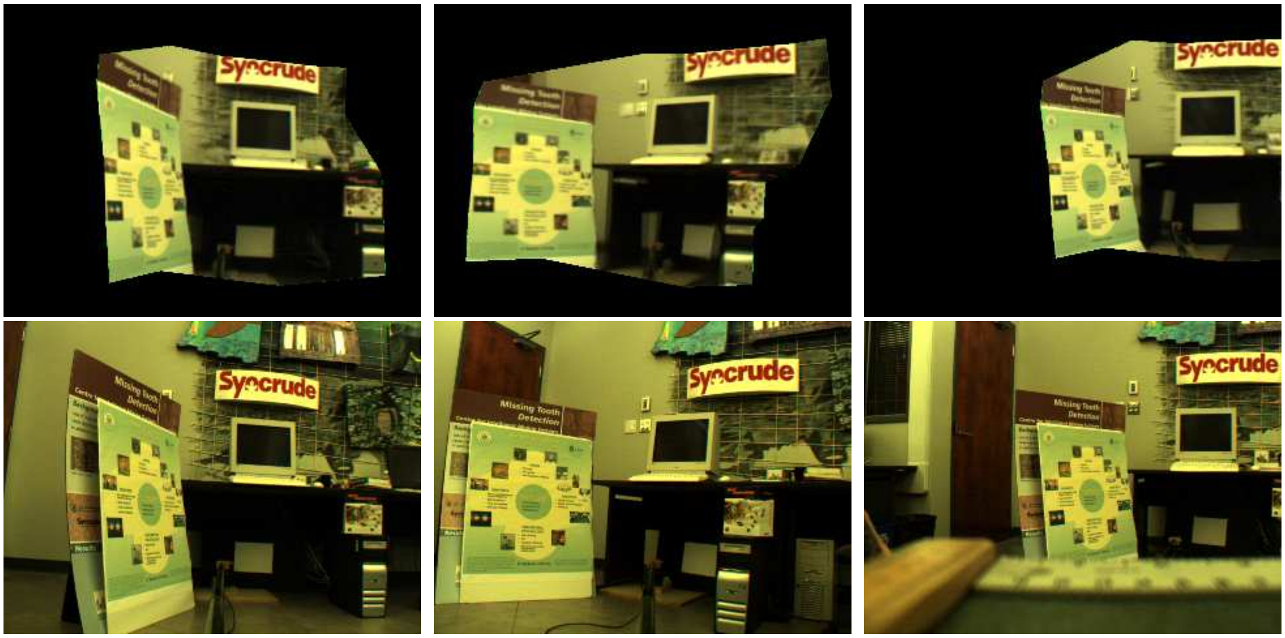

Fig. 4. Examples of predictive views (top row) and the corresponding actual images (bottom row). Results for the entire track sequence are shown in `video1` [1].

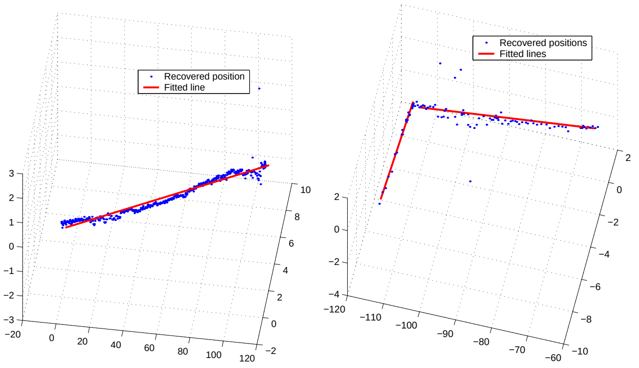

Fig. 5. Recovered positions for the straight line trajectory (left) and the 2 perpendicular lines trajectory (left). The red line are the fitted 3D lines to each line segment.

from lines, planes) is higher that the accuracy in measured distances. This is due to the difficulty in calibrating a projective structure from only natural scene data.

For each recovered position we generated the view predicted from the model (`video1` [1]). Figure 4 (top row) shows examples of rendered views along the two trajectories using the dynamic texture model. Comparing them with the real views (bottom row), we notice that the dynamic texture model produces good quality renderings that realistically represent the actual images. The limited field of view is due to the viewing frustum defined in the original training sequence that was uses in building the model.

VIII. Discussion

A main consideration in designing robotic tele-operation systems is the quality of sensory feedback provided to the human operator. For effective tele-operation the operator must get the feeling of being present in the remote site and get immediate visual feedback from his or her motion commands. We showed how a geometric model can be estimated from images and then used both for stabilizing SSD tracking and to display a predictive view for the operator. For generating a realistic looking view of the remote scene, a time-varying *dynamic texture* is overlaid to the projected geometry. Our technique eliminates the need for expensive range sensors and calibrated setups to capture the remote scene geometry, and instead uses inexpensive consumer web or video cameras with a standard PC's.

The model is initialized from a training sequence but can be improved during tracking by adding/removing patches while they become visible/occluded. While the model tracking can handle large changes the dynamic texture is precomputed at the beginning and in the current implementation is not updated over time. We are developing an on-line algorithm that is using incremental PCA to automatically update the texture basis. Another limitation of the present system is the manual way of selecting new patches that can be replaced with an automatic feature detector. Our model directly relates geometric robot pose and image views, and this also can support control interfaces where the motion goal is specified in image space instead of robot motor space. One such possible intuitive interaction paradigm is tele-operating the robot by "pointing" in the image space or by dragging the model viewpoint to obtain the desired next view, and then have the robot move to this location using visual servo control.

Appendix
Computing derivatives images

We compute the variability basis from spatial derivatives of template intensities and inner derivatives of the warp. As mentioned before, the 2D warp parameters μ (homography parameters) are functions of 3D rotation and translation parameters $\mathbf{p}$, the 3D control points $\mathbf{Y}_j$ and the position of the control points in the template image $\mathbf{y}_{0j}$:

$$\mathbf{y}_{0j} = W(\mu(\mathbf{p}))(P\mathbf{Y}_j) = H\mathbf{y}_j \quad j = 1,4 \tag{13}$$

where

$$H = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 \\ \mu_4 & \mu_5 & \mu_6 \\ \mu_7 & \mu_8 & 1 \end{bmatrix} \tag{14}$$

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 homography parameters μ are directly computed from the warp expression

$$W(\mathbf{x}_k; \mu) = H\mathbf{x} \tag{15}$$

$$\frac{\partial W}{\partial \mu} = \begin{bmatrix} u & 0 & v & 0 & 1 & 0 & -\frac{uc_2}{c_1} & -\frac{vc_2}{c_1} \\ 0 & u & 0 & v & 0 & 1 & -\frac{uc_3}{c_1} & -\frac{vc_3}{c_1} \end{bmatrix} \tag{16}$$

where $\mathbf{x} = [u, v]^T$, $c_1 = 1 + \mu_7 u + \mu_8 v$, $c_2 = \mu_1 u + \mu_3 v + \mu_5$, and $c_3 = \mu_2 u + \mu_4 v + \mu_6$.

However, the explicit dependency between the 2D parameters μ and the 3D motion parameters $\mathbf{p}$ is in general difficult to obtain, but Equation 13 represents their implicit dependency, so $\frac{\partial \mu}{\partial \mathbf{p}}$ are computed using the implicit function theorem. Equation 13 can be written in the form

$$A(\mathbf{p})\mu(\mathbf{p}) = B(\mathbf{p}) \tag{17}$$

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 $\mathbf{y}_j$.

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

$$\frac{\partial A}{\partial p}\mu + A\frac{\partial \mu}{\partial p} = \frac{\partial B}{\partial p} \tag{20}$$

For a given $\mathbf{p}$ value μ can be linearly computed from Equation 17 and then $\frac{\partial \mu}{\partial p}$ is computed from Equation 20.

References

[1] On-line mpeg movie of the experiments. http://www.inrialpes.fr/movi/people/Cobzas/ICRA05/video1.mpg

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

[3] J. Baldwin, A. Basu, and H. Zhang. Panoramic video with predictive windows for telepresence applications. In *Int. Conf. on Robotics and Automation*, 1999.

[4] M. Barth, T. Burkert, C. Eberst, N. Stöffler, and G. Färber. Photo-realistic scene prediction of partially unknown environments for the compensation of time delays in presence applications. In *Int. Conf. on Robotics and Automation*, 2000.

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

[6] D. Cobzas, K. Yerex, and M. Jagersand. Dynamic textures for image-based rendering of fine-scale 3d structure and animation of non-rigid motion. In *Eurographics*, 2002.

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

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

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

[10] R. Held, A. Efstathiou, and M. Greene. Adaptation to displaced and delayed visual feedback from the hand. *J. Exp Psych*, 72:871–891, 1966.

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

[12] M. Jagersand, D. Cobzas, and K. Yerex. Modulating view-dependent textures. In *Eurographics short presentation*, 2004.

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

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

[15] 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.

[16] T. B. Sheridan. Space teleoperation through time delay: Review and prognisis. *IEEE Tr. Robotics and Automation*, 9, 1993.

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

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

[19] P. Willemsen, M. Colton, S. Creem-Regehr, and W. Thompson. The effects of head-mounted display mechanics on distance judgments in virtual environments. In *ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*, pages 35–48, 2004.

[20] K. Yerex, D. Cobzas, and M. Jagersand. Predictive display models for tele-manipulation from uncalibrated camera-capture of scene geometry and appearance. In *Proc. of ICRA*, 2003.