Robot Path Planning Algorithms

Dr. Essa Alghannam

CONTENTS

جامعة المنارة
MANARA UNIVERSITY

PART ONE

Path Planning

Path Planning

- Path planning: is to find an optimal path (smooth and feasible path) with the least cost on the road network graph. (A* and Dijkstra)

- **Trajectory planning is an essential task in mobile robot navigation that involves:**

1. generating a smooth and feasible path for the robot to follow from its current position A to a desired goal position B while avoiding obstacles (Path planning)
2. adhering to constraints such as maximum velocity and acceleration between the given position A and position B.

- whereas motion control is the process of generating a sequence of control signals to follow the planned Trajectory.

Motion planning algorithm evaluation index

01

Completeness

Using this algorithm, all solved problems can be solved within a limited time

02

Optimality

Using this algorithm, the optimal path can be found (shortest distance, minimum time consumption, minimum energy consumption, etc.)

PART TWO

Bug algorithm

Bug algorithm 1

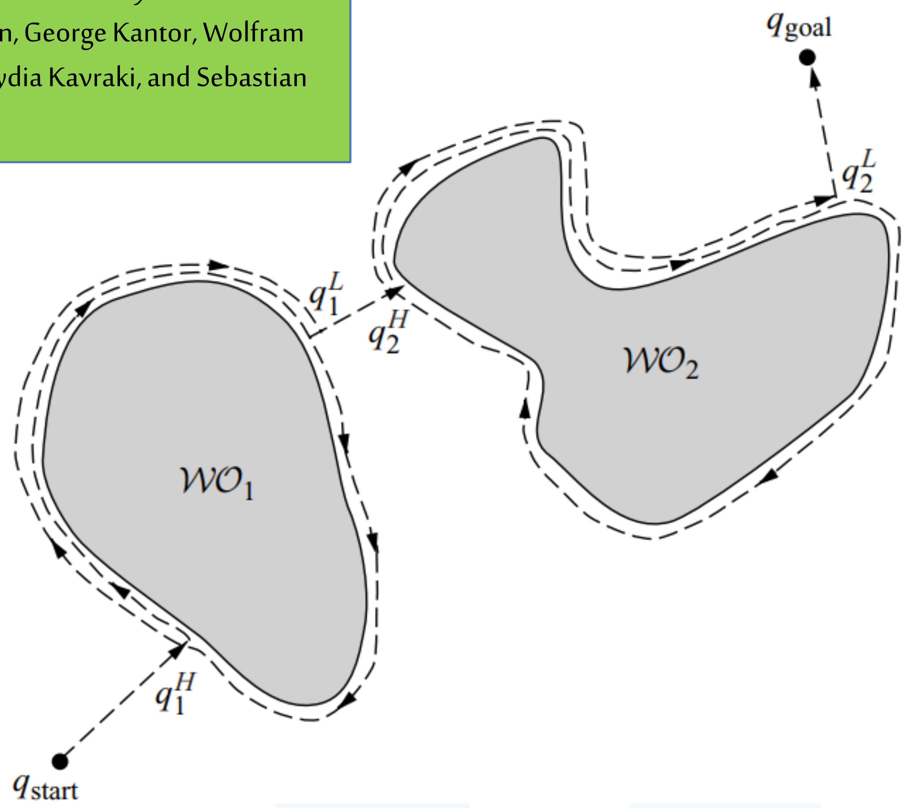
جامعة المنارة
MANARA UNIVERSITY

Algorithm 1 Bug1 Algorithm

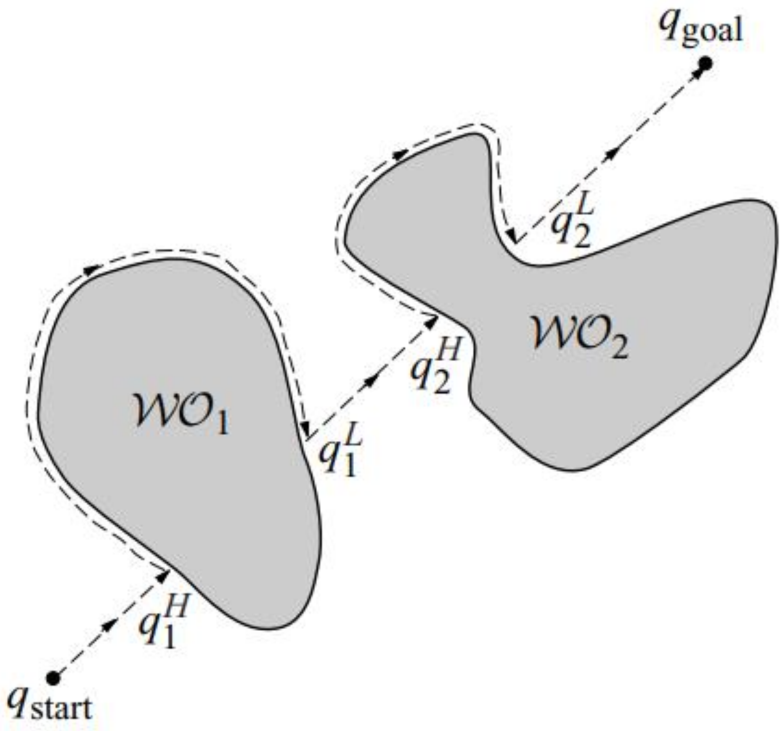
Input: A point robot with a tactile sensor
Output: A path to the q_{goal} or a conclusion no such path exists

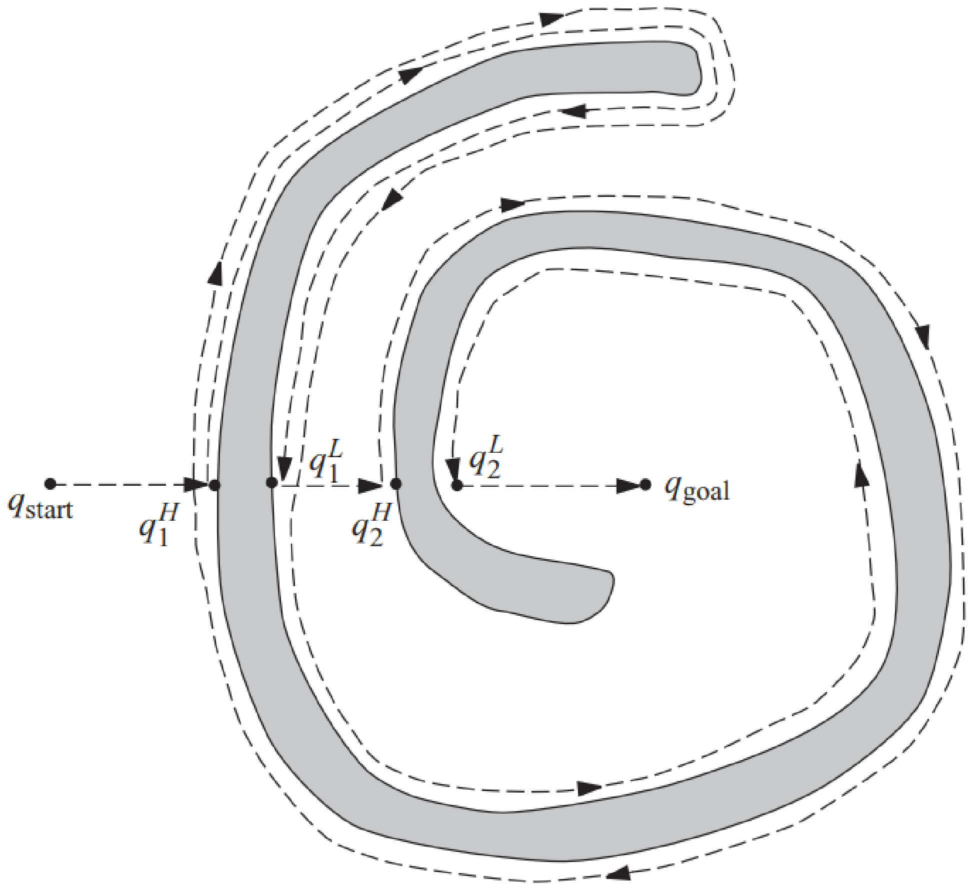
while Forever **do**
  repeat
    From q_{i-1}^L, move toward q_{goal}.
  until q_{goal} is reached **or** an obstacle is encountered at q_i^H.
  if Goal is reached **then**
    Exit.
  end if
  repeat
    Follow the obstacle boundary.
  until q_{goal} is reached **or** q_i^H is re-encountered.
  Determine the point q_i^L on the perimeter that has the shortest distance to the goal.
  Go to q_i^L.
  if the robot were to move toward the goal **then**
    Conclude q_{goal} is not reachable and exit.
  end if
end while

Principles of Robot Motion
Theory, Algorithms, and Implementation
Howie Choset, Kevin Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia Kavraki, and Sebastian Thrun



Bug algorithm 2





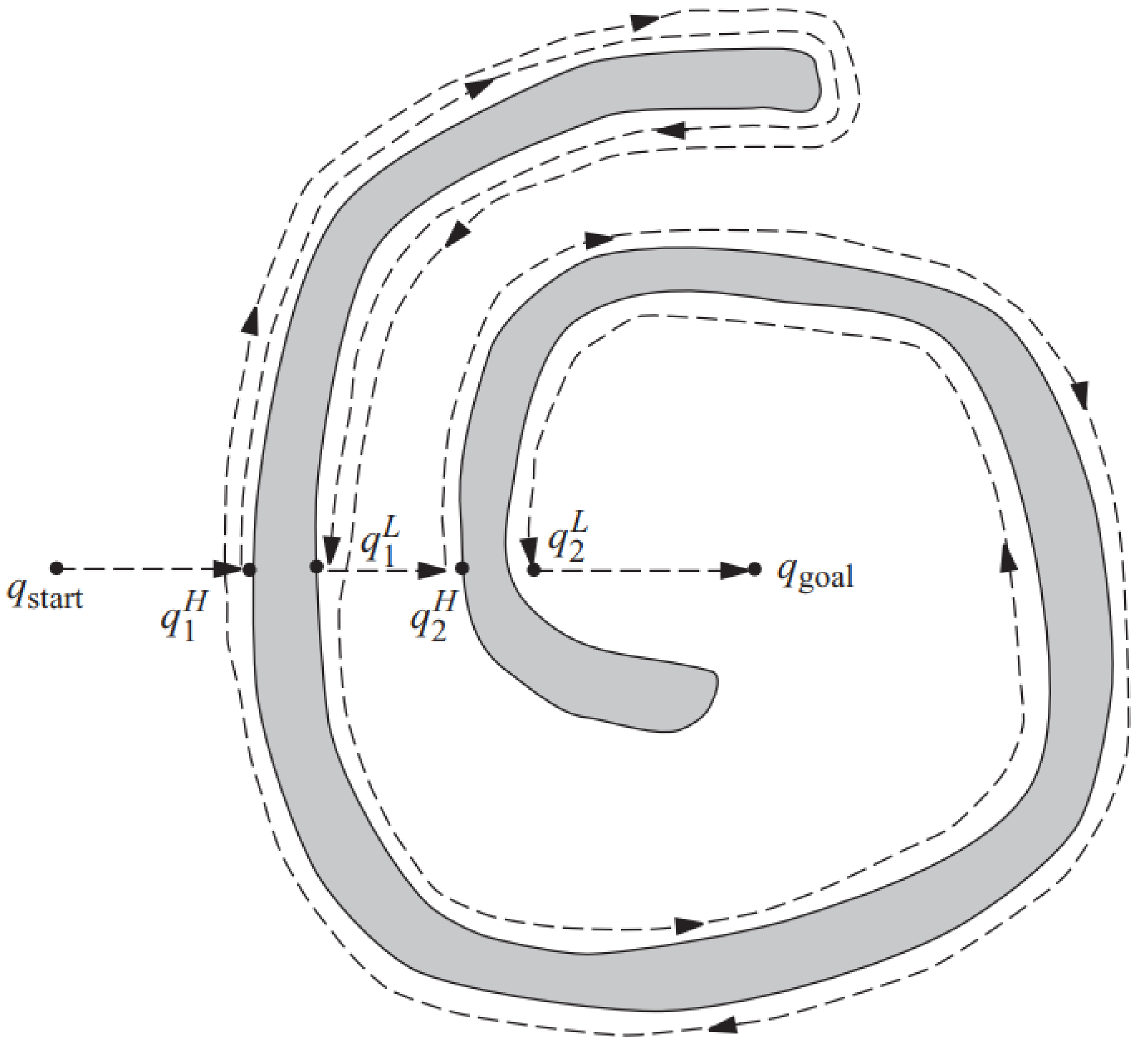
Bug algorithm 2

Algorithm 2 Bug2 Algorithm

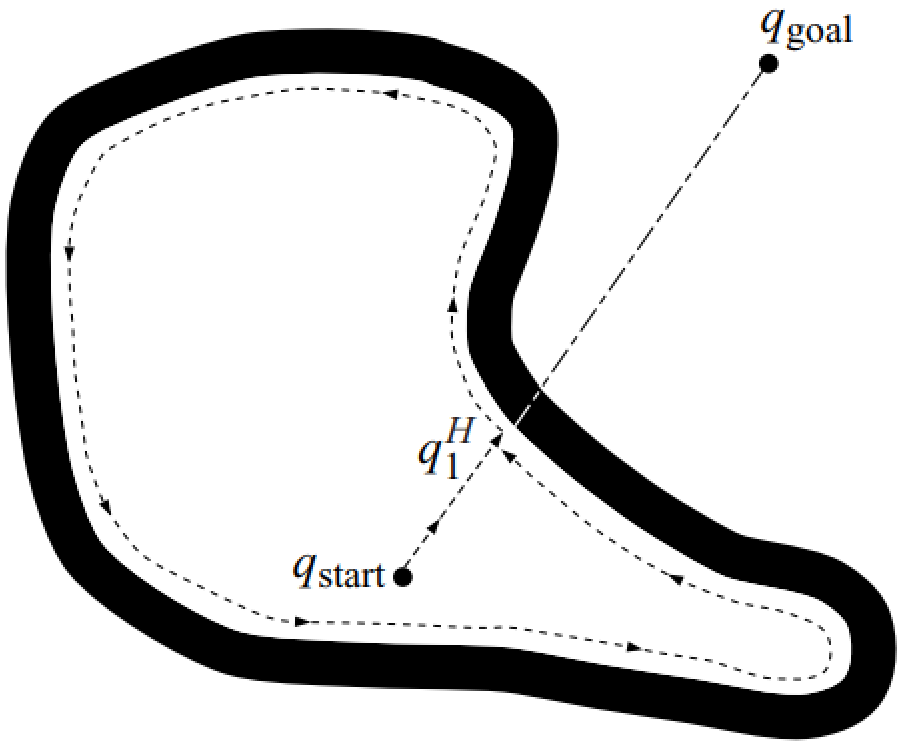
Input: A point robot with a tactile sensor

Output: A path to q_{goal} or a conclusion no such path exists

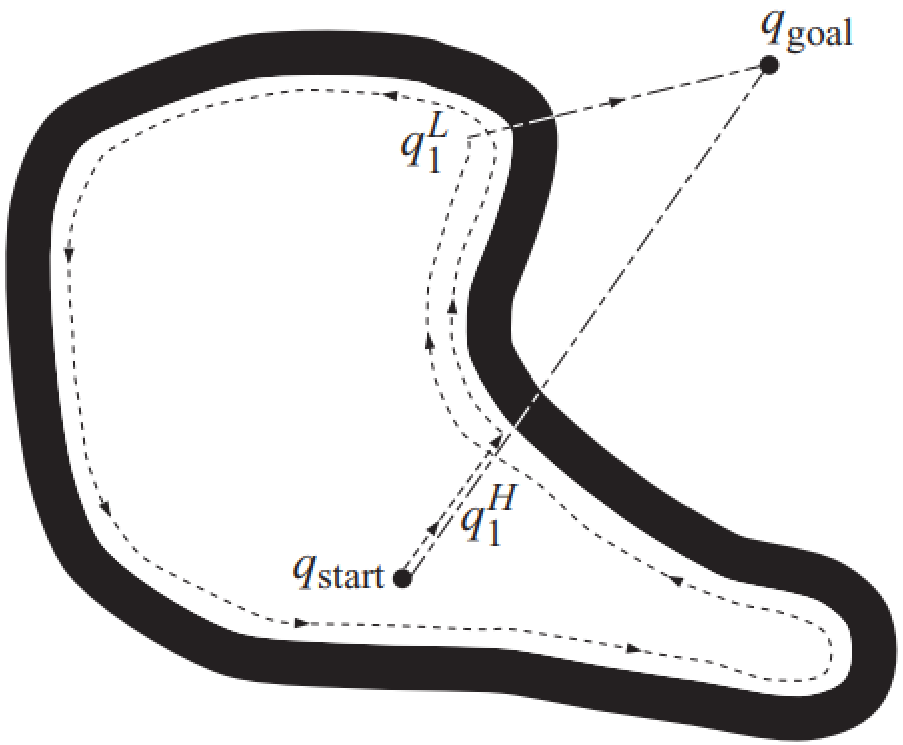
while True **do**
  repeat
    From q^L_{i-1}, move toward q_{goal} along m-line.
  until
  q_{goal} is reached **or**
  an obstacle is encountered at *hit point* q^H_i.
  Turn left (or right).
  repeat
    Follow boundary
  until
   q_{goal} is reached **or**
   q^H_i is re-encountered **or**
   m-line is re-encountered at a point m such that
    $m \neq q^H_i$ (robot did not reach the hit point),
    $d(m, q_{\text{goal}}) < d(m, q^H_i)$ (robot is closer), and
    if robot moves toward goal, it would not hit the obstacle
  if Goal is reached **then**
    Exit.
  end if
  if q^H_i is re-encountered **then**
    Conclude goal is unreachable
  end if
  Let $q^L_{i+1} = m$
  Increment i
end while



Bug algorithm 1 & 2



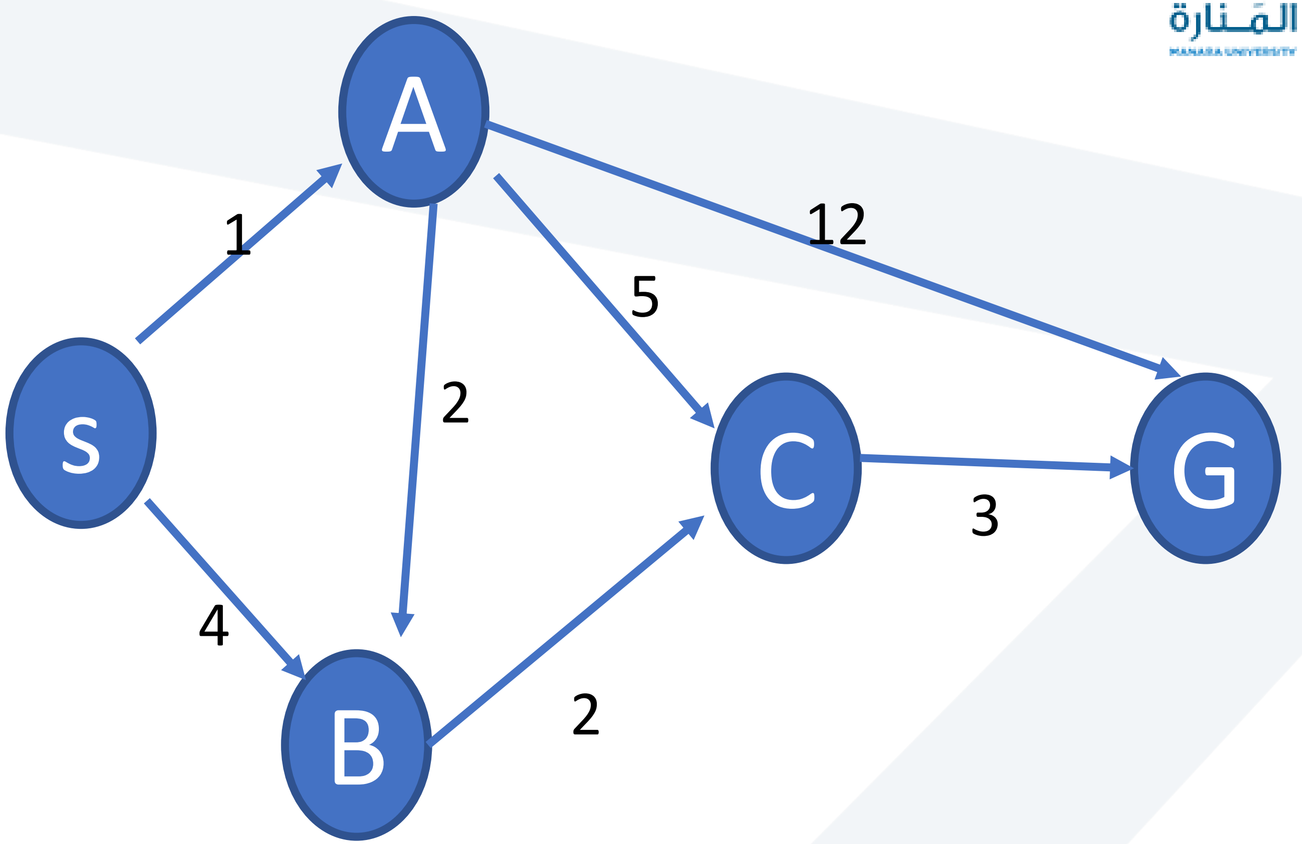
The Bug2 algorithm
reports failure.



The Bug1 algorithm reports the goal is unreachable

PART THREE

Dijkstra

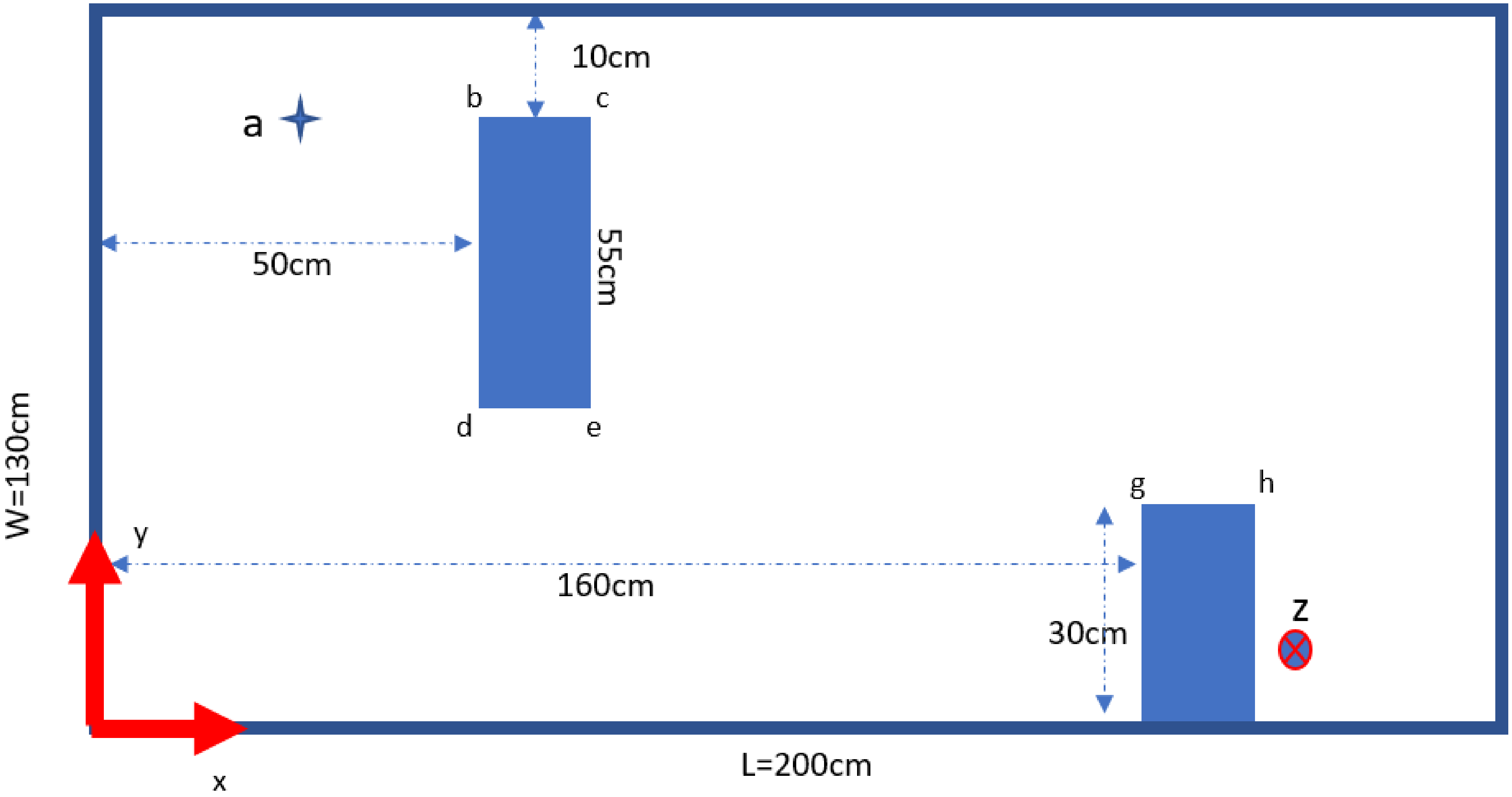


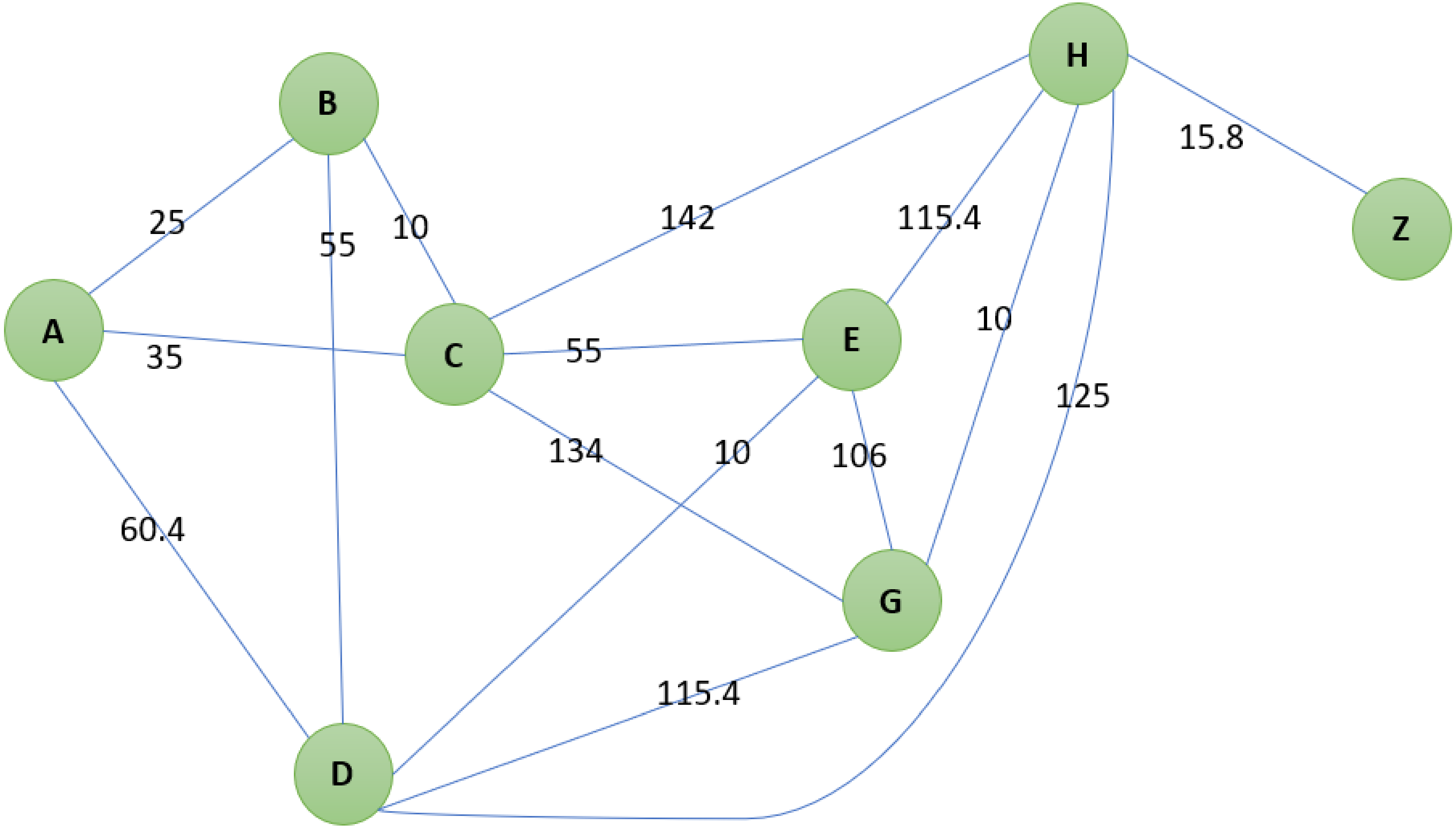
	S	A	C	B	G
S	-	1S	INF	4S	INF
A	-	-	6A	3A	13A
B	-	-	5B	-	13A
C	-	-	-	-	8C

$$S \rightarrow A \rightarrow B \rightarrow C \rightarrow G$$

مسألة

- لدينا بيئة مستطيلة الشكل موضحة الأبعاد جانبا W*L تحوي مجموعة من العوائق الساكنة باللون الأزرق التي تم تكبيرها بمقدار نصف قطر هيكل الروبوت واعتماد روبوت نقطي متنقل موجود في موضع بدائي في الفضاء الديكارتي $a(25cm,120cm,90^{o})$ ويريد التجول باتجاه موضع نهائي $z(175cm,15cm,0^{o})$ كما في الصورة التالية (مسقط رأسي):
- استخدم خوارزمية Dijkstra التقليدية لتوليد المسار من موضع البداية حتى النهاية


10cm
a
b
c
50cm
55cm
d
e
W=130cm
g
h
y
160cm
30cm
z
x
L=200cm



A
B
C
D
E
G
H
Z
25
55
10
35
60.4
142
55
134
10
115.4
10
106
125
15.8
115.4

- إذا كانت العقدة غير مُزاره بعد مثل **D, Z** في هذا السطر ولا يمكن الوصول لها من **العقدة الحالية مثل C هنا**، عندها نحافظ على أقل رقم حصلت عليه في العمود
- نضع خط صغير إذا كانت مُزاره سابقا مثل B, A, C
- إذا كان يمكن الوصول لها نحسب رقمها الجديد ونعتمد أصغر رقم في العمود نفسه مثل E, G , H

visited vertex	A	B	C	D	E	G	H	Z
A	0	INF	INF	INF	INF	INF	INF	INF
A	-	25A	35A	60.4A	INF	INF	INF	INF
B	-	-	35B	~~80B~~ 60.4A	INF	INF	INF	INF
C	-	-	-	60.4A	90C	169.5C	177.1C	INF
D	-	-	-	-	70.4D	~~60.4+115.4~~ 169.5C	~~60.4+125~~ 177.1C	INF
E	-	-	-	-	-	~~70.4+106~~ 169.5C	~~70.4+115.4~~ 177.1C	INF
G	-	-	-	-	-	-	~~169.5+10~~ 177.1C	INF
H	-	-	-	-	-	-	-	177.1+15.8= 192.8H

All vertex

Z – H – C – B –A

المسار المتولد هو الأقصر ولكن الخوارزمية تستغرق وقتا أطول لحسابه ولمعالجته. من الممكن تحسين أداء الخوارزمية بإضافة معامل البعد عن الهدف والبعد عن نقطة الانطلاق في الحسابات إضافة لأوزان الانتقال كما في *A.

Homework!

محاكاة للخوارزمية السابقة مع إمكانية التحكم
بأبعاد الخريطة وتموضع الحواجز ونقطتي الانطلاق والهدف

جامعة المنارة
MANARA UNIVERSITY

PART FOUR

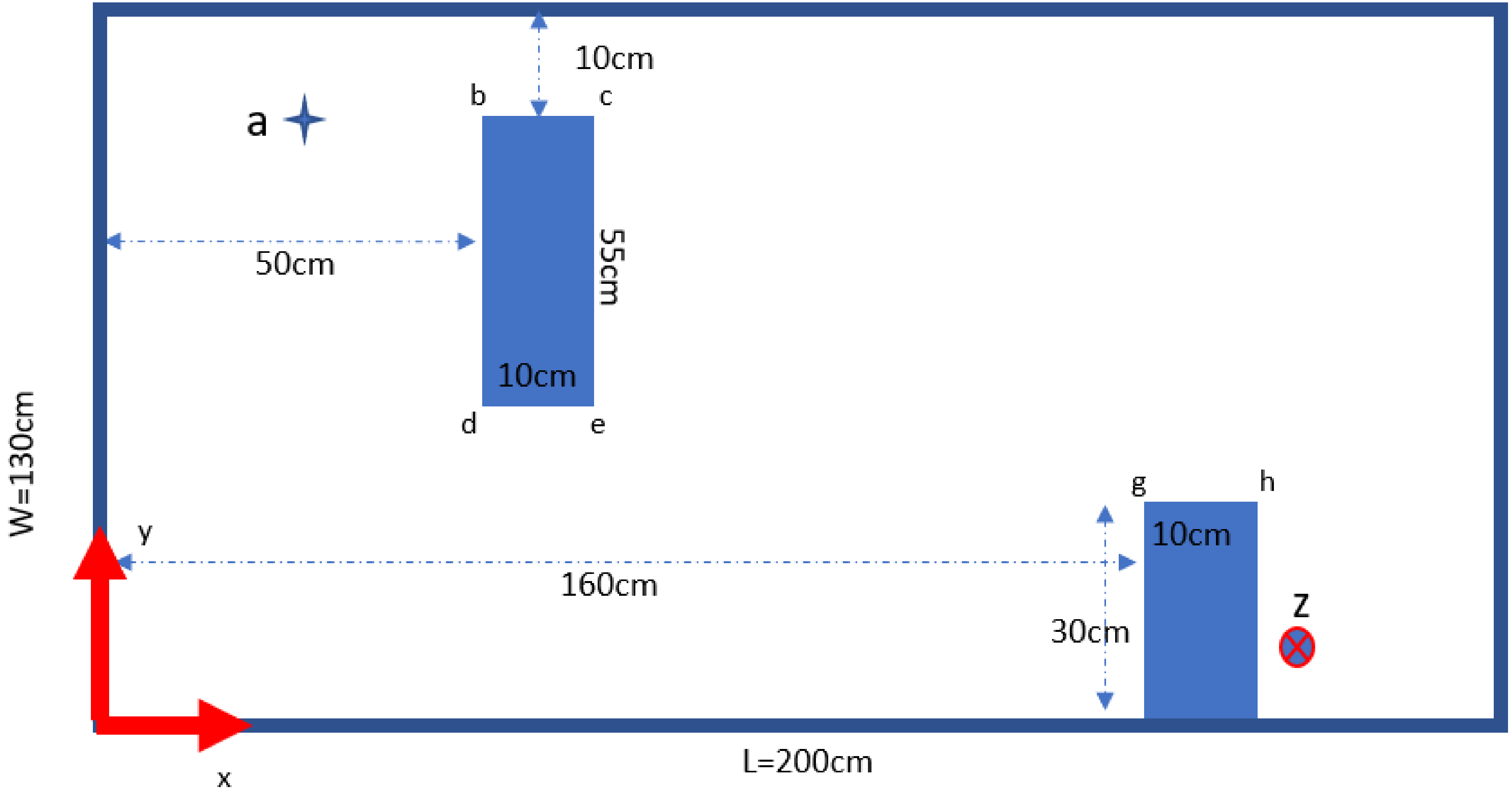
A-star

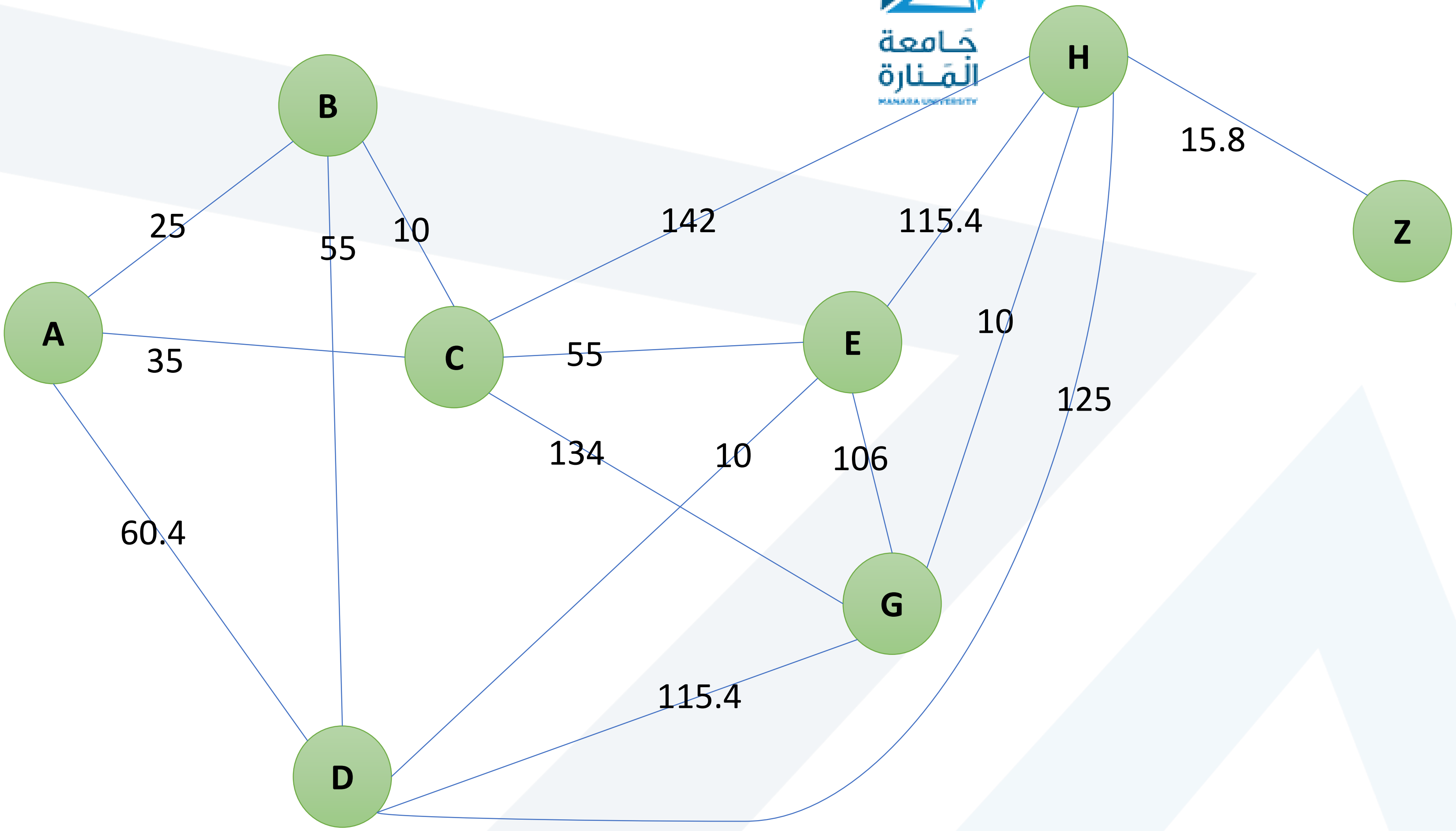
03 A-star

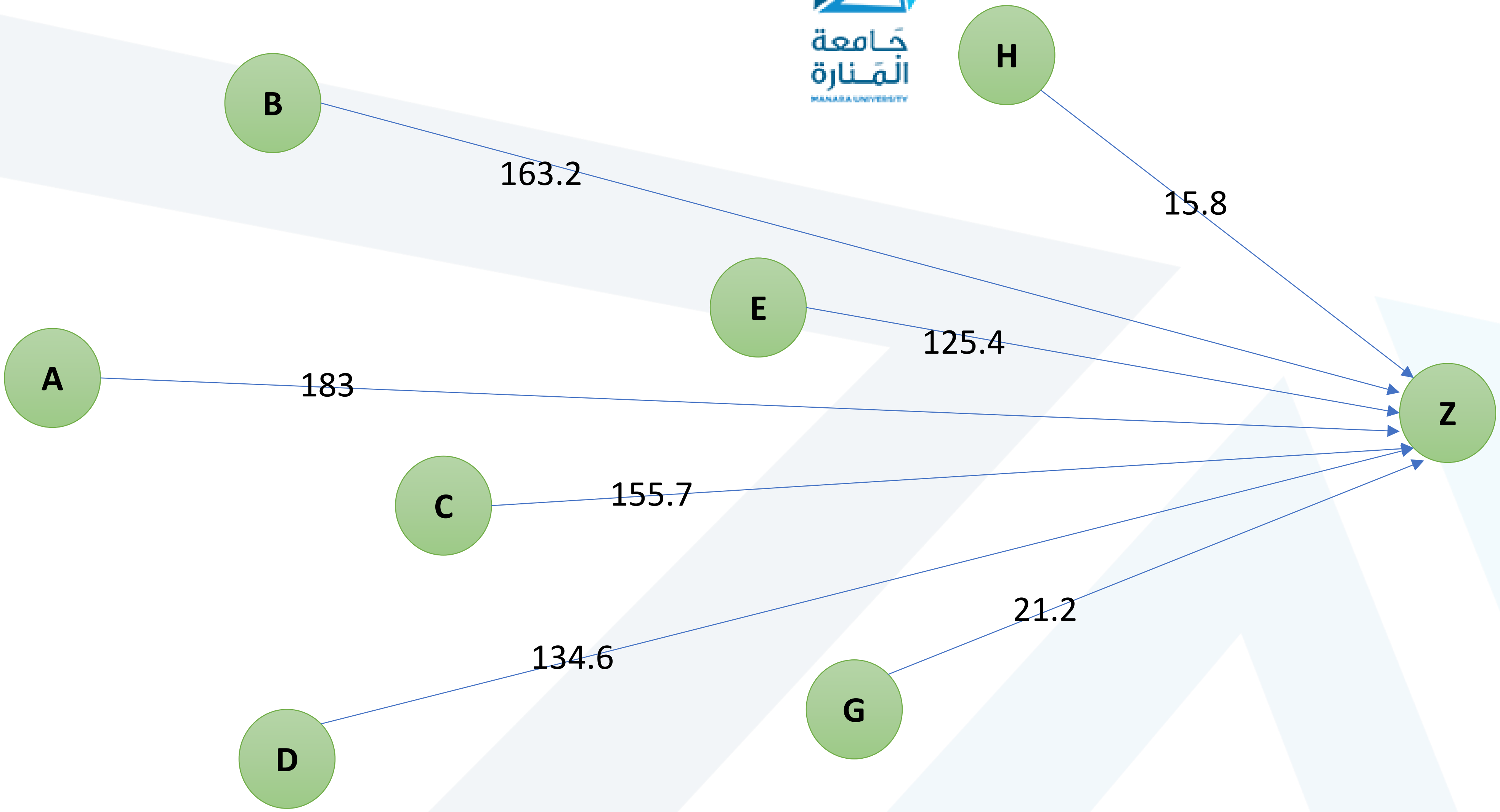
هي خوارزمية بحث تجريبية تعتمد على عاملين أساسيين هما:

- G(n) وهو كلفة الانتقال من عقدة إلى أخرى ويختلف هذا المتغير بين كل عقدتين
- H(n) هو التابع بين العقدة الحالية والهدف
- نعتمد على المتغير F من أجل الدمج بين العاملين حيث: F(n)هو مجموع المتغيرين G+H ويعبر عن أقل كلفة للانتقال من عقدة إلى العقدة التالية

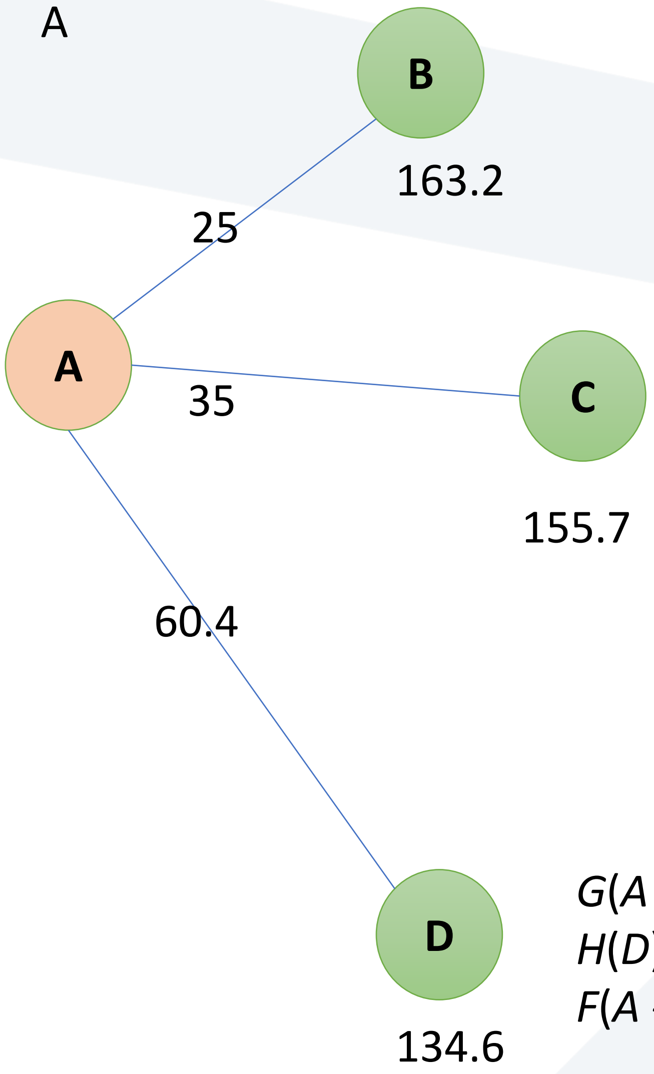
في المثال السابق استخدم A-star




B
H
15.8
Z
25
55
10
142
115.4
A
35
C
55
E
10
125
134
10
106
60.4
G
115.4
D


B
H
163.2
15.8
E
125.4
A
183
Z
C
155.7
21.2
134.6
G
D

A

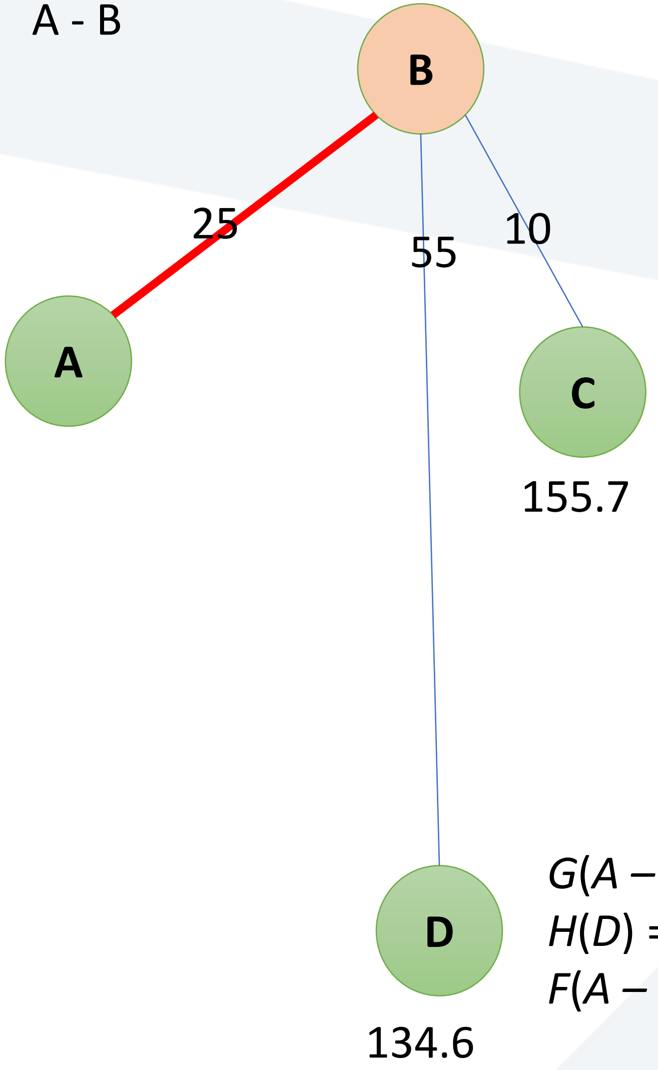


$G(A - B) = 25$
$H(B) = 163.2$
$F(A - B) = 188.2$

$G(A - C) = 35$
$H(C) = 155.7$
$F(A - C) = 190.7$

$G(A - D) = 60.4$
$H(D) = 134.6$
$F(A - D) = 195$

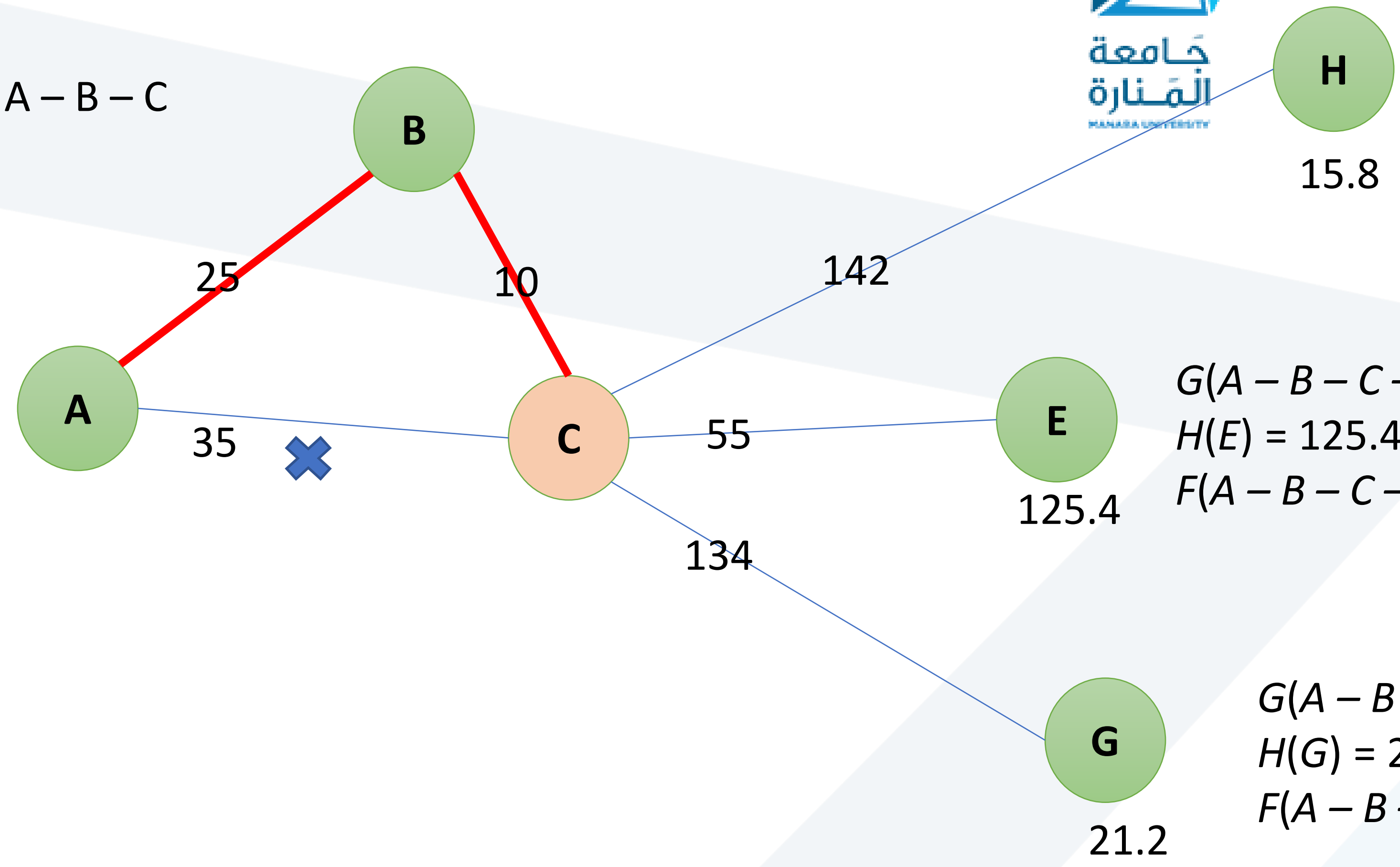
A - B



$G(A - B - C) = 35$
$H(C) = 155.7$
$F(A - B - C) = 190.7$

$G(A - B - D) = 80$
$H(D) = 134.6$
$F(A - B - D) = 214.6$

A – B – C



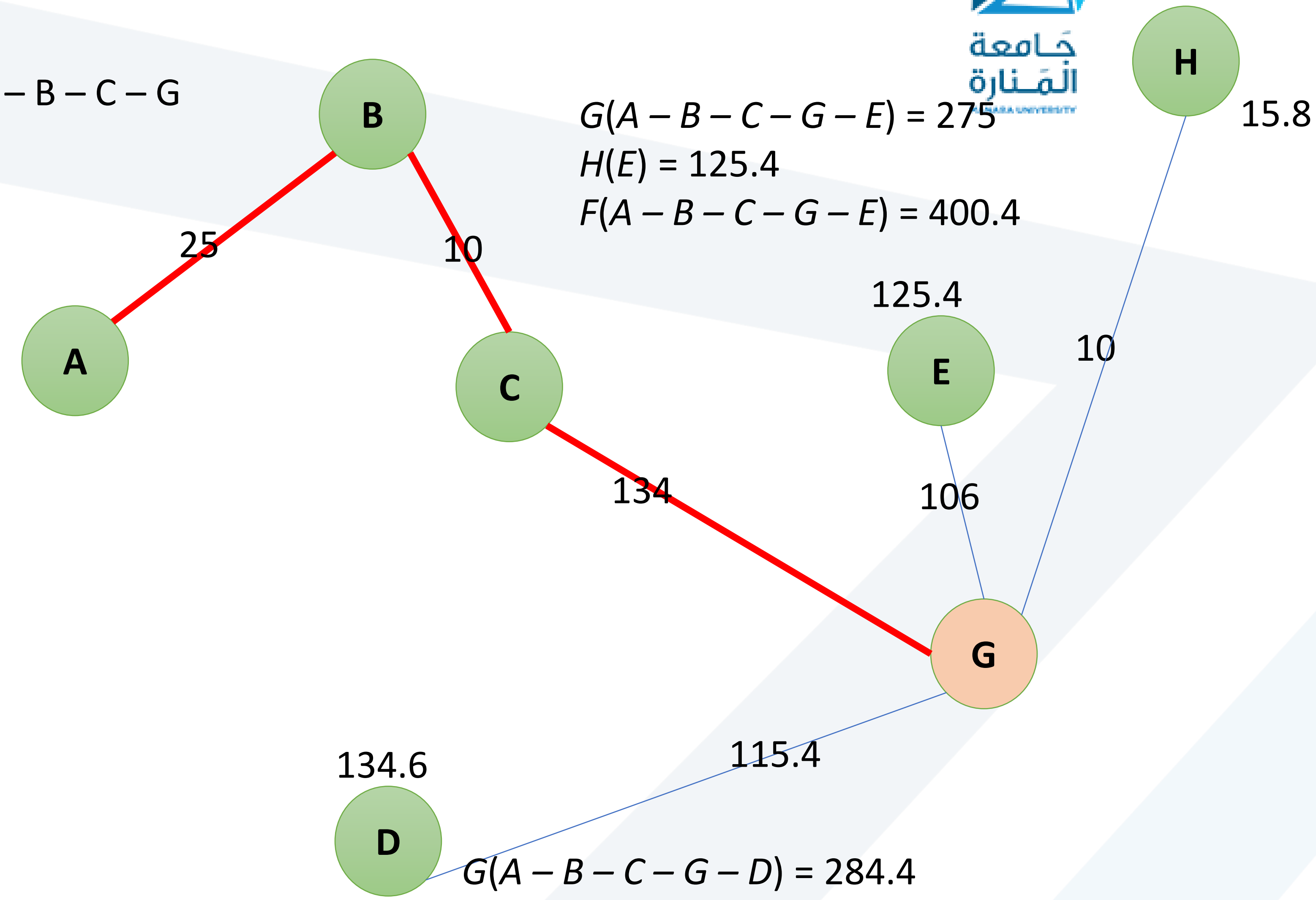
$G(A-B-C-H) = 177$
$H(H) = 15.8$
$F(A-B-C-H) = 192.8$

$G(A-B-C-E) = 90$
$H(E) = 125.4$
$F(A-B-C-E) = 215.4$

$G(A-B-C-G) = 169$
$H(G) = 21.2$
$F(A-B-C-G) = 190.2$

✖ Already visited

A – B – C – G



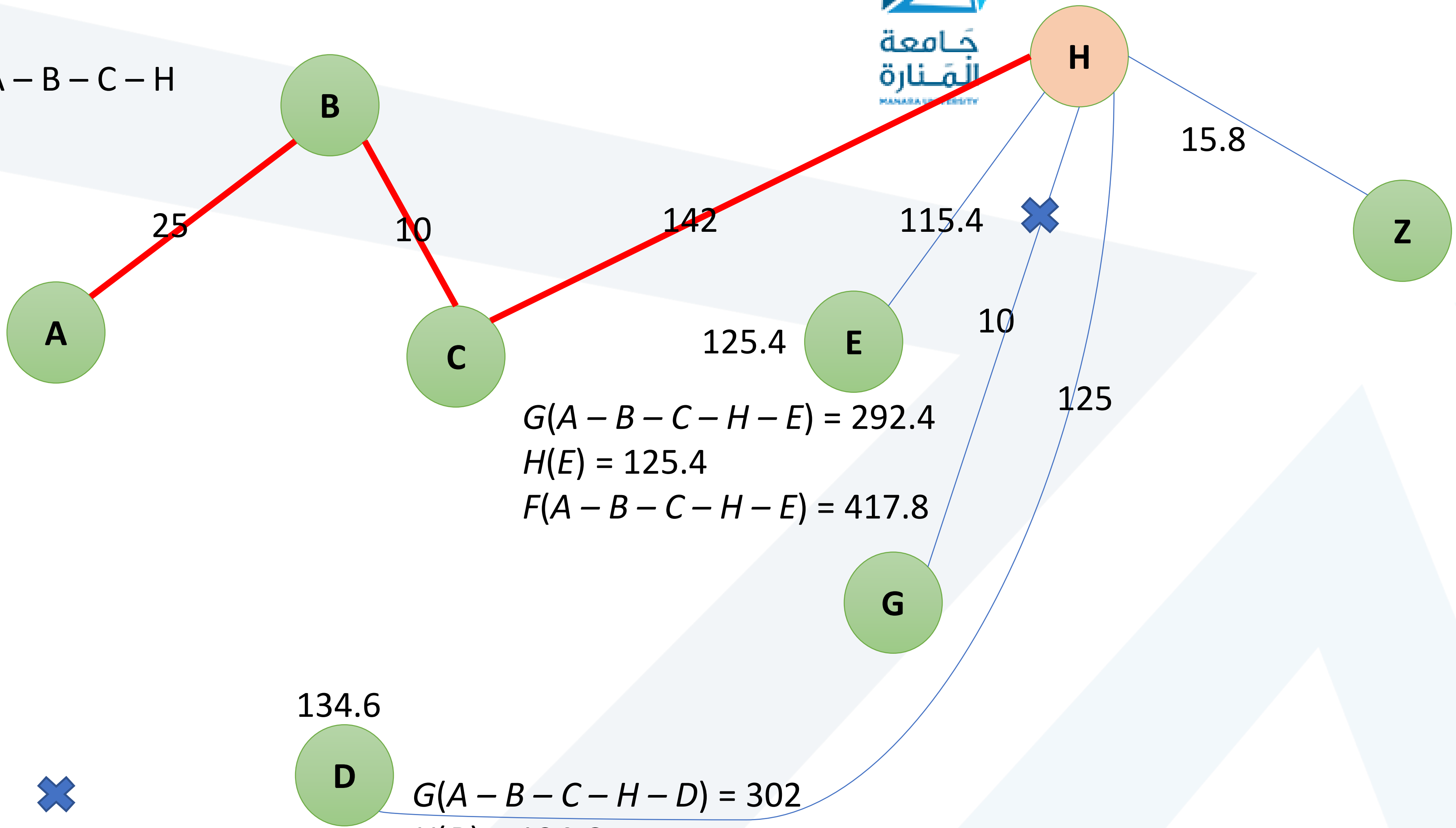
$G(A-B-C-G-E) = 275$
$H(E) = 125.4$
$F(A-B-C-G-E) = 400.4$

$G(A-B-C-G-H) = 179$
$H(H) = 15.8$
$F(A-B-C-G-H) = 194.8$ ✓

$G(A-B-C-G-D) = 284.4$
$H(D) = 134.6$
$F(A-B-C-G-D) = 419$

However H has been visited through C in the previous step. We go to the previous step and complete the path through C to H

A – B – C – H



$G(A-B-C-H-Z) = 192.8$

$H(Z) = 0$

$F(A-B-C-H-Z) = 192.8$

$G(A-B-C-H-E) = 292.4$

$H(E) = 125.4$

$F(A-B-C-H-E) = 417.8$

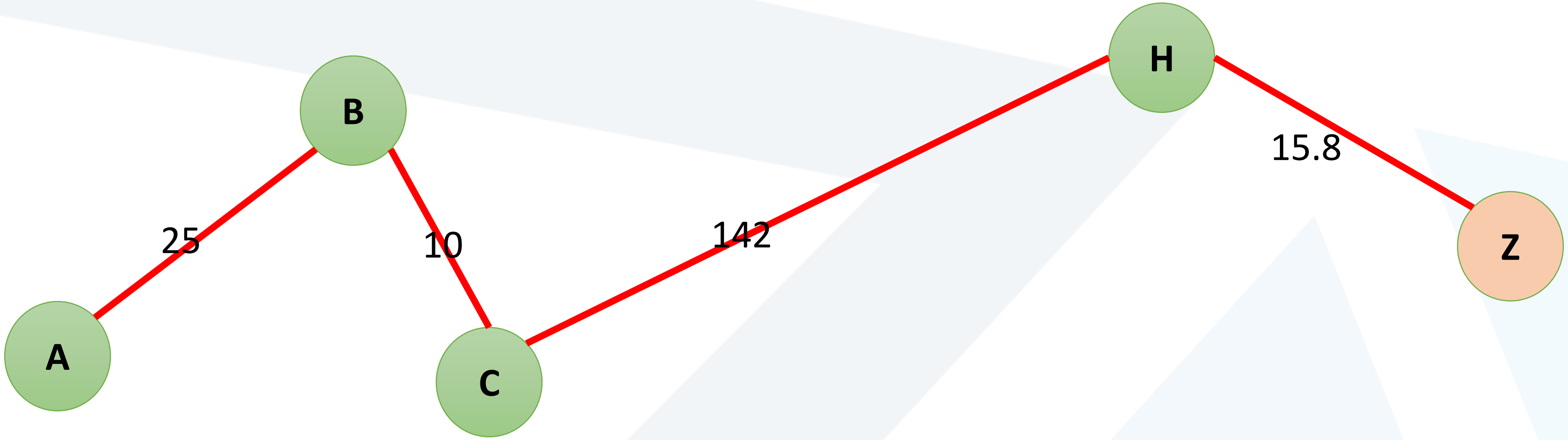
$G(A-B-C-H-D) = 302$

$H(D) = 134.6$

$F(A-B-C-H-D) = 436.6$

Already visited

A – B – C – H – Z



$G(A-B-C-H-Z) = 192.8$

$H(Z) = 0$

$F(A-B-C-H-Z) = 192.8$

Homework

- Write a Code to program it.
- Draw an environment with obstacles
- Detect the vertices.
- Detect the start and end point
- Detect the path

جامعة
المنارة
MANARA UNIVERSITY

PART FIVE

Artificial potential field method

Gradient of a Scalar Function

Say that we have a function, $f(x,y) = 3x^2y$. Our partial derivatives are:

$$\frac{\partial f(x,y)}{\partial x} = \frac{\partial}{\partial x}3x^2y = 6yx$$

$$\frac{\partial f(x,y)}{\partial y} = \frac{\partial}{\partial y}3x^2y = 3x^2$$

If we organize these partials into a horizontal vector, we get the **gradient** of $f(x,y)$, or $\nabla f(x,y)$:

$$[\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}] = [6yx, 3x^2]$$

What happens when we have two functions? Let's add another function, $g(x,y) = 2x+y^8$. The partial derivatives are:

$$\frac{\partial g(x,y)}{\partial x} = \frac{\partial}{\partial x}(2x + y^8) = 2$$

$$\frac{\partial g(x,y)}{\partial y} = \frac{\partial}{\partial y}(2x + y^8) = 8y^7$$

So the gradient of g(x,y) is:

$$\nabla g(x,y) = [\frac{\partial g(x,y)}{\partial x}, \frac{\partial g(x,y)}{\partial y}] = [2, 8y^7]$$

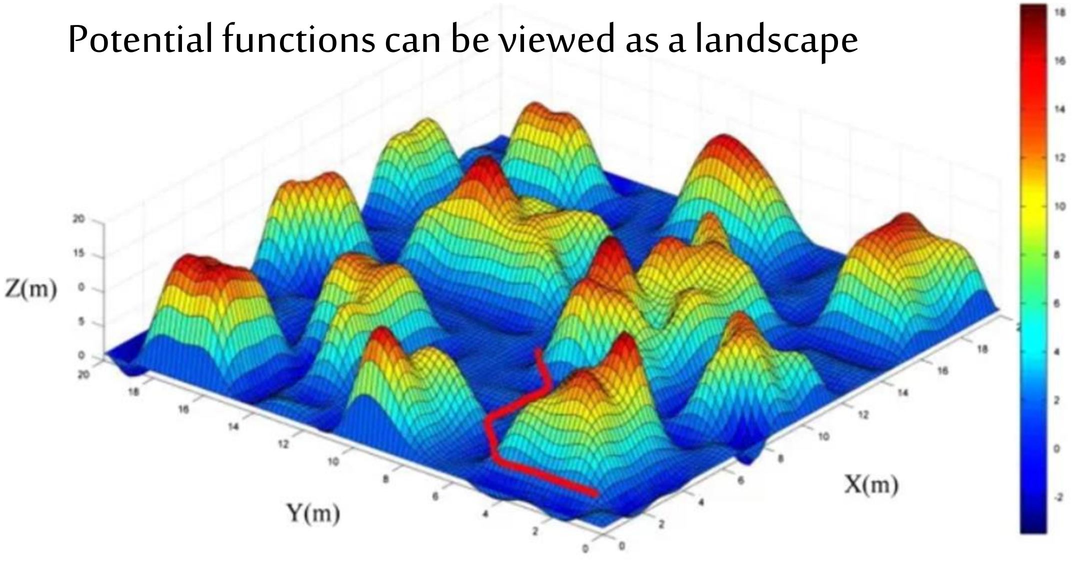
Gradient of a Vector Function

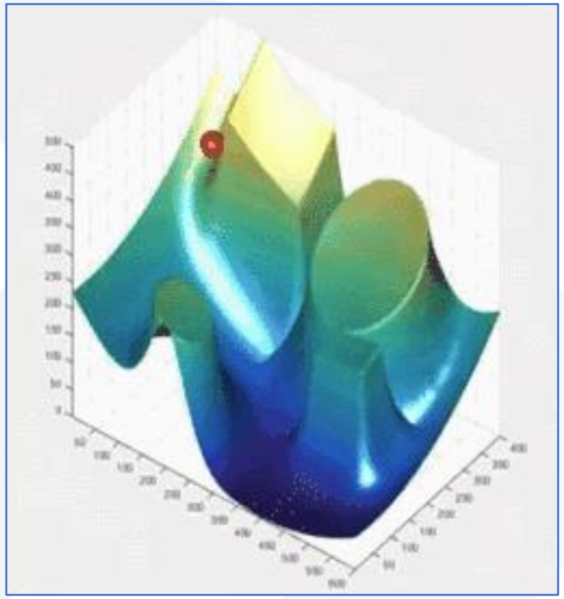
Now that we have two functions, how can we find the gradient of both functions? If we organize both of their gradients into a single matrix, we move from vector calculus into matrix calculus. This matrix, and organization of the gradients of multiple functions with multiple variables, is known as the **Jacobian matrix.**

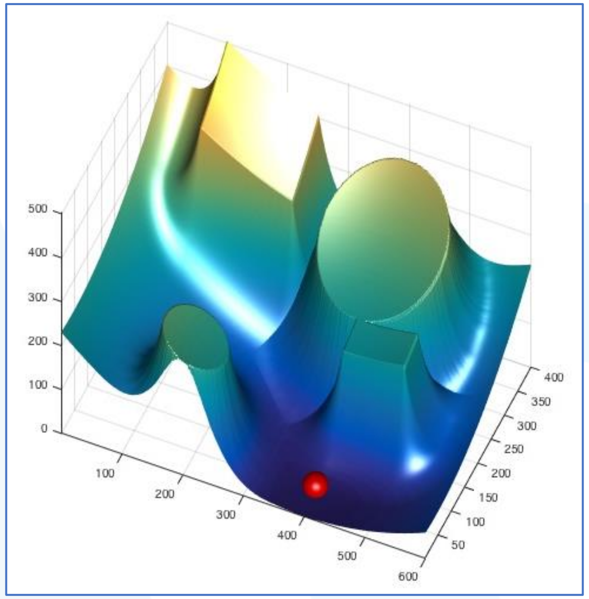
$$J = \begin{bmatrix} \nabla f(x,y) \\ \nabla g(x,y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial f(x,y)}{\partial y} \\ \frac{\partial g(x,y)}{\partial x} & \frac{\partial g(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} 6yx & 3x^2 \\ 2 & 8y^7 \end{bmatrix}$$

05 Artificial potential field method (incomplete and not optimal)

Potential functions can be viewed as a landscape



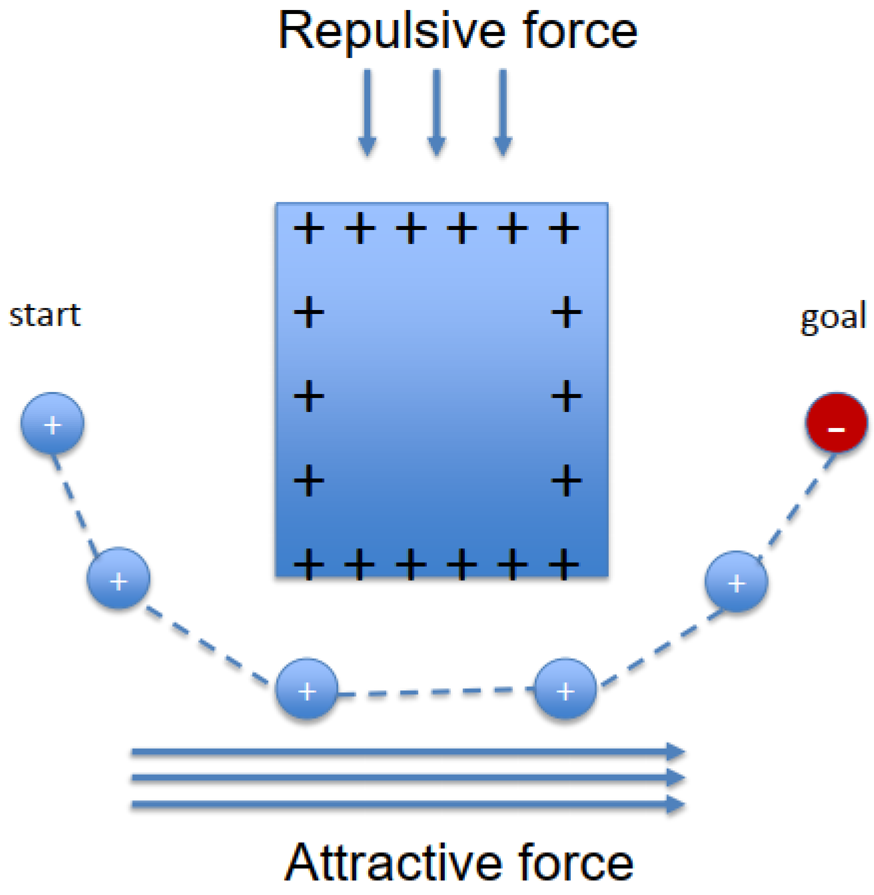




- The idea of a potential field is taken from nature.
- For instance, a small ball rolling in a hill.
- The idea is that depending on the hill's slope, the ball can arrive at the source of the field the valley in this example.

Artificial Potential Field Method

a charged particle navigating a magnetic field



A potential function is a function that may be viewed as energy. The gradient of the energy is force.

- Potential function guides the robot as if it were a particle moving in a gradient field.
- Analogy: robot is positively charged particle, moving towards negative charge goal. attractive force
- Obstacles have "repulsive" positive charge. same charge as robot – repelling force

The idea is that depending on **the strength of the field ,** the particle can arrive at the source of the field, the magnet in this example.

Attractive Field + Repulsive Field = Resultant sum

Attractive Field

7	7	6	6	7	7	8
6	6	5	5	6	6	7
4	5	4	4	5	6	6
5	2	1	1	2	5	6
4	1	Goal		1	4	6
4	1			1	4	6
5	2	1	1	2	5	6
4	5	4	4	5	6	6

Repulsive Field

0	0	10	70	100	100	70
0	0	10	100	Obstacle		100
0	0	10	100			100
0	0	10	100			100
0	0	10	70	100	100	70
0	0	0	10	10	10	10
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Resultant sum

7	7	16	76	107	107	78
6	6	15	105	Obstacle		107
4	5	14	104			106
5	2	11	101			106
4	1	Goal		101	104	76
4	1			11	14	16
5	2	1	1	2	5	6
4	5	4	4	5	6	6

The target point is at the bottom. Attractive Potential Function is distance from goal.

The farther away from the target point, the higher the attractive potential field (large potential energy), goal state has zero potential energy.

↑ Green cells with "lowest" value of potential (most possible robot paths to Goal).

At the same time, in order to avoid collision, a repulsive potential field is added around the obstacle (the potential energy is maximum at the obstacle). Obstacles create high energy.

Robot "attracted" to the goal and "repelled" from the obstacles.

Path of robot is from state of high energy to low (zero) energy at the goal. Robot moves from high-value to low-value Using a "downhill" path (i.e negative of the gradient). This is known as gradient descent –follow a functional surface until you reach its minimum. Gradient descent follows energy minimization path to goal so Path is negative gradient, largest change means high energy

Robot transition to lowest value of potential field.

5	2	22	25	16
7	19	28	36	6
4	11	**Robot**	43	16
15	29	21	54	26
9	17	15	16	6

In our case robot has four possible motion transitions (choices to move) forward, back, left and right. Before transit to the next cell (discrete position) the robot check the cell with lowest value of the potential. The process repeats while the robot achieves a destination.

جَامعة
المَنارة

Artificial Potential Field algorithm can be easily explained by dividing the main idea for two sub-tasks:

- the robot space (here XY plan) has to be divided by a grid of certain discrete positions of the robot, where potential field can be computed at each: q=(x,y)

- For each discrete position of robot q=(x,y),

the artificial potential field consists of two different components:

1- Attractive field of the destination

2- repulsive field of obstacles

They are sum up to compute the potential field acting on the robot:

$$U(q) = U_{att}(q) + U_{rep}(q)$$

Total Potential Function

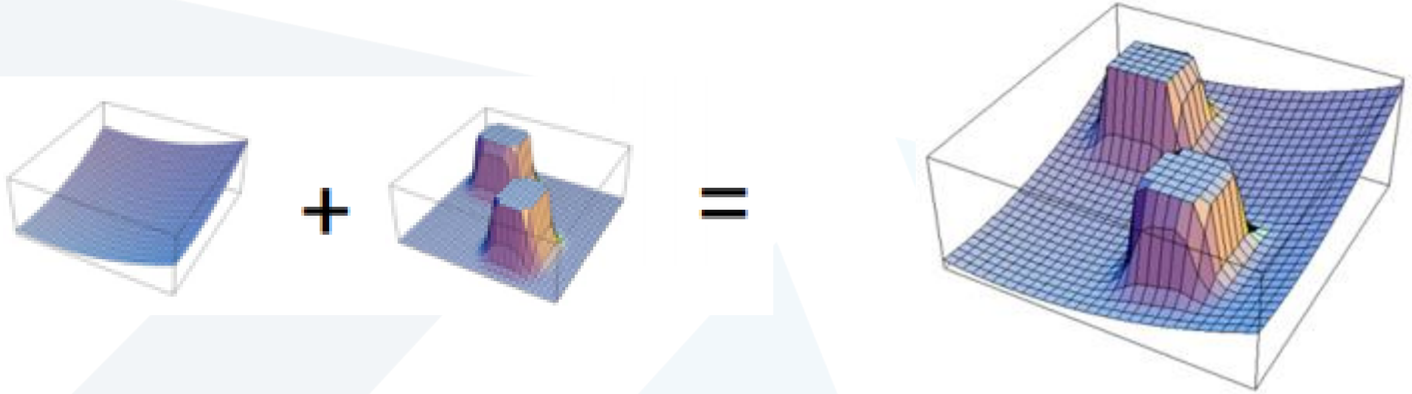
محصلة الحقول

– U_{att} is the “attractive” potential --- move to the goal

– U_{rep} is the “repulsive” potential --- avoid obstacles

The artificial force acting on the robot at the position:

$$F(q) = -\nabla U(q)$$



where $\nabla U(q)$ donetes the gradient vector of U at position q.

velocity becomes the gradient

$$\nabla U = \begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \end{bmatrix}$$

Computing Attractive Potential

- حقل الجذب Attractive field ترتبط بشكل مباشر بقيمة المسافة الإقليدية ما بين نقطة الوقوف الحالية للروبوت و نقطة الهدف
- وهي تتناقص بشكل تدريجي كلما اقترب الروبوت من الهدف

Quadratic Potential

$$U_{att}(q) = \frac{1}{2} k_{att} \rho_{goal}^2$$

whose gradient is

$$\nabla U_{att}(q) = \nabla \left(\frac{1}{2} k_{att} d^2(q, q_{goal}) \right)$$

$$F_{att}(q) = k_{att} (q, q_{goal})$$

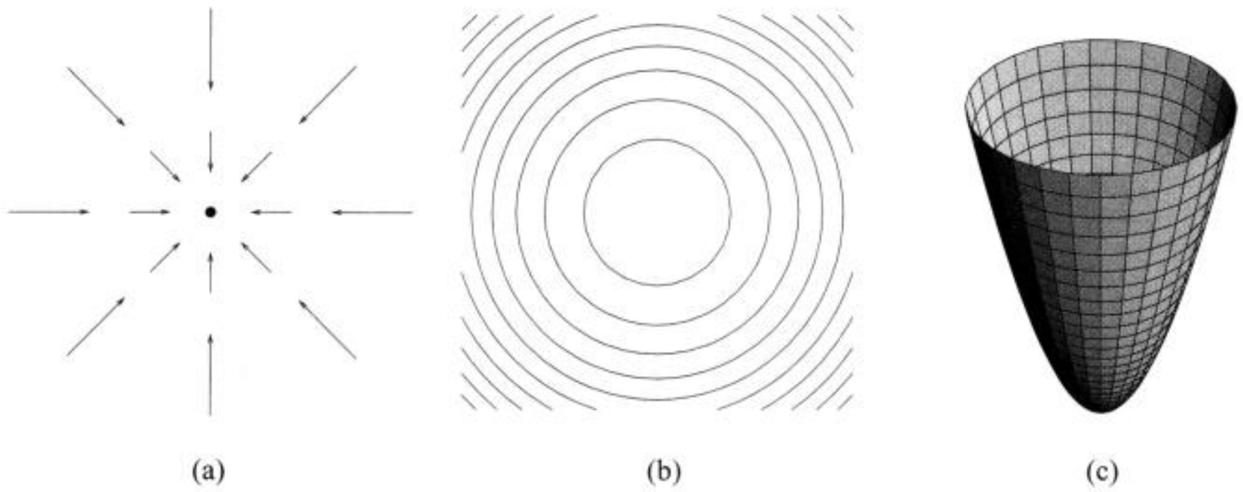


Figure 4.4 (a) Attractive gradient vector field. (b) Attractive potential isocontours. (c) Graph of the attractive potential.

where k_{att} positive scaling factor and ρ_{goal} denotes the Euclidean distance.

Computing Repulsive Potential

The repulsive field "generated" by obstacles can be computed according to the following formula. As expected, the value increases, while the robot approaches to obstacle.

$$U_{rep} = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{if } \rho(q) \geq \rho_0 \end{cases}$$

where k_{rep} is a positive scaling factor, $\rho(q)$ denotes as a minimal distane from q to the object and

ρ_0 the distance of influcence of the object.

Strength of repulsive force should increase as we near the obstacle

- Compute potential in terms of distance to closest obstacle
- Multiple obstacles: compute repulsive potential over all obstacles

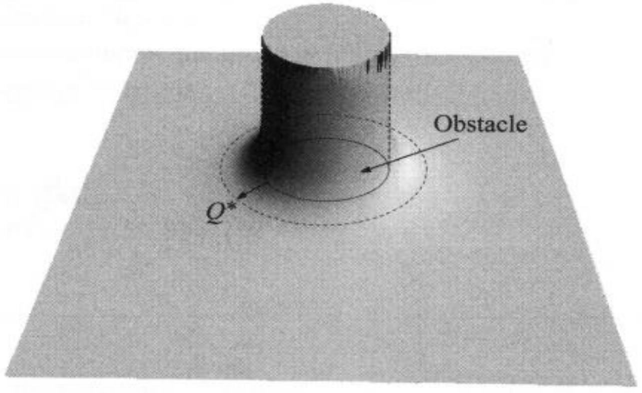


Figure 4.5 The repulsive gradient operates only in a domain near the obstacle.

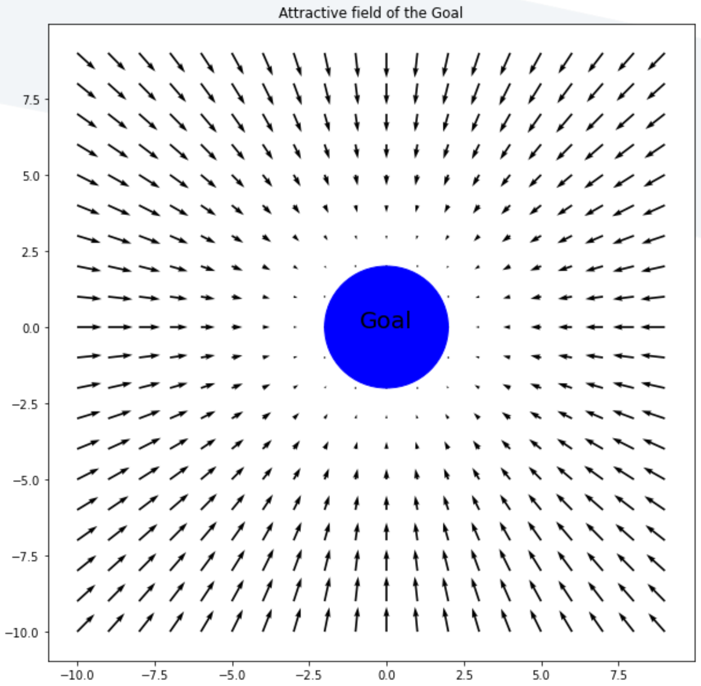
جامعة المنارة
MANARA UNIVERSITY

$$U_{rep} = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{if } \rho(q) \geq \rho_0 \end{cases}$$

whose gradient is

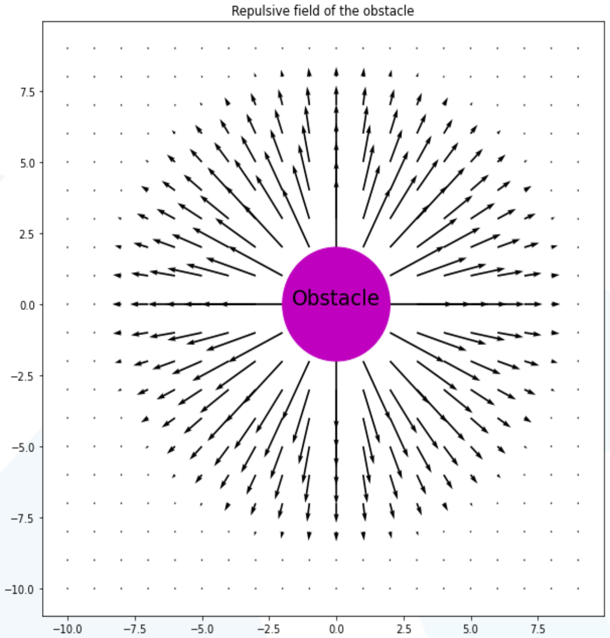
$$\nabla U_{\text{rep}}(q) = \begin{cases} k_{rep} \left(\frac{1}{\rho_0} - \frac{1}{\rho(q)} \right) \frac{1}{\rho^2(q)} \nabla \rho(q), & \rho(q) \leq \rho_0 \\ 0, & \rho(q)) > \rho_0 \end{cases}$$

Example:



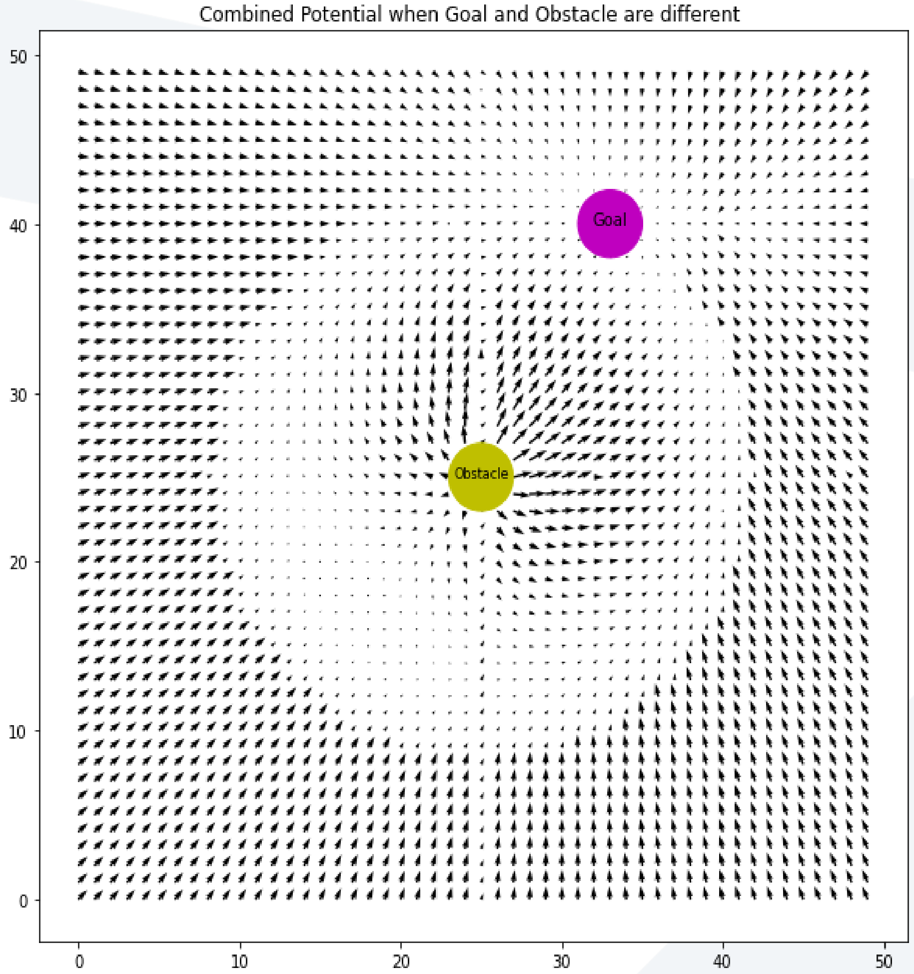
- We can also define another behavior that allows the robot to avoid obstacles. We make each obstacle generate a repulsive field around it. If the robot approaches the obstacle, a repulsive force will act on it, pushing it away from the obstacle.

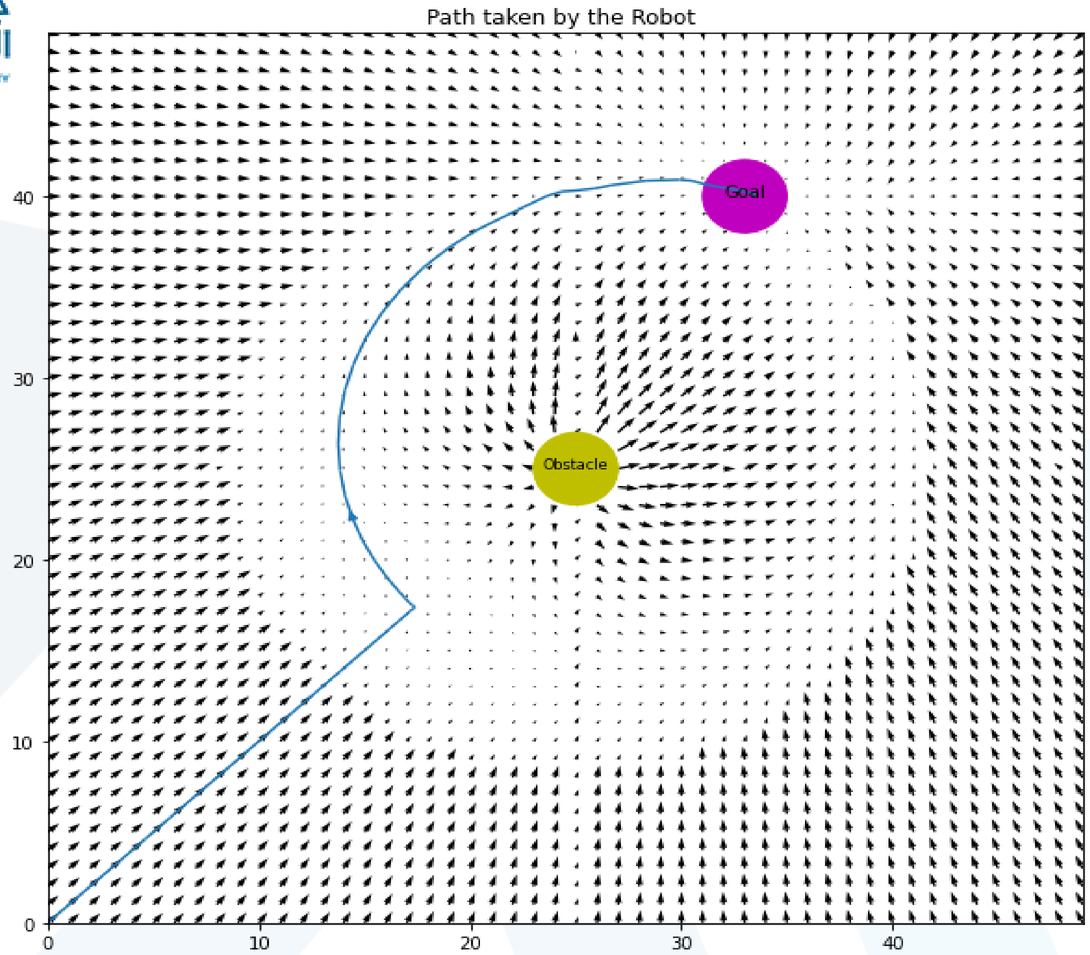
Example:



- For instance, let's assume that there is no obstacle in the environment and that the robot should seek this goal. In conventional planning, one should calculate the robot's relative position to the goal and then apply the suitable forces that will drive the robot to the goal.
- In the potential field approach, we create an attractive field going inside the goal. The potential field is defined across the entire free space, and in each time step, we calculate the potential field at the robot position and then calculate the induced force by this field. The robot then should move according to this force.

Example:





The two behaviors, seeking and avoiding, can be combined by combining the two potential fields; the robot then can follow the force induced by the new field to reach the goal while avoiding the obstacle

حقل التجاذب والتنافر

$$Vg=Kg*rg = Kg * \sqrt{(x-xg)^2 + (y-yg)^2}$$

Vg : حقل التجاذب بين الروبوت والهدف

Kg : ثابت يحدد تصميميا يعطي ثقل الهدف

rg : المسافة ما بين الروبوت والهدف

$$Vo=Ko/ro = Ko/\sqrt{(x-xo)^2 + (y-yo)^2}$$

Vo : حقل التجاذب بين الروبوت والعائق

Ko : ثابت يحدد تصميميا يعطي ثقل للعائق

ro : المسافة ما بين الروبوت والعائق

$$V=Vg+Vo$$

V: الحقل الكلي

Vo: مجموع كل حقول العوائق

إحداثيات الهدف في المثال (10,10)

$$F_{Gx} = -\frac{\partial V_G}{\partial x} = K_G \frac{(10 - x)}{r_G}$$

$$F_{Gy} = -\frac{\partial V_G}{\partial y} = K_G \frac{(10 - y)}{r_G}$$

FGx: القوة المتولدة عن جذب الهدف للروبوت على المحور x

FGy: القوة المتولدة عن جذب الهدف للروبوت على المحور y

$$V_o = \frac{K_o}{\sqrt{(x-x_o)^2+(y-y_o)^2}} = \frac{K_o}{r_o}$$

$$F_{oix} = -\frac{\partial V_{oi}}{\partial x} = -\frac{\frac{\left(-(x-x_{oi})\right)(Ko)}{r_{oi}}}{{r_{oi}}^2} = -K_o \frac{(x_{oi}-x)}{{r_{oi}}^3}$$

$$F_{oiy} = -\frac{\partial V_{oi}}{\partial y} = -K_o \frac{(y_{oi}-y)}{{r_{oi}}^3}$$

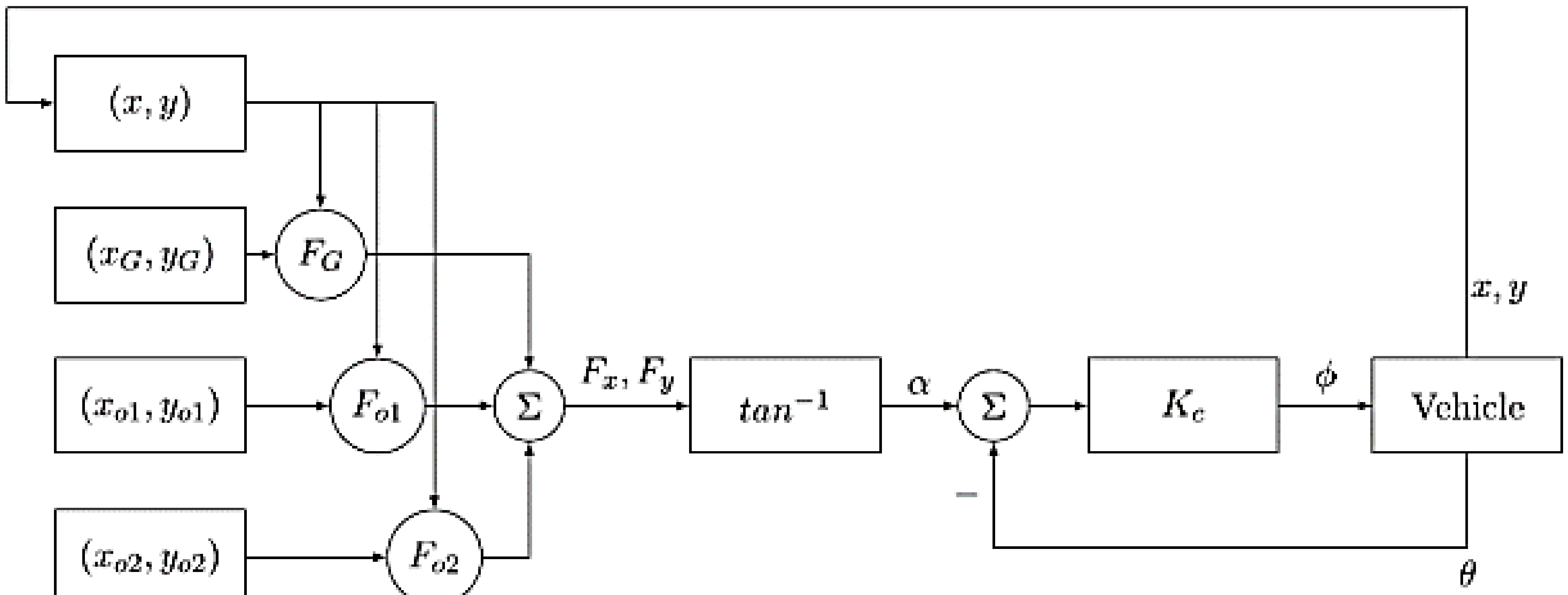
Foix: قوة التنافر المتولدة عن العائق (i) على المحور x

Foiy: قوة التنافر المتولدة عن العائق (i) على المحور y

$$\alpha = tan^{-1}(\frac{F_y}{F_x})$$

Fx : تساوي مجموع القوى على المحور x

Fy : تساوي مجموع القوى على المحور y



(x, y)
(x_G, y_G)
F_G
(x_o1, y_o1)
F_o1
(x_o2, y_o2)
F_o2
Σ
F_x, F_y
tan^-1
α
Σ
K_c
ϕ
Vehicle
x, y
−
θ

Limitations of the Artificial Potential Field Method

01 When the object is far away from the target point, the gravitational force will become particularly large, and the relatively small repulsive force may even be ignored, and obstacles may be encountered on the path of the object.

02 When there are obstacles near the target point, the repulsive force will be very large and the gravitational force will be relatively small, making it difficult for the object to reach the target point.

03 There is a local minimum problem when faced with specific obstacles.

Local minimum

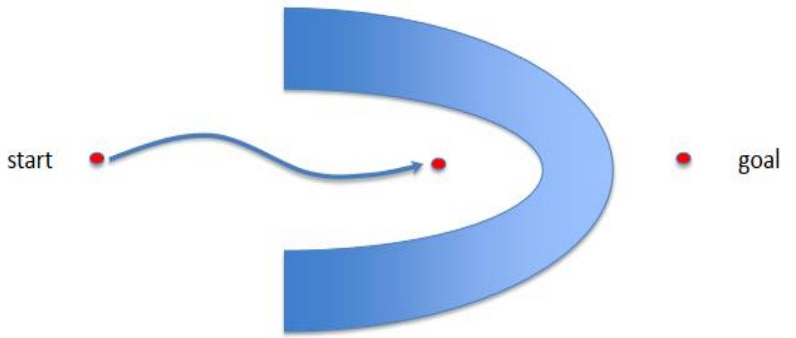
- Local minimum:

  attractive force (goal) = repulsive force (obstacles)

- Local minimum: attractive force = repulsive force

  Solution: Take a random walk – perturb out of minima

  Need to remember where you have been!



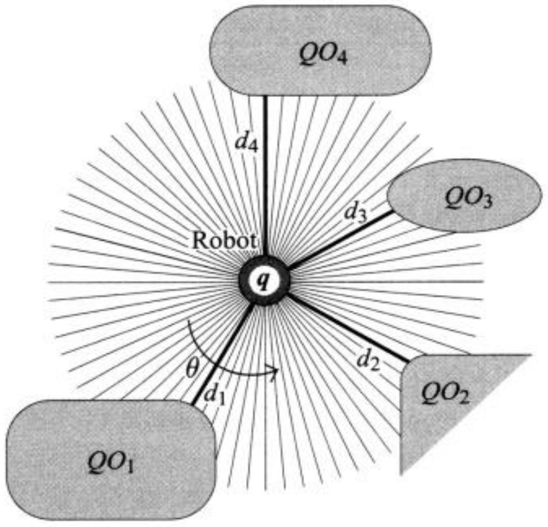
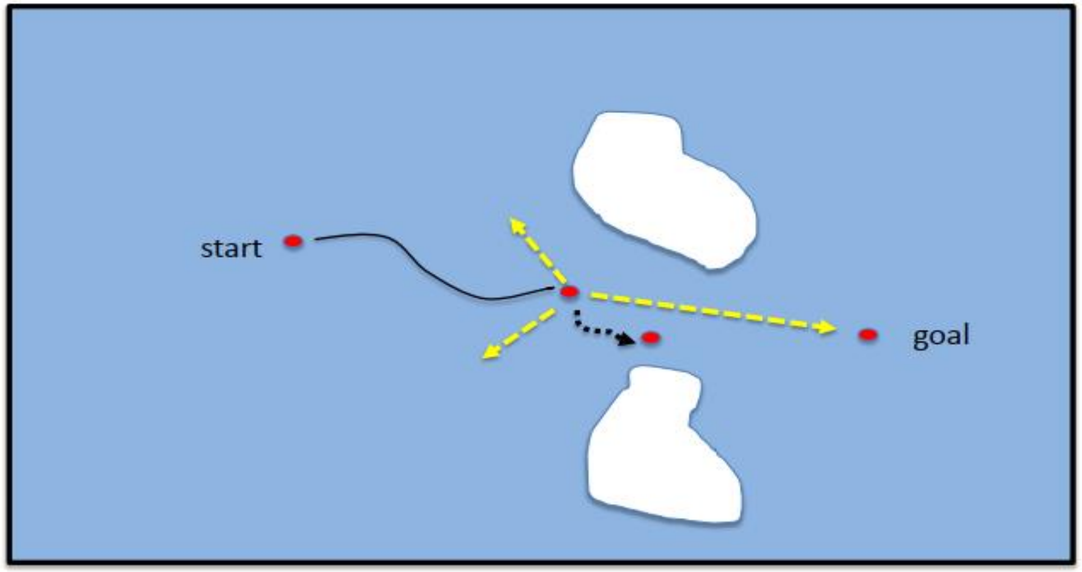


Figure 4.7 Local minima of rays determine the distance to nearby obstacles.



Potential Fields Summary

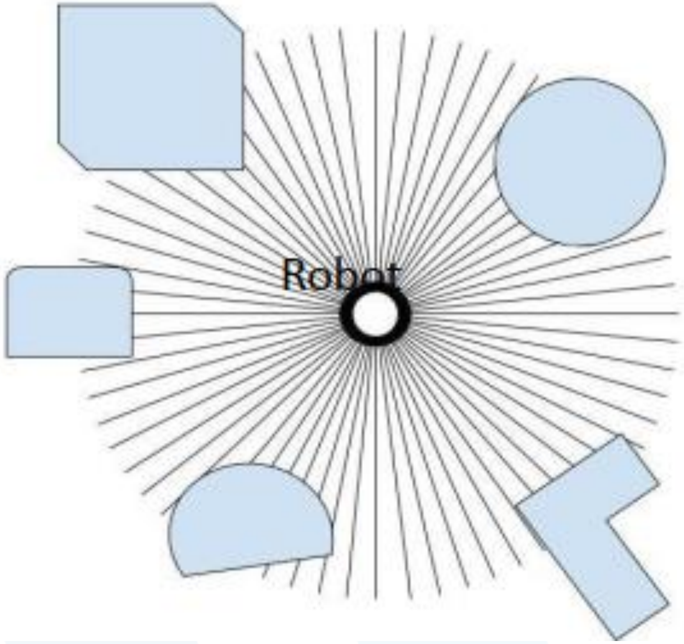
More than just a path planner: Provides simple control function to move robot: gradient descent

- Allows robot to move from wherever it finds itself
- Can get trapped in local minima
- Can be used as online, local method:

- The potential field map is computed once before robot movement. However this assumption is valid only for the static environments. In dynamic spaces, where the obstacles can change the position the potential field has to be computed as frequent as possible (in order to avoid collision).

– As robot encounters new obstacles via sensors, it can compute the Potential Function online

– Laser/sonar scans give online distance Computation to obstacles



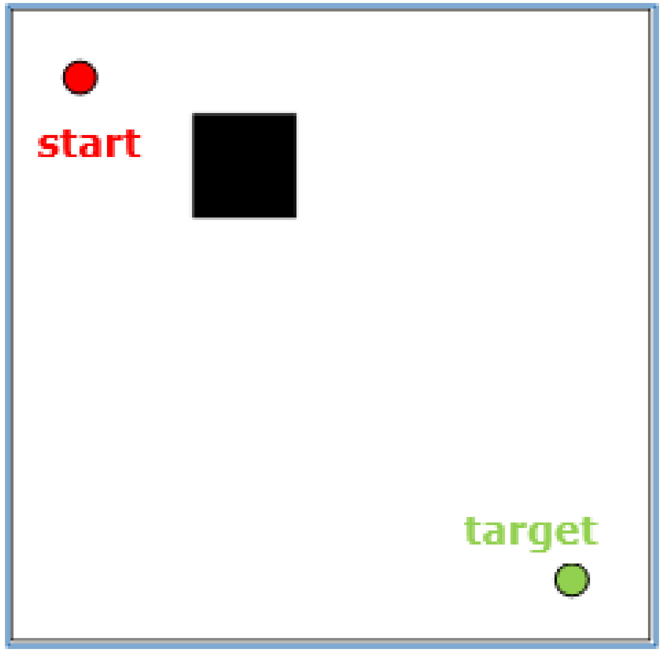
Program flow chart

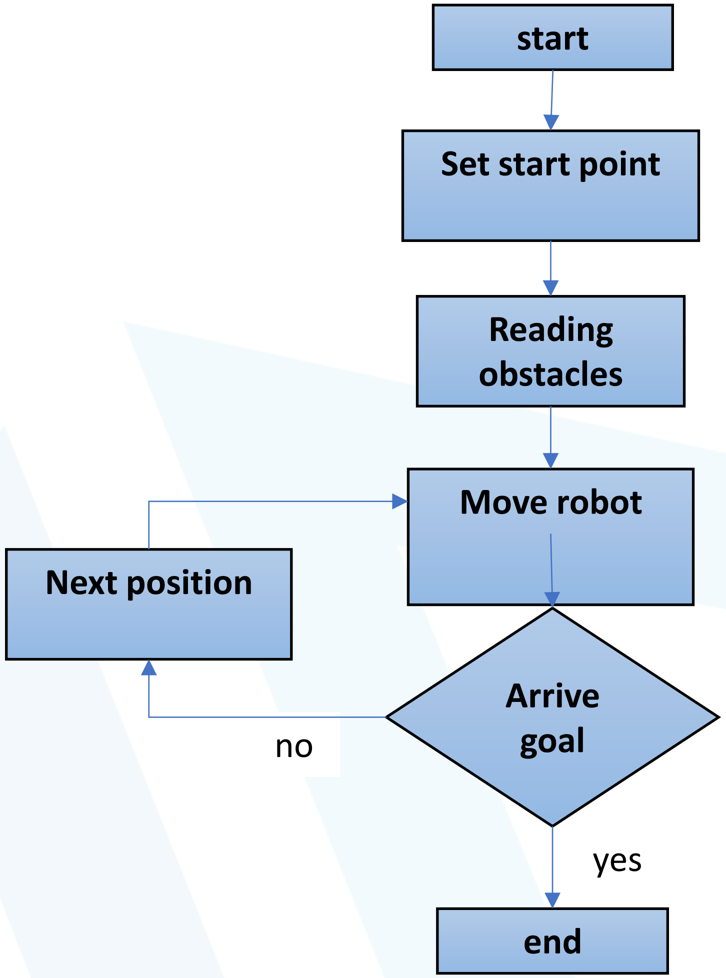
Set starting and target positions:

```
source=[50 50];
goal=[450 450];
```

Reading obstacles:

```
obstacle(:,:,1)=
int16(im2bw(imread('map1.bmp')));
```

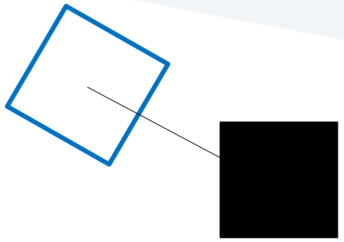




Artificial Potential Field Method

Determine next step:

(1) Calculate gravitational potential and repulsive potential

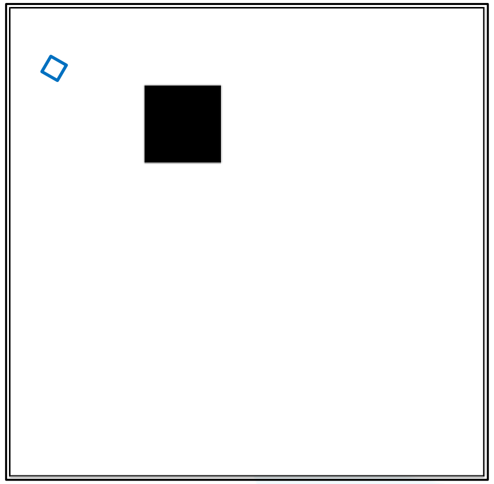


(2) Calculate the next feed angle

(3) Calculate the next step position

Draw the robot:

```
plotRobot(position,direction,map,HalfDiagonalDistance);
```



Determine whether the end point has been reached:

```
distanceGoal=( sqrt(sum((currentPosition-goal).^2)));
if distanceGoal<distanceThreshold, pathFound=true;
end
```

Gradient Descent:

q(0)=q_start

$i = 0$

while $\| \nabla U(q(i)) \| > \varepsilon$ do

$q(i+1) = q(i) - \alpha(i) \nabla U(q(i))$

$i=i+1$

THANKS

انتهت المحاضرة