Autonomous Robotics: lecture notes

Karel Zimmermann

1 Lidar, camera and their mutual calibration

In this section, we formulate lidar-lidar calibration and camera-lidar calibration from known correspondences as optimization problems a derive closed-form solution.

1.1 Lidar-lidar calibration

Lidar is a sensor, which repeatedly measure the depth in its field-of-view using time-of-flight principle to provide 3D pointclouds. When two lidars are available, they both provide measurements relative to their own coordinate frame. To transform measurement from one lidar to another, unknown transformation $\mathsf{g} \in SO(3)$ needs to be estimated.

Pairs of 3D points from pointclouds $\mathbf{p}_i = (p_x, p_y, p_z)^\top$ (from the first lidar) and $\mathbf{q}_i = (q_x, q_y, z)^\top$ (from the second lidar), which both correspond to the same physical point in the real world are called 3D-3D correspondences. The Euclidean transformation g between lidars, aligns pairs from 3D-3D correspondence to the same point:

$$\mathbf{q}_i = \mathbf{R}\mathbf{p}_i + \mathbf{t} \quad \forall_{i=1\ldots N}$$

Since measurements contains noise, the set of equations does not have an exact solution with respect to $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathcal{R}^3$. Assuming Gaussian noise and i.i.d. measurements (see Appendix A for the derivation), we estimate the unknown parameters $\mathbf{R}^*, \mathbf{t}^*$ as follows:

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 \tag{1}$$

This problem has the following closed-form solution:

$$\begin{aligned} \mathbf{R}^* &= \mathbf{V}\mathbf{U}^\top, \\ \mathbf{t}^* &= \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}, \end{aligned}$$

where $\mathbf{U}\mathbf{S}\mathbf{V}^\top = \mathbf{H}$ is SVD decomposition of 3×3 matrix $\mathbf{H} = \sum_i \mathbf{p}_i'\mathbf{q}_i'^\top$ with

$$\mathbf{p}_i' = \mathbf{p}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{p}_i}_{\widetilde{\mathbf{p}}}, \quad \mathbf{q}_i' = \mathbf{q}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{q}_i}_{\widetilde{\mathbf{q}}}$$

Proof: We will first show that the problem (1) splits into two sub-problems, then particular solutions for $\mathbf{R}^*$ and $\mathbf{t}^*$ are derived.

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 = \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}(\mathbf{p}'_i + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}'_i - \widetilde{\mathbf{q}}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}')^\top (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}') =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \underbrace{\sum_i 2(\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i)\mathbf{t}'}_{=0} + \|\mathbf{t}'\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\mathbf{t}'\|_2^2 \tag{2}$$

Minimum of $\|\mathbf{t}'\|_2^2$ is zero. We can achieve this minimum by choosing

$$\mathbf{t} = \widetilde{\mathbf{q}} - \mathbf{R}\widetilde{\mathbf{p}}.$$

Since the first term does not depend on $\mathbf{t}$, this choice is the optimal translation $\mathbf{t}*$. Substituting this into Eq. (2), problem reduces to

$$\arg\min_{\mathbf{R}\in SO(3)} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 = \arg\min_{\mathbf{R}\in SO(3)} \sum_i \mathbf{p}'^\top_i \mathbf{p}'_i - 2\mathbf{q}'^\top_i \mathbf{R}\mathbf{p}'_i + \mathbf{q}'^\top_i \mathbf{q}'_i =$$

$$= \arg\max_{\mathbf{R}\in SO(3)} \sum_i \mathbf{q}'^\top_i \mathbf{R}\mathbf{p}'_i = \arg\max_{\mathbf{R}\in SO(3)} \operatorname{trace}\{\sum_i \mathbf{R}\mathbf{p}'_i \mathbf{q}'^\top_i\} = \arg\max_{\mathbf{R}\in SO(3)} \operatorname{trace}\{\mathbf{R}\mathbf{H}\} = \mathbf{V}\mathbf{U}^\top,$$

where $\mathbf{U}\mathbf{S}\mathbf{V}^\top = \mathbf{H}$ is SVD decomposition of $\mathbf{H}$. Proof of the last equality follows from showing, that substitution of $\mathbf{R} = \mathbf{V}\mathbf{U}^\top$ yields value of criterion function, which is better than any other rotation.

$$\operatorname{trace}\{\mathbf{R}\mathbf{H}\} = \operatorname{trace}\{\mathbf{V}\mathbf{U}^\top\mathbf{U}\mathbf{S}\mathbf{V}^\top\} = \operatorname{trace}\{\mathbf{V}\mathbf{S}\mathbf{V}^\top\}$$

$$= \operatorname{trace}\{\underbrace{(\mathbf{V}\sqrt{\mathbf{S}})}_{\mathbf{A}}\underbrace{(\sqrt{\mathbf{S}}\mathbf{V})^\top}_{\mathbf{A}^\top}\} \geq \operatorname{trace}(\mathbf{R}\mathbf{A})\mathbf{A}^\top$$

□

Publishing static transformation between two coordinate frames in ROS/python:

```
broadcaster = tf2_ros.StaticTransformBroadcaster()
transformation = geometry_msgs.msg.TransformStamped()
```

```
# fill translation and rotation into transform
broadcaster.sendTransform(transformation)
```

See detailed description here:
http://wiki.ros.org/action/fullsearch/tf2/Tutorials

1.2 Camera-lidar calibration

Camera is sensor, which repeatedly record visual images in its field of view. Projection of 3D point $\mathbf{p} \in \mathcal{R}^3$ in the camera coordinate frame on 2D point $\mathbf{u} \in \mathcal{R}^2$ in the image plane is estimated as follows:

$$\lambda\overline{\mathbf{u}} = \begin{bmatrix} s_x & s_o & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p} = \mathbf{K}\mathbf{p},$$

where f is the focal length, o_x, o_y is a center of image plane and s_x, s_y, s_o are scalings, $\mathbf{K} \in \mathcal{R}^{3\times 3}$ is regular matrix. All these scalar variables are called intrinsic parameters of the camera. $\overline{\mathbf{u}}$ are homogeneous coordinates of point $\mathbf{u}$. Set of 3D points $\{\lambda\mathbf{K}^{-1}\overline{\mathbf{u}} \mid \lambda \in \mathcal{R}\}$, which all project to the same pixel $\mathbf{u}$ is called a ray.
Let us have a 3D point $\mathbf{q}$ in the coordinate frame of other sensor (e.g. lidar). Projection of this point to the camera image plane consists of two steps: (i) transformation from the lidar coordinate frame to the camera coordinate frame and (ii) projection of the point in camera coordinate frame on the image plane. Resulting concatenated transformation is

$$\lambda\overline{\mathbf{u}} = \mathbf{K}(\mathbf{R}\mathbf{q} + \mathbf{t}) = \mathbf{K}[\mathbf{R}\ \mathbf{t}]\overline{\mathbf{q}}$$

Pairs of 2D point from camera image plane $\mathbf{u}_i = (u_x, u_y)^\top$ and 3D points from lidar coordinate frame $\mathbf{q}_i = (q_x, q_y, q_z)^\top$, which corresponds to the same point in the real world are called 2D-3D correspondences.
Camera calibration from 2D-3D correspondences is the search for matrix $\mathbf{P} = \mathbf{K}[\mathbf{R}\ \mathbf{t}]$, which aligns 2D-3D correspondences on each other $\lambda\mathbf{u}^i = \mathbf{P}\mathbf{q}^i$. Scalar value λ can be eliminated and the expression translates to homogeneous set of linear equations.

$$\underbrace{\begin{bmatrix} -\overline{\mathbf{q}}_i^\top & \mathbf{0}^\top & u_{xi}\overline{\mathbf{q}}_i^\top \\ \mathbf{0}^\top & -\overline{\mathbf{q}}_i^\top & u_{yi}\overline{\mathbf{q}}_i^\top \end{bmatrix}}_{\mathbf{A}_{[2\times 12]}} \underbrace{\begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}}_{\mathbf{p}_{[12\times 1]}} = \mathbf{0}_{[2\times 1]}$$

Since measurements contains noise, the set of equations does not have an exact nontrivial solution with respect to $\mathbf{p} \in \mathcal{R}^{12}$. Assuming i.i.d. measurements and Gaussian noise on the right-hand side of the homogeneous system (which is typically wrong and value normalization have to be done prior to the calibration), the ML estimate is as follows:

$$\mathbf{p}^* = \text{argmin}\|\mathbf{A}\mathbf{p}\| \quad \text{subject to} \quad \|\mathbf{p}\| = 1 \tag{3}$$

This problem has closed-form solution, which is equal to the eigen-vector of $\mathtt{A}^\top\mathtt{A}$ with the smallest corresponding eigen-value (MATLAB tip: `[W D]=EIG(A'*A); p=W(:,1)`, Python tip: `numpy.linalg.eig`). It is the same as the singular-vector of `A` which corresponds to the smallest singular-value (MATLAB tip: `[U S V]=SVD(A); p=V(:,end)`, Python tip: `numpy.linalg.svd`). For the sake of completeness, derivation of this solution is provided in the next paragraph.

Solution $\mathbf{p}^*$ is reshaped into matrix $\mathbf{P}$. Since scale does not matter, we choose normalize matrix $\mathbf{P}$ as follows $\mathbf{P} := \mathbf{P}/\|[\mathbf{p}_{31}, \mathbf{p}_{32}, \mathbf{p}_{33}]\|$. Eventually, matrix

$$\mathbf{P} = [\underbrace{\mathbf{KR}}_{\mathbf{B}} \;\; \underbrace{\mathbf{Kt}}_{\mathbf{c}}] = [\mathbf{B} \;\; \mathbf{c}],$$

is decomposed on $\mathbf{K}, \mathbf{R}, \mathbf{t}$ using QR decomposition of $\mathbf{B}$ as follows:

$$\mathbf{KR} = \mathbf{B} \tag{4}$$

$$\mathbf{t} = \mathbf{K}^{-1}\mathbf{c} \tag{5}$$

Proof:

We solve problem (3) by introducing Lagrange function

$$L(\mathbf{p}, \lambda) = \|\mathtt{A}\mathbf{p}\| + \lambda(1 - \|\mathbf{p}\|) = \tag{6}$$

$$= \mathbf{p}^\top\mathtt{A}^\top\mathtt{A}\mathbf{p} + \lambda(1 - \mathbf{p}^\top\mathbf{p}). \tag{7}$$

Critical points (i.e. points in which local extrema can be achieved) of the Lagrange function are found by equaling derivatives to zero

$$\frac{\partial L(\mathbf{p}, \lambda)}{\partial \mathbf{p}} = 2\mathtt{A}^\top\mathtt{A}\mathbf{p} - 2\lambda\mathbf{p} = \mathbf{0} \tag{8}$$

$$\frac{\partial L(\mathbf{p}, \lambda)}{\partial \lambda} = 1 - \mathbf{p}^\top\mathbf{p} = \mathbf{0}. \tag{9}$$

Equation (8) is simply rewritten as the characteristic equation

$$(\mathtt{A}^\top\mathtt{A} - \lambda\mathtt{I})\mathbf{p} = \mathbf{0}, \tag{10}$$

of $\mathtt{A}^\top\mathtt{A}$. Therefore, every eigen-vector $\mathbf{p}$ of $\mathtt{A}^\top\mathtt{A}$ with corresponding eigen-values λ is critical point and the one which yields the smallest criterion value $\|\mathtt{A}\mathbf{p}\|$ of problem (3) is chosen. Using equation (10) and the constraint (9), it is shown that the criterion values in critical points are equal to corresponding eigen-values:

$$\|\mathtt{A}\mathbf{p}\| = \mathbf{p}^\top\mathtt{A}^\top\mathtt{A}\mathbf{p} = \mathbf{p}^\top\lambda\mathbf{p} = \lambda\mathbf{p}^\top\mathbf{p} = \lambda\|\mathbf{p}\| = \lambda.$$

Therefore the solution of problem (3) is the eigen-vector of $\mathtt{A}^\top\mathtt{A}$ with the smallest eigen-value.

TF message:

2 Motion estimation from camera and lidar measurements

Many robots are equipped by the camera and the lidar. We assume that camera and lidar are calibrated and depth measurements are transformed to the camera coordinate frame. Consequently, depth of some pixels in image plane is also known.
The robot moved between two consecutive time instances time 1 and time 2, which caused change of the camera coordinate frame. The task is to estimate the relative motion of the robot between these two time instances.
Let us assume that in both of these times, camera images were captured and a feature detector such as ORB, SIFT, LIFT estimated 2D-2D correspondences (i.e. pixel coordinates of the same scene point in these two images). If also lidar measurements are available for the 2D-2D correspondence, the 3D position $\mathbf{x}_2 = (x_2, y_2, z_2)^\top$ of 3D point in time 2 and 3D position $\mathbf{x}_1 = (x_1, y_1, z_1)^\top$ of 3D point in time 1 are known.
Since the scene is assumed to be rigid, the motion corresponds to the transformation from Special Euclidean group $\mathcal{SE}(3)$. This transformation is uniquely determined by rotation $\mathbf{R} \in \mathcal{SO}(3)$ and translation $\mathbf{t} \in \mathcal{R}^3$, which aligns corresponding points on each other:

$$\mathbf{x}_1 = \mathbf{R}\mathbf{x}_2 + \mathbf{t}$$

2.1 3D-3D correspondences

If depth of both points $\mathbf{x}_1$ and $\mathbf{x}_2$ is known, we have 3D-3D correspondence, which implies the three following constraints on the motion:

$$f_{33}(\mathbf{R}, \mathbf{t}) = \mathbf{x}_1 - (\mathbf{R}\mathbf{x}_2 + \mathbf{t}) = \mathbf{0}$$

2.2 2D-3D correspondences

If depth of $\mathbf{x}_1$ is unknown, we have 2D-3D correspondence. The unknown depth d_1, is eliminated from the three equations as follows:

$$d_1 \hat{\mathbf{x}}_1 = \mathbf{R}\mathbf{x}_2 + \mathbf{t}$$

$$\begin{aligned} d_1 \hat{x}_1 &= \mathbf{R}_1^\top x_2 + t_1 \\ d_1 \hat{y}_1 &= \mathbf{R}_2^\top y_2 + t_2 \\ d_1 \hat{z}_1 &= \mathbf{R}_3^\top z_2 + t_3 \end{aligned}$$

$$\begin{aligned} \hat{x}_1(\mathbf{R}_3^\top z_2 + t_3) - \hat{z}_1(\mathbf{R}_1^\top x_2 + t_1) &= 0 \\ \hat{y}_1(\mathbf{R}_3^\top z_2 + t_3) - \hat{z}_1(\mathbf{R}_2^\top x_2 + t_2) &= 0 \end{aligned}$$

Consequently, each 2D-3D correspondences yields two motion constraints

$$f_23(\mathbf{R}, \mathbf{t}) = 0$$

2.3 2D-2D correspondences

If depth of both points from the 2D-2D correspondence is unknown, we have only the 2D-2D correspondence. The unknown depths d_1, d_2 are eliminated from the three motion equations

$$d_1\hat{x}_1 = \mathbf{R}_1^\top d_2\hat{x}_2 + t_1 \tag{11}$$
$$d_1\hat{y}_1 = \mathbf{R}_2^\top d_2\hat{y}_2 + t_2 \tag{12}$$
$$d_1\hat{z}_1 = \mathbf{R}_3^\top d_2\hat{z}_2 + t_3 \tag{13}$$

Consequently, each 2D-2D correspondences yields only one motion constraint:

$$f_{22}(\mathbf{R}, \mathbf{t}) = \begin{bmatrix} -\hat{y}_1 t_3 + \hat{z}_1 t_2 \\ \hat{x}_1 t_3 + \hat{z}_1 t_1 \\ -\hat{x}_1 t_2 + \hat{y}_1 t_1 \end{bmatrix} \mathbf{R}\,\mathbf{x}_2 = 0$$

Usually mixed correspondences are available due to different framerates of sensors and missing 3D measurements on reflective surfaces. Consequently, all four types of motion constraints f_33, f_23, f_32, f_22 appears in the motion estimation process.
Since measurements are noisy, the set of equations is overdetermined. Assuming the Gaussian noise and i.i.d. samples (which is typically not the case in reality), we search for the maximum likelihood estimate of rotation and translation which minimize L_2 norm between correspondences.

$$(\mathbf{R}^*, \mathbf{t}^*) = \underset{\mathbf{R}\in\mathcal{SO}(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \left\| \begin{matrix} \sum_i f_{33}^i(\mathbf{R}, \mathbf{t}) \\ \sum_i f_{23}^i(\mathbf{R}, \mathbf{t}) \\ \sum_i f_{32}^i(\mathbf{R}, \mathbf{t}) \\ \sum_i f_{22}^i(\mathbf{R}, \mathbf{t}) \end{matrix} \right\|_2^2,$$

where i denotes correspondence index. There is no closed-form solution to this problem, therefore Levenberg-Marquardt method for nonlinear least-squares optimization is typically used.

Appendix A: MAP and ML estimate

We are given model $y = p(y|\mathbf{x}, \mathbf{w})$ with parameters $\mathbf{w}$, which estimates dependent variable y from a given i.i.d. measured data $\mathcal{D} = \{\mathbf{x}_1, y_1 \ldots \mathbf{x}_N, y_N\}$ searches for the most probable parameters $\mathbf{w}$ given the measured data $\mathcal{D}$. We search for the most probable parameters $\mathbf{w}$ of the probability distribution, given measured data $\mathcal{D}$.

$$\arg\max_{\mathbf{w}} p(\mathbf{w}|\mathcal{D}) = \arg\max_{\mathbf{w}} \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} =$$

$$= \arg\max_{\mathbf{w}} p(\mathcal{D}|\mathbf{w})p(\mathbf{w}) = \arg\max_{\mathbf{w}} p(\mathbf{x}_1, y_1 \dots \mathbf{x}_N, y_N|\mathbf{w})p(\mathbf{w}) =$$

$$= \arg\max_{\mathbf{w}} \left(\prod_i p(\mathbf{x}_i, y_i|\mathbf{w})\right) p(\mathbf{w}) = \arg\max_{\mathbf{w}} \left(\prod_i p(y_i|\mathbf{x}_i, \mathbf{w})p(\mathbf{x}_i)\right) p(\mathbf{w}) =$$

$$= \arg\max_{\mathbf{w}} \left(\sum_i \log(p(y_i|\mathbf{x}_i, \mathbf{w})) + \log p(\mathbf{x}_i)\right) + \log p(\mathbf{w})$$

$$= \arg\max_{\mathbf{w}} \left(\sum_i \log(p(y_i|\mathbf{x}_i, \mathbf{w}))\right) + \log p(\mathbf{w}) =$$

$$= \arg\min_{\mathbf{w}} \Big(\sum_i \underbrace{-\log(p(y_i|\mathbf{x}_i, \mathbf{w}))}_{\text{loss}}\Big) + \Big(\underbrace{-\log p(\mathbf{w})}_{\text{regularizer } R(\mathbf{w})}\Big)$$

This is called Maximum A Posteriori (MAP) estimate of parameters $\mathbf{w}$. Especially for no aprior knowledge $p(\mathbf{w}) = \text{const.}$, regularizer equals to zero and the we obtain Maximum Likelyhood (ML) estimate of parameters $\mathbf{w}$.

$\mathbf{L_2}$-loss:
For Gaussian likelihood $p(y_i|\mathbf{x}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(f(\mathbf{x}_i,\mathbf{w})-y_i)^2}{2\sigma^2}\right)$, the ML estimate of $\mathbf{w}$ is minimization of L_2 regression loss

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_i \left(f(\mathbf{x}_i, \mathbf{w}) - y_i\right)^2$$

Logistic loss:
For dichotomy classification problem with

$$p(y|\mathbf{x}, \mathbf{w}) = \begin{cases} \sigma(f(\mathbf{x}, \mathbf{w})) & y = +1 \\ 1 - \sigma(f(\mathbf{x}, \mathbf{w})) & y = -1 \end{cases},$$

the ML estimate of $\mathbf{w}$ is minimization of logistic loss

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_i \log\left[1 + \exp(-y_i\, f(\mathbf{x}_i, \mathbf{w}))\right]$$

Cross-entropy loss:
For multi-class classification problem with

$$p(y_i|\mathbf{x}_i, \mathbf{W}) = \begin{bmatrix} \exp(f(\mathbf{x}_i, \mathbf{w}_1)) \\ \exp(f(\mathbf{x}_i, \mathbf{w}_2)) \\ \exp(f(\mathbf{x}_i, \mathbf{w}_3)) \end{bmatrix} \;/\; \sum_k \exp(f(\mathbf{x}_i, \mathbf{w}_k) = \mathbf{s}(\mathbf{f}(\mathbf{x}_i, \mathbf{W})),$$

the ML estimate of $\mathbf{w}$ is minimization of the cross-entropy loss:

$$\mathtt{W}^* = \arg\min_{\mathtt{W}} \sum_i -\log \mathbf{s}_{y_i}(\mathbf{f}(\mathbf{x}_i, \mathtt{W})),$$

where $\mathbf{s}$ is called soft-max function.

Common regularizers:

For Gaussian prior on parameter distribution (we assume that parameters a normally, independently distributed around zero), we obtain L_2 regularizer:

$$p(\mathbf{w}) = \mathcal{N}_{\mathbf{w}}(\mathbf{0}, \lambda\mathbb{I}) \quad \Rightarrow \quad R(\mathbf{w}) = \mathbf{w}^\top \mathbf{w}$$

.