sensors

MDPI

Article

Proposals for Surmounting Sensor Noises

Andre Pittella [1] and Timothy Sands [2,*]

[1] Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14850, USA
[2] Department of Mechanical Engineering (SCPD), Stanford University, Stanford, CA 94305, USA
* Correspondence: dr.timsands@alumni.stanford.edu

Abstract: Classical and optimal control architectures for motion mechanics in the presence of noisy sensors use different algorithms and calculations to perform and control any number of physical demands, to varying degrees of accuracy and precision in regards to the system meeting the desired end state. To circumvent the deleterious effects of noisy sensors, a variety of control architectures are suggested, and their performances are tested for the purpose of comparison through the means of a Monte Carlo simulation that simulates how different parameters might vary under noise, representing real-world imperfect sensors. We find that improvements in one figure of merit often come at a cost in the performance in the others, especially depending on the presence of noise in the system sensors. If sensor noise is negligible, open-loop optimal control performs the best. However, in the overpowering presence of sensor noise, using a control law inversion patching filter performs as the best replacement, but has significant computational strain. The control law inversion filter produces state mean accuracy matching mathematically optimal results while reducing deviation by 36%. Meanwhile, rate sensor issues were more strongly ameliorated with 500% improved mean and 30% improved deviation. Inverting the patching filter is innovative but consequently understudied and lacks well-known equations to use for tuning gains. Therefore, such a patching filter has the additional drawback of having to be tuned through trial and error.

Keywords: sensor fusion; sensor noise; optimization; feedback; real-time optimization; velocity-based controller

check for updates

Citation: Pittella, A.; Sands, T. Proposals for Surmounting Sensor Noises. *Sensors* **2023**, *23*, 3169. https://doi.org/10.3390/s23063169

Academic Editor: Sylvain Girard

Received: 13 November 2022
Revised: 14 March 2023
Accepted: 14 March 2023
Published: 16 March 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Artemis I will be the first integrated flight test of NASA's deep space exploration system: the Orion spacecraft, Space Launch System (SLS) rocket, and the ground systems at Kennedy Space Center in Cape Canaveral, Florida. The first in a series of increasingly complex missions, Artemis I (Figure 1) will be an uncrewed flight that will provide a foundation for human deep space exploration and demonstrate our commitment and capability to extend human existence to the Moon and beyond. During this flight, the uncrewed Orion spacecraft will launch on the most powerful rocket in the world and travel thousands of miles beyond the Moon, farther than any spacecraft built for humans has ever flown, over the course of about a three-week mission [1].

1.1. Introduction to the Problem

Dealing with the fusion of potentially poor, noisy sensors is a ubiquitous challenge that has a long lineage leading to several disparate approaches. Figure 2 illustrates a use case, where noise from position and velocity sensors degrades the performance of state-of-the-art nonlinear adaptive spacecraft control methods. Similar to the problem defined in Section 2, the simulated machinery is performing an angular reorientation.



Figure 1. Lunar lander concept sought for the Artemis program, image courtesy NASA [2], image use in accordance with NASA image use policy. Reprinted/adapted with permission from Ref. [3]. NASA.

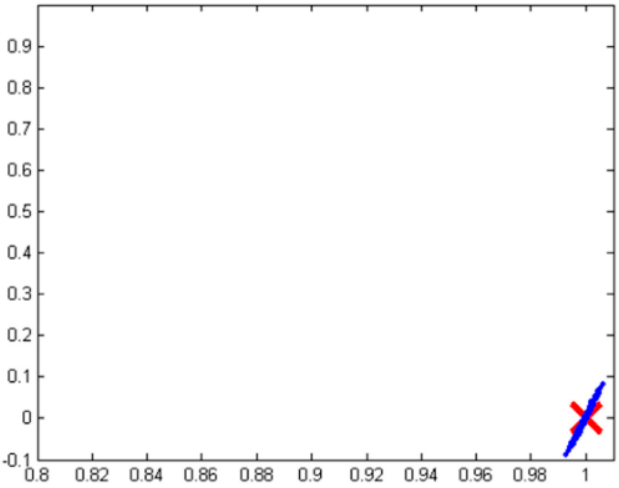


Figure 2. Illustration of deleterious effects of position and rate sensor noise on tracking performance of modern algorithms. Image taken from [4]. The desired state is indicated by a red 'x', and the result of simulations are the blue data points. Horizontal axis is angular position and vertical is angular velocity (rate), similar to the state being used in the problem defined in Section 2.

1.2. Review of the State-of-the-Art Alternatives to Address the Problem

Some researchers focus on improving the sensor itself or its signal using internal algorithms. Others seek to develop architectures that prove robust to fused noisy sensor data. Wang et al. proposed a novel method for accurate, autonomous and real-time orbit determinations for geo-referencing with a standalone global positioning system receiver [5]. Xiong et al. addressed systematic centroid errors and poor attitude accuracy by augmenting star trackers with an image intensifier [6]. Kim et al. proposed an algorithm for determining the orbit of a geostationary satellite using single-epoch measurements from a global positioning system receiver with sparse visibility of the global positioning system satellites [7]. Takayama illustrated weaknesses with global navigation system signals and proposed novel sensor noise models used to enhance sensor sensitivity [8].

Leake et al. proposed dealing with sensor issues with improved sensor algorithms, proposing a non-dimensional star identification algorithm compared in terms of accuracy, speed, and robustness to the so-called pyramid algorithm [9]. Marin et al. sought to enhance star trackers by sensor and sensor fusion algorithms to provide a smoother and faster output [10]. Perov et al. sought to utilize the principle of phase interferometer, using multiple receiving antennas [11]. Wang et al. sought to integrate communications navigation with global positioning system sensors [12]. Christian proposed autonomous augmentation using optical navigation by relativistic perturbation of starlight [13]. Fan et al.

investigated a plume noise suppression algorithm based on star point shape and the angular distance between stars [14]. Opromolla et al. proposed dealing with sensor issues algorithmically by using a model-based three-dimensional template matching technique for pose acquisition of an uncooperative space object [15].

Rather than dealing with sensor issue algorithmically as proposed by Opromolla, Chen et al. instead focused on the control algorithm using the sensor measurements and proposed a velocity-based impedance control scheme illustrating efficacy in both speed and robustness [16]. This manuscript parallels the insights of Chen et al. and enhances the velocity-based logic with optimization methods of Pontryagin akin Sandberg's recent recitation [17]. Sandberg's very recent improvements follow a lineage of small improvements from its provenance in the nonlinear adaptive control methods of the 1990s, as proposed by Slotine [18], and afterward improved by Fossen [19], Sands [20,21], who offered experimental validation in [22]. Most recently, Raigoza [23] augmented the method with autonomous collision avoidance, and Wilt [24] evaluated efficacy in the face of uniform variations in mass moment of inertia (e.g., from fuel slosh). This manuscript proposes and illustrates methods that assume velocity-based control logic like Chen but seek to induce open-loop optimal results like Sandberg, Raigoza, and Wilt, and thus mitigate deleterious effects of noisy sensors.

Controllers are differential equations that are designed to be able to manipulate some state variable by arbitrarily changing a variable connected to it through a physical law or equation. They often feature several points of adjustment and tunability to fit figures of merit desired by the designer. The figures of merit utilized here include spacecraft tracking accuracy (how well the controller reaches the desired target), precision (as measured by deviation), and cost (numerical evaluation of the resources used). Computational burden is also monitored since difficulties such as singular matrix inversions are involved. A variety of control system architectures were evaluated, and their ability is assessed to perform the task of a rest-to-rest reorientation normalized to unity. The different architecture's performance under the previously mentioned figures of merit will be compared to show how they respond to a same noise environment, and which one best mitigates the effects of noisy sensors.

1.3. Novelties Presented

1. Open-loop optimal results are analytically calculated providing a performance benchmark for comparing other methods.
2. Inspired by Opromolla, 2015 [11], velocity-based classical control is investigated, especially since it was utilized as the comparative benchmark by Sandberg (2022) [17], Raigoza (2022) [23], and Wilt (2022) [24], while the open-loop optimal results in item #1 are used as comparative benchmark here.
3. Real-time optimal control is compared since it was the latest cited proposals of 2022, and can be considered state of the art for this application, as an analytic optimal solution is found, which is not always possible for a generic system.
4. Double-integrator patching filters are implemented seeking to match open-loop optimal results amidst the fusion of noisy sensors.
5. System-inverting patching filters are implemented, also seeking to match open-loop optimal results amidst the fusion of noisy sensors.

1.4. Feedback Control System Topology

This paper will focus on changes to the control architecture (depicted in Figure 3) rather than making direct improvements or modifications to the sensors. Noise is accepted as a fact of the environment, and the main effort is to test control architectures on how they are able to perform in the presence of said noise. The desired state and system dynamics are defined in the following section.

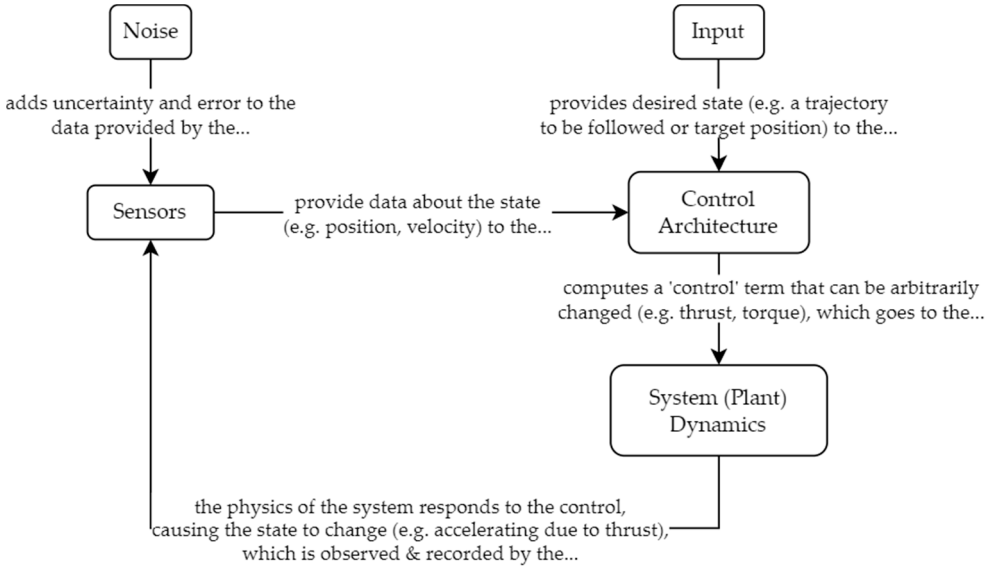


Figure 3. Flowchart showing the topology of a generic system, explaining the connection between the sensors, the control architecture, and the system dynamics.

2. Control System Architectures

This section offers formulas used in the study, and the implementation of the formulas in simulations is provided in Appendices A and B. The aim of each architecture, and the selection of equations thereof, is to reach the desired state outlined in Section 2.1, even in the presence of noise from the sensor. The formulas provided show how the control of different architectures is computed, but not explicitly how they work around noise.

2.1. The Task at Hand

The controllers will perform a rest-to-rest reorientation of one unit of rotation (indicated by variable θ whose rate is indicated by variable ω) scaled to unity over unit of time scaled to unity. The equations for such a maneuver are listed in Equations (1)–(3). Controller cost will be used as a key figure of merit, where a quadratic cost computation will be used indicated by variable J in Equation (4).

$$\theta(0) = \omega(0) = 0 \tag{1}$$

$$\theta(1) = \theta_d = 1 \tag{2}$$

$$\omega(1) = 0 \tag{3}$$

$$J = \frac{1}{2}\int_0^1 u(t)^2 dt \tag{4}$$

with state $\theta(t)$, rate $\omega(t)$, desired state θ_d, quadratic cost functional $J(t)$, and control variable $u(t)$. The state, rate, and control are connected in accordance with Equations (5) and (6).

$$\omega = \dot{\theta} \tag{5}$$

$$u = \frac{1}{I}\dot{\omega} \tag{6}$$

for moment of inertia I, where dotted variables indicate derivatives in time.

2.2. Proportional Plus Velocity (P+V) Control

The P+V controller is a form of classical control, tuned using two different gains: a proportional gain K_P applied to the state error, added to a velocity gain K_V applied to the rate, *not the rate error* (see (14)). P+V controllers can be easily tuned to fit desired settling

times and damping ratios. For a maximum tolerance of 2% error at the desired settling time, Equation (7):

$$t_s = -\frac{\ln\left(0.02 \times \sqrt{1-\zeta^2}\right)}{\zeta\omega_n} \tag{7}$$

can be used, where ζ is the damping ratio of the controller and ω_n is its natural frequency response. For a P+V controller, these are known to be Equations (8) and (9).

$$\omega_n = \sqrt{K_P} \tag{8}$$

and

$$\zeta = \frac{K_V}{2\sqrt{K_P}} \tag{9}$$

For a settling time of ≤ 1 s and a damping ratio of 0.7, the gains are calculated to be $K_P = 37$ and $K_V = 8.5$.

P+V controllers are always able to reach the desired value, but it is the selection of gains that affects how quickly it is reached. This ability is hampered by noise, especially on the rate sensor. Because the end rate is 0, any error in rate detection will cause a control to move the system off of the desired value. Additionally, the settling time is only for the state target, so no estimation of the rate target settling time is used to set the gains.

2.3. Open-Loop Optimal Control

For a given problem with dynamics, boundary conditions, and a cost functional, an optimal solution exists that minimizes the output of the cost functional, which is found using Pontryagin's method [25]. Using Equations (1)–(6), the task described in *the task at hand* (from Section 2.1) has an optimal control u^* of the form in Equation (10).

$$u^*(t) = \theta_d(at + b) \tag{10}$$

By integrating twice and applying the initial and final boundary conditions, the linear system in Equation (11) is found:

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1/6 & 1/2 & 1 & 1 \\ 1/2 & 1 & 1 & 0 \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ \theta_d \\ 0 \end{Bmatrix} \tag{11}$$

where the solution to Equation (11) can be applied to (10) to yield Equation (12):

$$u^*(t) = \theta_d(-12t + 6) \tag{12}$$

which has a minimal quadratic cost of $J^* = 6$ and exact achievement of the target end conditions. While open-loop optimal control yields the perfect results on paper, it is completely blind to noise and perturbation.

2.4. Real-Time Optimal Control

Real-time optimal control, or RTOC, is a modification of the previous iteration of optimal control that considers the current state of the system, allowing the controller to adjust for noise and perturbations. The top two rows of the matrix in (11) are the equations for $\theta^*(t)$ and $\omega^*(t)$, the optimal forms of the state and rate variables, evaluated at the initial

conditions. By modifying these rows such that the forms are evaluated at the current time t_0, the linear system changes to:

$$\begin{bmatrix} t_0^3/6 & t_0^2/2 & t_0 & 1 \\ t_0^2/2 & t_0 & 1 & 0 \\ 1/6 & 1/2 & 1 & 1 \\ 1/2 & 1 & 1 & 0 \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \end{Bmatrix} = \begin{Bmatrix} \theta(t_0) \\ \omega(t_0) \\ \theta_d \\ 0 \end{Bmatrix} \tag{13}$$

The vector of unknowns on the left-hand side is calculated by inverting the matrix and multiplying it by the vector of knowns on the right-hand side. The unknowns a and b are then used in (10) to make an instantaneous optimal control, calculated for the current state of the system. Therefore, the controller can adjust its control output in the presence of noise and perturbations.

While being able to change course based on the current state of the system is an advantage over open-loop optimal control, RTOC has a large computational demand in its inversion of a 4×4 matrix. Additionally, the matrix becomes singular as $t_0 \to 1$, which causes problems as singular matrices are uninvertible. In simulation, the effect of this is the control shooting off to infinity towards the end of the runtime, leading to huge errors in the state and/or rate. To avoid this behavior, a switch is worked in so that, when the matrix determinant approaches 0, the controller switches to open-loop optimal control for the remainder of the runtime.

2.5. Patching Filter: Double Integrator

One way to combine the analytical accuracy of optimal control and the computational simplicity and tunability of classical P+V control is to impose a 'patching' filter that integrates the optimal control $u^*(t)$ twice to find the optimal state function $\theta^*(t)$, which is given to a P+V controller as the desired value instead of a constant command θ_d.

2.6. Patching Filter: Double Integrator, Tuned

However, the result of the previous architecture may not be desirable. Should the gains of the original P+V controller be modifiable, tuning them around the patching filter is a viable strategy. This architecture is not as well studied as the original P+V control, so handy equations based around desired settling time and damping ratio are not available. Therefore, the gains will be tuned by hand, i.e., trial and error, until the result meets acceptable figures of merit.

2.7. Patching Filter: Control Law Inversion

Should the gains not be modifiable, there is another way to adjust the architecture. Instead of directly integrating the optimal control, the control law for the P+V controller can be solved for an optimal input such that, after all the gains are applied and the control calculated, the P+V controller output matches that of the optimal control $u^*(t)$, where the asterisk '*' indicates optimality. The P+V control law is listed in Equation (14). Solving for the desired state results in Equation (15).

$$u = K_P(\theta_d - \theta) - K_V\omega \tag{14}$$

$$\theta_d = \frac{1}{K_P}u + \frac{K_V}{K_P}\omega + \theta \tag{15}$$

Entering the optimal values u^*, ω^*, and θ^* gives the optimal input. For a patching filter that works in the Laplace domain as a transfer function, this can be rewritten as Equation (16), which can be simplified to become Equation (17).

$$\Theta_d(s) = U^*(s)\left(\frac{1}{K_P} + \frac{K_V}{K_P s} + \frac{1}{s^2}\right) \tag{16}$$

$$\frac{\Theta_d(s)}{U^*(s)} = \left(\frac{s^2 + K_V s + K_P}{K_P s^2}\right) \tag{17}$$

Equation (17) is the patching filter to be applied to the optimal control yielding the optimal state trajectory input for the P+V controller.

3. Results and Analysis

The following simulations were performed in MATLAB Simulink, using the ode4 (Runge—Kutta) integrator and a 0.01s step size. Using a random number generator with a Gaussian distribution, a noise signal was added to the state and rate with variances of 0.0001 with zero mean. The moment of inertia was also varied randomly but uniformly to ±10% of its true value.

Monte Carlo Simulation

Displayed in Figure 4 are the results of 1000 simulations of each of the control architectures outlined in the previous section. Table 1 contains the numerical performances of each architecture for direct comparison.

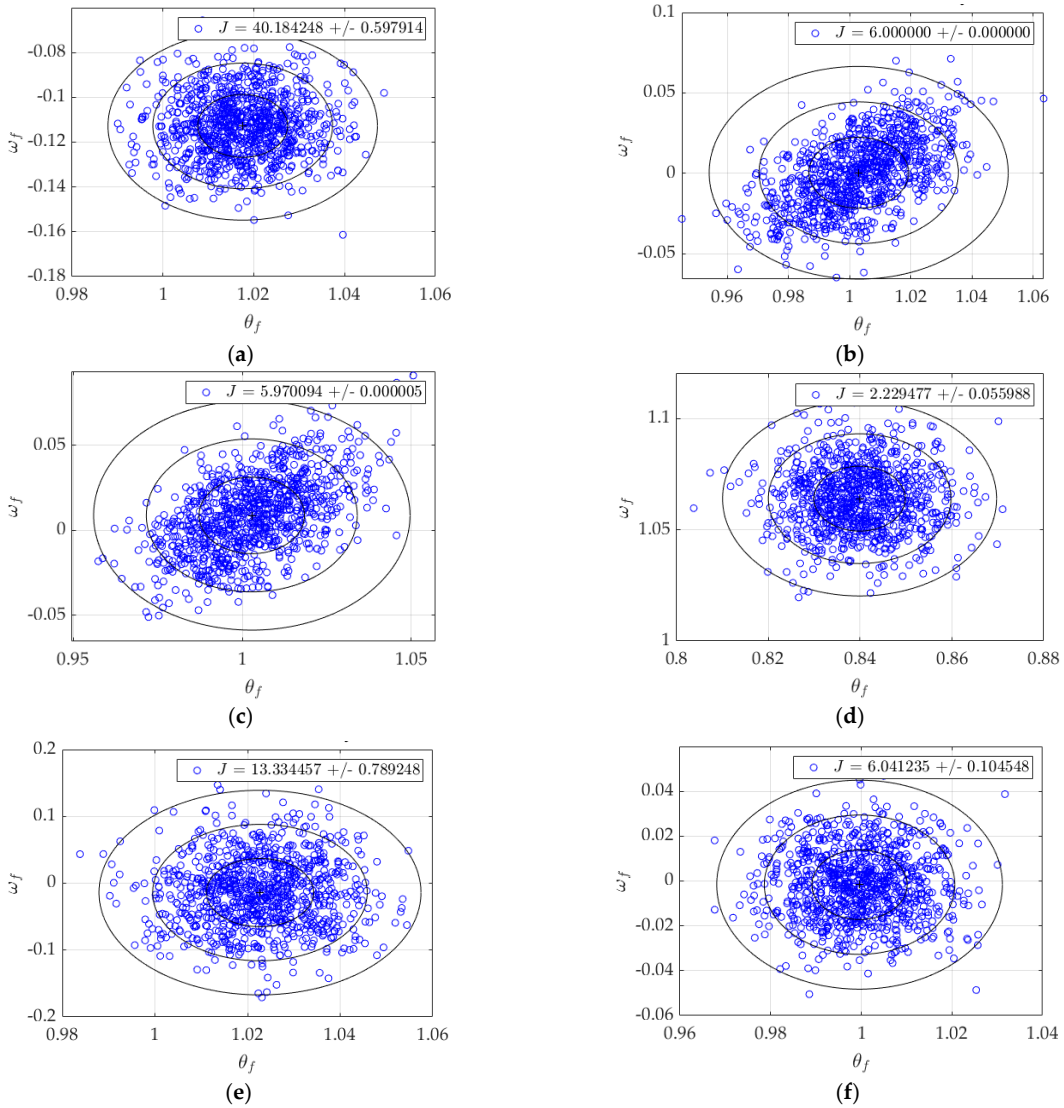


Figure 4. Monte Carlo plots, N = 1000. Rings show 1-sigma, 2-sigma, and 3-sigma deviations from the mean, denoted by a +. (**a**) P+V control; (**b**) open-loop optimal control; (**c**) RTOC; (**d**) double-integrator patching filter; (**e**) double integrator, tuned; (**f**) control law inversion patching filter.

Table 1. Numerical results of Monte Carlo simulations, $N = 1000$. σ refers to the standard deviation of a quantity. All other numerical results represent the mean over all simulations.

Architecture	θ_f	σ_θ	ω_f	σ_ω	J	σ_J
P+V [1]	1.0176	0.0099	−0.1128	0.0140	40.1842	0.5979
Open-loop optimal	1.0029	0.0163	0.0004	0.0220	6.0000	0.0000
RTOC	1.0028	0.0225	0.0087	0.0156	5.9701	0.00001
Double Integrator [1]	0.8399	0.0100	1.0640	0.0146	2.2295	0.0560
Double Int., Tuned [2]	1.0227	0.0116	−0.0140	0.0510	13.3345	0.7892
Control Inversion [1]	0.9996	0.0105	−0.0016	0.0155	6.0412	0.1045
P+V [1]	1.0176	0.0099	−0.1128	0.0140	40.1842	0.5979

[1] $K_P = 37, K_V = 8.5$. [2] $K_P = 78, K_V = 0.65$.

From Table 1, the relationships between accuracy (the final values of θ & ω), precision (the magnitude of their spread, seen in σ), and cost (J) can be evaluated. The P+V controller, while having the smallest standard deviations for both values, has notable error in ω_f and the highest quadratic cost, 301% the cost of the next-highest cost. Open-loop optimal has some of the most accurate mean values, but the second-highest spread in both state and rate. Depicted in Figures 5 and 6, RTOC is similar, with lower cost (−0.5%) and rate spread (−29%) but higher state spread (38%) and final rate error (218%).

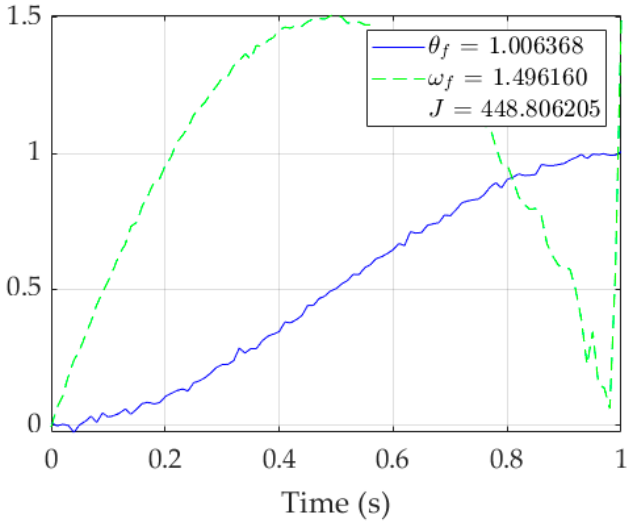


Figure 5. Singular run of RTOC using *pinv*(A) without open-loop switching. Note the deterioration in $\omega(t)$ starting after time = 0.8 s, indicating the function's inability to reliably solve the system as it approaches singularity.

The double-integrator patching filter has the second-lowest spreads in both state and rate and the lowest cost, but the final values are completely off the mark; 16% error in state and 943% more rate error than the next highest architecture. After some tuning, it performs similarly to the P+V controller but with higher spreads (17.2% and 264%, respectively), lower cost (−66.8%), and a more accurate final rate (12% the error of the P+V). Finally, the control law inversion patching filter strikes a balance between all figures of merit: lowest state error, third lowest state spread, second lowest rate error, third lowest state spread, and a cost only marginally higher (0.7%) than the optimal $J^* = 6$, where the asterisk '*' indicates optimality. However, it took the longest to run after RTOC.

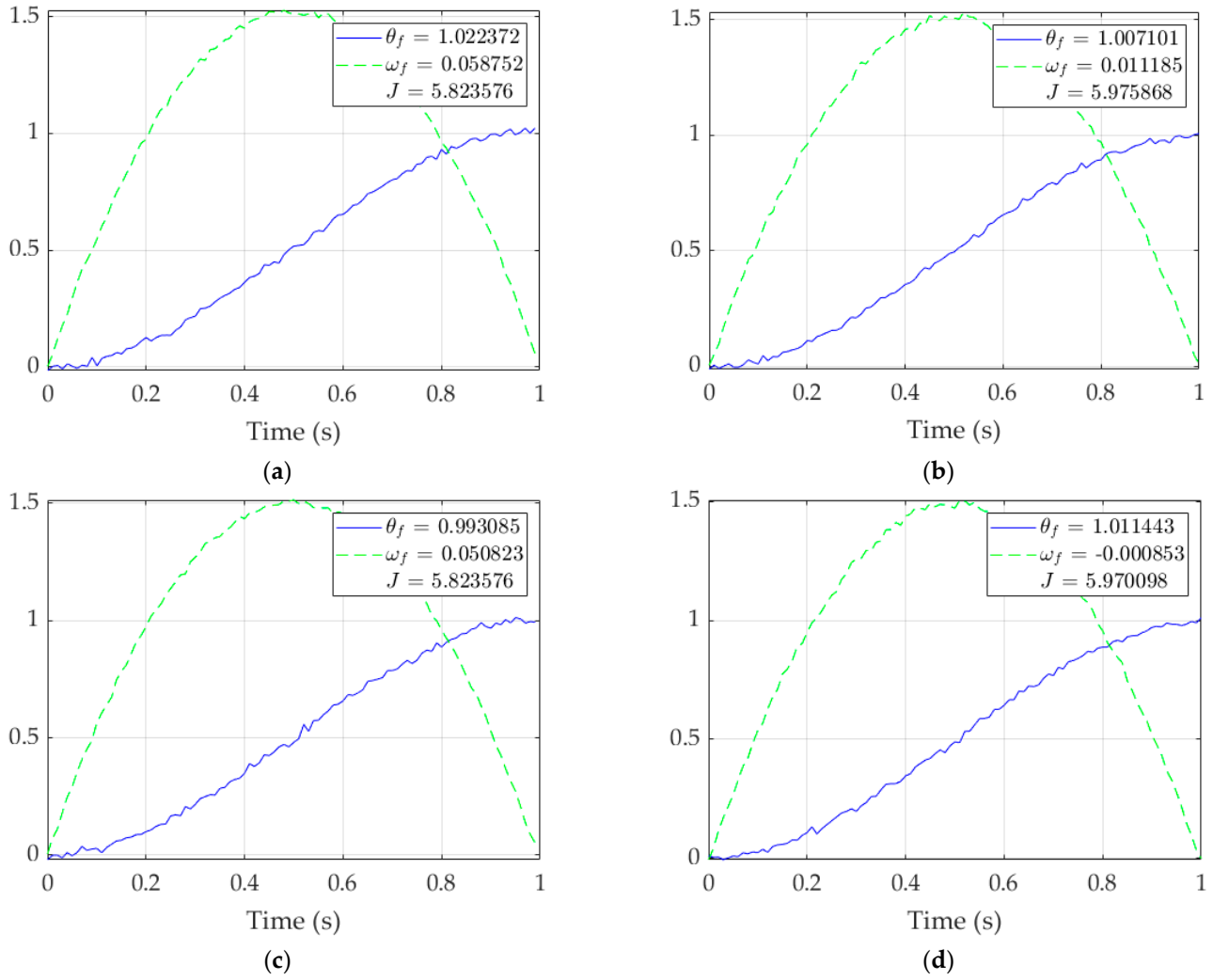


Figure 6. Singular runs of RTOC for each matrix inverse procedure. **Solid blue line**: $\theta(t)$; **dashed green line**: $\omega(t)$. (**a**) $(A)^{-1}$; (**b**) $1\backslash A$; (**c**) inv(A); (**d**) pinv(A). Note the failure of plots (**a**,**c**) to reach the right side, where time = 1 s.

4. Conclusions

From the results shown in Section 3 summarized in Table 2, there is no one control that performs the best in every single regard.

Table 2. Numerical results of Monte Carlo simulations, N = 1000. σ refers to the standard deviation of a quantity. All other numerical results represent the mean over all simulations. Percent differences are compared to the benchmark open-loop optimal results.

Architecture	θ_f	σ_θ	ω_f	σ_ω	*Cost*	σ_J
P+V [1]	1%	−39%	−28,300%	−36%	570%	60%
Open-loop optimal	–	–	–	–	–	–
RTOC	0%	38%	2075%	−29%	0%	0%
Double Integrator [1]	−16%	−39%	265,900%	−34%	−63%	6%
Double Int., Tuned [2]	2%	−29%	−3600%	132%	122%	79%
Control Inversion [1]	**0%**	**−36%**	–	**−30%**	**1%**	**10%**

[1] $K_P = 37, K_V = 8.5$. [2] $K_P = 78, K_V = 0.65$.

4.1. Recapped Research

1. Velocity-based classical control was investigated in the prequels and established as the comparative benchmark here where open-loop optimal results in item #2 were used for initial comparison. *The controller achieved the best angle tracking deviation, but the worst cost performance.*
2. Open-loop optimal results were presented analytically providing a performance benchmark for comparing other proposed methods.
3. Real-time optimal control was presented and compared to the benchmarks.
4. Double-integrator patching filters were introduced seeking to match open-loop optimal results amidst fusion of noisy sensors. *These filters achieved the best velocity tracking performance.*
5. System-inverting patching filters were implemented also seeking to match open-loop optimal results amidst fusion of noisy sensors. These patching filters achieved the best overall performance with notable improvements amidst no dramatic performance degradation in any categories of performance.

4.2. Concluding Results

The P+V controller has high cost and rate error, the optimal controls have low cost and high accuracy but higher susceptibility to sensor noise, while the double-integrator patching filter is difficult to tune and costly. The control law inversion patching filter, however, can be seen as the best alternative to open-loop optimal control, especially in the presence of sensor noise and inertia uncertainties, as it had a cost only slightly higher (0.7%) with smaller standard deviations (35.6% less spread in state, 29.5% less spread in rate) and final results more accurate than any other architecture (14.3% being the state error of the runner-up, 18.4% the rate error of the runner-up). The key tradeoff was computational burden since control inversion optimization took 25–40% longer to run than some of the other architectures.

Recommendation: *In instances where performance must be induced on pre-existing systems, control inversion patching filters are advised.* □

Author Contributions: Methodology, A.P.; Software, A.P.; Validation, T.S.; Formal analysis, A.P.; Investigation, A.P.; Resources, T.S.; Data curation, A.P.; Writing—original draft, A.P.; Writing—review & editing, T.S.; Visualization, A.P.; Supervision, T.S.; Project administration, T.S.; Funding acquisition, T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. SIMULINK Simulation Code Used to Replicate the Presented Results

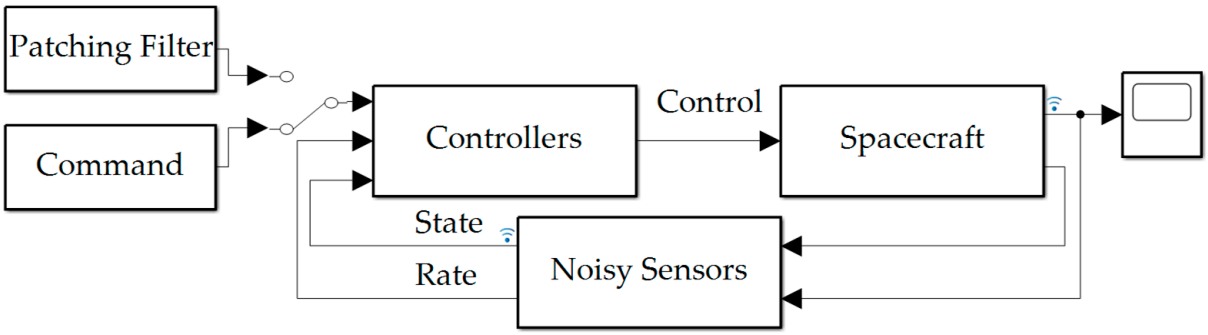


Figure A1. Patching filter simulation topology.

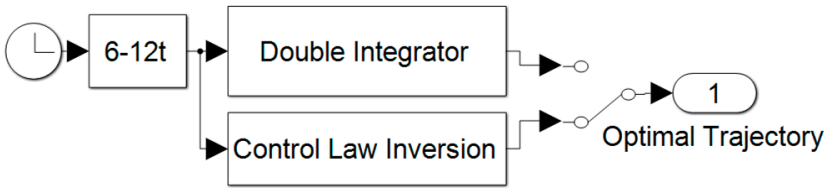


Figure A2. Pre-filtering simulation topology.

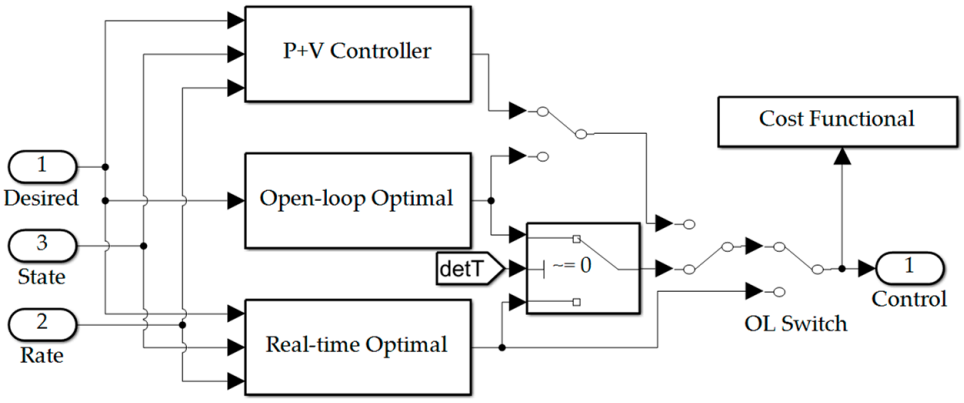


Figure A3. Controllers' simulation topology.

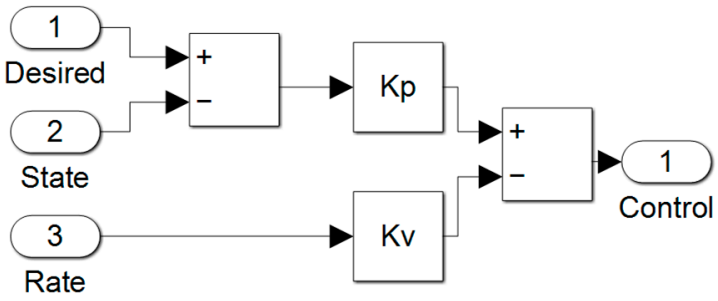


Figure A4. P+V controller topology.

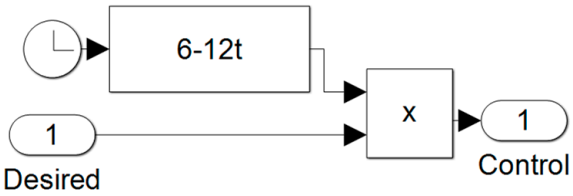


Figure A5. Open-loop optimal controller topology.

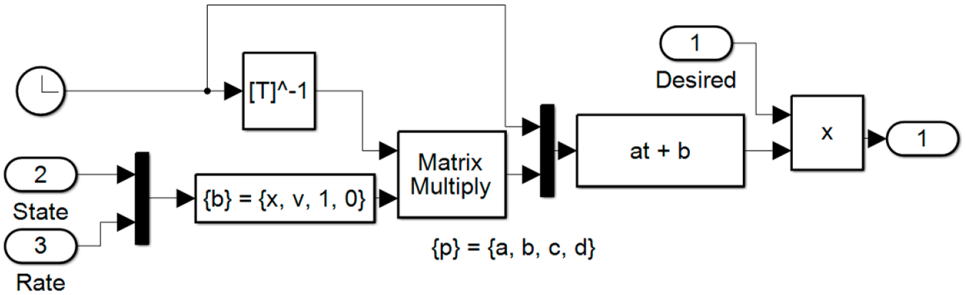


Figure A6. Closed-loop real-time optimal controller topology.

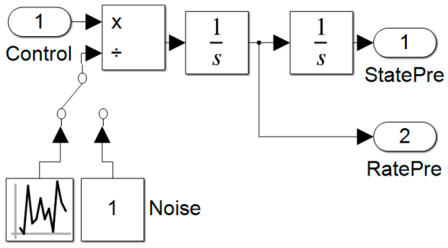


Figure A7. Spacecraft simulation topology.

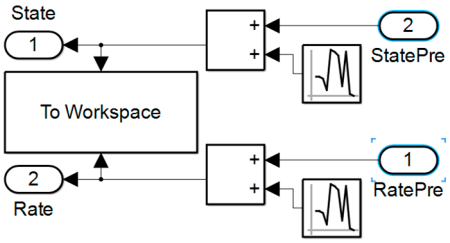


Figure A8. Noisy sensor-fusion simulation topology.

Appendix B. MATLAB Wrapper Code

```
% If you are running this script, go into the Simulink model callbacks.
% Under StopFcn, set 'graphit' to 0, or MATLAB will plot the individual
% simulation results . . . n times. This cuts down total run time by about
% half.
close all
n = 1000;
state_f = zeros(1,n);
rate_f = zeros(1,n);
J_stor = zeros(1,n);
for i = 1:n
    rng('shuffle');
    simout = sim('FinalProject');
    state_f(i) = simout.state(end);
    rate_f(i) = simout.rate(end);
    J_stor(i) = simout.J(end);
    i
end
scatter(state_f, rate_f, 'MarkerEdgeColor', 'b')
box on
grid on
hold on
title('Inertia, Position, and Velocity Noise')
xlabel('$$\theta_{f}$$','interpreter','latex')
ylabel('$$\omega_{f}$$','interpreter','latex')
sig_theta = std(state_f);
sig_omega = std(rate_f);
sig_J = std(J_stor);
t = 0:0.01:2*pi;
x0 = mean(state_f);
y0 = mean(rate_f);
Jm = mean(J_stor);
x1 = x0 + sig_theta*cos(t);
x2 = x0 + 2*sig_theta*cos(t);
x3 = x0 + 3*sig_theta*cos(t);
```

```
y1 = y0 + sig_omega*sin(t);
y2 = y0 + 2*sig_omega*sin(t);
y3 = y0 + 3*sig_omega*sin(t);
plot(x1,y1,'k-')
plot(x2,y2,'k-')
plot(x3,y3,'k-')
plot(x0,y0,'k+')
hold off
legend(sprintf('$$J$$ = %f +/− %f', Jm, sig_J), 'interpreter', 'latex')
ax = gca;
ax.FontSize = 16;
ax.FontName = 'Palatino Linotype';
```

References

1. Hambleton, K. Artemis I Map, 9 February 2018. Available online: https://www.nasa.gov/image-feature/artemis-i-map (accessed on 12 November 2022).
2. Mahoney, E. Fast-Track to the Moon: NASA Opens Call for Artemis Lunar Landers, 30 September 2019. Available online: https://www.nasa.gov/feature/fast-track-to-the-moon-nasa-opens-call-for-artemis-lunar-landers/ (accessed on 12 November 2022).
3. Media Usage Guidelines. Available online: https://www.nasa.gov/multimedia/guidelines/index.html (accessed on 12 November 2022).
4. Sands, T. Comparison and Interpretation Methods for Predictive Control of Mechanics. *Algorithms* **2019**, *12*, 232. [CrossRef]
5. Wang, F.; Gong, X.; Sang, J.; Zhang, X. A Novel Method for Precise Onboard Real-Time Orbit Determination with a Standalone GPS Receiver. *Sensors* **2015**, *15*, 30403–30418. [CrossRef] [PubMed]
6. Xiong, K.; Jiang, J. Reducing Systematic Centroid Errors Induced by Fiber Optic Faceplates in Intensified High-Accuracy Star Trackers. *Sensors* **2015**, *15*, 12389–12409. [CrossRef] [PubMed]
7. Kim, G.; Kim, C.; Kee, C. Coarse Initial Orbit Determination for a Geostationary Satellite Using Single-Epoch GPS Measurements. *Sensors* **2015**, *15*, 7878–7897. [CrossRef] [PubMed]
8. Takayama, Y.; Urakubo, T.; Tamaki, H. Novel Process Noise Model for GNSS Kalman Filter Based on Sensitivity Analysis of Covariance with Poor Satellite Geometry. *Sensors* **2021**, *21*, 6056. [CrossRef] [PubMed]
9. Leake, C.; Arnas, D.; Mortari, D. Non-Dimensional Star-Identification. *Sensors* **2020**, *20*, 2697. [CrossRef] [PubMed]
10. Marin, M.; Bang, H. Design and Simulation of a High-Speed Star Tracker for Direct Optical Feedback Control in ADCS. *Sensors* **2020**, *20*, 2388. [CrossRef] [PubMed]
11. Perov, A.; Shatilov, A. Deeply Integrated GNSS/Gyro Attitude Determination System. *Sensors* **2020**, *20*, 2203. [CrossRef] [PubMed]
12. Wang, L.; Lü, Z.; Tang, X.; Zhang, K.; Wang, F. LEO-Augmented GNSS Based on Communication Navigation Integrated Signal. *Sensors* **2019**, *19*, 4700. [CrossRef] [PubMed]
13. Christian, J.A. StarNAV: Autonomous Optical Navigation of a Spacecraft by the Relativistic Perturbation of Starlight. *Sensors* **2019**, *19*, 4064. [CrossRef] [PubMed]
14. Fan, Q.; Cai, Z.; Wang, G. Plume Noise Suppression Algorithm for Missile-Borne Star Sensor Based on Star Point Shape and Angular Distance between Stars. *Sensors* **2019**, *19*, 3838. [CrossRef] [PubMed]
15. Opromolla, R.; Fasano, G.; Rufino, G.; Grassi, M. A Model-Based 3D Template Matching Technique for Pose Acquisition of an Uncooperative Space Object. *Sensors* **2015**, *15*, 6360–6382. [CrossRef] [PubMed]
16. Chen, C.; Nie, H.; Chen, J.; Wang, X. A Velocity-Based Impedance Control System for a Low Impact Docking Mechanism (LIDM). *Sensors* **2014**, *14*, 22998–23016. [CrossRef] [PubMed]
17. Sandberg, A.; Sands, T. Autonomous Trajectory Generation Algorithms for Spacecraft Slew Maneuvers. *Aerospace* **2022**, *9*, 135. [CrossRef]
18. Slotine, J.-J.E.; Weiping, L. *Applied Nonlinear Control*; Prentice-Hall: Hoboken, NJ, USA, 1991.
19. Fossen, T. Comments on Hamiltonian adaptive control of spacecraft by Slotine, J.J.E. and Di Benedetto, M.D. *IEEE Trans. Autom. Control* **1993**, *38*, 671–672. [CrossRef]
20. Sands, T.; Kim, J.; Agrawal, B. Spacecraft fine tracking pointing using adaptive control. In Proceedings of the 58th International Astronautical Congress, Hyderabad, India, 24–28 September 2007; International Astronautical Federation: Paris, France, 2007.
21. Sands, T.; Lorenz, R. Physics-Based Automated Control of Spacecraft. In Proceedings of the AIAA Space Conference & Exposition, Pasadena, CA, USA, 14–17 September 2009.
22. Sands, T.; Kim, J.J.; Agrawal, B.N. Spacecraft Adaptive Control Evaluation. In Proceedings of the Infotech@ Aerospace, Garden Grove, CA, USA, 19–21 June 2012; American Institute of Aeronautics and Astronautics: Reston, VA, USA, 2012; pp. 2012–2476.
23. Raigoza, K.; Sands, T. Autonomous Trajectory Generation Comparison for De-Orbiting with Multiple Collision Avoidance. *Sensors* **2022**, *22*, 7066. [CrossRef] [PubMed]

24. Wilt, E.; Sands, T. Microsatellite Uncertainty Control Using Deterministic Artificial Intelligence. *Sensors* **2022**, *22*, 8723. [CrossRef] [PubMed]
25. Ross, I.M. *A Primer on Pontryagin's Principle in Optimal Control*; Collegiate Publisher: New York, NY, USA, 2015.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.