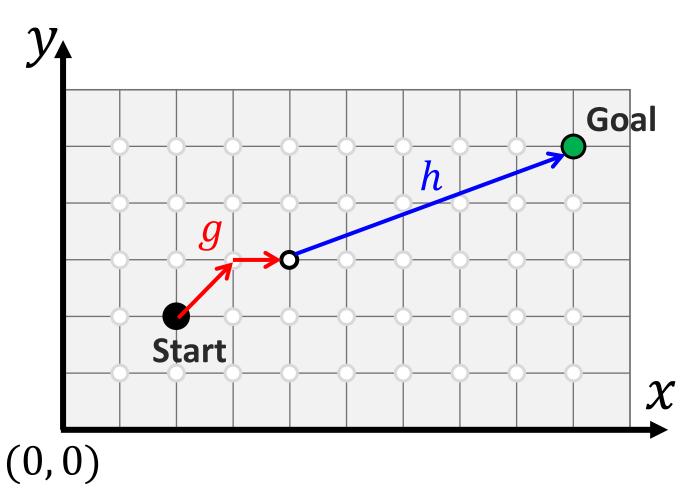


Example on A* Path Planning Algorithm

A* Algorithm





Definition of cost:

 $g(x,y) - \underline{exact\ cost}$ of the path from the **Start** node to node (x,y) $h(x,y) - \underline{heuristic\ estimated\ cost}$ from node (x,y) to the **Goal** node f(x,y) = g(x,y) + h(x,y) total cost of a node (x,y)

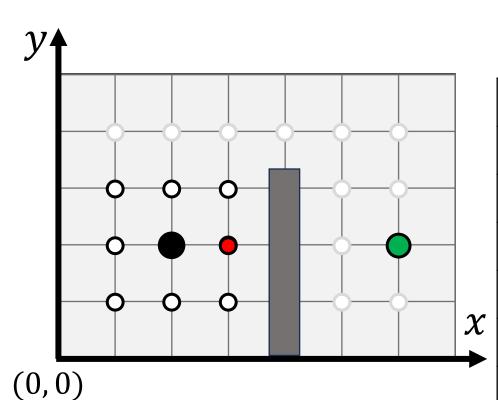
Searching procedure:

- 1) Calculate f = g + h for nodes nearby current node (record costs and source node in open list);
- 2) Move current node to the node with lowest *f* (record in close list);
- 3) Current node is **Goal**, retrieve path from **Goal** to **Start**





A* Algorithm (round 1)



Open List (searched nodes)

Node	g	h	f	Source
(1,1)	1.4	5.1	6.5	(2,2)
(1,2)	1	5	6	(2,2)
(1,3)	1.4	5.1	6.5	(2,2)
(2,1)	1	4.1	5.1	(2,2)
(2,3)	1	4.1	5.1	(2,2)
(3,1)	1.4	3.1	4.5	(2,2)
(3,2)	1	3	4	(2,2)
(3,3)	1.4	3.1	4.5	(2,2)

Close List (arrived nodes)

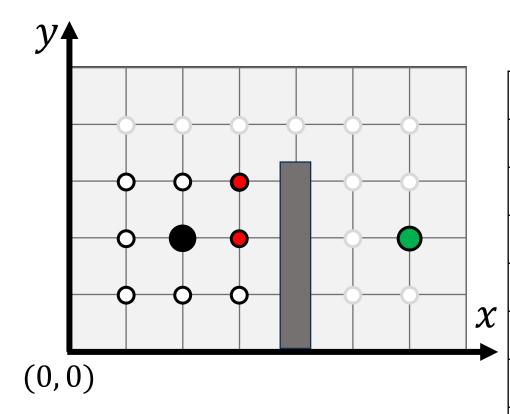
Node	f	Source
Start	ı	1
(3,2)	4	(2,2)
(Curi	rent i	node)

- Start
- Goal





A* Algorithm (round 2)



Open List (searched nodes)

Node	g	h	f	Source
(1,1)	1.4	5.1	6.5	(2,2)
(1,2)	1	5	6	(2,2)
(1,3)	1.4	5.1	6.5	(2,2)
(2,1)	1	4.1	5.1	(2,2)
(2,3)	1	4.1	5.1	(2,2)
(3,1)	1.4	3.1	4.5	(2,2)
(3,2)	1	3	4	(2,2)
(3,3)	1.4	3.1	4.5	(2,2)

Close List (arrived nodes)

Node	f	Source
Start	ı	-
(3,2)	4	(2,2)

Start

Goal





A* Algorithm (round 3-5)

Open List (searched nodes)

Node Source gh (2,2)(1,1)6.5 1.4 5.1 (1,2)5 6 (2,2)(1,3)(2,2)6.5 1.4 5.1 (2,1)(2,2)5.1 4.1 (2,3)(2,2)4.1 5.1 (3,1)4.5 (2,2)1.4 3.1 (2,4)2.8 4.5 7.3 (3,3)(3,4)(3,3)2.4 3.6 6.0 (3,3)(4,4)2.8 2.8 5.6

Close List (arrived nodes)

Node	f	Source
Start	1	-
(3,2)	4	(2,2)
(3,3)	4.5	(2,2)

<i>y</i>					
)——
		—)—
	-		•		
		—	-		
(0,	0)				

Sta	rt

Goal





A* Algorithm (round 6)

Open List (searched nodes)

		Jeur	SIICU	nouc.)
	Node	g	h	f	Source
	(1,1)	1.4	5.1	6.5	(2,2)
	(1,2)	1	5	6	(2,2)
	(1,3)	1.4	5.1	6.5	(2,2)
-	(2,4)	2.8	4.5	7.3	(3,3)
x	(3,4)	2.4	3.6	6.0	(3,3)
\	(4,4)	2.8	2.8	5.6	(3,3)
\ \	(3,4)	2.4	3.6	6.0	(2,3)
	(2,4)	2	4.5	6.5	(2,3)
	(1,4)	2.4	5.4	7.8	(2,3)
	x	Node (1,1) (1,2) (1,3) (2,4) (3,4) (4,4) (3,4) (2,4)	Node g (1,1) 1.4 (1,2) 1 (1,3) 1.4 (2,4) 2.8 (3,4) 2.4 (4,4) 2.8 (3,4) 2.4 (2,4) 2	Node g h (1,1) 1.4 5.1 (1,2) 1 5 (1,3) 1.4 5.1 (2,4) 2.8 4.5 (3,4) 2.4 3.6 (4,4) 2.8 2.8 (3,4) 2.4 3.6 (2,4) 2 4.5	(1,1) 1.4 5.1 6.5 (1,2) 1 5 6 (1,3) 1.4 5.1 6.5 (2,4) 2.8 4.5 7.3 x (3,4) 2.4 3.6 6.0 (4,4) 2.8 2.8 5.6 (3,4) 2.4 3.6 6.0 (2,4) 2 4.5 6.5

Close List (arrived nodes)

Node	f	Source
Start	ı	1
(3,2)	4	(2,2)
(3,3)	4.5	(2,2)
(2,1)	5.1	(2,2)
(2,3)	5.1	(2,2)
(3,1)	4.5	(2,2)





A* Algorithm (round 7)

Open List (searched nodes)

(searchea noaes)					
Node	g	h	f	Source	
(1,1)	1.4	5.1	6.5	(2,2)	
(1,2)	1	5	6	(2,2)	
(1,3)	1.4	5.1	6.5	(2,2)	
(2,4)	2	4.5	6.5	(2,3)	
(3,4)	2.4	3.6	6.0	(3,3)	
(1,4)	2.4	5.4	7.8	(2,3)	
(5,4)	3.8	2.2	6.0	(4,4)	
(5,3)	4.2	1.4	5.6	(4,4)	

Close List (arrived nodes)

Node	f	Source
Start	1	-
(3,2)	4	(2,2)
(3,3)	4.5	(2,2)
(2,1)	5.1	(2,2)
(2,3)	5.1	(2,2)
(3,1)	4.5	(2,2)
(4,4)	5.6	(3,3)

-				
-		•	\Diamond	
-	•	•	 	

Start

Goal





A* Algorithm (round 8)

Open List (searched nodes)

Node	g	h	f	Source
(1,1)	1.4	5.1	6.5	(2,2)
(1,2)	1	5	6	(2,2)
(1,3)	1.4	5.1	6.5	(2,2)
(2,4)	2	4.5	6.5	(2,3)
(3,4)	2.4	3.6	6.0	(3,3)
(1,4)	2.4	5.4	7.8	(2,3)
(5,4)	3.8	2.2	6.0	(4,4)
•••				
(6,2)	5.6	0	5.6	(5,3)

Close List (arrived nodes)

Node	f	Source
Start	I	-
(3,2)	4	(2,2)
(3,3)	4.5	(2,2)
(2,1)	5.1	(2,2)
(2,3)	5.1	(2,2)
(3,1)	4.5	(2,2)
(4,4)	5.6	(3,3)
(5,3)	5.6	(4,4)

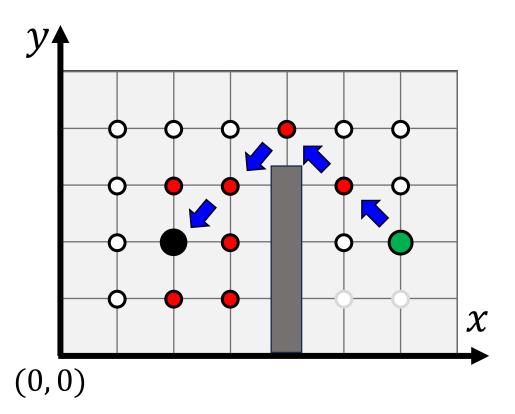
y	•						
	— Ö				•	-	
			•		-		
	0	•	•	\blacksquare	_	_0_	
(0, (0)						

- Start
- Goal





A* Algorithm (round 8)



- **Start**
- Goal

Cost: f = g + h



Node	f	Source
Start	ı	-
(3,2)	4	(2,2)
(3,3)	4.5	(2,2)
(2,1)	5.1	(2,2)
(2,3)	5.1	(2,2)
(3,1)	4.5	(2,2)
(4,4)	5.6	(3,3)
(5,3)	5.6	(4,4)
(6,2)	5.6	(5,3)

Reversely trace back trajectory

Goal