shipout/backgroundshipout/foreground

Lq-Bundle Adjustment

Khurrum Aftab Richard Hartley
Australian National University, ACT 2601, Australia
firstname.lastname@anu.edu.au

Abstract

In this paper we propose several methods to solve for a robust solution of bundle adjustment, a non-linear parameter estimation problem. Given a set of images of a scene, bundle adjustment simultaneously estimates camera parameters and 3*D structure of the scene. Generally, a least squares criterion is minimized by using the Levenberg-Marquardt (LM) method, a non-linear least squares optimization method. It is known that the least squares methods are not robust to outliers, even a single outlier can deviate the solution from its true value. Therefore, we propose several methods to minimize an Lq cost function, for* $1 \leq q < 2$*. The* L_q *cost function minimizes the the sum of the q-th power of errors. In addition to the Lq cost function, we propose several methods to minimize the Huber function. The proposed methods have an advantage of using the Levenberg-Marquardt (LM) method to find a robust solution of the problem. Our experimental results confirm that the proposed methods are more robust to outliers than the standard least squares method. This being so, the proposed methods trivially fit in the existing literature of the least squares bundle adjustment and have potential of being used as standard methods for bundle adjustment.*

1. Introduction

This paper presents extremely simple techniques to find a robust solution, especially an L_q solution, of the bundle adjustment problem and therefore contradict the perception that the L_q-bundle adjustment can not be done easily. An advantage of the proposed techniques is that they rely on the Levenberg-Marquardt (LM) method, a least squares minimization technique, to find a desired solution, even an L_q solution. Given a set of error vectors the LM method minimizes the sum of squared errors. We show that the minimization of a desired cost function can be achieved by using a modified difference vector or error vector, that is a vector representing the difference between measured image points and predicted image points, in the LM method.

The minimization of an L_q cost function is achieved by using two different methods. The first method, referred as the L_q *method*, minimizes the sum of the q-th power of errors by using modified error vectors in the LM method. The second method is an *Iterative Re-weighted Least Squares (IRLS)* technique where the sum of the q-th power of errors is minimized by iteratively minimizing a weighted least squares cost function, that is a weighted sum of squared errors. In addition to the L_q cost function, we propose a method to minimize the sum of the L_1 norms of error vectors, we refer to it as the *Absolute Value* method.

In addition to the L_q cost function, we propose several methods to minimize the Huber function. In robust statistics, the Huber function is a very popular differentiable function that has a hybrid behavior of a linear function and a non-linear function. A straightforward way of applying the Huber function for vector valued inputs is to apply the Huber function on each component of the input vector. However, we apply the Huber function on error vectors in the following two ways: Firstly, we apply the Huber function on the magnitude of error vectors instead of on each component of error vectors; we refer to this function as the *Isotropic Huber function*. Secondly, we introduce a re-thresholded version of the Huber function, we refer to it as the *Re-thresholded Huber* function. The Re-thresholded Huber function takes advantage of the iterative behavior of the LM method and decreases the threshold value of the Huber function after a fixed number of iterations. As a result, after some iterations the resultant function will have a dominant linear behavior and consequently increased robustness.

Given two or more images of a scene, bundle adjustment [6, 17, 3] simultaneously estimates camera matrices and 3D structure of the scene and is therefore a non-linear optimization problem. Several techniques exist in the literature to minimize a non-linear least squares cost function, such as the Gauss-Newton method, the Leveberg-Marquardt method, Trust Region methods, etc. The Levenberg-Marquardt method has become a fairly standard technique to solve the bundle adjustment problem. In this paper, we only focus on the Levenberg-Marquardt method for the minimization of desired cost functions.

Several strategies have been proposed to make bundle adjustment a practically feasible method, such as using sparse bundle adjustment [6, 2], hierarchical bundle adjustment [13], incremental bundle adjustment [14] that tunes parameters after every new frame arrives, spectral partitioning approach [15] for dividing large problem into smaller sub-problems, real time bundle adjustment [12], relative camera motion instead of absolute positions [7], probabilistic approach [4], etc. In this paper our focus is on increasing the robustness of bundle adjustment.

In recent years, the attention of a large group of research community has been directed to robust parameter estimation methods, especially L_1 optimization methods. A recent paper [16] uses Wiberg algorithm to solve the L_1 bundle adjustment problem but it require very different formulation and does not trivially fits in the paradigm of the existing bundle adjustment literature. On the other hand, we try to minimize this trade-off of simplicity and robustness by presenting methods that use the Levenberg-Marquardt method to find a robust solution of the bundle adjustment problem, especially an L_q solution for $1 \leq q < 2$. Thus, the proposed methods trivially fit in the existing literature of the bundle adjustment and only require a slight modification of the existing bundle adjustment implementations.

In summary, this paper presents extremely simple and easily visualizable techniques to find a robust solution of the bundle adjustment problem, especially an L_q solution. As far as the practicalities of the proposed methods is concerned, a simple approach and re-usability of the existing least squares implementation (with minor changes) makes the proposed algorithms a perfect candidate to replace a commonly used least squares bundle adjustment technique.

2. Theoretical Background

The Levenberg-Marquardt (LM) method [10, 11] is a numerical method for solving non-linear least squares problems. Let $c(\beta)$ be a vector valued function and $g(\beta) = c(\beta)^T c(\beta)$. The LM method minimizes a squared norm of a vector valued function

$$\min_{\beta} \; c(\beta)^T c(\beta) \; . \tag{1}$$

A current estimate of parameters β is updated in a direction δ such that $g(\beta + \delta) \leq g(\beta)$. This process is repeated until convergence. The resultant cost function $g(\cdot)$ is then,

$$g(\beta + \delta) = c(\beta + \delta)^T c(\beta + \delta) \; . \tag{2}$$

By using the first order approximation of c, we get

$$g(\beta + \delta) \approx \left(c(\beta) + \mathtt{J}_c \delta\right)^T \left(c(\beta) + \mathtt{J}_c \delta\right) \; , \tag{3}$$

where $\mathtt{J}_c$ is the Jacobian of vector $c(\beta)$. Assuming that the function is linear; by taking the derivative and equating to zero we get,

$$\mathtt{J}_c^T \mathtt{J}_c \delta = -\mathtt{J}_c^T c(\beta) \; , \tag{4}$$

for details see [6]. In the *Gauss-Newton* method, the value of δ is obtained as the solution to the above *normal equations*. On the other hand, in the case of the *Levenberg-Marquardt* method the normal equations of the Gauss-Newton method (4) are replaced with the *augmented normal equations* as,

$$(\mathtt{J}_c^T \mathtt{J}_c + \lambda \mathtt{I})\delta = -\mathtt{J}_c^T c(\beta) \; , \tag{5}$$

where λ is known as a *damping factor* that enables the LM algorithm to behave like both the steepest descent and Gauss-Newton methods and has a higher convergnce rate than both the algorithms. Due to its high convergence rate the LM algorithm has become a standard method for solving non-linear least squares problems.

3. Problem Formulation

Let $\mathbf{x}_i \in \mathbb{R}^2$ be image measurement vectors[1] approximating the true values $\bar{\mathbf{x}}_i$ and $\mathbf{X}$ be a vector obtained by concatenating all measurements: $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$, where N is the total number of measurements. The measurement vector $\mathbf{X}$ is assumed to be related by a non-linear function f to a parameter vector β. The mapping of parameter vector $\beta \in \mathbb{R}^M$ to measurement vectors $\hat{\mathbf{X}} \in \mathbb{R}^{2N}$, that is $\hat{\mathbf{X}} = f(\beta)$. We seek a parameter vector β for which the desired cost function C has minimum value, $\min_\beta \; C(\mathbf{X}, \beta)$. Typically, the LM method minimizes the squared norm of a vector function. Therefore, we can write the cost function C as inner product of two vectors as

$$\min_{\beta} \; \mathbf{E}^T \mathbf{E} \; , \tag{6}$$

where $\mathbf{E}$ is an error vector computed from the values of $\mathbf{X}$ and $f(\beta)$. Depending on the cost function to be minimized, there are several ways of computing the error vector $\mathbf{E}$. For example, when minimization of the least squares cost is desired then the vector $\mathbf{E}$ is taken as the difference of vectors $\mathbf{X}$ and $f(\beta)$, $\mathbf{E} = \mathbf{X} - f(\beta)$. Then from (6), the least squares cost function to be minimized is

$$C_2(\mathbf{X}, \beta) = \sum_i \|\mathbf{e}_i\|^2 = \sum_i d(\mathbf{x}_i, f_i(\beta))^2 \; , \tag{7}$$

where $d(\mathbf{x}_i, f_i(\beta))$ is the Euclidean distance between a measured value $\mathbf{x}_i$ and a predicted value $f_i(\beta)$; and $\mathbf{e}_i \in \mathbb{R}^2$ is computed as difference of the vectors $\mathbf{x}_i$ and $f_i(\beta)$, $\mathbf{e}_i = \mathbf{x}_i - f_i(\beta)$, while it is assumed that the measurement $\mathbf{x}_i$ is related by a function f_i to a parameter vector β. From here on we assume that $\mathbf{e}_i = \mathbf{x}_i - f_i(\beta)$.

In the rest of the paper we assume that the vector $\mathbf{E}$ is formed by concatenation of error vectors $\mathbf{e}_i$, where $\mathbf{e}_i$ is a vector representing the difference between a measured value $\mathbf{x}_i$ and a predicted value $f_i(\beta)$ and is computed as $\mathbf{e}_i = \mathbf{x}_i - f_i(\beta)$.

[1] In general, the theory applies to measurements in spaces of any dimension.

3.1. Solution Strategies:

In this section we discuss the techniques that are used to minimize a desired robust cost function.

• **Modified Error Vector:** One of the techniques to achieve the minimization of a desired cost function is to replace the error vector $\mathbf{E}$ in (6) with a modified error vector $\mathbf{E}'$. This new vector is obtained by modifying each component of $\mathbf{E}$. The resultant minimization function is,

$$\min_{\beta} \ (\mathbf{E}')^T(\mathbf{E}') = \min_{\beta} \ \sum_i \ \|\mathbf{e}'_i\|^2 \ . \tag{8}$$

Since the minimization function is still a squared norm of the modified error vector, the resultant cost function can be minimized by using the LM method.

• **Attenuation Factor:** Another strategy to minimize a desired cost function is to multiply each vector $\mathbf{e}_i$ by an *attenuation factor* α_i, such that the inner product of the modified error vectors $\mathbf{e}'_i = \alpha_i \mathbf{e}_i$ result in minimization of a desired function $\psi(\cdot)$. The minimization function takes the following form,

$$\min_{\beta} \sum_i \ \|\mathbf{e}'_i\|^2 = \min_{\beta} \sum_i \ \alpha_i^2 \|\mathbf{e}_i\|^2 \ , \tag{9}$$

where $\alpha_i^2 \|\mathbf{e}_i\|^2 = \psi(\|\mathbf{e}_i\|)$. Same as before, the above minimization problem can be solved by using the LM method. Furthermore, the value of the attenuation factor is computed as

$$\alpha_i = \psi(\|\mathbf{e}_i\|)^{1/2} / \|\mathbf{e}_i\| \ . \tag{10}$$

In this case, the value of the attenuation factor determines the impact of the corresponding error vector on the resultant cost function. Thus, the resultant function can be made robust to outliers by increasing the impact of inliers and decreasing the impact of outliers.

4. Existing Techniques

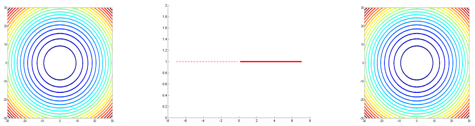
In this section we discuss some of the existing techniques to solve the bundle adjustment problem. We are particularly interested in the Squared Error function and the Huber function. This section briefly describes the Squared Error function. However, the Huber function is discussed in section 6.

4.1. L_2 or Squared Error Function (SE):

The Minimization of a squared distance function is a fairly standard and widely used technique. For some scalar input ϵ, the *Squared Error (SE)* function ψ_{SE} is,

$$\psi_{SE}(\epsilon) = \epsilon^2 \ . \tag{11}$$

An advantage of using the Squared Error function is that it is a continuous function and is at least twice differentiable; but unfortunately it suffers from a major drawback when it comes to robustness against outliers.



(a) SE Contours (b) SE Attenuation (c) SE Resultant Contours

Figure 1. *Squared Error Function (2D): (a) shows an isotropic contour plot of $\mathbf{e}'^T_i \mathbf{e}'_i / \alpha_i^2$ for two dimensional error vectors. A plot of the constant attenuation factor α_i is shown in (b). Isotropic contours and a constant attenuation factor indicates that both inlier and outlier data is treated equally, therefore the Squared Error function is not robust against outliers and the resultant function also has isotropic contours, as shown in (c).*

For some vector $\mathbf{e}_i$, the Squared Error function ψ_{SE} is $\psi_{SE}(\|\mathbf{e}_i\|) = \|\mathbf{e}_i\|^2$. The value of α_i from (10) is equal to 1. By replacing $\mathbf{e}_i$ with $\mathbf{e}'_i = \alpha_i \mathbf{e}_i$ in (6), the resultant cost function C_2, is

$$C_2(\mathbf{X}, \beta) = \sum_i \ \mathbf{e}'^T_i \mathbf{e}'_i = \sum_i \ \mathbf{e}_i^T \mathbf{e}_i \tag{12}$$

It can easily be seen that C_2 minimizes the sum of the squared Euclidean distances. Since the Squared Error function is a quadratic function, the effect of outliers on the overall cost function is magnified; as a consequence the resultant cost function is very sensitive to outliers.

For 2D vectors $\mathbf{e}_i$, the quadratic behavior of the Squared Error function is evident from the isotropic contours of $\mathbf{e}'^T_i \mathbf{e}'$, as shown in fig 1(c). A constant plot of the attenuation factor in fig 1(b) shows that the Squared Error function assigns equal weights to all the error vectors and does not distinguish between the inlier and outlier data. As a result, the resultant function is senstive to outliers.

5. Proposed L_q Cost Functions

This section presents three ways of minimizing an L_q cost function. We are particularly interested in the following two forms of the L_q cost functions: Firstly, an L_q cost function that minimizes the sum of the q-th power of error, that is $\min_\beta \sum_i \ \|\mathbf{e}_i\|^q$. Secondly, an L_1 cost function that minimizes the sum of the L_1 norms of error vectors, as $\min_\beta \sum_{i,j} \ e_i^j$. In order to distinguish both the functions we refer to the first function as the L_q *cost function* and the second function as the *Absolute Value function*. Below we discuss these techniques in detail.

5.1. L_q Cost Function:

The L_q cost function finds the minimum of, $C_{Lq} = \sum_i \|\mathbf{e}_i\|^q$, by associating an attenuation factor α_i with each corresponding error vector $\mathbf{e}_i$.

For some scalar input ϵ, the L_q function ψ_{Lq} is defined as

$$\psi_{Lq}(\epsilon) = |\epsilon|^q \ . \tag{13}$$

For large values of input the L_q function, for $1 \leq q < 2$, has a relatively smaller output than the Squared Error function. Thus, the resultant function is more robust to outliers than tha Squared Error function.

For some vector $\mathbf{e}_i$, the L_q function ψ_{Lq} is $\psi_{Lq}(\|\mathbf{e}_i\|) = \|\mathbf{e}_i\|^q$. The value of α_i from (10) is computed as

$$\alpha_i = \frac{\psi_{Lq}(\|\mathbf{e}_i\|)^{1/2}}{\|\mathbf{e}_i\|} = \frac{\|\mathbf{e}_i\|^{q/2}}{\|\mathbf{e}_i\|} = \|\mathbf{e}_i\|^{(q-2)/2} \ . \tag{14}$$

By replacing the value of $\mathbf{e}_i$ with $\mathbf{e}'_i = \alpha_i \mathbf{e}_i$ in (6), we get

$$\begin{aligned} C_2(\mathbf{X}, \beta) &= \sum_i \mathbf{e}'^T_i \mathbf{e}'_i = \sum_i \alpha_i^2 \mathbf{e}_i^T \mathbf{e}_i \\ &= \sum_i \|\mathbf{e}_i\|^{q-2} \|\mathbf{e}_i\|^2 = \sum_i \|\mathbf{e}_i\|^q \ . \end{aligned} \tag{15}$$

Thus, the proposed modification results in minimization of $C_{Lq}(\mathbf{X}, \beta)$ instead of $C_2(\mathbf{X}, \beta)$. This enables us to use the Levenberg-Marquardt method to minimize the L_q cost, that would otherwise require relatively complicated strategies to find a solution. It is evident from non-isotropic contours of the L_q cost function, in fig 2(c), that the L_q cost function is more robust to outliers than the Squared Error function and the primary reason for this robustness is the attenuation factor, as shown in fig 2(b).

5.2. L_q Optimization using Iterative Re-Weighted Least Square (IRLS):

The technique proposed in this section uses an *Iteratively Re-Weighted Least Squares (IRLS)* approach to solve for an L_q solution of the problem. The IRLS technique iteratively solves a weighted least squares cost to minimize a relatively robust cost function than the standard least squares cost. The weighting factor in this case has the same role as the attenuation factor in the case of the L_q cost function, that is to increase or decrease the influence of a particular error vector on the resultant function. Thus, the robustness of an IRLS cost function depends on the weight values used.

A general form of the IRLS objective function is

$$\min_\beta \sum_i w_i \|\mathbf{x}_i - f_i(\beta)\|^2 \ , \tag{16}$$

where w_i is a scalar value associated with its corresponding error vector $\mathbf{e}_i$. There can be several choices of how to compute weights. However, if we choose weights as $\|\mathbf{e}_i\|^{q-2}$ and perform iterations until convergence, then the IRLS technique will results in minimization of the L_q cost function, for details see [1]. On the other hand, if we choose different weights then the resultant function may be more robust to outliers than the Squared Error function or even the L_q function, but in that case the resultant function will no longer be the L_q cost function, possibly not even convex.

At iteration t, a current estimate of parameters β^t is updated as,

$$\beta^{t+1} = \underset{\beta}{\operatorname{argmin}} \sum_i w_i^t \|\mathbf{x}_i - f_i(\beta)\|^2 \ , \tag{17}$$

where w_i^t is some positive weighting term and is computed as $w_i^t = \|\mathbf{x}_i - f_i(\beta^t)\|^{q-2}$.

Since the Levenberg-Marquardt method is a numerical optimization algorithm, a modified solution of (6) at iteration t is computed as

$$C_{IRLS}(\mathbf{X}, \beta) = \sum_i w_i^t \|\mathbf{e}_i^t\|^2 \ , \tag{18}$$

where $\mathbf{e}_i^t = \mathbf{x}_i - f_i(\beta^t)$ and

$$w_i^t = \|\mathbf{e}_i^t\|^{q-2} \ . \tag{19}$$

In this case, it is obvious that C_{IRLS} is equivalent to C_q. A contour plot of the resultant function, in fig 2(f), shows that the IRLS function has the same contours as the L_q cost function. Furthermore, weights in this case play the same role as the attenuation factors in the case of the L_q cost function, which is to reduce the effect of outliers.

5.3. Absolute Value Function (AV):

The Absolute Value function minimizes the sum of the L_1 norms of error vectors. The desired minimization function is $\min_\beta \sum_i \|e_i\|_1$, where $\| \cdot \|_1$ is the L_1 norm.

For some scalar input ϵ the Absolute Value function f_{AV} is defined as

$$f_{AV}(\epsilon) = \begin{cases} \sqrt{\epsilon} & \text{if } \epsilon \geq 0 \\ -\sqrt{-\epsilon} & \text{otherwise} \end{cases} \ . \tag{20}$$

Since, we are only interested in the squared value of functions, the squared Absolute Value function ψ_{AV} for some scalar input ϵ is,

$$\psi_{AV}(\epsilon) = f_{AV}(\epsilon)^2 = |\epsilon| \ . \tag{21}$$

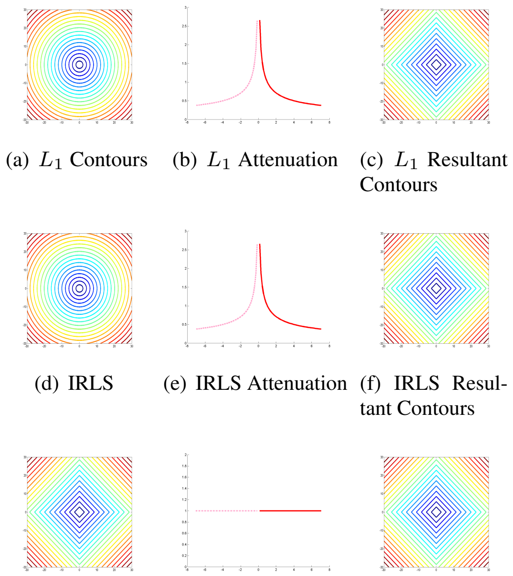
Let $\mathbf{e}'$ be the modified error vector obtained by applying the Absolute Value function f_{AV} on each component $\mathbf{e}$ as,

$$\mathbf{e}' = \left(f_{AV}(e^1), f_{AV}(e^2), \ldots, f_{AV}(e^n)\right)^T \ , \tag{22}$$

where e^j is the j-th component of $\mathbf{e}$.

By replacing $\mathbf{e}_i$ with $\mathbf{e}'_i$ in (6), we get

$$C_2(\mathbf{X}, \beta) = \sum_i \mathbf{e}'^T_i \mathbf{e}'_i = \sum_{i,j} |e_i^j| = \sum_i \|\mathbf{e}_i\|_1 \ .$$



(a) L_1 Contours (b) L_1 Attenuation (c) L_1 Resultant Contours

(d) IRLS (e) IRLS Attenuation (f) IRLS Resultant Contours

(g) AV Contours (h) AV Attenuation (i) AV Resultant Contours

Figure 2. *Proposed L_q Functions (2D): The above figure shows plots of the L_1, IRLS and Absolute Value functions. The left column of the above figure shows contour plots of ${\mathbf{e}'}_i^T{\mathbf{e}'}_i/\alpha_i^2$ for two dimensional error vectors. The middle column shows plots of attenuation factors α_i. The right column plots ${\mathbf{e}'}_i^T{\mathbf{e}'}_i$, that is contour plots obtained by multiplying the left and middle columns. In the case of the L_1 and IRLS functions, robustness against outliers is achieved through the attenuation factor. The bell shaped curve of the attenuation factor indicates that the effect of inliers is magnified and the effect of outliers is reduced on the resultant function. However, in case of the Absolute Value function, non-isotropic contours indicates that the robustness against outliers is encoded in the function itself and is not achieved through the attenuation factor.*

Since the above equation minimizes a squared error function, the Absolute Value function can be minimized by using the LM method. Furthermore, the non-isotropic contours of the Absolute Value function, in fig 2(g), show that unlike the previously proposed techniques where attenuation factor played a major role in achieving robustness against outliers, in this case the robustness against outliers is encoded in the function itself.

6. Proposed Huber Cost Functions

In this section we discuss the Huber function and a standard way of applying the Huber function to vector inputs, followed by the proposed Isotropic Huber function and the re-thresholded Huber function.

6.1. Huber Function:

In robust statistics, a very popular function to minimize is the *Huber* function [9]. Given a threshold value b, the Huber function $h(\cdot)$ for a scalar input ϵ is,

$$h(\epsilon) = \begin{cases} \epsilon & \text{if } |\epsilon| \leq b \\ \sqrt{(2\epsilon - b)b} & \text{if } \epsilon > b \\ -\sqrt{-(2\epsilon + b)b} & \text{if } \epsilon < -b \end{cases} . \quad (23)$$

Since we are only interested in a squared function, the squared Huber function $\psi_H(\cdot)$ is,

$$\psi_H(\epsilon) = \begin{cases} \epsilon^2 & \text{if } |\epsilon| < b \\ 2b|\epsilon| - b^2 & \text{otherwise} \end{cases} . \quad (24)$$

The Huber function shows a hybrid behavior depending on the value of the threshold b; for inliers ($|\epsilon| < b$) the Huber function has a quadratic behavior, whereas for outliers ($|\epsilon| \geq b$) the Huber function has a linear behavior. This in turn makes the Huber function robust to outliers and differentiable at the minimum.

6.2. Standard Huber Function

A standard way of applying the Huber function on some vector valued input $\mathbf{e}_i$, is to apply the Huber function $h(\cdot)$ on each component e_i^j of the vector as,

$${\mathbf{e}'}_i = \left(h(e_i^1), h(e_i^2), \ldots, h(e_i^n)\right)^T . \quad (25)$$

By replacing the value of $\mathbf{e}_i$ with ${\mathbf{e}'}_i$ in (6) we get

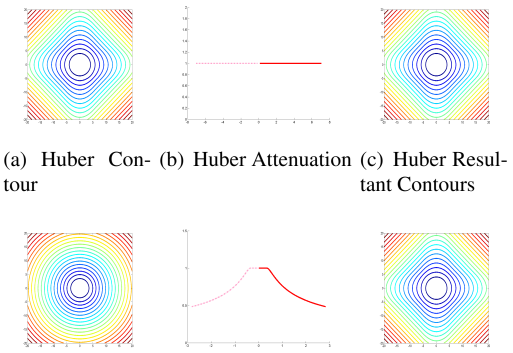
$$C_2(\mathbf{X}, \beta) = \sum_i {\mathbf{e}'}_i^T {\mathbf{e}'}_i = \sum_{i,j} \psi_H(e_i^j) \quad (26)$$

Thus, the minimization of C_2 results in the minimization of a hybrid function that adapts the minimization strategy based on each component of error vectors.

A contour plot of ${\mathbf{e}'}_i^T{\mathbf{e}'}_i$ for a two dimensional input vector is shown in fig 3. The non-isotropic contours in fig 3(a) show that the resultant function adapts its behavior depending on the value of individual components of the error vector. Thus, instead of treating a whole point $\mathbf{e}_i$ as an outlier or inlier, the influence of the higher components of the error vector is mitigated to increase the robustness of the cost function to outliers.

6.3. Proposed Isotropic Huber (IsoH) Function

In this section we propose a slightly different application strategy of the Huber function for vector valued inputs, we refer to it as *Isotropic Huber* function because of its isotropic contours. Instead of changing the behavior of the Huber function on the basis of each component of the error vectors as mentioned before, the Isotropic Huber function adapts its behavior on the basis of the magnitudes of the error vectors, that is $\|\mathbf{e}_i\|$. Thus, the resultant cost function



(a) Huber Contour (b) Huber Attenuation (c) Huber Resultant Contours

(d) IsoH Contours (e) IsoH Attenuation (f) IsoH Resultant Contours

Figure 3. *Huber Functions (2D): The above figure shows plots of the Huber and Isotropic Huber functions. The left column of the above figure shows contour plots of $\mathbf{e'}_i^T \mathbf{e'}_i/\alpha_i^2$ for two dimensional error vectors. The middle column shows plots of attenuation factors α_i. The right column plots $\mathbf{e'}_i^T \mathbf{e'}_i$, that is contour plots obtained by multiplying the left and middle columns. The non-isotropic contours and a constant attenuation factor of the Huber function indicates that the robustness to outliers is encoded in the function itself. On the other hand, the isotropic contours and a bell shaped curve of attenuation factor of the Isotropic Huber function indicates that the robustness against outlier is achieved through the attenuation factor.*

deals with outliers in a more intuitive way than the Huber cost function C_H, described earlier.

For some input vector $\mathbf{e}_i$, the Isotropic Huber function $\psi_{IsoH}(\cdot)$, is $\psi_{IsoH}(\|\mathbf{e}_i\|) = \psi_H(\|\mathbf{e}_i\|)$. The value of α_i from (10) is computed as,

$$\alpha_i = \frac{\psi_H(\|\mathbf{e}_i\|)^{1/2}}{\|\mathbf{e}_i\|} \,. \tag{27}$$

Finally, a modified error vector is computed as $\mathbf{e'}_i = \alpha_i \mathbf{e}_i$. By substituting the value of $\mathbf{e'}_i$ in (6) we get

$$C_2(\mathbf{X}, \beta) = \sum_i \alpha_i^2 \mathbf{e}_i^T \mathbf{e}_i = \sum_i \psi_H(\|\mathbf{e}_i\|) \,. \tag{28}$$

For two dimensional input vectors, a contour plot of $\mathbf{e'}_i^T \mathbf{e'}_i$ is shown in fig 3(f). The bell shaped attenuation curve (in fig 3(e)) together with the isotropic contours (in fig 3(d)) makes the resultant function $\mathbf{e'}_i^T \mathbf{e'}_i$ robust to outliers by assigning less weight to outliers, as shown in fig 3(f).

6.4. Re-Thresholded Huber Function:

Here, we take advantage of the iterative behavior of the LM method and update, specifically reduce, the threshold value of the Huber function after some fixed number of iteration. Therefore, it is referred as the *Re-Thresholded Huber* function. Given an initial threshold value, the Re-Thresholded Huber function solves for (28) using the LM method and after some iterations the threshold value is reduced by some factor. This way, after some number of iterations the Huber function will have a dominant linear behavior and consequently increased robustness.

7. Experiments

We apply the proposed algorithms to a subset (20 views) of the popular and publicly available NotreDame dataset. Our experiments are focused on the following aspects of the proposed algorithms: The first experiment shows the convergence behavior of the proposed algorithms on real data. The second experiment is designed to show the robustness of the proposed algorithms to different percentage of outliersin the data set. The last experiment shows the convergence behavior of the L_1-bundle adjustment from different but sufficiently close starting points.

In our experiments we assume that the intrinsic camera parameters are known. Therefore, we only solve for rotations and translations of cameras, and 3D structure of the scene. Furthermore, errors are computed against know ground truth values. Note that the error curves shown in the figures are not monotonically decreasing because the errors reported are different than the error minimized by the bundle adjustment algorithms, that is the re-projection error. Furthermore, the plots shown in the figures are cropped to enhance the visibility.

Starting Point: Since the primary purpose of this paper is not on how to generate a good starting point for bundle adjustment, we add noise to the known ground truth parameters and use these parameters as a starting point.

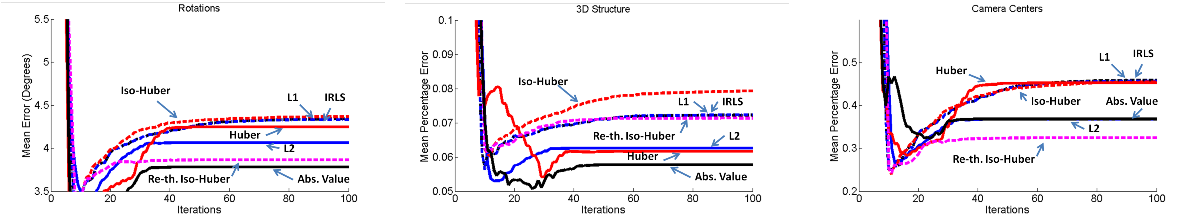
- *Rotation parameters:* Each camera rotation is perturbed by a maximum angle of 5°, by creating a rotation matrix from a random vector (maximum magnitude 5°) and then multiplying with the ground truth rotation.

- *3D points and Camera centers:* Each component of $\mathbf{X}$ (3D point) and $\mathbf{C}$ (camera center) is scaled by a maximum factor of 0.2.

Outliers: For $n\%$ of outliers, a Gaussian noise of zero mean and standard deviation $\sigma = 25$ is added to $n\%$ of the projections of every 3D point.

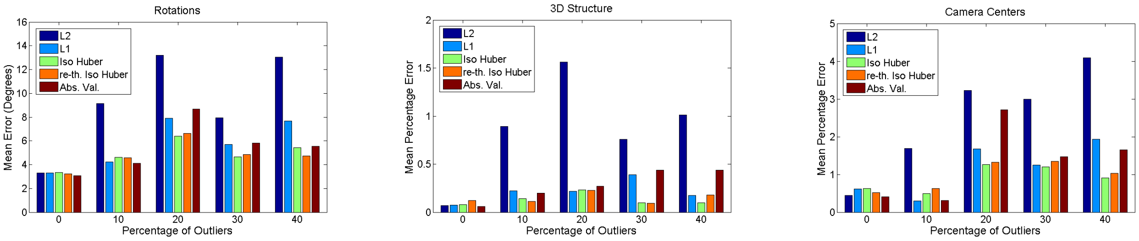
Error Measures: All errors are computed with respect to the known ground truth as follows:

- *Error in Rotations:* Let $\{\mathtt{R}_i\}$ be a set of estimated rotation matrices and $\{\hat{\mathtt{R}}_i\}$ be the corresponding set of ground truth rotation matrices. Let $\mathtt{T}$ be the L_2 mean [5] of the rotation matrices $\mathtt{R}_i \hat{\mathtt{R}}_i^T$. Then, error in degrees for each rotation matrix is computed by finding the difference in angle between the ground truth rotation $\hat{\mathtt{R}}_i$ and the transformed rotation $\mathtt{R}_i^{'} = \mathtt{T}\mathtt{R}_i$. Finally, the mean of errors is reported.



(a) Mean Error Rotations (Degrees) (b) Mean Percentage Error 3D Points (c) Mean Percentage Error Camera Centers

Figure 4. *Convergence Behavior: The above figures show the convergence behavior of the proposed algorithms on real data, with no outliers added explicitly. The above plots show that the Absolute Value and re-thresholded Isotropic Huber techniques give better results than the rest of the techniques. Thus, whenever there are not many outliers in the dataset then the Absolute value and re-thresholded Isotropic Huber techniques are recommended.*



(a) Mean Error Rotations (Degrees) (b) Mean Percentage Error 3D Points (c) Mean Percentage Error Camera Centers

Figure 5. *Robustness to Outliers: The above figures show the results of the proposed methods in the presence of different number of outliers in the dataset. In this case the percentage of outliers is varied from* 0% *to* 40% *with an increment of* 10%*. The results have shown that the re-thresholded Isotropic Huber technique has the least error, except when there are no outliers in the dataset. Furthermore, as expected, the results of the* L_2 *method are far from the ground truth than the rest of the techniques. Therefore, in the presence of outliers the re-thresholded Isotropic Huber technique is recommended because of its superior results and adaptive threshold values.*

- *Error in Camera Centers and* $3D$ *points:* We report a mean percentage error between true and estimated 3D point clouds. Let $\{\mathbf{x}_i\}$ be a set of k ground truth points and $\{\mathbf{y}_i\}$ be a set of estimated points. Let $\mathtt{T}$ be the transformation between the points clouds, computed using the Horn's method [8]. The mean percentage error e between points is computed as,

$$e = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{\mathrm{abs}(\|\mathbf{x}_{\mathrm{i}}\| - \|\mathtt{T}\mathbf{y}_{\mathrm{i}}\|)}{\|\mathbf{x}_i\|}\times 100\right) ,$$

The same technique is used for computing errors in camera centers.

In the rest of the section we will discuss our experimental results.

7.1. Convergence Behavior

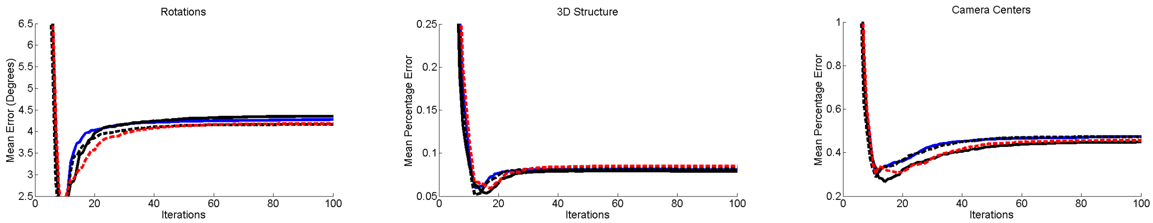
This experiment demonstrates the convergence behavior of the proposed algorithms on real data, with no outliers added explicitly. It evident from fig 4 that the results of the Absolute Value and re-thresholded Isotropic Huber techniques are closer to the ground truth than the rest of the techniques. Thus, whenever there are not many outliers in the dataset then the Absolute value and re-thresholded Isotropic Huber techniques are recommended. Since the LM method is a numerical optimization method, the results of the L_1 and IRLS methods are the same.

7.2. Robustness to Outliers

This experiment demonstrates the robustness of the proposed algorithms against outliers. Therefore, we add different percentage of outliers to the dataset ranging from 0% to 40% with an increment of 10%. We are particularly interested in the L_1, L_2, Absolute Value, Isotropic Huber and re-thresholded Isotropic algorithms. As expected, the results of the L_2 algorithm are far from the ground truth values than the rest of the techniques and the parameter estimates of the re-thresholded Isotropic Huber algorithm are closer to ground truth than the other proposed algorithms, as shown in fig 5.

7.3. Convergence from Different Starting Points

This experiment shows that the convergence behavior of the L_1-bundle adjustment does not change much with a small change in the starting point of the algorithm. In order to generate different starting points for our algorithm, in addition to the basic noise in parameters as defined before, we further perturb the rotations by 1°, and camera center and 3D points by a factor of 0.05. Each colored line in fig 6



(a) Mean Error Rotations (Degrees) (b) Mean Percentage Error 3D Points (c) Mean Percentage Error Camera Centers

Figure 6. *Convergence from Different Starting Points (L_1-bundle adjustment): The above figures show the convergence behavior of the L_1 method from different starting points. Each colored line represents a different starting point of the algorithm. Results have shown that the convergence behavior of the L_1 method does not change much with a change in the starting points of the algorithm and that the parameters recovered are generally close to each other.*

indicates the convergence of the L_1 technique from a different starting point. These plots have shown that the results of the L_1-bundle adjustment will not change drastically with a small change in initial parameter estimates.

8. Conclusion

In summary, we proposed several techniques to solve for a robust solution of a non-linear parameter estimation problem, namely the bundle adjustment problem. The proposed techniques minimize L_q cost functions, for $1 \leq q < 2$, and the Huber cost function. An advantage of the proposed techniques is that an existing implementation of the L_2 bundle adjustment can be modified to minimize a desired cost function because the proposed techniques still use the LM method to find a robust solution. Furthermore, our experimental results on the NotreDame set showed that the proposed techniques are more robust to outliers than the L_2 bundle adjustment. A Simple approach and easy implementation makes the proposed algorithms practically feasible.

The applicability of proposed techniques is not only limited to bundle adjustment problem but a wide class of non-linear parameter estimation problems, that use the Levenberg-Marquardt method for minimization, can be solved robustly using these techniques.

References

[1] K. Aftab, R. Hartley, and J. Trumpf. Generalized Weiszfeld algorithm for Lq optimization. to appear in PAMI. 4

[2] S. Agarwal, N. Snavely, S. Seitz, and R. Szeliski. Bundle adjustment in the large. In *Computer Vision ECCV 2010*. Springer, 2010. 2

[3] C. Engels, H. Stewnius, and D. Nistr. Bundle adjustment rules. In *In Photogrammetric Computer Vision*, 2006. 1

[4] A. Eudes and M. Lhuillier. Error propagations for local bundle adjustment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2

[5] R. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 6

[6] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision – 2nd Edition*. Cambridge University Press, 2004. 1, 2

[7] S. Holmes, G. Sibley, G. Klein, and D. Murray. A relative frame representation for fixed-time bundle adjustment in sfm. In *IEEE International Conference on Robotics and Automation*, 2009. 2

[8] B. K. P. Horn, H. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOURNAL OF THE OPTICAL SOCIETY AMERICA*, 1988. 7

[9] P. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 1964. 5

[10] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 1944. 2

[11] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Indust. Appl. Math.*, 1963. 2

[12] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006. 2

[13] H. Shum, Q. Ke, and Z. Zhang. Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999. 2

[14] D. Steedly and I. Essa. Propagation of innovative information in non-linear least-squares structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, 2001. 2

[15] D. Steedly, I. Essa, and F. Dellaert. Spectral partitioning for structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003. 2

[16] D. Strelow. General and nested wiberg minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 2

[17] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *ICCV '99: Proceedings of the International Workshop on Vision Algorithms*. Springer-Verlag, 2000. 1