AAE 590FE Final Project: Estimating Rolling Resistance and Position for a Balancing Robot

Henry Emmert
Purdue University
West Lafayette, IN
hemmert@purdue.edu

***Abstract*—A two-wheeled balancing robot is a useful platform for experimenting with control and estimation algorithms in a real-world application. This paper is based on a balancing robot built as a personal project to test control algorithms. When testing these control algorithms, it became apparent that the optimal control gains often depend heavily on the rolling resistance between the wheels and the ground. In particular, balancing on a hard floor and a rug require significantly different control gains. The robot also had a tendency to drift across the floor because only the attitude angle was being controlled. These observations motivate this paper, which investigates estimating the rolling resistance and position of the robot using measurements from a low-cost IMU and ultrasonic distance sensor. The extended Kalman filter (EKF) and unscented Kalman filter (UKF) are investigated as estimation algorithms, and further analyses study filter consistency, model mismatch, and allowable noise for the distance sensor.**

I. Introduction

The balancing robot in this paper has two coaxial wheels and a center of mass well above the wheel axis, making the uncontrolled dynamics unstable. Previous work has been done to implement PID, LQR, and LQE controllers to successfully balance the robot using an IMU to measure the attitude angle, and two small motors (one for each wheel) as control inputs. With only an IMU, the position of the robot is not observable. Therefore, an ultrasonic distance sensor is added in this paper to measure the distance to a wall in front of the robot. An encoder could also solve this observability problem, but the distance sensor was chosen to make the problem more interesting.

This system will be investigated by modeling the nonlinear dynamics, measurements, and estimation and control algorithms in a Simulink model. Previous work was based on the simplified dynamics of an inverted pendulum on a cart, but this neglects the inertia properties of the wheel and robot body, and does not include a term for rolling resistance. The nonlinear dynamics in this paper will be based on a more accurate set of dynamics equations derived in [1]. These include inertia properties and rolling resistance, along with a term for motor friction, which all improve the accuracy of the model.

This model will be used for practical analyses related to implementing a state estimation algorithm on a physical robot. These include testing different estimation algorithms with varying accuracy and computational cost, studying the effect of model mismatch on estimation performance, and testing different noise levels for the distance sensor to help define requirements for choosing the sensor.

The following problem formulation section will provide a detailed description of the system, the equations of motion, and measurement equations. The methodology section will describe the sensor noise statistics, estimation algorithms, filter tuning, and simulation setup. The results section will include verification and comparison of the estimation algorithms, Monte Carlo simulations with model mismatches, and a sweep of different noise levels for the distance sensor. The paper will conclude with some key takeaways, especially as they relate to the real-world implementation of these estimation algorithms.

II. Problem Formulation

The balancing robot has two coaxial wheels, each connected to a motor mounted on the body of the robot. The robot body has a center of mass significantly above the wheels, so the system behaves similarly to an inverted pendulum on a cart.

The model of the robot in this paper reduces the system to 2 degrees of freedom; the pitch angle of the body (θ) and the rotation angle of the wheel (φ). This assumes that the robot only moves forwards and backwards, and that it can only tilt in its pitch axis. It is also assumed that the wheel does not slip on the ground, so the wheel rotation angle is linearly proportional to the horizontal position of the robot (x). This position is measured from the center of the wheel.

θ	body pitch angle
φ	wheel rotation angle
x	horizontal position
τ_0	applied torque
m	mass of robot body
m_w	mass of wheel
R	radius of wheel
L	distance from center of wheel to center of mass
I	inertia of robot body
I_w	inertia of wheel

The IMU and the distance sensor are both located at a distance l_s from the center of the wheel, between the wheel and the center of mass. The accelerometer and gyroscope measurements will be passed through a complimentary filter, and the output of that filter will be used as body pitch angle measurements in the Kalman filters. This avoids the

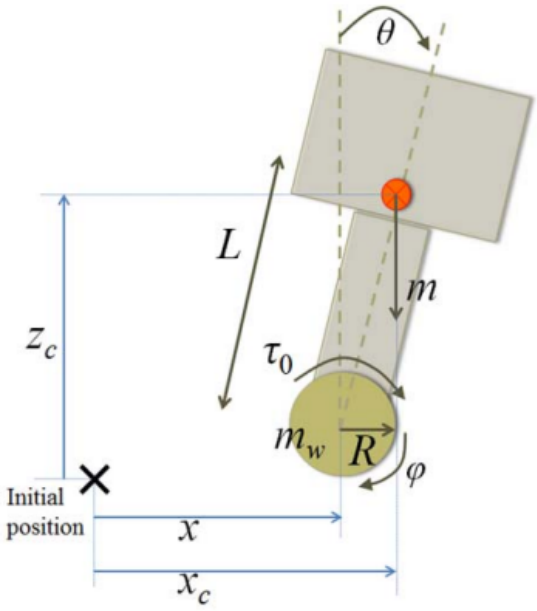

Fig. 1. Robot diagram from [1].

complexity of inertial navigation filtering for the purposes of this paper. Therefore, the Kalman filters will treat this measurement as a one-to-one mapping with the body pitch angle state (θ). However, a true measurement function is still derived and used for generating the IMU measurements so that the estimates coming from the complimentary filter are realistic. The accelerometer measurements are generated by calculating the acceleration at the IMU and adding gaussian white noise. Gyroscope measurements are generated by adding gaussian white noise to the body angle rate state ($\dot{\theta}$). The standard deviation of the white noise is roughly based on the performance of an MPU-6050 IMU.

The distance sensor is pointed +90 degrees clockwise from the line connecting the center of the wheel to the center of mass, meaning that it points in the $+x$ direction when $\theta = 0$. The sensor measures the distance between itself and a fixed wall some distance x_w from the initial position. The measurement function that maps the current state to the true distance sensor measurement is used for generating measurements, as well as in the update step of the Kalman filters. Gaussian white noise is added to the distance measurement with a standard deviation of 1 cm.

A. Nonlinear Dynamics Equations

The non-linear equations of motion from [1] are:

$$M = \begin{bmatrix} I_w + (m_w + m)R^2 & mRL\cos\theta \\ mRL\cos\theta & I + mL^2 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \tag{1}$$

$$N = \begin{bmatrix} 0 & -mRL\sin\theta\,\dot{\theta} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & N_{12} \\ 0 & 0 \end{bmatrix} \tag{2}$$

$$M \begin{bmatrix} \ddot{\varphi} \\ \ddot{\theta} \end{bmatrix} + N \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ -mgL\sin\theta \end{bmatrix} = \begin{bmatrix} \mu \\ \chi \end{bmatrix} \tag{3}$$

Where the torques μ and χ can be written in terms of the rolling damping ratio (β_r) and friction damping ratio (β_f). The rolling damping ratio models the rolling resistance between the wheel and the ground, and the friction damping ratio models the friction in the motors.

$$\begin{bmatrix} \mu \\ \chi \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tau_0 - \begin{bmatrix} \beta_r + \beta_f & -\beta_f \\ -\beta_f & \beta_f \end{bmatrix} \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \end{bmatrix} \tag{4}$$

(3) and (4) can be rewritten as:

$$\ddot{\varphi} = \frac{\mu - M_{12}\ddot{\theta} - N_{12}\dot{\theta}}{M_{11}} \tag{5}$$

$$\ddot{\theta} = \frac{[M_{21}(\mu - N_{12}\dot{\theta}) - M_{11}(\chi + mgLsin\theta)]}{M_{21}M_{12} - M_{11}M_{22}} \tag{6}$$

This gives us the full nonlinear dynamics:

$$\frac{d}{dt}\begin{bmatrix} \varphi \\ \theta \\ \dot{\varphi} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \frac{\mu - M_{12}\ddot{\theta} - N_{12}\dot{\theta}}{M_{11}} \\ \frac{[M_{21}(\mu - N_{12}\dot{\theta}) - M_{11}(\chi + mgLsin\theta)]}{M_{21}M_{12} - M_{11}M_{22}} \end{bmatrix} \tag{7}$$

The Kalman filter algorithms will estimate the rolling damping ratio (β_r), so this is added as a fifth state. This leads to the full continuous dynamics function:

$$\frac{d}{dt}\begin{bmatrix} \varphi \\ \theta \\ \dot{\varphi} \\ \dot{\theta} \\ \beta_r \end{bmatrix} = \begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \frac{\mu - M_{12}\ddot{\theta} - N_{12}\dot{\theta}}{M_{11}} \\ \frac{[M_{21}(\mu - N_{12}\dot{\theta}) - M_{11}(\chi + mgLsin\theta)]}{M_{21}M_{12} - M_{11}M_{22}} \\ 0 \end{bmatrix} \tag{8}$$

$$\frac{d\boldsymbol{x}}{dt} = f(\boldsymbol{x}) \tag{9}$$

B. Dynamics Jacobian

The dynamics Jacobian (F) is solved analytically using the following matrix of partial derivatives:

$$F = \begin{bmatrix} \frac{\partial f_1}{\partial \boldsymbol{x_1}} & \frac{\partial f_1}{\partial \boldsymbol{x_2}} & \frac{\partial f_1}{\partial \boldsymbol{x_3}} & \frac{\partial f_2}{\partial \boldsymbol{x_4}} & \frac{\partial f_2}{\partial \boldsymbol{x_5}} \\ \frac{\partial f_2}{\partial \boldsymbol{x_1}} & \frac{\partial f_2}{\partial \boldsymbol{x_2}} & \frac{\partial f_2}{\partial \boldsymbol{x_3}} & \frac{\partial f_2}{\partial \boldsymbol{x_4}} & \frac{\partial f_2}{\partial \boldsymbol{x_5}} \\ \frac{\partial f_3}{\partial \boldsymbol{x_1}} & \frac{\partial f_3}{\partial \boldsymbol{x_2}} & \frac{\partial f_3}{\partial \boldsymbol{x_3}} & \frac{\partial f_3}{\partial \boldsymbol{x_4}} & \frac{\partial f_3}{\partial \boldsymbol{x_5}} \\ \frac{\partial f_4}{\partial \boldsymbol{x_1}} & \frac{\partial f_4}{\partial \boldsymbol{x_2}} & \frac{\partial f_4}{\partial \boldsymbol{x_3}} & \frac{\partial f_4}{\partial \boldsymbol{x_4}} & \frac{\partial f_4}{\partial \boldsymbol{x_5}} \\ \frac{\partial f_5}{\partial \boldsymbol{x_1}} & \frac{\partial f_5}{\partial \boldsymbol{x_2}} & \frac{\partial f_5}{\partial \boldsymbol{x_3}} & \frac{\partial f_5}{\partial \boldsymbol{x_4}} & \frac{\partial f_5}{\partial \boldsymbol{x_5}} \end{bmatrix}$$

Filling in the trivial partial derivatives:

$$F = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \frac{\partial f_3}{\partial \boldsymbol{x_2}} & \frac{\partial f_3}{\partial \boldsymbol{x_3}} & \frac{\partial f_3}{\partial \boldsymbol{x_4}} & \frac{\partial f_3}{\partial \boldsymbol{x_5}} \\ 0 & \frac{\partial f_4}{\partial \boldsymbol{x_2}} & \frac{\partial f_4}{\partial \boldsymbol{x_3}} & \frac{\partial f_4}{\partial \boldsymbol{x_4}} & \frac{\partial f_4}{\partial \boldsymbol{x_5}} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

The remaining nontrivial partial derivatives are evaluated by finding the partial derivative of each individual term (M_{11}, M_{12}, M_{21}, M_{22}, N_{12}, μ, χ, and $-mgLsin\theta$) with respect to $\boldsymbol{x_2}$, $\boldsymbol{x_3}$, $\boldsymbol{x_4}$, and $\boldsymbol{x_5}$, then combining these partials using the chain rule and quotient rule to get the complete partial derivative.

From (1) and (2):

$$\begin{aligned} M_{11} &= I_w + (m_w + m)R^2 \\ M_{12} &= mRL\cos\theta \\ M_{21} &= M_{12} \\ M_{22} &= I + mL^2 \\ N_{12} &= -mRL\sin\theta\dot{\theta} \end{aligned} \tag{11}$$

The numerator and denominator from (6) are written as:

$$\begin{aligned} \ddot{\theta} &= \frac{n_{\ddot{\theta}}}{d_{\ddot{\theta}}} \\ n_{\ddot{\theta}} &= M_{21}(\mu - N_{12}\dot{\theta}) - M_{11}(x + -mgLsin\theta) \\ d_{\ddot{\theta}} &= M_{21}M_{12} - M_{11}M_{22} \end{aligned} \tag{12}$$

Partial derivatives of M_{12}, N_{12}, and $-mgLsin\theta$:

$$\begin{aligned} \frac{\partial M_{12}}{\partial\theta} &= -mRL\sin\theta \\ \frac{\partial N_{12}}{\partial\theta} &= -mRL\cos\theta\dot{\theta} \\ \frac{\partial N_{12}}{\partial\dot{\theta}} &= -mRL\sin\theta \\ \frac{\partial(-mgLsin\theta)}{\partial\theta} &= -mgL\cos\theta \end{aligned} \tag{13}$$

Partial derivatives of μ and χ:

$$\begin{aligned} \frac{\partial\mu}{\partial\dot{\varphi}} &= -\beta_f - \beta_r \\ \frac{\partial\mu}{\partial\dot{\theta}} &= \beta_f \\ \frac{\partial\mu}{\partial\beta_r} &= -\dot{\varphi} \\ \frac{\partial\chi}{\partial\dot{\varphi}} &= \beta_f \\ \frac{\partial\chi}{\partial\dot{\theta}} &= -\beta_f \end{aligned} \tag{14}$$

Partial derivatives of $n_{\ddot{\theta}}$ and $d_{\ddot{\theta}}$:

$$\begin{aligned} \frac{\partial d_{\ddot{\theta}}}{\partial\theta} &= M_{21}\frac{\partial M_{12}}{\partial\theta} + M_{12}\frac{\partial M_{21}}{\partial\theta} = 2M_{12}\frac{\partial M_{12}}{\partial\theta} \\ \frac{\partial n_{\ddot{\theta}}}{\partial\theta} &= \frac{\partial M_{12}}{\partial\theta}(\mu - N_{12}\dot{\theta}) + M_{12}\left(-\frac{\partial N_{12}}{\partial\theta}\dot{\theta}\right) + M_{11}\frac{\partial(-mgLsin\theta)}{\partial\theta} \\ \frac{\partial n_{\ddot{\theta}}}{\partial\dot{\varphi}} &= M_{12}\frac{\partial\mu}{\partial\dot{\varphi}} - M_{11}\frac{\partial\chi}{\partial\dot{\varphi}} \\ \frac{\partial n_{\ddot{\theta}}}{\partial\dot{\theta}} &= M_{12}\frac{\partial\mu}{\partial\dot{\theta}} - M_{12}\left(\frac{\partial N_{12}}{\partial\dot{\theta}}\dot{\theta} + N_{12}\right) + M_{11}\left(-\frac{\partial\chi}{\partial\dot{\theta}}\right) \\ \frac{\partial n_{\ddot{\theta}}}{\partial\beta_r} &= M_{12}\frac{\partial\mu}{\partial\beta_r}. \end{aligned} \tag{15}$$

Applying the quotient rule to get the partial derivatives of (6), which correspond to the f_4 row of Jacobian elements:

$$\begin{aligned} \frac{\partial f_4}{\partial\boldsymbol{x_2}} &= \frac{\left(\frac{\partial n_{\ddot{\theta}}}{\partial\theta}\right)d_{\ddot{\theta}} - n_{\ddot{\theta}}\left(\frac{\partial d_{\ddot{\theta}}}{\partial\theta}\right)}{d_{\ddot{\theta}}^2} \\ \frac{\partial f_4}{\partial\boldsymbol{x_3}} &= \frac{\left(\frac{\partial n_{\ddot{\theta}}}{\partial\dot{\varphi}}\right)d_{\ddot{\theta}}}{d_{\ddot{\theta}}^2} \\ \frac{\partial f_4}{\partial\boldsymbol{x_4}} &= \frac{\left(\frac{\partial n_{\ddot{\theta}}}{\partial\dot{\theta}}\right)d_{\ddot{\theta}}}{d_{\ddot{\theta}}^2} \\ \frac{\partial f_4}{\partial\boldsymbol{x_5}} &= \frac{\left(\frac{\partial n_{\ddot{\theta}}}{\partial\beta_r}\right)d_{\ddot{\theta}}}{d_{\ddot{\theta}}^2} \end{aligned} \tag{16}$$

Finally, the partial derivatives of (5) complete the remaining f_3 row of Jacobian elements:

$$\begin{aligned} \frac{\partial f_3}{\partial\boldsymbol{x_2}} &= \frac{-\frac{\partial N_{12}}{\partial\theta}\dot{\theta} - M_{12}\frac{\partial f_4}{\partial\boldsymbol{x_2}} - \frac{\partial M_{12}}{\partial\theta}\ddot{\theta}}{M_{11}} \\ \frac{\partial f_3}{\partial\boldsymbol{x_3}} &= \frac{\frac{\partial\mu}{\partial\dot{\varphi}} - M_{12}\frac{\partial f_4}{\partial\boldsymbol{x_3}}}{M_{11}} \\ \frac{\partial f_3}{\partial\boldsymbol{x_4}} &= \frac{\frac{\partial\mu}{\partial\dot{\theta}} - \frac{\partial N_{12}}{\partial\dot{\theta}}\dot{\theta} - N_{12} - M_{12}\frac{\partial f_4}{\partial\boldsymbol{x_4}}}{M_{11}} \\ \frac{\partial f_3}{\partial\boldsymbol{x_5}} &= \frac{\frac{\partial\mu}{\partial\beta_r} - M_{12}\frac{\partial f_4}{\partial\boldsymbol{x_5}}}{M_{11}} \end{aligned} \tag{17}$$

C. Nonlinear Measurement Equations

The acceleration vector at the IMU is calculated in the body frame and used to generate accelerometer measurements. The body frame aligns with the inertial north-east-down (NED) frame when $\theta = 0$. The "north" direction is in the $+x$ direction and the "down" direction is into the ground. The nonlinear mapping from the state to the true acceleration at the IMU is:

$${}^{N}T^{B} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{18}$$

$$\begin{aligned} {}^{N}\boldsymbol{g} &= \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \\ {}^{N}\boldsymbol{a}_w &= \begin{bmatrix} \ddot{\varphi}R \\ 0 \\ 0 \end{bmatrix} \end{aligned} \tag{19}$$

$$\begin{aligned} {}^{B}\boldsymbol{g} &= \left({}^{N}T^{B}\right)^{\top}{}^{N}\boldsymbol{g} \\ {}^{B}\boldsymbol{a_w} &= \left({}^{N}T^{B}\right)^{\top}{}^{N}\boldsymbol{a_w} \end{aligned} \tag{20}$$

$${}^{B}\boldsymbol{a}_c = \begin{bmatrix} \ddot{\theta}l_s \\ 0 \\ \dot{\theta}^2 l_s \end{bmatrix}, \tag{21}$$

$${}^{B}\boldsymbol{a}_{\text{IMU}} = {}^{B}\boldsymbol{g}_{\text{body}} + {}^{B}\boldsymbol{a}_w + {}^{B}\boldsymbol{a}_c \tag{22}$$

Where:

${}^{N}T^{B}$	rotation matrix from NED to body frame
g	acceleration due to gravity (scalar)
l_s	distance from wheel center to sensors
$\boldsymbol{a}_g$	acceleration due to gravity
$\boldsymbol{a}_w$	acceleration of the center of the wheel
$\boldsymbol{a}_c$	acceleration at the IMU due to body rotation
$\boldsymbol{a}_{\text{IMU}}$	total acceleration at the IMU

The true gyroscope measurements are simply the body pitch rate state. Noise is then added to the accelerometer and gyroscope measurements before they are combined in a complimentary filter as follows:

$$\theta_{\text{accel}} = \tan^{-1}\left(\frac{\boldsymbol{a}_{\text{IMU, 1}} + \boldsymbol{w}_{\text{accel, 1}}}{\boldsymbol{a}_{\text{IMU, 2}} + \boldsymbol{w}_{\text{accel, 2}}}\right) \tag{23}$$

$$\dot{\theta}_{\text{gyro}} = \dot{\theta} + w_{\text{gyro}} \tag{24}$$

$$\begin{aligned} \theta_k &= \left(\theta_{k-1} + \dot{\theta}_{\text{gyro}} dt\right)\alpha + \theta_{\text{accel}}(1-\alpha) \\ \alpha &= \frac{T_f}{T_f + dt} \end{aligned} \tag{25}$$

Where:

θ_{accel}	body pitch angle measurement from accelerometer
$\dot{\theta}_{\text{gyro}}$	body pitch rate measurement from gyroscope
$\boldsymbol{w}_{\text{accel}}$	accelerometer noise vector
w_{gyro}	gyroscope noise
θ_k	current filtered body angle measurement
θ_{k-1}	previous filtered body angle measurement
dt	complimentary filter sample time
α	filter weight factor
T_f	tunable filter time constant

The distance sensor measurements are derived by:

$$\begin{aligned} x &= \varphi R \\ x_\theta &= l_s sin\theta \\ x_s &= x_w - x - x_\theta \end{aligned} \tag{26}$$

$$d_m = \frac{x_s}{cos\theta} = \frac{x_w - \varphi R - l_s sin\theta}{cos\theta} \tag{27}$$

Where:

x_θ	horizontal displacement of the distance sensor due to the body angle
x_w	distance from the initial position to the wall
x_s	horizontal distance from the distance sensor to the wall
d_m	distance measured by the distance sensor

Since the complimentary filter output is being treated as a direct measurement, the nonlinear measurement function is:

$$\boldsymbol{z} = \begin{bmatrix} d_m \\ \theta_m \end{bmatrix} = \begin{bmatrix} \frac{x_w - \varphi R - l_s sin\theta}{cos\theta} \\ \theta \end{bmatrix} \tag{28}$$

$$\boldsymbol{z} = h(\boldsymbol{x}) \tag{29}$$

In the sensor model, gaussian white noise is added directly to d_m.

D. Measurement Jacobian

The measurement Jacobian (H) is solved analytically using the following matrix of partial derivatives:

$$H = \begin{bmatrix} \frac{\partial h_1}{\partial \boldsymbol{x_1}} & \frac{\partial h_1}{\partial \boldsymbol{x_2}} & \frac{\partial h_1}{\partial \boldsymbol{x_3}} & \frac{\partial h_1}{\partial \boldsymbol{x_4}} & \frac{\partial h_1}{\partial \boldsymbol{x_5}} \\ \frac{\partial h_2}{\partial \boldsymbol{x_1}} & \frac{\partial h_2}{\partial \boldsymbol{x_2}} & \frac{\partial h_2}{\partial \boldsymbol{x_3}} & \frac{\partial h_2}{\partial \boldsymbol{x_4}} & \frac{\partial h_2}{\partial \boldsymbol{x_5}} \end{bmatrix}$$

Filling in the trivial partial derivatives:

$$H = \begin{bmatrix} \frac{\partial h_1}{\partial \boldsymbol{x_1}} & \frac{\partial h_1}{\partial \boldsymbol{x_2}} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{30}$$

Taking the remaining partial derivatives:

$$\begin{aligned} \frac{\partial h_1}{\partial \boldsymbol{x_1}} &= -\frac{R}{cos\theta} \\ \frac{\partial h_1}{\partial \boldsymbol{x_2}} &= \frac{-l_s cos^2\theta - (\varphi R - x_w + l_s sin\theta) sin\theta}{cos^2\theta} \end{aligned} \tag{31}$$

III. Methodology

First, a preliminary analysis of the dynamics function, measurement function, and their respective noise statistics gives some insight for the design of the Kalman filters.

A. Observability

Examining the dynamics Jacobian (F) when $\dot{\varphi} = 0$, $\frac{\partial f_3}{\partial \boldsymbol{x_5}}$ and $\frac{\partial f_4}{\partial \boldsymbol{x_5}}$ each evaluate to zero. This reduces the Jacobian to:

$$F = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \frac{\partial f_3}{\partial \boldsymbol{x_2}} & \frac{\partial f_3}{\partial \boldsymbol{x_3}} & \frac{\partial f_3}{\partial \boldsymbol{x_4}} & 0 \\ 0 & \frac{\partial f_4}{\partial \boldsymbol{x_2}} & \frac{\partial f_4}{\partial \boldsymbol{x_3}} & \frac{\partial f_4}{\partial \boldsymbol{x_4}} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{32}$$

Since the fifth column of F contains only zeros, it is guaranteed that the observability matrix (O) will not be full rank, and therefore the system is not observable if $\dot{\varphi} = 0$. This result is intuitive because the rolling damping coefficient (β_r) has no effect on the dynamics when the wheel is stationary. This also means that the observability is weaker when $\dot{\varphi}$ is close to zero. In order to properly test the performance of the β_r estimate, the initial state in the simulations includes a nonzero θ_0 so that the controller immediately applies torque to the wheels, which makes $\dot{\varphi}$ nonzero. This also means that a Kalman filter using linear dynamics must be linearized about some nonzero value of $\dot{\varphi}$.

Computing the observability matrix for the initial state:

$$\boldsymbol{x} = \begin{bmatrix} 0 & 5^\circ & 5^\circ/s & 0 & 0.001 \end{bmatrix}^\top \tag{33}$$

$$F = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & -383.4 & -7.198 & 3.8681 & -290.6 \\ 0 & 109.6 & 1.229 & -0.610 & 46.980 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{34}$$

$$H = \begin{bmatrix} -0.0331 & 0.0317 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{35}$$

$$O = \begin{bmatrix} H \\ HF \\ HF^2 \\ HF^3 \\ HF^4 \end{bmatrix} \tag{36}$$

$$rank(O) = 5 \tag{37}$$

Therefore, the system is observable when $\dot{\varphi} \neq 0$.

B. Noise Statistics

The process noise statistics are analyzed by mapping the probability density associated with the covariance matrix (P) through the nonlinear dynamics function (f) using the equation:

$$p(\boldsymbol{x}_{k+1}) = |det\left(\frac{df^{-1}(\boldsymbol{x}_{k+1})}{d\boldsymbol{x}_{k+1}}\right)|p(f^{-1}(\boldsymbol{x}_{k+1})) \tag{38}$$

The dynamics function is discretized so that it maps $\boldsymbol{x}_k$ to $\boldsymbol{x}_{x+p}$ instead of mapping $\boldsymbol{x}$ to $\frac{d\boldsymbol{x}}{dt}$. This discrete version of the mapping is denoted as f_d. This discretization is used because the discrete Jacobian ($F_d = e^{Fdt}$) is invertible. This avoids the need to find $\frac{df^{-1}(\boldsymbol{x})}{d\boldsymbol{x}}$ analytically because $\frac{df_d^{-1}(\boldsymbol{x})}{d\boldsymbol{x}}$ is given by F_d^{-1}.

The full 5-dimensional covariance matrix can be mapped through f_d, and 2-dimensional slices of the resulting distribution can be plotted to get a sense of the nonlinear effects of the dynamics function. Plotting some of these slices in figures 2 and 3, it is clear that some combinations of states behave more linearly than others, but none of them are significantly outside of the 3σ bounds of the linear transformation of the covariance matrix. This is partially due to the small size of the step size in the discretization (0.001 seconds), which matches the sample time of the estimation algorithms in the simulations. The nonlinear effects would be expected to increase if the sampling time was increased.

A similar process can be used to find the transformed noise statistics of the measurements by mapping $p(\boldsymbol{x})$ through the measurement function (h).

Figure 4 shows that the nonlinear effects of the measurement function are significant, and can be estimated more accurately using a scaled unscented transform.

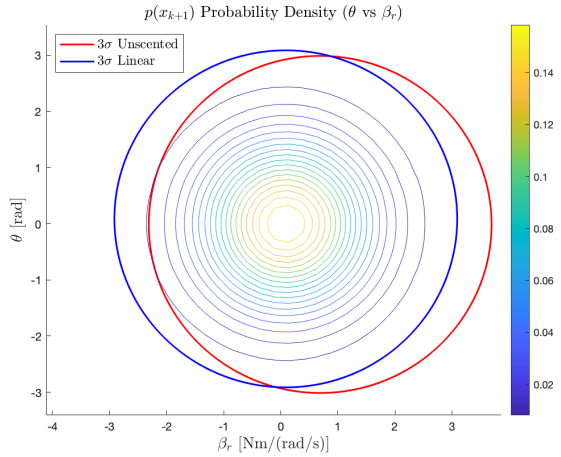

Fig. 2. θ vs β_r slice of the transformed dynamics probability distribution.

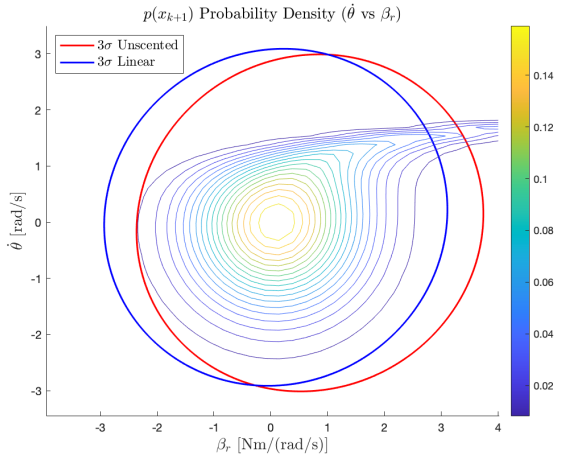

Fig. 3. $\dot{\theta}$ vs β_r slice of the transformed dynamics probability distribution.

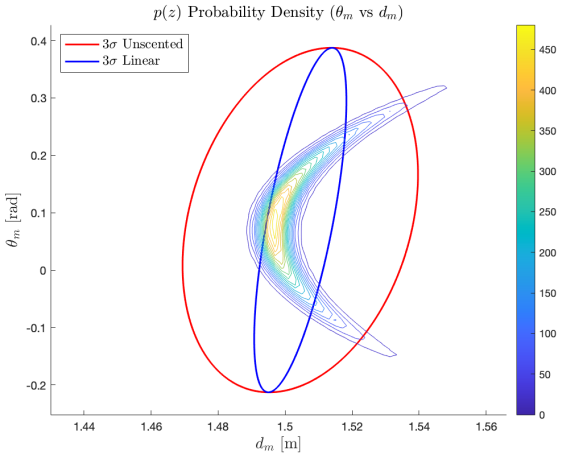

Fig. 4. θ_m vs d_m slice of the transformed measurement probability distribution.

C. Extended Kalman Filter

The extended Kalman filter is implemented in the simulation with the following steps:

Propagation Step:

$$\dot{\hat{\boldsymbol{x}}} = f(\hat{\boldsymbol{x}}) \tag{39}$$

$$\dot{P} = F(\hat{\boldsymbol{x}})P + PF(\hat{\boldsymbol{x}})^\top + GQG^\top \tag{40}$$

Gain and Update Step:

$$P_{\phi,k}^- = \phi P_k^- \phi^\top \tag{41}$$

$$W_k = H_k(\hat{\boldsymbol{x}}_k^-)P_{\phi,k}^- H_k(\hat{\boldsymbol{x}}_k^-)^\top + R \tag{42}$$

$$C_k = P_{\phi,k}^- H_k^\top(\hat{\boldsymbol{x}}_k^-) \tag{43}$$

$$K_k = C_k W_k^{-1} \tag{44}$$

$$\hat{\boldsymbol{z}}_k = h(\hat{\boldsymbol{x}}_k^-) \tag{45}$$

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k) \tag{46}$$

$$P_k^+ = (I - K_k H_k)P_{\phi,k}^-(I - K_k H_k)^\top + K_k R K_k^\top \tag{47}$$

The filter is tuned by adjusting the process noise covariance (Q), measurement noise covariance (R) and scaling matrix matrix (ϕ) until the simulated state estimate errors remain mostly within the 3σ bounds of the innovations covariance matrix (P_k^+). First, R is increased until the φ, θ, $\dot{\varphi}$, and $\dot{\theta}$ state estimate errors remain within the 3σ bounds. The β_r state is more difficult to tune because the 3σ bounds collapse toward zero very quickly. Increasing Q corresponding to β_r somewhat improves this behavior, but increasing it too much causes the estimate error to increase dramatically. Therefore, a final step is taken to apply a scaling factor (ϕ) to the covariance matrix (P_k^-) in the update step. This forces the covariance term corresponding to the 3σ bounds for β_r to collapse more slowly, allowing the β_r estimate error to remain within the bounds consistently. The final tuned values are:

$$R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \tag{48}$$

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix} \tag{49}$$

$$\phi = diag(\begin{bmatrix} 0 & 0 & 0 & 0 & 1.9 \end{bmatrix}) \tag{50}$$

This significant tuning was required due to two key violations of the assumptions in the filter. Firstly, the filter's derivation assumes that the measurement noise is zero-mean. This is significantly violated in the case of the body angle measurement (θ_m) from the IMU and complimentary filter. The error in this measurement has a bias that changes slowly throughout the simulation, as shown in figure 5. This partially explains the need to add more noise to the filter so that the estimate errors still fall within the 3σ bounds.

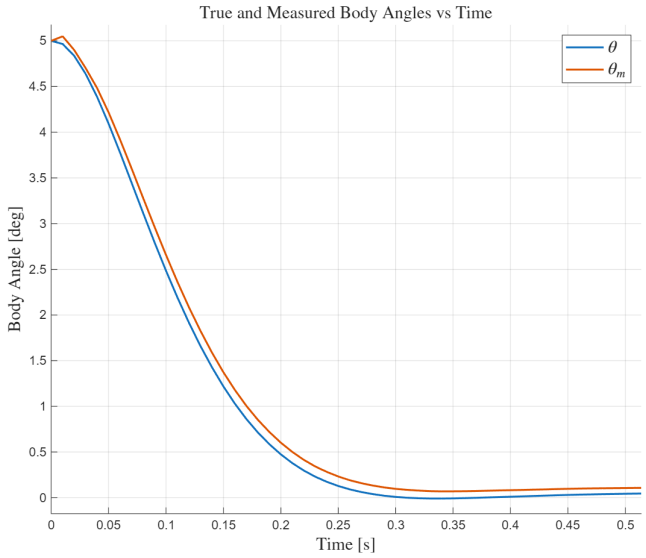

Fig. 5. θ and θ_m vs time shows the bias in the body angle measurement.

Another significantly violated assumption is that the filter assumes linear mappings of the covariance matrix through both the dynamics and measurement functions. In figure 4, it is clear that the linear estimate of the covariance matrix does not accurately capture the noise mapping through the measurement function. This violation can be addressed by the unscented Kalman filter.

D. Unscented Kalman Filter

Motivated by the significant amount of tuning required for the extended Kalman filter to produce consistent estimates, the unscented Kalman filter is implemented as follows:

$$\begin{aligned} n &= 5 \\ \alpha &= 0.2 \\ \beta &= 2 \\ \kappa &= 0 \\ \lambda &= \alpha^2(n+\kappa) - n \end{aligned} \tag{51}$$

Propagation Step:

$$\begin{aligned} \chi_{k-1}^{(0)} &= \hat{\boldsymbol{x}}_{k-1}^+ \\ \chi_{k-1}^{(i)} &= \hat{\boldsymbol{x}}_{k-1}^+ + \sqrt{n+\lambda}[\sqrt{P_{k-1}^+}]_i \quad i = 1, ..., n \\ \chi_{k-1}^{(i+n)} &= \hat{\boldsymbol{x}}_{k-1}^+ - \sqrt{n+\lambda}[\sqrt{P_{k-1}^+}]_i \quad i = 1, ..., n \end{aligned} \tag{52}$$

$$
\begin{aligned}
w_0^{(m)} &= \frac{\lambda}{n+\lambda} \\
w_0^{(c)} &= \frac{\lambda}{n+\lambda} + (1-\alpha^2+\beta) \\
w_i^{(m)} &= \frac{1}{2(n+\lambda)} \qquad i = 1, ..., 2n \\
w_i^{(c)} &= \frac{1}{2(n+\lambda)} \qquad i = 1, ..., 2n
\end{aligned} \tag{53}
$$

$$
\chi_k^{(i)} = f(\chi_{k-1}^{(i)}), i = 0, ..., 2n \tag{54}
$$

$$
\begin{aligned}
\hat{\boldsymbol{x}}_k^- &= \Sigma_{i=0}^{2n} w_i^{(i)} \chi_k^{(i)} \\
P_k^- &= \Sigma_{i=0}^{2n} w_i^{(c)} \left(\chi_k^{(i)} - \hat{\boldsymbol{x}}_k^-\right)\left(\chi_k^{(i)} - \hat{\boldsymbol{x}}_k^-\right)^\top + Q
\end{aligned} \tag{55}
$$

Gain and Update Step:

$$
\begin{aligned}
\chi_{k-1}^{-(0)} &= \hat{\boldsymbol{x}}_{k-1}^- \\
\chi_{k-1}^{-(i)} &= \hat{\boldsymbol{x}}_{k-1}^- + \sqrt{n+\lambda}[\sqrt{P_{k-1}^-}]_i \quad i = 1, ..., n \\
\chi_{k-1}^{-(i+n)} &= \hat{\boldsymbol{x}}_{k-1}^- - \sqrt{n+\lambda}[\sqrt{P_{k-1}^-}]_i \quad i = 1, ..., n
\end{aligned} \tag{56}
$$

$$
\begin{aligned}
w_0^{(m)} &= \frac{\lambda}{n+\lambda} \\
w_0^{(c)} &= \frac{\lambda}{n+\lambda} + (1-\alpha^2+\beta) \\
w_i^{(m)} &= \frac{1}{2(n+\lambda)} \qquad i = 1, ..., 2n \\
w_i^{(c)} &= \frac{1}{2(n+\lambda)} \qquad i = 1, ..., 2n
\end{aligned} \tag{57}
$$

$$
Z_k^{(i)} = h(\chi_{k-1}^{-(i)}), i = 0, ..., 2n \tag{58}
$$

$$
\begin{aligned}
\hat{\boldsymbol{z}}_k^- &= \Sigma_{i=0}^{2n} w_i^{(m)} Z^{(i)} \\
P_{z,k} &= \Sigma_{i=0}^{2n} w_i^{(c)} \left(Z_k^{(i)} - \hat{\boldsymbol{z}}_k\right)\left(Z_k^{(i)} - \hat{\boldsymbol{z}}_k\right)^\top + R \\
P_{xz,k} &= \Sigma_{i=0}^{2n} w_i^{(c)} \left(\chi_k^{-(i)} - \hat{\boldsymbol{x}}_k^-\right)\left(Z_k^{(i)} - \hat{\boldsymbol{z}}_k\right)^\top
\end{aligned} \tag{59}
$$

$$
\begin{aligned}
K_k &= P_{xz,k} P_{z,k}^{-1} \\
\hat{\boldsymbol{x}}_k^+ &= \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k) \\
P_k^+ &= P_k^- - P_{xz,k} K_k^\top - K_k P_{xz,k}^\top + K_k P_{z,k} K_k^\top
\end{aligned} \tag{60}
$$

This unscented Kalman filter is tuned using a similar method as for the extended Kalman filter. First, R is increased until the φ, θ, $\dot{\varphi}$, and $\dot{\theta}$ estimate errors consistently remain within the 3σ bounds, then Q is increased until the β_r estimate error also remains within the 3σ bounds.

$$
R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.01 \end{bmatrix} \tag{61}
$$

$$
Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \times 10^{-7} \end{bmatrix} \tag{62}
$$

The unscented Kalman filter has the advantage of not needing a scaling factor (ϕ), and the overall performance is generally better. The tuned measurement and process covariance matrices are generally smaller for the UKF, with the exception of the process noise terms Q_{11} and Q_{22} which correspond to $\dot{\varphi}$ and $\dot{\theta}$. Although the unscented transform avoids the linear approximations for mapping the covariance matrix through f and h, this filter still assumes that the measurement noise is zero-mean, which is being violated by θ_m.

E. Simulation Setup

The physical parameters for the robot are rough estimates based on the actual robot.

m	0.5 kg
m_w	0.05 kg
R	0.033 m
L	0.15 m
I	0.004 kg*m^2
I_w	1×10^{-4} kg*m^2
l_s	0.1 m
β_f	0.001 N*m/(rad/s)

The initial state vector is held constant between each simulation so that the results can be easily compared. It includes an initial 5° offset in θ so that the PID controller begins commanding torque to the motors at the initial time step. This causes the wheels to immediately start spinning so that $\dot{\varphi} \neq 0$ and β_r is observable. As the simulation continues, the PID controller maintains balance by regulating θ toward zero. There is no controller regulating the wheel angle (φ), so the wheels will continue to move until they are eventually stopped by the rolling damping. However, this happens slowly enough that $\dot{\varphi}$ remains nonzero for the entire simulation. The initial rolling damping coefficient (0.001 N*m/(rad/s)) is chosen to roughly approximate the rolling resistance on a hard floor.

$$
\boldsymbol{x}_0 = \begin{bmatrix} 0 & 5^\circ & 0 & 0 & 0.001 \end{bmatrix}^\top
$$

First, a set of Monte Carlo simulations are conducted for each filter to test the consistency of the state estimate errors remaining within the 3σ bounds. The only difference between each trial is the random seed for generating the sensor noise. After that, the same Monte Carlo analysis is performed with a discrete change in the true rolling damping coefficient (β_r) at $t = 5$ seconds. This simulates the robot driving from a hard floor onto a rug with significantly more rolling resistance, and is used to compare how well each filter converges after the change in β_r. Next, this analysis is repeated with varying amounts of model mismatch between the plant and the filter propagation steps. Mismatches in the physical

parameters of the robot are generated by sampling random changes in the physical parameters listed above, and repeating this process with varying standard deviations for the random changes. This provides some insight into how accurately the physical parameters need to be characterized before the filter can be successfully implemented in the real world. Finally, a set of simulations without model mismatch is conducted for a sweep of distance sensor noise levels to determine the acceptable noise standard deviation for a distance sensor in this application.

IV. Results

A. Consistency Testing

The state estimate errors are plotted with the 3σ bounds of the innovations covariance matrix to confirm that the errors generally remain within these bounds. These plots are generated by repeating the same simulation with a different random seed for the sensor noise generation. In figure 6, the estimate errors from the extended Kalman filter remain well within the 3σ bounds for every state except for the rolling damping coefficient (β_r). The error in the β_r estimate remains generally within the bounds, with a few exceptions. Most of these exceptions occur when the bias of the θ estimate is the largest, around 3-4 seconds into the simulation. The φ, θ, $\dot{\varphi}$, and $\dot{\theta}$ state errors are much further from the 3σ bounds due to the increases in R during the tuning process. The θ error plot in figure 6 shows some significant bias due to the bias in θ_m shown in figure 5. The additional measurement noise added to R ensures that the θ error consistently remains within the 3σ bounds, even with this bias.

The consistency test plots follow similar trends for the unscented Kalman filter in figure 7. The φ, θ, $\dot{\varphi}$, and $\dot{\theta}$ state errors remain relatively far from the 3σ bounds while the β_r error is much closer. Compared to the extended Kalman filter, the unscented Kalman filter has generally less error in each of the five state estimates due to the more accurate transformations of the covariance matrix in the propagation and update steps. It is also clear from the θ error plot in figure 7 that there is still a significant bias in the θ estimate due to the bias in θ_m. This again requires the R matrix to be tuned so that the 3σ bounds remain wide enough to contain the θ error. Overall, figures 6 and 7 provide some confidence that both filters will not diverge. Future work should include running these consistency tests for longer and with more trials, along with a sweep of different initial conditions to further ensure reliability.

B. Step Change in Rolling Damping Coefficient

A step change in the plant's rolling damping coefficient is introduced to test how each filter responds to dynamic changes in rolling resistance. This models the behavior of the robot driving from a hard surface onto a rug. The initial β_r is still 0.001 N*m/(rad/s), but a discrete jump to 0.002 N*m/(rad/s) is added at precisely $t = 5$ seconds. These values are very rough approximations and are meant to test how small of a change in β_r the Kalman filters can reliably detect. If the

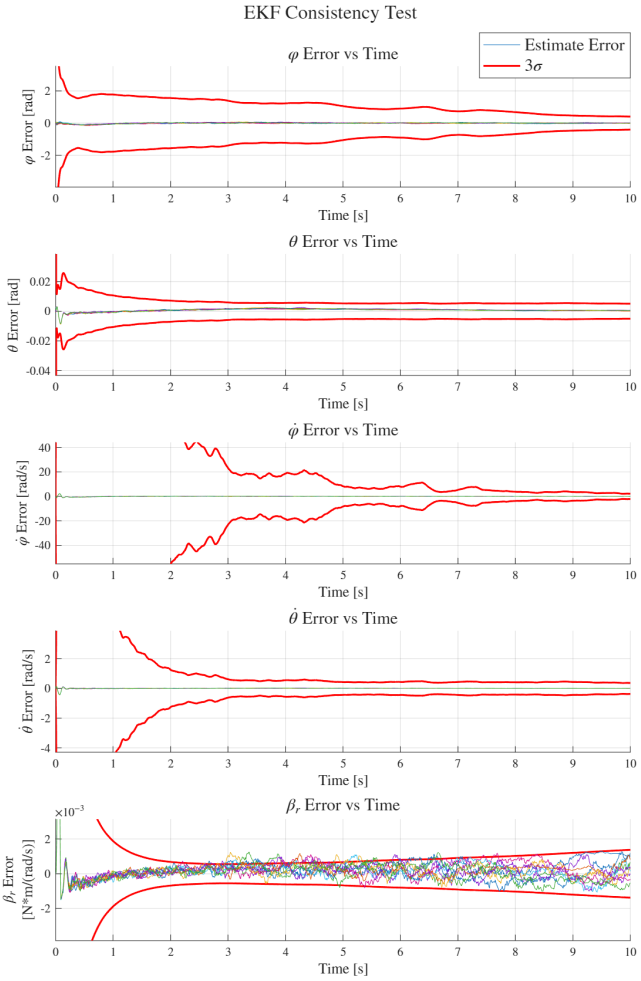

Fig. 6. Consistency test for the extended Kalman filter to make sure that the state estimate errors remain mostly within the 3σ bounds of the innovations covariance.

β_r estimate is to be used to schedule controller gains for the robot, the minimum detectable difference in β_r helps inform how precisely these gains can be scheduled.

In figure 8, the β_r estimate responses show a settling time of roughly one second for most of the trials. However, the noise in this estimate continues to increase over time, so by 10 seconds it would be difficult to discern between 0.001 and 0.002 N*m/(rad/s). We see a similar increase in noise in figure 6, so the increase is not solely due to the jump in β_r. One possible explanation for this is that as the robot regains its balance from the initial 5° offset in θ, the decrease in $\dot{\varphi}$ makes the β_r observability weaker. It was shown in (32) that β_r is no longer observable when $\dot{\varphi} = 0$.

For the unscented Kalman filter in figure 9, there is much less noise in the β_r estimate. When the estimate reaches a steady state, it is clearly discernible between 0.001 and 0.002 N*m/(rad/s) in nearly all trials. However, the unscented Kalman filter's estimate takes about 4 seconds to settle as opposed to the extended Kalman filter's 1 second. It is possible that this response time could be improved with more tuning

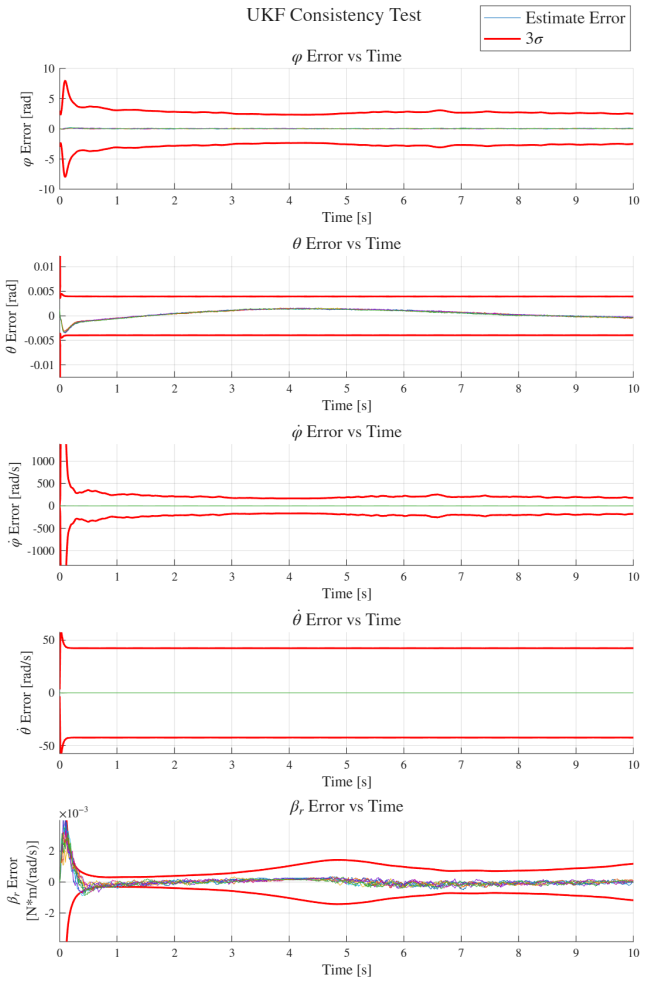

Fig. 7. Consistency test for the unscented Kalman filter to make sure that the state estimate errors remain mostly within the 3σ bounds of the innovations covariance.

of R and Q, but this tuning process is time-consuming and tends to involve a lot of trial and error. Overall, the unscented Kalman filter would still be preferred by these metrics because although it reaches the steady state more slowly, it can reliably discern between the two values of β_r.

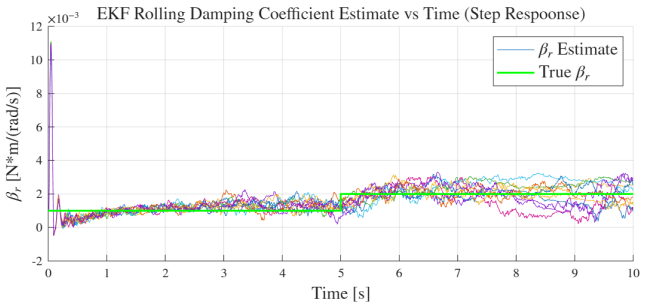

Fig. 8. Extended Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds.

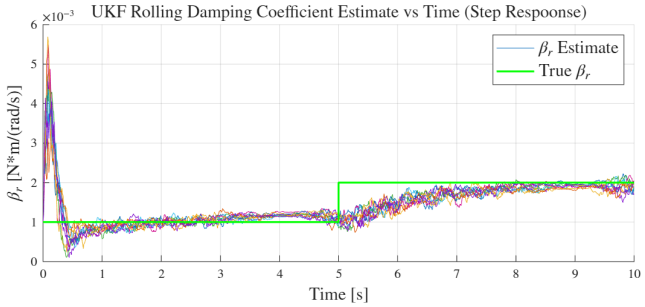

Fig. 9. Unscented Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds.

C. Model Mismatch

Model mismatch as added between the plant model and the propagation step models of the two Kalman filters. This mismatch is added in the form of variations in the physical parameters defined in III-E. Some of these parameters, such as the inertia and motor friction, are not trivial to determine precisely for the real system. Therefore, it is useful to know how accurately they need to be characterized for the purposes of accurate Kalman filter estimates. The mismatch is added by offsetting each parameter from its true value by a random amount for each trial. This random amount is chosen by a uniform distribution between $+X\%$ and $-X\%$ of the true value. For example, if the true mass was 100 kg and $X = 15\%$, the mismatch offset would be chosen from a uniform distribution between -15 and +15 kg for each trial. Simulations are conducted for $X = 5\%, 15\%, 25\%$, and 50% to cover a range of model mismatches.

As expected, the error in the β_r estimate from both Kalman filters increases as more model mismatch is introduced in figures 10 and 11. The estimates corresponding to 5% and 15% model mismatch remain reasonably accurate, and many of the trials still accurately distinguish between 0.001 and 0.002 N*m/(rad/s). As the model mismatch continues to increase towards 25% and 50%, the estimates begin to degrade significantly. If these physical parameters of the robot can be characterized within an accuracy of at least 5-15%, that should be enough to produce reasonable β_r estimates. If any particular parameter proves to be more difficult to estimate within that accuracy, further simulations may be used to isolate the impact of model mismatch in that particular parameter.

D. Distance Sensor Noise Sweep

Finally, a sweep of different noise standard deviations for the distance sensor is conducted to determine the required accuracy of this sensor for useful β_r estimates. The distance sensor noise is modeled as additive gaussian white noise with a standard deviation between 1 mm and 2 cm. In reality, for an ultrasonic distance sensor, the noise is also a function of the distance from the sensor to the target it is measuring, but it is reasonably constant for the nominal distance of $x_w = 1.5$ m used in these simulations.

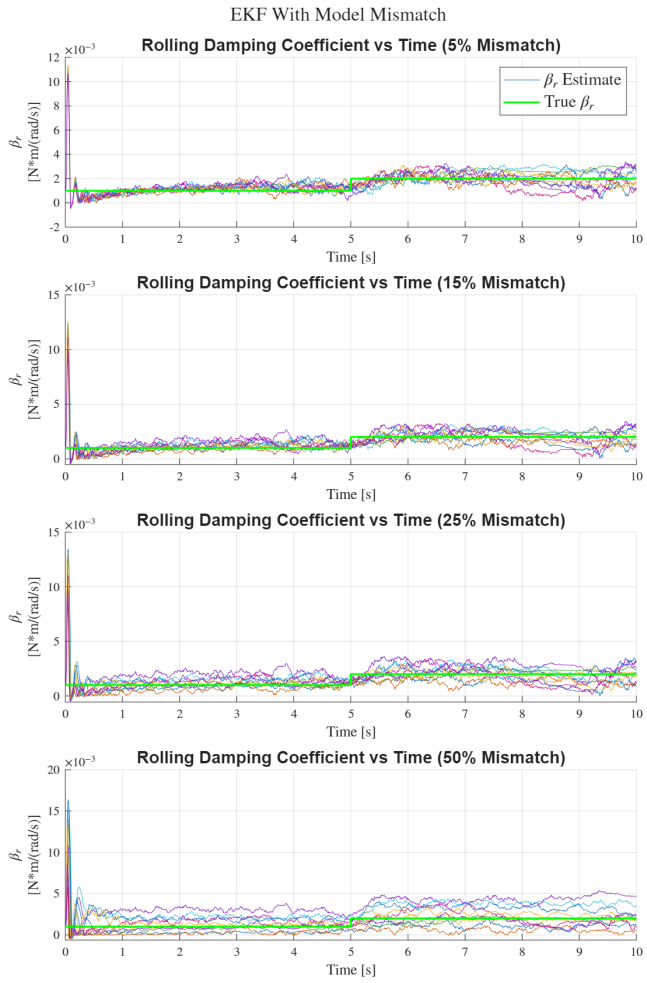

Fig. 10. Extended Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds with varying model mismatch.

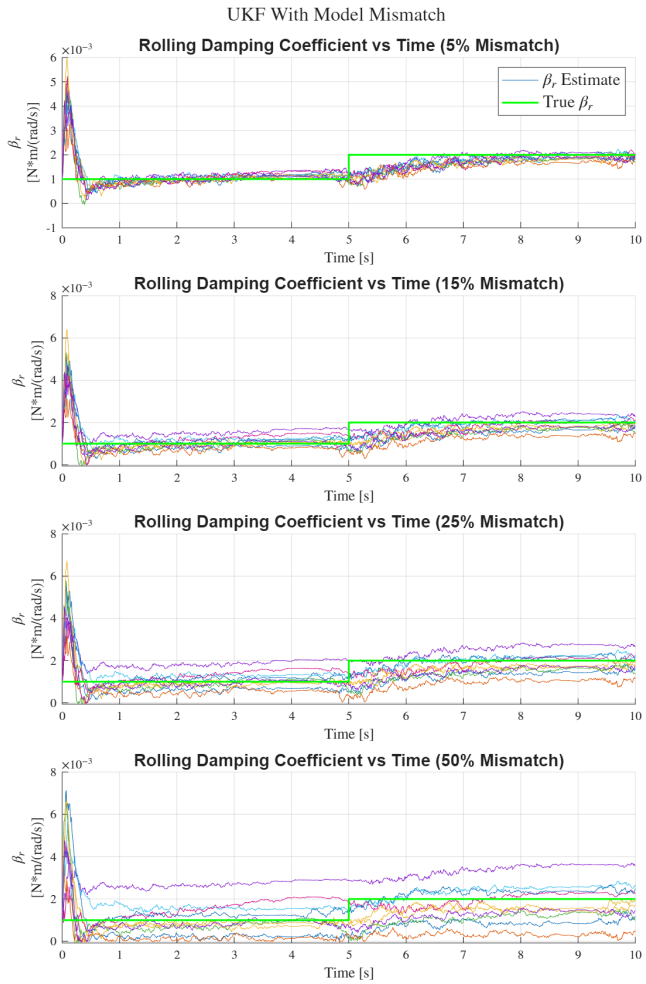

Fig. 11. Unscented Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds with varying model mismatch.

Intuitively, the error in the β_r estimate increases as the noise in the distance measurement increases, as shown in figures 12 and 13. For the lowest sensor noise standard deviation of 1 mm, the β_r estimates improve significantly compared to the 1 cm standard deviation used in all the previous simulations. Both Kalman filters can consistently distinguish between β_r values of 0.001 and 0.002 N*m/(rad/s) with a 1 mm standard deviation. As the standard deviation increases, the performance deteriorates much faster for the extended Kalman filter than for the unscented Kalman filter. This may be because of the significant nonlinear effects in the measurement mapping function shown in figure 4. As the noise increases, the linear transformation of the covariance matrix becomes a worse approximation of the true probability density. This means that if more noise is present, the advantage of the unscented Kalman filter is more pronounced. These results show that a high-quality distance sensor may be worth the extra cost, especially if it is being used with an extended Kalman filter rather than an unscented Kalman filter.

V. Conclusions

This paper demonstrates that estimating the rolling resistance for a balancing robot is feasible with low-cost sensors and imperfect modeling. The main challenge comes from tuning the Kalman filters, especially when the assumptions used to derive the filters are significantly violated. In the case of the extended Kalman filter, the assumption of linear covariance transformations is significantly violated when the covariance is updated using the linearized measurement Jacobian H to approximate the nonlinear function h, but this violation is addressed in the unscented Kalman filter, which uses scaled unscented transforms instead of linear transformations for mapping the covariance through the dynamics and measurement functions. This substantially reduced the error in the rolling damping coefficient estimates, but both filters still suffer from the significant bias in the body angle measurement (θ_m). The body angle measurement comes from a complimentary filter which works by applying a low-pass filter to the angles calculated from the accelerometer measurements, and applying a high-pass filter to the angles

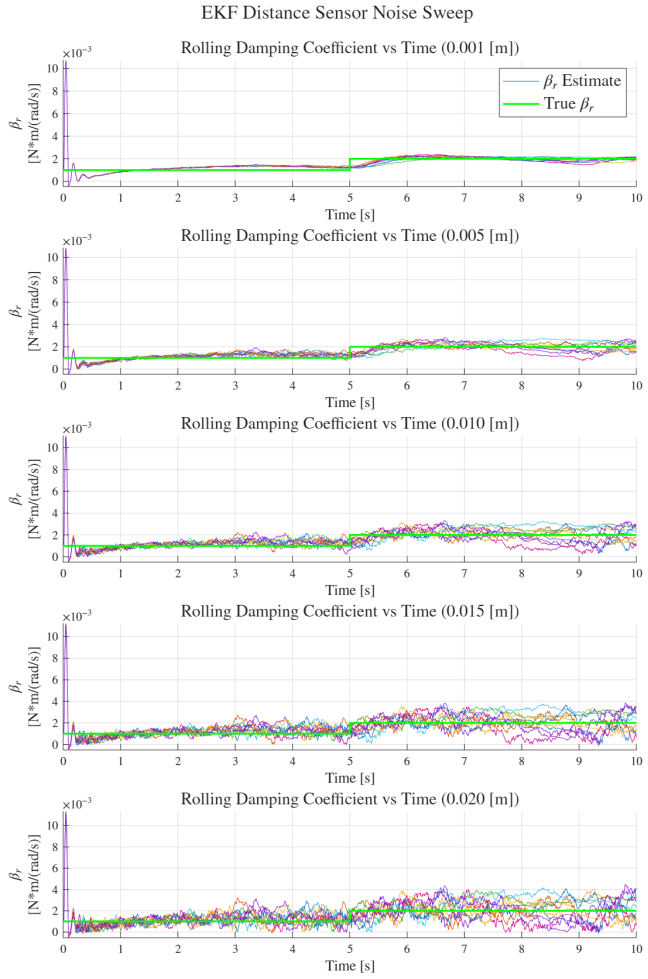

Fig. 12. Extended Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds with varying distance sensor noise standard deviations.

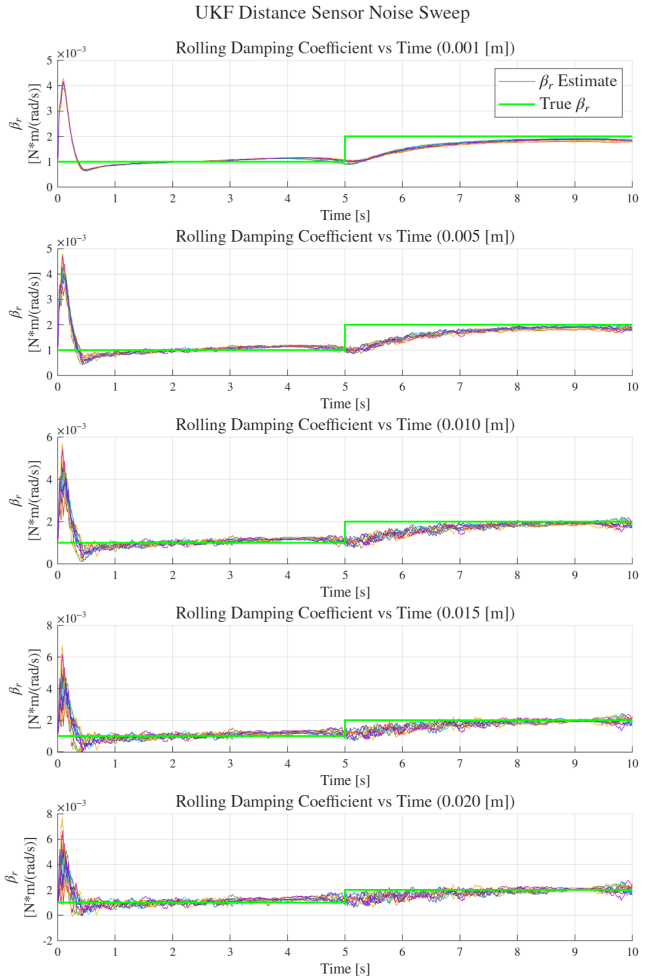

Fig. 13. Unscented Kalman filter rolling damping coefficient estimate (β_r) vs time with a step change at $t = 5$ seconds with varying distance sensor noise standard deviations.

calculated from the gyroscope measurements. This method is accurate enough to balance the robot without any additional filtering, but using this output as a measurement in a Kalman filter is problematic because the error is significantly biased and the Kalman filters assume unbiased noise. Therefore, a better inertial measurement of the body angle could potentially improve the performance of these Kalman filters substantially. A better estimate could decrease the error bias and make the angle estimate behave more like a zero-mean measurement. Alternatively, the complimentary filter approach could be used with an additional measurement bias state in the Kalman filters. This approach would estimate the measurement bias at each step so that it could be subtracted from the the body angle measurement, and this new measurement would have noise closer to zero-mean. The first method would be preferable as a starting point, and the second method could potentially be added if better performance is still required.

Another consideration for future work is the computational cost of these algorithms. The scaled unscented transforms in the unscented Kalman filter require significantly more computation than the linear transformations in the extended Kalman filter. This is especially true in the propagation step because the dynamics need to be integrated for 11 different sigma points for the unscented transform. One way to reduce this computational cost would be to assume linear dynamics for the propagation step. Instead of numerically integrating the sigma points for each propagation step, the Jacobian can be discretized at the initial sigma points and used to propagate them with a linear transformation rather than numerical integration. Another way to reduce the computational cost would be to run the estimator at a lower rate. In these simulations, the estimator runs at 1000 hz. This relatively high rate was chosen so that this paper could focus on other aspects of the filters without taking the frequency into account. Future work could look at the two filters' performance over a sweep of sample rates to help determine the minimum sample rate required for accurate state estimates.

REFERENCES

[1] Y. Ding, J. Gafford, and M. Kunio, "Modeling, Simulation and Fabrication of a Balancing Robot," Harvard University.