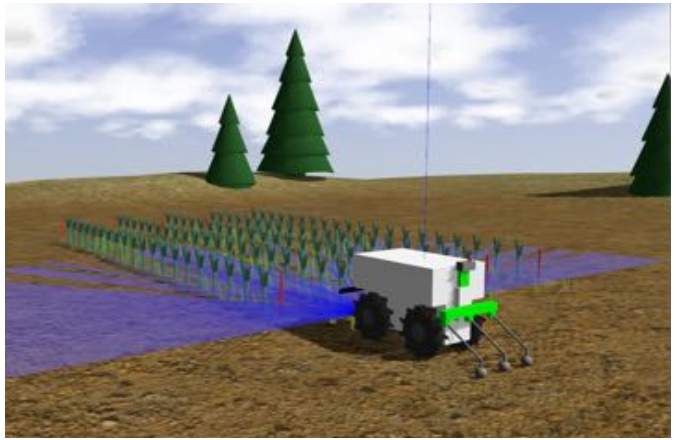
TESTING AUTONOMOUS SYSTEMS IN SIMULATION

Jérémie Guiochet

LAAS-CNRS, Université de Toulouse 3

jeremie.guiochet@laas.fr

http://homepages.laas.fr/guiochet



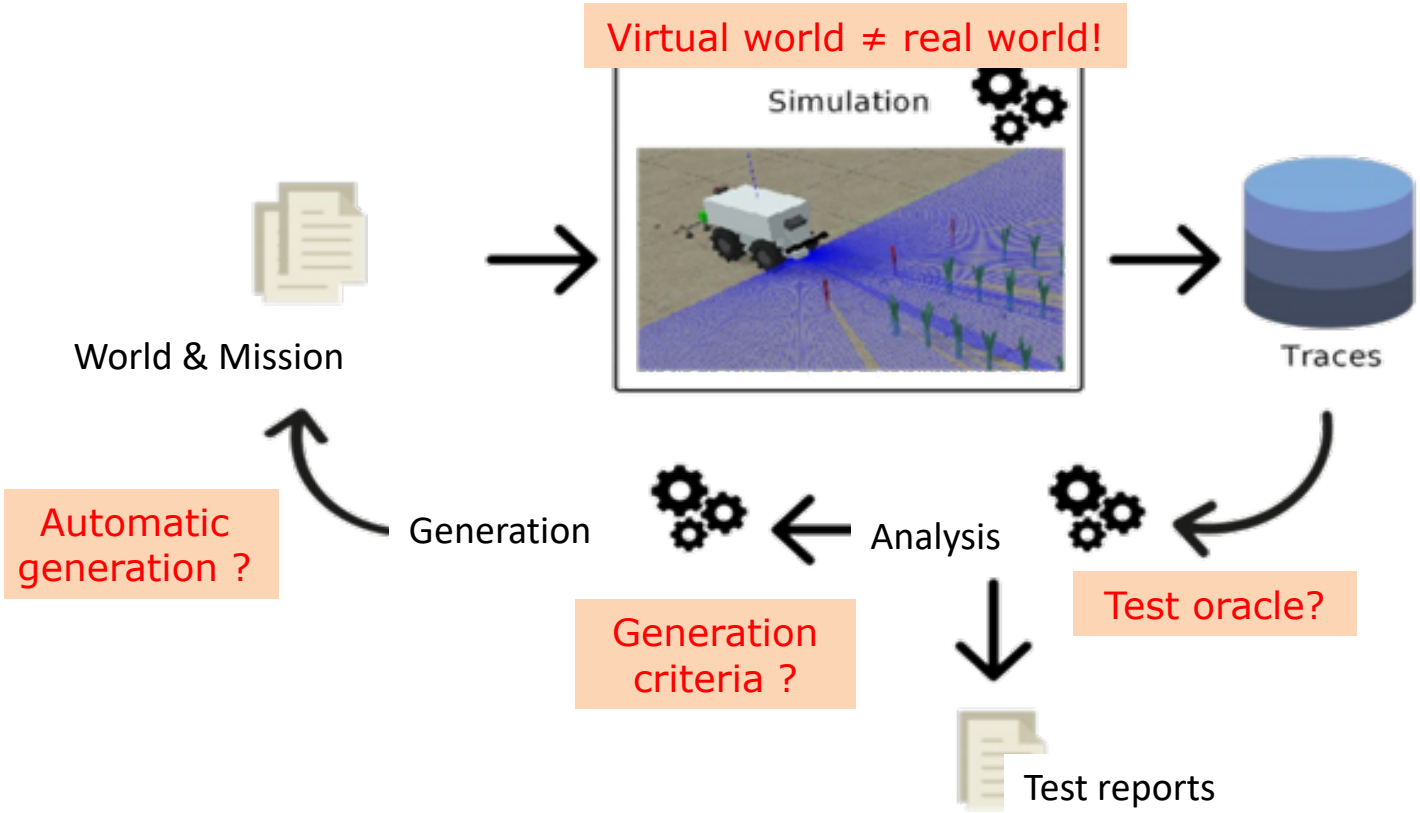
Context@LAAS

- *Trustworthy Computing Systems and Networks* (RISC) department
- *Dependable Computing and Fault Tolerance (TSF)* research group
- Safety critical autonomous systems:
  - Hazardous scenario identification for human-robot interactions
  - Run-time monitoring (mobile robots and AI-based functions)
  - Safety case confidence estimation (applied to autonomous functions)
  - Test in simulation of autonomous systems
    - Avec Hélène Waeselynck : 1 thèse soutenue (avec Félix Ingrand), 3 en cours, 2 projets européens, 2 projets nationaux déposés

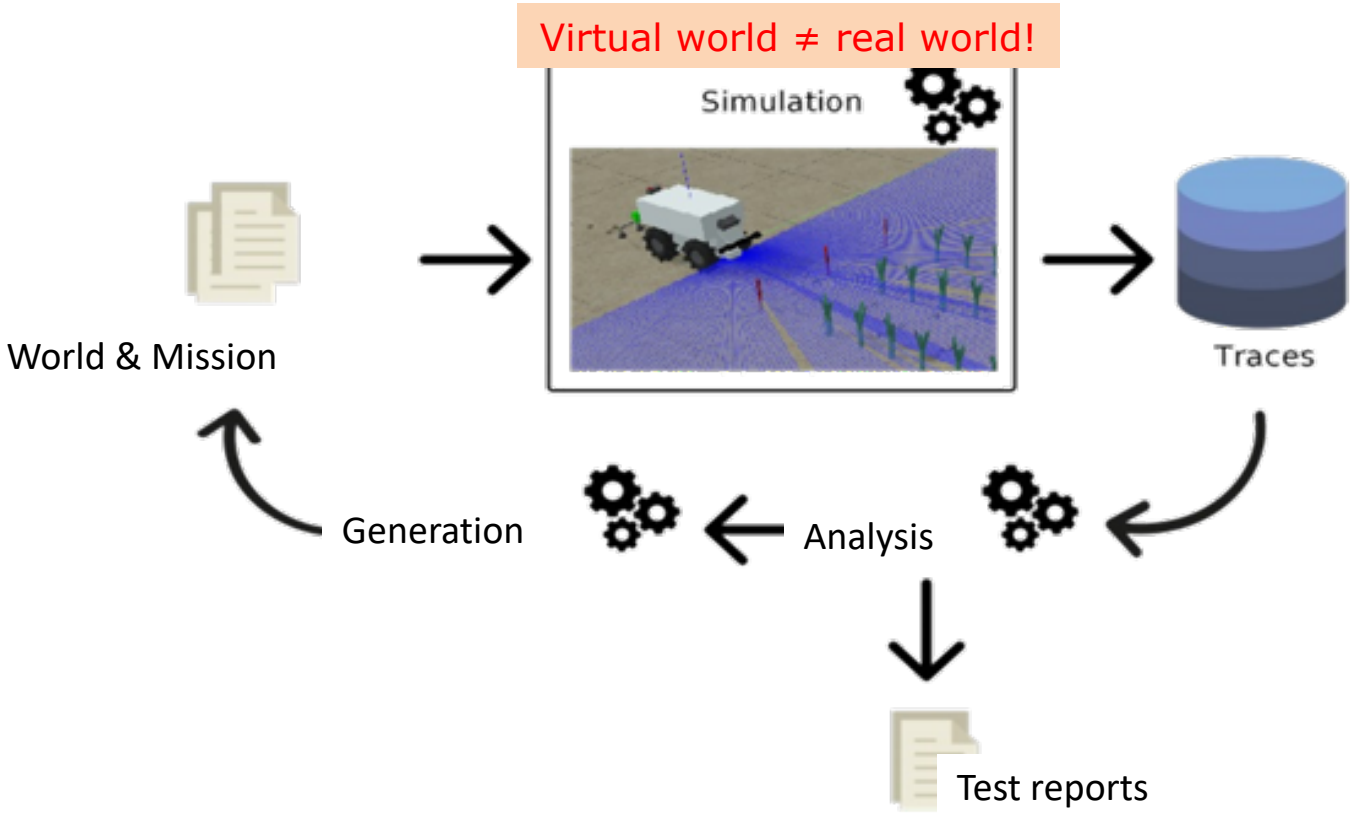
Testing

- Test in the field: ressources consuming, limited to few test cases, damages to the system or the environment (including humans)
- New opportunities with recent advances in simulation
  - Model in the loop simulation
  - Hardware in the loop simulation
  - Software in the loop simulation

Testing in simulation (Sw in the loop) - Challenges



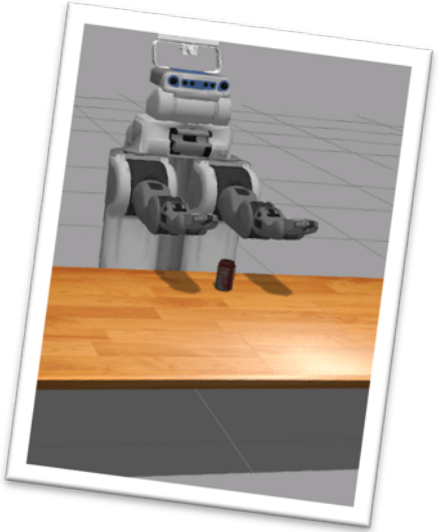
Testing in simulation (Sw in the loop) - Challenges

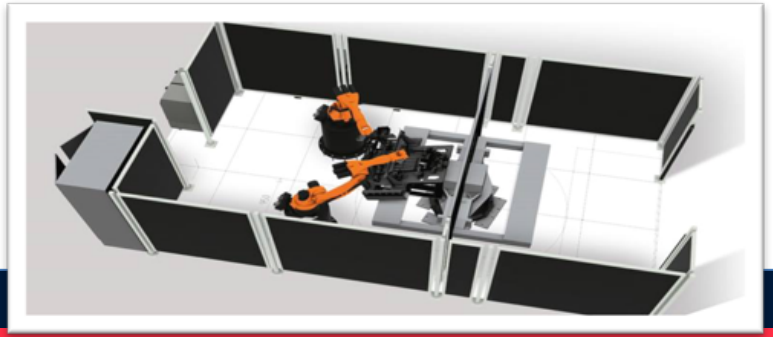


Virtual world ≠ real world!

Virtual world ≠ real world!

- “System” simulators
  - Generic (and open) simulators in robotics (e.g., Gazebo/Ignition, Unity, Unreal, Morse, etc.)
  - Not open industrial robot simu (KukaSim, Fanuc RobotGuide, Staubli SRS, etc.)
  - Autonomous vehicle simulators
- Main features
  - Fidelity of the simulation (physical engine, env., sensors/actuators, etc.)
  - Interoperability
  - Openness / Community





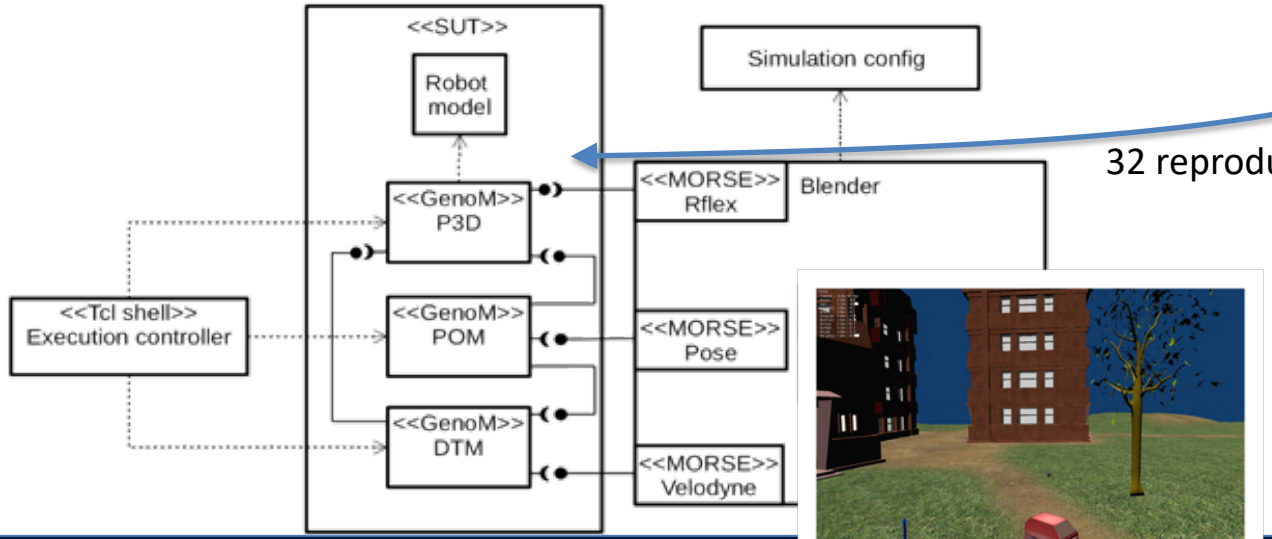
Virtual world ≠ real world!

- Challenge : validity of results in simulation in order to switch to production
  - Which level of fidelity is required regarding specific testing objectives ?
  - All bugs can be found with simulation ?

Can robot bugs be reproduced in simulation?

Bugs in the *Mana* navigation software and analysis of their reproductibility in simulation (2005-2015 history of code)

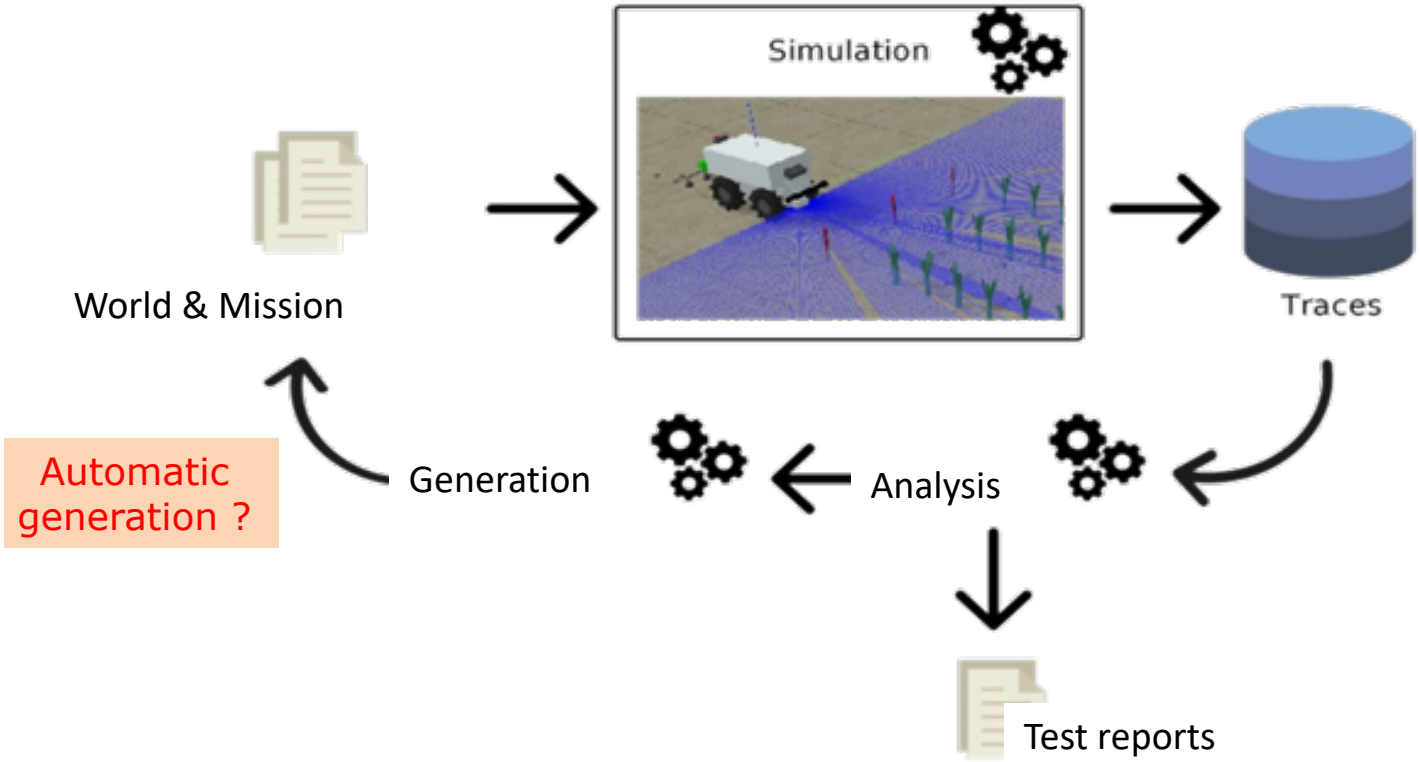
P3D	69 commits	12 bugs
LibP3D	154 commits	14 bugs
DTM	50 commits	3 bugs
POM	83 commits	4 bugs
Total	**356 commits**	**33 bugs**

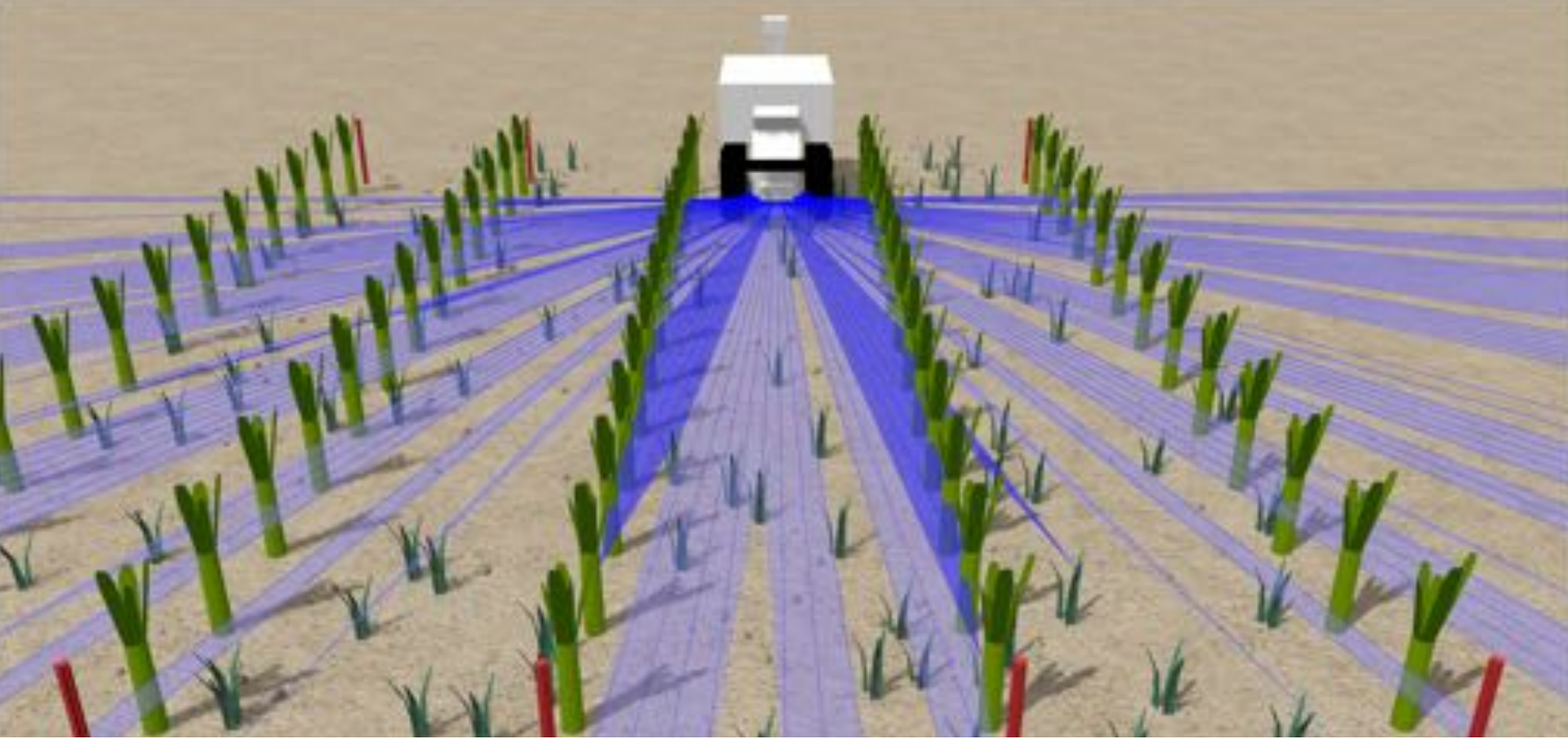


32 reproducible

1 non reproducible

Testing in simulation (Sw in the loop) - Challenges



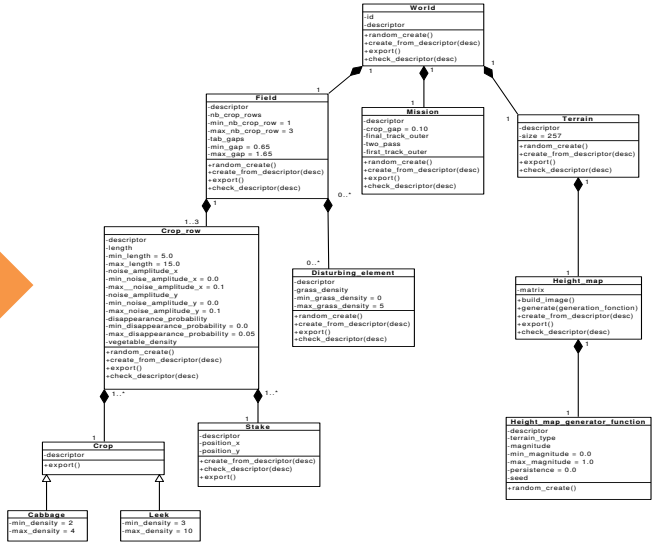


Automatic generation of worlds and mission

https://www.laas.fr/projects/taf/

Automatic generation of worlds and mission

A world



A model

A generator

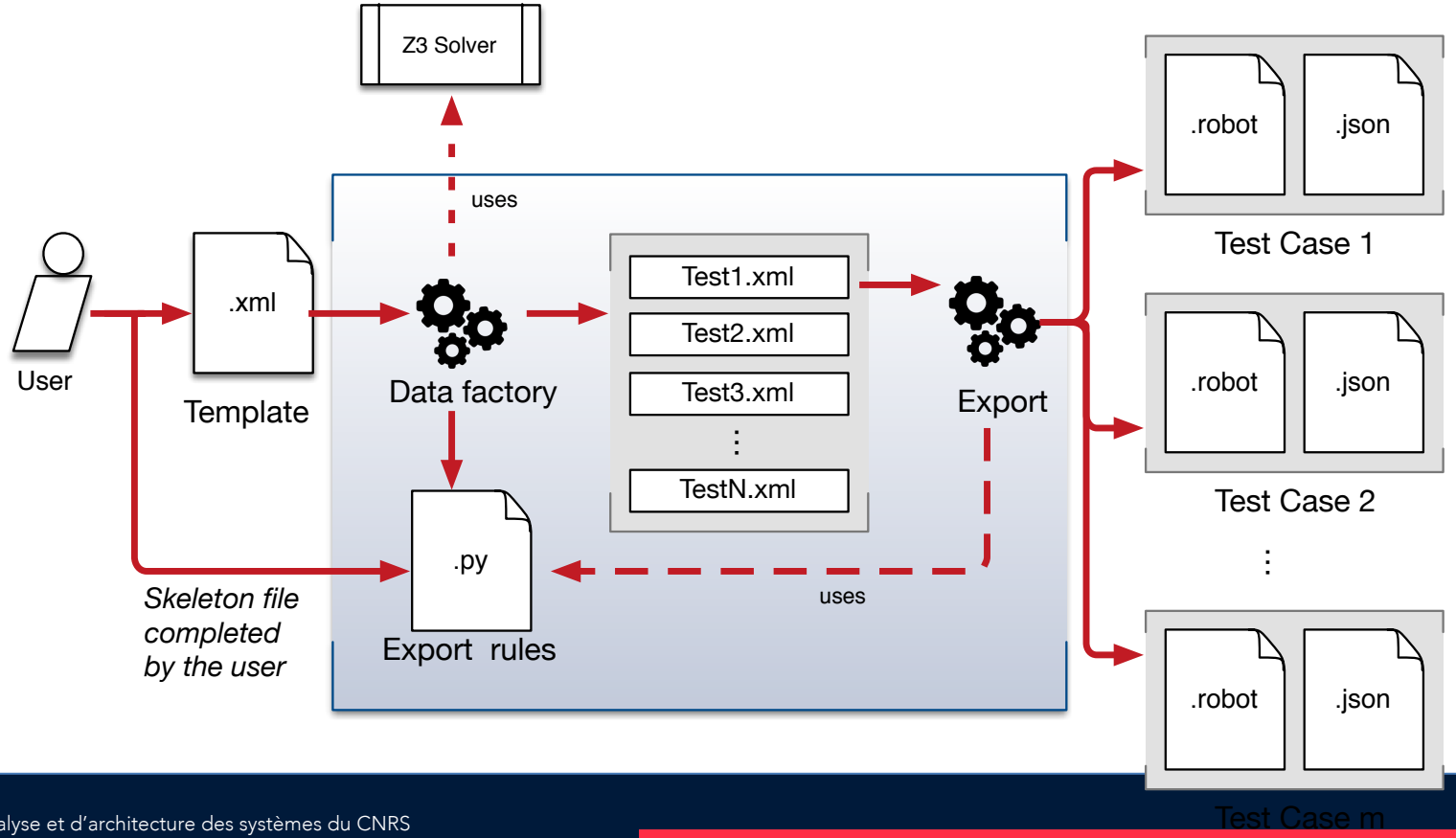
Files for the simu

Automatic generation of worlds and mission

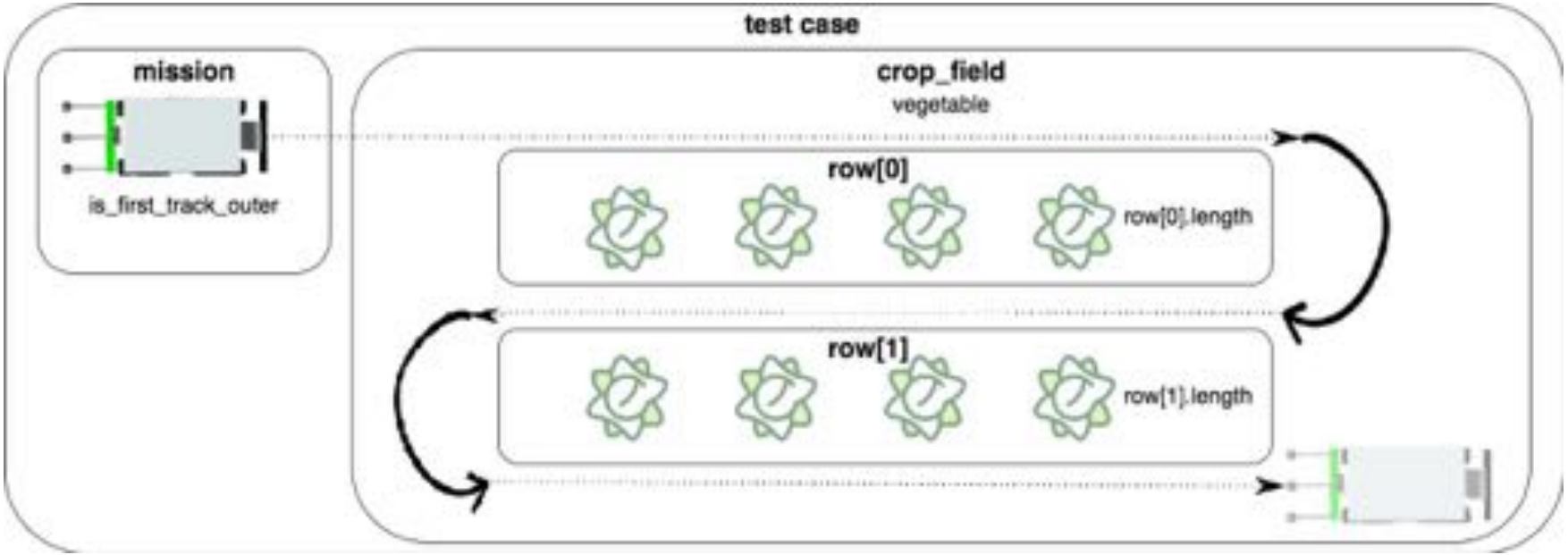
- Challenge : from a specification of worlds and missions, generate automatically several consistent and diverse test cases
  - World and mission description language
  - With constraints
  - Solve these constraints with diversity to generate test case

TAF : Testing Automation Framework © LAAS

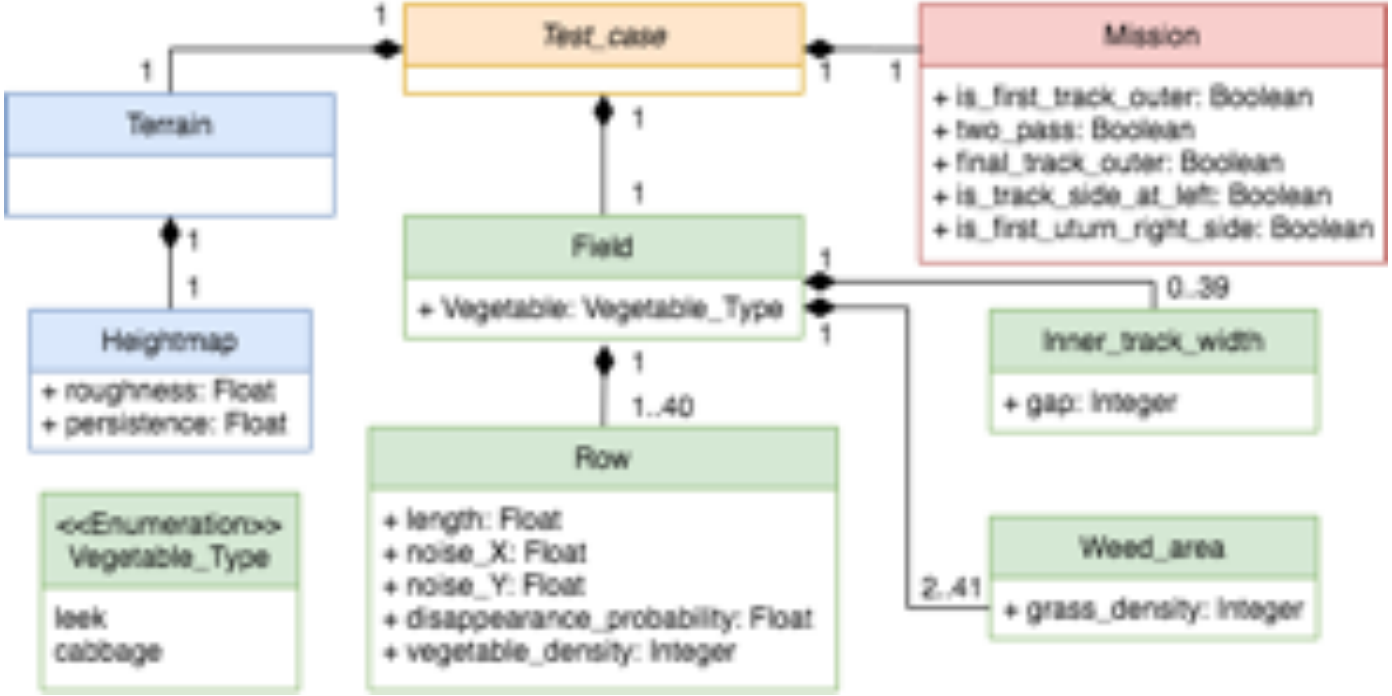
https://www.laas.fr/projects/taf/



Oz Example (1)


test case
mission
is_first_track_outer
crop_field
vegetable
row[0]
row[0].length
row[1]
row[1].length

Oz Example (2)


Test_case
Mission
+ is_first_track_outer: Boolean
+ two_pass: Boolean
+ final_track_outer: Boolean
+ is_track_side_at_left: Boolean
+ is_first_uturn_right_side: Boolean
Terrain
Field
+ Vegetable: Vegetable_Type
Heightmap
+ roughness: Float
+ persistence: Float
Inner_track_width
+ gap: Integer
<<Enumeration>>
Vegetable_Type
leek
cabbage
Row
+ length: Float
+ noise_X: Float
+ noise_Y: Float
+ disappearance_probability: Float
+ vegetable_density: Integer
Weed_area
+ grass_density: Integer
1
0..39
1..40
2..41

Oz Example (extract) XML-TAF

```
<node name="field" nb_instances="1">

<parameter name="vegetable"
  type="string" values="cabbage;leek"
  weights="5;7" />
  <node name="row" min="1" max="100">
    <parameter name="length" type="real"
min="10.0" max="100.0"/>
    <constraint
    name="interval" types="forall"
    expressions="
      row[i]\length INFEQ 1.1*row[i-1]\length;
       row[i]\length SUPEQ 0.9*row[i-
1]\length"
    quantifiers="i"
    ranges="[1, row.nb_instances-1]"/>
  </node>
[...]
</node>
```

Template input file

```
<node name="field" instance="0/0">
  <parameter name="vegetable" values="leek"/>

  <node name="row" instance="0/9">
   <parameter name="length"
values="44.99391"/>
  </node>

  <node name="row" instance="1/9">
   <parameter name="length"
values="40.81666"/>
  </node>

  <node name="row" instance="2/9">
   <parameter name="length"
values="44.89833"/>
  </node>

  <node name="row" instance="3/9">
   <parameter name="length"
values="49.38816"/>
  </node>
[...]
```

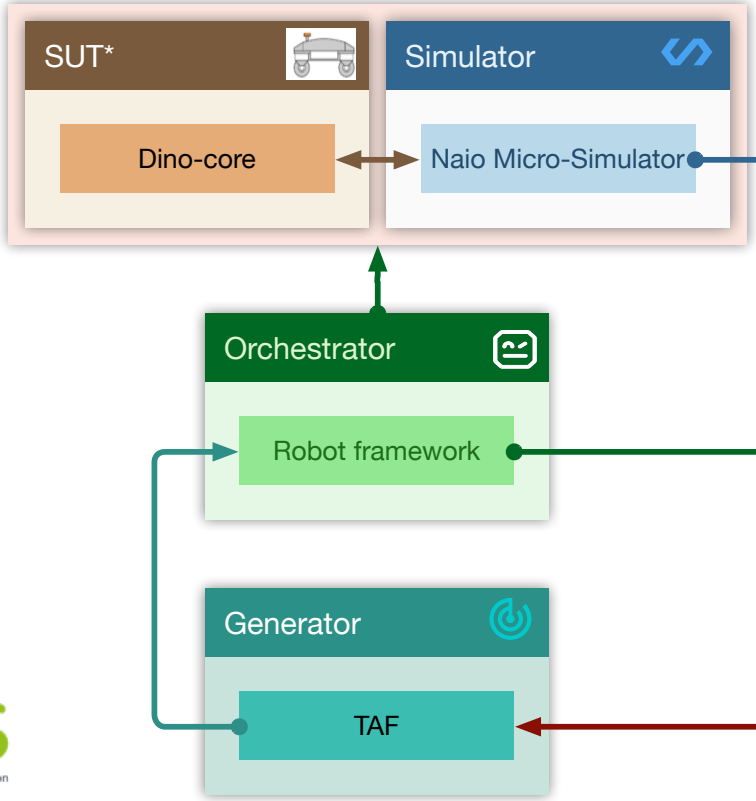
Generated output file (before export)

Other examples

- Grayscale bitmap (#Bitmap)
- Tree = connected acyclic graph (#Tree)
- Database (#TaxPayer)

Example in the TARIS Framework@Naio

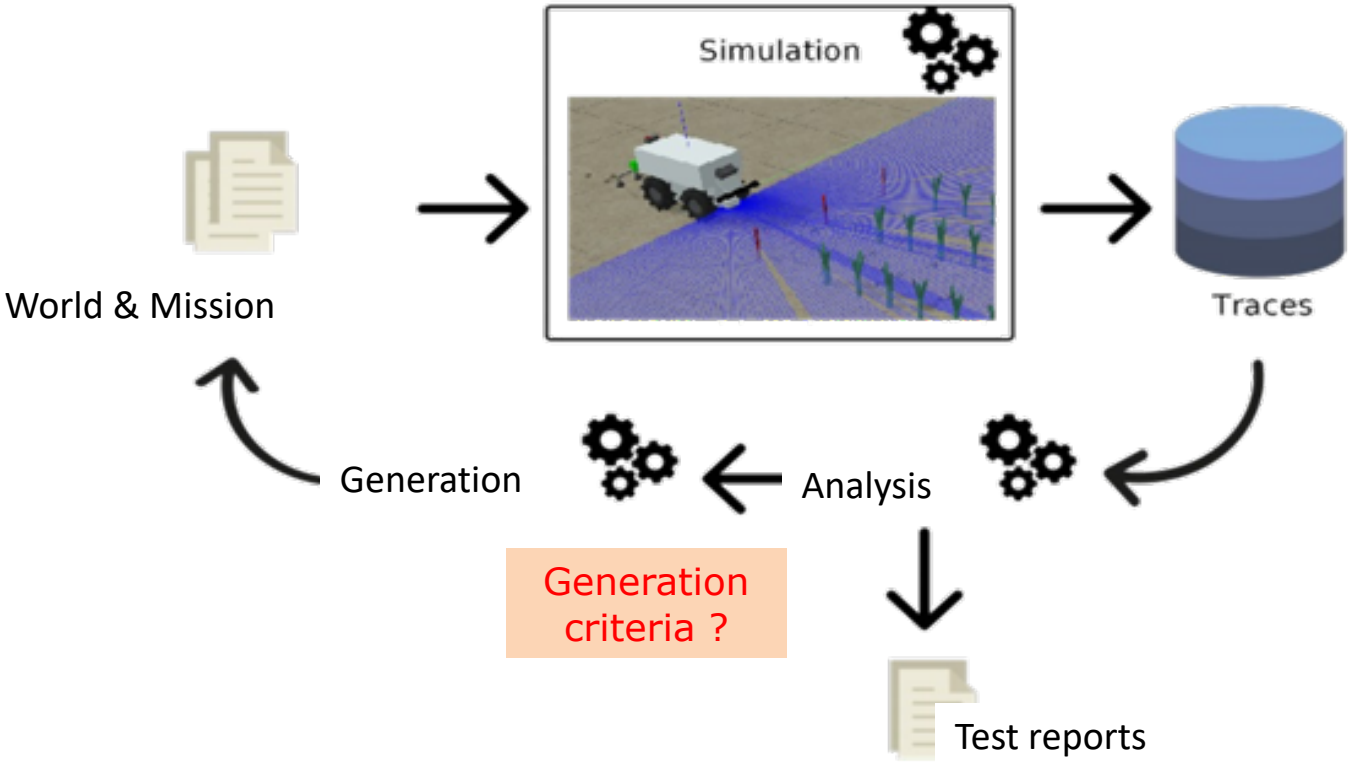
**SUT : System Under Test*



naïo Technologies

taris
Testing Agricultural Robots In Simulation

Testing in simulation (Sw in the loop) - Challenges



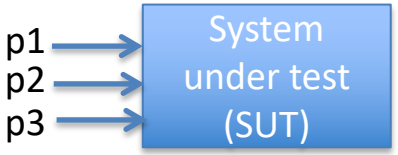
Generation criteria ? = Test selection

Test selection

- Which criteria to select test cases ?
  - Search-based (meta-heuristics, e.g., genetic algorithms) based on fitness functions (e.g., distance to the crops, or code coverage)
  - Combinatorial : optimize situation coverage, e.g. pair-wise testing

Pair-wise testing

- Test all pairs of values
- E.g. consider 3 boolean variables p1, p2, p3



p1	p2	p3
0	0	0
1	1	1

All values appear once
-> Not effective

p1	p2	p3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

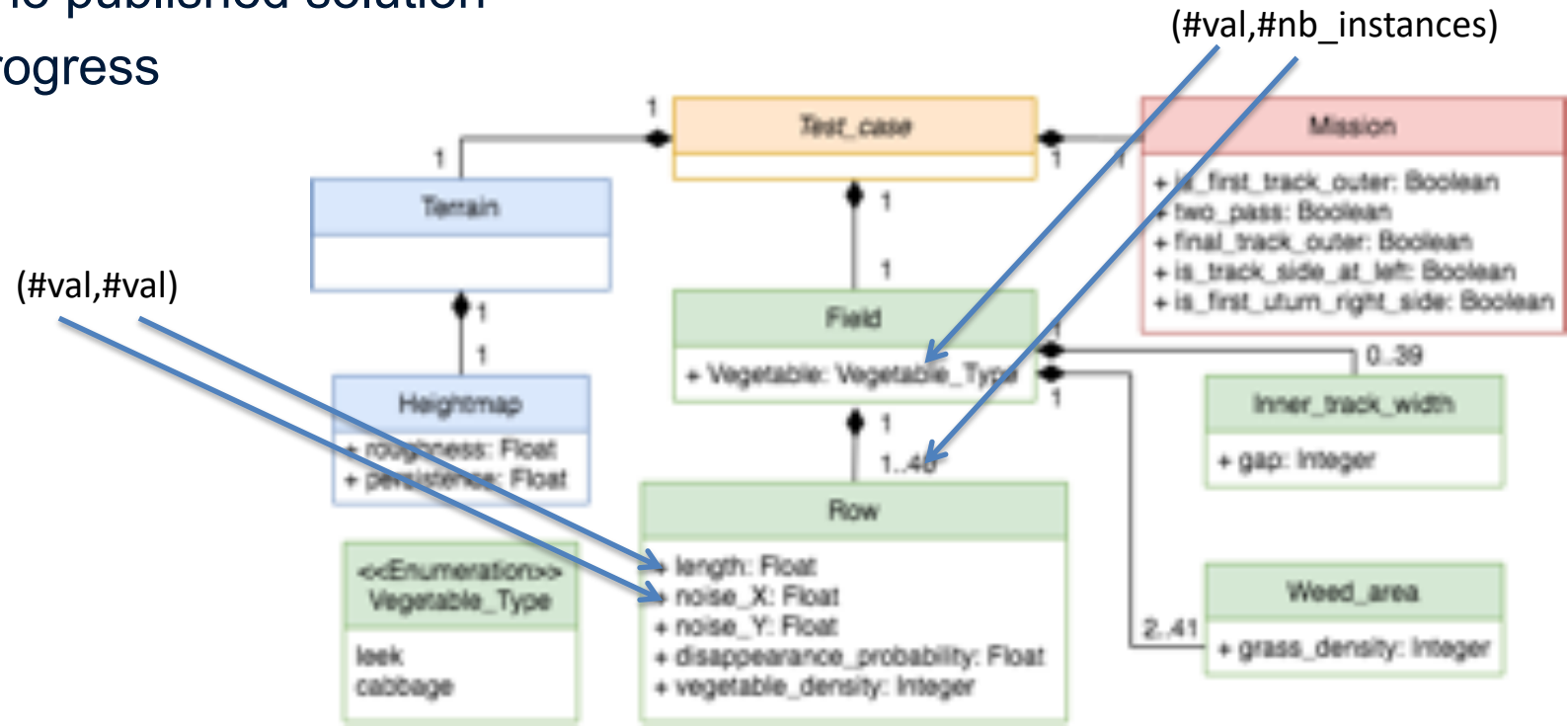
All combination appear once -> not efficient

p1	p2	p3
0	0	0
0	1	1
1	0	1
1	1	0

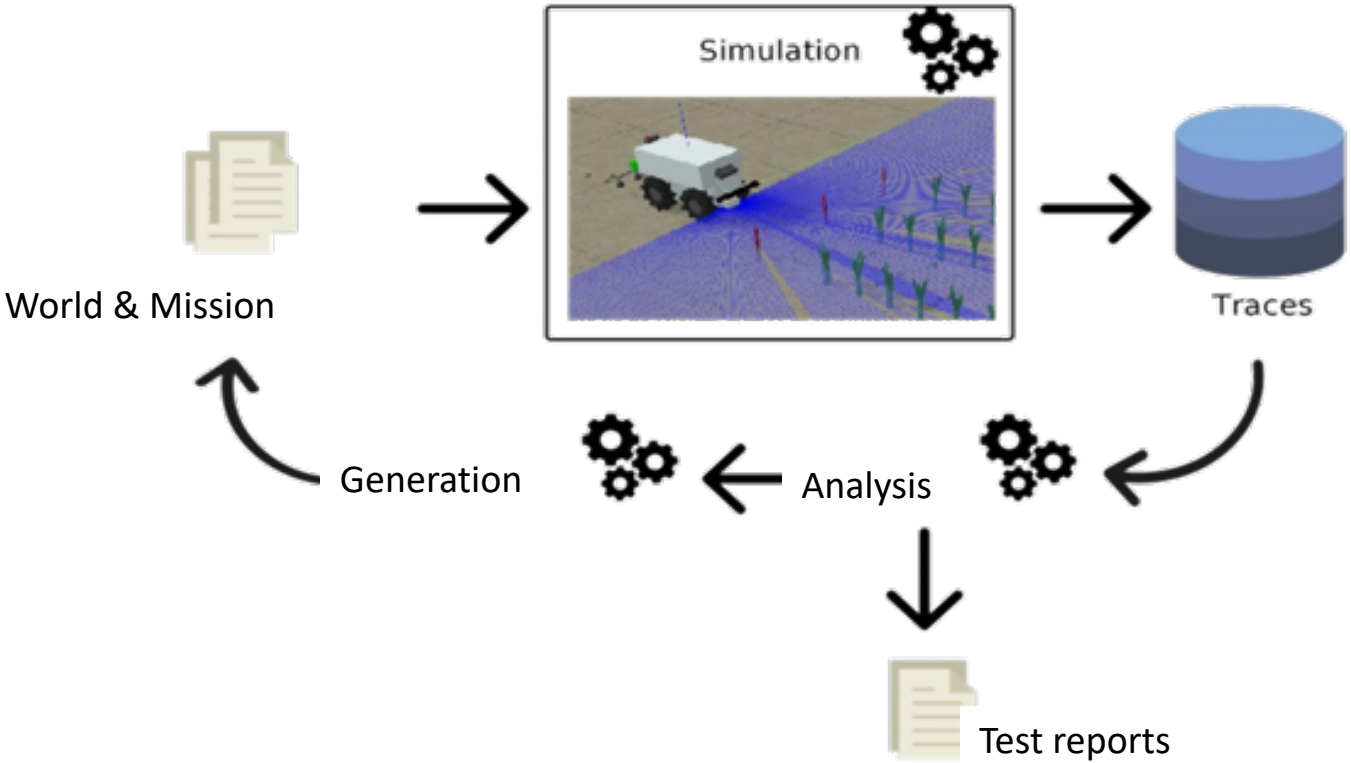
All value pairs appear once
-> yeah !

Pair-wise testing for structured and multi-instances data

- Currently no published solution
- Work in progress



Final objective (in 1 or 2 years)



Conclusion

- Validation based on testing in simulation under investigation
- First results on simulators fidelity, reproducibility, test case generation
- On-going work on test selection
- Application to various case-studies (2 projects under selection : test in simulation of underwater robot & fishes, test in simulation of mobile robots for train inspection)

> THANKS !

> Test our tool (opensource/python)

- https://www.laas.fr/projects/taf/
- Video tuto / examples / etc.