Can Robot Navigation Bugs be Found in Simulation? An Exploratory Study

Thierry Sotiropoulos
Jérémie Guiochet Félix Ingrand Hélène Waeselynck

LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France
Emails: firstname.name@laas.fr

Toulouse, SHARC conference, June 26, 2017

Outline

Context

Safety challenges

- Deployment of autonomous robots in environments:
  ⇒ **unstructured**, **unknown**, and **human-shared**
- **Navigation** is **critical** and must be **validated**

Testing

- Test in **real worlds**
  - Expensive
  - Limited number of test situations
  - Hazardous
- Test in **simulation**
  - Cheaper
  - Potentially more complete in terms of simulated situations
  - Risk-free
  - Gap between simulation and reality ⇒ analysis of the trigger and effects of bugs

Simulator

Modular Open Robots Simulation Engine (MORSE)

- Software-in-the-loop testing (real robot modules are executed)
- Based on the Blender game engine
- Provides basic world generation tools
- Provides a library of sensors and effectors
- Mainly used for prototyping purposes

Robot in the simulation environment MORSE

System Under Test

Navigation service

- From **Mana** meta-package[1] (robotic modules for Mana robotic platform)
- Encompassing localisation (**pom-genom** module), local-planning (**p3d-genom** module) and 3D mapping (**dtm-genom** module)

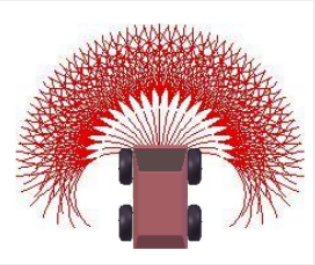
Mana robot on field

[1] http://robotpkg.openrobots.org/robotpkg/meta-pkgs/mana/index.html

Core: P3D Local Path-planning

Description

- **Academic implementation** of NASA's GESTALT algorithm for **Mars exploration** rovers
- Considers a fixed number of **arc-shaped paths** in front of the robot
- Considers on each path different points (called **nodes**)
- Minimizes **traversability-stability** cost and the distance to target arrival point
- Cost is **infinite** if the terrain is unknown (no perception)



P3D arcs in front of robot
(for `depth=2` and `nbArcs=20`)

Objectives

Relevance of simulation-based testing

🎯 Study of the reproducibility of bugs in simulation

- Bugs extracted from **navigation services** of Mana
- Impact of the simulator fidelity level
  - **RQ1: Can robot navigation bugs be reproduced in low-fidelity simulation?**
- Inputs to consider
  - **RQ2: Which inputs are to be considered to trigger the bugs?**
- Observations and oracle procedures to consider
  - **RQ3: Which observation data and oracle procedures should be considered?**

Outline

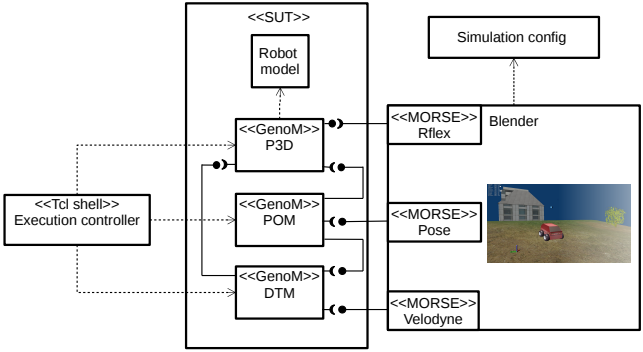
Physical Fidelity

Physical fidelity

- No inertia
- No reaction between wheels and ground
- No slippery areas

Remark

- MORSE may offer **more realistic simulation** of the physics ...
- ... but at the price of **longer computing times** and greater effort !



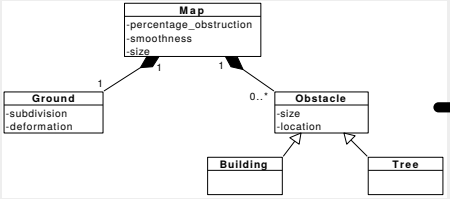
Simplified diagram of the test architecture

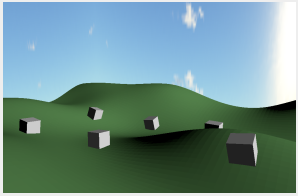
Test Inputs Models

Mission Model

- Starting point
- Arrival point

World model





Simplified world model and generated world

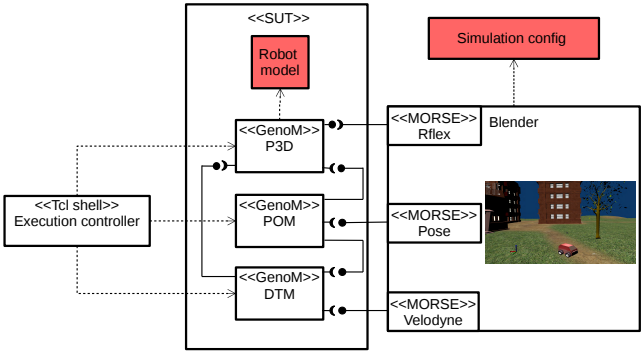
Configuration

Additional input configuration

- Physical robot configuration (e.g., size, sensors)
- Parameters of the navigation algorithms (e.g., number of arcs explored by P3D)

Remark

The developers did not archive the configuration files



Simplified diagram of the test architecture

Outputs and Oracle

Outputs

- **Robot's point of view**
  - Timestamped perceived positions
  - Perceived map at the end of the run
  - Error messages
- **External observer's point of view**
  - Timestamped real positions
  - Collision events
  - Timeout events

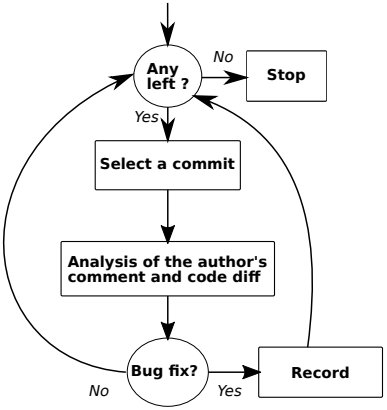
Oracle

- **Only** collision detected

Workflow

Approach

We considered the commits of P3D, libP3D, DTM and POM (less than 400)



Analysis workflow

Workflow

Approach

Bug fixes are recorded in a form

- **Location**
- **Fault**
- **Failure**
- **Time to fix**
- **Description**
- **Reproducibility**
  - Overall Judgment: *not reproducible/reproducible in theory/reproduced*
  - Constraint(s) on the simulation fidelity
  - Constraint(s) on the world/mission
  - Constraint(s) on the configuration data
  - Raw data to observe
  - Post-Processing to detect misbehavior

Form to be filled for each bug

Workflow

Reproducible in theory or *reproduced*?

1. **Re-create** the software version **before** the commit
2. **Inject** the identified bug into the **current** version

Workflow

Reproducible in theory or *reproduced*?

1. **Re-create** the software version **before** the commit
2. **Inject** the identified bug into the **current** version

Problems

- Developers **did not archive** all versions of modules and libraries
- Developers **did not archive** configuration files
- **Test scripts no longer work** for old version

Workflow

Reproducible in theory or *reproduced*?

1. ~~**Re-create** the software version **before** the commit~~
2. **Inject** the identified bug into the **current** version

Workflow

Reproducible in theory or *reproduced*?

1. ~~**Re-create** the software version **before** the commit~~
2. **Inject** the identified bug into the **current** version

Problems

- Some bugs affect a function that no longer exists
- Some bugs require to undo changes in several parts of the software
  - **Not always possible to inject the bug** $\Rightarrow$ discussion with engineer

Outline

Research Question 1

RQ1: Can robot navigation bugs be reproduced in low-fidelity simulation?

Bugs and their Reproducibility

Not reproducible	Reproducible in theory	Reproduced
1	21	11

Judgment about the reproducibility of bugs

Comments

- Only one bug is deemed not reproducible (mechanical vibration during **spot turn**)
- Reproducible in theory:
  - 10 memory leaks and 1 out-of-range indexing of an array (out of the scope)
  - 4 bugs in the spot turn function (no longer exists)
  - 3 affecting the processing of sensor data (sensor not available in MORSE)
  - 3 affecting `P3D_Blocked` error (unrealistic P3D configuration)
- We add **inertia** to the baseline

Research Question 2

RQ2: Which inputs are to be considered to trigger the bugs?

Inputs: Worlds, Missions and Configuration (RQ2)

Comments

- 7 bugs do not need trigger conditions
- Some bugs require combinations of conditions

	Trigger conditions
Mission	Destination point behind the robot at start point
	Start point and destination points do not have the same Y value
	Long distance between start and destination point
	Several way-points per mission, or several missions in sequence
	Mission abortion and replacement
World	Dead end
	Hole
	Large map
Config.	P3D depth > 1
	Goal tolerance is small (tested with 0.1)
	Specific sensors
	Incorrect P3D parameters

Inputs and configurations used to trigger the faults

Research Question 3 (RQ3)

RQ3: Which observation data and oracle procedures should be considered?

Observation Data and Oracle Procedures (RQ3)

- Infinite spot turn
- Failure to align to the target destination point
- Jerks in angular speed commands
- Robot does not immediately stop after detecting an error
- The robot arrives successfully at destination but considers itself as blocked
- The robot brakes too late when arriving at destination
- The speed commands are not refreshed and retain their value forever
- P3D does not start
- Execution crash
- Unexpected mission failure
- The robot goes round and round in circles until time-out
- The robot falls into a hole
- The robot has an absurd trajectory

List of encountered failures

Observation Data and Oracle Procedures (RQ3)

Comments

- Raw data collected by baseline are almost complete (except speed commands sent to the wheels)
- High diversity of failures → properties
- Need of some reference to distinguish performance-related issues from legetimate behavior → non regression testing

Possible properties

- *Requirements attached to mission phases* (inital bad alignment to the destination)
- *Thresholds related to robot movement* (maximal variation of speed commands)
- *Catastrophic events* (collision)
- *Requirements attached to error reports* (stop immediately after reporting an error)
- *Perception requirements* (maximal unknown areas in the perceived map)

Conclusion

Exploration of the reproducibility in simulation of bugs

- Bugs affecting the navigation software of an outdoor robot
- Bugs collected using manual analysis of commit history
- **Recommandations:**
  - Consider the configuration files as an integral part of the software
  - Use appropriate tools (e.g., *Valgrind*) to detect programmation bugs non specific to robot navigation

Insights into domain-specific triggers and effects

- Many navigation bugs do not require high physical fidelity
- Interesting improvements concerning inputs (situation-based testing)
- Definition of properties covering requirement attached to mission phases, thresholds related to robot movement, catastrophic events, requirements attached to error reports and perception requirements.

Conclusion

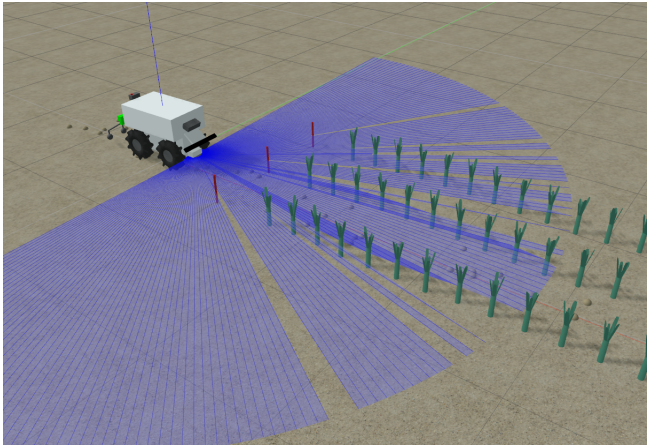
Exploration of the reproducibility in simulation of bugs

- Bugs affecting the navigation software of an outdoor robot
- Bugs collected using manual analysis of commit history
- **Recommandations:**
  - Consider the configuration files as an integral part of the software
  - Use appropriate tools (e.g., *Valgrind*) to detect programmation bugs non specific to robot navigation

Insights into domain-specific triggers and effects

- Many navigation bugs do not require high physical fidelity
- Interesting improvements concerning inputs (situation-based testing)
- Definition of properties covering requirement attached to mission phases, thresholds related to robot movement, catastrophic events, requirements attached to error reports and perception requirements.

Future Work



Naïo *Oz* agricultural robot