

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

□ Continuous-time optimization via B-spline interpolation:

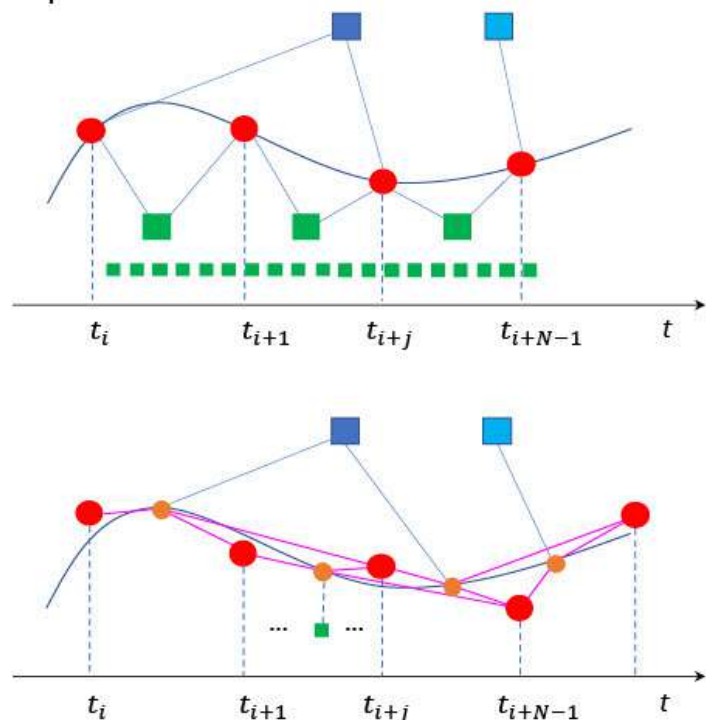
▪ **KEY TAKEOUT:**

A pose \mathbf{T}_t at time $t \in [t_i, t_{i+1})$ can be linearly interpolated from N control points

$\mathbf{T}_i, \mathbf{T}_{i+1}, \dots, \mathbf{T}_{i+N-1}$ via some time dependent weights.

▪ **ADVANTAGES:**

- Can integrate arbitrary number of sensors (multiple IMUs, multiple monocular VIOs...)
- Can estimate *time delay*.
- Can directly compensate for motion distortion in the MAP optimization.



B-SPLINE CONTINUOUS-TIME OPTIMIZATION

□ Continuous-time optimization via B-spline interpolation:

- B-spline is defined by:
 - Its order N (or degree $N - 1$), knot length Δt ,
 - $K \geq N$ knots (also called control points):

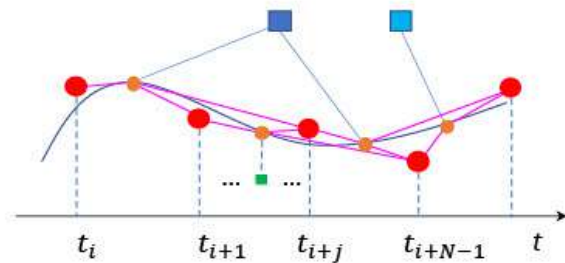
$$\{\mathbf{T}_i \triangleq (\mathbf{R}_i, \mathbf{p}_i)\}_{i=0}^{K-1}, \mathbf{R}_i \in \text{SO3}, \mathbf{p}_i \in \mathbb{R}^3$$

$$\mathbf{p}(s(t)) = \mathbf{p}_i + \sum_{j=1}^{N-1} \lambda_j(s(t)) \mathbf{p}_{i+j} = \mathbf{p}_i + \sum_{j=1}^{N-1} \tilde{\lambda}_j(s(t)) (\mathbf{p}_{i+j} - \mathbf{p}_{i+j-1})$$

$$\mathbf{R}(s(t)) = \mathbf{R}_i \prod_{j=1}^{N-1} \text{Exp}\left(\tilde{\lambda}_j(s(t)) \overbrace{\text{Log}(\mathbf{q}_{i+j-1}^{-1} \circ \mathbf{q}_{i+j})}^{d_j}\right) = \mathbf{R}_i \prod_{j=1}^{N-1} A_j$$

$$\underbrace{(1, \lambda_1, \lambda_2, \dots, \lambda_{N-1})}_{N \times 1} \triangleq \boldsymbol{\lambda}(s) = \underbrace{\mathbf{B}(N)}_{N \times N} \underbrace{(1, s, s^2, \dots, s^{N-1})}_{N \times 1}$$

$$\underbrace{(1, \tilde{\lambda}_1, \tilde{\lambda}_2, \dots, \tilde{\lambda}_{N-1})}_{N \times 1} \triangleq \boldsymbol{\lambda}(s) = \underbrace{\tilde{\mathbf{B}}(N)}_{N \times N} \underbrace{(1, s, s^2, \dots, s^{N-1})}_{N \times 1}$$



$$s \triangleq \frac{(t - t_i)}{\Delta t}, \forall t \in [t_i, t_{i+1}),$$

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

□ Continuous-time optimization via B-spline interpolation:

- B-spline is defined by:
 - Its order N (or degree $N - 1$), knot length Δt ,
 - $K \geq N$ knots (also called control points):

$$\{\mathbf{T}_i \triangleq (\mathbf{R}_i, \mathbf{p}_i)\}_{i=0}^{K-1}, \mathbf{R}_i \in \text{SO3}, \mathbf{p}_i \in \mathbb{R}^3$$

$$\mathbf{p}(s(t)) = \mathbf{p}_i + \sum_{j=1}^{N-1} \lambda_j(s(t)) \mathbf{p}_{i+j} = \mathbf{p}_i + \sum_{j=1}^{N-1} \tilde{\lambda}_j(s(t)) (\mathbf{p}_{i+j} - \mathbf{p}_{i+j-1})$$

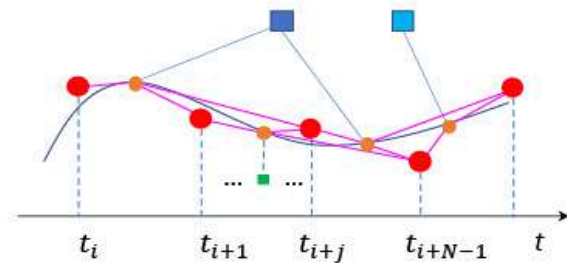
$$\mathbf{R}(s(t)) = \mathbf{R}_i \prod_{j=1}^{N-1} \text{Exp}\left(\overbrace{\tilde{\lambda}_j(s(t)) \text{Log}(\mathbf{q}_{i+j-1}^{-1} \circ \mathbf{q}_{i+j})}^{d_j}\right) = \mathbf{R}_i \prod_{j=1}^{N-1} A_j$$

$$\underbrace{(1, \lambda_1, \lambda_2, \dots, \lambda_{N-1})}_{N \times 1} \triangleq \boldsymbol{\lambda}(s) = \underbrace{\mathbf{B}(N)}_{N \times N} \underbrace{(1, s, s^2, \dots, s^{N-1})}_{N \times 1}$$

$$\underbrace{(1, \tilde{\lambda}_1, \tilde{\lambda}_2, \dots, \tilde{\lambda}_{N-1})}_{N \times 1} \triangleq \boldsymbol{\lambda}(s) = \underbrace{\tilde{\mathbf{B}}(N)}_{N \times N} \underbrace{(1, s, s^2, \dots, s^{N-1})}_{N \times 1}$$

KEY TAKEOUT:

A pose \mathbf{T}_t at time $t \in [t_i, t_{i+1})$ can be **linearly interpolated** from N control points $\mathbf{T}_i, \mathbf{T}_{i+1}, \dots, \mathbf{T}_{i+N-1}$ via some **time dependent weights**.



$$s \triangleq \frac{(t - t_i)}{\Delta t}, \forall t \in [t_i, t_{i+1}),$$

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

□ Gauss-Newton B-SPLINE Optimization on Manifold:

Need to calculate the gradient → Jacobian.

For 1st order observations, the Jacobian is:

$$J_{\mathbf{T}_j}^z = \frac{\partial h(\mathbf{T}_t, \check{\mathbf{Z}})}{\partial \mathbf{T}_j} = \begin{array}{|c|c|} \hline \frac{\partial h(\mathbf{T}_t, \check{\mathbf{Z}})}{\partial \mathbf{T}_t} & \frac{\partial h(\mathbf{T}_t, \check{\mathbf{Z}})}{\partial \mathbf{T}_j} \\ \hline \end{array}$$

Calculated as in Discrete Time case

▪ KEY TAKEOUT:

Calculate the Jacobian as normal, then multiple it with $\frac{\partial \mathbf{T}_t}{\partial \mathbf{T}_j}$

$$\frac{\partial \mathbf{p}_t}{\partial \mathbf{p}_j} = \lambda_j$$

$$\frac{\partial \mathbf{R}_t}{\partial \mathbf{R}_j} = \tilde{\lambda}_j \left(\prod_{m=j+1}^{m=N-1} A_m^\top \right) J_r(\tilde{\lambda}_j d_j) J_r^{-1}(d_j) - \tilde{\lambda}_{j+1} \left(\prod_{m=j+2}^{m=N-1} A_m^\top \right) J_r(\tilde{\lambda}_{j+1} d_{j+1}) J_r^{-1}(-d_{j+1})$$

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

- Gauss-Newton B-SPLINE Optimization on Manifold:

Acceleration factor:

$$J_{\mathbf{R}_j}^a = \frac{\partial}{\partial \mathbf{R}_j} [\mathbf{R}_t^{-1}(\ddot{\mathbf{p}}_t + \mathbf{g}) - \ddot{\mathbf{a}}] = [\mathbf{R}_t^{-1} \ddot{\mathbf{p}}_t]_{\times} \frac{\partial \mathbf{R}_t}{\partial \mathbf{R}_j}$$

$$J_{\mathbf{p}_j}^a = \frac{\partial}{\partial \mathbf{p}_j} [\mathbf{R}_t^{-1}(\ddot{\mathbf{p}}_t + \mathbf{g}) - \ddot{\mathbf{a}}] = \mathbf{R}_t^{-1} \frac{\partial \ddot{\mathbf{p}}_t}{\partial \mathbf{p}_j}$$

$$\frac{\partial \ddot{\mathbf{p}}_t}{\partial \mathbf{p}_j} = \ddot{\lambda}_j = \frac{\lambda_j}{\Delta t^2} \frac{ds^j}{ds^2}$$

Previous slide

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

□ Gauss-Newton B-SPLINE Optimization on Manifold:

Special factor based on IMU:

$$J_{\mathbf{R}_j}^{\omega} = \frac{\partial}{\partial \mathbf{R}_j} (\boldsymbol{\omega}_t - \tilde{\boldsymbol{\omega}}) = \frac{\partial \boldsymbol{\omega}_t}{\partial \mathbf{R}_j},$$

$$\boldsymbol{\omega}_t = \boldsymbol{\omega}^{(N)},$$

$$\boldsymbol{\omega}^{(j)} = \mathbf{A}_{j-1}^{\top} \boldsymbol{\omega}^{(j-1)} + \dot{\tilde{\lambda}}_{j-1} \mathbf{d}_{j-1}$$

$$\frac{\partial \boldsymbol{\omega}_t}{\partial \mathbf{R}_j} = \frac{\partial \boldsymbol{\omega}^{(j)}}{\partial \mathbf{d}_j} J_r^{-1}(\mathbf{d}_j) - \frac{\partial \boldsymbol{\omega}^{(j+1)}}{\partial \mathbf{d}_{j+1}} J_r^{-1}(-\mathbf{d}_{j+1})$$

$$\frac{\partial \boldsymbol{\omega}^{(j)}}{\partial \mathbf{d}_j} = \left(\prod_{m=j+1}^{m=N-1} \mathbf{A}_m^{\top} \right) \left(\tilde{\lambda}_j \mathbf{A}_j^{\top} [\boldsymbol{\omega}^{(j)}]_{\times} J_r(-\tilde{\lambda}_j \mathbf{d}_j) + \dot{\tilde{\lambda}}_j I \right)$$

B-SPLINE CONTINUOUS-TIME OPTIMIZATION

☐ Verified with Ceres Automatic derivatives

- Exactly the same as automatic derivatives.
- 80% faster
- More concise and clearly written code 😊 .

```
Analytic Jacobian: Size 12 42. Params: 14. Cost: 97.901182. Time: 1.604472.
-0.0212424 -0.008386231 -0.0102263 -0.221867 0.308899 -0.0741156
-0.0652494 -0.0218993 0.0053849 -0.322417 -0.169202 -0.153942
0.004888893 -0.008622768 -0.0198806 0.168155 0.0555763 -0.3822
0.0307997 -0.0155772 -0.134613 1.44186 0.026484 -7.44535
0.0265558 -0.00173455 -0.0575933 1.20734 0.103925 -3.29583
0.163524 0.0711059 0.054218 7.18386 -3.61627 1.55767
AutoDiff Jacobian: Size 12 42. Params: 14. Cost: 97.901182. Time: 6.257612.
-0.0212424 -0.008386231 -0.0102263 -0.221867 0.308899 -0.0741156
-0.0652494 -0.0218993 0.0053849 -0.322417 -0.169202 -0.153942
0.004888893 -0.008622768 -0.0198806 0.168155 0.0555763 -0.3822
0.0307997 -0.0155772 -0.134613 1.44186 0.026484 -7.44535
0.0265558 -0.00173455 -0.0575933 1.20734 0.103925 -3.29583
0.163524 0.0711059 0.054218 7.18386 -3.61627 1.55767
TMU Jacobian PASSED! Time reduced: 83.95 %
Pose Analytic Jacobian: Size 6 42. Params: 12. Cost: 1.310115. Time: 0.770401.
0.00942791 0.00213064 0.00466093 0.174813 -0.157822 0.0714925
0.000517306 0.0094067 -0.00377328 0.173525 0.146474 0.0542254
-0.0029327 0.000677144 0.00817702 -0.00809921 -0.0127591 0.241422
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
Pose AutoDiff Jacobian: Size 6 42. Params: 12. Cost: 1.310115. Time: 3.879575.
0.00942791 0.00213064 0.00466093 0.174813 -0.157822 0.0714925
0.000517306 0.0094067 -0.00377328 0.173525 0.146474 0.0542254
-0.0029327 0.000677144 0.00817702 -0.00809921 -0.0127591 0.241422
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
Pose Jacobian PASSED! Time reduced: 80.14 %
Lidar Analytic Jacobian: Size 1 42. Params: 12. Cost: 10.542036. Time: 0.538072.
-0.00113691 0.00528997 -0.0106898 0.4619 0.998523 -1.08712
Lidar AutoDiff Jacobian: Size 1 42. Params: 12. Cost: 10.542036. Time: 3.418527.
-0.00113691 0.00528997 -0.0106898 0.4619 0.998523 -1.08712
Lidar Jacobian PASSED! Time reduced: 84.26 %
```