sim_rrt

October 5, 2020

1 RRT Sampling-Based Motion Planning

```
In [8]: # The autoreload extension will automatically load in new code as you edit files,
        # so you don't need to restart the kernel every time
        %load_ext autoreload
        %autoreload 2

        import numpy as np
        import matplotlib.pyplot as plt
        from P2_rrt import *

        plt.rcParams['figure.figsize'] = [8, 8] # Change default figure size
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

1.0.1 Set up workspace

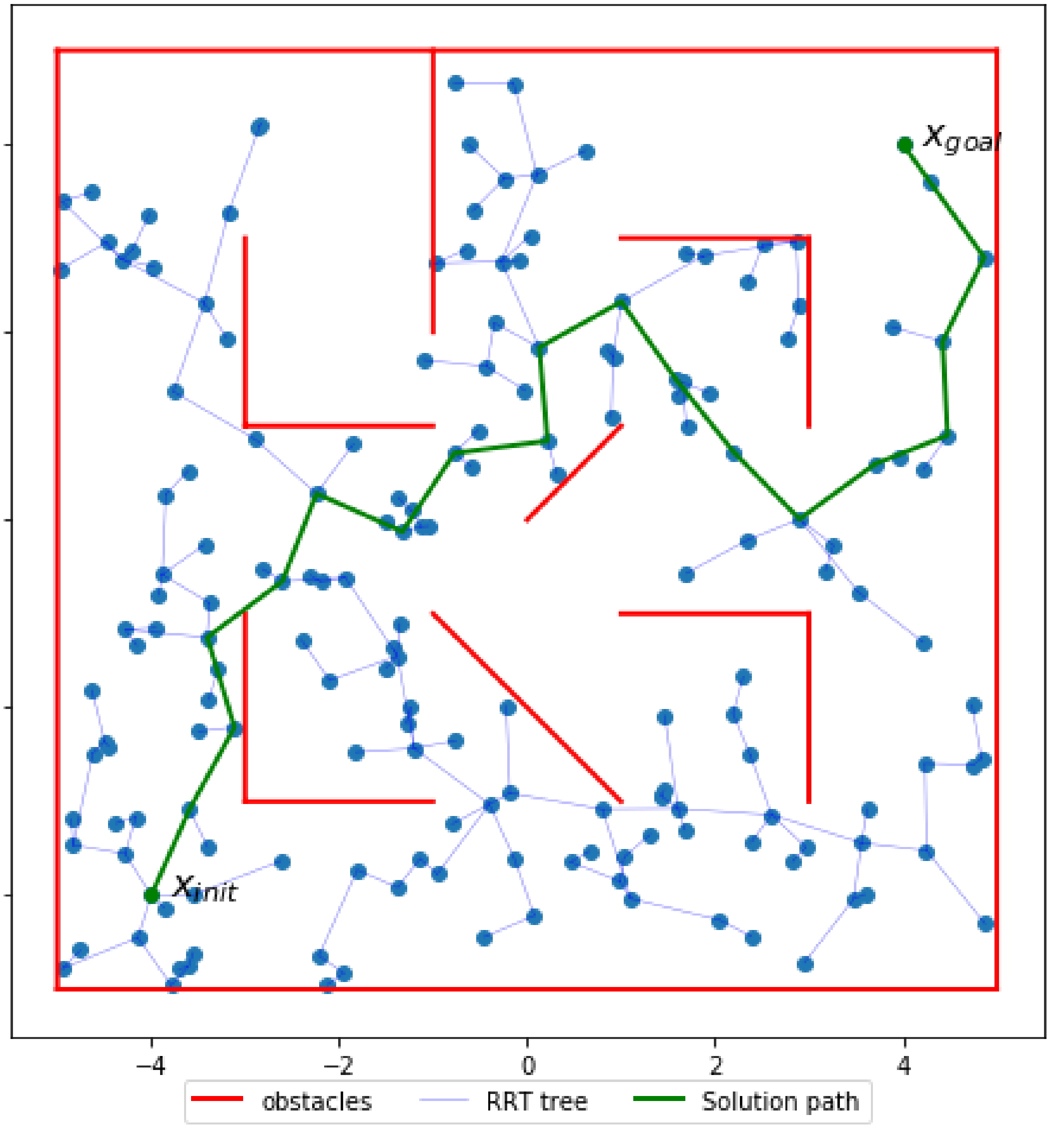
```
In [9]: MAZE = np.array([
            (( 5, 5), (-5, 5)),
            ((-5, 5), (-5,-5)),
            ((-5,-5), ( 5,-5)),
            (( 5,-5), ( 5, 5)),
            ((-3,-3), (-3,-1)),
            ((-3,-3), (-1,-3)),
            (( 3, 3), ( 3, 1)),
            (( 3, 3), ( 1, 3)),
            (( 1,-1), ( 3,-1)),
            (( 3,-1), ( 3,-3)),
            ((-1, 1), (-3, 1)),
            ((-3, 1), (-3, 3)),
            ((-1,-1), ( 1,-3)),
            ((-1, 5), (-1, 2)),
            (( 0, 0), ( 1, 1))
        ])

        # try changing these!
        x_init = [-4,-4] # reset to [-4,-4] when saving results for submission
        x_goal = [4,4] # reset to [4,4] when saving results for submission
```

1.1 Geometric Planning

```
In [13]: grrt = GeometricRRT([-5,-5], [5,5], x_init, x_goal, MAZE)
         grrt.solve(1.0, 2000)
```

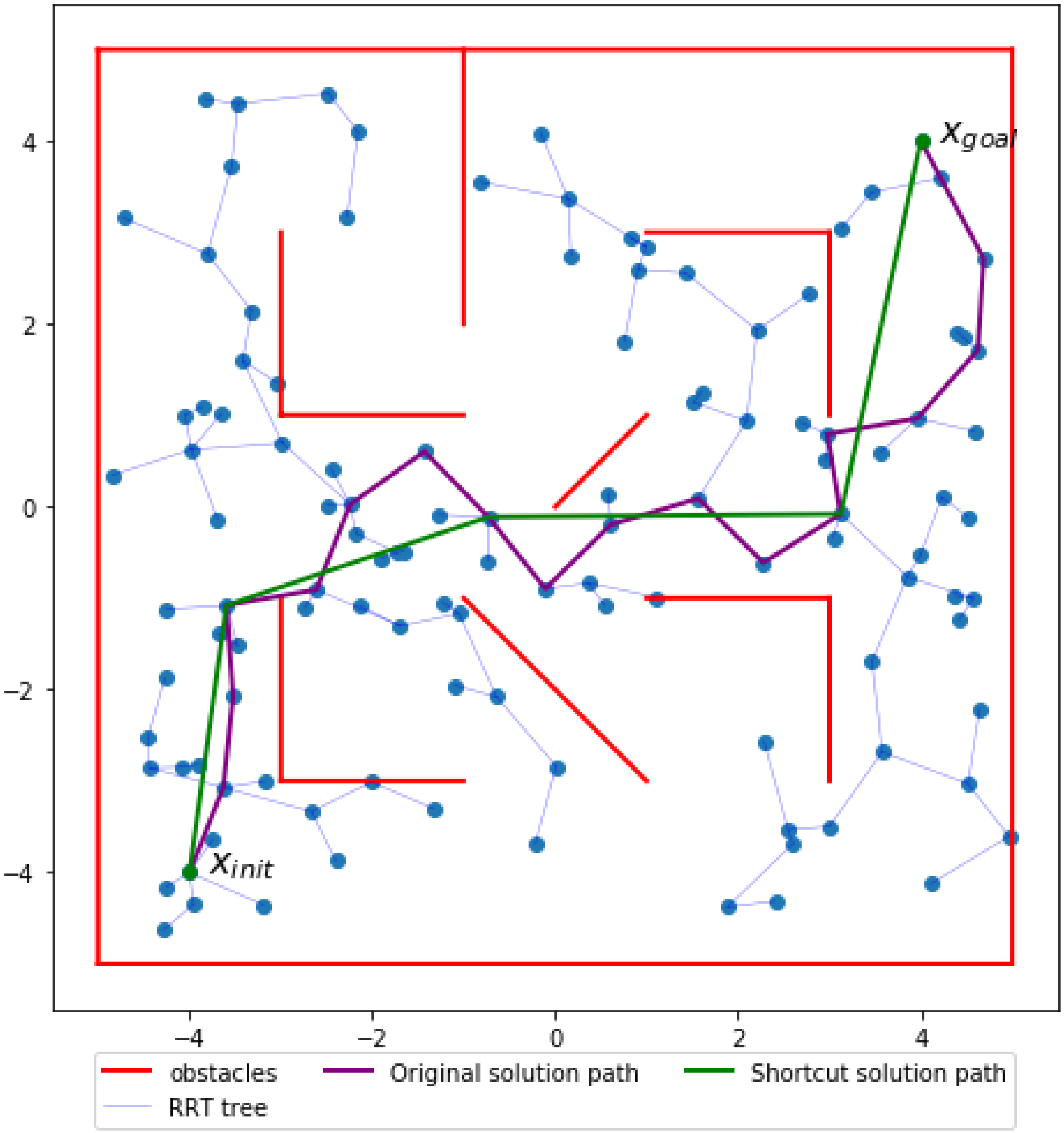
```
Out[13]: True
```



1.1.1 Adding shortcutting

```
In [19]: grrt.solve(1.0, 2000, shortcut=True)
```

```
Out[19]: True
```

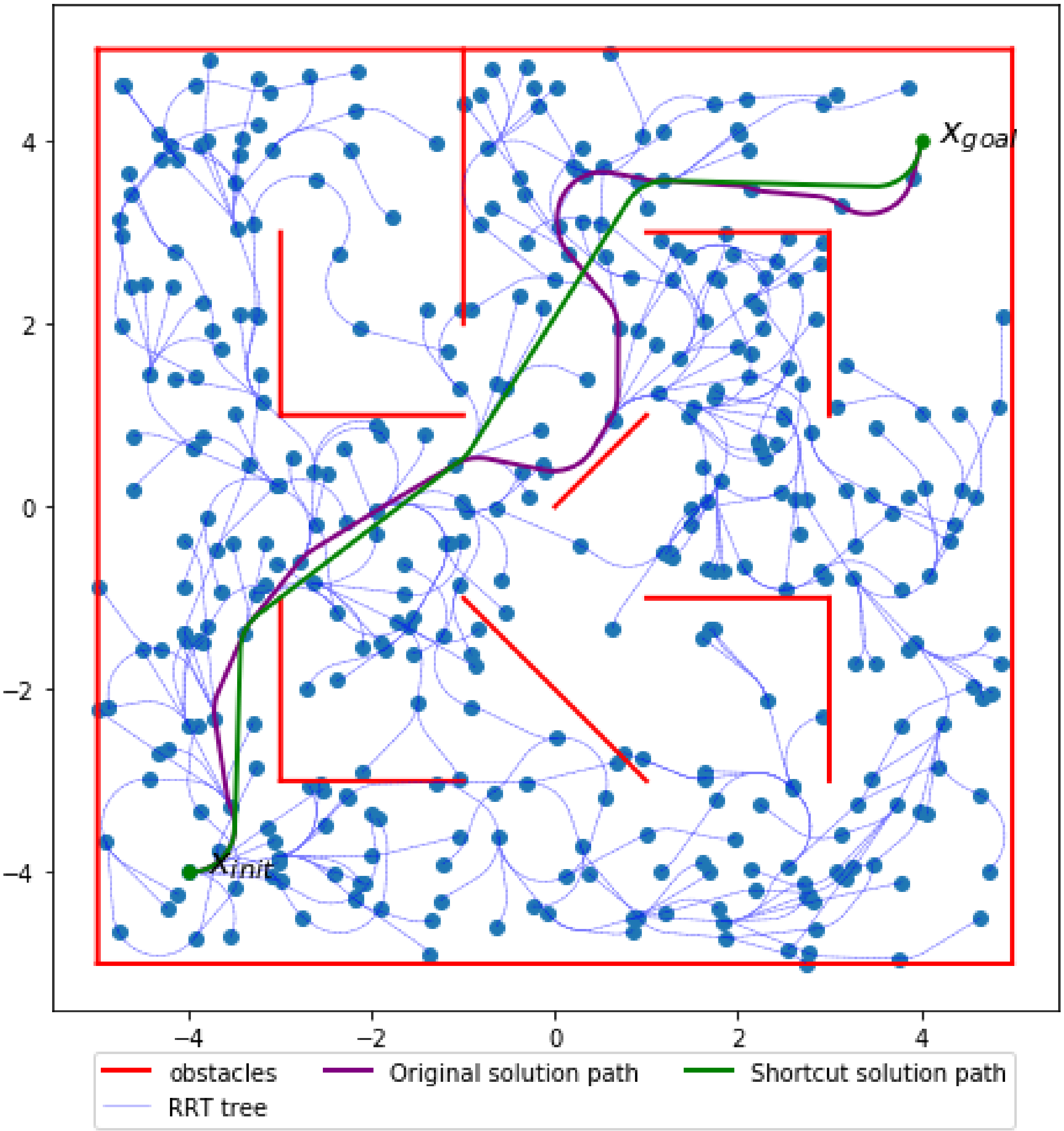


1.2 Dubins Car Planning

```
In [39]: x_init = [-4,-4,0]
         x_goal = [4,4,np.pi/2]

         # Occasionally, depending on the generated problem, the solution will fail
         # due to numerical precision invalidating the state equality comparison with the goal
         drrt = DubinsRRT([-5,-5,0], [5,5,2*np.pi], x_init, x_goal, MAZE, .5)
         drrt.solve(1.0, 1000, shortcut=True)
```

```
Out[39]: True
```



```
In [ ]:
```