CSE-481
Robotics Capstone

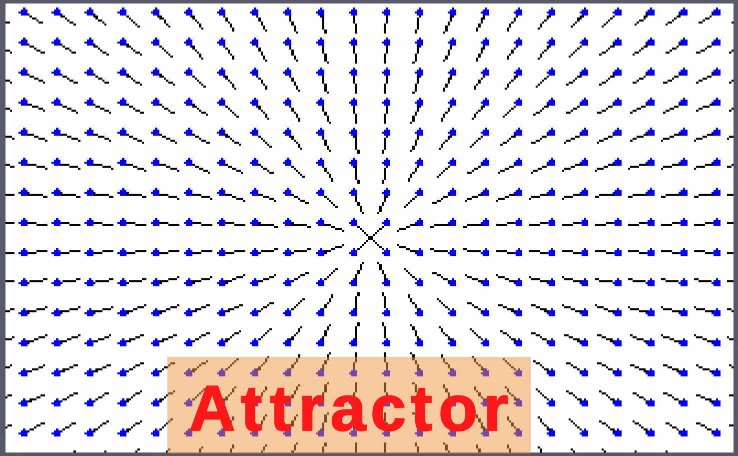
Motion planning

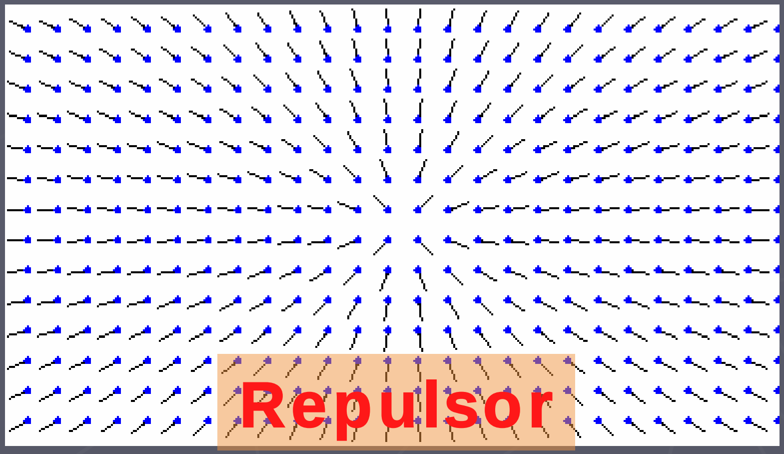
Behavior-based Control

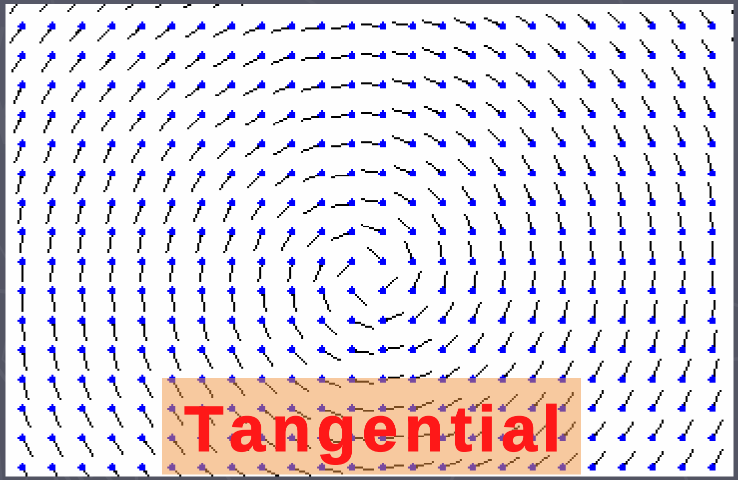
Potential Field Methodologies

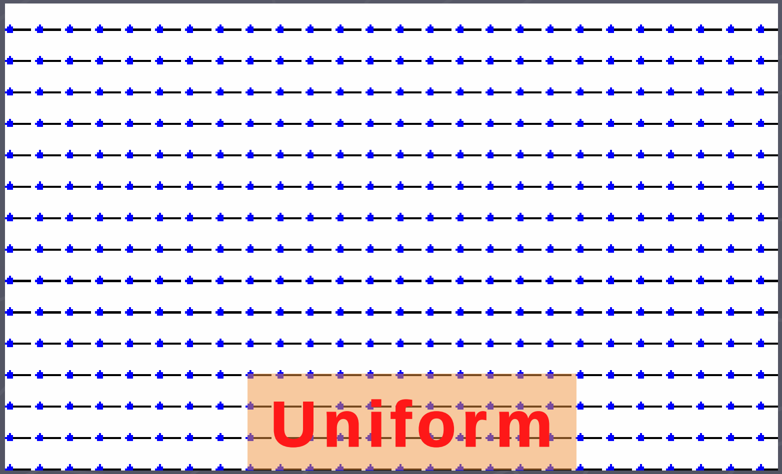
- Treat robot as particle acting under the influence of a potential field
- Robot travels along the derivative of the potential
- Field depends on obstacles, desired travel directions and targets
- Resulting field (vector) is given by the summation of primitive fields
- Strength of field may change with distance to obstacle/target

Four Primitive Motion Fields

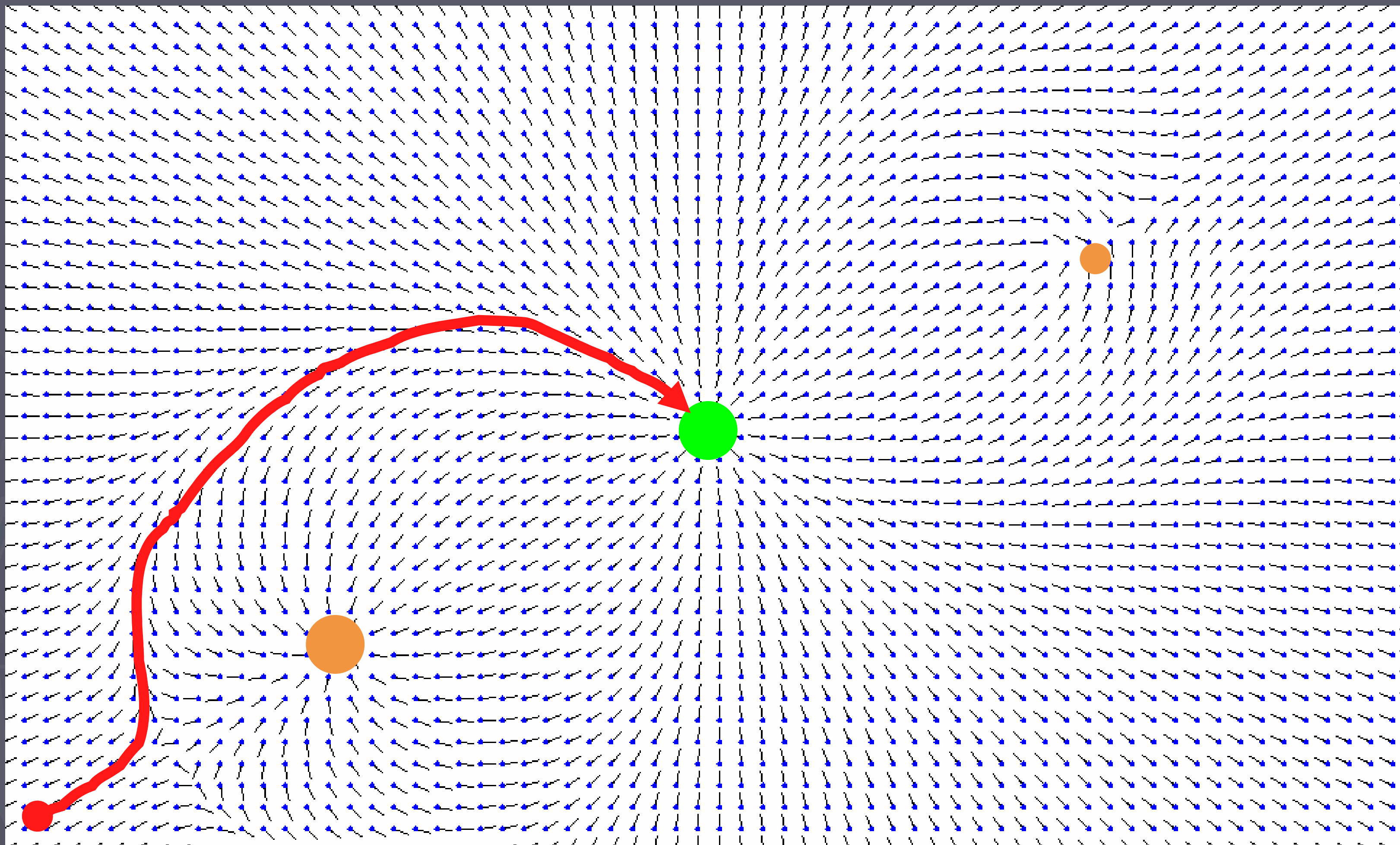








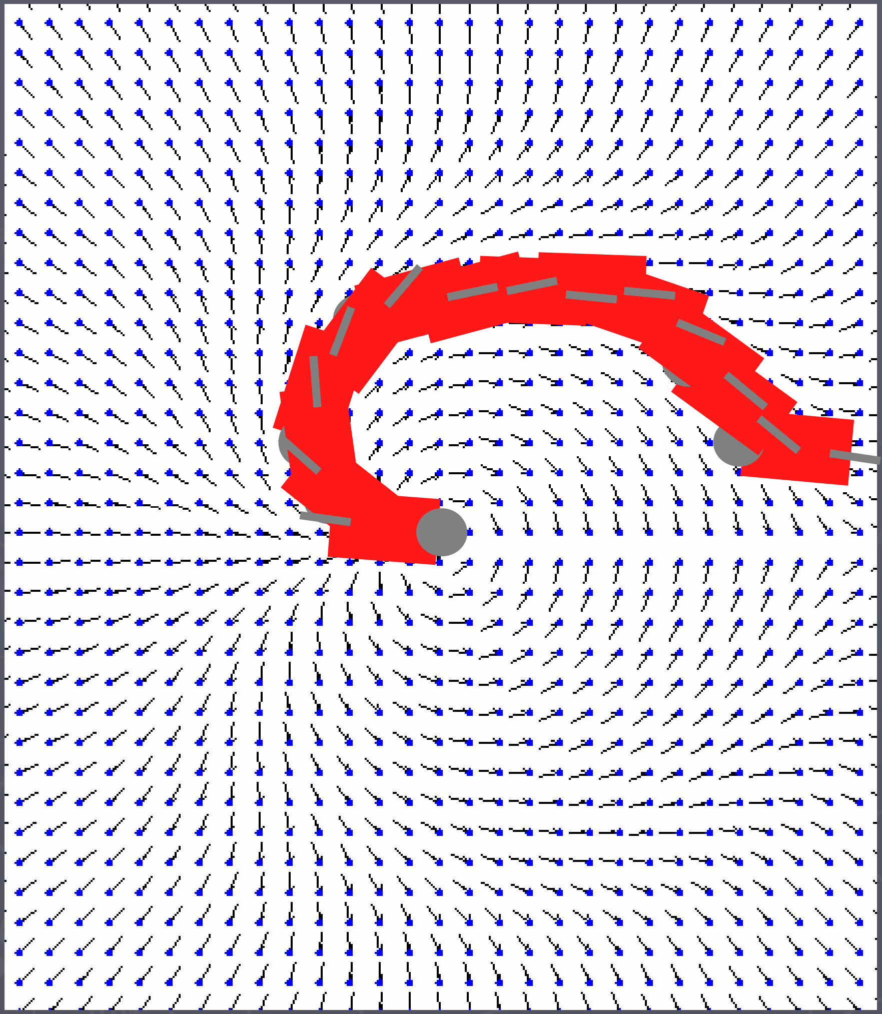
Combined Motion Fields (Goal with Obstacles)



Docking Fields

- Complex motion field
- Used to approach a goal from a particular direction
- Combines
  - Attractor field to guide robot near target
  - Tangential field to guide robot around target
  - Docking "cone" to attract robot when it is in the correct position

Docking Field



- Difficult to find good parameters
- Smooth transitions are important

Characteristics of Potential Fields

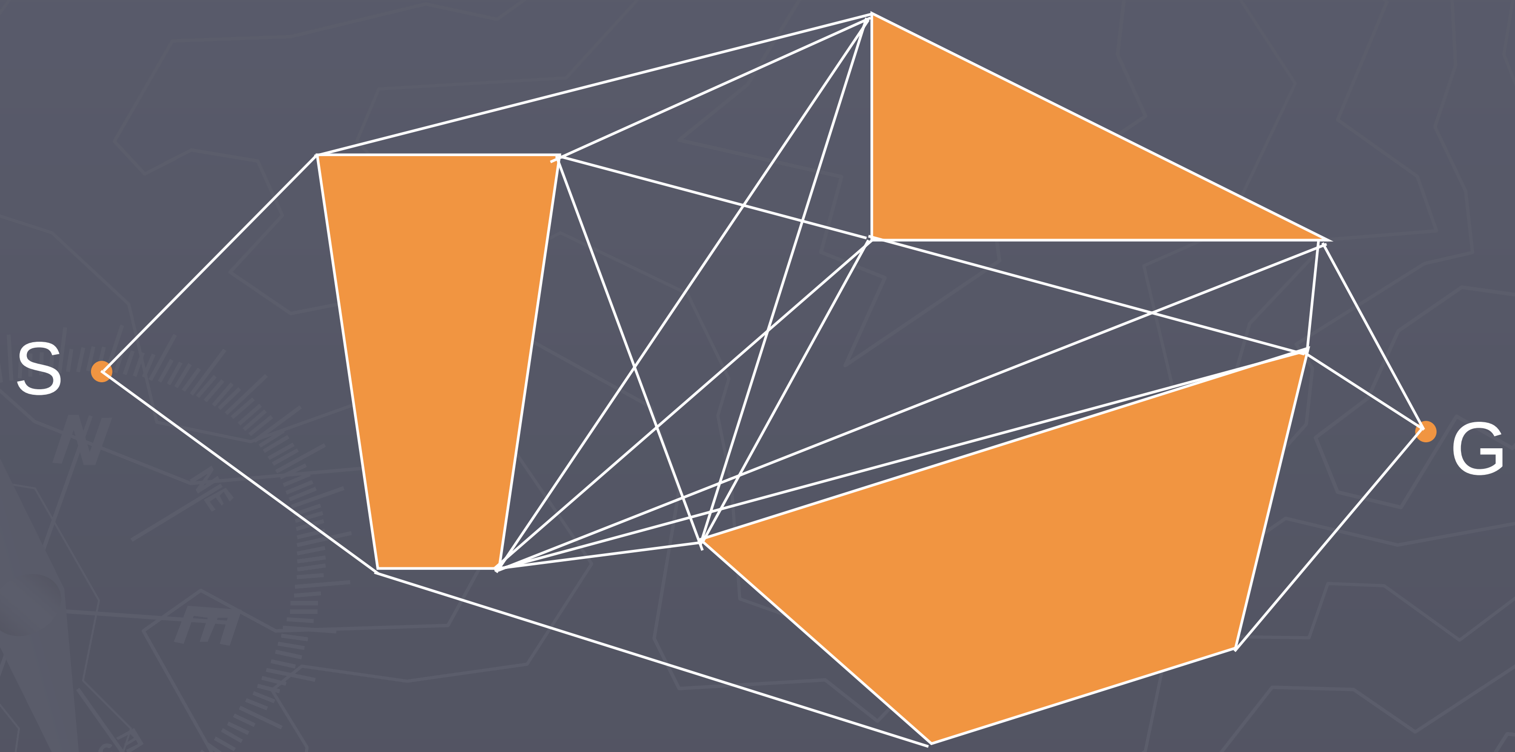
- Easy to visualize
- Easy to combine different fields
- High update rates necessary
- Parameter tuning important
- Difficult to generate complex motion
- No optimality

Path Planning Techniques

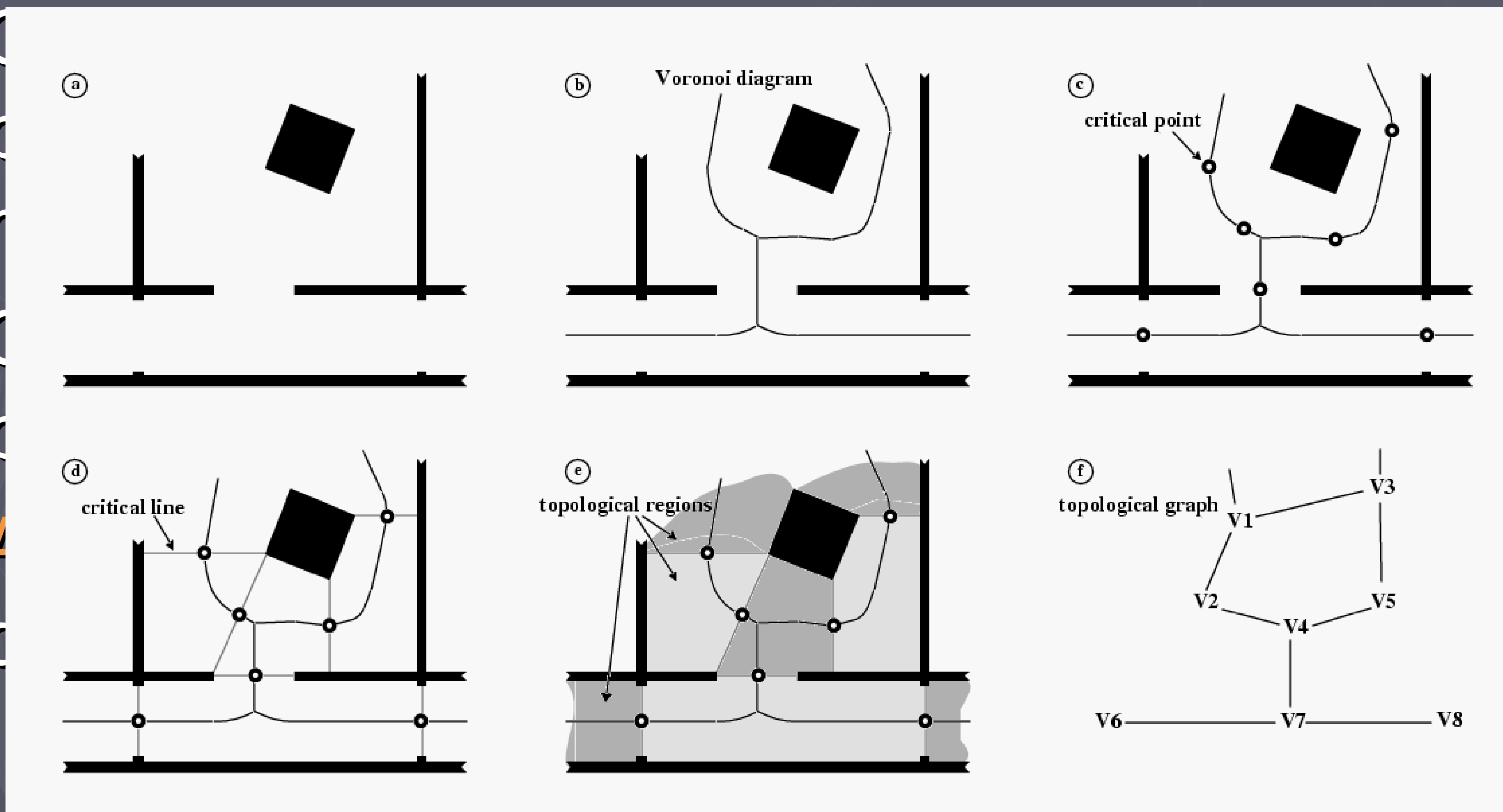
Discretization of Continuous Space

- Extract a discrete search space / graph from the continuous map.
- Redcuce path planning to graph search
  - Visibility graphs / roadmaps
  - Voronoi diagrams
  - Discrete grids

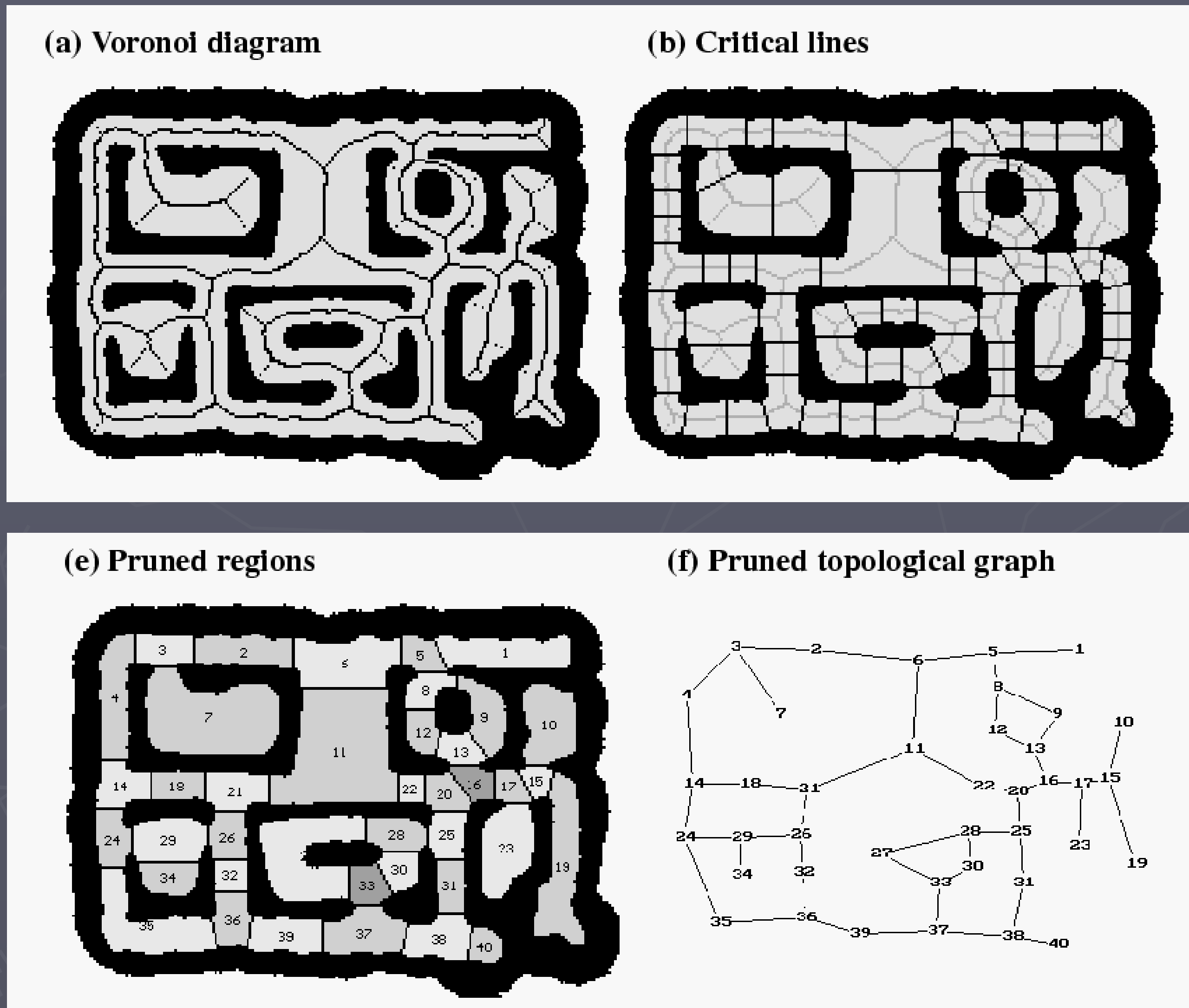
Visibility Graph / Roadmap



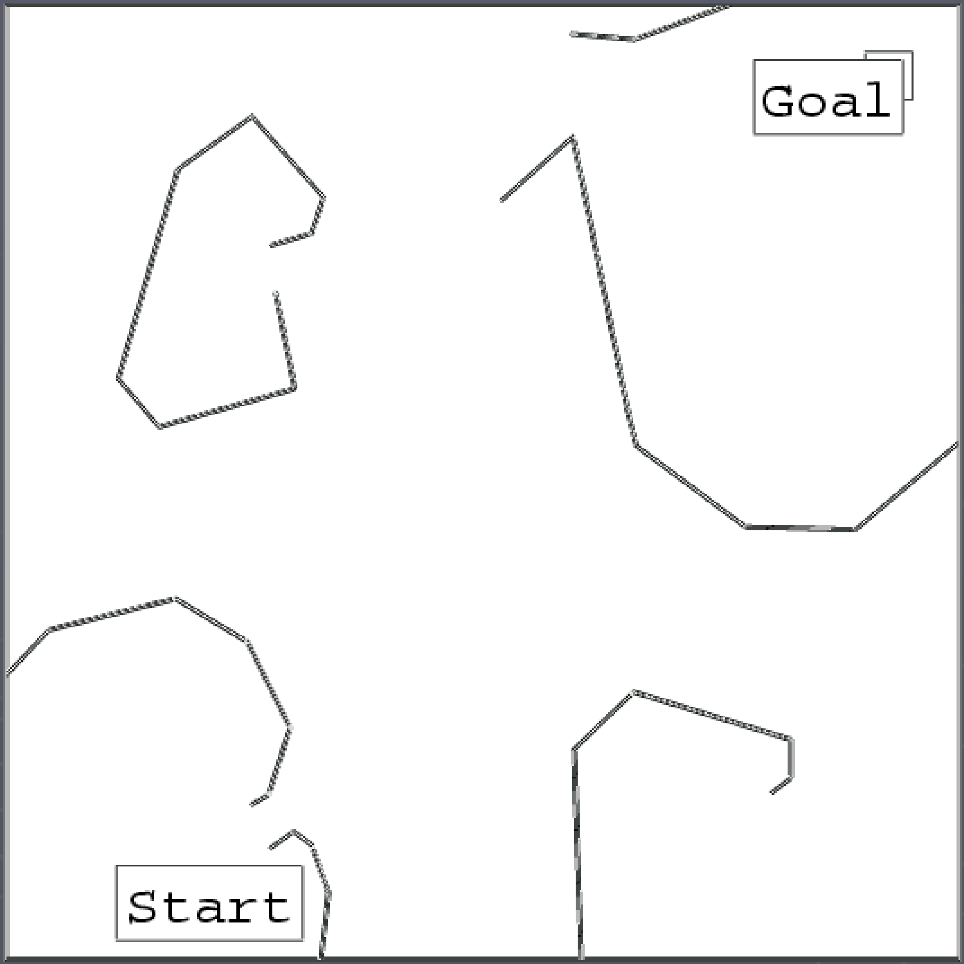
Voronoi Diagrams



Voronoi Example

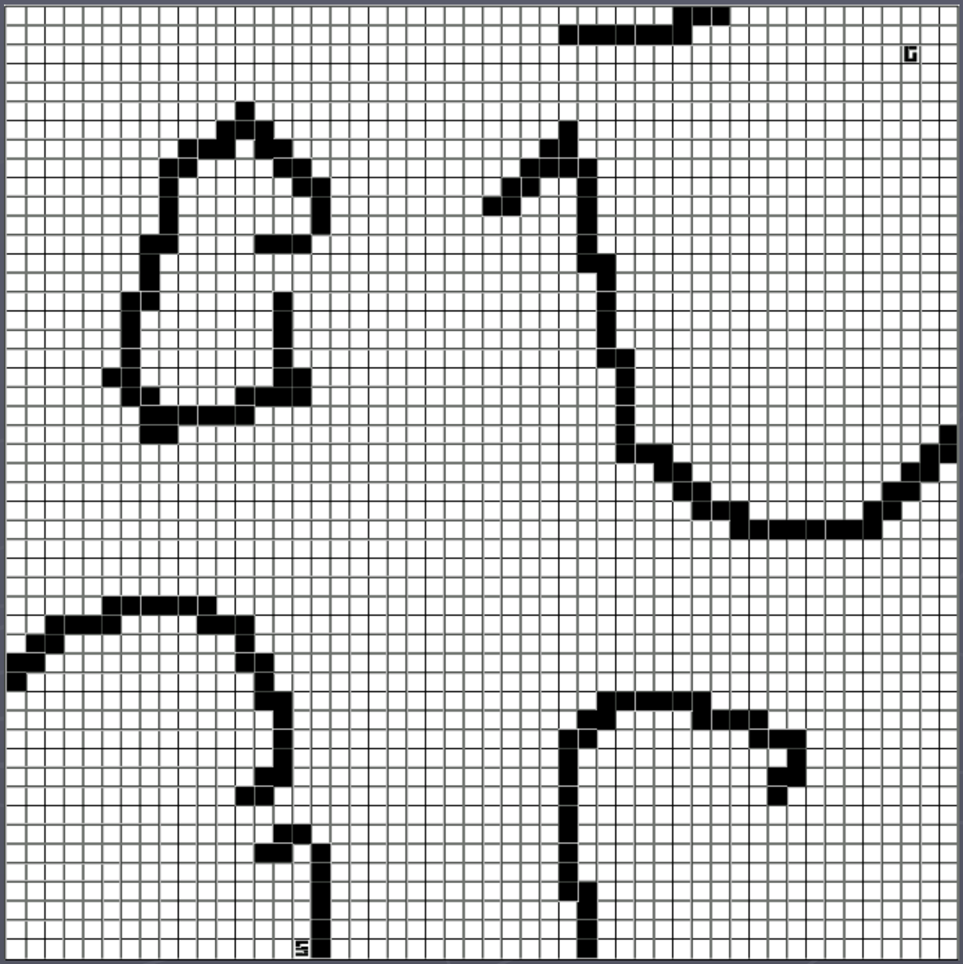


Continuous Environments

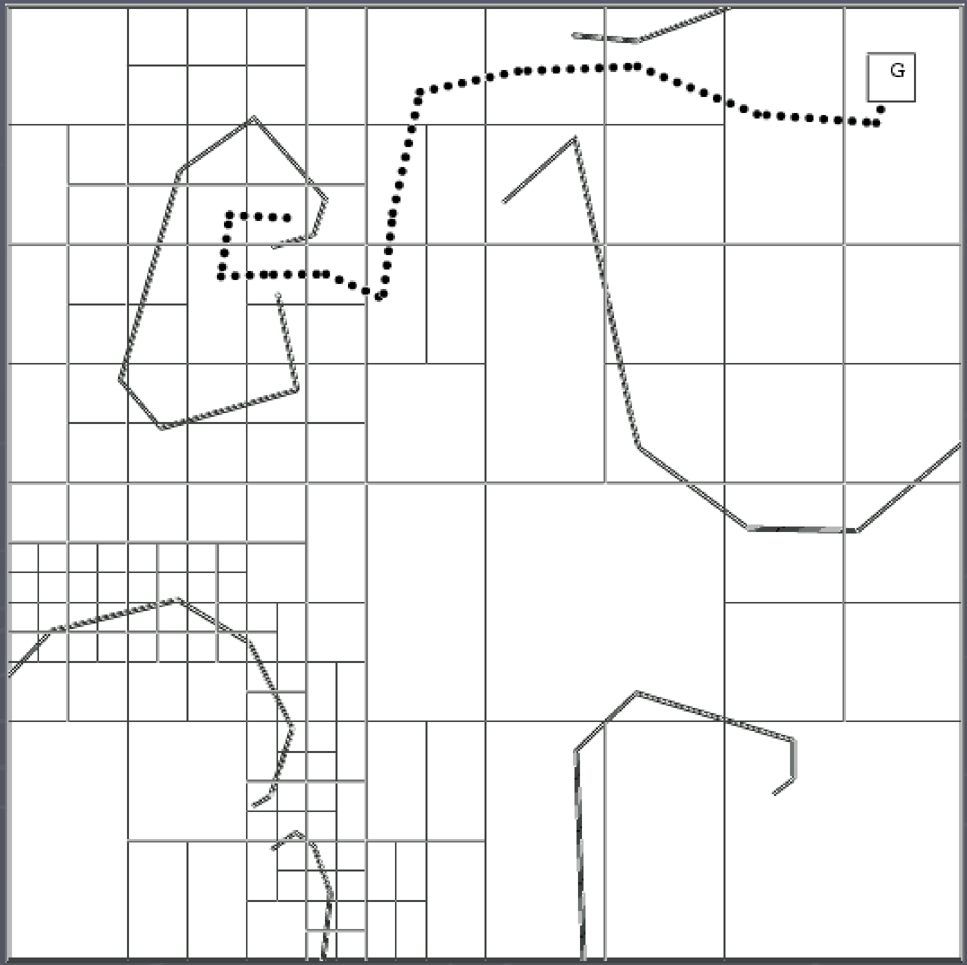


From: A Moore & C.G. Atkeson "The Parti-Game Algorithm for Variable Resolution Reinforcement Learning in Continuous State spaces," Machine Learning 1995

Approximate Cell Decomposition
[Latombe 91]



Parti-Game
[Moore 96]



Search on the Graph

- Can apply A* to find optimal path to goal.
- Only expands nodes needed to find optimal path.
- Not sufficient if motion is uncertain.
- In some cases, replanning is efficient enough.
- D* extends A* to efficient re-planning.

Gradient Method (Konolige)

- Path is list of points $P=\{p_1, p_2, \ldots p_k\}$
- p_k is only point in goal set
- Cost of path is separable into intrinsic cost at each point along with adjacency cost of moving from one point to the next

$$F(P) = \sum_i I(p_i) + \sum_i A(p_i, p_{i+1})$$

  - Adjacency cost typically Euclidean distance
  - Intrinsic cost typically occupancy, distance to obstacle

Navigation Function

- Assignment of potential field value to every element in configuration space [Latombe].
- Goal set is always downhill, no local minima.
- Navigation function of a point is cost of minimal cost path that starts at that point.

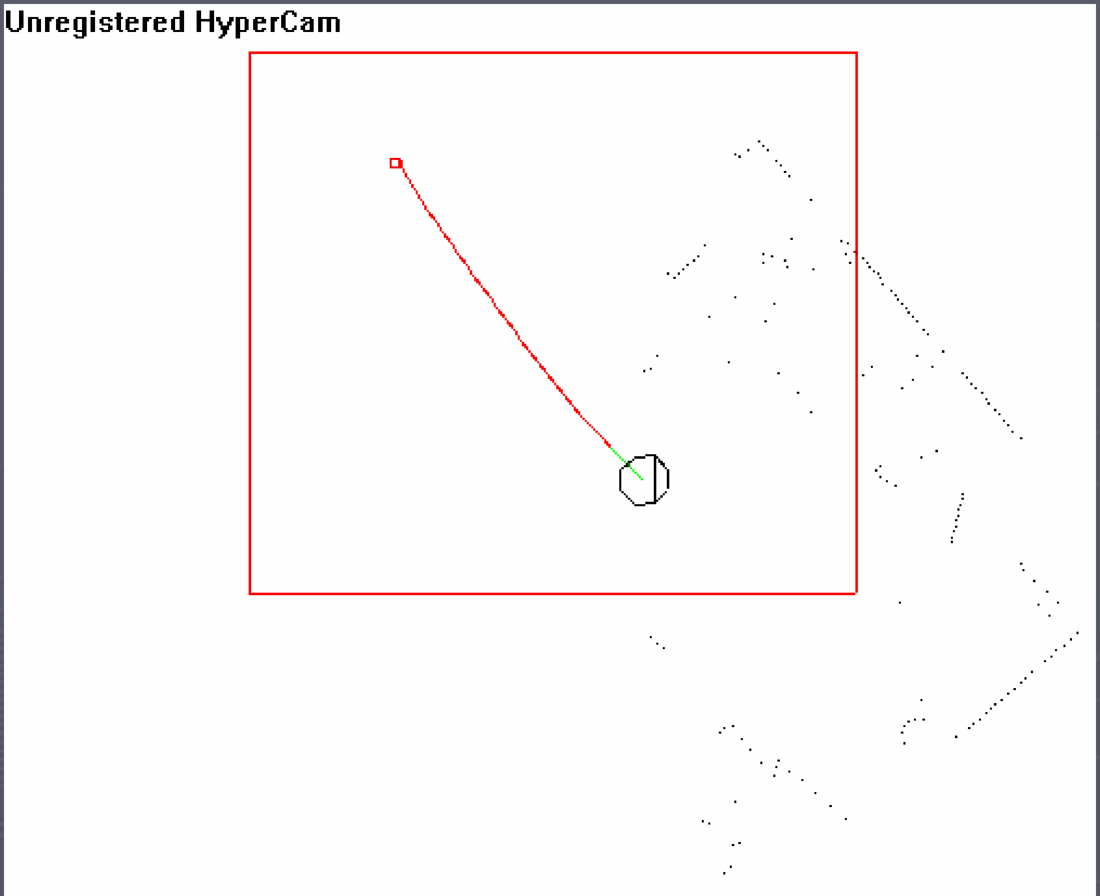
$$N_k = \min_{P_k} F(P_k)$$

Computation of Navigation Function

- Initialization
  - Points in goal set ← 0 cost
  - All other points ← infinite cost
  - Active list ← goal set

- Repeat
  - Take point from active list and update neighbors
  - If cost changes, add the point to the active list
- Until active list is empty

Example Run



Further Details

- Keep paths away from obstacles by setting intrinsic cost dependent on distance from obstacles.
- Local perceptual space depends on quality of sensors and odometry.
- Can easily achieve 10Hz update rate on low-end PC (10 x 10m LPS, 10cm resol.).
- Performs superior to human operator.

Combination of Reactive Behaviors

Combining Reactive Behaviors

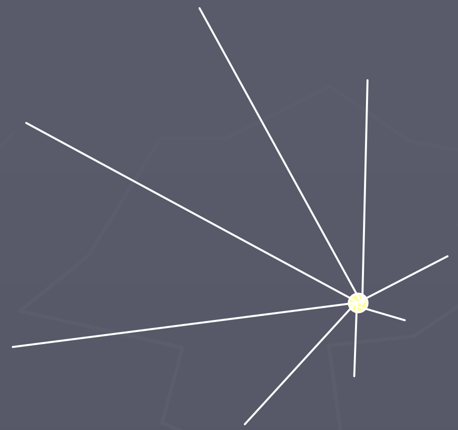
- Reactive behaviors don't scale well.
- Need to find a way to combine reactive behaviors into a larger behavior system.
- Three ideas (other methods possible):
  - Competition - reactive behaviors compete for control of the robot.
  - Subsumption - reactive behaviors selectively take control of the robot.
  - Sequencing - reactive behaviors are sequenced by a higher level controller.

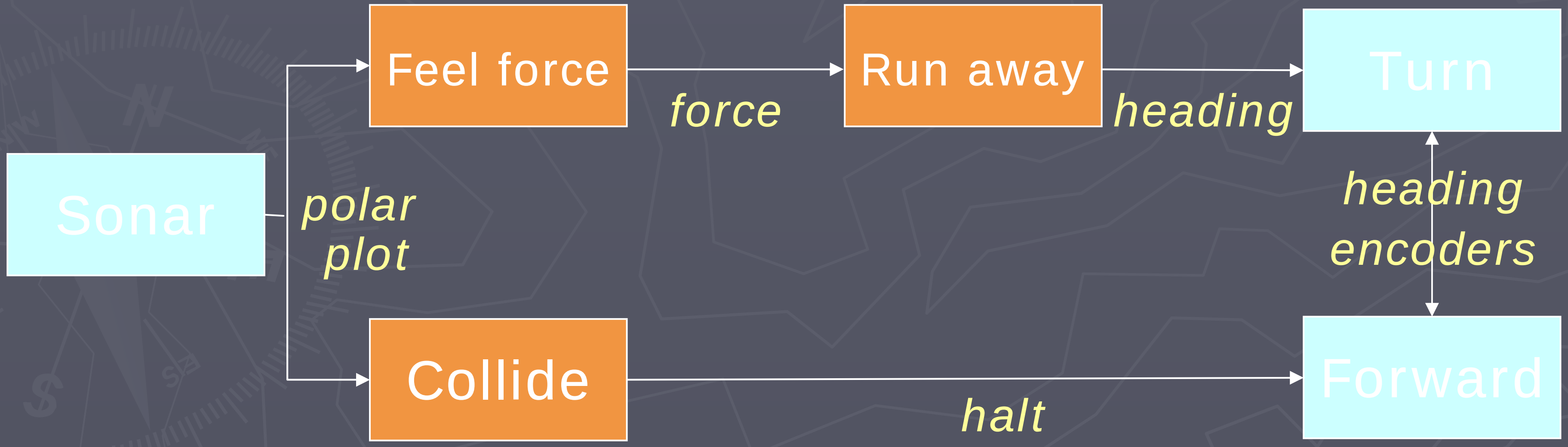
Subsumption Architecture

- Introduced by Rodney Brooks ’86.
- Behaviors are networks of sensing and acting modules (augmented finite state machines AFSM).
- Modules are grouped into layers of competence.
- Layers can subsume lower layers.
- No internal state!

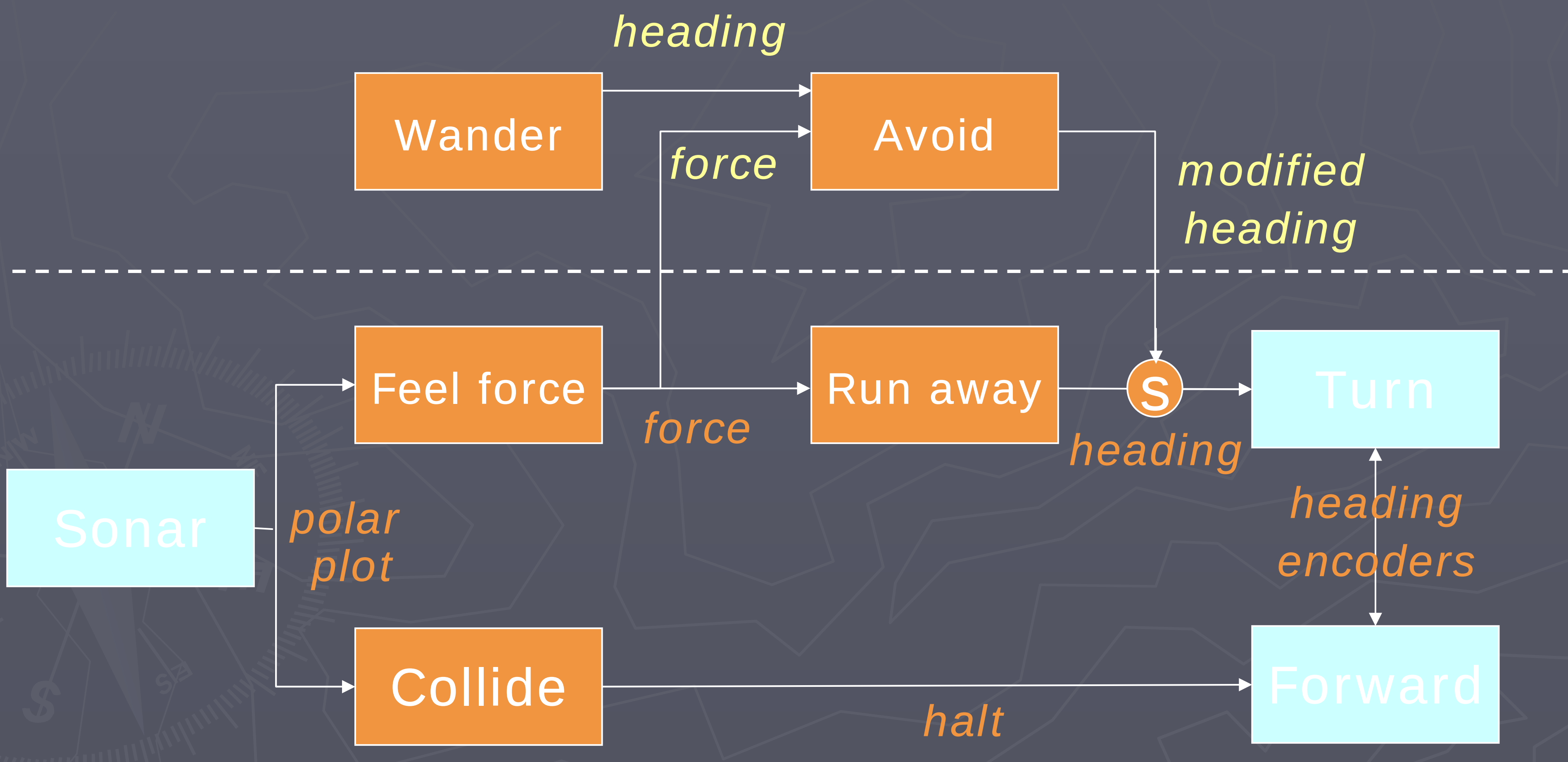
Level 0: Avoid

Polar plot of sonars

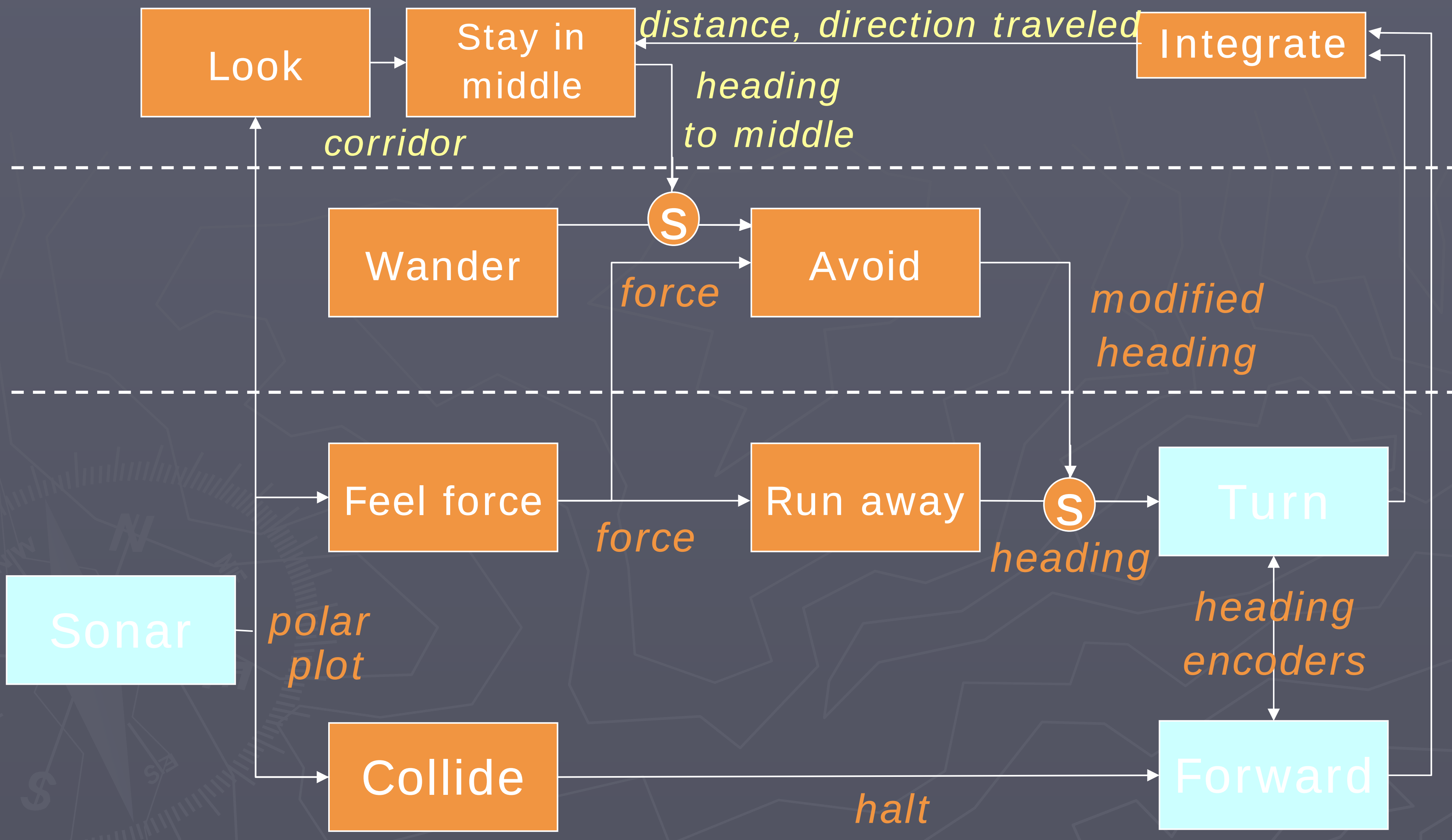




Level 1: Wander



Level 2: Follow Corridor



Behaviors Design

- Behavior design is more of an art form than a science.
- Good behaviors produce smoothly varying control signals.
- Control signals that oscillate or otherwise jump around lead to poor control performance.
- Oscillation amongst behaviors needs to be avoided because it leads to oscillatory control signals.

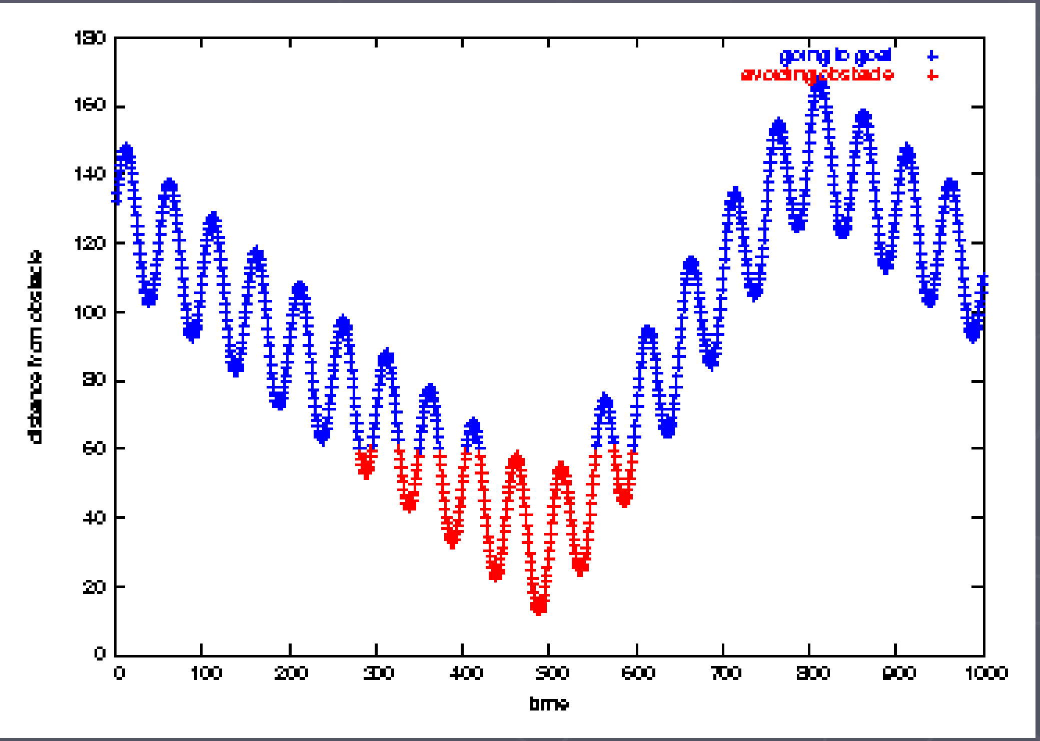
Avoiding Oscillation

- Oscillation can occur any time there is a transition path among a set of states.
- It usually happens at the boundary between states.
- There are two basic ways to reduce oscillation:
  - Merge 2 similar states together.
  - Add hysteresis to the transition rules.
- Oscillation can also occur if there are states where the robot fails to make progress towards the goal.

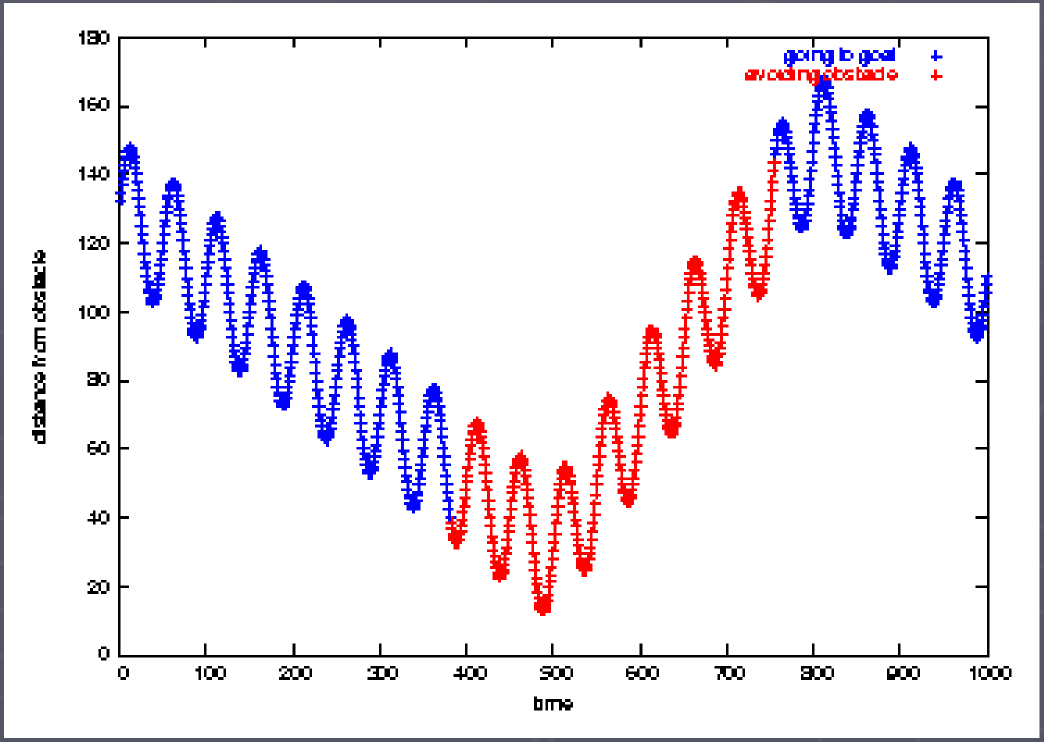
Hysteresis

- A system is said to exhibit hysteresis when the behavior of the system depends not only on its current state, but also on its history.
- In the context of behaviors, hysteresis refers to the creation of a buffer zone between two states.
- Within the buffer zone, the robot simply uses whatever state it was using when it entered the buffer zone.
- This is sometimes called a dual threshold because there are 2 thresholds involved instead of one.

No Hysteresis



Hysteresis



going to goal
avoiding obstacle
distance from obstacle
time
180
160
140
120
100
80
60
40
20
0
0
100
200
300
400
500
600
700
800
900
1000