Informe Tarea 2: "Extended Kalman Filters y Unscented Kalman Filters"

Inteligencia Artificial

Luciano Andrés Villagrán Cabala

Entrega: 9 de Octubre 2022

Índice

1. Introducción y contexto

El objetivo de esta tarea es analizar la trayectoria y estimar la posición de un robot submarino en un cierto instante de tiempo por medio de sensores. Para lograr este cometido, se utiliza un tipo de filtro de Kalman llamado **Unscented Kalman Filters (UKF)**. Estos representan un filtro equivalente a los **Extended Kalman Filters (EKF)**, pero superiores en rendimiento computacional, sobre todo cuando la no linealidad del problema es de mayor grado. El robot submarino en cuestión posee tres tipos de sensores: un IMU (Inertial Measurement Unit) que permite medir aceleraciones lineales y velocidades angulares, un compás que indica una posición relativa y encoders que permiten medir velocidad longitudinal.

Con esta información en consideración se procederá a responder las preguntas solicitadas en el enunciado. Tres de dichas preguntas piden realizar además comparaciones gráficas, las cuales podrán ser vistas más adelante en el presente documento.

2. Comparación EKF y UKF

En primer lugar, cabe destacar que ambos tipos de filtros son útiles a la hora de enfrentar problemas no lineales. La decisión sobre que tipo usar dependerá de los índices de no linealidad, tanto del proceso como del modelo de observaciones.

Por su parte, los **EKF** basan su funcionamiento en el uso de dos funciones no lineales de tipo diferenciable, sin embargo dichas funciones no pueden ser aplicadas directamente a la covarianza del sistema, por lo que se genera una matriz de derivadas parciales o Jacobiano, el cual en cada paso temporal (instante) es evaluado con los valores actuales de los estados predichos. Las matrices resultantes a cada paso temporal son entonces usadas en las ecuaciones de los filtros de Kalman, con lo que se logra linealizar ambas funciones iniciales alrededor de un valor actual para la estimación realizada.

Los **UKF** en cambio, son usados cuando el grado de no linealidad de las funciones es muy alto, ya que en estos casos los EKF suelen tener un rendimiento muy pobre. La principal diferencia entre ambos tipos de filtros de Kalman recae en que los UKF plantean una solución al problema expuesto recientemente, mediante el uso de una técnica determinista de muestreo se elige un conjunto mínimo de puntos (llamados puntos sigma) alrededor de la media, dichos puntos luego se propagan a través de las funciones no lineales, generando una nueva media y una nueva covarianza estimada. De esta forma se evita la necesidad de calcular los Jacobianos, lo que resulta una tarea difícil en si misma, sobre todo para el caso de funciones complejas.

Finalmente cabe destacar que la dificultad de implementar cualquier tipo de filtro de Kalman para sistemas no lineales se origina en la estabilidad numérica requerida para que sea preciso. En este sentido, los problemas de estabilidad que afrontan los UKF generalmente nacen de la aproximación de la raíz cuadrada de la matriz de covarianza, mientras que los problemas de los EKF tienen su base en aproximaciones realizadas en múltiples pasos del desarrollo de las series de Taylor durante el transcurso del algoritmo. Con todo esto en consideración es posible afirmar que los UKF tienden a ser más robustos y más precisos que los EKF en sus estimaciones del error en todas las direcciones.

3. Comparación de trayectorias, sin cambios

Para llevar a cabo lo solicitado en el enunciado se hace uso del código entregado, el archivo llamado *tarea2.py* solo fue modificado en lo referente a la generación de los gráficos. Las líneas de código que cumplen dicha función pueden observarse a continuación.

```
plt.figure(1)
plt.plot(t,real_state)
plt.xlabel("Factores de estado")
plt.ylabel("Valores de cada factor")
plt.title('48 Trayectorias reales, sin ajustes')
plt.show()
plt.savefig("real-b.png")

plt.figure(2)
plt.plot(t,state_estimator.get_state())
plt.xlabel("Factores de estado")
plt.ylabel("Valores de cada factor")
plt.title('48 Trayectorias estimadas, sin ajustes')
plt.show()
plt.savefig("estimado-b.png")
```

Dichas líneas estan ubicadas justo despues de los *print* en el código entregado. Cabe mencionar además la definición de la variable "t" la cual es creada justo antes de iniciar el ciclo *for* y responde a la siguiente sentencia.

```
t = np.linspace(1, 6, 6)
```

Con todo esto en consideración se obtienen los gráficos de las figuras 1 y 2.

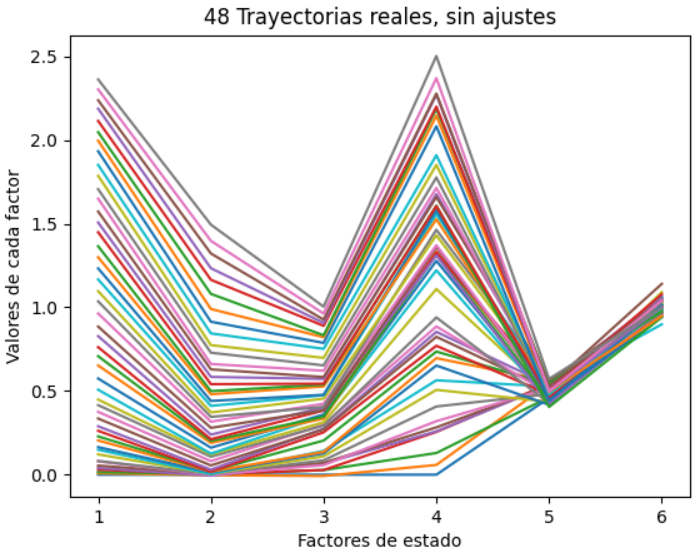

Figura 1: Trayectoria real del robot.

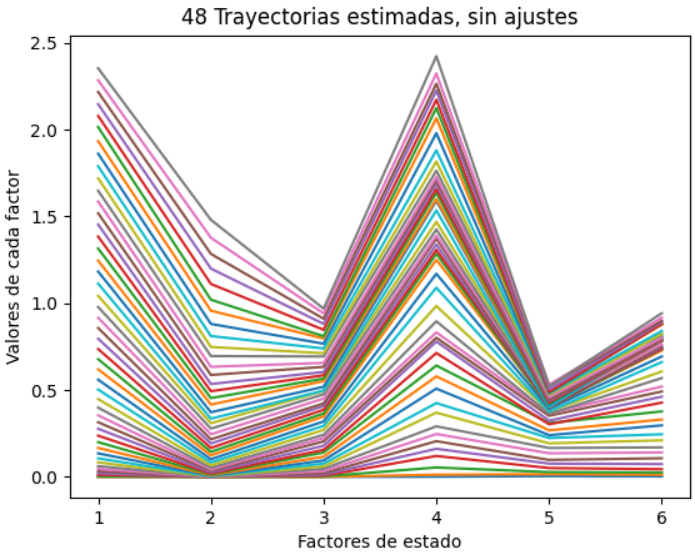

Figura 2: Trayectoria estimada del robot.

Al comparar ambos gráficos obtenidos se pueden notar claras diferencias en los valores mostrados, sobre todo para los estados 2, 4 y 6. Considerando este último se podría llegar a afirmar que el uso de UKF en este caso dispersa los resultados del final de la trayectoria para la gran mayoría de los 48 casos generados.

4. Comparación de trayectorias, cambios en organización de sensores

De manera similar al caso anterior, para realizar lo solicitado se tiene el archivo *tarea2c.py* el cual solo fue modificado para incluir el gráfico de la figura 3 (ya que la trayectoria real se mantiene como en la figura 1), además de lograr la reorganización del funcionamiento de los sensores. Este último cambio puede verse reflejado en las siguientes líneas de código.

```
r_comp_enc = np.zeros([2, 2])
r_comp_enc[0][0] = 0.02
r_comp_enc[1][1] = 0.001
```

Donde se ajusta el error del *compass* y del *encoder* para funcionar en conjunto.

```
compass_hdg = row[9]
encoder_vel = row[10]
comp_enc_data = np.array([encoder_vel, compass_hdg])
```

Donde se le generan datos al par de sensores trabajando juntos, de manera similar a los sensores IMU.

```
state_estimator.update([2, 3], comp_enc_data, r_comp_enc)
```

Donde se realiza la actualización conjunta del comportamiento estimado de los sensores en sus distintos estados. A partir de estos ajustes se puede observar el gráfico presente en la figura 3.

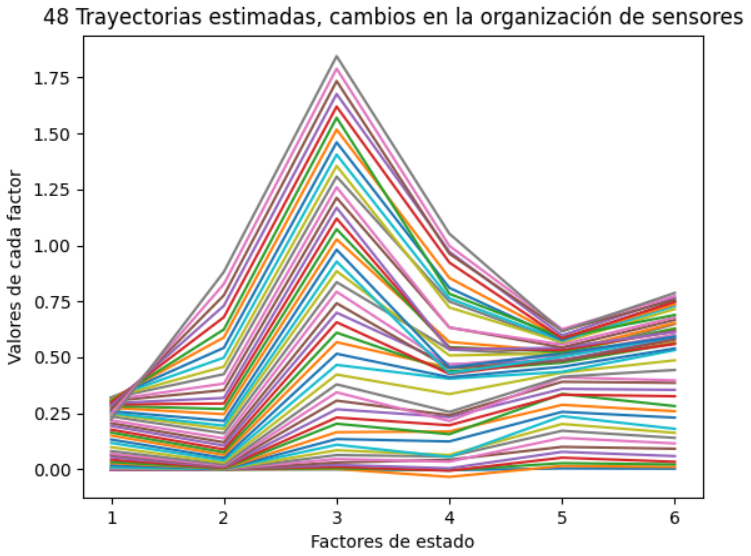

Figura 3: Trayectoria estimada del robot, sensores compass y encoder trabajando juntos.

Tras analizar el gráfico de la figura 3 y compararlo con los 2 anteriores se puede notar un claro error en el funcionamiento del UKF. Es posible que dicho error en el comportamiento de la trayectoria sea debido a utilizar conjuntamente sensores que se pensaron para utilizarse por separado, como es el caso del *compass* y el *encoder*, ya que ambos registran valores de distintas magnitudes físicas, las cuales no deben mezclarse para obtener una correcta estimación de la trayectoria del robot. Cabe destacar sin embargo que el algoritmo entregado mantiene un comportamiento cercano a lo esperado al menos en términos generales, es decir que el error aquí expuesto puede llegar a ser fácilmente detectable y corregible.

5. Comparación de trayectorias, cambios en el ruido del modelo

Siguiendo la misma lógica del primer caso, se tiene el archivo *tarea2d.py* el cual solo fue modificado de modo de poder graficar los datos necesarios, además de modificar los valores presentes en el ruido del modelo y en el ruido de los sensores. Todos los valores pertinentes existentes en el sistema fueron duplicados, como puede observarse en las siguientes líneas de código.

```
q = np.eye(6)
q[0][0] = 0.0002
q[1][1] = 0.0002
q[2][2] = 0.0008
q[3][3] = 0.005
q[4][4] = 0.005
q[5][5] = 0.005

r_imu = np.zeros([2, 2])
r_imu[0][0] = 0.02
r_imu[1][1] = 0.06
r_compass = np.zeros([1, 1])
r_compass[0][0] = 0.04
r_encoder = np.zeros([1, 1])
r_encoder[0][0] = 0.002

state_estimator = UKF(6, q, np.zeros(6), 0.0002*np.eye(6), 0.04, 0.0, 2.0, iterate_x)
```

```
compass_hdg = row[9]
compass_data = np.array([2*compass_hdg])

encoder_vel = row[10]
encoder_data = np.array([2*encoder_vel])

imu_accel = row[7]
imu_yaw_rate = row[8]
imu_data = np.array([2*imu_yaw_rate, 2*imu_accel])
```

Estos cambios generan una trayectoria que mantiene cierta similitud con el comportamiento base original observable en la figura 1 pero es notoriamente más dispersa y errática, como puede apreciarse en la figura 4 presente a continuación.

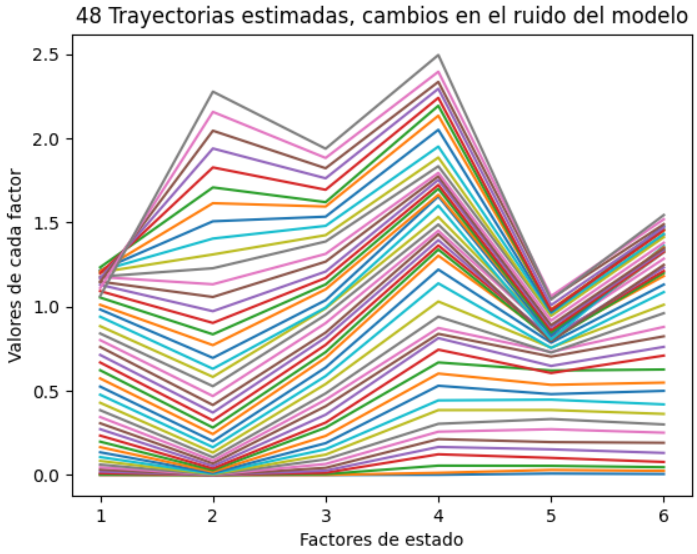

Figura 4: Trayectoria estimada del robot, valores relacionados al ruido duplicados.

Las diferencias observables entre esta última figura y las anteriores son explicados claramente por el aumento del ruido del sistema, este aumento provoca muestras o registros de valores poco claros y poco certeros, los cuales "contaminan" el resultado final de la estimación de la trayectoria del robot.

6. Conclusiones

Tras haber realizado todos los ejercicios propuestos en el enunciado por medio de investigación y trabajo en el código entregado, se ha logrado una mayor comprensión del fenómeno estudiado en cátedra. Sumado a esto, cabe destacar la ventaja para el estudio del curso que conlleva el haber obtenido distintas representaciones visuales dentro de un contexto de aplicación práctica de los filtros de Kalman, especialmente los UKF, logrando así un entendimiento más profundo sobre los mecanismos y algoritmos con los que se puede trabajar en el ámbito de la predicción en Inteligencia Artificial.

7. Bibliografia

- Material de la cátedra del curso.
- https://en.wikipedia.org/wiki/Kalman_filter
- https://en.wikipedia.org/wiki/Extended_Kalman_filter