InFuse Data Fusion Methodology for Space Robotics, Awareness and Machine Learning

Mark Post, Romain Michalec, Alessandro Bianco, Xiu Yan, Andrea de Maio, Simon Lacroix, Jérémi Gancet, Shashank Govindaraj, Xavier Martinez-Gonzalez, Iyas Dalati, et al.

InFuse Data Fusion Methodology for Space Robotics, Awareness and Machine Learning

Mark Post[a]* Romain Michalec[a] Alessandro Bianco[a] Xiu Yan[a] Andrea De Maio[b] Quentin Labourey[b] Simon Lacroix[b] Jeremi Gancet[c] Shashank Govindaraj[c] Xavier Martinez-Gonzalez[c] Iyas Dalati[c] Raúl Domínguez[d] Bilal Wehbe[d] Alexander Fabisch[d] Enno Röhrig[d] Fabrice Souvannavong[e] Vincent Bissonnette[e] Michal Smíšek[f] Nassir W. Oumer[f] Lukas Meyer[f] Rudolph Triebel[f] Zoltán-Csaba Márton[f]

[a] *University of Strathclyde, Department of Design, Manufacture, and Engineering Management, 75 Montrose St., Glasgow G1 1XJ, United Kingdom, mark.post@strath.ac.uk*
[b] *LAAS-CNRS, 7 avenue du Colonel Roche, 31031 Toulouse, France, simon.lacroix@laas.fr*
[c] *Space Applications Services NV, Leuvensesteenweg 325, 1932 Zaventem, Belgium, jeremi.gancet@spaceapplications.com*
[d] *DFKI, Robert-Hooke-Straße 1, 28359 Bremen, Germany, raul.dominguez@dfki.de*
[e] *Magellium SAS, 24 rue Hermès, 31521 Ramonville-Saint-Agne, France, fabrice.souvannavong@magellium.fr*
[f] *DLR, Münchener Straße 20, 82234 Weßling, Germany, michal.smisek@dlr.de*
* Corresponding Author

Abstract

Autonomous space vehicles such as orbital servicing satellites and planetary exploration rovers must be comprehensively aware of their environment in order to make appropriate decisions. Multi-sensor data fusion plays a vital role in providing these autonomous systems with sensory information of different types, from different locations, and at different times. The InFuse project, funded by the European Commission's Horizon 2020 Strategic Research Cluster in Space Robotics, provides the space community with an open-source Common Data Fusion Framework (CDFF) by which data may be fused in a modular fashion from multiple sensors. In this paper, we summarize the modular structure of this CDFF and show how it is used for the processing of sensor data to obtain data products for both planetary and orbital space robotic applications. Multiple sensor data from field testing that includes inertial measurements, stereo vision, and scanning laser range information is first used to produce robust multi-layered environmental maps for path planning. This information is registered and fused within the CDFF to produce comprehensive three-dimensional maps of the environment. To further explore the potential of the CDFF, we illustrate several applications of the CDFF that have been evaluated for orbital and planetary use cases of environmental reconstruction, mapping, navigation, and visual tracking. Algorithms for learning of maps, outlier detection, localization, and identification of objects are available within the CDFF and some early results from their use in space analogue scenarios are presented. These applications show how the CDFF can be used to provide a wide variety of data products for use by awareness and machine learning processes in space robots.
Keywords: (Sensor Fusion, Rover, Space Robotics, Classification)

Introduction

Robotics, intended as the integration of mechanical, electrical and software components is a very young discipline. Starting from the assembly of mechanical components that superficially mimic human actions, its goal evolved ever more ambitiously to the endowment of total human capability to mechanical components. Among these capabilities is the capacity to process the information from the senses, and acquire a more complete and general understanding of situation and places, in order to carry actions and pursue objectives. This more limited goal is the target of many industrial and scientific applications, such as space exploration, autonomous driving, manufacturing, rescue mission in dangerous situations, precision surgery, just to name a few.

Practices and standards in robotic development are continuously in evolution as new technology opens the possibility to new actuation and sensing methods, and increased computational power allows the robot to "think" more thoroughly about the operation carried out. In the context of software development of robotic applications, this evolution starts with general programming: modules were developed ad hoc, possibly integrating software components according to the application needs. During these efforts, real time communication of robotic processes was a common recurring problem in many applications. These

problems were addressed by modern robotic frameworks such as ROS. Their widespread acceptance had led to the growth of framework communities. They started to contribute to the frameworks by writing modules for the solution of some common robotic problems such as sensor data representation, environmental mapping and path planning. However, no industrial standard has ever been set, and these frameworks have been mostly used for rapid prototyping.

The InFuse project is our effort to move robotic software development practices to the next step: a standardised and comprehensive development environment for industrial applications, with particular focus on space applications. InFuse represents "Sensor Fusion", and it aims to develop a Common Data Fusion Framework (CDFF) containing the following features: (i) a library of reusable Data Fusion Node (DFN) modules implementing sensor fusion algorithms and validated for industrial applications, (ii) an easy method to quickly and reliably combine the DFNs into software modules (Data Fusion Processing Compounds, or DFPCs) meant for sensor data processing, (iii) tools for data collection and testing of the DFPCs, (iv) suitable conversion mechanisms between different representations of the sensor data, and (v) convenient integration with process communication frameworks such as ROS.

Previous papers have focused on the design of the framework [1] and the methods implemented to accomplish data fusion [2]. In this paper, we demonstrate the use of InFuse for space robotics tasks through its application as a means of environmental mapping and structural classification of the environment.

Description of the Common Data Fusion Framework (CDFF)

The framework consists of three major components, named CDFF-Core, CDFF-Support, and CDFF-Dev. CDFF-Core is a set of basic data fusion modules called Data Fusion Nodes (DFNs), each of which performs a specific data processing task commonly part of robotic perception tasks, for instance, feature extraction in an image or Kalman filtering.

CDFF-Support is a set of more fully-fledged Data Fusion Processing Compounds (DFPCs), assembled by connecting DFNs together into larger software modules that generate specific data fusion products (e.g. pose estimation) from specific sensor data inputs. CDFF-Support also includes two software modules required for the actual execution of these DFPCs on a robotic system: an Orchestrator which coordinates the data fusion processes running on the system, and a Data Product Manager which maintains the data fusion products pertaining to environment representation during the lifetime of the system.

CDFF-Dev provides software development, performance analysis and data management tools for implementing and evaluating data fusion algorithms. Contrary to CDFF-Core and CDFF-Support, which are deployed on the robotic system, CDFF-Dev tools are meant to be used in a development environment during implementation and exploitation activities.

Environmental Reconstruction

The most fundamental use of the CDFF is to reconstruct a model of the environment around a robot by fusing data from multiple sensors or multiple samples. We have applied two main approaches to the reconstruction of 3D objects from pairs of stereo images. During both approaches we use a rectified image pair to construct a 3D point cloud by computation of a disparity map [3][4] as implemented in the OpenCV and Point Cloud Library (PCL) libraries. Then, 3D point clouds are fused together by locating them in the coordinate system aligned with the first camera pose. The two approaches differ in the way the pose of each point cloud is computed in the main reference system.

(1) The first approach is based on 3D registration: (i) we extract 3D features from a point cloud [5], (ii) we compute their descriptors [6], and (iii) we find the transform that allows to overlap one set of keypoints over the other [7].

(2) The second approach is based on 2D matching: (i) we extract 2D features from each image [8], (ii) we compute their descriptors [9], (iii) we find the best matches between left and right images and we compute the 3D points they represent by triangulation [10], (iv) we find the best matches between consecutive left images so that we can associate the current features with the past triangulated 3D points at a previous instant in time, (v) matches are filtered out on the base of the fundamental matrix relation, then (vi) we compute the position of the left camera with respect to the same left camera at a previous time instant by mean of the perspective-n-point [11] solver as implemented in OpenCV.

Table 1: Steps involved in 3D object reconstruction. The column on the left lists the DFNs that make up a 3D object reconstruction DFPC. The column on the right

lists examples of algorithms that can make up a DFN: indeed, a variety of algorithms may be used for the purpose of a specific DFN, or in other words, a DFN may have multiple algorithmic implementations.

3D registration approach	
DFNs used successively	For instance
ImageFiltering StereoReconstruction FeatureExtraction3D FeatureDescription3D FeatureMatching3D	UndistortionRectification DisparityMapping HarrisDetector3D SHOTDescriptor3D RANSAC3D
2D feature matching approach	
DFNs used successively	For instance
ImageFiltering StereoReconstruction FeatureExtraction2D FeatureDescription2D FeatureMatching2D Reconstruction2DTo3D PerspectiveNPoint FundamentalMatrixComputation	UndistortionRectification DisparityMapping HarrisDetector2D ORBDescriptor2D FLANNMatcher Triangulation IterativePNPSolver RANSAC

As part of CDFF-Support, we provide three variants of each approach to 3D reconstruction, in other words six different algorithmic implementations of a 3D object reconstruction DFPC.

(1) The first variant of the 3D registration approach works as described in Table 1. It is implemented using algorithms available in the PCL library [12]. The second variant uses ICP on the extracted features instead of feature description and feature matching. It uses the ICP implementations available in the PCL and in CloudCompare [13], encapsulated in a DFN named Registration3D. The third variant uses ICP on the whole reconstructed point cloud, rather than just on features.

Table 2: Final steps of 3D object reconstruction, in the 3D registration approach

3D registration approach: three DFPC variants		
Registration FromStereo	Sparse Registration FromStereo	Dense Registration FromStereo
FeatureExtraction3D FeatureDescription3D FeatureMatching3D	FeatureExtraction3D - Registration3D (ICP)	- - Registration3D

(2) The first variant of the 2D matching approach works as described in Table 1. It reconstructs 3D scenes and objects from stereo images using algorithms available in OpenCV. The other variants are extensions which optimize the time series of camera pose estimates under geometrical 3D transformation constraints, for the second variant, and projection matrix constraints defined by the matches between triangulated 3D points and their 2D image pixels, for the third variant. Both optimization problems are solved numerically using the Ceres library [14].

Table 3: Optimization steps involved in 3D object reconstruction, in the 2D feature matching approach

2D feature matching approach: three DFPC variants		
Reconstruction FromStereo	Estimation From Stereo	Adjustment From Stereo
-	Transform3DEstimation (least squares optimization)	BundleAdjustment (using Ceres)

Those six DFPC implementations have been tested using data from ROS [15] bag files recorded by a mobile computing platform developed for easy handheld operation and easy mounting on mobile robots: the Handheld Central Rover Unit (HCRU). Those files contain a stream of rectified stereo images showing a stationary object from multiple viewpoints on a circular trajectory around it. Those images, extracted in PNG format and subsampled at one image per second, are given as input to the 3D reconstruction DFPC. Figure 1 shows the object in a sample input image, and Figure 2 shows the reconstructed 3D point cloud after ten image pairs have been processed by the SparseRegistrationFromStereo DFPC.

Figure 3 shows a view of the reconstructed point cloud in CDFF-Dev's 3D visualizer. Using CDFF-Dev's tools, the logged data can be replayed, the relevant data given as input to the DFPC, and the resulting data fusion product displayed in an interactive 3D visualization software as the data is replayed. In addition to point clouds, the visualizer tool can also display coordinate frames (two frames, connected by a red line, are visible in Figure 3), trajectories, maps, and meshes loaded from URDF. This data fusion product visualizer, whose development is ongoing, will be very helpful for developing and examining the results of data fusion solutions (DFPCs).

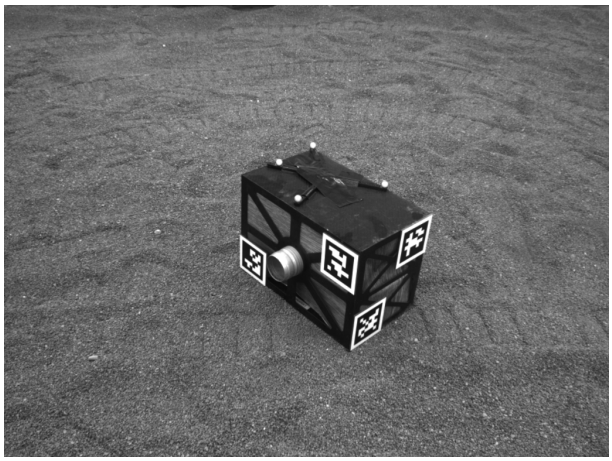

Figure 1: Sample input image from a stereo pair

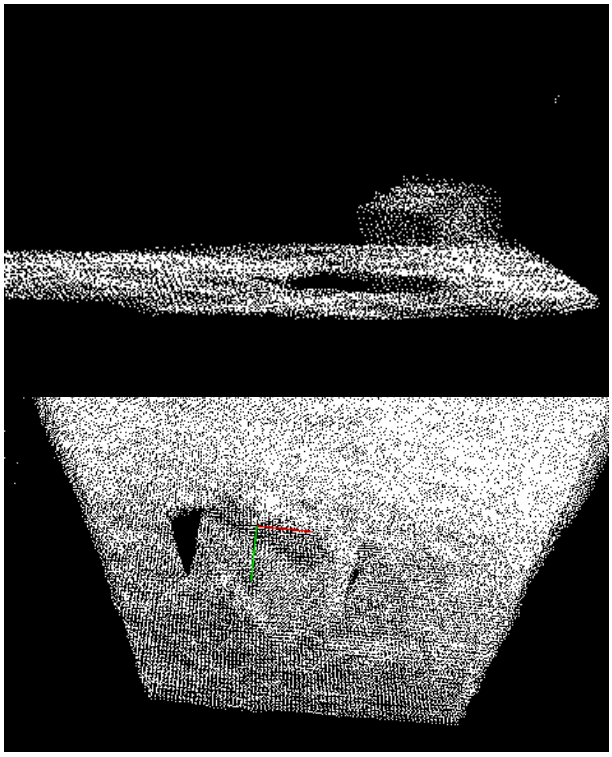

Figure 2: Two views of the point cloud reconstructed from ten image pairs

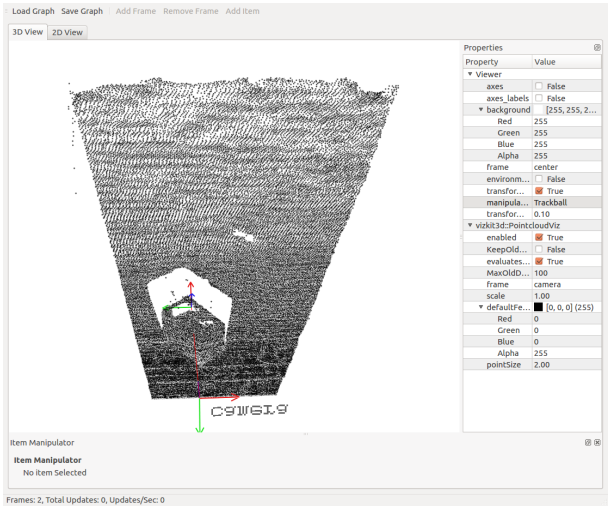

Figure 3: View of the reconstructed point cloud in CDFF-Dev's log replay tool

Choosing the algorithm parameters that lead to the best reconstruction result is difficult, because so many DFNs, so many parameters. A challenge we are currently facing is that the complete point cloud is correctly reconstructed only while the error is relatively small. After a certain number of frames, large position errors may break the reconstruction. To prevent this accumulation of error, we are developing a DFN that can combine successive point clouds at a desired resolution while limiting error accumulation through statistical filtering. To date, the best performing DFPCs are SparseRegistrationFromStereo with PCL's ICP, and ReconstructionFromStereo. Dense- and Sparse-RegistrationFromStereo produce similar results, with Dense taking longer. RegistrationFromStereo results in a much larger error, and when using CloudCompare in place of ICP as a registration DFN, a larger error is observed as well. Estimation- and Adjustment-FromStereo have frequently shown poor results too, as bundle adjustment fails to converge in the presence of many outliers. One cause of this may be the small number of poses over which the optimization is run, but this is necessary for robotic navigation in many situations. Failed reconstructions cause the algorithm to skip steps, and to complete faster. Hence, some time measurements on full reconstructions are not representative of a complete functional application. Nevertheless, Table 4 gives an estimate of processing times for the environmental reconstruction DFPCs.

Table 4: Observed processing time of 5 image pairs by the six DFPC implementations. Times are measured on an Ubuntu 16.04 LTS platform running on an Intel i7-3770 CPU, with 8GB of memory.

Implementation	Time to process 5	Quality of the reconstruction

	image pairs	
RegistrationFromStereo	117 s	Bad
SparseRegistrationFromStereo (with PCL ICP)	8.96 s	Good
DenseRegistrationFromStereo (with PCL ICP)	7.27 s	Good
SparseRegistrationFromStereo (with CloudCompare ICP)	10.7 s	Poor
DenseRegistrationFromStereo (with CloudCompare ICP)	42.9 s	Bad
ReconstructionFromStereo	5.30 s	Poor
EstimationFromStereo	6.48 s	Bad
AdjustmentFromStereo	6.63 s	Bad

Environmental Mapping

For environment mapping, the main approach that is used in the CDFF is Digital Elevation Map (DEM) built from sensed or reconstructed point clouds. A DEM is a cartesian grid where each cell represents a spatial area perceived by the rover, and contains the mean elevation given out by its sensors. The process for building such a DEM is the following: (i) the input point cloud is first transformed into the world's frame of reference, (ii) then projected on a cartesian grid of predefined dimensions and scale (this is called the local map), (iii) and finally this cartesian grid is fused into the internal model containing previous observations of the world by the rover. As the DFNs used in this case are very simple, only one algorithm implementation has been proposed for each of the DFNs. Table 5 lists the basic DFNs for building DEMs.

Table 5: Summary of the different DFNs used in the DEM building DFPC.

DEM Buiding DFNs
PointcloudTransform
Rasterization (i.e. projection of the point cloud)
Fusion

As for 3D and 2D reconstruction, each DFN has been tested on recorded data (in the form of ROS bag files) recorded on the Mana LAAS rover at the Centre National d'Études Spatiales (CNES) in Toulouse, France, in the test area shown in Figure 4. The bags contain a stream of robot poses (needed for the point cloud transform) as well as acquired point cloud data from a velodyne sensor.

Figure 4: Aerial view of the field used for DEM building with the CDFF

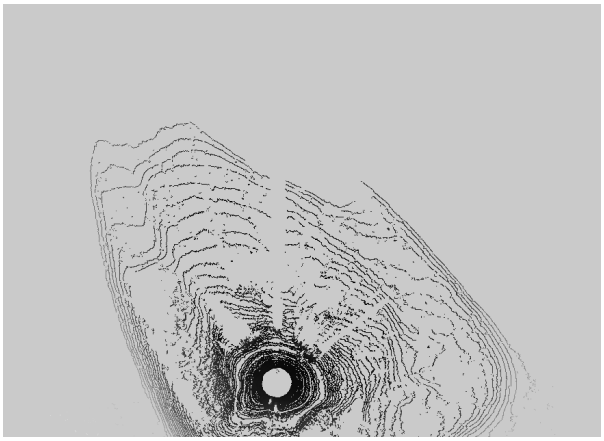

Figure 5: An example of a local map of the field shown in Figure 4 during the rover's run

Using the DFPC for producing a DEM, Figure 5 shows a DEM produced using this function of the CDFF. Finally, Figure 6 shows an example of the fused map over several observations of the environment by the rover: the gradient in grey correspond to the gradient in elevation.

The results to date, while convincing of function, are not ideal, as what is supposed to be a straight wall can appear several time in different orientations. This comes from the fact that DEM building can only be as good as the quality of localization of the rover. If a mismatch happens between the point cloud and the pose at which it was acquired, the resulting DEM will be corrupted. A

small error in orientation can produce remarkable errors in the DEM if the obstacles are at good distance to the rover. Future DFPCs will make use of improved multi-sensor localization to prevent DEM corruption.

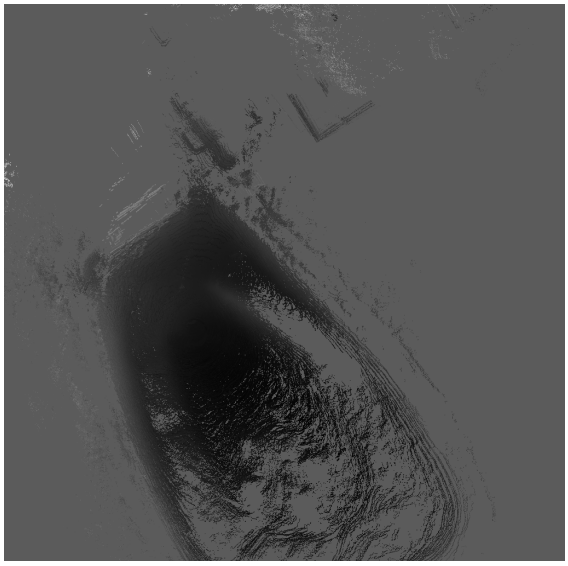

Figure 6: The fused map shown in Figure 5 using multiple observations in the form of a gradient map

Evaluation of CDFF Outputs

A large amount of data is produced in the many processes contained in the CDFF, especially taking into account the number of DFNs that could be producing data. It is desirable to be able to assess the quality of this data to enable a continuous evaluation of DFNs and DFPCs. We define a set of data quality assessment functions to be applied to the vision-based process pipeline composed by extraction and detection, matching, triangulation and pose estimation, which is used in several DFPCs. Four DFNs used in this process are considered, applied in order for vision processing with their functions clearly indicated by their names:

1. FeatureExtraction2D
2. FeatureMatching2D
3. Reconstruction2DTo3D
4. PerspectiveNPoint

A set of indicators intrinsically present in the data produced by DFNs is used for this scope. The definition of data quality assessment methods, such as visual feature matchability, enables a continuous evaluation of these algorithms [16].

In feature extraction the features are ranked by their response value. It is desirable to extract as many matchable features as possible, in fact any unmatched image feature is discarded in successive steps. We compared the response of features with respect to the features that were matched in the next DFN. This way one could obtain a preliminary idea of the goodness of a feature set and eventually select a subset to feed to the matcher (or even re-extract). While, some patterns were found, it would be statistically incorrect claim a general link between response and matchability. A high response does not imply repeatability [16]. Further in the processing pipeline, we assessed how matching distance could be used for similar goals. Using PnP estimation incorporated in a RANSAC [17] scheme benefits of large sets of matches (which directly generate 3D points). However, the set of matches surviving the filtering process has to be accurate or will produce too many outliers to handle by RANSAC. We define a data quality assessment function based on the ratio between the sum of distances of accepted matches (i.e. good matches) and the average distance of all the matches. This function (to be ideally minimized) favors large set of accepted matches, still penalizing large average distances. It proved useful as a predictor of poor matching performances. Concerning triangulation, it is possible to propagate the 2D point extraction uncertainty through the triangulation process. This yields the 3D point covariance which directly represents the accuracy of the point cloud. The work from Beder and Steffen [18] shows how to estimate the covariance matrix of a 3D point in Euclidean space and proposes a scalar measure based on the matrix singular values to evaluate it. To predict poor estimation by PnP we used RANSAC inlier percentage levels, which are a direct measure of when the pose estimation cannot be trusted. Low level inliers, even dropping to zero in some cases, always resulted in a completely off estimation. Having a predictor of reasonable motion estimation is crucial in pose estimation for dead reckoning processes such as visual odometry, where a single large error can compromise the localization accuracy.

Table 6: figures of merit used within visual processing DFNs within the CDFF

DFN	Output type	Figure of merit
FeatureExtraction2D	Visual Features	mean response
FeatureMatching2D	2D Point Matches	$\frac{\sum_{m=0}^{M} d(m)}{\sum_{n=0}^{N} \frac{d(n)}{N}}$
Reconstruction2DTo3D	3D Point Cloud	mean roundness, depth axis standard deviation
PerspectiveNPoint	Pose	percentage of inliers

(+RANSAC)	Estimation (w/inliers)	

In summary, we have defined figures of merit for the output of four DFNs used in vision processing within the CDFF. While evaluating a set of visual features remains a difficult task without resorting to machine learning methods, other figures of merit proved helpful in predicting poor data outputs, which often coincide with sub-optimal performances by DFNs. Table 6 summarizes the described figures of merit, where M is the number of accepted matches, N is the number of total matches and $d(x)$ represents the computed matching distance for match x.

Outlier Detection Capabilities

Outlier or anomaly detection is a field of machine learning. During the training process usually only unlabeled data is given. Most of the data points are inliers, some of them could be outliers. The model should learn what is normal (inliers) and distinguish anomalies (outliers). In the context of space robotics, outlier detection can be used to detect dangerous or unexpected situations. This information can be used to go to a safe state in critical situations.

We present a use case with the robot Asguard IV[1], the fourth generation of the robot Asguard [19]. Based on acceleration measurements of an IMU it will detect unexpected situations during locomotion, for example, slipping, skid, or tilting. The dataset has been recorded in an artificial space environment crater at a movement speed of about 0.15 meters per second.

The data is preprocessed before given to the outlier detection algorithm. We compute the variance in each dimension over a sliding window of 200 samples and it as a feature vector. A preliminary comparison of several algorithms has been done with pySPACE [20], which is a Python framework that wraps a lot of classification and outlier detection algorithms. Methods like Rapid Outlier Detection [21], Isolation Forests [22], One-class SVM, or standard methods like the Mahalanobis distance, Gaussian Mixture Models, or k-Means [23] have been compared. The original dataset has been split into a training set (60%) and a validation set (40%) and the data has been annotated for this analysis. For comparison reasons, a threshold optimization has been applied so that every algorithm marks about 2-3% of the validation set as outliers. As performance metric we use the average of outlier recall and precision because of the high number of inliers. Results of this preliminary evaluation can be found in Table 7. Training supervised, that is, giving only inliers to the outlier detection algorithms in the training phase, generally yields better results. The overall performance, however, is low because the recall is usually low, that is, not all outliers are detected. Most algorithms have a threshold that can be used to increase the amount of detected outliers with the drawback that the precision decreases, that is, the number of false positives increases. To obtain better outlier detection results, more data would have to be acquired.

The Gaussian Mixture Model, Mahalanobis distance, and k-Means algorithms are implemented as data fusion nodes so that models trained in pySPACE can be easily exported to the CDFF.

Table 7: Comparison of several outlier detection algorithms.

Implementation	Parameters	Performance [%]
Gaussian Mixture Model	trained only on inliers, 3 Gaussians	55.3
One-Class SVM	trained only on inliers, kernel: RBF, nu: 0.15, gamma: 0.1	52.4
Rapid Outlier Detection	trained only on inliers	54.7
Rapid Outlier Detection		52.2
k-Means	trained only on inliers	50.1
Isolation Forest	trained only on inliers	54.2
Isolation Forest		50.2
Mahalanobis Distance	trained only on inliers	50.9

Application of InFuse to Visual Tracking

The CDFF also includes a visual tracking DFPC which exploits a geometric model of an object to align the image edges, consequently enabling to estimate absolute pose. This DFPC is composed of several DFNs such as Canny edge detection, Image Filtering, Image gradient computation, Kalman prediction and Kalman correction. The parameters and a geometric model of the visual tracking DFPC can be configured and set in text file, which are in turn used for configuration its DFNs internally.

[1] https://robotik.dfki-bremen.de/en/research/robot-systems/asguard-iv.html

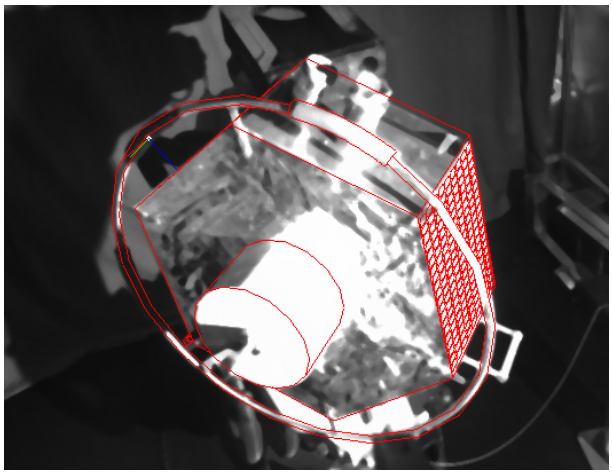

Figure 7: Visualization of the Visual tracking DFPC on DLR OOS-sim data: the alignment of model edges (in red) onto image edges indicate a correct pose estimation and tracking.

We demonstrate here a sample pose (image shown in Figure 7, taken from DLR OOS-sim) where a pose tracking is successful, indicated by the precise alignment of model contours onto the image at the estimated pose. The visual tracking DFPC is a typical InFuse application for an on-orbit satellite servicing. For an orbit-servicing, the target satellite needs to be tracked so that a servicer satellite or robot can autonomously replace parts or refuel it.

Application of InFuse to Rover Localisation in an Unstructured Environment

The InFuse CDFF already includes support for external libraries to provide visual localisation functions. Three main implementations are provided. Two visual odometry, one designed for research and education fully implemented in InFuse by CNRS/LAAS, one optimised for space exploration rovers provided by CNES EDRES SDK and integrated by Magellium, with an interface shown in Figure 8. The open source Stereo SLAM ORB-SLAM is also included for long range localisation.

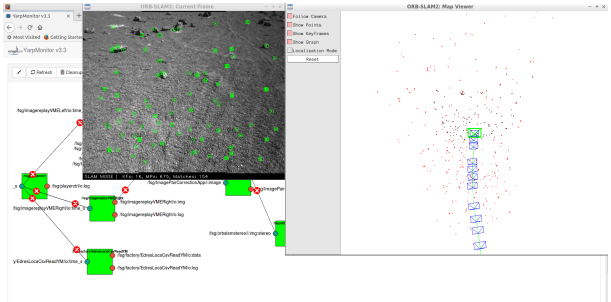

Figure 8: Interface for SLAM using EDRES SDK

Performances are evaluated using metrics described in [24] to provide a statistical analysis of algorithms performances. We are using two kinds of datasets for evaluation in the context planetary exploration, some of which are planned for release following the completion and public release of InFuse source code.

1) Datasets coming from outdoor experiments done on the CNES Mars yard (SEROM). The ground truth is provided by an RTK-GPS and a one axis fiber Gyro for the heading. Data acquisition was performed by CNRS/LAAS rovers Mana and Minnie shown in Figure 9. This setup allows to evaluate the algorithms performances on long straight trajectories of 70m in a Mars relevant environment.

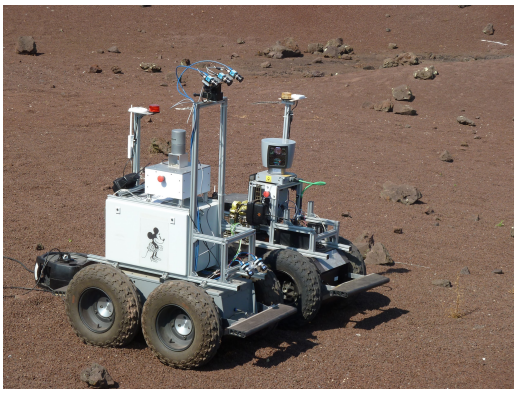

Figure 9: CNRS/LAAS rovers Mana (right) and Minnie (left) used for CDFF data acquisition.

2) Datasets coming from indoor experiments done in the DLR Planetary Exploration Lab (PEL). Data acquisition was performed by the ExoMars BB2 prototype rover shown in Figure 10 with the HCRU running the CDFF on board. The ground truth is provided by a tracking system for the full pose of sensors and a homemade scanner for the DEM, and the error of the most recent test is shown in Figure 11. This smaller setup provides high accuracy ground truth (<1mm in position and <1° in orientation for poses and less than 4mm accuracy in XYZ for the DEM) and has the ability to configure the slope intensity. It will enable the analysis of fusion algorithms when strong wheel slippage is encountered and that incoherent informations are then provided by localisation subsystems like wheel odometry and visual odometry.

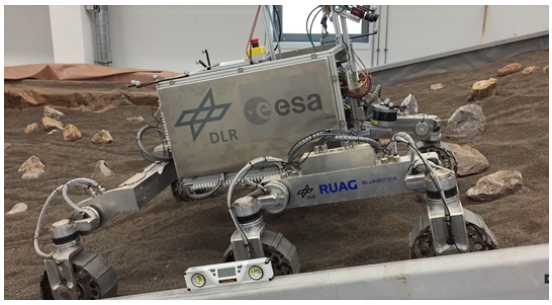

Figure 10: ExoMars BB2 Prototype Rover

First results obtained on a reference outdoor dataset (SEROM) are promising and inline with expected performances. The next step is to analyse results on all collected datasets to confirm this first result and to get a deeper analysis of algorithms robustness in the context of planetary exploration.

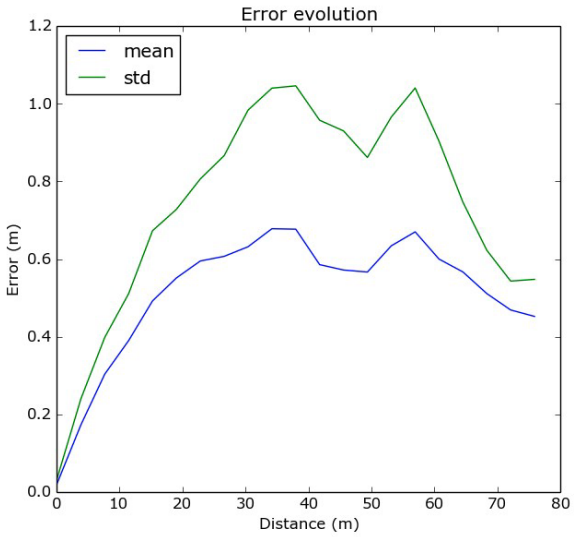

Figure 11: Localization error in unstructured environment obtained in SEROM testing

Conclusions

In this paper, we have presented the Common Data Fusion Framework (CDFF) created within the InFuse project. The CDFF provides the first modular open-source framework for fusion of robotic data using a wide variety of algorithms, and is specifically focused on providing data products for space robotics both in orbit and on other planets. Among other functions, the CDFF provides data fusion for environmental reconstruction from multiple sensors and views, map generation for navigational learning and reasoning, visual identification and tracking of objects, and localization in both structured and unstructured environments. Algorithms for machine learning and data analysis are also included for uses such as data quality assessment and outlier detection, and a product visualizer with log replay is included within the suite of development tools to facilitate analysis and debugging of implementations. Work on the CDFF is ongoing and a public release of InFuse source code and data is planned for January 2019.

Acknowledgements

We thank Dennis Hemker for curating the dataset that is recorded on Asguard IV and used for outlier detection, and Roberto Lampariello for providing data collection from the DLR OOS-SIM orbital simulator.

The InFuse project is funded under the European Commission's Horizon 2020 Space Strategic Research Cluster Operational Grants, grant number 730014.

References

[1] S. Govindaraj, J. Gancet, M. Post, R. Dominguez, S. Lacroix, M. Smisek, J. Hidalgo-Carrio, B. Wehbe, A. Fabisch, A. De Maio, N. Oumer, V. Bissonnette, Z. -C., Marton, S. Kottath, C. Nissler, X.-T, Yan. "InFuse: a Comprehensive Framework for Data Fusion in Space Robotics", 14th ESA Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), Netherlands, June 2017.

[2] R. Dominguez, S. Govindaraj, J. Gancet, M. Post, R. Michalec, N. W. Oumer, B. Wehbe, A. Bianco, A. Fabisch, S. Lacroix, A. De Maio, Q. Labourey, F. Souvannavong, V. Bissonnette, M. Smisek, X. Yan. "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.

[3] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 30(2):328–341, 2008.

[4] L.Wang, M. Liao, M. Gong, R. Yang, D. Nister. "High-Quality Real-Time Stereo Using Adaptive Cost Aggregation and Dynamic Programming". Third International Symposium on 3D Data Processing, Visualization and Transmission, 2006.

[5] I. Sipiran, and B. Bustos. “Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes”. The Visual Computer, 2011.

[6] F. Tombari, S. Salti, and L. Di Stefano. “Unique Signatures of Histograms for Local Surface Description”. European Conference on Computer Vision. ECCV 2010.

[7] D. Holz, A.E. Ichim, F. Tombari, R.B. Rusu, S. Behnke. “Registration with Point Cloud Library”. IEEE Robotics and Automation Magazine, Volume 22, Issue 4, pp. 110-124, December 2015.

[8] C. Harris, M. Stephens. and A Combined Corner and Edge Detector. In Proc. of Fourth Alvey Vision Conference, 1988.

[9] E. Rublee, V. Rabaud, K. Konolige, G. Bradski. “ORB: An efficient alternative to SIFT or SURF”, International Conference on Computer Vision, 2011.

[10] R. Hartley, A.Zisserman. “Multiple View Geometry in Computer Vision”. Second Edition. Book. 2014. Cambridge University Press.

[11] V. Lepetit, M. Moreon-Noguer, P. Fua. “Accurate O(n) Solution to the PnP Problem”, International Journal of Computer Vision. 2009.

[12] A. Aldoma, Z.C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R.B. Rusu, S. Gedikli, and M. Vincze. “Tutorial: Point Cloud Library. Three-Dimensional Object Recogniton and 6 DoF Pose Estimation”. IEEE Robotics and Automation Magazine, Volume 19, Issue 2, pp. 80-91, September 2012.

[13] CloudCompare Official Website: http://www.cloudcompare.org/release/notes/20171026/

[14] S. Agarwal, K. Mierle and Others, "Ceres Solver", "\url{http://ceres-solver.org}”.

[15] L. Zhang, R. Merrifield, A. Deguet, and G.Z. Yang. “Powering the world’s robots—10 years of ROS”. Science Robotics, 2017.

[16] Hartmann, Wilfried, Michal Havlena, and Konrad Schindler. "Predicting matchability." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014.

[17] M.A. Fischler and R.C. Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. Communication of the ACM, Volume 24, Issue 6, June 1981.

[18] Beder, C. and Steffen, R. (2006). Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence. In Joint Pattern Recognition Symposium, pages 657–666. Springer.

[19] M. Eich, F. Grimminger, S. Bosse, D. Spenneberg, F. Kirchner. “ASGUARD: A Hybrid Legged Wheel Security and SAR-Robot Using Bio-Inspired Locomotion for Rough Terrain”, IARP/EURON Workshop on Robotics for Risky Interventions and Enviromental Surveillance (IARP/EURON-08), January 7-8, Benicassim, Spain, (IARP/EURON), Benicàssim, Online-Proceedings, Jan/2008.

[20] M. M. Krell, S. Straube, A. Seeland, H. Wöhrle, J. Teiwes, J.H. Metzen, E. A. Kirchner, F. Kirchner. “pySPACE — a signal processing and classification environment in Python”, Frontiers in Neuroinformatics 7(40), 2013, doi: 10.3389/fninf.2013.00040.

[21] M. Sugiyama, K. Borgwardt. “Rapid Distance-Based Outlier Detection via Sampling”, Advances in Neural Information Processing Systems 26, pages 467 - 475, 2013, Curran Associates, Inc.

[22] F. T. Liu, K. M. Ting, Z.-H. Zhou. “Isolation forest.” Data Mining, 2008. ICDM‘08. Eighth IEEE International Conference on.

[23] C. Bishop. “Pattern Recognition and Machine Learning”, Springer, 2011.

[24] R. Kuemmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner. On measuring the accuracy of SLAM algorithms. Auton. Robots, 27:387–407, 2009