

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

Week 8 (Introduction to Path Planning)

Dr Guohao Zhang, and Dr Lingxiao Wu

Assisted by

Mr Zekun ZHANG, Mr Mingda YE, Ms Jingxiaotao FANG, Mr Chin Lok TSANG,
Mr Di HAI, Mr Zhengdao LI

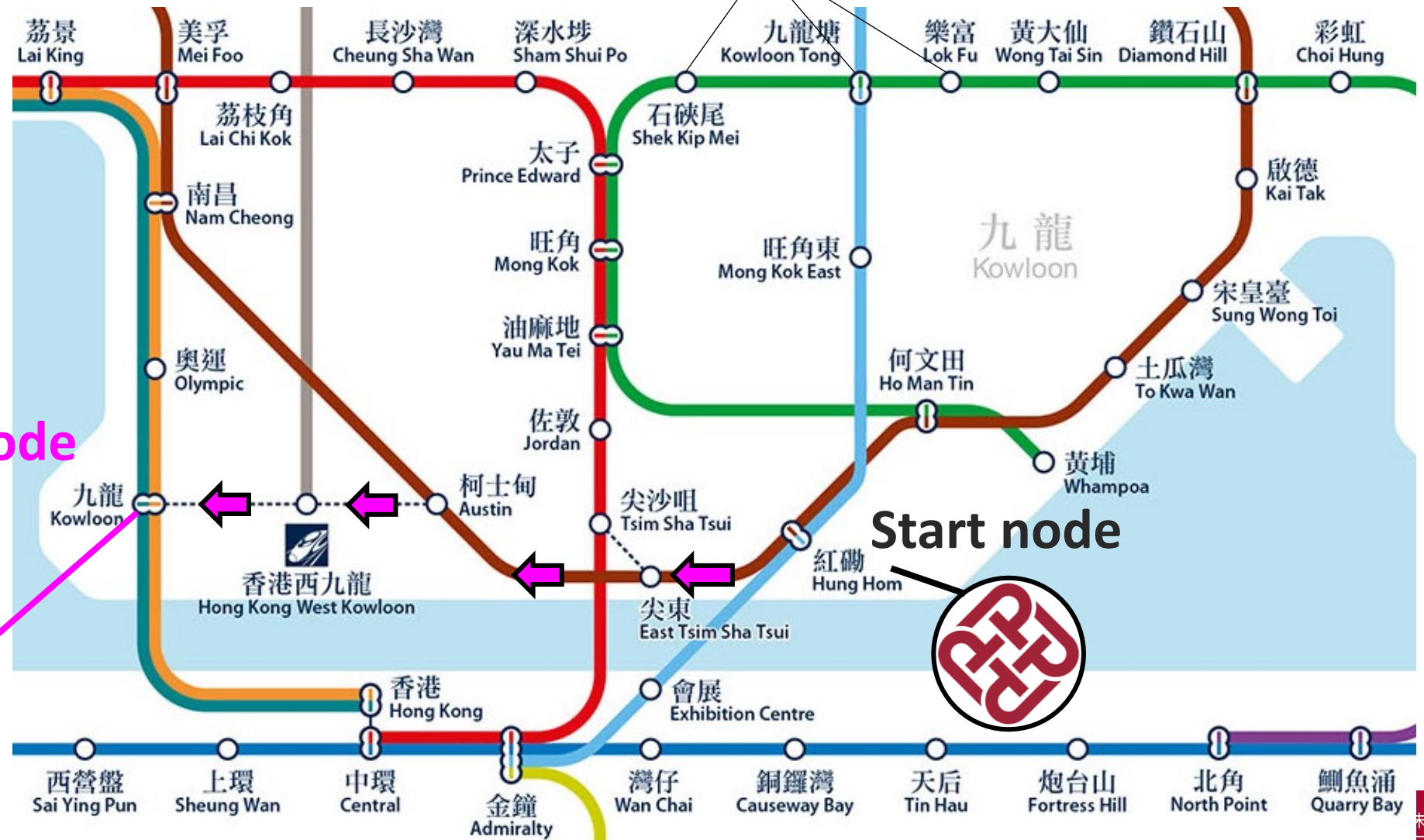
Introduction to A* Path Planning Algorithm

Daily Path Planning

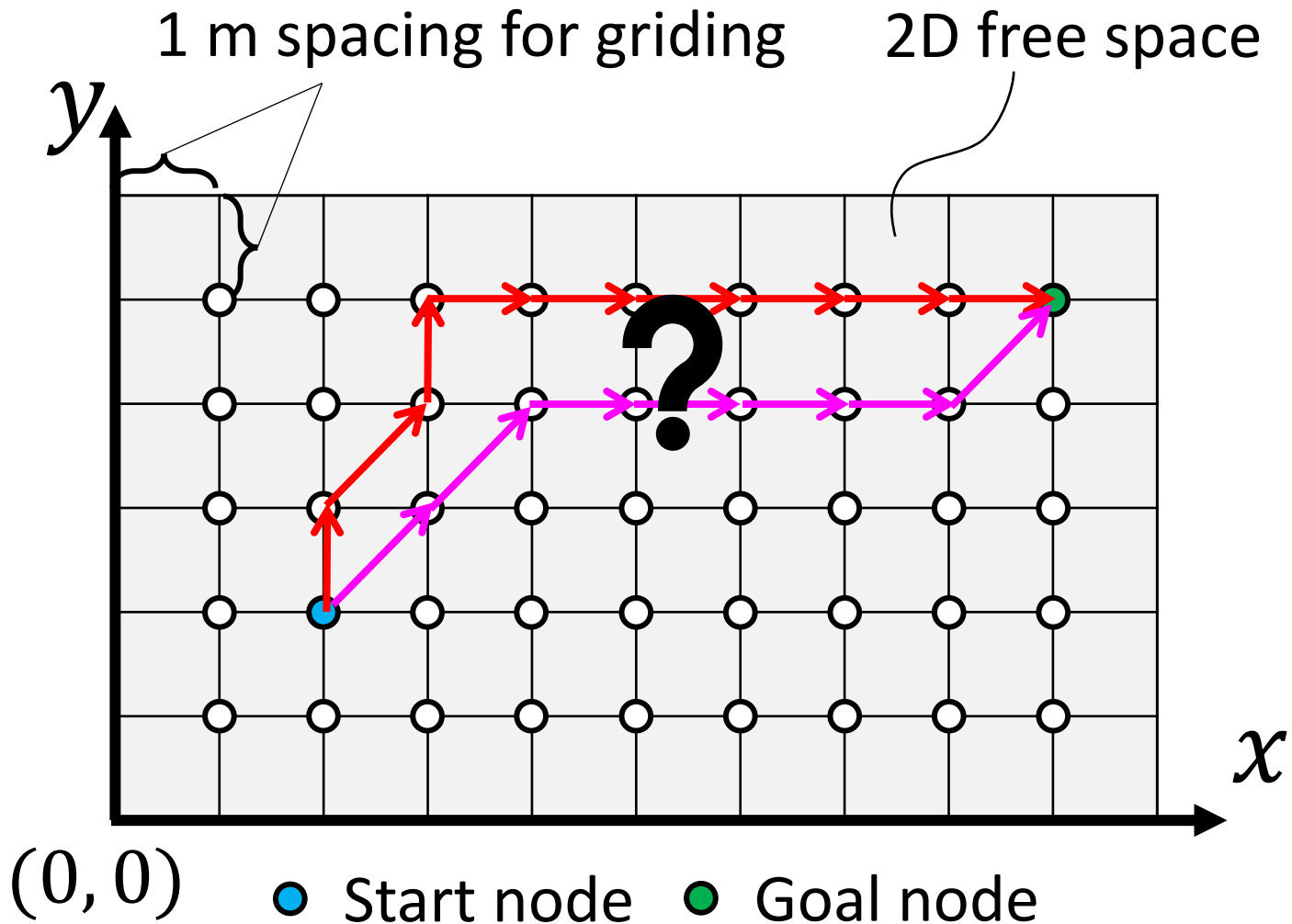
MTR stations – available **node** to go



Goal node



Definition of Path Planning



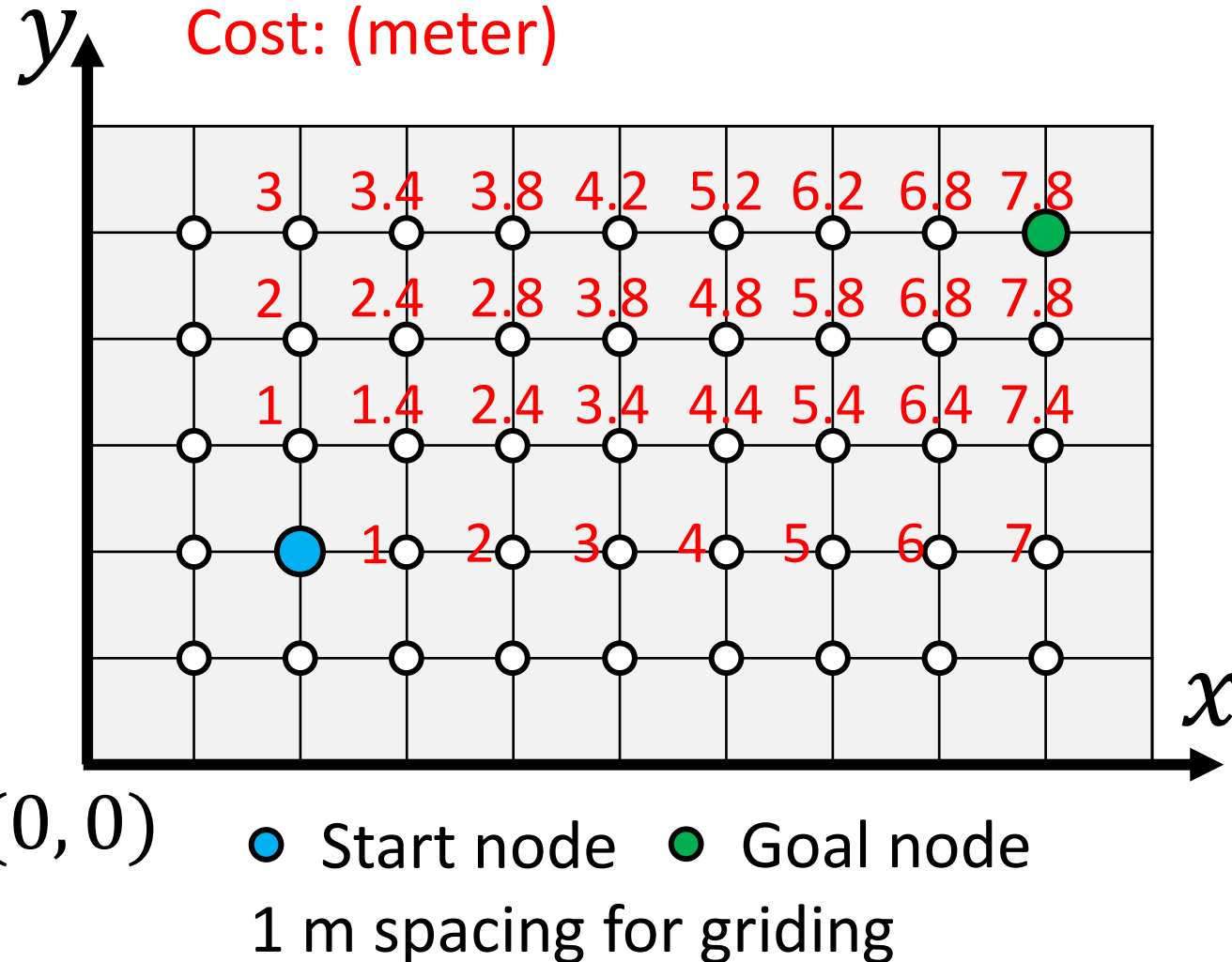
- **Node** — All potential position you can go across with a unique position (x, y)

- **Search space** — A collection of nodes, like all board positions of a board game.

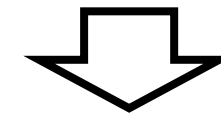
- **Objective of path planning**— Find the shortest routes with smallest cost from start node to goal node.

Which one is better?

Path Planning by Checking All Available Nodes



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



Record its shortest path
between Start node

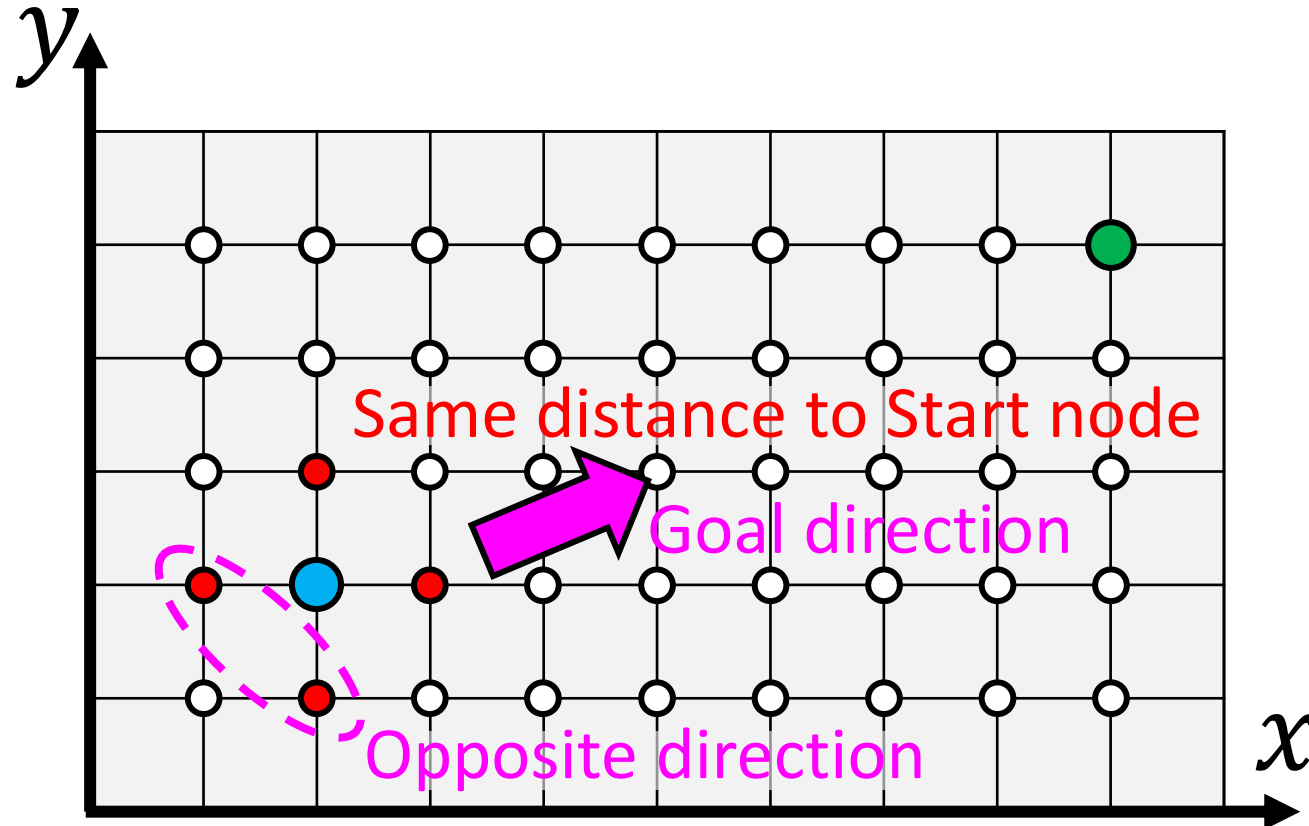
Reach Goal



Retrieve the shortest path
(Dijkstra's algorithm)

Need higher efficiency!

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



(0, 0) ● Start node ● Goal node
1 m spacing for gridding

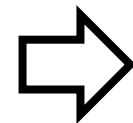
Test each possible nodes
one-by-one from Start node



Record its shortest path
between Start node

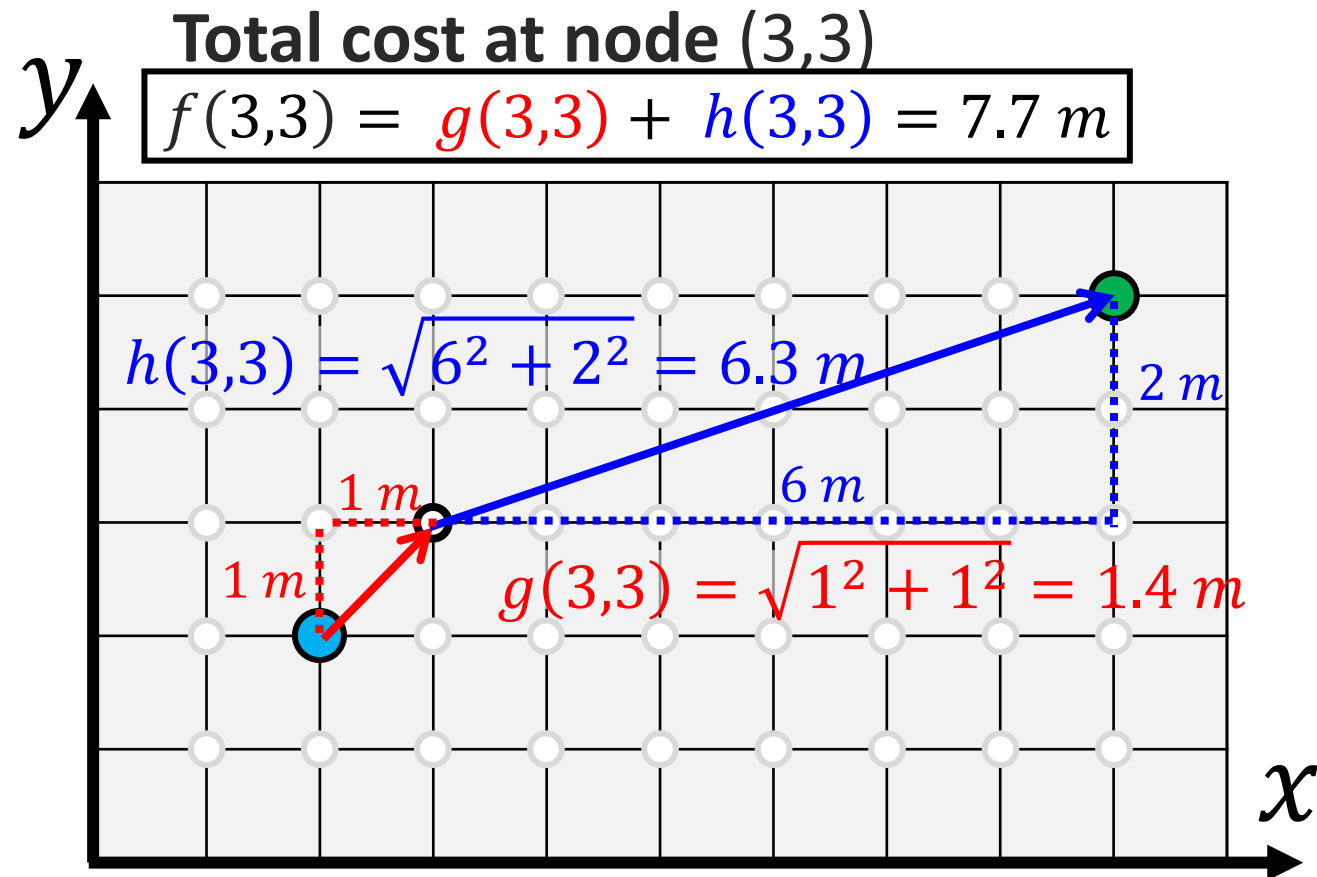


We also know a rough
direction to Goal!



A-star Search Algorithm

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



Definition of cost:

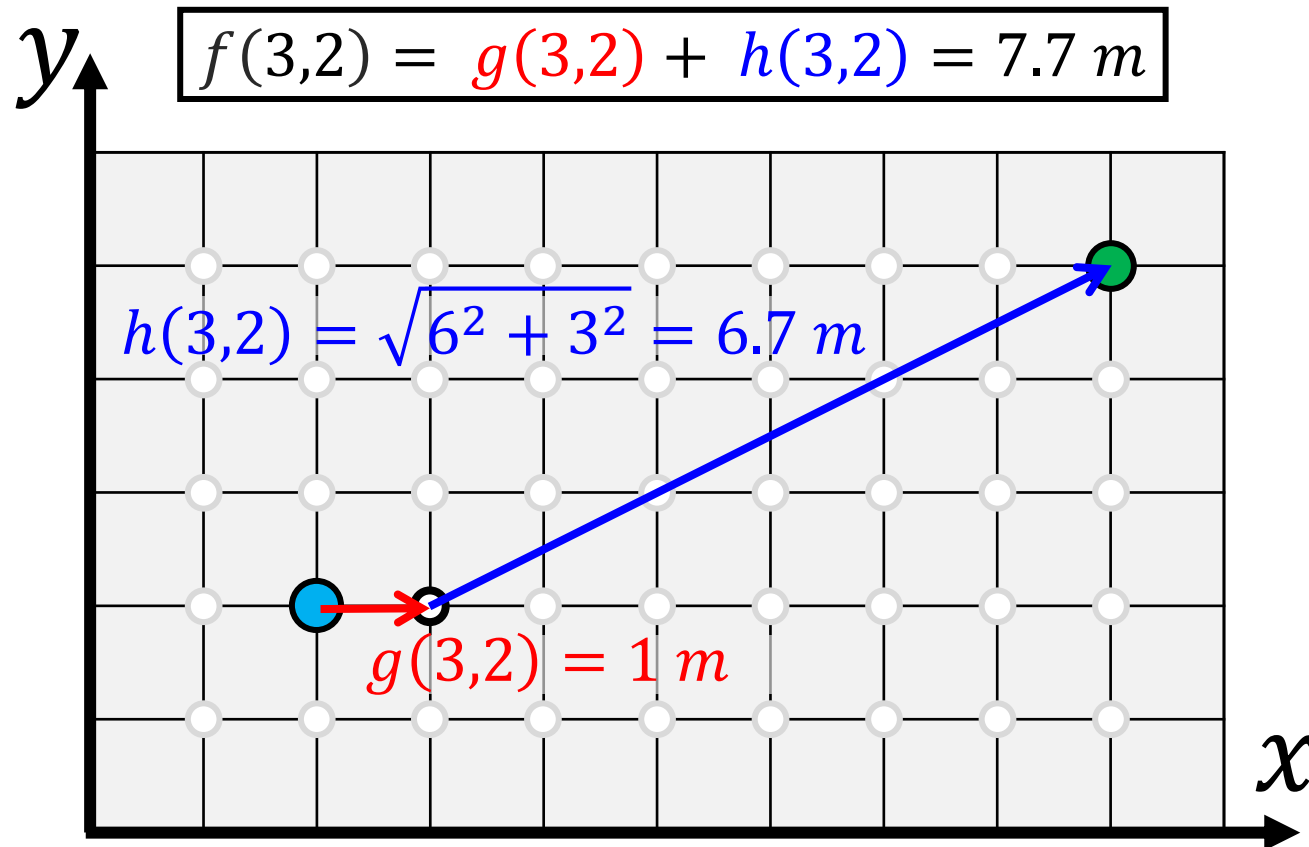
$g(x, y)$ — this represents the exact cost of the path **from** the **Start** node to node (x, y)

$h(x, y)$ — 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)

- (0, 0) ● Start node
- Goal node 1 m spacing for gridding

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



Definition of cost:

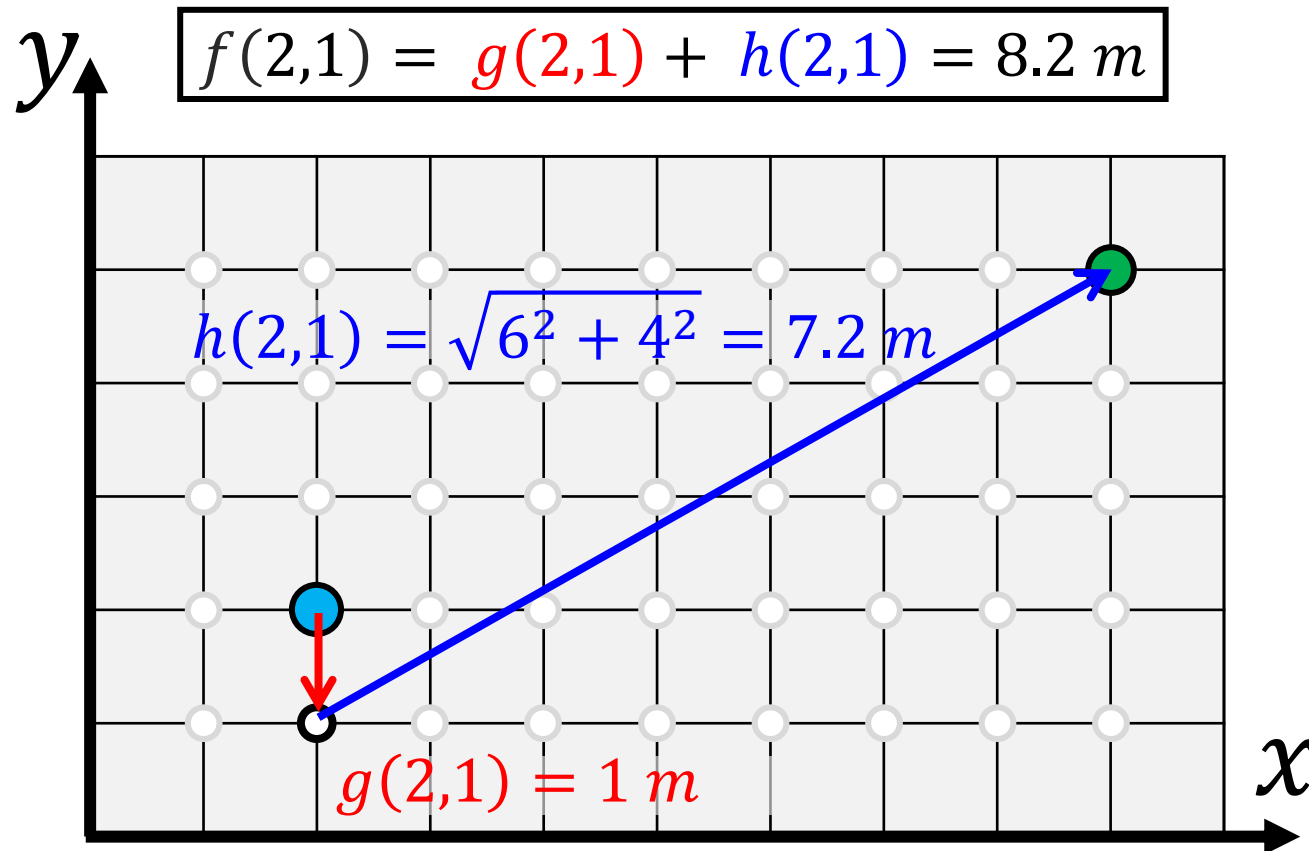
$g(x, y)$ — this represents the exact cost of the path **from** the **Start** node to node (x, y)

$h(x, y)$ — 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)

(0, 0) ● Start node ● Goal node
1 m spacing for gridding

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



Definition of cost:

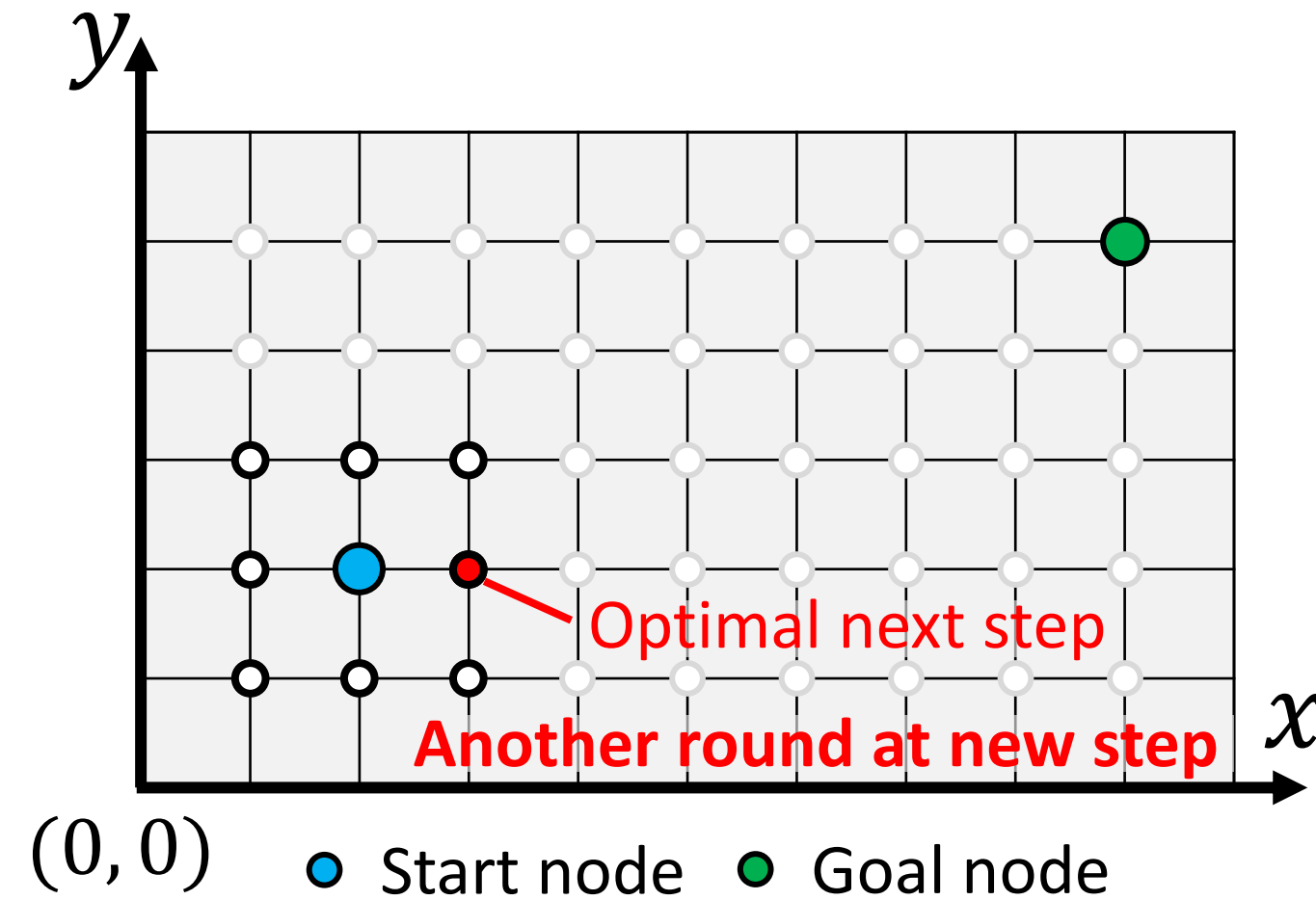
$g(x, y)$ — this represents the exact cost of the path **from** the **Start** node to node (x, y)

$h(x, y)$ — 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)

(0, 0) ● Start node ● Goal node
1 m spacing for gridding

Total Costs at Neighbouring Nodes



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

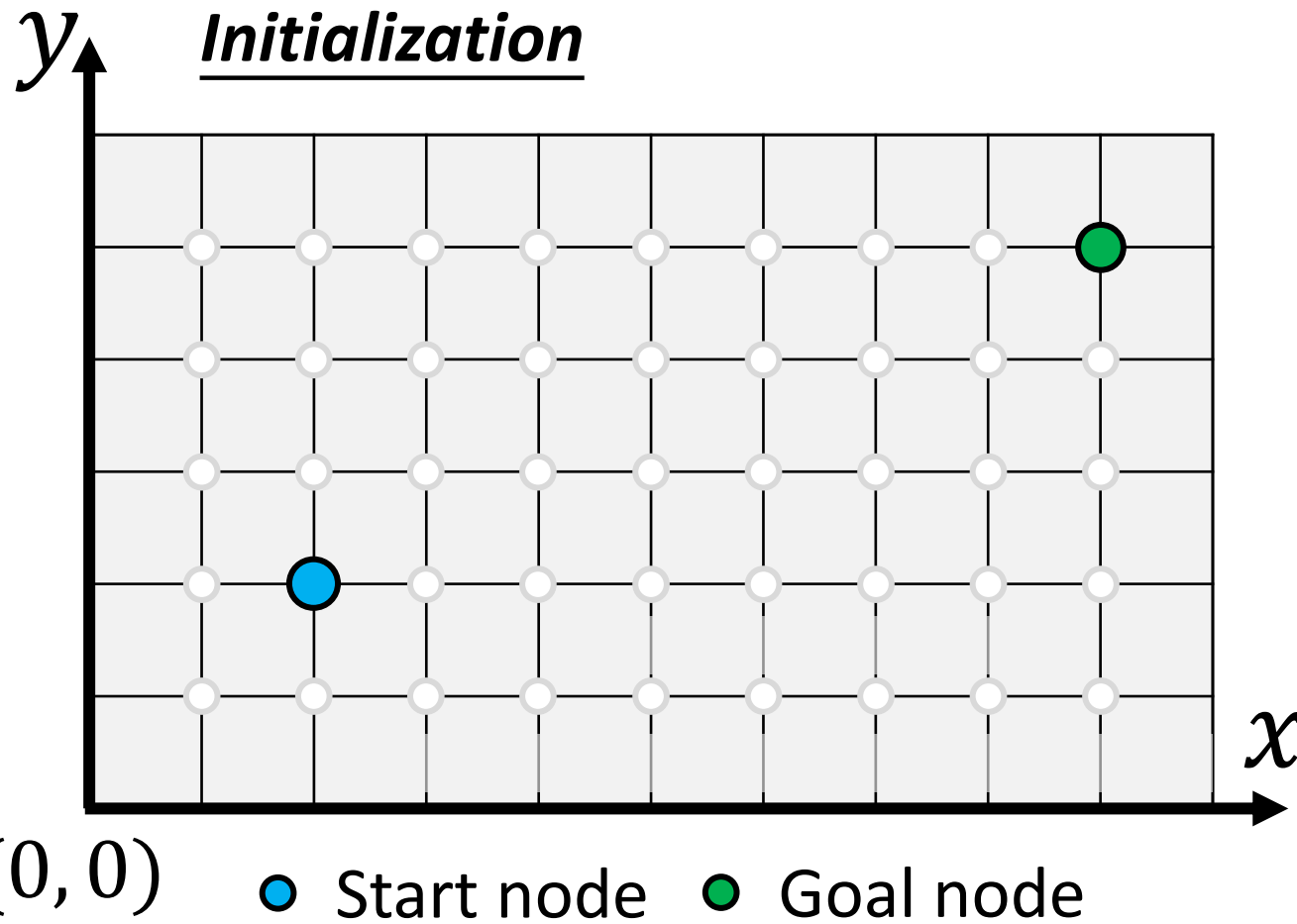
| Node ID | $g(x, y)$ | $h(x, y)$ | $f(x, y)$ |
|---------|-----------|-----------|-----------|
| (1,1) | 1.4 | 8.9 | 10.3 |
| (1,2) | 1 | 8.5 | 9.5 |
| (1,3) | 1.4 | 8.2 | 9.6 |
| (2,1) | 1 | 8.1 | 9.1 |
| (2,3) | 1 | 7.3 | 8.3 |
| (3,1) | 1.4 | 7.2 | 8.6 |
| (3,2) | 1 | 6.7 | 7.7 |
| (3,3) | 1.4 |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

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



Opening Minds • Shaping the Future • 啟迪思維 • 成就未來

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



Open List
(searched nodes)

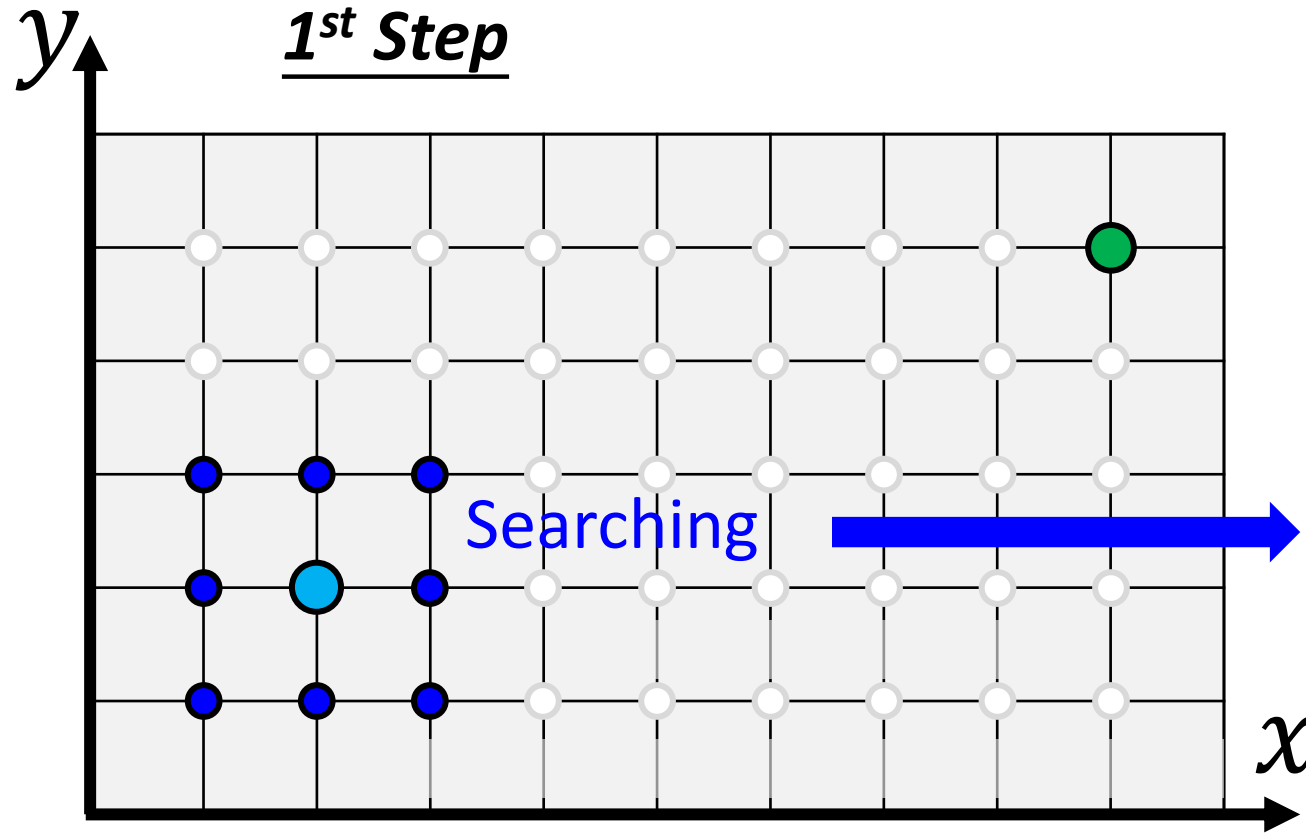
| Node | f | Source |
|------|-----|--------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Close List
(arrived nodes)

| Node | f | Source |
|-------|-----|--------|
| Start | - | - |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

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

1st Step



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) |

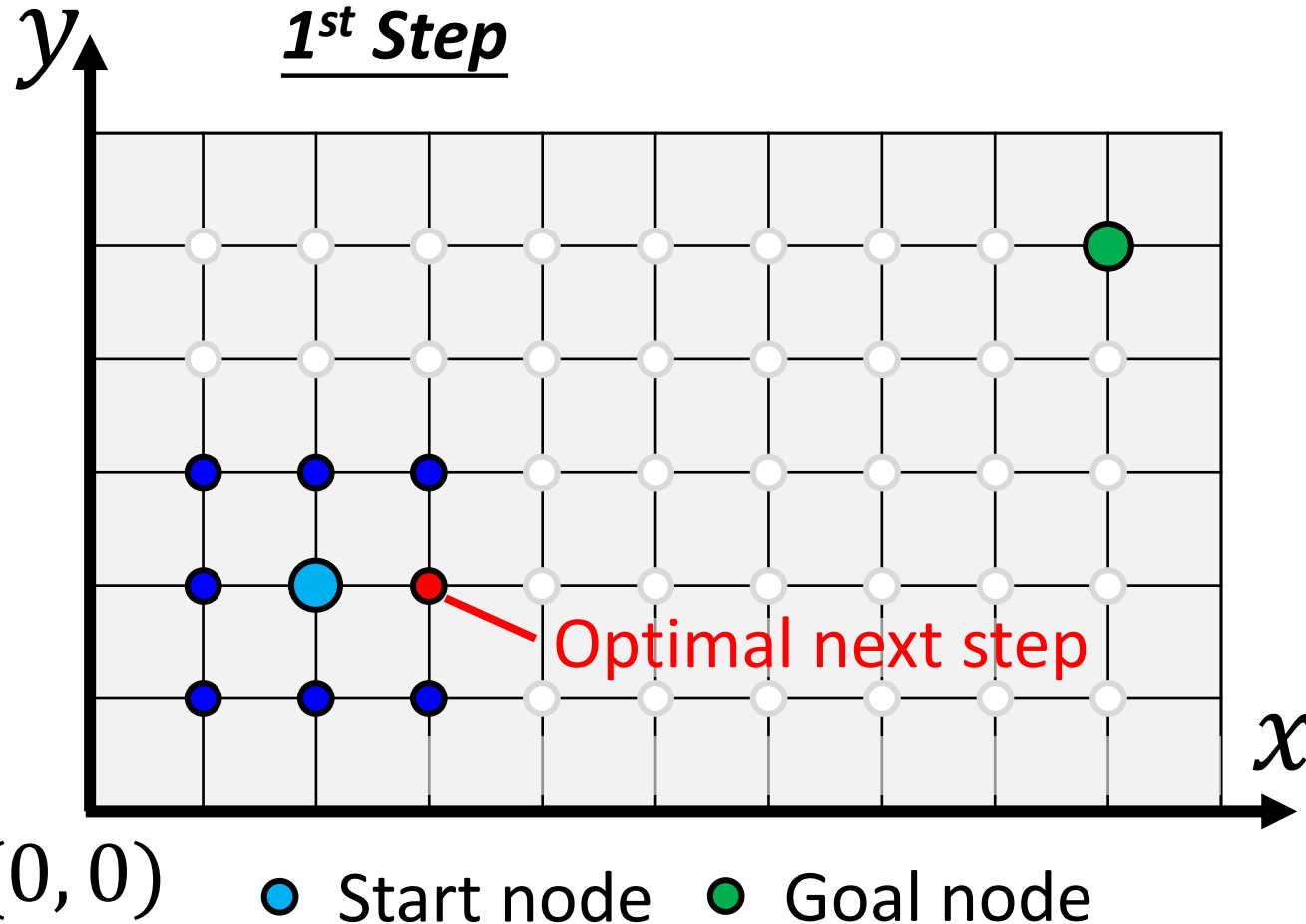
Close List
(arrived nodes)

| Node | f | Source |
|-------|-----|--------|
| Start | - | - |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

(0,0) ● Start node ● Goal node

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

1st Step

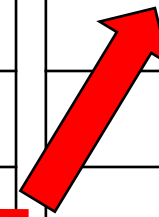


Open List
(searched nodes)

| 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) |

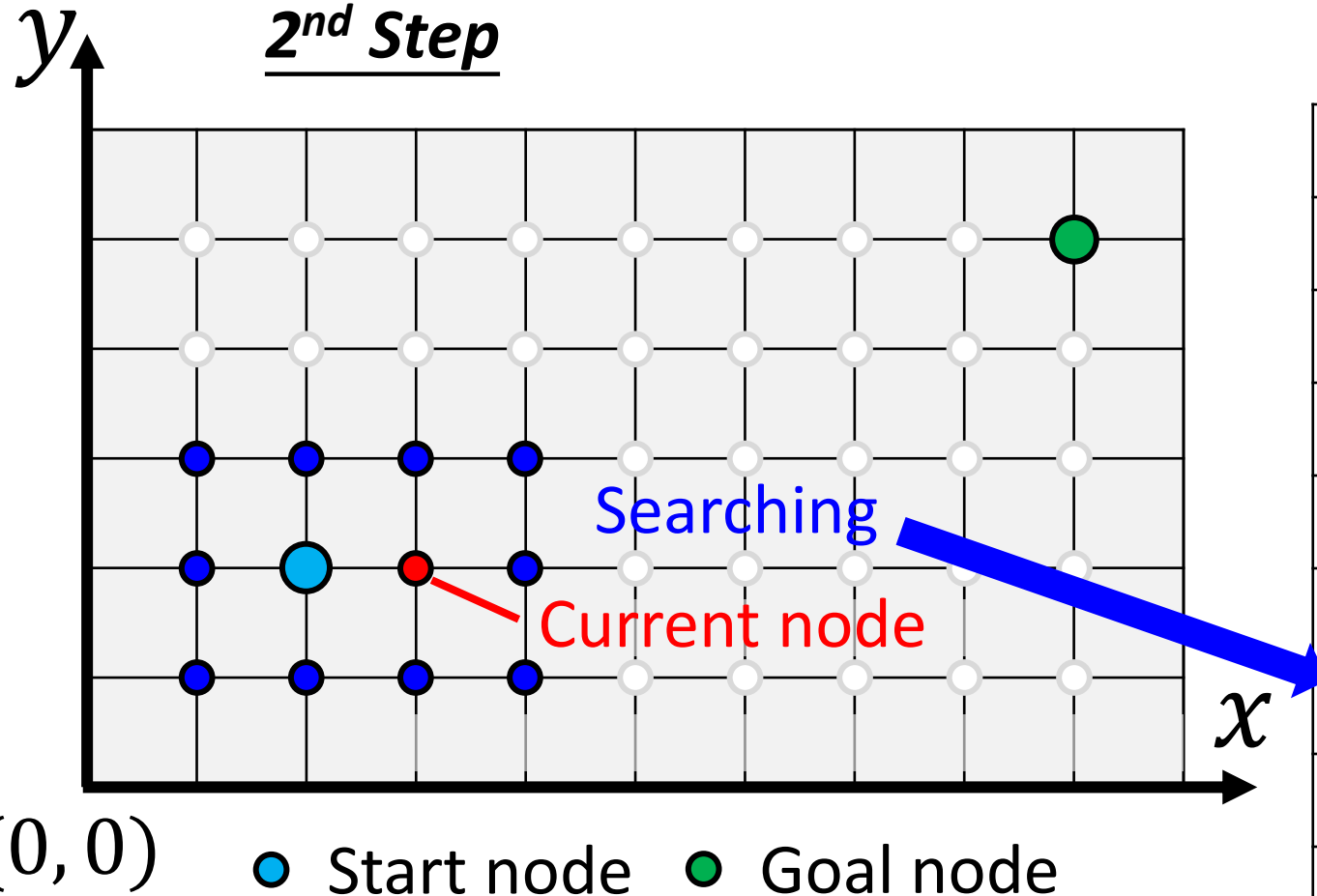
Close List
(arrived nodes)

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



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

2nd Step



Open List
(searched nodes)

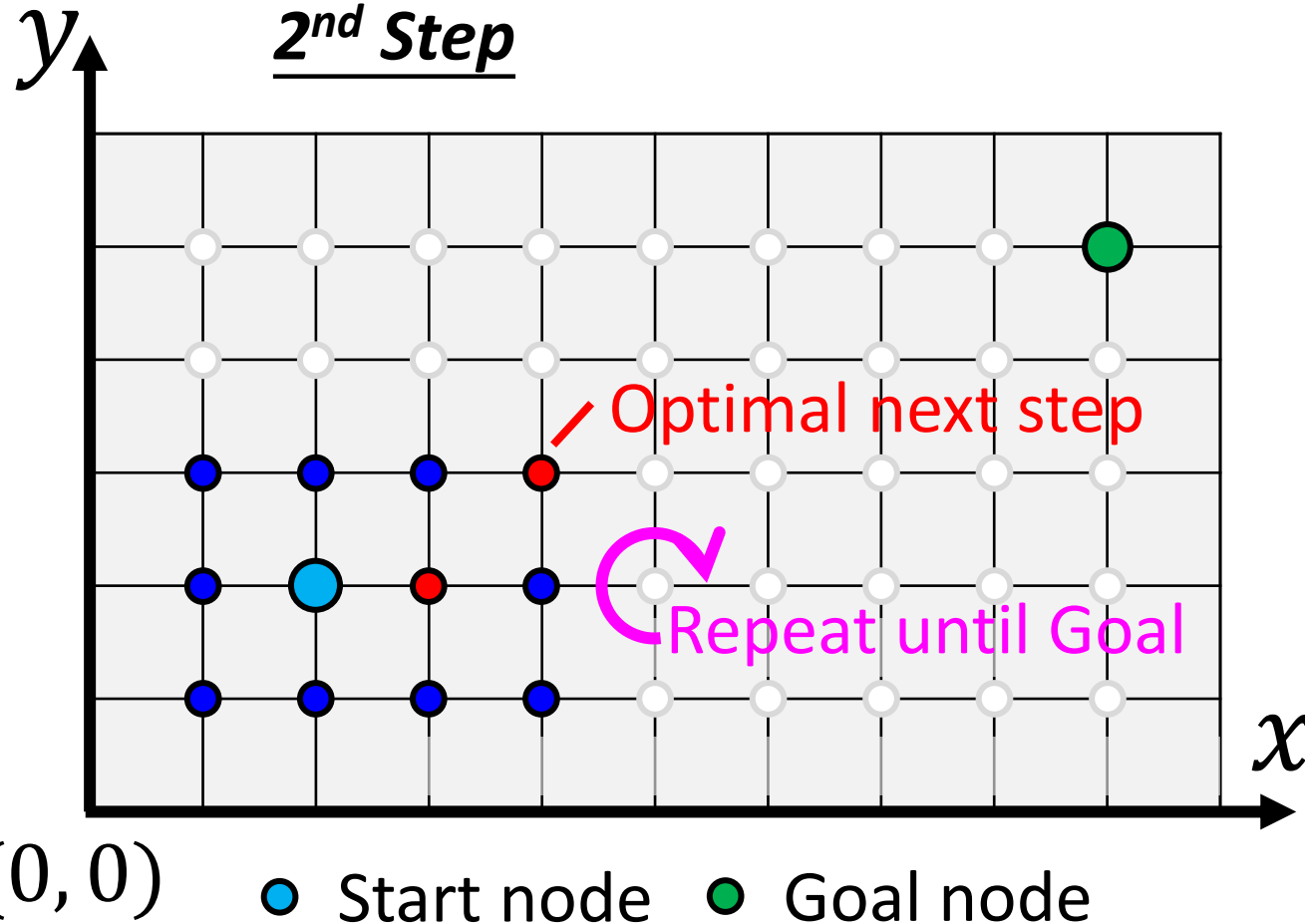
| Node | f | Source |
|-------|-----|--------|
| (1,1) | 8.9 | (2,2) |
| ⋮ | ⋮ | ⋮ |
| (3,1) | 7.2 | (2,2) |
| (3,3) | 6.3 | (2,2) |
| ⋮ | ⋮ | ⋮ |
| (4,1) | 7.8 | (2,2) |
| (4,2) | 7.8 | (2,2) |
| (4,3) | 8.8 | (2,2) |

Close List
(arrived nodes)

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

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

2nd Step

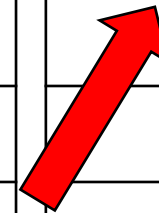


Open List
(searched nodes)

| 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) |

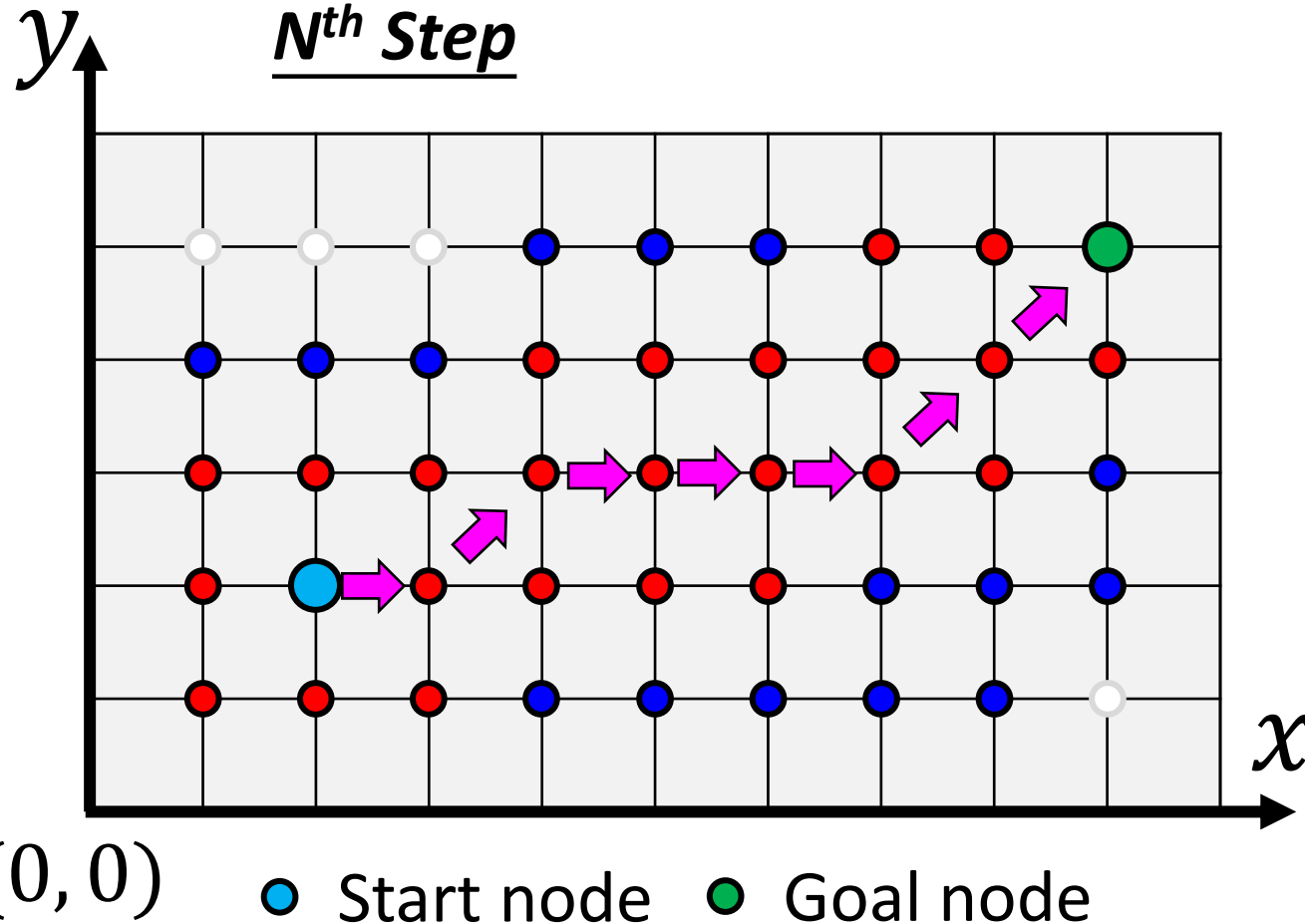
Close List
(arrived nodes)

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



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

N^{th} Step



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

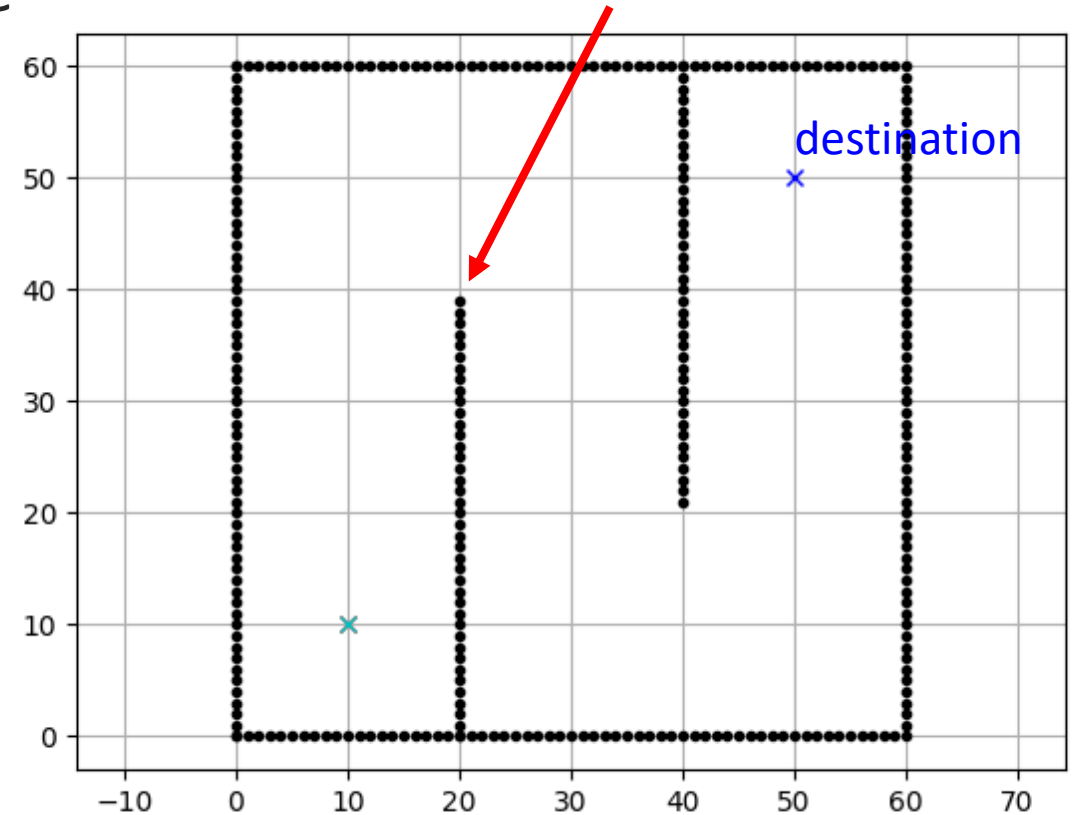
Each time A* enters a node, it calculates the cost: $f(n)$ - n being the neighboring node

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!



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:
$$C = C_F \cdot \Delta F \cdot T + C_T \cdot T + C_c$$

* Related to travelling time

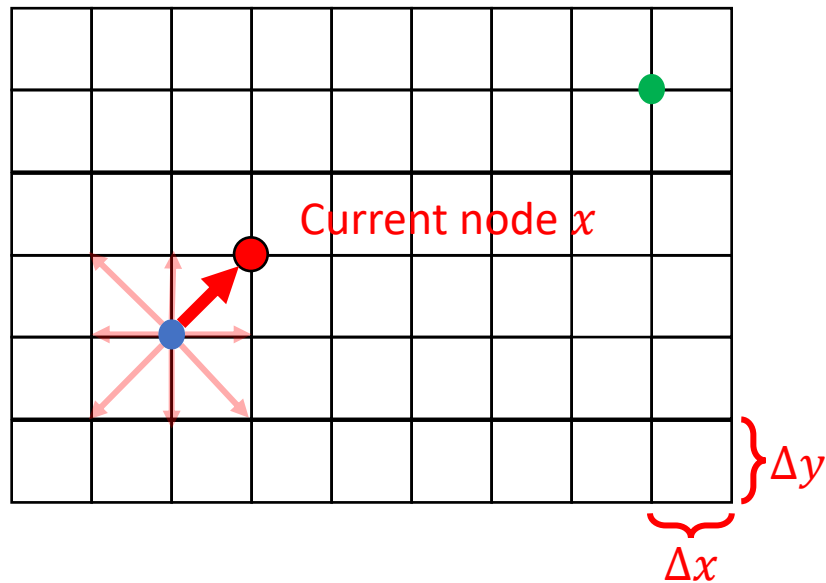
With

- 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



$$\text{Total cost: } f(x, y) = g(x, y) + h(x, y)$$

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)

● Goal node ● Start node

Path Planning Task for Out Flight

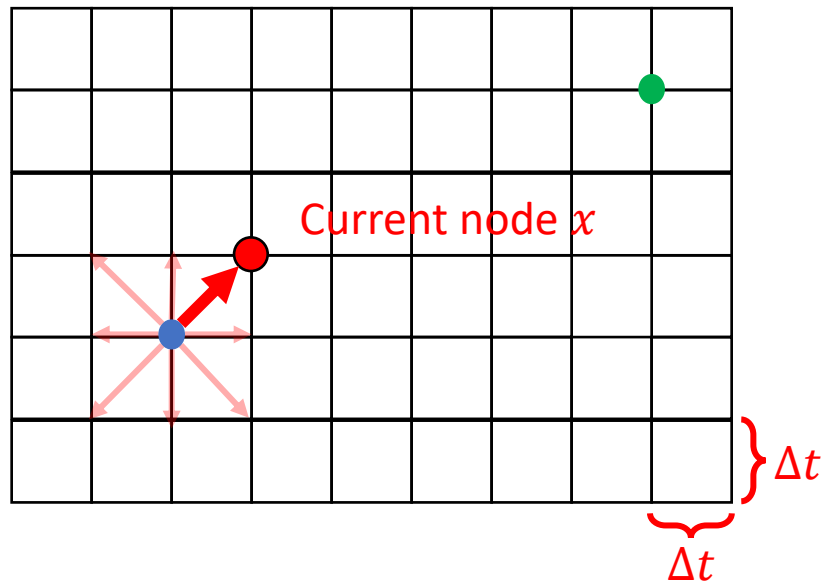
$$\text{Total cost: } f(n) = g(n) + h(n)$$

Traveling time (our case for flight)

Time cost on each step:

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

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

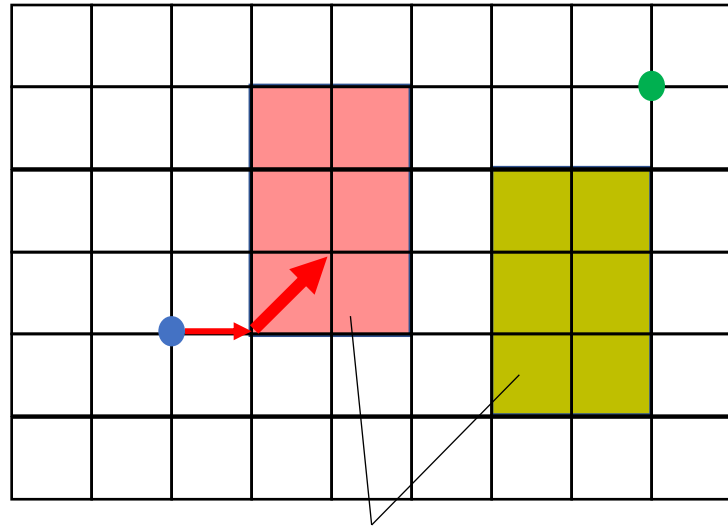


Example for first step:

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

● 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:

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

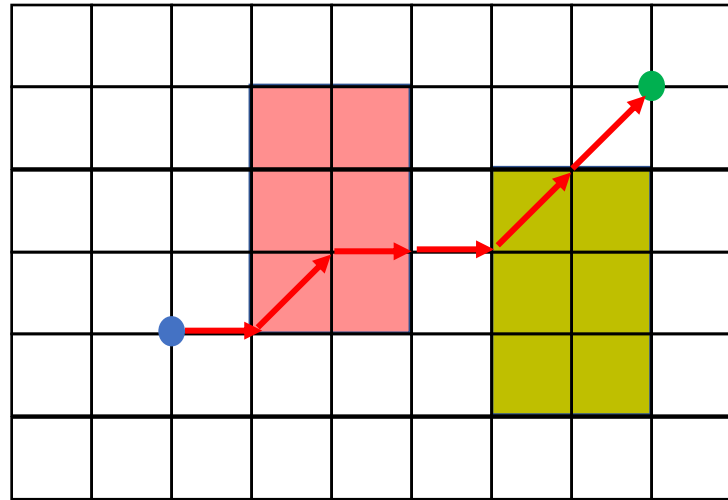
Previous cost
from start

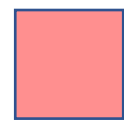
**Additional time
at current step**


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
➔

$$T_{best} = \min[f(goal)]$$

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

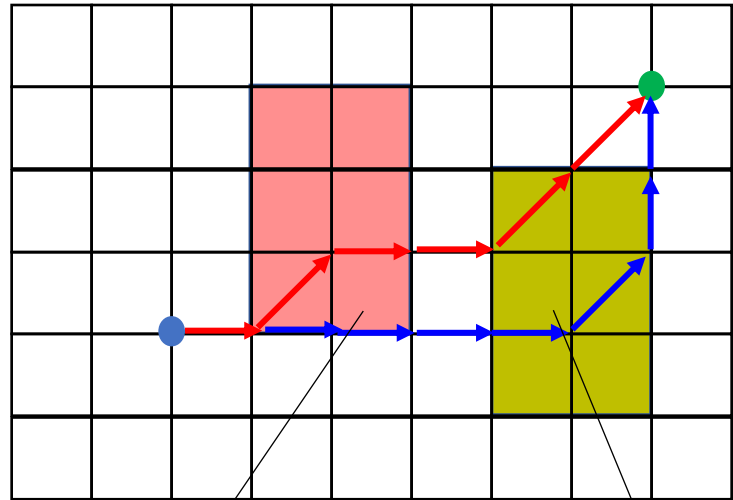
Additional cost on each node

Trip cost for planned path:

$$C = C_F \cdot \Delta F \cdot T_{best} + C_T \cdot T_{best} + C_c$$

● Goal node ● Start node α – additional time factor

How it choose different routes?



lower α ? huge α

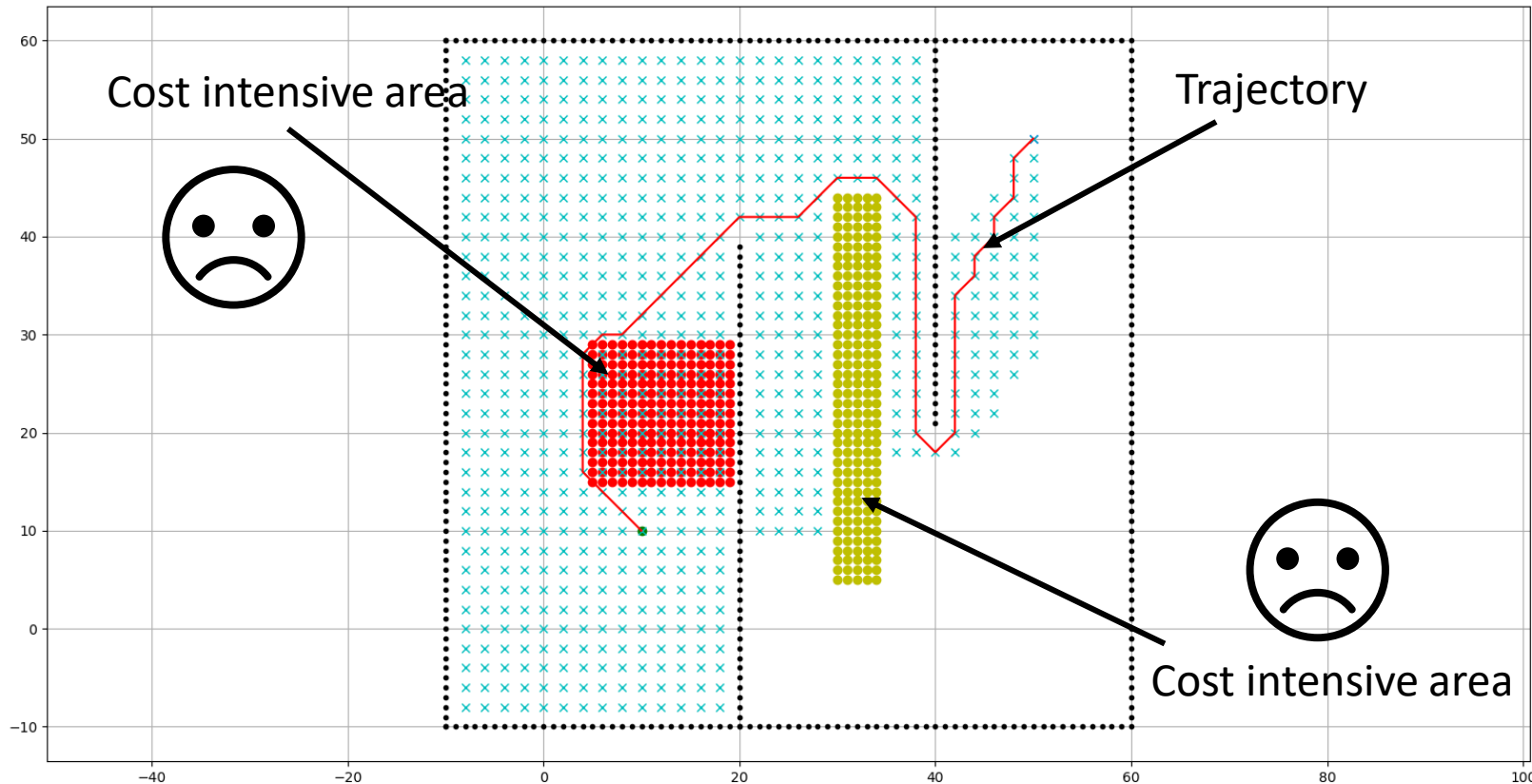
Cost Intensive Areas: The cost for flying through such area is **increased** due to airflow, legal restrictions and other reasons.

Time for the planned path:

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

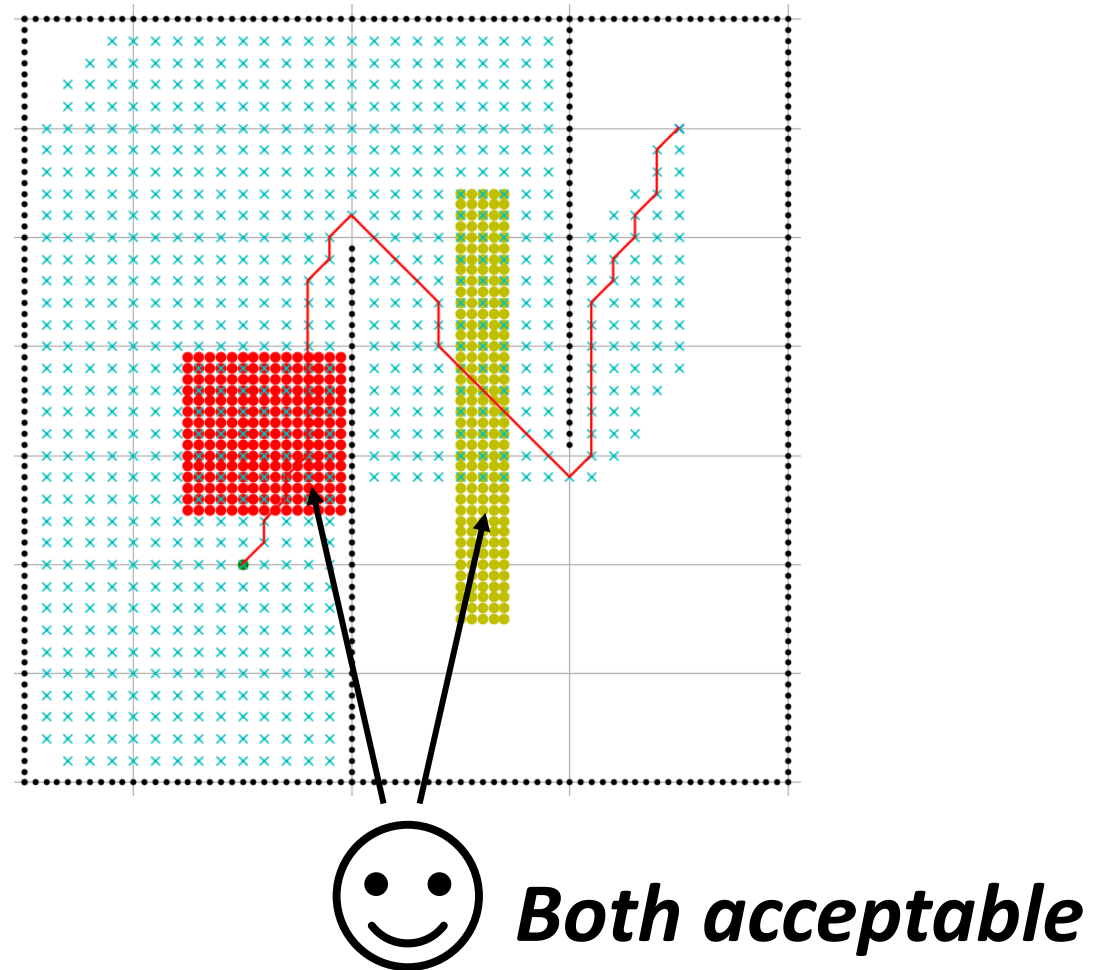
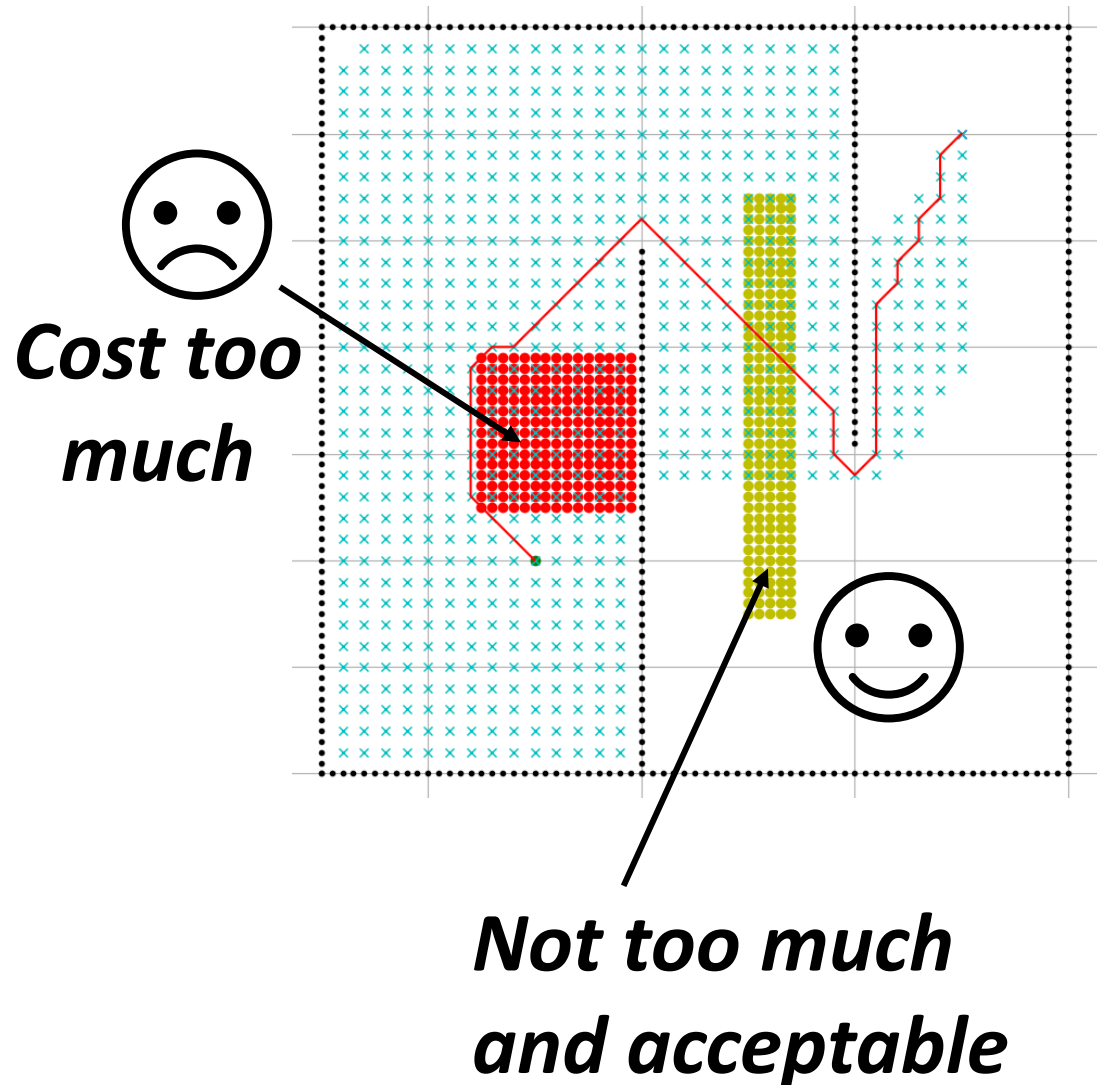
Depending on the extra time accumulated more specifically, the additional time factor

Example of Flight Path Planning



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

Example of Flight Path Planning



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