# TOWARDS EUCLIDEAN RECONSTRUCTION FROM VIDEO SEQUENCES

Dimitri Bulatov
*Research Institute for Optronics and Pattern Recognition, Germany*

Keywords: Calibration, Dense Reconstruction, Euclidean Reconstruction.

Abstract: This paper presents two algorithms needed to perform a dense 3D-reconstruction from video streams recorded with uncalibrated cameras. Our algorithm for camera self-calibration makes extensive use of the constant focal length. Furthermore, a fast dense reconstruction can be performed by fusion of tessellations obtained from different sub-sequences (LIFT). Moreover, we will present our system for performing the reconstruction in a projective coordinate system. Since critical motions are common in the majority of practical situations, care has been taken to recognize and deal with them.

## 1 INTRODUCTION

Considerable progress was made in the recent years in the areas of **Computer Vision** and **3D-Reconstruction** from video sequences recorded with a single uncalibrated camera. There are two principal approaches for reconstruction: the first uses methods of projective geometry; the task is to determine projective matrices and 3D-points in some projective frame and then to use the additional knowledge (such as known principal point or zero skew of the cameras) to transform the cameras and points into a Euclidean frame. In the second approach, these constraints are imposed at the beginning, in order to avoid any spurious results. If necessary, some additional information is roughly estimated (such as unknown focal length), and by the end of reconstruction, all irregularities are supposed to be corrected by means of bundle adjustment. Examples of successfully dealing with projective geometry (the first strategy) are shown in (Nister2001) and (Pollefeys2002). On the other hand, (Mar2006) shows excellent results of dealing with the second strategy. Nevertheless, many of these algorithms are developed for ”favorable videos” and ”favorable geometry”, such as slow, smooth, almost circular motion around a non-planar object: these algorithms work well after being applied to these favorable scenes, but often turn out to be not successful for almost any **critical motion** such as forward motion, pure translation etc. But in reality, every practical application of ”structure from motion” algorithms – consider the area of robotics, navigation or military applications – constantly requires dealing with critical motions. Our videos, recorded mostly for military applications, are usually taken from mini-planes or mini-drones, carrying some small cameras (see Fig. 1), so in general, the resolution is poor, the effects of the interlacing, lens distortion and blurring are strong, and since the motion of these unmanned vehicles is influenced by wind and other similar effects, the trajectory of the camera is usually not suitable for the reconstruction. Therefore it turned out to be quite important to recognize and to deal with critical motions.

In many cases, we will use methods from projective geometry, see for example (HarZis2000). These methods allow working extensively with linear equations and contribute to numerical stability and robustness of the majority of the problems. In our implementation, the cameras and points in space are obtained in some projective frame from interest points detected in the images. ”Projective” means here that the cameras and points are projectively distorted: for example, the ratios between line segments will not be the same as in the world coordinate frame. Although it is not possible to recover the absolute position, orientation and scaling of the scene just from the video stream, our task will be to determine a 3D-**rectification homography** which transforms the projectively distorted model to a Euclidean (i.e. ratio- and angle-preserving) coordinate frame. Detecting the rectification homography is the key-



Bulatov D. (2008).
TOWARDS EUCLIDEAN RECONSTRUCTION FROM VIDEO SEQUENCES.
In *Proceedings of the Third International Conference on Computer Vision Theory and Applications*, pages 476-483
DOI: 10.5220/0001072304760483

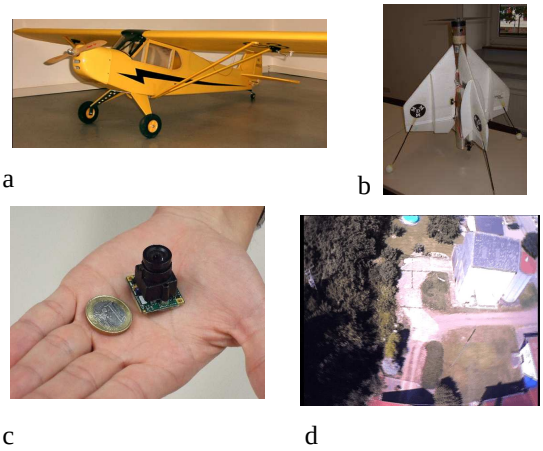


Figure 1: a) mini-plane, b) mini-drone M3D (product of EADS LFK) carrying small cameras c) used for recording cityscapes as shown in d). Note the effects of interlacing, blurring and lens distortion.

point of our method: if it works well, the object will be clearly recognizable and all additional (rather time-consuming) steps, such as **bundle adjustment** in order to refine the results or **tessellation** for better visualization can optionally follow.

**Notation:** we denote 2D-/3D- points in the projective coordinates by column vectors: $\mathbf{x} = (x\,y\,w)^T \in \mathbb{P}^2$, $\mathbf{X} = (X\ Y\ Z\ W)^T \in \mathbb{P}^3$ respectively. Points in Euclidean coordinates will be denoted by $\hat{\mathbf{x}}$ and $\hat{\mathbf{X}}$ respectively.

By $\mathbf{x}_j^i$, respectively $\hat{\mathbf{x}}_j^i$, we will refer to the point number $i$ in view number $j$. Camera matrices (to what we shall simply refer as "cameras") are denoted by $P$ in the projective, and $\hat{P}$ in the Euclidean frame. We denote by $K$ the calibration matrix, by $R$ rotation matrices and by $\mathbf{t}$ camera centers in the world coordinate system. Then, the well-known relation $\hat{P} = KR[I_3 \mid -\mathbf{t}]$ holds. Here $I_k$ is the $k \times k$ identity matrix.

For a matrix $A$, the symbols $(A)_l, (A)^l$ denote the $l$-th row/column of $A$, and $(A)_{\{l\}}, (A)^{\{l\}}$ denote the matrix after its $l$-th row/column has been extracted. As usual, $A^T$ denotes the transpose of $A$. The operator $[\mathbf{x}]_\times$ denotes,as usually, the cross product with the vector $\mathbf{x}$.

By $\Im_r$, we will refer to the image contained in a frame number $r$ of the sequence. All other notation will be introduced later.

**Organization:** in Sect. 2, we will give a brief introduction of our system whose main part takes place in the projective coordinate frame. Section 3 describes our calibration algorithm for Euclidean rectification as well as the method to perform a dense reconstruction. Section 4 shows experimental results of the algorithm for different kinds of video sequences. Conclusions and outlook are given in Sect. 5.

## 2 PROJECTIVE RECONSTRUCTION

Given a video sequence, **interest points** are found in the first frame using Harris Corner Detector, see (Harris1998) for details. Moreover, new features will be found in periodic lags (refreshing). These features are tracked from frame to frame by the Lucas-Kanade algorithm ((KLT1981)). It is quite important to have correspondence points over many images in order to obtain a wide baseline. For the reconstruction, the sequence will be automatically partitioned into subsequences. The first frame of every sub-sequence will be called first **key-frame** of this sub-sequence. We find the second key-frame such as the pair of key-frames has a favorable geometry for reconstruction: we calculate two penalty terms, $GRIC(F)$ and $GRIC(H)$, using formulae (2) and (3) as proposed in (Pollefeys2003) ($GRIC$ is the abbreviation for **Geometric Robust Information Criterion**, introduced by Pollefeys). Since we work with fundamental matrices, the error terms $\varepsilon_F$ for the fundamental matrix respectively $\varepsilon_H$ for the homography are:

$$\varepsilon_F(\mathbf{x}_1,\mathbf{x}_2) = \max\left(\frac{\mathbf{x}_2^T F \mathbf{x}_1}{||\mathbf{x}_1||}\,,\,\frac{\mathbf{x}_2^T F \mathbf{x}_1}{||\mathbf{x}_2||}\right), \text{ and}$$

$$\varepsilon_H(\mathbf{x}_1,\mathbf{x}_2) = ||\hat{\mathbf{x}}_2 - \hat{\mathbf{x}}_2^*||,\ \mathbf{x}_2^* = H\mathbf{x}_1.$$

The reconstruction begins as soon as $GRIC(F) < GRIC(H)$. Note that if either $GRIC(F) \geq GRIC(H)$ for a rather large number of frames or the number of points seen in the first and in the current image of the sub-sequence reduces dramatically, reconstruction has to be performed even though the geometry is apparently not favorable. Once two key-frames are determined, the fundamental matrix $F$ between them is calculated via RANSAC and the relative orientation of two cameras is obtained as pointed out in (HarZis2000): $P_1 = [I_3 \mid \mathbf{0}_3]$, $P_2 = [[\mathbf{e}_2]_\times F \mid \mathbf{e}_2]$ where $\mathbf{e}_2$ is the epipole (projection of the first camera center into the second image).

The task now is to determine the points in space (resulting from the inliers of the fundamental matrix) and the parameters of the intermediate cameras. We triangulate the points seen in both key-frames linearly ((HarZis2000), chapter 12) and obtain the camera matrices by means of **RANSAC with $T_{d,d}$-test** as described in (Mat2004). Generating models (fundamental matrices, camera matrices, homographies) from

parameter sets contaminated with outliers is an indispensable part of our algorithms, so in the majority of cases, robust methods must be applied and every possibility of speeding up the processing must be considered. Therefore, manipulating simple RANSAC by means of $T_{d,d}$ test (with $d$ = 1 or 2) has turned out to be quite useful in our implementation. Also, we must take care of critical motions since the results obtained during this stage of reconstruction of a sub-sequence will be used to obtain camera parameters in the following frames. The following observations have been made:

- If for a large number of frames $GRIG(F) > GRIG(H)$, then either the scene contains some dominant plane(s) or the baseline made up by the cameras between two key-frames is not wide enough. In the first case, the linear solution for camera resection will not work ((HarZis2000), pp. 178–180). In certain cases, one can use homography-based reconstruction methods as the method of camera resection by **plane-by-parallax**, as proposed in (HarZis2000), chapter 18, see also (Mat2005).
- If the epipole lies inside of the image domain, the points close to the epipole should be discarded from triangulation, because their position in at least one direction will be unstable. Another possibility is to take only the points which satisfy some severe cost function such as:

$$\sum_{i=1}^{2}(\hat{\mathbf{x}}_i^* - \hat{\mathbf{x}}_i)^2 < s \cdot \exp\left(-\frac{b}{d_i^2}\right),\ \mathbf{x}_i^* = P_i\mathbf{X}\,,$$

  where $P_1, P_2$ are the camera matrices extracted from the key-frames, $\mathbf{x}, \mathbf{X}$ is a 2D (respectively: corresponding 3D) point, $d_i$ is the distance from $\hat{\mathbf{x}}_i$ to the epipole $\hat{\mathbf{e}}_i$ and $s, b$ are some positive constants.
- The forward and backward motion usually has both of the negative effects described above. Actually, the homography will be the suitable model to describe the position of points in the direction of the epipole and the epipole will be found inside of the image. In this case, we not only discard the points close to the epipole but also reduce the threshold $s$ by the factor 2.

The reconstruction of a sub-sequence continues by extrapolation of the previous results to the frames after the second key frame. We obtain new camera matrices by resection with the already known 3D-points (via RANSAC followed by a non-linear error minimization) and we obtain new 3D-points by triangulation from the known cameras (usually 3–5). The frame, where the number of either triangulation- or resection-inliers is small, marks the end of the subsequence. If the number of the unfeasible frame is $n$, then the frame number $n-1$ is the last frame of the first sub-sequence and the first key-frame of the next sub-sequence is $n-2$. This is because we cannot trust the camera number $n$ of the first sub-sequence, and, as we will see below, we need at least a double camera overlap. Of course, the second reconstruction will be obtained in a different coordinate system, therefore both reconstructions are "fused" by means of the common cameras $P_{n-2}^{old}, P_{n-1}^{old}, P_1^{new}, P_2^{new}$ and points $\mathbf{X}^{new}, \mathbf{X}^{old}$ seen both in old and new views. The task is to find a 3D-homography $H$ which satisfies $P^{old} = P^{new}H$ and $\mathbf{X}^{old} = H^{-1}\mathbf{X}^{new}$ (such a homography exists by Theorem 9.10 in (HarZis2000)). The method we propose works as follows:

First of all, the linear solution is calculated: if we consider camera matrices $P^{old}, P^{new}H$ as row vectors with 12 elements, the vector representing the algebraic error from a single camera pair is $(P^{old})_k(P^{new}H)_1 - (P^{old})_1(P^{new}H)_k$ for $k = 2, \ldots, 12$. Clearly, each pair of projection matrices contributes 11 equations, therefore a double camera overlap is enough to determine 16 entries of the homogeneous quantity $H$. In order to refine the initial value for $H$, the squared geometric error

$$\varepsilon = \sum_{j=1}^{overlap}\left(P_j^{new}H\mathbf{X}^{old} - \hat{\mathbf{x}}_{n-j}\right)^2 \qquad (1)$$

is calculated for each 3D-points $\mathbf{X}^{old}$ obtained in the first reconstruction and visible in the relevant views. Similar error is obtained for 3D-points in the new coordinate frame. Now, if the error obtained by reprojecting an old 3D-point with the new cameras (as in (1) ) or vice versa is low, this point is considered to be an inlier. In the case where there are only a few inliers, the initial estimate of $H$ is poor. In this situation (which, for example, can happen if the centers of both cameras coincide), we consider just a single camera overlap $P_1^{new}, P_{n-2}^{old}$ and the correspondences of reprojected points $\mathbf{X}, \mathbf{x}$, as pointed out in (Nister2001), pp. 64–65. Four such correspondences are enough to generate a RANSAC-hypothesis from which $H$ can be computed. At each case, after an initial estimate of $H$ has been obtained, the iterative minimization of the error given by (1) is performed over all inliers. Given $H$, the new cameras and points can be mapped into the old coordinate frame.

# 3 AUTO-CALIBRATION AND EUCLIDEAN RECONSTRUCTION

## 3.1 Auto-Calibration

The starting point of any rectification algorithm is a projective reconstruction given by a set of $n$ cameras $P_i$ and points in space, $\mathbf{X}^j$. The task is to find a so called rectifying spatial homography $H$ such as the transformed cameras $\hat{P}_i = P_i H$ and points $\hat{\mathbf{X}}_j = H^{-1}\mathbf{X}_j$ represent a ratio- and angle-preserving reconstruction of the scene. If the first camera is given in the form $P_1 = [I_3 \mid \mathbf{0}_3]$, then, according to (HarZis2000), pg. 460, $H$ can be chosen as follows:

$$H = \begin{bmatrix} K & \mathbf{0}_3 \\ -\hat{\mathbf{p}}_\infty^T K & 1 \end{bmatrix}, \qquad (2)$$

where $K$ is the constant but unknown **calibration matrix** and $\hat{\mathbf{p}}_\infty$ is the **plane at infinity**. We store the unknown entries of $K$ in the column vector $\mathbf{k} = \mathbf{k}(K) = [f\ a\ s\ u\ v]^T$, they correspond respectively to the focal length, aspect ratio, skew and two coordinates of the principal point. There are 8 degrees of freedom (5 for $\mathbf{k}$ and 3 for $\hat{\mathbf{p}}_\infty$), so the minimization of some geometrically meaningful cost function is to be performed over the 8-tuples $[\mathbf{k}^T\ \ \hat{\mathbf{p}}_\infty]$. Before this can be done, initial values of the parameters must be obtained. At the beginning of the optimization, we set $a = s = u = v = 0$. For the focal length $f$, the formula obtained in (Bougnoux1998),

$$f^2 = -\frac{\mathbf{b}'^T[e']_\times \tilde{I}_2 F \mathbf{b}\mathbf{b}^T F^T \mathbf{b}'}{\mathbf{b}'^T[e']_\times \tilde{I}_2 F \tilde{I}_2 F^T \mathbf{b}'},$$

with $\tilde{I}_2 = \mathrm{diag}(1\ 1\ 0), \mathbf{b}, \mathbf{b}'$ principal points of some pair of cameras, $F$ the fundamental matrix resulting from these cameras and $e'$ the epipole, can be taken into consideration. Also, the image diagonal is an acceptable initial estimate of $f$. The parameters of $\mathbf{p}_\infty$ (the homogeneous representation of $\hat{\mathbf{p}}_\infty$) can be estimated with cheirality inequalities as Nistér pointed out in (Nister2000s). The main theorem proved in his paper says that if there is some plane $\mathbf{p_0}$ which for all $i = 2, ..., n$ satisfies the relation:

$$\begin{aligned} &\mathrm{sgn}\left[(\mathbf{p}_0 \cdot \mathbf{C}(P_{i-1}))(\mathbf{p}_0 \cdot \mathbf{C}(P_i))\right] = \\ &\mathrm{sgn}\left[(\mathbf{p}_\infty \cdot \mathbf{C}(P_{i-1}))(\mathbf{p}_\infty \cdot \mathbf{C}(P_i))\right], \end{aligned} \qquad (3)$$

then there is a continuous path from $\mathbf{p}_0$ to $\mathbf{p}_\infty$ such that no camera center is met on this path. Here we denote by $\mathbf{C}(P) = [c_1\ c_2\ c_3\ c_4]^T$ the camera center, normalized as follows: $c_l = (-1)^l \det(P^{\{l\}}), l \in \{1, ..., 4\}$.

If all 3D-points have the last homogeneous coordinate 1, then $\mathrm{sgn}(depth(\mathbf{X}, P)) = \mathrm{sgn}(w \cdot c_4)$ where $w = (P\mathbf{X})_3$ is the third element of $P\mathbf{X}$. For all points $\mathbf{X}_j$ visible by the pair of cameras $P_{i-1}, P_i$, we calculate $\xi_j = \mathrm{sgn}\left[(P_i\mathbf{X}_j)_3 (P_{i-1}\mathbf{X}_j)_3\right]$. Then, by multiplying $P_i, i \in \{2, ..., n\}$ by $\mathrm{sgn}\left(0.5 + \sum_j \xi_j\right)$, we ensure that the majority of $\mathbf{X}_j$ are either in front or behind both of the cameras (with respect to $\mathbf{p}_\infty$, which in (Nister2000s) is denoted by "untwisted pair"). With this normalization, all $\mathbf{p}_\infty \cdot \mathbf{C}(P_i)$ must have the same sign, so recalling (3) and setting $\mathbf{p}_0 \cdot \mathbf{C}(P_1) > 0$, the task is to find $\mathbf{p}_0$ which satisfies $\mathbf{p}_0 \cdot \mathbf{C}(P_i) > 0$ for all $i$. The problem formulated as:

- find a maximal scalar $\delta$ subject to:

$$\left[\mathbf{C}_i \quad -|\mathbf{C}_i|\right] \begin{bmatrix} \mathbf{p}_0 \\ \delta \end{bmatrix} > 0 \text{ and } \left|\mathbf{p}_0^l\right| \leq 1, l \in \{1, ..., 4\}$$

can be solved, for example, by the **Simplex Algorithm**. Note that the last condition allows obtaining a unique solution for the homogeneous quantity $\mathbf{p}_0$. This $\mathbf{p}_0$ is an acceptable initial estimate for $\mathbf{p}_\infty$ because in the optimization round, we can move along a continuous path not crossing the camera centers. We refine the initial estimate using the knowledge about (nearly) square pixels and the principal point. Since $PH = \hat{P} = KR[I_3 | -\mathbf{t}]$, we have $(PH)^{\{4\}} = KR$. For a matrix $A$, we define the operator $\mathcal{R}(A) = K/K_{3,3}$ where $K$ is the upper triangular matrix resulting from the RQ-decomposition of $A$, in other words

$$K = \left(chol\left[(AA^T)^{-1}\right]\right)^{-1}$$

for a non-singular matrix $A$. Then we know that the matrix $AK^{-1}$ is a rotation matrix and our cost function results in comparing $\mathcal{R}(PH)^{\{4\}}$ with the "ideal" calibration matrix $\mathrm{diag}[f\ f\ 1]$ which corresponds to the vector $\mathbf{k_0} = [f\ 0\ 0\ 0\ 0]$:

$$\sum_{\substack{1 \leq j \leq 5 \\ 1 \leq i \leq n}} \left(\frac{\mathbf{k}(K)_j - \mathbf{k}_j^0}{(\Gamma_{ij}\mathbf{k}_1)}\right)^2, \quad K = \mathcal{R}\left(P_i H(\mathbf{k}, \hat{\mathbf{p}}_\infty)^{\{4\}}\right) \qquad (4)$$

Here $H(\mathbf{k}, \hat{\mathbf{p}}_\infty)$ is the term for $H$ as in (2), $\mathbf{k}_1$ is the new focal obtained as the result of an iteration and $\Gamma_{ij}$ are the weights representing the reliability of the constraints. For example, we can choose $\Gamma_{ij} = \gamma_i \gamma_j$, where $\gamma_i$ is the average reprojection error of all points observed in the camera number $i$ and $\gamma_j$ say how reliable the knowledge about camera parameters is (we take $\gamma_2 = \gamma_3 = \gamma_4 = \gamma_5 = 1, \gamma_1 = 1000$ which means that the focal length is unknown but constant). After several iterations, the improved estimates of skew, aspect ratio, principal point and focal length are obtained, we update $\mathbf{k}^0$ by $\mathbf{k}(K)$ in (4) and we set $\gamma_1 = 1$. We optimize (4) by means of the **Levenberg-Marquardt iterative algorithm**.

**Remark 1.** The optimization stage of auto-calibration is fast, because all derivatives needed for the Jacobian can be written analytically since the terms

for the inverse of a $3 \times 3$ matrix and its **Cholesky-Decomposition** can be performed manually. Moreover, this method usually converges after only 6–8 iterations. Other advantages of this algorithm compared to other algorithms are: the fact of constant focal length is used extensively, the quality of every single camera is taken into account and the initial value of the plane at infinity is determined in a robust way (such as not *all* scene points have to lie in front of *all* cameras, as for example in the case of forward motion).

## 3.2 Dense Reconstruction

In this subsection, we describe our method used to generate textured maps. The points visible in the first key-frame $\Im$ of a sub-sequence are partitioned into triangles (for example, by means of the **Delaunay Triangulation**). If we assume that a triangle $\triangle^j$ in the image plane corresponds to a feasible (covered with the object texture) triangle in space, we can calculate the support plane for $\triangle^j$ which we call $\mathfrak{E}^j$. If $\mathbf{x} \in \triangle^j$, then the corresponding 3D-point $\mathbf{X}$ can be calculated in the projective frame either from the relation:

$$\begin{cases} P\mathbf{X} = \mathbf{x} \\ \mathfrak{E}^j\mathbf{X} = 0\,, \end{cases} \tag{5}$$

or, to speed up the processing, by means of 2D-homographies. Using operators $(\cdot)_l, (\cdot)_{\{l\}}, (\cdot)^l, (\cdot)^{\{l\}}$ defined above, we have:

**Result.** Any of three homographies

$$H_l = \left( (P)^{\{l\}} - (P)^l \cdot (\mathfrak{E})^{\{l\}} / (\mathfrak{E})^l \right)^{-1},$$

such as $\mathfrak{E}^l \neq 0, l \in \{1,2,3\}$, maps the triangle in image $\Im$ into the corresponding triangle in space. The point $\mathbf{X}$ corresponding to $\mathbf{x}$ is obtained as follows:

$$(\mathbf{X})_{\{l\}} = H_l\mathbf{x}, (\mathbf{X})_l = -(\mathfrak{E})^{\{l\}} \cdot (\mathbf{X})_{\{l\}} / (\mathfrak{E})^l.$$

To prove the formula above, we consider (5), and we extract $(\mathbf{X})_l$ from its second equation. Now we insert $(\mathbf{X})_l = -(\mathfrak{E})^{\{l\}} \cdot (\mathbf{X})_{\{l\}} / (\mathfrak{E})^l$ into the first equation and obtain $\mathbf{x} = H_l^{-1}(\mathbf{X})_{\{l\}}$. We only allow $l \in \{1,2,3\}$, because we suppose that the Euclidean reconstruction is given on this stage, so $\mathbf{X}$ has the last coordinate 1.

For better numerical conditioning, we choose $l = \arg\max(|(\mathfrak{E}^k)|), k \in \{1,2,3\}$. Now we can store the numbers $l^j$, planes $\mathfrak{E}^j$ and the corresponding homographies $H_l^j$ for every triangle $\triangle^j$. Also, we stabilize the calculations by selecting dominant planes (via RANSAC), correcting the positions of 3D-points and preferring the triangles lying completely in these planes. Now, obtaining an initial hypothesis of every pixel $\hat{\mathbf{x}}$ inside of the convex hull of all detected points can be performed rather quickly, as pointed out in the scheme below:

$$\hat{\mathbf{x}} \rightarrow \triangle^j \rightarrow l, H_l^j, \mathfrak{E}^j \rightarrow \hat{\mathbf{X}} \tag{6}$$

Then, the unfeasible triangles can be detected by the back-projection of the hypothesized points $\mathbf{X}$ into the images close to $\Im$. If the scene is not too homogeneous, then the intensity differences between the outliers must be large. Let $n$ the number of images to compare ($n$ = 3–5 in our experiments), $\Im_1$ our reference image and $\Im_2, ..., \Im_n$ images used to determine the feasibility of $\triangle^j \subseteq \Im_1$. Let $A^j$ be the total number of local overlaps (how many times a point from $\triangle^j$ was projected inside the images $\Im_2, ..., \Im_n$). The cost function we use to determine the feasibility of $\triangle^j$ is:

$$\varepsilon(j) = (2 - \xi^j)\log(A^j)^{-2} \sum_{\substack{\hat{\mathbf{x}} \in \triangle^j \\ i=2,...,n}} \delta^j(\hat{\mathbf{x}}, i)^2\,, \tag{7}$$

where $\delta^j(\hat{\mathbf{x}}, i) = \Im_1(U(\hat{\mathbf{x}})) - \Im_i(U(\hat{P}_i\mathbf{X}))$ is the intensity difference inside of a small window $U$ around a relevant pixel and $\xi^j$ is zero if $\triangle^j$ does not lie inside one of the dominant planes and 1 if it does. All triangles, for which the cost function does not exceed a given threshold, are declared as feasible. Contrary to (MorKan2000) who proposes optimizing the results of the triangulation over all possible triangulations, we prefer use the 3D-points generated from other sub-sequences in order to fill the holes caused by unfeasible triangles. This seems to be a logical approach because partitioning the video sequences into sub-sequences (and stitching these sub-sequences as described in Sect. 2) is a consequence of the fact that the object is seen from different positions. In order to provide the texture of every of these views, a reference image from a sub-sequence must be taken. We call this method **"Local Incremental Fusion of Tessellations"**, **LIFT**. Suppose we are given $m$ sub-sequences (i.e. reference images $\Im_{r_1}, ..., \Im_{r_m}$ for which we have triangulations, support planes and homographies. The task is to compute the feasibilities for the triangles of the last sub-sequence. The computation algorithm works as follows ($s_1, s_2, s_3$ are constant thresholds):

for every pixel $\hat{\mathbf{x}} = \hat{\mathbf{x}}_{r_m}$ in $\Im_{r_m}$
  determine $j$ such as $\hat{\mathbf{x}} \in \triangle^j$
  extract $\mathfrak{E}^j$ and $H^j$, then calculate $\hat{\mathbf{X}}$ using (6)
  increase area $A^j$ and set status = 0;
  for $i = 1, ..., m-1$
    reproject $\hat{\mathbf{X}}$ with camera $\hat{P}_{r_i}$ to obtain $\hat{\mathbf{x}}_{r_i}$
    if $\hat{\mathbf{x}}_{r_i}$ lies inside of a feasible triangle in $\Im_{r_i}$
      compare the support plane $\mathfrak{E}$ of this triangle with $\mathfrak{E}^j$
      if $||\mathfrak{E} - \mathfrak{E}^j|| < s_1$
        *(it is approximately the same point)*
        increase overlap$^j$, set status = 1 and break
  if status == 0

*(an occluded point or is not inside of all previous images)*
reproject $\hat{\mathbf{X}}$ into the neigboring images $\Im_{r_m+1},\ldots,\Im_{r_m+n}$
calculate intensity differences proceeding from $\hat{\mathbf{x}}$ with (7)
add the squared sum of these errors to $\delta^j$.
for every $j$
if overlap$^j/A^j > s_2$ or $\varepsilon^j > s_3$ (as in (7))
the triangle $j$ is declared unfeasible

Finally, feasible triangles from all sub-sequences will be given their texture, as shown in the images below.

# 4 RESULTS

We will present results from three movies taken with three different cameras. The first movie (**"House"**, 400 frames, 105 camera positions – because every 4-th frame was taken) was recorded with a handheld camera around a toy-house, so its resolution as well as the trajectory of the camera is good. The only difficulties the system has to deal with are the large number of outliers and the configuration of inliers: in many frames they are nearly coplanar which makes the camera resection quite difficult. The result of the calibration algorithm is illustrated in two Fig. 2, with the texturation obtained with our method of local incremental fusion LIFT.

The second sequence (**"Infrared"**, some 150 frames) was recorded by an infrared camera and shows a sky-scraper in Frankfurt-upon-Oder. As in most infrared sequences, the percentage of tracking outliers is large, due to dead pixels. Moreover, almost all of the 3D-points are situated either far away from the object or in some dominant planes, which makes the usual determination of $\mathbf{p}_\infty$ quite hard. Nevertheless, the result of our calibration algorithm was refined by bundle adjustment, and the results of our method are shown in Fig. 3.

The sequence (**"Cityscape"**, 20 frames) is obtained from our mini-plane and shows a typical view of a cityscape as in Fig. 1. Also here, the results of reconstruction are good (Fig. 4) compared with the quality of the input video.

In all sequences, the calibration matrix was very close to what we have estimated by using a calibration plane, therefore we can assume that the small deviations were caused by lens distortion effects. The small effects of projective distortion in the sequence **"Infrared"** were eliminated by means of bundle adjustment.

# 5 CONCLUSIONS AND FUTURE WORK

**Conclusions.** We have presented a system which is able to perform the Euclidean reconstruction from video sequences recorded with a single camera. The system can recognize some important critical motions (such as forward and backward motion) and deal with them, such that even in the case of not favorable geometry, the results of reconstruction are acceptable. Another advantage is that the system is robust: for example, outliers caused by small moving objects in the images will be detected by robust algorithms and excluded from consideration.

The structure of the system allows detecting and tracking points, performing and stitching projective reconstructions from frame to frame. In other words, there is no need of exhaustive matching of pairs or triples of frames (as in (Mar2006) or (Nister2000)) to find a pair or a triple with a favorable geometry. The reconstruction can be stopped anytime, if necessary, given that the reconstruction between the first pair of key-frames was performed. Then, the calibration process is quite fast and as result, a sparse cloud of 3D-points and the camera trajectory will be obtained. The computation times of the first draft of our algorithm lie between 10 and 15 frames/sec., therefore the hope to achieve a real-time reconstruction exists. Extracting and fusing dense models obtained from several sub-sequences as described above is also a fast process (because the optimization is performed over triangles rather than over points), but before this can be done, the error minimization over all points and all cameras must be performed to optimize the results of the sparse reconstruction which is a rather time-consuming process.

**Future Work.** Our next step towards the dense reconstruction will be the search of a global algorithm which considers the triangulation from the reference frames of all sub-sequences at the same time and deals with occlusions. The task is to refine the initial result obtained by LIFT. Thus, the local cost function given by (7) has to be modified. Still, our biggest problem remains the quality of our videos. We deinterlace the images, if necessary, but the blurring effects are in many cases very strong. Lens distortion is also a serious problem: without distortion correction, the assumption of linear transformations between images does not hold, so the complete reconstruction algorithm is likely to collapse. At the moment, we estimate the distortion coefficients before the flight and undistort the images, but future work includes automatic recognition and correction of lens distortion.

Figure 2: Results of reconstruction of sequence **”House”**: three views from the original sequence, result of sparse reconstruction given in points and straight lines and camera trajectory. Below are two snap-shots from the textured model. Note the small number of undetected unfeasible triangles.

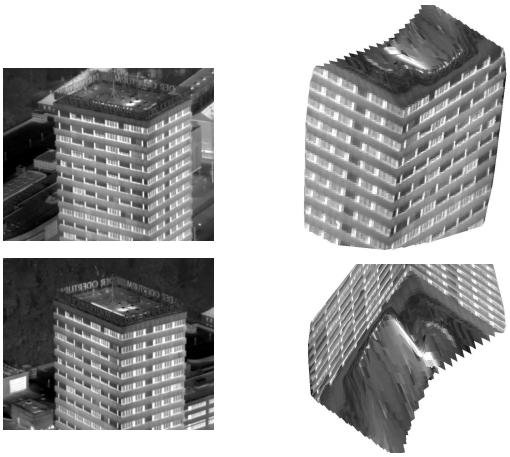

Figure 3: Results of reconstruction of sequence **"Infrared"**.

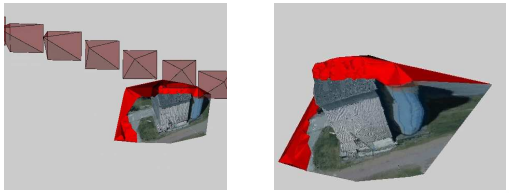

Figure 4: Results of reconstruction of sequence **"Cityscape"**. We show cameras trajectory and a dense point cloud inside of the convex hull of Harris-Points detected only in the first view. Points outside the convex hull are marked by red.

## REFERENCES


Bougnoux S., From Projective to Euclidean space under any practical situation, a criticism of self-calibration. In Proceedings of the International Conference on Computer Vision (ICCV), Bombay, India, pp. 790-796, January 1998

Harris C. G., Stevens M. J., A Combined Corner and Edge Detector. In Proceedings of 4th Alvey Vision Conference, pp. 147-151, 1998

Hartley R., Zisserman A., Multiple View Geometry in Computer Vision. Cambridge University Press, 2000

Lucas B., Kanade T., An Iterative Image Registration Technique with an Application to Stereo Vision. In Proceedings of 7th International Joint Conference on Artificial Intelligence (IJCAI), pp. 674-679, 1981

Martinec D., Pajdla T., 3D reconstruction by gluing pairwise Euclidean reconstructions, or 'how to achieve a good reconstruction from bad images'. In Proceedings of the 3D Data Processing, Visualization and Transmission conference (3DPVT) , University of North Carolina, Chapel Hill, USA, June 2006.

Matas J., Chum O., Randomized Ransac with $T_{d,d}$-test. Image and Vision Computing, 22(10) pp. 837–842, September 2004.

Matas J., Chum O., Werner T., Two-view geometry estimation unaffected by a dominant plane. In Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 772–780, Los Alamitos, California, USA, June 2005

Morris D. Kanade T., Image-Consistent Surface Triangulation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR–00), volume 1, pages 332–338, Los Alamitos, 2000. IEEE

Nistér D., Automatic dense reconstruction from uncalibrated video sequences. PhD Thesis, Royal Institute of Technology KTH, Stockholm, Sweden, March 2001

Nistér D., Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In Proceedings of the European Conference on Computer Vision, ECCV, Vol. 1, pp. 649–663, 2000

Nistér D., Untwisting a projective reconstruction. International Journal of Computer Vision, 60(2) pp. 165–183

Pollefeys M., Obtaining 3D Models with a Hand-Held Camera/3D Modeling from Images. Tutorial notes, presented at Siggraph 2002/2001/2000, 3DIM 2001/2003, ECCV 2000, http://www.cs.unc.edu/ marc/tutorial/

Pollefeys M., Verbiest F., Van Gool L., Surviving dominant planes in uncalibrated structure and motion recovery. Computer Vision – ECCV 2002, 7th European Conference on Computer Vision, Lecture Notes in Computer Science, Vol. 2351, pp. 837–851, 2003