RRT*-guided robot learning for planning

CS 593

Milestone 2

Shivam Bhat (PUID : 33760929)
Vidyaa Krishnan Nivash(PUID : 34741863)

Problem statement

Problem statement
Methodology (Full pipeline)
What has been achieved so far M1-M2
including videos, results, etc
What's left (Describe M3)
A slide describing the contribution of each member of the team Conclusion

https://piazza.com/class/lcoyhxjko165ul/post/149

Project Presentation (PDF only)
Zip folder containing Readme + Demo Videos + Code
Only one of the group members needs to submit it.

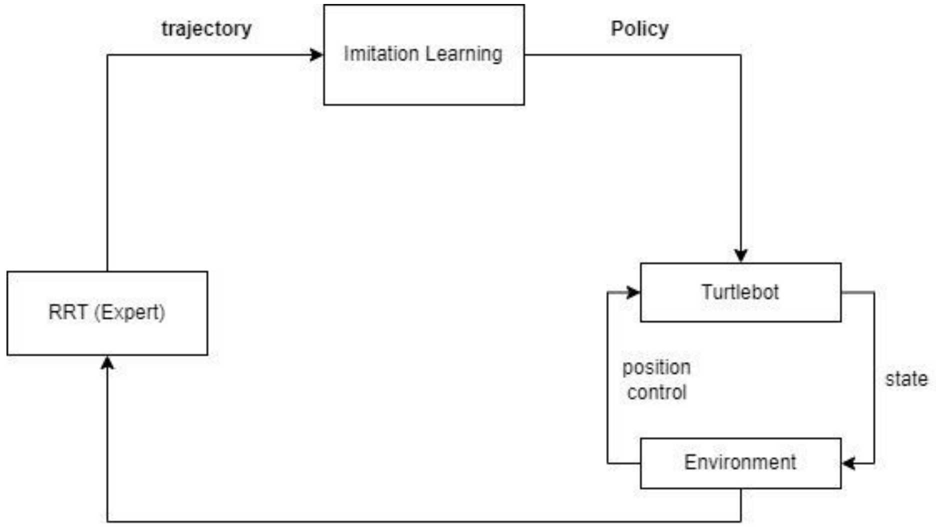
Problem statement

1. Setup maze environment with Turtlebot in Pybullet
2. Run RRT* to gather data, including local lidar observations
3. Implement an imitation learning policy to generate the next waypoint toward the goal using only the local lidar observations

Problem statement

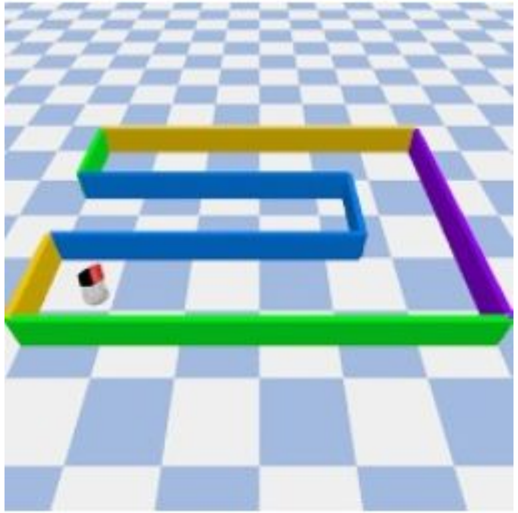
1. In this work we implement an imitation learning policy for a turtlebot in a maze environment.
2. RRT* is a probabilistically complete algorithm used for path planning in robotics that aims to find an optimal path from the start to the goal while avoiding obstacles.
3. Initially RRT* is used to gather data, including local lidar observations.
4. Finally an imitation learning policy is implemented to generate the next waypoint toward the goal using only the local lidar observations.

Methodology



Milestone 1

1. Setup Maze Environment
2. Setup Turtlebot
3. Route Turtlebot using a random policy
4. Setup Obstacles
5. Code and Documented Report



Maze environment with turtlebot

Milestone 2

1. Map 3D to a 2D environment
2. Implement RRT* for this mapped environment
3. RRT* bounded inside the maze boundary - checks robot collision with walls and obstacles
4. Routing of Turtlebot along found path in 3D environment
5. Implement Lidar
6. Collect Lidar data

Milestone 3

1. Implement an imitation learning policy to generate the next waypoint toward the goal using only the local lidar observations
2. Turtlebot should be able to navigate its on using the learned policy

Imitation Learning

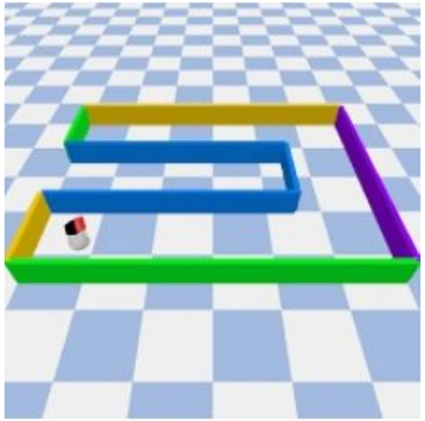
Imitation learning involves learning a policy by observing expert demonstrations
There are different ways in which imitation learning can be performed:

- Behaviour cloning - just takes the expert demonstrations and performs supervised learning to get a policy
- Interactive learning - takes the expert state, action pair and interacts with environment to update its policy eg: DAgger, SEARN
- GAIL - Has a generator and discriminator to learn a cost function from the expert demonstrations and updates the policy using inverse reinforcement learning method

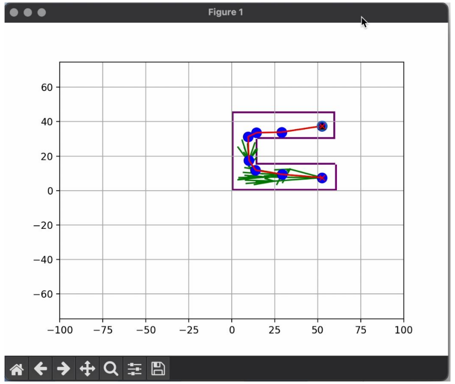
We are planning to implement behaviour cloning as a first step and then implement Dagger

Results

RRT*



3D Env

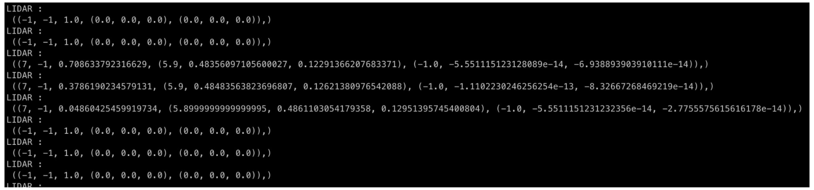


RRT* in 2D Env

Results

_p.rayTest(rayFromPosition, rayToPosition, physicsClientId*)

objectUniqueId	int	object unique id of the hit object
linkIndex	int	link index of the hit object, or -1 if none/parent.
hit fraction	float	hit fraction along the ray in range [0,1] along the ray.
hit position	vec3, list of 3 floats	hit position in Cartesian world coordinates
hit normal	vec3, list of 3 floats	hit normal in Cartesian world coordinates



```
LIDAR :
 ((-1, -1, 1.0, (0.0, 0.0, 0.0), (0.0, 0.0, 0.0)),)
LIDAR :
 ((-1, -1, 1.0, (0.0, 0.0, 0.0), (0.0, 0.0, 0.0)),)
LIDAR :
 ((7, -1, 0.708633792316629, (5.9, 0.48356097105600027, 0.12291366207683371), (-1.0, -5.551115123128089e-14, -6.938893903910111e-14)),)
LIDAR :
 ((7, -1, 0.3786190234579131, (5.9, 0.48483563823696807, 0.12621380976542088), (-1.0, -1.1102230246256254e-13, -8.32667268469219e-14)),)
LIDAR :
 ((7, -1, 0.04860425459919734, (5.8999999999999995, 0.4861103054179358, 0.12951395745400804), (-1.0, -5.5511151231232356e-14, -2.7755575615616178e-14)),)
LIDAR :
 ((-1, -1, 1.0, (0.0, 0.0, 0.0), (0.0, 0.0, 0.0)),)
LIDAR :
 ((-1, -1, 1.0, (0.0, 0.0, 0.0), (0.0, 0.0, 0.0)),)
LIDAR :
 ((-1, -1, 1.0, (0.0, 0.0, 0.0), (0.0, 0.0, 0.0)),)
LIDAR :
```

Contribution

Both team members contributed equally towards every part of the project.

Both discussed and contributed equally toward fundamental code modules and architecture as it is an essential step for setup and understanding the given problem statement. Post that we divided our work in terms of code modules

Shivam B- Tech Design,Environment and Maze Setup, RRT*, Lidar

Vidhya K- Literature Review, Position Control,Robot Motion, Imitation learning

References

1. http://www.cs.columbia.edu/~allen/F15/NOTES/icckinematics.pdf
2. https://gerardmaggiolino.medium.com/creating-openai-gym-environments-with-pybullet-part-2-a1441b9a4d8e
3. https://github.com/astrid-merckling/bullet_envs

Thanks