SOAR2D

Jonathan Voigt
University of Michigan
Soar Workshop 28

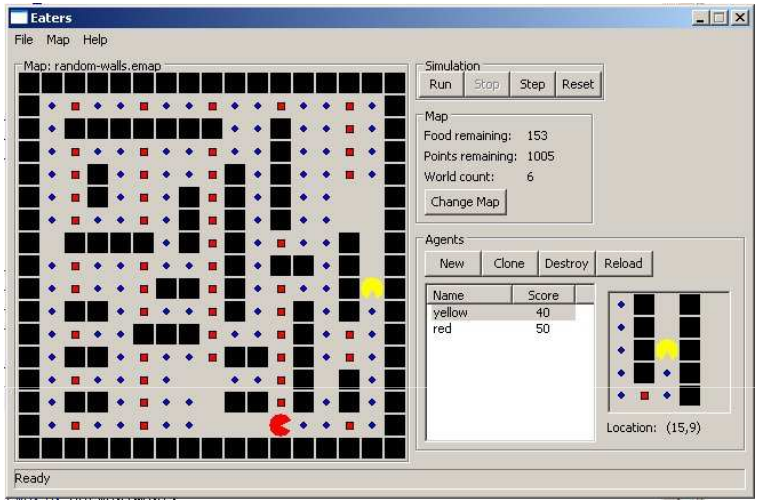
Outline

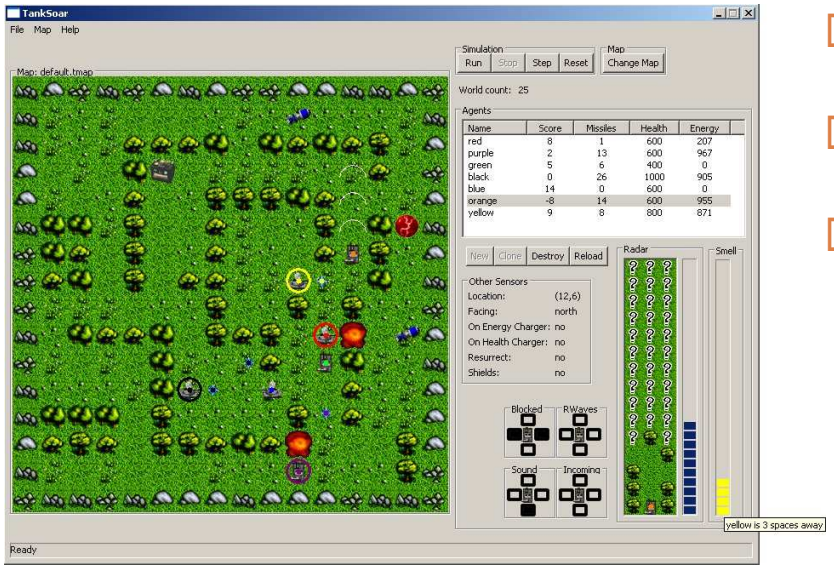
- About Soar2D
- What's new with Eaters and Tanksoar
- New Room and Taxi environments
- Soar2D features
- Future work

About Soar2D

- Cross-platform framework for building Soar simulations
  - Mainly turn-based, grid map simulations for instruction and research
- Initial motivations:
  - Reduce code duplication between Eaters and Tanksoar
    - Soar integration details
    - Configuration, logging, performance
  - Add new functionality to Eaters and Tanksoar for research
    - Data-driven game rules and objects so researchers can tweak them
  - Create new environments similar to Eaters and Tanksoar
    - And do it fast
  - No new prerequisites
    - Use same technology as debugger
- Included in Soar 8.6.4
  - More powerful and correct than versions of Java Eaters and Tanksoar included in Soar 8.6.3 and before

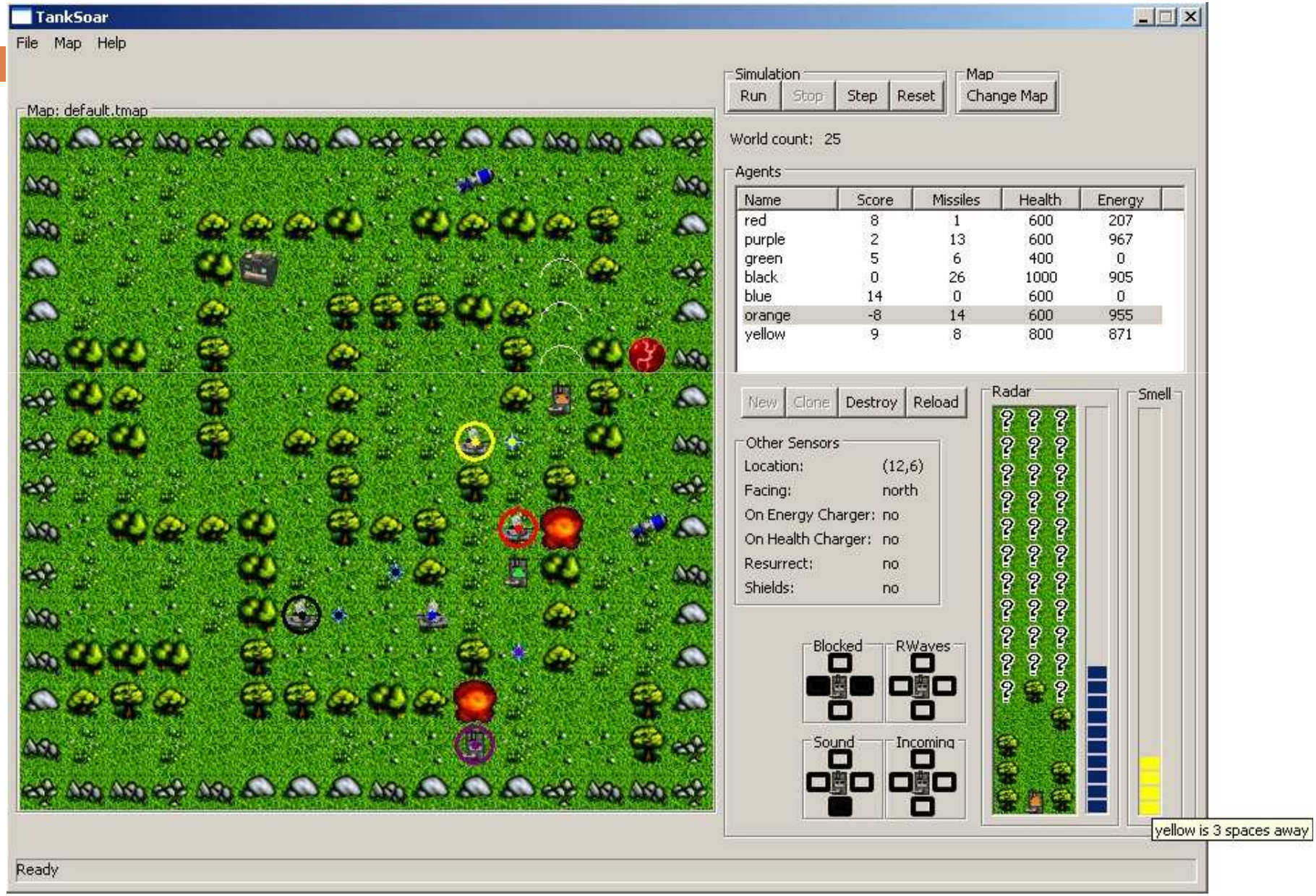
What's new in Eaters and Tanksoar



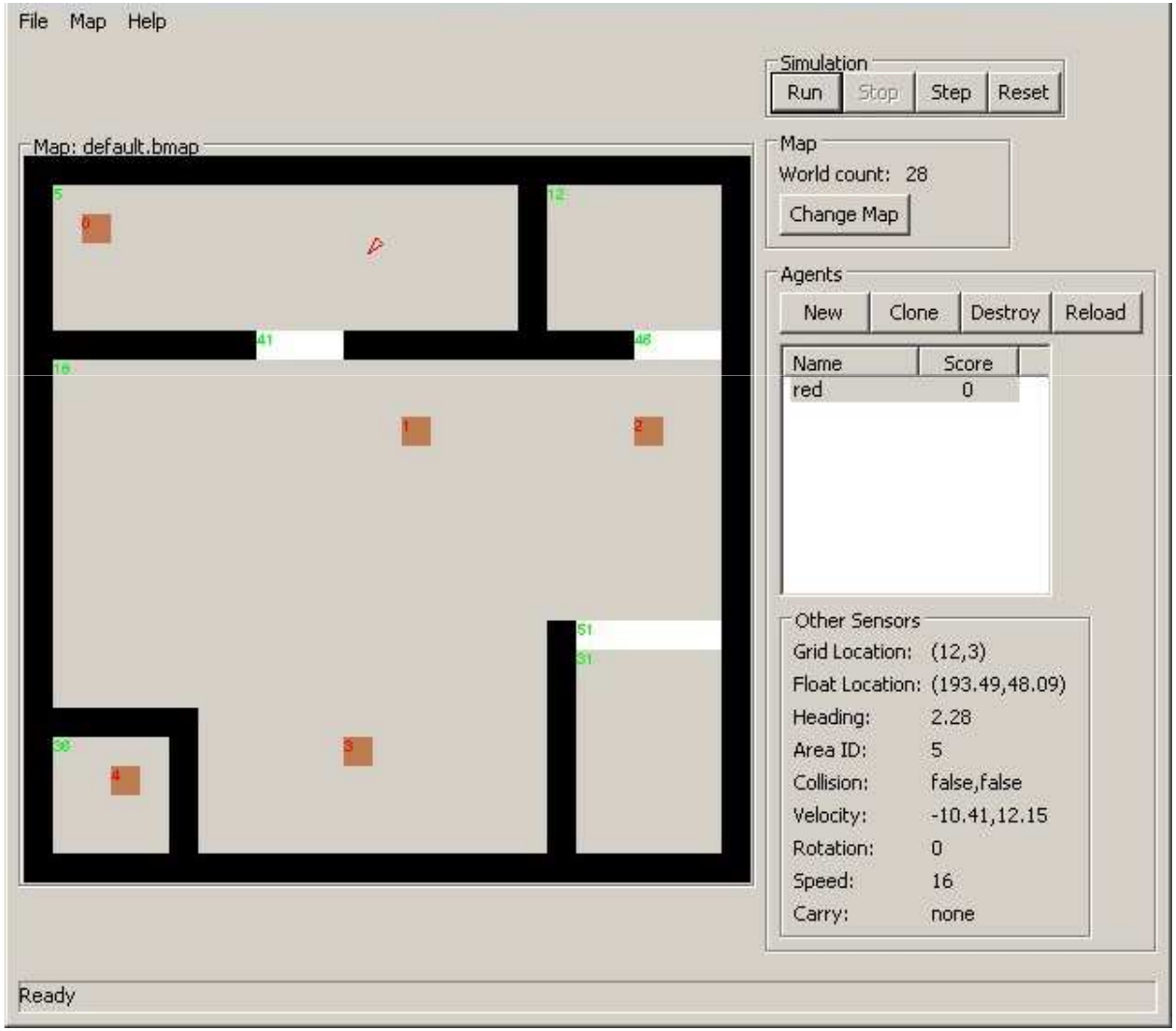


- Single window
- More agent information
- Easier human control of agents for debugging
  - Go 1-on-1 against your tank
- Configuration manager
- Map editor
- Tanksoar: Some new graphics and notations

What's new in Eaters and Tanksoar

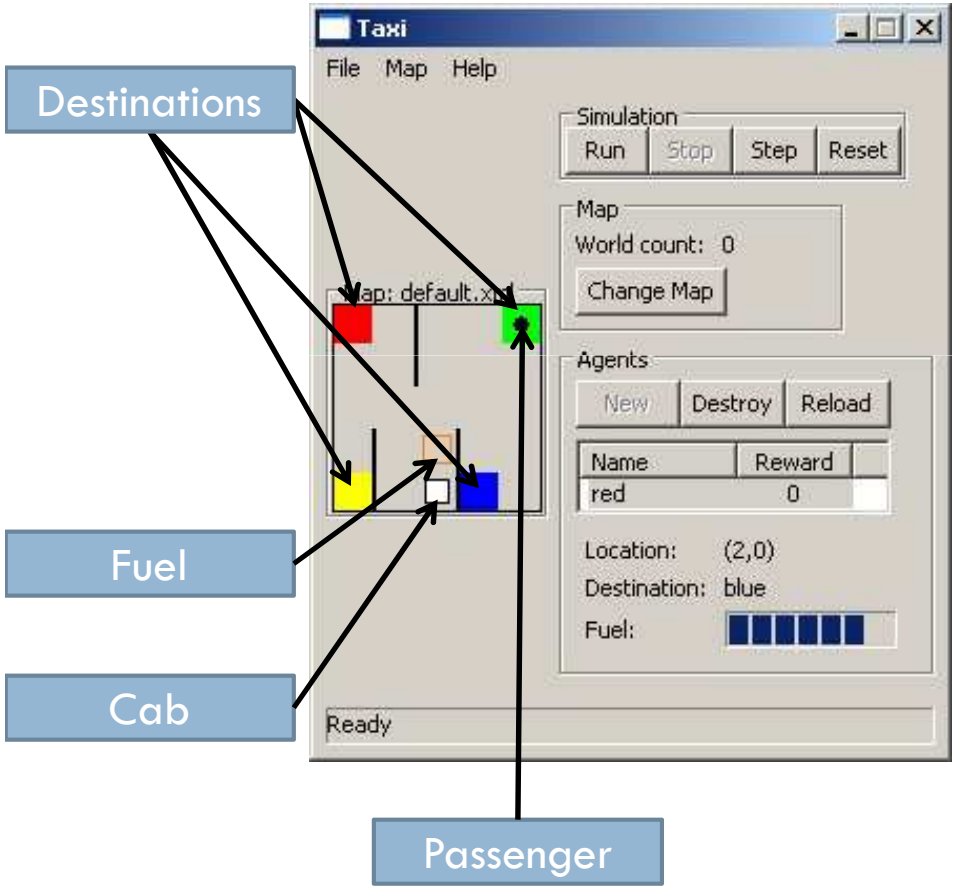


New: Room Environment



- Rooms with objects separated by gateways
- Agents move and rotate in continuous space
  - Very simple motion model
- Discrete mode available for more simple Eaters and Tanksoar-like movement
- Objects adhere to underlying grid squares
  - Agent can interact with objects, carry them
- Navigation aids to walls, objects and gateways
- Simple vision cone

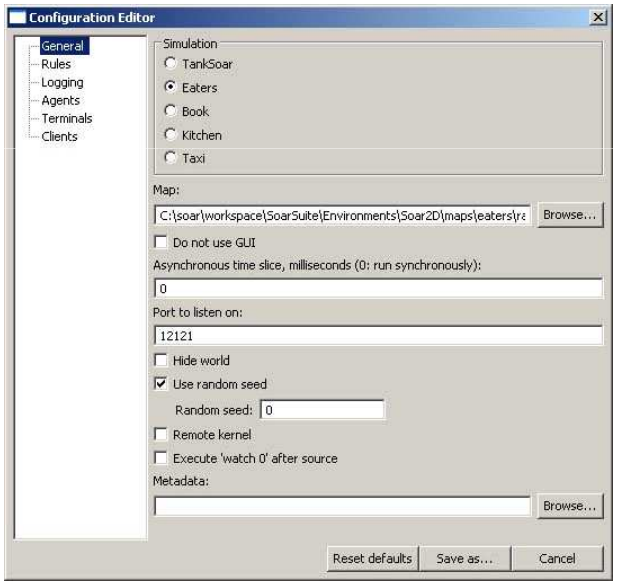
New: Taxi Environment



- Implementation of taxi environment described by MAXQ paper (Dietterich, 1998)
  - Used in Soar-RL research by Nate Derbinsky
- Taxi can move in four directions, pick up and drop off passenger, object is to successfully transport passenger to correct destination
- Uses one fuel unit each move

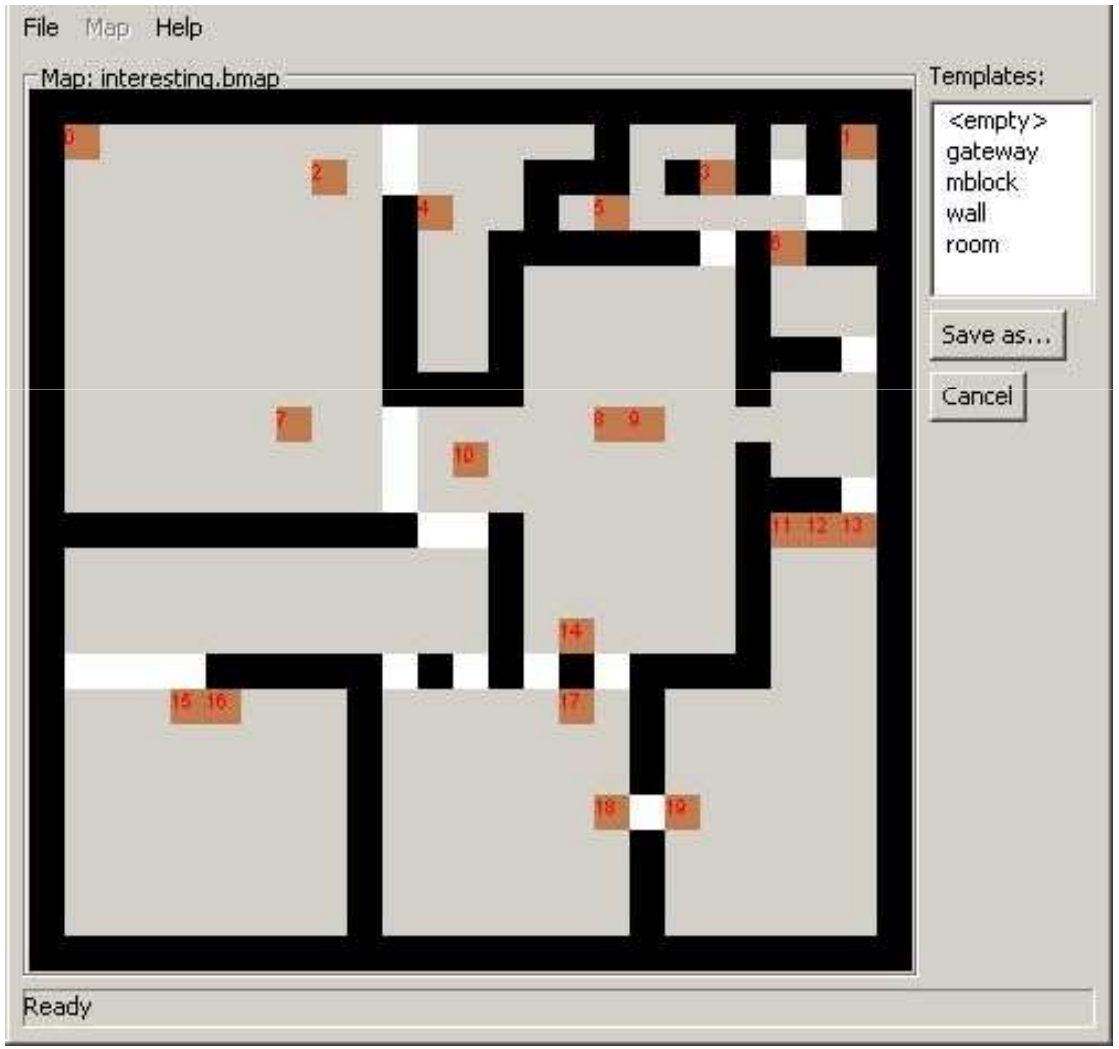
Configuration

- Files instead of command line options
- Many features configurable, including:
  - Initial state
  - Rule variations
  - Simulation modes
  - Termination conditions
  - Control of third-party clients
  - Logging options



- Configuration management inside Soar2D

Map Editor


File
Map
Help
Map: interesting.bmap
Templates:
<empty>
gateway
mblock
wall
room
Save as...
Cancel
Ready

Map File Format

- Section 1—Cell object list: Object classes
  - Define properties, for example:
    - ID to use on input link
    - Color, shape, value
  - Define behavior, for example:
    - Charging ability to energy charger
    - Flying, health and energy-modifying behavior to missiles
    - Consumption behavior to food and missile packs
  - Property and behavior flags trigger different parts of the code
  - Behavior can be specific to events, such as collisions (missiles, chargers, food) or world updates (food value decay, missiles flying)
- Section 2—Cells: Instantiate the objects
  - Flags available for random placement
- Section 3—Metadata: Optionally specify file with extra, arbitrary information to associate with map on the input link

Logging

- Goal: enough output to reproduce run
- Uses standard Java logging mechanisms
- Configurable logging levels and targets
- XML output possible

19 INFO orange: (move: north)(fire)

19 INFO yellow: (move: west)(fire)(shields: on)

19 INFO orange score: -3 -> -4 (yellow-41)

19 INFO yellow score: 0 -> 2 (yellow-41)

19 INFO orange score: -4 -> -5 (yellow-45)

19 INFO yellow score: 2 -> 4 (yellow-45)

19 INFO orange score: -5 -> -7 (fragged)

19 INFO yellow score: 4 -> 7 (fragged orange)

19 INFO orange: Spawning at (12,5), facing east

20 INFO red: (rotate: left)(radar: on)(radar-power: 3)(shields: off)

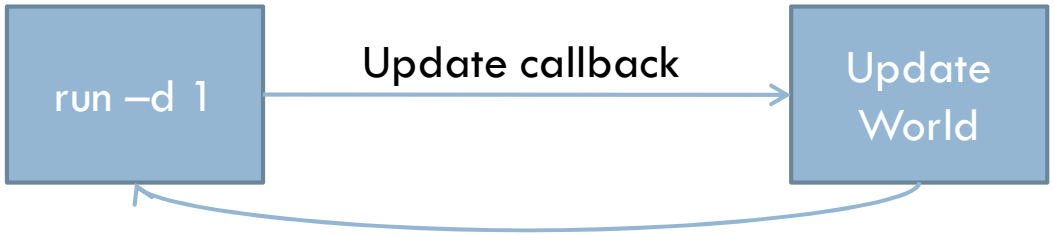
20 INFO purple: (move: east)

20 INFO green: (rotate: right)(radar: on)(radar-power: 4)

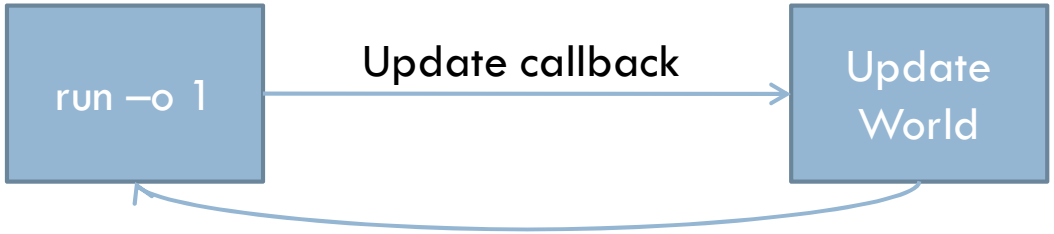
20 INFO black: (move: east)

Simulation Modes

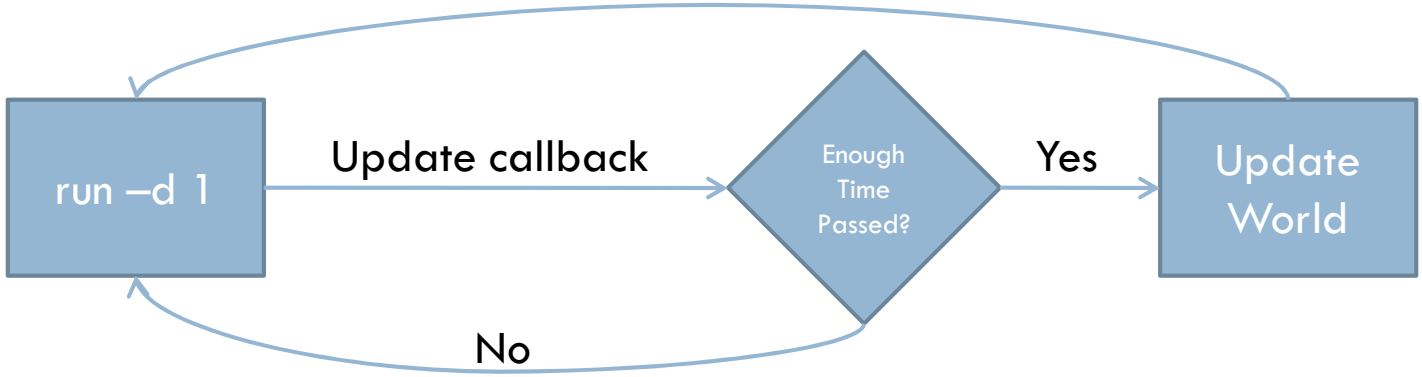
- By cycle



- By output



- By time slice



Headless Mode

- Skip GUI code to achieve maximum speed for experiments
  - Log provides output to files or console
- Built for use in conjunction with scripting environments such as Nate's SoarSim

Future Work

- Playback from log
  - Motivation: Make inspection of runs easier
- Increased performance
  - Motivation: Students running long experiments with it
  - Scott Wallace, “Is there any way to get it to run slower?”
- Code cleanup, bug fixes, modularization, documentation
  - Motivation: Easier for people not named Voigt to change the code fast
  - Waiting for current research to finish, fixes and docs coming this summer
- Better graphics
  - Motivation: Make it more interesting
- Better interface so other AI systems can use it
  - Motivation: Competition between cognitive architectures
  - Storm architecture successfully hacked in last year