Hybrid Model: Overview

- 1990’s saw evolution of architectures labeled *reactive planning*
  - Developed in response to shortcomings of Reactive approach:
    * Could not deal with problems that require cognitive activities (planning, learning, problem solving)
      - e.g., optimal paths, map making, self-monitoring
    * Design is not straightforward
  - Now known as the Hybrid deliberative/Reactive Model

- Hybrid approach combines best of Hierarchical and Reactive models:
  1. Slow deliberative modules function asynchronously of faster reactive modules
  2. Modular design enables swapping of modules based on application

- Hybrid model naturally composed of 2 components:
  1. Deliberative
    - Performs all cognitive tasks - not just path planning
  2. Reactive

![Diagram of plan-act cycle]

- Advantages to separating deliberative from reactive:
  1. Deliberative deals with global knowledge:
    - Not useful for micromanagement
    - Uses symbolic processing
      * Reactive uses sensors and effectors
  2. Planning and world modeling are computationally complex
    - Reactive is real-time
Hybrid Model: Overview (2)

- General algorithm:
  1. Create a plan
  2. Unleash behaviors to execute the plan

- Sensing is more complex than in Reactive Model:
  1. Sensors local to behaviors
  2. World model may share sensor input with behaviors
  3. World model may have its own dedicated sensors
  4. World model may have access to output from perceptual schemas (eavesdropping)
  5. Behavior may use world model as virtual sensor
Hybrid Model: Overview (3)

• Minor nuances between Hybrid and other models:
  1. Term ”behavior” used more generally than in Reactive Model
     – Can include reflexive, innate, and learned behaviors
     – Sometimes referred to as a skill instead of behavior
  2. Tends to use sequences of behaviors, rather than primitive behaviors
  3. World model contains egocentric knowledge
     – Used for
       (a) Behavior management
           * Selecting which behaviors to use
       (b) Performance monitoring
           * Determining whether making progress towards goal
Hybrid Model: Organization

• Hybrid architectures categorized on following points:

1. How does model differentiate between reaction and deliberation?
   – Affects how functionality is distributed among modules
   – What modules have access to world model
   – What kinds of knowledge belong in world model
2. How does model organize responsibilities in deliberative part?
   – Modularity supports portability and reuse
3. How does overall behavior emerge?
   – Subsumption?
   – Potential fields?
   – Voting?
   – Fuzzy logic?
   – Filtering?

• Modules generally supported by all Hybrid architectures:

1. Sequencer
   – General tasks:
     * Generates sets of behaviors needed to achieve a goal
     * Sequences behaviors
     * Ids activation conditions
   – Often implemented in terms of a dependency network or finite state machine
2. Resource manager
   – Allocates resources to behaviors
3. Cartographer
   – General tasks:
     * Handles mapping
     * Manages spatial information
   – Usually contains world model
Hybrid Model: Organization (2)

4. Mission planner
   – Interface between controller and robot
   – General tasks:
     * Receives task from person
     * Builds missions in robot-centric terms

5. Performance Monitor and Problem Solver
   – Actual planner
   – Requires self-awareness
Hybrid Model: Architecture Organization

1. Managerial
   • Evolved directly from Reactive Model
   • Considered bottom-up approach
     – Behaviors are basic components
   • Modules layered
   • Each layer responsible for finer grain of detail
   • Highest level purely deliberative
   • Lowest level reactive
   • Level can only modify level immediately below
   • If module cannot perform task, asks for help from level above
     – Called *failing upwards*

2. State hierarchies
   • Evolved directly from Reactive Model
   • Also considered bottom-up approach
   • Module category based on robot state
   • States based on time:
     – Past
     – Present
     – Future
   • Reactive modules act only in Present state
     – No self-awareness
     – No knowledge of past or future
   • Deliberative modules access Past, Present, and Future states
   • Organized in levels in same manner as Managerial approach
Hybrid Model: Architecture Organization (2)

3. Model-oriented

• Very similar to Hierarchical
• Based on ”traditional” AI planning theory
• Top-down paradigm
• Behaviors have access to world model
• Based on symbolic manipulation of world model
• World model can act as virtual sensor to behaviors
• What differentiates from Hierarchical model:
  (a) World model more general, less detailed
  (b) Uses distributed processing
      – Slow deliberative modules run asynchronously from faster reactive ones
  (c) Uses sensor fusion to handle failure and uncertainty
• NOTE: Faster CPUs have alleviated processing bottleneck that once existed
Hybrid Model: Managerial Model Example - Autonomous Robot Architecture (AuRA)

- Based on schema theory
- Three levels:

1. Deliberative component
   (a) Cartographer
      - Generates maps
      - Navigates
      - Can accept *a priori* maps
Hybrid Model: Managerial Model Example - Autonomous Robot Architecture (AuRA) (2)

(b) Planner - plans path
   – Consists of three submodules:
     i. Mission planner
        * Interfaces with people
     ii. Navigator
        * Plans path as sequence of subtasks
        * Interfaces with Cartographer
     iii. Pilot
        *Ids behaviors needed
        * Generates info needed for subtask
        * Passes info to Motor Schema Manager

2. Reactive component
   (a) Sensor system
      – Contains perceptual schemas
      – Uses potential field representation
   (b) Motor system
      – Contains motor schemas
      – Motor Schema Manager
        * Generates behaviors
        * Selects appropriate schemas for subtask

3. Homeostatic system
   – Neither purely deliberative nor purely reactive
   – Modifies relations between behaviors by modifying gains
   – Models relative priorities of behaviors, based on situation
Hybrid Model: Managerial Model Example - Autonomous Robot Architecture (AuRA) (3)

- Comparison of AuRA with respect to common components of emergent behavior:

<table>
<thead>
<tr>
<th>Component</th>
<th>AuRA Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequencer</td>
<td>Navigator, Pilot</td>
</tr>
<tr>
<td>Resource Manager</td>
<td>Motor Schema Manager</td>
</tr>
<tr>
<td>Cartographer</td>
<td>Cartographer</td>
</tr>
<tr>
<td>Mission Planner</td>
<td>Mission Planner</td>
</tr>
<tr>
<td>Performance Monitoring Agent</td>
<td>Pilot, Navigator, Mission Planner</td>
</tr>
<tr>
<td>Emergent behavior</td>
<td>Vector summation</td>
</tr>
<tr>
<td></td>
<td>Spreading activation of behavior</td>
</tr>
<tr>
<td></td>
<td>Homeostatic control</td>
</tr>
</tbody>
</table>
Hybrid Model: Managerial Model Example - Sensor Fusion Effects Architecture (SFX)

- Original intent was to simulate biological sensing pathways to handle fusion and sensor failure
  - In cats, sensor processing local to each sensor
  - Sensor information pathway branches at higher level:
    * To *superior collicus*, which handles motor actions
    * To cerebral cortex, which handles cognitive functions
  - SFX duplicates this in terms of reactive and deliberative layers, respectively

- Components:
Hybrid Model: Managerial Model Example - Sensor Fusion Effects Architecture (SFX) (2)

1. Deliberative component
   – Modules are separate and independent software agents
   – Each works on a specific aspect of the current task
   – Goal is to achieve high degree of modularity and orthogonality among them
   – Two layers:
     (a) Mission Planner
        * Constitutes the top layer
        * Interacts with people
        * Specifies problem constraints for lower-level agents
     (b) Resource management
        * Handled by
          i. Task Manager
          ii. Sensing Manager
          iii. Effector Manager
        * These agents may not relax constraints imposed by Mission Planner
        * These collectively determine best allocation of resources given a set of perceptual and motor schemas for a behavior
     (c) Sensing Manager
        * Consists of software agents for
          i. Monitoring task performance
          ii. Monitoring whether environment has changed
        * Its purpose is to find alternative perceptual schemas or behaviors in the event of sensor failure
        * This allows robot to continue with given task without coming to a halt
     (d) Cartographer
        * Handles map-making and path planning
     (e) Performance agents
        * Monitor progress fo robot toward goal
        * If no progress observed, inform agent in upper level
2. Reactive component
   – Behaviors combined using \textit{filtering}
     * Similar to subsumption, but lower-level behaviors override higher-level ones
   – Two layers:
     (a) Strategic
         * High-level
     (b) Tactical
         * Low-level
         * Deal with immediate issues
   – Behaviors influence each other by influencing outputs of others similar to subsumption

• Comparison of SFX with respect to common components of emergent behavior:

<table>
<thead>
<tr>
<th>Component</th>
<th>SFX Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequencer</td>
<td>Task Manager</td>
</tr>
<tr>
<td>Resource Manager</td>
<td>Sensing and Task Managers</td>
</tr>
<tr>
<td>Cartographer</td>
<td>Cartographer</td>
</tr>
<tr>
<td>Mission Planner</td>
<td>Mission Planner</td>
</tr>
<tr>
<td>Performance Monitoring Agent</td>
<td>Performance and Habitat Monitors</td>
</tr>
<tr>
<td>Emergent behavior</td>
<td>Strategic behaviors grouped into abstract behaviors or scripts</td>
</tr>
<tr>
<td></td>
<td>These filtered by tactical behaviors</td>
</tr>
</tbody>
</table>
Hybrid Model: State Hierarchy Example - 3-Tiered Architecture (3T)

- Components:

1. Planner
   - Top layer
   - Purely deliberative
   - Generates subtasks and coordinates mapping
   - Reasons about the future, using info about the past and present
Hybrid Model: State Hierarchy Example - 3-Tiered Architecture (3T) (2)

2. Sequencer
   – Middle layer
   – Receives subtasks from Planner
   – Reactive planner
     * Selects behaviors (skills) from library to achieve goal and sequences them to complete subtask
   – Monitors performance using info from past and present

3. Controller/Skill Manager
   – Lowest level
   – Reactive component
   – Event is checkpoint associated with skill used to monitor performance
   – Operates in present

- Organization actually based on update rates (as opposed to state):
  – Fast algorithms in Skill Manager
  – Slow in Planner

- Comparison of 3T with respect to common components of emergent behavior:

<table>
<thead>
<tr>
<th>Component</th>
<th>3T Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequencer</td>
<td>Sequencer</td>
</tr>
<tr>
<td>Resource Manager</td>
<td>Sequencer (Agenda)</td>
</tr>
<tr>
<td>Cartographer</td>
<td>Planner</td>
</tr>
<tr>
<td>Mission Planner</td>
<td>Planner</td>
</tr>
<tr>
<td>Performance Monitoring Agent</td>
<td>Planner</td>
</tr>
<tr>
<td>Emergent behavior</td>
<td>Behaviors grouped into skills</td>
</tr>
<tr>
<td></td>
<td>Skills grouped into task networks</td>
</tr>
</tbody>
</table>
Hybrid Model: Model-oriented Example - Saphira

- Based on following aspects considered essential for successful robot operation:

  1. Coordination
     - Of sensors and actuators
     - Of goals
  2. Coherence
     - Maintains global world models
  3. Communication
     - Between controller and robot

- Components:
Hybrid Model: Model-oriented Example - Saphira (2)

1. Deliberative
   – Distributed among independent software agents
   – Components:
     (a) Procedural Reasoning System - Lite
     * Converts natural language commands into navigation tasks and perceptual routines
     (b) Local Perceptual Space
     * World model

2. Reactive
   (a) Consists of behaviors
   – Access world model as a virtual sensor
   – Generate fuzzy rules that control actuators

• Comparison of Saphira and general Hybrid architecture:

<table>
<thead>
<tr>
<th>Component</th>
<th>Saphira Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequencer</td>
<td>Topological Planner, Navigation Tasks</td>
</tr>
<tr>
<td>Resource Manager</td>
<td>PRS-lite</td>
</tr>
<tr>
<td>Cartographer</td>
<td>LPS</td>
</tr>
<tr>
<td>Mission Planner</td>
<td>PRS-lite</td>
</tr>
<tr>
<td>Performance Monitoring Agent</td>
<td>PRS-lite</td>
</tr>
<tr>
<td>Emergent behavior</td>
<td>Behaviors fused with fuzzy logic</td>
</tr>
</tbody>
</table>
Hybrid Model: Evaluation of Hybrid Model

• Modularity
  – Very modular
  – Layered architecture, with modularized components within layers

• Niche targetability
  – Very
  – Can be used for situations in which planning and/or pure reaction are called for

• Robustness
  – Usually include modules to monitor performance

• Hybrid v Hierarchical Model
  1. Better reflects principles of software engineering
  2. Uses world model for purely symbolic purposes (???)
  3. Frame problem not as acute
     – Execution is reactive
     – While thinks in closed world, acts in open world
     – Hierarchical has problems when environment does not conform to world model
  4. Plans not complete to nth detail
     – Plans consist of subtasks
     – Each executed and evaluated
  5. Ethology-oriented

• Speed issues
  – Faster than Hierarchical
    * If plan fails in Hierarchical, complete new plan generated
    * Hybrid generates plan 1X
    * If problem arises with plan, reactive component deals with it
  – Can be faster than Reactive
    * Reactive does not necessarily generate behaviors that achieve goal by most direct sequence of actions
Hybrid Model: Interleaving of Deliberative and Reactive Control

- Goal seeking
  - Goals achieved by moving from \textit{waypoint} to \textit{waypoint} * (i.e., subgoal to subgoal)
  - If problem arises with plan, reactive component deals with it

- Virtual sensors
  - Deliberative component supplies info to reactive component
  - Form of selective attention

- Fault tolerance
  - Reactive component signals deliberative on sensor failure
  - Deliberative can generate alternative plan or behavior