Homotopy Constrained Planning

Muhammad Suhail
msaleem2@andrew.cmu.edu

Ramkumar Natarajan
rnataraj@cs.cmu.edu

***Abstract*— Planning a feasible path for robots constrained to specific topological class has application in several areas. In this project, we illustrate the importance of homotopy constrained planning, by searching over H-signature augmented graphs and discussing its application and impact in two different domains namely - i) Point robot ii) n-DOF manipulator. The other part of the work deals with combining the homotopy constrained planning with simple trajectory optimization method like gradient descent to find a smooth trajectory with globally optimal cost. We note that the trajectory optimization can find the global optima only in conjunction with homotopy constrained planning and finds a locally optimal solution with respect to an initial seed otherwise.**

I. INTRODUCTION

Homotopy classes for trajectories are defined for a fixed start and goal location in an environment. Two trajectories are said to be in the same homotopy class, if and only if, they can be continuously deformed into one another, without intersecting any obstacles. For a fixed environment there are countably infinite number of homotopy classes as the path could loop around an obstacle infinitely many times.

For a group of mobile robots, planning paths in different homotopy classes could result in an increase in visibility. In some cases avoiding or planning in specific homotopy classes might prove fruitful, due to the dynamic/probabilistic nature of the environment in the other homotopy classes. Homotopy constrained planning also has a significant impact in planning for manipulators and in the field of trajectory optimization, which are explained in detail in the below sections.

II. SIGNATURES

Signatures are means of uniquely representing paths in different homotopy classes in a neat and concise way, so that they could be easily integrated in the planning process. One of the primary methods adopted for constructing the signatures is the beam based reduced word approach [2]. For a 2-D environment, in this technique we consider representative points for every obstacle in the environment. From these points beams are projected vertically upwards as shown in the figure. Depending upon the beam and the direction in which the trajectory crosses the beam, a letter (t_k or $t_k^{'}$) is appended onto the reduced word. Also it has to be noted that two consecutive letters of the form $t_j t_k t_k^{'}$ is equivalent to t_j as the path from $t_j t_k t_k^{'}$ involves crossing beam k twice consecutively in opposite directions and can be continuously shifted to paths belonging to the t_j homotopy class (this is the *reduction* part of the reduced word approach).

The two authors contributed equally

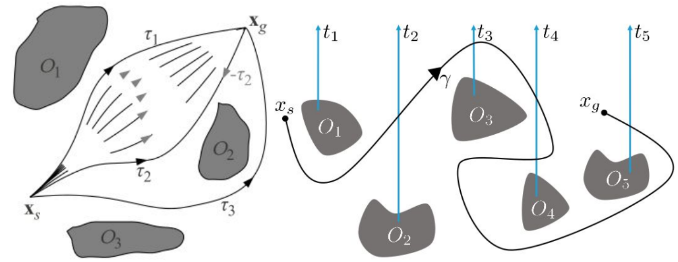

Fig. 1. Left[1]: τ_1 and τ_2 are in the same homotopy, whereas τ_3 is in a different homotopy. Right[2]: Signature of the given trajectory is $t_2 t_3 t_4 t_4^{'} t_5^{'} = t_2 t_3 t_5^{'}$

Bhattacharya *et al.*[1] uses a L-value based representation for the signature construction. One of the major advantages of using the reduced word approach over the L-value based approach is the fact that we can choose the homotopy in which we would like to plan in advance. In the L-value based approach, the signatures are computed only after you find at least one path in that particular homotopy, hence you can not choose a specific homotopy while defining the goal initially.

III. SIGNATURE AUGMENTED GRAPH

Thus the planning is done on a H-signature augmented graph ($\mathcal{G}_\mathcal{H}$), which has been clearly explained by Bhattacharya *et al.* [1] as follows,
$\mathcal{G}_\mathcal{H} = (\mathcal{V}_\mathcal{H}, \mathcal{E}_\mathcal{H})$
where,

1) $\mathcal{V}_\mathcal{H}$ = (v, h), $v \ \epsilon$ Vertices of the original graph, h=$\mathcal{H}$(v$_s$ v), H((v$_s$ v)) is the signature of the trajectory from start state to state v.
2) An edge $(v, h) \rightarrow$ (v$^{'}$, $h^{'}$) is in $\mathcal{E}_\mathcal{H}$ for $(v, h)\epsilon \ \mathcal{V}_\mathcal{H}$ and $(v^{'}, h^{'}) \ \epsilon \ \mathcal{V}_\mathcal{H}$, iff
   a) The edge $v \rightarrow v^{'} \ \epsilon \ \mathcal{E}$ and,
   b) $h^{'} = h + \mathcal{H}(v \rightarrow v^{'})$, where $\mathcal{H}(v \rightarrow v^{'})$ is the signature of the edge $v \rightarrow v^{'} \ \epsilon \ \mathcal{E}$
3) The cost/weight associated with an edge $(v, h) \rightarrow (v^{'}, h^{'})$ is same as that associated with edge $(v \rightarrow v^{'}) \ \epsilon \ \mathcal{E}$.

IV. PLANNING FOR A MOBILE POINT ROBOT

We wanted to produce homotopy constrained plans for the point robot in a 2D environment (500×500) (8-connected grid), hence the state space was three dimensional (X,Y, and Signature). Initially, we set out with the aim of constructing intelligent heuristics which would help the search explore only in the required homotopy class thereby reducing the number of expansion significantly. We tried leveraging the properties of the signatures (both L-value and Reduced Word), however, finding such a heuristic is a really difficult task and will require considerably more time.

The Heuristic used by Bhattacharya *et al.*[1] was a 2-D Euclidean Heuristic. Although it serves the purpose, it results in a lot of expansions when a path in the required homotopy is extremely suboptimal. In such a case, we would first find all the paths which are more optimal in the 2D state space before finding a path to the goal with the right signature.

Another heuristic which will help guide the search better is the mismatch heuristic which is usually associated with Symbolic planning. This heuristic is basically the total number of letters which are mismatched between the current state's signature and the goal state's signature.

For the planner, we have implemented both an A-star planner making use of the Euclidean heuristic and an MHA-star planner making use of the Euclidean heuristic as the anchor search and weighted combination of the Euclidean Heuristic and the mismatched heuristic described above. It was seen that MHA-star, results in considerably lesser number of expansions when the trajectory to the goal is extremely suboptimal as compared to the optimal path to the 2Dimensional goal without considering signatures/homotopy classes. MHA-star expands 24560948 states to find the path shown in Fig 2 (Left), for the same configuration, A-star expands 36239923 states.

V. Planning for a N-DoF Planar Manipulator

For a manipulator, we define the homotopy with respect to the trajectory of the end-effector. In manipulation planning, homotopy arises when we need to reason about which way to grasp an object around an obstacle. Reasoning in terms of homotopy becomes further critical for dual-arm planning when we need to manipulate an object together with both the arms from one side of the obstacle to the other. In this case, the end-effector trajectory of both the arms from the base position should be in the same homotopy for grasping an object.

We demonstrate the working of our homotopy constrained manipulation planner on a 5-DoF planar manipulator. We use the same beam-based reduced word approach as explained before to identify and constrain the planner with different homotopy classes. The state space consists of the joint-angles of the planar arm and the H-signature of the end-effector trajectory from the start configuration. The goal consists of the desired end-effector pose and the desired homotopy class through which to reach the desired end-effector pose. Although the planning is done in the C-space with the joint-angles, it is important to note that the cost function is in terms of task-space variables. The cost function is a weighted combination of the Euclidean distance between the end-effector poses and the H-signature mismatch. This focuses search to find a homotopy constrained end-effector trajectory that is minimally sub-optimal in terms of trajectory length.

The plan is obtained using an ARA-star search as the state space is extremely large for A-star to converge. Thus during every expansion of a vertex in the graph, we solve forward kinematics to find the end-effector pose and the corresponding task-space h-signature to augment the state space (which is in terms of the joint angles). The search terminates when the node whose C-space configuration equals the desired task-space end effector pose and contains the desired h-signature is expanded. The results are visualized in Fig. 3

VI. Homotopy Planning with Trajectory Optimization

Trajectory optimization is an optimal control problem which is typically used in continuous spaces and require an initial seed to find a trajectory that optimizes an objective cost function. The trajectory is parametrized in terms of state variables and control inputs. Many of these methods operate on gradients and local approximations and thus require the cost function to be smoothly differentiable. As a consequence, trajectory optimization methods holds the reputation to find feasible trajectories for continuous dynamical systems. However, these methods find the optimal trajectory with respect to an initial seed and often times end up in local minimum because of bad initial seeds. We note that in most cases these local minimums are caused by obstacles in the environment and they can be mitigated by optimizing initial seeds from all non-looping homotopy classes. Further, we assert that gradient based trajectory optimization always converge to a minimum in the same homotopy class as the initial seed. This is because, as these methods follow the direction of negative gradient at every point along the trajectory they do not explore the space of the cost function on the other side of the obstacle (see Fig. 4).

In our approach, we identify all the non-looping homotopy classes using the permutation of letters in the signature such that a single representative letter does not appear twice in the reduced word. Next we run the point robot discrete planner on the augmented graph to find a discrete (non-smooth) initial seed for every homotopy class. We perform gradient descent over the given smooth cost function using every non-looping discrete initial seed obtained in the previous step to obtain a C_2 continuous trajectory by enforcing smoothness constraints. From the trajectories resulting from gradient descent the one with least cost is the C_2 smooth globally optimal solution. The results are visualized in [Fig.]

References

[1] Bhattacharya, Subhrajit, Vijay Kumar, and Maxim Likhachev. "Search-based path planning with homotopy class constraints." Third Annual Symposium on Combinatorial Search. 2010.

[2] Ranganeni, Vinitha, Oren Salzman, and Maxim Likhachev. "Effective Footstep Planning for Humanoids Using Homotopy-Class Guidance." arXiv preprint arXiv:1712.00531 (2017).

APPENDIX

Video clips of the simulation and experimental results are published in this link.

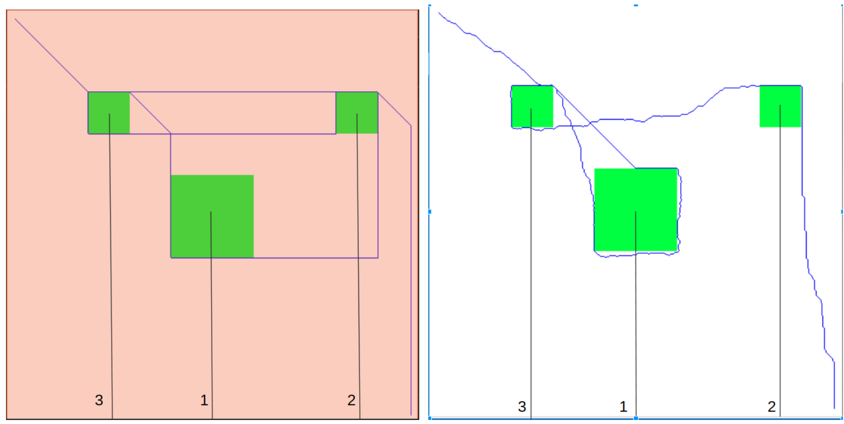

Fig. 2. **Left:** Path planned for signature 123 for point robot using A-star. **Right:** Path planned for signature 13 for point robot using MHA-star

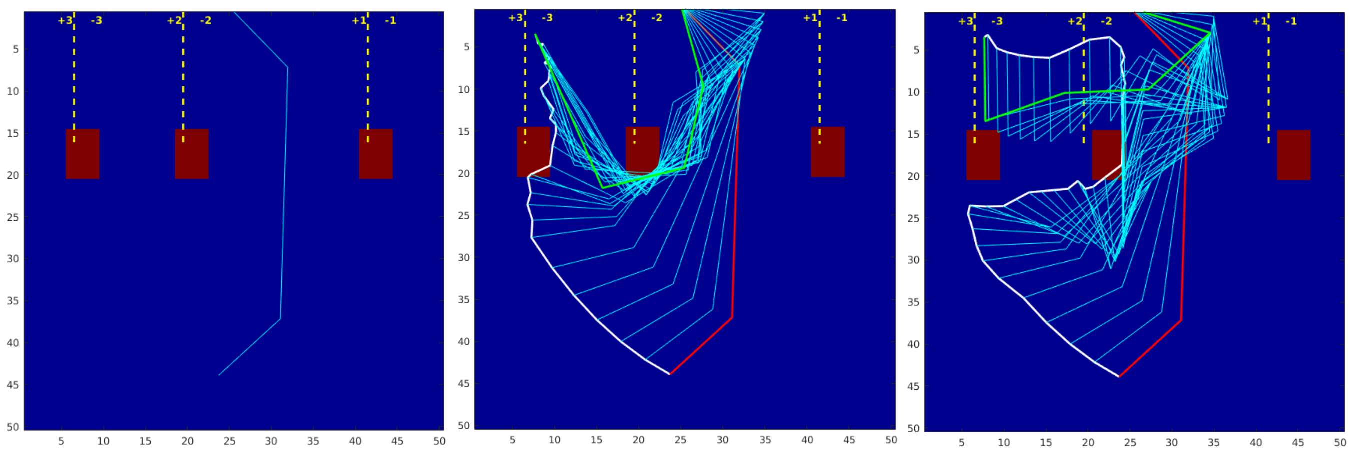

Fig. 3. **Left:** 2D 5 DoF planar arm planning environment with different homotopy classes identified using signatures. **Center:** ARA-star planner with $NULL$ homotopy class constraint for end effector trajectory. **Right:** ARA-star planner with -2 homotopy class constraint for end effector trajectory. The initial and final configuration of the planar arm is shown in red and green respectively. The planned end effector trajectory is shown in white.

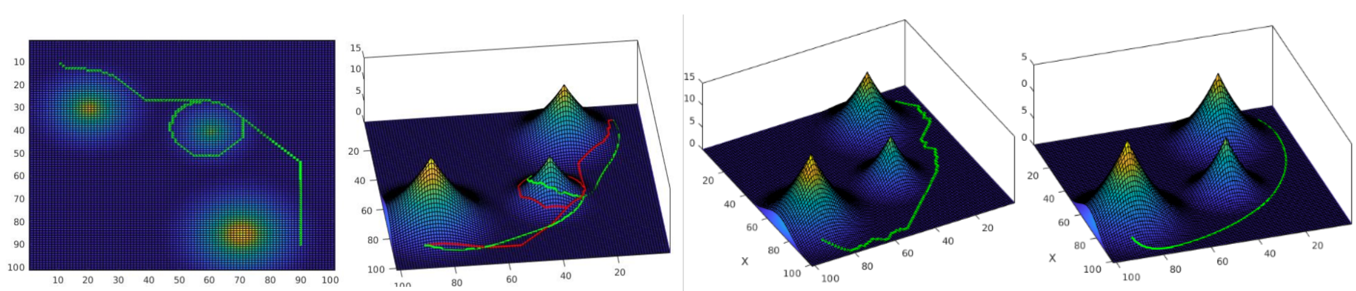

Fig. 4. **Left:** The discrete planner constrained to find a path in one of the looped homotopy classes. The path is used as an initial seed to find the smooth trajectory with gradient descent. **Right:** The globally optimal trajectory across all the non-looping homotopy classes found through our approach.