RobDT: AI-enhanced Digital Twin for Space Exploration Robotic Assets

Marco Bozzano[1], Riccardo Bussola[1], Marco Cristoforetti[1], Srajan Goyal[1], Martin Jonáš[1], Konstantinos Kapellos[2], Andrea Micheli[1], Davide Soldà[1], Stefano Tonetta[1], Christos Tranoris[2], and Alessandro Valentini[1]

[1] Fondazione Bruno Kessler, Trento, Italy
{bozzano,rbussola,mcristofo,sgoyal,mjonas,
amicheli,dsolda,tonettas,alvalentini}@fbk.eu
[2] Trasys International, Bruxelles, Belgium
{konstantinos.kapellos,christos.tranoris}@trasysinternational.com

Abstract. Model-Based System and Software Engineering (MBSE) technology such as simulation has been adopted for decades by the space industry. During the lifecycle of a space mission a number of models are developed to support simulation and other analysis capabilities addressing needs specific for the different project phases. Typical concerns are: feasibility assessment, design optimization and validation, system performance and safety assessments, detail design verification and on-board software validation. In this context, symbolic and data-driven AI techniques can provide advanced capabilities to support the online operations of space missions. One of the main challenges to enable AI in the virtual flight segment is the problem of combining heterogeneous models in a common framework.

The ROBDT project aims at developing a Robotic Digital Twin framework that combines data-driven models, physics-based and symbolic models and uses online data and data analytics to adapt the models at runtime. The digital twin will support the robotic asset operations by providing timing and reliable prediction and by supporting what-if analysis to assess multiple scenarios. In this paper, we present the architecture of the ROBDT framework and the preliminary achievements.

Keywords: Digital Twins · Space Domain · Planning and Scheduling · Diagnosis · Monitoring.

1 Introduction

In this paper, we present the objectives and the early achievements of the ROBDT system, which is under development by the "Robotic Digital Twin" Activity funded by ESA and led by TRASYS in collaboration with FBK and GMV. In robotics, Digital Models of the target systems are traditionally used in all phases of a mission under the name 'Virtual Flight Segment' ranging from design to the development and the operations. These digital systems struggle to fully support their objectives in particular when the involved models are not able to capture the complete physical reality. This is particularly true when the operations environment evolves during the mission (e.g., when discovering a new planetary area).

The ROBDT activity proposes a new framework (see Figure 1) where engineering methods and AI techniques are integrated into a coherent Robotic Digital Twin Framework in order to allow:

- *On-line update of the system models*: The appropriate combination of data-driven and physics-based simulation models enables the application of on-line data analytics for adapting at runtime the models of the virtual asset guaranteeing a high-fidelity representation of the physical asset and its environment.
- *Planning and what-if analyses*: A digital twin enables planning of actions and what-if analyses based on more reliable models. These analyses allow to synthesize unexpected scenarios and study the response of the system as well as the corresponding mitigation strategies. This kind of analysis without jeopardizing the real asset is only possible via a digital twin.
- *Plan monitoring and fault diagnosis*: Telemetry data are monitored to detect and identify anomalies. Diagnosis is performed to enable a retrospective analysis to extract the root causes of the observed failures. This is essential in order to support timely recovery from problematic situations and/or safely operate the real asset in a degraded mode of operation.

In this paper, we present the proposed software architecture to build the ROBDT system and the corresponding functionalities, with particular attention to the interplay and synergies between engineering methods, symbolic and data-driven AI. E.g., in a planetary exploration mission, the terrain model used for planning is the same as the one used for simulation, and it is adapted with machine learning algorithms based on the telemetry data. Finally, we describe the preliminary achievement in the demonstrator based on the ExoMars planetary exploration mission.

The rest of the paper is organized as follows: Section 2 discusses the related work; Section 3 details the framework's components; Section 4 describes the rover demonstrator; and Section 5 draws the conclusions.

2 Related work

Analysed at its roots, NASA pioneered the concept of twins in the 1960s with Apollo project where two identical spacecraft were built, with the one on Earth called twin which reflected (or mirrored) the status of the spacecraft on a mission. Since then, *digital twin* (DT) concept has been applied extensively in the fields of manufacturing and robotics [2, 9, 10, 33, 35, 36, 43]. Many research works discussed the connotation, definition of digital twin concept, independently of the industry field [11, 16, 19, 22, 31, 34]. The term is not always used consistently and is explained in various ways from different perspectives. Considering the scope of this study, the following definition from [34] (adapted from [31]) is considered as the baseline:

"A Digital Twin is defined as a dynamic and self-evolving digital/virtual model or simulation of a physical asset (part, machine, process, human, etc.) representing its exact state at any given point of time enabled through real-time bi-directional data assimilation as well as keeping the historical data, for real-time prediction, monitoring, control and optimization of the asset to improve decision making throughout the life cycle of the asset."

Combination of data with models is at the heart of any DT. However, the specific types of models and integration of data strongly depend on the required services built on top of the DT.

With regards to online model adaptation, different methods have been proposed in literature to handle various uncertainties and partial observability (cfr., e.g., [15, 8, 23]). Digital twin frameworks based on supervised/unsupervised ML methods, such as dynamic Bayesian networks [13], particle filters [41], stacked denoising autoencoders (SDA) [24], etc., have been shown to enable continuous adaptation of physics based models for end-to-end uncertainty quantification, optimal decision-making, anomaly detection and the prediction of future conditions.

Hybrid modeling approaches have been widely used in scientific applications to embed the knowledge from simplified theories of physics-based models directly into an intermediate layer of the neural network (see for example [4]). Within this paradigm, physics-informed learning (PIL) [30, 39, 14] is based on regularization design for discriminative properties, while physics-augmented learning (PAL) [20] is based on model design for generative properties.

Reinforcement learning (RL) algorithms typically replace the traditional model-based planning and control processes. Recent scientific applications [37, 42, 32] have utilized the potential of RL for the inference of physics-based model parameters. They have been shown to: provide accurate real-time dynamical calibration, adapt to new scenarios, scale to large datasets and high-dimensional spaces, and to be robust to observation and model uncertainty. Within the context of DT, RL algorithms have been applied in the field of manufacturing [43], robotics [27, 32] and autonomous driving [42, 28], to provide services such as real-time model adaptation, anomaly detection, or what-if analysis.

In the context of planning, [26] proposes a production system control concept where a digital twin and an automated AI planner are tightly integrated together as one smart production planning and execution system. In the context of run-time monitoring, [25] proposes a formal specification framework to facilitate and automate specification extraction in natural language from documentation, as well as formalizing them for the digital twin. [21] proposes an approach for active monitoring of a neural network (NN) deployed in the real-world, which detects previously unseen inputs and creates interpretable queries for incremental NN adaptation to ensure safe behaviour.

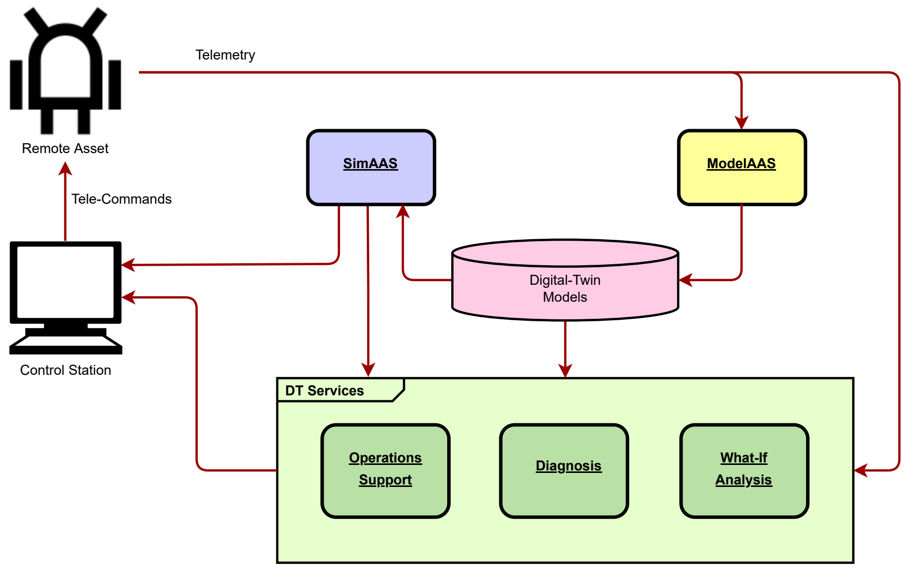

Fig. 1. The ROBDT architecture.

Overall, with respect to the above mentioned works, ROBDT presents some novel contributions. On one side, it is focused on space robotic systems, with the relevant peculiarities as for example the communication delays. On the other side, it provides a unique combination of symbolic automated reasoning, simulation, and machine learning techniques.

3 The ROBDT framework

3.1 System architecture

Figure 1 shows the ROBDT architecture as a first layer decomposition of the software into the following components:

Simulation As A Service (SimAAS): consists of multiple simulators which simulate the functionality of the robotic assets as well as the mechanisms for their upload to the system, their configuration and management (start, stop, delete), and finally the monitoring of their status. It involves the use of Kubernetes [18] and associated services.

Model As A Service (MAAS): provides ML Ops capabilities to the ROBDT by facilitating tasks such as data preparation, model training and model serving, while also enabling easy, repeatable, portable deployments on diverse infrastructure. It is based on the Kubeflow [17] open-source project. In the demonstrator, two such models and the corresponding pipelines are proposed: the wheel-terrain interaction model and the Data Handling Subsystem (DHS) update model.

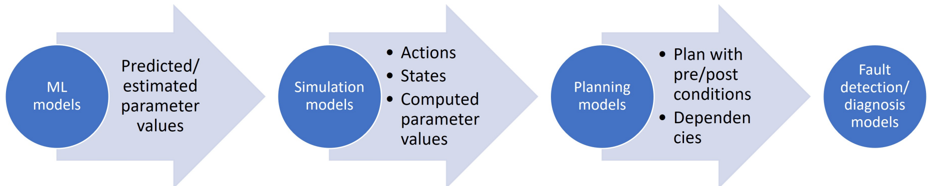

Fig. 2. Information exchanged between ROBDT models.

Digital Twin Models: the models are handled by a Digital Twin Manager (DTM), which manages DT definitions and facilitates operations associated with launching, monitoring and stopping the corresponding simulations by hiding the complexities of Kubernetes' APIs that are used in SimAAS to perform the same operations.

What-if Analysis (WIA): allows simulating the system from its current state or from a hypothetical state according to a given scenario with the additional possibility to check whether a certain goal condition is satisfied, or it is violated. In the context of this activity the WIA component focuses on the simulation of activity plans under various conditions with the additional capability of automatic activity plan generation,

Diagnosis (FDIRPM) : allows detecting faults in the current execution or on historic data, identifying causes of the faults based on their models, and providing the corresponding feedback to the operators. In the context of this activity, we propose to focus on the detection of faults during the execution of an activity plan and to propose a recovery action by generating an alternative one.

Operations Support (SCIDET): supports engineering or science operations planning and assessment. In this activity, a "scientific agent" is integrated to detect predefined patterns of interest or novelty on on-line or historical images acquired by the robotic asset.

These components use various heterogeneous models and there is a strong interplay between data-driven models, physics-based, and symbolic models, as summarized in Figure 2. In the following sections, we describe in more detail the components using these models.

3.2 Simulator

The simulation capabilities are provided by the instantiation of the SIMROB multi-asset space robotics simulator [12]. A high-level breakdown in models of SIMROB is depicted in Figure 3. It includes:

- The **Assets models** that allow to compute the state evolution of the subsystems of the assets that are under control. For each asset, it mainly includes models of its mechanical, electrical and thermal dynamics, of its data handling, communications and payload subsystems as well as the model of its control software. In particular:

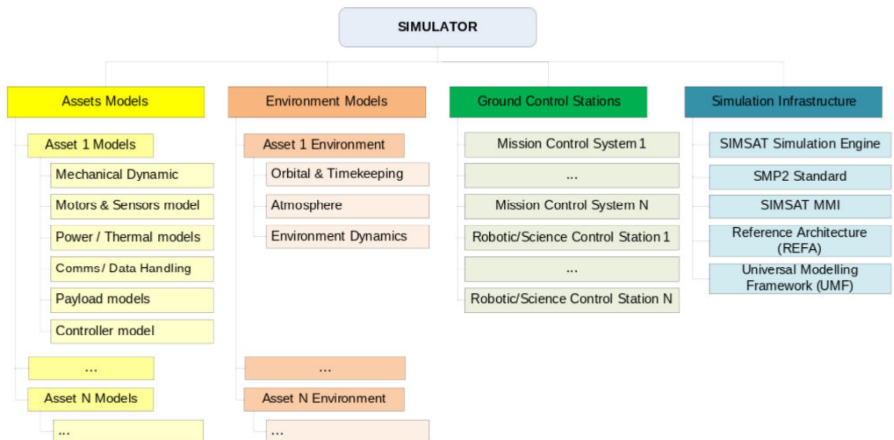

Fig. 3. The SIMROB simulator breakdown.

- The *Mechanical Dynamic Model* provides the state evolution of a multi-body actuated and sensorised mechanism possibly subject to external forces. Main elements of the models are the bodies (rigid or flexible) with their inertial parameters, the joints (actuated or passive) including fixed, prismatic, rotoid and universal joints, and finally their topology.
- The *Power Generation & Distribution Model* predicts the instantaneous energy flow in the power conditioning and distribution network in terms of current, voltages and losses at each relevant node, the state of charge and the behaviour of the battery, replicates the PCDE control logic and behaviour and finally simulates the power interfaces conditions (voltage, current) for all connected units.
- The *Controller Model* is a model of particular importance as it reproduces the role of the onboard flight software ranging from asset control to instruments control and mission management. It is structured in a three-layered architecture. At the lowest level, close to the actuators and sensors, the Functional layer implements the Actions representing the elementary Activities of the asset. The Executive layer implements the Tasks defined as a logical and temporal composition of Actions. Finally, the Mission layer handles mission planning and scheduling aspects executing Activity Plans either created on-ground or on-board.

– The **Environment models** are in charge of the simulation of the environments that interact with the assets. These models encompass the reproduction of planetary and orbiter ephemerides, the provision of planetary atmospheric data as well as the morphology and the topology of the environment in which the assets evolve. In particular:
  - The *Atmosphere Model* provides at any time and location of the asset the relevant atmospheric characteristics. For example, for Mars surface

operations, elements such as the air temperature, the surface temperature, the air specific heat capacity, the wind velocity, the dust optical depth, the atmospheric chemistry, as well as the radiation flux at ground level are important inputs for the simulation.
- The *Environment Dynamic Model* represents the topological, the morphological and the mechanical dynamics aspects of the environment in which the assets evolve. It concerns exclusively the assets which operate in close loop with their environment: a rover moving on a terrain, a robotic arm grasping an object, a robotic system using exteroceptive sensors (e.g. imagers) to perceive its environment. The outputs of this model are forces/torques applied at a given point, distances measured from a given point, images generated from a given point-of-view, etc. These measures are used by the mechanical dynamic model of the asset to evaluate its state after impact/contact, and by the sensor models (imagers, distance, force/torque sensors, etc) to compute their outputs.

– The **Simulation framework** is based on SIMULUS. It covers both the execution of a spacecraft simulation and the simulator architectural design. The SIMSAT Simulation Engine is the 'engine' of the simulator, the SIMSAT Man-Machine Interface provides the interface between the user and the simulator, the Generic Models (GENM) comprise a suite of reusable generic simulation models, the SMP2 standard enables reuse and portability of the simulation models, the Reference Architecture (REFA) establishes a suitable breakdown of simulators into models and finally the Universal Modelling Framework (UMF) supports an efficient and smooth approach of software development for SMP2 simulations.

3.3 Model updaters

In the ROBDT architecture, the Model As A Service (MAAS) is a component whose goal is to provide ML Ops capabilities to the system. The MAAS facilitates data preparation, model training, and model serving and enables easy, repeatable, portable deployments on diverse infrastructure. Serving the models from MAAS allows specific digital twin components to access the model predictions for different use such as simulation and diagnosis.

Infrastructure The MAAS is based on a specific component of the Kubeflow open-source project: Kubeflow Pipeline. Kubeflow is a service capable of making deployments of machine learning (ML) workflows on Kubernetes simple, portable and scalable. The Kubeflow Pipeline is a platform in Kubeflow that has a UI and an API for triggering and tracking experiments, jobs, and runs. It also contains the engine used to manage and execute the ML workflows created by the Model Training Pipeline steps. The Kubeflow Pipelines is a platform for building and deploying portable, scalable machine learning (ML) workflows based on Docker containers. The role of the ROBDT's private Docker Registry is to manage the Docker images into which the various steps of model training pipelines are packaged. Based on Seldon's Core, the Model Server exposes trained models as web APIs, providing its clients the ability to make predictions and get information on the models. The MAAS offers interaction with the Monitoring and Control Station (MCS) component to obtain the historical and real-time telemetries. Those data are requested from the model updaters and used to build the training dataset.

Model Updater Component Based on the functionality provided by Kubeflow it is possible to trigger the training of a model manually or automatically at regular intervals. Both can be performed and configured via Kubeflow's Pipeline UI or the corresponding API. Once the pipeline is triggered, Kubeflow Pipeline's orchestration engine starts executing the described workflow. The pipeline is made up of Docker images related to each other as a graph through input and output files dependencies. The pipeline steps are the following:

- *Telemetry acquisition*: the first step of the pipeline is responsible for the telemetry acquisition. The operation is done by contacting the MCS component by REST API, which provides the requested data in JSON format.
- *Data pre-processing*: the historical telemetries of the mission are pre-processed for the training and testing phase of the model.
- *Training*: the training dataset is obtained from the previous step and loaded using a custom DataLoader. The model is instantiated and trained; the model parameters are saved on a dedicated, persistent volume. As a result of this step, the best performing model is available for the next steps.
- *Test*: the model runs on the testing dataset to validate the performance. The score is then passed to the next step to provide information about the behavior of the updated model.
- *Deployment*: to enable inference service to the other ROBT components, this final step provides the configuration for the Seldon Core platform about the model parameters and the Python's handler script.

3.4 Planner and what-if analysis

One of the main problems when managing a remote asset is to plan the activities to be performed ahead-of-time, because a significant time delay can hinder direct tele-commanding. For example, it is not possible to tele-command an asset on the surface of Mars due to the communication delay that is in the order of tens of minutes. For this reason, the remote assets are equipped with autonomous executors of mission plans that need to be properly formulated on-ground ahead of time and uploaded before execution. Designing activity plans that are robust and achieve a desired objective is no trivial task when the complexity of the target system is more than trivial. For this reason, automated planning technologies have been historically employed in space applications. The usefulness of such technique is not limited to the generation of plans for the immediate future, but also to investigate hypothetical situations and to perform so-called "what-if analyses" (WIA): before committing to a specific plan or addressing a problematic situation, a planning and simulation system allows the study of different plans and objectives in different real or hypothetical situations. Finally, WIA allows for retrospective analyses: in light of new model information, one can re-assess past decision in order to improve future decision-making.

Within the ROBDT framework, WIA is seen as a service that takes advantage of the superior precision of digital twin models: thanks to the strong alignment between the models and the physical assets it is possible to provide more realistic estimations of the costs of a certain plan and ultimately to provide a better support for decision-making. Moreover, an interesting and pivotal feature offered by digital twins is the evolution of models, making it possible to adapt automated planning to the degradation of capabilities or to the evolving conditions of the environment.

On the practical side, the overall idea behind our digital-twin-enhanced WIA is to maintain a model-based approach for planning and high-level simulation, but to allow for parameters that are to be estimated/learned from the telemetry data within the digital twin. Concretely, we will study the behavior of automated planning when some parameters (in particular the duration of some activities) are estimated by means of ML models. Ideally, having a more precise timing model of the system will allow a less conservative planning which in turn will allow to fully exploit the remote asset capabilities.

3.5 Fault detection and diagnosis

Diagnosing faults is essential in order to detect and identify anomalies that could endanger the real asset. To this aim, the system is equipped with a Fault Detection, Isolation, and Recovery (FDIR) component that monitors the telemetry data and the execution of the activities, in their initial, in progress and terminating phase. A relevant aspect strictly related to the monitoring phase is the ability to perform its task even in absence of complete information on the state; for example, both the state of some components and the currently executed action may be unknown. In such a case, a set of belief states that are compatible with the observations, strictly contained in the set of all possible states, is considered and the diagnosis phase can be employed with this partial information. With the aim of addressing the partial observability problem we have used NuRV [5], [6], [7]; this tool is able to generate a monitor for a specific LTL property to be monitored on a given system. In this case the system consists of the plan, which can be seen as a loop-free algorithm, synthesized by the planner component, while the conditions to be monitored are pre, in progress and post conditions. In case an anomaly is detected, a retrospective analysis based on the DT models and the historical information on past states, is carried out to localize the possible faults and identify the root causes of the failure. Diagnosis is based on a fault model which describes the effects of faults, their dependencies and the fault propagation rules. The adaptation of the DT models at run-time can improve the situational awareness of the real asset and provide a more precise analysis with respect to the FDIR capabilities of the physical system alone. When an anomaly is detected, the FDIR component can trigger a reconfiguration of the system, e.g. to continue operation in a degraded mode. FDIR can also be used to aid predictive maintenance by supporting the detection of the performance degradation of some component. Finally, the FDIR component provides a service to the what-if analysis functionality, namely it supports the planning activities by monitoring the plan execution in order to detect and identify unexpected outcomes of the actions.

4 The ROBDT case study

For the demonstration of the ROBDT framework, we chose to work on a planetary robotic asset provided by TRASYS. Because of the inherent uncertainty of the robot-environment interaction, data-based models are well suited. Moreover, it is an ideal case study for path planning and monitoring. Let us consider a typical scenario prepared for a 'sol' execution from the ExoMars planetary exploration mission: the 'Drilling site approach and surface sample acquisition'. The following activities shall be performed autonomously under the constraints of the available power, memory capacity for data storage, and duration (single sol). Initially, the rover waits the transition Night to Day to wake-up (driven by the rover PCDE, when the solar panels generate power greater than a threshold (20W)) and configures accordingly the rover for the day activities. In particular, the subsystems involved for travelling are warmed-up and moved to a 'standby' state. These steps involve several uncertainties, mainly the exact local Mars time at which the rover wakes-up as well as the warm-up durations, which all depend on the external conditions (e.g., atmospheric temperatures, relative orientation of the rover solar panels with respect to the Sun, etc.).

At completion of the rover configuration, the rover starts traveling to reach the outcrop whose position has been identified from ground. Although the duration of the travel depends on the topology and characteristics of the encountered terrain, it can be *estimated* at planning time. At arrival at the outcrop, the rover is unconfigured from travelling operations and configured for drilling: travel related units are switched off while the drill box and the drill are warmed-up and moved to the 'standby' state. Again, the expected durations (and therefore power consumptions) have to be estimated as they depend on the time in the 'sol' that the rover reached the outcrop. At the next step the drill box is deployed, the drill is initialised, and reaches the soil to collect the surface sample (10cm depth). Afterwards, the drill retracts. The duration of the sampling procedure, and therefore the power consumption, depends on the hardness of the soil. Finally, images of the environment shall be acquired and downloaded to guarantee that the ground planning team has enough information for planning for the next sol. After establishing the communications with the orbiter and transferring the acquired data, the rover waits the transition Day to Night (driven by the rover PCDE, when the solar panels generate power less than the threshold of 20W), configures the Rover for night and 'sleeps' waiting for the next plan to be uploaded for execution.

In this scenario, the use case foresees:

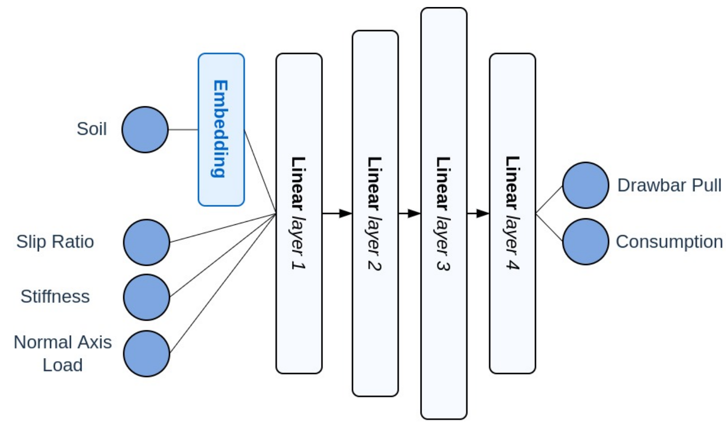

Fig. 4. The architecture of the Fully connected neural network for the WTI model.

– For the ROBDT system, there are two specific machine learning models that are adapted online: the wheel terrain interaction (WTI) model and the DHS model that predicts the Actuator Drive Electronics (ADE) warm-up time. Those elements run on top of the MAAS component taking advantage of the Kubeflow Pipeline architecture. In both cases, the approach used to fix the model architecture is the same. As an example, we present here the case of the WTI model. The purpose of the WTI model is to estimate the drawbar pull of the robotic asset and the power consumption of the motors. These calculations are made from the terrain and rover characteristics given as input from pre-processed telemetries. In particular, the inputs of the model are the terrain type (class), slip ratio (%), stiffness (%), and normal axis load (Newton). The outputs are the average drawbar pull (Newton) and the power consumption (Watt). Given the small number of input features of the model, the advantage of using Deep Learning solutions is not clear, as they usually require many parameters and are therefore heavy to train. For this reason, we are testing two different solutions to achieve the described functionality: the first is based on a Deep Learning Fully Connected Neural Network, and the second uses Boosted Decision Tree. While both can fit in the pipeline based on the MAAS, predictive power and computational resources required for the online adaptation can be different and must be carefully evaluated. The framework adopted for the development of the fully connected neural network is Pytorch. The neural network solution is composed of an initial embedding layer that maps the terrain class to a higher dimensional space. Then, this embedding is merged with the other features forming the input of the first layer of the network. A full representation of the network architecture is presented in Figure 4. For the boosted decision tree we used the Catboost Python library [29].
– For the planning part, we are setting up and experimenting with a fully-integrated solution, which is capable of planning with semi-opaque models.

In particular, we have access to a model of the activities and tasks of the rover and we need to automatically synthesize activity plans. However, some of the actions have simulated effects, meaning that the consequences of applying such actions are not modeled but can only be simulated, and the duration of some actions are also not formally modeled. Most notably, among these "evaluatable" quantities, we have the ADE warm-up timing that is estimated by a learned and evolving ML model as discussed above. We use the AIPlan4EU Unified Planning library [1] to model simulated effects and the TAMER planner[3] [38] for the actual plan generation. Preliminary results show that the approach is capable of generating valid plans quickly, and we are currently working on experimenting with the quantities estimated by means of ML.

- As for the fault detection part, we are monitoring the execution of the plan. The telemetry provides complete information about the state components, but no information about the state of the task execution. Looking at the case study, the plan provided by the planner requires first to run a task that waits for the rover to warm up and switch on a set of subsystems for traveling. Upon completion of this action, a state component is modified. This state component acts as a precondition for a task that is used for updating the rover heading estimate with a value provided by ground. If the monitor notices that the heating level has just been changed, but the precondition state component of the heating level update task has not been satisfied before, it can decide that the heating update task has been executed violating its preconditions in one of the possible belief states will report this violation to the operator and to the diagnostic component.
- Given the anomalies identified by the monitoring component, the goal of the diagnosis component is to provide a list of most probable explanations for these anomalies. The explanations are identified using a fault propagation graph (FPG), which describes how failures of one subsystem or component of the rover can cause failures of other components. In particular, for ROBDT, we construct the FPG as follows. First, for each task, we use the DT specification to identify the actions of other subsystems that can cause a failure of the given task. For example, the warm-up task that prepares the rover for travel depends on warming up the navigation cameras, localization cameras, actuator drive etc. Second, we use the description of hardware implementation and FMECA tables to describe how failures in the hardware components can cause failures of the higher level subsystems. For example, the actuator drive depends on working hold-down release mechanism, which in turn depends on working motors, motor heaters, etc. We then use efficient techniques rooted in formal methods [3] that for each set of identified failures list all the possible root causes.

  As a result, if the monitoring component reports an anomaly in the warm-up task, we can list failure of motors as one of the root causes (among many others). More interestingly, if the monitoring component reports sev-

[3] https://tamer.fbk.eu

eral anomalies, which all transitively depend on the motors, the diagnosis component can report the motor failure as the most probable root cause as it is more probable than multiple separate failures of independent subsystems.

5 Conclusions

In this paper, we presented the architecture of the Robotic Digital Twin, which is under development as part of the ROBDT study funded by ESA. We detailed the planning and monitoring functionalities and the related combination of data-driven physics-based, and symbolic models. Finally, we described the preliminary achievement in the demonstrator based on the ExoMars planetary exploration mission. In the remaining activities of the project, which will conclude within 2022, we will complete the prototype implementation and will evaluate it on the described scenario, thus identifying strengths and weaknesses of the approach. In the future, we are going to use the same infrastructure to validate and verify autonomous systems with AI/ML components with a simulation-based system level approach. This is part of VIVAS, another ESA-funded study started in May 2022 [40].

References

1. AIPlan4EU: The AIPlan4EU unified planning library. https://github.com/aiplan4eu/unified_planning
2. Booyse, W., Wilke, D.N., Heyns, S.: Deep digital twins for detection, diagnostics and prognostics. Mechanical Systems and Signal Processing **140**, 106612 (2020)
3. Bozzano, M., Cimatti, A., Pires, A.F., Griggio, A., Jonáš, M., Kimberly, G.: Efficient SMT-based analysis of failure propagation. In: Silva, A., Leino, K.R.M. (eds.) Computer Aided Verification - 33rd International Conference, CAV 2021, July 20-23, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12760, pp. 209–230. Springer (2021). https://doi.org/10.1007/978-3-030-81688-9_10
4. Chao, M.A., Kulkarni, C., Goebel, K., Fink, O.: Fusing physics-based and deep learning models for prognostics. Reliability Engineering & System Safety **217**, 107961 (2022)
5. Cimatti, A., Tian, C., Tonetta, S.: Assumption-based runtime verification with partial observability and resets. In: International Conference on Runtime Verification. pp. 165–184. Springer (2019)
6. Cimatti, A., Tian, C., Tonetta, S.: Nurv: a nuxmv extension for runtime verification. In: International Conference on Runtime Verification. pp. 382–392. Springer (2019)
7. Cimatti, A., Tian, C., Tonetta, S.: Assumption-based runtime verification of infinite-state systems. In: International Conference on Runtime Verification. pp. 207–227. Springer (2021)
8. Damianou, A., Lawrence, N.D.: Deep gaussian processes. In: Artificial intelligence and statistics. pp. 207–215. PMLR (2013)
9. Glaessgen, E., Stargel, D.: The digital twin paradigm for future nasa and us air force vehicles. In: 53rd AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference 20th AIAA/ASME/AHS adaptive structures conference 14th AIAA. p. 1818 (2012)

10. Hochhalter, J., Leser, W.P., Newman, J.A., Gupta, V.K., Yamakov, V., Cornell, S.R., Willard, S.A., Heber, G.: Coupling damage-sensing particles to the digitial twin concept. Tech. rep. (2014)
11. Jones, D., Snider, C., Nassehi, A., Yon, J., Hicks, B.: Characterising the digital twin: A systematic literature review. CIRP Journal of Manufacturing Science and Technology **29**, 36–52 (2020)
12. Kapellos, K.: Simulus based simulations of human-robotic operations. ESAW (2021)
13. Kapteyn, M.G., Pretorius, J.V., Willcox, K.E.: A probabilistic graphical model foundation for enabling predictive digital twins at scale. Nature Computational Science **1**(5), 337–347 (2021)
14. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nature Reviews Physics **3**(6), 422–440 (2021)
15. Kennedy, M.C., O'Hagan, A.: Bayesian calibration of computer models. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **63**(3), 425–464 (2001)
16. Kritzinger, W., Karner, M., Traar, G., Henjes, J., Sihn, W.: Digital twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine **51**(11), 1016–1022 (2018)
17. Kubeflow. https://www.kubeflow.org
18. Kubernetes. https://www.kubernetes.io
19. Liu, M., Fang, S., Dong, H., Xu, C.: Review of digital twin about concepts, technologies, and industrial applications. Journal of Manufacturing Systems **58**, 346–361 (2021)
20. Liu, Z., Chen, Y., Du, Y., Tegmark, M.: Physics-augmented learning: A new paradigm beyond physics-informed learning. arXiv:2109.13901 (2021)
21. Lukina, A., Schilling, C., Henzinger, T.A.: Into the Unknown: Active Monitoring of Neural Networks. In: Feng, L., Fisman, D. (eds.) Runtime Verification. pp. 42–61. Springer International Publishing, Cham (2021)
22. Madni, A.M., Madni, C.C., Lucero, S.D.: Leveraging digital twin technology in model-based systems engineering. Systems **7**(1), 7 (2019)
23. Marmin, S., Filippone, M.: Variational calibration of computer models. arXiv preprint arXiv:1810.12177 (2018)
24. Meraghni, S., Terrissa, L.S., Yue, M., Ma, J., Jemei, S., Zerhouni, N.: A data-driven digital-twin prognostics method for proton exchange membrane fuel cell remaining useful life prediction. Int. Journal of Hydrogen Energy **46**(2), 2555–2564 (2021)
25. Naumchev, A., Sadovykh, A., Ivanov, V.: VERCORS: Hardware and Software Complex for Intelligent Round-Trip Formalized Verification of Dependable Cyber-Physical Systems in a Digital Twin Environment (Position Paper). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **11771 LNCS**, 351–363 (oct 2019). https://doi.org/10.1007/978-3-030-29852-4_30, https://link.springer.com/chapter/10.1007/978-3-030-29852-4_30
26. Novák, P., Vyskočil, J., Wally, B.: The Digital Twin as a Core Component for Industry 4.0 Smart Production Planning. IFAC-PapersOnLine **53**(2), 10803–10809 (2020). https://doi.org/https://doi.org/10.1016/j.ifacol.2020.12.2865, https://www.sciencedirect.com/science/article/pii/S2405896320336314
27. Oh, M.h., Iyengar, G.: Sequential anomaly detection using inverse reinforcement learning. p. 1480–1490. KDD '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3292500.3330932, https://doi.org/10.1145/3292500.3330932

28. Pires, F., Ahmad, B., Moreira, A.P., Leitão, P.: Recommendation system using reinforcement learning for what-if simulation in digital twin. In: 2021 IEEE 19th International Conference on Industrial Informatics (INDIN). pp. 1–6 (2021). https://doi.org/10.1109/INDIN45523.2021.9557372
29. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: Catboost: unbiased boosting with categorical features. Advances in neural information processing systems **31** (2018)
30. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational physics **378**, 686–707 (2019)
31. Rasheed, A., San, O., Kvamsdal, T.: Digital twin: Values, challenges and enablers from a modeling perspective. Ieee Access **8**, 21980–22012 (2020)
32. Regatti, J.R., Deshmukh, A.A., Cheng, F., Jung, Y.H., Gupta, A., Dogan, U.: Offline rl with resource constrained online deployment (2021). https://doi.org/10.48550/ARXIV.2110.03165, https://arxiv.org/abs/2110.03165
33. Resman, M., Protner, J., Simic, M., Herakovic, N.: A five-step approach to planning data-driven digital twins for discrete manufacturing systems. Applied Sciences **11**(8), 3639 (2021)
34. Singh, M., Fuenmayor, E., Hinchy, E.P., Qiao, Y., Murray, N., Devine, D.: Digital twin: origin to future. Applied System Innovation **4**(2), 36 (2021)
35. Söderberg, R., Wärmefjord, K., Carlson, J.S., Lindkvist, L.: Toward a digital twin for real-time geometry assurance in individualized production. CIRP annals **66**(1), 137–140 (2017)
36. Sun, X., Bao, J., Li, J., Zhang, Y., Liu, S., Zhou, B.: A digital twin-driven approach for the assembly-commissioning of high precision products. Robotics and Computer-Integrated Manufacturing **61**, 101839 (2020)
37. Tian, Y., Chao, M.A., Kulkarni, C., Goebel, K., Fink, O.: Real-time model calibration with deep reinforcement learning. arXiv preprint arXiv:2006.04001 (2020)
38. Valentini, A., Micheli, A., Cimatti, A.: Temporal planning with intermediate conditions and effects. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020. pp. 9975–9982. AAAI Press (2020), https://ojs.aaai.org/index.php/AAAI/article/view/6553
39. Viana, F.A., Nascimento, R.G., Dourado, A., Yucesan, Y.A.: Estimating model inadequacy in ordinary differential equations with physics-informed neural networks. Computers & Structures **245**, 106458 (2021)
40. VIVAS. https://es.fbk.eu/index.php/projects/vivas/
41. Ward, R., Choudhary, R., Gregory, A., Jans-Singh, M., Girolami, M.: Continuous calibration of a digital twin: Comparison of particle filter and bayesian calibration approaches. Data-Centric Engineering **2** (2021)
42. Wu, J., Huang, Z., Hang, P., Huang, C., De Boer, N., Lv, C.: Digital twin-enabled reinforcement learning for end-to-end autonomous driving. In: 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI). pp. 62–65 (2021). https://doi.org/10.1109/DTPI52967.2021.9540179
43. Xia, K., Sacco, C., Kirkpatrick, M., Saidy, C., Nguyen, L., Kircaliali, A., Harik, R.: A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence. Journal of Manufacturing Systems **58**, 210–230 (2021)