Robocentric SLAM

S. P. Arjun Ram * and Renato Zanetti†
Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, Texas 78712

I. Introduction

The future of planetary exploration relies on autonomous rovers which can navigate themselves in unknown environments [1]. The problem of Simultaneous Localization and Mapping (SLAM) for robotic systems has been extensively studied for spatial exploration [2, 3, 4]. The idea of using a robotic agent that combines the problem of building a map of a new space along with locating the agent in the map being built is ideal for a rover exploring a planet. The robot performing the exploration task is usually equipped with some form of sensor, often visual, that helps the robot observe its surroundings as well as some form of odometry that helps the robot measure its own motion. Map building is often done in the form of features extracted from sensor measurements of the surroundings, stored along with their locations in a known frame of reference. Knowledge of the position of the rover at the time of observation is required to place new features on the map or refine their estimated position. The interlink between robot and feature positions leads to correlations between their estimates; having both the position of the rover and features in the filter's state accounts for these correlations and helps reduce errors in both. Planning of the rover's path by the motion controller requires the rovers current location relative to the environment. The SLAM objective is to provide the motion controller with the position of the rover and the map of features as needed.

The nonlinear SLAM problem is usually formulated as either a recursive filter or via sparse optimization. Keyframe-based Bundle Adjustment (BA) techniques [5, 6] aggregate measurements at different times and use a numerical optimizer to compute the SLAM solution. Alternatively, recursive implementations only process the latest measurement and are typically based on either the Extended Kalman Filter (EKF) or the particle filter. Particle filters (unlike the EKF) are nonlinear estimators, and their implementations for SLAM applications, such as FAST-SLAM [7], have their own strengths and limitations in terms of complexity and consistency. In this paper we focus on the EKF-SLAM approach and represent the state and the uncertainty with an estimated mean and a covariance matrix. EKF-SLAM employs linearization to apply the Kalman filter algorithm to the nonlinear propagation and measurement functions. Linearization can affect the consistency of the filter as situations arise when the linear approximation of the function is not sufficiently accurate. This is particularly problematic because the SLAM system is inherently nonlinear and unobservable, a combination known to cause divergence in the EKF. This phenomenon led to extensive work to study the consistency of the EKF-SLAM showing that the algorithm is eventually bound to be inconsistent [8, 9, 10, 11].

*Graduate Research Assistant and Ph.D. Candidate, Email: arjun.ram@utexas.edu
†Assistant Professor, Email: renato@mail.utexas.edu. AIAA Associate Fellow

Divergence of the SLAM algorithm can be detrimental to any exploration task since it can lead to loss of localization feedback for controlling the rover as well as map feature locations becoming inconsistent since they are interlinked with the position.

The SLAM linearization approximation introduces apparent observability to the unobservable subspace [12]. As a result, the covariance estimates of the EKF undergo reduction in directions of the state-space where no information is actually available, making the filter more confident than it should, thus creating inconsistency and even divergence. A classic example to demonstrate the divergence problems of EKF-SLAM is given in Ref .[11], where a stationary robot with no process noise, observing a single stationary feature, eventually diverges. A relative measurement between the robot and the features (for example a LIDAR returning range and bearing angles) is the only sensor available, and hence the absolute positions in the global frame are not observable, and neither is the global heading angle.

A source of inconsistency is given by the linearization of the rover's heading error which affects the rotation transformation of the odometry; small errors in heading can lead to large errors in position. For relative navigation applications however, we do not need to formulate the SLAM problem in the world-centric form, leading to the development of robocentric methods. In the robocentric approach [9, 13, 14], the global position of the robot is kept as a state in the EKF together with position of the features relative to it. The robocentric features positions are fully observable and the measurement model can be linearized more accurately providing much better consistency characteristics for the EKF algorithm. Ref .[13], in order to reduce the complexity of the propagation step, does not propagate the change in robocentric position of the features due to the robot's motion; rather, it appends all odometry values starting from the latest measurement update step as components of the state vector. When a new measurement is available from the LIDAR, the global robot position is updated followed by the features positions, while the odometry states are discarded. The robocentric mapping idea presented in Ref .[13] has much better consistency properties as compared to the traditional EKF because the odometry states have small uncertainty as they get reset often, but some of the underlying observability issues are still present. Ref .[14] modifies this approach for visual-inertial odometry and reduces the computational complexity by not mapping the features and thus reducing the size of the state-space, an approach similar to Ref .[15] but including the robocentric idea. The robocentric approach is readily applicable and an excellent choice of SLAM for relative navigation problems where the objective is to navigate a robot to one or more of the features being observed.

In this work, we propose two key modifications to the robocentric mapping idea. First, the odometry measurements are used to propagate both the robot's position and the robocentric features' positions, with all relevant correlation terms accounted for. Second, we include second order terms of the Taylor series expansion of the heading error during the propagation step to greatly improving the consistency of the filter over time. The only minor disadvantage to this method is the increased computational complexity in roto-translating the map features positions during the propagation step whereas they are stationary in the classic EKF-SLAM formulation. In the counter example proposed by Julier and

Uhlmann [11], with a stationary robot and no process noise, our proposed algorithm never diverges, but neither does the original robocentric approach [13]. However, when we make a slight modification to the counter example in Ref .[11] and add process noise to the propagation step, the algorithm in Ref .[13] diverges, while the method proposed in this paper does not. Moreover, the proposed algorithm is more consistent than existing methods in the presented simulated scenarios involving a moving robot observing features using a Lidar. The effectiveness of the method is tested in a 2D simulation as well as in experiments on a ground rover and compared against Ref .[13].

The contributions of this paper are: analysis of the observability of the states of the EKF in the global and robocentric frame for a better understanding of how the transformation impacts the filter, and introduction of the above-mentioned modifications to the existing robocentric SLAM algorithm to improve its consistency and robustness. Throughout this paper, predicted values are represented by a bar on the variable $\bar{a}$ while updated estimates are represented by a hat $\hat{a}$.

II. EKF-SLAM Algorithm

Consider a rover navigating in a 2D environment, equipped with odometry and a LIDAR. The odometry can be provided using any available sensor like processing the measurements of an IMU or wheel encoders. The path taken by the rover is assumed prescribed and the localization and mapping module does not receive feedback inputs from the motion controller. We assume a feature detection technique identifies and extracts points of interest in the planetary environment from the LIDAR measurement data. The data used by EKF-SLAM are the range and angle measurements from the rover to these features. The odometer is providing measurements of linear translational and rotational velocity, which can be integrated over time to obtain the distance moved and change in heading of the rover since the last odometer reading. We also assume Gaussian random noise with known covariance matrices Q and R corrupting the odometry and LIDAR measurements respectively.

A. Classical EKF-SLAM

Standard SLAM applications use the position of the rover $\boldsymbol{r}_r^G = \left[x_r^G, y_r^G, \theta_G^R\right]^T$ and the landmark features $\boldsymbol{r}_{f_i}^G = [x_{f_i}^G, y_{f_i}^G]^T$, where $i = 1, 2, ...$ uniquely identify the features. The superscript G indicates the quantity is expressed in a fixed global frame of reference $\{G\}$ and the quantity θ_G^R is the angle from the global to a robot-fixed robot-centered frame $\{R\}$. Since the center and orientation of the global reference frame is unobservable, it is typically chosen to be the initial location of the robot setting the initial uncertainty to zero, which was shown to produce better filter consistency than choosing any other fixed global frame with non-zero initial uncertainty in the robot position [13]. The state estimates at any time t_k are given by

$$\hat{\boldsymbol{x}}_k^G = \left[\hat{\boldsymbol{r}}_r^G(k)^T, \hat{\boldsymbol{r}}_{f_1}^G(k)^T, \hat{\boldsymbol{r}}_{f_2}^G(k)^T ...\right]^T \tag{1}$$

The propagation step for the classical EKF-SLAM algorithm adds the odometry $\boldsymbol{u}_k^R = [\delta x_k^R, \delta y_k^R, \delta\theta_k^R]^T$ to the robot position states. The odometry measurement is obtained in a robocentric frame of reference $\{R\}$.

$$\begin{bmatrix} \bar{x}_r^G(k+1) \\ \bar{y}_r^G(k+1) \\ \bar{\theta}^G(k+1) \end{bmatrix} = \begin{bmatrix} \hat{x}_r^G(k) \\ \hat{y}_r^G(k) \\ \hat{\theta}^G(k) \end{bmatrix} + \begin{bmatrix} & & 0 \\ & T\left(\hat{\theta}_G^R(k)\right)^T & 0 \\ & & \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{u}_k^R \tag{2}$$

where

$$T(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}$$

so that $T\left(\hat{\theta}_G^R(k)\right)$ is the direction cosine matrix (DCM) to change coordinates from the global frame to the robocentric frame. The features' positions remain the same across the time propagation phase of the filter.

P^G is the covariance of the state $\boldsymbol{x}_k^G$ given by:

$$P^G = \begin{bmatrix} P_{rr}^G & P_{rf}^G \\ P_{fr}^G & P_{ff}^G \end{bmatrix} \tag{3}$$

where we have divided the state covariance into the cross covariance of the robot position and feature positions and their auto-covariances. The propagation of the covariance evaluated at the predicted state at time t_{k+1} is given by

$$\bar{P}_{rr}^G(k+1) = \left[\frac{\partial \bar{r}_r^G(k+1)}{\partial \hat{r}_r^G(k)}\right]^T \hat{P}_{rr}^G(k) \left[\frac{\partial \bar{r}_r^G(k+1)}{\partial \hat{r}_r^G(k)}\right] + \left[\frac{\partial \bar{r}_r^G(k+1)}{\partial u}\right]^T Q \left[\frac{\partial \bar{r}_r^G(k+1)}{\partial u}\right] \tag{4}$$

$$\bar{P}_{rf}^G(k+1) = \left[\frac{\partial \bar{r}_r^G(k+1)}{\partial \hat{r}_r^G(k)}\right]^T \hat{P}_{rf}^G(k) \tag{5}$$

$$\bar{P}_{ff}^G(k+1) = \hat{P}_{ff}^G(k) \tag{6}$$

The noisy measurement from the LIDAR at true robot position $[x_r^G, y_r^G, \theta_G^R]^T$ to the n-th landmark feature at true global position $[x_{f,n}^G, y_{f,n}^G]^T$ in the global frame is given by

$$\tilde{z}_n = z_n + \begin{bmatrix} w_r \\ w_\phi \end{bmatrix} \tag{7}$$

where $[w_r, w_\phi]$ are the noise corrupting the range and bearing respectively, with covariance matrix R, and

$$z_n = \begin{bmatrix} r_n \\ \phi_n \end{bmatrix} = \begin{bmatrix} \sqrt{(x^G_{f,n} - x^G_r)^2 + (y^G_{f,n} - y^G_r)^2} \\ \tan^{-1}\left((y^G_{f,n} - y^G_r)/(x^G_{f,n} - x^G_r)\right) - \theta^R_G \end{bmatrix} \tag{8}$$

The filter's predicted measurement is given by

$$\bar{z}_n = \begin{bmatrix} \bar{r}_n \\ \bar{\phi}_n \end{bmatrix} = \begin{bmatrix} \sqrt{(\bar{x}^G_{f,n} - \bar{x}^G_r)^2 + (\bar{y}^G_{f,n} - \bar{y}^G_r)^2} \\ \tan^{-1}\left((\bar{y}^G_{f,n} - \bar{y}^G_r)/(\bar{x}^G_{f,n} - \bar{x}^G_r)\right) - \bar{\theta}^R_G \end{bmatrix} \tag{9}$$

The Jacobian of the measurement equation is

$$H^G_n = \begin{bmatrix} H^G_{r,n} & H^G_{f_1,n} & H^G_{f_2,n} & \dots \end{bmatrix} \tag{10}$$

where

$$H^G_{r,n} = \frac{\partial \boldsymbol{z}_n}{\partial \boldsymbol{r}^G_r} \tag{11}$$

and

$$H^G_{f_j,n} = \begin{cases} \frac{\partial \boldsymbol{z}_n}{\partial \boldsymbol{r}^G_{f_j}} & \text{if } n = j \\ \mathbf{0} & \text{if } n \neq j \end{cases} \tag{12}$$

The measurement residual is given by

$$\boldsymbol{y}_n = \tilde{\boldsymbol{z}}_n - \bar{\boldsymbol{z}}_n \tag{13}$$

and the innovation covariance matrix is

$$S = H^G_n \bar{P}^G (H^G_n)^T + R \tag{14}$$

The Kalman gain is given by

$$K = \bar{P}^G (H^G_n)^T S^{-1} \tag{15}$$

The updated state and covariance matrix are

$$\hat{\boldsymbol{x}} = \bar{\boldsymbol{x}} + K\boldsymbol{y}_n \tag{16}$$

$$\hat{P} = \bar{P} - KH_n\bar{P} \tag{17}$$

B. Robocentric EKF-SLAM

To handle the well-documented inconsistencies in the above described classic EKF-SLAM, Ref .[13] proposes the use of a robocentric frame of reference in which the features locations are stored with respect to a moving frame attached to the robot. Often times the relative position of the rover with respect to specific features and locations is all that is needed to accomplish mission objectives; for example obstacle avoidance, rendezvous with another rover, searching for a specific location on the planet, or close proximity operations for in-orbit satellite repair. In those scenarios the global position of neither the robot nor features is of importance. Thus, in these applications, robocentric SLAM algorithms provide a more stable method to calculate the needed information. The state vector of the robocentric EKF includes the position of the robot in the global frame $\{G\}$ (which can also be removed when purely relative information is needed) $\boldsymbol{r}_r^G = \left[x_r^G, y_r^G, \theta_G^R\right]^T$, and the positions of the features $\boldsymbol{r}_{f_i}^R = [x_{f_i}^R,\ \ y_{f,i}^R]^T$ expressed in the robocentric frame $\{R\}$. The state estimate at any time t_k is given by

$$\hat{\boldsymbol{x}}_k^R = \left[\hat{\boldsymbol{r}}_r^G(k)^T, \hat{\boldsymbol{r}}_{f_1}^R(k)^T, \hat{\boldsymbol{r}}_{f_2}^R(k)^T \ldots\right]^T \tag{18}$$

The global position of the features can be retrieved as needed.

In Ref .[13] the odometry data are added to the state vector during the time propagation step, increasing the state dimension. The measurement update step uses this augmented state to correct for the global position of the robot, the robot-relative positions of the features, and the odometry measurement added to the state. A third filter step is then introduced, the composition step, which transforms the feature positions to the current robocentric frame using the updated odometry data and then discards the odometry components of the state estimate.

In the classic EKF-SLAM approach, the global positions of the robot and of the features are individually not observable, but their difference is. Therefore neither uncertainty collapses but relative measurements build correlation between the two. The filter updates the different state components based on the relative uncertainty between the robot's position and the feature's. If the robot's position was known exactly, for example, its estimate would not change when a LIDAR measurement is processed but only the estimate of the feature's position would change. Larger uncertainty also results in the measurement Jacobian being potentially evaluated at an estimated state value far from the truth.

Ref .[13] effectively introduces a new global frame after each measurement update, so that the odometry states and the features are individually unobservable, but their difference is. This method performs better than the classic EKF-SLAM in terms of consistency because the uncertainty associated with the odometry states is much smaller than the typical uncertainty associated with the robot's position, hence the filter "knows" where to apply the measurement update. Yet, under challenging scenarios (for example a long measurement drop-off or very noisy odometry) the original robocentric EKF-SLAM can still diverge. The stationary robot observing a stationary feature counter example [11] is an example where EKF-SLAM diverges and switching to a robocentric frame helps with filter consistency. However, if

process noise is added to this same example scenario, the robocentric approach in Ref .[13] does diverge after extended periods of time. To mitigate this behavior, we introduce a new robocentric EKF-SLAM formulation.

III. Modified Robocentric EKF-SLAM

In the proposed algorithm, we recommend the use of the robocentric frame but eliminate the process of appending the odometry data to the state. Instead, we use odometry in the time propagation step to move the estimated position of the robot and at the same time transform the estimated feature positions to the new robocentric frame, removing the need for the composition step of Refs.[13, 14]. Additionally we introduce a second order correction in the propagation to enhance robustness of the algorithm. The propagation and update steps are described next.

A. Propagation

Like before, the true position of the robot is $\boldsymbol{r}_r^G = \left[x_r^G, y_r^G, \theta_G^R\right]^T$ and the odometry measurement is modeled as

$$\boldsymbol{u}_k^R = \begin{bmatrix} T\left(\theta_G^R(k)\right) \begin{bmatrix} x_r^G(k+1) - x_r^G(k) \\ y_r^G(k+1) - y_r^G(k) \end{bmatrix} \\ \theta_G^R(k+1) - \theta_G^R(k) \end{bmatrix} + \boldsymbol{v}_k \tag{19}$$

the heading angle θ_G^R is counted positive from G to R so that $T\left(\theta_G^R(k)\right)$ is the DCM to transform coordinates from G to R. The odometry noise $\boldsymbol{v}_k$ is assumed white, zero-mean, with covariance matrix Q.

Analogous to IMU-based dead reckoning, the odometry measurements $\boldsymbol{u}_k^R = [\delta \boldsymbol{r}_r^R(k)^T, \delta\theta_k^R]^T = [\delta x_k^R, \delta y_k^R, \delta\theta_k^R]^T$ are used to propagate the estimated state as:

$$\bar{\boldsymbol{r}}_r^G(k+1) = \hat{\boldsymbol{r}}_r^G(k) + \begin{bmatrix} T\left(\hat{\theta}_G^R(k)\right)^T \delta \boldsymbol{r}_r^R(k) \\ \delta\theta_k^R \end{bmatrix} \tag{20}$$

This equation is obtained linearizing all the errors and then taking the expected value. Starting from

$$T\left(\theta_G^R(k)\right) = T\left(e_\theta(k)\right) T\left(\hat{\theta}_G^R(k)\right) \tag{21}$$

where the heading angle estimation error is given by $e_\theta(k) = \theta_G^R(k) - \hat{\theta}_G^R(k)$ and taking the expected value of the right-hand side of Eq. (21) we obtain

$$E\left\{T\left(e_\theta(k)\right) T\left(\hat{\theta}_G^R(k)\right)\right\} = E\left\{T\left(e_\theta(k)\right)\right\} T\left(\hat{\theta}_G^R(k)\right) \tag{22}$$

$$E\left\{T\left(e_\theta(k)\right)\right\} = E\left\{\begin{bmatrix} \cos(e_\theta(k)) & \sin(e_\theta(k)) \\ -\sin(e_\theta(k)) & \cos(e_\theta(k)) \end{bmatrix}\right\} \approx E\left\{\begin{bmatrix} 1 & e_\theta(k) \\ -e_\theta(k) & 1 \end{bmatrix}\right\} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

this approximation leads to equation (20); so long as both $e_\theta(k)$ and $\delta \boldsymbol{r}_r^R(k)$ are small it produces satisfactory results.

Since the rover roto-translates, the robocentric coordinates of the j-th landmark feature roto-translate in the opposite direction

$$\bar{\boldsymbol{r}}_{f_j}^R(k+1) = T(\delta\theta_k^R)\left(\hat{\boldsymbol{r}}_{f_j}^R(k) - \delta \boldsymbol{r}_r^R(k)\right) \tag{23}$$

Similar to the discussion above, Eq. (23) is accurate so long as the uncertainty associated with $e_\theta(k)$ and the value $\boldsymbol{r}_{f_j}^R(k) + \delta \boldsymbol{r}_r^R(k)$ are small. In our analysis we found this not to be always true which leads to filter divergence. Our proposed approach is to include key second order terms:

$$E\left\{T\left(e_\theta(k)\right)\right\} \approx E\left\{\begin{bmatrix} 1 - e_\theta(k)^2/2 & e_\theta(k) \\ -e_\theta(k) & 1 - e_\theta(k)^2/2 \end{bmatrix}\right\} = \begin{bmatrix} 1 - Q(3,3)/2 & 0 \\ 0 & 1 - Q(3,3)/2 \end{bmatrix}$$

where $Q(3,3)$ is the variance of $\delta\theta_k^R$. By adding the second order terms, Eq. (23) is replaced by

$$\bar{\boldsymbol{r}}_{f_j}^R(k+1) = T(\delta\theta_k^R)\left(\hat{\boldsymbol{r}}_{f_j}^R(k) - \delta \boldsymbol{r}_r^R(k)\right) - \frac{1}{2}T(\delta\theta_k^R)\left(\hat{\boldsymbol{r}}_{f_j}^R(k) - \delta \boldsymbol{r}_r^R(k)\right) Q(3,3) \tag{24}$$

For the covariance time propagation, we define the matrices

$$F = \begin{bmatrix} F_{xx} & 0_{3\times2} & \dots & 0_{3\times2} \\ 0_{2\times3} & T\left(\delta\theta_k^R\right) & \dots & 0_{2\times2} \\ \vdots & \ddots & \vdots & \vdots \\ & \dots & & T\left(\delta\theta_k^R\right) \end{bmatrix} \tag{25}$$

where

$$F_{xx} = \begin{bmatrix} I_{2\times2} & dT\left(\hat{\theta}_G^R(k)\right)^T \delta \boldsymbol{r}_r^R(k) \\ 0_{1\times2} & 1 \end{bmatrix} \tag{26}$$

and

$$B = \begin{bmatrix} & & 0 \\ T\left(\hat{\theta}_G^R(k)\right)^T & & \\ & & 0 \\ 0 \quad & 0 & 1 \\ T(\delta\theta_k^R) & & dT(\delta\theta_k^R)\left(\hat{\boldsymbol{r}}_{f_1}^R(k) - \delta\boldsymbol{r}_r^R(k)\right) \\ T(\delta\theta_k^R) & & dT(\delta\theta_k^R)\left(\hat{\boldsymbol{r}}_{f_2}^R(k) - \delta\boldsymbol{r}_r^R(k)\right) \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{27}$$

where

$$dT(\alpha) = \begin{bmatrix} -\sin(\alpha) & \cos(\alpha) \\ -\cos(\alpha) & -\sin(\alpha) \end{bmatrix} \tag{28}$$

The classic EKF first order covariance propagation is given by

$$\bar{P}_{k+1} = FP_kF^T + BQB^T \tag{29}$$

The covariance of the landmark features is further increased to include the second order terms

$$\bar{P}_{f_i,f_j}(k+1) \Leftarrow \bar{P}_{f_i,f_j}(k+1) + \frac{1}{2}Q(3,3)^2T(\delta\theta_k^R)A_{ij}T(\delta\theta_k^R)^T \tag{30}$$

for all i and j, where P_{f_i,f_j} are the 2×2 components of the covariance matrix whose row corresponds to the i-th landmark and whose column corresponds to the j-th landmark, and where

$$A_{ij} = \left(\hat{\boldsymbol{r}}_{f_i}^R(k) - \delta\boldsymbol{r}_r^R(k)\right)\left(\hat{\boldsymbol{r}}_{f_j}^R(k) - \delta\boldsymbol{r}_r^R(k)\right)^T \tag{31}$$

Two unique features about our algorithm are worth mentioning. First, this propagation step converts the features coordinates from robocentric frame at time t_k to the robocentric frame at time t_{k+1} fully accounting for the common process noise terms corrupting the propagation of both the robot position and the features positions. Second, we include selected second order terms to aid the consistency of the filter. While the second order terms we chose to include are sufficient to ensure consistency in all of the test performed, it is possible that different scenarios might necessitate additional second order terms, or perhaps even higher than second order terms. The extreme conditions that might require the extra terms however, seem unlikely for practical situations.

In all examples and experiments performed, the second order components associated with the term $T\left(\theta_G^R(k)\right)^T\delta\boldsymbol{r}_r^R(k)$ in the robot's position propagation were negligible because the odometry term $\delta\boldsymbol{r}_r^R(k)$ is small. It is plausible, however,

that long propagation steps due to measurement blackouts would cause this term to be needed.

B. Update

The predicted measurement for the i^{th} landmark is given by

$$\bar{\boldsymbol{z}}_i = \begin{bmatrix} \sqrt{(\bar{x}^R_{f_i})^2 + (\bar{y}^R_{f_i})^2} \\ \tan^{-1}\left(\bar{y}^R_{f_i} / \bar{x}^R_{f_i}\right) \end{bmatrix} \tag{32}$$

while the measurement Jacobian is given by

$$H^R = \begin{bmatrix} 0_{2\times 3} & \dots & H^R_{f_i} & \dots \end{bmatrix} \tag{33}$$

where

$$H^R_{f_i} = \begin{bmatrix} \frac{\bar{x}^R_{f_i}}{\bar{z}_i(1)} & \frac{\bar{y}^R_{f_i}}{\bar{z}_i(1)} \\ \frac{-\bar{y}^R_{f_i}}{\bar{z}_i(1)^2} & \frac{\bar{x}^R_{f_i}}{\bar{z}_i(1)^2} \end{bmatrix} \tag{34}$$

Matrix H^R above is linearized along the estimated relative position between the robot and the features. The global position or heading of the robot are not used in evaluating this derivative and hence their uncertainties do not contribute to errors in the Jacobian's evaluation. The update equations follow the conventional Kalman filter approach shown in Eqs. 13 - 17.

C. New Features

When a feature that is not yet part of the state is detected by the LIDAR, the new measurement is used to initialize the feature's estimate rather than to update the state. Moreover, the state covariance matrix is also augmented to include the uncertainty in the position of the feature as seen by the robot's current frame. If the measurements received for this new feature are given by the range and heading $[r_i, \phi_i]^T$, the state is augmented as:

$$\bar{\boldsymbol{x}}_{k+1} \Leftarrow \left[\bar{\boldsymbol{x}}^T_{k+1}, r_i \cos\phi_i, r_i \sin\phi_i\right]^T \tag{35}$$

and covariance as:

$$\bar{P}_{k+1} \Leftarrow \begin{bmatrix} \bar{P}_{k+1} & 0 \\ 0 & H_{z_i} R H^T_{z_i} \end{bmatrix} \tag{36}$$

where R is the measurement noise covariance matrix and

$$H_{z_i} = \begin{bmatrix} \cos\phi_i & -r\sin\phi_i \\ \sin\phi_i & r\cos\phi_i \end{bmatrix} \tag{37}$$

Once again, the global position of the robot does not play a role in the addition of new features to the state, since all position estimates for the filter are stored in the robocentric frame of reference.

IV. Observability Discussion

It is well known that the global frame and azimuth are unobservable for terrestrial SLAM problems. The classical EKF-SLAM parameterizes the state with the global position of both the robot and the features, and with this parameterization the difference between the two is observable, while a global roto-translation of both is unobservable. With the proposed robocentric formulation, the state vector is naturally divided into its observable and unobservable components: the robocentric features are observable and the global robot position is not. That is to say, the initial global uncertainty of the robot's location and heading cannot be improved upon using odometry and lidar measurements only. This is one of the reasons many SLAM algorithms define an arbitrary global frame to coincide with the robot's initial pose, hence the initial covariance associated with the robot's state is set to zero.

Perfect odometry (or zero process noise) would keep the robot's and the features' uncertainty completely uncorrelated. Under this hypothetical scenario the robot's azimuth uncertainty would remain constant, and it's location uncertainty will grow linearly with slope determined by the initial azimuth uncertainty. In the absence of process noise, the robocentric feature's uncertainty will monotonically decrease to zero while they are observed. The counter example from Ref . [11] is a stationary robot problem observing a single feature and showed that EKF-SLAM is inevitably doomed to diverge. The divergence is due to the combined effect of the non-linearity of the system which is also unobservable. The robocentric approach however does not suffer from this divergence, because the observable and unobservable states are separated and hence the filter "knows" where to apply the measurement update.

Adding process noise (odometry uncertainty) correlates the robot's global position to the features robocentric position. This allows for some of the uncertainty added to the robot's position from process noise to be scaled back with a lidar measurement update. The robot's pose is still unobservable (only uncertainty added during propagation can be removed with measurement updates, the initial uncertainty of the global frame remains).

When adding process noise to the counter example in Ref . [11] the robocentric approach diverges when the second order components are omitted. The proposed addition of the second order components (Eqs. 23 and 30) compensates for the nonlinearities that cause this divergence and the proposed solution is able to handle this very challenging scenario. The process noise enters the system nonlinearly and causes the correlation between robot states and landmark features.

The feature's uncertainty and this correlation are then "used" by the filter to distribute the measurement update between the robot states and the feature. Adding the second order contributions allows for correctly accounting for the uncertainty and hence correctly distributing the measurement update and avoiding divergence.

V. Experiments

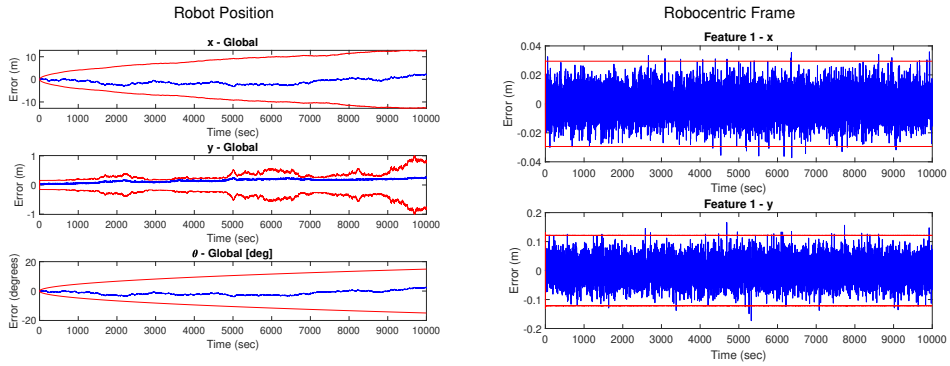
Since planetary exploration datasets or experiments are difficult to obtain, the performance of the proposed algorithm is tested in simulation in a 2D SLAM environment as well as on the dataset from Ref . [16]. A ground rover equipped with an odometer and a lidar is considered. We assume the odometry data provides distance moved by the robot in the forward direction (X axis in the robot frame) and the change in heading angle of the robot (angular velocity). The Lidar can recognize features in the environment at up to a distance of 100m from the robot and at an angle of within 15 degrees in either direction of the rover's heading. The odometer measurement error has standard deviation of 2cm/sec in linear velocity and 0.1 deg/sec in angular velocity, uncorrelated with each other. The range and bearing measurement are assumed to also be uncorrelated, with an error standard deviation of 1cm in range and 0.05 degrees in bearing for each measurement.

The counterexample from Ref . [11] consists of a static robot (which knows it is static, hence odometry is not needed) observing a single feature. From Ref . [11] it is known that the classic EKF-SLAM diverges in this scenario while both our proposed approach and the robocentric SLAM from Ref . [13] perform well. The first test case shown is a variation of the counterexample from Ref . [11] in which the robot is stationary but it does not know it is. Hence, odometry is used by the robot and the odometry error corrupts the estimate. Figs. 1 and 2 show the performance of both our proposed algorithm and the robocentric SLAM from Ref . [13]. It can be seen that while our algorithm performs correctly the robocentric approach from Ref . [13] diverges.

In the second test case the robot follows a typical exploration circular path surrounded by features, shown in Fig. 3. Fig. 4 shows the performance of the proposed algorithm while Fig. 5 shows the performance of the robocentric algorithm from Ref .[13]. The results show that our algorithm is able to estimate both the robot's position and the features locations, while the algorithm from Ref . [13] is only able to correctly estimate the features. Monte Carlo simulation runs (zoomed in on the time scale for better visibility) for the second test case are shown in Figs. 6 and 7.

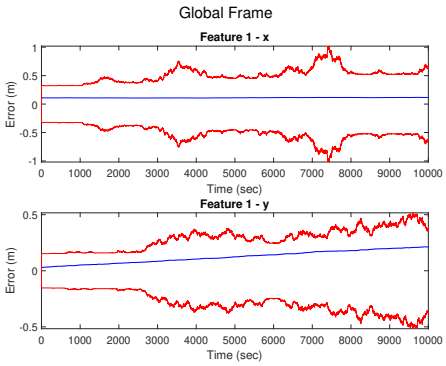
The third test case shown is the real-world data-set from Ref . [16]. The data-set was collected using a set of robots equipped with wheel odometry which provides forward translational and angular velocity, a camera providing range and bearing measurements to features which are barcodes in the environment and the true locations of the robot and the features using a motion capture system. The noise characteristics of the odometer and camera system are provided in Ref .[16]. Fig. 8 shows the performance of the proposed algorithm for this data-set which can be seen to perform correctly in estimating both the robot's state and the features' location.

The fourth and last test case is on a real 4-wheeled robot with linear and angular odometry and a camera detecting



(a) Robot Position in Global Frame

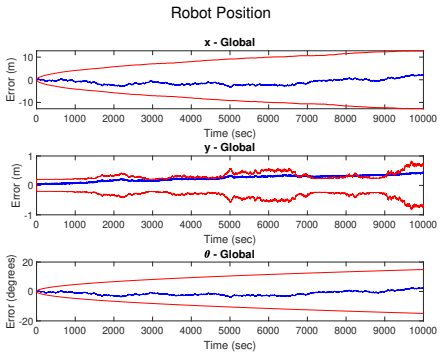
(b) Feature in Local Frame



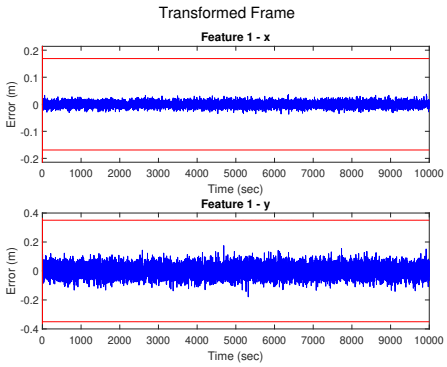
(c) Global Features

Fig. 1 Robot and Feature position from the proposed algorithm for the stationary robot case from Ref . [11]

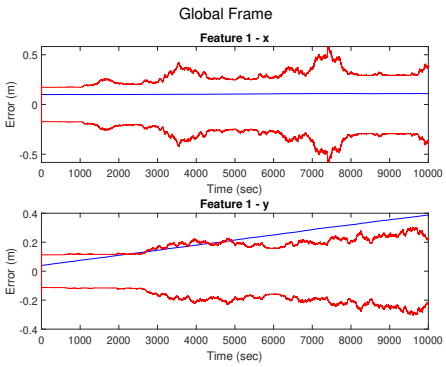
QR codes for features (Fig. 9) providing range and bearing measurements. The setup and robot are shown in Fig. 10. Fig. 12 shows the performance of the proposed algorithm for the experimental setup with the robot following the trajectory shown in Fig. 11. The odometer error standard deviation is 2cm/sec in linear velocity and 0.01 deg/sec in angular velocity, while the LIDAR measurements have 5 cm in range and 5 degrees in bearing.



(a) Robot Position in Global Frame



(b) Feature in Local Frame



(c) Global Features

Fig. 2 Robot and Feature position from Ref . [13] for the stationary robot case from Ref . [11]

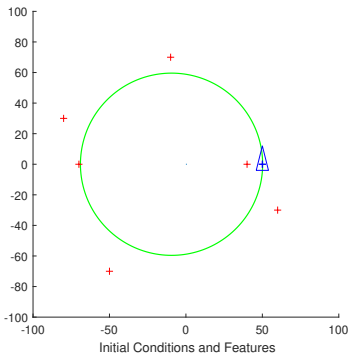
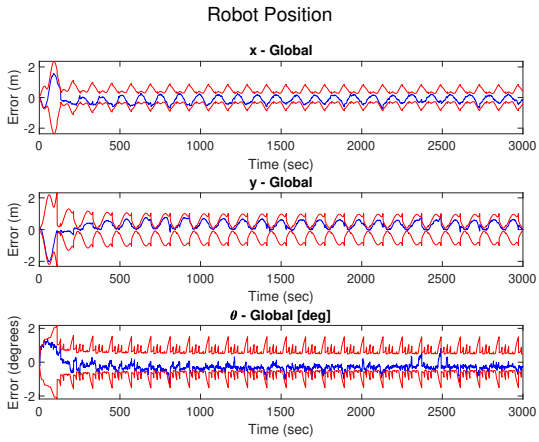
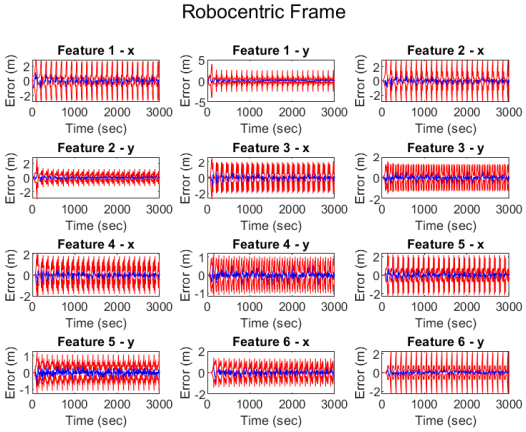


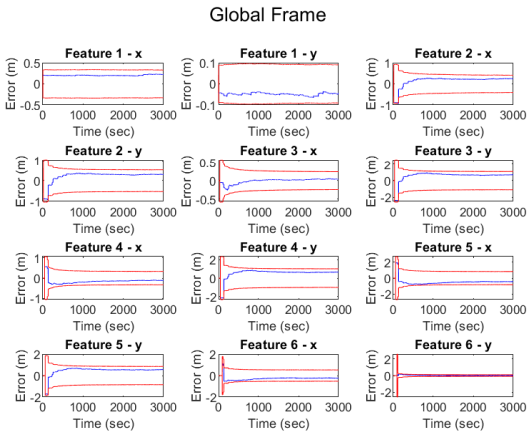
Fig. 3 Simulation test case 2



(a) Robot Position in Global Frame

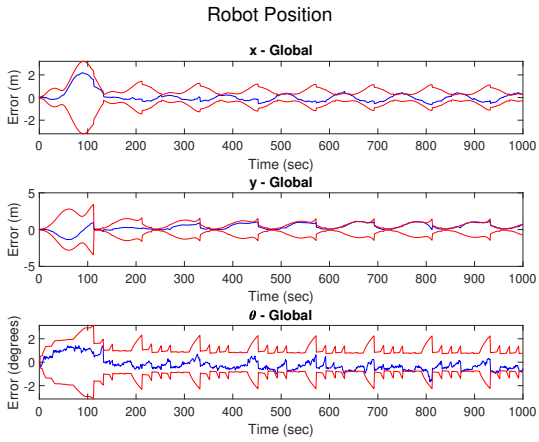


(b) Local Features

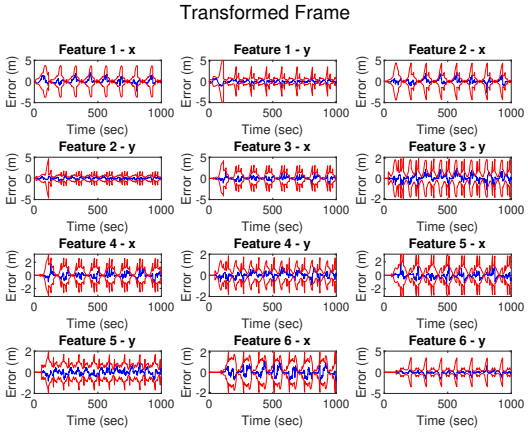


(c) Global Features

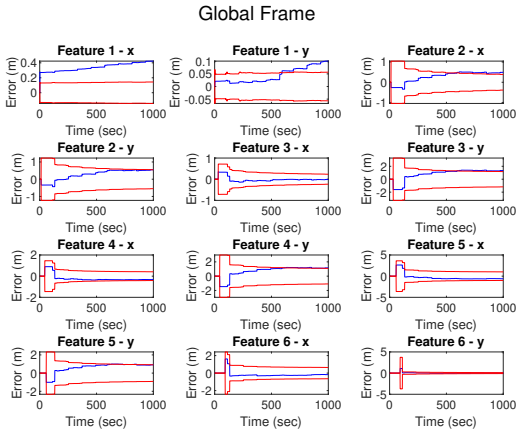
Fig. 4 Performance of the proposed algorithm in test case 2



(a) Robot Position in Global Frame for Ref . [13]

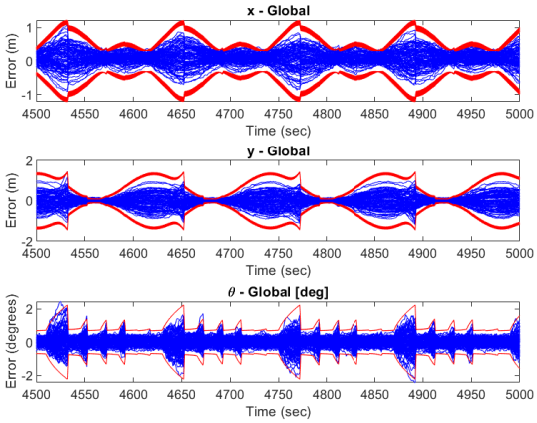


(b) Local Features for Ref . [13]

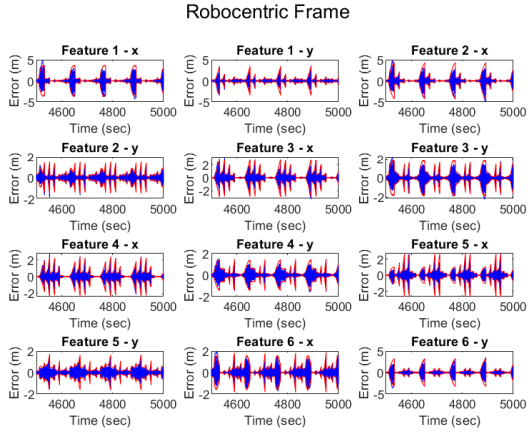


(c) Global Features for Ref . [13]

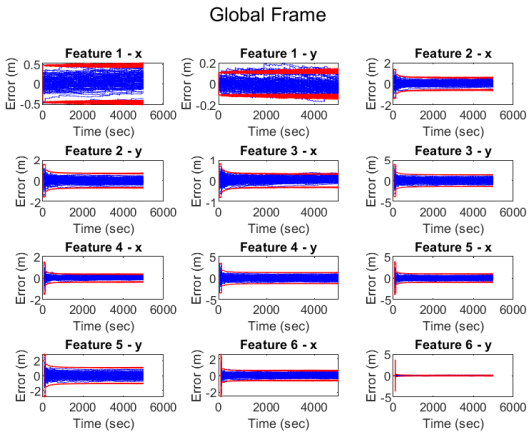
Fig. 5 Performance of the robocentric algorithm from Ref . [13] in test case 2



(a) Robot Position in Global Frame

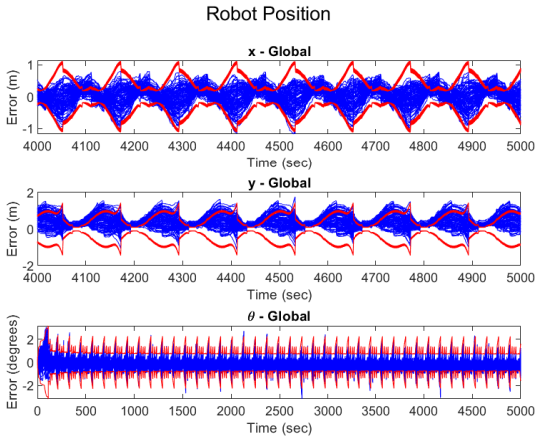


(b) Local Features

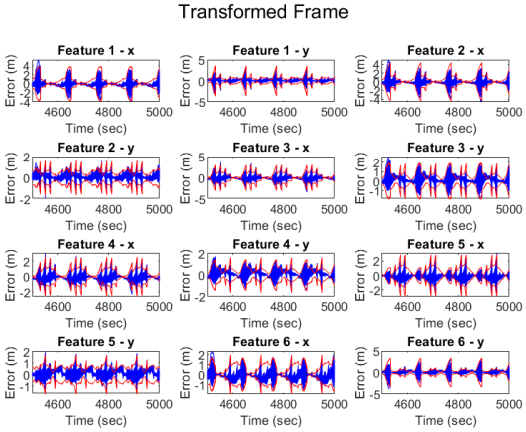


(c) Global Features

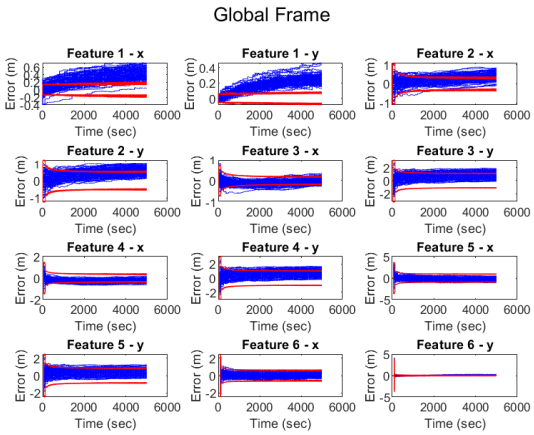
Fig. 6 Monte Carlo Performance of the proposed algorithm in test case 2



(a) Robot Position in Global Frame for Ref . [13]

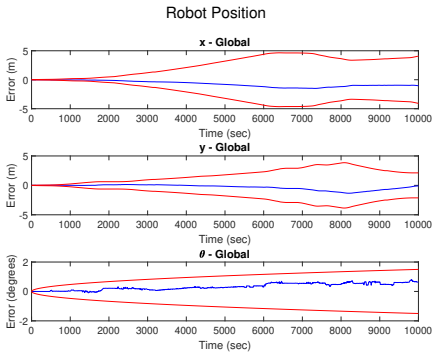


(b) Local Features for Ref . [13]

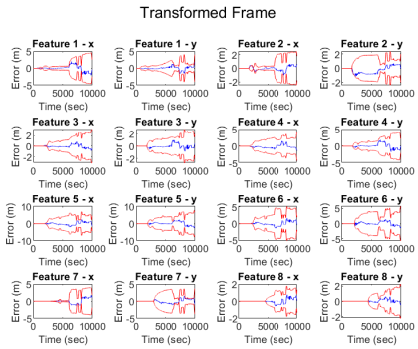


(c) Global Features for Ref . [13]

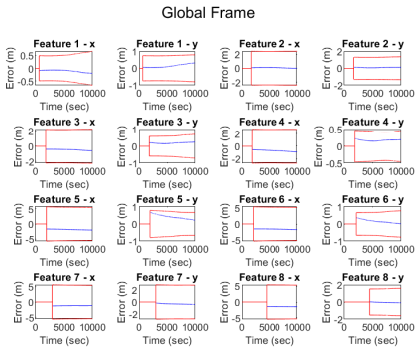
Fig. 7 Monte Carlo Performance of the robocentric algorithm from Ref . [13] in test case 2



(a) Robot Position in Global Frame



(b) Local Features



(c) Global Features

Fig. 8 Performance of the proposed algorithm using the data-set from Ref [16]

Fig. 9 Features used in the experiments

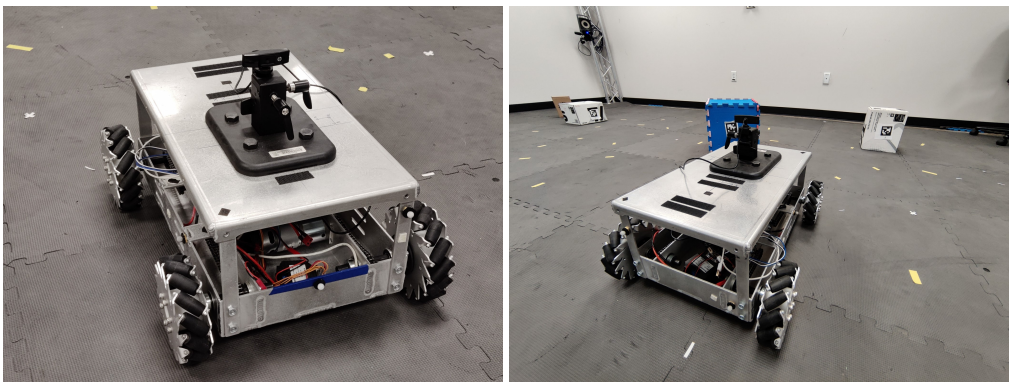


Fig. 10 Robot used for experiments

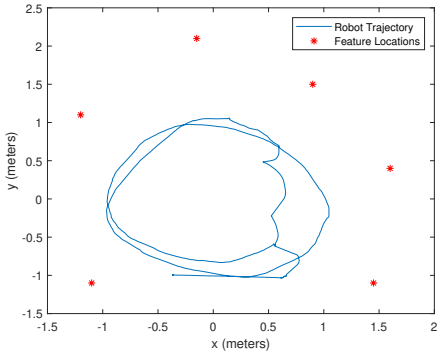


Fig. 11 Trajectory followed in the Experimental Setup

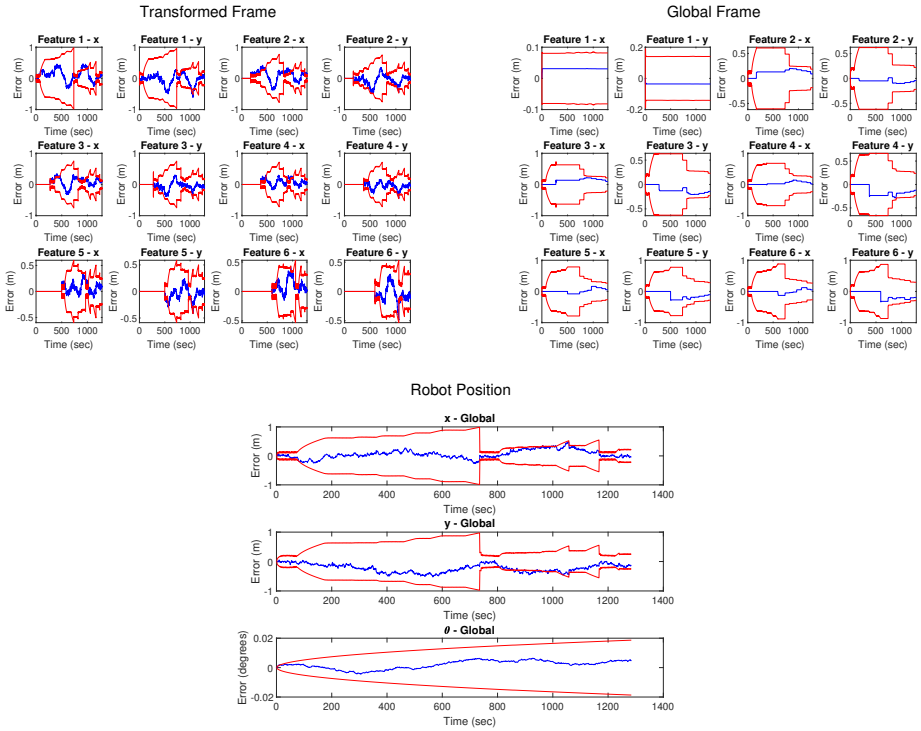


Fig. 12 Experimental performance of the proposed algorithm (test case 4)

VI. Conclusions

The problem of planetary exploration by a ground rover performing Simultaneous Localization and Mapping (SLAM) was considered. A novel robocentric Extended Kalman Filter (EKF) based SLAM algorithm was proposed which uses a second order linearization for the propagation function and transforms the full feature state to a frame with its origin at the robots position before every update step. This idea improves upon existing robocentric EKF-SLAM algorithms in the literature and prevents divergence in challenging scenarios where existing methodologies failed. The proposed algorithm also performed consistently in the stationary robot with process noise scenario which was previously not possible. The algorithm is tested in simulation, with a real world data-set, and on a physical robot; and performed successfully in all the scenarios including those where existing EKF-SLAM algorithms failed.

References

[1] Wong, C., Yang, E., Yan, X.-T., and Gu, D., "Adaptive and intelligent navigation of autonomous planetary rovers — A survey," *2017 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2017, pp. 237–244. https://doi.org/10.1109/AHS.2017.8046384.

[2] Cheeseman, P., Smith, R., and Self, M., "A stochastic map for uncertain spatial relationships," *4th International Symposium on Robotic Research*, MIT Press Cambridge, 1987, pp. 467–474.

[3] Leonard, J. J., and Durrant-Whyte, H. F., "Simultaneous map building and localization for an autonomous mobile robot." *IROS*, Vol. 3, 1991, pp. 1442–1447.

[4] Geromichalos, D., Azkarate, M., Tsardoulias, E., Gerdes, L., Petrou, L., and Perez Del Pulgar, C., "SLAM for autonomous planetary rovers with global localization," *Journal of Field Robotics*, Vol. 37, No. 5, 2020, pp. 830–847. https://doi.org/https://doi.org/10.1002/rob.21943, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21943.

[5] Mur-Artal, R., and Tardós, J. D., "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, Vol. 33, No. 5, 2017, pp. 1255–1262.

[6] Engel, J., Schöps, T., and Cremers, D., "LSD-SLAM: Large-scale direct monocular SLAM," *European conference on computer vision*, Springer, 2014, pp. 834–849.

[7] Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al., "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," *IJCAI*, Vol. 3, 2003, pp. 1151–1156.

[8] Bailey, T., Nieto, J., Guivant, J., Stevens, M., and Nebot, E., "Consistency of the EKF-SLAM algorithm," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2006, pp. 3562–3568.

[9] Castellanos, J. A., Neira, J., and Tardós, J. D., "Limits to the consistency of EKF-based SLAM," *IFAC Proceedings Volumes*, Vol. 37, No. 8, 2004, pp. 716–721.

[10] Huang, S., and Dissanayake, G., "Convergence analysis for extended Kalman filter based SLAM," *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, IEEE, 2006, pp. 412–417.

[11] Julier, S. J., and Uhlmann, J. K., "A counter example to the theory of simultaneous localization and map building," *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, Vol. 4, IEEE, 2001, pp. 4238–4243.

[12] Huang, G. P., Mourikis, A. I., and Roumeliotis, S. I., "Observability-based rules for designing consistent EKF SLAM estimators," *The International Journal of Robotics Research*, Vol. 29, No. 5, 2010, pp. 502–528.

[13] Castellanos, J. A., Martinez-Cantin, R., Tardós, J. D., and Neira, J., "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robotics and autonomous systems*, Vol. 55, No. 1, 2007, pp. 21–29.

[14] Huai, Z., and Huang, G., "Robocentric visual-inertial odometry," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 6319–6326.

[15] Mourikis, A. I., and Roumeliotis, S. I., "A multi-state constraint Kalman filter for vision-aided inertial navigation," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, IEEE, 2007, pp. 3565–3572.

[16] Leung, K. Y., Halpern, Y., Barfoot, T. D., and Liu, H. H., "The UTIAS multi-robot cooperative localization and mapping dataset," *The International Journal of Robotics Research*, Vol. 30, No. 8, 2011, pp. 969–974.