ET Robocon 2014 – Architect Requirements

1. MODEL ARCHITECTURE

* Final model specifications
  + Host Model
    1. Must contain a simulation and a standalone version
       1. Standalone version contains Bluetooth communication blocks
       2. Standalone version is a closed-loop simulation with robot models in the loop
    2. Computer vision algorithm
       1. Must be referenced by host model
       2. Reads data from camera or video file
       3. Outputs robot positions to host algorithm
    3. Host algorithm Stateflow chart
       1. Must be referenced by host model
       2. Contains parallel states for:
          - Sphero algorithm
          - Zumobot algorithm
          - Bluetooth + position command processing
  + Robot Models:
    1. Each robot model must be associated with a main template, and should look consistent.
       1. NXT
       2. Hydra
       3. Zumobot
    2. Robot model components:
       1. Plant model:
          - Must be able to switch between simulation and hardware
       2. State estimation:
          - Should always be the same state estimator independent of simulation or hardware (all discrepancies should be taken care of in the plant variant)
       3. Controller:
          - Reusable components must be linked to either atomic subcharts or top-level Simulink Functions.
  + Modeling Standards
    1. Variable standards:
       1. Variables must adhere to the modeling conventions specified in the standards documents
    2. Utilities standards:
       1. All reusable Simulink functions must exist in a library outside the robot or host model

1. COMMUNICATIONS

TBD

1. COMPUTER VISION
2. Object identification
   1. The system should be able to identify Spheros and Zumobots on the live captured video.
3. Object tracking
   1. The system should be able to track the identified object over the time from the beginning till the object goes outside of the captured range of the camera
4. Object localization
   1. The system should be able to calibrate the distortion due to the camera characteristics or the placement of the camera.
   2. After calibration, the system should be able to calculate the x-y coordinates of the tracked object
   3. The system should be able to pass the x-y location of the objects to the downstream systems (e.g., comm system and Sphero controller) in proper format
5. Self-maintain and self-calibrate
   1. After some initial tune up with a given setup (e.g., setting up the camera mount and the environment lighting), the system should require minimal or no further tune up, if it runs later with similar settings.
6. Low latency
   1. The system should run fast enough in accordance to the downstream requirement (The latency should not exceed TBD ms @ TBD fps capture rate)
7. High accuracy
   1. The x-y information should be accurate enough to reflect the actual physical location of the object. The error should not exceed TBD inch / TBD%
8. PHYSICAL MODELING
9. Simulation models must be completely compatible with modeling standards and architecture
10. Simulation and hardware models must have the same interface data properties.
11. Must have simulation models:
    1. NXT
       1. Need 3-D Contact model
       2. Ramp
       3. Flexible bridge
       4. Must be able to test balancing controllers.
    2. Hydra
       1. Need 3-D Contact model
       2. Must be able to test balancing controllers.
    3. Sphero
    4. Zumobot
12. Must have a multi-robot simulation that mimics the final performance.
13. All parameters must be tunable (no hard coded)
14. All parameters pertaining to plants must be in structures
15. Optional (nice to have): Need a good way to get actual data from hardware and tune parameters.
16. MINIONS

* Sphero
  + Interface
    1. Sphero should communicate via Bluetooth and host Bluetooth should follow a protocol/package structure which target understands
    2. 1 sphero should go to a desired location or track a desired path
    3. The lag in the communication should be in acceptable range   
       [TBD with Computer Vision team]
    4. The host should be able to interrupt the Sphero motion.
    5. The initial orientation of the Sphero should be known to the camera
       - This can be done by manual orienting the sphero in a desired orientation   
         *OR*
       - This can be done by camera + sphero performing a straight line motion
    6. Nice-to-have: Sphero should be able to communicate sensor data back to the host.
  + Algorithm
    1. Multiple spheros should be able to do a uniform circular motion being equidistant from each other around a desired object with a desired radius [ a swarm algorithm for uniform circular motion]
    2. From a group of spheros performing circular motion, if 1 sphero is shot in a direction, the remaining spheros should again maintain equal distance.
    3. Object avoidance
    4. path planning : need to go to a target using object avoidance
    5. Saturation on the controls
* Zumobot
  + Interface
    1. Hardware: Zumobot(s) + Arduino Mega 2560 + Bluetooth module & shield
    2. Zumobot should be able to communicate via Bluetooth and host Bluetooth should follow a protocol which target understands
    3. Should be able to reach a desired location communicated to it.
    4. Host should be able to trigger ON/ OFF
  + Algorithm
    1. Should be able to work in tandem to surround a target location (with a desired radius) [Capture mode]
    2. Object avoidance
    3. path planning : need to go to a target using object avoidance
    4. Saturation on the controls
* Host algorithm for minions:
  1. Should be able to toggle different operation modes (different state machines)
  2. The whole algorithm should be complete in the allotted time.

1. HYDRA

* Performance Requirements:
  1. Be awesome and chill! While balancing of course.
  2. (Nice to have): Final blow by rotating around itself and swinging the tail.
* Hardware Construction Requirements:
  1. EV3 should be easily accessible such that batteries can be replaced often and buttons are accessible.
  2. It needs to be robust to accidental forces (like moving of the tail while holding)
  3. Gears should be able to put in neutral easily
  4. Weight should be well balanced on all sides to ensure controllability
  5. Implement ability to sense range of motions such that it does not exceed the physical limits and ensures easier calibration.
  6. Wires to the peripherals should not hinder the movements of the robot.
  7. Two tail sections that are controllable with sufficient ranges of motion.
* Software Requirements:
  1. Co-ordinate between the two tail motions to balance the robot.
  2. Communicate with the master via Bluetooth.
  3. (Nice to have) Track the minions during battle with the head.
  4. (Nice to have) Be able to strafe with legs.
  5. (Nice to have) 360 degree rotation.
  6. (Nice to have) Moving on a given trajectory.

1. LEGO

* NXT:
  1. Communicate with the host through Bluetooth.
  2. Balance the robot.
  3. Robot should be able to track a line.
  4. Robot should track a commanded turn/movement.
  5. Ensure that motor receives a value between [-100 100].
  6. Robot should remain stable on rope bridge.
  7. Complete the pre-stage portion of the performance within 1 minute.
  8. Complete performance in 2 minutes
     1. Go up Mt. Olympus in 30 seconds.
     2. Capable of activating pressure switch.
     3. Come back down in 30 seconds.
     4. Push Godzilla off the stage.
* Pressure-switch:
  1. Program the Arduino to use the pressure switch to detect the NXT and raise a flag.
  2. Detect a force between 0 and 2 lbs
  3. Activate a servo motor to raise a flag to a height of 18 inches