Derivation of Jacobian Matrix in: Simultaneous Localization and Calibration

Qian-Yi Zhou* Vladlen Koltun†

Abstract

This supplementary document includes the complete derivation of Jacobian matrix in Section 2.2 of our CVPR 2014 paper "Simultaneous Localization and Calibration: Self-Calibration of Consumer Depth Cameras" [2].

1. Optimization Objective

As stated in the paper, the energy function $E(\mathbf{T}, C)$ is a function of the camera pose trajectory $\mathbf{T} = \{\mathbf{T}_i\}$ and a calibration function $C(\cdot)$. For every frame i, $\mathbf{T}_i$ is a rigid transformation that maps the depth image D_i from its local coordinate frame to the global world frame. C is a trilinear interpolation function of a control lattice $\mathbf{V} = \{\mathbf{v}_l\} \subset \mathbb{P}^3$:

$$C(\mathbf{p}) = \sum_l \gamma_l(\mathbf{p}) C(\mathbf{v}_l), \tag{1}$$

where $\{\gamma_l(\mathbf{p})\}$ are trilinear interpolation coefficients. They are computed once for all input points and remain constant henceforth.

The energy function is defined as:

$$E(\mathbf{T}, C) = E_a(\mathbf{T}, C) + \lambda E_r(C), \tag{2}$$

where $E_a(\mathbf{T}, C)$ is the alignment term that measures the point-to-plane distance between corresponding point pairs:

$$E_a(\mathbf{T}, C) = \sum_{i,j} \sum_{(\mathbf{p},\mathbf{q}) \in \mathcal{K}_{i,j}} \left((\mathbf{p}' - \mathbf{q}') \cdot \mathbf{n}'_{\mathbf{p}} \right)^2. \tag{3}$$

Here the points $\mathbf{p}'$ and $\mathbf{q}'$ are points $\mathbf{p}$ and $\mathbf{q}$, transformed from their local coordinate frames to the world frame. We ignore the distortion effect on normals and directly apply the rigid transformation $\mathbf{T}_i$ on $\mathbf{n}_{\mathbf{p}}$ to obtain $\mathbf{n}'_{\mathbf{p}}$:

$$\begin{aligned} \mathbf{p}' &= \mathbf{T}_i C(\mathbf{p}), && (4)\\ \mathbf{q}' &= \mathbf{T}_j C(\mathbf{q}), && (5)\\ \mathbf{n}'_{\mathbf{p}} &\approx \mathbf{T}_i \mathbf{n}_{\mathbf{p}}. && (6) \end{aligned}$$

For clarity, let $\mathbf{e} = (\mathbf{p}, \mathbf{q})$ denote the corresponding point pair, Equation (3) can be written as:

$$E_a(\mathbf{T}, C) = \sum_{i,j} \sum_{\mathbf{e} \in \mathcal{K}_{i,j}} \left(r_a^{\mathbf{e}} \right)^2, \tag{7}$$

where

$$\begin{aligned} r_a^{\mathbf{e}} &= (\mathbf{p}' - \mathbf{q}') \cdot \mathbf{n}'_{\mathbf{p}} && (8)\\ &= \left(\mathbf{T}_i C(\mathbf{p}) - \mathbf{T}_j C(\mathbf{q}) \right)^\top \mathbf{T}_i \mathbf{n}_{\mathbf{p}}. && (9) \end{aligned}$$

The regularization energy term $E_r(C)$ is inspired by elasticity theory [1, 3]:

$$E_r(C) = \sum_{\mathbf{v} \in \mathbf{V}} \sum_{\mathbf{u} \in \mathcal{N}_{\mathbf{v}}} \| C(\mathbf{u}) - {}^{\mathbf{v}}\mathbf{R}_{C(\mathbf{v})} \mathbf{u} \|^2, \tag{10}$$

where $\mathcal{N}(\mathbf{v})$ is the set of neighbors of $\mathbf{v}$ in $\mathbf{V}$ and the transform ${}^{\mathbf{v}}\mathbf{R}_{C(\mathbf{v})} \in SE(3)$ is a local linearization of C at $\mathbf{v}$. Please refer to Section 3.2 of our ICCV 2013 paper [3] for detailed derivation of the regularization energy term.

2. Gauss-Newton Method

We minimize $E(\mathbf{T}, C)$ using the Gauss-Newton method. Let $\mathbf{x}$ be the vector of variables that includes all the parameters of $\mathbf{T}$ and C. The calibration function C is parameterized by the calibrated position $C(\mathbf{v})$ of each lattice point $\mathbf{v}$. The transformations $\mathbf{T}_i$ are parameterized by its local linearization during iteration, as described below.

In iteration 0 the variables are initialized with the vector $\mathbf{x}^0$ that includes the camera poses from an initial rigid alignment of the input images $\{\mathcal{D}_i\}$ and a stationary function C that maps all the lattice points to themselves. In each subsequent iteration $k+1$, for $k \geq 0$, we locally linearize $\mathbf{T}_i$ around $\mathbf{T}_i^k$. Specifically, we parameterize $\mathbf{T}_i$ by a 6-vector $\xi_i = (\alpha_i, \beta_i, \gamma_i, a_i, b_i, c_i)$ that represents an incremental transformation relative to $\mathbf{T}_i^k$. Here (a_i, b_i, c_i) is a translation vector, which we will denote by $\mathbf{t}_i$, and $(\alpha_i, \beta_i, \gamma_i)$ can be interpreted as angular velocity, denoted by ω_i. $\mathbf{T}_i$ is approximated by a linear function of ξ_i:

$$\mathbf{T}_i \approx \begin{pmatrix} 1 & -\gamma_i & \beta_i & a_i \\ \gamma_i & 1 & -\alpha_i & b_i \\ -\beta_i & \alpha_i & 1 & c_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{T}_i^k. \tag{11}$$

*Stanford University
†Adobe Research

The full parameter vector is updated in iteration $k+1$ as follows[1]:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}. \tag{12}$$

Here $\Delta\mathbf{x}$ is a vector that collates $\{\xi_i\}$ and $\{\Delta C(\mathbf{v})\}$. It is computed by solving the following linear system:

$$\mathbf{J}_\mathbf{r}^\top \mathbf{J}_\mathbf{r} \Delta\mathbf{x} = -\mathbf{J}_\mathbf{r}^\top \mathbf{r}. \tag{13}$$

Here $\mathbf{r} = \mathbf{r}(\mathbf{x})$ is the residual vector that collects $\{r_a^\mathbf{e}\}$ and $\mathbf{J}_\mathbf{r} = \mathbf{J}_\mathbf{r}(\mathbf{x})$ is its Jacobian, both evaluated at $\mathbf{x}^k$. The detailed derivation of the Jacobian matrix will be given in Section 3.

Once the adjustment $\Delta\mathbf{x}$ is computed, $C^k(\mathbf{v})$ is straightforwardly updated by applying the additive increment $\Delta C(\mathbf{v})$:

$$C^{k+1}(\mathbf{v}) = C^k(\mathbf{v}) + \Delta C(\mathbf{v}). \tag{14}$$

To update the transformations, we apply Equation (11) and then map the transformation matrices back into the $SE(3)$ group, i.e.,

$$\begin{aligned}\mathbf{T}_i^{k+1} \;=\; & \begin{bmatrix} 1 & 0 & 0 & x_i \\ 0 & 1 & 0 & y_i \\ 0 & 0 & 1 & z_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ & \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ & \cdot \begin{bmatrix} \cos\beta_i & 0 & \sin\beta_i & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta_i & 0 & \cos\beta_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ & \cdot \begin{bmatrix} \cos\gamma_i & -\sin\gamma_i & 0 & 0 \\ \sin\gamma_i & \cos\gamma_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \mathbf{T}_i^k. \end{aligned} \tag{15}$$

In the next iteration, we re-parameterize $\mathbf{T}_i$ around $\mathbf{T}_i^{k+1}$ and repeat.

3. Derivation of Jacobian Matrix

The partial derivative of $\{r_a^\mathbf{e}\}$ with respect to $C(\mathbf{v}_l)$ is straightforward using Equation (9) and (1):

$$\frac{\partial r_a^\mathbf{e}}{\partial C(\mathbf{v}_l)} \;=\; \left(\gamma_l(\mathbf{p})\mathbf{T}_i^k - \gamma_l(\mathbf{q})\mathbf{T}_j^k\right)^\top \mathbf{T}_i^k \mathbf{n}_\mathbf{p}. \tag{16}$$

To derive the partial derivative of $\{r_a^\mathbf{e}\}$ with respect to ξ_i, we first derive the partial derivative of $\mathbf{f}(\xi_i, \mathbf{u}) = \mathbf{T}_i\mathbf{u}$ and $\mathbf{g}(\xi_i, \mathbf{n}) = \mathbf{T}_i\mathbf{n}$, with respect to ξ_i. Here $\mathbf{u}$ is the homogeneous coordinate of any point and $\mathbf{n}$ is the homogeneous coordinate of any unit direction vector (normal). With Equation (11), we have:

$$\begin{aligned}\mathbf{f}(\xi_i, \mathbf{u}) = \mathbf{T}_i\mathbf{u} \;&\approx\; \mathbf{T}_i^k\mathbf{u} + \omega_i \times \mathbf{T}_i^k\mathbf{u} + \mathbf{t}_i, && (17)\\ \mathbf{g}(\xi_i, \mathbf{n}) = \mathbf{T}_i\mathbf{n} \;&\approx\; \mathbf{T}_i^k\mathbf{n} + \omega_i \times \mathbf{T}_i^k\mathbf{n}. && (18)\end{aligned}$$

Let $\left[\mathbf{T}_i^k\mathbf{u}\right]_\times$ and $\left[\mathbf{T}_i^k\mathbf{n}\right]_\times$ be the skew-symmetric matrices form of the cross product with $\mathbf{T}_i^k\mathbf{u}$ and $\mathbf{T}_i^k\mathbf{n}$. The former equations can be written in the matrix multiplication form:

$$\begin{aligned}\mathbf{f}(\xi_i, \mathbf{u}) \;&\approx\; \mathbf{T}_i^k\mathbf{u} + \left[-\left[\mathbf{T}_i^k\mathbf{u}\right]_\times \middle| \mathbf{I}\right]\xi_i, && (19)\\ \mathbf{g}(\xi_i, \mathbf{n}) \;&\approx\; \mathbf{T}_i^k\mathbf{n} + \left[-\left[\mathbf{T}_i^k\mathbf{n}\right]_\times \middle| \mathbf{0}\right]\xi_i, && (20)\end{aligned}$$

where $\mathbf{I}$ is the 3×3 identity matrix and $\mathbf{0}$ is the 3×3 zero matrix. Their Jacobian matrices with respect to ξ_i are:

$$\begin{aligned}\frac{\partial\mathbf{f}}{\partial\xi_i} \;&\approx\; \left[-\left[\mathbf{T}_i^k\mathbf{u}\right]_\times \middle| \mathbf{I}\right], && (21)\\ \frac{\partial\mathbf{g}}{\partial\xi_i} \;&\approx\; \left[-\left[\mathbf{T}_i^k\mathbf{n}\right]_\times \middle| \mathbf{0}\right]. && (22)\end{aligned}$$

Using this result on Equation (4), (5), and (6), we have:

$$\begin{aligned}&\frac{\partial\mathbf{p}'}{\partial\xi_i} \approx \left[-\left[\mathbf{T}_i^k C^k(\mathbf{p})\right]_\times \middle| \mathbf{I}\right], \quad \frac{\partial\mathbf{p}'}{\partial\xi_j} = \mathbf{0}, && (23)\\ &\frac{\partial\mathbf{q}'}{\partial\xi_i} = \mathbf{0}, \quad \frac{\partial\mathbf{q}'}{\partial\xi_j} \approx \left[-\left[\mathbf{T}_i^k C^k(\mathbf{q})\right]_\times \middle| \mathbf{I}\right], && (24)\\ &\frac{\partial\mathbf{n}'_\mathbf{p}}{\partial\xi_i} \approx \left[-\left[\mathbf{T}_i^k \mathbf{n}_\mathbf{p}\right]_\times \middle| \mathbf{0}\right], \quad \frac{\partial\mathbf{n}'_\mathbf{p}}{\partial\xi_j} = \mathbf{0}. && (25)\end{aligned}$$

Using product rule, we have:

$$\begin{aligned}\frac{\partial r_a^\mathbf{e}}{\partial\xi_i} \;&=\; (\mathbf{p}' - \mathbf{q}')^\top\left(\frac{\partial\mathbf{n}'_\mathbf{p}}{\partial\xi_i}\right) + (\mathbf{n}'_\mathbf{p})^\top\left(\frac{\partial\mathbf{p}'}{\partial\xi_i} - \frac{\partial\mathbf{q}'}{\partial\xi_i}\right)\\ &\approx\; \left[\mathbf{T}_i^k\mathbf{n}_\mathbf{p} \times \left(\mathbf{T}_i^k C^k(\mathbf{p}) - \mathbf{T}_j^k C^k(\mathbf{q})\right) \middle| \mathbf{0}\right]\\ &\quad + \left[\mathbf{T}_i^k C^k(\mathbf{p}) \times \mathbf{T}_i^k\mathbf{n}_\mathbf{p} \middle| \mathbf{T}_i^k\mathbf{n}_\mathbf{p}\right]\\ &=\; \left[\mathbf{T}_j^k C^k(\mathbf{q}) \times \mathbf{T}_i^k\mathbf{n}_\mathbf{p} \middle| \mathbf{T}_i^k\mathbf{n}_\mathbf{p}\right], && (26)\\ \frac{\partial r_a^\mathbf{e}}{\partial\xi_j} \;&=\; (\mathbf{p}' - \mathbf{q}')^\top\left(\frac{\partial\mathbf{n}'_\mathbf{p}}{\partial\xi_j}\right) + (\mathbf{n}'_\mathbf{p})^\top\left(\frac{\partial\mathbf{p}'}{\partial\xi_j} - \frac{\partial\mathbf{q}'}{\partial\xi_j}\right)\\ &\approx\; \left[\mathbf{T}_i^k\mathbf{n}_\mathbf{p} \times \mathbf{T}_j^k C^k(\mathbf{q}) \middle| -\mathbf{T}_i^k\mathbf{n}_\mathbf{p}\right]. && (27)\end{aligned}$$

References

[1] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3), 2007.

[2] Q.-Y. Zhou and V. Koltun. Simultaneous localization and calibration: Self-calibration of consumer depth cameras. In *CVPR*, 2014.

[3] Q.-Y. Zhou, S. Miller, and V. Koltun. Elastic fragments for dense scene reconstruction. In *ICCV*, 2013.

[1] Since $\{\xi_i\}$ are local linearization around $\mathbf{T}_i^k$, the corresponding part in $\mathbf{x}^k$ is $\mathbf{0}$.