LECTURE 2 - 2

The Reactive Paradigm

Introduction to AI Robotics (Ch. 4)

Quote of the Week

"In the fifties, it was predicted that in 5 years robots would be everywhere.
In the sixties, it was predicted that in 10 years robots would be everywhere.
In the seventies, it was predicted that in 20 years robots would be everywhere.
In the eighties, it was predicted that in 40 years robots would be everywhere..."

Marvin Minsky

ANNOUNCEMENTS

- Lab 2 - Random Wander, Obstacle Avoidance demonstration is due on *Thursday, 3/18/10*
- The lab memo and code is due on Angel by midnight on *Thursday, 3/18/10*
- Quiz 4 on Lecture 2 – 2, Ch. 4 on *Monday, 3/22/10*

OBJECTIVES

Upon completion of this lecture the student should be able to:

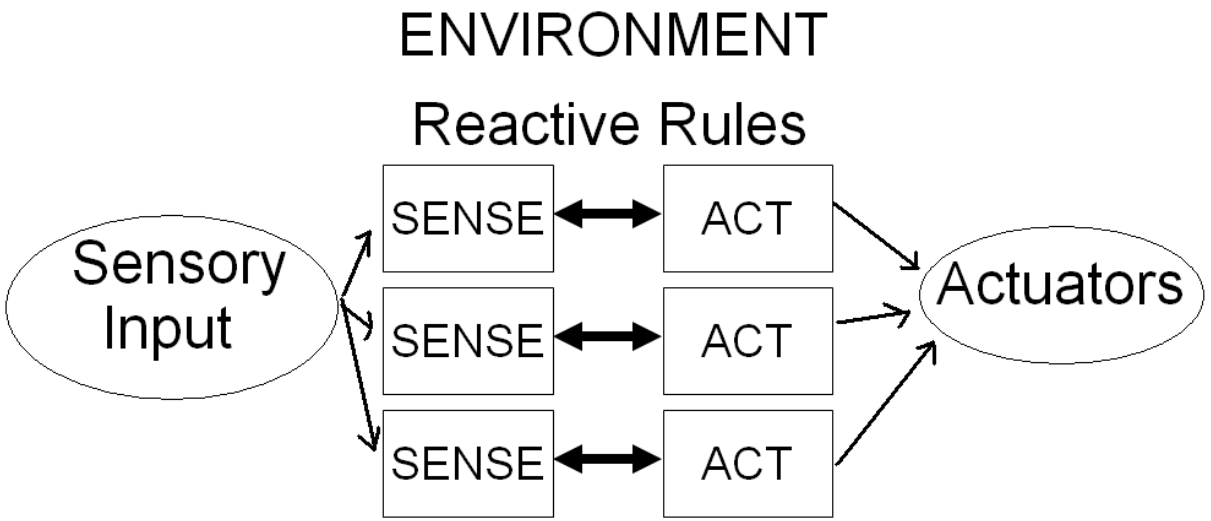
- Define the reactive paradigm in terms of the 3 primitives and sensing organization
- List the characteristics of a reactive robotic system
- Describe two common methods for combining behaviors in a reactive architecture
- Evaluate subsumption and potential fields architectures

REACTIVE CONTROL

- Reactive Control is one of the mostly commonly used methods for robot control and is based on a tight connection between the robot's sensors and effectors
- They do not use any internal representations and do not look ahead at the possible outcomes of their actions
- They operate on a short time scale and react to the current sensory information
- They have reactive rules (i.e. reflexes) to specific sensory input
- Complex computation is removed entirely in favor of fast, stored pre-computed responses

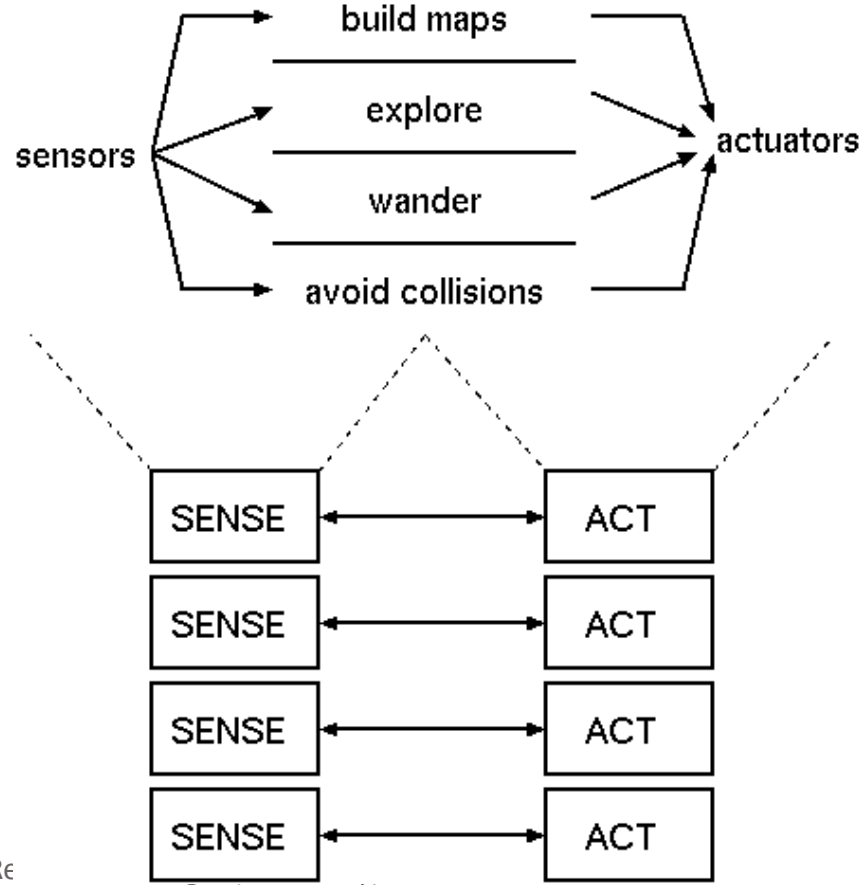
STIMULI AND BEHAVIORS

- The robot system has a set of situations (stimuli or coordinations) and a set of actions (responses, actions, behaviors)
  - The situations may be based on sensory inputs or on internal state
  - Examples are obstacle avoidance or random wander



VERTICAL DECOMPOSITION

Biological systems are more vertical



MUTUALLY EXCLUSIVE CONDITIONS

- To keep a reactive system simple have one unique behavior for each stimuli. The conditions are ***mutually exclusive*.**
- As the sensor state space grows the space may become unwieldy or intractable
- Coming up with the complete set of rules for the state space is typically done at design time not run time
- Typically there are only rules for important events and a default response for all others

STUCK SITUATIONS

- A reactive controller may be get stuck if there is a default rule that covers some of the states. In wall following this may be resolved by the following:
  - Introduce randomness to get the robot unstuck from a corner by having it turn by a random angle instead of a fixed one
  - Keep a history and remember the direction the robot turned last and use that information to make the decision about the turn direction

ACTION SELECTION

- ***Action Selection*** is the process of deciding among multiple possible behaviors when they are not mutually exclusive
- ***Command arbitration*** is the process of selecting one behavior from multiple candidates
- ***Command fusion*** is the process of combining multiple candidate behaviors into a single output behavior for the robot
- Reactive systems must support parallelism and the program must be able to multitask

TWO MAIN REACTIVE SYSTEMS

- Subsumption Architectures
  - Layers of behaviors
  - Control relationships
- Potential fields
  - Concurrent behaviors
  - How to navigate
- They are equivalent in power
- Sometimes there is a mixture of the layers and concurrency

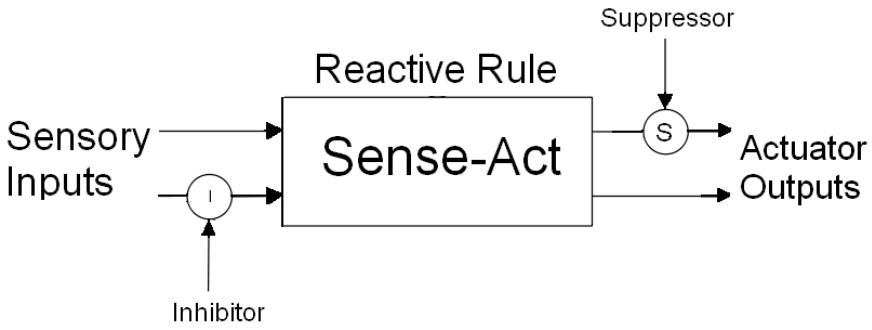
SUBSUMPTION ARCHITECTURE

- ***Subsumption Architecture (SA)*** is a form of reactive control developed by Rodney Brooks in 1985
- SA builds systems incrementally from simple parts to more complex parts
- The complex parts use the simple existing components as much as possible
- The layers are added from the bottom up with the bottom layer being 0 and the most simple

INHIBITION AND SUPPRESSION

- The inputs of a layer/modules may be ***inhibited*** so that it receives no sensory inputs. It will compute no reaction and send no output to effectors or other modules
- Substitutes input going to a module

- The output of a layer/modules may be ***suppressed*** so that it receives sensory inputs but performs no computation and cannot control any effectors or other modules.
- Turns off output from a module



SUBSUMPTION ARCHITECTURE

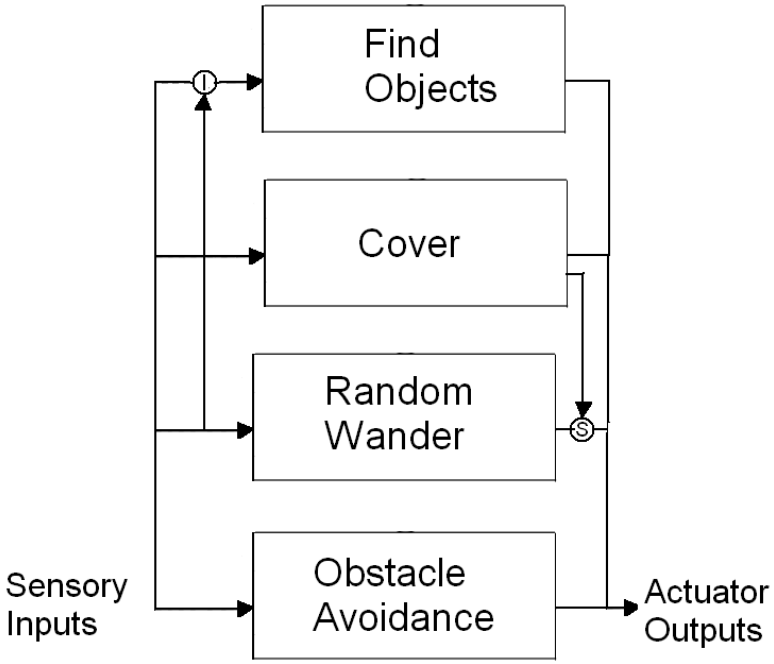
- Higher layers can assume the existence of lower ones and the goal they are achieving or use the lower ones to achieve their own goals
- Higher layers can subsume lower ones by using them while they are running or by suppressing them selectively
- The controller is bottom up because it progresses from simpler to more complex as layers are added incrementally
- The architectures should be taskable accomplished by a higher level turning lower levels on and off

REPRESENTATION AND SUMMARY

- In ***Subsumption Architecture*** the world is its own best model
- If the world can provide the information directly through sensing is best rather than to store it internally
- Components are task-achieving actions/behaviors
- Lowest layers handle the most basic tasks
- Higher layers exploit the existing ones

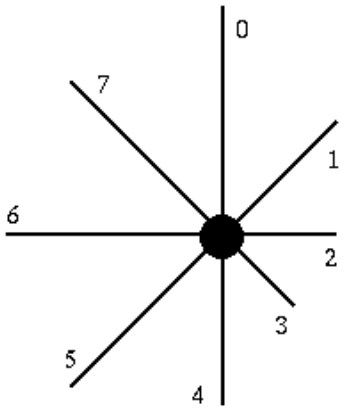
SUBSUMPTION-BASED ROBOT CONTROL SYSTEM

- The goal is to have very few connections between different layers
- The only intended connections are inhibition and suppression
- There are strongly coupled connections within layers
- Loosely coupled connections between layers

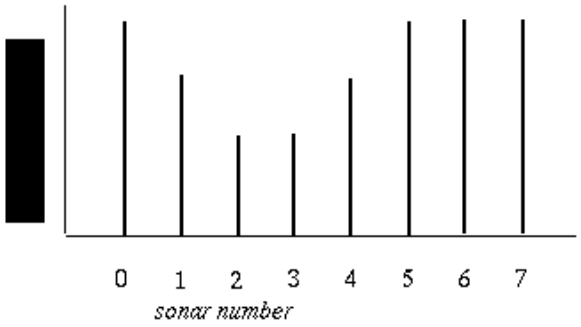


PERCEPTION POLAR PLOT

- The perception plot is egocentric to eliminate a need for memory representation
- Sonar 2 and 3 indicate that there is a wall and no need for memory or reasoning

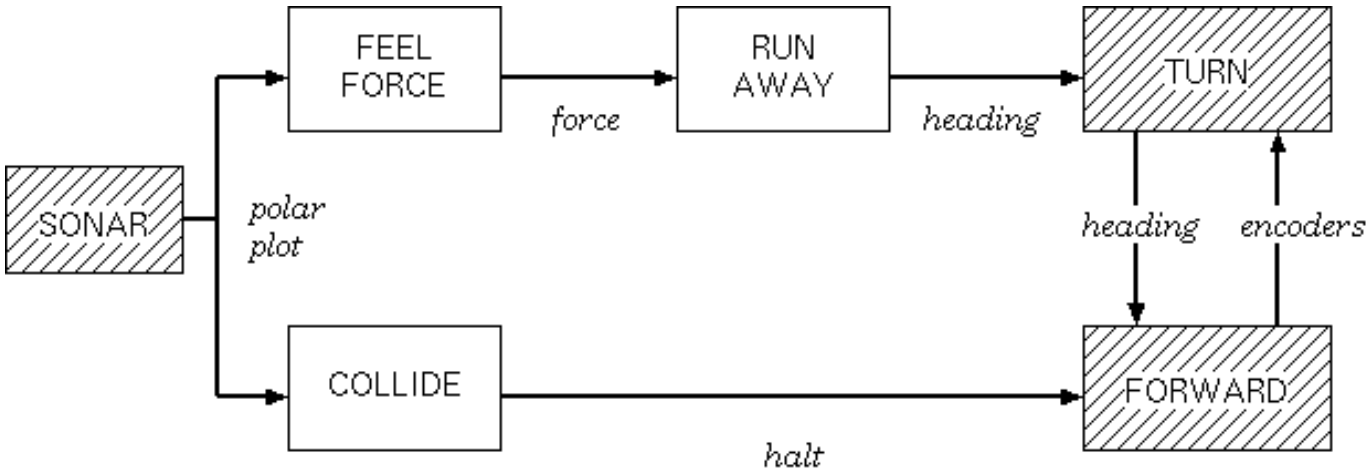


a.

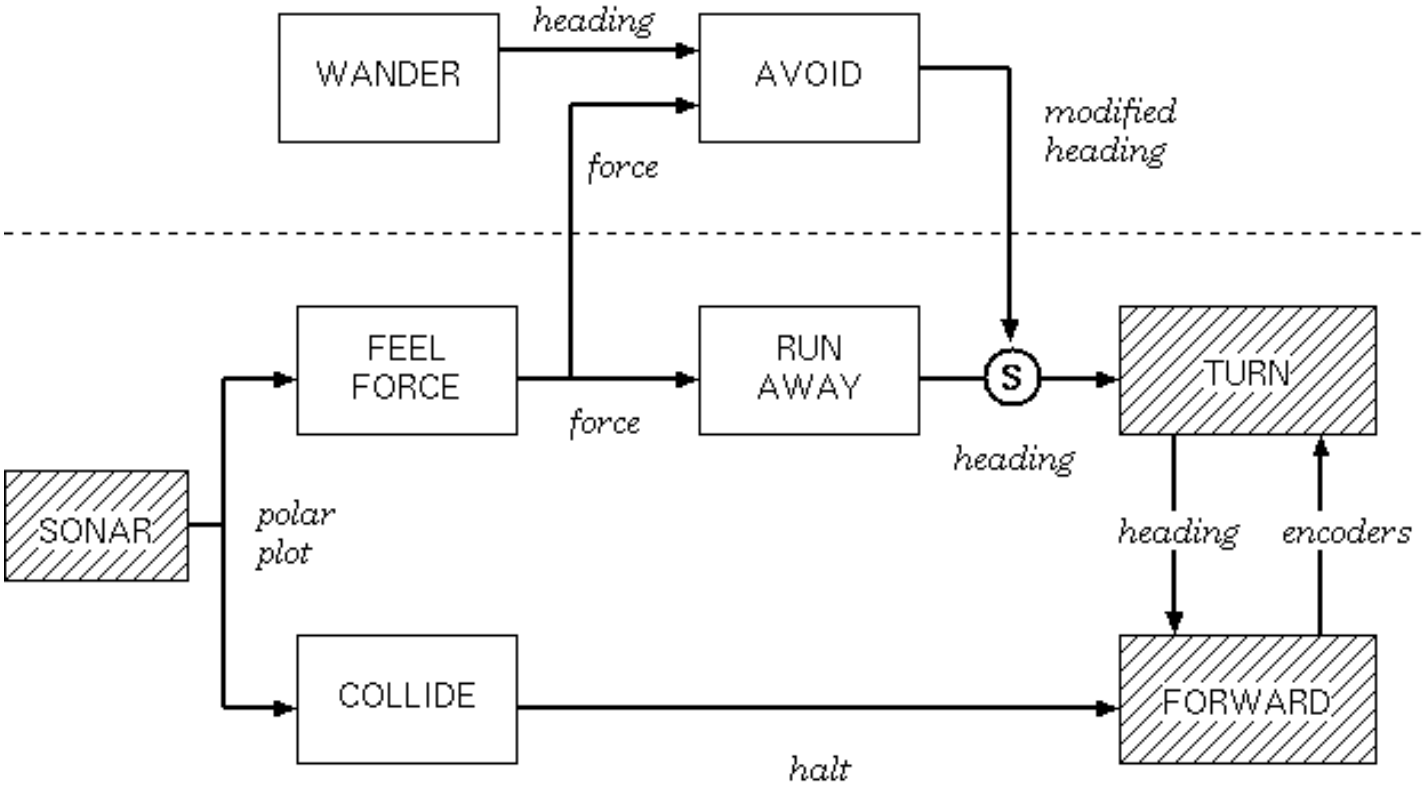


b.

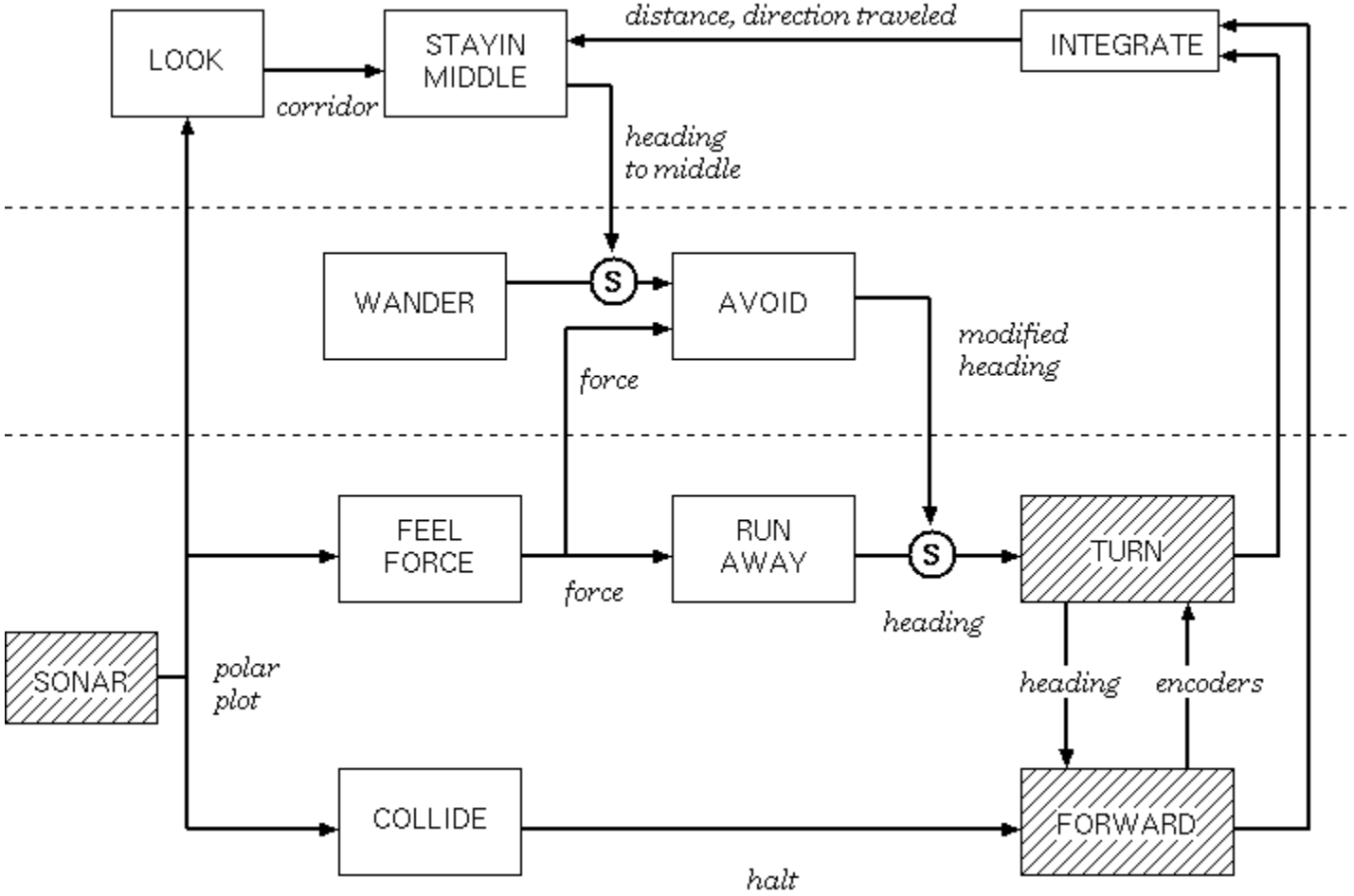
LEVEL 0: RUN AWAY



LEVEL 1: WANDER



LEVEL 2: FOLLOW CORRIDOR

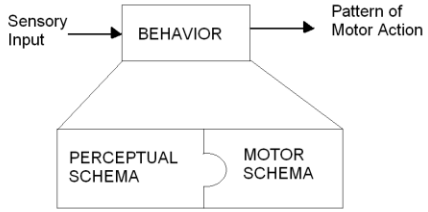


LIMITATIONS

- Minimal state if any
- No memory
- No learning
- No internal models/representations of the world

SCHEMA THEORY

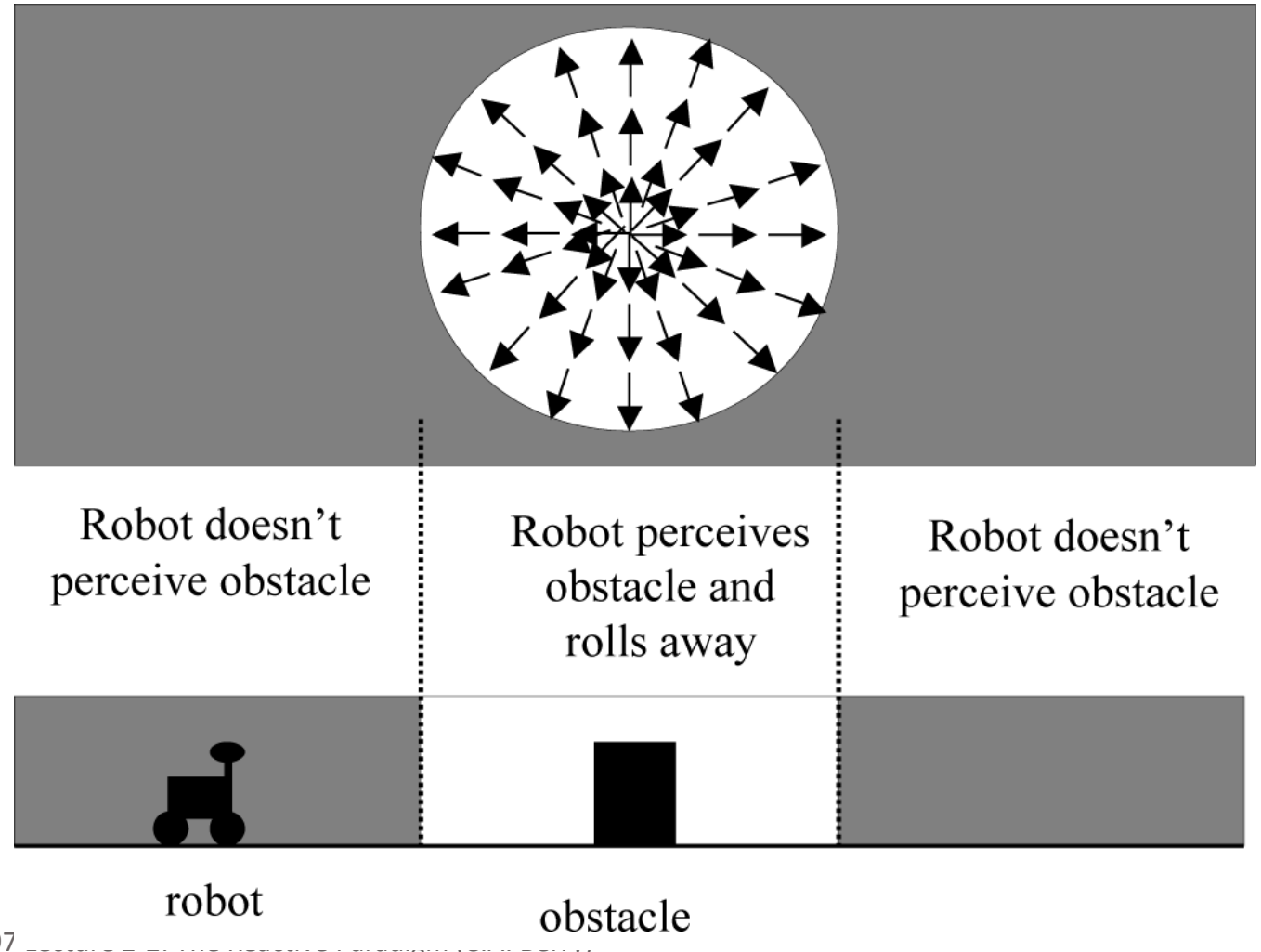
- Schema is used to express the basic unit of activity
- A schema consists both of the knowledge of how to act and /or perceive as well as the computational process by which it uses to accomplish the activity
- A schema class in C would contain both data and methods
- The ***motor schema*** represents the template for the physical activity
- The ***perceptual schema*** embodies the sensing



POTENTIAL FIELDS

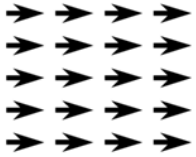
- The motor schema can be expressed with potential fields methodology
  - A potential field can be constructed from primitives summed together
  - The behavior output are combined using vector summation
- The robot feels a vector or force from each behavior
  - Magnitude
  - Direction
- Every point in space represents a force as a field that it would feel at that point

ROBOT RUN AWAY VIA POTENTIAL FIELDS

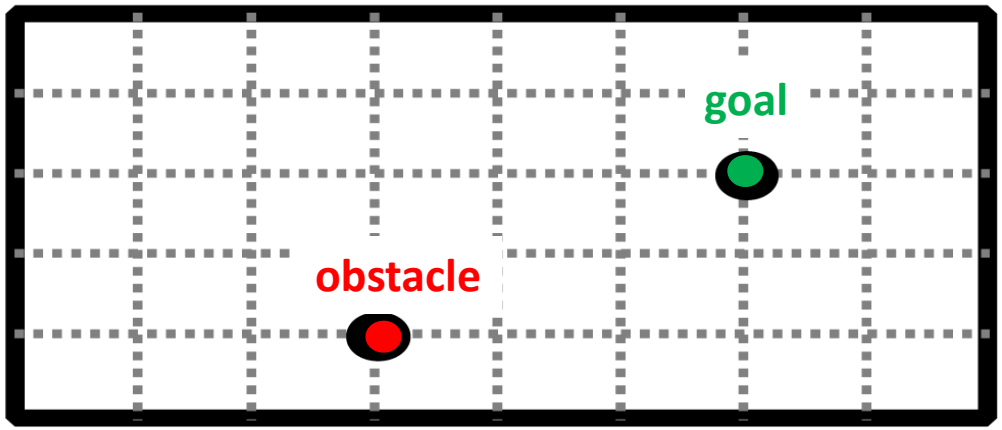


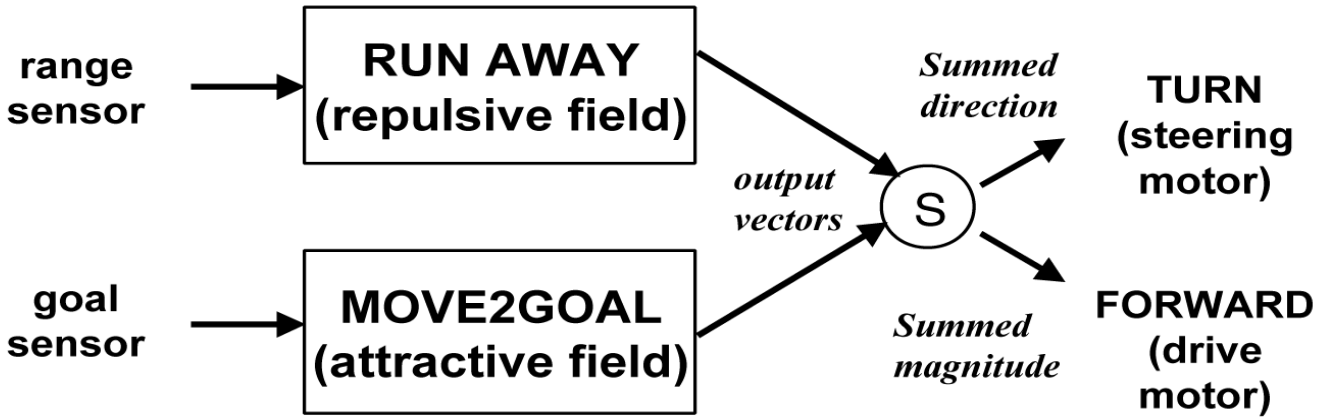
PRIMITIVE POTENTIAL FIELDS

- Uniform
  - Move in a particular direction, wall following
- Repulsion
  - Run away, obstacle avoidance
- Attraction
  - Move to goal
- Tangential
  - Move through the door, docking
- Random
  - Potential field?

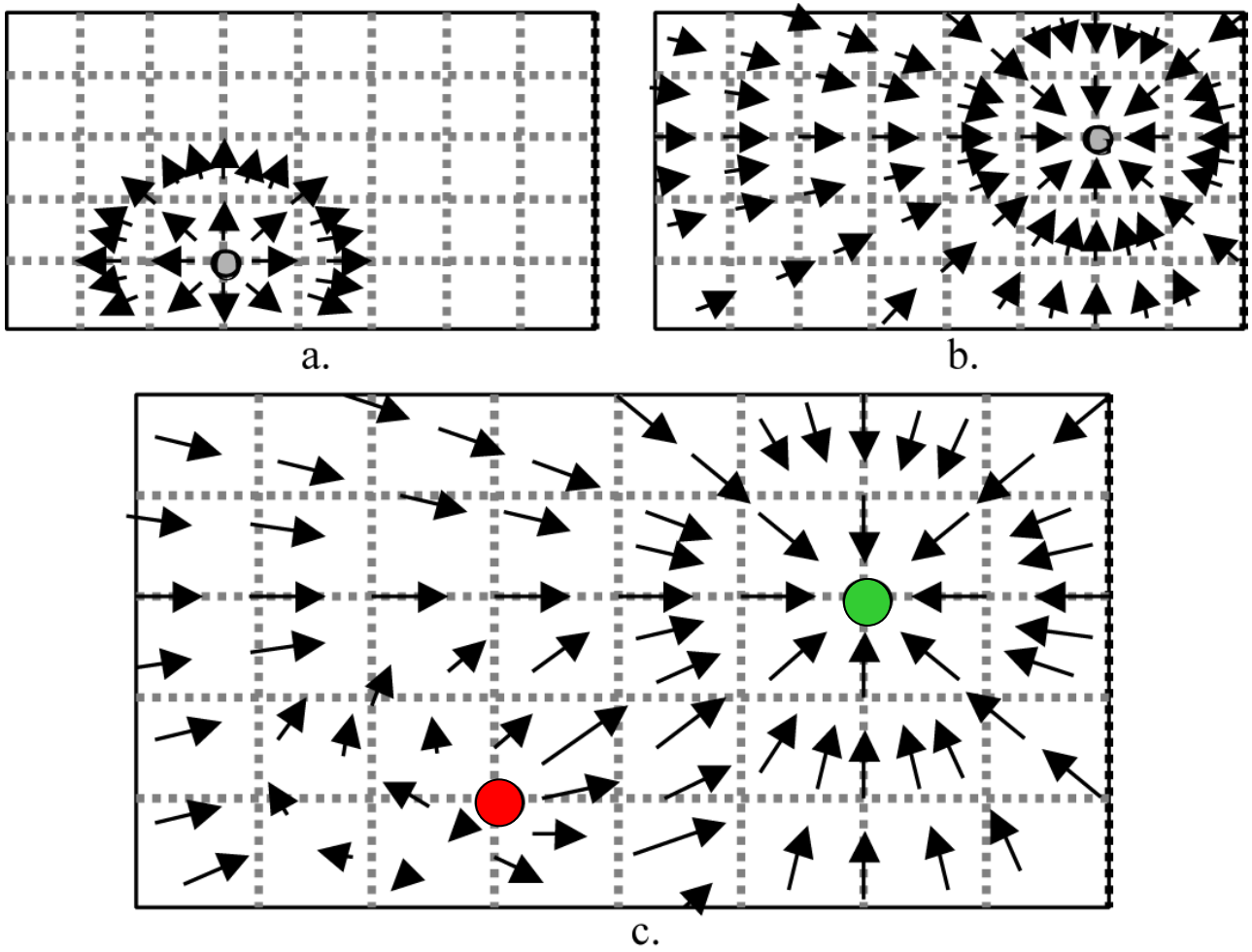


COMBINING FIELDS FOR EMERGENT BEHAVIOR



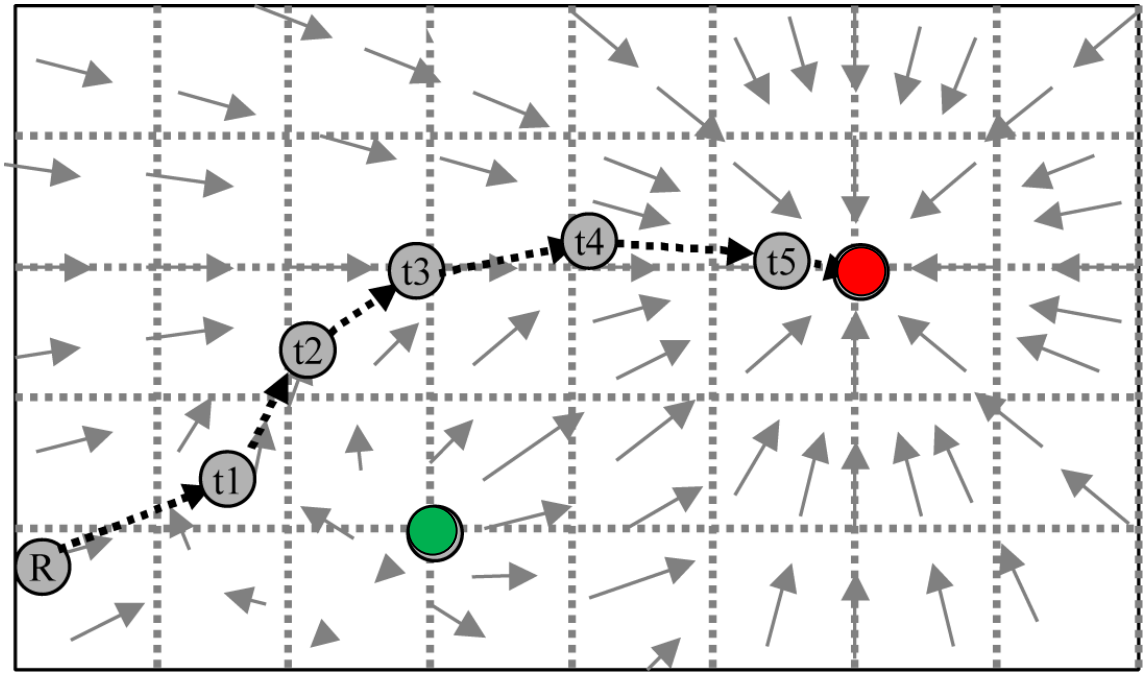


ATTRACT AND REPLUSIVE FIELDS

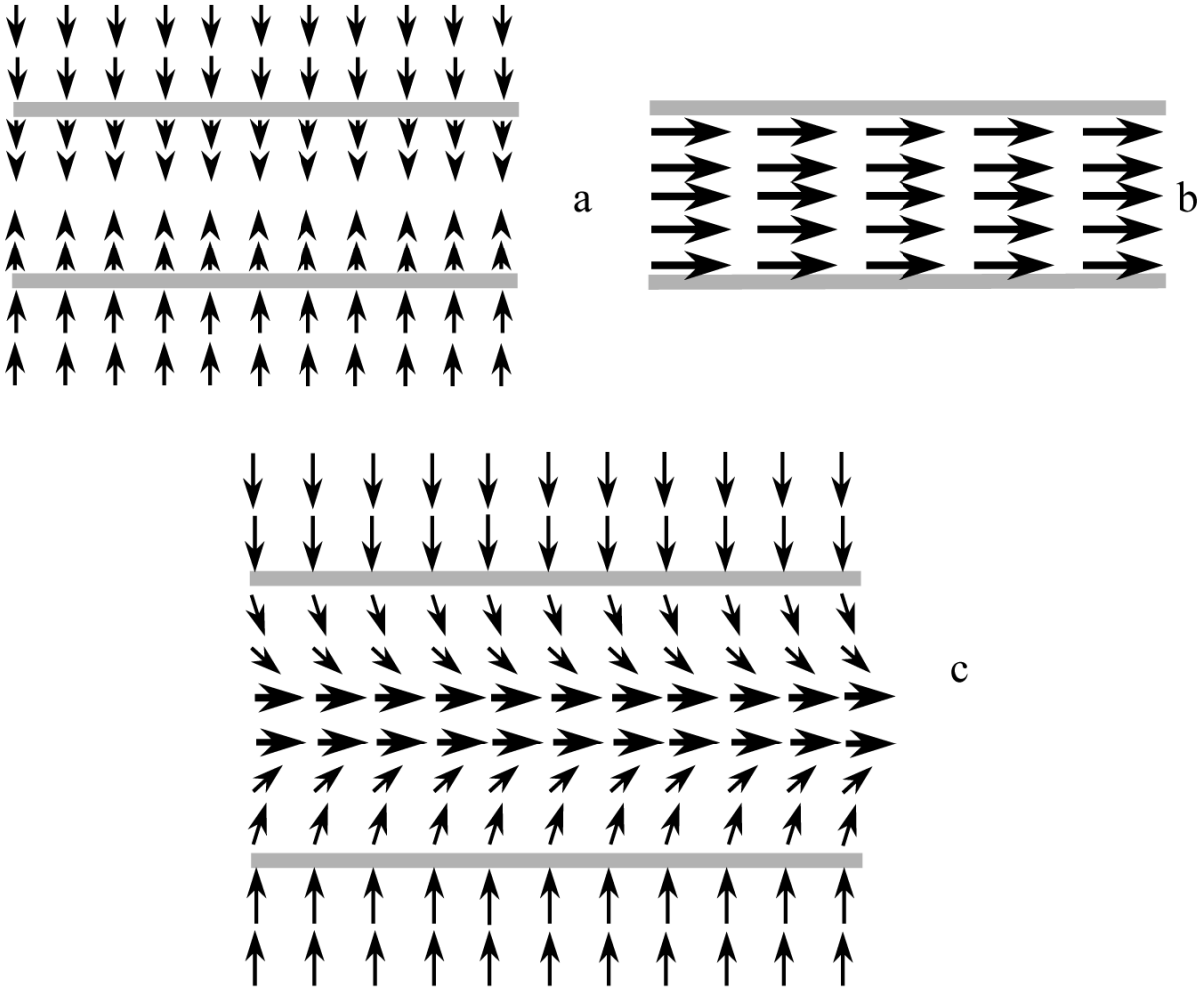


ROBOT'S PATH

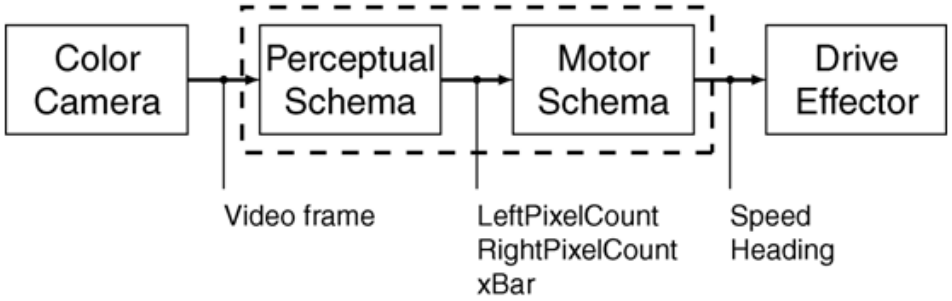
- If the robot starts at the location this is the path it takes
- The robot only feels the vector at the current location and then moves
- The robot never computes the field of vectors only the local vector

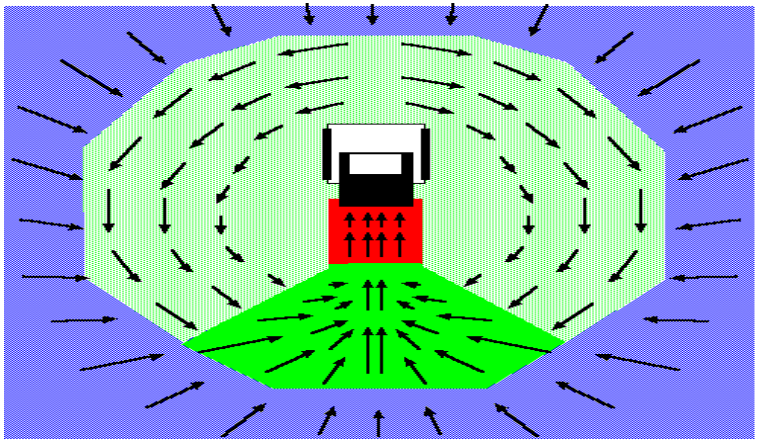


FOLLOW SIDEWALK POTENTIAL FIELDS



DOCKING USING POTENTIAL FIELDS





•*Arkin and Murphy, 1990, Questa, Grossmann, Sandini, 1995, Tse and Luo, 1998, Vandorpe, Xu, Van Brussel, 1995. Roth, Schilling, 1998, Santos-Victor, Sandini, 1997*

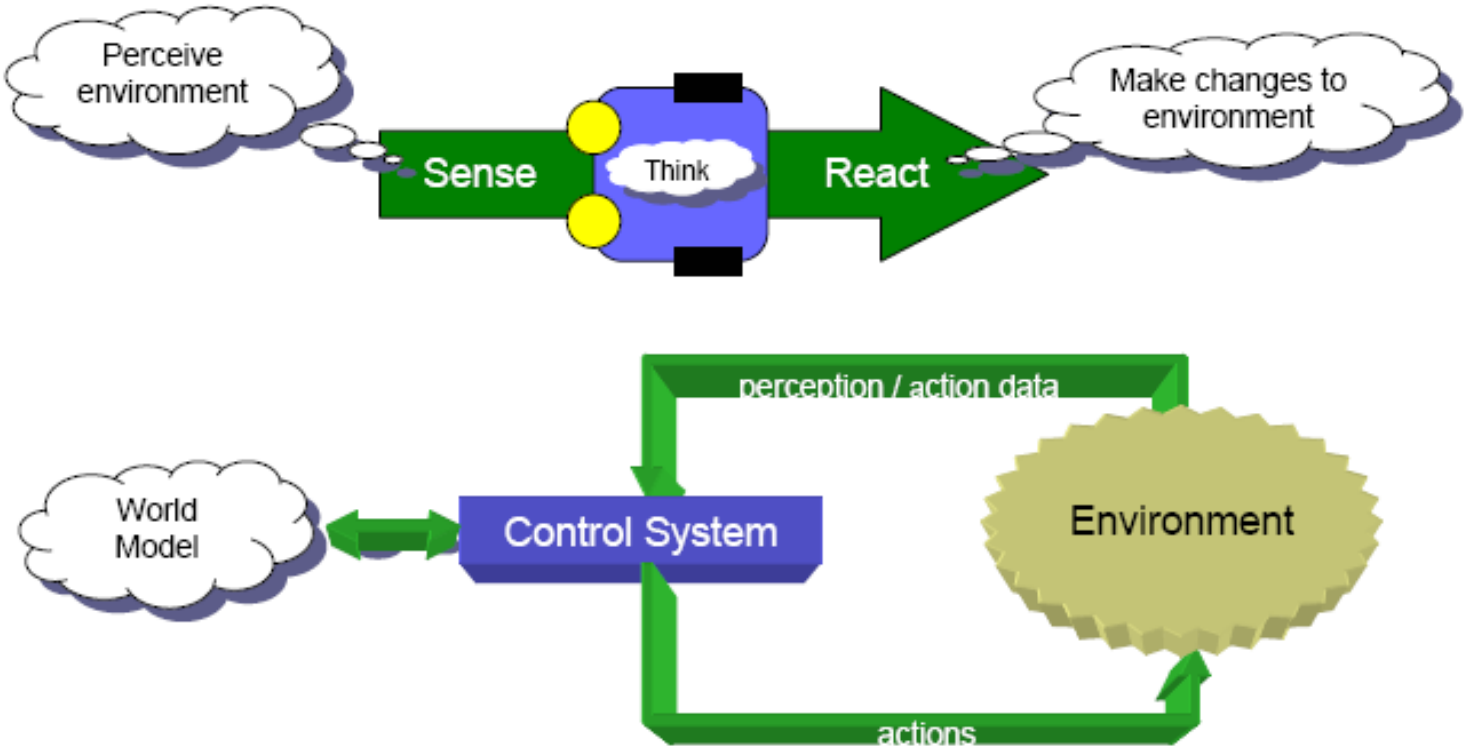
ADVANTAGES AND DISADVANTAGES

- Advantages
  - Potential fields are easy to visualize
  - Easy to build up software libraries
  - Fields can be parameterized
  - Combination mechanisms are fixed and tweaked with gains
- Disadvantages
  - Local minima problem if vectors sum to 0
  - Jerky motion

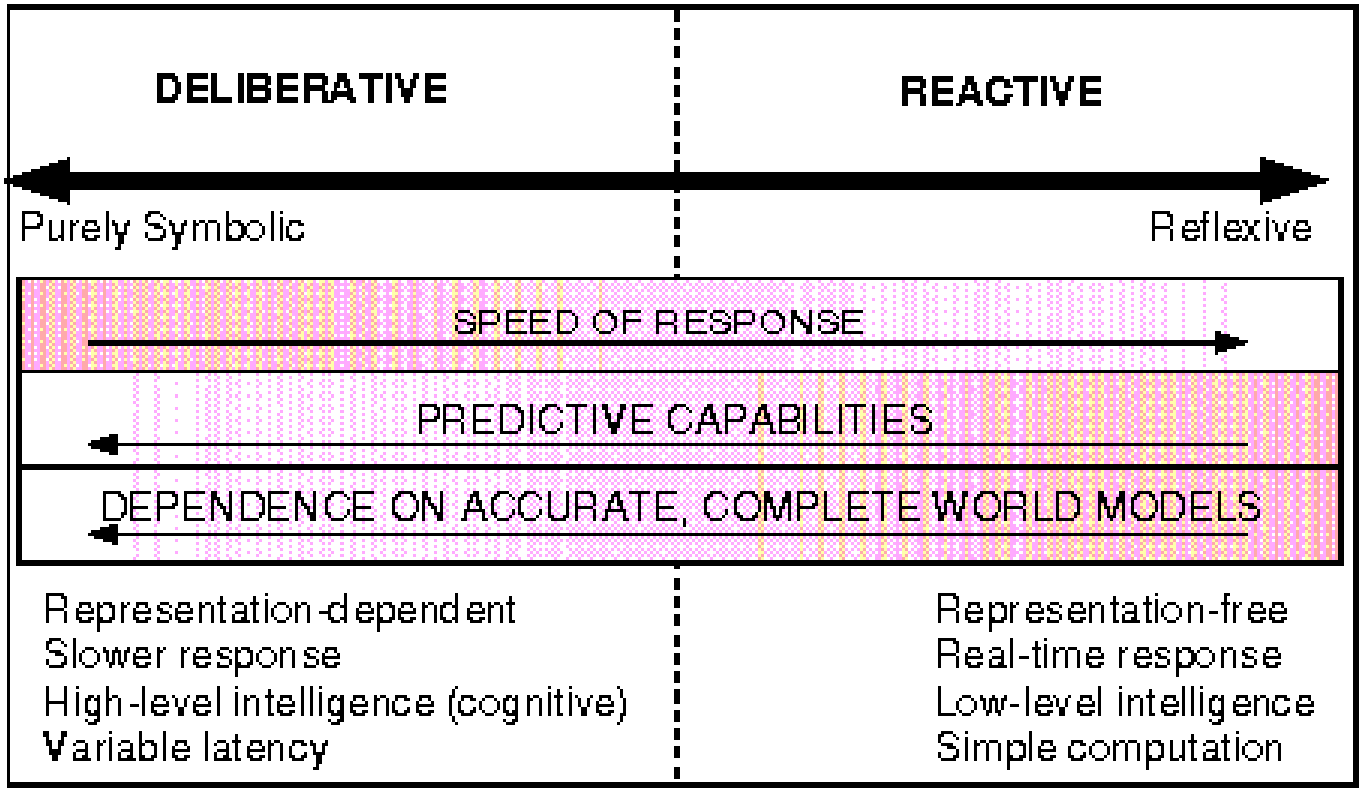
HIERARCHICAL VS. REACTIVE CONTROL

- Hierarchical (deliberative) control *(Think hard then act)*
  - *Sense → Plan → Act*
  - Involves planning to avoid bad solutions
  - Flexible for increasing complexity
  - It is slow and speed decreases with complexity
  - Requires large amounts of accurate information
- Reactive Control *(Don't think, react)*
  - *Sense → Act*
  - Fast, regardless of complexity
  - Built-in or learned from looking in the past
  - Limited flexibility for increased complexity

HIERARCHICAL VS. REACTIVE CONTROL



SPECTRUM OF ROBOT CONTROL



From "Behavior-Based Robotics" by R. Arkin, MIT Press, 1998

HYBRID CONTROL

THINK AND ACT INDEPENDENTLY AND CONCURRENTLY

- *Plan, Sense $\rightarrow$ Act*
- Combination of reactive and deliberative control
- Reactive layer is the bottom level and deals with immediate reaction
- Deliberative layer is the top level and creates plans
- The middle layer connects the two layers
- Typically called the “three-layer system”
- Reactive and deliberative layers have different time-scales and representations (signals, symbols)
- Hybrid control is one of the two dominant control paradigms in robotics

BEHAVIOR-BASED CONTROL (BBC)

THINK THE WAY YOU ACT

- *Sense* $\rightarrow$ *Act*
- Inspired by biology and has the same capabilities as hybrid control
- BBC acts reactively and deliberately
- BBC is built from layers but no intermediate layer
- BBC has a uniform representation and time-scale
- ***Behaviors*** are concurrent processes that take inputs from sensors and other behaviors and send outputs to the robot's actuators or other behaviors to achieve some goals
- Thinking performed through a network of behaviors and uses distributed representations
- Responds in real-time (reactive) and allows for a variety of behavior coordination mechanisms

CLASSICAL VS. BEHAVIOR-BASED CONTROL