AAE1001 Introduction to Artificial Intelligence and Data Analytics in Aerospace and Aviation Engineering

Week 10 (Introduction to Path Planning)

Dr **Guohao Zhang**, and Dr Lingxiao Wu
Assisted by
Mr Feng HUANG (Darren), Mr Penghui Xu
Mr Zekun Zhang and Miss Hongmin Zhang

Introduction to A* Path Planning Algorithm

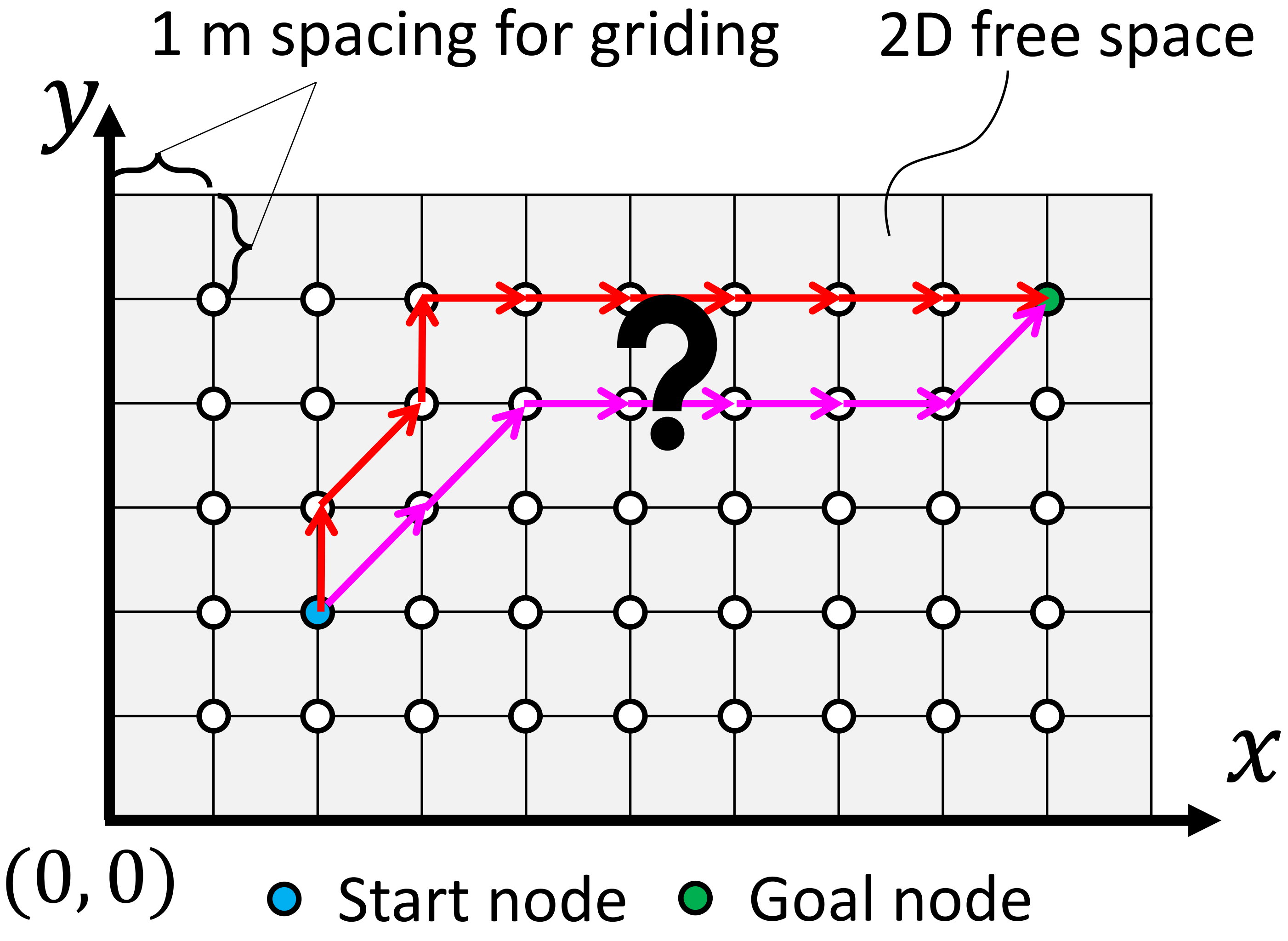
Daily Path Planning

MTR stations – available **node** to go

Goal node

Start node

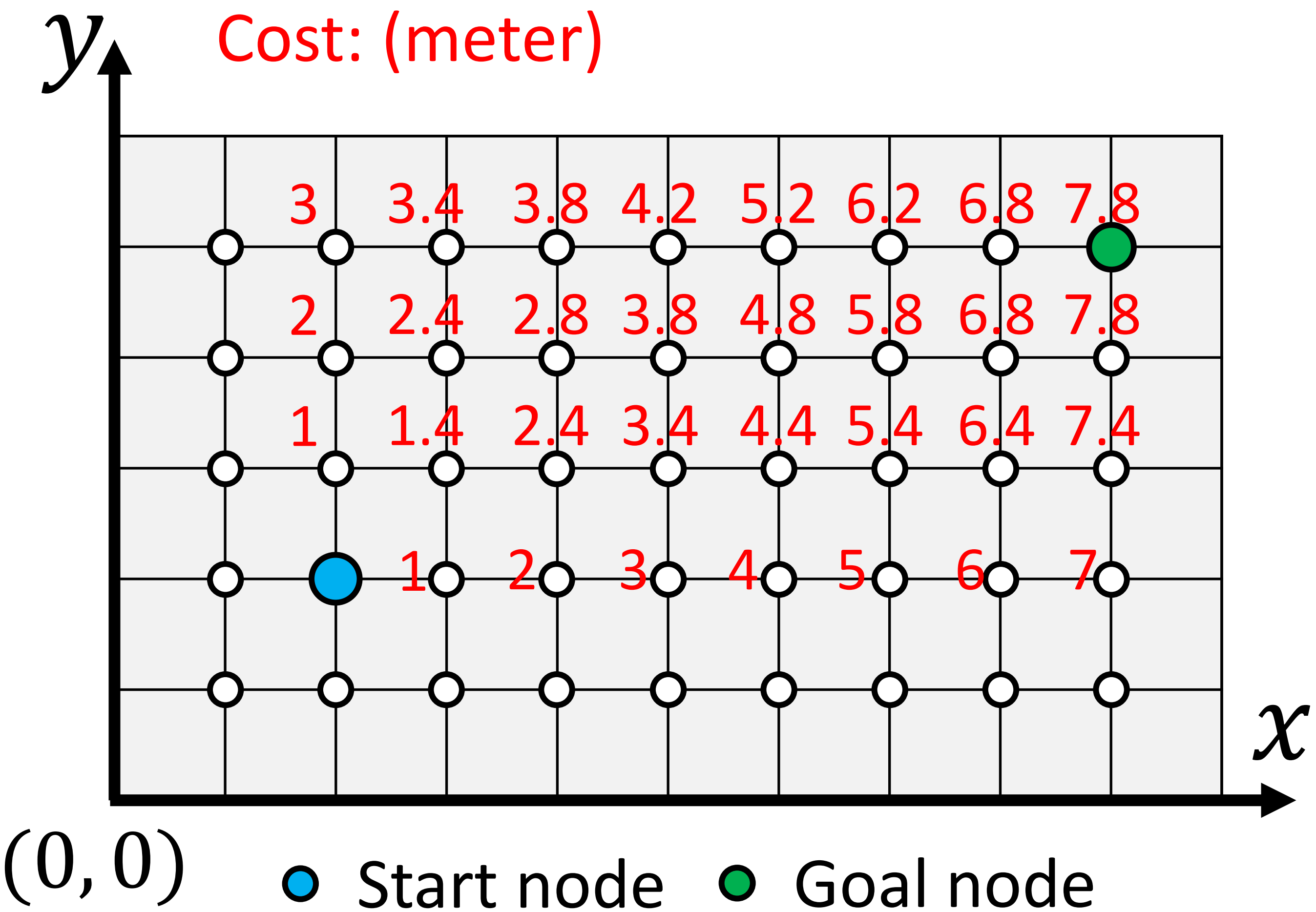
Definition of Path Planning



- **Node** — All potential position you can go across with a unique position (x, y)
- **Search space** — A collection of nodes, like all board positions of a board game.
- **Objective of path planning**— Find the shortest routes with smallest cost from start node to goal node.

Which one is better?

Path Planning by Checking All Available Nodes



Start node   Goal node

1 m spacing for griding

Test each possible nodes one-by-one from Start node

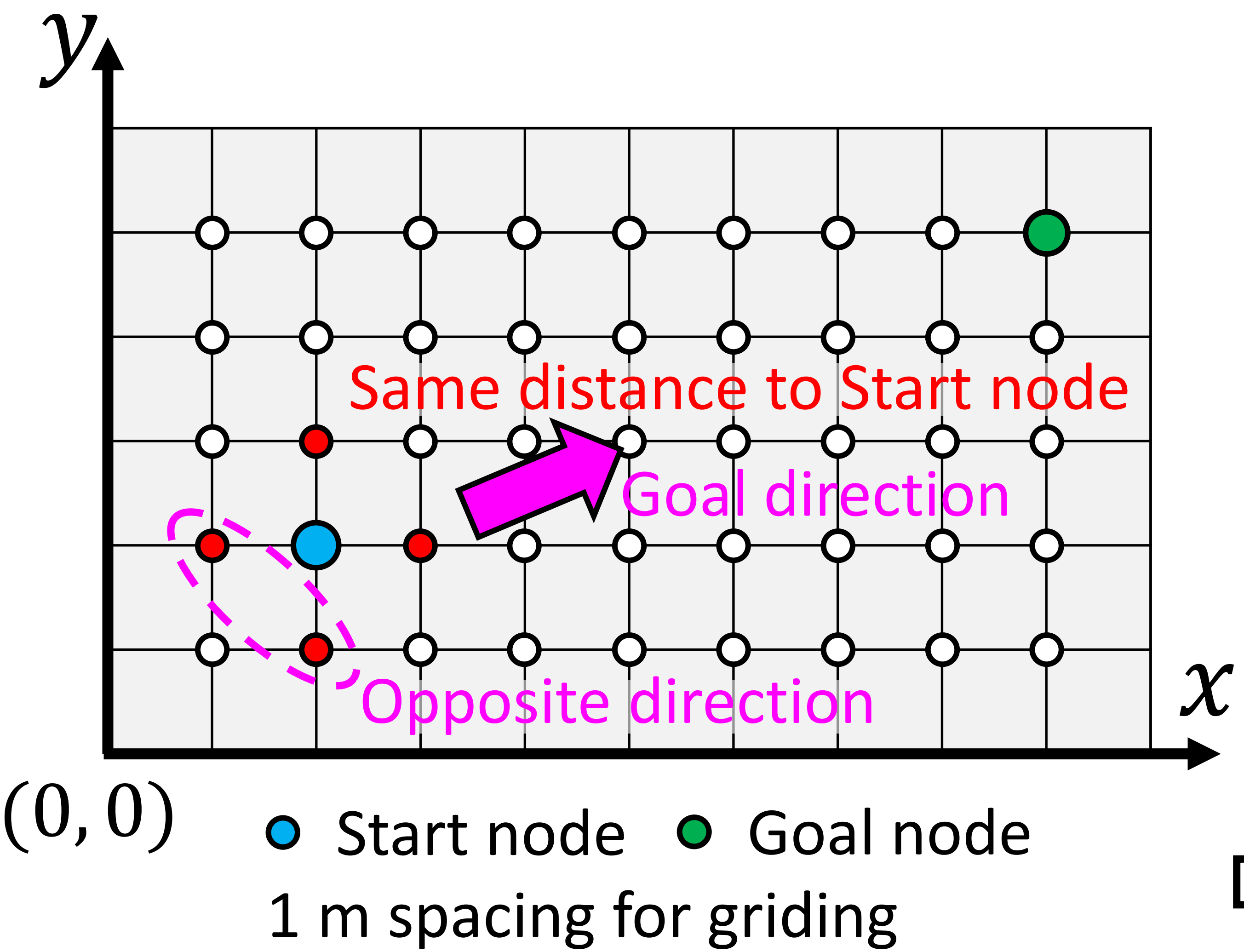
Record its shortest path between Start node

Reach Goal

Retrieve the shortest path (Dijkstra's algorithm)

Need higher efficiency!

Path Planning by Checking **NOT** All Available Nodes



Test each possible nodes one-by-one from Start node

Record its shortest path
between Start node

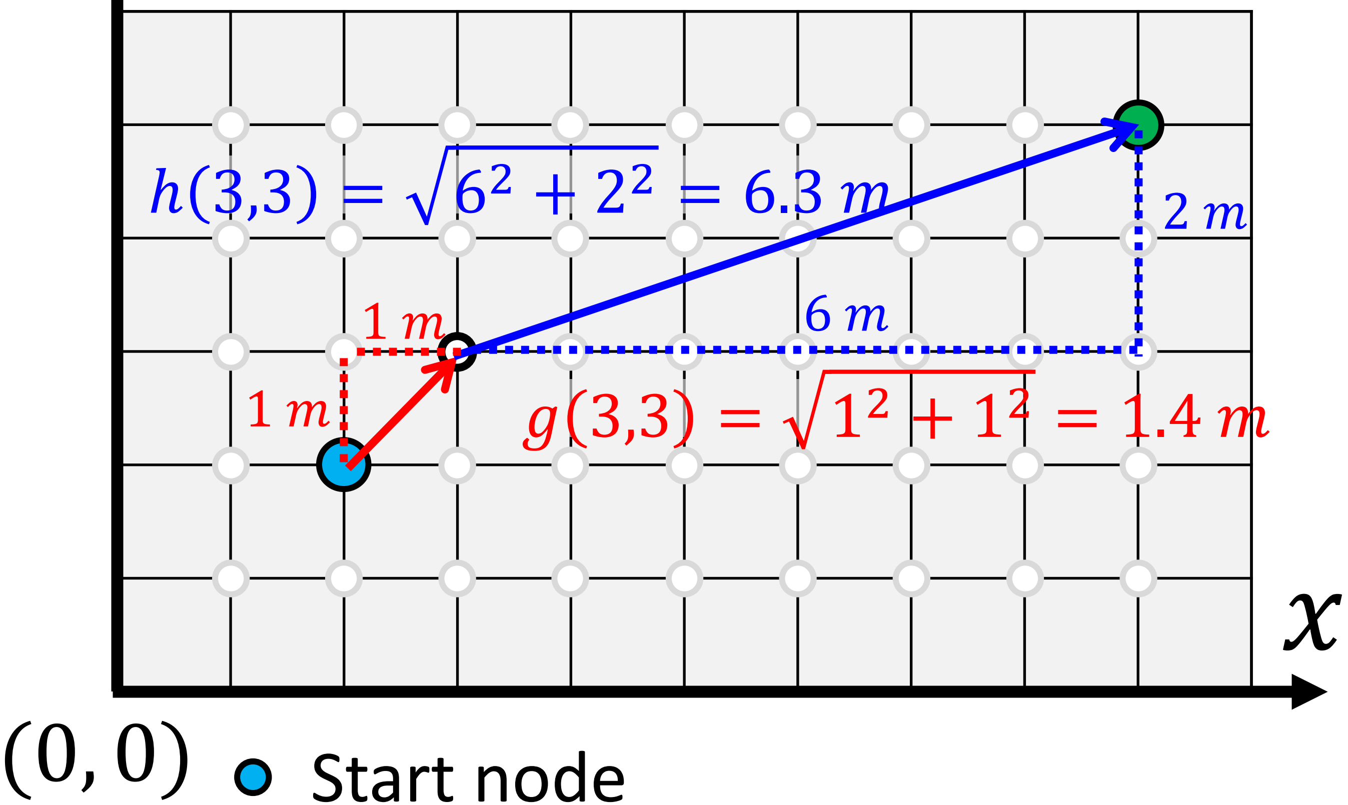
We also know a rough direction to Goal!

⇨ **A-star Search Algorithm**

Path Planning by A-star (A*) Search Algorithm

Total cost at node (3,3)

$$f(3,3) = g(3,3) + h(3,3) = 7.7\ m$$



- Start node
- Goal node

1 m spacing for griding

Definition of cost:

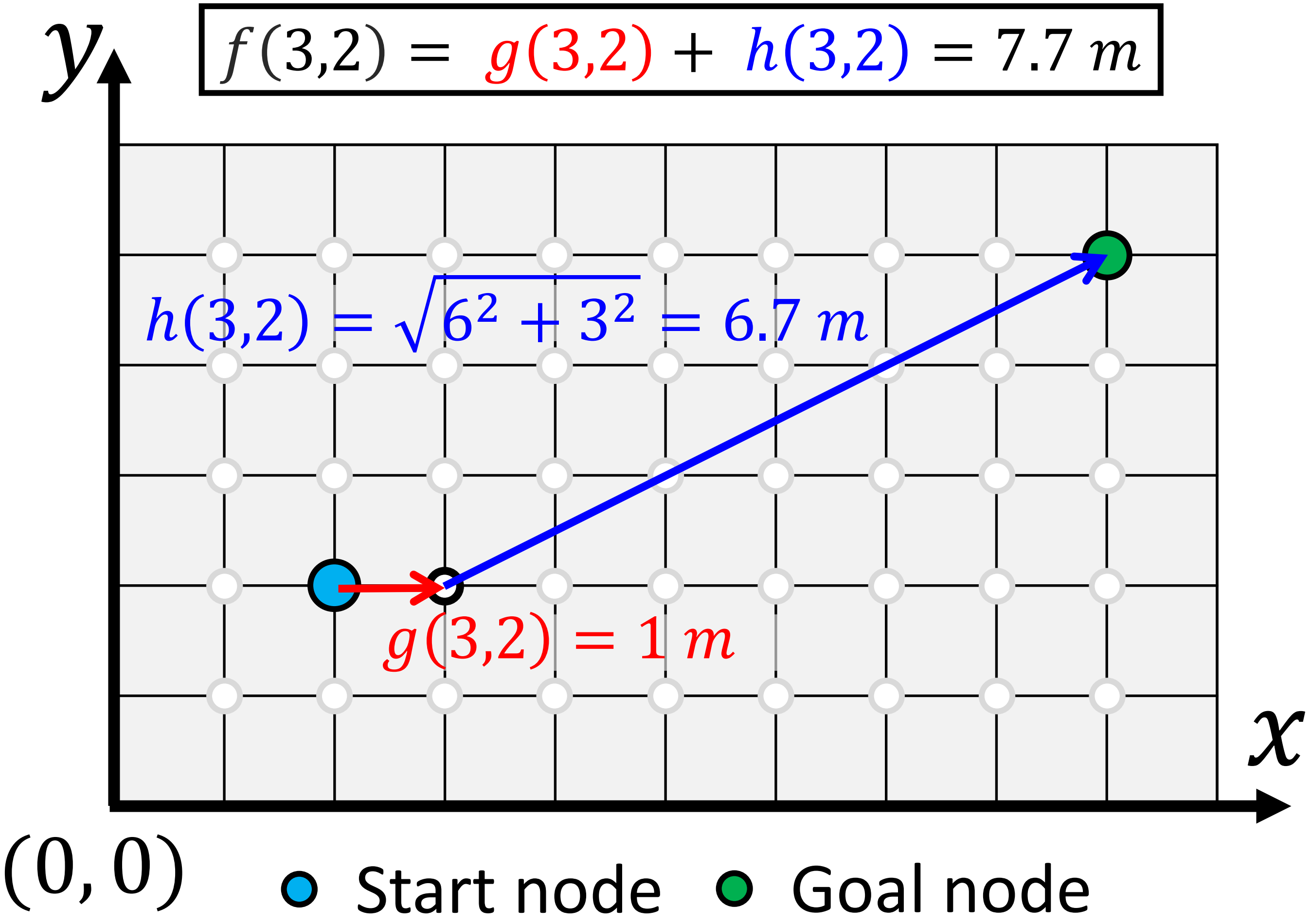
$g(x, y)$ — this represents the ***exact cost*** of the path **from** the **Start** node to node (x, y)

$h(x, y)$ — this represents the **heuristic** ***estimated cost*** from node (x, y) **to** the **Goal** node

$f(x, y) = g(x, y) + h(x, y)$ — **total** cost of a neighboring node (x, y)

A-star (A*) Path Planning – Cost at Node (3,2)

$$f(3,2) = g(3,2) + h(3,2) = 7.7\ m$$



1 m spacing for griding

Definition of cost:

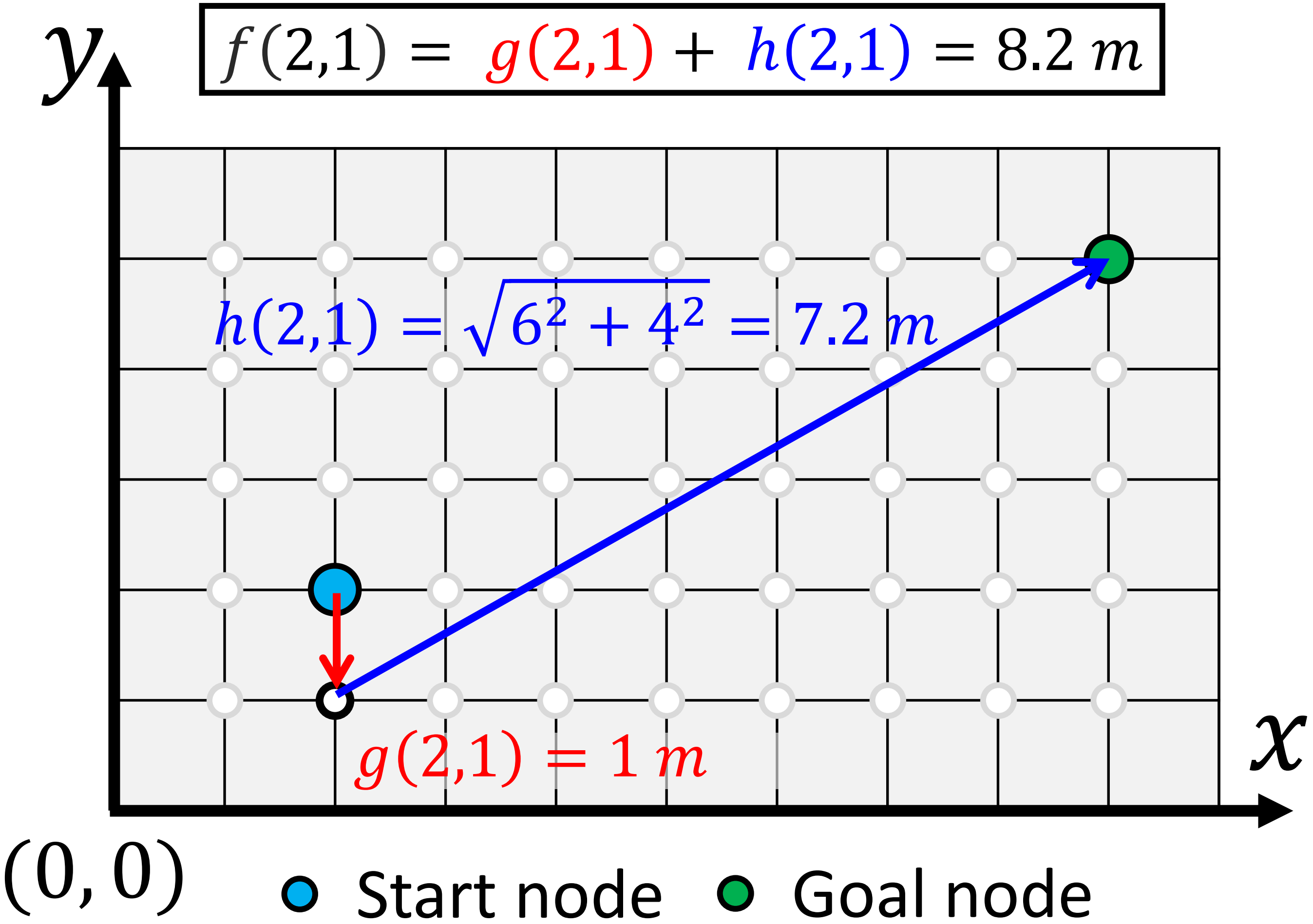
$g(x,y)$ — this represents the ***exact cost*** of the path **from** the **Start** node to node (x,y)

$h(x,y)$ — this represents the heuristic ***estimated cost*** from node (x,y) **to** the **Goal** node

$f(x,y) = g(x,y) + h(x,y)$ — **total** cost of a neighboring node (x,y)

A-star (A*) Path Planning – Cost at Node (2,1)

$f(2,1) = g(2,1) + h(2,1) = 8.2\ m$



1 m spacing for griding

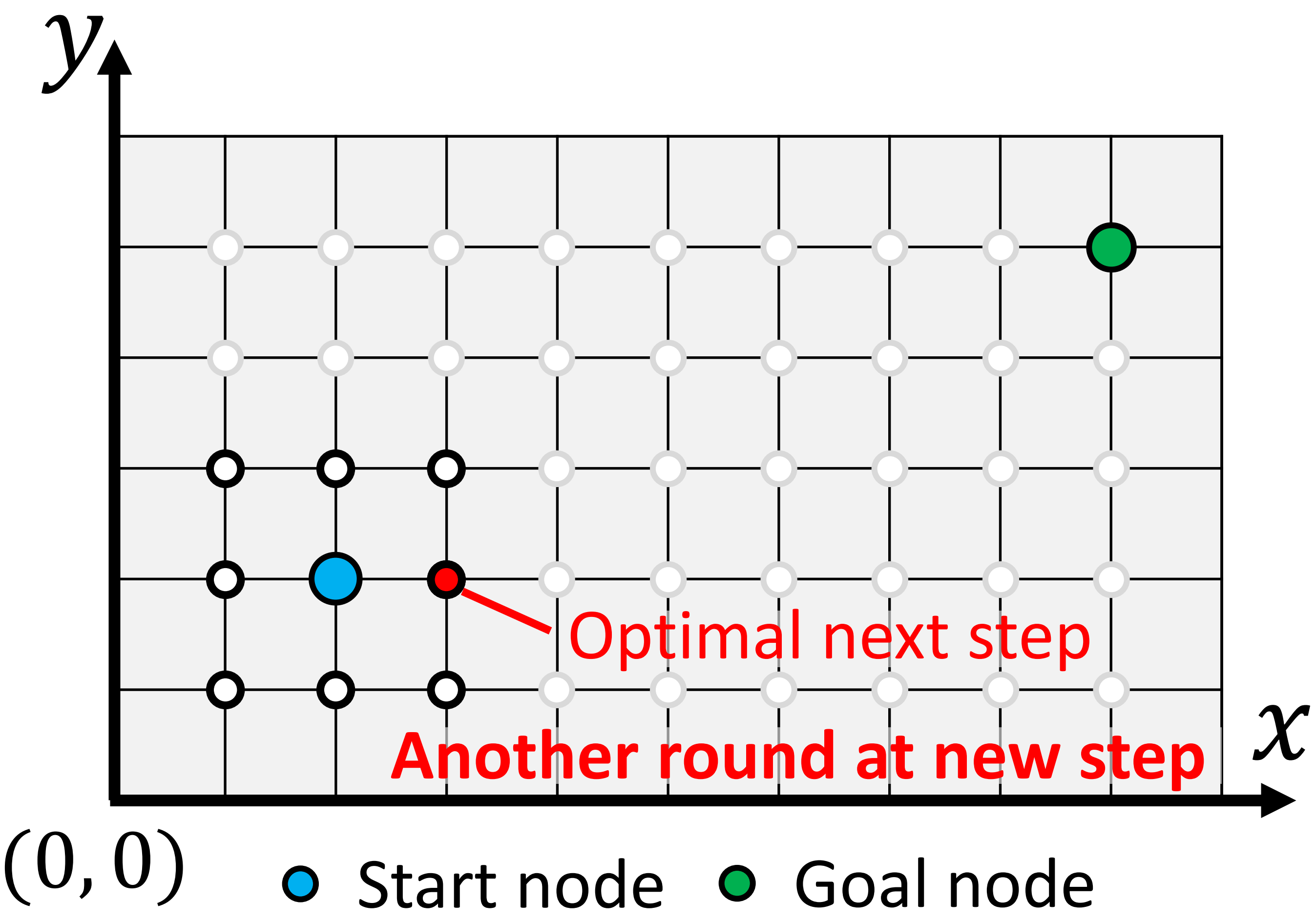
Definition of cost:

$g(x, y)$ — this represents the ***exact cost*** of the path **from** the **Start** node to node (x, y)

$h(x, y)$ — this represents the heuristic ***estimated cost*** from node (x, y) **to** the **Goal** node

$f(x, y) = g(x, y) + h(x, y)$ — **total** cost of a neighboring node (x, y)

Total Costs at Neighbouring Nodes

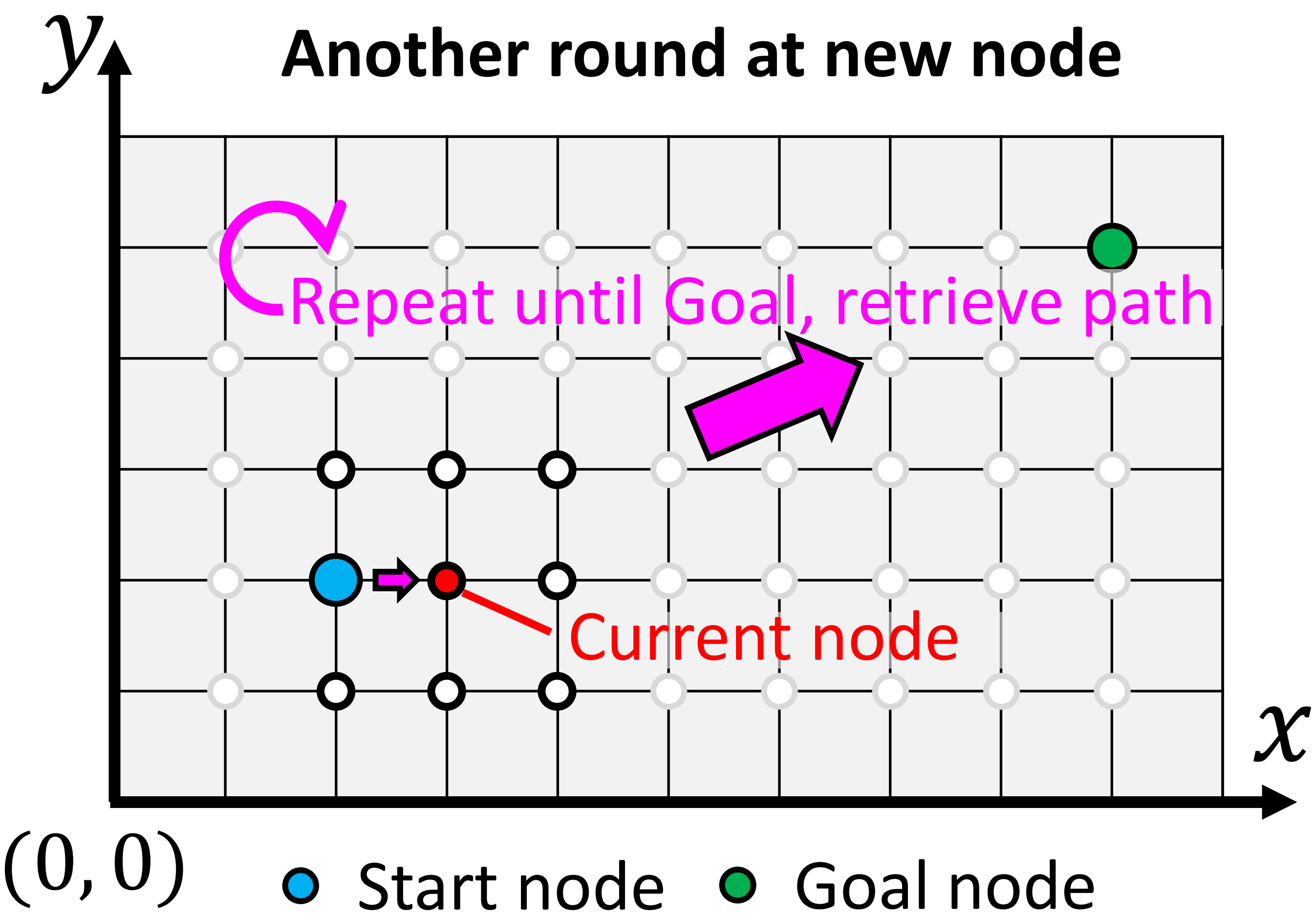


Exact cost from Start *Estimated cost to Goal* *Total cost*

Node ID	$g(x,y)$	$h(x,y)$	$f(x,y)$
(1,1)	1.4	8.9	10.3
(1,2)	1	8.5	9.5
(1,3)	1.4	8.2	9.6
(2,1)	1	8.1	9.1
(2,3)	1	7.3	8.3
(3,1)	1.4	7.2	8.6
(3,2)	1	6.7	7.7
(3,3)	1.4	6.3	7.7

Total Costs at Neighbouring Nodes – Round 2

Another round at new node

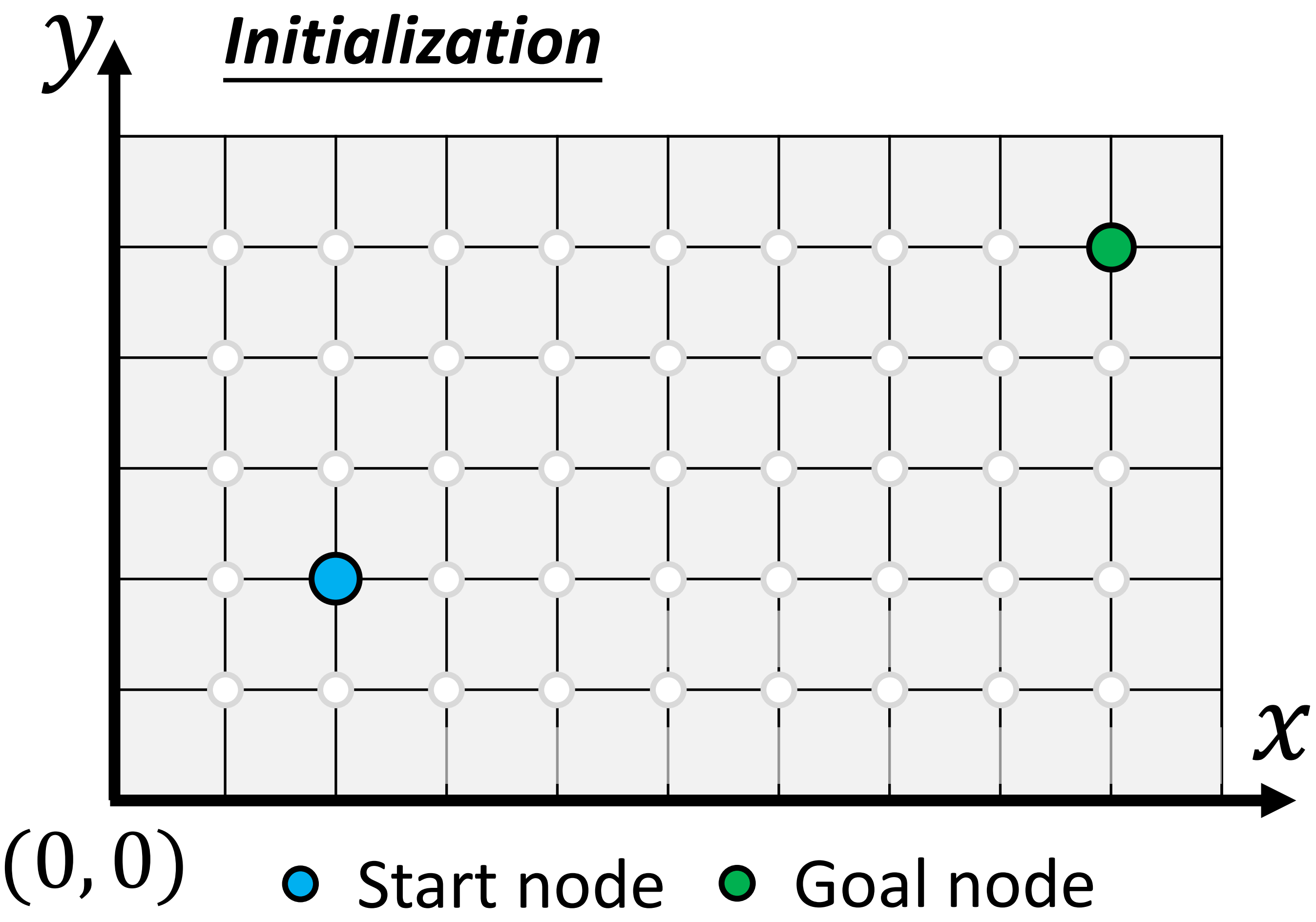


Exact cost from Start — *Estimated cost to Goal* — *Total cost*

Node ID	$g(x,y)$	$h(x,y)$	$f(x,y)$
(2,1)			
(2,2)			
(2,3)			
(3,1)			
(3,3)			
(4,1)			
(4,2)			
(4,3)			

How does computer search step by step and got path?

A-star (A*) Path Planning – Record Path



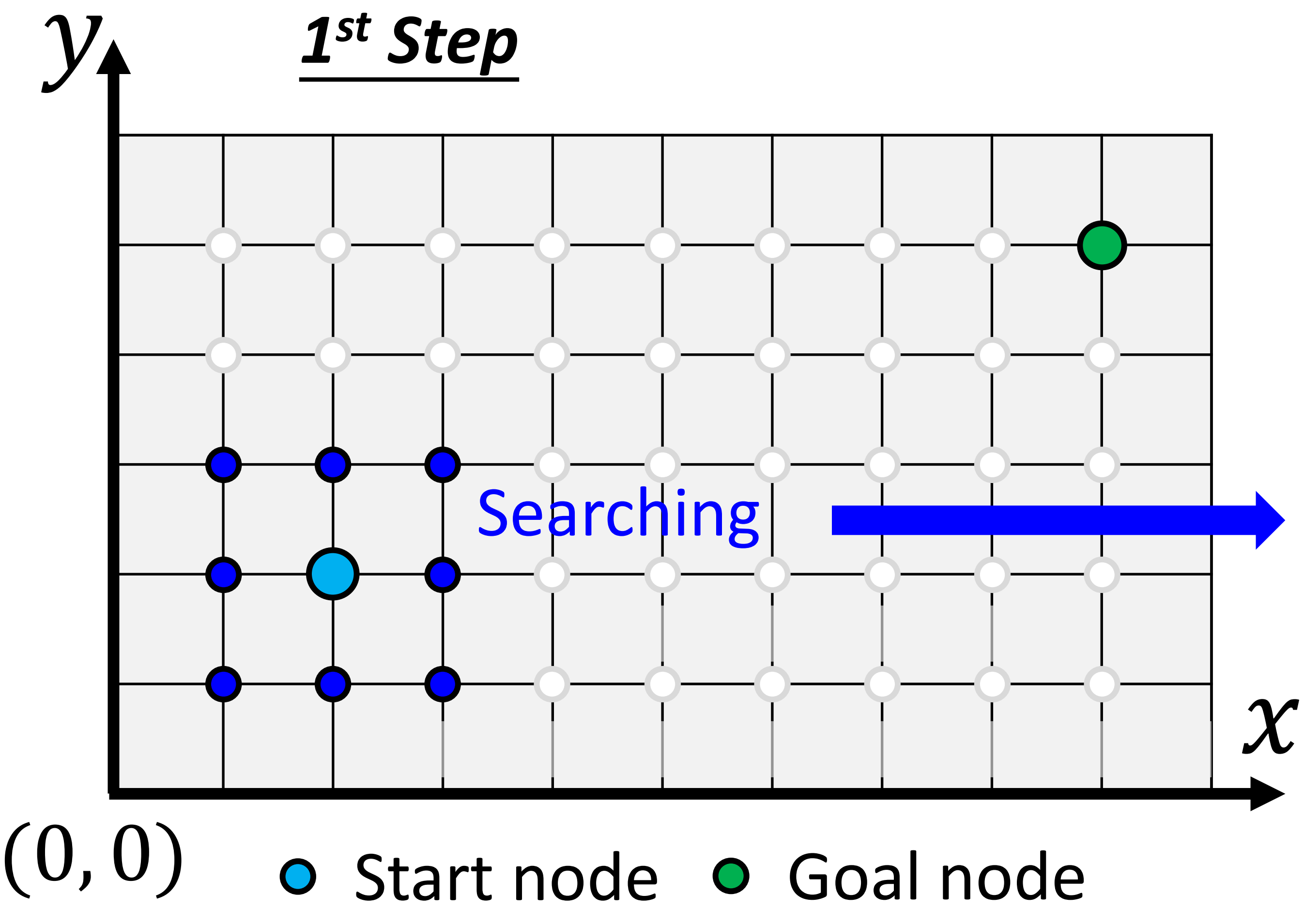
Open List
(searched nodes)

Node	f	Source

Close List
(arrived nodes)

Node	f	Source
Start	-	-

A-star (A*) Path Planning – Record Path



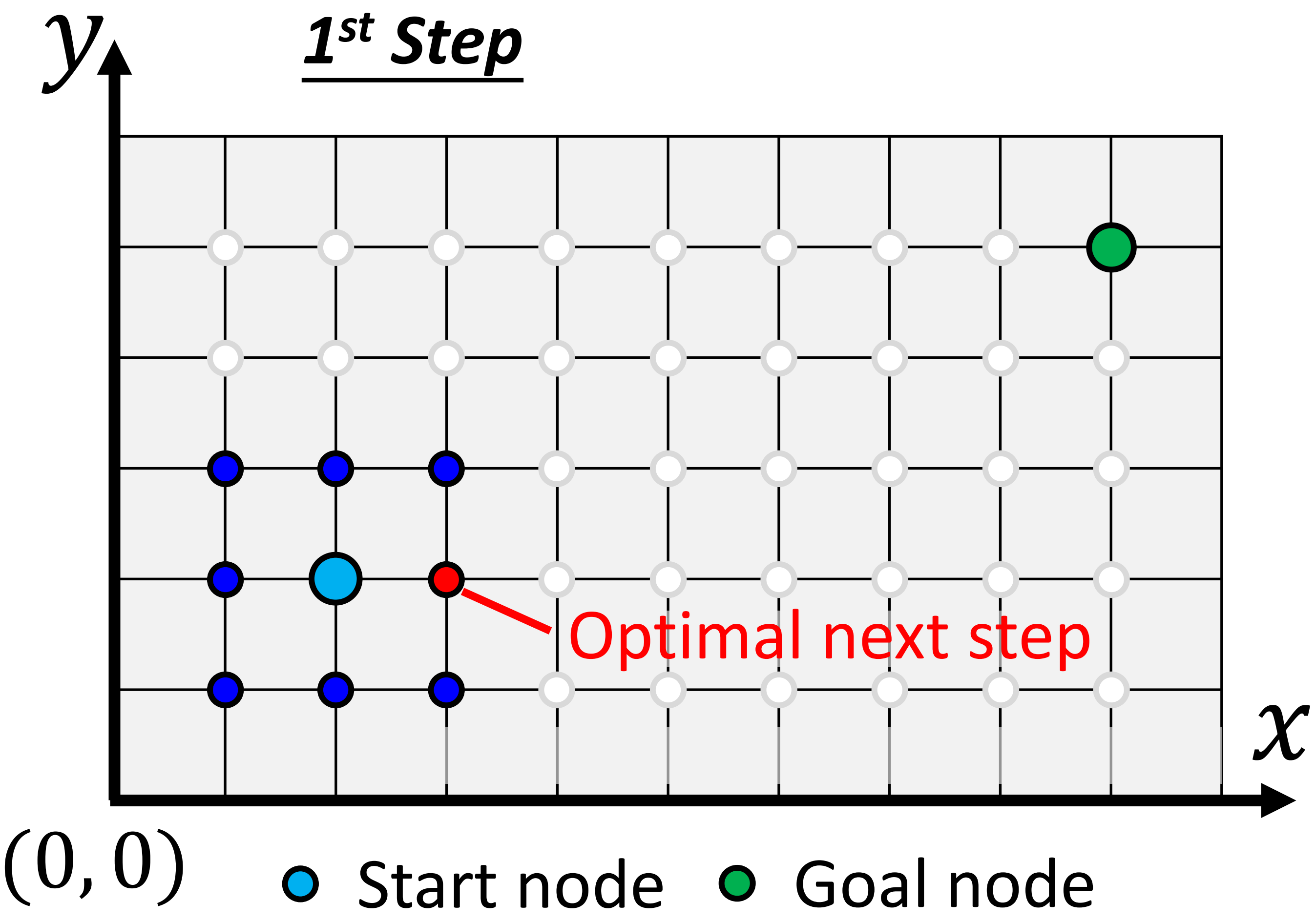
Open List
(searched nodes)

Node	f	Source
(1,1)	8.9	(2,2)
(1,2)	8.5	(2,2)
(1,3)	8.2	(2,2)
(2,1)	8.1	(2,2)
(2,3)	7.3	(2,2)
(3,1)	7.2	(2,2)
(3,2)	6.7	(2,2)
(3,3)	6.3	(2,2)

Close List
(arrived nodes)

Node	f	Source
Start	-	-

A-star (A*) Path Planning – Record Path



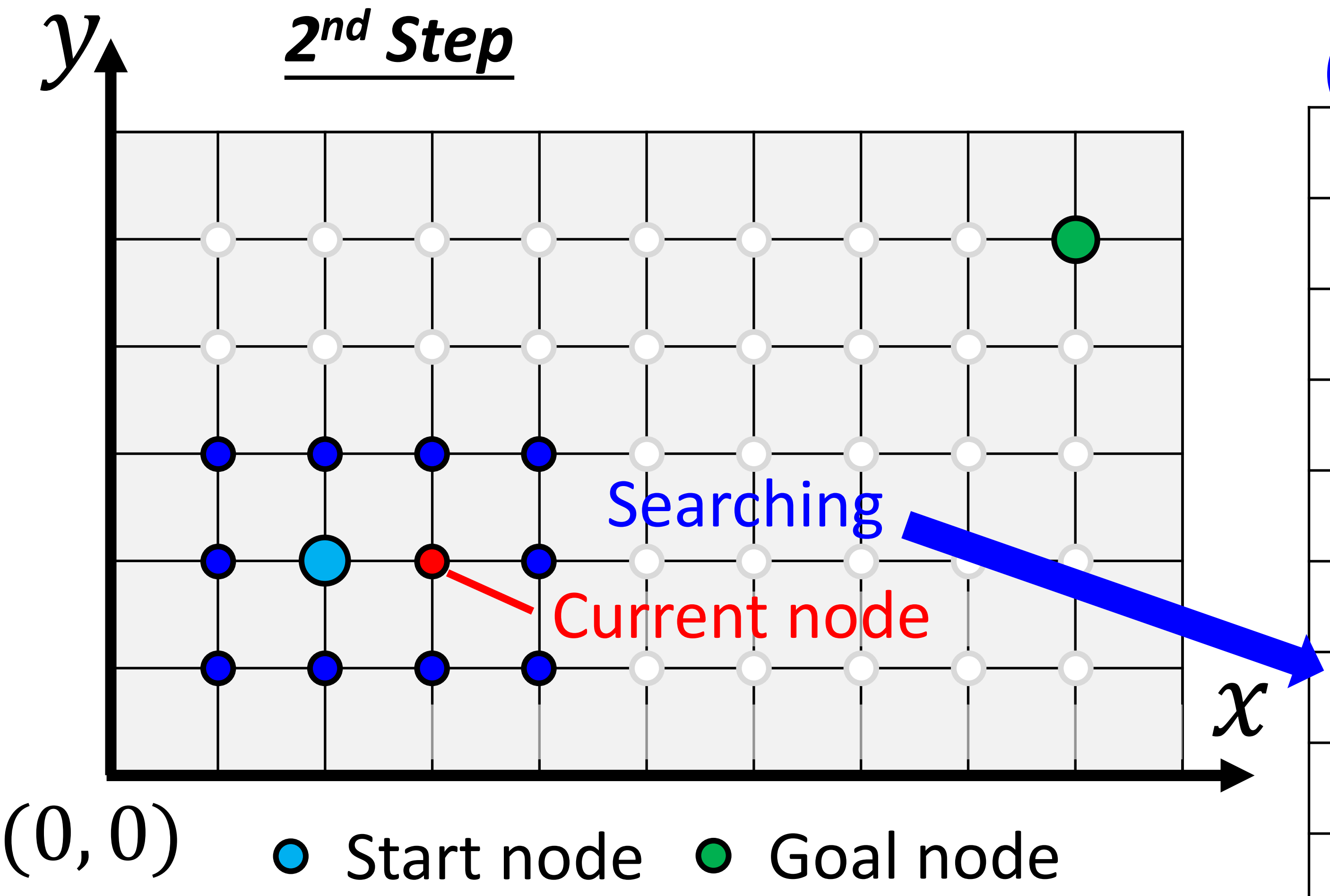
Open List
(searched nodes)

Node	f	Source
(1,1)	8.9	(2,2)
(1,2)	8.5	(2,2)
(1,3)	8.2	(2,2)
(2,1)	8.1	(2,2)
(2,3)	7.3	(2,2)
(3,1)	7.2	(2,2)
~~(3,2)~~	~~6.7~~	~~(2,2)~~
(3,3)	6.3	(2,2)

Close List
(arrived nodes)

Node	f	Source
Start	-	-
(3,2)	6.7	(2,2)

A-star (A*) Path Planning – Record Path



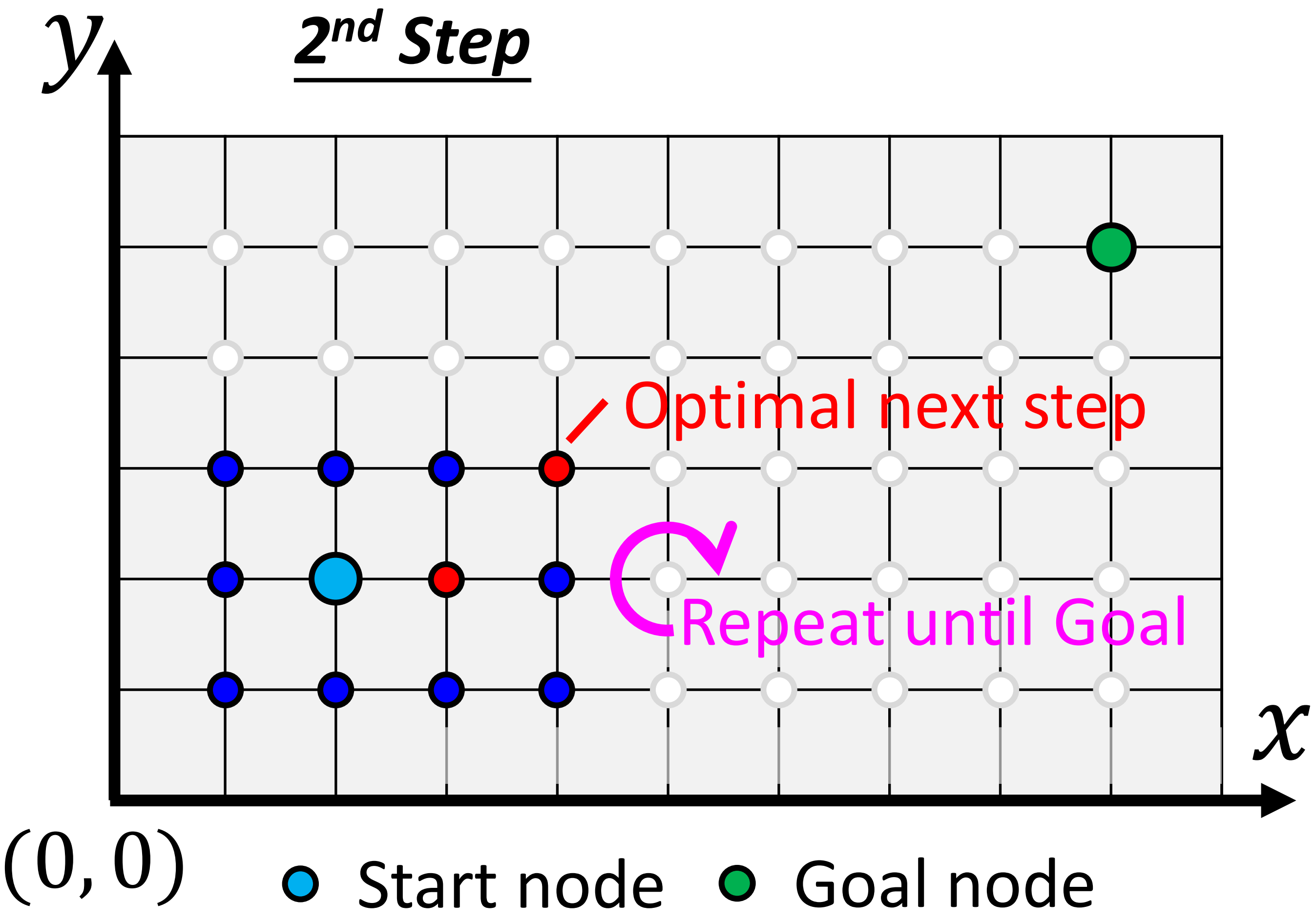
Open List (searched nodes)

Node	f	Source
(1,1)	8.9	(2,2)
⋮	⋮	⋮
(3,1)	7.2	(2,2)
(3,3)	6.3	(2,2)
⋮	⋮	⋮
(4,1)	7.8	(2,2)
(4,2)	7.8	(2,2)
(4,3)	8.8	(2,2)

Close List (arrived nodes)

Node	f	Source
Start	-	-
(3,2)	6.7	(2,2)

A-star (A*) Path Planning – Record Path



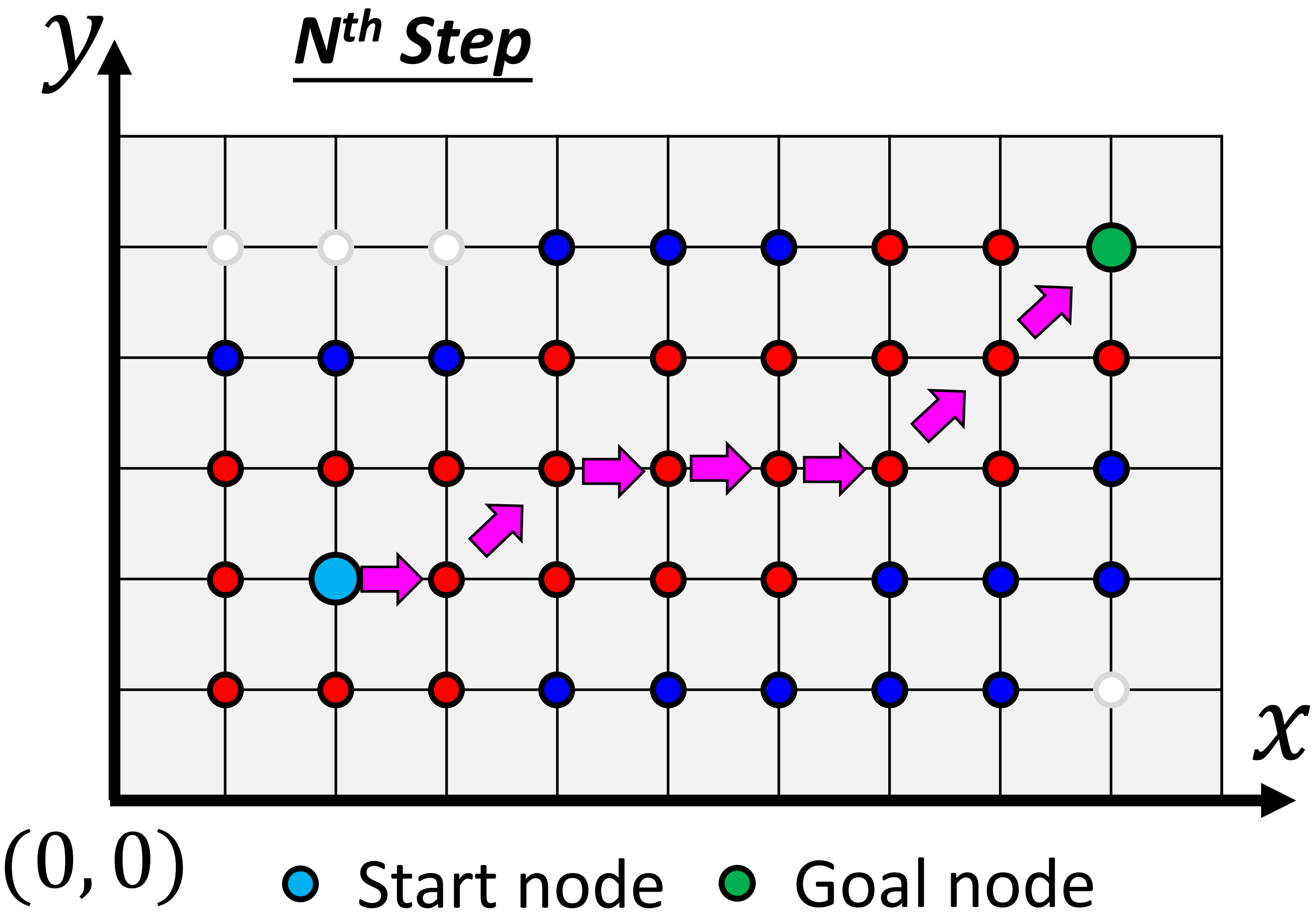
Open List
(searched nodes)

Node	f	Source
(1,1)	8.9	(2,2)
⋮	⋮	⋮
(3,1)	7.2	(2,2)
(3,3)	6.3	(2,2)
⋮	⋮	⋮
(4,1)	8.8	(3,2)
(4,2)	7.8	(3,2)
~~(4,3)~~	~~7.8~~	~~(3,2)~~

Close List
(arrived nodes)

Node	f	Source
Start	-	-
(3,2)	6.7	(2,2)
(4,3)	7.8	(3,2)

A-star (A*) Path Planning – Retrieve Best Path



Open List
(searched nodes)

Node	f	Source
(1,1)	8.9	(2,2)
⋮	⋮	⋮
(3,1)	7.2	(2,2)
(3,3)	6.3	(2,2)
⋮	⋮	⋮
⋮	⋮	⋮
⋮	⋮	⋮
~~Goal~~	~~x~~	~~(x,y)~~

Close List
(arrived nodes)

Node	f	Source
Start	-	-
(3,2)	6.7	(2,2)
(4,3)	7.8	(3,2)
⋮	⋮	⋮
⋮	⋮	⋮
Goal	x	(x,y)

Trace back arrived nodes

A-star Example

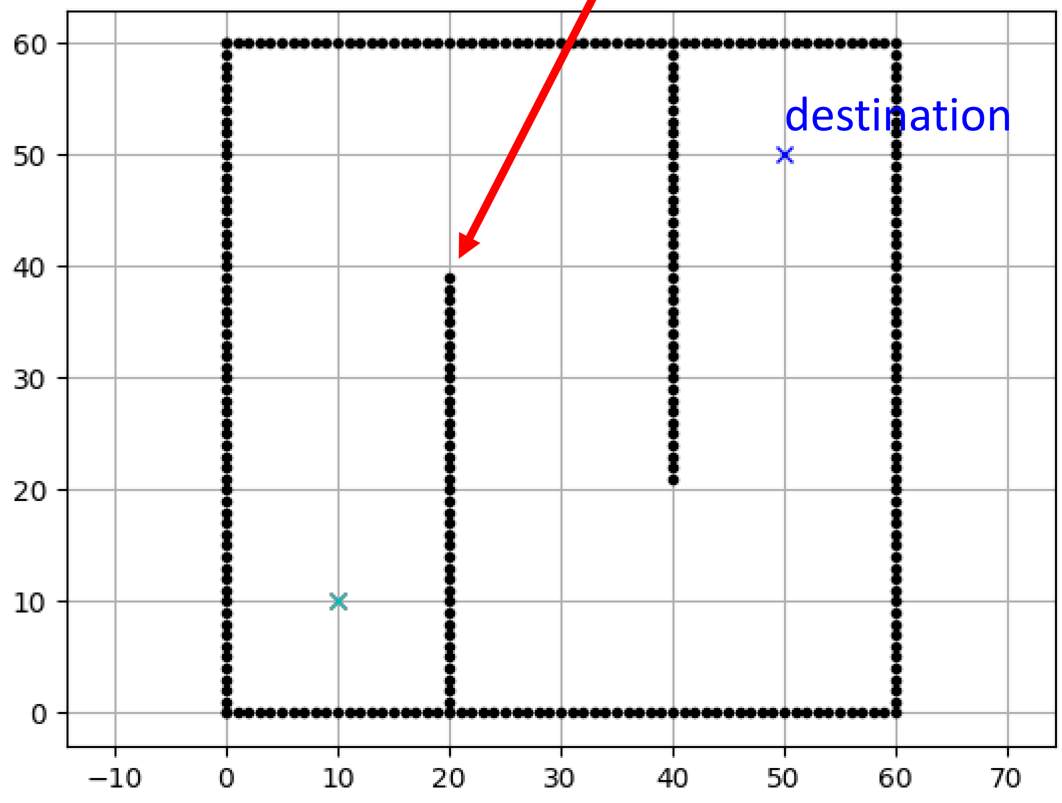
Each time A* enters a node, it calculates the cost: f(n) - n being the neighboring node

It travel to all of the neighboring nodes, and then enters the node with the lowest value of f(n)

These values we calculate using the following formula:

$$f(x, y) = g(x, y) + h(x, y)$$

Wall (obstacles) cannot go through!



https://towardsdatascience.com/a-star-a-search-algorithm-eb495fb156bb *Source: PythonRobotics*

Trip Cost of Flight

Trip Cost of Flight

The fundamental rationale of the cost index concept is to achieve minimum **trip cost** by means of a trade-off between operating costs per hour and incremental fuel burn.

Trip Cost: $$\boxed{C = C_F \cdot \Delta F \cdot T + C_T \cdot T + C_c}$$

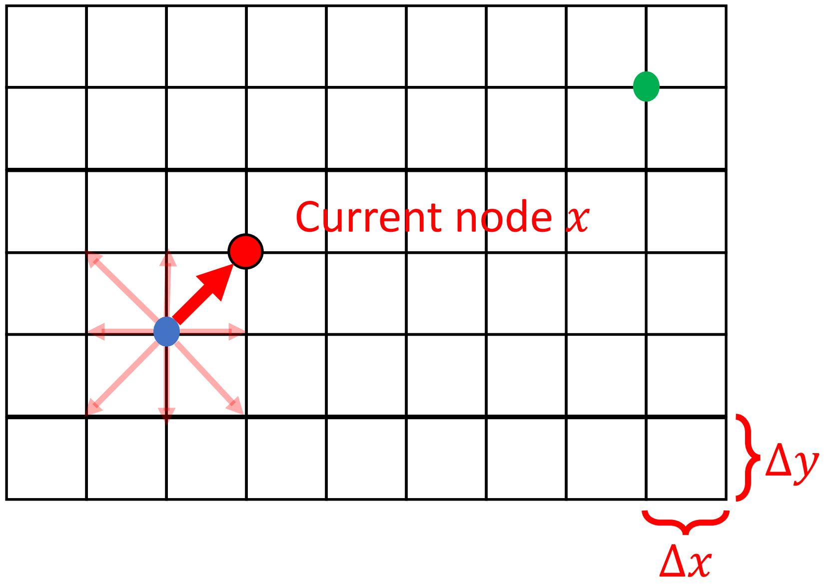
* Related to travelling time

With

- C_F=cost of fuel per kg
- C_T=time related cost per minute of flight
- C_c=fixed cost independent of time
- ΔF=trip fuel (e.g., 3000kg/h)
- T=trip Time (e.g., 8 hours from Hong Kong to Paris)

Consider this relationship to imitate the path planning for flights?

Conventional Path Planning Task



Goal node   Start node

Total cost: $f(x, y) = g(x, y) + h(x, y)$

Distance information (conventional)

Example for one step:

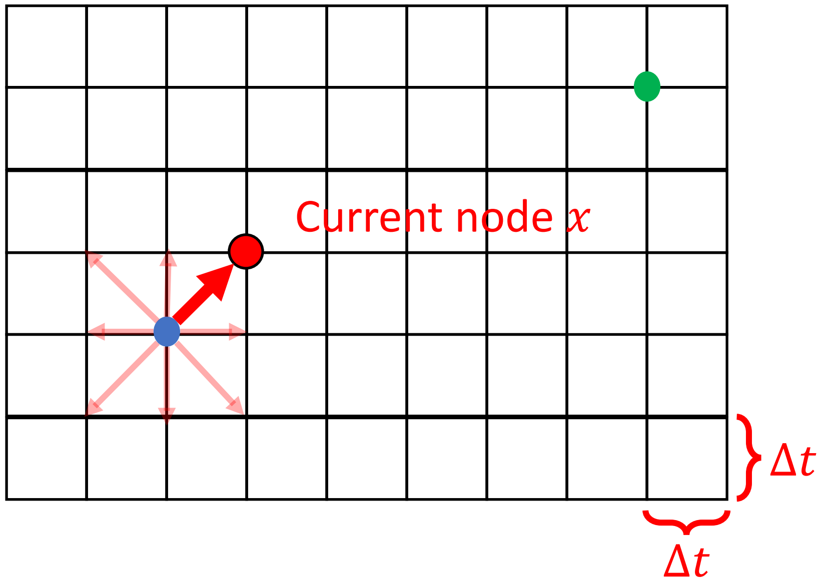
$$g(x, y) = \sqrt{\Delta x^2 + \Delta y^2}$$

Objective: Find the path with the lowest traveling distance. (in the unit of meter)

Path Planning Task for Out Flight

Total cost: $f(n) = g(n) + h(n)$

Traveling time (our case for flight)



Example for first step:

$$g(n) = s(n) = \sqrt{2}\Delta t$$

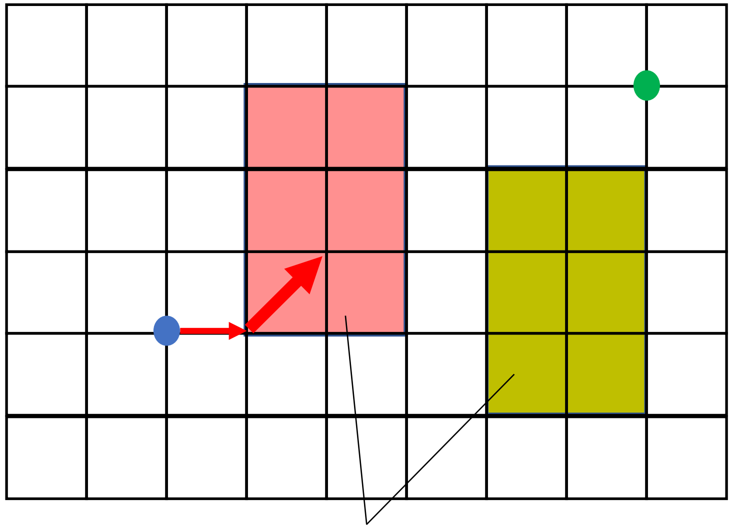
Time cost on each step:

$$s(n) = \begin{cases} \Delta t, & vertical/horizontal\ motion \\ \sqrt{2}\Delta t, & diagonal\ motion \end{cases}$$

Objective: Find the path with the lowest traveling time. (in the unit of min)

● Goal node ● Start node n – index of nodes

Flight Planning Considering Cost Intensive Areas



Cost intensive area (addition time)

Cost Intensive Areas: The cost for flying through such area is **increased** due to airflow, legal restrictions and other reasons.

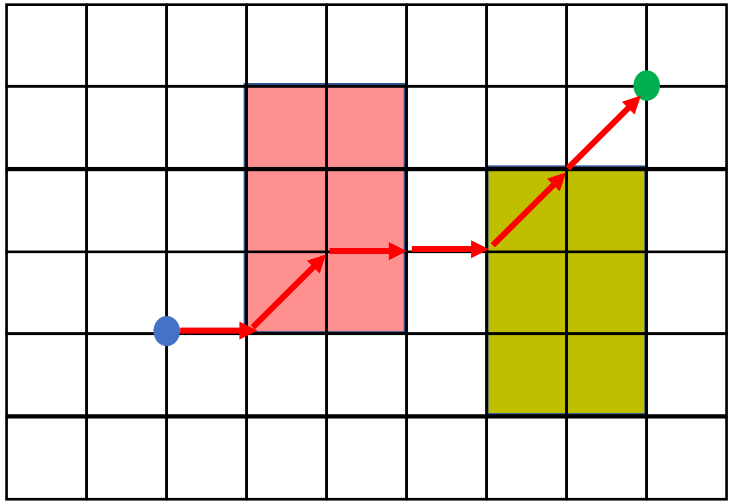
Total cost for one node:

$$f(n) = \underbrace{g(n)}_{\text{Previous cost from start}} + \underbrace{\alpha \cdot s(n)}_{\text{Additional time at current step}} + h(n)$$

α – additional time factor (equal zero for normal area)

● Goal node ● Start node

Flight Path Planning Trip Cost



Cost Intensive Areas: The cost for flying through such area is **increased** due to airflow, legal restrictions and other reasons.

Total cost for one node:

$$f(n) = g(n) + \alpha \cdot s(n) + h(n)$$

Path planning

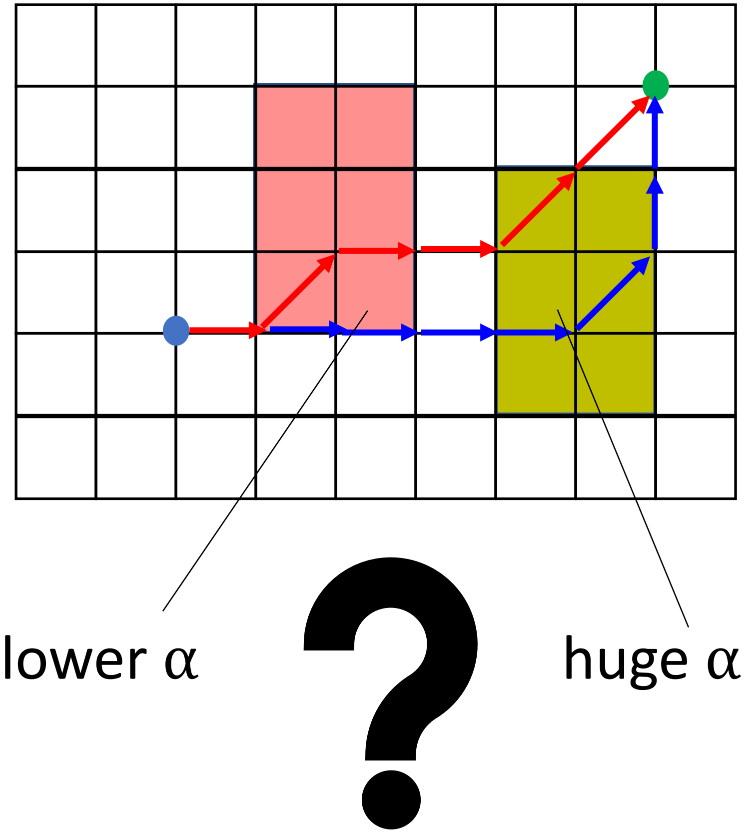
Additional cost on each node

$$T_{best} = \min[f(goal)]$$
$$= \min\left[g(goal) + \alpha_1 s(n_1) + \alpha_2 s(n_2) + \cdots + \alpha_{goal} s(n_{goal})\right]$$

Trip cost for planned path: $\boxed{C = C_F \cdot \Delta F \cdot T_{best} + C_T \cdot T_{best} + C_c}$

- Goal node
- Start node
- α – additional time factor

How it choose different routes?



Cost Intensive Areas: The cost for flying through such area is **increased** due to airflow, legal restrictions and other reasons.

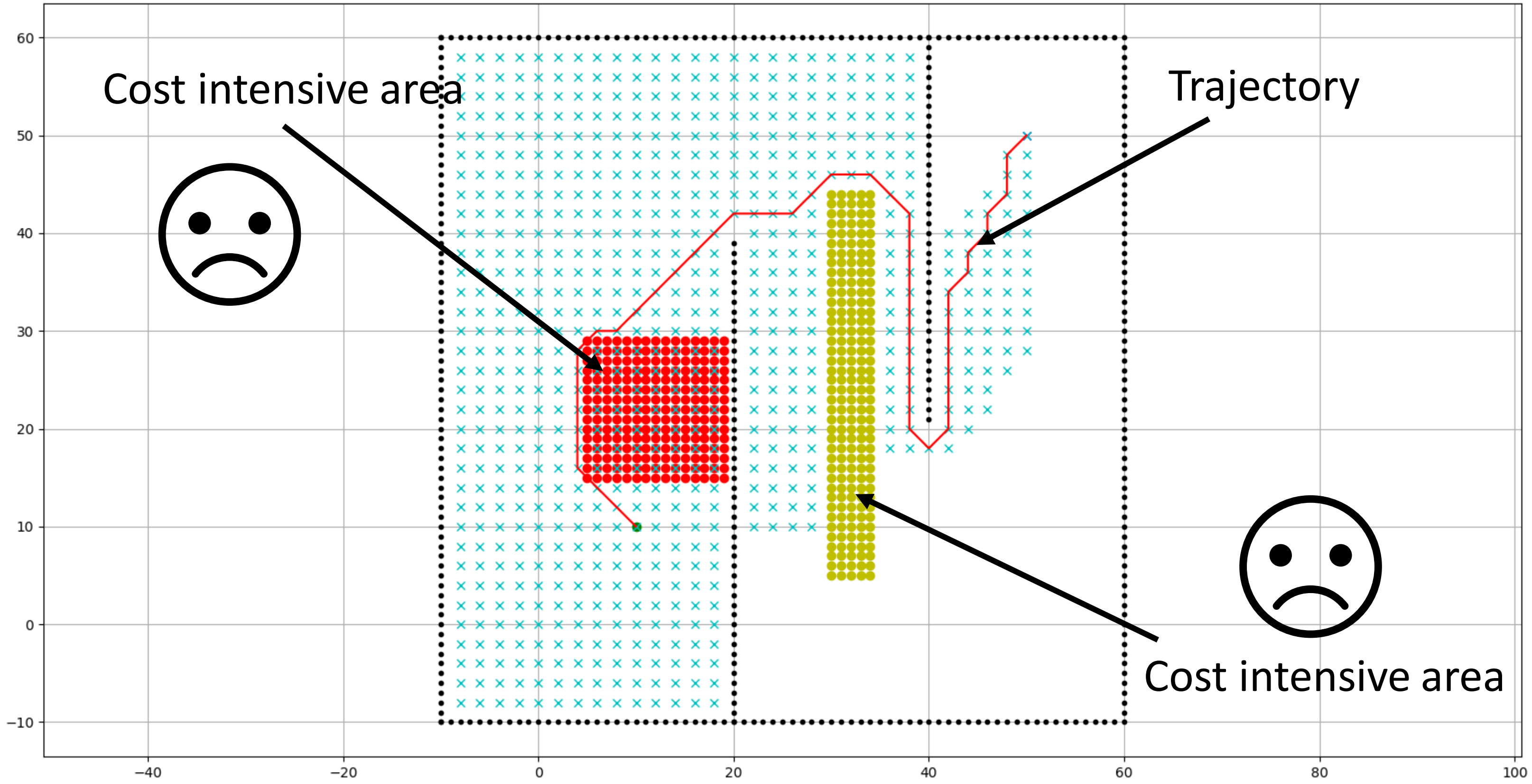
Time for the planned path:

$$T_{best} = \min[g(goal) + \underline{\alpha_1 s(n_1)} + \underline{\alpha_2 s(n_2)} + \cdots + \underline{\alpha_{goal} s(n_{goal})}]$$

Depending on the extra time accumulated more specifically, the additional time factor

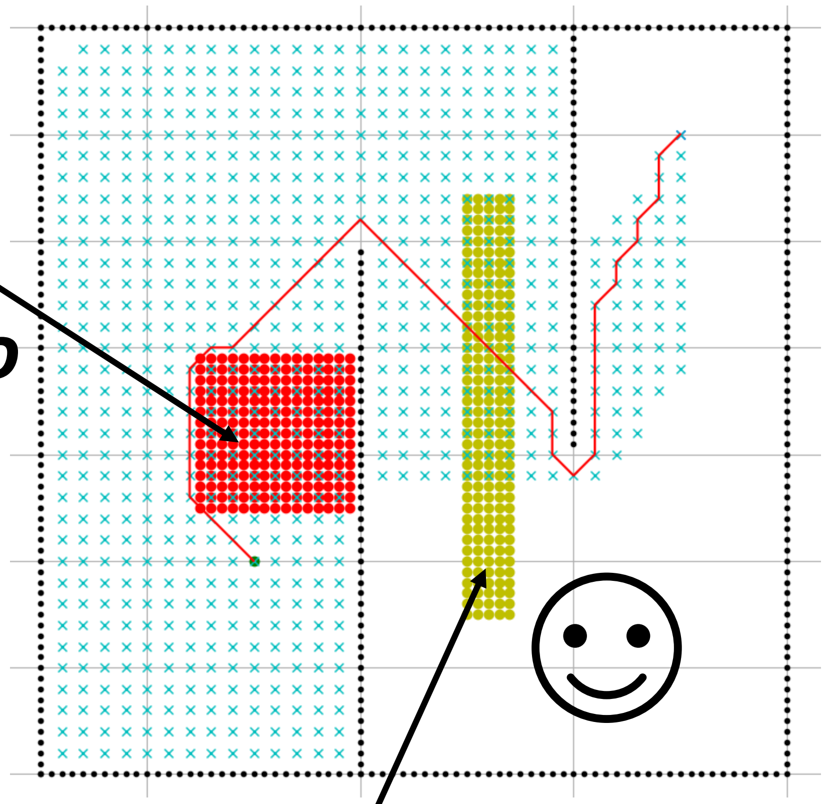
Goal node   Start node   α – additional time factor

Example of Flight Path Planning



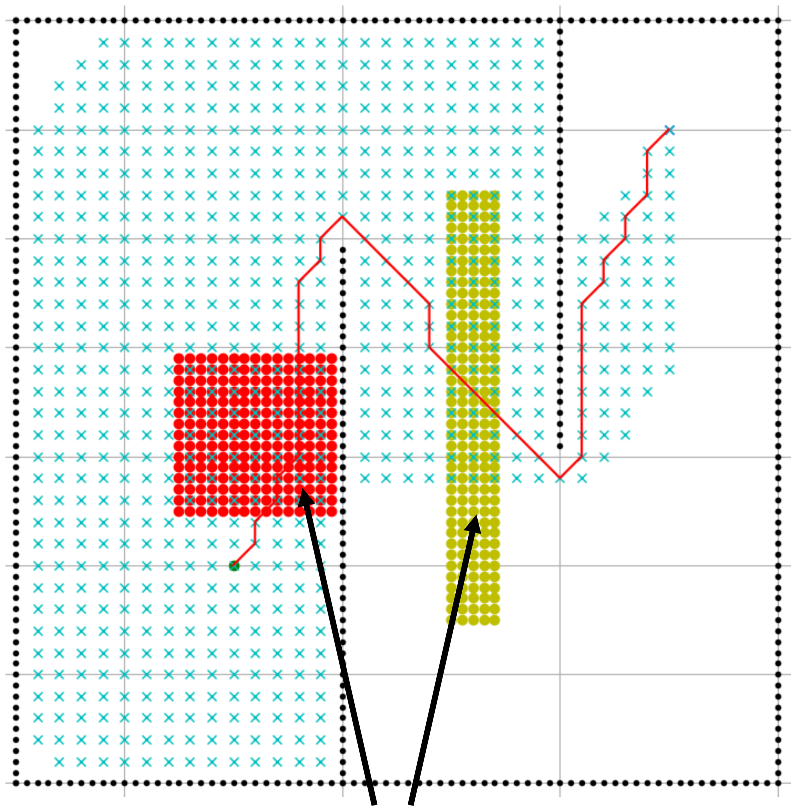
Cost is way too high for going through, better to avoid!

Example of Flight Path Planning



Cost too much

Not too much and acceptable



Both acceptable

Flight Path Planning Project

You will be creating and completing your own path planning program based on groups

You can find the project tasks / requirements in the GitHub homepage

Additional resources could be found inside the course GitHub repository

- Video tutorial
- Tutorial slides