Diffusion-Guided Multi-Arm Motion Planning

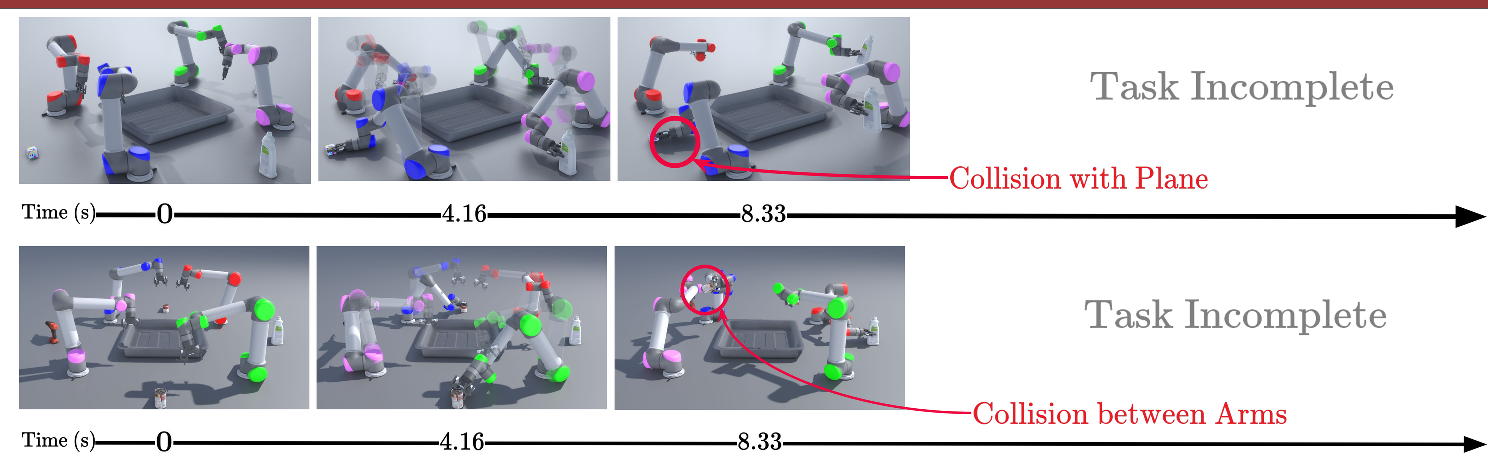
Viraj Parimi, Brian Williams
Massachusetts Institute of Technology

SCAN ME

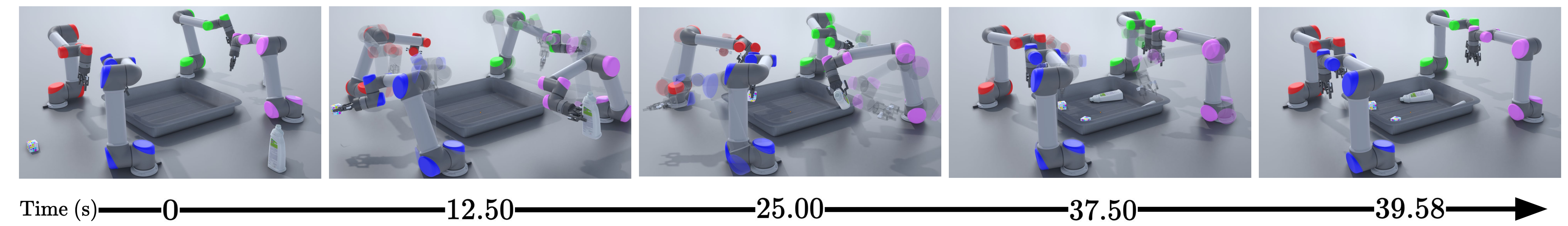
Enabling scalable coordination of multiple robotic arms by combining diffusion-based models with MAPF-inspired search without relying on massive datasets

① Motivation

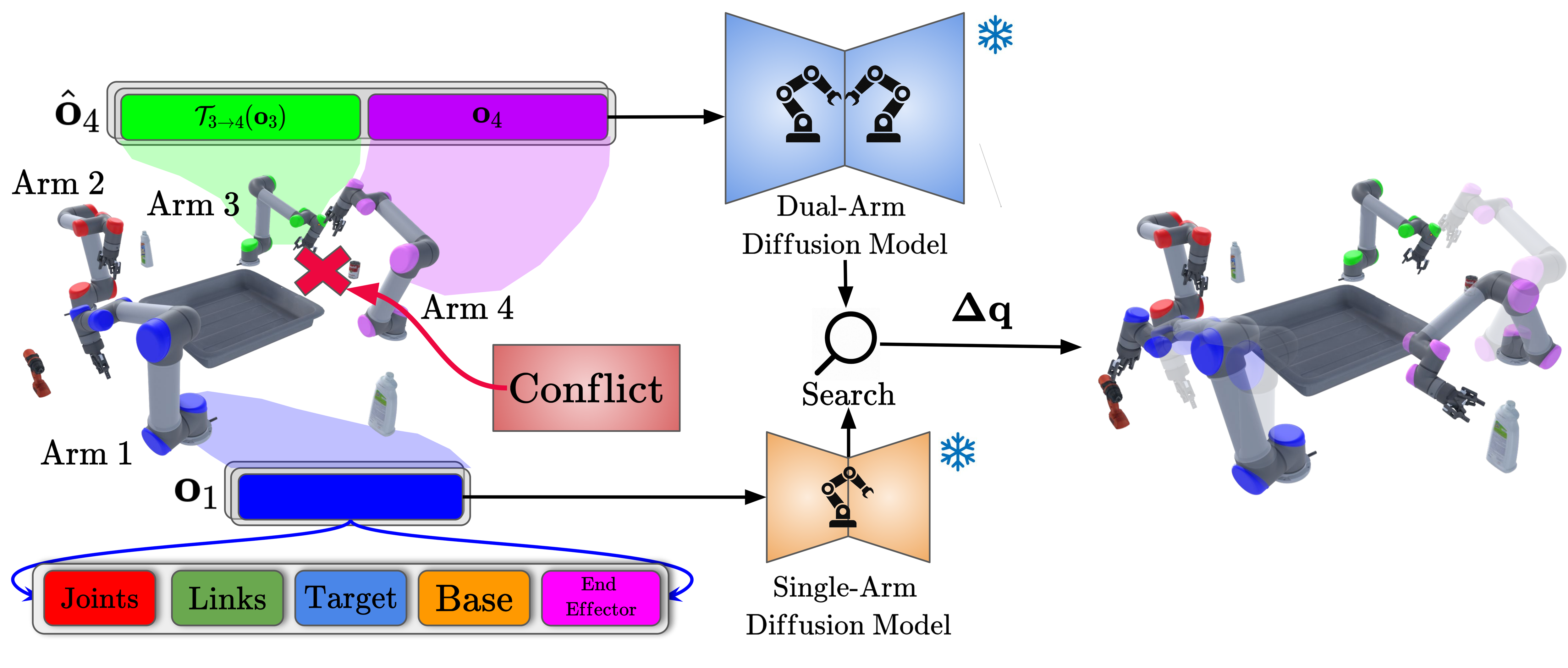
- Learning-based Multi-Arm Motion Planners [1]
  - Extremely fast
  - Reliance on massive datasets
- Search-based Multi-Arm Motion Planners [3]
  - Scale very well
  - Slower runtime



With Diffusion-Guided Search



② High-Level Overview



③ Algorithmic Details

Algorithm 1 Diffusion-Guided Multi-Arm Planner (DG-MAP)

Input: Models $\epsilon_{\theta_1}, \epsilon_{\theta_2}$
Output: Collision-free plan $\{\Delta q_i\}_{i=1}^N$ or best effort
Initialize: Frontier set $\mathcal{F} \leftarrow \emptyset$, collision cache $\mathcal{C} \leftarrow \emptyset$
for $i = 1 \dots N$ **do**
  $o_i \leftarrow \texttt{GetObs(i)}$
  $\mathcal{P}_i \leftarrow \epsilon_{\theta_1}(o_i, \cdot, \cdot)$
end for
// Generate initial plans independent of each other
$\mathcal{N}_0 \leftarrow \texttt{Node}(b = \vec{0}, \mathcal{K} = \emptyset)$
// Node stores indices and conflicting plan indices
$\mathcal{F}.\texttt{Insert}(\mathcal{N}_0, g(\mathcal{N}_0))$
while $\mathcal{F}$ not empty and time not exceeded **do**
  $\mathcal{N} \leftarrow \mathcal{F}.\texttt{ExtractMin}()$
  $\tau \leftarrow \{\Delta q_i^{\mathcal{N}.b_i}\}_{i=1}^N$
  $c \leftarrow \texttt{FindFirstCollision}(\tau, \mathcal{C})$
  if c is **null then**
    return τ, t^* // Solution found
  end if
  $(i, j, \hat{t}) \leftarrow c$ // Conflicting pair and time
  $t^* = \min(t^*, \hat{t})$ // Update earliest collision time
  $\kappa_i = \mathcal{N}.\mathcal{K} \cup \mathcal{N}.b_i$ // Attempt to fix for arm i
  $\texttt{Rebranch}(i, \kappa_i)$
  $\texttt{Repair}(i, j, \kappa_i)$
  $\kappa_j = \mathcal{N}.\mathcal{K} \cup \mathcal{N}.b_j$ // Attempt to fix for arm j
  $\texttt{Rebranch}(j, \kappa_j)$
  $\texttt{Repair}(j, i, \kappa_j)$
end while
return Best plan found in $\mathcal{F}$ based on cost // Timeout or failure

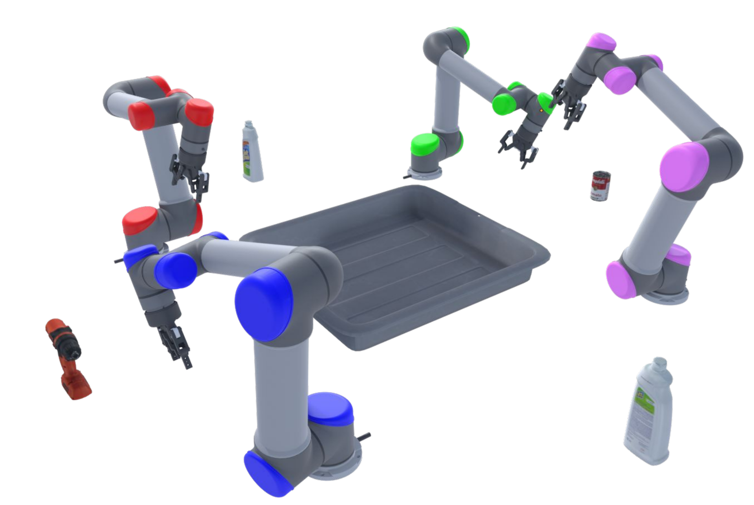
Algorithm 2 Generate Successors by Rebranch

Input: ego arm, conflicts κ
for $m = 1 \dots |\mathcal{P}_{\text{ego}}|$ **do**
  if $m \notin \kappa$ **then**
    $b_{\text{ego}}^{\text{new}} \leftarrow m$
    $b' \leftarrow (b_1, \dots, b_{\text{ego}}^{\text{new}}, \dots, b_N)$
    $\mathcal{N}' \leftarrow \texttt{Node}(b = b', \mathcal{K} = \kappa)$
    $\mathcal{F}.\texttt{Insert}(\mathcal{N}', g(\mathcal{N}'))$
  end if
end for

Algorithm 3 Generate Successors by Repair

Input: ego arm, other arm, conflicts κ, Model ϵ_{θ_2}
$\hat{o}_{\text{ego}} \leftarrow \texttt{GetPairedObs(ego, other)}$
Sample $\{\Delta q_{\text{ego}}^{\text{new},m}\}_{m=1}^B$ using $\epsilon_{\theta_2}(\hat{o}_{\text{ego}}, \cdot, \cdot)$
for $m = 1 \dots B$ **do**
  $b_{\text{ego}}^{\text{new}} \leftarrow \mathcal{P}_{\text{ego}}.\texttt{Update}(\Delta q_{\text{ego}}^{\text{new},m})$
  $b' \leftarrow (b_1, \dots, b_{\text{ego}}^{\text{new}}, \dots, b_N)$
  $\mathcal{N}' \leftarrow (b = b', \mathcal{K} = \kappa)$
  $\mathcal{F}.\texttt{Insert}(\mathcal{N}', g(\mathcal{N}'))$
end for

④ Pick-and-Place



Method	Success (↑)	Steps (↓)
Baseline-LD	0.375	**4479**
Baseline-ED	0.714	6018
DG-MAP	**0.890**	5390

Table 3: Success rate (%, higher is better) along with average number of steps (lower is better) taken by the methods to complete the full cycle of multi-arm pick-and-place task.

⑤ Comparison with Limited Data Baseline

Arms	Easy		Medium		Hard		Average	
	Baseline LD	DG-MAP Ours (↑)	Baseline LD	DG-MAP Ours (↑)	Baseline LD	DG-MAP Ours (↑)	Baseline LD	DG-MAP Ours (↑)
3	0.349	**0.984**	0.426	**0.975**	0.460	**0.965**	0.412	**0.975**
4	0.032	**0.981**	0.024	**0.969**	0.060	**0.953**	0.039	**0.968**
5	0.224	**0.972**	-	**0.958**	-	**0.905**	0.075	**0.945**
6	0.145	**0.976**	0.011	**0.921**	0.008	**0.888**	0.055	**0.928**
7	0.113	**0.955**	-	**0.918**	-	**0.907**	0.038	**0.926**
8	0.067	**0.951**	-	**0.933**	-	**0.888**	0.022	**0.924**

⑥ Comparison with Extended Data Baseline

Arms	Easy		Medium		Hard		Average	
	Baseline ED	DG-MAP Ours (↑)	Baseline ED	DG-MAP Ours (↑)	Baseline ED	DG-MAP Ours (↑)	Baseline ED	DG-MAP Ours (↑)
3	0.980	**0.984**	0.973	**0.975**	0.943	**0.965**	0.965	**0.975**
4	0.973	**0.981**	0.961	**0.969**	0.950	**0.953**	0.961	**0.968**
5	0.963	**0.972**	0.947	**0.958**	0.891	**0.905**	0.934	**0.945**
6	0.950	**0.976**	0.887	**0.921**	0.882	**0.888**	0.906	**0.928**
7	0.950	**0.955**	0.913	**0.918**	0.891	**0.907**	0.917	**0.926**
8	**0.951**	**0.951**	0.911	**0.933**	0.859	**0.888**	0.907	**0.924**

References

[1] H. Ha, J. Xu, and S. Song. Learning a decentralized multi-arm motion planner. In Conference on Robotic Learning (CoRL), 2020.
[2] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In Proceedings of Robotics: Science and Systems (RSS), 2023.
[3] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar, et al. Multi-agent pathfinding: Definitions, variants, and benchmarks. In Proceedings of the International Symposium on Combinatorial Search, 2019
[4] Z. Wang, J. J. Hunt, and M. Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning, 2023. URL https://arxiv.org/abs/2208.06193

Acknowledgements

This work was supported by the Defence Science and Technology Agency (DSTA). Any opinions, findings and conclusions in this material are those of the author(s) and do not necessarily reflect the views of DSTA.