On-line Convex Optimization based Solution for Mapping in VSLAM

by

Abdul Hafeez, Shivudu Bhuvanagiri, Madhava Krishna, C.V.Jawahar

in

IROS-2008
(*Intelligent Robots and Systems*)

Report No: IIIT/TR/2008/178

Centre for Robotics
International Institute of Information Technology
Hyderabad - 500 032, INDIA
September 2008

On-line Convex Optimization based Solution for Mapping in VSLAM

A.H. Abdul Hafez[1], Shivudu Bhuvanagiri[1], K Madhava Krishna[1] and C.V. Jawahar[1]

***Abstract*— This paper presents a novel real-time algorithm to sequentially solve the triangulation problem. The problem addressed is estimation of 3D point coordinates given its images and the matrices of the respective cameras used in the imaging process. The algorithm has direct application to real time systems like visual SLAM. This article demonstrates the application of the proposed algorithm to the mapping problem in visual SLAM. Experiments have been carried out for the general triangulation problem as well as the application to visual SLAM. Results show that the application of the proposed method to mapping in visual SLAM outperforms the state of the art mapping methods.**

I. INTRODUCTION

The triangulation problem in computer vision is defined as the estimation of an unknown 3D point given its images from different views; the camera pose parameters are assumed to be known. Convex optimization techniques produce an optimal solution for this problem by minimizing the L_∞ norm. This paper deals with the case when the image and camera information are available sequentially. A new image measurement and the corresponding camera matrix are assumed to be provided at every time instant. The solution needs to be recursively integrated with time while keeping the optimality of the solution through convex optimization.

Convex optimization has been widely accepted as a powerful tool to solve many engineering problems [1]. Its use has been extensively explored for solving a family of geometric reconstruction problems in computer vision. A wide variety of computer vision problems can be reformulated as convex optimization problems by some algebraic manipulations [2]. Convex optimization, most importantly, does not have the pitfall of local minima assuring to achieve an optimal solution. In addition, Jacobian estimation through building a linearized model is not required in contrast to iterative minimization methods like gradient descent and Newton methods. The optimization is done here by replacing the L_2 norm objective function by the L_∞ norm *i.e.*, the maximum re-projection error (image point distance) for a set of image points.

Hartley and Schaffalitzky have shown in [3] that most of the multi-view computer vision problems have a global minimum by minimizing the L_∞ norm. Soon after that Kahl in [4], and Ke and Kanade in [5] have independently shown that this L_∞ is a quasi-convex function. Thus, it can be efficiently solved as a *sequence of second order cone programs* (SOCP) using the well known bisection algorithm. The solution proposed by Ke and Kanade [5] uses the m^{th} smallest norm L_m instead of L_∞. This is due to the sensitivity of L_∞ to noise and outliers while the former one is robust. However, the solution in [5] becomes a local solution by considering the L_m norm as an objective function. Later, it was shown that another robustness technique can be introduced to the minimization of the L_∞ norm while retaining the ability to produce a global solution [6].

A wide variety of multi-view computer vision problems can be solved efficiently by minimizing the L_∞ norm using convex optimization framework. These problems may include triangulation, camera re-sectioning, plane to plane homography construction, estimation of camera motion with a given rotation, *etc.* [2], [4]. However, the existing framework is suitable only for problems related to building a model of a given environment. For example, given a video sequence with a calibrated camera, the 3D model of the scene can be efficiently built. This is usually known in computer vision community as the *batch processing* techniques. However, many computer vision problems need to be processed *on-line* satisfying the real time constraints.

This paper demonstrates the adaption of triangulation problem to real time situations. To adapt to the real time requirements, we formulate the solution of the quasi-convex problem using bisection algorithm as an on-line recursive method. The only sequential solution to the minimization of L_∞ norm has been recently proposed in [7]. On contrary to the method in [7] we do not assume an initial solution of the problem is available. This solution is obtained by running the full bisection algorithm which is usually time consuming. However, the algorithm proposed in this article keeps track of the changes in the estimated variables by updating the objective function and runs only one or two iterations of the bisection algorithm. It starts from a uniform distribution of the unknown variable and at each iteration it updates the objective function giving an approximated estimate of the state variable. These estimates are assumed to converge to the accurate solution after a few iterations.

The real time solution has applications in many problems like visual SLAM, structure from motion (SFM), virtual reality, visual servoing. The adaptation to the real time case demonstrated by our algorithm can be applied to a wide variety of computer vision problems solved by minimizing the L_∞ using convex optimization framework.

The sequential triangulation algorithm proposed in this paper is suitable to be used within VSLAM framework due to the analogy of VSLAM problem with the SFM problem. In case enough parallax is available, the 3D estimate is sufficiently accurate and available immediately at the second iteration (when the second image measurement is acquired). For far features, a deterministic estimate is also available

[1] International Institute of information Technology, Gachibowli, Hyderabad-500032, India hafez@research.iiit.ac.in

at the second observation; it can be also used in the EKF framework.

The contribution of the paper is an efficient real time solution to the mapping problem in visual SLAM. The algorithm can run in the real time since the time needed for running SOCP program is a few milliseconds (5-8 ms) only. It produces a deterministic solution as soon as two images are available. The accuracy is reflected by re-projection error which is as small as 1-2 pixels on average. This technique is consistent with the EKF framework for visual SLAM where the inverse depth parametrization is used without any delay for the feature initialization.

II. BACKGROUND OF VSLAM AND MAPPING

The simultaneous localization and mapping (SLAM) [8], [9] is the problem of building a consistent map of the environment, *i.e.* a set of 3D features or landmarks, while simultaneously trying to localize in the map being constructed. Currently, robotics and computer vision researchers have attacked the problem with vision as the primary sensing modality. Consequently, the problem becomes visual SLAM.

Visual SLAM problem is more analogous to the well known structure from motion (SFM) problem in the computer vision community. Particularly, successful results were obtained by using Bayesian filtering for casual or recursive SFM based only on the observations up to the current time. Thus, the visual SLAM problem can be defined as the simultaneous real time estimation of the 3D pose of a moving camera and the structure of the environment. The pioneering work on visual SLAM has been presented by Davison [10] using the Extended Kalman Filter (EKF). More recently, Eade and Drummond [11] presented a monocular version of the FastSLAM algorithm [9] that uses particle filter to represent the motion distribution. Earlier attempt to use particle filter in visual SLAM were presented in [12], [13]. A robust real-time visual SLAM using scale prediction and exemplar based feature description has been recently proposed in [14].

Davison has presented in [10] a visual SLAM algorithm within a stochastic framework by using EKF, and an efficient method for feature tracking and matching based on active search. The uncertainty of the 3D estimates was used to constrain the search within an elliptical region in the image. Scalable monocular visual SLAM using FastSLAM algorithm [9], [15] was presented in [11]. The FastSLAM algorithm represents the motion distribution by a set of particles M and one Kalman filter for each particle, each one of the M features is estimated independently using one Kalman filter. This takes benefits of the fact that the estimates of 3D features are conditionally independent from each other given an estimate of the camera pose. The work presented in [13] was one of the early works that uses particle filter to represent the motion hypothesis by a set of particles but the focus of the paper is on robust camera localization.

A. Mapping in Visual SLAM Problem

The feature initialization process in [10] is based on a one dimensional particle filter to represent the depth of a newly observed feature. The filter is a set of particles (hypotheses) that is uniformly distributed along the projection ray of the feature. The image measurements collected from the consecutive frames are used to iteratively update the distribution of the depth particles. The feature initialization process is completed by inserting the estimated 3D feature along with its uncertainty in the map. This insertion is done when the depth distribution has converged to Gaussian. One may note that the depth range is limited since the number of particles is determined in advance and cannot be chosen arbitrarily. Since the number of depth particles affect the performance in the real time, it is chosen in such a way to keep the computations as less as possible.

On the other hand, in [11] the depth hypotheses are drawn uniformly in the inverse depth. This take advantages from the fact that the images of these uniform hypotheses in the inverse depth appear uniformly distributed along the epipolar line corresponding to the concern feature in later views. In fact, Kalman filter is used to update the inverse depth estimate due to the near linearity and faster convergence to Gaussian distribution.

More recently, Montiel *et al.* have proposed in [16], a unified inverse depth parametrization for monocular SLAM. The proposed parametrization is able to handle both initialization and tracking of near and far features with standard EKF. This parametrization maintains the direction of the 3D feature in the camera frame where it was initially observed and detected in addition to the camera position as well as the inverse depth in the initial camera frame. Furthermore, it has been also shown in [16] that the feature initialization does not need any filtering technique, and hence the feature is introduced directly in the map. However, the advantage of this parametrization is restricted to the possibility of directly introducing the feature into the map and use it to update the state vector. In fact, the depth of the 3D feature is available only when enough parallax is available.

In other words, the only improvement of the mentioned unified parametrization is unifying the initialization and the tracking stages of certain features on the account of increasing the size of state vector. We propose in this paper that the original 3-vector style of feature parameterization can still be used in EKF framework where the inverse depth is used instead of depth. Since the inverse depth is used, it will be possible to handle far features when there is no knowledge about depth uncertainty nor enough observation to reduce it is available.

B. Mapping via Convex Optimization

Since knowledge about the uncertainty of the depth is not available and there are no sufficient observations to estimate it, we propose a deterministic model. The 3D feature estimates are initialized using a robust and deterministic method built within convex optimization framework. In this

case, an estimate of the depth (inverse depth) can be sequentially available and consistent with the EKF framework due to the sequential triangulation algorithm proposed in this paper (Section III-B). However, the uncertainty of the image measurement is propagated back to the Cartesian space and introduced to the covariance matrix of the state vector.

III. MAPPING OF 3D FEATURES IN VSLAM BY CONVEX OPTIMIZATION

A. Triangulation by Convex Optimization

The mapping process can be modeled as a quasi-convex optimization problem with respect to the 3D coordinates of the features (X, Y, Z). The model considers that N images of a 3D point M with $\mathcal{P}_i$ cameras are available, where $i = 1, \cdots, N$ and $\mathcal{P}_i$ is the i^{th} camera matrix. Since the camera matrices $\mathcal{P}_i$ are estimated as pose distribution, independently of the 3D structure, our mapping process can be reduced to a simple triangulation problem of the 3D point M given its N images and N camera matrices. This problem has been formulated within a quasi-convex optimization framework [4], [17], [5], [18]. To adapt to the real time requirements, we formulate the solution of the quasi-convex problem using bisection algorithm as an on-line recursive method.

The convex optimization problem is one that minimizes an objective function $f_0(M)$ subject to a set of constraints $f_i(M)$; where $i = 0, \cdots, N$, and the functions $f_i(M)$ with $i = 0, \cdots, N$ are convex. Given the function $f_0(M)$ is quasi-convex and the constraint functions are convex functions, the problem is a quasi-convex optimization problem and can be efficiently solved by solving a set of convex feasibility problems using the bisection algorithm. Kahl proposed in [4] to formulate the triangulation problem as

$$\begin{aligned} &\min \quad \max_i d(\hat{m}_i - \mathcal{P}_i M) \\ &\text{subject to} \quad \lambda_i(M) > 0 \qquad i = 1, \cdots, N. \end{aligned} \tag{1}$$

$\lambda_i(M)$ is depth of the point in image i. The function $d(\hat{m}_i - \mathcal{P}_i M)^2 = \frac{f_1(M)^2 + f_2(M)^2}{\lambda_i(M)^2}$ here is the Euclidean distance between the image measurement $\hat{m}_i$ and the projection of the 3D point M to the image of the camera $\mathcal{P}_i$. This distance can be written in more detail as

$$d_i^2 = \left(\hat{m}_i^1 - \frac{P_i^1 M}{P_i^3 M}\right)^2 + \left(\hat{m}_i^2 - \frac{P_i^2 M}{P_i^3 M}\right)^2. \tag{2}$$

Here, P_i^j is the j^{th} row of the i^{th} camera matrix $\mathcal{P}_i$.

This problem can be solved using the bisection algorithm via a sequence of convex feasibility problems of the form

$$\begin{aligned} &\text{find} \quad M \\ &\text{subject to} \quad \| \left[f_{i1}(M)^2, f_{i2}(M)^2\right] \| \leq \gamma \lambda_i(M) \\ &\qquad \lambda_i(M) > 0 \qquad i = 1, \cdots, N. \end{aligned} \tag{3}$$

The convex feasibility problem tries to find out whether the optimal solution $f_0^*(M)$ is less or more than a given value γ. Thus, the quasi-convex function given in Eq. (1) can be solved using a sequence of feasibility problems. The

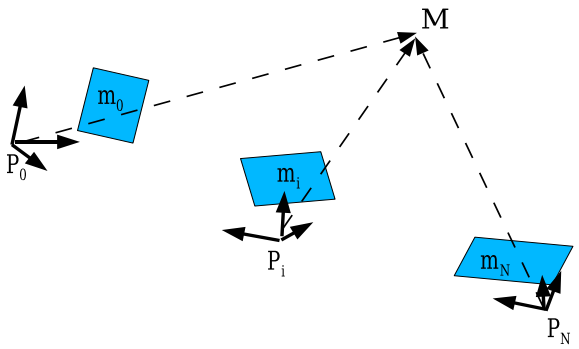

Fig. 1. The triangulation problem from multiple views. Given N camera matrices $\mathcal{P}_i$ along with the image measurements of a 3D point M from the corresponding N images m_i, compute the unknown point M.

authors in [4], [5] solve the problem given a set of N images of the 3D point M. The bisection algorithm iterates over the measurement values k times such that the objective function converges to a value $f_0^*(M) \leq \epsilon$. As illustrated in Algorithm 1, the algorithm starts from a known lower and higher bound $\left[\gamma^l, \gamma^h\right]$ of the unknown optimal solution f_0^*. The first step is to solve the feasibility problem given in (3) for the lower half of the range $\left[\gamma^l, \gamma^h\right]$. If it is feasible for this lower half, update the higher bound feasibility range as $\gamma^h = (\gamma^l + \gamma^h)/2$. If the problem in (3) is not feasible, the optimal solution satisfies $f_0^*(x) > \gamma$. In this case the lower bound is updated as $\gamma^l = (\gamma^l + \gamma^h)/2$. After certain number of iterations, say k, and solving a set of k feasibility problems, the range of the feasibility becomes $\gamma^h - \gamma^l \leq \epsilon$ and the produced solution is optimal. In fact, the k iterations, usually k is from 5 to 10, cannot be performed in the real time (the video rate).

Algorithm 1 Off-line Solution to the Quasi-convex Problem via Bisection Algorithm

1: **Input:** Given N image measurements, the range $\left[\gamma^l, \gamma^h\right]$ of the optimal value $f_0^*(M)$, and tolerance $\epsilon > 0$.
2: **Repeat**
   a) $\gamma = (\gamma^l + \gamma^h)/2$.
   b) Solve the convex feasibility problem as in (3).
   c) **If** feasible, $\gamma^h = \gamma$;
      else, $\gamma^l = \gamma$.
3: **Until** $\gamma^h - \gamma^l \leq \epsilon$.

B. On-line Bisection algorithm for Mapping

We propose an efficient on-line sequential bisection algorithm that satisfies the real time constraints. In addition, an inverse depth representation of the features is given along with uncertainty computation.

Recollecting from the previous section, the original *batch* 3D feature estimate problem is defined as follows. Given a set of N image correspondences $\hat{m}_i$ and the camera(s) information, estimate the 3D point M with minimal re-projection error over these image point correspondences. This problem is solved by Kahl in [4]. If some initial optimal solution is already available from observed N image correspondences and their respective camera matrices, the problem definition

takes the form: Given a set of N image correspondences and an optimal estimate M_N^* of its corresponding 3D point, estimate the new optimal 3D point M_{N+1}^* when a new image measurement $\hat{m}_{N+1}$ is acquired. This solution can still be sequentially updated to be an optimal one when new observations are available. This problem is discussed by Seo *et al.* in [7]. This article attacks the case when there is no initial optimal solution available.

Algorithm 2 On-line Sequential Solution to the Quasi-convex Problem via Bisection Algorithm

1: **Input**: $N = 2$ image measurements, optimal value $f_0^*(M) \in \left[\gamma^l, \gamma^h\right]$ and tolerance ϵ. Initially, the time parameter t is set to N.
2: **Repeat**
   a) Collect the measurements from the N^{th} image.
   b) **If** $d(\hat{m}_N - \mathcal{P}_N M) \geq \gamma^h$ **then**, $\gamma^h = d(\hat{m}_N - \mathcal{P}_N M)$.
   c) $\gamma = (\gamma^l + \gamma^h)/2$.
   d) Solve the convex feasibility problem as in (3).
   e) **If** feasible, $\gamma^h = \max_i d(\hat{m}_i - \mathcal{P}_i M)$, **else**, $\gamma^l = \gamma$.
   f) $N = N + 1$
3: **Until** $\gamma^h - \gamma^l \leq \epsilon$

The proposed method does not assume any initial optimal solution. In addition, the image correspondences $\hat{m}_i$ are presented sequentially starting from $i = 1$, till $i = N$, the time instant when the $\hat{m}_N$ measurement is presented. The problem here is to sequentially estimate the optimal solution M_N^* as soon as two image measurements are available. It is not necessary for this solution to be optimal for the first few frames or iterations. In other words, our task is to find a 3D estimate M_i, starting from $i = 2$ that converges to the optimal one M_i^* soon after a few image measurements, say k images are available.

Let us recall our assumption about a moving camera attached to frame F^t and has the camera matrix $\mathcal{P}_t$ at every time instant t. The camera frame F^0 is assumed to be the reference frame. Let us note that γ is set as an upper bound of the objective function in problem (1). Consequently, we can say that $d(\hat{m}_i - \mathcal{P}_i M) < \gamma$ for $i = 1, \cdots, N$. Assume that the optimal value γ^* is bounded between a known lower threshold γ^l and higher one γ^h. Finally, the sequential bisection algorithm works as described in Algorithm 2.

The algorithm starts as soon as two views $N = 2$ are available. At each time instant t, the algorithm solves single convex feasibility problem of the form in (3) as one iteration of the quasi-convex triangulation problem given in (1). Whenever new image measurements are available in the next time instant, the objective function $\max_i d(\hat{m}_i - \mathcal{P}_i M)$ and the bounding thresholds are updated based on the new measurements.

Recollecting that γ^h is the higher bound of the objective function, i.e. the L_∞ norm, this γ^h should be equal or greater than the re-projection error $d_i = d(\hat{m}_i - \mathcal{P}_i M)$ in all the considered images. Thus, for the N^{th} observed image, if the re-projection error d_N is greater than the higher bound γ^h, then this higher bound is set equal to the re-projection error d_N. Following this the convex feasibility problem in (3) is solved once. This feasibility problem results in an update of the higher bound γ^h and the lower one γ^l if the problem is feasible; as well as an estimate of the 3D point M with a re-projection error within the range (γ^l, γ^h). Then, new iteration of convex feasibility problem is considered. This process is repeated whenever new image is acquired providing new measurement. The solution of this problem is observed to converge within 5-10 frames. Mapping process for another 3D feature can be started to be solved within a few frames.

C. Uncertainty Estimation of Mapped Features

The uncertainty in the 3D feature estimates by solving the convex feasibility problem can be approximated as Gaussian. To have the benefits of the inverse depth parametrization, the 3D feature will be introduced into the map as $\hat{M} = [X,\ Y,\ S]^T$ where $S = 1/Z$ is the inverse of the depth. Recall that the on-line bisection algorithm solves one convex feasibility problem at each time iteration on obtaining new image measurements. In addition, the solution returns one arbitrary value for the 3D estimate that satisfies the set of convex constraints posed by the problem; and belongs to the 3D domain of M determined by the current bounds $\left[\gamma^l, \gamma^h\right]$.

Consider that the returned value by the convex feasibility program at time t is $\hat{M}_t$ where $t = 1, \cdots, N$, the mean of the 3D feature estimate is the usual sum $\bar{M} = \frac{1}{N}\sum_{t=1}^{N} \hat{M}_t$ over the set of all the N scene estimates. Similarly, the covariance matrix $Q_{\hat{M}} = \frac{1}{N+1}\sum_{t=1}^{N}[\hat{M} - \bar{M}][\hat{M} - \bar{M}]^T$. One may note that the mean and covariance can be updated recursively without having to store the previous estimates. The mean 3D coordinates at time instant t is given as

$$\bar{M}_t = \frac{t-1}{t}\bar{M}_{t-1} + \frac{1}{t}\hat{M}_t,$$

while the covariance matrix is updated as

$$Q_t = \frac{1}{t+1}\left\{t\ Q_{t-1} + [\hat{M}_t - \bar{M}_t][\hat{M}_t - \bar{M}_t]^T\right\}.$$

This estimate is assumed to converge to Gaussian distribution within a few iterations of solving the convex feasibility problem.

In fact this uncertainty computation can be applied to the depth (inverse depth) while the uncertainty in the X and Y coordinates need to be updated with the uncertainty measurement space $\mathbb{R}$ in the image measurement x and y. More information about uncertainty propagation can be found in [19]. The presented paper addresses only *mapping problem* in VSLAM framework and all the necessary processing is done as in [16]. Regarding the uncertainty covariance matrix, we are working on providing a regrigous method to compute it. Further work also includes developing a complete system (feature estimation process, uncertainties and SLAM algorithm).

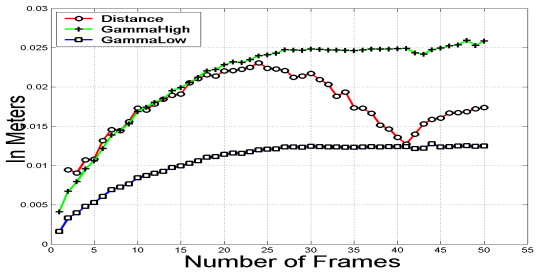

Fig. 2. The L_∞ error (red) in pixels over the iterations of the initialization process along with the upper (green) and lower (blue) bounds of the error within the on-line bisection algorithm.

IV. EXPERIMENTAL RESULTS

In this section we show the results that demonstrate the efficiency of our proposed method. We present two kinds of results. First, we show preliminary results illustrating the on-line triangulation algorithm. Second, the application of the algorithm to mapping for VSLAM is shown. The data considered for experiments is the real video sequence acquired with hand-held low cost Unibrain IEEE 1394 camera, with a 90° field of view and 320x240 resolution monochrome at 30 fps. In fact this sequence is available on the web by the authors of [16]. In addition, we have used the Matlab code provided by them for both comparison of their mapping method with ours and for plugging our mapping algorithm to the VSLAM framework. In other words, all kind of processing steps are same as in [16] except the mapping stage. We compare our results with [16] which is considered as the best reported algorithm.

A. Results from On-line Triangulation

The on-line triangulation algorithm is applied to data collected from the mentioned video sequence. The camera matrices corresponding to the sequence images are collected from the localization stage output of the VSLAM framework. However, we present these camera matrices (camera pose

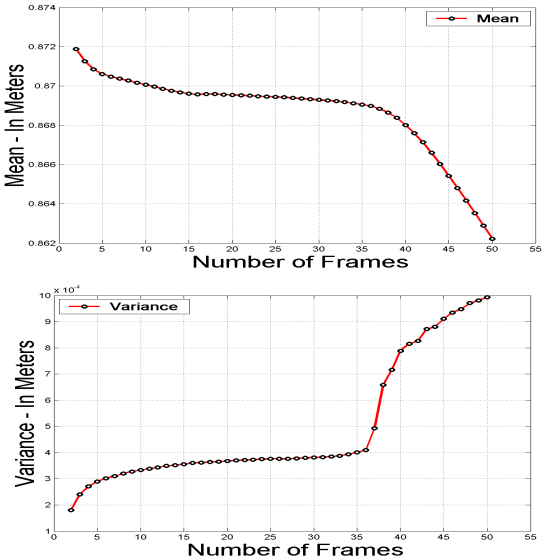

Fig. 3. The depth distribution during the feature initialization using the convex feasibility problem. The mean at the top and variance at the bottom. It is also to be noted that the difference in the values of ordinate across the frame is of order 10^{-3}. We see a convergence in mean and the minimal difference shows that the initial estimate is resonably accurate.

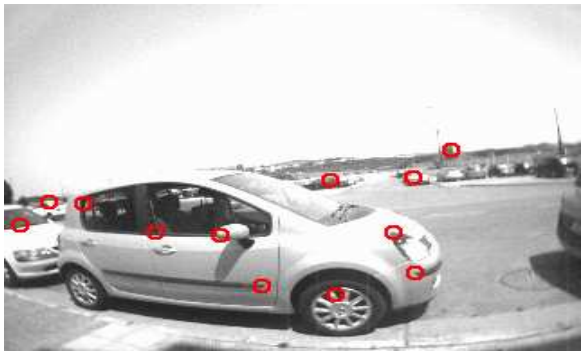

Fig. 4. A picture from the video sequence with near features.

Fig. 5. A picture from the video sequence with distant features.

with respect to the reference frame) to the triangulation algorithm assuming that they are correct estimates. The L_∞ error, averaged over all features, along with its boundaries γ^l and γ^h are illustrated in Fig. 2. It is clear that the L_∞ error converges to small values within the acceptable range of γ^l and γ^h. The mean and variance of the estimated depth are shown in Fig. 3. The mean at the top and variance at the bottom. One may note that there are no notable variations of the mean after the third iteration.

B. Results from Mapping and VSLAM

To illustrate the applicability of our on-line triangulation method to VSLAM, we have applied it for mapping many features detected and extracted from the said video sequence. We have selected features that belong to two different classes, particularly far features that appear to be at infinity and near features that provide enough parallax. In addition, we compare the re-projection error for the two classes of features with the mapping method used in [16] which is the only available VSLAM implementation for use. Figures 4 and 5 show the selected near and far features respectively.

The re-projection error in pixels, averaged over all features, for near features is shown in Fig. 6. The average RMS error is less than three pixel which is rather very small. Indeed, the mapping of the 3D features is precise and accurate. Similarly, the re-projection error in pixels for far features is shown in Fig. 8. The average is about five pixels. This means that the accuracy of the 3D estimates is less for far features than the one for near features. This is due the amount of parallax available from near features motion.

A comparison with the mapping results of the method presented in [16] is potrayed in Figs. 7 and 9 for near and far

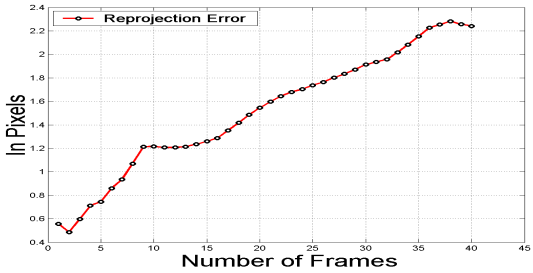

Fig. 6. The RMS re-projection error (averaged over all features) in pixels for a set of near features over the iterations of the initialization process.

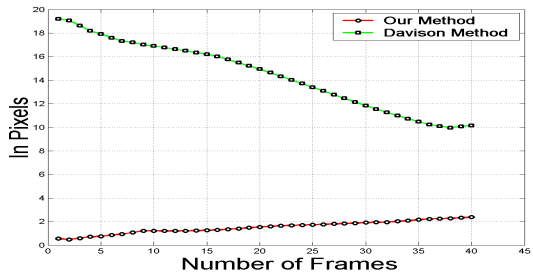

Fig. 7. A comparison with work in [16], our method in red color, of the RMS re-projection error in pixels for a set of near features over the iterations of the initialization process . It is average of all feature.

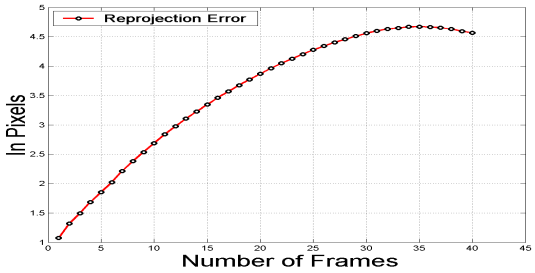

Fig. 8. The RMS re-projection error (averaged over all features) in pixels for a set of distant features over the iterations of the initialization process.

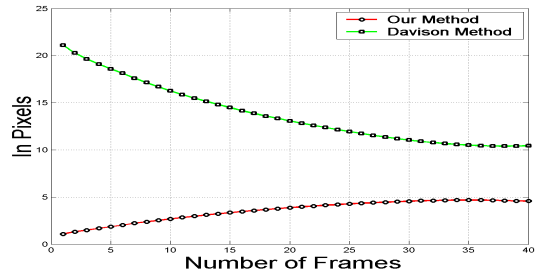

Fig. 9. A comparison with work in [16], ours in red color, of the RMS re-projection error in pixels for a set of distant features over the iterations of the initialization process (Average of all feature at the top).

features respectively. In the case of near feature, our mapping method clearly performs better than the method in [16]. Even for far features the re-projection error is minimal. The error of our mapping method is presented in red color while the error from the method presented in [16] is in green color.

V. Conclusion

We have presented an efficient sequential algorithm for solving the triangulation problem with application to mapping in VSLAM. The algorithm starts from unknown estimate and its output converges to the actual values within a few iterations. It estimates the depth of the image features for VSLAM system. The proposed method has demonstrated an improved 3D estimates during the mapping stage of the visual features on comparison with state of the art algorithms. The time needed for running SOCP program is a few milliseconds (5-8 ms) only which makes the algorithm suitable for real-time applications. The minimal re-projection error for near and far features delineates the efficacy of the algorithm.

References

[1] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[2] R. Hartley and F. Kahl, "Optimal algorithms in multiview geometry," in *Computer Vision – ACCV 2007*, ser. LNCS, Y. Yagi, S. B. Kang, I. S. Kweon, and H. Zha, Eds., vol. 4843. Springer, 2007, pp. 13–34.

[3] R. Hartley and F. Schaffalitzky, "L_∞ minimization in geometric reconstruction problems," in *Proceedings CVPR'04*, vol. 01. Los Alamitos, CA, USA: IEEE Computer Society, 2004, pp. 504–509.

[4] F. Kahl, "Multiple view geometry and the L_∞ norm," in *Proceedings of ICCV'05*, Beijing, China, 2005, pp. 1002–1009.

[5] Q. Ke and T. Kanade, "Quasiconvex optimization for robust geometric reconstruction," in *Proceedings of ICCV '05*, Beijing, China, 2005, pp. 986–993.

[6] K. Sim and R. Hartley, "Removing outliers using the l_∞ norm," in *Proceedings of CVPR'06*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 485–494.

[7] Y. Seo and R. Hartley, "Sequential L_∞ norm minimization for triangulation," in *Computer Vision – ACCV 2007*, ser. LNCS, I. S. K. Yasushi Yagi, Sing Bing Kang and H. Zha, Eds., vol. 4844. Springer, 2007, pp. 322–331.

[8] G. Dissanayake, P. Newman, H. Durrant Whyte, S. Clark, and M. Csorba, "A solution to the simultaneous location and map building (slam) problem," *IEEE Transaction on Robotics and Automation*, vol. 17, no. 2, pp. 229–241, May 2001.

[9] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," Proc. of AAAI, Edmonton,Canada, 2002.

[10] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *International Conference on Computer Vision, ICCV'03*, Nice, France, October 2003.

[11] E. Eade and T. Drummond, "Scalable monocular slam," in *Proceedings of CVPR'98*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, 2006, pp. 469–476.

[12] R. Sim, P. Elinas, M. Griffin, and J. J. Little, "Vision-based SLAM using the Rao-Blackwellised particle filter," in *Proceedings of the IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR)*, Edinburgh, Scotland, 2005, pp. 9–16.

[13] M. Pupilli and A. Calway, "Real-time camera tracking using a particle filter," in *British Machine Vision Conference, BMVC'05*, Oxford, September 2005, pp. 519–528.

[14] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway, "Robust real-time visual slam using scale prediction and exemplar based feature description," in *Proceedings of CVPR'07*. Minneapolis, Minnesota, USA: IEEE Computer Society, 2007.

[15] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of IJCAI'03*. Acapulco, Mexico: IJCAI, 2003.

[16] J. M. M. Montiel, J. Civera, and A. J. Davison, "Unified inverse depth parametrization for monocular slam," in *Robotics: Science and Systems*, 2006.

[17] F. Kahl and D. Henrion, "Globally optimal estimates for geometric reconstruction problems," in *Proceedings of ICCV '05*, Beijing, China, 2005, pp. 978–985.

[18] Q. Ke and T. Kanade, "Uncertainty models in quasiconvex optimization for geometric reconstruction," in *Proceedings of CVPR'06*, New York, NY, USA, 2006, pp. 1199–1205.

[19] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2003.