

# Gaussian Process Estimation of Odometry Errors for Localization and Mapping

Javier Hidalgo-Carrió, Daniel Hennes, Jakob Schwendner and Frank Kirchner<sup>1</sup>

**Abstract**—Since early in robotics the performance of odometry techniques has been of constant research for mobile robots. This is due to its direct influence on localization. The pose error grows unbounded in dead-reckoning systems and its uncertainty has negative impacts in localization and mapping (i.e. SLAM). The dead-reckoning performance in terms of residuals, i.e. the difference between the expected and the real pose state, is related to the statistical error or uncertainty in probabilistic motion models. A novel approach to model odometry errors using Gaussian processes (GPs) is presented. The methodology trains a GP on the residual between the non-linear parametric motion model and the ground truth training data. The result is a GP over odometry residuals which provides an expected value and its uncertainty in order to enhance the belief with respect to the parametric model. The localization and mapping benefits from a comprehensive GP-odometry residuals model. The approach is applied to a planetary rover in an unstructured environment. We show that our approach enhances visual SLAM by efficiently computing image frames and effectively distributing keyframes.

## I. INTRODUCTION

The use of proprioceptive and stereoceptive sensors to localize a robot and map its surrounding is a common practice in robotics. The technique is well established in robotics and has special attention in GPS-denied environments as planetary rovers. Measuring and counting the joints displacement of a robot chassis gives an initial guess to derive the robot pose. Understanding the contribution of each contact point is of use in a dead-reckoning process [1], [2]. SLAM uses optimization methods in combination with landmarks in order to correct the robot state initially predicted by odometry. Three techniques are typically applied: filtering, fixed-lag smoothing and full smoothing [3]. They slightly diverse in the methodology but all use some degree of odometry in order to estimate at initial guess of the robot pose. The importance of having a good guess of the odometry estimation is necessary to establish the selection of keyframes. This can be a difficult task since wheel odometry performance is not deterministic due to its uncertainty from the interaction with the ground. Wheel odometry may perform well (i.e. 1.2 % error per distance traveled) in flat terrain but it can rapidly degrade in slippery terrains.

This work explains the methodology of training a Gaussian process on the odometry residuals and the application to robot localization and mapping. A Gaussian process on the residuals estimates the quality of wheel odometry. The

All authors are with the DFKI - Robotics Innovation Center, Robert-Hooke-Str. 1, 28359 Bremen, Germany {name}.{surname}@dfki.de  
<sup>1</sup>Department of Mathematics and Informatics, University of Bremen, 28359 Bremen, Germany

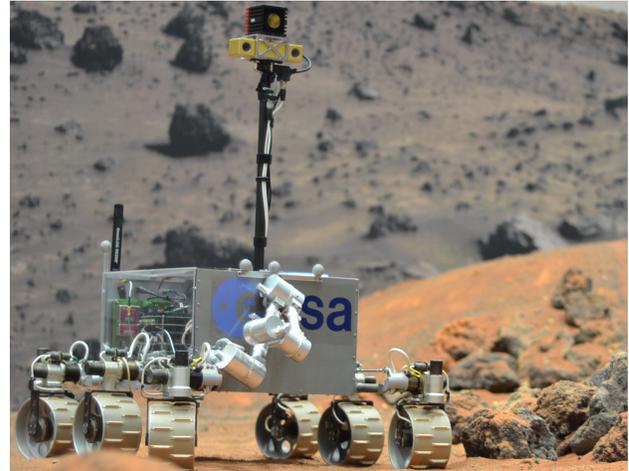


Fig. 1: The ExoMars Test Rover (ExoTeR) during the test at the Planetary Robotics laboratory at ESA.

difference between the parametric model and ground truth serves to train a Gaussian process regression model. The model is used in a state-of-the-art SLAM approach and applied in a challenging environment. The main contributions of this study are:

- To train a Gaussian process on odometry residuals to model poor traction performance.
- To assess the kernel for the Gaussian process regression.
- To efficiently query the computation of image features and effectively select keyframes based on traction performance.

First, we present a review of related work in the context of odometry, localization and mapping and Gaussian processes applied in robotics. Section III describes the parametric model for computing a fully 3d-odometry model. The Gaussian process is explained in Section IV. Afterward, prediction quality of the learned process is presented in Section V in order to address the feasibility of the approach. The Gaussian process is used in a visual simultaneous localization and mapping (e.g. ORB-SLAM [4]) to estimate the robot pose. Experiments with a planetary rover in a representative environment are presented. Section VI includes a conclusion and a final discussion for future work.

## II. RELATED WORK

The simplest manner to localize a robot is wheel odometry which involves the calculation of the robot's body displacement from encoders readings. Odometry is commonly enhanced with inertial sensors to estimate the robot's attitude

which the heading is the less observable angle. Odometry and inertial sensors are combined together in a dead-reckoning process to integrate accumulative displacements as an initial guess of the robot's pose. The estimation is perturbed by systematic [5], [6] or non-systematic errors due to poor traction performance [7]. Systematic errors are well characterized in the literature [8], [9]. However, non-systematic errors are complex, difficult to predict and not possible to fully correct unless other perception means are present. Visual odometry [10] has gained robustness in the last decade with excellent results.

The error propagates unbounded unless a SLAM back-end is available to close the loop by revisiting the place. SLAM was initially developed using filtering techniques [11], [12] while full-smoothing methods [13], [14] guarantee highest accuracy. However, real-time computation become infeasible since the map and associated graph grow over time. Carlone et al [15] defines a set of target variables to deal with smaller graphs. The solution enhances computational efficiency and robustness in the back-end. The relation with the perception front-end and robot navigation demands is still desirable in robotics.

The combination of visual and wheel odometry methods are of interest to provide informative inputs to the path planning and assess the entrapment risk. Angelova et al [16] present a learning method to predict slippage based on camera images and an adaptive mixture of local experts. Fuzzy-logic is applied in [17] to provide a motor current-compensated odometry. The results keep the error to 2% of the total travel distance without using a SLAM solution. Gaussian processes provide a probabilistic approach of learning kernel machines with promising results to probabilistic robotics. Ko et al [18] apply a Gaussian process to learn the residuals of the dynamic model of a robotic blimp. The work was afterward applied to dynamic state estimation and control the blimp with an Unscented Kalman Filter (GP-UKF). In general, Gaussian processes have several advantages for robotics since they are a practical tool for solving a diverse set of problems as motion planning [19] and occupancy maps [20].

Though Gaussian processes have been used for non-linear regression, to the best of our knowledge, its application to model the odometry error as residual between the parametric model and a realistic odometry output has not yet been addressed in the literature.

### III. PARAMETRIC 3D-ODOMETRY MODEL

Motion models together with attitude kinematics from Inertial Navigation Systems (INS) is the traditional approach for conventional vehicle odometry. We use an Enhanced 3d-odometry to fully model the chassis kinematics of outdoor robots. The model is described in [2] and requires the definition of points in contact with the terrain and its relative angle. One single contact angle is necessary for wheeled mobile robots, which is perpendicular to the direction of motion (i.e. y-axis). A transformation matrix computes the

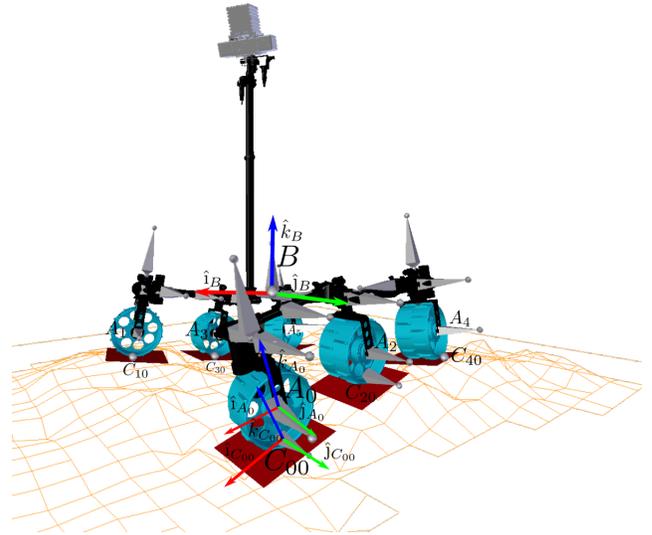


Fig. 2: Illustration of the ExoTeR rover kinematics model.  $B$  is the body frame located at chassis's geometric center,  $A_i$  is the wheel frame and  $C_{il}$  the contact point frame. 3d-deltas-poses are computed using a weighted solution (5) among the contact-points.

localization of each contact point frame with respect to the body frame.

Mobile robots are commanded by target velocities. Velocity kinematics is deduced by derivation of the transformation matrix. The transformation of the rover body at time step  $k-1$  ( $\bar{B}$ ) to the rover body at time step  $k$  ( $B$ ) is defined as  $T_{\bar{B},B} = T_{\bar{B},\bar{C}_{il}} T_{\bar{C}_{il},C_{il}} T_{C_{il},B}$ . The mapping between the body frame Cartesian rate vector  $\dot{\mathbf{u}} = [\dot{x}_B \ \dot{y}_B \ \dot{z}_B \ \dot{\phi}_B \ \dot{\theta}_B \ \dot{\psi}_B]$  and the joint space rate vector with the contact rate angle and slip rate vector is solved by a Jacobian matrix. The resulting Jacobian matrix  $J_{il}$  related to the contact point  $il$  has the form:

$$[\dot{x}_B \ \dot{y}_B \ \dot{z}_B \ \dot{\phi}_B \ \dot{\theta}_B \ \dot{\psi}_B]^T = J_{il} [\dot{\mathbf{q}} \ \boldsymbol{\varepsilon}_{il} \ \dot{\boldsymbol{\delta}}_{il}]^T \quad (1)$$

Equation (1) defines the contribution of each kinematic chain to the resulting body motion in  $\dot{\mathbf{u}}$ . The  $J_{il}$  matrix is of size  $6 \times (n+4)$  where  $n$  corresponds to the DoF of the mechanical chassis. The composite rover equations are obtained by combining the Jacobian matrices for all kinematics chains into a sparse matrix equation of appropriate dimensions where  $i$  is the number of wheels (e.g.  $i=0, \dots, 5$ ) and  $l$  corresponds to contact points per each wheel (e.g.  $l=0$ ).

$$\begin{bmatrix} I_{6 \times 6} \\ I_{6 \times 6} \\ \vdots \\ I_{6 \times 6} \end{bmatrix} \begin{bmatrix} \dot{x}_B \\ \dot{y}_B \\ \dot{z}_B \\ \dot{\phi}_B \\ \dot{\theta}_B \\ \dot{\psi}_B \end{bmatrix} = J \begin{bmatrix} \dot{\mathbf{q}} \\ \boldsymbol{\varepsilon} \\ \dot{\boldsymbol{\delta}} \end{bmatrix} \equiv S\dot{\mathbf{u}} = J\dot{\mathbf{p}} \quad (2)$$

The kinematics relates the rover pose rate to joints and inertial sensor rate quantities. Equation (2) defines the para-

metric motion model. Robot's sensor availability determines sensed and non-sensed quantities and (2) separates into the following form:

$$\begin{bmatrix} S_s & S_n \end{bmatrix} \begin{bmatrix} \dot{\mathbf{u}}_s \\ \dot{\mathbf{u}}_n \end{bmatrix} = \begin{bmatrix} J_s & J_n \end{bmatrix} \begin{bmatrix} \dot{\mathbf{p}}_s \\ \dot{\mathbf{p}}_n \end{bmatrix} \quad (3)$$

Rearranging into non-sensed (left-side) and sensed (right-side) quantities, the resulting equation is obtained:

$$\begin{bmatrix} S_n & -J_n \end{bmatrix} \begin{bmatrix} \dot{\mathbf{u}}_n \\ \dot{\mathbf{p}}_n \end{bmatrix} = \begin{bmatrix} -S_s & J_s \end{bmatrix} \begin{bmatrix} \dot{\mathbf{u}}_s \\ \dot{\mathbf{p}}_s \end{bmatrix} \equiv \Omega \mathbf{v} = \mathbf{b} \quad (4)$$

where  $\Omega$  is the matrix which dimensions depend on the sensing capabilities of the rover and directly influence the existence of a solution. The solution to the overdetermined system above is based on minimizing the error vector  $E = \mathbf{e}^T C \mathbf{e}$ , where  $C$  encodes the individual contribution of each kinematics chain to the estimated solution:

$$E = \mathbf{e}^T C \mathbf{e} = (\mathbf{b} - \Omega \mathbf{v})^T C (\mathbf{b} - \Omega \mathbf{v}) \quad (5)$$

#### IV. GAUSSIAN PROCESSES FOR 3D-ODOMETRY

The correctness of odometry is highly influenced by the amount of wheel slippage, which in turn depends on the maximum usable tractive force between ground and the wheel. This Section describes the application of Gaussian processes to model a non-linear regression function between the parametric model and the real odometry output.

##### A. Gaussian Processes

Gaussian processes (GPs) are a powerful, non-parametric tool for learning regression functions from sample data. GPs are flexible, work nice with poor and noisy data and therefore are very practical to solve a real-world scenario. A Gaussian process is a probability distribution over functions. Think a Gaussian process as a Gaussian distributing over an infinite long vector of data. However, an infinite vector is impractical and the marginalization function allows to work with a finite subset without losing generality.

Assume we have a training set of data,  $D = \langle X, \mathbf{y} \rangle$ , where  $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$  is a matrix containing d-dimensional input examples  $\mathbf{x}_i$  and  $\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]$  is a matrix containing o-dimensional training set  $\mathbf{y}_i$  (i.e. multioutput GP). The GP assumes that data is illustrated with a noisy function such as:

$$\mathbf{y}_i = f \mathbf{x}_i + \boldsymbol{\varepsilon} \quad (6)$$

where  $\boldsymbol{\varepsilon}$  is a zero-mean Gaussian noise with variance  $\sigma^2$ , i.e.  $\mathcal{N}(0, \sigma^2)$ . The prediction over the noisy output  $\mathbf{y}$  is a multivariable Gaussian of the input matrix  $X$ .

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | 0, K(X, X) + \Sigma) \quad (7)$$

where  $K \equiv K(X, X)$  is the kernel matrix with elements  $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$  defined by the kernel function  $k$  and  $\Sigma = \text{diag}(\sigma_1^2 I, \dots, \sigma_n^2 I)$ . We select a training set of data  $\mathbf{x}_*$  and the GP defines a predictive distribution over the output  $\mathbf{y}$  with mean

$$GP_\mu(\mathbf{x}_*, D) = k_*^T [K + \Sigma]^{-1} \mathbf{y} \quad (8)$$

and variance

$$GP_\Sigma(\mathbf{x}_*, D) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T [K + \Sigma] \mathbf{k}_* \quad (9)$$

$\mathbf{k}_*$  is the vector defined by the kernel values between  $\mathbf{x}_*$  and the training input  $X$  and  $K$  is the  $n \times n$  kernel matrix of the training input values. The prediction uncertainty, captured by the variance  $GP_\Sigma$ , depends on the process noise and the correlation between the test input and the training data. Covariance function are semi-positive defined functions where all the modeling occurs. The covariance function has a set of free parameters  $\Theta$  and the learning process optimizes the values given a training set of data. The most widely used kernel function is the squared exponential

$$k_{rbf}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 e^{-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T W (\mathbf{x} - \mathbf{x}')} \quad (10)$$

and hyperparameters  $\Theta = [W, \sigma_f^2, \Sigma]$  characterize the kernel function (10) and the process noise (7).

##### B. GP Modeling of Discrete Time Dynamic Processes

A discrete-time dynamic process can be understood as a series of states at a certain time-stamp which evolve over time as:

$$\mathbf{s}(k+1) = \mathbf{s}(k) + g(\mathbf{s}(k), \dot{\mathbf{u}}(k)) \quad (11)$$

where  $k$  is the time index and  $g$  is the function which described the dynamics of the system (e.g. rover pose rates) given a certain state  $s$  and the input vector  $\dot{\mathbf{u}} = [\dot{\mathbf{u}}_s, \dot{\mathbf{p}}_s]$ . A Gaussian process can be used to learn the dynamic process described by the function  $g$ . The result will be a GP which predicts the delta between two consecutively states  $\mathbf{y}_k = \mathbf{s}(k+1) - \mathbf{s}(k)$  given a vector of inputs. To perform such prediction the output for the parametric model should be part of the training data. This is because the Gaussian process assumes a zero-mean function (7) and robot odometry is clearly not a zero-mean. This is related to the modeling which appears to be in the covariance function. In this work we use the GP to learn the residual between the parametric model and the expected data. However, it would be also interesting to entirely model the wheel odometry using GPs.

##### C. Odometry Residuals from Gaussian Processes

Because the parametric 3d-odometry model gives reasonable good estimates under reliable ground-traction conditions. We model a zero-mean function over the odometry residual. The dynamic system equations for the GP are

$$\mathbf{s}(k+1) = \mathbf{s}(k) + g(\mathbf{s}(k), \dot{\mathbf{u}}(k)) + f(\mathbf{s}(k), \dot{\mathbf{u}}(k)) \quad (12)$$

where function  $f$  is modelled by the GP and describes the odometry residual and function  $g$  describes the change in state given by the parametric model. The training set  $D$  for the GP is a sequence of observed states and inputs. They are used to learn the parameters of the non-linear function

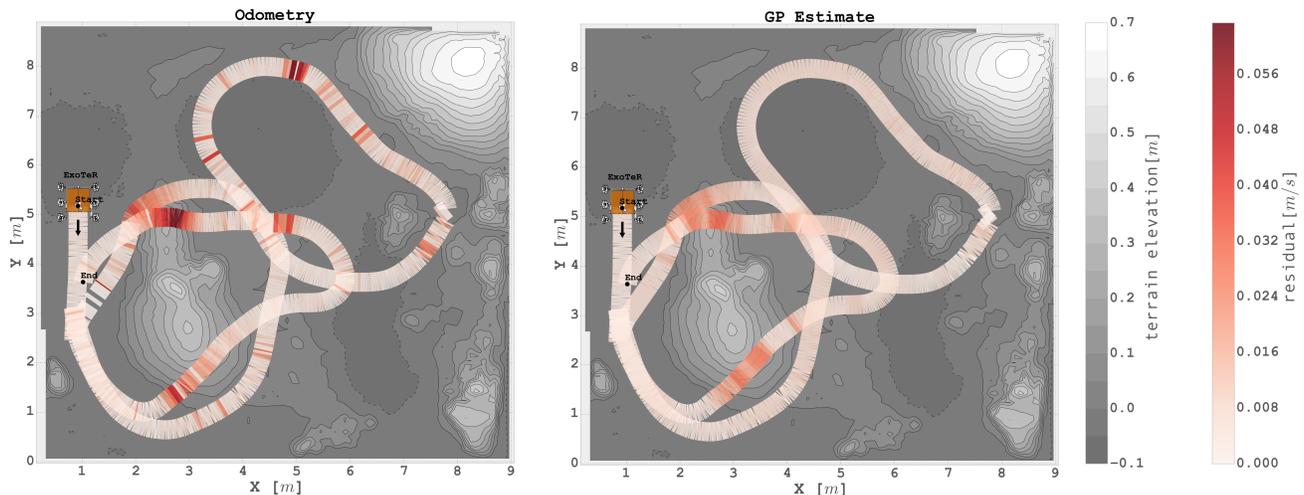


Fig. 3: Ground truth odometry residual and GP estimate for the test data. Traversed trajectory and Digital Elevation Map (DEM) of the Mars-like testbed is depicted together with the residual (red color-bar).

$f$ . The training data are of the form  $\mathbf{x}_k = [\mathbf{s}(k), \dot{\mathbf{u}}(k)]$  and the residual  $\mathbf{y}_k = \mathbf{s}(k+1) - \mathbf{s}(k) - g(\mathbf{s}(k), \dot{\mathbf{u}}(k))$

## V. RESULTS

The ExoMars Test Rover (ExoTeR), a laboratory rover prototype that resembles in scale the ExoMars rover mobility configuration [21], is the robotic platform for evaluating the methodology. ExoTeR’s sensor suite includes a stereo camera pair, an Inertial Measurement Unit (IMU) and actuator encoders and potentiometers. The experiments are performed on a Mars-like testbed in the Planetary Robotics Laboratory (PRL) of the European Space Research and Technology Centre (ESTEC) - the largest site of the European Space Agency (ESA) (see Fig. 1). The testbed comprises a  $9 \times 9$  m test area with different soil types and a set of 12 Vicon cameras for the ground truth measurements [22].

A set of reflective markers are located on the ExoTeR in order to track its position and orientation during the training experiments. The Vicon system captures the ground truth data. The accuracy of the system is around 1 cm in position and 0.2 degree in orientation depending on the number of cameras tracking the markers. ExoTeR is remotely driven in the testbed area describing all possible maneuvers (e.g. forward, backward, turn-on-spot, Ackerman) at widely different inclinations and terrain characteristics. The absolute position and heading are eliminated from the training data. The input vector is composed by pitch and roll orientation angles, joints position and speed, angular velocities sensed

by gyroscopes and linear accelerations from the IMU. The relative delta displacement (body linear velocities) is the output vector. Rover’s linear velocities are calculated by low-pass filtering the delta position readings from the Vicon system. The hyperparameters of the Gaussian process are optimized using conjugated gradient optimization. The GP estimates a three dimensional output for the residual in each direction of motion. A different set of collected data is used to evaluate the accuracy of the estimated residuals (i.e. test data).

The Gaussian process is trained off-line using GPy [23]. After the GP kernel is learnt, the prediction is made online on the rover based on sensed current inputs. The GP prediction and the localization and mapping are executed on-board the ExoTeR rover running the Rock real-time framework <sup>1</sup>.

### A. Comparing Prediction Quality

Three kernels, linear, squared exponential and Matern 5/2, are trained to determine which fits the regression more accurate. The kernels are separated and combined as  $\check{k}_{rbf} = k_{rbf} + k_{lin}$  where

$$k_{lin}(\mathbf{x}, \mathbf{x}') = \sigma_l^2 \mathbf{x} \cdot \mathbf{x}' \quad (13)$$

The kernels are fitted with normalized training data. This enforces features scaling, making gradient descent to converge more easily. Test data, not included in the training data, are used to verify the prediction quality of the Gaussian process odometry residuals. Rover’s body velocities are used to compare the GP non-linear regression model on the estimate. The squared exponential fits the residuals with lower root mean square error (RMSE) and mean absolute error (MAE). The lineal kernel does not improve the estimates. Table I shows the error metrics for each of the kernels.

The GP predicted residual is compared with the ground truth odometry residual computed from the Vicon system

TABLE I: Error per kernel evaluated with the test data.

Kernel	RMSE[m]	MAE[m]
rbf: $[k_{rbf}]$	<b>0.004494</b>	<b>0.002127</b>
rbf + linear: $[\check{k}_{rbf}]$	0.004909	0.002588
matern 5/2: $[k_{m52}]$	0.005562	0.002635
matern 5/2 + linear: $[\check{k}_{m52}]$	0.004909	0.002588

<sup>1</sup>The Robot Construction Kit (Rock) <http://www.rock-robotics.org>

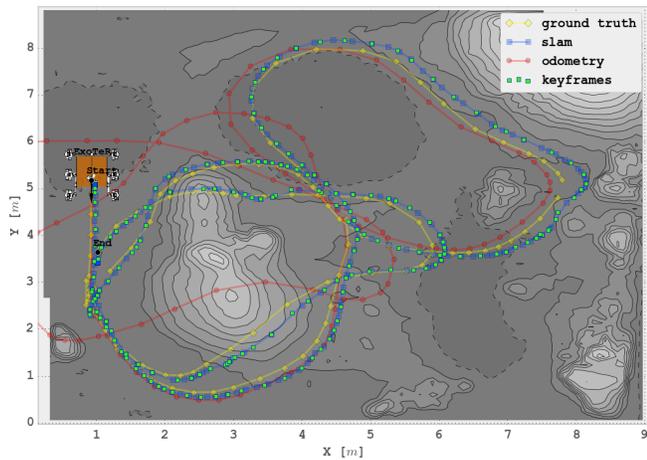


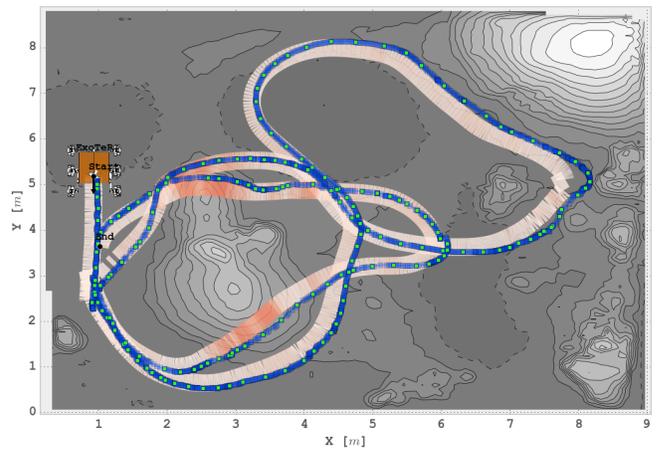
Fig. 4: ExoTeR test experiments and trajectories comparison.

along the track. Fig. 3 shows the GP estimate together with the elevation map and contours lines of the testbed. The GP-Odometry residual characterizes the most relevant zones of odometry poor performance. From the localization and mapping perspective, the information of poor traction and therefore odometry estimates, is more valuable than the absolute quantity. Therefore, such information is used to select desirable image frames in localization and mapping.

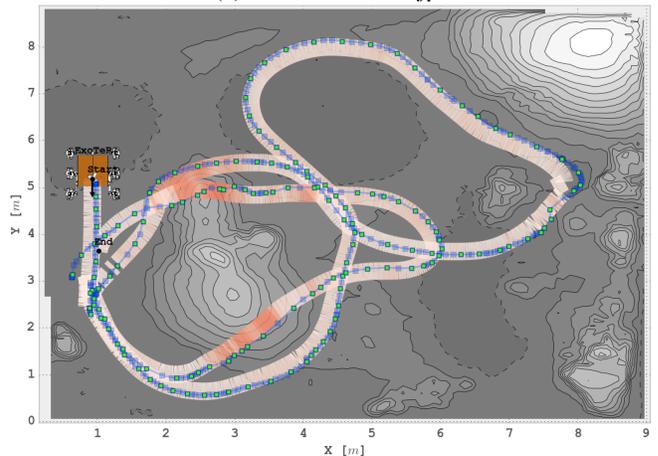
### B. Application to Localization and Mapping

Wheel odometry performs as good as visual odometry when residuals are small. Residuals give provide information when wheel odometry produces accurate estimates and when support by visual odometry is needed. The wheel odometry can equally support features tracking by providing an initial guess between two consecutive image frames (e.g. constant velocity model). Then, the loss of camera poses is reduced and relocalization is not often needed.

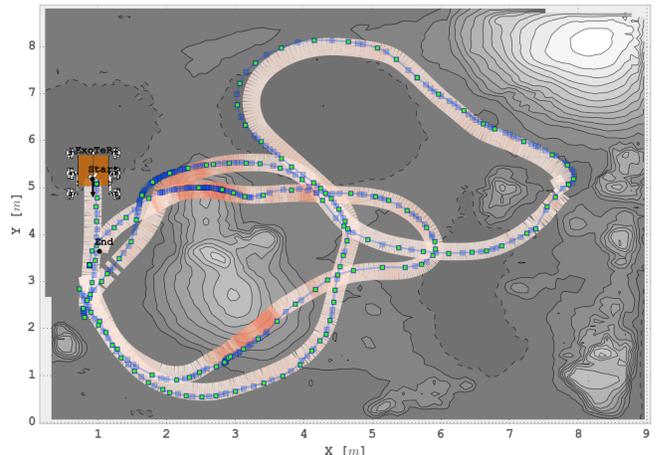
The information gained by the GP is used to inform the localization and mapping in order to compute visual odometry only when required. It furthermore reduces the keyframes in the optimization back-end. Our technique applies to any full SLAM approach. We perform experiments with ORB-SLAM [4] due to its robustness, accuracy and application of ORB binary features in Mars analog scenarios. Table II shows the results of calculating ExoTeR’s pose running three different SLAM schemes for the test data trajectory. *ORB-SLAM 2.5 fps* computes the localization and mapping by processing a new image frame at a rate of 2.5 fps. *ORB-SLAM with GP* is our approach using the GP estimate into the SLAM and providing an adaptive approach. *ORB-SLAM 0.5fps* computes the localization and mapping with lower frequency of 0.5 fps. Without the GP prediction the image frames are equally distributed along the trajectory. Fig. 5 depict the trajectory for the evaluated approaches. The number of frames can be seen along with the total traverse of 42.35 m. *ORB-SLAM with GP* adapts to the wheel odometry reducing the number of image frames in the visual odometry. The GP approach computes five times less number of frames



(a) ORB-SLAM 2.5fps



(b) ORB-SLAM 0.5fps



(c) ORB-SLAM with GP

Fig. 5: SLAM results: (a) shows the distribution of images and keyframes without GP, (b) still computes more images frames with worse performance than (c) which uses the GP odometry residual.

without a significant penalty in accuracy.

## VI. CONCLUSION

This work describes the first insight of using GP to model odometry residuals and the application in localization and mapping. The SLAM benefits from selecting desirable

TABLE II: ExoTeR’s pose results for the different SLAM schemes.

Scheme	#Frames	#Keyframes	RMSE [m]	Max E.[m]	Final E.[m]	Travel Distance E.[%]
<i>ORB-SLAM 2.5fps</i>	2582	182	0.144	<b>0.455</b>	<b>0.204</b>	<b>0.48</b>
<i>ORB-SLAM 0.5fps</i>	500	150	0.204	0.729	0.520	1.22
<i>ORB-SLAM with GP</i>	<b>484</b>	<b>135</b>	<b>0.145</b>	0.468	0.264	0.62

image frames in the frontend. The methodology has been demonstrated using ORB-SLAM in a planetary rover navigating an unstructured environment on loosely terrain. The GP information is used to selectively identify the frame among with to preintegrate the delta displacement given by the odometry and inertial sensors. It results in efficient selection of keyframes to be incorporated in a full SLAM problem.

We presented an offline learning approach, where the model is learned on training runs and evaluated in a test environment. The strategy might perform poorly in highly dynamic terrains where traction performance change significantly from one location to another. Future work includes on-line learning, using visual odometry as target inputs instead of ground truth data. Initially visual odometry will be queried frequently, with decreasing model uncertainty the frequency of visual odometry can be reduced and computational effort can be saved. Techniques such as Incremental Local Gaussian Regression [24] would allow to online learn and adapt the localization and mapping schema and the visual odometry to the dynamics of the environment. Odometry residuals might also provide a cue to identify the terramechanics of the terrain and inform the path planning component about potential hazards.

#### ACKNOWLEDGMENT

The authors would like to thank all members of the Planetary Robotics Lab at ESA, for their support to acquire ground truth measurements. The Entern project is funded by the Space Agency of the German Aerospace Center with federal funds of the Federal Ministry of Economics and Technology (BMWi) in accordance with the parliamentary resolution of the German Parliament, grant no. 50RA1406/07.

#### REFERENCES

- [1] J. Hidalgo-Carrio, J. Schwendner, and F. Kirchner, “Planetary Rover Localization Design: Antecedents and Directions.” in *IEEE Intelligent Vehicles Symposium Workshops*, 2012.
- [2] J. Hidalgo-Carrio, A. Babu, and F. Kirchner, “Static forces weighted Jacobian motion models for improved Odometry,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, sep 2014, pp. 169–175.
- [3] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “On-Manifold Preintegration Theory for Fast and Accurate Visual-Inertial Navigation,” p. 18, dec 2015.
- [4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: A Versatile and Accurate Monocular SLAM System,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, oct 2015.
- [5] P. Lamon and R. Siegwart, “3D Position Tracking in Challenging Terrain,” *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 167–186, feb 2007.
- [6] M. Tarokh and G. J. McDermott, “Kinematics Modeling and Analyses of Articulated Rovers,” *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 539–553, 2005.
- [7] D. Helmick, Y. Cheng, and S. Roumeliotis, “Path following using visual odometry for a Mars rover in high-slip environments,” *IEEE Aerospace Conference Proceedings*, vol. 2, pp. 772–789, 2004.
- [8] J. Borenstein and L. Feng, “Measurement and Correction of Systematic Odometry Errors in Mobile Robots,” *IEEE Transactions on Robotics*, vol. 12, pp. 869–880, 1996.
- [9] L. Ojeda and J. Borenstein, “Methods for the Reduction of Odometry Errors in Over-Constrained Mobile Robots,” *Autonomous Robots*, pp. 273–286, 2004.
- [10] D. Scaramuzza and F. Fraundorfer, “Visual Odometry [Tutorial],” *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, dec 2011.
- [11] Anastasios Mourikis and S. I. Roumeliotis, “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation,” Dept. of Computer Science and Engineering, University of Minnesota, Tech. Rep., 2006.
- [12] J. Civera, A. Davison, and J. Montiel, “Inverse Depth Parameterization for Monocular SLAM,” *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 932–945, oct 2008.
- [13] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, “A Tutorial on Graph-Based SLAM,” *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.
- [14] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “iSAM2: Incremental smoothing and mapping using the Bayes tree,” *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, dec 2011.
- [15] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, “Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014, pp. 4290–4297.
- [16] A. Angelova, L. Matthies, D. Helmick, G. Sibley, and P. Perona, “Learning to Predict Slip for Ground Robots,” in *IEEE International Conference on Robotics and Automation*, 2006, pp. 3324–3331.
- [17] L. Ojeda, D. Cruz, G. Reina, J. Borenstein, and S. Member, “Current-Based Slippage Detection and Odometry Correction for Mobile Robots and Planetary Rovers,” *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 366–378, 2006.
- [18] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 742–747.
- [19] M. Mukadam, X. Yan, and B. Boots, “Gaussian Process Motion planning,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2016, pp. 9–15.
- [20] J. Wang and B. Englot, “Fast, accurate gaussian process occupancy maps via test-data octrees and nested Bayesian fusion,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2016, pp. 1003–1010.
- [21] P. Poulakis, J. L. Vago, D. Loizeau, C. Vicente-Arevalo, A. Hutton, R. McCoubrey, J. Arnedo-Rodriguez, J. Smith, B. Boyes, S. Jessen, A. Otero-Rubio, S. Durrant, G. Gould, L. Joudrier, Y. Yushtein, C. Alary, E. Zekri, P. Baglioni, A. Cernusco, F. Maggioni, R. Yague, and F. Ravera, “Overview and development status of the Exomars rover mobility subsystem,” in *Advanced Space Technologies for Robotics and Automation*, 2015, pp. 1 – 8.
- [22] J. Hidalgo, P. Poulakis, A. Barrientos, and J. Del-cerro, “ESTEC Testbed Capabilities of the performance Characterization of Planetary Rover Localization Sensors,” in *Symposium on Advanced Space Technologies in Robotics and Automation*, no. 1, 2011, pp. 1–8.
- [23] GPpy, “GPpy: A gaussian process framework in python,” <http://github.com/SheffieldML/GPy>, since 2012.
- [24] F. Meier, P. Hennig, and S. Schaal, “Incremental Local Gaussian Regression,” in *Advances in Neural Information Processing Systems*, 2014, pp. 972–980.