Reactive Paradigm: Basics

- Based on ethology
- Vertical decomposition, as opposed to horizontal decomposition of hierarchical model

- Primitive behaviors at bottom
- Higher behaviors at top
- Each layer has independent access to sensors and actuators, or shares sensors with other behaviors

- Complex behavior emergent from simpler ones
Reactive Paradigm: Basics (2)

- Behavior is direct mapping of sensory inputs to pattern of motor action used to achieve a goal
  - All actions are result of behavior
- PLAN plays no role
- Behavior consists of
  1. At least one perceptual schema
     - Input comes from a dedicated sensor, or one shared with other behavior
     - If shared, access is independent
     - May be affordance-based, or process input
     - Any memory used is local to schema
  2. At least one motor schema
- Result of this architecture is real-time operation
- General characteristics of robots based on Reactive paradigm:
  1. They are situated agents that operate in an ecological niche
  2. Behaviors are basic building blocks; behavior is emergent
  3. Only local, behavior-specific sensing is allowed
  4. Naturally follow good software design principles
  5. Are biology based
- Advantages of Reactive paradigm
  1. Inherently modular
  2. Real-time operation
  3. Enable libraries of behaviors
- Paradigm must provide mechanisms for
  1. Triggering behaviors
  2. Handling multiple active behaviors
- Formalized approaches to Reactive paradigm:
  1. Subsumption
     - Hardware-based
  2. Potential fields
     - Software-based
Reactive Paradigm: Subsumption

- Due to Brooks
- Behavior modeled as network of sensing and acting modules
  - Implemented as Augmented Finite State Machines
  - AFSM is interface between schemas and coordinated control strategy of behavioral system
- Subsumption behavior is combination of one or more schemas into an abstract behavior
- No external program explicitly coordinates behaviors
  - Behavior is stimulus-response
- Control results from following
  - Modules grouped into layers of competence
    * Lower are primitive, higher are more abstract
  - Modules in higher layers can subsume lower ones
  - No internal state
  - Goals achieved by activating appropriate layers
  - Correlation of behaviors achieved by
    1. Inhibition
      * Output of subsuming module connected to output of subsumed one
      * Will block output of subsumed module
    2. Suppression
      * Output of subsuming module connected to input of subsumed one
      * Will replace input of subsumed module
Reactive Paradigm: Subsumption Behaviors

1. **Run-away**
   - Approaching objects result in flee behavior
   - Range finders arrayed around entire robot
   - Modules:
     (a) *Feel Force*
     - Finds vector sum of ranges
     - If closeness threshold is reached, move in opposite direction of strongest vector
     (b) *Runaway*
     - Receives vector from Feel force, generates heading
     (c) *Turn*
     - Activates steering actuator
     (d) *Forward*
     - Controls motor
     - Uses heading magnitude for control
     (e) *Collision*

Subsumption: Run Away

- Above can be represented in terms of behaviors:
  - *Feel Force*: perceptual schema for Runaway
  - *Runaway*: motor schema for Runaway
  - *Collide*: perceptual and motor schemas for Collide
2. Exploration

- Robot randomly moves around environment, avoiding obstacles
- Modules:
  (a) *Wander*
    - Generates random heading
  (b) *Avoid*
    - Combines vectors from *Wander* and *Feel Force*

**Subsumption: Wander**

```
SONAR -> FEEL FORCE
<table>
<thead>
<tr>
<th>polar plot</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLLIDE</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>TURN</td>
</tr>
<tr>
<td>heading</td>
</tr>
<tr>
<td>force</td>
</tr>
<tr>
<td>modified heading</td>
</tr>
<tr>
<td>force</td>
</tr>
<tr>
<td>heading</td>
</tr>
<tr>
<td>encoders</td>
</tr>
<tr>
<td>heading</td>
</tr>
<tr>
<td>halt</td>
</tr>
</tbody>
</table>
```

**Subsumption: Wander**
Subsumption: Wander as Behaviors

- **WANDER**
  - **desired heading**
- **AVOID**
  - **safe heading**
- **RUN AWAY**
- **COLLIDE**
  - **halt**
- **FORWARD**

**SONAR**
Reactive Paradigm: Subsumption Behaviors (2)

3. Follow corridor

- Robot moves down center of hallway
- Modules:
  (a) *Look*
  - Determines center of corridor using range finders
  (b) *Integrate*
  - Keeps robot on course
  - Monitors distance and direction traveled since last *Look* update
  (c) *Stay in Middle*
  - Keeps robot in center
  - Calculates vector sum of old *Look* vector and vector generated by *Stay in Middle*

Subsumption: Follow Corridor
Reactive Paradigm: Subsumption Behaviors (n)

(d) Problems:
  i. Integrate overrides real-world observations
  ii. If Look doesn’t update, motion based on Integrate, which may have error

Solution:
  – Impose time limits on suppression and inhibition
  – If in effect for an arbitrary limit, turn off and allow lower-level behaviors to take charge
Reactive Paradigm: Summary of Subsumption

- Defines behavior as tight coupling of SENSE-ACT pairs
  - Not schema based
  - Modules grouped into layers of competence
- Higher layers may subsume (inhibit/suppress) lower layers
  - Lower layers always active
- Design and implementation not algorithmic
- No planning involved
- No world model involved
- Behaviors usually released by affordances
- Perception is egocentric and distributed
Reactive Paradigm: Potential Field Approach Overview

- Behaviors represented by vectors
- Emergent behavior is vector sum of individual behavior vectors
- **Potential field:** array (field) of vectors at a given point in time
  - Vectors associated with goals and objects in world
  - Each entry represents force affecting robot at a given location from a given source
  - Represents (usually) 2D region of space
  - Controls motors

- Types of fields:
  1. Uniform
     - **Uniform Field**
     - \[ \rightarrow \rightarrow \rightarrow \]
     - \[ \rightarrow \rightarrow \rightarrow \]
     - \[ \rightarrow \rightarrow \rightarrow \]
  2. Perpendicular
     - **Normal Field**
     - \[ \uparrow \uparrow \uparrow \uparrow \]
     - \[ \uparrow \uparrow \uparrow \uparrow \]
     - \[ \uparrow \uparrow \uparrow \uparrow \]
     - \[ \uparrow \uparrow \uparrow \uparrow \]
  3. Attractive
     - **Attractive Field**
     - \[ \nabla \nabla \nabla \nabla \]
     - \[ \rightarrow \rightarrow \bullet \rightarrow \]
     - \[ \rightarrow \rightarrow \rightarrow \rightarrow \]
     - \[ \uparrow \uparrow \]
4. Repulsive
   Repulsive Field
   
5. Tangential
   Tangential Field
Reactive Paradigm: Potential Field Approach Overview (3)

- **Magnitude profile** of field
  - Change of magnitude as function of location
  - Types:
    1. Constant: Uniform speed while in range of field
    2. Linear: Speed changes uniformly in relation to distance from object
    3. Exponential: Speed changes by factor of 2 per unit distance from object

- Magnitude can also depend on stimulus strength (independent of distance)
- Potential field not maintained internally (no world model)
typedef struct vector {
    double magnitude;
    double direction;
} vector;

vector repulsive (double d, double max) {
    vector v;
    if (d < DMAX) {
        v.direction = -180.0;
        v.magnitude = (max - d)/max;
    } else {
        v.direction = 0.0;
        v.magnitude = 0.0;
    }
    return v;
}

vector runaway () {
    double reading;
    vector v;
    reading = readSonar();
    v = repulsive (reading, DMAX);
    return v;
}

...
Reactive Paradigm: Potential Field Approach: Combining Behaviors

- Simple combination: Single goal, single obstacle
  - Move2Goal is attractive, range is infinite
  - Runaway is repulsive, range is finite
  - Magnitude of resultant vector controls speed
  - Robot may not follow absolute force contours
    Distance traveled in a given time interval depends on vector magnitude

- One behavior per sensor
Reactive Paradigm: Comparison of Potential Field and Subsumption Approaches

1. Level 0: Obstacle avoidance (herding)
   - Potential field generates vector sum of each sensor
   - Does not use Collide (If included, generates null vector)
   - If actual collision occurs, means Runaway has failed - treat as panic mode

2. Level 1: Wander
   - Generate uniform field for set time interval
   - Does not use Avoid; result combined with Runaway

3. Level 2: Follow Corridor
   - Requires only 2 behaviors: Runaway and Follow Corridor
   - Follow Corridor consists of
     (a) 2 repulsive fields perpendicular to walls that linearly decrease towards center
     (b) Uniform field parallel to walls

```
<table>
<thead>
<tr>
<th>Normal Field</th>
<th>Uniform Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>↓↓↓↓↓↓↓↓</td>
<td>→→→→→→→→</td>
</tr>
<tr>
<td>↑↑↑↑↑↑↑↑</td>
<td>→→→→→→→→</td>
</tr>
</tbody>
</table>

Resultant Field
```

- Does not use Wander
4. Docking behavior

- Robot must align with specific part of object
- Field consists of:
  (a) Attractive field to guide robot to object
  (b) Selective attractive field to guide robot to specific part of object
  (c) Tangential field to guide robot to selective attractive field
Reactive Paradigm: Potential Field Evaluation

• Positives:
  – Easy to visualize
  – Easy to combine fields
  – Can parameterize fields (range of effect, magnitude, etc.)

• Negatives:
  – Vector sum may equal 0
    * Possible solutions:
      1. Motor schema generates random noise vectors
      2. Navigation templates
        · Incorporate intelligence into Avoid behavior
        · **Strategic vector** = sum of all vectors except Avoid
        · Result is direction to go if no obstacle were present
        · Implement as tangential field around obstacle, direction based on robot’s position and strategic vector
      3. Express fields as harmonic function
        · Guaranteed to be non-null
Reactive Paradigm: Potential Field Summary

- Behavior consists of one or more motor and perceptual schemas and/or behaviors
  - Motor schema is potential field
- Behaviors operate concurrently, output vectors summed
  - No layers
- Designer responsible for control program (for coordination)
- Perception based on affordances
- Perceptions can be shared among behaviors
- A priori knowledge can be supplied to perceptual schemas to emulate specialized sensors
Reactive Paradigm: Evaluation

- Subsumption and Potential Field approaches essentially equivalent
- Modularity: Supported by both (subsumption in hardware, potential fields in software)
- Niche targetability: High for both
- Ease of portability:
  - Not portable to environments requiring planning
  - Usually upper levels require modification based on task
- Robustness: No mechanism for id’ing failure of higher-level modules