FONDAZIONE BRUNO KESSLER

SOLENIX

TRASYS INTERNATIONAL
BUSINESS UNIT OF NRB

Verification and Validation of Autonomous Systems

ESA AO/1-10409/21/NL/MGu

Study Overview

"to propose and demonstrate a **generic Verification and Validation methodology** based on the usage of the **System-level Simulation** Facilities, specifically targeted at **autonomous systems** using **AI-models**"

Focus

System level V&V of autonomous systems integrating AI/ML components

Problem

AI/ML approach very effective for implementing autonomy, but not suitable "as-is" for mission-critical systems

Approach

Define a Generic Quality Framework for AI/ML models and a Verification & Validation Methodology based on System-Level Simulation Facilities

Expected Outcome

- Quality Framework
- Generic V&V Methodology
- Demonstrator on relevant use case

Consortium

Consortium

SOLENIX

Solenix Engineering GmbH (DE)
Prime Contractor

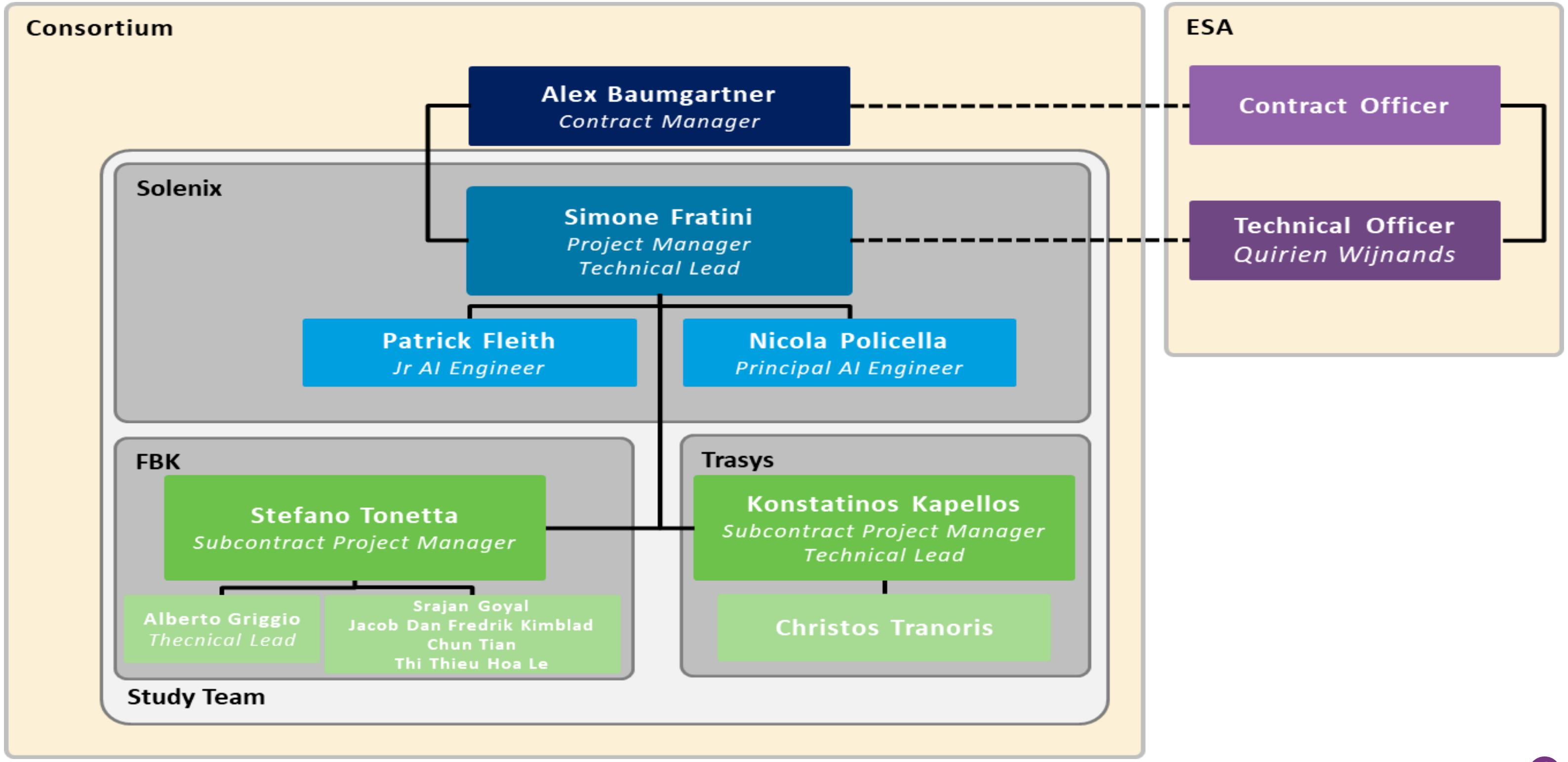
FONDAZIONE BRUNO KESSLER

Fondazione Bruno Kessler
Subcontractor

TRASYS INTERNATIONAL BUSINESS UNIT OF NRB

Trasys International
Subcontractor

Study Team



Consortium

Alex Baumgartner
Contract Manager

Study Team

Solenix

Simone Fratini
Project Manager
Technical Lead

Patrick Fleith
Jr AI Engineer

Nicola Policella
Principal AI Engineer

FBK

Stefano Tonetta
Subcontract Project Manager

Alberto Griggio
Thecnical Lead

Srajan Goyal
Jacob Dan Fredrik Kimblad
Chun Tian
Thi Thieu Hoa Le

Trasys

Konstatinos Kapellos
Subcontract Project Manager
Technical Lead

Christos Tranoris

ESA

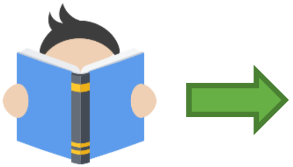
Contract Officer

Technical Officer
Quirien Wijnands

Summary of Project Activities

Task 1 – Quality Model & V&V Philosophy

- Analysis and report on:
  - AI-model life-cycles (Quality Model and metrics)
  - Available software frameworks to support the development, verification, and validation of AI-models
  - Current common practises of testing, verification and validation of autonomous systems making use of AI
  - Usage of system-level simulators for the verification and validation purposes of AI-models
- Investigation and proposal of a verification and validation philosophy

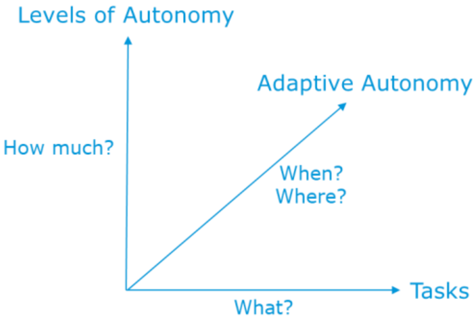


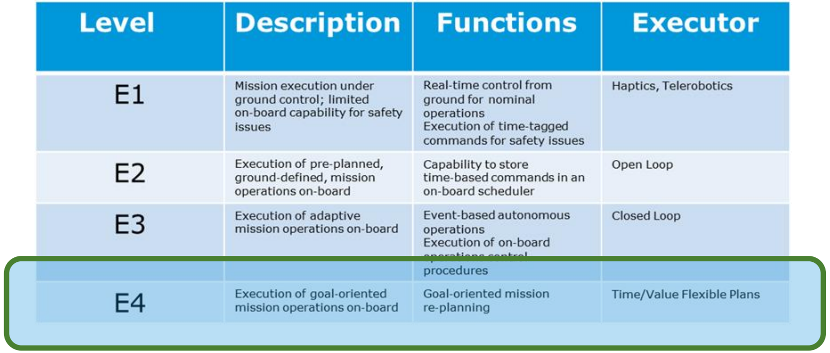
D1 - Analysis Report on Verification and Validation of AI using Simulators

Autonomous Systems in Space Exploration

- Motivations:
  - communications constraints and latency
  - uncertainty in the operational scenarios
  - limited access and availability

- Opportunities:
  - New Scenarios
  - Opportunistic Science
  - Increase Flexibility & Robustness
  - Reduce Costs





Level	Description	Functions	Executor
E1	Mission execution under ground control; limited on-board capability for safety issues	Real-time control from ground for nominal operations Execution of time-tagged commands for safety issues	Haptics, Telerobotics
E2	Execution of pre-planned, ground-defined, mission operations on-board	Capability to store time-based commands in an on-board scheduler	Open Loop
E3	Execution of adaptive mission operations on-board	Event-based autonomous operations Execution of on-board operations control procedures	Closed Loop
E4	Execution of goal-oriented mission operations on-board	Goal-oriented mission re-planning	Time/Value Flexible Plans

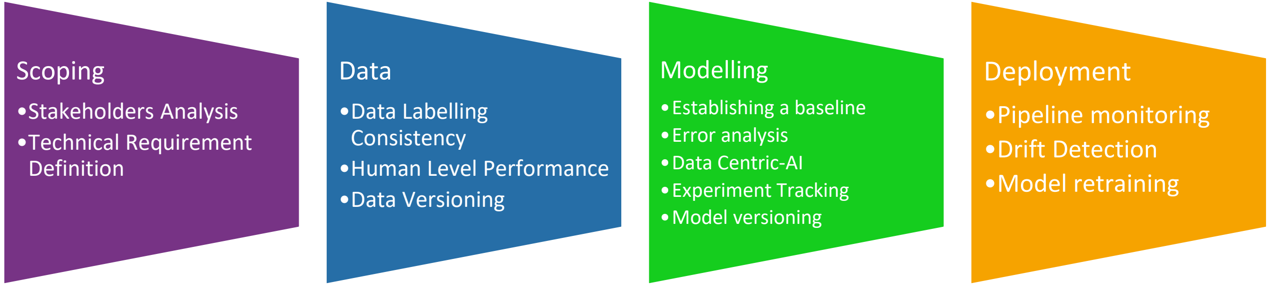
ECSS autonomy levels

SoA on AI/ML use for Autonomy In Space

- Surveyed platforms:
  - The Remote Agent Experiment (NASA/JPL, 1999)
  - MER Exploration Rovers (NASA/JPL, 2004)
  - The autonomous science experiment on NASA’s EO-1 (NASA/JPL, 2005)
  - GOAC (ESA, 2011), GOTCHA (ESA, 2016) & ERGO (ESA, 2018)
  - Intelligent Payload Experiment (IPEX) (NASA/JPL, 2013)
  - IRONCAP (ESA, 2012)
  - OPS-SAT Autonomous Experiment (ESA, 2019)
  - MARS2020 Perseverance Rover (NASA/JPL, 2020)
  - Robotic Digital-Twin, ROBDT (ESA, 2021)

Review - AI-model life-cycles (Quality Model and metrics) 1/3

- AI/ML project life cycle that integrates the V&V process
  - Emphasis in Data-Centric AI methods, and Machine Learning Model Operationalization Management (MLOps).

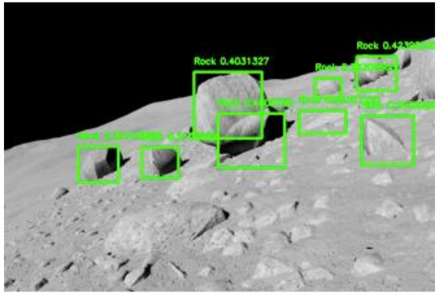


Scoping
- Stakeholders Analysis
- Technical Requirement Definition

Data
- Data Labelling Consistency
- Human Level Performance
- Data Versioning

Modelling
- Establishing a baseline
- Error analysis
- Data Centric-AI
- Experiment Tracking
- Model versioning

Deployment
- Pipeline monitoring
- Drift Detection
- Model retraining



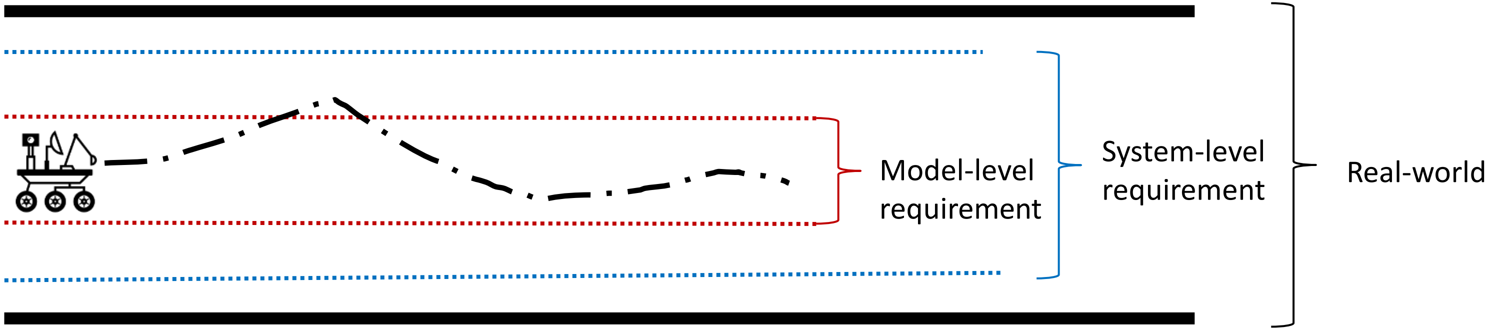
Draft D1 – Review - AI-model life-cycles (Quality Model and metrics) 2/3

- **Properties of V&V AI/ML models**
  - Robustness,
  - Stability,
  - Explainability,
  - Performance,
  - Compute Resources Usage,
  - Traceability,
  - Repeatability,
  - Reproducibility
  - Maintainability
  - Criticality

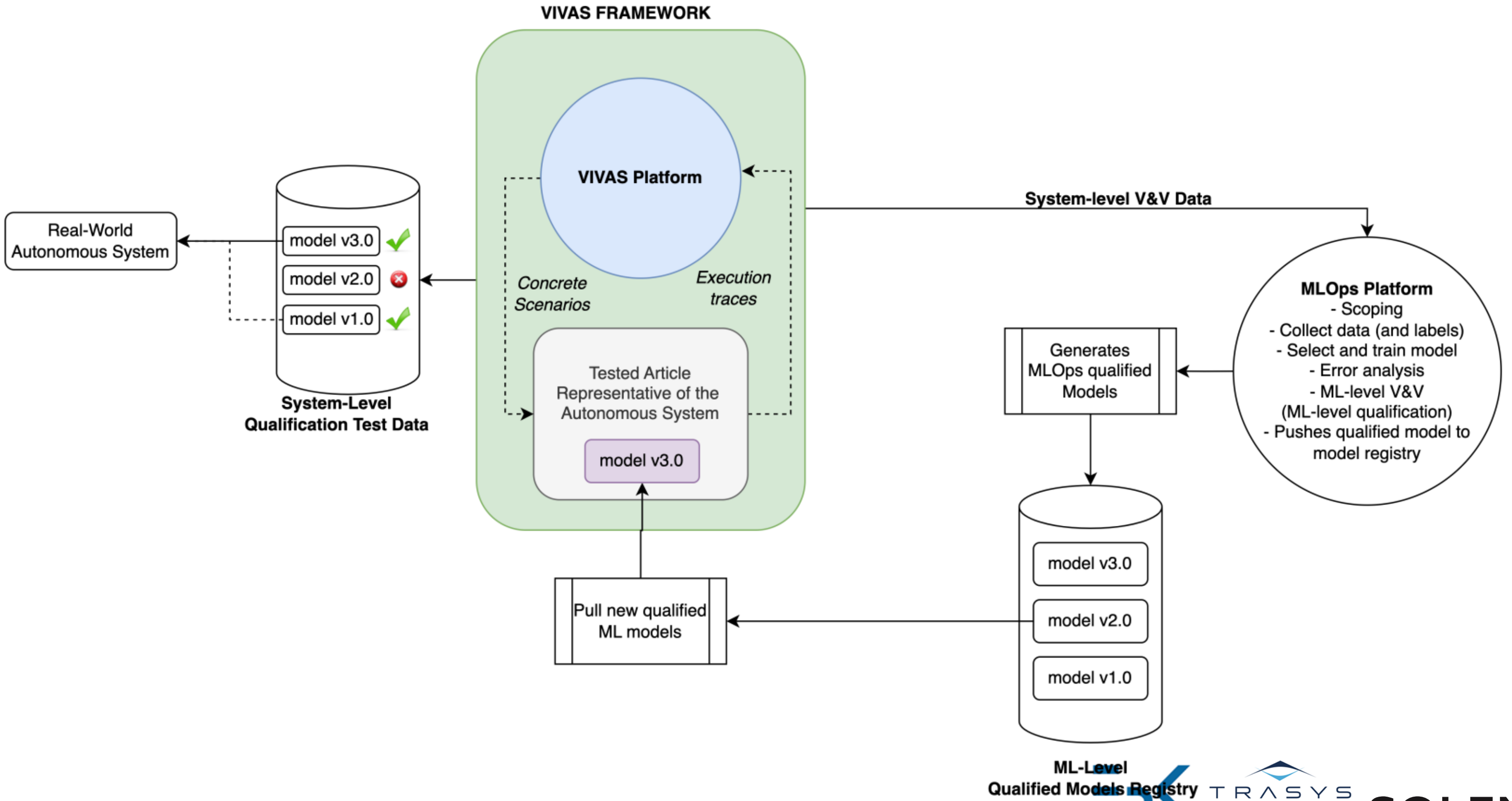
@ AI/ML Model level

Draft D1 – Review - AI-model life-cycles (Quality Model and metrics) 3/3

- **Mapping of model-level properties to system-level properties**
  - Some system-level metrics can be very similar to model-level metrics
- **Example: Path Finder Algorithm**
  - Model level metric: minimum clearance distance to obstacle of 100cm
  - System-level metric: minimum clearance distance to obstacle of 50cm
  - A useful tool to map lower-level properties to system-level properties can be to plot a dependency graph



VIVAS Framework Interface with Model Development Life-Cycle


VIVAS FRAMEWORK
VIVAS Platform
Concrete Scenarios
Execution traces
Tested Article Representative of the Autonomous System
model v3.0
Real-World Autonomous System
model v3.0 ✓
model v2.0 ✗
model v1.0 ✓
System-Level Qualification Test Data
System-level V&V Data
MLOps Platform
- Scoping
- Collect data (and labels)
- Select and train model
- Error analysis
- ML-level V&V
(ML-level qualification)
- Pushes qualified model to model registry
Generates MLOps qualified Models
model v3.0
model v2.0
model v1.0
ML-Level Qualified Models Registry
Pull new qualified ML models

SoA on V&V for Autonomous Systems using AI

- Review of the state of the art along 3 dimensions:
  - V&V of ML models in isolation (especially NNs)
  - V&V of planning components in autonomous systems
  - System-level V&V of autonomous systems with AI/ML components
- Analysis of two simulation-based V&V frameworks for autonomous systems with AI – same philosophy as VIVAS
  - VerifAI
  - Pegasus

SoA on V&V for Autonomous Systems using AI

- V&V of ML components in isolation
  - Multiple approaches considered:
    - Formal methods
    - Testing
    - Quantitative verification
- V&V of planning components in autonomous systems
  - Focus on testing-based methods
  - Discuss the problem of coverage

SoA on V&V for Autonomous Systems using AI

- System-level V&V of autonomous systems with AI/ML components
  - Identify key challenges (e.g. lack of formal models, uncertainties in ML behaviors, need for adequate env models, heterogeneity)
  - Focus on technique based on simulation-based testing and scenario generation
    - Analysis of two frameworks similar to VIVAS (but in automotive domain)
      - VerifAI - https://berkeleylearnverify.github.io/VerifiedAIWebsite/
      - Pegasus - https://www.pegasusprojekt.de/en/

SoA on System Level Simulators for V&V

- SCENIC: Domain-specific scenario description PP language to specify environments
- CARLA & LGSVL: Urban driving simulators
- VIRES: used in the PEGASUS project
    - *OpenDRIVE*: open format specification to describe a road network's logic
        - Hierarchy, …, various types of lanes, signs and signals incl. dependencies, …
    - *OpenCRG*: defines a file format for the description of road surfaces
    - *OpenSCENARIO*: defines a file format for the description of the dynamic content of driving and traffic simulators
        - … to describe complex, synchronized manoeuvres that involve multiple entities like vehicles, pedestrians and other traffic participants,
        - ... on driver actions (e.g., performing a lane change),
        - … storyboard, which is subdivided in stories, acts and sequences.

Task 2 – Use case Identification & VIVAS Architectural Design

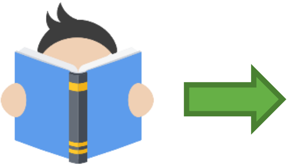
- Objectives:
  - Architectural Design of the V&V Framework
  - Identification of the Use Cases
  - Simulator Design for the Use Cases

D2 - ViVAS: Identification and definition of the Use-Case

D3 - ViVAS: architectural design of the Verification and Validation framework

Task 3 – PoC Implementation & Demonstration

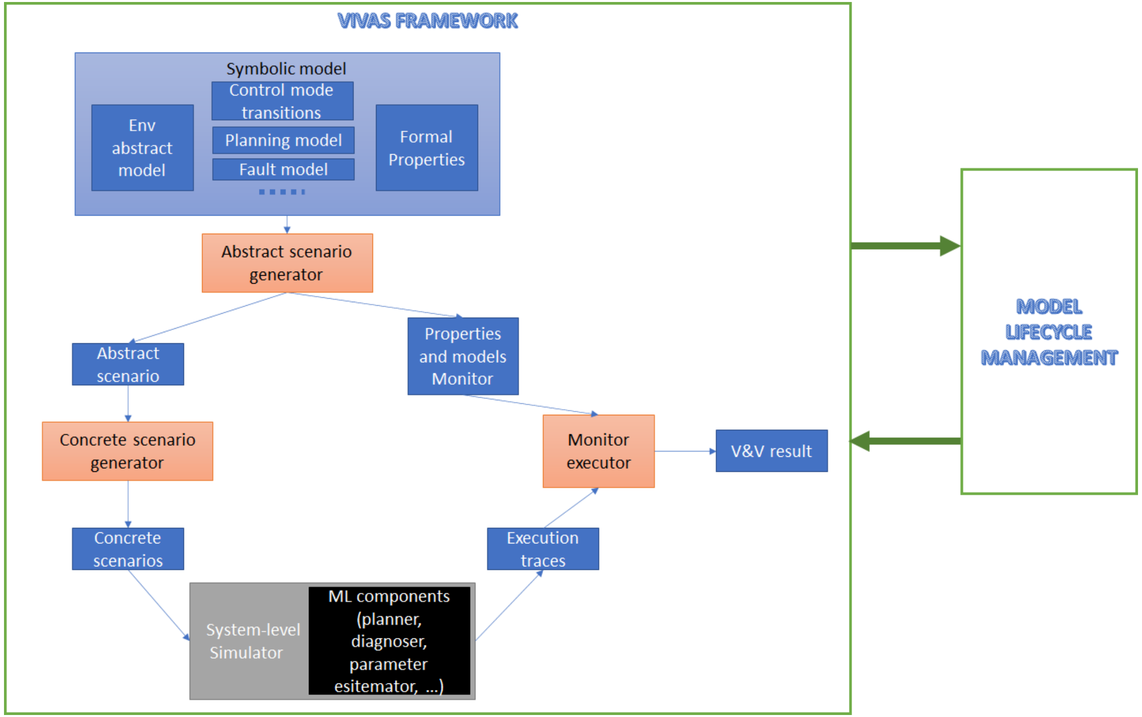
- Objectives:
  - Implementation of the V&V Framework
  - Implementation of the Use Cases
  - Simulator Adaptation for the Use Cases



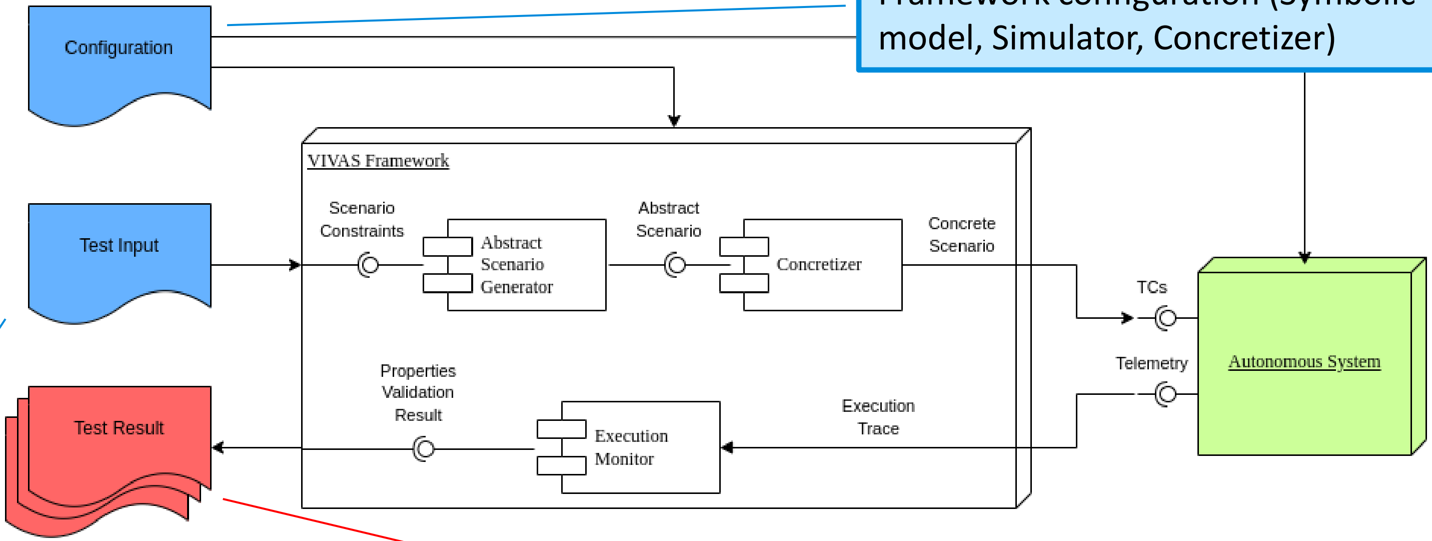
D4 - PoC Detailed Design

D5 – PoC Test Report

VIVAS Architectural Design & Implementation



VIVAS Implementation Overview

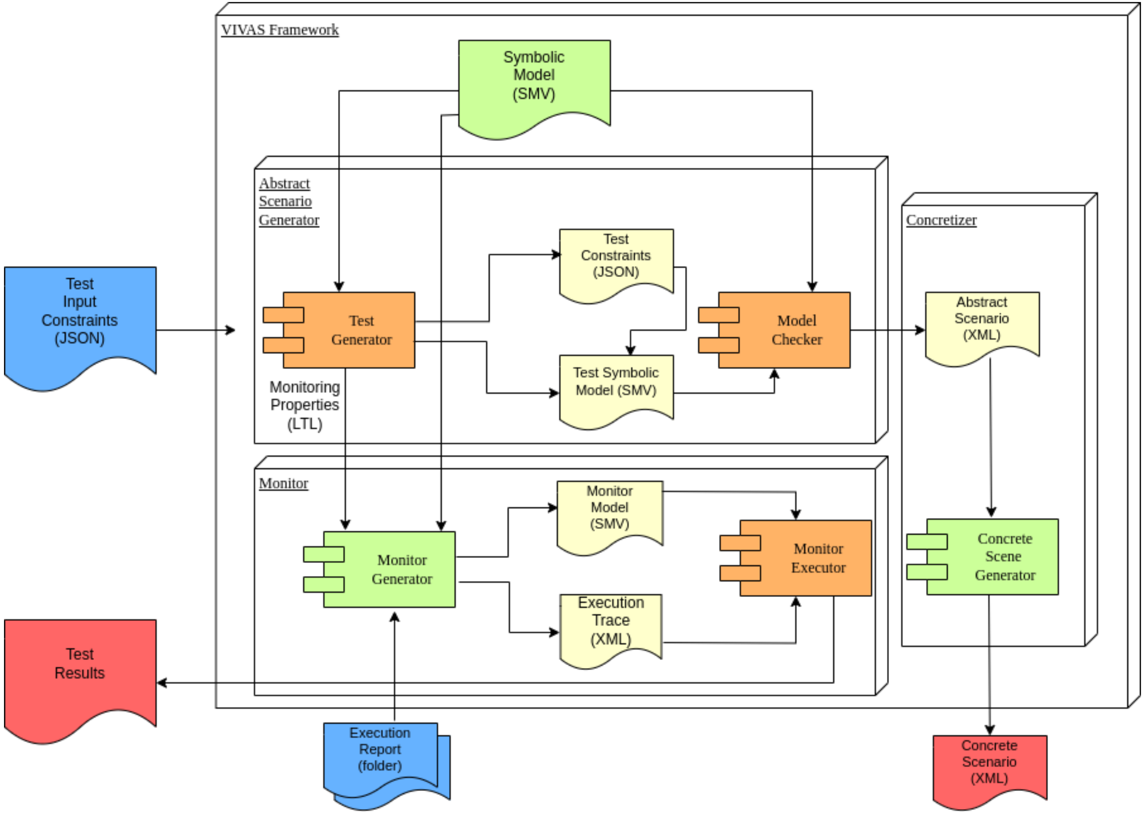


Framework configuration (Symbolic model, Simulator, Concretizer)

Test generation configuration: defines the space for test constraints and coverage conditions

Framework execution results: concrete test cases, coverage report

VIVAS Framework Components and Interfaces



🟦 User Inputs

🟥 Output artifacts

🟨 Internal artifacts

🟧 Generic components

🟩 Use-case specific components

Symbolic Model

- Defines the abstract representation of the system under test
    - autonomous system and its environment
- Used for generation of abstract test cases
- Written as a symbolic transition system in SMV language
    - Typically by a domain expert (possibly aided by a FV expert)
    - **Necessarily use-case specific**

Abstract scenario generator

- Generates abstract test cases via symbolic model checking, based on:
  - Symbolic model
  - **Test input file,** specifying the space of the constraints for test generation
    - N-dimensional domain of predicates on the symbolic model
    - One abstract test per point in the domain

Concrete Scene Generator

- Translates an abstract scenario to a concrete set of inputs for the system-level simulator
  - From execution trace of the symbolic model to specific input format for the system-level simulator
  - Can be a 1:N mapping
    - In cases in which the abstract model is approximated
      - Example: from a discrete orbital position to a set of curves for temperatures and energy fluxes, with concrete values obtained via sampling
  - **Necessarily use-case specific**

Property Monitor

- Checks whether the execution traces of the system-level simulator satisfy the property of interest
  - To detect interesting test cases leading to property violations

- Checks that the execution traces comply with the abstract constraints used for scenario generation
  - To measure the actual coverage of the test suite in the concrete executions

- **Generic part** (based on the NuRV symbolic monitor) **+ use-case-specific part** (for translating simulator output to NuRV model)

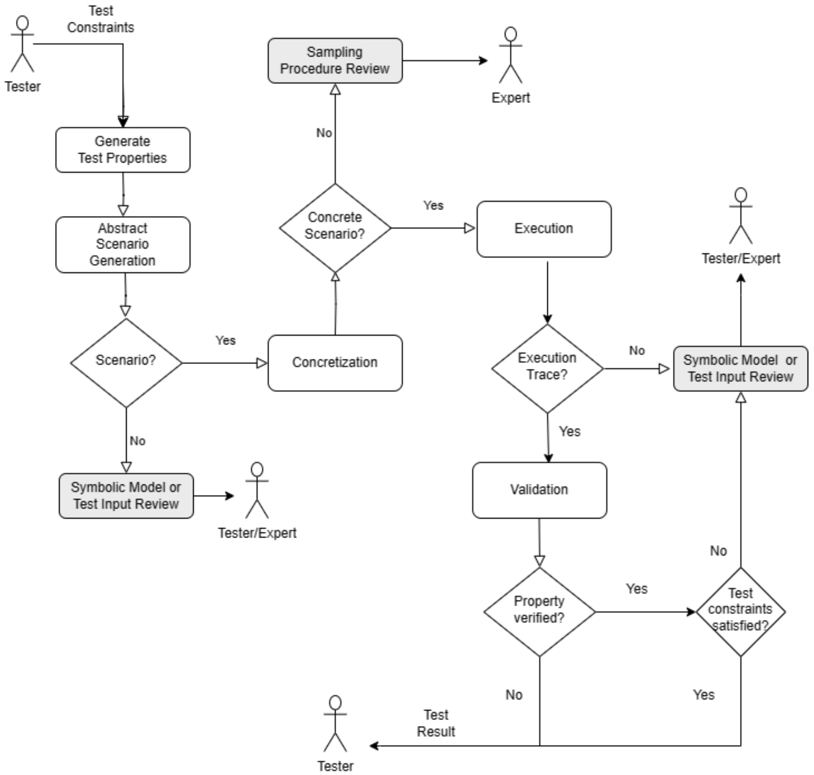
Property Monitor and Coverage Check

- **Abstract scenarios:** enumerated wrt the test input space definition
  - They cover the constraint space by construction
- **Concrete scenarios** are also **expected** to cover the (concretization of) the constraint space
  - The VIVAS monitor can check the simulation runs to ensure that this is the case
  - Needs a proper definition of the meaning of the abstract constraints in the concrete simulation runs
    - What the symbolic model constraints mean in the real execution

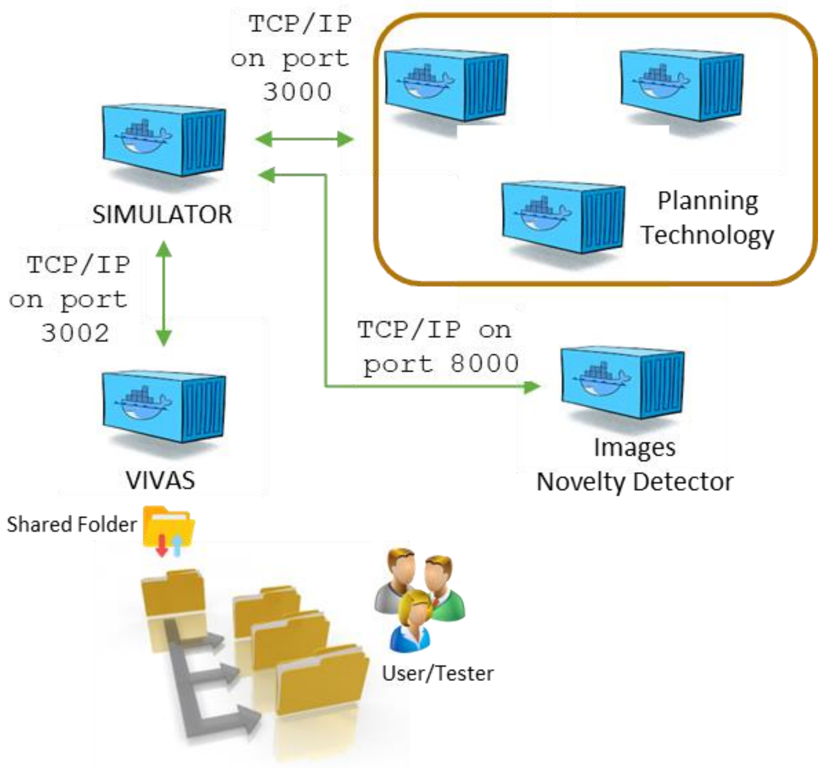
Property Monitor – Test Results

- For each test case, 4 possible outcomes:
  - **Property satisfied:**
    - Scenario constraints satisfied: **OK**, test works as expected
    - Scenario constraints violated: **WARNING**, problem with coverage
  - **Property violated:**
    - Scenario constraints satisfied: **KO**, found problematic scenario
    - Scenario constraints violated: **WARNING**, found unexpected problematic scenario

VIVAS Framework – User Interaction Summary



Deployment



- VIVAS is deployed in 6 docker containers:
  - VIVAS-Framework
  - Planning Technology:
    - WIA Planner
    - RaaS Service
    - DHS Model
  - 3dROV Simulator
  - Novelty Detection Model

- Shared folder for I/O
- Python script to launch tests

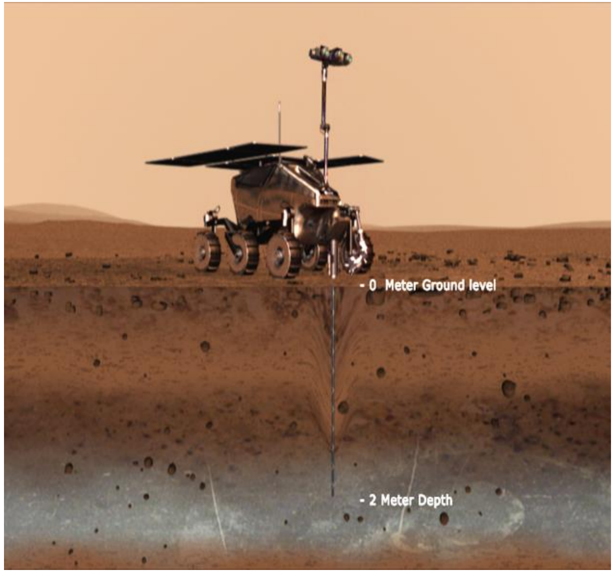
Use Case Scenario

- Scenario:
  - Planetary robotic asset
  - “Goal Oriented” Behavior (E4)
  - Use of AI/ML models
  - Strong interaction with the environment

- **Focus on the suitability for V&V**
- **Based on the simulator**

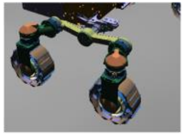
- Use Case: executing activities subject to
  - Resource availabilities
  - Execution uncertainties

- Validation of System Level Properties

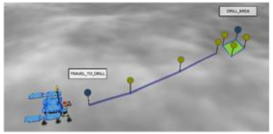


Autonomous System Model

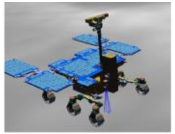
- Rover Moving Over a Grid
- Performing science in different locations
- Under different environmental conditions



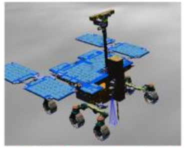
ADE Warm-up
(ML Model – resources estimation)



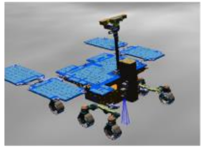
Travel to science of interest
(ML Models – Wheel/Terrain interaction, opportunistic science)



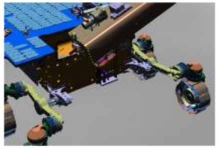
Science - reconfiguration
(FDIR-PM - Synchronization)

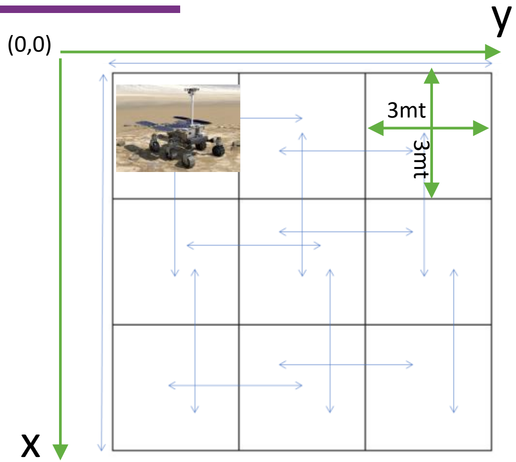


Drill – Sample collection
(FDIR-PM)



Science
(What-if)

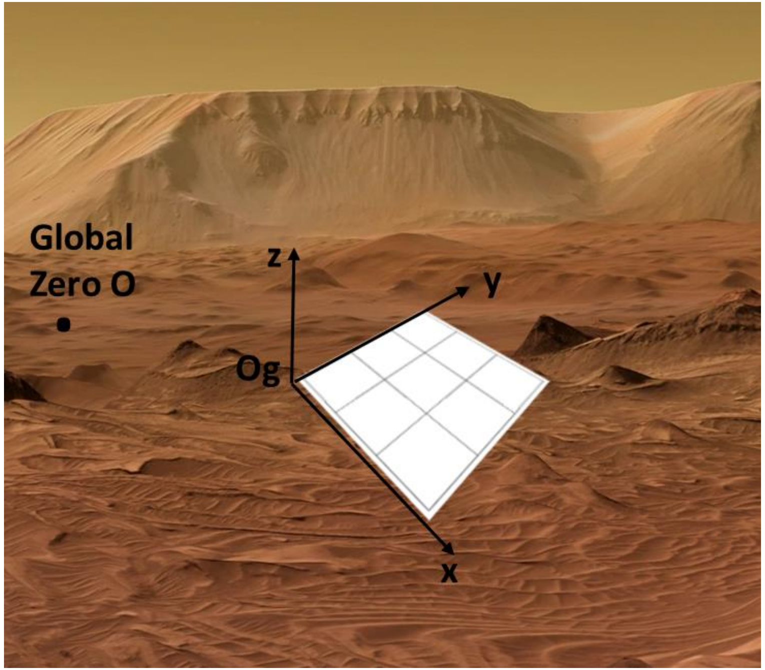




1. RV_WakeUp()
2. GNC_FPATH_Straight(2,0.0,0.0,3.0,0.0,...,32.0)
3. CLUPI_AcquireZScience(0,0,1,1,...,1)
4. Collect_Sample(4)
5. CLUPI_AcquireZScience(0,0,1,1,...,1)
6. Deliver_Sample()
7. GNC_FPATH_Straight(2,0.0,0.0,3.0,0.0,...,32.0)
8. ...
9. Deliver_Sample()

Mapping grid on Terrain

- Global Zero “O”:
  - SYS_ROVER_LONGITUDE = λ (O)
  - SYS_ROVER_LATITUDE = Φ(O)
- Grid & Rover Centered in Og:
  - LocX, LocY, LocZ
  - RotX, RotY, RotZ
- Og at the top left of location [0,0]
  - ENV_ROVER_POS = POS_0_0
- Objects relative to O

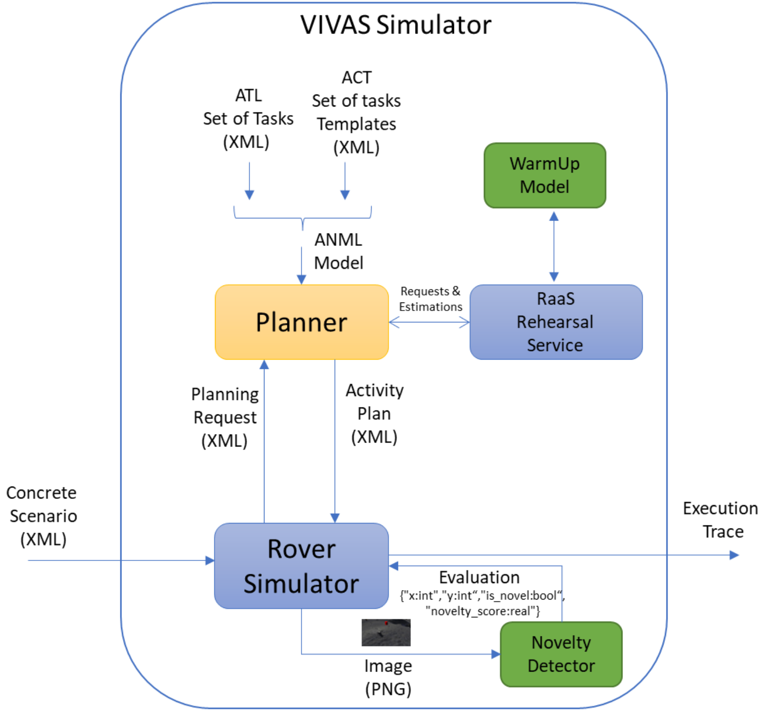


Design & Implementation of the Autonomous System

- Simulators:
  - Rover Simulator
  - Plan Rehearsal

- ML Models:
  - WarmUp (U/C 1 & 2)
  - Image Novelty Detector (U/C 2)

- Planner



Simulator Design for the Use Cases

- Starting point: 3DROV simulator with ExoMars models
- Rover Models:
  - Power, Thermal, etc. maintained,
  - Controller model updated to include the Activities related to the target 'scenario'
    - Interfaced with the 'Automatic Activities Planner' (WIA)
    - Interfaced with the 'Novelty detection' ML model
- Payload Models: removed the models not included in the target 'scenario'
- Environment Models:
  - *Atmosphere*: outputs generated by the 'input concrete scenario'
  - *Terrain*:
    - Used, off-line, to generate images to train the 'Novelty detection' ML model
    - During simulation, to generate images based on the 'input concrete scenario'
- Log
  - Subsystem state evolution maintained with very minor updates (WSD)
  - Reports: updated to include trace of all the exchanges with the external systems

U/C 1 – “Resource Allocation Estimation”

- Objectives:
  - Test the “quality” of a resource estimator ML model in terms of:
    - Accuracy in production/consumption estimation
    - Dependency from environmental conditions
    - Support for different behaviors of the autonomous system
  - Test of the capability of VIVAS in supporting the above analysis

Validation

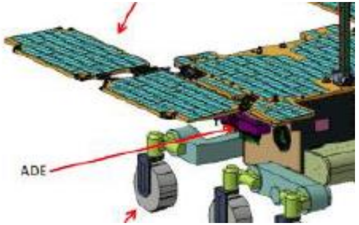
- The rational is estimate “how good” is the prediction to support system-level beahviour

- Mission achievement
  - Can the plan be successfully executed in within the mission horizon?

- Can the plan be successfully executed with the available resources?
  - Resource allocation
  - Is the allocation congruent with an efficient use of the asset?

Resource Estimator Model (from RobDT)

- **Overall Goal**: improve reliability and precision of on-ground planning and forecasting in order to aid decision-making
- **Idea**: maintain a model-based approach, but allow for parameters that are to be estimated/learned from the telemetry data
- **Concrete case-study**: Warm-Up activity of the Actuator Drive Electronics (ADE)
  - ADE shall be heated before activating the motors of the rover
  - Duration of warm-up depends on external atmospheric conditions and is important from planning and what-if analysis
  - Use telemetry data to estimate the expected duration of the warm-up a
  - The planning model will mix learned elements and symbolic knowledge will be able to invoke the estimation while planning
  - Planning model evolves together with the DT



Warm-Up model - Features

- Training:
  - Longitude [0,360]
  - Latitude [-90,90]
  - Ls: The solar longitude
  - LTST (Local True Solar Time)
  - Air Temperature as given by the Mars Climate Database (MCD)
  - Scenario (such as ‘worm scenario, cold scenario, dust scenario, etc)

- ADE Warm-up duration evaluated by an engineering formula established during the rover engineering activities

- Prediction:
  - the location (longitude/latitude) the rover operates
  - the season in the year (Ls)
  - the time on the sol (Ltst)

U/C 1 - Methodology

- System Level Properties Considered:
  - Mission achievement
  - Makespan
- Test the accuracy in resource estimation
  - Run the autonomous system with the ML model estimator
  - Run the autonomous system without the ML model estimator
  - Compare the results
- Test the dependency from environmental conditions
  - Run the same set of goals under different:
    - Start Time of the Day
    - Soil Type
    - Location
  - Observe and compare:
    - plan execution time
    - resource consumption
- Support different behaviors of the system:
  - Test different set of goals:
    - In situ drilling & analysis
    - Move, drill & analyze
  - Check achievement

U/C 1 – Results (1/2)

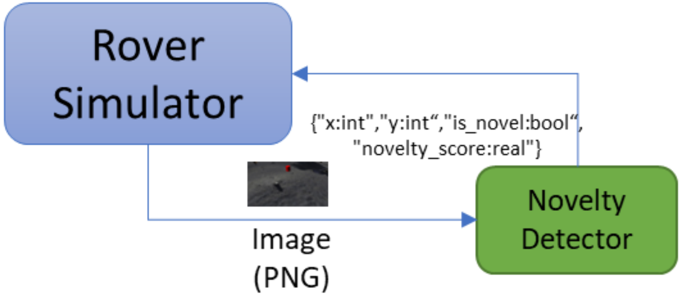
- The first set of tests is done with 3 different goals:
  - Collect a sample
  - Move, and collect a sample
  - Move on different path and collect the sample.
- Tests result showed that the estimation was adequate for sample collection activities but seems to overestimate time/resources when rover moving activities are involved, especially on good illuminating conditions
- Analysis:
  - The time/resource estimation via ML seems accurate for scientific activities but not for rover moving activities. It is then suggested to re-train the model considering wider rover traverses
  - The time/resource estimation seems more accurate in poor illuminating conditions, probably underestimating the power generated by the solar panels in plain sun. It is suggested to check the estimation of the solar panels power generation.

U/C 1 – Results (2/2)

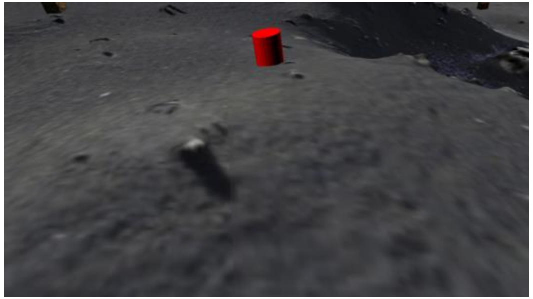
- The second set of tests was done to estimate the dependency of the evaluation from the soil type.
  - Results showed that durations and power consumptions are correctly estimated for different soil types.
- The third set was conducted on different locations on the map.
  - Results showed that the dependency is related to the different illuminating conditions given by the different locations, hence results and analysis are in line with those obtained with the first set of tests

U/C 2&3 - Novelty Detection & Opportunistic Science

- Rover Moving Over a Grid to perform science operations
- Acquiring Images on the path
- Analyzing Images to identify novel objects
- Take Actions on Novel Object



- Strategies:
  - Identify & Report (U/C 2)
  - Analyze (U/C 3)



Novelty Detection ML Model

Training / Validation Data

- **Training:** Zone 1, explored terrain with typical features:1200 images, i.e. 79200 patches
- **Validation:** Zone 2: unexplored terrain with typical features: 1200 images, i.e. 79200 patches
- **Test:** Zone 2: unpexplored terrain with novel feature: 219 novel patches

- **Varying parameters:**
  - Position on grid
  - Rover pose
  - Sun elevation
  - Camera direction (front, rear, right, left)

Model Architecture

Convolutional Auto-Encoder, 2938 trainable parameters

Layer	Dimensions
Input Patch	**112 x 112 x 3**
Conv2D + MaxPool	56 x 56 x 12
Conv2D + MaxPool	28 x 28 x 8
Conv2D + MaxPool	14 x 14 x 3
Conv2D + Usampling	28 x 28 x 8
Conv2D + Usampling	56 x 56 x 12
Conv2D + Usampling:	112 x 112 x 3
Reconstructed Patch	**112 x 112 x 3**

U/C 2&3 - Novelty Detection ML Model

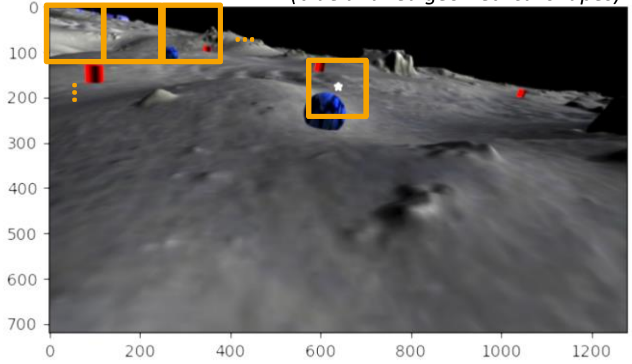
- **Inference Pipeline**

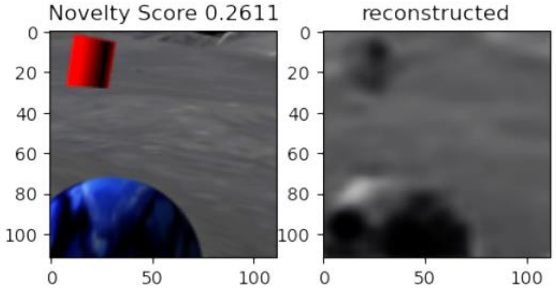
  For each image

  - For each Patch:
    - Feed patch to the CAE NN
    - Output a reconstructed image
    - Compute dissimilarity (novelty) with SSIM loss
  - Check the most “novel” patch in the image and classify image ”is_novel = True” if novelty > 0.25
  - Return image x, y center coordinates of the most novel patch:

```
{"is_novel": True, "score": 0.279, "coordinates": [(x1, y1), (x2, y1), (x1, y2),
(x2, y2)]}
```

Example Image with Novelties (blue and red geometrical shapes)





U/C 2 – Novelty Detection in Images

- **Objectives:**
  - To assess the **system-level performances** of the autonomous system in terms of capabilities to identify novel targets of interests for possible opportunistic science scenarios:
    - **Detection Probability**
      - *Given a novelty in the vicinity of the autonomous system, what is the probability that the system will identify and report this region of interest as "novel"?*
    - **False Alarm Rate:**
      - *Given that there are no novelties in the vicinity of the autonomous system, what is the probability that the system wrongly reports a region of interest?*

U/C 2 – Test Cases

- **Detection Probability**
  - 2 travel paths
  - 3 Time of the day
    - (illumination condition affecting images)
  - 6 positions for a novel target
    - (different backgrounds and distance to camera)
  - 3 types of targets

108 simulations scenarios.

- **False Alarm Rate**
  - Rover to visit all tiles on the terrain.
  - No target of interest placed on the terrain

135 images captured and analyzed by the autonomous system.

U/C 2 – Test Results

- **Detection Probability**
  - 108 simulations scenarios.
    - 60 detections
  - **System-Level metric: 0.55 (60/108)**
  - (Model-Level spec): 0.80
    - Delta due to the uncertainties and inaccuracies that emerge at system level

- **False Alarm Rate**
  - 132 images captured
    - 5 False detections
  - **System-Level metric: 0.04 (5/132)**
  - (Model-Level spec): 0.16
    - This better metric at system-level should be taken with a word of caution given the limited size of the terrain used for demonstration purpose (3x3) grid

U/C 2 – Test Results

- **Detection Probability**
  - **System-Level 0.55 vs 0.80 (Model-Level)**

- **False Alarm Rate**
  - **System-Level metric: 0.04 vs 0.16 (Model-Level)**

Overall test outcome:

- At system-level, this initial Verification & Validation Test Campaigns shows lower (better) false detection rate, but lower (worst) probability of novelties detection. Systems Engineer would have gain inside about:
  - Component-to-System **Delta Performances**
  - Compliance with their **system-level detection requirements**
  - Insight that it is **possible to adjust the detection threshold** (system-level false detection is better than expected) to increase the detection probability (which is significantly lower)
  - Possibility to carry out fine **error analysis**
- **Suggest to carry more tests with VIVAS and use the images generated during the test-campaign and either ignored or flagged as false-positive to re-train the model**

U/C 3 – Detect & Analyze

- **Objectives:**
  - To assess the capability of the autonomous system to support **opportunistic science**:
    - Identify novel targets of interests
    - Stop the activities and travel to the Region of Interest (RoI) identified
    - Carry on analysis in the RoI
    - Go back on the original path and resume the original plan
- **Tests**:
  - On different paths
  - With Object of Interest placed along and away from the path
- **Results**:
  - The system is able to support opportunistic science in within the boundaries tested (objects not too far and limited operational grid)
  - Further tests are suggested to analyse the behaviour of the system with false detections
  - Trajectory Optimization needed for practical applications

Conclusions

Achievements

- Demonstrated the **feasibility of a model-based approach to system-level validation and verification** of autonomous systems integrating AI/ML components

- Implemented the **VIVAS Framework**, a general, domain independent **support for qualification of AI/ML in operation**, leveraging the rigorousness provided by the model checking approach

- The test cases have been deployed using 3DROV and RobDT, ESA state-of-the-art technologies for autonomous planetary robotic assets

Recommendations

- Consider a follow-up:
  - to analyze the results of VIVAS runs providing more detailed insights on tests results
  - to integrate VIVAS in an MLOps loops for ML models qualification

- Investigate the customization of VIVAS in various scenarios:
  - Robotic
  - Autonomous Platforms for EO

- Consider the possible integration of VIVAS as a building block for future deployments like:
  - The DT (digital twin) infrastructure for Exomars
  - Exomars ground rover control infrastructure

Thank You

Solenix Engineering GmbH
Spreestrasse 3
64295 Darmstadt
Germany

info@solenix.de

www.solenix.de

FONDAZIONE BRUNO KESSLER

TRASYS INTERNATIONAL BUSINESS UNIT OF NRB

SOLENIX