Low-cost Lunar Lander Design Model

Emily Pitts
Department of ECE
Utah State Univerity
Logan, UT 84322
emily.pitts@usu.edu

I. Coordinate Systems

In determining the details of our project, we need to keep careful track of which coordinate system they are expressed in. There are four coordinate systems we need to be concerned with: the body frame fixed to the lander, the inertial frame whose origin is fixed at the center of the moon, the moon frame whose origin is at the center of the moon that rotates with the moon, and the camera frame that represents the rotation of the camera with respect to the lander. With the guidance of Dr. Christensen, we determined the appropriate frames and their orientations. The frames are shown below.

[TODO: FIX THIS IMAGE]

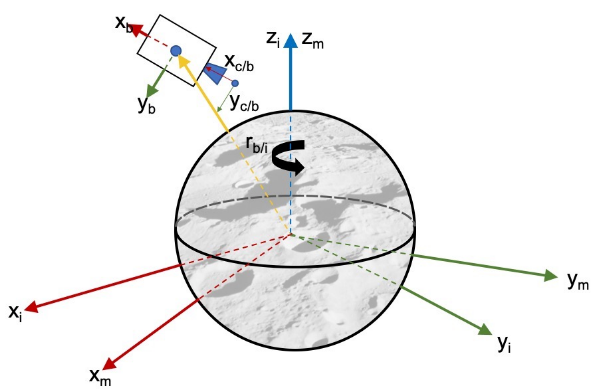

Fig. 1: The coordinate systems used in our problem

In figure 1, please note that the inertial frame and the moon frame have their origins at the center of the moon, and that the r vector is not a coordinate system, but an expression of the position of the lander in relation to the moon. In fact, r has 3 dimensions, but it is difficult to show in this image.

II. State Vectors

We now need to determine what variables we have, which parts of the problem can be simplified, and the relationship between the truth state, the design state, the navigation state, and the error inherent in any system such as this.

The truth state describes the dynamics of the lunar lander. It needs to contain all information we might wish to consider in our navigation. In this project, we need to incorporate not only the dynamics of the lander itself, but also the altimeter and camera we will be using, as these will be able to rotate with respect to the lander. With the guidance of Dr. Christensen, we determined the state variables we need.

Note that there are many other possible state variables, such as angular rate or a set of states for a gyro. We have carefully selected our variables in order to fully support the problem but also to solve a simplified problem.

The truth state vector given by 1

$$\boldsymbol{x_t} = \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ q^m_i \\ q^c_b \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} = \begin{bmatrix} x_v \\ \boldsymbol{p} \end{bmatrix} \tag{1}$$

where

$$x_v = \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ \boldsymbol{q}^m_i \end{bmatrix} \tag{2}$$

and

$$\boldsymbol{p} = \begin{bmatrix} \boldsymbol{q}^c_b \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} \tag{3}$$

$\boldsymbol{r}^i_{b/i}$ is the position of the vehicle in the body frame with respect to the inertial frame as expressed in the inertial frame, and it contains an x-, y-, and z-component. $\boldsymbol{v}^i_{b/i}$ is the velocity of the vehicle in the body frame with respect to the inertial frame as expressed in the inertial frame, and it contains an x-, y-, and z-component. q^m_i is the quaternion representing the attitude orientation of the lander in the inertial frame with respect to the moon. q^c_b is the quaternion representing the orientation of the body frame with respect to the camera. b_r is the bias in the clock seen as the bias in the measured range, $\boldsymbol{\epsilon}^i_g$ is the bias in the gravity of the moon, h_t is the height of terrain as seen from the bottom of the lander, and $\boldsymbol{b}_a$ is the bias in the accelerometer in the body frame.

The design state vector is given by 4

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} \tag{4}$$

Note that the design state vector does not contain either quaternion. This is because the design state vector contains everything we will feed into the Kalman filter, and there are some states we cannot improve upon by means of estimation. We removed the attitude quaternion, q_i^m, and the camera quaternion, q_b^c, because we can determine these better by means other than our Kalman filter.

The navigation state vector is given by 5. Note that the elements are the same, except they are notated by a hat. We will inject errors into the navigation state using calculated errors.

$$\hat{\boldsymbol{x}} = \begin{bmatrix} \hat{\boldsymbol{r}}^i_{b/i} \\ \hat{\boldsymbol{v}}^i_{b/i} \\ \hat{b}_r \\ \hat{\boldsymbol{\epsilon}}^i_g \\ \hat{h}_t \\ \hat{\boldsymbol{b}}^b_a \end{bmatrix} \tag{5}$$

The error vector is given by 6. This state vector will be used to store calculated errors

$$\boldsymbol{\delta x} = \begin{bmatrix} \delta\boldsymbol{r}^i_{b/i} \\ \delta\boldsymbol{v}^i_{b/i} \\ \delta b_r \\ \delta\boldsymbol{\epsilon}^i_g \\ \delta h_t \\ \delta\boldsymbol{b}^b_a \end{bmatrix} \tag{6}$$

III. Relationships

We need to derive a mapping between the truth, design, navigation, and error state vectors. Following the procedure outlined in section 3.1 of the debugging guide, we have identified the relationship between various state vectors. We say that the correct state estimates, or the state after accounting for error, is given as

$$\begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{r}}^i_{b/i} \\ \hat{\boldsymbol{v}}^i_{b/i} \\ \hat{b}_r \\ \hat{\boldsymbol{\epsilon}}^i_g \\ \hat{h}_t \\ \hat{\boldsymbol{b}}^b_a \end{bmatrix} + \begin{bmatrix} \delta\boldsymbol{r}^i_{b/i} \\ \delta\boldsymbol{v}^i_{b/i} \\ \delta b_r \\ \delta\boldsymbol{\epsilon}^i_g \\ \delta h_t \\ \delta\boldsymbol{b}^b_a \end{bmatrix} = \boldsymbol{c}(\hat{\boldsymbol{x}}, \delta\boldsymbol{x}) \tag{7}$$

The equation used to insert errors into the navigation states is a rearrangement of equation 7

$$\begin{bmatrix} \hat{\boldsymbol{r}}^i_{b/i} \\ \hat{\boldsymbol{v}}^i_{b/i} \\ \hat{b}_r \\ \hat{\boldsymbol{\epsilon}}^i_g \\ \hat{h}_t \\ \hat{\boldsymbol{b}}^b_a \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} - \begin{bmatrix} \delta\boldsymbol{r}^i_{b/i} \\ \delta\boldsymbol{v}^i_{b/i} \\ \delta b_r \\ \delta\boldsymbol{\epsilon}^i_g \\ \delta h_t \\ \delta\boldsymbol{b}^b_a \end{bmatrix} = \boldsymbol{i}(\boldsymbol{x}, \delta\boldsymbol{x}) \tag{8}$$

Note that the navigation states are marked as estimations, by use of a hat Equation 9 computes the error in the estimation, which is used to create δx.

$$\begin{bmatrix} \delta\boldsymbol{r}^i_{b/i} \\ \delta\boldsymbol{v}^i_{b/i} \\ \delta b_r \\ \delta\boldsymbol{\epsilon}^i_g \\ \delta h_t \\ \delta\boldsymbol{b}^b_a \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} - \begin{bmatrix} \hat{\boldsymbol{r}}^i_{b/i} \\ \hat{\boldsymbol{v}}^i_{b/i} \\ \hat{b}_r \\ \hat{\boldsymbol{\epsilon}}^i_g \\ \hat{h}_t \\ \hat{\boldsymbol{b}}^b_a \end{bmatrix} = \boldsymbol{e}(\hat{\boldsymbol{x}}, \boldsymbol{x}) \tag{9}$$

As we previously noted, the design states are just a subset of the truth states, carefully selected to simplify our problem. The relationship between the truth state and the design state vectors, then, is merely a matrix multiplication that picks off the states we deemed appropriate for the design state. Equation 10 shows this relationship.

$$\boldsymbol{x} = \begin{bmatrix} I_{6\times6} & \boldsymbol{0}_{6\times8} & \boldsymbol{0}_{6\times8} \\ \boldsymbol{0}_{8\times6} & \boldsymbol{0}_{8\times8} & \boldsymbol{0}_{8\times8} \\ \boldsymbol{0}_{8\times6} & \boldsymbol{0}_{8\times8} & I_{8\times8} \end{bmatrix} \begin{bmatrix} \boldsymbol{r}^i_{b/i} \\ \boldsymbol{v}^i_{b/i} \\ \boldsymbol{q}^m_i \\ \boldsymbol{q}^c_b \\ b_r \\ \boldsymbol{\epsilon}^i_g \\ h_t \\ \boldsymbol{b}^b_a \end{bmatrix} = \boldsymbol{m}(\boldsymbol{x_t}) \tag{10}$$

IV. Verification of States

Now we need to verify that our state relationships are valid. Using the provided code, we successfully passed all assertions. We provide the values used below for reference. Gray cells indicate that the state is not in the measured vector and green cells indicated a match in estimated error and the error provided to the code by us.

Verification Values				
Variable	Truth	Inject Error	Estimate Error	Correct Error
r_x	1.5755E+06	1.5755E+06	100	1.5755E+06
r_y	-1.7875E+05	-1.7895E+05	200	-1.7875E+05
r_z	-9.2832E+05	-9.2862E+05	300	-9.2832E+05
v_x	-5.0450E+02	-5.0550E+02	1	-5.0450E+02
v_y	-1.4165E+03	-1.4185E+03	2	-1.4165E+03
v_z	-5.8329E+02	-5.8629E+02	3	-5.8329E+02
q_a^m	1			
q_i^m	0			
q_j^m	0			
q_k^m	0			
q_a^c	1			
q_i^c	0			
q_j^c	0			
q_k^c	0			
b_r	0	-1.0000E-03	1.0000E-03	0
ϵ_x^g	0	-2.0000E-03	-2.0000E-03	0
ϵ_y^g	0	-3.0000E-03	-3.0000E-03	0
ϵ_z^g	0	-8.7266E-04	-8.7266E-04	0
h_t	0	-7.2722E-04	-7.2722E-04	0
b_x^a	0	-4.8481E-08	-4.8481E-0	0
b_y^a	0	-9.6962E-08	-9.6962E-08	0
b_z^a	0	-1.4544E-07	-1.4544E-07	0

V. Nonlinear State Propagation And Measurement Modeling

We now have to define differential equations for the truth state in order to propagate the states through time. The truth and navigation models we will use in our EKF and in our simulations have both nonlinear dynamics and measurements. The truth state dynamics are a function of time, inputs, and the truth state vector determined in section II. We assume process noise and measurement noise are zero-mean white noise. The navigation state dynamics are a function of time, measurement inputs, and the navigation vector we determined in section II.

A. Truth State Dynamics

The truth state dynamics are given by

$$\dot{\boldsymbol{r}}^i_{b/i} = \boldsymbol{v}^i_{b/i} \tag{11}$$

$$\dot{\boldsymbol{v}}^i_{b/i} = \boldsymbol{a}^i_{grav} + \boldsymbol{a}^i_{thr} + \boldsymbol{\epsilon}^i_g + w^i_a, \tag{12}$$

$$\boldsymbol{a}^i_{grav}(\boldsymbol{r}^i_{b/i}) = -\frac{\mu}{\|\boldsymbol{r}^i_{b/i}\|^3}\boldsymbol{r}^i_{b/i} \tag{13}$$

where Equation 13 is the acceleration due to the gravity of the moon, and $\boldsymbol{a}^i_{thr}$ is the acceleration due to thrust (to be calculated using the guidance law in **??** and requiring a desired final position, velocity, and acceleration). $\boldsymbol{\epsilon}^i_g$ is anomalous acceleration, and w^i_a is the process noise we include in order to account for other accelerations. Next, we define the states relating to rotation, the quaternions. We assume that the camera is fixed so in the camera quaternion, the rate of change is zero.

$$\dot{\boldsymbol{q}}^m_i = \frac{1}{2}\omega^m_{m/i} \otimes \boldsymbol{q}^m_i \tag{14}$$

$$\omega^m_{m/i} = \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2.7 \times 10^{-6} \end{bmatrix} \tag{15}$$

$$\dot{q}^c_b = 0 \tag{16}$$

We model the biases are exponentially correlated random variables (ECRVs). We use a process noise, w, and a time constant, τ, in order to propagate these states. Note that $\tau \approx \frac{T}{2}$

$$\dot{b}_r = -\frac{1}{\tau_r}b_r + w_r, \tag{17}$$

$$\dot{b}_{ax} = -\frac{1}{\tau_{ax}}b_{ax} + w_{ax} \tag{18}$$

Last, we handle the gravity error and terrain height states. Note that there is a velocity term, the lateral velocity of the lander, used to force the gravity error to have the correct units. We also have process noise terms for both the gravity error and terrain height states.

$$\dot{\boldsymbol{\epsilon}}_g = -\frac{\|\boldsymbol{v}_\perp\|}{d_g}\boldsymbol{\epsilon}_g + w_d \tag{19}$$

$$\boldsymbol{v}_\perp = \boldsymbol{v}^i_{b/i} - \boldsymbol{\omega}^i_{m/i} \times \boldsymbol{r}^i_{o/i} - (\boldsymbol{v}^i_{b/i} \cdot \hat{i}_r)\hat{i}_r \tag{20}$$

$$\hat{i}_r = \frac{\boldsymbol{r}^i_{b/i}}{\|\boldsymbol{r}^i_{b/i}\|} \tag{21}$$

$$\dot{h} = -\frac{\|\boldsymbol{v}_\perp\|}{d_h}h + w_h \tag{22}$$

B. Sensor Models

Now we need to account for sensor models, which consist of a camera and an accelerometer. We define an acceleration term, used to model non-gravitational acceleration, as

$$a^i = a^i_{thr} + w^i_a \tag{23}$$

where a^i_{thr} is the acceleration due to thrust, and w^i_a is the measured [TODO: What is this term called?]. We can then model the accelerometer with both a bias a measurement noise as

$$\tilde{\boldsymbol{a}}^b = T^b_i\boldsymbol{a}^i + b_a + n_a \tag{24}$$

Note that we need the transform from the inertial frame to the body frame, which is where the sensor will be. The sensor model for the camera is more complex, and can be better understood with a diagram The line of sight vector, l, is defined

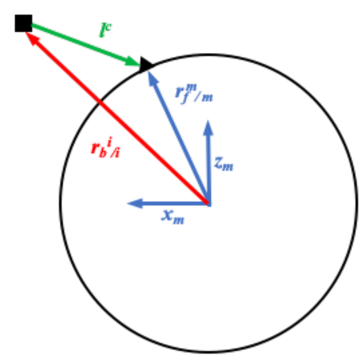

Fig. 2: A diagram for the camera sensor model

as

$$\boldsymbol{l}^c = T^c_bT^b_i(T^i_m\boldsymbol{r}^m_{f/m} - \boldsymbol{r}^i_{b/i}) \tag{25}$$

which makes the measurement model, $\tilde{z}$, is defined as

$$\tilde{\boldsymbol{z}} = \begin{bmatrix} \frac{l_x}{l_z} \\ \frac{l_y}{l_z} \end{bmatrix} + \nu_c \tag{26}$$

C. Design Model Differential Equations

We can now define our design and navigation state equations. The design state model is determined from the truth state model, as is found to be

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{r}}^i_{b/i} \\ \dot{\boldsymbol{v}}^i_{b/i} \\ \dot{b}_r \\ \dot{\boldsymbol{\epsilon}}_g \\ \dot{h} \\ \dot{b}_{ax} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}^i_{b/i} \\ \boldsymbol{a}^i + \boldsymbol{a}^i_{grav} + \boldsymbol{\epsilon}^i_g \\ -\frac{1}{\tau_r}b_r + w_r \\ -\frac{\|\boldsymbol{v}_\perp\|}{d_g}\boldsymbol{\epsilon}_g + \boldsymbol{w}_d \\ -\frac{\|\boldsymbol{v}_\perp\|}{d_h}h + w_h \\ -\frac{1}{\tau_{ax}}b_{ax} + w_{ax} \end{bmatrix} \tag{27}$$

D. Navigation Model Differential Equations

$$\dot{\hat{r}}^i_{b/i} = \hat{v}_{b/i} \tag{28}$$

$$\dot{\hat{v}}_{b/i} = T^i_b(\tilde{a} - \hat{b}_{ax}) \tag{29}$$

$$\dot{\hat{b}}_r = -\frac{1}{\tau_r}\hat{b}_r \tag{30}$$

$$\dot{\hat{\epsilon}}_g = -\frac{\|\hat{v}_\perp\|}{d_g}\epsilon_g \tag{31}$$

$$\dot{\hat{h}} = -\frac{\|\hat{v}_\perp\|}{d_h}\hat{h} \tag{32}$$

$$\dot{\hat{b}}_a x = -\frac{1}{\tau_a x}\hat{b}_a x \tag{33}$$

VI. Model Linearization

In a Kalman filter, the non-linear design states are linearized about the current state estimation to produce

$$\delta\dot{\boldsymbol{x}} = F(\hat{\boldsymbol{x}})\delta\boldsymbol{x} + B\omega \tag{34}$$

We will use the Jacobian of $f(x,y)$, evaluated at $x = \hat{x}$, to obtain the F matrix. Note that here we treat $\boldsymbol{v}_\perp$ as a constant. We do this because $\boldsymbol{v}_\perp$ is very non-linear, so treating it as a constant will greatly simplify the process of taking the Jacobian. F is found to be,

$$F = \left.\frac{df(x,u)}{dx}\right|_{x=\hat{x}}$$
$$= \begin{bmatrix} \mathbf{0}_3 & I_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \frac{-\mu(I_3 - 3i_r i_r^T)}{\|\hat{\boldsymbol{r}}\|^3} & \mathbf{0}_3 & \mathbf{0}_{3\times1} & I_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \frac{-1}{\tau_r} & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \frac{\|\hat{\mathbf{v}}_\perp\|}{dg} I_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 & \frac{\|\hat{\mathbf{v}}_\perp\|}{dh} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 & \mathbf{0}_3 & \frac{-1}{\tau_{accl}} \end{bmatrix} \tag{35}$$

We check this linearization by comparing the results of using this in the simulation to the results of using the non-linear state propagation in simulation. The resulting errors are on the order of 10E-3 over one Kalman cycle, which indicates that we have correctly linearized and we can use this design state.

VII. Linear Measurement Modeling

We are now going to create a linear model of our measurement model. We will achieve an equation in the form of Equation 36

$$\delta\tilde{z} = H(\hat{\boldsymbol{x}})\delta\boldsymbol{x} + G\nu \tag{36}$$

We will obtain H by using a Jacobian linearization about our state estimate, as shown in Equation 37

$$H = \left.\frac{dh(\boldsymbol{x})}{d\boldsymbol{x}}\right|_{x=\hat{x}} \tag{37}$$

We use our design model to obtain H. This makes our measurement model

$$\tilde{x} = h(\boldsymbol{l}(\boldsymbol{x})) + \boldsymbol{\nu} \tag{38}$$

where

$$h(\boldsymbol{l}(\boldsymbol{x})) = \begin{bmatrix} l_x/l_z \\ l_y/l_z \end{bmatrix} \tag{39}$$

Note that we will have to use the chain rule, because l^c is a function of $\boldsymbol{x}$, as shown in Equation 25. The Jacobian used to obtain H is then,

$$H = \left.\frac{dh}{d\boldsymbol{l}}\frac{d\boldsymbol{l}}{d\boldsymbol{x}}\right|_{x=\hat{x}} \tag{40}$$

where

$$\frac{dh}{d\boldsymbol{l}} = \begin{bmatrix} 1/\tilde{l}_z & 0 & -\tilde{l}_x/\tilde{l}_z^2 \\ 0 & 1/\tilde{l}_z & -\tilde{l}_y/\tilde{l}_z^2 \end{bmatrix} \tag{41}$$

and

$$\frac{d\boldsymbol{l}}{d\boldsymbol{x}} = \begin{bmatrix} -T_b^i T_i^b I_3 & 0_{3\times11} \end{bmatrix} \tag{42}$$

Note that in Equation 41, we are assuming that the attitude is well known (which is reflected in our design state equations), so we evaluate l^c at the truth state. Also, note that in Equation 42 we assume that the location of the feature, r_f^m, to be well known, so we use the truth state.

Putting all of this together, we get an H,

$$H = \begin{bmatrix} 1/\tilde{l}_z & 0 & -\tilde{l}_x/\tilde{l}_z^2 \\ 0 & 1/\tilde{l}_z & -\tilde{l}_y/\tilde{l}_z^2 \end{bmatrix} \begin{bmatrix} -T_b^i T_i^b I_3 & 0_{3\times11} \end{bmatrix} \tag{43}$$

We can obtain the camera measurement residual by,

$$\delta\tilde{z} = \tilde{z} - h(\hat{\boldsymbol{x}^-}) \tag{44}$$

where $\hat{\boldsymbol{x}^-}$ is the state immediately before the measurement. This residual will be compared to the residual of our linearized form in order to validate our linearization. Table I compares the propagated errors from both methods in the first Kalman update.

Residuals of Non-Linear and Linear Measurements		
$\delta\tilde{z}$(nl)	$\delta\tilde{z}$(l)	Difference
-1.7016E-4	-1.7023E-4	7.3131E-8
1.3747E-3	8.449E-5	1.2903E-3

TABLE I: Validation of the linearization

We can see that the values in Table I are sufficiently small, meaning that our linearization can be used in our Kalman filter.