SOAR ON SPLINTER

Jonathan Voigt

Soar Workshop 29

Goals

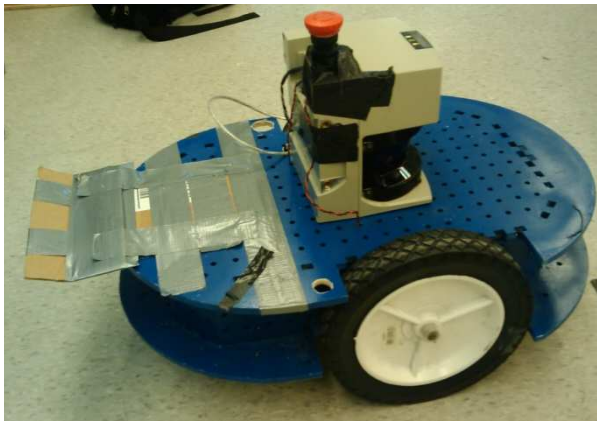
- Research in real-time Soar systems
- Soar as a controller for robotics systems
  - Continuous environments
  - Actions that take time to complete and can be interrupted
  - Uncertain and noisy sensors
  - Large scale space
- Find interesting Soar tasks in simple robotics

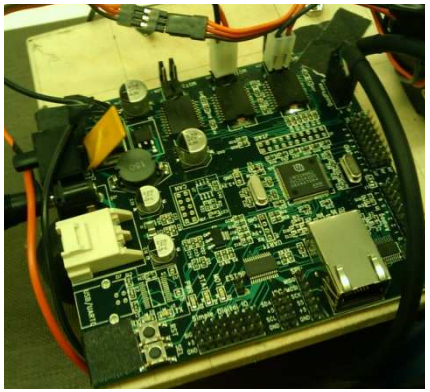
Prior Work

- Paul Benjamin @ Pace
- Hero Soar System
- Soar Tech and Roomba platform

Splinter

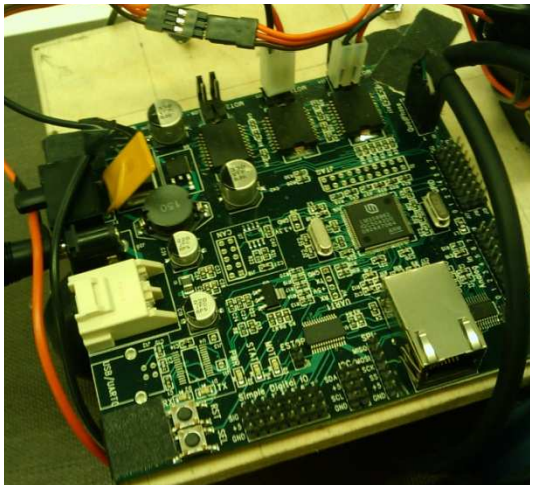
- Splinter is a simple, custom robotics platform brought to Michigan by Edwin Olson
  - Wood frame (namesake), SICK laser ranger
  - Differential drive (two independently powered wheels, turns in place)
- Splinter's nerve center: Orc (aka µOrc) board
  - Developed by Edwin Olson
  - Communicates easily over Ethernet (UDP) with Java interface
- Orc and Splinter used in classes at UM & MIT
- Very inexpensive





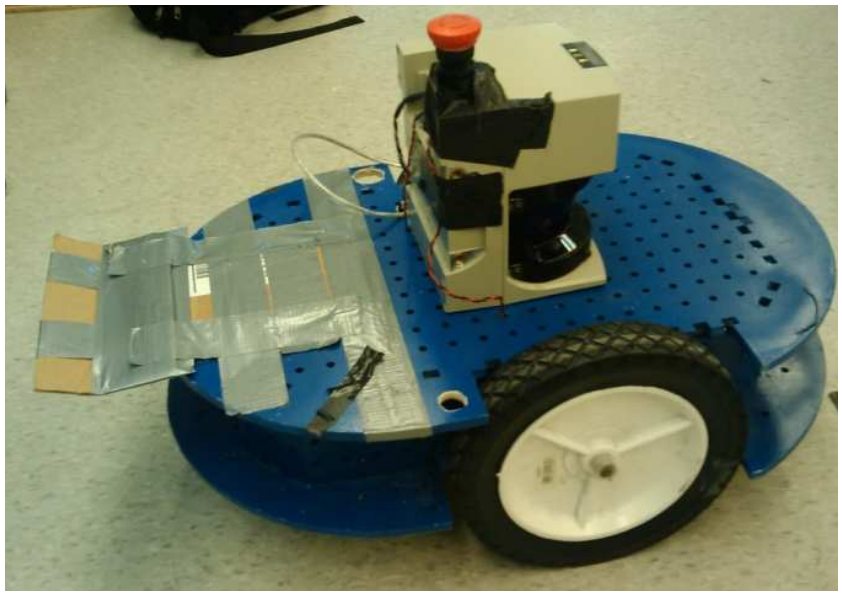
Orc Board Specifications

- 50MHz ARM CortexM3 Microcontroller (Luminary LM3S8962)
  - 64KB SRAM
  - 256KB Single-cycle FLASH
  - Real-time kernel
- 100bT Ethernet
- 3 High-current H-bridges with braking, open-circuit, and current sensing. These can be used to control (among other things) bi-directional brushed DC motors.
- 2 quadrature phase decoders
- 8 14-bit digital-to-analog converters
- 8 "Flexible" digital I/O pins. Currently implemented:
  - Servo control (for Futaba-style servos)
  - Digital in (very low latency)
  - Digital out (very low latency)
- 8 "Dumb" digital I/O pins, ideal for bump sensing and lower bandwidth applications.
- Dedicated emergency stop input
- I2C expansion bus
- SPI expansion bus
- Optional serial-to-USB adapter
- Optional CAN bus interface



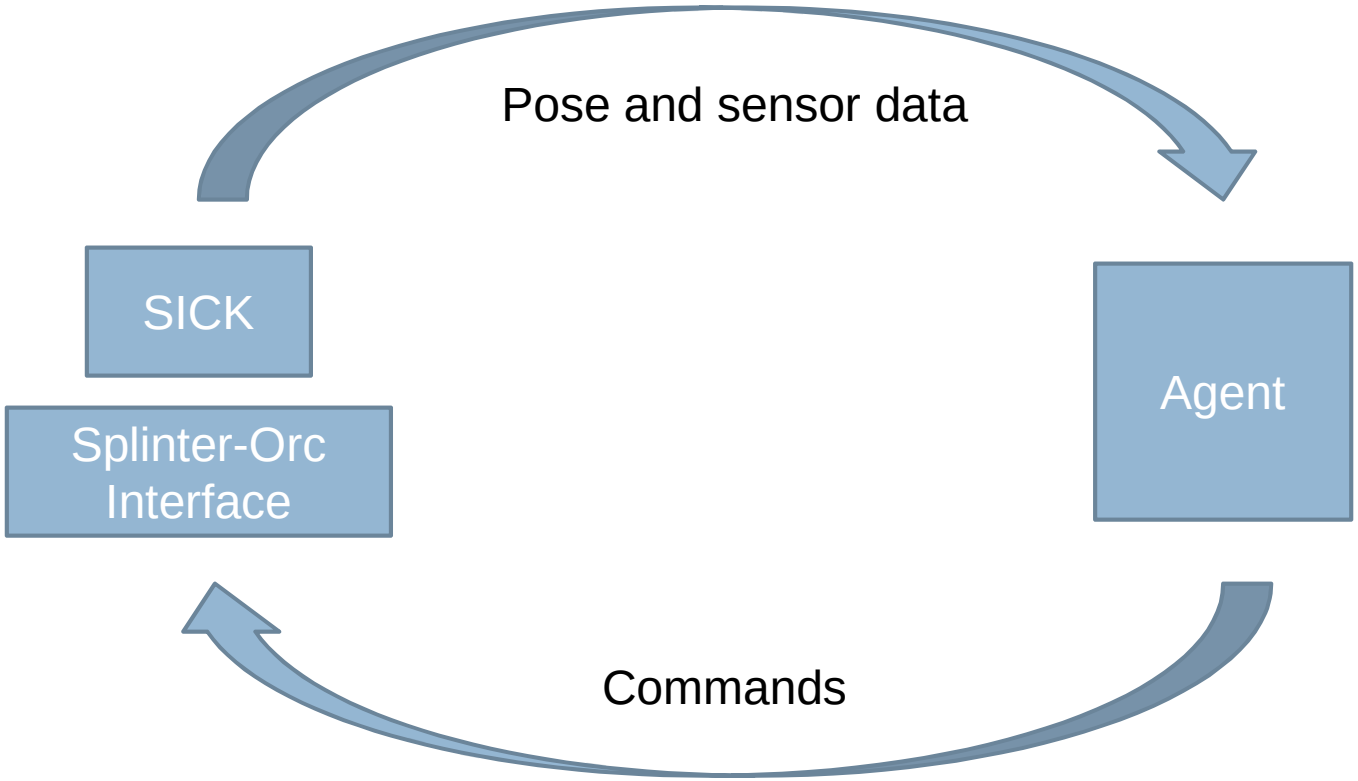
http://orcboard.org

Splinter Components

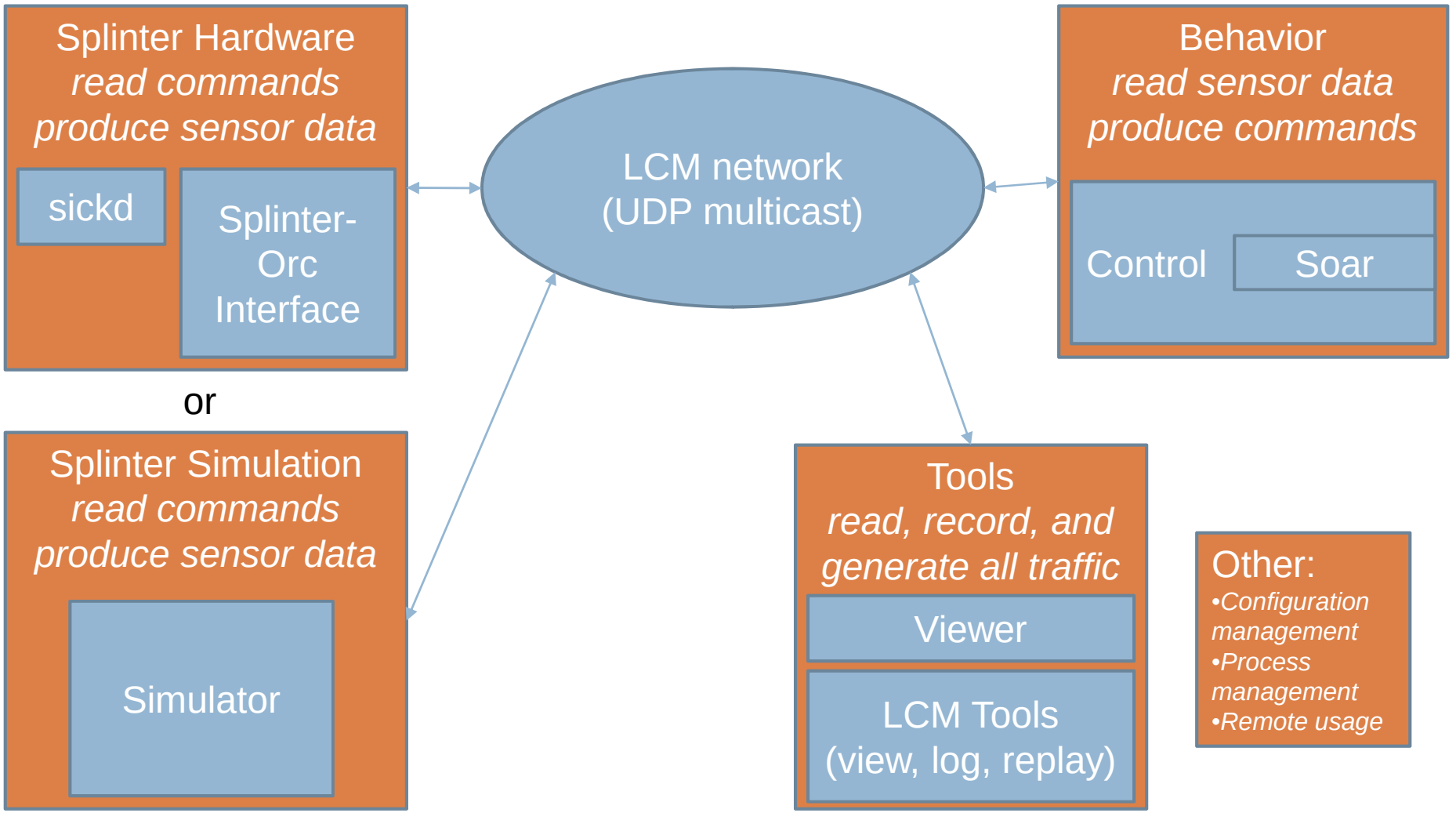


- SICK laser ranger
  - 180 degrees of range data to about 50 meters
- Two bi-directional brushed DC motors
  - High-quality odometry
- Space and support for more
  - Gyro
  - Camera
  - Bump sensors
  - and more

Sensors and Commands



Splinter Software

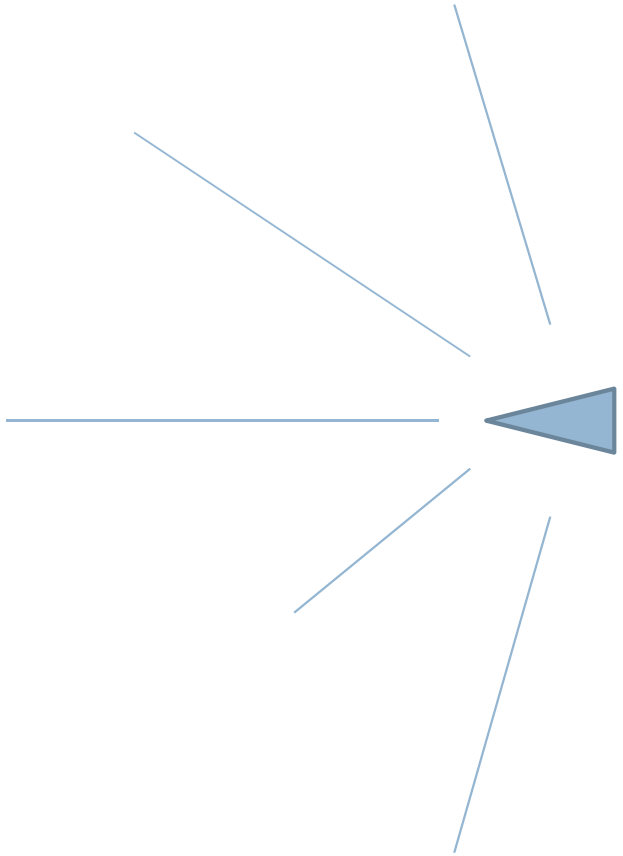


Behavior Module

- The behavior module essentially produces differential drive commands
- Control loops (like PID) and algorithms turn higher-level commands from the agent in to low level differential drive commands
- Splinter sensor and state data is picked up and given to agent input
- Utilities are available for agents to use
  - Waypoint system, reducing the amount of math and state the agent needs to keep track of
  - Messaging system, facilitates agent to agent and human to agent communication
- Manual override controls are available to help with developing agents and tuning control loops
  - Web interface
  - Gamepad

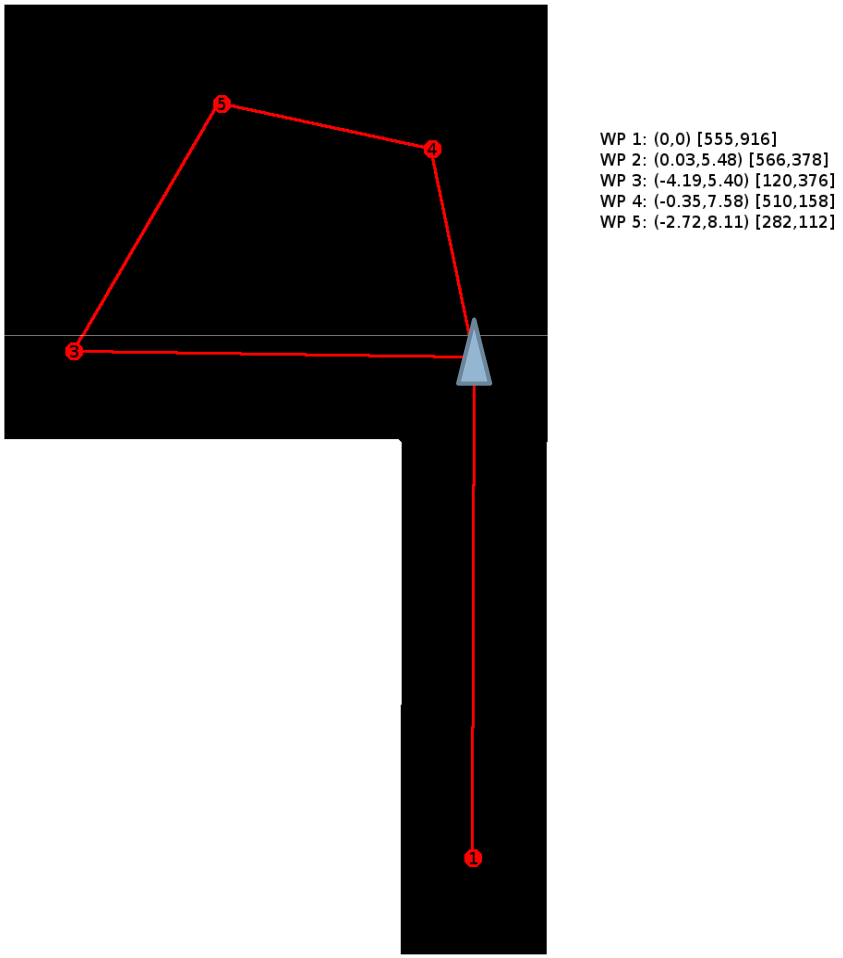
Range Data

- The SICK produces an enormous amount of range data
  - 180 ranges at 1 degree increments 75 times per second
- The behavior module “distills” the ranger data down to a smaller set
  - 5 ranges representing 36 degree arcs (front, front-left, left, etc.)



Waypoints

- Agents can define arbitrary waypoints managed by behavior module
- Convenient things are computed, such as distance to waypoint, relative angles, etc.
- Helps when building and maintaining a map



Agent Input

- pose (x, y, z, yaw, velocities)
- waypoints (one per waypoint)
  - id string, waypoint pose, distances, relative angles
- received messages
  - id string, who it is from, message content
- range data
  - arc size, id, distance to nearest obstruction in arc

Agent Output

Drive commands

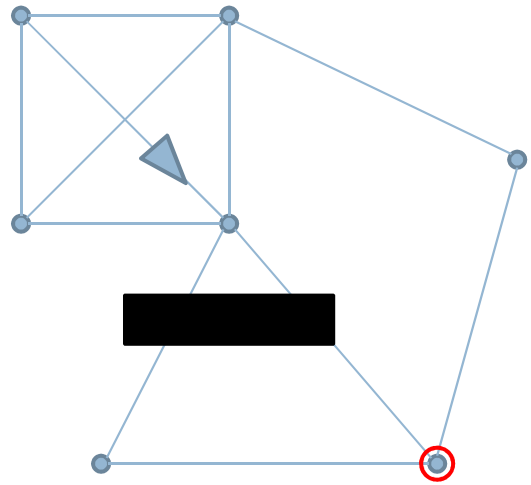
- motor
  - Create a low-level differential drive command (skip PID)
- velocity
  - Set a velocity vector to maintain, expressed in linear (m/s) and angular (rad/s) velocity
- heading
  - Turn to a target heading
- stop
  - Smooth stop
- estop
  - Emergency stop all

Other commands

- add-waypoint
  - Define a waypoint
- remove-waypoint
- enable-waypoint
  - Restore a hidden waypoint
- disable-waypoint
  - Hide a waypoint
- send-message
- remove-message
  - Remove a message by id
- clear-messages
- configure
  - Configure PID, input link representations (things like use ints instead of floats, etc.), other

Soar Programs on Splinter

- Robust agents
- Move around maps using waypoints
- Local collision avoidance
- Re-plan around broken map links



Nuggets & Coal

Nuggets

- Cheap platform other groups can use
  - Contact April Lab at UMich
- Good visualization tools are very important and worth the time
  - You can't debug what you can't see
- Porting from simulation to real hardware does work
  - Simulation isn't always "doomed to succeed"
- LCM simplifies synchronization issues
- Drove implementation of 64-bit Linux support

Coal

- Need real localization (SLAM, SSH, GPS)
  - We cheat and use odometry only, it degrades over time
- Perception & Control
  - Not sure about what level Soar should be controlling things
  - Still not sure of the best way to represent sensor data on the input link
- Many simple robotics tasks "boring" for Soar
- Simulation physics need to match reality