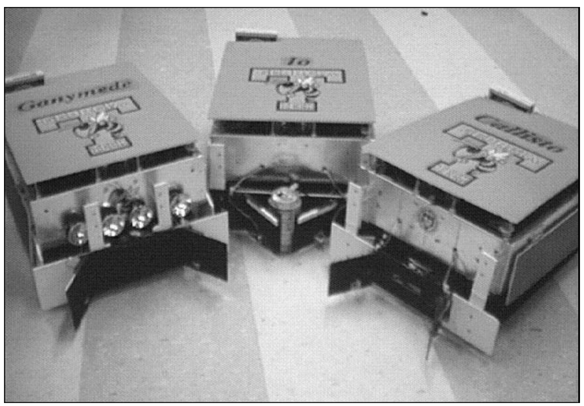
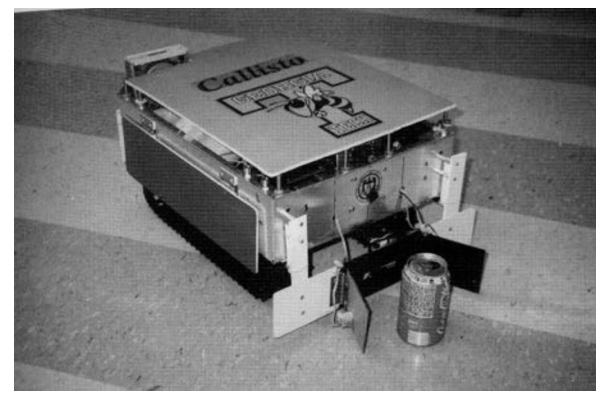
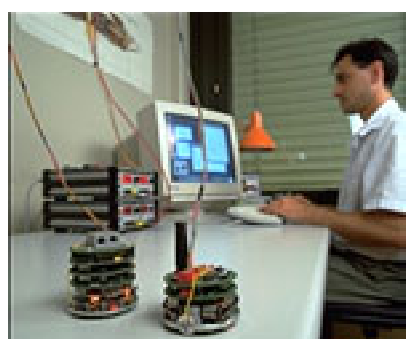
Potential Fields

October 7, 2014



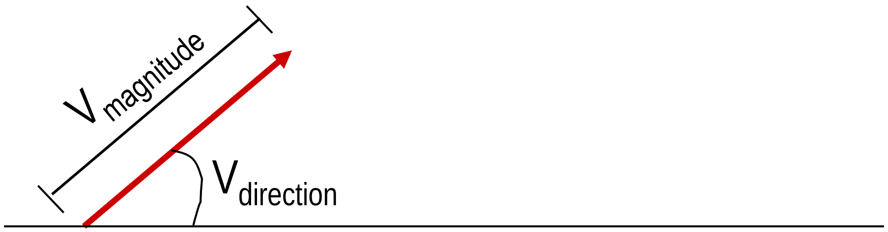




Potential Fields

Introduction to Potential Fields:

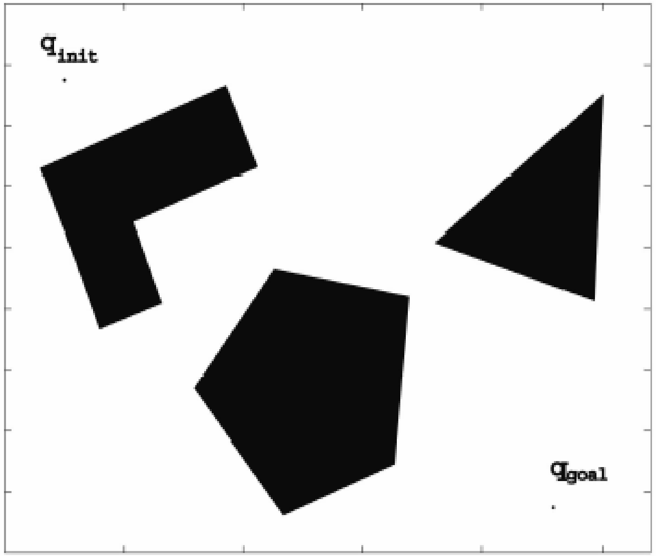
- Potential field: array (or field) of vectors representing space
- Vector **v** = (*m*,*d*)*:* consists of magnitude (*m*) and direction (*d*)
- Vector represents a force
- Typically drawn as an arrow:



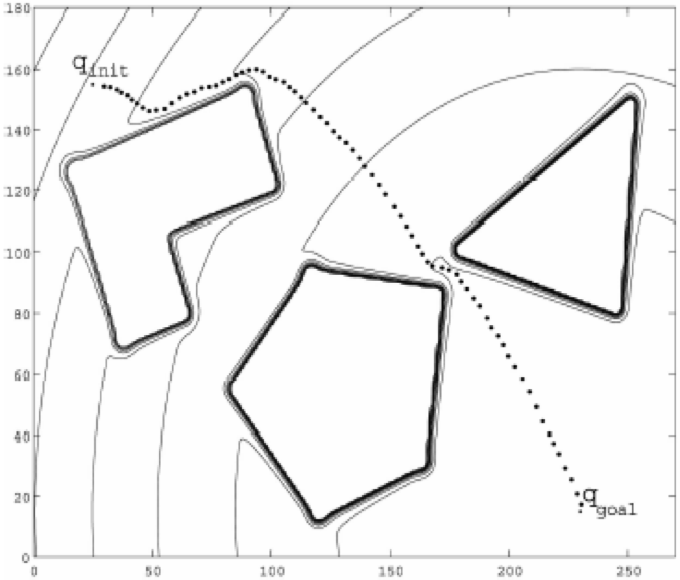
Length of arrow = *m* = magnitude

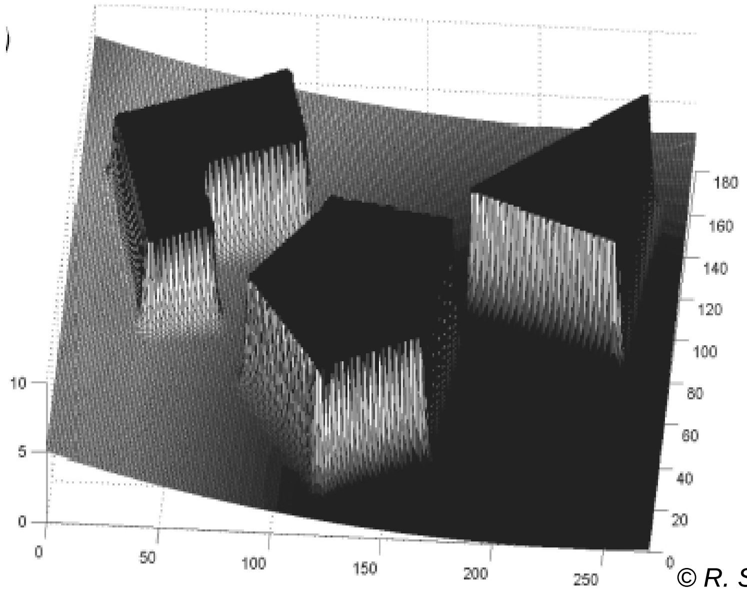
Angle of arrow = *d* = direction

Potential Field Path Planning



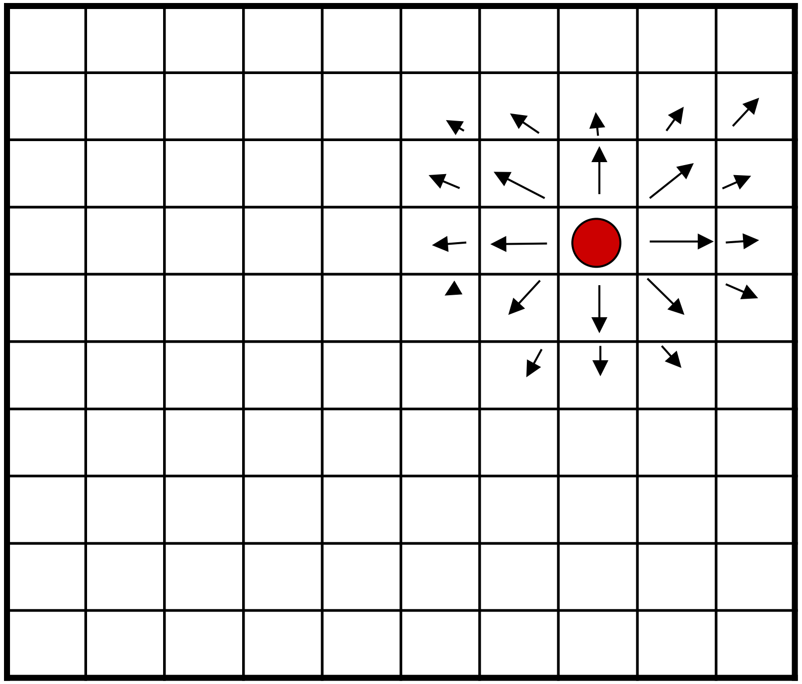
- Robot is treated as a *point under the influence* of an artificial potential field.
  - Generated robot movement is similar to a ball rolling down the hill
  - Goal generates attractive force
  - Obstacles are repulsive forces
- Note that this is more than just path planning: it is also a control law for the robot's motion





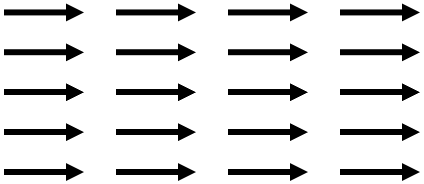
Potential Fields – More detail

- Vector space is 2D world, like bird's eye view of map
- Map divided into squares, creating (x,y) grid
- Each element represents square of space
- Perceivable objects in world exert a force field on surrounding space

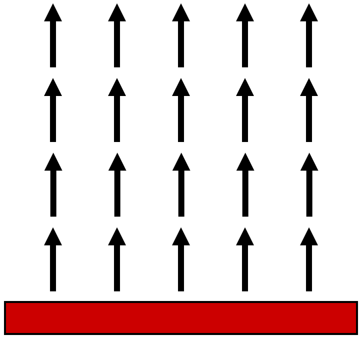


Some Primitive Types of Potential Fields

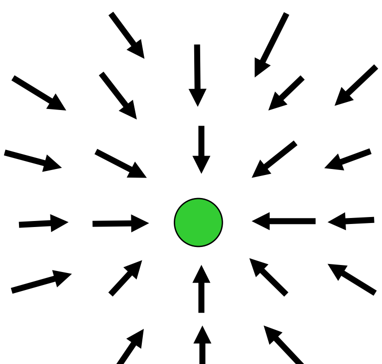
Uniform



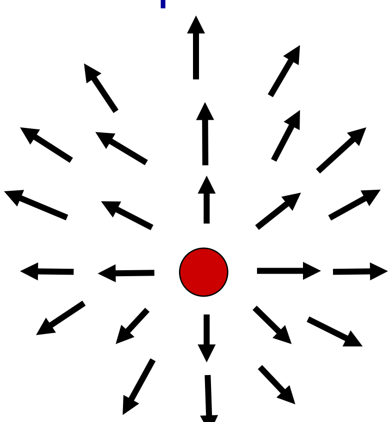
Perpendicular



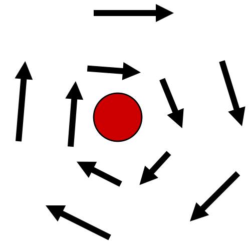
Attraction



Repulsion



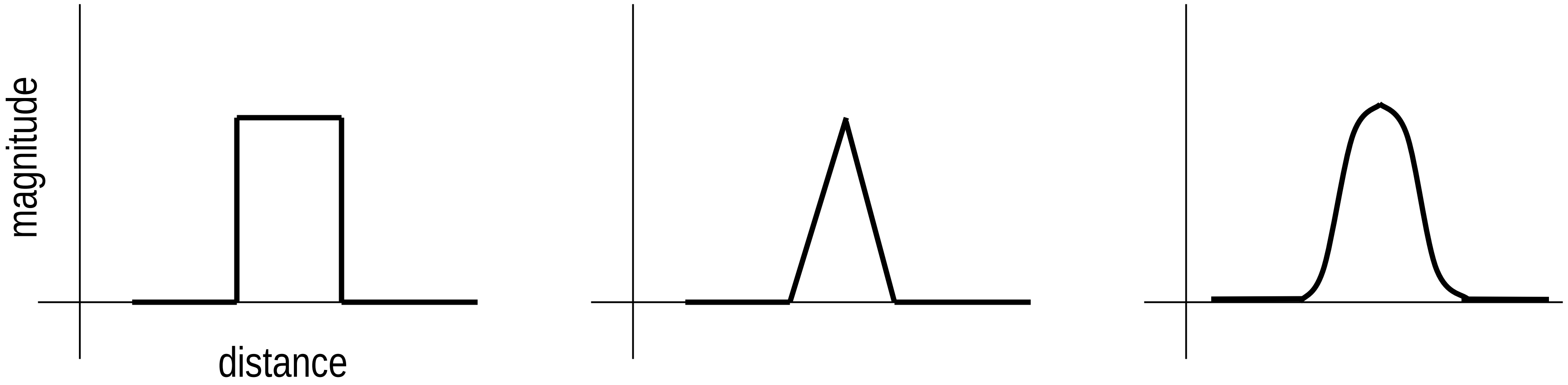
Tangential



Magnitude Profiles

- Change in velocity in different parts of the field

(See your text for 3D versions of these profiles)



Constant

Linear Dropoff

Exponential Dropoff

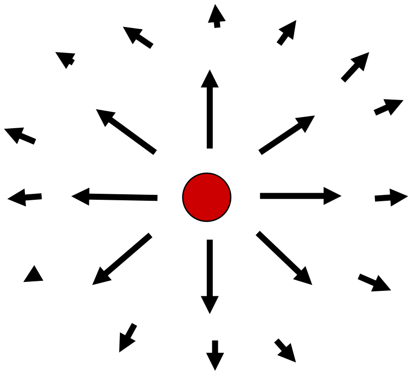
➔ *Field closest to an attractor/repellor will be stronger*

Programming a Single Potential Field

- Repulsive field with linear drop-off:

$$V_{direction} = 180^o$$

$$V_{magnitude} = \begin{cases} \frac{(D-d)}{D} & \text{for } d \leq D \\ 0 & \text{for } d > D \end{cases}$$



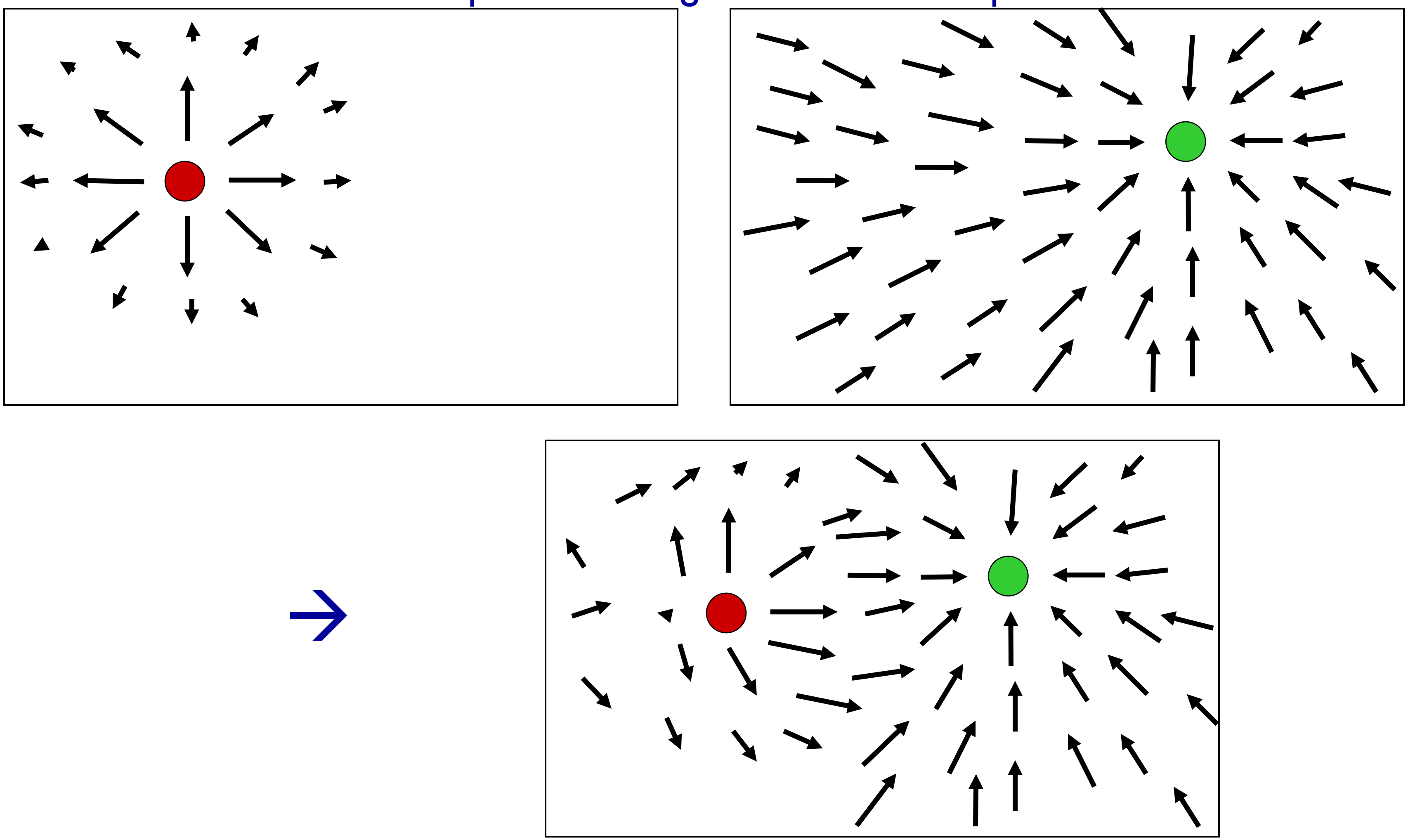
where *D* is max range of field's effect

Important Note:
Entire Field Does Not Have to Be Computed

- Only portion of field affecting robot is computed
- Robot uses functions defining potential fields at its position to calculate component vector

Combining Fields/Behaviors

- Compute each behavior’s potential field
- Sum vectors at robot’s position to get resultant output vector



Issues with Combining Potential Fields

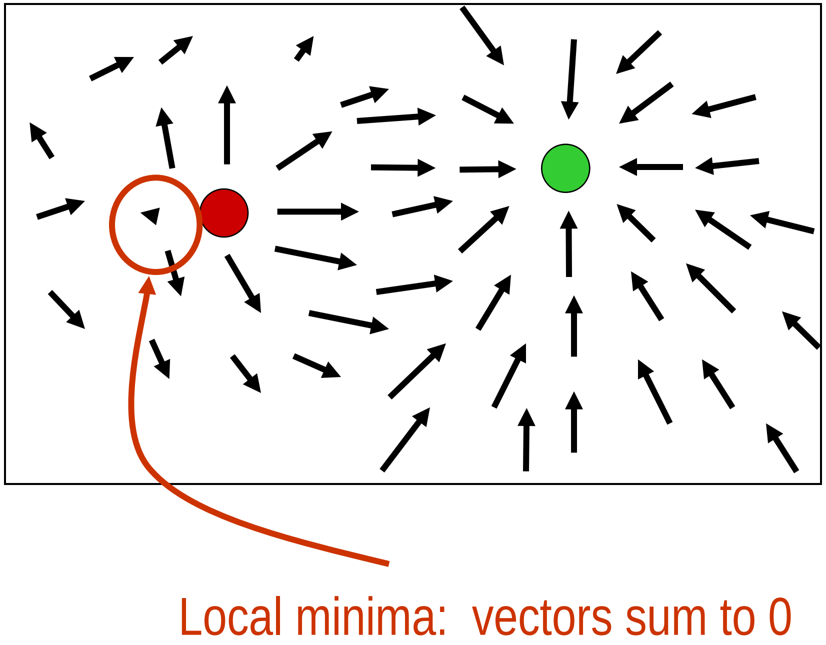
- Impact of update rates:
  - Lower update rates can lead to “jagged” paths

- Robot treated as point:
  - ➔ Expect robot to change velocity and direction instantaneously (can’t happen)

- Local minima:
  - Vectors may sum to 0.

The Problem of Local Minima

- If robot reaches local minima, it will just sit still



Solutions for Dealing with Local Minima

- Inject noise, randomness:
  - “Bumps” robot out of minima
- Include “avoid-past” behavior:
  - Remembers where robot has been and attracts the robot to other places
- Use “Navigation Templates” (NaTs):
  - The “avoid” behavior receives as input the vector summed from other behaviors
  - Gives “avoid” behavior a preferred direction
- Insert tangential fields around obstacles

Again now, with more math: Potential Field Generation

- Generation of potential field function *U(q)* for robot at point *q* :
  - attracting (goal) and repulsing (obstacle) fields
  - summing up the fields $U(q) = U_{att}(q) + U_{rep}(q)$
  - functions must be differentiable

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

- Generate artificial force field *F(q)* as the gradient of the potential field:

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

$$F(q) = F_{att}(q) + F_{rep}(q)$$
$$= -\nabla U_{att}(q) - \nabla U_{rep}(q)$$

Converting to robot control

- Set robot velocity (v_x, v_y) proportional to the force $F(q)$ generated by the field
  - the force field drives the robot to the goal
  - robot is assumed to be a point mass

Mathematical Representation: **Attractive Potential Field**

- Parabolic function representing the Euclidean distance $\|q - q_{goal}\|$ to the goal:

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

where k_{att} is a positive scaling factor, and $\rho_{goal}(q)$ is distance $\|q - q_{goal}\|$

- Attracting force converges linearly towards 0 (goal):

$$\begin{aligned} F_{att}(q) &= -\nabla U_{att}(q) \\ &= -k_{att} \cdot \rho_{goal}(q) \nabla \rho_{goal}(q) \\ &= -k_{att} \cdot (q - q_{goal}) \end{aligned}$$

Mathematical Representation: Repulsive Potential Field

- Should generate a barrier around all the obstacles:
  - strong if close to the obstacle
  - no influence if far from the obstacle

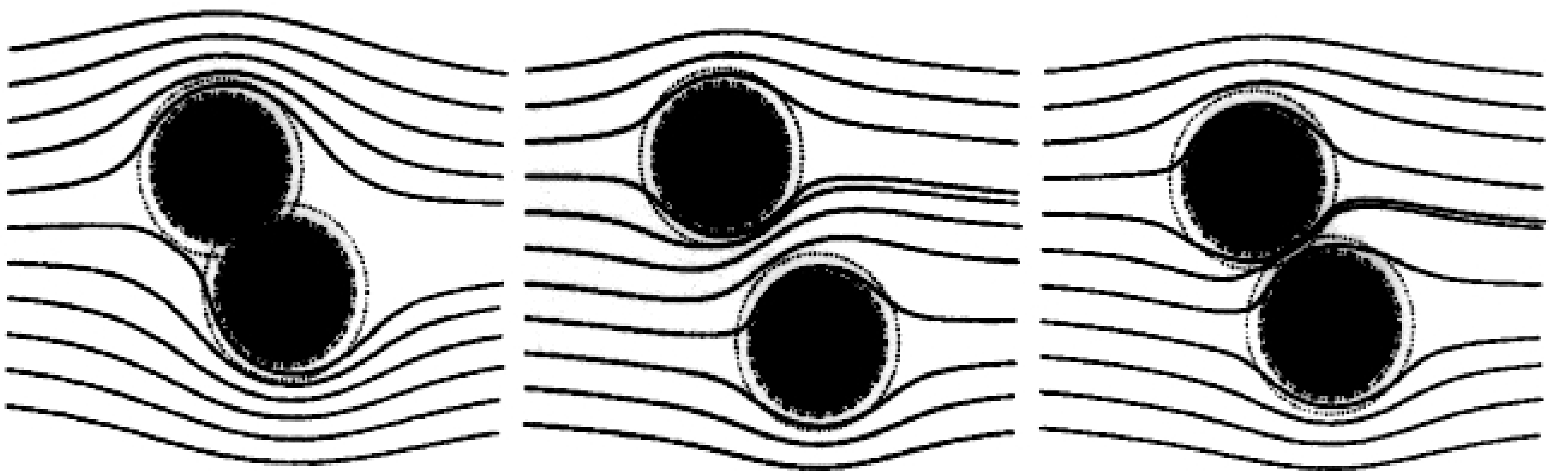
$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho(q) \le \rho_0 \\ 0 & \text{if } \rho(q) \ge \rho_0 \end{cases}$$

  - $\rho(q)$: minimal distance to the obst. from *q*; ρ_0 is distance of influence of obst.
  - Field is positive or zero and *tends to infinity* as *q* gets closer to the obstacle

$$F_{rep}(q) = -\nabla U_{rep}(q) = \begin{cases} k_{rep} \left(\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right) \frac{1}{\rho^2(q)} \frac{q - q_{obstacle}}{\rho(q)} & \text{if } \rho(q) \le \rho_0 \\ 0 & \text{if } \rho(q) \ge \rho_0 \end{cases}$$

Potential Field Path Planning: Using Harmonic Potentials

- Hydrodynamics analogy
  - robot is moving similar to a fluid particle following its stream
- Ensures that there are no local minima

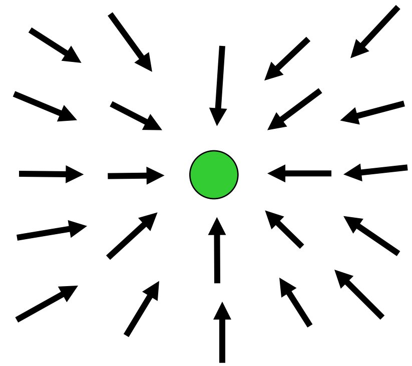


- Note:
  - Complicated, only simulation shown

Motor Schemas:
Example Motor Schema Encodings

- Move-to-goal (ballistic):

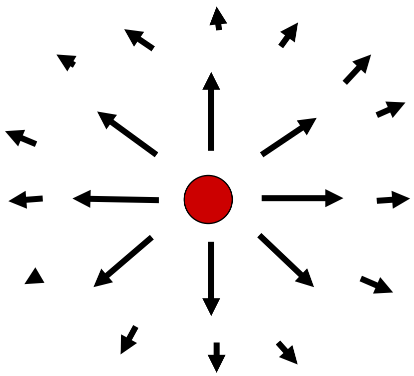
  $V_{magnitude}$ = fixed gain value
  $V_{direction}$ = towards perceived goal



- Avoid-static-obstacle:

$$V_{magnitude} = \begin{cases} 0 & \text{for } d > S \\ \dfrac{S-d}{S-R} * G & \text{for } R < d \leq S \\ \infty & \text{for } d \leq R \end{cases}$$

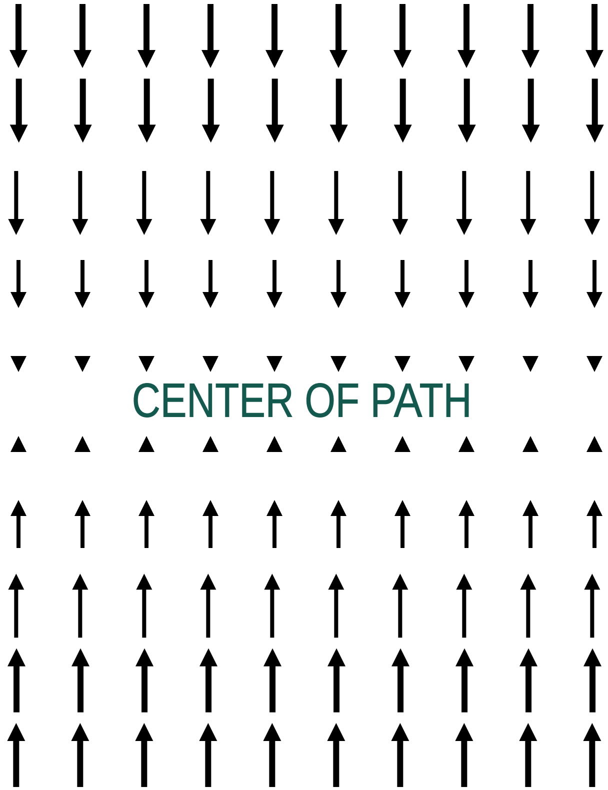
where S = sphere of influence of obstacle
R = radius of obstacle
G = gain
d = distance of robot to center of obstacle



More Motor Schema Encodings

- Stay-on-path:

$$V_{magnitude} = \begin{cases} P & \text{for } d > (W/2) \\ \dfrac{d}{W/2} * G & \text{for } d \leq (W/2) \end{cases}$$



where:

W = width of path

P = off-path gain

G = on-path gain

D = distance of robot to center of path

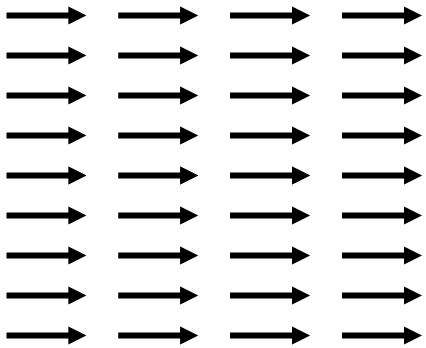
$V_{direction}$ = along a line from robot to center of path, heading toward centerline

More Motor Schema Encodings (con't.)

- Move-ahead:

  $V_{magnitude}$ = fixed gain value

  $V_{direction}$ = specified compass direction



- Noise:

  $V_{magnitude}$ = fixed gain value

  $V_{direction}$ = random direction changed every *p* time steps