Signatory of DORA
https://doi.org/10.56183/iberotecs.v3i1.596
Ibero-American Journal of Engineering & Technology Studies
OPEN ACCESS
BY-NC-ND
Received: 11/01/2023
Accepted: 29/03/2023
Published: 15/04/2023

Aplicación de los filtros de Kalman en los sistemas de navegación autónoma

Application of Kalman filters in autonomous navigation systems

Leonela Del Rocio De La A Salinas
leo_1019@hotmail.es
https://orcid.org/0000-0002-0320-4397
Universidad Estatal Península de Santa Elena, Ecuador

Rolando Edison Carrera-Fernández
r.carrera@hotmail.com
https://orcid.org/0000-0002-1965-4915
Universidad Católica de Santiago de Guayaquil, Ecuador

Ismael Elías Erazo-Velasco
ismael.erazo@utelvt.edu.ec
https://orcid.org/0000-0002-7647-4611
Universidad Técnica Luis Vargas Torres de Esmeraldas, Ecuador

RESUMEN

El presente trabajo trata sobre la aplicación de los filtros de Kalman en los sistemas de navegación autónoma. En este contexto, los sistemas de navegación autónoma dependen de una variedad de sensores para recopilar información sobre el entorno y tomar decisiones en consecuencia. Sin embargo, la información que se recopila de estos sensores suele estar contaminada por ruido, errores y otros tipos de incertidumbre. Por lo tanto, se utilizan filtros de Kalman para mejorar la precisión de la información. El artículo explica cómo funciona el filtro de Kalman y cómo se aplica en los sistemas de navegación autónoma. Además, se mencionan algunas de las aplicaciones del filtro de Kalman en los sistemas de navegación autónoma, como la estimación de la posición, la velocidad y la orientación del vehículo, así como la fusión de datos de sensores para mejorar la precisión de la estimación del estado del vehículo. En general, el artículo destaca la importancia del filtro de Kalman en los sistemas de navegación autónoma para mejorar la precisión de la estimación del estado del vehículo y, por lo tanto, mejorar la seguridad y eficiencia de la navegación autónoma.

Palabras claves: Filtro de Kalman, robot diferencial, espacio de estados, simulación en Python, señales aleatorias.

ABSTRACT

This paper deals with the application of Kalman filters in autonomous navigation systems. In this context, autonomous navigation systems rely on a variety of sensors to collect information about the environment and make decisions accordingly. However, the information collected from these sensors is often contaminated by noise, errors, and other types of uncertainty. Therefore, Kalman filters are used to improve the accuracy of the information. The article explains how the Kalman filter works and how it is applied in autonomous navigation systems. In addition, some of the applications of the Kalman filter in autonomous navigation systems are mentioned, such as the estimation of vehicle position, speed and orientation, as well as the fusion of sensor data to improve the accuracy of the estimation of the vehicle. vehicle status. Overall, the article highlights the importance of the Kalman filter in autonomous navigation systems to improve the accuracy of vehicle state estimation and thus improve the safety and efficiency of autonomous navigation.

Keywords: Kalman filter, differential robot, state space, Python simulation, random signals.

INTRODUCCIÓN

Los sistemas de navegación autónoma, como los, drones y robots, dependen de una variedad de sensores para recopilar información sobre su entorno y tomar decisiones en consecuencia. Sin embargo, la información que se recopila de estos sensores suele estar contaminada por ruido, errores y otros tipos de incertidumbre. Para mejorar la precisión de la información, se utilizan filtros como los filtros de Kalman (Alvarez Casadiego, 2021).

Los filtros de Kalman son una clase de algoritmos que permiten estimar el estado de un sistema dinámico a partir de un conjunto de mediciones ruidosas. Se basan en la teoría de probabilidad y en el análisis de señales para obtener una estimación del estado del sistema con una precisión mayor que la que se podría obtener simplemente a partir de las mediciones (Bravo, 2013).

Orígenes de los filtros de Kalman

Los filtros de Kalman fueron desarrollados por Rudolf Kalman en la década de 1960. Inicialmente, se utilizaron en la industria aeroespacial para el seguimiento de objetos en movimiento, pero posteriormente se extendieron a otros campos, como la ingeniería eléctrica, la mecánica y la robótica (Cardona, 2008).

El filtro de Kalman es un algoritmo matemático que utiliza la estadística bayesiana para estimar el estado de un sistema dinámico a partir de una serie de medidas imprecisas y ruidosas. El filtro de Kalman funciona en dos fases: predicción y corrección (Guevara Niño, 2020).

En la fase de predicción, se utiliza el modelo matemático del sistema para predecir el estado futuro del sistema. La predicción se basa en el estado actual del sistema y en la entrada del sistema, como la velocidad y la aceleración. Esta predicción tiene en cuenta la incertidumbre del sistema y se representa mediante una distribución de probabilidad (Méndez Mejía, 2023). En la fase de corrección, se utiliza la información de los sensores para actualizar la predicción y mejorar la

estimación del estado del sistema. Esta actualización se basa en la ley de Bayes, que establece cómo se actualiza la distribución de probabilidad a partir de nuevas evidencias (Parra Tsunekawa, 2014).

La salida del filtro de Kalman es una estimación del estado del sistema con una medida de incertidumbre.

El filtro de Kalman básico

El filtro de Kalman básico utiliza un modelo de estado del sistema y un modelo de medición para estimar el estado del sistema., modelo de estado describe cómo evoluciona el estado del sistema a lo largo del tiempo, mientras que el modelo de medición describe cómo se relaciona el estado del sistema con las mediciones realizadas. El filtro de Kalman utiliza estas dos descripciones para obtener una estimación óptima del estado del sistema (Ávila, 2015).

Variantes del filtro de Kalman

Existen varias variantes del filtro de Kalman, cada una con sus propias características y aplicaciones. Algunas de las variantes más comunes son el filtro de Kalman extendido (EKF), el filtro de Kalman unscented (UKF), el filtro de Kalman de múltiples modelos (MMKF) y el filtro de Kalman de partículas (PF) (Vivanco, 2015).

Aplicaciones en sistemas de navegación autónoma

Los sistemas de navegación autónoma utilizan una variedad de sensores para medir la posición, velocidad y orientación del vehículo. Estos sensores pueden incluir GPS, IMU, LIDAR, cámaras, entre otros. Sin embargo, cada uno de estos sensores tiene sus propias limitaciones y errores de medición, lo que puede afectar la precisión y confiabilidad de la estimación de estado del sistema (Puertas Ramírez, 2018).

Los filtros de Kalman permiten combinar las mediciones de múltiples sensores y el modelo del sistema para estimar de manera óptima el estado del sistema en cada instante de tiempo. Los filtros de Kalman utilizan dos modelos matemáticos del sistema: el modelo de transición de estado y el modelo de observación (Penizzotto, 2019).

El modelo de transición de estado describe cómo evoluciona el estado del sistema en el tiempo, mientras que el modelo de observación describe cómo se relacionan las mediciones con el estado del sistema (Ávila, 2015).

Los filtros de Kalman se dividen en dos fases: la fase de predicción y la fase de actualización. En la fase de predicción, se utiliza el modelo de transición de estado para predecir el estado del sistema en el siguiente instante de tiempo (Méndez Mejía, 2023).

En la fase de actualización, se utiliza el modelo de observación y las mediciones disponibles para actualizar la estimación del estado del sistema.

En sistemas de navegación autónoma, los filtros de Kalman se utilizan para estimar la posición, velocidad y orientación del vehículo en cada instante de tiempo. El modelo de transición de estado puede ser un modelo cinemático simple del vehículo, mientras que el modelo de observación puede incluir mediciones de GPS, IMU, LIDAR, cámaras, entre otros. (Marquetti Gómez, Doctoral dissertation, Universidad Central" Marta Abreu" de Las Villas))

Uno de los mayores desafíos en la implementación de filtros de Kalman en sistemas de navegación autónoma es la selección de los parámetros del filtro, como la matriz de covarianza del ruido del proceso y la matriz de covarianza del ruido de medición (Cardona, 2008).

La selección adecuada de estos parámetros es esencial para lograr una estimación precisa y confiable del estado del sistema.

Robots diferenciales de navegación

Los robots diferenciales son una clase de robots móviles que se utilizan en diversas aplicaciones, como en la exploración espacial, la agricultura, la construcción y la industria manufacturera. Estos robots utilizan un modelo cinemático que describe su movimiento y se utiliza para planificar su trayectoria (Méndez Mejía, 2023).

Un robot diferencial es un tipo de robot móvil que se mueve mediante la rotación independiente de dos ruedas, lo que permite cambiar su posición y orientación en un espacio bidimensional. Para modelar el comportamiento cinemático de un robot diferencial en espacios de estados, se utiliza el modelo de velocidad de cada rueda (Vilcahuamán Espinoza, 2014).

En este modelo, se considera que la velocidad lineal del robot está dada por la media de las velocidades de las ruedas, mientras que la velocidad angular está dada por la diferencia entre las velocidades de las ruedas dividida por la distancia entre ellas (Penizzotto, 2019).

Matemáticamente, este modelo se expresa de la siguiente manera:

$V = (vr + vl) / 2$

$\omega = (vr - vl) / d$

Donde v es la velocidad lineal del robot, ω es la velocidad angular, vr y vl son las velocidades de las ruedas derecha e izquierda, respectivamente, y d es la distancia entre las ruedas.

Una vez obtenidas las velocidades lineal y angular, se pueden calcular la posición y orientación del robot en el espacio de estados. Para esto, se utiliza la fórmula de integración de Euler:

$X_{K+1} = X_K + V * \cos(\theta_k) * Ts$

$Y_{k+1} = Y_K + V * \sin(\theta_k) * Ts$

$\theta_k = \theta_{k-1} + \omega * Ts$

donde X_{K+1}, Y_{K+1} y θ_k representan la posición y orientación del robot, respectivamente, y Ts es el intervalo de tiempo entre dos medidas consecutivas de las velocidades de las ruedas.

Modelo matemático del robot diferencial es espacios estados

Un robot diferencial es un tipo de robot móvil que utiliza dos ruedas para el movimiento y dirección. Para describir el movimiento y la posición de un robot diferencial, se puede utilizar un modelo en espacio de estados (Penizzotto, 2019).

Este modelo se compone de dos ecuaciones diferenciales, una para la posición y otra para la orientación del robot.

La posición del robot se puede describir utilizando dos variables: la distancia recorrida por la rueda izquierda (x1) y la distancia recorrida por la rueda derecha (x2). La orientación del robot se puede describir utilizando un ángulo (θ) (Marquetti Gómez, Doctoral dissertation, Universidad Central" Marta Abreu" de Las Villas).

Las ecuaciones diferenciales que describen la dinámica del robot se pueden escribir como:

$$\frac{dx_1}{dt} = v \cos(\theta) \qquad \{6\}$$

$$\frac{dx_2}{dt} = v \sin(\theta) \qquad \{7\}$$

$$\frac{dx_3}{dt} = \omega \qquad \{8\}$$

Donde v es la velocidad lineal del robot, ω es la velocidad angular y θ es la orientación del robot en radianes.

Estas ecuaciones diferenciales se pueden expresar en forma matricial en un vector de estado

$x=[x_1, x_2, \theta]$ y un vector de entrada $u=[v, \omega]$ de la siguiente manera:

$$\frac{dx}{dt} = f(x, u)$$

Donde f(x,u) es una función que define las ecuaciones diferenciales del modelo. En el caso del robot diferencial, esta función se puede escribir como:

$$f(x,u) = \begin{vmatrix} v\cos(\theta) \\ v\sin(\theta), \\ \omega \end{vmatrix}$$

Este modelo en espacio de estados se puede utilizar para controlar el movimiento del robot diferencial. Por ejemplo, se puede diseñar un controlador que tome como entrada la posición deseada del robot y utilice este modelo para calcular los valores de v y ω que se deben aplicar para que el robot se mueva hacia la posición deseada (Parra Tsunekawa, 2014).

Figura 1. Robot diferencial de navegación.

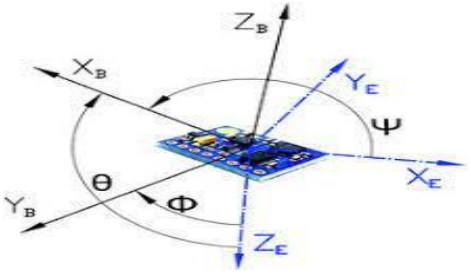
Sistemas de medición inercial de movimiento (IMU)

Los sensores IMU, que son dispositivos que miden la aceleración, la velocidad angular y la orientación de un objeto en el espacio tridimensional (Méndez Mejía, 2023).

Estos sensores se utilizan en una amplia variedad de aplicaciones, desde robots autónomos hasta sistemas de navegación de aviones y drones. los sensores IMU están compuestos por acelerómetros y giroscopios, y que su precisión y capacidad para medir los cambios de movimiento en tiempo real los hace especialmente útiles para aplicaciones que requieren alta precisión y rapidez de respuesta.

A pesar de sus limitaciones, los sensores IMU siguen siendo una herramienta esencial en una amplia variedad de industrias (Guevara Niño, 2020).

Figura 2. Sensor IMU (acelerómetro y giroscopio).



Modelación y simulación de un filtro de Kalman para un robot diferencial en Python

```
"""
FILTRO DE KALMAN PARA UN SISTEMA DE NAVEGACION AUTONOMA TERRESTRE
author:  ISMAEL ERAZO Y LEONELA SALINAS
"""
import sys
import pathlib
sys.path.append(str(pathlib.Path(__file__).parent.parent.parent))
import math
import matplotlib.pyplot as plt
import numpy as np
import scipy.linalg
from utils.angle import rot_mat_2d

# Covariance for UKF simulation
Q = np.diag([
    0.1,  # variance of location on x-axis
    0.1,  # variance of location on y-axis
    np.deg2rad(1.0),  # variance of yaw angle
    1.0  # variance of velocity
]) ** 2  # predict state covariance
R = np.diag([1.0, 1.0]) ** 2  # Observation x,y position covariance

#  Simulation parameter
INPUT_NOISE = np.diag([1.0, np.deg2rad(30.0)]) ** 2
GPS_NOISE = np.diag([0.5, 0.5]) ** 2

DT = 0.1  # time tick [s]
SIM_TIME = 50.0  # simulation time [s]

#  UKF Parameter
ALPHA = 0.001
BETA = 2
KAPPA = 0

show_animation = True
def calc_input():
    v = 1.0  # [m/s]
    yawRate = 0.1  # [rad/s]
    u = np.array([[v, yawRate]]).T
    return u

def observation(xTrue, xd, u):
    xTrue = motion_model(xTrue, u)

    # add noise to gps x-y
    z = observation_model(xTrue) + GPS_NOISE @ np.random.randn(2, 1)

    # add noise to input
    ud = u + INPUT_NOISE @ np.random.randn(2, 1)

    xd = motion_model(xd, ud)

    return xTrue, z, xd, ud


def motion_model(x, u):
    F = np.array([[1.0, 0, 0, 0],
                  [0, 1.0, 0, 0],
                  [0, 0, 1.0, 0],
                  [0, 0, 0, 0]])

    B = np.array([[DT * math.cos(x[2]), 0],
                  [DT * math.sin(x[2]), 0],
                  [0.0, DT],
                  [1.0, 0.0]])

    x = F @ x + B @ u

    return x


def observation_model(x):
    H = np.array([
        [1, 0, 0, 0],
        [0, 1, 0, 0]
    ])

    z = H @ x

    return z
```

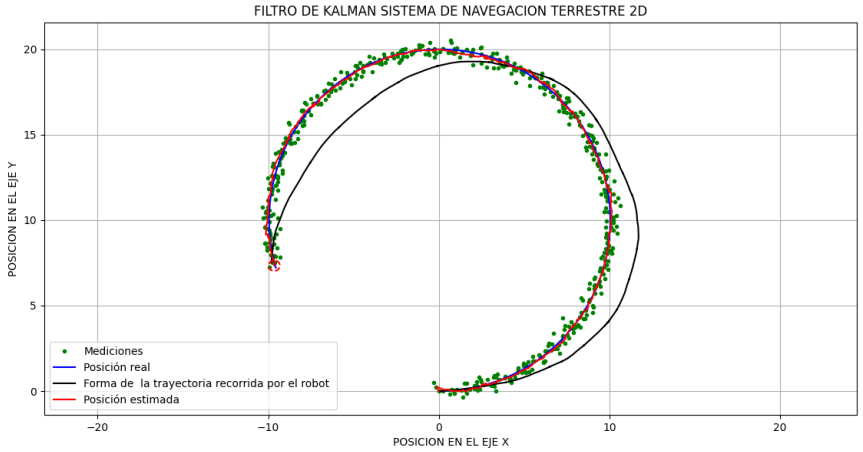
```
def generate_sigma_points(xEst, PEst, gamma):
    sigma = xEst
    Psqrt = scipy.linalg.sqrtm(PEst)
    n = len(xEst[:, 0])
    # Positive direction
    for i in range(n):
        sigma = np.hstack((sigma, xEst + gamma * Psqrt[:, i:i + 1]))

    # Negative direction
    for i in range(n):
        sigma = np.hstack((sigma, xEst - gamma * Psqrt[:, i:i + 1]))

    return sigma


def predict_sigma_motion(sigma, u):
    """
        Sigma Points prediction with motion model
    """
    for i in range(sigma.shape[1]):
        sigma[:, i:i + 1] = motion_model(sigma[:, i:i + 1], u)

    return sigma


def predict_sigma_observation(sigma):
    """
        Sigma Points prediction with observation model
    """
    for i in range(sigma.shape[1]):
        sigma[0:2, i] = observation_model(sigma[:, i])

    sigma = sigma[0:2, :]

    return sigma


def calc_sigma_covariance(x, sigma, wc, Pi):
    nSigma = sigma.shape[1]
    d = sigma - x[0:sigma.shape[0]]


    P = Pi
    for i in range(nSigma):
        P = P + wc[0, i] * d[:, i:i + 1] @ d[:, i:i + 1].T
    return P


def calc_pxz(sigma, x, z_sigma, zb, wc):
    nSigma = sigma.shape[1]
    dx = sigma - x
    dz = z_sigma - zb[0:2]
    P = np.zeros((dx.shape[0], dz.shape[0]))

    for i in range(nSigma):
        P = P + wc[0, i] * dx[:, i:i + 1] @ dz[:, i:i + 1].T

    return P


def ukf_estimation(xEst, PEst, z, u, wm, wc, gamma):
    #  Predict
    sigma = generate_sigma_points(xEst, PEst, gamma)
    sigma = predict_sigma_motion(sigma, u)
    xPred = (wm @ sigma.T).T
    PPred = calc_sigma_covariance(xPred, sigma, wc, Q)

    #  Update
    zPred = observation_model(xPred)
    y = z - zPred
    sigma = generate_sigma_points(xPred, PPred, gamma)
    zb = (wm @ sigma.T).T
    z_sigma = predict_sigma_observation(sigma)
    st = calc_sigma_covariance(zb, z_sigma, wc, R)
    Pxz = calc_pxz(sigma, xPred, z_sigma, zb, wc)
    K = Pxz @ np.linalg.inv(st)
    xEst = xPred + K @ y
    PEst = PPred - K @ st @ K.T

    return xEst, PEst


def plot_covariance_ellipse(xEst, PEst):  # pragma: no cover
```

```
    Pxy = PEst[0:2, 0:2]
    eigval, eigvec = np.linalg.eig(Pxy)

    if eigval[0] >= eigval[1]:
        bigind = 0
        smallind = 1
    else:
        bigind = 1
        smallind = 0

    t = np.arange(0, 2 * math.pi + 0.1, 0.1)
    a = math.sqrt(eigval[bigind])
    b = math.sqrt(eigval[smallind])
    x = [a * math.cos(it) for it in t]
    y = [b * math.sin(it) for it in t]
    angle = math.atan2(eigvec[1, bigind], eigvec[0, bigind])
    fx = rot_mat_2d(angle) @ np.array([x, y])
    px = np.array(fx[0, :] + xEst[0, 0]).flatten()
    py = np.array(fx[1, :] + xEst[1, 0]).flatten()
    plt.plot(px, py, "--r")


def setup_ukf(nx):
    lamb = ALPHA ** 2 * (nx + KAPPA) - nx
    # calculate weights
    wm = [lamb / (lamb + nx)]
    wc = [(lamb / (lamb + nx)) + (1 - ALPHA ** 2 + BETA)]
    for i in range(2 * nx):
        wm.append(1.0 / (2 * (nx + lamb)))
        wc.append(1.0 / (2 * (nx + lamb)))
    gamma = math.sqrt(nx + lamb)

    wm = np.array([wm])
    wc = np.array([wc])

    return wm, wc, gamma


def main():
    print(__file__ + " start!!")
```

Resultados obtenidos

En la Figura 3 podemos observar como el filtro de Kalman va estimando las posiciones y las velocidades de lo robots diferencial de navegación, a medida que tengamos más información sobre señales de la posición y velocidad de robots proporcionadas por el sensor IMU, vamos estimar con mayor la trayectoria descrita por el robot, a través de esta información podemos realizar también un mapa global del entorno donde se mueve nuestro robot diferencial.

Figura 3. Graficas obtenidas por la simulación en Python.



CONCLUSIONES

En conclusión, el modelo cinemático de un robot diferencial es fundamental para describir su movimiento y planificar su trayectoria. El modelo cinemático se basa en la geometría del robot y la relación entre la velocidad de las ruedas y la velocidad del robot en su conjunto. Además, los algoritmos de planificación de trayectorias utilizan el modelo cinemático para encontrar la mejor trayectoria para el robot. En conjunto, el modelo cinemático permite a los robots diferenciales moverse de manera precisa y eficiente en una variedad de aplicaciones.

REFERENCIAS

Alvarez Casadiego, N. &. (2021). Implementación de algoritmos filtro de Kalman y filtro de Partículas para localización y mapeo simultáneo aplicado a un robot móvil en ambientes interiores con variaciones de iluminación. Guayaquil - Ecuador.

Ávila, M. A. (2015). Sistema de localización autónoma para robots móviles basado en fusión de sensores propioceptivos. Revista Politécnica, 75-84.

Bravo, V. A. (2013). Análisis y aplicación del filtro de Kalman a una señal con ruido aleatorio. Madrid: Scientia et technica, 18(1), 267-274.

Cardona, L. C. (2008). Sistema de Navegación para vehículos no tripulados. Congreso Latinoamericano de Control Automático,, (págs. 1-9). Monterrey .

Enciso Salas, L. M. (2015). Diseño de un sistema de navegación autónomo para robots móviles usando fusión de sensores y controladores neuro difusos. PROC NAT ACAD SCI USA, 13.-18.

Font, J. M. (2006). Posicionamiento de robots móviles mediante un filtro de Kalman angular y triangulación. Información tecnológica, 17(5),, 9-14.

García Aguilar, J. (2019). Desarrollo de sistemas de navegación autónoma en interiores para un UAV. Ingeneus .

Guevara Niño, B. D. (2020)}). Posicionamiento de un automóvil en espacios reducidos utilizando un sistema de navegación inercial con filtros de kalman. Lima.

Marquetti Gómez, Y. (Doctoral dissertation, Universidad Central" Marta Abreu" de Las Villas)). Integración GPS/INS para la navegación de vehículos autónomos. Quito .

Méndez Mejía, M. (2023). Integración del filtro de Kalman a un Sistema de Posicionamiento Global (GPS) para aplicación en vehículos autónomos. Guadalajara .

Parra Tsunekawa, S. I. (2014). Generación de mapa de entorno para navegación de vehículo terrestre autónomo. Ingeneus .

Penizzotto, F. A. (2019). Sistema de control basado en fusión para la navegación autónoma de vehıculos. Bogota .

Puertas Ramírez, D. (2018). Análisis y filtrado de datos de sistema de navegación por satélite para navegación autónoma de vehículos. Santa Fe: (Bachelor's thesis).

Vilcahuamán Espinoza, G. (2014). Diseño e implementación de un sistema de navegación autónomo basado en el filtro extendido de Kalman en el módulo robótico CoroBot empleando lenguaje C. JAM CHEM SOC ISSN: 0002-7863, 5-8.

Vivanco, P. J. (2015). Filtro de Kalman extendido aplicado en la navegación de un AUV. Ingenius:. Ingenius: Revista de Ciencia y Tecnología, (13), 12-19., 3-5.