Potential fields for navigation

EECS 367
Intro. to Autonomous Robotics

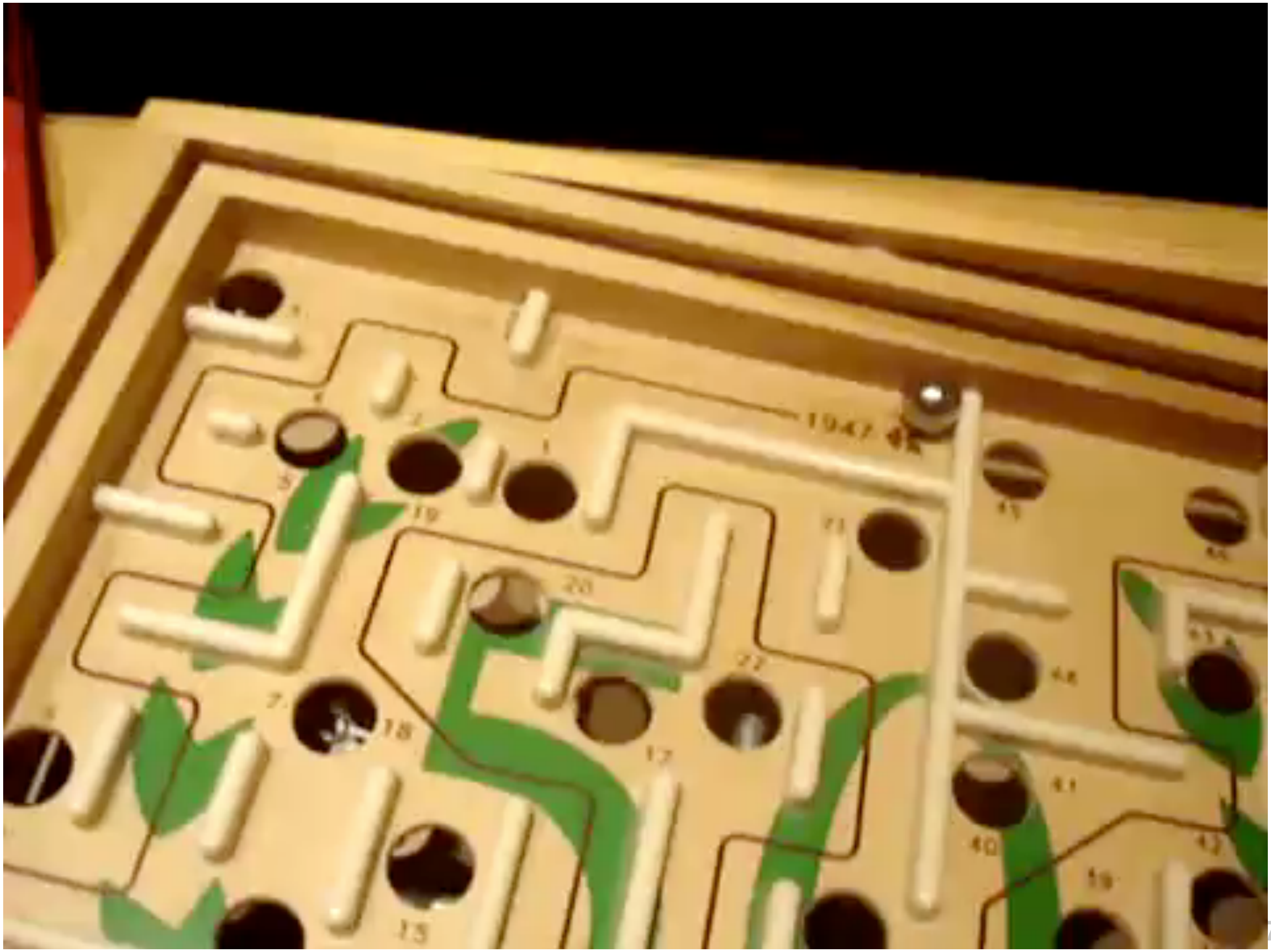
ME/EECS 567 ROB 510
Robot Modeling and Control

Fall 2019

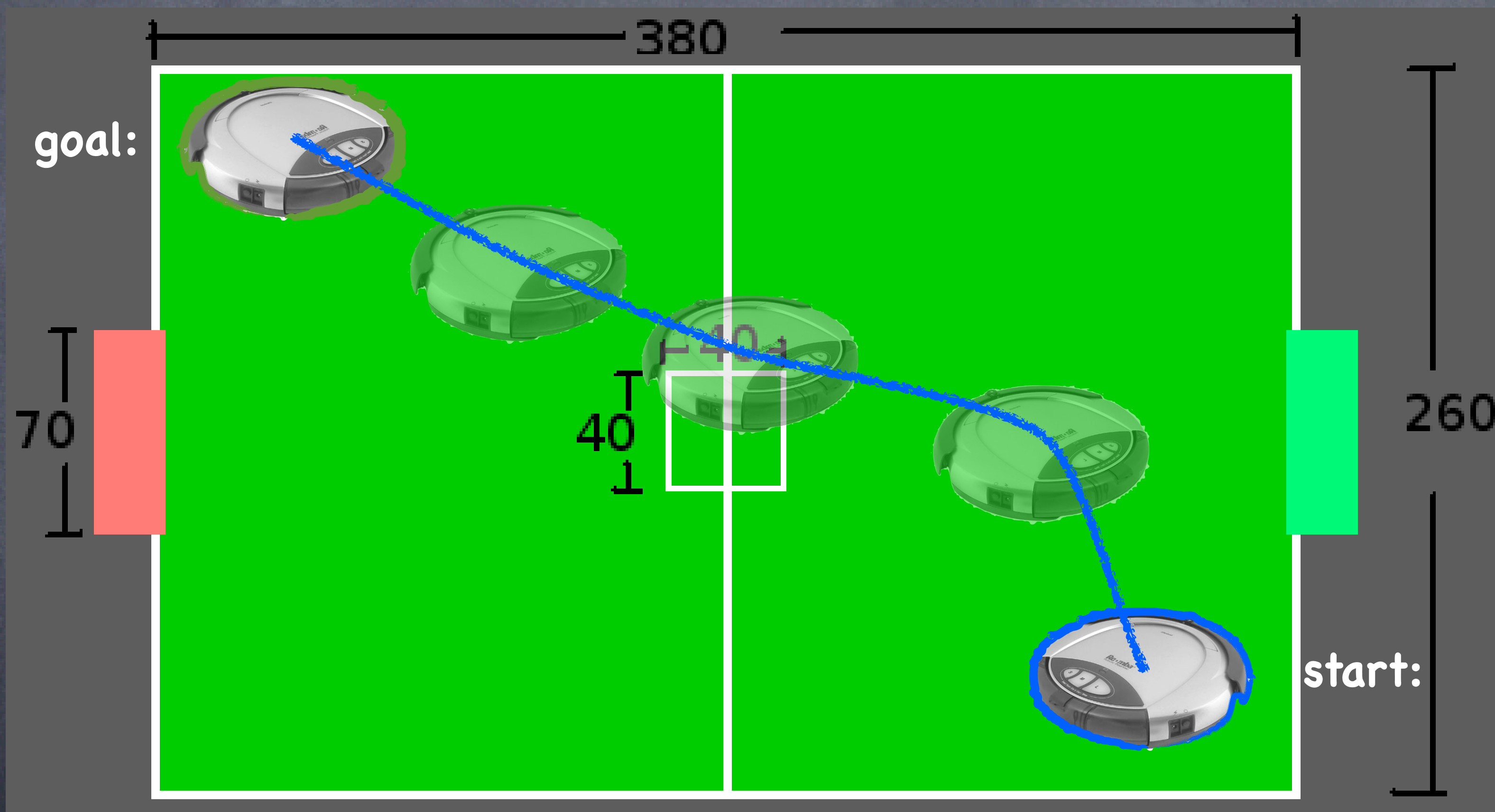
autorob.org

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**

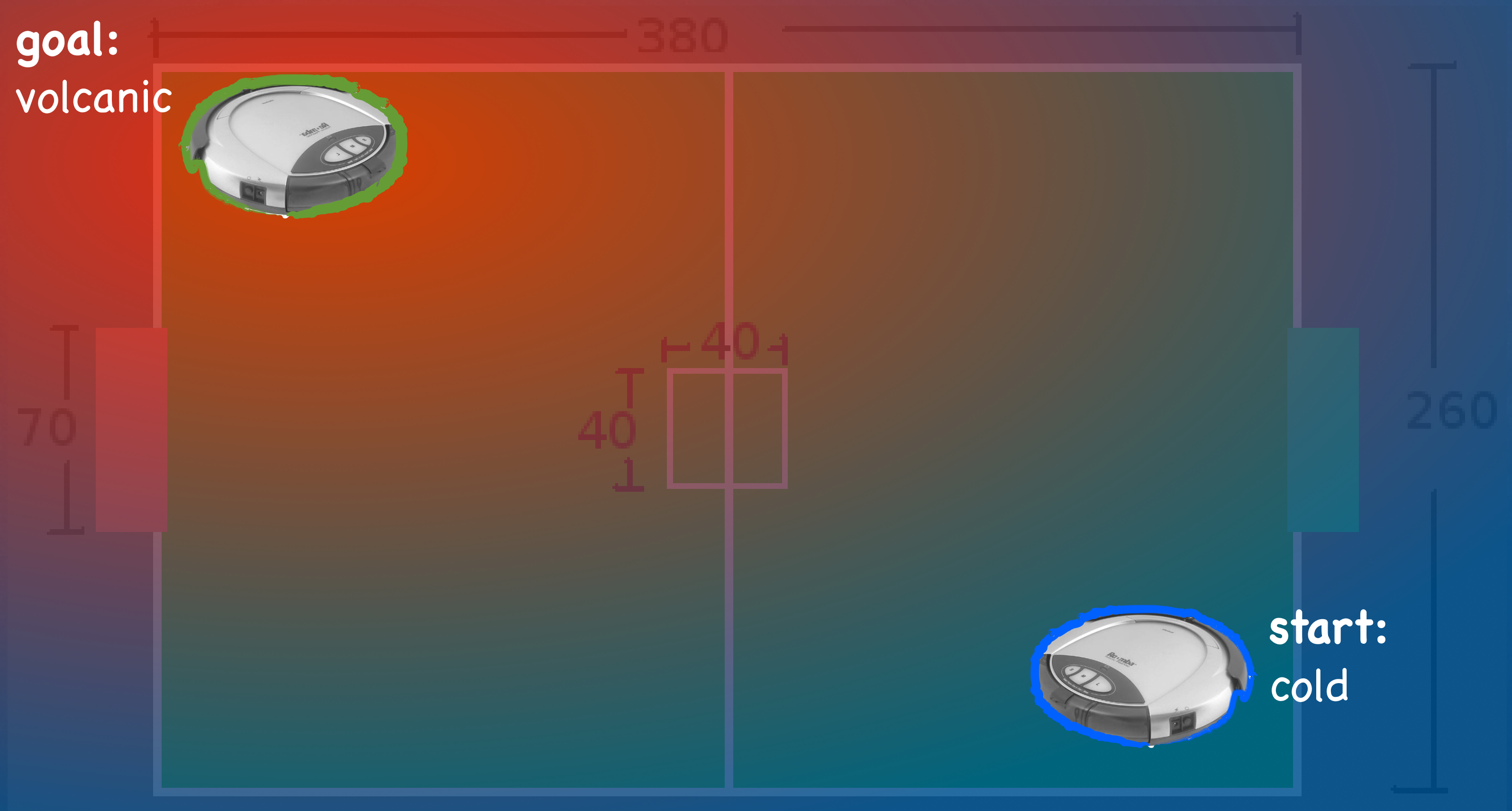

1947

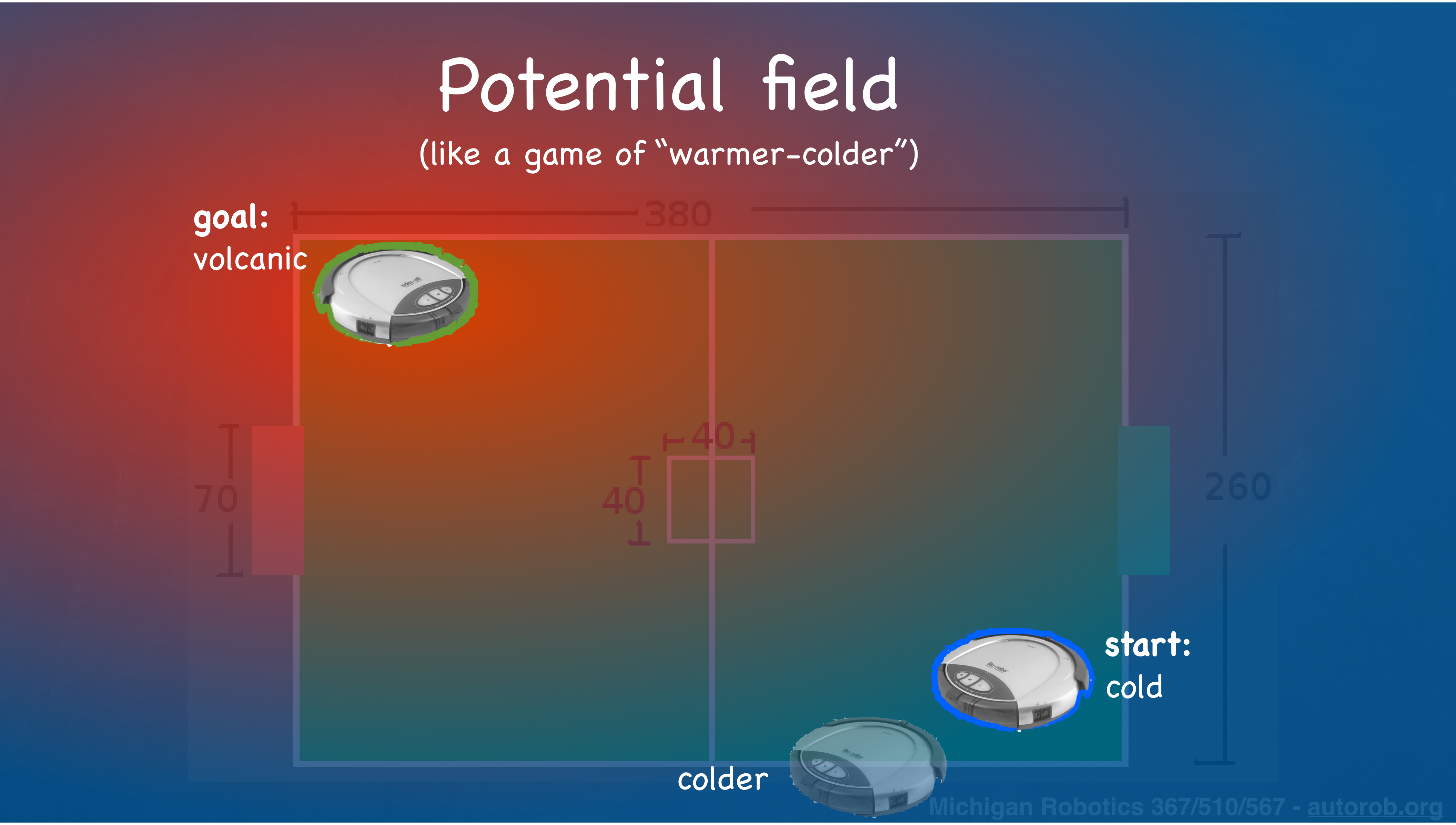
Navigation (again)



Potential field

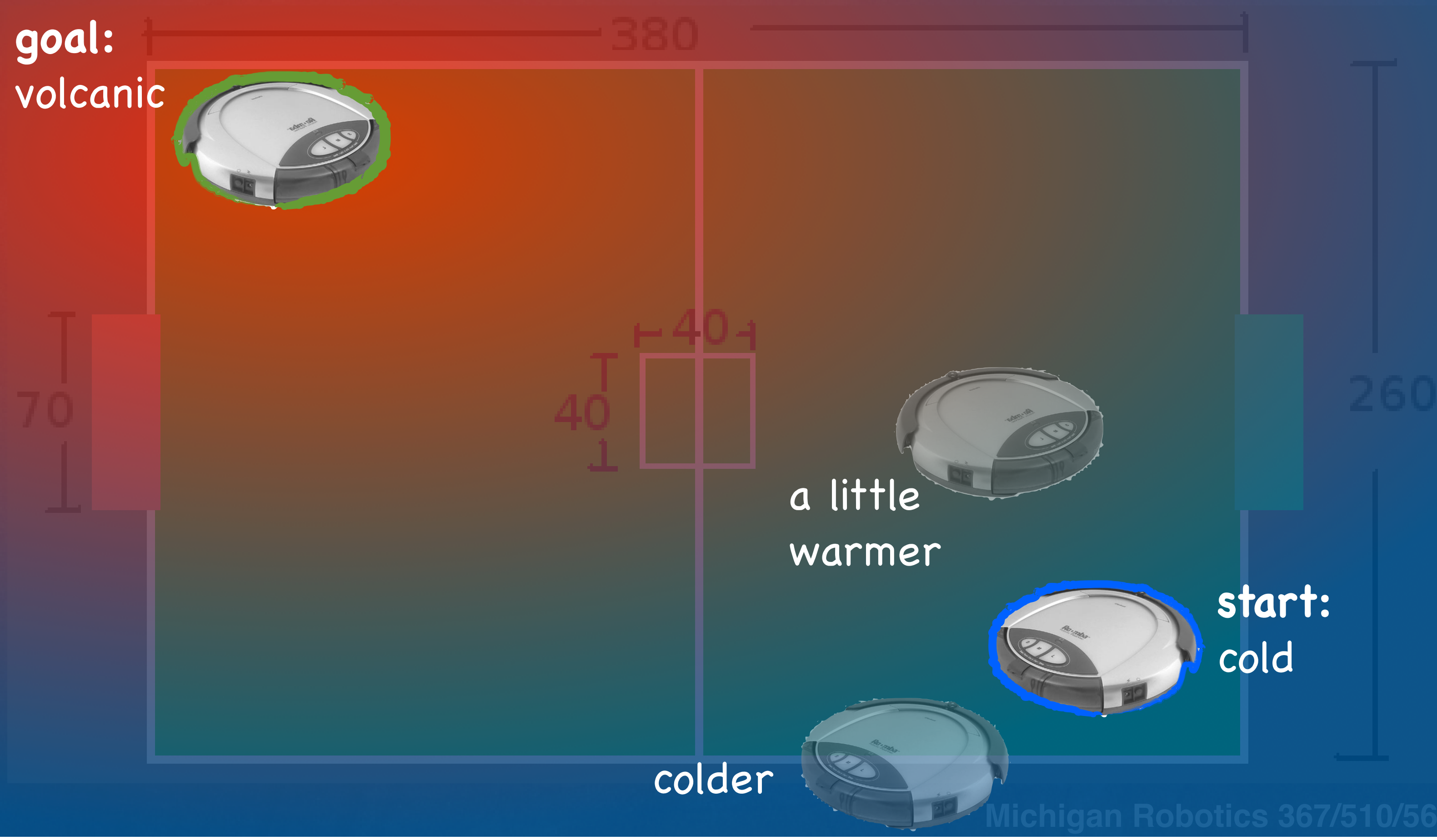
(like a game of "warmer-colder")




Potential field
(like a game of "warmer-colder")
goal:
volcanic
380
70
40
40
260
start:
cold
colder
Michigan Robotics 367/510/567 - autorob.org

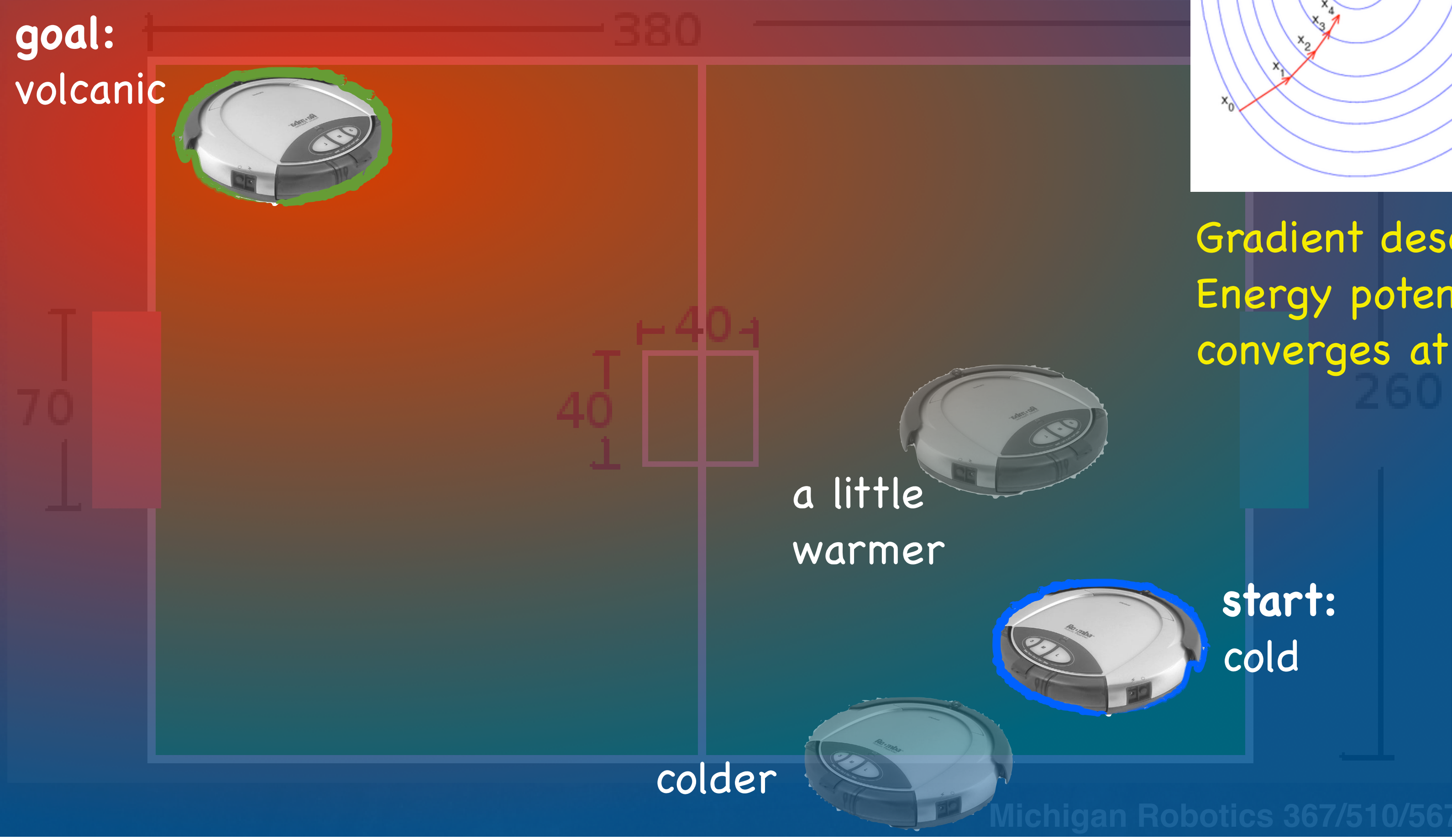
Potential field

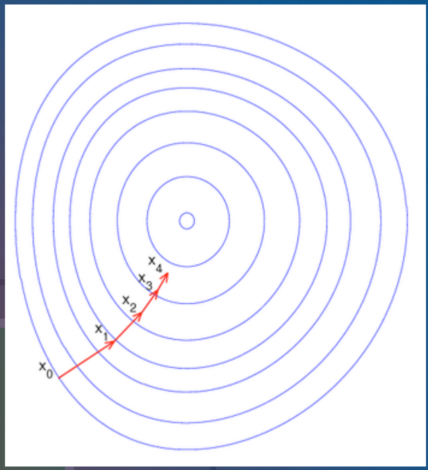
(like a game of “warmer-colder”)



Potential field

(like a game of "warmer-colder")

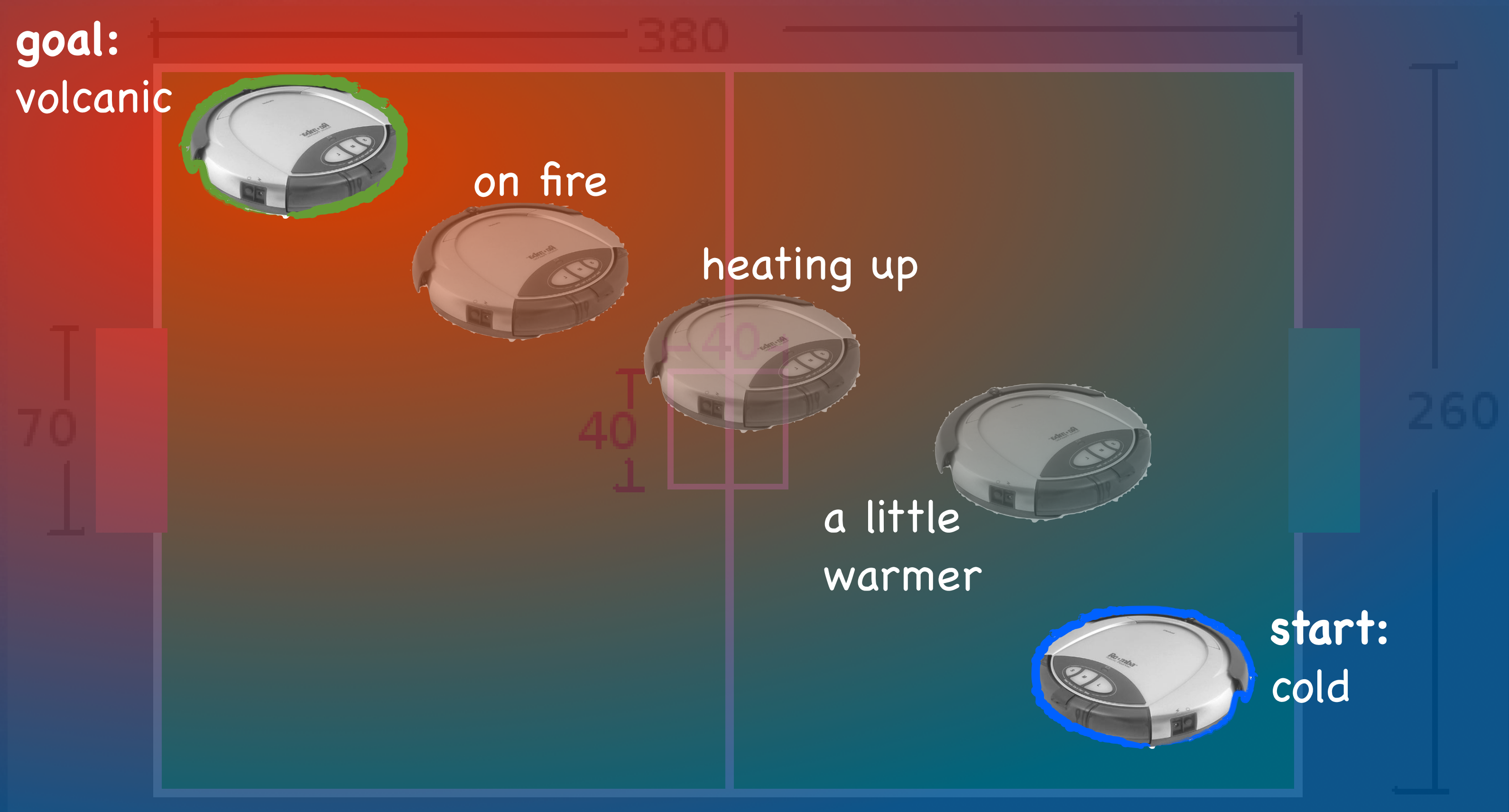




Gradient descent:
Energy potential
converges at goal

Potential field

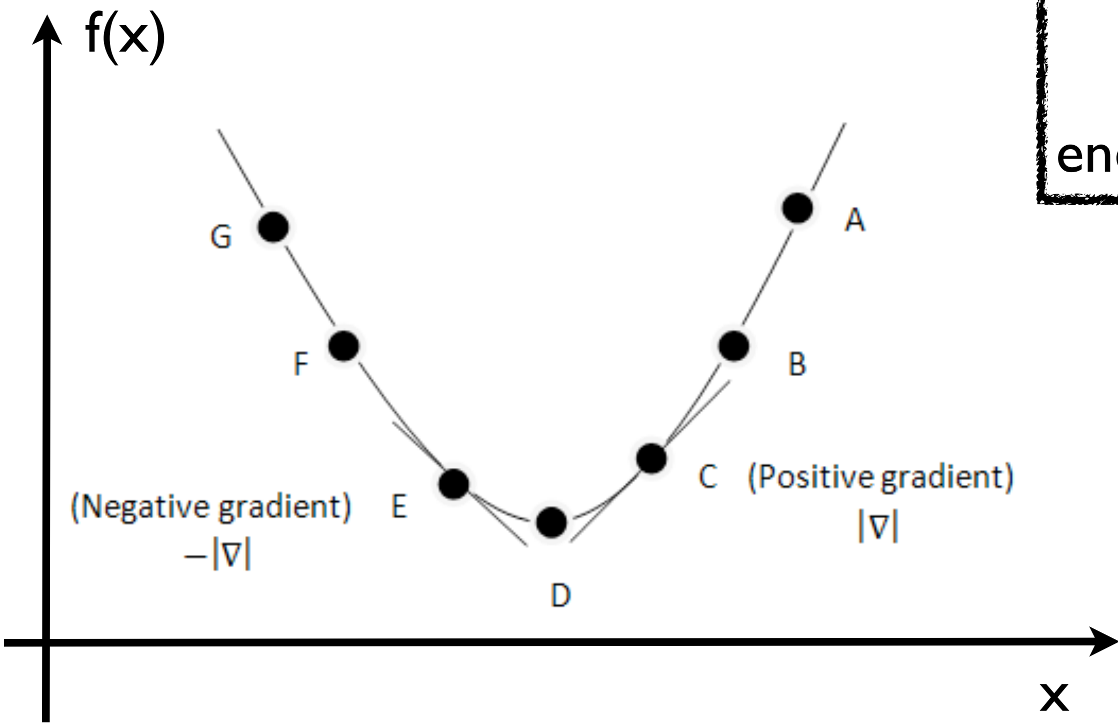
(like a game of "warmer-colder")



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

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

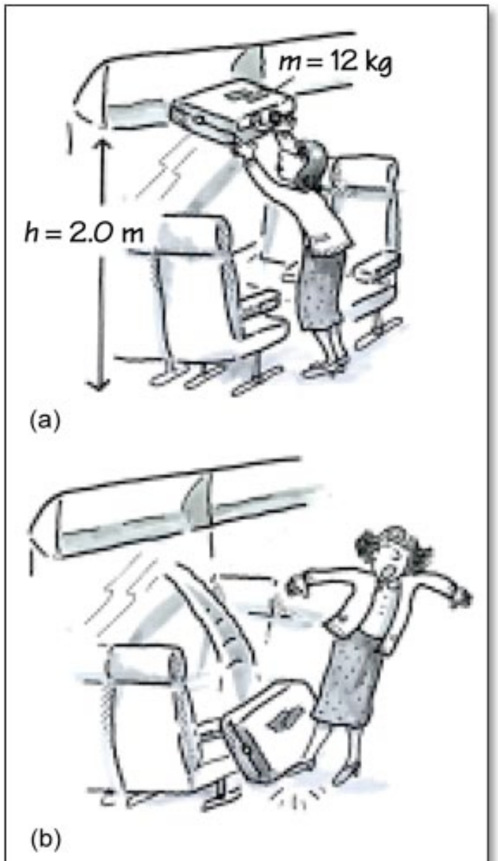
  $i \leftarrow i+1$

end

Derivative assumed to be direction of steepest ascent away from goal

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

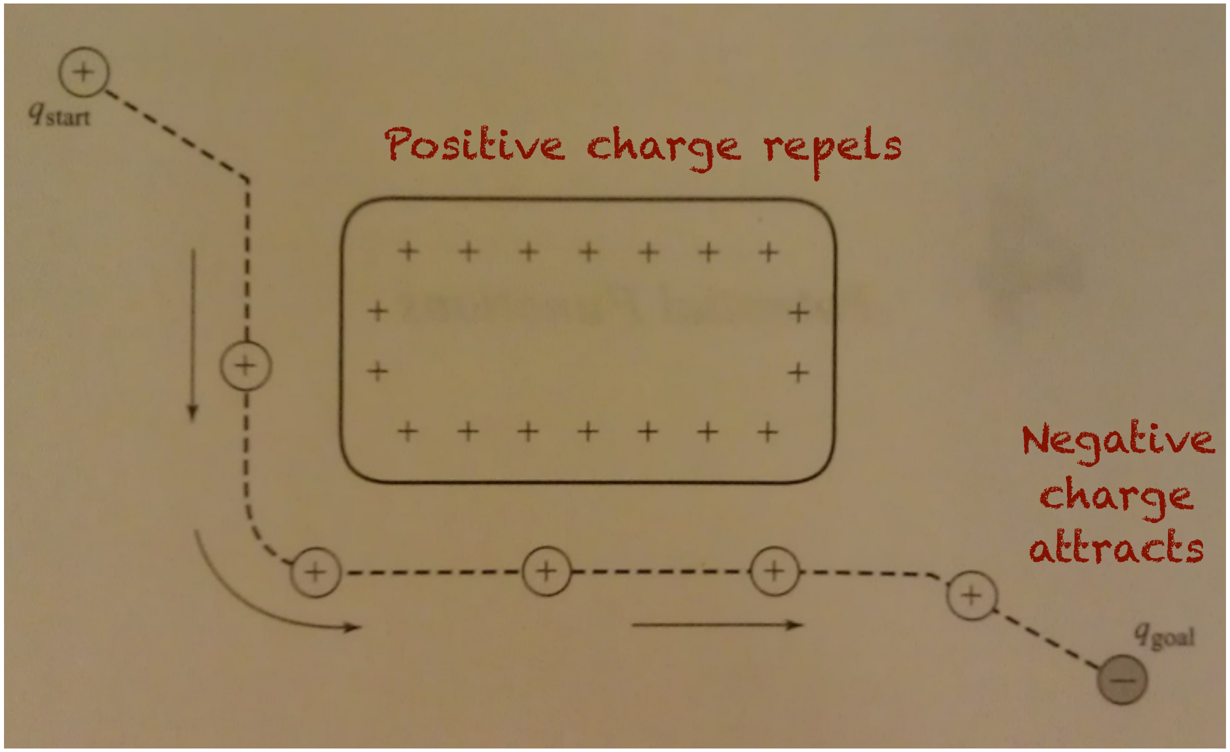
Potential Energy



- 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

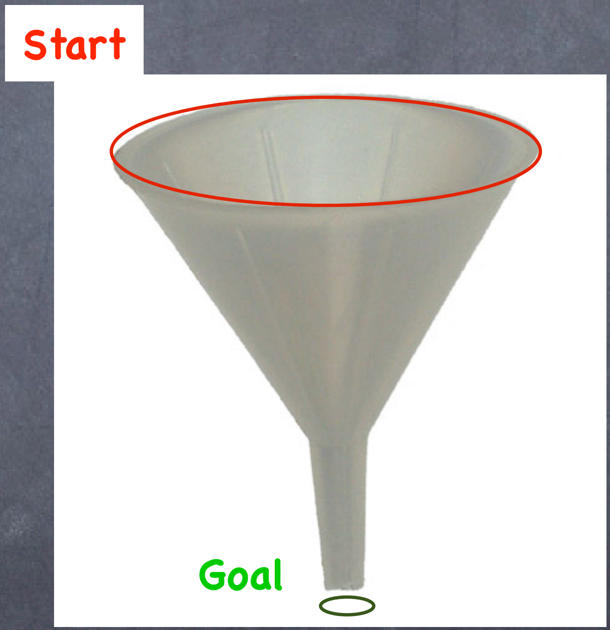
Charged Particle Example

Positively charged particle follows potential energy to goal



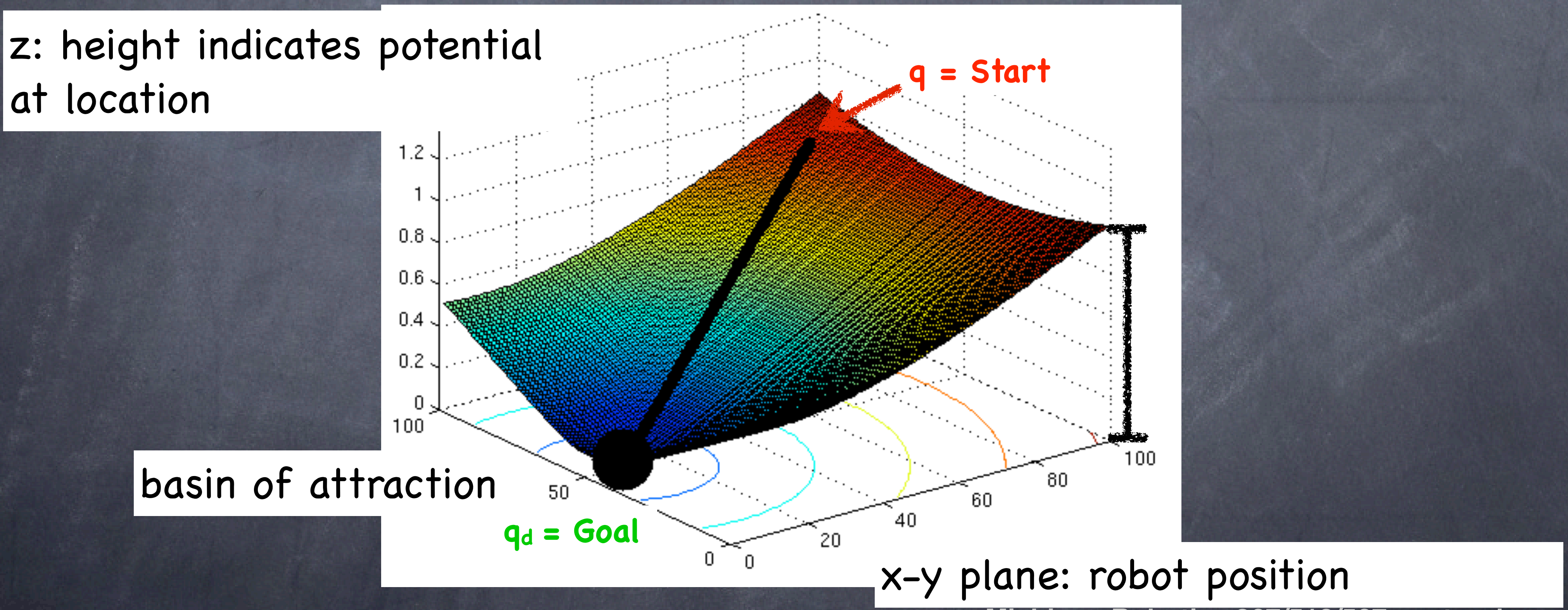
Convergent Potentials

let's call these "attractor landscapes"

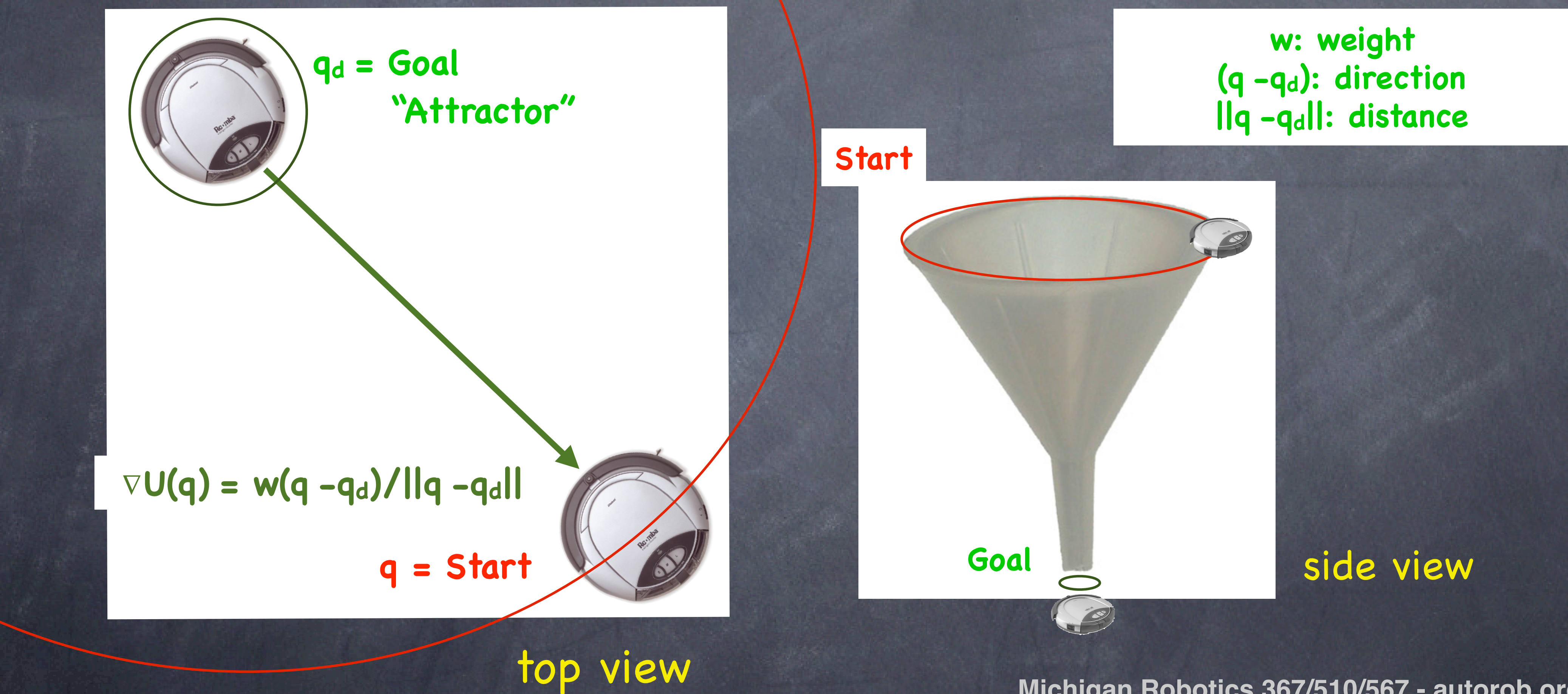


basin of attraction

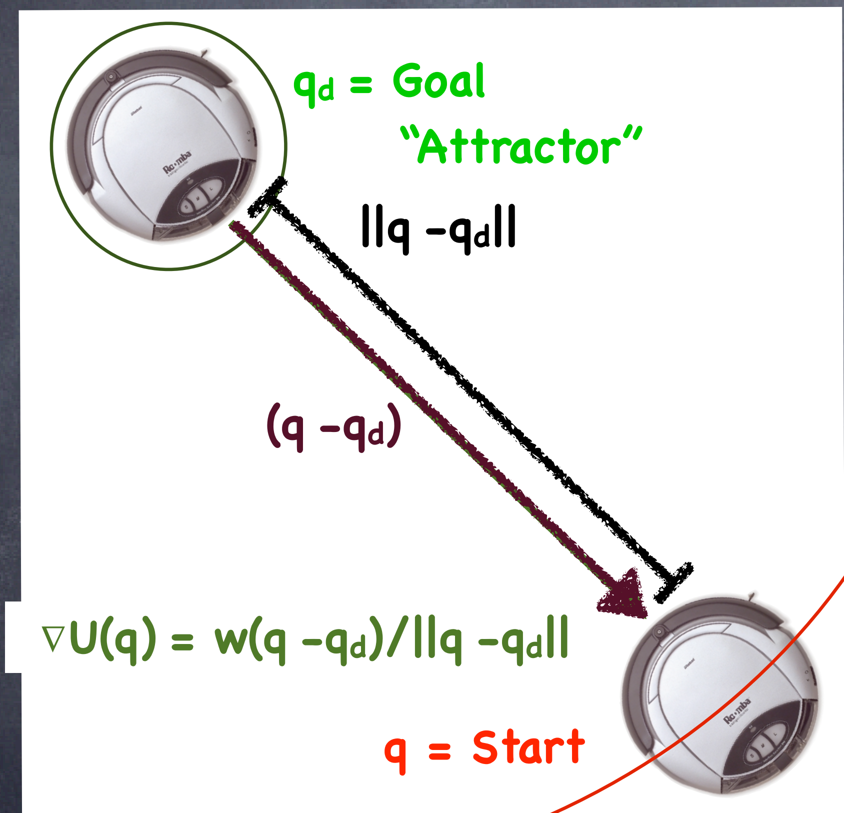
2D potential navigation



"Cone" Attractor



"Cone" Attractor

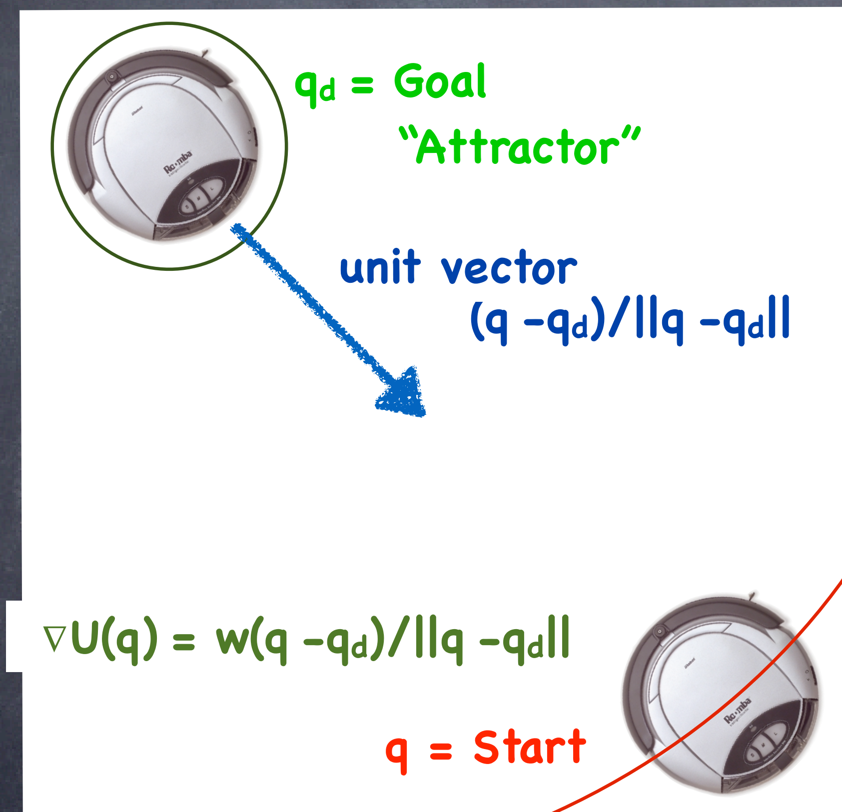


top view

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

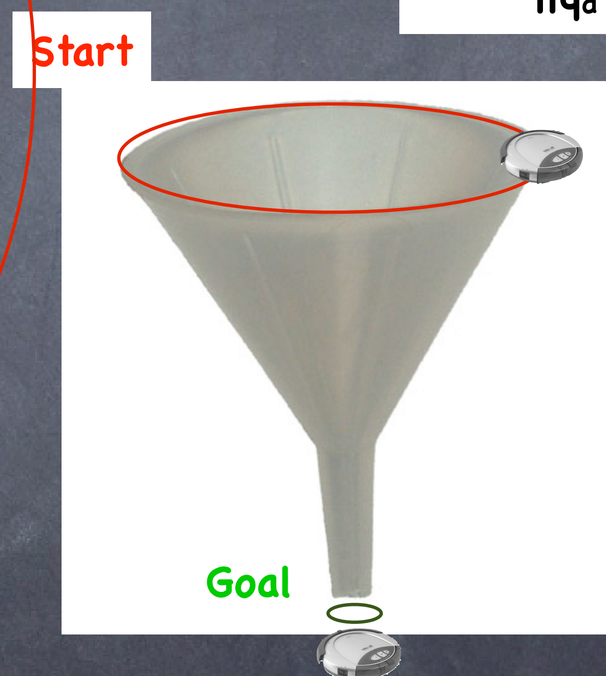
side view

"Cone" Attractor



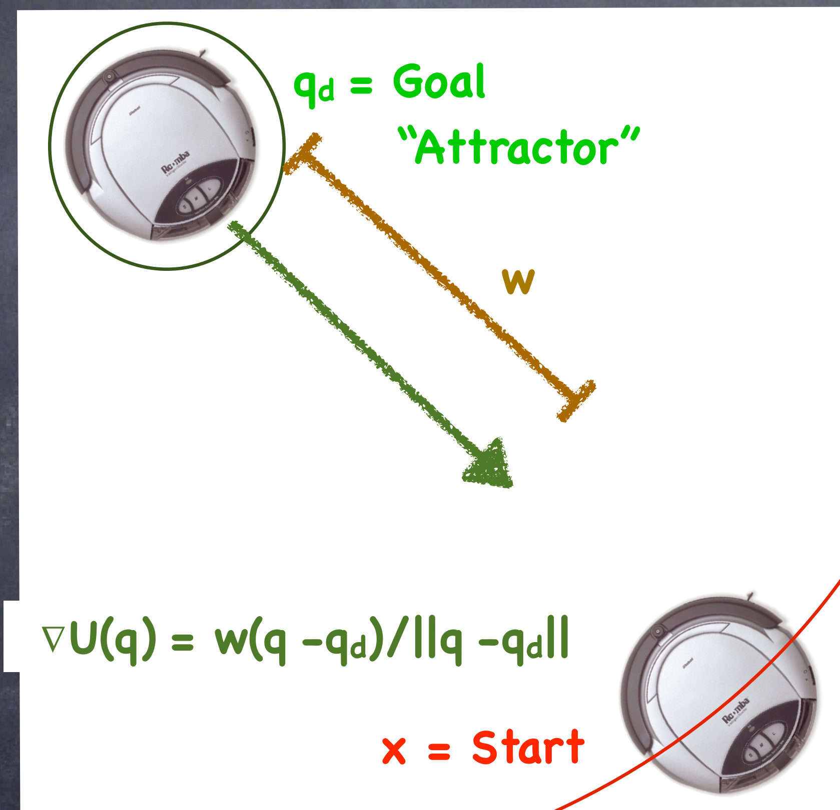
top view

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



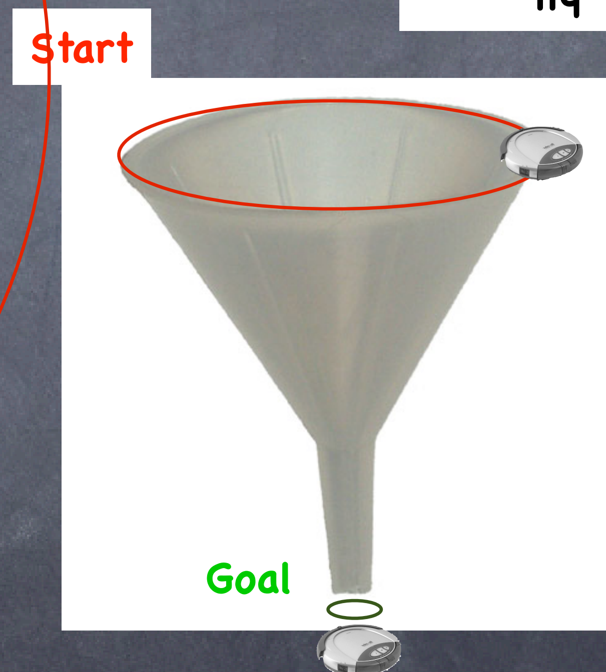
side view

“Cone” Attractor



top view

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

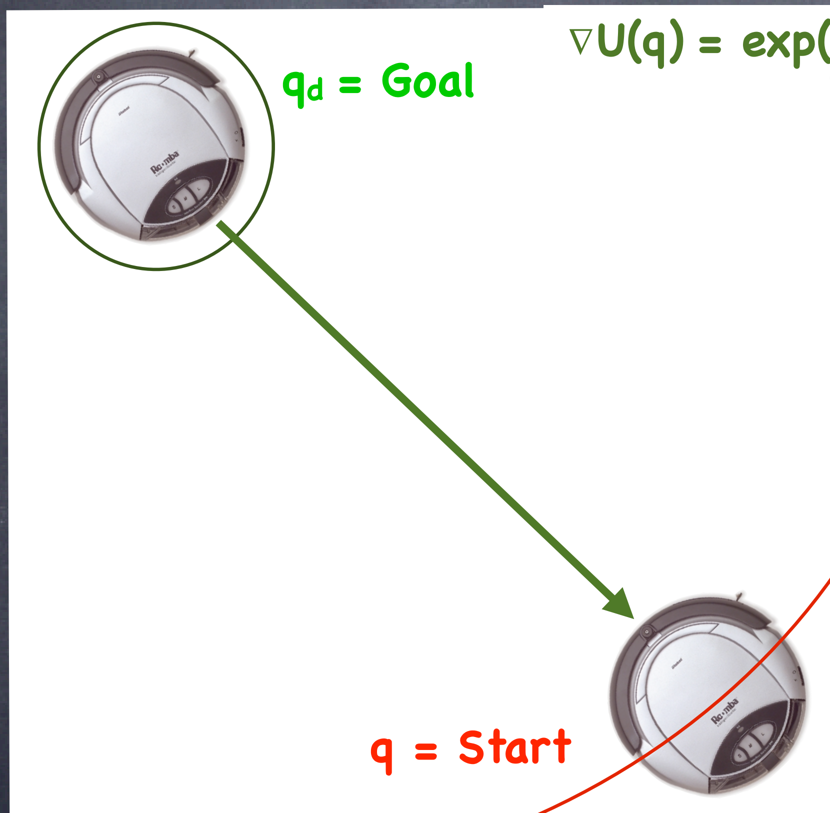


side view

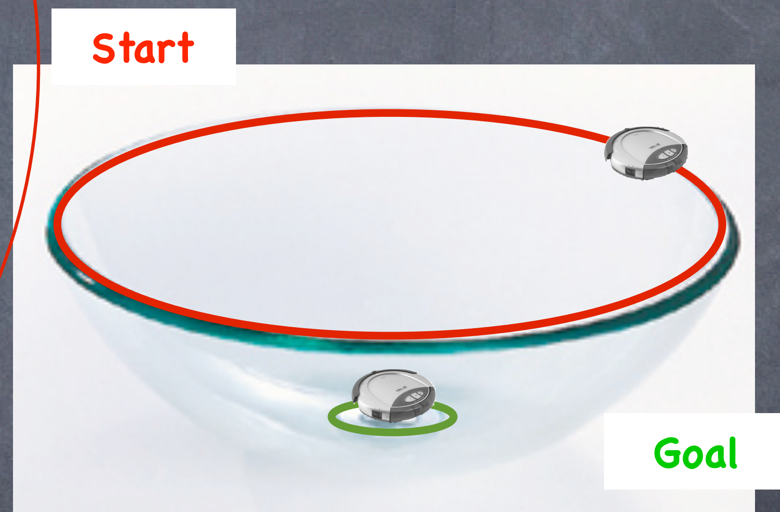
Can we modulate the
range of a potential field?

"Bowl" Attractor

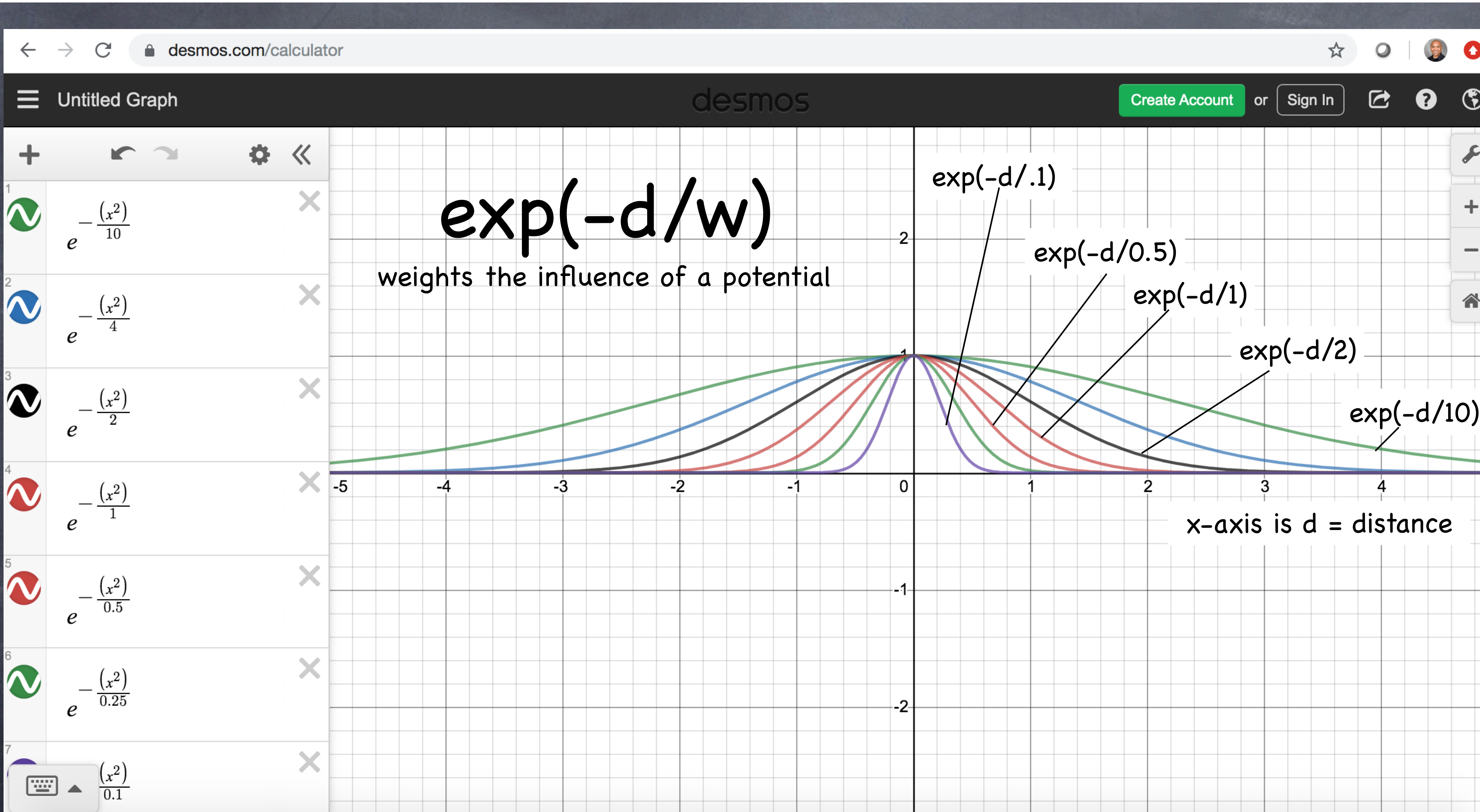
$\nabla U(q) = \exp(-\|q - q_d\|/w)\,(q - q_d)$



top view

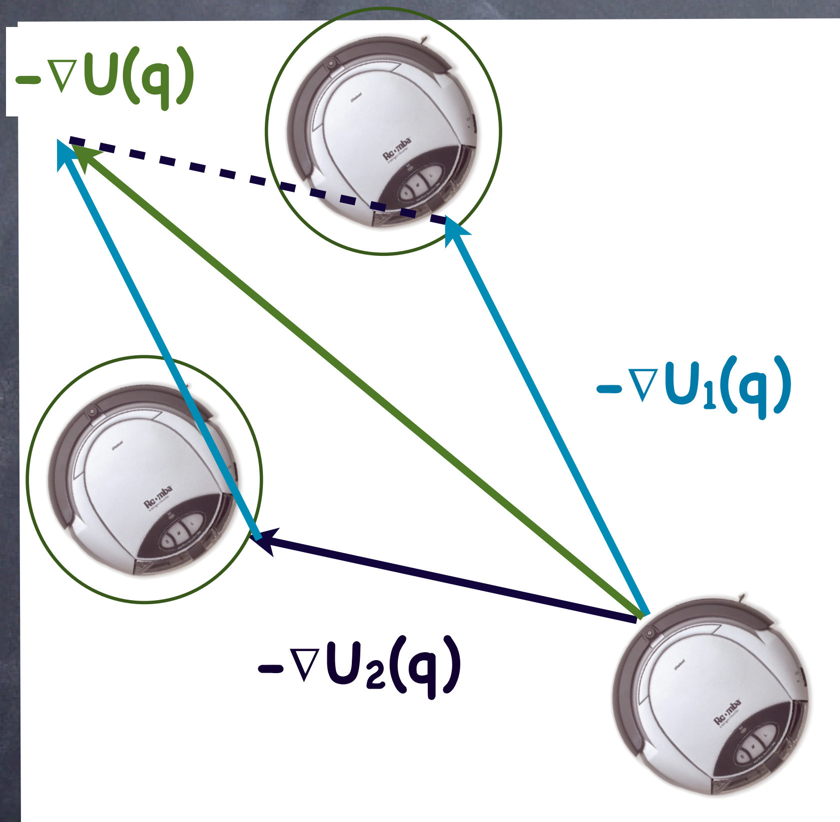


side view


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

Can we combine multiple potentials?

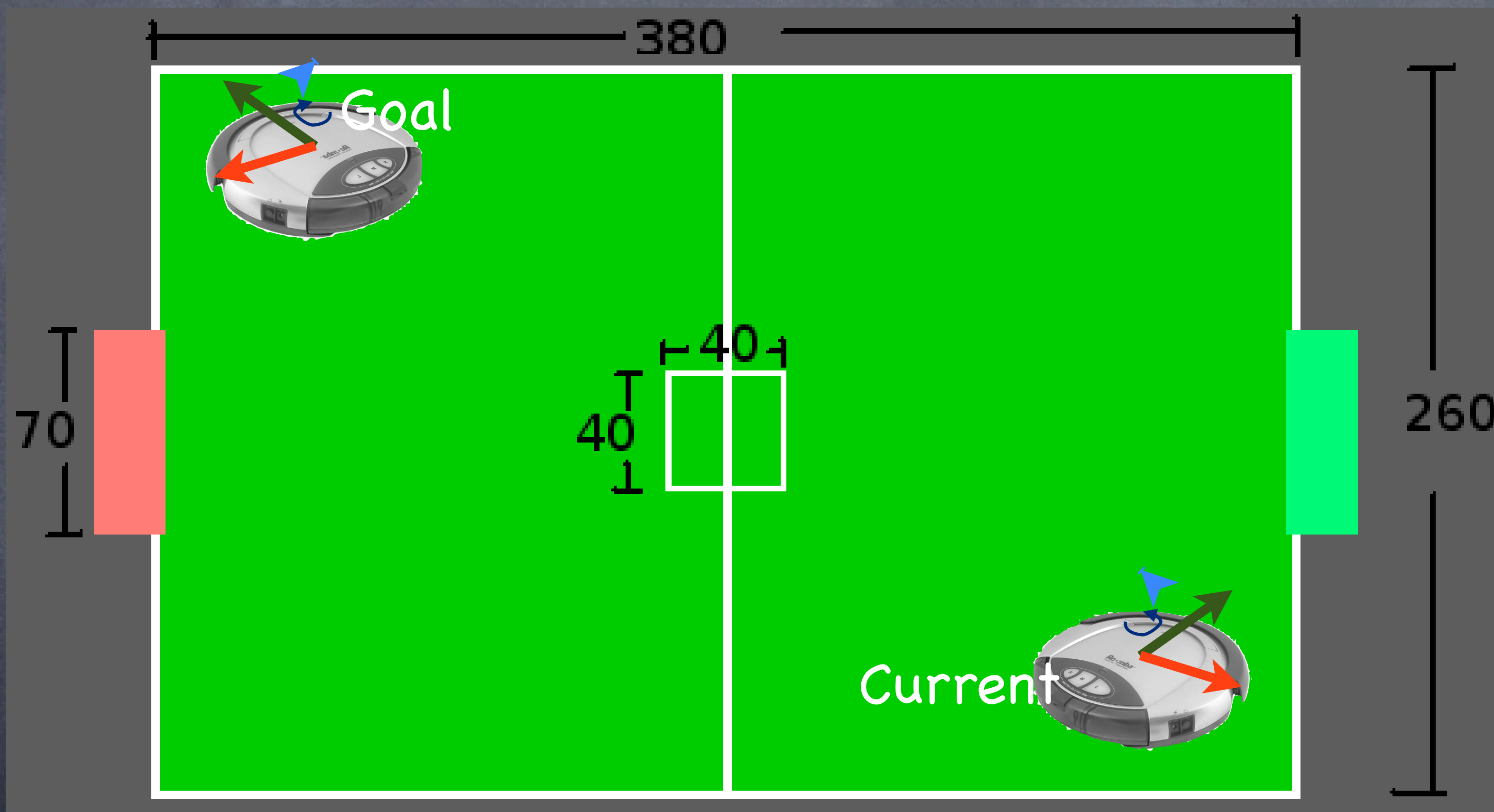
Multiple potentials



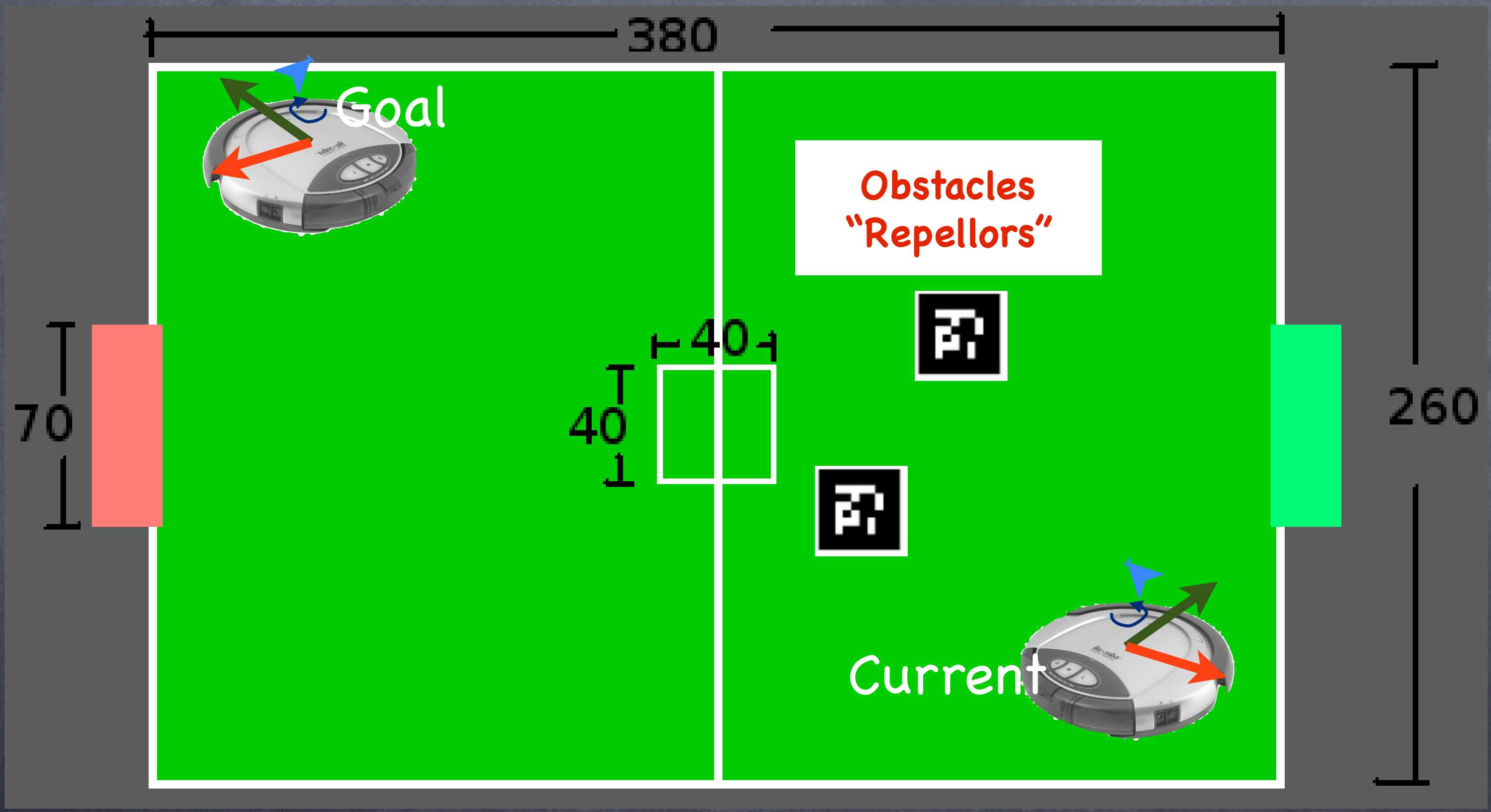
- Output of potential field is a vector
- Combine multiple potentials through vector summation

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

describe performance for this case

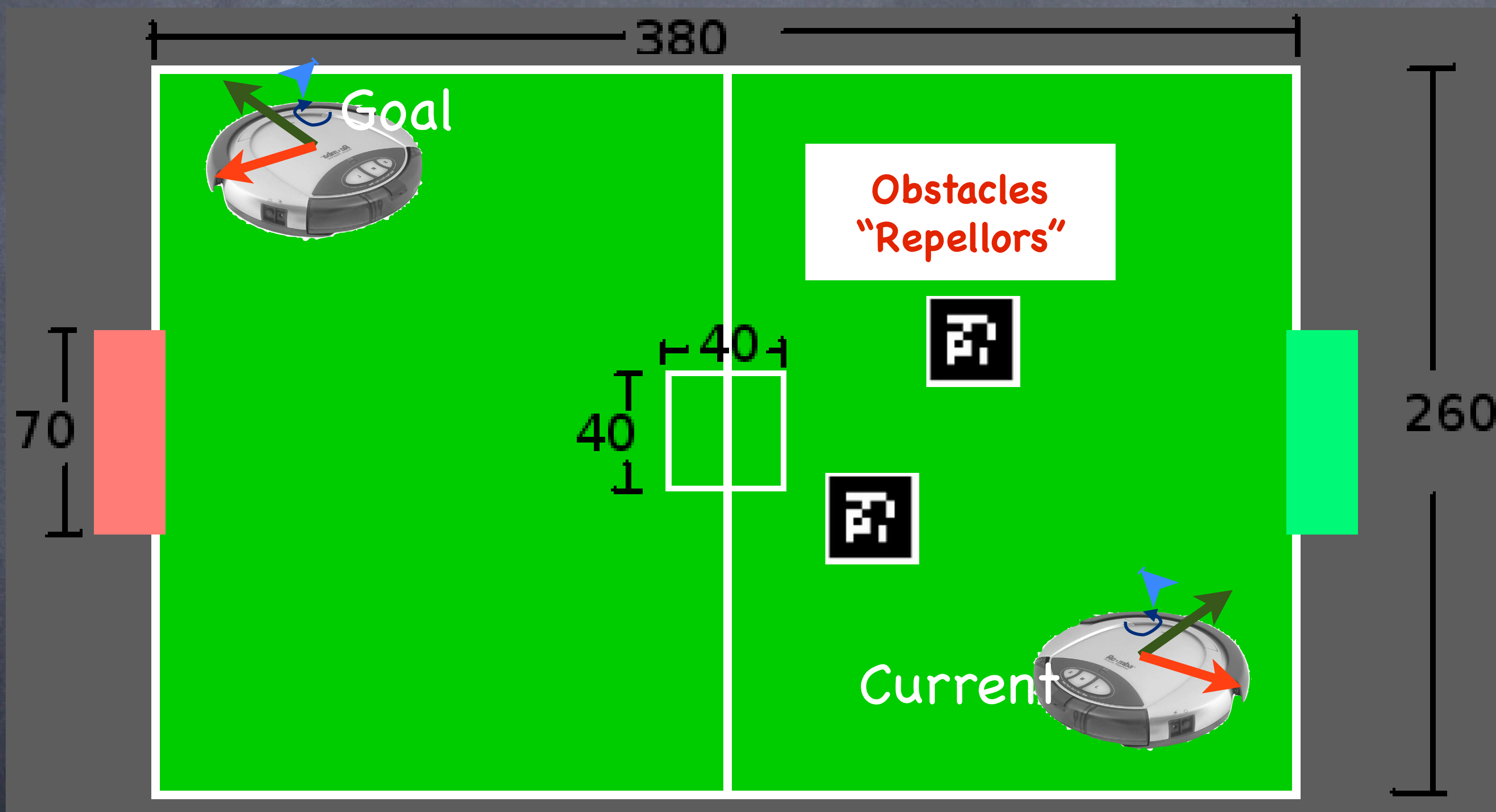


describe performance for this case



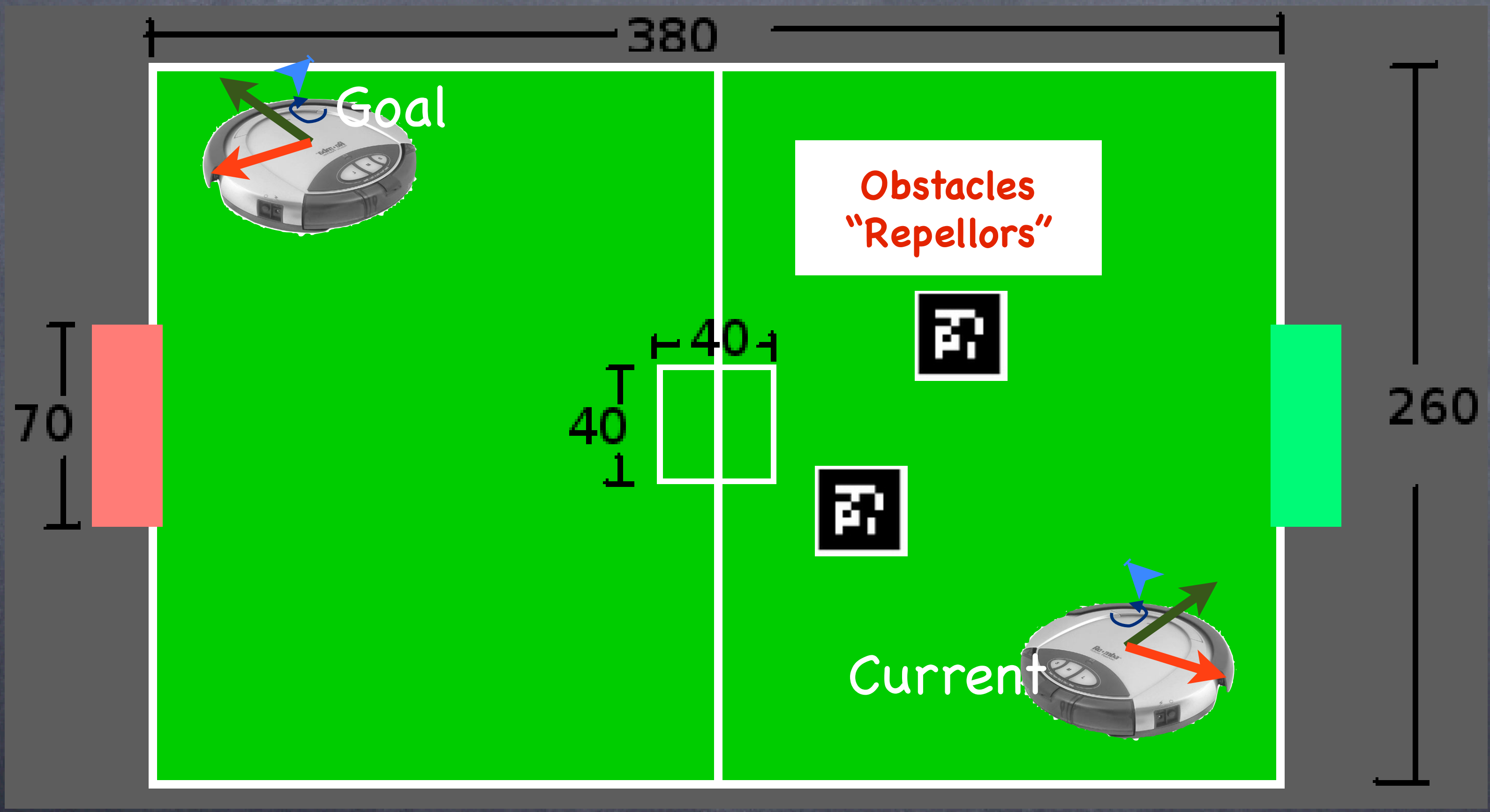
describe performance for this case

how do we deal with repellors?



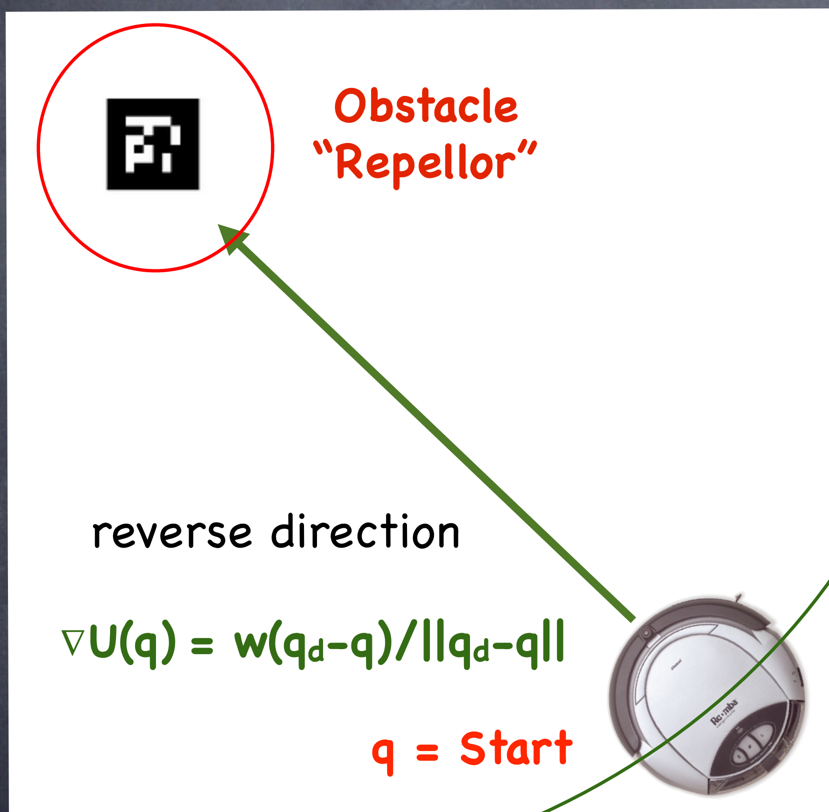
add sum of repulsive potentials

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



“Cone” Repellor

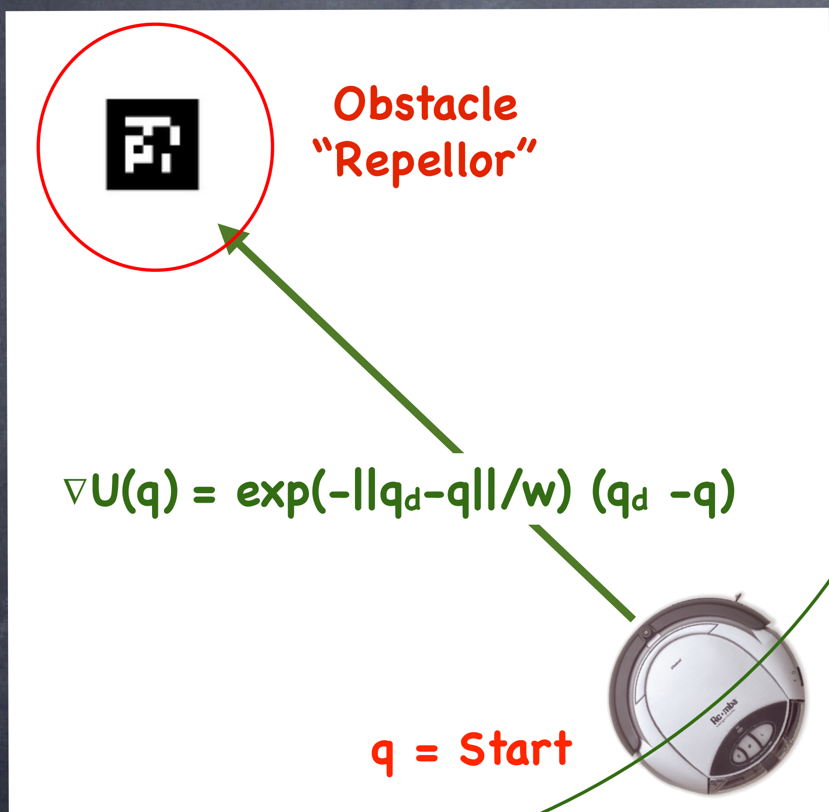
potential problems?



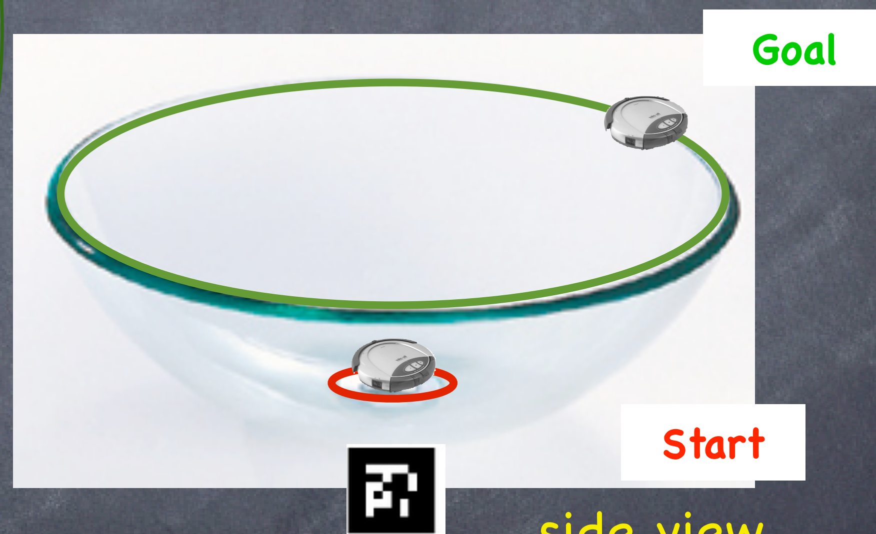
top view

side view

"Bowl" Repellor

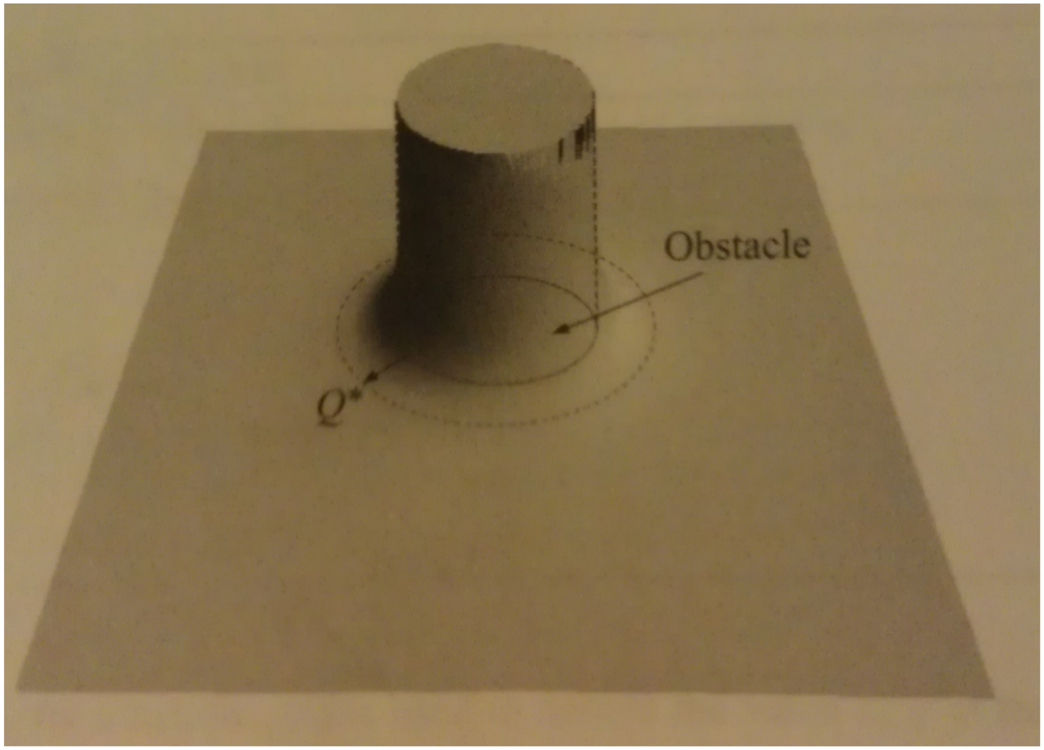


top view



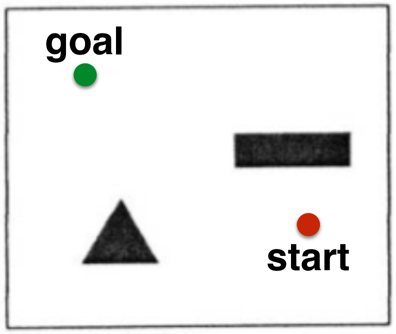
side view

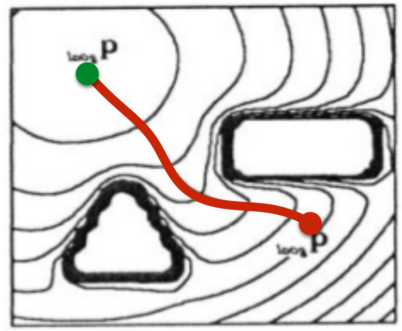
repellor should only have local influence,
repelling only around boundary improves path



2 Obstacle example

configuration space

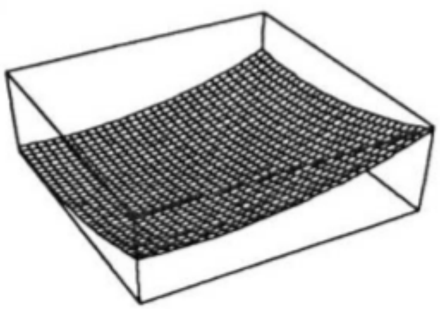




path from descent on gradient field

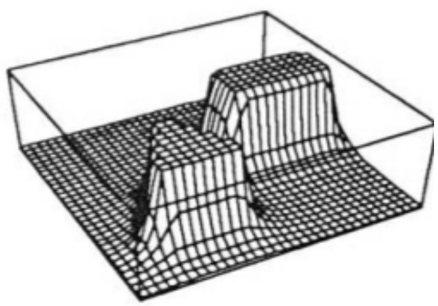
resulting gradient field

attractor field



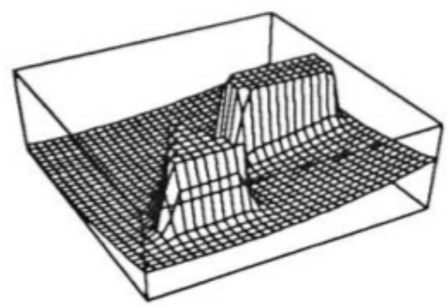
+

repellor fields

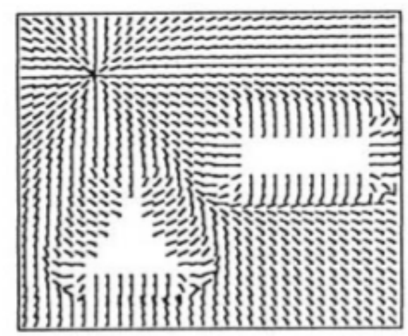


=

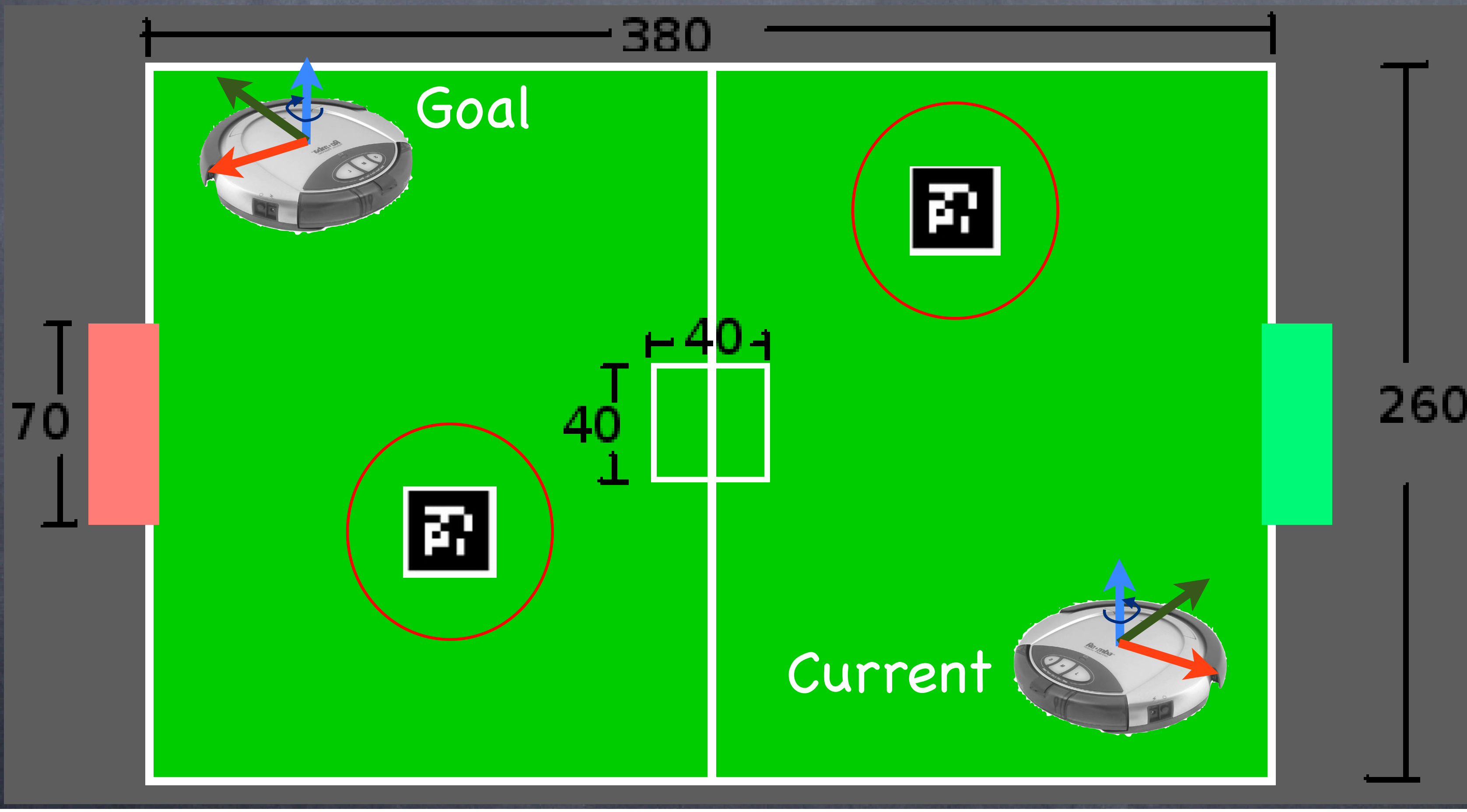
combined potential



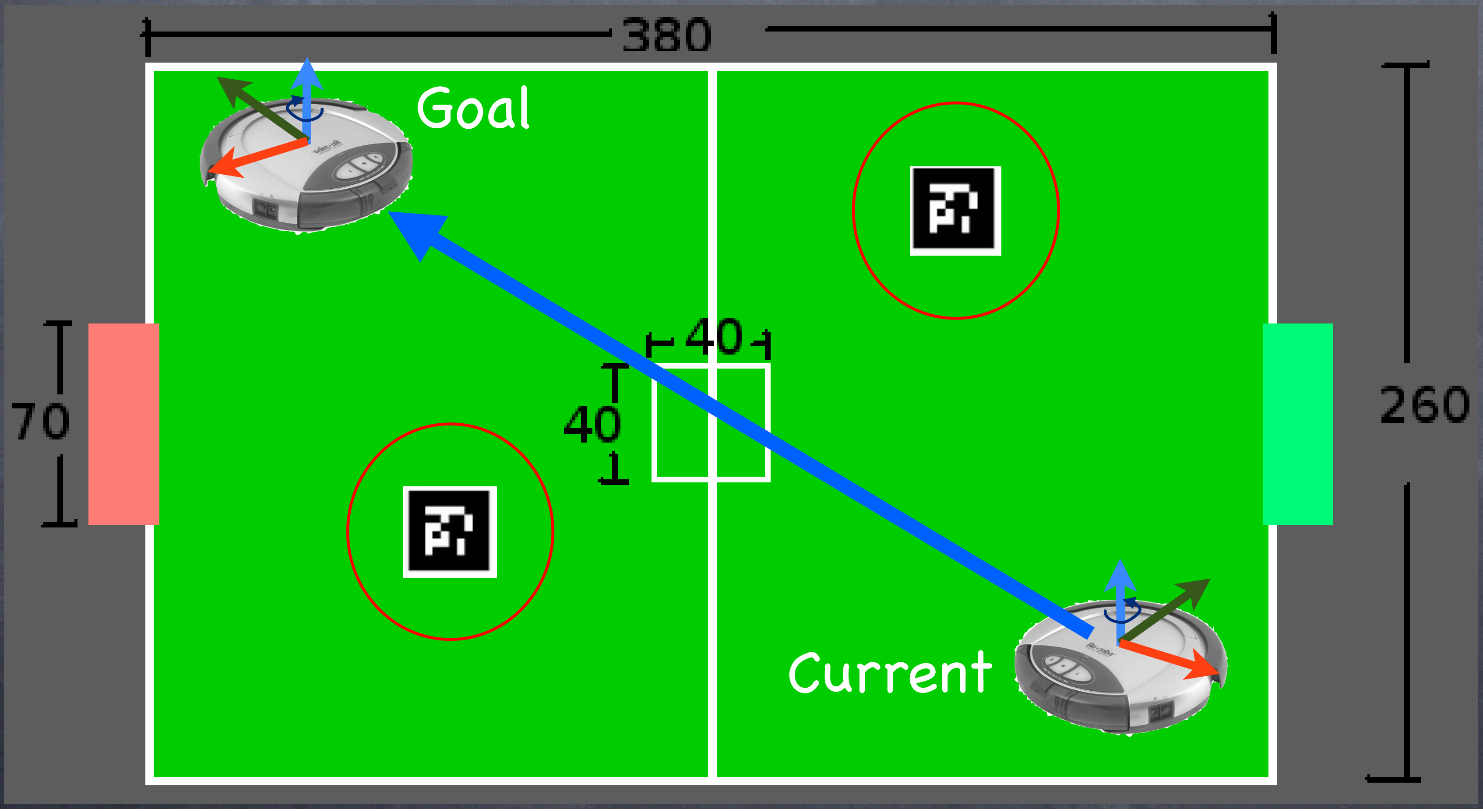
gradient field



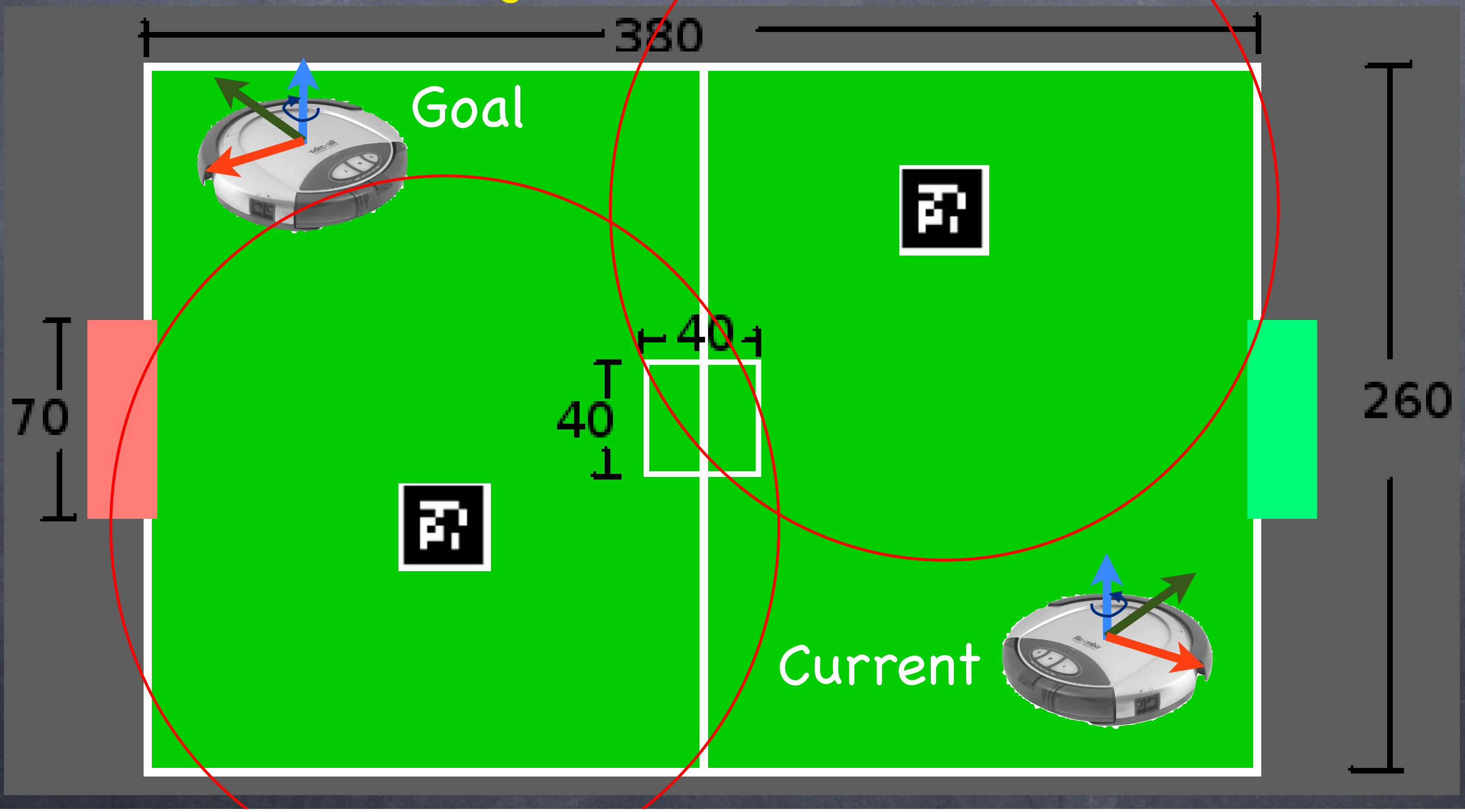
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



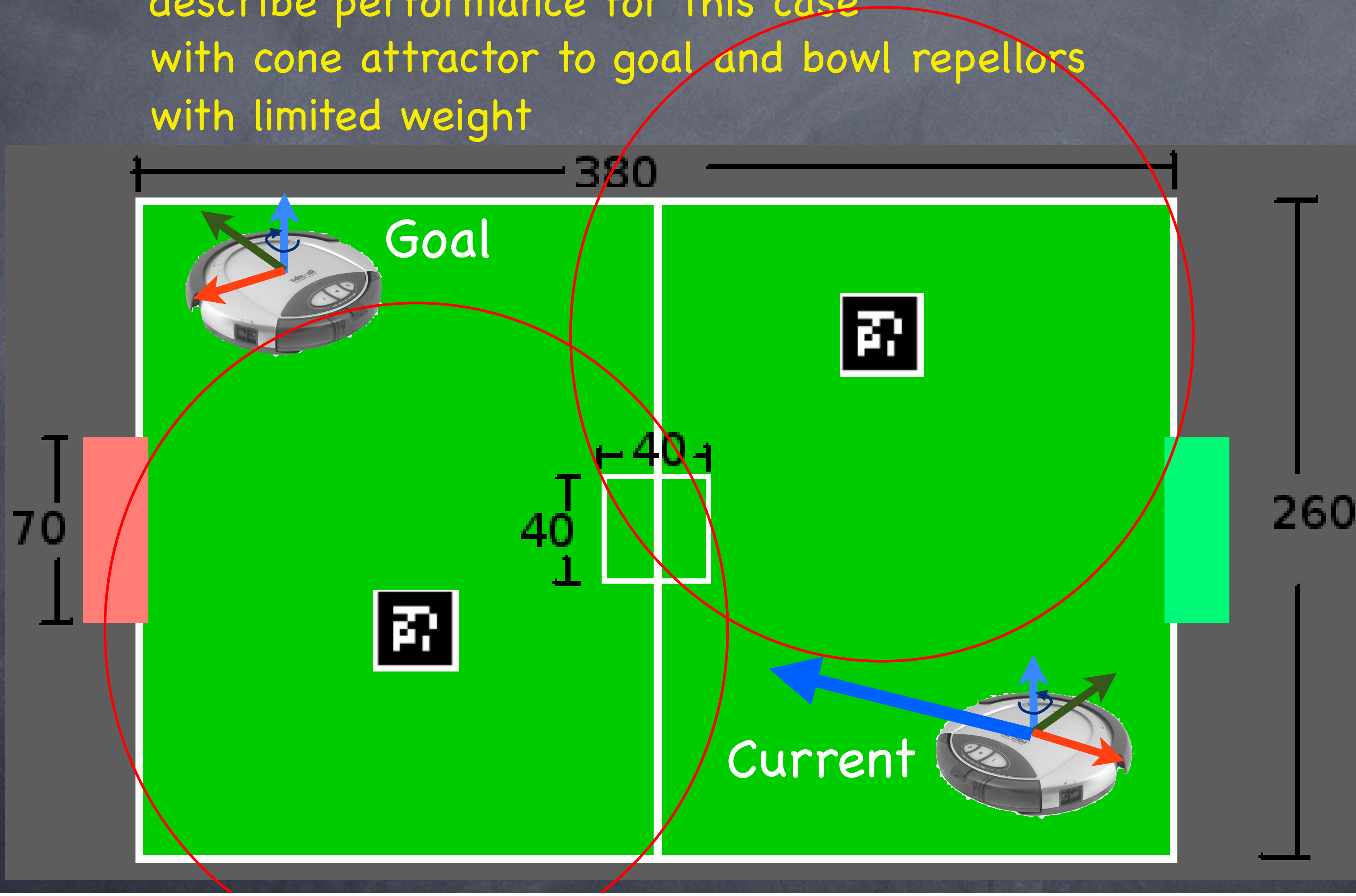
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



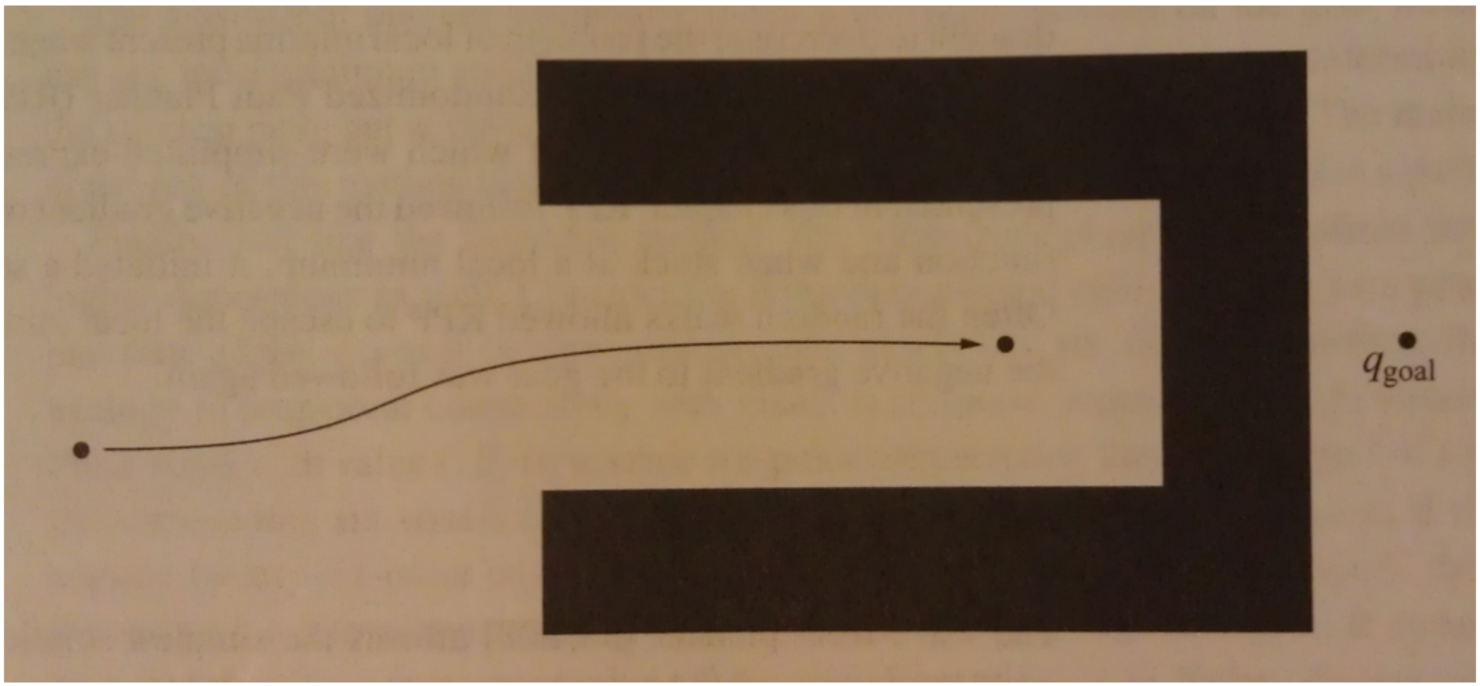
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



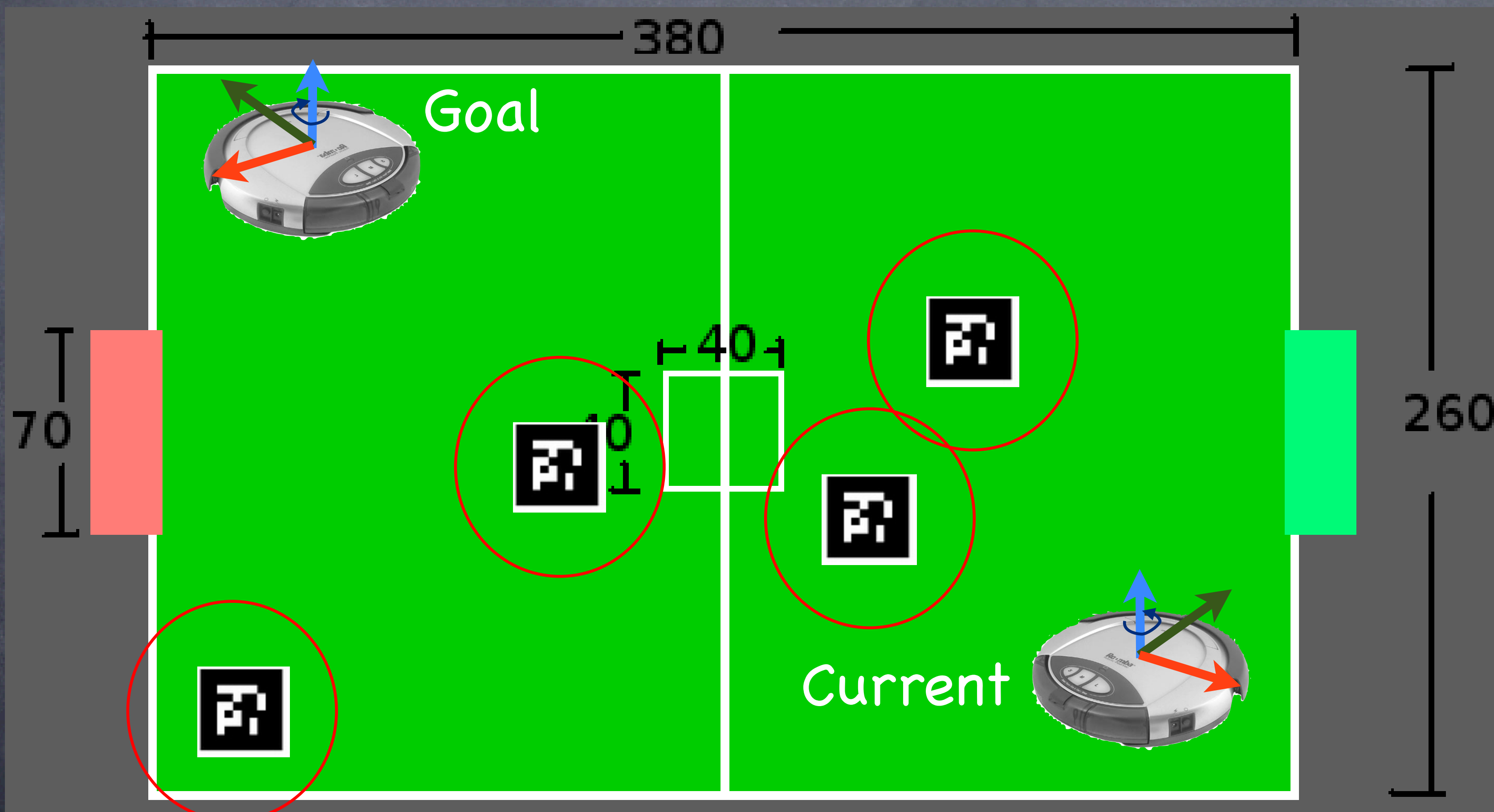
describe performance for this case
with cone attractor to goal and bowl repellors
with limited weight



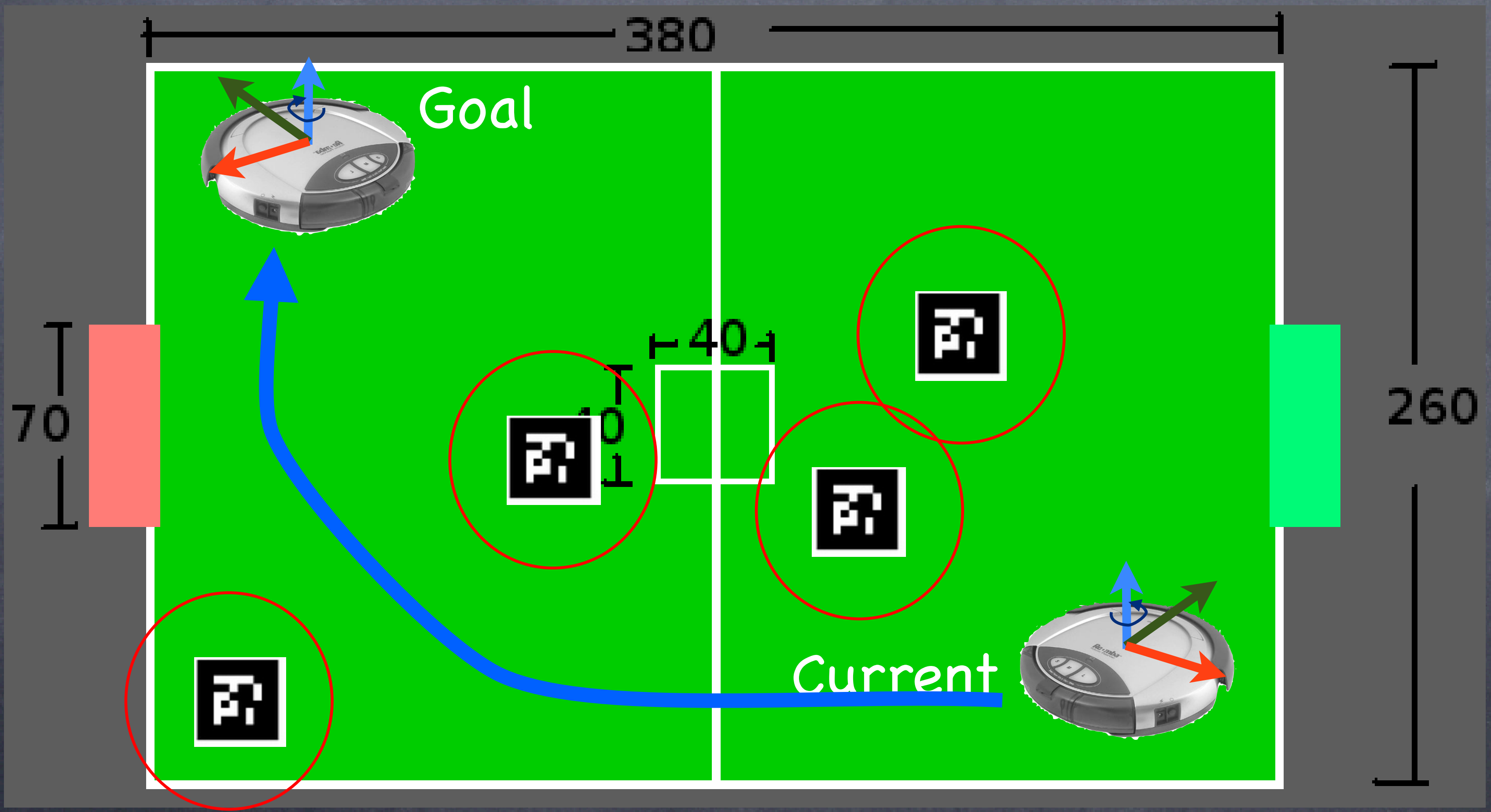
Local Minima



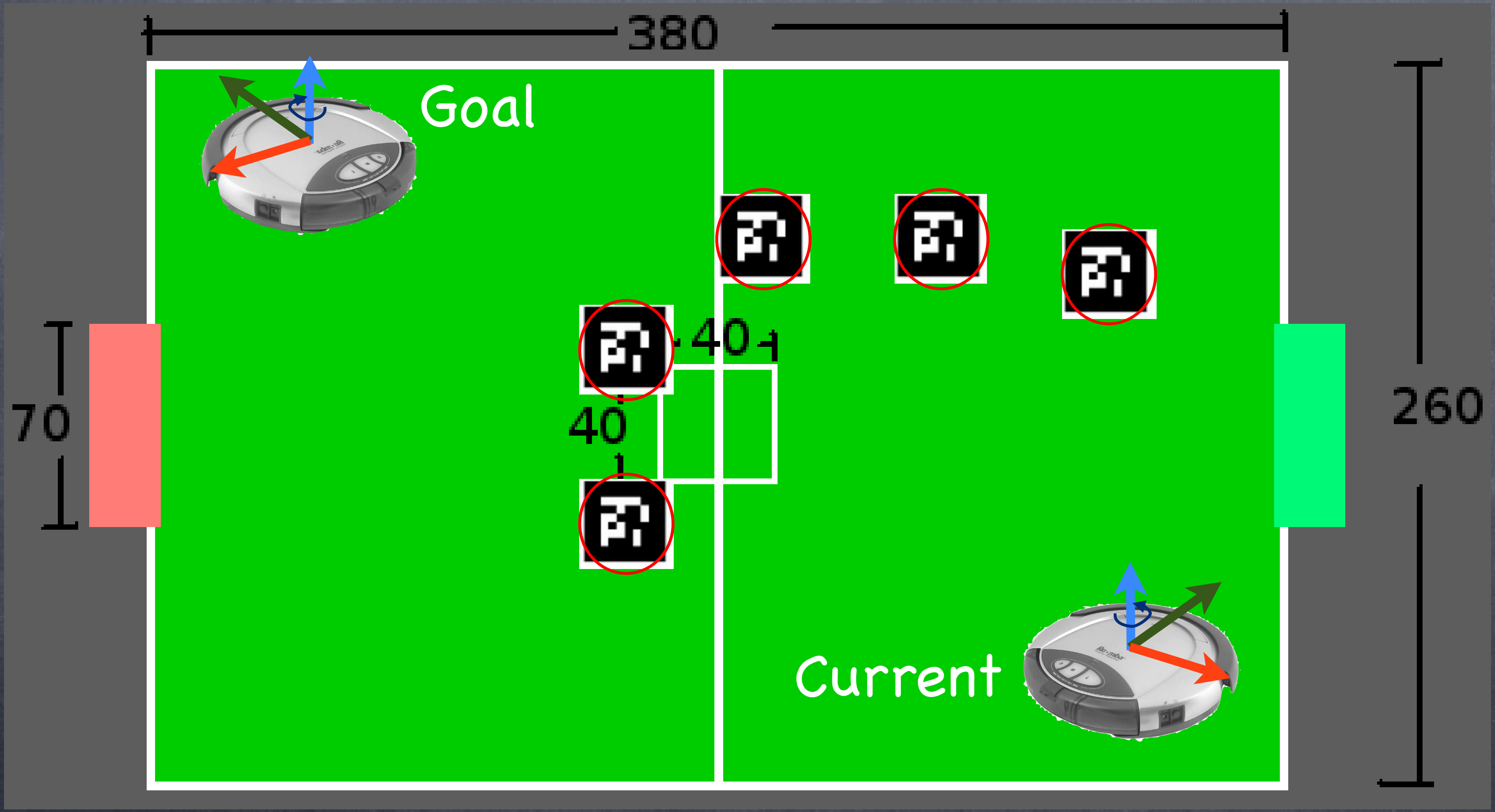
describe performance for this case



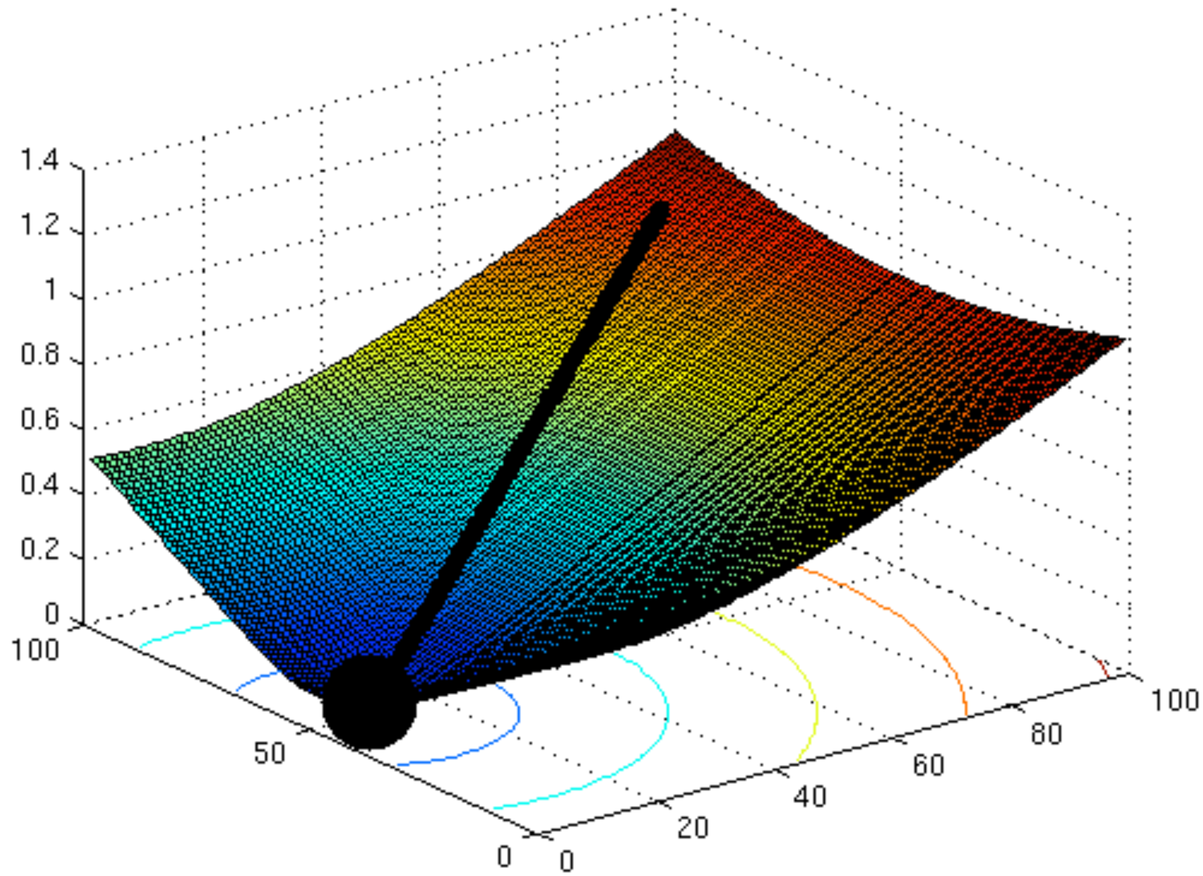
describe performance for this case



describe performance for this case

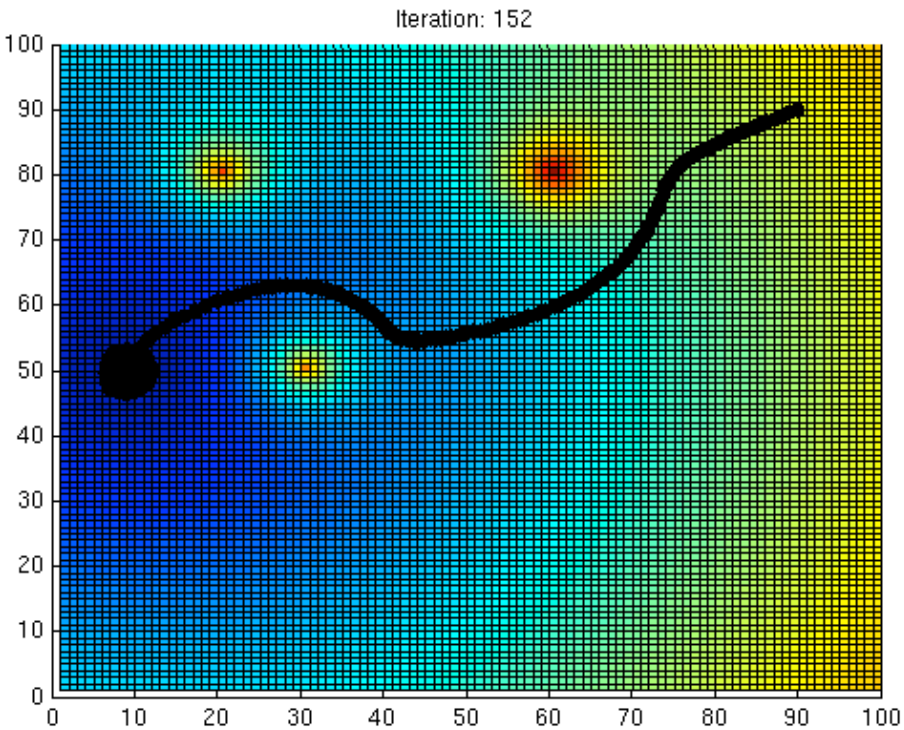


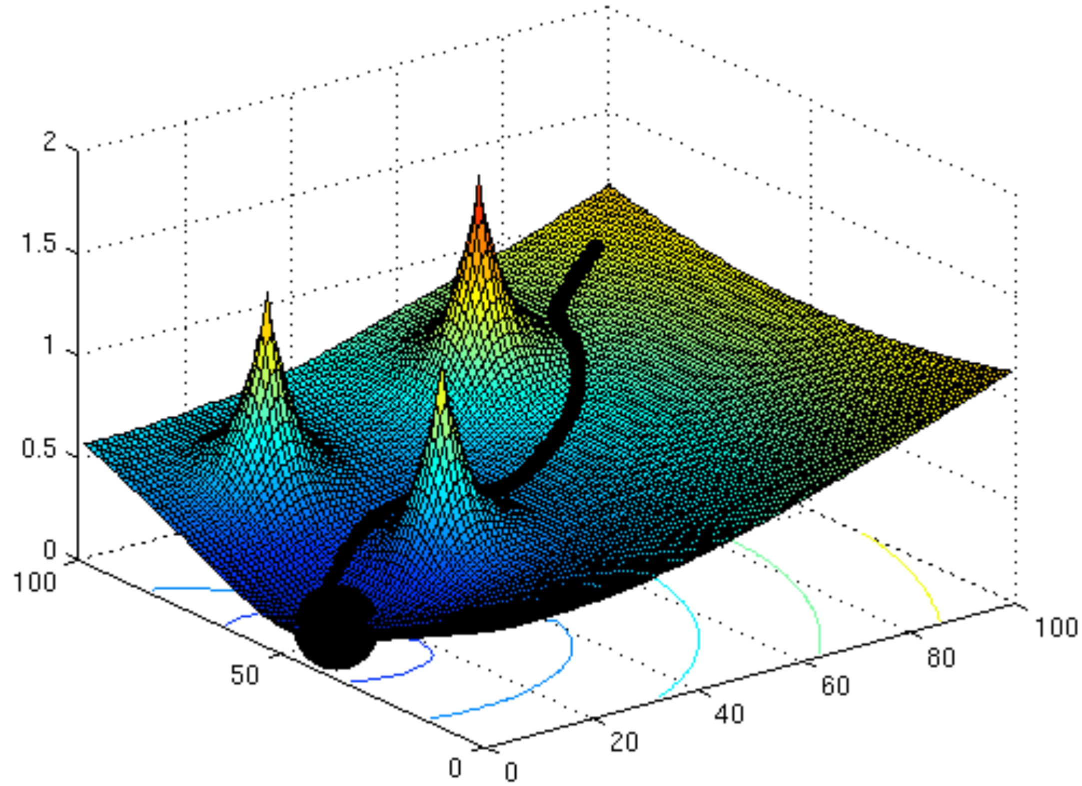
matlab example



pfield.m [1 5 8 12]

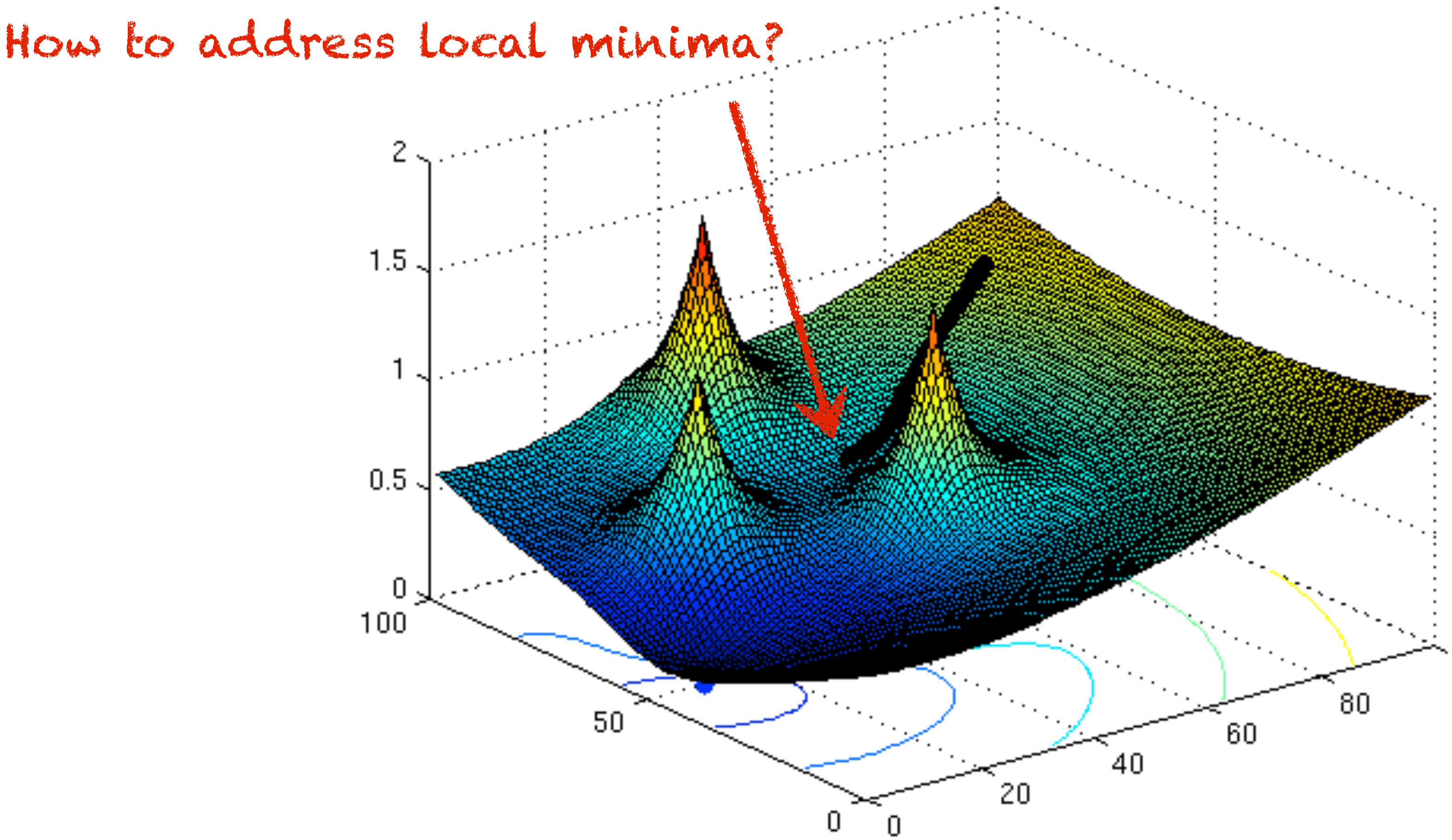
matlab example





pfield.m [1 5 8 12]

matlab example



pfield.m [1 5 8 12]

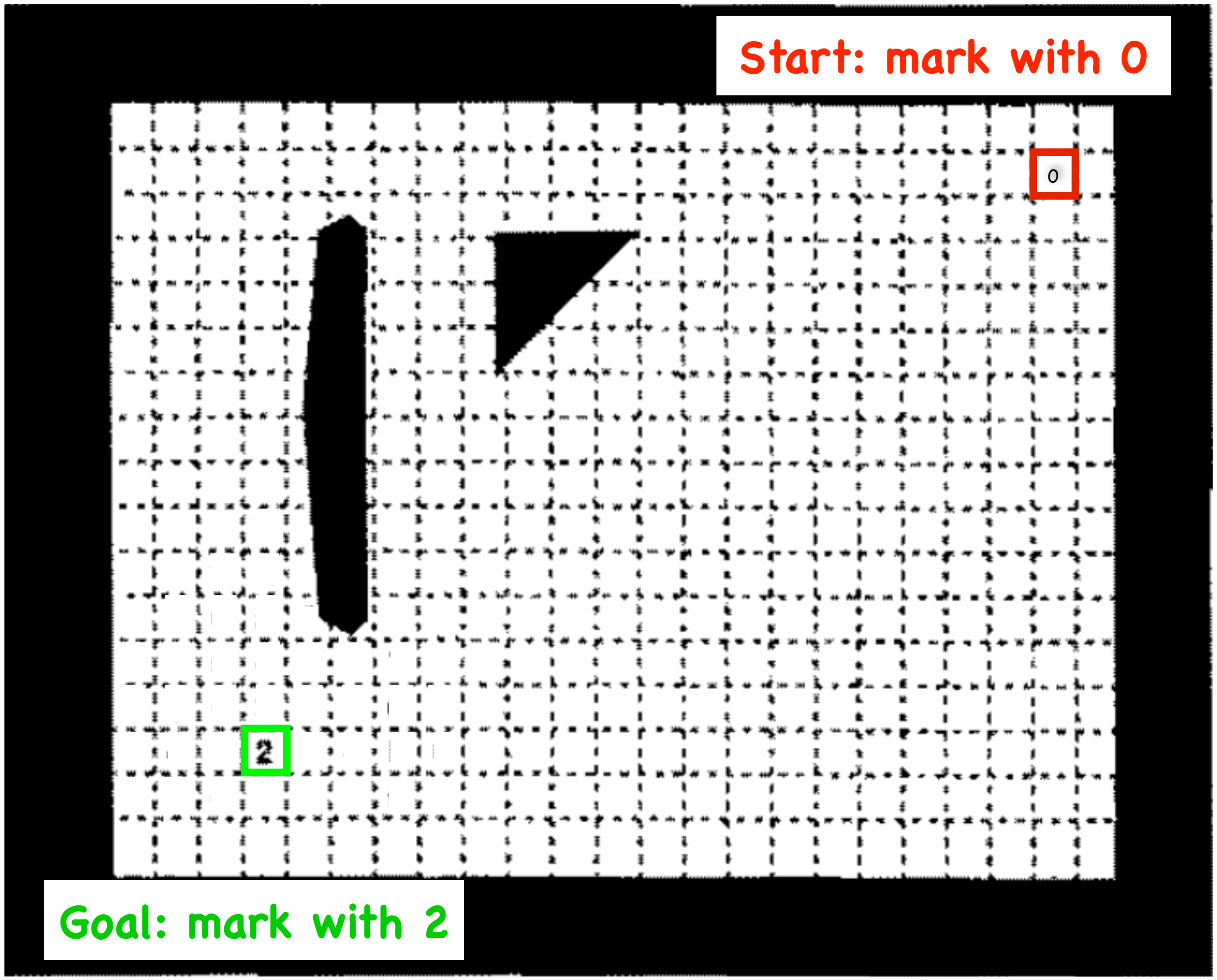
How can we get out of local minima?

How can we get out of local minima?

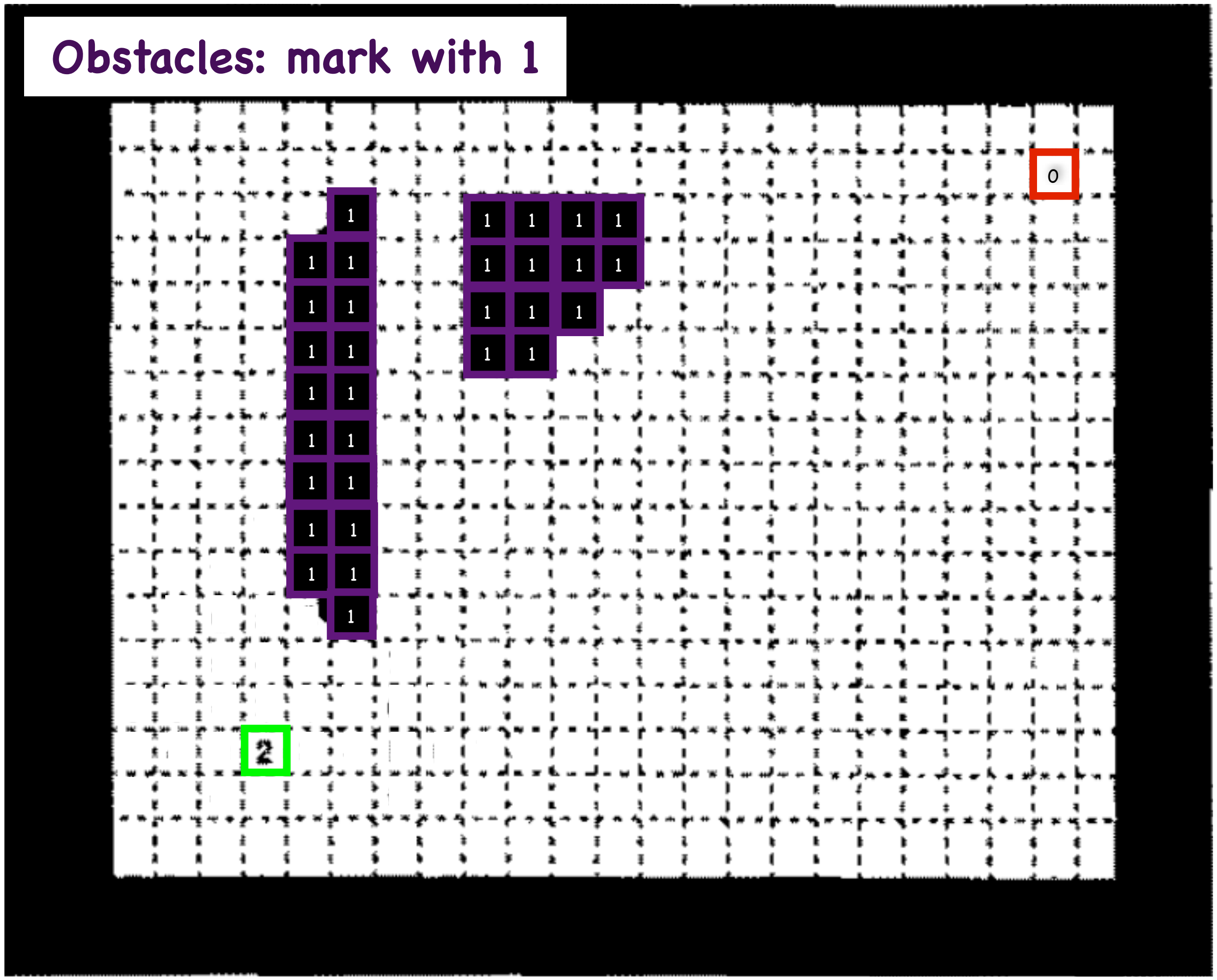
Go back to planning.

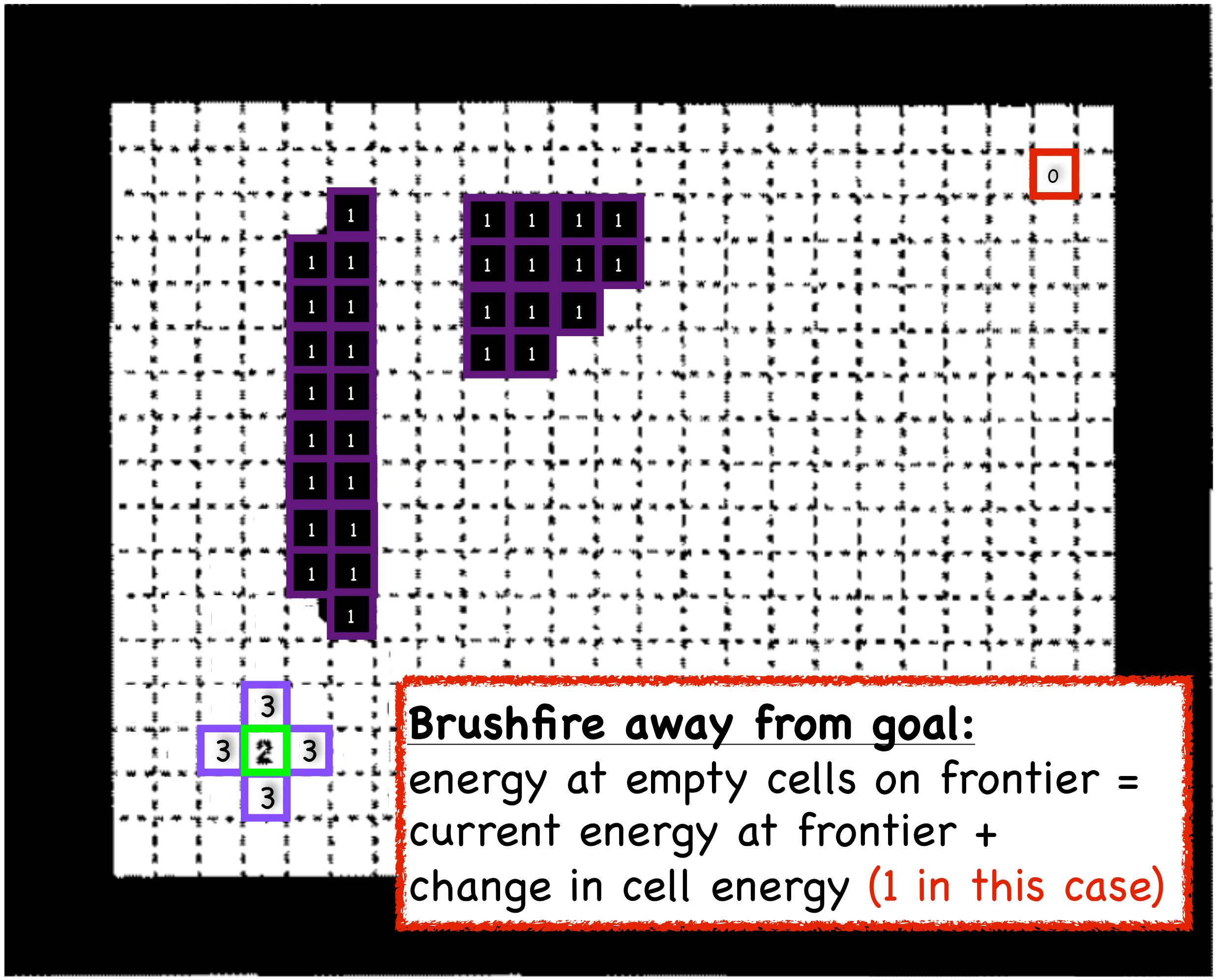
Wavefront Planning

- 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


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

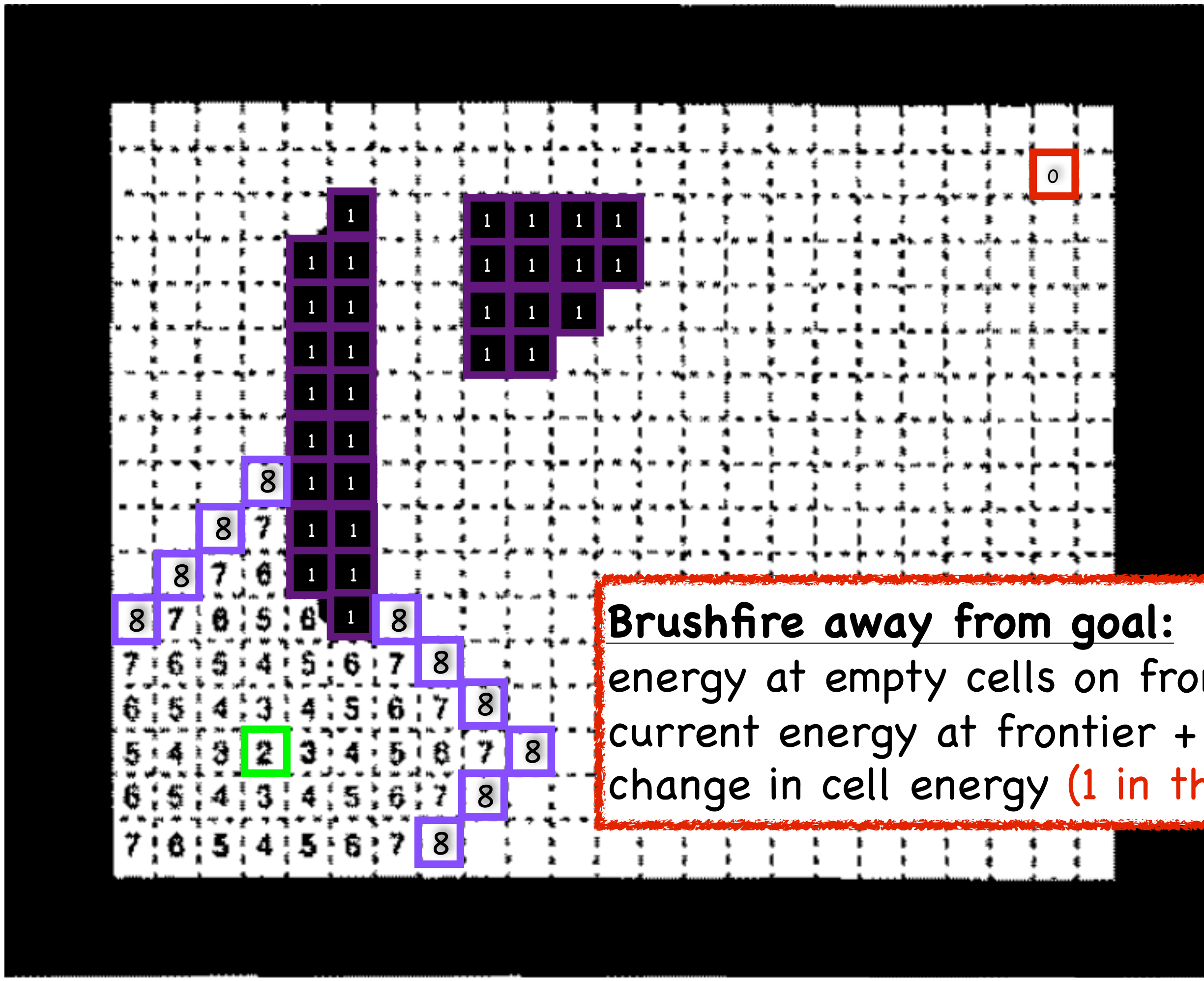
Obstacles: mark with 1



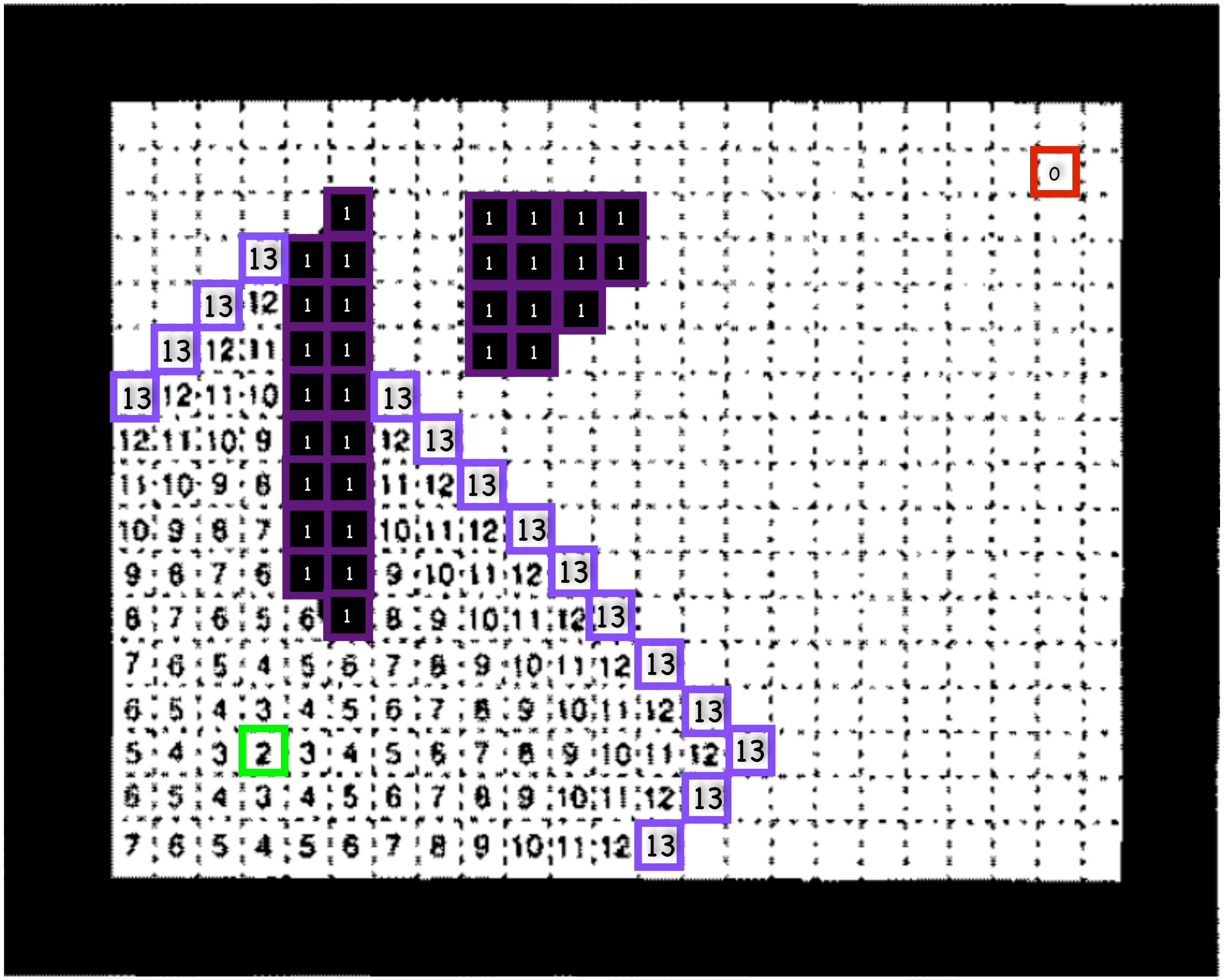


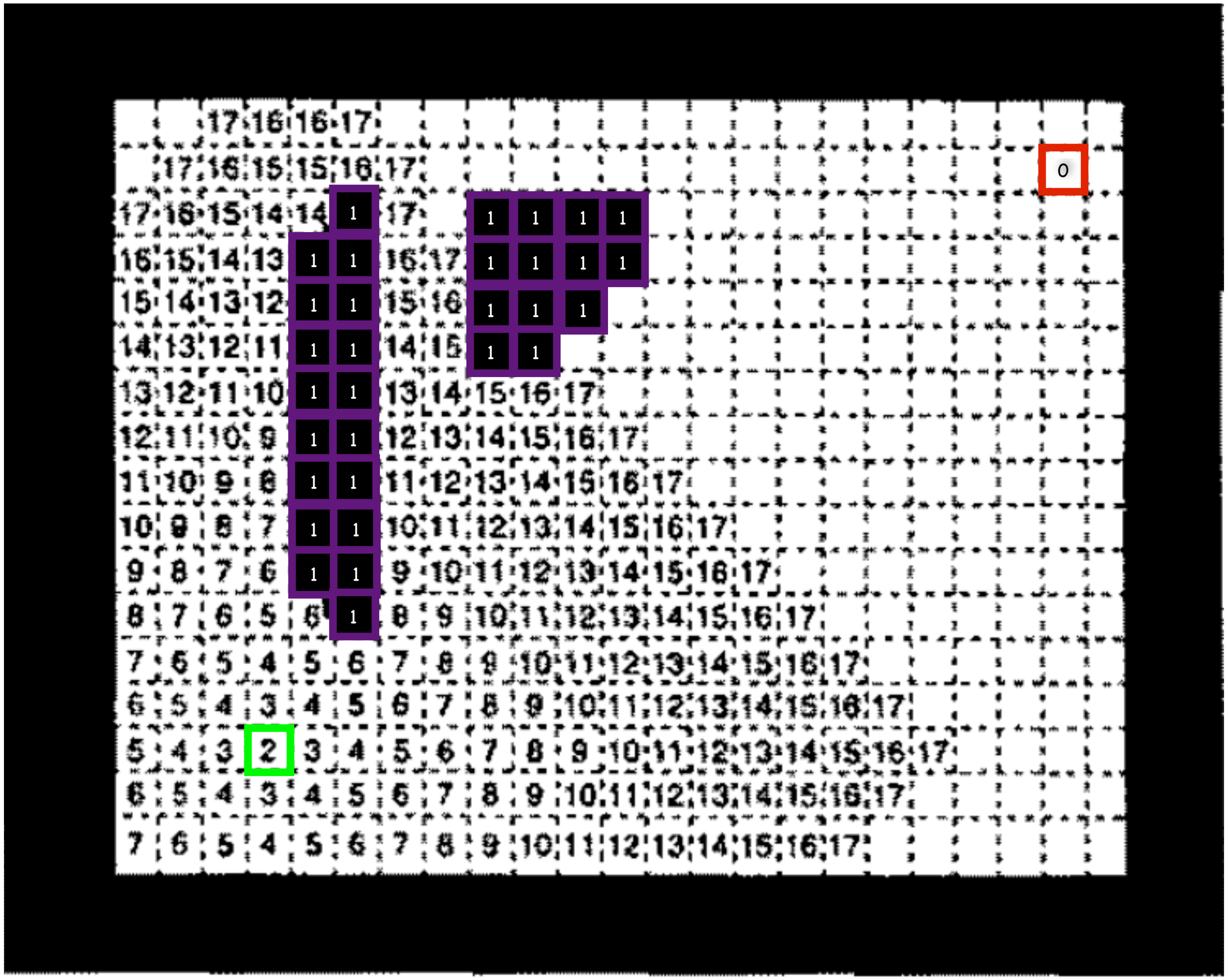
Brushfire away from goal:

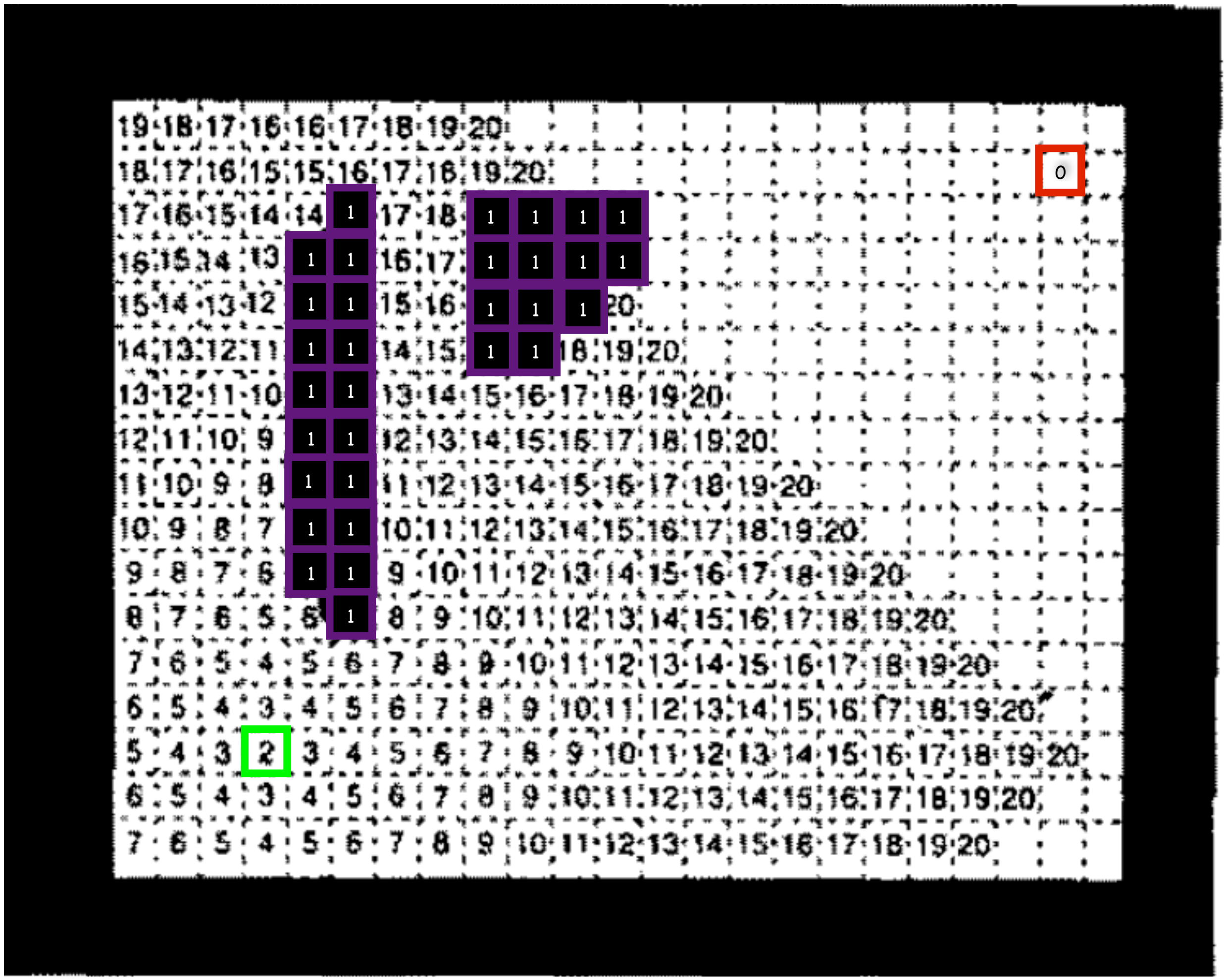
energy at empty cells on frontier =
current energy at frontier +
change in cell energy (1 in this case)

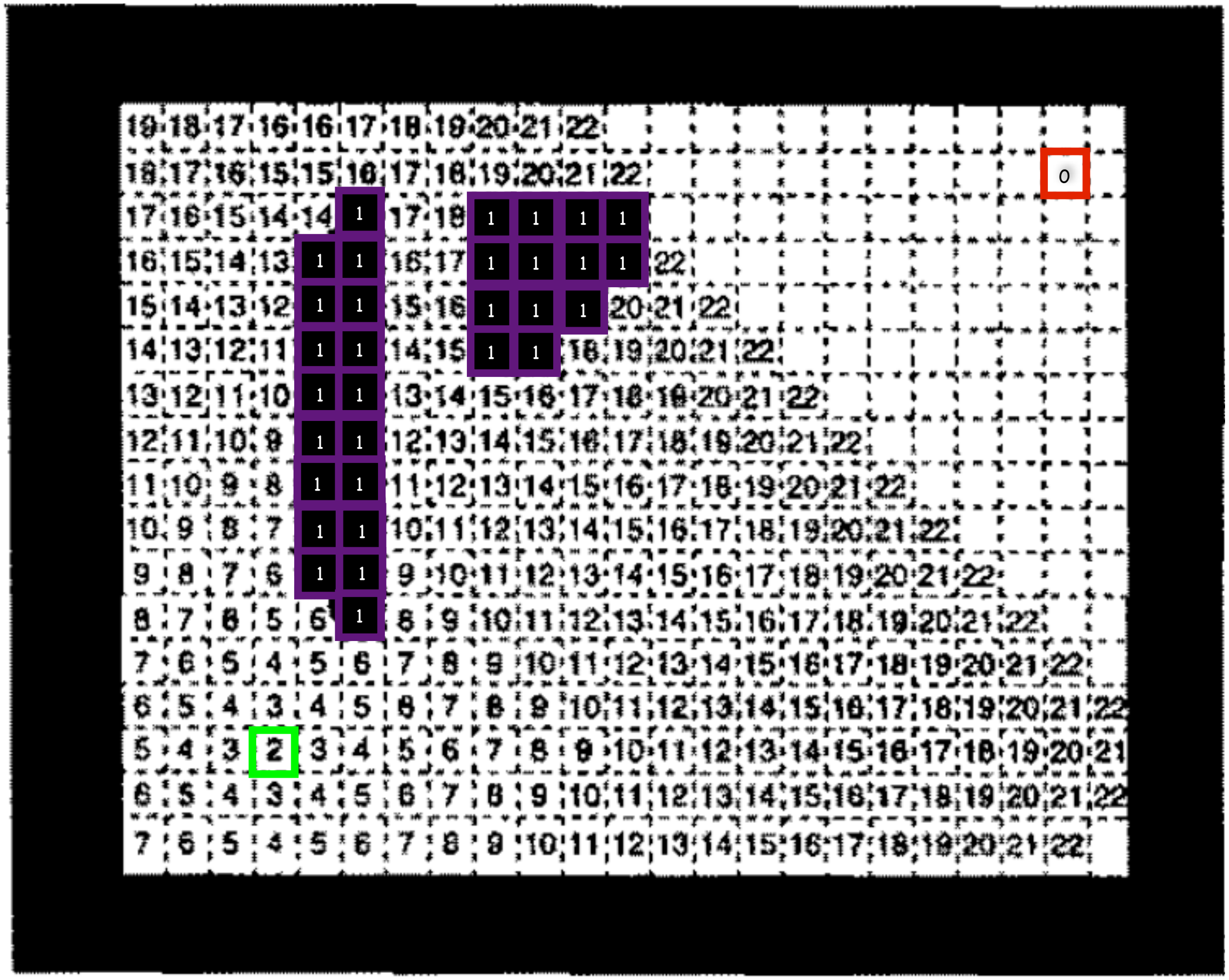


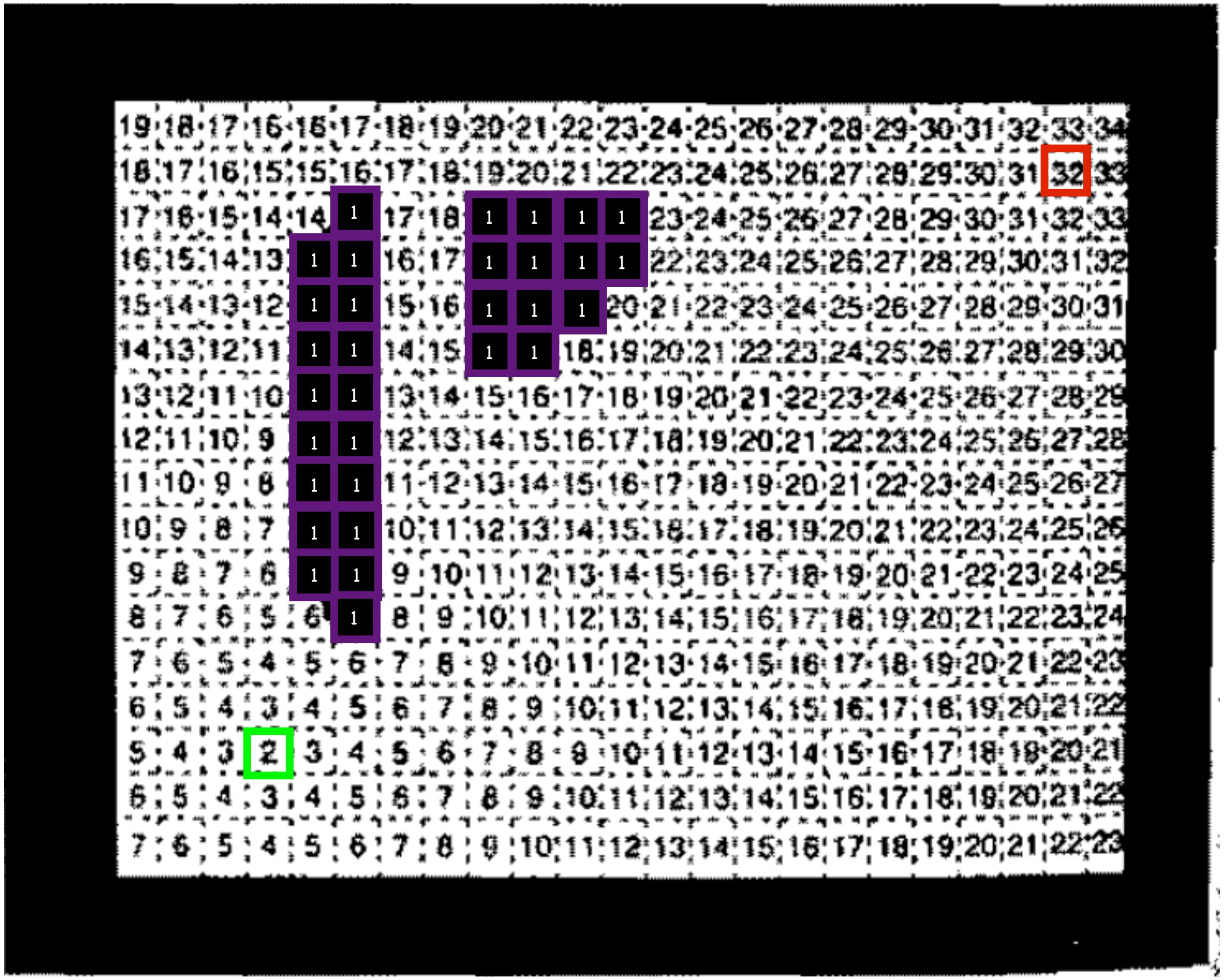
Brushfire away from goal:
energy at empty cells on frontier = current energy at frontier + change in cell energy (1 in this case)



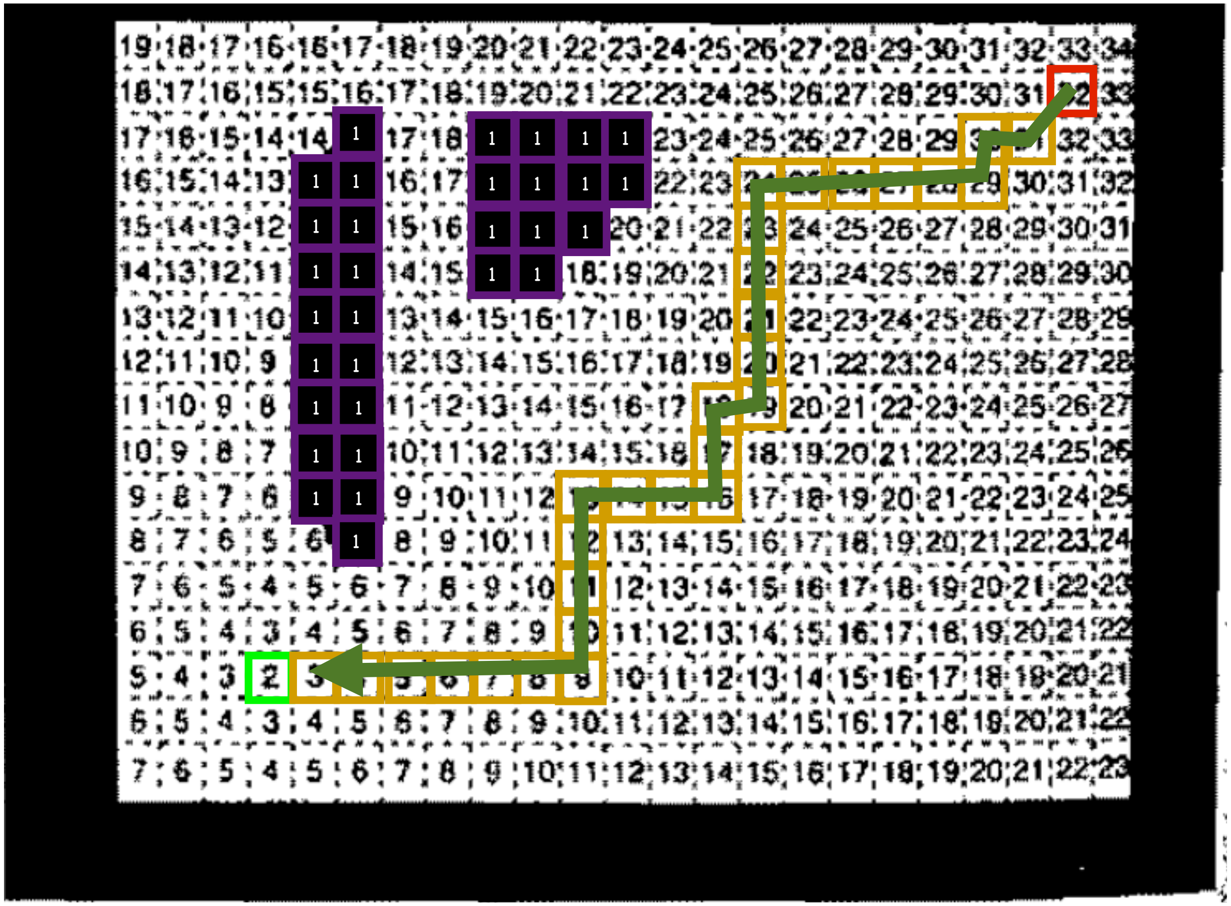






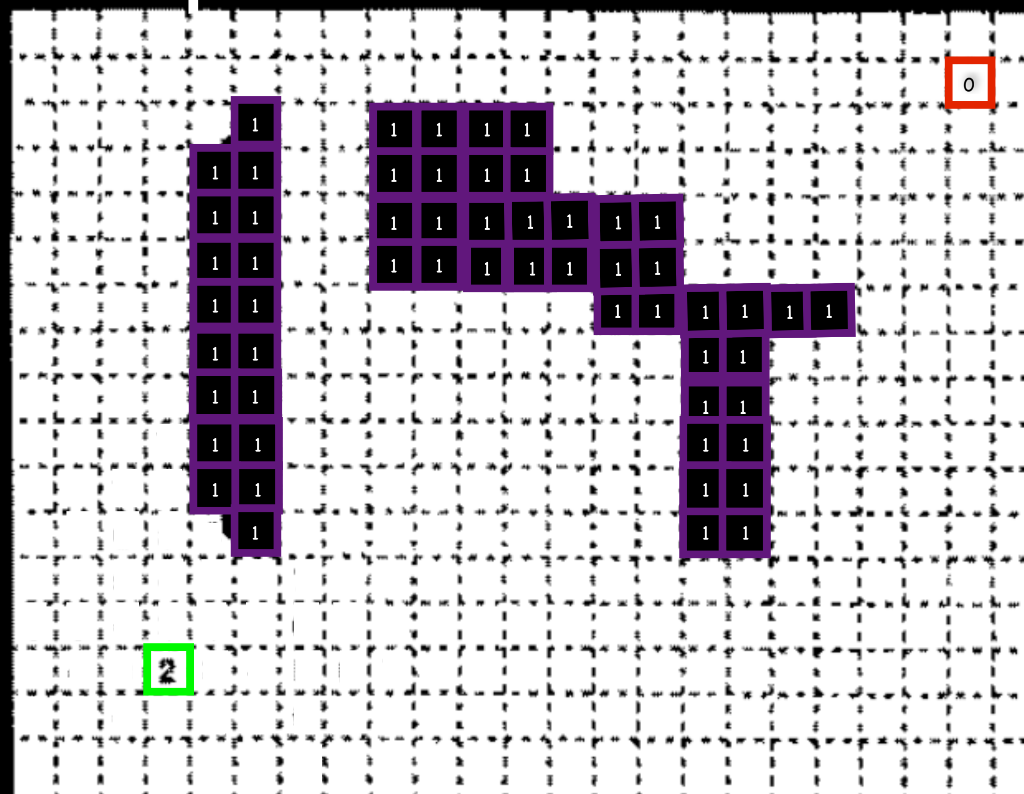


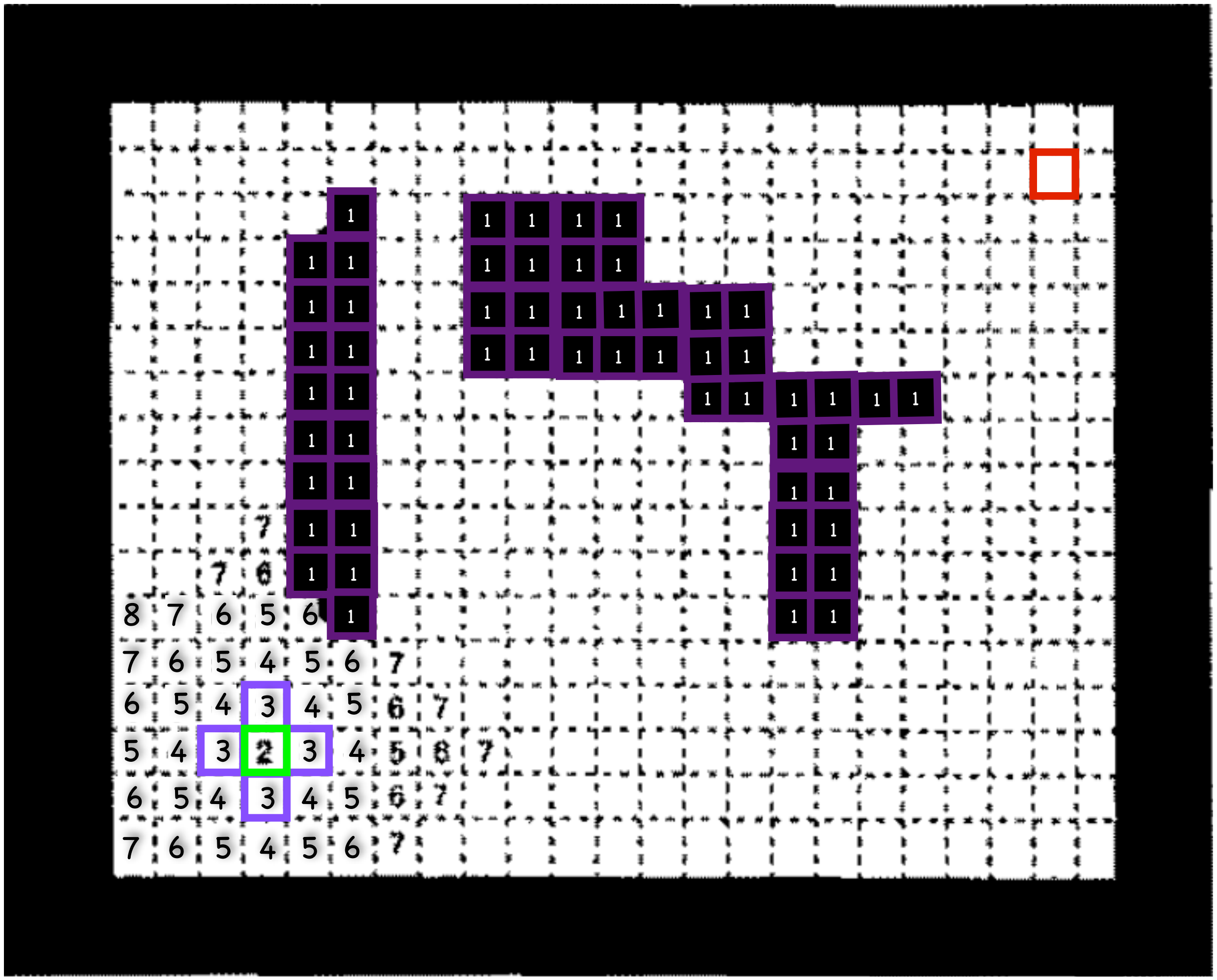
Once start reached, follow brushfire potential to goal

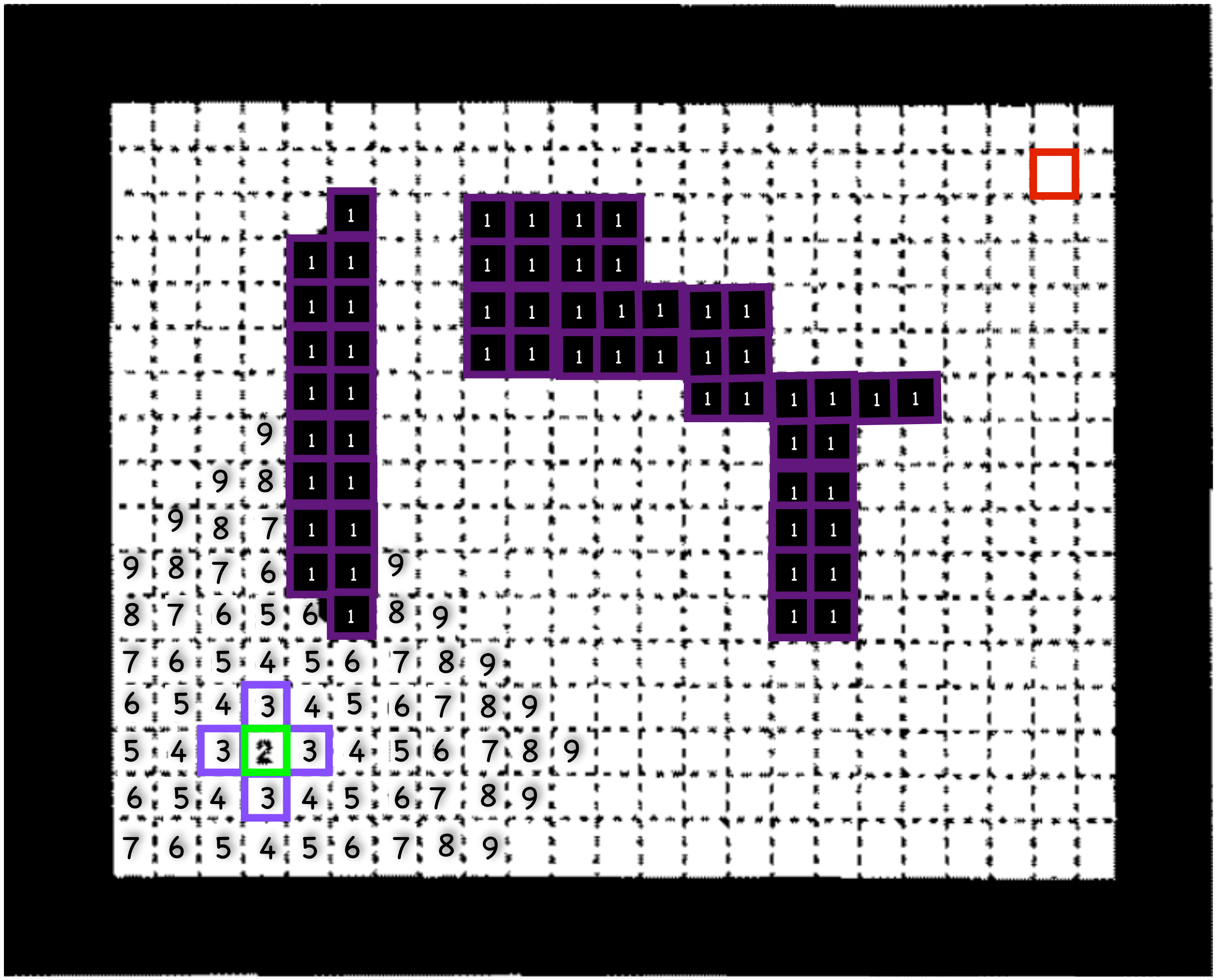


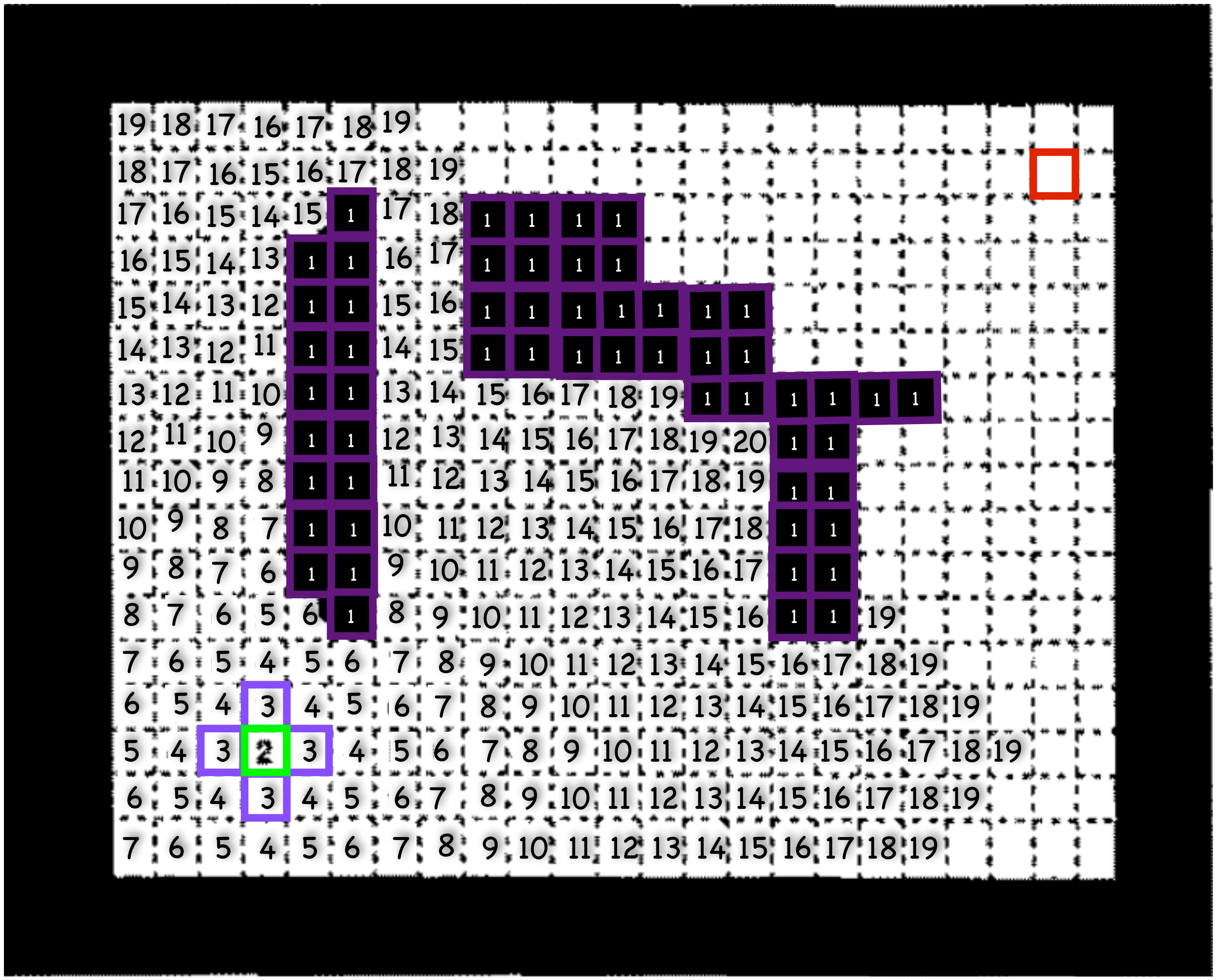
Example with Local Minima

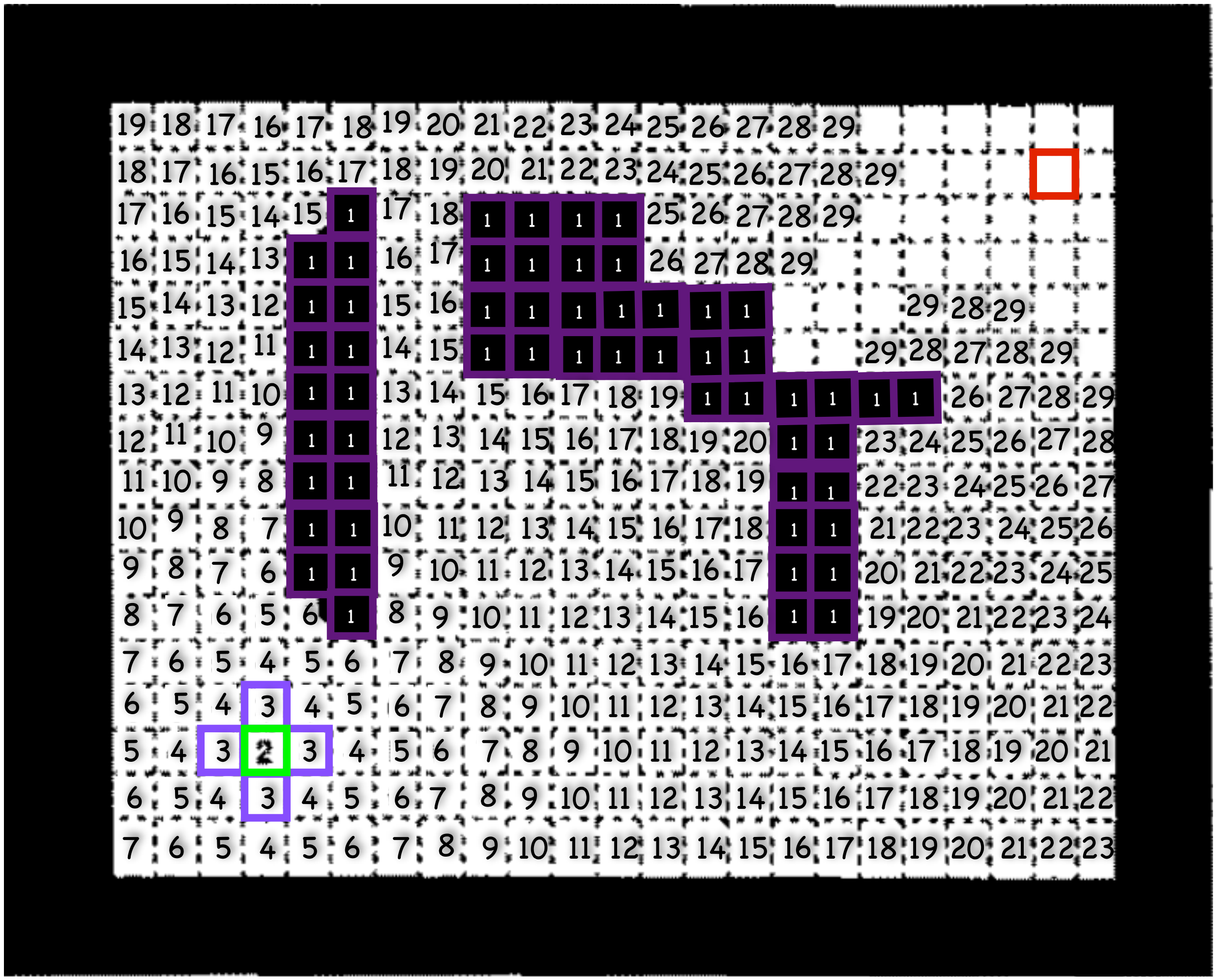
Example with Local Minima

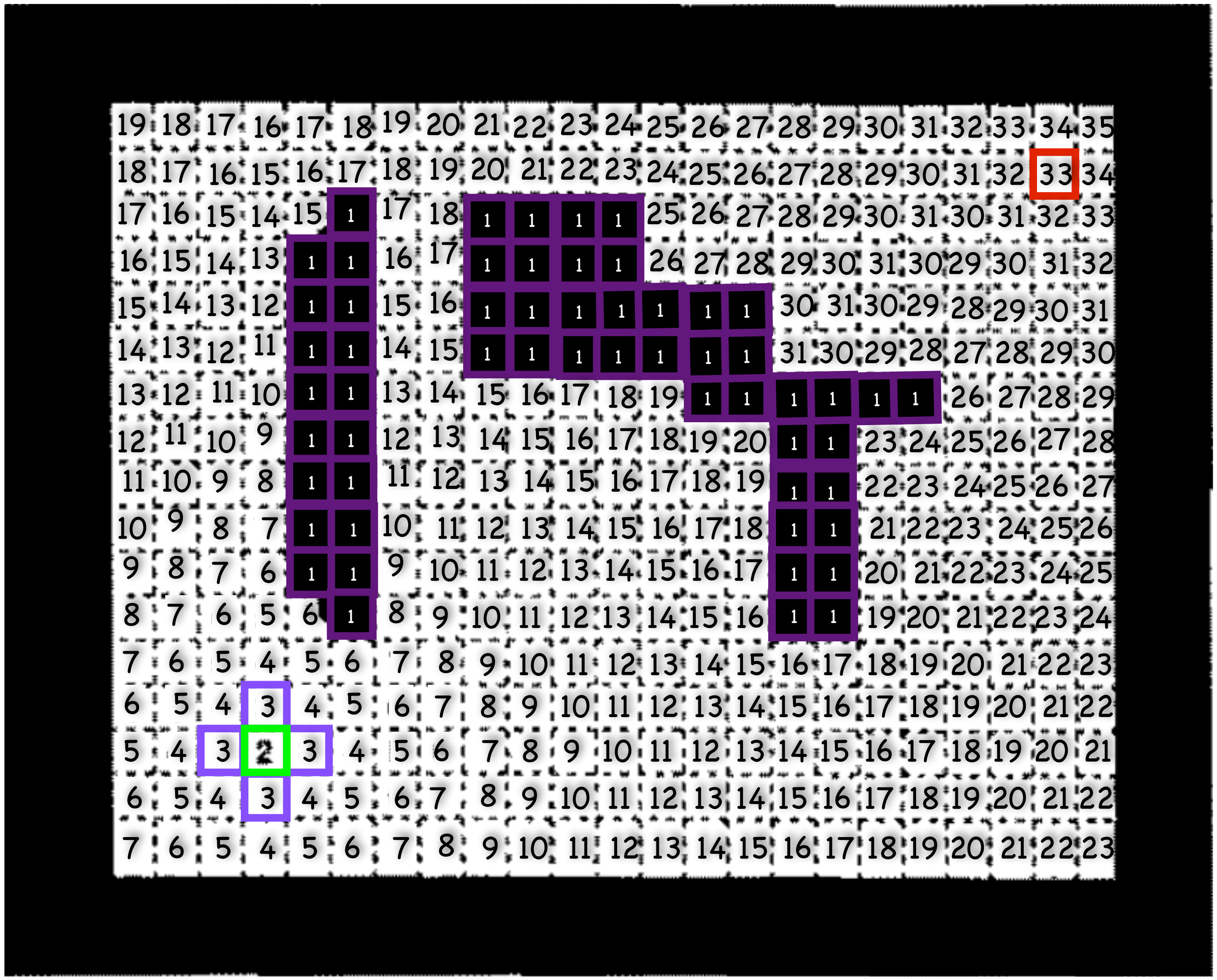


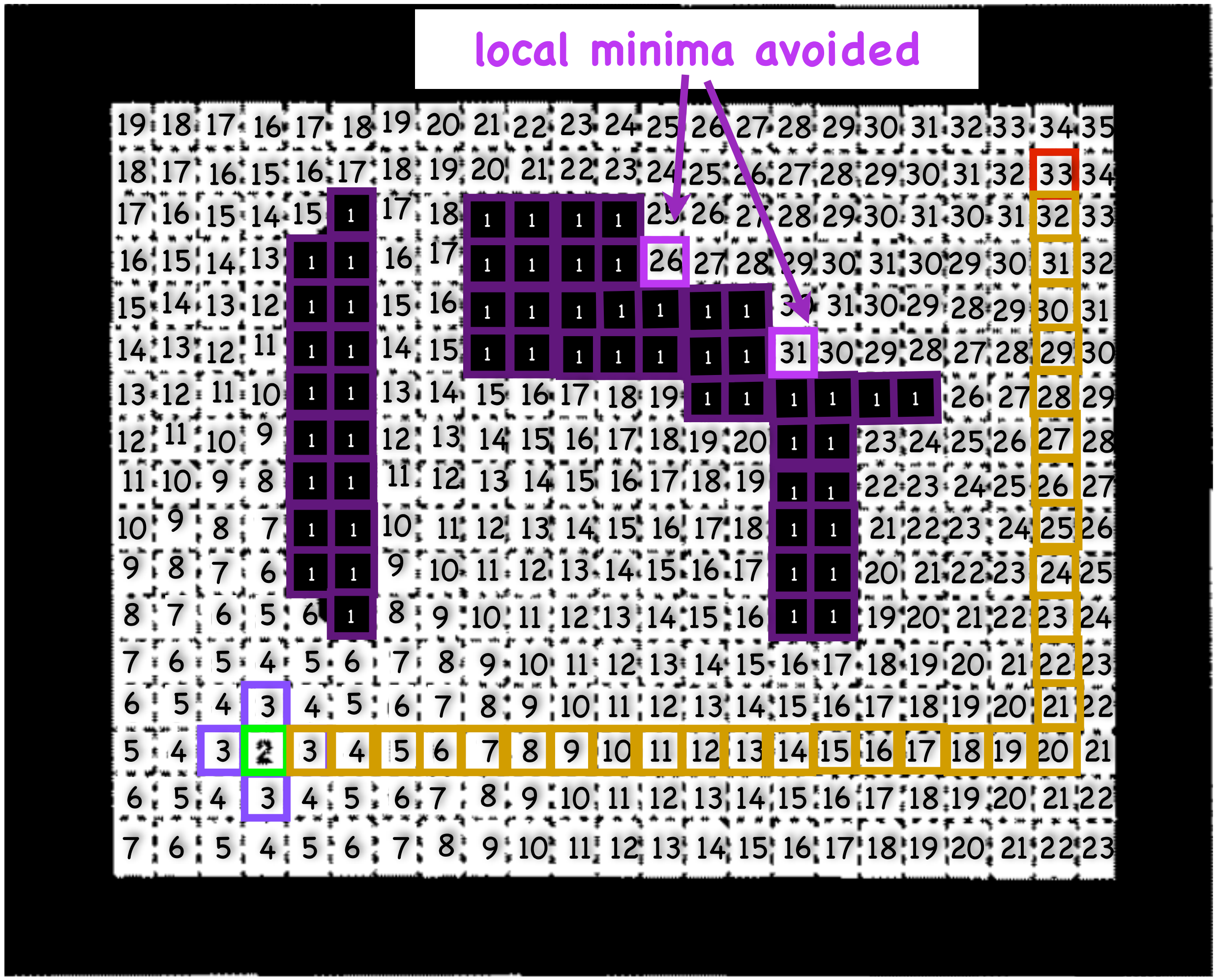




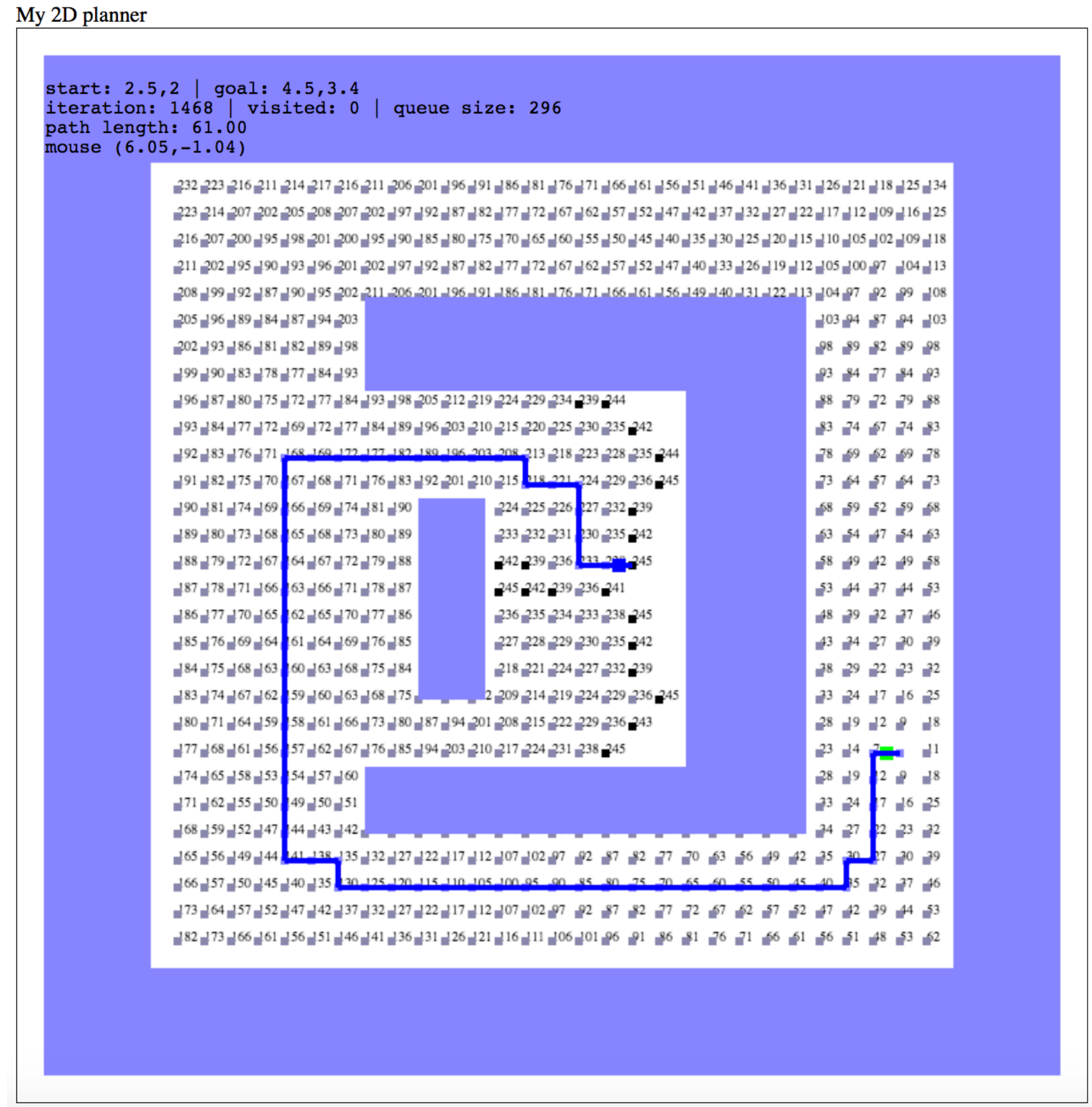

19 18 17 16 17 18 19
18 17 16 15 16 17 18 19
17 16 15 14 15 1 17 18 1 1 1 1
16 15 14 13 1 1 16 17 1 1 1 1
15 14 13 12 1 1 15 16 1 1 1 1 1 1 1
14 13 12 11 1 1 14 15 1 1 1 1 1 1 1
13 12 11 10 1 1 13 14 15 16 17 18 19 1 1 1 1 1 1
12 11 10 9 1 1 12 13 14 15 16 17 18 19 20 1 1
11 10 9 8 1 1 11 12 13 14 15 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






local minima avoided

Maithili's wavefront planner



Can we extend potential fields for arm navigation?

Potential Fields for Robot Arm

- Define endeffector goal as the attractive potential with cone potential
- Define repulsive potentials wrt. collision objects
    - Select points on robot links with “bowl” potential from nearest object
- Use manipulator Jacobian to transform potential at each point into velocites at robot joints
- Weighted sum of transformed velocities to generate control

Navigation Recap

Navigation 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

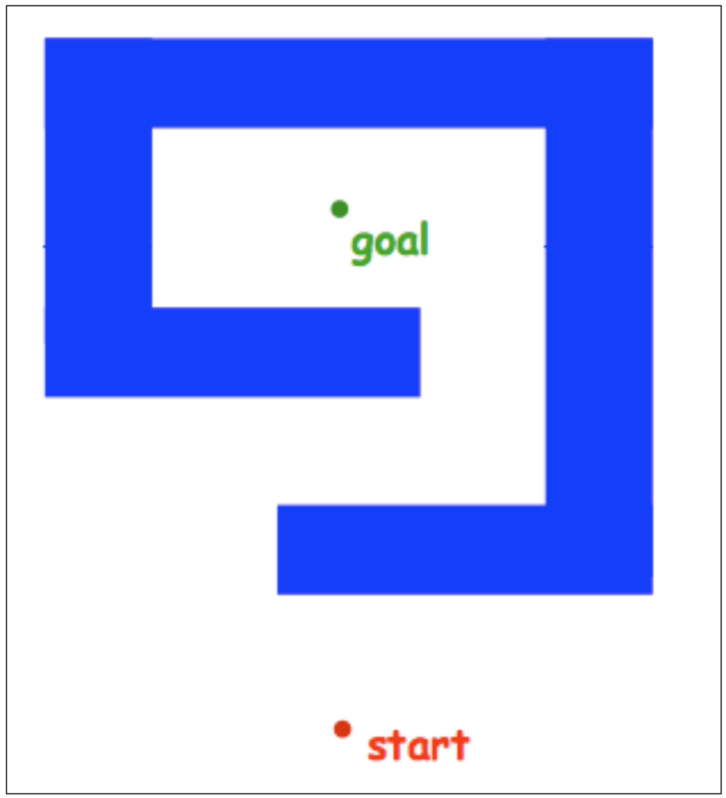
Grid Search/Wavefront
- Complete
- General C-spaces
- Limited dimensionality

Random walk
- Will find path eventually

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

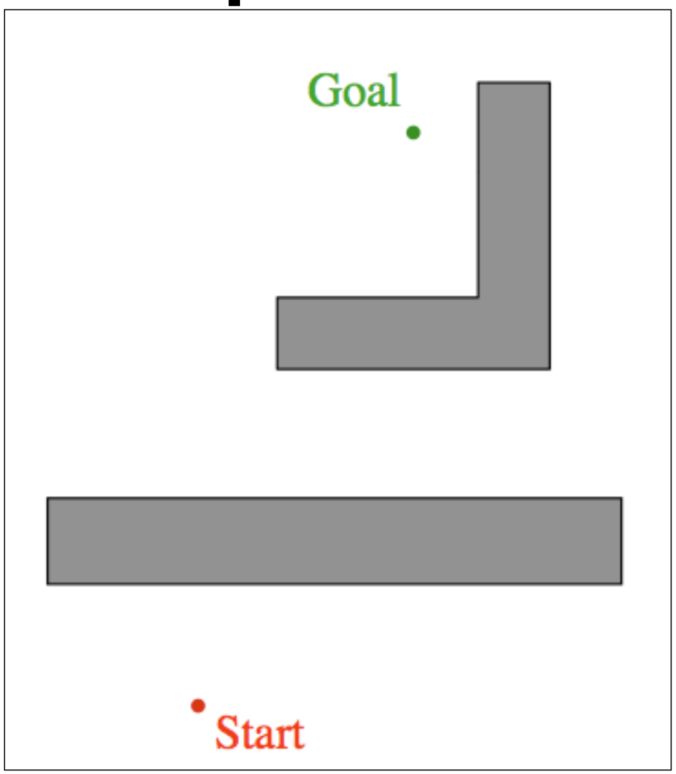
Compare: BugX examples

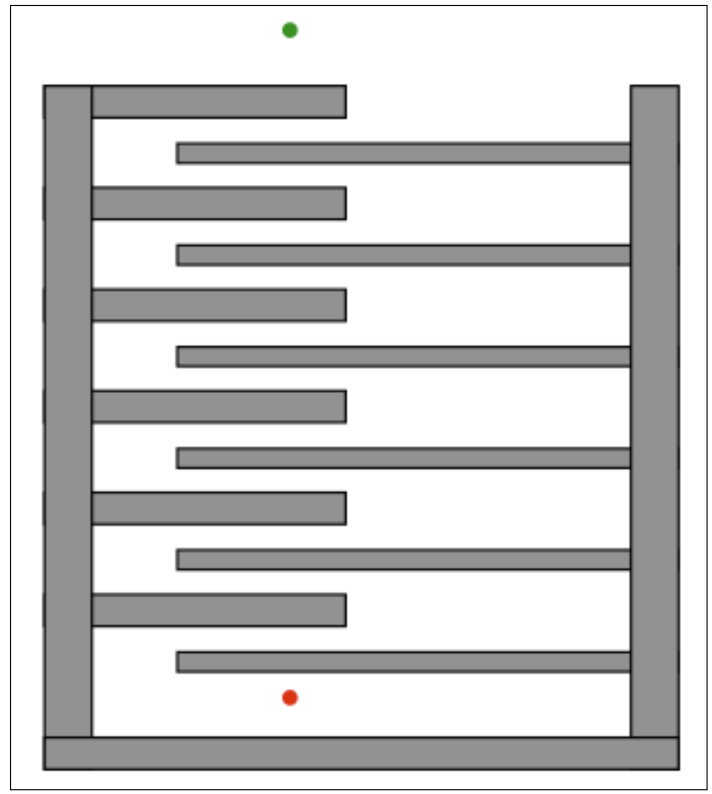
Describe likely paths from potential field and wavefront planners



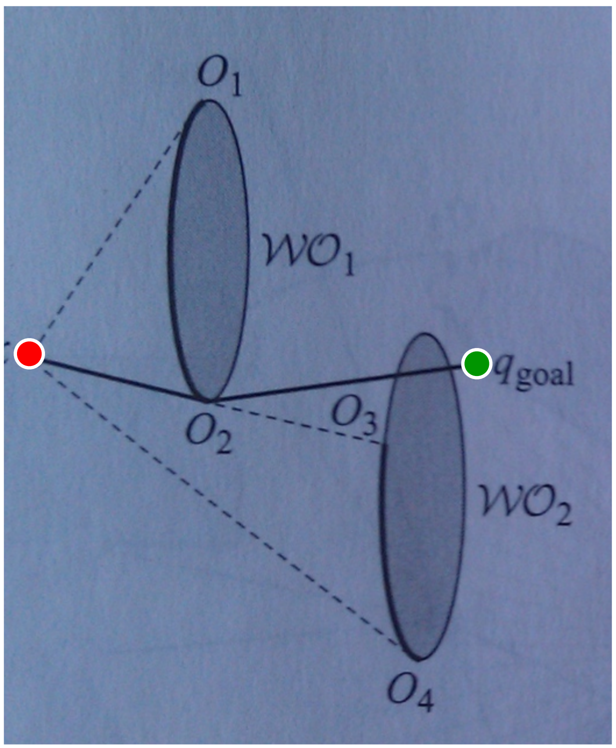
Compare: BugX examples

Describe likely paths from potential field and wavefront planners

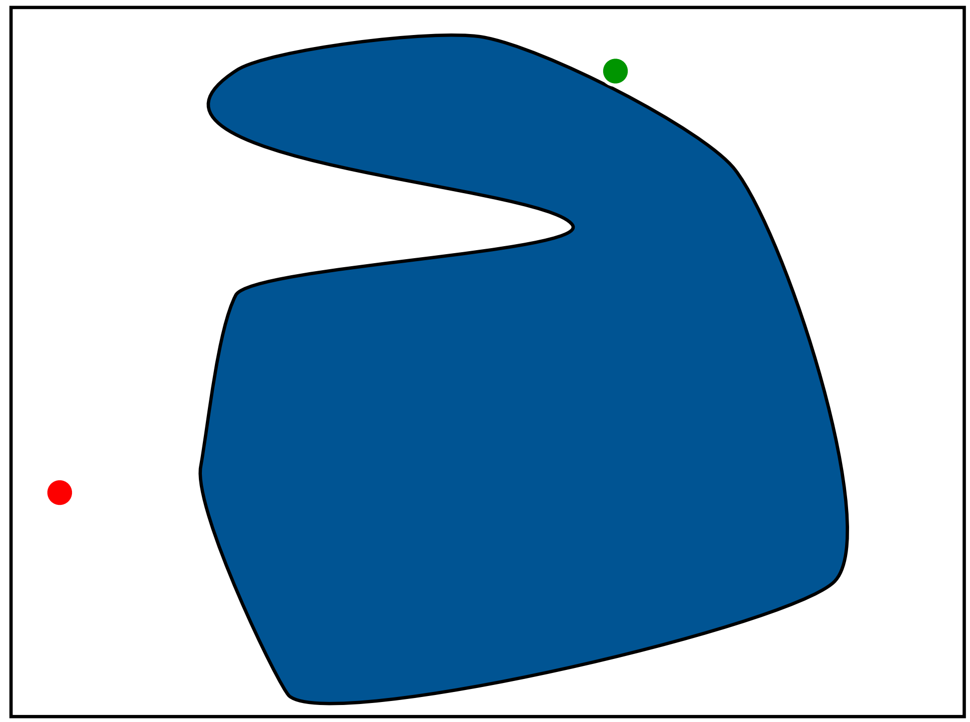




Compare: BugX examples

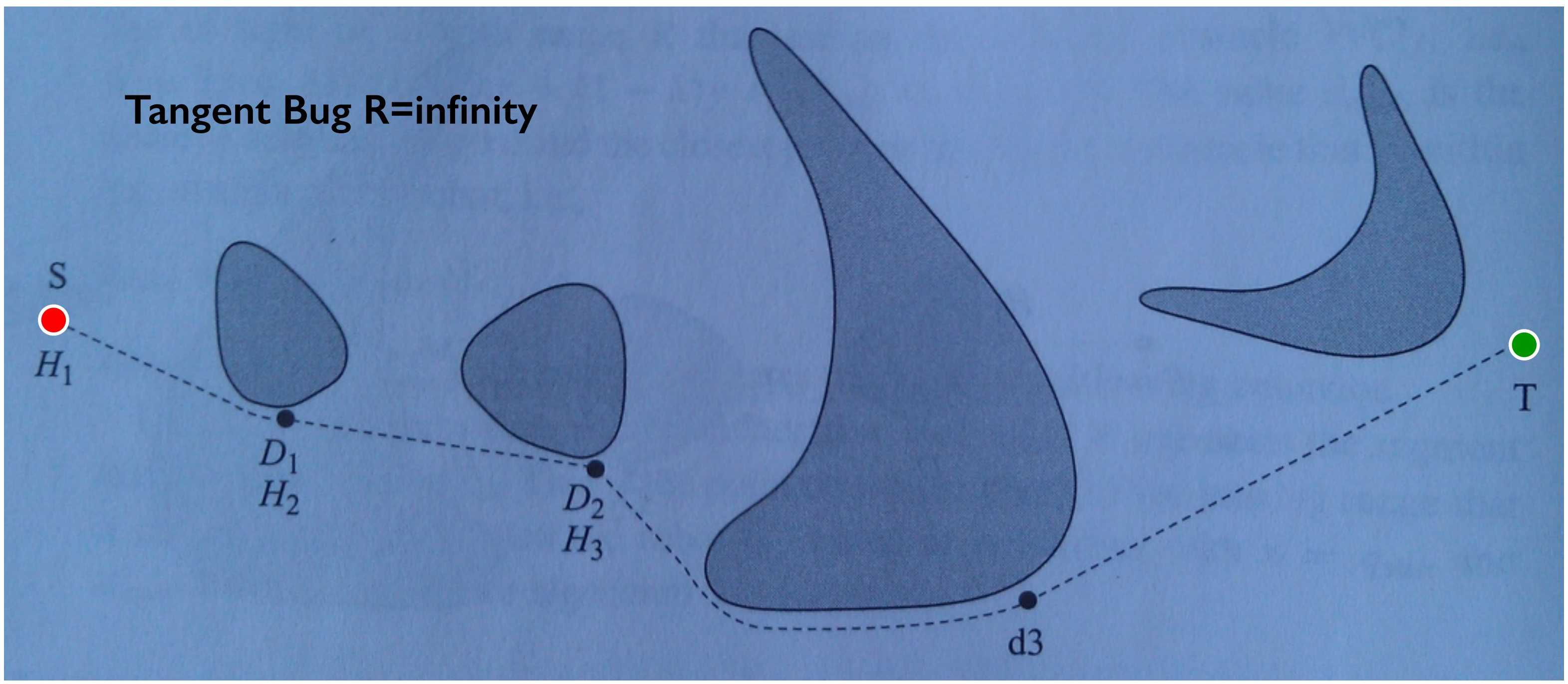


Describe likely paths from potential field and wavefront planners



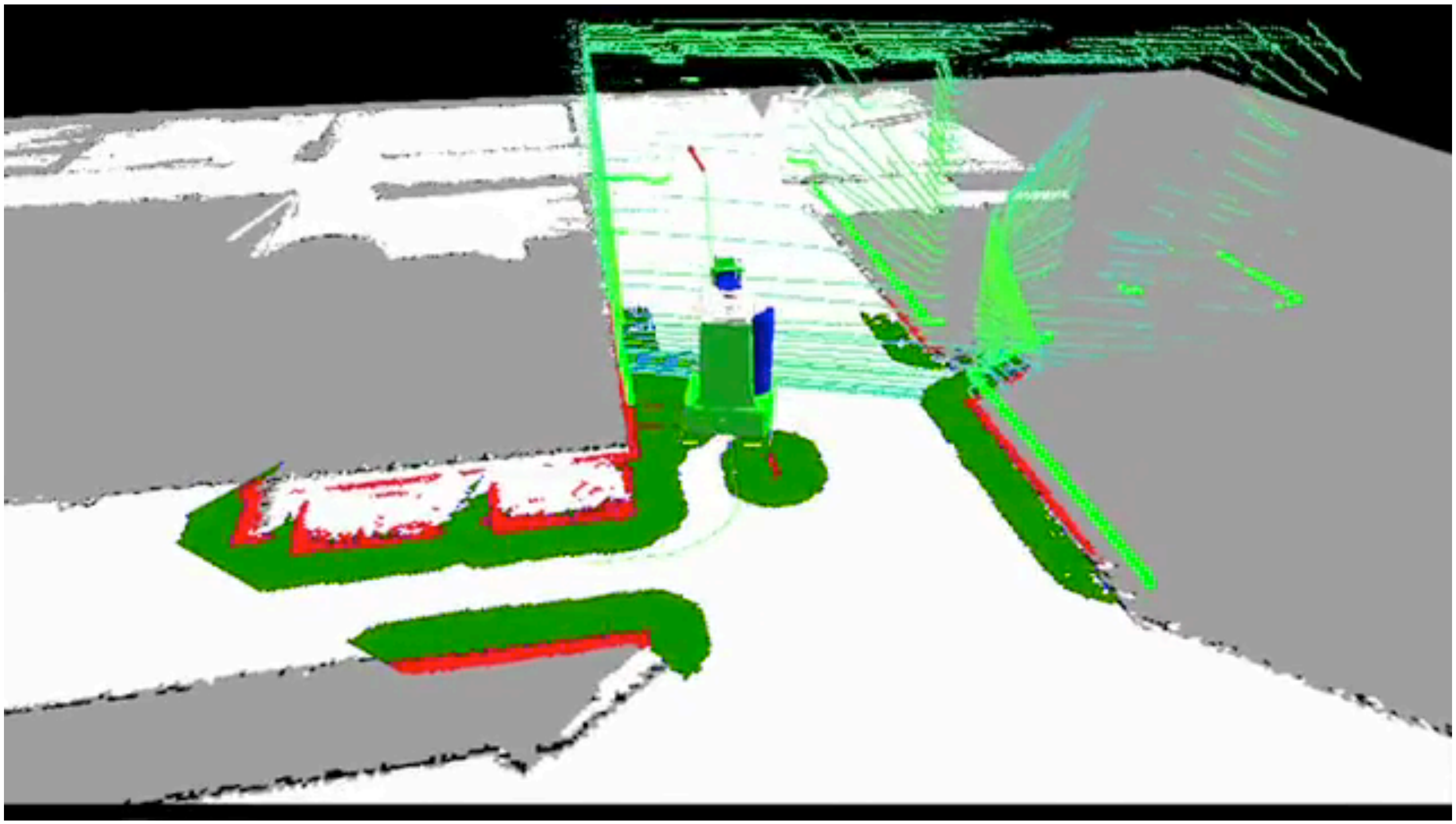
Compare: BugX examples

Describe likely paths from potential field and wavefront planners



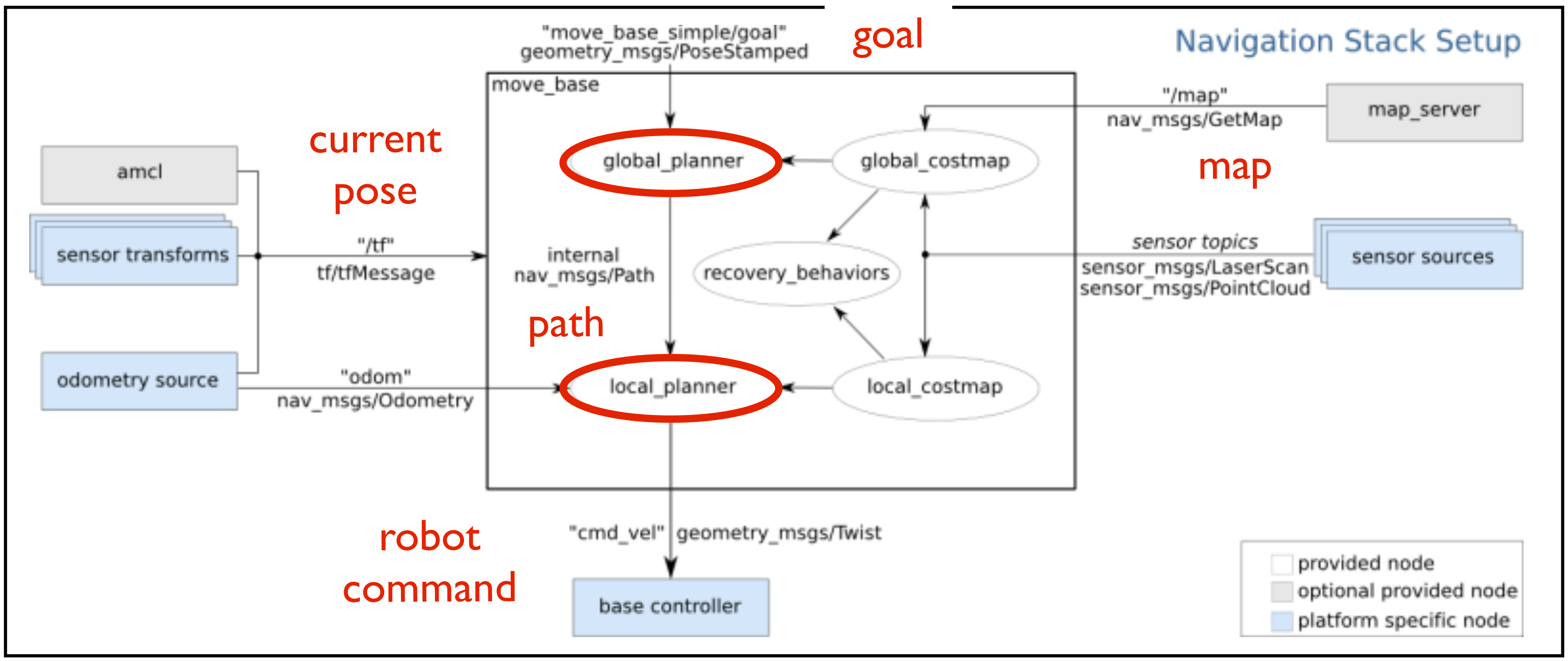
Navigation in ROS

ROS Navigation Stack



http://www.ros.org/wiki/navigation

move_base



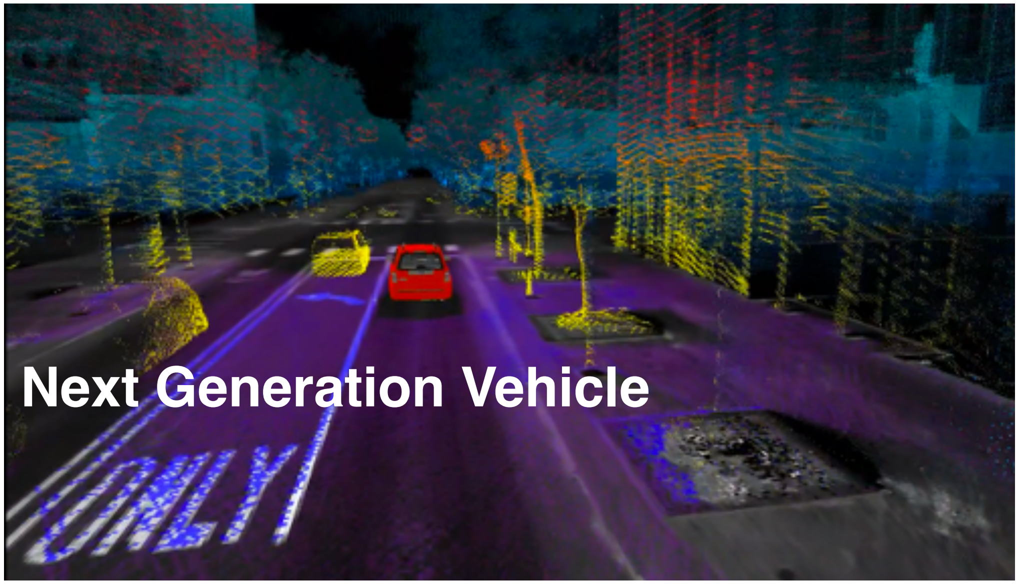
Core of nav_stack for planning and motion control

Costmap in ROS

- Treat as a Minkowski sum over the given map

http://www.ros.org/wiki/costmap_2d

Navigation in LCM

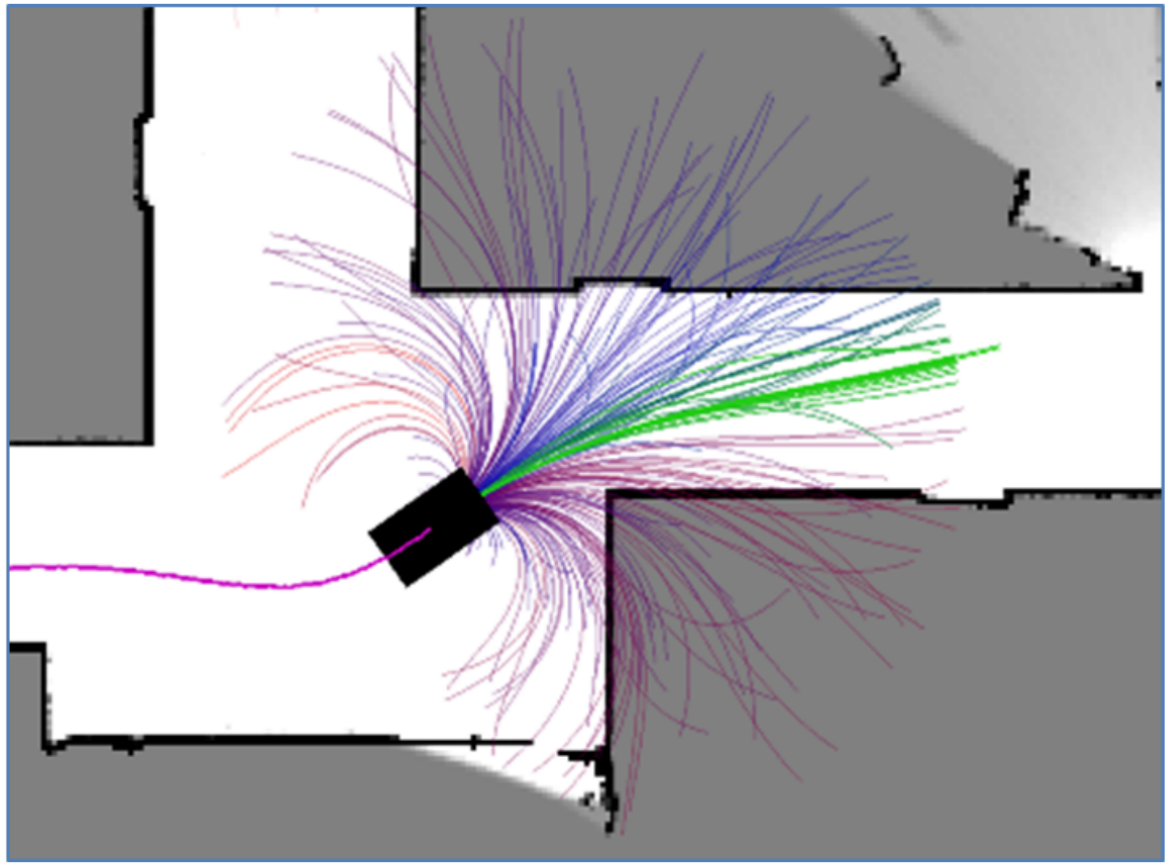

Next Generation Vehicle
ONLY


BotLab EECS 467 / ROB 550


MagicBots

[Park, Johnson, Kuipers 2012]

Vulcan Navigation



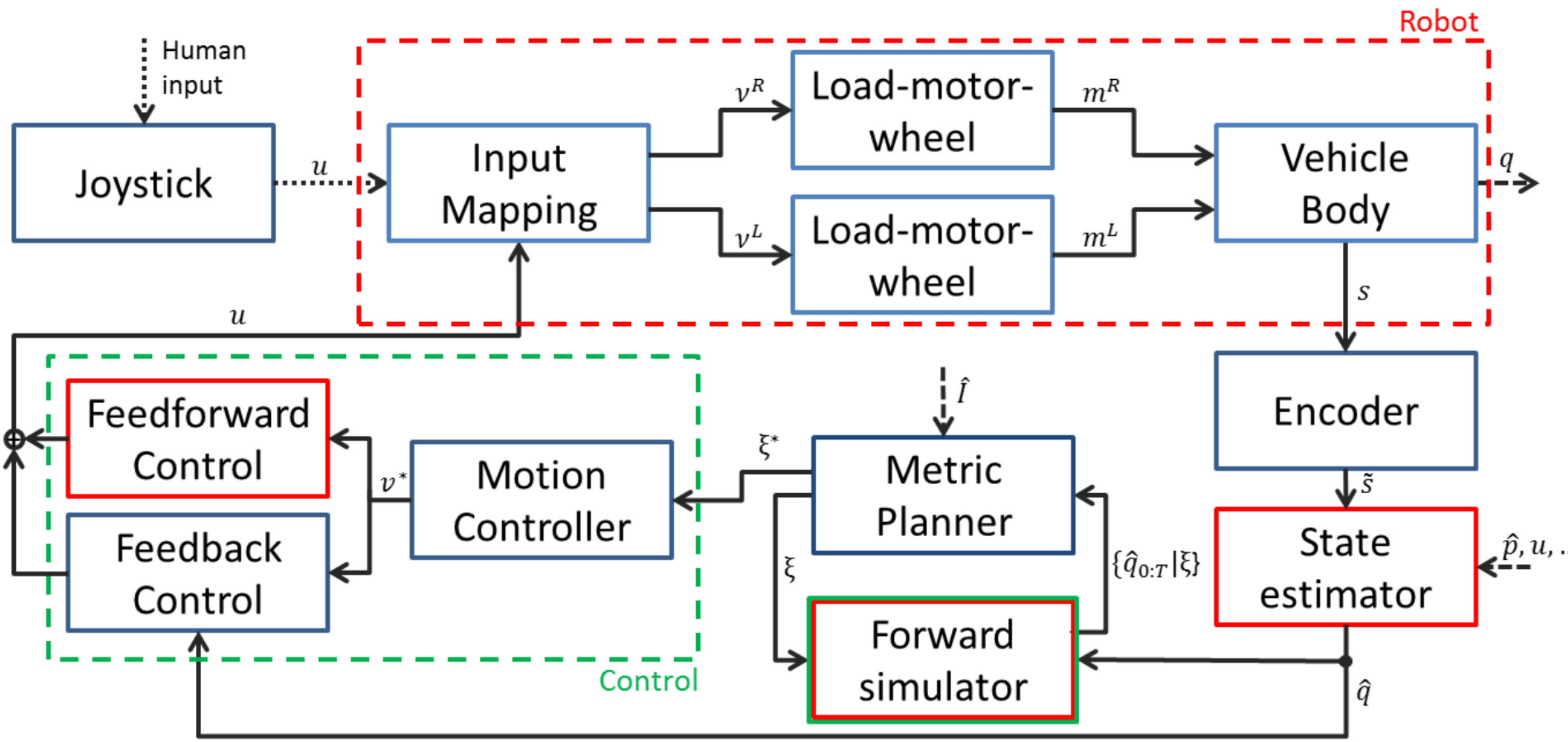
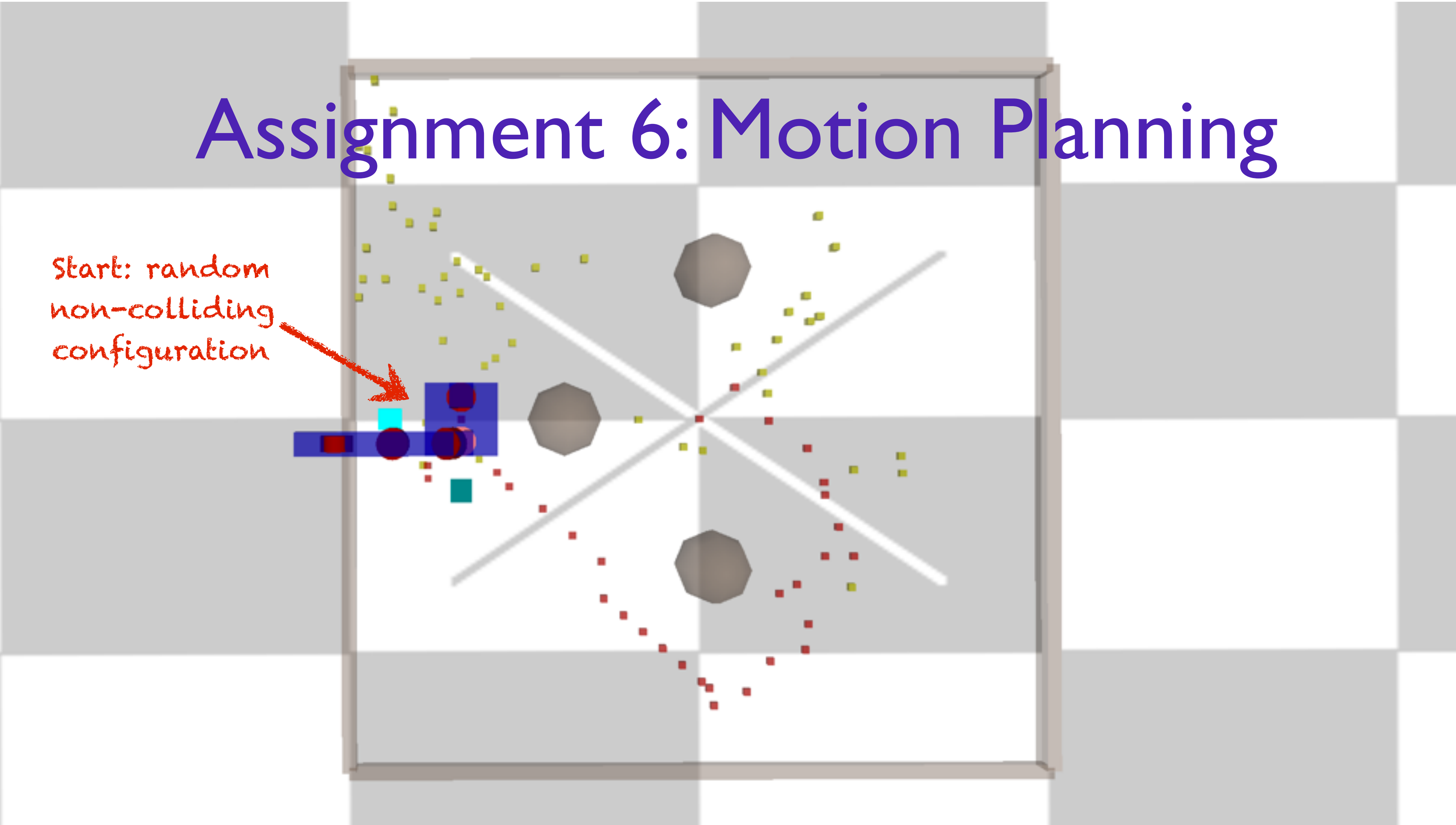


Figure A.1: Vulcan system diagram.

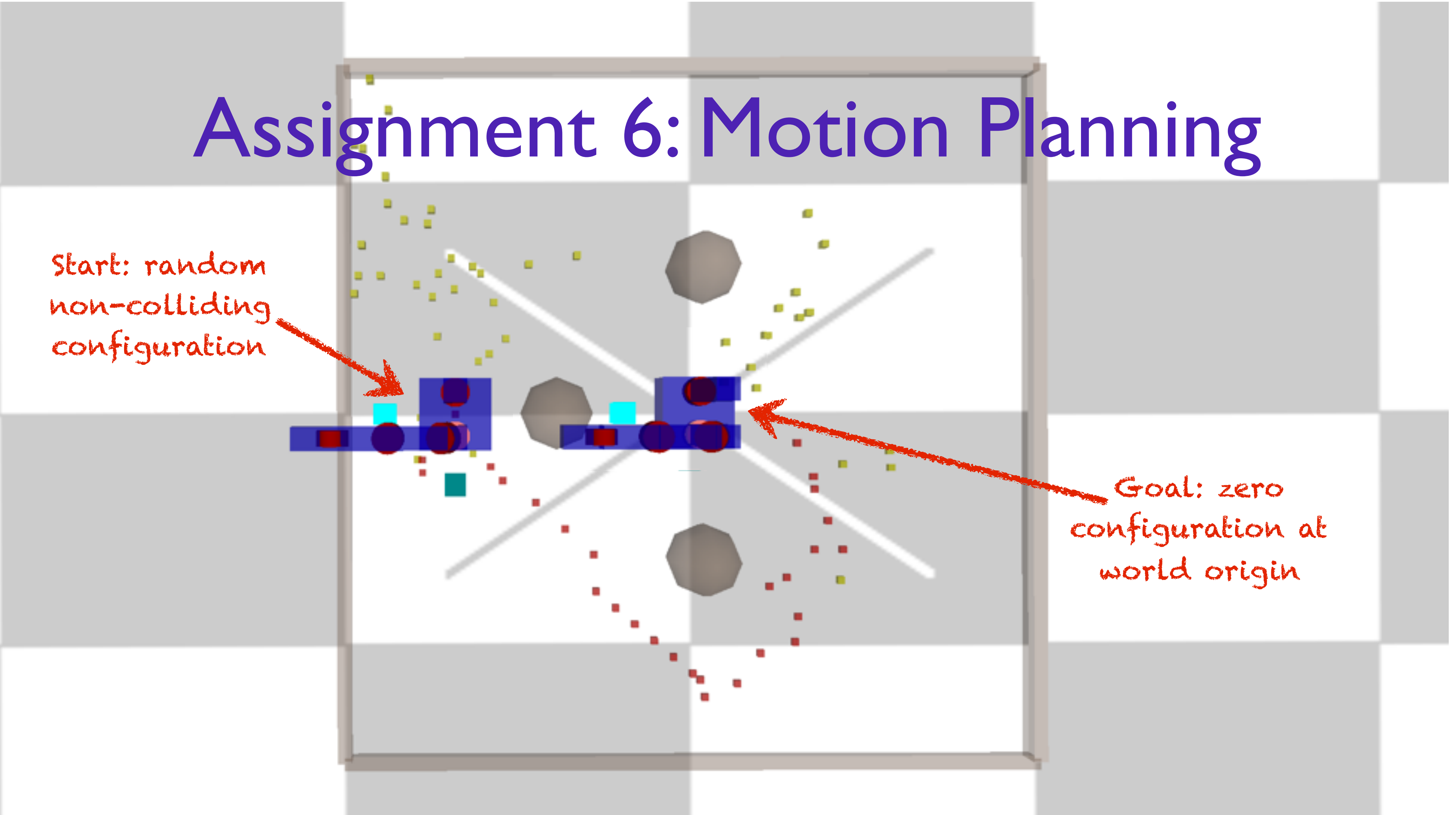
Assignment 6: Motion Planning

- Generate a collision free motion plan to the world origin and zero joint angle configuration

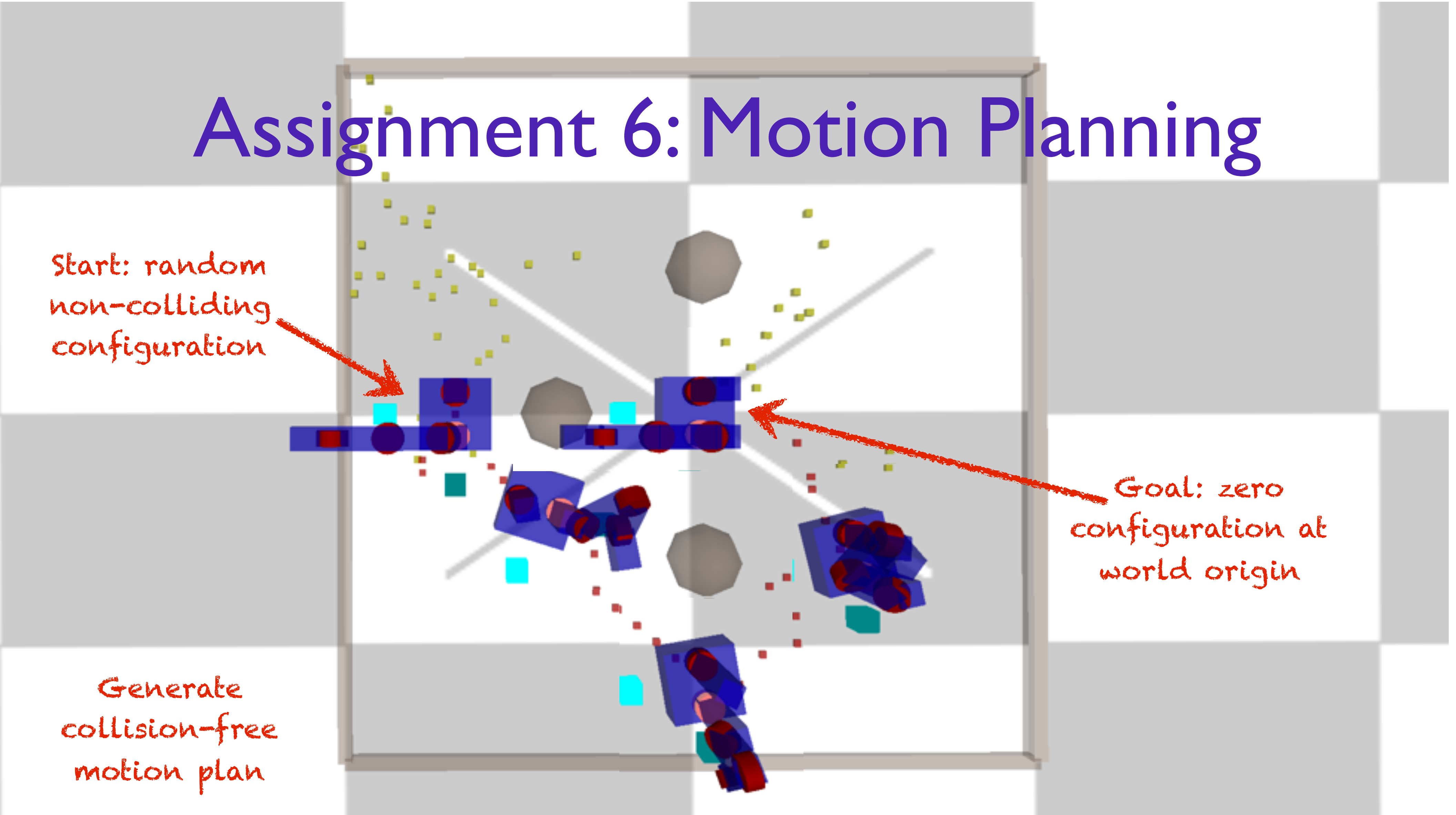
Assignment 6: Motion Planning

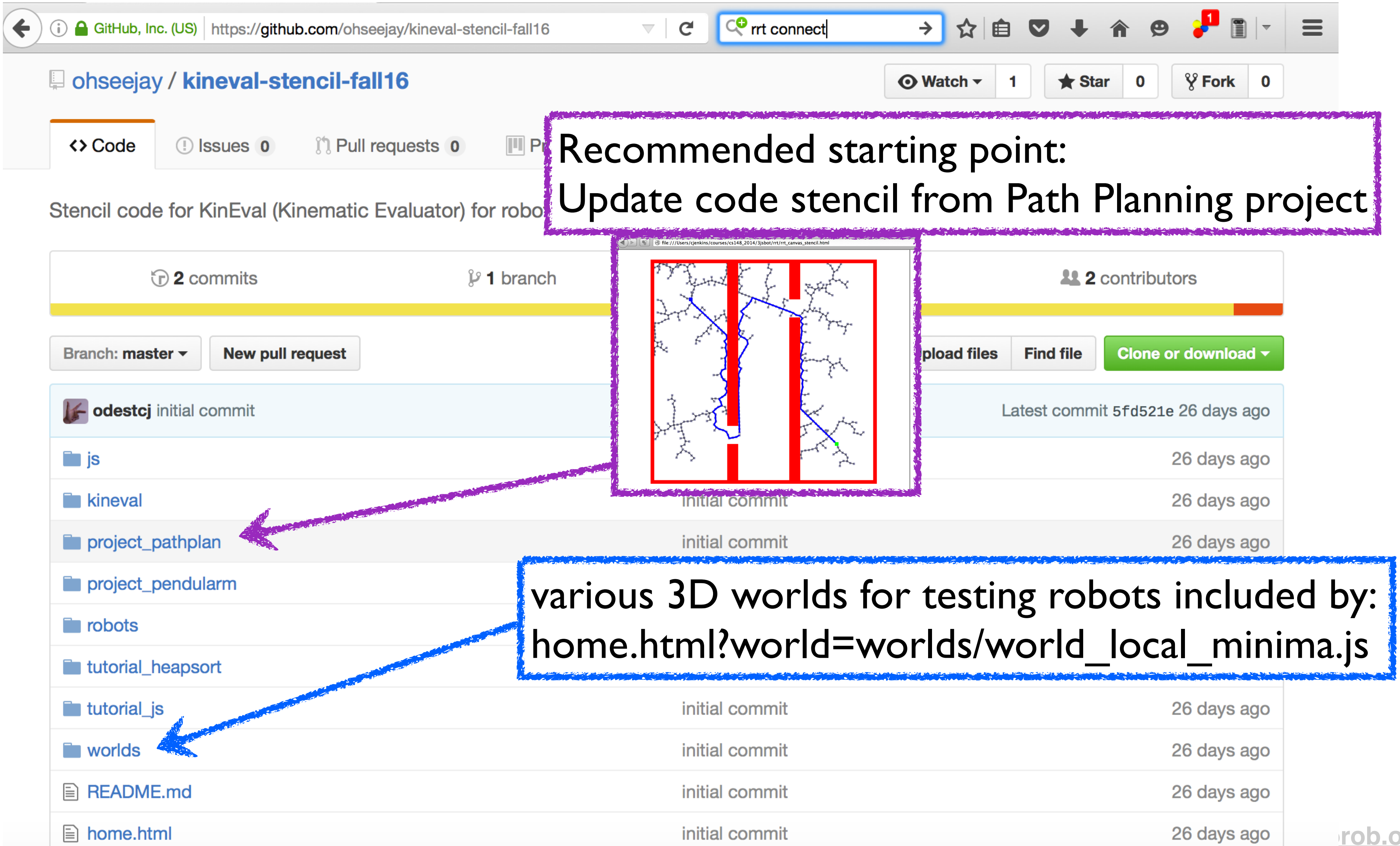


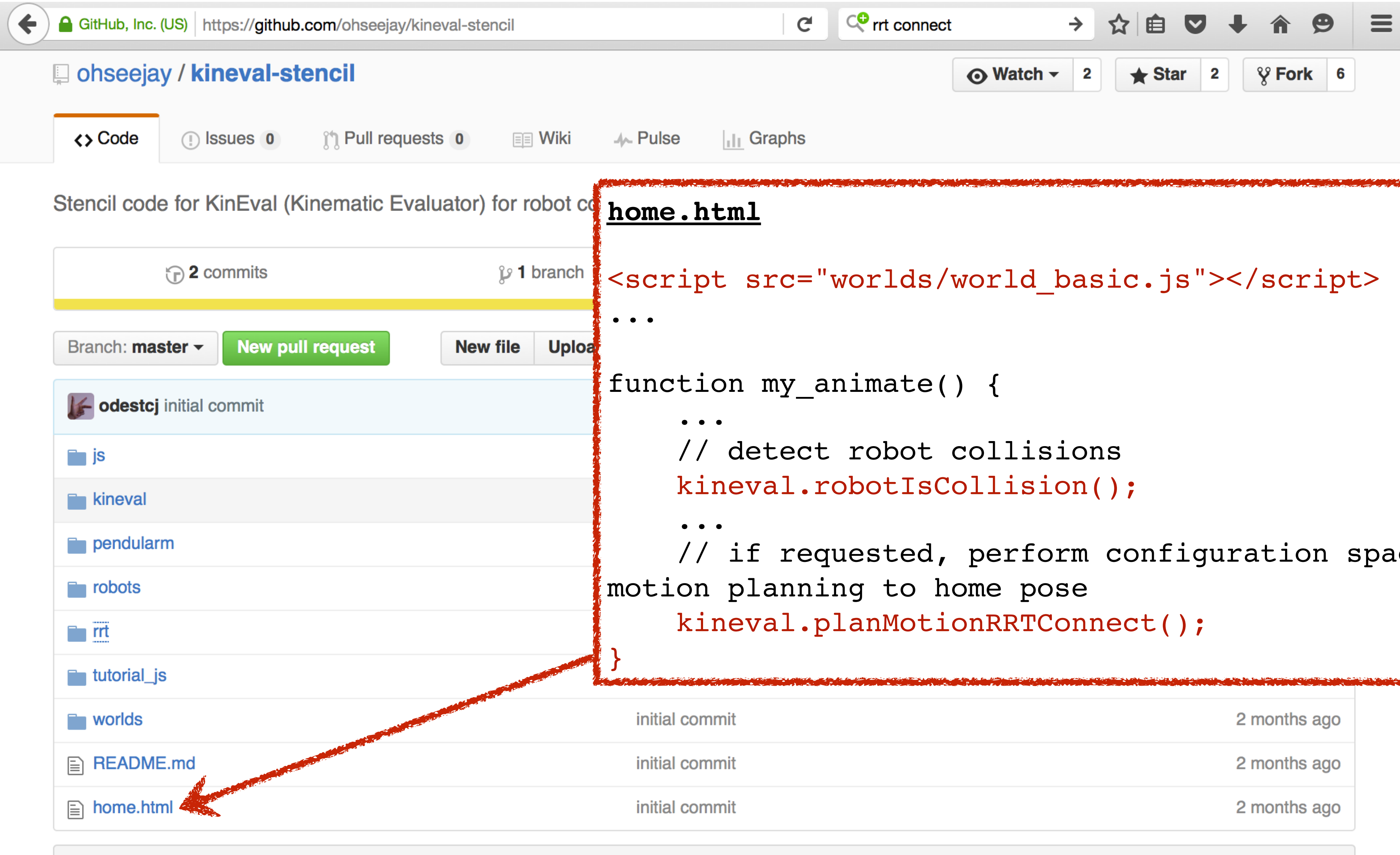
Assignment 6: Motion Planning



Assignment 6: Motion Planning




GitHub, Inc. (US) https://github.com/ohseejay/kineval-stencil-fall16
rrt connect
ohseejay / kineval-stencil-fall16
Watch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Recommended starting point:
Update code stencil from Path Planning project
Stencil code for KinEval (Kinematic Evaluator) for robo
2 commits
1 branch
2 contributors
Branch: master
New pull request
pload files
Find file
Clone or download
odestcj initial commit
Latest commit 5fd521e 26 days ago
js
26 days ago
kineval
initial commit
26 days ago
project_pathplan
initial commit
26 days ago
project_pendularm
robots
tutorial_heapsort
various 3D worlds for testing robots included by:
home.html?world=worlds/world_local_minima.js
tutorial_js
initial commit
26 days ago
worlds
initial commit
26 days ago
README.md
initial commit
26 days ago
home.html
initial commit
26 days ago



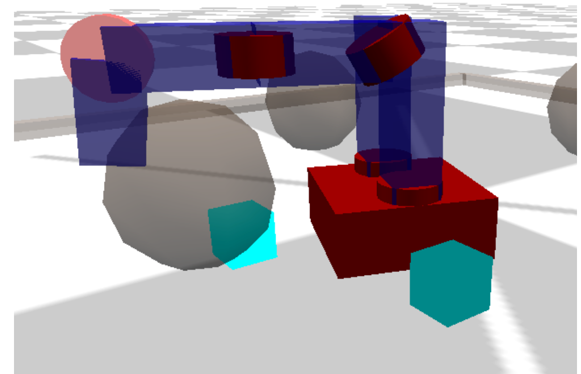
home.html

```
<script src="worlds/world_basic.js"></script>
...

function my_animate() {
    ...
    // detect robot collisions
    kineval.robotIsCollision();
    ...
    // if requested, perform configuration space
motion planning to home pose
    kineval.planMotionRRTConnect();
}
```

world file can be alternatively loaded by a script tag

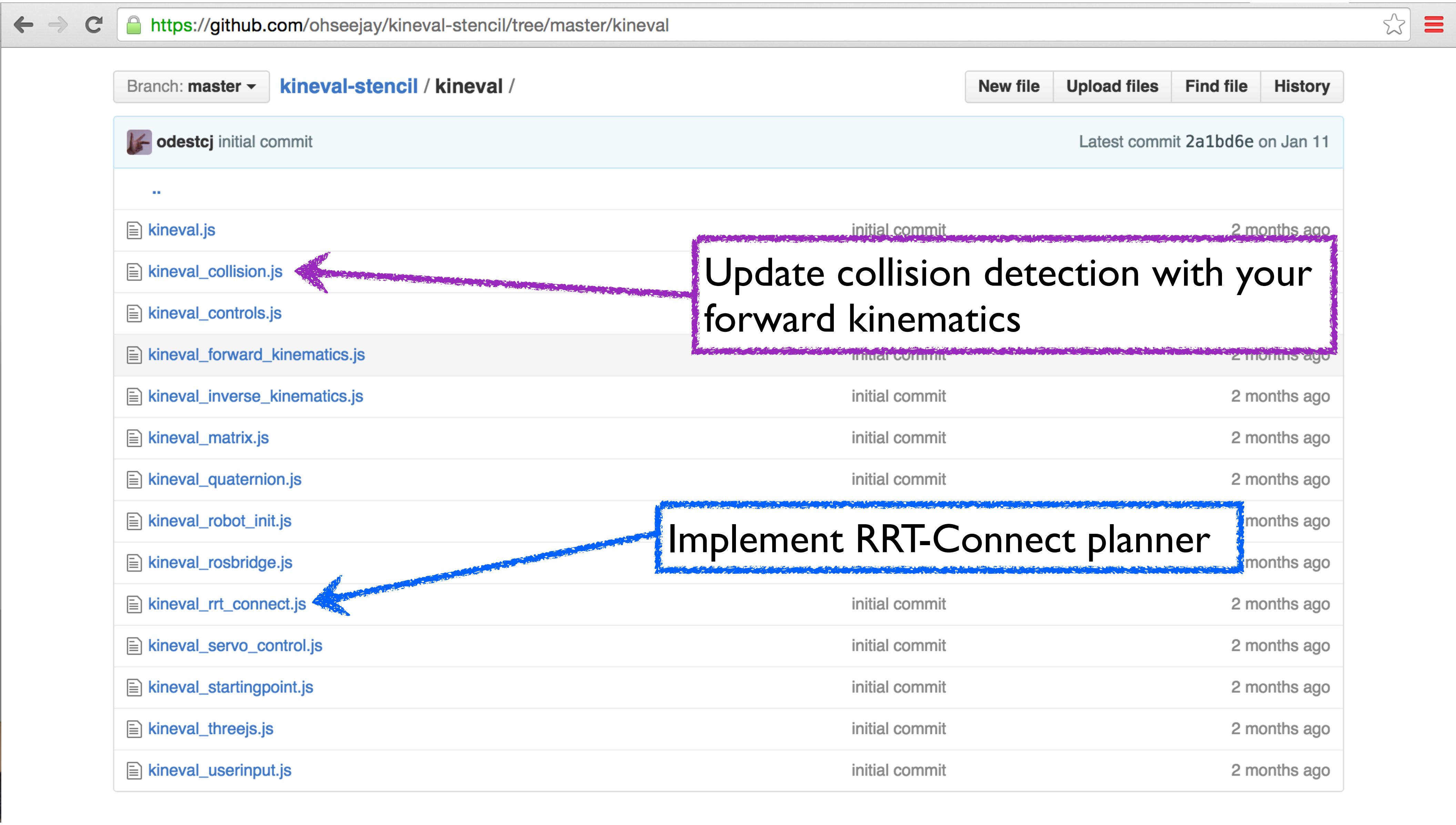
home.html

```
<script src="worlds/world_basic.js"></script>
...

function my_animate() {
    ...
    // detect robot collisions
    kineval.robotIsCollision();
    ...
    // if requested, perform configuration space motion planning to home pose
    kineval.planMotionRRTConnect();

}
```



detect if current configuration is in collision (colliding link turns red)

iterate motion planner


https://github.com/ohseejay/kineval-stencil/tree/master/kineval
Branch: master
kineval-stencil / kineval /
New file
Upload files
Find file
History
odestcj initial commit
Latest commit 2a1bd6e on Jan 11
..
kineval.js
kineval_collision.js
kineval_controls.js
kineval_forward_kinematics.js
kineval_inverse_kinematics.js
kineval_matrix.js
kineval_quaternion.js
kineval_robot_init.js
kineval_rosbridge.js
kineval_rrt_connect.js
kineval_servo_control.js
kineval_startingpoint.js
kineval_threejs.js
kineval_userinput.js
initial commit
2 months ago
Update collision detection with your forward kinematics
Implement RRT-Connect planner

input: q (robot configuration)
output: false (for no collision)
or name of link in collision

kineval_collision.js

```
kineval.poseIsCollision = function robot_collision_test(q) {
    // perform collision test of robot geometry against planning world

    // test base origin (not extents) against world boundary extents
    if ((q[0]<robot_boundary[0][0])||(q[0]>robot_boundary[1][0])||
        (q[2]<robot_boundary[0][2])||(q[2]>robot_boundary[1][2]))
        return robot.base;

    // traverse robot kinematics to test each body for collision
    // STENCIL: implement forward kinematics for collision detection
    return robot_collision_forward_kinematics(q);

}
```

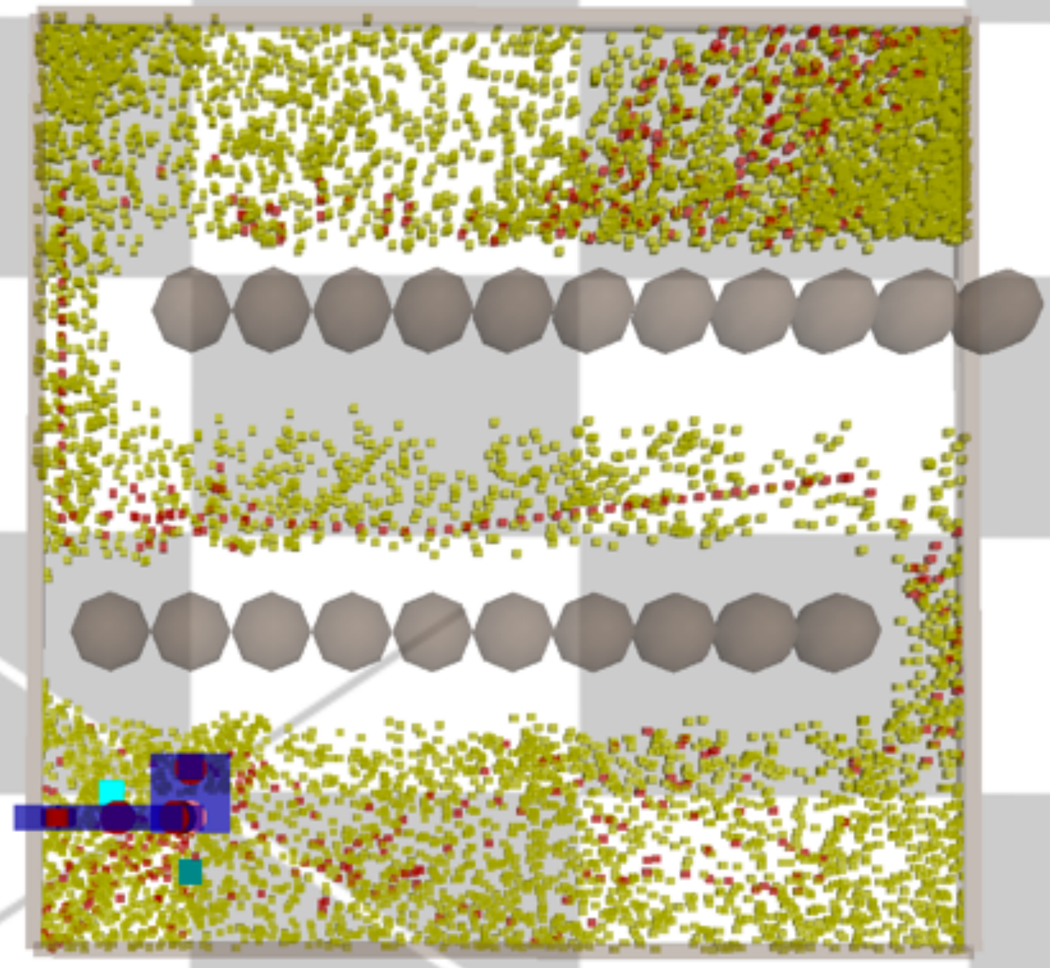
"electric fence" world boundary detection is provided

Uncommment this call; and
Implement this function to traverse forward kinematics for collisions;
Use provided link collision function for AABB test of each link

kineval_rrt_connect.js

```
kineval.robotRRTPlannerInit = function robot_rrt_planner_init() {

   // set start (q_init) and goal (q_goal) configurations

}

function robot_rrt_planner_iterate() {

    var i;
    rrt_alg = 1;  // 0: basic rrt (OPTIONAL), 1: rrt_connect (REQUIRED)

    if (rrt_iterate && (Date.now()-cur_time > 10)) {
        cur_time = Date.now();

        // implement one planning iteration here with calls to rrt_extend, etc.

        // if plan found, store configuration sequence in kineval.motion_plan
        //   kineval.motion_plan[kineval.motion_plan_traversal_index]
    }
}
```

make sure to test against all provided worlds!




Next Lecture:
Collision Detection