Fast Inverse Compositional Image Alignment with Missing Data and Re-weighting

Vincent Lui
vincent.lui@monash.edu
Dinesh Gamage
dinesh.gamage@monash.edu
Tom Drummond
tom.drummond@monash.edu

ARC Centre of Excellence for Robotic Vision
Monash University
Clayton, Australia

Abstract

This paper proposes a novel method of performing inverse compositional image alignment which elegantly deals with missing data and re-weighting, and does not require the Jacobians and Hessian to be re-computed at every iteration. We show how missing data and re-weighting can be handled through preconditioning. We propose a few preconditioning techniques and analyse how each technique models the effects of missing data and re-weighting for inverse composition. We show through extensive experiments on different applications that our method improves the convergence rate of the conventional re-weighted inverse compositional method while remaining robust to outliers. We also show that the the update parameters are usually underestimated and how this can be used to further speed up convergence of image alignment methods.

1 Introduction

Image alignment is the problem of deforming an image template to align the template with a reference image. It has wide ranging applications in computer vision such as optical flow [16], tracking [8], and image mosaic-ing [15]. Image alignment is often performed using variants of the Lucas-Kanade algorithm [16]. Among these variants, the inverse compositional (IC) method [2] and the efficient second-order minimization (ESM) method [5] are the most efficient variants for image alignment. The benefit of IC is that the Jacobian matrix and its Hessian can be pre-computed, whereas ESM provides a second-order approximation of the Hessian and thus converges in fewer number of iterations.

The choice of using IC or ESM depends on the cost of re-computing the Jacobian at every iteration. While ESM is computationally more efficient in image alignment with homographies [17], IC is still computationally more efficient in image alignment problems which use RGB-D data [14] where it is expensive to re-compute the Jacobian. In this paper, we look at methods to accelerate the convergence of IC and ESM algorithms while remaining robust to outliers. Our contributions are as follows:

- The conventional re-weighted IC algorithm [4] requires the Jacobian and its Hessian to be re-computed at every iteration. We show that re-weighted IC can still be performed

Pages 54.1-54.12
DOI: https://dx.doi.org/10.5244/C.29.54

without needing to re-compute the Jacobian and its Hessian by using a preconditioning strategy. We provide an analysis for a number of approaches (see Section 2.2).

- Through extensive experiments on various image alignment problems with different warping models such as affine warps, homographies, and SE(3) warps using RGB-D data, that our proposed methods are equally as robust and provide faster convergence than the original re-weighted IC method (see Sections 3.1 and 3.2).

- Further, we show how consideration of the effects of image noise and/or spectral aliasing over the scale of the deformation can be used to further speed up all of these methods (see Section 3.3).

1.1 Related Work

Image alignment approaches can be divided into energy-based methods [2, 10, 16, 19] and learning-based methods [11, 13]. Energy-based methods perform image alignment by minimizing an energy function based on the photometric error between pixels in the template and reference image. In contrast, learning-based methods have an offline learning stage [13] which randomly warp samples of the initial template to learn how to predict parameter updates during run-time. The offline learning phase is usually time consuming. Holzer *et al.* [11] proposed a few strategies to reduce the learning time using dimensionality reduction.

Energy-based image alignment methods are usually variants of the Lucas-Kanade (LK) algorithm [16], with [3] providing an excellent survey. The LK algorithm can be formulated as *additive* or *compositional*, and these can be further formulated as *forward* or *inverse*. This results in four variants: the forward additive (FA) method, forward compositional (FC) method [19], inverse additive (IA) method [10], and the inverse compositional (IC) method [2]. The FA and FC variants can be extended through the efficient second-order minimization (ESM) method [5, 17] which provides a second-order approximation of the Hessian. The problem of re-weighting in the LK algorithm is well studied in literature [1, 4, 7, 10]. Baker *et al.* [4] studied the use of weighting functions in the different variants of the LK algorithm, and showed that re-weighting via M-estimators in the IC method necessitates the re-computation of the Jacobians and Hessian at every iteration, making it as expensive as the canonical LK algorithm. The authors also studied the use of weighted L2 norms in the LK algorithm, and proposed to weigh every pixel by the magnitude of its image gradient. Dellaert and Collins [7] proposed the use of binary weights based on a threshold on the gradient magnitude. Hager and Belhumeur [10] assume that the unweighted Hessian is a good approximate to the weighted Hessian, so only the Jacobians need to be re-computed at every iteration. Ashraf *et al.* [1] proposed to formulate the LK algorithm in the Fourier domain, and showed how it can be done through a pre-computed weighting matrix and how it can be extended to an IC formulation.

2 Inverse Composition with Preconditioning

We first provide a brief summary of the Lucas-Kanade (LK) algorithm [16], its inverse compositional (IC) variant, and the efficient second-order minimization (ESM) technique [5]. For full derivations, we refer the reader to [3] for the LK method and [5] for ESM. The LK algorithm minimizes the cost function C_{LK}, which is the sum of squared errors over all pixels

$\mathbf{x}$ between a template image T and a reference image I that is warped onto the template:

$$C_{\text{LK}} = \sum_{\mathbf{x}} [I(\mathbf{M}(\mathbf{x};\mathbf{p})) - T(\mathbf{x})]^2, \tag{1}$$

where $\mathbf{p}$ is the parameter vector and $\mathbf{x}$ is the pixel location in the template. $\mathbf{M}(\mathbf{x};\mathbf{p}))$ is a warping function which maps the pixel $\mathbf{x}$ in the template T to a sub-pixel location $I(\mathbf{M}(\mathbf{x};\mathbf{p}))$ in the reference image I. The cost in Equation (1) is minimized iteratively using a first-order approximation of the cost function. This requires computing the Jacobian $J_{\mathbf{x}} = \nabla I \frac{\partial \mathbf{M}}{\partial \mathbf{p}}$ for every pixel. The partial derivative $\frac{\partial \mathbf{M}}{\partial \mathbf{p}}$ has to be evaluated at $\mathbf{M}(\mathbf{x};\mathbf{p})$, and hence $J_{\mathbf{x}}$ has to be re-computed at every iteration. We denote this Jacobian as $J(\delta)$. The IC algorithm switches the role between the template T and the reference image I, and minimizes the cost function C_{IC}, which is the sum of squared errors

$$C_{\text{IC}} = \sum_{\mathbf{x}} [T(\mathbf{M}(\mathbf{x};\Delta\mathbf{p})) - I(\mathbf{M}(\mathbf{x};\mathbf{p}))]^2 \tag{2}$$

with respect to $\Delta\mathbf{p}$. As $J_{\mathbf{x}}$ is now computed using the template image T which does not depend on $\mathbf{p}$, it can be pre-computed as $J(0)$. ESM linearizes the cost function in Equation (1) using a second-order Taylor expansion. The only difference with the LK algorithm is the way in which the Jacobian $J_{\mathbf{x}}$ is computed for each pixel. The Jacobian for ESM is an average of the Jacobian $J(\delta)$ from the canonical LK algorithm which has to be computed at every iteration, and the Jacobian $J(0)$ from the IC method which can be pre-computed:

$$J_{\mathbf{x}_{\text{ESM}}} = 0.5[J(\delta) + J(0)]. \tag{3}$$

2.1 Minimizing the Cost Function

The cost functions in Section 2 are minimized by updating the parameter vector $\mathbf{p}$ by $\Delta\mathbf{p}$ iteratively. $\mathbf{p}$ is related to the error vector $\mathbf{e}$ as

$$J\Delta\mathbf{p} = \mathbf{e}, \tag{4}$$

where J is an $m \times n$ Jacobian matrix ($m \geq n$). At each iteration, we form the normal equations

$$H\Delta\mathbf{p} = J^T\mathbf{e}, \tag{5}$$

where $H = J^T J$ is the Hessian. $\Delta\mathbf{p}$ is then computed as $\Delta\mathbf{p} = H^{-1}J^T\mathbf{e}$. If re-weighting is necessary due to outliers and/or unused pre-computed Jacobians due to missing data (e.g. part of the template image may not be aligned with the reference image), Equation (5) can be re-written as

$$(J^T W J)\Delta\mathbf{p} = J^T W\mathbf{e}, \tag{6}$$

where W is usually a diagonal matrix. The value $w_{\mathbf{x}}$ in each row is the weight of the pixel location $\mathbf{x}$, and it can be either a robust function $\rho(;\sigma)$ or a spatially aware re-weighting function which weighs each measurement based on the magnitude of its image gradient [4].

2.2 Re-weighting through QR Factorization

As H and its weighted version $J^T W J$ is symmetric and positive-definite, its pseudo-inverse H^{-1} is usually computed through a Cholesky factorization [2]. We propose using a QR

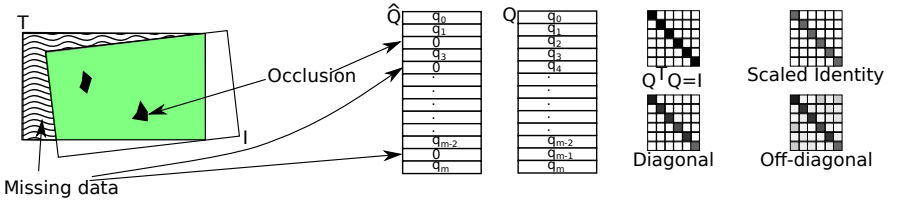

Figure 1: Left: Missing data and re-weighting causes the factor Q to turn into a matrix $\hat{Q}$ that is no longer orthogonal. Right: The proposed preconditioners P. The product $P^{-1}(\hat{Q}^T\hat{Q})$ should be as close to I as possible. Here, a darker shade represents a value closer to 1.0.

factorization [9] on the Jacobian matrix J instead. Equation (4) can now be written as

$$Q\begin{bmatrix} R \\ 0 \end{bmatrix}\Delta\mathbf{p} = \mathbf{e}, \tag{7}$$

where Q is an $m \times m$ orthogonal matrix where $Q^T = Q^{-1}$, and R is an $m \times n$ upper triangular matrix. The factor Q has a condition number of of 1.0 and handles the dimensionality of the problem as every row in Q corresponds to the Jacobian of the pixel location $\mathbf{x}$. The conditioning in the Jacobian J, on the other hand, is stored in the factor R. To compute $\Delta\mathbf{p}$, both sides of (7) are multiplied by Q^T. As $Q^TQ = I$, $\Delta\mathbf{p}$ can now be computed as

$$\Delta\mathbf{p} = R^{-1}Q^T\mathbf{e}. \tag{8}$$

If re-weighting is performed, Equation (7) can be re-written as

$$\hat{Q}\begin{bmatrix} R \\ 0 \end{bmatrix}\Delta\mathbf{p} = \hat{\mathbf{e}} \tag{9}$$

where $\hat{Q} = \sqrt{W_\mathbf{x}}Q$ and $\hat{\mathbf{e}} = \sqrt{W_\mathbf{x}}\mathbf{e}$. $\hat{Q}$ is not an orthogonal matrix and every column vector in $\hat{Q}$ has a magnitude that is less than 1.0 (see Figure 1). Hence, the product $\hat{Q}^T\hat{Q}$ is not equal to the identity matrix I, i.e. $\hat{Q}^T\hat{Q} \neq I$. $\Delta\mathbf{p}$ in Equation (8) now has to be solved by taking $(\hat{Q}^T\hat{Q})^{-1}$ into account:

$$\Delta\mathbf{p} = R^{-1}(\hat{Q}^T\hat{Q})^{-1}\hat{Q}^T\hat{\mathbf{e}}. \tag{10}$$

If we substitute $\hat{Q} = \sqrt{W_\mathbf{x}}Q$ and $\hat{\mathbf{e}} = \sqrt{W_\mathbf{x}}\mathbf{e}$ into Equation (10), the re-weighting problem can be re-formulated using the original factors Q and R:

$$\Delta\mathbf{p} = R^{-1}(Q^TWQ)^{-1}QW\mathbf{e}. \tag{11}$$

2.3 Re-weighting as a Preconditioning Problem

In Equation (11), the factors Q and R can be pre-computed. At every iteration, the weight matrix W, the error vector $\mathbf{e}$, and the product $(\hat{Q}^T\hat{Q})^{-1}$ has to be computed. We propose a preconditioning approach [8] to approximate $(\hat{Q}^T\hat{Q})^{-1}$, where the goal is to construct a matrix P so that the product $P^{-1}(\hat{Q}^T\hat{Q})$ has a smaller condition number compared to $\hat{Q}^T\hat{Q}$. Ideally, the matrix P is $P = \hat{Q}^T\hat{Q}$ as $P^{-1}(\hat{Q}^T\hat{Q})$ has a condition number of 1.0. However, this requires solving P^{-1} which is computationally as expensive as solving the original problem. At the other extreme, $P = I$ is computationally cheapest, but does nothing to reduce the condition number of the problem. Here, we propose a few preconditioning matrices P (see Figure 1) that lie between the two extremes $P = \hat{Q}^T\hat{Q}$ and $P = I$:

1. The *scaled identity matrix*

$$P = kI, \tag{12}$$

where k is a scalar value, $k \geq 1.0$. The value k can be chosen as the inverse of the average weight, i.e. $k = \frac{N}{\sum_{\mathbf{x}} w_{\mathbf{x}}}$, where N is the number of measurements.

2. The *diagonal/Jacobi preconditioner*

$$P = \text{diag}(\hat{Q}^T \hat{Q}) = \Lambda, \tag{13}$$

which is computed by accumulating the sum of weighted, squared elements in the corresponding column j of the pre-computed factor Q. More concretely, $\Lambda_j = \sum_i w_i q_{i,j}^2$ where i represents the i-th row.

3. Finally, the Jacobi preconditioner in (13) can be extended to include off-diagonal elements as well, resulting in

$$P = \text{diag}(\sqrt{\Lambda})(I + \varepsilon)\text{diag}(\sqrt{\Lambda}), \tag{14}$$

where ε is a symmetric matrix, and $\varepsilon_{ij} = 0 \, \forall \, i = j$. The off-diagonal elements $\varepsilon_{ij} \, \forall \, i \neq j$ can be computed as

$$\varepsilon_{ij} = \frac{\sum_r w_r q_{ri} q_{rj}}{\sqrt{\Lambda_i}\sqrt{\Lambda_j}}. \tag{15}$$

For the preconditioners in Equation (12) and (13), the inverse P^{-1} is simply an inversion of every diagonal element in P. For Equation (14), assuming that $\varepsilon_{ij} \approx 0$, $(I + \varepsilon)^{-1}$ can be computed with a first-order Neumann series [18] approximation:

$$\begin{aligned} (I + \varepsilon)^{-1} &= \textstyle\sum_n^{\infty} (-1)^n \varepsilon^n = I - \varepsilon + \varepsilon^2 - \varepsilon^3 \cdots \\ &\approx I - \varepsilon \end{aligned}. \tag{16}$$

The inverse of (14) can then be computed as

$$P^{-1} = (\sqrt{\Lambda})^{-1}(I - \varepsilon)(\sqrt{\Lambda})^{-1}. \tag{17}$$

The cost of our proposed methods is compared with the conventional re-weighted IC algorithm in Table 1, where n is the number of warp parameters and N is the number of pixels. For the re-weighting cost of our method, the first term is the cost to compute P and the second term is the cost to compute P^{-1}. For the conventional re-weighted IC, the first term is the cost to compute the Hessian H and the second term is the cost to compute the H^{-1}. Using Householder transformations, the number of operations required for QR factorization is $2n^2N - 2n^3/3$. As $n \ll N$, the pre-computation cost is approximately $O(n^2N)$ and is similar to the conventional re-weighted IC method. The per-iteration cost of the scaled identity and diagonal preconditioners are the cheapest, while the off-diagonal preconditioner is asymptotically as expensive as the original re-weighted IC algorithm.

3 Results

We evaluate the performance of our proposed methods on image alignment with affine warps, homographies, and SE(3) warps. All experiments were conducted on a machine

Cost to:	Eq. (12)	Eq. (13)	Eq. (14)	Re-weighted IC
Pre-compute	$O(n^2N)$	$O(n^2N)$	$O(n^2N)$	$O(n^2N)$
Re-weight	$O(N)+O(n)$	$O(nN)+O(n)$	$O(n^2N)+O(n^2)$	$O(n^2N)+O(n^3)$

Table 1: Cost of the proposed preconditioners compared with the canonical re-weighted IC.

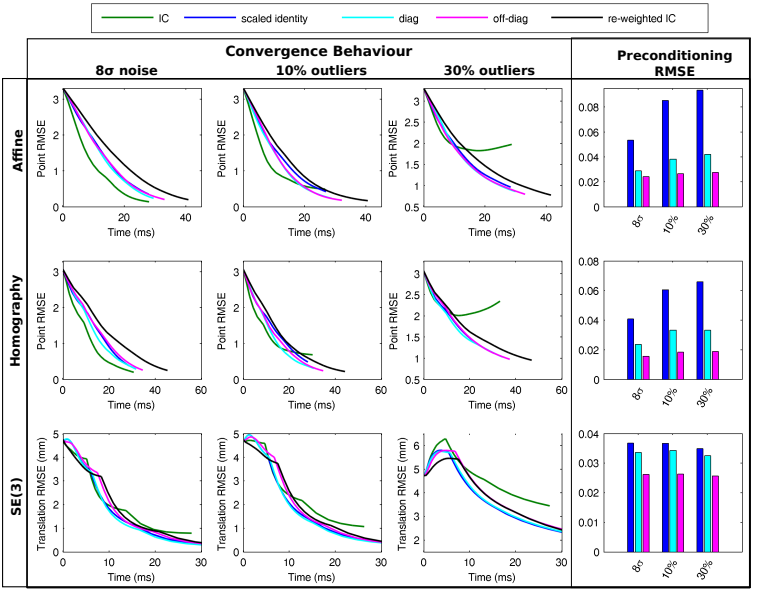

Figure 2: Comparison with the canonical IC and the re-weighted IC algorithm. Top: Affine warps. Middle: Homography. Bottom: SE(3) warps. The first 3 columns from the left show convergence behaviour. The rightmost column shows the preconditioning RMSE.

with a 2.2GHz processor. A C++ implementation of our approach will be made available at `https://bitbucket.org/whvlui/fast_ic/`. We first compare our proposed methods with the canonical IC algorithm and its re-weighted variant in Section 3.1 using a robust M-estimator. Next, we evaluate the performance of our proposed methods using a spatially aware re-weighting function and compare our best performing methods with ESM in Section 3.2. Finally, we consider the effects of noise and/or spectral aliasing on convergence in Section 3.3.

3.1 Comparison with Inverse Composition

In this experiment, we use the Huber M-Estimator [12] for all methods. Noise and/or outliers are added to the images and we measure:

- The convergence behaviour of our proposed methods compared with the original IC algorithm and its re-weighted variant for affine warps, homographies, and SE(3) warps. For affine warps and homographies, we measure the root mean squared error (RMSE) between the true and predicted pixel locations in the reference image. For SE(3) warps, we measure the RMSE between the true and predicted SE(3) pose. Here, we plot the

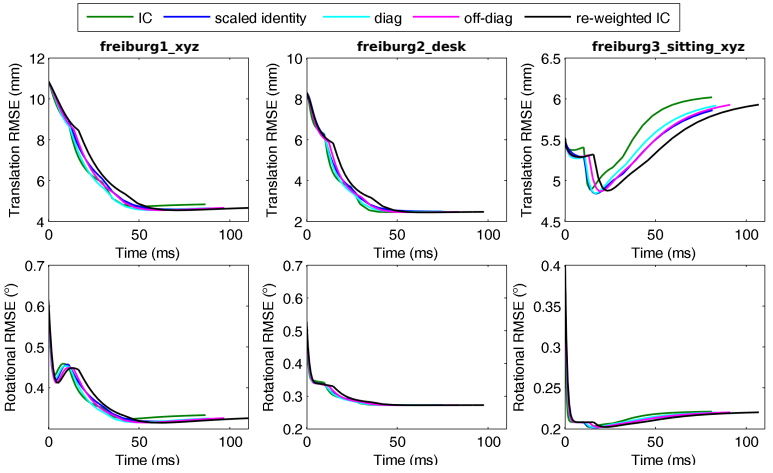

Figure 3: Comparison with canonical IC and re-weighted IC algorithm using real image sequences from TUM RGB-D dataset [20].

convergence behaviour against the average time taken for each iteration.

- How accurately each preconditioner approximates $\hat{Q}^T\hat{Q}$. We measure the RMSE between the product $P^{-1}(\hat{Q}^T\hat{Q})$ and the identity matrix I:

$$\text{Preconditioning RMSE} = \sqrt{\frac{1}{n \cdot m}\sum_{i}^{n}\sum_{j}^{m}((P^{-1}\hat{Q}^T\hat{Q})_{ij} - I_{ij})}. \tag{18}$$

Our experimental setup is similar to [3]. For affine warps and homographies, the template image T is 100×100 pixels, and is manually selected in an image (see Figure 7). For SE(3) warps, the template image T is an RGB-D image from the TUM RGB-D dataset [20] where every pixel has a corresponding depth value. To generate random affine warps, Gaussian noise is added with a certain variance to 3 canonical points on the template, resulting in 3 perturbed points that are used to define the warp parameters. Random homographies are generated similarly, but using 4 canonical points. For SE(3) warps, the translations are sampled from a uniform distribution of ± 0.01m whereas the rotations are sampled from a uniform distribution of $\pm 1^\circ$.

For the noise experiment, Gaussian noise of up to 8σ intensity levels is added to both the template and reference image. For the outlier experiment, the outliers are created by filling a square area in the reference image with intensity values from natural images. 20 iterations is performed for each method for affine warps and homographies. For SE(3) warps, we use 3 pyramid levels for each algorithm, where each level is a factor of 2 smaller than the previous level. 20 iterations is performed for the 2nd level, followed by 10 in the 1st level, and 5 in the 0th level. Each algorithm is allowed to break from the iterations early if the average residual photometric error no longer decreases for a few iterations. Additionally, only pixels with gradient values larger than a certain threshold are used.

The average result obtained from 500 warps is shown in Figure 2. The rightmost column in Figure 2 shows that the off-diagonal preconditioner is, unsurprisingly, the most accurate,

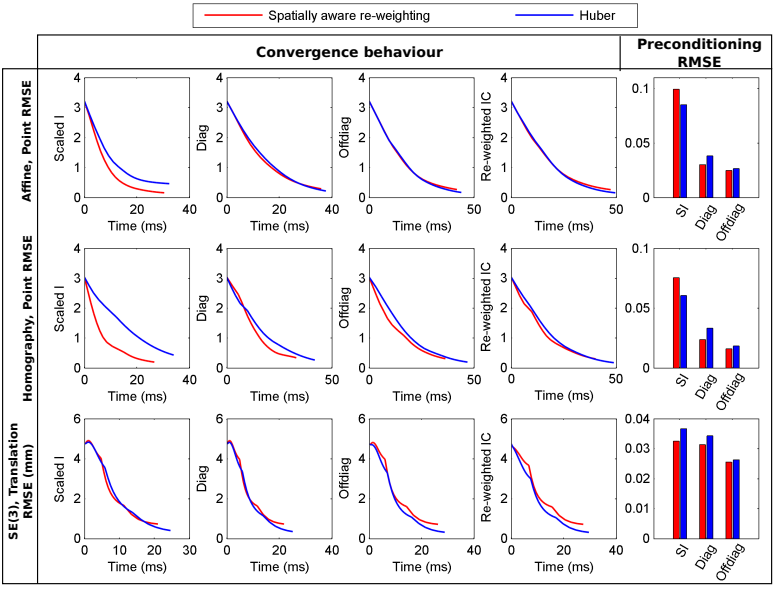

Figure 4: Comparing spatially aware re-weighting (Equation 19) and Huber re-weighting with 10% outliers. Top: Affine warps. Middle: Homography. Bottom: SE(3) warps.

followed by the diagonal method, and then the scaled identity method. In terms of convergence, the canonical IC algorithm is not robust to outliers. The proposed preconditioners are as robust as the conventional re-weighted IC algorithm in all cases. On average, both the scaled identity and the diagonal preconditioners converge in less time compared to the re-weighted IC algorithm. As expected, the off-diagonal method is almost as expensive as the conventional re-weighted IC (see Section 2.2).

We have also evaluated our proposed methods using three real image sequences from the TUM RGB-D dataset [20]. Here, we measure the average convergence using image pairs formed from consecutive images in each sequence, similar to how visual odometry is performed. The result is shown in Figure 3. The canonical IC algorithm along with the diagonal preconditioner display the fastest convergence, followed by the off-diagonal preconditioner and the scaled identity preconditioner, and finally the conventional re-weighted IC. In terms of robustness, our proposed methods converge to the same RMSE values as the conventional re-weighted IC algorithm, whereas the canonical IC algorithm converges to RMSE values that are slightly higher in the *freiburg1_xyz* and the *freiburg3_sitting_xyz* dataset.

3.2 Spatially Aware Re-weighting vs M-Estimators

We repeat the 10% outlier variant of the experiment in Section 3.1 with the following spatially aware re-weighting function:

$$w_x = \frac{|\nabla T(\mathbf{x})|^2}{|\nabla T(\mathbf{x})|^2 + e_x^2}. \tag{19}$$

The intuition behind this weighting function is that pixels with a high gradient are less likely to be affected by noise compared to pixels with a low gradient. The results are shown in

Figure 4. For affine warps and homographies, spatially aware re-weighting provides similar results except for the scaled identity preconditioner, which shows a huge improvement compared to the Huber re-weighting function. This is because the scaled identity preconditioner merely uses the average weight from all the measurements. For SE(3) warps, the Huber re-weighting function provides slightly better results in all cases.

We repeat the same experiment and compare the best performing methods with ESM, which is the scaled identity preconditioner with spatially aware re-weighting for affine warps and homographies, and the diagonal preconditioner with Huber re-weighting for SE(3) warps. The result (see Figure 5) shows that the preconditioned IC method converges as quickly as ESM for affine warps and homographies and converges faster than ESM for SE(3) warps.

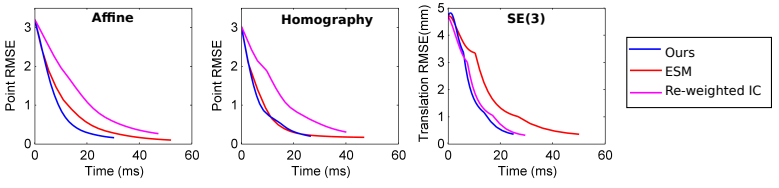

Figure 5: Comparison between the best performing preconditioned IC algorithms with ESM.

3.3 Step Size vs Convergence

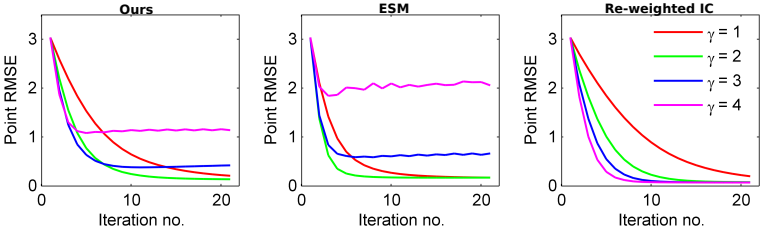

Figure 6: Effects of amplifying step size γ on convergence behaviour. Left: IC with scaled identity preconditioning. Center: ESM. Right: Re-weighted IC.

Here, we look at the effects of noise and/or spectral aliasing on the convergence of image alignment algorithms. If we assume Gaussian intensity noise $\delta \sim N(0, \sigma_\delta)$ in the images and that the corresponding noise Jacobian has a Gaussian distribution $\varepsilon \sim N(0, \sigma_\varepsilon)$, then we can re-write Equation (5) as $(J+\varepsilon)\Delta\mathbf{p} = \mathbf{e}+\delta$, which can then be expanded as

$$(J^TJ + \varepsilon^TJ + J^T\varepsilon + \varepsilon^T\varepsilon)\Delta\mathbf{p} = J^T\mathbf{e} + \varepsilon^T\mathbf{e} + J^T\delta + \varepsilon^T\delta. \tag{20}$$

In Equation (20), all the terms involving ε and δ are ≈ 0, except for $\varepsilon^T\varepsilon$ because these terms involve a large number of independent entries in ε and/or δ. The term $\varepsilon^T\delta$ is also ≈ 0 because ε and δ are independent. This gives:

$$(J^TJ + \varepsilon^T\varepsilon) = J^T\mathbf{e}. \tag{21}$$

Hence, $\varepsilon^T\varepsilon$ acts as a regularisation term in the least squares optimization, causing damping and thus causes the system to underestimate the update parameters. It is then advantageous

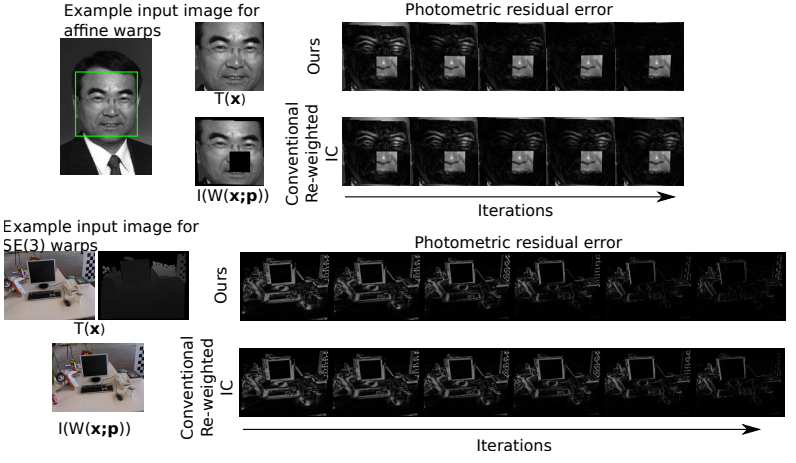

Figure 7: Qualitative results showing the how the residual photometric error changes with the number of iterations.

to amplify the update step by a parameter γ. As discussed in [4], choosing the best step size can be an involved process. We solve this by slowly increasing the step size until the algorithms start showing signs of divergence. The value of γ is then heuristically chosen to be $\gamma = \gamma_{\max}/2$. Figure 6 shows the convergence behaviour for homographies with a few different step sizes. From our experiments, a good value for γ is between 1.5 to 2.0.

4 Conclusion

We have presented a new approach to perform inverse compositional image alignment with missing data and re-weighting which does not need the Jacobian and its Hessian to be re-computed at every iteration. This is possible by solving the problem using QR factorization instead of a Cholesky factorization, followed by a preconditioning strategy. We have showed through experiments with different warp models such as affine warps, homographies, and SE(3) warps that our method is equally as robust and faster than existing image alignment techniques. Among the proposed preconditioning strategies, the diagonal preconditioner provides the best trade-off between computational efficiency and robustness. Finally, we showed that image alignment algorithms typically underestimate the step size of the parameter update due to damping caused by noise and/or spectral aliasing. We showed that a moderate amplification of the step size further improves the convergence of image alignment algorithms. Some qualitative results of our approach is shown in Figure 7 and the supplementary document.

Acknowledgements

The authors wish to thank Dr. Khurrum Aftab for useful discussions. This work was supported by the Australian Research Council Centre of Excellence for Robotic Vision (project number CE1401000016).

References

[1] A.B. Ashraf, S. Lucey, and T. Chen. Fast image alignment in the Fourier domain. In *Computer Vision and Pattern Recognition*, pages 2480–2487, 2010.

[2] S. Baker and I. Matthews. Equivalence and Efficiency of Image Alignment Algorithms. In *Computer Vision and Pattern Recognition*, pages 1090 – 1097, 2001.

[3] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[4] S. Baker, R. Gross, T. Ishikawa, and I. Matthews. Lucas-Kanade 20 Years On : A Unifying Framework : Part 2. Technical report, 2003.

[5] S. Benhimane and E. Malis. Real-time image-based tracking of planes using Efficient Second-order Minimization. In *International Conference on Intelligent Robot and Systems*, volume 1, pages 943–948, 2004.

[6] K. Chen. *Matrix Preconditioning Techniques and Applications*. Cambridge University Press, 2005.

[7] F. Dellaert and R. Collins. Fast Image-Based Tracking by Selective Pixel Integration. In *Proc. ICCV Workshop on Frame-Rate Vision*, pages 1–22, 1999.

[8] J. Engel, J. Sturm, and D. Cremers. Semi-dense Visual Odometry for a Monocular Camera. *Intl. Conference on Computer Vision*, pages 1449–1456, 2013.

[9] G. Golub and C.V. Loan. *Matrix Computations*. John Hopkins University Press, 3rd edition.

[10] G. Hager and P. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. *IEEE transactions on pattern analysis and machine intelligence*, 20(10):1025–1039, 1998.

[11] S. Holzer, M. Pollefeys, S. Ilic, D. Tan, and N. Navab. Online Learning of Linear Predictors for Real-Time Tracking. *Intl. Journal of Computer Vision*, 111(1):12–28, 2014.

[12] P. Huber. *Robust Statistics*. Wiley, 1981.

[13] F. Jurie and M. Dhome. Hyperplane Approximation for Template Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):996–1000, 2002.

[14] S. Klose, P. Heise, and A. Knoll. Efficient compositional approaches for real-time robust direct visual odometry from RGB-D data. In *IEEE International Conference on Intelligent Robots and Systems*, pages 1100–1106, 2013. ISBN 9781467363587. doi: 10.1109/IROS.2013.6696487.

[15] S. Lovegrove and A.J. Davison. Real-time spherical mosaicing using whole image alignment. In *European Conference on Computer Vision*, pages 73–86, 2010.

[16] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[17] C. Mei, S. Benhimane, E. Malis, and P. Rives. Efficient Homography-based Tracking and 3-D Reconstruction for Single Viewpoint Sensors. *IEEE Transactions on Robotics*, 24:1352 – 1364, 2008.

[18] K.B. Peterson, M.S. Pederson, K.S. Larsen, L. Christiansen, K. Hansen, L. He, L. Thibaut, M. Barao, S. Hattinger, V. Sima, and W. The. The matrix cookbook. Technical report, 2006.

[19] H.Y. Shum and R. Szeliski. Construction and refinement of panoramic mosaics with global and local alignment. *Intl. Journal of Computer Vision*, 36(2):101–130, 2000.

[20] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.