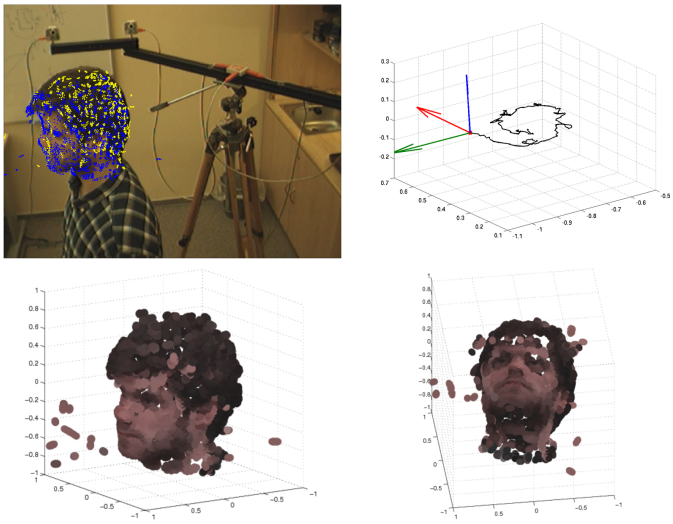
Multiview 3D Tracking with an Incrementally Constructed 3D Model

Abstract

We propose a multiview tracking method for rigid objects. Assuming that a part of the object is visible in at least two cameras, a partial 3D model is reconstructed in terms of a collection of small 3D planar patches of arbitrary topology. The 3D representation, recovered fully automatically, allows to formulate tracking as gradient minimization in pose (translation, rotation) space. As the object moves, the 3D model is incrementally updated. A virtuous circle emerges: tracking enables composition of the partial 3D model; the 3D model facilitates and robustifies the multiview tracking.

Experimentally, we demonstrate that the interleaved track-and-reconstruct approach successfully tracks a 360 degrees turn-around and a wide range of motions. Monocular tracking is also possible after the model is constructed. Using more cameras, however, significantly increases stability in critical poses and moves.



1 Introduction

Existing multiview approaches mostly track objects as blobs. Blob representation assumes that the appearance of the object does not significantly change when the object spins. Global object position is sought and the methods do not attempt to recover the *orientation* of the object [1, 2].

Several approaches build elaborated 3D models from many cameras. However, the methods do not really track objects and heavily rely on carefully constructed and expensive setup and require special scene arrangement since they are based on scene/object segmentation [3, 4, 5, 6]. Würmlin et al. [6] propose dynamic 3D point samples for streaming 3D video. This point based representation somehow resembles our model. However, the method does not track object and needs many cameras and very precise pixel-wise motion segmentation.

Most of the *model-based tracking* methods use off-line prepared 3D model. Comprehensive survey of such methods has been recently published by Lepetit et al. [7]. Vacchetti et al. [8] propose a stable tracker based on matching with keyframes. The method demonstrates impressive results on out-of-plane rotation data. Still, it cannot track complete spin of the object and needs off-line human intervened composition of keyframes which are essential for the stability. Muñoz et al. [9] suggest a method that can track even deformable objects. Their model is composed of small textured planar patches, set of shape bases, and set of texture bases. The tracking procedure needs a reference image and optimises over local shape deformations, colour/texture changes and overall motion. Results on real data, however, demonstrate successful tracking only of small variations in object pose and negligible local deformations.

We propose a combined method that track objects in 3D and constructs a point based appearance model simultaneously. The primary interest is the object tracking and detection. The model is rather simple. However simple, the model is rich enough for recovering *orientation* of the object. The tracking can follow a complete object turn-around. Rothganger et al. [10] compose a 3D model from small planar patches. The patches are reconstructed from multiview correspondences. The authors photograph an object from several viewpoints, find corresponding image patches by affine covariant feature matching and reconstruct the patches in 3D. In fact, it would be possible to use this model in our tracking. Any complete off-line built model [11] could be used, too.

Cobzas and Jagersand [12] propose monocular, registration-based, 3D camera tracking of small planar patches. The 3D planar patches are estimated from the data. Although the formulation of the tracking resembles our one in spirit there are several differences. The patch based

model is initialised at the beginning of the sequence (about 100 frames) by using a standard 2D patch based tracker. Then the algorithm switches to track and refine the model using 3D model based tracking. They track the camera, assuming a rigid scene hence, they can assume colour constancy. Our method builds the model from the very beginning of the sequence. The tracked object changes its position and orientation w.r.t. to light sources. In this case, colour constancy cannot be assumed even for Lambertian surfaces and our method reflects this.

2 3D tracking

An object O is modelled as a triplet (X, T, N) where X is a set of 3D points, $T : X \to \mathcal{R}$ assigns albedo and $N : X \to \mathcal{S}^3$ a normal to each point $\mathbf{x}$ in X, where $\mathcal{S}^3$ is a sphere.

Assuming rigidity, the motion of points $\mathbf{x}$ in X between two time instances t_1 and t_2 is

$$\mathbf{x}^{t_2} = \mathtt{R}\mathbf{x}^{t_1} + \mathbf{d},$$

where $\mathtt{R}$ represents rotation and $\mathbf{d}$ is translation. When the rotation is small [13] (e.g. between two consecutive video frames) the motion equation simplifies to

$$\mathbf{x}^{t} = (\mathtt{I} + \mathtt{D})\mathbf{x}^{t-1} + \mathbf{d}, \tag{1}$$

where the rotation matrix $\mathtt{R}$ was replaced by an antisymmetric matrix $\mathtt{D}$ and an identity matrix $\mathtt{I}$. Matrix $\mathtt{D}$ is defined by three parameters $\mathbf{u} = [D_1, D_2, D_3]^T$;

$$\mathtt{D} = \begin{bmatrix} 0 & D_3 & -D_2 \\ -D_3 & 0 & D_1 \\ D_2 & -D_1 & 0 \end{bmatrix}.$$

3D tracking is defined as the process of finding motion parameters $\mathtt{D}, \mathbf{d}$ which minimize the following image dissimilarity

$$\sum_{\mathbf{x} \in X} \left[T\left(\mathbf{x}^{t-1}\right) - I\Big(f\left(\mathbf{x}^{t}\right)\Big) \right]^2, \tag{2}$$

where $I : \mathcal{R}^2 \to \mathcal{R}$ assigns intensity to each point. The projection function f maps 3D points to image coordinates, $f : \mathcal{R}^3 \to \mathcal{R}^2$ and depends on internal and external parameters of camera, see Appendix A for details. The generalization to color and more cameras is discussed in Section 4.

Substituting from equation (1) for $\mathbf{x}^t$ in the dissimilarity function (2) and simplifying notation by introducing $\mathbf{x}^{t-1} = \mathbf{x}$, a cost function in six unknowns is obtained

$$J(\mathbf{u}, \mathbf{d}) = \sum \Big[T\left(\mathbf{x}\right) - I\Big(f\left(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d}\right)\Big) \Big]^2, \tag{3}$$

where the sum is over all $\mathbf{x} \in X$ as in (2); starting from (3) the summation range is omitted for brevity. We seek motion

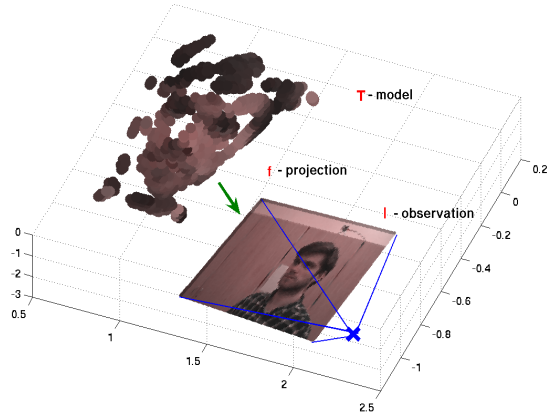


Figure 1: Model (template) T is projected by projection function f and compared to the current observation I.

parameters $\mathbf{u}$ and $\mathbf{d}$ that minimize dissimilarity $J(\mathbf{u}, \mathbf{d})$. At the minimum, the partial derivatives with respect to all variabless must be zero,

$$\frac{\partial J(\mathbf{u}, \mathbf{d})}{\partial \mathbf{d}} = \mathbf{0}, \quad \frac{\partial J(\mathbf{u}, \mathbf{d})}{\partial \mathbf{u}} = \mathbf{0},$$

which yields the following two triplets of equations

$$\sum \Big[T(\mathbf{x}) - I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big) \Big] \frac{\partial I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big)}{\partial \mathbf{d}} = \mathbf{0}, \tag{4}$$

$$\sum \Big[T(\mathbf{x}) - I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big) \Big] \frac{\partial I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big)}{\partial \mathbf{u}} = \mathbf{0}, \tag{5}$$

There is no closed-form solution for $(\mathbf{u}, \mathbf{d})$. We therefore apply Newton-Raphson minimization, approximating $I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big)$ by its first-order Taylor expansion

$$I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big) \approx I\big(f(\mathbf{x})\big) + \mathbf{g}^T(\mathtt{D}\mathbf{x} + \mathbf{d}), \tag{6}$$

where

$$\mathbf{g}^T = I'^T\big(f(\mathbf{x})\big) f'(\mathbf{x}); \tag{7}$$

$I' : \mathcal{R}^2 \to \mathcal{R}^2$ is the gradient of image I and $f' : \mathcal{R}^3 \to \mathcal{R}^{2\times 3}$ is the Jacobian of the projection function f.

Differentiating the linear approximation (6) leads to

$$\frac{\partial I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big)}{\partial \mathbf{d}} \approx \mathbf{g}^T\mathbf{d}, \tag{8}$$

$$\frac{\partial I\big(f(\mathbf{x} + \mathtt{D}\mathbf{x} + \mathbf{d})\big)}{\partial \mathbf{u}} \approx \frac{\partial \mathbf{g}^T\mathtt{D}\mathbf{x}}{\partial \mathbf{u}}. \tag{9}$$

Applying the approximations (8), (9), equations (4), (5) are simplyfied to

$$\sum \big[T(\mathbf{x}) - I\big(f(\mathbf{x})\big) - \mathbf{g}^T\mathtt{D}\mathbf{x} - \mathbf{g}^T\mathbf{d} \big] \frac{\partial \mathbf{g}^T\mathtt{D}\mathbf{x}}{\partial \mathbf{u}} = \mathbf{0} \tag{10}$$

$$\sum \big[T(\mathbf{x}) - I\big(f(\mathbf{x})\big) - \mathbf{g}^T\mathtt{D}\mathbf{x} - \mathbf{g}^T\mathbf{d} \big] \mathbf{g} = \mathbf{0} \tag{11}$$

Simple algebraic manipulations confirms the following two identities hold

$$\begin{aligned} \mathbf{g}^T \mathtt{D}\mathbf{x} &= (\mathbf{g} \times \mathbf{x})^T \mathbf{u}, \\ \frac{\partial \mathbf{g}^T \mathtt{D}\mathbf{x}}{\partial \mathbf{u}} &= (\mathbf{g} \times \mathbf{x}), \end{aligned}$$

where $\times$ is the cross product. Equations (10) and (11) can be compactly represented as a system of six linear equations $\mathtt{A}$.

$$\mathtt{A} \begin{bmatrix} \mathbf{u} \\ \mathbf{d} \end{bmatrix} = \mathbf{b}, \tag{12}$$

where

$$\begin{aligned} \mathtt{A} &= \sum \begin{bmatrix} (\mathbf{g} \times \mathbf{x})(\mathbf{g} \times \mathbf{x})^T & (\mathbf{g} \times \mathbf{x})\mathbf{g}^T \\ \mathbf{g}(\mathbf{g} \times \mathbf{x})^T & \mathbf{g}\mathbf{g}^T \end{bmatrix}, \\ \mathbf{b} &= \sum \big[T(\mathbf{x}) - I\big(f(\mathbf{x})\big)\big] \begin{bmatrix} (\mathbf{g} \times \mathbf{x}) \\ \mathbf{g} \end{bmatrix}. \end{aligned}$$

Assuming regular $\mathtt{A}$, the sought solution, approximately minimizing equation (3), is

$$\begin{bmatrix} \mathbf{u} \\ \mathbf{d} \end{bmatrix} = \mathtt{A}^{-1}\mathbf{b}. \tag{13}$$

The 6×6 matrix $\mathtt{A}$ consists of four 3×3 sub-matrices and is block-wise symmetrical. Unknown motion parameters $\mathbf{d}$, $\mathbf{u}$ are two 3×1 vectors and $\mathbf{b}$ is 6×1.

At least six points are required for $rank(\mathtt{A}) = 6$. In practice, many more points are visible and naturally increase the robustness and stability. If the object is weakly textured image derivatives $\mathbf{g}$ may get close to zero and matrix $\mathtt{A}$ nearly singular. Texture properties needed for reliable tracking of the object are discussed in [14]. Unlike [14], we optimize over the whole object not just over a small patch, which naturally increases the robustness.

Newton-Raphson iterations are carried out until convergence or a maximum number of steps N. Experiments showed the process converged usually in $8 - 10$ iterations. Convergence may, however, require more iterations when the motion is fast, so N was set to 20.

Extension to RGB tracking is straightforward. The solution is same as (13), where $\sum$ stands for three sums over all points, all cameras where points are visible and RGB channels.

3 Compensation of Illumination Effects

Intensity recorded at model acquisition is dependent on the light sources positions. As the object moves, the set of light source visible from a point and the angle with the corresponding normal changes. We model these effect by assuming

- there are no cast shadows,
- the light sources are distant.

Under this assumption, intensities of the all points with identical normals $\mathbf{n}$ will be amplified by illuminance $E(\mathbf{n})$. Considering different illumination of each normal would add too many degrees of freedom to the optimization. Because of robustness, the points are clustered into several groups $G_1, \ldots, G_n$ accrording to their normals. Thus, the illumination effect on i-th cluster in each time can be approximated by illumination

$$\mathtt{E}_i^* = \arg\min_{\mathtt{E}_i} \sum_{\mathbf{x} \in G_i} \|\mathtt{E}_i I(f(\mathbf{x})) - T(\mathbf{x})\|_2^2. \tag{14}$$

Let us denote

$$J(\mathtt{E}_i) = \sum_{\mathbf{x} \in G_i} \|\mathtt{E}_i I(f(\mathbf{x})) - T(\mathbf{x})\|_2^2 =$$

$$\sum_{\mathbf{x} \in G_i} I^T(f(\mathbf{x}))\mathtt{E}_i^T \mathtt{E}_i I(f(\mathbf{x})) - 2T^T(\mathbf{x})\mathtt{E}_i I(f(\mathbf{x})) + T^T(\mathbf{x})T(\mathbf{x}),$$

then minimization yields the following matrix equation

$$\frac{\partial J(\mathtt{E}_i)}{\partial \mathtt{E}_i} = \sum_{\mathbf{x} \in G_i} -2T(\mathbf{x})I^T(f(\mathbf{x})) + 2\mathtt{E}_i^* I(f(\mathbf{x}))I^T(f(\mathbf{x})) = \mathtt{0} \tag{15}$$

and least square solution is

$$\mathtt{E}_i^* = \left[\sum_{\mathbf{x} \in G_i} I(f(\mathbf{x}))I^T(f(\mathbf{x}))\right]^{-1} \sum_{\mathbf{x} \in G_i} T(\mathbf{x})I^T(f(\mathbf{x})) \tag{16}$$

4 Tracking-Modeling Algorithm

Minimal configuration able to build the model must contain at least two cameras at stereo configuration. For tracking, one camera is sufficient. Our camera setup consists of several calibrated cameras where pairs are employed for stereo-based reconstructing parts of the model while all the cameras are used for minimization. As the object moves, further parts (sides) of the object are incrementally constructed.

Assuming no model available the tracking starts with a stereo-based reconstruction [15] of the visible part of the object. Albedo of the object si determined from the average of its projections used in the time of reconstruction. Once the partial model is known, it can be used for pose estimation. If a significant part of the object, which is not reconstructed, is visible in stereo pair, the reconstruction is called again. Another part of the surface is reconstructed and aligned with the existing one by the tracking.

During tracking, significant part of the reconstructed surface may become unobservable due to occlusions. In order

to track the complete 360-degrees turn we must solve the visibility problem. We decided to use fish-scales [16] representation. The fish-scales are small oriented planar patches. Knowledge of surface orientation in each fish-scale allow:

- Efficient decision between visible and unvisible parts of the model in particular cameras.
- Compensation of illumination effects.
- Decreasing influence of fishscales at accute angles by weighting. We use weights equal to cosine of the angle between surface normal and line of sight.

Fish-scales are obtained by local clustering directly from the cloud of points. Small cluster of points are replaced by ellipses with the half-axes corresponding to two main eigenvectors of their covariance matrix. The sought normal vector to the ellipse is the third eigenvector. Computation of fish-scales representation is much easier then a complete surface triangulation. Still the fish-scales are proved to be sufficient representation.

The complete algorithm can be summarized as follows:

1. Capture images
2. **Reconstruct** visible surface from the stereo and add it to the model, if neccesary.
3. **Correct the model** to compensate illumination effects. For all i and each $\mathbf{x} \in G_i$ recompute model intensity $T_{\text{compensated}}(\mathbf{x}) = \mathtt{E}_i T(\mathbf{x})$
4. **Compute pose** of the object by iterating equation (13). Fish-scales contributions are weighted according to the angle of projection.
5. **Update matrices** $\mathtt{E}_1, \ldots, \mathtt{E}_n$ and **goto 1**.

5 Experiments

The data were captured in an office. We used four firewire cameras with resolution 640×480 connected to Linux operated computers. The acquisition was TCP/IP synchronized and the setup was calibrated. Note that the total cost of the setup (without computers) is less than 500 USD and calibration is easy since a free software for automatic (self)calibration exists.

Two different sequences were used. In the *human sequence* the person makes a variety of motions. The individual turns, shakes and tilts the head, moves up and down and away from the scene. The camera setup consists of two close cameras for stereo reconstruction and two other cameras spanning together approximately a half-circle. The human turns of 360-degrees during the first 310 frames. The rest 630 frames show wide range of motions and leaving the scene.

The *book sequence* poses slightly different challenges. The book is a relatively thin object and in some poses the dominant plane is invisible. The camera setup consists of three cameras located near each other. Two of them are used as stereo and one additional is used with these for tracking. The model of the book is again incrementally constructed from a stereo pair and tracked over all cameras.

Despite of the challenges, in both sequences objects are tracked successfully and their shapes are correctly reconstructed. We provide experiments to present accuracy and robustness of multiview tracking and reasonable usability for monocular tracking. In section 5.1 we show, that the accuracy of multiview tracking is sufficient for incremental model construction without an additional alignment. Section 5.2 compares monocular and polynocular tracking. We show that the monocular tracking usually provides poses which looks corectly only in the used camera. This results are useful for, e.g., augmented reality applications. The estimation of a real 3D position is inherently ill-posed and polynocular tracking is the only solution. Robustness is tested in the section 5.3 on a sequence with the book where the tracking survives even the frames where the dominant plane is absent. Experiment showing illuminance effects compensation is provided in section 5.4. The possibility of real-time application is considered in section 5.5.

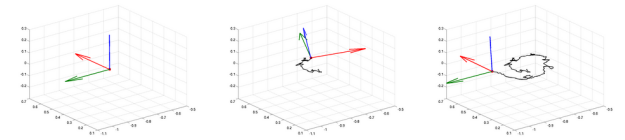
In images, the projections of visible points are denoted by blue and invisible by yellow. Readers are encouraged to zoom-in to the images in the electronic version of the document and play the accompanying video sequences.

5.1 Interleaved Tracking and Constructing Model

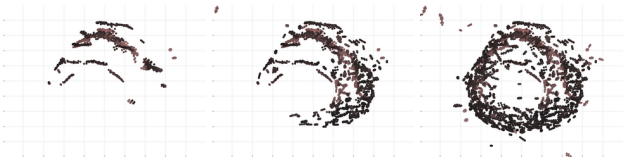
The first experiment demonstrates the interleaving of tracking and incremental construction of the model. The model is constructed during the 360 turn. The process starts with partial reconstruction in the first frame, see the most left column of Figure 2 and movie `exp1_track3D.avi`. The tracker is initiated by this partial model. As the human is turning, the model, is augmented by adding further partial reconstructions. Once the 360 turn is finished the model is complete and later reconstruction are not required.

Please note that the 3D model is actually only a side product of the tracking. Its visual aspect cannot match those models created with specialized stereo-algorithms or visual-hull based algorithms. Despite of its simplicity and certain coarseness allows for very stable pose estimation which is the primary goal of the proposed algorithm.

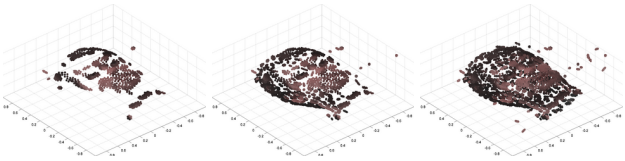
Multiview tracking; blue are visible, yellow invisible (occluded) projections



Corresponding poses and path recorded



Incremental construction of the model, as seen from top

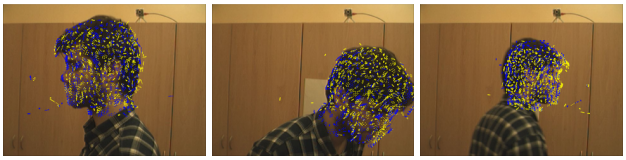


Incremental construction of the model, an overall view

Figure 2: **Incremental model construction from partial 3D reconstructions and registered by 3D tracking**. Rows 1-3: Different views with projected model. Row 4: Total position and orientation in 3D. Rows 5-6: incrementally constructed 3D model. Columns correspond to frames 1, 100 and 310.

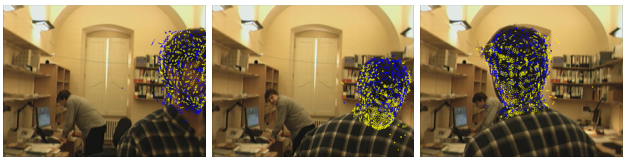
5.2 Monocular Model-Based 3D Tracking

In the case of monocular tracking a 3D model and initial position are considered to be known in advance (e.g. we use the model from previous experiment). The results, where the head was successfully tracked over the 630 frames, although both 3D translation and out-of-plane rotation were involved in the sequence, are shown in Figure 3 and in movie `exp2_comparison.avi`. *ve videu se objevuji dve barvy trasy a neni jasne ktera patri k cemu. Asi by bylo dobre opet pripravit readme soubor, ktery by vysvetlil na co se maji pozorovatele v jednostlivych videich soustredit.* The model position from the view of the camera used for tracking looks correctly, however, since only a single camera was used, the recovered 3D position is inaccurate due to the inherently ill-posedness of the monocular tracking, see row 2 in Figure 3. Naturally, the more cameras are used for the optimization, the more accurate 3D pose is estimated. The results from the same sequence, where object is tracked over all cameras, are provided in the last row of Figure 3.



Tracking camera, in monocular tracking, this is the only one used for optimization. Results of monocular tracking projected

Monocular tracking results as projected to a camera which approximately orthogonal to the tracking one.



Polynocular tracking. The same camera as above. Note the essentially more consistent 3D pose.

Figure 3: **Comparison of monocular (rows 1-2) and polynocular (row 3) tracking**. **Monocular:** Row 1: tracking camera, Row 2: observing camera (shows that, accuracy in orthogonal direction is low). **Polynocular:** Row 3: The same camera with the projected model from multiview tracking.

5.3 Robustness against Critical Poses

A thin object, like the book used in the experiment, may easily appear in poses which are inherently challenging for a tracking algorithm. If only the back is visible, the tracking may get unstable. Even during multiview tracking it may happen that most of the object is visible only in one camera. Still, one good view assures the stability untill the object is again visible in more cameras.

Nasledujici odstavec je jeste nejaky divny, ale momentalne nejak nemam silu ho opravit If the critical pose happens, where the front cover of the book has to be tracked virtually from the single view, the position of the model does not correspond to the projection in the cameras where only small fraction of the book is observable. After the object passes critical position, the model converges to the true position (see Figure 4 and movie `exp3_book.avi`).

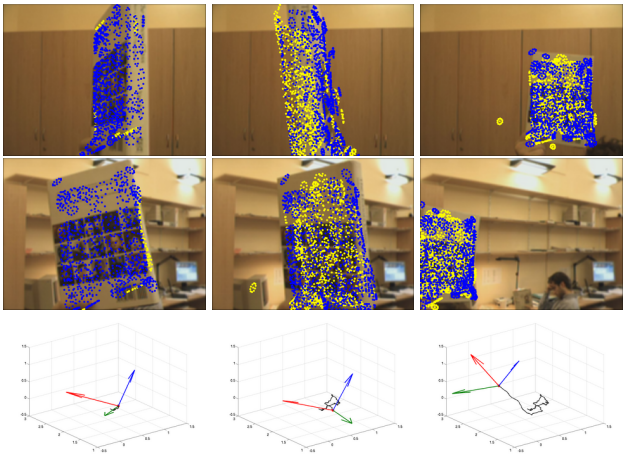


Figure 4: **Book tracking**: Rows 1-2: different cameras with projected model, row 3: shows total position and orientation in 3D space, columns correspond to frames 55, 205 and 265. The second column shows the book in a critical position where dominant plane is visible only in one camera.

5.4 Compensation of Illuminance Effects

The model points are clustered in 14 equally distributed clusters according to their normals. Each cluster is associated with illuminance constant E_i which changes during the tracking to best fit the observed data. Figure 5.4-top shows camera view with projected model, where colors of particular fish-scales correspond to the values of illuminance constants. Higher values corresponds to the recently illuminated points and vice versa. One can see that in this case light sources were located on the left side of the object which corresponds to the reality.

The office has several light sources placed on opposite walls and oriented to the irregulatly arched ceiling. Corresponding changes of the illuminance constant E_6 during 360 turn are shown at Figure 5.4-bottom. Two significant changes during the turn corresponding the light sources are clearly visible. Dynamic changes of the illuminance during the tracking are also visible in movie `exp4_illuminance.avi`). *Je to trochu neprehledne (video i obrazek) i pro mne, ktery vi (doufam) o co jde.*

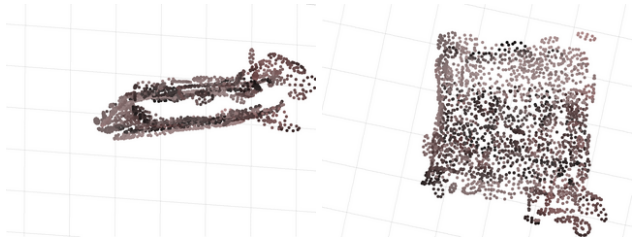


Figure 5: Book Model: Different views of the book model. Small non-planarity in one corner is the reconstructed hand.

Neslo by ukazat traba body jenom z toho clusteru 6, pro ktery se ukazuje jeho prubeh behem otocky?

5.5 Speed Evaluation

Speed was tested on the sequence introduced in the first two experiments (i.e. 4 cameras, RGB images). Slightly-optimized implementation in Matlab runs cca 1.8 s/frame on AMD-64b linux running machine. We show that color do not provide significant information for tracking by succesfull tracking of the same grayscaled sequence. Tracking of grayscale sequence takes approximatelly 800 ms/frame. Typically, multiple cameras are connected to different computers. Hence, all the contributions to the $\mathtt{A}, \mathbf{b}$ from equation 13 can be computed independently in the particular computers. Using such a system, a frame rate of 5 frames per second can be achieved with the current implementation.

6 Conclusions

U zaveru jsem trochu ztratil sily. Jeste to bude potrebovat nejake maso We proposed a fully automatic unified approach of multiview/monocular 3D object tracking interleaved with incremental model construction. Hence, neither model nor initialization are needed to be known in advance. We formulate tracking as a gradient based method minimizing projection dissimilarity. We propose using stereo based method for partial reconstruction and object representation in form of oriented points. This solution means that the method does not need distinguished regions or feature points needed for wide-baseline matching.

We showed that monocular tracking is also possible if the model is available. Even though model projection to the tracking camera looks correctly, the projection to other cameras reveals 3D inaccuracy. On the other hand, if someone is interested in an augmented reality application (e.g. adding a cap to the human), monocular tracking can provide sufficiently pleasant results. Using more cameras, however, significantly increases stability and accuracy in critical poses

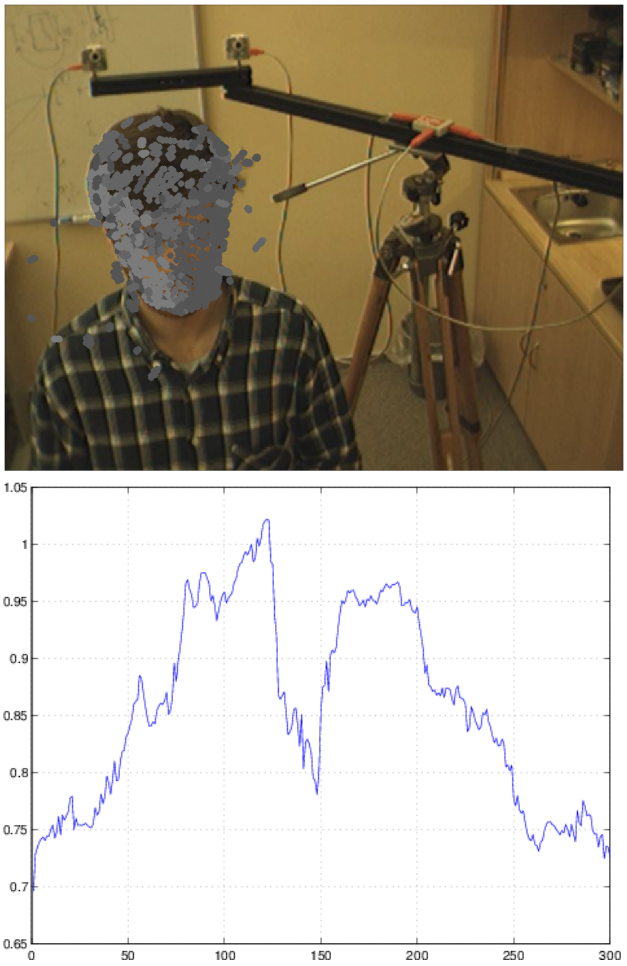


Figure 6: **Top**: The image with projected model. Colors correspond to the computed illuminance $\mathtt{E}_i$ of each particular cluster. **Right**: Values of $\mathtt{E}_6$ during the the 360 turn.

and moves. Exact 3D pose may be necessary in many application ranging from virtual reality, human computer interfaces to visual surveillance.

We experimentally demonstrated that the proposed interleaved approach, starting virtually from nothing, successfully tracks a complete turn-around and wide-ranging motion. An appearance 3D model is delivered as a side product. We demonstrated the robustness of our method on a sequence with a thin object where the dominant plane was often tracked only from one view.

Appendix A

A 3D point $\mathbf{x}$ is projected to 2D image (pixel) coordinates $\mathbf{p}$ as

$$\begin{bmatrix} \lambda\mathbf{p} \\ \lambda \end{bmatrix} = \mathtt{P} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix},$$

where $\mathtt{P}$ is 3×4 camera matrix [13] and $\lambda \in \mathcal{R}$. Let the camera matrix be parametrized as

$$\mathtt{P} = \begin{bmatrix} \mathbf{m}_1^T & t_1 \\ \mathbf{m}_2^T & t_2 \\ \mathbf{m}_3^T & t_3 \end{bmatrix} \tag{17}$$

the function $f : \mathcal{R}^3 \rightarrow \mathcal{R}^2$ projecting 3D point to the camera coordinates is

$$f(\mathbf{x}) = \begin{bmatrix} \frac{\mathbf{m}_1^T\mathbf{x}+t_1}{\mathbf{m}_3^T\mathbf{x}+t_3} \\ \frac{\mathbf{m}_2^T\mathbf{x}+t_2}{\mathbf{m}_3^T\mathbf{x}+t_3} \end{bmatrix}. \tag{18}$$

Differentiating f with respect to $\mathbf{x}$ we obtain $f' : \mathcal{R}^3 \rightarrow \mathcal{R}^{2\times 3}$ Jacobian matrix function, which consists of elements

$$f'_{pq} = \frac{m_{pq}(\mathbf{m}_3^T\mathbf{x} + t_3) - m_{3q}(\mathbf{m}_1^T\mathbf{x} + t_p)}{(\mathbf{m}_3^T\mathbf{x} + t_3)^2} \tag{19}$$

where $m_{pq}, p = 1 \ldots 2, q = 1 \ldots 3$ is q-th elements of $\mathbf{m}_p^T$.

References

[1] Francois Fleuret, Richard Lengagne, and Pascal Fua, "Fixed point probability field for complex occlusion handling," in *IEEE International Conference on Computer Vision*, 2005.

[2] Anurag Mittal and Larry S. Davis, "M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo," in *The seventh European Conference on Computer Vision, ECCV2002*, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. May 2002, number 2350 in LNCS, pp. 18–36, Springer.

[3] J. Carranza, C. Theobalt, M. Magnor, and H.-P. Seidel, "Free-viewpoint video of human actors," *ACM Transaction on Computer Graphics*, vol. 22, no. 3, July 2003.

[4] Takeo Kanade, P.J. Narayanan, and Peter W. Rander, "Virtualized reality: Concepts and early results," in *IEEE Workshop on the Representation of Visual Scenes*, June 1995, pp. 69–76.

[5] Buehler C. Raskar R. McMillan L. Matusik, W. and S. Gortler, "Image–based visual hulls," in *Proceedings of ACM SIGGRAPH 2000*, 2000.

[6] Stephan Würmlin, Edouard Lamboray, and Markus Gross, "3D video fragments: Dynamic point samples for real-time free-viewpoint video," *Computers and Graphics*, vol. 28, no. 1, pp. 3–14, 2004.

[7] Vincent Lepetit and Pascal Fua, "Monocular model-based 3D tracking of rigid objects," *Foundations and Trends in Computer Graphics and Vision*, vol. 1, no. 1, pp. 1–89, 2005.

[8] Luca Vacchetti, Vincent Lepetit, and Pascal Fua, "Stable real-time 3D tracking using online and offline information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1385–1391, 2004.

[9] E. Muñoz, J.M. Buenaposada, and L. Baumela, "Efficient model-based 3D tracking of deformable objects," in *Proceedings of the IEEE International Conference on Computer Vision*, China, October 2005, pp. 877–882.

[10] F. Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce, "3D object modeling and recognition using affine-invariant patches and multi-view spatial constraints," in *International Conference on Computer Vision and Pattern Recognition*, 2003, vol. 2, pp. 272–277.

[11] David Nistér, "Automatic passive recovery of 3d from images and video," in *Second International Symposium on 3D Data Processing, Visualization and TRansmission (3DPVT04)*, 2004, Invited paper.

[12] Dan Cobzas and Martin Jagersand, "3D SSD tracking from uncalibrated video," in *Workshop on Spatial Coherence for Visual Motion Analysis (SCVMA), in conjunction with ECCV 2004*, 2004.

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

[14] Jianbo Shi and Carlo Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593 – 600.

[15] Jana Kostková and Radim Šára, "Stratified dense matching for stereopsis in complex scenes," in *BMVC 2003: Proceedings of the 14th British Machine Vision Conference*, Richard Harvey and J. Andrew Bangham, Eds., London, UK, September 2003, vol. 1, pp. 339–348, British Machine Vision Association.

[16] Radim Šára and Ruzena Bajcsy, "Fish-scales: Representing fuzzy manifolds," in *Proc. 6th International Conference on Computer Vision*, Sharat Chandran and Uday Desai, Eds., New Delhi, India, January 1998, IEEE Computer Society, pp. 811–817, Narosa Publishing House.