DATA FUSION FRAMEWORK FOR PLANETARY AND ORBITAL ROBOTICS APPLICATIONS

Shashank Govindaraj(1), Jeremi Gancet(1), Irene Sanz Nieto(1), Xavier Martinez-Gonzalez(1), Iyas Dalati(1), Raul Dominguez(2), Alexander Fabisch(2), Mark Post(3), Romain Michalec(4), Alessandro Bianco(3), Vincent Bissonnette(4), Simon Lacroix(5), Andrea de Maio(5), Michal Smicek(6), Nassir Oumer(6)

(1) Space Applications Services, Leuvensesteenweg 325, 1932 Sint-Stevens-Woluwe, Belgium, shashank.govindaraj@spaceapplications.com
(2) DFKI, Robert-Hooke-Straße 1, 28359 Bremen, Germany, raul.dominguez@dfki.de
(3) University of Strathclyde, Richmond Street 16, G1 1XQ, Glasgow, United Kingdom, mark.post@strath.ac.uk
(4)LAAS-CNRS, 7 Avenue du colonel roche, Toulouse, France, simon.lacroix@laas.fr
(5) Magellium SAS, rue Hermes, 24, Ramonville St Agne Cedex, 31521, France, vincent.bissonnette@magellium.fr
(6) DLR, Münchener Straße 20, Weßling, 82234, Germany, michal.smisek@dlr.de

ABSTRACT

In space robotics, a wide range of sensor data fusion methods are required to accomplish challenging objectives for exploration, science and commercial purposes. This includes navigation for planetary and guidance for orbital robotics, scientific prospecting, and on-orbit servicing. InFuse provides a comprehensive data fusion framework or toolset to fuse and interpret sensor data from multiple sensors. This project represents an optimal approach to develop software for robotics: a standardized and comprehensive development environment for industrial applications, with particular focus on space applications where components can be connected, tested offline, evaluated and deployed in any preferred robotic framework, including those devised for space or terrestrial applications. This paper discusses the results of verification and validation of data fusion methods for robots deployed in orbital and planetary scenarios using data sets collected in simulation and outdoor analogue campaigns.

1. OBJECTIVE

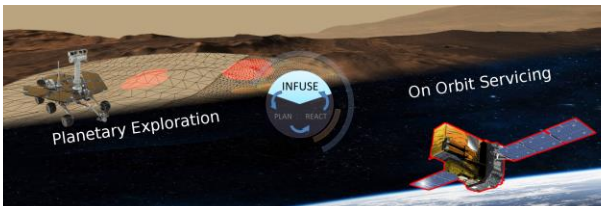


Figure 3. Illustration of reference scenarios addressed by InFuse.

In the context of perception, localization and mapping in space robotics, common evaluation and deployment frameworks are crucial for efficiently developing reliable robotics solutions. Qualification of software for space follows a complex approach that is challenging. Methods and tools that make it easier to develop software for space and evaluate it as early as the prototype stages are highly valuable [1]. InFuse addresses perception and data fusion within the context of two reference scenarios of space robotic: planetary exploration and on-orbit servicing [2], as depicted in Fig. 1.

Perception and data fusion capabilities are selected for answering the need to perceive, analyse and interpret the environment as well as satisfying constraints related to reliability, memory footprint and computation complexity, inherent to space vehicles. InFuse [3] represents “Sensor Fusion”, and provides a Common Data Fusion Framework (CDFF) containing the following features:

1. Library of reusable Data Fusion Nodes (DFNs) implementing sensor fusion algorithms and validated for critical applications
2. An easy method to quickly and reliably combine DFNs into software modules (Data Fusion Processing Compounds (DFPCs) meant for processing sensor data
3. Tools for data collection and offline testing of DFPCs
4. Suitable convertors between different representations of sensor data
5. Convenient integration with robotic control frameworks such as ROS, ROCK or GenoM.

This paper describes the background and reference space mission scenarios, for which data fusion solutions were developed. It also presents approaches and scope for innovation in novel data fusion approaches to achieve perception, mapping and localization for specific applications. The paper aims to demonstrate the developments, results and usability of the CDFF software components that can be composed to create functional and re-usable sensor fusion solutions that can be deployed on robotic agents.

2. CDFF OVERVIEW AND ARCHITECTURE

InFuse provides navigation and perception functionalities. Navigation pertains to positioning in general: of a rover, of particular elements in the environment, of a servicing satellite with respect to an object in space to service. Perception pertains on the one hand to the detection and modelling of the environment (be it the whole environment surrounding the rover or some specific elements in the environment), and on the other hand to track specific environment features during motions. Mature state of the art perception and localization libraries are integrated into the framework (Fig. 1) to cover these objectives, as well as more commonly used libraries within the robotics domain.

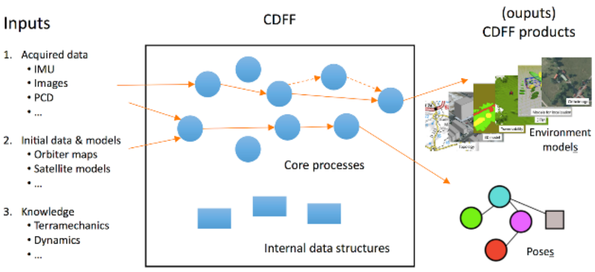


Figure 1. CDFF Data Fusion framework overview

The CDFF primarily supports sensory input from the standard suite of sensors [6] currently used in orbital and planetary applications (Tab. 1). The possibility to further extend the CDFF to include support of other types of sensors in the future is available by design.

Table 1. Sensor data and associated sensors types

Type of Data	Name of Sensor
Image Data	Close-up High-Resolution Camera
Extended Image Data	Hyper / Multispectral Camera
Depth Image Data	TOF Camera
Depth Image Data	Structured Light Vision
Depth Image Data	Visible Stereo Camera
Depth Image Data	IR Stereo Camera
Depth Data	3D LIDAR
Planar Depth Data	2D LIDAR
Force Data	Contact Sensor
Torque Data	Force/Torque Sensor
Angular Data	Joint Position Encoder
Angular Magnetic Data	Magnetic Field Sensor (Magnetometer)
Angular Velocity Data	Angular Velocity Sensor (Gyroscope)
Angle of Acceleration Data	Linear Acceleration Sensor (Accelerometer)

2.1. CDFF-Core

The CDFF-Core [4] provides the "core" data fusion set of state of the art algorithms and techniques currently in use and applicable to the Planetary and Orbital RIs within the scope of Infuse, as a collection of ready-to-use libraries or packages. These libraries represent the CDFF's processing methods necessary to fuse sensory data. The CDFF-Core is implemented in a modular fashion with a common interface to allow high flexibility in configuration as well as in operation as a distributed system on multiple platforms. The core libraries are to be deployed on the target system (robotic platform) as well as on the designer's environment.

Examples of core libraries include low-level functions such as feature detection, registration and recognition, data association, state estimation, outlier removal, and filtering which can be assembled into building environmental representations, achieving 3D object reconstruction or SLAM. The preliminary set of algorithms CDFF core libraries are shown in Tab. 2.

2.2. CDFF-Support

The CDFF-Support [4] provides the necessary tools for the instantiation and execution of Data Fusion Processing Compounds. Under the CDFF-Support functionalities can be found the DFPC, the Orchestrator, and the Data Product Management Tool (DPM) as shown in Fig. 2.

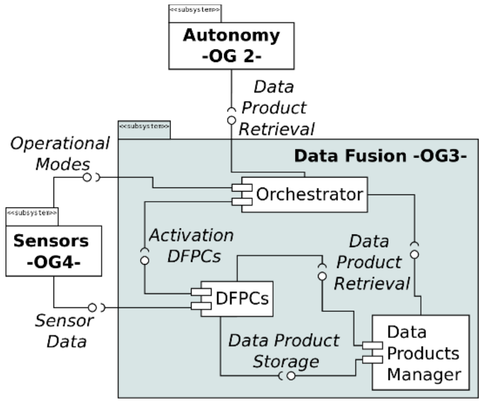


Figure 2. Interaction among CDFF-Support components and interfaces with OG2 and OG4

The Data Fusion Processing Compound (DFPC) is a combination of DFNs designed to provide a certain data product (e.g. pose estimation, map...). DFPCs defines connections between input and output ports of DFNs.

1. The Orchestrator is the component that deals with the activation and deactivation of DFPCs. It is also the component that receives and answers the requests from the Autonomy Module – Operational Grant (OG2). The selection of one or another DFPC is done based on availability and quality of data sources and on the data product required.
2. The Data Product Management (DPM) Tool acts as a long-term memory [5] for data fusion products generated by the DFPCs to be used by planners.

Along with the CDFF-Core entities, all components of CDFF-Support will be deployed in the Target System. They will also be available for programming and testing in the Developer Environment. Some of the data fusion methods (DFPCs) developed for the planetary and orbital tracks are as follows: visual odometry, visual SLAM, visual Map-based Localisation, 3D model detection and tracking, 3D reconstruction of objects, Lidar based tracking, point cloud model-based localization, haptic scanning, absolute localization, DEM (Digital Elevation Model) generation, Lidar pose graph SLAM, generation of navigation maps.

3. SCENARIOS

InFuse addresses perception and data fusion within the context of two reference scenarios of space robotic: planetary exploration and on-orbit servicing, as depicted in Fig. 1. Perception and data fusion capabilities are selected for answering the need to perceive, analyse and interpret the environment as well as satisfying constraints related to reliability, memory footprint and computation complexity, inherent to space vehicles.

The planetary track focuses on surface exploration, autonomous navigation [2] and science, and rendezvous. A nominal sequence of tasks done by a rover is : a) Creation of an initial panorama, b) Production of navigation maps, c) Path planning, d) Path execution and hazard detection / avoidance, e) Rover self-localization (odometry / SLAM), f) Updating of navigation maps, g) Back to d. or detection of a point of interest - POI, h) Visual servoing towards the POI i) Soil sample acquisition, j) Global localisation using orbiter data, k) Going back to lander, l) Visual servoing towards the lander, m) Visual servoing of the robotics arm for sample transfer.

The orbital track focuses on rendezvous for on-orbit servicing operations such as refueling, re-configuration of hardware modules and repair of parts. A nominal sequence of tasks performed by a chaser spacecraft is: a) Detection of a target satellite far away (it might be done manually by ground observation to determine its orbit), b) Bearing tracking, c) Initial approach, d) 3D modeling of the satellite, e) 3D tracking, f) Final approach, g) Visual servoing of the robotics arm, h) Docking and berthing.

Within these scenarios, InFuse deals with perception and data fusion. This translates into data production for the Autonomy Framework. Three data classes are identified: localization (self or w.r.t a target), landmark / object management (detection and tracking), environment modeling (DEM, navigation maps, structured 3D models). From this point of view, we identify a few data production use cases covering both reference scenarios:

Planetary exploration [7]:
a) Rover localization in its environment
b) Relative localization of or w.r.t a fixed or moving asset
c) Production of DEM and navigation maps

One possible illustration of the use case "Rover localization in its environment" is provided in Fig. 4.

On-orbit servicing:
a) Bearing only localization for approach (target position and direction estimation w.r.t. chaser / Long range)
b) Localization w.r.t satellite for rendezvous and orbital parameter estimation (target pose estimation w.r.t. chaser close perception)
c) Target pose estimation w.r.t. chaser / Docking (Visual servoing)
d) 3D reconstruction of a target.

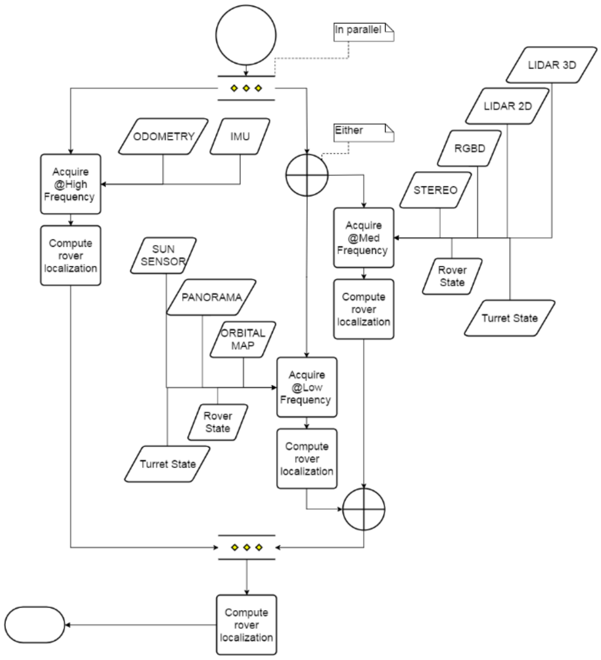


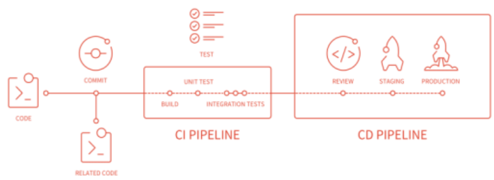
Figure 4. Functional description of a set of DFPCs within the use case for Rover localization

4. TESTING AND VALIDATION APPROACH

These DFPCs were validated using a rich data set from DLR OOS-Sim [10] for Orbital Track (OT) scenarios. The Planetary Track (PT) scenarios involved the use of mobile robots - Mana & Minnie from LAAS-CNRS, ExoMars BB2 from DLR and Sherpa from DFKI in indoor and outdoor Martian analogs [11]. Some of the verification and validation results are presented in this paper. Additionally, an offline validation tool using data sets were used to test the functional aspects of DFPCs using these data sets, profile the algorithms and visualize the results. A continuous integration solution has been deployed for remote testing of DFNs and DFPCs, analysis of memory leaks and performance metrics of unit tests to maintain a desired quality of developed software.

4.1. CONTINUOUS INTEGRATION SUPPORT

Integrating many contributions into a solid framework requires good software practices. In order to improve the Technology readiness level of Infuse, a continuous integration approach was used. The full source code and CI pipeline (Fig. 5) are available at this address. https://gitlab.com/h2020src/OG3.



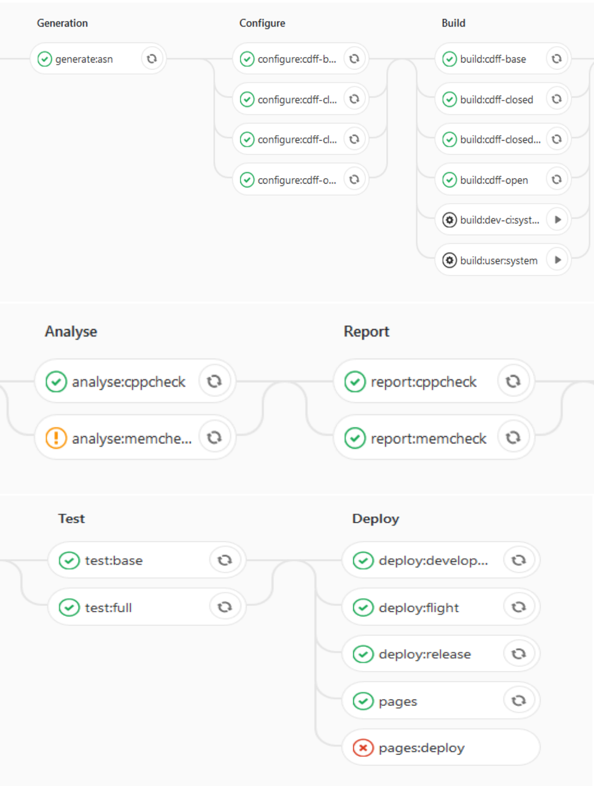


Figure 5. The CI pipeline used during Infuse.

Before accepting code into the main stable branch, a series of automatic checks and rules are applied to the code. Code is accepted once the requirements are satisfied. The CI includes MISRA C and MISRA C++ coding guidelines, CPPCheck as static analyser, Valgrind as dynamic analyser, Catch as testing framework and Gitlab-CI as CI/CD framework.

In addition to the integration testing, deployment testing is conducted in order to verify that Infuse will work on a variety of environments and improve the solidity of the whole framework. For this purpose, four different docker images are used to simulate different levels of dependencies installations (required, optional and closed sources).

5. DFPC VALIDATION RESULTS

5.1. Model-based visual tracking – OT

This DFPC implements a model-based tracker developed specifically for the Orbital Track. The validation was performed by DLR using DLR's OOS-Sim dataset, consisting on close-range recordings of a target and servicer satellites performing manoeuvres in various lighting conditions.

The lighting conditions explored in the dataset are the following:

- Eclipse condition: no sun simulator, using target illumination unit
- Suboptimal lighting condition: sun simulator at 90 deg wrt the optical axis of the camera
- Optimal lighting condition: sun simulator at 45 deg wrt optical axis of the camera

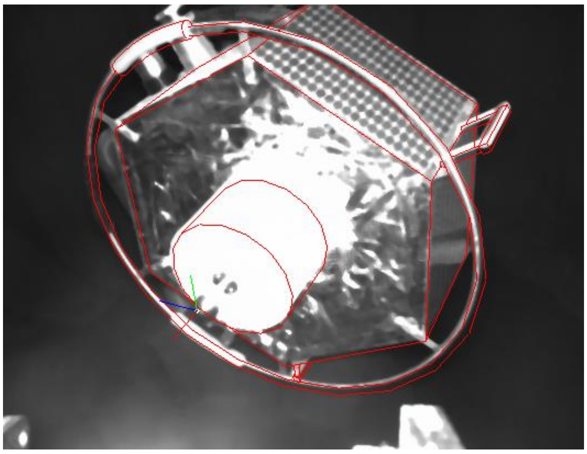


Figure 6. Visualization of the algorithm output.

The test cases in the dataset in terms of motion profiles are the following:

- Predominant translational motion along z-axis of the servicer
- Predominant rotational motion about either x (lateral) or y axis (longitudinal) of the servicer
- Roto-translation motion with angular and linear velocity of 1 deg/s and 1cm/s respectively

The complete pose of both target and servicing satellites are known and used as ground truth for the tracker results comparison. The validation criteria used in these tests is the following:

- Position error < 0.05m
- Orientation error < 5 degrees
- Update frequency > 0.1Hz

Table 2: Test results summary per scenario

Scenarios	Result
10 different tests with different conditions: Target: stationary, rotation or translation Servicer: stationary, translation Lighting: eclipse condition, 90 degrees (suboptimal condition) & 45 degrees (optimal condition)	Passed
Target: 60 deg rotation about y-axis Servicer: stationary Lighting: eclipse condition Target: stationary Servicer: 70 cm translation along x -axis, with a velocity 1 cm/s. Lighting: eclipse condition Target: rotation of 60 deg about x-axis, with angular velocity 1 deg/s Servicer: 70 cm translation along x-axis, with a velocity 1 cm/s. Lighting: eclipse condition	Partially passed
Target: stationary Servicer: 70 cm translation along x -axis, with a velocity 1 cm/s. Lighting: 90 degree, suboptimal condition	Not passed

5.2. Mid-Range 3D Model Tracking – OT

This DPFC implements a model-based tracker developed by Magellium for the Orbital Track. As the previous one, the validation was performed using DLR's OOS-Sim dataset. Two motion profiles have been analysed, each of them in the three lightning conditions that are contained in the dataset, namely eclipse, suboptimal lighting and optimal lightning conditions.

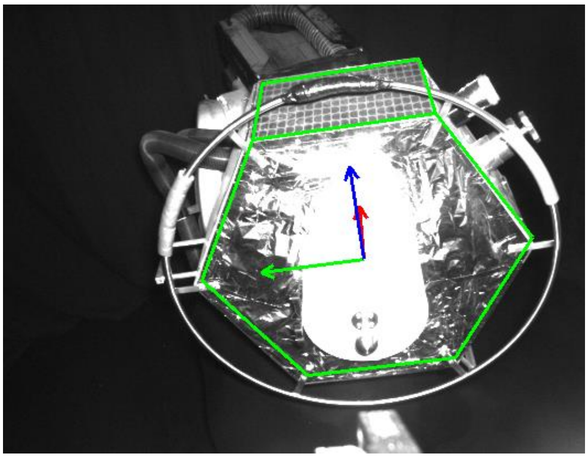


Figure 7. Visualization of the algorithm output.

The validation criteria for these tests is the following:

- Position error < 10 cm at 10 m range
- Angular accuracy < 10 degrees

Table 3: Test results summary per scenario

Scenarios	Result
Lighting condition: eclipse / suboptimal / optimal Target: stationary Servicer: 70 cm translational motion in +x direction (1 cm/s)	Passed
Input data are left images from a stereo-camera pair. Lighting condition: eclipse / suboptimal / optimal Target: 60 deg rotation about the x-axis Servicer: 70 cm translational motion in +x direction	Passed

5.3. Model Based Tracking – PT

This DFPC implements a model based tracker [9] developed by Space Applications Services for the Planetary Track and specialized in detecting wheeled rovers. The validation of this DFPC was performed on two different types of datasets:

- Dataset 1 [DS1] – Mana & Minnie: In this dataset, the target was the Mana robot and the Minnie robot is static and gathering the data through its stereo camera. The pose of both robots was obtained using a DGPS system coupled with IMU information. This information was used as the ground truth for the experiments.
- Dataset 2 [DS2] – Sherpa + HCRU**:** This dataset was performed using DLR's Handheld Central Rover Unit (HCRU), developed as part of the InFuse project, and DFKI's SherpaTT rover. The SherpaTT rover had another HCRU unit integrated inside it that recorded the output of the sensors mounted on of both the robot and the HCRU. The pose of the SherpaTT rover was obtained using a DGPS system coupled with IMU information. This information was used as the ground truth for the experiments.

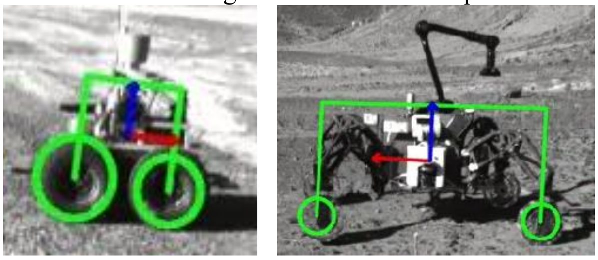


Figure 8. Visualization of the algorithm output for DS1-Mana & Minnie (left) and DS2 - Sherpa (right).

Given the specific characteristics of the two datasets, the validation criteria vary from one to the other.
The validation criteria for DS1 are the following:

- Position error
  - x axis < 15 cm
  - y axis < 30 cm
  - z axis < 15 cm
- Orientation error
  - x axis < 20 degrees
  - y axis < 20 degrees
  - z axis < 10 degrees

The validation criteria for DS2 are the following:

- Position error
  - x axis < 45 cm
  - y axis < 30 cm
  - z axis < 15 cm
- Orientation error
  - x axis < 30 degrees
  - y axis < 30 degrees
  - z axis < 20 degrees

Table 4: Test results summary per dataset, specifying the range distance in which the target is moving

DS	Scenario	Distance	Result
DS1	Straight lines (3 passes)	[4.6, 6.7] m	Passed
	Slanted path, from left to right and front to back	[7.9, 13] m	Partially passed
	Slanted path, from right ot left and back to front. Sun to the left of the camera, shadowing the wheels of the robot	[6, 12] m	Not passed
DS2	Straight path, from right to left	[5.2, 5.4] m	Passed
	Slanted path, from left to right and back to front	[7.7, 6.8] m	Partially passed
	Straight path, from right to left	[10.1, 11.55] m	Not passed

5.4. Reconstruction 3D – PT

The validation of this DFPC was performed by University of Strathclyde using different datasets, all of them recording an object that is constantly in the field of view of the camera while said camera moves in a circle around the object. The datasets used for validation are the following:

- Dataset 1 [DS1]: undistorted and rectified images of a crawler robot in DLR's PEL facility. Recorded with diffused light condition. The object is colourful, has many 3D features and doesn't have reflecting surfaces.
- Dataset 2 [DS2]: undistorted and rectified monochrome images of a boulder in the desert of Morocco. Recorded with sun illumination in the early morning, which adds long shadows to the dataset. The object has many 3D features and no reflecting surfaces.

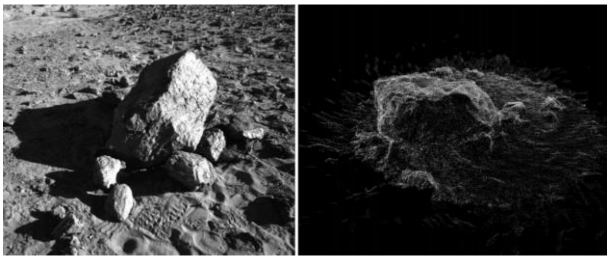


Figure 9. Boulder input from DS2 dataset (left) and visualization of the algorithm output (right).

The validation procedure includes the computing of the following metrics:

- Percentage of outliers = (Number of outliers) * 100 / (Number of points)
- Position Estimation Error (PEE) = ABS (measured camera distance - ground truth camera distance) *100/ (camera operational distance)
- Shape Similarity (SS)= 100*(1 - (average of absolute difference of dimensions) / (maximum dimension))
- Maximum dimensional analysis error (MDE) = 100 * maximum of ((difference of dimensions) / (ground truth dimension))

The validation criteria for these tests is the following:

- Outliers < 10%
- PEE < 1% of camera operational distance
- SS >= 90%
- MDE < 10%

Table 5: Test results summary per dataset. All numerical values are represented as percentage.

DS	Outliers	PEE	SS	MDE	Result
DS1	0.26	0.46	95	22.96	Not passed
DS2	0.00	2.53	97.98	7.34	Passed

5.5. DEM Building – PT

This DFPC implementation of a DEM building algorithm was developed by LAAS-CNRS for the Planetary Track. The validation was performed using the following datasets:

- Dataset 1 [DS1]: indoors test in DLR's PEL facility, using a robot moving towards a cube at a 15 degrees' slope.
- Dataset 2 [DS2]: Merzouga site 1km trajectory during Morocco trials. The rover has a linear speed of 40 cm/s and the laser scans are acquired at 1Hz.
- Dataset 3 [DS3]: Mummy site 800m trajectory during Morocco trials. The rover has a linear speed of 40 cm/s and the laser scans are acquired at 1Hz.

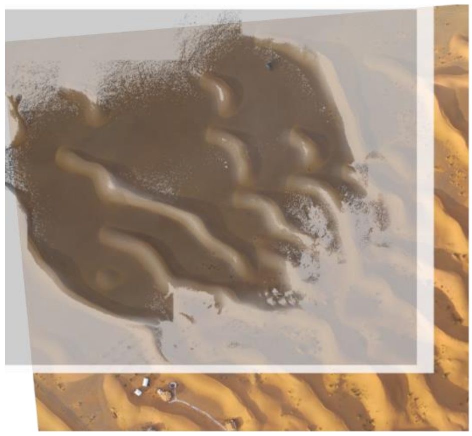


Figure 10. Resulting DEM from DS2 superposed on top of the ground truth map of the area.

Table 6: Test results summary per dataset

DS	DEM size	Resolution	Result
DS1	10x10 m	4 cm	Passed
DS2	30x30 m	10 cm	Passed
DS3	30x30 m	10 cm	Passed

There is no quantitative assessment of the precision of the DEM built, as it would not only require a ground truth of the terrain, but also an absolute precision of the DEM localization with respect to the ground truth with a precision below the DEM pixels. Also, the uncertainties in the estimation of the elevation come mainly from the precision of the acquired data and of the robot localization at the time of the data acquisition, not from the DEM building process itself.

5.6. Visual Odometry – PT

The validation of this DFPC was performed by Magellium [8] using different datasets:

- Dataset 1 [DS1]: Undistorted and rectified images captured by the HCRU in DLR's PEL facility while moving back and forth at ~2cm/s, with 2x 180° turn in place. The rectified images have a resolution of 1032x772 pixels, and are received at a rate of ~4Hz. The surface is flat with very few rocks. Full intensity lab lightning is used.
- Dataset 2 [DS2]: Raw images captured by Minnie's NavCam while performing a 1km long trajectory in Merzouga site during the Morocco field trials. The rover has a linear speed of 30cm/s. The input images are acquired at ~4Hz and have a resolution of 1920x1080, but are degraded by a factor of 3 in both height and width before processing by the visual odometry.

- Dataset 3 [DS3]: Raw images captured by Minnie while performing a 650m long trajectory in the Kess-kess site during the Morocco field trials. The rover has a linear speed of 30cm/s. The input images are acquired at ~2Hz and have a resolution of 1920x1080, but are degraded by a factor of 3 before processing by the visual odometry

The acceptance criteria for these tests is that the relative localisation error shall be less than 2% of the total travelled distance.

Table 7: Test results summary per dataset

DS	Error [%]	Result
DS1	1	Passed
DS2	2	Passed
DS3	2.61	Not passed

5.7. Visual Map-based Localisation – PT

This DFPC implementation by Magellium a Visual SLAM algorithm that uses maps of the traversed paths that were recorded previously. The validation of this DFPC was performed using a dataset containing raw images captured by Minnie while performing a 650m long trajectory in the Kess-kess site during the Morocco field trials. For the purpose of this test, we only replayed half of the trajectory. The rover has a linear speed of 30cm/s. The input images are acquired at ~2Hz and have a resolution of 1920x1080.

The map was obtained using another dataset capturing the same path in the Kess-kess site. Similar to the previous test results, the acceptance criteria for these tests is that the relative localisation error shall be less than 2% of the total travelled distance. The test was passed, with a final relative position accuracy is ~1.5%. The localisation was less sensitive to perturbations in rover pitch than other SLAM implementations. This could be explained by the fact that the second dataset contains stereo images acquired at 4Hz, but also by the pre-existing SLAM map which acts as a support, preventing diverging estimations.

5.8. Pose-Graph SLAM – PT

The validation was performed by LAAS-CNRS by analysing the error on the rover position, using the rover's DGPS position as ground truth. More specifically, the following parameters were analysed:

- Growth of the robot position error as a function of distance in the absence of loop closures
- Reduction of the error after a loop closure is detected and integrated in the robot pose estimation

The validation of this DFPC was performed using the different datasets. The following paragraphs present the scenarios and the results obtained per scenario.

Scenario 1: This dataset consists on lidar scans acquired by the Mana rover while traversing a 200m long trajectory on the Merzouga site. Fig. 11 and Fig. 12 show the result of the PG SLAM algorithm, comparing the estimated poses with the ground truth coming from the DGPS for the X and Y axis as well as the Z (elevation) axis.

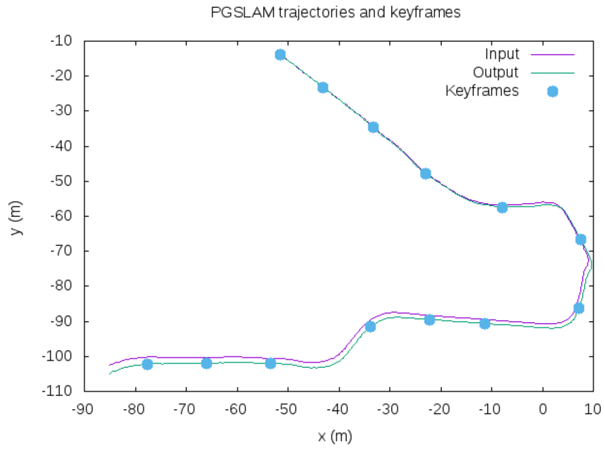


Figure 11: PGSLAM results compared with the ground truth for the x and y axis.

While the approach has shown good time performances, the errors in the estimation of the robot pitch angle yields a significant drift in the estimation of the robot elevation. The solution would be to adapt the core algorithm of PG-SLAM (Iterative Closest Point algorithm) so that the estimated position is constrained by the robot attitude angles estimated by the odometry/IMU position estimate.

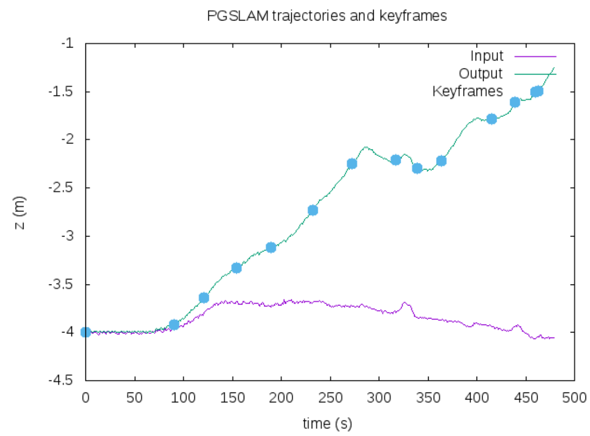


Figure 12: PGSLAM results compared with the ground truth for the z axis.

Scenario 2: This dataset consists on lidar scans acquired by the Mana rover while traversing a 1000m long trajectory on the Mummy site. Fig. 13 and Fig. 14 show the result of the PG SLAM algorithm, comparing the estimated poses with the ground truth coming from the DGPS for the X and Y axis as well as the Z (elevation) axis. The errors in the estimation of the robot pitch angle yields a significant drift in the estimation of the robot elevation, and a gross heading estimation error made the trajectory estimate totally wrong after about 200 m of motions. While some loop closures have been properly detected and yielded a spatially consistent estimate of the trajectory in its northest part, no loop closure could be detected when the robot came back to the position where the gross angular estimation has been made.

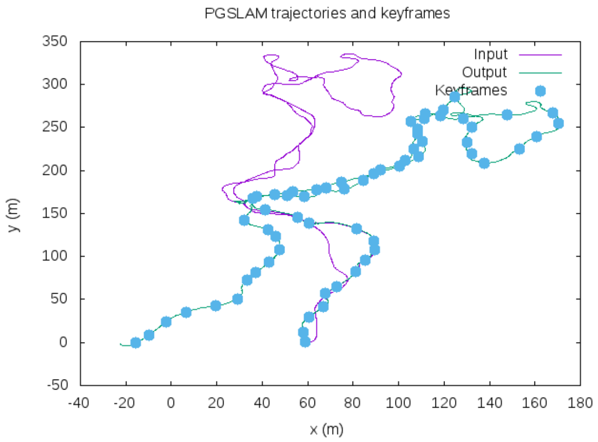


Figure 13: PGSLAM results compared with the ground truth for the x and y axis.

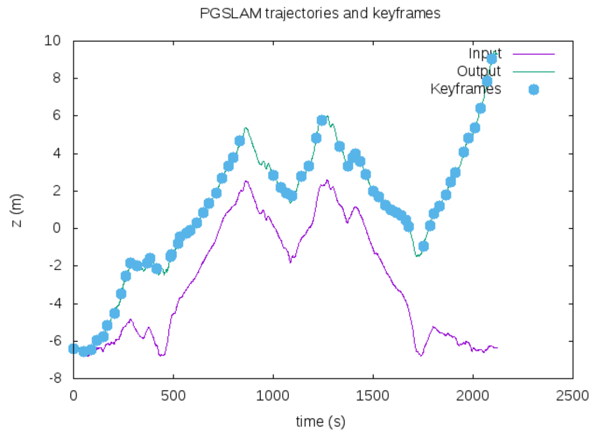


Figure 14: PGSLAM results compared with the ground truth for the z axis.

5.9. Performance evaluation

This section offers a brief performance evaluation of the presented DFPCs. The aim of this evaluation is to give an idea as to what the execution time and maximum memory usage is for some of these algorithms.

Table 8: Test results summary per dataset

DFPC	Execution time [s]	Max memory
Model based visual tracking	0.001848	491.3 MB
Mid-Range 3D Model Tracking	0.1311	391.6 MB
Model Based Tracking	0.0587	1.152 GB
3D reconstruction	1.0006	778.7 MB
DEM building	0.0058	361.9 MB
Visual Map-based Localisation	0.0287	847.8 MB

The performance evaluation was performed analysing the unit tests. All binaries were profiled using the same machine, in order to produce comparable results. The user time obtained when executing the binaries was used to create the final numbers, being this the actual CPU time used in the process, without taking into account other processes or time the process gets blocked. The user time is then multiplied with the processing percentage per function obtained using callgrind, and the maximum memory consumed by the process is extracted using massif, both of them Valgrind tools [12].

6. CONCLUSION

InFuse provides a comprehensive, modular data fusion system for future space robotic missions. InFuse consists of optimized perception techniques to estimate localization and model the surroundings of a robotic platform in order to support the planning and execution. The results of the validation and verification tests with data sets provide an overview of the current performance. In addition, InFuse aims to increase the overall performances of the offered DFPCs by providing tools to perform fast prototyping with core data fusion libraries before their final integration on the target platform. Ideally, this approach will bring improvements in the parameterisation of DFNs and will enable a rapid and probably an autonomous reconfiguration of DFPCs.

7. ACKNOWLEDGEMENTS

InFuse is funded under the European Commission Horizon 2020 Space Strategic Research Clusters - Operational Grants, grant number 730014.

REFERENCES

1. Ofir, C., Yael, E. (2008). A sensor fusion framework for online sensor and algorithm selection. RAS, Volume 56, Issue 9, Pages 762-776.

2. Gianfranco, V., Daniel, N., Michel D., Raffele, M., Javier R., Daniel J. (2015). D3.1-Compendium of SRC activities. EC H2020 SRC in Space Robotics Technologies.

3 Govindaraj, S., et al., (2017) InFuse: A Comprehensive Framework for Data Fusion in Space Robotics, ASTRA 2017.

4. Dominguez, R., et al., A common data fusion framework for space robotics - architecture and data fusion methods. 14th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Madrid, Spain, 4-6 June, 2018.

5. Carrió, J.H., Arnold, S., Böckmann, A., Born, A., Domínguez, R., Kirchner, F. (2016). EnviRe - Environment Representation for Long-term Autonomy. In AI for Long-term Autonomy Workshop of the Int. Conf. on Robotics and Automation (ICRA).

6. Chandrasekaran, B.; Gangadhar, S.; Conrad, J.; A survey of multisensor fusion techoniques, architectures and methodologies, IEEE SoutheastCon, USA, March 2017.

7. Maurette, M. (2003). Mars rover autonomous navigation. Autonomous Robots, 14(2-3):199-208.

8. Souvannavong, F., Lemarechal, C., Rastel, L., Maurette, M. (2010). Vision-Based Motion Estimation for the ExoMars Rover, I-SAIRAS

9. Issa, A., Nesnas, D., Max, B., Richard, Madison. (2004). Visual Target Tracking for Rover-based Planetary Exploration, Proceedings of the IEEE Aerospace Conference.

10 Oumer N.W. et al. (2018). Orbital Reference Implementation and Associated EGSE: Detailed Design. Deliverable 5.1 of the Horizon 2020 InFuse Project. URL: https://www.h2020-infuse.eu/documents.

11. Souvannavong F. et al. (2018). Planetary Reference Implementation and Associated EGSE: Detailed Design. Deliverable 5.2 of the Horizon 2020 InFuse Project. URL: https://www.h2020-infuse.eu/documents.

12. Valgrind. Electronic Notes in Theoretical Computer Science 89:44–66