Skoltech

Skolkovo Institute of Science and Technology

L14: SE3 Optimization and Point Cloud Alignment

Perception in Robotics

Prof. Gonzalo Ferrer,

Skoltech, 11 March 2021

Summary of L13 on RBT

- The group of rotations SO(3)

$$SO(3) = \{R \in \mathbb{R}^{3\times 3} \,|\, RR^{\top} = I \,,\, det(R) = 1\}$$

- The group of Rigid Body Transformations SE(3)

$$SE(3) = \left\{ T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \,|\, R \in SO(3) \,,\, t \in \mathbb{R}^3 \right\}$$

- Group properties: Matrix multiplication, Associativity, Closure of T1*T2 \in SE(3), Identity element and Inverse.

- Uses of RBT:

  * transform a point,

  * transform a frame,

  * 3D pose.

Summary of L13 on RBT

- Lie Algebra for rotations, infinitesimal increments $\dot{R} = WR$
- Lie Algebra for RBT, infinitesimal increments

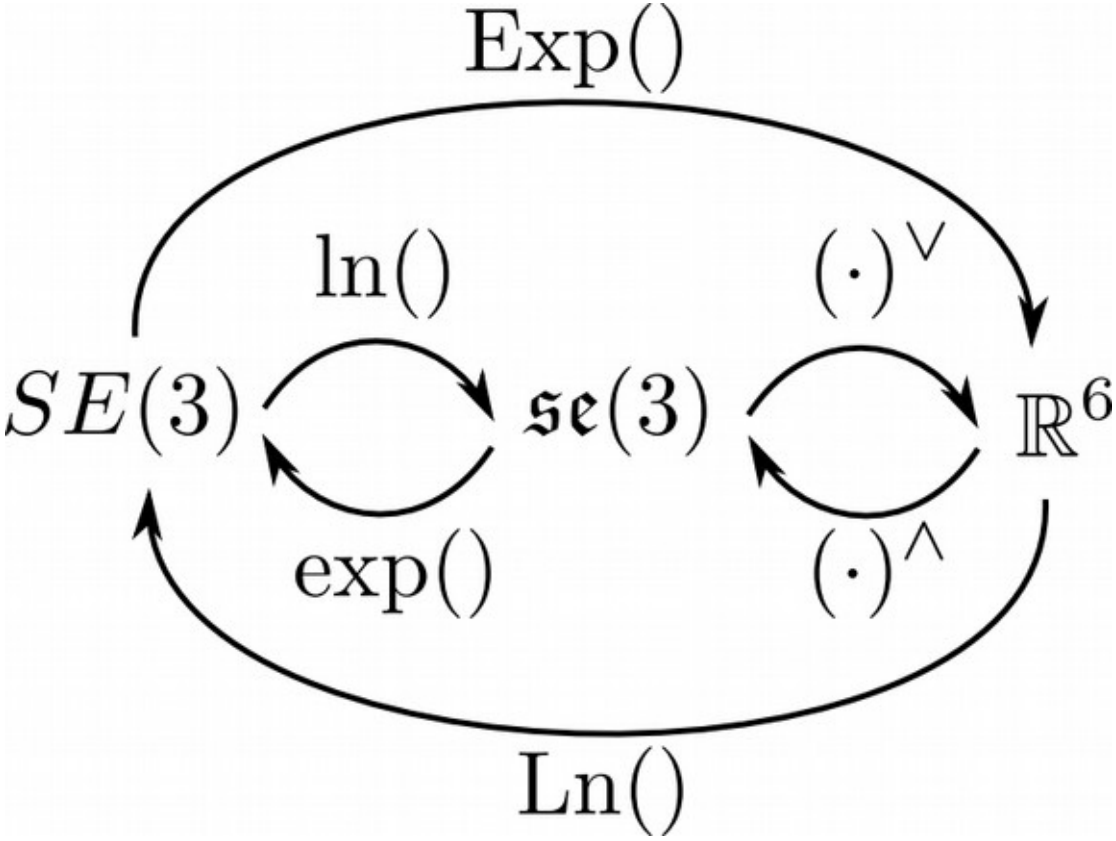
$$\mathcal{W}|_{t=1} = \xi^{\wedge} = \begin{bmatrix} \theta^{\wedge} & \rho \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\theta_3 & \theta_2 & \rho_1 \\ \theta_3 & 0 & -\theta_1 & \rho_2 \\ -\theta_2 & \theta_1 & 0 & \rho_3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

- The Exponential Map

$$\exp(\theta^{\wedge}) = I + \theta^{\wedge} + \frac{1}{2!}(\theta^{\wedge})^2 + \frac{1}{3!}(\theta^{\wedge})^3 + \ldots = \sum_{n}^{\infty} \frac{1}{n!}(\theta^{\wedge})^n.$$

- The logarithm

Summary of L13 on RBT



The graphic summarizes the different maps between the group of RBT and its tangent space.

Differentiating over SE(3)

Let's define a function: $f(T) : SE(3) \rightarrow \mathbb{R}^N$

The definition of derivative can't be applied directly to elements of the SE(3) group since the group operation is **matrix multiplication**:

$$\frac{\partial f(T)}{\partial T} = \lim_{\Delta T \to 0} \frac{f(T + \Delta T) - f(T)}{\Delta T}.$$

The solution to this is to define a Retraction Map, such that: $f_T(\xi) = f(Exp(\xi)T)$

Now the definition of directional derivative can be applied:

$$\left.\frac{\partial Exp(\xi)}{\partial \xi}\right|_{\xi=0} = \left.\frac{\partial}{\partial \xi}\left(I + \xi^{\wedge} + o(||\xi||^2)\right)\right|_{\xi=0} = \frac{\partial}{\partial \xi}\begin{bmatrix} 0 & -\theta_3 & \theta_2 & \rho_1 \\ \theta_3 & 0 & -\theta_1 & \rho_2 \\ \theta_2 & \theta_1 & 0 & \rho_3 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \mathbf{G}$$

Example of differentiation: Transforming a vector

Let's consider the function: $f(T) = T \cdot p$

Following the previous derivation, we apply the retraction map:

$$f_T(\xi) = Exp(\xi) \cdot T \cdot p$$

Now, we can define a derivative around the zero element in the retraction:

$$\left.\frac{\partial f_T}{\partial \xi}\right|_{\xi=0} = \left.\frac{\partial}{\partial \xi}\left(Exp(\xi) \cdot T \cdot p\right)\right|_{\xi=0} = \mathbf{G} \cdot T \cdot p$$

Example of differentiation: Transforming a vector

We can analyze the problem element by element of the xi vector

$$\left.\frac{\partial f_T}{\partial \xi_1}\right|_{\xi_1=0} = G(1)\cdot T\cdot p = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{G_1} \cdot T\cdot p$$

If we do for each G(i) and group terms, the results is the following:

$$\left.\frac{\partial f_T}{\partial \xi}\right|_{\xi=0} = [-(T\cdot p)^\wedge \mid I\,]_{3\times 6}$$

Optimization: Gradient-based and Gauss-Newton

Gradient based methods are the most straight-forward approach for to optimize and require a step update proportional to the gradient:

$$\Delta\xi = -\alpha \frac{\partial f_T}{\partial \xi}\bigg|_{\xi=0}$$

Second-order methods use information about the Hessian H to update the variables:

$$\Delta\xi = -\alpha (H)^{-1} \frac{\partial f_T}{\partial \xi}\bigg|_{\xi=0}$$

We have already discussed about Gauss-Newton in L10 by solving the LLSQ, and it is a variant of a second order method, where the Hessian is approximated (A'*A)

Optimization: Levenberg-Marquardt

The Levenberg-Marquardt method is a variant of second order methods, where we are adding a regularizing term:

$$\Delta\xi = -\alpha(H + \lambda diag(H))^{-1} \frac{\partial f_T}{\partial \xi}\bigg|_{\xi=0}$$

The improvement is more robustness in convergence, since the lambda parameter regularizes how much the update can change.

In the limit of lambda dominating, this is equivalent to gradient descent.

For lambda zero, this is equivalent to a second order method.

Updating an Element of the SE3 Group

Now that we have defined what is an update for a function, let's take a look at what the update of this new value should look like:

$$x' = x + \Delta x$$

This update is the standard update, however we know that the summation is not the group operation for SE3, and therefore, it is not guaranteed that the update will be in the group.

What we want, is to calculate an update from any of the methods proposed ealier: Gradient descen, GN, LM, and update the state variables as:

$$T' = \mathrm{Exp}(\Delta\xi) \cdot T$$

Random variables in SE(3)

For state estimation we are interested in obtaining PDF's of variables.

$$x + \eta, \qquad \eta \sim \mathcal{N}(0, R)$$

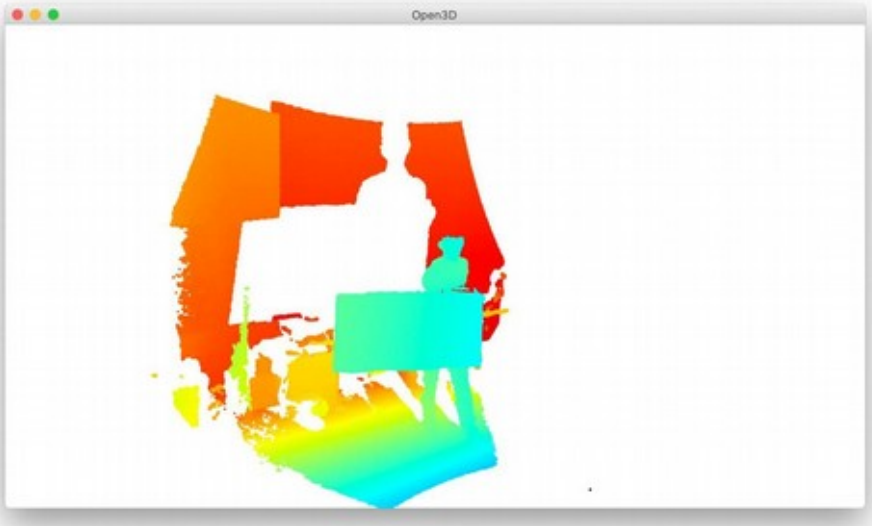
How can that be done is we have the group of RBT?

$$T = Exp(\delta)\bar{T}, \qquad \delta \sim \mathcal{N}(0, \Sigma_\delta)$$

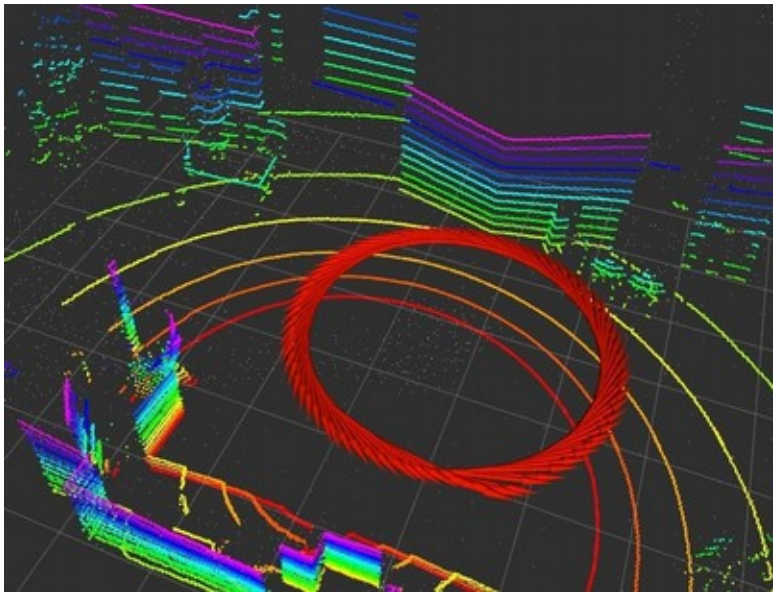
Point Clouds

A Point Cloud (PC) is a set of points. It can be either 3d or 2D $z_i = \{p_1, p_2, \ldots, p_k\}$

Points are observed from the sensor at pose x_i



Point cloud obtained from an RGB-D camera



PC from a Lidar VLP16

Point Cloud Alignment: Problem statement

The problem is to find the transformation (RBT) from a pair of poses x_i, x_j such that

$$^{j}T_i \cdot z_i = z_j$$

This problem is also known as Registration, Alignment, scanmatching (2D)

This definition considers two unrealistic conditions:

- We sample the same exact point in both point clouds, without noise
- Point correspondences are correct.

We detail algorithms for solving PC alignment in the next slides.

SVD method (Arun'87)

If we have enough points, we can solve the problem by minimizing the following cost function:

$$J({}^{j}T_i) = \sum_{k}^{K} ||{}^{j}T_i z_i^k - z_j^k||^2$$

The solution is decoupled in two:

Solving for Translation

$${}^{j}t_i + E\{z_i\} = E\{z_j\}$$

Solving for Rotation:

It computes the cross-covariance of points and applies and SVD method to obtain the rotation between them (details in the paper).

SVD method (Arun’87)

This method requires only 2 points in 2D and 3 points in 3D

Drawbacks:

- Sensitive to outliers.
- Requires known correspondences.

Optimization

We can simply solve the minimization problem directly:

$$J({}^{j}T_i) = \sum_{k}^{K} ||{}^{j}T_i z_i^k - z_j^k||^2$$

As discussed earlier, one can calculate the gradients of the function, and apply optimization methods, even for RBT.

$$J(T) = \sum_{k}^{K} ||r_k||^2 = \sum_{k}^{K} r_k^\top r_k$$

$$\left.\frac{\partial J_T}{\partial \xi}\right|_{\xi=0} = \sum_{k}^{K} \frac{\partial r_k}{\partial \xi}^\top r_k$$

RANSAC (Fischler-81)

Random Sample Consensus (RANSAC) is a general algorithm for robust parameter estimation, meaning that it rejects outliers from inliers from a large number of samples *P*.

The original algorithm can be described in the next steps:

1) Select a small random subset of samples *S*.
2) Fit a model *M* with *S*. For instance, Arun selecting 3 points
3) Based on this model *M*, classify which points from the large set *P* are within the same error tolerance as in *M* (inliers) into a new set *S**. This is the consensus set of *S*.
4) If the number of inliers |*S**| is greater than some threshold, fit the model again with all the consensus set.
5) If |S*| is smaller than a threshold, repeat 1).
6) After some number of iterations, return the largest consensus set found.

Iterative Closest Point (ICP) Besl'91

Given two PC with no a priori correspondences

1) Find the closest points

$$d(p_i, z_j) = \min_{p_{z_j} \in z_j} ||p_{z_j} - p_i||$$

Brute force search of a single point vs all other points. Other data structures can be used, such as KD-tree

2) Align the two PC after find the closest correspondences.

3) Do iteratively while convergence.