UNIVERSITY OF COLORADO AT BOULDER

ASEN 5044 - STATISTICAL STATE ESTIMATION FOR DYNAMICAL SYSTEMS

FALL 2024

PROFESSOR: DR. NISAR AHMED

Project Report 1 (Cooperative Air-Ground Robot Localization)

TEAM MEMBERS:

NICHOLAS MARTINEZ

WHIT WHITTALL

MICHAEL BERNABEI

December 13, 2024

Contents

List of Figures

List of Algorithms

1 PROGRESS REPORTS

1.1 PROGRESS REPORT FOR PART I

Team Member Contributions
Nicholas Martinez:

1. Nonlinear System Modeling with ODE45

Whit Whittall:

1. Linearized DT LTV System Modeling

Micheal Bernabei

1. Computation of CT Jacobian Matrices

Our group is nearly complete with Part I Deterministic System Analysis. At this time, we've computed the CT Jacobian matrices. This yielded a time varying system, so we have skipped observability and stability analysis. We performed a full nonlinear simulation of the system dynamics. We simulated the linearized DT dynamics and validated these results against the nonlinear simulation and the posted "solution sketches" on canvas. Our results match the posted results on canvas. The only thing we have not done at this stage is simulated the measurement models.

1.2 PROGRESS REPORT FOR PART II

Team Member Contributions
Nicholas Martinez:

1. tbd

Whit Whittall:

1. tbd

Micheal Bernabei

1. Truth Model Testing

tbd

2 Part I. of Cooperative Air-Ground Robot Localization

2.1 Part I

We are given the Equation Of Motion (EOM) for the Unmanned Ground Vehicle (UGV). The EOM is,

$$\begin{aligned}
\dot{\xi_g} &= v_g \cos\theta_g + \tilde{w}_{x,g} \\
\dot{\eta_g} &= v_g \sin\theta_g + \tilde{w}_{y,g} \\
\dot{\theta_g} &= \frac{v_g}{L} \tan\phi_g + \tilde{w}_{\omega,g}
\end{aligned}$$

and for the Unmanned Aerial Vehicle (UAV) we have the following EOM,

$$\begin{aligned}
\dot{\xi_a} &= v_a \cos\theta_a + \tilde{w}_{x,a} \\
\dot{\eta_a} &= v_a \sin\theta_a + \tilde{w}_{y,a} \\
\dot{\theta_a} &= \omega_a + \tilde{w}_{\omega,a}
\end{aligned}$$

where $\tilde{w}_a = [\tilde{w}_{x,a}, \tilde{w}_{y,a}, \tilde{w}_{\omega,a}]^T$ and $\tilde{w}_g = [\tilde{w}_{x,g}, \tilde{w}_{y,g}, \tilde{w}_{\omega,g}]^T$ are the process noise for the UAV And UGV respectively. We are also given the following sensing model,

$$y(t) = \begin{bmatrix} \arctan\left(\frac{\eta_a - \eta_g}{\xi_a - \xi_g}\right) - \theta_g \\ \sqrt{(\xi_g - \xi_a)^2 + (\eta_g - \eta_a)^2} \\ \arctan\left(\frac{\eta_g - \eta_a}{\xi_g - \xi_a}\right) - \theta_a \\ \xi_a \\ \eta_a \end{bmatrix} + \tilde{\mathbf{v}}(t)$$

where $\tilde{\mathbf{v}}(t) \in \mathbb{R}^5$ is the sensor error vector. Finally, we are given the combined states, control inputs, and disturbance inputs as,

$$\begin{aligned}
\mathbf{x}(t) &= [\xi_g \quad \eta_g \quad \theta_g \quad \xi_a \quad \eta_a \quad \theta_a]^T, \\
\mathbf{u}(t) &= [\mathbf{u}_g \ \ \mathbf{u}_a]^T, \\
\tilde{\mathbf{w}}(t) &= [\tilde{\mathbf{w}}_\mathbf{g} \ \ \tilde{\mathbf{w}}_\mathbf{a}]^T
\end{aligned}$$

The state is $x = [\xi_g \ \eta_g \ \theta_g \ \xi_a \ \eta_a \ \theta_a]^T = [x_1 \ x_2 \, x_3 \ x_4 \ x_5 \ x_6]^T$ and our inputs $u = [\mathbf{u}_g \ \mathbf{u}_a]^T = [v_g \ \ \phi_g \ \ v_a \ \ \phi_a]^T = [u_1 \ \ u_2 \ \ u_3 \ \ u_4]^T$. We then have the following after substituting in our state and input variables,

$$\dot{x} = \begin{bmatrix} \dot{\xi}_g \\ \dot{\eta}_g \\ \dot{\theta}_g \\ \dot{\xi}_a \\ \dot{\eta}_a \\ \dot{\theta}_a \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} \mathcal{F}_1(x,u) \\ \mathcal{F}_2(x,u) \\ \mathcal{F}_3(x,u) \\ \mathcal{F}_4(x,u) \\ \mathcal{F}_5(x,u) \\ \mathcal{F}_6(x,u) \end{bmatrix} = \begin{bmatrix} u_1 \cos x_3 \\ u_1 \sin x_3 \\ \frac{u_1}{L} \tan u_2 \\ u_3 \cos x_6 \\ u_3 \sin x_6 \\ u_4 \end{bmatrix}$$

$$y = \begin{bmatrix} \arctan\left(\frac{x_5 - x_2}{x_4 - x_1}\right) - x_3 \\ \sqrt{(x_1 - x_4)^2 + (x_2 - x_5)^2} \\ \arctan\left(\frac{x_2 - x_5}{x_1 - x_4}\right) - x_6 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \mathcal{H}_1(x,u) \\ \mathcal{H}_2(x,u) \\ \mathcal{H}_3(x,u) \\ \mathcal{H}_4(x,u) \\ \mathcal{H}_5(x,u) \end{bmatrix}$$

We now need to compute the partials of $\mathcal{F}_{1 \ldots 6}$ with respect to x,

$$\begin{array}{llllll}
\frac{\partial \mathcal{F}_1}{\partial x_1} = 0 & \frac{\partial \mathcal{F}_2}{\partial x_1} = 0 & \frac{\partial \mathcal{F}_3}{\partial x_1} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_1} = 0 & \frac{\partial \mathcal{F}_5}{\partial x_1} = 0 & \frac{\partial \mathcal{F}_6}{\partial x_1} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial x_2} = 0 & \frac{\partial \mathcal{F}_2}{\partial x_2} = 0 & \frac{\partial \mathcal{F}_3}{\partial x_2} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_2} = 0 & \frac{\partial \mathcal{F}_5}{\partial x_2} = 0 & \frac{\partial \mathcal{F}_6}{\partial x_2} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial x_3} = -u_1 \sin x_3 & \frac{\partial \mathcal{F}_2}{\partial x_3} = u_1 \cos x_3 & \frac{\partial \mathcal{F}_3}{\partial x_3} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_3} = 0 & \frac{\partial \mathcal{F}_5}{\partial x_3} = 0 & \frac{\partial \mathcal{F}_6}{\partial x_3} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial x_4} = 0 & \frac{\partial \mathcal{F}_2}{\partial x_4} = 0 & \frac{\partial \mathcal{F}_3}{\partial x_4} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_4} = 0 & \frac{\partial \mathcal{F}_5}{\partial x_4} = 0 & \frac{\partial \mathcal{F}_6}{\partial x_4} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial x_5} = 0 & \frac{\partial \mathcal{F}_2}{\partial x_5} = 0 & \frac{\partial \mathcal{F}_3}{\partial x_5} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_5} = 0 & \frac{\partial \mathcal{F}_5}{\partial x_5} = 0 & \frac{\partial \mathcal{F}_6}{\partial x_5} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial x_6} = 0 & \frac{\partial \mathcal{F}_2}{\partial x_6} = 0 & \frac{\partial \mathcal{F}_3}{\partial x_6} = 0 & \frac{\partial \mathcal{F}_4}{\partial x_6} = -u_3 \sin x_6 & \frac{\partial \mathcal{F}_5}{\partial x_6} = u_3 \cos x_6 & \frac{\partial \mathcal{F}_6}{\partial x_6} = 0
\end{array}$$

and with respect to u,

$$\begin{array}{llllll}
\frac{\partial \mathcal{F}_1}{\partial u_1} = \cos x_3 & \frac{\partial \mathcal{F}_2}{\partial u_1} = \sin x_3 & \frac{\partial \mathcal{F}_3}{\partial u_1} = \frac{\tan u_2}{L} & \frac{\partial \mathcal{F}_4}{\partial u_1} = 0 & \frac{\partial \mathcal{F}_5}{\partial u_1} = 0 & \frac{\partial \mathcal{F}_6}{\partial u_1} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial u_2} = 0 & \frac{\partial \mathcal{F}_2}{\partial u_2} = 0 & \frac{\partial \mathcal{F}_3}{\partial u_2} = \frac{u_1}{L} \sec^2 u_2 & \frac{\partial \mathcal{F}_4}{\partial u_2} = 0 & \frac{\partial \mathcal{F}_5}{\partial u_2} = 0 & \frac{\partial \mathcal{F}_6}{\partial u_2} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial u_3} = 0 & \frac{\partial \mathcal{F}_2}{\partial u_3} = 0 & \frac{\partial \mathcal{F}_3}{\partial u_3} = 0 & \frac{\partial \mathcal{F}_4}{\partial u_3} = \cos x_6 & \frac{\partial \mathcal{F}_5}{\partial u_3} = \sin x_6 & \frac{\partial \mathcal{F}_6}{\partial u_3} = 0 \\
\frac{\partial \mathcal{F}_1}{\partial u_4} = 0 & \frac{\partial \mathcal{F}_2}{\partial u_4} = 0 & \frac{\partial \mathcal{F}_3}{\partial u_4} = 0 & \frac{\partial \mathcal{F}_4}{\partial u_4} = 0 & \frac{\partial \mathcal{F}_5}{\partial u_4} = 0 & \frac{\partial \mathcal{F}_6}{\partial u_4} = 1
\end{array}$$

finally we compute $\mathcal{H}_{1 \ldots 5}$ with respect to x. In the following we show the two most complex partial derivative computations. The remaining partials were computed using similar techniques and therefore we omit them for brevity.

Utilize the chain rule with $u = (\frac{x_5 - x_2}{x_4 - x_1})$, then,

$$
\begin{aligned}
\frac{\partial \mathcal{H}_1}{\partial x_1} &= \frac{\partial \mathcal{H}_1}{\partial u} \times \frac{\partial u}{\partial x_1} \\
&= \left(\frac{1}{(\frac{x_5 - x_2}{x_4 - x_1})^2 + 1} \right) \times \left(\frac{0 \times (x_4 - x_1) - (x_5 - x_2) \times -1}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{(\frac{x_5 - x_2}{x_4 - x_1})^2 + 1} \right) \times \left(\frac{(x_5 - x_2)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{\frac{(x_5 - x_2)^2}{(x_4 - x_1)^2} + 1} \right) \times \left(\frac{(x_5 - x_2)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{\frac{(x_5 - x_2)^2 + (x_4 - x_1)^2}{(x_4 - x_1)^2}} \right) \times \left(\frac{(x_5 - x_2)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{(x_4 - x_1)^2}{(x_5 - x_2)^2 + (x_4 - x_1)^2} \right) \times \left(\frac{(x_5 - x_2)}{(x_4 - x_1)^2} \right) \\
&= \frac{x_5 - x_2}{(x_5 - x_2)^2 + (x_4 - x_1)^2}
\end{aligned}
$$

and we show the case when the partial we are computing is in the numerator,

$$
\begin{aligned}
\frac{\partial \mathcal{H}_1}{\partial x_2} &= \frac{\partial \mathcal{H}_1}{\partial u} \times \frac{\partial u}{\partial x_2} \\
&= \left(\frac{1}{(\frac{x_5 - x_2}{x_4 - x_1})^2 + 1} \right) \times \left(\frac{-1 \times (x_4 - x_1) - (x_5 - x_2) \times 0}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{(\frac{x_5 - x_2}{x_4 - x_1})^2 + 1} \right) \times \left(-\frac{(x_4 - x_1)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{\frac{(x_5 - x_2)^2}{(x_4 - x_1)^2} + 1} \right) \times \left(-\frac{(x_4 - x_1)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{1}{\frac{(x_5 - x_2)^2 + (x_4 - x_1)^2}{(x_4 - x_1)^2}} \right) \times \left(-\frac{(x_4 - x_1)}{(x_4 - x_1)^2} \right) \\
&= \left(\frac{(x_4 - x_1)^2}{(x_5 - x_2)^2 + (x_4 - x_1)^2} \right) \times \left(-\frac{(x_4 - x_1)}{(x_4 - x_1)^2} \right) \\
&= -\frac{x_4 - x_1}{(x_5 - x_2)^2 + (x_4 - x_1)^2}
\end{aligned}
$$

Now, let $u = (x_1 - x_4)^2 + (x_2 - x_5)^2$, $v = (x_1 - x_4)^2$, and $w = (x_1 - x_4)$, then for our next complex partial we have,

$$
\begin{aligned}
\frac{\partial \mathcal{H}_1}{\partial x_1} &= \frac{\partial \mathcal{H}_1}{\partial u} \times \frac{\partial u}{\partial v} \times \frac{\partial v}{\partial w} \times \frac{\partial w}{\partial x_1} \\
&= \left(\frac{1}{2} \times \frac{1}{\sqrt{u}}\right) \times (1) \times \Big(2(w)\Big) \times (1) \\
&= \left(\frac{1}{2} \times \frac{1}{\sqrt{(x_1 - x_4)^2 + (x_2 - x_5)^2}}\right) \times (1) \times \Big(2(x_1 - x_4)\Big) \times (1) \\
&= \frac{x_1 - x_4}{\sqrt{(x_1 - x_4)^2 + (x_2 - x_5)^2}}
\end{aligned}
$$

therefore we have the following Jacobians,

$$
\frac{\partial f}{\partial x} = \begin{bmatrix}
0 & 0 & -u_1 \sin x_3 & 0 & 0 & 0 \\
0 & 0 & u_1 \cos x_3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & u_3 \sin x_6 \\
0 & 0 & 0 & 0 & 0 & -u_3 \cos x_6 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
\frac{\partial f}{\partial u} = \begin{bmatrix}
\cos x_3 & 0 & 0 & 0 \\
\sin x_3 & 0 & 0 & 0 \\
\frac{\tan u_2}{L} & \frac{u_1 \sec^2 u_2}{L} & 0 & 0 \\
0 & 0 & \cos x_6 & 0 \\
0 & 0 & \sin x_6 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

$$
\frac{\partial h}{\partial x} = \begin{bmatrix}
\frac{x_5 - x_2}{(x_4 - x_1)^2 + (x_2 - x_5)^2} & -\frac{x_4 - x_1}{(x_4 - x_1)^2 + (x_2 - x_5)^2} & -1 & -\frac{x_5 - x_2}{(x_4 - x_1)^2 + (x_5 - x_2)^2} & \frac{x_4 - x_1}{(x_4 - x_1)^2 + (x_5 - x_2)^2} & 0 \\
\frac{x_1 + x_4}{\sqrt{(x_1 + x_4)^2 + (x_2 + x_5)^2}} & \frac{x_2 + x_5}{\sqrt{(x_1 + x_4)^2 + (x_2 + x_5)^2}} & 0 & \frac{x_1 + x_4}{\sqrt{(x_1 + x_4)^2 + (x_2 + x_5)^2}} & \frac{x_2 + x_5}{\sqrt{(x_1 + x_4)^2 + (x_2 + x_5)^2}} & 0 \\
-\frac{x_2 - x_5}{(x_1 - x_4)^2 + (x_2 - x_5)^2} & \frac{x_1 - x_4}{(x_1 - x_4)^2 + (x_2 - x_5)^2} & 0 & \frac{x_2 - x_5}{(x_1 - x_4)^2 + (x_2 - x_5)^2} & -\frac{x_1 - x_4}{(x_1 - x_4)^2 + (x_2 - x_5)^2} & -1 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
$$

$$
\frac{\partial h}{\partial u} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
$$

2.2 Part II

The CT Jacobian matrices show that the cooperative air-ground localization system is time varying. For this reason, we skipped observability and stability analysis. Furthermore, it is dependent on the state and inputs. In order simulate the DT LTV system, we linearize around a known nominal trajectory. We find this trajectory by solving the nonlinear ODEs with inital state $x = [10, 0, \pi/2, -60, 0, -\pi/2]^T$ and inputs $u = [2, -\pi/18, 12, \pi/2]^T$. We define "eulerized" DT Jacobians,

$$\tilde{F}_k = \Delta T * \tilde{A}_{nom[k]}$$

$$\tilde{G}_k = \Delta T * \tilde{B}_{nom[k]}$$

Where,

$$\tilde{A}_{nom[k]} = \frac{\partial f}{\partial x}\Big|_{t=t_k, nom[k]}$$

$$\tilde{B}_{nom[k]} = \frac{\partial f}{\partial u}\Big|_{t=t_k, nom[k]}$$

With these DT jacobians, we can model the system as,

$$x(k+1) = x_{nom,k} + \tilde{F}_k \delta x_k + \tilde{G}_k \delta u_k$$

The full state results of modeling using initial perturbation state conditions of $\delta x = [0, 1, 0, 0, 0, 0.1]$ are shown below.

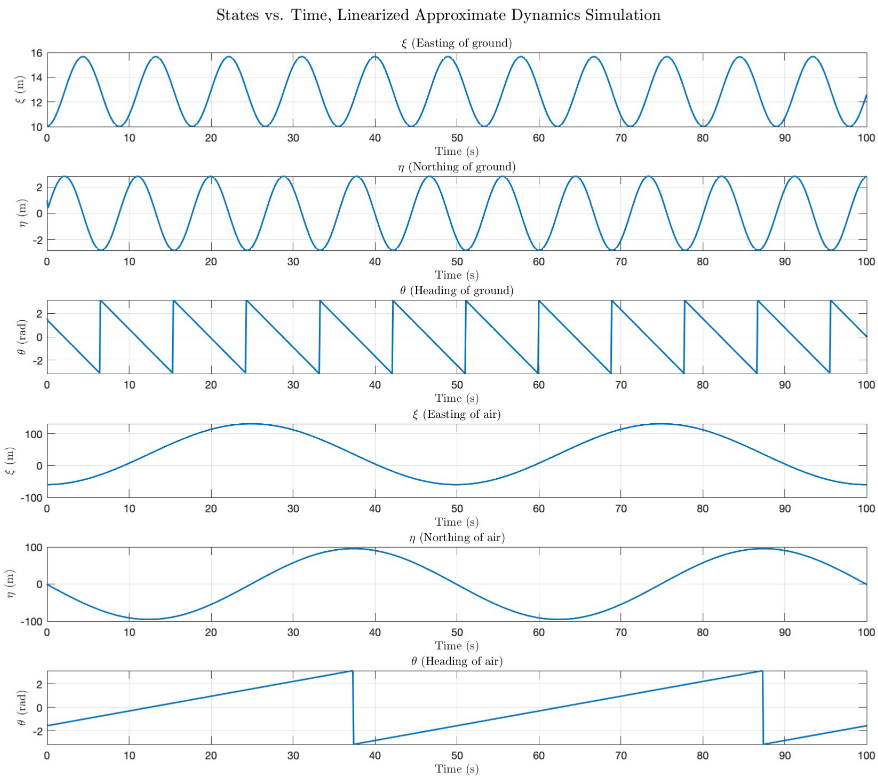

Figure 2.1: Linearized Full State Dynamics

The perturbation states from the same simulation are shown below.

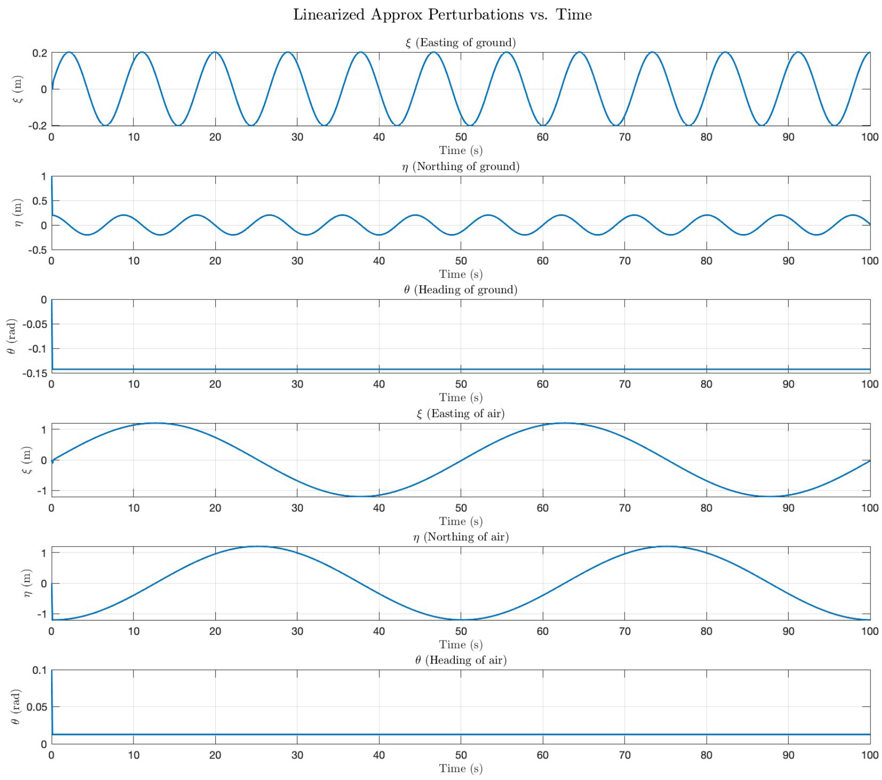

Figure 2.2: Linearized Perturbation State Dynamics

2.3 Part III

To validate the DT LTV model we simulated in part 2, we performed a full nonlinear simulation of the system with the same initial perturbation state conditions of $\delta x = [0, 1, 0, 0, 0, 0.1]$. We performed this simulation using ODE45 in MATLAB. The results of the simulation are shown below.

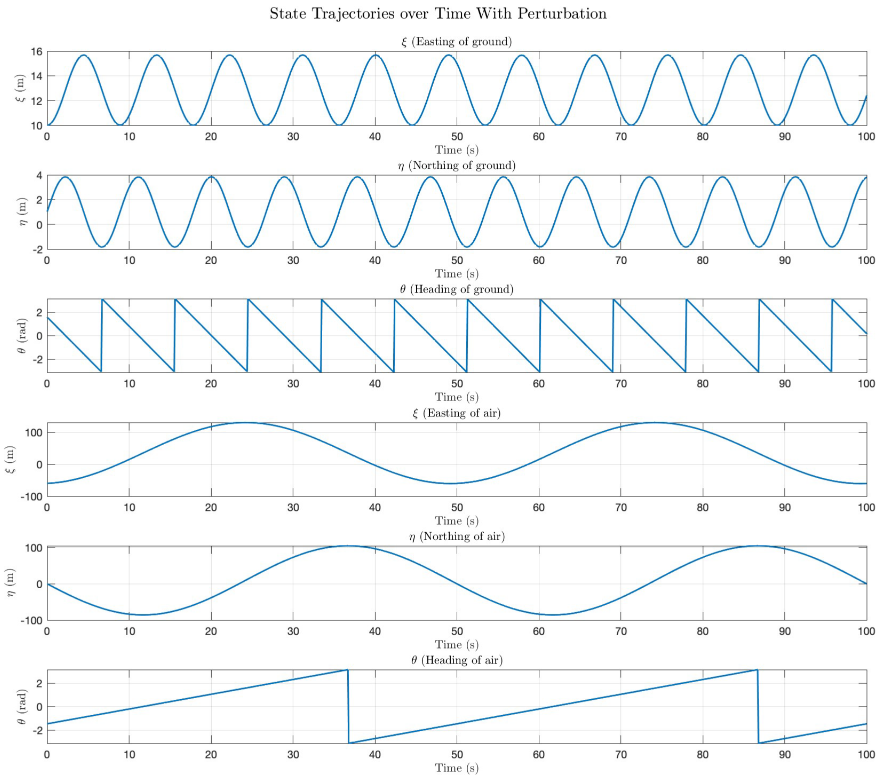

Figure 2.3: Non-linear Simulated State Dynamics

The linearized approximate dynamics simulation results are very close to that of the nonlinear simulation. By closely comparing the two plots, we were able to observe a very slight phase shift between the states of the two simulations, but the linearized simulation matches the nonlinear simulation very well.

3 Part II. Stochastic Nonlinear Filtering

3.1 Part a.

3.2 Part b.

3.2.1 Explanation of NEES Test Statistic and Plot

The pseudo code for how the test was setup is located here NEES Test Setup Psuedo Code. Essentially, we followed the model provided in lecture 29 of class. That is, we have a script that reads in x_k (ground truth) and simulated measurements y_k to our Kalman Filter code. The Kalman filter code then generates $(\hat{x}_k, P_k^+)$ during the predictor-corrector step of the algorithm. During the predictor-corrector step we produce the NEES test statistic $\epsilon_{x,k}$ and store it for the simulation run. Then at the end of the Monte Carlo simulations we compute $\bar{\epsilon}_{x,k}$ for each run and store these values for plotting against the confidence bounds computed by the χ^2 function provided by MATLAB. The confidence bound has values r_1x and r_2x. They are centered on a value of 6 in the NEES case because our Degrees-of-Freedom (DOF) is 6 in the case of our dynamical system.
We are currently in the process of tuning our LKF filter and therefore expect that our $\bar{\epsilon}_{x,k}$ values will be outside the confidence bounds for the NEES statistic centered on 6. See NEES Plot for LKF.

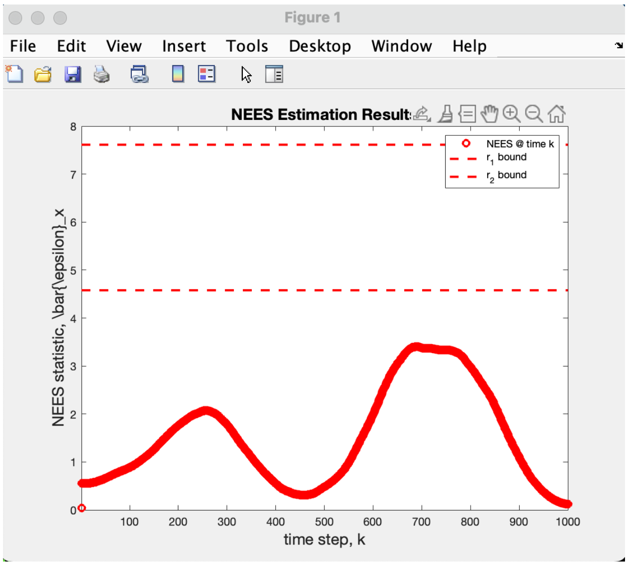

Figure 3.1: NEES Plot for LKF

Algorithm 1 Truth Model Test (TMT) Algorithm for NEES

numberOfSimulations = 100
NEESSimRun is allocated to length of numberOfSimulations
numberTimeSteps = 1000
while numberOfSimulations > 0 **do**
    NEESStorage is initialized to the length of numberTimeSteps
    while numberTimeSteps > 0 **do**
        Perform LKF perdictor-corrector step
        Compute NEES statistic
        Save NEES Statistic in NEESStorage
    end while
    save NEESStorage in NEESSimRun
end while
Allocate NEESBar with length equal to simulation runs
Compute NEES mean for each run in NEESSimRun and put in NEESBar
Tune α for use in MATLAB χ^2 call
Calculate degrees of freedom for MATLAB χ^2 call
Compute confidence interval using χ^2 function in MATLAB
Plot NEES test

3.3 Part C.

The NIS test statistic mirrors the algorithmic steps and computations required by the NEES statistic. Hence, we will omit the step-by-step details for the NIS computation and ask the reader to see the previous section for exact steps in computing the statistic. Also see NIS Test Setup Psuedo Code for pseudo code. The main difference between the NEES And NIS test statistic is that the NIS test statistic works with measurement values versus state. Hence, we have $\epsilon_{y,k}$ instead of $\epsilon_{x,k}$.
We are currently in the process of tuning our LKF filter and therefore expect that our $\bar{\epsilon}_{y,k}$ values will be outside the confidence bounds for the NIS statistic centered on 5. We have also noticed that they are dramicatlly deviated from the confidence bound and are taking steps to ensure the statistic is being computed correctly. See NIS Plot for LKF.

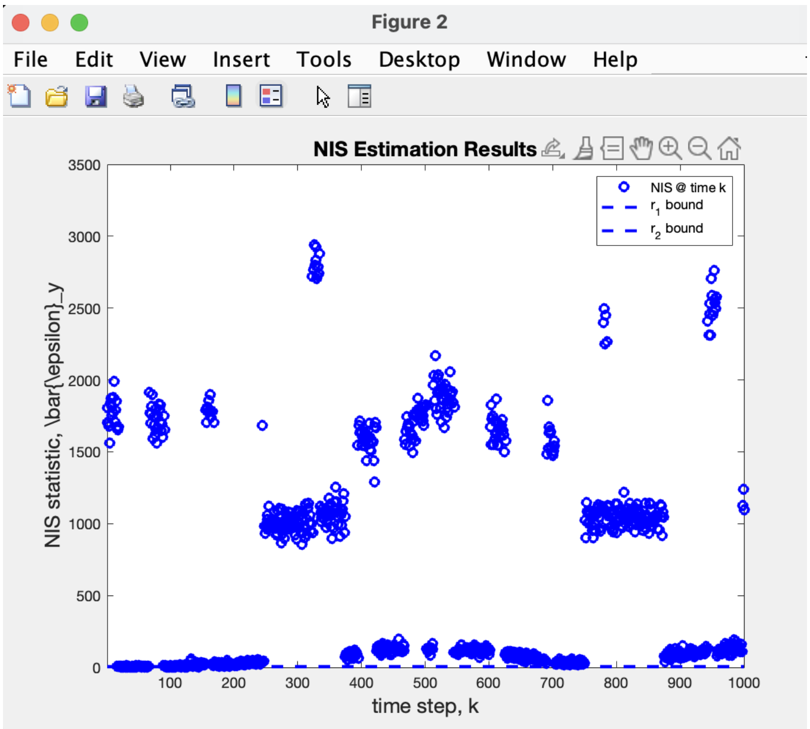

Figure 3.2: NIS Plot for LKF

3.3.1 Explanation of NIS Test Statistic And Plot

Algorithm 2 Truth Model Test (TMT) Algorithm for NIS

```
numberOfSimulations = 100
NISSimRun is allocated to length of numberOfSimulations
numberTimeSteps = 1000
while numberOfSimulations > 0 do
    NISStorage is initialized to the length of numberTimeSteps
    while numberTimeSteps > 0 do
        Perform LKF perdictor-corrector step
        Compute NIS statistic
        Save NIS Statistic in NISStorage
    end while
    save NISStorage in NISSimRun
end while
Allocate NISBar with length equal to simulation runs
Compute NIS mean for each run in NISSimRun and put in NISBar
Tune α for use in MATLAB χ^2 call
Calculate degrees of freedom for MATLAB χ^2 call
Compute confidence interval using χ^2 function in MATLAB
Plot NIS test
```

4 Appendix

We have attached a link to our team GitHub below. All code used to simulate the system and generate plots can be found at:
https://github.com/bernabei24/final_project_asen_5044_f24/tree/main