Tracking Deformable Linear Objects Under Occlusion

Jingyi Xiang[1]
with Holly Dinkel[2], Harry Zhao[2], Tim Bretl[2], Brian Coltin[3], Trey Smith[3]

September 15, 2022

[1] Department of Electrical & Computer Engineering, University of Illinois at Urbana-Champaign
[2] Department of Aerospace Engineering, University of Illinois at Urbana-Champaign
[3] Intelligent Robotics Group, NASA Ames Research Center

About Me

- I am a junior in Electrical Engineering
- I joined this group in January 2022

OUTLINE

- Motivation
- Tracking algorithm overview
- Challenges for the current tracking algorithm
- A potential solution
  - Euclidean vs. geodesic distance
  - Revisiting the Motion Coherence Theory

MOTIVATION

Robotic manipulation of deformable linear objects (DLOs) has many practical applications:

- Autonomous knot tying for surgical robots
- Autonomous cable routing for industrial robots

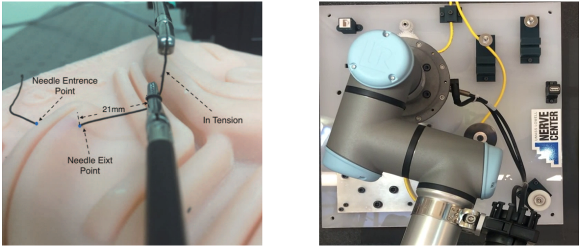


Figure 1: Left: Surgical robot performs knot-tying[1]. Right: Industrial robot performs cable routing[2].

[1] Lu, Chu, and Cheng 2016

[2] Keipour, Bandari, and Schaal 2022

Motivation

- Wire perception in 3D for manipulation tasks
- Directly obtain deformable object shape estimate from sensor data

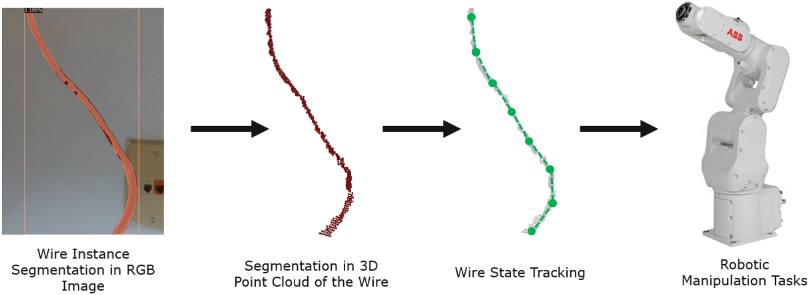

1 100%
ABB
Wire Instance Segmentation in RGB Image
Segmentation in 3D Point Cloud of the Wire
Wire State Tracking
Robotic Manipulation Tasks

Figure 2: System pipeline.

Tracking Algorithm Overview

- Assume a DLO can be represented by M nodes. The node positions are denoted by $Y = \{y_1, \dots, y_m\} \in \mathbb{R}^{M \times 3}$, where $y_m \in \mathbb{R}^3$ denotes the position of the mth node.
- The DLO's point cloud preceived by the depth camera is denoted by $X = \{x_1, \dots, x_n\} \in \mathbb{R}^{N \times 3}$, where $x_n \in \mathbb{R}^3$ denotes the position of the nth point and there are N points in total.
- The nodes Y serves as the centroids and the point cloud X are the randomly sampled points from the M Gaussian distributions.

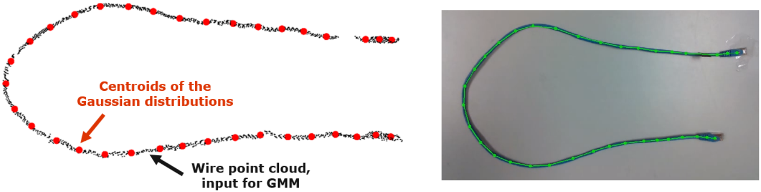

Centroids of the Gaussian distributions
Wire point cloud, input for GMM

Figure 3: Left: DLO point cloud and nodes visualized in 3D. Right: DLO point cloud and nodes visualized in a 2D image.

Tracking Algorithm Overview

- Gaussian Mixture Model (GMM) clusters data into a **finite** number of Gaussian distributions[3].
- The parameters of the Gaussian distributions are unknown and need to be estimated from the data given.

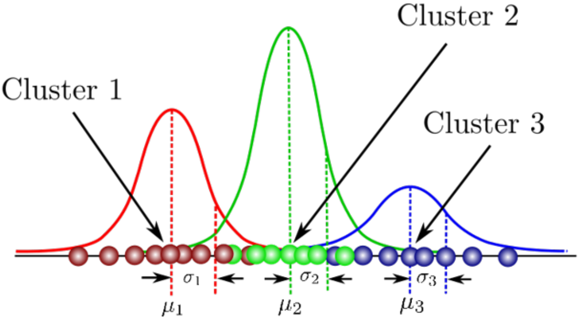
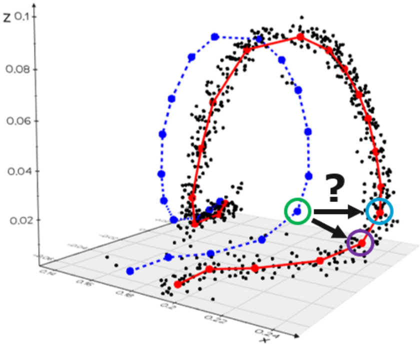


Figure 4: A simple example of GMM-based clustering.

[3] Bishop et al. 1995

Tracking Algorithm Overview



Blue dots
Registered nodes for the point cloud at time step t-1

Black dots
Point cloud at time step t, slightly displaced from the point cloud at time step t-1

Red dots
Registered nodes for the point cloud at time step t

Figure 5: Finding node correspondence between frames.

Tracking Algorithm Overview

Assign correspondence for nodes in consecutive frames using the Motion Coherence Theory[4].

- Motion Coherence Theory: nodes close to each other tends to move in similar directions.

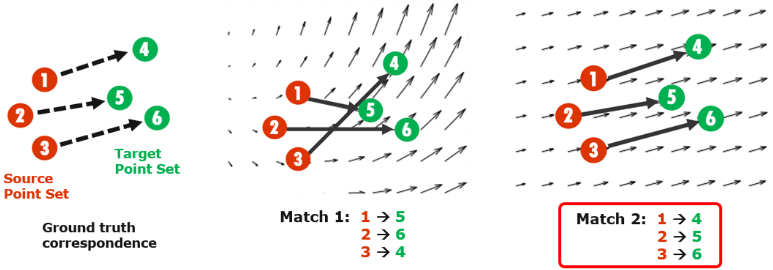


Figure 6: The Motion Coherence Theory.

[4] Yuille and Grzywacz 1989

Tracking Algorithm Overview

DLO tracking:

1. Extract the nodes using Gaussian Mixture Model clustering.
2. Find the node correspondence between frames using the Motion Coherence Theory.

Current Challenges: Tip Occlusion

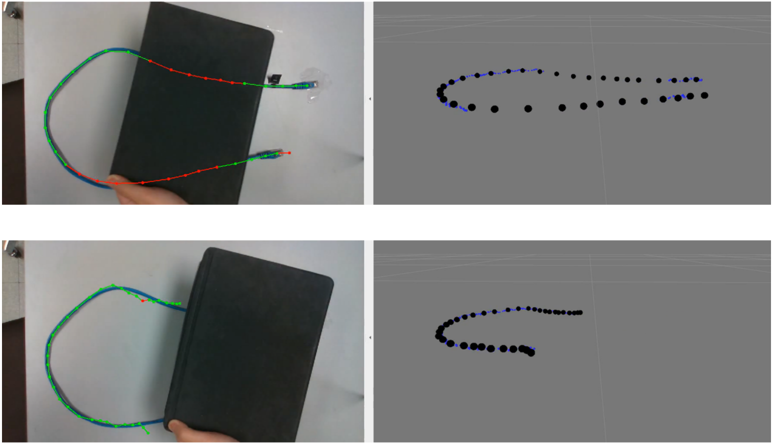


Figure 7: Tracking failure case.

Current Challenges: Tip Occlusion

- Before processing the current frame that has occlusion, divide the nodes into visible nodes and occluded nodes.

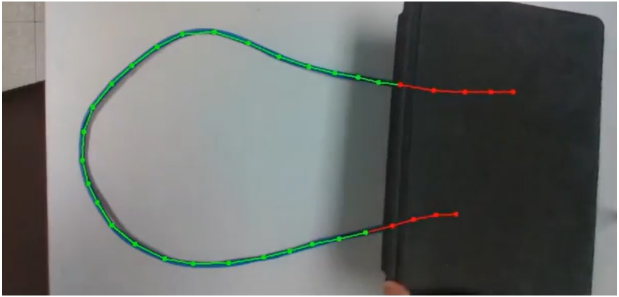


Figure 8: Dividing the object into two parts: Green: visible nodes, Red: occluded Nodes.

- Since the visible nodes are staying stationary, the occluded nodes are **probably** also staying stationary.

Problem Statement

Given how the visible nodes move between frames, how should the occluded nodes move?

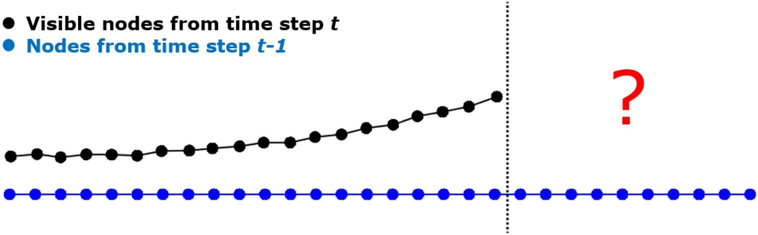

● Visible nodes from time step *t*
● Nodes from time step *t-1*
?

Figure 9: Illustration of the problem.

Determine Neighboring Nodes

Motion Coherence Theory: nodes **close to each other** tends to move in similar directions.

- The closer the two nodes are, the more ones motion affects another.

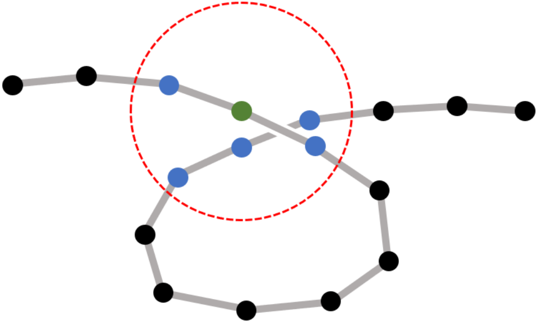


Figure 10: The node of interest and the neighboring nodes right next to it.

Determine Neighboring Nodes

If the green node is moving upward, the neighboring nodes should also move upward according to the Motion Coherence Theory.

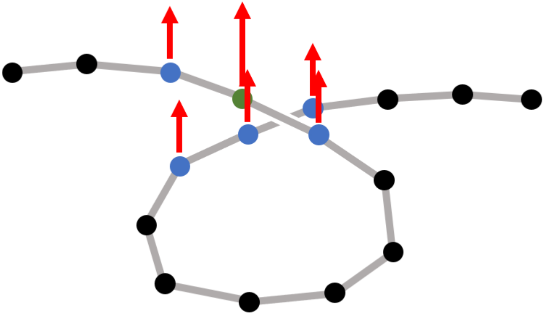


Figure 11: Nodes that should move together based on Euclidean distance.

Determine Neighboring Nodes

Only the neighboring nodes that are also from the top part of the DLO should move together.

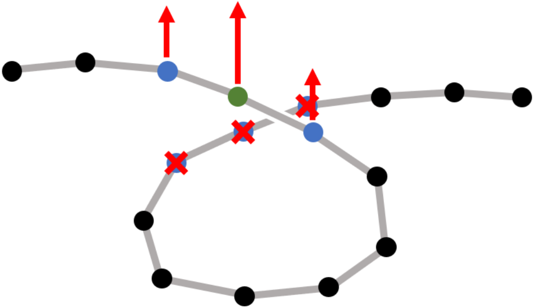


Figure 12: Incorrect node grouping based on Euclidean distance.

Determine Neighboring Nodes

It is more appropriate to group nodes together based on their **geodesic distances** to each other.

- Geodesic distance: the distance between two points on the surface of the object[5].

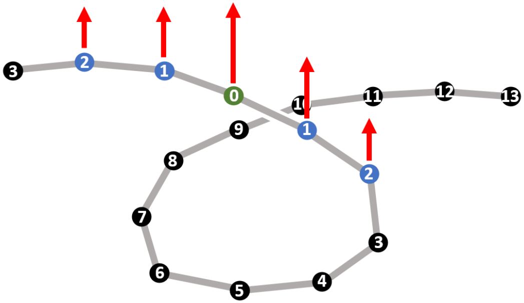


Figure 13: Nodes that should move together based on geodesic distance.

[5] Ruan, McConachie, and Berenson 2018

Motion Coherence Theory: The Problem

If a node A from time step $t-1$, corresponds to a node B from time step t, then a velocity vector can be assigned at the position of node A based on the difference in spatial positions of the nodes and the time between two time steps. Assume the time difference between frames is **normalized**.

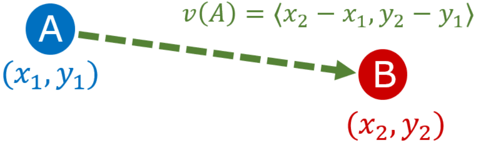


Figure 14: Blue: node A from time step $t-1$. Red: node B from time step t. Green: velocity vector assigned at (x_1, y_1).

Motion Coherence Theory: The Problem

- M nodes would lead to $M!$ possible matches between two consecutive frames and $M!$ possible velocity assignments at the nodes from time step $t-1$.
- Some of these matches will produce smoother velocity fields than the others. The Motion Coherence Theory proposes that the most possible match is the one produces the most smooth velocity field[6].

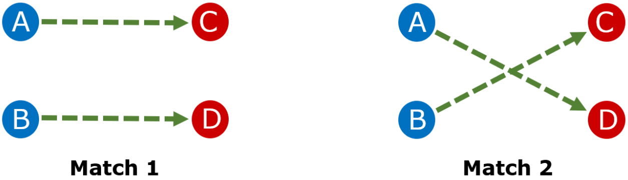


Figure 15: The two possible matches for $M = 2$. Blue: nodes from time step $t-1$. Red: nodes from time step t.

[6] Yuille and Grzywacz 1989

The Motion Coherence Theory

Formally, the Motion Coherence Theory states the following:

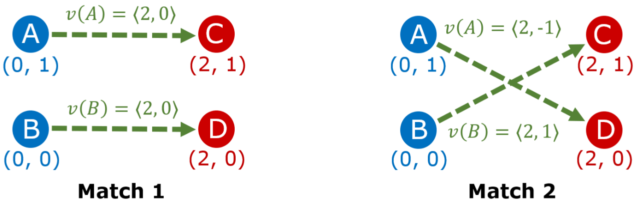
Let the measured velocity of node y_m be $M(\vec{u_m})$, where $\vec{u_m}$ is the true velocity of the node.

The smoothing stage constructs a velocity field, $v(Y)$, such that the following functional is minimized:

$$E(v(Y), \vec{U}) = \sum_{m=1}^{M} \|v(y_m) - M(\vec{u_m})\|^2 + \lambda \int_{\mathbb{R}^D} \sum_{k=0}^{\infty} c_k \|D^k v(Y)\|^2$$

Where $\lambda \geq 0$ and $c_k \geq 0$ are constants determining the strength of the smoothing and D^k is the derivative operator.

Motion Coherence Theory: Simple Example



$v(A) = \langle 2, 0 \rangle$
A
(0, 1)
C
(2, 1)
$v(B) = \langle 2, 0 \rangle$
B
(0, 0)
D
(2, 0)
Match 1
A
(0, 1)
$v(A) = \langle 2, \text{-}1 \rangle$
C
(2, 1)
B
(0, 0)
$v(B) = \langle 2, 1 \rangle$
D
(2, 0)
Match 2

Figure 16: The two possible matches for $M = 2$.

- Let $\lambda = c_0 = c_1 = 1$.
- Match 1:

  $k = 0$: $\|v(A)\|^2 + \|v(B)\|^2 = (2^2 + 0^2) + (2^2 + 0^2) = 8$

  $k = 1$: $\|v(A) - v(B)\|^2 = |2 - 2|^2 + |0 - 0|^2 = 0$

  Total cost $E = \lambda(c_0 \cdot 4 + c_1 \cdot 0)$ **= 8**
- Match 2:

  $k = 0$: $\|v(A)\|^2 + \|v(B)\|^2 = (2^2 + (-1)^2) + (2^2 + 1^2) = 10$

  $k = 1$: $\|v(A) - v(B)\|^2 = |2 - 2|^2 + |-1 - 1|^2 = 4$

  Total cost $E = \lambda(c_0 \cdot 6 + c_1 \cdot 4)$ **= 14**

Motion Coherence Theory: Smoothness Operator

The term $\lambda \int_{\mathbb{R}^D} \sum_{k=0}^{\infty} c_k \|D^k v(Y)\|^2$ measures the smoothness of the velocity field. Two things that determine how the c_k values interacts with the velocity field:

- The **form** of c_k (e.g., $c_k = 2^k$ vs. $c_k = 2k$)
- The **value** of c_k (e.g., $c_k = 2^k$ vs. $c_k = 200^k$)

The set of values $\{c_0, \ldots, c_k\}$ is called the **smoothness operator**.

Motion Coherence Theory: Smoothness Operator

- To investigate how the **value** of c_k affects the velocity field, consider smoothness operator of the form

$$c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0$$

- Three β values will be used to align the blue nodes to the black nodess: $\beta = 0.05$, $\beta = 1$, $\beta = 2$

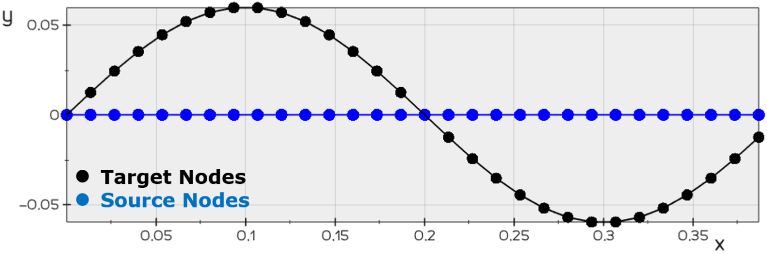


Figure 17: Caption

Motion Coherence Theory: Smoothness Operator

- $\beta = 0.05$ for $c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0$
- Source and target nodes **completely aligned**.

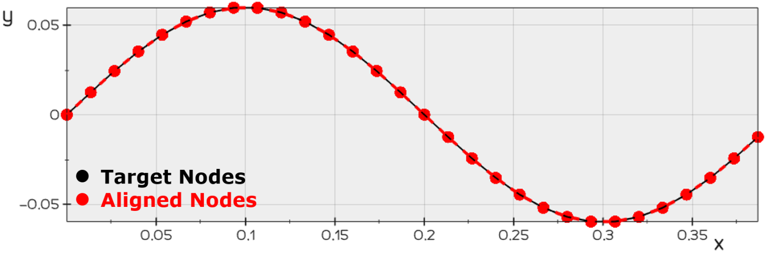


Figure 18: Alignment result for β = 0.05.

Motion Coherence Theory: Smoothness Operator

- $\beta = 1$ for $c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0$
- Source and target nodes **roughly aligned**.

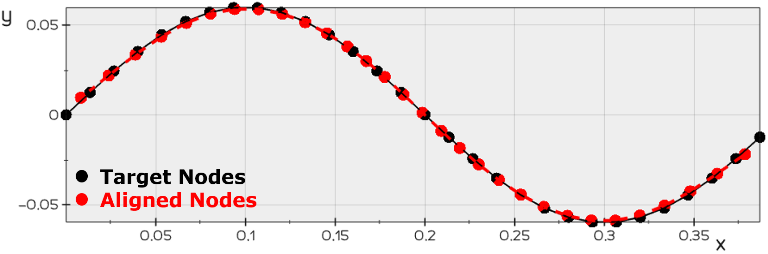


Figure 19: Alignment result for β = 1.

Motion Coherence Theory: Smoothness Operator

- $\beta = 2$ for $c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0$
- Source and target nodes **not aligned**.

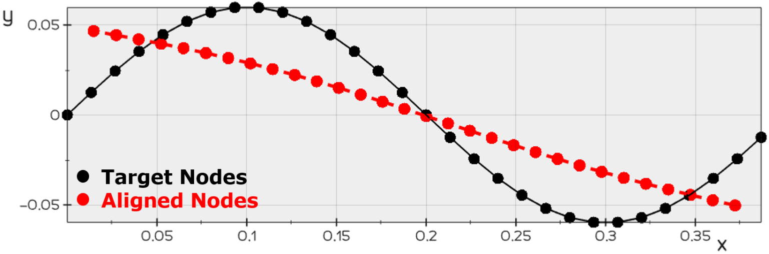


Figure 20: Alignment result for β = 2.

Motion Coherence Theory: Smoothness Operator

In general, the **larger** the values of $\{c_0, \ldots, c_k\}$, the **less** the nodes will move relative to each other.

- Reminder:

$$E(v(Y), \vec{U}) = \sum_{m=1}^{M} \|v(y_m) - M(\vec{u_m})\|^2 + \lambda \int_{\mathbb{R}^D} \sum_{k=0}^{\infty} c_k \|D^k v(Y)\|^2$$

Motion Coherence Theory: Smoothness Operator

- To investigate how smoothness operators with different **form** interacts with the velocity field, three smoothness operators will be used to impute the velocity of nodes $y_{21}, \dots, y_{30}$ given the velocity of nodes $y_1, \dots, y_{20}$.
- All three smoothness operator will be written in terms of β. The largest β value they can have (that still keeps the object deformable) will be used.

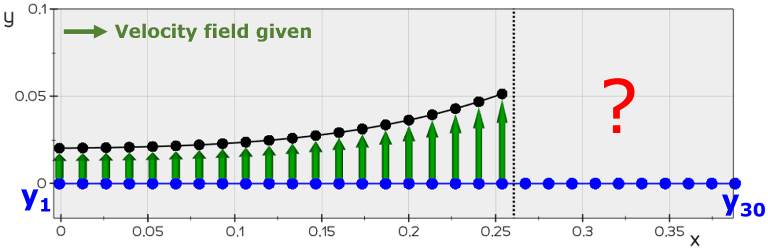


Figure 21: Experiment setup.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 1:

$$c_0 = 1,\ c_1 = \beta^2,\ c_2, \ldots, c_k = 0;\ \beta = 16$$

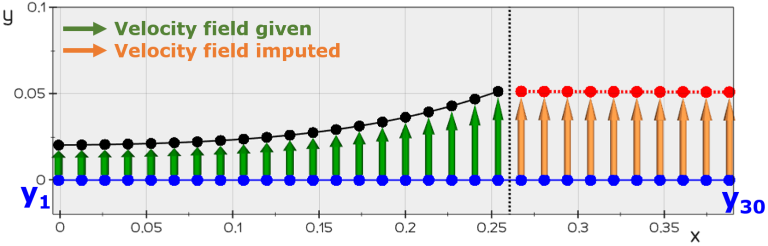


Figure 22: Velocity imputation result for smoothness operator 1.

The Motion Coherence Theory: Smoothness Operator

- Minimizing $\lambda \int_{\mathbb{R}^3} \left(\|D^0 v(Y)\|^2 + \beta^2 \|D^1 v(Y)\|^2\right)$
- To minimize the second term, $\|D^1 v(y_{21})\|^2, \ldots, \|D^1 v(y_{30})\|^2$ should be as close to 0 as possible.

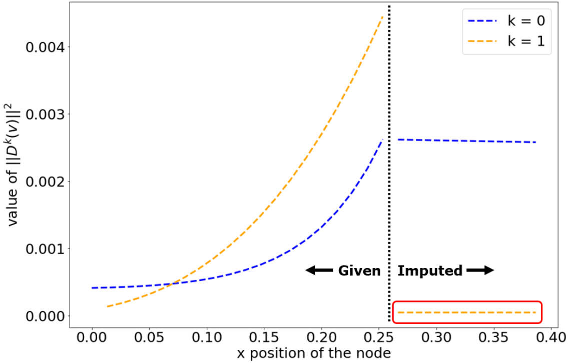


Figure 23: $\|D^m(v)\|^2$ vs. node x position plot for smoothness operator 1.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 2:

$$c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0;\ \beta = 1$$

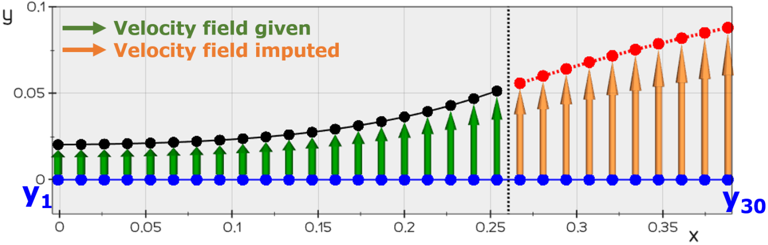


Figure 24: Velocity imputation result for smoothness operator 2.

The Motion Coherence Theory: Smoothness Operator

- Minimizing $\lambda \int_{\mathbb{R}^3} \left(\|D^0 v(Y)\|^2 + \beta^2 \|D^1 v(Y)\|^2 + \frac{\beta^4}{4} \|D^2 v(Y)\|^2 \right)$

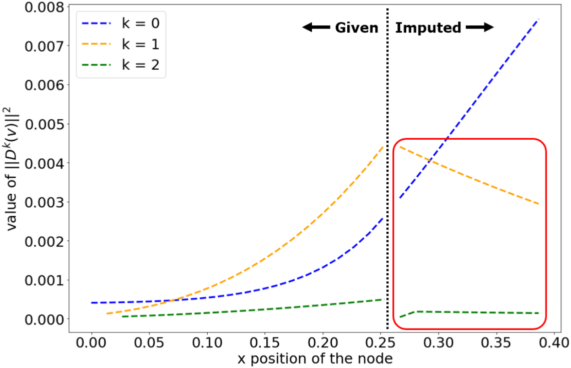


Figure 25: $\|D^m(v)\|^2$ vs. node x position plot for smoothness operator 2.

THE MOTION COHERENCE THEORY: SMOOTHNESS OPERATOR

- Minimizing $\lambda \int_{\mathbb{R}^3} \left(\|D^0 v(Y)\|^2 + \beta^2 \|D^1 v(Y)\|^2 + \frac{\beta^4}{4} \|D^2 v(Y)\|^2 \right)$

vs.

- Minimizing $\lambda \int_{\mathbb{R}^3} \left(\|D^0 v(Y)\|^2 + \beta^2 \|D^1 v(Y)\|^2 \right)$

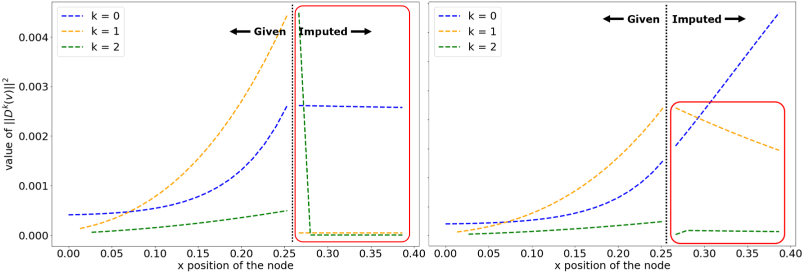


Figure 26: Plot comparison between smoothness operator 1 and 2.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 3:

$$c_0 = 1,\ c_1 = \frac{\beta^2}{2},\ c_2 = \frac{\beta^4}{8}, \ldots, c_k = \frac{\beta^{2k}}{(k!2^k)};\ \beta = 0.3$$

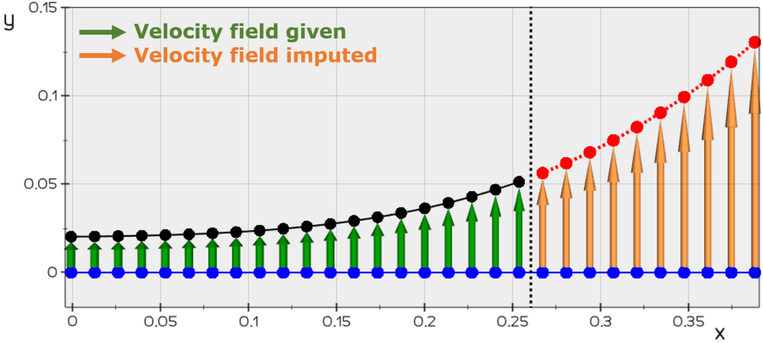


Figure 27: Velocity imputation result for smoothness operator 3.

Motion Coherence Theory: Smoothness Operator

- All the c_k values are non-zero, minimizing $\lambda \int_{\mathbb{R}^3} \sum_{k=0}^{\infty} c_k \|D^k v(Y)\|^2$
- Requiring all derivatives of v to be as smooth as possible.

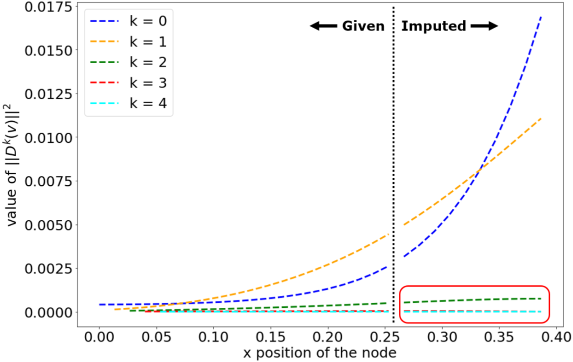


Figure 28: $\|D^m(v)\|^2$ vs. node x position plot for smoothness operator 3.

Motion Coherence Theory: Smoothness Operator

In general, the **more** higher derivatives the smoothness operator is penalizing (the more non-zero c_k terms for larger k),

- The **smoother** the overall imputation result.
- The **smaller** the values c_k can have.

Motion Coherence Theory: Smoothness Operator

- How does noise affect the performance of difference smoothness operators?
- A small amount of noise (uniform distribution from 0-1.5mm) was added in a random direction (uniform distribution from 0-360 degrees) to slightly displace the black nodes.

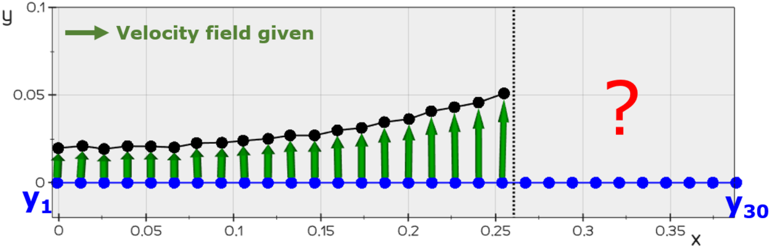


Figure 29: Velocity field with noise.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 1:

$$c_0 = 1,\ c_1 = \beta^2,\ c_2, \dots, c_k = 0;\ \beta = 16$$

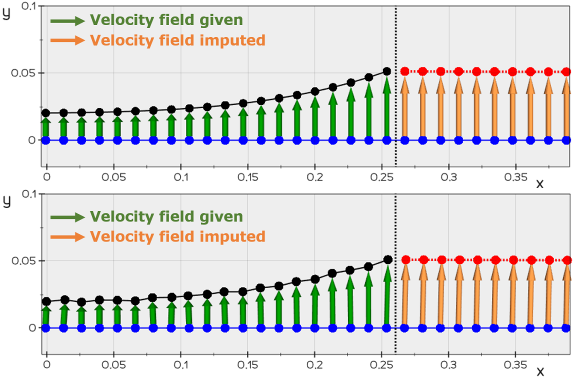


Figure 30: Top: imputation without noise. Bottom: imputation with noise.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 2:

$$c_0 = 1,\ c_1 = \beta^2,\ c_2 = \frac{\beta^4}{4},\ c_3, \ldots, c_k = 0;\ \beta = 1$$

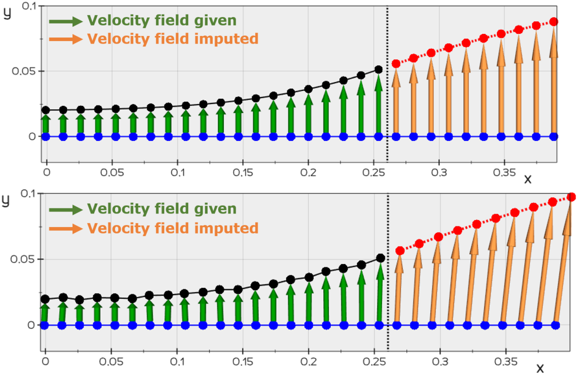


Figure 31: Top: imputation without noise. Bottom: imputation with noise.

Motion Coherence Theory: Smoothness Operator

- Smoothness operator 3:

$$c_0 = 1,\ c_1 = \frac{\beta^2}{2},\ c_2 = \frac{\beta^4}{8}, \ldots, c_k = \frac{\beta^{2k}}{(k!2^k)};\ \beta = 0.3$$

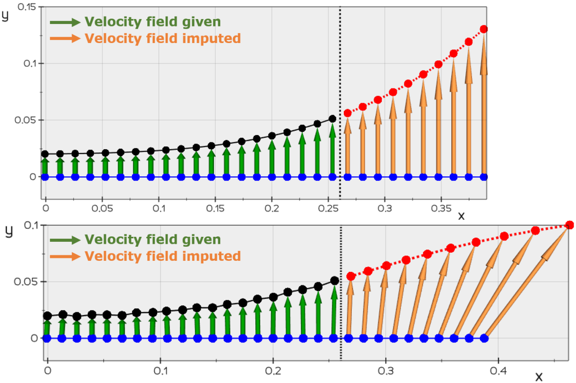


Figure 32: Top: imputation without noise. Bottom: imputation with noise.

Motion Coherence Theory: Smoothness Operator

Trade-off:

- Smoothness operators with more terms penalizing the higher derivatives leads to a smoother velocity field. **However**, such smoothness operators are also **more vulnerable** under the influence of noise.

SUMMARY

Improved DLO tracking under occlusion:

- Register the visible nodes
- Apply the Motion Coherence Theory to impute the velocities of the occluded nodes
  - Replace Euclidean distance with geodesic distance to better represent the relative position between nodes
  - Select an appropriate smoothness operator to balance between velocity field smoothness and noise rejection

Results: Stationary Object

Improved DLO tracking under occlusion

Results: Moving Object

- Tracking a rod (rigid) and a rope (non-rigid) under occlusion.
- Both objects are pinned in the middle and partially occluded. The right tip of the object was then pushed to create motion.
- The tracking parameters used for both objects are exactly the same.

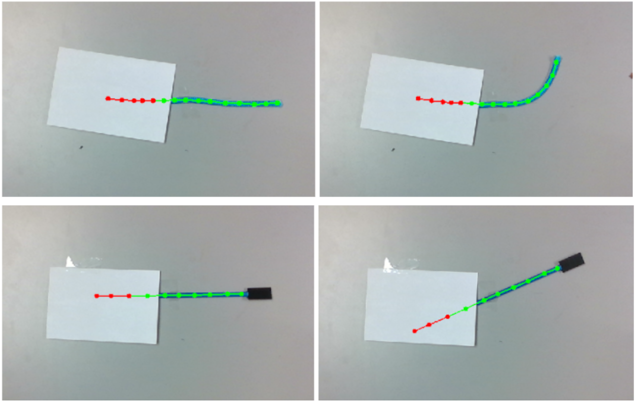


Figure 33: Top row: rope. Bottom row: rod.

References

Bishop, Christopher M et al. (1995). *Neural networks for pattern recognition*. Oxford university press.

Keipour, Azarakhsh, Maryam Bandari, and Stefan Schaal (2022). "Efficient spatial representation and routing of deformable one-dimensional objects for manipulation". In: *arXiv preprint arXiv:2202.06172*.

Lu, Bo, Henry K Chu, and Li Cheng (2016). "Dynamic trajectory planning for robotic knot tying". In: *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, pp. 180–185.

Ruan, Mengyao, Dale McConachie, and Dmitry Berenson (2018). "Accounting for directional rigidity and constraints in control for manipulation of deformable objects without physical simulation". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 512–519.

Yuille, Alan L and Norberto M Grzywacz (1989). "A mathematical analysis of the motion coherence theory". In: *International Journal of Computer Vision* 3.2, pp. 155–175.