LEARN TO RECONFIGURE: SELF-SUPERVISED SOLUTION OPERATORS FOR MODULAR ROBOTIC SWARMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Can deep learning enable fast and safe planning of modular robotic swarms across a vast variety of shape reconfigurations? Modular robotic swarms must change shape quickly while maintaining connectivity throughout the motion. These coupled constraints make planning expensive and restrict exploration in reinforcement learning. We propose a self-supervised solution operator that learns across shape pairs by differentiating task and constraint losses through the swarm dynamics, without demonstrations or precomputed optimal solutions. A graph transformer represents individual modules and their evolving interactions, producing one velocity per module per control period. A fixed projection then maps the learned proposal toward the admissible set, and an independent verifier certifies collision avoidance, connectivity, speed limits, and arrival within a stated tolerance. On two planar benchmarks, the pipeline certifies 71.7% of tasks, compared with 0.8% for a transport baseline and none for an admissible-action reinforcement learning baseline. Three-dimensional experiments demonstrate the same pipeline with up to 1,000 modules.

1 INTRODUCTION

A swarm of identical modules that rearranges itself into any task-demanded shape is a long-standing goal of robotics (Yim et al., 2007; Seo et al., 2019). We want one set of hardware to serve many purposes. In orbit, a spacecraft assembled from interchangeable components can be re-formed for a new mission without a second launch (Foust et al., 2020; Liao et al., 2025; Gregg et al., 2024). Indoors, modules that assemble into a chair or a table on demand are adaptive furniture (Spröwitz et al., 2014; Hauser et al., 2020), and collectives that build and rebuild around their occupants are adaptive architecture (Petersen et al., 2019). At the micron scale, magnetic micro-robot swarms switch formation to pass a constriction or carry a load (Xie et al., 2019; Dong & Sitti, 2020). In each case, the reconfiguration has to be fast, and it has to be safe.

During reconfiguration, modules need to coordinate to make room for other modules without breaking the contact network that supports the assembly. Collision avoidance is pairwise, whereas connectivity depends on the entire swarm. Preserving both constraints throughout a fast motion is difficult while matching the final shape. The challenge grows as the number of modules and the variety of target shapes increase.

Existing approaches address different aspects of this problem. Trajectory optimization can explicitly incorporate dynamics and constraints, but it generally requires solving a new optimization problem for every task, making online planning costly. Mean-field control and optimal transport provide scalable descriptions of large populations in terms of distributions (Carmona & Delarue, 2018; Benamou & Brenier, 2000; Cardaliaguet et al., 2016), but realizing these distributions with finitely many modules still requires resolving individual collisions and contacts. Reinforcement learning (RL) amortizes decision-making across tasks, yet learning coordinated motions from sampled rewards is difficult when successful reconfiguration depends on long sequences of tightly coupled actions. Moreover, reward penalties alone do not guarantee safety. Action masks and safety shields can enforce admissibility (Alshiekh et al., 2018; Dobreva et al., 2025), but overly restrictive action sets may hinder exploration and impede progress toward the target.

We therefore choose self-supervised solution-operator learning, which combines amortized planning with explicit use of the known dynamics. A shared neural network learns a map from the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

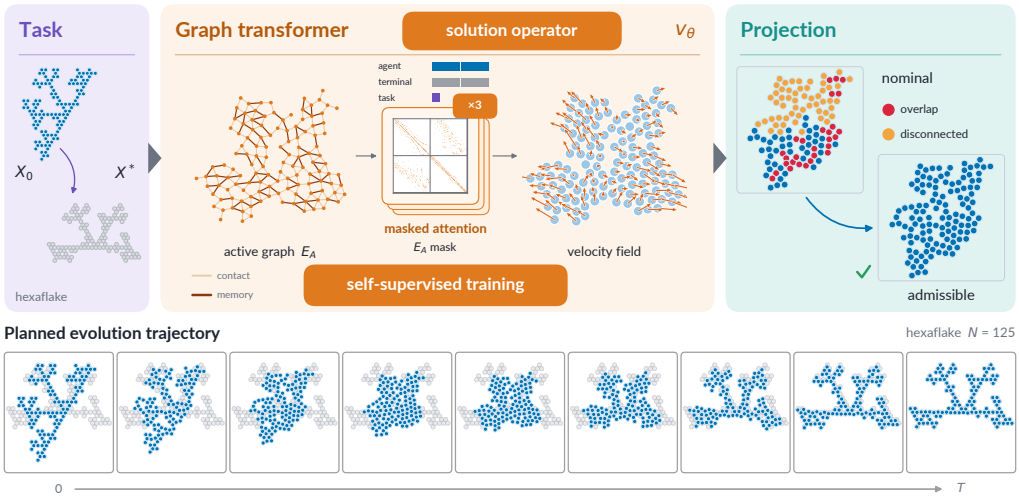

Figure 1: Reconfiguration pipeline: the graph transformer generates a nominal trajectory, a fixed projection enforces the hard constraints, and re-timing enforces the speed cap. The verifier accepts only trajectories satisfying the safety and terminal tests.

current swarm configuration, target shape, and available time budget to the velocities of all modules. Rather than relying on a dataset of precomputed solutions, the operator is trained directly through the dynamics, making the learning procedure self-supervised. This approach builds on recent self-supervised solution operators for optimal and mean-field control (Huang & Lai, 2025; Xu et al., 2025; Sikka et al., 2026a). It offers fast, model-based planning after training. The learned operator provides a high-quality motion proposal with fast inference, but its predictions are not guaranteed to satisfy the constraints because these are enforced only through soft penalties during training. We therefore project the predicted motion onto the constraint set, combining the efficiency of operator learning with a formal guarantee of safety.

Figure 1 illustrates how we combine the learned guide with a constraint-projection safety step. Our graph transformer preserves module-level states and uses contact history to coordinate local interactions while maintaining communication across separating groups. Its shared token-processing architecture naturally accommodates varying numbers of modules without relying on a mean-field approximation. Across our benchmarks, the solution-operator guide followed by safety projection achieves accurate terminal-shape matching while satisfying all hard safety constraints. In contrast, most plans produced by the baselines fail: the transport baseline, which directly solves the mean-field control problem, violates the hard constraints on most tasks, whereas the reinforcement-learning baseline remains feasible but often fails to reach the target shape.

Our contributions are summarized as follows:

- A multi-agent formulation and certification pipeline that retain pairwise exclusion, global connectivity, and speed limits throughout reconfiguration. The verifier certifies accepted plans, while allowing the learned proposal to be imperfect.
- Self-supervised training through unfolded dynamics, conditioned on a normalized time budget, yielding learned proposals that make fixed projection succeed on substantially more tasks than straight-line proposals (Table 1).
- A graph transformer whose attention follows a connected, history-dependent interaction graph, combining module-level geometry with global task information (Section 4.3).

2 Related Work

Reconfiguration and constrained control. Motion planning for many interacting bodies is computationally difficult (Spirakis & Yap, 1984; Hopcroft et al., 1984). Modular reconfiguration methods exploit motion primitives (Akitaya et al., 2021), decentralized repair rules (Sikka et al., 2026b),

or learned policies (Wu et al., 2023; Dobreva et al., 2025). Our continuous sphere model instead permits any velocity within a speed cap. Assignment-based planning (Turpin et al., 2014) motivates our travel-time bound, while density-constrained transport motivates the optimization baseline. Barrier functions and shields provide complementary mechanisms for safe control (Ames et al., 2019; Qin et al., 2021; Zhang et al., 2025; Alshiekh et al., 2018). Here, a fixed offline projection is followed by explicit trajectory certification. The projection descends from the discrete contact model of the work Maury & Venel (2011), which projects a desired crowd velocity onto the admissible non-overlap set; we project whole trajectories and add a connectivity term.

Solution operators and learned planning. DeepONet and Fourier neural operators learn mappings from problem data to solutions (Lu et al., 2021; Li et al., 2021). Direct optimization through differentiable dynamics (Hu et al., 2020; Xu et al., 2022; Onken et al., 2023) and self-supervised control operators (Huang & Lai, 2025; Xu et al., 2025; Gao et al., 2026; Sikka et al., 2026a) motivate our training objective. Learned trajectory priors have also been combined with optimization and projection (Celestini et al., 2024), from projected generative models for a single plan (Christopher et al., 2024) to their multi-robot successors (Liang et al., 2025; Idoko et al., 2025). We adapt this combination to self-supervised modular reconfiguration, including certification of global connectivity as well as collision avoidance and terminal arrival.

Graphs for swarm coordination. Connectivity-preserving control maintains links as agents move (Ji & Egerstedt, 2007; Zavlanos & Pappas, 2008; Zavlanos et al., 2011), and graph networks share control rules across agents (Tolstaya et al., 2019; Li et al., 2020). Graphormer motivates geometric attention biases (Ying et al., 2021); GraphGPS and Exphormer combine graph structure with global communication (Rampášek et al., 2022; Shirzad et al., 2023). Dynamic interaction graphs also underpin learned simulators (Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021). Our specific adaptation uses contact history, separation rates, and delayed edge release to keep the attention graph connected even when a nominal trajectory temporarily separates the physical swarm. This is an architectural prior, not a physical connectivity guarantee.

3 Problem Statement

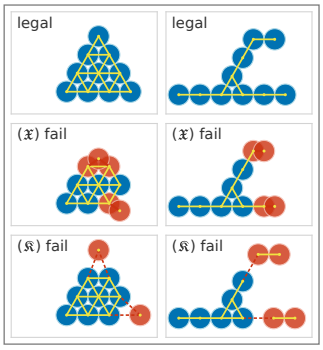

Figure 2: Examples of failures of exclusion ($\mathfrak{X}$) (middle) and connectivity ($\mathfrak{K}$) (bottom).

Reshaping a connected assembly into a new structure requires each module to find a target position, while leaving room for its neighbors and preserving a chain of contacts to every other module. Modules are interchangeable, so any module may occupy any target site. We seek the shortest motion that achieves the target shape, allowing a common translation but keeping its orientation fixed.

Let X_0 denote the starting module positions and X^* the desired target sites. At time t, $X(t) = (x_1(t), \dots, x_N(t))$ lists all centers in $\mathbb{R}^d$. Each module has radius R and diameter $D = 2R$. And $d_{ij}(X) = \|x_i - x_j\|_2$ is the distance between modules i and j. A contact edge joins centers with $D \le d_{ij} \le (1+\varepsilon)D$, where ε sets the contact-band width. We assume velocity control with maximum speed $v_{\max}$. Actuation, docking, and contact forces are outside this kinematic model. A permutation $\pi \in S_N$ assigns modules to target sites, and $\tau \in \mathbb{R}^d$ translates the whole target, so $\pi X^* + \tau = (x^*_{\pi(i)} + \tau)_{i=1}^N$. Writing the centroid as $\bar{X} = N^{-1}\sum_i x_i$, the minimum-time problem is

$$
\begin{aligned}
&\inf_{T,\, X(t)} \quad T \\
&\text{s.t.} \quad \dot{x}_i(t) = v_i(t), \qquad X(0) = X_0 \\
&\qquad X(T) = \pi X^* + \tau \quad \text{for some } \pi \in S_N,\ \tau \in \mathbb{R}^d && (\mathfrak{T}) \text{ terminal} \\
&\qquad d_{ij}(X(t)) \ge 2R \quad \forall\, i < j,\ \forall t \in [0,T] && (\mathfrak{X}) \text{ exclusion} \\
&\qquad \lambda_2\big(L_\varepsilon(X(t))\big) > 0 \quad \forall t \in [0,T] && (\mathfrak{K}) \text{ connectivity} \\
&\qquad |v_i(t)| \le v_{\max} \ \text{ a.e.} && (\mathfrak{V}) \text{ speed cap}
\end{aligned}
\tag{1}
$$

The four conditions require arrival, non-overlap, connectivity, and bounded speed. $L_\varepsilon(X)$ is the contact Laplacian; its second-smallest eigenvalue λ_2 is positive precisely when its positive-weight graph is connected (Fiedler, 1973) (boundary convention in Appendix A.1). Figure 2 illustrates the two state constraints.

4 METHOD

Problem (1) requires fast shape matching while maintaining collision avoidance and global connectivity as contacts change. Repeated trajectory optimization is costly, and varying duration changes the rollout length, complicating gradient-based training. To address these challenges, we first formulate a fixed-horizon trajectory loss, then train a shared, budget-conditioned solution operator through unfolded dynamics, using a history-dependent interaction graph to coordinate modules. Since training penalties do not guarantee feasibility, we finally project the proposed trajectory into the admissible set, and independently verify the predicted trajectories at deployment (Figure 1).

4.1 A FIXED-HORIZON RELAXATION OF ONE TASK

For a task (X_0, X^*) and fixed horizon T, we replace the terminal equality and state constraints by penalties that supply gradients when a proposal is infeasible. The speed cap remains hard through the network output. Forward Euler unfolds the dynamics as $X_{k+1} = X_k + \Delta\tau\, v_k$, so each velocity influences later states and the final shape. The loss mirrors the three constraints and adds an effort regularizer. Let $(\cdot)_+ = \max\{0, \cdot\}$, $h_{ij}(X) = d_{ij}^2(X) - D^2$, and $0 < \gamma\Delta\tau < 1$. Then

$$J_{\mathfrak{X}}[X, v] = \int_0^T \frac{1}{\binom{N}{2}} \sum_{i<j} \Big((1 - \gamma\Delta\tau)\, h_{ij}(X(t)) - \min_{s\in[0,1]} h_{ij}\big(X(t) + s\Delta\tau\, v(t)\big) \Big)_+^2 \, dt$$

penalizes violations of a discrete-time barrier condition (Ames et al., 2019). The segment minimum detects collisions between frames. From a feasible state, zero penalty implies non-overlap throughout the step, and a positive penalty discourages rapid loss of clearance. Lemma B.1 proves that $J_{\mathfrak{X}} = 0$ implies $(\mathfrak{X})$ along a continuous trajectory.

$$J_{\mathfrak{K}}[X] = \int_0^T \sum_{(ij)\in \mathrm{MST}(X(t))} \big(d_{ij}(X(t)) - 2R(1+\varepsilon) \big)_+^2 \, dt$$

penalizes stretched edges of the Euclidean minimum spanning tree (MST). For non-overlapping configurations, the band graph is connected exactly when every MST edge lies within the contact threshold, up to the boundary convention above. This provides a geometric connectivity loss without selecting a spectral-gap target. Trees are recomputed at scored steps and held fixed in backpropagation. Lemma B.2 gives the corresponding implication from $J_{\mathfrak{K}} = 0$ to $(\mathfrak{K})$, subject to the outer-boundary qualification.

$$J_{\mathfrak{T}}[X, X^*] = \min_{\pi\in S_N,\ \tau\in\mathbb{R}^d} \frac{1}{N} \sum_{i=1}^N \big| x_i(T) - x^*_{\pi(i)} - \tau \big|^2$$

measures final configuration matching. It is the squared 2-Wasserstein distance modulo translation. Centroid alignment gives the optimal τ. The Hungarian algorithm finds an optimal matching π. Holding the assignment fixed gives the exact local gradient wherever it is unique. Finally, $J_{\mathfrak{E}}[v] = \int_0^T N^{-1} \sum_i |v_i(t)|^2 \, dt$ discourages unnecessary motion. These terms define the relaxation

$$\begin{aligned} \min_{v(\cdot)} \quad & J[v] \;= w_{\mathfrak{X}} J_{\mathfrak{X}}[X, v] + w_{\mathfrak{K}} J_{\mathfrak{K}}[X] + w_{\mathfrak{T}} J_{\mathfrak{T}}[X, X^*] + w_{\mathfrak{E}} J_{\mathfrak{E}}[v] \\ & \text{s.t.} \quad \dot{x}_i(t) = v_i(t), \qquad |v_i(t)| \le v_{\max}, \qquad X(0) = X_0, \end{aligned} \tag{2}$$

with fixed positive weights. Implementation uses temporal means rather than unnormalized integrals so longer budgets do not automatically receive larger running losses. The terminal error is evaluated at X_K, and connectivity is subsampled (Appendix B.1). Finite penalty weights do not enforce hard feasibility.

4.2 TRAINING ACROSS TASKS AND TIME BUDGETS

Directly optimizing duration changes the integer number of rollout steps, making the computation graph discontinuous. We instead learn to complete tasks under prescribed budgets and leave duration selection to deployment. To compare budgets across shape pairs, we normalize by a displacement-based time scale. Every exact-arrival plan requires at least

$$\hat{T}(X_0, X^*) = \frac{1}{v_{\max}} \min_{\pi \in S_N,\, \tau \in \mathbb{R}^d} \max_i \big| x_{0,i} - x^*_{\pi(i)} - \tau \big|,$$

because each module must cover its assigned displacement within the speed cap. Minimization over translations requires branch and bound. For training, we use the cheaper centroid-aligned Hungarian assignment (π_0, τ_0) already needed by the network: $\hat{T}_0 = \max_i |x_{0,i} - x^*_{\pi_0(i)} - \tau_0| / v_{\max} \geq \hat{T}$. This is a normalization scale, not a certified lower bound; reported bounds use Appendix A.3. Each training rollout follows

$$\begin{aligned} &T = \kappa \hat{T}_0, \qquad \kappa \sim U[\kappa_{\min}, \kappa_{\max}], \qquad K = \lceil T/\Delta\tau \rceil, \\ &v_k = v_\theta\big(X_k,\ \pi_k X^* + \tau_k,\ X^* - \overline{X^*},\ 1 - k/K,\ \kappa,\ s\big), \\ &X_{k+1} = X_k + \Delta\tau\, v_k, \end{aligned}$$

for $k < K$, with T clipped as in Table 2 and rounded up to K Euler steps. Conditioning uses the effective $\kappa = T/\hat{T}_0$ after clipping. This dimensionless slack specifies the available time relative to the task scale, while $1 - k/K$ is the fraction remaining. (π_k, τ_k) is a periodically refreshed centroid-aligned Hungarian matching; s includes lattice, size, and normalized horizon features. The output head enforces $|v_i| \leq v_{\max}$. Backpropagation through the Euler steps trains the same v_θ at every step, with discrete assignments and graph updates detached. The task-averaged objective is

$$\mathcal{L}(\theta) = \mathbb{E}_{(X_0, X^*, T)} \sum_{m \in \{\mathfrak{X}, \mathfrak{K}, \mathfrak{T}, \mathfrak{E}\}} w_m\, g_m\, J_m[X_\theta, v_\theta], \quad g_m = \exp\Big(- \sum_{m' \prec m} \frac{J_{m'}}{\max\{\mathrm{ema}_{m'}, \phi_{m'}\}} \Big), \tag{3}$$

The detached gates downweight lower-priority losses while higher-priority violations remain large, reducing competition between shape matching and safety. Moving averages $\mathrm{ema}_{m'}$ and floors $\phi_{m'}$ normalize their scales; a small positive gate floor keeps every loss active. The default order is $\mathfrak{X} \succ \mathfrak{K} \succ \mathfrak{T} \succ \mathfrak{E}$. These gates are a training curriculum, not a constraint guarantee.

Randomizing and supplying κ teaches one operator to respond to different deadlines. Tighter budgets demand faster progress, while larger budgets allow detours and coordinated rearrangements. At deployment, a grid of budgets can be proposed, projected, retimed, and verified, retaining the shortest accepted plan. This is a heuristic minimum-time search; feasibility need not vary monotonically with budget. Our experiments use the single value $\kappa = 2.6$ and report durations after retiming.

4.3 TRANSFORMER ON THE ACTIVE GRAPH AS THE GUIDE

A graph transformer processes each module separately while sharing parameters across the swarm. Geometric attention biases encode interactions and task tokens communicate global shape and budget information. Its N agent tokens and N target tokens admit different swarm sizes without changing network dimensions, although transfer across sizes is not evaluated here. Agent tokens encode centered position, matched-site offset, contact degree, remaining time, and tether statistics; target tokens encode centered sites and degrees. Two global tokens carry task features and a learned register. Each agent token is decoded into its velocity (Figure 3; Appendix B.2).

Attention is dense with an additive logit term. On an allowed pair a learned bias from smooth edge features (radial basis functions of d_{ij}, a smooth contact indicator, the strain rate $\dot{d}_{ij}$), off an allowed pair a large negative constant. Disk pairs are allowed along the active graph $E_A(t)$, disk-to-site pairs along the matched site and its k nearest sites, site pairs along the goal's band graph, and the task tokens attend everywhere (Appendix B.2).

With $E_P(t) = \{ij : d_{ij}(t) \leq (1 + \varepsilon)D\}$ the band graph, E_A starts as $E_P(0)$ and is updated once per refresh period $\Delta\tau$. Every pair that came into band contact is added. An edge that has separated past a release radius, is still separating and has exhausted a dwell timer is eligible for

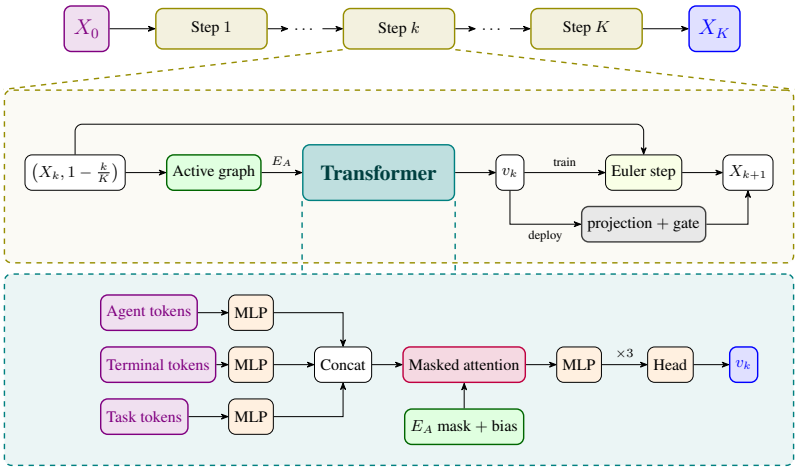

Figure 3: Our solution operator architecture. Each step updates the active graph, queries the shared transformer, and advances the positions by forward Euler. Attention uses the active graph E_A. At deployment, projection and verification follow the complete nominal rollout, as shown in Figure 1.

release. Eligible edges are dropped from the longest down, each only if the remaining graph still connects its endpoints, so $E_A(t)$ is connected for all t and $E_P(t) \subseteq E_A(t)$ (Appendix B.3). An edge that cannot be dropped is a tether, retaining communication across a gap in the nominal swarm. This history prevents the network from losing information about groups that need to reconnect. Connectivity of E_A does not imply physical contact connectivity.

4.4 Deployment: the trajectory safety projection and certification

At deployment, v_θ is evaluated once per refresh period $\Delta\tau$, held between queries, and integrated at step Δt. We project the complete nominal trajectory into the admissible set, allowing neighboring frames to adjust together. Reusing K for its number of segments, the fixed penalty program balances fidelity and smoothness against exclusion and connectivity:

$$\min_{X_1,\dots,X_K} \sum_{k,i} |x_{k,i} - x_{k,i}^{\mathrm{nom}}|^2 + \beta \sum_{k,i} |x_{k+1,i} - 2x_{k,i} + x_{k-1,i}|^2 + \mu \sum_k \Big[\sum_{i<j} \big(d_{\mathrm{ex}}^2 - d_{ij}^2(X_k)\big)_+^2 + \sum_{ij \in \mathcal{T}_k} \big(d_{ij}^2(X_k) - d_{\mathrm{keep}}^2\big)_+^2 \Big]. \tag{4}$$

The fidelity term preserves the guide's progress, while the second-difference term discourages abrupt framewise corrections. X_0 is fixed and the terminal frame is free; there is no terminal attraction or snapping. Exclusion is penalized at frames and midpoints with $d_{\mathrm{ex}} > D$. Each $\mathcal{T}_k$ is an MST over all centers, allowing projection across disconnected groups; d_{keep} lies below the verifier's safe-contact threshold (Appendix B.4). This approximate projection need not produce a feasible trajectory. To restore the speed cap, segment duration h_k is increased to $\max\{h_k, \max_i |x_{k+1,i} - x_{k,i}|/v_{\max}\}$ without changing the geometric path. The verifier then checks the retimed trajectory, including whether the terminal match survived. Only accepted plans are certified.

5 Experiments

Benchmarks. A task is an ordered pair of shapes from one procedural family, rendered as touching disks on that family's lattice, so every start and goal satisfies ($\mathfrak{X}$) and ($\mathfrak{K}$) exactly. The Vicsek benchmark is the level-3 square Vicsek fractal at $N = 64$; the hexaflake benchmark is the level-3 hexaflake on the triangular lattice with two of the six branches dropped at every node, at $N = 125$. Each supplies 180 ordered pairs of distinct held-out shapes; the two differ in lattice, density and how much of a shape is a thin arm. The space benchmark of Figure 6 samples triply periodic minimal surfaces to $N = 512$ sites of the face-centered cubic lattice, held out by surface. MNIST digits at $N = 160$ and ModelNet10 solids at $N = 1000$ are reported in Appendix E.3.

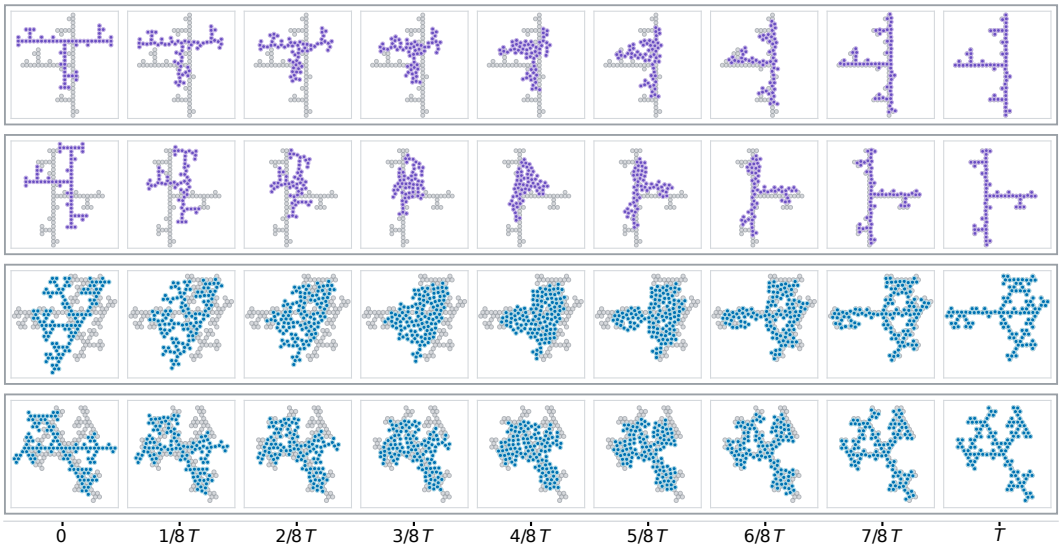

Figure 4: Two Vicsek tasks (top) and two hexaflake tasks (bottom), each generated by one learned solution operator guide with trajectory projection. Gray marks target sites; purple and blue mark modules. More examples appear in Figures 9 and 10.

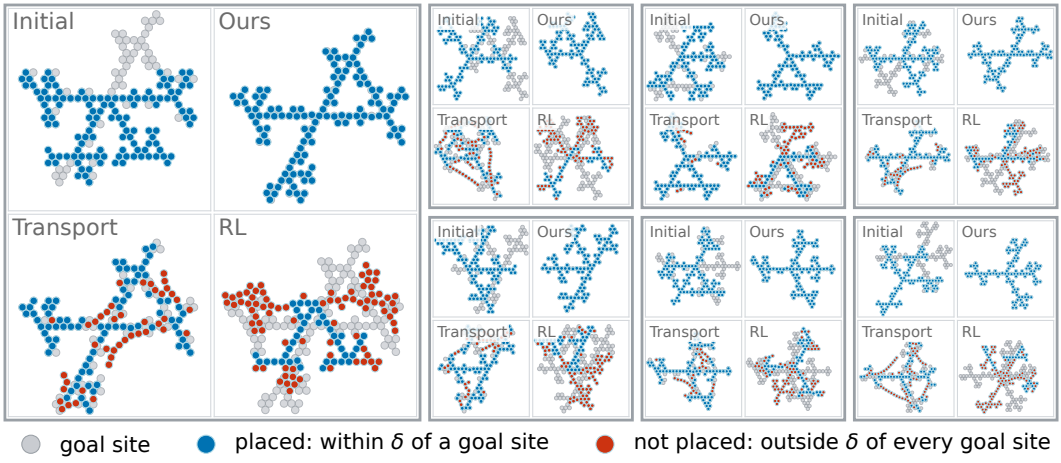

Figure 5: Hexaflake tasks at $\kappa = 2.6$; the enlarged one is where the arms differ most, the six others evenly spaced ranks of our certified makespan. Gray is a goal site; in the final frames, blue is a module within δ of one under the covering translation and red one outside. The reinforcement learning swarm is drawn shifted so its centroid is the goal's.

Baselines. We compare with density-capped optimal transport realized module by module (Appendix C.1) and multi-agent PPO (Yu et al., 2022; Nayak et al., 2023), trained on each family's training split with admissible-action masking (Appendix C.2). The latter preserves safety by construction but may fail to reach the target. All methods use the same verifier.

Protocol. All methods receive the same tasks. Ours and RL use $T = 2.6\hat{T}_0$ on the planar benchmarks; projected trajectories are then retimed. Transport searches its own horizon ladder (Appendix C.1). Certification uses $\delta = 0.35D$ for arrival and the numerical safety tolerances of Appendix A.2; no method passes the stricter arrival test at $10^{-5}D$. "Legal" means the trajectory passes the safety checks, whereas "certified" also requires arrival. Planning time excludes the shared lower-bound computation and subsequent verification. We report one training run per benchmark, without multi-seed uncertainty estimates.

Table 1: Planar benchmarks; learned and RL methods use $\kappa = 2.6$. Certified: percentage of tasks accepted. Placed: mean number of target sites covered within $\delta = 0.35D$ after a translation search, shown as mean count / N. $T/\hat{T}$: median candidate duration divided by the task's lower bound, after retiming where applicable. Wall: median planning seconds. Means and medians include rejected candidates; coverage and duration alone do not imply success. See Appendix D.2.

Method	Certified (%) ↑	Placed ↑	$T/\hat{T}$ ↓	Wall (s) ↓
Vicsek: $N = 64$, 180 tasks				
no motion	—	36.8/64	—	0
transport baseline	1.7	47.6/64	2.63	70
RL baseline	0.0	28.3/64	2.60	11
straight-line, with projection	1.7	64.0/64	5.35	75
ours, without the projection	0.0	63.9/64	2.60	2
ours, full pipeline	**73.9**	**63.9/64**	**3.03**	**31**
hexaflake: $N = 125$, 180 tasks				
no motion	—	57.9/125	—	0
transport baseline	0.0	91.5/125	2.63	127
RL baseline	0.0	52.3/125	2.60	27
straight-line, with projection	1.7	64.0/64	5.35	75
ours, without the projection	0.0	124.5/125	2.60	1
ours, full pipeline	**69.4**	**124.2/125**	**2.93**	**37**

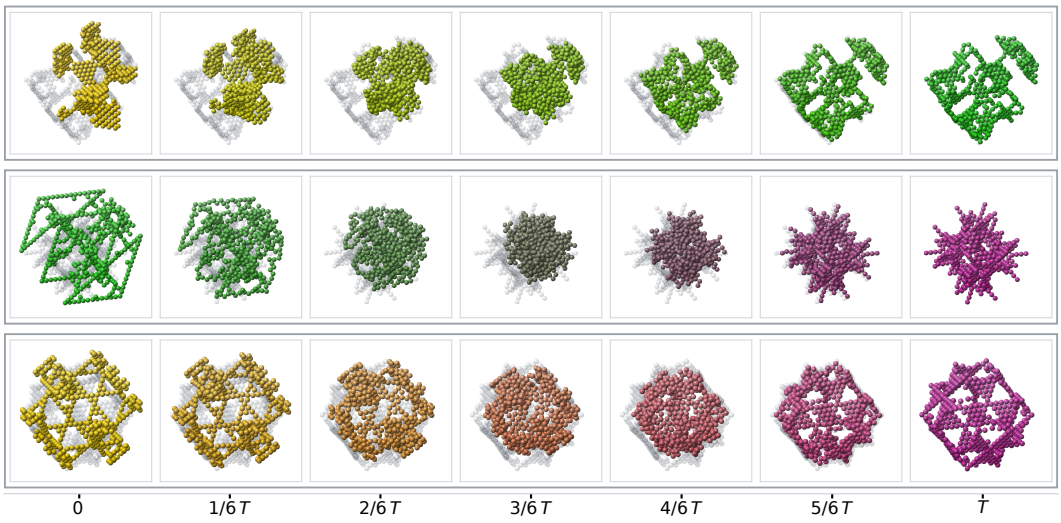

Figure 6: Three tasks of the space benchmark: triply periodic minimal surfaces at $N = 512$ spheres on the face-centered cubic lattice, six shapes over the three tasks and each task between two surfaces. More results are in Figure 11.

5.1 One operator, many tasks

Table 1 separates shape coverage from successful reconfiguration. Our method covers $63.9/64$ and $124.2/125$ target sites on average, but certifies 73.9% and 69.4% of tasks. It passes safety checks on 145 and 135 tasks, respectively. Vicsek refusals comprise 35 failed connectivity certificates and 12 terminal misses; hexaflake refusals comprise 24 overlaps, 21 failed connectivity certificates, and 10 terminal misses. Transport certifies 1.7% of Vicsek tasks and 0.0% of hexaflake tasks. RL preserves safety on every task but certifies none. Its restricted action menu limits progress (Appendix C.2); these results characterize this baseline, not all RL methods. Figure 5 compares seven hexaflake tasks on which all three methods pass safety checks: mean coverage is 125.0, 85.6, and 46.1 sites for ours, transport, and RL.

Projection increases execution time. Restoring the speed cap raises median duration from the scheduled $2.6\hat{T}_0$ to $3.03\hat{T}_0$ and $2.93\hat{T}_0$. Transport's median candidate ratio is 2.63, but almost all candidates are rejected, so this is not a comparison of successful execution times. Our median planning costs are lower than transport's, with projection accounting for most of the computation.

5.2 What the projection and the guide each contribute

Removing projection preserves near-complete coverage but yields no legal trajectory on either benchmark. Replacing the guide with straight motion toward matched sites and retaining projection certifies only 1.7% and 1.1% of tasks, predominantly failing connectivity, with larger duration ratios. Thus both the learned trajectory and its projection contribute to certification. These ablations do not isolate the benefit of the attention topology (Appendix E).

6 Conclusion

We presented a self-supervised graph-transformer solution operator that combines learned planning, fixed trajectory projection, and independent certification for modular swarm reconfiguration. Training directly through the dynamics amortizes planning across shape pairs and budgets without demonstrations, while individual-agent modeling retains exclusion and connectivity constraints. The pipeline certifies most tasks in both planar benchmarks, 57.8% of surface tasks, and 60.0% of solid-shape tasks, demonstrating its applicability in two and three dimensions with up to 1,000 modules. These results show how learned coordination and trajectory projection together support reconfiguration with explicit safety and arrival checks. Feasibility and global minimum-time optimality are not guaranteed for every input, but the results provide a foundation for further gains: improving safety projection can address residual overlaps at larger swarm sizes and reduce the budget overruns introduced by retiming. Evaluating transfer across agent counts is another promising step toward more versatile swarm planning.

The use of large language models (LLMs)

We used LLMs to do primary literature search, to write parts of the research code, and to polish text. We read every cited paper ourselves. Every count we report was measured by the code it describes. The authors are responsible for the paper.

Reproducibility statement

Appendix A.2 states the verifier as an algorithm with its tolerances and the lower bound with its termination rule; Appendix B.4 gives every constant of the projection and the retiming; Appendix D gives the benchmark construction, the metric definitions and, in Table 2, the architecture, optimizer, loss weights, gate constants, budgets, batch sizes, learning rates, seeds, step counts, selected checkpoints and hardware of every training run. The task lists are fixed-seed draws stated there. Code will be released upon acceptance.

References

Hugo A. Akitaya, Esther M. Arkin, Mirela Damian, Erik D. Demaine, Vida Dujmović, Robin Flatland, Matias Korman, Belen Palop, Irene Parada, André van Renssen, and Vera Sacristán. Universal reconfiguration of facet-connected modular robots by pivots: the O(1) musketeers. *Algorithmica*, 83(5):1316–1351, 2021.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *AAAI Conference on Artificial Intelligence*, 2018.

Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: theory and applications. In *18th European Control Conference (ECC)*, pp. 3420–3431, 2019.

Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the Monge–Kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000.

Pierre Cardaliaguet, Alpár R. Mészáros, and Filippo Santambrogio. First order mean field games with density constraints: pressure equals price. *SIAM Journal on Control and Optimization*, 54 (5):2672–2709, 2016.

René Carmona and François Delarue. *Probabilistic Theory of Mean Field Games with Applications I: Mean Field FBSDEs, Control, and Games*, volume 83 of *Probability Theory and Stochastic Modelling*. Springer, Cham, 2018. doi: 10.1007/978-3-319-58920-6. URL https://doi.org/10.1007/978-3-319-58920-6.

Davide Celestini, Daniele Gammelli, Tommaso Guffanti, Simone D'Amico, Elisa Capello, and Marco Pavone. Transformer-based model predictive control: Trajectory optimization via sequence modeling. *IEEE Robotics and Automation Letters*, 9(11):9820–9827, 2024.

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

Jacob K. Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 89307–89333, 2024.

Nadia Dobreva, Emmanuel Blazquez, Jai Grover, Dario Izzo, Yuzhen Qin, and Dominik Dold. Reconfiguration of pivoting cube ensembles under local sensing constraints using geometric deep learning, 2025.

Xiaoguang Dong and Metin Sitti. Controlling two-dimensional collective formation and cooperative behavior of magnetic microrobot swarms. *The International Journal of Robotics Research*, 39(5): 617–638, 2020.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2): 298–305, 1973.

Rebecca C. Foust, E. Sorina Lupu, Yashwanth K. Nakka, Soon-Jo Chung, and Fred Y. Hadaegh. Autonomous in-orbit satellite assembly from a modular heterogeneous swarm. *Acta Astronautica*, 169:191–205, 2020.

Suyi Gao, Mo Zhou, and Rongjie Lai. Self-supervised in-context operator learning for stochastic mean-field control. arXiv:2608.18282, 2026.

Christine E. Gregg, Damiana Catanoso, Olivia Irene B. Formoso, Irina Kostitsyna, Megan E. Ochalek, Taiwo J. Olatunde, In Won Park, Frank M. Sebastianelli, Elizabeth M. Taylor, Greenfield T. Trinh, and Kenneth C. Cheung. Ultralight, strong, and self-reprogrammable mechanical metamaterials. *Science Robotics*, 9(86):eadi2746, 2024.

Simon Hauser, Mehmet Mutlu, Pierre-Alexandre Léziart, Hala Khodr, Alexandre Bernardino, and Auke Jan Ijspeert. Roombots extended: Challenges in the next generation of self-reconfigurable modular robots and their application in adaptive and assistive furniture. *Robotics and Autonomous Systems*, 127:103467, 2020.

John E. Hopcroft, Jacob T. Schwartz, and Micha Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the "warehouseman's problem". *The International Journal of Robotics Research*, 3(4):76–88, 1984.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. DiffTaichi: Differentiable programming for physical simulation. In *International Conference on Learning Representations*, 2020.

Han Huang and Rongjie Lai. Unsupervised solution operator learning for mean-field games. *Journal of Computational Physics*, 537:114057, 2025.

Shengyi Huang and Santiago Ontañón. A closer look at invalid action masking in policy gradient algorithms. In *Proceedings of the 35th International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2022. doi: 10.32473/flairs.v35i.130584.

Simon Idoko, Prajyot Jadhav, and Arun Kumar Singh. Flow-opt: Scalable centralized multi-robot trajectory optimization with flow matching and differentiable optimization. *arXiv preprint arXiv:2510.09204*, 2025.

Meng Ji and Magnus Egerstedt. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703, 2007.

Qingbiao Li, Fernando Gama, Alejandro Ribeiro, and Amanda Prorok. Graph neural networks for decentralized multi-robot path planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Jinhao Liang, Jacob K. Christopher, Sven Koenig, and Ferdinando Fioretto. Simultaneous multi-robot motion planning with projected diffusion models. In *International Conference on Machine Learning*, volume 267 of *PMLR*, pp. 37162–37180, 2025.

Dengliang Liao, Xingyi Pan, Zhengtao Wei, and Ti Chen. Assembly and reconfiguration of space structure using heterogeneous satellite swarms. *Acta Astronautica*, 229:166–180, 2025.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Bertrand Maury and Juliette Venel. A discrete contact model for crowd motion. *ESAIM: Mathematical Modelling and Numerical Analysis*, 45(1):145–168, 2011.

Siddharth Nayak, Kenneth Choi, Wenqi Ding, Sydney Dolan, Karthik Gopalakrishnan, and Hamsa Balakrishnan. Scalable multi-agent reinforcement learning through intelligent information aggregation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25817–25833, 2023.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 278–287, 1999.

Derek Onken, Levon Nurbekyan, Xingjian Li, Samy Wu Fung, Stanley Osher, and Lars Ruthotto. A neural network approach for high-dimensional optimal control applied to multiagent path finding. *IEEE Transactions on Control Systems Technology*, 2023.

Nicolas Papadakis, Gabriel Peyré, and Edouard Oudet. Optimal transport with proximal splitting. *SIAM Journal on Imaging Sciences*, 7(1):212–238, 2014.

Kirstin H. Petersen, Nils Napp, Robert Stuart-Smith, Daniela Rus, and Mirko Kovac. A review of collective robotic construction. *Science Robotics*, 4(28):eaau8479, 2019.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

Zengyi Qin, Kaiqing Zhang, Yuxiao Chen, Jingkai Chen, and Chuchu Fan. Learning safe multi-agent control with decentralized neural barrier certificates. In *International Conference on Learning Representations*, 2021.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems*, 2022.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Jungwon Seo, Jamie Paik, and Mark Yim. Modular reconfigurable robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:63–88, 2019.

Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J. Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, 2023.

Sidhdharth D. Sikka, Suyi Gao, Zehui Lu, Rongjie Lai, and Shaoshuai Mou. Neural operator learning for collision-aware trajectory planning of spacecraft swarms. *arXiv preprint arXiv:2608.00320*, 2026a.

Sidhdharth D. Sikka, Yue Shen, and Shaoshuai Mou. Stress-sharing for decentralized fault repair in modular spacecraft. *arXiv preprint arXiv:2607.13444*, 2026b.

Paul Spirakis and Chee-Keng Yap. Strong NP-hardness of moving many discs. *Information Processing Letters*, 19(1):55–59, 1984.

Alexander Spröwitz, Rico Moeckel, Massimo Vespignani, Stéphane Bonardi, and Auke Jan Ijspeert. Roombots: A hardware perspective on 3D self-reconfiguration and locomotion with a homogeneous modular robot. *Robotics and Autonomous Systems*, 62(7):1016–1033, 2014.

Ekaterina Tolstaya, Fernando Gama, James Paulos, George J. Pappas, Vijay Kumar, and Alejandro Ribeiro. Learning decentralized controllers for robot swarms with graph neural networks. In *Conference on Robot Learning*, 2019.

Matthew Turpin, Nathan Michael, and Vijay Kumar. CAPT: Concurrent assignment and planning of trajectories for multiple robots. *The International Journal of Robotics Research*, 33(1):98–112, 2014.

Lei Wu, Bin Guo, Qiuyun Zhang, Zhuo Sun, Jieyi Zhang, and Zhiwen Yu. Learning to self-reconfigure for freeform modular robots via altruism proximal policy optimization. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 2023.

Hui Xie, Mengmeng Sun, Xinjian Fan, Zhihua Lin, Weinan Chen, Lei Wang, Lixin Dong, and Qiang He. Reconfigurable magnetic microrobot swarm: Multimode transformation, locomotion, and manipulation. *Science Robotics*, 4(28):eaav8006, 2019.

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. In *International Conference on Learning Representations*, 2022.

Wuzhe Xu, Jiequn Han, and Rongjie Lai. Self-supervised amortized neural operators for optimal control: Scaling laws and applications. *arXiv preprint arXiv:2512.24897*, 2025.

Mark Yim, Wei-Min Shen, Behnam Salemi, Daniela Rus, Mark Moll, Hod Lipson, Eric Klavins, and Gregory S. Chirikjian. Modular self-reconfigurable robot systems: challenges and opportunities for the future. *IEEE Robotics & Automation Magazine*, 14(1):43–52, 2007.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, 2021.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24611–24624, 2022.

Michael M. Zavlanos and George J. Pappas. Distributed connectivity control of mobile networks. *IEEE Transactions on Robotics*, 24(6):1416–1428, 2008.

Michael M. Zavlanos, Magnus B. Egerstedt, and George J. Pappas. Graph-theoretic connectivity control of mobile robot networks. *Proceedings of the IEEE*, 99(9):1525–1540, 2011.

Songyuan Zhang, Oswin So, Kunal Garg, and Chuchu Fan. GCBF+: A neural graph control barrier function framework for distributed safe multi-agent control. *IEEE Transactions on Robotics*, 41: 1533–1552, 2025.

A THE CERTIFIED BRACKET

A.1 THE BAND LAPLACIAN

The band Laplacian of (1) is

$$L_\varepsilon(X) = \operatorname{diag}(W\mathbf{1}) - W, \qquad W_{ij} = \psi\big(d_{ij}(X)\big) \ \ (i \neq j),$$
$$\psi(d) = \min\Big\{1,\ \Big(\frac{2R(1+\varepsilon) - d}{2R\varepsilon}\Big)_+\Big\}, \qquad \lambda_2(L) = \min_{u \perp \mathbf{1},\ \|u\|=1} u^\top L u.$$

The kernel of a nonnegative weighted Laplacian is spanned by the indicators of its connected components, so $\lambda_2(L_\varepsilon(X)) > 0$ if and only if the graph with edges $\{ij : \psi(d_{ij}) > 0\}$ is connected. That graph is the band graph of Section 3 without its outer edge. The two differ only when a pair sits exactly at $d_{ij} = 2R(1+\varepsilon)$, and every threshold the verifier applies lies strictly inside the band, so no verdict depends on the convention.

A.2 THE VERIFIER

A plan is a sequence of knot times $0 = t_0 < \cdots < t_K = T$ and configurations $X(t_0), \ldots, X(t_K)$, linear between knots. The verifier decides the four conditions of (1) in the order below, and the first failure names the refusal. Units are those of Section 3, with $D = 2R$.

1. *Binding and speed.* The plan starts at the instance's X_0 and names its X^*. Every step satisfies $|x_i(t_{k+1}) - x_i(t_k)|/(t_{k+1} - t_k) \le v_{\max} + \eta$, with $\eta = 10^{-6}D$.
2. *Sampling.* Each step is subdivided into samples at most $h = 0.025D/v_{\max}$ apart, every knot a sample, so a step lengthened by retiming is sampled more finely.
3. *Exclusion, on segments.* For a pair with relative position r at t_k and relative displacement u over the step, the minimum distance on the step is $\|r + s^*u\|$ with $s^* = \operatorname{clip}_{[0,1]}(-r^\top u/\|u\|^2)$, and $s^* = 0$ if $u = 0$. The plan is refused if any pair falls below $D - \eta$. A k-d tree prunes the pairs that cannot reach D within the step. This is exact for linear motion.
4. *Connectivity, on the safe subgraph.* At each sample t, with h_t the time to the next sample, the safe subgraph is $\{ij : d_{ij}(t) \le 2R(1+\varepsilon) - 2v_{\max}h_t\}$. Two disks change their distance at rate at most $2v_{\max}$, so a contact in it at t is still in the band until the next sample. Union-find decides whether it spans; if it does at every sample, ($\mathfrak{K}$) holds on $[0, T]$.
5. *Terminal.* The final configuration is translated so that its centroid matches the goal's, and each goal site is matched to its nearest final center. The plan is accepted if the matching is a bijection and every distance is at most δ. We use $\delta = 10^{-5}D$, at which no arm of this paper is accepted, and $\delta = 0.35D$, at which all certification rates are stated.

Connectivity is a sufficient condition, since the safe subgraph omits the outer $2v_{\max}h_t$ of the band: an acceptance proves ($\mathfrak{K}$), and a refusal means the certificate failed at some sample.

A.3 THE LOWER BOUND

The bound $\hat{T}$ of Section 4.2 minimizes the bottleneck displacement over both π and τ. The minimum over τ is what makes it a bound: the verifier accepts the goal at any translation, so a bottleneck at one fixed translation can exceed the optimum.

In the plane the minimum over τ is computed by branch and bound from the centroid difference $\bar{\tau}$. The objective is 1-Lipschitz in τ, and the identity $\sum_i |x_{0,i} - x^*_{\pi(i)} - \tau|^2 = \sum_i |x_{0,i} - x^*_{\pi(i)} - \bar{\tau}|^2 + N|\tau - \bar{\tau}|^2$, valid for every π, certifies the search box. Each box carries a lower envelope of the bottleneck. The search stops at an absolute gap of 10^{-3} diameters or after two seconds, and a truncated search returns the envelope, so the bound only loosens. Every fractal task closed to the target gap; a few MNIST tasks stopped at the cap with gaps up to 0.29 diameters.

At $N = 1000$ this search is too slow for a sweep. There $\hat{T}$ is the root-mean-square bound $\big(\min_\pi \frac{1}{N}\sum_i |x_{0,i} - x^*_{\pi(i)} - \bar{\tau}|^2\big)^{1/2}/v_{\max}$, exact in τ at the centroid difference and weaker than

the bottleneck by about a factor of two, while the budget there is κ times the bottleneck at the centroid alignment, which is not itself a bound. A space plan therefore reads as a larger multiple of $\hat{T}$ than a plane plan at the same slack. Every ratio we report is against the certified bound of its own benchmark.

The tables divide candidate duration by $\hat{T}$, including rejected plans. For an accepted plan, the bound for arrival within δ is $(\hat{T} - \delta/v_{\max})_+$ (Section 3). When positive, the ratio to this bound is $(T/\hat{T})/(1 - \delta/(v_{\max}\hat{T}))$. Acceptance at tolerance δ bounds $T^\star_\delta$, not the exact-arrival optimum $T^\star$.

B METHOD DETAILS

B.1 THE LOSS ON THE ROLLOUT

Exclusion. The barrier condition $\dot{h}_{ij} \geq -\gamma h_{ij}$ with $h_{ij} = d_{ij}^2 - 4R^2$, integrated over one step of length $\Delta\tau$, reads $h_{ij}(X + \Delta\tau v) \geq (1 - \gamma\Delta\tau)h_{ij}(X)$. $J_{\mathfrak{X}}$ replaces the end-point value by the minimum of h_{ij} along the straight segment from X to $X + \Delta\tau v$, the quantity the verifier tests, so a pair that overlaps in the middle of a step and separates by its end is still penalized. The sum is divided by $\binom{N}{2}$, and $\gamma\Delta\tau < 1$ keeps the factor positive.

Connectivity. The longest edge of the Euclidean minimum spanning tree $\mathrm{MST}(X)$ is the smallest threshold at which the distance graph on the centers is connected, so the band graph is connected exactly when every tree edge is at most $2R(1+\varepsilon)$ long, up to the boundary convention of Appendix A.1. The precise zero-loss implication is stated in Lemma B.2. The gradient of $J_{\mathfrak{K}}$ acts on the two disks of each stretched edge.

The following lemmas concern the continuous-time losses of Section 4.1. Assume $T > 0$, $N \geq 2$, and $X : [0, T] \to (\mathbb{R}^d)^N$ is continuous.

Lemma B.1 (Zero exclusion loss)**.** *If $0 < \gamma\Delta\tau < 1$ and $J_{\mathfrak{X}}[X, v] = 0$, then $d_{ij}(X(t)) \geq D$ for every $i < j$ and $t \in [0, T]$; hence ($\mathfrak{X}$) holds.*

Proof. Since $s = 0$ is included in the segment minimum, each summand in $J_{\mathfrak{X}}$ is at least $\big(\gamma\Delta\tau\,(-h_{ij}(X(t)))_+\big)^2$. Zero loss therefore gives $h_{ij}(X(t)) \geq 0$ almost everywhere. Continuity extends this inequality to every $t \in [0, T]$. □

Lemma B.2 (Zero connectivity loss)**.** *Suppose ($\mathfrak{X}$) holds and write $r_c = D(1+\varepsilon)$. If $J_{\mathfrak{K}}[X] = 0$, then the graph with edges $D \leq d_{ij}(X(t)) \leq r_c$ is connected for every $t \in [0, T]$. In particular, ($\mathfrak{K}$) holds if also $d_{ij}(X(t)) \neq r_c$ for every $i < j$ and $t \in [0, T]$.*

Proof. If the stated graph were disconnected at some t_0, every edge across a cut separating its components would have length greater than r_c. By continuity, these lengths exceed r_c by a fixed positive amount on an interval of positive length. Every spanning tree crosses this cut, so the MST penalty has positive integral, a contradiction. Under the additional assumption, the connecting edges have length strictly below r_c and thus positive Laplacian weights, giving $\lambda_2(L_\varepsilon) > 0$. □

The boundary qualification is necessary for Appendix A.1's kernel: two stationary modules at distance r_c have $J_{\mathfrak{K}} = 0$ but $\lambda_2(L_\varepsilon) = 0$.

On the rollout. Each integral is a mean over the K steps. The connectivity term is evaluated every second step in the plane and every eighth in space (Table 2), and the energy term is the mean squared speed. Rollouts in a batch are padded to the longest K and masked. The loss is a nonconvex, piecewise smooth function of the KNd velocities of one task: the assignment, the tree and the active set of each positive part are discrete, and the loss is differentiable wherever they are locally constant, which is almost everywhere.

B.2 TOKENS AND ATTENTION

Agent token i is an MLP of $(x_i - \bar{X},\ y_i - x_i,\ \deg_i,\ (T-t)/T,\ \ell_i,\ s_i)$, with y_i the disk's matched site and $\deg_i = \frac{1}{z}\sum_j \sigma_{ij}$ its smooth band degree, where $\sigma_{ij} = \sigma\big((2R(1+\varepsilon) - d_{ij})/w\big)$ for a sigmoid of width w and z the kissing number in dimension d (6 in the plane, 12 in space). The two tether statistics are computed over the active edges $ij \in E_A$ (Appendix B.3),

$$\ell_i = \frac{1}{6}\sum_{j:\, ij \in E_A} (1 - \sigma_{ij}), \qquad s_i = \tanh \frac{\sum_{j:\, ij \in E_A}(1-\sigma_{ij})\,\big(d_{ij} - 2R(1+\varepsilon)\big)_+}{\sum_{j:\, ij \in E_A}(1-\sigma_{ij})},$$

the load, a smooth count of active edges beyond contact, and the stretch, their weighted mean excess length. Terminal token j is an MLP of $\big(x_j^* - \overline{X^*},\ \deg_j^*\big)$, the centered goal site and its smooth degree in the goal's band graph, fixed for the task. One task token is an MLP of the task scalars (horizon, lower bound, lattice) and the other a learned register.

The attention logit is $a_{ij} = q_i \cdot k_j/\sqrt{d} + b_{ij}$ with

$$b_{ij} = \begin{cases} \mathrm{MLP}\big(\mathrm{RBF}(d_{ij}),\ \sigma_{ij},\ \tanh(\dot{d}_{ij}/2v_{\max})\big) & ij \in E_A, \\ -M & \text{otherwise}, \end{cases}$$

M large and finite. Disk-to-site pairs on the cross support, the matched site and its k nearest sites, get the same form from $\mathrm{RBF}(|x_i - x_j^*|)$ and a matched-site indicator; site-to-site pairs are allowed along the goal's band graph; the rows and columns of the task tokens are zero, so they attend everywhere. Every feature is a smooth function of X, and the supports are the only discrete objects.

B.3 THE ACTIVE GRAPH UPDATE

Let $E_P(t) = \{ij :\ d_{ij}(t) \le 2R(1+\varepsilon)\}$ be the band graph. E_A starts as $E_P(0)$ and is updated once per refresh period $\Delta\tau$ from the realized positions in three steps:

$$\begin{aligned} &\text{connect: } ij \notin E_A \text{ and } \min_{[t-\Delta\tau,\, t]} d_{ij} \le 2R(1+\varepsilon) \ \Rightarrow\ ij \in E_A; \\ &\text{eligible: } ij \in E_A, \quad d_{ij}(t) > \tfrac{5}{2}R, \quad \frac{d_{ij}(t) - d_{ij}(t-\Delta\tau)}{\Delta\tau} \text{ above a small threshold,} \\ &\qquad\qquad \text{and the edge's dwell timer has expired;} \\ &\text{release: for eligible } ij \text{ in decreasing } d_{ij}\text{:} \quad \text{drop } ij \iff E_A \setminus \{ij\} \text{ connects } i, j. \end{aligned}$$

The connect step uses the minimum distance over the period, so a contact that opened and closed between two updates is still recorded. The dwell timer is a per-edge counter that refills while the pair is in contact and halves at each update while the edge is eligible, so an edge has to keep separating over several updates before it can go. The release pass is one union-find sweep from the longest eligible edge down. Two invariants follow: $E_A(t)$ is connected for all t, and $E_P(t) \subseteq E_A(t)$ after every update. The constants are listed in Appendix D.5.

An eligible edge that cannot be dropped is a tether. It is the only path in E_A between the two sides of a gap the swarm has opened, so it is the attention edge through which the separated parts still see each other. The edge set carries no gradient.

B.4 THE PROJECTION

The raw rollout is forward Euler on the guide's field at the control step $\Delta t = 2h$, with velocities above $v_{\max}$ rescaled to it and nothing else enforced. Its frames $X_0^{\mathrm{nom}}, \dots, X_K^{\mathrm{nom}}$ are the data of (4); X_0 is pinned and $X_1, \dots, X_K$ are the variables. Each sum in (4) is divided by KN.

Terms. Fidelity and smoothness are quadratic. The two penalties are squared hinges of squared distances, quartic in X, so the program is not quadratic and is solved by penalty continuation with Adam. Smoothness is the discrete acceleration with $\beta = 100$; at $\beta = 1$ the worst step of the projected path runs at close to six times the speed cap. Exclusion uses $d_{\mathrm{ex}} = 2.012R$ at every frame and at every midpoint $\frac{1}{2}(X_k + X_{k+1})$. The midpoint rows are needed: the start configuration is tangent, so pinning X_0 while X_1 moves sends the chord below $2R$ on step zero. Connectivity uses

$d_{\text{keep}} = 2.03R$, under the verifier's threshold $d_s = 2R(1+\varepsilon) - 2v_{\max}h = 2.1R$, on the Euclidean minimum spanning tree $\mathcal{T}_k$ of the current X_k, rebuilt by Prim's algorithm at each outer iteration. The norm is convex, so an edge inside d_{keep} at both ends of a segment stays inside it throughout, and connectivity needs no midpoint rows.

Solve. The penalty weight starts at $\mu = 1$ and is multiplied by 10 at each of seven outer iterations, ending at 10^6; each runs 250 Adam steps at learning rate 0.01 on all free frames at once. Projecting one frame at a time satisfies exclusion exactly and still fails the gate on speed and on between-frame connectivity. Distances are in float64, since the single-precision error exceeds the exclusion tolerance. A step evaluates all pairs at every frame and midpoint, $O(KN^2)$; Appendix D.4 gives the neighbour list used at $N = 1000$.

What the verifier then finds. Retiming gives step k the duration $\max\{\Delta t,\ \max_i \|x_{k+1,i} - x_{k,i}\|/v_{\max}\}$, so the speed cap holds and no distance changes. Exclusion is a penalty; the ratchet drives the residual overlap to about 10^{-7}, and the verifier measures it against its tolerance. The terminal frame is free, so the terminal test is the same as for any other plan. Connectivity is imposed at frames only, and the tree that spans X_k need not span X_{k+1}, so a sample between two connected frames can carry no spanning safe subgraph. That is the main refusal on the Vicsek family.

Connectivity between frames. One term closes that gap. Adding

$$\sum_{k=0}^{K-1} \sum_{ij \in \mathcal{T}_k} \left(\|x_{k+1,i} - x_{k+1,j}\|^2 - d_{\text{keep}}^2 \right)_+^2$$

to (4) asks the tree of X_k to be short at X_{k+1} as well. By convexity of the norm every edge of $\mathcal{T}_k$ then stays inside d_{keep} along the whole segment. Retiming only lengthens a step and the verifier subdivides any step longer than h, so its threshold never falls below d_s, and the safe subgraph at every sample contains a spanning tree. The term costs $N - 1$ rows per step against the $O(N^2)$ of exclusion, and a stiffer program that takes longer to solve. The rows are penalties, so a residual survives; the margin $d_s - d_{\text{keep}} = 0.07R$ absorbs it and the verifier measures the result. Every number in Section 5 and Appendix E.3 was produced with these rows switched off; the current code switches them on by default.

C Baselines

C.1 The transport baseline

The transport baseline plans with a density in place of N modules. The swarm is a mass density $\rho(t,\cdot) \geq 0$ of total mass N on a box $\Omega \subset \mathbb{R}^d$ with momentum $m = \rho v$. Exclusion ($\mathfrak{X}$) becomes the cap $\rho \leq \rho_{\max}$, the density of the lattice packing, one cell per module; the speed cap ($\mathfrak{V}$) becomes the cone $|m| \leq v_{\max}\rho$. Connectivity ($\mathfrak{K}$) has no expression on densities, since connected support is not preserved under weak limits, so it is dropped, and module identities with it. At a fixed horizon T the plan is the minimizer of

$$\min_{\rho,\, m} \int_0^T \int_\Omega \frac{|m|^2}{2\rho}\, dx\, dt \quad \text{s.t.} \quad \partial_t \rho + \nabla \cdot m = 0, \qquad m \cdot \nu = 0 \text{ on } \partial\Omega,$$
$$\rho(0) = \rho_0, \quad \rho(T) = \rho^*, \qquad 0 \leq \rho \leq \rho_{\max}, \qquad |m| \leq v_{\max}\,\rho, \tag{5}$$

the Benamou–Brenier formulation of optimal transport under a density cap (Benamou & Brenier, 2000; Cardaliaguet et al., 2016). The kinetic term is jointly convex in (ρ, m), the cap is linear and the cone is convex, so with ($\mathfrak{K}$) absent the problem at fixed T is convex and feasibility is monotone in T. The endpoint densities are the unions of the lattice cells at the occupied sites, so the cap is tight at both ends; the goal is first shifted onto the start's centroid and the two masses are equalized.

Solve. (5) is discretized on a staggered space-time grid, ρ at time levels and cell centres and m on cell faces at interval midpoints, and solved by a first-order primal-dual iteration (Chambolle & Pock, 2011; Papadakis et al., 2014). The proximal step of the kinetic energy with the cap and

the cone is pointwise and closed form. The continuity constraint is dualized, and its dual step is preconditioned by the exact inverse of the discrete space-time Neumann Laplacian, one cosine transform per iteration. The horizon is not optimized: the solver runs a ladder of three horizons $\kappa\hat{T}$, $\kappa \in \{1.25, 1.75, 2.5\}$, and accepts the first rung whose continuity residual is small and whose density $\{\rho > 0.1\,\rho_{\max}\}$ stays in one connected component at every time level; if none passes, the smallest residual is kept.

Realization. A density path induces no module trajectories. Modules follow the velocity $v = m/\rho$, interpolated in space and time, at the control period, and over the last leg the field is blended into a straight drive to Hungarian-matched sites: a mean-field plan names no module-to-site assignment, and without one the terminal equality is unreachable. The rollout runs half again past the horizon for that leg, which is why this arm's makespans sit at 1.5 times a rung of the ladder. Every control step passes through an online velocity filter, a small quadratic program that keeps each close pair from closing below $2R$ and each spanning contact inside the safe band over the step, and a step that still fails the verifier's segment and safe-subgraph checks is backtracked to a hold. Holding is always legal, so every refusal of this arm is terminal. A final snap onto the sites, under the same step check, closes the last half diameter. Each task has five minutes of wall clock; spare budget realizes the remaining rungs and keeps the best plan. On the plane families every task was re-planned here; in space the trajectories are read from the campaign that produced them. The result goes through the same verifier as every other arm.

C.2 The reinforcement learning baseline

The baseline's action set is admissible by construction: every trajectory it can emit satisfies ($\mathfrak{X}$), ($\mathfrak{K}$) and ($\mathfrak{V}$) exactly from the first random rollout on, and learning is spent on arriving. This is the masked-action route to hard constraints (Dobreva et al., 2025), with the mask derived from (1).

Decision process. An episode is one task at a budget $T = \kappa\hat{T}_0$ drawn as in Section 4.2, split into K control periods $\Delta\tau = T/K$. The goal is centroid-aligned to the start and matched once by the Hungarian algorithm, which gives module i a site y_i; fixing (π, τ) restricts ($\mathfrak{T}$), so the arm pursues a harder terminal condition than the benchmark scores. A control step opens with a drift g_k, one displacement shared by the whole swarm and chosen by a swarm-level head. Each module then picks a deviation m_i from a finite menu $\mathcal{A}$ of M directions and Q geometric magnitudes, and its displacement is $g_k + m_i$ clipped into the ball of radius $v_{\max}\Delta\tau$, which is ($\mathfrak{V}$). The drift is what lets a swarm at exact contact move at all: against neighbours at rest, even a rigid translation is refused.

Admission. Within a step the modules commit one at a time in breadth-first order on a spanning tree $\mathcal{T}_k$ of the current safe contact graph, with parent $p(i)$ and children $\mathrm{Ch}(i)$. A module's assumed motion $\tilde{\Delta}_j$ is its realized displacement once it has committed and g_k before. With $\underline{d}_{ij}$ the minimum distance of two linearly moving points over the step, a deviation realizing Δ_i is offered to module i when

$$\underline{d}_{ij}(\Delta_i, \tilde{\Delta}_j) \geq 2R \;\; \forall j \neq i, \qquad \left|x_i + \Delta_i - x_{p(i)} - \Delta_{p(i)}\right| \leq d_s,$$
$$\left|x_i + \Delta_i - x_c - \tilde{\Delta}_c\right| \leq d_s \;\; \forall c \in \mathrm{Ch}(i),$$

with $d_s = 2R(1+\varepsilon) - 2v_{\max}h$ the verifier's safe-subgraph threshold. The zero deviation always passes, so the offered set is never empty, and by induction from a legal X_0 every tree edge stays inside d_s and every pair clears $2R$ on the whole segment. The tree rule, the finite menu and the single drift cost optimality, not legality.

Policy and reward. Module i observes its nearest peers and nearest sites inside a sensing radius, encoded by shared message passing (Nayak et al., 2023), together with its time fraction, offset to y_i, degree, clearance and last offered fraction of the menu. A module head emits logits z_i over $\mathcal{A}$ and a swarm head on the pooled embedding emits the drift's. Module actions are drawn from the masked categorical $\pi_\theta(\Delta \mid o_i) \propto e^{z_{i\Delta}}\,\mathbf{1}[\Delta \in \mathcal{F}_i]$ over the offered set $\mathcal{F}_i$, which keeps the policy gradient unbiased when the mask is part of the state (Huang & Ontañón, 2022). With N_k the modules within δ of their site, module i receives

$$r_{i,k} = \tfrac{w_M}{N}\,(N_k - N_{k-1}) - \tfrac{w_T}{K} + w_F\,\mathbf{1}[N_k = N] + w_S\big(|y_i - x_{i,k-1}| - \gamma\,|y_i - x_{i,k}|\big),$$

progress on ($\mathfrak{T}$), a time cost, an arrival bonus and potential-based shaping, which cannot move the optimum (Ng et al., 1999). No term refers to a constraint, because none can be violated.

Training and deployment. Multi-agent PPO (Schulman et al., 2017; Yu et al., 2022) with one shared network, a centralized critic, generalized advantage estimation, the clipped surrogate on both heads (the drift against the swarm-mean advantage) and an entropy bonus; one run of ten and a half hours per family, tens of millions of environment steps. Checkpoints are selected on held-out training shapes by arrival, then station count, then finish time. Deployment is greedy at the study budget, through the same verifier as every other arm. Settings: $M = 16$, $Q = 6$, $K = 128$, two message-passing layers of width 256, $w_M = 10$, $w_S = w_T = 1$, $w_F = 5$. On the benchmark the arm is offered a third (Vicsek) or a tenth (hexaflake) of its menu per decision and declines to deviate from the drift on 16% and 73% of decisions: the swarm stays legal and jams.

D Experimental setup

D.1 Benchmarks

A Vicsek or hexaflake shape is grown by the usual recursion with a fixed number of branches omitted at every node, under one seed, and its cells are embedded at contact distance on the family's lattice. Seeds are filtered to be distinct up to translation, and the task list is a fixed-seed shuffle of the ordered pairs of the held-out split. The surfaces of Figure 6 are level sets of the standard trigonometric approximants of six triply periodic minimal surfaces, sampled to exactly $N = 512$ face-centered cubic sites in a box scaled to that budget; the split is by surface, so all thirty-two test shapes are Schwarz P and no training task is drawn from it. A digit shape is the N brightest connected pixels of an MNIST image on the square lattice, or the N brightest connected axial sites on the hexagonal one; the same image index is used for every task, so a task is identified by its two digits and its lattice. A ModelNet10 shape is the largest connected component of a voxelization of a test-split mesh onto the face-centered cubic lattice at exactly N sites, with the ordered class pairs drawn once under a fixed seed.

D.2 Metrics

Placed: partial shape coverage. For final centers $F = (f_i)$ and target sites $G = (g_j)$, the reported count is

$$P(F, G) = \max_{\tau \in \mathcal{C}} \sum_{j=1}^{N} \mathbf{1}\Big[\min_i \|f_i + \tau - g_j\|_2 \le \delta\Big], \qquad \delta = 0.35D.$$

The candidate set $\mathcal{C}$ is obtained by binning the differences $g_j - f_i$ on a δ-grid and averaging the differences in each of the eight most-populated bins. Thus the translation search is heuristic and can undercount the maximum over all translations. Because target sites are at least D apart and $\delta < D/2$, one module cannot cover two sites. Tables 1 and 3 report the arithmetic mean of P across all tasks, including rejected trajectories, followed by the total number of sites N. For example, $63.9/64$ means an average of 63.9 covered sites out of 64. The no-motion row applies the same metric to X_0, measuring initial shape overlap.

Certified: complete verified reconfiguration. This is 100 times the number of tasks passing every verifier test divided by the total number of tasks, reported as a percentage rounded to one decimal place. It requires segmentwise collision avoidance, the continuous-time connectivity certificate, the speed cap, and terminal arrival. The terminal test aligns centroids and requires a bijection within δ; it does not use the covering translation above. Consequently, full coverage need not imply certification. A legal trajectory passes the safety tests but may miss the target. All certification rates use $\delta = 0.35D$; none of the reported methods is accepted at the stricter $10^{-5}D$ arrival tolerance.

Duration and computational cost. $T/\hat{T}$ is the median of per-task candidate durations divided by their lower bounds, using retimed durations where applicable and including rejected candidates. It is not an optimality guarantee for a rejected plan; the tolerance-adjusted bound for an accepted plan is $(\hat{T} - \delta/v_{\max})_+$. Wall time is the median planning time over all tasks, including rollout, projection,

and retiming for our method, but excluding the shared lower-bound computation and subsequent verification (Appendix D.4). Dashes denote unreported certification or duration for the no-motion reference.

D.3 Training runs and checkpoint selection

Each benchmark has its own training run of the architecture of Section 4.3, stopped on a held-out placement count (Table 2). The Vicsek, MNIST and surface runs are the method of Section 4 as stated: the barrier form of ($\mathfrak{X}$), the spanning-tree form of ($\mathfrak{K}$), the plain terminal cost, and the slack among the task inputs. The hexaflake and ModelNet10 runs differ in one setting, the priority order of (3): their terminal term leads and is ungated, with weight 3 in place of 1, because the projection can enforce exclusion and connectivity after the rollout but cannot manufacture a terminal match the field never reached.

Scoring a sweep of checkpoints through the projection and the gate costs about twenty seconds a task, so selection has two stages. The first eliminates on the raw field's assignment count over a held-out set disjoint from the benchmark, which is cheap and puts every collapsed checkpoint far below the pack; the second re-scores the survivors through the projection and the verifier and decides. The first stage cannot rank: on one run the top twenty checkpoints by raw count span less than one module of 125 while their acceptance through the projection runs from none to all. What separates checkpoints is whether the field is projectable, which the raw field cannot show. This procedure selected the hexaflake checkpoint. The other four were selected on a placement count through an earlier per-step velocity filter, a quadratic program like the one in the transport baseline's realizer (Appendix C.1), and deployed through the projection without reselection: three before the projection existed, on held-out tasks, and the ModelNet10 one on six fixed tasks of its own benchmark, which is why the thirty ModelNet10 tasks we report share no shape with those six.

D.4 Deployment in space and compute

The dilation in space. The three-dimensional rollout prepends a uniform dilation of the start about its centroid by a factor a few percent above one. Every pair distance grows by that factor, so exclusion holds and every contact at $2R$ stays inside the band: the phase is legal by construction, and it opens the close packing enough for the projection to have room. It is applied to every arm on that benchmark.

The projection at a thousand modules. The dense exclusion sum wants about twenty-eight gigabytes and over a second per Adam step at $N = 1000$. It is evaluated instead on a neighbour list rebuilt at every outer iteration, with a cutoff well outside the barrier's support: about six thousand of the $499\,500$ pairs per frame carry the whole term, and restricting the sum to them moves the projected path by 2×10^{-11}. The spanning trees and the diagnostics are computed a slice of frames at a time. Neither changes a term of the objective, and neither is switched on below this size.

Cost. Our wall-clock covers the guide's forward passes, one per control period on a CPU worker, the projection's penalty solve on one GPU, and the retiming. It excludes the lower bound, computed once per task and shared by every arm, and the verifier, which runs afterwards. The projection is nearly all of it: without it the arm is about ten times faster and inadmissible. The transport baseline's time covers its solve, its realization and its online filter; the reinforcement learning baseline's covers its rollout with the admission mask. At a thousand modules the network's forward pass grows with the swarm and the balance shifts. The ratio between arms is the part to trust: the plane numbers were measured under one process pool, while in space our model's and the baseline's trajectories come from the campaigns that produced them, measured on the same machine at different times and under different loads.

D.5 Settings

Table 2 lists the training run behind every reported number. Units are the disk diameter $D = 2R$ and $v_{\max} = 1$, so a time is a distance in diameters. The training budget $T = \kappa \hat{T}_0$ is clipped to the range $[T_{\min}, T_{\max}]$ of the table, so the largest slacks are seen only on tasks with short bounds. Shared by every run:

Table 2: The five training runs. "checkpoint" is the step whose weights every reported number of that benchmark uses; "selected on" is the quantity that chose it, on tasks held out from those reported, and the deployment path it was scored through.

	Vicsek	hexaflake	MNIST	surfaces	ModelNet10
d, N	2, 64	2, 125	2, 160	3, 512	3, 1000
shapes, train / test	448 / 64	448 / 64	images 2–4999 / 0, 1	224 / 32	train / test split
tasks	180	180	180	90	30
priority order	$\mathfrak{X}\mathfrak{K}\mathfrak{T}\mathfrak{E}$	$\mathfrak{T}\mathfrak{X}\mathfrak{K}\mathfrak{E}$	$\mathfrak{X}\mathfrak{K}\mathfrak{T}\mathfrak{E}$	$\mathfrak{X}\mathfrak{K}\mathfrak{T}\mathfrak{E}$	$\mathfrak{T}\mathfrak{X}\mathfrak{K}\mathfrak{E}$
budget clip $[T_{\min}, T_{\max}]$	$[4, 24]$	$[4, 24]$	$[4, 16]$	$[6, 18]$	$[6, 16]$
rematch / $\mathfrak{K}$ cadence	4 / 2	4 / 2	4 / 2	16 / 8	16 / 8
batch (tasks)	36	18	12	9	6
learning rate	10^{-4}, 3×10^{-5}	5×10^{-5}	5×10^{-5}	10^{-4}	10^{-4}
seed	1	1	2	0	0
launches, hours	2, 11.7	1, 5.5	3, 11.3	1, 11.4	1, 11.1
checkpoint (step)	3 359	5 550	10 865	4 609	2 507
selected on	placed, filter	gate, projection	placed, filter	placed, filter	placed, filter

- Network: token width 192, four heads, three blocks, feed-forward width 384, 16 radial basis bins out to $3D$, the matched site and its 3 nearest sites on the cross support, two task tokens; 1.01 million parameters.
- Optimizer: AdamW with weight decay 10^{-3} and gradient clipping at 1; slack $\kappa \sim U[1.5, 3.5]$.
- Time steps: refresh period $\Delta\tau = 0.25$, control step $\Delta t = 0.05$, verifier step $h = 0.025$.
- Loss: barrier gain $\gamma = 3.05$; weights $w_{\mathfrak{X}}, w_{\mathfrak{K}}, w_{\mathfrak{T}}, w_{\mathfrak{E}} = 1000, 3, 1, 0.01$ ($w_{\mathfrak{T}} = 3$ for hexaflake and ModelNet10); gate moving averages at rate 0.99 with floors $\phi = 3 \times 10^{-6}, 10^{-6}, 10^{-2}, 0.05$ for $\mathfrak{X}, \mathfrak{K}, \mathfrak{T}, \mathfrak{E}$, the exponent capped at 0.7 and the gate floored at 0.01.
- Active graph: connect at 1.1, eligible beyond 1.25 at a strain rate above $0.01\, v_{\max}$; the dwell counter starts at 0.3, refills by 0.35 per update in contact and allows release below 0.05.
- Verifier: $\varepsilon = 0.1$, $\eta = 10^{-6}$, $\delta \in \{10^{-5}, 0.35\}$; the projection constants of Appendix B.4.

Training ran on one NVIDIA RTX 4090. The benchmarks ran the forward passes on CPU workers and the projection on one GPU, an RTX 4090 or a Quadro RTX 5000. A run is one optimizer trajectory, in two cases restarted from an earlier checkpoint of itself at a lower learning rate, as the table notes.

E Additional results

The attention topology of Section 4.3 is not ablated: every model in this paper uses the active graph, so no comparison isolates it. Replacing it by full attention, by the instantaneous contact graph or by a spanning tree of the current positions would show what the memory in E_A contributes, and each such arm is a retraining, so we leave it out rather than report it from a shorter run. The same holds for the priority gates of (3) and for the number of blocks.

E.1 The three arms side by side

Figures 5, 7 and 8 put the three arms on the same tasks, twelve Vicsek and nineteen hexaflake, and show the terminal condition only, since every trajectory in them is legal: ours by the projection and the gate, the baselines by the construction of their realizers (Appendices C.1 and C.2). A tile is one task: the start, then the final frame of each arm over the goal, drawn under the translation that covers the most sites for that frame. That is the frame the placed column of Table 1 is counted in, so a red module is one that column leaves out. The reinforcement learning swarm ends displaced as a whole, so its final frame is drawn translated to the goal's centroid, keeping the labels of the covering frame; a blue module there need not sit on a drawn site. The enlarged tile of each figure

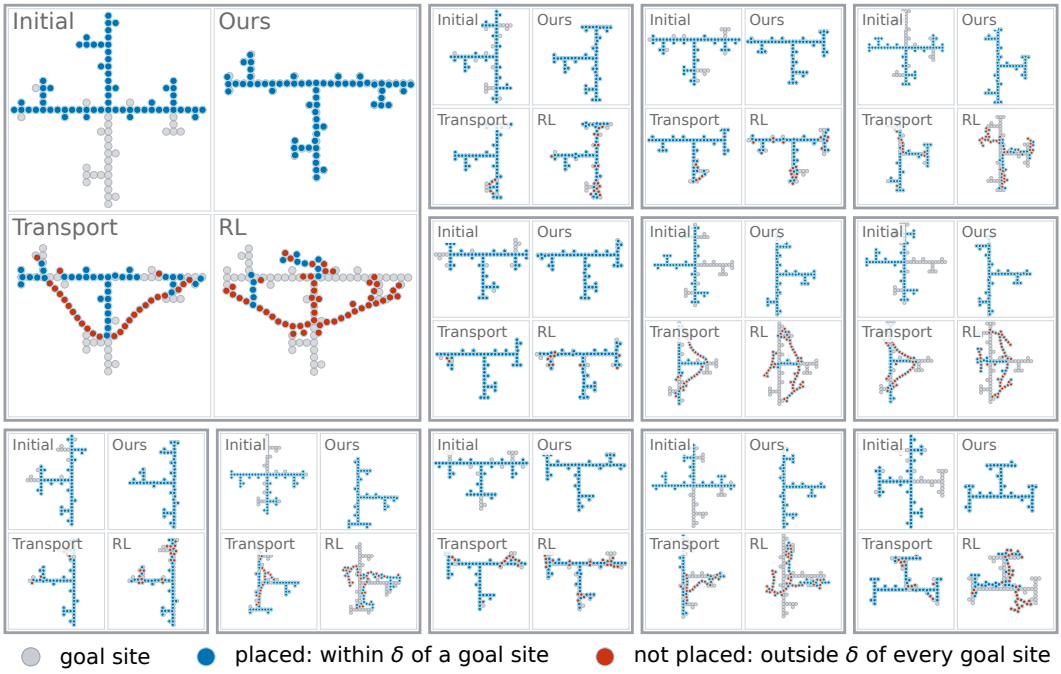

Figure 7: Vicsek tasks, $N = 64$, in the conventions of Figure 5 with a third row: in each tile the start, then the final frames of our model, the transport baseline and the reinforcement learning baseline.

is the certified task on which the arms separate most; the small tiles are evenly spaced ranks of the verified makespan over the tasks the gate certifies for our model, with distinct shapes where the split allows. The bottom row of Figure 7 adds five such ranks among tasks that share no shape with the seven above it, and Figure 8 draws all twelve of its tasks from those that share no shape with Figure 5. Over the twelve Vicsek tasks our model places 64.0 of 64 sites on average, the transport baseline 47.3 and the reinforcement learning baseline 31.0; over the seven hexaflake tasks of Figure 5, 125.0, 85.6 and 46.1 of 125; over the twelve of Figure 8, 125.0, 92.9 and 52.8.

E.2 More tasks of each benchmark

Figures 9, 10 and 11 extend Figures 4 and 6. Tasks are evenly spaced ranks of the verified makespan over the certified lower bound among the tasks the gate accepts, and share no shape with the main-text figures or with each other where the split allows.

E.3 The MNIST and ModelNet10 benchmarks

These two benchmarks predate the procedural families and run through the same pipeline, budget and gate; their shapes are constructed as in Appendix D.1. The MNIST benchmark has $N = 160$ disks on a square or a hexagonal lattice, every ordered pair of distinct digits on both lattices, 180 tasks. The ModelNet10 benchmark has $N = 1000$ spheres on the face-centered cubic lattice, one seeded draw per ordered pair of distinct classes, 90 tasks, of which we run thirty: every third, three per start class, with the three that share a shape with the checkpoint-selection tasks (Appendix D.3) replaced by the next task of the list. The gate certifies 52 of the 90 surface tasks (57.8%). At $N = 1000$ these are the largest instances we run. The results of our pipeline are demonstrated in Figure 12 and 13.

Two things in Table 3 do not appear on the procedural families. At $N = 160$ the transport baseline is close on the terminal count, and the separation is almost entirely in what the gate certifies. At $N = 1000$ the separation is in both: our model covers essentially the whole goal where the baseline covers about half, on the same tasks, budget and gate, and the gate certifies 18 of our 30 plans. The count of 987.9 is a floor: on two tasks the covering vote of Appendix D.2 settles on the wrong

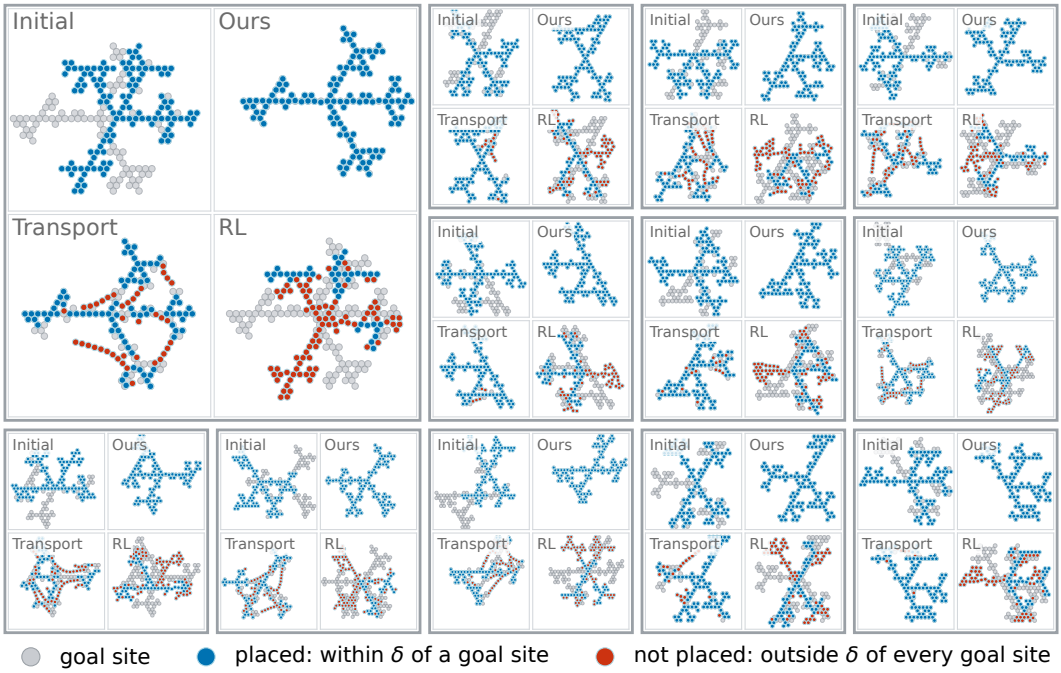

Figure 8: More hexaflake tasks, $N = 125$, in the layout of Figure 7 and the conventions of Figure 5, none sharing a shape with the tasks there.

Table 3: MNIST and ModelNet10 at $\kappa = 2.6$, in the columns of Table 1. The space rows carry the dilation prefix of Appendix D.4, and their $T/\hat{T}$ is against the certified root-mean-square bound while the budget was sized by the larger bottleneck scale (Appendix A.3).

Method	Certified (%) ↑	Placed ↑	$T/\hat{T}$ ↓	Wall (s) ↓
MNIST: $N = 160$, 180 tasks				
no motion	—	90.8/160	—	0
transport baseline	11.1	152.7/160	1.88	185
ours, without the projection	0.0	158.8/160	2.60	3
ours, full pipeline	**94.4**	**155.1/160**	**2.80**	**37**
ModelNet10: $N = 1000$, 30 tasks				
no motion	—	350.0/1000	—	0
transport baseline	0.0	516.1/1000	5.65	141
ours, without the projection	0.0	997.8/1000	5.41	28
ours, full pipeline	**60.0**	**987.9/1000**	**8.15**	**161**

translation and reads 839 and 871 where an assignment places all 1000, one of them a certified plan. Space is also where the projection is hardest. Eight of the twelve refusals are a residual overlap, a few thousandths of a diameter at the median and fourteen at worst, left by a barrier normalized over eighteen times more frame-module pairs than in the plane; the other four are a terminal miss, three by one or two modules and one where the field itself stopped sixty short. None is a cut in the contact graph.

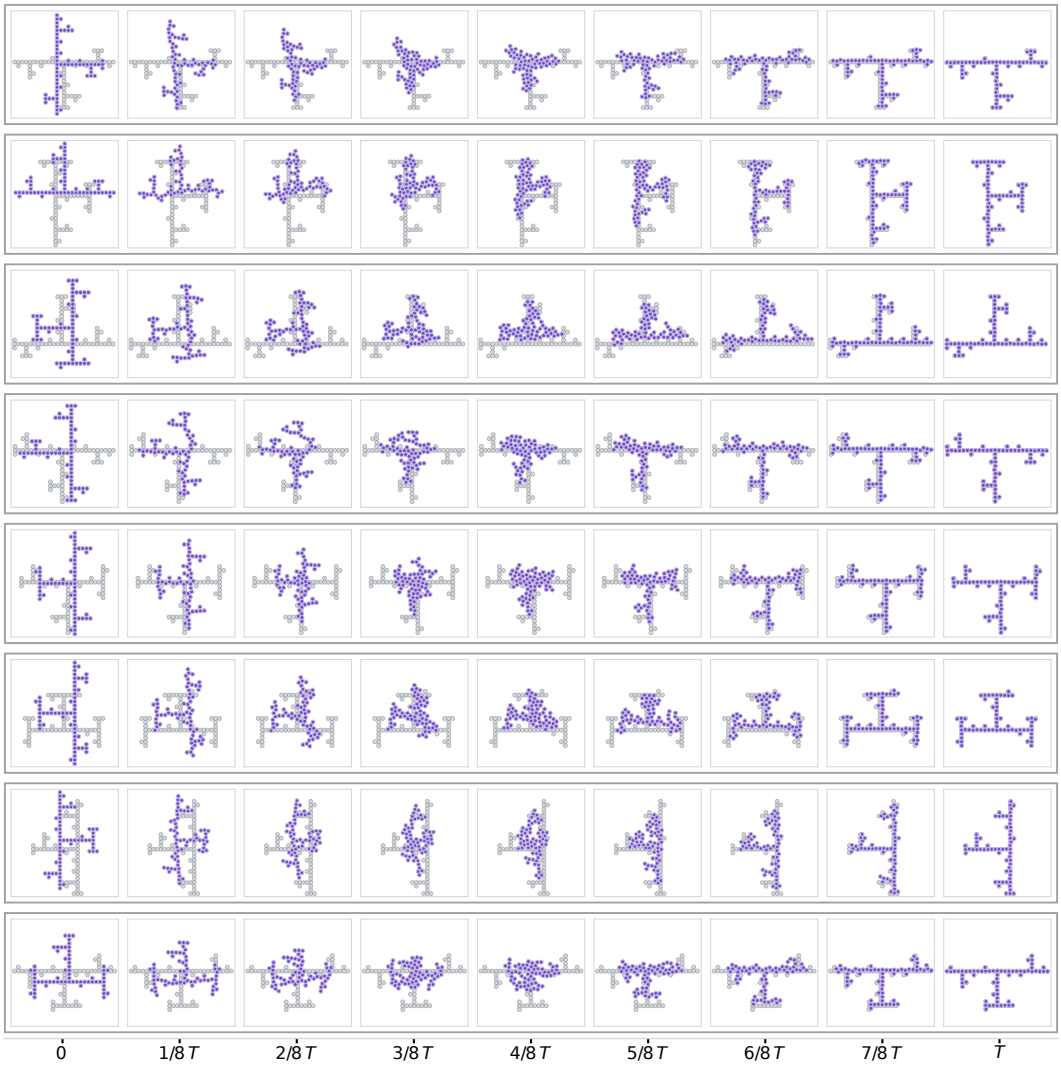

Figure 9: More tasks of the Vicsek benchmark, $N = 64$, in the conventions of Figure 4.

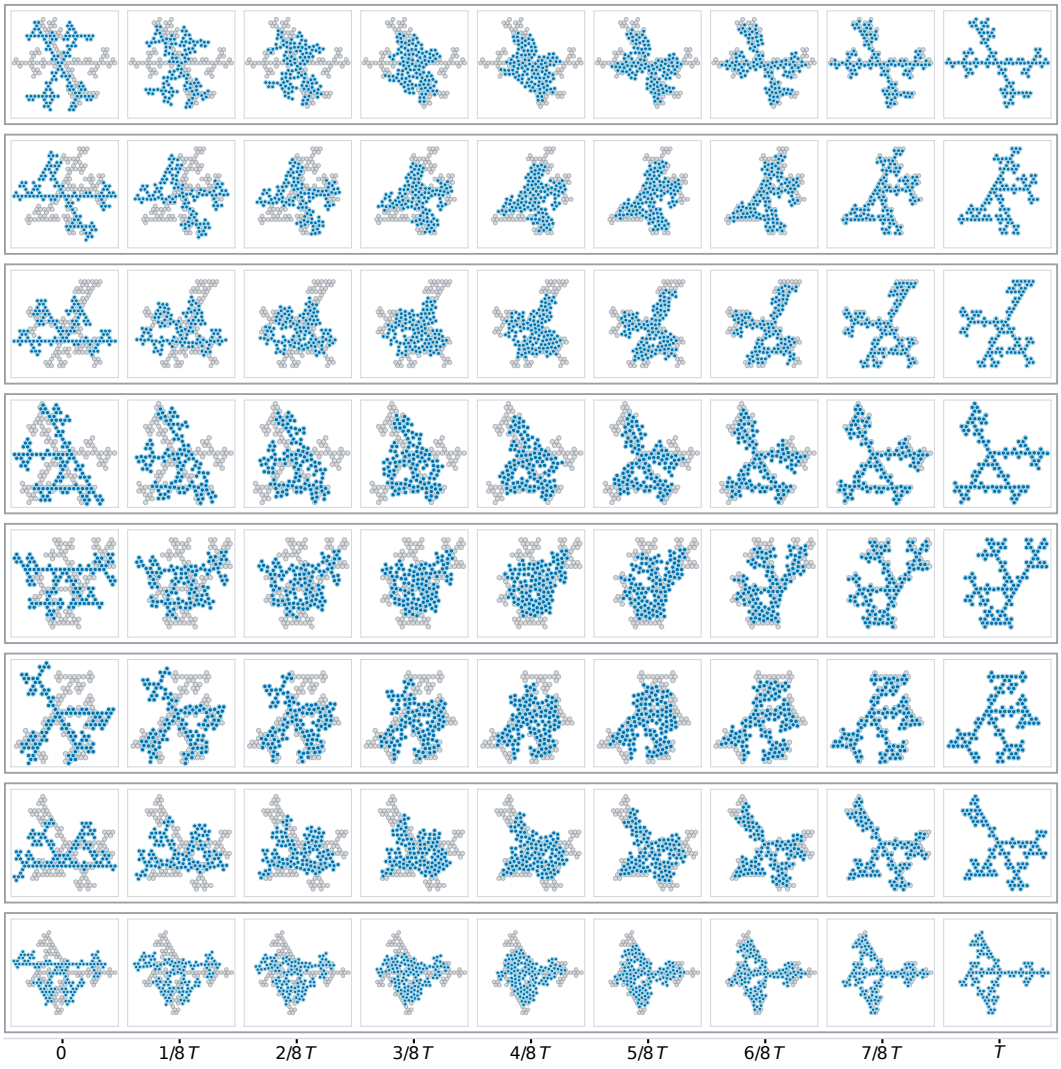

Figure 10: More tasks of the hexaflake benchmark, $N = 125$, in the conventions of Figure 4.

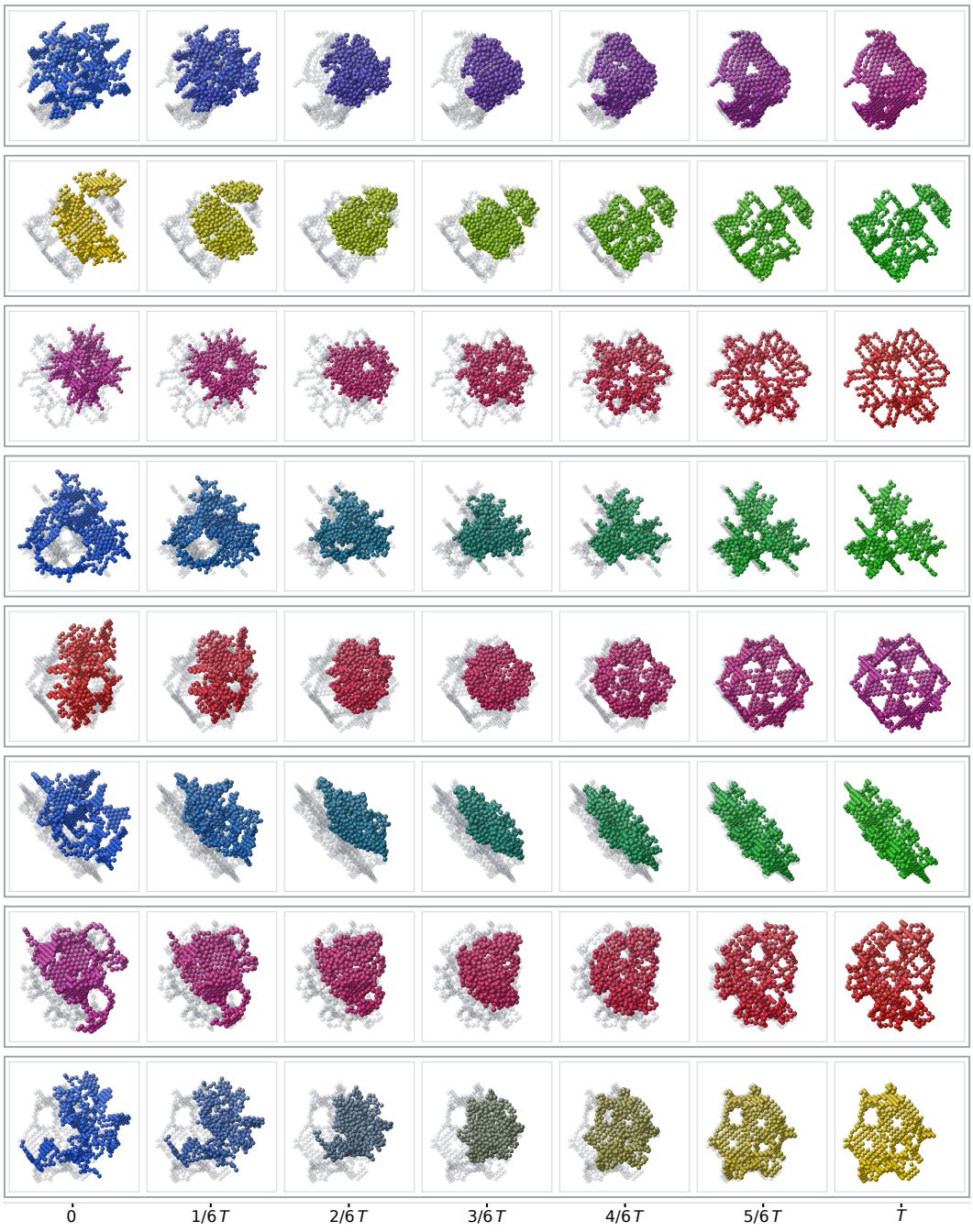

Figure 11: More tasks of the surface benchmark of Figure 6, every shape a panel of the archive's own gallery and every task reaching the held-out surface at one end.



Figure 12: Tasks of the MNIST benchmark, $N = 160$, in the conventions of Figure 4: evenly spaced ranks of the verified makespan over the certified tasks, with distinct digits and both lattices.



Figure 13: Tasks of the ModelNet10 benchmark, $N = 1000$ spheres on the face-centered cubic lattice, chosen by the same rule with distinct start and goal classes. The pale solid behind each frame is the goal, drawn where the trajectory ends.