Bi-objective Optimization for Robust RGB-D Visual Odometry

Tao Han[1], Chao Xu[1], Ryan Loxton[2], Lei Xie[1]

1. State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems & Control, Zhejiang University, Hangzhou 310027, China
E-mail: thancn@gmail.com, cxu@zju.edu.cn, leix@iipc.zju.edu.cn

2. Department of Mathematics & Statistics, Curtin University, Perth 6102, Australia
E-mail: r.loxton@curtin.edu.au

Abstract: This paper considers a new bi-objective optimization formulation for robust RGB-D visual odometry. We investigate two methods for solving the proposed bi-objective optimization problem: the weighted sum method (in which the objective functions are combined into a single objective function) and the bounded objective method (in which one of the objective functions is optimized and the value of the other objective function is bounded via a constraint). Our experimental results for the open source TUM RGB-D dataset show that the new bi-objective optimization formulation is superior to several existing RGB-D odometry methods. In particular, the new formulation yields more accurate motion estimates and is more robust when textural or structural features in the image sequence are lacking.

Key Words: Bi-objective Optimization, Visual Odometry, Motion Estimation, Robotics

1 INTRODUCTION

Visual odometry is an important area of information fusion in which the central aim is to estimate the pose of a robot using data collected by visual sensors [1]. Because nearly all robotic tasks require knowledge of the pose of the robot, visual odometry plays a critical role in robot control, simultaneous localization and mapping (SLAM) and robot navigation, especially when external reference information about the environment (such as GPS data) is unavailable. Visual odometry can be viewed as a particular instance of the general pose tracking problem, which is the most fundamental perception problem in robotics [2].

To date, a variety of different visual odometry methods based on different sensor information have been studied and widely implemented. One of the most well-known methods is the iterative closest point (ICP) algorithm [3], which estimates the robot's pose by minimizing the distance between corresponding points in two laser scanning snapshots. However, this method can easily become trapped in local optima if a good initial guess is not provided. In addition to the ICP algorithm and its variants, odometry methods using camera images have also been studied [4] [5]. Such methods usually extract point features from the camera images and match them through a series of steps, including descriptor matching, RANSAC and bundle adjustment. Due to their expensive computational burden, these approaches are usually too slow for real-time application. One way of improving computational efficiency is to use sparse point features, but this approach does not fully exploit the available image data, ignoring much relevant information.

Recently, with RGB-D cameras becoming smaller and cheaper, the opportunity has arisen to develop RGB-D odometry methods that exploit both intensity and depth information. One such method was proposed by the Computer Vision Group at the Technical University of Munich (TUM). In this method, a single-objective optimization problem is formulated to penalize the intensity difference between corresponding pixels in consecutive images [6] [7]. This method can be implemented in real-time even on a single-core CPU. However, the image depth information is only used to determine the relationship between corresponding pixels in consecutive images for intensity residual comparison; depth residuals are not considered. Thus, a new bi-objective optimization problem was subsequently proposed in [8] to minimize both depth and intensity residuals, with the aim of improving estimation performance.

In this paper, we consider the same bi-objective optimization formulation as in [8]. Our aims are twofold: (i) to propose new computational approaches for solving this bi-objective optimization formulation; and (ii) to explore and quantify the advantages of the bi-objective optimization formulation for improving estimation robustness. The first computational approach we investigate, the so-called weighted sum method, involves integrating the two objective functions into a single objective using a weighting factor. We derive a new formula for adaptive calculation of this weighting factor, which is crucial to estimation accuracy. Our formula is based on a novel *image complexity metric* and differs from the corresponding formula in [8], which uses the ratio of median intensity and median depth values to calculate the weighting factor. The second computational approach we investigate, the so-called bounded objective method, involves optimizing one of the objective functions while the other objective function is bounded

This work was supported by the National High Technology Research and Development Program of China (863 Program) (2012AA041701) and the National Natural Science Foundation of China (61104048, 61473253).

via a constraint. Again, our new *image complexity metric* is used, this time to determine an appropriate objective bound. To evaluate performance, the open source TUM RGB-D dataset [9] was used. The computational results demonstrate that our new methods generally give results of superior accuracy compared with the methods in [6] [7] [8].

2 SINGLE-OBJECTIVE OPTIMIZATION FOR VISUAL ODOMETRY

The camera motion in 3-D space has six degrees of freedom and can be denoted as a 6-D vector in a manifold:

$$\boldsymbol{\xi} = [\nu_1\ \nu_2\ \nu_3\ \psi_1\ \psi_2\ \psi_3]^\top ,$$

where ν_1, ν_2, ν_3 are the translation components of the motion (which form a Euclidean space) and ψ_1, ψ_2, ψ_3 are the rotation components of the motion (which span over the non-Euclidean 3-D rotation group *SO(3)*). To estimate $\boldsymbol{\xi}$, we consider a world point ρ_i and assume that its brightness stays the same in two consecutive images. This is the so-called photo-consistency assumption [7], which can be expressed mathematically by

$$I_1(\boldsymbol{x}_i) = I_2(\boldsymbol{y}_i(\boldsymbol{\xi}^*)),$$

where $\boldsymbol{x}_i \in \mathbb{R}^2$ represents the mapping coordinate of the world point ρ_i in the first image and $\boldsymbol{y}_i(\boldsymbol{\xi}^*) \in \mathbb{R}^2$ represents the corresponding coordinate of ρ_i in the second image when given the true value of the camera motion $\boldsymbol{\xi}^*$. Moreover, $I_1(\cdot)$ and $I_2(\cdot)$ are the brightness (or intensity) values of the specified coordinates in the first and second images, respectively.

Based on the photo-consistency assumption, we can define the intensity difference corresponding to the motion estimate $\boldsymbol{\xi}$ as

$$r_I^{(i)}(\boldsymbol{\xi}) = I_2(\boldsymbol{y}_i(\boldsymbol{\xi})) - I_1(\boldsymbol{x}_i).$$

According to the results in [7], the more accurate the camera motion estimate, the smaller the residual $r_I^{(i)}(\boldsymbol{\xi})$. Thus, estimation quality in visual odometry can be assessed by considering the following least-squares objective function, which is the sum of residual squares for n world points:

$$F_I(\boldsymbol{\xi}) = \sum_{i=1}^{n} \left\{ r_I^{(i)}(\boldsymbol{\xi}) \right\}^2 .$$

Then the problem of determining the camera motion can be formulated as a least-squares optimization problem, i.e.,

$$\underset{\boldsymbol{\xi}}{\text{minimize}}\ F_I(\boldsymbol{\xi}).$$

To improve robustness, weighted residuals can be used to reduce the effect of noise and outliers in the image data. This motivates the following weighted objective function in quadratic form as in [7]:

$$F_I(\boldsymbol{\xi}) = [\boldsymbol{r}_I(\boldsymbol{\xi})]^\top \boldsymbol{\Omega}_I [\boldsymbol{r}_I(\boldsymbol{\xi})] , \qquad (1)$$

where

$$\boldsymbol{r}_I(\boldsymbol{\xi}) = \left[r_I^{(1)}(\boldsymbol{\xi})\ r_I^{(2)}(\boldsymbol{\xi})\ \cdots\ r_I^{(n)}(\boldsymbol{\xi}) \right]^\top ,$$

and $\boldsymbol{\Omega}_I$ is a diagonal weight matrix.

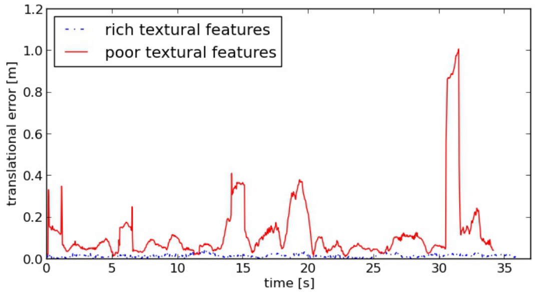


Figure 1: Motion estimation accuracy of the single-objective Gauss-Newton method for the TUM RGB-D dataset.

3 BI-OBJECTIVE OPTIMIZATION FOR RGB-D ODOMETRY

Traditional cameras only provide image intensity information. RGB-D cameras, on the other hand, provide image intensity and image depth information, both of which can be used for visual odometry. For example, in the odometry methods introduced by the TUM Computer Vision Group [6] [7], the relationship between corresponding pixels in consecutive images is expressed in terms of the depth information in the first image, and the intensity information of both images is used to define the motion estimation residuals as in Section 2. More precisely, the relationship between corresponding pixels in consecutive images is defined by a warping function as follows:

$$\boldsymbol{y}_i(\boldsymbol{\xi}) = \boldsymbol{\tau}(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i)),$$

where $D_1(\boldsymbol{x}_i)$ is the depth value of the pixel in the first image and $\boldsymbol{\tau}(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))$ is the warping function for calculating the mapping coordinate $\boldsymbol{y}_i$ in the second image. For the specific form of the warping function $\boldsymbol{\tau}(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))$, we refer the reader to [7].

Although single-objective optimization-based odometry methods are computationally fast and effective, they can produce poor results in some situations. For example, when textural features in the image sequence are poor, trajectory estimation accuracy will decrease dramatically. This is because the objective function $F_I(\boldsymbol{\xi})$ only depends on image intensity information, and thus it can become nonconvex when image textural features are lacking. In this case, the optimal motion estimates obtained by applying an optimization iterative procedure may only be locally optimal. To investigate this hypothesis, we applied the single-objective optimization approach (implemented using the Gauss-Newton method) to image sequences 2 and 4 from the *Structure vs. Texture* category in the TUM RGB-D dataset [9]. Our results are shown in Figure 1, where the solid line shows estimation error for image sequence 2 (poor texture) and the dashed line shows estimation error for image sequence 4 (rich texture). From the results, we see that the translation error of the motion estimates increases significantly when textural features are lacking. This motivates the new bi-objective optimization formulation proposed in [8], in which both image intensity and image depth residuals are minimized to improve robustness.

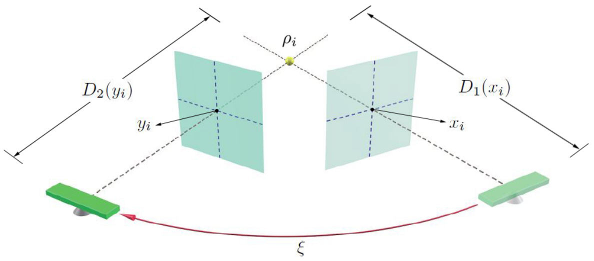


Figure 2: Motion estimation via RGB-D odometry: ρ_i is the world point under consideration, x_i and y_i are the pixels corresponding to ρ_i, and $D_1(x_i)$ and $D_2(y_i)$ are the depth values corresponding to ρ_i.

The extension of RGB-D odometry using bi-objective optimization is inspired by the ICP algorithm and its variants, which estimate the sensor motion by minimizing residual coordinate differences, instead of image intensity values. Since RGB-D cameras can provide both intensity and depth information simultaneously, we want to take full advantage of this feature by comparing depth differences, just as the ICP algorithm compares coordinate differences. Thus, we now consider two residuals instead of one:

$$\begin{cases} r_I^{(i)}(\boldsymbol{\xi}) = I_2(\boldsymbol{\tau}(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))) - I_1(\boldsymbol{x}_i), \\ r_D^{(i)}(\boldsymbol{\xi}) = D_2(\boldsymbol{\tau}(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))) \\ \qquad\qquad - [T(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))]_z, \end{cases} \tag{2}$$

where $D_1(\cdot)$ and $D_2(\cdot)$ are the depth values of the specified coordinates in the first and second images, and $T(\boldsymbol{\xi}, \boldsymbol{x}_i, D_1(\boldsymbol{x}_i))$ projects the 3-D coordinate of world point ρ_i from the first camera coordinate system to the second camera coordinate system based on the homogeneous transformation matrix for $\boldsymbol{\xi}$. Operator $[\ ]_z$ selects the coordinate value along the z-direction. See the diagram in Figure 2 for an explanation of the notation.

Based on $r_D^{(i)}(\boldsymbol{\xi})$ defined in (2), we consider the following objective function similar to (1):

$$F_D(\boldsymbol{\xi}) = [\boldsymbol{r}_D(\boldsymbol{\xi})]^\top \boldsymbol{\Omega}_D \, [\boldsymbol{r}_D(\boldsymbol{\xi})], \tag{3}$$

where

$$\boldsymbol{r}_D(\boldsymbol{\xi}) = \left[r_D^{(1)}(\boldsymbol{\xi}) \; r_D^{(2)}(\boldsymbol{\xi}) \; \cdots \; r_D^{(n)}(\boldsymbol{\xi}) \right]^\top,$$

and $\boldsymbol{\Omega}_D$ is a diagonal weight matrix.

Combining objectives (1) and (3), we consider the following bi-objective optimization problem:

$$\underset{\boldsymbol{\xi}}{\text{minimize}} \;\; \boldsymbol{F}(\boldsymbol{\xi}) = [F_I(\boldsymbol{\xi}), F_D(\boldsymbol{\xi})]^\top. \tag{4}$$

3.1 Weighted Sum Method

The weighted sum method is the most common approach to solving multi-objective optimization problems. In this method, the individual objective functions are assigned different weights and then added together to form a single objective function. More specifically, for individual objective functions $\Psi_1, \Psi_2, \ldots, \Psi_n$ and decision vector $\boldsymbol{\alpha}$, the combined objective function is

$$\Psi(\boldsymbol{\alpha}) = \sum_{i=1}^{q} \omega_i \Psi_i(\boldsymbol{\alpha}), \tag{5}$$

where ω_i are the weights. If all of the weights are positive, then the minimum of (5) is Pareto optimal for the original multi-objective problem [10].

In essence, the objective weights provide additional degrees of freedom in the optimization problem. For our odometry problem (4), the new single-objective optimization problem is defined as

$$\underset{\boldsymbol{\xi}}{\text{minimize}} \;\; F(\boldsymbol{\xi}) = \omega_I F_I(\boldsymbol{\xi}) + \omega_D F_D(\boldsymbol{\xi}).$$

Notice that by dividing $F(\boldsymbol{\xi})$ by ω_I, we can obtain an equivalent optimization problem as follows:

$$\underset{\boldsymbol{\xi}}{\text{minimize}} \;\; \bar{F}(\boldsymbol{\xi}) = F_I(\boldsymbol{\xi}) + \lambda F_D(\boldsymbol{\xi}), \tag{6}$$

where $\lambda = \omega_D / \omega_I$. Thus, we only need to consider a single weighting factor λ.

Problem (6) can be solved using the Gauss-Newton method. To do this, we linearize the residuals $\boldsymbol{r}_I(\boldsymbol{\xi})$ and $\boldsymbol{r}_D(\boldsymbol{\xi})$ using the Taylor expansion proposed in [11]:

$$\begin{cases} \boldsymbol{r}_I(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi}) \simeq \boldsymbol{r}_I(\boldsymbol{\xi}) + \boldsymbol{J}_I(\boldsymbol{\xi})\Delta\boldsymbol{\xi}, \\ \boldsymbol{r}_D(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi}) \simeq \boldsymbol{r}_D(\boldsymbol{\xi}) + \boldsymbol{J}_D(\boldsymbol{\xi})\Delta\boldsymbol{\xi}, \end{cases}$$

where the operator $\oplus$ maps a local variation $\Delta\boldsymbol{\xi}$ in the Euclidean space to a variation on the manifold, $\Delta\boldsymbol{\xi} \mapsto \boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi}$ (for more details, see [12]); and $\boldsymbol{J}_I(\boldsymbol{\xi})$ and $\boldsymbol{J}_D(\boldsymbol{\xi})$ are the Jacobians defined by

$$\boldsymbol{J}_I(\boldsymbol{\xi}) = \left. \frac{\partial \boldsymbol{r}_I(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi})}{\partial \Delta\boldsymbol{\xi}} \right|_{\Delta\boldsymbol{\xi}=0},$$
$$\boldsymbol{J}_D(\boldsymbol{\xi}) = \left. \frac{\partial \boldsymbol{r}_D(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi})}{\partial \Delta\boldsymbol{\xi}} \right|_{\Delta\boldsymbol{\xi}=0}.$$

Then the objective function in (6) can be approximated by a quadratic function of $\Delta\boldsymbol{\xi}$:

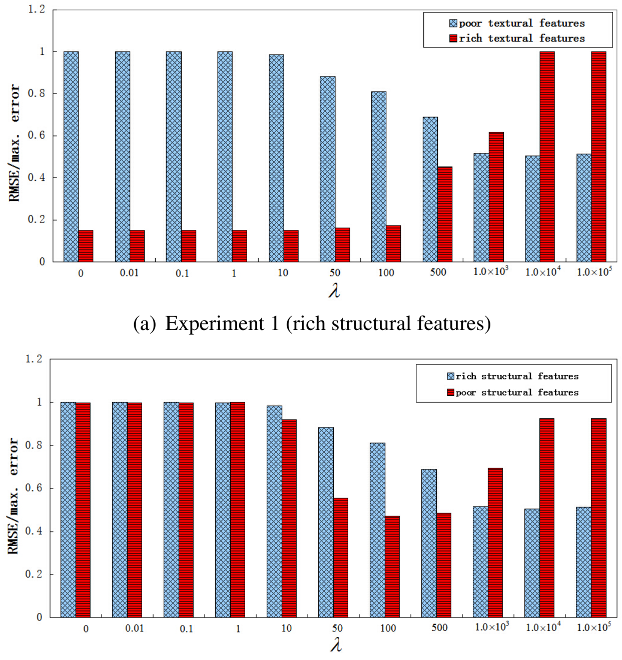
$$\begin{aligned} \bar{F}(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi}) \simeq (a_I + \lambda a_D) + 2(\boldsymbol{b}_I^\top + \lambda \boldsymbol{b}_D^\top)\Delta\boldsymbol{\xi} \\ + \Delta\boldsymbol{\xi}^\top(\boldsymbol{H}_I + \lambda \boldsymbol{H}_D)\Delta\boldsymbol{\xi}, \end{aligned} \tag{7}$$

where $a_j = [\boldsymbol{r}_j(\boldsymbol{\xi})]^\top \boldsymbol{\Omega}_j \boldsymbol{r}_j(\boldsymbol{\xi})$, $\boldsymbol{b}_j = [\boldsymbol{J}_j(\boldsymbol{\xi})]^\top \boldsymbol{\Omega}_j \boldsymbol{r}_j(\boldsymbol{\xi})$ and $\boldsymbol{H}_j = [\boldsymbol{J}_j(\boldsymbol{\xi})]^\top \boldsymbol{\Omega}_j \boldsymbol{J}_j(\boldsymbol{\xi})$ $(j = I, D)$.

Suppose that at iteration k, we have the motion estimate $\boldsymbol{\xi}^k$. Then the increment $\Delta\boldsymbol{\xi}^k$ should be chosen to minimize $\bar{F}(\boldsymbol{\xi}^k \oplus \Delta\boldsymbol{\xi}^k)$. According to the Gauss-Newton method, by differentiating (7) for $\boldsymbol{\xi} = \boldsymbol{\xi}^k$, the optimal value of $\Delta\boldsymbol{\xi}^k$ satisfies the linear system

$$(\boldsymbol{H}_I^k + \lambda \boldsymbol{H}_D^k)\Delta\boldsymbol{\xi}^k = -(\boldsymbol{b}_I^k + \lambda \boldsymbol{b}_D^k), \tag{8}$$

where $\boldsymbol{b}_j^k$ denotes $\boldsymbol{b}_j$ with $\boldsymbol{\xi} = \boldsymbol{\xi}^k$ and $\boldsymbol{H}_j^k$ denotes $\boldsymbol{H}_j$ with $\boldsymbol{\xi} = \boldsymbol{\xi}^k$. To solve this linear system, methods such as Cholesky decomposition can be used. After solving (8),



(a) Experiment 1 (rich structural features)

(b) Experiment 2 (poor textural features)

Figure 3: Ratio of root mean square error (RMSE) and maximum error for two computational experiments using the TUM RGB-D dataset.

the updated motion estimate is given by $\boldsymbol{\xi}^{k+1} = \boldsymbol{\xi}^k \oplus \Delta\boldsymbol{\xi}^k$. This iterative process continues until convergence is achieved.

The effectiveness of the weighted sum method depends crucially on the weighting factor λ, which must be selected a priori and reflects the preference of the decision maker. A good choice for λ can result in more accurate trajectory estimates when compared to single-objective odometry methods, but a poor choice for λ may lead to unacceptable results. Systematic approaches to selecting the weights in multi-objective optimization problems have been developed (see, for example, [13]), but few of them have been investigated in the context of visual odometry. Tykkala et al. [8] proposed a method that determines λ based on the ratio of median intensity and median depth values:

$$\lambda = \left|\text{median}(I)/\text{median}(D)\right|^2,$$

where I denotes the list of intensity values and D denotes the list of depth values.

To explore the importance of the weight λ, we conducted two computational experiments with the TUM RGB-D dataset. For our first experiment, we used two image sequences from the *Structure vs. Texture* category in the TUM RGB-D dataset: one containing images with poor textural features and one containing images with rich textural features. The structural features in both image sequences were rich. We observed that for the first sequence with poor textural features, the error decreases as λ is increased, but for the second sequence with rich textural features, the opposite occurs (see Figure 3(a)). We believe that this is because the intensity objective function F_I tends to be non-convex when images lack textural features. In this case, large values of λ magnify the relative importance of the depth objective function F_D, thus potentially preventing the overall objective function in (6) from becoming non-convex.

For our second experiment, we again used two image sequences from the *Structure vs. Texture* category in the TUM RGB-D dataset: this time the first image sequence contained images with poor structural features and poor textural features, and the second image sequence contained images with rich structural features and poor textural features. As expected, the error decreases as λ increases for the image sequence with rich structural features (see Figure 3(b)). This is because F_D is likely to be convex when images contain rich structural information, and a large λ will increase F_D's relative influence in the overall objective function.

Based on the experimental results in Figure 3, we believe that the key to finding an optimal λ is to design a metric to measure textural and structural information. To do this, we consider the concept of *image complexity*, which is a measure of the inherent difficulty of finding a true target in a given image [14]. Peters et al. [14] has summarized many image complexity metrics for automatic target recognizers. Unfortunately, image complexity is a task-dependent notion and there is no universal metric applicable to all situations. After testing several of the metrics in [14], we designed our own metric for intensity complexity defined as follows:

$$\begin{aligned}\pi(I) = \frac{1}{(v-2)(h-2)} \sum_{i=2}^{v-1}\sum_{j=2}^{h-1} \{|I(i+1,j) \\ -I(i-1,j)| + |I(i,j+1) - I(i,j-1)|\},\end{aligned} \tag{9}$$

where v and h are the number of pixel rows and pixel columns, respectively, and $I(\cdot,\cdot)$ denotes the intensity value at the specified pixel. For depth complexity, we use the analogue of (9) for the depth values:

$$\begin{aligned}\pi(D) = \frac{1}{(v-2)(h-2)} \sum_{i=2}^{v-1}\sum_{j=2}^{h-1} \{|D(i+1,j) \\ -D(i-1,j)| + |D(i,j+1) - D(i,j-1)|\},\end{aligned} \tag{10}$$

where $D(\cdot,\cdot)$ denotes the depth value at the specified pixel. To standardize the intensity data I and the depth data D, we define the following scaling factor as the ratio of the variance between them:

$$\gamma = \frac{\sigma^2(I)}{\sigma^2(D)}. \tag{11}$$

Combining (9)-(11), we calculate the value of weight λ as follows:

$$\lambda = \frac{\phi\gamma^2\pi(D)^2}{\pi(I)^2}, \tag{12}$$

where γ is as defined in (11) and ϕ is an adjustable constant. Notice that large values of $\pi(I)$ indicate rich textural features, and large values of $\pi(D)$ indicate rich structural features. Thus, we have deliberately chosen the value of

λ in (12) to be inversely proportional to $\pi(I)$, and proportional to $\pi(D)$. The idea is to use large values of λ when the image sequence is rich in structure and/or poor in texture, and small values of λ when the image sequence is poor in structure and/or rich in texture.

3.2 Bounded Objective Method

The bounded objective method is another method for solving multi-objective optimization problems [13]. In this method, we minimize one of the objective functions (considered as the most important, or primary, objective), while the other objective functions are bounded using additional constraints.

For our odometry problem, we select $F_I(\boldsymbol{\xi})$ as the primary objective function. The bi-objective optimization problem in (4) then becomes

$$\begin{aligned} &\underset{\boldsymbol{\xi}}{\text{minimize}}\ F_I(\boldsymbol{\xi}) \\ &\text{subject to } F_D(\boldsymbol{\xi}) \leq \epsilon_D, \end{aligned} \tag{13}$$

where ϵ_D is an upper bound for the least-squares sum of depth residuals. To solve the optimization problem in (13), we can again use the first-order Taylor expansions of $\boldsymbol{r}_I(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi})$ and $\boldsymbol{r}_D(\boldsymbol{\xi} \oplus \Delta\boldsymbol{\xi})$. The optimal increment $\Delta\boldsymbol{\xi}$ at point $\boldsymbol{\xi}$ is then given by the solution of the following problem:

$$\begin{aligned} &\underset{\Delta\boldsymbol{\xi}}{\text{minimize}}\ \Delta\boldsymbol{\xi}^\top \boldsymbol{H}_I \Delta\boldsymbol{\xi} + 2\boldsymbol{b}_I^\top \Delta\boldsymbol{\xi} + a_I \\ &\text{subject to } \Delta\boldsymbol{\xi}^\top \boldsymbol{H}_D \Delta\boldsymbol{\xi} + 2\boldsymbol{b}_D^\top \Delta\boldsymbol{\xi} + a_D \leq \epsilon_D, \end{aligned} \tag{14}$$

where $\boldsymbol{H}_I$, $\boldsymbol{H}_D$, $\boldsymbol{b}_I$, $\boldsymbol{b}_D$, a_I and a_D are as defined in (7). Problem (14) is a *quadratically constrained quadratic program* (QCQP). The general form for a QCQP is

$$\begin{aligned} &\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{minimize}}\ \boldsymbol{\alpha}^\top \boldsymbol{H}_0 \boldsymbol{\alpha} + 2\boldsymbol{b}_0^\top \boldsymbol{\alpha} + a_0 \\ &\text{subject to } \boldsymbol{\alpha}^\top \boldsymbol{H}_i \boldsymbol{\alpha} + 2\boldsymbol{b}_i^\top \boldsymbol{\alpha} + a_i \leq 0,\ i = 1, \ldots, q. \end{aligned}$$

QCQPs are of both theoretical and practical significance [15]. Because the matrices H_I and H_D are positive semidefinite, problem (14) is a convex QCQP. To solve this convex QCQP, we first transform it into a *second-order cone programming* (SOCP) problem and then apply SOCP techniques [16]. The general form for a SOCP problem is

$$\begin{aligned} &\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{minimize}}\ \boldsymbol{c}^\top \boldsymbol{\alpha} \\ &\text{subject to } \|\boldsymbol{A}_i \boldsymbol{\alpha} + \boldsymbol{p}_i\| \leq \boldsymbol{q}_i^\top \boldsymbol{\alpha} + d_i,\ i = 1, \ldots, q. \end{aligned}$$

The norm appearing in the constraints is the standard Euclidean norm, i.e., $\|\boldsymbol{u}\| = (\boldsymbol{u}^\top \boldsymbol{u})^{1/2}$. We first rewrite (14) as follows:

$$\begin{aligned} &\underset{\Delta\boldsymbol{\xi}}{\text{minimize}}\ \left\|\boldsymbol{\Omega}_I^{1/2} \boldsymbol{J}_I \Delta\boldsymbol{\xi} + \boldsymbol{\Omega}_I^{1/2} \boldsymbol{r}_I\right\|^2 \\ &\text{subject to } \left\|\boldsymbol{\Omega}_D^{1/2} \boldsymbol{J}_D \Delta\boldsymbol{\xi} + \boldsymbol{\Omega}_D^{1/2} \boldsymbol{r}_D\right\|^2 \leqslant \epsilon_D. \end{aligned} \tag{15}$$

By adding a new optimization variable $t \in \mathbb{R}$, we can transform (15) into the following SOCP form:

$$\begin{aligned} &\underset{(\Delta\boldsymbol{\xi}, t)}{\text{minimize}}\ t \\ &\text{subject to } \left\|\boldsymbol{\Omega}_I^{1/2} \boldsymbol{J}_I \Delta\boldsymbol{\xi} + \boldsymbol{\Omega}_I^{1/2} \boldsymbol{r}_I\right\| \leqslant t \\ &\qquad\qquad \left\|\boldsymbol{\Omega}_D^{1/2} \boldsymbol{J}_D \Delta\boldsymbol{\xi} + \boldsymbol{\Omega}_D^{1/2} \boldsymbol{r}_D\right\| \leqslant \sqrt{\epsilon_D}. \end{aligned} \tag{16}$$

Problem (16), which is equivalent to (14) and (15) (see [16]), is clearly in the general SOCP form shown above.

To solve the SOCP problem in (16), we can use ECOS, an SOCP solver developed by Domahidi et al. [17]. ECOS implements an interior point method to solve SOCPs in the following standard form [18]:

$$\begin{aligned} &\underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{minimize}}\ \boldsymbol{c}^\top \boldsymbol{\alpha} \\ &\text{subject to } \boldsymbol{G}\boldsymbol{\alpha} + \boldsymbol{s} = \boldsymbol{h},\ \boldsymbol{s} \in \boldsymbol{K}, \end{aligned}$$

where $\boldsymbol{\alpha}$ is a vector of optimization variables, $\boldsymbol{s}$ is a vector of slack variables and $\boldsymbol{K}$ is the cone

$$\boldsymbol{K} = \prod_{\mu=1}^{N} \{(u_0, \boldsymbol{u}_1) \in \mathbb{R} \times \mathbb{R}^{m_\mu - 1}\ :\ u_0 \geq \|\boldsymbol{u}_1\|\}.$$

To reformulate (16) into the standard form required by ECOS, we set

$$\boldsymbol{\alpha} = \begin{bmatrix} t \\ \Delta\boldsymbol{\xi} \end{bmatrix},$$

and

$$\boldsymbol{G} = \begin{bmatrix} -1 & \boldsymbol{0}_6^\top \\ \boldsymbol{0}_n & -\boldsymbol{\Omega}_I^{1/2} \boldsymbol{J}_I \\ 0 & \boldsymbol{0}_6^\top \\ \boldsymbol{0}_n & -\boldsymbol{\Omega}_D^{1/2} \boldsymbol{J}_D \end{bmatrix},\ \boldsymbol{h} = \begin{bmatrix} 0 \\ \boldsymbol{\Omega}_I^{1/2} \boldsymbol{r}_I \\ \sqrt{\epsilon_D} \\ \boldsymbol{\Omega}_D^{1/2} \boldsymbol{r}_D \end{bmatrix},$$

where $\boldsymbol{0}_n$ denotes the zero column vector in $\mathbb{R}^n$.

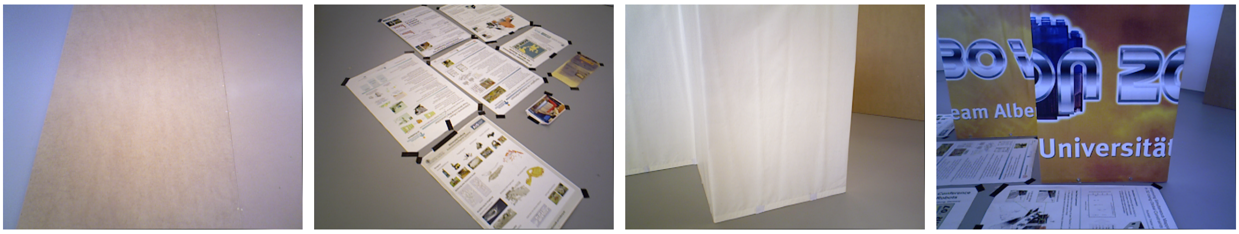
The upper bound ϵ_D of the depth objective $F_D(\boldsymbol{\xi})$ is a parameter that needs to be selected before starting the optimization procedure. This parameter plays the same role as λ in (6), i.e., balancing the relative importance of the depth and intensity objectives. However, compared to λ, the upper bound ϵ_D has a more explicit mathematical meaning and is easier to select a priori. In fact, since the value of $F_D(\boldsymbol{\xi})$ can be measured directly when the true value of the camera motion $\boldsymbol{\xi}^*$ is substituted into $F_D(\boldsymbol{\xi})$, we can usually determine an appropriate range for ϵ_D through experimentation. In our algorithm, we choose the value of ϵ_D according to image depth complexity as follows:

$$\epsilon_D = \begin{cases} \epsilon_{\text{max}}, & \text{if } \pi(D) \leq \delta, \\ \epsilon_{\text{min}}, & \text{otherwise}, \end{cases}$$

where $\epsilon_{\text{min}} \ll \epsilon_{\text{max}}$, δ is an adjustable threshold and $\pi(D)$ is the depth metric in (10).

4 PERFORMANCE EVALUATION

For performance evaluation, we conducted a series of numerical experiments using image sequences from the *Structure vs. Texture* category in the TUM RGB-D dataset.



(a) Poor structure & poor texture (b) Poor structure & rich texture (c) Rich structure & poor texture (d) Rich structure & rich texture

Figure 4: The four types of images in the *Structure vs. Texture* category in the TUM RGB-D dataset.

Table 1: RMSE results for image sequences 1-4 in the *Structure vs. Texture* category.

Method	Poor structure Rich texture [m/s]	Rich structure Poor texture [m/s]	Poor structure Poor texture [m/s]	Rich structure Rich texture [m/s]
Single objective	0.041667	0.125235	0.249357	0.015956
Tykkala's method	0.035970	0.106649	**0.165702**	0.016078
Weighted sum	0.034464	**0.088853**	0.178571	**0.015101**
Bounded objective	**0.032715**	0.095749	0.178994	0.015330

Table 2: RMSE results for image sequences 5-8 in the *Structure vs. Texture* category.

Method	Poor structure Rich texture [m/s]	Rich structure Poor texture [m/s]	Poor structure Poor texture [m/s]	Rich structure Rich texture [m/s]
Single objective	0.110646	0.074372	0.170460	0.015597
Tykkala's method	0.094845	0.077504	0.129923	0.014728
Weighted sum	**0.078033**	0.076853	**0.123848**	**0.014284**
Bounded objective	0.098715	**0.066008**	0.152104	0.015269

These images were created using wooden panels to create structure and colorful plastic foils to create texture. The images can be classified into four types as shown in Figure 4.

We considered all 8 image sequences in the *Structure vs. Texture* category. Sequences 1-4 contain images taken at a close distance from the panels and wooden surfaces; sequences 5-8 contain images taken at a far distance. We used 4 different optimization methods to calculate the estimated robot trajectory for each image sequence: the single objective method [7], Tykkala's original bi-objective method [8], and our new weighted sum and bounded objective methods. For each image sequence, we calculated the root mean square error (RMSE) of the drift between the estimated trajectories and the exact trajectory (which is supplied with the TUM RGB-D dataset). All computations were done on a ThinkPad E431 laptop with dual-core Intel i5-3210M CPU (2.50GHz) and 4 GB RAM.

To ensure identical experimental conditions for each optimization method, the weight matrices $\boldsymbol{\Omega}_I$ and $\boldsymbol{\Omega}_D$ at each iteration k were chosen as diagonal matrices in which the i-th diagonal elements are defined, respectively, by

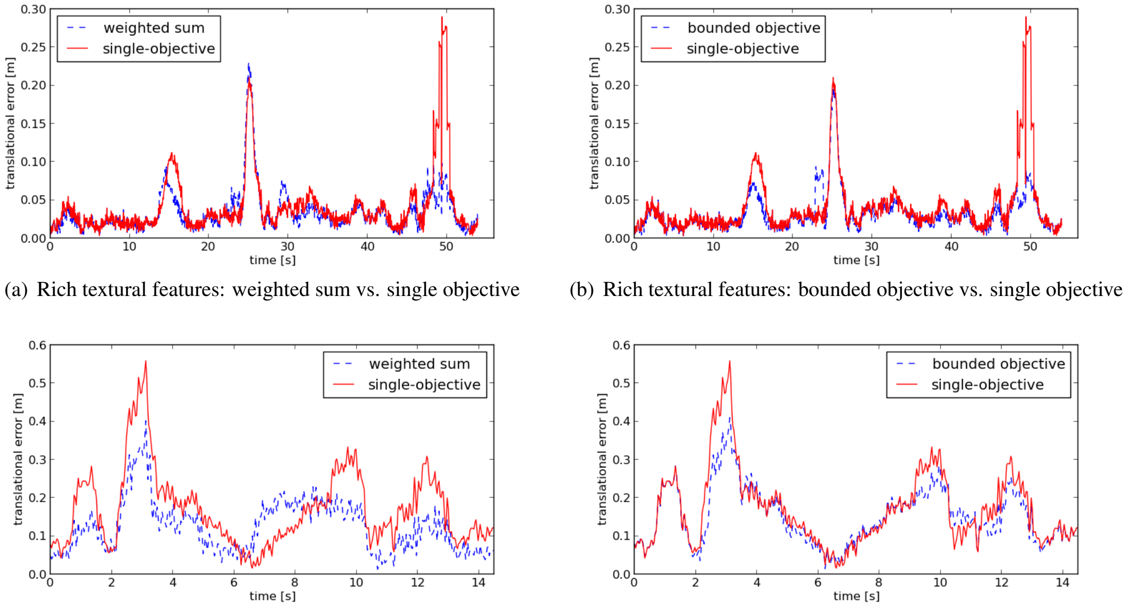
$$\omega(r_I^{(i)}(\boldsymbol{\xi}^{k-1})) = \frac{\nu + 1}{\sigma^2\nu + (r_I^{(i)}(\boldsymbol{\xi}^{k-1}))^2},$$

$$\omega(r_D^{(i)}(\boldsymbol{\xi}^{k-1})) = \frac{\nu + 1}{\sigma^2\nu + (r_D^{(i)}(\boldsymbol{\xi}^{k-1}))^2},$$

where $\boldsymbol{\xi}^{k-1}$ is the motion estimate at iteration $k-1$ and ν and σ are parameters. For more details about how to choose the values of ν and σ, see [7].

The results of our experiments are given in Table 1 and Table 2 (the per-frame translational errors are also shown in Figure 5). It can be seen that the RMSE of the single-objective optimization method increases considerably when textural features are poor. Compared to single-objective optimization, our new bi-objective methods, the weighted sum method and the bounded objective method, give better performance, especially in the scenarios with poor textural features. Tykkala's method in [8], which also uses bi-objective optimization, has similar performance to our methods. Our conclusion is that the new bi-objective optimization formulation for RGB-D odometry can improve motion estimates by reducing the chances of the optimization problem being non-convex.

We also measured the average runtime for each method to perform one match between two images, using all 8 image sequences in the *Structure vs. Texture* category. From Table 3, we can see that our weighted sum method needs 49.09% more time to accomplish one match than the method based on single objective optimization. But as the time for one match is much less than one second, our weighted sum method can still be implemented as a real-time approach. The bounded sum method, however, due to its expensive computational cost, cannot currently work in a real-time environment. The main reason for the large computational burden is that the algorithms used to solve the SOCP are numerical approximation algorithms. They need more computations and iterations than the analytic algorithms, such as Gauss-Newton algorithm, used in the



(a) Rich textural features: weighted sum vs. single objective

(b) Rich textural features: bounded objective vs. single objective

(c) Poor textural features: weighted sum vs. single objective

(d) Poor textural features: bounded objective vs. single objective

Figure 5: Per-frame translational errors for the single objective method, the weighted sum method, and the bounded objective method.

Table 3: Computation times for the different RGB-D odometry methods in our experiments.

Method	CPU Time [ms]
Single objective	15.42
Tykkala's method	21.06
Weighted sum	22.99
Bounded objective	7093.00

weighted sum method. Nevertheless, since it depends on an easily-quantifiable parameter, the bounded sum method is still a promising alternative to other methods in bi-objective optimization for visual odometry.

5 CONCLUSION

In this paper, we studied two methods for solving a new bi-objective optimization formulation for robust RGB-D odometry. Both methods involve converting the bi-objective optimization problem into a single-objective problem. The weighted sum method involves minimizing the weighted linear sum of intensity and depth residuals. The bounded objective method involves minimizing the intensity residual subject to a bound on the depth residual. The experimental results show that both methods yield precise motion estimates and perform reliably even when the textural information in the image sequence is poor. The bounded objective method is considerably slower than the weighted sum method. Thus, our current focus is on developing a parallel algorithm for enhancing real-time performance. We also hope to expand these ideas to other problems in robotics such as motion control, SLAM and navigation. One of the main contributions of our work is a discussion of how to use depth and intensity metrics to choose the parameters in both methods.

REFERENCES

[1] D. Nistér, O. Naroditsky, and J. Bergen, Visual Odometry, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 652-659, 2004.

[2] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics, MIT Press, 2005.

[3] P. J. Besl, and N. D. McKay, Method for Registration of 3-D Shapes, in Robotics-DL Tentative, International Society for Optics and Photonics, 586-606, 1992.

[4] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments, in Experimental Robotics, Springer, 477-491, 2014.

[5] H. Strasdat, J. Montiel, and A. J. Davison, Scale Drift-Aware Large Scale Monocular SLAM, in Robotics: Science and Systems, Vol. 2, No. 3, 5-12, 2010.

[6] H. Steinbrucker, J. Sturm, and D. Cremers, Real-Time Visual Odometry from Dense RGB-D Images, in Proceedings of the IEEE International Conference on Computer Vision Workshops, 719-722, 2011.

[7] C. Kerl, J. Sturm, and D. Cremers, Robust Odometry Estimation for RGB-D Cameras, in Proceedings of the IEEE International Conference on Robotics and Automation, 3748-3754, 2013.

[8] T. Tykkala, C. Audras, and A. I. Comport, Direct Iterative Closest Point for Real-Time Visual Odometry, in Proceedings of the IEEE International Conference on Computer Vision Workshops, 2050-2056, 2011.

[9] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, A Benchmark for the Evaluation of RGB-D SLAM Systems, in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 573-580, 2012.

[10] L. Zadeh, Optimality and Non-scalar-valued Performance Criteria, IEEE Transactions on Automatic Control, Vol. 8, No. 1, 59-60, 1963.

[11] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, G2O: A General Framework for Graph Optimization, in Proceedings of the IEEE International Conference on Robotics and Automation, 3607-3613, 2011.

[12] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, An Invitation to 3-D Vision: From Images to Geometric Models, Springer, 2003.

[13] R. T. Marler, and J. S. Arora, Survey of Multi-objective Optimization Methods for Engineering, Structural and Multidisciplinary Optimization, Vol. 26, No. 6, 369-395, 2004.

[14] R. A. Peters, and R. N. Strickland, Image Complexity Metrics for Automatic Target Recognizers, in Proceedings of the Automatic Target Recognizer System and Technology Conference, 1-17, 1990.

[15] C. Lu, S. Fang, Q. Jin, Z. Wang, and W. Xing, KKT Solution and Conic Relaxation for Solving Quadratically Constrained Quadratic Programming Problems, SIAM Journal on Optimization, Vol. 21, No. 4, 1475-1490, 2011.

[16] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, Applications of Second-order Cone Programming, Linear Algebra and its Applications, Vol. 284, No. 1, 193-228, 1998.

[17] A. Domahidi, E. Chu, and S. Boyd, ECOS: An SOCP Solver for Embedded Systems, in Proceedings of the European Control Conference, 3071-3076, 2013.

[18] S. Andersen, J. Dahl, and L. Vandenberghe, CVXOPT: A Python Package for Convex Optimization, version 1.1.6, Available at cvxopt.org, 2013.