



Universidad  
Carlos III de Madrid

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

**TESIS DE MÁSTER**

**MODELADO DINÁMICO DE ROBOTS HUMANOIDES  
MEDIANTE ÁLGEBRA ESPACIAL**

Autor: Miguel González-Fierro Palacios

Director: Paolo Pierro y Carlos Balaguer Bernaldo de Quirós

MÁSTER OFICIAL EN  
ROBÓTICA Y AUTOMATIZACIÓN

LEGANÉS, MADRID  
DICIEMBRE 2009



UNIVERSIDAD CARLOS III DE MADRID  
MASTER OFICIAL EN ROBÓTICA Y AUTOMATIZACIÓN

El tribunal aprueba la tesis de Master titulada  
**"MODELADO DINÁMICO DE ROBOTS HUMANOIDES  
MEDIANTE ÁLGEBRA ESPACIAL "** realizado por  
**Miguel González-Fierro Palacios.**

Fecha: Diciembre 2009

Tribunal: \_\_\_\_\_  
Dr. Alberto Jardón Huete

\_\_\_\_\_  
Dra. Concha Monje Micharet

\_\_\_\_\_  
Dr. Higinio Rubio Alonso



*A mi padre*



# Índice general

<b>Índice de Figuras</b>	<b>IX</b>
<b>Agradecimientos</b>	<b>xv</b>
<b>Resumen</b>	<b>xvii</b>
<b>Abstract</b>	<b>xix</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Motivación y origen de la tesis . . . . .	1
1.2. Objetivos . . . . .	2
1.3. Organización del documento . . . . .	3
<b>2. Estudio dinámico del robot humanoide</b>	<b>5</b>
2.1. Introducción . . . . .	5
2.2. Dinámica de un sistema de cuerpos rígidos . . . . .	7
2.2.1. Ecuación del movimiento . . . . .	8
2.2.2. Clasificación de las restricciones . . . . .	9
2.2.3. Conectividad . . . . .	12
2.3. Formulación clásica de la dinámica . . . . .	13
2.3.1. Estado del arte . . . . .	13

2.3.2. Formulación de Lagrange . . . . .	15
2.3.3. Formulación de Newton-Euler . . . . .	16
<b>3. Notación espacial de la dinámica</b>	<b>19</b>
3.1. Introducción . . . . .	19
3.2. Álgebra de vectores espaciales . . . . .	20
3.2.1. Velocidad espacial . . . . .	20
3.2.2. Fuerza espacial . . . . .	23
3.2.3. Producto escalar . . . . .	24
3.2.4. Transformación de coordenadas . . . . .	25
3.2.5. Operador producto cruzado espacial . . . . .	28
3.2.6. Diferenciación . . . . .	30
3.2.7. Aceleración . . . . .	31
3.2.8. Momento . . . . .	34
3.2.9. Inercia . . . . .	35
3.2.10. Ecuación de movimiento . . . . .	36
3.3. Formulación espacial de la dinámica de robots humanoides . . . . .	37
3.3.1. Dinámica inversa: Recursive Newton-Euler Algorithm . . . . .	37
3.3.2. Dinámica directa: Composite Rigid Body Algorithm . . . . .	40
3.3.3. Dinámica directa: Articulated Body Algorithm . . . . .	43
<b>4. Locomoción de un robot humanoide</b>	<b>47</b>
4.1. Introducción . . . . .	47
4.2. Análisis de la locomoción bípeda . . . . .	48
4.3. Criterio de estabilidad . . . . .	50
4.4. Modelo del péndulo invertido . . . . .	53
4.5. Control de estabilidad . . . . .	56
<b>5. Modelo dinámico del robot humanoide HOAP-3</b>	<b>59</b>
5.1. Introducción . . . . .	59
5.2. El robot humanoide HOAP-3 . . . . .	59

5.3. Herramienta de simulación . . . . .	65
5.4. Librerías dinámicas . . . . .	68
5.5. Modelo 3D del robot . . . . .	70
5.5.1. Modelo del torso del robot . . . . .	71
5.5.2. Modelo del cuerpo completo del robot . . . . .	75
5.5.3. Modelo de doble péndulo invertido . . . . .	80
<b>6. Resultados</b>	<b>81</b>
6.1. Introducción . . . . .	81
6.2. Estudio dinámico del torso del robot . . . . .	82
6.3. Estudio dinámico del cuerpo completo del robot . . . . .	90
6.4. Estudio dinámico del doble péndulo invertido . . . . .	96
6.5. Tiempos de ejecución . . . . .	98
<b>7. Conclusiones y trabajos futuros</b>	<b>101</b>
7.1. Conclusiones . . . . .	101
7.2. Trabajos futuros . . . . .	103
Referencias . . . . .	105



# Índice de figuras

2.1.	Clasificación de restricciones cinemáticas . . . . .	9
2.2.	Ejemplos de restricciones cinemáticas . . . . .	11
2.3.	Sistema de cuerpos fijados a una base y grafo . . . . .	13
3.1.	Velocidad de un sólido rígido . . . . .	21
3.2.	Coordenadas de Plücker para la velocidad espacial . . . . .	22
3.3.	Fuerza actuando en un sólido rígido (a) y bases de Plücker para las coordenadas de fuerza (b) . . . . .	23
3.4.	Funciones de rotación y transformadas de Plücker . . . . .	27
3.5.	Propiedades del producto cruzado euclídeo . . . . .	28
3.6.	Producto cruzado espacial . . . . .	29
3.7.	Propiedades del producto cruzado espacial . . . . .	30
3.8.	Aceleración de un cuerpo rotando uniformemente . . . . .	31
3.9.	Esquema de la demostración de la fórmula de la velocidad espacial	32
3.10.	Momento espacial . . . . .	34
3.11.	Fuerzas actuando en el eslabón i . . . . .	39
3.12.	Algoritmo RNEA y sus ecuaciones . . . . .	40
3.13.	Términos no nulos de la matriz H dependiendo de la conectividad de la cadena cinemática. . . . .	43

3.14. Comparación de la fuerza en un cuerpo rígido (a) con la de un cuerpo articulado (b) . . . . .	44
3.15. Definición de cuerpos articulados en ABA (a) y solución de la aceleración para la articulación 1 (b) . . . . .	45
4.1. Ciclo del movimiento . . . . .	49
4.2. Estabilidad de la locomoción bípeda . . . . .	50
4.3. Fuerzas que actúan en el pie de un humanoide . . . . .	51
4.4. Compensación del ZMP . . . . .	52
4.5. Región de estabilidad a) en soporte simple, b) en doble soporte . .	53
4.6. Peñdulo invertido con restricción en el eje Z . . . . .	54
4.7. Control de estabilidad de un robot humanoide . . . . .	55
4.8. Doble péndulo invertido . . . . .	57
5.1. Robot humanoide HOAP-3 . . . . .	60
5.2. Grados de libertad del robot HOAP-3 . . . . .	61
5.3. Articulaciones del HOAP y motor al que hace referencia . . . .	62
5.4. PC embebido . . . . .	63
5.5. Dimensiones y sensores del HOAP-3 . . . . .	64
5.6. Ejemplo de Open Inventor . . . . .	66
5.7. Parámetros de Denavit-Hartenberg . . . . .	67
5.8. Modelo 3D del HOAP con las articulaciones a cero. . . . .	68
5.9. Sistema de referencia del mundo en OpenInventor . . . . .	72
5.10. Simulación del torso del robot . . . . .	75
5.11. Longitudes de los eslabones del HOAP . . . . .	77
5.12. Simulación del cuerpo completo del HOAP . . . . .	80
5.13. Simulación del doble péndulo invertido . . . . .	80
6.1. Trayectoria 1 en el modelo del torso . . . . .	82
6.2. Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	83

6.3. Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	84
6.4. Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	86
6.5. Trayectoria 2 en el modelo del torso . . . . .	86
6.6. Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	87
6.7. Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	88
6.8. Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	89
6.9. Trayectoria circular con el modelo completo del robot . . . . .	90
6.10. Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	91
6.11. Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	92
6.12. Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo) . . . . .	93
6.13. Locomoción bípeda, robot real y simulado . . . . .	94
6.14. Pares en la pierna en vuelo (a) y en la pierna de apoyo . . . . .	95
6.15. Estructura generalizada de un robot humanoide . . . . .	96
6.16. Resultante de las fuerzas producidas por un paso. Arriba el modelo completo y abajo el modelo del doble péndulo . . . . .	97
6.17. Tiempos de ejecución de los algoritmos dinámicos . . . . .	99



# Agradecimientos

Esta tesis ha supuesto un auténtico reto para mí debido a la complejidad de la materia y a otros motivos. Quisiera agradecer a toda la gente que me ha ayudado y apoyado en todo este tiempo.

En primer lugar quiero dar gracias a Berta por apoyarme siempre y estar a mi lado en todos los momentos. A mi madre por sus sabios consejos que me han acompañado a lo largo de toda mi vida. A mi hermana, abuelos y demás familiares por su ayuda siempre que les he necesitado.

Al profesor Carlos Balaguer, por haber confiado en mí y ofrecerme la oportunidad de trabajar en lo que más me apasiona. A Paolo Pierro por su ayuda y buenos consejos. A todas las personas que me han ayudado en el RoboticsLab, Javi, Víctor, Juan, César, Dani, etc y por los buenos momentos que hemos pasado.

Este trabajo va dedicado especialmente a mi padre, Félix González-Fierro, un hombre que me ha demostrado que los grandes problemas de la vida se pueden superar si se lucha por las personas que se quiere y por uno mismo. Es por ello que le quiero y le admiro.

Gracias a todos. Un abrazo.

Miguel



# Resumen

La dinámica es una parte fundamental, no sólo para la locomoción de un robot humanoide, sino también en cualquier tipo de movimiento, que relaciona las fuerzas y aceleraciones que aparecen en el sistema. Se utiliza en generación de trayectorias, control de estabilidad, diseño mecánico y simulación.

Debido al alto coste computacional que conlleva obtener la dinámica de un sistema complejo de muchos grados de libertad, como el de un humanoide, se han desarrollado distintos modelos simplificados, que permiten calcular la dinámica en tiempo real, a cambio de perder exactitud en los cálculos.

La obtención de un modelo dinámico completo del robot, permite conocer con exactitud todas las fuerzas y pares que aparecen en cada uno los eslabones y las articulaciones del robot en cualquier tipo de movimiento. Gracias a ello, se pueden validar y desarrollar nuevos modelos simplificados de locomoción en tiempo real o calcular las fuerzas que aparecen al realizar un movimiento complejo, como por ejemplo un baile.

En este trabajo, se ha obtenido un modelo dinámico completo del robot humanoide HOAP-3 y se han realizado pruebas introduciendo diferentes trayectorias, para finalmente validar y comentar los resultados.



# Abstract

Dynamics is a fundamental part for a humanoid robot, not only for locomotion, but also for any kind of movement. It relates the forces and accelerations that appear in the system. It is used in trajectory generation, stability control, mechanical design and simulation.

Due to the high computational cost of obtaining the dynamics of a system with many degrees of freedom, several simplified models have been developed. These models allow to solve the dynamical problem in real time in despite of losing accuracy in calculations.

A complete dynamic model of the robot involves an exact knowledge of all the forces and torques that appear in every link and joint of the robot. Therefore, new simplified models of real time locomotion and other kinds of movements, like dancing, can be studied and developed.

In this work, a complete dynamic model of the robot has been obtained, and different proofs have been achieved by testing movement trajectories. Finally, results have been validated and discussed.



# Capítulo 1

## Introducción

### 1.1. Motivación y origen de la tesis

En los últimos años se ha dado un fuerte impulso a la investigación de robots humanoides, desarrollándose robots con alta capacidad de manipulación, interacción y movilidad. Algunos ejemplos son el WABIAN-2 de la Universidad de Waseda (Ogura y cols., 2006), el ASIMO de Honda (Sakagami y cols., 2002), el HRP-2 del National Institute of Advanced Industrial Science and Technology of Japan (Kaneko y cols., 2004) o el RH-1 diseñado en la Universidad Carlos III de Madrid (Arbulú, Kaynov, Cabas, y Balaguer, 2009). Pero para que estos robots interactúen con el ser humano y se introduzcan en su entorno, es necesario aumentar su seguridad, manipulabilidad, estabilidad y su capacidad de interacción.

El estudio de las fuerzas y momentos que se producen en el robot es de vital importancia si se quieren alcanzar estos objetivos. El problema de la dinámica en un robot humanoide es un tema de interés desde hace varias décadas y juega un papel de gran relevancia en el control del movimiento del robot.

La dinámica se define mediante la ecuación del movimiento, que relaciona los pares en las articulaciones con las aceleraciones. Mediante la dinámica directa se obtienen las aceleraciones partiendo de los pares y se suele utilizar en simulación. La dinámica inversa realiza la operación contraria, permite obtener los pares articulares partiendo de las aceleraciones. Ésta se suele utilizar para generación de trayectorias, control de la estabilidad y diseño mecánico del robot.

La idea de esta tesis nace de la necesidad de obtener un modelo dinámico completo del humanoide HOAP-3. Partiendo de este modelo, programado offline en Matlab, que permite obtener los pares producidos en todas las articulaciones del robot, se pueden desarrollar algoritmos de control de movimiento y de control de la estabilidad desarrollando modelos reducidos que puedan ser aplicados en tiempo real. Asimismo, el modelo dinámico completo constituye una herramienta muy útil a la hora de validar estos algoritmos.

Para la obtención de la dinámica del robot se ha utilizado la notación de vectores espaciales de 6D, que se diferencia de la clásica notación de 3D en una mayor rapidez de computación y una mayor claridad en los algoritmos matemáticos, lo que proporcionan una notación concisa para describir velocidad, aceleración, inercia y fuerza.

## 1.2. Objetivos

Se van a enumerar los objetivos principales de esta tesis de máster:

1. Construcción de un modelo del tronco del robot HOAP-3.
2. Construcción de un modelo del cuerpo completo del robot.
3. Construcción de un modelo de un doble péndulo invertido.
4. Implementación de la dinámica inversa y directa en estos modelos.

5. Aplicación de este desarrollo para la validación de un control de estabilidad.
6. Verificación y discusión de los resultados obtenidos.
7. Análisis de los tiempos de ejecución de los algoritmos dinámicos.

### **1.3. Organización del documento**

El documento se va a organizar de la siguiente manera: en el Capítulo 2 se hablará de la dinámica de los robots humanoides prestando atención a las dos teorías más destacadas relacionadas con el tema, la teoría de Lagrange y la de Newton-Euler. En el Capítulo 3 se presentará la notación de vectores espaciales y se definirá su álgebra, para terminar aplicando esta notación para la obtención de la dinámica inversa y directa. Seguidamente, en el Capítulo 4, se desarrollarán las características más importantes de la locomoción bípeda y se hará referencia a un modelo reducido de doble péndulo para controlar la estabilidad del robot. En el Capítulo 5 se hará una breve descripción del robot HOAP-3 y se tratará de los modelos dinámicos construidos mediante estas técnicas. A continuación, en el Capítulo 6, se expondrán y discutirán los resultados obtenidos y por último, en el Capítulo 7 se terminará analizando las conclusiones.



Capítulo **2**

# Estudio dinámico del robot humanoide

## 2.1. Introducción

La dinámica se ocupa de la relación entre las fuerzas que actúan sobre un cuerpo y el movimiento que en él se origina. Por lo tanto, el modelo dinámico de un robot tiene por objeto conocer la relación entre el movimiento del robot y las fuerzas implicadas en el mismo (Barrientos, Peñín, Balaguer, y Aracil, 2007).

Esta relación se obtiene mediante el denominado modelo dinámico, que relaciona matemáticamente:

- La localización del robot, definida por sus variables articulares y sus derivadas: velocidad y aceleración.
- Las fuerzas y pares aplicados en las articulaciones.
- Los parámetros dimensionales y otras propiedades del robot, como longitud, masa o inercias de sus elementos.

La obtención de este modelo para mecanismos de uno o dos grados de libertad no es excesivamente compleja, pero a medida que el número de grados de libertad aumenta, el planteamiento y obtención del modelo se complica enormemente. Por este motivo, no siempre es posible obtener un modelo dinámico expresado de una forma cerrada, esto es, mediante una serie de ecuaciones, normalmente del tipo diferencial de segundo orden, cuya integración permita conocer el movimiento que surge al aplicar unas fuerzas o qué fuerzas hay que aplicar para obtener un movimiento determinado.

El modelo dinámico debe ser resuelto entonces de manera iterativa mediante la utilización de un procedimiento numérico. El problema de la obtención del modelo dinámico de un robot es, por lo tanto, uno de los aspectos más complejos de la robótica, lo que ha llevado a ser obviado en numerosas ocasiones, utilizando únicamente la cinemática para programar los movimientos del robot. Sin embargo, el modelo dinámico es imprescindible para conseguir los siguientes fines:

- Simulación del movimiento del robot.
- Diseño y evaluación de la estructura mecánica del robot.
- Dimensionamiento de los actuadores.
- Diseño y evaluación del control dinámico del robot.

Este último fin es, evidentemente, de gran importancia, pues de la calidad del control dinámico del robot depende la precisión y velocidad de sus movimientos.

En la actualidad existen dos enfoques diferentes para abordar este problema (Katic y Vukobratovic, 2003). El primero de ellos asume que el modelo del robot y del entorno es conocido, mediante el conocimiento completo de las masas e

inercias existentes en cada uno de los eslabones del robot se pueden obtener movimientos dinámicamente estables, debido al elevado coste computacional las trayectorias de movimiento se suelen programar off-line (Hirai, Hirose, Haikawa, y Takenaka, 1998; Yamaguchi, Soga, Inoue, y Takanishi, 1999). El segundo enfoque utiliza un conocimiento limitado de la dinámica del sistema, representando el robot como un péndulo invertido o variantes de éste, para conseguir calcular las trayectorias del robot en tiempo real (Kanehiro, Kaneko, Yokoi, Hirukawa, y Kajita, 2001). Este segundo método se apoya sustancialmente en el control realimentado para corregir los errores producidos por la simplificación del sistema. Este trabajo se va a centrar en el primero de los enfoques y se va a tener en cuenta el modelo completo del robot.

Es importante hacer notar que el modelo dinámico completo de un robot debe incluir no solo la dinámica de sus elementos (barras o eslabones), sino también la propia de sus sistemas de transmisión, de los actuadores y sus equipos electrónicos de mando. Estos elementos incorporan al modelo dinámico nuevas inercias, rozamientos, saturaciones de los circuitos electrónicos, etc, aumentando aun más su complejidad. Por último, es preciso señalar que si bien en la mayor parte de las aplicaciones reales de robótica, las cargas e inercias manejadas no son suficientes como para originar deformaciones en los eslabones del robot, en determinadas ocasiones no ocurre así, siendo preciso considerar al robot como un conjunto de eslabones no rígidos. Aplicaciones de este tipo pueden encontrarse en la robótica espacial o en robots de grandes dimensiones.

## 2.2. Dinámica de un sistema de cuerpos rígidos

Un sistema de cuerpos rígidos es un conjunto de sólidos rígidos conectados por articulaciones y sobre el que actúan varias fuerzas. El efecto de una articulación es imponer una restricción de movimiento entre los dos cuerpos que

conecta, algunos movimientos relativos son permitidos en determinadas direcciones pero no en otras. Para ser más precisos, una articulación introduce una restricción de fuerza en el sistema.

### 2.2.1. Ecuación del movimiento

En general, la ecuación de movimiento para un sistema de cuerpos rígidos se expresa de forma canónica, según la notación de (Barrientos y cols., 2007), como

$$D(q)\ddot{q} + H(q, \dot{q}) + C(q) = \tau \quad (2.1)$$

donde  $q$ ,  $\dot{q}$  y  $\ddot{q}$  son los vectores para la posición, velocidad y aceleración generalizados y  $\tau$  es el vector de fuerzas o pares generalizadas.  $D$  es la matriz de inercia generalizada y  $C$  es la matriz de gravedad generalizada; ambas dependen de  $q$ .  $H$  es la matriz de fuerzas de Coriolis y centrípetas y depende de los valores de  $q$  y  $\dot{q}$ . Al mismo tiempo, conviene destacar que el vector de fuerzas generalizadas debe incluir todos los pares que intervienen en el movimiento, es decir

$$\tau = \tau_{motor} - \tau_{perturbador} - \tau_{rozamiento viscoso} - \tau_{rozamiento seco} \quad (2.2)$$

La expresión anterior (Eq. 2.2) es por tanto no lineal, no siendo trivial obtener a partir de ella el modelo dinámico directo, el cual proporciona la trayectoria seguida como consecuencia de la aplicación de unos pares determinados.

Para ser más estrictos, las matrices  $D$ ,  $H$  y  $C$  no dependen exclusivamente de  $q$  y  $\dot{q}$ , sino también del propio sistema de cuerpos rígidos. Por lo tanto, sería más correcto escribir las matrices como  $D(modelo, q)$ ,  $H(modelo, q, \dot{q})$  y  $C(modelo, q)$  donde *modelo* se refiere al conjunto de datos que describe el sistema de cuerpos rígidos en términos de sus componentes, es decir, el número de cuerpos y articulaciones, el tipo de articulación (prismática, rotativa o esférica), la manera en que están conectados, la masa e inercia de los cuerpos, etc.

El objetivo de la dinámica es averiguar el valor de  $\ddot{q}$  partiendo de  $\tau$ , lo que se denomina dinámica directa, u obtener el valor de  $\tau$  partiendo de  $\ddot{q}$ , es decir, resolver la dinámica inversa. Es posible expresar estos algoritmos en función de las variables de las que dependen:

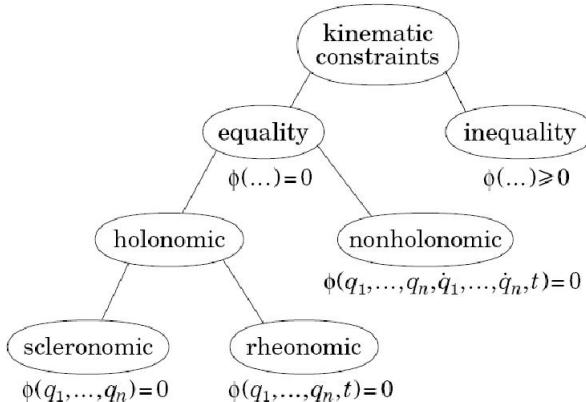
$$\ddot{q} = FD(\text{modelo}, q, \dot{q}, \tau)$$

$$\tau = ID(\text{modelo}, q, \dot{q}, \ddot{q})$$

### 2.2.2. Clasificación de las restricciones

A la hora de definir un sistema de cuerpos rígidos, es absolutamente necesario definir las restricciones a las que está sometido. El profundo conocimiento de éstas permitirá modelar un sistema con fidelidad y exactitud.

Las restricciones cinemáticas pueden ser clasificadas de acuerdo con el tipo de restricción que impone al sistema de cuerpos rígidos. En la Figura 2.1 se muestra una clasificación jerárquica.



**Figura 2.1:** Clasificación de restricciones cinemáticas

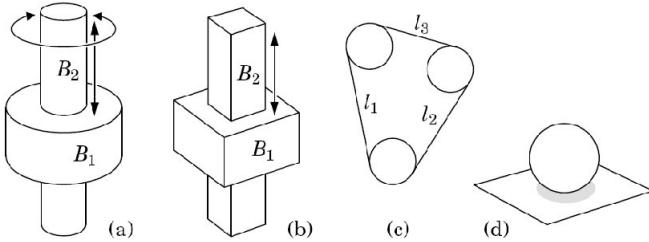
La primera distinción se halla entre restricciones de igualdad o de desigualdad. Las primeras, delimitan un contacto físico permanente entre dos cuerpos y las segundas, aparecen cuando los cuerpos son libres de mantener contacto

o separarse. Las restricciones de desigualdad se usan para describir fenómenos como colisiones o pérdida de contacto.

La siguiente distinción se produce entre restricciones holonómicas y no holonómicas. Las holonómicas son restricciones de la posición y típicamente representan contacto deslizante. Las no holonómicas hacen referencia a las variables de velocidad y especifican contacto giratorio.

Para finalizar, existen las restricciones escleronómicas y reónomicas. Las primeras son función solamente de la posición y las últimas son además función del tiempo.

A modo de ejemplo aclarativo, en la Figura 2.2 se muestran algunos tipos de restricciones. Los diagramas (a) y (b) muestran una articulación cilíndrica y una prismática respectivamente. Ambas representan restricciones escleronómicas. La articulación cilíndrica permite dos grados de libertad, giro y deslizamiento, mientras que la articulación prismática solo permite un grado de libertad de deslizamiento. Si se desea hacer que la restricción cilíndrica dependa del tiempo, entonces, el resultado es una restricción reónica de un grado de libertad. Si se realiza la misma operación con la articulación prismática, se obtiene una restricción reónica de cero grados de libertad.



**Figura 2.2:** Ejemplos de restricciones cinemáticas

Las restricciones cinemáticas generalmente restringen el movimiento entre dos cuerpos, sin embargo, es posible encontrar situaciones en las que se relacionen más de dos cuerpos; este caso es el representado en la Figura 2.2(c). En este ejemplo, se muestra un sistema de cuerpos rígidos en 2D en el cual tres círculos del mismo tamaño están restringidos a moverse dentro del perímetro de una cuerda. Asumiendo que la cuerda está tensa, los tres cuerpos son libres de moverse de forma que satisfagan la ecuación  $l_1 + l_2 + l_3 + 2\pi r = p$ , donde  $p$  y  $r$  es el perímetro de la cuerda y el radio del cuerpo, respectivamente.

La Figura 2.2(d) muestra una esfera descansando sobre un plano. Si la esfera está restringida a rodar sin deslizar, entonces posee tres grados de libertad instantáneos, puede rodar en dos direcciones y girar sobre su punto de contacto. Este sistema constituye una restricción no holonómica.

Es evidente que no es posible para los cuerpos de la Figura 2.2(a) y la Figura 2.2(b) separarse uno de otro. Por el contrario, es posible que la cuerda de la Figura 2.2(c) se afloje o que la esfera de la Figura 2.2(d) pierda el contacto con el suelo. Si se conoce con suficiente exactitud las fuerzas que actúan en el sistema y se asegura que estas acciones no pueden aparecer, se pueden modelar estas restricciones como restricciones de igualdad, sin embargo, en caso contrario, se modelarían como restricciones de desigualdad.

### 2.2.3. Conectividad

Un sistema de cuerpos rígidos se forma a partir de la unión de las partes que lo componen. Las articulaciones son las responsables de las restricciones cinemáticas del sistema, por lo cual se pueden definir como la relación cinemática entre dos o más cuerpos del sistema.

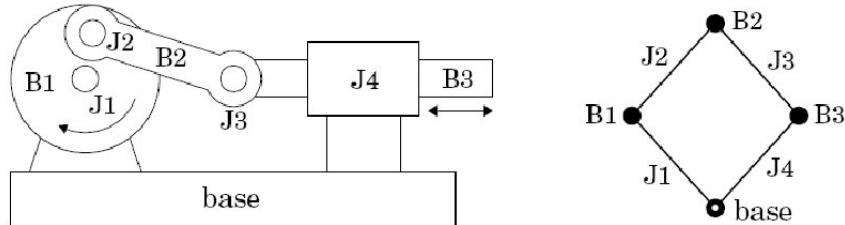
Para obtener la cinemática o dinámica de un sistema complejo como puede ser un robot humanoide, en primer lugar hay que construir un modelo del sistema, teniendo en cuenta las restricciones de cada uno de sus cuerpos, la conectividad entre ellos y otras propiedades físicas. La adecuada construcción de este modelo va a establecer de manera determinante la correcta obtención de los resultados cinemáticos o dinámicos.

Una forma sencilla y comprensible de caracterizar la conectividad existente entre cada una de las partes de un sistema de cuerpos rígidos es construir el grafo de conectividad. El grafo de conectividad debe cumplir las siguientes propiedades:

1. Los nodos representan cuerpos.
2. Los arcos representan articulaciones.
3. Uno de los nodos simboliza la base y el resto representa cuerpos móviles.
4. El grafo está conectado, es decir, siempre existe un arco entre dos cuerpos.

En la Figura 2.3 está representado un sistema móvil fijado a una base. Como se puede observar, se trata de un mecanismo compuesto por tres cuerpos móviles ( $B_1$  a  $B_3$ ), una base fija, tres articulaciones de revolución ( $J_1$  a  $J_3$ ) y una prismática,  $J_4$ . En el grafo de conectividad situado a la derecha, se muestra cada uno de los cuerpos conectados a cada una de las articulaciones. De forma general, en un mecanismo robótico, a los cuerpos se les suele denominar eslabones, y a un

conjunto de eslabones unidos por articulaciones se les suele denominar cadena cinemática.



**Figura 2.3:** Sistema de cuerpos fijados a una base y grafo

## 2.3. Formulación clásica de la dinámica

### 2.3.1. Estado del arte

Existen una gran variedad de trabajos relacionados con el comportamiento dinámico de robots datando los primeros de principios de la década de los 80. Gran parte de este trabajo es aplicable a una amplia gama de campos, incluso fuera de la robótica. (Featherstone, 1987) y (Hollerbach, Johnson, Lozano-Perez, Mason, y Brady, 1982) son los primeros libros de referencia de la dinámica del robot, más actualmente, en (Sciavicco, Villani, Oriolo, y Siciliano, 2009) se hace un profundo estudio de los últimos avances en cinemática y dinámica.

El enfoque clásico de la definición de las ecuaciones de movimiento se basa en la formulación de Lagrange, lo que llevó a algoritmos lentos de órdenes de magnitud de  $O(n^4)$  (Kahn y Roth, 1971), (Uicker, 1967). El primer algoritmo en  $O(n)$  para la dinámica inversa utiliza la formulación de Newton-Euler (Stepanenko y Vukobratovic, 1976), (Orin, McGhee, Vukobratovic, y Hartoch, 1979), sin embargo, el algoritmo más citado es el *Recursive Newton Euler Algorithm* (RNEA) desarrollado por (Luh, Walker, y Paul, 1980). En ese mismo año, se descubrió un algoritmo recursivo Lagrangiano en  $O(n)$ , pero resultó ser más

lento que el RNEA(Hollerbach, 1980).

El RNEA se utilizó para desarrollar el algoritmo *Composite Rigid Body Algorithm* (CRBA) para la dinámica directa (Walker y Orin, 1982), que se ejecuta en un tiempo de  $O(n^3)$  y que fue bastante eficaz para  $n$  pequeñas. Su eficacia está directamente relacionada con la eficiencia de cálculo de la matriz de inercia. Mejoras adicionales (Featherstone, 1987), (Balafoutis y Patel, 1989) y (McMillan y Orin, 1998) han llevado a un aumento de velocidad de en un factor de dos. Más tarde se publicó el *Modified Composite Rigid Body Algorithm* que es muy eficiente para el cálculo de la matriz de inercia (Lilly y Orin, 1991).

El algoritmo en  $O(n)$  más antiguo conocido para la dinámica directa se basa en programación dinámica (Vereshchagin, 1970). Otro algoritmo en  $O(n)$  fue desarrollado para mecanismos con articulaciones esféricas (Armstrong, 1979) y más tarde se desarrolló el *Articulated Body Algorithm* (ABA) (Featherstone, 1983) para manipuladores con articulaciones de un solo grado de libertad, seguida de una versión que incluía un modelo de articulación general y que resultó más rápido (Featherstone, 1987). (Johanni, Otter, y Brandl, 1986) hizo nuevas mejoras para que el ABA fuera comparable en velocidad al CRBA para  $n = 6$ . Años más tarde, otros avances fueron publicados basándose en el algoritmo ABA (McMillan y Orin, 1995).

El álgebra de operadores espaciales fue desarrollado utilizando paralelismos entre el filtro de Kalman y la dinámica directa (Rodríguez, 1987). Esto permitió una factorización alternativa de la matriz de inercia, basándose en el algoritmo ABA (Rodríguez, Kreutz-Delgado, y Jain, 1991). El álgebra de vectores espaciales se usó para demostrar que tanto CRBA como ABA son dos métodos que permiten resolver el mismo sistema lineal (Pai, Cloutier, y Ascher, 1997). Una eficiente formulación recursiva de Lagrange para la dinámica inversa y directa

para cadenas en serie con eslabones flexibles fue desarrollado en (Book, 1984).

Los sistemas de cadena cinemática cerrada resultan más complicados y requieren de unos algoritmos más complejos que las estructuras de cadena cinemática abierta, este problema ha sido ampliamente tratado y se discute en detalle en (Roberson y Schwertassek, 1988) y (Wittenburg, 1977).

Los avances más notables en los últimos años en algoritmos dinámicos están íntimamente relacionados con el desarrollo de ordenadores con múltiples procesadores. Esto ha permitido obtener algoritmos dinámicos mucho más rápidos, hasta de un orden de magnitud de  $O(\log(n))$  utilizando programación en paralelo. Un ejemplo se puede observar en (Featherstone, 1999a) y (Featherstone, 1999b), donde se computa la ecuación de movimiento de dos cuerpos rígidos independientes y finalmente se calcula la ecuación de la unión de las dos mitades. El primer artículo se centra en cadenas cinemáticas abiertas y el segundo en cadenas cerradas. Otros ejemplos de algoritmos de orden  $O(\log(n))$  son (Sharf, D'Eleuterio, y Fijany, 1995) y (Featherstone y Fijany, 1999).

### 2.3.2. Formulación de Lagrange

Una de las formulaciones más conocidas para obtener las ecuaciones de movimiento para un sistema de cuerpos rígidos complejo es la formulación de Lagrange. Esta formulación, permite derivar las ecuaciones de movimiento de una forma sistemática y con independencia del sistema de referencia. Para ello parte de un conjunto de variables  $q_i$ , denominadas coordenadas generalizadas, que en el caso de un humanoide son las posiciones de las articulaciones.

De la misma manera, es necesario describir las fuerzas que generan el movimiento como fuerzas generalizadas, éstas engloban los pares de los actuadores de las articulaciones, los pares de fricción y los pares de generados por el contacto con

el entorno.

La ecuación Lagrangiana o Lagrangiano es función de las posiciones generalizadas, las velocidades generalizadas y el tiempo:

$$L(q, \dot{q}) = T(q, \dot{q}) - V(q) \quad (2.3)$$

donde T y V denotan respectivamente la energía cinética y potencial del sistema.

La ecuación de movimiento de un sistema de sólidos rígidos según la teoría de Lagrange se enuncia de la siguiente manera:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta} = \tau \quad (2.4)$$

Cabe destacar que aunque los algoritmos basados en este método son más lentos computacionalmente que los basados en la formulación de Newton-Euler, tienen la ventaja de que sólo necesitan computar la energía potencial y cinética, por lo que resultan menos propensos a errores y, además, permiten reducir el número de ecuaciones necesarias para describir el movimiento del sistema.

### 2.3.3. Formulación de Newton-Euler

La formulación de Newton-Euler se basa en el balance de todas las fuerzas que actúan sobre los eslabones del robot. Esto resulta en ecuaciones que permiten una solución recursiva, lo que presenta una gran ventaja, ya que los algoritmos recursivos alcanzan una velocidad de computación mucho mayor que los no recursivos. Se les llama recursivos porque hacen uso de relaciones de recurrencia para calcular secuencias de resultados intermedios. Una relación de recurrencia es una fórmula que define el elemento siguiente como una secuencia de los elementos anteriores.

La formulación de Newton-Euler se basa en dos ecuaciones, la primera es la ecuación del movimiento translacional del centro de masas de Newton:

$$f_i - f_{i+1} = m_i \ddot{r}_{CM} - m_i g \quad (2.5)$$

donde  $f_i$  es la fuerza que el eslabón  $i - 1$  ejerce sobre el  $i$ ,  $f_{i+1}$  es la fuerza que el eslabón  $i$  ejerce sobre el  $i + 1$ ,  $\ddot{r}_{CM}$  es la aceleración del centro de masas,  $m_i$  es la masa del eslabón y  $g$  es la aceleración de la gravedad.

La segunda ecuación en que se basa esta formulación es la ecuación para el movimiento rotativo de Euler, que se define de la siguiente manera:

$$T_i - T_{i+1} = I_i \alpha_i + \omega_i \times (I_i \omega_i) \quad (2.6)$$

donde  $T_i$  es el par que ejerce el eslabón  $i - 1$  sobre el eslabón  $i$  respecto al origen de coordenadas del  $i - 1$ ,  $T_{i+1}$  es el par ejercido por el eslabón  $i$  sobre el  $i + 1$  con respecto al origen del eslabón  $i$ ,  $I_i$  es el tensor de inercia del eslabón  $i$ ,  $\alpha_i$  es la aceleración angular de dicho eslabón y  $\omega_i$  es su velocidad angular.



Capítulo **3**

# Notación espacial de la dinámica

## 3.1. Introducción

Construir el modelo dinámico completo de un robot humanoide es complicado debido al alto número de grados de libertad que estos robots suelen tener. El enfoque de vectores espaciales de  $6D$  proporciona una notación compacta que permite simplificar las ecuaciones y mejorar el tiempo de computación en los algoritmos de control. Los vectores espaciales usados para la obtención de la dinámica se basan en dos componentes principales, la velocidad espacial y la fuerza espacial. Partiendo de estos dos componentes se desarrolla toda una teoría que permite definir el movimiento del humanoide.

En este capítulo se presenta el fundamento matemático de esta Tesis de Máster, el álgebra de vectores espaciales. Los conceptos desarrollados a continuación han sido extraídos de (Featherstone, 2008).

## 3.2. Álgebra de vectores espaciales

### 3.2.1. Velocidad espacial

En primer lugar se va a dar una explicación general de la velocidad espacial de un cuerpo rígido, esta definición toma su idea principal del teorema de Chasles (Poole, Safko, y Goldstein, 2000):

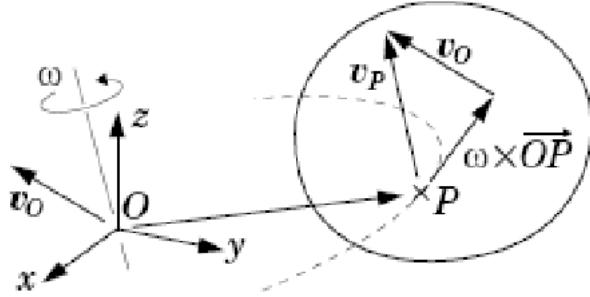
*El desplazamiento más general de un sólido rígido es una traslación más una rotación.*

Dado un sólido rígido  $B$  y dado un punto  $O$  situado en cualquier lugar del espacio, la velocidad de dicho sólido se define, de forma general, como una traslación de velocidad lineal  $v_0$ , de un punto del cuerpo que en un instante de tiempo concreto coincide con  $O$ , más una rotación de velocidad angular  $\omega$  respecto a un eje que pasa por  $O$ .

En consecuencia, se puede definir la velocidad de cualquier otro punto  $P$  del sólido rígido como:

$$v_p = v_o + \omega \times \overline{OP} \quad (3.1)$$

Donde  $P$  es el punto desde donde se calcula la velocidad del sólido y  $\overline{OP}$  es la distancia entre puntos (Figura 3.1). La ecuación (3.1) especifica claramente las dos contribuciones existentes en la velocidad de un sólido, la velocidad lineal y la angular.



**Figura 3.1:** Velocidad de un sólido rígido

Si se introduce el origen de coordenadas cartesiano  $O_{xyz}$  en el que se definen las bases ortonormales  $\{i, j, k\} \in E^3$ , se puede expresar  $v_o$  y  $\omega$  en términos de estas coordenadas cartesianas como:

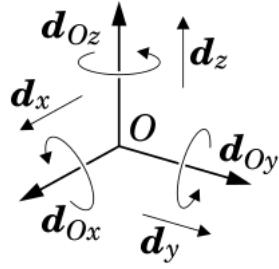
$$\omega = \omega_x i + \omega_y j + \omega_z k \quad (3.2)$$

$$v_o = v_{ox} i + v_{oy} j + v_{oz} k \quad (3.3)$$

Una vez definido esto, es posible expresar la velocidad del sólido como la suma de seis movimientos elementales, tales como una rotación de magnitud  $\omega_x$  respecto al eje  $O_x$ , una rotación de magnitud  $\omega_y$  respecto al eje  $O_y$ , una rotación de magnitud  $\omega_z$  respecto al eje  $O_z$  y tres traslaciones de magnitudes  $v_{ox}$ ,  $v_{oy}$  y  $v_{oz}$  en las direcciones  $x$ ,  $y$  y  $z$ .

El objetivo es obtener el vector de la velocidad espacial del sólido que define un movimiento translacional y rotacional. Por ello se definen las siguientes bases, denominadas bases de movimiento de Plücker, ilustradas en la Figura 3.2 como:

$$D_o = \{d_{ox}, d_{oy}, d_{oz}, d_x, d_y, d_z\} \in M^6 \quad (3.4)$$



**Figura 3.2:** Coordenadas de Plücker para la velocidad espacial

Por lo tanto, se puede definir la velocidad espacial de un sólido rígido mediante la ecuación:

$$\hat{v} = \omega_x d_{ox} + \omega_y d_{oy} + \omega_z d_{oz} + v_{ox} d_x + v_{oy} d_y + v_{oz} d_z \quad (3.5)$$

Esto según la notación de Plücker puede ser expresado como:

$$\hat{v} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ v_{ox} \\ v_{oy} \\ v_{oz} \end{bmatrix} = \begin{bmatrix} \omega \\ v_o \end{bmatrix} \quad (3.6)$$

obteniendo finalmente un vector 6D a partir de dos vectores 3D.

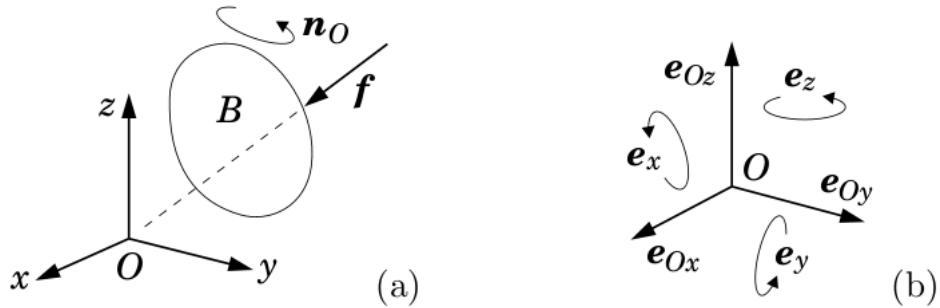
Aunque las coordenadas de Plücker datan de 1886, la utilización de esta nomenclatura para definir movimiento es posterior (Featherstone, 2006). La aplicación de esta notación a los algoritmos de obtención de la dinámica de un robot humanoide produce, según el autor, una notable reducción del tiempo computacional en comparación con la notación clásica de vectores 3D.

Otra aportación importante de la notación espacial es que el vector velocidad espacial proporciona una *descripción completa* de la velocidad del sólido rígido,

o lo que es lo mismo, se puede demostrar que es invariante respecto a la localización del sistema de referencia.

### 3.2.2. Fuerza espacial

Para definir la fuerza espacial se seguirá una metodología similar. Dado un punto arbitrario  $O$  colocado en cualquier lugar del espacio, la fuerza generalizada que actúa en un sólido rígido  $B$  consiste en una fuerza lineal  $f$ , actuando a lo largo de una línea que pasa por  $O$ , junto con un par  $n_o$ , que representa el momento total respecto de  $O$  (Ver Figura 3.3(a)).



**Figura 3.3:** Fuerza actuando en un sólido rígido (a) y bases de Plücker para las coordenadas de fuerza (b)

Siguiendo la misma sistemática que en el caso de la velocidad espacial, se va a representar la fuerza espacial en el sólido como la concatenación de dos vectores 3D,  $f$  tomará el valor de  $\omega$  y  $n_o$  el lugar de  $v_o$ . El par con respecto a cualquier punto  $P$  se define como:

$$\mathbf{n}_p = \mathbf{n}_o + \mathbf{f} \times \overline{OP} \quad (3.7)$$

De la misma forma, se introduce el origen de coordenadas cartesianas  $O_{xyz}$  en el que se definen las bases ortonormales  $\{i, j, k\} \in E^3$ , entonces, se define  $f$  y  $n_o$  en términos de las coordenadas cartesianas.

$$\mathbf{f} = f_x i + f_y j + f_z k \quad (3.8)$$

$$n_o = n_{ox}i + n_{oy}j + n_{oz}k \quad (3.9)$$

Se definen las siguientes bases de Plücker para las coordenadas de fuerza (Ver Figura 3.3(b)):

$$\epsilon_o = \{e_x, e_y, e_z, e_{ox}, e_{oy}, e_{oz}\} \in F^6 \quad (3.10)$$

Por lo tanto, se puede definir la fuerza del sólido rígido de la misma forma en estas coordenadas:

$$\hat{f} = n_{ox}e_x + n_{oy}e_y + n_{oz}e_z + f_xe_{ox} + f_ye_{oy} + f_ze_{oz} \quad (3.11)$$

y de la misma forma expresamos el vector de la fuerza espacial de 6D partiendo de dos vectores 3D.

$$\hat{f} = \begin{bmatrix} n_{ox} \\ n_{oy} \\ n_{oz} \\ f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} n_o \\ f \end{bmatrix} \quad (3.12)$$

La fuerza espacial proporciona, al igual que la velocidad espacial, una *descripción completa* de las fuerzas que actúan en el cuerpo y además es invariante respecto del sistema de referencia.

### 3.2.3. Producto escalar

El producto escalar se define en el álgebra espacial de modo que uno de los argumentos debe ser un vector de movimiento y el otro un vector de fuerza. De esta forma se obtiene una magnitud escalar que representa energía o potencia.

Dado  $m \in M^6$  y  $f \in F^6$  el producto escalar de ambos se puede definir como  $m.f$  o  $f.m$ . Sin embargo, las operaciones  $m.m$  o  $f.f$  no están definidas.

El productor escalar crea una conexión entre  $M^6$  y  $F^6$ , lo que provoca algunas propiedades interesantes:

1. Se usan coordenadas duales en  $M^6$  y  $F^6$ .
2. Los vectores de fuerza y movimiento obedecen a distintas reglas de transformación de coordenadas.
3. Se define el operador producto cruzado, diferente para movimientos y fuerzas.

Un sistema dual en  $M^6$  y  $F^6$  es cualquier sistema de coordenadas formado por dos bases  $\{d_1, \dots, d_6\} \in M^6$  y  $\{e_1, \dots, e_6\} \in F^6$  de forma que cumplan la siguiente condición:

$$d_i e_j = \begin{cases} 1 & \text{si } i=j \\ 0 & \text{otro caso} \end{cases} \quad (3.13)$$

La propiedad más importante de los sistemas de coordenadas duales es que si  $\underline{m}$  y  $\underline{f}$  son vectores que representan respectivamente  $m \in M^6$  y  $f \in F^6$ , entonces se define el producto escalar como:

$$\underline{m} \cdot \underline{f} = \underline{m}^T \cdot \underline{f} \quad (3.14)$$

La consecuencia inmediata de esto es que se necesitan diferentes matrices de transformación de coordenadas para movimientos y fuerzas. En efecto, si  $X$  produce una transformación en un vector de movimiento y  $X^*$  realiza la misma transformación pero en un vector de fuerza, ambas matrices se relacionan de la siguiente manera:

$$X^* = X^{-T} \quad (3.15)$$

### 3.2.4. Transformación de coordenadas

Como se ha apuntado anteriormente, las transformaciones de coordenadas de vectores espaciales de movimiento y fuerza no son las mismas. Supongamos

que se quiere pasar de un sistema de coordenadas  $A$  a uno  $B$ , la matriz de transformación para el vector de movimiento se escribe habitualmente como  ${}^B X_A$  y la misma transformación para el vector de fuerza se escribe  ${}^B X_A^*$ . Ambas matrices vienen relacionadas por la ecuación (3.15) de la forma  ${}^B X_A^* = {}^B X_A^{-T}$ .

La fórmula para  ${}^B X_A$  depende solamente de la posición y orientación del sistema de coordenadas  $B$  con respecto al sistema  $A$ , por lo cual, puede ser expresado como el producto de dos simples transformaciones, una traslación y una rotación.

**- Rotación:**

Sean  $A$  y  $B$  dos sistemas de coordenadas cuyo origen  $O$  es común. Sea también el vector espacial  $\hat{m} \in M^6$  formado por dos vectores 3D,  $m$  y  $m_0$ . Así mismo, sea  $E$  la matriz de rotación de  $3 \times 3$  que transforma del sistema coordenado  $A$  al  $B$  (ver Figura 3.4). Según esto se puede escribir:

$${}^B \hat{m} = \begin{bmatrix} {}^B m \\ {}^B m_0 \end{bmatrix} = \begin{bmatrix} E^A m \\ E^A m_0 \end{bmatrix} = \begin{bmatrix} E & 0 \\ 0 & E \end{bmatrix} {}^A \hat{m} \quad (3.16)$$

luego

$${}^B X_A = \begin{bmatrix} E & 0 \\ 0 & E \end{bmatrix} \quad (3.17)$$

La transformación equivalente para el vector fuerza espacial es exactamente la misma.

$${}^B X_A^* = {}^B X_A^{-T} = \begin{bmatrix} E & 0 \\ 0 & E \end{bmatrix} \quad (3.18)$$

**- Traslación:**

Sean  $O$  y  $P$  dos puntos del espacio, origen de dos sistemas coordinados que guardan la misma orientación. Sea el vector espacial  $\hat{m} \in M^6$ , formado por dos vectores 3D,  $m$  y  $m_0$ . Cualquier movimiento puede ser expresado de forma generalizada como:

$$m_P = m_O + m \times \overline{OP} \quad (3.19)$$

Por lo tanto, en coordenadas espaciales:

$$\hat{m}_P = \begin{bmatrix} m \\ m_0 \end{bmatrix} = \begin{bmatrix} m \\ m_0 - r \times m \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -r \times & 1 \end{bmatrix} \hat{m}_O \quad (3.20)$$

donde  $r = \overline{OP}$ . En consecuencia:

$${}^P X_O = \begin{bmatrix} 1 & 0 \\ -r \times & 1 \end{bmatrix} \quad (3.21)$$

y

$${}^P X_O^* = {}^P X_O^{-T} = \begin{bmatrix} 1 & -r \times \\ 0 & 1 \end{bmatrix} \quad (3.22)$$

Para facilitar la transcripción de las transformaciones de Plücker se va a recurrir a la Figura 3.4. Según ella, se expresa  $\text{rot}_x(\theta)$  como la rotación espacial de  $\theta$  grados sobre el eje  $X$ . Así mismo, se puede escribir  ${}^B X_A = \text{rot}(E)xlt(r)$ , lo que describe la transformación de A hasta B como una rotación más una traslación.

---

$\text{rx}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{bmatrix}$	$\text{ry}(\theta) = \begin{bmatrix} c & 0 & -s \\ 0 & 1 & 0 \\ s & 0 & c \end{bmatrix}$	$\text{rz}(\theta) = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$(c = \cos(\theta), \quad s = \sin(\theta))$		
$\text{rot}(E) = \begin{bmatrix} E & \mathbf{0} \\ \mathbf{0} & E \end{bmatrix}$	$\text{xlt}(r) = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ -r \times & \mathbf{1} \end{bmatrix}$	$\text{rot}_x(\theta) = \text{rot}(\text{rx}(\theta))$ $\text{rot}_y(\theta) = \text{rot}(\text{ry}(\theta))$ $\text{rot}_z(\theta) = \text{rot}(\text{rz}(\theta))$

---

**Figura 3.4:** Funciones de rotación y transformadas de Plücker

La expresión  $a \times$  define un operador denominado producto cruzado euclídeo. Si  $a$  es un vector cartesiano 3D, entonces  $a \times$  es una matriz  $3 \times 3$  expresada según la fórmula:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \quad (3.23)$$

Este operador posee ciertas propiedades matemáticas que son listadas en la Figura 3.5.

---


$$\begin{aligned} \mathbf{a} \times &= -\mathbf{a} \times^T \\ (\lambda \mathbf{a}) \times &= \lambda (\mathbf{a} \times) && (\lambda \text{ is a scalar}) \\ (\mathbf{a} + \mathbf{b}) \times &= \mathbf{a} \times + \mathbf{b} \times \\ (\mathbf{E} \mathbf{a}) \times &= \mathbf{E} \mathbf{a} \times \mathbf{E}^{-1} && (\mathbf{E} \text{ is orthonormal}) \\ (\mathbf{a} \times) \mathbf{b} &= -(\mathbf{b} \times) \mathbf{a} \\ (\mathbf{a} \times \mathbf{b}) \cdot &= \mathbf{a} \cdot \mathbf{b} \times && ((\mathbf{a} \times \mathbf{b})^T = \mathbf{a}^T \mathbf{b} \times) \\ (\mathbf{a} \times \mathbf{b}) \times &= \mathbf{a} \times \mathbf{b} \times - \mathbf{b} \times \mathbf{a} \times \\ (\mathbf{a} \times \mathbf{b}) \times &= \mathbf{b} \mathbf{a}^T - \mathbf{a} \mathbf{b}^T \\ \mathbf{a} \times \mathbf{b} \times &= \mathbf{b} \mathbf{a}^T - (\mathbf{a} \cdot \mathbf{b}) \mathbf{1}_{3 \times 3} \end{aligned}$$


---

**Figura 3.5:** Propiedades del producto cruzado euclídeo

### 3.2.5. Operador producto cruzado espacial

Sea  $r \in E^3$  un vector 3D que está girando con una velocidad angular  $\omega$ . La derivada de ese vector viene dada por la fórmula  $\dot{r} = \omega \times r$ . En este contexto  $\omega \times$  está actuando como un operador de diferenciación, que pasa de  $r$  a  $\dot{r}$ .

El mismo significado matemático puede ser extrapolado a los vectores espaciales. Al igual que en casos anteriores, el operador producto cruzado espacial tiene dos vertientes, una para vectores de movimiento y otra para vectores de fuerza.

Entonces, sean  $\hat{m} \in M^6$  y  $\hat{f} \in F^6$  dos vectores espaciales (que pudieran estar fijos a un sólido rígido) moviéndose a una velocidad  $v \in M^6$ . Se definen dos nuevos operadores,  $\times$  y  $\times^*$  que satisfacen las siguientes ecuaciones:

$$\dot{\hat{m}} = \hat{v} \times \hat{m} \tag{3.24}$$

y

$$\dot{\hat{f}} = \hat{v} \times^* \hat{m} \tag{3.25}$$

Estos operadores se pueden definir de una forma explícita tal y como se muestra en la Figura 3.6. Cada entrada de la tabla es la derivada temporal del vector base situado en la fila superior, asumiendo una velocidad igual a la base de la columna de la izquierda.

$\times$	$d_{Ox}$	$d_{Oy}$	$d_{Oz}$	$d_x$	$d_y$	$d_z$
$d_{Ox}$	<b>0</b>	$d_{Oz}$	$-d_{Oy}$	<b>0</b>	$d_z$	$-d_y$
$d_{Oy}$	$-d_{Oz}$	<b>0</b>	$d_{Ox}$	$-d_z$	<b>0</b>	$d_x$
$d_{Oz}$	$d_{Oy}$	$-d_{Ox}$	<b>0</b>	$d_y$	$-d_x$	<b>0</b>
$d_x$	<b>0</b>	$d_z$	$-d_y$	<b>0</b>	<b>0</b>	<b>0</b>
$d_y$	$-d_z$	<b>0</b>	$d_x$	<b>0</b>	<b>0</b>	<b>0</b>
$d_z$	$d_y$	$-d_x$	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>

$\times^*$	$e_x$	$e_y$	$e_z$	$e_{Ox}$	$e_{Oy}$	$e_{Oz}$
$d_{Ox}$	<b>0</b>	$e_z$	$-e_y$	<b>0</b>	$e_{Oz}$	$-e_{Oy}$
$d_{Oy}$	$-e_z$	<b>0</b>	$e_x$	$-e_{Oz}$	<b>0</b>	$e_{Ox}$
$d_{Oz}$	$e_y$	$-e_x$	<b>0</b>	$e_{Oy}$	$-e_{Ox}$	<b>0</b>
$d_x$	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	$e_z$	$-e_y$
$d_y$	<b>0</b>	<b>0</b>	<b>0</b>	$-e_z$	<b>0</b>	$e_x$
$d_z$	<b>0</b>	<b>0</b>	<b>0</b>	$e_y$	$-e_x$	<b>0</b>

Figura 3.6: Producto cruzado espacial

Usando las fórmulas representadas en la Figura 3.6 se pueden definir las dos vertientes del producto cruzado. La primera toma dos vectores de movimiento como argumento y produce un vector de movimiento de esta forma:

$$m_1 \times m_2 = \begin{pmatrix} m_1 \\ m_{1O} \end{pmatrix} \times \begin{pmatrix} m_2 \\ m_{2O} \end{pmatrix} = \begin{pmatrix} m_1 \times m_2 \\ m_1 \times m_{2O} + m_{1O} \times m_2 \end{pmatrix} \quad (3.26)$$

El segundo vector toma como argumentos un vector de movimiento y un vector de fuerza y resulta en un vector de fuerza.

$$m \times f = \begin{pmatrix} m \\ m_O \end{pmatrix} \times \begin{pmatrix} f \\ f_O \end{pmatrix} = \begin{pmatrix} m \times f_O + m_O \times f \\ m \times f \end{pmatrix} \quad (3.27)$$

Por último, en la Figura 3.7 se definen algunas de las propiedades matemáticas que posee este operador.

---

$\mathbf{v} \times^* = -\mathbf{v} \times^T$		
$(\lambda \mathbf{v}) \times = \lambda (\mathbf{v} \times)$	$(\lambda \mathbf{v}) \times^* = \lambda (\mathbf{v} \times^*)$	(λ is a scalar)
$(\mathbf{u} + \mathbf{v}) \times = \mathbf{u} \times + \mathbf{v} \times$	$(\mathbf{u} + \mathbf{v}) \times^* = \mathbf{u} \times^* + \mathbf{v} \times^*$	
$(\mathbf{X}\mathbf{v}) \times = \mathbf{X} \mathbf{v} \times \mathbf{X}^{-1}$	$(\mathbf{X}\mathbf{v}) \times^* = \mathbf{X}^* \mathbf{v} \times^* (\mathbf{X}^*)^{-1}$	(Plücker transform)
$\mathbf{u} \times \mathbf{v} = -\mathbf{v} \times \mathbf{u}$		
$(\mathbf{u} \times \mathbf{v}) \cdot = -\mathbf{v} \cdot \mathbf{u} \times^*$	$(\mathbf{u} \times^* \mathbf{f}) \cdot = -\mathbf{f} \cdot \mathbf{u} \times$	
$(\mathbf{u} \times \mathbf{v}) \times = \mathbf{u} \times \mathbf{v} \times - \mathbf{v} \times \mathbf{u} \times$	$(\mathbf{u} \times \mathbf{v}) \times^* = \mathbf{u} \times^* \mathbf{v} \times^* - \mathbf{v} \times^* \mathbf{u} \times^*$	

---

**Figura 3.7:** Propiedades del producto cruzado espacial

### 3.2.6. Diferenciación

La derivada de un vector espacial se define de la siguiente manera: sea  $A$  un sistema de coordenadas de Plücker, que se mueve a una velocidad espacial  $v_A$ . Sean  ${}^A m$  y  ${}^A f$  los vectores coordinados que representan a los vectores espaciales  $\hat{m} \in M^6$  y  $\hat{f} \in F^6$ . Entonces la derivada respecto del tiempo se puede expresar como:

$$\left( \frac{dm}{dt} \right)_A = \frac{d^A m}{dt} + {}^A v_A \times {}^A m \quad (3.28)$$

y

$$\left( \frac{df}{dt} \right)_A = \frac{d^A f}{dt} + {}^A v_A \times^* {}^A f \quad (3.29)$$

La derivada respecto del tiempo de un vector espacial que cambia solamente debido a que se está moviendo viene dada por:

$$\frac{d}{dt} s = v \times s \quad (3.30)$$

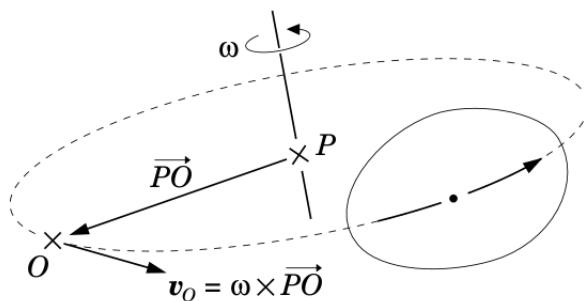
donde  $v$  es la velocidad de  $s$ . Esta fórmula es útil para diferenciar cantidades que no cambian por sí mismas, sino porque están referenciadas a un sólido rígido en movimiento.

### 3.2.7. Aceleración

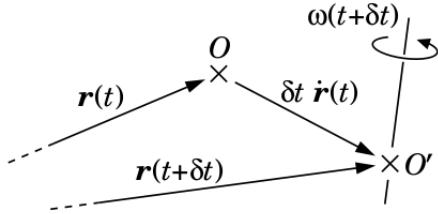
Se define la aceleración espacial de un sólido rígido como la tasa de variación de su velocidad espacial. Sin embargo, esto significa que la aceleración espacial difiere ligeramente de la aceleración clásica de un cuerpo rígido, como se explicará a continuación.

En la Figura 3.8 se considera un sólido rotando a una velocidad angular constante respecto a un eje fijo del espacio. Si la velocidad es constante, la aceleración es cero, sin embargo, en el caso de que estuviéramos hablando de aceleración espacial, ésta es, en general, distinta de cero.

En primer lugar, hay que tener en cuenta que la aceleración espacial es una propiedad del cuerpo como un todo, no una propiedad de puntos individuales fijados al cuerpo. En segundo lugar,  $\hat{v}_O$  no es la velocidad de un punto fijo del cuerpo, sino una medida del flujo de puntos que atraviesan  $O$ . Por tanto,  $\hat{v}_O$  no es la aceleración de un punto individual, sino la tasa de variación del flujo de puntos.



**Figura 3.8:** Aceleración de un cuerpo rotando uniformemente



**Figura 3.9:** Esquema de la demostración de la fórmula de la velocidad espacial

Para demostrar esto, se pasará a observar la Figura 3.9, en donde se introduce el punto  $O'$ , que coincide con  $O$  en el tiempo  $t$ , y el vector  $r$ , que representa la posición de  $O'$  como función del tiempo. Por lo tanto,  $\dot{r}$  es la velocidad de  $O'$  y  $\ddot{r}$  su aceleración.

En los textos clásicos de mecánica, la aceleración de un cuerpo rígido viene especificada en función de dos vectores 3D,  $\omega$  y  $\ddot{r}$ . A continuación se va a ver cómo difiere esto de la aceleración espacial. Se parte de la definición de derivada.

$$\frac{d}{dt}v_O(t) = \lim_{\delta t \rightarrow 0} \frac{v_O(t + \delta t) - v_O(t)}{\delta t} \quad (3.31)$$

En un primer momento  $v_O(t) = \dot{r}(t)$  porque  $O'$  coincide con  $O$ . Sin embargo,  $v_O(t + \delta t)$  no es igual a  $\dot{r}(t + \delta t)$  ya que  $O'$  se ha movido respecto de  $O$  una cantidad  $\delta t \dot{r}(t)$ , por lo que se tiene que:

$$v_O(t + \delta t) = \dot{r}(t + \delta t) - \omega \times \delta t \dot{r}(t) \quad (3.32)$$

Sustituyendo la ecuación (3.32) en la ecuación (3.31) se obtiene:

$$\frac{d}{dt}v_O(t) = \lim_{\delta t \rightarrow 0} \frac{\dot{r}(t + \delta t) - \omega \times \delta t \dot{r}(t) - \dot{r}(t)}{\delta t} = \ddot{r} - \omega \times \dot{r} \quad (3.33)$$

Lo que implica que:

$$\hat{a}_O = \begin{bmatrix} \dot{\omega} \\ \ddot{r} - \omega \times \dot{r} \end{bmatrix} \quad (3.34)$$

Comparando la expresión (3.34) con la notación de la mecánica clásica, se define un nuevo vector 6D al que se denomina *aceleración clásica*, cuya fórmula es:

$$\hat{a}'_O = \begin{bmatrix} \dot{\omega} \\ \ddot{r} \end{bmatrix} \quad (3.35)$$

De esta forma podemos definir la aceleración espacial.

$$\hat{a}_O = \hat{a}'_O - \begin{bmatrix} 0 \\ \omega \times v_O \end{bmatrix} \quad (3.36)$$

La gran ventaja que supone la aceleración espacial frente a la aceleración clásica se produce al relacionar las velocidades y aceleraciones de dos cuerpos, ya que la aceleración espacial obedece las mismas combinaciones y transformaciones coordenadas que la velocidad.

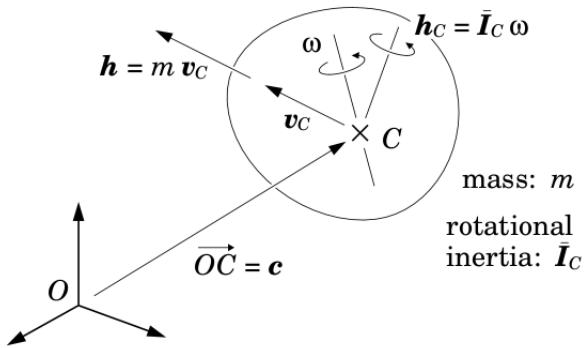
Si, por ejemplo, los cuerpos  $B_1$  y  $B_2$  tienen velocidades  $v_1$  y  $v_2$ , y  $v_{rel}$  es la velocidad relativa de  $B_2$  respecto de  $B_1$ , entonces la velocidad espacial de un cuerpo respecto del otro se puede expresar como

$$\hat{v}_2 = \hat{v}_1 + \hat{v}_{rel} \quad (3.37)$$

Si derivamos se obtiene la siguiente expresión:

$$\frac{d}{dt} \hat{v}_2 = \frac{d}{dt} \hat{v}_1 + \frac{d}{dt} \hat{v}_{rel} \rightarrow \hat{a}_2 = \hat{a}_1 + \hat{a}_{rel} \quad (3.38)$$

La ecuación (3.38) supone una ventaja muy importante en el cálculo de aceleraciones para sistemas de múltiples cuerpos rígidos como los robots humanoides. Al vector aceleración espacial se le denomina *vector verdadero*, porque obedece las mismas reglas de transformación de coordenadas y posee las mismas propiedades algebraicas que la velocidad. Esto evita la aparición de términos de Coriolis, como en el caso de la *aceleración clásica* y presenta una notación más compacta que permite reducir el tiempo de computación.



**Figura 3.10:** Momento espacial

### 3.2.8. Momento

Al igual que en el caso de las fuerzas y velocidades, se puede definir el momento espacial como la composición de dos vectores 3D,  $h = mv_C$  donde  $h$  es el momento lineal,  $m$  la masa y  $v_C$  la velocidad lineal con la que se mueve el cuerpo; y  $h_C = I_C\omega$ , donde  $h_C$  es el momento angular,  $I_C$  es la inercia respecto del centro de masas y  $\omega$  es la velocidad de rotación. En la Figura 3.10 se observa un esquema del momento espacial.

Por lo tanto y siguiendo la notación antes mencionada, se puede definir el momento espacial de la siguiente manera:

$$\hat{h}_C = \begin{bmatrix} h_C \\ h \end{bmatrix} = \begin{bmatrix} I_C\omega \\ mv_C \end{bmatrix} \quad (3.39)$$

En general, el momento respecto de cualquier punto O se expresa de la forma

$$h_O = h_C + \overline{OC} \times h \quad (3.40)$$

Si se sustituye (3.39) en (3.40) se obtiene:

$$\hat{h}_O = \begin{bmatrix} h_O \\ h \end{bmatrix} = \begin{bmatrix} I_C\omega + \overline{OC} \times mv_C \\ mv_C \end{bmatrix} = \begin{bmatrix} 1 & \overline{OC} \times \\ 0 & 0 \end{bmatrix} \hat{h}_C \quad (3.41)$$

### 3.2.9. Inercia

El momento de inercia es una magnitud escalar que refleja la distribución de masas de un cuerpo respecto a un eje de giro. Se podría entender como la resistencia que tiene un cuerpo a adquirir una aceleración angular.

La fórmula general para el momento de inercia es:

$$I = \int_V r^2 dm \quad (3.42)$$

Extendiendo esta fórmula a cada uno de los ejes coordinados se obtiene el tensor de inercia de  $3 \times 3$ .

Para definir la inercia espacial basta fijarse en la siguiente ecuación:

$$\hat{I}_C = \begin{bmatrix} I_C & 0 \\ 0 & m_1 \end{bmatrix} \quad (3.43)$$

La regla de transformación de coordenadas para la inercia espacial es:

$$I_B = {}^B X_A^* I_A {}^A X_B \quad (3.44)$$

Caben destacar varias propiedades de la inercia espacial:

1. Para definir la inercia espacial de un cuerpo rígido sólo son necesarios 10 parámetros.
2. Esta matriz es simétrica y definida positiva.
3. La inercia total de un cuerpo compuesto de varias partes es igual a la suma de todas las inercias de las partes.

$$I_{TOT} = \sum I_i \quad (3.45)$$

4. La energía cinética de un cuerpo rígido dadas su velocidad espacial y su inercia espacial es:

$$T = \frac{1}{2} v \cdot I v \quad (3.46)$$

### 3.2.10. Ecuación de movimiento

La dinámica de un sistema de cuerpos rígidos se define mediante la ecuación de movimiento. La ecuación espacial de movimiento de un cuerpo rígido establece que la fuerza neta que actúa sobre el cuerpo es igual a la tasa de variación de su momento.

$$f = \frac{d}{dt}(Iv) = Ia + v \times^* Iv \quad (3.47)$$

Muchas veces se escribe esta ecuación como:

$$f = Ia + p \quad (3.48)$$

donde  $I$  y  $p$  son los coeficientes y  $f$  y  $a$  las variables. Se denomina fuerza parcial a  $p$  y se define como el valor que  $f$  debe tomar para producir aceleración nula.

Si se tiene en cuenta que la fuerza neta,  $f_{neta}$ , es la suma de una fuerza desconocida,  $f_u$ , y de la fuerza de la gravedad,  $f_g$ , la ecuación se puede escribir como:

$$f_{neta} = Ia + v \times^* Iv \quad (3.49)$$

de tal forma que:

$$f_u = Ia + p \quad (3.50)$$

y

$$p = v \times^* Iv - f_g \quad (3.51)$$

Tomando como base la ecuación de movimiento, ecuación (3.49), se han ido desarrollando todos los algoritmos dinámicos en sus diferentes modalidades. Partiendo de este principio, se pueden calcular los pares y aceleraciones de un sistema de cuerpos rígidos.

### 3.3. Formulación espacial de la dinámica de robots humanoides

#### 3.3.1. Dinámica inversa: Recursive Newton-Euler Algorithm

La dinámica inversa trata el problema de la obtención de los pares que se producen en las articulaciones a partir de la aceleración en un sistema de sólidos rígidos. Se utiliza habitualmente en control de movimiento, construcción y optimización de trayectorias y en diseño mecánico de robots o figuras animadas. La dinámica inversa en su forma más genérica se puede expresar como:

$$\tau = ID(modelo, q, \dot{q}, \ddot{q})$$

donde  $q, \dot{q}, \ddot{q}$  y  $\tau$  son los vectores generalizados de posición, velocidad, aceleración y par.

Para la obtención de la dinámica inversa el algoritmo más utilizado es el Recursive Newton-Euler Algorithm (RNEA) y es el que se va a utilizar en esta Tesis de Máster. El algoritmo RNEA utilizado aquí, tiene una complejidad de  $O(n)$ , donde  $n$  es el número de grados de libertad. Uno de sus principales atributos es que es recursivo, es decir, cada una de las variables depende de un conjunto de variables previas. A continuación se va a realizar un esquema de la programación de este algoritmo para cadenas cinemáticas abiertas.

**Paso 1:** Calcular la velocidad y aceleración de cada uno de los eslabones de la cadena cinemática.

Sean  $v_i$  y  $a_i$  la velocidad y aceleración del eslabón  $i$ . La velocidad de un eslabón en una cadena cinemática se puede definir recursivamente como la suma de la velocidad del eslabón padre, más la velocidad de la articulación que lo conecta con su eslabón padre.

$$v_i = v_{\lambda(i)} + S_i \dot{q}_i \quad (v_0 = 0) \quad (3.52)$$

donde  $v_i$  es la velocidad espacial del eslabón  $i$ ,  $v_{\lambda(i)}$  es la velocidad espacial del eslabón padre,  $S_i$  es el subespacio de vectores espaciales y  $\dot{q}_i$  es la velocidad a la que gira el motor de la articulación.

La relación de recurrencia de la aceleración viene dada por:

$$a_i = a_{\lambda(i)} + S_i \ddot{q}_i + v_i \times S_i \dot{q}_i \quad (a_0 = 0) \quad (3.53)$$

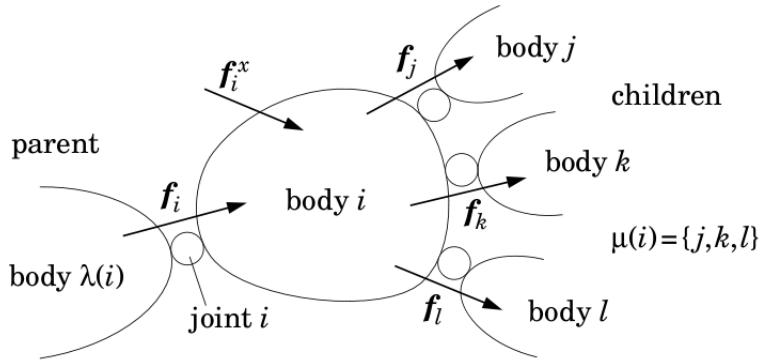
Iteraciones sucesivas desde  $i = 1 \dots N_B$ , siendo  $N_B$  el número de eslabones o cuerpos, calcularán las velocidades y aceleraciones de todos los eslabones de la cadena cinemática.

**Paso 2:** Sea  $f_i^B$  la fuerza neta que actúa sobre el eslabón  $i$ , sustituyendo en la ecuación de movimiento (3.49) se obtiene:

$$f_i^B = I_i a_i + v_i \times^* I_i v_i \quad (3.54)$$

**Paso 3:** Refiriéndose a la Figura 3.11, sea  $f_i$  la fuerza transmitida desde el eslabón padre  $\lambda(i)$  al eslabón  $i$  a través de la articulación  $i$  y sea  $f_i^x$  la fuerza externa actuando sobre el cuerpo  $i$ . Las fuerzas externas pueden ser debidas a campos de fuerzas, contactos físicos, etc. Entonces, la fuerza neta en el eslabón  $i$  se enuncia como:

$$f_i = f_i^B - f_i^x + \sum_{j \in \mu(i)} f_j \quad (3.55)$$



**Figura 3.11:** Fuerzas actuando en el eslabón  $i$

Computando sucesivamente la fórmula desde  $i = 1 \dots N_B$  se obtienen las fuerzas espaciales que atraviesan cada articulación. Ahora, solo falta obtener los pares generalizados para cada articulación.

$$\tau_i = S_i^T f_i \quad (3.56)$$

En la práctica, el algoritmo funciona mejor si los cálculos para el eslabón  $i$  son realizados en las coordenadas del propio eslabón, es decir, en coordenadas articulares. En la Figura 3.12 se hace una comparación de las ecuaciones generales y las ecuaciones en coordenadas articulares y se presenta el algoritmo según estas coordenadas.

Siempre es interesante tratar de sacar un sentido físico a las ecuaciones, por lo tanto, para finalizar, cabe destacar que este algoritmo no sólo calcula los pares en las articulaciones. Existen otras cantidades, como  $v_i$  o  $f_i$  que pueden ser muchas veces útiles. Por enunciar un ejemplo, la fuerza espacial  $f_i$  puede servir al ingeniero que diseña mecánicamente el robot para calcular los esfuerzos que debe soportar la estructura.

---

Basic Equations:	Algorithm:
$\mathbf{v}_0 = \mathbf{0}$	$\mathbf{v}_0 = \mathbf{0}$
$\mathbf{a}_0 = -\mathbf{a}_g$	$\mathbf{a}_0 = -\mathbf{a}_g$
$\mathbf{v}_i = \mathbf{v}_{\lambda(i)} + \mathbf{S}_i \dot{\mathbf{q}}_i$	<b>for</b> $i = 1$ to $N_B$ <b>do</b>
$\mathbf{a}_i = \mathbf{a}_{\lambda(i)} + \mathbf{S}_i \ddot{\mathbf{q}}_i + \dot{\mathbf{S}}_i \dot{\mathbf{q}}_i$	$[\mathbf{X}_J, \mathbf{S}_i, \mathbf{v}_J, \mathbf{c}_J] =$ jcalc(jtype( $i$ ), $\mathbf{q}_i, \dot{\mathbf{q}}_i$ )
$\mathbf{f}_i^B = \mathbf{I}_i \mathbf{a}_i + \mathbf{v}_i \times^* \mathbf{I}_i \mathbf{v}_i$	${}^i\mathbf{X}_{\lambda(i)} = \mathbf{X}_J \mathbf{X}_T(i)$
$\mathbf{f}_i = \mathbf{f}_i^B - \mathbf{f}_i^x + \sum_{j \in \mu(i)} \mathbf{f}_j$	<b>if</b> $\lambda(i) \neq 0$ <b>then</b>
$\boldsymbol{\tau}_i = \mathbf{S}_i^T \mathbf{f}_i$	${}^i\mathbf{X}_0 = {}^i\mathbf{X}_{\lambda(i)} {}^{\lambda(i)}\mathbf{X}_0$
<b>Equations in Body Coordinates:</b>	<b>end</b>
$\mathbf{v}_0 = \mathbf{0}$	$\mathbf{v}_i = {}^i\mathbf{X}_{\lambda(i)} \mathbf{v}_{\lambda(i)} + \mathbf{v}_J$
$\mathbf{a}_0 = -\mathbf{a}_g$	$\mathbf{a}_i = {}^i\mathbf{X}_{\lambda(i)} \mathbf{a}_{\lambda(i)} + \mathbf{S}_i \ddot{\mathbf{q}}_i$ + $\mathbf{c}_J + \mathbf{v}_i \times \mathbf{v}_J$
$\mathbf{v}_{Ji} = \mathbf{S}_i \dot{\mathbf{q}}_i$	$\mathbf{f}_i = \mathbf{I}_i \mathbf{a}_i + \mathbf{v}_i \times^* \mathbf{I}_i \mathbf{v}_i - {}^i\mathbf{X}_0^* \mathbf{f}_i^x$
$\mathbf{c}_{Ji} = \mathring{\mathbf{S}}_i \dot{\mathbf{q}}_i$	<b>end</b>
$\mathbf{v}_i = {}^i\mathbf{X}_{\lambda(i)} \mathbf{v}_{\lambda(i)} + \mathbf{v}_{Ji}$	<b>for</b> $i = N_B$ to 1 <b>do</b>
$\mathbf{a}_i = {}^i\mathbf{X}_{\lambda(i)} \mathbf{a}_{\lambda(i)} + \mathbf{S}_i \ddot{\mathbf{q}}_i + \mathbf{c}_{Ji} + \mathbf{v}_i \times \mathbf{v}_{Ji}$	$\boldsymbol{\tau}_i = \mathbf{S}_i^T \mathbf{f}_i$
$\mathbf{f}_i^B = \mathbf{I}_i \mathbf{a}_i + \mathbf{v}_i \times^* \mathbf{I}_i \mathbf{v}_i$	<b>if</b> $\lambda(i) \neq 0$ <b>then</b>
$\mathbf{f}_i = \mathbf{f}_i^B - {}^i\mathbf{X}_0^* \mathbf{f}_i^x + \sum_{j \in \mu(i)} {}^i\mathbf{X}_j^* \mathbf{f}_j$	$\mathbf{f}_{\lambda(i)} = \mathbf{f}_{\lambda(i)} + {}^{\lambda(i)}\mathbf{X}_i^* \mathbf{f}_i$
$\boldsymbol{\tau}_i = \mathbf{S}_i^T \mathbf{f}_i$	<b>end</b>

---

Figura 3.12: Algoritmo RNEA y sus ecuaciones

### 3.3.2. Dinámica directa: Composite Rigid Body Algorithm

El problema de la dinámica directa consiste en obtener las aceleraciones de un sistema de sólidos rígidos en función de las fuerzas aplicadas. Se suele usar mayoritariamente para simular movimientos de robots y otros sistemas. En su forma más genérica se puede expresar como:

$$\ddot{\mathbf{q}} = FD(modelo, q, \dot{q}, \boldsymbol{\tau})$$

donde  $q, \dot{q}, \ddot{q}$  y  $\boldsymbol{\tau}$  son los vectores generalizados de posición, velocidad, aceleración y par.

Existen dos enfoques del problema de la dinámica directa para una cadena cinemática:

1. Formular una ecuación de movimiento para el sistema completo y resolverlo para las variables de aceleración.
2. Propagar las restricciones de movimiento de un eslabón al siguiente de forma que se calcule la aceleración de una articulación cada vez.

El primer enfoque incluye calcular los elementos de la matriz de inercia del espacio articular, de tamaño  $n \times n$ , y luego resolver un conjunto de  $n$  ecuaciones lineales para las variables de aceleración. Este método se denomina Composite Rigid Body Algorithm (CRBA) y se va a discutir en este apartado.

El segundo punto de vista incluye realizar una serie de operaciones fijas para cada uno de los eslabones de la cadena cinemática, este algoritmo se llama Articulated Body Algorithm (ABA) y será tratado en el apartado siguiente.

Según la notación de (Featherstone, 2008), la ecuación de movimiento de una cadena cinemática se puede expresar de la siguiente forma:

$$\tau = H(q)\ddot{q} + C(q, \dot{q}, f^x) \quad (3.57)$$

donde  $q, \dot{q}, \ddot{q}$  y  $\tau$  son los vectores generalizados de posición, velocidad, aceleración y par,  $f^x$  es el vector de fuerzas externas,  $H$  es la matriz de inercia del espacio articular, y  $C$  la matriz de fuerza parcial.  $H$  y  $C$  son los coeficientes de la ecuación y  $\ddot{q}$  y  $\tau$  son las variables.

Dado  $\tau$ , el problema de la dinámica directa se puede calcular siguiendo el siguiente procedimiento:

1. Calcular  $C$ .
2. Calcular  $H$ .
3. Resolver  $H\ddot{q} = \tau - C$  para  $\ddot{q}$ .

Para obtener la matriz  $C$ , basta calcular la dinámica inversa para una aceleración cero:

$$C = ID(modelo, q, \dot{q}, 0, f^x)$$

Existen varias maneras de calcular la matriz  $H$ , sin embargo, la más eficiente es el algoritmo CRBA. A continuación se va a explicar el principio matemático que rige este algoritmo y se estudiará la obtención de la matriz  $H$ .

La energía cinética de una cadena cinemática (también llamado árbol cinemático) es la suma de las energías cinéticas de sus eslabones.

$$T = \frac{1}{2}v^T I v \quad (3.58)$$

dado que:

$$v = S\dot{q} \quad (3.59)$$

sustituyendo (3.59) en (3.58).

$$T = \frac{1}{2}(S\dot{q})^T I S\dot{q} = \frac{1}{2}\dot{q}^T S^T I S\dot{q} \quad (3.60)$$

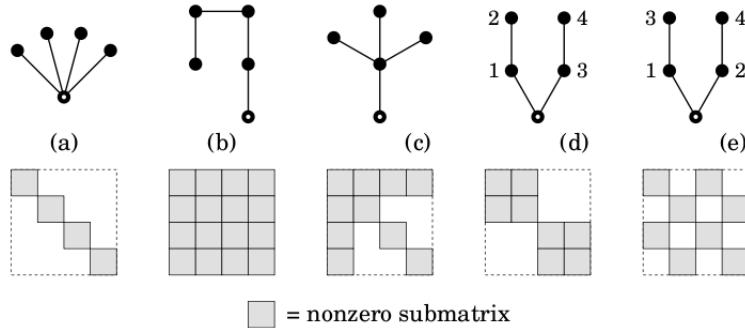
La matriz de inercia del espacio articular está relacionada con la energía cinética de la forma:

$$T = \frac{1}{2}\dot{q}^T H \dot{q} \quad (3.61)$$

y comparando las ecuaciones (3.60) y (3.61) se obtiene que:

$$H = S^T I S \quad (3.62)$$

El algoritmo CRBA implica que  $H$  depende de la conectividad de la cadena cinemática, o lo que es lo mismo, de cómo están conectadas las ramas del árbol cinemático. Esto implica que algunos términos de la matriz  $H$  serán automáticamente cero de acuerdo a su conectividad,  $H_{ij}$  será cero si  $i$  no es el antecesor de  $j$ , ni su predecesor, ni  $i$  es igual a  $j$ . Resumiendo, los términos serán cero cuando  $i$  y  $j$  estén en diferentes ramas del árbol (Ver Figura 3.13).



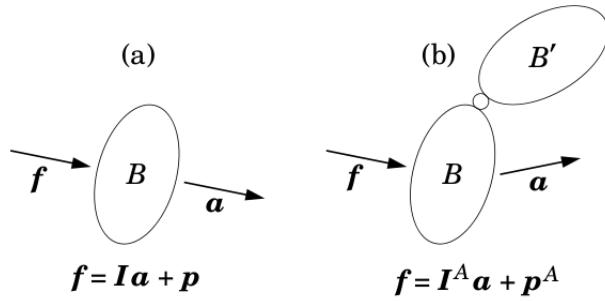
**Figura 3.13:** Términos no nulos de la matriz  $H$  dependiendo de la conectividad de la cadena cinemática.

Por último, como  $H$  es simétrica y definida positiva, tiene inversa, por lo que se puede despejar  $\ddot{q}$  de la ecuación (3.63).

$$H\ddot{q} = \tau - C \quad (3.63)$$

### 3.3.3. Dinámica directa: Articulated Body Algorithm

El problema de la dinámica directa presenta dos conjuntos de incógnitas: las aceleraciones y las fuerzas de restricción en las articulaciones. Usualmente no es posible obtener esas incógnitas de forma local, pero se pueden formular un conjunto de ecuaciones que las satisfagan. Los métodos de propagación trabajan calculando los coeficientes de estas ecuaciones localmente y los propagan a los cuerpos vecinos hasta llegar a un cuerpo en el que se pueda resolver la ecuación de forma local. Posteriormente, se puede resolver las ecuaciones para los cuerpos vecinos hasta obtener la solución para el sistema entero. Este es el principio fundamental en el que se basa el algoritmo Articulated Body Algorithm.



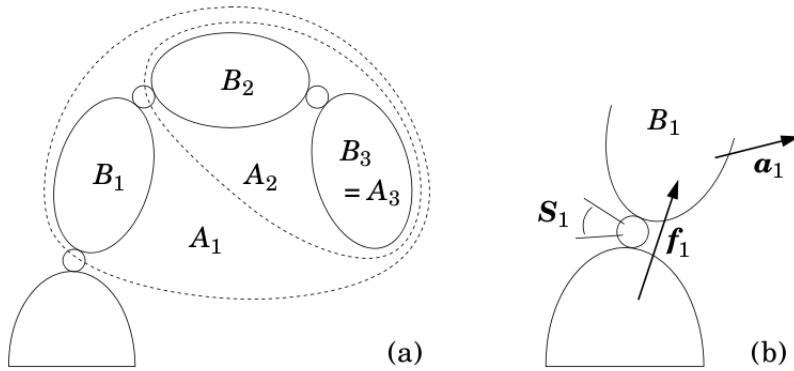
**Figura 3.14:** Comparación de la fuerza en un cuerpo rígido (a) con la de un cuerpo articulado (b)

La inercia de un cuerpo articulado es la inercia que aparece en el cuerpo cuando éste es parte de un sistema de cuerpos rígidos. En la Figura 3.14(a) se observa la relación entre la fuerza aplicada,  $f$ , y la aceleración,  $a$ . Si ahora se considera un sistema compuesto por dos cuerpos unidos por una articulación (Figura 3.14(b)), la misma fuerza produce una aceleración diferente, por lo que la relación entre  $f$  y  $a$  es:

$$f = I^A\mathbf{a} + \mathbf{p}^A \quad (3.64)$$

La ecuación (3.64) es la denominada *ecuación de movimiento del cuerpo articulado*. Las matrices  $I^A$  y  $\mathbf{p}^A$  son la inercia del cuerpo articulado y la fuerza parcial del cuerpo articulado del cuerpo  $B$ , respectivamente, en el sistema articulado de cuerpos  $B$  y  $B'$ .

A continuación se va a explicar el algoritmo ABA. Sea una cadena cinemática que contiene  $N_B$  cuerpos. Se puede definir un conjunto de cuerpos articulados  $A_1 \dots A_{N_B}$  tales que  $A_i$  contiene todos los cuerpos y articulaciones desde la articulación  $i$  (Ver Figura 3.15(a)).



**Figura 3.15:** Definición de cuerpos articulados en ABA (a) y solución de la aceleración para la articulación 1 (b)

Si se considera el movimiento de \$B\_1\$, como se observa en la Figura 3.15(b), \$f\_1\$ es la fuerza transmitida a través de la articulación 1, luego la ecuación de movimiento es:

$$f_1 = I_1^A \mathbf{a}_1 + p_1^A \quad (3.65)$$

La aceleración del cuerpo \$B\_1\$ debe satisfacer la ecuación:

$$\mathbf{a}_1 = \mathbf{a}_0 + \mathbf{c}_1 + S_1 \ddot{\mathbf{q}}_1 \quad (3.66)$$

donde \$\mathbf{a}\_0\$ es la aceleración de la base y \$\mathbf{c}\_1 = \mathbf{v}\_i \times S\_i \dot{\mathbf{q}}\_i\$. Además, se debe satisfacer esta ecuación:

$$\tau_1 = S_1^T f_1 \quad (3.67)$$

Combinando las ecuaciones (3.65), (3.66) y (3.67) se obtiene:

$$\tau_1 = S_1^T (I_1^A (\mathbf{a}_0 + \mathbf{c}_1 + S_1 \ddot{\mathbf{q}}_1) + p_1^A) \quad (3.68)$$

de donde se puede extraer la ecuación de la aceleración \$\ddot{\mathbf{q}}\_1\$.

$$\ddot{\mathbf{q}}_1 = (S_1^T I_1^A S_1)^{-1} (\tau_1 - S_1^T I_1^A (\mathbf{a}_0 + \mathbf{c}_1) - S_1^T p_1^A) \quad (3.69)$$

De la ecuación (3.69) se extrae que conociendo \$I\_1^A\$ y \$p\_1^A\$ es posible calcular directamente \$\ddot{\mathbf{q}}\_1\$, sin la necesidad del cálculo de otras aceleraciones. Esto es posible

debido a que la ecuación (3.65) ya tiene en cuenta los efectos dinámicos de todo el conjunto de cuerpos en  $A_1$ . Una vez se ha calculado  $\ddot{q}_1$ , se puede calcular la aceleración del siguiente cuerpo simplemente sustituyendo  $a_0 = \ddot{q}_1$  y realizando el proceso recursivamente para cada eslabón hijo.

Finalmente, para calcular el algoritmo ABA, es necesario conocer la matriz de inercia del cuerpo articulado  $I_1^A$  y la fuerza parcial  $p_1^A$ , éstas se calculan partiendo de los componentes de los cuerpos articulados posteriores:

$$I_i^A = \sum_{j \in A_i} I_j \quad (3.70)$$

y

$$p_i^A = \sum_{j \in A_i} p_j \quad (3.71)$$

Capítulo **4**

# Locomoción de un robot humanoide

## 4.1. Introducción

El desarrollo de robot móviles bípedos ha despertado un gran interés en los últimos años debido al tipo de aplicaciones y ventajas que estos sistemas poseen. En un entorno adaptado a las necesidades humanas, el robot humanoide se posiciona como el candidato ideal para ofrecer asistencia al ser humano en las tareas más tediosas o peligrosas.

Aunque se ha avanzado bastante en cuanto a lo robótica móvil bípeda, aún existen muchos problemas abiertos. Los robots más avanzados con los que se cuenta actualmente presentan movimientos torpes o lentos comparados con los seres humanos, así como poco eficientes desde el punto de vista energético. Algunos de los problemas aún abiertos en locomoción humanoide corresponden a problemas del tipo dinámico, como lo son el balanceo del centro de masa, compensación de fuerzas dinámicas durante la caminata, etc. La manera de mejorar

estos sistemas no sólo depende de técnicas de control y generación de trayectorias, sino de análisis y metodologías de locomoción basadas en sistemas biológicos.

Los sistemas biológicos bípedos han evolucionado de manera natural por milenios, mejorando y optimizando sus formas de locomoción. Es por tanto lógico pensar que a través del estudio y análisis de la locomoción bípeda en seres humanos, se podrán obtener algoritmos de control que, implementados en robots humanoides, imiten de la forma más perfecta posible la caminata humana.

## **4.2. Análisis de la locomoción bípeda**

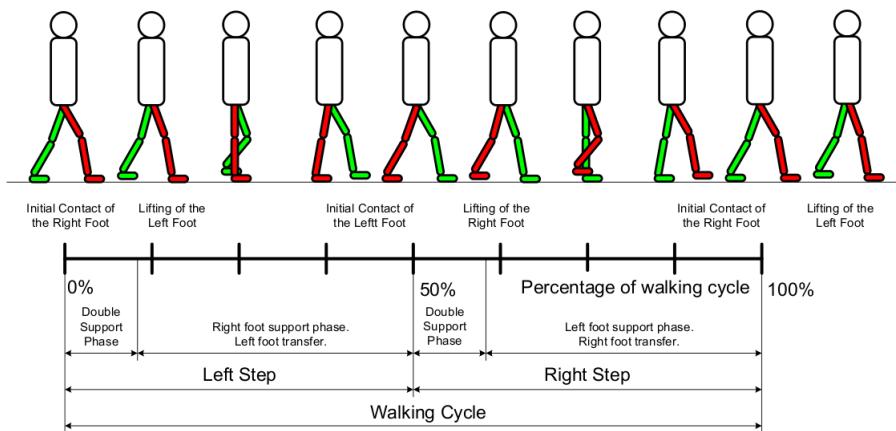
Las ecuaciones del movimiento para la locomoción bípeda de un humanoide son muy complejas, hay que tener en cuenta las ecuaciones de todos los grados de libertad de las cadenas cinemáticas del robot y añadir las propias del movimiento de la base libre del mecanismo (el tronco del humanoide), las que se derivan del mantenimiento de la estabilidad necesaria durante la locomoción y las dificultades que introduce el contacto de los pies con el suelo, que producen cadenas cinemáticas cerradas. El resultado es un sistema algebraico diferencial variable en el tiempo.

Una de las características a tener en cuenta en el análisis de la locomoción bípeda, es la periodicidad del movimiento, cuyo periodo es un paso: esto supone que la posición y velocidad al principio y al final de cada paso deben ser las mismas.

Otro detalle fundamental de la caminata bípeda es el cambio de posición continuo del pie de apoyo. Existen dos estados en el proceso de locomoción, la fase de doble soporte, que supone alrededor del 20 % del tiempo del ciclo y la fase de simple soporte, que supone el 80 % del tiempo del ciclo. En la primera

situación el movimiento es estable, sin embargo, en la segunda no lo es. Se produce una situación estáticamente inestable cuando un pie está en el suelo y el otro pasa de una posición retrasada a una adelantada produciendo aceleraciones que afectan al mecanismo.

En la Figura 4.1 se muestra un esquema del ciclo de locomoción de un bípedo. El ciclo está dividido en dos fases, la fase de paso derecho y la fase de paso izquierdo. El paso izquierdo comienza cuando el pie derecho toca el suelo. En ese momento se transmite el peso del cuerpo de un pie a otro, produciéndose aceleraciones laterales que deben ser controladas mediante algoritmos de estabilización. A continuación, se produce el balanceo del pie izquierdo en el estado de soporte simple, es decir, con un sólo pie apoyado en el suelo. El peso se transmite a este pie llegándose al estado de doble soporte, momento en el que los dos pies tocan el suelo simultáneamente. Finalmente se repite el ciclo para el pie contrario.



**Figura 4.1:** Ciclo del movimiento

### 4.3. Criterio de estabilidad

Para lograr una caminata estable, es necesario satisfacer algún criterio de estabilidad en la planificación de trayectorias, por lo que el método ZMP (Zero Moment Point) y sus variantes, proporcionan un poderoso enfoque para determinar trayectorias en la locomoción bípeda. El concepto de ZMP (Vukobratovic y Borovac, 2004), es uno de los métodos más utilizados tanto por su simplicidad como por su efectividad.

El ZMP se define como el punto con respecto al cual el momento creado por las fuerzas de inercia y de gravedad no tiene componentes en el eje horizontal (Dasgupta y Nakamura, 1999). Esta teoría asume que la superficie de contacto es plana y que el pie no desliza. Es por lo tanto un indicador de la estabilidad, si el ZMP se encuentra en el polígono de soporte el movimiento es estable, en caso contrario, es inestable (Figura 4.2).

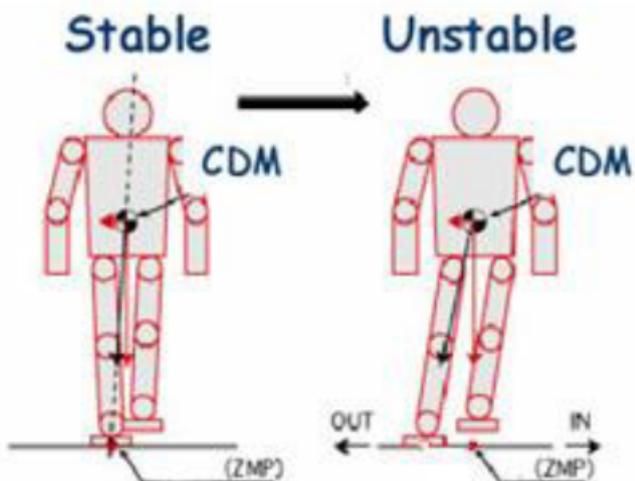
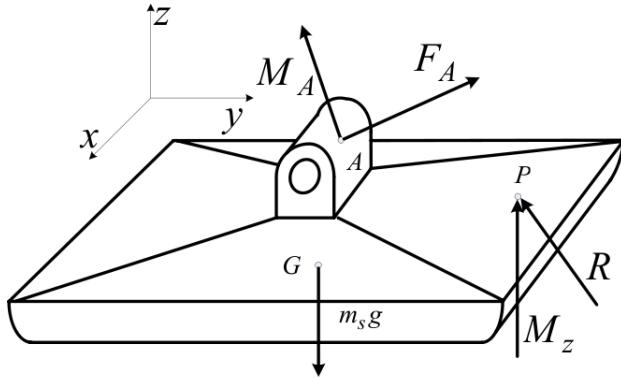


Figura 4.2: Estabilidad de la locomoción bípeda



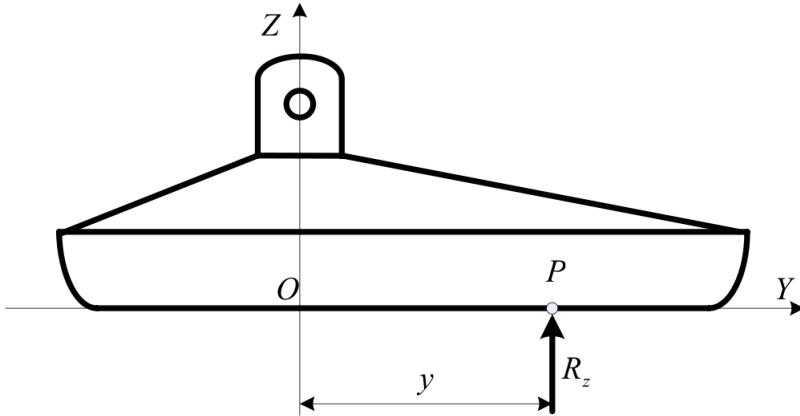
**Figura 4.3:** Fuerzas que actúan en el pie de un humanoide

Para computar la posición del ZMP es necesario calcular un equilibrio de fuerzas en el pie de soporte (Figura 4.3). La proyección de estas ecuaciones en el plano horizontal es la base de la computación del punto  $P$ , que es el punto de aplicación de la fuerza de reacción con el suelo  $R$ .

$$R + F_A + m_sg = 0 \quad (4.1)$$

$$\overline{OP} \times R + \overline{OG} \times m_sg + M_A + M_z + \overline{OA} \times F_A = 0 \quad (4.2)$$

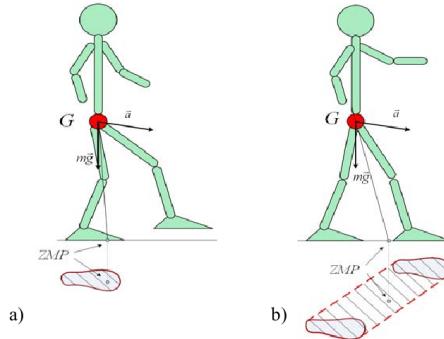
En general, la fuerza y momento resultante de reacción con el suelo tiene tres componentes  $R(R_x, R_y, R_z)$  y  $M(M_x, M_y, M_z)$ . Suponiendo que no hay deslizamiento, los componentes horizontales de la fuerza de reacción,  $R_x$  y  $R_y$  se equilibrarán con las componentes horizontales de  $F_A$ . De la misma forma, el momento de las fuerzas de reacción verticales  $M_z$  se equilibrará con la componente vertical del momento que actúa en el cuerpo  $M_A$  y el momento creado por las componentes horizontales de  $F_A$ .  $R_z$  es la fuerza que equilibra las fuerzas verticales del cuerpo.



**Figura 4.4:** Compensación del ZMP

Para compensar las componentes horizontales de  $M_A$  producidas por un movimiento o una carga adicional, es necesario cambiar el punto de aplicación de la fuerza de reacción  $R_z$  (Figura 4.4). Si este punto está dentro del área cubierta por la planta del pie, entonces el sistema está en equilibrio ya que las componentes horizontales del momento de reacción se anulan,  $M_X = 0$  y  $M_Y = 0$ . Según el momento  $M_{Ax}$  va aumentando, el punto de aplicación de la fuerza  $R_z$  se va acercando al borde del pie. Si es punto sale fuera del borde, aparecerán momentos  $M_X \neq 0$  y  $M_Y \neq 0$ , lo que ocasionará que el mecanismo rote alrededor del borde del pie.

Por lo tanto, se define el ZMP como el punto  $P_{ZMP} = (x_{ZMP}, y_{ZMP}, 0)$  respecto del cual el momento de la fuerza de reacción no tiene componentes horizontales. Esto se puede entender como un criterio de estabilidad, ya que cuando el ZMP está dentro del polígono de soporte, el movimiento es estable, en caso contrario no lo es. En la Figura 4.5 se representan los polígonos de soporte, tanto de la fase de apoyo simple como de la fase de apoyo doble. Como se puede observar, en la fase de apoyo simple el polígono de soporte es mucho menor que en el caso de la fase de apoyo doble, es por ello que allí se producirán las mayores inestabilidades.

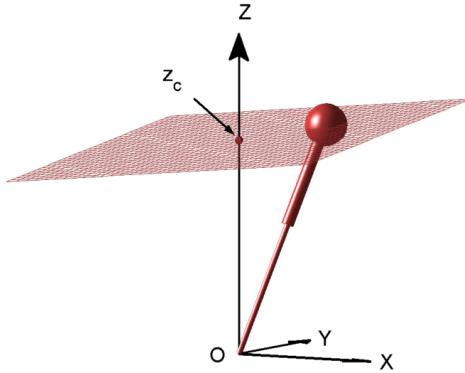


**Figura 4.5:** Región de estabilidad a) en soporte simple, b) en doble soporte

Para poder calcular el ZMP es necesario conocer cada una de las fuerzas y momentos que actúan en la estructura en tiempo real, así como las fuerzas y momentos de reacción. La computación de todas estas fuerzas y momentos es lenta, debido al alto número de grados de libertad que poseen las plataformas humanoides, por ello, se han utilizado diversos modelos simplificados que permiten generar trayectorias en tiempo real, un ejemplo es el modelo del péndulo invertido, del que se hablará a continuación.

#### 4.4. Modelo del péndulo invertido

Este método para generar trayectorias de movimiento, se basa en un conocimiento limitado de la localización del centro total de masas del humanoide y del momento angular. El bípedo se representa como un péndulo invertido plano cuya base simula la articulación del tobillo del robot, calculando la trayectoria en el plano frontal. Para generación de trayectorias 3D este método es conocido como Three-Dimensional Linear Inverted Pendulum Mode (3D-LIPM) (Kanehiro y cols., 2001).



**Figura 4.6:** Peñdulo invertido con restricción en el eje Z

Cuando un bípedo está, dentro de su ciclo de locomoción, en la fase de soporte simple, su dinámica se puede representar como un péndulo invertido, cuyo extremo representa la masa total del robot y su centro de gravedad y que posee una barra telescópica sin masa, que puede variar su longitud (Figura 4.6). Suponiendo que el péndulo se mueve sobre el plano que intersecta  $z_c$ :

$$z = k_x x + k_y y + z_c \quad (4.3)$$

En el caso de que el robot se mueva en terreno plano ( $k_x = k_y = 0$ ), la ecuación dinámica del péndulo se puede expresar como

$$\ddot{y} = \frac{g}{z_c} y - \frac{1}{mz_c} \tau_x \quad (4.4)$$

$$\ddot{x} = \frac{g}{z_c} x + \frac{1}{mz_c} \tau_y \quad (4.5)$$

donde  $m$  es la masa del péndulo,  $g$  es la aceleración de la gravedad y  $\tau_x, \tau_y$  es el par respectivo del eje  $X$  e  $Y$  respectivamente.

Para el 3D-LIMP se puede obtener el ZMP como

$$x_{ZMP} = -\frac{\tau_y}{mg} \quad (4.6)$$

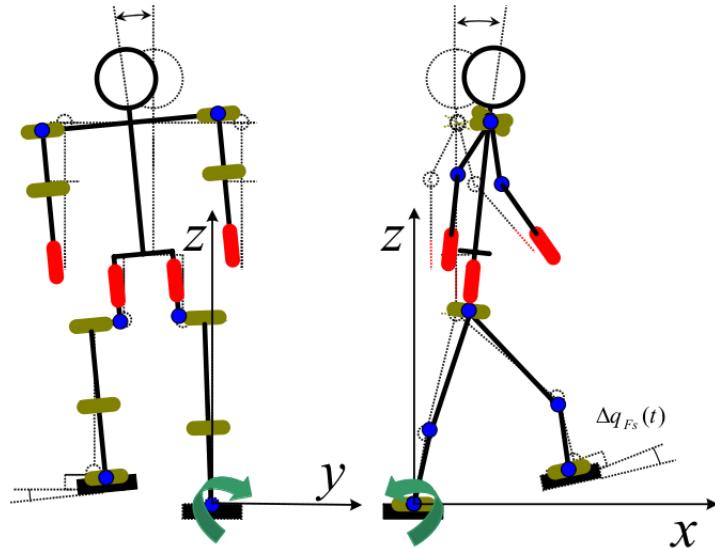
$$y_{ZMP} = \frac{\tau_x}{mg} \quad (4.7)$$

donde  $x_{ZMP}$  y  $y_{ZMP}$  es la posición del ZMP sobre el suelo. Sustituyendo las ecuaciones (4.6) y (4.7) en (4.4) y (4.5) se obtienen las denominadas *ecuaciones ZMP*.

$$x_{ZMP} = x - \frac{z_c \ddot{x}}{g} \quad (4.8)$$

$$y_{ZMP} = y - \frac{z_c \ddot{y}}{g} \quad (4.9)$$

El modelo del péndulo invertido, permite relacionar el ZMP con el centro de gravedad (COG) del robot. Estas son las condiciones que permiten la generación de trayectorias de movimiento estables para robots bípedos.



**Figura 4.7:** Control de estabilidad de un robot humanoide

## 4.5. Control de estabilidad

Una aplicación directa del modelo del péndulo invertido es el control de la estabilidad del humanoide (Kaynov, 2008). Mediante un doble péndulo invertido, cuyo primer eslabón representa la pierna de apoyo y cuyo segundo eslabón representa el tronco del robot, se puede controlar la posición del tobillo y de la cadera para evitar que se produzcan inestabilidades tanto en el plano frontal como en el sagital (Figura 4.7).

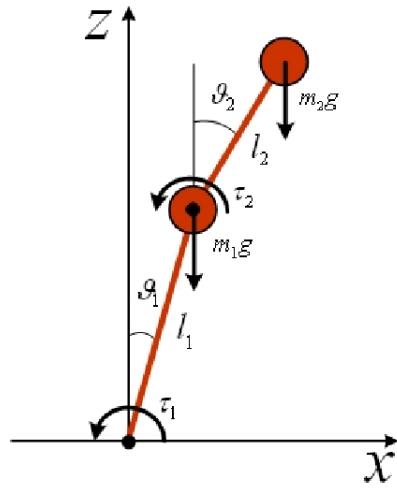
Un doble péndulo (Figura 4.8) consiste en un péndulo unido a otro. Se consideran dos masas puntuales,  $m_1$ , situada en la cadera y  $m_2$ , situada en el centro de masas del robot, cuya suma es la masa total del robot y que están unidas entre sí por una cuerda sin masa de longitud  $l_1$  y  $l_2$ . Teniendo en cuenta el ángulo con el tobillo,  $\theta_1$ , y el ángulo de la cadera,  $\theta_2$ , se puede obtener la posición de las dos masas como:

$$x_1 = l_1 \sin \theta_1 \Rightarrow \dot{x}_1 = l_1 \cos \theta_1 \dot{\theta}_1 \quad (4.10)$$

$$z_1 = l_1 \cos \theta_1 \Rightarrow \dot{z}_1 = -l_1 \sin \theta_1 \dot{\theta}_1 \quad (4.11)$$

$$x_2 = l_1 \sin \theta_1 + l_2 \sin \theta_2 \Rightarrow \dot{x}_2 = l_1 \cos \theta_1 \dot{\theta}_1 + l_2 \cos \theta_2 \dot{\theta}_2 \quad (4.12)$$

$$z_2 = l_1 \cos \theta_1 + l_2 \cos \theta_2 \Rightarrow \dot{z}_2 = -l_1 \sin \theta_1 \dot{\theta}_1 - l_2 \sin \theta_2 \dot{\theta}_2 \quad (4.13)$$



**Figura 4.8:** Doble péndulo invertido

Sabiendo que la energía potencial y cinética del sistema es:

$$V = m_1gz_1 + m_2gz_2 \quad (4.14)$$

$$T = \frac{1}{2}m_1v_1^2 + \frac{1}{2}m_2v_2^2 \quad (4.15)$$

y sabiendo que el Lagrangiano se define como la diferencia entre energía cinética y potencial.

$$L = T - V \quad (4.16)$$

sustituyendo las ecuaciones (4.10),(4.11),(4.12),(4.13),(4.14) y (4.15) en la ecuación del Lagrangiano (4.16) se obtiene:

$$\begin{aligned} L = & \frac{1}{2}m_1l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2l_2^2\dot{\theta}_2^2 + m_2l_1l_2\dot{\theta}_1\dot{\theta}_2 \cos(\theta_1 - \theta_2) - \\ & - (m_1 + m_2)gl_1 \cos \theta_1 + m_2gl_2 \cos \theta_2 \end{aligned} \quad (4.17)$$

A partir del Lagrangiano se pueden obtener los pares generados en el movimiento.

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}_i} \right) - \frac{\partial L}{\partial \theta_i} = \tau_i \quad (4.18)$$

donde  $\theta_i$  son las coordenadas generalizadas del sistema y  $\tau_i$  son los pares aplicados.

Derivando(4.18) y sustituyendo para  $\theta_1$  y  $\theta_2$  se obtienen los pares articulares.

$$\begin{aligned}\tau_1 = & (m_1 + m_2)l_1^2\ddot{\theta}_1 + m_2l_1l_2\ddot{\theta}_2 \cos(\theta_1 - \theta_2) + \\ & + m_2l_1l_2\dot{\theta}_2^2 \sin(\theta_1 - \theta_2) - (m_1 + m_2)gl_1 \sin \theta_1 \quad (4.19)\end{aligned}$$

$$\begin{aligned}\tau_2 = & m_2l_2^2\ddot{\theta}_2 + m_2l_1l_2\ddot{\theta}_1 \cos(\theta_1 - \theta_2) - \\ & - m_2l_1l_2\dot{\theta}_1^2 \sin(\theta_1 - \theta_2) - m_2gl_2 \sin \theta_2 \quad (4.20)\end{aligned}$$

Para estabilizar el robot se realiza un desacople entre la parte superior del péndulo y la inferior, de forma que se controle el robot como dos péndulos invertidos, uno para el tobillo que controla el ZMP, y otro para el tronco, que controla la postura del robot. Implementando un controlador LQR se obtiene un estabilizador para el humanoide (Kaynov, Souères, Pierro, y Balaguer, 2009).

Como ya se ha explicado anteriormente, tanto el modelo del doble péndulo invertido como otros modelos reducidos se basan en un conocimiento limitado de la dinámica del robot para conseguir un control computacionalmente rápido del mismo. En esta Tesis de Máster, se busca validar estos modelos comparándolos con la dinámica de un modelo completo del robot. Para ello, en el capítulo 6 se van a comparar las fuerzas que aparecen en cada uno de los eslabones, tanto del doble péndulo como del modelo completo y se estudiarán las diferencias entre ambos.

Capítulo **5**

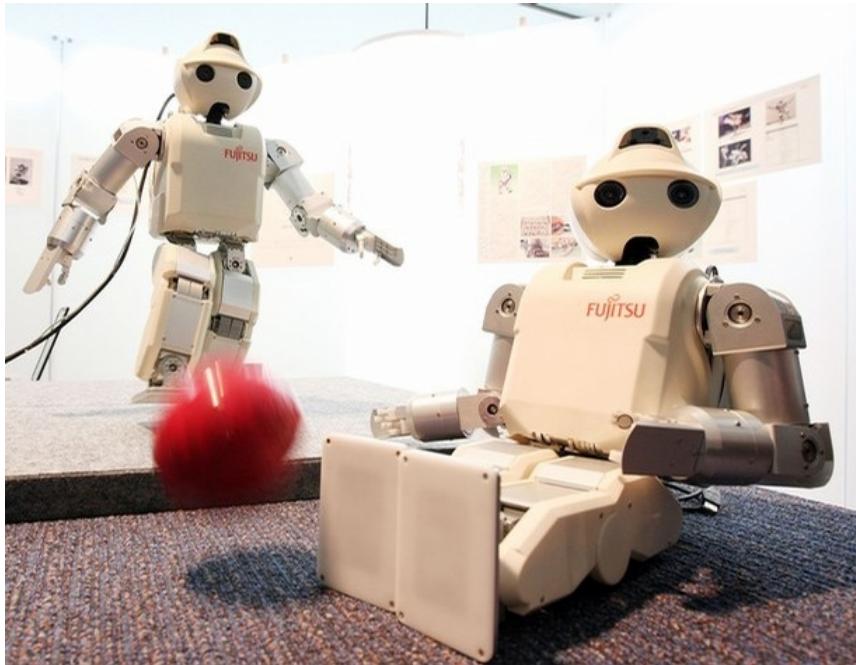
# Modelo dinámico del robot humanoide HOAP-3

## 5.1. Introducción

En este trabajo se ha obtenido el modelo dinámico completo del robot humanoide HOAP-3. Para la construcción de este modelo , las masas e inercias de cada uno de los eslabones se han obtenido del manual que proporciona el fabricante.

## 5.2. El robot humanoide HOAP-3

La plataforma usada para la obtención del modelo dinámico en esta Tesis de Máster es el robot humanoide HOAP-3 (Figura 5.1). Se trata de un humanoide de tamaño y peso medio, 60 cm y 9 kg aproximadamente, diseñado y fabricado por Fujitsu. HOAP son las siglas de “Humanoid for Open Architecture Platform” y este modelo es la evolución de otros anteriores, llamados HOAP y HOAP-2.



**Figura 5.1:** Robot humanoide HOAP-3

El robot HOAP-3 posee 28 grados de libertad distribuidos como se puede observar en la Figura 5.2. Posee 6 grados de libertad en cada pierna, 6 en cada brazo, 3 en la cabeza y 1 en la cadera. Aunque posee 28 grados de libertad, dispone solamente de 23 motores. Los 21 primeros controlan piernas y brazos según se muestra en la Figura 5.3, de estos motores el fabricante no da las especificaciones técnicas aunque sí proporciona una tabla con los pares nominales y máximos que puede soportar cada motor. Estos motores disponen de encoders relativos y existe la posibilidad de controlarlos en posición y en velocidad.

Los otros dos motores, el 22 y 23, no disponen de encoder como el resto. El motor 22 es el encargado de controlar los 3 grados de libertad de la cabeza (pitch, roll y yaw). El motor 23 controla la rotación de las dos manos y el agarre de las manos.

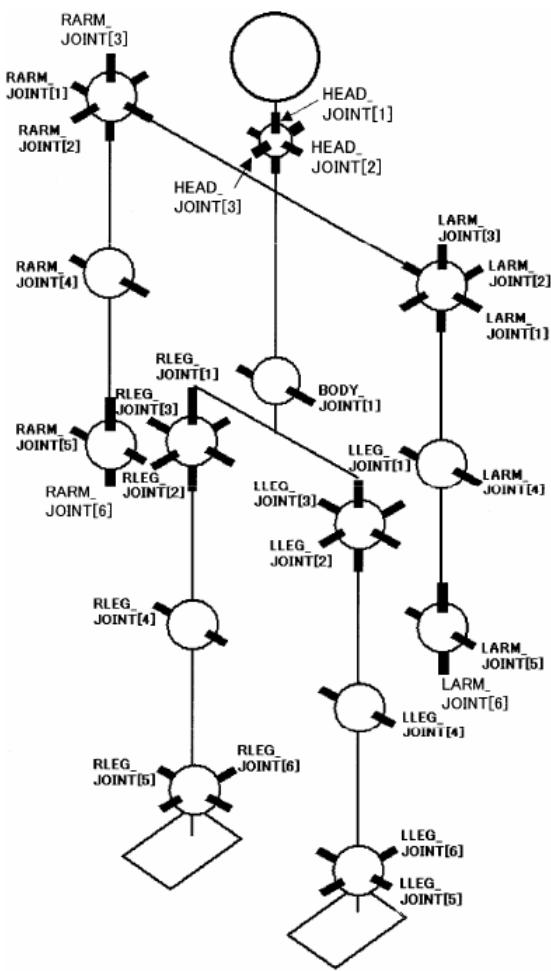


Figura 5.2: Grados de libertad del robot HOAP-3

Joint name	Flexibility	Device ID
HEAD_JOINT[1]	Head torsion	22
HEAD_JOINT[2]	Head pitch	22
HEAD_JOINT[3]	Head roll	22
BODY_JOINT[1]	Waist pitch	21
RLEG_JOINT[1]	Right hip joint torsion	1
RLEG_JOINT[2]	Right hip joint roll	2
RLEG_JOINT[3]	Right hip joint pitch	3
RLEG_JOINT[4]	Right knee	4
RLEG_JOINT[5]	Right Ankle pitch	5
RLEG_JOINT[6]	Right Ankle roll	6
RARM_JOINT[1]	Right shoulder Pitch	7
RARM_JOINT[2]	Right shoulder roll	8
RARM_JOINT[3]	Right shoulder torsion	9
RARM_JOINT[4]	Right elbow	10
RARM_JOINT[5]	Right fingers open/close	23
RARM_JOINT[6]	Right hand torsion	23
LLEG_JOINT[1]	Left hip joint torsion	11
LLEG_JOINT[2]	Left hip joint roll	12
LLEG_JOINT[3]	Left hip joint pitch	13
LLEG_JOINT[4]	Left knee	14
LLEG_JOINT[5]	Left Ankle pitch	15
LLEG_JOINT[6]	Left Ankle roll	16
LARM_JOINT[1]	Left shoulder Pitch	17
LARM_JOINT[2]	Left shoulder roll	18
LARM_JOINT[3]	Left shoulder torsion	19
LARM_JOINT[4]	Left elbow	20
LARM_JOINT[5]	Left fingers open/close	23
LARM_JOINT[6]	Left hand torsion	23

**Figura 5.3:** Articulaciones del HOAP y motor al que hace referencia

El robot lleva incorporado un PC-104 embebido en su interior, exactamente en la parte de atrás (Figura 5.4). Se trata de un Pentium de 1.1 Ghz con 512 Mb de RAM y una memoria Compact Flash de 1 Gb. Posee conexión Wifi IEEE802.11g y 4 puertos USB.

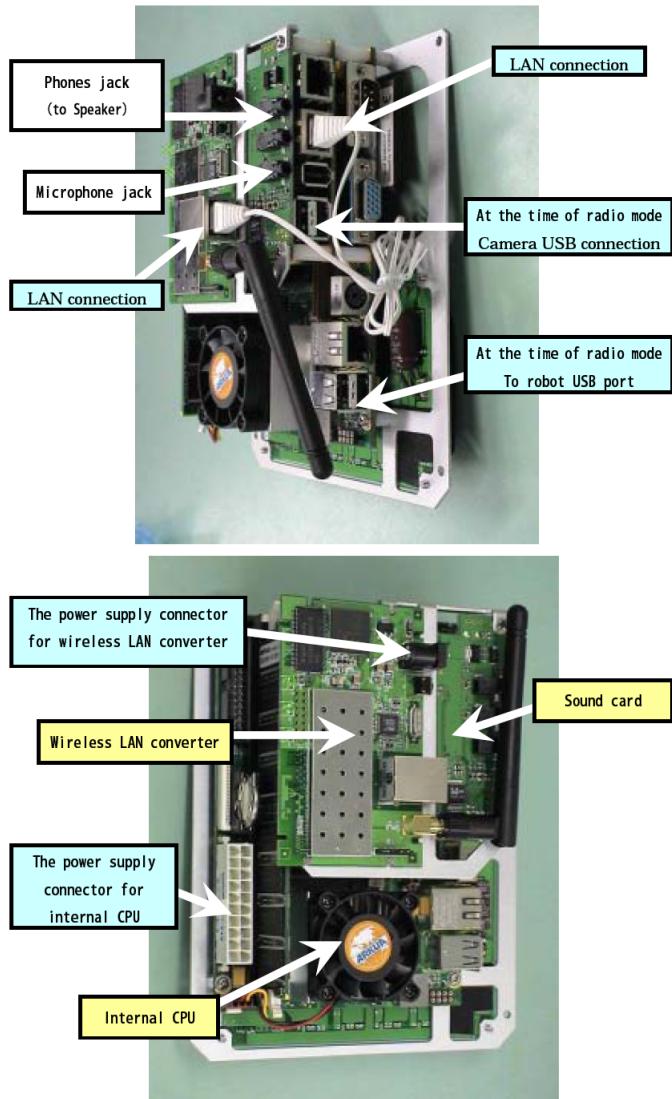
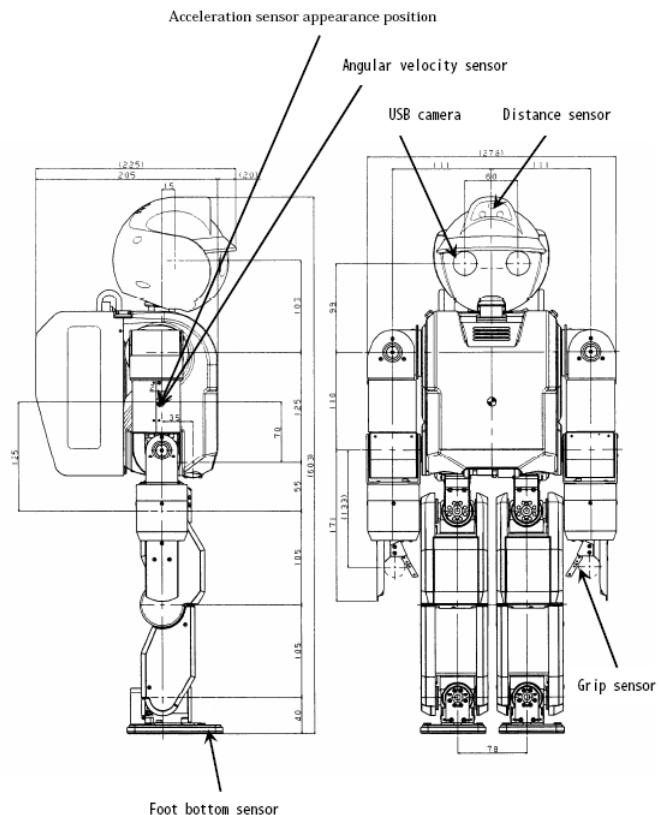


Figura 5.4: PC embebido

El sistema operativo que incorpora el robot es un Linux en tiempo real, basado en Fedora Core 1 con el kernel 2.4. Es importante destacar que sin el parche de tiempo real no se pueden controlar los servos del robot.

Con el robot, el fabricante proporciona un ordenador que es un clon exacto del que lleva incorporado el HOAP, con el mismo hardware y software instalado. Para controlar al robot existen dos métodos de conexión, el primero es a través de una conexión inalámbrica, por telnet, gracias a un router que también proporciona el fabricante, la segunda opción es una conexión directa por medio de un cable usb que conecta el pc externo directamente con los motores y los drivers del robot.

Además, el robot se puede alimentar directamente por cable o mediante una batería de NiMH de 24V, esto último ofrece la ventaja de una gran versatilidad en el movimiento.



**Figura 5.5:** Dimensiones y sensores del HOAP-3

Para completar las funcionalidades de este humanoide, se han añadido un conjunto de sensores que le permiten desenvolverse con naturalidad en cualquier entorno. En la Figura 5.5 se observan las dimensiones del robot en milímetros junto con algunos sensores que incorpora. El robot dispone de dos cámaras para visión estereoscópica, un micrófono, un altavoz, sensores infrarrojos de distancia, sensores de fuerza FSR en pies y manos y giróscopos y acelerómetros en los 3 ejes.

No cabe duda que el humanoide HOAP-3 es una plataforma muy completa y versátil, que reúne todas las características necesarias para desarrollar cualquier tipo de investigación en robótica. De hecho, en la actualidad, existen multitud de centros de investigación, tanto en Europa como en el resto del mundo, trabajando con este robot.

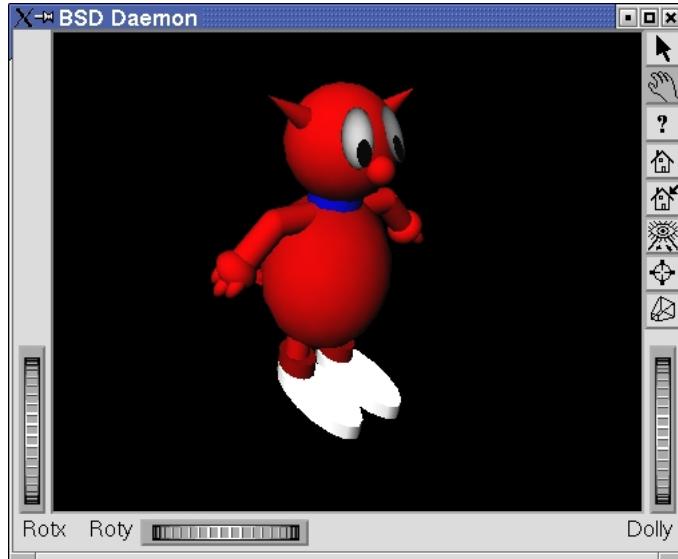
### 5.3. Herramienta de simulación

Para las simulaciones realizadas en esta Tesis de Máster se ha utilizado OpenInventor<sup>1</sup>. OpenInventor es un conjunto de librerías gráficas escritas en C++ que están basadas en OpenGL. Permiten manejar una amplia gama de objetos 3D como cubos, polígonos, materiales, cámaras, luces o textos y correr simulaciones con ellos. Para dibujar estos objetos y crear una escena OpenInventor utiliza una interfaz muy sencilla llamada SceneViewer. En la Figura 5.6 se muestra un ejemplo de una simulación en esta interfaz.

El hecho de que OpenInventor sea de código abierto y que se programe en C++, proporciona al programador gran libertad de acción y lo convierte en un candidato muy interesante para simular aplicaciones de robótica.

---

<sup>1</sup><http://oss.sgi.com/projects/inventor/>



**Figura 5.6:** Ejemplo de Open Inventor

Uno de los objetivos principales de cualquier simulador es su utilización como plataforma de prueba de algoritmos antes de implementarlos en el robot real, básicamente para prevenir accidentes desagradables. Es por ello que se busca siempre que el simulador se comporte de la forma más parecida posible al robot real.

Siguiendo esta filosofía, para el posicionamiento de cada uno de los eslabones del robot se han utilizado los ejes que aparecen en la Figura 5.7. Este conjunto de sistemas coordenados obedecen a las reglas de Denavit-Hartenberg (Barrientos y cols., 2007), que mediante 4 parámetros, permiten obtener la posición del end-effector del manipulador en función de las coordenadas de la base. Cada articulación gira en torno a su eje Z, estableciéndose el sentido de giro positivo mediante la regla de la mano derecha.

El objetivo que se persigue es poder introducir en el simulador las mismas

trayectorias que se introducen en el robot real, de esta forma el paso del simulador al robot real es inmediato y no es necesario adaptar los valores cuando se produce el cambio de uno a otro. En la Figura 5.8 se observa un fotograma del modelo completo del robot con todas las articulaciones a cero. Como se puede observar la Figura 5.7 y la Figura 5.8 muestran el robot en la misma posición, esto es debido la forma en que se han obtenido los parámetros de Denavit-Hartenberg.

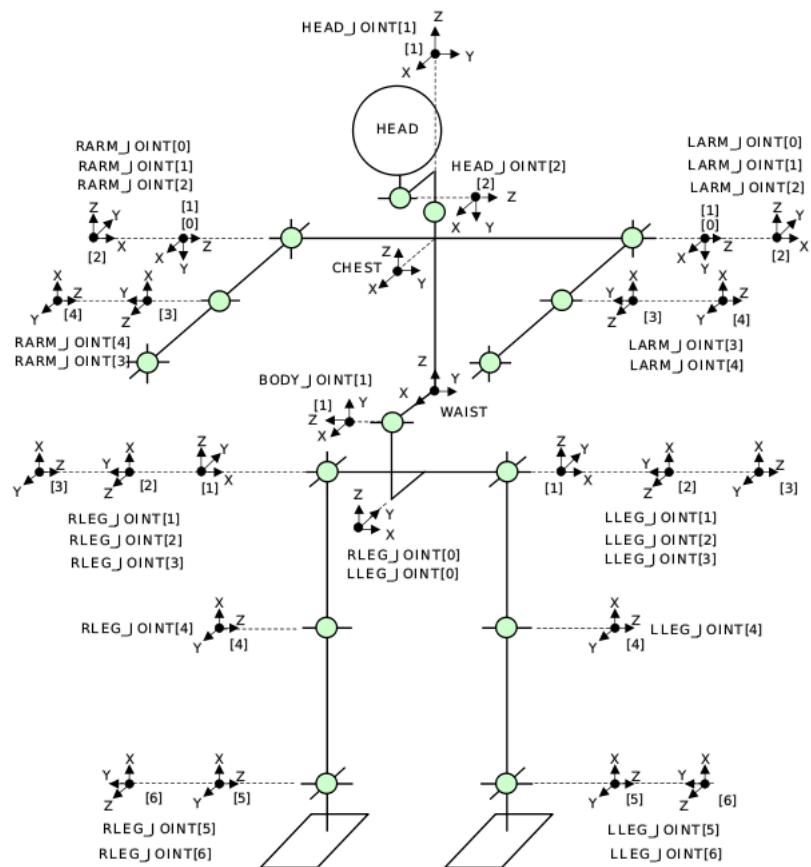
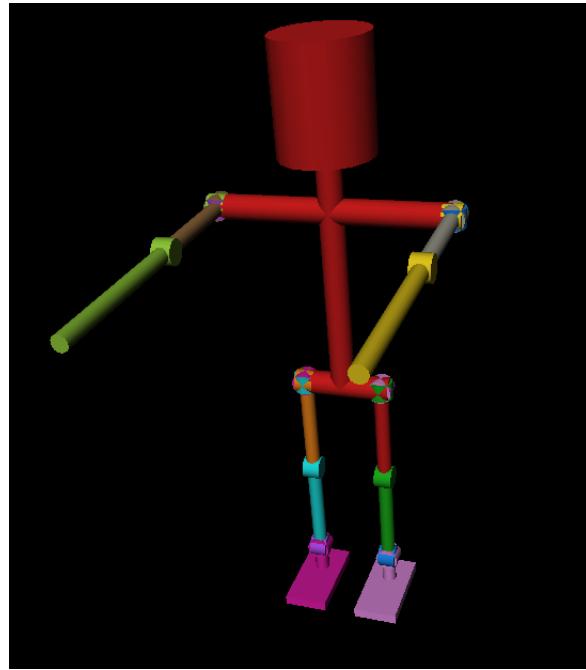


Figura 5.7: Parámetros de Denavit-Hartenberg



**Figura 5.8:** Modelo 3D del HOAP con las articulaciones a cero.

#### 5.4. Librerías dinámicas

Para la simulación y obtención de la dinámica del humanoide HOAP-3 se han utilizado un conjunto de librerías creadas por R. Featherstone<sup>2</sup>, que proporcionan un punto de partida muy útil para la programación de simulaciones complejas.

Las librerías, programadas en MATLAB, disponen de una amplia gama de algoritmos dinámicos, todos ellos basados en el enfoque de vectores espaciales. A continuación se citan algunos ejemplos:

- Matriz de transformación de coordenadas espaciales ( $Xrotx$ ,  $Xroty$ ,  $Xrotz$ ,  $Xtrans$ ).

---

<sup>2</sup><http://users.cecs.anu.edu.au/roy/spatial/documentation.html>

- Producto escalar para vectores de movimiento y de fuerza ( $crm, crf$ ).
- Dinámica inversa por el método RNEA ( $ID$ )
- Dinámica directa por el método CRBA y ABA ( $FDcrb, FDab$ )
- Cálculo de la ecuación de movimiento ( $HandC$ )
- Simulación de robots mediante OpenInventor(*drawmodel*)

La función *drawmodel* merece una mención especial ya que es la que permite que las simulaciones se lleven a cabo desde MATLAB. Esta función es realmente un script que enlaza el código de MATLAB con las librerías OpenInventor escritas en C++ y permite visualizar el modelo del robot y dotarlo de movimiento.

Los argumentos de esta función son tres: *drawmodel* (*model, delta\_t, joint\_vals*). El argumento *model* es una estructura que contiene el modelo completo del robot que va a ser simulado, incluyendo el número de articulaciones, cómo están conectadas cada una de ellas, el sentido de giro de cada eslabón, la apariencia que va a tener en el simulador y algunas características más. En la sección siguiente se explicará con más detalle este argumento y se describirán los distintos modelos del robot utilizados en la Tesis. El campo *delta\_t* hace referencia al intervalo entre los distintos frames de la simulación y *joint\_vals* es una matriz que contiene una fila por cada frame de la animación y una columna por cada valor de la articulación, por lo tanto, *joint\_vals(i,j)* representa el valor de la articulación *j* en el frame *i*.

Si a la función se le proporciona sólo el primer argumento, se genera el archivo *data\_km.iv*, cuyo formato es propio de OpenInventor. SceneViewer muestra el archivo como una imagen estática con todas las articulaciones a cero. Si por el contrario se proporcionan tres argumentos, *drawmodel* crea el archivo *data\_mv.iv*. Al abrirlo con SceneViewer, se genera la animación del robot en un bucle infinito, con una pequeña parada entre la posición final y la inicial, lo que permite al

usuario analizar fácilmente ambos estados.

## 5.5. Modelo 3D del robot

Como ya se ha apuntado anteriormente, el argumento *model* de la función *drawmodel* es el que permite construir el modelo de cualquier robot. Este argumento posee 6 atributos que definen completamente el robot y permite simularlo en OpenInventor, estos son:

**NB** Número de cuerpos en la cadena excluyendo la base fija. *NB* es además el número de articulaciones.

**parent** *parent(i)* es el cuerpo padre o antecesor del cuerpo *i*. La base fija se define como el cuerpo 0 y el resto de los cuerpos se numeran desde 1 hasta *NB* en un orden tal que cada cuerpo tiene un número mayor que el de su padre. La articulación *i* conecta el cuerpo *parent(i)* con el cuerpo *i*.

**pitch** Define el tipo de articulación que puede ser de revolución (*pitch(i)=0*), prismática (*pitch(i)=inf*) o helicoidal (*pitch(i)=otro valor*). Las variables de las articulaciones de revolución y helicoidales son ángulos en radianes y las variables para articulaciones prismáticas son desplazamientos lineales.

**Xtree** Es la matriz de transformación desde el sistema de referencia del cuerpo *parent(i)* al sistema de referencia del cuerpo *i*.

**I** Es el tensor de inercia espacial del cuerpo *i* expresado en coordenadas de *i*.

**appearance** Son las instrucciones de dibujo de cada cuerpo usadas por la función *drawmodel* para simular el robot y su movimiento. *appearance{1}* representa la base fija y *appearance{i+1}* representa el aspecto del cuerpo *i*. Se permiten dibujar las siguientes primitivas:

1. {'box', [xlo xhi; ylo yhi; zlo zhi]}. Dibuja un paralelepípedo. *xhi* representa el máximo valor de *x*, *xlo* el mínimo y así sucesivamente.

2.  $\{ \text{'cyl'}, [x \text{ } y \text{ } z], r, h, 'X' \}$ . Dibuja un cilindro situado en la posición  $[x \text{ } y \text{ } z]$ , con radio  $r$ , altura  $h$  y cuyo eje principal tiene la misma dirección que el eje X.
3.  $\{ \text{'vertex'}, [...] \}$ . Especifica una lista de vértices para su posterior uso en los comandos *line* y *face*. Se trata de una matriz de  $n$  filas y 3 columnas, donde  $n$  es el número de vértices y la fila  $i$  especifica las coordenadas  $x, y$  y  $z$  de cada vértice  $i$ .
4.  $\{ \text{'line'}, [...] \}$ . Dibuja una línea a partir de un conjunto de vértices. Se trata de un vector que contiene el número del vértice por el que pasa la línea. Por ejemplo, si el vector contiene  $[1 \text{ } 2 \text{ } 3 \text{ } 4]$ , la línea empieza en el vértice 1, pasa por el 2 y el 3 y termina en el 4.
5.  $\{ \text{'face'}, [...], [...], [...] \}$ . Dibuja un conjunto de caras creando un poliedro. Cada vector contiene una lista de números de vértices que a su vez definen un polígono. Deben estar ordenados en sentido antihorario y todos los vértices deben estar en el mismo plano.

En esta Tesis de Máster se han simulador tres modelos distintos del robot. Un modelo del tronco del robot con la base fija en la cadera, un modelo del cuerpo completo del robot con la base fija en el pie de apoyo y un modelo de doble péndulo invertido.

### 5.5.1. Modelo del torso del robot

En primer lugar se ha construido el modelo del torso del robot tomando como base fija la cadera, creando dos cadenas cinemáticas de 4 grados de libertad para los brazos.

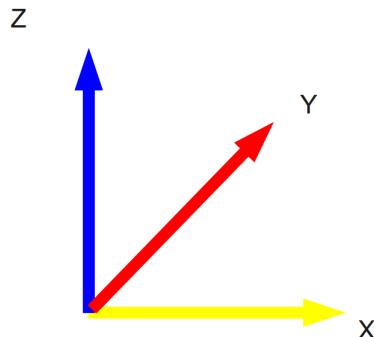
Para ello, en primer lugar, se determina la estructura que se quiere simular (*trunk*) y se definen el número de cuerpos (*NB*), el tipo de articulaciones (*pitch*) y como está conectada la cadena cinemática (*parent*). El argumento de *trunk.parent*

es un vector cuyos dos primeros valores son 0 y 0. Esto quiere decir que desde la base fija salen dos cadenas cinemáticas. Después, los siguientes valores especifican que las dos cadenas cinemáticas tienen 4 articulaciones cada una y no existen más bifurcaciones. A continuación se muestra el código en MATLAB.

*Definición de las cadenas cinemáticas*

```
nb=8;
trunk.NB=nb;
trunk.pitch = zeros(1,nb); % all revolute joints
trunk.parent=[0 0 1 2 3 4 5 6 7];
```

En robótica, es bastante habitual definir todas las propiedades de las articulaciones respecto a un sistema de referencia local, en vez de hacerlo respecto al sistema de referencia del mundo. Teniendo en cuenta que el sistema de referencia del mundo en OpenInventor es el de la Figura 5.9 y tomando los sistemas de referencia de la Figura 5.7, se definen los ejes de cada articulación. Estos ejes son los sistemas de referencia de los dos brazos que se observan en la Figura 5.7, que siguen las reglas de Denavit-Hartenberg.



**Figura 5.9:** Sistema de referencia del mundo en OpenInventor

*Definición de los sistemas de referencia de cada articulación*

```
% Length of arm's body in m
L1=0.111;
L2=L1;
L3=0.171;

trunk . Xtree{1}=Xrotz(-pi/2)*Xroty(pi/2)*Xtrans([L1 0 0]);
trunk . Xtree{2}=Xrotz(-pi/2)*Xroty(pi/2)*Xtrans([-L1 0 0]);
trunk . Xtree{3}=Xrotz(pi/2)*Xrotx(pi/2);
trunk . Xtree{4}=Xrotz(pi/2)*Xrotx(pi/2);
trunk . Xtree{5}=Xrotx(pi/2)*Xroty(-pi/2);
trunk . Xtree{6}=Xrotx(pi/2)*Xroty(-pi/2);
trunk . Xtree{7}=Xrotx(pi/2)*Xtrans([0 0 L2]);
trunk . Xtree{8}=Xrotx(pi/2)*Xtrans([0 0 L2]);
```

La longitud  $L1$  hace referencia a la distancia entre el pecho y el hombro,  $L2$  es la distancia del biceps del robot y  $L3$  la del antebrazo (Ver Figura 5.5).

Las rotaciones y traslaciones de  $trunk.Xtree\{1\}$  se definen de la siguiente manera: en primer lugar se produce una traslación de valor  $L1$  respecto del eje  $X$  del sistema de referencia anterior, en este caso el sistema de referencia del mundo; a continuación se produce un giro respecto al eje  $Y$  de  $\pi/2$  y por último, se produce un giro respecto al eje  $Z$  de  $-\pi/2$ . El resultado es el sistema de referencia de la articulación  $LARM\_JOINT[1]$  que se muestra en la Figura 5.7.

El resto de transformaciones para los demás sistemas de referencia se explican de forma equivalente, teniendo en cuenta que  $trunk.Xtree\{2\}$  define el sistema de referencia de la articulación  $RARM\_JOINT[1]$ ,  $trunk.Xtree\{3\}$  define el de  $LARM\_JOINT[2]$  y así sucesivamente.

El siguiente paso es dibujar el robot en OpenInventor. Para ello se utiliza *appearance*. Para simplificar el código, la apariencia del robot simulado está compuesta de cilindros y paralelepípedos. A continuación se muestra un extracto del código de MATLAB.

*Definición de la apariencia del robot*

```
% Robot appearance
trunk.appearance{1}={{'cyl',[0 0 0],0.008,L1,'Z'},...
{'cyl',[0 0 0],0.006,2*L1,'X'},...
{'cyl',[0 0 L1/2],0.03,0.08,'Z'}}; %fixed base
trunk.appearance{2}={{'cyl',[0 0 0],0.008,0.012,'Z'}};
trunk.appearance{3}={{'cyl',[0 0 0],0.008,0.012,'Z'}};
trunk.appearance{4}={{'cyl',[0 0 0],0.008,0.012,'Z'}};
trunk.appearance{5}={{'cyl',[0 0 0],0.008,0.012,'Z'}};
trunk.appearance{6}={{'cyl',[0 0 0],0.008,0.012,'Z'},...
{'cyl',[0 0 L2/2],0.005,L2,'Z'}};
trunk.appearance{7}={{'cyl',[0 0 0],0.0008,0.0012,'Z'},...
{'cyl',[0 0 L2/2],0.005,L2,'Z'}};
trunk.appearance{8}={{'cyl',[0 0 0],0.008,0.012,'Z'},...
{'cyl',[0 L3/2 0],0.005,L3,'Y'}};
trunk.appearance{9}={{'cyl',[0 0 0],0.008,0.012,'Z'},...
{'cyl',[0 L3/2 0],0.005,L3,'Y'}};
```

En la Figura 5.10 se muestra el modelo del torso robot en posición de reposo simulado mediante OpenInventor.

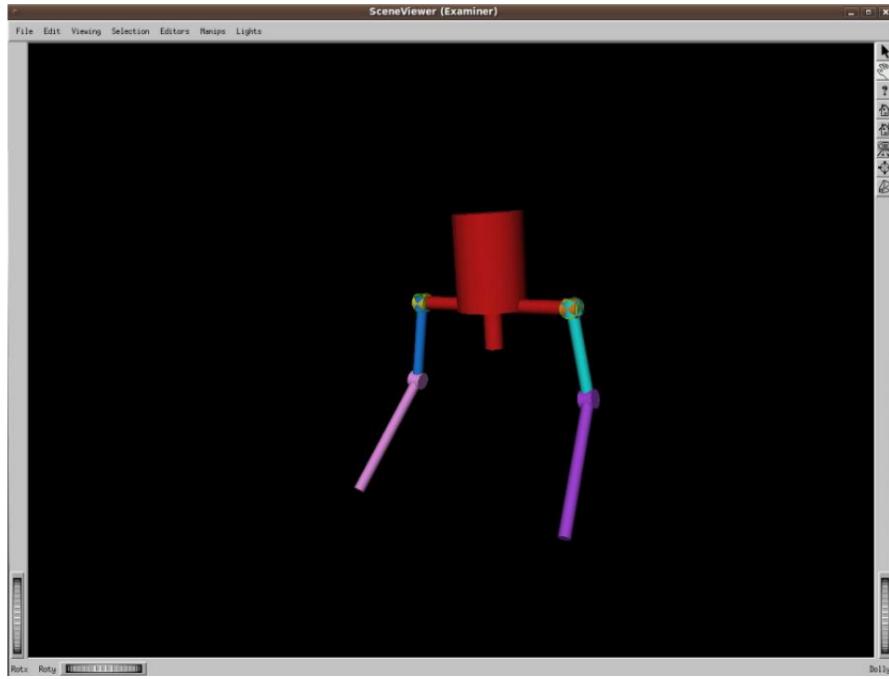


Figura 5.10: Simulación del torso del robot

### 5.5.2. Modelo del cuerpo completo del robot

Para poder programar cualquier trayectoria de movimiento en un robot humanoide, es necesario elegir cuidadosamente en qué lugar se va a situar el sistema de referencia global del robot o, en otras palabras, elegir la localización de la base móvil del robot.

Mediante este sistema de referencia global se define cada uno de los sistemas de referencia locales para cada articulación del robot, y además, define la posición global del robot respecto al sistema de referencia del mundo.

Una opción habitual es situar la base móvil del robot en su centro de gravedad, normalmente cerca de la cadera. Partiendo de ese punto, el humanoide se construye como un conjunto de cuatro cadenas cinemáticas abiertas, dos cadenas

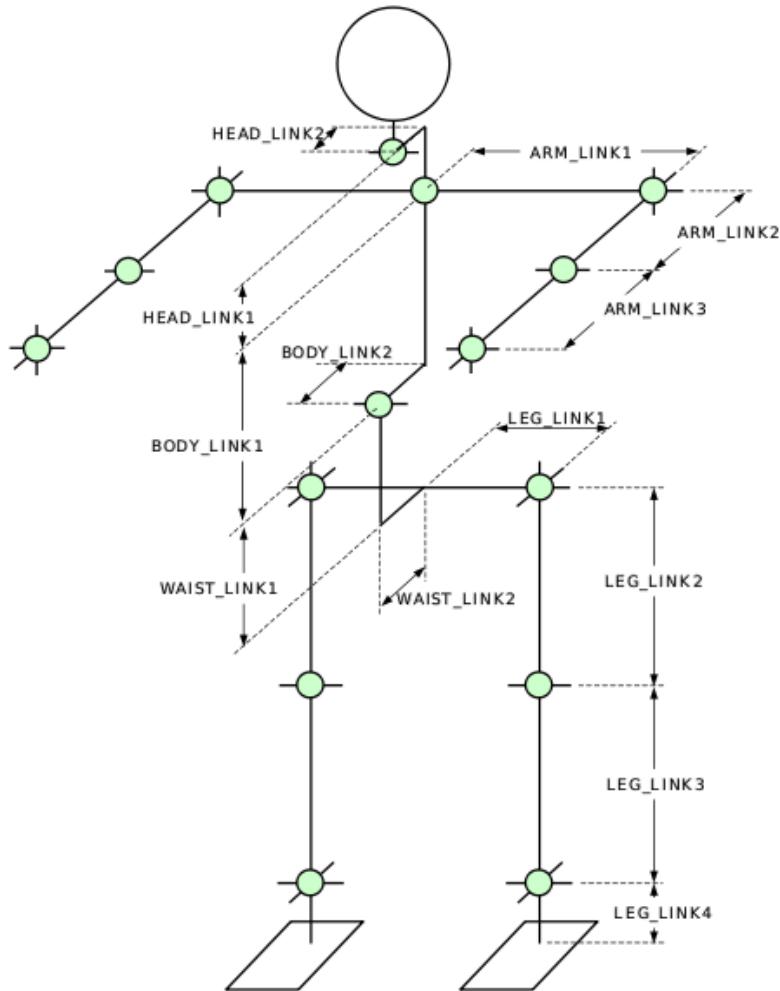
para los brazos y dos cadenas para las piernas. Este enfoque es muy utilizado por su simplicidad.

Otro enfoque es situar la base del robot en el pie de apoyo e ir cambiando a cada paso el sistema de referencia de un pie a otro. Un ejemplo se puede encontrar en (Arbulú y Balaguer, 2009). La ventaja que supone este enfoque es que elimina la necesidad de calcular las fuerzas de contacto, por lo menos en soporte simple, ya que el punto de contacto es el mismo que el sistema de referencia, desde donde se miden todas las fuerzas. Situando el sistema de referencia en el pie de apoyo, el tobillo y la rodilla soportan toda la fuerza ejercida por el peso del cuerpo, más la fuerza ejercida por el movimiento. Situando el sistema de referencia en la cadera, las fuerzas que aparecerían en el tobillo y la rodilla serían las equivalentes al peso del robot si está sentado, a menos claro, que computasemos las fuerzas de contacto.

En esta Tesis de Máster se ha construido el modelo del cuerpo completo del robot con la base fija en la pierna de apoyo, para analizar la dinámica de la locomoción bípeda. Con este modelo, se ha obtenido las fuerzas espaciales que aparecen en cada eslabón y los pares en las articulaciones de la fase de apoyo simple, tanto de la pierna de apoyo como de la pierna en vuelo. Esta fase, que ocupa alrededor de un 80 % del ciclo total de la caminata, es donde se producen los mayores pares y es la fase más crítica desde el punto de vista de la estabilidad. Es por ello que el estudio se ha centrado exclusivamente en esta parte.

La programación del modelo de cuerpo completo es muy parecida al del modelo del torso. Sin embargo, en este caso se crea una estructura bastante más complicada. Situando la base en el pie derecho, se define la cadena cinemática de la pierna derecha, que termina en la cadera. A partir de ese punto nacen otras

tres cadenas cinemáticas, correspondientes a los dos brazos y a la pierna izquierda.



**Figura 5.11:** Longitudes de los eslabones del HOAP

En primer lugar hay que definir las distancias de cada eslabón tomando como referencia la Figura 5.11.

*Longitudes de los eslabones del HOAP en m*

```
ARM_LINK1=0.111;
ARM_LINK2=0.111;
ARM_LINK3=0.171;
LEG_LINK1=0.039;
LEG_LINK2=0.105;
LEG_LINK3=0.105;
LEG_LINK4=0.040;
BODY_LINK1=0.125;
BODY_LINK2=0.035;
HEAD_LINK1=0.103;
HEAD_LINK2=0.015;
WAIST_LINK1=0.055;
WAIST_LINK2=0.035;
```

A continuación se definen el número de eslabones (*NB*), el tipo de articulaciones (*pitch*), la conectividad de cada una (*parent*) y por último, la matriz de transformación de cada articulación (*Xtree*). Si se presta atención a la secuencia de números definidos por el parámetro *parent*, se observa que el número 7 se repite 3 veces. De esta forma se definen las 3 cadenas cinemáticas anteriormente mencionadas que salen de la cadera, los dos brazos y la pierna izquierda.

*Parámetros del robot*

```
nb=21; % 4 for each arm, 6 for each leg and 1 for the waist
stance.NB=nb;
stance.pitch = zeros(1,nb); % all revolute joints
stance.parent= [0 1 2 3 4 5 6 7 8 9 10 7 12 13 14 7 16 ...
... 17 18 19 20];
```

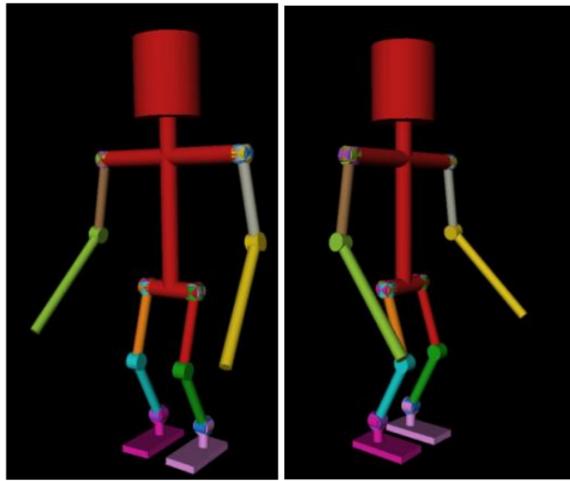
*Matrices de transformación de los eslabones*

```

stance . Xtree{1} = Xrotz(pi)*Xroty(pi/2);
stance . Xtree{2} = Xrotx(-pi/2);
stance . Xtree{3} = Xrotx(pi/2)*Xtrans([LEG_LINK3 0 0]);
stance . Xtree{4} = Xtrans([LEG_LINK3 0 0]);
stance . Xtree{5} = Xrotx(-pi/2);
stance . Xtree{6} = Xrotx(-pi/2)*Xrotz(-pi/2);
stance . Xtree{7} = Xrotx(pi/2)*Xrotz(-pi/2)*Xtrans ([ ...
... LEG_LINK1 -WAIST_LINK2 WAIST_LINK1 ]);
stance . Xtree{8} = Xrotx(pi)*Xtrans([-BODY_LINK2 ...
... BODY_LINK1 ARM_LINK1]);
stance . Xtree{9} = Xrotz(pi/2)*Xrotx(pi/2);
stance . Xtree{10} = Xrotx(pi/2)*Xroty(-pi/2);
stance . Xtree{11} = Xrotx(pi/2)*Xtrans([0 0 ARM_LINK2]);
stance . Xtree{12} = Xrotx(pi)*Xtrans([-BODY_LINK2 ...
... BODY_LINK1 -ARM_LINK1]);
stance . Xtree{13} = Xrotz(pi/2)*Xrotx(pi/2);
stance . Xtree{14} = Xrotx(pi/2)*Xroty(-pi/2);
stance . Xtree{15} = Xrotx(pi/2)*Xtrans([0 0 ARM_LINK2]);
stance . Xtree{16} = Xrotz(pi/2)*Xrotx(-pi/2)*Xtrans ([ ...
... -WAIST_LINK2 -WAIST_LINK1 -LEG_LINK1 ]);
stance . Xtree{17} = Xrotz(pi/2)*Xrotx(pi/2);
stance . Xtree{18} = Xrotx(pi/2);
stance . Xtree{19} = Xtrans([-LEG_LINK2 0 0]);
stance . Xtree{20} = Xtrans([-LEG_LINK3 0 0]);
stance . Xtree{21} = Xrotx(-pi/2);

```

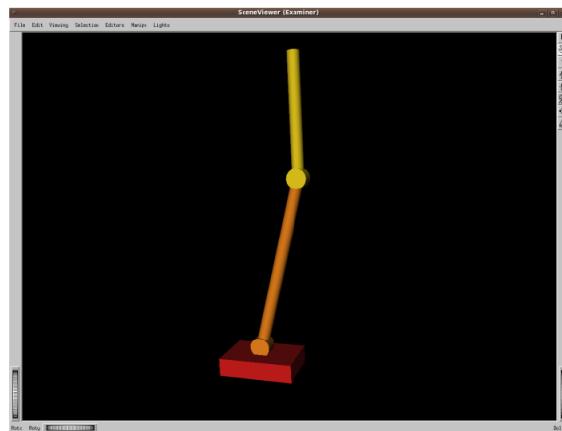
Finalmente en la Figura 5.12 se muestra el resultado en OpenInventor del cuerpo completo del robot.



**Figura 5.12:** Simulación del cuerpo completo del HOAP

### 5.5.3. Modelo de doble péndulo invertido

Para validar el control de estabilidad mediante el doble péndulo invertido se ha construido un modelo de éste utilizando las librerías de Featherstone (Figura 5.13). El objetivo es comparar los dos casos para su análisis y validación: la dinámica del robot real con la dinámica del doble péndulo obtenida por medio del algoritmo RNEA, basándose en vectores espaciales.



**Figura 5.13:** Simulación del doble péndulo invertido

Capítulo **6**

# Resultados

## 6.1. Introducción

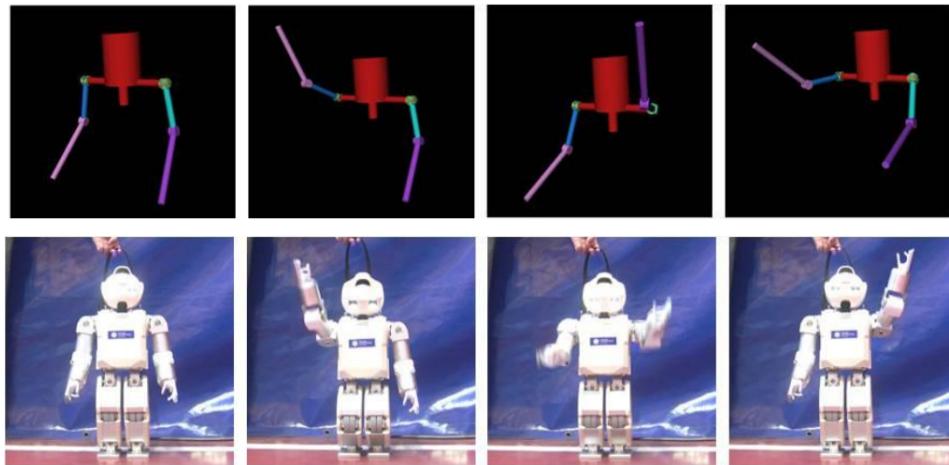
Una vez obtenido el modelo dinámico del robot, se va a pasar a analizar y validar los resultados. La obtención del modelo dinámico completo es una herramienta muy útil que puede servir de base para un estudio profundo de los movimientos de un robot humanoide.

Aunque este trabajo se centra en el estudio de la dinámica del robot, es evidente que para la obtención de las trayectorias se ha realizado un estudio cinemático preliminar. Se ha obtenido la cinemática directa por el método de Denavit-Hartenberg y la cinemática inversa por el método de la pseudoinversa, realizando un desacople cinemático de las piernas para evitar las cadenas cinemáticas cerradas. Gracias a esto, se han programado todo tipo de trayectorias en el humanoide, como caminar hacia delante y hacia atrás, pasos laterales, giros y otros movimientos. Las trayectorias son polinómicas de orden 3.

## 6.2. Estudio dinámico del torso del robot

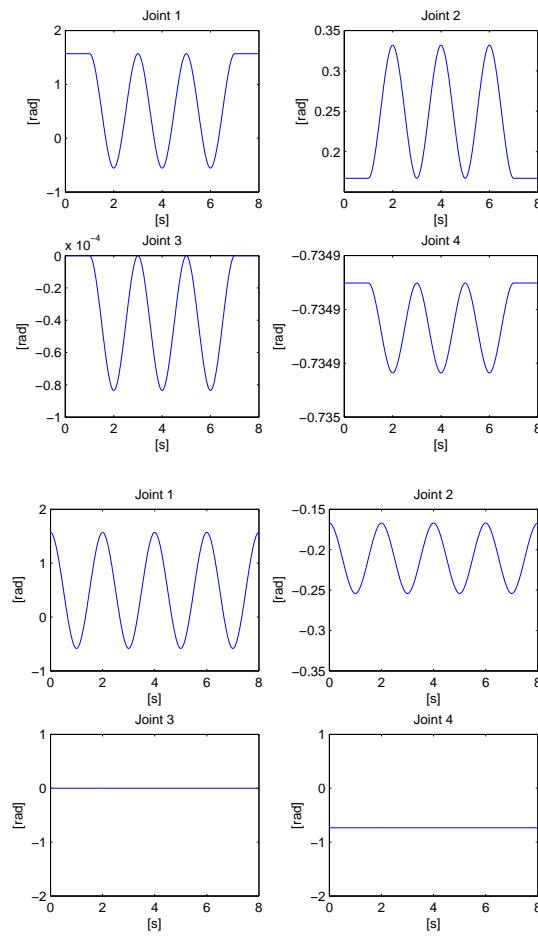
Con el modelo del torso se han probado dos trayectorias. Ambas fueron programadas para una demo del HOAP<sup>1</sup> en las XXX Jornadas de Automática celebradas en Valladolid en Septiembre de 2009, donde el robot tenía que realizar distintos movimientos de baile basándose en un conjunto de primitivas que se iban intercalando al ritmo de la música.

La primera de estas trayectorias es un movimiento de los brazos subiendo y bajando (Figura 6.1). Utilizando el algoritmo RNEA y partiendo de las posiciones, velocidades y aceleraciones de cada uno de las articulaciones, se obtienen los pares. Se han numerado las articulaciones según el criterio de Denavit-Hartenberg explicado anteriormente en la Figura 5.7.

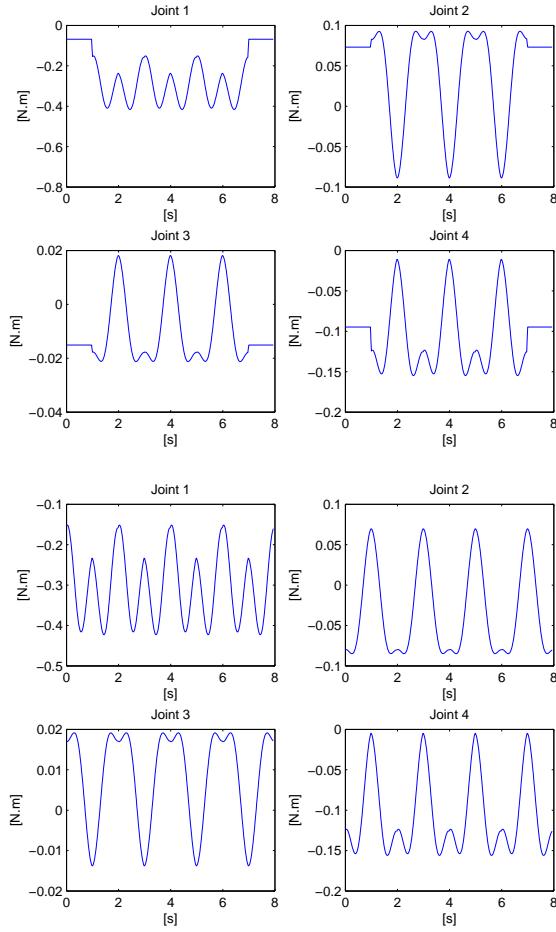


**Figura 6.1:** Trajetoria 1 en el modelo del torso

<sup>1</sup><http://www.youtube.com/watch?v=mu5psxG7bwA>



**Figura 6.2:** Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

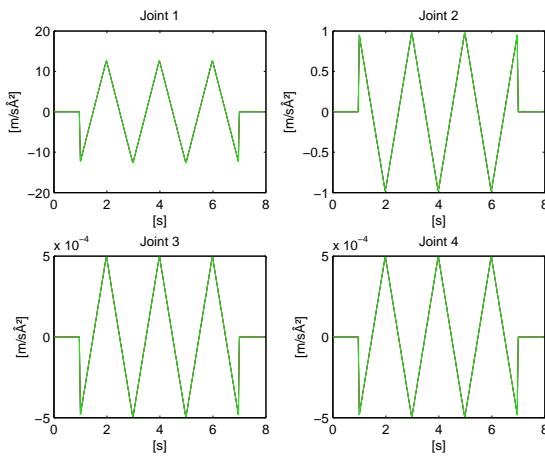


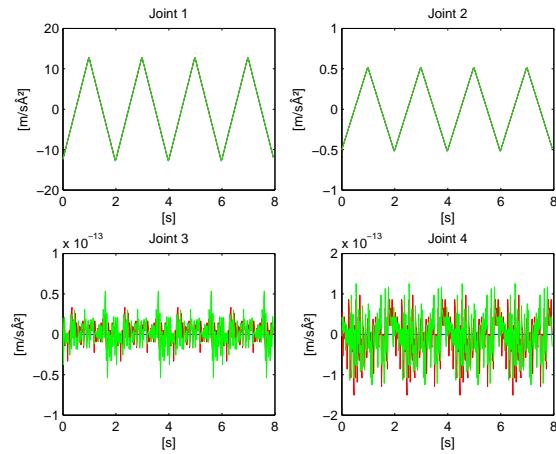
**Figura 6.3:** Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

A la vista de las gráficas de posiciones y pares de las articulaciones (Figura 6.2 y Figura 6.3) se extraen diversas conclusiones. Los pares producidos en general son muy pequeños, esto es lógico ya que el peso de los brazos es pequeño. Cabe destacar también, que el mayor par corresponde a la articulación 1, que es donde se produce un mayor movimiento. Esta articulación realiza el movimiento circular que hace que todo el brazo se mueva. También se puede observar que los pares en ambos brazos son muy parecidos, esto también es lógico ya que se

ha programado un movimiento casi idéntico. Pero la obtención de los pares mediante el algoritmo RNEA con vectores espaciales no es suficiente para darlos por auténticos, hace falta algún tipo de validación. En este trabajo se van a hacer dos validaciones con distintos métodos para asegurar que el modelo dinámico obtenido es el real o, estrictamente hablando, el modelo más real que se puede obtener.

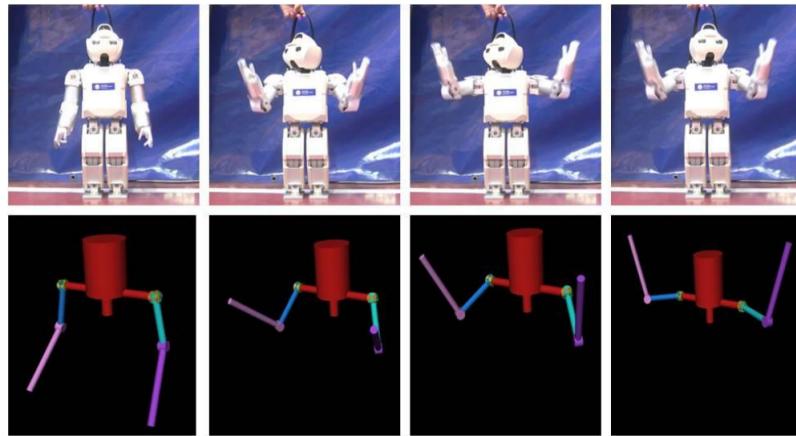
Las validaciones consistirán en realizar la dinámica directa partiendo de los pares previamente obtenidos, mediante los algoritmos CRBA y ABA, explicados en el capítulo 3. Gracias a estos algoritmos se obtienen las aceleraciones provocadas por los pares de cada articulación. En la Figura 6.4 se hace una comparación de la aceleración real (color azul) proporcionada por la segunda derivada de la posición, la aceleración obtenida por el método CRBA (color verde) y la aceleración obtenida por el método ABA (color rojo). Como se puede observar el resultado es muy satisfactorio ya que las tres aceleraciones coinciden.



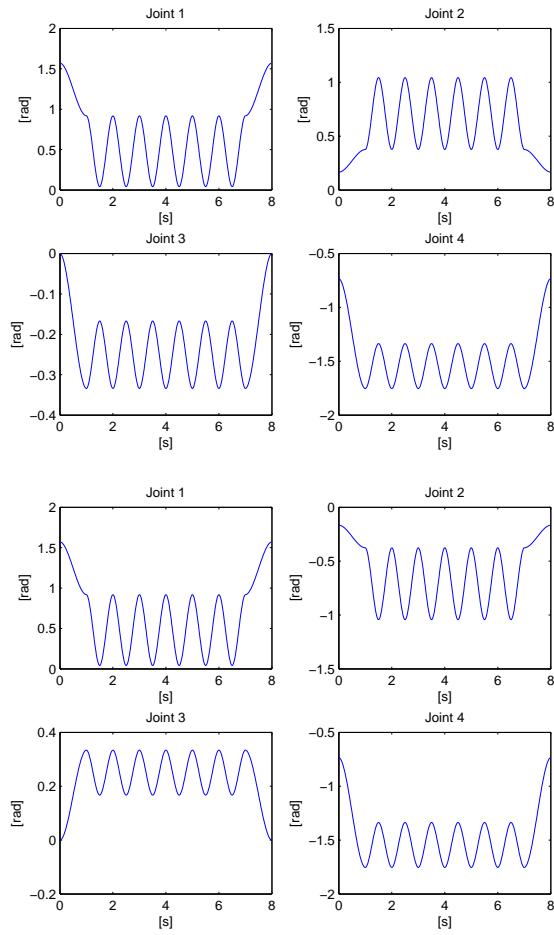


**Figura 6.4:** Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

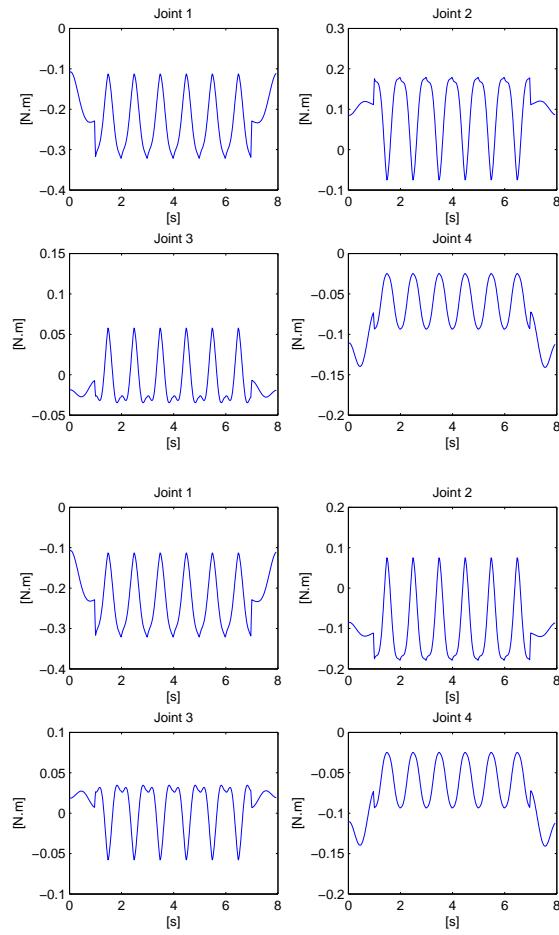
A continuación se evaluará una trayectoria más compleja, la trayectoria 2 (Figura 6.5). Esta trayectoria comienza con una elevación de los brazos desde la posición de inicio, realizándose seguidamente cuatro movimientos transversales hacia arriba y hacia abajo.



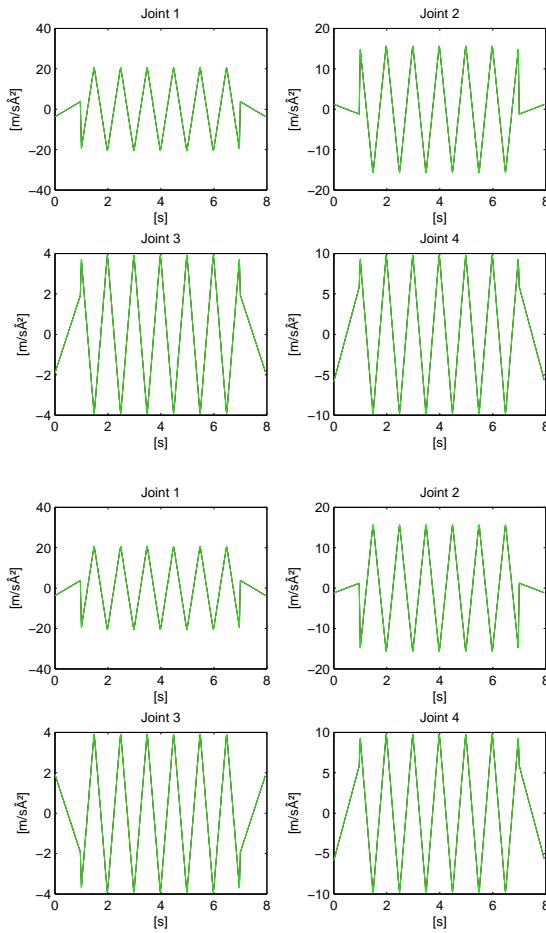
**Figura 6.5:** Trayectoria 2 en el modelo del torso



**Figura 6.6:** Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)



**Figura 6.7:** Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)



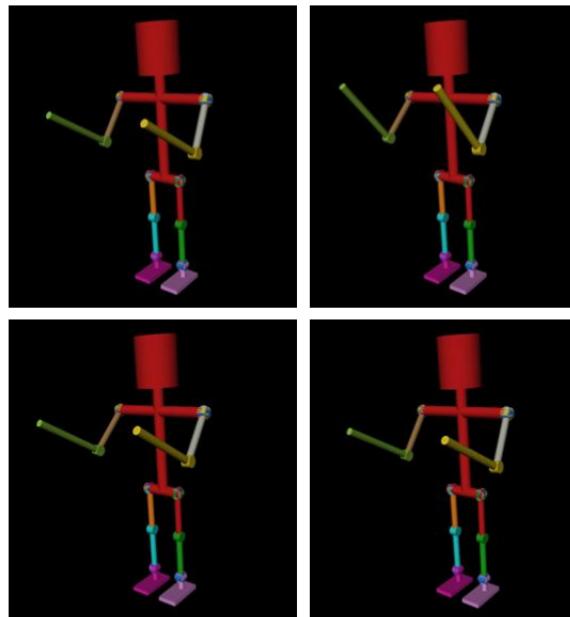
**Figura 6.8:** Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

En el caso de la trayectoria 2 los pares obtenidos son ligeramente mayores (Figura 6.7), aunque siguen siendo pequeños por el poco peso de los brazos. Esto es debido a que la trayectoria programada, cuyas posiciones se representan en la Figura 6.6, produce unas aceleraciones más elevadas que en el caso anterior. Se observa además que al realizar la dinámica directa las aceleraciones obtenidas por los métodos CRBA y ABA son iguales a la aceleración inicial, lo que permite validar los resultados (Figura 6.8).

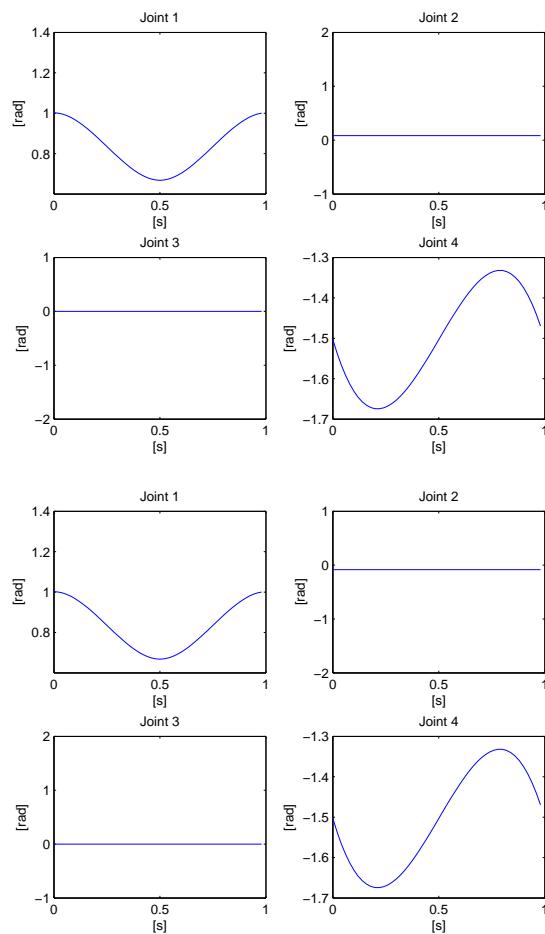
### 6.3. Estudio dinámico del cuerpo completo del robot

La obtención del modelo dinámico completo del robot constituye un punto de partida esencial para el desarrollo de cualquier tipo de investigación relacionada con el movimiento y la locomoción del humanoide. Conociendo completamente las fuerzas y momentos que aparecen en el robot, se puede realizar un estudio profundo del comportamiento de éste.

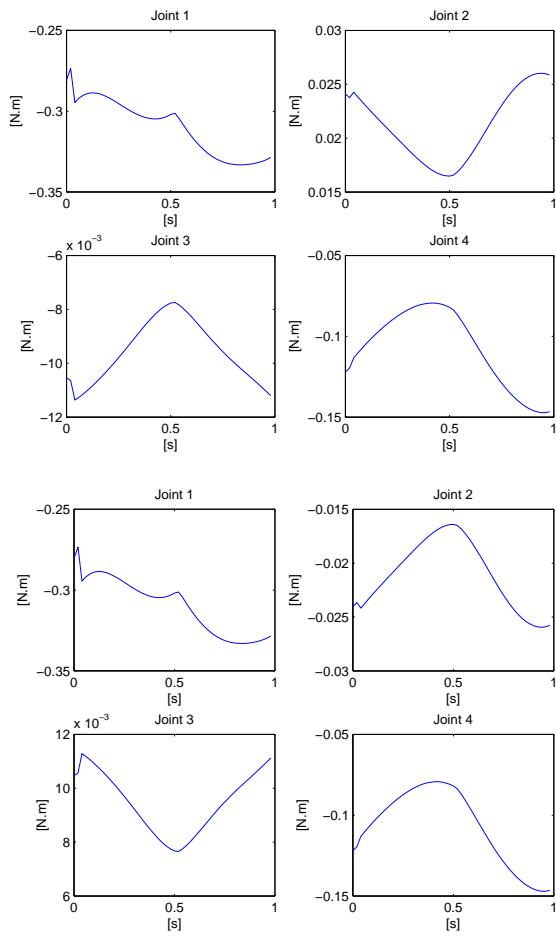
La primera prueba que se ha realizado en el modelo del cuerpo completo del robot es una trayectoria simple circular como se muestra en la Figura 6.9. Se ha programado una trayectoria circular idéntica para los dos brazos y se han obtenido los pares y aceleraciones como en los modelos del torso.



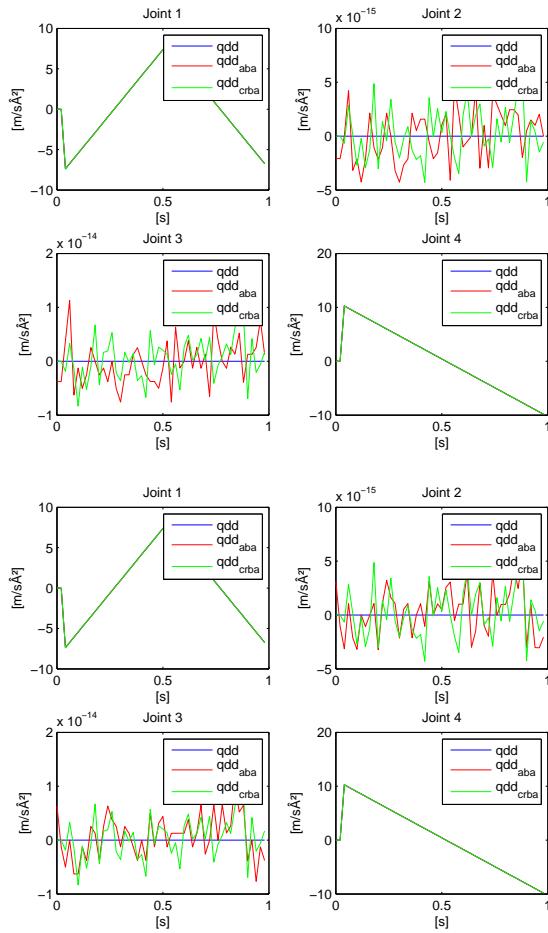
**Figura 6.9:** Trayectoria circular con el modelo completo del robot



**Figura 6.10:** Posición de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

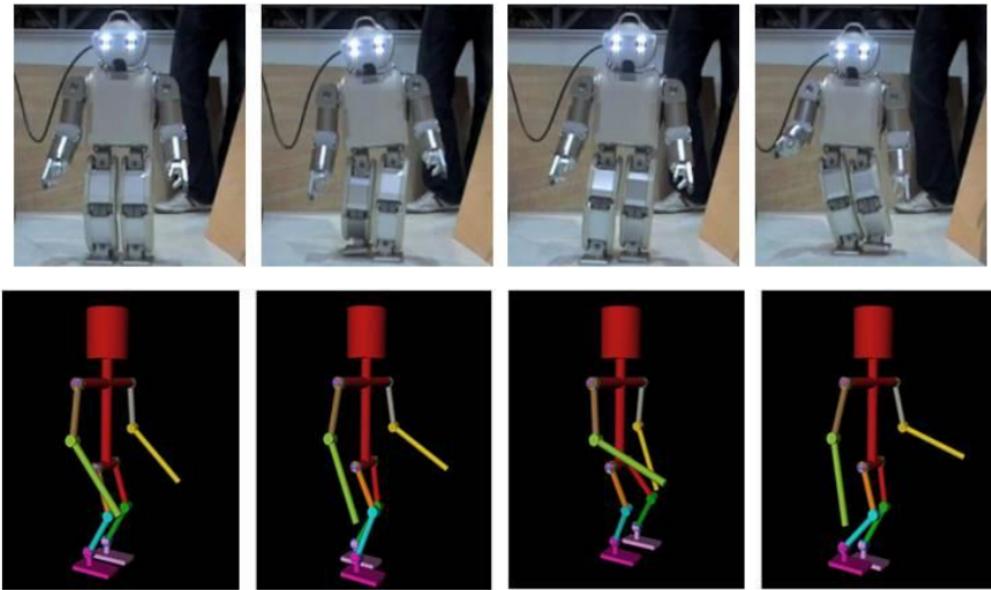


**Figura 6.11:** Pares de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)



**Figura 6.12:** Aceleración de las articulaciones del brazo izquierdo (arriba) y derecho (abajo)

Al igual que en el modelo del torso, se contemplan unos pares muy pequeños (Figura 6.11) en las articulaciones y cabe destacar que el mayor par se produce en la articulación 1 que es la que produce un mayor movimiento (Figura 6.10). En ambas extremidades se produce el mismo par, debido a un movimiento similar, y para una mayor validación del modelo basta observar las gráficas de las aceleraciones (Figura 6.12) en las que se observa que la aceleración real y las obtenidas mediante dinámica directa por los métodos CRBA y ABA son idénticas.



**Figura 6.13:** Locomoción bípeda, robot real y simulado

En la Figura 6.13 se muestra un conjunto de fotogramas donde aparece el HOAP caminando hacia delante, y una simulación en OpenInventor de la caminata del robot. Es importante destacar que la trayectoria que se está simulando es exactamente la misma que se introduce en el robot para que camine, sin necesidad de cambiar una línea de código . De esta forma se obtiene un modelo que permite obtener los pares en las articulaciones y las fuerzas espaciales en cada eslabón, partiendo de unas trayectorias introducidas, que son las mismas que utiliza el robot para efectuar los movimientos.

Tomando como punto de apoyo la pierna derecha, la pierna izquierda se balancea desarrollando unos pares muy pequeños en comparación con los de la pierna de apoyo que soporta todo el peso (Figura 6.14). Cabe destacar que las articulaciones donde se genera mayor variación de par es en la 3, 4 y 5 que justamente coincide con las articulaciones cuyo eje es perpendicular al plano sagital, es decir son las que producen el movimiento de la pierna hacia delante.

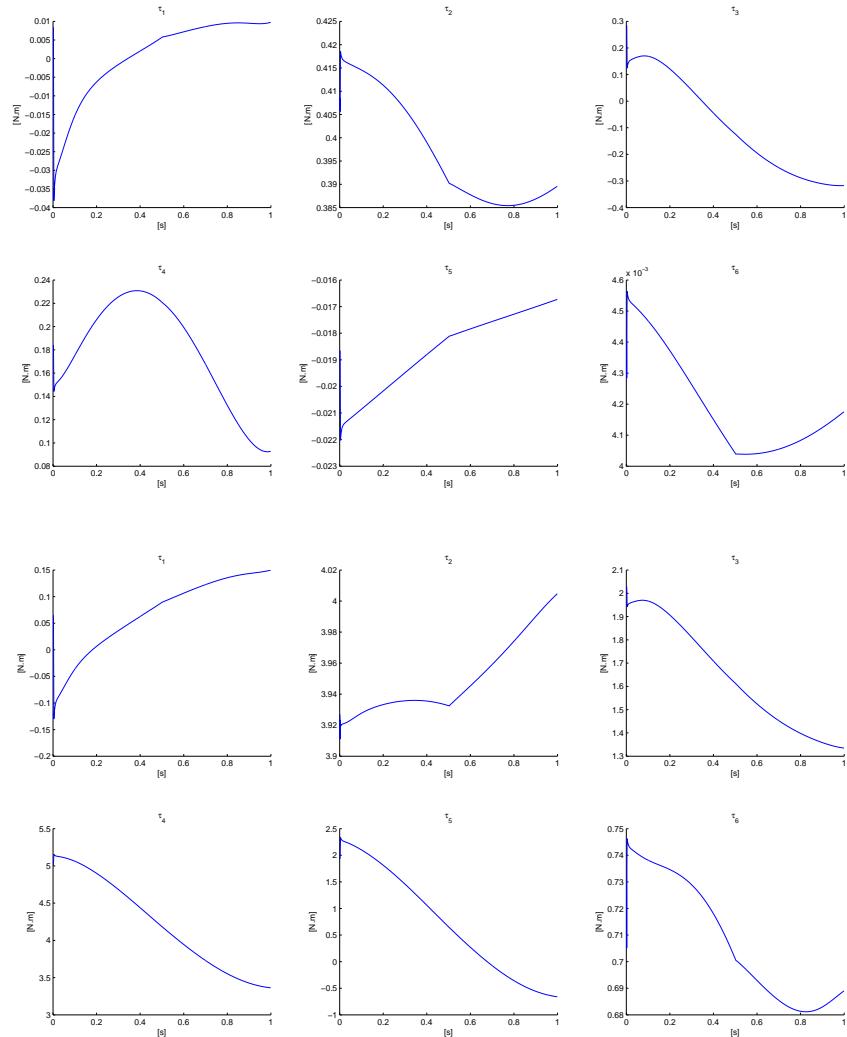
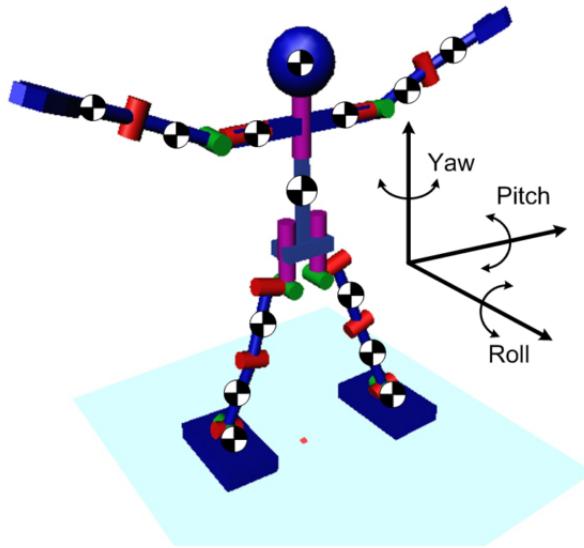


Figura 6.14: Pares en la pierna en vuelo (a) y en la pierna de apoyo



**Figura 6.15:** Estructura generalizada de un robot humanoide

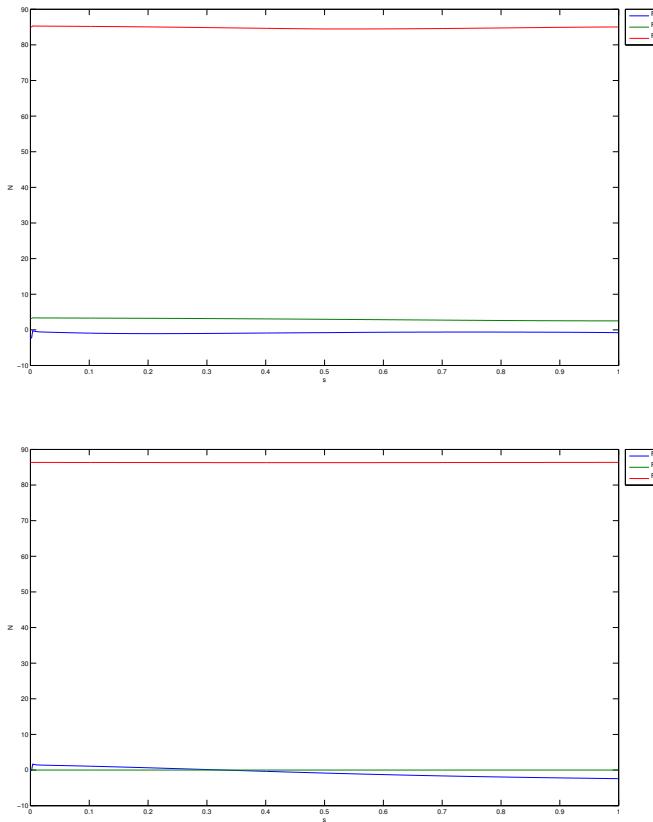
#### 6.4. Estudio dinámico del doble péndulo invertido

En este apartado se va a presentar una comparación entre las fuerzas que aparecen en el modelo del doble péndulo invertido y las que aparecen en el modelo completo del robot. Con ello lo que se pretende es verificar hasta qué punto son válidos los modelos reducidos.

El algoritmo RNEA permite conocer los pares que se producen en las articulaciones y, además de esto, las fuerzas y momentos que aparecen en cada uno de los eslabones. Si se quiere obtener la resultante de las fuerzas que actúan sobre el conjunto de sólidos rígidos, es necesario proyectar todas las fuerzas sobre un mismo sistema de referencia, que en este caso va a ser el que aparece la Figura 6.15.

Para poder afirmar que un modelo reducido, como el del doble péndulo invertido, es equivalente al modelo real, las fuerzas que se producen en ambos

modelos al realizar el mismo movimiento han de ser parecidas. Para comparar los dos modelos se ha utilizado la misma trayectoria programada en el apartado anterior, sin embargo, esta vez se han obtenido las fuerzas que atraviesan cada eslabón y se ha obtenido la resultante (Figura 6.16).



**Figura 6.16:** Resultante de las fuerzas producidas por un paso. Arriba el modelo completo y abajo el modelo del doble péndulo

Como se puede observar, las gráficas son muy parecidas.  $F_x$  es la fuerza resultante en el eje Roll,  $F_y$  es la fuerza en el eje Pitch y  $F_z$  es la fuerza en el eje Yaw (ver Figura 6.15). Los valores positivos indican compresión.

Analizando las gráficas se observa que la componente de mayor valor es  $F_z$ , que se corresponde con el peso del robot. En el caso del modelo de cuerpo completo el primer valor es de  $83,3N$ , y en el caso del péndulo es de  $85,7N$ . Respecto a las fuerzas producidas en el plano horizontal destaca que en ambos casos son muy pequeñas, por lo que el robot está en una posición casi totalmente estable. En el péndulo ambas fuerzas son cercanas a cero desviándose ligeramente la  $F_y$  en el modelo completo que comienza con un valor de  $5,5N$ .

Teniendo en cuenta que la masa total del robot es de  $8,8Kg$  (sin baterías), se obtiene un peso estático de  $8,8 \times 9,8 = 86,24N$ . Sabiendo esto, se puede deducir que la aportación de la aceleración producida por el movimiento del paso hacia delante es muy pequeña. Esto hace que al movimiento se le pueda considerar estático.

La conclusión obtenida es que para el caso de la trayectoria programada, donde la cadera realiza un movimiento pequeño y la velocidad no es muy elevada, las fuerzas que aparecen en el doble péndulo son prácticamente las mismas que aparecen en el modelo completo, por lo que su comportamiento es equivalente y la simplificación del robot como un doble péndulo es correcta.

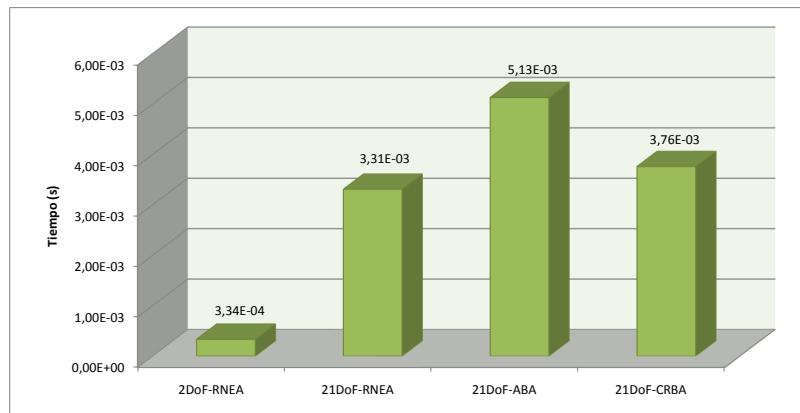
## 6.5. Tiempos de ejecución

En cualquier algoritmo de control de movimiento de un robot humanoide, el tiempo de computación es además de fundamental, excluyente. Un algoritmo que tarde en ejecutarse más de un determinado tiempo límite, no puede ser aplicado en tiempo real en el robot. Ahora bien, evaluar este tiempo límite es un tema que requiere una profunda discusión.

En el caso particular del HOAP-3, el tiempo mínimo de muestreo es de 1 ms, sin embargo, se han programado movimientos con tiempos de muestreo de 2 y

5ms. El tiempo de muestreo usado en todas las trayectorias que aparecen en esta Tesis de Máster ha sido de 2 ms.

A continuación se van a comparar los tiempos de ejecución de los diferentes algoritmos usados en este trabajo (Figura 6.17). Las dos primeras entradas de la gráfica se corresponden con el algoritmo RNEA de dinámica inversa. La entrada *2DoF-RNEA* hace referencia al algoritmo del doble péndulo de dos grados de libertad, la entrada *21DoF-RNEA* hace referencia a la dinámica inversa del modelo de cuerpo completo. Como se puede observar, el algoritmo de 2 grados de libertad es diez veces más rápido que el de 21 grados de libertad, es por ello que es fácil de implementar en tiempo real.



**Figura 6.17:** Tiempos de ejecución de los algoritmos dinámicos

Las entradas *21DoF-ABA* y *21DoF-CRBA* hacen referencia a los algoritmos ABA y CRBA de dinámica directa en el modelo del cuerpo completo. Como se ve en la gráfica, ambos algoritmos superan los 3 ms, siendo ligeramente más lento el algoritmo ABA.

Analizando estos resultados se extrae una conclusión prometedora. Existe un algoritmo de dinámica inversa (RNEA), y otro de dinámica directa (CRBA), cuyos tiempos de ejecución son de pocos milisegundos, en torno a 3ms, para un humanoide de 21 grados de libertad. Por lo tanto, un trabajo futuro muy interesante sería tratar de implementar estos algoritmos dinámicos en tiempo real en el robot, usando un lenguaje de programación más potente como C o C++ que permitiera reducir los tiempos de ejecución y usando un tiempo de muestreo de mayor, por ejemplo 5 ms.

# Conclusiones y trabajos futuros

## 7.1. Conclusiones

La obtención de un modelo completo de la dinámica del robot, permite conocer con gran exactitud, las fuerzas y aceleraciones que aparecen en cada uno de sus eslabones. La ventaja de este conocimiento radica en que este modelo sirve como punto de referencia para la validación de otros modelos más simplificados que se pueden ejecutar en tiempo real.

Para la realización de este trabajo se ha utilizado la teoría del álgebra espacial desarrollada por Featherstone, que trata los algoritmos de dinámica directa e inversa mediante vectores espaciales de 6 dimensiones. Este enfoque permite una mayor claridad en las ecuaciones dinámicas, evitando la necesidad de calcular algunos términos de la ecuación de movimiento basado en vectores tridimensionales, como los términos centrífugos y de Coriolis, además de reducir el tiempo de computación.

Basándose en los objetivos iniciales de esta tesis, se van a discutir los resultados obtenidos:

**Construcción de un modelo del tronco del robot HOAP-3.** En primer lugar se ha modelado la parte superior del cuerpo del robot en el simulador Open-Inventor y se han introducido trayectorias de movimiento para finalmente comprobar la similitud con el comportamiento real del robot.

**Construcción de un modelo del cuerpo completo del robot.** Partiendo de una base fija, situada en el pie de apoyo, se ha modelado el robot como un árbol cinemático ramificado y se han implementado varias trayectorias de movimiento. Este modelo se ha diseñado principalmente para obtener las fuerzas y pares producidos en el pie de apoyo y en la pierna en vuelo durante la caminata.

**Construcción de un modelo de un doble péndulo invertido.** Se ha construido un modelo de doble péndulo invertido de dos grados de libertad para validar un control de estabilidad.

**Implementación de la dinámica inversa y directa en estos modelos.** Para cada modelo de los anteriormente mencionados, se ha realizado la dinámica inversa utilizando el algoritmo Recursive Newton Euler (RNEA), obteniendo los pares articulares partiendo de las aceleraciones producidas por las trayectorias programadas. Además, se ha realizado la dinámica directa mediante los métodos Composite Rigid Body Algorithm (CRBA) y Articulated Body Algorithm (ABA), obteniéndose las aceleraciones articulares partiendo de los pares calculados con el método RNEA. Por último se han comparado las aceleraciones obtenidas mediante la dinámica directa con la aceleración original de las trayectorias para comprobar la validez de los métodos estudiados. Todos estos algoritmos están programados siguiendo la notación espacial de vectores de seis dimensiones.

**Aplicación de este desarrollo para la validación de un control de estabilidad.**

Se han comparado la resultante de todas las fuerzas producidas en el modelo completo del robot, cuando realiza un movimiento de paso hacia delante, con la resultante de un modelo simplificado de doble péndulo invertido. Se ha comprobado la similitud de los resultados lo que permite la validación de este modelo simplificado.

**Verificación y discusión de los resultados obtenidos.** Se han validado y discutido los resultados obtenidos.

**Análisis de los tiempos de ejecución de los algoritmos dinámicos.** Se han analizado los tiempos de ejecución de los algoritmos de dinámica inversa y directa, y se han obtenido tiempos de ejecución en torno a 3 ms para el modelo completo del robot y de en torno a 0,3 ms para el modelo de doble péndulo invertido.

## 7.2. Trabajos futuros

Para finalizar, se proponen una serie de trabajo futuros que se pueden desarrollar partiendo de esta tesis.

En primer lugar, este trabajo puede ser un punto de referencia y de validación de resultados para todos los desarrollos relacionados con modelos simplificados, desde modelos de generación de trayectorias hasta para controles de estabilidad del humanoide.

En el diseño mecánico y dimensionamiento de los motores en robots humanoides, también puede ser útil este trabajo, ya que permite conocer con bastante exactitud los pares que se producen en cada una de las articulaciones.

Partiendo de un desarrollo mejorado y optimizado de este modelo y utilizando un lenguaje de programación más eficiente como C++, se podría tratar de obtener el modelo dinámico del robot para su aplicación en tiempo real. Calculando la resultante de las fuerzas y momentos del robot completo, se podría generar una trayectoria de caminata estable basándose en el criterio de ZMP. Para ello sería necesario aumentar el tiempo de muestreo.

## Referencias

- Arbulú, M., y Balaguer, C. (2009). Real-time gait planning for rh-1 humanoid robot using local axis gait algorithm. *International Journal of Humanoid Robotics*, 6(1), 71–91.
- Arbulú, M., Kaynov, D., Cabas, L. M., y Balaguer, C. (2009). The rh-1 full-size humanoid robot: design, walking pattern generation and control. *Journal of Applied Bionics and Biomechanics.*, 6(3), 301–344.
- Armstrong, W. W. (1979). Recursive solution to the equations of motion of an n-link manipulator. En *Proceedings of the 5th world congress on theory of machines and mechanisms* (pp. 1343–1346).
- Balafoutis, C. A., y Patel, R. V. (1989). Efficient computation of manipulator inertia matrices and the direct dynamics problem. *IEEE Transactions on Systems, Man, and Cybernetics*, 19, 1313–1321.
- Barrientos, A., Peñín, L. F., Balaguer, C., y Aracil, R. (2007). *Fundamentos de robótica* (2<sup>a</sup> ed.). Mc Graw-Hill.
- Book, W. J. (1984). Recursive lagrangian dynamics of flexible manipulator arms. *International Journal of Robotics Research*, 3, 87–101.
- Dasgupta, A., y Nakamura, Y. (1999). Making feasible walking motion of humanoids. En *Proceedings of the ieee international conference on robotics and automation* (pp. 1044–1049).
- Featherstone, R. (1983). The calculation of robot dynamics using articulated-body inertias. *International J. of Robotics*, 2(1), 13–29.
- Featherstone, R. (1987). *Robot dynamics algorithms*. Kluwer Academic Publishers.
- Featherstone, R. (1999a). A divide-and-conquer articulated-body algorithm for parallel  $O(\log(n))$  calculation of rigid-body dynamics. part 1: Basic algorithm. *I. J. Robotic Res.*, 18(9), 867–875.
- Featherstone, R. (1999b). A divide-and-conquer articulated-body algorithm for parallel  $O(\log(n))$  calculation of rigid-body dynamics. part 2: Trees, loops, and accuracy. *I. J. Robotic Res.*, 18(9), 876–892.

- Featherstone, R. (2006). Plucker basis vectors. En *Icra* (pp. 1892–1897).
- Featherstone, R. (2008). *Rigid body dynamics algorithms*. Springer.
- Featherstone, R., y Fijany, A. (1999). A technique for analyzing constrained rigid-body systems and its application to the constraint force algorithm. *IEEE Transactions on Robotics and Automation*, 15, 1140–1144.
- Hirai, K., Hirose, M., Haikawa, Y., y Takenaka, T. (1998). The development of the honda humanoid robot. En (pp. 1321–1326). Proceedings of IEEE International Conference on Robotics and Automation.
- Hollerbach, J. M. (1980). A recursive lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. *IEEE Trans. Syst., Man, and Cybern.*, 730–736.
- Hollerbach, J. M., Johnson, T. L., Lozano-Perez, T., Mason, M. T., y Brady, M. (1982). *Robot motion: Planning and control*. MIT Press.
- Johanni, R., Otter, M., y Brandl, H. (1986). A very efficient algorithm for the simulation of robots and similar multibody systems without inversion of the mass matrix. En *Proceedings of the ifac/ifip/imacs international symposium on theory of robots* (pp. 95–100).
- Kahn, M. E., y Roth, B. (1971). The near-minimum-time control of open-loop articulated kinematic chains. *J. Dynamic Systems, Measurement and Control*(93), 164–172.
- Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H., y Kajita, S. (2001). The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation. En *Proceedings of the ieee conference on intelligent robots and systems* (pp. 239–246).
- Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., y cols. (2004). Humanoid robot hrp-2. En (pp. 1083–1090). Proceedings of IEEE International Conference on Robotics and Automation.
- Katic, D., y Vukobratovic, M. (2003). Survey of intelligent control techniques for humanoid robots. *Jounal of Intelligent and Robotics Systems*(37), 117–141.

- Kaynov, D. (2008). *Open motion control architecture for humanoid robots*. Tesis Doctoral no publicada.
- Kaynov, D., Souères, P., Pierro, P., y Balaguer, C. (2009). A practical decoupled stabilizer for joint-position controlled humanoid robots. En *Iros'09: Proceedings of the 2009 ieee/rsj international conference on intelligent robots and systems* (pp. 3392–3397). Piscataway, NJ, USA: IEEE Press.
- Lilly, K. W., y Orin, D. E. (1991). Alternate formulations for the manipulator inertia matrix. *International Journal of Robotics Research*, 10, 64–74.
- Luh, J. Y. S., Walker, M. W., y Paul, R. P. C. (1980). On-line computational scheme for mechanical manipulators. *J. Dynamic Systems, Meas. and Control*, 102, 69–76.
- McMillan, S., y Orin, D. E. (1995). Efficient computation of articulated-body inertias using successive axial screws. *IEEE Transactions on Robotics and Automation*, 11, 606–611.
- McMillan, S., y Orin, D. E. (1998). Forward dynamics of multilegged vehicles using the composite rigid body method. En *Proceedings of the ieee international conference on robotics and automation* (pp. 464–470).
- Ogura, Y., Aikawa, H., Kondo, H., Morishima, A., Lim, H. ok, y Takanishi, A. (2006). Development of a new humanoid robot wabian-2. En (pp. 76–81). International Conference on Proceedings of IEEE Robotics and Automation.
- Orin, D. E., McGhee, R. B., Vukobratovic, M., y Hartoch, G. (1979). Kinematic and kinetic analysis of open-chain linkages utilizing newton-euler methods. *Mathematical Biosciences*, 43, 107–130.
- Pai, D. K., Cloutier, B. P., y Ascher, U. M. (1997). Forward dynamics, elimination methods, and formulation stiffness in robot simulation. *International Journal of Robotics Research*, 16, 749–758.
- Poole, C., Safko, J., y Goldstein, H. (2000). *Classical mechanics (third edition)*. Addison Wesley.

- Roberson, R. E., y Schwertassek, R. (1988). *Dynamics of multibody systems*. Springer-Verlag.
- Rodríguez, G. (1987). Kalman filtering, smoothing, and recursive robot arm forward and inverse dynamics. *IEEE Journal on Robotics and Automation*, 6, 624–639.
- Rodríguez, G., Kreutz-Delgado, K., y Jain, A. (1991). A spatial operator algebra for manipulator modeling and control. *The International Journal of Robotics Research*, 10, 371–381.
- Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., y Fujimura, K. (2002). The intelligent asimo: System overview and integration. En (pp. 2478–2483). IEEE/RSJ international conference on intelligent robots and systems.
- Sciavicco, L., Villani, L., Oriolo, G., y Siciliano, B. (2009). *Robotics. modelling, planning and control*. Springer.
- Sharf, I., D'Eleuterio, G. M. T., y Fijany, A. (1995). Parallel  $O(\log(n))$  algorithms for computation of manipulator forward dynamics. *IEEE Transactions on Robotics and Automation*, 11, 389–400.
- Stepanenko, Y., y Vukobratovic, M. (1976). Dynamics of articulated open-chain active mechanisms. *Mathematical Biosciences*, 28, 137–170.
- Uicker, J. J. (1967). Dynamic force analysis of spatial linkages. *J. Appl. Mechanics*(34), 418–424.
- Vereshchagin, A. F. (1970). Computer simulation of the dynamics of complicated mechanisms of robot manipulators. *Engineering Cybernetics*, 6, 65–70.
- Vukobratovic, M., y Borovac, B. (2004). Zero-moment point - thirty five years of its life. *International Journal of Humanoid Robotics*, 1(1), 157–173.
- Walker, M. W., y Orin, D. E. (1982). Efficient dynamic computer simulation of robotic mechanisms. *J. Dyn. Systems, Meas. and Control, ASME Trans*, 104(3), 205–211.
- Wittenburg, J. (1977). *Dynamics of systems of rigid bodies*. B. G. Teubner.

Yamaguchi, J., Soga, E., Inoue, S., y Takanishi, A. (1999). Development of a bipedal humanoid robot - control method of whole body cooperative dynamic biped walking. En (pp. 3801–3806). Proceedings of IEEE International Conference on Robotics and Automation.