Soar Technology

Thinking *inside* the box.

# TEAL

## Macro Learning for Robotic Command and Control

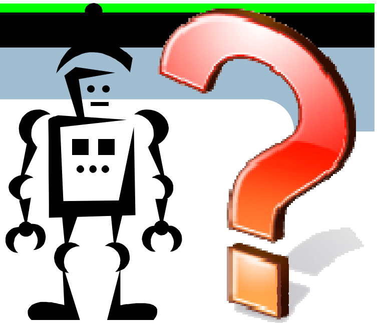
5 May 2008

Geoffrey Morgan, Bob Bechtel, Frank Koss, Steve Furtwangler
{gmorgan, bob.bechtel, koss, sfurtwangler}@soartech.com

# TEAL: Agenda

- The Problem
- Our Solution
- System Description
- Nuggets
- Coal
- Future Work

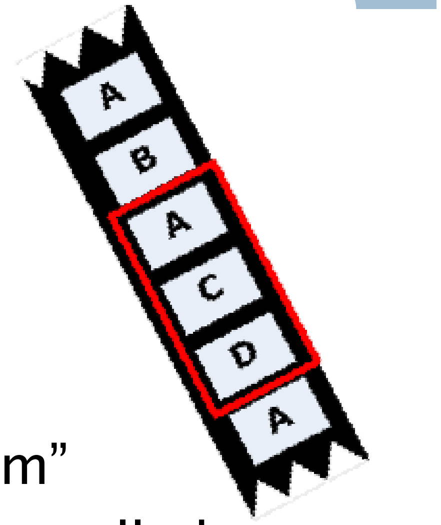
# TEAL: Problem



- The role robots will play in civilian and military markets is still emerging
- Complex robot behaviors are valuable for automating routine and boring tasks
- Complex behaviors can be highly contingent on purpose and environment – both of which may change over a system’s useful lifetime
- Identifying all complex behaviors needed to support users is a difficult (if not impossible) task
- Implementing complex behaviors is expensive

Complex behaviors are valuable and necessary, but expensive!

# TEAL: Our small slice of Solution



- Process:
  - Person uses basic robot capabilities, which are conceptualized as goals
  - As they use the robot, they create a “goal stream”
  - TEAL finds “interesting” segments of the stream, called ***sequences***
  - Sequences are presented to the user for validation
  - The user validates and names the new macro sequence, or rejects it
  - The newly named macro sequence becomes a user command
  - When the new command is used, the user specifies all required parameters up front, then the task is executed

# TankSoar Example

- User performs the actions “shoot missile” and “move backward” in a 1-2 sequence one or more times
- The SRR examines the goal stream and notices that “move backward” often follows “shoot missile”
- SRR generates a validation request to the user, asking if they want a macro command available that does {1-Shoot Missile, 2-Move Backward}
- User provides the name “Skirmish” and validates the sequence
- In the command list presented to the user, “Skirmish” is now available as a choice.
- When “Skirmish” is presented to the User, the Facilitator bundles up the Shoot Missile and Move Backward tasks in a sequential container and sends them to the execution system

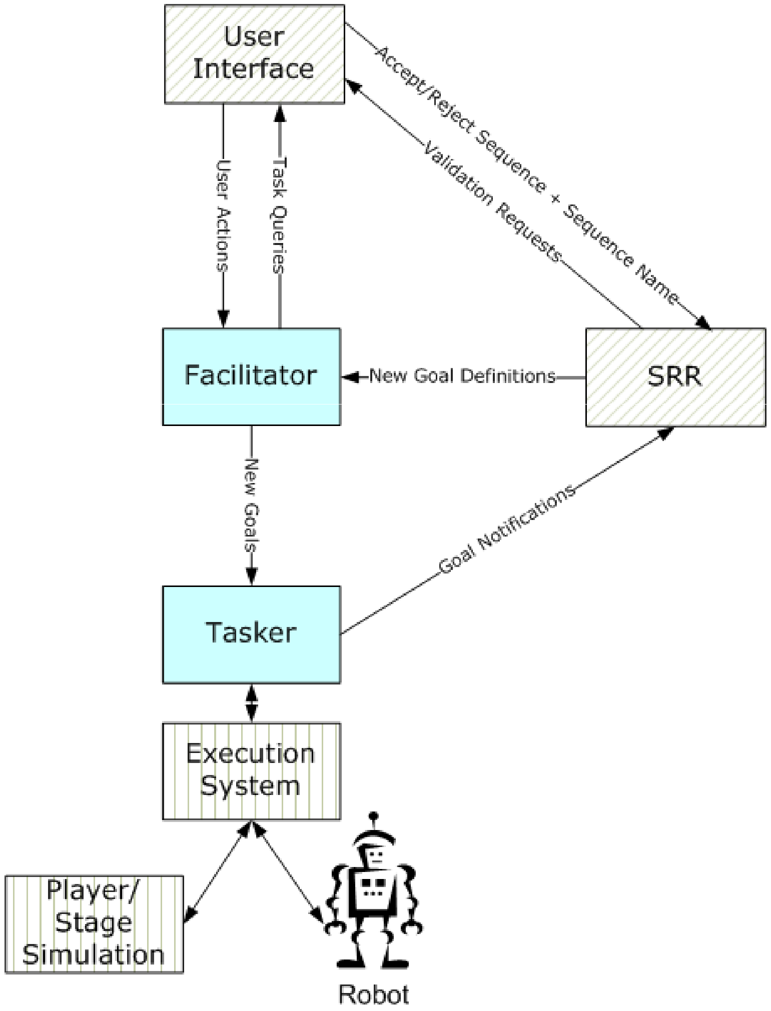
# TEAL:  Some Questions raised by our solution

- Do “goals” correspond to how a user thinks about their process?
  - We hope so.  We think most robot users have goals in mind that they’re attempting to solve with the robot.  However, these goals are typically imposed by the user and are not a basic unit of the control system.  In our system, the control system speaks in goals
- How do you define that a part of the goal-stream is “interesting”?
  - We use an algorithm to model an “attention” process – more on this later
- What happens if the goal-stream is noisy or error-prone?
  - We hope to find out soon.  If errors or noise are distributed randomly, a sufficiently conservative attention algorithm should cancel them out
- How will the user manage a growing list of commands?
  - A concern we plan to address in the next year

Soar Technology
Thinking *inside* the box.

# System Description

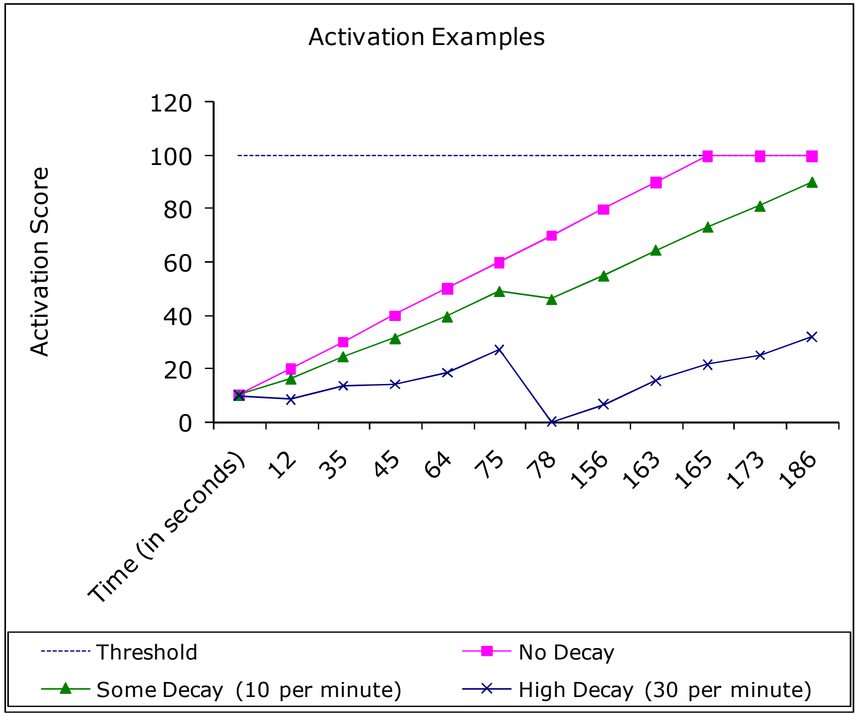
- User Interface – User interacts with System here
- Facilitator - A Soar agent that interprets user actions and generates goals for the Tasker
- SRR – The "Sequence Repetition Recognizer", implements an attention algorithm
- Tasker – A Soar agent that queues goals for execution



Soar Technology
Thinking *inside* the box.

# TEAL: The SRR’s attention algorithm

- We assume that things the user does in a particular order sufficiently frequently may be valuable to automate
- Every time a sequence (like {A, C, D} or {shoot-missile, move-backward}) occurs, it is ***activated***
- Activation level can ***decay*** based on time and the number of events that separate the last activation from the first
- Once the activation level reaches above the given threshold, the sequence is sent to the user for ***validation***



*Use ⇒ Value*

# TEAL – Learned Products

- TEAL generates new declarative knowledge (new goal definitions)
- These new goal definitions wrap existing primitives inside a sequential container
- This is different from Soar chunking – chunking produces new procedural knowledge

- TankSoar Example
  - <sequence>
    - <sequence-name> Skirmish </sequence-name>
    - <goal>
      - <order> 1 </order>
      - <type> shoot-missile </type>
    - </goal>
    - <goal>
      - <order> 2 </order>
      - <type> move-backward </type>
    - </goal>
  - </sequence>

# TEAL: Where are we?

- Attention algorithm and macro learning implemented
- Minimal user command validation interface in place
- Connection to simulated robot behavior (OTB and Player/Stage) demonstrated

# TEAL: Nuggets

- Examining a goal-stream to suggest new goals to the user seems to work
- Using a sequential container to bundle multiple primitive goals enables macro learning
- The approach is relatively generic to any set of goals that can be characterized

# TEAL: Coal

- Many robotic systems have no obvious goal set characterization
- Not sure yet what effect noisy data streams will have
- Do not have a good user interface that makes it easy and elegant to add new commands
- Context-free learning is much less powerful and interesting than contextual learning

# TEAL: Future Work

- Several “knobs” (time decay, sequence size, etc.) are available in the attention algorithm – characterize the space
- Learn macros involving multiple vehicles
- Add contextual information to both the automatic goal generation step and the sequence recognition step
- Move from simulation to the physical world - actual robots