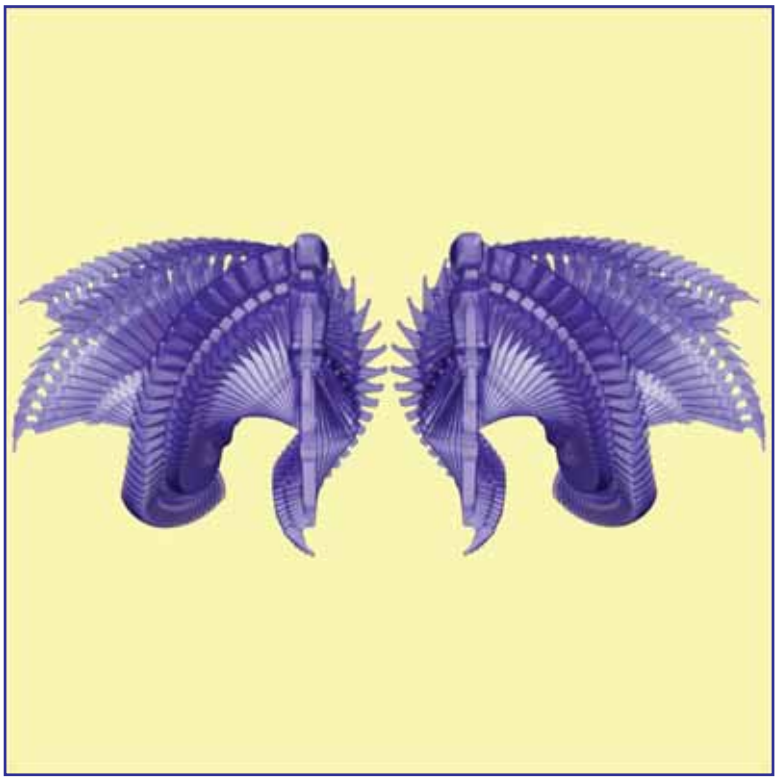
CENGAGE Learning™

Física para videojuegos

- Animación basada en principios físicos y matemáticos.
- Una referencia básica para estudiantes y profesionistas dedicados al desarrollo de videojuegos, gráficos, ingeniería y matemáticas.
- Una exhaustiva investigación que muestra cómo los modelos matemáticos se derivan de los principios físicos y matemáticos.
- Una concisa y clara explicación de cómo los modelos matemáticos son resueltos por computadora de forma eficiente y estable.



Erleben / Sporring / Henriksen / Dohlmann

Física para videojuegos

Kenny Erleben, Jon Sporring, Knud Henriksen y Henrik Dohlmann

Traducción

Jorge Manzano Olmos

Traductor profesional

Revisión Técnica

Héctor Manuel Gómez Gutiérrez

Maestro en Ciencias en Física

CENGAGE Learning™

Australia • Brasil • Corea • España • Estados Unidos • Japón • México • Reino Unido • Singapur

Física para videojuegos
Kenny Erleben, Jon Sporring, Knud Henriksen y Henrik Dohlmann

Presidente de Cengage Learning Latinoamérica
Fernando Valenzuela Migoya

Director de producto y desarrollo Latinoamérica
Daniel Oti Yvonnet

Director editorial y de producción Latinoámerica
Raúl D. Zendejas Espejel

Editor
Pablo Miguel Guerrero Rosas

Coordinadora de producción editorial
Abril Vega Orozco

Editora de producción
Gloria Luz Olguín Sarmiento

Coordinador de manufactura
Rafael Pérez González

Diseño de portada
Tyler Creative

Imagen de portada
Kenny Erleben

Composición tipográfica
Black Blue

Se terminó de imprimir en
octubre de 2011
en los talleres de
Edamsa Impresiones, S.A. de C.V.
en Av. Hidalgo 111, colonia San Nicolás Tolentino
Iztapalapa, México D.F. CP 09850

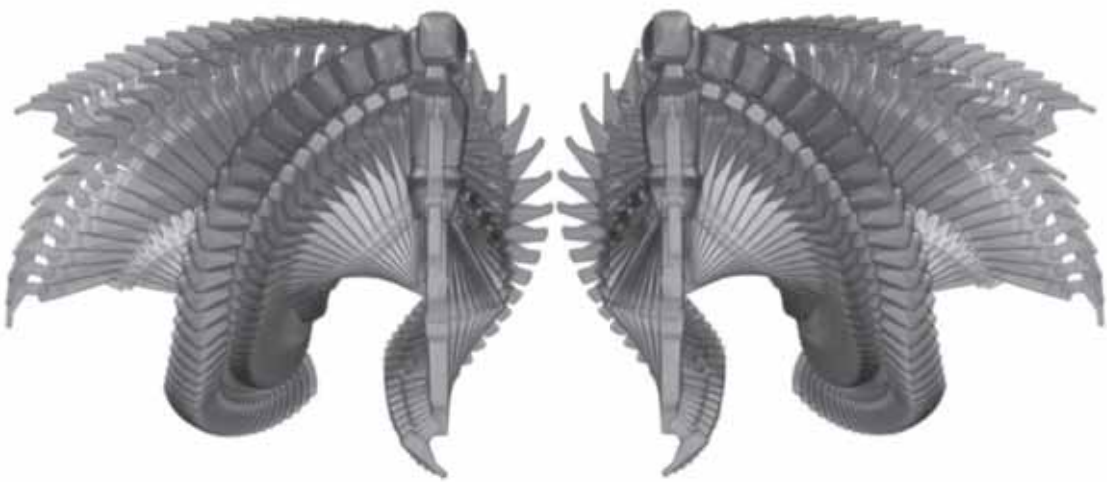
Traducido del libro:
Physics-Based Animation
Publicado en inglés por Cengage Learning/ Delmar Charles River Media, INC
© 2005
ISBN 10: 1-58450-380-7

Datos para catalogación bibliográfica:
Física para videojuegos
Erleben, Kenny, Jon Sporring, Knud Henriksen y Henrik Dohlmann
ISBN-13: 978-607-481-506-1

Visite nuestro sitio en:
http://latinoamerica.cengage.com

Impreso en México
1 2 3 4 5 6 7 14 13 12 11

Contenido



1

Introducción

Una meta a largo plazo en las gráficas computacionales es aumentar el realismo y la *credibilidad* en animaciones y películas generadas en computadora. Entre los trabajos originales destacan los de Armstrong *et al*. (1985), Terzopoulos *et al*. (1987), Moore *et al*. (1988), Hahn (1988), Barzel *et al*. (1988), Platt *et al*. (1988), Baraff (1989) y Barsky *et al*. (1991). La creencia general es que al mejorar y modernizar la creación de imágenes, la falta de *realismo físico* y credibilidad será más obvia y en consecuencia más molesta para el observador común. El argumento principal para lograr la meta de más realismo ha sido usar la física para modelar el comportamiento y el movimiento de los modelos en computadora. Hoy esos esfuerzos han culminado en lo que se suele llamar *animación* o modelado *basado en la física*.

La animación basada en física es un campo altamente interdisciplinario que se basa en teorías de la ingeniería (Zienkiewicz *et al*., 2000; Stewart *et al*., 1996; Pfeiffer *et al*., 1996b; Enright *et al*., 2002), de la física (Baraff, 2001) y de las matemáticas. Algunos de los modelos de simulación más notables se basan en la robótica (Craig, 1986; Featherstone, 1998) y en la mecánica de los sólidos (House *et al*., 2000; Teran *et al*., 2003). Hasta donde sabemos, el campo de la animación basado en la física fue bautizado en un curso de la Association for Computing Machinery's Special Interest Group on Computer Graphics (ACM SIGGRAPH) en 1987, en la conferencia "Temas de modelado basado en la física" organizado por Alan H. Barr. En años recientes, el énfasis hacia la animación basada en física sobre algoritmos computacionalmente eficientes ha sembrado un campo nuevo: la *simulación plausible* (Barzel *et al*., 1996).

En una segmentación en la producción de una película, en general se cree que el uso de la física inhibe la creatividad y la estética de un animador. La razón es bastante obvia: es difícil confiar en un modelo físico y al mismo tiempo usar todos los aspectos de los *principios de animación* (Lassiter, 1987; Frank *et al*., 1995). Si bien algunos principios se concentran en la implementación de la realidad física, como *aplastar y estirar*, *tiempos y movimientos* y *desacelerar y acelerar*, la mayor parte se refieren a exagerar el movimiento para entretener a la audiencia y mantener su atención. Por lo anterior, los animadores prefieren movimientos surrealistas en lugar del entretenimiento, y no se tientan el corazón para dotar a sus personajes de poderes sobrenaturales. Es así que siguen un método apoltronado de sólo representar lo que sea visible, siempre que el movimiento y la escena muestren una historia interesante.

En este libro no se exponen los principios de la animación, y tampoco se intenta calificar o cuantificar la credibilidad o plausibilidad de los métodos y algoritmos presentados. No le conciernen el proceso de dibujar o los sistemas de animación. En su lugar, está dedicado ante todo a presentar al lector un fundamento firme y sólido para comprender cómo se deducen los modelos matemáticos a partir de principios físicos y matemáticos; en segundo lugar, está destinado a estudiar la forma en que se resuelven los modelos matemáticos en forma eficiente, firme y estable en una computadora. Como tal, esta obra presenta al lector los campos de la física, matemáticas y análisis numérico; sin embargo, no sustituye a los libros de texto en esos campos.

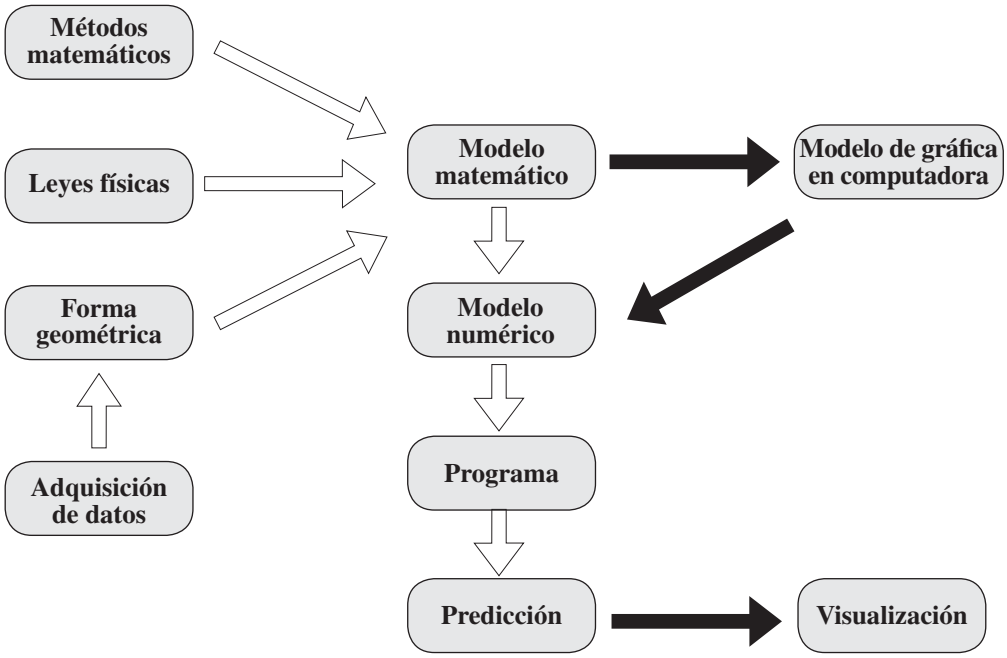


Figura 1.1 Perspectiva esquemática de la animación y simulación basada en la física. Las flechas blancas son típicas de las disciplinas de ingeniería, mientras que las flechas negras son pasos adicionales usuales que se toman en gráficas computacionales.

1.1 Modelo de gráficas computacionales

Una perspectiva de la animación basada en la física consiste en muchas leyes teóricas y métodos de física y matemáticas, con algo de geometría adicionada, y todo esto se mezcla para obtener un modelo matemático del mundo real que, una vez obtenido, se puede transformar en un modelo numérico que a su vez se programa en una computadora. Entonces el programa puede hacer predicciones sobre el mundo real que se usan para estimar dónde cabe esperar que se muevan los objetos (cinemática y dinámica directas), o bien para calcular la forma en que deben manejarse para obtener cierto movimiento deseado (cinemática y dinámica de retroceso). Esta concepción del mundo se esquematiza en la figura 1.1 con las flechas en blanco, y es típica de las disciplinas de ingeniería. En este campo, una meta frecuente es hacer predicciones muy precisas sobre el mundo real a largo plazo, es decir, asegurar la estabilidad de un puente o pronosticar la eficiencia de un sistema de calentamiento.

Usando como ejemplo un simulador quirúrgico, describiremos los pasos típicos indicados en la figura 1.1: el punto inicial es casi siempre una especie de paso de *adquisición y modelado de datos*, en el que se escanea a un paciente, y la imagen tridimensional que resulta se segmenta en objetos anatómicos importantes, que se representan como *formas geométricas*. Esos objetos se amplifican entonces con *leyes físicas* relevantes, con frecuencia en forma de un sistema de energía. Para hacer una simulación física, el sistema de energía se analiza con *métodos* o *herramientas matemáticas* como cálculo de variaciones, y el sistema se convierte en un conjunto de ecuaciones diferenciales parciales. Reiteradamente, el *modelo matemático* es un conjunto de ecuaciones diferenciales parciales junto con una descripción detallada de la geometría que funciona como

condiciones en la frontera. Para implementar el modelo matemático en una computadora, a menudo se discretizan las ecuaciones, es decir, aplicando el método adecuado de diferencias finitas, que a su vez se programa en C++ o en algún otro lenguaje similar de programación. Por último, el simulador puede correr y usarse para hacer *predicciones* sobre el futuro, es decir, sobre qué pasará si ciertas estructuras se cambian o eliminan.

No obstante, aun con los impresionantes logros en las disciplinas físicas, matemáticas y de ingeniería, con frecuencia sucede que ni el modelo matemático ni las técnicas numéricas son adecuados para la animación en computadora. Ésta favorece el gran detalle visual, el alto rendimiento y el poco consumo de memoria. En contraste, los métodos de ingeniería con frecuencia no se interesan en los detalles visuales momentáneos, ya que su objetivo principal muchas veces es predecir el comportamiento a largo plazo y/o la gran precisión. En consecuencia, a menudo el tiempo de cómputo y el consumo de memoria son altos, ya que si la tarea es predecir si un puente durará 100 años, entonces se justifica hacer que decenas o cientos de computadoras resuelvan el problema en varios días o semanas. Sin embargo, en la animación por computadora más bien podríamos visualizar la caída de un puente, con atractivo visual, en una computadora *laptop* en tiempo real. Debido a esas diferencias entre las gráficas de ingeniería y en computadora, los científicos informáticos modifican los modelos matemáticos para favorecer los detalles visuales y escogen técnicas numéricas que propicien la rapidez, fortaleza y estabilidad, más que la exactitud y la convergencia. Estas alternativas pueden llamarse *modelo de gráficas computacionales*. Resumiendo, los objetivos de la animación basada en física consisten en modelar el mundo real, pero a diferencia de las disciplinas de ingeniería, los métodos de gráficas en computadora favorecen el atractivo visual y los pocos recursos computacionales.

1.2 Taxonomía de los métodos de animación basada en la física

Al máximo nivel, el campo de la animación y la simulación basada en la física se pueden subdividir en primer lugar en dos grandes grupos:

Cinemática. Estudio del movimiento sin tener en cuenta masas o fuerzas.

Dinámica. Estudio del movimiento teniendo en cuenta masas y fuerzas.

El asunto no termina allí, porque la cinemática y la dinámica tienen dos subgrupos:

Reversa (o inversa). Estudio del movimiento conociendo los puntos inicial y final.

Directo (o avance). Estudio del movimiento dado sólo en el punto de partida.

En el primer subgrupo, el caso típico es cuando se conoce adónde ir, pero se necesita saber cómo hacerlo. Por ejemplo, uno podría conocer la posición final de un marco de herramientas de un manipulador robótico sin saber qué fuerzas y pares aplicar a los actuadores para llegar al destino final. En otras palabras, la cinemática y la dinámica inversas calculan el movimiento "en reversa". La cinemática y dinámica directas hacen exactamente lo contrario. Para el mismo ejemplo, lo que se conoce es la posición inicial del marco de herramientas, así como las fuerzas y pares que desarrollan los actuadores. Entonces, la meta es predecir el destino final. En la figura 1.2 se muestra la diferencia entre cinemática/dinámica de avance y reversa.

Hay numerosas técnicas y métodos en la animación y simulación basadas en física, y es probable que el lector ya haya oído hablar de ellos. En la tabla 1.1 hemos tratado de clasificar algunas de las técnicas y métodos más populares, de acuerdo con los cuatro subgrupos presentados arriba.

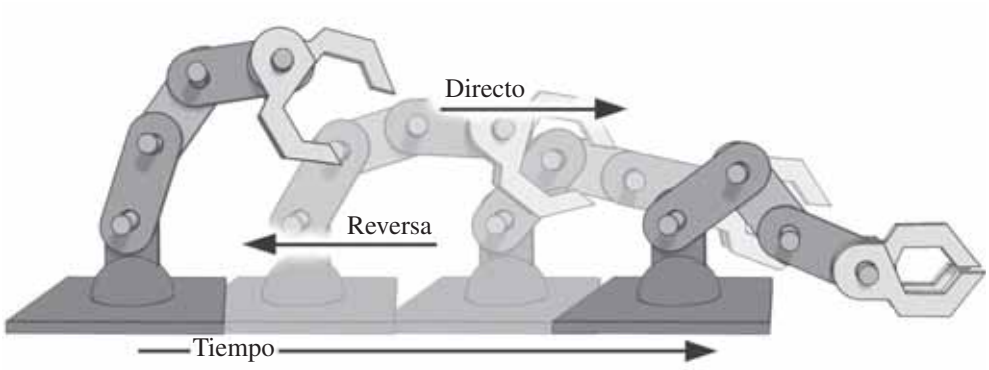


Figura 1.2 Ejemplo que muestra la diferencia entre cinemática/dinámica directa y en reversa. En el caso del movimiento inverso, se dan las posiciones inicial y final del sujetador robótico, mientras que en el caso del movimiento directo, sólo se conoce la posición inicial.

	Reversa	**Avance**
Cinemática	• Descenso cíclico en coordenadas • Método jacobiano	• Animación basada en tiras • Animación de marco clave (interpolación) • Soluciones de forma cerrada • Deformación de forma libre
Dinámica	• Método recursivo de Newton Euler • Problemas de optimización	• Método de Featherstone (método de cuerpo articulado) • Método del cuerpo rígido compuesto • Sistemas de partículas, sistemas masa-resorte • Método de elemento finito • Simulación basada en restricción

Tabla 1.1 Métodos comunes y su clasificación en cinemática/dinámica directa y en reversa.

1.3 Cómputo científico *vs* gráficas computacionales en la práctica

El campo de *cómputo científico* domina los métodos de ingeniería. Al conocer algunas aplicaciones, se aprecia que la diversidad es grande. A continuación se describen ejemplos de simuladores actuales para proyectos reales. Ya que los autores son daneses, la mayor parte de los ejemplos son daneses; sin embargo, vemos que la variedad de problemas y sus demandas computacionales son internacionales.

Túneles de viento numéricos en RISOE

El túnel de viento numérico en RISOE (`www.risoe.dk/vea-aed/numwind`) usa un generador de malla hiperbólica para dominios tanto bi como tridimensionales. La ecuación de Navier-Stokes se resuelve usando el programa EllipSys2D/3D, que es un código de volumen finito incompresible. El cálculo de un rotor de molino de viento estacionario toma unas 50 CPU horas. Al usar 14 CPU (3.2 GHz Pentium M, RAM de 2 GB) en grupo, la simulación tarda unas cuatro horas. Los cálculos no estacionarios tardan de tres a cuatro veces más (Johansen, 2005).

Deformación de materiales a escala atómica

En materiales a escala atómica, las simulaciones físicas de deformación plástica en cobre nanocristalino se hacen en un grupo (Niflheim, 2.1 Teraflops) de PC ordinarias. Una simulación comprende 100 nodos que corren en paralelo y con frecuencia tarda semanas en completarse (`www.dcsc.dk`). En química cuántica, los problemas contienen a menudo 10^9 variables y las simulaciones pueden durar hasta 10 días o más (Rettrup, 2005).

Mecánica de fluidos computacional

La mecánica de fluidos computacional se aplica a una gran variedad de problemas de flujo. En un caso típico, los flujos tridimensionales no estacionarios requieren $10 - - 100 \times 10^6$ nodos de red y usan hasta 1 000 CPU horas por simulación (`www.dcsc.dk`). También se simula la mecánica de sólidos (`www.dcsc.sdu.dk`), pero con frecuencia no se visualiza (Vinter, 2005).

Astrofísica computacional

La astrofísica computacional (`www.nbi.ku.dk/side22730.htm`) consiste en simular la formación de galaxias, estrellas y planetas. Algunos métodos que se usan son hidrodinámica suavizada de partículas y refinamiento adaptativo acoplado. Con frecuencia, los cálculos tardan semanas o meses en completarse (Nordlund, 2005).

Reportes meteorológicos

Los reportes meteorológicos se simulan y guardan periódicamente en disco en el DMI (`www.dmi.dk`). La información se resguarda cada tres horas. En 2.5 nodos de red se calculan reportes de 48 horas usando el tamaño de incremento de tiempo de dos minutos. La cantidad total de cálculos es del orden de 10^{12} y se resuelven en supercomputadoras muy grandes (Sørensen, 2005).

En esta breve reseña de computación científica es evidente que las simulaciones se basan en grandes supercomputadoras, con frecuencia agrupadas. La cantidad de datos es astronómica y los tiempos de cálculo van desde minutos a horas, días o semanas, y no son raros los que tardan meses. La visualización va desde flechas sencillas que ilustran campos de flujo hasta visualización científica bastante avanzada.

Algo importante a destacar es que con frecuencia el concepto de tiempo real no es un término muy útil en el cómputo científico. Por ejemplo, el tiempo simulado en química es del orden de picosegundos, pero el cálculo tarda días en terminarse. La idea principal de las simulaciones es, a menudo, ver un proceso químico en cámara lenta para observar qué sucede. Eso no se puede hacer en el laboratorio. En contraste, las simulaciones gráficas astrofísicas y oceánicas son demasiado lentas y en consecuencia se realizan a mayores velocidades. En simulaciones de flujo marino, se representa un par de siglos en un día. En conclusión, en cómputo científico los resultados se obtienen en lapsos que van de 24 horas a 30 días (Vinter, 2005).

En cuanto a la animación basada en física, el panorama es un poco diferente, desde la perspectiva del cómputo gráfico.

Fenómenos de fluidos

La simulación de humo para representar fenómenos en gran escala (Rasmussen *et al*., 2003) requiere el uso de medio millón de partículas, se tarda de 2 a 10 segundos por cuadro y el dibujo toma de 5 a 10 minutos por cuadro (2.2 GHz, Pentium 4). En McNamara *et al*. (2003), el control clave de cuadros en simulaciones de humo tarda de 2 a 24 horas en ejecutarse en un procesador Pentium 4 de 2 GHz. La simulación de humo dirigido al objetivo (Fattal *et al*., 2004) tarda 50 segundos en un procesador Pentium 4 de 2.4 GHz en 2D, en una red de 256^2 para un segundo de animación; en 3D, en una red de 128^3, toma 35 minutos.

Sistemas de partículas

En Feldman *et al*. (2003), las explosiones con partículas suspendidas son animadas. El tiempo de simulación va de 4 a 6 segundos por cuadro en un Pentium 4 de 3 GHz. Incluyendo los tiempos de creación de imagen, un segundo simulado es del orden de minutos. La animación de fluidos viscoelásticos (Goktekin *et al*., 2004) con una red de 40^3 se tarda media hora por segundo de animación en un Pentium 4 de 3 GHz. Para la animación de interacción bidireccional entre objetos rígidos y fluidos (Carlson *et al.*, 2004) usando una red de $64 \times 68 \times 84$ con un Pentium 4 de 3 GHz y 1 GB de RAM, se necesitan 401 pasos de simulación por cada segundo de animación, con un tiempo promedio de CPU por simulación de 27.5 segundos.

Animación de telas

La simulación de telas sin penetración (Bridson *et al*., 2002) en un trozo de 150×150 nodos corre a dos minutos por cuadro en un Pentium 3 de 1.2 GHz. Para desenredar telas (Baraff *et al*., 2003b) en un modelo con 14 K vértices, el costo adicional es 0.5 de segundos de tiempo de simulación por cuadro en un Pentium 4 de 2 GHz. El cambio de topología de malla (Molino *et al*., 2004) durante la simulación, con 1 K triángulos, se tarda de 5 a 20 minutos por cuadro, y con 380 K tetraedros toma 20 minutos. La simulación de cuerpo rígido apilado (Guendelman *et al*., 2003) con simulaciones de 500 a 1 000 objetos se tarda, en promedio, de 3 a 7 minutos por cuadro.

Videojuegos

Las consolas de juegos, como Play Station 2 (`www.playstation.com`) sólo tienen 32 MB de RAM y 6.2 GFLOPS, y las PC domésticas normales son en su mayor parte inferiores a las que usan los investigadores de gráficas computarizadas. Esto ha requerido cierto razonamiento creativo en el desarrollo de videojuegos para conseguir los requisitos de desempeño. Además, debe haber tiempo aparte para otras tareas en aplicaciones como los videojuegos. Por ejemplo, en *Hitman* de IO-Interactive, sólo se usa de 5 a 10% del tiempo de cuadro para simulación física (Jakobsen, 2005). En el futuro esas limitaciones de hardware pueden cambiar. Por ejemplo, las GPU recientes no tienen problemas

de latencia y serían posibles 60 GFLOPS (`www.nvidia.com`) en ellas; también las PPU (`www.ageia.com`) parecen ser una tecnología emergente. Por último, los chips celulares (`www.ibm.com/news/us/en/2005/02/2005_02_08.html`) también prometen.

Después de conocer esta pequeña encuesta de la literatura sobre la animación basada en física sobre gráficas computacionales, es evidente que se usan tiempos de cuadro que van del orden de segundos a horas, corriendo en PC solas. En conclusión, el trabajo de diseño para gráficas computacionales y cómputo científico requiere tiempos de terminación razonablemente bajos.

1.4 Puntos de estudio en el futuro

La animación basada en física es un campo grande y crece rápidamente. Cada año se presentan nuevas técnicas para animar fenómenos nuevos y los métodos existentes mejoran en velocidad, exactitud y detalle visual. No es posible explicar todo en un libro, y no se necesita decir que el mantenerse actualizado es una actividad importante de todo animador basado en física. Un buen lugar para comenzar a conocer las normas de la industria son los motores físicos, muchos de los cuales se muestran en la tabla 1.2.

En seguida mencionamos brevemente algunos tópicos que nos parecen interesantes, pero no los expondremos en este libro por limitaciones de espacio y por razones pedagógicas.

Métodos recursivos para mecánica con juntas

Los *métodos recursivos* para mecánica con juntas también se llaman *métodos de coordenada mínima,* y en la literatura se explica extensamente la teoría; véase, por ejemplo, Featherstone (1998).

Programación matemática

La *programación matemática* es, entre otras cosas, el estudio de problemas de complementariedad. La historia de esta disciplina comienza en la década de 1940. Es un campo gigantesco y por sí mismo merece varios libros de texto. En este libro recomendamos una biblioteca existente, como la de Path (2005) para un curso, o consultar, por ejemplo, Cottle *et al*. (1992) y Murty (1988).

Objetos deformables con *materiales de propiedades no lineales anisotrópicas*

Las deformaciones no lineales y grandes siguen siendo un problema en las investigaciones actuales y no son algo que creemos que se pueda incluir en un libro de texto introductorio.

Corte y fractura de objetos deformables

En fecha reciente se han publicado trabajos sobre fractura, pero el corte y la fractura están justo en la frontera de las aplicaciones actuales en tiempo real.

Métodos de conjunto de nivel de partículas y estructuras adaptativas de datos para simulación de agua

Hemos optado por enfocarnos en el método más clásico y presentamos al lector una buena comprensión de las bases. No se le debe dificultar el proseguir con las publicaciones recientes; véase, por ejemplo, Osher *et al*. (2003).

Control en animaciones basadas en física

En fecha reciente se ha tratado de incluir el control del animador usando métodos de dinámica en reversa.

Nombre	Notas
DynaMechs	S. McMillan inició su desarrollo en 1991 (`http://dynamechs.sourceforge.net`)
Renderware Physics	Unidad comercial de Criterion Software establecida en 1993 (`www.renderware.com/physics.asp`)
Havok	Fundado en 1998 (`www.havok.com`)
Meqon	Comenzó como proyecto universitario en 1999; D. Gustafsson y M. Lysén fundaron la compañía en 2002 (`www.meqon.com`)
Ipion	Comprado por Havok en junio de 2000 (`www.ipion.com`)
Open Dynamics Engine	Parece haberse iniciado alrededor de 2000 a 2001 por R. Smith, anterior empleado de MathEngine (`www.ode.org`)
Novodex	Comenzó en la primavera de 2002 con A. Moravansky y M. Müller (`www.novodex.com`)
Tokamak	D. Lam es el autor original de Tokamak, y parece haber iniciado en 2003 (`www.tokamakphysics.com`)
Newton Game Dynamics	Al parecer inició en 2003 (`www.physicsengine.com/`)
Karma	Desarrollado por MathEngine, fue adquirido por Criterion Software en julio de 2003 (`www.mathengine.com`)
Vortex	Desarrollado por CMLabs (`www.cm-labs.com`)
Free Cloth	D. Pritchard parece haberlo iniciado en 2002 (`sourceforge.net/projects/freecloth`)
OpenTissue	K. Erleben, H. Dohlmann, J. Sporring y K. Henriksen lo iniciaron en noviembre de 2001 (`www.opentissue.org`)
AGEIA	Empresa fundada en 2002; en marzo de 2005 AGEIA anuncia una nueva clase de unidad procesadora de física (PPU; *physics processing unit*), PhysX (`www.ageia.com`)

Tabla 1.2 Lista de motores físicos comerciales y públicos disponibles.

Hemos omitido toda teoría y métodos relacionados con cómo controlar y manipular una animación, porque no consideramos que sean parte de un motor físico de bajo nivel.

Detección continua de colisiones

Ciertos problemas tediosos en la animación son los efectos túnel y de sobrecorrección. Un remedio para esos problemas es la detección continua de colisiones, que promete generar un mejor punto de contacto y determinar áreas de contacto en simulaciones dinámicas. El campo es bastante nuevo, pero muy prometedor. Sin embargo, opinamos que todavía no hay una norma firmemente establecida, por lo que hemos omitido este material. Los lectores interesados pueden consultar Redon *et al*. (2002), Redon (2004a), Redon (2004b) y Redon *et al*. (2004a).

Métodos simplex para detección de colisiones

Los métodos simplex (plural de *simple*) para detección de colisiones se basan en el concepto matemático de un conjunto de variables linealmente independientes. De acuerdo con nuestra experiencia, esos métodos son de difícil acceso para los alumnos y no son esenciales, lo cual es la principal razón de porqué los hemos omitido. Unas buenas referencias son Bergen (2003b), Bergen (2001) y Bergen (1999).

1.5 Guía para el lector

Habiendo enseñado gráficas y animación por computadora durante varios años, hemos visto que carecemos de una introducción completa a la animación basada en la física. En consecuencia, primero comenzamos a escribir notas sobre los métodos matemáticos necesarios para los simuladores, y después notas para describir los diversos métodos de animación. Por eso los destinatarios principales de este libro son universitarios y profesionistas de ciencia informática. La preparación de nuestros alumnos suele versar sobre un buen conocimiento del arte de la programación, introducción a gráficas por computadora y lo que llamamos *madurez matemática*. Para nosotros, madurez matemática significa que los alumnos no necesariamente tienen conocimientos matemáticos rigurosos, pero les gusta aprender nuevas técnicas en esta disciplina. Por tanto, nuestro método es práctico: tratamos de enseñar teorías como las que presenta este libro, y subrayamos que los estudiantes deben poder implementar las teorías en simuladores en trabajo real. Esperamos que esta obra refleje esas metas, en su caso, por los numerosos ejemplos de pseudocódigo. Casi todos los algoritmos que se describen se han implementado en el proyecto adjunto de código abierto (OpenTissue, 2005).

El libro contiene cinco partes que reflejan la diversidad del campo de la animación basada en la física. Por consiguiente, se puede leer de las siguientes diversas maneras.

Geometría del movimiento

Para el lector interesado en la geometría del movimiento y la cinemática, y para quienes desean tener control total de las pautas de movimiento, sugerimos la parte I, que describe el diseño de parámetros de movimiento, independientemente de la física del mundo real.

De partículas a cuerpos rígidos

El punto de partida más común en la animación basada en la física son los sistemas de partículas (capítulo 8), seguido por el estudio de animación de cuerpos rígidos en la parte II; después por una investigación avanzada en objetos deformables en la parte III, para concluir con un estudio de dinámica de fluidos computacional en el capítulo 11.

Dinámica de fluidos computacional

Algunos lectores encontrarán que la dinámica de fluidos computacional es sencilla y optarán por leer el capítulo 8, seguido por el estudio de este tema en el capítulo 11 y los modelos de continuo con diferencias finitas en el capítulo 9. De acuerdo con nuestra experiencia, el análisis de elemento finito en el capítulo 10 es el gran reto para los novatos.

Detección de colisiones

La detección de colisiones se explica en la parte IV y con frecuencia se aborda en paralelo con los temas anteriores.

Compendio matemático y físico

La parte V está escrita como un apéndice muy amplio, y contiene gran parte de la física y las matemáticas necesarias para comprender las teorías en este libro.

¡Feliz lectura!

Parte I

La cinemática

La cinemática es el estudio del movimiento de las partes sin considerar masas ni fuerzas. En consecuencia, se pueden ignorar las leyes de Newton. La aplicación principal de los métodos cinemáticos es la animación preconfigurada, donde no se requiere simulación física, que es demasiado compleja para ocuparse de ella o no se apega a los *principios de animación* (Frank *et al.*, 1995; Lassiter, 1987). Muchas películas animadas por computadora usan mucho la cinemática para el movimiento; gran parte de la teoría puede considerarse así como una extensión natural de las técnicas de paro-movimiento desarrolladas por Disney a principios del siglo XX. Este campo se divide en cinemática directa y cinemática inversa, dependiendo de si se conoce o no la configuración final.

La cinemática directa (en avance) se caracteriza por el movimiento que ordena el usuario o alguna función. Requiere mucho trabajo manual; sin embargo, se usa mucho en la producción fílmica, donde los animadores especifican cada parámetro como una función del tiempo. Con frecuencia, ellos mismos hacen el movimiento de las figuras que van a animar. Para reducir el tiempo de producción se suele aplicar una técnica de visión computarizada llamada *captura del movimiento*.

La captura del movimiento es el proceso de rastrear puntos fijados a un actor mediante video, reconstruyendo el lugar de los puntos en tres dimensiones y usando la reconstrucción como entrada para la cinemática directa. Con este método los animadores pueden crear movimientos asombrosamente realistas que obedecen las leyes de la física. La captura de movimiento es una gran herramienta en la producción cinematográfica, donde se usa el movimiento reconstruido para especificar los parámetros de un movimiento específico. También se puede utilizar como instrumento para crear nuevas secuencias de movimiento para videojuegos. En este caso, los datos de captura de movimiento se consideran una muestra discreta de movimiento físico y se obtiene la nueva secuencia ajustando un modelo continuo a los datos discretos. Una de esas técnicas se llama *mezcla de movimiento* o *mezcla de animación*. Las técnicas de mezclado se encuentran a menudo en motores de juegos como los que se mencionan en `www.devmaster.net/`. Hay tres problemas básicos cuando se mezcla el movimiento.

Distorsión de tiempo-movimiento. Cómo adaptar las secuencias de movimiento para que coincidan en el tiempo. En otras palabras, se debe determinar qué cuadro en un movimiento corresponde a qué cuadro en otro movimiento.

Alineación del movimiento. Un movimiento puede ser hacia la izquierda y el otro hacia la derecha. Entonces, ¿qué posición debe tener la mezcla de movimientos?

Igualación de restricción del movimiento. Si se desea combinar un movimiento de correr y caminar, se deben hacer coincidir cuidadosamente los cuadros donde los pies correspondientes tocan el suelo, para no tener un resultado de deslizamiento o flotación.

La mezcla de movimiento se usa en videojuegos para crear transiciones entre datos de movimiento capturado y se ve en muchos juegos con tirador en tercera persona, como Hitman®. Con frecuencia, las técnicas son especiales y dependen mucho de que los animadores especifiquen completamente *a priori* las transiciones. Las investigaciones recientes se enfocan en hacer que los movimientos se combinen en forma más natural, teniendo en cuenta el equilibrio de una persona.

También se pueden generar los parámetros para la cinemática directa usando *programas de movimiento* (Barsky *et al.*, 1991). En estos programas se indican parámetros de junta en forma de alguna función continua del tiempo, como $\cos(t)$, que genera un movimiento oscilatorio. Las funciones de movimiento se deben diseñar con cuidado para el movimiento específico, como el modo de andar bípedo (Parent, 2001; cap. 6.2) o el nado de los peces (Terzopoulos *et al.*, 1994).

La cinemática inversa maneja la interpolación de una figura desde puntos inicial y final especificados. Se usa también con frecuencia en programas de animación como Maya® o 3DMax®. Aun para objetos simples, como el brazo humano, casi siempre habrá varias soluciones, ya que el codo se puede mover aun cuando la mano y el hombro estén fijos. Por consiguiente, un problema central en la cinemática inversa es seleccionar uno de varios movimientos posibles. Se busca entonces una solución que genere un movimiento de aspecto natural. Esto se logra adicionando restricciones extras a la cinemática inversa o combinándola con la dinámica, por ejemplo, usando principios de energía mínima.

Por último, muchas veces se combinan la cinemática directa y la inversa en aplicaciones reales. Por ejemplo, se puede recurrir a la primera para producir movimientos de aspecto natural en algunos objetos, y a la segunda para otros. Otro ejemplo es usar la captura de movimiento para generar estadísticas sobre movimientos realistas y utilizar esta información para seleccionar soluciones a problemas cinemáticos inversos. Muchos motores físicos admiten además la mezcla de cinemática directa e inversa con objetos físicamente simulados. Un caso típico en una producción cinematográfica es el manejo de la cinemática inversa o animación basada en la física para determinar un conjunto preliminar de parámetros para un movimiento específico. Después se calcula con detalle la animación, en la que se graban los parámetros de todos los objetos en cada cuadro. A esto se le llama *baking* (Kačić-Alesić *et al.*, 2003). Después de terminar los objetos animados, el animador puede hacer ajustes finos en los movimientos usando el método de cinemática directa para producir animaciones entretenidas. En conclusión, en las aplicaciones reales se usan las técnicas en forma indistinta y en combinación, como pasos de pre y posprocesamiento.

En la parte I describiremos, en los capítulos 2 y 3, los fundamentos de la cinemática directa e inversa de figuras articuladas, y el tema del movimiento programado en el capítulo 4.

2

Figuras articuladas

Este capítulo describe las *figuras articuladas* y su representación. Esa representación se usará después en animación y simulación de figuras articuladas, las cuales se pueden concebir como un brazo robótico o un brazo humano formados por varias barras macizas, conectadas entre sí por juntas que se mueven independientemente. Cuando se mueven todas las juntas, el movimiento general de una figura articulada puede ser muy complejo. El objetivo de este capítulo es presentar métodos que faciliten la descripción de la cinemática de figuras articuladas, que después se usará para animar y simular las figuras articuladas.

El capítulo está dividido en tres partes. La sección 2.1 define con precisión qué significa una figura articulada y cómo se construye. Las últimas dos secciones explican dos métodos diferentes para describir tales figuras. La sección 2.2 presenta un método general llamado *coordenadas de junta apareada,* y la sección 2.3 describe un método especializado denominado *Denavit-Hartenberg*.

2.1 Eslabones y juntas

Una *figura articulada* es una construcción formada por *eslabones* y *juntas*. Los diferentes eslabones están conectados por juntas, que tienen cierto grado de libertad. Se puede concebir un *eslabón* como una barra maciza que no puede cambiar su forma ni su longitud. Por consiguiente, se le considera un cuerpo rígido que define la relación entre dos ejes de juntas vecinas (véase la figura 2.1). Una junta se puede caracterizar como una conexión entre dos eslabones vecinos. Una junta tiene varios grados de libertad, es decir, puede girar respecto de *uno, dos* o *tres* ejes, o trasladarse a lo largo de *uno, dos* o *tres* ejes.

Un ejemplo de figura articulada con una junta "revoluta" (giratoria respecto de un pasador) se ve en la figura 2.1. Se puede decir que los eslabones y las juntas son los bloques de construcción que forman una figura articulada, por ejemplo, los robots industriales, la maquinaria de construcción y el esqueleto humano.

Los eslabones y las juntas se numeran de 0 a *N,* y una figura articulada siempre comienza con la $junta_0$, que está fija en cierto *sistema coordenado de base*. La numeración de los eslabones y juntas es muy importante. Una junta dentro de una figura articulada, como la $junta_i$, conecta al $eslabón_{i-1}$ con el $eslabón_i$, estando el $eslabón_{i-1}$ más cerca de la base.

En la figura 2.1 se aprecia una junta revoluta, es decir, que puede girar alrededor de *un* eje. En general, las juntas reciben el nombre según lo que hacen o pueden hacer:

Junta revoluta o rotativa. Junta que puede girar en torno a *uno* o *más* ejes. La figura 2.2a muestra una junta revoluta con *un* grado de libertad, pero en general ésta puede tener hasta tres grados de libertad, es decir, girar alrededor de los tres ejes coordenados.

Junta prismática. Junta que se puede trasladar a lo largo de *uno* o *más* ejes. La figura 2.2b muestra una junta prismática con *un* grado de libertad, pero en general puede tener hasta tres grados de libertad, es decir, trasladarse a lo largo de los tres ejes coordenados.

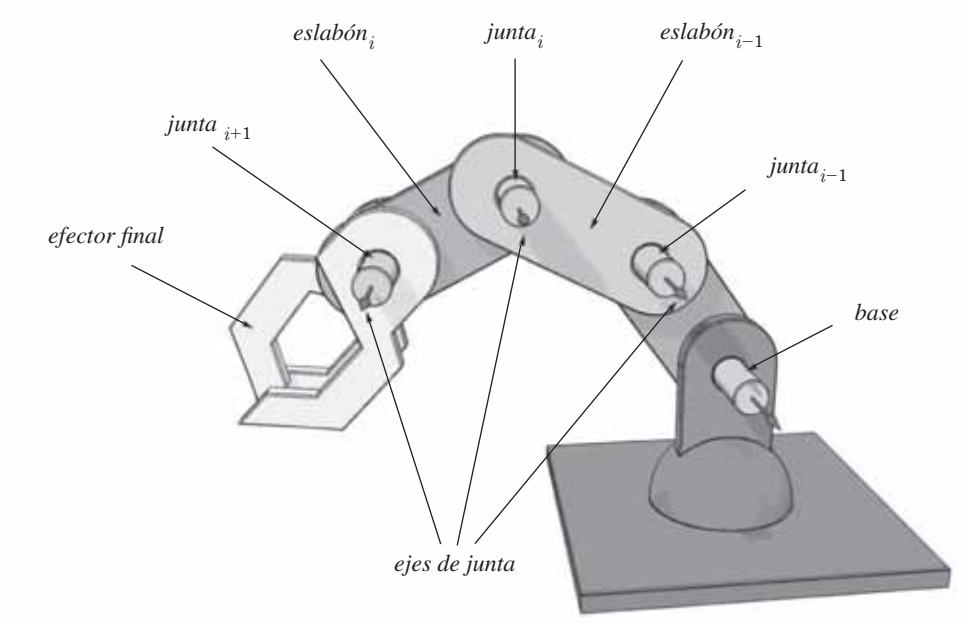
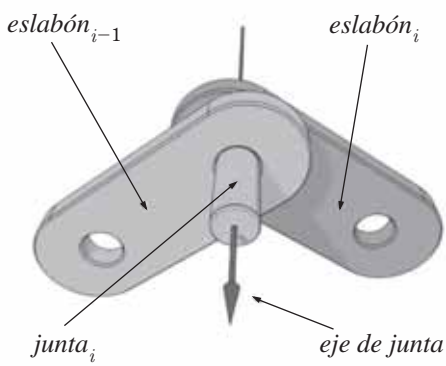
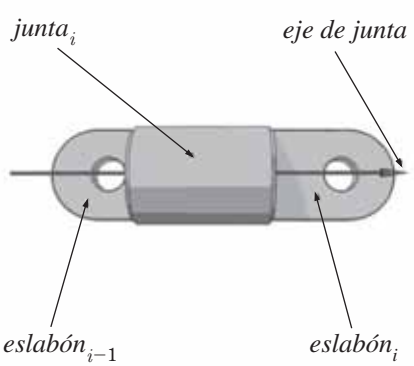


Figura 2.1 Ejemplo de figura articulada con una junta revoluta, la cual gira respecto de *un* eje. Es muy importante la numeración de las juntas, es decir, la junta$_i$ conecta el eslabón$_{i-1}$ con el eslabón$_i$.



***a*)** Junta revoluta (rotación)



***b*)** Junta prismática (traslación)

Figura 2.2 Ejemplos de diversas juntas que se usan para formar figuras articuladas.

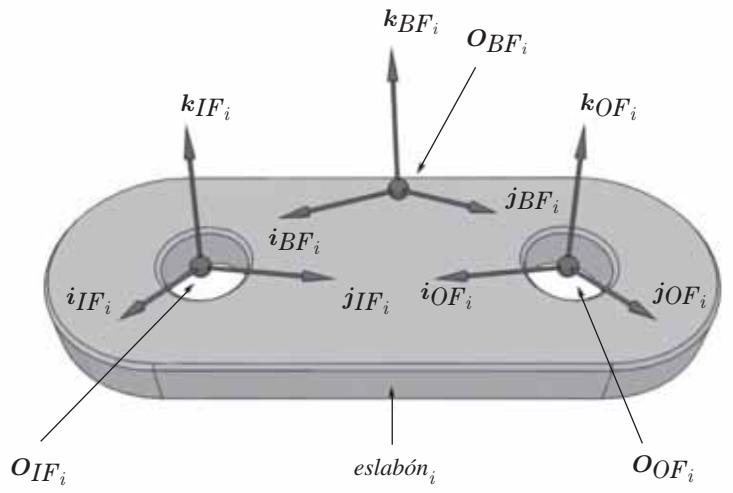


Figura 2.3 Los tres sistemas coordenados asociados con cada $eslabón_i$: BF_i, IF_i y OF_i. En este marco de cuerpo BF_i se especifican los orígenes y los ejes coordenados del marco interno IF_i y el marco externo OF_i.

2.2 Coordenadas apareadas de junta

Como se indicó antes, una figura articulada se construye con eslabones y juntas. Como cada junta puede girar o trasladarse, el movimiento de los eslabones y juntas individuales puede ser muy complejo. Eso se debe a que el movimiento de la $junta_j$ y el $eslabón_j$ afectan el movimiento de la $junta_i$ y del $eslabón_i$ cuando $i > j$. Por ejemplo, si la junta base gira o se traslada, su movimiento afecta a todas las demás juntas y eslabones de la figura articulada. En consecuencia, es muy complicado describir el movimiento de una figura articulada en un sistema coordenado común a todas las juntas y eslabones. Esa dificultad se puede superar introduciendo sistemas coordenados locales para todas las juntas y eslabones, estableciendo transformaciones entre estos sistemas.

Se puede describir una figura articulada mediante el método de *coordenadas apareadas de junta* (Featherstone, 1998). El método es muy general y se apoya en asociar tres sistemas coordenados predefinidos con cada eslabón. Para el $eslabón_i$, esos sistemas coordenados reciben los nombres de *marco de cuerpo* (BF_i), *marco interior* (IF_i) y *marco exterior* (OF_i), y todos se asocian con el $eslabón_i$. Los orígenes del marco interior y el exterior están en sus respectivos ejes de junta. Así se facilita transformar entidades entre eslabones sucesivos, como veremos después. Esos sistemas coordenados se ilustran en la figura 2.3 y se explican a continuación.

Marco del cuerpo (BF_i) (propiamente marco de referencia del cuerpo). Es un sistema coordenado *local* que se asocia con el $eslabón_i$, cuya geometría se describe en este sistema. En general, el origen y los ejes de este sistema coordenado se pueden elegir de manera arbitraria, pero para manejar con facilidad las figuras articuladas se recomienda definir al origen de BF_i como *el centro de masa* del $eslabón_i$. También se recomienda seleccionar un sistema de coordenadas ortogonales. Si el $eslabón_i$ tiene ejes

de simetría, se sugiere elegir algunos de ellos como ejes base para el sistema local de coordenadas ortogonales. En lo que sigue se supondrá que todos los sistemas son de este tipo.

El marco interior (IF_i) (propiamente: marco de referencia interior). Es un sistema coordenado *local* que se asocia con el $eslabón_i$. En general, este sistema se asocia con la $junta_i$. Tiene su origen en el eje de ésta y posee un eje paralelo a la dirección del movimiento de la junta. El origen y los ejes coordenados de este sistema de coordenadas se especifican en el marco del cuerpo.

El marco exterior (OF_i) (propiamente: marco de referencia exterior). Es un sistema coordenado *local* asociado con el $eslabón_i$. Usualmente, este sistema coordenado se asocia con la $junta_{i+1}$. Se origina en el eje de la $junta_{i+1}$, y tiene un eje paralelo a la dirección de movimiento de esa junta. El origen y los ejes coordenados de este sistema se especifican en el marco del cuerpo.

El resto de esta sección contiene subsecciones donde se deducen transformaciones entre los distintos marcos de referencia.

1. Calcular la transformación ${}^{BF_i}\boldsymbol{T}_{IF_i}$ del marco interior al marco del cuerpo (sección 2.2.1).
2. Calcular la transformación ${}^{BF_i}\boldsymbol{T}_{OF_i}$ del marco exterior al marco del cuerpo (sección 2.2.2).
3. Calcular la transformación ${}^{OF_{i-1}}\boldsymbol{T}_{IF_i}$ del marco interior del $eslabón_i$ al marco exterior del $eslabón_{i-1}$.
4. Calcular la transformación ${}^{(i-1)}\boldsymbol{T}_i$ que transforma entidades del marco del cuerpo del $eslabón_i$ al marco del cuerpo del $eslabón_{i-1}$.

La notación ${}^{HASTA}\boldsymbol{T}_{DESDE}$ quiere decir que la transformación modifica una entidad indicada en el marco con coordenadas *DESDE* a coordenadas en el marco *HASTA*.

Cuando todas las transformaciones se han derivado, es posible transformar las coordenadas de un punto $\boldsymbol{p}$ especificado en el marco del cuerpo del $eslabón_i$ a coordenadas en cualquier otro marco del cuerpo, por ejemplo, en el marco del cuerpo del $eslabón_j$. Eso es muy general, pero facilita transformar las coordenadas de un punto $\boldsymbol{p}$ especificado en cualquier marco del cuerpo del $eslabón_i$ al marco de referencia de coordenadas base, el marco del cuerpo del $eslabón_0$. Así es posible modificar posiciones, orientaciones, veloci-dades y aceleraciones de un punto $\boldsymbol{p}$, especificado en cualquier marco de cuerpo, hasta el marco base, que se puede usar como sistema coordenado común a todas las juntas y eslabones.

2.2.1 La transformación ${}^{BF_i}\boldsymbol{T}_{IF_i}$

En esta sección se describe cómo hacer una transformación (aplicación o $mapeo$) del marco interior al marco del cuerpo. Más específicamente, el problema es: dado un punto $\boldsymbol{p}$ en coordenadas de marco interno, determinar sus coordenadas en el marco del cuerpo.

Con las definiciones de $eslabón_i$ en la sección 2.2, sean $\boldsymbol{o}_{IF_i}$, $\boldsymbol{i}_{IF_i}$, $\boldsymbol{j}_{IF_i}$ y $\boldsymbol{k}_{IF_i}$ el origen y los vectores base del marco interno expresados en el marco del cuerpo (véase la figura 2.4).

La relación entre el marco interno y el marco del cuerpo es la siguiente: dado un punto $\boldsymbol{p} = (x, y, z)^T$ en el marco interno IF_i, sus coordenadas en el marco del cuerpo, BF_i, se pueden expresar con una rotación angular φ en torno a un eje $\boldsymbol{u}$ seguida por una traslación $\boldsymbol{r}_{IF_i}$. Esta transformación se llama ${}^{BF_i}T_{IF_i}$, en la que el subíndice IF_i y el superíndice BF_i indican que las coordenadas de un punto $\boldsymbol{p}$ especificado en coordenadas del marco interior se transforman en coordenadas del marco del cuerpo.

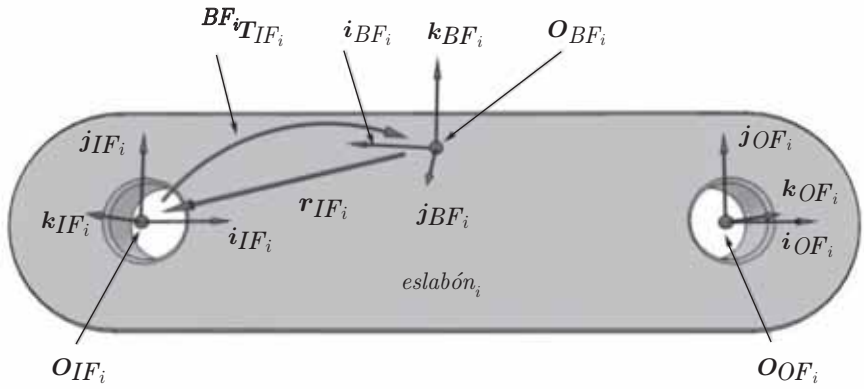


Figura 2.4 Transformación entre el marco interior IF_i y el marco del cuerpo BF_i para el $eslabón_i$. El origen y los ejes coordenados del marco interior IF_i se especifican en el marco del cuerpo BF_i.

En coordenadas homogéneas, la transformación del marco interior al marco del cuerpo se indica con una matriz:

$$^{BF_i}\boldsymbol{T}_{IF_i} = \boldsymbol{T}_{IF_i}(\boldsymbol{r}_{IF_i})\boldsymbol{R}_{IF_i}(\varphi_{IF_i}, \boldsymbol{u}_{IF_i}) \tag{2.1}$$

Las matrices de traslación y rotación son las siguientes.

$$\boldsymbol{T}_{IF_i}(\boldsymbol{r}_{IF_i}) = \begin{bmatrix} \mathbf{1} & \boldsymbol{r}_{IF_i} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.2a}$$

$$\boldsymbol{R}_{IF_i}(\varphi_{IF_i}, \boldsymbol{u}_{IF_i}) = \begin{bmatrix} \boldsymbol{i}_{IF_i} & \boldsymbol{j}_{IF_i} & \boldsymbol{k}_{IF_i} & \mathbf{0} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2b}$$

donde el símbolo **1** representa una matriz identidad de 3×3 y los vectores $\boldsymbol{i}_{IF_i}$, $\boldsymbol{j}_{IF_i}$, $\boldsymbol{k}_{IF_i}$ y $\mathbf{0}$ son vectores columna 3×1. Con ello ambas matrices se vuelven de 4×4. La matriz de traslación y la matriz de rotación son constantes y sólo dependen de la posición y orientación relativa de los marcos interior y del cuerpo. Por consiguiente, la transformación resultante $^{BF_i}\boldsymbol{T}_{IF_i}$ también es constante.

2.2.2 La transformación $^{BF_i}\boldsymbol{T}_{OF_i}$

En esta sección se describe cómo hacer una transformación del marco interior al marco del cuerpo. Más específicamente, el problema es: dado un punto $\boldsymbol{p}$ en las coordenadas del marco exterior, determinar sus coordenadas en el marco del cuerpo.

Recordemos las definiciones de $eslabón_i$ en la sección 2.2; sean $\boldsymbol{O}_{OF_i}$, $\boldsymbol{i}_{OF_i}$, $\boldsymbol{j}_{OF_i}$ y $\boldsymbol{k}_{OF_i}$ el origen y los vectores base del marco exterior, expresados en el marco del cuerpo (véase la figura 2.5). Se puede especificar una transformación entre el marco exterior OF_i y el marco del cuerpo, BF_i, definiendo el origen y los vectores base del marco exterior en el sistema coordenado del marco del cuerpo. Representemos por $^{BF_i}\boldsymbol{T}_{OF_i}$ esta transformación, donde el subíndice OF_i y el superíndice BF_i indican que las coordenadas de un punto $\boldsymbol{p}$ expresadas en coordenadas del marco exterior se transforman en coordenadas de marco del cuerpo.

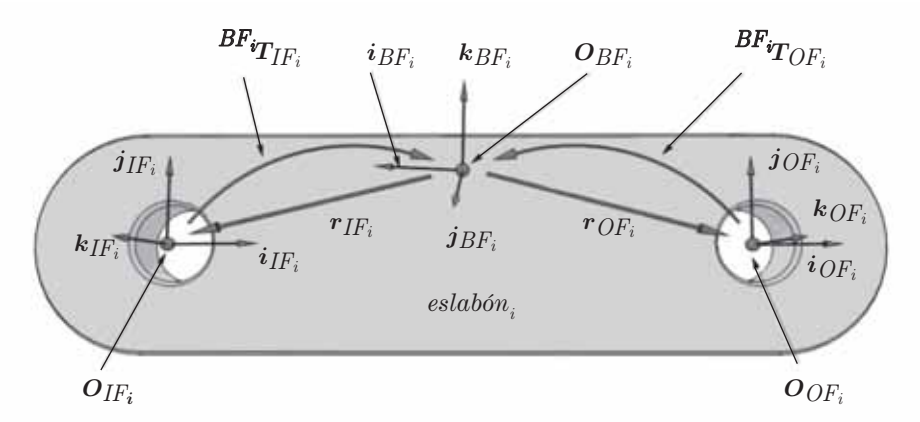


Figura 2.5 Transformación entre el marco exterior OF_i y el marco del cuerpo BF_i para el $eslabón_i$. El origen y los ejes coordenados del marco exterior OF_i se especifican en el marco del cuerpo BF_i.

En coordenadas homogéneas, la transformación del marco exterior al marco del cuerpo se indica con una matriz

$$^{BF_i}\boldsymbol{T}_{OFi} = \boldsymbol{T}_{OFi}(\boldsymbol{r}_{OFi})\boldsymbol{R}_{OFi}(\varphi_{OF_i}, \boldsymbol{u}_{OF_i}) \tag{2.3}$$

Las matrices de traslación y rotación son las siguientes.

$$\boldsymbol{T}_{OF_i}(\boldsymbol{r}_{OF_i}) = \begin{bmatrix} \mathbf{1} & \boldsymbol{r}_{OF_i} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.4a}$$

$$\boldsymbol{R}_{OF_i}(\varphi_{OF_i}, \boldsymbol{u}_{OF_i}) = \begin{bmatrix} \boldsymbol{i}_{OF_i} & \boldsymbol{j}_{OF_i} & \boldsymbol{k}_{OF_i} & \mathbf{0} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4b}$$

siendo **1** una matriz identidad de 3×3 y los vectores $\boldsymbol{i}_{OF_i}$, $\boldsymbol{j}_{OF_i}$, $\boldsymbol{k}_{OF_i}$ y **0** son vectores columna de 3×1, con lo cual ambas matrices son 4×4. Tanto la matriz de traslación como la matriz de rotación son constantes, y sólo dependen de la posición y orientación relativa del marco interno y el marco del cuerpo. En consecuencia, la transformación resultante $^{BF_i}\boldsymbol{T}_{OF_i}$ también es constante.

2.2.3 La transformación $^{OF_{i-1}}\boldsymbol{T}_{IF_i}(\boldsymbol{d}_i, \varphi_i, \boldsymbol{u}_i)$

En las secciones anteriores se dedujeron las transformaciones de eslabón $^{BF_i}\boldsymbol{T}_{IF_i}$ y $^{BF_i}\boldsymbol{T}_{OF_i}$ (véanse las secciones 2.2.1 y 2.2.2). Esas transformaciones son locales respecto del eslabón en cuestión, que es el *eslabón*$_i$. En esta sección se deducirá una transformación del marco interior del $eslabón_i$ al marco exterior del $eslabón_{i-1}$. La representaremos como $^{OF_{i-1}}\boldsymbol{T}_{IF_i}$.

El $eslabón_{i-1}$ y el $eslabón_i$ están conectados por la $junta_i$. La relación entre el marco interior del $eslabón_i$ y el marco exterior del $eslabón_{i-1}$ se define con una transformación conjunta, formada por una rotación (φ_i, $\boldsymbol{u}_i$) y una traslación $\boldsymbol{d}_i$:

$$^{OF_{i-1}}\boldsymbol{T}_{IF_i}(\boldsymbol{d}_i, \varphi_i, \boldsymbol{u}_i) = \boldsymbol{T}_i(\boldsymbol{d}_i)\boldsymbol{R}_i(\varphi_i, \boldsymbol{u}_i) \tag{2.5}$$

donde φ_i es el ángulo de rotación, $\boldsymbol{u}_i$ es el eje de rotación y $\boldsymbol{d}_i$ es el vector de traslación (véase la figura 2.6). Para facilitar la notación, se omite el índice i en las dos ecuaciones siguientes. Eso quiere decir que $\varphi_i = \varphi$,

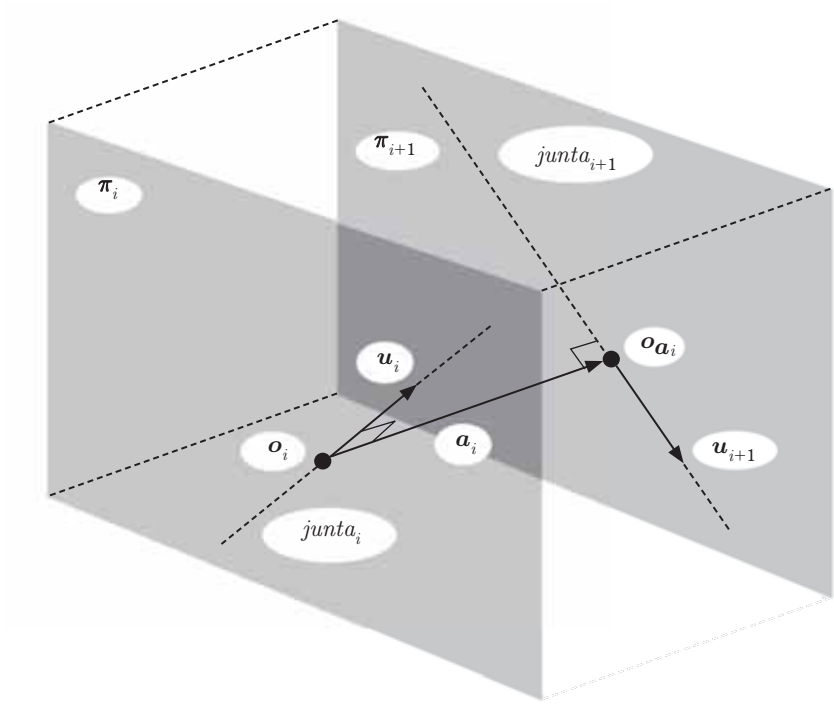


Figura 2.9 Los ejes de la $junta_i$ y la $junta_{i+1}$ están contenidos en dos planos paralelos π_i y π_{i+1}, respectivamente. La distancia perpendicular entre los planos π_i y π_{i+1} es igual a $\|\boldsymbol{a}_i\|_2$.

El vector $\boldsymbol{c}_i = \boldsymbol{u}_i \times \boldsymbol{u}_{i+1} / \|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2$ puede ir tanto del eje de la $junta_i$ al eje de la $junta_{i+1}$ como en dirección opuesta, dependiendo de las orientaciones de los vectores $\boldsymbol{u}_i$ y $\boldsymbol{u}_{i+1}$. Si $\boldsymbol{c}_i$ va del eje de la $junta_{i+1}$ al eje de la $junta_i$, el producto punto $(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i) \cdot \boldsymbol{c}_i$ es negativo y el vector eslabón $\boldsymbol{a}_i$ tendrá la orientación correcta.

En el otro caso, cuando el vector $\boldsymbol{c}_i$ va del eje de la $junta_i$ al eje de la $junta_{i+1}$, como se ve en la figura 2.10, es obvio que la orientación será correcta.

2.3.1.5 Caso especial: los ejes de la junta se intersecan

Si los ejes de la $junta_i$ y la $junta_{i+1}$ se cruzan, entonces la distancia más corta a_i entre ellos es igual a cero, y el vector eslabón será el vector nulo, que tiene cualquier dirección. En este caso, se selecciona la longitud de eslabón $a_i = 0$ y el vector eslabón para que sea igual al producto cruz entre los vectores de dirección $\boldsymbol{u}_i$ y $\boldsymbol{u}_{i+1}$, entre los ejes de la $junta_i$ y la $junta_{i+1}$. Esto es, se seleccionan

$$\boldsymbol{a}_i = \frac{\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2} \tag{2.39a}$$

$$a_i = 0 \tag{2.39b}$$

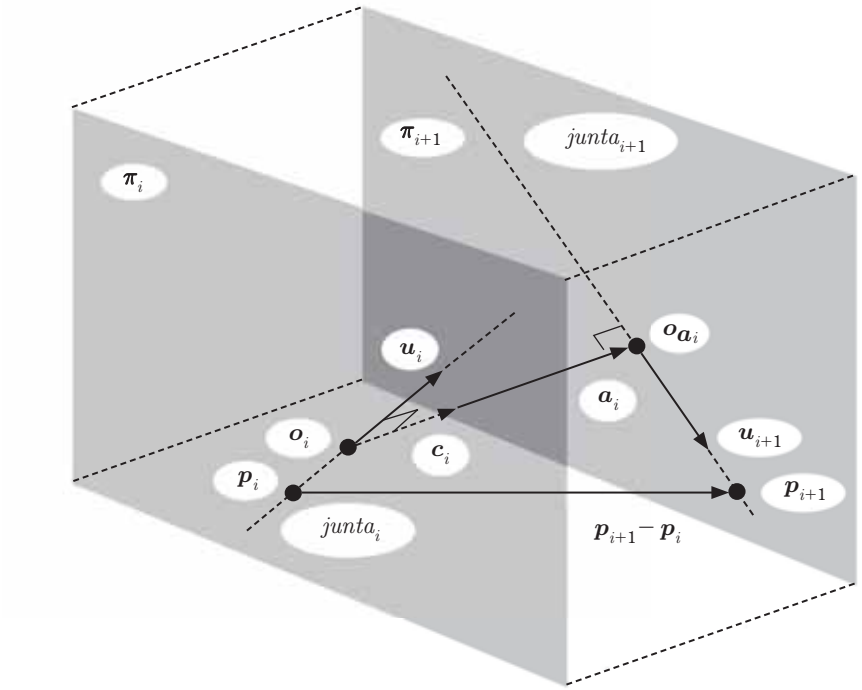


Figura 2.10 Una manera fácil de calcular la longitud de eslabón $\boldsymbol{a}_i$. Sea el vector $\boldsymbol{c}$ igual al producto cruz unitario entre los vectores de dirección de los ejes, es decir, que el vector $\boldsymbol{c}_i$ se define como $\boldsymbol{c}_i = \frac{\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2}$. También, sean $\boldsymbol{p}_i$ y $\boldsymbol{p}_{i+1}$ puntos arbitrarios en los ejes, por ejemplo, los que se indican en (2.10). Entonces, la longitud de eslabón es igual al producto punto entre el vector $\boldsymbol{c}_i$ y el vector $(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i)$. Esto es, $a_i = (\boldsymbol{p}_{i+1} - \boldsymbol{p}_i) \cdot \boldsymbol{c}_i$.

Para este caso especial conviene seleccionar un vector eslabón extraño tal que no siga la ecuación (2.38), porque la longitud de eslabón a_i es un parámetro de Denavit-Hartenberg, y el vector eslabón $\boldsymbol{a}_i$ se usa para asignar el marco de coordenadas, el eje $\boldsymbol{x}_i$ del $eslabón_i$. Además, el origen $\boldsymbol{o}_i$ y el punto $\boldsymbol{o}_{\boldsymbol{a}_i}$ están definidos en forma única como el punto de intersección.

2.3.1.6 Caso especial: los ejes de junta son paralelos

Si los ejes de la $junta_i$ y de la $junta_{i+1}$ son paralelos, no hay una distancia más corta única entre ellos (véase la figura 2.11). En este caso especial se calculan como sigue el vector eslabón y su longitud:

$$\boldsymbol{a}_i = (\boldsymbol{p}_{i+1} - \boldsymbol{p}_i) - \left((\boldsymbol{p}_{i+1} - \boldsymbol{p}_i) \cdot \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2} \right) \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2} \tag{2.40a}$$

$$a_i = \|\boldsymbol{a}_i\|_2 \tag{2.40b}$$

donde $\boldsymbol{p}_i$ y $\boldsymbol{p}_{i+1}$ son puntos conocidos, y $\boldsymbol{u}_i$ y $\boldsymbol{u}_{i+1}$ son vectores de dirección conocidos de los ejes de la $junta_i$ y la $junta_{i+1}$ (véase 2.10). El origen $\boldsymbol{o}_i$ se puede seleccionar en forma arbitraria y el punto $\boldsymbol{o}_{\boldsymbol{a}_i}$ se expresa, ob-

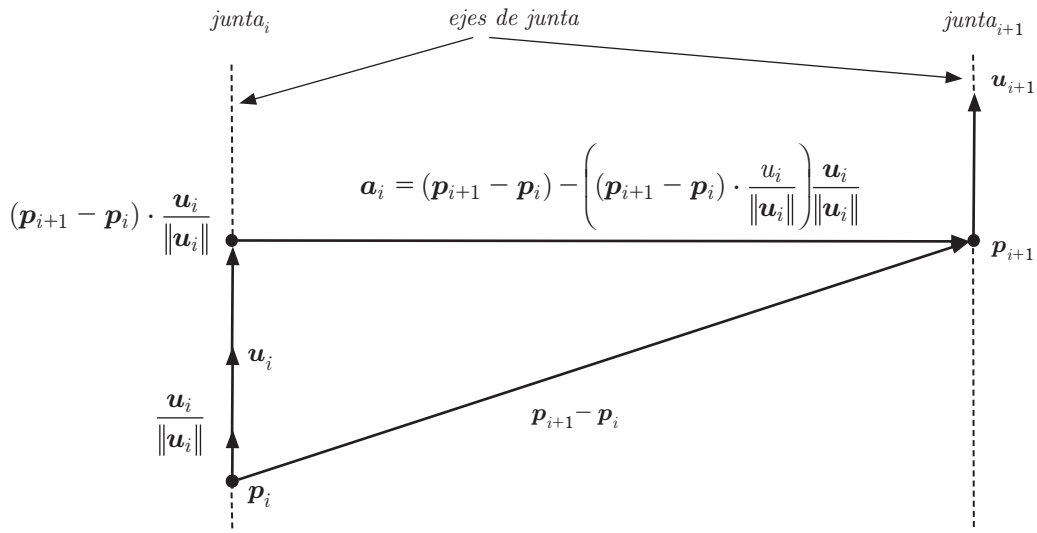


Figura 2.11 Los ejes de la junta son paralelos y entonces no hay distancia más corta única entre ellos.

viamente, por $\boldsymbol{o}_i$ y $\boldsymbol{a}_i$. Ya que el origen $\boldsymbol{o}_i$ se selecciona en forma arbitraria, sería mejor elegirlo de tal manera que la mayor parte de los parámetros de Denavit-Hartenberg sean iguales a cero.

2.3.1.7 Caso especial: la primera junta

Una figura articulada debe comenzar en algún lugar, por lo que el problema consiste en saber cuál es la primera junta, porque no hay un eslabón antes de ella. Entonces se introduce un *eslabón base*, representado por el $eslabón_0$. El marco de eslabón para $eslabón_0$ se puede seleccionar de forma arbitraria, pero con astucia se hace coincidir con el marco de eslabón del $eslabón_1$ cuando la figura articulada está en posición de reposo. Entonces, la mayor parte de los parámetros de Denavit-Hartenberg serán iguales a cero.

2.3.1.8 Caso especial: la última junta

Una figura articulada debe terminar en algún lugar, por lo que la última junta constituye un problema, porque no hay eslabón que la siga. En general, se puede seleccionar el marco de coordenadas del último eslabón en forma arbitraria, porque no hay eslabón físico; es decir, la figura articulada termina en el eje de la $junta_N$ y no hay vector de eslabón $\boldsymbol{a}_N$. Lo único con que se cuenta es el eje de la $junta_N$, que viene a ser el eje z_N. Pero lo mejor es seleccionar el origen $\boldsymbol{o}_N$, el vector eslabón $\boldsymbol{a}_N$ (el eje $\boldsymbol{x}_N$), el desplazamiento de eslabón d_N y el torcimiento de eslabón α_N de tal modo que la mayor parte de los parámetros de Denavit-Hartenberg sean iguales a cero.

2.3.2 Fijación del marco de coordenadas

Dada una figura articulada, lo primero que se debe hacer es fijar un sistema coordenado a cada eslabón. A esos sistemas coordenados se les llama *marcos de eslabón*. El procedimiento para fijar estos marcos se indica a continuación

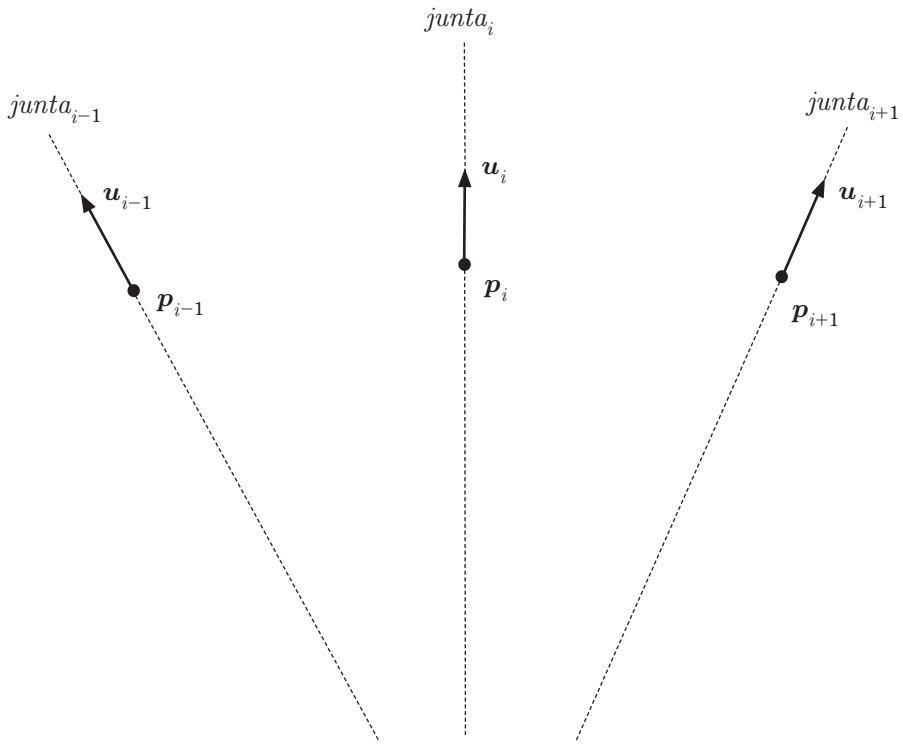


Figura 2.12 Ejes de junta de una figura articulada. Los ejes se definen por las ecuaciones paramétricas $\boldsymbol{l}_i(s) = \boldsymbol{p}_i + s\boldsymbol{u}_i$. Imagine que los ejes de junta son los vectores $\boldsymbol{u}_i$.

1. Identificar los ejes de junta.
2. Identificar las perpendiculares comunes a ejes de junta sucesivos.
3. Fijar marcos de coordenadas a cada eje de junta.

En seguida explicaremos estos pasos.

Primero se identifican los ejes de la junta de la figura articulada (véase la figura 2.12). El eje de la $junta_i$ se define por la ecuación (2.10a), y por comodidad la repetimos a continuación:

$$\boldsymbol{l}_i(s) = \boldsymbol{p}_i + s\boldsymbol{u}_i \tag{2.41}$$

Después se identifica la perpendicular común a ejes de junta vecinos; esto es, la perpendicular común $\boldsymbol{a}_i$ a los ejes de la $junta_i$ y la $junta_{i+1}$. También se identifica el punto $\boldsymbol{o}_i$ donde la perpendicular común corta el eje de la $junta_i$. Eso se ilustra en la figura 2.13.

En la sección 2.3.1.3 se demostró cómo calcular la distancia a_i más corta entre los ejes de la $junta_i$ y la $junta_{i+1}$. También se mostró que la distancia más corta es a lo largo de la perpendicular entre los ejes de la $junta_i$ y la $junta_{i+1}$. Por último, se analizó cómo calcular el lugar de la distancia más corta $\boldsymbol{o}_i$ en el eje

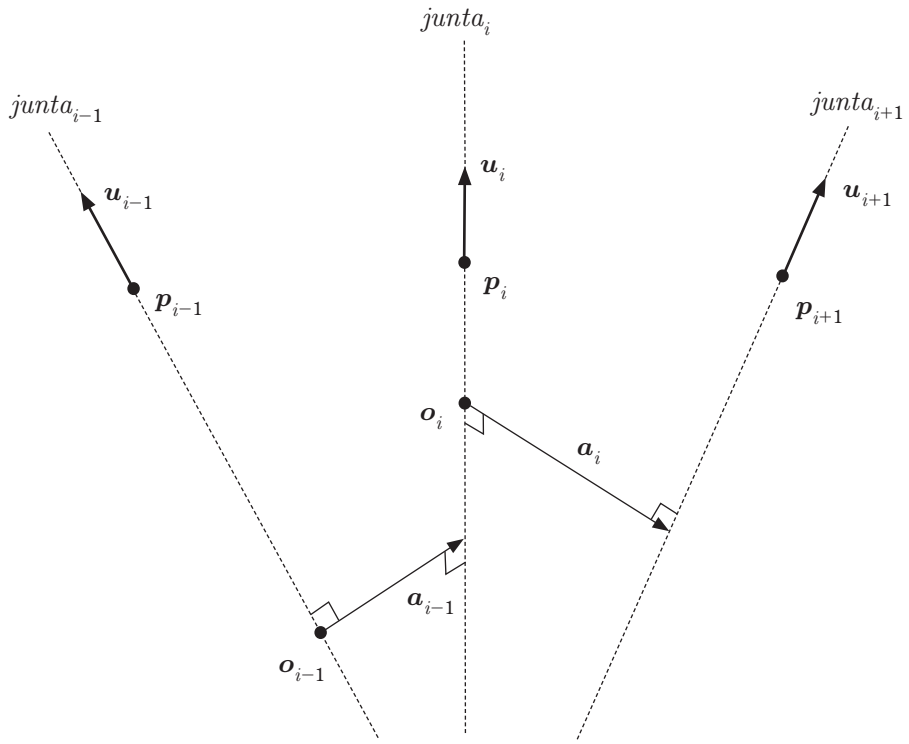


Figura 2.13 Los vectores eslabón $\boldsymbol{a}_i$ y los orígenes $\boldsymbol{o}_i$.

de la $junta_i$. Lo que aquí se necesita es el vector eslabón $\boldsymbol{a}_i$ y su intersección $\boldsymbol{o}_i$ con el eje de la $junta_i$ [véanse (2.31a) y (2.32a) en la sección 2.3.1.3].

Por último, el $i^{ésimo}$ marco de eslabón, que se muestra en la figura 2.14, se puede construir como sigue:

1. **El origen.** Sea el origen del $i^{ésimo}$ marco del eslabón en el punto $\boldsymbol{o}_i$ en el eje de la $junta_i$.
2. **El eje $\boldsymbol{z}_i$.** Sea el eje $\boldsymbol{z}_i$ a lo largo del $i^{ésimo}$ eje de la junta. Esto es, sea $\boldsymbol{z}_i$ paralelo al vector $\boldsymbol{u}_i$, de acuerdo con (2.41)

$$\boldsymbol{z}_i = \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2} \tag{2.42}$$

3. **El eje $\boldsymbol{x}_i$.** Sea el eje $\boldsymbol{x}_i$ a lo largo del vector eslabón $\boldsymbol{a}_i$ de (2.32a)

$$\boldsymbol{x}_i = \frac{\boldsymbol{a}_i}{\|\boldsymbol{a}_i\|_2} \tag{2.43}$$

4. **El eje $\boldsymbol{y}_i$.** Sea el eje $\boldsymbol{y}_i$ tal que los vectores $\boldsymbol{x}_i$, $\boldsymbol{y}_i$ y $\boldsymbol{z}_i$ formen un sistema coordenado ortogonal derecho. Esto es, sea $\boldsymbol{y}_i$ definido por

$$\boldsymbol{y}_i = \frac{\boldsymbol{z}_i \times \boldsymbol{x}_i}{\|\boldsymbol{z}_i \times \boldsymbol{x}_i\|_2} \tag{2.44}$$

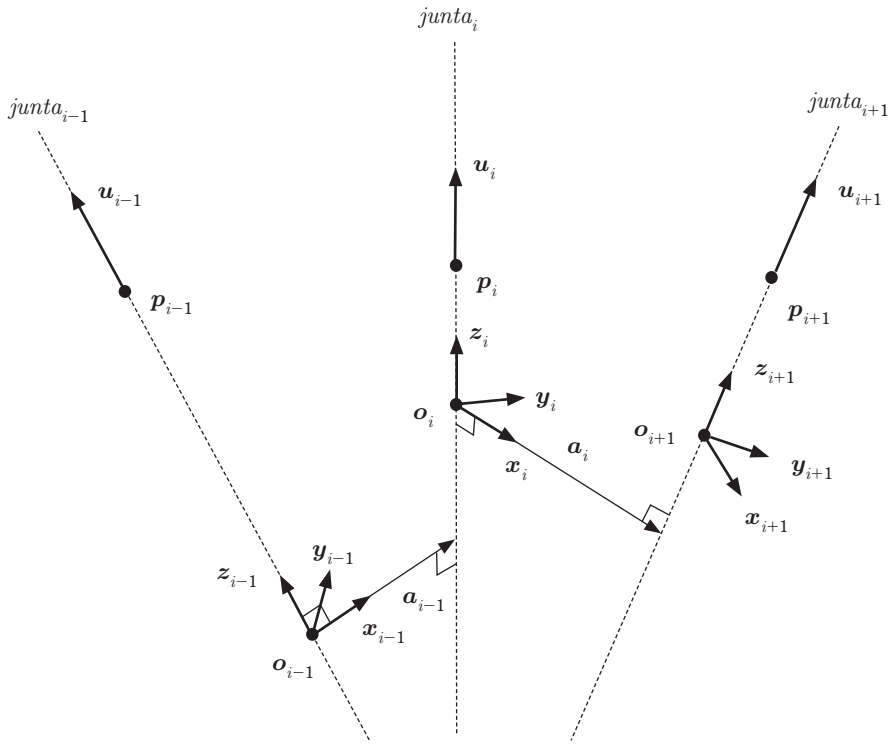


Figura 2.14 La figura articulada y sus marcos de eslabón $\boldsymbol{o}_i$, $\boldsymbol{x}_i$, $\boldsymbol{y}_i$ y $\boldsymbol{z}_i$.

2.3.3 El torcimiento α_i de eslabón

El torcimiento de eslabón es el ángulo α_i entre los ejes de la *junta*$_i$ y la *junta*$_{i+1}$. Ese ángulo α_i se mide en torno al eje $\boldsymbol{x}_i$. Los ángulos positivos se miden en sentido contrario al de las manecillas del reloj, viendo desde la punta del vector $\boldsymbol{x}_i$ hacia su pie (véase la figura 2.15). En forma específica, el torcimiento α_i es el ángulo entre los vectores de dirección de los ejes de junta $\boldsymbol{u}_i$ y $\boldsymbol{u}_{i+1}$, medido en torno al vector eslabón $\boldsymbol{a}_i$.

Recordemos algunas propiedades de los productos escalar y vectorial.

$$\boldsymbol{u}_i \cdot \boldsymbol{u}_{i+1} = \|\boldsymbol{u}_i\|_2 \, \|\boldsymbol{u}_{i+1}\|_2 \cos \alpha_i \qquad 0 \leq \alpha_i \leq \pi \tag{2.45a}$$

$$\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2 = \|\boldsymbol{u}_i\|_2 \, \|\boldsymbol{u}_{i+1}\|_2 \operatorname{sen} \alpha_i \qquad 0 \leq \alpha_i \leq \pi \tag{2.45b}$$

que equivalen a

$$\cos \alpha_i = \frac{\boldsymbol{u}_i \cdot \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i\|_2 \, \|\boldsymbol{u}_{i+1}\|_2} \qquad 0 \leq \alpha_i \leq \pi \tag{2.46a}$$

$$\operatorname{sen} \alpha_i = \frac{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2}{\|\boldsymbol{u}_i\|_2 \, \|\boldsymbol{u}_{i+1}\|_2} \qquad 0 \leq \alpha_i \leq \pi \tag{2.46b}$$

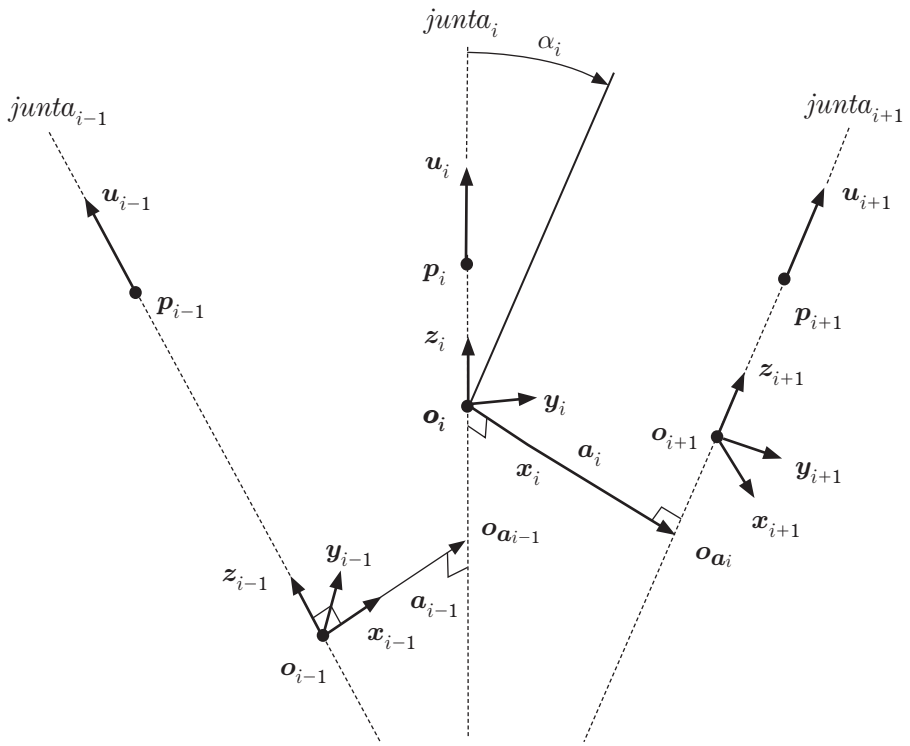


Figura 2.15 Marcos de eslabón y torcimiento α_i de eslabón. En la figura, el torcimiento de eslabón α_i es negativo.

Obsérvese que como $0 \leq \text{sen}\ \alpha_i \leq 1$, y $-1 \leq \cos\ \alpha_i \leq 1$, el ángulo α_i siempre estará en el intervalo $0 \leq \alpha_i \leq \pi$ (véase la figura 2.16). A continuación mostraremos cómo calcular en forma correcta el torcimiento de eslabón α_i.

Recordaremos que la función matemática $\arctan : \mathbb{R} \circledR \mathbb{R}$ regresa un valor en el intervalo $(-\pi, \pi)$. Sea la función $\arctan 2 : \mathbb{R}^2 \circledR \mathbb{R}$ definida por

$$\arctan 2(n,d) = \begin{cases} \arctan\left(\dfrac{n}{d}\right) & \text{si} \quad n > 0 \ \wedge \ d > 0 \\ \arctan\left(\dfrac{n}{d}\right) & \text{si} \quad n < 0 \ \wedge \ d > 0 \\ \arctan\left(\dfrac{n}{d}\right) + \boldsymbol{\pi} & \text{si} \quad n > 0 \ \wedge \ d < 0 \\ \arctan\left(\dfrac{n}{d}\right) - \boldsymbol{\pi} & \text{si} \quad n < 0 \ \wedge \ d < 0 \end{cases} \tag{2.47}$$

Esto es, si se calculara de forma ingenua el torcimiento de eslabón α_i como

$$\alpha_i = \arctan 2\left(\frac{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}, \frac{\boldsymbol{u}_i \cdot \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}\right) \tag{2.48}$$

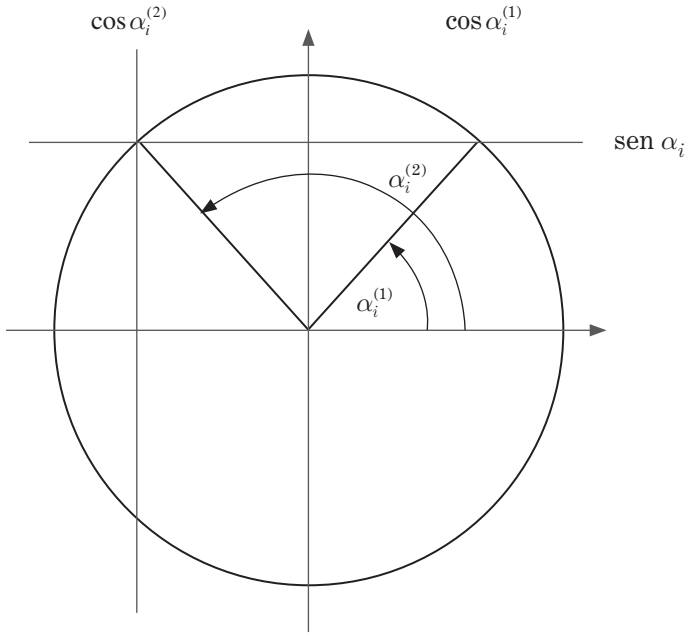


Figura 2.16 El ángulo de torcimiento α_i siempre es positivo y está en el intervalo $0 \leq \alpha_i \leq \pi$. Ya que $0 \leq \text{sen } \alpha_i \leq 1$ y $-1 \leq \cos \alpha_i \leq 1$, hay dos ángulos posibles, $\alpha_i^{(1)}$ y $\alpha_i^{(2)}$, dependiendo del signo del cos α_i.

el resultado sería incorrecto algunas veces, porque de acuerdo con (2.46b), sen α_i siempre es positivo. Eso quiere decir que el ángulo α_i siempre será uno de los dos ángulos positivos, $\alpha_i^{(1)}$ o $\alpha_i^{(2)}$, como se observa en la figura 2.16; es decir, α_i siempre será positivo.

En el producto vectorial (2.46b), el ángulo α_i se mide en torno al eje $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$. Si el vector $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$ tiene la misma dirección que el vector eslabón $\boldsymbol{a}_i$, el ángulo α_i en (2.48) es correcto (véase la figura 2.17). Si el vector $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$ tiene dirección contraria a la del vector eslabón $\boldsymbol{a}_i$, entonces el ángulo α_i en (2.48) no es correcto. El ángulo α_i correcto será el negativo del ángulo en (2.48); véase la figura 2.18.

Para diferenciar entre los dos casos en que los vectores $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$ y $\boldsymbol{a}_i$ tienen direcciones iguales o contrarias, basta ver el signo de su producto punto $(\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}) \cdot \boldsymbol{a}_i$. Esto se expresa en la siguiente ecuación.

$$\alpha_i = \begin{cases} +\arctan2\left(\dfrac{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}, \dfrac{\boldsymbol{u}_i \,.\, \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}\right) & \text{si } (\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}) \,.\, \boldsymbol{a}_i \geq 0 \\ -\arctan2\left(\dfrac{\|\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}\|_2}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}, \dfrac{\boldsymbol{u}_i \,.\, \boldsymbol{u}_{i+1}}{\|\boldsymbol{u}_i\|_2 \|\boldsymbol{u}_{i+1}\|_2}\right) & \text{si } (\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}) \,.\, \boldsymbol{a}_i < 0 \end{cases} \tag{2.49}$$

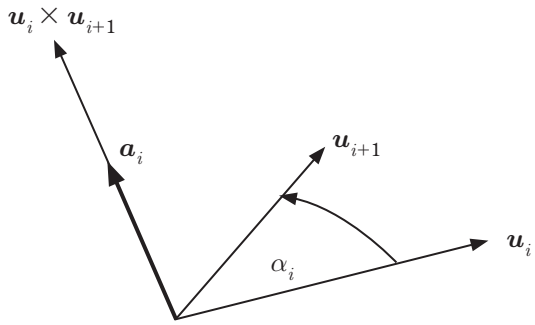


Figura 2.17 El ángulo de torcimiento es positivo porque la dirección del vector $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$ tiene la misma dirección que el vector eslabón $\boldsymbol{a}_i$.

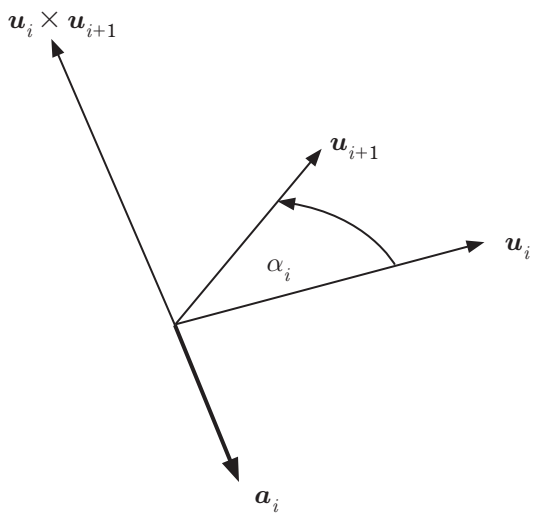


Figura 2.18 El ángulo de torcimiento α_i es negativo porque la dirección del vector $\boldsymbol{u}_i \times \boldsymbol{u}_{i+1}$ tiene la dirección contraria a la del vector eslabón $\boldsymbol{a}_i$.

Temas cubiertos

- **Cinemática**
  Figuras articuladas, cinemática directa e inversa, e interpolación de movimiento
- **Animación multicuerpo**
  Penalización, impulso y animación multicuerpo basada en estrés
- **Objetos deformables**
  Sistemas de partículas, modelos continuos con diferencias finitas, método de elemento finito y dinámica de fluidos por computadora
- **Detección de colisiones**
  Detección de colisiones de fase amplia y estrecha, determinación de contacto, jerarquía de volumen limitante y algoritmos basados en volumen

Física para videojuegos

Erleben, Sporring, Henriksen, Dohlmann

El *boom* de los videojuegos y la industria de las películas animadas continúa llevando a la comunidad gráfica a la insaciable búsqueda de mayor realismo, credibilidad y velocidad de animación. En la actualidad, para alcanzar la calidad esperada por la audiencia de videojuegos y películas, los programadores necesitan comprender e implementar la animación basada en la física. Con el fin de facilitar este entendimiento, la presente obra está escrita para mostrar a los estudiantes y profesionistas la teoría detrás de los modelos matemáticos y las técnicas requeridas para la animación. No tiene el objetivo de enseñar los principios básicos de la animación, sino cómo transformar la teoría en habilidades prácticas. Detalla cómo los modelos matemáticos se derivan de principios físicos y matemáticos, y explica cómo estos modelos son resueltos por medio de computadora de forma eficiente y estable.

Este excelente libro abarca todos los temas relacionados con la animación basada en física, incluyendo la detección de colisiones, geometría, mecánica, ecuaciones diferenciales, matrices, cuaternios y más. Ofrece una gran cobertura de los algoritmos de detección de colisiones y un excelente repaso de los principales sistemas físicos. Además, contiene numerosos ejemplos y muestra una gran cantidad de pseudocódigos para la mayoría de los algoritmos.

Esta obra es ideal para estudiantes de diseño gráfico, animación digital, investigadores de campo y profesionistas que trabajan en la industria fílmica y de los videojuegos.

Novedades

- Gran cobertura de los algoritmos de detección de colisiones
- Excelente repaso de las partes de los sistemas físicos
- Numerosos ejemplos a detalle de los pseudocódigos para la mayoría de los algoritmos
- Un sitio web donde se incluyen pseudocódigos del libro y ejercicios por capítulo

Biografía de los autores

Kenny Erleben obtuvo su maestría en el Departamento de Ciencias Computacionales de la Universidad de Copenhague y actualmente labora como profesor asistente. **Jon Sporring** tiene una maestría también del Departamento de Ciencias Computacionales de la Universidad de Copenhague, donde trabaja como profesor de cátedra. **Knud Henriksen** cuenta con maestría en ciencias computacionales y es profesor asociado en la misma universidad. **Henrik Dohlmann** obtuvo su maestría en la misma institución y actualmente colabora en un proyecto entre el Departamento de Ciencias Computacionales y la Escuela de Odontología.

Imagen de portada: Kenny Erleben. Diseño de portada: Tyler Creative

ISBN-13: 978-607-481-506-1
ISBN-10: 607-481-506-2

CENGAGE Learning™

http://latinoamerica.cengage.com