MATLAB을 활용한 Path Planning Algorithm 성능 비교

3차 연구 발표회

정도현

Contents

The process so far

- **1차 연구 : 특정 색상의 손글씨 인식**

ㄴ CNN, 비전 처리 (파이썬)

- **2차 연구 : Path Planning에 활용되는 알고리즘 개발**

ㄴ 알고리즘 (파이썬)

Subject of the 3rd Project

- **3차 연구 : MATLAB을 활용한 Path Planning Algorithm 성능 비교**

ㄴ 알고리즘 (matlab)

< 선정 이유 >

- 지난 2차 연구에서 개발 언어와 시각화 방법의 차이로 인해 3가지 알고리즘에 대한 정확한 성능 평가에 대한 비교가 이루어지지 못해 아쉬움이 남았다.

< 목표 >

1. 같은 Map 환경에 대해 적용할 수 있는 RRTStar, AStar 알고리즘을 매트랩으로 구현
2. 각 알고리즘의 성능 비교 (경과 시간, 이동 거리)

Code of RRT Star Algorithm

```
% import example map
image = imread('testMap2.jpg');
image = imresize(image, [600, 600]);
grayimage = rgb2gray(image);
bwimage = grayimage < 0.5;
map = binaryOccupancyMap(bwimage, 1);
show(map)
```

Code of RRT Star Algorithm

```
% define robot dimensions and inflate the map
% to ensure that the robot not collide with any obstacles, you should
% inflate the map by the dimension of the robot before supplying to the PRM
% path planner
robotRadius = 0.05;
mapInflated = copy(map); % 원본 map 복사
inflate(mapInflated, robotRadius);
```

Code of RRT Star Algorithm

```
% RRT Star

ss = stateSpaceSE2; % state space

sv = validatorOccupancyMap(ss); % 2차원 그리드 맵 기반의 상태 유효성 검사(충돌 여부)

sv.Map = mapInflated;

sv.ValidationDistance = 0.1;

ss.StateBounds = [mapInflated.XWorldLimits; mapInflated.YWorldLimits; [-pi, pi]]; %ss의 상한/하한/각도

planner = plannerRRTStar(ss, sv, ContinueAfterGoalReached=true, MaxIterations=50000, MaxconnectionDistance=10);


% ContinueAfterGoalReached : 목표지점에 도달해도 계속 최적화할지 결정
```

Code of RRT Star Algorithm

```
% testMap1

start = [550, 2300, 0];

goal = [1900, 2650, 0];

% testMap2

start = [100, 550, 0];

goal = [600, 600, 0];

rng(10, 'twister') % 난수 생성기

tic

[pathObj, solnInfo] = plan(planner, start, goal); % 경로 계획. path와 solutionInfo 도출.
```

Code of RRT Star Algorithm

% rng 함수 : rand, randn, randi, randperm 함수가 난수열을 생성하는 방법을 결정

% https://kr.mathworks.com/help/matlab/ref/rng.html

값	설명
0	시드값 0을 사용하여 생성기를 초기화
양의 정수	지정된 양의 정수 시드값을 사용하여 생성기를 초기화
'default'	시드값 0을 사용하여 메르센 트위스터 생성기를 초기화(디폴트 설정)
'shuffle'	현재 시간을 기준으로 생성기를 초기화하여 rng를 호출할 때마다 다른 난수열 생성
구조체	Type, Seed 및 State 필드가 있는 구조체에 포함된 설정을 기반으로 생성기 초기화

Code of RRT Star Algorithm

```
% pathObj.States
```

편집기 - RRTStar.m

pathObj | pathObj.States

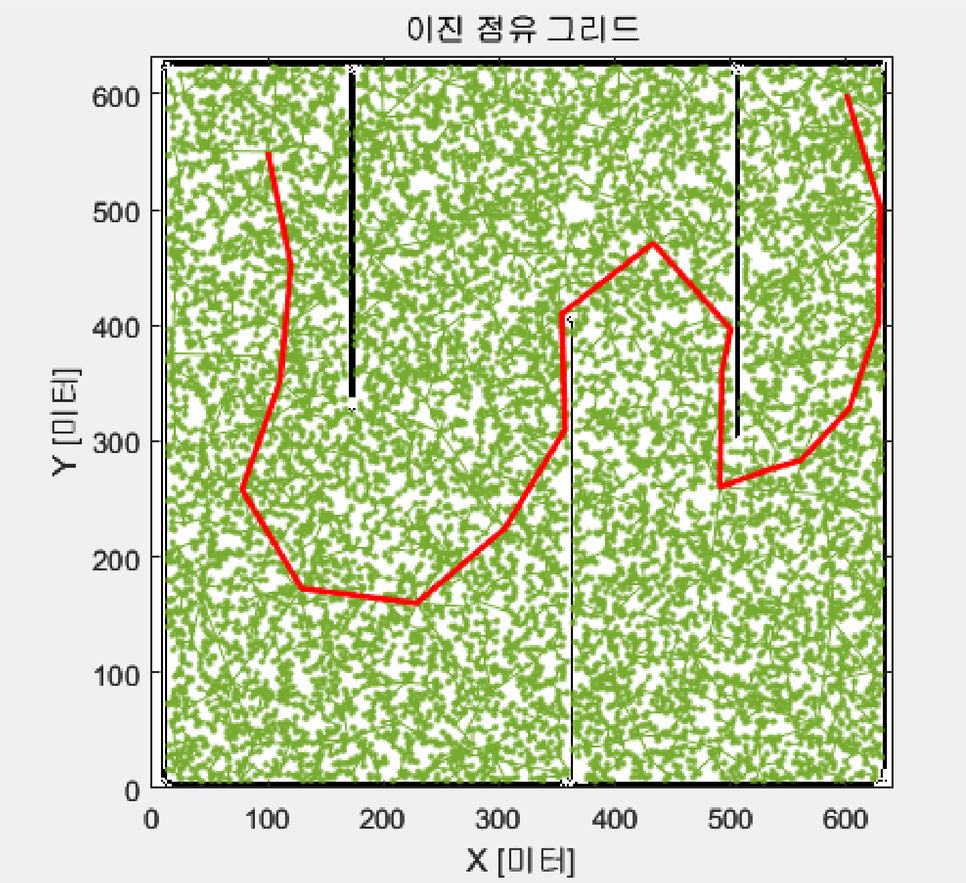
pathObj.States

	1	2	3
1	100	550	0
2	119.9762	452.0157	-0.5491
3	110.7484	352.4427	0.1944
4	77.6917	258.0648	1.0350
5	129.4067	172.4760	2.1201
6	228.6151	159.9220	2.9969
7	304.8503	224.6383	2.9176
8	356.9133	310.0154	-1.9675
9	353.9123	409.9686	-0.0950
10	433.0606	471.0877	-0.0644
11	499.9593	396.7605	0.5377
12	492.5948	360.0871	-0.2155
13	490.5119	260.1102	1.4508
14	533.4717	275.5487	2.6607
15	560.6333	283.3871	1.7873
16	602.4253	327.8054	-1.9197
17	627.8117	402.7818	-0.7848
18	629.1323	502.7727	0.0741
19	600.4300	598.5650	0.0011
20	600	600	0

Code of RRT Star Algorithm

```
% 결과 시각화

map.show

hold on % 좌표축에 plot이 새로 추가될 때, 기존 plot이 삭제되지 않도록 현재 좌표축을 plot을 유지.

% tree expansion

plot(solnInfo.TreeData(:,1), solnInfo.TreeData(:,2), '.-') % Tree의 x, y 좌표를 표시

% draw path

plot(pathObj.States(:,1), pathObj.States(:,2), 'r-', 'LineWidth', 2) % Path의 x, y 좌표를 표시

toc

% 이동 거리 계산

len = pathLength(pathObj);

disp("Path Length = " +num2str(len))
```

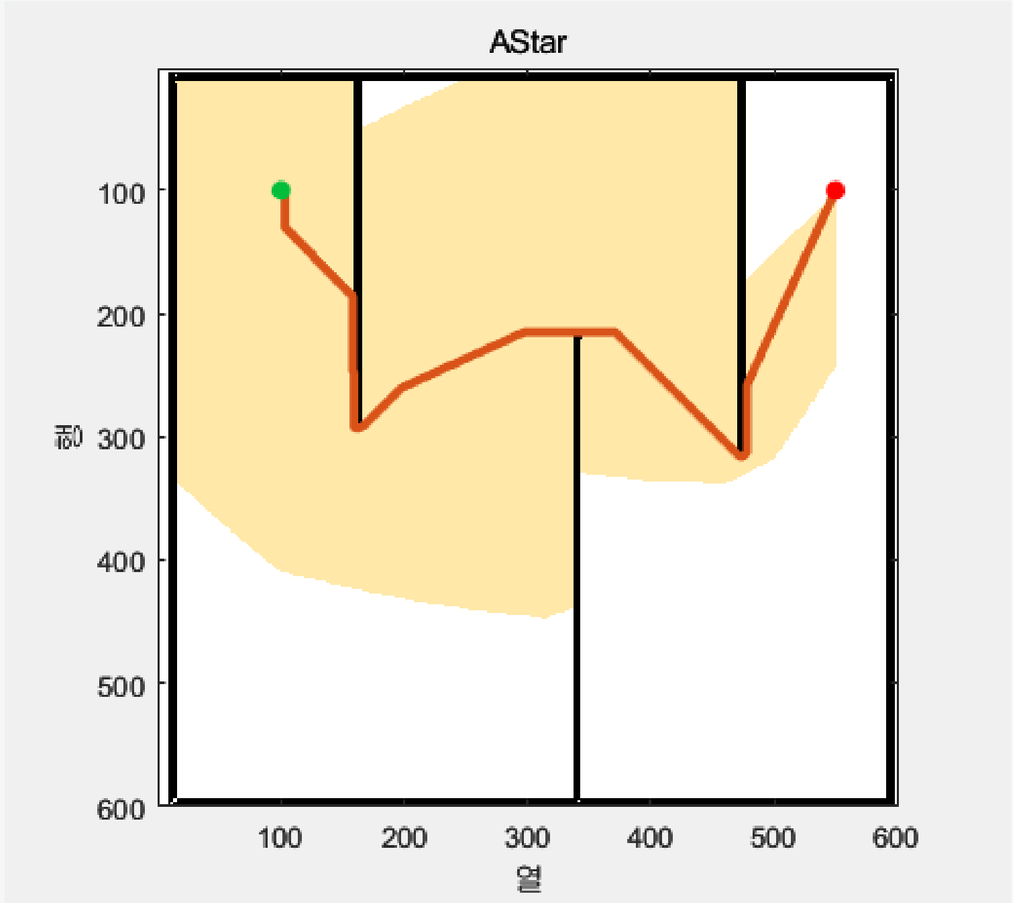
Code of RRT Star Algorithm



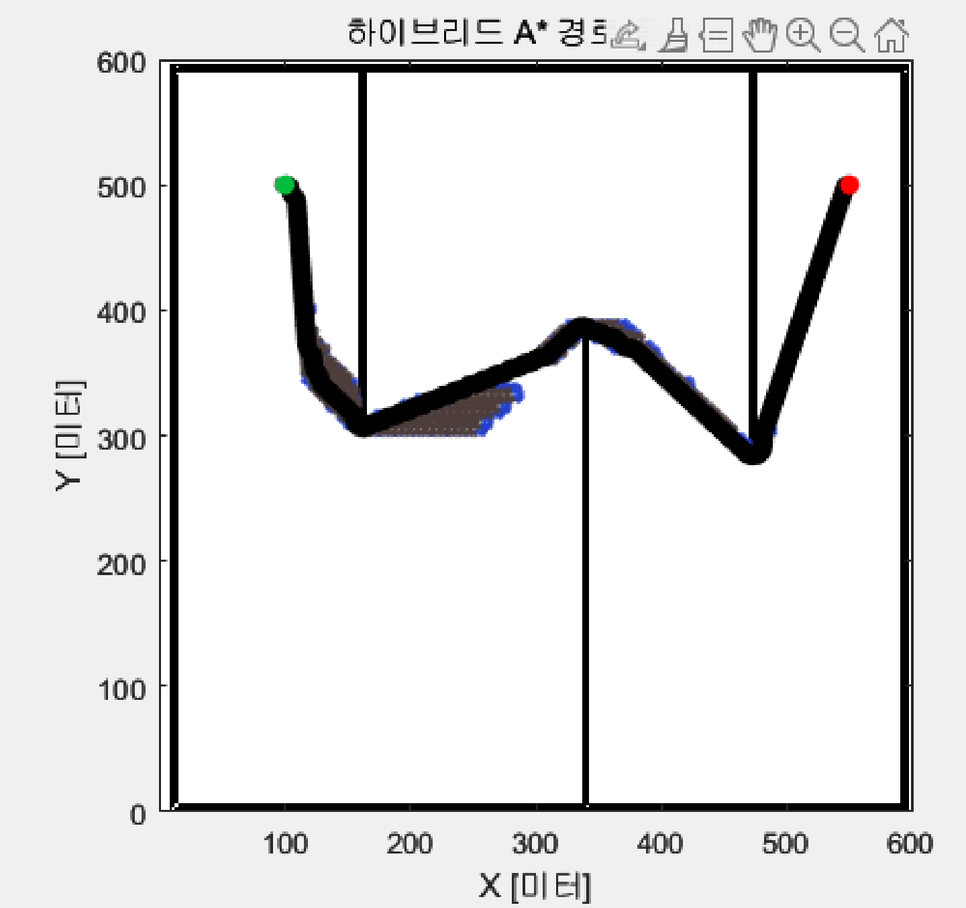
Code of A Star Algorithm

```
% A Star

planner = plannerAStarGrid(mapInflated, GCost = "Euclidean");


% GCost : 그리드에서 임의의 두 점 사이를 이동하는 일반적인 비용

% HCost : 그리드의 어느 한 지점부터 목표 지점까지 발견법에 근거하여 구한 비용

% 그 외에 우선 순위 결정, 대각선 탐색 등의 조건들을 설정할 수 있음
```

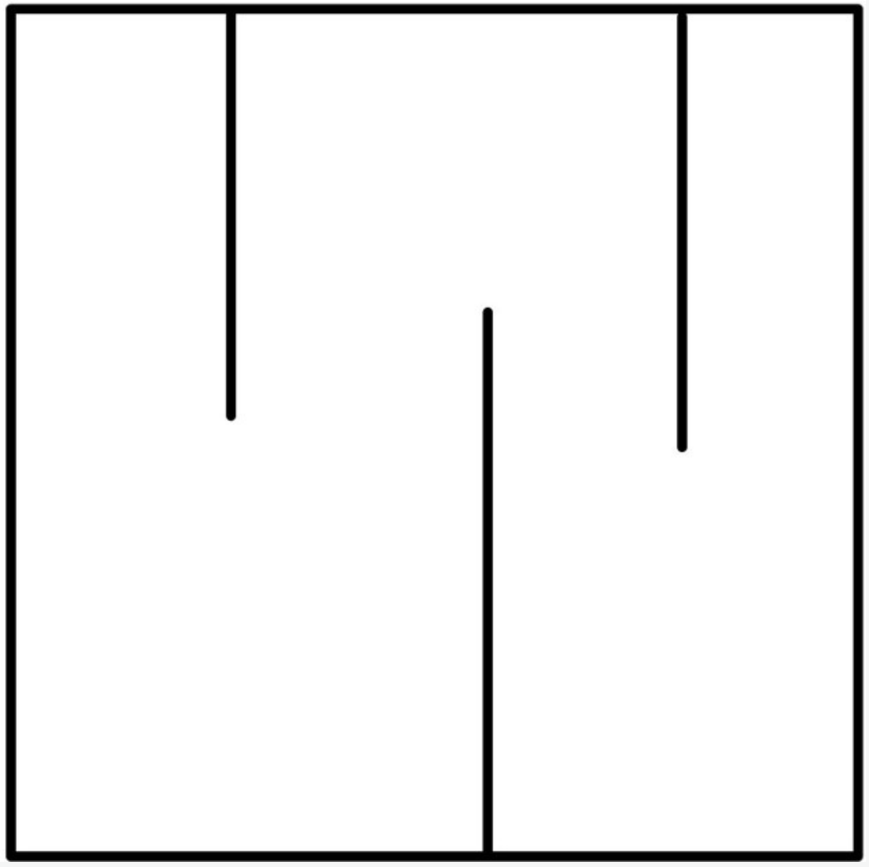
Code of A Star Algorithm



Code of A Star Algorithm



Assess performance over elapsed time & distance traveled



```
>> RRTStar
경과 시간은 4.621184초입니다.
Path Length = 1080.0828
>> AStar
경과 시간은 0.499841초입니다.
>> HybridAStar
경과 시간은 5.018758초입니다.
Path Length = 821.3983
```

Subject for 사미용두

Part 1. Mapping

-다양한 센서를 활용한 mapping

Part 2. Path Planning

-다양한 알고리즘을 활용한 path planning

Part 3. Robotic Motion Planning & Manipulating

-경로를 기반으로 한 자율 주행

Part 4. Simulating

-시뮬레이터를 활용한 시험 주행

MATLAB을 활용한 Path Planning Algorithm 성능 비교

-끝-

정도현