Goal

Lecture 14
Planning - V - Potential Fields

Start

Course Logistics

- Project 5 was posted on 02/28 and is due on 03/20 (extended by a week).
- Forming groups for P7 and Final Project
  - We will send a google-form today for students to form groups of 4.
  - This will be due on 03/20.
  - UNITE students will have different group formations 3 and 4. Karthik will reach out to them.
- Project 6 will be posted on 03/20 and will be due on 03/27.
- Quiz 7 will be posted tomorrow at noon and will be due on Wed at noon.

Course Logistics

Updated accordingly

- Project 5 was posted on 02/28 and i
- Forming groups for P7 and Final Pro
  - We will send a google-form today
  - This will be due on 03/20.
  - UNITE students will have differen out to them.
- Project 6 will be posted on 03/20 an
- Quiz 7 will be posted tomorrow at no

Snapshot of Planned Schedule

CSCI5551-Spring-24-Calendar : Sheet1

Lec #	Date	Topic	Project Announcement	Project Due	Pre-class Quiz
1	01/17	Introduction			
2	01/22	Planning I - Path Planning			
3	01/24	Linear Algebra Refresher	P1: JS, BFS, DFS		Q1
4	01/29	Representations I - Transformations			
5	01/31	Representations II - Rotations - Quaternions	P2: Forward Kinematics	P1: Due	Q2
6	02/05	Manipulation I - Forward Kinematics			
7	02/07	Manipulation II - Inverse Kinematics	P3: Robot Dance	P2: Due	Q3
8	02/12	Manipulation III - Inverse Kinematics			
9	02/14	Manipulation - New Frontiers	P4: Inverse Kinematics	P3: Due (extended to 02/15)	Q4
10	02/19	Planning II - Bug Algorithms			
11	02/21	Planning III - Configuration Space			Q5
12	02/26	Planning IV - Sampling-based Planning			
13	02/28	Planning V - Collision Detection	P5: Planning	P4: Due	Q6
14	03/04	Spring Break			
15	03/06	Spring Break			
16	03/11	Planning VI - Potential Fields	*Forming groups for P7 and FP*		
17	03/13	Motion Control			Q7
18	03/18	Mobile Robotics I - Probability			
19	03/20	Mobile Robotics II - Sensor and Motion Models	P6: Mobile Manipulation	P5: Due	Q8
20	03/25	Mobile Robotics III - Kalman	FP Proposals Request		
21	03/27	Mobile Robotics IV - Localization	P7: Real Robot Challenge	P6: Due	Q9
22	04/01	Mobile Robotics V - Localization			
23	04/03	Mobile Robotics VI - Mapping			Q10
24	04/08	Mobile Robotics VII - SLAM		FP Proposals Due	
25	04/10	Open Ended Final Project Pitches			Q11
26	04/15	Open Ended Final Project Pitches			
27	04/17	Open Ended Final Project Pitches		P7: Due	Q12
28	04/22	Guest Lectures / Extra office hours			
29	04/24	Guest Lectures / Extra office hours			*Extra Q13*
30	04/29	Guest Lectures / Extra office hours			
31	05/01	Guest Lectures / Extra office hours		FP Posters Due	
32	05/06	Poster Day		FP Videos Due	

RRT Algorithm

RRT Algorithm

Extend graph towards a random configuration and repeat

BUILD_RRT(q_{init})
$\mathcal{T}$.init(q_{init});
for $k = 1$ **to** K **do**
    $q_{rand} \leftarrow$ RANDOM_CONFIG();
    EXTEND($\mathcal{T}, q_{rand}$);
Return $\mathcal{T}$

[Kuffner, LaValle 2000]

RRT Algorithm

Extend graph towards a random configuration and repeat

```
BUILD_RRT(q_init)
  T.init(q_init);
  for k = 1 to K do
      q_rand ← RANDOM_CONFIG();
      EXTEND(T, q_rand);
  Return T
```

Figure 3: The EXTEND operation.

[Kuffner, LaValle 2000]

RRT Algorithm

Extend graph towards a random configuration and repeat

```
BUILD_RRT(q_init)
  T.init(q_init);
  for k = 1 to K do
      q_rand ← RANDOM_CONFIG();
      EXTEND(T, q_rand);
  Return T
```

```
EXTEND(T, q)
  q_near ← NEAREST_NEIGHBOR(q, T);
  if NEW_CONFIG(q, q_near, q_new) then
      T.add_vertex(q_new);
      T.add_edge(q_near, q_new);
      if q_new = q then
          Return Reached;
      else
          Return Advanced;
  Return Trapped;
```

Extend graph towards a random configuration

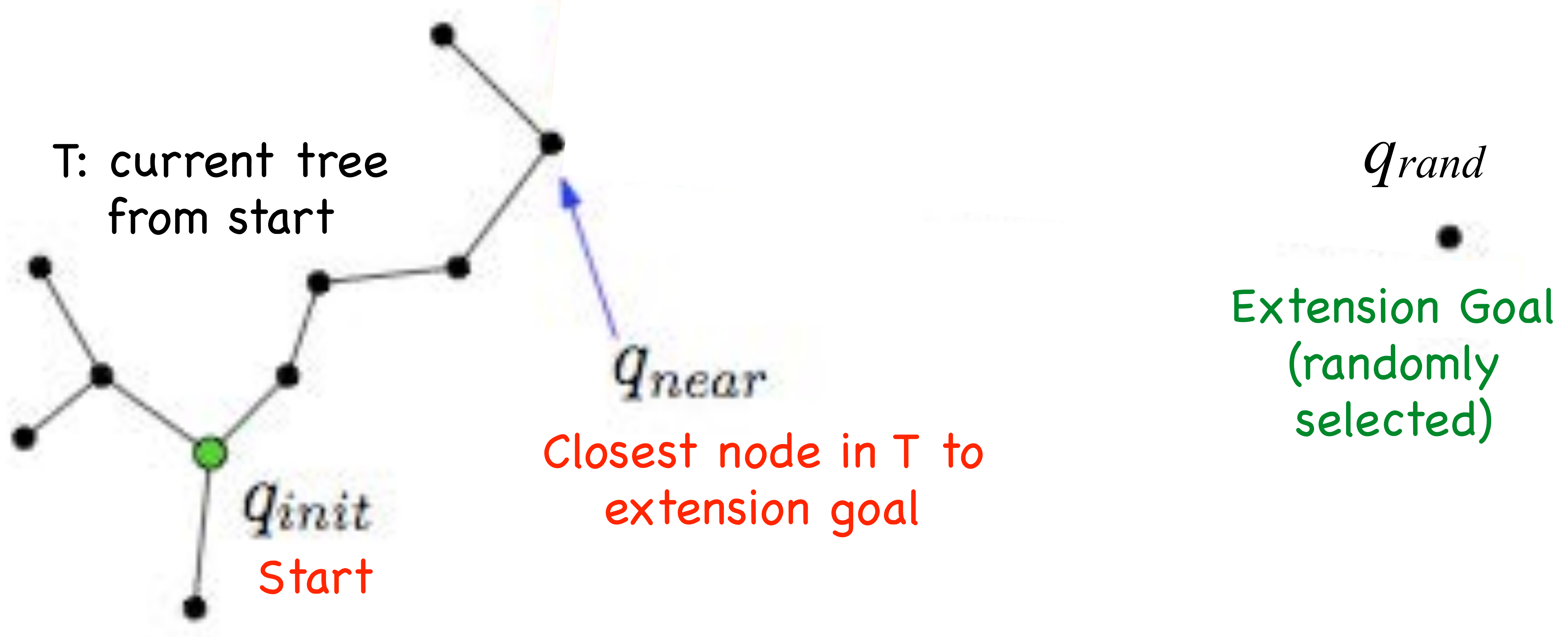


Figure 3: The EXTEND operation.

Slide borrowed from Michigan Robotics autorob.org

[Kuffner, LaValle 2000]

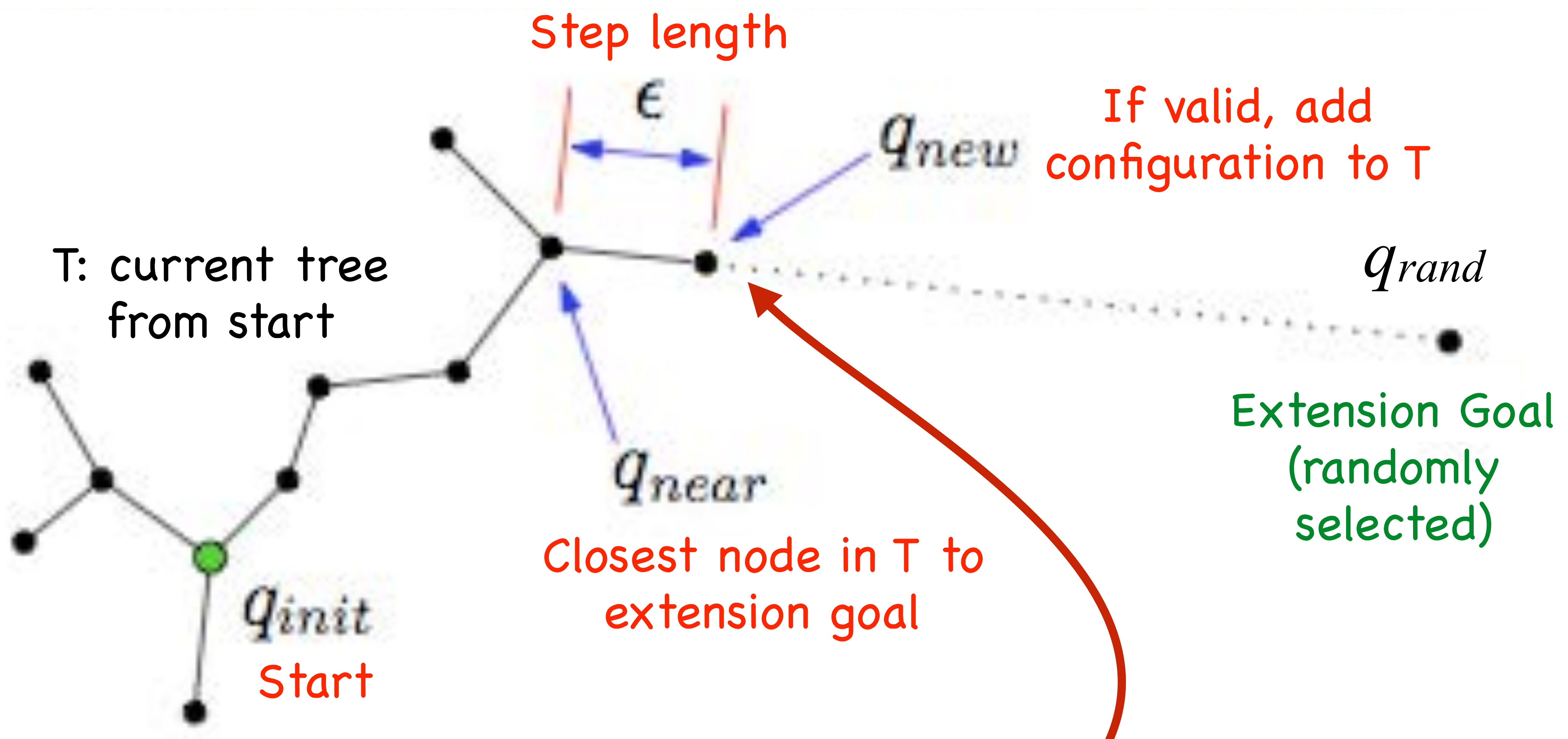
RRT Algorithm

Extend graph towards a random configuration and repeat

```
BUILD_RRT(q_init)
  T.init(q_init);
  for k = 1 to K do
      q_rand ← RANDOM_CONFIG();
      EXTEND(T, q_rand);
  Return T
```

```
EXTEND(T, q)
  q_near ← NEAREST_NEIGHBOR(q, T);
  if NEW_CONFIG(q, q_near, q_new) then
      T.add_vertex(q_new);
      T.add_edge(q_near, q_new);
      if q_new = q then
          Return Reached;
      else
          Return Advanced;
  Return Trapped;
```

Extend graph towards a random configuration



Generate and test new configuration along vector in C-space from q_{near} to q_{rand}

RRT* Algorithm

RRT*

Algorithm 6: RRT*

```
V ← {x_init}; E ← ∅;
for i = 1, ..., n do
    x_rand ← SampleFree_i;
    x_nearest ← Nearest(G = (V, E), x_rand);
    x_new ← Steer(x_nearest, x_rand) ;
    if ObtacleFree(x_nearest, x_new) then
        X_near ← Near(G = (V, E), x_new, min{γ_RRT*(log(card(V))/card(V))^(1/d), η}) ;
        V ← V ∪ {x_new};
        x_min ← x_nearest; c_min ← Cost(x_nearest) + c(Line(x_nearest, x_new));
        foreach x_near ∈ X_near do                    // Connect along a minimum-cost path
            if CollisionFree(x_near, x_new) ∧ Cost(x_near) + c(Line(x_near, x_new)) < c_min then
                x_min ← x_near; c_min ← Cost(x_near) + c(Line(x_near, x_new))
        E ← E ∪ {(x_min, x_new)};
        foreach x_near ∈ X_near do                    // Rewire the tree
            if CollisionFree(x_new, x_near) ∧ Cost(x_new) + c(Line(x_new, x_near)) < Cost(x_near)
            then x_parent ← Parent(x_near);
            E ← (E \ {(x_parent, x_near)}) ∪ {(x_new, x_near)}
return G = (V, E);
```

Karaman & Frazzo

RRT*

Algorithm 6: RRT*

$V \leftarrow \{x_{\text{init}}\};\ E \leftarrow \emptyset;$

for $i = 1, \ldots, n$ **do**

$\quad x_{\text{rand}} \leftarrow \texttt{SampleFree}_i;$

$\quad x_{\text{nearest}} \leftarrow \texttt{Nearest}(G = (V, E), x_{\text{rand}});$

$\quad x_{\text{new}} \leftarrow \texttt{Steer}(x_{\text{nearest}}, x_{\text{rand}})\ ;$

$\quad$ **if** $\texttt{ObtacleFree}(x_{\text{nearest}}, x_{\text{new}})$ **then**

$\quad\quad X_{\text{near}} \leftarrow \texttt{Near}(G = (V, E), x_{\text{new}}, \min\{\gamma_{\text{RRT}^*}(\log(\text{card}\,(V))/\,\text{card}\,(V))^{1/d}, \eta\})\ ;$

$\quad\quad V \leftarrow V \cup \{x_{\text{new}}\};$

$\quad\quad x_{\min} \leftarrow x_{\text{nearest}};\ c_{\min} \leftarrow \texttt{Cost}(x_{\text{nearest}}) + c(\texttt{Line}(x_{\text{nearest}}, x_{\text{new}}));$

$\quad\quad$ **foreach** $x_{\text{near}} \in X_{\text{near}}$ **do** // Connect along a minimum-cost path

$\quad\quad\quad$ **if** $\texttt{CollisionFree}(x_{\text{near}}, x_{\text{new}}) \wedge \texttt{Cost}(x_{\text{near}}) + c(\texttt{Line}(x_{\text{near}}, x_{\text{new}})) < c_{\min}$ **then**

$\quad\quad\quad\quad x_{\min} \leftarrow x_{\text{near}};\ c_{\min} \leftarrow \texttt{Cost}(x_{\text{near}}) + c(\texttt{Line}(x_{\text{near}}, x_{\text{new}}))$

$\quad\quad E \leftarrow E \cup \{(x_{\min}, x_{\text{new}})\};$

$\quad\quad$ **foreach** $x_{\text{near}} \in X_{\text{near}}$ **do** // Rewire the tree

$\quad\quad\quad$ **if** $\texttt{CollisionFree}(x_{\text{new}}, x_{\text{near}}) \wedge \texttt{Cost}(x_{\text{new}}) + c(\texttt{Line}(x_{\text{new}}, x_{\text{near}})) < \texttt{Cost}(x_{\text{near}})$

$\quad\quad\quad$ **then** $x_{\text{parent}} \leftarrow \texttt{Parent}(x_{\text{near}});$

$\quad\quad\quad\quad E \leftarrow (E \setminus \{(x_{\text{parent}}, x_{\text{near}})\}) \cup \{(x_{\text{new}}, x_{\text{near}})\}$

return $G = (V, E);$

FIND x_{new}

FIND neighbors to x_{new} **in** G

ADD x_{new} **to** G

FIND edge to x_{new} **from neighbors with least cost**

ADD that to G

REWIRE the edges in the neighborhood if any least cost path exists from the root to the neighbors via x_{new}

Karaman & Frazzo

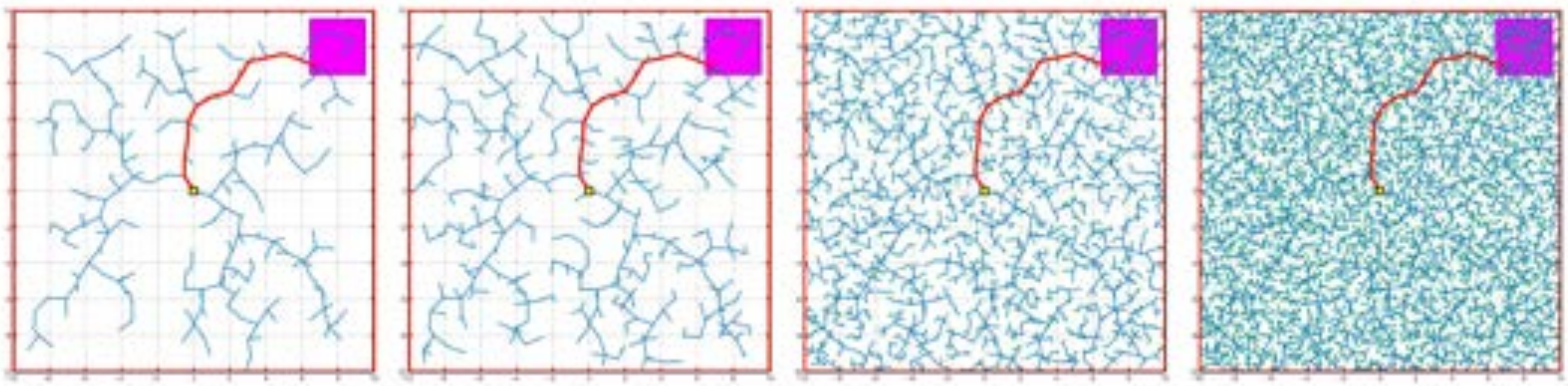
RRT*

- Asymptotically optimal
- Main idea:
  - Swap new point in as parent for nearby vertices who can be reached along shorter path through new point than through their original (current) parent

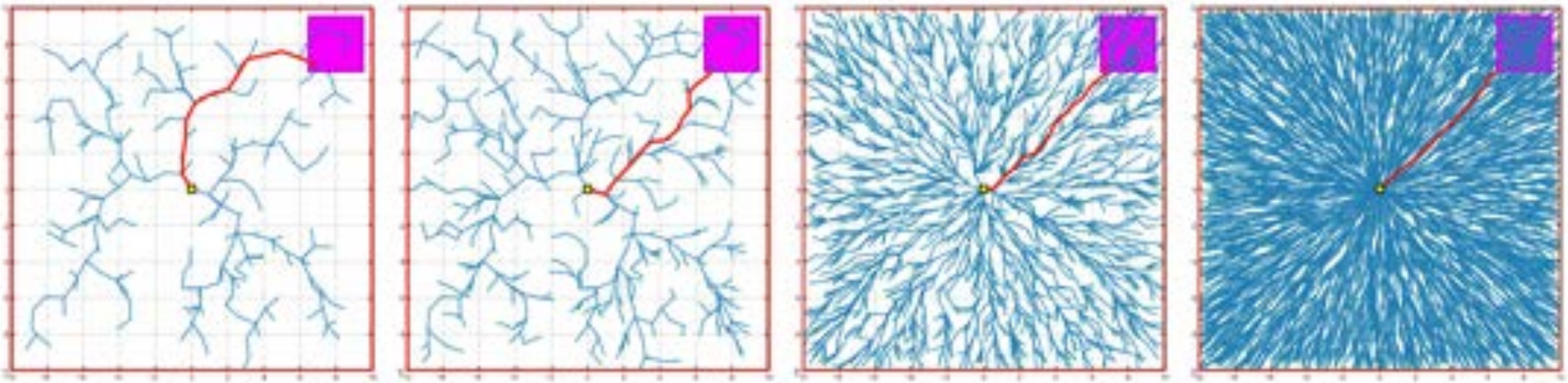
Demonstration - https://demonstrations.wolfram.com/RapidlyExploringRandomTreeRRTAndRRT/

RRT*

RRT



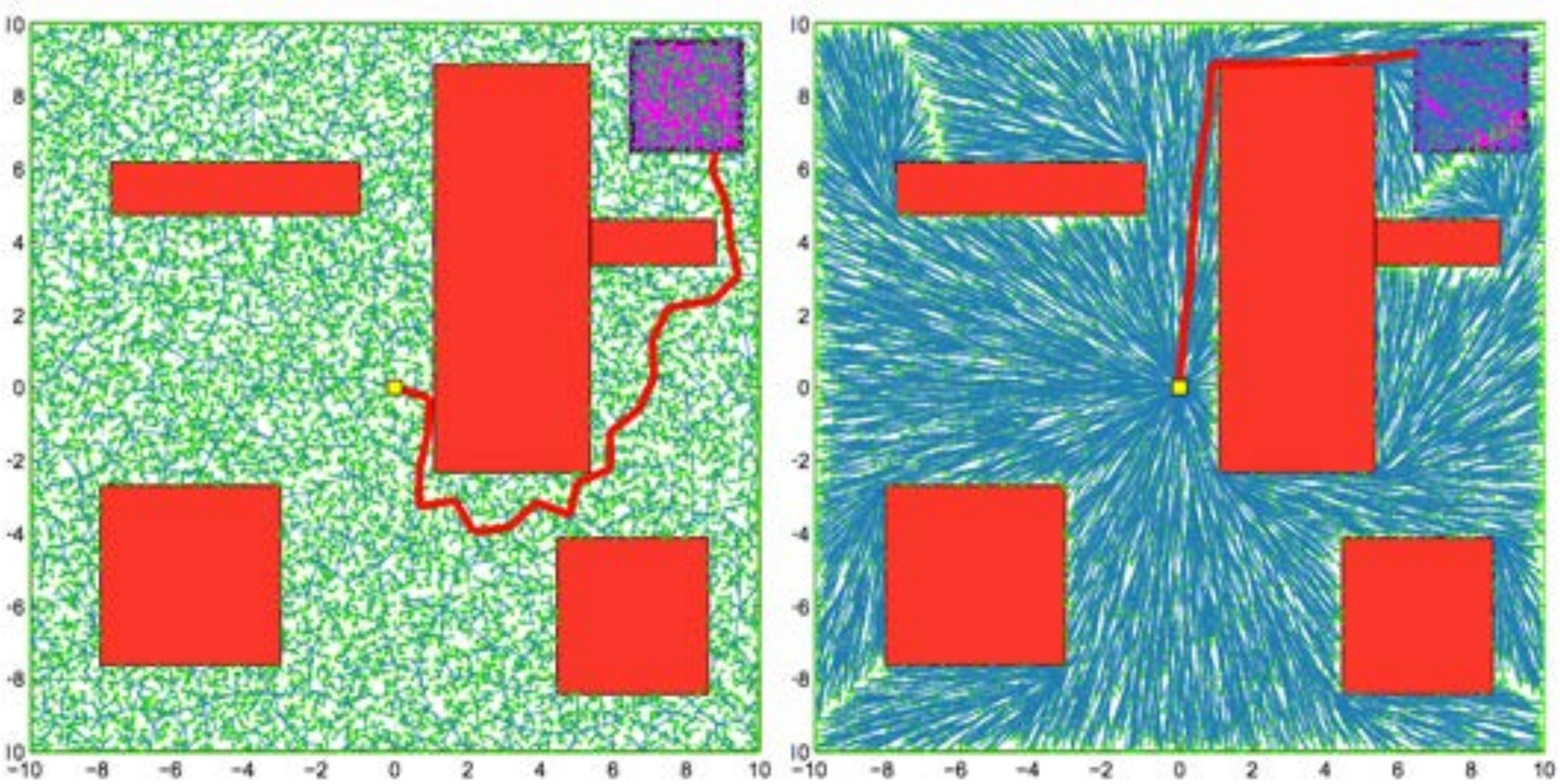
RRT*



Source: Karaman and Frazzoli

RRT*

RRT | RRT*



Source: Karaman and Frazzoli

Smoothing

Randomized motion planners tend to find not so great paths for execution: very jagged, often much longer than necessary.

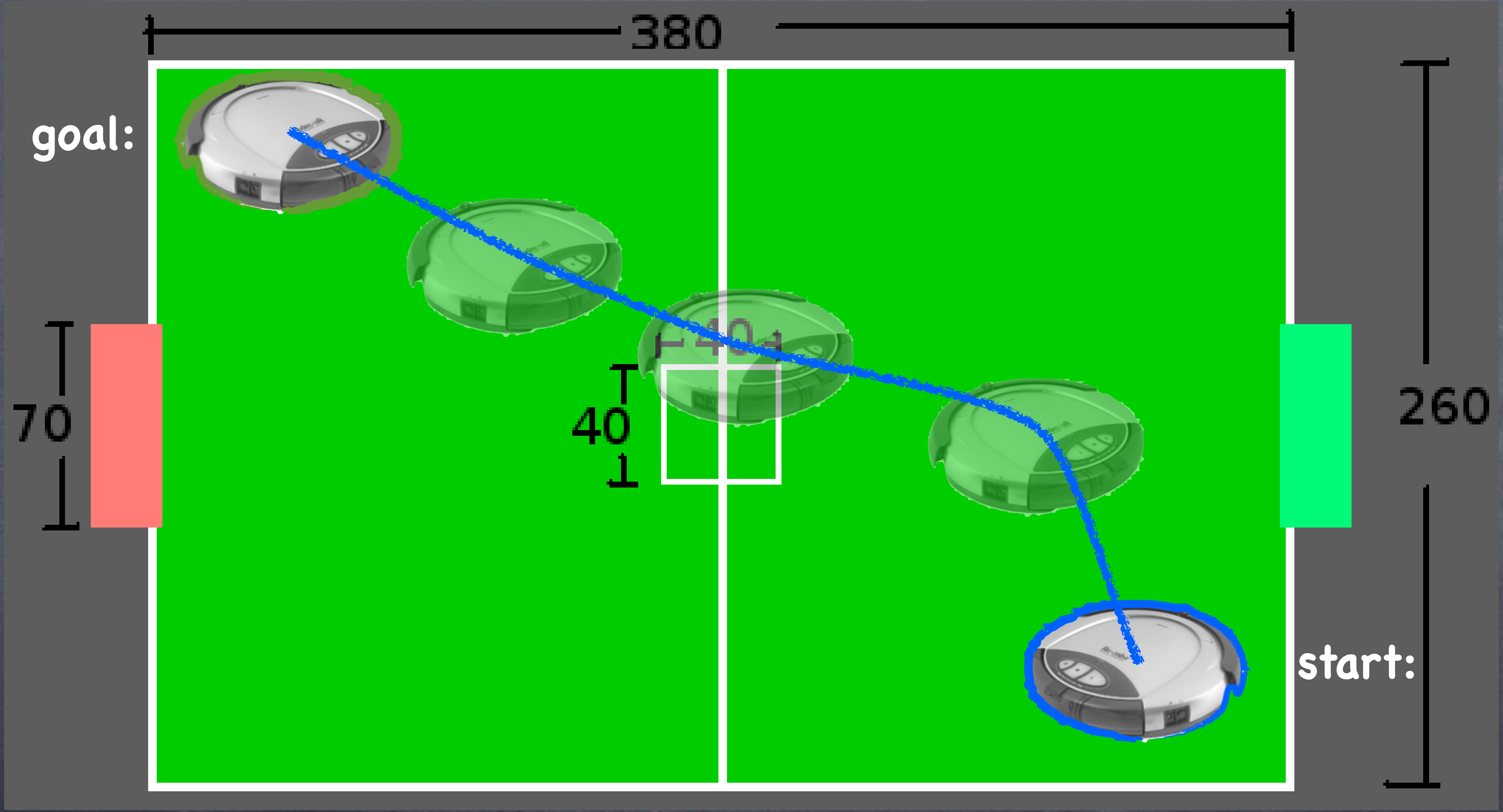
→ In practice: do smoothing before using the path

- Shortcutting:
  - along the found path, pick two vertices x_{t1}, x_{t2} and try to connect them directly (skipping over all intermediate vertices)
- Nonlinear optimization for optimal control
  - Allows to specify an objective function that includes smoothness in state, control, small control inputs, etc.

Approaches to motion planning

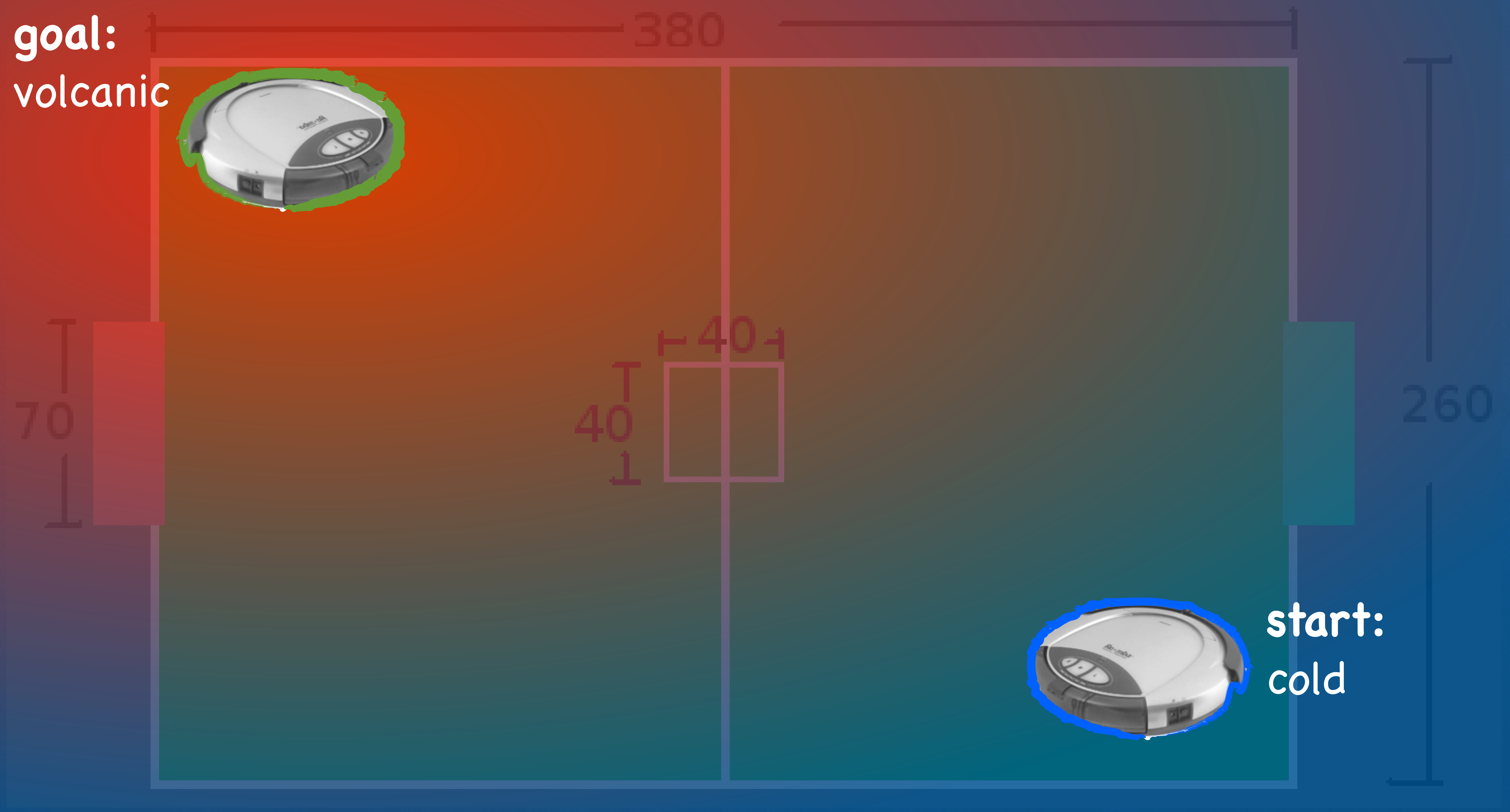
- Bug algorithms: Bug[0-2], Tangent Bug
- Graph Search (fixed graph)
    - Depth-first, Breadth-first, Dijkstra, A-star, Greedy best-first
- Sampling-based Search (build graph):
    - Probabilistic Road Maps, Rapidly-exploring Random Trees
- **Optimization and local search:**
    - **Gradient descent, Potential fields, Simulated annealing, Wavefront**

Navigation (again)



Potential field

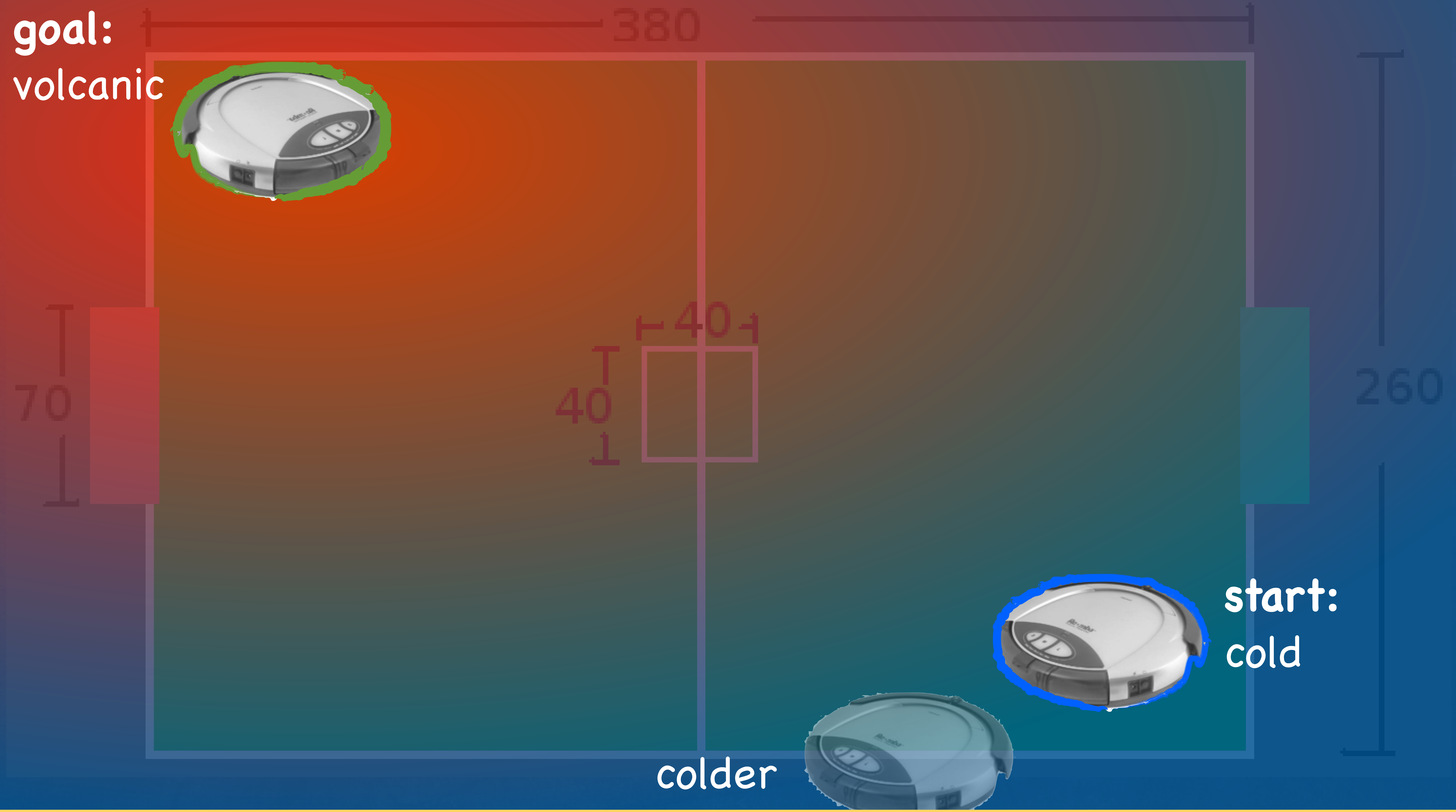
(like a game of "warmer-colder")



Slide borrowed from Michigan Robotics autorob.org

Potential field

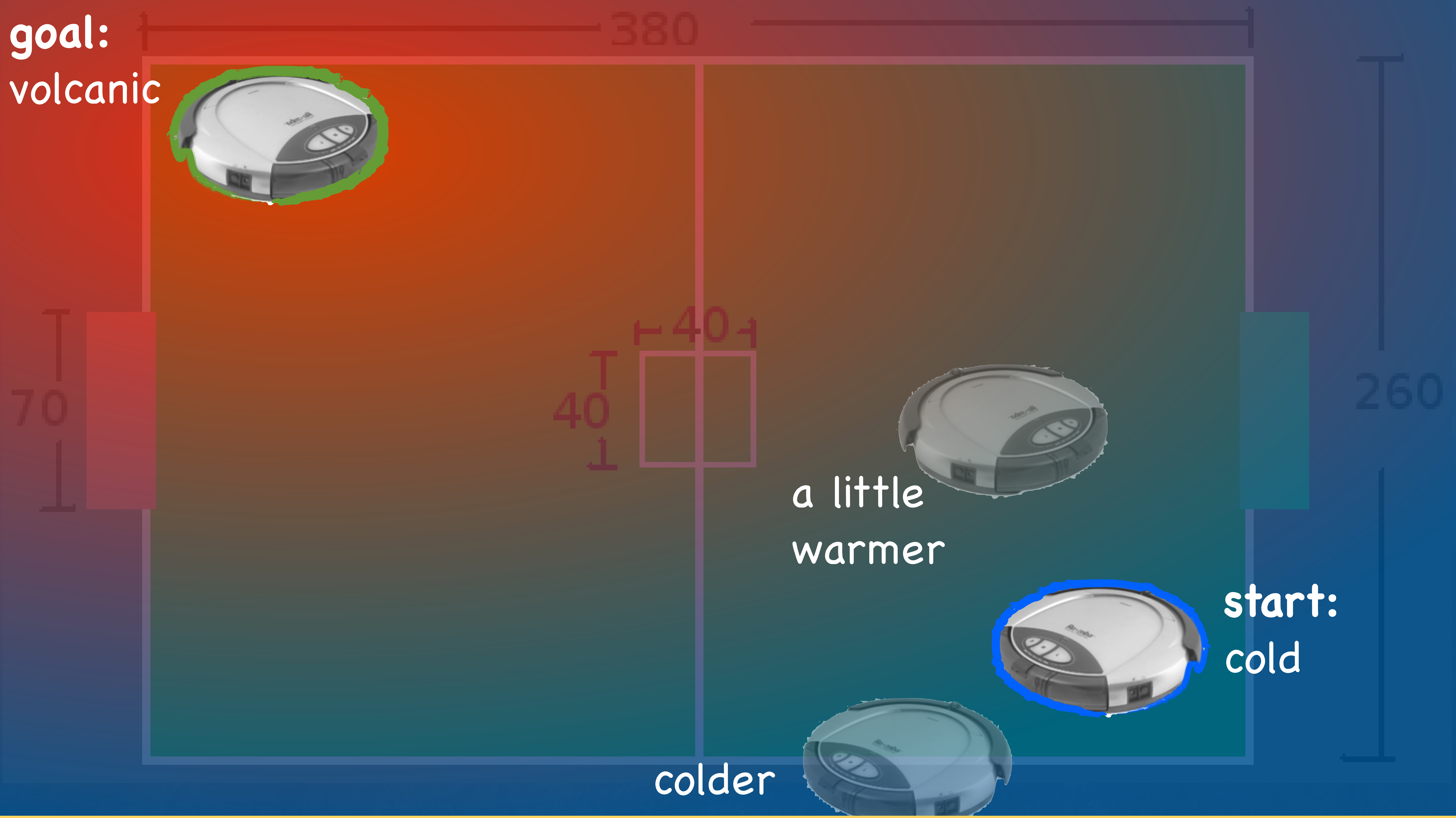
(like a game of "warmer-colder")



Slide borrowed from Michigan Robotics autorob.org

Potential field

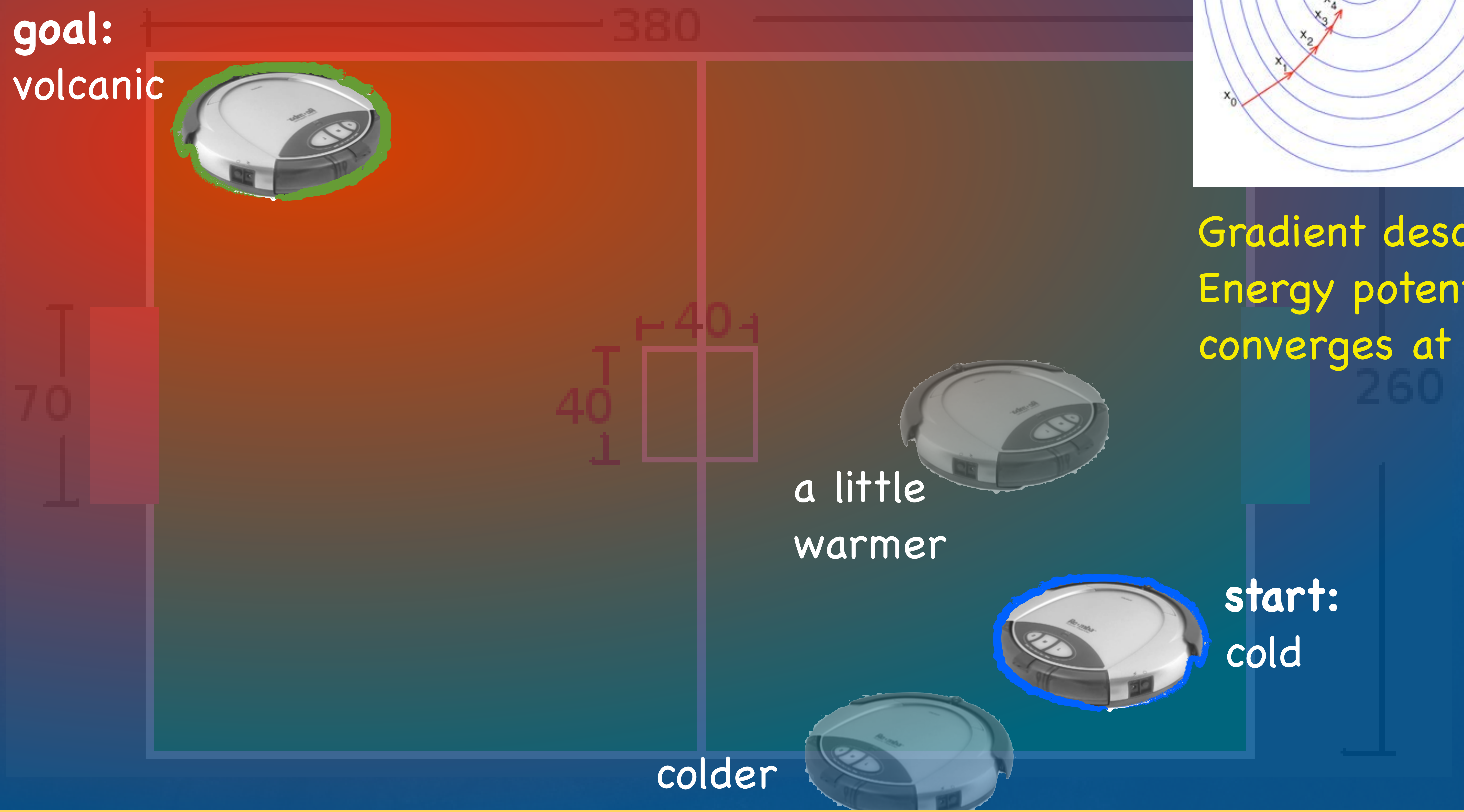
(like a game of "warmer-colder")



Slide borrowed from Michigan Robotics autorob.org

Potential field

(like a game of "warmer-colder")

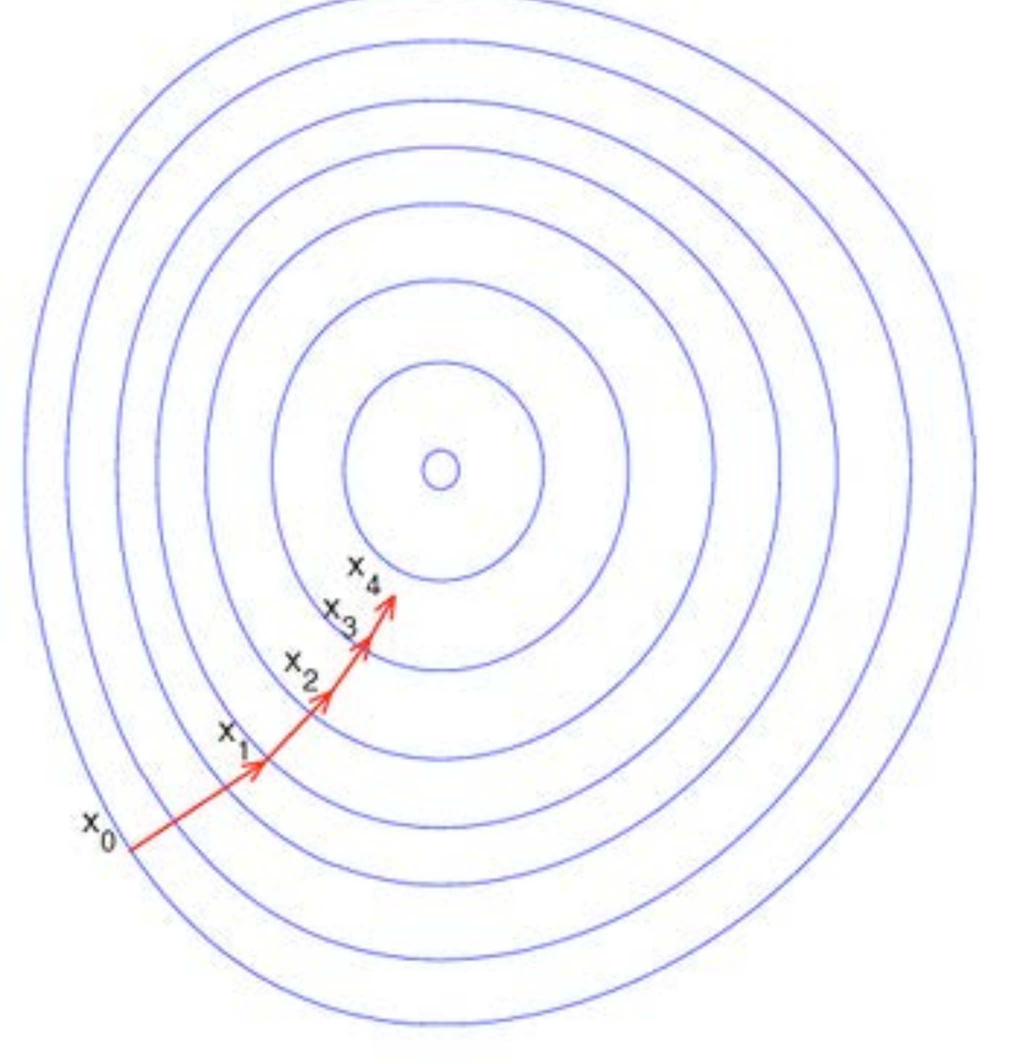


goal:
volcanic

a little
warmer

start:
cold

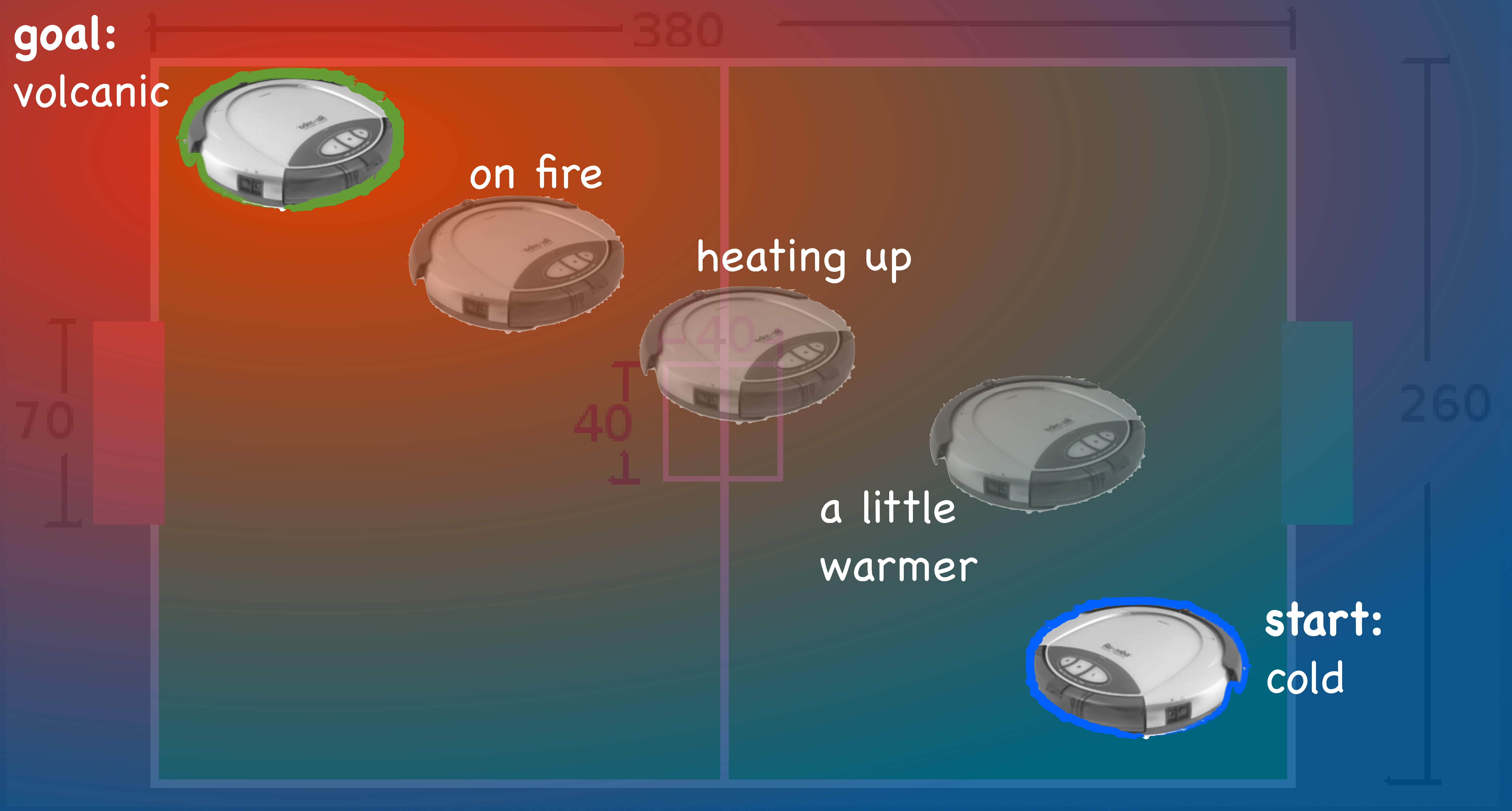
colder



Gradient descent:
Energy potential
converges at goal

Potential field

(like a game of "warmer-colder")



How do we define a potential field?

Potential Field

- A potential field is a differentiable function $U(q)$ that maps configurations to scalar "energy" value
- At any q, gradient $\nabla U(q)$ is the vector that maximally increases U
- At goal q_{goal}, energy is minimized such that $\nabla U(q_{goal}) = 0$
- Navigation by descending field $-\nabla U(q)$ *to goal*

Gradient Descent Algorithm:

$q_{path}[0] \leftarrow q_{start}$

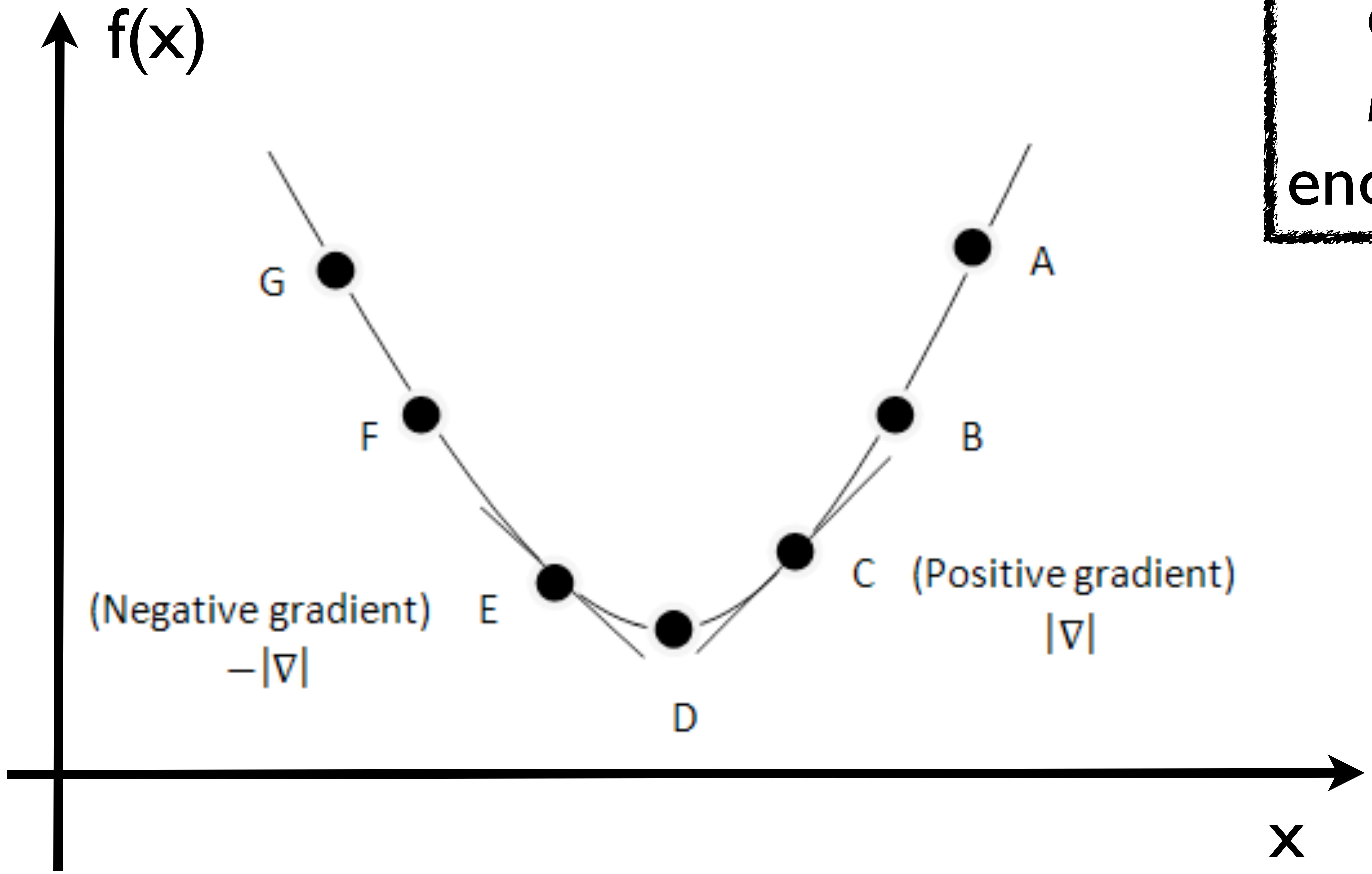
$i \leftarrow 0$

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

  $q_{path}[i+1] \leftarrow q_{path}[i] - \alpha \nabla U(q_{path}[i])$

  $i \leftarrow i+1$

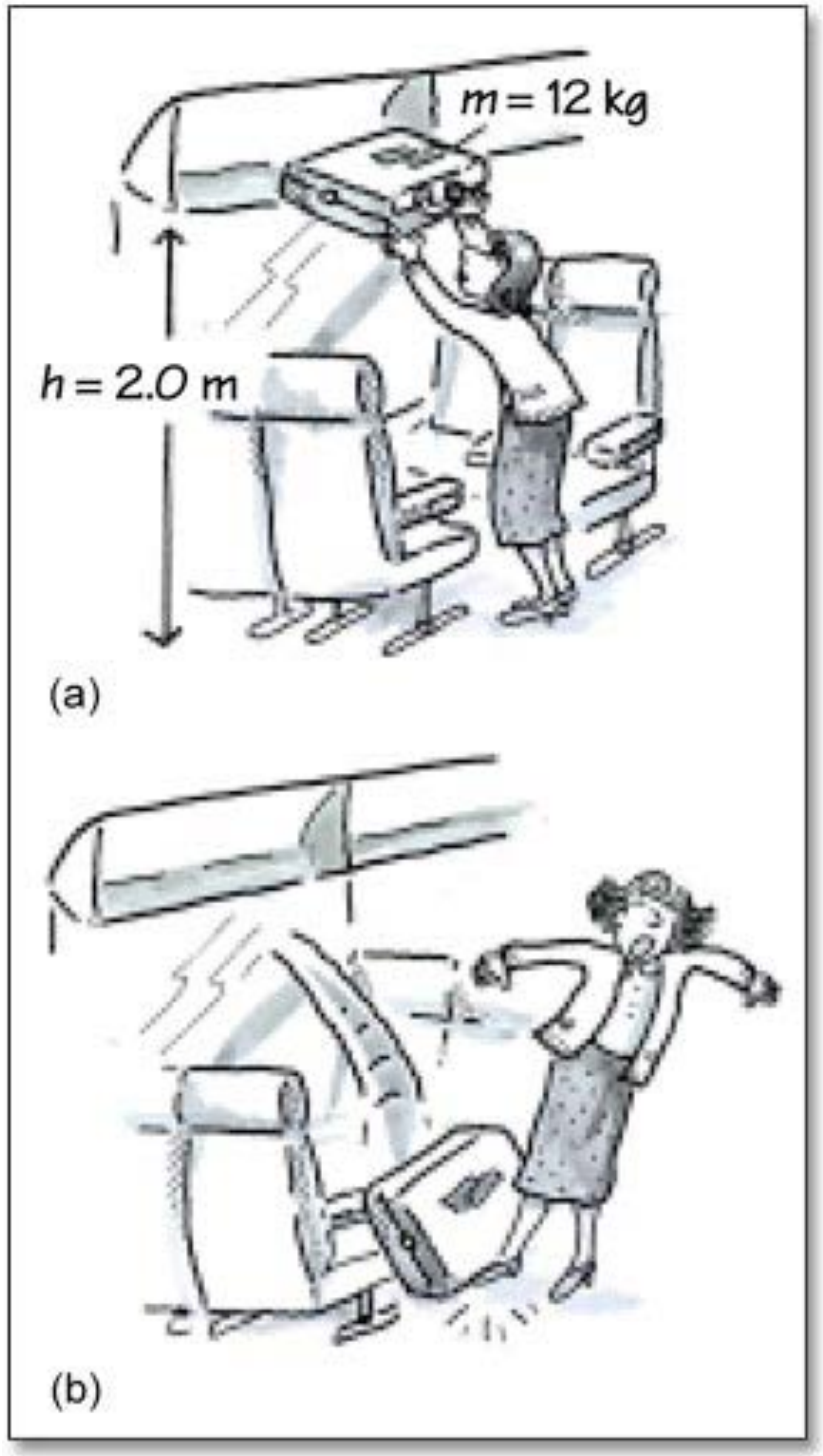
end



Derivative assumed to be direction of steepest ascent away from goal

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \gamma_n \nabla F(\mathbf{x}_n)$$

Potential Energy

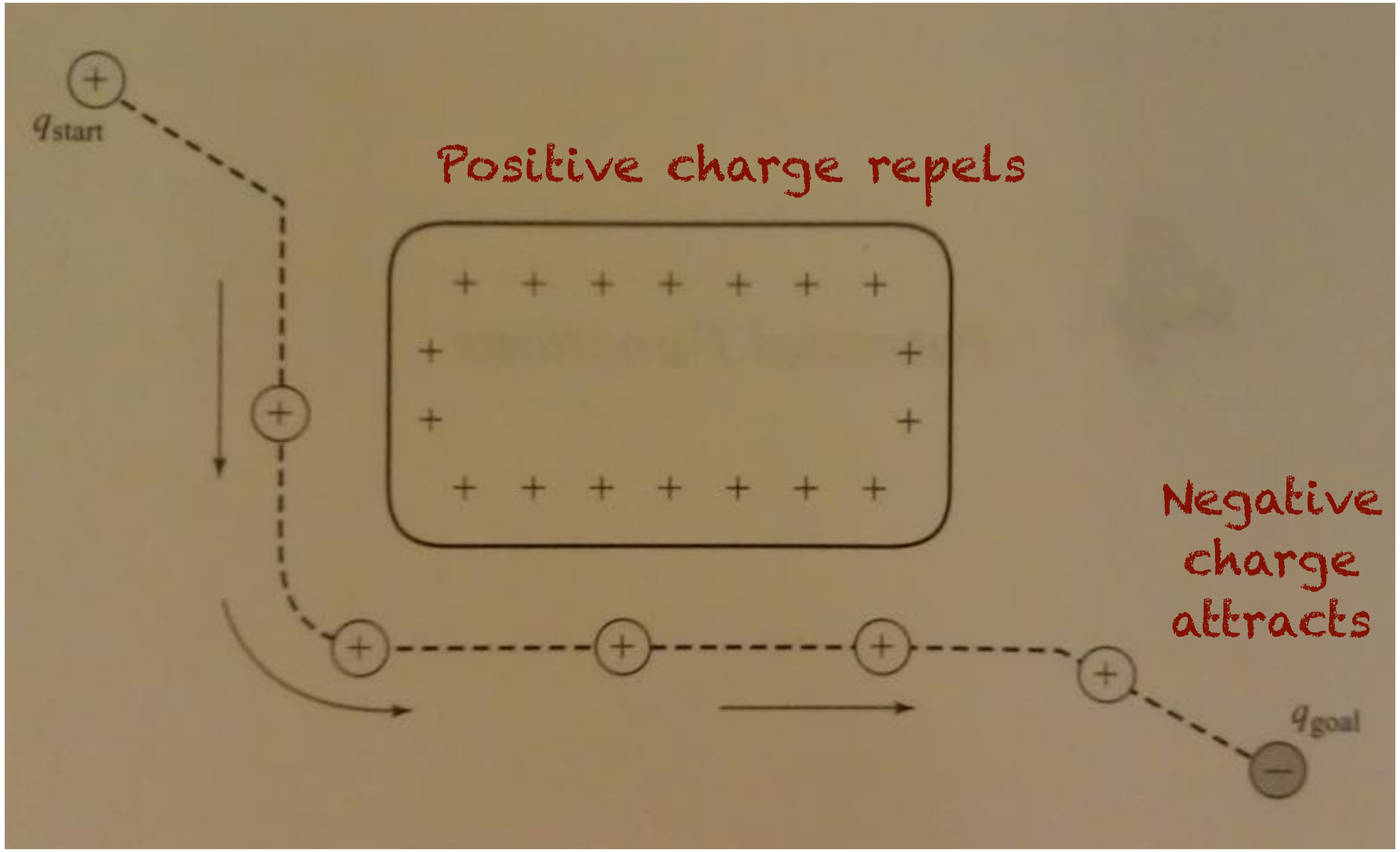


- Energy stored in a physical system
- Kinetic motion caused by system moving to lower energy state
- For objects acting only w.r.t. gravity
  - potential_energy = mass*height*gravity

Slide borrowed from Michigan Robotics autorob.org

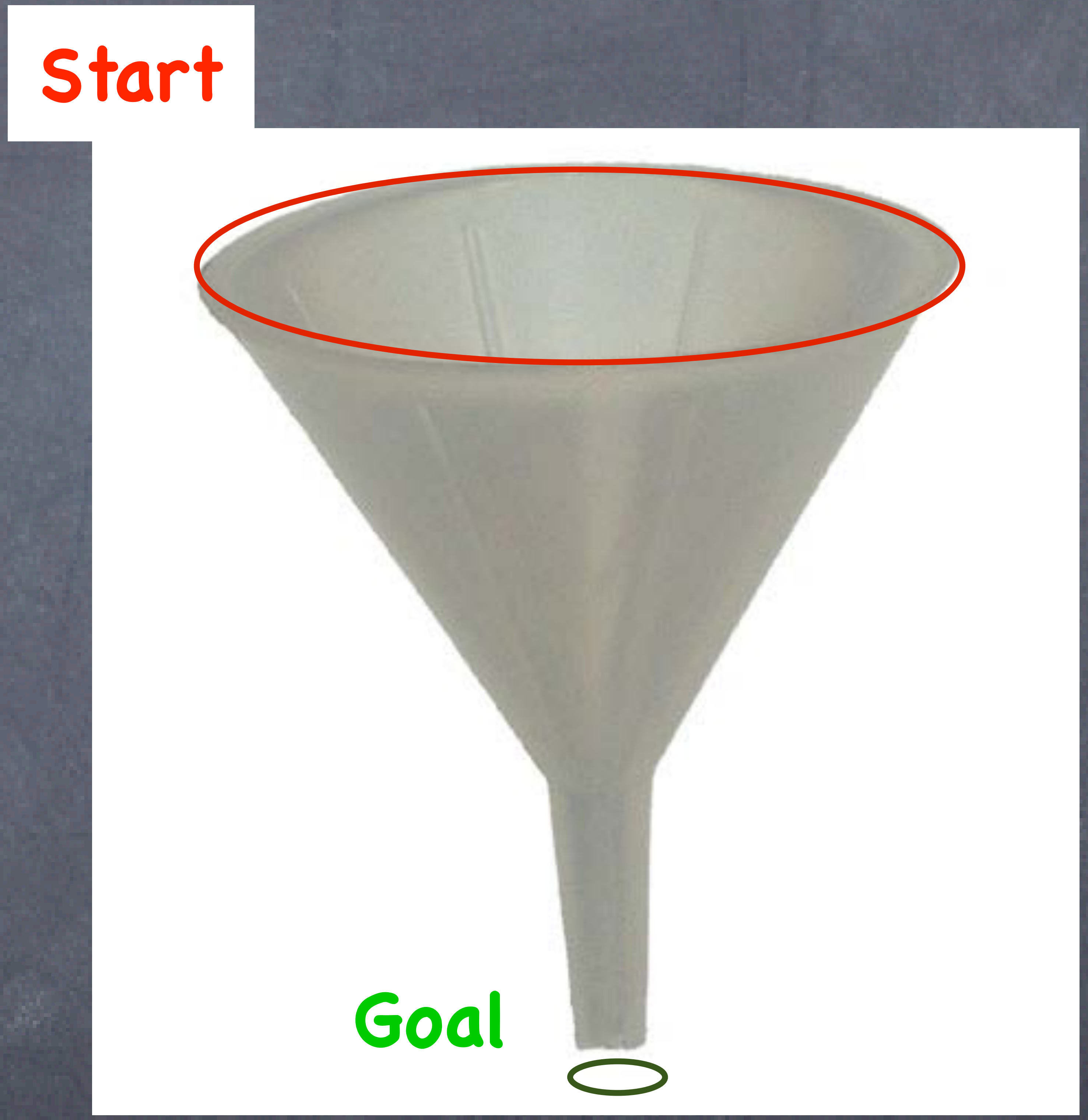
Charged Particle Example

Positively charged particle follows potential energy to goal



Convergent Potentials

let's call these "attractor landscapes"

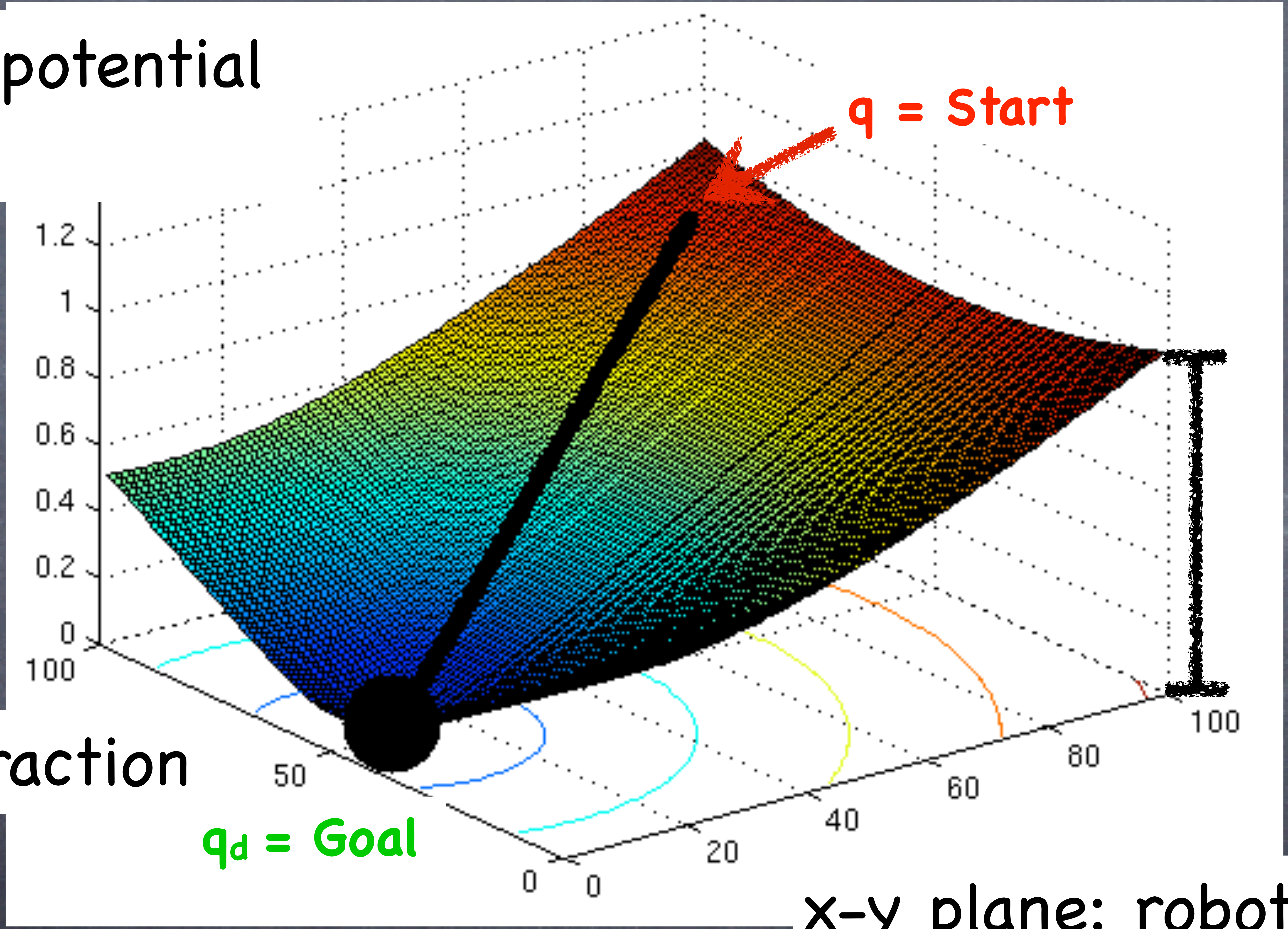


basin of attraction

Slide borrowed from Michigan Robotics autorob.org

2D potential navigation

z: height indicates potential at location

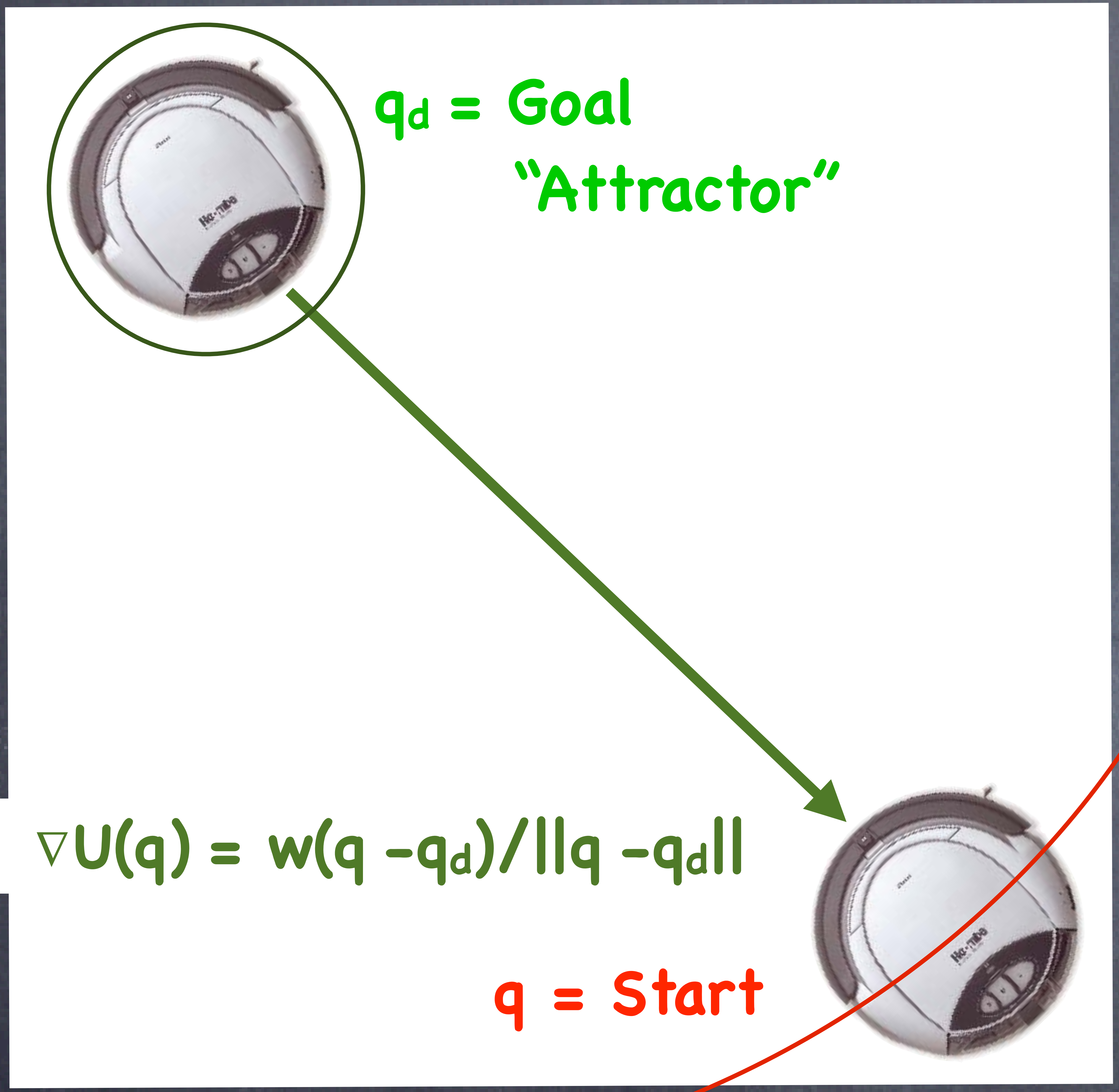


basin of attraction

x-y plane: robot position

“Cone” Attractor

top view

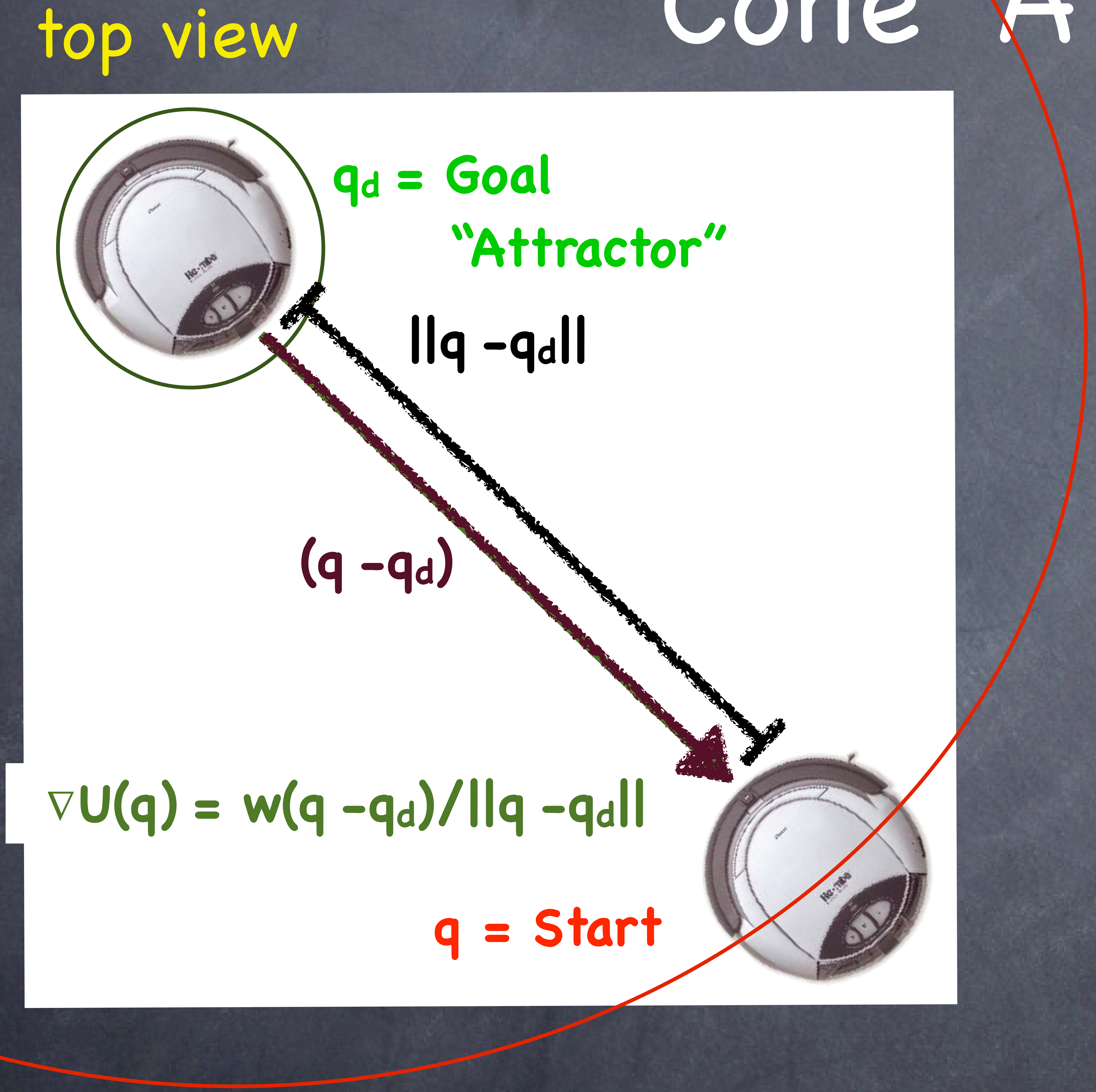


$\nabla U(q) = w(q - q_d)/||q - q_d||$

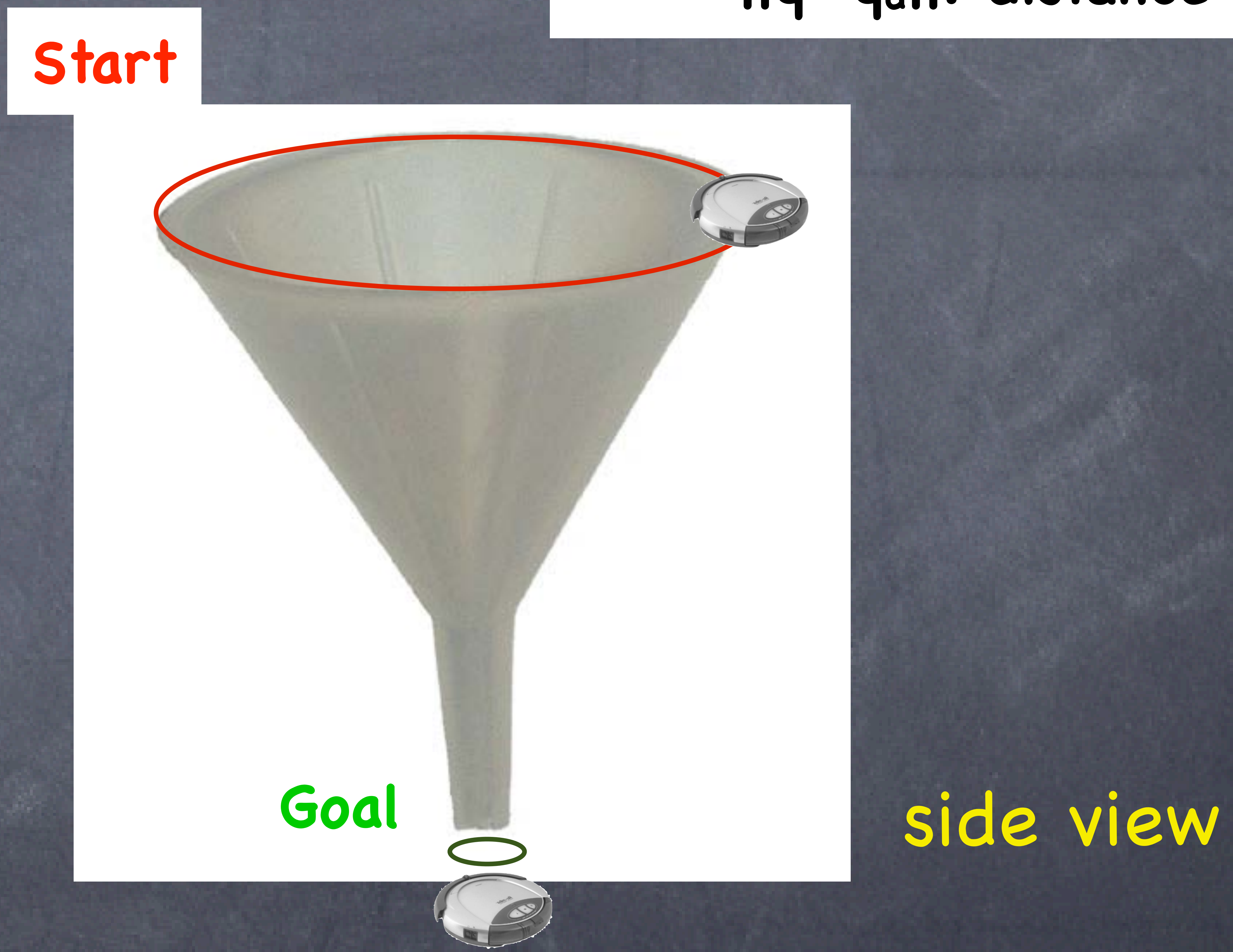
w: weight
$(q - q_d)$: direction
$||q - q_d||$: distance

side view

"Cone" Attractor

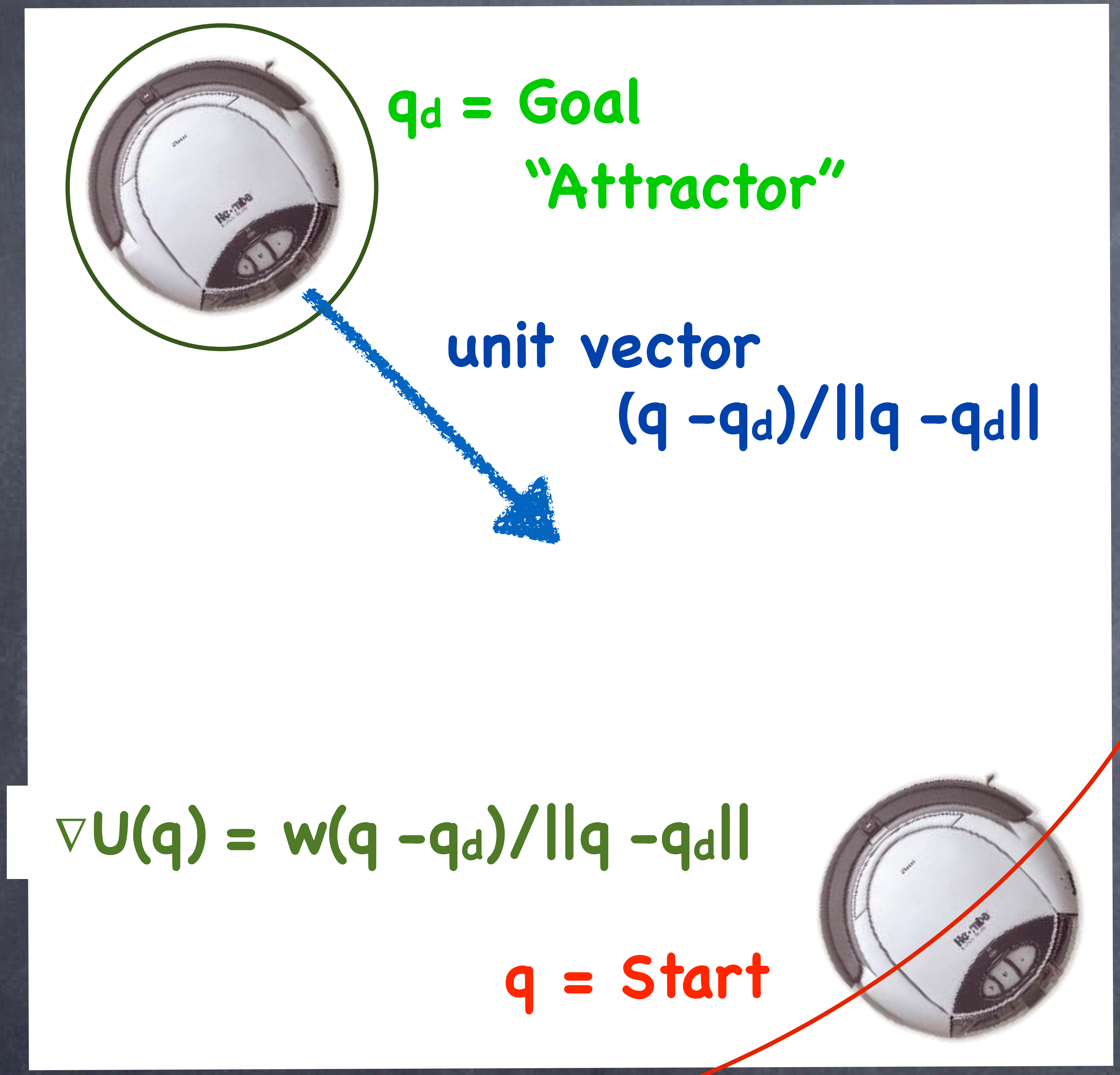


w: weight
$(q - q_d)$: direction
$||q - q_d||$: distance



"Cone" Attractor

top view

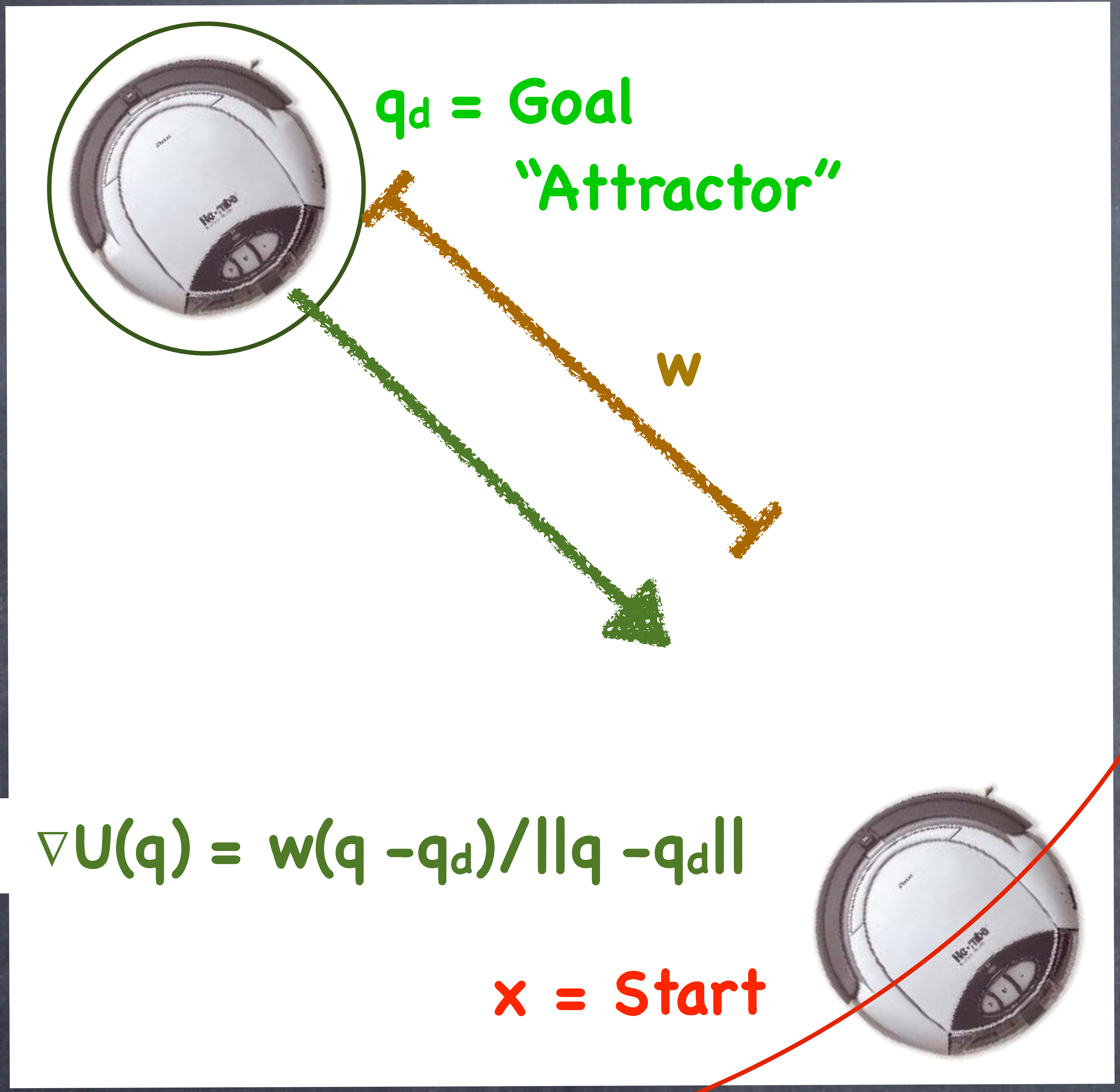


w: weight
$(q_d - q)$: direction
$\|q_d - q\|$: distance

side view

"Cone" Attractor

top view



$$\nabla U(q) = w(q - q_d)/\|q - q_d\|$$

w: weight (< 1)
$(q - q_d)$: direction
$\|q - q_d\|$: distance

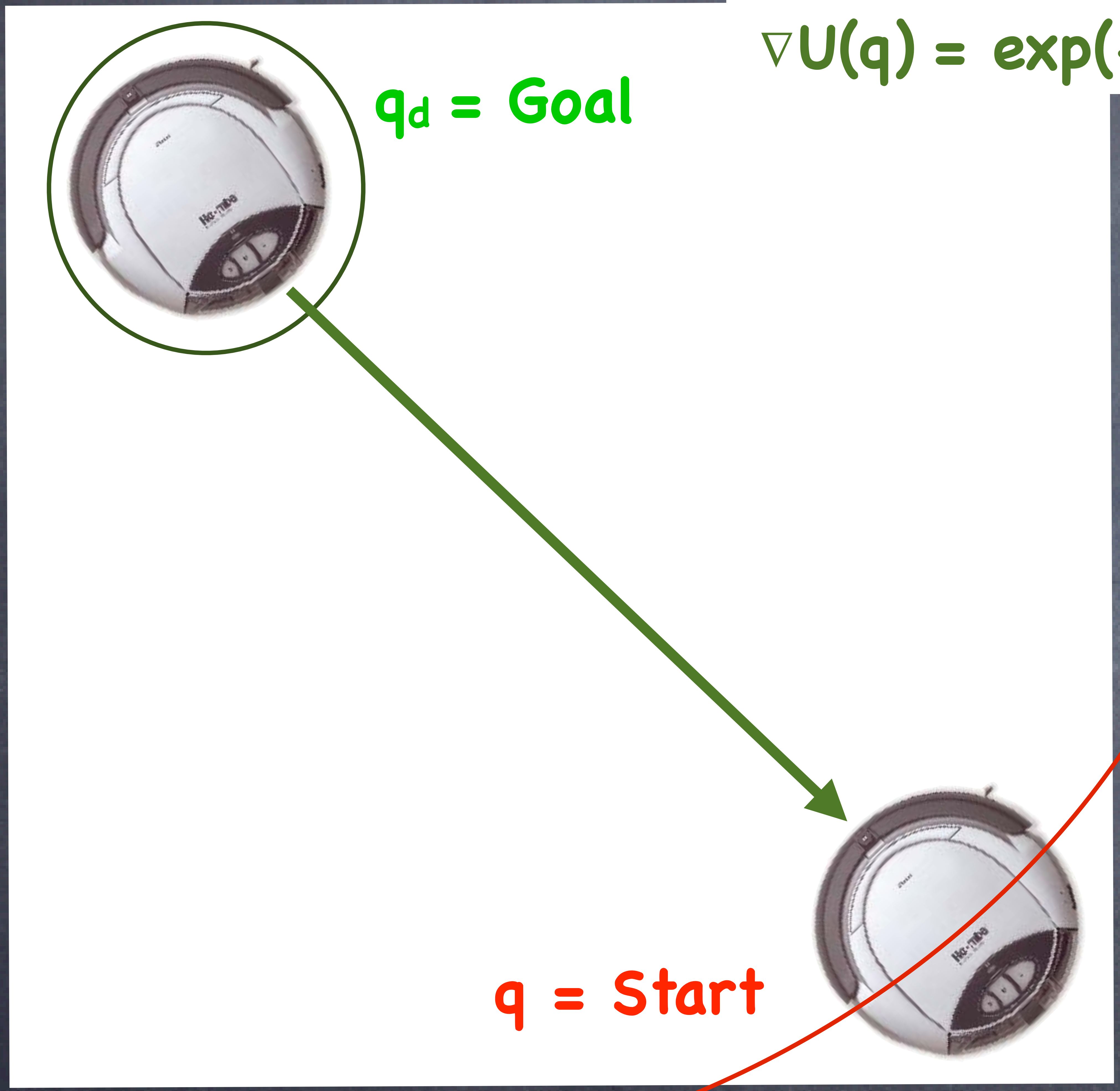
side view

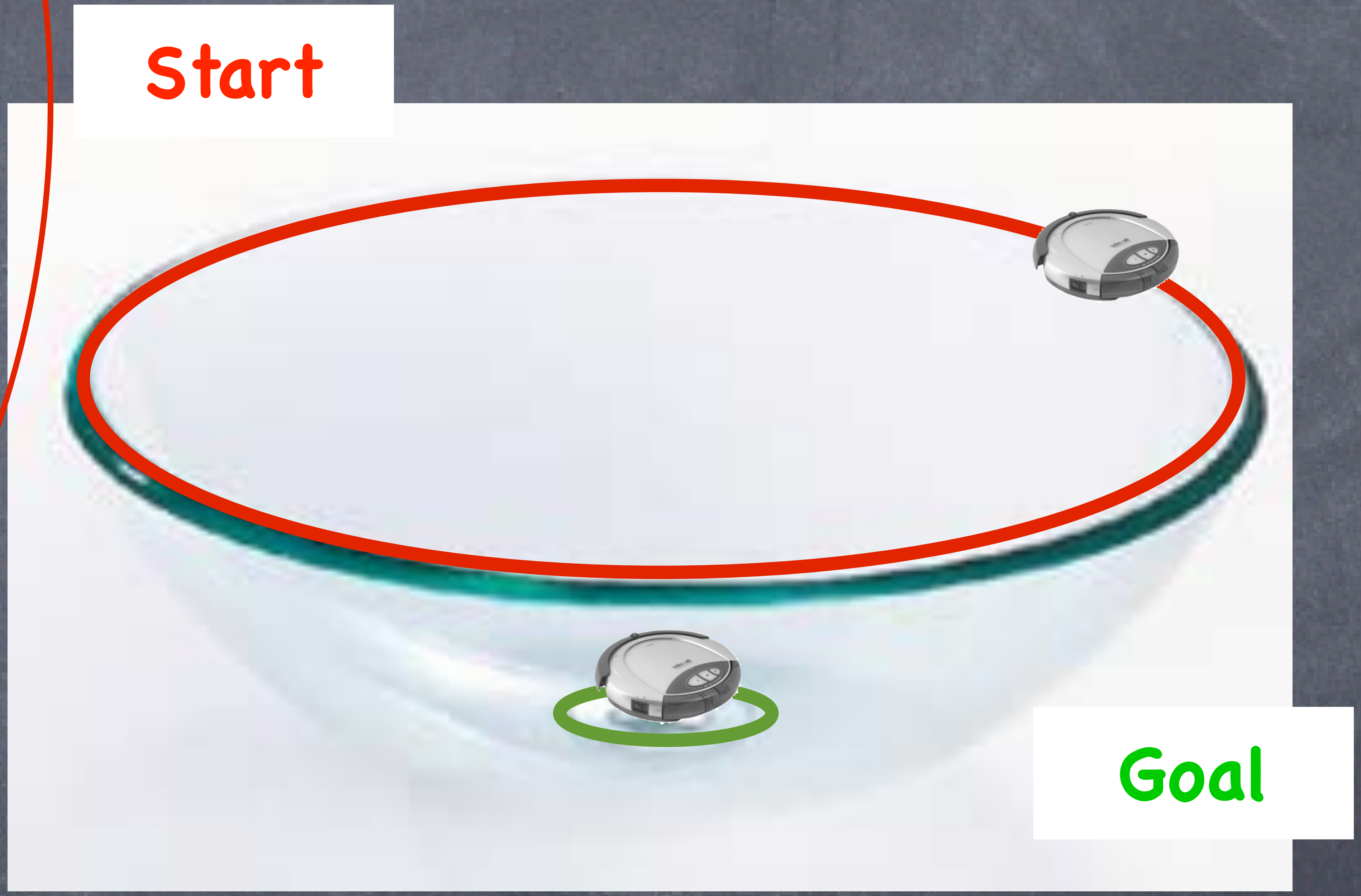
Can we modulate the range of a potential field?

"Bowl" Attractor

$\nabla U(q) = \exp(-||q - q_d||/w)\ (q - q_d)$

top view





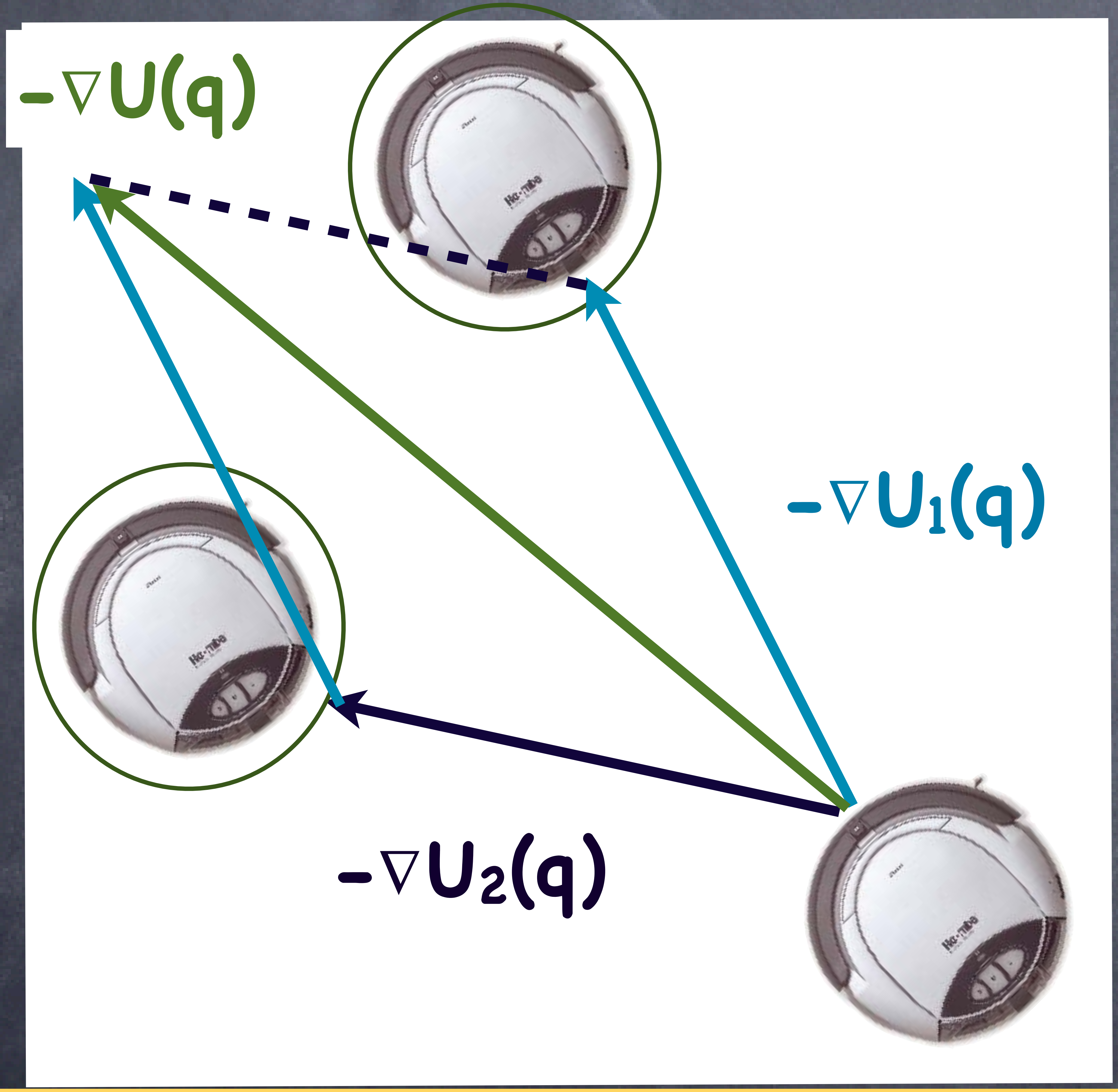
side view

Slide borrowed from Michigan Robotics autorob.org


desmos.com/calculator
Untitled Graph
desmos
Create Account
or
Sign In
exp(-d/w)
weights the influence of a potential
exp(-d/.1)
exp(-d/0.5)
exp(-d/1)
exp(-d/2)
exp(-d/10)
x-axis is d = distance
$e^{-\frac{(x^2)}{10}}$
$e^{-\frac{(x^2)}{4}}$
$e^{-\frac{(x^2)}{2}}$
$e^{-\frac{(x^2)}{1}}$
$e^{-\frac{(x^2)}{0.5}}$
$e^{-\frac{(x^2)}{0.25}}$
-5
-4
-3
-2
-1
0
1
2
3
4
2
-1
-2

Can we combine multiple potentials?

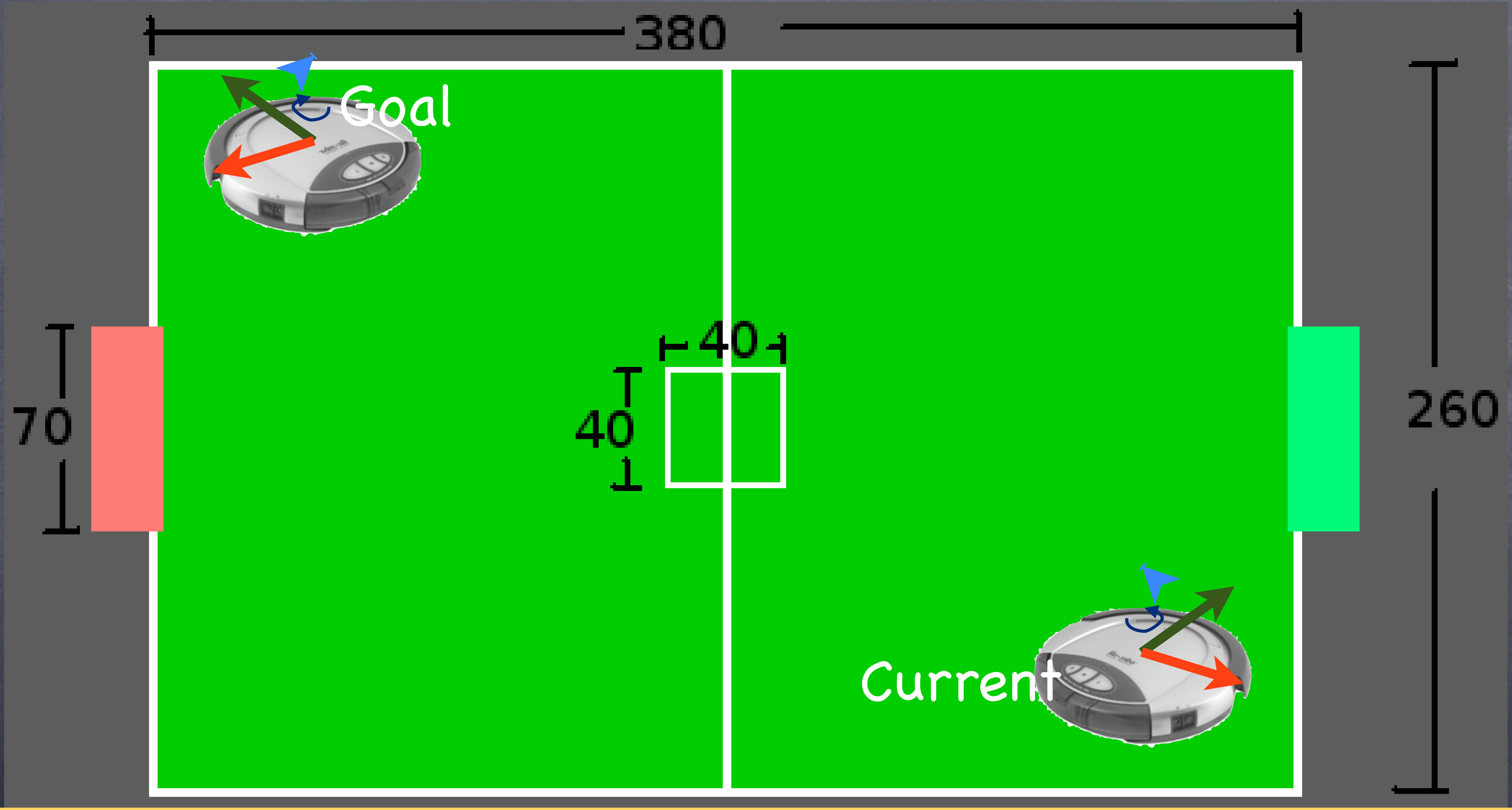
Multiple potentials



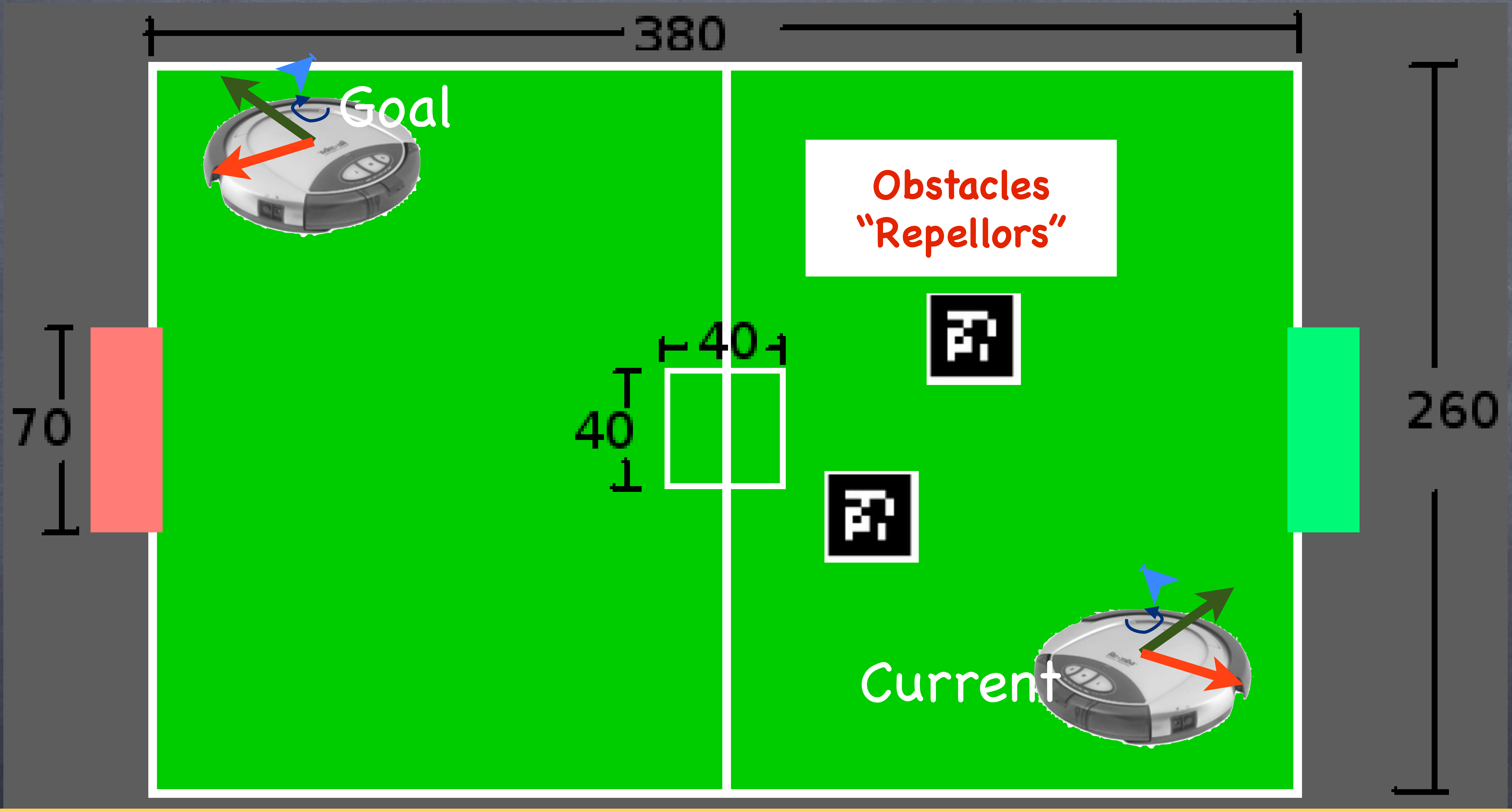
- Output of potential field is a vector
- Combine multiple potentials through vector summation

$$U(q) = \Sigma_i U_i(q)$$

describe performance for this case

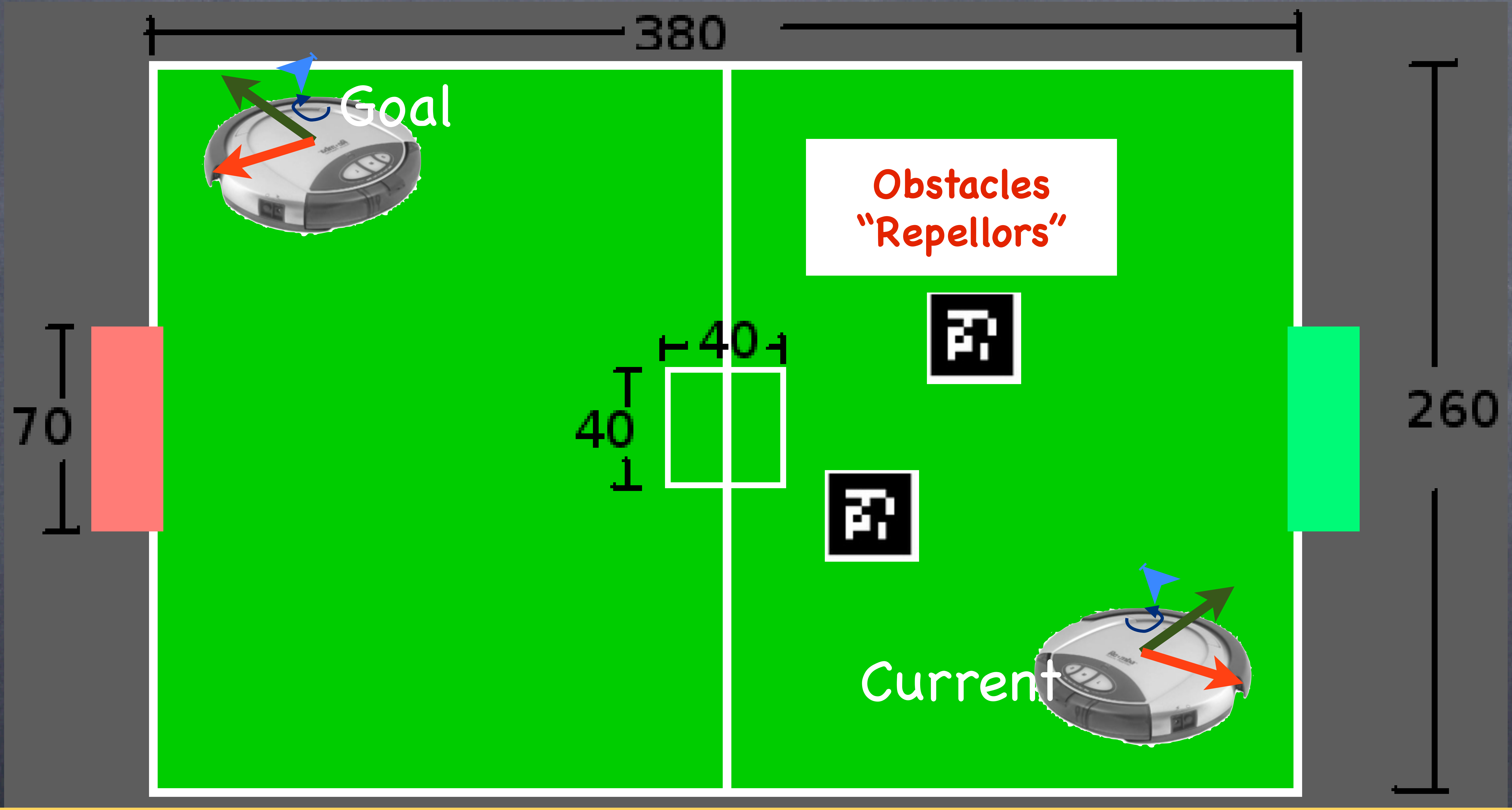


describe performance for this case



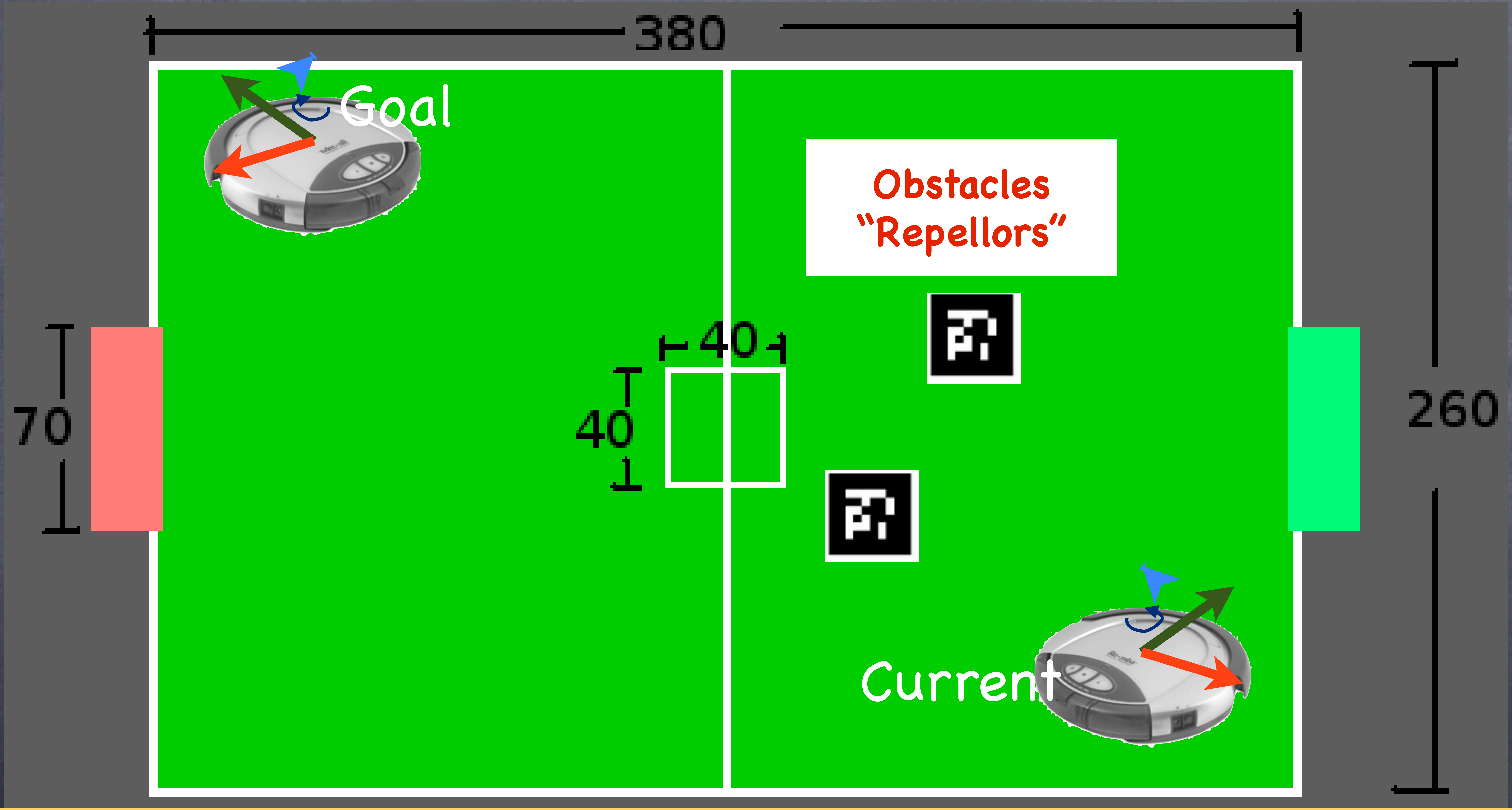
describe performance for this case

how do we deal with repellors?

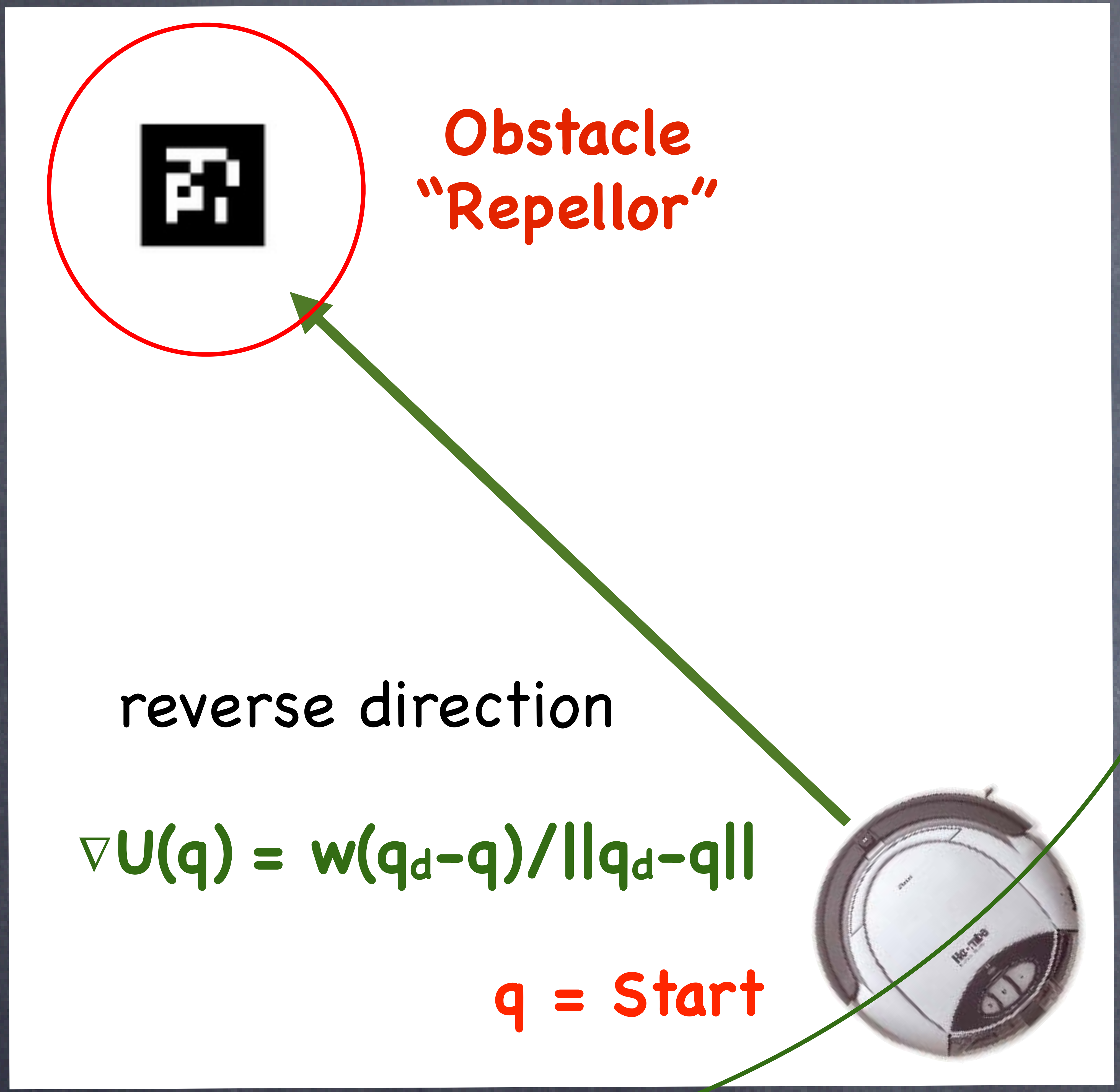


add sum of repulsive potentials

$$U(q) = U_{attracts}(q) + U_{repellors}(q)$$



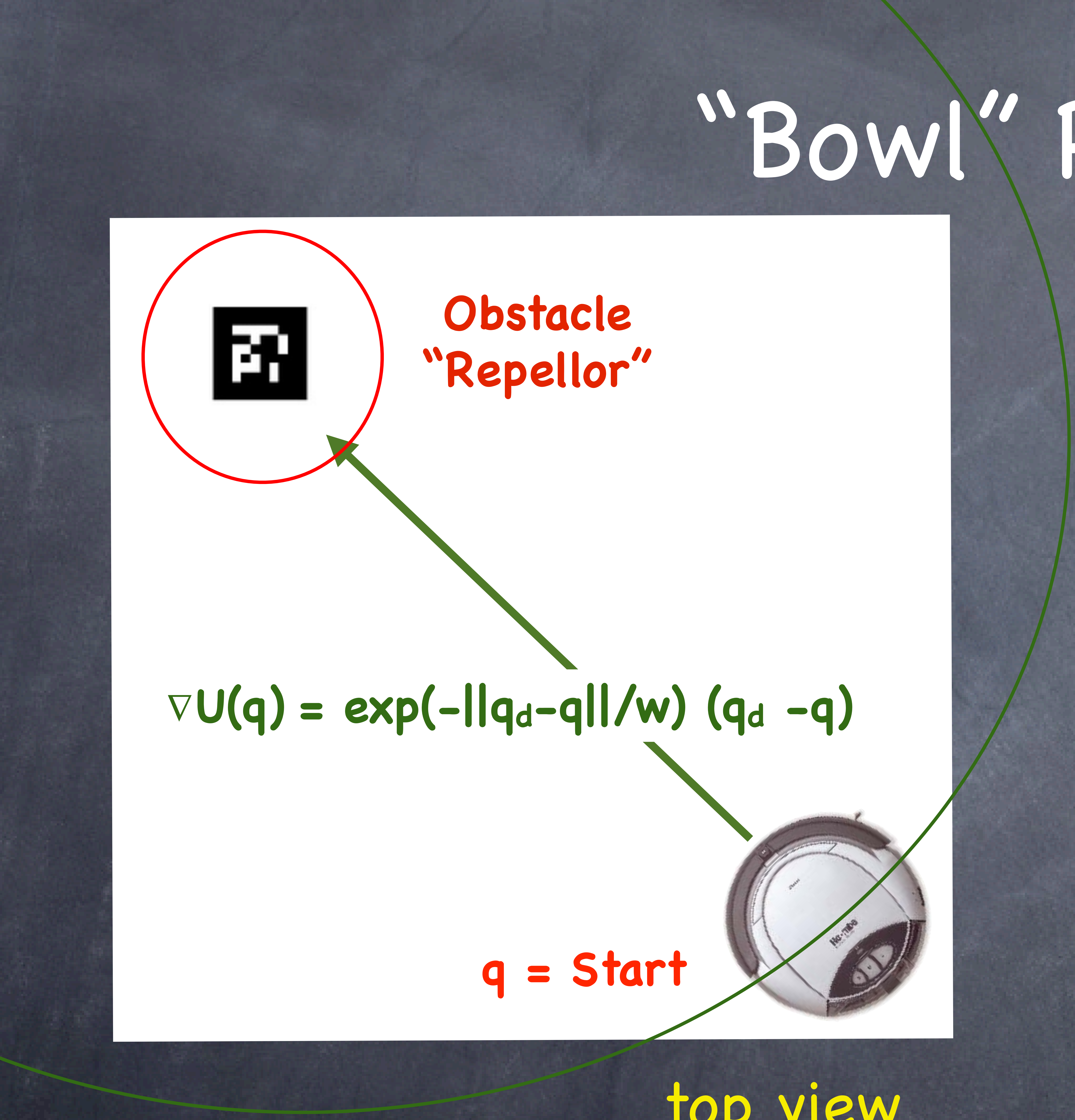
“Cone” Repellor



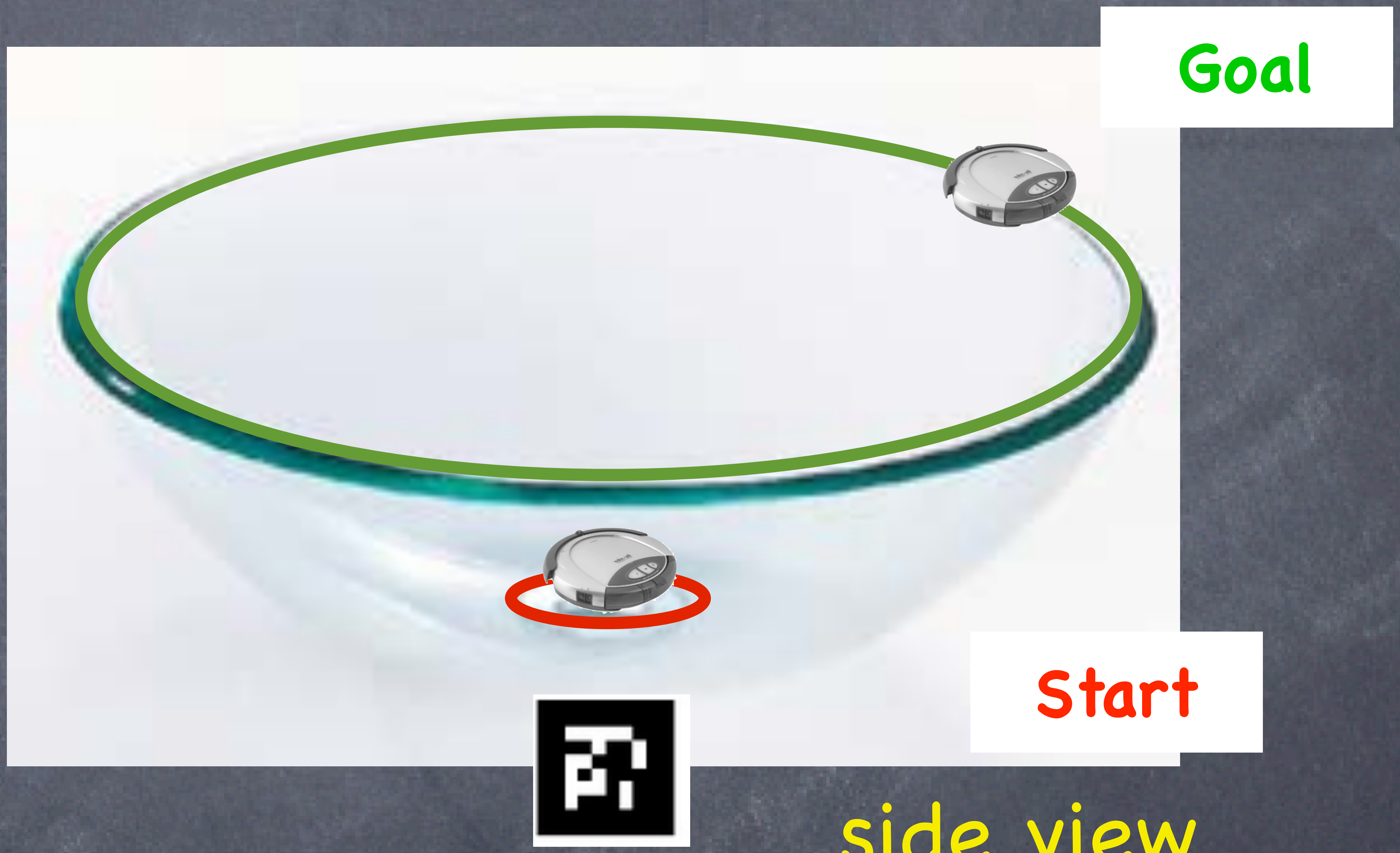
potential problems?

side view

“Bowl” Repellor

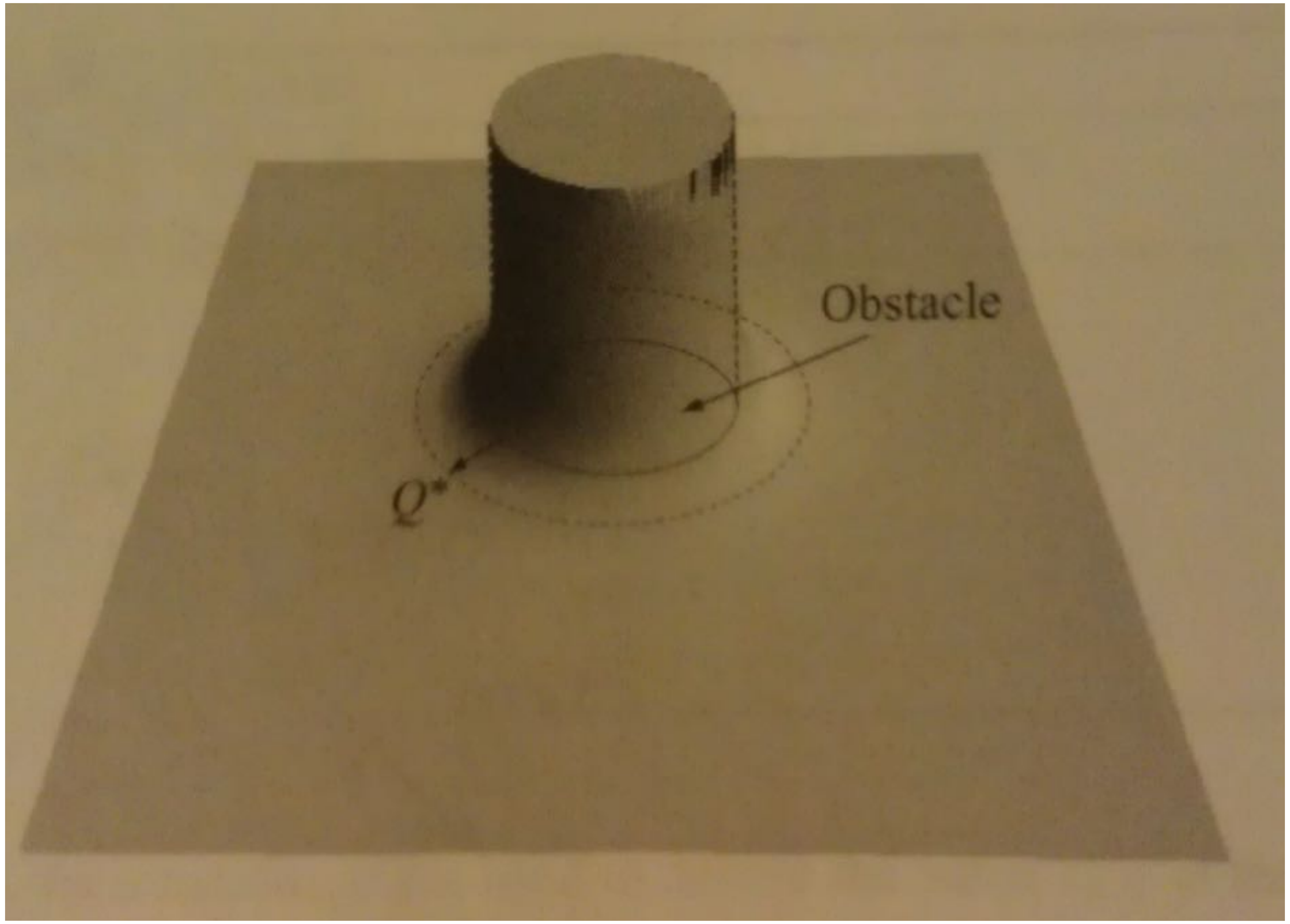


top view



side view

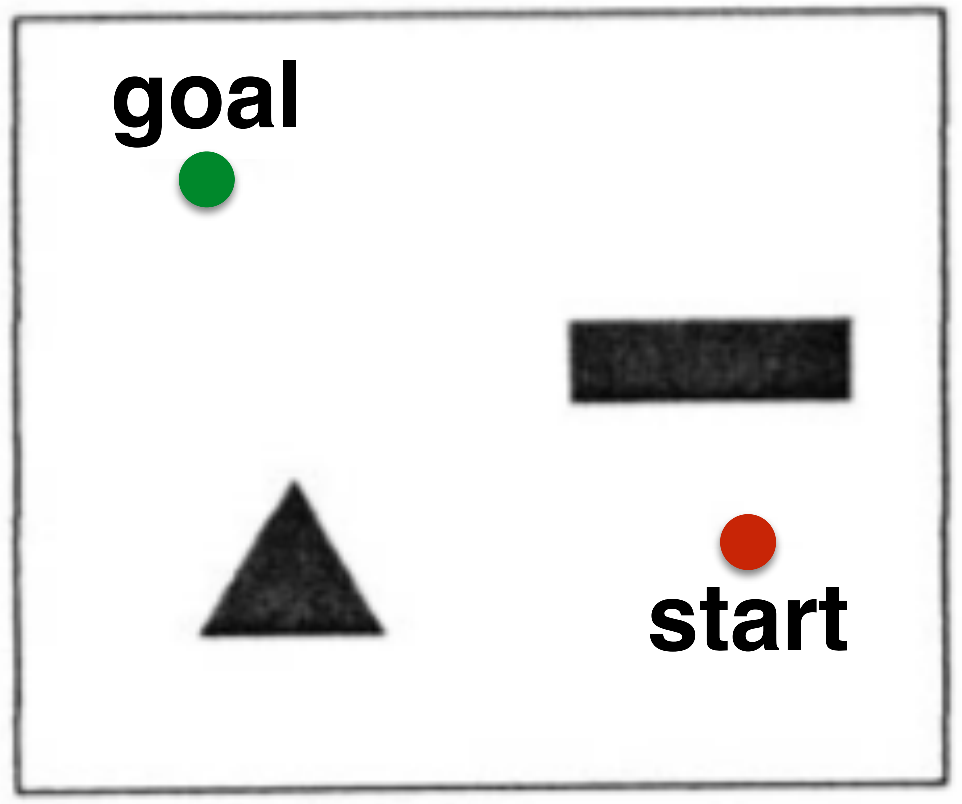
repellor should only have local influence,
repelling only around boundary improves path

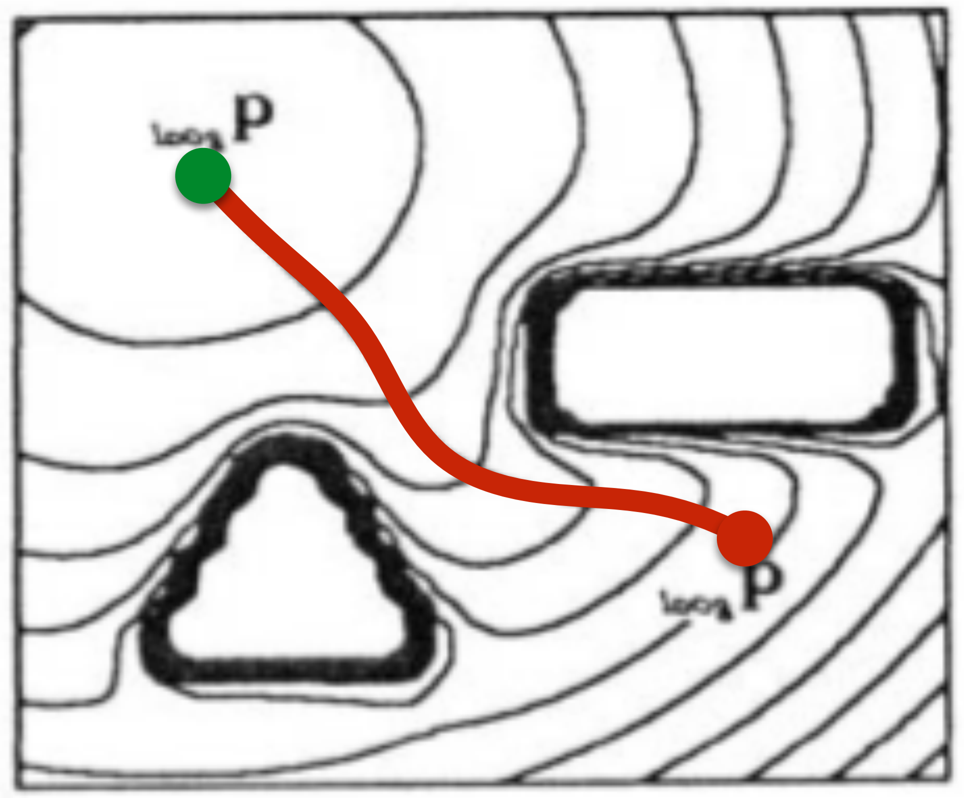


Slide borrowed from Michigan Robotics autorob.org

2 Obstacle example

configuration space





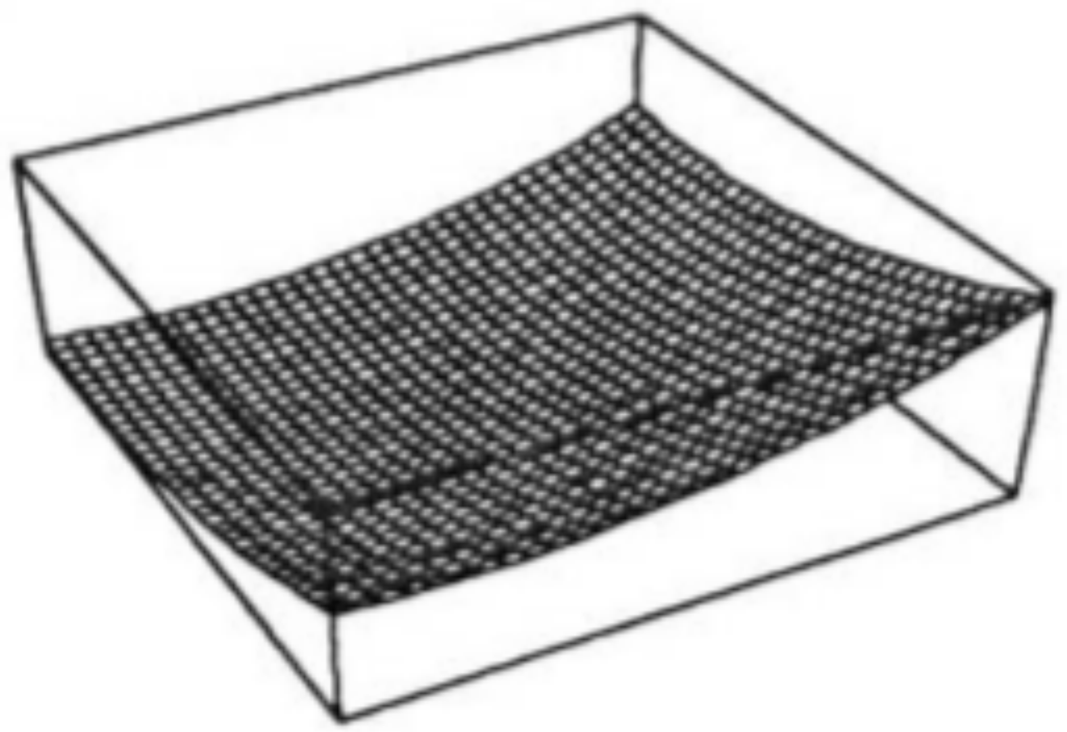
path from descent on gradient field

resulting gradient field

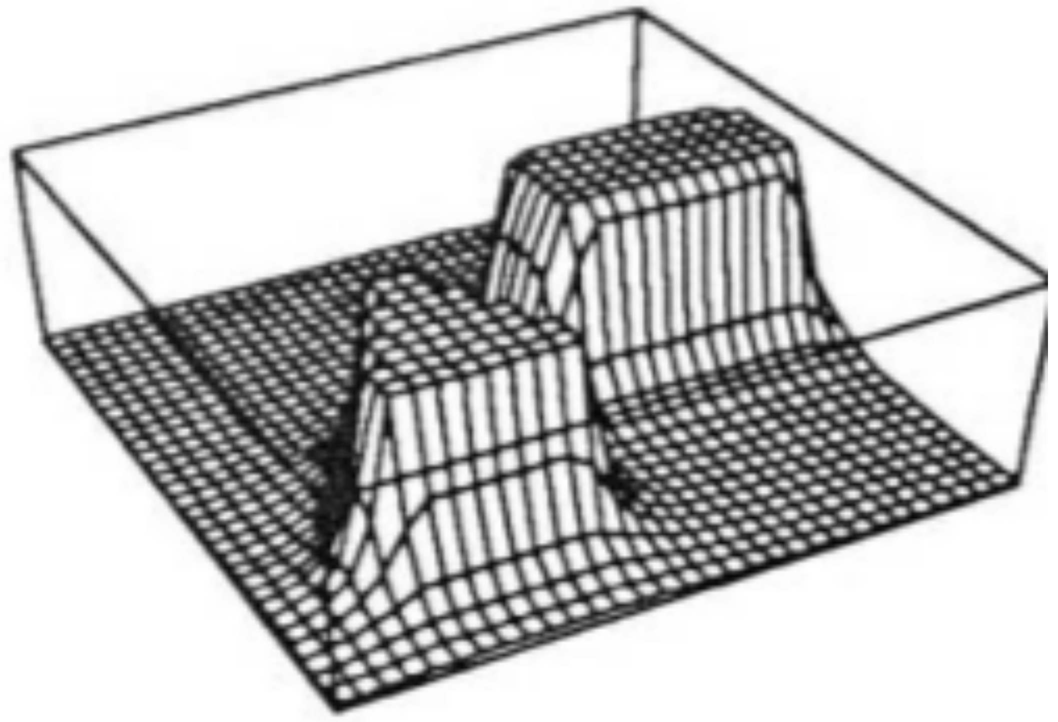
attractor field

repellor fields

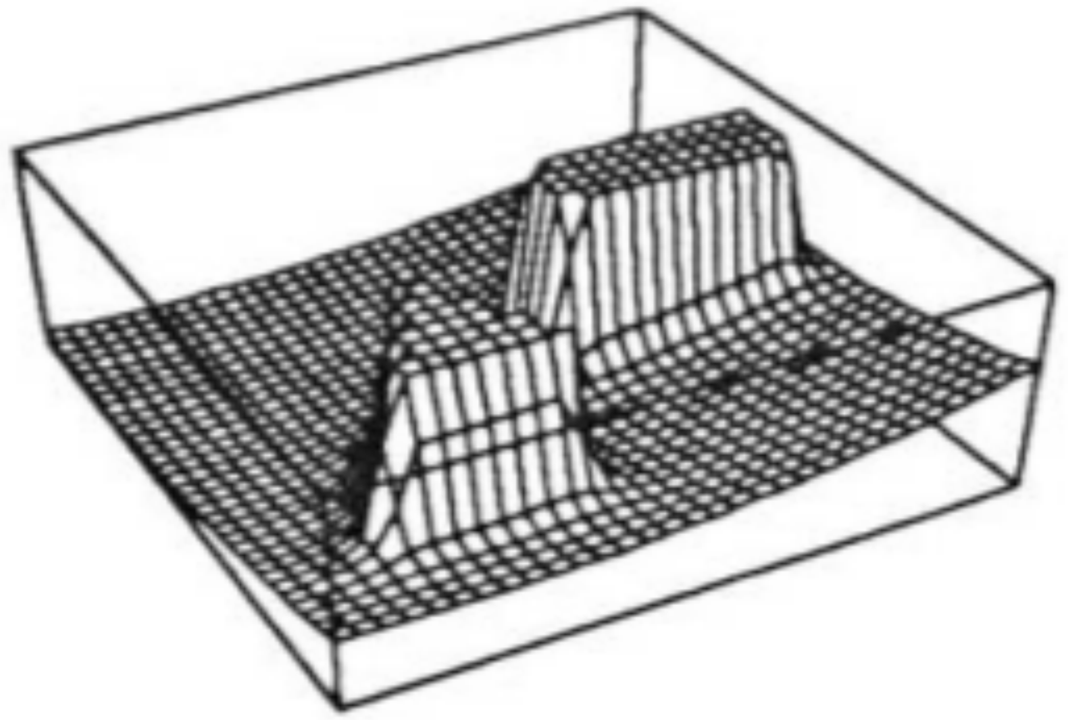
combined potential

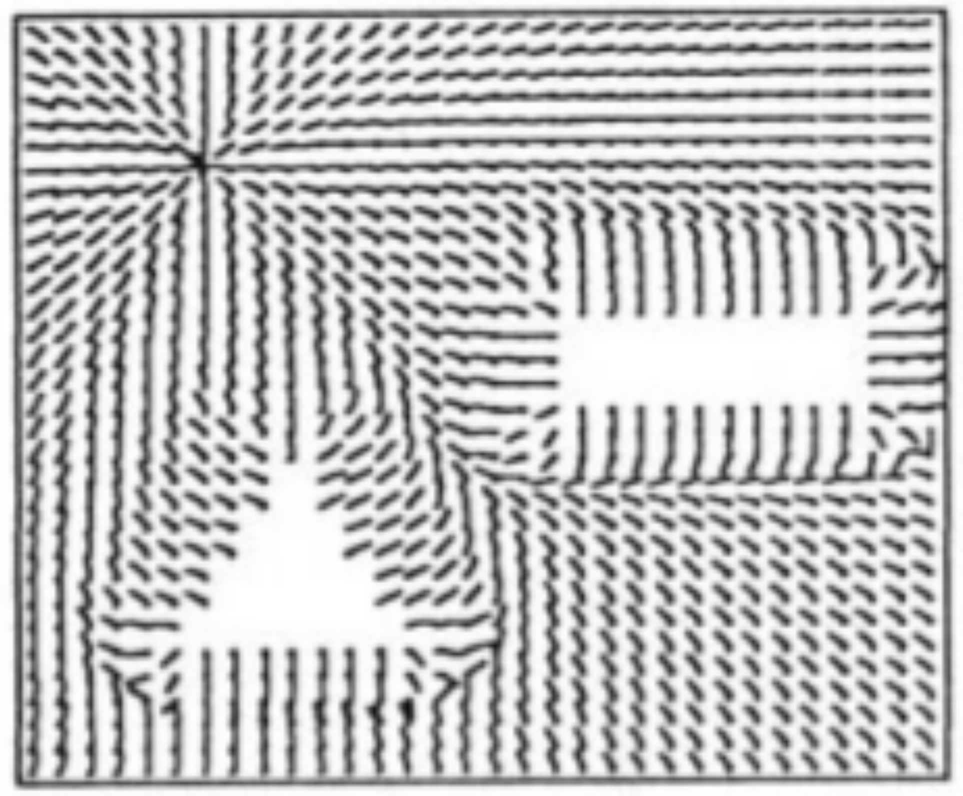


+

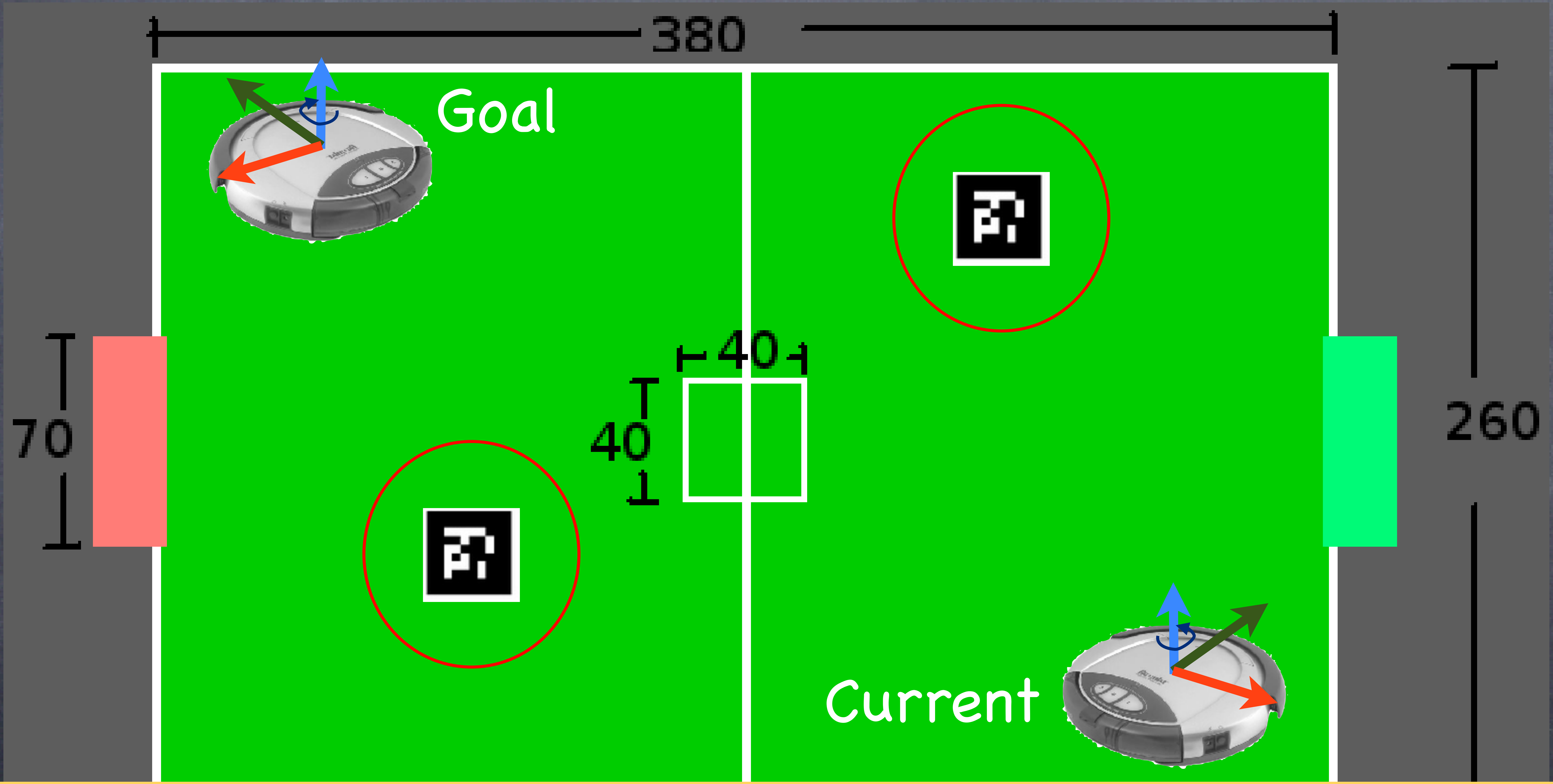


=

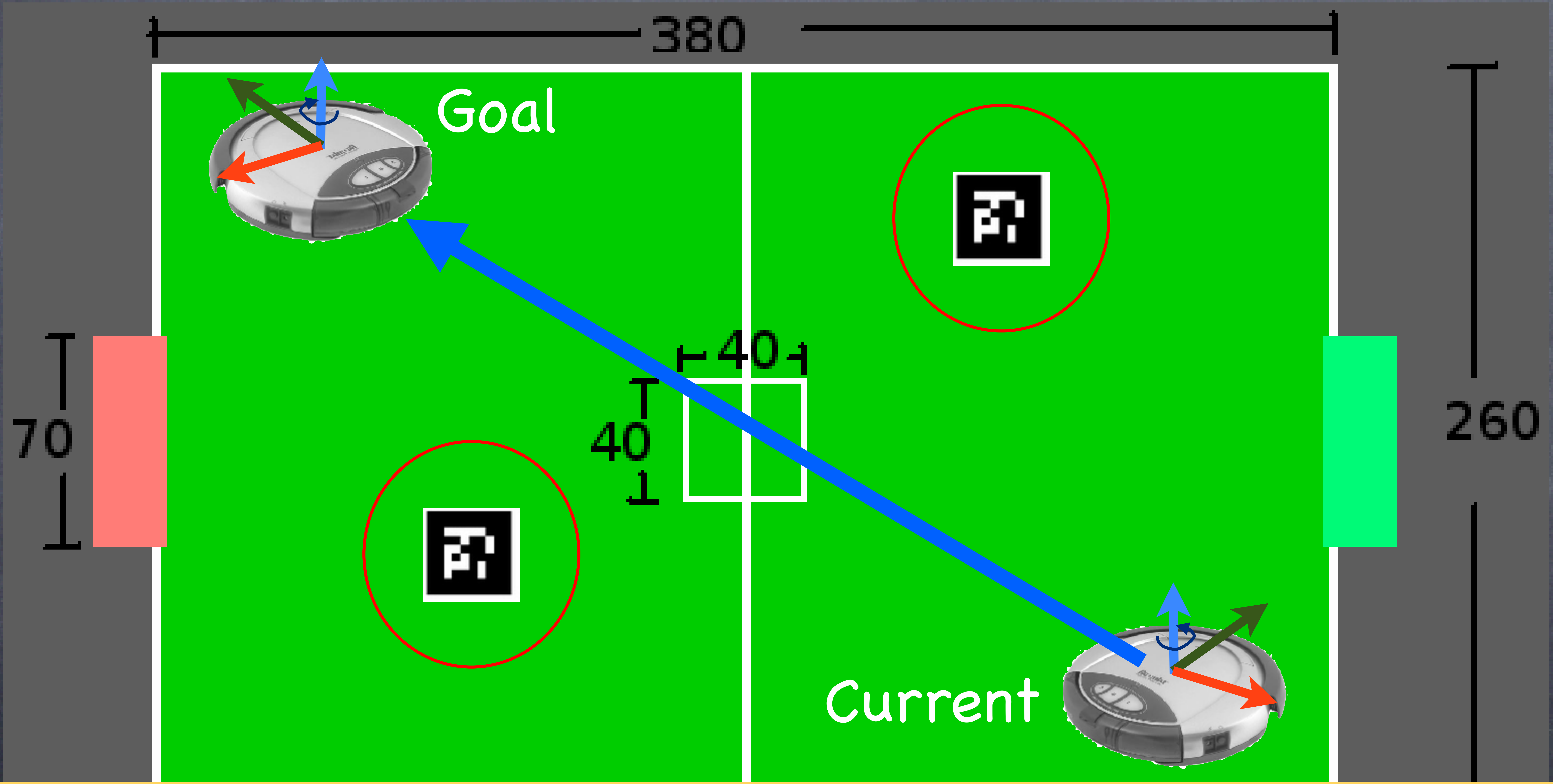




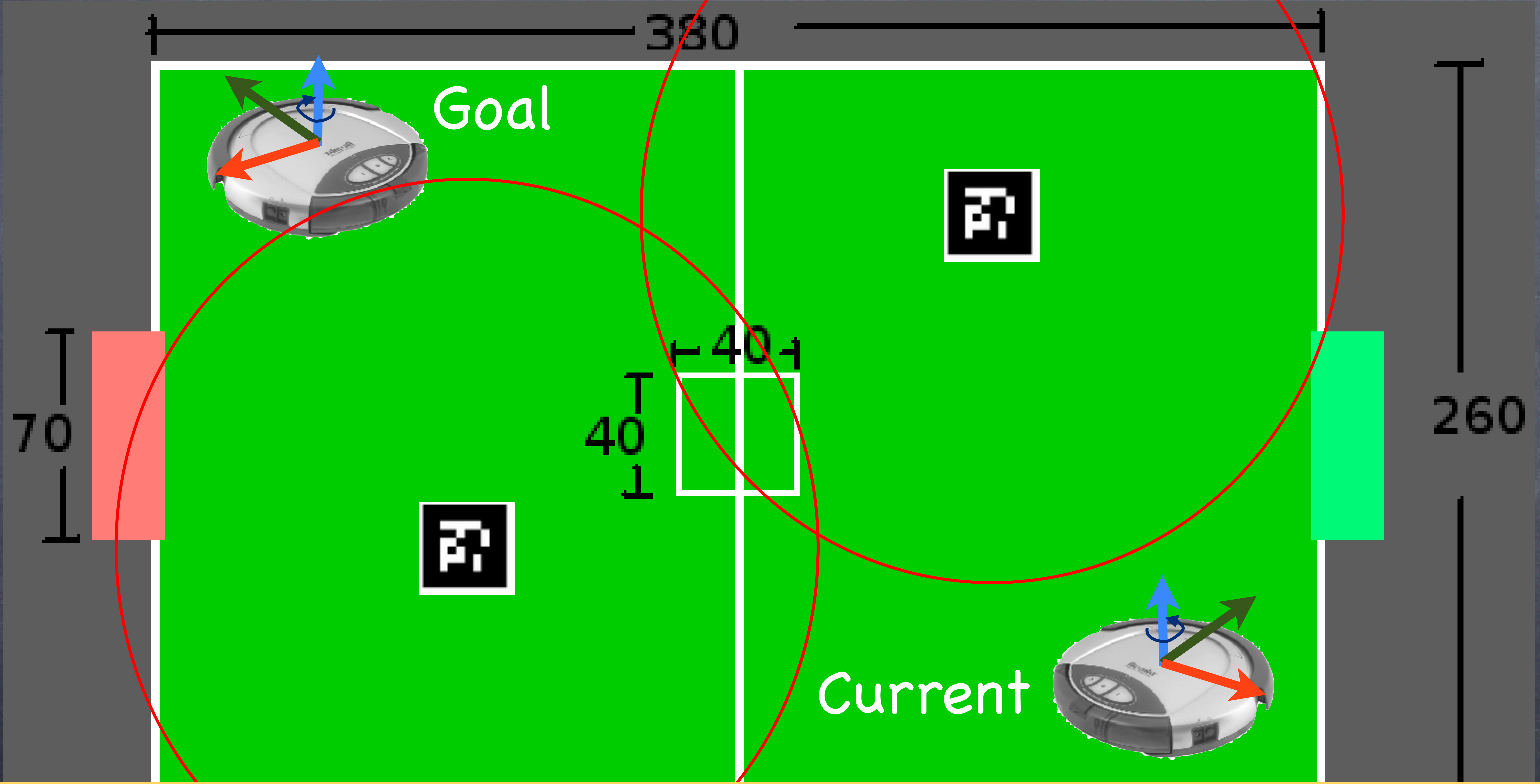
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



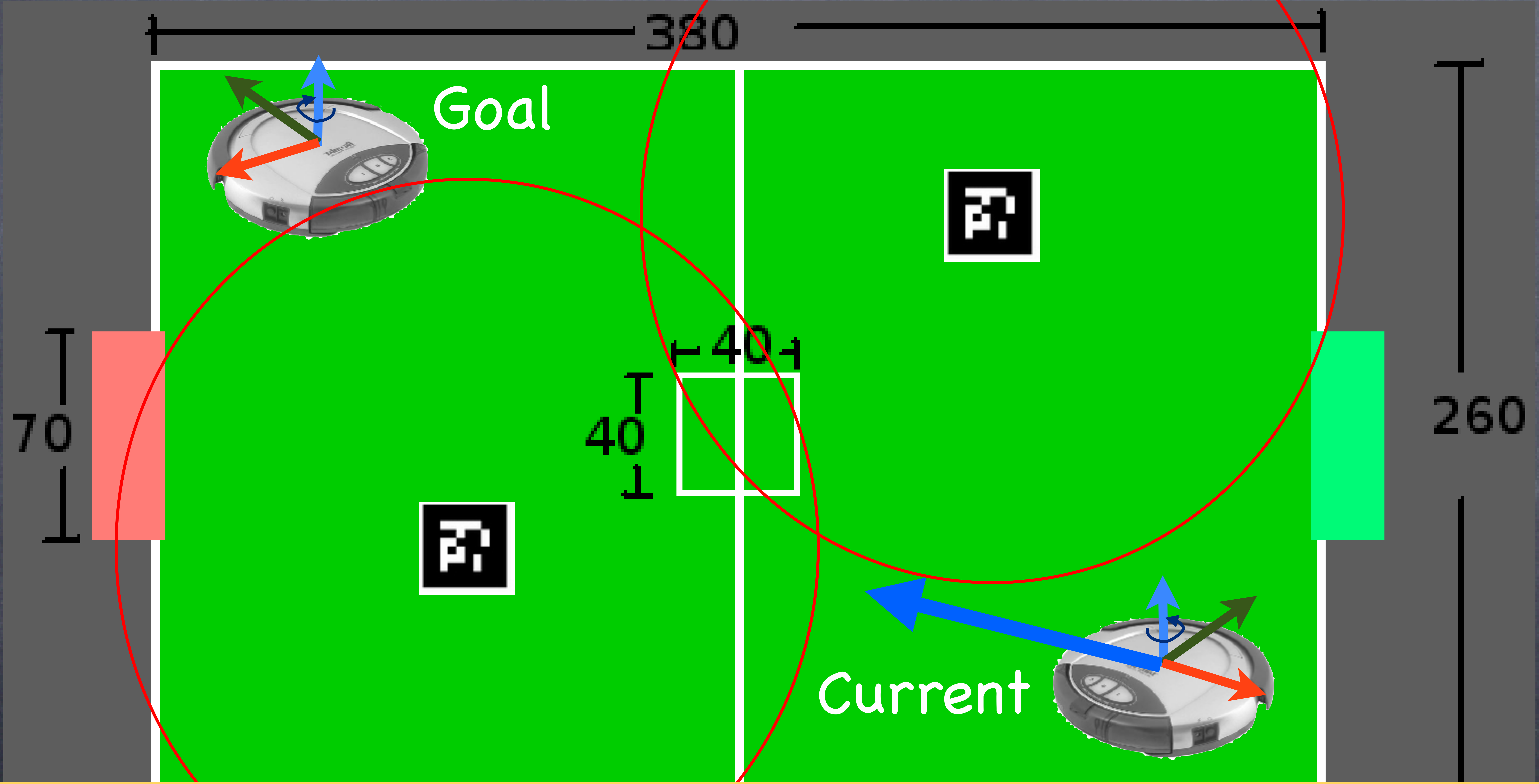
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



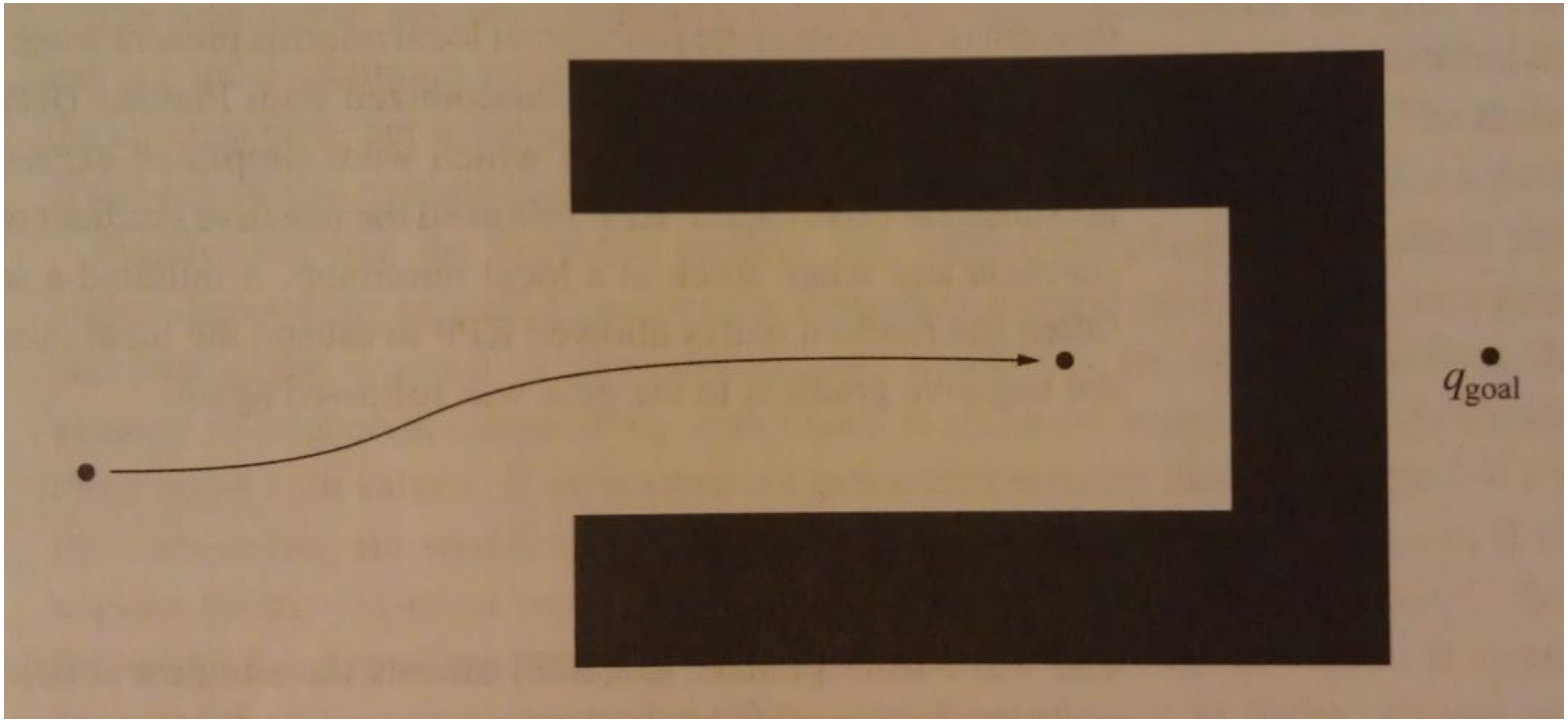
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight

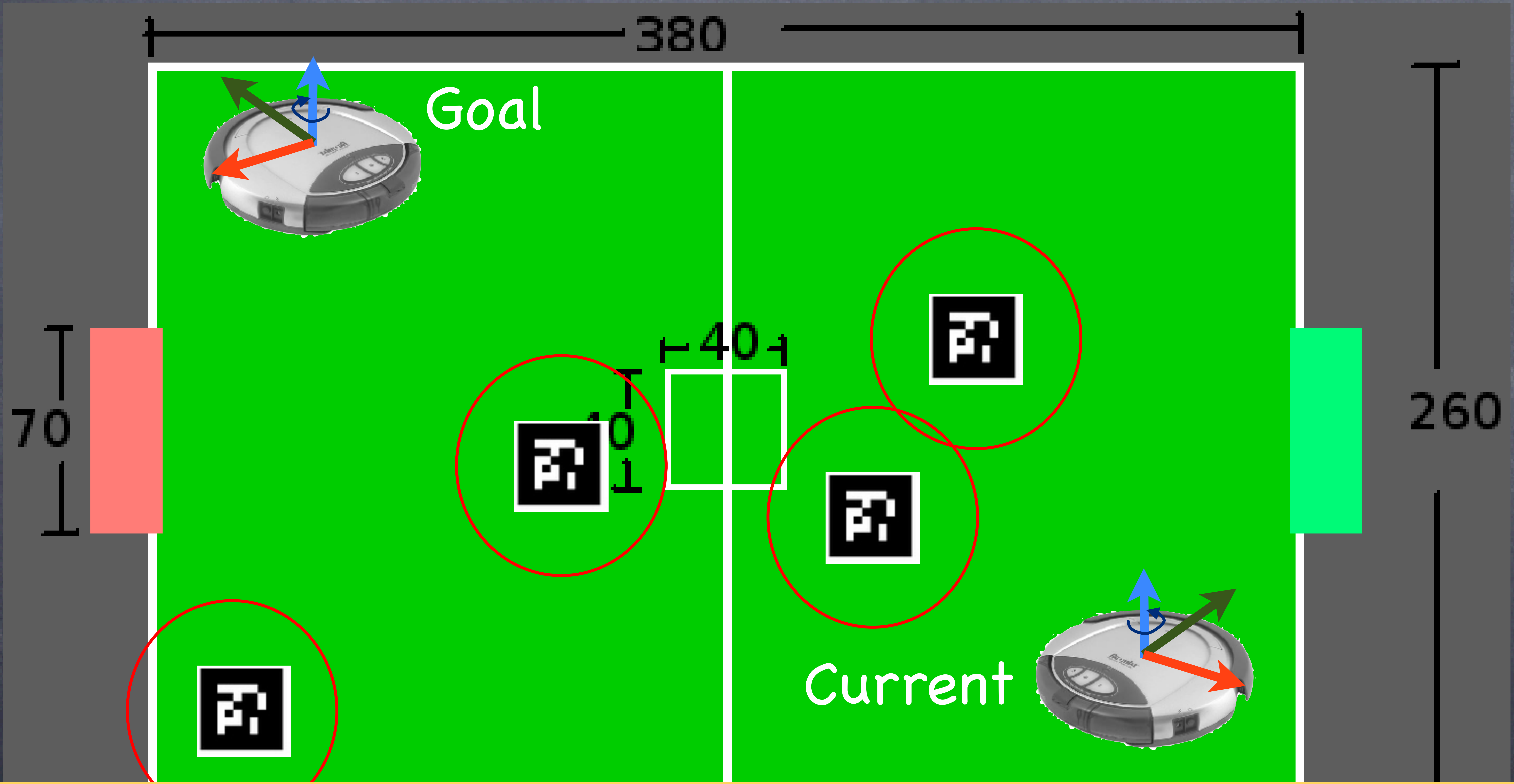


Local Minima

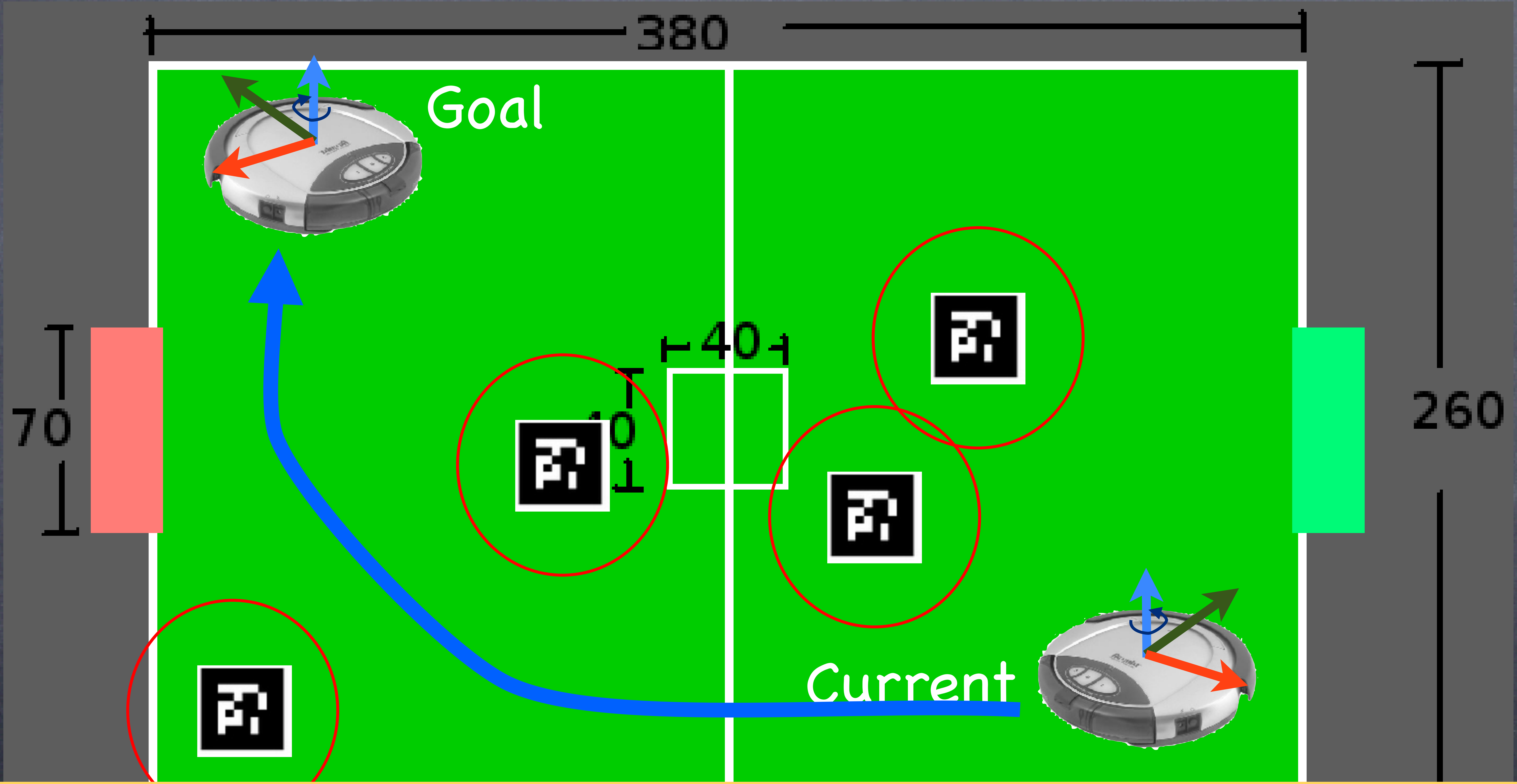


Slide borrowed from Michigan Robotics autorob.org

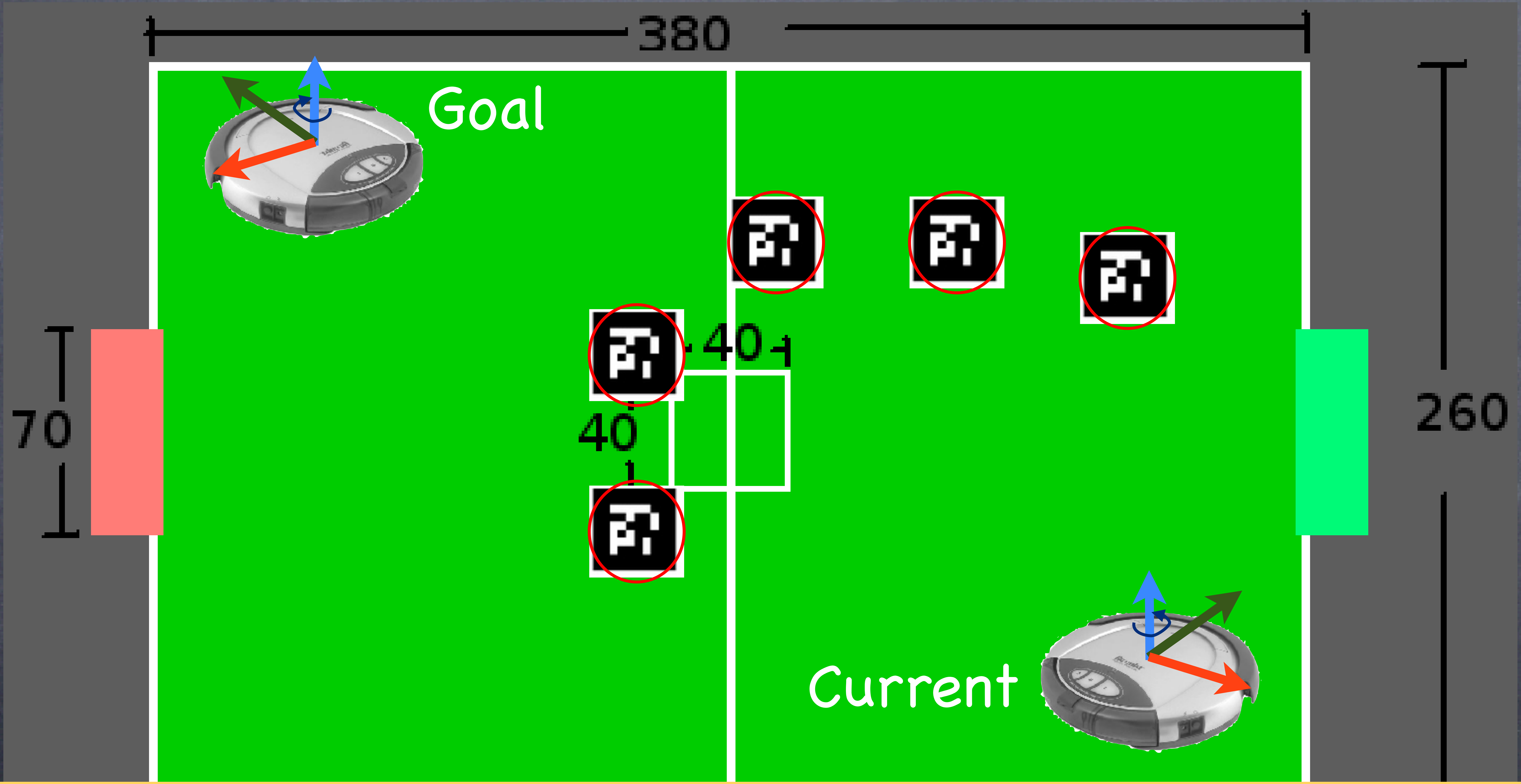
describe performance for this case



describe performance for this case

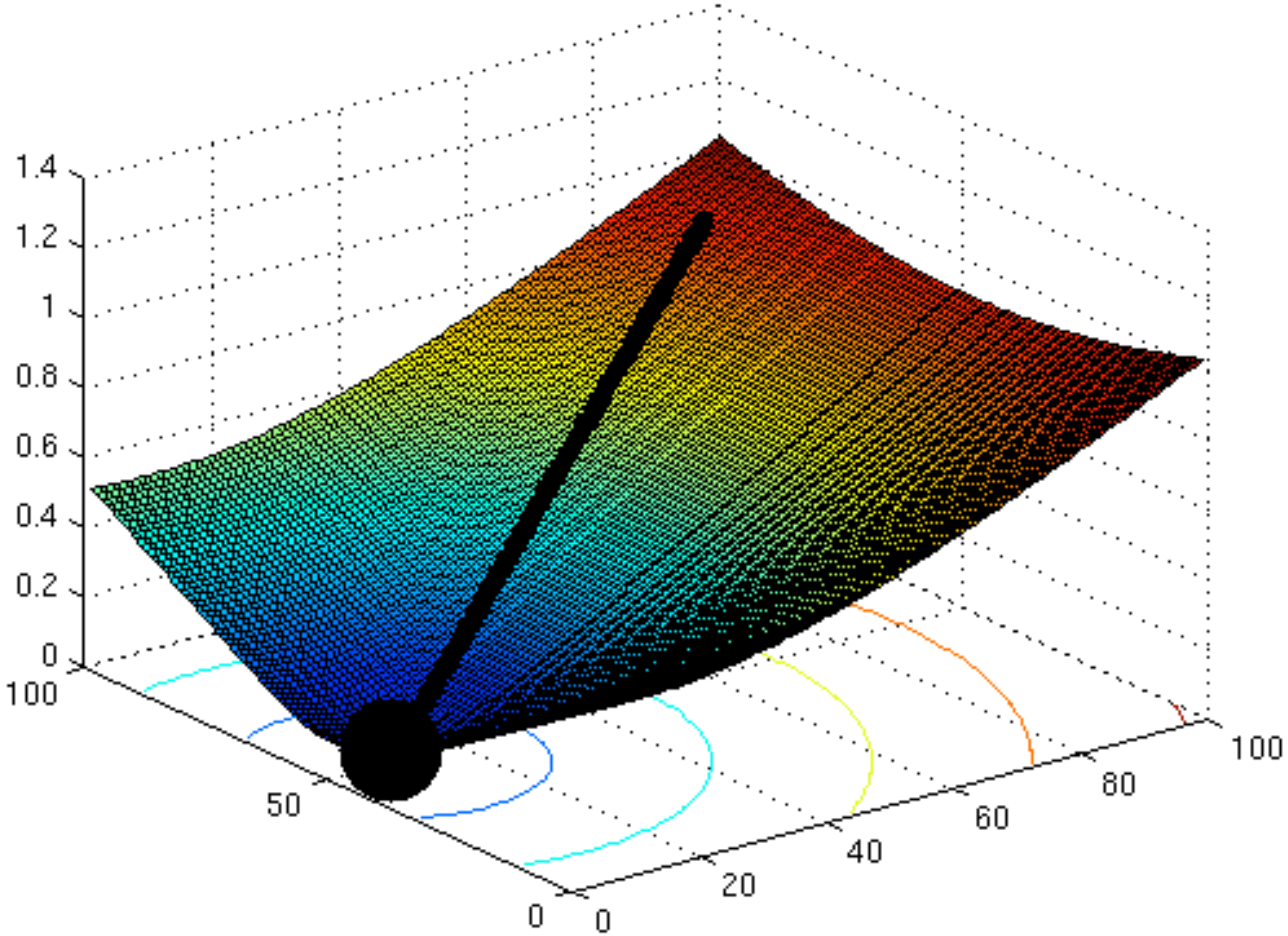


describe performance for this case



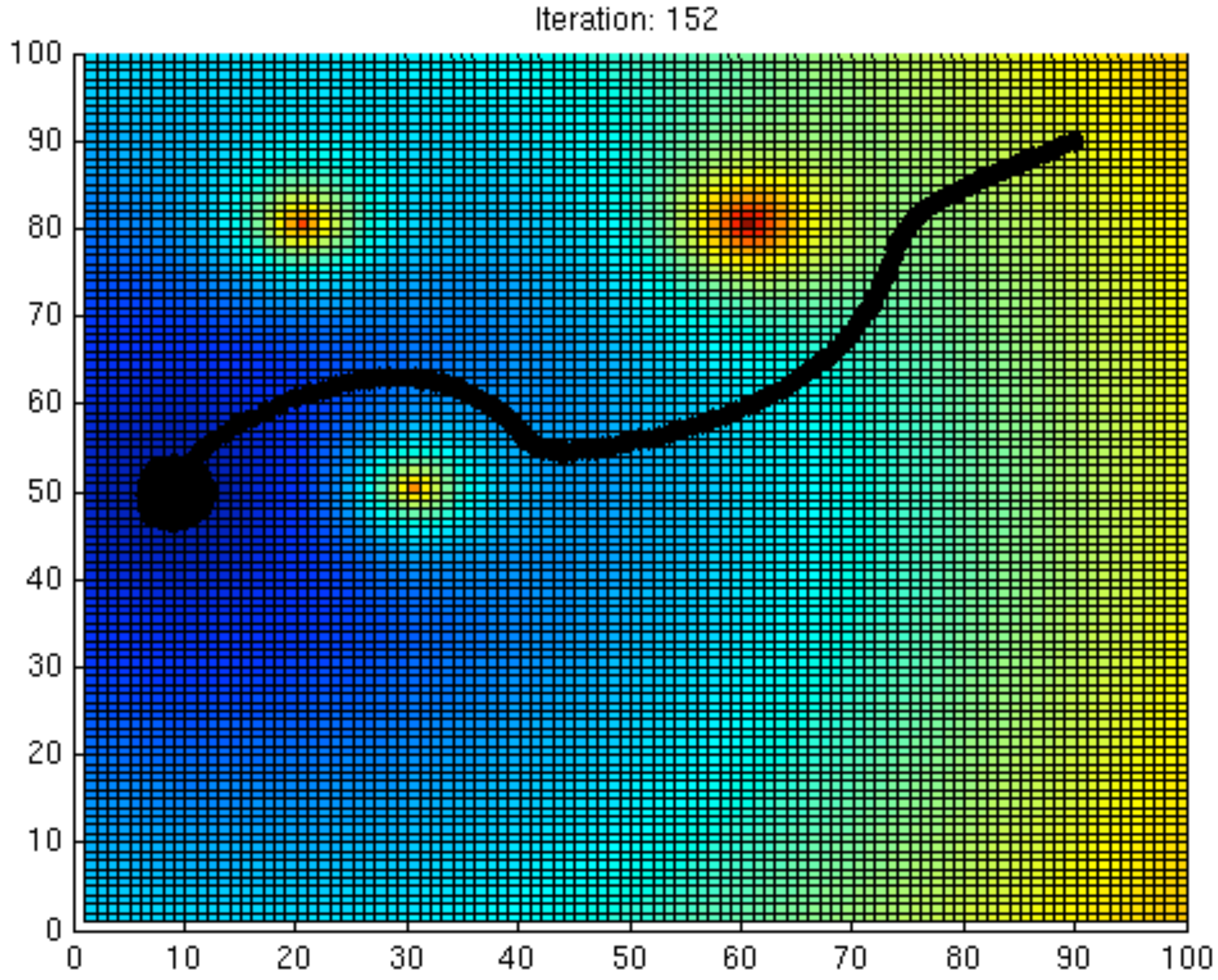
Slide borrowed from Michigan Robotics autorob.org

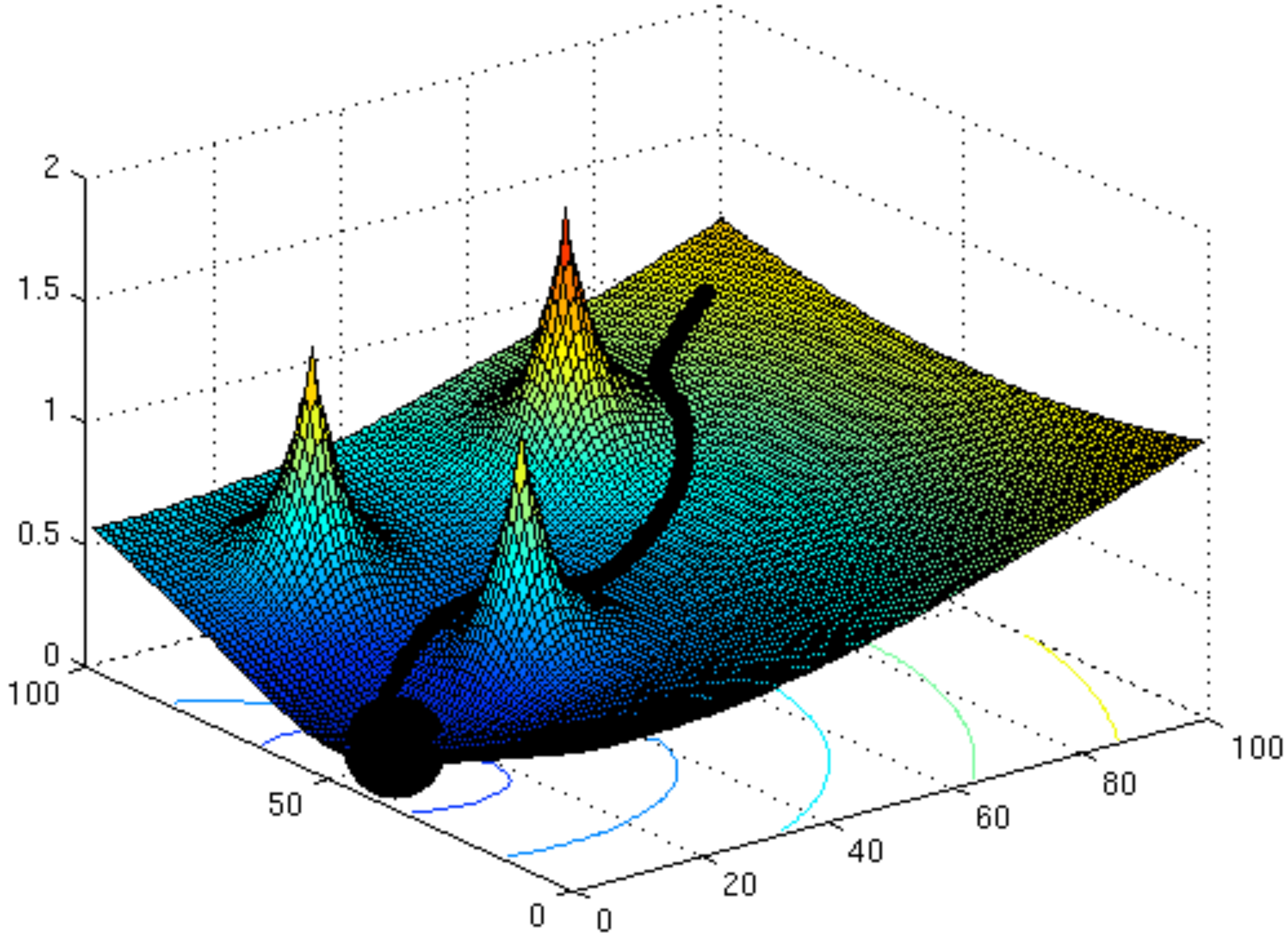
matlab example



pfield.m [1 5 8 12]

matlab example

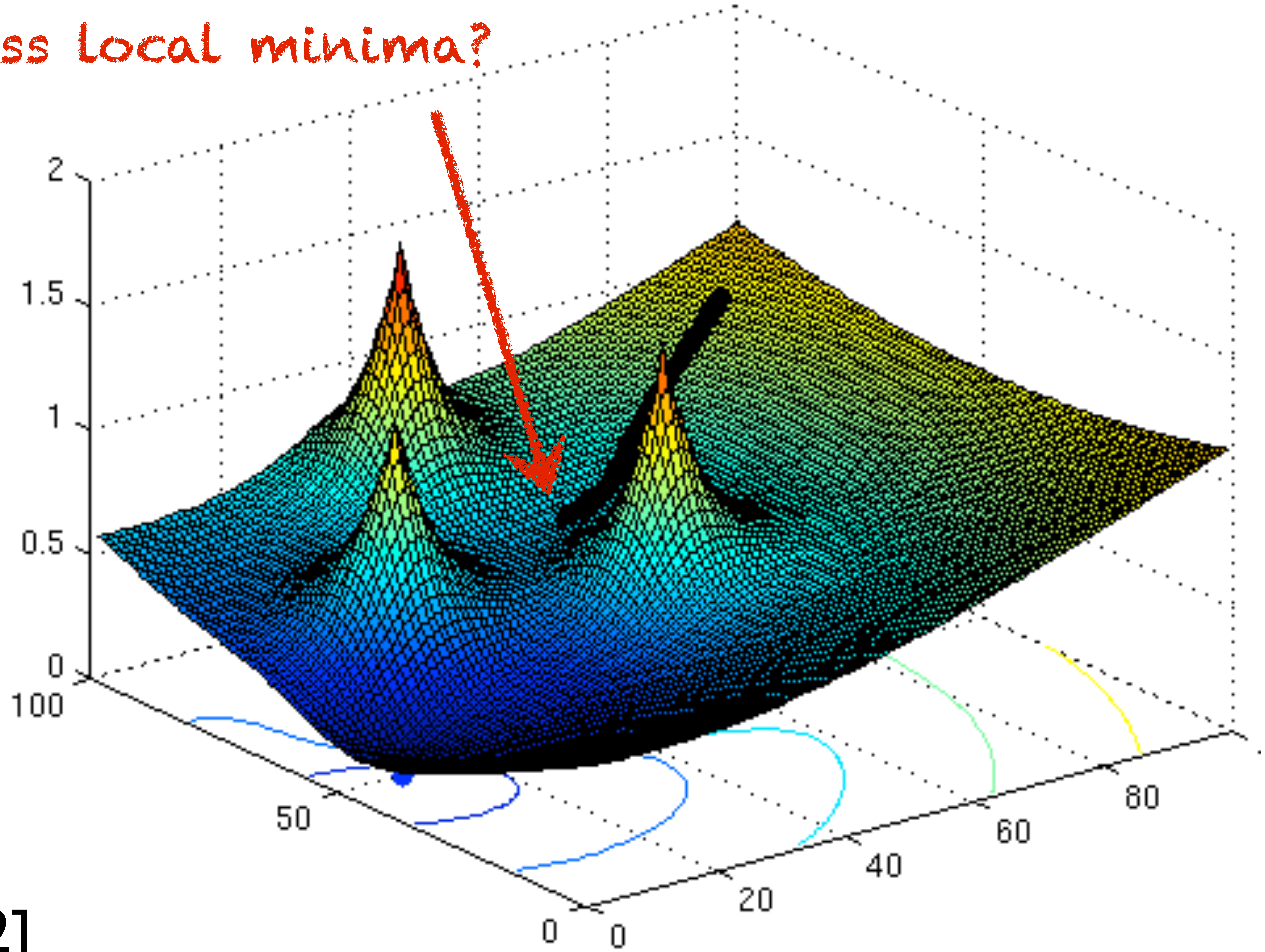




pfield.m [1 5 8 12]

matlab example

How to address local minima?



pfield.m [1 5 8 12]

How can we get out of local minima?

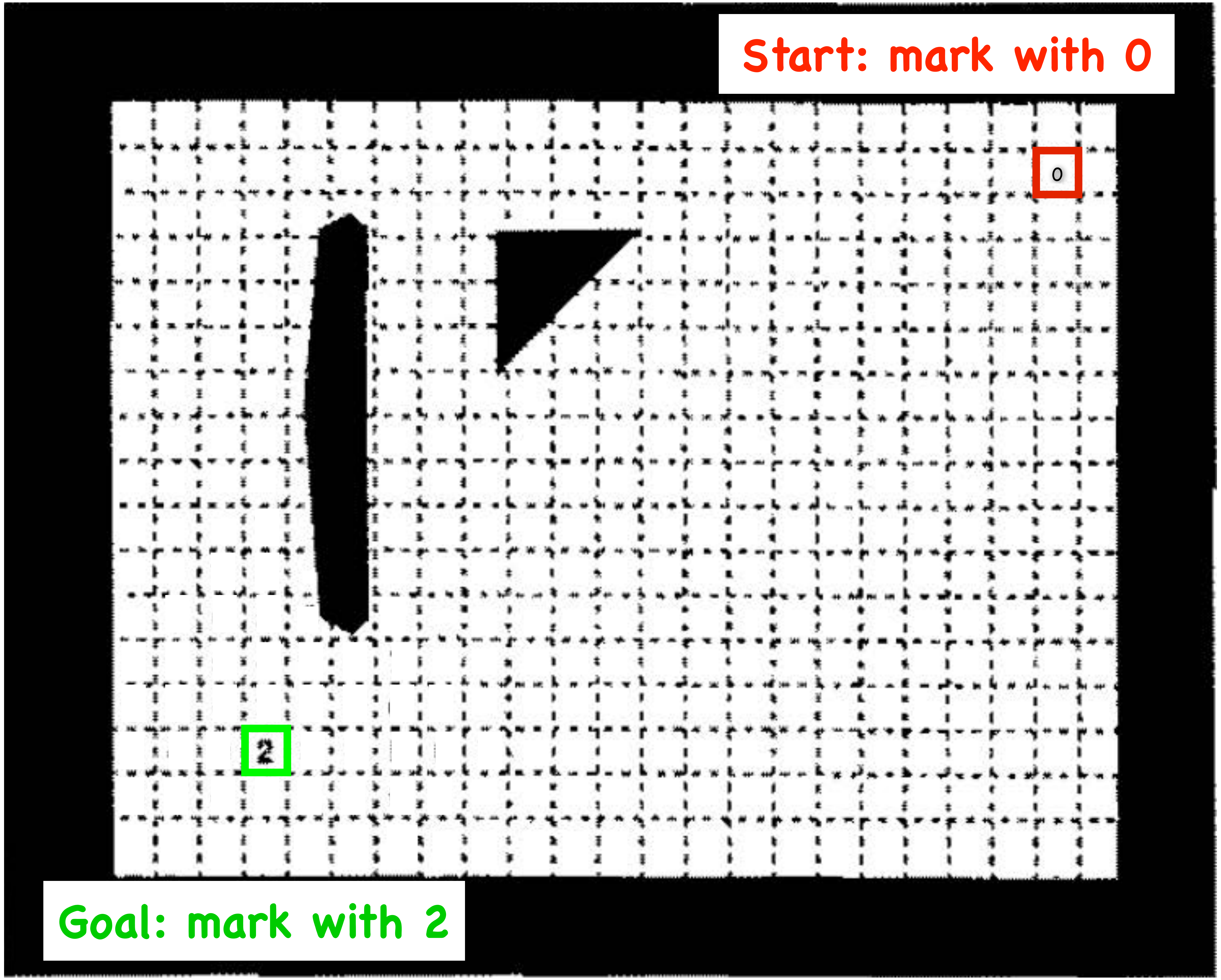
How can we get out of local minima?

Go back to planning.

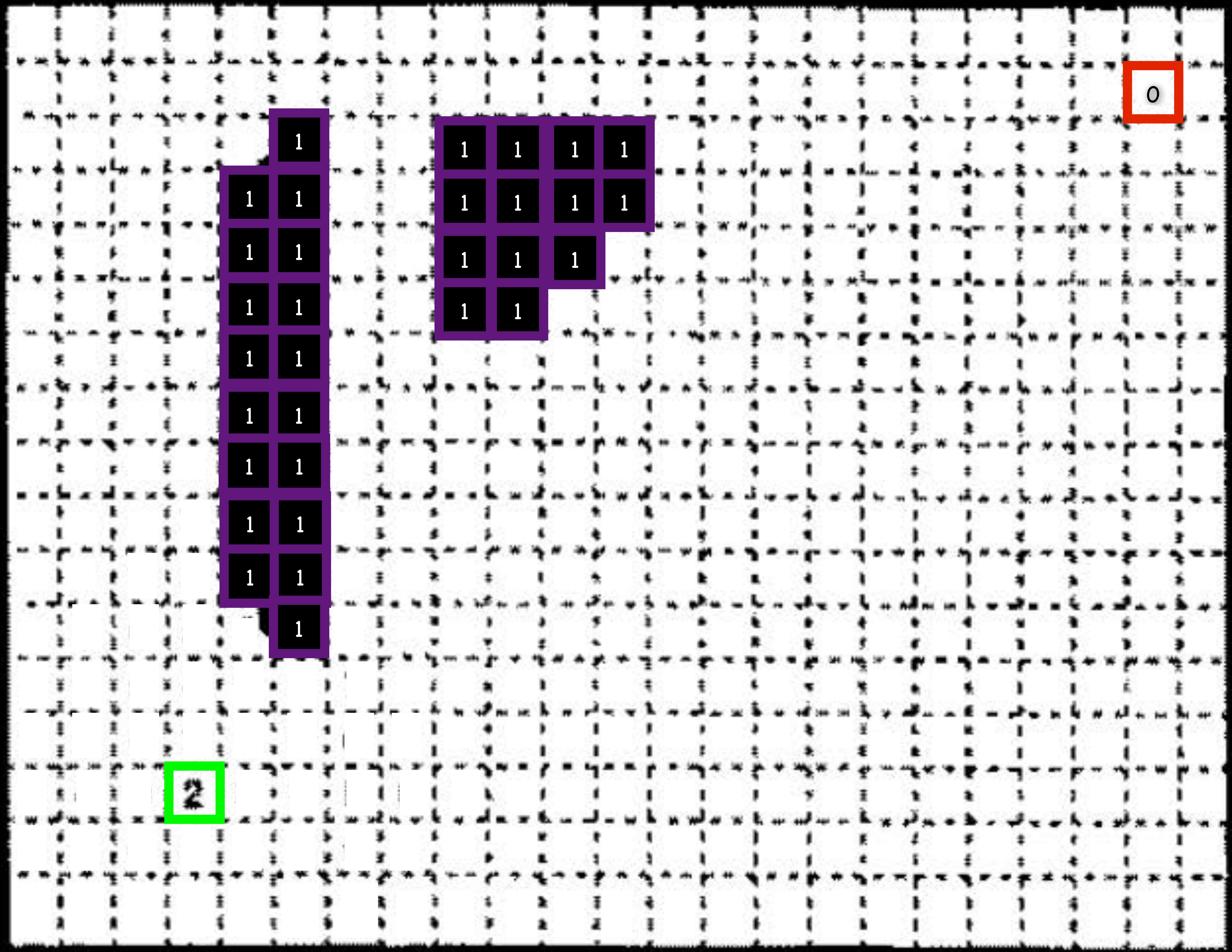
Wavefront Planning

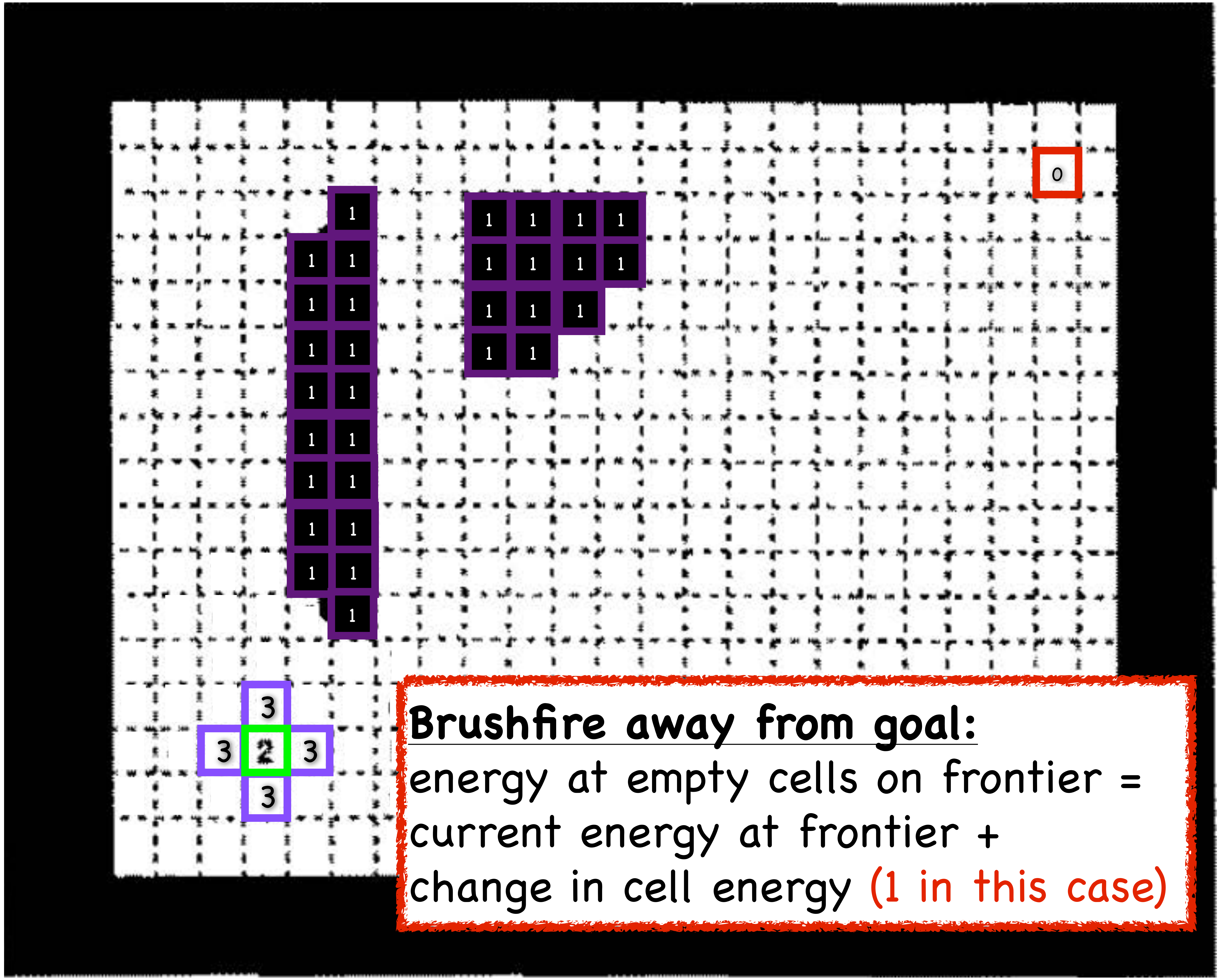
- Discretize potential field into grid
  - Cells store cost to goal with respect to potential field
  - Computed by Brushfire algorithm (essentially BFS)
- Grid search to find navigation path to goal

Slide borrowed from Michigan Robotics autorob.org


Start: mark with 0
0
2
Goal: mark with 2

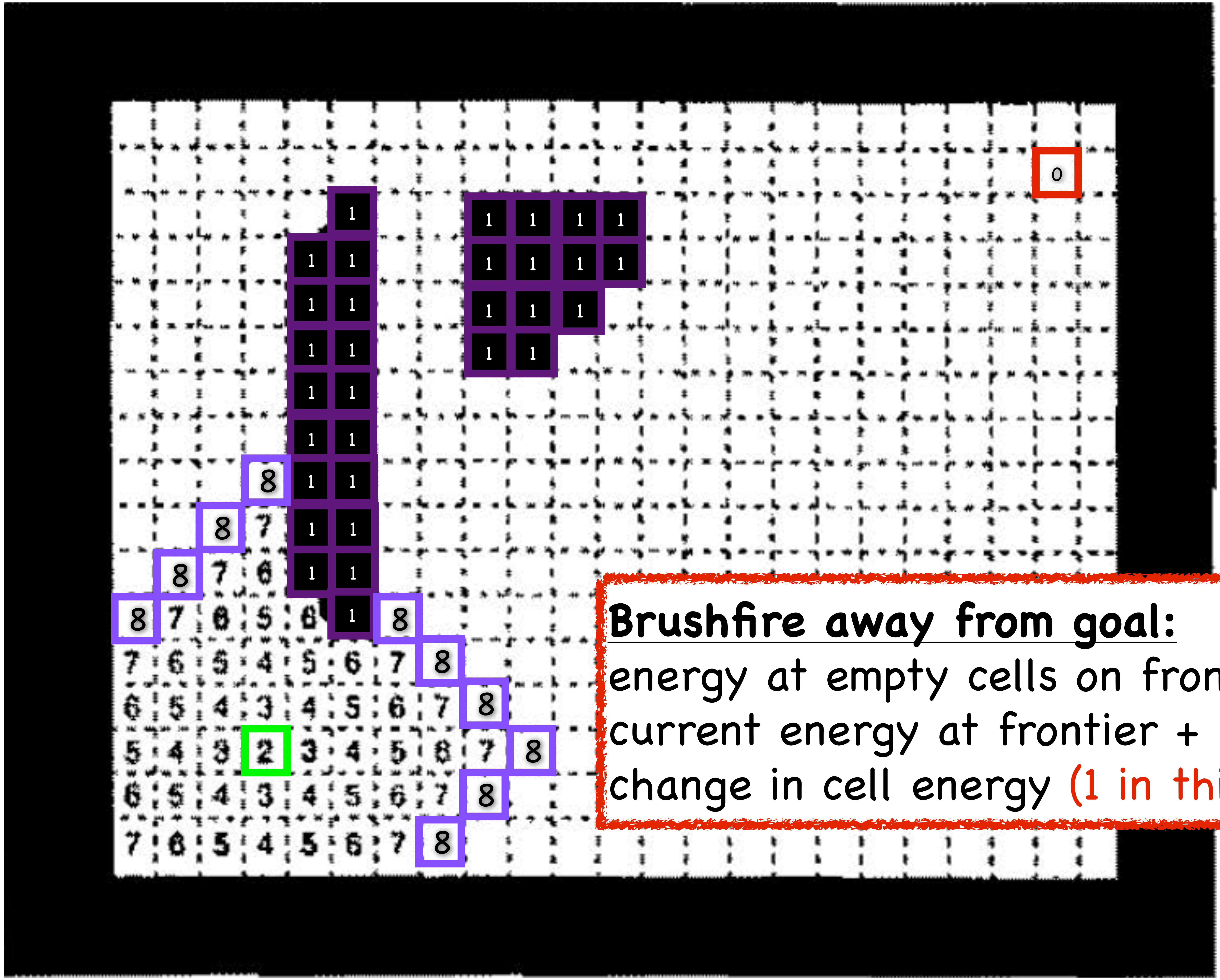
Obstacles: mark with 1



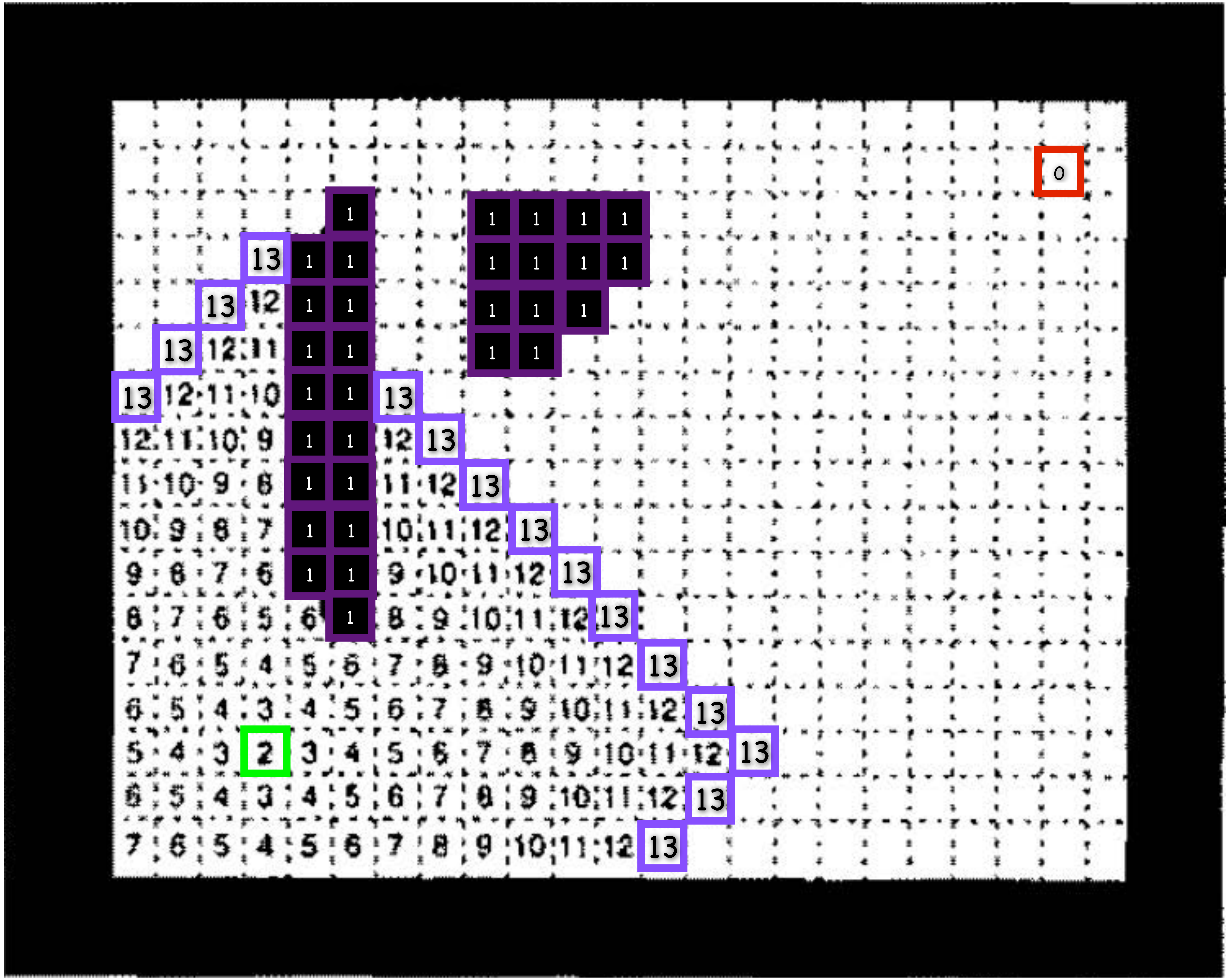


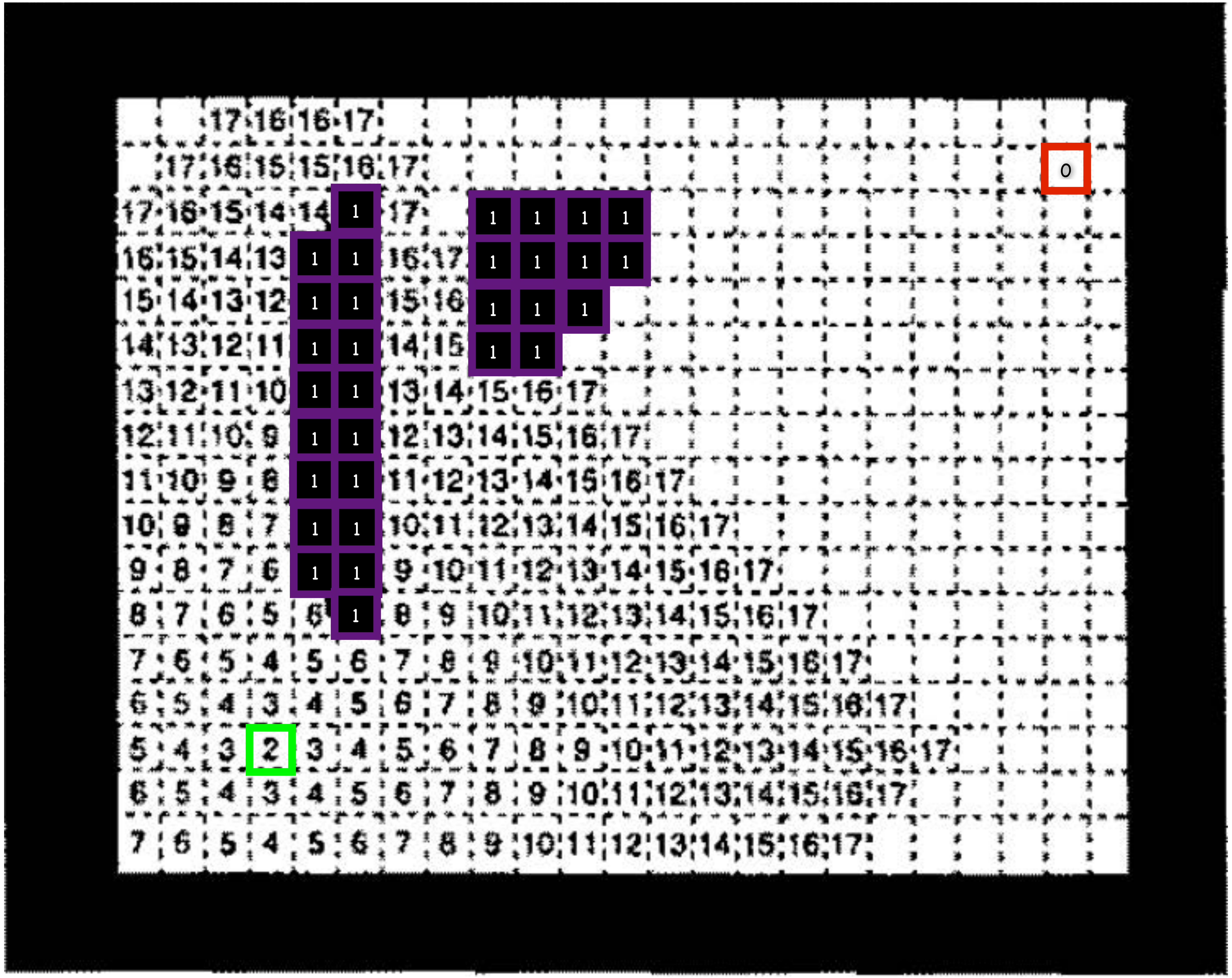
Brushfire away from goal:

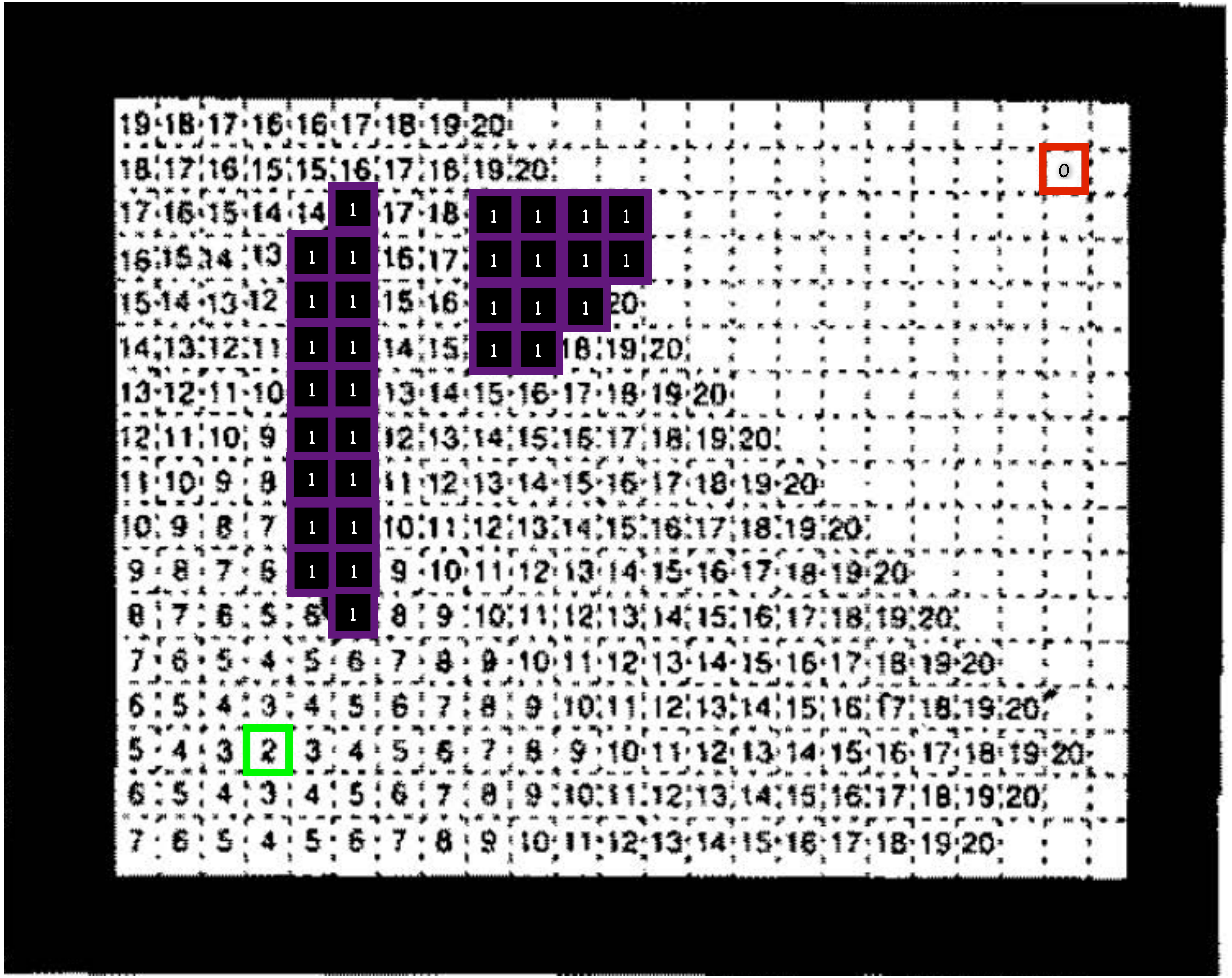
energy at empty cells on frontier =
current energy at frontier +
change in cell energy (1 in this case)

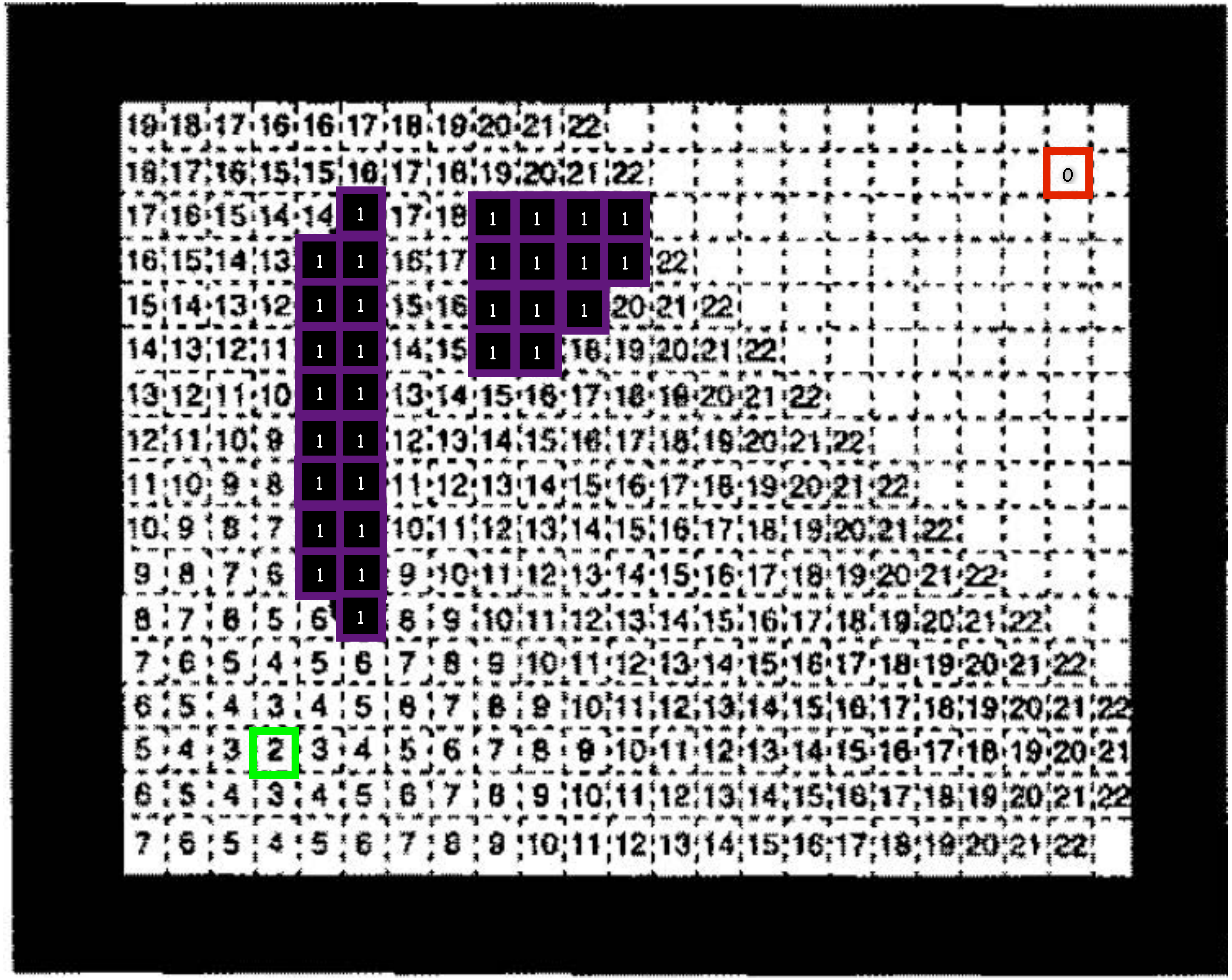


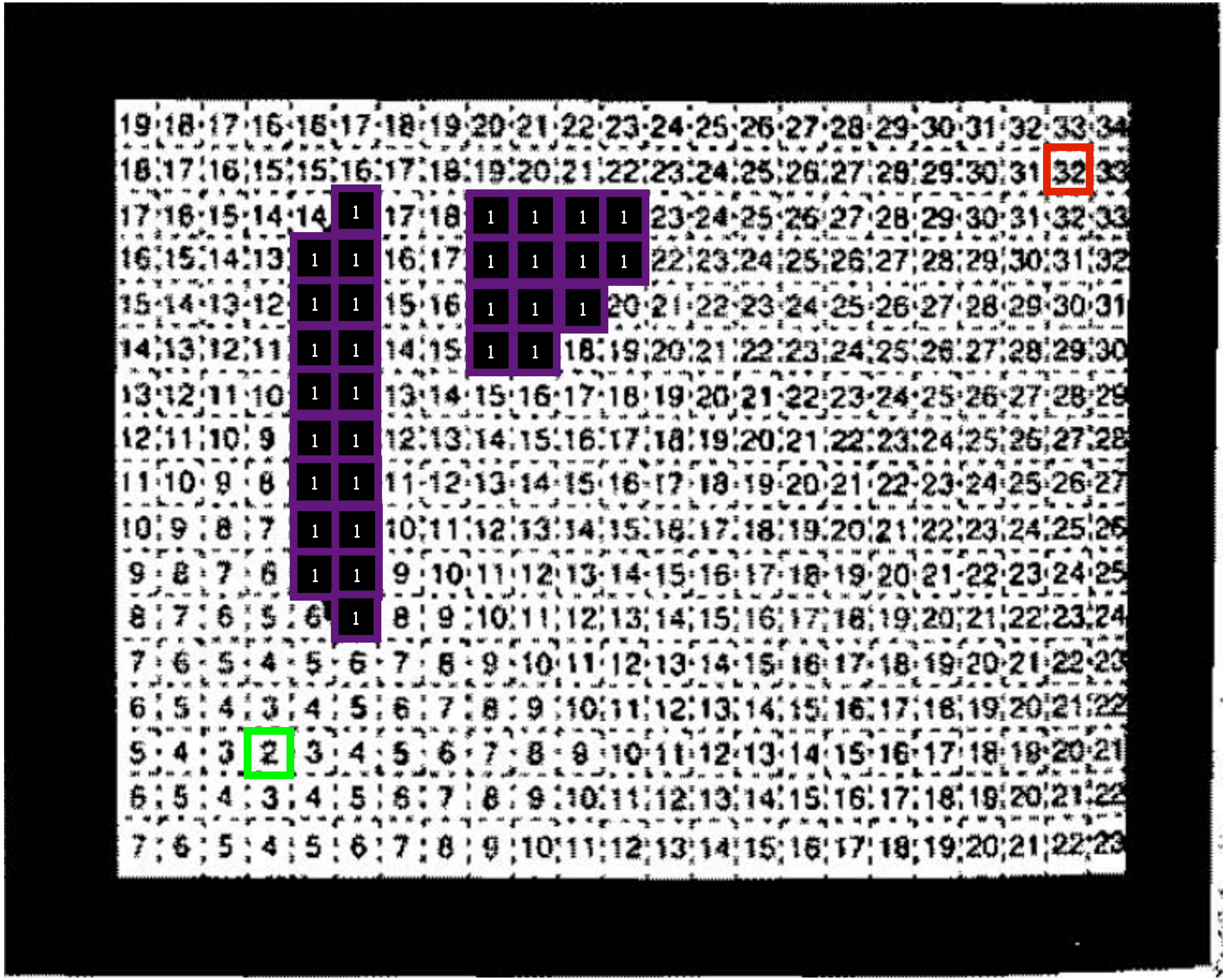
Brushfire away from goal:
energy at empty cells on frontier =
current energy at frontier +
change in cell energy (1 in this case)



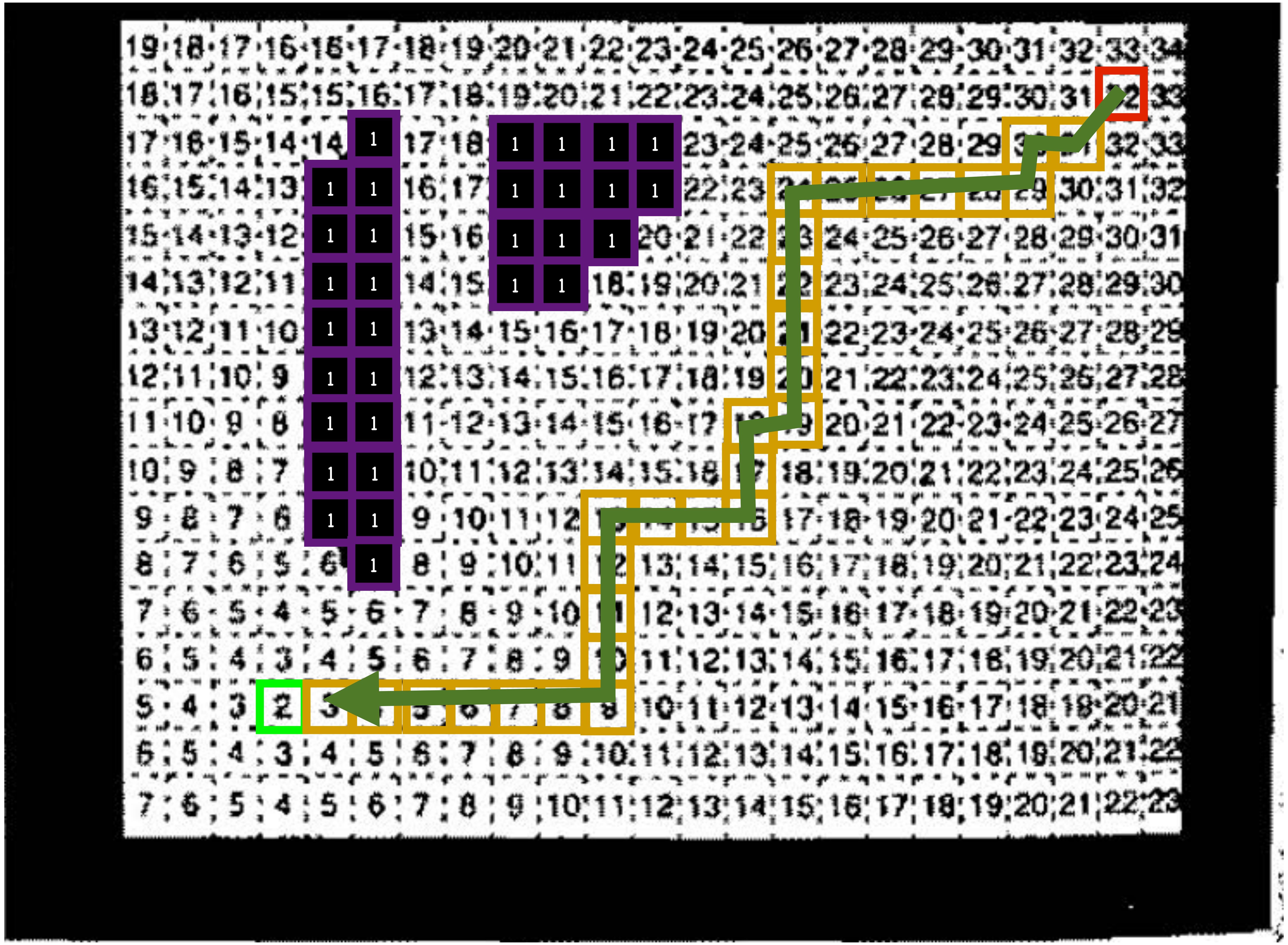





19 18 17 16 16 17 18 19 20 21 22
18 17 16 15 15 16 17 18 19 20 21 22 0
17 16 15 14 14 1 17 18 1 1 1 1
16 15 14 13 1 1 16 17 1 1 1 1 22
15 14 13 12 1 1 15 16 1 1 1 20 21 22
14 13 12 11 1 1 14 15 1 1 18 19 20 21 22
13 12 11 10 1 1 13 14 15 16 17 18 19 20 21 22
12 11 10 9 1 1 12 13 14 15 16 17 18 19 20 21 22
11 10 9 8 1 1 11 12 13 14 15 16 17 18 19 20 21 22
10 9 8 7 1 1 10 11 12 13 14 15 16 17 18 19 20 21 22
9 8 7 6 1 1 9 10 11 12 13 14 15 16 17 18 19 20 21 22
8 7 6 5 6 1 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
5 4 3 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

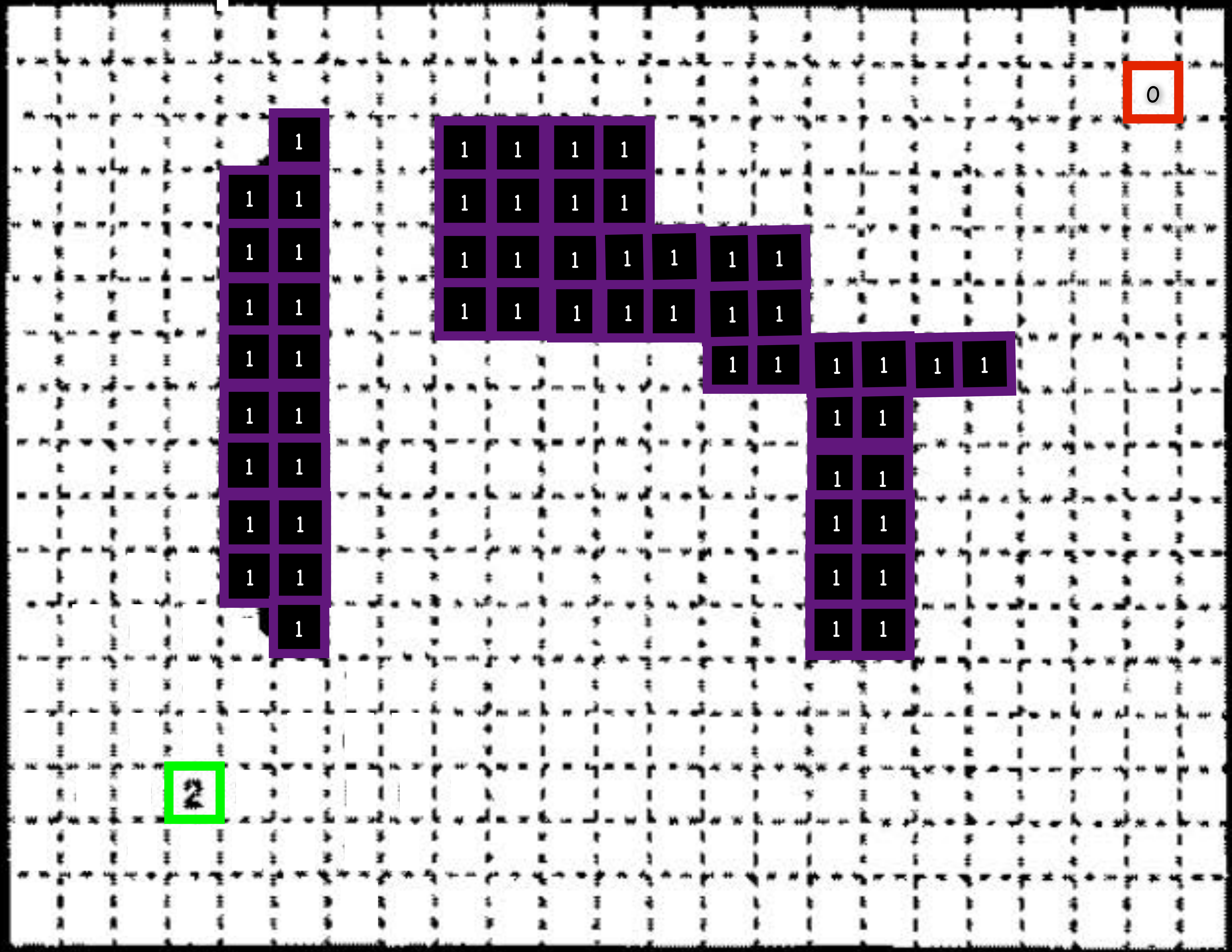


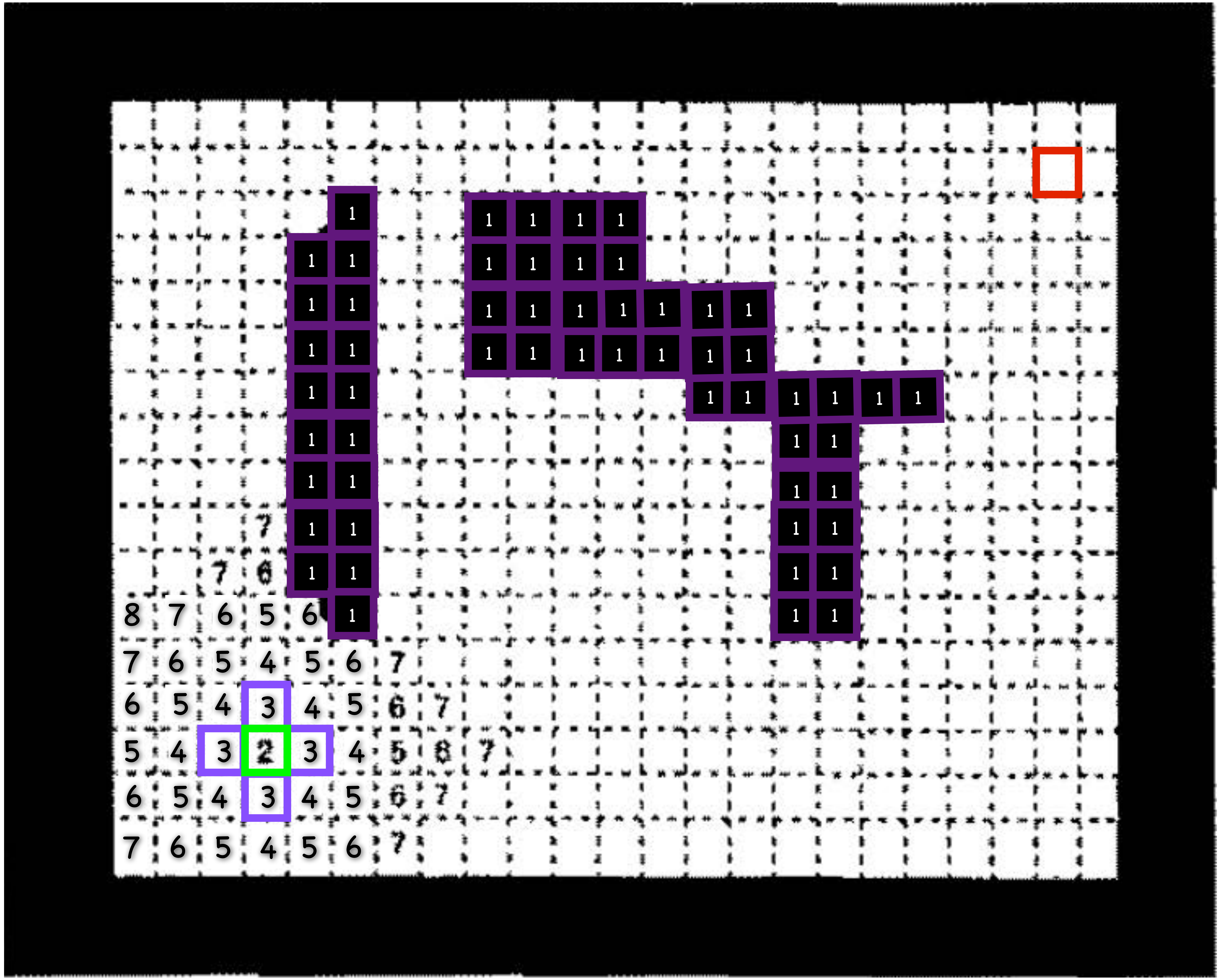
Once start reached,
follow brushfire potential to goal

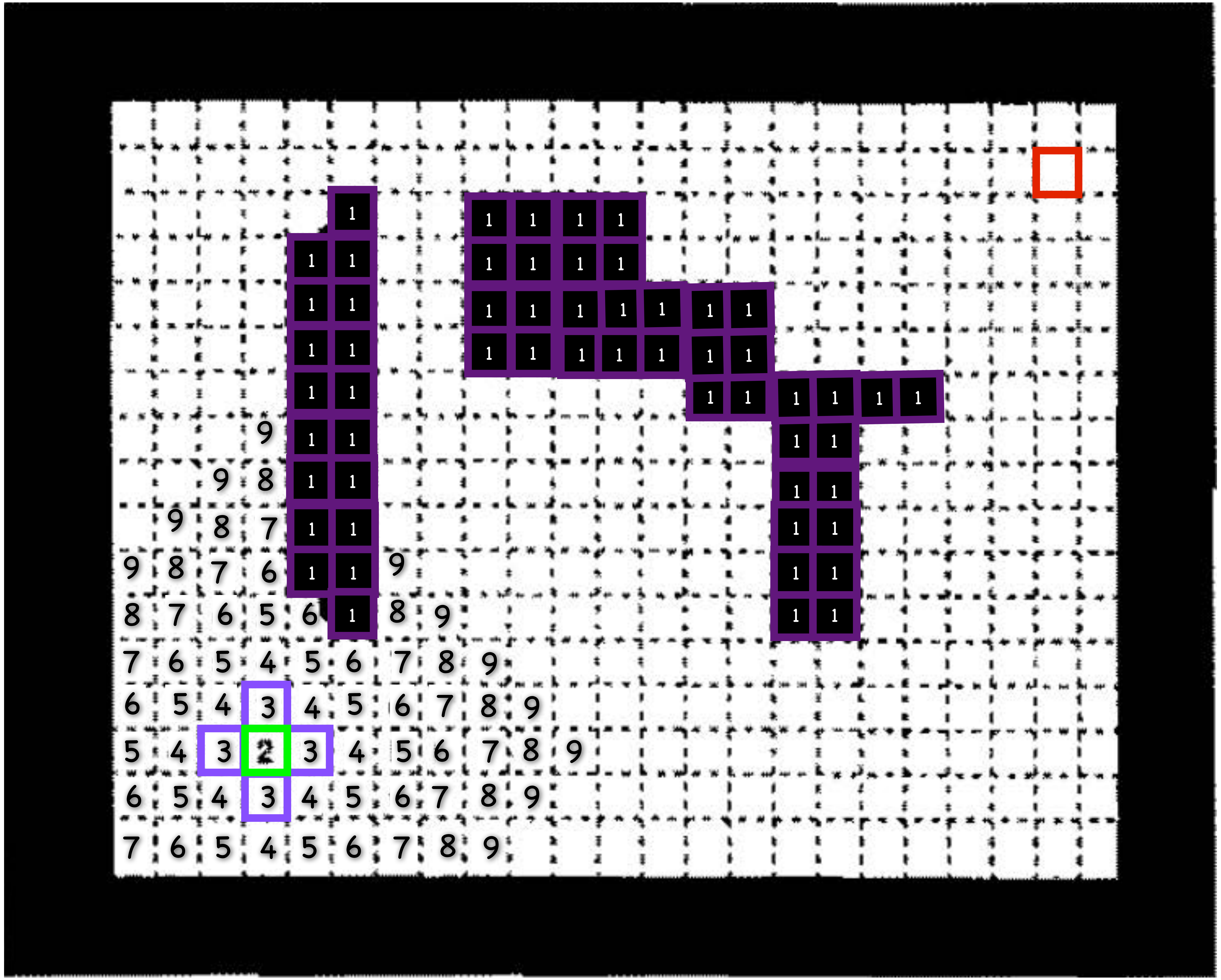


Example with Local Minima

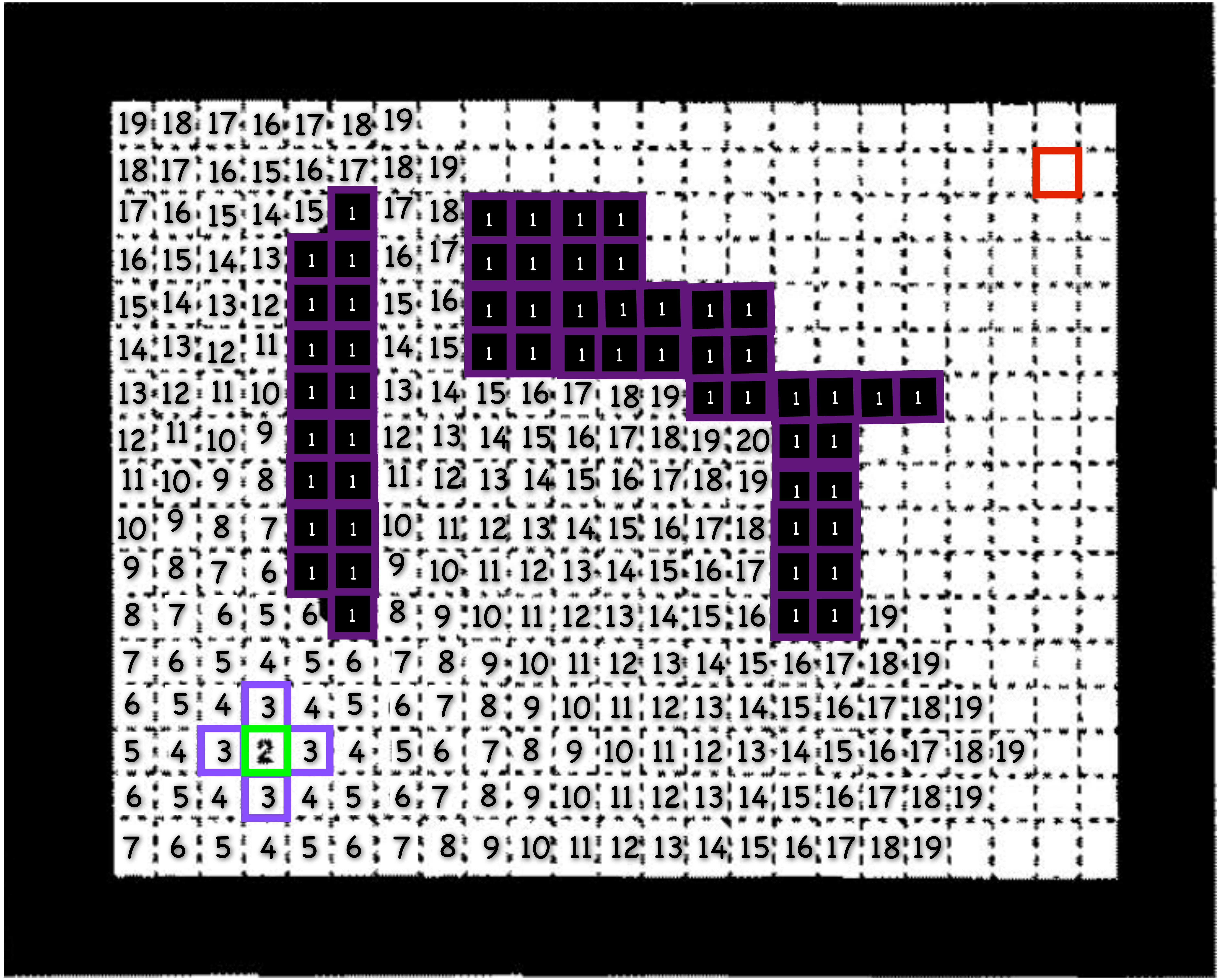
Example with Local Minima



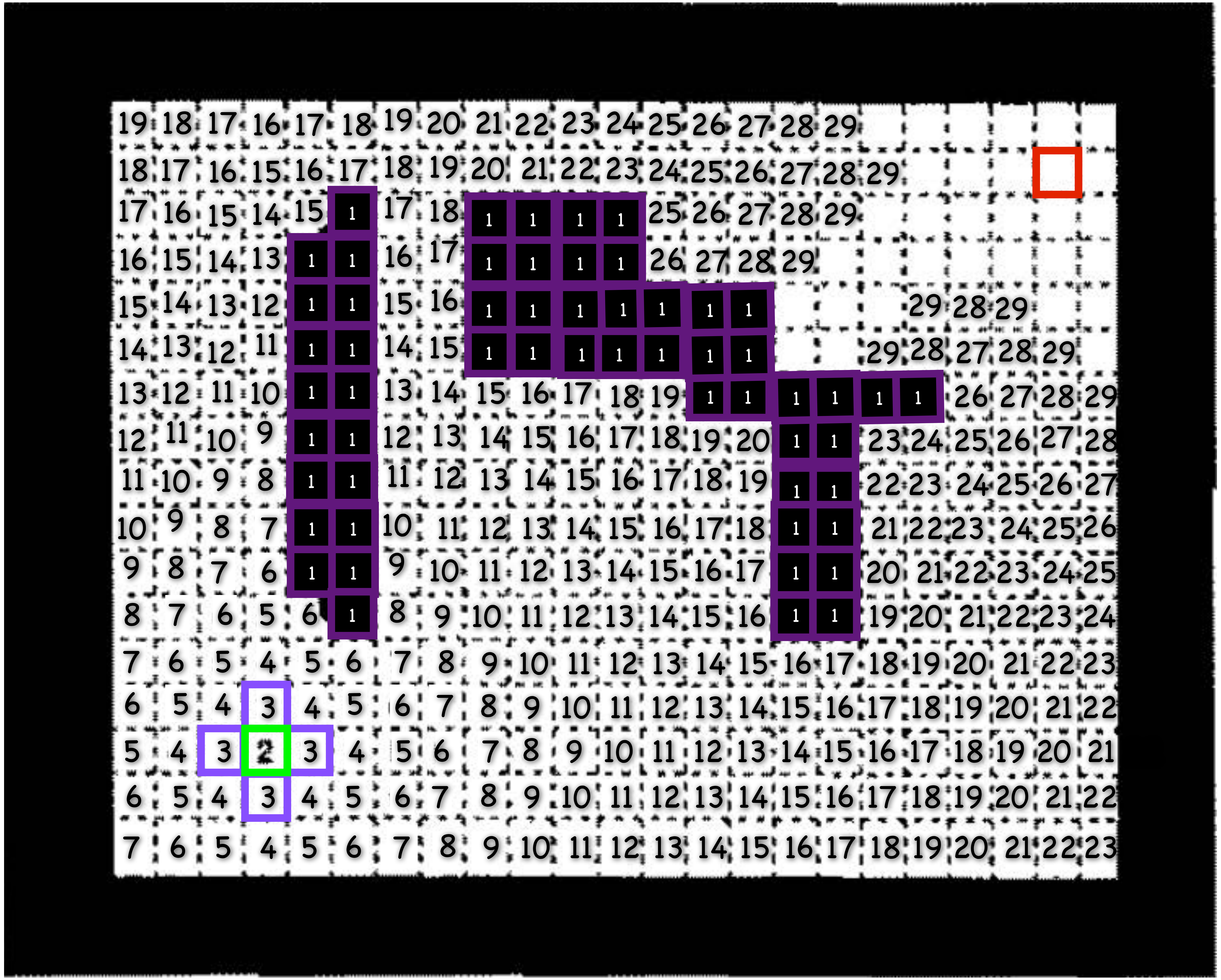


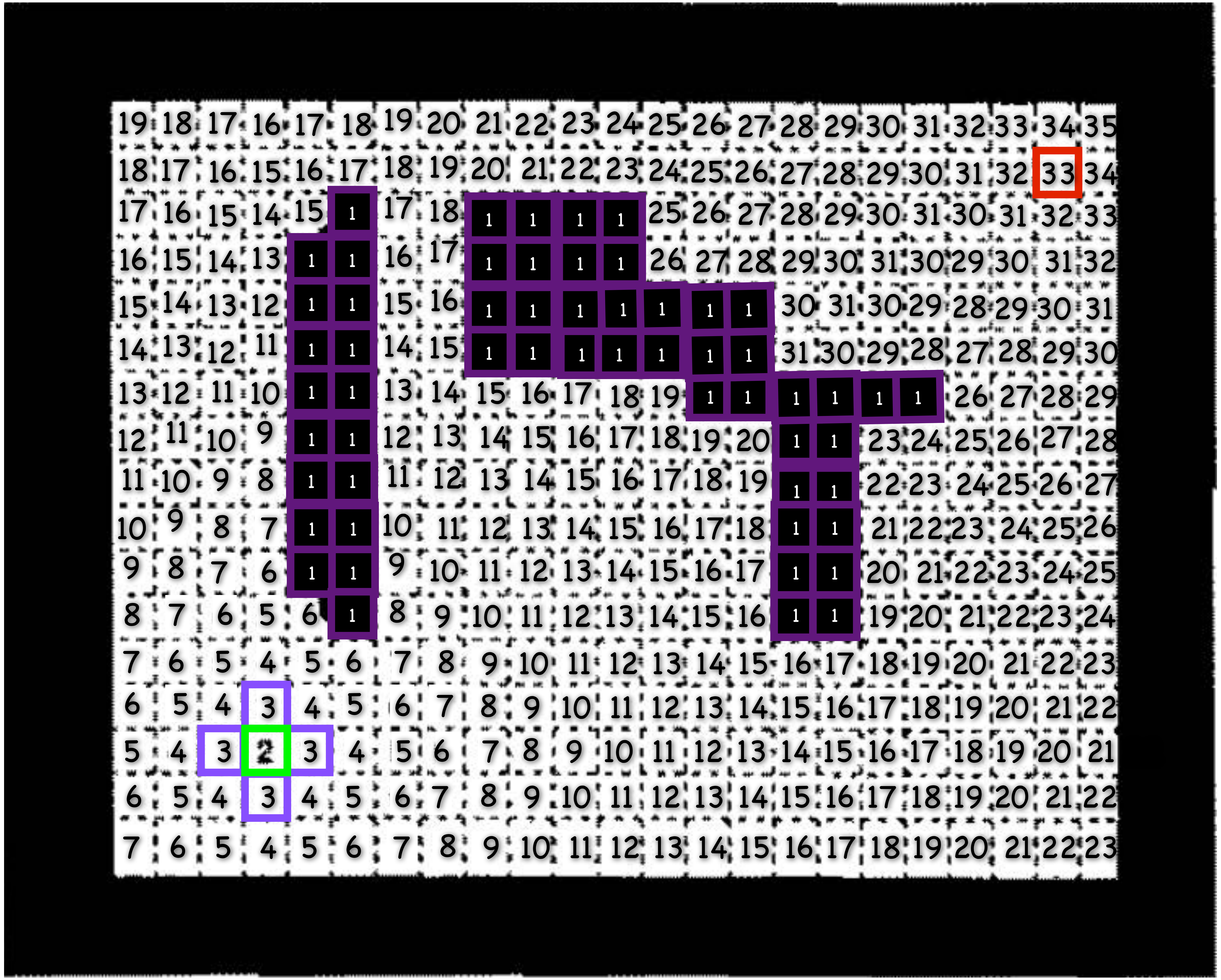


9
9 8
9 8 7
9 8 7 6 9
8 7 6 5 6 8 9
7 6 5 4 5 6 7 8 9
6 5 4 3 4 5 6 7 8 9
5 4 3 2 3 4 5 6 7 8 9
6 5 4 3 4 5 6 7 8 9
7 6 5 4 5 6 7 8 9

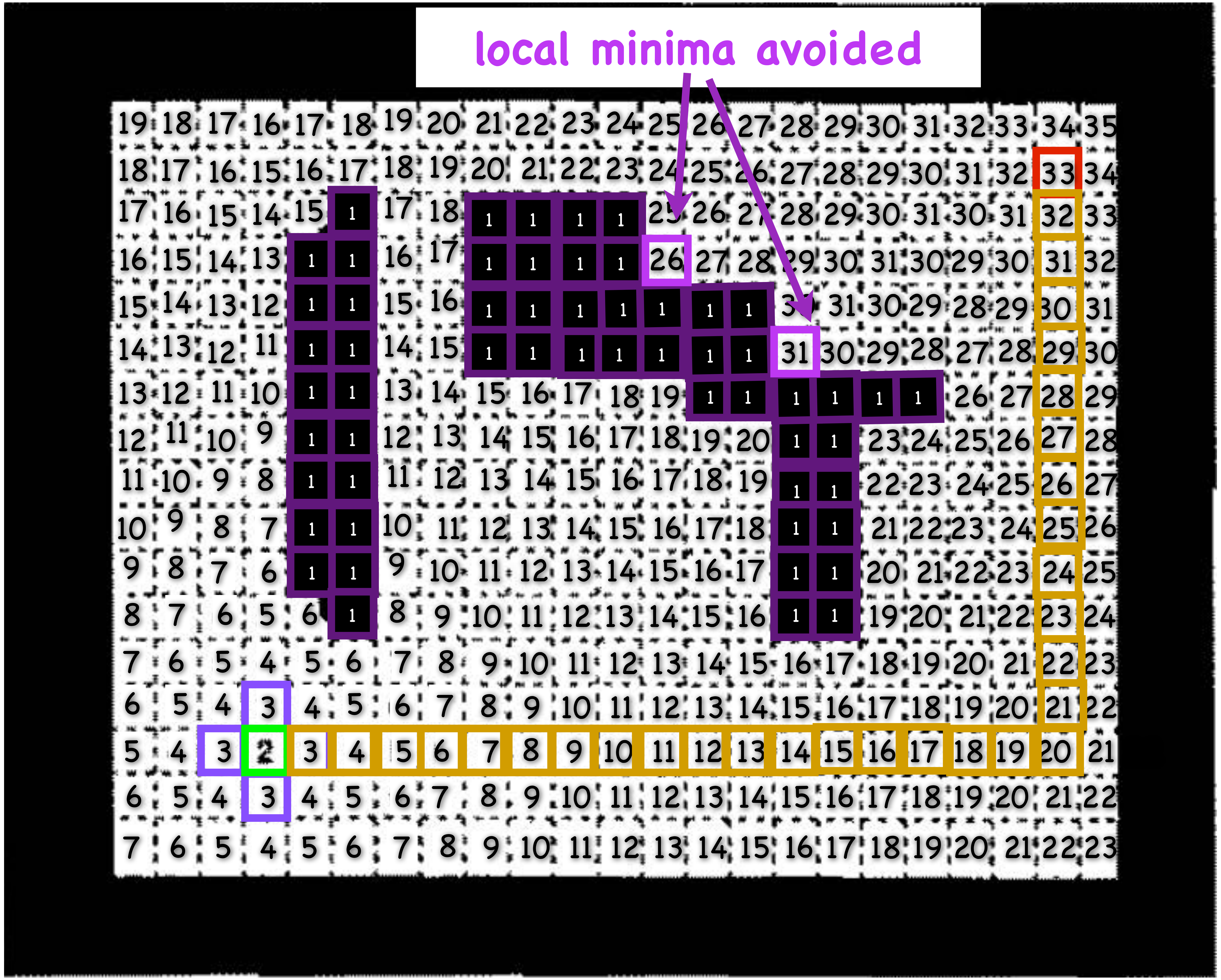


19 18 17 16 17 18 19
18 17 16 15 16 17 18 19
17 16 15 14 15 1 17 18 1 1 1 1
16 15 14 13 1 1 16 17 1 1 1 1
15 14 13 12 1 1 15 16 1 1 1 1 1 1 1
14 13 12 11 1 1 14 15 1 1 1 1 1 1 1
13 12 11 10 1 1 13 14 15 16 17 18 19 1 1 1 1 1 1
12 11 10 9 1 1 12 13 14 15 16 17 18 19 20 1 1
11 10 9 8 1 1 11 12 13 14 15 16 17 18 19 1 1
10 9 8 7 1 1 10 11 12 13 14 15 16 17 18 1 1
9 8 7 6 1 1 9 10 11 12 13 14 15 16 17 1 1
8 7 6 5 6 1 8 9 10 11 12 13 14 15 16 1 1 19
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
5 4 3 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

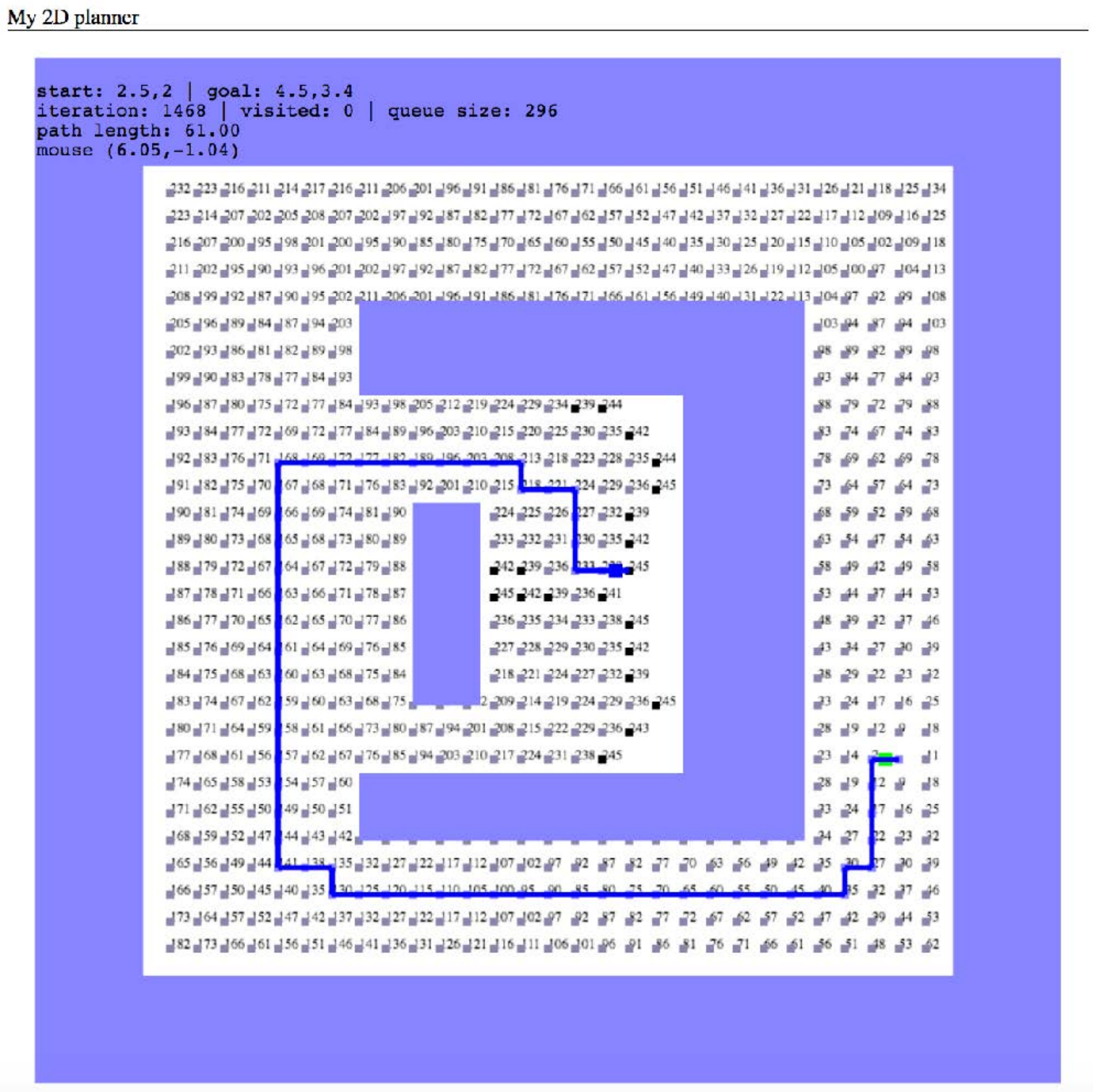

19 18 17 16 17 18 19 20 21 22 23 24 25 26 27 28 29
18 17 16 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
17 16 15 14 15 1 17 18 1 1 1 1 25 26 27 28 29
16 15 14 13 1 1 16 17 1 1 1 1 26 27 28 29
15 14 13 12 1 1 15 16 1 1 1 1 1 1 1 29 28 29
14 13 12 11 1 1 14 15 1 1 1 1 1 1 1 29 28 27 28 29
13 12 11 10 1 1 13 14 15 16 17 18 19 1 1 1 1 1 1 26 27 28 29
12 11 10 9 1 1 12 13 14 15 16 17 18 19 20 1 1 23 24 25 26 27 28
11 10 9 8 1 1 11 12 13 14 15 16 17 18 19 1 1 22 23 24 25 26 27
10 9 8 7 1 1 10 11 12 13 14 15 16 17 18 1 1 21 22 23 24 25 26
9 8 7 6 1 1 9 10 11 12 13 14 15 16 17 1 1 20 21 22 23 24 25
8 7 6 5 6 1 8 9 10 11 12 13 14 15 16 1 1 19 20 21 22 23 24
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
5 4 3 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23



19 18 17 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
18 17 16 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
17 16 15 14 15 1 17 18 1 1 1 1 25 26 27 28 29 30 31 30 31 32 33
16 15 14 13 1 1 16 17 1 1 1 1 26 27 28 29 30 31 30 29 30 31 32
15 14 13 12 1 1 15 16 1 1 1 1 1 1 1 30 31 30 29 28 29 30 31
14 13 12 11 1 1 14 15 1 1 1 1 1 1 1 31 30 29 28 27 28 29 30
13 12 11 10 1 1 13 14 15 16 17 18 19 1 1 1 1 1 1 26 27 28 29
12 11 10 9 1 1 12 13 14 15 16 17 18 19 20 1 1 23 24 25 26 27 28
11 10 9 8 1 1 11 12 13 14 15 16 17 18 19 1 1 22 23 24 25 26 27
10 9 8 7 1 1 10 11 12 13 14 15 16 17 18 1 1 21 22 23 24 25 26
9 8 7 6 1 1 9 10 11 12 13 14 15 16 17 1 1 20 21 22 23 24 25
8 7 6 5 6 1 8 9 10 11 12 13 14 15 16 1 1 19 20 21 22 23 24
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
5 4 3 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
6 5 4 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
7 6 5 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23


local minima avoided

Kineval wavefront planner



Slide borrowed from Michigan Robotics autorob.org

Planning Recap

Recap

Bug X
- Complete
- Non-optimal
- Planar

Subsumption and FSMs
- Fast but not adaptive
- Emphasis on good design

Potential Fields
- Complete in special cases
- Non-optimal
- General C-spaces
- Scales w/dimensionality

Grid Search/Wavefront
- Complete
- General C-spaces
- Limited dimensionality

Random walk
- Will find path eventually

Sampling roadmaps/RRT
- Probabilistically complete
- General
- Tractable (with good sampling)
- Scales w/dimensionality
- Not necessarily optimal

Next Lecture
Motion Control