Kalman Filter Milestone 1

The Kalman Crew
Syed Shehroze Hussain Rizvi (30076)
Muhammad Shayan Hussain (30077)
Aun Haider (29120)
Moiz Lotia (31584)

I. Problem Description

The objective of this project is to estimate the motion of an object moving in three-dimensional space when complete and accurate measurements of its motion are not directly available. In practical tracking systems, sensors are affected by noise and can only provide partial information about the system state. As a result, quantities such as velocity, acceleration, and higher-order motion derivatives cannot be measured directly and must be inferred from noisy observations.

In this problem, the object's motion is modeled using a constant jerk assumption, which assumes that the rate of change of acceleration remains constant over short time intervals. This model is expressive enough to capture smooth yet rapidly changing motion while remaining mathematically tractable. Due to sensor noise and modeling uncertainty, direct measurements of the true system state are corrupted, making accurate estimation challenging.

To address these challenges, a state-space estimation framework based on Kalman filtering is employed. The Kalman Filter provides an optimal recursive solution for estimating the system state by combining a mathematical motion model with noisy sensor measurements. This milestone focuses on formulating and deriving the mathematical model required for state estimation, laying the foundation for later implementation of Linear and Extended Kalman Filters.

II. State Definition

The system state describes the motion of an object in three-dimensional space along the x, y, and z axes. For each spatial axis, we track four motion parameters:

Position (p): This is the location in three-dimensional space where the object is at the current time. This is the only quantity which is being measured directly, though measurements are corrupted by noise. The unit is meters (m).

Velocity (v): The rate of change of position with respect to time. The unit for velocity is (m/s). Velocity is the first derivative of the position.

Acceleration (a): The rate of change of velocity with respect to time. The unit for acceleration is (m/s^2). Acceleration is the second derivative of the position.

Jerk (j): The rate of change of acceleration with respect to time. The unit for jerk is (m/s^3). Jerk is the third derivative of the position.

Jerk is necessary because real objects have inertia and finite response times. Jerk represents the smoothness of motion. Infinite jerk would imply instantaneous force changes, which is physically impossible. The constant jerk model captures the continuous transition, providing faster convergence during movements, reduced estimation lag, and smoother trajectory estimates.

A. *Equations*

$$\begin{aligned} p(t) &= \text{position} \\ v(t) &= \frac{dp(t)}{dt} \\ a(t) &= \frac{d^2p(t)}{dt^2} \\ j(t) &= \frac{d^3p(t)}{dt^3} \end{aligned}$$

III. Continuous-Time Motion Model

The continuous-time motion model is based on several key assumptions. First, motion along the three spatial axes is considered independent, allowing the same formulation to be applied separately to the x, y, and z directions. Second, jerk is assumed to remain constant over sufficiently small time intervals, which implies that acceleration varies linearly with time within each interval. The system is modeled as linear, and external disturbances or modeling uncertainties are not explicitly included at this stage; these effects will later be incorporated as process noise in the discrete-time formulation.

$$x_x = \begin{bmatrix} p_x \\ v_x \\ a_x \\ j_x \end{bmatrix}, \quad x_y = \begin{bmatrix} p_y \\ v_y \\ a_y \\ j_y \end{bmatrix}, \quad x_z = \begin{bmatrix} p_z \\ v_z \\ a_z \\ j_z \end{bmatrix}$$

The continuous-time kinematic relationships are defined as:

$$\begin{aligned}\dot{p}(t) &= v(t)\\ \dot{v}(t) &= a(t)\\ \dot{a}(t) &= j(t)\\ \dot{j}(t) &= 0\end{aligned}$$

where the dot above each variable denotes differentiation with respect to time.

The final equation, $\dot{j}(t) = 0$, represents the constant jerk assumption, meaning jerk does not change within the considered time interval.

These equations show that each variable is dependent on the others.

IV. Discrete-Time State-Space Model

We discussed the continuous-time model above, but machines work at discrete time periods. So instead of describing how things change continuously, we will describe the state at time $t + \Delta t$.

Assuming jerk is constant over Δt.

Now we will integrate the acceleration equation over Δt:

$$a_k(t + \Delta t) = a_k(t) + j_k(t)\Delta t \tag{1}$$

Now we will integrate the velocity equation over Δt:

$$v_k(t + \Delta t) = v_k(t) + a_k(t)\Delta t + \frac{1}{2}j_k(t)\Delta t^2 \tag{2}$$

Now we will integrate the position equation over Δt:

$$p_k(t+\Delta t) = p_k(t)+v_k(t)\Delta t+\frac{1}{2}a_k(t)\Delta t^2+\frac{1}{6}j_k(t)\Delta t^3 \tag{3}$$

We have defined the equations in the form of k, where k could be any axis: x, y, or z.

Now that we have derived the equations, we will represent each variable in matrix form.

For a single axis, the state vector is:

$$x_k = \begin{bmatrix} p_k \\ v_k \\ a_k \\ j_k \end{bmatrix} \tag{4}$$

We want it for $k + 1$.

In physics, to calculate the evolution of the system over Δt, we use the matrix exponential equation which is:

$$e^{A\Delta t} \tag{5}$$

where A is the continuous-time system matrix. It is used to represent relationships in physics.

$$\dot{x}(t) = Ax(t) \tag{6}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{7}$$

We know that position depends on velocity, so the position element in row 1, column 2 has the value 1 in the matrix. The same rule applies to the rest of the values, which formats our matrix A.

Now instead of computing this matrix exponential every time, we compute:

$$F = e^{A\Delta t} \tag{8}$$

And that F becomes our state transition matrix.

So, from the equation above we form:

$$x_{k+1} = F_{(1D)}x_k + w_k \tag{9}$$

where w_k is the process noise vector.

$1D$ represents a single axis. It indicates that the axes are not linked to each other.

So from the equations we derived above for position, velocity, acceleration, and jerk, the matrix will be:

$$F_{1D} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & \frac{\Delta t^3}{6} \\ 0 & 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

The resulting 12×12 matrix is shown in Figure 1.

$$F = \begin{bmatrix}
1 & \Delta t & \frac{\Delta t^2}{2} & \frac{\Delta t^3}{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} & \frac{\Delta t^3}{6} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} & \frac{\Delta t^3}{6} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Fig. 1. State transition matrix F (12×12).

Each 4×4 block along the diagonal corresponds to one spatial dimension (x, y, or z), and the off-diagonal blocks are zero because we assume motion along each axis is independent.

V. Measurement Model

The measurement model describes how the sensor observes the system state. In this problem, only the position of the object is measured, not its velocity, acceleration, or jerk. The measurement vector at time step k is:

$$z_k = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} + v_k \tag{11}$$

where v_k is the measurement noise vector, which represents random errors introduced by the sensor. This noise is typically modeled as zero-mean Gaussian white noise with covariance matrix R.

The relationship between the measurement and the state can be written as:

$$z_k = Hx_k + v_k \tag{12}$$

where H is the measurement matrix. Since only position is measured, the measurement matrix selects only the position components from the full state vector:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{13}$$

This matrix extracts the position states p_x, p_y, and p_z from the 12-dimensional state vector, ignoring velocity, acceleration, and jerk.

VI. Process Noise Model

The process noise accounts for uncertainties in the motion model. These uncertainties arise from unmodeled dynamics, external disturbances, or simplifications made in the constant jerk assumption. The process noise is represented by the vector w_k, which is assumed to be zero-mean Gaussian white noise with covariance matrix Q.

The process noise covariance matrix Q must be carefully designed to reflect the expected uncertainty in the system. In many applications, the noise is assumed to affect jerk most directly, since jerk is assumed constant in the model, but in reality, it may vary due to external forces or changes in system behavior.

A common approach is to model the process noise as continuous-time white noise acceleration or jerk, which is then discretized for use in the Kalman Filter. The resulting Q matrix captures how uncertainties propagate through the state variables over time.

For a single axis, the process noise covariance can be derived from the continuous-time noise spectral density q. If we assume the noise affects jerk, the discrete-time covariance matrix for one axis is:

$$Q_{1D} = q \begin{bmatrix} \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} \\ \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} & \Delta t \\ \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t & 1 \\ \frac{\Delta t^2}{2} & \Delta t & 1 & 0 \end{bmatrix} \tag{14}$$

The full 12×12 process noise covariance matrix is constructed by placing Q_{1D} along the diagonal for each axis:

$$Q = \begin{bmatrix} Q_{1D} & 0 & 0 \\ 0 & Q_{1D} & 0 \\ 0 & 0 & Q_{1D} \end{bmatrix} \tag{15}$$

where the off-diagonal blocks are zero because the process noise is assumed to be independent across the three spatial dimensions.

The resulting 12×12 matrix is shown in Figure 2.

$$Q = q \begin{bmatrix} \frac{\Delta t^7}{252} & \frac{\Delta t^6}{72} & \frac{\Delta t^5}{30} & \frac{\Delta t^4}{24} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\Delta t^6}{72} & \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\Delta t^5}{30} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{\Delta t^4}{24} & \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\Delta t^7}{252} & \frac{\Delta t^6}{72} & \frac{\Delta t^5}{30} & \frac{\Delta t^4}{24} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\Delta t^6}{72} & \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\Delta t^5}{30} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\Delta t^4}{24} & \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta t^7}{252} & \frac{\Delta t^6}{72} & \frac{\Delta t^5}{30} & \frac{\Delta t^4}{24} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta t^6}{72} & \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta t^5}{30} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\Delta t^4}{24} & \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t \end{bmatrix}$$

Fig. 2. Process noise covariance matrix Q (12×12).

The parameter q controls the magnitude of the process noise and is typically tuned based on the characteristics of the system being tracked. Higher values of q indicate greater uncertainty in the motion model, which causes the Kalman Filter to rely more on measurements rather than predictions.

VII. Measurement Noise Model

The measurement noise represents the random errors introduced by the sensor when measuring the object's position. This noise is modeled as zero-mean Gaussian white noise with covariance matrix R.

The measurement noise covariance matrix is a 3×3 diagonal matrix:

$$R = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{bmatrix} \tag{16}$$

where σ_x^2, σ_y^2, and σ_z^2 are the variances of the measurement noise in the x, y, and z directions, respectively. If the sensor has equal noise characteristics in all directions, this simplifies to:

$$R = \sigma^2 I_3 \tag{17}$$

where I_3 is the 3×3 identity matrix and σ^2 is the variance of the position measurement noise.

The measurement noise covariance is usually determined experimentally by characterizing the sensor or specified by the sensor manufacturer. It quantifies how much we trust the sensor measurements. Lower values of R indicate more reliable measurements, causing the Kalman Filter to give more weight to the measurements. Higher values indicate noisy measurements, causing the filter to rely more on the model predictions.

VIII. Linear Kalman Filter

The Linear Kalman Filter is an optimal recursive algorithm for estimating the state of a linear dynamic system from a series of noisy measurements. It operates in two main steps: prediction and update.

A. Prediction Step

In the prediction step, the filter uses the motion model to estimate the state at the next time step based on the current estimate. This is done before the new measurement is available.

State Prediction:

The predicted state estimate is computed using the state transition matrix:

$$\hat{x}_{k|k-1} = F\hat{x}_{k-1|k-1} \tag{18}$$

where:

- $\hat{x}_{k|k-1}$ is the predicted state estimate at time k given measurements up to time $k-1$
- F is the state transition matrix
- $\hat{x}_{k-1|k-1}$ is the updated state estimate from the previous time step

This equation represents how the state evolves according to the motion model. It projects the previous state forward in time using the known dynamics of the system.

Covariance Prediction:

The predicted covariance matrix, which represents the uncertainty in the state estimate, is updated as:

$$P_{k|k-1} = FP_{k-1|k-1}F^T + Q \tag{19}$$

where:

- $P_{k|k-1}$ is the predicted covariance at time k given measurements up to time $k-1$
- $P_{k-1|k-1}$ is the updated covariance from the previous time step
- F^T is the transpose of the state transition matrix
- Q is the process noise covariance matrix

This equation propagates the uncertainty forward in time. The term $FP_{k-1|k-1}F^T$ represents how the previous uncertainty evolves through the dynamics, and the Q term adds additional uncertainty due to process noise.

B. Update Step

In the update step, the filter incorporates the new measurement to refine the predicted state estimate. This is where the measurement information is fused with the model prediction.

Kalman Gain:

The Kalman gain determines how much the measurement should influence the state estimate:

$$K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + R)^{-1} \tag{20}$$

where:

- K_k is the Kalman gain at time k
- H is the measurement matrix
- H^T is the transpose of the measurement matrix
- R is the measurement noise covariance matrix

The Kalman gain is a weighting factor that balances the trust between the model prediction and the measurement. If the predicted covariance $P_{k|k-1}$ is large (high uncertainty), the gain increases, giving more weight to the measurement. If the measurement noise R is large (unreliable sensor), the gain decreases, relying more on the prediction.

State Update:

The state estimate is corrected using the measurement residual:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - H\hat{x}_{k|k-1}) \tag{21}$$

where:

- $\hat{x}_{k|k}$ is the updated state estimate at time k given measurements up to time k
- z_k is the measurement at time k
- $(z_k - H\hat{x}_{k|k-1})$ is the innovation or measurement residual

The innovation represents the difference between the actual measurement and the predicted measurement. It quantifies how much the measurement disagrees with the prediction. The Kalman gain scales this innovation to produce the correction applied to the predicted state.

Covariance Update:

Finally, the covariance matrix is updated:

$$P_{k|k} = (I - K_kH)P_{k|k-1} \tag{22}$$

where:

- $P_{k|k}$ is the updated covariance at time k
- I is the identity matrix

This equation updates the uncertainty after incorporating measurement information. Before measurement, uncertainty is high. After measurement, uncertainty decreases.

In the update stage of the Kalman Filter, the goal is to correct the predicted state using the new measurement in an intelligent and balanced way. First, the filter calculates a value called the Kalman gain, which decides how much we should trust the model prediction and how much we should trust the sensor measurement. If the prediction is very uncertain, the filter gives more weight to the measurement. If the sensor is noisy, it relies more on the model. Next, the predicted state is adjusted by adding a correction based on the difference between what was measured and what was predicted. This difference shows how wrong the prediction was. Finally, the uncertainty of the estimate is updated and usually becomes smaller, because we have now used real measurement information. Together, these steps allow the filter to continuously improve its estimate over time. Even though only position is directly measured in this problem, the filter is still able to improve the estimates of velocity, acceleration, and jerk by combining the motion model with the measurement information in a recursive and systematic way.

IX. Extended Kalman Filter

The standard Linear Kalman Filter works only when both the system model and the measurement model are linear. However, in many real-world applications such as robotics, aerospace navigation, and object tracking, the system or the sensors behave nonlinearly. To handle nonlinear systems, the Extended Kalman Filter (EKF) is used. The EKF extends the standard Kalman Filter by approximating nonlinear functions using first-order linearization. It is also divided into two steps:

A. Prediction

In the prediction step of the Extended Kalman Filter (EKF), the filter estimates where the system will be in the next moment using the nonlinear motion model. Unlike a simple Kalman Filter that uses a constant state transition matrix, the EKF calculates a Jacobian matrix of the nonlinear model, which helps approximate how the state changes with time. The predicted state is obtained by applying the nonlinear state function to the previous estimate, giving the expected position or condition of the system before seeing the new measurement. At the same time, the predicted covariance, which represents the uncertainty of the state estimate, is updated using the Jacobian of the motion model. To account for errors and uncertainties in the model itself, the process noise covariance (Q) is added to the predicted covariance. This way, after the prediction step, we have both a predicted state estimate and its associated uncertainty, ready for the next step where new measurements will be incorporated.

B. Update

The update stage in the EKF is different from the Linear Kalman Filter because the measurement model is nonlinear. Instead of a simple matrix, measurements are represented by a nonlinear function $h(x)$. To handle this, the EKF calculates the Jacobian of the measurement function, which replaces the usual H matrix and shows how the measurements relate to the state.

Kalman Gain:

The Kalman gain is calculated using the predicted covariance, the measurement Jacobian, and the measurement noise (R). It tells the filter how much the predicted state should be corrected based on the new measurement. If the gain is high, the measurement is trusted more; if low, the prediction is trusted more.

State Update:

The predicted state is then corrected using the difference between the actual measurement and the predicted measurement, called the innovation. The Kalman gain scales this difference, adjusting the state to match the measurement more accurately.

Covariance Update:

Finally, the covariance matrix is updated to show reduced uncertainty after using the measurement. This updated covariance reflects how confident the filter is in the corrected state, finishing the update step.

X. Nonlinear Measurement Model

In practical sensing systems such as radar or vision-based tracking, measurements are often obtained in spherical coordinates rather than Cartesian coordinates. Although the state vector of the system contains position, velocity, acceleration, and jerk components in Cartesian form, only the position states are required to compute the measurements in this model.

The measurement vector at time step n is defined as:

$$z_n = \begin{bmatrix} r \\ \theta \\ \phi \end{bmatrix} \tag{23}$$

where:

- r is the range (radial distance)
- θ is the azimuth angle (horizontal bearing)
- ϕ is the elevation angle (vertical bearing)

Range (r): This is the straight-line distance from the origin (or sensor) to the object. It is calculated as the square root of the sum of squares of x, y, and z positions:

$$r = \sqrt{p_x^2 + p_y^2 + p_z^2} \tag{24}$$

Range provides the scale or magnitude of the position vector. Without range information, we know only the direction to the target, not its distance.

Azimuth angle (θ): This is the angle in the xy-plane measured from the x-axis to the point (like a compass direction). It is given by:

$$\theta = \arctan 2(p_y, p_x) \tag{25}$$

It indicates where the object is in the horizontal plane relative to the sensor's orientation.

Elevation angle (ϕ): This is the angle from the xy-plane up to the object (how "high" it is). It is given by:

$$\phi = \arctan 2\left(p_z, \sqrt{p_x^2 + p_y^2}\right) \tag{26}$$

Positive elevation indicates the object is above the sensor; negative elevation indicates it is below.

XI. Jacobian Matrix

In the measurement model, measurements are calculated using complex mathematical functions of position (involving square roots and angles). The standard Kalman Filter needs measurements in a simple form where you just multiply the position by a matrix. So, it cannot work directly with these complex calculations.

The Extended Kalman Filter (EKF) solves this by approximating the complex measurement function with a simpler, straight-line version. It does this using something called a Jacobian matrix, which is calculated based on the current best guess of where things are. This Jacobian matrix becomes the simplified measurement matrix that the EKF uses in its update step.

The nonlinear measurement function can be written as:

$$z_k = h(x_k) \tag{27}$$

The Jacobian matrix is defined as the matrix of partial derivatives of the measurement function with respect to the state variables:

$$H_k = \frac{\partial h(x)}{\partial x} \tag{28}$$

This value is calculated at the predicted state estimate. Since the measurement only depends on position, the partial derivatives with respect to velocity, acceleration, and jerk are zero.

We will derive the matrix by calculating the partial derivative of each component with respect to the position vector.

A. Partial Derivatives of r

$$\frac{\partial r}{\partial p_x} = \frac{p_x}{r} \tag{29}$$

$$\frac{\partial r}{\partial p_y} = \frac{p_y}{r} \tag{30}$$

$$\frac{\partial r}{\partial p_z} = \frac{p_z}{r} \tag{31}$$

B. Partial Derivatives of θ

$$\frac{\partial \theta}{\partial p_x} = \frac{-p_y}{p_x^2 + p_y^2} \tag{32}$$

$$\frac{\partial \theta}{\partial p_y} = \frac{p_x}{p_x^2 + p_y^2} \tag{33}$$

$$\frac{\partial \theta}{\partial p_z} = 0 \tag{34}$$

C. Partial Derivatives of ϕ

Let:

$$\rho = \sqrt{p_x^2 + p_y^2} \tag{35}$$

Then:

$$\frac{\partial \phi}{\partial p_x} = \frac{-p_x p_z}{r^2 \rho} \tag{36}$$

$$\frac{\partial \phi}{\partial p_y} = \frac{-p_y p_z}{r^2 \rho} \tag{37}$$

$$\frac{\partial \phi}{\partial p_z} = \frac{\rho}{r^2} \tag{38}$$

So, the final Jacobian matrix will be:

$$H_k = \begin{bmatrix} \frac{p_x}{r} & 0 & 0 & 0 & \frac{p_y}{r} & 0 & 0 & 0 & \frac{p_z}{r} & 0 \\ 0 & 0 \\ \frac{-p_y}{p_x^2+p_y^2} & 0 & 0 & 0 & \frac{p_x}{p_x^2+p_y^2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 \\ \frac{-p_x p_z}{r^2\rho} & 0 & 0 & 0 & \frac{-p_y p_z}{r^2\rho} & 0 & 0 & 0 & \frac{\rho}{r^2} & 0 \\ 0 & 0 \end{bmatrix} \tag{39}$$

As the only measurement is position, all the other values are 0.

The Jacobian matrix represents the sensitivity of each measurement component to small changes in position. It provides a local linear approximation of the nonlinear measurement function around the current estimate. During the EKF update step, this matrix replaces the constant measurement matrix used in the Linear Kalman Filter.

Because the measurement model is nonlinear, the Jacobian must be recomputed at every time step using the current predicted state estimate. This dynamic linearization enables the EKF to handle nonlinear measurements while maintaining a recursive estimation framework.