TESTING AUTONOMOUS ROBOTS: A DISCUSSION ON PERFORMANCES OBTAINED DURING THE ERGO FIELD TESTS

Jorge Ocón [(1)], Francisco Colmenero [(1)], Iulia Dragomir [(1)], Enrique Heredia [(1)], Mercedes Alonso [(1)], Joaquin Estremera [(1)], Robert Marc [(2)], Piotr Weclewski [(2)], Thomas Keller [(3)], Mark Woods [(4)], Spyros Karachalios [(4)]

[(1)] GMV Aerospace and Defence, Isaac Newton 11, PTM, Tres Cantos, 28760, Spain, Email: jocon@gmv.com
[(2)] Airbus Defence and Space Ltd., Gunnels Wood Road, Stevenage, SG1 2AS, UK, Email: gnc.uk @airbus.com
[(3)] University of Basel, Spiegelgasse 1, Basel 4051, Switzerland, Email: tho.keller@unibas.ch
[(4)] Scisys UK Ltd, Methuen Park, Chippenham, SN14 0GB, UK, Email: mark.woods@scisys.co.uk

ABSTRACT

The European Robotic Goal-Oriented Autonomous Controller (ERGO) (http://www.h2020-ergo.eu) is one of the PERASPERA SRC first call projects. The focus of this project is to develop a framework for long-range autonomy that allows commanding a spacecraft via high-level goals. The developed ERGO framework achieves this aim by providing a new paradigm based on components and tools. Two main components, that include Artificial Intelligence technology, are at the core of ERGO: an *on-board mission planner*, able to dynamically generate plans on-board from high-level goals, and a *scientific agent*, that detects targets of interest from images. Additionally, other autonomous capabilities that can be used and tailored for different robotics platforms, such as *rover navigation* and *robotic arm motion planning*, are provided by ERGO. In this paper we discuss the results and performances achieved in the field tests of two applications developed with the ERGO framework: an orbital and a planetary mission.

1 INTRODUCTION

The first call of the PERASPERA SRC [1] aimed to develop the basic blocks for bringing intelligence and autonomy to space applications in general, and robotics in particular. Within this effort, the ERGO project is focused on extending such applications with autonomy, for which it has developed the ERGO framework [2] [3] [4].

The ERGO framework is a set of components and tools for the development of highly autonomous systems. More specifically, the framework allows one to command a spacecraft via high-level goals that are dynamically transformed on-board into plans. The framework also enables opportunistic science (i.e., looking for targets of interest), while the system is performing other activities such as traverses or pick/drop operations. If a target of interest is found, the system is able to dynamically modify the plan in order to gather science.

Moreover, the framework implements a rigorous model-based development approach that enables the following properties: modularity, reuse of components, compatibility (with other frameworks), and formally checked reliability and resilience. For this ERGO uses the TASTE toolset [5], a model-driven architecture tool developed by ESA, in combination with the formal verification and validation techniques provided by the BIP tools [6] [7] [8].

The ERGO framework and approach are generic enough to be tailored to any space robotics systems for different environments such as orbital, deep space probe, and planetary exploration missions, but also to future robotics terrestrial applications demanding a high level of autonomy.

The capabilities of this framework have been tested within the ERGO project in two scenarios: an orbital scenario simulating an in-orbit servicing mission, and a planetary scenario, inspired from the Mars Sample Return (MSR), simulating long traverses and pick up/drop operations.

In this paper, we present the validation scenarios of the ERGO framework and the results obtained during the field tests. We discuss the problems found during these tests, the performance of the system, and possible further improvements. We conclude with the lessons learned and a set of goals that future autonomous space robotics missions should achieve.

2 OVERVIEW OF THE ERGO FRAMEWORK

The ERGO framework consists of different reusable modules as showed in Fig. 1, each providing new capabilities for the developed system. The core tools provide generic features, such as telecommunication, mission planning and runtime enforcement. The specific components target precise features of space robotics applications, such as rover guidance and robotic arm motion planning, and can be configured based on the needs of the developed system. Finally, platform-

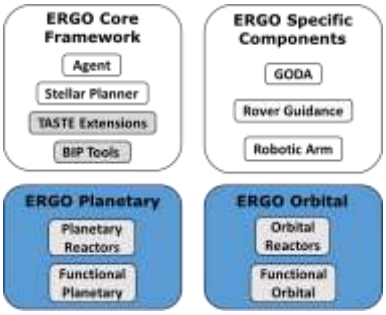

Figure 1. The ERGO Framework packages.

dependent components (ERGO Planetary and Orbital) enable tailoring the ERGO framework to the chosen robotics platform, e.g., the SherpaTT rover [9] for a planetary scenario and UR5 robotic arm [10] running in Platform-Art [11] for an orbital scenario. It is worth mentioning that the architecture of the ERGO framework allows tackling each problem independently, and therefore uses state-of-the-art algorithms for each of the proposed components.

2.1 Design Approach

The use of ERGO framework is based on a model-driven development process supported by the TASTE toolset. TASTE is the middleware chosen by the ESROCOS framework [12], another project of the PERASPERA first call aimed to the development of a robotics operating system. The decision to use TASTE in ERGO paves the way for a complete interoperability of both frameworks in future applications.

The key concept of this approach is the system design, which consists of multiple components communicating via message exchanges/signals, possibly deployed on different architectures.

A developer will start by designing the architecture of an application in a platform-independent component-based fashion. Components can be user-defined ones or those existing from the ERGO framework, therefore allowing for a maximal reuse. For each user-defined component, the developer will model how it will interact with the others in terms of strongly typed interfaces (in ASN.1, a language for describing structured data types, that is independent of the programming language), and will connect them by means of connectors. Also, the functionality of the component is modelled/coded using different languages (e.g., SDL, C/C++). Platform-dependent details, such as deployment hardware elements, are modelled next. Finally, executable code is generated from the system design and with respect to the targeted platform. TASTE provides support for different platforms (Linux 32 bit, Linux 64-bit, SPARC), which is automatically handled in the code generation phase ensuring the proper delivery and reception of messages between components.

The design approach used by ERGO sets up good practices of system development: separation of concerns, modularity, compatibility and interoperability of components and frameworks, reduced dependencies, and reuse of already developed components. Moreover, the ERGO design approach is completed with formal verification and validation techniques based on the BIP tools, which ensure increased reliability and resilience at runtime with synthesised Fault Detection, Isolation and Recovery (FDIR) components [7] [8].

2.2 ERGO Agent

In ERGO, the intelligence of the system is embedded into the so-called ERGO agent. The aim of the ERGO Agent is manifold: (1) ensure the communication between the ground and the spacecraft, (2) generate the plans that are to be executed, and, eventually, modify the on-board plan dynamically when the conditions require it, (3) scan images for target detection, (4) ensure the consistent execution of on-board activities, and (5) control the platform in a sense/act approach. The agent follows the Teleo-reactor Executive (T-REX) paradigm [13], in which components are embedded into the so-called *reactors (components of the agent)*, each of them providing a common interface for an agent controller [4]. In our architecture, the following reactors are available, and they are explained hereafter:

- Ground control interface
- Mission planner
- Scientific detector
- Command dispatcher

Please note that other reactors can be easily added to the framework/application by means of extending the reactor classes available.

2.2.1 Ground Control Interface Reactor

This component handles the telecommands and telemetry that are received/sent from/to ground, and *the level of autonomy*. This *level of autonomy* follows the ECSS standard definition [14] that is:

- Level 1 (*E1*): only direct telecommanding is allowed.
- Level 2 (*E2*): time-tagged telecommanding is allowed.
- Level 3 (*E3*): adaptive (event-driven) telecommanding is allowed. On-board procedures can be executed based on events. This level enables the execution of an uplinked plan.
- Level 4 (*E4*): goal-commanding is allowed. This highest level of autonomy is the *main*

objective of ERGO. In this level both AI components in ERGO (Stellar, the mission planner, and GODA, the scientific detector) are activated.

The rationale for handling multiple levels of autonomy is to cope with the needs of different on-board situations. For instance, planning shall be disabled during maintenance operations, or under critical circumstances.

2.2.2 Mission Planner (Stellar)

ERGO's mission planner, Stellar, is a heuristic search temporal planner based on Fast Downward [15]. It has an interface for external functions, handles temporal constraints similar to OPTIC [16], but is tailored to minimize memory consumption by not using state annotations. Fast Downward was extended to reason with snap-actions, where the start- and end-point of each PDDL durative action are considered to be separate instantaneous actions. We used a technique from OPTIC called compression safety to reduce the size of the search space. It recognises where it is completeness-preserving to apply the end of an action as soon as it has started.

The search algorithm is guided by a simple heuristic based on the successful FF heuristic [17]. We relax the task for the heuristic computation by ignoring its numerical aspects and by compiling the temporal aspects into two classical actions. We also use preferred operators, a technique that prefers actions from the relaxed plan in the search and had a strong impact on performance in the classical setting.

We furthermore build upon the temporal constraint management approach of OPTIC, making a distinction between logical and temporal consistency. To circumvent the usage of memory inefficient annotations to each state, we recreate these annotations by inspecting the preconditions and effects of the snap-actions by iterating backwards through the plan. Therefore, only the constraints themselves are recorded in each state.

Finally, to allow the usage of sub-solvers (e.g. for path or motion planning) in the planning process, we follow the approach from [18] that allow the definition of external solvers as modules. When loading the PDDL file Stellar looks for a dynamic library with the name of the module and imports all functions from it. The dynamic library can be implemented independently of the planning algorithm and may make calls to other reactors.

2.2.3 Scientific Detector (GODA)

Bandwidth or communication limitations make real-time control of instruments for scientific discovery difficult or impossible. For planetary rovers there is a trade-off between detailed observation to ensure targets are not missed, which requires slow traverses to downlink all data, and maintaining sufficient progress to visit many science targets. The ability for a robotic system to interpret the data captured by its sensors, assess its situation in the environment and then perhaps alter its plan to account for this new knowledge about the world is a crucial component for a system with E4 level of autonomy.

For ERGO, SCISYS built the Goal Oriented Data Analysis (GODA) component based on the systems developed in the successful ESA project MASTER – Mobile Autonomous Scientist for Terrestrial and Extra-terrestrial Research [19]. MASTER focused on the problems of detecting (and defining) novelty, and the need for expert scientist input to be used to train the detector. As well as integrating with the rest of the ERGO system, improvements and new components were added to GODA.

Since MASTER was primarily concerned with detecting novelties, the workflow has been changed to accommodate for the GODA output which is primarily targets and goals of scientific interest. Combining the output of GODA with the inputs from the scientists for mission goals can greatly reduce the amount of time that is spent by scientists searching in images for features of interest. Moreover, GODA gives the ability to the scientists to process offline a large number of images and to get an automated report on the findings.

The GODA component pairs a MASTER-like detector with a goal generation component. The goal generation component maps detections of phenomena of interest into concrete goals for the planner to achieve. For example, detection of novelty could trigger a goal to acquire high resolution imagery, or by detecting known phenomena it could potentially work in tandem with mission planning to provide cost estimates to deviate trajectories to capture serendipitous science.

2.2.4 Command Dispatcher Reactors

The so-called command dispatchers are the interface of the agent with the functional layer. The T-REX paradigm models the system as a set of variables that change their value during the execution, i.e., *timelines* [13]. The command dispatcher reactors handle the values of the timelines used to interface with the functional layer of the robotic platform.

2.3 ERGO Specific Components

As mentioned above, ERGO includes a set of components that can be used in specific robotics

applications in order to provide a higher level of autonomy. These components, rover guidance and robotic arm, are described next.

2.3.1 Rover Guidance

Rover Guidance (RG) is aimed to provide navigation analysis, path planning, trajectory control, resources estimation and hazard prevention (HP) and avoidance for a planetary rover, as illustrated in Fig. 2.

RG must ensure the rover navigates only in safe areas of the terrain. These decisions are based on orbital and local data provided by external interfaces in the form of 2.5D digital elevation maps (DEMs). RG is responsible to close a mobility control loop by commanding directly the locomotion system with generic Ackerman commands based internal autonomy algorithms.

The long-distance goal location is provided by ground control or the ERGO mission planner, while the estimated rover position/attitude is provided by an external sensor data fusion system. RG is perception agnostic, so this data can be obtained, for example, by fusing inputs from LiDAR (or stereo-camera), IMU and HiRISE orbital images.

RG implements novel navigation architecture: using dynamically reconfigurable multi-mode autonomy together with a Hazard Prevention (HP) module, checking for path safety. The system is capable to adapt the amount of planning depending on the traversed terrain difficulty, performing increased local planning for more challenging terrains.

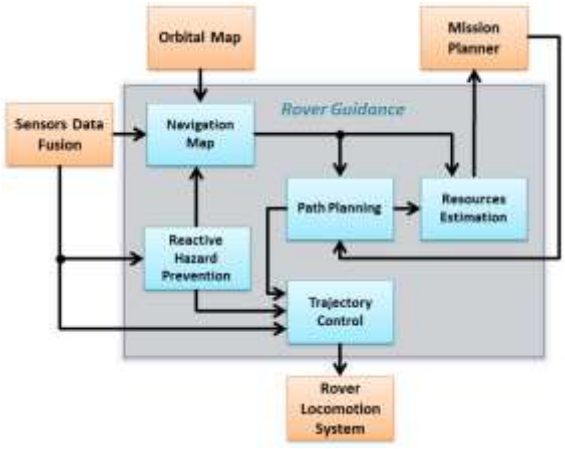

Figure 2. Architectural overview of the RG building blocks (blue) including external systems (orange).

2.3.2 Robotic Arm

The functionality of the robotic arm component is twofold. On one hand, given a movement to be performed by the robotic arm in order to pick/move/drop an object or a single move operation, it identifies and returns the corresponding set of low-level commands to perform the movement, together with the timing related to the operation, as well as the energy required. This is what we call the *motion planning* capability.

On the other hand, the robotic arm component is also responsible of the *execution of the movement* of the robotic arm, when it is needed, and the *control* of the robotic arm during its movement. For this, the component provides primitives to perform atomic operations, such as picking an object, dropping it or moving the arm to a given position (e.g., home).

Additionally, the robotic arm component includes functionalities for correcting the arm position based on images obtained from a camera, updating its knowledge of the environment also with the use of the camera and ensuring the mechanical safety of the robotic arm with different measurements.

3 THE ORBITAL SCENARIO

The reference mission for the orbital track is the on-orbit servicing mission (Fig. 3), where a damaged spacecraft can have one of its modules replaced autonomously by a servicer spacecraft.

3.1 Scenario Description

In this scenario, a servicing spacecraft (*chaser*) first approaches a faulty or serviced spacecraft (*target*). Following the iBoss concept [20], the target consists of a set of building blocks or *advanced payload modules* (APMs) that can be exchanged or replaced at will. Then the chaser will perform the required operation: (1) replace in orbit some faulty/damaged APMs, thus repairing the target, or (2) reconfigure the target´s APMs based on the defined needs.

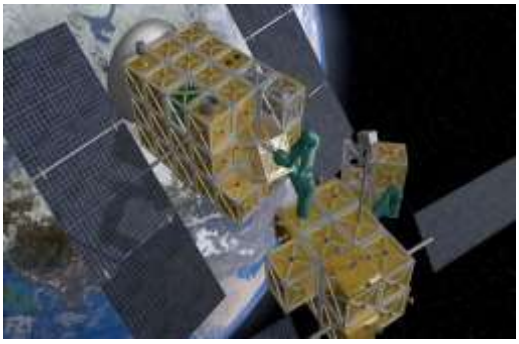

Figure 3. Orbital scenario: a chaser approaches a target for spacecraft repair/reconfiguration.

The robotic platform used in this scenario is the GMV´s Platform-arm dynamic test bench. The environment is adapted to the scenario with one Kuka robot holding a tray and simulating the movement of the chaser. On top of the tray, an UR5 robotic arm is used to perform the

servicing. The target is simulated by a vertical platform holding the APMs, as illustrated in Fig. 4.

Figure 4. Orbital scenario field test set-up.

3.2 Evaluation Criteria

The aim of the orbital scenario is to evaluate the autonomy performances. Besides the nominal execution(s) in all E1-E4 autonomy modes, the architecture and test environment must also allow demonstrating reactivity to runtime modifications. Two different sources of modifications are considered here:

- *Failures*: such as pieces or tools not present in the expected place or found in a different attitude, obstacles in the visual field, failure in grasping pieces, excessive torque, relative deviation between chaser and target, etc.
- *Deviations with respect to the nominal mission*, such as reconfiguration of the spacecraft due to mission constraints (deadlines exceeded, for instance).

In both cases, re-planning needs to be performed based on updated information from the environment. For that purpose, feedback information is obtained by passive visual means (camera) and from the robot end-effector (force/torque of the robotic arm).

For the sake of clarity, the aim of this scenario is to validate the mission planner and robotic arm components. The scientific detector and rover guidance are not part of the orbital application, since they are not needed.

3.3 Results

The testing approach consisted of two phases: a preliminary evaluation in order to detect and correct possible errors beforehand, and the demonstration field test.

The first phase tackled, in an exhaustive manner, basic functionalities of the scenario, such as nominal missions in E1-E4 autonomy modes, E4 missions requiring re-planning and downgrade of autonomy due to multiple reasons. During this phase, several errors were identified and corrected in the mission planner reactor and robotic arm component. In some cases, the planner took too long to find a valid plan, which led to an improved internal representation for the mission planner in the field tests.

The demonstration field tests covered more complex executions of the basic functionalities tested before. At this point, all tests met the set criteria and the following achievements are reported:

- Successful execution of E1-E4 telecommands.
- Successful re-planning and goal execution in E4 (e.g., faulty APM).
- Successful downgrade of autonomy when conditions require it (e.g., no feasible plan, dangerous situations demanding instructions/ reconfiguration from ground).
- Successful validation of the FDIR components that ensure the robotic arm safety in dangerous situations (e.g., excessive torque, relative deviation between chaser and target).

Additionally, several software metrics have been evaluated during the field tests. The performances achieved are given in Table 1. In this scenario, the physical parameters of the platform used are bounded by its software. The speed of the robotic arm was set at 0.5rad/sec, while the acceleration at 0.18rad/sec^2.

Table 1. Performances of the orbital application in the field tests.

Metric	Description	Value
RT	Total running time	45 min
NG	Number of goals executed in E1-E4 autonomy modes	3
NPDO	Number of executed pick, drop and home operations (ops)	12 pick ops 12 drop ops 4 home ops
TPS	Total planning time	2min

4 THE PLANETARY SCENARIO

The reference mission for the planetary track is inspired by the Mars Sample Return (MSR) mission that covers the concepts and requirements of the Martian Long Range Autonomous Scientist.

4.1 Scenario Description

This scenario consists of a planetary exploration rover able to pick samples with a robotic arm, as well as to take images of scientific interest. The scenario allows the following functionalities:

1. *Setup multi-sol operations* (Ground Control): Ground Control configures the robot and uploads the operations, which might be single or multi-sol. Operations received from Ground Control will be any of the following types.

2. *Traverse*: The rover must perform a long range traverse, in the range of 1km, to a specified position. A command to take an image with a given heading, once the final position has been reached by the rover, can be implicit in the traverse operation.
3. *Opportunistic science*: During the long traverse, the rover is allowed to perform opportunistic science. Based on the images taken from the SherpaTT's camera, the scientific agent may detect targets of interest. When this occurs, new plans are dynamically generated to analyse the newly detected target.
4. *Sample collection*: the rover can be requested by ground to pick or drop samples at different locations by using its robotic arm.

Figure 5. DFKI's SherpaTT rover being tested in Moroccan desert.

The robotic platform used in the planetary scenario is the SherpaTT rover from DFKI. SherpaTT is a 4-wheeled planetary exploration rover with an actuated suspension system developed for high mobility in irregular terrain. The rover is able to use energy efficient wheeled locomotion (in contrast to legged locomotion) to cover long distances, and at the same time to negotiate difficult terrain by dynamically adapting the wheel suspension to slopes and obstacles. Fig. 5 shows the SherpaTT rover in the Moroccan desert during the test fields.

4.2 Evaluation Criteria

The aim of the planetary scenario is to evaluate the autonomy performances of the overall ERGO framework in an application built with all the developed components.

Five main tests were conducted for the planetary scenario:

- *E1/E2/E3 missions* checked the viability of commanding the rover in low levels of autonomy.
- *E4 nominal missions* checked that from a set of goals sent to the rover, an on-board plan is generated and nominally executed.
- *E4 non-nominal missions* checked that the mission planner re-plans due to abnormal circumstances during the execution (e.g., exceeded deadlines, low battery, etc.).
- *E4 missions with opportunistic science and re-planning* checked that, while a nominal plan is being executed, the scientific agent detects a new target and injects a new goal into the system. The plan is modified accordingly and executed.
- *Long traverse missions* checked the rover guidance capabilities. The rover is capable of traversing more than 1km autonomously through harsh terrain, when a long traverse on-board procedure is uplinked.

4.3 Results

The same validation approach as in the orbital scenario has been followed for the planetary scenario. The first phase, conducted in Bremen at DFKI facilities, targeted the validation of the components and their integration in the system. At this phase many issues were found and corrected. These errors were related to the integration of components in the TASTE design, integration of Mission Planner with GODA, tuning of the Rover Guidance and Robotic Arm for the SherpaTT, and integration of ERGO with the SherpaTT. At the end of the preliminary tests, the following had been achieved:

- Successful execution of E1/E2/E3 telecommands.
- Rover Guidance was able to perform traverses of up to 40m in natural terrain.
- Robotic Arm was able to perform pick and drop operations.
- Successful image acquisition with the on-board camera, used for the configuration of GODA to detect targets of interest.
- Successful validation of the FDIR components that ensure the rover safety in failure conditions.

The demonstration field tests were performed at Erfoud, in the Moroccan desert, that provides a representative analogue setting for Martian environment. During this phase the tests briefly described in Sec. 4.2 were performed. Several issues have been corrected during the tests, mostly related to integration, further fine tuning of the components parameters, and the TASTE middleware. The main achievements of the field tests are the following:

- The ERGO Agent worked continuously and reliably in all autonomy levels (E1 to E4) for hours.
- GODA was able to detect serendipitous events in the environment.
- The Mission Planner was able to continuously plan and alter the plans either due to abnormal circumstances or in order to investigate the target of interest detected by GODA.
- The Rover Guidance was able to traverse more than 1km expected in the long traverse test: 1.3km in a single sol.

Besides the scenario success criteria, several metrics have been evaluated during the field tests. These parameters cover the overall tests performances, and performances of specific components such as Mission Planner, GODA and Rover Guidance. The results obtained are given in Table 2.

Table 2. Performances of the planetary application in the field tests.

Metric	Description	Value
RT	Total running time	9h39min
CD	Estimated distance covered in autonomous mode	1396m
MS	Number of goals accomplished in E4	10 goals
PMS	Ratio of accomplished/ total goals in E4	41%
APT	Average planning time	9.07sec
SDPT	Variance of the planning time	9.65sec
NIG	Number of images analysed by GODA	285 images
GGFDR	Ratio false positive goals/total number of goals generated by GODA. (Parameterized by *P*, minimum classification probability set by experts)	11.7% ($P \geq 0.8$)
		27.2% ($P \geq 0.6$)
ATSR	Ratio average navigation speed / maximum rover speed	0.32m/sec
AP	Average speed for approaching targets	0.048m/sec

While the results achieved validate and improve, to the best of our knowledge, the state of the art for autonomous rover planetary explorations, further enhancements can be considered for the ERGO framework. The main observations made during the field tests are:

- The Mission Planner generates sub-optimal plans.
- The current policy of discarding goals when there is no planning solution is not optimal.
- GODA generates a rather large number of false positives that are given as goals to Mission Planner. In consequence, the rover could be blocked for large amounts of time due to continuous re-planning.
- While Rover Guidance was able to perform a 1.4km autonomous traverse, this could be further improved by parallelizing computations and therefore reducing the number of stops.

All these enhancements are currently under study in the Autonomous Decision Making in very long traverses (ADE) project of the PERASPERA second call.

5 CONCLUSION

The ERGO framework has been developed to fulfil a set of characteristics required for autonomy in future space robotics applications. Its components and tools have been tested in two different use cases, and will be used in the PERASPERA second call for developing (other) applications. The main features are:

1. *Goal-commanding* and *selectable levels of autonomy* from the Ground Control, implemented in the ERGO agent.
2. *On-board planning*, by the Stellar planner. The mission planner additionally allows dynamic re-planning triggered by new goals received from ground, new goals received from the opportunistic science, and invalid constraints of the current plan.
3. *Opportunistic science* by GODA. Any detected target of interest enables the re-planning of the current goals by Stellar such that it can be further inspected and investigated.
4. *Autonomous guidance* for rovers that performs path planning, hazard avoidance and trajectory control.
5. *Motion planning and robotic arm control* for gripping and grasping objects of interest.
6. *Rigorous model-based design approach* for developing applications based on TASTE toolchain for system design and BIP tools for offline and online formal verification and validation.

The capabilities of the ERGO framework make it suitable for future use in applications in which autonomy will be required. These applications also cover terrestrial needs for autonomy in domains as nuclear, oil & gas, rescue, etc.

As mentioned above, the ERGO framework is currently being enhanced in the ADE project of the PERASPERA second call. While the performances obtained in the field tests by the framework are high-grade, we believe

that it can be further extended and optimised. Several optimisations have already been identified during the field tests with respect to several components and are under study. Other enhancements in terms of capabilities (e.g., combined motion of rover and robotic arm) are being added to the framework. Finally, we plan to increase the TRL level of this framework, such that it will be a good candidate for developing future robotics applications in many domains.

6 REFERENCES

1. Plan European Roadmap and Activities for Space Exploitation of Robotics and Autonomy (PERASPERA). Online at http://www.h2020-peraspera.eu
2. Ocón, J., Delfa, J.M., Medina, A., Lachat, D., Marc, R., Woods, M., Wallace, I., Coles, A.I., Coles, A.J., Long, D., Keller, T., Helmert, M., Bensalem, S. (2017). ERGO: A Framework for the Development of Autonomous Robots. In Proc. 15th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), European Space Agency, Noordwijk, The Netherlands.
3. Ocón, J., Buckley, K., Colmenero, F.J., Bensalem, S., Dragomir, I., Karachalios, S., Woods, M., Pommerening, F., Keller, T. (2018). Using the ERGO Framework for Space Robotics in a Planetary and an Orbital Scenario. In Proc. 14th Symposium in Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), European Space Agency, Noordwijk, The Netherlands.
4. Ocón, J., Colmenero, F.J., Estremera, J., Buckley, K., Alonso, M., Heredia, E., Garcia, J., Coles, A.I., Coles, A.J., Martinez, M., Savas, E., Pommerening, F., Keller, T., Karachalios, S., Woods, M., Dragomir, I., Bensalem, S., Dissaux, P., Schach, A., Marc, R., Weclewski, P. (2018). The ERGO Framework and its Use in Planetary/Orbital Scenarios. In Proc. 69th International Astronautical Congress (IAC), IAF, Bremen, Germany.
5. Perrotin, M., Conquet, E., Delange, J., Tsiodras, T. (2012). TASTE – An open-source tool-chain for embedded system and software development. In Proc. Embedded Real Time Software and Systems Conference (ERTSS), SAE, Toulouse, France.
6. Basu, A., Bozga, M., Sifakis, J. (2006). Modeling heterogeneous real-time components in BIP. In Proc. 4th Int. Conf. on Software Engineering and Formal Methods (SEFM), IEEE.
7. Dragomir, I., Iosti, S., Bozga, M., Bensalem, S. (2018). In Proc. 8th Int. Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA), Springer.
8. Dragomir, I., Bensalem, S. (2019). Rigorous Design of FDIR Systems with BIP. In Proc. 1st Interactive Workshop on the Industrial Application of Verification and Testing (InterAVT), EPTCS. (To appear)
9. Cordes, F., Babu, A. (2016). SherpaTT: A versatile hybrid wheeled-leg rover. In Proc. 13th Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), European Space Agency, Noordwijck, The Netherlands.
10. https://www.universal-robots.com/products/ur5-robot/
11. https://www.gmv.com/en/Products/platform/
12. Muñoz, M., Montano, G., Wirkus, M., Hoeflinger, K., Silveira, D., Tsiogkas, N., Hugues, J., Bruyninckx, H., Dragomir, I., Muhammad, A. (2017). ESROCOS: A Robotic Operating System for Space and Terrestrial Applications. In Proc. 15th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), European Space Agency, Noordwijk, The Netherlands.
13. McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., McEwen, R. (2007). T-REX: A model-based architecture for AUV control. In 3rd Workshop on Planning and Plan Execution for Real-World Systems.
14. ECSS Secretariat. (2005). *ECSS-E-70-11 Space Segment Operability*. European Space Agency, Noordwijk, The Netherlands.
15. Helmert, M. (2006). The Fast Downward Planning System. In Journal of Artificial Intelligence Research, vol. 26, pp. 191-246, AI Access Foundation.
16. Benton, J., Coles, A., Coles, A. (2012). Temporal Planning with Preferences and Time-dependent Continuous Costs. In Proc. 22nd Int. Conf. on Automated Planning and Scheduling (ICAPS), AAAI.
17. Hoffmann, J., Nebel, B. (2001). The FF Planning System: Fast Plan Generation through Heuristic Search. In Journal of Artificial Intelligence Research, vol. 14, pp 253-202, AI Access Foundation.
18. Dornhege, C., Eyerich, P., Keller, T., Trüg, S., Brenner, M., Nebel, B. (2009). Semantic attachments for domain-independent planning systems. In Proc. 19th Int. Conf. on Automated Planning and Scheduling (ICAPS), AAAI.
19. Wallace, I., Woods, M. (2015). MASTER: A Mobile Autonomous Scientist for Terrestrial and Extra-Terrestrial Research. In Proc. 13th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), European Space Agency, Noordwijk, The Netherlands.
20. Kreisel, J. (2017). iBOSS: intelligent Building Blocks for On-Orbit Satellite Servicing and Assembly. In European Robotics Forum.

ACKNOWLEDGMENTS

We would like to thank the European Commission and the members of the PERASPERA programme support activity (ESA as coordinator, ASI, CDTI, CNES, DLR, and UKSA) for their support and guidance in the ERGO activity. We also thank the ERGO Consortium for their collaboration and support during the field tests. Finally, we are grateful to Malte Wirkus and Thomas Vögele from DFKI for their support in the integration of the ERGO software on the SherpaTT rover.

This project has received funding from the European Union´s HORIZON 2020 research and innovation programme under grant agreement No 730086.