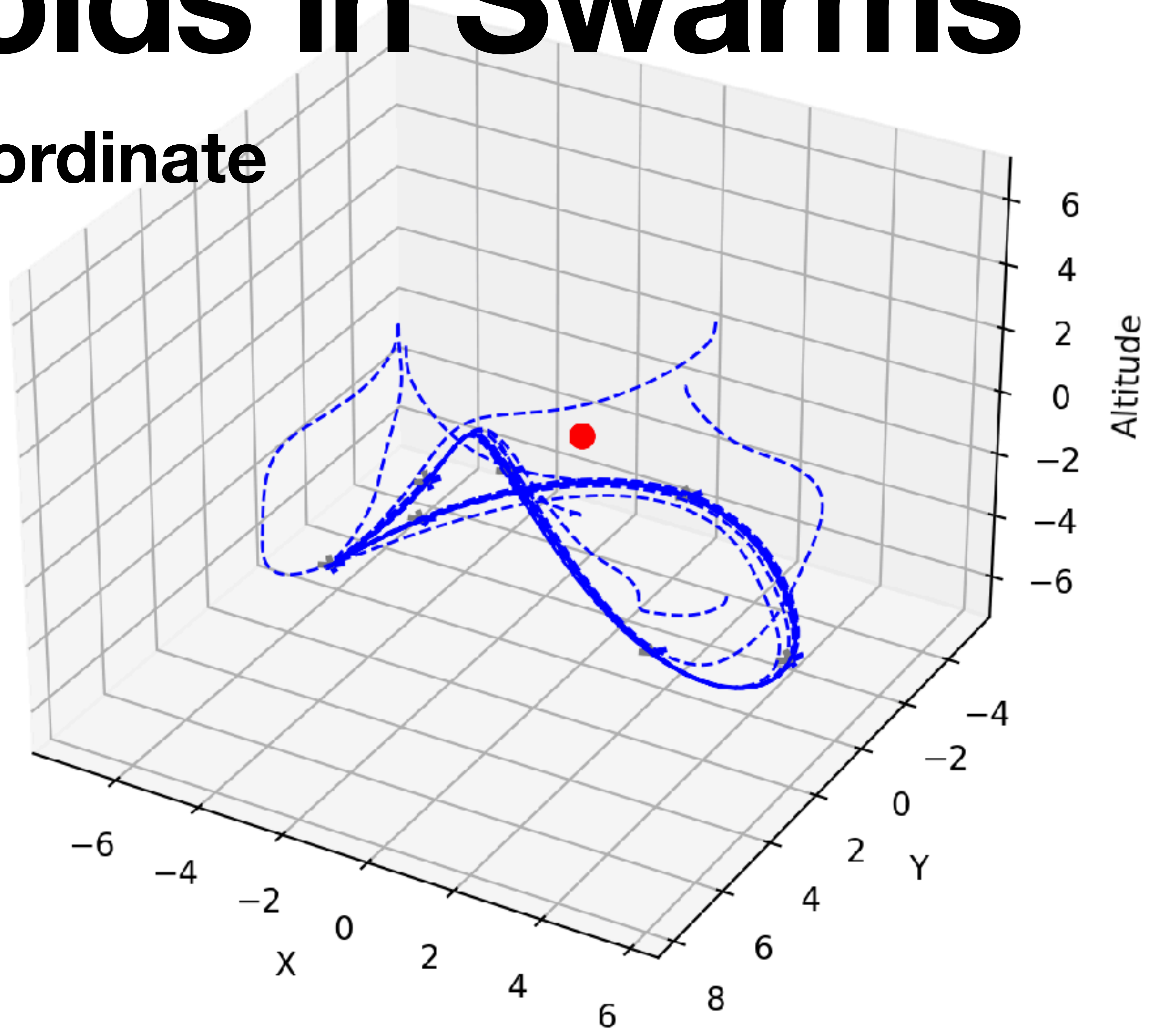
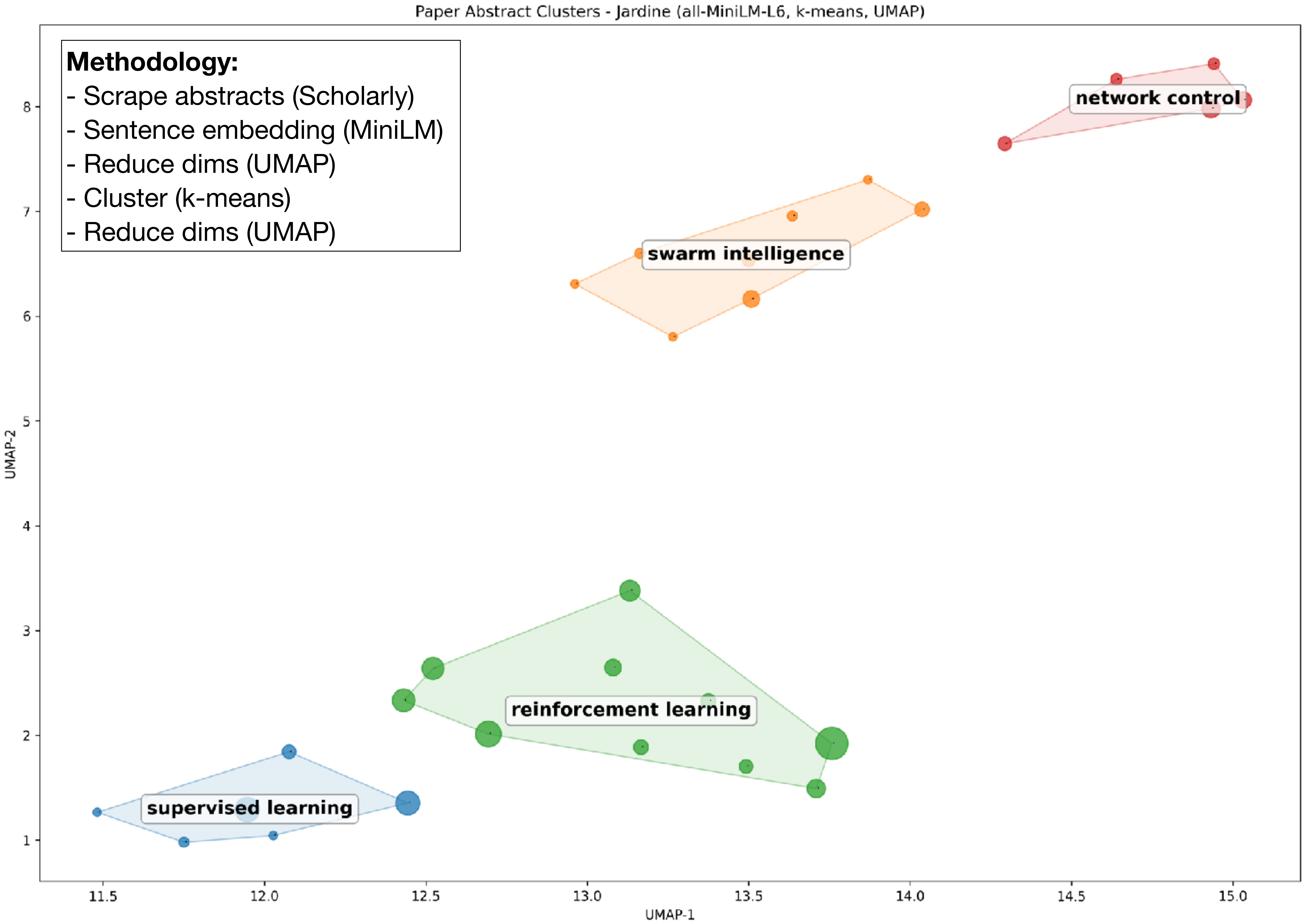
Emergent Manifolds in Swarms

Hidden Spaces for Robots to Coordinate



P. Travis Jardine, PhD
Adjunct Assistant Professor
School of Computing
Queen's University

*When **integrated** into a **control**-theoretic framework, machine **learning** can improve the performance of **physical systems** while preserving **guarantees** of stability and robustness*



Paper Abstract Clusters - Jardine (all-MiniLM-L6, k-means, UMAP)
Methodology:
- Scrape abstracts (Scholarly)
- Sentence embedding (MiniLM)
- Reduce dims (UMAP)
- Cluster (k-means)
- Reduce dims (UMAP)
network control
swarm intelligence
reinforcement learning
supervised learning
UMAP-1
UMAP-2
11.5
12.0
12.5
13.0
13.5
14.0
14.5
15.0
1
2
3
4
5
6
7
8

Outline

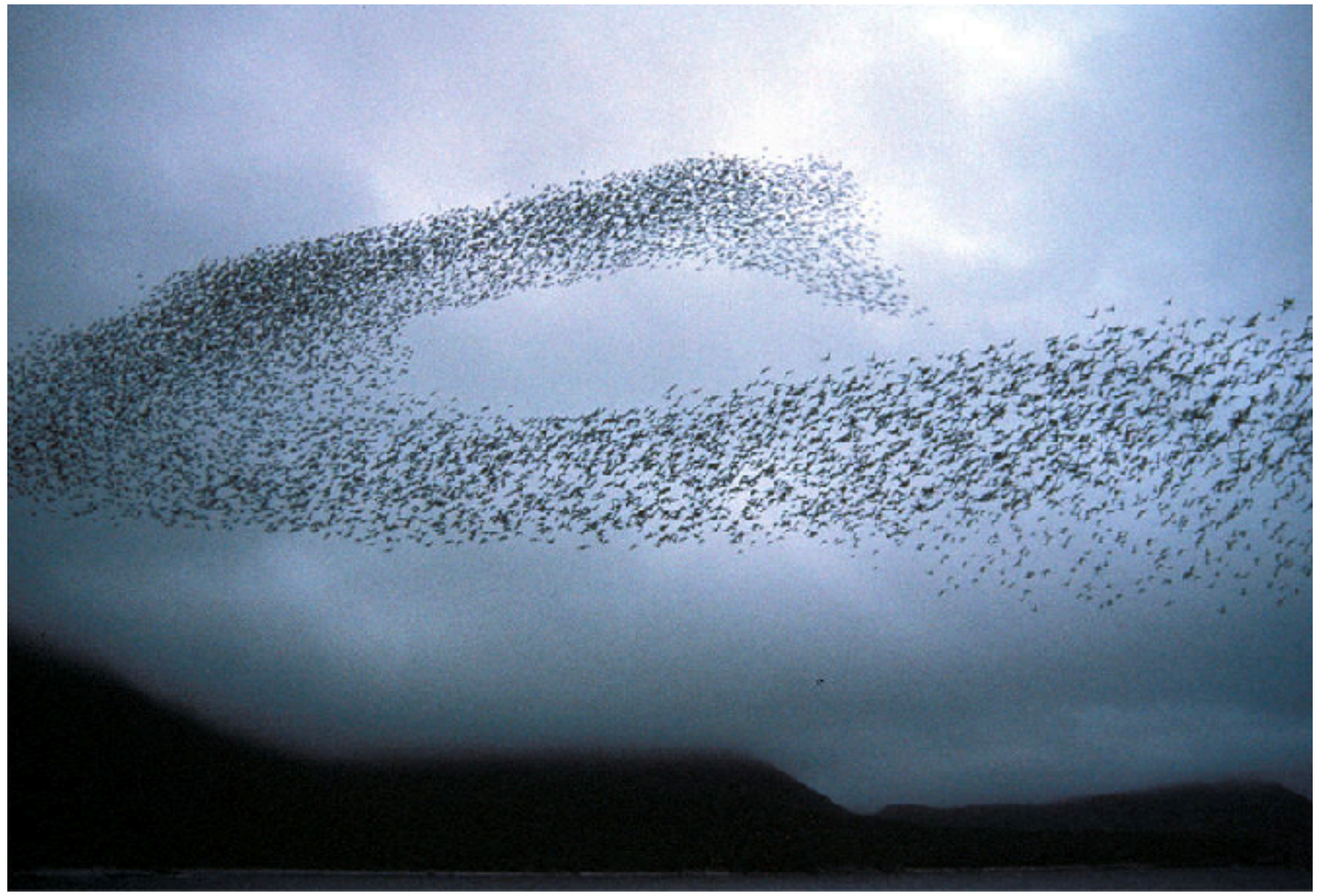
- What is swarming?
- Classical approaches
- New approaches
- Swarm embeddings
- Learning applications
- Future work

What is swarming?

Background

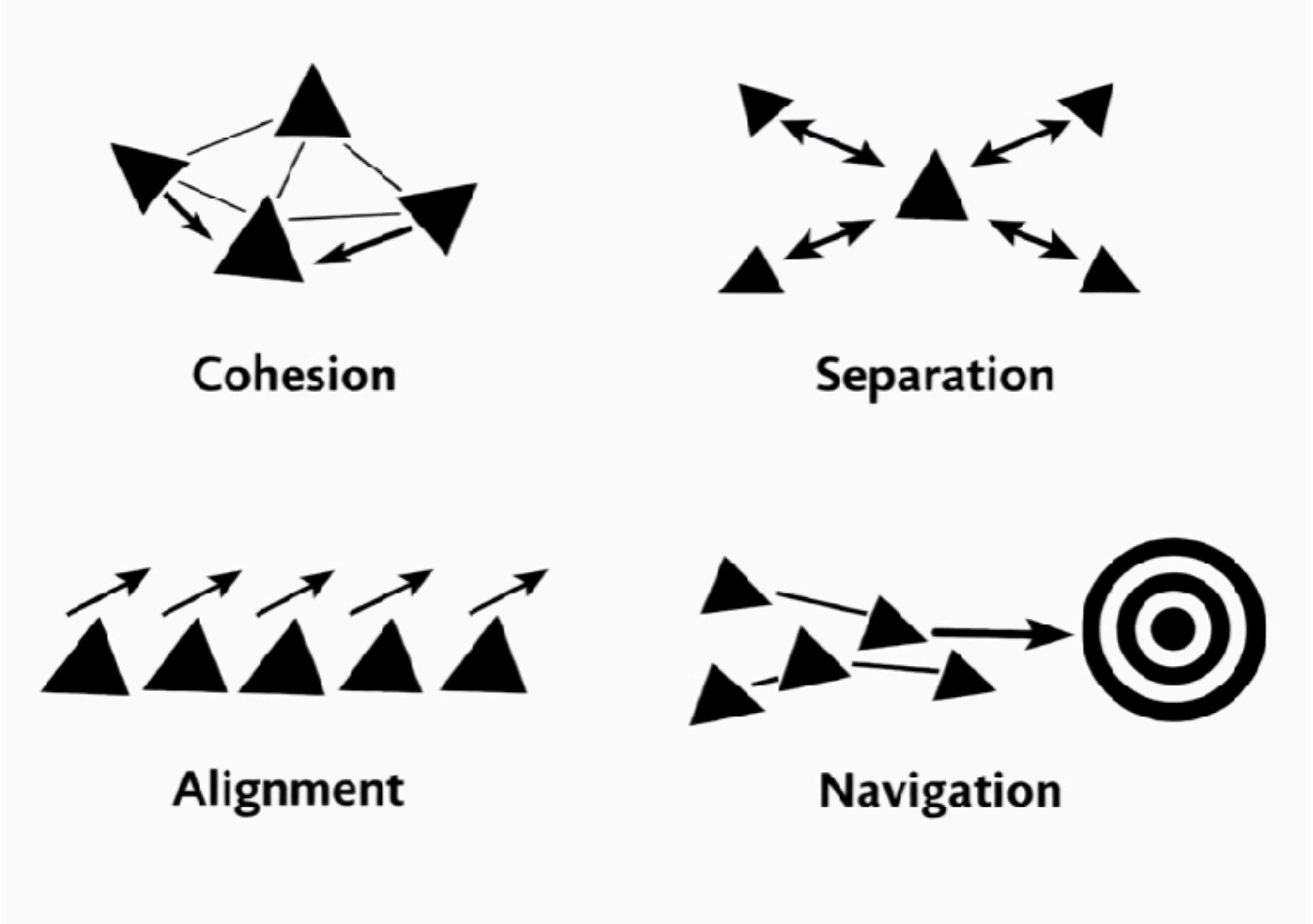
Why robots work together

- Tasks completed better, faster, or with less effort
- Centralized or decentralized
- “Swarm” robotics
  - Decentralized + asynchronous + local
  - Emergent behaviour of collective
  - Scales well
  - Simple/inexpensive individual agents
  - Robust to agent loss/corruption



Background
Flocking

- Reynolds (1987): early framework
- Coordinated motion through **cohesion**, **alignment**, and **separation**
- **Navigation** avoids fragmentation
- Control solution finds equilibrium in **position** and **velocities**



Background

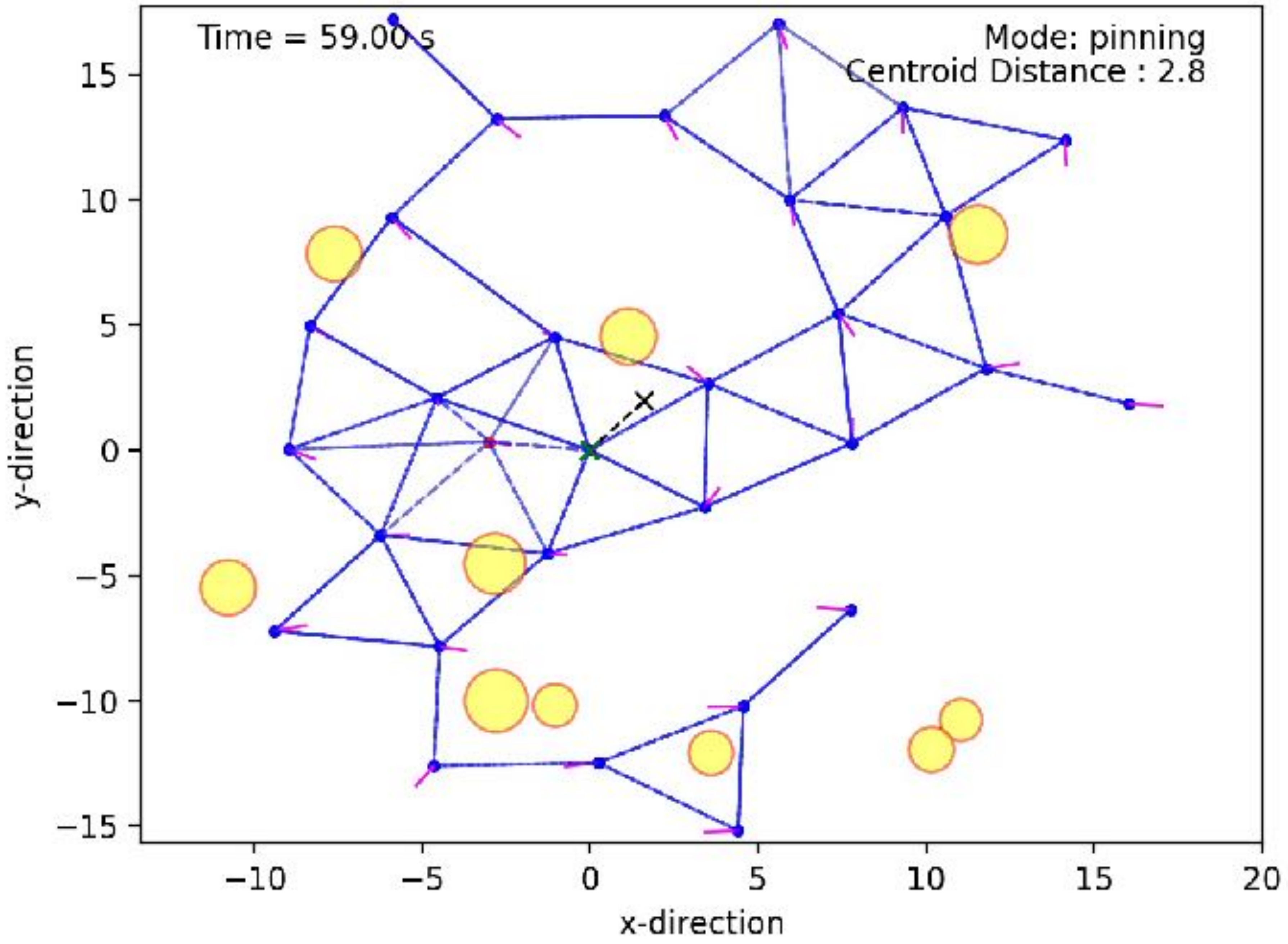
Flocking

- Olfati-Saber (2006): adapted for control theory using **gradient**- and **consensus**-based algorithms

- Assumption: equilibrium on **shared inter-agent distances** forms a lattice

- Minimization of energy **potential functions** through gradient descent

- Compatible with **Lyapunov stability** analysis



Background

Dynamic Encirclement

- Polar coordinates

$$\mathbf{x}_i(t) = \begin{bmatrix} x_i(t) \\ y_i(t) \\ z_i(t) \end{bmatrix} = \begin{bmatrix} r_d \cos(\phi_i(t)) \\ r_d \sin(\phi_i(t)) \\ 0 \end{bmatrix}$$

- Two-part controller

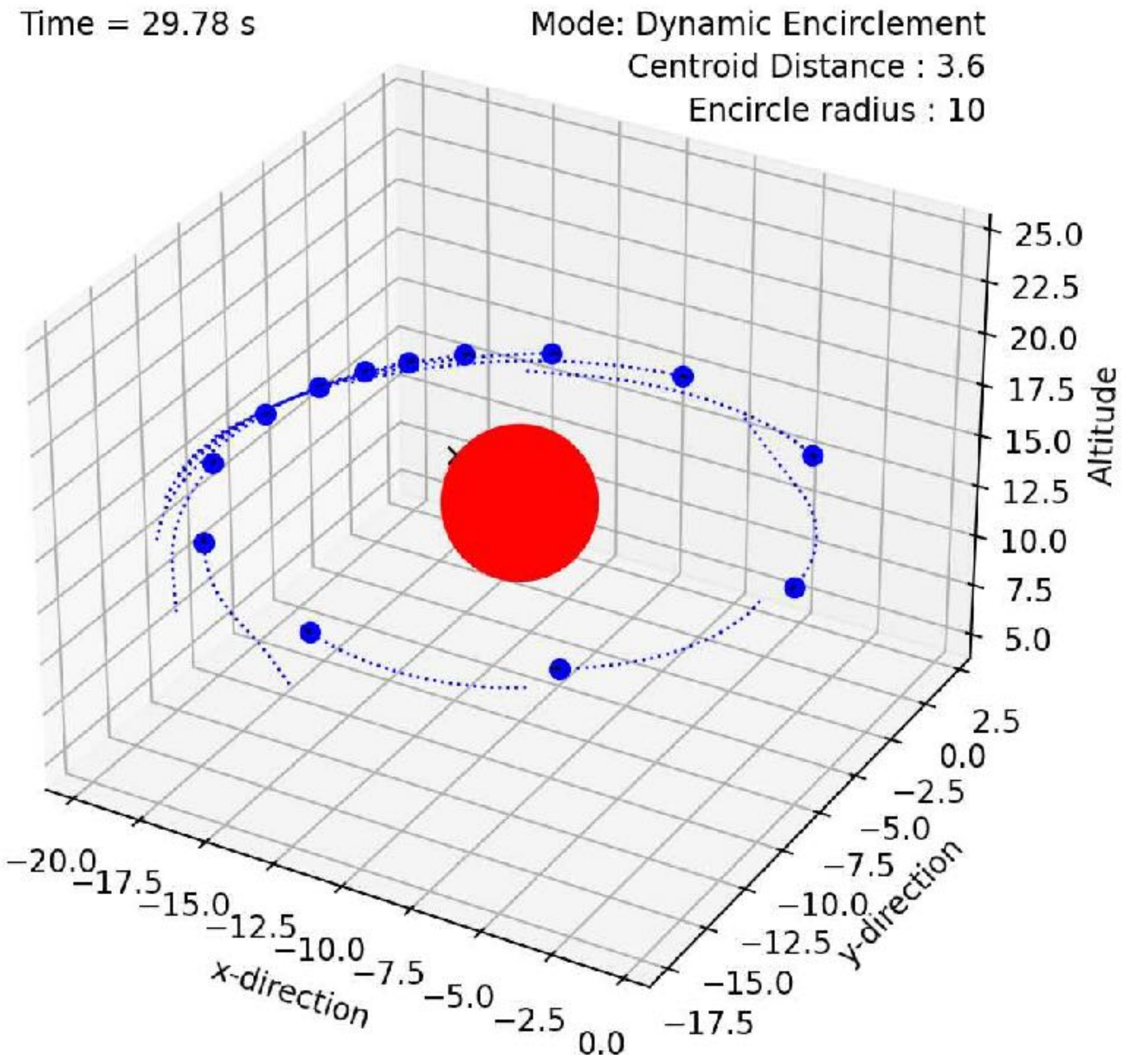
Desired, Agent, Leading/Lagging

$$\dot{\phi}_{d,i} = \dot{\phi}_d + \underbrace{\frac{k_\phi(2\phi_i - \phi_j - \phi_k)}{3}}_{\text{Angular adjustment}},$$

Desired, Agent

$$\dot{r}_{d,i} = \underbrace{-k_r(r_d - r_i)}_{\text{Radial adjustment}}.$$

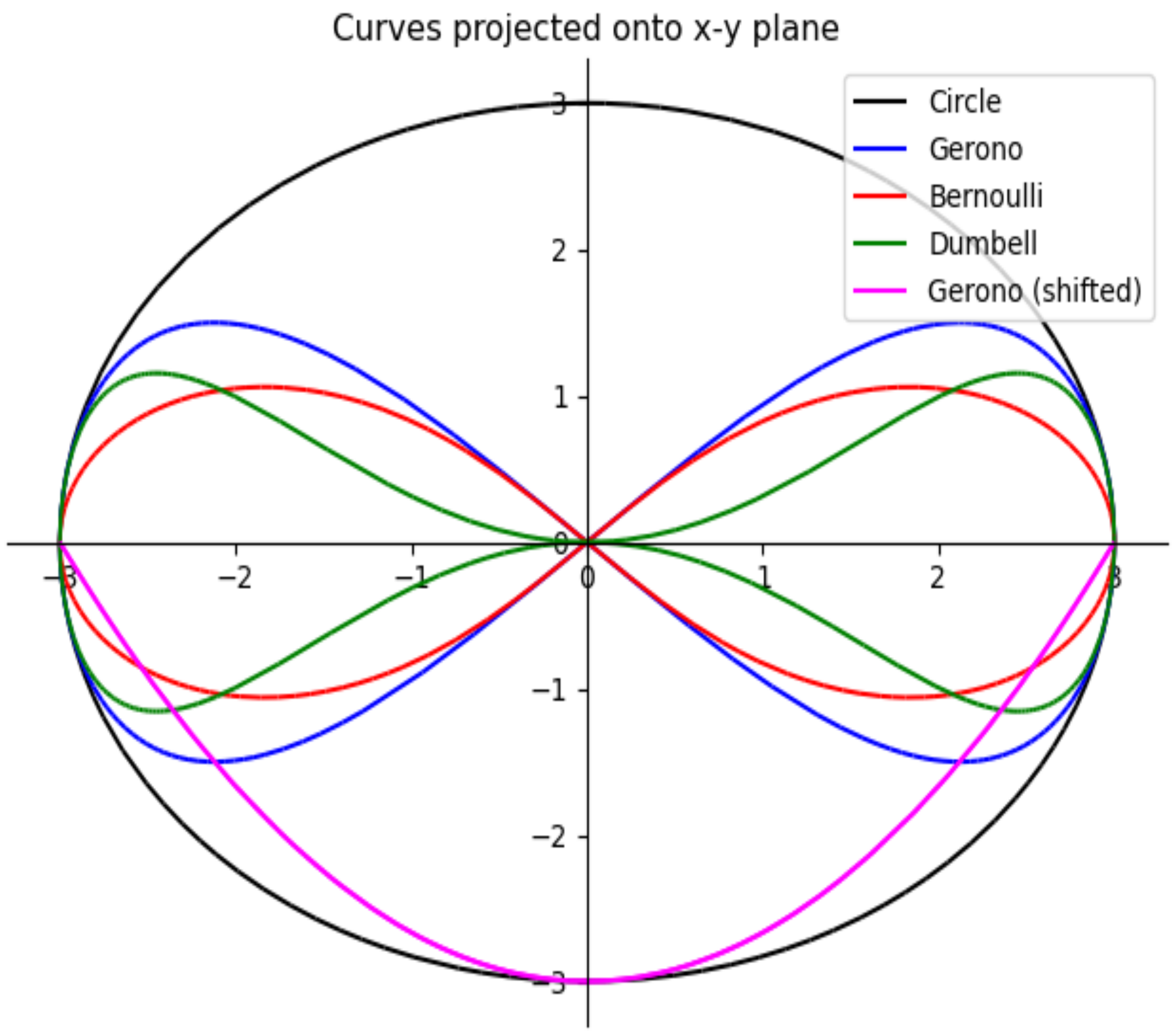
- When each agent adjusts locally, emergent trajectory is globally stable



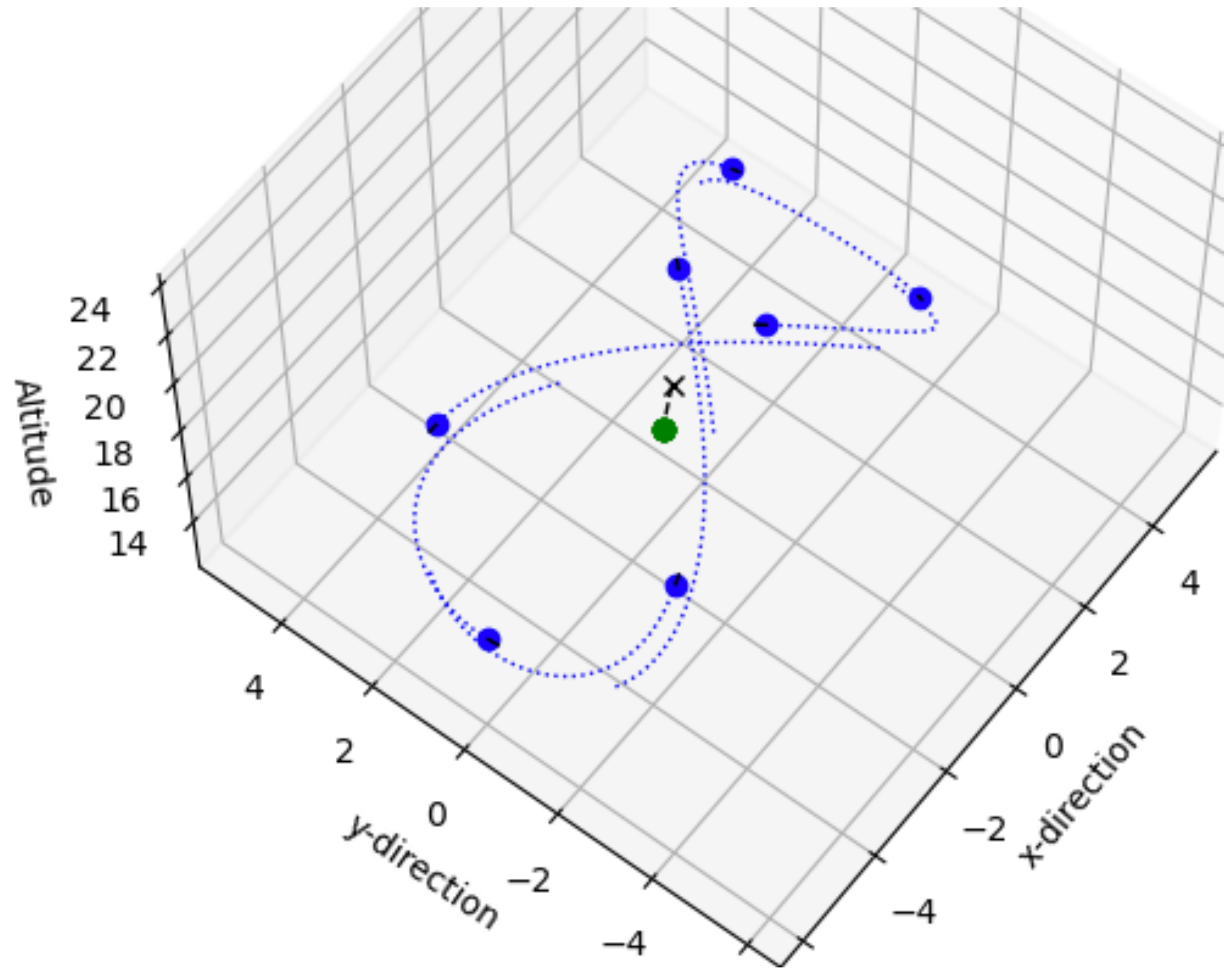
What about other shapes?

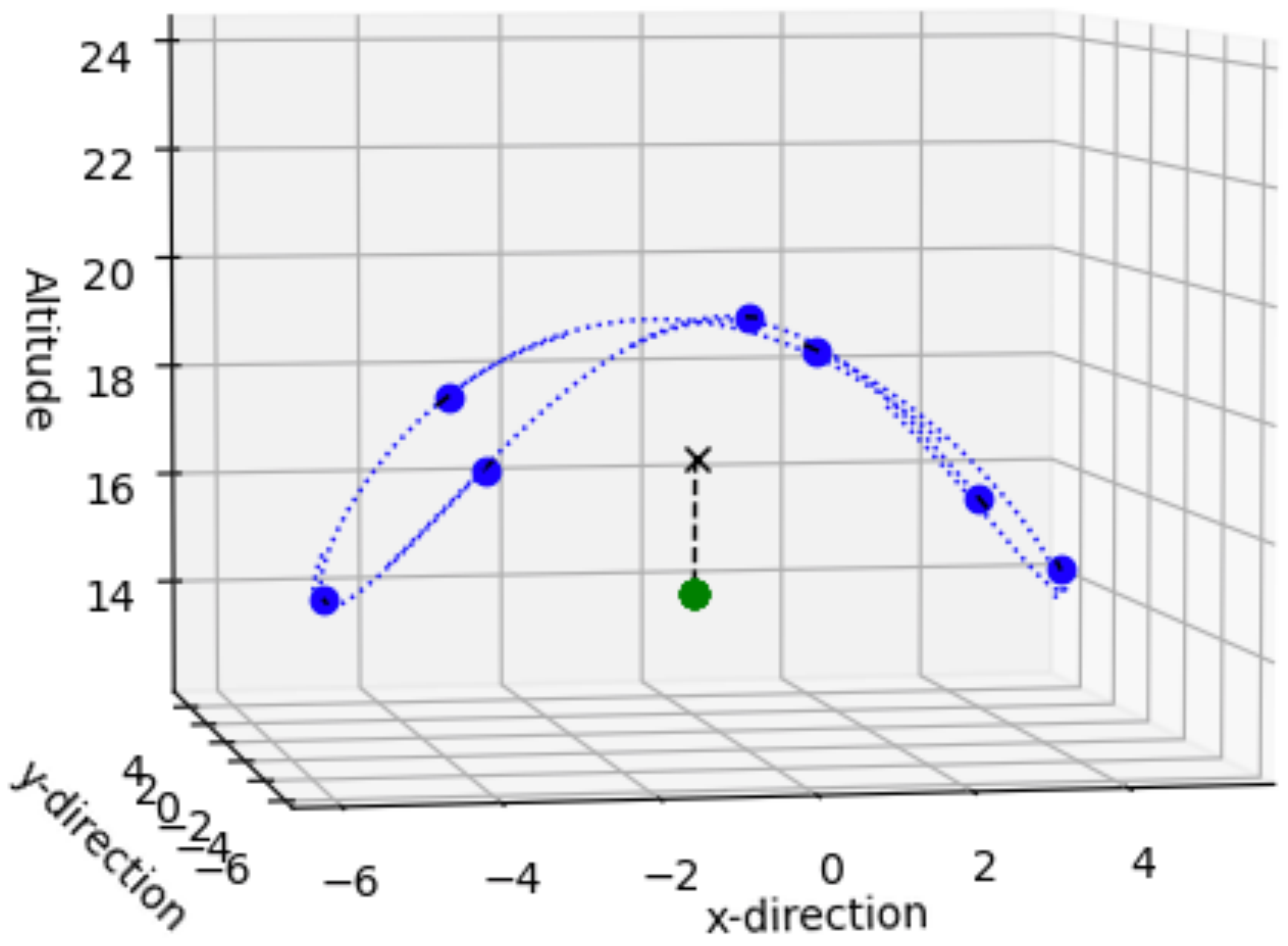
Lemniscates

- 2D Bernoulli, Gerono, Booth (1600's)
- Common applications in surveillance, sensing, communications
- Produces smooth control signals, excites various modes, low mechanical strain
- Typically predetermined trajectory/plan
- Is there a way to produce using swarming principles?



Lemniscatic Arc





Parametric Equations, $\mathbf{x}_i(t) = \begin{bmatrix} r_d \cos(\phi_i(t)) \\ r_d \sin(\phi_i(t)) \cos(\phi_i(t)) \\ 0.5 r_d \sin^2(\phi_i(t)) \end{bmatrix}$

Can we produce these using swarming?

Background

Quaternion Rotations

- Rotation around roll, pitch, yaw (ϕ, θ, ψ)
- 3D position vector converted to 4D quaternion vector

$$\bar{\mathbf{x}} = [0\ x\ y\ z]^T$$

- Compact, efficient, numerically stable
- Avoids “gimbal lock”

$$\bar{\mathbf{x}}' = \rho * \bar{\mathbf{x}} * \rho^{-1}$$

Rotation quaternion:

$$\boldsymbol{\rho} = \begin{bmatrix} c(\phi/2)c(\theta/2)c(\psi/2) + s(\phi/2)s(\theta/2)s(\psi/2) \\ s(\phi/2)c(\theta/2)c(\psi/2) - c(\phi/2)s(\theta/2)s(\psi/2) \\ c(\phi/2)s(\theta/2)c(\psi/2) + s(\phi/2)c(\theta/2)s(\psi/2) \\ c(\phi/2)c(\theta/2)s(\psi/2) - s(\phi/2)s(\theta/2)c(\psi/2) \end{bmatrix}$$

Hamilton product:

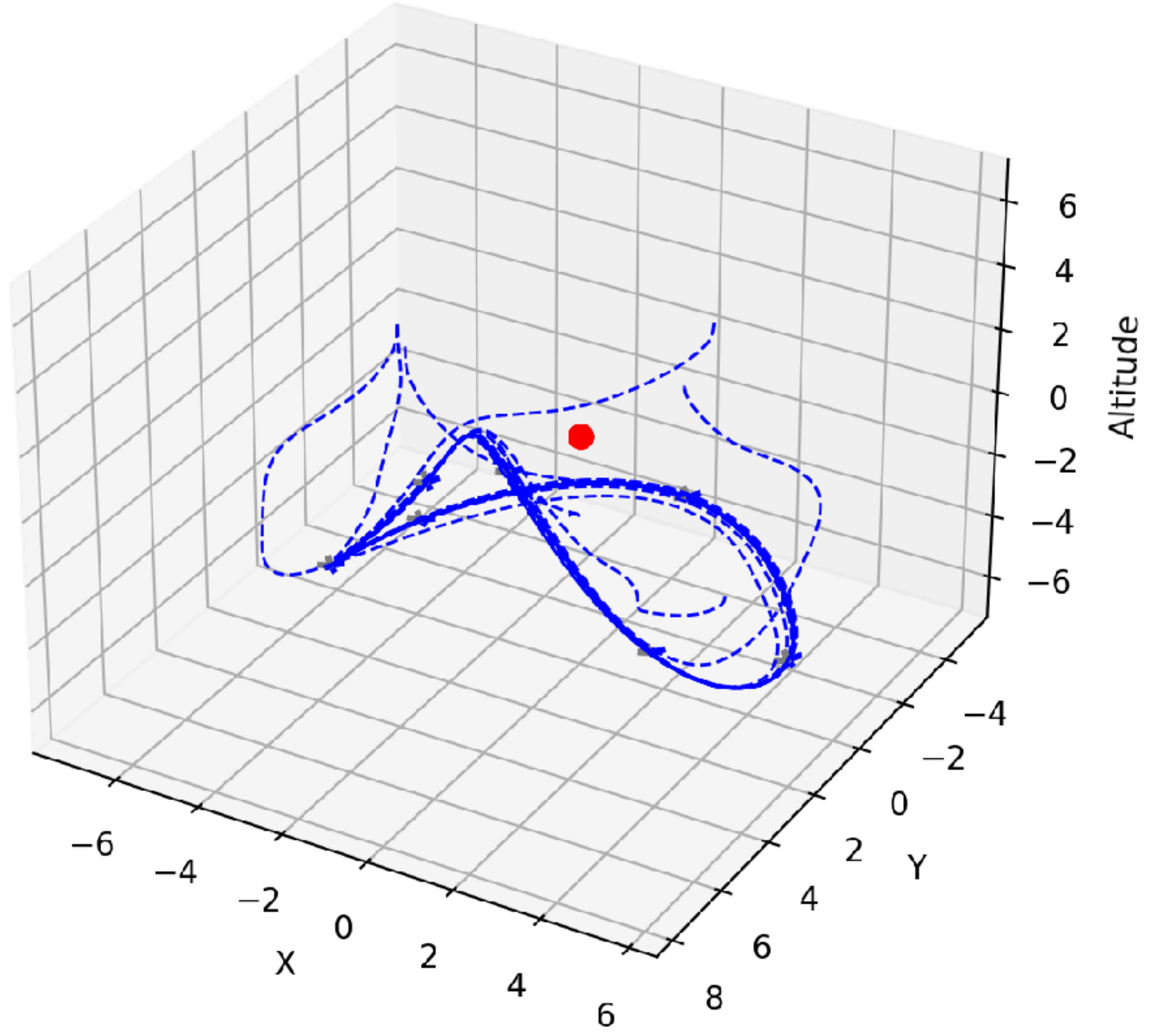
$$\boldsymbol{\rho}_0 * \boldsymbol{\rho}_1 = \begin{bmatrix} \boldsymbol{\rho}_0[0]\boldsymbol{\rho}_1[0] - \boldsymbol{\rho}_0[1]\boldsymbol{\rho}_1[1] - \boldsymbol{\rho}_0[2]\boldsymbol{\rho}_1[2] - \boldsymbol{\rho}_0[3]\boldsymbol{\rho}_1[3] \\ \boldsymbol{\rho}_0[0]\boldsymbol{\rho}_1[1] + \boldsymbol{\rho}_0[1]\boldsymbol{\rho}_1[0] + \boldsymbol{\rho}_0[2]\boldsymbol{\rho}_1[3] - \boldsymbol{\rho}_0[3]\boldsymbol{\rho}_1[2] \\ \boldsymbol{\rho}_0[0]\boldsymbol{\rho}_1[2] + \boldsymbol{\rho}_0[2]\boldsymbol{\rho}_1[0] + \boldsymbol{\rho}_0[3]\boldsymbol{\rho}_1[1] - \boldsymbol{\rho}_0[1]\boldsymbol{\rho}_1[3] \\ \boldsymbol{\rho}_0[0]\boldsymbol{\rho}_1[3] + \boldsymbol{\rho}_0[3]\boldsymbol{\rho}_1[0] + \boldsymbol{\rho}_0[1]\boldsymbol{\rho}_1[2] - \boldsymbol{\rho}_0[2]\boldsymbol{\rho}_1[1] \end{bmatrix}$$

Lemniscatic Arc

- Leverage known solution for encirclement
- Biject agents onto virtual circle using custom quaternion rotation
- Angular position $\hat{\phi}_i$ and radius r_o
- Regulate states on virtual circle using established techniques

$$\begin{bmatrix} \cos(\hat{\phi}_i/2) \\ \sin(\hat{\phi}_i/2) \\ 0 \\ 0 \end{bmatrix} * \begin{bmatrix} 0 \\ r_o \cos(\hat{\phi}_i) \\ r_o \sin(\hat{\phi}_i) \\ 0 \end{bmatrix} * \begin{bmatrix} \cos(\hat{\phi}_i/2) \\ -\sin(\hat{\phi}_i/2) \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} r_o \cos(\hat{\phi}_i) \\ r_o \sin(\hat{\phi}_i) \cos(\hat{\phi}_i) \\ 0.5 r_o \sin^2(\hat{\phi}_i) \end{bmatrix}$$

Quaternion | Circle | Quaternion Inverse | Lemniscate

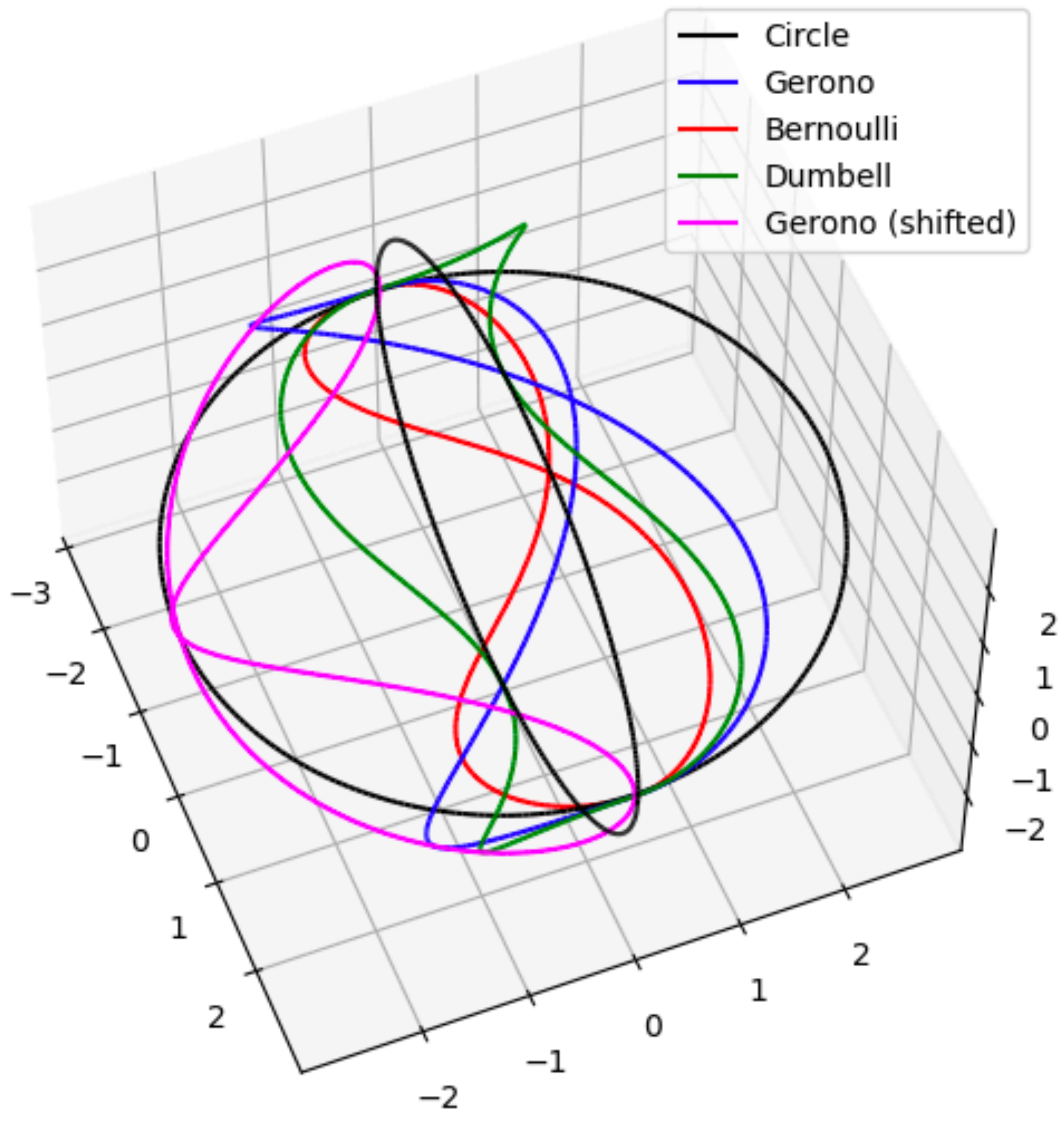


Can we generalize this?

Swarming

Generalized Curves

Examples:



Highlights:

- Form curves from “twisting” circle
- Swarming principles
- Defined as functional variations on circle
- Conceptualized as control on a geometric embedding
- Reduce dimensionality from 3D to 2D
- Provably stable

Parametric equations:

$$\begin{bmatrix} r(\alpha(\phi)^2 + \beta(\phi)^2)\cos(\phi) \\ r(\alpha(\phi)^2 + \beta(\phi)^2)\sin(\phi) \\ 2r\alpha(\phi)\beta(\phi)\sin(\phi) \end{bmatrix}$$

Radius (scale)

Angular position

Functional parameters define geometry

Swarming

Generalized Curves - Formulation

- Derivation

$$\begin{bmatrix}\alpha(\phi)\\ \beta(\phi)\\ 0\\ 0\end{bmatrix} * \begin{bmatrix}0\\ r\cos(\phi)\\ r\sin(\phi)\\ 0\end{bmatrix} * \begin{bmatrix}\alpha(\phi)\\ -\beta(\phi)\\ 0\\ 0\end{bmatrix} = \begin{bmatrix}0\\ r(\alpha(\phi)^2+\beta(\phi)^2)\cos(\phi)\\ r(\alpha(\phi)^2+\beta(\phi)^2)\sin(\phi)\\ 2r\alpha(\phi)\beta(\phi)\sin(\phi)\end{bmatrix}$$

Quaternion — Embedding — Inverse — Desired shape

- **Stabilize** states on circular embedding using established techniques

$$\lim_{t\to\infty}\hat{\boldsymbol{x}}(t) = \begin{bmatrix}r_d\cos(\dot{\phi}_d t)\\ r_d\sin(\dot{\phi}_d t)\\ 0\end{bmatrix}$$

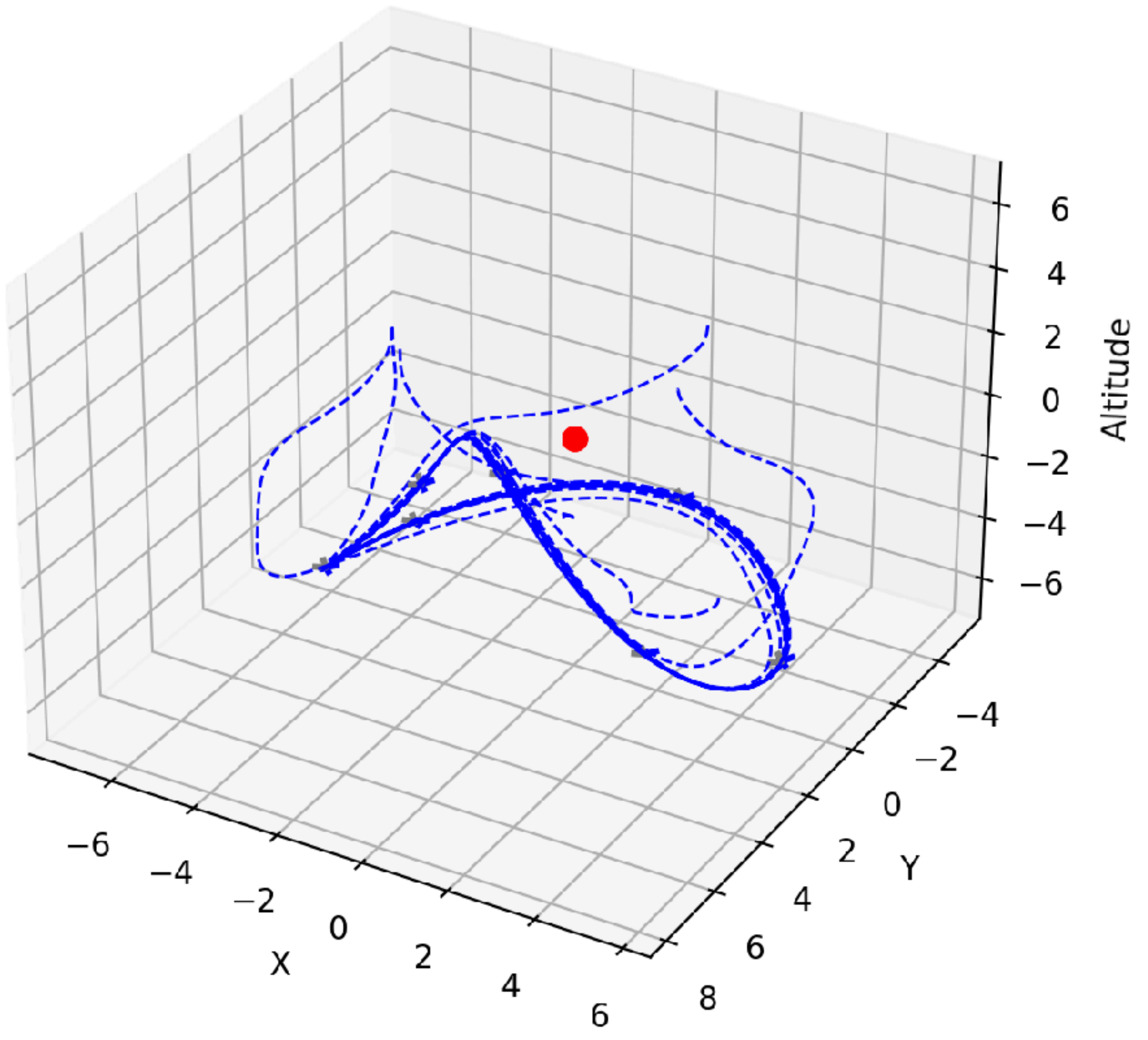
Desired radius (r_d) — Desired speed ($\dot{\phi}_d$)

- Apply transformation locally to produce desired shape

- Global trajectory is **homeomorphic** to circular stable embedding

Swarming

Generalized Curves - Results



video

- Implemented on simulated quadcopter dynamics
- 7 agents initiated in random locations
- Quickly converge to desired geometry

Novel contributions

- Low-dimensional latent representations for swarm coordination
- Topological approaches to collective stability
- Integration of geometric embeddings and control-theoretic structure

Can we apply learning?

Learning

- Consider the earlier dynamics as driving an equilibrium manifold
- Collective structure is emergent, leaderless
- Can we optimize the emergent structure using learning?
- Example:
  - Learn orientation of swarm structure using reinforcement learning
  - Adjust orientation of embedding (Altitude/Azimuth)
  - Rewards coordinated by leader agent

Learning

Formulation

- Reinforcement Learning: Continuous Action Learning Automata (CALA)
- Maintain normal distribution over action space, $a_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$
- Mean μ_t and variance σ_t^2 are updated through rewards $r(a_t)$

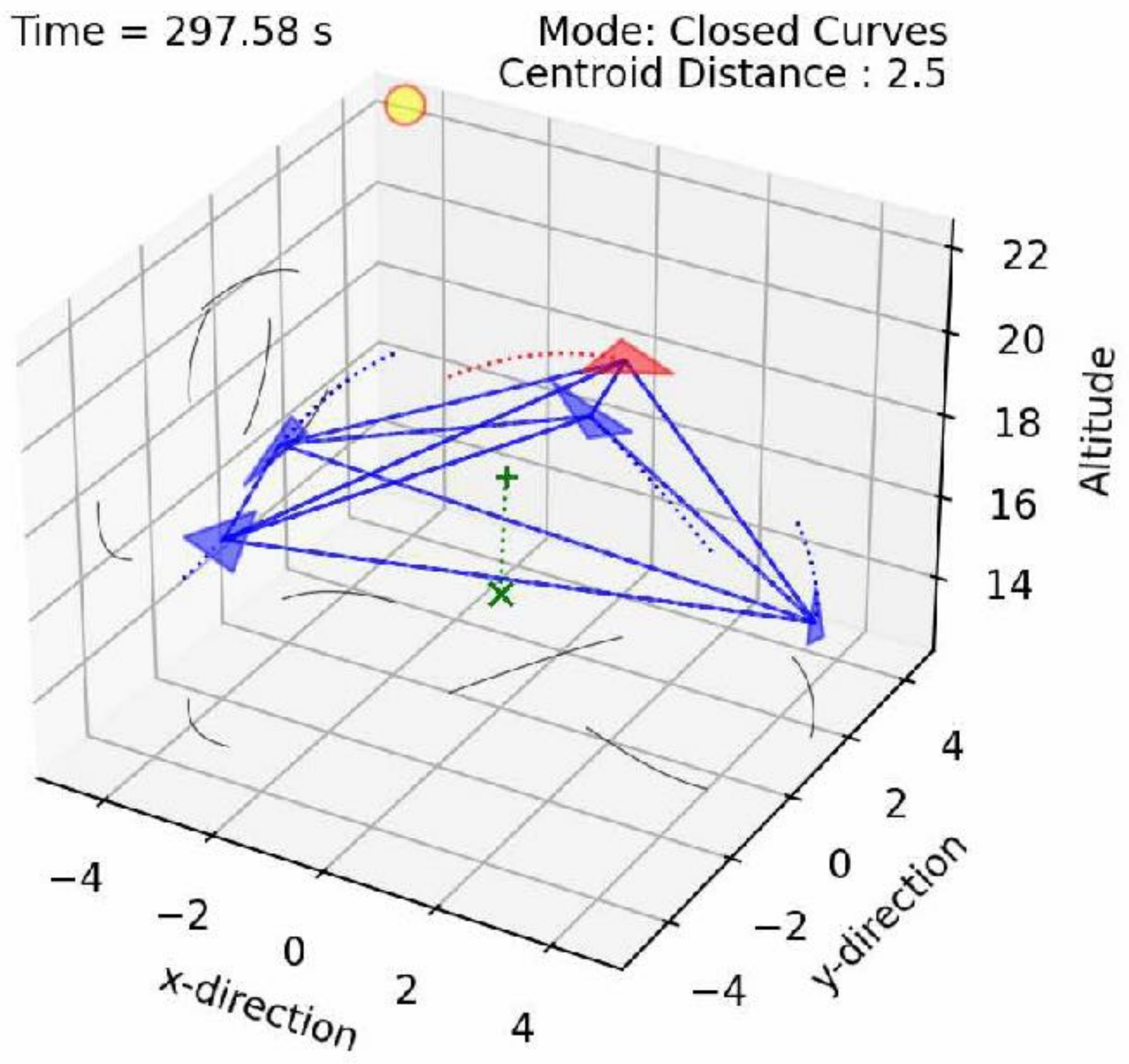
$$\mu_{t+1} = \mu_t + \lambda[r_t(a_t) - r_t(\mu_t)][a_t - \mu_t]$$

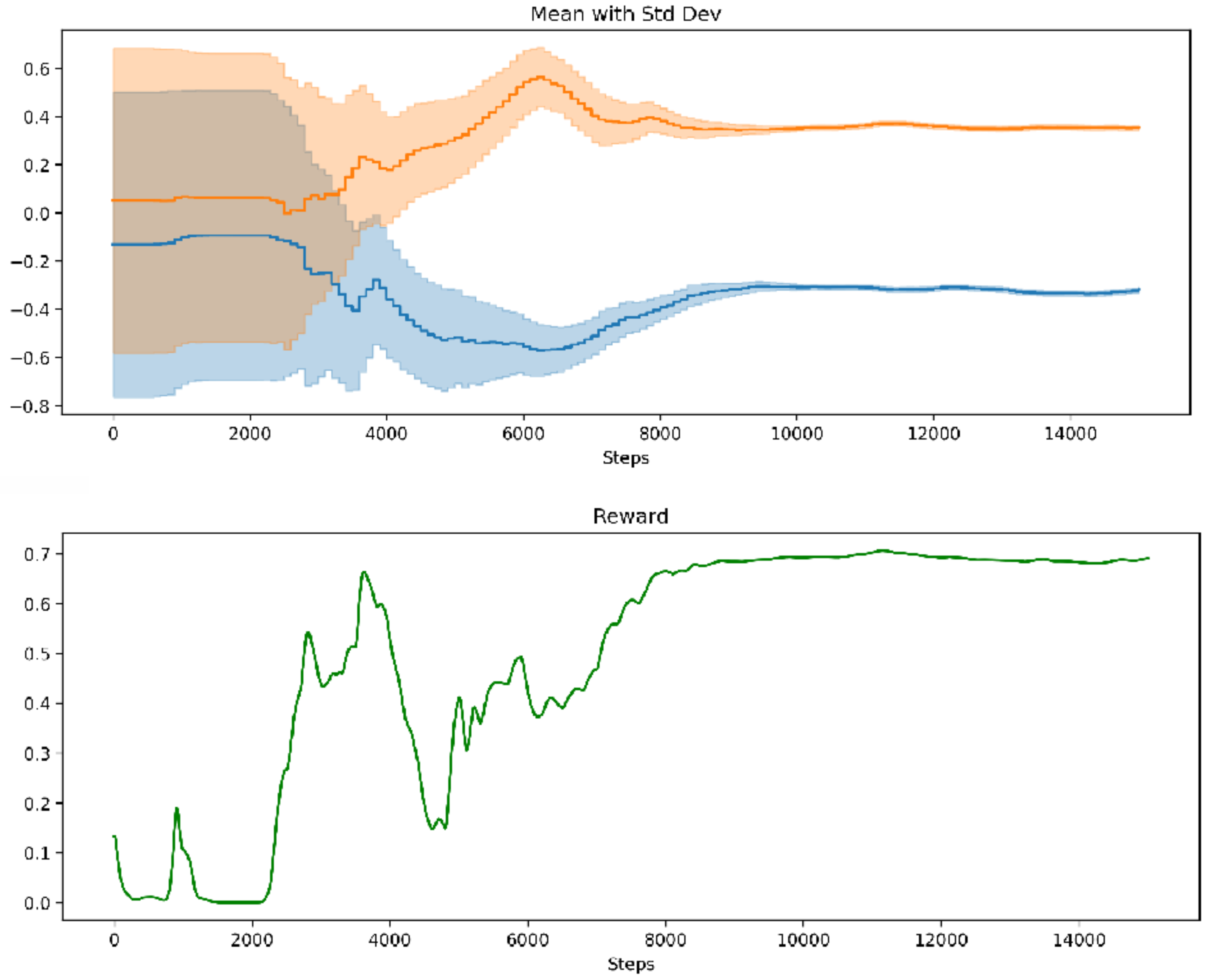
$$\sigma_{t+1}^2 = \sigma_t^2 + \lambda[r_t(a_t) - r_t(\mu_t)][(a_t - \mu_t)^2 - \sigma_t^2]$$

- Where λ controls the learning rate
- We used this to orient the swarm towards a target while maintaining structure

Learning

Initial results





Note: Red is leader agent

Learning

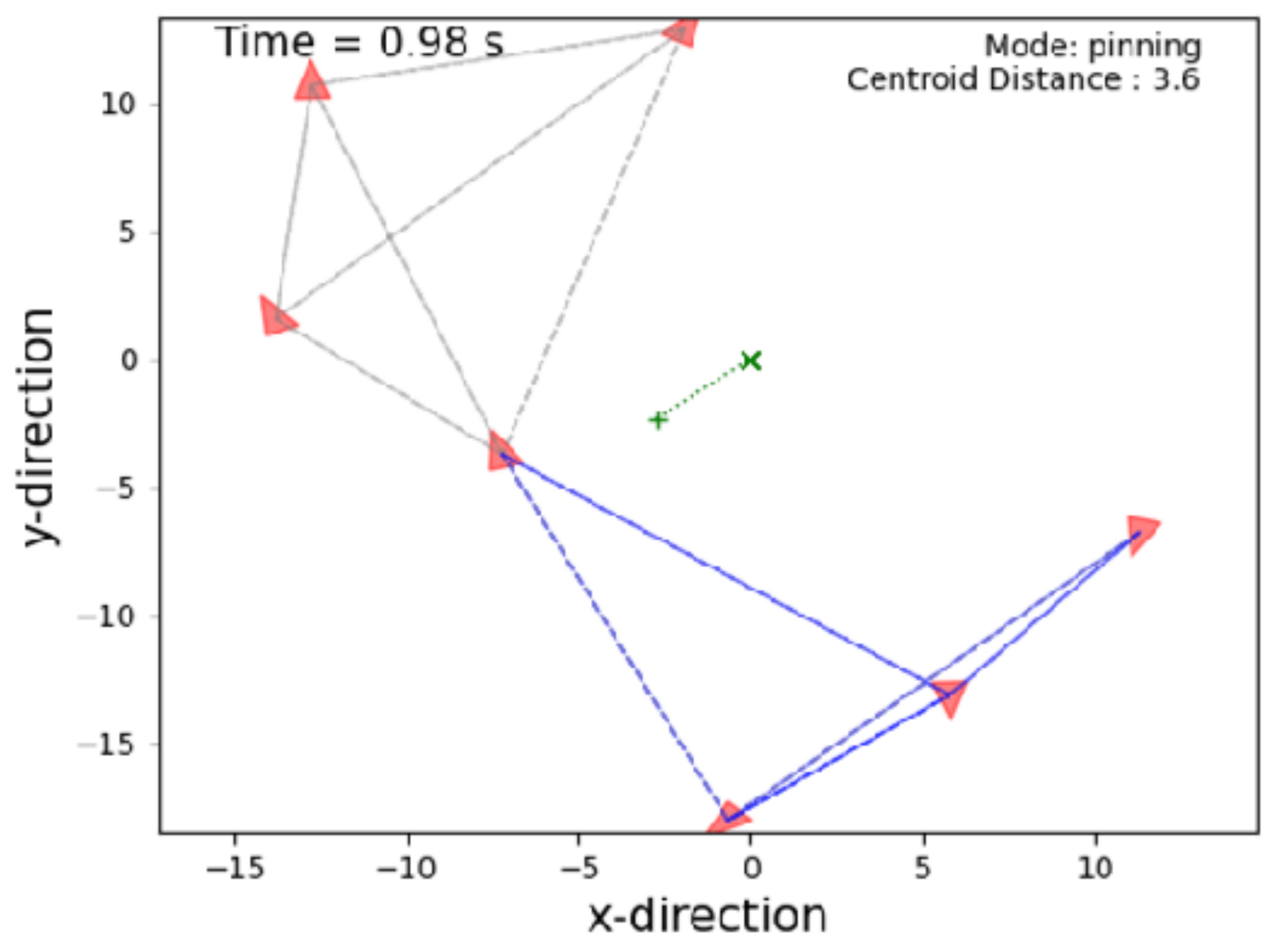
Future work

- Not truly centralized yet, as all agents need to agree on a leader
- Fully decentralized risks instability, as agents draw from different distributions
- Investigating consensus-based approaches to sharing learning results

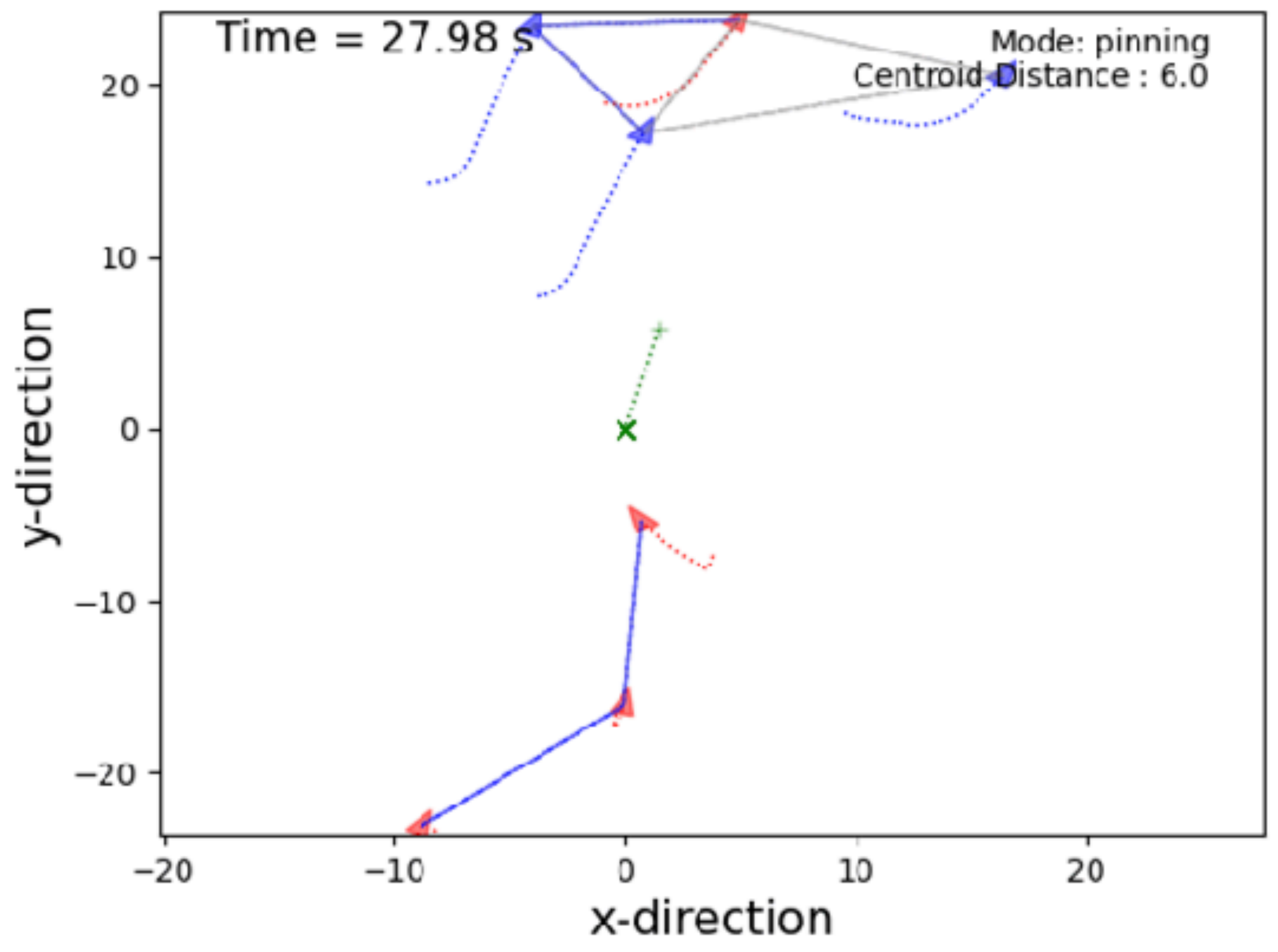
Future Work

Flocking manifolds

- Recall: flocking assumes shared inter-agent distances
- Stability guarantees rely on this assumption (else, flock collapses)
- Relaxing this constraint could permit greater diversity of agents and/or optimization of geometry
- Interesting application of learning to determine local inter-agent distances
- How to learn without sacrificing stability?



(a) Initial state



(b) Flock collapses

Future Work

Flocking manifolds

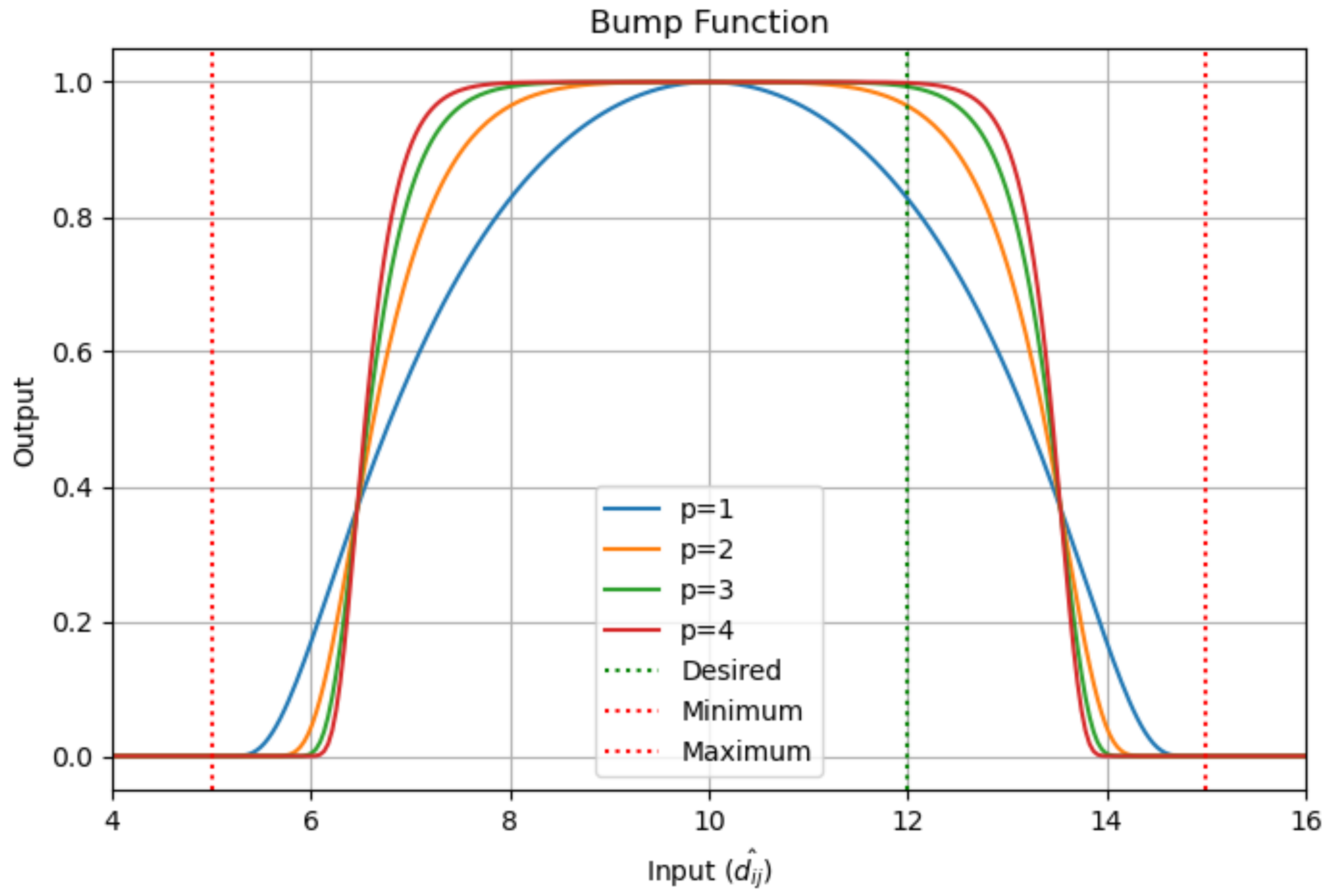
- Negotiate agreement on inter-agent distances

Parameter (d_{ij}) — Estimated ($\hat{d}_{ji}$)

$$V_{ij}^d(d_{ij}, \hat{d}_{ji}) = \frac{1}{2} k_d V_i^b(\hat{d}_{ji})(d_{ij} - \hat{d}_{ji})^2,$$

$$\dot{d}_{ij} = -\nabla_{d_{ij}} V_{ij}^d(\cdot)$$

- Agents adjust their desired separation to minimize potential
- Relies on estimated parameters of neighbour (derived from filter)

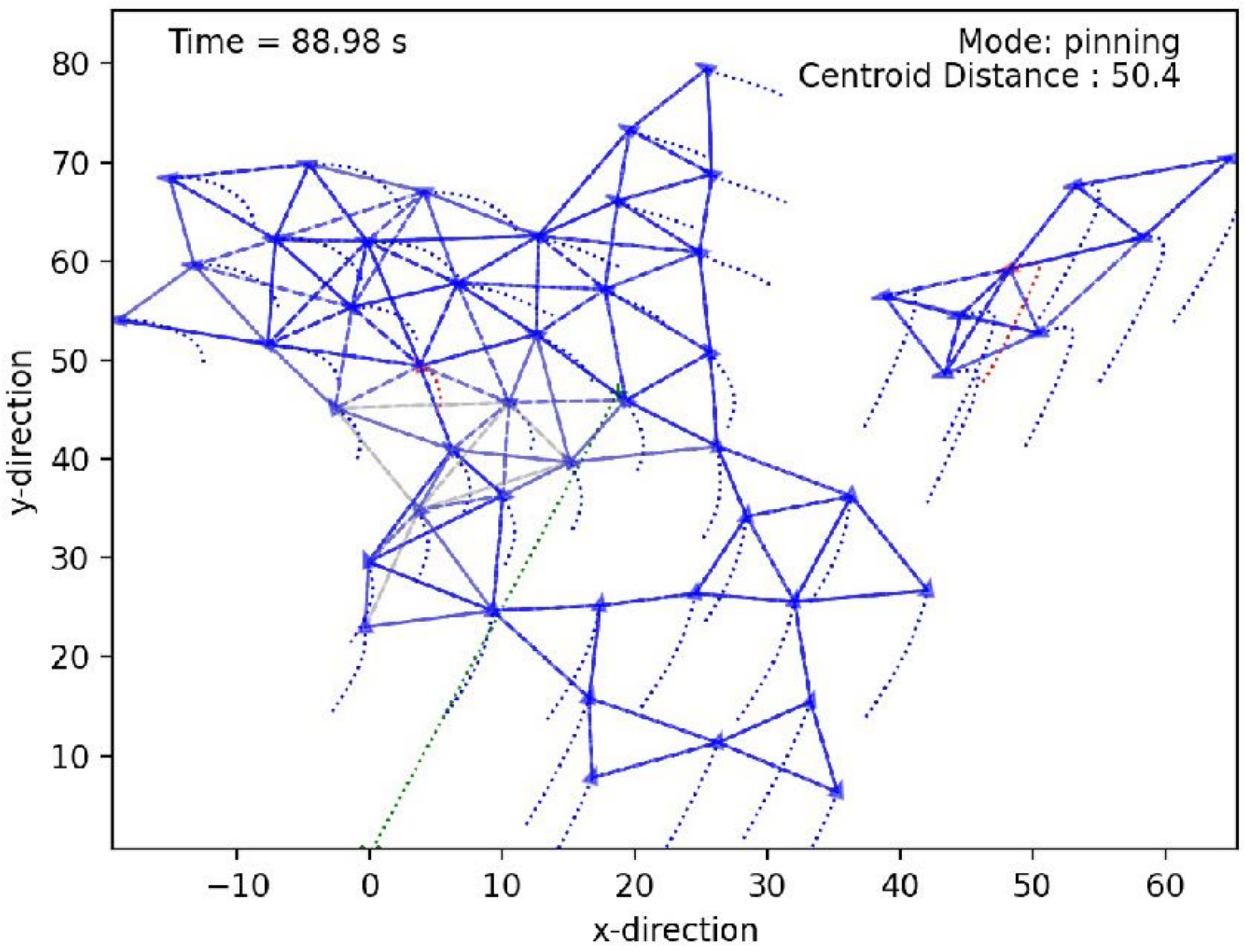


$V_i^b(\cdot)$: 'bump' function that enforces constraints

Future Work

Flocking manifolds

- Initial results promising



References

Our related work

- Jardine and Givigi, Flocks, Mobs, and Figure: Swarming as a Lemniscatic Arch, published in *IEEE Transactions on Network Science and Engineering*, 2023
- Jardine and Givigi, Emergent Homeomorphic Curves in Swarms, published in *Automatica*, 2025
- Silveria, Cabral, Jardine, and Givigi, Decentralized Swarm Control via SO(3) Embeddings for 3D Trajectories, published in *IEEE Robotics and Automation Letters*, 2025
- Jardine and Givigi, Agree to Disagree: Consensus-Free Flocking under Constraints, Proceedings of IEEE SYSCON, 2026

Conclusion

- Robots don’t need to coordinate in physical space
- By coordinating in latent embeddings, complex trajectories can emerge
- Simple, decentralized rules and local information
- Generates equilibrium manifold on which other activities can occur
- Future applications in multi-agent learning