Trust Me I am Autonomous

Ambiant Intelligence axis

Jérémie Guiochet

Decisional autonomy
Decisional autonomy & robotics

> Sense/plan/act paradigm
> Unstructured environment
> No human intervention

1966 – Shakey the robot

Self Driving Uber sensor suite

- 7 Cameras
- 1 Laser
- Inertial Measurement Units

Custom compute and data storage
360° radar coverage

Advanced Technologies Group
This is it …

Collaborative work on Dependable robots@LAAS
Verification

Requirements / Analysis / Design

System behavior model

System properties

Model checking

Formal verification of the model (model execution)

Test

Verification of the effective execution of the code

MODEL CHECKING FOR AUTONOMOUS SOFTWARE
Model checking of autonomous software

> Objectives: what do robotic software developers want?
  - Check that the robot behave safely (e.g. stop in time when an obstacle has been detected, speed remain in bound, etc.)
  - Check that the robot has a consistent perception/action loop (e.g., laser scan freq and range, speed control, freq and value, time for an emergency stop, etc.)

> Issues: why not use basic model checkers?
  - No formal specifications of autonomous systems
  - No behavior models of autonomous software
  - Semantic gap between robot software and model checker tools

The proposed approach

Robotic software with Genom @ Laas

| process Manager (&tick: bool, ...) is
| states start, manage
| from start
| wait [0,0];
| on tick;
| tick := false;
| if (...) /* no active activity */ then to start
| else to manage end
| from manage
| wait [0,0];
| ... /* execute one active activity */
| if (...) /* no more activities */ then to start
| else to manage end
| Fiacre + TINA @ Laas
Results

> **Properties successfully checked with Fiacre/TINA** (e.g. Schedulability of execution tasks, no deadlock, position port update bounded in time)

> **GenoM specifications were good for verification**
  - Codel granularity
  - Internal and external shared data access is fully specified
  - Automata specification provides execution sequence and time/period management
  - Task are clearly specified (how many, periodic, sporadic)

> **Limits**
  - Based on the hypothesis that specifications are correct (verification is not validation)
  - Verification limited to properties of the software modules (not including operating system and hardware artefacts)
  - Genom Codels still need to be validated

> **A complementary approach... testing**

**TESTING NAVIGATION IN VIRTUAL WORDS**
Testing in simulation

Virtual world ≠ real world!

Input model of worlds and missions?

World & Mission

Generation

Analysis

Test oracle? (No ground truth about the decisions to take)

Test reports
Two case studies (outdoor navigation)

Mana
- Outdoor experiments @LAAS
- Generic navigation missions
- Path planning similar to NASA's GESTALT algorithm for Mars exploration rovers
- 35 KLOC including 3D mapping, localisation, path planning
- MORSE simulator (based on the Blender game engine)

Oz
- Agricultural robots developed and commercialized by Naïo Technologies
- Weeding missions
- Proprietary and mission-specific software
- 151 KLOC (also including modules we do not test: control of weeding tools & user interface)
- Gazebo simulator

Can robot navigation bugs be reproduced in simulation?

In-depth analysis of identified bugs in the Mana navigation software and analysis of their reproducibility in simulation (2005-2015 history of code)

<table>
<thead>
<tr>
<th>Library</th>
<th>Commits</th>
<th>Bugs</th>
</tr>
</thead>
<tbody>
<tr>
<td>P3D</td>
<td>69</td>
<td>12</td>
</tr>
<tr>
<td>LibP3D</td>
<td>154</td>
<td>14</td>
</tr>
<tr>
<td>DTM</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>POM</td>
<td>83</td>
<td>4</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>356</strong></td>
<td><strong>33</strong></td>
</tr>
</tbody>
</table>

32 reproducible
1 non reproducible
World & mission Models (2)

- Oz model: 31 generation parameters (some with interdependancies)
- Grammar-based approach to manage the parameters
  Descriptor: genotype of a world or a world element
- Descriptors can have wildcards
  $R = \text{any random world}$
  $f = 0..+0..-R$ (flat terrain, any mission and field)

Oracle? A navigation system is not deterministic!

- Non deterministic trajectory -> non deterministic verdict
- No ground truth for decisional functions ($\text{Mission Failure} \neq \text{Fail verdict}$)
  → Definition of measures and classes of properties

Oz experiment: 30% of the test cases do not yield a consistent verdict for 5 repeated runs
Results for robot testing under simulation

> Results
  - Preliminary validation of procedural 3D world generation with a grammar, reproducibility, and measures for oracle
  - Successful application to an industrial case study (full code access)

> Limits / perspectives
  - only random testing -> search-based testing (fitness, evolutionary algorithms, etc.)
  - only static situations -> dynamic situations (e.g. mobile obstacles at the right time and location)
  - confidence in virtual test results (statistics), e.g. how to use them in certification?

To conclude

> Trust in autonomous system could be achieved through 3 activities:
  - Social acceptance criteria elicitation (not presented here)
  - Structured arguments: how to argue that acceptability objectives are reached / combine assurance building blocks (not presented here)
  - Assurance building blocks, based on dependability techniques (presented here)

> Verification is part of assurance building blocks (e.g. model checking and test in simulation)
  - What is the confidence level? (still open issues, e.g. verification of machine learning algorithms)
  - What is the contribution in a structured argumentation?

> Work in progress at LAAS (not exhaustive):
  - Several running phds (safety monitoring, testing, model checking, etc.) and starting in september/october 2018 (safety argumentation, verification, etc.)
  - A European project SAS (Safer Autonomous Systems) nov2018-nov2022
  - A RTTRA chantier “TrustMeIA” led by the LAAS
  - Contracts/PhDs in verif/decisional : eHorizon (Continental) , FURIOUS (Safran), etc.
> Thank you, questions ?

A path for trusting autonomous systems

Social acceptability criteria elicitation
- E.g.: Is an autonomous vehicle able to avoid an obstacle or just try to stop?

How to argue that acceptability objectives are reached / combine assurance building blocks
- E.g.: combining virtual testing and operational testing on 1 million miles is sufficient?

Argumentation building blocks / risk treatment / dependability techniques
- E.g.: verification of AI components?