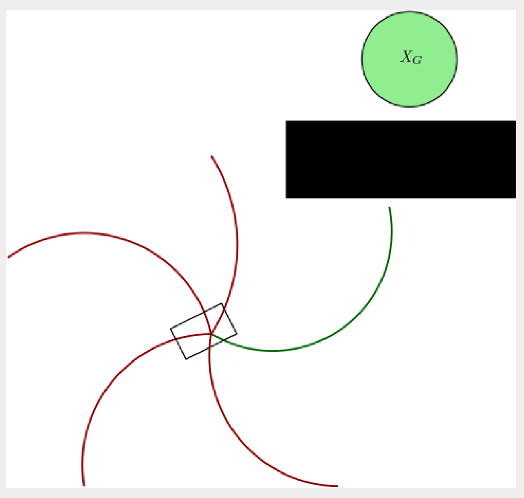
Learning Efficient Maneuver Sets for Kinodynamic Motion Planning

Aravind Sivaramakrishnan, Zakary Littlefield and Kostas E. Bekris
Department of Computer Science, Rutgers, the State University of New Jersey

Motivation

If a kinodynamic planner **[1]** has access to local maneuvers that balance an exploitation-exploration trade-off, the planner's per iteration performance is significantly improved.

- Exploitation maneuvers guide the system towards the goal given local heuristic information
- Exploration maneuvers move the system in different directions so as to deal with situations that the heuristic does not provide good guidance.



Exploitative (green) and Explorative (red) maneuvers for a robot planning to reach X_G (green circle) behind an obstacle (black box).

Motion planning with informed maneuvers

- Informed maneuvers can be computed online employing a metric similar to **[2]**, tailored to each state of the robot selected for propagation during planning.

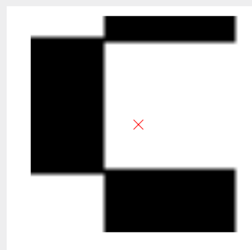
	Iteration	Comp. Time	Path Cost
Random	1471	0.2	50.47
Curated	686	12.15	48.13

First solution statistics between DIRT using random and curated maneuvers.

- Very effective in finding a high-quality solution in fewer number of iterations but computational becomes prohibitive.
- **Goal:** Develop an approach that achieves the same objective as the curation but can generate the maneuvers fast.

Input to the learning process

- A regular set of points $\mathbb{X}_{local}$ in the vicinity of x_0 are collision checked to generate a binary 2D map o_{local} indicating the presence of obstacles in the workspace.
- The heuristic $h(x)$ is also evaluated at each $x \in \mathbb{X}_{local}$, resulting in a 2D matrix h_{local}.



o_{local}

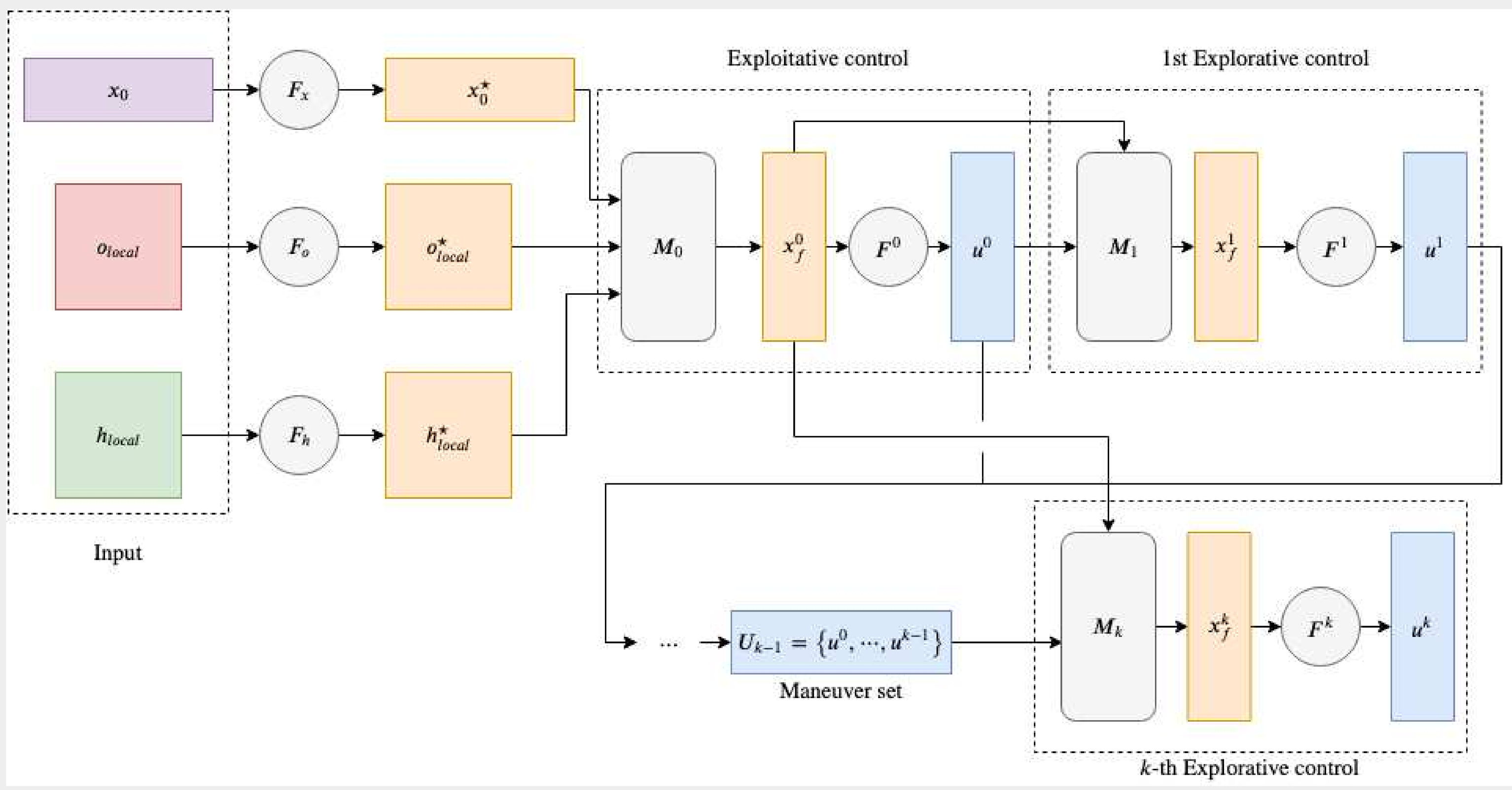
h_{local}

Proposed architecture

- Multi-layered neural networks F_x, F_o, F_h act on the inputs to produce $x_0^*, o_{local}^*, h_{local}^*$.
- An operator $M_0(x_0^*, o_{local}^*, h_{local}^*)$ produces feature vector x_f^0.
- Exploitative control u^0 is obtained as $u^0 = F^0(x_f^0)$, where F^0 is also a neural network.
- Remaining N exploratory controls are obtained as follows.

$$x_f^k = M_k(x_f^0, U_{k-1})$$
$$u^k = F^k(x_f^k)$$

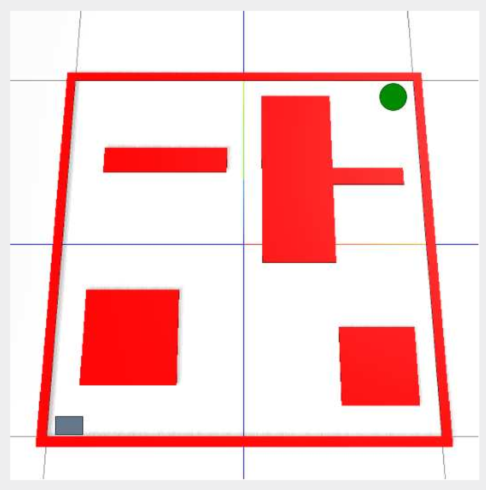
where for all $k \geq 1$, $U_k = \{u^0, u^1, .., u^{k-1}\}$. For the exploitative control ($k = 0$), U_{k-1} is the empty set.



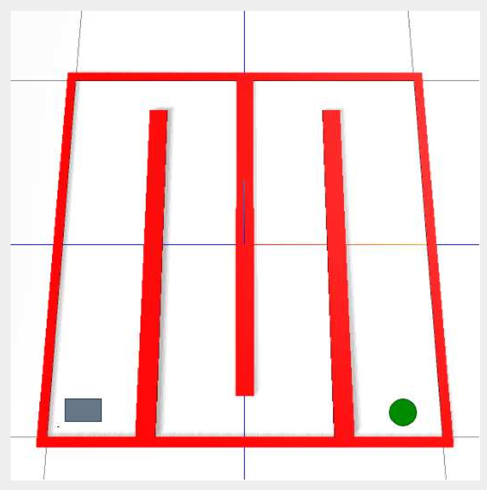
Computation graph of $U_k = \hat{f}(x_0, o_{local}, h_{local})$. For $k = N$, $U_k = \hat{U} = \{u^0, \cdots, u^N\}$.

Evaluation

- We consider a treaded vehicle with a 5 dim. state space (SE(2) augmented by steering angle and forward velocity) and 2 dim. control space (acceleration of left and right treads).
- For training, obstacles are randomly placed in a 2 dim. workspace so they cover one-third of the reachable workspace.
- The DIRT planner **[1]** is executed with the online curation procedure on multiple problem instances in such workspaces.
- Euclidean distance to the goal in the workspace is used as the heuristic function. The cost is the duration of the solution trajectory.
- For each node x_0 the planner selects to propagate, the training process stores the o_{local} and h_{local} maps, as well as a maneuver set $\hat{U}$ of size 5 curated from a set of 1000 randomly sampled maneuvers.
- Two networks - fully connected (`FC`) and convolutional (`Conv`)



`Greedy`



`Explore`

Environments: The grey rectangle us the starting pose of the robot (facing right) and the green circle is the goal region. The robot must avoid the red obstacles.

Solution statistics

We compare the performance of DIRT (after 50k iterations) for the following maneuvers:

- random (DIRT - Random)
- only exploitative control predicted by the network (DIRT - FC (Exploit), DIRT - Conv (Exploit))
- both exploitative and explorative controls are predicted by the networks ((DIRT - FC (All), DIRT - Conv (All))

Algorithm	NumSolns	FirstSolnIters	FirstSolnCost	FinalSolnIters	FinalSolnCost
DIRT - Random	30	3446.67	59.64	23277.57	49.44
DIRT - FC (Exploit)	30	2246.67	56.54	17050.37	49.89
DIRT - FC (All)	30	**620**	**47.58**	**16921.5**	**45.47**
DIRT - Conv (Exploit)	30	3366.67	65.03	27774.67	48.38
DIRT - Conv (All)	30	2006.67	54.8	25671.07	48.16

Solution statistics for `Greedy`. All values are averaged over `NumSolns`. Best values highlighted in bold.

Algorithm	NumSolns	FirstSolnIters	FirstSolnCost	FinalSolnIters	FinalSolnCost
DIRT - Random	30	15666.67	163.60	33254.13	149.47
DIRT - FC (Exploit)	29	**12000**	155	31794.86	140.06
DIRT - FC (All)	30	18766.67	**133.83**	**28119.66**	**130.92**
DIRT - Conv (Exploit)	29	27666.67	182.16	39924.96	172.14
DIRT - Conv (All)	30	14066.67	143.71	28194.83	139.43

Solution statistics for `Explore`. All values are averaged over `NumSolns`. Best values highlighted in bold.

Challenges

- An improved learning process is necessary to increase the rate of collision-free maneuvers.
- Current cost of network inference is more expensive than returning a random control.
- For higher dimensional systems, more complex environments and realistic sensing input, there are additional considerations related to data efficiency and uncertainty that must be mitigated.

References

1. Littlefield, Z., and Bekris, K. E. 2018. Efficient and asymptotically optimal kinodynamic motion planning via dominance-informed regions. In *IROS*.
2. Green, C.J., and Kelly, A. 2007. Toward optimal samplingin the space of paths. In *ISRR*.

{as2578, zwl2, kb572}@rutgers.edu