Simulación de robot bípedo

Daniela Blanco
José Manuel Marrón Ruiz
Jorge Alfredo Delgado Meraz

18 de Mayo de 2015

1 Introducción

El proyecto consiste en simular el comportamiento de un robot bípedo autobalanceable utilizando librerías de cómputo numérico de Python, como numpy y matplotlib. Se contemplaron tres etapas en el desarrollo del proyecto:

1. Evolución del sistema dinámico sin aplicación de control
2. Evolución del sistema dinámico con aplicación del control (PID)
3. Evolución del sistema dinámico con aplicación del filtro de Kalman y el control (PID)

Debido a la naturaleza del proyecto, no obtuvimos directamente las ecuaciones dinámicas del sistema, sino que utilizamos las ecuaciones obtenidas por Ooi[1]. De la misma forma, utilizamos las mismas condiciones físicas y eléctricas del robot utilizado en la tesis previamente citada.
El objetivo del proyecto es observar el comportamiento esperado del robot bajo diferentes condiciones iniciales y bajo diferentes simulaciones, con el propósito de observar la diferencias que presentan y extraer conclusiones sobre el movimiento y la dinámica de un robot bípedo.

2 Marco teórico

2.1 Métodos de Runge-Kutta

Consiste en un conjunto de métodos genéricos iterativos, explícitos e implícitos que sirven para resolver ecuaciones diferenciales. Tiene el error local de truncamiento del mismo orden que los métodos de Taylor, pero tiene la ventaja que prescinden del cálculo y evaluación de las derivadas de la función f(t,y). Se define como el orden del método el numero n, variable que representa la cantidad de términos utilizados. En nuestro proyecto implementamos orden 4. Se usan las fórmulas expresadas en la siguiente imagen:

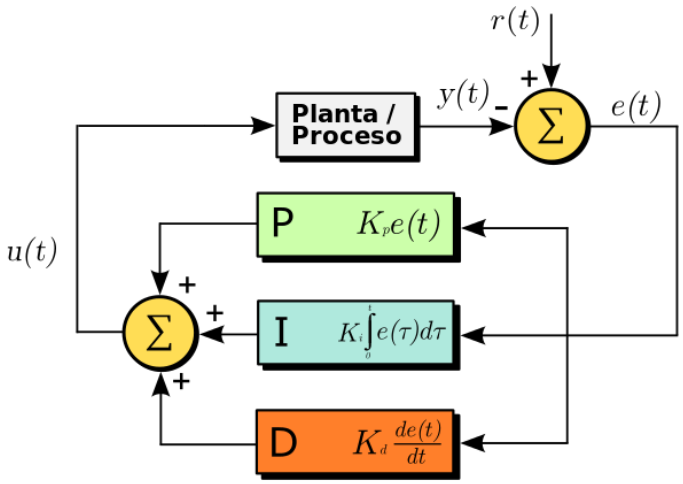
$$k_1 = h\, f(t_i, y_i)$$
$$k_2 = h\, f\left(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_1\right)$$
$$k_3 = h\, f\left(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_2\right)$$
$$k_2 = h\, f(t_i + h, y_i + k_3)$$
$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

2.2 Controlador

El sistema de control en el desarrollo auto-balanceable es muy importante para que funcione. Hay muchos sistemas de control pero nosotros decidimos utilizar el controlador PID.
El PID es un mecanismo de control por retroalimentación que funciona calculando el error entre un valor medio y un valor deseado. Es muy usado en sistemas de control industrial. Consiste en tres parámetros :

1. **Proporcional**: Depende del error actual. Produce una señal de control proporcional al error de la salida del proceso respecto a la posición que se desea.
2. **Integral**: Depende de los errores pasados. Produce una señal que proporciona una corrección para poder compensar las perturbaciones y mantener el proceso controlado en el punto que se desea.
3. **Derivativo**: Predice errores futuros y de esta manera anticipa el efecto de la acción proporcional para poder estabilizar el proceso mas rápidamente después de alguna perturbación.

Al sumar estos tres parámetros se obtiene el ajuste deseado al proceso. En el caso de nuestro proyecto en la etapa en la que aplicamos el controlador lo que logramos fue que la posición y la velocidad del robot bípedo se estabilizara, y en el caso del chasis que se balanceara para evitar que el robot se cayera.
La siguiente imagen muestra la manera en la que el PID funciona y la ecuación:



$$u(t) = K_p[e(t) + \frac{1}{T_i}\int e(t)dt + T_d\frac{d}{dt}e(t)] \tag{1}$$

2.3 Filtro de Kalman

El filtro de Kalman es un algoritmo que fue desarrollado por Rudolf E. Kalman en 1960. Sirve para poder estimar el estado no medible de un sistema dinámico lineal a partir de mediciones de ruido blanco (señal aleatoria que se caracteriza porque sus valores de señal en dos instantes de tiempo distintos no tienen correlación).
Este algoritmo es un método recursivo que estima el estado lineal mediante la minimización de la media del error cuadrático. Al conocer el comportamiento actual del proceso al que se aplica (en nuestro caso el robot bípedo), el filtro estima el estado futuro del sistema sumando un termino de corrección proporcional al factor que se predijo.
Las ecuaciones que representan el sistema son:

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \tag{2}$$

$$z_k = H_{k-1}x_k + v_k \tag{3}$$

en donde,

- A y B son las mismas matrices de la dinámica del robot
- x_k es la matriz de estados (al igual que X_{k-1} solo que un estado adelante)
- U_{k-1} es generada por el controlador
- H_k es una matriz de 2X4 con unos en la posición (1,1) y (2,2) y ceros en las demás posiciones la cual indica la relación entre mediciones y el vector de estado al momento k

- v_k es ruido blanco generado por la función de numpy, randn
- Z_k es el vector de posiciones al momento k.

2.3.1 Algoritmo del filtro de Kalman

Predicción

$$x_{k|k-1} = \Phi_k x_{k-1|k-1} + B_k u_k \tag{4}$$

$$P_{k|k-1} = \Phi_k P_{k-1|k-1} \Phi_k^T + Q_k \tag{5}$$

Actualización

$$y_k = z_k - H_k x_{k|k-1} \tag{6}$$

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{7}$$

$$x_{k|k} = x_{k|k-1} + K_k y_k \tag{8}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{9}$$

donde,

- Φ_k : Matriz de transición de estados
- $x_{k—k\text{-}1}$: Estimación futura del vector de estados
- $P_{k—k\text{-}1}$: Covarianza del error asociada a la estimación futura
- Q_k : La matriz de covarianza de ruido de la observación
- R_k : La matriz de covarianza de ruido de las mediciones
- K_k : Ganancia de Kalman

3 Modelado del sistema

La dinámica del robot debe ser descrita por un modelo matemático para facilitar el desarrollo y eficiencia del sistema de control de balance angular. Para la implementación física del robot, debemos considerar dos modelos matemáticos: un péndulo invertido y un modelo lineal para un motor DC. El modelo del motor requiere de conocimientos de electrónica y circuitería, por lo que excede los propósitos del presente proyecto y nos concentraremos en la formulación teórica del auto-balance y su modelado matemático.

3.1 Modelo dinámico para un péndulo invertido de dos ruedas

Primeramente analizaremos el péndulo y la dinámica de las ruedas por separado, y al final combinaremos ambos modelos.

3.1.1 Ruedas

Como las ecuaciones de las ruedas son análogas entre sí, sólo mostraremos el desarrollo para la rueda derecha. Por la segunda ley de Newton, la suma de las fuerzas horizontales en x es: $\sum Fx = Ma$

$$M_\omega \ddot{x} = H_{fR} - H_R \tag{10}$$

La suma de fuerzas alrededor de la rueda nos da: $\sum M_o = I\alpha$

$$I_w \ddot{\theta}_w = C_R - H_{_{fR}} r \tag{11}$$

Acomodando los términos y substituyendo los parámetros del motor DC, la ecuación anterior se convierte en:

$$I_w \ddot{\theta}_w = \frac{-k_m k_e}{R}\dot{\theta}_w + \frac{k_m}{R}V_a - H_{_{fR}} r \tag{12}$$

Para la rueda derecha

$$\Rightarrow M_w \ddot{x} = \frac{-k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_L \tag{13}$$

Para la rueda izquierda

$$\Rightarrow M_w \ddot{x} = \frac{-k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_R \tag{14}$$

Tomando en cuenta las transformaciones:

$\ddot{\theta}_w r = \ddot{x} \Rightarrow \ddot{\theta}_w = \frac{\ddot{x}}{r}$

$\dot{\theta}_w r = \dot{x} \Rightarrow \dot{\theta}_w = \frac{\dot{x}}{r}$

Y sumando las ecuaciones de las ruedas, obtenemos finalmente:

$$2(M_w + \frac{I_w}{r^2})\ddot{x} = \frac{-2k_m k_e}{Rr^2}\dot{x} + \frac{2k_m}{Rr}V_a - (H_L + H_R) \tag{15}$$

3.1.2 Péndulo Invertido

A continuación, mostramos cómo se modela el chasis del robot, lo cual se puede realizar como un péndulo invertido. De nuevo, por la segunda Ley de Newton, la suma de fuerzas *horizontales* es:

$\sum F_x = M_p\ddot{x}$

$$(H_L + H_R) - M_p l\ddot{\theta}_p + M_p l\dot{\theta}_p^2 \sin(\theta_p) = M_p\ddot{x} \tag{16}$$

La suma de fuerzas *perpendiculares* al péndulo son:

$\sum F_{xp} = M_p\ddot{x}\cos(\theta_p)$

$$(H_L+H_R)\cos(\theta_p)+(P_L+P_R)\sin(\theta_p)-M_p g\sin(\theta_p)-M_p l\ddot{\theta_p} = M_p\ddot{x}\cos(\theta_p) \tag{17}$$

La suma de *momentos* alrededor del centro de masa del péndulo:

$\sum M_o = I\alpha$, es decir

$$-(H_L + H_R)l\cos(\theta_p) - (P_L + P_R)l\sin(\theta_p) - (C_L + C_R) = I_p\ddot{\theta}_p \tag{18}$$

Definiendo el torque aplicado al péndulo y después de una transformación lineal tenemos:

$$C_L + C_R = \frac{-2k_m k_e}{R}\frac{\dot{x}}{r} + \frac{2k_m}{R}V_a \tag{19}$$

Substituyendo en la ecuación (9) anterior y re-ordenando:

$$-(H_L + H_R)l\cos\theta_p - (P_L + P_R)l\sin\theta_p - (\frac{-2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R}V_a) = I_p\ddot{\theta}_p \tag{20}$$

De esta forma obtenemos las dos siguientes ecuaciones:

$$(I_p + M_p l^2)\ddot{\theta}_p - \frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R}V_a + M_p g l\sin(\theta_p) = -M_p l\ddot{x}\cos(\theta_p) \tag{21}$$

$$\frac{2k_m}{Rr}V_a = (2M_w + \frac{2I_w}{r^2} + M_p)\ddot{x} + \frac{2k_m k_e}{Rr^2}\dot{x} + M_p l\ddot{\theta_p}\cos(\theta_p) - M_p I\dot{\theta}_p^2\sin(\theta_p) \tag{22}$$

4.2 Obtención del Sistema

Linealizando las ecuaciones (12) y (13), asumiendo que $\theta_p = \pi + \phi$, donde ϕ representa el ángulo vertical en dirección ascendente y re-ordenando dichas ecuaciones, obtenemos:

$$\ddot{\phi} = \frac{M_p l}{I_p + M_p l^2}\ddot{x} + \frac{2k_m k_e}{Rr(I_p + M_p l^2)}\dot{x} - \frac{2k_m}{R(I_p + M_p l^2)}V_a + \frac{M_p g l}{I_p + M_p l^2}\theta \quad (23)$$

$$\ddot{x} = \frac{2k_m}{Rr(2M_w + \frac{2I_w}{r^2} + M_p)}V_a - \frac{2k_m k_e}{Rr^2(2M_w + \frac{2I_w}{r^2} + M_p)}\dot{x} + \frac{M_p l}{2M_w + \frac{2I_w}{r^2} + M_p}\ddot{\phi} \quad (24)$$

Después de sustituir las ecuaciones y de una serie de manipulaciones algebráicas, obtenemos el siguiente *sistema*:

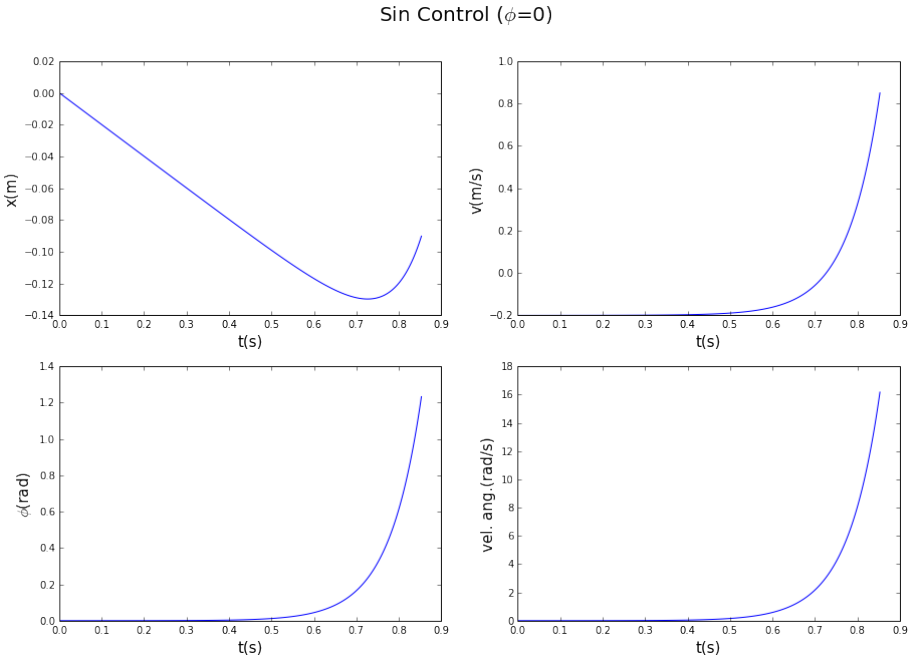
$$\begin{pmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{2k_m k_2(M_p lr - I_p - M_p l^2)}{Rr^2\alpha} & \frac{M_p^2 g l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2k_m k_e(r\beta - M_p l)}{Rr^2\alpha} & \frac{M_p g l \beta}{\alpha} & 0 \end{pmatrix} \begin{pmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{2k_m(I_p + M_p l^2 - M_p lr)}{Rr\alpha} \\ 0 \\ \frac{2k_m(M_p l) - r\beta}{Rr\alpha} \end{pmatrix} V_a \quad (25)$$

donde: $\beta = (2M_w + \frac{2I_w}{r^2} + M_p), \qquad \alpha = (I_p\beta + 2M_p l^2[M_w + \frac{I_w}{r^2}])$

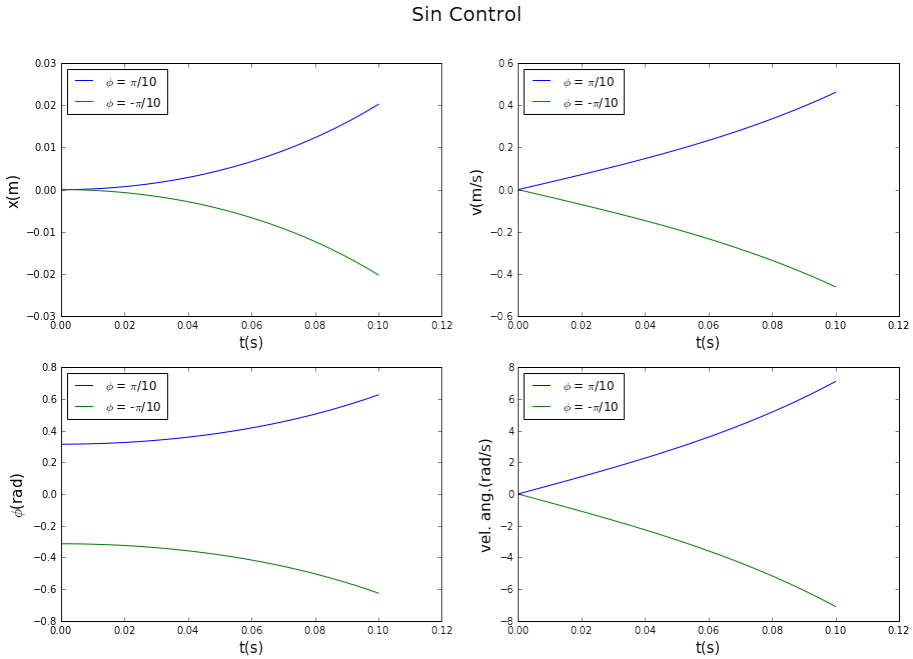
En este modelo se asume que las ruedas del robot siempre estarán en contacto con la tierra (suelo) y que no hay "deslizamiento" (slip) de las llantas (por ejemplo, suelo mojado).

4 Resultados

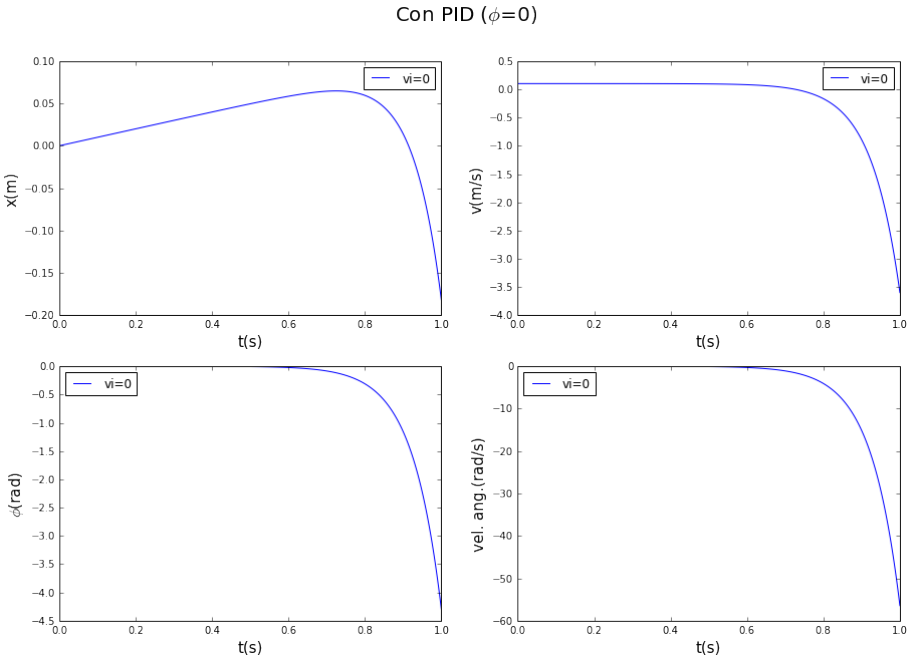
Tras hacer simulaciones al evolucionar el sistema sin ningún tipo de control externo, se obtuvo la gráfica siguiente. Esta gráfica se obtuvo a partir de las condiciones iniciales [0,1,0,0], es decir, el robot vertical pero con una velocidad inicial en x de 1 m/s. Se puede observar como el robot se mueve en un principio en la dirección de la velocidad y a una tasa casi constante. Sin embargo, en cuanto llega a un punto inestable, el robot trata de re-estabilizarse modificando su dirección de movimiento diametralmente, pero no logra la estabilización.



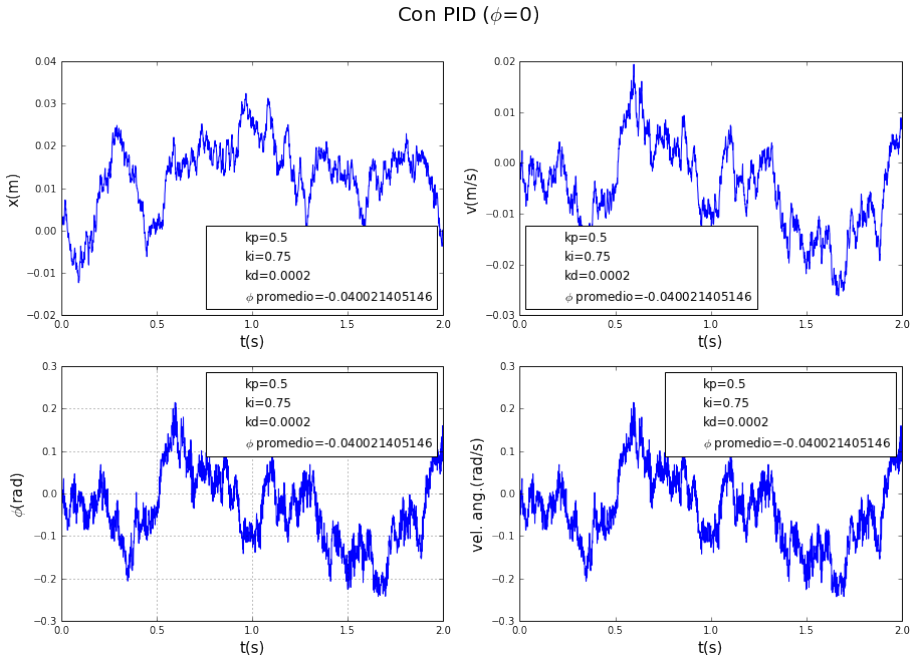
La gráfica a continuación muestra el comportamiento del robot bajo diferentes ángulos iniciales (en este caso de +-π/10). Se logra observar en las gráficas como al no tener una velocidad en X, el robot únicamente cae, modificando la posición debido a esta caida.



La siguiente gráfica nos provee el comportamiento del sistema tras aplicar un control al sistema. El sistema está constantemente midiendo su orientación mediante sensores (cuyo comportamiento simplificamos obteniendo el ángulo en cada momento del tiempo y agregando una componente de ruido con una distribución normal). Se puede observar como el sistema, a pesar de permanecer estable durante más tiempo, finalmente termina por desestabilizarse. Esto nos muestra la dificultad de controlar este tipo de sistemas mediante métodos tradicionales, como es el controlador PID.



Finalmente, la siguiente gráfica nos da información sobre como evoluciona el sistema utilizando un filtro de Kalman en conjunto con el controlador PID. Se puede observar como en este caso el sistema sí logra estabilizarse al menos durante la duración de la simulación. Esto se debe a que el filtro de Kalman, siendo un método de predicción estocástica, logra adivinar con bastante precisión el movimiento del robot y por lo tanto, logra pasar al controlador una señal mejor procesada. Así, podemos ver como el sistema se mantiene dentro de límites bastante correctos, pues el robot permanece con un ángulo cercano a la vertical durante toda la simulación.



Tras la simulación con el filtro de Kalman, procedimos a buscar, mediante prueba y error, valores para K_p,K_i y K_d que lograran que el sistema permaneciera estable. Tras varias pruebas y simulaciones, terminamos con los valores de 0.5, 0.75 y 0.00002 respectivamente, que son los valores que se ven en la gráfica. Es decir, hubo que hacer una investigación exhaustiva de estas constantes antes de poder llegar a un punto estable del sistema.
Habiendo encontrado estas constantes, nos dimos a la tarea de medir la eficiencia de las mismas, con el afán de ver si realmente en todos los casos estas constantes eran suficientemente buenas para el robot que simulamos. Simulamos el robot con el filtro de Kalman mil veces, utilizando los valores obtenidos, y los resultados obtenidos fueron los que se muestran en la siguiente tabla:

Ángulo medio (radianes)	Varianza media (radianes)
-0.00180966930094	0.0244610975524

Es decir, el ángulo medio obtenido se acerca mucho a los 0 grados, mientras que la varianza media de las mil simulaciones fue de un poco menos que 1.5 grados. Es decir, los valores obtenidos logran una eficiencia bastante buena dadas las condiciones del robot.

5 Conclusiones

Después de las simulaciones ejecutadas es necesario decir que el robot bípedo es un sistema que se vuelve inestable muy rápidamente. Las simulaciones sin

ningún tipo de control tardan milisegundos en llegar a un punto inestable. Es por ello que la teoría de control es de vital importancia para el estudio de este tipo de proyectos. Por otra parte, notamos a partir de las simulaciones hechas con el controlador PID (sin Filtro de Kalman), que a pesar de realizar varias simulaciones con diferentes valores de K_p,K_i y K_d, después de un tiempo no demasiado largo, el sistema entraba a un estado inestable. Por esta razón, cuestionamos en este trabajo la eficiencia de un controlador PID para lograr la estabilización de un robot bípedo. Las simulaciones hechas evolucionando el sistema utilizando el filtro de Kalman brindaron resultados más positivos. El controlador PID, junto con el filtro, mantenían el sistema en un estado estable tras segundos de simulación. El desarrollo de esta simulación implicó la gran dificultad de no poder comprobar los resultados de las simulaciones realizadas empíricamente ya que el proyecto involucraba la construcción del robot. Para solventar esta problemática recurrimos a resultados experimentales obtenidos por Ooi[1] y Adeel(et. al)[2].

References

[1] Rich Chi Ooi, *Balancing a Two-Wheeled Autonomous Robot*, The University of Western Australia, School of Mechanical Engineering, 2003.

[2] Umar Adeel, K.S.Alimgeer, et. al *Autonomous Dual Wheel Self Balancing Robot Based on Microcontroller*, COMSATS Institute of Information Technology, Pakistan, 2013.