Datastructures

struct simplestate :
- Used for backwards Dijkstra
- stores x, y and g-value

struct state :
- Used for A*
- Attributes :
  - coord (x, y)
  - g, h, f
  - number of steps
  - pointer to parent state

Global datastructures :

heuristic : Unordered map <int, int>, int
→ to store heuristic values of relevant states calculated in backward Dijkstra

gval : Unordered map <int, int>, int
→ to store g value of state in A*
global for testing purposes

path : stack of state pointers
→ contains sequence of optimal states for planner to follow

→ Filled inside A*

Trajpoints: Unordered map <int, int>, int

→ Contains goal states and their g values

→ Filled in fcn get_goalpoints

Optimality

- Uses A* with backward A* as heuristic
- Limited by maximum step size
- selects closest reachable point on target trajectory and follows it backwards

⇒ suboptimal due to

- limited step size
- sub optimal goal point search

Offline planner

get_goalpoints

- Finds set of goalpoints from target trajectory
- Selects goalpoints depending on manhattan distance to robot starting position
- initial g-value depends on index

Find Traj Index

- given points x, y finds index of trajectory at which point occurs

backward_djikstra

- initialized open and closed list
- calls getgoalpoints and adds those points into open_list
- runs backward djikstra

A star

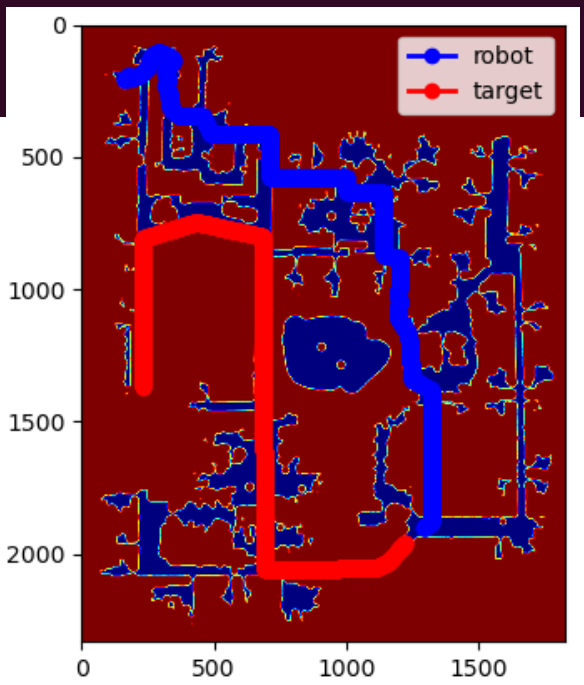
- does A* while also keeping track of number of steps taken to reach a state
- only processes states that do not overshoot step limit

Map1:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map1.txt
map size: 1825,2332
collision threshold: 100
robot pose: 159,208
target_steps: 5345

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 2871
moves made = 2871
path cost = 2871
```

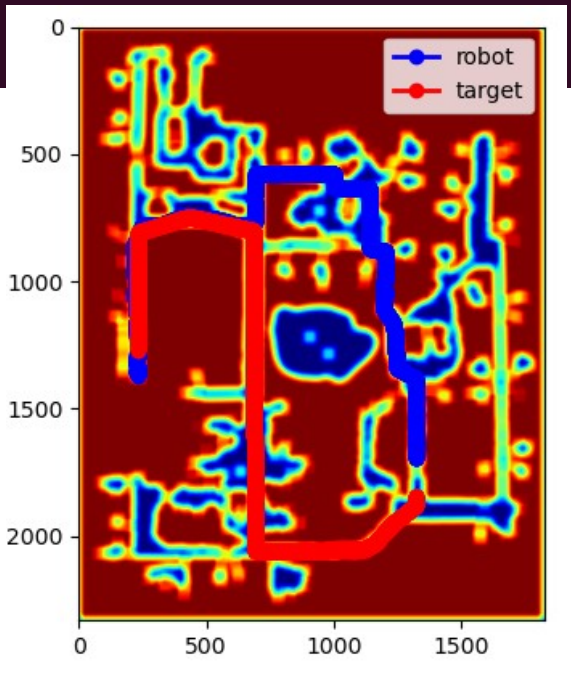


Map2:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map2.txt
map size: 1825,2332
collision threshold: 6500
robot pose: 231,1369
target_steps: 5245

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 2954
moves made = 2953
path cost = 5059711
```

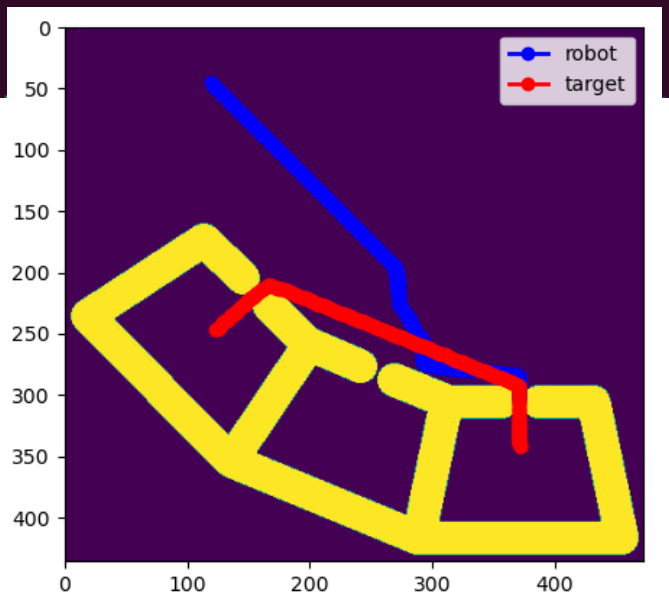


Map3:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map3.txt
map size: 473,436
collision threshold: 100
robot pose: 119,45
target_steps: 792

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 335
moves made = 334
path cost = 335
```

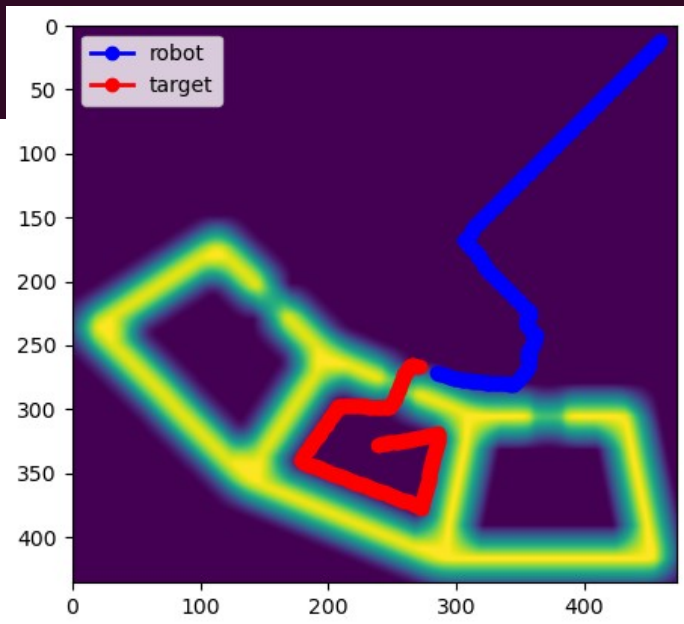


Map4:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map4.txt
map size: 473,436
collision threshold: 5000
robot pose: 459,12
target_steps: 792

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 336
moves made = 335
path cost = 24192
```

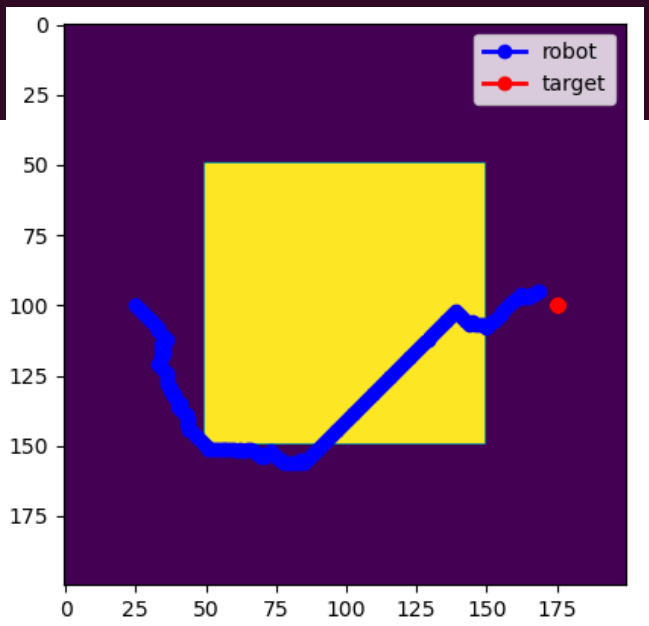


Map5:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map5.txt
map size: 200,200
collision threshold: 100
robot pose: 25,100
target_steps: 182

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 175
moves made = 175
path cost = 3115
```

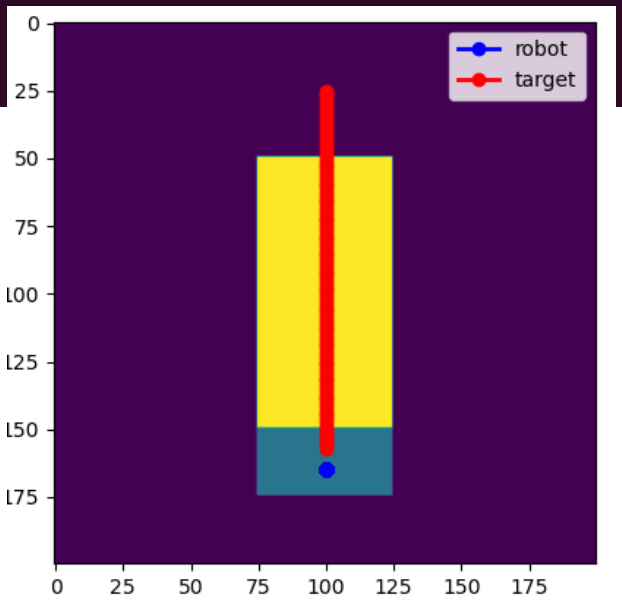


Map6:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map6.txt
map size: 200,200
collision threshold: 100
robot pose: 100,165
target_steps: 141

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 140
moves made = 0
path cost = 2800
```

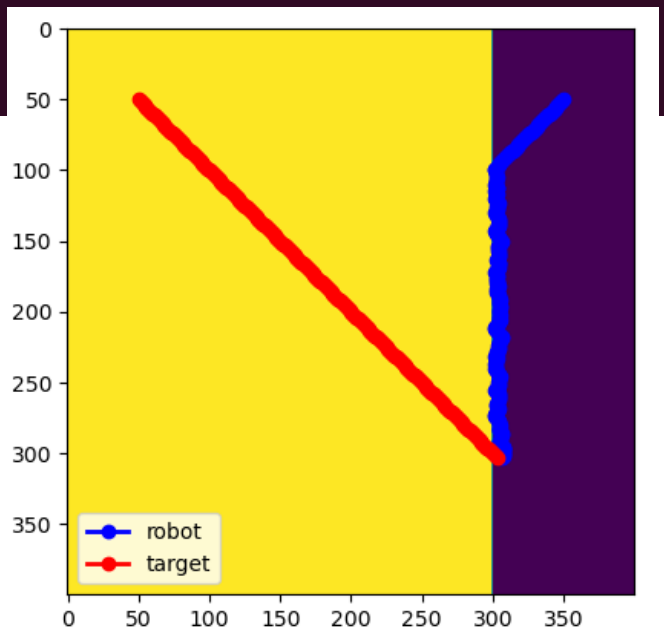


Map7:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map7.txt
map size: 400,400
collision threshold: 50
robot pose: 350,50
target_steps: 301

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 255
moves made = 255
path cost = 255
```

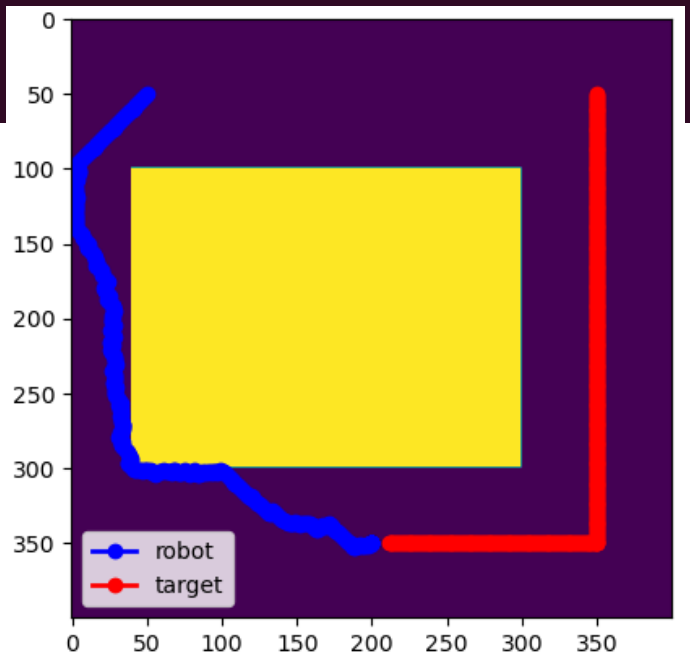


Map8:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map8.txt
map size: 400,400
collision threshold: 50
robot pose: 50,50
target_steps: 452

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 451
moves made = 410
path cost = 451
```



Map9:

```
Reading problem definition from: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/maps/map9.txt
map size: 400,400
collision threshold: 50
robot pose: 50,200
target_steps: 602

Running planner
Writing robot trajectory to: /home/kaust/Downloads/16782_HW1_fall24_v1/16782-HW1/code/output/robot_trajectory.txt

RESULT
target caught = 1
time taken (s) = 376
moves made = 363
path cost = 376
```