Software Engineering for Robotics
RoboChart, RoboSim, and more

Ana Cavalcanti

University of York

February 2018

Thanks: Rob Hierons, Wei Li, Alvaro Miyazawa, Alexandre Mota, Pedro Ribeiro, Augusto Sampaio, Jon Timmis, Jim Woodcock
Overview

- EPSRC projects: RoboCalc and RoboTest
- RoboCalc start date: 01.09.2015
- RoboTest start date: 01.04.2018
- Duration: five years each
- One research assistant in Electronic Engineering
- Three research assistants in Computer Science
- To come: one research assistant in Brunel, one in Electronic Engineering, and one in Computer Science
## Advisory board

<table>
<thead>
<tr>
<th>Institution</th>
<th>Main contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adelard LLP</td>
<td>Robin Bloomfield</td>
</tr>
<tr>
<td>ASV</td>
<td>Richard Daltry/Vince Dobbin</td>
</tr>
<tr>
<td>BlueBear</td>
<td>Yoge Patel</td>
</tr>
<tr>
<td>Bristol Robotics Laboratory</td>
<td>Tony Pipe/Alan Winfield</td>
</tr>
<tr>
<td>D-RisQ</td>
<td>Colin O’Halloran &amp; Nick Tudor</td>
</tr>
<tr>
<td>Engineering Consultant Specialists</td>
<td>Ron Bell</td>
</tr>
<tr>
<td>Flightworks</td>
<td>Michael Pilmoor</td>
</tr>
<tr>
<td>Intel</td>
<td>Sean Fennelly</td>
</tr>
<tr>
<td>LDRA</td>
<td>Mike Hennel</td>
</tr>
<tr>
<td>Liverpool CAST</td>
<td>Michael Fisher</td>
</tr>
<tr>
<td>VSI, Germany</td>
<td>Jan Peleska</td>
</tr>
<tr>
<td>York Robotics Laboratory</td>
<td>Jon Timmis</td>
</tr>
<tr>
<td>Academy of Sciences, China</td>
<td>Naijun Zhan</td>
</tr>
<tr>
<td>UFPE, Brazil</td>
<td>Augusto Sampaio</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Software Engineering for Robotics  RoboChart, RoboSim, and more

Software engineering in Robotics

1st phase: Abstract model

state machine

2nd phase: Simulation

controller code | hardware simulation | discrete environment simulation

3rd phase: Implementation

low-level code | robot | environment
Our vision

RoboWorld+RoboChart Model
Our vision
Our vision
Our vision

RoboWorld+RoboChart Model → Proof Model → Properties of interest

Model checkers
Theorem provers → valid?
Our vision
Our vision

RoboWorld+RoboChart Model → Proof Model → Model checkers Theorem provers → valid? 

Properties of interest

NO

EPSRC RoboCalc

ROBOSIM Model

AUTOMATIC GENERATION

AUTOMATIC GENERATION
Our vision

Diagram:

1. RoboWorld+RoboChart Model
2. Proof Model
3. Properties of interest
4. Model checkers/Theorem provers
5. valid?
6. RoboSim Model
7. Robotics Simulator
8. valid?
Our vision
Our vision

1. RoboWorld+RoboChart Model
2. Proof Model
3. Properties of interest
4. Model checkers Theorem provers
5. RoboSim Model
6. Robotics Simulator
7. Deployment Code

Flow chart:
- RoboWorld+RoboChart Model → Proof Model → Properties of interest → Model checkers Theorem provers → RoboSim Model → Robotics Simulator → Deployment Code
- Valid? YES: Next Step
- Valid? NO: Previous Step
Our vision

1. RoboWorld+RoboChart Model
2. Proof Model
3. Model checkers Theorem provers
4. Properties of interest
5. RoboSim Model
6. Robotics Simulator
7. Test cases
8. Deployment Code
9. System Testing
10. Test cases
11. correct?
12. EPSRC RoboTest
13. AUTOMATIC GENERATION
14. CONVERSION
Our vision

- RoboWorld + RoboChart Model
- Proof Model
- Model checkers Theorem provers
- ROBOSTIC SIMULATOR
- Test cases
- System Testing
- Deployment Code
- Code Proofs
- Test results and assumptions
- Environment restrictions

Properties of interest:

Automatic Generation: RoboWorld + RoboChart Model

Requirements:

Certified for safe use

Valid? YES

Deployment Code

System Testing

Test cases

Conversion

RoboSim Model

Robotic Simulator

Valid? YES

RoboChart Model

Properties of interest

Automatic Generation

Valid? NO

Correct? YES

Correct? NO

Test results and assumptions

Environment restrictions

Certified for safe use
Key ideas

- Notations akin to those in current use
  - RoboChart
  - RoboEnv
  - RoboSim

- Mathematical principles
  - Like in any engineering discipline
  - Hidden from practitioners

- Specialisation and automation for usability and scalability

- Various specialised and modern techniques
  - Model checkers
  - Theorem provers
  - Test-case generators
  - Simulators
RoboTest Framework

RoboChart Model

Simulation

Robotic System

Tests for Simulation

Tests for Systems

DEVELOPMENT

AUTOMATIC GENERATION

EXECUTION

FEEDBACK

DEPLOYMENT

CONVERSION

EXECUTION

FEEDBACK
Key questions

1. Effectiveness: if we do not find a fault, what can we conclude?
2. Soundness: if we do find a fault, is it a problem in the robotic system or in the test?

Have enough simulations and tests been run? Are feasible inputs being considered with the right verdict?

Challenges

1. Definition of a rich semantic model;
2. Interplay between the notions of test for SIL, HIL, and deployment;
3. Test generation based on timed, probabilistic, and hybrid models for the robots and their environment; and
4. Tractability for a variety of simulation tools and application areas.
RoboChart Models

- Identifies a robotic system
- Single robotic platform
- Interaction with robotic platform is via
  - Shared variables
  - Operations
  - Events
- One of more controllers
  - Models a specific behaviour
  - Contains: variables, operations, events
  - Behaviour: state machines, with time and probability
  - Multiple state-machines can be used to define behaviour.
There are several (commercial) simulators.

- Netlogo, MASON, JASON, ...
- Argos, Enki, ...
- Webots, Microsoft Robotics Developer Studio
- Simulink + Stateflow

RoboChart is not a simulation language.

- Based on interrupts: events are handled as they become available.
- No notion of cycle
- We still want a diagrammatic notation for simulation.
- We want to relate RoboChart models to RoboSim models.
A very small example - State machine

```
SMovement

Movement
lv: real
PI: real
av: real
MBC
Obstacle

Moving
entry move(lv, 0);
wait (1)

Turning
entry move(0, av)

[since(MBC) >= PI/av]

obstacle
```
Simulation cycle

```
while true
read sensors
process data
write to actuators
```

{ cycle period }
Example: simple robot - RoboSim

Simulation_SimFW [cycle = 1]

- PI: real, av: real, lv: real
- MBC

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>i ISensors</td>
<td>R IOperations</td>
</tr>
</tbody>
</table>

SMoving
- entry $move(lv, 0)
- exec [since(MBC)>PI/av]
- exec [not $obstacle]
- exec [since(MBC)>PI/av]
- exec [$obstacle]/#MBC

DMoving
- exec [not $obstacle]

DTurning
- exec [since(MBC)<PI/av]

STurning
- entry $move(0, av)
- obstacle

IOperations
- move(ls: real, as: real)
What is a correct simulation?

- RoboChart describes *order, availability, and time* of accesses to global variables, events *and* operation calls *of the platform*.

- RoboSim describes *order, availability, and time* of registerRead *and* registerWrite *register reads and writes in a cycle*. 
What is a correct simulation?

We compare RoboChart + Assumptions ...

- Calculated model of RoboChart
- Assumptions:
  - Connection of input events to *registerRead* events
  - Connection of operation calls and output events to *registerWrite* events
  - Events do not occur between cycles.
  - Each cycle has at most one occurrence of each output.
  - In each cycle, each operation is called at most once.

... with RoboSim model.

- Calculated model of RoboSim
So, what next?

RoboChart: Software Engineering for Robotics
- Open access journal
- Special session in IROS
- RoboChart in IROS

RoboSim
- Implementation
- Refinement technique
- Soundness

... and more
- Properties language
- Modelling of environment
- Test generation