AAS 23-025

BENCHMARKING STATE ESTIMATION ALGORITHMS FOR OUTPUT FEEDBACK MODEL PREDICTIVE CONTROL OF SPACECRAFT AUTONOMOUS RENDEZVOUS AND DOCKING*

An Dang,† Sean A. Phillips,‡ and David A. Copp†

We present multiple state estimation algorithms that can be used in spacecraft autonomous rendezvous, proximity operations, and docking (ARPOD) missions and compare the performance of the resulting closed-loop systems when these estimation algorithms are combined with model predictive control (MPC). We consider a passive target object and a chaser spacecraft. The system dynamics are described using standard nonlinear equations for spacecraft relative motion on orbit. The sensing capability of the chaser spacecraft depends on its proximity to the target: at farther distances, only relative angle measurements are available; at closer distances, both relative angle and range measurements are available. In all cases, the measurements are nonlinear functions of the system's states, in this case relative positive and velocity. Moreover, the measurements are noisy, and the system dynamics have process disturbances. Therefore, state estimation algorithms are required to construct state estimates from these erroneous, noisy output feedback measurements. Specifically, we present and compare Extended Kalman Filter (EKF), Particle Filter (PF), and Moving Horizon Estimation (MHE) algorithms and analyze resulting fuel consumption, mission time, state estimation error, and computation time when solving an ARPOD benchmark problem with MPC paired with each of these state estimation algorithms. The tradeoffs between computation time and performance are clear: the EKF approach requires the least computation time but results in the worst overall performance, the MHE approach requires the most computation time but produces the best results, and the PF approach performs in the middle for both computation time and performance. The particular impact of distance to the target and process noise/disturbances on the performance are presented for each state estimation algorithm.

INTRODUCTION

Recently, it has become significantly easier to access the space domain. This is due, in large part, to more launch options both domestically and abroad. This ease of access also opens the potential for emerging space applications, from commercial constellation-level applications like global internet service providers and imaging for ground navigation applications to proximity operations like in-space servicing and manufacturing, or identification and removal of orbital debris. These complex operations will require a large amount of operational burden and may require increasing spacecraft autonomy to alleviate this burden. Recently many researchers in government labs have released space-trusted autonomy readiness levels to apply to advanced satellite operations.[1] These applications also require safety and robustness to model errors, disturbances, and sensor

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED. PUBLIC AFFAIRS APPROVAL #AFRL-2023-1823.

†University of California, Irvine. Irvine, CA 92697, USA.

‡Space Vehicles Directorate, Air Force Research Laboratory, Kirtland AFB, NM 87117, USA. The views expressed are those of the authors and do not reflect the official guidance or position of the United States Government, the Department of Defense or of the United States Air Force.

noise.[2,3] These challenges emphasize the need for advanced estimation and control techniques for autonomous spacecraft.

In this work, we consider the problem of a spacecraft performing an Autonomous Rendezvous, Proximity Operations, and Docking (ARPOD) mission. We follow the spacecraft benchmark problem formulation proposed in (Ref. 4) and specifically consider a chaser spacecraft tasked with rendezvous and docking with a passive (non-actuated) target spacecraft on orbit. The different phases of an ARPOD mission are shown in Figure 1. During the ARPOD mission, the sensing capability of the chaser spacecraft depends on its proximity to the target. At further distances (Phase 1), only relative angle (or bearing) measurements are available. At closer distances (Phases 2 and 3), both relative angle and range measurements are available. In all cases, these measurement equations are nonlinear functions of the system's states, which include relative positive and velocity. Moreover, the measurements may be noisy, and the system may experience process noise/disturbances. Therefore, state estimation algorithms are required to construct state estimates from these erroneous, noisy output feedback measurements. Then, using these state estimates, a feedback controller can be designed to calculate the actuation required to complete the mission.

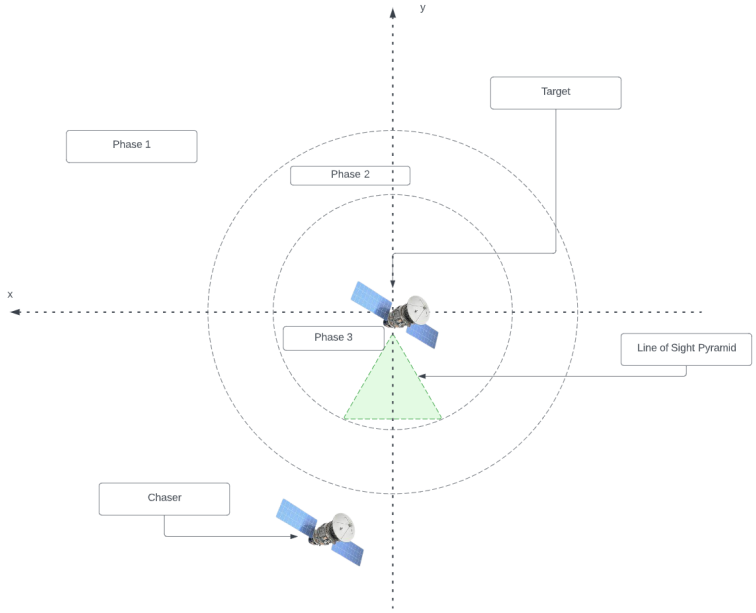

Figure 1 Depiction of ARPOD mission phases. Phase 1 of the mission is the initial phase of the rendezvous maneuver when only relative angle measurements are available. Phase 2 involves the rendezvous maneuver when both relative angle and range measurements are available within 1 km of the target. Phase 3 is the docking phase and starts within 100 m of the target.

In this paper, we present several state estimation algorithms that enable effective control of spacecraft performing the ARPOD mission set and provide an in-depth comparison of the resulting estimation performance. Furthermore, we couple these estimation techniques with a control algorithm, namely, a model predictive control (MPC) algorithm, and evaluate the closed-loop performance. MPC is a finite-horizon online optimization approach for feedback control that is popular in numerous applications because it can explicitly accommodate constraints on the system dynamics, states,

and inputs.[5] MPC has been applied in several spacecraft applications,[6] and particularly in problems involving spacecraft relative motion.[7–13] The Extended Kalman Filter (EKF) (see, e.g., (Ref. 14)) is the most popular state estimator to be paired with MPC for spacecraft relative motion (see, e.g., (Ref. 4, 9, 15)), however, it is possible to use other state estimators, such as a Particle Filter (PF)[16] or Moving Horizon Estimation (MHE).[17] There are advantages and disadvantages to each of these estimators, and an evaluation of EKF versus MHE is given by (Ref. 18).

We design multiple state estimation algorithms for this spacecraft rendezvous and docking problem and compare their performance when used with a benchmark MPC framework for spacecraft relative motion, described in (Ref. 8). We perform a rigorous numerical analysis wherein we simulate a chaser spacecraft performing autonomous rendezvous and docking with each of the state estimators (EKF, PF, and MHE) using output feedback nonlinear measurements and considering measurement noise and process noise/disturbances. We perform numerous simulations of the closed-loop system comprising each state estimator combined with the MPC algorithm for the following three cases:

Case 1) Chaser spacecraft starts in Phase 1. There is additive noise in the sensor measurements.

Case 2) Chaser starts in Phase 1. There are additive process disturbances and measurement noise.

Case 3) Chaser starts in Phase 2. There are additive process disturbances and measurement noise.

For each case, we compare the resulting performance of using each state estimator in terms of estimation error, computation time, mission time, and fuel consumption.

PROBLEM FORMULATION

We consider a passive (non-actuated) target spacecraft that is orbiting in a circular Keplerian orbit. Then the equations for relative motion between an active chaser spacecraft and the passive target are[19]

$$\begin{aligned} \ddot{x} - 2n\dot{y} - n^2(R+x) + \mu\frac{R+x}{((R+x)^2+y^2+z^2)^{\frac{3}{2}}} &= u_x \\ \ddot{y} + 2n\dot{x} - n^2 y + \mu\frac{y}{((R+x)^2+y^2+z^2)^{\frac{3}{2}}} &= u_y \\ \ddot{z} + \mu\frac{z}{((R+x)^2+y^2+z^2)^{\frac{3}{2}}} &= u_z. \end{aligned} \tag{1}$$

These nonlinear translational spacecraft dynamics describe the chaser spacecraft's position and velocity in a local-vertical/local-horizontal (LVLH) frame with reference fixed on the center of gravity of the target spacecraft. The states $\mathbf{x} := \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^\top \in \mathbb{R}^6$ include the spacecraft's position and velocity. The state x denotes the position in the radial direction (from earth), also known as the cross-track direction, y denotes the position in the in-track direction, and z denotes the position in the cross-track direction that completes the right-hand coordinate system with x and y. We consider six continuously variable thrusters that provide thrust in the positive and negative direction of each dimension and define the thrust actuation control inputs as $\mathbf{u} := \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^\top \in \mathbb{R}^3$. R denotes the orbit radius of the target spacecraft, $n = \sqrt{\mu/R^3}$ is the angular speed, or "mean motion," of the target through its orbit, and μ is the Earth's gravitational constant.

When the chaser is close to the target, the nonlinear spacecraft relative motion dynamics (1) can be linearized around the target's position, which gives the following Hill-Clohessy-Wiltshire (HCW) equations[20,21]

$$\ddot{x} - 3n^2x - 2n\dot{y} = u_x, \qquad \ddot{y} + 2n\dot{x} = u_y, \qquad \ddot{z} + n^2z = u_z.$$

These continuous linear time-invariant dynamics can be written in a state-space form as $\dot{\mathbf{x}} = A_c\mathbf{x} + B_c\mathbf{u}$. We can convert these continuous-time dynamics to the following discrete-time linear time-invariant dynamics

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{d}_t, \tag{2}$$

where $\mathbf{x}_t$ and $\mathbf{u}_t$ denote the states and control inputs at time step t, respectively, and $\mathbf{d}_t := \begin{bmatrix} d_x & d_y & d_z & d_{\dot{x}} & d_{\dot{y}} & d_{\dot{z}} \end{bmatrix}^\top \in \mathbb{R}^6$ is an unknown vector that accounts for possible additive disturbances at time step t. These disturbances may capture actuator uncertainty due to errors or misalignment or may capture unmodeled dynamics or process noise. We assume these disturbances are normally distributed random variables such that $\mathbf{d}_t \sim \mathcal{N}(\bar{\mathbf{d}}, \sigma_d)$ for all times t.

The conversion to discrete time is done using the state transition matrix such that $A = \exp^{hA_c}$, where h is the sampling time step. The discrete-time matrix B depends on the type of control considered. If control inputs are constant for each sample time interval h, as they are in this work, then $B = A_cB_c$. Therefore, A and B are given as

$$A = \begin{bmatrix} 4 - 3\cos(nh) & 0 & 0 & (1/n)\sin(nh) & (2/n)(1-\cos(nh)) & 0 \\ 6(\sin(nh) - nh) & 1 & 0 & (2/n)(\cos(nh)-1) & (1/n)(4\sin(nh) - 3nh) & 0 \\ 0 & 0 & \cos(nh) & 0 & 0 & (1/n)\sin(nh) \\ 3n\sin(nh) & 0 & 0 & \cos(nh) & 2\sin(nh) & 0 \\ 6n(\cos(nh)-1) & 0 & 0 & -2\sin(nh) & 4\cos(nh) - 3 & 0 \\ 0 & 0 & -n\sin(nh) & 0 & 0 & \cos(nh) \end{bmatrix},$$

$$B = \begin{bmatrix} (1/n^2)(1-\cos(nh) & (1/n^2)(2nh - 2\sin(nh)) & 0 \\ (1/n^2)(2(\sin(nh) - nh) & -3h^2/2 + (4/n^2)(1-\cos(nh)) & 0 \\ 0 & 0 & (1/n^2)(1-\cos(nh)) \\ \sin(nh)/n & (2/n)(1-\cos(nh)) & 0 \\ (2/n)(\cos(nh)-1) & -3h + (4/n)\sin(nh) & 0 \\ 0 & 0 & \sin(nh)/n \end{bmatrix}.$$

Measurement Model

With each of the state estimation algorithms we assume only output feedback is available given by the following nonlinear measurement models (see, e.g., Ref. 4). The measurement model when the chaser has only relative angular measurements to the target (i.e., bearings-only) is defined as

$$\mathbf{y}_t = g(\mathbf{x}_t) = \begin{bmatrix} \alpha_t \\ e_t \end{bmatrix} + \mathbf{v}_t = \begin{bmatrix} \arctan(y_t/x_t) \\ \arcsin(z_t/\rho_t) \end{bmatrix} + \mathbf{v}_t, \tag{3}$$

where $\rho_t \in \mathbb{R}$ denotes the magnitude of the relative displacement vector $[x_t\ y_t\ z_t]^\top$ at time t, i.e., $\rho_t := \| \ [x_t\ y_t\ z_t]^\top \ \| = \sqrt{x_t^2 + y_t^2 + z_t^2}$, the vector $\mathbf{v}_t \in \mathbb{R}^2$ denotes measurement noise at time t, $\alpha_t \in \begin{bmatrix} -\pi & \pi \end{bmatrix}$ denotes the horizontal (or azimuth) angle from the chaser to the target, and $e_t \in \begin{bmatrix} -\pi & \pi \end{bmatrix}$ denotes the elevation angle from the chaser to the target. We assume the measurement noise are normally distributed random variables such that $\mathbf{v}_t \sim \mathcal{N}(\bar{\mathbf{v}}, \sigma_v)$ for all times t. This is the measurement model for Phase 1 of the ARPOD mission, which means the spacecraft's state is only partially observable during this initial phase.

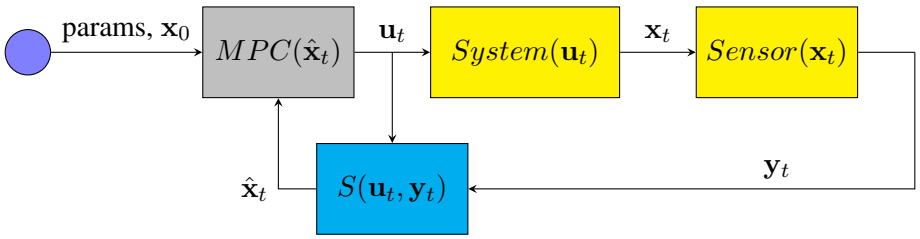

Figure 2 Block diagram of the closed-loop system.

If a range measurement is also available (i.e., in Phase 2 of the ARPOD mission), then the measurement model is defined as

$$\mathbf{y}_t = g(\mathbf{x}_t) = \begin{bmatrix} \alpha_t \\ e_t \\ \rho_t \end{bmatrix} + \mathbf{v}_t. \tag{4}$$

In (4), $\mathbf{v}_t \in \mathbb{R}^3$ denotes measurement noise on both angle and range measurements at time t.

Therefore, for the rendezvous and docking mission, the problem is to find control inputs $\mathbf{u}_t$ that drive the state of the chaser spacecraft $\mathbf{x}_t$ to zero (i.e., the origin of the LVLH reference frame centered at the target's position) given noisy nonlinear angle and range measurements, as in (3) and (4), and in the presence of disturbances $\mathbf{d}_t$. In the examples to follow, we use the linear model (2) to design the control and estimation algorithms, but we simulate the nonlinear dynamics (1).

STATE ESTIMATION ALGORITHMS

We consider the following three state estimation algorithms: Extended Kalman Filter (EKF), Particle Filter (PF), and Moving Horizon Estimation (MHE). Each of these is combined with a benchmark MPC algorithm resulting in feedback controlled motion of the chaser spacecraft for the ARPOD mission. A block diagram of the closed-loop system is shown in Figure 2, where $S(\mathbf{u}_t, \mathbf{y}_t)$ denotes the state estimation algorithm, and $\hat{\mathbf{x}}_t$ denotes the state estimate at time t. The EKF has been used extensively in the spacecraft relative motion literature. The PF (Bayesian filtering) has the advantage of working with nonlinear systems and approximating non-Gaussian distributions. MHE has similar advantages to the PF and also explicitly handles constraints.

For a fair comparison of these estimation algorithms, each use the dynamics (2) and have the same information about noise and disturbances. For example, the process disturbance covariances are assumed to be known and are used in the formulation of each of these algorithms. These algorithms are each paired with the same MPC algorithm and are given the same initial conditions and mission parameter values. The differences in state estimation performance result in different trajectories for the chaser spacecraft. To compare these differences, we perform numerous simulations of each closed-loop system with varying initial conditions and analyze resulting performance in terms of state estimation error, computation time, spacecraft fuel consumption, and mission time.

Next we describe the design of each of these three estimation algorithms for use in the ARPOD mission.

Extended Kalman Filter (EKF)

The Kalman Filter is a well-known state estimation algorithm that utilizes the fusing of system dynamics and sensor measurements to estimate a system's state. It is the optimal, unconstrained, linear state estimator, so it requires that the system dynamics are linear with possible additive Gaussian disturbances $\mathbf{d}_t$ as in the dynamics (2), and that the output equation is linear with possible additive Gaussian noise $\mathbf{v}_t$ in the sensor measurements. Because the output equations (3) and (4) are nonlinear, the Kalman filter cannot be used. Instead, we use the EKF, which first linearizes the nonlinear system and then applies the Kalman filter to obtain the state estimates. The EKF can be formulated as described in, e.g., (Ref. 14).

Particle Filter (PF)

The Kalman Filter represents state estimates as Gaussian random variables where the mean is the model and the covariance is its Gaussian noise. In contrast, the PF can accommodate arbitrary distributions by using a form of importance sampling with a large quantity of weighted particles. Using the ideas of importance sampling, with enough particles, the PF can take a baseline distribution that we give it and use that baseline distribution to closely estimate the true distribution.

One of the main benefits of the PF is its ability to handle nonlinear dynamics and non-Gaussian noise. With an arbitrarily large number of particles, the PF is able theoretically to estimate the state perfectly, which gives it the ability to achieve high accuracy. A drawback of the PF is its high computation requirements, which increases with the number of particles. The computation required for acceptable accuracy from a PF may be achievable with modern computing capabilities. Moreover, its computations can be parallelized to speed up processing and make it applicable for embedded systems that include GPU technology.

For this spacecraft application, we follow the PF formulation described in (Ref. 16).

Moving Horizon Estimation (MHE)

MHE involves the solution of a finite horizon online optimization problem in which an objective function is minimized given a sequence of past measurements and the system dynamics. The optimization problem is solved at each time step when a new measurement is available, and the oldest measurement in the finite window of past measurements is forgotten. Like the PF, MHE has the advantage that it can accommodate nonlinear systems and arbitrary distributions, and in addition, it can accommodate constraints. The main drawback is that it can be computationally expensive to solve an optimization problem at each time step.

For the spacecraft rendezvous and docking problem, we can formulate the MHE optimization problem

$$\min_{\hat{\mathbf{x}}_{t-L:t}} \sum_{k=t-L}^{t-1} (\hat{\mathbf{x}}_{k+1} - (A\hat{\mathbf{x}}_k + B\mathbf{u}_k))^\top Q_{\text{MHE}} (\hat{\mathbf{x}}_{k+1} - (A\hat{\mathbf{x}}_k + B\mathbf{u}_k)) + (\mathbf{y}_k - g(\hat{\mathbf{x}}_k))^\top R(\mathbf{y}_k - g(\hat{\mathbf{x}}_k)) \quad (5)$$

which is unconstrained due to the incorporation of the linear dynamics (2) and the nonlinear measurement function (3) or (4) (depending on the phase of the mission) into the objective. The diagonal weighting matrices Q_{MHE} and R can be designed relative to the noise covariances, and L is the

length of the finite backward horizon. This problem (5) is solved online at each sampling time step t in a receding horizon fashion, and the solution is the sequence of past state estimates $\hat{\mathbf{x}}_{t-L:t}$ that minimizes the objective.

MPC FOR SPACECRAFT RENDEZVOUS AND DOCKING

We pair each of the state estimation algorithms described in the State Estimation Algorithms Section with a benchmark MPC algorithm, which is formulated as in (Ref. 8) and summarized as follows. The resulting closed-loop system is shown in Figure 2. MPC involves the solution of a finite horizon online optimization problem that finds a sequence of future control actions that minimize an objective function, given an estimate of the state at the current time. The objective of the MPC problem for rendezvous and docking is to control all of the chaser spacecraft's states (i.e., position and velocity) to zero in the LVLH reference frame that is centered at the target's position while minimizing fuel consumption. Therefore, we can define the objective function to be minimized as the following quadratic function that penalizes nonzero states (to drive the chaser to the target) and nonzero control inputs (to penalize fuel consumption)

$$J_t(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{t:t+T-1}) = \sum_{k=t}^{t+T} \hat{\mathbf{x}}_k^\top Q \hat{\mathbf{x}}_k + \hat{\mathbf{u}}_k^\top R_u \hat{\mathbf{u}}_k, \tag{6}$$

where T is the number of time steps in the finite forward horizon, and Q and R_u are diagonal weighting matrices. The optimization variables are the sequence of future control inputs $\hat{\mathbf{u}}_{t:t+T-1} := \{\hat{\mathbf{u}}_t,\ \hat{\mathbf{u}}_{t+1},\ \ldots,\ \hat{\mathbf{u}}_{t+T-1}\}$. Our goal, given a current estimate of the state $\hat{\mathbf{x}}_t$, is to choose these optimization variables to minimize the objective function (6) while satisfying several constraints, including satisfying the dynamics (2) (without the additive disturbance $\mathbf{d}_t$). We assume a constant thrust is applied continuously during each time step and that there is a maximum thrust $\bar{u}$, which results in the constraint

$$\|\mathbf{u}_t\|_\infty \leq \bar{u}. \tag{7}$$

We define an additional constraint as the chaser gets near to the target so that it can stage itself for docking by entering a line-of-sight (LoS) region for advanced sensors that are available in Phase 3 of the mission. We formulate this LoS constraint as a linear pyramid constraint in 3D as[4]

$$\begin{bmatrix} \sin\frac{\theta_1(k)}{2} & \cos\frac{\theta_1(k)}{2} & 0 \\ \sin\frac{\theta_1(k)}{2} & -\cos\frac{\theta_1(k)}{2} & 0 \\ \sin\frac{\theta_2(k)}{2} & 0 & \cos\frac{\theta_2(k)}{2} \\ \sin\frac{\theta_2(k)}{2} & 0 & -\cos\frac{\theta_2(k)}{2} \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

Finally, we apply a terminal velocity constraint defined as

$$\|[0_{3\times 3}\ \ I_{3\times 3}]\mathbf{x}_T\|_2 \leq \bar{V}, \tag{9}$$

where $\bar{V}$ is the maximum velocity in m/s.

Given the objective (6) and the variables and constraints described above, we solve the following optimization problem at each time t in a receding horizon fashion

$$\begin{aligned} &\min_{\hat{\mathbf{u}}_{k:k+T-1}} \quad J_t(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_{t:t+T-1}) \\ &\text{subject to (2), (7), (8), (9).} \end{aligned} \tag{10}$$

The solution to the optimization problem (10) is the sequence of future control inputs that minimize the objective function and is denoted as $\hat{\mathbf{u}}^*_{t:t+T-1} := \{\hat{\mathbf{u}}^*_t,\ \hat{\mathbf{u}}^*_{t+1},\ \ldots,\ \hat{\mathbf{u}}^*_{t+T-1}\}$. At each time step t the first element of the optimal control sequence $\hat{\mathbf{u}}^*_{t:t+T-1}$ is applied to the system, resulting in the control law

$$\mathbf{u}_t = \hat{\mathbf{u}}^*_t.$$

In order to successfully transition from Phase 2 to Phase 3 of the ARPOD mission, the MPC algorithm must be able to maneuver the chaser spacecraft into the LoS region before it is within 100 m (the radius of Phase 3) from the target. Incorporating a state constraint that requires the chaser's position to be within the LoS at the end of Phase 2 is an option, but it may lead to challenges with feasibility of the MPC optimization problem. The way we handle this challenge in this work is to, rather than drive the state to the origin, drive the state to a position at the center of the LoS region while in Phase 2. This can be done by editing the $\hat{\mathbf{x}}_k^\top Q \hat{\mathbf{x}}_k$ term in the objective function.

NUMERICAL EXAMPLES

In this section, we compare the results of simulating each of the estimation algorithms with the benchmark MPC for the spacecraft rendezvous and docking problem. Each simulation involves solving the nonlinear relative motion dynamics (1) with the nonlinear measurements (3) and (4), state estimates from a state estimation algorithm described in the State Estimation Algorithms Section above, and MPC as described in the MPC Section above. All simulations use the system parameters given in Table 1, the MPC parameters given in Table 2, and the specific estimation algorithm parameter values given in Table 3. The true disturbance and noise covariances given in Table 1 are known and are used in the design of the estimation algorithms, and the noise covariances are updated in each phase according to the values used in (Ref. 4). All simulations are run on a machine with a 4.7 GHz 16-Core AMD Ryzen 6800HS processor and use MATLAB's `ODE45` function to solve the nonlinear dynamics of the chaser spacecraft (i.e., (1)).

Table 1 System parameters.

Parameter	Description	Value	Units
μ	Earth's gravitational constant	3.986×10^{14}	m^3/s^2
R	Geostationary orbit semi-major axis	$35.786 \times 10^6 + r$	m
m	Mass of chaser spacecraft	500	kg
$\bar{u}$	Maximum thrust	10	N
h	Sampling time step	1	s
$\bar{\mathbf{d}}$	Mean process noise/disturbance value	$0_{6\times1}$	km, km/s
$\bar{\mathbf{v}}$	Mean measurement noise value	$0_{3\times1}$	rad, km
$\sigma_{d,x}$	Position disturbance covariance	1×10^{-3}	km^2
$\sigma_{d,\dot{x}}$	Velocity disturbance covariance	1×10^{-10}	$(km/s)^2$
σ_α	Angle measurement covariance	1×10^{-3}	rad^2
σ_e	Angle measurement covariance	1×10^{-3}	rad^2
$\sigma_{\rho 2}$	Phase 2 distance measurement covariance	1×10^{-2}	km^2
$\sigma_{\rho 3}$	Phase 3 distance measurement covariance	1×10^{-5}	km^2

In order to benchmark the algorithms, we run numerous simulations with randomly varying initial conditions using each algorithm and compare the total fuel consumption, algorithm computation time, mission time, and root mean squared error (RMSE) of the state estimates. We simulate the

Table 2 MPC Parameters.

Parameter	Description	Value	Units
T	Forward Horizon	10	-
$Q_{\mathbf{x}}$	Position weighting matrix	$100 \cdot I_{3\times3}$	-
$Q_{\dot{\mathbf{x}}}$	Velocity weighting matrix	$10000 \cdot I_{3\times3}$	-
$R_{\mathbf{u}}$	Input weighting matrix	$I_{3\times3}$	-

Table 3 Estimation Algorithm Parameters.

EKF Parameter	Description	Value	Units
Σ_0	Initial state covariance	1×10^{-50}	km^2
Q_d	Process disturbance covariance	1×10^{-20}	km^2
R_y	Measurement covariance	1×10^{-3}	rad^2
PF Parameter	**Description**	**Value**	**Units**
n	Number of particles	1000	-
Q_d	Process disturbance covariance	1×10^{-20}	km^2
$R_{\alpha,e}$	Measurement angle covariance	1×10^{2}	rad^2
R_ρ	Measurement distance covariance	1×10^{3}	km^2
S	Effective sampling size threshold	700	-
MHE Parameter	**Description**	**Value**	**Units**
L	Backward horizon length	10	-
Q_{MHE}	Process disturbance covariance weight	1×10^{20}	km^2
$R_{\alpha,e}$	Angle measurement covariance weight	1×10^{3}	rad^2
R_ρ	Distance measurement covariance weight	1×10^{2}	km^2

closed-loop system with each state estimation algorithm 100 times in three different Cases to investigate the impact of starting distance from the target and measurement and process noise. Thus, we run a total of 300 simulations per algorithm. The Cases are described as follows.

Case 1 involves simulation of the entire ARPOD mission from Phase 1 through Phase 3. It only includes measurement noise $\mathbf{v}_t$ and does not include process noise (i.e., $\mathbf{d}_t = 0$ for all t). The initial conditions are uniformly sampled from positions from -5 km to 5 km and velocities from -1 m/s to 1 m/s.

Case 2 is the same as Case 1 but with the addition of non-zero process noise $\mathbf{d}_t$ as described in 1, which significantly affects the performance of all of the state estimators.

Case 3 is the same as Case 2, but it starts the chaser spacecraft in a position in Phase 2, so angle and range measurements are both available from the start. For this case, the initial conditions are uniformly sampled from positions from -1 km to 1 km and velocities from -0.01 m/s to 0.01 m/s.

As described in Table 1, we model the additive noise as zero-mean Gaussian random variables. Thus $\bar{\mathbf{d}} = 0$ and $\bar{\mathbf{v}} = 0$, and the covariances are described as Σ_p for the process noise covariance, Σ_{s_1} for the Phase 1 sensor noise covariance, Σ_{s_2} for the Phase 2 sensor noise covariance, and Σ_{s_3}

for the Phase 3 sensor noise covariance:

$$\begin{aligned}\Sigma_p &= \text{diag}[\sigma_{d,x}, \sigma_{d,x}, \sigma_{d,x}, \sigma_{d,\dot{x}}, \sigma_{d,\dot{x}}, \sigma_{d,\dot{x}}] \\ \Sigma_{s_1} &= \text{diag}[\sigma_\alpha, \sigma_e] \\ \Sigma_{s_2} &= \text{diag}[\sigma_\alpha, \sigma_e, \sigma_{\rho 2}] \\ \Sigma_{s_3} &= \text{diag}[\sigma_\alpha, \sigma_e, \sigma_{\rho 3}]\end{aligned}$$

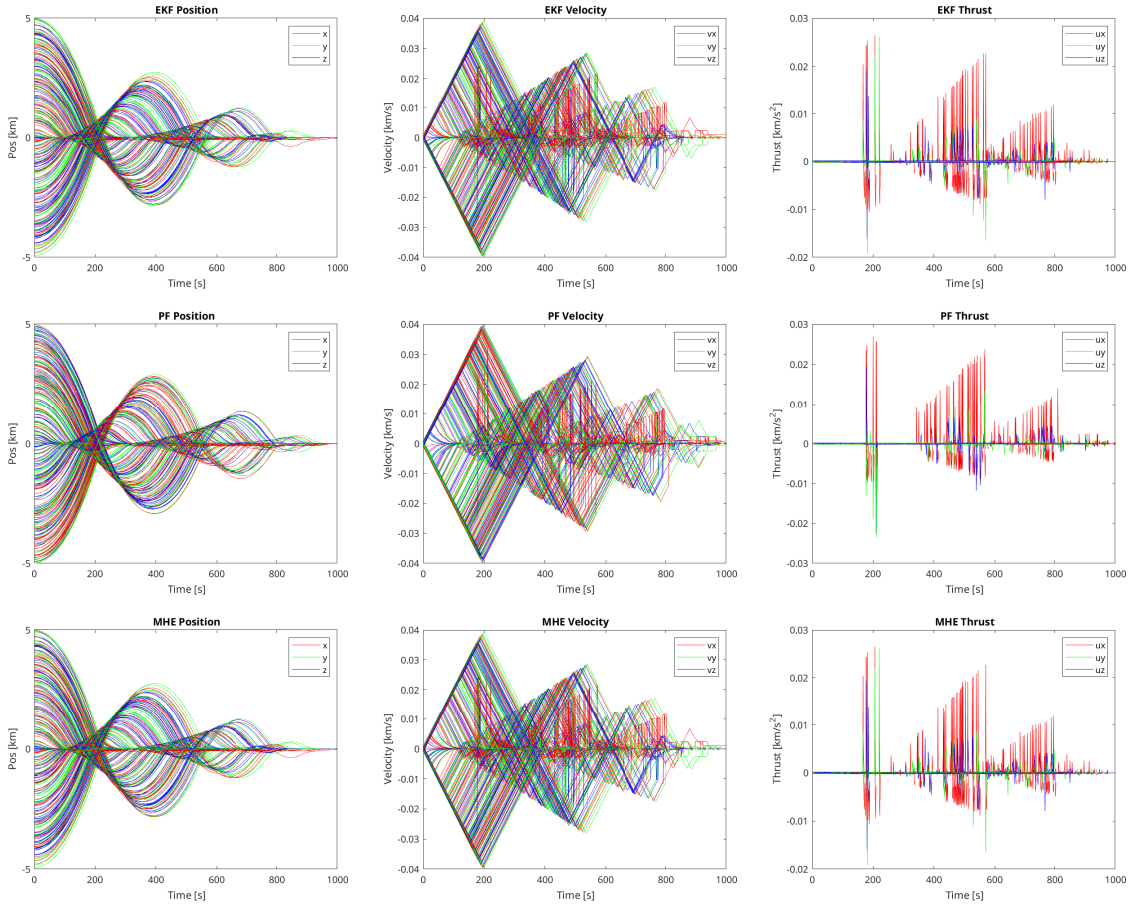

Figure 3 Simulation data for Case 1. The chaser position (left column), velocity (middle column), and thrust (right column) are shown for 100 simulations with the EKF (top row), PF (middle row), and MHE (bottom row).

The simulations highlight the tradeoff between computation time and performance, including estimation accuracy and fuel consumption. The Case 1 results are shown in Figures 3, 4, and 5 and Table 4. The statistics for the performance metrics shown in Figure 5 and Table 4 are calculated for all 100 simulations of each state estimator and are broken down by the phase of the mission. The estimation error is calculated as the root mean squared difference between the state estimates and the true state at each time step. The computation time (or runtime) is calculated as the time it takes to solve for the state estimate at each time step (i.e., it does not include the computation time to solve the MPC problem). Then the statistics are computed over all time steps in each mission phase for all 100 simulations.

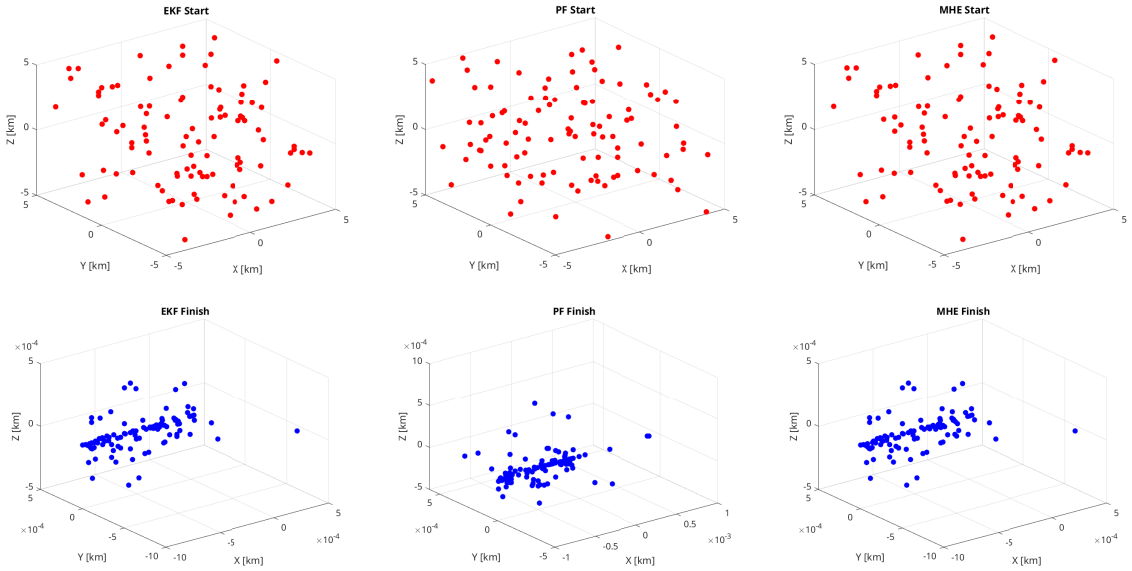

Figure 4 Mission start (top row) and finish (bottom row) locations for all 100 simulations in Case 1 with the EKF (left column), PF (middle column), and MHE (right column). Note the difference in axis scaling in the bottom plots.

For Case 1, the performance differences between the algorithms, in terms of estimation accuracy and fuel consumption, are minimal, but there is a difference in computation time. The EKF requires the least computation time while the PF requires the most computation time. This case gives a baseline for the results in Cases 2 and 3. Because of the nature of the EKF and the PF, the runtime metric of these two is relatively consistent throughout all of the simulations. The runtime for MHE, on the other hand, increases with increased computational complexity, such as including process noise as in Cases 2 and 3.

The results for Case 2 are shown in Figures 6, 7, and 8 and Table 5. These results show the increased MHE runtime as compared to the EKF and the PF but also the better performance of the MHE in terms of estimation accuracy and fuel consumption as compared to the other estimators. Note, using the EKF does not always result in successful docking, so the fuel consumption may be irrelevant if the mission was not completed. While the PF and EKF runtimes stay relatively the same as in Case 1, the MHE median runtime overall increases by 70ms per timestep. This is due to the formulation of the dynamics in the MHE optimization problem (5). The addition of process noise causes the MHE to require more iterations to converge. The process noise/disturbance also affects the other state estimators in different ways. The EKF performance is heavily impacted by the process disturbance, which can be seen in Figure 6. The EKF graphs show high variance and diverging state values over time. In Figure 8, the estimation error of the EKF is much larger than the other state estimators with the exception of Phase 3. The PF has much higher fuel consumption compared to MHE during Phases 2 and 3. The median fuel consumption of the MHE approach shown in Figure 8 is lower than the median fuel consumption of the PF approach, which means that the MHE is not only able to achieve low estimation errors like the PF but also enables the MPC path planner to rendezvous and dock with the target spacecraft more efficiently.

The Case 3 simulations do not have the challenges related to partial observability because the

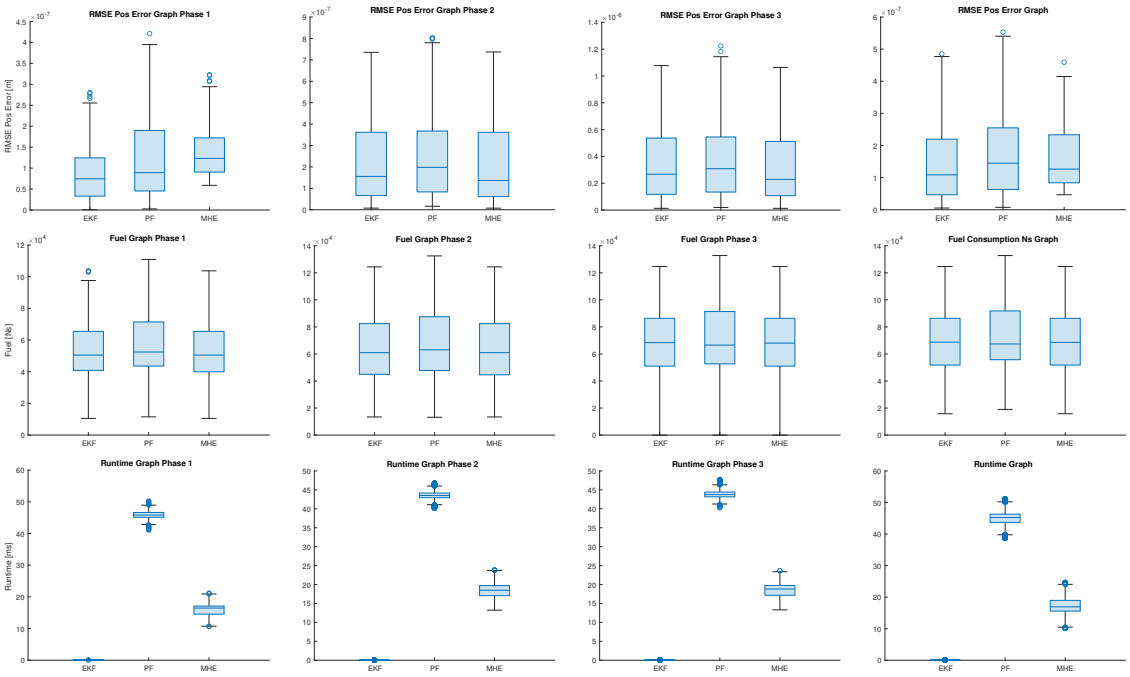

Figure 5 Statistics for the estimation accuracy, fuel consumption, and runtime metrics over all 100 simulations in Case 1. The columns show the statistics for the different mission phases, and the rows show the different metrics. The box shows values within the upper and lower quartiles, the horizontal line shows the median, and circles outside the whiskers represent outliers in the data.

chaser spacecraft starts in Phase 2. The MHE is the most successful in achieving docking. The differences between the PF and MHE are highlighted in Figures 9 and 11. Figure 9 shows high variance in the position, velocity, and thrust of the PF simulations, which shows that despite the ability of the PF to achieve small estimation errors in Phase 2, its estimation may vary too much to achieve efficient control with MPC. This is further shown in Figure 11 where the PF has mostly higher fuel consumption compared to MHE. While it is mostly able to produce accurate state estimates, the PF's procedure of taking the weighted average of many particles makes it prone to the noise of those particles. This is why the PF plots in Figure 9 do not converge in the same amount of time as MHE. In fact, most of the PF simulations take 1500 seconds, which is the maximum time the spacecraft is allocated to dock. Thus the spacecraft is often unable to dock with the PF. This is further shown in Figures 10 and 11 where the PF simulations seem to end near the origin, but the high fuel consumption shows that it is struggling to dock with the noisy estimates.

The tradeoff in runtime and performance is highlighted with the EKF being the fastest estimator with the worst performance while the MHE is the slowest estimator with the best performance. The EKF, while being fast, does not perform well when process noise/disturbances are introduced; often the chaser does not successfully dock. The PF can reduce the error caused by the introduced noise, but its implementation results in estimates that may be too noisy for the MPC to robustly rendezvous/dock. This leaves the MHE to be the best performing state estimator when combined with the MPC for rendezvous/docking.

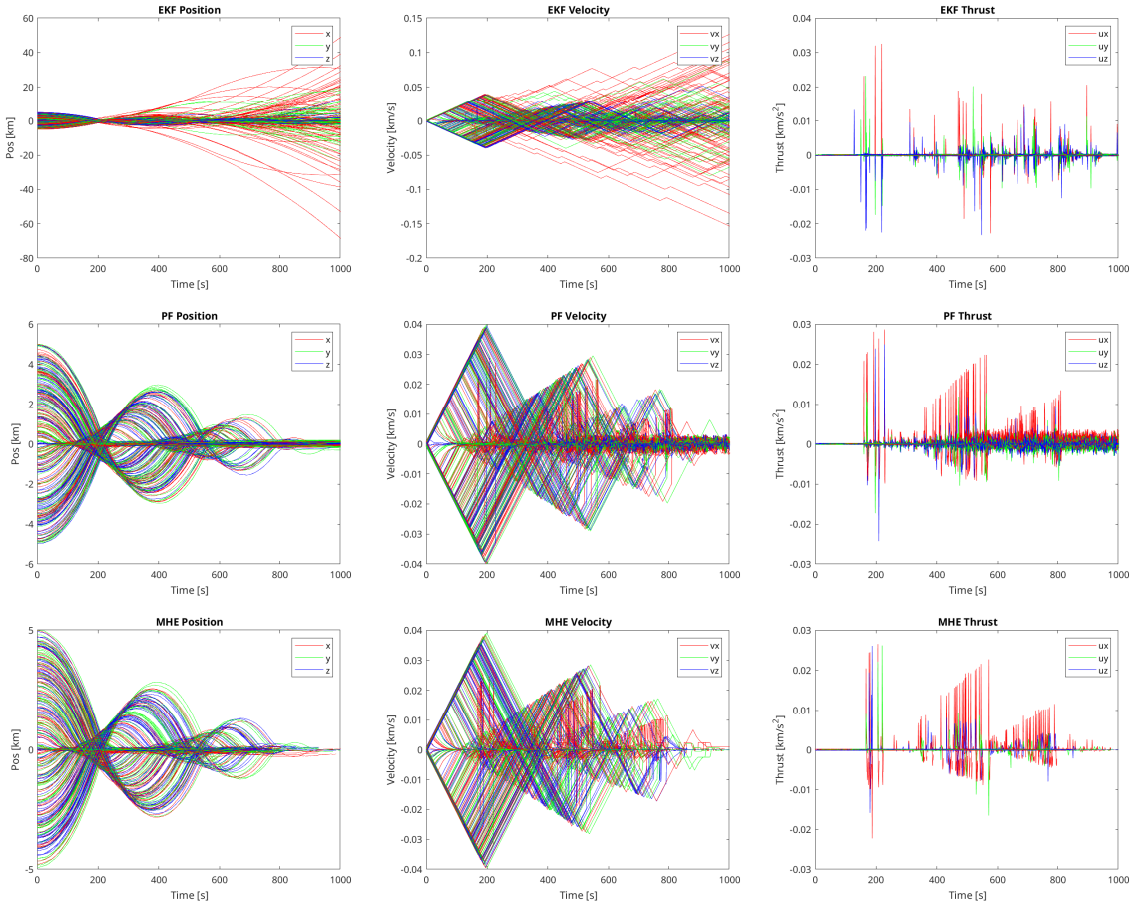

Figure 6 Simulation data for Case 2. The chaser position (left column), velocity (middle column), and thrust (right column) are shown for 100 simulations with the EKF (top row), PF (middle row), and MHE (bottom row).

Comments on implementation

Each algorithm required a tuning process to maximize the performance of the algorithm. The tunable EKF parameters are the process and sensor covariance matrices. These determine the state calculation throughout the process. The EKF process covariance should be tuned first as it is in the prediction step of the algorithm. Once the prediction is tuned, then the sensor covariance should be updated to match the ground truth as much as possible. The PF has more parameters than just the process and sensor covariances. The number of particles should be tuned according to the computational ability of the machine running it. Generally, more particles mean more accuracy in exchange for longer runtime. The ESS Threshold should also be tuned according to the computational ability of the machine since it performs resampling over all of the particles. The higher the ESS Threshold, the more resampling is performed which results in higher performance for the PF in exchange for longer runtime. The process covariance for the PF should be tuned much the same as the EKF, but the sensor covariance should not be. The sensor covariance determines the weights of the particles during estimation which if it is too inaccurate, would result in numerical errors and prevent the PF from performing effective state estimation. The tuning process should then be increasing/decreasing the sensor covariance such that it can appropriately weigh the particles according to the sensor mea-

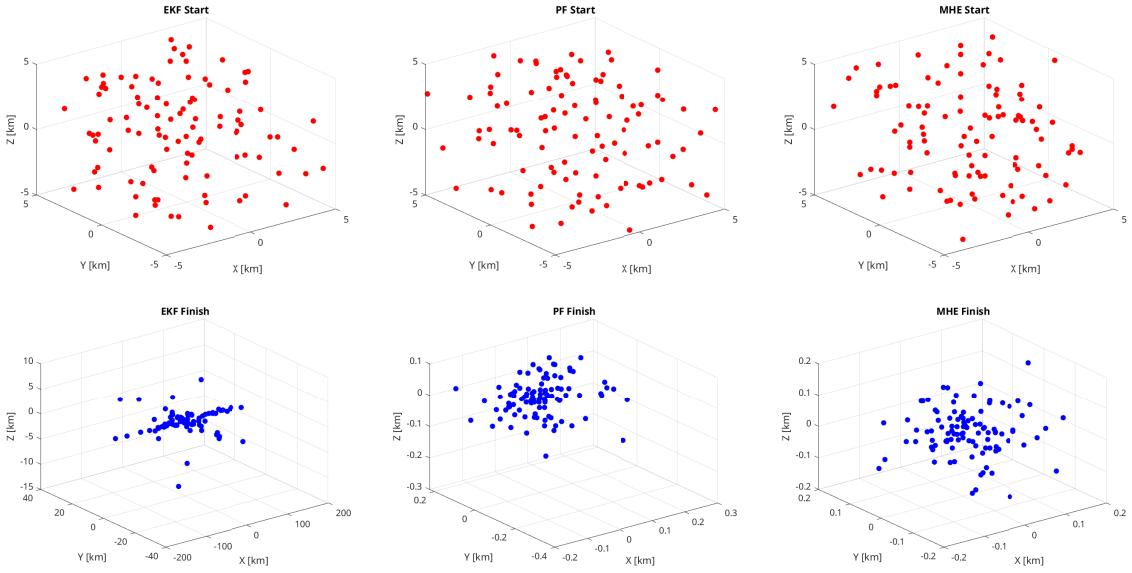

Figure 7 Mission start (top row) and finish (bottom row) locations for all 100 simulations in Case 2 with the EKF (left column), PF (middle column), and MHE (right column). Note the difference in axis scaling in the bottom plots.

surement without decreasing the weights so small that they result in numerical errors (such as a divide by zero error). The MHE should be tuned much the same as the EKF, however since the MHE uses weights in the objective function, the weighting matrices should be increased if there is higher weighted importance on that specific element.

There are several variations on these state estimation algorithm implementations that could improve their performance. One improvement for the PF could be to utilize other resampling methods other than ESS Threshold resampling, which could boost the PF performance. For MHE, a forgetting factor could be added to weight older measurements differently than more recent measurements, which could improve performance during particular spacecraft maneuvers.

When in Phase 1, it is better to keep the covariance/cost matrix for the algorithms to greatly outweigh the sensor covariance to improve performance when the state is only partially observable. For the EKF and PF, the process covariance matrices should have much smaller elements to provide a higher weight in the model. For the MHE, the cost matrix for the dynamics Q_{MHE} should be much higher than R in order to put more weight on optimizing for states that follow the dynamics than the sensor measurements.

CONCLUSION

We presented an Extended Kalman Filter (EKF), a Particle Filter (PF), and Moving Horizon Estimation (MHE) for state estimation in a spacecraft autonomous rendezvous and docking mission when only noisy nonlinear relative angle and range measurements are available and in the presence of process disturbances. We compared the performance of these three estimation algorithms when paired with a benchmark Model Predictive Controller in terms of estimation accuracy, computation time, overall mission time, and fuel consumption. We found a tradeoff between computational time and performance, and differences in performance between the estimation algorithms were

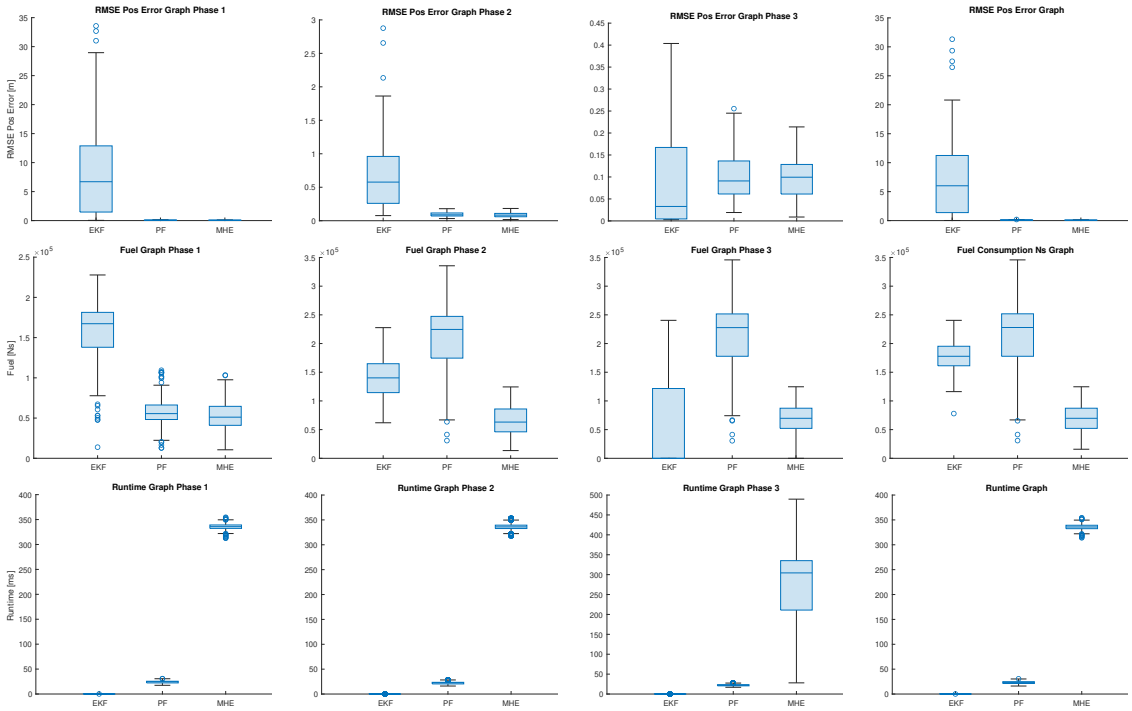

Figure 8 Statistics for the estimation accuracy, fuel consumption, and runtime metrics over all 100 simulations in Case 2. The columns show the statistics for the different mission phases, and the rows show the different metrics. The box shows values within the upper and lower quartiles, the horizontal line shows the median, and circles outside the whiskers represent outliers in the data.

more significant when there was process noise. MHE had the smallest estimation error and fuel consumption but the largest computation time. The EKF had the smallest computation time but the largest estimation error and greater fuel consumption than MHE. Moreover, using the EKF did not result in successful docking when there were process disturbances/noise. The PF performed almost as well as MHE in terms of estimation accuracy but required a little less computation time and had greater and much more varying fuel consumption between simulations.

ACKNOWLEDGEMENT

This work was supported by UCI's Undergraduate Research Opportunities Program (UROP) and the Air Force Office of Scientific Research (AFOSR) through the Air Force Research Laboratory (AFRL) Summer Faculty Fellowship Program.

REFERENCES

[1] K. L. Hobbs, J. B. Lyons, M. S. Feather, B. P. Bycroft, S. Phillips, M. Simon, M. Harter, K. Costello, Y. Gawdiak, and S. Paine, "Space Trusted Autonomy Readiness Levels," *IEEE Aerospace*, IEEE, 2023.

[2] C. Petersen, S. Phillips, K. L. Hobbs, and K. Lang, "Challenge Problem: Assured Satellite Proximity Operations," *31st AAS/AIAA Space Flight Mechanics Meeting*, 2021.

[3] K. Lang, C. Klett, K. Hawkins, E. Feron, P. Tsiotras, and S. Phillips, "Formal Verification Applied to Spacecraft Attitude Control," *AIAA Scitech 2021 Forum*, 2021.

[4] C. Jewison and R. S. Erwin, "A spacecraft benchmark problem for hybrid control and estimation," *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 2016, pp. 3300–3305.

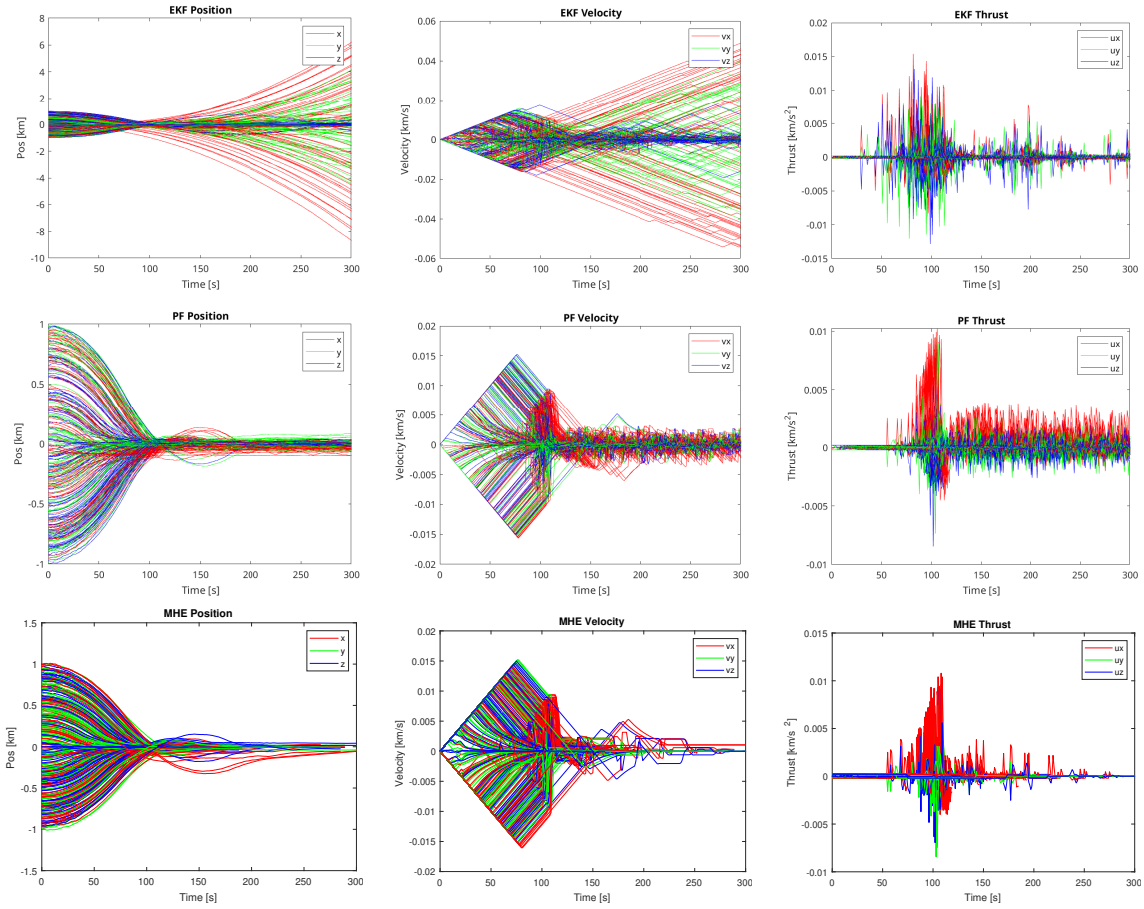

Figure 9 Simulation data for Case 3. The chaser position (left column), velocity (middle column), and thrust (right column) are shown for 100 simulations with the EKF (top row), PF (middle row), and MHE (bottom row).

[5] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*, Vol. 2. Nob Hill Publishing Madison, WI, 2017.

[6] D. Malyuta, Y. Yu, P. Elango, and B. Açıkmeşe, "Advances in trajectory optimization for space vehicle control," *Annual Reviews in Control*, Vol. 52, 2021, pp. 282–315.

[7] A. Weiss, I. Kolmanovsky, M. Baldwin, and R. S. Erwin, "Model predictive control of three dimensional spacecraft relative motion," *2012 American Control Conference (ACC)*, IEEE, 2012, pp. 173–178.

[8] C. Jewison, R. S. Erwin, and A. Saenz-Otero, "Model predictive control with ellipsoid obstacle constraints for spacecraft rendezvous," *IFAC-PapersOnLine*, Vol. 48, No. 9, 2015, pp. 257–262.

[9] A. Weiss, M. Baldwin, R. S. Erwin, and I. Kolmanovsky, "Model predictive control for spacecraft rendezvous and docking: Strategies for handling constraints and case studies," *IEEE Transactions on Control Systems Technology*, Vol. 23, No. 4, 2015, pp. 1638–1647.

[10] E. N. Hartley, "A tutorial on model predictive control for spacecraft rendezvous," *2015 European Control Conference (ECC)*, IEEE, 2015, pp. 1355–1361.

[11] E. N. Hartley, P. A. Trodden, A. G. Richards, and J. M. Maciejowski, "Model predictive control system design and implementation for spacecraft rendezvous," *Control Engineering Practice*, Vol. 20, No. 7, 2012, pp. 695–713.

[12] E. N. Hartley, M. Gallieri, and J. M. Maciejowski, "Terminal spacecraft rendezvous and capture with LASSO model predictive control," *International Journal of Control*, Vol. 86, No. 11, 2013, pp. 2104–2113.

[13] A. Zaman, A. A. Soderlund, C. Petersen, and S. Phillips, "Autonomous Satellite Rendezvous and Proximity Operations via Model Predictive Control Methods," *AAS Spaceflight Mechanics Meeting*, 2021.

[14] A. Gelb *et al.*, *Applied optimal estimation*. MIT press, 1974.

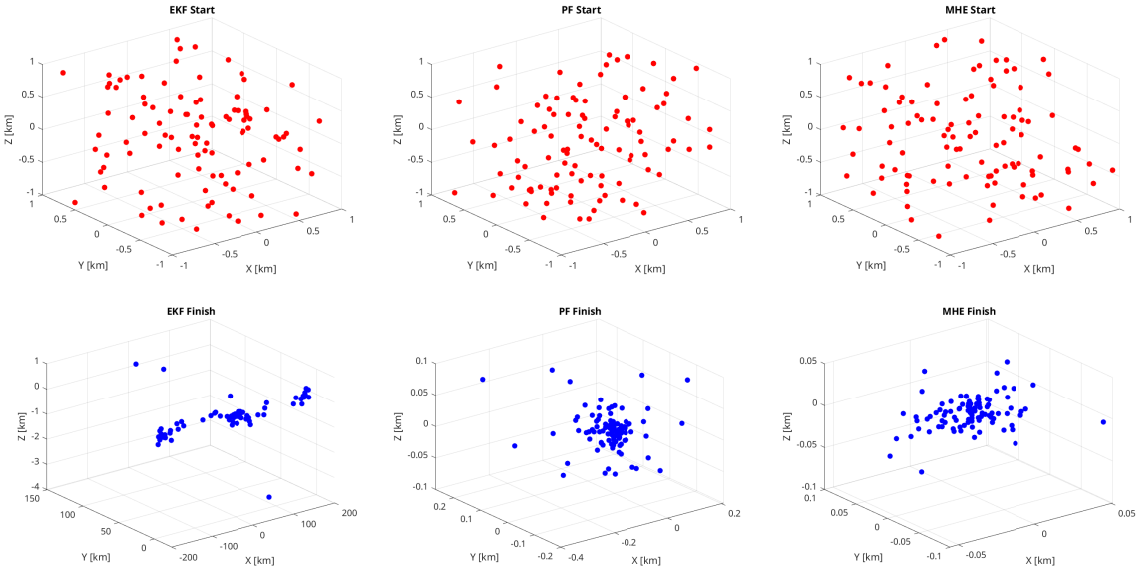

Figure 10 Mission start (top row) and finish (bottom row) locations for all 100 simulations in Case 3 with the EKF (left column), PF (middle column), and MHE (right column). Note the difference in axis scaling in the bottom plots.

[15] C. W. Hays, K. Miller, A. Soderlund, S. Phillips, and T. Henderson, "Autonomous Local Catalog Maintenance of Close Proximity Satellite Systems on Closed Natural Motion Trajectories," *AAS GNC Conference*, 2023.

[16] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on signal processing*, Vol. 50, No. 2, 2002, pp. 174–188.

[17] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations," *IEEE transactions on automatic control*, Vol. 48, No. 2, 2003, pp. 246–258.

[18] E. L. Haseltine and J. B. Rawlings, "Critical evaluation of extended Kalman filtering and moving-horizon estimation," *Industrial & engineering chemistry research*, Vol. 44, No. 8, 2005, pp. 2451–2460.

[19] B. Wie, *Space vehicle dynamics and control*. AIAA, 1998.

[20] G. W. Hill, "Researches in the lunar theory," *American journal of Mathematics*, Vol. 1, No. 1, 1878, pp. 5–26.

[21] W. Clohessy and R. Wiltshire, "Terminal guidance system for satellite rendezvous," *Journal of the Aerospace Sciences*, Vol. 27, No. 9, 1960, pp. 653–658.

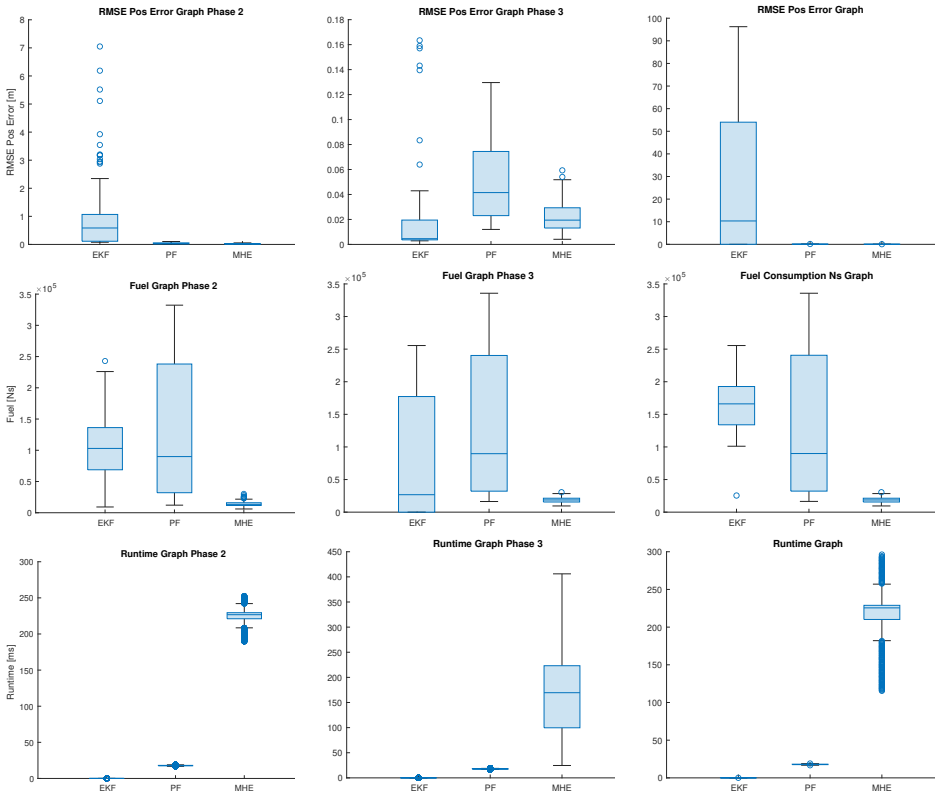

Figure 11 Statistics for the estimation accuracy, fuel consumption, and runtime metrics over all 100 simulations in Case 3. The columns show the statistics for the different mission phases, and the rows show the different metrics. The box shows values within the upper and lower quartiles, the horizontal line shows the median, and circles outside the whiskers represent outliers in the data.

Table 4 Estimation performance for Case 1 with measurement noise $\mathbf{v}_t$. $E(\mathbf{x})$ and $E(\dot{\mathbf{x}})$ denote the root mean squared error of the state's position and velocity, respectively. c.t. and m. t. denote "computation time" at each time step and "mission time," respectively.

		Phase 1			Phase 2			Phase 3			Entire Mission		
Metric	Units	EKF	PF	MHE	EKF	PF	MHE	EKF	PF	MHE	EKF	PF	MHE
mean{E(x)}	m	1e-7	1.3e-7	1.6e-7	2.3e-7	2.8e-7	2.2e-7	3.5e-7	4.1e-7	3.3e-7	1.5e-7	1.9e-7	1.8e-7
stdev{E(x)}	m	8.9e-8	1.1e-7	1.5e-7	2e-7	2.4e-7	2.1e-7	3e-7	3.4e-7	3e-7	1.3e-7	1.6e-7	1.6e-7
max{E(x)}	m	3.6e-7	4.6e-7	1.5e-6	9e-7	1e-6	9e-7	1.4e-6	1.5e-6	1.4e-6	6.3e-7	6.6e-7	1.2e-6
min{E(x)}	m	7e-10	2.5e-9	5.9e-8	7.5e-9	1.6e-8	7.4e-9	1.3e-8	1.9e-8	1.3e-8	5.3e-9	7.4e-9	4.7e-8
mean{E(x)}	m/s	4.7e-10	5.6e-10	4.5e-9	6.3e-10	7.2e-10	6e-10	7.3e-10	7.7e-10	6.4e-10	5.3e-10	6.2e-10	3e-9
stdev{E(x)}	m/s	3e-10	3.4e-10	9.7e-9	3.8e-10	4.3e-10	3.9e-10	4.6e-10	4.5e-10	4.1e-10	3.2e-10	3.7e-10	7.4e-9
max{E(x)}	m/s	1.3e-9	1.5e-9	9.5e-8	1.7e-9	1.7e-9	1.7e-9	2.5e-9	1.9e-9	1.8e-9	1.4e-9	1.5e-9	7.4e-8
min{E(x)}	m/s	2.7e-11	5.8e-11	1.2e-9	4.9e-11	9.7e-11	4.8e-11	5.5e-11	1e-10	5e-11	4.4e-11	8.6e-11	9.7e-10
mean{c.t.}	s	7.7e-5	4.6e-2	1.7e-2	6.2e-5	4.3e-2	1.9e-2	5.8e-5	4.5e-2	2e-2	7.2e-5	4.5e-2	1.8e-2
stdev{c.t.}	s	7e-4	9.5e-3	5.8e-3	1.2e-4	7.3e-3	5.8e-3	5e-5	1.2e-2	5.8e-3	5.6e-4	9.1e-3	5.9e-3
max{c.t.}	s	1e-1	2.3e-1	2.7e-1	10e-3	2.1e-1	1e-1	9.7e-4	2e-1	9.8e-2	1e-1	2.3e-1	2.7e-1
min{c.t.}	s	0	0	0	1.8e-5	2.7e-2	1e-2	1.8e-5	2.7e-2	1.2e-2	0	0	0
mean{m.t.}	s	-	-	-	-	-	-	-	-	-	5.8e2	6.1e2	5.7e2
stdev{m.t.}	s	-	-	-	-	-	-	-	-	-	2e2	2e2	2e2
max{m.t.}	s	-	-	-	-	-	-	-	-	-	1e3	1e3	1e3
min{m.t.}	s	-	-	-	-	-	-	-	-	-	1.7e2	1.8e2	1.7e2
mean{fuel}	Ns	-	-	-	-	-	-	-	-	-	1.4e2	1.5e2	1.4e2
stdev{fuel}	Ns	-	-	-	-	-	-	-	-	-	4.8e1	4.9e1	4.8e1
max{fuel}	Ns	-	-	-	-	-	-	-	-	-	2.5e2	2.7e2	2.5e2
min{fuel}	Ns	-	-	-	-	-	-	-	-	-	3.2e1	3.8e1	3.2e1

Table 5 Estimation performance for Case 2 with measurement noise $\mathbf{v}_t$ and process disturbances $\mathbf{d}_t$. $E(\mathbf{x})$ and $E(\dot{\mathbf{x}})$ denote the root mean squared error of the state's position and velocity, respectively. c.t. and m. t. denote "computation time" at each time step and "mission time," respectively.

		Phase 1			Phase 2			Phase 3			Entire Mission		
Metric	Units	EKF	PF	MHE	EKF	PF	MHE	EKF	PF	MHE	EKF	PF	MHE
mean{E(x)}	m	1.2e1	6.9e-2	6.3e-2	1.4e0	9.9e-2	8.9e-2	1.1e-1	1.1e-1	1e-1	1.1e1	9.4e-2	7.3e-2
stdev{E(x)}	m	1.4e1	2.6e-2	2.8e-2	2.1e0	4.5e-2	4e-2	1.3e-1	6.2e-2	5e-2	1.4e1	4e-2	3.1e-2
max{E(x)}	m	7.5e1	1.5e-1	1.5e-1	1.3e1	2.8e-1	2.1e-1	6.5e-1	3.4e-1	2.4e-1	7.2e1	2.6e-1	1.7e-1
min{E(x)}	m	4e-2	3.1e-2	2.1e-2	7.7e-2	3.2e-2	1.8e-2	2.9e-3	1.9e-2	8.9e-3	2e-2	3.8e-2	2.1e-2
mean{E(x)}	m/s	5.5e-5	1e-7	8.9e-8	6.1e-6	4.2e-7	2.2e-7	3.1e-6	6.8e-7	3.5e-7	5.1e-5	4.3e-7	1.4e-7
stdev{E(x)}	m/s	7.5e-5	8.1e-8	8.2e-8	8.6e-6	3.3e-7	2.1e-7	6.3e-6	6.1e-7	3.3e-7	7.1e-5	3.5e-7	1.3e-7
max{E(x)}	m/s	4.4e-4	4.2e-7	4.3e-7	4.3e-5	2e-6	1.2e-6	3.5e-5	3.8e-6	1.9e-6	4.2e-4	2.2e-6	7.4e-7
min{E(x)}	m/s	2e-8	6.6e-9	6.4e-9	5.1e-8	1.8e-8	4.1e-9	2e-8	2.3e-8	6.4e-9	5.1e-8	1.2e-8	5.8e-9
mean{c.t.}	s	7.5e-5	2.5e-2	3.1e-1	6.9e-5	2.3e-2	3.3e-1	5.9e-5	2.3e-2	2.7e-1	7.3e-5	2.4e-2	3.1e-1
stdev{c.t.}	s	4.5e-4	5.2e-3	7.8e-2	1.8e-4	4.1e-3	5e-2	2.6e-5	4.2e-3	9.9e-2	4.1e-4	4.6e-3	7.2e-2
max{c.t.}	s	7.7e-2	1.9e-1	1.6e0	1.2e-2	1.2e-1	1.6e0	1.1e-3	1.1e-1	1.6e0	7.7e-2	1.9e-1	1.6e0
min{c.t.}	s	0	0	0	2.5e-5	1.6e-2	1.1e-2	4.1e-5	1.7e-2	1.9e-2	0	0	0
mean{m.t.}	s	-	-	-	-	-	-	-	-	-	1.5e3	1.3e3	5.8e2
stdev{m.t.}	s	-	-	-	-	-	-	-	-	-	9.4e1	3.3e2	1.9e2
max{m.t.}	s	-	-	-	-	-	-	-	-	-	1.5e3	1.5e3	1e3
min{m.t.}	s	-	-	-	-	-	-	-	-	-	5.6e2	1.9e2	1.7e2
mean{fuel}	Ns	-	-	-	-	-	-	-	-	-	3.5e2	4.2e2	1.4e2
stdev{fuel}	Ns	-	-	-	-	-	-	-	-	-	5.8e1	1.2e2	4.8e1
max{fuel}	Ns	-	-	-	-	-	-	-	-	-	4.8e2	6.9e2	2.5e2
min{fuel}	Ns	-	-	-	-	-	-	-	-	-	1.6e2	6.2e1	3.2e1

Table 6 Estimation performance for Case 3 with measurement noise $\mathbf{v}_t$ and process disturbances $\mathbf{d}_t$. $E(\mathbf{x})$ and $E(\dot{\mathbf{x}})$ denote the root mean squared error of the state's position and velocity, respectively. c.t. and m. t. denote "computation time" at each time step and "mission time," respectively.

		Phase 2			Phase 3			Entire Mission		
Metric	Units	EKF	PF	MHE	EKF	PF	MHE	EKF	PF	MHE
mean{E(x)}	m	3.4e0	4.3e-2	2.3e-2	6.6e-2	5.4e-2	2.5e-2	2.7e1	4.9e-2	2.2e-2
stdev{E(x)}	m	7.8e0	3e-2	1.9e-2	1.1e-1	3.7e-2	1.8e-2	3.2e1	3.5e-2	1.5e-2
max{E(x)}	m	5.6e1	1.5e-1	1e-1	5.1e-1	1.8e-1	10e-2	9.6e1	1.6e-1	8e-2
min{E(x)}	m	7.8e-2	8.8e-3	5.2e-3	2.9e-3	1.2e-2	4.1e-3	6.6e-3	1e-2	5.2e-3
mean{E(x)}	m/s	1.1e-5	2.1e-7	2.7e-8	1.1e-7	3.1e-7	2.2e-8	1.6e-4	2.7e-7	2.7e-8
stdev{E(x)}	m/s	2.8e-5	2.6e-7	7.3e-8	1.8e-7	3.8e-7	3.1e-8	1.9e-4	3.4e-7	4.9e-8
max{E(x)}	m/s	2.1e-4	1.2e-6	7.2e-7	9.2e-7	1.7e-6	1.9e-7	5.8e-4	1.4e-6	4.7e-7
min{E(x)}	m/s	3.2e-9	1.2e-9	2.6e-9	3.2e-9	2e-9	1e-9	3.3e-9	1.3e-9	7e-9
mean{c.t.}	s	7.4e-5	2e-2	2.1e-1	4.9e-5	1.9e-2	1.6e-1	6e-5	1.9e-2	2e-1
stdev{c.t.}	s	6.2e-4	9.4e-3	5e-2	9.4e-5	8.6e-3	7.3e-2	4.7e-4	8.9e-3	6.4e-2
max{c.t.}	s	5.7e-2	1.5e-1	7.8e-1	1.3e-2	1.4e-1	5.3e-1	9.2e-2	1.5e-1	7.8e-1
min{c.t.}	s	0	0	0	2.7e-5	1.5e-2	2.5e-2	0	0	0
mean{m.t.}	s	-	-	-	-	-	-	1.5e3	8e2	1.4e2
stdev{m.t.}	s	-	-	-	-	-	-	1.4e2	6.1e2	4.7e1
max{m.t.}	s	-	-	-	-	-	-	1.5e3	1.5e3	3.2e2
min{m.t.}	s	-	-	-	-	-	-	1.1e2	1.1e2	7.3e1
mean{fuel}	Ns	-	-	-	-	-	-	3.3e2	2.7e2	3.7e1
stdev{fuel}	Ns	-	-	-	-	-	-	8.3e1	2.1e2	9e0
max{fuel}	Ns	-	-	-	-	-	-	5.1e2	6.7e2	6.1e1
min{fuel}	Ns	-	-	-	-	-	-	5.1e1	3.3e1	1.9e1