2018 Annual American Control Conference (ACC)
June 27–29, 2018. Wisconsin Center, Milwaukee, USA

Ground Target Tracking Using a Monocular Camera and IMU in a Nonlinear Observer SLAM Framework

Jerel Nielsen[1] and Randal Beard[2]

Abstract— **This paper discusses moving ground target estimation using a recently proposed, uniformly globally asymptotically stable observer for simultaneous localization and mapping (SLAM), which only requires a monocular camera and an inertial measurement unit (IMU). The observer operates under a persistent excitation constraint and requires the inertial origin to remain in view throughout its operation. The contributions of this paper are an iterative procedure to increase convergence speed of nonlinear observers and a process for keeping the inertial origin within view. Simulation results demonstrate fast observer convergence and tracking of a maneuvering ground target relative to the observer's landmark estimates.**

I. Nomenclature

R_a^b	Rotation from reference frame a to b
$\hat{a}$	Estimate of true variable a
$\bar{a}$	Measurement of a
$\dot{a}$	Time derivative a

Superscript

i	Expressed in the inertial coordinate frame
b	Expressed in the vehicle body coordinate frame
cb	Expressed in the camera body coordinate frame
c	Expressed in the camera coordinate frame
$\top$	Matrix transpose

Subscript

a/b	Velocity or angular rate of frame a w.r.t. frame b
ab	Vector from a to b

II. Introduction

Autonomously tracking multiple targets has remained an active area of research for many years, due to the many applications, including law enforcement [1], air traffic control [2], collision avoidance [3], simultaneous localization and mapping (SLAM) [4], tracking space debris [5], and others. There are often three main difficulties in multiple target tracking (MTT): collecting sensor measurements of the targets, associating these measurements with the correct targets (data association), and filtering clutter from true target measurements. Measurement collection can be computationally expensive when image processing is required, for example, when tracking with video cameras. Data association also becomes more difficult as the number of targets and the amount of clutter in the measurements increases.

Many algorithms attempt to solve the MTT problem, such as multiple hypothesis tracking (MHT) [6], joint probabilistic data association (JPDA) [7], and probabilistic hypothesis density (PHD) [8] algorithms. These suffer from computational complexity, requirements of unknown prior information, poor track continuity, or large variance in target estimates. Recursive RANSAC (R-RANSAC) is a recently introduced MTT algorithm [9], [10], [11], [12], [13] that improves upon many of these issues. R-RANSAC extends the traditional random sample consensus (RANSAC) algorithm to recursively estimate multiple dynamic signals in clutter. It stores a set of track hypotheses and identifies the best hypothesis of each target's track, and given a sliding window of measurements, the algorithm either updates the existing hypotheses with a Kalman update or generates new tracks with the set of measurements using RANSAC. R-RANSAC can run in real time on desktop processors or with GPU image processing, but efforts have been made to improve real-time viability for aerial vehicles with low computational power, to track targets in scenes with a large amount of clutter [14].

The R-RANSAC algorithm can represent tracks directly in the image plane [12], or it can represent tracks in the inertial reference frame [15]. In order to track in the inertial frame, measurements are first projected from pixel coordinates into the inertial frame via a perspective projection. However, due to ease of implementation, MTT is typically done in the image frame using nearly constant velocity, acceleration, or jerk models for motion propagation. Object tracking in the image frame has several advantages, including noise introduced into the measurement only comes from one sensor (the camera), image noise is typically minimal for modern cameras, and the tracker does not depend on camera pose estimation. Advantages of inertial tracking include the ability to use target specific motion propagation models, tracks are readily available for multi-vehicle problems, the tracker can account for geography, and poor homography estimates do not directly affect the tracker.

In order to track targets in the inertial reference frame, inertial measurements must be provided to the target tracker. These inertial measurements can be obtained with a SLAM or some visual odometry (VO) algorithms. Many state-of-

*This research was supported through the U.S. Department of Defense SMART Scholarship program and by the Center for Unmanned Aircraft Systems (C-UAS), a National Science Foundation-sponsored industry/university cooperative research center (I/UCRC) under NSF Award No. IIP-1161036 along with significant contributions from C-UAS industry members, and in part by AFRL grant FA8651-13-1-0005.

[1]Jerel Nielsen is a graduate student in the Electrical and Computer Engineering department at Brigham Young University, Provo, UT 84602, USA jerel.nielsen@gmail.com

[2]Randal Beard is a professor of the Electrical and Computer Engineering department at Brigham Young University, Provo, UT 84602, USA beard@byu.edu

the-art SLAM [16] and VO [17] alorithms use nonlinear optimization techniques to estimate camera pose as well as 3D landmark positions. Others, such as [18], [19], take a filter-based approach and may include all desired estimates in, for example, an extended Kalman filter, unscented Kalman filter, or particle filter. These filter-based SLAM algorithms lack stability guarantees, recently however, others have begun to emerge [20], [21] with provable stability under persistent excitation constraints. In [21], the authors derive a linear, time-varying Kalman filter, where a measured velocity is required for input as well as an accurate attitude estimate. One concern with this filter is that its estimates converge but to an unknown inertial origin location. Additionally, velocity measurements for a multi-copter require an accurate drag model with no wind or a downward-facing camera to run optical flow with well-estimated scene depth. Similarly, the algorithm in [20] requires an accurate attitude estimate but doesn't require a velocity measurement. This is because it directly integrates accelerometer measurements for position and velocity, while simultaneously estimating a velocity bias.

The primary contributions of this paper are two modificaitons to enable a practical implementation of the observer in [20], that is, sliding window observer iteration and origin shifting. In [20], the observer assumes that the inertial origin is viewed by the camera at all times. This guarantees that position estimation converges to something that is measured, i.e. the landmark. However, it is not practical to assume that a single landmark will always be observed. This is overcome by occasionally updating the origin to a new inertial position that is in the camera's field of view whenever the previous origin is about to leave the current field of view. Position and landmark estimates are then shifted appropriately. This approach is similar to changing node frames as in [22] and also allows us to operate in a relative manner, while estimating the vector from the first origin simultaneously by simply storing the vectors associated with each origin update. Another issue with the nonlinear observer in [20] is that estimates do not usually converge quick enough for real time implementation. It is not desirable to fly around the starting point to force landmark and position estimates to converge over a ten to twenty second period. To compensate for this, we allow the observer to iterate on a sliding window of saved velocities and bearing measurements. We store the estimated velocity and bearing measurements for a time window, integrate the position estimate backward in time to the beginning of the window, and then re-run the observer over this window, using the saved data. Each time step, data older than the window size is dropped, new data is added, and the observer processes the new time window. While, this iterative procedure only runs for a short time on initialization and after origin updates, it greatly decreases the time to convergence. Finally, in order to estimate the position of a maneuvering target, we take three or more estimates of stationary landmarks close to the target to approximate a plane and then scale the bearing vector to the moving target so that it intersects this plane [23].

The data pipeline for tracking targets in an inertial frame is shown in Fig. 1. The nonlinear observer recently introduced in [20] is used to localize the vehicle and landmarks simultaneously in the inertial frame. Given these landmark estimates and the bearing vectors to the targets, we produce inertial measurements of the ground targets, which are then passed in a multiple target tracker (R-RANSAC) to create ground tracks for each of the targets. The pipeline requires four coordinate frames. The inertial frame is a fixed, Euclidean reference frame parameterized in North-East-Down coordinates. The body frame defines the translation and attitude of the vehicle body with respect to the inertial frame. The camera body frame is the translation and rotation of the camera with respect to the vehicle body. Finally, the camera frame's z-axis is aligned with the camera body frame's x-axis and its x-axis aligned with the camera body frame's y-axis, where the camera frame's z-axis aligns with the camera's optical axis with the y-axis down and x-axis to the right.

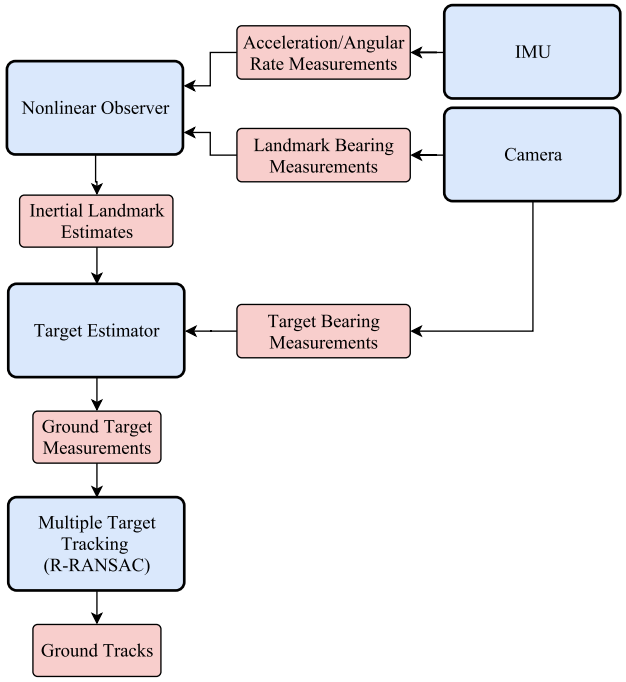

Fig. 1. The data pipeline for multiple target tracking in an inertial frame. Blue blocks represent functions and the red blocks represent outputs of these functions.

III. Nonlinear Observer SLAM

A. Observer Overview

There are four different observers derived in [20], but we are interested in the fourth one, which assumes an unknown velocity bias and unknown landmark positions. Let the state be defined by $\mathbf{x} = \begin{bmatrix}\mathbf{p}^i_{ib}; & \mathbf{p}^i_{il,1}; & \cdots; & \mathbf{p}^i_{il,N-1}\end{bmatrix}$ for estimating N landmark positions. The inertial kinematics of the vehicle body and j^{th} landmark are given by

$$\dot{\mathbf{p}}^i_{ib} = R^i_b \mathbf{v}^b_{b/i} \tag{1}$$

$$\dot{\mathbf{v}}^i_{b/i} = R^i_b \bar{\mathbf{a}}^b_{b/i} + \mathbf{g}^i \tag{2}$$

$$\dot{\mathbf{p}}^i_{il,j} = \mathbf{0}, \tag{3}$$

where $j = 0, 1, \ldots, N-1$, $\bar{\mathbf{a}}^b_{b/i}$ is the acceleration measured by the IMU, $\mathbf{g}^i$ is the gravity vector in the inertial frame, and $\mathbf{v}^b_{b/i}$ is the velocity of the vehicle in the body frame. The camera provides bearing vector measurements to each landmark given by $\mathbf{y}^c_j = \mathbf{p}^c_{cl,j} / \left\| \mathbf{p}^c_{cl,j} \right\|$, which are transformed into the inertial frame for use in the observer by

$$\mathbf{p}^i_{bl,j} = R^i_b \left(\mathbf{p}^b_{bc} + R^b_{cb} R^{cb}_c \mathbf{p}^c_{cl,j} \right) \tag{4}$$

$$\mathbf{y}^i_j = \frac{\mathbf{p}^i_{bl,j}}{\left\| \mathbf{p}^i_{bl,j} \right\|}. \tag{5}$$

In (4), we note that $\mathbf{p}^c_{cl,j}$ is not available. Therefore, bearing measurements are simply rotated into the inertial frame by $\mathbf{y}^i_j = R^i_b R^b_{cb} R^{cb}_c \mathbf{y}^c_j$. This introduces error into the bearing measurements that increases inversely with the distance to the landmarks, however, placing the camera and IMU near the body center, minimizes this error.

Because the multi-copter is constantly in motion, the IMU cannot correctly meausure the direction of gravity for attitude estimation, as in [24], and therefore an alternative form of attitude estimation is needed. Velocity-aided attitude observers, e.g. [25], show great promise for the camera-IMU sensor combination but throughout this paper, it is assumed that attitude is known. Having dropped superscripts for simplicity, the nonlinear observer derived in [20] is then given by

$$\dot{\hat{\mathbf{x}}} = \begin{bmatrix} \hat{\mathbf{v}} \\ \mathbf{0} \end{bmatrix} - kM\Pi\hat{\mathbf{x}} \tag{6}$$

$$\dot{\hat{\mathbf{v}}} = R^i_b \bar{\mathbf{a}}^b + \mathbf{g} \tag{7}$$

$$\dot{\hat{\boldsymbol{\beta}}} = -M\Pi M\hat{\boldsymbol{\beta}} - M\Pi\hat{\mathbf{x}} \tag{8}$$

$$\dot{M} = I - kM\Pi M \tag{9}$$

$$\hat{\mathbf{x}}^\beta = \hat{\mathbf{x}} + M\hat{\boldsymbol{\beta}}, \tag{10}$$

where $\hat{\mathbf{x}}$ is the biased state estimate, $\hat{\mathbf{x}}^\beta$ is the unbiased state estimate, $\hat{\boldsymbol{\beta}}$ is the estimated velocity bias, $k > 0$ is the observer gain, M is a real valued matrix, and

$$\Pi = \begin{bmatrix} \sum_{j=0}^{N-1} \pi_{y,j} & -\pi_{y,1} & -\pi_{y,2} & \cdots & -\pi_{y,N-1} \\ -\pi_{y,1} & \pi_{y,1} & 0 & \cdots & 0 \\ -\pi_{y,2} & 0 & \pi_{y,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -\pi_{y,N-1} & 0 & 0 & 0 & \pi_{y,N-1} \end{bmatrix} \tag{11}$$

$$\pi_{y,j} = I - \mathbf{y}_j \mathbf{y}_j^\top. \tag{12}$$

We initialize $\hat{\boldsymbol{\beta}}$ with zeros because the velocity error due to acceleration integration is initially small and M as identity, although it could be any symmetric, positive definite matrix.

B. Origin Shifting

The nonlinear observer defines the inertial origin as $\mathbf{p}^i_{il,0} = \mathbf{0}$, which means that the observer state is defined relative to the landmark $\mathbf{p}^i_{il,0}$. Note that the landmark $\mathbf{p}^i_{il,0}$ is not contained in the state because it is the origin and is always zero. The observer requires a bearing measurement to the origin landmark at all times, and therefore we must select a new origin whenever $\mathbf{p}^i_{il,0}$ is about to leave the camera's field of view. Since the observer is globally stable, we could select any landmark, even a newly acquired one but this would not allow us to easily track change in position from the first origin. Therefore, we select the new origin to be one of the landmark estimates that has already converged and is expected to leave the frame later than other current landmark estimates.

Assume that we have a history of origins $\mathcal{O}_0 \ldots \mathcal{O}_p$, where $\mathcal{O}_0$ is the initial origin and $\mathcal{O}_p$ is the origin at the current time step. Let $\mathbf{d}$ represent the vector pointing from $\mathcal{O}_0$ to $\mathcal{O}_p$. The estimate $\hat{\mathbf{d}}$ is initialized to a zero vector and every time the origin changes, we have $\hat{\mathbf{d}} = \hat{\mathbf{d}} + \hat{\mathbf{p}}_{il,j}$, where we've selected the j^{th} landmark $\hat{\mathbf{p}}_{il,j}$ to be the new origin. Additionally, the position estimate and each landmark estimate has to be shifted to account for the change in origin by

$$\hat{\mathbf{x}}_{new} = \hat{\mathbf{x}}_{old} - \begin{bmatrix} \hat{\mathbf{p}}_{il,j} \\ \vdots \\ \hat{\mathbf{p}}_{il,j} \end{bmatrix}. \tag{13}$$

It is easy to see that error in $\hat{\mathbf{d}}$ accumulates over time because imperfect estimates $\hat{\mathbf{p}}_{il,j}$ are continually added to it, however, this is true of many SLAM algorithms operating without some global measurement, such as GPS.

C. Sliding Window Observer Iteration

The nonlinear observer struggles to converge quickly on startup. Increasing the gain helps to a limited extent but causes a larger error in steady state, in addition to M growing rapidly beyond machine precision, due to the large error at startup when the gain is too large. Another possible method of improving convergence is to save all measurements and state data over a time window and guess the initial conditions iteratively, then select the guess that produced the least amount of error over the window. Guess-and-check methods are costly. Therefore, we could improve this by integrating backward in time from the end of the window to the beginning, use this as the new initial guess, re-run the observer over the window using the saved data, and repeat. The problem with this method is that measurement windows with insufficient excitation drive the estimates away from truth, so we improve upon this by allowing the window to slide with time. A sliding window gives us the ability to intelligently estimate the initial state iteratively, while continually using new measurements that have varying levels of excitation for each of the landmarks. Thus, in order to reduce the convergence time of the observer without introducing instability, we allow the observer to iterate on a sliding window of saved bearing measurements and velocities. This process is outlined in Algorithm 1 for a window of size S time steps.

In line 8 of Algorithm 1, we use Euler integration to propagate the position backward in time to the beginning of the sliding window. We do not backward integrate M and $\hat{\boldsymbol{\beta}}$. This is because $\hat{\boldsymbol{\beta}}$ depends on M and M must be a solution to the Ricatti equation given in (9), which means that M must also be positive definite. Backward integration of M is given by $M(t-1) = M(t) - I\Delta t$, which removes the positive definite guarantee on M. Therefore, we simply save the values of M and $\hat{\boldsymbol{\beta}}$ at the beginning of each window and use these initial values for forward integration. Another subtlety of this algorithm is that landmark estimates are not backward integrated. Recall that the observer is operating in the inertial frame, hence static landmark kinematics given in (3) do not change the landmark estimates.

Algorithm 1 Sliding Window Observer Iteration

1: **for** each t **do**
2: Save $\mathbf{y}_{0\ldots N-1}$ and $\hat{v}$ to history window.
3: **if** history size $> S$ **then**
4: Trim oldest values of $\mathbf{y}_{0\ldots N-1}$ and $\hat{v}$.
5: **end if**
6: **if** history size $= S$ **then**
7: **for** $j = 0$ to $T-1$ **do**
8: **for** $m = S-2$ to 0 **do**
9: $\hat{\mathbf{p}}^i_{ib} = \hat{\mathbf{p}}^i_{ib} - \hat{v}_m \Delta t$
10: **end for**
11: $M = M_{saved}$
12: $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{saved}$
13: **for** $n = 0$ to $S-2$ **do**
14: $M = M + (I - kM\Pi_n M)\,\Delta t$
15: $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}} + \left(-M\Pi_n M\hat{\boldsymbol{\beta}} - M\Pi_n\hat{\mathbf{x}}\right)\Delta t$
16: $\hat{\mathbf{x}} = \hat{\mathbf{x}} + \left(\begin{bmatrix}\hat{\mathbf{v}}_n \\ \mathbf{0}\end{bmatrix} - kM\Pi_n\hat{\mathbf{x}}\right)\Delta t$
17: **end for**
18: **end for**
19: $M_{saved} = M$
20: $\hat{\boldsymbol{\beta}}_{saved} = \hat{\boldsymbol{\beta}}$
21: **end if**
22: **end for**

The observer iteration procedure reduces convergence time of the observer but also accelerates observer divergence from truth when observability is low. In practice, observer iteration also hinders the observer's ability to track the velocity bias, which causes the velocity error to slowly grow, so iteration should only be used briefly on initialization and after origin changes. In general, observer iteration improves state error reduction, given a time window where observations are persistently exciting and velocity error is small. Because global, exponential stability of the nonlinear observer, assuming persistent excitation, is proven in [20], any set of initial conditions are driven toward the truth by the observer, resulting in less error at the end of a time window than at the beginning, given a large enough window. While the error in velocity is small over a small window, backward integration of the position results in a new initial condition with less error than the original initial condition.

IV. Simulation Results

To investigate the performance of the nonlinear observer and its modifications in a SLAM setting, we setup a multi-copter simulation, where the multi-copter uses a PID controller to fly a user-defined 3D path. Control inputs are computed based on the multi-copter's true state, so that we can easily compare results on the same observed state. This work is done in simulation prior to hardware implementation so that observer performance can easily be separated from the quality of data association of camera measurements. Therefore, the supplied measurements are created by computing unit vectors pointing to landmarks in the vehicle body frame. This is less realistic in that there is no simulated camera or restrictions from having a finite field of view, but we can still demonstrate the validity and shortcomings of the proposed methods in an ideal scenario. In the following simulations, the multi-copter takes off near the initial origin and flies a figure-eight path at a nearly constant altitude of ten meters. The simulated accelerometer has added Gaussian noise set to a standard deviation of 0.1 m/s^2 with a small, constant bias. It is assumed that a separate observer provides accurate attitude estimates, therefore we simply assume it is known. Sliding window iteration has parameters $T = 1$, $S = 20$, and only runs for two seconds after initialization and origin shifts. The landmark of the first origin is located at zero to simplify plotting.

The figures in this section demonstrate the performance of this observer, in addition to running it with sliding window iteration, origin shifting, and using it for target tracking in the inertial frame. Figs. 2 and 3 show the position and velocity estimation performance of the observer with and without sliding window iteration enabled. Fig. 4 is just one of ten inertial landmark estimates over time, shown as an example of landmark convergence. We see that sliding window iteration of the observer greatly reduces the convergence time on startup. Fig. 5 shows that the observer is able to track the general trend of the velocity bias over time with and without sliding window iteration.

Results for origin shifting are also shown in Figs. 2-4, where we shift the inertial origin to the location of the second landmark $\mathbf{p}^i_{il,1}$ every five seconds and replace $\mathbf{p}^i_{il,1}$ with a random, new landmark until the thirty second mark. We do this because there is no field of view constraint in the simulation, but we still need to show how the observer behaves when the origin changes. This results in small, sudden changes in the position and velocity estimates due to the poor initial estimation of the new landmark. In Fig. 4, we see that the new landmark's north and east estimates converge quickly, while covergence of the down estimate is delayed by a second or two. This happens because the observer projects the estimate onto the measurement direction, decreasing the landmark vector's length significantly, which is then increased as more observations arrive. An improved method of initializating landmark estimates would likely reduce this initial error, and due to the consensus style

of estimation, diminish the jumps in the other estimates. Additionally, origin drift is apparent in Fig. 4, but the drift of the iterated observer's estimates is reduced because of the fast convergence of landmark estimates.

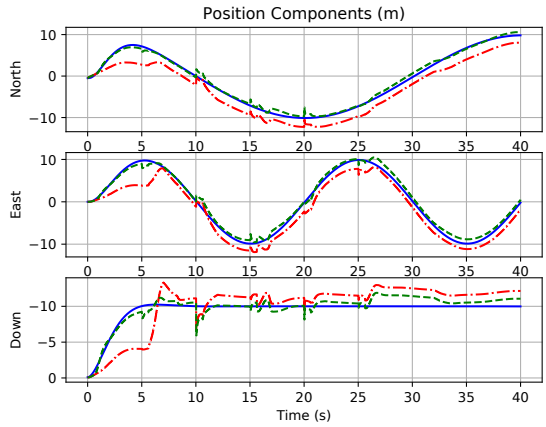

Fig. 2. Position components of the observer with origin shifting every five seconds compared against truth. Solid blue is ground truth, dash-dot red is the estimate without sliding window iteration, and dashed green is the estimate with sliding window iteration.

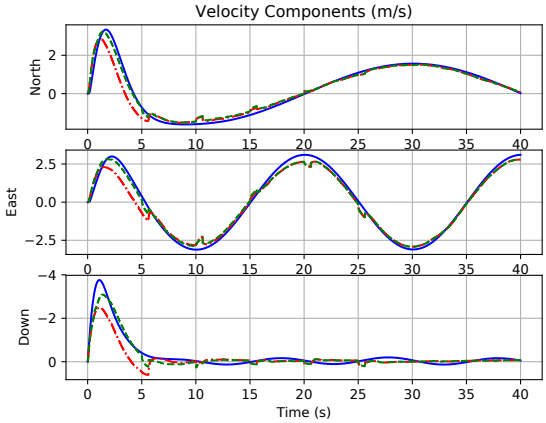

Fig. 3. Velocity components of the observer with origin shifting every five seconds compared against truth. Solid blue is ground truth, dash-dot red is the estimate without sliding window iteration, and dashed green is the estimate with sliding window iteration.

Finally, Fig. 6 shows the inertial position estimation of a maneuvering target over time using the planar approximation method presented in [23], where it is assumed that target bearing vector is measured. This requires the landmark estimates to be defined with respect to the vehicle body, which is accomplished for the j^{th} landmark by $\mathbf{p}^i_{bl,j} = \mathbf{p}^i_{il,j} - \mathbf{p}^i_{ib,j}$. This remains in the inertial frame and creates vectors pointing from the vehicle body to the landmarks. Now, we see from Fig. 6 that the tracker works about as well as the position estimation, being affected by the changing origin and like the other estimates, quickly converging toward the truth after each origin shift.

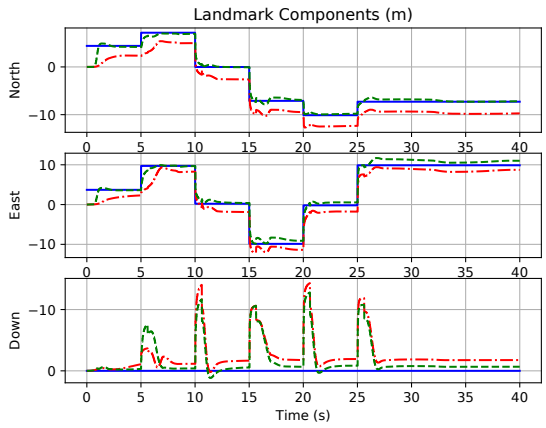

Fig. 4. Landmark position components of the observer with origin shifting every five seconds compared against truth. Solid blue is ground truth, dash-dot red is the estimate without sliding window iteration, and dashed green is the estimate with sliding window iteration.

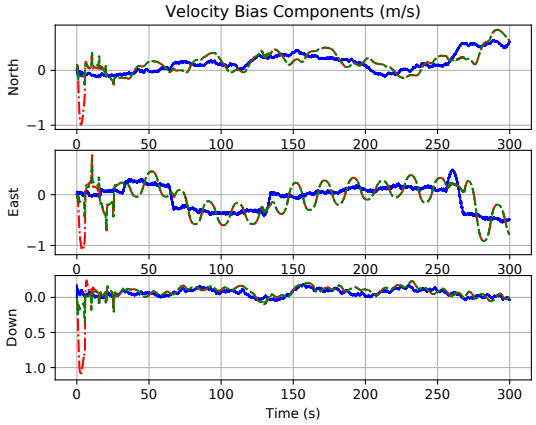

Fig. 5. Velocity bias components of the observer with and without sliding window iteration enabled on startup compared against truth. Solid blue is ground truth, dash-dot red is the estimate without sliding window iteration, and dashed green is the estimate with sliding window iteration.

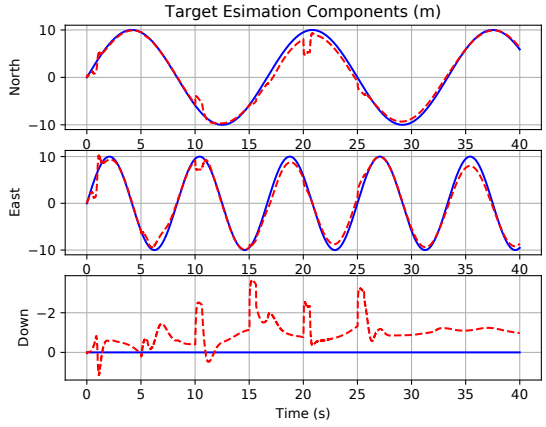

Fig. 6. Target position components with origin shifting every five seconds compared against truth using sliding window iteration. Solid blue is ground truth and dashed red is the estimate.

V. Conclusions

We presented two methods necessary for a practical implementation of the observer from [20], namely, origin shifting and sliding window observer iteration. Origin shifting suffered from large errors immediately after the change in origin, but due to the global stability of the observer and sliding window observer iteration, this error quickly diminishes. Smoothing the estimation during origin changes is to be carefully examined in future work. One possibility for smoothing is to use a second observer for the initial estimation new landmarks only, so that newly added landmarks don't negatively affect current position and landmark estimates. Sliding window iteration dramatically improves observer convergence speed during sequences of persistent excitation and is necessary for hardware implementation of this observer. We require further analysis of this method to better specify optimal conditions of its operation, such as how long it should run after newly established origins. Poor initial estimates combined with suboptimal observability of landmarks can cause observer iteration to degrade filter performance, therefore, detection of these scenarios should also be investigated. Lastly, we showed that this observer could be used to track a maneuvering ground target with accuracy comparable to its own position estimation.

References

[1] D. W. Murphy and J. Cycon, "Applications for mini VTOL UAV for law enforcement," in *Sensors, C3I, Information, and Training Technologies for Law Enforcement*, vol. 3577. International Society for Optics and Photonics, 1999, pp. 35–44.

[2] I. Hwang, H. Balakrishnan, K. Roy, and C. Tomlin, "Multiple-target tracking and identity management in clutter, with application to aircraft tracking," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 4. IEEE, 2004, pp. 3422–3428.

[3] P. J. Escamilla-Ambrosio and N. Lieven, "A multiple-sensor multiple-target tracking approach for the autotaxi system," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 601–606.

[4] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, 2007.

[5] B. A. Jones, D. S. Bryant, B.-T. Vo, and B.-N. Vo, "Challenges of multi-target tracking for space situational awareness," in *Information Fusion (FUSION), 2015 18th International Conference on*. IEEE, 2015, pp. 1278–1285.

[6] D. Reid, "An algorithm for tracking multiple targets," *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[7] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*. IEEE, 1980, pp. 807–812.

[8] B.-N. Vo and W.-K. Ma, "The Gaussian mixture probability hypothesis density filter," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[9] P. C. Niedfeldt, "Recursive-RANSAC: A novel algorithm for tracking multiple targets in clutter," Ph.D. dissertation, 2014, available at http://scholarsarchive.byu.edu/etd/4195/.

[10] P. C. Niedfeldt, K. Ingersoll, and R. W. Beard, "Comparison and analysis of recursive-RANSAC for multiple target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 1, pp. 461–476, 2017.

[11] P. C. Niedfeldt and R. W. Beard, "Convergence and complexity analysis of recursive-RANSAC: a new multiple target tracking algorithm," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 456–461, 2016.

[12] K. Ingersoll, P. C. Niedfeldt, and R. W. Beard, "Multiple target tracking and stationary object detection in video with recursive-RANSAC and tracker-sensor feedback," in *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*. IEEE, 2015, pp. 1320–1329.

[13] P. C. Niedfeldt and R. W. Beard, "Multiple target tracking using recursive RANSAC," in *American Control Conference (ACC), 2014*. IEEE, 2014, pp. 3393–3398.

[14] F. Yang, W. Tang, and H. Lan, "A density-based recursive RANSAC algorithm for unmanned aerial vehicle multi-target tracking in dense clutter," in *Control & Automation (ICCA), 2017 13th IEEE International Conference on*. IEEE, 2017, pp. 23–27.

[15] J. Y. Sakamaki, R. W. Beard, and M. D. Rice, "Cooperative estimation for a vision-based target tracking system," in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*. IEEE, 2016, pp. 878–885.

[16] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[17] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[18] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 298–304.

[19] A. Barrau and S. Bonnabel, "An ekf-slam algorithm with consistency properties," *arXiv preprint arXiv:1510.06263*, 2015.

[20] F. Le Bras, T. Hamel, R. Mahony, and C. Samson, "Observers for position estimation using bearing and biased velocity information," in *Sensing and Control for Autonomous Vehicles*. Springer, 2017, pp. 3–23.

[21] T. A. Johansen and E. Brekke, "Globally exponentially stable Kalman filtering for SLAM with AHRS," in *Information Fusion (FUSION), 2016 19th International Conference on*. IEEE, 2016, pp. 909–916.

[22] D. P. Koch, D. O. Wheeler, R. W. Beard, T. W. McLain, and K. M. Brink, "Relative multiplicative extended Kalman filter for observable GPS-denied navigation," 2017, available at http://scholarsarchive.byu.edu/facpub/1963/.

[23] J. Nielsen and R. W. Beard, "Relative target estimation using a cascade of extended Kalman filters," in *Proceedings of the 30th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2017)*, Portland, Oregon, September 2017, pp. 2273–2289.

[24] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on automatic control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[25] G. Allibert, R. Mahony, and M. Bangura, "Velocity aided attitude estimation for aerial robotic vehicles using latent rotation scaling," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1538–1543.