TOWARD LARGE SCALE MODEL CONSTRUCTION FOR VISION-BASED GLOBAL LOCALISATION

Pierre Lothe, Steve Bourgeois, Fabien Dekeyser
CEA LIST, Point Courrier 94, Gif-sur-Yvette, F-91191 France
pierre.lothe@cea.fr, steve.bourgeois@cea.fr, fabien.dekeyser@cea.fr

Eric Royer, Michel Dhome
LASMEA UMR 6602, Université Blaise Pascal/CNRS, Clermont-Ferrand, France
eric.royer@lasmea.univ-bpclermont.fr, michel.dhome@lasmea.univ-bpclermont.fr

Keywords: SLAM, 3D reconstruction, non-rigid ICP, global localisation, trajectometry.

Abstract: Monocular SLAM reconstruction algorithm advancements enable their integration in various applications: trajectometry, 3D model reconstruction, etc. However proposed methods still have drift limitations when applied to large-scale sequences. In this paper, we propose a post-processing algorithm which exploits a CAD model to correct SLAM reconstructions. The presented method is based on a specific deformable transformations model and then on an adapted non-rigid ICP between the reconstructed 3D point cloud and the known CAD model. Experimental results on both synthetic and real sequences point out that the 3D scene geometry regains its consistency and that the camera trajectory is improved: mean distance between the reconstructed cameras and the ground truth is less than 1 meter on several hundreds of meters.

1 INTRODUCTION

Simultaneous Localization and Mapping (SLAM) and Structure From Motion (SFM) methods enable to reconstruct both the trajectory of a moving camera and the environment. However, the first proposed algorithms present a significant computation burden which avoids large-scale sequence reconstruction. Recent works tend to overcome this limit by reducing the problem complexity while still only using easily embeddable and low-cost materials.

For example, Nister (Nister et al., 2004) does not use any global optimisation. It enables him to speed up the computation time but it is very sensitive to error accumulations since the 3D scene geometry is never questioned. Davison (Davison et al., 2007) proposes a Kalman-filter based solution. This method reaches real-time if the number of landmarks is quite small. Another approach is to use a full non-linear optimisation of the scene geometry: Royer (Royer et al., 2005) uses a hierarchical bundle adjustment in order to build large-scale scenes. Afterwards, Mouragnon (Mouragnon et al., 2006) proposes an incremental non-linear minimisation method in order to almost completely avoid computer memory problem by only optimizing the position of the geometry scene on the few last cameras.

Nevertheless, monocular SLAM and SFM methods still present limitations: the trajectory and 3D point cloud are known up to a similarity. Thus, all the displacements and 3D positions are relative and it is not possible to obtain an absolute localisation of each reconstructed element. Besides, as well as being prone to numerical errors accumulation (Davison et al., 2007; Mouragnon et al., 2006), monocular SLAM algorithms may present scale factor drift: their reconstructions are done up to a scale factor, theoretically constant on the whole sequence, but it appears that it changes in practice. Even if these errors could be tolerated for guidance applications which only use local relative displacements, it becomes a burning issue for other applications using SLAM reconstruction results like trajectometry or global localization.

To improve SLAM results, a possible approach consists in injecting a priori knowledge about the trajectory (Levin and Szeliski, 2004) or about the environment geometry (Sourimant et al., 2007). This last kind of information can be provided by CAD models. Furthermore, even if high precision models are still unusual, coarse 3D city models are now wide spread in GIS databases.

More precisely, we introduce in this paper a solution which consists in estimating a non-rigid deformation of the SLAM reconstruction with respect to a coarse CAD model of the environment (i.e. only composed of vertical planes representing buildings fronts). First, a model of SLAM reconstruction error is used to establish a specific non-rigid transformations model of the reconstructed environment with few degrees of freedom (section 2). Then, we estimate the parameters of these transformations which allow to fit the SLAM reconstruction on the CAD model through a non-rigid ICP optimisation (section 3). The complete refinement process is then evaluated on both synthetic and real sequences (section 4).

2 TRANSFORMATIONS MODEL

The registration of 3D point cloud and CAD model has been widely studied. Main used methods are Iterative Closest Point ICP (Rusinkiewicz and Levoy, 2001; Zhao et al., 2005) or Levenberg-Marquardt (Fitzgibbon, 2001). Nevertheless, these methods only work originally with Euclidean transformations or similarities, what is not adapted to our problem. To overcome with this limit, non-rigid fitting methods have been proposed. Because of their high degree of freedom, this category of algorithm needs to be constrained. For example (Castellani et al., 2007) use regularization terms to make his transformation be in accordance with his problem scope. Therefore we will now introduce the specific non-rigid transformations we use for our problem.

2.1 Non-Rigid Transformations Model

The constraints we used to limit the applicable transformations on the 3D SLAM reconstruction are based on experimental results. We observe that the scale factor is nearly constant on straight lines trajectory and change during turnings (see figure 5(b)). So we have decided to consider trajectory straight lines as rigid body while articulations are put in place in each turning. Thus the selected transformations are piecewise similarities with joint extremities constraints.

Thus, the first step is to automatically segment our reconstructed trajectory. We use the idea suggested by Lowe in (Lowe, 1987). He proposes to cut recursively a set of points into segments with respect both to their length and their deviation. So we split the reconstructed trajectory (represented as a set of temporally ordered key cameras) into m different segments $(\mathcal{T}_i)_{1\leq i\leq m}$ whose extremities are cameras denoted $(\mathbf{e}_i, \mathbf{e}_{i+1})$.

2.2 Transformations Parametrisation

Once the trajectory is split into segments, each 3D reconstructed point must be associated to one cameras segment. We say that a segment “sees” a 3D point if at least one camera of the segment observes this point. There are two cases for point-segment association. The simplest is when only one segment sees this point: the point is then linked to this segment. The second appears when a point is seen by two or more different segments. In this case, we have tested different policies which give the same results and we arbitrarily decided to associate the point to the last segment which sees it.

We call $\mathcal{B}_i$ a fragment composed both of the cameras of $\mathcal{T}_i$ (i.e. those included between $\mathbf{e}_i$ and $\mathbf{e}_{i+1}$) and of the associated reconstructed 3D points. Obviously, for $2 \leq i \leq m-1$, the fragment $\mathcal{B}_i$ shares its extremities with its neighbours $\mathcal{B}_{i-1}$ and $\mathcal{B}_{i+1}$.

We saw in section 2.1 that applied transformations are piecewise similarities with joint extremities. Practically, those transformations are parametrized with the 3D translation of the extremities $(\mathbf{e}_i)_{1\leq i\leq m+1}$. From these translations, we will deduce the similarities to apply to each fragment (i.e. its cameras and 3D points). Precisely, since camera is embedded on a land vehicle, the roll angle is not called into question. Then, each extremity $\mathbf{e}_i$ has 3 degrees of freedom and so each fragment has 6 degrees of freedom as expected.

Figure 1 presents an example of extremity displacement.

3 NON-RIGID ICP

Once we have defined the used non-rigid transformations (section 2.1) and their parametrisation (section 2.2), we can use additional information provided by CAD model to improve the SLAM reconstruction. Thus, we propose in this part a non-rigid ICP between the reconstructed 3D point cloud and CAD model: first we present the cost function we minimize, the different optimization steps and then parameters initialisation.

Cost Function. Once all the pre-treatment steps are done, we dispose of both the fragmented reconstruction and a simple coarse 3D model of the scene.

In our work, we want to fit the 3D point cloud onto the CAD model. The base cost function ε is then the normal distance d between a 3D point and the CAD model $\mathcal{M}$, i.e. the distance between a 3D point Q_i and

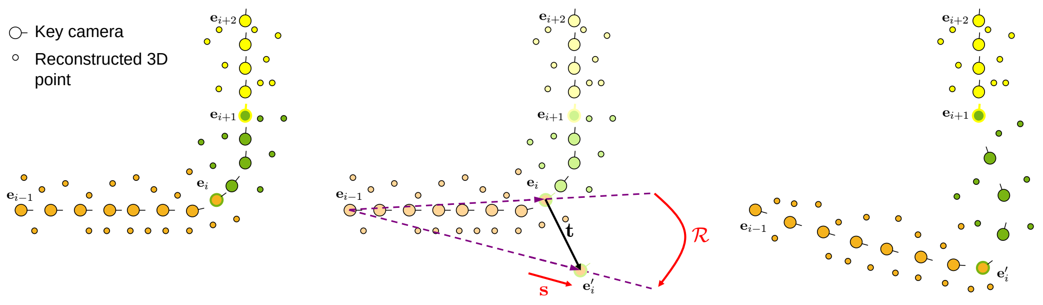


Figure 1: **An example of fragment-based reconstruction transformation.** (a) The segmentation of the original reconstruction. (b) A 3D translation of the extremity $\mathbf{e}_i$. The similarity $\mathcal{S}(\mathbf{e}_{i-1}, s, \mathcal{R})$ is then deduced from this 3D displacement and applied to the whole fragment $\mathcal{B}_{i-1}$. An equivalent treatment is realized on $\mathcal{B}_i$. (c) The result of the transformation: the two fragments linked to the moved extremity have been modified.

the plane $\mathcal{P}_{g_i}$ it belongs to:

$$\begin{aligned} \varepsilon(Q_i, \mathcal{M}) &= d(Q_i, \mathcal{M}) \\ &= d(Q_i, \mathcal{P}_{g_i}) \end{aligned} \quad (1)$$

Non-linear Minimization. The aim of this step is to transform the different fragments to minimise the distance between the reconstructed point cloud and the CAD model, i.e. solving the problem:

$$\min_{\mathbf{e}_1,\ldots,\mathbf{e}_{m+1}} \sum_i d(Q_i, \mathcal{M})^2 \quad (2)$$

Point-Plane Association. The problem in point-plane association is that we do not know in reality on which CAD model plane $\mathcal{P}_{g_i}$ belongs each 3D reconstructed point. So we make the hypothesis that it is the nearest one. Equation 1 becomes:

$$\varepsilon(Q_i, \mathcal{M}) = \min_{\mathcal{P}_j \in \mathcal{M}} d(Q_i, \mathcal{P}_j) \quad (3)$$

Furthermore, to reduce algorithm complexity, we consider that for a point Q_i, the nearest plane $\mathcal{P}_{h_i}$ does not change during the minimization. Thus, the selection between 3D point and corresponding plane can be done outside the minimization:

$$\forall Q_i, \mathcal{P}_{h_i} = \underset{\mathcal{P} \in \mathcal{M}}{argmin}\ d(Q_i, \mathcal{P}) \quad (4)$$

Besides, the distance d takes into account that the planes are finite: to be associated to a plane $\mathcal{P}$, a 3D point Q must have its normal projection inside $\mathcal{P}$ bounds.

Robust Estimation. There are two cases where the association $(Q_i, \mathcal{P}_{h_i})$ can be wrong: if the initial position of Q_i is too far from its real position or if it is not (in the real scene) on the CAD model. In those two cases, $d(Q_i, \mathcal{P}_{h_i})$ could make the minimization fail. To limit this effect, we insert a robust M-estimator ρ in equation (2):

$$\min_{\mathbf{e}_1,\ldots,\mathbf{e}_m} \sum_i \rho(d(Q_i, \mathcal{P}_{h_i})) \quad (5)$$

We chose to use the Tukey M-estimator (Huber, 1981). The M-estimator threshold can be automatically fixed thanks to the Median Absolute Deviation (MAD). The MAD works with the hypothesis that the studied data follow a Gaussian distribution around the model. This assumption could be done for each individual fragment but not for the whole reconstruction. So we decided to use a different M-estimator threshold ξ_j per fragment. This implies that we have to normalize the Tukey values on each fragment:

$$\rho'_{l_i}(d(Q_i, \mathcal{P}_{h_i})) = \frac{\rho_{l_i}(d(Q_i, \mathcal{P}_{h_i}))}{\max_{Q_j \in \mathcal{B}_{l_i}} \rho_{l_i}(d(Q_j, \mathcal{P}_{h_j}))} \quad (6)$$

where l_i is the index of the fragment owning Q_i and ρ_{l_i} the Tukey M-estimateur used with the threshold ξ_{l_i}.

Fragment Weighting. With this cost function, each fragment will have a weight in the minimization proportional to the number of 3D points it contains. Then, fragments with few points could be not optimize in favour of the others. To give the same weight to each fragment, we must unified all the Tukey values of their 3D points with respect to their cardinal:

$$\rho^*_{l_i}(d(Q_i, \mathcal{P}_{h_i})) = \frac{\rho'_{l_i}(d(Q_i, \mathcal{P}_{h_i}))}{card(\mathcal{B}_{l_i})} \quad (7)$$

and the final minimization problem is:

$$\min_{\mathbf{e}_1,\ldots,\mathbf{e}_{m+1}} \sum_i \rho^*_{l_i}(d(Q_i, \mathcal{P}_{h_i})) \quad (8)$$

that we solve using the Levenberg-Marquardt algorithm (Levenberg, 1944).

Iterative Optimisation. Practically, several non-linear minimisations are done successively with computing the point-plane association before each one of them. It enables 3D points to change their associated plane without losing cost-function computation time.

Initialisation. Non-linear algorithms require a correct initialisation: the 3D reconstruction should be placed in the same frame than the CAD model. To realize this stage, estimating an initial rigid transformation is sufficient when the 3D reconstruction is accurate. However, the drift of the scale factor frequently observed with SLAM reconstruction may induce important geometrical deformations. Therefore, to ensure convergence of the algorithm, we chose to place roughly each extremity $\mathbf{e}_i$ around the CAD model. It can be done automatically if the sequence is synchronised with GPS data for example. Otherwise, it could be realized manually through graphic interface.

4 EXPERIMENTAL RESULTS

In this part, we will present results on both synthetic and real sequences. The SLAM algorithm we use for our reconstruction is the one proposed by Mouragnon (Mouragnon et al., 2006).

4.1 Synthetic Sequence

Figure 4 presents the different steps of our experiment. The synthetic sequence (based on 3D model figure 4(a)) have been made by using the SLAM algorithm in a textured 3D world generated with a 3D computer graphics software. The followed trajectory is represented by the red arrows in figure 4(a).

Figure 2 underlines the original SLAM method drift. The scale factor is measured by computing the distance between two successive cameras and then computing the ratio for ground truth and reconstruction results. We see that this scale factor decreases in the course of the journey, making the 3D SLAM reconstruction being distorted (figure 4(b)). As consequence, the camera trajectory does not loop anymore.

For ICP initialisation, we have manually simulated GPS results: random position error have been assigned to each extremity (figure 4(c)).

Then we observe that after our treatment (figure 4(d)) the loop is restored while no loop constraint is directly included into our transformation model. Besides the 3D reconstructed point cloud regains its consistency. Table 1 confirms those enhancements: the average distance between the reconstructed cameras and the ground truth is reduced from more than 4 meters to about 50 centimetres. Statistics before ICP have been computed on the 5591 reconstructed 3D points (among the 6848 proposed by SLAM) kept as inliers by the ICP step. Furthermore, we can notice in figure 3 that only errors along the direction of the trajectory remain significant. This is due to the fact that we have supposed the error on scale factor strictly constant on each segment. Although this hypothesis reveals to be only a rough approximation.

Table 1: **Numerical results on the synthetic sequence.** Each value is a mean over all the reconstruction.

	Before ICP	After ICP
Camera distance to ground truth (m)	4.61	**0.51**
Standard Deviation (m)	2.25	**0.59**
3D points-Model distance (m)	3.37	**0.11**
Standard Deviation (m)	3.9	**0.08**
Tukey threshold	×	0.38

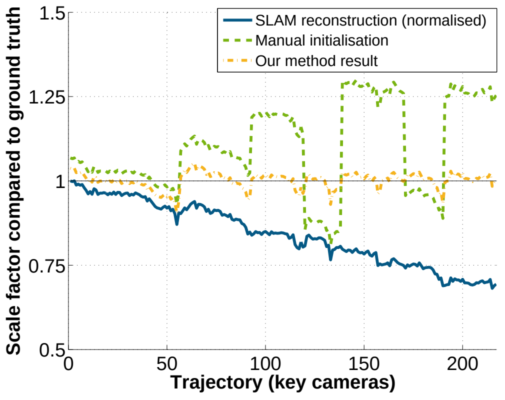
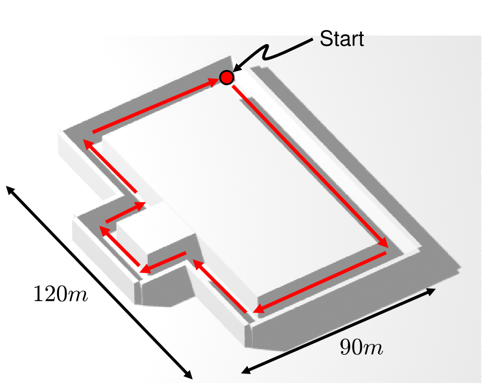


Figure 2: **Scale factor evolution.** We can observe the scale factor drift on the original SLAM reconstruction. After our method, scale factor is centred around 1.

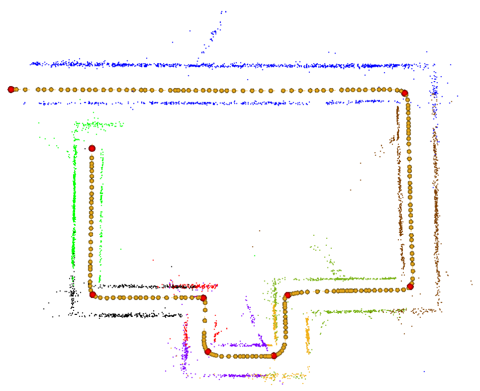
4.2 Real Sequence

The real sequence (figure 5) is a about 600m long loop. The SLAM reconstruction (figure 5(b)) is a good example of scale factor drift: we can see that after the roundabout, there is a radical change of scale factor and then the loop is not respected.

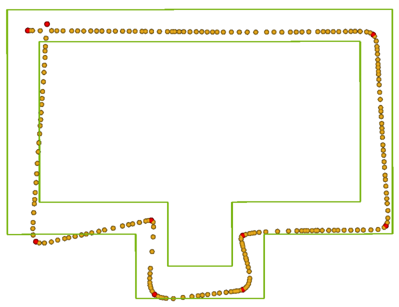
Since no ground truth is available for this sequence, we can only have a visual appreciation of our results. Nevertheless, two recognizable places in figures 5(a) and 6(b) tend to point out our method consistency: we find in "*A*" a crossroad due to a single way tunnel. Besides, in "*B*" we can show that we pass twice on the same lane because of a car breakdown.



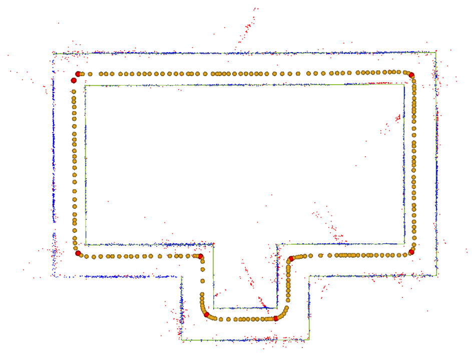
(a)



(b)

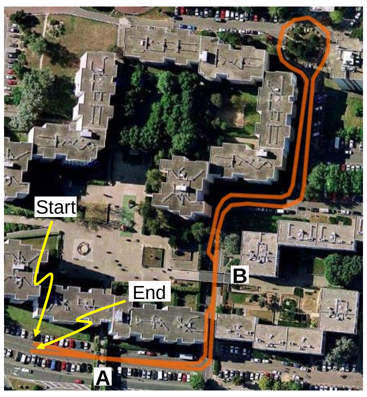


(c)

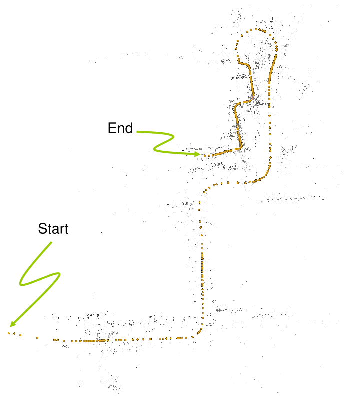


(d)

Figure 4: **Synthetic sequence treatment.** (a) Synthetic 3D model. (b) Top view of its 3D reconstruction with (Mouragnon et al., 2006) algorithm. Orange spheres are cameras while red ones are the fragment extremities. The automatic trajectory and point-fragment segmentation is also represented: the reconstructed 3D points are coloured by fragment belonging. Figure (c) is the user initialisation of the trajectory around the CAD model (in green) before the ICP step. Figure (d) shows the 3D reconstruction after our method: blue 3D points are inliers and red are outliers.



(a)



(b)

(c)

Figure 5: **Real sequence reconstruction using (Mouragnon et al., 2006).** (a) is a satellite map of the reconstructed trajectory displayed in (b). (c) is a frame of the video sequence.

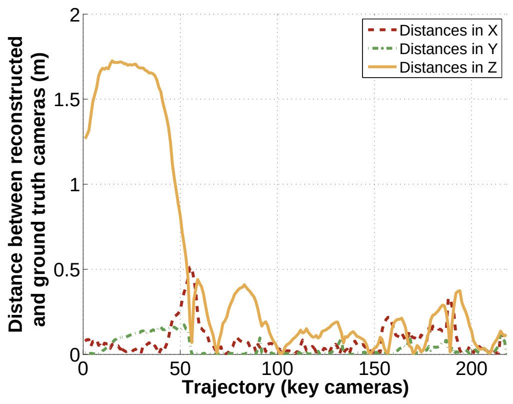
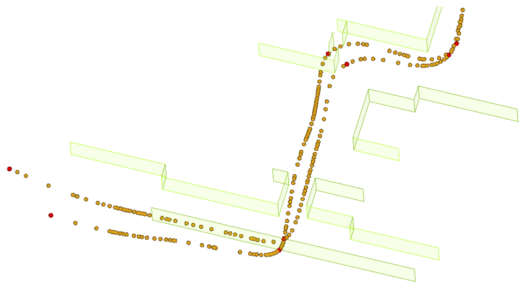


Figure 3: **Residual values on distances between reconstructed cameras and ground truth.** The (X, Y, Z) coordinates frame is relative to each camera : Z is the optical axis, X is the latitude axis and Y the altitude.

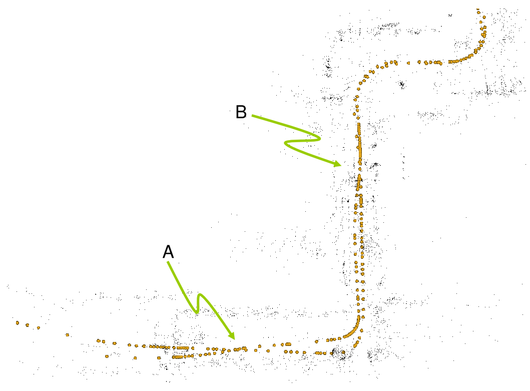
5 CONCLUSION

We have presented in this paper a post-processing algorithm that improves SLAM reconstruction. Our method uses an automatic fragmentation of the reconstructed scene and then a non-rigid ICP between SLAM reconstruction and a coarse CAD model of the environment. Synthetic and real experiments point out that method can clearly reduced the 3D positions error, in particular in X and Y directions: trajectory loop are restored and cameras position errors are reduced to about 50 centimetres on several hundreds meters sequences.

To improve our method, it would be interesting to evaluate the performance of an automatic initialisation (which could be provided by an additional sensor: GPS for example). We are also working on further refinements of the reconstruction by including the CAD model information into a bundle adjustment process.



(a)



(b)

Figure 6: **Real sequence treatment.** (a) ICP initialisation for real sequence. (b) Correction of real sequence reconstruction thanks to our method. "*A*" and "*B*" correspond to "*A*" and "*B*" places in figure 5(a).

REFERENCES

Castellani, U., Gay-Bellile, V., and Bartoli, A. (2007). Joint reconstruction and registration of a deformable planar surface observed by a 3d sensor. In *3DIM*, pages 201–208.

Davison, A., Reid, I., Molton, N., and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *PAMI*, 26(6):1052–1067.

Fitzgibbon, A. (2001). Robust registration of 2d and 3d point sets. In *BMVC*, pages 411–420.

Huber, P. (1981). *Robust Statistics*. Wiley, New-York.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 2:164–168.

Levin, A. and Szeliski, R. (2004). Visual odometry and map correlation. In *CVPR*, pages 611–618.

Lowe, D. G. (1987). Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., and Sayd, P. (2006). Real time localization and 3d reconstruction. In *CVPR*, pages 363–370.

Nister, D., Naroditsky, O., and Bergen, J. (2004). Visual odometry. In *CVPR*, pages 652–659.

Royer, E., Lhuillier, M., Dhome, M., and Chateau, T. (2005). Localization in urban environments: Monocular vision compared to a differential gps sensor. In *CVPR*, pages 114–121.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the ICP algorithm. In *3DIM*, pages 145–152.

Sourimant, G., Morin, L., and Bouatouch, K. (2007). Gps, gis and video fusion for urban modeling. In *CGI*.

Zhao, W., Nister, D., and Hsu, S. (2005). Alignment of continuous video onto 3d point clouds. *PAMI*, 27(8):1305–1318.