Deformable Linear Object Tracking as Non-Rigid Point Set Registration

Jingyi Xiang[1]

February 7, 2023

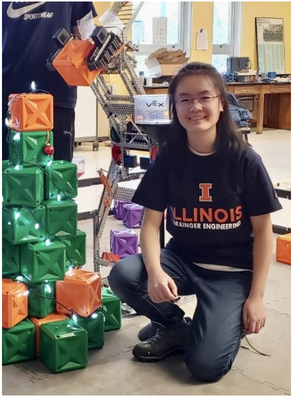
[1] Department of Electrical & Computer Engineering, University of Illinois at Urbana-Champaign

About Me

My name is Jingyi Xiang; I am a junior in Electrical Engineering and I joined the Bretl Research Group in January 2022. My current research is focused on deformable linear object perception and tracking.

Fun facts about me:

- For a third of my life I studied music and arts
- For another third of my life I wanted to become a theoretical physicist



OUTLINE

- Representing Deformable Linear Objects
  - Gaussian Mixture Model Clustering
  - Expectation-Maximization
- Non-Rigid point set registration
  - Measuring the Smoothness of a Functional
  - Optimization
- Challenges

Motivation

- As part of the *Representing and Manipulating Deformable Linear Objects (RMDLO)* project, one of our goals is to track the shape of deformable linear objects for manipulation.

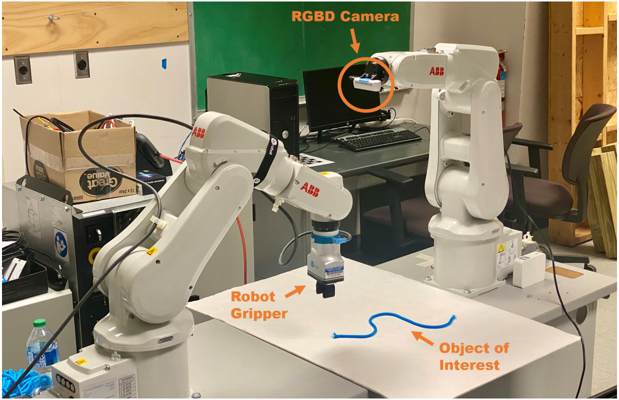


Figure 1: Lab setup.

- At each time step, the RGBD camera receives a point cloud of the DLO that consists of thousands of points.

Figure 2: DLO point cloud received by the RGBD camera, downsampled.

- We can use clustering to reduce the number of samples. By connecting the adjacent nodes, we can get a piecewise linear curve that approximates the current shape of the DLO.

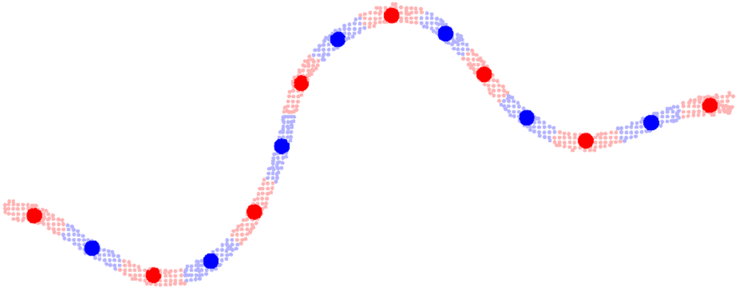


Figure 3: The DLO point cloud clustered into 15 nodes.

Gaussian Mixture Model Clustering

- Gaussian Mixture Model (GMM) clusters data into a **finite** number of Gaussian distributions[1].
- The parameters of the Gaussian distributions are unknown and need to be estimated from the data given.

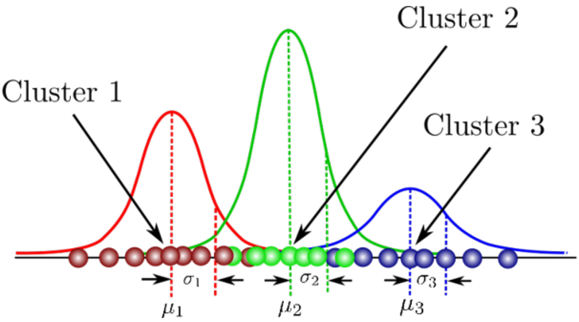


Figure 4: A simple example of GMM-based clustering.

[1] Bishop et al. 1995

- Assume a DLO can be represented by M nodes. The node positions at time step t are denoted by $\mathbf{Y}^t_{M \times D} = (\mathbf{y}^t_1, \ldots, \mathbf{y}^t_m)^T$, where $\mathbf{y}^t_m \in \mathbb{R}^3$ denotes the position of the mth node.
- The DLO point cloud received by the depth camera at time step t is denoted by $\mathbf{X}^t_{N \times D} = (\mathbf{x}^t_1, \ldots, \mathbf{x}^t_n)^T$, where $\mathbf{x}^t_n \in \mathbb{R}^3$ denotes the position of the nth point and there are N points in total.
- The collection of nodes $\mathbf{Y}^t$ serves as the centroids and the point cloud $\mathbf{X}^t$ are the randomly sampled points from the M Gaussian distributions.
- We further assume each Gaussian probability distribution has equal membership probability $\frac{1}{M}$ and variance σ^2.

- The probability distribution of $\mathbf{X}^t$ then becomes

$$p(\mathbf{x}_n^t) = \sum_{m=1}^{M} \frac{1}{M} \mathcal{N}(\mathbf{x}_n^t;\ \mathbf{y}_m^t,\ \sigma^2 \mathbf{I})$$
$$= \sum_{m=1}^{M} \frac{1}{M} \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2}\right)$$

- The goal of GMM clustering is to estimate the centroid positions $\mathbf{Y}^t$ and the variance σ^2 that maximizes the probability of observation $\mathbf{X}^t$:

$$(\mathbf{Y}^{t*}, \sigma^{2*}) = \underset{\mathbf{Y}^t, \sigma^2}{\operatorname{argmax}} \left(\prod_{n=1}^{N} p(\mathbf{x}_n^t) \right)$$

Gaussian Mixture Model Clustering

- Maximizing the probability of observation $\mathbf{X}^t$ is equivalent to minimizing its negative log likelihood

$$\mathcal{L}(\mathbf{Y}^t, \sigma^2 | \mathbf{X}^t) = -\log\left(\prod_{n=1}^{N} p(\mathbf{x}_n^t)\right) = -\sum_{n=1}^{N} \log\left(\sum_{m=1}^{M+1} p(m) p(\mathbf{x}_n^t | m)\right)$$

$$(\mathbf{Y}^{t*}, \sigma^{2*}) = \underset{\mathbf{Y}^t, \sigma^2}{\operatorname{argmin}} \mathcal{L}(\mathbf{Y}^t, \sigma^2 | \mathbf{X}^t)$$

- Since the summation inside $\log(\cdot)$ makes convex optimization impossible, we instead minimize its upper bound

$$E(\mathbf{Y}^t, \sigma^2) = \sum_{n=1}^{N} \sum_{m=1}^{M} p(m | \mathbf{x}_n^t) \log(p(m) p(\mathbf{x}_n^t | m))$$

which simplifies to

$$E(\mathbf{Y}^t, \sigma^2) = \sum_{n=1}^{N} \sum_{m=1}^{M} p(m | \mathbf{x}_n^t) \frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2} + \frac{\log(\sigma^2) D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} p(m | \mathbf{x}_n^t)$$

EXPECTATION-MAXIMIZATION

- We can solve this optimization problem iteratively using the Expectation-Maximization algorithm[2].
- The centroid positions $\mathbf{Y}^t$ are initialized to 0 and the variance σ^2 is initialized to $\frac{1}{DMN} \sum_{m=1}^{M} \sum_{n=1}^{N} \|\mathbf{y}_m^t - \mathbf{x}_n^t\|^2$.

[2] Dempster, Laird, and Rubin 1977

Expectation-Maximization

- **E-step:**
  The probability distribution $p(m|\mathbf{x}_n^t)$ is calculated from the $\mathbf{Y}^t$ and σ^2 found in the last iteration:

$$p(m|\mathbf{x}_n^t) = \frac{\exp\left(-\frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2}\right)}{\sum_{m=1}^{M}\exp\left(-\frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2}\right)}$$

- **M-step:**
  Plugging the new $p(m|\mathbf{x}_n^t)$ back into $E(\mathbf{Y}^t, \sigma^2)$, we can compute $\mathbf{Y}^t$ and σ^2 by letting $\frac{\partial E(\mathbf{Y}^t,\sigma^2)}{\partial \mathbf{Y}^t} = 0$ and $\frac{\partial E(\mathbf{Y}^t,\sigma^2)}{\partial \sigma^2} = 0$. We then have

$$\mathbf{y}_m^t = \frac{\sum_{n=1}^{N} p(m|\mathbf{x}_n^t)\mathbf{x}_n^t}{\sum_{n=1}^{N} p(m|\mathbf{x}_n^t)}$$

$$\sigma^2 = \frac{\sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t)\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{\sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t)D}.$$

- The E-step and the M-step are performed alternately until $\mathbf{Y}^t$ and σ^2 converge.

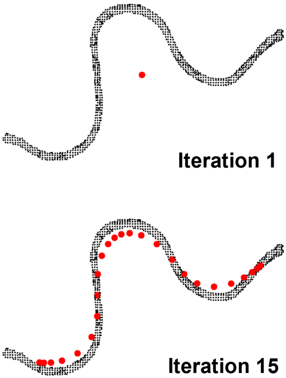


Figure 5: Clustering results from iterations 1, 10, 15, and 20, respectively.

- To better represent the shape of the DLO, we need to figure out the connectivity between nodes. We can encode the connectivity information into $\mathbf{Y}^t$ by ordering $\mathbf{Y}^t$ such that adjacent nodes are connected.

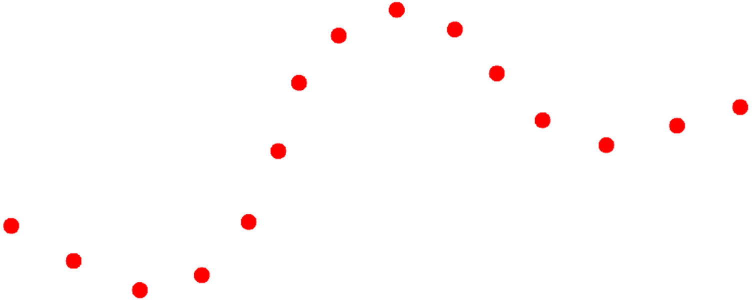


Figure 6: GMM clustering result.

- A naive method is to create a weighted complete graph from the nodes computed, then find the shortest path visiting all nodes exactly once.

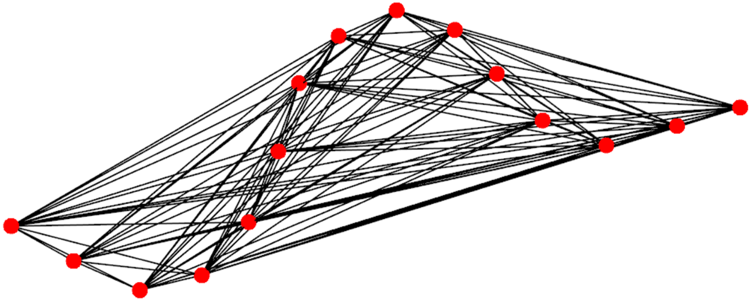


Figure 7: The complete graph created from a set of nodes.

- A naive method is to create a weighted complete graph from the nodes computed, then find the shortest path visiting all nodes exactly once.

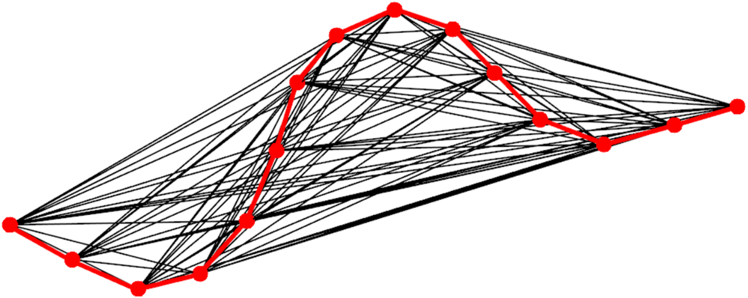


Figure 8: The shortest path visiting all nodes exactly once.

- Naive methods do not always work. Consider the scenario below:

Figure 9: Node ordering failure case.

- In some situations, it is not possible to extract the DLO shape from a single frame of data.

- If we have a set of correctly ordered nodes $\mathbf{Y}^{t-1}$ from time step $t-1$ and a set of unordered nodes $\mathbf{Y}^{t}$ from time step t, how can we find the correspondence between $\mathbf{Y}^{t}$ and $\mathbf{Y}^{t-1}$ so that $\mathbf{Y}^{t}$ is correctly ordered?

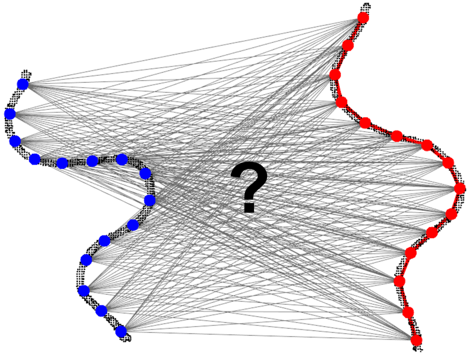


Figure 10: Red: Correct DLO shape estimate from time step $t-1$; Blue: GMM clustering results from time step t; Gray: All possible M^2 matchings.

- Non-rigid point set registration: finding correspondence between a **source point set** and a **target point set**. One of the most popular non-rigid point set registration algorithms is Coherent Point Drift[3].

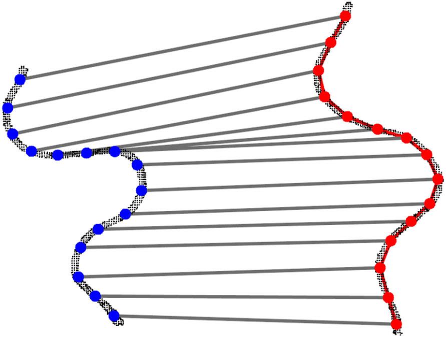


Figure 11: Red: Source point set $\mathbf{Y}^{t-1}$; Blue: Target point set $\mathbf{Y}^{t}$; Gray: Correspondences.

[3]Myronenko and Song 2010

Non-Rigid Point Set Registration

- CPD: the most probable matching between point sets is the one which produces the most spatially smooth velocity field.

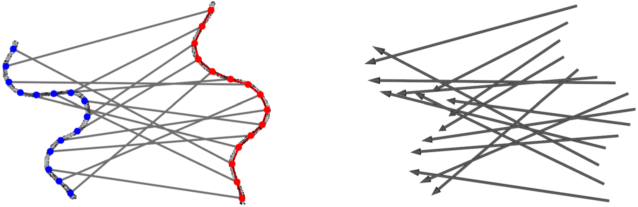


Figure 12: A non-smooth velocity field produces incorrect matchings.

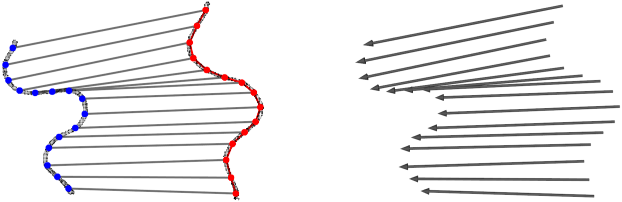


Figure 13: A smooth velocity field produces good matchings.

- To quantitatively measure the smoothness of the velocity field, define a velocity function $v(\mathbf{z})$ such that $\mathbf{Y}^t = \mathbf{Y}^{t-1} + v(\mathbf{Y}^{t-1})$. Note that v is a function of **spatial positions**, not time.
- One way of measuring the smoothness of a function is by measuring how oscillatory it is. This is equivalent to passing it through a high-pass filter in the frequency domain and integrating the resulting power.

Function 1: $f_1(t) = cos(\frac{\pi}{2}t)$ Function 2: $f_2(t) = cos(4\pi t)$

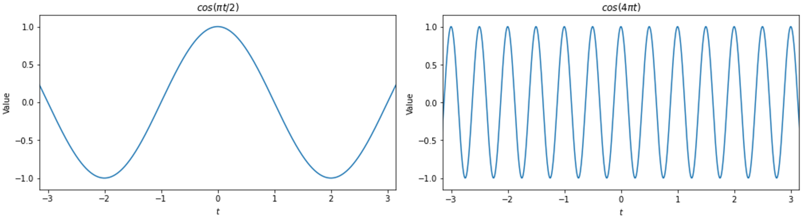


Figure 14: Plot of $f_1(t)$ and $f_2(t)$.

- We define $H(\omega)$ as an ideal high-pass filter with cutoff frequency at 1 rad/s to quantitatively measure the smoothness of f_1 and f_2:

$$H(\omega) = \begin{cases} 0 & \text{for } -1 < \omega < 1 \\ 1 & \text{otherwise} \end{cases}$$

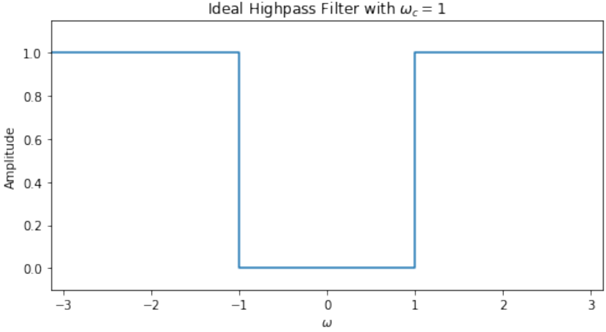


Figure 15: Ideal high-pass filter $H(\omega)$ with cutoff frequency at 1 rad/s.

- Function 1:

$$f_1(t) = cos(\frac{\pi}{2}t) \overset{\mathcal{F}}{\longleftrightarrow} F_1(\omega) = \frac{1}{2}\{\delta(\omega - \frac{1}{4}) + \delta(\omega + \frac{1}{4})\}$$

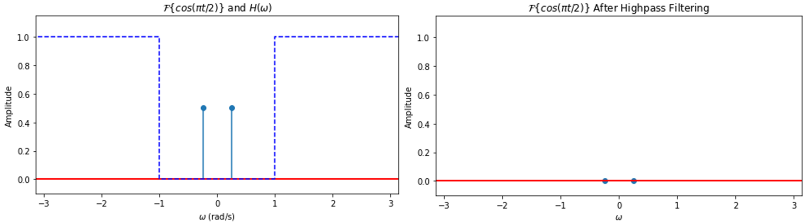


Figure 16: Left: Fourier Transform of function 1. Right: Applying $H(\omega)$ to function 1 filters out low-frequency content. Here, $\int_{-\infty}^{\infty} H(\omega)F_1(\omega)d\omega = 0$.

- Function 2: $f_2(t) = cos(4\pi t) \overset{\mathcal{F}}{\longleftrightarrow} F_2(\omega) = \frac{1}{2}\{\delta(\omega - 2) + \delta(\omega + 2)\}$

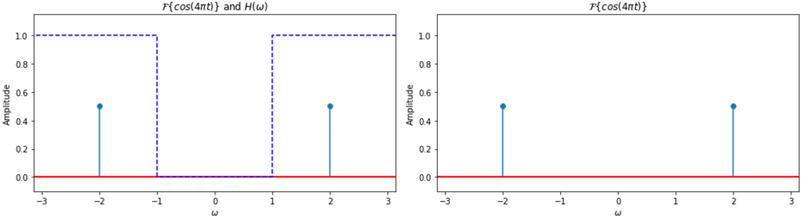


Figure 17: Left: Fourier Transform of function 2. Right: Applying $H(\omega)$ to function 2 does not filter out anything because function 2's frequency content lies in the pass band of $H(\omega)$. Here, $\int_{-\infty}^{\infty} H(\omega)F_2(\omega)d\omega = 1$.

- Since $\int_{-\infty}^{\infty} H(\omega)F_1(\omega)d\omega < \int_{-\infty}^{\infty} H(\omega)F_2(\omega)d\omega$, $f_1(t)$ has less high frequency content.
- The function $f_1(t)$ is smoother than the function $f_2(t)$.

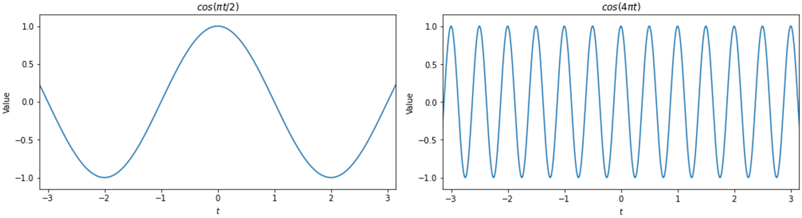


Figure 18: Comparison of $f_1(t)$ and $f_2(t)$ in the time domain.

- The cost function for GMM clustering is

$$E(\mathbf{Y}^t, \sigma^2) = \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t) \frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2} + \frac{\log(\sigma^2)D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t).$$

- Replace $\mathbf{Y}^t$ with $\mathbf{Y}^{t-1} + v(\mathbf{Y}^{t-1})$ and add the smoothness term to the cost function to obtain

$$E(v(\mathbf{z}), \sigma^2) = \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t) \frac{\|\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1}))\|^2}{2\sigma^2} + \frac{\log(\sigma^2)D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t) + \frac{\lambda}{2} \int_{\mathbb{R}^D} \frac{|\tilde{v}(\mathbf{s})|^2}{\tilde{G}(\mathbf{s})} d\mathbf{s},$$

where $\mathbf{z}$ is a spatial domain variable, $\mathbf{s}$ is a frequency domain variable, $\tilde{v}(\mathbf{s})$ is the Fourier Transform of $v(\mathbf{z})$, $1/\tilde{G}(\mathbf{s})$ is a high-pass filter, and $\frac{\lambda}{2}$ is a parameter weighting the smoothness term in optimization.

- Specifically, $1/\tilde{G}(\mathbf{s})$ takes the form $e^{\beta^2\|\mathbf{s}\|^2/2}$ so that $G(\mathbf{z}) = e^{-\|\mathbf{s}\|^2/(2\beta^2)}$ is Gaussian.
- The parameter β controls the frequency range included in the high-pass filter.
- Larger β values result in a high-pass filter with a narrower stop band which produces a smoother velocity field.

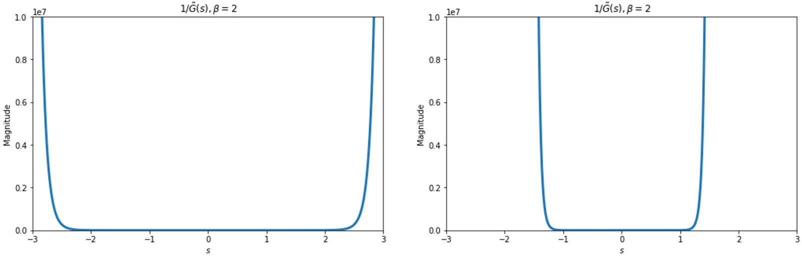


Figure 19: High-pass filter $1/\tilde{G}(\mathbf{s})$ with $\beta = 2$ and $\beta = 4$ respectively.

- Objective: Find $v(\mathbf{z})$ and σ^2 that minimize the cost function $E(v(\mathbf{z}), \sigma^2)$.
- Approach: Substitute $1/\tilde{G}(\mathbf{s})$ with $e^{\beta^2\|\mathbf{s}\|^2/2}$ and recognize $\frac{\lambda}{2}\int_{\mathbb{R}^D} |\tilde{v}(\mathbf{s})|^2/\tilde{G}(\mathbf{s})d\mathbf{s}$ is

$$\frac{\lambda}{2}\int_{\mathbb{R}^D}\sum_{l=0}^{\infty}\frac{\beta^{2l}}{2^l l!}\|\mathbf{D}^l v(\mathbf{z})\|^2 d\mathbf{z} = \frac{\lambda}{2}\|\mathbf{K}v(\mathbf{z})\|^2$$

  in the spatial domain. Here, $\mathbf{D}$ is a derivative operator with $\mathbf{D}^{2l}v = \triangledown^{2l}v$ and $\mathbf{D}^{2l+1}v = \triangledown(\triangledown^{2l}v)$, $\mathbf{K}$ is a pseudo-differential operator and $\|\cdot\|$ is the norm operator.
- The cost function then becomes

$$E(v(\mathbf{z}), \sigma^2) = \sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t)\frac{\|\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1}))\|^2}{2\sigma^2} + \frac{\log(\sigma^2)D}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t) + \frac{\lambda}{2}\|\mathbf{K}v(\mathbf{z})\|^2.$$

- We can solve for $v(\mathbf{z})$ using regularization theory. $E(v(\mathbf{z}), \sigma^2)$ can be divided into two parts, the empirical cost functional E_{emp} and the regularizer cost functional E_{reg}:

$$\begin{aligned} E_{emp} &= \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t) \frac{\|\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1}))\|^2}{2\sigma^2} \\ &\quad + \frac{\log(\sigma^2) D}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} p(m|\mathbf{x}_n^t) \\ E_{reg} &= \frac{\lambda}{2} \|\mathbf{K} v(\mathbf{z})\|^2 \end{aligned}$$

- E_{emp} describes the goodness of fit of $\mathbf{Y}^t$ to the original data, $\mathbf{X}^t$
- E_{reg} describes smoothness of the velocity field, $v(\mathbf{z})$

- To minimize $E(v(\mathbf{z}), \sigma^2) = E_{emp} + E_{reg}$, we need to find $v(\mathbf{z})$ such that the Fréchet differential of $E(v(\mathbf{z}), \sigma^2)$ is zero.
- Definition of the Fréchet differential:

$$df(x, h) = \lim_{\epsilon \to 0} \frac{f(x + \epsilon h) - f(x)}{\epsilon}$$

Optimization

- The Fréchet differential for E_{emp} is

$$dE_{emp} = \frac{d}{d\epsilon} E_{emp}(v(\mathbf{z}) + \epsilon h(\mathbf{z}))\bigg|_{\epsilon=0}$$

$$= -\frac{1}{\sigma^2} \sum_{m=1}^{M} h(\mathbf{z}) \sum_{n=1}^{N} (\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)$$

$$= \left\langle h(\mathbf{z}),\ -\sum_{m=1}^{M} \sum_{n=1}^{N} \frac{1}{\sigma^2} (\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)\delta(\mathbf{z} - \mathbf{y}_m) \right\rangle.$$

- The Fréchet differential for E_{reg} is

$$dE_{reg} = \frac{d}{d\epsilon} \left(\frac{\lambda}{2} \int_{-\infty}^{\infty} \mathbf{K}(v(\mathbf{z}) + \epsilon h(\mathbf{z}))\mathbf{K}(v(\mathbf{z}) + \epsilon h(\mathbf{z}))d\mathbf{z} \right) \bigg|_{\epsilon=0}$$

$$= \lambda \int_{-\infty}^{\infty} \mathbf{K}h(\mathbf{z})\mathbf{K}v(\mathbf{z})d\mathbf{z}$$

$$= \left\langle \mathbf{K}h(\mathbf{z}),\ \lambda\mathbf{K}v(\mathbf{z}) \right\rangle.$$

- Following $\langle \mathbf{K}h(\mathbf{z}), v(\mathbf{z})\rangle = \langle h(\mathbf{z}), \tilde{\mathbf{K}}v(\mathbf{z})\rangle$, we can rewrite dE_{reg} as

$$dE_{reg} = \left\langle \mathbf{K}h(\mathbf{z}),\ \lambda\mathbf{K}v(\mathbf{z}) \right\rangle = \left\langle h(\mathbf{z}),\ \lambda\tilde{\mathbf{K}}\mathbf{K}v(\mathbf{z}) \right\rangle$$

  where $\tilde{\mathbf{K}}$ is the adjoint operator of pseudo-differential operator $\mathbf{K}$.

- $dE_{emp} + dE_{reg} = 0$ then yields

$$\left\langle h(\mathbf{z}),\ \lambda\tilde{\mathbf{K}}\mathbf{K}v(\mathbf{z}) - \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{1}{\sigma^2}(\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)\delta(\mathbf{z} - \mathbf{y}_m) \right\rangle = 0$$

  The functional $h(\mathbf{z})$ is a constant fixed of $\mathbf{z}$, so for this inner product to hold,

$$\tilde{\mathbf{K}}\mathbf{K}v(\mathbf{z}) - \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{1}{\sigma^2\lambda}(\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)\delta(\mathbf{z} - \mathbf{y}_m) = 0$$

- This is the Euler-Lagrange equation of $E(v(\mathbf{z}), \sigma^2)$.

- **The Euler-Lagrange equation of** $E(v(\mathbf{z}), \sigma^2)$ is

$$\tilde{\mathbf{K}}\mathbf{K}v(\mathbf{z}) = \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{1}{\sigma^2\lambda}(\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)\delta(\mathbf{z} - \mathbf{y}_m).$$

- Denote operator $\mathbf{L} = \tilde{\mathbf{K}}\mathbf{K}$. For pseudo-differential operator $\|\mathbf{K}v(\mathbf{z})\|^2 = \int_{\mathbb{R}^D} \sum_{l=0}^{\infty} \frac{\beta^{2l}}{2^l l!}\|\mathbf{D}^l v(\mathbf{z})\|^2 d\mathbf{z}$, $\mathbf{L} = \tilde{\mathbf{K}}\mathbf{K} = \sum_{l=0}^{\infty} \frac{(-1)^l \beta^{2l}}{l! 2^l} \nabla^{2l}$ [4].
- Differential function with the form $\mathbf{L}f(\mathbf{z}) = \phi(\mathbf{z})$ has solution $f(\mathbf{z}) = \int_{\mathbb{R}^D} G(\mathbf{z} - \zeta)\phi(\zeta)d\zeta$, where G satisfies $\mathbf{L}G(\mathbf{z}) = \delta(\mathbf{z})$. Therefore,

$$v(\mathbf{z}) = \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{1}{\sigma^2\lambda}(\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1})))p(m|\mathbf{x}_n^t)G(\mathbf{z} - \mathbf{y}_m).$$

[4] Chen and Haykin 2002

- Since $\mathbf{L} = \sum_{l=0}^{\infty} \frac{(-1)^l \beta^{2l}}{l! 2^l} \nabla^{2l}$ and $\mathbf{L}G(\mathbf{z}) = \delta(\mathbf{z})$, we can solve for $\tilde{G}(\mathbf{s})$ and $G(\mathbf{z})$ through Fourier Transform:

$$\sum_{l=0}^{\infty} \frac{(-1)^l \beta^{2l}}{l! 2^l} \nabla^{2l} G(\mathbf{z}) = \delta(\mathbf{z})$$

$$\tilde{G}(\mathbf{s}) = \frac{1}{\sum_{l=0}^{\infty} \frac{\beta^{2l}}{l! 2^l} \|\mathbf{s}\|^2} = e^{-\beta^2 \|\mathbf{s}\|^2 / 2}; \quad G(\mathbf{z}) = e^{-\|\mathbf{z}\|^2 / (2\beta^2)}$$

- Alternatively, we can write $v(\mathbf{z})$ as

$$v(\mathbf{z}) = \sum_{m=1}^{M} \mathbf{w}_m G(\mathbf{z} - \mathbf{y_m})$$

$$\mathbf{w}_m = \sum_{n=1}^{N} \frac{1}{\sigma^2 \lambda} (\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1}))) p(m | \mathbf{x}_n^t)$$

- Going back to the cost function

$$E(v(\mathbf{z}), \sigma^2) = \sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t)\frac{\|\mathbf{x}_n^t - (\mathbf{y}_m^{t-1} + v(\mathbf{y}_m^{t-1}))\|^2}{2\sigma^2} + \frac{\log(\sigma^2)D}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} p(m|\mathbf{x}_n^t) + \frac{\lambda}{2}\int_{\mathbb{R}^D}\frac{|\tilde{v}(\mathbf{s})|^2}{\tilde{G}(\mathbf{s})}d\mathbf{s}$$

- Define the following notations
  - $\mathbf{W}_{M\times D}$ is the collection of weights, $(\mathbf{w}_1, \ldots, \mathbf{w}_M)^T$
  - $\mathbf{G}_{M\times M}$ is the kernel matrix with $\mathbf{G}(i,j) = G(\mathbf{y}_i - \mathbf{y}_j)$
- We can now write $v(\mathbf{y}_m^{t-1})$ as $\mathbf{G}(m,\cdot)\mathbf{W}$. Since $\mathbf{G}$ is known, we only need to solve for the weights $\mathbf{W}$.

- For better readability, denote $\mathbf{X}^t$ as $\mathbf{X}$ and $\mathbf{Y}^{t-1}$ as $\mathbf{Y}_0$. Rewriting $E(v(\mathbf{z}), \sigma^2)$ in matrix form, we get

$$\begin{aligned} E(\mathbf{W}, \sigma^2) = & \frac{1}{2\sigma^2}\{\mathrm{tr}(\mathbf{X}^T d(\mathbf{P}^T \mathbf{1})\mathbf{X}) - 2\mathrm{tr}(\mathbf{Y}_0^T \mathbf{P}\mathbf{X}) - 2\mathrm{tr}(\mathbf{W}^T \mathbf{G}\mathbf{P}\mathbf{X}) \\ & + \mathrm{tr}(\mathbf{Y}_0^T d(\mathbf{P1})\mathbf{Y}_0) + 2\mathrm{tr}(\mathbf{W}^T \mathbf{G} d(\mathbf{P1})\mathbf{Y}_0) + \mathrm{tr}(\mathbf{W}^T \mathbf{G} d(\mathbf{P1})\mathbf{G}\mathbf{W})\} \\ & + \frac{D}{2}\mathbf{1}^T \mathbf{P1} \log(\sigma^2) + \mathrm{tr}(\mathbf{W}^T \mathbf{G}\mathbf{W}), \end{aligned}$$

  where
  - $\mathbf{P}_{M \times N}$ is the posteriori probability matrix with entries $\mathbf{P}(m, n) = p(m|\mathbf{x}_n^t)$
  - $d(\mathbf{a})$ is the diagonal matrix constructed from vector $\mathbf{a}$
  - $\mathrm{tr}(\mathbf{m})$ is the trace of matrix $\mathbf{m}$
  - $\mathbf{1}$ is a column vector of ones

- **E-step:**
  The posteriori probability matrix $\mathbf{P}$ is calculated from the $\mathbf{Y}^t$ and σ^2 found in the last iteration:

$$\mathbf{P}(m,n) = \frac{\exp\left(-\frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2}\right)}{\sum_{m=1}^{M}\exp\left(-\frac{\|\mathbf{x}_n^t - \mathbf{y}_m^t\|^2}{2\sigma^2}\right)}$$

- **M-step:**
  Plugging the new $\mathbf{P}$ back into $E(\mathbf{W},\sigma^2)$, we can compute $\mathbf{W}$ and σ^2 by letting $\frac{\partial E(\mathbf{W},\sigma^2)}{\partial \mathbf{W}} = 0$ and $\frac{\partial E(\mathbf{W},\sigma^2)}{\partial \sigma^2} = 0$. We then have

$$\mathbf{W} = \left(d(\mathbf{P1})\mathbf{G} + \lambda\sigma^2\mathbf{I}\right)^{-1} \cdot (\mathbf{PX} - d(\mathbf{P1})\mathbf{Y}_0)$$

$$\sigma^2 = \frac{1}{\mathbf{1}^T\mathbf{P1}D}(tr(\mathbf{X}^T d(\mathbf{P}^T\mathbf{1})\mathbf{X}) - 2tr((\mathbf{PX})^T(\mathbf{Y}_0 + \mathbf{GW})) + tr((\mathbf{Y}_0 + \mathbf{GW})^T d(\mathbf{P1})(\mathbf{Y}_0 + \mathbf{GW})))$$

- The final solution is $\mathbf{Y}^t = \mathbf{Y}^{t-1} + \mathbf{GW}$.

Similar to GMM clustering, we repeat the Expectation-Maximization process until $\mathbf{W}$ and σ^2 converge.

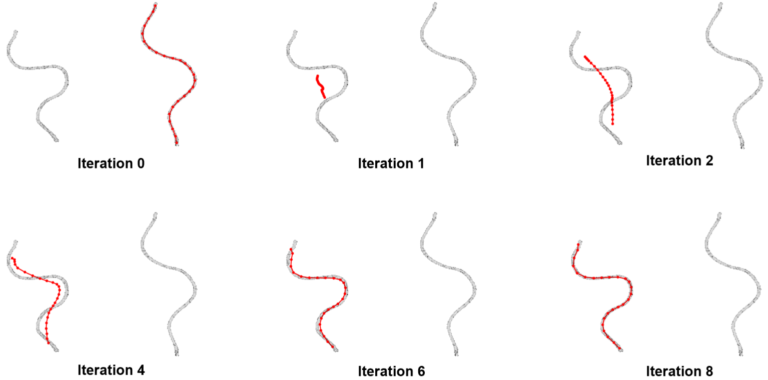


Figure 20: Non-rigid registration result for iteration 0, 1, 2, 4, 6, and 8, respectively.

CHALLENGES

- Every cost term added to $E(\mathbf{W}, \sigma^2)$ must be optimized through EM.
- Consider a length preservation constraint restricting the total length of the predicted DLO to length L. This leads to the cost term

$$\left\| \sum_{m=1}^{M-1} \left\| (\mathbf{y}_{m+1}^{t-1} + v(\mathbf{y}_{m+1}^{t-1})) - (\mathbf{y}_m^t + v(\mathbf{y}_m^{t-1})) \right\|^2 - L \right\|^2$$

  which cannot be written into the form of $\langle h(\mathbf{z}), f(\mathbf{z}) \rangle$ for computing the Fréchet differential.
- Physical properties of the DLO are often only considered in post-processing.

One of the major drawbacks of treating DLO tracking as a non-rigid point set registration algorithm: the physical properties of the object are not explicitly represented. Existing DLO tracking methods use different techniques to overcome this issue:

- CPD+Physics (2017) and Structure Preserved Registration (2019) use physics simulators for post-processing.
- Structure Preserved Registration (2019) and Constrained Deformable Coherent Point Drift (2019) adds locally linear embedding as an additional cost term in the EM process to preserve local topology.
- Constrained Deformable Coherent Point Drift (2019) and Constrained Deformable Coherent Point Drift 2 (2021) use constrained optimization for DLO length preservation in post-processing.
- Constrained Deformable Coherent Point Drift 2 (2021) uses gripper motion information to predict the shape of DLO.

Takeaways

- For any additional cost terms added to the non-rigid point set registration process, $v(\mathbf{z}) = \sum_{m=1}^{M} \mathbf{w}_m G(\mathbf{z} - \mathbf{y}_m)$ must still minimize the total cost.
- Existing algorithms add convex constraints and post-processing steps to improve tracking performance without changing the cost functional in EM.

ACKNOWLEDGEMENTS

- I would like to thank my collaborators Holly Dinkel, Harry Zhao, Naixiang Gao, Brian Coltin, Trey Smith, and Tim Bretl for their technical support and valuable feedback.

Scan the below QR code to check out our software!

URL: `https://github.com/RMDLO/trackdlo`