# Underwater Demonstrator for Autonomous In-Orbit Assembly of Large Structures

Virtual Conference 19–23 October 2020

Christian Ernst Siegfried Koch[1], Marko Jankovic[1], Sankaranarayanan Natarajan[1], Shubham Vyas[1], Wiebke Brinkmann[1], Vincent Bissonnette[2], Thierry Germa[2], Alessio Turetta[3], Frank Kirchner[1,4]

[1]*German Research Center for Artificial Intelligence, Robert-Hooke-Straße 1, 28203 Bremen, Germany, E-mail: christian.koch@dfki.de*
[2]*Magellium, 1 rue Ariane, 31750 Ramonville St Agne, France, E-mail: vincent.bissonnette@magellium.fr*
[3]*Graal Tech S.r.k., Via Tagliolini 26, 16152 Genova, Italy, E-mail: alessio.turetta@graaltech.it*
[4]*University of Bremen, Robotics Lab, Robert-Hooke-Straße 1, 28203 Bremen, Germany, E-mail: frank.kirchner@dfki.de*

## ABSTRACT

The PULSAR project aims to develop key-technologies to enable the autonomous assembly of large structures in space. Similar to industrial applications, the assembly process relies on robotic systems capable of assembling modular elements to form a complex structure. However, the in-space assembly provides exceptional challenges necessitating innovation in fields such as free-floating manipulation and autonomous robotics. This paper provides details on the PULSAR project and, more specifically, on a hardware-in-the-loop demonstrator dLSAFFE, developed to show the assembly process of a large telescope mirror in a micro-gravity environment.

## 1 INTRODUCTION

Autonomous assembly of large structures in space is a key challenge for future missions that will necessitate structures too large to be self-deployed as a single piece. The James Webb Space Telescope [1] has reached this limit with a mirror of 6.5 meter diameter. Concepts for the next generation telescope LUVOIR [2], expected by astronomers in the late 2030s, currently rely on the development of novel launch vehicles with larger payload fairing, e.g. NASA's Space Launch System. On-orbit assembly technologies will deliver from these restraints: Large structures can be assembled in-situ from modular parts, allowing for smaller and less expensive launch systems. On-orbit assembly, however, poses many challenges in fields such as assembly automation and autonomous robotics.

The EU funded project PULSAR (Prototype of an Ultra-Large Structure Assembly Robot) aims to develop and demonstrate technology for an on-orbit precise modular assembly of very large structures by an autonomous robotic system. In this context, PULSAR focuses on the assembly of the primary mirror of a space telescope with a diameter of 8 meters. Key challenges, e.g. high precision and control of the free-floating manipulation are addressed in software simulations and hardware-in-the-loop (HIL) demonstrators. One HIL demonstrator is dLSAFFE, the demonstrator of Large Structure Assembly in a Free-Floating Environment. In the context of PULSAR, dLSAFFE aims to demonstrate the assembly at near 1:1 scale with representative hardware and software.

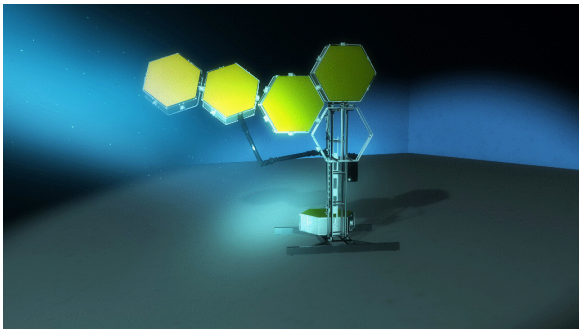

*Figure 1: Visualization of dLSAFFE in the large test basin at DFKI Bremen*

The need for large structures in space goes beyond telescopes and also concerns solar arrays for power plants, light sails to reach outermost regions of the solar system, or heat shields to land on Mars.

The field of space robotics was identified as key to increase Europe's competitiveness in the space sector. Consequently, the European Commission deployed a Strategic Research Cluster (SRC) on "Space Robotics Technologies" within the Horizon 2020 research program [3]. The SRC is divided into three stages, each comprising of several projects, or Operation Grants (OGs). In the first stage, common building blocks for space robotics were developed, three of which are ESROCOS [4], I3DS [5], and SIROM.[6] European Space Robotics Control and Operating System (ESROCOS) is an open-source robot control operating system (RCOS) targeted towards space robotics. It was based around the ASSERT Set of Tools for Engineering (TASTE) toolchain developed by the European Space Agency (ESA) [7]. I3DS is a suit comprising perception sensors selected for space robotics and data processing hardware. SIROM

is an electro-mechanical-thermal interface for coupling modular payloads. The PULSAR project belongs to the second stage that aims to further develop and validate the common building blocks. For example, a standard interface based on the SIROM design is used for coupling the modular mirror tiles, and the visual perception hardware is selected from the I3DS sensor suit. Furthermore, the software architecture is developed and implemented using the ESROCOS framework.

## 2 PULSAR SCENARIO

The scenario in PULSAR is the assembly of the primary mirror of a space telescope. The mirror has a diameter of 8 meters and consists of 36 hexagonal mirror tiles (SMT – segmented mirror tiles) arranged in three rings around a central tile. The assembly process utilizes an autonomous robotic system, the Robotic Assembly System (RAS). The RAS will extract individual SMTs from a storage unit and assemble them at the pre-defined position. The SMTs are coupled with neighboring SMTs via the electro-mechanical and thermal Standard Interface (SI). The same interface is used as an end effector (EE) by the RAS to manipulate the SMTs.

The RAS consists of a 6 degree-of-freedom (DoF) articulated robot manipulator and a rail system acting as prismatic joint at the base of the manipulator. The rail system extends the manipulator workspace and allows it to move between the storage unit and the assembly area. However, the reach of the RAS is limited and the outermost ring of SMTs in the mirror assembly is unreachable. One key-concept in PULSAR is Extended Mobility, allowing the assembly of structures larger than the workspace of the robot. The limitation is overcome by implementing a secondary assembly site, where a set of 5 SMTs is pre-assembled. This Pre-assembly is then subsequently manipulated as a single unit as visualized in Fig. 2.

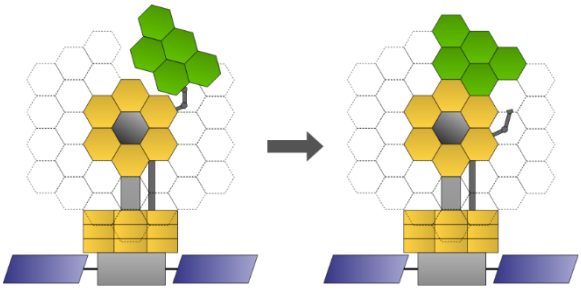

*Figure 2: Manipulating and assembling the pre-assembly (green) to build the outer rings*

The assembly process consists of several repeating steps: (1) Mating of two SIs. Two cases can be considered: the RAS EE mates with a target SMT, e.g. to extract it from the storage, or an SMT is assembled at a desired location. (2) Un-mating of two SIs. Again, two cases can be considered: the RAS releases a previously grasped SMT, or SMTs are released from the spacecraft (S/C) structure, e.g. the storage. (3) Manipulation of SMTs. The RAS manipulates one or multiple SMTs towards a target pose. The different steps are commanded sequentially according to a pre-defined assembly plan. In each step, the configuration of the system, including location of the SMTs, is known. However, to account for uncertainties, the mating process is guided by visual feedback: Two cameras, one at the RAS EE and one external camera observing the mirror assembly, are used for visual pose estimation based on the detection of fiducial markers.

The assembly process is demonstrated within the PULSAR project by three demonstrators, covering different aspects of the scenario. The demonstrator presented in this paper, dLSAFFE, aims to demonstrate the assembly process at near 1:1 scale with a realistic workspace representation. Emphasis is put on the Extended Mobility functionality. Three demonstration scenarios with up to five SMTs are considered: (1) assembly of the inner ring, (2) assembly of the pre-assembly and (3) assembly of the outer ring, i.e. the concept of Extended Mobility. For the final demonstration, a subset of the pre-assembly is considered. The three scenarios are shown in Fig. 3.

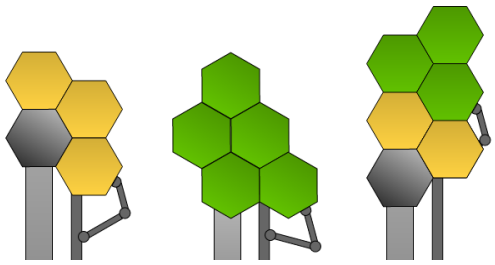

*Figure 3: Final stages of the demonstration scenarios: assembly of inner ring (left), pre-assembly (middle), assembly of outer ring (right)*

The demonstration is set in a zero-buoyancy environment, the large water basin at DFKI Bremen. Underwater experiments in the context of space robotics, allows to simulate micro-gravity by carefully balancing weight and buoyancy, components or missions can be put into a floating state, so that in-orbit-like conditions can be simulated. For dLSAFFE, RAS and SMTs are designed to be neutrally buoyant. With this set-up, the assembly and manipulation of large structures with a representative lightweight robot can be performed.

The large basin at DFKI Bremen is used as testing facility. The basin covers an area of 23 meters length

by 19 meters width with a depth of 8 meters. The facility is equipped with a 12 tons gantry crane allowing to handle large experiment set-ups [8].

Setting large scale experiments in the field of space robotics in an underwater environment poses several challenges. For example, the submerged equipment must be adapted and simplified for underwater usage. Further, the large scale of the experiments poses logistic challenges that need to be addressed within the project. One adaptation regards the SIs and affects the mating process. To reduce complexity, the SIs on the SMTs rely on permanent magnets and do not require electricity for mating. The mating between SMTs is thus fully passive. On the S/C structure and the RAS EE, actively controllable electromagnetic interfaces are used. An admittance control scheme is used to exploit the magnetic forces experienced during the mating process. The forces, detected by the RAS force/torque sensor, are felt as a pull towards the mating position. Further, the force reading is used to detect the successful mating.

## 3 DLSAFFE DEMONSTRATOR

The aim of dLSAFFE is to demonstrate the assembly process at near 1:1 scale with each SMT measuring about 1.3 meters in diameter. During the demonstrations, one section of the mirror is assembled, stretching approximately 4 meters across. dLSAFFE is designed to accommodate such a large experiment and the demonstration is placed in the large water basin at DFKI. During the demonstration, a total height of 6 meters will be reached.

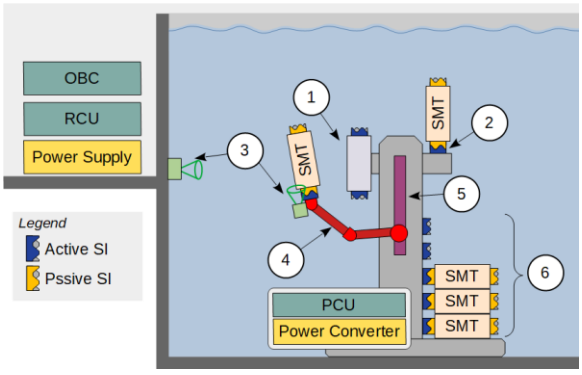


*Figure 4: dLSAFFE Hardware Architecture. (1) Final Assembly Site, (2) Pre-Assembly Site, (3) cameras, (4) RAS arm, (5) RAS linear rail, (6) SMT* storage

However, the mock-up is equipped with the RAS, a 6 DoF robotic manipulator sitting on a 1 DoF linear rail. For dLSAFFE, simplified SMTs are designed, characterized by a low drag and neutral buoyancy. Each SMT measures 1.3 meters in diameter. The demonstration scenario considers one section of the mirror consisting of five modules, stretching approximately 4 meters across, reaching a total height of 6 meters.

### 3.1 Hardware Architecture

Fig. 4 shows the hardware architecture of dLSAFFE. As can be seen, only a subset of components needs to be in the water while, for example, computational units can remain on land. The submergible part of the demonstrator features the S/C mock-up measuring approximately 3 meters in height with a 5 square meter footprint. The structure of the S/C mock-up is made of aluminum strut profiles by Bosch Rexroth. This allows for short-term adaptations if needed. A platform serves as footing and ensures the stability of the structure on ground. A tower is mounted on the platform serving as a supporting element for the RAS, the pre-assembly site (PAS), the final assembly site (FAS) and the SMT storage.

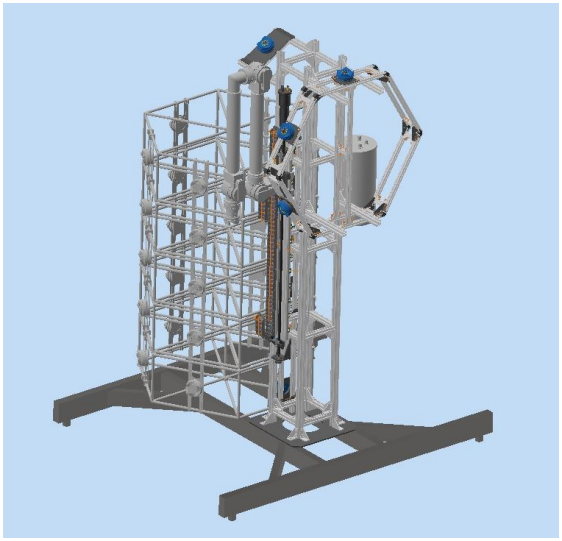

*Figure 5: CAD visualization of dLSAFFE. In the back, the SMTs in their storage location can be seen.*

The FAS is the central tile of the mirror assembly. It is designed as a hexagonal prism in the same dimensions as the SMTs. Three sides of the FAS feature active (controllable) SIs to which the SMTs are assembled. The FAS is mounted on the front side of the tower whereas the SMT storage and the PAS are located on the back of the tower. The SMT storage provides five active SIs to secure the SMTs in their initial configuration. The PAS provides an additional SI to support the pre-assembly process. The RAS is mounted to one side of the tower. It consists of a 6 DoF manipulator arm and a linear rail. The manipulator arm can be moved by the linear rail in order to reach all the SMTs in their storage as well as the

target assembly locations on the FAS and PAS. The manipulator arm is equipped with a camera and a six-axis force/torque sensor. The CAD model of the dLSAFFE S/C mock-up is presented in Fig. 5.

The demonstrator is controlled by three computational units: The On-board-computer (OBC), the Platform Control Unit (PCU), and a sensor data acquisition unit from the I3DS project referred to as RCU. The OBC serves as the main computational unit and is located on land. The RCU, also located on land, interfaces the cameras. The PCU, on the other hand, is located on the submergible platform and interfaces the active SI, actuators, and remaining sensors. For communication and power supply, the underwater demonstrator is connected to land via an umbilical, i.e. power- and fiber-optical data cables.

### 3.2 Robotic Assembly System

The RAS consists of two components as shown in-Fig. 6: a 6 DoF manipulator arm and a 1 DoF linear rail. The system is mounted vertically to the side of the S/C tower structure.

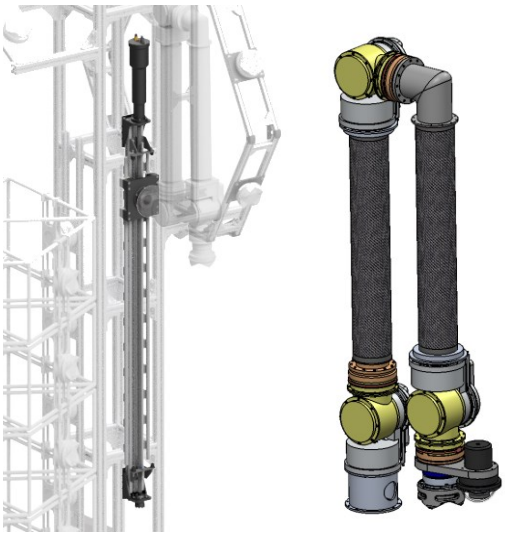

*Figure 6: Robotic Assembly System (RAS): rail (left) and 6 DoF manipulator (right)*

The manipulator has been made by exploiting the available design of the Underwater Modular Arm (UMA) commercialized by Graal Tech. It is characterized by three identical joint modules, with two motion axes each, for a total of 6 DoF, providing a sufficient dexterity during the various manipulation phases. The joints are made of anodized aluminum, while the connecting bodies are made of carbon fiber tubes filled with air. One of them is a cylinder, while the other one is L-shaped (with a 90 ° bend) for minimizing the arm size when stowed. The overall length when totally stretched is slightly more than 2 meters. The weight in air is around 20 kilograms, while it has been designed to result neutrally buoyant in water. The rail acts as a prismatic joint allowing to vertically move the base of the manipulator along a distance of 1.3 meters. The rail is implemented as a spindle drive and carriage system, tailored to the requirements of underwater usage. A commercial off-the-shelf geared motor inside a custom waterproof housing is used as actuator. It is designed to support the manipulator arm both in water as well as in air and emits self-braking behavior when not powered.

### 3.3 Mirror Tiles and Standard Interfaces

Each SMT is shaped as a hexagonal prism, as shown in Fig. 7. The SMTs consist of a skeleton made of carbon fiber tubes glued together with 3D printed components providing a proper sealing for ensuring water tightness. The SIs are mounted on each lateral side of the tiles to H-shaped carbon fiber plates clamped to the skeleton through 4 anodized aluminum fixing elements. Every SI is attached to its plate with screws allowing an easy removal and replacement, if necessary. The SMTs are equipped with fiducial markers for vision passed pose estimation. To aid the assembly process in the presence of uncertainties, the markers are used to detect the precise location of the SI and the entire SMT.

The SMTs have been designed to be neutrally buoyant in water. Moreover, its symmetric design guarantees an even mass distribution, which is necessary to avoid hydrostatic moments in water, while the open framed structure (without planar faces) significantly reduces the hydrodynamic drag during motions in water.

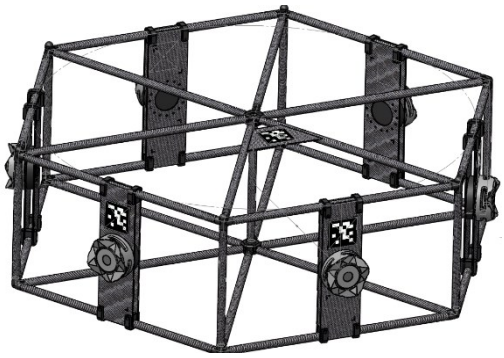

*Figure 7: Segmented Mirror Tile (SMT) showing SI and AprilTag [9] placement*

Every SI is characterized by a 90 ° symmetry, with the two mating parts that need to be have a relative angle of 45 ° to properly engaged into each other. The connections are established using magnetic coupling, as it allows a simple enough design compatible with the underwater environment. To allow the disas-

sembly of engaged interfaces three different kinds of SIs have been designed: 2 passive and 1 active. In one of the passive configurations there is a disc of Neodymium N45 permanent magnet, coated with a nickel layer as a protection against corrosion. In the other passive configuration, the magnet is substituted by an iron cylinder, mounted in the same position. In the active SI, around the iron cylinder, an electrical coil is introduced. This way, whenever required, an electric current within the coil originates an electromagnetic force with the opposite polarity than those acting between magnet and iron. For this reason, the net intensity of the adhesive force between the two surfaces results significantly reduced and the disassembly can be operated by a simple motion of the arm. The design of the SI is shown Fig. 8.

*Figure 8: Standard Interfaces (SI) for coupling SMTs*

### 3.4 Software Architecture

Fig. 9 visualizes the simplified software architecture of dLSAFFE. The architecture is divided in three layers, Sequencing, Functional and Hardware. The Sequencing layer comprises high-level control components. The central component is the Sequencer. It monitors and orchestrates the assembly process. The Ground Control Interface provides feedback and intervention options for the human operator. The Functional layer comprises perception and mid-level control functions, i.e. the RAS motion planning and control components. The Mating Detector uses the force/torque sensor to detect the successful mating of SIs. The Hardware layer comprises all the drivers of the actuators and sensors. They are interfaced by components from both other layers.

While some functionality, e.g. drivers, are specific to dLSAFFE, several functions are shared as Core Components among the three demonstrators of PULSAR. For example the RAS Motion Planner and Controller and the Perception Functions are considered Core Components and thus designed with generality in mind and not tailored to the specifics of each demonstrator.

The software stack is deployed on a distributed system among three different computational units: OBC, PCU and RCU. The OBC runs the majority of higher-level components, while the PCU acts as an interface to the individual components and runs drivers and the RAS Controller. The RCU interfaces the cameras and runs preprocessing functions.

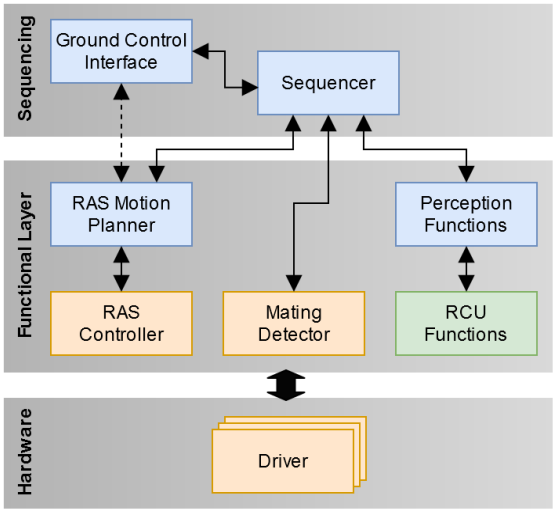


*Figure 9: dLSAFFE Software Architecture. The colors of the components denote the deployment: OBC (blue), PCU (yellow), RCU (green)*

The implementation of dLSAFFE software architecture, especially on a representative hardware, is bound to require a significant software engineering effort which might be mitigated by using one of the popular RCOS, such as the Robot Operating System (ROS) [10], the Open Robot Control Software (Orocos) [11] or the Robot Construction Kit (Rock) [12], just to name a few. However, none of these frameworks have been developed with critical applications in mind, and therefore lack the Reliability, Availability, Maintainability and Safety (RAMS) characteristics required for space applications. For this reason, adapting the existing frameworks to comply with the mentioned requirements is deemed impractical. Finally, the existing solutions currently used in the space industry are usually tied to a specific robotic system and are mostly closed-source [4].

To bridge this gap, in recent years the ESROCOS RCOS [4] has been developed as an open-source framework specifically for space robotics and with RAMS requirements in mind. It builds upon the TASTE toolchain [7]. and provides a set of tools and software components to support the development and deployment of robotics applications with demanding RAMS requirements [4].

The PULSAR project relies on the heritage of ESROCOS by developing and implementing the dLSAFFE software architecture (see Fig. 9) within the mentioned framework, as outlined in what follows.

### 3.5 RAS Motion Planner and Controller

The objective of the RAS Motion Planner component is to generate trajectories in joint space for the RAS to accomplish desired manipulation tasks. The generated trajectories should fulfill constraints such as collision avoidance, joint limits, shortest path, etc., based on the desired input from the Ground Control Interface and/or the Sequencer. The component uses a Motion planner framework [13] to generate an optimal collision-free path in a cluttered environment. The framework provides an interface to different state-of-the-art motion planners [14]–[16], kinematic [17], [18] and collision libraries [19], [20]. Moreover, the framework is designed using a Factory method pattern design [21], which provides a user with a possibility to easily experiment with different combinations of motion planner, kinematic and collision libraries to solve complex planning problems by simply changing a configuration file. Finally, the framework is platform-independent and has been already used on real systems within the Rock framework [12]. An example implementation of the RAS Motion Planner component within the functional layer of ESROCOS is displayed in Fig. 10, illustrating the logical interaction between various components of the example RAS. Each visible component is modelled as a separate TASTE function using C++ programming  The logical dependencies between the components are defined using a set of provided and required interfaces which are to be specified by a user via a GUI and are handled subsequently by TASTE without any user input.

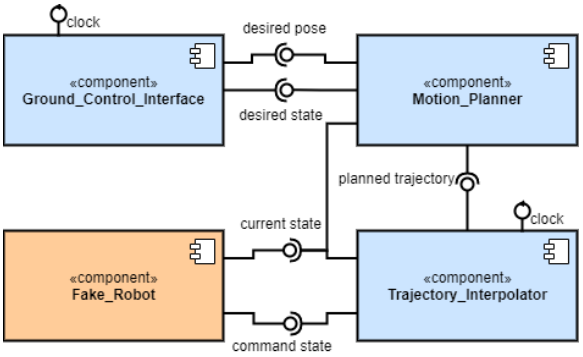


*Figure 10. RAS Motion Planner example implementation in ESROCOS/TASTE*

A brief description of the implemented functions and their expected purpose are outlined as follows:

- *Ground_Control_Interface* queries the user for input every 20 seconds and provides the motion planner with goal pose or joint states.
- *Motion_Planner* plans an optimal, collision free motion of the manipulator based on the user input and current state of the robot.
- *Trajectory_Interpolator* interpolates the planed trajectory (whenever available) and supplies the robot with the joint commands.
- *Fake_Robot* provides the pervious functions with the current state of the joints and uses the command from the trajectory interpolator to update its current state.

The objective of the RAS Controller component is to generate joint positions and velocities to be sent to the manipulator and rail drivers to achieve the desired motion despite disturbances that will affect the execution of the planned motion. Therefore, while the expected input data are the current and desired positions and velocities of manipulator joints in time, the expected output data are positions and velocities of manipulator joints in time to be sent to the drivers of the rail and manipulator joints.

Due to the very nature of the envisioned mating process and the magnetic properties of the interfaces to be used in the dLSAFFE, physical interaction between a robot manipulator and the environment is to be expected and needs to be properly accounted for within the control scheme of the manipulator during the mating process of two interfaces. However, the same control scheme needs to be able to perform the planned motion despite the hydrodynamic effect that will occur during the motion of the RAS and tiles through the water. To account for these requirements the current implementation of the dLSAFFE control architecture is envisioned to be composed out of two separate components: the Feedback Control and Admittance Control, which are activated by the user depending on the task to be completed.

The purpose of the Feedback Control is to use the measured joint positions and velocities to achieve the desired trajectories calculated by the motion planner despite the external disturbances or modelling errors. Therefore, it is envisioned to date to be a PID controller that derives the necessary joint trajectories of the RAS rail and manipulator to achieve the planned motion.

The function of the Admittance Control is instead to use the measured joint positions and velocities, along with the measured force/torque applied on the end-effector to derive the appropriate joint trajectories to achieve a compliant behaviour of the robot during the mating process of interfaces.

The implementation of the RAS Controller within the ESROCOS framework is still an ongoing process and is therefore not presented in this paper.

### 3.6 Perception Functions

The perception functions process images captured on the demonstrator cameras to increase the autonomy level and the safety of the assembly process.

Input images are first triggered and recovered from the cameras by the perception component using the Integrated 3D Sensors (I3DS) framework [5]. The software framework of I3DS is a set of C++ libraries for creating sensor interfaces (using their drivers) in a standardized way. The interfaces use ASN.1 messages and transport sensor data through a ZeroMQ communication layer.

Sensor acquisition is based on a subscriber/publisher pattern to handle frame acquisition at a given frequency, and a client/server pattern for command request. The EE and external cameras are each connected to a server and a publisher whereas perception functions define subscribers and callback functions to manage frame reception.

In dLSAFFE, two primary perception functions are used: Tile Localization and Assembly Monitoring. Using the AprilTag [9] fiducial markers mounted on the sides and back of the segmented mirror tiles, as shown on Fig. 7, Tile Localization will detect the tiles, based on their unique IDs, in images captured by the External or EE-mounted cameras. Tag corners coordinates are then extracted, and the pose of the tile with respect to the camera frame is computed using Perspective-n-Point methods. The pose of the tile is finally computed using the 2D/3D correspondences, between 2D coordinates extracted from AprilTags detection and 3D coordinates of the tile model. This localization is used to enable a safe, step-by-step approach to grasp the tiles with the manipulator end-effector, instead of relying only on open-loop localization based on manipulator forward kinematics.

Further emphasizing safety, the Assembly Monitoring function enables automatic detection of invalid conditions during the assembly. By comparing the currently localized tiles in the observed scene with an internally held ground truth model of each assembly step, it raises an error whenever an out-of-bounds tile of subassembly pose is detected. The error can then be handled by the autonomy component to perform a user-defined recovery or wait for operator intervention.

## 4 CONCLUSION AND FUTURE WORK

This paper provides an overview about the PULSAR project and, more specifically, presents the underwater demonstrator dLSAFFE. With dLSAFFE, the process of on-orbit modular assembly can be demonstrated in a zero-buoyancy environment, mimicking zero-gravity conditions.

Currently, the dLSAFFE demonstrator is being assembled at DFKI. Parallelly, tests are being carried out for the use of admittance control for mating of the magnetic interfaces. The results from these tests will lead to the development of the admittance control strategy as well as the development of algorithms for detection of successful mating between 2 interfaces. After the assembly of the demonstrator, it will be tested under water in the large water basin at DFKI. These tests will be used to characterize the influence of hydrodynamic forces on the system and thus, develop methods to exert the required restoring force to carry out the tile assembly as required. Following this, the complete assembly of a large segment of the mirror will be carried out. Subsequently. as the controller for the robot manipulator system is being developed in ESROCOS, an evaluation of ESROCOS will be carried out to validate its applicability in such simulated scenarios. Having a functioning system also provides a testbed for performing the tile assembly with a free-floating base in the future.

**Acknowledgement**

The PULSAR project is funded under the European Commission's Horizon 2020 Space Strategic Research Clusters Operational Grants, grant number 821858.

**References**

[1] J. P. Gardner, "About Webb Space telescope," *James Webb Space Telescope*, Sep. 18, 2020. http://tinyurl.com/y3col462 (accessed Sep. 18, 2020).

[2] The LUVOIR Team, "The LUVOIR Mission Concept Study Interim Report," *ArXiv180909668 Astro-Ph*, Sep. 2018, Accessed: Sep. 18, 2020. [Online]. Available: http://arxiv.org/abs/1809.09668.

[3] W. Brinkmann *et al.*, "Space robotic systems and artificial intelligence in the context of the European space technology roadmap," in *Proceedings of Space Tech Conferences 2019*, Bremen, Germany, Nov. 2019, p. 12, Accessed: Sep. 14, 2020. [Online]. Available: http://tinyurl.com/yxf99a9k.

[4] M. M. Arancón *et al.*, "ESROCOS: Development and Validation of a Space Robotics Framework," in *Proceedings of the 15th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2019)*, Noordwijk, The Netherlands, 2019, p. 8, Accessed: Sep. 09, 2020. [Online]. Available: http://tinyurl.com/yynta4zr.

[5] V. Dubanche and S. Andiappane, "Development of I3DS: An Integrated Sensors Suite for Orbital Rendezvous and Planetary Exploration," in *Proceedings of 'The 14th International Symposium on Artificial Intelligence, Robotics and Automation in Space - iSAIRAS'*, Madrid, Spain, Jun. 2018, p. 8, Accessed: Sep. 15, 2020. [Online]. Available: http://tinyurl.com/y6ovyu75.

[6] J. Vinals *et al.*, "Future Space Missions with Reconfigurable Modular Payload Modules and Standard Interface – An Overview of the SIROM Project," in *In Proceedings of the 69th International Astronautical Congress (IAC-2018)*, Bremen, Germany, 2018, p. 11, Accessed: Oct. 11, 2019. [Online]. Available: http://tinyurl.com/y6xdryg9.

[7] A. Perrotin, E. Conquet, P. Dissaux, T. Tsiodras, and J. Hugues, "The TASTE Toolset: turning human designed heterogeneous systems into computer built homogeneous software," presented at the ERTS2 2010, Embedded Real Time Software & Systems, Toulouse, France, May 2010, Accessed: Dec. 13, 2019. [Online]. Available: http://tinyurl.com/y3v4zkaq.

[8] DFKI, "Maritime Exploration Hall," *Robotics Innovation Center - DFKI GmbH*, Sep. 18, 2020. http://tinyurl.com/v5qmlgt (accessed Sep. 18, 2020).

[9] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 4193–4198, doi: 10/ghbm3h.

[10] M. Quigley *et al.*, "ROS: an open-source robot operating system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009, vol. 3, p. 5, Accessed: Sep. 16, 2020. [Online]. Available: http://tinyurl.com/yysd3nhj.

[11] H. Bruyninckx, "Open robot control software: the OROCOS project," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, May 2001, vol. 3, pp. 2523–2528 vol.3, doi: 10/fc8w2w.

[12] DFKI, "the Robot Construction Kit (Rock)," 2019. https://www.rock-robotics.org/ (accessed Sep. 10, 2020).

[13] B. Asadi and S. Natarajan, "Motion Planning for Manipulators," presented at the RIC Project Day "Framework & Standardization" and "Manipulation & Control," Bremen, Germany, Jun. 19, 2014, Accessed: Sep. 16, 2020. [Online]. Available: http://tinyurl.com/y63e9h7g.

[14] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robot. Autom. Mag.*, vol. 19, no. 4, pp. 72–82, Dec. 2012, doi: 10/gdnbz5.

[15] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 4569–4574, doi: 10/fxbq67.

[16] J. Schulman *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1251–1270, Aug. 2014, doi: 10/f6j8k3.

[17] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov. 2015, pp. 928–935, doi: 10/ghbtq4.

[18] R. Smits and E. Aertbelien, "Kinematics and Dynamics Library (KDL)," *Orocos Kinematics and Dynamics*, 2020. https://orocos.org/kdl (accessed Sep. 10, 2020).

[19] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 3859–3866, doi: 10/ggs7v9.

[20] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning," 2019 2016. http://pybullet.org (accessed Sep. 14, 2020).

[21] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, *Design patterns: Elements of reusable object-oriented software*. Reading, MA: Addison-Wesley, 1995.