A Graduated Filter Method for Large Scale Robust Estimation Supplementary Material

Huu Le and Christopher Zach
Chalmers University of Technology, Sweden
{huul, zach}@chalmers.se

Abstract

This document provides further details for our paper, including detailed algorithm for the step computation, discussions on the choice of parameters and additional results for large-scale robust estimation problem.

1. Additional Experimental Results

In this section, we provide more experimental results for the robust bundle adjustment experiment. The list of 20 instances that are used for the experiments includes: Trafalgar-126, Trafalgar-138, Dubrovnik-150, Dubrovnik-16, Trafalgar-201, Dubrovnik-202, Trafalgar-21, Trafalgar-225, Dubrovnik-253, Ladybug-318,Final-93, Final-394, Venice-245, Dubrovnik-308, Dubrovnik-356, Venice-744, Venice-89, Venice-951, Trafalgar-39, Ladybug-49. Besides the results shown in the main manuscript, Figure 8 plots the results for 8 additional datasets.

2. Reconstruction Results

To demonstrate the concept of poor local minima, we show in this section the 3D reconstructed structures for two large datasets: Venice-89 and Final-394. The results are shown in Figure 9. Observe that by converging to lower robust costs, ASKER provides visually better structures compared to IRLS, which is easily trapped at a poor local minimum, resulting in higher objective values for most problem instances.

3. Detailed Algorithm for Step Computation

In this section, we summarize the detailed algorithm for step computation (Line 10, Algorithm 1). Note that we choose the initial value of λ to be $\lambda_{\text{init}} = 0.5$, and the values of λ is modified according to Algorithm 2. In particular, if $\mathbf{x}^{t+1}$ is accepted to the filter, we reduce λ, allowing more exploration for the following steps. However, when $\mathbf{x}^{t+1}$ returned by the cooperative step is not accepted, the value of λ is reset to λ_{init}, and the restoration step is executed.

Algorithm 2 Step Computation

Require: Current value $\mathbf{x}^t$, μ_f, μ_g, current damping value λ^t, $\lambda_{\text{init}} = 0.5$.
Compute $\mathbf{g}_f, \mathbf{g}_h, \mathbf{H}_f, \mathbf{H}_h$ from $\mathbf{x}^t$
Compute $\Delta\mathbf{x}$ using (13)
$\mathbf{x}^{t+1} \leftarrow \mathbf{x}^t + \Delta\mathbf{x}$
if $\mathbf{x}^{t+1} \notin \mathbb{F}$ **then**
  /*Perform Restoration Step*/
  $\Delta\mathbf{x} \leftarrow \gamma \begin{pmatrix} \mathbf{0} \\ -\mathbf{s} \end{pmatrix}$, γ is computed based on (15)
  $\mathbf{x}^{t+1} \leftarrow \mathbf{x}^t + \Delta\mathbf{x}$
  $\lambda^{t+1} \leftarrow \lambda_{\text{init}}$
else
  $\lambda^{t+1} \leftarrow \lambda^t/10$
end if
return $\mathbf{x}^t$

4. Parameter Choices

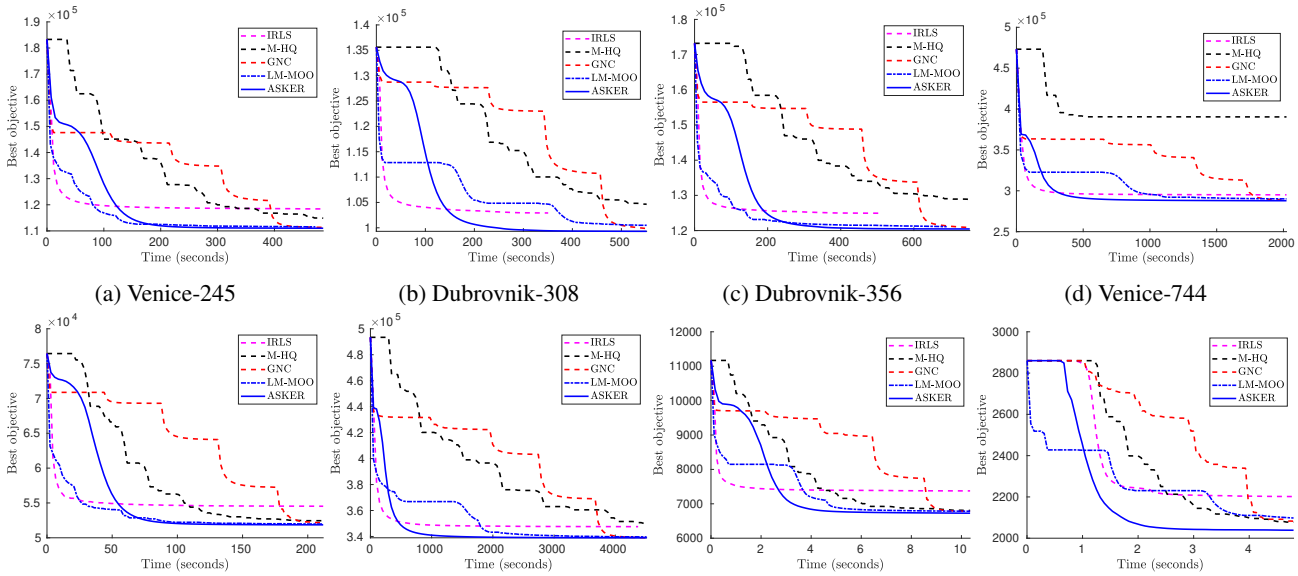
This section provides some study on the effects of parameter choices to the performance of our algorithm.

4.1. Filter Margin α

Figure 10 shows the evolution of the best cost with four different values of α. Observe that our algorithm is insensitive to the choice of $0.01 \leq \alpha \leq 0.1$, as the converged solutions are similar. However, when $\alpha \geq 0.2$ the converged objectives is higher, since the margin becomes unreasonably large that prevents the meaningful reduction provided by the cooperative step. In most experiments, we found $\alpha = 10^{-4}$ provides the best results, but other values of can also be used depending on the application and the starting values of the constraint violation.

4.2. Initial Scaling Values s_i

In Figure 11, we show the performance of our algorithm under different initialization values for s_i. Observe that when s_i are initialized with values s_0 that are less than 3, the converged objectives are poor, because the kernels are



(a) Venice-245 (b) Dubrovnik-308 (c) Dubrovnik-356 (d) Venice-744

(e) Venice-89 (f) Venice-951 (g) Trafalgar-39 (h) Ladybug-49

Figure 8: Additional Results for Robust Bundle Adjustment

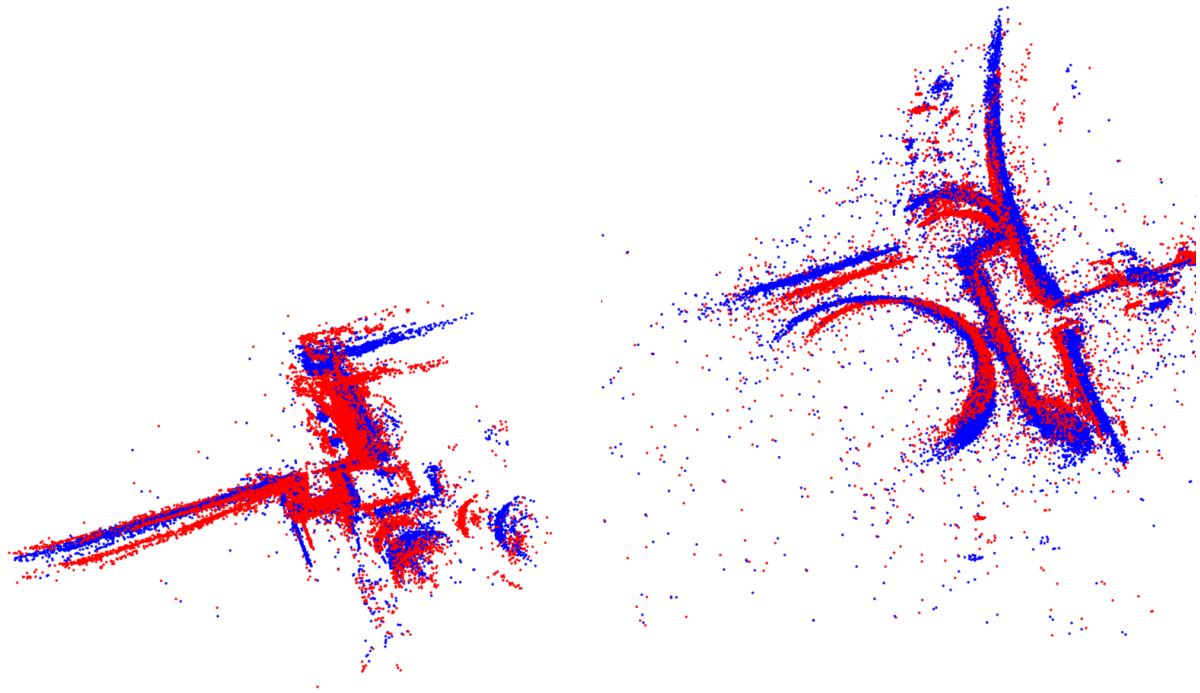


Figure 9: Top-down view of the reconstructed structures of Venice-89 (left) and Final-394 (right). Our results (ASKER) are plotted in blue, while IRLS results are shown in red. Observe that ASKER converges to better solutions, hence the 3D points form a better structure compared to IRLS.

not effectively scaled. When s_i are initialized to $s_0 \geq 3$, the converged objectives are similar. Figure 11 also demon-

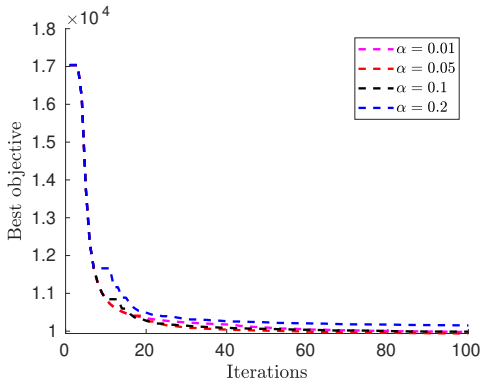


Figure 10: Convergence of algorithm for different values of alpha margin α.

strates that our algorithm is also insensitive to the initialization. In particular, even though when s_i are initialized to unnecessarily large values, the optimization process is still able to obtain competitive results within a few number of iterations.

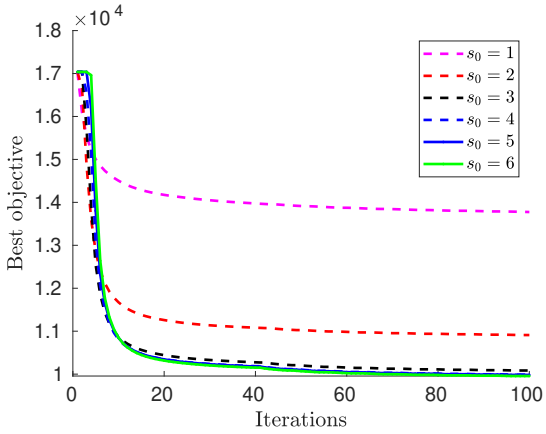


Figure 11: Convergence of algorithm for different initialization values of s_i.

5. Convergence

The convergence of our algorithm to $h = 0$ and a stationary solution of f is guaranteed by the nature of the filter algorithm [1]. In Figure 12, we plots the objective of h over the iterations (top), together with the evolution of the objective of the original function and f (bottom). Observe that f and h do not monotonically increased or decreased, but vary during the optimization process, as long as the pairs (f^t, h^t) are accepted to the filter. The objective of h converges to 0, while f and the real objective converge to the same value. The exploration of f and h in the filter drives the original objective to a local optimal solution. Note that after a number of iterations, the h becomes sufficiently small, and further exploration does not offer much improvement to the original objective. Therefore, one can define a early stopping criterion based on h. When all s_i are set to 0, our algorithm reverts to IRLS.

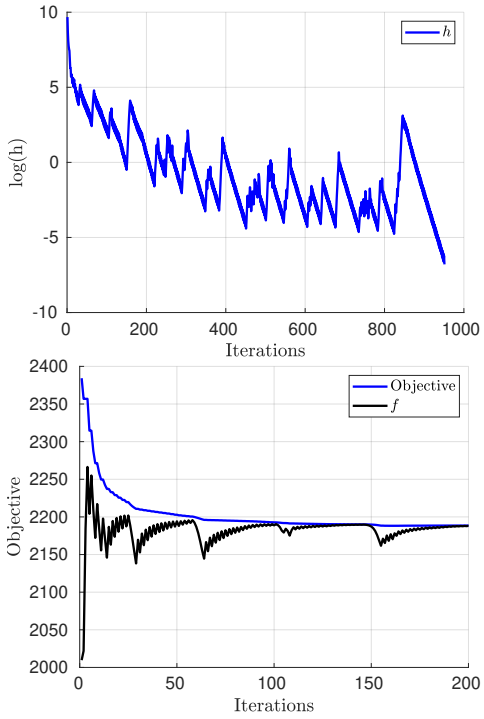


Figure 12: Convergence of algorithm to feasible region.

6. Gradient Computation

This section details the computation of $\mathbf{g}_f$, $\mathbf{g}_h$ and $\mathbf{H}_f$, $\mathbf{H}_h$ that are used in the step computation discussed in Section 5.2.1 of the main paper.

Recall the formulation of $f(\mathbf{x})$ and $h(\mathbf{x})$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \psi\left(\frac{\|\mathbf{r}_i(\boldsymbol{\theta})\|}{1+s_i^2}\right) \qquad h(\mathbf{x}) = \sum_i s_i^2, \tag{17}$$

where $\mathbf{x} = [\boldsymbol{\theta}^T \ \mathbf{s}^T]^T$ and $\mathbf{s} = [s_1 \dots s_N]^T$ as described in the main paper.

For brevity, let $\hat{\mathbf{r}}_i$ denote the "scaled" residual obtained by dividing the original residual by $1 + s_i^2$, i.e.,

$$\psi(\|\hat{\mathbf{r}}_i(\mathbf{x})\|) = \psi\left(\frac{\|\mathbf{r}_i(\boldsymbol{\theta})\|}{1+s_i^2}\right) = \psi\left(\left\|\begin{bmatrix} \frac{\mathbf{r}_i^1(\boldsymbol{\theta})}{1+s_i^2} \\ \vdots \\ \frac{\mathbf{r}_i^p(\boldsymbol{\theta})}{1+s_i^2} \end{bmatrix}\right\|\right) \tag{18}$$

where $\mathbf{r}_i^j$ $(j = 1 \dots p)$ is the j-th element of the residual vector $\mathbf{r}_i \in \mathbb{R}^p$.

Also, consider the following first-order approximation,

$$\psi(\|\hat{\mathbf{r}}_i(\mathbf{x}+\Delta\mathbf{x})\|) = \psi(\|\hat{\mathbf{r}}_i(\mathbf{x})+\mathbf{J}_i\Delta\mathbf{x})\|), \tag{19}$$

where $\mathbf{J}_i$ is the Jacobian of $\hat{\mathbf{r}}_i(\mathbf{x}+\Delta\mathbf{x})$ w.r.t. $\Delta\mathbf{x}$ evaluated at $\Delta\mathbf{x}=\mathbf{0}$.

To employ non-linear least squares solvers for our method, at each step, we utilize a convex quadratic majorizer $\hat{\psi}$ for each scaled residual vector $\hat{\mathbf{r}}_i$ such that $\psi(r) \leq \hat{\psi}(r)$. In this work, we make use of the well-known IRLS majorizer:

$$\hat{\psi}(\|\mathbf{x}+\mathbf{J}_i\Delta\mathbf{x}\|) = \frac{1}{2}\omega(\|\hat{\mathbf{r}}_i\|)(\|\hat{\mathbf{r}}_i+\mathbf{J}_i\Delta\mathbf{x}\|^2-\|\hat{\mathbf{r}}_i\|^2)+\psi(\|\hat{\mathbf{r}}_i\|) \tag{20}$$

where $\hat{\mathbf{r}}_i$ is a short hand notation for $\hat{\mathbf{r}}_i(\mathbf{x})$, and $\omega(r)$ with $r \in \mathbb{R}_+$ is defined as

$$\omega(r) := \frac{\psi'(r)}{r}, \tag{21}$$

which acts as the weight for the residuals.

For brevity, let ω_i denote $\omega(\|\hat{\mathbf{r}}_i\|)$. The equation (20) can be rewritten as

$$\hat{\psi}(\|\mathbf{x}+\mathbf{J}_i\Delta\mathbf{x}\|) = \frac{1}{2}\|\sqrt{\omega_i}\hat{\mathbf{r}}_i+\sqrt{\omega_i}\mathbf{J}_i\Delta\mathbf{x}\|^2-\frac{\omega_i}{2}\|\hat{\mathbf{r}}_i\|^2+\psi(\|\hat{\mathbf{r}}_i\|) \tag{22}$$

Based on (22), we can define a new Jacobian matrix and residual function for the function $\hat{\psi}$

$$\tilde{\mathbf{r}}_i = \sqrt{\omega_i}\hat{\mathbf{r}}_i \qquad\qquad \tilde{\mathbf{J}}_i = \sqrt{\omega_i}\mathbf{J}_i, \tag{23}$$

Then, under the LM framework, the gradient $\mathbf{g}_f$ and the approximated Hessian $\mathbf{H}_f$ can be computed as

$$\mathbf{g}_f = \sum_i \tilde{\mathbf{J}}_i^T\tilde{\mathbf{r}}_i \qquad\qquad \mathbf{H}_f = \sum_i \tilde{\mathbf{J}}_i^T\tilde{\mathbf{J}}_i \tag{24}$$

For the case of h, its gradient and Hessian can be computed in closed form:

$$\mathbf{g}_h = 2[\mathbf{0}\ \ \mathbf{s}]^T \qquad\qquad \mathbf{H}_h = 2\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_{N\times N}\end{bmatrix} \tag{25}$$

In our experiments, to obtain better results, in the cooperative step, we increase the damping for $\mathbf{H}$ to decrease the convergence rate of $h(\mathbf{x})$ (to zero). In particular, we use

$$\mathbf{H}_h = 2\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & (1+\lambda_h)\mathbf{I}_{N\times N}\end{bmatrix} \tag{26}$$

where λ_h is initially set to 2. During the iterations, λ_h is decreased by $\lambda_h \leftarrow 0.9\lambda_h$ if the new solution is accepted by the filter, otherwise reset to $\lambda_h \leftarrow 2$.

References

[1] Ademir A Ribeiro, Elizabeth W Karas, and Clóvis C Gonzaga. Global convergence of filter methods for nonlinear programming. *SIAM Journal on Optimization*, 19(3):1231–1249, 2008.