UNIVERSIDAD PÚBLICA DE NAVARRA
NAFARROAKO UNIBERTSITATE PUBLIKOA

Departamento de Ingeniería Mecánica, Energética y de Materiales

upna
Universidad
Pública de Navarra
Nafarroako
Unibertsitate Publikoa

TESIS DOCTORAL

Modelado simbólico para la simulación en tiempo real de sistemas multicuerpo.

Realizada por: Aitor Plaza Puértolas
Dirigida por: Dr. Javier Ros Ganuza

Pamplona/Iruñea Diciembre 2015

Tesi hau neskei
eskaini nahi diet:
Olhain eta Xantal

AGRADECIMIENTOS

Tras el esfuerzo que me ha supuesto realizar esta tesis doctoral quiero aprovechar este espacio para dar las gracias a quienes me han apoyado, ayudado, motivado y acompañado en este camino. Sobre todo, a los que han tenido paciencia conmigo.

En primer lugar, quisiera dar las gracias a mi director de tesis, el Dr. Javier Ros. Gracias a él estoy aquí y esta tesis está terminada. Él ha sabido animarme y guiarme, transmitirme su pasión por la ciencia y por el el mundo *Multibody*. Ha sido capaz de tenerme paciencia, confiar en mí y haberme escuchado cuando hacía falta. *Milesker Javier!!*.

Por otro lado, quiero agradecer sinceramente a mis compañeros y amigos Dr. Xabier Iriarte y Dr. Jokin Aginaga. Ellos han sido capaces de motivarme y apoyarme, de tal forma que este camino no haya sido tan largo. *Milesker!!*. De igual manera, quisiera agradecer al resto de los miembros de grupo de investigación IMAC, al cual tengo el honor de pertenecer.

Quisiera también dar las gracias a mi madre y a mi hermano. Ellos también han sabido apoyarme, aunque muchas veces no se lo haya dicho, estoy sinceramente agradecido. *Eskerrik asko!!*

A mi amigos, a los que me preguntaban: *¿Cómo va esa tesis?*, a los que me preguntaban sin enterarse de nada: *Ah, ¿estás haciendo una tesis?*, e incluso a los que preguntaban: *Pero... ¿vas a hacer del mundo un lugar mejor?*, gracias!!, *eskerrik asko!!*.

Beste aldetik, esker beroenak Maya familiari, familiako beste bat banintz bezala onartzeagatik

Azkenik, etxeko neskei nire esker beroena. Haiek jakin izan dute pazientzia izaten nirekin, batez ez zu Xantal. Olhain, zu behar bada ez zara konturatu, baina jakin ezazu nire bizitza aldatu didazula eta ezagutzen zaitudanetik egunek beste kolore bat dutela. Aprobetxatu nahi dut tarte hau zuri, Olhain, bizitza zoriontsu eta oparo bat desiratzeko. Bihotzez, milesker neskak!!!

ÍNDICE GENERAL

1. Introducción **1**

1.1. Modelado simbólico 3

1.2. Estado del arte. Códigos para el modelado simbólico 4

1.2.1. Códigos genéricos de modelado simbólico 4

1.2.2. Códigos simbólicos para el modelado multicuerpo . . . 4

1.2.3. Cinemática en sistemas multicuerpo 8

1.2.4. Dinámica en sistemas multicuerpo 8

1.2.5. Simulación en tiempo real 9

1.3. Objeto de esta Tesis Doctoral 10

1.4. Estructura de la Tesis Doctoral 12

2. Introducción al modelado de sistemas multicuerpo **15**

2.1. Descripción de sistema multicuerpo 15

2.2. Modelado cinemático 16

2.2.1. Coordenadas generalizadas 16

2.2.2. Concepto de Observador o Referencia, y de Velocidad y Aceleración de un punto 17

2.2.3. Composición de movimientos 19

2.2.4. Ecuaciones cinemáticas 20
2.3. Modelado Dinámico 23
2.3.1. Planteamiento de las ecuaciones dinámicas 24
2.4. Estructura del sistema multicuerpo 28
2.5. Simulación de sistemas multicuerpo 30
2.5.1. Problema de ensamblaje 30
2.5.2. Problema de posición, velocidad y aceleración inicial . . 32
2.5.3. Simulación cinemática 34
2.5.4. Corrección de la posición y la velocidad 37
2.6. Simulación dinámica directa 39
2.7. Cálculo de las matrices y vectores del problema DSM y exportación de los mismos 40
2.7.1. Matrices del problema cinemático 41
2.7.2. Matrices del problema dinámico 43
2.7.3. Exportación de matrices para la resolución del problema DSM 44
2.8. Coordenadas sin inercia asociada 44
2.8.1. Ejemplo de sistema con coordenadas sin inercia asociada 45

3. Atomización. Algoritmos simbólicos. Exportación 47
3.1. Proceso de atomización. 48
3.1.1. Atomización “sobre la marcha” 49
3.1.2. Atomización completa de expresiones 51
3.1.3. Atomización en Robotran. Problemática 51
3.2. Implementación de la atomización 52
3.2.1. Estado original de las funciones de atomización y desatomización 53
3.2.2. Atomización con búsqueda basada en tablas “hash” . . . 55
3.3. Estructura de árbol de una expresión 58
3.3.1. Búsqueda en profundidad con pre-orden 59
3.4. Atomización y desatomización. Algoritmos recursivos 63
3.4.1. Algoritmo de atomización 63
3.4.2. Algoritmo de desatomización 67

3.5. Algoritmo reursivo de sustitución de un símbolo 68
3.5.1. Algoritmo reursivo de sustitución de un conjunto de símbolos 70
3.6. Algoritmo reursivo de derivación 73
3.7. Exportación simbólica basada en la atomización 76
3.7.1. Exportación de una expresión 77
3.7.2. Búsqueda de átomos de una expresión atomizada 77
3.7.3. Exportación de matrices y vectores 79
3.7.4. Optimización en la exportación 82
3.7.5. Exportación de matrices conjunta 88

4. Modelado simbólico de sistemas multicuerpo **93**
4.1. Caracterización cinemática de sistemas multicuerpo 94
4.1.1. Estructura de bases 95
4.1.2. Vector y tensor cartesiano. Representación y álgebra . . . 97
4.1.3. Estructura de puntos 99
4.1.4. Caracterización cinemática de cada sólido. Referencia . . 100
4.2. Modificador “gravedad”: Atomización y Parentesización 102
4.2.1. Ejemplo: Vector de posición entre dos puntos 103
4.3. Operadores cinemáticos 105
4.3.1. Operador: Matriz de rotación 106
4.3.2. Operador: Vector de posición 109
4.3.3. Operador: Vector de velocidad angular 113
4.3.4. Operador: Vector de velocidad de un punto respecto a una referencia 115
4.4. Tensores no cartesianos 119
4.4.1. Matrices jacobianas como aplicaciones lineales 120
4.4.2. Tensores no cartesianos. Cambios de base 120
4.5. Cálculo de la matriz de masa **M** 123
4.5.1. PPV: Torsores 123
4.5.2. Torsor de inercia de D’Alembert 123
4.5.3. Optimizaciones para el cálculo de la matriz **M** 127

4.5.4. Optimización de **M** basada en coordenadas sin inercia asociada . . . 134
4.6. Cálculo del vector δ . . . 135
4.6.1. Vector δ . . . 135
4.6.2. Optimizaciones en el cálculo . . . 136
4.7. Representación matricial . . . 139
4.8. Términos comunes entre **M** y δ, bucle de cálculo . . . 141

5. Expresiones trigonométricamente simplificables 143
5.1. Expresión trigonométricamente simplificables con origen "vectorial" . . . 144
5.1.1. Ejemplo: producto escalar de dos vectores . . . 144
5.2. Nueva estructura simbólica de vector: Vector extendido . . . 146
5.2.1. Operaciones entre vectores extendidos . . . 147
5.2.2. Ejemplo de vector extendido . . . 150
5.3. Simplificaciones de origen "tensorial" . . . 151
5.4. Nueva estructura simbólica de tensor: Tensor extendido . . . 152
5.4.1. Operaciones entre tensores extendidos . . . 153
5.4.2. Ejemplo de trabajo con tensores extendidos . . . 155
5.5. Matriz jacobiana de un vector extendido . . . 156
5.6. Orden de los elementos dentro de las nuevas estructuras . . . 156
5.7. Optimización del vector/tensor extendido . . . 158
5.8. Simplificaciones trigonométricas en el campo de la DSM . . . 160
5.8.1. Matrices jacobianas de velocidad lineal . . . 161
5.8.2. Matrices jacobianas de velocidad angular . . . 162
5.8.3. Matrices anti-simétricas . . . 163
5.8.4. Productos que implican el tensor de inercia . . . 164
5.8.5. Productos que implican el vector centro de masa . . . 165
5.8.6. Producto vectorial entre la velocidad angular y vector de posición . . . 167
5.8.7. Simplificaciones trigonométricas en otras formulaciones 168
5.9. Simplificaciones trigonométricas con origen en la suma de ángulos . . . 169

5.10. ESST: Mecanismo de cuatro barras 170
5.10.1. Cálculo de la matriz de masa 174

6. Resultados y análisis **181**
6.1. Algoritmos genéricos: atomización, sustitución y derivación . . 181
6.2. Atomización "sobre la marcha".Generación de código 184
6.2.1. Exportación conjunta 186
6.3. Evaluación numérica 188
6.4. Cálculos cinemáticos. Parentesización 189
6.5. Modificador "Gravedad". Reciclabilidad de átomos 190
6.6. Matriz $\mathbf{M}$ y vector δ 191
6.6.1. Coordenadas sin inercia asociada. $\mathbf{M}$ y δ. 194
6.7. Expresiones susceptibles de ser simplificables trigonométricamente 195

7. Conclusiones **197**
7.1. Parentesización y reciclabilidad de átomos 197
7.1.1. Atomización 198
7.1.2. Operadores cinemáticos 198
7.1.3. Gravedad 199
7.2. Algoritmos de sustitución y derivación recursivos 199
7.3. Generación de código 200
7.3.1. Exportación conjunta 201
7.4. Cálculo de $\mathbf{M}$ y δ 201
7.5. Expresiones susceptibles de simplificación 202
7.6. Otros 203

8. Líneas futuras **205**
8.1. Algoritmos de sustitución y diferenciación 205
8.2. Cálculo de la matriz de observación 206
8.3. Transferencias de inercia 207
8.4. Vectores de aceleración 207
8.5. Modelado de sistemas flexibles 208
8.6. Exportación conjunta de todo el sistema 208

8.7. Estructura de la matriz de masa. Descomposición LDL 209

A. Ejemplos **211**
A.1. Bloque-peńdulo 211
A.2. Cuadrilátero articulado 213
A.3. Biela, manivela, disco 215
A.4. Manipulador paralelo "Stewart" 218
A.5. Locomotora FEVE 3800 219
A.5.1. Ecuaciones geométricas 224
A.5.2. Coordenadas totales del modelo 226

B. Código simbólicos generado. Mecanismo de cuatro barras **229**
B.1. Componentes del vector $\mathbf{r}_O^C$ calculado según distintos métodos . 229
B.2. Matriz $\mathbf{M}$ y vector $\boldsymbol{\delta}$ juntos 230
B.3. Vector $\boldsymbol{\Phi}$ y matriz $\boldsymbol{\Phi}_q$ juntos 236

Bibliografía **239**

Índice de figuras **244**

Índice de tablas **246**

NOTACIÓN **247**

CAPÍTULO 1

INTRODUCCIÓN

En el contexto de la ingeniería es normal el uso de modelos matemáticos de sistemas para la realización de todo tipo de experimentos y desarrollos teóricos. Concretamente, en el contexto de la ingeniería mecánica (teoría de máquinas y mecanismos, robótica, teoría de control, realidad virtual, bio-mecánica, etc..) los modelos multicuerpo son uno de los pilares fundamentales.

El modelo ha de representar fielmente el sistema y ha se ser construido atendiendo a las reglas de la física conocida. Así pues, se entiende por modelo a un conjunto ecuaciones que representan las relaciones causa-efecto entre las entradas y salidas del sistema físico a estudio. A esta fase de generación de una "buena representación" del sistema real (modelo) se le conoce como "modelado".

Se define la fase de "análisis" al estudio desde el punto de vista numérico del comportamiento del modelo basado en las ecuaciones que lo describen.

Estos modelos son utilizados para la toma de decisiones (optimización, diseño, control,...) y para el análisis numérico (simulación en el tiempo, posición de equilibrio, análisis modal, etc ...).

Cuando el modelo tiene un tamaño relativamente "pequeño", las ecuaciones cinemáticas y dinámicas que lo representan pueden, en general, ser construidas "a mano". Aunque esté sujeto a errores humanos, este método tiene la ventaja que se generan ecuaciones optimizadas en términos de expresiones trigonométricas y aritméticas. La fase de análisis consiste en sustituir en esas ecuaciones los símbolos por su valor numérico.

La forma "clásica" de modelado y análisis de sistemas multicuerpo basada en métodos computacionales se denomina "generación numérica". En cada instante temporal del análisis el modelo ha de ser reconstruido en función del estado de sistema y de sus parámetros. Es decir, en cierta media, las fases de modelado y análisis se han de realizar a la vez y para cada estado del sistema. En [1] se describen los métodos y algoritmos más usados en el análisis dinámico y cinemático desde el punto de vista de la generación numérica.

Frecuentemente, muchos de los parámetros del sistema son cero y en la generación numérica son tratados de igual forma que los no nulos. Esto es debido a que siempre se utiliza un modelo general. Así que se presenta la imposibilidad de aprovechar dicho hecho para la disminución del costo computacional. Nótese que en caso de realizar las operaciones "a mano", se evita la aparición de estas cantidades nulas mediante su directa eliminación.

La denominada "generación simbólica" toma las ventajas de los dos métodos anteriores y se implementa usualmente de forma computacional. En el "modelado simbólico", se utilizan únicamente operadores aritméticos y símbolos para construir un conjunto de ecuaciones que representen el modelo. En la fase de análisis, en un proceso que se denomina "evaluación numérica", estas cadenas de texto (símbolos) son sustituidas un valor numérico. A continuación se realizan, con datos puramente numéricos, los análisis pertinentes. Para realizar la fase de análisis, las ecuaciones que describen los modelos se expresan en forma de código. Este procedimiento se denomina "generación de código". Por medio del código generado se realiza la evaluación numérica de las distintas funciones de forma óptima.

La "generación simbólica" presenta, por un lado, la ventaja de los métodos manuales, es decir, las ecuaciones dinámicas y cinemáticas solo se han de calcular una vez. Por otro lado, presenta la ventaja de que la expresiones que sean nulas, directamente desaparecen de la ecuaciones y por lo tanto no han de ser tratadas en el análisis. Por último, tienen la ventaja de que se puede trabajar

con sistemas de gran tamaño, ya que es el ordenador el que realiza los cálculos.

Esta tesis se centra principalmente en el estudio del modelado simbólico enfocado a sistemas de tipo multicuerpo con la idea de generar ecuaciones altamente optimizadas.

1.1. Modelado simbólico

Es sobradamente conocido [2, 3, 4, 5] (entre otros) que el modelado simbólico de sistemas multicuerpo produce código eficiente para la evaluación numérica de las funciones que típicamente aparecen en las formaciones multicuerpo (matrices jacobianas, matriz de masa, vectores de fuerzas, etc..).

Generalmente, la eficiencia del código simbólico justifica su uso frente al equivalente numérico. Esto es particularmente cierto cuando se usan métodos que eviten la re-evaluación de sub-expresiones que aparecen repetidamente en las funciones a evaluar. Este proceso de buscar en una función (o expresión simbólica) sub-expresiones que se repitan y sustituirlas por "variables auxiliares" de forma que se evalúen un única vez [2] se denomina, en esta tesis, "atomización" [3].

A pesar de las ventajas anteriormente citadas, la generación simbólica de ecuaciones en sistemas multicuerpo puede convertirse en una ardua tarea con una alta complejidad. Si la generación simbólica no es tratada adecuadamente puede llegar a limitar, e incluso a no hacer posible, el trabajo con sistemas multicuerpo medianamente complejos. Esta es una de las limitaciones más importantes del modelado simbólico.

Por un lado, estas limitaciones tienen origen en el conocido problema (Sección 5.7.3 de [2]) de la "explosión simbólica". Es decir, el tamaño de las expresiones simbólicas crece de forma exponencial con en tamaño del problema, llegando a un punto en que el trabajo con ellas se hace imposible. Por otro lado, además, está la tarea de la generación de código optimizado para realizar el análisis numérico, es decir, la de la atomización. Para sistemas medianamente complejos la atomización tiene una alta complejidad computacional y que puede convertirse en difícil, sino imposible de llevar a cabo.

1.2. Estado del arte. Códigos para el modelado simbólico

En esta sección se presentan algunos de los paquetes de álgebra simbólica para el modelado simbólico de sistemas multicuerpo que existen en la actualidad. Se describen sus principales características, sobre todo en cuanto a la formulación y la parametrización (tipo de coordenadas) con las que trabajan.

1.2.1. Códigos genéricos de modelado simbólico

La primera aproximación al modelado simbólico de sistema multicuerpo está basada en el uso de motores simbólicos de propósito general como Symbolic Math Toolbox de MATLAB [6], Maple [7], Sympy [8] o Xcas/Giac [9] que ayudan a la implementación de formulaciones genéricas con cualquier parametrización.

Esta forma de proceder es análoga a que se si realizara "a mano", pero con el apoyo de estos motores simbólicos para la realización de los cálculos más tediosos. Estos motores simbólicos son capaces de generar código optimizado para ser usado en la fase de análisis en distintos lenguajes de programación numéricos.

Esta forma de trabajar tiene la limitación de que el motor simbólico trata todos los sistemas por igual, independientemente de la naturaleza del problema, es decir, no se saca provecho de la estructura particular de los sistemas en la dinámica de sistemas multicuerpo.

1.2.2. Códigos simbólicos para el modelado multicuerpo

En la actualidad, en el campo de la Dinámica de Sistemas Multicuerpo (DSM) existen un amplio conjunto de paquetes enfocados a la generación simbólica de sistemas multicuerpo: 3D_Mec [10], Robotran [11], MapleSim [7], NeweulM^2[12], PyDy[13], Dymola [14], Wolfram SystemModeler [15]. Cada uno de ellos afronta los problemas propios del modelado simbólico adoptando distintos compromisos.

1.2.2.1. Paquetes comerciales

MapleSim y Neweul-M^2 son dos paquetes comerciales para el diseño y simulación de sistemas multicuerpo basados, respectivamente, en los motores simbólicos Maple y Symbolic Math Toolbox de MATLAB.

Tienen la ventaja de que el trabajo con ellos es muy amigable para el usuario y no son requeridos grandes conocimientos de mecánica para su uso.

Aun así, tienen la desventaja de que el usuario no sabe qué está haciendo y no tiene control sobre la formulación y parametrización que utilizan, quedando limitada a la elección hecha por el paquete. Neweul-M^2 está basado en la formulación de Newton-Euler [16] con coordenadas naturales. En [5] se observa que MapleSim también utiliza la misma formulación con el mismo tipo de coordenadas. Ambos paquetes, delegan en su respectivo motor simbólico la generación y optimización del código a exportar. Además tanto Neweul-M^2 como MapleSim tiene herramientas para realizar el análisis numérico de los sistemas, basadas en MATLAB y en Maple, respectivamente.

Entre otros paquetes comerciales se pueden citar Dymola, basado en el paquete Modelica [17] y Wolfram SystemModeler[18] basado en el código Mathematica. Al igual que los anteriores, permiten realizar las fases de modelado y de análisis en en mismo entorno y ambos son totalmente opacos al usuario final.

1.2.2.2. Robotran

Robotran [11, 19] dispone de un motor simbólico propio, razón por la que se ha considera separadamente de los paquetes anteriores.

Robotran trabaja con una formulación recursiva específica, en coordenadas relativas y con un proceso propio de atomización “sobre la marcha”.

El conjunto de formulación recursiva y atomización “sobre la marcha”, reduce enormemente el esfuerzo computacional requerido. Esta es, probablemente, la mayor ventaja de Robotran. Pero, a su vez, esta característica hace que no sea extensible a otras formulaciones y parametrizaciones.

Robotram permite generar y exportar código para diversos lenguajes (C, MATLAB, Python) en los que realizar el análisis numérico del sistema.

1.2.2.3. Librería PyDy

PyDy es un librería simbólica bajo licencia GPL[1] para Python enfocada al modelado simbólico de sistemas multicuerpo. Esta basada en el motor simbólico Sympy y al igual que los anteriores trabaja con un formulación cerrada basada en las ecuaciones de Kane [20]. La generación y optimización del código a exportar se delega íntegramente en el motor simbólico, si bien, al ser una librería para Python, la fase de análisis puede ser realizada en ese mismo lenguaje de programación.

1.2.2.4. 3D_Mec

3D_Mec es un programa desarrollado por el grupo de investigación al que pertenece el autor de esta tesis y que principalmente está enfocado a la docencia en el campo de DSM.

Dispone de su propio lenguaje de programación y de un motor simbólico propio que permite realizar operaciones con expresiones simbólicas. Su funcionalidad para el desarrollo de las ecuaciones de la DSM es completa, aun así presenta ciertas limitaciones. Principalmente, el motor simbólico tiene limitación en cuanto al tamaño de las expresiones que se pueden generar y no dispone de métodos de optimización de las mismas.

Permite exportar código a los lenguajes C y MATLAB. 3D_Mec también permite realizar las fases de modelado y de análisis (únicamente la simulación numérica) en el mismo entorno.

1.2.2.5. Librería lib_3D_MEC-GiNaC

Como punto de partida de los algoritmos presentados en esta tesis doctoral, se ha de citar la librería simbólica lib_3D_MEC-GiNaC [3].

Es una librería simbólica, para el lenguaje de programación *C++* de código abierto enfocada al campo de la DSM. Está basa en el motor simbólico GiNaC[2].

[1] General Public License: www.gnu.org/licenses/gpl-3.0.en.html

[2] www.ginac.de. Una librería para el álgebra simbólica distribuida bajo licencia GPL y que permite programar algoritmos simbólicos directamente sobre el lenguaje *C++*

lib_3D_MEC-GiNaC ofrece una colección de datos abstractos como vectores, tensores, puntos, bases, matrices,... y una serie de operadores/funciones orientados al campo de la dinámica multicuerpo. Es agnóstica en relación a la parametrización y a la formulación elegidas, que quedan a la elección del usuario.

La librería, en el momento de iniciar esta tesis, sí que disponía de un método propio de atomización muy poco eficaz, que limitaba su uso a problemas de pequeño tamaño, además de generar código no completamente óptimo. Por lo que el proceso de optimización y de exportación de las expresiones se delegaba en el motor simbólico Maple.

1.2.2.6. Códigos simbólicos orientados a la robótica

Dentro del abanico de códigos simbólicos orientados al modelado multicuerpo existe un conjunto de códigos simbólicos específicamente orientados a la robótica.

Así, en [21] se presentan cronológicamente diversos códigos orientados a la robótica computacional simbólica, enfatizando las características de cada unos de ellos.

Más recientemente, en [22] se presenta el código simbólico SYMORO+, enfocado al modelado automatizado y simbólicos de robots. Este paquete permite generar simbólicamente los modelos cinemáticos y dinámicos directo e inverso, así como modelos orientados a la identificación de los parámetros inerciales.

Por último, se ha de citar "The Symbolic Robot Modeling Toolbox", una colección de funciones orientadas a la modelización simbólica de robots. Provee las herramientas necesarias para la obtención de las expresiones simbólicas de la cinemática y dinámica de robots. Perteneciente al paquete Robotics Toolbox de MATLAB [23]. En [24] se realiza una introducción detallada de la cinemática y dinámica de manipuladores de robóticos de tipo serie usando el citado paquete.

1.2.3. Cinemática en sistemas multicuerpo

Frecuentemente, los sistemas multicuerpo se describen utilizando una topología de árbol [2] (Capítulo 3). Esta topología permite identificar la recursividad subyacente en los cálculos cinemáticos que es mayormente utilizada en combinación con una parametrización en coordenadas relativas y que constituye la base de las formulaciones recursivas [25].

En estas formulaciones, la cinemática de cada sólido se define a partir del sólido anterior según la citada estructura de árbol.

En [26] se propone un conjunto de algoritmos simbólicos que sacan provecho de dicha estructura de árbol de sistema y, además, se propone un método para la resolución de las ecuaciones geométricas de forma simbólica.

Los algoritmos recursivos presentados en las anteriores dos referencias tienen dos inconvenientes:

1. Requieren una estructura de árbol y una acotación en coordenadas relativas. Este hecho limita la recursividad al nivel de esta estructura.
2. No ofrecen libertad sobre la base de proyección en que se expresan los resultados. Esto puede conllevar la generación de expresiones simbólicas simplificables trigonométricamente, que aumentan la complejidad computacional del modelo.

1.2.4. Dinámica en sistemas multicuerpo

Como limitaciones que actualmente presenta la DSM simbólica pueden citarse [2]. En la sección 5.7.3 se dice que el Principio de las Potencias Virtuales es un formalismo inapropiado para sistemas de medio y gran tamaño, debido al crecimiento exponencial del número de operaciones simbólicas a realizar. Esto es debido, según el autor, a que se han de obtener las ecuaciones del movimiento "en extenso" y, dadas las características del proceso de atomización que se propone, no se puede aprovechar los beneficios de este proceso.

Es por esto que, la mayor parte de los autores [26, 27, 28], proponen trabajar con formulaciones recursivas de orden $\mathcal{O}(N^3)$ cuando el tamaño del sistema es medianamente grande. En la referencia anterior [2] (Sección 5.7.3) se justifica

el porqué.

1.2.5. Simulación en tiempo real

El concepto de tiempo real es usado ampliamente en diferentes contextos, generalmente técnicos. En general se dice que un proceso es en tiempo real cuando la respuesta del ordenador es tan rápida o más que la velocidad requerida por el usuario o proceso a analizar.

En el contexto de la dinámica multicuerpo, se entiende que un cálculo se realiza en tiempo real cuando el estado del sistema en un instante de tiempo, a partir de otro instante de tiempo dado, se determina en un menor tiempo que la diferencia entre dichos instantes.

En la simulación de sistemas multicuerpo en tiempo real el mundo virtual y el mundo real se unen. Esto puede ser útil para una gran numero de aplicaciones en el contexto de la ingeniería, como son el diseño, control y testeo de sistemas. Por ejemplo, puede ser usada para predecir el comportamiento de un robot modelado en un entorno de control real [29], para localizar el punto de contacto rueda/vía en vehículos ferroviarios [30] o para evaluar el comportamiento del conductor en vehículo por medio de un simulador [31].

En la denominada "generación numérica", la simulación en tiempo real depende de principalmente de cinco factores [32]:

- Cómo se ha modelado el sistema (parametrización).
- Qué principios de la mecánica se ha empleado (Euler-Newton, Potencias Virtuales, ...).
- La formulación utilizada (métodos recursivos, formulación con penalizadores, ...).
- Los algoritmos numéricos para la resolución numérica de los diferentes sistemas.
- El tipo de integrador (implícito, explícito, multi-paso, ...).

En la "generación simbólica", una vez modelado el sistema, la simulación está principalmente condicionada por tres factores:

- Depende del tamaño de las ecuaciones, matrices, vectores simbólicos,

que han de ser evaluados numéricamente.

- Los algoritmos numéricos para la resolución numérica de los diferentes sistemas.
- El tipo de integrador (implícito, explícito, multi-paso, ...).

Se ha destacar que la influencia del integrador y de los algoritmos numéricos es independiente del tipo de generación, sea numérica o simbólica.

En cambio, el tamaño de las expresiones simbólicas a ser evaluadas numéricamente depende directamente de la parametrización, principios mecánicos y formulación utilizada. A diferencia de la "generación numérica", donde el modelo ha de ser generado para cada estado del sistema, en la "generación numérica" el modelo ha de ser generados una vez y evaluado numéricamente para diferentes estados del sistema. El proceso de evaluación numérica de expresiones simbólicas tiene un costo computacional mucho menor que el de generación numérica.

En cada paso de integración, las expresiones simbólicas han de ser evaluadas numéricamente, por lo que la velocidad de evaluación depende directamente también de cómo de optimizadas se hayan generado y exportado. Es en este caso donde se refleja la importancia de la atomización. Es decir, cuanto mejor atomizadas (cuanto más se evite, por medio de variable auxiliares, repetir cálculos) estén las expresiones, más rápido se evaluarán numéricamente y más rápido se podrá realizar la simulación (Tiempo real).

1.3. Objeto de esta Tesis Doctoral

El modelado simbólico para sistemas multicuerpo es una herramienta con gran potencial pero que tiene como mayor limitación un alto coste computacional, ligado a la generación de las ecuaciones, matrices y vectores implicados en el modelo del sistema.

Es por ello que, en esta tesis, se presenta un conjunto de técnicas enfocadas superar las limitaciones del modelado simbólico.

Así pues, los objetivos específicos han sido:

I Realizar un conjunto de métodos y de algoritmos que permitan atomizar

expresiones simbólicas de una manera eficaz y, que una vez implementadas, realizaren esta operación con el menor coste computacional.

El proceso de atomización ha de ser independiente del tipo de formulación y parametrización utilizadas.

II Crear un conjunto de algoritmos simbólicos genéricos. Es decir:

- Crear un algoritmo que realice la operación de sustitución de símbolos en expresiones simbólicas atomizadas, sin que la desatomización sea requerida, y que produzca un resultado así mismo atomizado.
- Crear un algoritmo que realice la derivación de expresiones atomizadas respecto a símbolos obteniendo un resultado también atomizado, sin la necesidad de desatomizar.

III Proponer un conjunto de algoritmos que saquen partido de la atomización, de forma que el código que a ser exportado sea óptimo, para que su evaluación numérica sea lo más eficiente posible.

Demostrar la relación entre la atomización y la eficacia numérica.

IV Proponer un conjunto de métodos y de algoritmos que, sacando partido de la atomización, calculen magnitudes cinemáticas (vector de posición, vector velocidad angular y vector velocidad de un punto respecto a una referencia) con el fin de obtener expresiones simbólicas óptimamente parentesizadas y atomizadas.

Estos algoritmos ha de permitir obtener las componentes de las magnitudes cinemáticas representadas en la base que se considere "óptima".

V Proponer un método para el cálculo de la matriz de masa y del vector de fuerzas generalizadas de un sistema por medio del Principio de las Potencias Virtuales de manera que genere estos elementos atomizados y parentesizados óptimamente.

Demostrar que el citado principio es válido para problemas de tamaño mediano y grande produciendo resultados simbólicos tan buenos como con otros formalismos propuestos en la literatura.

VI Realizar un análisis sobre el origen de las expresiones susceptibles de ser simplificadas trigonométricamente y proponer un método para evitar que aparezcan, evitando así tener que realizar dicha simplificación.

1.4. Estructura de la Tesis Doctoral

La tesis está dividida en un total de siete capítulos, además de este capítulo de introducción.

En el capítulo 2 se realiza una introducción a la DSM y se describen, sin entrar en profundidad, los métodos utilizados para el modelado dinámico y cinemático de sistemas multicuerpo. Este capítulo tiene el objetivo de que la tesis sea autocontenida, es decir, que las referencias que se hagan a lo largo de la misma a aspectos relativos a la DSM estén presentes en la tesis. Además, este capítulo tiene el objetivo de facilitar al lector la nomenclatura utilizada a lo largo de la tesis.

En el capítulo 3 se presentan los algoritmos simbólicos de tipo genérico, es decir, no enfocados al contexto de la DSM. El capítulo está dividido en tres partes, en la primera parte (secciones 3.1 a 3.4) se presenta el proceso de atomización y se propone un conjunto de técnicas para realizar este proceso de la manera más eficaz posible. En la siguiente sección, se propone un conjunto de algoritmos para la sustitución de un símbolo en expresiones (sección 3.5) de forma que esta operación se realice directamente en expresiones ya atomizadas. A continuación, se pospone otro algoritmo para la diferenciación simbólica de expresiones atomizadas con respecto a símbolo (sección 3.6). En la ultima parte del capítulo (sección 3.7) se proponen un conjunto de técnicas y algoritmos para la generación y exportación de código optimizado basado en expresiones atomizadas.

El capítulo 4 se presenta un conjunto de técnicas y de algoritmos enfocados a ser usados en el contexto de la DSM. En primer lugar se realiza un análisis de la topología que presentan los sistemas multicuerpo y de las ventajas que pueden ser obtenidas a partir de esta. A continuación, en 4.3, se propone un conjunto de operadores para el cálculo de magnitudes cinemáticas (vectores de posición y de velocidad), con resultados optimizados en cuanto a tamaño de las expresiones y tiempo de cálculo. El cálculo de estos vectores de posición y de velocidad es independiente del formulismo y de la parametrización usados. En la sección 4.4 se propone un extensión del elemento tensor en el contexto de la DSM, generalizándolo a tensores de rango dos. Desde la sección 4.5 a la sección 4.8 se propone un método para el cálculo de la matriz de masa y el vector de fuerzas generalizadas basado en el PPV que saque máximo provecho

de la atomización y realice estos cálculos de la forma más rápida posible.

El capítulo 5 se presenta una técnica para reducir el tamaño de las expresiones simbólicas evitando que aparezcan expresiones sean susceptibles de ser simplificadas, tanto algebraica como trigonométricamente, sin perder la independencia respecto al tipo de formulación o parametrización que se utilice.

En el capítulo 6 se realiza un análisis de los algoritmos y técnicas que se presentan en la tesis. Los resultado obtenidos se evalúan en términos de tamaño de las expresiones y del código generado y tiempo de computación.

Por último, en el capítulo 7 se presentan las conclusiones obtenidas en esta tesis doctoral y en el capítulo 8 una serie de líneas futuras de trabajo.

CAPÍTULO 2

INTRODUCCIÓN AL MODELADO DE SISTEMAS MULTICUERPO

En este capítulo se hace una breve introducción teórico general sobre el modelado de sistemas multicuerpo.

Se describe el comportamiento cinemático y dinámico de un sistema multicuerpo con la intención introducir los conceptos y la notación usados en este documento.

De igual manera se hace una introducción a la simulación de sistemas multicuerpo y a los distintos elementos necesarios para poder llevarla a cabo.

2.1. Descripción de sistema multicuerpo

Un sistema multicuerpo es una colección de partículas naturales: cuerpos rígidos o flexibles. Estos cuerpos están sometidos a acciones procedentes de

su interacción con cuerpos del mismo sistema o del exterior.

La Dinámica de Sistemas Multicuerpo (DSM) estudia el comportamiento de dicho sistema. El problema de "dinámica directa" tiene por objeto, dadas las condiciones iniciales del sistema, determinar el estado del mismo bajo la influencia de las distintas acciones exteriores. El problema de "dinámica inversa" tiene por objeto determinar qué acciones son necesarias para que el sistema se encuentre en un estado determinado.

Esta tesis únicamente se centra en el concepto de cuerpo rígido. Un cuerpo se puede suponer rígido si no presenta deformación o su deformación es tan pequeña que no afecta su movimiento general. Aun así, la mayoría de la técnicas presentadas sí que son extensibles al campo de los sólidos flexibles.

Los cuerpos pueden estar conectados por diversos tipos de "enlaces" que restringen su movimiento relativo. Generalmente, este tipo de enlaces entre sólidos se pueden considerar sin masa.

Las fuerzas necesarias para garantizar dinámicamente las restricciones cinemáticas impuestas por lo enlaces se denominan "fuerzas de enlace". En el contexto particular del Principio de las Potencias Virtuales (PPV) estas fuerzas pueden ser caracterizadas mediante los multiplicadores del Lagrange.

Además de estas fuerzas, pueden actuar sobre el sistema fuerzas de otra naturaleza como fuerzas externas o bien, fuerzas especificadas mediante una ley constitutiva en términos del estado del sistema.

2.2. Modelado cinemático

El modelado cinemático tiene como fin la designación de una parametrización del sistema que permita determinar la posición, velocidad y aceleración de cada uno de los sólidos que componen dicho sistema.

2.2.1. Coordenadas generalizadas

En un modelo multicuerpo genérico la posición y orientación de cada uno de los elementos que lo compone puede ser fijada mediante dos magnitudes. Su posición puede ser acotada mediante el vector de posición $\mathbf{r}$ de uno de sus

puntos respecto de un punto arbitrario conocido y su orientación a través de una matriz **R** de cambio de base respecto a una orientación arbitraria conocida.

Tanto el vector **r** como la matriz **R** pueden depender de un conjunto de variables. Si estas variables dependen del tiempo se denominan *Coordenadas Generalizadas*, en cambio, si no dependen del tiempo se denominan *Parámetros geométricos*.

Se define el *Vector de coordenadas Generalizadas* como: $\boldsymbol{q} = [q_1, q_2, ..., q_p]^\mathsf{T}$, donde $q_i; i = 1, ..., p$ son coordenadas generalizadas.

Las coordenadas generalizadas pueden ser distancias y ángulos y, principalmente, pueden ser clasificadas en función de cómo se definan en absolutas, relativas o naturales.

Su derivada con respecto al tiempo $\dot{\boldsymbol{q}} = [\dot{q}_1, \dot{q}_2, ..., \dot{q}_p]^\mathsf{T}$ se denomina *Vector de Velocidades Generalizadas*.

Por último, la derivada segunda $\ddot{\boldsymbol{q}} = [\ddot{q}_1, \ddot{q}_2, ..., \ddot{q}_p]^\mathsf{T}$ se denomina *Vector de Aceleraciones Generalizadas*.

2.2.2. Concepto de Observador o Referencia, y de Velocidad y Aceleración de un punto

Un *Observador* (o *Referencia*) puede entenderse por una pareja formada por un *punto* (O) y una *base* ($\mathcal{B}$). El punto hace referencia al "Lugar del espacio desde donde se observa" y la base a la "Orientación desde la que se observa".

Los conceptos de vector de posición, velocidad y aceleración son dependientes del observador.

Se define el vector $\mathbf{r}^P_{O_R}$ como el vector de posición de un punto genérico P con respecto al punto O_R, donde el subíndice R indica que el punto es fijo en la referencia R.

El vector velocidad de un punto genérico P con respecto a la referencia R se define como:

$$\mathbf{v}^P_R = \left.\frac{d\mathbf{r}^P_{O_R}}{dt}\right|_R \tag{2.1}$$

donde R indica la referencia en la cual se calcula la derivada.

El vector aceleración del punto P con respecto a la referencia R, análogamente, se define:

$$\mathbf{a}_R^P = \left.\frac{d\mathbf{v}_R^P}{dt}\right|_R \tag{2.2}$$

Sea $\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}$ la matriz de cambio de base que proyecta las componentes de un vector definidas en la base $\mathcal{B}_1$ sobre la base $\mathcal{B}_2$, es decir:

$$\left\{\mathbf{u}\right\}_{\mathcal{B}_2} = \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}\left\{\mathbf{u}\right\}_{\mathcal{B}_1} \tag{2.3}$$

donde $\left\{\mathbf{u}\right\}_{\mathcal{B}}$ indica la tres tupla formada por las componentes del vector $\mathbf{u}$ proyectadas en la base $\mathcal{B}$.

La derivada de una vector arbitrario $\mathbf{u}$ con respecto a una referencia arbitraria se calcula según:

$$\left.\frac{d\mathbf{u}}{dt}\right|_{R_1} = \left.\frac{d\mathbf{u}}{dt}\right|_{R_2} + \widetilde{\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}}\mathbf{u} \tag{2.4}$$

en donde $\mathcal{B}_1$ es la base asociada la referencia R_1, $\mathcal{B}_2$ es la base asociada la referencia R_2 y

$$\widetilde{\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}} = \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}\cdot\dot{\mathbf{R}}_{\mathcal{B}_1}^{\mathcal{B}_2\,\mathsf{T}} \tag{2.5}$$

Se denomina vector velocidad angular de la referencia $\mathcal{B}_2$ respecto a la referencia $\mathcal{B}_1$ al vector[1] $\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}$, asociado a la matriz anti-simétrica $\widetilde{\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}}$.

Por consiguiente,

$$\left.\frac{d\mathbf{u}}{dt}\right|_{R_1} = \left.\frac{d\mathbf{u}}{dt}\right|_{R_2} + \boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}\wedge\mathbf{u} \tag{2.6}$$

[1] en esta tesis se usan indistintamente la notación $\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}$ como la notación $\boldsymbol{\omega}_{R_1}^{R_2}$

La aceleración angular se define:

$$\dot{\boldsymbol{\omega}}_{\mathcal{B}_1}^{\mathcal{B}_2} = \left.\frac{d\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}}{dt}\right|_{R_1} \tag{2.7}$$

2.2.3. Composición de movimientos

Sean las referencias (u observadores) R_1 y R_2 , sean O_1 y O_2, dos puntos fijos, respectivamente, en las citadas referencias. Por composición, para un punto genérico P, se tiene:

Composición de vectores de posición:

$$\mathbf{r}_{O_1}^{P} = \mathbf{r}_{O_1}^{O_2} + \mathbf{r}_{O_2}^{P} \tag{2.8}$$

Composición de velocidades. Derivando respecto al tiempo en la referencia R_1 la expresión anterior, se obtiene:

$$\mathbf{v}_{R_1}^{P} = \mathbf{v}_{R_2}^{P} + \mathbf{v}_{R_1}^{O_2} + \boldsymbol{\omega}_{R_1}^{R_2} \wedge \mathbf{r}_{O_2}^{P} \tag{2.9}$$

Composición de aceleraciones. Derivando respecto al tiempo en la referencia R_1 la expresión anterior, se obtiene:

$$\mathbf{a}_{R_1}^{P} = \mathbf{a}_{R_2}^{P} + \mathbf{a}_{R_1}^{O_2} + \boldsymbol{\omega}_{R_1}^{R_2} \wedge (\boldsymbol{\omega}_{R_1}^{R_2} \wedge \mathbf{r}_{O_2}^{P}) + 2\boldsymbol{\omega}_{R_1}^{R_2} \wedge \mathbf{v}_{R_2}^{P} \tag{2.10}$$

Es decir, si se conocen los vectores de posición, velocidad y aceleración de un punto respecto a una referencia, las ecuaciones anteriores permiten obtener las mismas magnitudes del punto P pero expresadas respecto a otra referencia.

2.2.4. Ecuaciones cinemáticas

En general, pueden existir relaciones entre las coordenadas generalizadas. Estas se denominan *relaciones geométricas de enlace* o *ecuaciones cinemáticas para las coordenadas generalizadas* y toman la forma general.

$$\mathbf{\Phi}(\boldsymbol{q}, t) = \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_q \end{bmatrix} = \mathbf{0} \tag{2.11}$$

donde g es el número de ecuaciones geométricas independientes.

Para un sistema acotado con p coordenadas generalizadas, existe un número $m = p - g$ de coordenadas independientes. Diferentes parametrizaciones pueden conducir a valores diferentes de p y g pero m es un invariante.

Las derivadas de las ecuaciones geométricas con respecto al tiempo expresan las relaciones que existen entre las velocidades generalizadas. Se denominan *ecuaciones de enlace cinemático* o *ecuaciones cinemáticas para las velocidades generalizadas*.

$$\dot{\mathbf{\Phi}} = \frac{d\mathbf{\Phi}}{dt}(\boldsymbol{q}, t) = \mathbf{0} \tag{2.12}$$

Aplicando la regla de la cadena se ve que estas ecuaciones son lineales en las velocidades generalizadas $\dot{\boldsymbol{q}}$.

$$\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} + \mathbf{\Phi}_{\mathbf{t}}(\dot{\boldsymbol{q}}, t) = \mathbf{0} \tag{2.13}$$

Donde

$$\mathbf{\Phi}_{\boldsymbol{q}} = \frac{\partial \mathbf{\Phi}}{\partial \boldsymbol{q}} \tag{2.14}$$

es la matriz jacobiana de las ecuaciones geométricas con respecto al vector de coordenadas generalizadas y

$$\mathbf{\Phi_t} = \frac{\partial \mathbf{\Phi}}{\partial t} \tag{2.15}$$

es el vector resultante de derivar parcialmente las ecuaciones geométricas con respecto al tiempo.

Se ha de notar que:

$$\mathbf{\Phi}_q = \dot{\mathbf{\Phi}}_{\dot{q}} \tag{2.16}$$

Donde:

$$\dot{\mathbf{\Phi}}_{\dot{q}} = \frac{\partial \dot{\mathbf{\Phi}}}{\partial \dot{\boldsymbol{q}}} \tag{2.17}$$

Así pues, la ecuación 2.13 se puede escribir como:

$$\dot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}_{\dot{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} + \mathbf{\Phi_t}(\dot{\boldsymbol{q}}, t) = \mathbf{0} \tag{2.18}$$

Además puede existir un conjunto r de relaciones adicionales entre las velocidades generalizadas que no tienen un origen geométrico. A estas se les denomina ecuaciones *no-holónomas* debido a que no provienen de la derivada con respecto al tiempo de ninguna relación entre coordenadas. Esta ecuaciones provienen de relaciones que deben satisfacer las velocidades generalizadas, como por ejemplo la condición de no deslizamiento.

Las ecuaciones *no-holónomas* pueden expresarse de forma general como:

$$\dot{\mathbf{\Phi}}^{\mathbf{NH}} = \begin{bmatrix} \Phi_1^{NH} \\ \Phi_2^{NH} \\ \vdots \\ \Phi_r^{NH} \end{bmatrix} = \dot{\mathbf{\Phi}}_{\dot{q}}^{\mathbf{NH}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} + \mathbf{\Phi}_{\mathbf{t}}^{\mathbf{NH}}(\boldsymbol{q}, t) = \mathbf{0} \tag{2.19}$$

En este documento el superíndice **NH** hace referencia a *No-Holónomo*.

Se denomina $c = g + r$ al número total de ecuaciones de enlace cinemático.

Se denomina *grados de libertad* al conjunto de velocidades generalizadas independientes, donde n es el número de grados de libertad del sistema, es decir: $n = p - c$. Nótese que $n \leq m$.

Es importante notar que si no existen ecuaciones no-holónomas ($r = 0$) se cumple que $m = n$ y por lo tanto la matriz jacobiana de velocidades ($\dot{\mathbf{\Phi}}_{\dot{q}}$) y de posiciones coinciden ($\mathbf{\Phi}_q$). En caso de existir relaciones no-holónomas, la matriz jacobiana de posición coincide solo con la sub-matriz de la matriz jacobiana de velocidades o aceleraciones asociada a las ecuaciones holónomas. Este hecho puede tenerse en cuenta para obtener una ventaja computacional en la resolución de los diferentes sistemas de ecuaciones involucran las citadas matrices jacobianas.

De esta forma, el conjunto completo de ecuaciones para las velocidades generalizadas puede escribirse como:

$$\begin{bmatrix} \dot{\mathbf{\Phi}}_{\dot{q}} \\ \dot{\mathbf{\Phi}}^{\mathbf{NH}}_{\dot{q}} \end{bmatrix} \dot{q} + \begin{bmatrix} \mathbf{\Phi_t} \\ \mathbf{\Phi^{NH}_t} \end{bmatrix} = \mathbf{0} \tag{2.20}$$

Por brevedad en la notación, y mientras no se indique lo contrario, este conjunto completo de ecuaciones para velocidades se denotará:

$$\dot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}_{\dot{q}}(q,t)\dot{q} + \mathbf{\Phi_t}(q,t) = \mathbf{0} \tag{2.21}$$

Es decir, de forma análoga a la ecuación: 2.18.

Este abuso en la notación puede llevar al error, así que se ha de prestar atención en no confundir la matriz jacobiana asociada a las ecuaciones geométricas ($\mathbf{\Phi}_q$) y la asociada a las velocidades ($\dot{\mathbf{\Phi}}_{\dot{q}}$).

En este documento se define $\boldsymbol{\beta} = -\mathbf{\Phi_t}(q,t)$, de forma que la ecuación anterior puede expresarse como:

$$\dot{\mathbf{\Phi}}_{\dot{q}}(q,t)\dot{q} = \boldsymbol{\beta} \tag{2.22}$$

Derivando respecto al tiempo las ecuaciones anteriores 2.21 o 2.19 (ecuacio-

nes de velocidades holónomas y no-holónomas), se obtienen las ecuaciones que relacionan las aceleraciones generalizadas.

$$\ddot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}_{\dot{q}}(q,t)\ddot{q} + \dot{\mathbf{\Phi}}_{q}(\dot{q},q,t)\dot{q} + \dot{\mathbf{\Phi}}_{\mathbf{t}}(\dot{q},q,t) = \mathbf{0} \tag{2.23}$$

Se ha de notar que las matrices jacobianas de este sistema para las aceleraciones coincide con las matrices del sistema para velocidades, es decir:

$$\dot{\mathbf{\Phi}}_{\dot{q}} = \ddot{\mathbf{\Phi}}_{\ddot{q}} \text{ y } \dot{\mathbf{\Phi}}_{q} = \ddot{\mathbf{\Phi}}_{\dot{q}} \tag{2.24}$$

Donde:

$$\ddot{\mathbf{\Phi}}_{\ddot{q}} = \frac{\partial \ddot{\mathbf{\Phi}}}{\partial \ddot{q}} \text{ y } \ddot{\mathbf{\Phi}}_{\dot{q}} = \frac{\partial \ddot{\mathbf{\Phi}}}{\partial \dot{q}} \tag{2.25}$$

Así pues, la ecuación 2.23 se puede escribir como:

$$\ddot{\mathbf{\Phi}} = \ddot{\mathbf{\Phi}}_{\ddot{q}}(q,t)\ddot{q} + \ddot{\mathbf{\Phi}}_{\dot{q}}(\dot{q},q,t)\dot{q} + \dot{\mathbf{\Phi}}_{\mathbf{t}}(\dot{q},q,t) = \mathbf{0} \tag{2.26}$$

De forma análoga, en este documento se denomina $\gamma = -\ddot{\mathbf{\Phi}}_{\dot{q}}(\dot{q},q,t)\dot{q} - \dot{\mathbf{\Phi}}_{\mathbf{t}}(\dot{q},q,t)$, de forma que la ecuación anterior puede expresarse como:

$$\ddot{\mathbf{\Phi}}_{\ddot{q}}(q,t)\ddot{q} = \gamma \tag{2.27}$$

Hay que destacar la linealidad de las ecuaciones anteriores en términos de las aceleraciones generalizadas.

2.3. Modelado Dinámico

El modelado dinámico de sistemas multicuerpo tiene como fin la obtención de las ecuaciones que describen el comportamiento dinámico del mismo.

2.3.1. Planteamiento de las ecuaciones dinámicas

Existen diversas formas de abordar la obtención de las ecuaciones de la dinámica. Un gran número de autores utilizan la formulación de Newton-Euler, mientras que otros se basan en las ecuaciones de Lagrange.

Aunque la mayoría de las técnicas presentadas en esta tesis están pensadas para ser de carácter general, independientes de la formulación, para obtener las ecuaciones dinámicas se ha utilizado el PPV (o principio de Jourdain).

Se ha decido usar el este principio por algunas de las ventajas que presenta:

- Es la metodología más utilizada ya que permite abordar el análisis dinámico sin tener que plantear un número elevado de ecuaciones.
- Produce las ecuaciones más compactas y con propiedades computacionales más interesantes (por ejemplo: simetría en la matriz de masa).
- Representa desde la perspectiva simbólica toda la riqueza conceptual de forma que las técnicas presentadas en esta tesis son de aplicabilidad general. Por ejemplo, tanto las ecuaciones de Lagrange, así como las de Newton-Euler pueden ser obtenidas a partir de este principio.

2.3.1.1. Acciones de inercia actuantes sobre un sólido rígido

Para un sólido[2] genérico S_k, donde B_k es un punto cualquiera cinemáticamente perteneciente a dicho sólido, G_k es su centro de masa[3], y m_k y $\mathbf{I}_{B_k}^{S_k}$ son, respectivamente, la masa y el tensor de inercia[4] definido en el punto B_k del sólido, las fuerzas y momentos *de inercia* o **de D'Almbert** vienen determinadas por:

[2]Por sólido rígido se entiende a un conjunto de puntos del espacio que se mueven de manera que las distancias relativas entre ellos no varían con el tiempo.

[3]Ese punto no tienen porque coincidir con la posición de ninguna de las partículas del sólido.

[4]Algunos autores, por simplicidad, definen este tensor respecto al centro de masas. En esta tesis, por generalizar su definición, se permite que sea definido respecto a cualquier punto del sólido y con sus componentes proyectadas de forma natural en la base del sólido.

$$\begin{aligned} \mathcal{F}^{S_k} &= -m_k \mathbf{a}_{RI}^{G_k} \\ \mathcal{M}_{B_k}^{S_k} &= -\left.\frac{d\mathbf{h}_{B_k}^{S_k}}{dt}\right|_{RI} - m_k \mathbf{r}_{B_k}^{G_k} \wedge \mathbf{a}_{RI}^{B_k} \end{aligned} \tag{2.28}$$

Donde $\mathbf{a}_{RI}^{G_k}$ es la aceleración lineal de centro de masas del sólido con respecto a la referencia inercial del sólido. $\mathbf{h}_{B_k}^{S_k}$ es el momento angular del sólido definido en el punto B_k del mismo y que se define como:

$$\mathbf{h}_{B_k}^{S_k} = \mathbf{I}_{B_k}^{S_k} \boldsymbol{\omega}_{RI}^{S_k} \tag{2.29}$$

Se denomina torsor[5] *de acciones de inercia* o *de acciones de inercia de D'Alembert* para un sólido genérico S_k aplicado en el punto B_k a la pareja de fuerza y momento de inercia:

$$\mathcal{W}_{B_k}^{S_k} = \begin{bmatrix} \mathcal{F}^{S_k} \\ \mathcal{M}_{B_k}^{S_k} \end{bmatrix} = \begin{bmatrix} -m_k \mathbf{a}_{RI}^{G_k} \\ -\left.\frac{d\mathbf{h}_{B_k}^{S_k}}{dt}\right|_{RI} - m_k \mathbf{r}_{B_k}^{G_k} \wedge \mathbf{a}_{RI}^{B_k} \end{bmatrix} \tag{2.30}$$

2.3.1.2. Acciones exteriores actuantes sobre un sólido rígido

Se denomina torsor de las w_k *acciones exteriores* al sólido S_k al torsor:

$$\boldsymbol{w}_{B_k}^{S_k} = \sum_{j=1}^{w_k} \boldsymbol{w}_{B_k j}^{S_k} = \sum_{j=1}^{w_k} \begin{bmatrix} \mathbf{f}^{S_k}{}_j \\ \mathbf{m}_{B_k j}^{S_k} \end{bmatrix} \tag{2.31}$$

Donde $\boldsymbol{w}_{B_k j}^{S_k}$ es el torsor asociado a la acción exterior j-ésima1 formado a la pareja de fuerza $\mathbf{f}^{S_k}{}_j$ y momento $\mathbf{m}_{B_k j}^{S_k}$.

[5] En inglés *wrench*

2.3.1.3. Ecuaciones dinámicas

Utilizando la notación de torsores empleada anteriormente las ecuaciones de Newton-Euler analíticas para un sistema formado por n_S sólidos puede escribirse:

$$\mathbf{e} = \sum_{k=1}^{n_S} \begin{bmatrix} -m_k \mathbf{a}_{RI}^{G_k} \\ -\frac{d\mathbf{h}_{B_k}^{S_k}}{dt}\Big|_{RI} - m_k \mathbf{r}_{B_k}^{G_k} \wedge \mathbf{a}_{RI}^{B_k} \end{bmatrix} + \sum_{k=1}^{n_S} \sum_{j=1}^{w_k} \begin{bmatrix} \mathbf{f}^{S_k}{}_j \\ \mathbf{m}_{B_k j}^{S_k} \end{bmatrix} = \mathbf{0} \tag{2.32}$$

2.3.1.4. Principio de las potencias virtuales

El PPV puede ser enunciado como:

La suma de la potencia virtual de las acciones actuantes sobre un sistema,incluidas las acciones de inercia, es igual a cero.

Es decir, es la versión analítica de la Segunda Ley de Newton.

En su aplicación resulta de interés la siguiente propiedad:

La potencia virtual de las acciones de enlace actuantes sobre el sistema es cero para velocidades virtuales compatibles con los enlaces.

La propiedad anterior no es otra cosa que la versión analítica de la Tercera Ley de Newton y puede ser utilizada para la caracterización analítica de las acciones de enlace.

Sea el torsor genérico $\boldsymbol{w}_B^S = \begin{bmatrix} \mathbf{f}^\mathsf{T} & \mathbf{m}_B^\mathsf{T} \end{bmatrix}^\mathsf{T}$ actuante sobre un punto B genérico perteneciente a un sólido S, entonces, la potencia virtual del torsor cuando solamente cambia la velocidad $\dot{q}_i$ y ésta toma el valor virtual $\dot{q}_i^v$ puede escribirse como:

$$\dot{\mathbf{w}}_{\dot{q}_i} = \begin{bmatrix} \mathbf{f}^S \\ \mathbf{m}_B^S \end{bmatrix}^\mathsf{T} \frac{\partial}{\partial \dot{q}_i} \begin{bmatrix} \mathbf{v}_{RI}^B \\ \boldsymbol{\omega}_{RI}^S \end{bmatrix} \dot{q}_i^v = \boldsymbol{w}_B^S \frac{\partial \boldsymbol{t}_B^S}{\partial \dot{q}_i} \tag{2.33}$$

donde $\mathbf{v}_{RI}^{B_S}$ es la velocidad del punto B perteneciente al sólido S respecto de una referencia inercial RI y $\boldsymbol{\omega}_{RI}^S$ es la velocidad angular del sólido S respecto a

la referencia RI. De manera dual al concepto de torsor, la pareja de velocidades $\boldsymbol{t}_B^S = \begin{bmatrix} \mathbf{v}_{RI}^{B_S\,\mathsf{T}} & \boldsymbol{\omega}_{RI}^{S\,\mathsf{T}} \end{bmatrix}^{\mathsf{T}}$ se le denomina "grupo cinemático" o "torsor cinemático".

Tomando los torsores de inercia de cada uno de los n_S sólidos y de las w_k acciones exteriores aplicadas sobre cada sólido S_k se plantea el PPV por cada velocidad generalizada $\dot{q}_i$ obteniéndose:

$$\mathrm{e}_i = \sum_{k=1}^{n_S} \begin{bmatrix} \mathcal{F}^k \\ \mathcal{M}_{B_k}^k \end{bmatrix}^{\mathsf{T}} \frac{\partial}{\partial \dot{q}_i} \begin{bmatrix} \mathbf{v}_{RI}^{B_k} \\ \boldsymbol{\omega}_{RI}^{S_k} \end{bmatrix} \dot{q}_i^v + \sum_{k=1}^{n_S} \sum_{j=1}^{w_k} \begin{bmatrix} \mathbf{f}^{S_k}{}_j \\ \mathbf{m}_{B_k j}^{S_k} \end{bmatrix}^{\mathsf{T}} \frac{\partial}{\partial \dot{q}_i} \begin{bmatrix} \mathbf{v}_{RI}^{B_k} \\ \boldsymbol{\omega}_{RI}^{S_k} \end{bmatrix} \dot{q}_i^v = 0 \tag{2.34}$$

Para un sistema con un número p de coordenadas generalizadas, de forma compacta, el conjunto completo de ecuaciones toma la siguiente forma:

$$\mathbf{e} = \sum_{k=1}^{n_S} \begin{bmatrix} \mathcal{F}^k \\ \mathcal{M}_{B_k}^k \end{bmatrix}^{\mathsf{T}} \frac{\partial}{\partial \dot{\boldsymbol{q}}} \begin{bmatrix} \mathbf{v}_{RI}^{B_k} \\ \boldsymbol{\omega}_{RI}^{S_k} \end{bmatrix} + \sum_{k=1}^{n_S} \sum_{j=1}^{w_k} \begin{bmatrix} \mathbf{f}^{S_k}{}_j \\ \mathbf{m}_{B_k j}^{S_k} \end{bmatrix}^{\mathsf{T}} \frac{\partial}{\partial \dot{\boldsymbol{q}}} \begin{bmatrix} \mathbf{v}_{RI}^{B_k} \\ \boldsymbol{\omega}_{RI}^{S_k} \end{bmatrix} = \mathbf{0} \tag{2.35}$$

Estas ecuaciones forman un sistema de p ecuaciones y como se ve en las mismas, el PPV permite obtener las ecuaciones dinámicas como un sumatorio de productos de torsores por velocidades virtuales (Fuerzas por velocidades y momentos por velocidades angulares).

Hay que remarcar que los factores $\dot{q}^v$ se han eliminado; se simplifican por ser su valor arbitrario, si son coordenadas independientes.

Los movimientos virtuales empleados son en general, compatibles con los enlaces presentes en el sistema mecánico, e incompatibles con otros. Por a la tercera ley de Newton, los torsores de enlace cuyos enlaces son respetados (no rotos) por la acotación de partida no producen potencia virtual. Por ello, solo las incógnitas de enlace asociadas a los enlaces que no son compatibles con la acotación del sistema, pueden aparecer en las ecuaciones resultantes. Por incógnita de enlace se entiende al conjunto de incógnitas utilizadas para caracterizar las fuerzas y momentos de enlace actuantes sobre los sólidos del sistema.

En las referencias [33, 4] puede encontrarse una descripción más detallada y

con más rigor del PPV.

2.3.1.5. Incógnitas de enlace

La ecuación anterior permite determinar en un instante arbitrario de tiempo t, el valor de las aceleraciones generalizadas $\ddot{\boldsymbol{q}}$ del sistema. Esta información es utilizada para realizar la integración numérica del sistema multicuerpo, denominada simulación dinámica, que se verá, más en detalle, más adelante en este capítulo.

Las citadas aceleraciones no son las únicas incógnitas del problema dinámico. También aparecen como incógnitas en el sistema anterior las incógnitas de enlace $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, ..., \epsilon_n]^{\mathsf{T}}$ asociadas a los enlaces que no son compatibles con la acotación del sistema. El número de incógnitas de enlace es igual al número de movimientos impedidos respecto al sistema de sólidos libre.

En un sistema planteado en términos de un conjunto de p coordenadas generalizadas $\boldsymbol{q}$ para el que existen $c = g + r$ relaciones cinemáticas entre las aceleraciones generalizadas $\ddot{\boldsymbol{q}}$, el conjunto de ecuaciones

$$\boldsymbol{\Phi}_{\dot{q}}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{\Phi}}_{\dot{q}}\dot{\boldsymbol{q}} + \dot{\boldsymbol{\Phi}}_{\mathbf{t}} = \mathbf{0} \tag{2.36}$$

ha de ser añadido al conjunto 2.35 de ecuaciones dinámicas para que este sistema de ecuaciones sea completo.

2.4. Estructura del sistema multicuerpo

Las ecuaciones dinámicas son lineales con respecto a las aceleraciones generalizadas, $\ddot{\boldsymbol{q}}$.

Se define la matriz de masa $\mathbf{M}$ como la parte que "acompaña" a las aceleraciones generalizadas en las ecuaciones de la dinámica y depende exclusivamente de las coordenadas generalizadas y de los parámetros del sistema. En general, la matriz $\mathbf{M}$ es simétrica y definida positiva.

El resto de términos de las ecuaciones de la dinámica se denomina vector $\boldsymbol{\delta}$.

Por otra parte las acciones de enlace son lineales ($\mathbf{V}_\epsilon$) con respecto a las incógnitas de enlace, $\boldsymbol{\epsilon}$.

El conjunto de las ecuaciones dinámicas toma la forma general:

$$\mathbf{M}\ddot{\boldsymbol{q}} + \mathbf{V}_\epsilon^\top \boldsymbol{\epsilon} = \boldsymbol{\delta} \tag{2.37}$$

O escrito en forma matricial:

$$\begin{bmatrix}\mathbf{M} & \mathbf{V}_\epsilon^\top\end{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{q}} \\ \boldsymbol{\epsilon}\end{bmatrix} = \boldsymbol{\delta} \tag{2.38}$$

Donde $\boldsymbol{\delta}$ o término independiente contiene el resto de fuerzas generalizadas: Coriolis, centrípetas, constitutivas y externas. En esta tesis ha este término se le ha denominado "vector de fuerzas generalizadas".

Si a estas ecuaciones les añadimos las relaciones de las aceleraciones generalizadas (Ecuación 2.27) para formar un sistema completo, se tiene que:

$$\begin{bmatrix}\mathbf{M} & \mathbf{V}_\epsilon^\top \\ \boldsymbol{\Phi}_{\dot{q}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{q}} \\ \boldsymbol{\epsilon}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\delta} \\ \boldsymbol{\gamma}\end{bmatrix} \tag{2.39}$$

2.4.0.6. Multiplicadores de Lagrange

La fuerza generalizada asociada a las acciones de enlace puede expresarse también en términos del vector de multiplicadores de Lagrange $\boldsymbol{\lambda}$.

Es decir,

$$\mathbf{V}_\epsilon^\top \boldsymbol{\epsilon} = \boldsymbol{\Phi}_{\dot{q}}^\top \boldsymbol{\lambda} \tag{2.40}$$

Así pues, el sistema completo de ecuaciones quedaría de la siguiente forma:

$$\begin{bmatrix}\mathbf{M} & \boldsymbol{\Phi}_{\dot{q}}^\top \\ \boldsymbol{\Phi}_{\dot{q}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\delta} \\ \boldsymbol{\gamma}\end{bmatrix} \tag{2.41}$$

Como la matriz de masa $\mathbf{M}$ es simétrica, la matriz del sistema completo también es simétrica, aunque, en general, no tiene porqué ser definida positiva.

2.4.0.7. Solución del sistema

A la solución del sistema de ecuaciones anterior para la obtención de las aceleraciones generalizadas $\ddot{\boldsymbol{q}}$ y de las incógnitas de enlace ($\boldsymbol{\lambda}$ o $\boldsymbol{\epsilon}$) conocidos los valores de las coordenadas y velocidades generalizadas en un instante concreto de tiempo t se denomina "problema dinámico".

2.5. Simulación de sistemas multicuerpo

Una simulación, por definición, es una imitación de un proceso, sistema o suceso del mundo real. Para realizar dicha simulación se requiere de un modelo, en este caso, el modelo multicuerpo, y la simulación se encarga de representar el comportamiento de este modelo a lo largo del tiempo.

En el contexto de las DSM las simulaciones más relevantes son la simulación analítica y la simulación dinámica.

2.5.1. Problema de ensamblaje

Antes de comenzar la simulación, las coordenadas generalizadas deben satisfacer el conjunto de ecuaciones geométricas y en general, el valor que toman las coordenadas generalizadas no tiene por qué satisfacerlas.

Se denomina "Problema de ensamblaje" a la determinación del conjunto de valores de las coordenadas que satisfagan la ecuación 2.11, es decir:

$$\boldsymbol{\Phi}(\boldsymbol{q}, t) = \mathbf{0} \tag{2.42}$$

Cuando las coordenadas generalizadas satisfacen la ecuación anterior se dice que sistema está "montado".

Se entiende por "valor inicial del vector de coordenadas generalizadas", $\boldsymbol{q}_0$,

al conjunto de valores iniciales que toman las coordenadas generalizadas al inicio de la simulación.

La forma más usual de resolver este sistema de ecuaciones no lineales es utilizar el método de Newton-Raphson que sustituye el sistema no lineal de ecuaciones por una aproximación en serie de Taylor de primer orden respecto de las incógnitas.

Así pues, las ecuaciones geométricas 2.11 linealizardas en torno al valor de $\boldsymbol{q}^k$, correspondiente a la iteración k-ésima se pueden escribir de la siguiente manera:

$$\boldsymbol{\Phi}(\boldsymbol{q}^{k-1}, t) + \boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}^{k-1}, t)(\boldsymbol{q}^k - \boldsymbol{q}^{k-1}) = \mathbf{0} \tag{2.43}$$

Si la iteración del algoritmo Newton-Raphson comienza con $k = 1$, la solución más próxima al conjunto de valores iniciales con la precisión deseada, tol_Φ, tol_q, se obtendrá cuando se cumpla $\left|\boldsymbol{\Phi}(\boldsymbol{q}^k, t)\right| < tol_\Phi$ y $\left|\boldsymbol{q}^k - \boldsymbol{q}^{k-1}\right| < tol_q$.

En general el número de ecuaciones geométricas es inferior al de coordenadas generalizadas, es decir, el sistema de ecuaciones lineal del método de Newton-Raphson es compatible indeterminado. Por lo tanto, habrá un conjunto infinito de soluciones, de entre las cuales habrá que elegir una. Se ha de elegir la solución que satisfaga el sistema y que modifique el valor de la iteración anterior $\boldsymbol{q}^{k-1}$ lo mínimo posible.

A tal fin, una posibilidad es la utilización de la pseudoinversa en la resolución de 2.43 de forma que en cada iteración la variación $\boldsymbol{q}^k - \boldsymbol{q}^{k-1}$ tiene norma mínima. Es decir:

$$\boldsymbol{q}^k = \boldsymbol{q}^{k-1} + \boldsymbol{\Phi}_{\boldsymbol{q}}^{\dagger}(\boldsymbol{q}^{k-1}, t) \cdot \boldsymbol{\Phi}(\boldsymbol{q}^{k-1}, t) \tag{2.44}$$

En ocasiones es inevitable la introducción de ecuaciones redundantes, en tal caso el sistema 2.43 puede ser incompatible, salvo en el entorno de la solución. Para evitar este problema, puede ser usada la misma solución basada en la pseudoinversa.

De la misma forma que los valores iniciales dados a las coordenadas gene-

ralizadas no tienen por qué satisfacer el conjunto de ecuaciones geométricas, el conjunto de valores iniciales, $\dot{\boldsymbol{q}}_0$, asignado a las velocidades generalizadas tampoco tiene por qué satisfacer las ecuaciones cinemáticas para las velocidades generalizadas.

El problema de velocidades inicial se resuelve de forma análoga obteniendo los valores del vector de velocidades generalizas que menos modifique dicho valor inicial de las mismas. Es decir:

$$\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}_0 + \dot{\boldsymbol{\Phi}}_{\dot{\boldsymbol{q}}}^{\dagger}(\boldsymbol{q}, \dot{\boldsymbol{q}}_0, t)\left(\boldsymbol{\beta} - \dot{\boldsymbol{\Phi}}_{\dot{\boldsymbol{q}}}(\boldsymbol{q}, \dot{\boldsymbol{q}}_0, t) \cdot \dot{\boldsymbol{q}}_0\right) \tag{2.45}$$

Se ha destacar que el problema de ensamblaje es equivalente a la proyección de las coordenadas y velocidades sobre el plano tangente a las restricciones (denominada "corrección") que es requerida en los problemas de integración asociados a la simulación dinámica del sistema.

Por último cabe señalar que corregir las aceleraciones generalizadas de un sistema, por lo general, tiene interés en caso de que se quiera realizar únicamente una simulación cinemática.

2.5.2. Problema de posición, velocidad y aceleración inicial

Las ecuaciones geométricas, en general, no forman un conjunto completo de ecuaciones (número de ecuaciones geométricas independientes es menor que número el de coordenadas generalizadas). Puede que se desee no sólo que el valor de las coordenadas generalizadas tome un valor compatible con las ecuaciones geométricas (problema de ensamblaje) sino que además el mecanismo presente una posición inicial específica. Ya se ha comentado que pueden existir infinitas soluciones y se puede desear estar cerca de una de ellas.

En este caso es necesario aumentar el conjunto de ecuaciones geométricas, con ecuaciones adicionales destinadas a fijar la posición del inicial sistema. El número de ecuaciones adicionales ha de coincidir con el número de coordenadas independientes del sistema, en el caso en que se desee una posición en que todas las coordenada generalizadas queden fijadas.

Generalmente estas ecuaciones que se añaden toman la forma:

$$q_j - q_{j_0} = 0 \tag{2.46}$$

El conjunto total de ecuaciones geométricas, denominado $\boldsymbol{\Phi}^{Init}(\boldsymbol{q}, t)$, esta formado por las ecuaciones geométricas vistas en 2.11 más el conjunto de ecuaciones anteriores, así pues:

$$\boldsymbol{\Phi}^{Init}(\boldsymbol{q}, t) = \begin{bmatrix} \boldsymbol{\Phi}(\boldsymbol{q}, t) \\ \vdots \\ q_j - q_{j_0} \\ \vdots \end{bmatrix} = \mathbf{0} \tag{2.47}$$

Para obtener el valor de las coordenadas generalizadas, de forma que el sistema se encuentre montado y en la posición deseada, se aplica el algoritmo de Newton-Raphson descrito en el apartado anterior al conjunto de ecuaciones geométricas ampliado con las ecuaciones adicionales para fijar la posición.

Análogamente, las ecuaciones cinemáticas para velocidades generalizadas no forman un conjunto completo de ecuaciones (número de ecuaciones para velocidades independientes es menor que el número de coordenadas generalizadas). Así si se desea inicializar las velocidades de forma que su valor sea compatible con ellas y además especificar la velocidad concreta del sistema, es necesario introducir un conjunto adicional de ecuaciones para que el sistema resultante sea compatible determinado.

Generalmente estas ecuaciones toman la forma:

$$\dot{q}_j - \dot{q}_{j_0} = 0 \tag{2.48}$$

De forma análoga al problema de posición, el conjunto total de ecuaciones cinemáticas para velocidades generalizadas, denominado $\dot{\boldsymbol{\Phi}}^{Init}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$, esta formado por las ecuaciones geométricas vistas en 2.12 más el conjunto de ecuaciones anteriores, así pues:

$$\dot{\mathbf{\Phi}}^{Init}(\boldsymbol{q},\dot{\boldsymbol{q}},t) = \begin{bmatrix} \dot{\mathbf{\Phi}}(\boldsymbol{q},\dot{\boldsymbol{q}},t) \\ \vdots \\ \dot{q}_j - \dot{q}_{j_0} \\ \vdots \end{bmatrix} = \mathbf{0} \tag{2.49}$$

Aunque por lo general carece de interés práctico, un problema análogo al anterior puede resolverse para el caso en que se desee especificar la aceleración del sistema en el instante inicial, por ejemplo en una simulación cinemática.

2.5.3. Simulación cinemática

Un problema que suele resultar de interés es la simulación únicamente del movimiento del sistema, para por ejemplo, conocer su espacio de trabajo.

Este problema puede ser abordado de dos formas, según la naturaleza del mismo.

2.5.3.1. Sin integración

Si el sistema no tiene restricciones no-holónomas basta con introducir un conjunto adicional de ecuaciones para las coordenadas, velocidades y aceleraciones generalizadas. Este conjunto de ecuaciones adicionales fijan la evolución de un conjunto de las coordenadas, velocidades y aceleraciones generalizadas elegidas como independientes para realizar la simulación. En este caso la forma general de las citadas ecuaciones es:

- $q_j - f_j(t) = 0$
- $\dot{q}_j - \dot{f}_j(t) = 0$
- $\ddot{q}_j - \ddot{f}_j(t) = 0$

De esta forma la simulación cinemática del mecanismo durante un intervalo de tiempo, se consigue mediante la solución para los diferentes instantes de tiempo de los problemas de posición, velocidad y aceleración (Sección 2.5.2) incrementados respectivamente con las ecuaciones para posiciones, velocidades y aceleraciones anteriores. Es decir:

$$\boldsymbol{\Phi}^{K.S.}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{\Phi}(\boldsymbol{q}) \\ \vdots \\ q_j - f_j(t) \\ \vdots \end{bmatrix} = \mathbf{0} \tag{2.50}$$

$$\dot{\boldsymbol{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} \dot{\boldsymbol{\Phi}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \vdots \\ \dot{q}_j - \dot{f}_j(t) \\ \vdots \end{bmatrix} = \mathbf{0} \tag{2.51}$$

y

$$\ddot{\boldsymbol{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \begin{bmatrix} \ddot{\boldsymbol{\Phi}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \\ \vdots \\ \ddot{q}_j - \ddot{f}_j(t) \\ \vdots \end{bmatrix} = \mathbf{0} \tag{2.52}$$

$t \leftarrow 0$
mientras $t < t_{final}$ **hacer**
  Resolver $\boldsymbol{\Phi}^{K.S.}(\boldsymbol{q}) = \mathbf{0}$
  Resolver $\dot{\boldsymbol{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{0}$
  Resolver $\ddot{\boldsymbol{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{0}$
  $t \leftarrow t + \Delta t$

Figura 2.1.: Resolución numérica sin integración

Así se obtendrían los valores de $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ en los citados instantes.

2.5.3.2. Con integración

Si se han resuelto los problemas de posición y velocidad iniciales, es posible realizar la simulación anterior sin especificar las ecuaciones adicionales para coordenadas independientes (o especificando sólo un subconjunto de estas). También es posible realizar la simulación sin especificar las ecuaciones para velocidades generalizadas (o especificando sólo un subconjunto de estas). En estos casos siempre es necesario que las ecuaciones para aceleraciones independientes sean especificadas.

El sistema $\frac{d}{dt}\boldsymbol{q} = \dot{\boldsymbol{q}}$ y $\frac{d}{dt}\dot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}$ es un sistema de ecuaciones diferenciales que puede ser integrado para obtener la evolución de $\boldsymbol{q}$ y de $\dot{\boldsymbol{q}}$. Así, se puede evitar el proporcionar ecuaciones para coordenadas y velocidades generalizadas, únicamente siendo necesarias las ecuaciones para las aceleraciones independientes. En el caso de que se tenga alguna de las ecuaciones para coordenadas o velocidades, esto permite sustituir los valores de la integración por los valores más exactos proporcionados por dichas ecuaciones.

Si alguna de las ecuaciones para el problema de posiciones $\boldsymbol{\Phi}^{K.S.}$ y de velocidades $\dot{\boldsymbol{\Phi}}^{K.S.}$ no está disponible, como es el caso de sistemas no-holónomos, basta con resolver las ecuaciones anteriores después de realizar la integración. Esto a su vez implica resolver los problemas de posición y velocidad iniciales.

La simulación cinemática de sistemas con ecuaciones no-holónomas requiere de la integración numérica descrita anteriormente. El algoritmo anterior queda como se ve en la figura 2.2:

Por simplicidad, el algoritmo de la figura 2.2 de integración se denomina *Método de Euler Explícito*, aunque puede ser extendido trivialmente a cualquier otro integrador.

Se trata de un procedimiento de integración numérica para resolver ecuaciones diferenciales ordinarias a partir de un valor inicial dado. Una posible interpretación es que en cada instante de tiempo las funciones del tiempo $\boldsymbol{q}$ y $\dot{\boldsymbol{q}}$ se pueden sustituir por su aproximación en serie de Taylor de primer orden.

$$\begin{aligned} \boldsymbol{q}(t+\Delta t) &\approx \boldsymbol{q}(t) + \dot{\boldsymbol{q}}(t)\Delta t \\ \dot{\boldsymbol{q}}(t+\Delta t) &\approx \dot{\boldsymbol{q}}(t) + \ddot{\boldsymbol{q}}(t)\Delta t \end{aligned} \tag{2.53}$$

$t \leftarrow 0$
Resolver $\mathbf{\Phi}^{Init}(\boldsymbol{q}) = 0$
Resolver $\dot{\mathbf{\Phi}}^{Init}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$
mientras $t < t_{final}$ **hacer**
  Resolver $\ddot{\mathbf{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{0}$
  $\boldsymbol{q} \leftarrow \boldsymbol{q} + \dot{\boldsymbol{q}} \cdot \Delta t$
  $\dot{\boldsymbol{q}} \leftarrow \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}} \cdot \Delta t$
  si *Ecuaciones no-holónomas* **entonces**
    Resolver $\mathbf{\Phi}^{K.S.}(\boldsymbol{q}) = \mathbf{0}$
    Resolver $\dot{\mathbf{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{0}$
  $t \leftarrow t + \Delta t$

Figura 2.2.: Integración numérica para cinemática (Euler explícito)

El error de la aproximación es de segundo orden, es decir, del orden de Δt^2 , así que si se desea precisión en la integración se deben elegir valores pequeños para Δt.

El algoritmo anterior tiene una mejora inmediata, consiste en utilizar la aproximación de segundo orden para obtener $\boldsymbol{q}$, es decir:

$$\boldsymbol{q} = \boldsymbol{q} + (\dot{\boldsymbol{q}} + \frac{1}{2}\ddot{\boldsymbol{q}}\Delta t)\Delta t \tag{2.54}$$

Aun así hay que destacar que a día de hoy existen numerosos métodos numéricos de integración y que cualquiera de ellos es válido para la integración en DSM.

2.5.4. Corrección de la posición y la velocidad

Durante la simulación cinemática o dinámica del sistema[6] puede ocurrir que el valor de las coordenadas generalizadas y el de las velocidades generalizadas

[6]Estos procedimientos normalmente implican la integración de un sistema de ecuaciones diferenciales

dejen de satisfacer el sistema de ecuaciones geométricas y el de ecuaciones cinemáticas para las velocidades generalizadas. Esto es debido a la imprecisión inherente al proceso de integración numérica.

Por ejemplo, con el algoritmo de integración de euler visto anteriormente, es fácil comprobar cómo en cada instante de tiempo se van sumando errores del orden de $\mathcal{O}(\Delta t^2)$, así que en el peor de los casos el error en el instante de tiempo t es del orden $\mathcal{O}(\frac{t}{\Delta t}\Delta t^2) = \mathcal{O}(t \cdot \Delta t)$.

En el esquema de integración no se ha hecho nada especial para que a lo largo del proceso se satisfagan las ecuaciones cinemáticas para coordenadas y velocidades. Así que los errores en la satisfacción de las ecuaciones geométricas y cinemáticas serán del mismo orden de magnitud.

El efecto físico del citado error es que los enlaces no garantizados por la acotación (para los que se habían planteado ecuaciones cinemáticas) dejan de satisfacerse poco a poco y el sistema que se está integrando deja de comportarse como el sistema original, lo que facilita la desestabilización del proceso de integración.

Evidentemente, la disminución del paso de integración Δt o la utilización de algoritmos de integración de orden superior hacen más pequeño el problema anterior. Esto lo único que supone es que la desestabilización del proceso de integración ocurra más tarde en el tiempo, pero no lo impide.

Una posible forma de evitar la indeseable desestabilización es mediante la utilización del algoritmo de corrección de coordenadas y velocidades generalizadas. Este algoritmo es análogo al problema de ensamblaje para posición y una versión de este mismo problema para velocidades.

Según este algoritmo, después de cada paso de integración los valores obtenidos para las coordenadas y velocidades generalizadas del sistema se modifican ligeramente para que satisfagan las ecuaciones cinemáticas.

En términos más matemáticos se dice que la solución obtenida se proyecta sobre el espacio de soluciones permitido. Este espacio es para las coordenadas generalizadas el que determinan las ecuaciones de enlace geométrico para coordenadas y para las velocidades generalizadas el determinado por las ecuaciones de enlace cinemático.

Así pues, se pueden resolver los problemas de corrección de posición y de

velocidad descritos anteriormente después de cada paso de integración, con lo que se consigue corregir la posición y la velocidad dentro del algoritmo de integración, de forma que se satisfagan las ecuaciones cinemáticas a lo largo de este. Esta corrección permite estabilizar el algoritmo de integración.

El algoritmo de integración numérica introduciendo las proyecciones (correcciones) ya citadas se detalla en la figura 2.2

$t \leftarrow 0$
Resolver $\mathbf{\Phi}^{Init}(\boldsymbol{q}) = 0$
Resolver $\dot{\mathbf{\Phi}}^{Init}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$
mientras $t < t_{final}$ **hacer**
  Resolver $\ddot{\mathbf{\Phi}}^{K.S.}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{0}$
  $\boldsymbol{q} \leftarrow \boldsymbol{q} + \dot{\boldsymbol{q}} \cdot \Delta t$
  $\dot{\boldsymbol{q}} \leftarrow \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}} \cdot \Delta t$
  Proyección de coordenadas
  Proyección de velocidad
  $t \leftarrow t + \Delta t$

Figura 2.3.: Integración numérica para cinemática con proyección

La pseudoinversa puede modificarse introduciendo una métrica para los errores en posiciones y velocidades. Esto confiere generalidad al planteamiento propuesto.

2.6. Simulación dinámica directa

en los problemas de simulación dinámica las aceleraciones generalizadas se obtienen mediante la solución del problema de aceleraciones incluyendo un conjunto de ecuaciones para las aceleraciones independientes. En la simulación dinámica éstas se obtienen del sistema de ecuaciones de la dinámica (Ecuaciones: 2.41) conocidos los valores de las coordenadas y velocidades generalizadas en un instante concreto de tiempo t.

El algoritmo de la figura 2.4 de integración numérica para la dinámica es

análogo al de la cinemática si bien el método para el cálculo de las aceleraciones es diferente.

Al igual que en el caso de la cinemática, después de realizar la integración, se ha de realizar una proyección de posiciones y velocidades ya que no se satisfacen las ecuaciones cinemáticas.

$t \leftarrow 0$
Resolver $\mathbf{\Phi}^{Init}(\boldsymbol{q}) = 0$
Resolver $\dot{\mathbf{\Phi}}^{Init}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$
mientras $t < t_{final}$ **hacer**
    Resolver $\mathbf{e}(\ddot{\boldsymbol{q}}, \boldsymbol{\lambda}) = 0$
    $\boldsymbol{q} \leftarrow \boldsymbol{q} + \dot{\boldsymbol{q}} \cdot \Delta t$
    $\dot{\boldsymbol{q}} \leftarrow \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}} \cdot \Delta t$
    Proyección de coordenadas
    Proyección de velocidad
    $t \leftarrow t + \Delta t$

Figura 2.4.: Integración numérica para dinámica

2.7. Cálculo de las matrices y vectores del problema DSM y exportación de los mismos

Una vez determinados simbólicamente la parametrización y las ecuaciones del sistema multicuerpo se han de obtener las matrices necesarias para la resolución de los diferentes problemas de la DSM. Como se ha comentado, sin pérdida de generalidad, en esta tesis se plantea una formulación basada el el PPV con multiplicadores de Lagrange.

Para obtener las matrices necesarias, como base de partida, se supone ya parametrizado el sistema por medio de $\boldsymbol{q}$,$\dot{\boldsymbol{q}}$ y $\ddot{\boldsymbol{q}}$. Se conocen los n_S sólidos y los w torsores. El conjunto w de torsores está formado por los torsores de gravedad y por los torsores asociados a fuerzas externas e internas del sistema. Se suponen ya conocidas las **e** ecuaciones dinámicas, obtenidas, por ejemplo,

mediante el PPV descrito en este capítulo.

De igual manera están determinas las ecuaciones geométricas $\mathbf{\Phi}$ y, en caso de haberlas, las ecuaciones no-holónomas $\dot{\mathbf{\Phi}}^{NH}$. También se conoce un conjunto de ecuaciones de posición $\mathbf{\Phi}^{Init}$ y velocidad iniciales $\dot{\mathbf{\Phi}}^{Init}$ para definir el estado inicial del sistema.

2.7.1. Matrices del problema cinemático

A partir de las restricciones geométricas $\mathbf{\Phi}$, tomando la derivada total respecto al tiempo se obtiene el vector simbólico de ecuaciones $\dot{\mathbf{\Phi}}$. A este conjunto de ecuaciones se le añade las ecuaciones no-holonomas para obtener el conjunto completo de ecuaciones de enlace cinemático, es decir:

$$\dot{\mathbf{\Phi}} = \begin{bmatrix} \dot{\mathbf{\Phi}} \\ \dot{\mathbf{\Phi}}^{NH} \end{bmatrix} \tag{2.55}$$

Nótese el abuso de notación al denotarlo también $\dot{\mathbf{\Phi}}$.

Este conjunto de ecuaciones se vuelve a derivar análogamente respecto del tiempo para obtener las ecuaciones cinemáticas para las aceleraciones generalizadas que se han denominado: $\ddot{\mathbf{\Phi}}$.

A continuación, se han de obtener los términos independientes de los problemas de posición y de velocidad, que según se vio en las ecuaciones 2.22 y 2.27 se han denominado $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ respectivamente.

El problema de velocidad (Ecuación 2.21) tiene la siguiente forma:

$$\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} - \boldsymbol{\beta} = \mathbf{0} \tag{2.56}$$

donde $\boldsymbol{\beta} = -\mathbf{\Phi}_{\mathbf{t}}(\boldsymbol{q}, t)$.

Este término independiente puede ser obtenido a partir de el conjunto de ecuaciones de velocidades $\dot{\mathbf{\Phi}}$, sustituyendo las velocidades generalizadas $\dot{\boldsymbol{q}}$ por 0 y multiplicado el resultado por -1.

La ecuación del problema de aceleraciones (2.23) tiene la siguiente forma:

$$\ddot{\mathbf{\Phi}} = \mathbf{\Phi}_q(\mathbf{q}, t)\ddot{\mathbf{q}} - \boldsymbol{\gamma} = \mathbf{0} \tag{2.57}$$

donde $\boldsymbol{\gamma} = -\dot{\mathbf{\Phi}}_q(\dot{\mathbf{q}}, \mathbf{q}, t)\dot{\mathbf{q}} - \dot{\mathbf{\Phi}}_t(\dot{\mathbf{q}}, \mathbf{q}, t)$.

Este término independiente puede ser obtenido a partir de el conjunto de ecuaciones de aceleraciones $\ddot{\mathbf{\Phi}}$, sustituyendo las aceleraciones generalizadas $\ddot{\mathbf{q}}$ por 0 y multiplicado el resultado por -1.

Las matrices jacobianas de posición $\mathbf{\Phi}_q$, velocidad $\dot{\mathbf{\Phi}}_{\dot{q}}$ y aceleración $\ddot{\mathbf{\Phi}}_{\ddot{q}}$ obtenerse según:

$$\mathbf{\Phi}_q = \frac{\partial \mathbf{\Phi}}{\partial \mathbf{q}} \quad , \quad \dot{\mathbf{\Phi}}_{\dot{q}} = \frac{\partial \dot{\mathbf{\Phi}}}{\partial \dot{\mathbf{q}}} \quad \text{y} \quad \ddot{\mathbf{\Phi}}_{\ddot{q}} = \frac{\partial \ddot{\mathbf{\Phi}}}{\partial \ddot{\mathbf{q}}} \tag{2.58}$$

Nótese que para un sistema holónomo, a nivel numérico las tres matrices jacobianas de posición, velocidad y aceleración son iguales. Es decir:

$$\mathbf{\Phi}_q = \dot{\mathbf{\Phi}}_{\dot{q}} = \ddot{\mathbf{\Phi}}_{\ddot{q}} \tag{2.59}$$

Pero en un problema no-holónomo únicamente las matrices jacobianas de velocidad y aceleración son iguales, es decir:

$$\mathbf{\Phi}_q \neq \dot{\mathbf{\Phi}}_{\dot{q}} = \ddot{\mathbf{\Phi}}_{\ddot{q}} \tag{2.60}$$

Para los cálculos cinemáticos, en el planteamiento definido en esta tesis, se va a usar siempre la matriz jacobiana de velocidades $\dot{\mathbf{\Phi}}_{\dot{q}}$.

2.7.1.1. Problema de posición y velocidad incial

Como se ha descrito en 2.5.2 para poder resolver los problemas de posición y velocidad iniciales se ha de completar el conjunto de ecuaciones geométricas de posición $\mathbf{\Phi}$ y velocidad $\dot{\mathbf{\Phi}}$ con el conjunto de ecuaciones iniciales de posiciones $\mathbf{\Phi}^{Init}$ y velocidades $\dot{\mathbf{\Phi}}^{Init}$)

Obviamente, por abuso de notación, al conjunto total de ecuaciones para resolver este problema se denomina también $\mathbf{\Phi}^{Init}$ para la posición inicial y $\dot{\mathbf{\Phi}}^{Init}$ para la velocidad inicial:

$$\mathbf{\Phi}^{Init} = \begin{bmatrix} \mathbf{\Phi} \\ \mathbf{\Phi}^{Init} \end{bmatrix} \quad \text{y} \quad \dot{\mathbf{\Phi}}^{Init} = \begin{bmatrix} \dot{\mathbf{\Phi}} \\ \dot{\mathbf{\Phi}}^{Init} \end{bmatrix} \tag{2.61}$$

Para resolver estas ecuaciones, se ha de obtener el término independiente del problema de velocidad $\boldsymbol{\beta}^{Init}$. Para ello se toma el conjunto de ecuaciones en velocidades $\dot{\mathbf{\Phi}}^{Init}$ y de sustituyen las velocidades generalizadas por 0 y el resultado se multiplica por -1.

Las matrices jacobianas de posición incial $\mathbf{\Phi_q}^{Init}$, velocidad inicial $\dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{Init}$ se obtienen mediante:

$$\mathbf{\Phi_q}^{Init} = \frac{\partial \mathbf{\Phi}^{Init}}{\partial \boldsymbol{q}} \quad \text{y} \quad \dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{Init} = \frac{\partial \dot{\mathbf{\Phi}}^{Init}}{\partial \dot{\boldsymbol{q}}} \tag{2.62}$$

2.7.2. Matrices del problema dinámico

Para la simulación dinámica (Sección 2.6) son necesarios la matriz de masa $\mathbf{M}$ y el vector de fuerzas generalizadas $\boldsymbol{\delta}$.

Al ser las ecuaciones dinámicas, $\mathbf{e}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\phi}, \boldsymbol{\tau})$, lineales en las aceleraciones, la matriz de masa $\mathbf{M}$ se obtiene como:

$$\mathbf{M}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{\partial \mathbf{e}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\phi}, \boldsymbol{\tau})}{\partial \ddot{\boldsymbol{q}}^{\mathsf{T}}} \tag{2.63}$$

El elevado coste computacional del cálculo simbólico de esta matriz hace que sea uno de los cuellos de botella del modelado simbólico.

El vector de fuerzas generalizadas $\boldsymbol{\delta}$ puede ser obtenido a partir de el conjunto de ecuaciones dinámicas, multiplicado por -1, y asignando a las aceleraciones generalizadas $\ddot{\boldsymbol{q}}$ un valor nulo. Simbólicamente, sustituyendo las aceleraciones generalizadas por cero.

2.7.3. Exportación de matrices para la resolución del problema DSM

Las matrices y vectores simbólicos necesarios para la solución de los distintos problemas de la DSM han ser exportados en forma de funciones que sean evaluadas por un código numérico.

Se ha de exportar las matrices y vectores siguientes:

- Problema de posición inicial: $\mathbf{\Phi}^{Init}$,$\mathbf{\Phi_q}^{Init}$,$\dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{Init}$ y $\boldsymbol{\beta}^{Init}$.
- Problema cinemático: $\mathbf{\Phi}$, $\dot{\mathbf{\Phi}}$, $\dot{\mathbf{\Phi}}_{\dot{q}}$, $\boldsymbol{\beta}$ y γ.
- Problema dinámico: $\mathbf{M}$, δ.
- Otros elementos como vector coordenadas $\boldsymbol{q}$, de velocidades $\dot{\boldsymbol{q}}$, de aceleraciones $\ddot{\boldsymbol{q}}$, de parámetros $\boldsymbol{\phi}$, multiplicadores de Lagrange $\boldsymbol{\lambda}$, etc...

2.8. Coordenadas sin inercia asociada

Según la naturaleza de las coordenadas puede darse el caso de que la matriz de masa de un sistema multicuerpo sea singular y que puede estar asociado a diversas causas.

Por ejemplo, a que se haya definido un número de coordenadas mayor que el estrictamente necesario. Esto puede ser debido, por ejemplo, a que el diseñador quisiese introducir una fuerza en función de una coordenada que no forme parte del modelo original.

Otro caso puede ser el de que el diseñador necesite el valor de una coordenada que, no formando parte del sistema multicuerpo, sea necesaria de cara al post-proceso de la información obtenida.

Por otro lado, están los problemas de contacto entre dos cuerpos, donde para determinar el punto de contacto se necesita un conjunto de coordenadas "extra" para definir las restricciones.

El estudio de este tipo de coordenadas ha sido tratado por diversos autores proponiendo diferentes tratamientos, así como diversas denominaciones para las coordenadas de este tipo. Así pues, en general, para problemas

donde se desea estudiar el contacto entre dos superficies, han sido denominadas "parámetros superficiales" (*surface parameters*) [34],[35],[36],[37] [38], [39] y [40], otros autores las han denominado "coordenadas no generalizadas" [34], "variables auxiliares" (*auxiliar variables*) [41] o "coordenadas mixtas" (*mixed coordinates*) [1].

La característica común de las citadas coordenadas es que no tienen una inercia asociada y con la intención de enfocar su tratamiento desde una perspectiva unificada, en este tesis, se ha decidido denominarlas "coordenadas sin inercia asociada".

La problematica introducida por las citadas coordenadas sin inercia asociada es la singularidad en la matriz de masa $\mathbf{M}$ debida a la aparición de filas y columnas nulas. Si bien esta matriz es singular, la matriz del sistema completo (2.41) no lo será.

2.8.1. Ejemplo de sistema con coordenadas sin inercia asociada

La figura 2.5 muestra la topología de la matriz de masa del ejemplo A.5, donde cada uno de los puntos representa un elemento no nulo. Toda la información dinámica del modelo se encuarta en la sub-matriz superior izquierda. El resto de sub-matrices están llenas de ceros, son los elementos asociado a las coordenadas sin inercia asociada.

El hecho de que toda la información de la dinámica solo aparezca en una sub-matriz, puede ser aprovechado para resolver el sistema dinámico de forma más eficiente. De igual manera, puede ser aprovechado para que la generación simbólica de la masa se realice también de forma más eficiente, como se verá en más adelante.

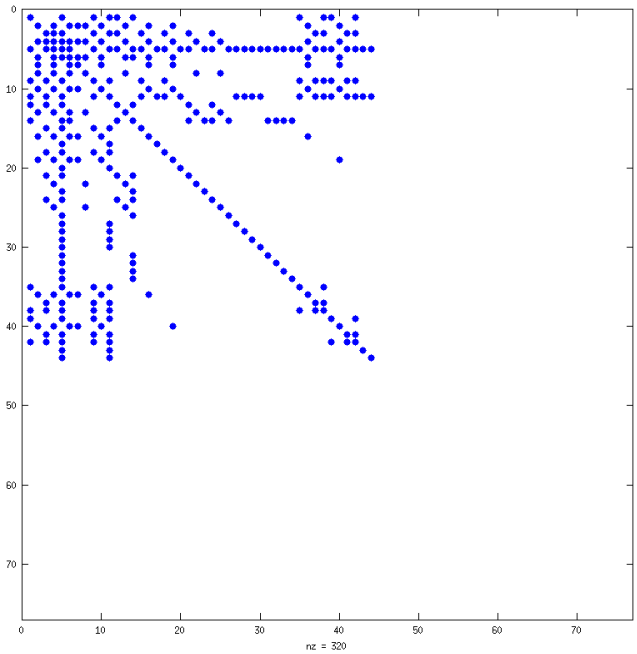


Figura 2.5.: Matriz de de masa del ejemplo A.5

CAPÍTULO 3

ATOMIZACIÓN. ALGORITMOS SIMBÓLICOS. EXPORTACIÓN

La eficiencia de los métodos simbólicos presentados en esta tesis depende principalmente de la metodología de atomización. Tanto el tamaño en memoria de la expresiones simbólicas como el tiempo de cálculo computacional están fuertemente ligados al proceso de atomización.

Por lo tanto, es necesaria una metodología capaz de obtener, de una forma eficiente, expresiones simbólicas atomizadas y trabajar con ellas para realizar algunas de las tareas necesarias (derivación y sustitución) del cálculo simbólico.

De igual manera, es necesaria un metodología que permita, a partir de expresiones atomizadas, generar y exportar código que será utilizado en la fase de análisis numérico.

En este capítulo se presenta un conjunto de métodos y algoritmos enfocados a la atomización y que permiten superar los problemas ligados a esta.

3.1. Proceso de atomización.

Como se ha visto en 1.1, se entiende por "atomización" la técnica que busca en un conjunto de expresiones simbólicas sub-expresiones que se repitan con el objetivo de sustituirlas por variables auxiliares de forma recursiva. Nótese que las sub-expresiones a su vez están sometidas a este proceso de atomización. Estas variables auxiliares definidas para las sub-expresiones se denominan "átomos".

El proceso de atomización presenta una serie de ventajas tanto en el cálculo simbólico como en el numérico.

En cálculo simbólico, permite disminuir el tamaño en memoria ocupado por las expresiones y permite acelerar (usando los algoritmos adecuados) las diferentes operaciones o manipulaciones simbólicas.

En cálculo numérico, reduce drásticamente el número de operaciones necesarias para evaluar numéricamente un conjunto de expresiones simbólicas. Además, al ser el tamaño ocupado en memoria por el conjunto de expresiones menor, la evaluación numérica de realiza de forma más rápida y la compilación de las expresiones se realiza también más rápido.

3.1.0.1. Atomización. Ejemplo

Sea el vector δ del ejemplo A.1 formado por dos expresiones simbólicas :

$$\delta = \begin{bmatrix} -F_b + \sin(-\theta)\ l_p\ m_p\ \dot{\theta}^2\ c_{vis}\ \dot{x} \\ M_p + (\theta - \theta_0)\ k - \sin(-\theta)\ l_p\ g\ m_p \end{bmatrix} \tag{3.1}$$

En el conjunto de expresiones anterior hay una sub-expresión común:

$\sin(-\theta)\ l_p\ m_p$,

por lo que esta puede ser sustituida, por ejemplo, por un conjunto de variables auxiliares, α_1 y α_2, que la contengan[1]:

[1]Nótese que existen otras opciones de atomización

$$\begin{aligned} \alpha_1 &= l_p\, m_p \\ \alpha_2 &= \sin(-\theta)\, \alpha_1 \\ \delta &= \begin{bmatrix} -F_b + \alpha_2\, \dot{\theta}^2\, c_{vis}\, \dot{x} \\ M_p + (\theta - \theta_0)\, k - \alpha_2\, g \end{bmatrix} \end{aligned} \tag{3.2}$$

A la hora de evaluar numéricamente el vector δ, se evita tener que repetir operaciones, únicamente es necesario calcular la sub-expresión $\sin(-\theta)\ l_p\ m_p$ una vez. Esto produce un ahorro total de cuatro operaciones algebraicas que no se han de realizar, tres productos y el cálculo de un seno. Además, simbólicamente la representación 3.1 es más breve que 3.2.

Al proceso contrario a la atomización, es decir, al de sustituir todos estos "átomos" por su expresión asociada en una expresión para obtenerla en forma expandida (o en extenso) se le denomina "desatomización".

En este documento se describen dos tipos de atomización, atomización "sobre la marcha" y atomización "completa"

3.1.1. Atomización "sobre la marcha"

En esta tesis se presenta un proceso de atomización "sobre la marcha" enfocado a ser de propósito general e independiente del formulismo[2].

La idea principal de la atomización "sobre la marcha" es que conforme se vaya construyendo simbólicamente el sistema, el resultado de cada operación simbólica sea atomizando.

La atomización se hace teniendo en cuenta hay dos tipos de operaciones:

Operaciones binarias Están formadas un operador y dos operandos, es decir, sumas, restas, productos o divisiones.

Funciones Están formadas por una función (en el campo DSM son principalmente funciones trigonométricas, exponenciales y de valor absoluto) y un argumento.

Las operaciones de tipo binario y funciones pueden darse entre variables

[2]Si bien este texto se centra en la DSM, el mismo procediendo de atomización es perfectamente válido en otros campos en lo que un cálculo simbólico sea necesario.

simbólicas, variables numéricas y átomos.

3.1.1.1. Ejemplo

A modo de ejemplo se analiza el suma de dos vectores, $\boldsymbol{u}$ y $\boldsymbol{v}$, cuyas componentes son conocidas respectivamente en las bases $\mathcal{B}_1$ y $\mathcal{B}_2$:

$$\{\boldsymbol{u}\}_{\mathcal{B}_1} = \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} \quad \text{y} \quad \{\boldsymbol{v}\}_{\mathcal{B}_2} = \begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix}_{\mathcal{B}_2} \tag{3.3}$$

Sea la matriz de cambio de base:

$$\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{3.4}$$

La suma de los vectores realizada en la base $\mathcal{B}_1$ mediante el proceso se atomización "sobre la marcha" se obtienen como se detalla a continuación:

$$\begin{aligned} \{\boldsymbol{u}+\boldsymbol{u}\}_{\mathcal{B}_1} = \{\boldsymbol{u}\}_{\mathcal{B}_1} + \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}\{\boldsymbol{v}\}_{\mathcal{B}_2} = \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix}_{\mathcal{B}_2} = \\ \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \alpha_1 & -\alpha_2 \\ 0 & \alpha_2 & \alpha_1 \end{bmatrix} \begin{Bmatrix} v_x \\ v_y \\ v_z \end{Bmatrix}_{\mathcal{B}_2} = \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} + \begin{Bmatrix} v_x \\ \alpha_1 v_y - \alpha_2 v_z \\ \alpha_2 v_y + \alpha_1 v_z \end{Bmatrix}_{\mathcal{B}_1} = \\ \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} + \begin{Bmatrix} v_x \\ \alpha_3 - \alpha_4 \\ \alpha_5 - \alpha_6 \end{Bmatrix}_{\mathcal{B}_1} = \begin{Bmatrix} u_x \\ u_y \\ u_z \end{Bmatrix}_{\mathcal{B}_1} + \begin{Bmatrix} v_x \\ \alpha_7 \\ \alpha_8 \end{Bmatrix}_{\mathcal{B}_1} = \begin{Bmatrix} v_x \\ \alpha_9 \\ \alpha_{10} \end{Bmatrix}_{\mathcal{B}_1} \end{aligned} \tag{3.5}$$

Donde α_i son las variables auxiliares o átomos:

$$\begin{aligned}
\alpha_1 &= \cos(\theta) & \alpha_2 &= \sin(\theta) \\
\alpha_3 &= \alpha_1 v_y & \alpha_4 &= \alpha_2 v_z \\
\alpha_5 &= \alpha_2 v_y & \alpha_6 &= \alpha_1 v_z \\
\alpha_7 &= \alpha_3 - \alpha_4 & \alpha_8 &= \alpha_5 - \alpha_6 \\
\alpha_9 &= u_y + \alpha_7 & \alpha_{10} &= u_z + \alpha_8
\end{aligned} \tag{3.6}$$

3.1.2. Atomización completa de expresiones

En caso de querer atomizar expresiones simbólicas completas, es decir no hayan sido obtenidas mediante un proceso continúo de operaciones binarias y funciones se ha de hacer un proceso atomización completo.

Este es el caso, por ejemplo, cuando una expresión es el resultado de un proceso de derivación (o de sustitución de símbolos) o una expresión simbólica cualquiera.

Nótese que el proceso de atomización "sobre la marcha" es un caso particular del proceso de atomización "completa" en donde la expresiones únicamente tienen un operador o una función. Así pues, en la sección 3.4 se presenta un algoritmo que realiza este proceso y que de igual manera es válido para el proceso de atomización "sobre la marcha".

3.1.3. Atomización en Robotran. Problemática

En [2] (Sección 5.5) se presenta la atomización utilizada por el paquete comercial Robotran [11] y en donde se le denomina"condensación". Este paquete dispone de un motor simbólico propio y también realiza un proceso propio de atomización "sobre la marcha".

El proceso de atomización se realiza como se describe a continuación:

1. Cada vez que se realiza una operación simbólica, el resultado es procesado (reorganizado y simplificado algebraica y trigonométricamente) y, a continuación, es almacenado en una variable auxiliar.
2. El conjunto de variables auxiliares es procesado de nuevo, donde las va-

riables auxiliares innecesarias o nulas son eliminadas.

3. Por último, este conjunto de variables auxiliares se reordena con la intención de que variables auxiliares que no dependan de otras puedan ser evaluadas numéricamente en paralelo por las unidades aritméticas del ordenador. La tecnología de compiladores actual parece que no es capaz de sacar ventaja de este hecho.

Pero esta metodología de atomización tiene una serie de problemas que hacen que no sea la mejor opción para el modelado simbólico de sistemas multicuerpo:

- Tiene un gran dependencia de la formulación en la que se esté usando, únicamente es óptima cuando se use una formulación de tipo recursivo.

  En formulaciones de tipo no recursivo esta metodología de atomización genera un gran número de variables auxiliares, donde muchas de ellas tienen asociada la misma sub-expresión simbólica, llegando a ocurrir que las funciones generadas sean incompatibles con los compiladores actuales.

  El autor indica que en caso de aparecer variables auxiliares que tengan asociada la misa expresión no se realiza ningún tratamiento especial. Ya que según indica, al usar una formulación de tipo recursivo, este hecho ocurre de forma marginal y no merece la pena ser tratado.

- Tiene un costo computacional añadido. Por cada nueva variable auxiliar que se crea, los elementos de su expresión asociada han de ser reordenados según unas reglas especiales citadas en la referencia anterior. Una vez reordenados, se ha de chequear la existencia de posibles simplificaciones de tipo algebraica y trigonométrico.

- Se ha de realizar un nuevo proceso de eliminación de las variables innecesarias y nulas, para evitar lo que denomina “explosión simbólica”.

3.2. Implementación de la atomización

Para la implementación de la técnica de atomización en un código simbólico es necesario disponer de las estructuras de datos que relacionen las distintas

expresiones simbólicas con sus correspondientes átomos asociados.

El proceso de atomización ha de ser bidireccional:

- A partir de una expresión simbólica, se ha de poder buscar si esta ya ha sido previamente atomizada y en caso afirmativo obtener dicho átomo. En caso negativo se ha de poder crear un nuevo átomo que represente a la expresión simbólica.

  Es decir: *Expresión* $\rightarrow$ *Átomo*.

- Dado un átomo se ha poder obtener la expresión asociada a él.

  Es decir: *Átomo*$\rightarrow$*Expresión*.

En esta sección se explican los tipos de estructuras que se han definido para realizar las dos tareas anteriores. Más adelante, en la sección 3.4 se explican los algoritmos implementados para el realizar el proceso de atomización de una forma eficiente.

3.2.1. Estado original de las funciones de atomización y desatomización

La situación de partida para este trabajo desarrollado en esta tesis en lo referente a la atomización es la situación presentada en [3].

En la citada referencia de presenta la librería simbólica lib_3D_MEC-GiNaC y en la que ya hay implementado un proceso de atomización.

3.2.1.1. Átomo

En esta librería existe definida una clase denominada *Atom*. Esta clase contiene un campo que permite almacenar una expresión simbólica asociada y, también, contiene un método que permite obtener dicha expresión simbólica.

Cada objeto (átomo) creado a partir de la clase representa a una variable simbólica y almacena la expresión simbólica asociada al mismo. Cada uno de los átomos es guardado en una tabla que se expande dinámicamente, denominada “tabla de átomos”.

3.2.1.2. Atomización por sustitución

Al estar este proceso de atomización pensado para ser independiente del tipo de formulación que se use, se tiene el problema de que aparecen átomos con la misma expresión asociada.

Para evitar este problema por cada expresión Exp que se desea atomizar el proceso que se ha de realizar es el siguiente:

1. Se recorre toda la tabla de átomos y se chequea si alguna de las expresiones asociadas Exp_i a los átomos Atm_i de esa tabla están presentes en la expresión original Exp.
2. En caso afirmativo se sustituye Exp_i por el átomo Atm_i en la expresión original Exp.
3. Una vez sustituidas todas las expresiones por sus correspondientes átomos en la expresión original, el resto de sub-expresiones Exp_{NoAtm} que están sin atomizar se asocian a un nuevo átomo Atm_{New}. Este nuevo átomo Atm_{New} se introduce en la tabla de átomos.
4. Estas sub-expresiones Exp_{NoAtm} se sustituyen en la expresión original Exp por el correspondiente átomo Atm_{New}.

El problema principal de esta técnica es que cada vez que se desea atomizar una expresión se tiene que recorrer toda la tabla de átomos.

Una búsqueda en una tabla de n elementos (átomos) de longitud tiene un tiempo de computación del orden de $\mathcal{O}(n)$ (tiempo de una búsqueda lineal). Si se tuvieran que atomizar m expresiones, el tiempo de computación sería del orden de $m \cdot \mathcal{O}(n)$. Además se ha de tener en cuenta que esta tabla de átomos crece durante el propio proceso de atomización, ya que se van creando nuevos átomos que se introducen en la misma, es decir, n no es constante. En el caso de que un sistema de ecuaciones a atomizar fuera "mediano", el tiempo de computación puede llegar a ser los suficientemente grande como para descartar esta técnica de atomización.

Es decir, se puede decir que se trata de un proceso de atomización mediante "fuerza bruta", en donde se prueban todas las posibilidades.

3.2.2. Atomización con búsqueda basada en tablas "hash"

Con la idea de mejorar el proceso de atomización existente se optó por usar una estructura de datos denominadas "tablas hash" en lugar del visto en el apartado anterior.

Una tabla hash (Capítulo 11 de [42]) es una estructura de datos que asocia claves con valores. La operación principal que soporta de manera eficiente es la búsqueda: permite el acceso al valor (átomo) almacenado a partir de una clave generada (usando la expresión simbólica). Funciona transformando la clave mediante una "función hash" en un número (hash) que identifica la posición donde la tabla hash localiza el valor deseado.

La figura 3.1 esquematiza la tabla hash T. Nótese que hay filas cuya clave está sin rellenar, esto es debido a cómo se implementan estas tablas que hace que no se llenen las filas forma consecutiva. Estas claves que todavía no han sido introducidas tienen asociado un valor 0.

Clave	Valor
Exp_1	Atm_1
Exp_2	Atm_2
Exp_3	Atm_3
-	0
Exp_4	Atm_4
-	0
-	0
Exp_5 -	Atm_5
-	0

Tabla 3.1.: Tabla hash T

Una tabla hash permite un acceso a los datos de manera muy rápida, generalmente con una complejidad computacional[3] de $\mathcal{O}(1)$, el borrado de datos y la inserción de un dato nuevo tienen el mismo orden.

Al ser una herramienta excelente para relacionar dos tipos de datos, pudiendo almacenar estas asociaciones, acceder a ellas e incluso borrarlas, se

[3]Nótese que en una tabla clásica de n elementos la complejidad computacional de la búsqueda es de $\mathcal{O}(n)$

decidió que este tipo tablas eran una buena herramienta para el proceso de atomización. Este hecho permitió que el tiempo de búsqueda e inserción de átomos pasara de tener una complejidad computacional de $\mathcal{O}(n)$ a $\mathcal{O}(1)$.

En el contexto de la atomización se realizan dos tareas, la inserción y la búsqueda de datos en la tabla hash.

Sea T la tabla anterior (Tabla 3.1) de tipo hash, entonces se pueden definir dos operaciones:

3.2.2.1. Inserción

El proceso de inserción consiste en almacenar en la tabla un átomo ($atom$) asociado a una expresión simbólica (exp).

$$T[exp] = atom$$

exp es la clave con la que se accede a la tabla y $atom$ es el valor. La función hash convierte la expresión en un índice único que indica la posición en la tabla T en la que almacena el átomo.

3.2.2.2. Búsqueda

La búsqueda consiste en la obtención del átomo ($atom$) asociado a una expresión simbólica (exp) que ya ha sido previamente introducida en la tabla.

$$atom = T[exp]$$

Nótese que exp es la clave con la que se accede a la tabla y que retorna el átomo asociado a dicha expresión. La función hash convierte la expresión en un índice único que indica la posición en la tabla T en la que almacena el átomo.

3.2.2.3. Proceso de atomización. Ejemplo

Sea la tabla de tipo hash 3.3, la cual ya contiene almacenados un cierto número de átomos del ejemplo anterior 3.1.1.1:

Expresión	Atomo
$\cos(\theta)$	α_1
$\sin(\theta)$	α_2
-	0
-	0
-	0
$\alpha_1 v_y$	α_3
-	0

Tabla 3.2.: Tabla hash T

Se desea atomizar la expresión $\alpha_2 v_z$:

1. En primer lugar se busca en la tabla hash si previamente ha sido atomizada. Al no estar en la tabla el resultado es nulo, es decir, $T[\alpha_2 v_z] = 0$. Para ello la función hash convierte la expresión $\alpha_2 v_z$ en un valor único que indica en qué posición (en este ejemplo la posición 4) debería estar introducido.
2. Al no estar atomizada, se crea un nuevo átomo, α_4 y se le asocia la expresión $\alpha_2 v_z$.
3. Se introduce en la tabla hash el nuevo átomo asociado la expresión, es decir $T[\alpha_2 v_z] = \alpha_4$. Para ello la función hash convierte la expresión $\alpha_2 v_z$ en un valor único que indica en qué posición (en este caso la posición 4) se ha de introducir.

La tabla contienen ahora un elemento más:

3.2.2.4. Implementación

La implementación de tablas de tipo hash y de los algoritmos de búsqueda y de inserción se ha de realizar teniendo en cuenta los problemas asociados a las mismas (implementación de la función de hash, tamaño de la tabla, colisio-

Expresión	Atomo
$\cos(\theta)$	α_1
$\sin(\theta)$	α_2
-	0
$\alpha_2 v_z$	α_4
-	0
$\alpha_1 v_y$	α_3
-	0

Tabla 3.3.: Tabla hash T

nes,...). No es el objetivo de esta tesis tratar con profundidad este tema.

En la citada referencia (Capítulo 11 de [42]) se puede encontrar una explicación con más detalle sobre la implementación de este tipo de tablas, sus problemas y cómo resolverlos.

3.3. Estructura de árbol de una expresión

Cabe recordar que en el contexto de la atomización un expresión está formada por un conjunto de constantes y variables (números, símbolos y átomos), además de funciones (seno, coseno, potencia, valor absoluto, etc...) y operadores (suma, producto, resta o división).

Se dice que el tamaño de una expresión es mayor que uno si esta contiene un operador o una función. El tamaño es igual a uno cuando contiene un valor numérico, uno simbólico o un átomo.

La expresión puede ser representada como una estructura de tipo árbol en donde cada una de las hojas (nodos terminales) son las variables y los nodos no terminales son los operadores o funciones.

En esta representación en forma de árbol, los nodos no terminales son de tipo n-anario, es decir, pueden tener desde un descendente (este es el caso de las funciones) o más de un descendiente.

Se denomina "sub-expresión" a una parte de la expresión. Las sub-expresiones son en sí mismas expresiones correctas. En esta representación en forma de

árbol, las sub-expresiones son el conjunto de variables, funciones y operadores descendientes de un nodo no terminal, incluido este mismo nodo.

Para representar en forma de árbol una expresión se define el orden de prioridad de los operadores[4] (Tabla: 3.4).

Operador	Símbolo	Prioridad
Suma/resta	+,-	1
Producto/División	*,/	2
Funciones	cos,sin,abs,pow...	3
Paréntesis	()	4

Tabla 3.4.: Prioridad de los operadores

3.3.0.5. Ejemplo: Estructura de árbol de una expresión

Sea la expresión $-F_b + \alpha_2\ \dot{\theta}^2\ c_{vis}\ \dot{x}$ del primer término del vector de la ecuación 3.2, donde $\alpha_1 = l_p\ m_p$ y $\alpha_2 = \sin(-\theta)\ \alpha_1$. En la figura 3.1 muestra su representación en forma de árbol.

Los nodos terminales son por ejemplo -1, c_{vis} y α_1. Los nodos no terminales son los operandos $+$ y $*$, así como las funciones $pow()$ y $sin()$.

Los nodos terminales α_1 y α_2 son de tipo átomo, así pues para acceder a todos los elementos de la expresión original se han de desatomimzar. Cada uno de ellos en sí es un árbol.

En este ejemplo $\alpha_2\ \dot{\theta}^2\ c_{vis}\ \dot{x}$ es lo que se denomina "sub-expresión".

3.3.1. Búsqueda en profundidad con pre-orden

En el contexto de la DSM una expresión se puede representar como un árbol que puede tener cualquier forma y número de elementos. Es decir, cada nodo no terminal puede tener hasta n nodos descendientes y por lo tanto, la topología del árbol es "a priori" desconocida.

[4]Este orden ha sido tomado de la librería GiNaC: Apartado "5.1.2 Accessing subexpressions" del manual de GiNaC.

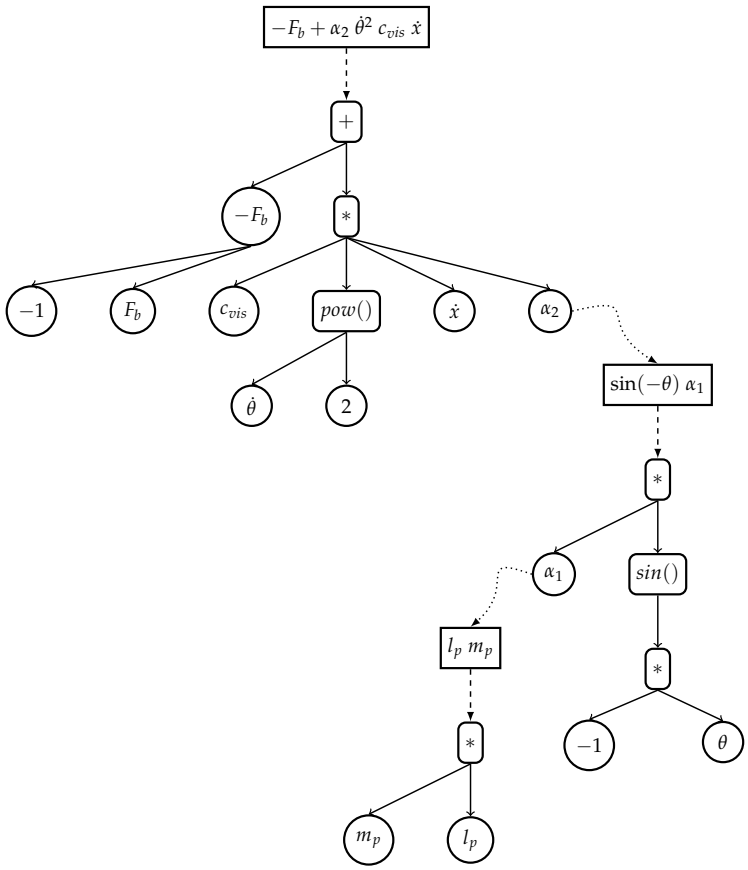


Figura 3.1.: Expresión expandida en forma de árbol

Representar una expresión en forma de árbol ayudará a entender de una forma más clara el algoritmo para recorrer la expresión presentado en este apartado.

Así pues, para acceder a todos los elementos de un árbol n-ario con topología desconocida en esta tesis se propone utilizar un algoritmo basado en el denominado "algoritmo de búsqueda en profundidad con pre-orden" (Sección 22.3 de [42]).

En la figura 3.2 se describe en pseudo-código el algoritmo que permite recorrer todos los nodos del árbol de manera ordenada. Se ha modificado para que

pueda recorrer expresiones con partes atomizadas.

Entrada: $ExpIn$: expresión simbólica

si *$ExpIn$ es de tipo átomo* **entonces**
  Obtener la expresión $AtmExp$ asociada al átomo
  recursive_tree($AtmExp$)
en otro caso
  si *Tamaño de $ExpIn > 1$* **entonces**
    para *Cada sub-expressión $ExpIN_i$ de $ExpIn$* **hacer**
      recursive_tree($ExpIN_i$)
  en otro caso
    Stop

Figura 3.2.: recursive_tree($ExpIN$)

Para recorrer un árbol no vacío con el algoritmo `recursive_tree` hay que realizar las siguientes operaciones:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$.
2. Si la expresión es de tipo átomo, se accede a la expresión $AtmExp$ asociada al mismo. Con esta expresión se llama al mismo algoritmo.
3. Si la expresión está formada por sub-expresiones, es decir, es un nodo no terminal, $ExpIn$ se divide en sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4. El mismo algoritmo accede a cada sub-expresión $ExpIn_i$. Nótese que cada sub-expresión $ExpIn_i$ es una expresión en sí.
4. Si la expresión no está formada por sub-expresiones está formada por un símbolo o un número, es decir, un nodo terminal, el algoritmo se para. El algoritmo regresa ("Backtracking") y repite el mismo proceso con cada una de las cada sub-expresión $ExpIn_i$ en que se ha dividido $ExpIn$.

En la figura 3.3 se muestra los pasos realizados para recorrer el árbol del ejemplo de la figura 3.1. Nótese que en los pasos $8 \rightarrow 9$ y $10 \rightarrow 11$ se produce un proceso de desatomización.

Basados en este algoritmo se han desarrollado otros algoritmos que realizan diferentes tareas con expresiones simbólicas. Se han desarrollado algoritmos recursivos de atomización y desatomización, un algoritmo de sustitución y un

algoritmo de diferenciación.

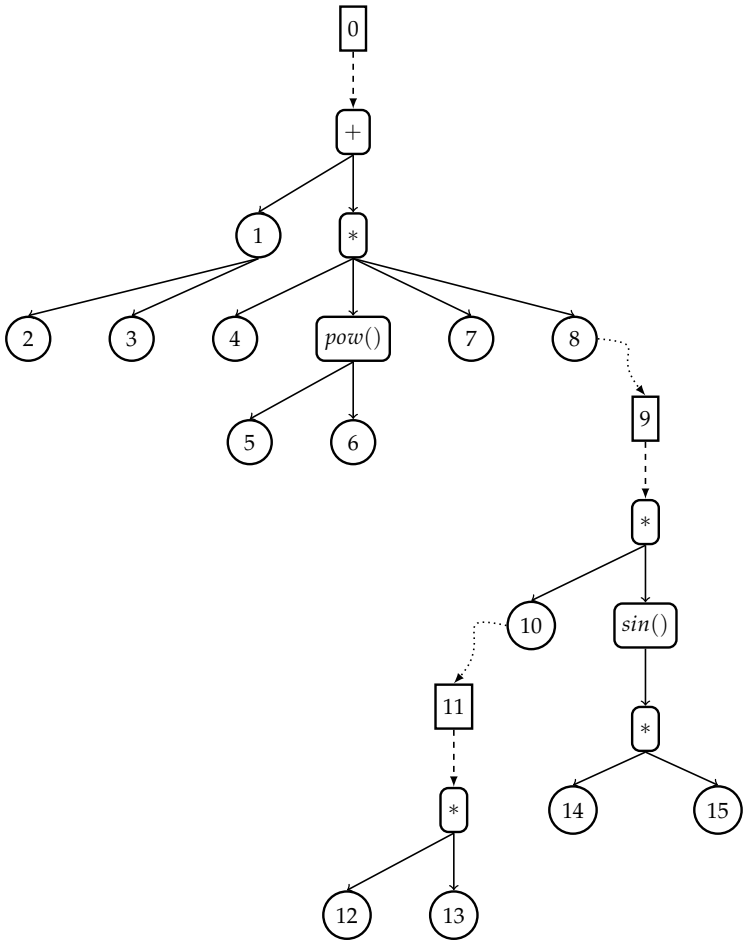


Figura 3.3.: Recorrido del árbol según el algoritmo recursivo

3.4. Atomización y desatomización. Algoritmos recursivos

En esta sección se presentan los algoritmos de atomización y del desatomización de expresiones simbólicas. Están basados en el algoritmo de la figura 3.2.

Para la búsqueda y creación de átomos se ha utilizado el método basado en tablas hash descrito en 3.2.2 ya que es el que ofrece mayor rapidez.

3.4.1. Algoritmo de atomización

En la figura 3.4 se describe en pseudo-código el algoritmo que se encarga de atomizar de forma recursiva la expresión de entrada $ExpIN$ y de retornar esta expresión atomizada $ExpOUT$.

La expresión $ExpIN$ puede estar sin atomizar, parcialmente atomizada o completamente atomizada. De igual manera, puede ser un único número, un único símbolo o un único átomo, en estos tres casos la expresión no se atomiza.

El algoritmo, para la búsqueda de átomos previamente generados, requiere de una tabla de tipo hash $AtmHashTable$ particular para cada sistema que se modele.

El algoritmo recursivo `atomize_ex` está dividido en dos partes. La primera parte recorre la expresión original buscando sub-expresiones que hayan sido previamente atomizadas y las sustituye por el correspondiente átomo. En la segunda parte crean nuevos átomos a partir de la sub-expresiones que no hubieran sido previamente atomizadas y se añaden a la tabla $AtmHashTable$.

3.4.1.1. Búsqueda de expresiones previamente atomizadas

La primera parte del algoritmo de la figura 3.4 realiza los siguientes pasos:

El algoritmo recursivo parte de la expresión original $ExpIn$.

1. Si la expresión es de tipo símbolo, número, átomo no se atomiza. Se copia en la expresión de salida $ExpOUT$ y se retorna este valor. Esta es una de

Entrada: $ExpIn$: expresión simbólica
Salida: $ExpOUT$: expresión completamente atomizada

si *($ExpIn$ es un símbolo, número o átomo)* **entonces**
  $ExpOUT \leftarrow ExpIn$
  devolver $ExpOUT$

si *($ExpIn$ existe en la tabla $AtmHashTable$)* **entonces**
  Obtener de $AtmHashTable$ el átomo Atm asociado a $ExpIn$
  $ExpOUT \leftarrow Atm$
  devolver $ExpOUT$
si *($-ExpIn$ existe en la tabla $AtmHashTable$)* **entonces**
  Obtener de $AtmHashTable$ el átomo Atm asociado a $-ExpIn$
  $ExpOUT \leftarrow -Atm$
  devolver $ExpOUT$
en otro caso
  $ExpAtm \leftarrow ExpIn$
  para *cada sub-expressión $ExpIN_i$ de $ExpIn$* **hacer**
    $ExpAtm_i \leftarrow$ atomize_ex($ExpIn_i$)
    Sustituir en $ExpAtm$ cada $ExpIn_i$ por su equivalente atomizado $ExpAtm_i$

si *($ExpAtm$ es un símbolo, número o átomo)* **entonces**
  $ExpOUT \leftarrow ExpAtm$
  devolver $ExpOUT$

si *($ExpAtm$ existe en la tabla $AtmHashTable$)* **entonces**
  Obtener de $AtmHashTable$ el átomo Atm asociado a $ExpAtm$
  $ExpOUT \leftarrow Atm$
si *($-ExpAtm$ existe en la tabla $AtmHashTable$)* **entonces**
  Obtener de $AtmHashTable$ el átomo Atm asociado a $-ExpAtm$
  $ExpOUT \leftarrow -Atm$
en otro caso
  Crear un nuevo átomo $AtmNew$
  Asocia a $AtmNew$ la expresión $ExpAtm$
  Introducir en la tabla $AtmHashTable$ el átomo $AtmNew$ y $ExpAtm$
  $ExpOUT \leftarrow AtmNew$
devolver $ExpOUT$

Figura 3.4.: atomize_ex($ExpIn$)

las condiciones de parada del algoritmo recursivo.

2. Si no se trata de uno de los tipo anteriores, se trata de una expresión compuesta por símbolos, números, átomos y/o funciones de los mismos.

   En este caso se chequea que la expresión de entrada $ExpIn$ no haya sido previamente atomizada buscándola en tabla de átomos $AtmHashTable$.

   a) En caso afirmativo, se copia en la expresión de salida $ExpOUT$ el átomo Atm obtenido de la tabla y se retorna este valor. Esta es otra de las condiciones de parada del algoritmo recursivo.

   b) Con la idea de optimizar el algoritmo, se chequea también si la misma expresión con signo negativo ha siso previamente atomizada.

   En caso afirmativo se copia en la expresión de salida $ExpOUT$ el átomo Atm multiplicado por -1 y se retorna este valor. Esta es otra de las condiciones de parada del algoritmo recursivo.

   c) En el caso de que la expresión de entrada $ExpIn$ no haya sido previamente atomizada lo que se tiene es una expresión compuesta por símbolos, números, átomos y/o funciones de los mismos, entonces:

   - Se copia en $ExpAtm$ la expresión $ExpIn$ y se divide la expresión en en sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4.
   - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$. El resultado $ExpAtm_i$ que devuelve el algoritmo recursivo corresponde a cada sub-expresión $ExpIn_i$ atomizada.
   - Se sustituye en $ExpAtm$ cada $ExpIn_i$ por su equivalente atomizado $ExpAtm_i$.

El resultado de la primera parte es $ExpAtm$, es decir la expresión original $ExpIn$ en donde las sub-expresiones que hubieran sido previamente atomizadas han sido sustituidas por el correspondiente átomo. Por lo tanto, $ExpAtm$ puede estar compuesto por símbolos, números, átomos y/o funciones de los mismos.

3.4.1.2. Creación de nuevos átomos

A continuación el algoritmo realiza los siguientes pasos:

1. Si la expresión $ExpAtm$ es de tipo símbolo, número, átomo no se atomiza. Se copia en la expresión de salida $ExpOUT$ y se retorna este valor. Esta es otra de las condiciones de parada del algoritmo recursivo.
2. Si no se trata de uno de los tipo anteriores, se trata de una expresión compuesta por símbolos, números, átomos y/o funciones de los mismos.

   En este caso se chequea que la expresión obtenida en la primera parte $ExpAtm$ no haya sido previamente atomizada buscándola en tabla hash de átomos $AtmHashTable$.

   a) En caso afirmativo, se copia en la expresión de salida $ExpOUT$ el átomo Atm obtenido de la tabla.

   b) Análogamente a como se ha procedido en la primera parte se chequea también si la misma expresión $ExpAtm$ con signo negativo ha siso previamente atomizada.

   En caso afirmativo se copia en la expresión de salida $ExpOUT$ el átomo Atm multiplicado por -1.

   c) En caso de que la expresión $ExpAtm$ no exista en la tabla de hash $AtmHashTable$:

   - Se crea un nuevo átomo $AtmNew$.
   - Se le asigna a $AtmNew$ la expresión $ExpAtm$.
   - Se introduce $AtmNew$ en la tabla de átomos.
   - Se copia en la expresión de salida $ExpOUT$ el átomo $AtmNew$.
3. Se retorna $ExpOUT$.

Este algoritmo es válido para un proceso de atomización de expresiones completas (Apartado 3.1.2) pero en este caso los átomos van a estar formados por operador que actúa sobre varios operandos. De igual manera, es válido tanto para la atomización para un proceso de atomización “sobre la marcha” (Apartado 3.1.1) donde todos lo átomos que se creen van a estar formados por una pareja de operandos y un operador o por una función y argumentos de la

misma.

3.4.2. Algoritmo de desatomización

En la figura 3.5 se describe en pseudo-código el algoritmo que se encarga de desatomizar de forma recursiva la expresión de entrada $ExpIN$ y de retornar la expresión $ExpOUT$ de forma extendida.

La expresión $ExpIN$ puede estar completamente atomizada, parcialmente atomizada o incluso sin atomizar.

Para el acceso a la expresión asociada a cada átomo se requiere de una estructura de datos como la presentada en 3.2 donde cada átomo dispone de un campo con dicha expresión asociada.

Entrada: *ExpIN*: expresión simbólica
Salida: *ExpOUT*: expresión extendida

si *ExpIn es de tipo átomo* **entonces**
    Obtener la expresión $AtmExp$ asociada al átomo
    $ExpUnatm \leftarrow$ unatomize_ex($AtmExp$)
    $ExpOUT \leftarrow ExpUnatm$
en otro caso
    si *tamaño de* $ExpIn > 1$ **entonces**
        $ExpUnatm \leftarrow ExpIn$
        para *cada sub-expresión* $ExpIn_i$ *de* $ExpIn$ **hacer**
            $ExpUnatm_i \leftarrow$ unatomize_ex($ExpIn_i$)
            Sustituir en $ExpUnatm$ cada $ExpIn_i$ por su equivalente desatomizado $ExpUnatm_i$
        $ExpOUT \leftarrow ExpUnatm$
    en otro caso
        $ExpOUT \leftarrow ExpIn$
devolver $ExpOUT$

Figura 3.5.: unatomize_ex($ExpIn$)

Para desatomizar una expresión el algoritmo recursivo `unatomice_ex` realiza los siguientes pasos:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$.
2. Si la expresión es de tipo átomo:
   - Se accede a la expresión $AtmExp$ asociada al mismo.
   - Con esta expresión se llama al mismo algoritmo. El valor que retorna el algoritmo $ExpUnatm$ se copia en $ExpOUT$.
3. Si la expresión no es de tipo átomo:
   a) Si la expresión está formada por sub-expresiones:
      - Se divide $ExpIn$ en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4. El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve el valor $ExpUnatm_i$
      - Se sustituye en $ExpUnatm$ cada $ExpIn_i$ por $ExpUnatm_i$, su equivalente desatomizado.
      - Se copia en $ExpOUT$ el valor devuelto $ExpUnatm_i$

   b) Si la expresión $ExpIn$ no está formada por sub-expresiones, es decir, es un símbolo o un número, se copia en la variable de salida $ExpOUT$
4. Se retorna el valor $ExpOUT$.

3.5. Algoritmo reursivo de sustitución de un símbolo

El hecho de que una expresión simbólica pueda estar parcial o completamente atomizada requiere la implementación de un algoritmo recursivo de sustitución de símbolos que considere la presencia de átomos.

El algoritmo que aquí se presenta accede a una expresión de entrada $ExpIn$ y sustituye en ella el símbolo Sym por el valor Val. Este mismo retorna la expresión $ExpOut$ obtenida completamente atomizada. Nótese que no es requerido un preproceso de desatomización, ni un post-proceso de atomización.

En la figura 3.6 se describe en pseudo-código el funcionamiento de dicho algoritmo recursivo de sustitución.

Entrada: $ExpIn$: expresión simbólica, Sym: símbolo, Val: valor numérico
Salida: $ExpOut$: expresión atomizada

si *$ExpIn$ es de tipo átomo Atm* **entonces**
  Obtener la expresión $AtmExp$ asociada al átomo Atm
  $ExpSubs \leftarrow$ recursive_expression_substitution ($AtmExp$, Sym, Val)
  si *$ExpSubs$ es igual a $AtmExp$* **entonces**
    $ExpOutUnatm \leftarrow Atm$
  en otro caso
    $ExpOutUnatm \leftarrow ExpSubs$
en otro caso
  si *tamaño de $ExpIn > 1$* **entonces**
    $ExpSubs \leftarrow ExpIn$
    para *cada sub-expressión $ExpIn_i$ de $ExpIn$* **hacer**
      $ExpSubs_i \leftarrow$recursive_expression_substitution ($ExpIn_i$, Sym, Val)
      Sustituir en $ExpSubs$ cada $ExpIn_i$ por $ExpSubs_i$
    $ExpOutUnatm \leftarrow ExpSubs$
  en otro caso
    si *$ExpIn$ es igual a Sym* **entonces**
      Sustituir en $ExpIn$ el símbolo Sym por el valor Val
    $ExpOutUnatm \leftarrow ExpIn$

$ExpOut \leftarrow$atomize_ex($ExpOutUnatm$)

devolver $ExpOut$

Figura 3.6.: recurvive_expression_substitution($ExpIn$, Sym, Val)

El algoritmo `recurvive_expression_substitution` realiza los siguientes pasos:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$.
2. Si la expresión es de tipo átomo, se accede a la expresión $AtmExp$ asociada al mismo Atm.

   Con esta expresión se llama al mismo algoritmo, que retorna la expresión $ExpSubs$. Pueden ocurrir dos casos:

   a) Si $ExpSubs$ es igual a $AtmExp$, (no se ha realizado ninguna sustitución) se copia en $ExpOUT$ el átomo Atm correspondiente a dicha

expresión.

b) En caso de que $ExpSubs$ y $AtmExp$ sean distintos (sí se ha realizado alguna sustitución), se copia en $ExpOutUnatm$ la expresión $ExpSubs$.

3. Si la expresión no es de tipo átomo:

   a) Si la expresión está formada por sub-expresiones, la expresión de entrada $ExpIn$ se copia en $ExpSubs$.

   - $ExpIn$ se divide en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4.
   - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve el valor $ExpSubs_i$.
   - Se sustituye en $ExpSubs$ cada $ExpIn_i$ por su equivalente devuelto por el algoritmo $ExpSubs_i$.
   - Se copia en $ExpOutUnatm$ el valor $ExpSubs$.

   b) Si la expresión $ExpIn$ no está formada por sub-expresiones, es decir, es un símbolo o un número:

   - Si Sym es igual a $ExpIn$ se sustituye el símbolo Sym por el valor Val.
   - Se copia en $ExpOutUnatm$ la expresión $ExpIn$.

4. Se atomiza la expresión $ExpOutUnatm$ y el resultado se copia en la variable de salida $ExpOUT$.
5. Se retorna el valor $ExpOUT$.

3.5.1. Algoritmo reursivo de sustitución de un conjunto de símbolos

El algoritmo anterior únicamente es válido cuando lo que se quiere sustituir sea un símbolo. Aun así, puede darse el caso de que se quiera realizar una sustitución de varios símbolos por un mismo valor en una expresión simbólica[5].

[5]En el contexto de la DSM simbólica es frecuente la sustitución de un conjunto de símbolos por, por ejemplo, el valor nulo. Véase, en la sección 2.7.1 como se calcula el vector β o γ

Utilizar algoritmo anterior para cada uno de los símbolos que se quiere sustituir tiene un alto costo computacional, se tiene que recorrer el mismo árbol de la expresión tantas veces como símbolos se quieran sustituir.

Entrada: $ExpIn$: expresión simbólica, $Vsym$: vector de símbolos, Val: valor numérico
Salida: $ExpOut$: expresión atomizada

si $ExpIn$ *es de tipo átomo* Atm **entonces**
  Obtener la expresión $AtmExp$ asociada al átomo Atm
  $ExpSubs \leftarrow$ recursive_expression_substitution ($AtmExp$, $Vsym$, Val)
  si $ExpSubs$ *es igual a* $AtmExp$ **entonces**
    $ExpOut \leftarrow Atm$
  en otro caso
    $ExpOutUnatm \leftarrow ExpSubs$
en otro caso
  si *tamaño de* $ExpIn > 1$ **entonces**
    $ExpSubs \leftarrow ExpIn$
    para *cada sub-expresión* $ExpIn_i$ *de* $ExpIn$ **hacer**
      $ExpSubs_i \leftarrow$recursive_expression_substitution ($ExpIn_i$, $Vsym$, Val)
      Sustituir en $ExpSubs$ cada $ExpIn_i$ por $ExpSubs_i$
    $ExpOutUnatm \leftarrow ExpSubs$
  en otro caso
    para *Cada símbolo* Sym_i *del vector de símbolos* $Vsym$ **hacer**
      si Sym_i *es igual a* $ExpIn$ **entonces**
        Sustituir en $ExpIn$ el símbolo Sym_i por el valor Val
    $ExpOutUnatm \leftarrow ExpIn$

$ExpOut \leftarrow$atomize_ex($ExpOutUnatm$)

devolver $ExpOut$

Figura 3.7.: recurvive_expression_substitution($ExpIn$, $Vsym$, Val)

En la figura 3.7 se describe en pseudo-código el algoritmo que permite realizar esta tarea evitando recorrer la misma expresión numerosas veces. El algoritmo accede a una expresión de entrada $ExpIn$ y sustituye en ella el conjunto de símbolos $Vsym$ por el valor Val. Este mismo retorna la expresión obtenida completamente atomizada. La expresión de entrada $ExpIn$ puede estar par-

cial, completamente atomizada o sin atomizar.

El algoritmo `recurvive_expression_substitution` realiza los siguientes pasos:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$.
2. Si la expresión es de tipo átomo, se accede a la expresión $AtmExp$ asociada al mismo Atm.

   Con esta expresión se llama al mismo algoritmo, que retorna la expresión $ExpSubs$. Pueden ocurrir dos casos:

   a) Si $ExpSubs$ es igual a $AtmExp$, (no se ha realizado ninguna sustitución) se copia en $ExpOUT$ el átomo Atm correspondiente a dicha expresión.

   b) En caso de que $ExpSubs$ y $AtmExp$ sean distintos (sí se ha realizado alguna sustitución), se copia en $ExpOutUnatm$ la expresión $ExpSubs$.
3. Si la expresión no es de tipo átomo:

   a) Si la expresión está formada por sub-expresiones, se copia la expresión de entrada $ExpIn$ en $ExpSubs$.

   - $ExpIn$ se divide en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4.
   - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve el valor $ExpSubs_i$.
   - Se sustituye en $ExpSubs$ cada $ExpIn_i$ por su equivalente devuelto por el algoritmo $ExpSubs_i$.
   - Se copia en $ExpOutUnatm$ el valor $ExpSubs$.

   b) Si la expresión $ExpIn$ no está formada por sub-expresiones, es decir, es un símbolo o un número:

   - Por cada símbolo Sym_i del vector de símbolos $Vsym$:
     - Si Sym_i es igual a $ExpIn$ se sustituye el símbolo Sym_i por el valor Val.
   - Se copia en $ExpOutUnatm$ la expresión $ExpIn$.

4. Se atomiza la expresión $ExpOutUnatm$ y el resultado se copia en la variable de salida $ExpOUT$.
5. Se retorna el valor $ExpOUT$.

3.6. Algoritmo reursivo de derivación

El hecho de que una expresión simbólica pueda estar parcial o completamente atomizada requiere la implementación de un algoritmo recursivo de derivación que considere la presencia de átomos.

El algoritmo accede a una expresión de entrada $ExpIn$ y la deriva respecto al símbolo Sym. Este mismo algoritmo retorna la expresión obtenida completamente atomizada.

En la figura 3.8 se describe en pseudo-código el funcionamiento de dicho algoritmo recursivo de derivación.

El algoritmo `recurvive_differentiation` realiza los siguientes pasos:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$.
2. Si la expresión es de tipo átomo:
   - Se accede a la expresión $AtmExp$ asociada al mismo Atm.
   - Con la expresión $AtmExp$ se llama al mismo algoritmo, que retorna la expresión $ExpDiff$.
   - Se copia en $ExpOutUnatm$ la expresión $ExpDiff$. Pueden darse dos casos:
     a) Si $ExpDiff$ es igual a $AtmExp$, se copia en $ExpOUT$ el átomo Atm correspondiente a dicha expresión.
     b) En caso de que $ExpDiff$ y $AtmExp$ sean distintos (sí se ha realizado alguna sustitución), se copia en $ExpOutUnatm$ dicha expresión $ExpDiff$.
3. Si la expresión no es de tipo átomo:
   a) Si la expresión está formada por sub-expresiones se ha de diferenciar tres casos, que sea una suma, un producto o una función de

Entrada: $ExpIn$: expresión simbólica, Sym: Símbolo
Salida: $ExpOut$: expresión atomizada

si *$ExpIn$ es de tipo átomo Atm* **entonces**
  Obtener la expresión $AtmExp$ asociada al átomo Atm
  $ExpDiff \leftarrow$ recursive_differentiation ($AtmExp$, Sym)
  si *$ExpDiff$ es igual a $AtmExp$* **entonces**
    $ExpOut \leftarrow Atm$
  en otro caso
    $ExpOutUnatm \leftarrow ExpDiff$
en otro caso
  si *tamaño de $ExpIn > 1$* **entonces**
    si *($ExpIn$ es una suma)* **entonces**
      $ExpDiff \leftarrow 0$
      para *cada sub-expressión $ExpIn_i$ de la expresión $ExpIn$* **hacer**
        $ExpDiff_i \leftarrow$recursive_differentiation ($ExpIn_i$, Sym)
        $ExpDiff = ExpDiff + ExpDiff_i$
      $ExpOutUnatm \leftarrow ExpDiff$
    si *($ExpIn$ es un producto)* **entonces**
      $ExpDiff \leftarrow 0$
      $ExpInCopy \leftarrow ExpIn$
      para *cada sub-expressión $ExpIn_i$ de la expresión $ExpIn$* **hacer**
        $ExpDiff_i \leftarrow$recursive_differentiation ($ExpIn_i$, Sym)
        Sustituir en $ExpInCopy$ la sub-expresión $ExpIn_i$ por $ExpDiff_i$
        $ExpDiff = ExpDiff + ExpInCopy$
      $ExpOutUnatm \leftarrow ExpDiff$
    si *($ExpIn$ es una función)* **entonces**
      $ExpDiffUnatm \leftarrow$ unatomize_ex($ExpIn$)
      $ExpDiff \leftarrow \frac{\mathrm{d}}{\mathrm{d}\,Sym} ExpDiffUnatm$
      $ExpOutUnatm \leftarrow ExpDiff$
  en otro caso
    si *($ExpIn$ es un número)* **entonces**
      $ExpDiff \leftarrow 0$
    en otro caso
      $ExpDiff \leftarrow \frac{\mathrm{d}}{\mathrm{d}\,Sym} ExpIn$
    $ExpOutUnatm \leftarrow ExpDiff$

$ExpOut \leftarrow$atomize_ex($ExpOutUnatm$)

devolver $ExpOut$

Figura 3.8.: recurvive_differentiation($ExpIn$, Sym)

sub-expresiones. Según cada caso de aplican las pertinentes reglas de derivación:

1) Si $ExpIn$ es una suma:
   - Se le asigna a $ExpDiff$ en valor 0.
   - $ExpIn$ se divide en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4. Por cada sub-expresión:
     - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve el valor $ExpDiff_i$. Es decir, la derivada de la sub-expresión $ExpIn_i$ respecto al símbolo Sym.
     - Se suma a $ExpDiff$ el valor $ExpDiff_i$ y se guarda en la misma variable $ExpDiff$.
   - Se copia en $ExpOutUnatm$ el valor $ExpDiff$.
2) Si $ExpIn$ es un producto:
   - Se le asigna a $ExpDiff$ en valor 0.
   - Se copia en $ExpInCopy$ la expresión $ExpIn$
   - $ExpIn$ se divide en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4. Por cada sub-expresión:
     - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve el valor $ExpDiff_i$. Es decir, la derivada de la sub-expresión $ExpIn_i$ respecto al símbolo Sym.
     - Sustituir en la expresión $ExpInCopy$ la sub-expresión devuelta por el algoritmo $ExpIn_i$ por $ExpDiff_i$.
     - Se suma a $ExpDiff$ el valor $ExpInCopy$ y se guarda en la misma variable $ExpDiff$.
   - Se copia en $ExpOutUnatm$ el valor $ExpDiff$.
3) Si $ExpIn$ es una función:
   - Se desatomiza la función $ExpIn$ y se asigna a la variable $ExpDiffUnatm$
   - Se deriva $ExpDiffUnatm$ respecto al símbolo Sym y el resultado se asigna a la variable $ExpDiff$

- Se copia en $ExpOutUnatm$ el valor $ExpDiff$.

b) Si la expresión $ExpIn$ no está formada por sub-expresiones, es decir, es un símbolo o un número:

- Si $ExpIn$ es un número se le asigna a la variable $ExpDiff$ el valor 0.
- Si $ExpIn$ es un símbolo se deriva $ExpIn$ respecto al símbolo Sym y el resultado se asigna a la variable $ExpDiff$
- Se copia en $ExpOutUnatm$ la expresión $ExpDiff$.

4. Se atomiza la expresión $ExpOutUnatm$ y el resultado se copia en la variable de salida $ExpOUT$
5. Se retorna el valor $ExpOUT$.

3.7. Exportación simbólica basada en la atomización

En el contexto de la DSM los códigos simbólicos generalmente están orientados hacia el cálculo de las ecuaciones cinemáticas y dinámicas, en forma de matrices y vectores necesarios para la solución de los distintos problemas del sistema.

La solución generalmente suele hacerse en un código numérico que evalúa los distintos vectores y matrices generados por el código simbólico y que realiza los cálculos que se necesiten. Por eso es conveniente que un código simbólico permita generar y exportar de todos los elementos necesarios para su evaluación en otro código de tipo numérico.

Si se utiliza la técnica de atomización presentada en 3.1, las distintas expresiones a exportar ya contienen las expresiones atomizadas, por lo tanto es necesario disponer de un método enfocado a generar código optimizado en base a dicha atomización. Como se verá más adelante trabajar con expresiones atomizadas beneficia enormemente al proceso de exportación.

En esta sección se propone un conjunto de algoritmos enfocados a procesar y optimizar los elementos a exportar. Las técnicas de exportación desarrolladas en esta tesis son generales, es decir, independientes del tipo de formulación y

parametrización.

En esta tesis únicamente se aborda el problema de la generación de los elementos que se ha de exportar. La generación de los distintos ficheros, depende exclusivamente de a qué lenguaje se quiera exportar, ya que cada uno tienen sus peculiaridades. Este tema en esta tesis no se van a tratar.

3.7.1. Exportación de una expresión

Para generar una función que devuelva la expresión evaluada numéricamente, se exporta la expresión sin desatomizar. Para evaluar numéricamente dicha expresión, previamente es necesario evaluar los átomos que en ella aparecen a través de sus expresiones asociadas. Sí, además, la expresión asociada a un átomo contiene otros átomos, estos ha de ser evaluados antes de evaluar dicha expresión asociada. Por lo tanto, cada expresión exportada ha de estar acompaña de todos los átomos que recursivamente contiene.

En la función que evalúa numéricamente la expresión primero se deben evaluar los átomos y a continuación la propia expresión. El orden en que se evalúan los átomos es importante ya que, en general, puede haber dependencia entre ellos. Es decir, se han de evaluar de tal forma que si la expresión asociada a un átomo contiene otros átomos, estos hayan sido ya previamente evaluados.

Por lo tanto antes de exportar un expresión, es necesario un algoritmo que busque todos los átomos de los que depende dicha expresión y los guarde de forma ordenada de tal manera que no haya dependencia entre ellos a la hora de la evaluación.

3.7.2. Búsqueda de átomos de una expresión atomizada

El algoritmo de la figura 3.9, recorre una expresión atomizada buscando los átomos de los que depende e introduciéndolos en una lista. Está basado en el algoritmo recursivo de "búsqueda en profundidad con pre-orden" en árboles (Figura 3.2).

Puede darse el caso de que en una misma expresión haya átomos que aparezcan varias veces (en distintas ramas del árbol). Para evitar que el algoritmo

introduzca estos átomos varias veces en la lista se usa una tabla de tipo hash que confirma si un átomo ya ha sido introducido o no en la lista. En esta tabla la "clave" es el átomo y el "valor" es uno o cero, siendo uno cuando se ha introducido en la tabla o cero cuando no. De esta forma el tiempo de búsqueda e inserción en la tabla es de $\mathcal{O}(1)$, frente a $\mathcal{O}(n)$, que es el tiempo de búsqueda en una lista convencional.

El algoritmo de la figura 3.9 toma la expresión de entrada $ExpIn$, una lista de átomos $ListIn$ y una tabla hash $AtomTableIn$. La expresión $ExpIN$ puede estar sin atomizar, parcialmente atomizada o completamente atomizada. Así mismo, retorna un lista de átomos $ListOut$ donde están los átomos de los cuales depende la expresión $ExpIn$ y una tabla hash $AtomTableOut$ donde se indican qué átomos están en la lista $ListOut$.

Entrada: $ExpIn$: expresión atomizada, $ListIn$: lista de átomos, $AtomTableIn$: Tabla hash
Salida: $ListOut$:lista de los átomos, $AtomTableOut$: Tabla hash
si *($ExpIn$ es un átomo Atm)* **entonces**
  si *Atm **no** está en la tabla $AtomTableIn$* **entonces**
    Introducir el átomo Atm en la lista $ListIn$
    Introducir el átomo Atm en la tabla $AtomTableIn$
  Obtener la expresión $AtmExp$ asociada al átomo Atm
  [$ListOut$,$AtomTableOut$] ← atoms_in_exp($AtmExp$, $ListIn$, $AtomTableIn$)
en otro caso
  si *Tamaño de $ExpIn > 1$* **entonces**
    para *cada sub-expressión $ExpIn_i$* **hacer**
      [$ListOut$,$AtomTableOut$] ← atoms_in_exp($AtmExp_i$, $ListIn$, $AtomTableIn$)
devolver $ListOut$,$AtomTableOut$

Figura 3.9.: atoms_in_exp(ExpIn, ListIn, AtomTableIn)

Para buscar los átomos de los que depende una expresión el algoritmo recursivo `atoms_in_exp` realiza los siguientes pasos:

1. El algoritmo recursivo parte de la expresión de entrada $ExpIn$

2. Si la expresión $ExpIn$ es un átomo:
   a) Si el átomo Atm no está en la tabla $AtomTableIn$:
      - Se introduce el átomo Atm en la lista $ListIn$
      - Se introduce el átomo Atm en la tabla $AtomTableIn$
   b) Se obtienen la expresión $AtmExp$ asociada al átomo Atm y se llama al mismo algoritmo, con la expresión $AtmExp$. La tabla y la lista con las que se llama al algoritmo son respectivamente $AtomTableIn$ y $ListIn$. El algoritmo retorna una lista de átomos donde están los átomos de los cuales depende $AtmExp$ y que se asigna a $ListOut$. También retorna la tabla hash donde se indican qué átomos están en la lista $ListOut$ y que se asigna a $AtomTableOut$.
3. Si la expresión $ExpIn$ no es un átomo y el tamaño es mayor que uno, es decir, contiene sub-expresiones se realizan los siguientes dos pasos:
   - Se divide $ExpIn$ en las sub-expresiones $ExpIn_i$ según las reglas establecidas en 3.4.
   - El mismo algoritmo accede a cada sub-expresión $ExpIn_i$ y devuelve la lista de átomos que se asigna a $ListOut$ y la tabla hash que se asigna a $AtomTableOut$.

La condición de parada de este algoritmo recursivo se da cuando la expresión de entrada es un símbolo o número, es decir, cuando no se cumple ninguno de los casos presentes en el algoritmo.

3.7.3. Exportación de matrices y vectores

Cuando se tienen matrices o vectores donde cada componente es un expresión simbólica atomizada se ha de buscar para cada componente los átomos de los que depende.

El algoritmo de la figura 3.10, toma una matriz T de tamaño n x m y accede a todos las componentes $ExpIn_{ij}$. Nótese que un vector es un caso particular de una matriz, con una única fila o columna. Por cada componente busca los átomos de los que depende y los introduce en la lista $AtmList$. El algoritmo retorna la lista $AtmList$, la cual contiene todos los átomos de los que depende

la matriz T sin que ninguno esté repetido.

Este algoritmo utiliza el algoritmo de búsqueda de átomos en expresiones `atoms_in_exp` (Figura 3.9).

Entrada: *Mat*: matriz $m \cdot n$ de expresiones simbólicas
Salida: *AtmList*: Lista de átomos de los que depende la matriz
Definir una lista *AtmList* vacía
Definir una tabla de hash *AtomHashTable* vacía

para $i = 0$ ***to*** n **hacer**
  para $j = 0$ ***to*** m **hacer**
    $ExpIn_{ij} \leftarrow Mat(i,j)$
    $[AtmListOut, AtomTableOut] \leftarrow$ atoms_in_exp($ExpIn_{ij}$, *AtmList*, *AtomHashTable*)
    $AtmList \leftarrow AtmListOut$
    $AtomHashTable \leftarrow AtmListOut$

devolver *AtmList*

Figura 3.10.: atom_list(Mat)

Nótese que los argumentos de entrada del algoritmo `atoms_in_exp` *AtmList* y *AtomHashTable* son copias de la lista *AtmListOut* y de la tabla *AtomTableOut* obtenidas en la iteración anterior. De esta manera, por medio de la tabla de hash, se asegura que no se introducen en la lista átomos que ya han sido previamente introducidos. Es decir, *AtmListOut* y *AtomTableOut* contienen todos los átomos, sin repetición, aparecidos en las expresiones de las iteraciones anteriores.

3.7.3.1. Ejemplo

Sea el vector δ de tamaño 2x1 correspondiente al vector de fuerzas generalizadas del mecanismo del ejemplo A.1. En este caso, el vector ha sido generado mediante un proceso de atomización sobre la marcha:

$$\delta = \begin{bmatrix} -\alpha_{58} \\ \alpha_{60} \end{bmatrix} \tag{3.7}$$

Donde:

$$\begin{aligned}
\alpha_{10} &= -l_p\, m_p & \alpha_{12} &= \sin(-\theta) \\
\alpha_{13} &= \alpha_{10}\, \alpha_{12} & \alpha_{38} &= \alpha_{13}\, \dot{\theta} \\
\alpha_{39} &= \alpha_{38}\, \dot{\theta} & \alpha_{6} &= -c_{vis}\, \dot{x} \\
\alpha_{57} &= \alpha_{39} + \alpha_{6} & \alpha_{58} &= \alpha_{57} + F_b \\
\alpha_{4} &= \theta_0 - \theta & \alpha_{5} &= -k\, \alpha_4 \\
\alpha_{15} &= \alpha_{13}\, g & \alpha_{59} &= \alpha_5 + \alpha_{15} \\
\alpha_{60} &= M_p + \alpha_{59}
\end{aligned} \tag{3.8}$$

Que representado en extenso:

$$\delta = \begin{bmatrix} -F_b + \sin(-\theta)\, l_p\, m_p\, \dot{\theta}^2 + c_{vis}\, \dot{x} \\ M_p - (\theta_0 - \theta)\, k - \sin(-\theta)\, l_p\, g\, m_p \end{bmatrix} \tag{3.9}$$

El código exportado en lenguaje C es:

```
atom10 = -l_p*m_p;
atom12 = sin(-theta);
atom13 = atom10*atom12;
atom38 = atom13*dtheta;
atom39 = atom38*dtheta;
atom6 = -C_vis*dx;
atom57 = atom39+atom6;
atom58 =  atom57+F_b;
atom4 =  theta_0-theta;
atom5 = -k*atom4;
atom15 = atom13*g;
atom59 = atom5+atom15;
atom60 = M_p+atom59;

delta[0] = -atom58;
delta[1] = atom60;
```

Listado 3.1: Ejemplo de exportación en C

Véase en el código el orden en que los átomos han sido exportados. Por ejemplo, los átomos `atom10` y `atom12` han se ser evaluados antes que el átomo `atom13`, ya que este depende de los dos anteriores.

De igual manera, véase que el átomo `atom13`, es usado dos veces, en los átomos `atom38` y `atom15` lo que permite reciclar el cálculo `-l_p*m_p*sin(-theta)` asociado a dicho átomo `atom13`.

3.7.4. Optimización en la exportación

El método de exportación anterior tiene el inconveniente de que cada átomo exportado contiene únicamente una función o dos elementos y una operación entre ellos. Esto es debido al proceso de atomización sobre la marcha. Este hecho hace que el listado de átomos generado por el algoritmo `atom_list` tenga un tamaño en átomos más grande de lo necesario.

Si se observa el listado anterior 3.7.3.1, hay átomos que únicamente aparecen una vez. Por ejemplo, los átomos `atom10`, `atom6`, `atom59`, etc...

El proceso de exportación se puede optimizar eliminando del listado a exportar los átomos que únicamente aparecen una vez y sustituyéndolos ahí donde aparecen por su expresión correspondiente.

En el ejemplo anterior, el átomo `atom10` se podría eliminar y sustituir por su expresión `-l_p*m_p` en el átomo `atom13` y ser eliminado, haciendo así que el tamaño del listado de átomos sea menor.

En el listado del ejemplo anterior hay únicamente un átomo, `atom13`, que está más de una vez. Todos los demás átomos sólo aparecen una vez y, por lo tanto, pueden ser sustituidos y eliminados del listado. Quedando el listado del ejemplo de la siguiente manera:

```
atom13 = -l_p*sin(-theta)*m_p;

_delta[0] = -(dtheta*dtheta)*atom13-F_b+c_vis*dx;
_delta[1] =  M_p+g*atom13+k*( theta-theta_0);
```

Listado 3.2: Ejemplo de exportación en C optimizada

Con la idea de realizar esta tarea, se proponen dos algoritmos que para ser más eficaces usa tablas de tipo hash.

El algoritmo de la figura 3.11 es el algoritmo principal y que, a su vez, utiliza el algoritmo `opt_unatomize_ex`, se ve en más detalle en el apartado siguiente.

El algoritmo `opt_atom_list` toma como entrada una matriz (o vector) y retorna esta matriz "optimizada" o "parcialmente desatomizada". Se entiende por matriz "optimizada" una en la que los átomos que únicamente aparecen una vez han sido sustituidos por su expresión asociada. Así para el ejemplo anterior el vector δ optimizado (parcialmente desatomizado) toma la forma:

$$\delta = \begin{bmatrix} -F_b + \alpha_{13}\dot{\theta}^2 + c_{vis}\ \dot{x} \\ M_p - (\theta_0 - \theta)\ k - g\ \alpha_{13} \end{bmatrix} \tag{3.10}$$

Este algoritmo realiza la tarea de optimización en cinco pasos:

0. Definir una lista general *AtmLstExp* y definir una tabla de tipo hash *RepAtmHashTableOut* donde la clave es el átomo y el valor el número de veces que aparece.
1. Obtener la lista de átomos *AtmLstTotal* correspondiente al elemento que se quiera exportar *MatIn*.

   Esto se realiza mediante el algoritmo `atom_list`.

   ★En el ejemplo anterior esta lista es: `atom10`, `atom12`, `atom13`, `atom38`, `atom39`, `atom6`, `atom57`, `atom58`, `atom4`, `atom5`, `atom15`, `atom59` ,`atom60`
2. Buscar y contar los átomos que aparecen en las expresiones asociadas a los átomos de la lista *AtmLstTotal*. Es decir:

   Para cada átomo Atm_i de la lista *AtmLstTotal*:

   a) Se obtiene la expresión Exp_i asociada al átomo Atm_i

   b) Se obtiene la lista *AtmLst* de átomos presentes en esa expresión Exp_i

   c) Para cada átomo $AtmExp_i$ de la lista anterior *AtmLst*

   - Si el átomo $AtmExp_i$ no está[6] en la lista general *AtmLstExp* se

[6]Chequear si $AtmExp_i$ no está en *AtmLstExp* se puede hacer de forma óptima mediante

Entrada: $MatIn$: matriz T $m \cdot n$ de expresiones simbólicas
Salida: $MatOut$: matriz T optimizada. $AtmLstOut$: Lista de átomos. $ExpLstOut$: Lista de expresiones
Definir la lista $AtmLstExp$ vacía
Definir una tabla de hash $RepAtmHashTable$ vacía

$AtmLstTotal \leftarrow$ atom_list($MatIn$)

para *cada átomo* Atm_i *de la lista* $AtmLstTotal$ **hacer**
  Obtener la expresión Exp_i asociada a Atm_i
  Obtener la lista $AtmLst$ de átomos presentes en la expresión Exp_i
  para *cada átomo* $AtmExp_i$ *de la lista* $AtmLst$ **hacer**
    si $AtmExp_i$ *no está en* $AtmLstExp$ **entonces**
      Introducir $AtmExp_i$ en la lista $AtmLstExp$
    Incrementar en $RepAtmHashTableOut$ el valor correspondiente a $AtmExp_i$ en una unidad

para $i = 0$ ***to*** n **hacer**
  para $j = 0$ ***to*** m **hacer**
    Obtener la lista $AtmLst$ de átomos presentes en la expresión $MatOut(i, j)$
    para *cada átomo* $AtmExp_i$ *de la lista* $AtmLst$ **hacer**
      si $AtmExp_i$ *no está en* $AtmLstExp$ **entonces**
        Introducir $AtmExp_i$ en la lista $AtmLstExp$
      en otro caso
        Incrementar en $RepAtmHashTableOut$ el valor correspondiente a $AtmExp_i$ en una unidad

para *cada átomo* Atm_i *de la lista* $AtmLstExp$ **hacer**
  si Atm_i *Aparece más de una vez en* $RepAtmHashTableOut$ **entonces**
    Añadir Atm_i a la lista $AtmLstOut$
    Obtener la expresión $AtmExp_i$ asociada a Atm_i.
    $Exp_i \leftarrow$ opt_unatomize_ex($AtmExp_i$,$RepAtmHashTableOut$)
    Añadir Exp_i a la lista $ExpLstOut$

para $i = 0$ ***to*** n **hacer**
  para $j = 0$ ***to*** m **hacer**
    $MatOut(i, j) \leftarrow$ opt_unatomize_ex($MatIn(i, j)$, $RepAtmHashTableOut$)

devolver $MatOut$,$AtmLstOut$, $ExpLstOut$

Figura 3.11.: opt_atom_list(Mat)

introduce en ella.

- Si el átomo $AtmExp_i$ sí está en la lista general $AtmLstExp$ se incrementa en una unidad el contador de ese átomo en la tabla hash $RepAtmHashTableOut$.

  Cada átomo tendrá asignado en la tabla un valor según el número de veces que aparezca.

3. Para cada elemento $MatIn(i,j)$ de $MatIn$ se han de buscar y contar los átomos que aparecen en las expresiones asociadas a los átomos de la lista $AtmLstTotal$. Es decir:

   a) Se obtiene la lista $AtmLst$ de átomos presentes en esa expresión $MatIn(i,j)$

   b) Para cada átomo $AtmExp_i$ de la lista anterior $AtmLst$

   - Si el átomo $AtmExp_i$ no está en la lista general $AtmLstExp$ se introduce en ella.
   - Si el átomo $AtmExp_i$ sí está en la lista general $AtmLstExp$ se incrementa en la tabla hash $RepAtmHashTableOut$ el contador de ese átomo en una unidad. Cada átomo tendrá asignado en la tabla un valor según el número de veces que aparezca.

4. Introducir en la lista $AtmLstOut$ los átomos que aparecen más de una vez según la tabla $RepAtmHashTableOut$.

   Sustituir en la expresión asociada a estos el resto de átomos (los que tienen el valor uno en $RepAtmHashTableOut$) por su expresión y guardar estas expresiones en la lista $ExpLstOut$. Es decir:

   Por cada átomo Atm_i de la lista $AtmLstExp$:

   - Si Atm_i Aparece más de una vez según $RepAtmHashTableOut$
     - Se añade el átomo Atm_i a la lista de salida $AtmLstOut$.
     - Se obtiene la expresión $AtmExp_i$ asociada a dicho Atm_i.
     - Se sustituyen en la expresión $AtmExp_i$ todos lo átomos que aparecen un única vez (según la tabla $RepAtmHashTableOut$)

una tabla hash.

por su expresión asociada. Esto se realiza mediante el algoritmo `opt_unatomize_ex`. Es resultado se guarda en Exp_i. Es análogo al un proceso de desatomización donde únicamente se desatomizan los átomos que aparecen una única vez según la tabla anterior.

- Se añade Exp_i a la lista de expresiones $ExpLstOut$.

5. Para cada elemento de $MatIn$:

   Se Sustituye en cada componente de $MatIn(i,j)$ los átomos que aparecen una única vez (según la tabla $RepAtmHashTableOut$) vez por su expresión asociada. Es resultado se guarda en el elemento correspondiente $MatOut(i,j)$ de la matriz de salida. Es análogo al un proceso de desatomización donde únicamente se desatomizan los átomos que aparecen una única vez según la tabla $RepAtmHashTableOut$.

3.7.4.1. Ejemplo

A fin de ayudar a la explicación anterior, se analiza como serían los cinco pasos del proceso de optimización en el ejemplo visto en 3.7.3.1.

En el punto 1, se obtiene la lista de todos los átomos que aparecen en las sub-expresiones del elemento a exportar (`delta`):

```
atom10 = -l_p*m_p;
atom12 = sin(-theta);
atom13 = atom10*atom12;
atom38 = atom13*dtheta;
atom39 = atom38*dtheta;
atom6  = -C_vis*dx;
atom57 = atom39+atom6;
atom58 =  atom57+F_b;
atom4  =  theta_0-theta;
atom5  = -k*atom4;
atom15 = atom13*g;
atom59 = atom5+atom15;
atom60 = M_p+atom59;
```

Siendo el elemento a exportar:

```
_delta[0] = -atom58;
_delta[1] = atom60;
```

En el punto 2, búsqueda de átomos que aparecen varías veces, se obtienen como resultado el átomo `atom13` que está dos veces.

En el punto 3, se sustituyen recursivamente en la expresión asociada a ese átomo todos los átomos que únicamente aparecen una vez, dando como resultado:

```
atom13 = -l_p*sin(-theta)*m_p;
```

En el punto 4, se han de buscar y contar lo átomos presentes en las expresiones del elemento a exportar. En este caso solo hay dos átomos (`atom58` y `atom60`) y cada uno aparece una vez.

Por último, en el punto 5, se sustituyen recursivamente en el elemento a exportar todos los átomos que únicamente aparecen una vez, obteniendo el resultado esperado:

```
_delta[0] = -(dtheta*dtheta)*atom13-F_b+c_vis*dx;
_delta[1] =  M_p+g*atom13+k*( theta-theta_0);
```

3.7.4.2. OPT_unatomize_ex

En la figura 3.12 se describe en pseudo-código el algoritmo que se encarga de desatomizar de forma selectiva, solo los átomos que tienen un uno asignado en la tabla *RepAtmHashTable*, es decir, los que aparecen una única vez.

El algoritmo toma como entrada la expresión $ExpIN$ y la tabla de átomos *RepAtmHashTable* que contiene un valor asociado a cada átomo. El algoritmo retorna la expresión $ExpOUT$.

La expresión $ExpIN$ puede estar completamente atomizada, parcialmente atomizada o incluso sin atomizar.

Entrada: $ExpIn$: expresión simbólica, $RepAtmHashTable$: Tabla hash
Salida: $ExpOut$: expresión simbólica

si $ExpIn$ *es de tipo átomo* Atm **entonces**
  si *el número asociado a* Atm *en* $RepAtmHashTable$ *es igual a* 1 **entonces**
    Obtener la expresión $AtmExp$ asociada al átomo
    $ExpUnatm \leftarrow$ OPT_unatomize_ex($AtmExp$,$RepAtmHashTable$)
    $ExpOUT \leftarrow ExpUnatm$
  en otro caso
    $ExpOUT \leftarrow Atm$
en otro caso
  si *tamaño de* $ExpIn > 1$ **entonces**
    $ExpUnatm \leftarrow ExpIn$
    para *cada sub-expressión* $ExpIn_i$ *de* $ExpIn$ **hacer**
      $ExpUnatm_i \leftarrow$OPT_unatomize_ex($ExpIn_i$,$RepAtmHashTable$)
      Sustituir en $ExpUnatm$ cada $ExpIn_i$ por su equivalente desatomizado $ExpUnatm_i$
    $ExpOUT \leftarrow ExpUnatm$
  en otro caso
    $ExpOUT \leftarrow ExpIn$
devolver $ExpOUT$

Figura 3.12.: opt_unatomize_ex

El algoritmo es una modificación del algoritmo de desatomización (3.4.2). Cada vez que se accede a una sub-expresión que se ha de desatomizar se chequea cuál el su valor en la tabla $RepAtmHashTable$. Si este valor es uno, se desatomiza. En caso de que sea mayor que uno se deja ese átomo sin desatomizar.

3.7.5. Exportación de matrices conjunta

En 2.7.3 se detalla el conjunto de elementos que se han de exportar para realizar la simulación dinámica de un sistema multicuerpo. En este apartado

se propone una optimización que permite evaluar ciertas funciones de una forma más eficiente.

El algoritmo de integración dinámica visto en la sección 2.4 puede ser completado (Figura 3.13) citando en qué momento se ha de evaluar numéricamente cada una de las funciones exportadas.

$t \leftarrow 0$
Evaluar $\mathbf{\Phi}^{Init}$,$\mathbf{\Phi_q}^{Init}$,$\dot{\mathbf{\Phi}}_{\dot{\mathbf{q}}}^{Init}$ y $\boldsymbol{\beta}^{Init}$
Problema de posición inicial $\mathbf{\Phi}^{Init}(\boldsymbol{q}) = 0$
Problema de velocidad inicial $\dot{\mathbf{\Phi}}^{Init}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$
mientras $t < t_{final}$ **hacer**
    Evaluar $\mathbf{M}$, δ, γ y $\dot{\mathbf{\Phi}}_{\dot{q}}$
    Resolver de problema dinámico $\mathbf{e}(\ddot{\boldsymbol{q}}, \boldsymbol{\lambda}) = 0$
    $\boldsymbol{q} \leftarrow \boldsymbol{q} + \dot{\boldsymbol{q}} \cdot \Delta t$
    $\dot{\boldsymbol{q}} \leftarrow \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}} \cdot \Delta t$
    Evaluar $\mathbf{\Phi}$ y $\mathbf{\Phi}_q$
    Proyección de coordenadas
    Evaluar $\dot{\mathbf{\Phi}}_{\dot{q}}$ y $\boldsymbol{\beta}$
    Proyección de velocidad
    $t \leftarrow t + \Delta t$

Figura 3.13.: Integración numérica. Evaluación de funciones

En el problema dinámico la matriz $\mathbf{M}$ y el vector δ, siempre se evalúan de forma conjunta. Por como están construidos (2.7.2), estos pueden contener átomos comunes. Entonces, lo más conveniente es exportarlos de forma conjunta. Aprovechando que el vector δ tienen un tamaño igual al número de filas de la matriz $\mathbf{M}$, se proponen exportarlos como una sola matriz, con el tamaño aumentado, es decir: $[\mathbf{M}|\delta]$.

De forma análoga, en la corrección de posición, se evalúan conjuntamente la matriz $\mathbf{\Phi}_q$ y el vector $\mathbf{\Phi}$, que también pueden contener átomos comunes. El tamaño del vector es igual al número de filas de la matriz, así que se puede proceder como se ha procedido en el caso anterior y crear una matiz aumentada $[\mathbf{\Phi}_q|\mathbf{\Phi}]$.

El software Robotran [11] exporta de una forma análoga esto dos elementos.

El código numérico que evalúe estos elementos deberá saber que la última columna de cada elemento corresponde al vector. Se deberá separar la matriz aumentada en los dos elementos, matriz y vector

Nótese que en caso de existir ecuaciones cinemáticas no-holónomas se han de exportar de forma conjunta la matriz $\dot{\mathbf{\Phi}}_{\dot{q}}$ y el vector $\dot{\mathbf{\Phi}}$, es decir, $[\dot{\mathbf{\Phi}}_{\dot{q}}|\dot{\mathbf{\Phi}}]$.

3.7.5.1. Ejemplo

Sean la matriz $\mathbf{M}$ y el vector δ correspondientes al vector de fuerzas generalizadas del mecanismo del ejemplo A.1.

En el listado de código de 3.7.5.1 muestra el código generado y exportado para la matriz $\mathbf{M}$. El código correspondiente al vector δ se puede ver en el listado 3.7.4 expuesto anteriormente.

```
atom14 = -l_p*m_p*cos(-theta);

_M[0] = -m_b-m_p;
_M[1] = atom14;
_M[2] = atom14;
_M[3] = -Iyy_p;
```

Listado 3.3: Exportación de la matriz de masa M

En el listado de código de 3.7.5.1 la matriz de masa $\mathbf{M}$ está representada (expresada en forma de vector, columna a columna) por los elementos `_MQ[0]`, `_MQ[1]`, `_MQ[2]` y `_MQ[3]`. Y el vector δ por lo elementos `_MQ[4]` y `_MQ[5]`.

```
atom10 = -l_p*m_p;
atom13 = sin(-theta)*atom10;
atom14 = atom10*cos(-theta);

_MQ[0] = -m_b-m_p;
_MQ[1] = atom14;
_MQ[2] = atom14;
```

```
_MQ[3] = -Iyy_p;
_MQ[4] = -(dtheta*dtheta)*atom13-F_b+c_vis*dx;
_MQ[5] = M_p+g*atom13+k*( theta-theta_0);
```

Listado 3.4: Exportación de la matriz aumentada MQ

El átomo `atom10` es común a los átomos `atom13` y `atom14` por lo que únicamente hay que evaluarlo una vez. Si los dos elementos se exportaran por separado, el átomo `atom10` debería ser evaluado dos veces aumentando el tiempo computacional de evaluación de la función.

CAPÍTULO 4

MODELADO SIMBÓLICO DE SISTEMAS MULTICUERPO

La parametrización del sistema multicuerpo está fuertemente ligada a la complejidad de las ecuaciones simbólicas que representan la cinemática y la dinámica del mismo. Dependiendo de cómo se parametrice el sistema se pueden obtener unas ecuaciones u otras. Esta parametrización es uno de los grados de libertad a la hora de modelizar el sistema ya que es el diseñador quien decide cómo ha de hacerlo.

En la literatura existen diversas posibilidades, por ejemplo, coordenadas absolutas, relativas, naturales [1] o cualquier combinación de ellas. Cada una de ellas requiere, si se desean resultados óptimos [1], un forma particular de plantear las ecuaciones.

En este capítulo se realiza, en primer lugar, un análisis (Sección 4.1) de la estructura que presentan los conjuntos de puntos y bases del sistema multicuerpo. En base a este análisis se propone un conjunto de técnicas (Sección 4.3) para

[1] Entendiéndose óptimo como la solución cuyas expresiones simbólicas son de un menor tamaño

la realización de cálculos cinemáticos (principalmente vectores de posición y de velocidad) que son independientes de la parametrización y que permiten obtener resultados óptimos. Es decir, sea cual sea el tipo de coordenadas definido las expresiones simbólicas obtenidas serán de un tamaño mínimo.

El tamaño de una expresión simbólica está ligado a la parentesización que esta presenta, es decir, a la capacidad de agrupar términos según la propiedad distributiva tanto de la suma como del producto. Estas técnicas presentadas permiten, no solo maximizar la parentesización, sino que en combinación con las técnicas de atomización presentadas en el Capítulo 3 permiten maximizar la reciclabilidad de los átomos. Como ya se verá, gracias a que las operaciones algebraicas se van a realizar siempre de forma idéntica, se facilita el trabajo al algoritmo de atomización asegurando que no se puedan generar dos átomos con expresiones asociadas idénticas.

Gracias a este conjunto de técnicas el diseñador delega en ellas el cálculo de las magnitudes cinemáticas. Así puede plantear las ecuaciones dinámicas por medio de la formulación que desee (y parametrizando el sistema como desee) sin tener que prestar atención a cálculos tediosos como son los de las citadas magnitudes cinemáticas.

Además, en la Sección 4.4, se presenta una generalización del elemento tensor de rango dos que permite trabajar con las distintas matrices jacobianas, debido a su naturaleza tensorial, que aparecen en el campo de la DSM.

Por último, se completa el capítulo con la descripción un método para el cálculo de la matriz $\mathbf{M}$ y el vector $\boldsymbol{\delta}$ basado en el PPV de forma que el resultado sea óptimo. Se entiende como "óptimo" aquel elemento cuyas expresiones simbólicas tienen la parentesización maximizada y con la máxima reciclabilidad de átomos.

4.1. Caracterización cinemática de sistemas multicuerpo

Desde una perspectiva clásica los sistemas multicuerpo se han definido por medio de una estructura de referencias a través de la cual se calculan las diferentes magnitudes cinemáticas de los cuerpos que forman el sistema.

En esta sección se describe la cinemática del sistema multicuerpo desde una perspectiva más general que la clásica. En esta perspectiva se consideran, por separado, los dos elementos fundamentales que definen la posición (punto) y orientación (base) de los sólidos en el espacio y topología de las estructuras que estos forman.

4.1.1. Estructura de bases

Sin pérdida de generalidad, se asume que el conjunto de bases usadas para definir las orientaciones de los diferentes cuerpos del sistema multicuerpo, así como las usadas para definir otro tipo de magnitudes (vectores y tensores) puede ser ordenado en un estructura con forma de árbol.

La raíz del árbol es la denominada "base inercial" o "base absoluta" y denotada[2] en este documento como $\mathcal{B}_{RI}$. Es la base a partir de la cual se construye el árbol de bases y siempre es fija.

Cada nueva base $\mathcal{B}_{i+1}$ se determina a partir de una base $\mathcal{B}_i$ previamente definida mediante una matriz $\mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_{i+1}}$ de cambio de base que, en general, puede ser función de las coordenadas generalizadas $\boldsymbol{q}$ y parámetros $\boldsymbol{p}$. Es decir:

$$\mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_{i+1}} = \mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_{i+1}}(\boldsymbol{q}, \boldsymbol{p}) \tag{4.1}$$

Cada base puede ser la base a partir de la cual se determinen una o más bases. En esta tesis se denomina "base anterior de la base $\mathcal{B}_{i+1}$" a la base $\mathcal{B}_i$. Es obvio que cada base, exceptuando la base inercial, tiene asociada una base anterior.

A modo de ejemplo, en la figura 4.1 se muestra un árbol de bases. Cada uno de los nodos de árbol representa una de las bases ($\mathcal{B}_{RI}$, $\mathcal{B}_1$, $\mathcal{B}_2$,...). Cada una de las uniones entre nodos representa la matriz de cambio de base ($\mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}$, $\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}$, $\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_3}$,...) que determina, a partir de otra base, una base según la dirección de la flecha.

Se dice que dos bases "están en la misma rama" cuando existe un camino

[2]En este documento las bases se han denotado con la letra $\mathcal{B}$

(conjunto de matrices de cambio de base) en dirección hacia en nodo raíz que une ambas bases. Por extensión se denomina "rama" a ese camino que une las citadas bases. Por ejemplo, las bases $\mathcal{B}_{RI}$ y $\mathcal{B}_4$ están en la misma rama, formada por las matrices $\mathbf{R}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}$, $\mathbf{R}^{\mathcal{B}_3}_{\mathcal{B}_1}$ y $\mathbf{R}^{\mathcal{B}_4}_{\mathcal{B}_3}$. En cambio, las bases $\mathcal{B}_6$ y $\mathcal{B}_9$ están en distintas ramas.

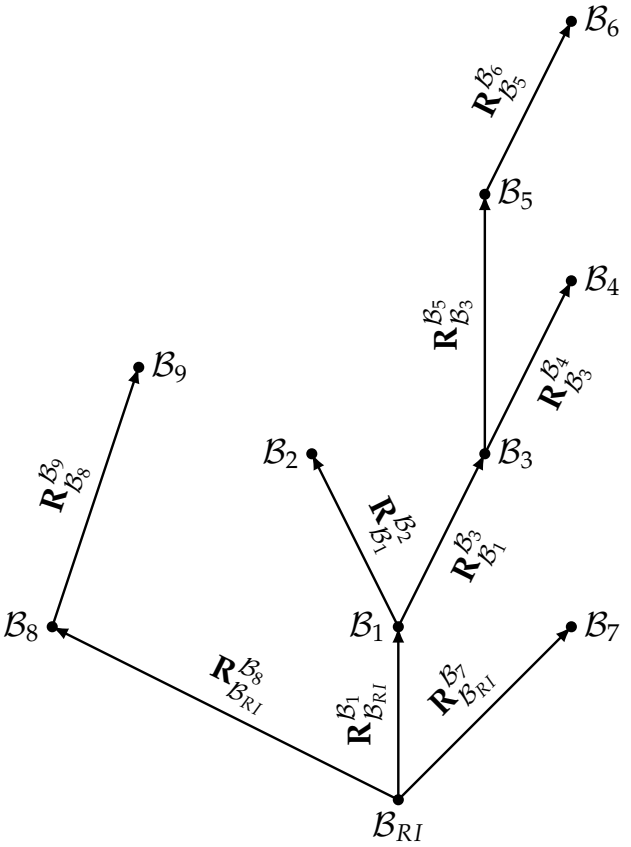


Figura 4.1.: Ejemplo de estructura de bases

4.1.2. Vector y tensor cartesiano. Representación y álgebra

En general, las expresiones de las ecuaciones de un sistema multicuerpo simbólico están ordenadas en forma de matriz.

Tanto vectores tridimensionales cartesianos[3] como tensores de rango dos cartesianos[4] son representados por matrices. Los vectores cartesianos por matrices de tamaño $3x1$ y los tensores cartesianos por matrices de tamaño $3x3$. Las componentes de estas matrices dependen de la base en que los vectores o los tensores estén representados.

Un vector arbitrario $\boldsymbol{v}$ se representa mediante la pareja formada por una base y matriz de $3x1$:

$$\left(\begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^{\mathsf{T}}, \mathcal{B}_v\right).$$

Un tensor arbitrario $\mathbf{T}$ se representa mediante la pareja la pareja formada por una base[5] y matriz de $3x3$:

$$\left(\begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{21} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix}, \mathcal{B}_T\right). \tag{4.2}$$

Por lo tanto, cada vector y cada tensor necesitan, implícitamente, tener una base "asociada" en la cual sus componentes tengan sentido.

4.1.2.1. Operaciones

En el contexto del cálculo simbólico de sistemas multicuerpo es necesario definir un conjunto de operaciones:

Sean $\boldsymbol{u}$ y $\boldsymbol{v}$ dos vectores cartesianos, $\mathbf{T}$ y $\mathbf{S}$ dos tensores cartesianos y a un símbolo genérico (escalar, símbolo o expresión simbólica).

[3] Vector cuyas componentes se representan en el espacio euclídeo tridimensional

[4] Tensor cuyas componentes se representan en el espacio euclídeo tridimensional

[5] Nótese que un tensor cartesiano ha de ser representado mediante dos bases. En este documento se realiza esta simplificación debido a que en este contexto las dos bases son la misma.

- Suma/resta de dos vectores cartesianos: $\boldsymbol{u} + \boldsymbol{v}$
- Suma/resta de dos tensores cartesianos: $\mathbf{T} + \mathbf{S}$
- Producto de un símbolo por un vector cartesiano: $a\ \boldsymbol{u}$
- Producto de un símbolo por un tensor cartesiano: $a\ \mathbf{T}$
- Producto de vector por vector: $\boldsymbol{u} \wedge \boldsymbol{v}$
- Producto de vector por tensor: $\boldsymbol{u}\ \mathbf{T}$
- Producto de una matriz por un vector. El producto vectorial entre dos vectores puede ser sustituido por el producto de una matriz anti-simétrica por un vector: $\widetilde{\boldsymbol{u}}\ \boldsymbol{v}$

Todas esta operaciones han de ser realizadas en una base que sea común a los elementos involucrados en la operación.

4.1.2.2. Cambio de base

Al poder estar las componentes de los vectores y tensores cartesianos definidas en cualquiera de las bases del sistema, al realizar una de las operaciones definidas anteriormente es necesario decidir en qué base se va a realizar. Por ejemplo, en el producto entre dos vectores cuyas componentes están representadas en bases distintas, la operación se puede realizar en la base del primero, en la base del segundo o en otra base del sistema.

Sean $\boldsymbol{u}$ un vector cartesiano, $\mathcal{B}_1$ y $\mathcal{B}_2$ dos bases y $\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}$ la matriz de rotación que determina la base $\mathcal{B}_2$ a partir de la base $\mathcal{B}_1$.

Sean $\left\{\boldsymbol{u}\right\}_{\mathcal{B}_2}$ las coordenadas del vector $\boldsymbol{u}$ representadas en la base $\mathcal{B}_2$. Se define cambio de base a la determinación de $\left\{\boldsymbol{u}\right\}_{\mathcal{B}_1}$, es decir, las coordenadas del mismo vector $\boldsymbol{u}$ representadas en la base $\mathcal{B}_1$, a partir de $\left\{\boldsymbol{u}\right\}_{\mathcal{B}_2}$ y se realiza como sigue:

$$\left\{\boldsymbol{u}\right\}_{\mathcal{B}_1} = \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2} \left\{\boldsymbol{u}\right\}_{\mathcal{B}_2} \tag{4.3}$$

4.1.2.3. Gravedad

En esta tesis se utiliza el modificador denominado "gravedad" para indicar en qué base se ha de realizar una operación entre vectores y/o tensores. Se puede dar el caso de que las bases asociadas a los elementos de la operación estén en la misma rama del árbol de bases o que que no.

Si las bases asociadas están en la misma rama existen dos posibilidades:

- "Gravedad DOWN": La base en que se realiza las operación es la base asociada a uno de los operandos que esté más abajo en la rama del árbol de bases.
- "Gravedad UP": La base en que se realiza las operación es la base asociada a uno de los operandos que esté más arriba en la rama del árbol de bases.

Por ejemplo, en el sistema de la figura 4.1 se quiere realizar una operación entre dos elementos cuyas bases asociadas son la base $\mathcal{B}_1$ y la base $\mathcal{B}_6$. Con el modificador "gravedad DOWN" la operación se realizará en la base $\mathcal{B}_1$, en cambio con el modificador "gravedad UP" se realizará en la base $\mathcal{B}_6$.

Cuando las bases asociadas están en distintas ramas operación se ha de realizar en la base común a las dos ramas que más arriba esté.

Por ejemplo, en el sistema de la figura 4.1 se quiere realizar una operación entre dos elementos cuyas bases asociadas son la base $\mathcal{B}_2$ y la base $\mathcal{B}_4$. La operación se realizará en la base $\mathcal{B}_1$, independientemente del sentido de la gravedad.

En general, todas la operaciones se realizarán con el modificador "gravedad UP". Como se verá en la sección 4.2 el modificador "gravedad" influye también en el orden en el que se han de realizar las operaciones entre vectores y/o entre tensores. En la citada sección se verá como este orden hace que se maximice la reciclabilidad de átomos y que la parentesización sea máxima.

4.1.3. Estructura de puntos

Análogamente al conjunto de bases, el conjunto de puntos presentes en el sistema multicuerpo se representa en un estructura con forma de árbol.

En dicha estructura de árbol hay un punto raíz, denominado[6] O (P_0 o B_0) y a partir de él se determinan el árbol de puntos.

Cada nuevo punto P_{i+1} se determina a partir de uno anterior P_i mediante un vector cartesiano $\mathbf{r}_{P_i}^{P_{i+1}}$ que, en general, puede ser función de las coordenadas generalizadas $\boldsymbol{q}$ y de los parámetros $\boldsymbol{p}$. Es decir:

$$\mathbf{r}_{P_i}^{P_{i+1}} = \mathbf{r}_{P_i}^{P_{i+1}}(\boldsymbol{q}, \boldsymbol{p}) \tag{4.4}$$

Cada punto puede ser el punto a partir del cual se determinen uno o más puntos. En esta tesis se denomina "punto anterior del punto P_{i+1}" al punto P_i. Es obvio que cada punto, a excepción del primero, tiene un punto anterior.

A modo de ejemplo, en la figura 4.2 se muestra un árbol de puntos. Cada uno de los nodos de árbol representa uno de los puntos (O, P_1, P_2...). Cada una de las uniones entre nodos representa el vector ($\mathbf{r}_{P_1}^{P_3}$, $\mathbf{r}_{P_6}^{P_7}$, ...) que determina, a partir de otro punto, un punto según la dirección de la flecha.

Se dice que dos puntos "están en la misma rama" cuando existe un camino (conjunto de vectores de posición) en dirección hacia en nodo raíz que une ambos puntos. Por extensión se denomina "rama" al camino que une los citados puntos. Por ejemplo, los puntos P_4 y O están en la misma rama, formada por los vectores $\mathbf{r}_O^{P_1}$, $\mathbf{r}_{P_1}^{P_3}$ y $\mathbf{r}_{P_3}^{P_4}$. En cambio, los puntos P_7 y P_8 están en distintas ramas.

4.1.4. Caracterización cinemática de cada sólido. Referencia

La definición cinemática de cada sólido S_i se realiza por medio una referencia R_i, es decir, de un par formado por un punto (B_i) de la estructura de puntos del sistema y una base ($\mathcal{B}_i$) de la estructura de bases del sistema.

La figura 4.3 muestra un sólido genérico S_i, con la referencia R_i y dos puntos arbitrarios P_{i_1} y P_{i_2} considerados pertenecientes al sólido.

Esta separación del sistema en estructura de bases y de puntos permite la definición de forma libre de referencias a partir de cualquier punto y de cualquier

[6] En este tesis, los puntos genéricos, en general, se han denotado con la letra P o B.

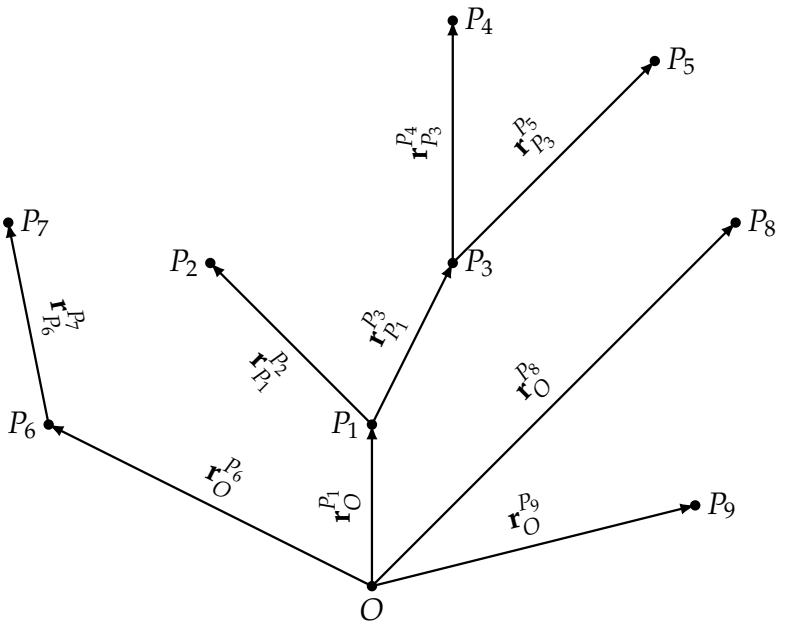


Figura 4.2.: Ejemplo de estructura de puntos

base. Nótese que en este contexto deja de tener sentido hablar de estructura de referencias ya que no existe como tal. El conjunto de referencias del sistema multicuerpo está formado por un conjunto de parejas de punto más base que previamente hayan sido definidas en el mismo.

4.1.4.1. Cadena cerrada

La estructuras de puntos y de bases son estructuras de cadena abierta, es decir, no hay ramas del árbol que formen lazos cerrados.

El cierre de las ramas, en caso de ser necesario, se realiza mediante una ecuación o varias de cierre según cuál sea la naturaleza del sistema.

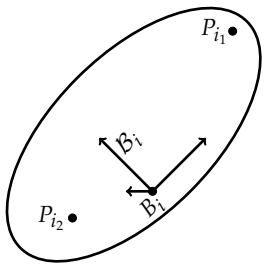


Figura 4.3.: Sólido genérico

4.2. Modificador "gravedad": Atomización y Parentesización

Como ya se ha comentado en la sección precedente, el modificador "gravedad", además de determinar en qué base se han de realizar las operaciones entre vectores y/o tensores, permite asegurar que estas operaciones se realizan siguiendo un orden específico. Este modificador no influye en las operaciones entre matrices de rotación por no tener naturaleza ni vectorial ni tensorial.

Cabe recordar que el modificador "gravedad" únicamente es válido cuando los elementos (puntos o bases) están en la misma rama del árbol correspondiente.

El orden en que se han de realizar las operaciones, dependiendo el modificador, ha de ser el siguiente:

"gravedad DOWN" La operación se ha de realizar siguiendo un camino descendiente a través de la correspondiente rama del árbol. Es decir, se ha comenzar operando con los elementos que más alejados (más arriba en la correspondiente rama del árbol) estén de la raíz del árbol e ir descendiendo y operando a lo largo de la rama.

"gravedad UP" La operación se ha de realizar siguiendo un camino ascendente a través de la correspondiente rama del árbol. Es decir, se ha comenzar operando con los elementos que más cerca (más abajo en la correspondiente rama del árbol) estén de la raíz del árbol e ir ascendiendo y operando a lo largo de la rama.

Realizar la operaciones siguiendo un orden específico asegura que las expresiones simbólicas asociadas a estas operaciones se realicen siempre en un orden específico y de la misma forma. Esto presenta dos ventajas:

- Favorece la parentesización en la expresiones simbólicas obtenidas.
- Favorece, en caso de usar técnicas de atomización, la reciclabilidad de átomos.

4.2.1. Ejemplo: Vector de posición entre dos puntos

A continuación se ilustra por medio de un ejemplo (el cálculo de un vector de posición) como el modificador contribuye a optimizar tanto la parentesización como la reciclabilidad de átomos. Además, en el mismo ejemplo se justifica porque es conveniente el uso del modificador como "gravedad UP".

Sean los árboles de bases (Figura 4.1) y puntos (Figura 4.2) vistos en la sección anterior donde cada uno de los vectores del árbol de puntos tiene sus componentes conocidas en una base distinta. Así:

$$\mathbf{r}_O^{P_1} \rightarrow \left\{\mathbf{r}_O^{P_1}\right\}_{\mathcal{B}_{RI}} \quad , \quad \mathbf{r}_{P_1}^{P_3} \rightarrow \left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1} \quad , \quad \mathbf{r}_{P_3}^{P_4} \rightarrow \left\{\mathbf{r}_{P_3}^{P_4}\right\}_{\mathcal{B}_3} \quad \text{y} \quad \mathbf{r}_{P_3}^{P_5} \rightarrow \left\{\mathbf{r}_{P_3}^{P_5}\right\}_{\mathcal{B}_3} \tag{4.5}$$

El vector de posición $\mathbf{r}_O^{P_4}$ se obtiene como: $\mathbf{r}_O^{P_4} = \mathbf{r}_O^{P_1} + \mathbf{r}_{P_1}^{P_3} + \mathbf{r}_{P_3}^{P_4}$.

El vector de posición $\mathbf{r}_O^{P_5}$ se obtiene como: $\mathbf{r}_O^{P_5} = \mathbf{r}_O^{P_1} + \mathbf{r}_{P_1}^{P_3} + \mathbf{r}_{P_3}^{P_5}$.

4.2.1.1. Parentesización

El cálculo de estos vectores cuando se utiliza el modificador "gravedad DOWN" se realiza como sigue:

$$\begin{aligned}
\left\{\mathbf{r}_{P_0}^{P_4}\right\}_{\mathcal{B}_{RI}} &= \left\{\mathbf{r}_O^{P_1}\right\}_{\mathcal{B}_{RI}} + \mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}\left(\left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1} + \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_3}\left\{\mathbf{r}_{P_3}^{P_4}\right\}_{\mathcal{B}_3}\right) \\
\left\{\mathbf{r}_{P_0}^{P_5}\right\}_{\mathcal{B}_{RI}} &= \left\{\mathbf{r}_O^{P_1}\right\}_{\mathcal{B}_{RI}} + \mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}\left(\left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1} + \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_3}\left\{\mathbf{r}_{P_3}^{P_5}\right\}_{\mathcal{B}_3}\right)
\end{aligned} \tag{4.6}$$

El cálculo de estos vectores cuando se utiliza el modificador "gravedad UP" se realiza como sigue:

$$\begin{aligned}\left\{\mathbf{r}_{P_0}^{P_4}\right\}_{\mathcal{B}_3} &= \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1}\left(\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_{RI}}\left\{\mathbf{r}_{O}^{P_1}\right\}_{\mathcal{B}_{RI}} + \left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1}\right) + \left\{\mathbf{r}_{P_3}^{P_4}\right\}_{\mathcal{B}_3} \\ \left\{\mathbf{r}_{P_0}^{P_5}\right\}_{\mathcal{B}_3} &= \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1}\left(\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_{RI}}\left\{\mathbf{r}_{O}^{P_1}\right\}_{\mathcal{B}_{RI}} + \left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1}\right) + \left\{\mathbf{r}_{P_3}^{P_5}\right\}_{\mathcal{B}_3}\end{aligned} \tag{4.7}$$

Nótese como en ambos casos, tanto con "gravedad DOWN" como con "gravedad UP" la parentesización es óptima[7].

4.2.1.2. Reciclabilidad de átomos

Cuando se realiza la operación "gravedad UP" (Ecuación 4.7) se puede ver que en ambas expresiones, gracias a la parentesización, hay un conjunto de operaciones y operandos agrupadas en un término que es común:

$$\mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1}\left(\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_{RI}}\left\{\mathbf{r}_{O}^{P_1}\right\}_{\mathcal{B}_{RI}} + \left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1}\right)$$

Este término está asociado a la parte común de ambos caminos (de $O \rightarrow P_4$ y de $O \rightarrow P_5$) en el árbol de puntos.

Si se trabaja con técnicas de atomización, las expresiones simbólicas en términos de átomos será iguales , maximizando así la reciclabilidad de átomos.

En caso de utilizar el modificador "gravedad DOWN" (Ecuación 4.6) este agrupación de operaciones no aparece. Los términos comunes en ambas ecua-

[7] En caso de no realizar este cálculo como se ha descrito en este sección se obtendría para el vector $\mathbf{r}_{P_0}^{P_4}$:

- Con "gravedad DOWN": $\left\{\mathbf{r}_{P_0}^{P_4}\right\}_{\mathcal{B}_{RI}} = \left\{\mathbf{r}_{O}^{P_1}\right\}_{\mathcal{B}_{RI}} + \mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}\left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1} + \mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_3}\left\{\mathbf{r}_{P_3}^{P_4}\right\}_{\mathcal{B}_3}$
- Con "gravedad UP": $\left\{\mathbf{r}_{P_0}^{P_4}\right\}_{\mathcal{B}_3} = \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_{RI}}\left\{\mathbf{r}_{O}^{P_1}\right\}_{\mathcal{B}_{RI}} + \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1}\left\{\mathbf{r}_{P_1}^{P_3}\right\}_{\mathcal{B}_1} + \left\{\mathbf{r}_{P_3}^{P_4}\right\}_{\mathcal{B}_3}$

Nótese como en ambas expresiones hay una matriz de cambio de base ($\mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_{RI}}$ o $\mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_3}$) que implica dos cambios de base. Es decir $\mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_{RI}} = \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1}\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_{RI}}$ y $\mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_3} = \mathbf{R}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_3}$). Mientras que en la ecuaciones 4.6 y 4.7 todas la matrices cambio de base únicamente implican un cambio de base.

ciones están asociados únicamente a operandos y únicamente se podrán reciclar átomos asociados a esos operandos.

4.2.1.3. Elementos con componentes nulas

Cuando uno de los elementos involucrados en las operaciones tenga todas sus componentes nulas, la base en la que se ha de representar el resultado es la base asociada al elemento cuyas componentes no son nulas, independientemente del modificador "gravedad".

Por ejemplo, sean las componentes de los vectores $\boldsymbol{u}$ y $\boldsymbol{v}$, representadas, respectivamente, en las bases $\mathcal{B}_u$ y $\mathcal{B}_v$:

$$\{\boldsymbol{u}\}_{\mathcal{B}_u} = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix}_{\mathcal{B}_u} \quad \text{y} \quad \{\boldsymbol{v}\}_{\mathcal{B}_v} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix}_{\mathcal{B}_v} \tag{4.8}$$

Entonces la suma de los vectores realiza como sigue:

$$\{\boldsymbol{u}\}_{\mathcal{B}_u} + \mathbf{R}_{\mathcal{B}_u}^{\mathcal{B}_v} \{\boldsymbol{v}\}_{\mathcal{B}_v} = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix}_{\mathcal{B}_u} \tag{4.9}$$

Esta forma de proceder evita, sin más cautela, la realización de cambios de base innecesarios.

4.3. Operadores cinemáticos

En esta sección se presenta un conjunto de cuatro operadores orientados a la realización de los cálculos típicos de la cinemática en sistemas multicuerpo.

Estos cuatro operadores están orientados a realizar las siguientes operaciones:

- Cálculo de la matriz de rotación entre dos bases.
- Cálculo del vector de posición de un punto respecto de otro.

- Cálculo del vector de velocidad angular entre dos bases.
- Cálculo del vector velocidad de un punto respecto a una referencia.

Esto operadores son independientes del tipo de formulación y/o de parametrización que se esté empleando. Pueden realizar los cálculos tanto con el modificador "gravedad DOWN" como "gravedad UP" maximizando la parentesización. Este hecho hace que el tamaño de las expresiones simbólicas de las componentes del vector o matriz resultado sea óptimo en términos de número de operaciones algebraicas necesarias para su evaluación numérica.

Este conjunto de operadores permite sacar provecho de la recursividad subyacente en las estructuras de puntos y bases y que es debida al hecho de que estos elementos (puntos y bases) estén definidos a partir de un elemento anterior.

4.3.1. Operador: Matriz de rotación

El operador "Matriz de rotación" calcula la matriz de rotación entre dos bases que están en una misma rama del árbol de bases. Para ello se recorre la citada rama y se realiza el producto entre todas las matrices de rotación intermedias. Estos matrices de rotación intermedias son las matrices a través de las cuales se determinan cada una de las bases de la citada rama.

La rama entre las dos bases se ha de recorrer, independientemente del sentido del modificador "gravedad", en sentido ascendente y la operación entre matrices se ha de realizar post-multiplicando estas. Así se maximizará la reciclabilidad de átomos como se demuestra a continuación.

4.3.1.1. Descripción

Sean las bases $\mathcal{B}_A$ y $\mathcal{B}_B$, las cuales se encuentran en la misma rama del árbol (formada por las bases $\mathcal{B}_A$,$\mathcal{B}_{A+1}$,... ,$\mathcal{B}_{B-1}$ y $\mathcal{B}_B$) siendo la base $\mathcal{B}_A$ la que más abajo se encuentra (más cerca de la base raíz) y la base $\mathcal{B}_B$ la que más arriba se encuentra (más alejada de la base raíz).

La matriz de rotación $\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B}$ se obtiene como:

$$\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B} = \left(\left(\left(\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_{A+1}}\, \mathbf{R}_{\mathcal{B}_{A+1}}^{\mathcal{B}_{A+2}}\right)\ \ldots\ \right) \mathbf{R}_{\mathcal{B}_{B-2}}^{\mathcal{B}_{B-1}}\right) \mathbf{R}_{\mathcal{B}_{B-1}}^{\mathcal{B}_B} \tag{4.10}$$

Los paréntesis han sido introducidos con la intención de remarcar el orden en que se ha de realizar esta operación para maximizar la reciclabilidad de átomos.

La matriz de rotación $\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}$ se obtiene como:

$$\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A} = \mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_{B-1}} \left(\mathbf{R}_{\mathcal{B}_{B-1}}^{\mathcal{B}_{B-2}} \left(\ldots \left(\mathbf{R}_{\mathcal{B}_{A+2}}^{\mathcal{B}_{A+1}}\, \mathbf{R}_{\mathcal{B}_{A+1}}^{\mathcal{B}_A} \right) \right) \right) \tag{4.11}$$

Nótese que $\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B} = \mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}{}^{\mathsf{T}}$ y por lo tanto conocido uno, la obtención del otro es inmediato.

Por lo tanto, para el cálculo de $\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}$ se ha realizar primero el cálculo de $\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B}$ y a continuación $\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B} = \mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}{}^{\mathsf{T}}$.

4.3.1.2. Algoritmo

El operador "Matriz de rotación" que calcula la matriz de rotación entre dos bases que están en una misma rama del árbol de bases se implementa mediante el algoritmo recursivo de la figura 4.4 y que se describe en este apartado.

En este algoritmo la base $\mathcal{B}_B$ es la que más arriba está en la cadena y la base $\mathcal{B}_A$ la que más abajo está y calcula la matriz $\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B}$ tal y como se detalla en la ecuación 4.10 .

$\omega_{OUT} \leftarrow$Rotation_Matrix($\mathcal{B}_A$, $\mathcal{B}_B$)

Entrada: $\mathcal{B}_A$: Primera base. $\mathcal{B}_B$: Segunda base.
Salida: Vector $\mathbf{R}_{OUT}$ de rotación

Obtener $\mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$

si $\mathcal{B}_{B-1}$!= $\mathcal{B}_A$ **entonces**
  $\mathbf{R}_{AUX} \leftarrow$ Rotation_Matrix($\mathcal{B}_A$, $\mathcal{B}_{B-1}$)
  $\mathbf{R}_{OUT} \leftarrow \mathbf{R}_{AUX}\, \mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$
si $\mathcal{B}_{B-1} = \mathcal{B}_A$ **entonces**
  $\mathbf{R}_{OUT} \leftarrow \mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$
devolver $\mathbf{R}_{OUT}$

Figura 4.4.: Algoritmo Rotation_Matrix

Cada vez que se llama a dicho algoritmo recursivo se realizan los siguientes pasos:

1. Se obtiene la matriz de rotación $\mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$ mediante la cual ha sido definida, a partir de la base $\mathcal{B}_{B-1}$ la base $\mathcal{B}_B$
2. Si la base $\mathcal{B}_{B-1}$ es distinta de $\mathcal{B}_A$ quiere decir que no se ha llegado abajo de la cadena, entonces:
   - Se llama al algoritmo recursivo con las bases $\mathcal{B}_A$, $\mathcal{B}_{B-1}$. La salida del algoritmo se asigna a la matriz auxiliar $\mathbf{R}_{AUX}$.
   - Se post-multiplica la matriz $\mathbf{R}_{AUX}$ por la matriz $\mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$ (calculada previamente). El resultado se asigna a la matriz de salida $\mathbf{R}_{OUT}$.
3. Si la base $\mathcal{B}_{B-1}$ es la base $\mathcal{B}_A$ quiere decir que ya se ha llegado al punto más bajo de la rama, entonces:
   - Se asigna a la matriz de salida $\mathbf{R}_{OUT}$ la matriz $\mathbf{R}^{\mathcal{B}_B}_{\mathcal{B}_{B-1}}$.

4.3.1.3. Matriz de rotación entre bases en distintas cadenas

En el caso en que las bases $\mathcal{B}_A$ y $\mathcal{B}_B$ se encuentren en ramas distintas, la solución tiene dos pasos.

En primer lugar se ha de buscar base $\mathcal{B}_C$ común a las dos ramas y se calculan mediante el algoritmo anterior las matrices de rotación $\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}$ y $\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_A}$.

Finalmente, la matriz de rotación se obtiene como:

$$\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B} = \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_A}{}^{\mathsf{T}} \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B} \tag{4.12}$$

4.3.2. Operador: Vector de posición

El operador "Vector de posición" calcula el vector de posición entre dos puntos que están en una misma rama.

Para ello se recorre la citada rama y se realiza la suma de todos los vectores de posición intermedios. Estos vectores intermedios son los vectores a través de los cuales se determinan cada uno de los puntos de la citada rama.

Si el modificador es "gravedad UP" el resultado es un vector cuyas componentes están representadas en la base que más arriba esté de todo el conjunto de bases asociadas a los vectores intermedios.

En cambio, si el modificador es "gravedad DOWN" el resultado es un vector cuyas componentes están representadas en la base que más abajo esté de todo el conjunto de bases asociadas a los vectores intermedios.

4.3.2.1. Descripción

Sean los puntos P_A y P_B, los cuales se encuentra en la misma rama del árbol (formada por P_A,P_{A+1},... ,P_{B-1} , P_B) siendo el punto P_A el que más abajo se encuentra (más cerca del punto raíz) y el punto P_B el que más arriba se encuentra (más alejado del punto raíz).

La suma de los vectores intermedios asociados a cada punto se realiza de-

pendiendo del sentido del modificador "gravedad".

Si el modificador es "gravedad DOWN" la suma se realiza:

$$\mathbf{r}_{P_A}^{P_B} = \mathbf{r}_{P_A}^{P_{A+1}} + \left(\mathbf{r}_{P_{A+1}}^{P_{A+2}} + ... + \left(\mathbf{r}_{P_{B-2}}^{P_{B-1}} + \mathbf{r}_{P_{B-1}}^{P_B}\right)\right) \tag{4.13}$$

Si el modificador es "gravedad UP" la suma se realiza:

$$\mathbf{r}_{P_A}^{P_B} = \left(\left(\left(\mathbf{r}_{P_A}^{P_{A+1}} + \mathbf{r}_{P_{A+1}}^{P_{A+2}}\right) + ...\right) + \mathbf{r}_{P_{B-2}}^{P_{B-1}}\right) + \mathbf{r}_{P_{B-1}}^{P_B} \tag{4.14}$$

Los paréntesis han sido introducidos con la intención de remarcar el orden en que se ha de realizar esta operación para maximizar la reciclabilidad de átomos.

Nótese que en caso de querer obtener el vector $\mathbf{r}_{P_B}^{P_A}$ el proceso es trivial, ya que $\mathbf{r}_{P_B}^{P_A} = -\mathbf{r}_{P_A}^{P_B}$

4.3.2.2. Algoritmo

El operador "Vector de posición" se implementa mediante el algoritmo recursivo de la figura 4.5.

El algoritmo recursivo toma como datos de entrada el punto P_A (el punto que más abajo está en la cadena), el punto P_B (el punto que más arriba está en la cadena) y el vector $\mathbf{r}_{IN}$.

Cada vez que se llama a dicho algoritmo recursivo se realizan los siguientes pasos:

1. Se obtiene el vector de posición $\mathbf{r}_{P_{B-1}}^{P_B}$ del punto P_B respecto al punto a partir del cual ha sido definido (P_{B-1}).
2. Si la gravedad es DOWN:
   - Si el punto $P_B - 1$ es distinto de P_A quiere decir que no se ha llegado abajo de la cadena, entonces:
     - Se suma el vector de entrada $\mathbf{r}_{IN}$ con el vector $\mathbf{r}_{P_{B-1}}^{P_B}$ y el resulta-

$\mathbf{r}_{OUT} \leftarrow$Position_Vector($P_A$, P_B, $\mathbf{r}_{IN}$)

Entrada: P_A: Primer punto. P_B: Segundo punto. $\mathbf{r}_{IN}$: Vector de entrada
Salida: Vector $\mathbf{r}_{OUT}$

Obtener $\mathbf{r}_{P_{B-1}}^{P_B}$

si *Gravedad DOWN* **entonces**
- **si** P_{B-1} != P_A **entonces**
  - $\mathbf{r}_{AUX} \leftarrow \mathbf{r}_{IN} + \mathbf{r}_{P_{B-1}}^{P_B}$
  - $\mathbf{r}_{OUT} \leftarrow$ Position_Vector(P_A, P_{B-1}, $\mathbf{r}_{AUX}$)
- **si** $P_{B-1} = P_A$ **entonces**
  - $\mathbf{r}_{OUT} \leftarrow \mathbf{r}_{IN} + \mathbf{r}_{P_{B-1}}^{P_B}$

si *Gravedad UP* **entonces**
- **si** P_{B-1} != P_A **entonces**
  - $\mathbf{r}_{AUX} \leftarrow$ Position_Vector(P_A, P_{B-1}, $\mathbf{r}_{IN}$)
  - $\mathbf{r}_{OUT} \leftarrow \mathbf{r}_{AUX} + \mathbf{r}_{P_{B-1}}^{P_B}$
- **si** $P_{B-1} = P_A$ **entonces**
  - $\mathbf{r}_{OUT} \leftarrow \mathbf{r}_{P_{B-1}}^{P_B}$

devolver $\mathbf{r}_{OUT}$

Figura 4.5.: Algoritmo Position_Vector

do se asigna al vector auxiliar $\mathbf{r}_{AUX}$.

- Se llama al algoritmo recursivo con los puntos P_A, P_{B-1} y el vector auxiliar $\mathbf{r}_{AUX}$. La salida del algoritmo se asigna a $\mathbf{r}_{OUT}$.

- Si el punto P_{B-1} es igual a P_A quiere decir que se ha llegado abajo de la cadena, entonces:
  - Se suma el vector de entrada $\mathbf{r}_{IN}$ con el vector $\mathbf{r}_{P_{B-1}}^{P_B}$ y el resultado se asigna al vector $\mathbf{r}_{OUT}$.

3. Si la gravedad es UP:
   - Si el punto $P_B - 1$ es distinto de P_A quiere decir que no se ha llegado abajo de la cadena, entonces:
     - Se llama al algoritmo recursivo con los puntos P_A, P_{B-1} y el vector de entrada $\mathbf{r}_{IN}$. La salida del algoritmo se asigna al vector auxiliar $\mathbf{r}_{AUX}$.
     - Se suma el vector auxiliar $\mathbf{r}_{AUX}$ con el vector $\mathbf{r}_{P_{B-1}}^{P_B}$ y el resultado se asigna al vector de salida $\mathbf{r}_{OUT}$.
   - Si el punto P_{B-1} es igual a P_A quiere decir que se ha llegado abajo de la cadena, entonces:
     - Se asigna al vector de salida $\mathbf{r}_{OUT}$ el vector $\mathbf{r}_{P_{B-1}}^{P_B}$.

Se ha de destacar que en la primera iteración el algoritmo se ha de llamar como sigue:

Position_Vector(P_A, P_B, $\mathbf{r}_{IN}$)

donde $\mathbf{r}_{IN}$ es un vector vacío (las componentes son cero y sin base asociada).

4.3.2.3. Vector de posición entre puntos en distintas cadenas

En el caso en que los puntos P_A y P_B para los que se quiere calcular el vector de posición $\mathbf{r}_{P_A}^{P_B}$ se encuentren en ramas distintas, la solución tiene dos pasos.

En primer lugar se ha de buscar el punto P_C común a las dos ramas en las que están los puntos y se calculan mediante el algoritmo anterior los vectores $\mathbf{r}_{P_C}^{P_B}$ y $\mathbf{r}_{P_C}^{P_A}$.

Nótese que al estar los puntos en dos ramas distintas del árbol el modificador gravedad deja de tener sentido.

Finalmente el vector de posición $\mathbf{r}_{P_A}^{P_B}$ se obtiene como:

$$\mathbf{r}_{P_A}^{P_B} = \mathbf{r}_{P_C}^{P_B} - \mathbf{r}_{P_C}^{P_A} \tag{4.15}$$

4.3.3. Operador: Vector de velocidad angular

El operador "Vector de velocidad angular" calcula la velocidad angular entre dos bases que están en una misma rama del árbol de bases.

Para ello se recorre la citada rama y se realiza la suma de todos los vectores de velocidad angular intermedio entre cada pareja de bases de la rama.Y a se ha visto en 4.1.1 que cada base está definida a partir de una base anterior mediante una matriz de rotación general, así pues obtener la velocidad angular de una base respecto a la base que está definida es directo.

Si el modificador es "gravedad UP" el resultado es un vector cuyas componentes están representadas en la base que más arriba esté en la citada cadena de bases.

En cambio, si el modificador es "gravedad DOWN" el resultado es un vector cuyas componentes están representadas en la base que más abajo esté en la cadena de bases.

4.3.3.1. Descripción

Sean las bases $\mathcal{B}_A$ y $\mathcal{B}_B$, las cuales se encuentran en la misma rama del árbol (formada por las bases $\mathcal{B}_A$,$\mathcal{B}_{A+1}$,... ,$\mathcal{B}_{B-1}$ y $\mathcal{B}_B$) siendo la base $\mathcal{B}_A$ la que más abajo se encuentra (más cerca de la base raíz) y la base $\mathcal{B}_B$ la que más arriba se encuentra (más alejada de la base raíz).

La suma de los vectores de velocidad angular intermedios asociados a cada base se realiza dependiendo del sentido del modificador "gravedad".

Si el modificador es "gravedad DOWN" la suma se realiza:

$$\boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_B} = \left(\left(\left(\boldsymbol{\omega}_{\mathcal{B}_{B-1}}^{\mathcal{B}_B} + \boldsymbol{\omega}_{\mathcal{B}_{B-2}}^{\mathcal{B}_{B-1}}\right) + ...\right) + \boldsymbol{\omega}_{\mathcal{B}_{A+1}}^{\mathcal{B}_{A+2}}\right) + \boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_{A+1}} \tag{4.16}$$

Si el modificador es "gravedad UP" la suma se realiza:

$$\boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_B} = \left(\left(\left(\boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_{A+1}} + \boldsymbol{\omega}_{\mathcal{B}_{A+1}}^{\mathcal{B}_{A+2}}\right) + ...\right) + \boldsymbol{\omega}_{\mathcal{B}_{B-1}}^{\mathcal{B}_{B-1}}\right) + \boldsymbol{\omega}_{\mathcal{B}_{B-1}}^{\mathcal{B}_B} \tag{4.17}$$

Los paréntesis han sido introducidos con la intención de remarcar el orden en que se ha de realizar esta operación para maximizar la reciclabilidad de átomos.

Nótese que en caso de querer obtener el vector $\boldsymbol{\omega}_{\mathcal{B}_B}^{\mathcal{B}_A}$ el proceso es inmediato, ya que $\boldsymbol{\omega}_{\mathcal{B}_B}^{\mathcal{B}_A} = -\boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_B}$.

4.3.3.2. Algoritmo

El operador "Vector de velocidad angular" se implementa mediante el algoritmo recursivo de la figura 4.6.

Al ser un algoritmo análogo, se ha de recorrer un rama del árbol sumando vectores intermedios, al algoritmo Position_Vector (Figura 4.5) carece de interés su descripción en detalle.

Se ha destacar que en la primera iteración el algoritmo se ha de llamar como sigue:

Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_B$,$\boldsymbol{\omega}_{IN}$)

donde $\boldsymbol{\omega}_{IN}$ es un vector vacío.

4.3.3.3. Velocidad angular entre bases en distintas ramas

En el caso en que las bases $\mathcal{B}_A$ y $\mathcal{B}_B$ se encuentren en ramas distintas, la solución tiene dos pasos.

En primer lugar se ha de buscar una base $\mathcal{B}_C$ común a las dos ramas y se calculan mediante el algoritmo anterior los vectores $\boldsymbol{\omega}_{\mathcal{B}_C}^{\mathcal{B}_B}$ y $\boldsymbol{\omega}_{\mathcal{B}_C}^{\mathcal{B}_A}$.

ω_{OUT} ←Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_B$,ω_{IN})

Entrada: $\mathcal{B}_A$: Primera base. $\mathcal{B}_B$: Segunda base. ω_{IN} Vector de entrada
Salida: Vector ω_{OUT} de velocidad angular

Obtener $\omega_{\mathcal{B}_{B-1}}^{\mathcal{B}_B}$

si *GRAVEDAD HACIA ABAJO* **entonces**
  si $\mathcal{B}_{B-1}$!= $\mathcal{B}_A$ **entonces**
    $\omega_{AUX} \leftarrow \omega_{IN} + \omega_{\mathcal{B}_{B-1}}^{\mathcal{B}_B}$
    $\omega_{OUT} \leftarrow$ Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_{B-1}$, ω_{AUX})
  si $\mathcal{B}_{B-1} = \mathcal{B}_A$ **entonces**
    $\omega_{OUT} \leftarrow \omega_{IN} + \omega_{\mathcal{B}_{B-1}}^{\mathcal{B}_B}$
si *GRAVEDAD HACIA ARRIBA* **entonces**
  si $\mathcal{B}_{B-1}$!= $\mathcal{B}_A$ **entonces**
    $\omega_{AUX} \leftarrow$ Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_{B-1}$, ω_{IN})
    $\omega_{OUT} \leftarrow \omega_{AUX} + \omega_{\mathcal{B}_{B-1}}^{\mathcal{B}_B}$
  si $\mathcal{B}_{B-1} = \mathcal{B}_A$ **entonces**
    $\omega_{OUT} \leftarrow \omega_{\mathcal{B}_{B-1}}^{\mathcal{B}_B}$

devolver ω_{OUT}

Figura 4.6.: Algoritmo Angular_Velocity_Vector

El vector velocidad angular $\omega_{\mathcal{B}_A}^{\mathcal{B}_B}$ será el resultado de: $\omega_{\mathcal{B}_A}^{\mathcal{B}_B} = \omega_{\mathcal{B}_C}^{\mathcal{B}_B} - \omega_{\mathcal{B}_C}^{\mathcal{B}_A}$.

4.3.4. Operador: Vector de velocidad de un punto respecto a una referencia

Este operador calcula la velocidad de un punto del árbol de puntos respecto a una referencia (pareja formada por un punto del árbol de puntos y una base del árbol de bases). Ambos puntos han de estar en la misma rama del árbol de puntos.

El cálculo del vector de velocidad se realiza por medio de la ecuación de composición de velocidades (Ecuación 2.9). Así pues, el operador recorre la ci-

tada rama,calcula los vectores de velocidad intermedios según la citada ecuación y realiza la suma de estos.

Si el modificador es "gravedad UP" el resultado es un vector cuyas componentes están representadas en la base que más arriba esté de todo el conjunto de bases asociadas a los vectores intermedios.

En cambio, si el modificador es "gravedad DOWN" el resultado es un vector cuyas componentes están representadas en la base que más abajo esté de todo el conjunto de bases asociadas a los vectores intermedios.

4.3.4.1. Descripción

La velocidad de un punto P, definido a través de un vector $\mathbf{r}^P_{O_{R_n}}$ desde el punto O_{R_n} de la referencia R_n, con respecto a la una referencia R_0 se calcula aplicando la siguiente ecuación de forma recursiva:

$$\mathbf{v}^P_{R_{n-1}} = \mathbf{v}^P_{R_n} + \boldsymbol{\omega}^{R_n}_{R_{n-1}} \wedge \mathbf{r}^P_{O_{R_n}} + \mathbf{v}^{O_{R_n}}_{R_{n-1}} \tag{4.18}$$

Se ha de parar cuando la referencia R_n coincide con la referencia R_0. Nótese que el término $\mathbf{v}^P_{R_n}$ es el término que marca la recursividad.

Si el modificador es "gravedad DOWN" la suma se realiza:

$$\begin{aligned}\mathbf{v}^P_{R_0} = &\left(\left(\left(\mathbf{v}^{O_{R_n}}_{R_{n-1}} + \boldsymbol{\omega}^{R_n}_{R_{n-1}} \wedge \mathbf{r}^P_{O_{R_n}}\right) + \left(\mathbf{v}^{O_{R_{n-1}}}_{R_{n-2}} + \boldsymbol{\omega}^{R_{n-1}}_{R_{n-2}} \wedge \mathbf{r}^P_{O_{R_{n-2}}}\right)\right) + \ ...\ \right) + \\ &\left(\boldsymbol{\omega}^{R_1}_{R_0} \wedge \mathbf{r}^P_{O_{R_1}} + \mathbf{v}^{O_{R_1}}_{R_0}\right)\end{aligned} \tag{4.19}$$

Si el modificador es "gravedad UP" la suma se realiza:

$$\begin{aligned}\mathbf{v}^P_{R_0} = &\left(\left(\left(\boldsymbol{\omega}^{R_1}_{R_0} \wedge \mathbf{r}^P_{O_{R_1}} + \mathbf{v}^{O_{R_1}}_{R_0}\right) + \ ...\ \right) + \left(\mathbf{v}^{O_{R_{n-1}}}_{R_{n-2}} + \boldsymbol{\omega}^{R_{n-1}}_{R_{n-2}} \wedge \mathbf{r}^P_{O_{R_{n-2}}}\right)\right) + \\ &\left(\mathbf{v}^{O_{R_n}}_{R_{n-1}} + \boldsymbol{\omega}^{R_n}_{R_{n-1}} \wedge \mathbf{r}^P_{O_{R_n}}\right)\end{aligned} \tag{4.20}$$

De igual manera que en los casos anteriores, la suma de los vectores de velocidad intermedios asociados a cada punto se realiza dependiendo del sentido del modificador "gravedad" de forma que se maximice la parentesización.

4.3.4.2. Algoritmo

El operador se implementa mediante el algoritmo recursivo de la figura 4.7 donde se muestra en pseudo-código cómo se realiza el cálculo del vector velocidad de un punto P_B con respecto a la referencia R_A (formada por P_A y $\mathcal{B}_A$).

Al ser un algoritmo análogo, se ha de recorrer un rama del árbol sumando vectores intermedios, a los algoritmos anteriores (Position_Vector y Angular_Velocity_Vector) carece de interés su descripción en detalle.

En la primera iteración el algoritmo se ha de llamar como sigue:

`Velocity_Vector`(R_A, P_B,P_B,$\mathbf{v}_{IN}$)

donde $\mathbf{v}_{IN}$ es un vector vacío.

Al igual que en los dos algoritmo anteriores, la suma de vectores se realiza antes o después de llamar a la función recursiva `Velocity_Vector` según el sentido de la gravedad. Proceder de esta forma hace que la parentesización sea óptima.

4.3.4.3. Cadenas distintas

En el caso en que los puntos (P_B: el punto del cual se quiere conocer la velocidad y P_A el punto de la referencia R_A) se encuentren en ramas distintas, la solución se ha de realizar mediante los siguientes pasos:

1. Buscar el punto común en las dos ramas O_C y a continuación se define una referencia R_C formada por el ese punto O_C y la base $\mathcal{B}_A$ de la referencia R_A con respecto a la cual se quiere calcular la velocidad.
2. Calcular el vector $\mathbf{v}_{R_C}^{P_B}$.
3. Calcular el vector $\mathbf{v}_{R_C}^{P_A}$. Siendo P_A el punto de la referencia R_A.
4. El resultado deseado es $\mathbf{v}_{R_A}^{P_B} = \mathbf{v}_{R_C}^{P_B} - \mathbf{v}_{R_C}^{P_A}$

Entrada: P_B: Punto del que se quiere calcular la velocidad, R_A: referencia respecto a la cual se calcula la velocidad, P_O: Punto, $\mathbf{v}_{IN}$: Vector velocidad entrada
Salida: $\mathbf{v}_{OUT}$: Vector velocidad

Obtener el punto P_A de la referencia R_A
Obtener el punto P_{O-1}: punto anterior de P_O
Obtener el vector de $\mathbf{r}^{P_O}_{P_{O-1}}$: vector a través del cual se ha definido el punto P_O a partir del punto P_{O-1}.
Obtener la base asociada $\mathcal{B}_1$ a dicho vector $\mathbf{r}^{P_O}_{P_{O1}}$.

si P_A != P_{O1} **entonces**
- Obtener el punto P_{O-2}: punto anterior de P_{O-1}
- Obtener el vector de $\mathbf{r}^{P_{O-1}}_{P_{O-2}}$: vector a través del cual se ha definido el punto P_{O-1} a partir del punto P_{O-2}.
- Obtener la base asociada $\mathcal{B}_2$ a dicho vector $\mathbf{r}^{P_{O-1}}_{P_{O-2}}$.
- $\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_2} \leftarrow$ Angular_Velocity_Vector($\mathcal{B}_1$, $\mathcal{B}_2$)
- $\mathbf{r}^{P_O}_{P_{O1}} \leftarrow$ Position_Vector(P_{O1}, P_0)
- Obtener $\mathbf{v}^{R_1}_{P_O}$ como derivada de las componentes de $\mathbf{r}^{P_O}_{P_{O1}}$ respecto al tiempo (por estar las componentes del vector expresadas en $\mathcal{B}_1$). R_1 es la referencia formada por P_{O1} y $\mathcal{B}_1$
- **si** *GRAVEDAD HACIA ARRIBA* **entonces**
  - $\mathbf{v}_{AUX} \leftarrow$ Velocity_Vector(R_A, P_B,P_{O1},$\mathbf{v}_{IN}$)
- $\mathbf{v}_{OUT} \leftarrow \mathbf{v}_{AUX} + \mathbf{v}^{R_1}_{P_O} + (\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_2} \wedge \mathbf{r}^{P_O}_{P_{O1}})$
- **si** *GRAVEDAD HACIA ABAJO* **entonces**
  - $\mathbf{v}_{OUT} \leftarrow$ Velocity_Vector(R_A, P_B,P_{O1},$\mathbf{v}_{AUX}$)

si P_A = P_{O1} **entonces**
- Obtener la base $\mathcal{B}_A$ de la referencia R_A
- $\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_A} \leftarrow$ Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_1$)
- $\mathbf{r}^{P_B}_{P_A} \leftarrow$Position_Vector($P_A$, P_B)
- Obtener $\mathbf{v}^{R_A}_{P_O}$ como derivada de las componentes de $\mathbf{r}^{P_O}_{P_A}$ respecto al tiempo (por estar las componentes del vector expresadas en $\mathcal{B}_A$). R_A es la referencia formada por P_A y $\mathcal{B}_A$
- $\mathbf{v}_{OUT} \leftarrow \mathbf{v}^{R_A}_{P_O} + (\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_A} \wedge \mathbf{r}^{P_B}_{P_A}$

devolver $\mathbf{v}_{OUT}$

Figura 4.7.: Velocity_Vector(R_A, P_B,P_{O1},$\mathbf{v}_{IN}$)

4.3.4.4. Velocidad de un punto como perteneciente a un sólido respecto a un referencia

El algoritmo de la figura 4.8 muestra en pseudo-código cómo se realiza el cálculo de del vector velocidad de un punto P_B como perteneciente a un sólido S_C respecto a un referencia R_A.

Entrada: P_B: Punto del que se quiere calcular la velocidad. R_A: referencia respecto a la cual se calcula la velocidad,S_C:Sólido al cual pertenece el P_B
Salida: $\mathbf{v}_{OUT}$: Vector velocidad
Definir los vectores vacíos **r**,$\boldsymbol{\omega}$ y **v** (necesario para los algoritmos recursivos)

Obtener la base $\mathcal{B}_A$ de la referencia R_A
Obtener la base $\mathcal{B}_C$ del sólido S_C
$\boldsymbol{\omega}_{\mathcal{B}_A}^{\mathcal{B}_C} \leftarrow$ Angular_Velocity_Vector($\mathcal{B}_A$, $\mathcal{B}_C$,$\boldsymbol{\omega}$)

Obtener el punto B_C del sólido S_C
$\mathbf{r}_{B_C}^{P_B} \leftarrow$ Position_Vector(B_C,P_B,**r**)

$\mathbf{v}_{R_A}^{P_C} \leftarrow$Velocity_Vector($R_A$, P_C,P_c,**v**)
$\mathbf{v}_{OUT} \leftarrow \mathbf{v}_{R_A}^{P_C} + \boldsymbol{\omega} \wedge \mathbf{r}_{B_C}^{P_B}$
devolver $\mathbf{v}_{OUT}$

Figura 4.8.: Velocity_Vector(R_A, P_B, S_C)

4.4. Tensores no cartesianos

En el campo de la DSM, generalmente, los tensores son de rango dos[8] y caracterizan a la inercia rotacional de un sólido rígido. Estos tensores pertenecen a un grupo al que se le da el nombre general de "tensores cartesianos".

[8]Nótese que los escalares son tensores de rango 0 y los vectores son tensores de rango 1. En esta tesis, al referirse a tensor, se hace referencia a los de rango dos

4.4.0.5. Tensores cartesianos

Se entiende por tensor cartesiano una *aplicación lineal* en el espacio euclídeo 3D de la mecánica . Por ejemplo, se entiende por tensor de inercia ($\mathbf{I}_B^S$) a la aplicación lineal que transforma el vector velocidad angular $\boldsymbol{\omega}_{RI}^S$ del sólido S, respecto a la referencia RI, en el vector momento cinético $\mathbf{h}_B^S$ del sólido respecto del punto B

Las reglas de cambio de base son las correspondientes al espacio $\mathbb{R}^3$:

Sean $\{\mathbf{T}\}_{\mathcal{B}_1}$ y $\{\mathbf{T}\}_{\mathcal{B}_2}$ las componentes del tensor cartesiano $\mathbf{T}$ representadas respectivamente en las bases $\mathcal{B}_1$ y $\mathcal{B}_2$.

Sea $\mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}$ la matriz de cambio de base que transforma las coordenadas de un elemento en la base $\mathcal{B}_2$ sobre la base $\mathcal{B}_1$, entonces:

$$\{\mathbf{T}\}_{\mathcal{B}_2} = \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}{}^{\mathsf{T}} \{\mathbf{T}\}_{\mathcal{B}_1} \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}.$$

4.4.1. Matrices jacobianas como aplicaciones lineales

Muchas de las matrices jacobianas que aparecen en las distintas formulaciones de la DSM también tienen naturaleza tensorial. Por ejemplo, es el caso de las matrices jacobianas necesarias en el cálculo de las ecuaciones dinámicas mediante el PPV (Ecuación 2.35). En esta ecuación los términos $\mathbf{v}_{\dot{q}}$ o $\boldsymbol{\omega}_{\dot{q}}$ son tensores.

$\mathbf{v}_{\dot{q}}$ o $\boldsymbol{\omega}_{\dot{q}}$ son las matrices jacobianas asociadas a una aplicación lineal entre dos espacios vectoriales, el espacio cartesiano y un espacio no cartesiano, el de las velocidades generalizadas. Son tensores de rango dos cuya dimension es $3\ x\ n$ (donde n es el número de velocidades generalizadas).

4.4.2. Tensores no cartesianos. Cambios de base

Para trabajar con las matrices jacobianas, aprovechando su naturaleza de tensor, se propone una generalización del elemento tensor de rango dos. El tensor cartesiano puede ser entendido como un caso particular.

Cuando un tensor no es cartesiano las reglas de cambio de base son diferentes:

Sea $\mathbf{v}_{\dot{q}}$ el tensor asociado (la matriz asociada) a una aplicación lineal que va del espacio vectorial V, espacio euclídeo de la mecánica, al espacio vectorial $\dot{Q}$, espacio vectorial de las velocidades generalizadas.

Sean $\mathcal{B}_{V1}$ y $\mathcal{B}_{V2}$ dos bases ortonormales de V y sean $\mathcal{B}_{Q1}$ y $\mathcal{B}_{Q2}$ dos bases del espacio vectorial $\dot{Q}$.

Sea la matriz $\mathbf{R}_{\mathcal{B}_{V1}}^{\mathcal{B}_{V2}}$ la matriz de cambio de base de $\mathcal{B}_{V2}$ a $\mathcal{B}_{V1}$ y sea la matriz $\mathbf{R}_{\mathcal{B}_{Q1}}^{\mathcal{B}_{Q2}}$ la matriz de cambio de base de $\mathcal{B}_{Q2}$ a $\mathcal{B}_{Q1}$, la relación entre los tensores asociados a la aplicación lineal $\mathbf{v}_{\dot{q}}$ según la base considerada es la siguiente:

$$\mathbf{v}_{\dot{q}\,\mathcal{B}_{V2}}^{\mathcal{B}_{Q2}} = \left(\mathbf{R}_{\mathcal{B}_{V1}}^{\mathcal{B}_{V2}}\right)^{\mathsf{T}} \cdot \mathbf{v}_{\dot{q}\,\mathcal{B}_{V1}}^{\mathcal{B}_{Q1}} \cdot \mathbf{R}_{\mathcal{B}_{Q1}}^{\mathcal{B}_{Q2}} \tag{4.21}$$

En el contexto de la DSM las bases $\mathcal{B}_{V1}$ y $\mathcal{B}_{V2}$ son las bases definidas en el espacio cartesiano y las bases $\mathcal{B}_{Q1}$ y $\mathcal{B}_{Q2}$ pueden ser entendidas como dos parametrizaciones distintas del mismo sistema multicuerpo.

Esta tesis, se va a centrar en el caso en que únicamente existe una parametrización del sistema multicuerpo a estudio. Así pues, los cambios parametrización no están contemplados y la ecuación anterior, por simplificación, se puede escribir como:

$$\mathbf{v}_{\dot{q}\,\mathcal{B}_{V2}} = \left(\mathbf{R}_{\mathcal{B}_{V1}}^{\mathcal{B}_{V2}}\right)^{\mathsf{T}} \cdot \mathbf{v}_{\dot{q}\,\mathcal{B}_{V1}} \tag{4.22}$$

Nótese que al ser uno de los espacios vectoriales constante, el cambio de base asociado al espacio cartesiano únicamente se ha de hacer pre-multiplicando por la izquierda.

4.4.2.1. Ejemplo

Sea el vector velocidad de un punto respecto a una referencia inercial, $\mathbf{v}$, la matriz jacobiana de dicho vector respecto al vector de velocidades generalizadas $\dot{\boldsymbol{q}}_1$ se escribe:

$$\mathbf{v}_{\dot{\boldsymbol{q}}_1} = \frac{\partial \mathbf{v}}{\partial \dot{\boldsymbol{q}}_1} \tag{4.23}$$

Sea $\mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_1}$ la matriz de rotación que determina, a partir de las componentes en la base $\mathcal{B}_1$, las componentes de un vector o tensor en la base $\mathcal{B}_2$, entonces se tiene que:

$$\left\{\mathbf{v}_{\dot{\boldsymbol{q}}_1}\right\}_{\mathcal{B}_2} = \left\{\frac{\partial \mathbf{v}}{\partial \dot{\boldsymbol{q}}_1}\right\}_{\mathcal{B}_2} = \mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_1} \left\{\frac{\partial}{\partial \dot{\boldsymbol{q}}_1}\mathbf{v}\right\}_{\mathcal{B}_1} = \mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_1} \left\{\mathbf{v}_{\dot{\boldsymbol{q}}_1}\right\}_{\mathcal{B}_1} \tag{4.24}$$

El cambio de base en el espacio cartesiano del tensor asociado a una aplicación lineal (entre dicho espacio cartesiano y el de las velocidades generalizadas) se ha de realizar pre-multiplicando por la matriz de cambio de base.

Con la intención de completar el ejemplo y aunque ya se ha comentado que en esta tesis no se habla de distintas paremetrizaciones, se muestra además cómo se realiza un cambio de parametrización con los tensores definidos en este ejemplo.

Sea $\mathbf{R}_{\mathcal{B}_{\mathcal{Q}_2}}^{\mathcal{B}_{\mathcal{Q}_1}}$ la matriz de cambio de base, cambio de parametrización, que transforma las velocidades generalizadas $\dot{\boldsymbol{q}}_1$ a la parametrización asociada a las velocidades generalizadas $\dot{\boldsymbol{q}}_2$, entonces el tensor asociado a la aplicación lineal en la nueva parametrización es:

$$\left\{\mathbf{v}_{\dot{\boldsymbol{q}}_2}\right\}_{\mathcal{Q}_2} = \left\{\mathbf{v}_{\dot{\boldsymbol{q}}_1}\right\}_{\mathcal{Q}_1} \mathbf{R}_{\mathcal{B}_{\mathcal{Q}_2}}^{\mathcal{B}_{\mathcal{Q}_1}} \tag{4.25}$$

4.5. Cálculo de la matriz de masa M

En esta sección se presenta un método para el cálculo de la matriz de masa **M** de un sistema basado en el PPV.

Este método saca provecho las técnicas de atomización "sobre la marcha" y de derivación de expresiones atomizadas vistas en esta tesis.

Del mismo modo, para los cálculos de vectores de posición y de velocidad se utilizan los operadores cinemáticos descritos en esta tesis. De esta forma, el método para el cálculo de la matriz de masa **M** es válido independientemente de la topología del sistema.

Así mismo, como ya ha quedado justificado en la sección 4.2 el modificador "gravedad" ha de ser "gravedad UP".

Combinando estas las técnicas de atomización, los operadores cinemáticos y el modificador "gravedad" se asegura que la matriz resultado está óptimamente atomizada (parentesización óptima) y que la reciclabilidad de átomos es máxima. Además la combinación de estas técnicas permite que el cálculo de esta matriz se realice más rápido que si se realiza como se propone en 2.7.2.

4.5.1. PPV: Torsores

Sea S_i un sólido genérico, donde B_i es un punto cualquiera que pertenece a dicho sólido, al que se le aplica un torsor $\left[\mathbf{f}_i^{\mathsf{T}} \quad \mathbf{m}_i^{\mathsf{T}}\right]^{\mathsf{T}}$ sobre dicho punto B_i.

Según el PPV (2.3.1.4) la contribución $\mathbf{e}_i$ de dicho torsor a las ecuaciones de la dinámica $\mathbf{e}$ se puede escribir como:

$$\mathbf{e}_i = \mathbf{f}_i \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} + \mathbf{m}_i \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} = \mathbf{0} \tag{4.26}$$

4.5.2. Torsor de inercia de D'Alembert

Sean, para este mismo sólido genérico, G_i su centro de gravedad, m_i la masa y $\mathbf{I}_{B_i}^{S_i}$ el tensor de inercia definido en el punto B_i del sólido.

Mediante las ecuaciones de Newton-Euler (2.28) se plantean las fuerzas y momentos de inercia de D'Alembert:

$$\begin{aligned} \mathcal{F}^i &= -m_i \mathbf{a}_{RI}^{G_i} \\ \mathcal{M}_{B_i}^i &= -\left.\frac{d\mathbf{h}_{B_i}^{S_i}}{dt}\right|_{RI} - m_i \mathbf{r}_{B_i}^{G_i} \wedge \mathbf{a}_{RI}^{B_i} \end{aligned} \tag{4.27}$$

La contribución $\mathbf{e}_i$ debida al torsor de inercia del sólido S_i a las ecuaciones dinámicas $\mathbf{e}$ es:

$$\mathbf{e}_i = \mathcal{F}^i \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} + \mathcal{M}_{B_i}^i \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \tag{4.28}$$

4.5.2.1. Regla de la cadena

Sea $f = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$ una función genérica y derivable; sean $\boldsymbol{q}(t)$ el vector de coordenadas generalizadas, $\dot{\boldsymbol{q}}(t)$ el vector de velocidades generalizadas y t el tiempo, entonces según la regla de cadena la derivada temporal de dicha función se puede escribir:

$$\frac{df}{dt} = \frac{\partial f}{\partial \boldsymbol{q}}\frac{d\boldsymbol{q}}{dt} + \frac{\partial f}{\partial \dot{\boldsymbol{q}}}\frac{d\dot{\boldsymbol{q}}}{dt} + \frac{\partial f}{\partial t} = \frac{\partial f}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial f}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial f}{\partial t} \tag{4.29}$$

Así pues, aplicando la regla de la cadena, la aceleración del centro de masas (G_i) del sólido respecto a una referencia inercial (RI) en un sistema expresado por las coordenadas generalizadas $\boldsymbol{q} = [q_1, q_2, ...q_n]^\mathsf{T}$ se puede escribir como[9]

$$\mathbf{a}_{RI}^{G_i} = \frac{\partial \mathbf{v}_{RI}^{G_i}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{G_i}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{G_i}}{\partial t} \tag{4.30}$$

[9]Nótese que las derivadas parciales, según lo visto en 4.4, son tensores de rango dos. Por lo tanto, las operaciones que en esta sección puedan aparecer entre tensores se han de realizar según las reglas de cambio de base para aplicaciones lineales explicadas en la sub-sección 4.4.2.

Por otro lado, según la "Regla de derivación de vectores en orientaciones móviles" la derivada temporal del momento cinético de un sólido genérico S_i respecto a la referencia inercial (RI) es:

$$\left.\frac{d\mathbf{h}_{B_i}^{S_i}}{dt}\right|_{RI} = \frac{d\mathbf{h}_{B_i}^{S_i}}{dt} + \boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{h}_{B_i}^{S_i} \tag{4.31}$$

Si de la ecuación anterior se toma el primer sumando y se aplica la regla de la cadena se tiene que:

$$\frac{d\mathbf{h}_{B_i}^{S_i}}{dt} = \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial t} \tag{4.32}$$

A partir de estos elementos se pueden definir el vector de fuerzas y de momentos de inercia.

4.5.2.2. Vector fuerza de inercia de D'Alembert

Como los puntos B_i (un punto general de sólido) y G_i (el centro de masa del sólido) son puntos pertenecientes cinemáticamente al sólido, existe una relación directa entre sus aceleraciones:

$$\mathbf{a}_{RI}^{G_i} = \mathbf{a}_{RI}^{B_i} + \dot{\boldsymbol{\omega}}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i} + \boldsymbol{\omega}_{RI}^{S_i} \wedge \left(\boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i}\right) \tag{4.33}$$

Aplicando la regla de la cadena a los términos $\mathbf{a}_{RI}^{B_i}$ y $\dot{\boldsymbol{\omega}}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i}$, la aceleración $\mathbf{a}_{RI}^{G_i}$ puede escribirse como:

$$\begin{aligned}\mathbf{a}_{RI}^{G_i} = &\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial t}\mathbf{v}_{RI}^{B_i} + \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t}\right) \wedge \mathbf{r}_{B_i}^{G_i} \\ &+ \boldsymbol{\omega}_{RI}^{S_i} \wedge \left(\boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i}\right)\end{aligned} \tag{4.34}$$

Sustituyendo la aceleración anterior en el torsor de inercia (Ecuación: 4.27) se tiene que en el vector fuerza de inercia toma la siguiente forma:

$$\begin{aligned}\mathcal{F}^i = &- m_i \left[\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \ddot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial t} \mathbf{v}_{RI}^{B_i} \right] - m_i \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \ddot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \wedge \mathbf{r}_{B_i}^{G_i} \right] \\ &- m_i \left[\boldsymbol{\omega}_{RI}^{S_i} \wedge \left(\boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i} \right) \right]\end{aligned} \tag{4.35}$$

4.5.2.3. Vector momento de inercia de D'Alembert

El vector momento de inercia se define como sigue:

$$\begin{aligned}\mathcal{M}_{B_i}^i = - \Bigg[&\frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \dot{\boldsymbol{q}}} \ddot{\boldsymbol{q}} + \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial t} + \boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{h}_{B_i}^{S_i} \\ &+ m_i \mathbf{r}_{B_i}^{G_i} \wedge \left(\frac{\partial \mathbf{v}_{RI}^{B}}{\partial \dot{\boldsymbol{q}}} \ddot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B}}{\partial t} \mathbf{v}_{RI}^{B} \right) \Bigg]\end{aligned} \tag{4.36}$$

El tensor de inercia de un sólido rígido no depende ni de las coordenadas generalizadas, ni de las velocidad generalizadas, ni del tiempo. Así pues, las derivadas parciales del momento angular respecto de las coordenadas, velocidades y tiempo, se pueden escribir:

$$\begin{aligned}\frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \boldsymbol{q}} &= \mathbf{I}_{B_i}^{S_i} \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \\ \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial \dot{\boldsymbol{q}}} &= \mathbf{I}_{B_i}^{S_i} \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \\ \frac{\partial \mathbf{h}_{B_i}^{S_i}}{\partial t} &= \mathbf{I}_{B_i}^{S_i} \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t}\end{aligned} \tag{4.37}$$

Y sustituyendo las tres ecuaciones anteriores en la ecuación (4.36) elvector momento de inercia se escribe:

$$\begin{aligned}\mathcal{M}^i_{B_i} = &- \mathbf{I}^{S_i}_{B_i}\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} - \mathbf{I}^{S_i}_{B_i}\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} - \mathbf{I}^{S_i}_{B_i}\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial t} - \boldsymbol{\omega}^{S_i}_{RI} \wedge \left(\mathbf{I}^{S_i}_{B_i}\boldsymbol{\omega}^{S_i}_{RI}\right) \\ &- m_i \mathbf{r}^{G_i}_{B_i} \wedge \left[\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial t}\right]\end{aligned} \tag{4.38}$$

4.5.2.4. Matriz de masa M

Si se sustituyen $\mathcal{F}^i$ (Ec: 4.35) y $\mathcal{M}^i_{B_i}$ (Ec: 4.38) en la ecuación 4.28 y se toman solamente los términos que contribuyen a la matriz de masa $\mathbf{M}$, es decir, los términos que están multiplicando a las aceleraciones $\ddot{\boldsymbol{q}}$, la contribución $\mathbf{M}_i$ de cada sólido S_i a la matriz de masa es[10]:

$$\begin{aligned}\mathbf{M}^i = &- m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} - m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \wedge \mathbf{r}^{G_i}_{B_i}\right) \\ &- \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \mathbf{I}^{S_i}_{B_i}\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} - m_i \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\mathbf{r}^{G_i}_{B_i} \wedge \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)\end{aligned} \tag{4.39}$$

Para todo un sistema formado por n_S sólidos rígidos la matriz de masa $\mathbf{M}$ es la suma de las contribuciones realizadas por cada uno de los sólidos, es decir:

$$\mathbf{M} = \sum_{i=1}^{n_S} \mathbf{M}^i \tag{4.40}$$

4.5.3. Optimizaciones para el cálculo de la matriz M

En este apartado se propone una serie de cambios en la ecuación 4.39 para que el cálculo de la matriz de masa contenga expresiones simbólicas de tamaño óptimo (paretensización maximizada). Además, que en caso de usar técnicas de atomización, se obtenga una matriz simétrica en términos de átomos.

[10] El superíndice T indica que se está trasponiendo el elemento. Los paréntesis han sido introducidos para remarcar qué elemento se está trasponiendo

Para ayudar a la explicación, la ecuación anterior 4.39 se reescribe como sigue:

$$\mathbf{M}^i = \mathbf{M}^i_{\mathbf{vv}} + \mathbf{M}^i_{\mathbf{v}\omega} + \mathbf{M}^i_{\omega\omega} + \mathbf{M}^i_{\omega\mathbf{v}} \tag{4.41}$$

En donde:

$$\begin{aligned}
\mathbf{M}^i_{\mathbf{vv}} &= -m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \\
\mathbf{M}^i_{\mathbf{v}\omega} &= -m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \wedge \mathbf{r}^{G_i}_{B_i} \right) \\
\mathbf{M}^i_{\omega\omega} &= -\left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{I}^{S_i}_{B_i} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \\
\mathbf{M}^i_{\omega\mathbf{v}} &= -m_i \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\mathbf{r}^{G_i}_{B_i} \wedge \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)
\end{aligned} \tag{4.42}$$

4.5.3.1. Tensor de inercia

El tensor de inercia está definido de forma "natural" en la base asociada al sólido. Por lo tanto cualquier representación que se haga de este tensor en otra base introducirá términos que harán que el tamaño de las componentes del tensor sea mayor de lo que es originalmente.

En la ecuación 4.41 el sumando $\mathbf{M}^i_{\omega\omega}$ es:

$$\mathbf{M}^i_{\omega\omega} = -\left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{I}^{S_i}_{B_i} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \tag{4.43}$$

El modificador "gravedad UP", sin mayor cautela, asegura que el vector velocidad angular sea obtenido en la base asociada al sólido, que a su vez es la base natural del tensor de inercia. Por lo tanto, esta operación se realiza en la base asociada al sólido de manera que el tamaño de la expresiones simbólicas será óptimo, en comparación con cuando se realizara en cualquier otra base.

4.5.3.2. Términos iguales

El segundo término de la ecuación 4.41 es:

$$\mathbf{M}^i_{\mathbf{v}\boldsymbol{\omega}} = -m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \wedge \mathbf{r}^{G_i}_{B_i} \right) \tag{4.44}$$

Como el producto vectorial es anti-simétrico, puede ser reordenado de la siguiente manera:

$$\mathbf{M}^i_{\mathbf{v}\boldsymbol{\omega}} = m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\mathbf{r}^{G_i}_{B_i} \wedge \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right) = m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\widetilde{\mathbf{r}^{G_i}_{B_i}} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right) \tag{4.45}$$

Si se desarrolla el paréntesis y si se tiene en cuenta que una de las propiedades de una matriz anti-simétrica $\widetilde{A}$ es que $-\widetilde{A} = \widetilde{A}^{\mathsf{T}}$ se tiene que:

$$\mathbf{M}^i_{\mathbf{v}\boldsymbol{\omega}} = -m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \widetilde{\mathbf{r}^{G_i}_{B_i}}^{\mathsf{T}} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \tag{4.46}$$

Se puede observar que este término es el transpuesto de $\mathbf{M}^i_{\boldsymbol{\omega}\mathbf{v}}$. Esto permite hacer ciertas simplificaciones en el cálculo simbólico: se calcula uno de estos dos términos y luego se obtiene el otro por transposición. Es decir:

$$\mathbf{M}^i_{\mathbf{v}\boldsymbol{\omega}} = {\mathbf{M}^i_{\boldsymbol{\omega}\mathbf{v}}}^{\mathsf{T}} \tag{4.47}$$

De esta forma, en primer lugar, se evita el tener que realizar productos y matrices jacobianas que ya están previamente calculados. Además se evita tener que atomizar de nuevo expresiones que ya están atomizadas. Por último, así se asegura que los dos términos están atomizados de igual manera.

4.5.3.3. Centro de masas

En el ámbito de la identificación de parámetros es común expresar el vector centro de gravedad en función de los primeros momentos de inercia, es decir:

$$\mathbf{r}_{B_i}^{G_i} = \left(\frac{mx_i}{m_i}, \frac{my_i}{m_i}, \frac{mz_i}{m_i} \right) \tag{4.48}$$

donde mx_i, my_iy mz_i son los primeros momentos de inercia del sólido S_i.

Al calcular $\mathbf{M}_{\mathbf{v}\boldsymbol{\omega}}^{i}$ se ha de realizar el producto entre la masa m_i y el vector $\mathbf{r}_{B_i}^{G_i}$ del sólido. Como se ve la masa del sólido correspondiente es susceptible de ser simplificada.

Al trabajar con técnicas de atomización "sobre la marcha", no se puede realizar esta operación de simplificación, ya que este producto no aparece explícitamente (la masa aparece en una expresión atomizada y las componentes del vector en otras). Para simplificar estos términos, se debería realizar una operación de des-atomización, la simplificación y a continuación una atomización del resultado.

La solución tomada es reordenar el producto, priorizando (representado mediante el uso de paréntesis) el sub-producto $m_i \, \widetilde{\mathbf{r}_{B_i}^{G_i}}$. De esta forma se asegura que primero se realiza esta operación y a continuación se atomiza.

Así pues, al ser la masa un escalar, la matriz $\mathbf{M}_{\mathbf{v}\boldsymbol{\omega}}^{i}$ se puede escribir como:

$$\mathbf{M}_{\mathbf{v}\boldsymbol{\omega}}^{i} = -\left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(m_i \, \widetilde{\mathbf{r}_{B_i}^{G_i}} \right)^{\mathsf{T}} \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \tag{4.49}$$

4.5.3.4. Orden de las operaciones. Simetría

Debido a las técnicas de atomización "sobre la marcha", al calcular la matriz de masa se ha detener en cuenta el orden en que se realizan algunas de las operaciones. Al ser la matriz de masa es simétrica, si las operaciones no se realizan en orden correcto, puede darse el caso de que el elemento en posición

(i, j) no esté atomizado de igual forma que el elemento en posición (j, i).

Se ha de prestar atención a los cuatro sumandos vistos en 4.41.

1. Término $\mathbf{M}^i_{\mathbf{vv}}$:

$$\mathbf{M}^i_{\mathbf{vv}} = -m_i \left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \tag{4.50}$$

   Este producto está compuesto por tres elementos, un escalar, una matriz y su transpuesta. Al tratarse de un producto de una matriz por su transpuesta (y por un escalar) el resultado ha de ser una matriz simétrica.

   En caso de realizar este producto tal y como está en la ecuación anterior, con técnicas de atomización "sobre la marcha" realizaría primero el producto entre el escalar y la matriz transpuesta y el resultado sería atomizado. A continuación, se realizaría el producto del resultado anterior por la matriz para posteriormente ser atomizado. El resultado atomizado no es simétrico en términos de átomos, si bien el resultado desatomizado sí es una matriz simétrica,

   La forma más sencilla de asegurar que este elementos es simétrico es priorizando las operaciones entre la matriz transpuesta y la matriz sin transponer. Es decir:

$$\mathbf{M}^i_{\mathbf{vv}} = -m_i \left(\left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right) \tag{4.51}$$

2. Términos $\mathbf{M}^i_{\mathbf{v}\omega}$ y $\mathbf{M}^i_{\omega\mathbf{v}}$.

   Al tratarse de la suma de una matriz y de su transpuesta (ya se ha visto que $\mathbf{M}^i_{\mathbf{v}\omega} = \mathbf{M}^i_{\omega\mathbf{v}}{}^{\mathsf{T}}$), el resultado ha de ser una matriz simétrica. Para que, mediante técnicas de atomización "sobre la marcha" el resultado sea simétrico en términos de átomos, esta suma ha de ser priorizada. Así pues, esto se asegura sin más que introduciéndolos entre paréntesis:

$$\left(\mathbf{M}^i_{\mathbf{v}\omega} + \mathbf{M}^i_{\omega\mathbf{v}}{}^{\mathsf{T}} \right) \tag{4.52}$$

3. Término $\mathbf{M}^i_{\omega\omega}$:

$$\mathbf{M}^i_{\omega\omega} = \left(\frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \mathbf{I}^{S_i}_{B_i} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \tag{4.53}$$

Este es un producto triple formado por una matriz transpuesta, el tensor de inercia y una matriz. Al ser las matrices iguales y el tensor de inercia simétrico, el resultado ha de ser simétrico. En caso de utilizar atomización "sobre la marcha" el resultado no es simétrico en átomos, pues se realiza primero el producto de la matriz transpuesta por el tensor, se atomiza, y el resultado se multiplica por la matriz, para de nuevo ser atomizado.

Nótese que este caso no se puede priorizar el producto entre las dos matrices, como se ha hecho en el primer caso.

Así pues, se propone un algoritmo que aprovecha la simetría del resultado y realiza esta operación de forma óptima.

En este producto, el elemento (i, j) será el resultado de multiplicar la fila i-ésima de la matriz transpuesta por el tensor de inercia y por la columna j-ésima de la matriz sin transponer. El resultado ha de ser simétrico, así pues se sabe que el elemento (j, i) será igual al (i, j).

El algoritmo de la figura 4.9 muestra como se realiza esta operación. En donde:

- **Om_dq** es la matriz $\left\{\frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{\omega}^{S_i}_{RI}\right\}_{\mathcal{B}_{S_i}}$
- **Om_dqT** es la matriz $\left\{\frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{\omega}^{S_i}_{RI}\right\}^{\mathsf{T}}_{\mathcal{B}_{S_i}}$
- **I** es el tensor de inercia $\left\{\mathbf{I}^{S_i}_{B_i}\right\}_{\mathcal{B}_{S_i}}$
- **Om_dq**$(i, *)$ hace referencia a la fila i-ésima de la matriz.
- **Om_dq**$(*, j)$ hace referencia a la columna j-ésima de la matriz.
- filas$(\cdot)$ hace referencia al número de filas de la matriz entre paréntesis.

sis.

- columnas(·) hace referencia al número de columnas de la matriz entre paréntesis.
- $\mathbf{M}_{\omega\omega}$ es la matriz resultado.

```
para i ← 1 to filas(Om_dqT) hacer
    RowI ← Om_dqT(i, ∗) · I
    para j ← i to columnas(Om_dq) hacer
        RowICol ← RowI · Om_dq(∗, j)
        si j = i entonces
            Mωω(i, j) ← RowICol
        en otro caso
            Mωω(i, j) ← RowICol
            Mωω(j, i) ← RowICol
devolver Mωω
```

Figura 4.9.: Cálculo de la matriz auxiliar $\mathbf{M}_{\omega\omega}$

4.5.3.5. Matriz M

De este modo, teniendo en cuenta las mejoras propuestas, la matriz de masa de un sistema formado por n_S sólidos se puede escribir como:

$$\mathbf{M} = \sum_{i=1}^{n_S} \mathbf{M}^i = \sum_{i=1}^{n_S} \left[\mathbf{M}^i_{\mathbf{vv}} + \left(\mathbf{M}^i_{\mathbf{v}\omega} + {\mathbf{M}^i_{\mathbf{v}\omega}}^{\mathsf{T}} \right) + \mathbf{M}^i_{\omega\omega} \right] \tag{4.54}$$

Donde:

$$\begin{aligned} \mathbf{M}^i_{\mathbf{vv}} &= -m_i \left(\left(\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \right) \\ \mathbf{M}^i_{\mathbf{v}\omega} &= -{\frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}}}^{\mathsf{T}} \left(m_i \widetilde{\mathbf{r}^{G_i}_{B_i}} \right)^{\mathsf{T}} \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \end{aligned} \tag{4.55}$$

Y donde $\mathbf{M}^i_{\omega\omega}$ se ha de obtener según el algoritmo descrito en el apartado anterior.

4.5.4. Optimización de M basada en coordenadas sin inercia asociada

Basándose en la existencia en ciertos sistemas multicuerpo de coordenadas sin inercia asociada (Sección 2.8) se puede optimizar, para este tipo de sistemas, el cálculo de la matriz de masa.

Ya que esas coordenadas no influyen en la energía cinética del sistema, como ya se ha visto en la citada sub-sección, las correspondientes filas y columnas de la matriz de masa son cero.

Sabiendo *a priori* qué elementos van a ser cero, se puede evitar su cálculo y crear una matriz de masa de tamaño inferior que contenga toda la información relativa a la dinámica del sistema. A continuación completarla con las filas y columnas de ceros correspondientes a las citadas coordenadas sin inercia asociada.

Según el método para el cálculo de la matriz de masa que se han presentado, se han de calcular matrices jacobianas respecto a las velocidades generalizadas. Si solo se hace este cálculo respecto a las velocidades generalizadas de coordenadas que sí influyen en la dinámica el tiempo de computación puede verse reducido de manera notable.

4.6. Cálculo del vector δ

Análogamente a como se ha procedido con la matriz de masa se propone un método para el cálculo del vector δ, término independiente del sistema multicuerpo (Sección 2.4).

De igual manera, se utilizan las técnicas de atomización y de derivación de expresiones atomizadas. También los operadores cinemáticos para los cálculos de vectores de posición y de velocidad. De igual manera, como ya ha quedado justificado en la sección 4.2, el modificador "gravedad" ha de ser "gravedad UP". Aun así, existe una excepción, asociada a los torsores de gravedad que se verá más adelante.

Combinando estas las técnicas de atomización, los operadores cinemáticos y el modificador "gravedad UP" se asegura que el vector resultado está óptimamente atomizado y que la reciclabilidad de átomos es máxima. Además la combinación de estas técnicas hace que el cálculo de este vector se realice más rápido que si se realiza como se propone en 2.7.2.

4.6.1. Vector δ

Este vector esta formado dos grupos de términos: términos que provienen de los torsores de inercia de D'Alembert y los que provienen del resto de torsores (debidos a la fuerza de la gravedad, a fuerzas constitutivas y/o a fuerzas exteriores). Por lo tanto, el cálculo de este vector se ha dividido en dos partes, las contribuciones de las fuerzas y momentos de inercia y las contribuciones del resto de torsores.

Se sustituyen $\mathfrak{F}^i$ (ecuación 4.35) y $\mathfrak{M}^i_{B_i}$ (ecuación 4.38) en la ecuación 4.26 y se toman solamente los términos que contribuyen al vector δ, es decir, todos los términos que no están multiplicando a las aceleraciones $\ddot{\boldsymbol{q}}$. Además, se toman las contribuciones de cada uno de los torsores W de distinto tipo.

Así se tiene que para un sistema formado por n_S solidos y por w torsores, donde w_i es el número de torsores que actúan sobre el sólido S_i, el vector δ se puede obtener de la siguiente manera:

$$
\begin{aligned}
\delta = &- \sum_{i=1}^{n_S} \sum_{j=1}^{w_i} \left[\left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{f}^{S_i}{}_j + \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{m}_{B_i j}^{S_i} \right] \\
&+ \sum_{i=1}^{n_S} \left[m_i \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[m_i \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \wedge \mathbf{r}_{B_i}^{G_i} \right] \\
&+ \sum_{i=1}^{n_S} \left[m_i \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\boldsymbol{\omega}_{RI}^{S_i} \wedge (\boldsymbol{\omega}_{RI}^{S_i} \wedge \mathbf{r}_{B_i}^{G_i}) \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} m_i \widetilde{\mathbf{r}_{B_i}^{G_i}} \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{I}_{B_i}^{S_i} \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{I}_{B_i}^{S_i} \boldsymbol{\omega}_{RI}^{S_i} \right]
\end{aligned}
\tag{4.56}
$$

4.6.2. Optimizaciones en el cálculo

Análogamente a como ocurre con el cálculo de la matriz de masa, el cálculo de este vector es susceptible de optimización, de manera que se generen expresiones óptimamente atomizadas y parentesizadas. Estas optimizaciones mejoran además la velocidad de cálculo de este término.

4.6.2.1. Bases de proyección

Como en el caso de la matriz de masa se pueden buscar bases de proyección para alguno de los sumandos donde la expresión tenga un tamaño menor. En los dos últimos sumandos, aparecen únicamente el tensor de inercia del sólido y el vector velocidad angular del mismo respecto a la base absoluta. Estos son:

$$\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \mathbf{I}_{B_i}^{S_i} \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t}\right) \tag{4.57}$$

y

$$\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{I}_{B_i}^{S_i} \boldsymbol{\omega}_{RI}^{S_i} \tag{4.58}$$

Sin más cautesl, el uso del modificador "gravedad UP" asegura que estas operaciones se realicen en la base $\mathcal{B}_{S_i}$ asociada a cada sólido y base natural del tensor $\mathbf{I}_{B_i}^{S_i}$. En esta base, el resultado es óptimo en ya el número de cambio de base realizado es mínimo.

4.6.2.2. Vector centro de gravedad

De igual manera que como se ha detallado en el apartado 4.5.3.3, el resultado del producto entre el vector del centro de gravedad de un sólido y la masa del sólido es susceptible de simplificación.

Todos los sumandos en los que aparece este producto $m_i\ \mathbf{r}_{B_i}^{G_i}$, este se ha de priorizar (mediante paréntesis) facilitando así, en caso de usar técnicas de atomización "sobre la marcha" el trabajo de simplificación.

4.6.2.3. Torsor de gravedad

Sea $\mathbf{f}^g$ el vector de gravedad[11] definido de forma "natural" en la base de la referencia absoluta RI:

$$\{\mathbf{f}^g\}_{RI} = \begin{Bmatrix} 0 \\ 0 \\ -g \end{Bmatrix}_{RI} \tag{4.59}$$

[11]Nótese que se hace referencia a la gravedad como magnitud física

La contribución δ_i al vector δ debida a la gravedad actuante sobre el sólido S_i es:

$$\delta_i = \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{f}^g + \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{m}_{B_i}^g \tag{4.60}$$

Donde:

$$\mathbf{m}_{B_i}^g = m_i \mathbf{r}_{B_i}^{G_i} \wedge \mathbf{f}^g \tag{4.61}$$

Nótese que al definir el modificador "gravedad UP" el vector $\mathbf{f}^g$ ha de ser representado en la base asociada al tensor $\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}}$ que está por encima de la base de la referencia inercial RI. Representar el vector $\mathbf{f}^g$ en dicha base, en general hace que la expresión simbólica asociada a las componentes tenga un tamaño grande debido a los cambios de base.

En cambio, con el modificador "gravedad DOWN" la base asociada al tensor $\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}}$ es la base de la referencia inercial RI, lo mismo que el vector $\mathbf{f}^g$. Por lo tanto, el producto se puede realizar en esa base sin necesidad de realizar ningún cambio de base adicional.

Por lo tanto, este es uno de los casos en que se ha de utilizar el modificador "gravedad DOWN" si se desea que el tamaño de las expresiones sea óptimo.

4.6.2.4. Torsores de acciones exteriores

Análogamente a como ocurre con el torsor de gravedad, las contribuciones δ_i al vector δ debidas a los torsores de acciones exteriores han de ser calculadas con el modificador "gravedad DOWN" si se desea que el tamaño de las expresiones sea óptimo.

Este hecho es debido a que la base de proyección "natural" de estos torsores es la base de la referencia absoluta RI. La justificación vista en el apartado anterior es igualmente válida para este caso.

4.6.2.5. Vector δ optimizado

Es decir, para un sistema multicuerpo formado por un número S de sólidos y un número w de torsores el vector $\boldsymbol{\delta}$ se puede calcular como sigue:

$$
\begin{aligned}
\boldsymbol{\delta} = &- \sum_{i=1}^{n_S} \sum_{j=1}^{w_i} \left[\left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{f}^{S_i}{}_j + \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{m}_{B_i j}^{S_i} \right] \\
&+ \sum_{i=1}^{n_S} \left[m_i \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(m_i \widetilde{\mathbf{r}_{B_i}^{G_i}} \right)^{\mathsf{T}} \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \mathbf{v}_{RI}^{B_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(\widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \left(m_i \cdot \mathbf{r}_{B_i}^{G_i} \right) \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \left(m_i \widetilde{\mathbf{r}_{B_i}^{G_i}} \right) \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \mathbf{I}_{B_i}^{S_i} \left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial t} \right) \right] \\
&+ \sum_{i=1}^{n_S} \left[\left(\frac{\partial \boldsymbol{\omega}_{RI}^{S_i}}{\partial \dot{\boldsymbol{q}}} \right)^{\mathsf{T}} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{I}_{B_i}^{S_i} \boldsymbol{\omega}_{RI}^{S_i} \right]
\end{aligned}
\tag{4.62}
$$

4.7. Representación matricial

Las ecuaciones para la obtención de la matriz de masa (Ecuación 4.54) y para la obtención del vector de fuerzas generalizas (Ecuación 4.62) pueden ser escritas en forma matricial.

Las matrices jacobianas de los vectores de velocidad (absoluta y angular) respecto a las velocidades generalizas para un sólido i-ésimo pueden ser escritas como:

$$\begin{aligned} \mathbf{v}^i_{\dot{\boldsymbol{q}}} &= \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \\ \boldsymbol{\omega}^i_{\dot{\boldsymbol{q}}} &= \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \dot{\boldsymbol{q}}} \end{aligned} \tag{4.63}$$

Análogamente, las matrices jacobianas de los vectores de velocidad (absoluta y angular) respecto a las coordenadas generalizas para un sólido i-ésimo pueden ser escritas como:

$$\begin{aligned} \mathbf{v}^i_{\boldsymbol{q}} &= \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial \boldsymbol{q}} \\ \boldsymbol{\omega}^i_{\boldsymbol{q}} &= \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial \boldsymbol{q}} \end{aligned} \tag{4.64}$$

Por último, las matrices jacobianas de los vectores de velocidad (absoluta y angular) respecto al tiempo para un sólido i-ésimo pueden ser escritas como:

$$\begin{aligned} \mathbf{v}^i_{t} &= \frac{\partial \mathbf{v}^{B_i}_{RI}}{\partial t} \\ \boldsymbol{\omega}^i_{t} &= \frac{\partial \boldsymbol{\omega}^{S_i}_{RI}}{\partial t} \end{aligned} \tag{4.65}$$

Todas las matrices jacobianas anteriores pueden ser agrupadas y para cada sólido definir las siguientes matrices:

$$\mathbf{J}^i_{\dot{\boldsymbol{q}}} = \begin{bmatrix} \mathbf{v}^i_{\dot{\boldsymbol{q}}} \\ \boldsymbol{\omega}^i_{\dot{\boldsymbol{q}}} \end{bmatrix} \qquad \mathbf{J}^i_{\boldsymbol{q}} = \begin{bmatrix} \mathbf{v}^i_{\boldsymbol{q}} \\ \boldsymbol{\omega}^i_{\boldsymbol{q}} \end{bmatrix} \quad \text{y} \quad \mathbf{J}^i_{t} = \begin{bmatrix} \mathbf{v}^i_{t} \\ \boldsymbol{\omega}^i_{t} \end{bmatrix} \tag{4.66}$$

Así pues, el cálculo de la contribución $\mathbf{M}^i i$ a la matriz de masa de un sólido S_i queda de la siguiente manera:

$$\mathbf{M}^i = -\mathbf{J}^{i\,\mathsf{T}}_{\dot{\boldsymbol{q}}}\, \mathbf{M}^i_c\, \mathbf{J}^i_{\dot{\boldsymbol{q}}} \tag{4.67}$$

Donde:

$$\mathbf{M}_c^i = \begin{bmatrix} m_i \mathbf{1}_3 & m_i \widetilde{\mathbf{r}_{B_i}^{G_i}}^{\mathsf{T}} \\ m_i \widetilde{\mathbf{r}_{B_i}^{G_i}} & \mathbf{I}_{B_i}^{S_i} \end{bmatrix} \tag{4.68}$$

De análoga forma puede escribirse la contribución δ_i de cada sólido S_i al vector δ quedando este como:

$$\begin{aligned} \delta^i &= \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} \mathbf{M}_c^i \mathbf{J}_{\boldsymbol{q}}^i \dot{\boldsymbol{q}} + \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} \mathbf{M}_c^i \mathbf{J}_t^i + \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{b}^i - \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} \boldsymbol{w}_{B_i}^i \\ &= \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} (\mathbf{M}_c^i (\mathbf{J}_{\boldsymbol{q}}^i \dot{\boldsymbol{q}} + \mathbf{J}_t^i) + \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{b}^i - \boldsymbol{w}_{B_i}^i) \end{aligned} \tag{4.69}$$

Donde:

$$\mathbf{b}^i = \begin{bmatrix} \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} (m_i \mathbf{r}_{B_i}^{G_i}) \\ \mathbf{I}_{B_i}^{S_i} \boldsymbol{\omega}_{RI}^{S_i} \end{bmatrix} \tag{4.70}$$

Y en donde $\boldsymbol{w}_{B_i}^i$ es el torsor de las w fuerzas y momentos constitutivos, de gravedad y/o exteriores aplicados sobre el sólido S_i.

$$\boldsymbol{w}_{B_i}^i = \sum_{j=1}^{W} \begin{bmatrix} \mathbf{f}^j \\ \mathbf{m}_{B_i}^j \end{bmatrix} \tag{4.71}$$

4.8. Términos comunes entre $\mathbf{M}$ y δ, bucle de cálculo

Entre la ecuación para el cálculo de la matriz $\mathbf{M}$ (Ecuación: 4.54) y la ecuación para el cálculo del vector δ (Ecuación: 4.62) hay una serie de elementos comunes.

Estos pueden ser aprovechados con la idea de no repetir cálculos previamente realizados (reciclado de átomos) y de maximizar la parentesización.

El algoritmo 4.10 esquematiza para un sistema con S sólidos cómo realizar el cálculo de $\mathbf{M}$ y δ de forma más rápida y generando expresiones de menor

tamaño ya que se aprovecha la asociatividad de algunos de los cálculos.

para $i \leftarrow 0$ **hasta** S **hacer**

- Calcular $\mathbf{v}_{RI}^{B_i}$ y $\boldsymbol{\omega}_{RI}^{S_i}$
- Calcular $\mathbf{J}_{\dot{\boldsymbol{q}}}^i$, $\mathbf{J}_{\boldsymbol{q}}^i$ y $\mathbf{J}_t^i$
- Calcular $\mathbf{M}_c^i$ y $\mathbf{b}^i$
- $\mathbf{M}^i = -\mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}}\, \mathbf{M}_c^i\, \mathbf{J}_{\dot{\boldsymbol{q}}}^i$
- $\boldsymbol{\delta}^i = \mathbf{J}_{\dot{\boldsymbol{q}}}^{i\,\mathsf{T}} \left(\mathbf{M}_c^i \left(\mathbf{J}_{\boldsymbol{q}}^i\, \dot{\boldsymbol{q}} + \mathbf{J}_t^i \right) + \widetilde{\boldsymbol{\omega}_{RI}^{S_i}} \mathbf{b}^i - \boldsymbol{w}_{B_i}^i \right)$
- $\mathbf{M} \leftarrow \mathbf{M}^i$
- $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta}^i$

Figura 4.10.: Cálculo de $\mathbf{M}$ y $\boldsymbol{\delta}$

CAPÍTULO 5

EXPRESIONES TRIGONOMÉTRICAMENTE SIMPLIFICABLES

En el contexto de la DSM aparecen de forma natural expresiones susceptibles de ser simplificadas trigonométricamente (ESST). Estas conllevan un coste computacional añadido en la evaluación numérica de las mismas.

Las ESST pueden ser eliminadas, a priori, mediante los métodos de simplificación trigonométrica comúnmente implementados en la mayoría de paquetes del álgebra simbólica.

Cuando se utilizan técnicas de atomización es imposible detectar y simplificar estas expresiones sin una previa desatomización. Además, una vez simplificadas se requiere una nueva atomización. Todo ello incrementa el coste computacional del proceso simbólico.

En [2] (Sección 5.4) se aborda este problema proponiendo una serie de funciones que realizan las típicas simplificaciones trigonométricas (coseno y seno

de la suma/resta de ángulos, $\sin^2 + \cos^2 = 1$, ...) que se pueden dar en el contexto de la DSM. Estas funciones toman las expresiones en extenso (desatomizadas) y realizan las simplificaciones pertinentes. Conforme se van realizando las simplificaciones, los distintos resultado se van asignando a variables auxiliares (átomos). Esto incrementa el coste computacional y, lo que es más importante, hace que se pierda la atomización ya realizada. Se obliga a crear nuevas variables auxiliares donde, según la citada referencia, estas variables auxiliares pueden estar repetidas o incluso ser nulas.

En este capítulo se analizan las ESST en el contexto de la DSM, cuándo aparecen, por qué y de qué tipo son. Como resultado del análisis se proponen una serie de métodos para evitar que estas aparezcan, de forma que se evita por completo el proceso de simplificación trigonométrica. Esto implica un importante ahorro computacional en el procesamiento simbólico: evitando atomizar, desatomizar, la aparición de átomos superfluos,...

5.1. Expresión trigonométricamente simplificables con origen "vectorial"

El punto de partida de este estudio es la presentación [43] en el que se estudia el origen de las ETTS con origen vectorial.

Sean, por ejemplo, dos vectores donde cada uno de ellos ha sido representado en una base de proyección distinta. Se ha de realizar una operación binaria (suma/resta, producto escalar o producto vectorial) entre ellos, ambos vectores deberán ser proyectados en una *base común*. En esa *base común* la operación se realiza como se realizaría con matrices regulares. Es al realizar las proyecciones de los vectores en la *base común* cuando aparecen las ESST.

5.1.1. Ejemplo: producto escalar de dos vectores

A modo de ejemplo se analiza el producto escalar de dos vectores. Este mismo análisis es fácilmente extensible a el caso del producto vectorial o el producto entre tensores.

Sea $\boldsymbol{v} = \boldsymbol{v}^{\mathcal{B}_A} + \boldsymbol{v}^{\mathcal{B}_B}$, donde $\boldsymbol{v}^{\mathcal{B}_A}$ y $\boldsymbol{v}^{\mathcal{B}_B}$ son dos vectores cuyas componentes

están definidas en las bases $\mathcal{B}_A$ y $\mathcal{B}_B$ respectivamente.

Sea $\boldsymbol{u} = \boldsymbol{u}^{\mathcal{B}_B} + \boldsymbol{u}^{\mathcal{B}_C}$, donde $\boldsymbol{u}^{\mathcal{B}_B}$ y $\boldsymbol{u}^{\mathcal{B}_C}$ son dos vectores cuyas componentes están definidas en las bases $\mathcal{B}_B$ y $\mathcal{B}_C$ respectivamente.

Sea $\mathbf{R}_{\mathcal{B}_j}^{\mathcal{B}_i}$ la matriz de cambio de base que determina las componentes de un vector en la base $\mathcal{B}_j$ a partir de las componentes en la base $\mathcal{B}_i$.

Se denota $\{\boldsymbol{v}\}_{\mathcal{B}_i}$ a las componentes del vector $\boldsymbol{v}$ proyectadas sobre la base $\mathcal{B}_i$.

Para realizar el producto escalar de $\boldsymbol{v} \cdot \boldsymbol{u}$ se han de realizar los siguientes pasos:

1. Obtener las componentes de los vectores $\boldsymbol{v}$ y $\boldsymbol{u}$ proyectadas en una única base:

$$\begin{aligned} \{\boldsymbol{v}\}_{\mathcal{B}_B} &= \mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}\{\boldsymbol{v}^{\mathcal{B}_A}\}_{\mathcal{B}_A} + \{\boldsymbol{v}^{\mathcal{B}_B}\}_{\mathcal{B}_B} \\ \{\boldsymbol{u}\}_{\mathcal{B}_C} &= \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{u}^{\mathcal{B}_B}\}_{\mathcal{B}_B} + \{\boldsymbol{u}^{\mathcal{B}_C}\}_{\mathcal{B}_C} \end{aligned} \tag{5.1}$$

2. Transformar los vectores $\{\boldsymbol{v}\}_{\mathcal{B}_B}$ y $\{\boldsymbol{u}\}_{\mathcal{B}_C}$ a una *base común* (la base C)[1]

$$\begin{aligned} \{\boldsymbol{v}\}_{\mathcal{B}_C} &= \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{v}\}_{\mathcal{B}_B} = \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}\{\boldsymbol{v}^{\mathcal{B}_A}\}_{\mathcal{B}_A} + \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{v}^{\mathcal{B}_B}\}_{\mathcal{B}_B} \\ \{\boldsymbol{u}\}_{\mathcal{B}_C} &= \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{u}^{\mathcal{B}_B}\}_{\mathcal{B}_B} + \{\boldsymbol{u}^{\mathcal{B}_C}\}_{\mathcal{B}_C} \end{aligned} \tag{5.2}$$

3. Realizar el producto escalar:

$$\begin{aligned} &\{\boldsymbol{v}\}_{\mathcal{B}_C} \cdot \{\boldsymbol{u}\}_{\mathcal{B}_C} = \\ &\left(\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_A}\{\boldsymbol{v}^{\mathcal{B}_A}\}_{\mathcal{B}_A} + \mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{v}^{\mathcal{B}_B}\}_{\mathcal{B}_B}\right)^{\mathsf{T}}\left(\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{u}^{\mathcal{B}_B}\}_{\mathcal{B}_B} + \{\boldsymbol{u}^{\mathcal{B}_C}\}_{\mathcal{B}_C}\right) = \\ &\{\boldsymbol{v}^{\mathcal{B}_A}\}_{\mathcal{B}_A}^{\mathsf{T}}\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_C}\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{u}^{\mathcal{B}_B}\}_{\mathcal{B}_B} + \{\boldsymbol{v}^{\mathcal{B}_A}\}_{\mathcal{B}_A}^{\mathsf{T}}\mathbf{R}_{\mathcal{B}_A}^{\mathcal{B}_B}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_C}\{\boldsymbol{u}^{\mathcal{B}_C}\}_{\mathcal{B}_C} + \\ &\{\boldsymbol{v}^{\mathcal{B}_B}\}_{\mathcal{B}_B}^{\mathsf{T}}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_C}\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B}\{\boldsymbol{u}^{\mathcal{B}_B}\}_{\mathcal{B}_B} + \{\boldsymbol{v}^{\mathcal{B}_B}\}_{\mathcal{B}_B}^{\mathsf{T}}\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_C}\{\boldsymbol{u}^{\mathcal{B}_C}\}_{\mathcal{B}_C} \end{aligned} \tag{5.3}$$

En la ecuación anterior en los términos primero y tercero aparece el producto

[1] Se debe de destacar que sería mucho más sencillo proyectar todo en la base $\mathcal{B}_B$, pero se ha decidido hacerlo así para ilustrar la problemática más claramente

de una matriz de cambio de base por su transpuesta donde este producto se simplifica a la matriz identidad, concretamente $\mathbf{R}_{\mathcal{B}_B}^{\mathcal{B}_C}\mathbf{R}_{\mathcal{B}_C}^{\mathcal{B}_B} = \mathbf{1}$. Este producto de una matriz cambio de base por su matriz transpuesta es el origen de las ESST. Esto es debido a que si realiza el producto, en general, se obtiene una matriz con expresiones más o menos complejas simplificables a cero o a uno.

Este ejemplo se ha preparado para ver claramente cuándo aparecen este tipo de simplificaciones, en un caso general no suelen apreciarse tan explícitamente.

5.2. Nueva estructura simbólica de vector: Vector extendido

Con la idea de evitar que las ESST aparezcan se propone una nueva representación para los vectores. Esta nueva representación se ha denominado "vector extendido"[2].

Sea el vector v, que se obtiene como resultado de la suma de varios vectores cuyas componentes se conocen en distintas bases ($\mathcal{B}_1$, ..., $\mathcal{B}_i$, ..., $\mathcal{B}_N$):

$$v = v^{\mathcal{B}_1} + ... + v^{\mathcal{B}_i} + ... + v^{\mathcal{B}_N} \tag{5.4}$$

Se define el "vector extendido" como una matriz en donde cada columna i contiene la parte del vector v cuyas componentes están expresadas de forma "natural" en la base $\mathcal{B}_i$. N corresponde al número total de bases utilizadas en la formulación. Es decir:

$$[\![v]\!] = \left[\ v^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ v^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ v^{\mathcal{B}_N}\ \right] \tag{5.5}$$

Cada una de las columnas de esta matriz se denominará "componente del vector extendido" o simplemente "componente".

[2]En este documento cunado se haga referencia a un vector cartesiano, se hablará de "vector". Mientras que la nueva estructura siempre siempre se indicará con el apellido "extendido"

Esta nueva estructura es óptima en el sentido en que cada componente $\boldsymbol{v}^{\mathcal{B}_i}$ del vector extendido está proyectada en la base donde tiene menor tamaño.

A continuación, se define cómo se han de realizar las distintas operaciones (suma/resta, producto escalar y vectorial) entre dos vectores extendidos.

5.2.1. Operaciones entre vectores extendidos

Sean los vectores extendidos $\boldsymbol{v}$ y $\boldsymbol{u}$ donde $\boldsymbol{v}^{\mathcal{B}_i}$ y $\boldsymbol{u}^{\mathcal{B}_j}$ han de pertenecer al mismo espacio vectorial.:

$$\begin{aligned}[\![\boldsymbol{v}]\!] &= \left[\ \boldsymbol{v}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_N}\ \right] \\ [\![\boldsymbol{u}]\!] &= \left[\ \boldsymbol{u}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \boldsymbol{u}^{\mathcal{B}_j} \ \middle|\ ... \ \middle|\ \boldsymbol{u}^{\mathcal{B}_M}\ \right]\end{aligned} \tag{5.6}$$

5.2.1.1. Suma de vectores extendidos

La suma se define según:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{v}]\!] + [\![\boldsymbol{u}]\!] = \left[\ \boldsymbol{v}^{\mathcal{B}_1} + \boldsymbol{u}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_i} + \boldsymbol{u}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_j} + \boldsymbol{u}^{\mathcal{B}_j} \ \middle|\ ...\ \right] \tag{5.7}$$

Resulta interesante definir la suma de un vector $\boldsymbol{u}^{\mathcal{B}_i}$ (proyectado de forma natural en la base $\mathcal{B}_i$) con el vector extendido $[\![\boldsymbol{v}]\!]$ como:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{v}]\!] + \boldsymbol{u}^{\mathcal{B}_i} = \left[\ \boldsymbol{v}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_i} + \boldsymbol{u}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \boldsymbol{v}^{\mathcal{B}_N}\ \right] \tag{5.8}$$

5.2.1.2. Producto de un escalar y un vector extendido

Sea a un escalar, el producto $a \cdot [\![\boldsymbol{v}]\!]$ se define como:

$$a \cdot [\![\boldsymbol{v}]\!] = \left[\ a\,\boldsymbol{v}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ a\,\boldsymbol{v}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ a\,\boldsymbol{v}^{\mathcal{B}_N}\ \right] \tag{5.9}$$

5.2.1.3. Producto escalar de vectores extendidos

El producto escalar de dos vectores extendidos se define como:

$$w = [\![\boldsymbol{v}]\!] \cdot [\![\boldsymbol{u}]\!] = \sum_{i=1}^{N} \sum_{j=1}^{M} \boldsymbol{v}^{\mathcal{B}_i} \, \boldsymbol{u}^{\mathcal{B}_j} \tag{5.10}$$

Nótese que la base de los productos $\boldsymbol{v}^{\mathcal{B}_i} \cdot \boldsymbol{u}^{\mathcal{B}_j}$ queda determinada cuando se resuelve la *base común* en que se proyecta el vector resultado.

Como se ha detallado en la sección 5.1 el producto escalar produce las ESST. La ecuación anterior permite ver que el uso de vectores extendidos evita por completo su aparición. Para ello basta que la base en que se realiza la operación sea la base $\mathcal{B}_i$, la base $\mathcal{B}_j$ o una base intermedia. El modificador "gravedad" (Apartado 4.1.2.3) pueden ser utilizado a este fin.

El producto escalar de un vector por un vector extendido se define según:

$$w = [\![\boldsymbol{v}]\!] \cdot \boldsymbol{u}^{\mathcal{B}_i} = \boldsymbol{v}^{\mathcal{B}_1} \, \boldsymbol{u}^{\mathcal{B}_i} + ... + \boldsymbol{v}^{\mathcal{B}_i} \, \boldsymbol{u}^{\mathcal{B}_i} + ... + \boldsymbol{v}^{\mathcal{B}_N} \, \boldsymbol{u}^{\mathcal{B}_i} \tag{5.11}$$

5.2.1.4. Producto vectorial de vectores extendidos

El producto vectorial de dos vectores extendidos se define según:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{v}]\!] \wedge [\![\boldsymbol{u}]\!] = \left[\; \boldsymbol{v}^{\mathcal{B}_1} \wedge \boldsymbol{u}^{\mathcal{B}_1} \; \middle| \; ... \; \middle| \; \boldsymbol{v}^{\mathcal{B}_1} \wedge \boldsymbol{u}^{\mathcal{B}_i} \; \middle| \; ... \; \middle| \; \boldsymbol{v}^{\mathcal{B}_i} \wedge \boldsymbol{u}^{\mathcal{B}_j} \; \middle| \; ... \; \right] \tag{5.12}$$

Nótese que la base de los productos $\boldsymbol{v}^{\mathcal{B}_i} \wedge \boldsymbol{u}^{\mathcal{B}_j}$ queda determinada cuando se resuelve la *base común* en que se proyecta el vector resultado. A fin de evitar la aparición de expresiones simplificables trigonométricamente, los mismos criterios utilizados en el caso del producto escalar son aplicables, es decir, realizar

la operación en la base $\mathcal{B}_i$, en la base $\mathcal{B}_j$ o en una intermedia.

El producto vectorial de un vector por un vector extendido se define como:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{v}]\!] \wedge \boldsymbol{u}^{\mathcal{B}_i} = \left[\begin{array}{c|c|c|c} \boldsymbol{v}^{\mathcal{B}_1} \wedge \boldsymbol{u}^{\mathcal{B}_i} & \ldots & \boldsymbol{v}^{\mathcal{B}_i} \wedge \boldsymbol{u}^{\mathcal{B}_i} & \ldots \end{array} \right| \left. \boldsymbol{v}^{\mathcal{B}_N} \wedge \boldsymbol{u}^{\mathcal{B}_i} \right] \tag{5.13}$$

5.2.1.5. Producto vectorial triple extendidos

Un caso especial a tener en cuenta es el producto vectorial de tres vectores extendidos. Con la idea de aprovechar la estructura de vector extendido se propone utilizar la denominada "fórmula de Lagrange".

Es decir, sean tres vectores extendidos $[\![\boldsymbol{u}]\!]$, $[\![\boldsymbol{v}]\!]$, y $[\![\boldsymbol{w}]\!]$ el producto vectorial triple se expresa:

$$[\![\boldsymbol{u}]\!] \wedge ([\![\boldsymbol{v}]\!] \wedge [\![\boldsymbol{w}]\!]) = [\![\boldsymbol{v}]\!]([\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{w}]\!]) - [\![\boldsymbol{w}]\!]([\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{v}]\!]) \tag{5.14}$$

El realizar este producto de esta forma permite sacar provecho de la estructura extendida ya que los sumandos $[\![\boldsymbol{v}]\!]([\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{w}]\!])$ y $[\![\boldsymbol{w}]\!]([\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{v}]\!])$ son a su vez vectores de tipo extendido y al realizar la suma, esta se puede realizar de forma óptima. Esta suma, como ya se ha visto, se realiza sumando entre sí elementos que están proyectados en la misma base.

Además al realizar el producto de esta forma, la expresiones obtenidas aparecen óptimamente parentesizadas.

5.2.1.6. Derivada temporal de vectores

La derivada temporal del vector extendido $[\![\boldsymbol{v}]\!]$ respecto a la referencia R se ha de hacer según las reglas de derivación temporal de vectores para cada una de los elementos del vector extendido.

$$[\![\dot{\boldsymbol{v}}]\!] = \left.\frac{d[\![\boldsymbol{v}]\!]}{dt}\right|_R = \sum_i \left.\frac{d\boldsymbol{v}^{\boldsymbol{\mathcal{B}_i}}}{dt}\right|_{R_i} + \sum_i [\![\boldsymbol{\omega}_R^{R_i}]\!] \wedge \boldsymbol{v}^{\boldsymbol{\mathcal{B}_i}} \tag{5.15}$$

Donde $\left.\frac{d\boldsymbol{v}^{\mathcal{B}_i}}{dt}\right|_i$ es la derivada temporal respecto a la referencia R_i de un vector que se expresa de forma natural en la base $\mathcal{B}_i$ de dicha referencia.

5.2.2. Ejemplo de vector extendido

En el siguiente ejemplo se muestra brevemente cómo se trabaja con los vectores extendidos.

Se definen los siguiente vectores extendidos pertenecientes al mismo espacio vectorial donde los elementos de la columna i-ésima tienen sus componentes definidas de forma natural en la base $\mathcal{B}_i$:

$$\begin{aligned}[\![\boldsymbol{u}]\!] &= \left[\begin{array}{c|c|c} \mathbf{0} & \boldsymbol{u}^{\mathcal{B}_2} & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{v}]\!] &= \left[\begin{array}{c|c|c} \mathbf{0} & \boldsymbol{v}^{\mathcal{B}_2} & \boldsymbol{v}^{\mathcal{B}_3} \end{array}\right]\end{aligned} \tag{5.16}$$

La suma de los vectores es:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{u}]\!] + [\![\boldsymbol{v}]\!] = \left[\begin{array}{c|c|c} \mathbf{0} & \boldsymbol{u}^{\mathcal{B}_2} + \boldsymbol{v}^{\mathcal{B}_2} & \boldsymbol{v}^{\mathcal{B}_3} \end{array}\right] \tag{5.17}$$

Hay que destacar que el resultado se trata también de un vector extendido. De nuevo hay que recordar que no se ha realizado ningún cambio de base. Simplemente se han sumado los elementos están proyectados en la misma base.

El producto escalar da como resultado una expresión escalar:

$$w = [\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{v}]\!] = \boldsymbol{u}^{\mathcal{B}_2}\, \boldsymbol{v}^{\mathcal{B}_2} + \boldsymbol{u}^{\mathcal{B}_2}\, \boldsymbol{v}^{\mathcal{B}_3} \tag{5.18}$$

En este caso sí que se han de realizar cambios base, los elementos de los vectores extendidos que están en distintas columnas han de ser proyectados en una *base común*. Hay dos soluciones posibles: bien se podrían proyectar las componentes $\boldsymbol{v}^{\mathcal{B}_3}$ en la base $\mathcal{B}_2$, o bien las componentes de $\boldsymbol{v}^{\mathcal{B}_2}$ en la base $\mathcal{B}_3$, cualquiera de las dos soluciones sería correcta y no contendría ESST.

A modo de ejemplo se expresa la primera de las dos posibilidades:

$$w = [\![\boldsymbol{u}]\!] \cdot [\![\boldsymbol{v}]\!] = \{\boldsymbol{u}^{\mathcal{B}_2}\}_{\mathcal{B}_2} \{\boldsymbol{v}^{\mathcal{B}_2}\}_{\mathcal{B}_2} + \{\boldsymbol{u}^{\mathcal{B}_2}\}_{\mathcal{B}_2} \mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_3} \{\boldsymbol{v}^{\mathcal{B}_3}\}_{\mathcal{B}_3} \tag{5.19}$$

Por último, en el producto vectorial de estos dos vectores extendidos también existen dos posibilidades. El producto $\boldsymbol{u}^{\mathcal{B}_2} \wedge \boldsymbol{v}^{\mathcal{B}_3}$ puede ser proyectado en la base $\mathcal{B}_2$ o en la base $\mathcal{B}_3$.

A modo de ejemplo, se muestran ambas soluciones:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{u}]\!] \wedge [\![\boldsymbol{v}]\!] = \left[\begin{array}{c|c|c} \mathbf{0} & \boldsymbol{u}^{\mathcal{B}_2} \wedge \boldsymbol{v}^{\mathcal{B}_2} + \boldsymbol{u}^{\mathcal{B}_2} \wedge \boldsymbol{v}^{\mathcal{B}_3} & \mathbf{0} \end{array} \right] \tag{5.20}$$

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{u}]\!] \wedge [\![\boldsymbol{v}]\!] = \left[\begin{array}{c|c|c} \mathbf{0} & \boldsymbol{u}^{\mathcal{B}_2} \wedge \boldsymbol{v}^{\mathcal{B}_2} & \boldsymbol{u}^{\mathcal{B}_2} \wedge \boldsymbol{v}^{\mathcal{B}_3} \end{array} \right] \tag{5.21}$$

5.3. Simplificaciones de origen "tensorial"

Este apartado se centra en las expresiones que tienen un origen en operaciones entre tensores de orden dos.

Sean dos tensores de rango dos, donde cada uno de ellos se representa de forma natural en una base de proyección distinta, para realizar una operación binaria entre ellos ambos han de ser proyectados a una *base común*. En dicha base la operación se realiza como se realizaría con matrices regulares.

El origen de las ESST con origen en operaciones entre tensores es similar a la explicada para vectores.

En este capítulo se va a trabajar con los tensores propuestos en la sección 4.4. Por simplicidad del texto, los tensores únicamente van a tener asociado un espacio vectorial, el cartesiano, considerándose el otro espacio vectorial como constante, es decir, no permitiéndose cambios de parametrización. Así pues, los cambios de base correspondientes a tensores se han de realizar según se explica en la citada sección.

Se indicarán como $\{\mathbf{T}\}_{\mathcal{B}_1}$ las componentes del tensor $\mathbf{T}$ (no cartesiano) proyectadas en la base $\mathcal{B}_1$ obviandose el otro espacio vectorial asociado al cambio de parametrización. De igual modo, las componentes tensor de inercia $\mathbf{I}$ proyectadas en la base $\mathcal{B}_1$ se indicarán como $\{\mathbf{I}\}_{\mathcal{B}_1}$.

5.4. Nueva estructura simbólica de tensor: Tensor extendido

Análogamente a como se ha procedido con los vectores se propone una nueva representación los tensores de rango dos. La nueva representación se ha denominado "tensor extendido".

Sea el tensor $\mathbf{T}$, obtenido como resultado de la suma de varios tensores, cuyas componentes se conocen en distintas bases ($\mathcal{B}_1$, $\mathcal{B}_2$,..., $\mathcal{B}_N$):

$$\mathbf{T} = \mathbf{T}^{\mathcal{B}_1} + ... + \mathbf{T}^{\mathcal{B}_i} + ... + \mathbf{T}^{\mathcal{B}_N} \tag{5.22}$$

Se define el "tensor extendido" como una matriz en donde cada columna i contiene la parte del tensor $\mathbf{T}$ cuyas componentes están expresadas de forma "natural" en la base $\mathcal{B}_i$. N corresponde al número total de bases utilizadas en la formulación del problema. Es decir:

$$[\![\mathbf{T}]\!] = \left[\begin{array}{c|c|c|c|c} \mathbf{T}^{\mathcal{B}_1} & ... & \mathbf{T}^{\mathcal{B}_i} & ... & \mathbf{T}^{\mathcal{B}_N} \end{array} \right] \tag{5.23}$$

Cada una de las columnas de esta matriz se denominará "componente del tensor extendido" o simplemente "componente".

Esta nueva estructura es óptima en el sentido en que cada componente $\mathbf{T}^{\mathcal{B}_i}$ del tensor extendido está representada en una base donde tiene un menor tamaño.

Para poder trabajar con esta representación se define cómo se han de realizar las distintas operaciones de suma/resta y producto entre dos tensores extendidos.

5.4.1. Operaciones entre tensores extendidos

Sean los tensores extendidos $[\![\mathbf{T}]\!]$ y $[\![\mathbf{S}]\!]$ donde $\mathbf{T}^{\mathcal{B}_i}$ y $\mathbf{S}^{\mathcal{B}_j}$ son aplicaciones lineales entre los mismos espacios vectoriales:

$$\begin{aligned}[\![\mathbf{T}]\!] &= \left[\ \mathbf{T}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_N}\ \right] \\ [\![\mathbf{S}]\!] &= \left[\ \mathbf{S}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \mathbf{S}^{\mathcal{B}_j} \ \middle|\ ... \ \middle|\ \mathbf{S}^{\mathcal{B}_M}\ \right]\end{aligned} \tag{5.24}$$

5.4.1.1. Suma de dos tensores

La suma se define según:

$$[\![\mathbf{W}]\!] = [\![\mathbf{T}]\!] + [\![\mathbf{S}]\!] = \left[\ \mathbf{T}^{\mathcal{B}_1} + \mathbf{S}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_i} + \mathbf{S}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_j} + \mathbf{S}^{\mathcal{B}_j} \ \middle|\ ...\ \right] \tag{5.25}$$

Resulta interesante definir la suma de un tensor $\mathbf{S}^{\mathcal{B}_i}$ y un tensor extendido $[\![\mathbf{T}]\!]$ como:

$$[\![\mathbf{W}]\!] = [\![\mathbf{T}]\!] + \mathbf{S}^{\mathcal{B}_i} = \left[\ \mathbf{T}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_i} + \mathbf{S}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ \mathbf{T}^{\mathcal{B}_N}\ \right] \tag{5.26}$$

5.4.1.2. Producto de un escalar y un tensor extendido

Sea a un escalar, el producto $a \cdot [\![\mathbf{T}]\!]$ se define como:

$$a \cdot [\![\mathbf{T}]\!] = \left[\ a\,\mathbf{T}^{\mathcal{B}_1} \ \middle|\ ... \ \middle|\ a\,\mathbf{T}^{\mathcal{B}_i} \ \middle|\ ... \ \middle|\ a\,\mathbf{T}^{\mathcal{B}_N}\ \right] \tag{5.27}$$

5.4.1.3. Producto de dos tensores

El producto entre dos tensores extendidos se define como:

$$\mathbf{W} = [\![\mathbf{T}]\!] \cdot [\![\mathbf{S}]\!] = \sum_{i=1}^{N} \sum_{j=1}^{M} \mathbf{T}^{\mathcal{B}_i} \, \mathbf{S}^{\mathcal{B}_j} \tag{5.28}$$

Nótese que los productos entre elementos que están representados en la misma base no requieren ningún cambio de base. Los productos entre elementos que están representados en distintas bases se han de proyectar a una *base común*[3] en la que realizar la operación.

Por último, el producto entre un tensor y un tensor extendido se define como

$$[\![\mathbf{W}]\!] = [\![\mathbf{T}]\!] \cdot \mathbf{S}^{\mathcal{B}_j} = \sum_{i=1}^{N} [\![\mathbf{T}^{\mathcal{B}_i} \, \mathbf{S}^{\mathcal{B}_j}]\!] \tag{5.29}$$

5.4.1.4. Producto entre un tensor y un vector

Sea $[\![\boldsymbol{u}]\!]$ un vector extendido y $[\![\mathbf{T}]\!]$ un tensor extendido los productos entre ambos elementos se pueden escribir de una manera reducida como sigue:

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{u}]\!] \cdot [\![\mathbf{T}]\!] = \sum_{i=1}^{N} \sum_{j=1}^{M} [\![\boldsymbol{u}^{\mathcal{B}_i} \, \mathbf{T}^{\mathcal{B}_j}]\!] \tag{5.30}$$

$$[\![\boldsymbol{w}]\!] = [\![\mathbf{T}]\!] \cdot [\![\boldsymbol{u}]\!] = \sum_{i=1}^{N} \sum_{j=1}^{M} [\![\mathbf{T}^{\mathcal{B}_j} \, \boldsymbol{u}^{\mathcal{B}_i}]\!] \tag{5.31}$$

Para evitar la aparición de ESST los productos del tipo $\boldsymbol{u}^{\mathcal{B}_i} \, \mathbf{T}^{\mathcal{B}_j}$ cuando $i \neq j$ requieren un cambio de base. Este se ha de realizar a la base $\mathcal{B}_i$, $\mathcal{B}_j$ o una que esté entre ambas en la correspondiente cadena de bases.

Los productos entre un vector y un tensor extendido o entre un tensor y un vector extendido respectivamente se definen de la siguiente manera:

[3] La *base común* puede ser la base de uno de los elementos, del otro o una base intermedia en la cadena de bases.

$$[\![\boldsymbol{w}]\!] = \boldsymbol{u}^{\mathcal{B}_i} \cdot [\![\mathbf{T}]\!] = [\![\boldsymbol{u}^{\mathcal{B}_i}]\!] \cdot [\![\mathbf{T}]\!] \tag{5.32}$$

y análogamente

$$[\![\boldsymbol{w}]\!] = [\![\boldsymbol{u}]\!] \cdot \mathbf{T}^{\mathcal{B}_i} = [\![\boldsymbol{u}]\!] \cdot [\![\mathbf{T}^{\mathcal{B}_i}]\!] \tag{5.33}$$

5.4.2. Ejemplo de trabajo con tensores extendidos

Es este ejemplo muestra cómo se trabaja con los tensores extendidos.

Se definen los siguiente tensores extendidos, ambos aplicaciones lineales entre los mismos espacios vectoriales, donde los elementos de la columna i-ésima tienen sus componentes definidas de forma natural en la base $\mathcal{B}_i$.

$$\begin{aligned} [\![\mathbf{T}]\!] &= \left[\begin{array}{c|c|c} \mathbf{T}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\mathbf{S}]\!] &= \left[\begin{array}{c|c|c} \mathbf{S}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{S}^{\mathcal{B}_3} \end{array}\right] \end{aligned} \tag{5.34}$$

La suma de los tensores extendidos es:

$$[\![\mathbf{W}]\!] = [\![\mathbf{T}]\!] + [\![\mathbf{S}]\!] = \left[\begin{array}{c|c|c} \mathbf{T}^{\mathcal{B}_1} + \mathbf{S}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{S}^{\mathcal{B}_3} \end{array}\right] \tag{5.35}$$

Nótese que no es necesario ningún cambio de base.

Análogamente a como ocurre con el producto de dos vectores extendidos el producto de los dos tensores extendidos tiene dos soluciones posibles dependiendo de cómo se hagan los cambios de base:

$$\begin{aligned} [\![\mathbf{W}]\!] = [\![\mathbf{T}]\!] \cdot [\![\mathbf{S}]\!] &= \left[\begin{array}{c|c|c} \mathbf{T}^{\mathcal{B}_1}\,\mathbf{S}^{\mathcal{B}_1} + \mathbf{T}^{\mathcal{B}_1}\,\mathbf{S}^{\mathcal{B}_3} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ &= \left[\begin{array}{c|c|c} \mathbf{T}^{\mathcal{B}_1}\,\mathbf{S}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{T}^{\mathcal{B}_1}\,\mathbf{S}^{\mathcal{B}_3} \end{array}\right] \end{aligned} \tag{5.36}$$

En este caso sí que se han de realizar cambios base, los elementos de los tensores extendidos que están en distintas columnas han de ser proyectados

en una base común donde realizar la operación. Hay dos posibles soluciones: bien se podrían proyectar las componentes $\mathbf{S}^{\mathcal{B}_3}$ en la base $\mathcal{B}_1$, o bien las componentes de $\mathbf{T}^{\mathcal{B}_1}$ en la base $\mathcal{B}_3$, cualquiera de las dos soluciones sería correcta y no contendría ESST.

5.5. Matriz jacobiana de un vector extendido

Es frecuente en el campo de la DSM el cálculo de la matriz jacobiana de un vector respecto a otro vector no cartesiano (por ejemplo, el vector $\dot{\boldsymbol{q}}$ de velocidades generalizadas).

Sea un vector extendido genérico $[\![v]\!]$:

$$[\![\boldsymbol{v}]\!] = \left[\begin{array}{c|c|c|c} \boldsymbol{v}^{\mathcal{B}_1} & \ldots & \boldsymbol{v}^{\mathcal{B}_i} & \ldots \end{array} \right| \boldsymbol{v}^{\mathcal{B}_N} \Big] \tag{5.37}$$

Su matriz jacobiana respecto a otro vector $\dot{\boldsymbol{q}}$ (no cartesiano) es:

$$\frac{\partial}{\partial \dot{\boldsymbol{q}}} [\![\boldsymbol{v}]\!] = \left[\begin{array}{c|c|c|c} \frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{v}^{\mathcal{B}_1} & \ldots & \frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{v}^{\mathcal{B}_i} & \ldots \end{array} \right| \frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{v}^{\mathcal{B}_N} \Big] \tag{5.38}$$

Si el vector $\dot{\boldsymbol{q}}$ es un elemento de un espacio vectorial, como se ha visto en la sección 4.4 la ecuación anterior puede interpretarse como un tensor extendido por ser una aplicación lineal entre el espacio cartesiano y el espacio de las velocidades generalizadas.

5.6. Orden de los elementos dentro de las nuevas estructuras

En los nuevos elementos vector y tensor extendidos aquí definidos no se ha hablado hasta ahora de cómo se han de ordenar los vectores/tensores dentro de la estructura que los almacena. Simplemente se ha detallado que han de agruparse según la base en la que sus componentes se proyectan de forma natural. Los elementos que tienen las componentes proyectadas en la misma

base se han de agrupar juntos, en forma de suma.

Si bien en principio el orden es independiente del funcionamiento de esta técnica, de cara a la implementación, es importante discutir en qué orden se pueden disponer los distintos elementos.

En el paradigma DSM propuesto en esta tesis (Apartado: 4.1.1) las bases se van definiendo de manera secuencial donde cada base se define respecto a una base previamente definida. Así pues, los elementos de vector o tensor extendidos se ordenan según el orden en que se han definido las bases.

Al crear un vector o tensor extendido, se creará con un número de componentes igual al número de bases existentes en el sistema en ese instante. Si bien realizarlo de esta manera, al implementarlo, acarrea un consumo de memoria mayor presenta la ventaja de que las operaciones de suma y producto se puede realizar de manera más eficaz al no tener que buscar la base "correspondiente" a cada uno de los elementos del vector/tensor extendido. La suma, por ejemplo, se puede hacer directamente elemento a elemento, donde algunos serán nulos, otros no y algunos incluso inexistentes.

Por ejemplo, sean los siguientes vectores extendidos:

$$\begin{aligned} [\![\boldsymbol{v}]\!] &= \left[\ \boldsymbol{v}^{\mathcal{B}_1} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_2} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_3} \ \middle|\ \mathbf{0} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_5} \ \right] \\ [\![\boldsymbol{u}]\!] &= \left[\ \boldsymbol{u}^{\mathcal{B}_1} \ \middle|\ \mathbf{0} \ \middle|\ \boldsymbol{u}^{\mathcal{B}_3} \ \middle|\ \mathbf{0} \ \right] \end{aligned} \tag{5.39}$$

La suma se realiza sumando la i-ésima componente del vector $[\![\boldsymbol{v}]\!]$ con la i-ésima del vector $[\![\boldsymbol{u}]\!]$ directamente.

$$[\![\boldsymbol{v}]\!] + [\![\boldsymbol{u}]\!] = \left[\ \boldsymbol{v}^{\mathcal{B}_1} + \boldsymbol{u}^{\mathcal{B}_1} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_2} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_3} + \boldsymbol{u}^{\mathcal{B}_3} \ \middle|\ \mathbf{0} \ \middle|\ \boldsymbol{v}^{\mathcal{B}_5} \ \right] \tag{5.40}$$

El vector resultante tienen un tamaño igual al del vector con mayor número de elementos.

5.7. Optimización del vector/tensor extendido

En esta sección se presenta una optimización que permite, en primer lugar, generar expresiones más compactas y, en segundo lugar, evitar cambios de base innecesarios (generalmente en el producto triple de tres vectores/tensores extendidos).

Para ello en cada operación entre vectores y/o tensores extendidos, cuando se tiene un elemento (vector/tensor) cuyas componentes se proyectan de forma natural en la base $\mathcal{B}_j$, este elemento ha de ser cambiado de base siguiendo la cadena de bases hacia la raíz de la estructura de bases. El cambio de ha realizar mientras las componentes de ese elemento proyectadas en cada base $\mathcal{B}_i$ ($i : j...0$) no cambien con respecto a las proyectadas en la base $\mathcal{B}_j$.

Los dos siguientes ejemplos muestran cuándo se producen esto hechos.

5.7.0.1. Ejemplo: Expresiones más compactas

Sea, por ejemplo, el siguiente vector extendido de velocidad angular del segundo sólido del mecanismo de cuatro barra que se detalla en A.2:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} & \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} & \mathbf{0} \end{array}\right] \tag{5.41}$$

donde

$$\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} = \left\{\begin{array}{c} 0 \\ \theta_1 \\ 0 \end{array}\right\}_{\mathcal{B}_1} \qquad \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}} = \left\{\begin{array}{c} 0 \\ \theta_2 \\ 0 \end{array}\right\}_{\mathcal{B}_2} \tag{5.42}$$

Si por ejemplo, se realiza el producto escalar de este vector extendido por sí mismo se tiene que:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] = \theta_1\theta_1 + \theta_1\theta_2 + \theta_2\theta_1 + \theta_2\theta_2 = \theta_1^2 + 2\theta_1\theta_2 + \theta_2^2 \tag{5.43}$$

Al tratarse de un mecanismo plano, las componentes velocidad angular no

cambian cuando se cambia de base de proyección. Así pues, tal y como se comenta en esta sección, las componentes del vector extendido han se ser cambiadas a la base que se encuentre más abajo en la cadena de bases para la cual éstas no cambian, es decir:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} & \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} & \mathbf{0} \end{array}\right] = \left[\begin{array}{c|c|c|c} \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.44}$$

Así pues, realizando ahora el producto escalar se tiene que:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] = (\theta_1 + \theta_2)(\theta_1 + \theta_2) = (\theta_1 + \theta_2)^2 \tag{5.45}$$

Como se ve, operando de esta forma, las expresiones aparecen mucho más agrupadas. Lo que al ser exportado genera menos átomos y menos operaciones algebraicas[4].

5.7.0.2. Ejemplo. Cambios de base innecesarios

En este caso se tienen tres tensores extendidos en un sistema con tres bases definidas ($\mathcal{B}_1$, $\mathcal{B}_2$ y $\mathcal{B}_3$). Cada uno de los tensores tiene una componente conocida de forma natural en una de las bases. Los tensores tienen todos el mismo tamaño nxn y son simétricos.

$$\begin{aligned} [\![\mathbf{T}]\!] &= \left[\begin{array}{c|c|c} \mathbf{T}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\mathbf{S}]\!] &= \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{S}^{\mathcal{B}_2} & \mathbf{0} \end{array}\right] \\ [\![\mathbf{W}]\!] &= \left[\begin{array}{c|c|c} \mathbf{0} & \mathbf{0} & \mathbf{W}^{\mathcal{B}_3} \end{array}\right] \end{aligned} \tag{5.46}$$

Además, para este ejemplo , el tensor $\mathbf{W}^{\mathcal{B}_3}$, proyectado de forma natural en la base $\mathcal{B}_3$ puede ser cambiado de base a la base $\mathcal{B}_2$ sin que sus componentes cambien. Es decir:

[4]En la operación $\theta_1^2 + 2\theta_1\theta_2 + \theta_2^2$ se han de realizar seis operaciones (dos sumas y cuatro productos). Mientras que en $(\theta_1 + \theta_2)^2$ se han de realizar únicamente dos operaciones (una suma y un producto).

$$\mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_3}\ \left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_3}\ \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_2} = \left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_2} \tag{5.47}$$

Se desea realizar el producto $[\![\mathbf{T}]\!] \cdot [\![\mathbf{W}]\!] \cdot [\![\mathbf{S}]\!]$.

Si se realiza el producto de los tres tensores sin realizar el cambio de base tal y como se propone en esta sección, se tiene que:

$$[\![\mathbf{T}]\!] \cdot [\![\mathbf{W}]\!] \cdot [\![\mathbf{S}]\!] = \left\{\mathbf{T}^{\mathcal{B}_1}\right\}_{\mathcal{B}_1}\ \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}\ \mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_3}\ \left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_3}\ \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_2}\ \left\{\mathbf{S}^{\mathbf{2}}\right\}_{\mathcal{B}_2} \tag{5.48}$$

En un código simbólico genérico que realizara esta operación, no se podría garantizar el orden en el que la operación anterior es realizada, por lo que el sub-producto $\mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_3}\ \left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_3}\ \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_2}$ no podría ser simplificado a $\left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_2}$ y, generalmente, se generarían ESST.

En cambio, si se realiza el cambio de base como se detalla en esta sección, es decir:

$$[\![\mathbf{W}]\!] = \left[\ \mathbf{0}\ \middle|\ \mathbf{0}\ \middle|\ \mathbf{W}^{\mathcal{B}_3}\ \right] = \left[\ \mathbf{0}\ \middle|\ \mathbf{W}^{\mathcal{B}_3}\ \middle|\ \mathbf{0}\ \right] \tag{5.49}$$

El producto de los tres tensores queda:

$$[\![\mathbf{T}]\!] \cdot [\![\mathbf{W}]\!] \cdot [\![\mathbf{S}]\!] = \left\{\mathbf{T}^{\mathcal{B}_1}\right\}_{\mathcal{B}_1}\ \mathbf{R}_{\mathcal{B}_1}^{\mathcal{B}_2}\ \left\{\mathbf{W}^{\mathcal{B}_3}\right\}_{\mathcal{B}_2}\ \left\{\mathbf{S}^{\mathcal{B}_2}\right\}_{\mathcal{B}_2} \tag{5.50}$$

Lo que evita, sin más cautela, que aparezcan ESST.

5.8. Simplificaciones trigonométricas en el campo de la DSM

Una vez realizado el estudio sobre las simplificaciones trigonométricas, de dónde provienen (Sección 5.1) y propuestas una nuevas estructuras (vector y tensor extendidos) que evitan la aparición de ESST, se analiza cuándo aparecen

estas expresiones en el contexto de la DSM.

Si bien esta sección se centra en aplicar los vectores y tensores extendidos para el cálculo de la matriz de masa y el vector de fuerzas generalizadas según la formulación aquí presentada y basada en el PPV (Secciones 4.5 y 4.6), hay que remarcar que estas mismas estructuras son válidas para cualquier otra formulación y metodología que se desee utilizar para el planteamiento de las ecuaciones dinámicas.

Las técnicas presentadas en este capítulo hacen que no aparezcan ESST y, además, el algoritmo de la Figura 4.10 para el cálculo de $\mathbf{M}$ y δ hará que el tamaño de las expresiones sea el menor posible. Aun así, hay una serie de factores a tener en cuenta y que se describen a continuación.

5.8.1. Matrices jacobianas de velocidad lineal

Tanto en el cálculo de $\mathbf{M}$ como de δ (Ecuaciones 4.54 y 4.62 respectivamente), para cada sólido, es necesario obtener la matriz jacobiana respecto de las velocidades generalizadas de la velocidad del punto genérico B del sólido, es decir, $\mathbf{v}_{RI}^{B}$. Por simplicidad, en esta sección, este vector se designará como $\mathbf{v}$

Este vector velocidad $\mathbf{v}$ es obtenido mediante composición de velocidades por medio del operador detallado en 4.3.4. Este mismo operador es válido para el trabajo con vectores extendidos simplemente sustituyendo en el correspondiente algoritmo los vectores por vectores extendidos.

El cálculo de la matriz jacobiana $\mathbf{v}_{\dot{q}}$ se define según:

$$\mathbf{v}_{\dot{q}} = \frac{\partial}{\partial \dot{\boldsymbol{q}}} \mathbf{v} \tag{5.51}$$

En 4.4 se ha justificado que estas matrices jacobianas son tensores por ser aplicaciones lineales entre dos espacios vectoriales y por lo tanto pueden ser tratadas como tensores extendidos. Así pues:

$$[\![\mathbf{v}_{\dot{q}}]\!] = \frac{\partial}{\partial \dot{\boldsymbol{q}}} [\![\mathbf{v}]\!] = \frac{\partial}{\partial \dot{\boldsymbol{q}}} \left[\; \dots \; \middle| \; \mathbf{v}^{\mathcal{B}_i} \; \middle| \; \dots \; \right] = \left[\; \dots \; \middle| \; \frac{\partial}{\partial \dot{\boldsymbol{q}}} \mathbf{v}^{\mathcal{B}_i} \; \middle| \; \dots \; \right] \tag{5.52}$$

donde $\mathbf{v}^{\mathcal{B}_i}$ representa la parte del vector $\mathbf{v}$ cuyas componentes se proyectan de forma natural en la base $\mathcal{B}_i$.

Por ejemplo, en la ecuación 4.54 el producto $[\![\mathbf{v}_{\dot{q}}]\!] \cdot [\![\mathbf{v}_{\dot{q}}]\!]$, realizado mediante tensores extendidos, evitará sin más cautela la aparición de ESST.

Además de la matriz jacobiana $\mathbf{v}_{\dot{q}}$, en el cálculo del vector de fuerzas generalizadas (4.6) es necesario el cálculo de las matrices jacobianas respecto a las coordenadas generalizadas y respecto al tiempo, es decir, $\mathbf{v}_q$ y $\mathbf{v}_t$.

Es conveniente remarcar que la matriz jacobiana $\mathbf{v}_q$ no tiene naturaleza tensorial, por no ser el espacio de las coordenadas generalizadas un espacio vectorial. Sí que tiene naturaleza de tensor el producto $\mathbf{v}_q\ \dot{\boldsymbol{q}}$.

Así pues, para el cálculo del vector de fuerzas generalizadas se han de usar los siguientes tensores extendidos: $[\![\mathbf{v}_q\ \dot{\boldsymbol{q}}]\!]$ y $[\![\mathbf{v}_t]\!]$.

5.8.2. Matrices jacobianas de velocidad angular

De forma análoga, en las mismas ecuaciones para el cálculo de $\mathbf{M}$ y de δ, para cada sólido, se han de realizar productos con el tensor $\boldsymbol{\omega}_{\dot{q}}$. Donde $\boldsymbol{\omega}_{\dot{q}}$ se define según:

$$\boldsymbol{\omega}_{\dot{q}} = \frac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{\omega}_{RI}^{S} \tag{5.53}$$

El vector $\boldsymbol{\omega}_{RI}^{S}$ (por simplicidad se denotará como $\boldsymbol{\omega}$) es obtenido mediante el operador detallado en 4.3.3. Este mismo operador es válido para trabajar con vectores extendidos simplemente sustituyendo en el correspondiente algoritmo los vectores por vectores extendidos.

Análogamente a como ocurre con la velocidad absoluta, una vez obtenido dicho vector extendido de velocidad angular, la matriz jacobiana $\boldsymbol{\omega}_{\dot{q}}$ tienen naturaleza tensorial y por lo tanto:

$$[\![\boldsymbol{\omega}_{\dot{q}}]\!] = \frac{\partial}{\partial \dot{\boldsymbol{q}}} [\![\boldsymbol{\omega}]\!] = \frac{\partial}{\partial \dot{\boldsymbol{q}}} \left[\ \dots \mid \boldsymbol{\omega}^{\mathcal{B}_i} \mid \dots\ \right] = \left[\ \dots \mid \tfrac{\partial}{\partial \dot{\boldsymbol{q}}} \boldsymbol{\omega}^{\mathcal{B}_i} \mid \dots\ \right] \tag{5.54}$$

donde $\boldsymbol{\omega}^{\mathcal{B}_i}$ representa la parte del vector $\boldsymbol{\omega}$ cuyas componentes se proyectan de forma natural en la base i-ésima.

El cálculo de matriz de masa realizado mediante tensores extendidos, evitará sin más cautela la aparición de ESST. Aun así, como más adelante se detalla, el producto $[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}_B^S]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$ se ha de realizar de un modo especial, debido a las características del tensor de inercia.

Además de la matriz jacobiana $\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}$, en el cálculo del vector de fuerzas generalizadas (4.6) es necesario el cálculo de las matrices jacobianas respecto a las coordenadas generalizadas y respecto al tiempo, es decir, $\boldsymbol{\omega}_{\boldsymbol{q}}$ y $\boldsymbol{\omega}_t$.

Al igual que en el caso anterior, la matriz jacobiana $\boldsymbol{\omega}_{\boldsymbol{q}}$ no tiene naturaleza tensorial, por no ser el espacio de las coordenadas generalizadas un espacio vectorial. Sí que tiene naturaleza de tensor el producto $\boldsymbol{\omega}_{\boldsymbol{q}}\ \dot{\boldsymbol{q}}$.

Así pues, para el cálculo del vector de fuerzas generalizadas se han de usar los siguientes tensores extendidos: $[\![\boldsymbol{\omega}_{\boldsymbol{q}}\ \dot{\boldsymbol{q}}]\!]$ y $[\![\boldsymbol{\omega}_t]\!]$.

5.8.3. Matrices anti-simétricas

En las mismas ecuaciones para el cálculo de $\mathbf{M}$ y de $\boldsymbol{\delta}$, aparecen dos matrices anti-simétricas, $\widetilde{\mathbf{r}_B^G}$ y $\widetilde{\boldsymbol{\omega}_{RI}^S}$, es decir, las asociadas al vector $\mathbf{r}_B^G$ de centro de masa del sólido y al vector $\boldsymbol{\omega}_{RI}^S$ de velocidad angular del sólido. Por simplicidad, se denotarán como: $\widetilde{\mathbf{r}}$ y $\widetilde{\boldsymbol{\omega}}$. Estas matrices tienen naturaleza tensorial y por lo tanto es tratada con el tensor extendido:

Para el vector de centro de masas se tienen que:

$$[\![\widetilde{\mathbf{r}}]\!] = \left[\ \mathbf{0}\ \middle|\ \ldots\ \middle|\ \widetilde{\mathbf{r}^{\mathcal{B}_i}}\ \middle|\ \ldots\ \middle|\ \mathbf{0}\ \right] \tag{5.55}$$

donde $\mathbf{r}^{\mathcal{B}_i}$ es el vector de posición del centro de masas del sólido respecto a un punto cualquiera de sólido. Sus componentes se proyectan de forma natural en la base $\mathcal{B}_i$ del sólido.

Y para el vector de velocidad angular:

$$[\![\widetilde{\boldsymbol{\omega}}]\!] = \left[\ \widetilde{\boldsymbol{\omega}}^{\mathcal{B}_1}\ \middle|\ \ldots\ \middle|\ \widetilde{\boldsymbol{\omega}}^{\mathcal{B}_i}\ \middle|\ \mathbf{0}\ \middle|\ \ldots\ \middle|\ \mathbf{0}\ \right] \tag{5.56}$$

donde $\widetilde{\boldsymbol{\omega}}^{\mathcal{B}_i}$ es la matriz anti-simétrica de la parte del vector de velocidad angular que se proyecta se forma natural en la base $\mathcal{B}_i$.

5.8.4. Productos que implican el tensor de inercia

Ya se ha adelantado en esta sección que en el producto $[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}_B^S]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$ dependiendo de cómo sea el tensor de inercia y de cómo sean las velocidades angulares, aparecerán ESST. Para evitar su aparición el producto que implica al tensor de inercia requiere un trato especial.

Sea el siguiente tensor extendido:

$$[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] = \left[\begin{array}{c|c|c|c|c} \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_1} & \ldots & \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i} & \ldots & \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_k} \end{array} \right] \tag{5.57}$$

donde k es la base asociada al sólido.

El tensor de inercia (por simplicidad se denotará $\mathbf{I}$) también se representa mediante un tensor extendido con un único elemento (las componentes del tensor de inercia en la base del sólido $\mathcal{B}_k$):

$$[\![\mathbf{I}]\!] = \left[\begin{array}{c|c|c|c|c} \mathbf{0} & \ldots & \mathbf{I}^{\mathcal{B}_k} & \ldots & \mathbf{0} \end{array} \right] \tag{5.58}$$

La base natural $\mathcal{B}_k$ del tensor de inercia siempre está más arriba en la cadena de bases que las bases de las distintas componentes de $[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$. En caso de que se tenga un tensor de inercia general el orden en que se haga el producto no es importante. Operando con tensores extendidos se evita la aparición de ESST.

$$\begin{aligned} [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] &= \sum_{i=1}^{k} \sum_{j=1}^{k} \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i\,\mathsf{T}}\, \mathbf{I}^{\mathcal{B}_k}\, \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j} \\ &= \sum_{i=1}^{k} \sum_{j=1}^{k} \{\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i}\}_{\mathcal{B}_i}^{\mathsf{T}}\, \mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_k}\, \{\mathbf{I}^{\mathcal{B}_k}\}_{\mathcal{B}_k}\, \mathbf{R}_{\mathcal{B}_k}^{\mathcal{B}_j}\, \{\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j}\}_{\mathcal{B}_j} \end{aligned} \tag{5.59}$$

En cambio, en los casos en que el tensor de inercia presente algún tipo de

simetría, pese a operar con tensores extendidos, sí que aparecen ESST.

La solución para evitar la aparición de estas expresiones es la comentada en la sección 5.7. Es decir, el tensor $[\![\mathbf{I}]\!]$ se ha de cambiar a la base más baja para la cual sus componentes no cambian. Una vez realizada esta operación, entonces sí que se realiza el producto deseado. Es decir, el producto anterior toma la forma:

$$\begin{aligned}[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] &= \sum_{i=1}^{k-l}\sum_{j=1}^{k-l} \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i\,\mathsf{T}}\, \mathbf{I}^{\mathcal{B}_{k-l}}\, \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j} \\ &= \sum_{i=1}^{k-l}\sum_{j=1}^{k-l} \{\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i}\}_{\mathcal{B}_i}^{\mathsf{T}}\, \mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_{k-l}}\, \{\mathbf{I}^{\mathcal{B}_{k-l}}\}_{\mathcal{B}_{k-l}}\, \mathbf{R}_{\mathcal{B}_{k-l}}^{\mathcal{B}_j}\, \{\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j}\}_{\mathcal{B}_j}\end{aligned} \tag{5.60}$$

donde $\mathcal{B}_{k-l}$ es una base en la que las componentes del tensor de inercia son las mismas que en la base $\mathcal{B}_k$.

5.8.4.1. Rotor esférico

Un caso particular es cuando tensor de inercia es un rotor esférico[5].

Sea $[\![\mathbf{I}]\!] = I\ \mathbf{1}$ (siendo I un escalar) entonces el producto $[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot \mathbf{I} \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$ se puede escribir como:

$$[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] = [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot I \cdot \mathbf{1} \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] = I \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] \tag{5.61}$$

5.8.5. Productos que implican el vector centro de masa

Análogamente a como ocurre con el tensor de inercia, el producto triple que implica al tensor anti-simétrico asociado al vector de centro de masa, $[\![\mathbf{v}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![m\ \widetilde{\mathbf{r}}_B^G]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$ (Ecuación 4.54), también debe ser estudiado para evitar

[5]Un sólido es un rotor esférico en uno de sus puntos B si en este punto los tres momentos de inercia son iguales

la aparición de ESST.

El tensor extendido asociado a dicho vector, que por simplicidad se escribirá como $\mathbf{r}$, tiene un único elemento en la columna asociada a la base del sólido. Es decir:

$$[\![m\,\widetilde{\mathbf{r}}]\!] = \left[\ \mathbf{0}\ \middle|\ \ldots\ \middle|\ (m\,\widetilde{\mathbf{r}})^{\mathcal{B}_k}\ \middle|\ \mathbf{0}\ \middle|\ \ldots\ \middle|\ \mathbf{0}\ \right] \tag{5.62}$$

donde $\mathcal{B}_k$ es la base de sólido.

La base natural $\mathcal{B}_k$ del tensor anti-simétrico siempre está más arriba en la cadena de bases que las bases de las distintas componentes de $[\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!]$ y de $[\![\mathbf{v}_{\dot{\boldsymbol{q}}}]\!]$. Por lo tanto, el orden en que se haga el producto no es importante y gracias a forma de operar con tensores extendidos las ESST no aparecen.

El producto triple se desarrolla como sigue:

$$\begin{aligned}[\![\mathbf{v}_{\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![m\,\widetilde{\mathbf{r}}]\!] \cdot [\![\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}]\!] &= \sum_{i=1}^{k}\sum_{j=1}^{k} \mathbf{v}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i\,\mathsf{T}}\ (m\,\widetilde{\mathbf{r}})^{\mathcal{B}_k}\ \boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j} \\ &= \sum_{i=1}^{k}\sum_{j=1}^{k} \{\mathbf{v}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_i}\}_{\mathcal{B}_i}^{\mathsf{T}}\ \mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_k}\ \{(m\,\widetilde{\mathbf{r}})^{\mathcal{B}_k}\}_{\mathcal{B}_k}\ \mathbf{R}_{\mathcal{B}_k}^{\mathcal{B}_j}\ \{\boldsymbol{\omega}_{\dot{\boldsymbol{q}}}^{\mathcal{B}_j}\}_{\mathcal{B}_j}\end{aligned} \tag{5.63}$$

En cambio, en los casos en que el tensor anti-simétrico $\widetilde{\mathbf{r}}$ tenga algún tipo de simetría sí que aparecen ESST. La solución para evitar la aparición de esas expresiones es la comentada en la sección 5.7.

Es decir, al definir el tensor $\widetilde{\mathbf{r}}$, éste se ha de cambiar a la base más baja para la cual sus componentes no cambian. Una vez realizada esta operación, entonces sí que se realiza el producto deseado. Así pues, el producto anterior toma la forma:

$$\begin{aligned}[\![\mathbf{v}_{\dot{q}}]\!]^{\mathsf{T}} \cdot [\![m\,\widetilde{\mathbf{r}}]\!] \cdot [\![\boldsymbol{\omega}_{\dot{q}}]\!] &= \sum_{i=1}^{k-l}\sum_{j=1}^{k-l} \mathbf{v}_{\dot{q}}^{\mathcal{B}_i\,\mathsf{T}}\ (m\,\widetilde{\mathbf{r}})^{\mathcal{B}_{k-l}}\ \boldsymbol{\omega}_{\dot{q}}^{\mathcal{B}_j} \\ &= \sum_{i=1}^{k-l}\sum_{j=1}^{k-l} \{\mathbf{v}_{\dot{q}}^{\mathcal{B}_i}\}_{\mathcal{B}_i}^{\mathsf{T}}\ \mathbf{R}_{\mathcal{B}_i}^{\mathcal{B}_{k-l}}\ \{(m\,\widetilde{\mathbf{r}})^{\mathcal{B}_{k-l}}\}_{\mathcal{B}_{k-l}}\ \mathbf{R}_{\mathcal{B}_{k-l}}^{\mathcal{B}_j}\ \{\boldsymbol{\omega}_{\dot{q}}^{\mathcal{B}_j}\}_{\mathcal{B}_j}\end{aligned} \tag{5.64}$$

donde $\mathcal{B}_{k-l}$ es una base en la que las componentes del tensor anti-simétrico son las mismas que en la base $\mathcal{B}_k$.

5.8.6. Producto vectorial entre la velocidad angular y vector de posición

Cuando, en esta tesis, se propone el cálculo de la velocidad absoluta de un punto respecto a una referencia por el método de composición de velocidades (Sec: 4.3.4) aparecen productos del tipo:

$$\boldsymbol{\omega}_{\mathcal{B}_{k-l}}^{\mathcal{B}_k} \wedge \mathbf{r}_{O_{R_k}}^{P} \tag{5.65}$$

donde $\boldsymbol{\omega}_{\mathcal{B}_{k-l}}^{\mathcal{B}_k}$ es el vector velocidad angular de la base $\mathcal{B}_k$ de la referencia R_k respecto a la base $\mathcal{B}_{k-l}$ de la referencia R_{k-l} y $\mathbf{r}_{O_{R_k}}^{P}$ es el vector de posición del punto P respecto al punto O_{R_k} de la referencia R_k.

El vector $\mathbf{r}_{O_{R_k}}^{P}$ se proyecta de forma natural en la base $\mathcal{B}_k$ asociada al sólido mientras que el vector $\boldsymbol{\omega}_{\mathcal{B}_{k-l}}^{\mathcal{B}_k}$ se expresa de forma natural en una base $\mathcal{B}_{k-l}$ que está por debajo de $\mathcal{B}_k$ en la misma rama del árbol de bases.

Cuando se trabaja con giros elementales, si la velocidad angular se expresa de forma natural con la misma expresión en las bases $\mathcal{B}_{k-l}$ y $\mathcal{B}_k$, resulta conveniente proyectarla en la base $\mathcal{B}_k$ y realizar el producto en esa base.

Al trabajar con vectores extendidos y aplicando las reglas presentadas en 5.7 se reduce, sin más cautela, la complejidad de las expresiones generadas.

5.8.7. Simplificaciones trigonométricas en otras formulaciones

Si bien en esta sección se explica la forma de obtener $\mathbf{M}$ y δ según la ecuaciones 4.54 y 4.62 mediante estructuras extendidas, hay que remarcar que todas las técnicas presentadas son de carácter general y por lo tanto fácilmente aplicables a cualquiera de las formulaciones típicas de la DSM.

Por ejemplo, para calcular las ecuaciones dinámicas por medio del principio de las potencias virtuales tal y como se ven en la sección 2.3.1.4 se debería realizar el producto:

$$\sum_{k=1}^{N_S} \begin{bmatrix} \mathcal{F}^{S_k} \\ \mathcal{M}^{S_k}_{B_k} \end{bmatrix}^{\mathsf{T}} \frac{\partial}{\partial \dot{\boldsymbol{q}}} \begin{bmatrix} \mathbf{v}^{B_{S_k}}_{RI} \\ \boldsymbol{\omega}^{S_k}_{RI} \end{bmatrix} \tag{5.66}$$

Si se toma de la ecuación anterior el producto $\mathcal{F}^{S_k} \frac{\partial}{\partial \dot{\boldsymbol{q}}} \mathbf{v}^{B_{S_k}}_{RI}$ el tensor asociado a $\frac{\partial}{\partial \dot{\boldsymbol{q}}} \mathbf{v}^{B_{S_k}}_{RI}$ y el vector $\mathcal{F}^k$ puede ser representado mediante las estructuras extendidas, es decir:

$$[\![\mathcal{F}^{S_k}]\!] = m_{S_k} \cdot [\![\mathbf{a}^{G_{S_k}}_{RI}]\!] \tag{5.67}$$

y análogamente el tensor:

$$[\![\mathbf{v}^{B_{S_k}}_{RI\ \dot{\boldsymbol{q}}}]\!] = \frac{\partial}{\partial \dot{\boldsymbol{q}}} [\![\mathbf{v}^{B_{S_k}}_{RI}]\!] \tag{5.68}$$

donde $[\![\mathbf{v}^{B_{S_k}}_{RI}]\!]$ y $[\![\mathbf{a}^{G_{S_k}}_{RI}]\!]$ se calculan por composición de movimientos.

El producto

$$m_{S_k} \cdot [\![\mathbf{a}^{G_k}_{RI}]\!] \cdot [\![\mathbf{v}^{B_{S_k}}_{RI\ \dot{\boldsymbol{q}}}]\!] \tag{5.69}$$

será óptimo y no presentará ESST.

En caso de que se desee trabajar con las ecuaciones de Lagrange se procede de forma análoga. La parte asociada a la inercia de traslación para un sólido

genérico S_k usando vectores extendidos toma la forma:

$$V = \frac{1}{2} \cdot m_{S_k} \cdot [\![\mathbf{v}_{RI}^{B_{S_k}}]\!] \cdot [\![\mathbf{v}_{RI}^{B_{S_k}}]\!] \tag{5.70}$$

En la expresión anterior , al operar con vectores extendidos se evitará la aparición de ESST. De forma análoga se procede con la parte asociada a los giros del sólido.

5.9. Simplificaciones trigonométricas con origen en la suma de ángulos

En el ámbito de la DSM aparecen otro tipo de simplificaciones trigonométricas y que generalmente son debidas a giros consecutivos respecto al mismo eje. Estas son se tipo "Seno o coseno de la suma/resta de ángulos" y sobre todo aparecen en mecanismos planos.

El siguiente ejemplo ilustra cómo se realizan los giros consecutivos en torno al mismo eje y justifica de la naturaleza de ESST.

Sea el esquema de bases de la figura 5.1 en donde la base $\mathcal{B}_2$ está definida a partir de la base $\mathcal{B}_1$ mediante un giro entorno al eje Z de valor θ_1 y la base $\mathcal{B}_3$ a partir de la base $\mathcal{B}_2$ mediante un giro θ_2 entorno también al eje Z.

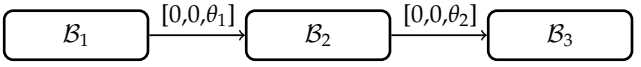


Figura 5.1.: Giros consecutivos entorno al mismo eje

La matriz de cambio que proyecta las coordenadas de un vector en la base $\mathcal{B}_1$ a la base $\mathcal{B}_3$ es:

$$\begin{aligned} \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1} = \mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_2}\,\mathbf{R}_{\mathcal{B}_2}^{\mathcal{B}_1} &= \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta_2\cos\theta_1 - \sin\theta_2\sin\theta_1 & -\cos\theta_2\sin\theta_1 - \cos\theta_1\sin\theta_2 & 0 \\ \cos\theta_2\sin\theta_1 + \cos\theta_1\sin\theta_2 & \cos\theta_2\cos\theta_1 - \sin\theta_2\sin\theta_1 & 0 \end{bmatrix} \end{aligned} \tag{5.71}$$

y aplicando las reglas de simplificación trigonométricas:

$$\mathbf{R}_{\mathcal{B}_3}^{\mathcal{B}_1} = \begin{bmatrix} \cos(\theta_2+\theta_1) & -\sin(\theta_2+\theta_1) & 0 \\ \sin(\theta_2+\theta_1) & \cos(\theta_2+\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.72}$$

La solución que aquí se propone para localizar estas simplificaciones pasa por detectar si cada vez que se genere una nueva base, $\mathcal{B}_k$, a través de un giro elemental, la base $\mathcal{B}_{k-1}$ anterior ha sido generada a través de un giro del mismo tipo a partir de la $\mathcal{B}_{k-2}$. En caso afirmativo, la nueva base $\mathcal{B}_k$ se puede definir a partir de la base $\mathcal{B}_{k-2}$ mediante un solo giro.

Para el ejemplo de la figura anterior, si se detecta que los cambios de base ocurren entorno al mismo eje, el nuevo esquema de bases sería:

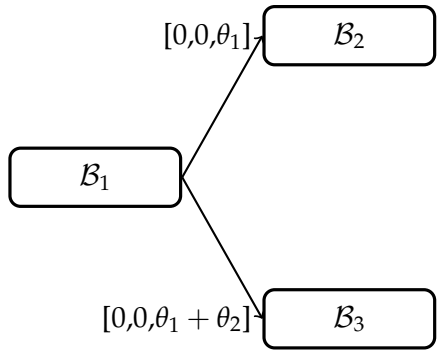


Figura 5.2.: Nuevo esquema de bases donde se han simplificado los giros

En esta figura 5.2 se ve que ahora la $\mathcal{B}_2$ está definida a partir de la $\mathcal{B}_1$ mediante un giro $\theta_1 + \theta_2$.

Esta forma de trabajar no rompe con lo presentado hasta ahora.

5.10. ESST: Mecanismo de cuatro barras

Para demostrar la eficacia del método de trabajo con vectores y tensores extendidos se propone el cálculo de la matriz de masa del mecanismo de cuatro

barras descrito en A.2.

La figura 5.3 esquematiza el citado mecanismo, donde se han suprimido los muelles y fuerzas exteriores por no jugar ningún papel en el cálculo de la citada matriz.

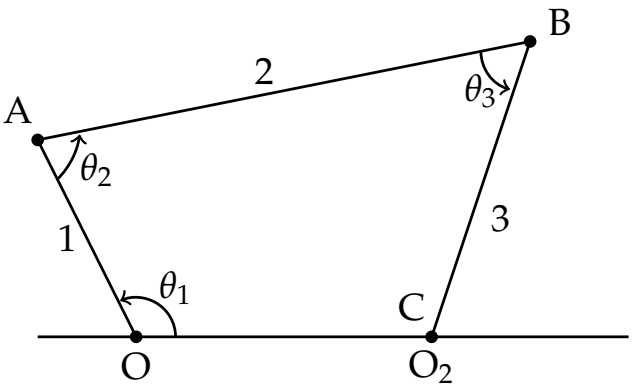


Figura 5.3.: Esquema del mecanismo

Tanto las velocidad angulares como las absolutas están formadas por sumas de vectores y donde cada uno tiene las componentes expresadas de forma natural en una base. Este hecho permite utilizar las estructuras extendidas presentadas en este capítulo para su definición.

Para la descripción de las velocidades se usa un vector extendido de cuatro elementos, por tener el mecanismo cuatro bases definidas ($\mathcal{B}_{RI}$, $\mathcal{B}_1$, $\mathcal{B}_2$ y $\mathcal{B}_3$). En el primer elemento del vector extendido se introducirán las componentes que se proyectan de forma natural en la base de la referencia inercial $\mathcal{B}_{RI}$, en el segundo elemento las que se proyectan en la base $\mathcal{B}_1$, etc... Es decir, un vector genérico $\boldsymbol{u}$ se representa como:

$$[\![\boldsymbol{u}]\!] = \left[\begin{array}{c|c|c|c} \boldsymbol{u}^{\mathcal{B}_{RI}} & \boldsymbol{u}^{\mathcal{B}_1} & \boldsymbol{u}^{\mathcal{B}_2} & \boldsymbol{u}^{\mathcal{B}_3} \end{array} \right] \tag{5.73}$$

De igual modo, se usa una estructura análoga (con cuatro elementos) para los tensores extendidos. A continuación se calculan las matrices jacobianas necesarias sólido a sólido.

5.10.0.1. Primer sólido

La velocidad absoluta del punto O del primer sólido es nula y la velocidad angular del mismo respecto a la referencia inercial es la correspondiente a un giro elemental en torno al eje perpendicular al mecanismo. La velocidad angular está proyectada de forma natural en la base asociada a dicho sólido. Ambas velocidades se representan como:

$$\begin{aligned} [\![\mathbf{v}_{RI}^{O}]\!] &= \left[\begin{array}{c|c|c|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{\omega}_{\mathcal{B}_{xyz}}^{\mathcal{B}_1}]\!] &= \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}_{\mathcal{B}_{xyz}}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} \end{array}\right] \end{aligned} \tag{5.74}$$

Las componentes velocidad angular, al tratarse de un mecanismo plano, no cambian cuando se cambia de base de proyección, porque según lo explicado en 5.7 han se ser cambiadas a la base que se encuentre más abajo en la cadena de bases, es decir:

$$[\![\boldsymbol{\omega}_{\mathcal{B}_{xyz}}^{\mathcal{B}_1}]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}_{\mathcal{B}_{xyz}}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} \end{array}\right] = \left[\begin{array}{c|c|c|c} \boldsymbol{\omega}_{\mathcal{B}_{xyz}}^{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.75}$$

Las matrices jacobianas respecto a las velocidades generalizadas:

$$\begin{aligned} [\![\mathbf{v}_{RI\,\dot{\boldsymbol{q}}}^{O}]\!] &= \frac{\partial [\![\mathbf{v}_{RI}^{O}]\!]}{\partial \dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{\omega}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}^{\mathcal{B}_1}]\!] &= \frac{\partial [\![\boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}]\!]}{\partial \dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \frac{\partial \boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}}{\partial \dot{\boldsymbol{q}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \end{aligned} \tag{5.76}$$

5.10.0.2. Segundo sólido

Las velocidades absolutas y angulares del segundo sólido representadas mediante un vector extendido son:

$$\begin{aligned} [\![\mathbf{v}_{RI}^{A}]\!] &= [\![\boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}]\!] \wedge [\![\mathbf{r}_{O}^{A}]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1} \wedge \mathbf{r}_{O}^{A} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_2}]\!] &= [\![\boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1}]\!] + [\![\boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2}]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}_{\mathcal{B}_{RI}}^{\mathcal{B}_1} & \boldsymbol{\omega}_{\mathcal{B}_1}^{\mathcal{B}_2} & \mathbf{0} \end{array}\right] \end{aligned} \tag{5.77}$$

Las componentes velocidad angular, al tratarse de un mecanismo plano, no cambian cuando se cambia de base de proyección y han se ser cambiadas a la base que se encuentre más abajo en la cadena de bases, es decir:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!] = \left[\begin{array}{c|c|c|c} \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.78}$$

Y las matrices jacobianas respecto a las velocidades generalizadas:

$$\begin{aligned} [\![\mathbf{v}^A_{RI\dot{\boldsymbol{q}}}]\!] &= \frac{\partial\,[\![\mathbf{v}^A_{RI}]\!]}{\partial\dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \mathbf{0} & \frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}\wedge\mathbf{r}^A_O\right)}{\partial\dot{\boldsymbol{q}}} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] &= \frac{\partial\,[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}}]\!]}{\partial\dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}\right)}{\partial\dot{\boldsymbol{q}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \end{aligned} \tag{5.79}$$

5.10.0.3. Tercer sólido

Por último, las velocidades absolutas y angulares del tercer sólido se representan mediante las siguientes estructuras extendidas. El origen de las expresiones puede ser obtenido a partir de la figura procediendo análogamente a como se ha realizado con el sólido dos.

$$\begin{aligned} [\![\mathbf{v}^B_{RI}]\!] &= \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}\wedge\mathbf{r}^A_O & \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}\wedge\mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}\wedge\mathbf{r}^B_A & \mathbf{0} \end{array}\right] \\ [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}}]\!] &= \left[\begin{array}{c|c|c|c} \mathbf{0} & \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} & \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} & \boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2} \end{array}\right] \end{aligned} \tag{5.80}$$

De igual modo a los casos anteriroes, las componentes velocidad angular han se ser cambiadas a la base que se encuentre más abajo en la cadena de bases, es decir:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}}]\!] = \left[\begin{array}{c|c|c|c} \boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} + \boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.81}$$

Y las matrices jacobianas respecto a las velocidades generalizadas:

$$
\begin{aligned}
[\![\mathbf{v}^B_{RI\,\dot{\boldsymbol{q}}}]\!] &= \frac{\partial [\![\mathbf{v}^B_{RI}]\!]}{\partial \dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \mathbf{0} & \frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O\right)}{\partial \dot{\boldsymbol{q}}} & \frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A\right)}{\partial \dot{\boldsymbol{q}}} & \mathbf{0} \end{array}\right] \\
[\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] &= \frac{\partial\, [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}})]\!]}{\partial \dot{\boldsymbol{q}}} = \left[\begin{array}{c|c|c|c} \frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} + \boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2}\right)}{\partial \dot{\boldsymbol{q}}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right]
\end{aligned}
\tag{5.82}
$$

5.10.1. Cálculo de la matriz de masa

La matriz de masa de este mecanismo se ha calculado como se propone en la ecuación 4.54.

Las operaciones entre tensores extendidos se han de realizar siguiendo las reglas ya comentadas y hay que tener en cuanta los distintos cambios de base que puedan aparecer.

$$
\begin{aligned}
\mathbf{M} = & -m_1 \cdot [\![\mathbf{v}^O_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^O_{RI\,\dot{\boldsymbol{q}}}]\!] - [\![\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^1_O]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] \\
& - [\![\mathbf{v}^O_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_1 \cdot [\![\widetilde{\mathbf{r}^{G_1}_O}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] - \left([\![\mathbf{v}^O_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_1 \cdot [\![\widetilde{\mathbf{r}^{G_1}_O}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]\right)^{\mathsf{T}} \\
& - m_2 \cdot [\![\mathbf{v}^A_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^A_{RI\,\dot{\boldsymbol{q}}}]\!] - [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^2_A]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] \\
& - [\![\mathbf{v}^A_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_2 \cdot [\![\widetilde{\mathbf{r}^{G_2}_A}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] - \left([\![\mathbf{v}^A_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_2 \cdot [\![\widetilde{\mathbf{r}^{G_2}_A}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]\right)^{\mathsf{T}} \\
& - m_3 \cdot [\![\mathbf{v}^B_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^B_{RI\,\dot{\boldsymbol{q}}}]\!] - [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^3_B]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] \\
& - [\![\mathbf{v}^B_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_3 \cdot [\![\widetilde{\mathbf{r}^{G_3}_B}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!] - \left([\![\mathbf{v}^B_{RI\,\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_3 \cdot [\![\widetilde{\mathbf{r}^{G_3}_B}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_{RI}\,\dot{\boldsymbol{q}}}]\!]\right)^{\mathsf{T}}
\end{aligned}
\tag{5.83}
$$

Para ilustrar cómo se realizarían las operaciones con las estructuras de tensor extendido se ha tomado los términos de la ecuación anterior más representativos de las casuísticas comentadas en las sección 5.8.

5.10.1.1. Velocidad absoluta

En primer lugar se va a ver el producto que implica las velocidades lineales y la masa. Para este ejemplo se ha tomado la parte correspondiente al sólido tres[6]:

$$m_3 \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!] \tag{5.84}$$

Si se desarrollan las estructuras extendidas[7] :

$$\begin{aligned}
m_3 \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!] = {} & m_3 \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right) + \\
& m_3 \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right) + \\
& m_3 \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right) + \\
& m_3 \left(\frac{\partial(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \left(\frac{\partial(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right) = \\
& m_3 \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_1} \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_1} + \\
& m_3 \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_1} \mathbf{R}^{\mathcal{B}_1}_{\mathcal{B}_2} \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_2} + \\
& m_3 \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_2} \mathbf{R}^{\mathcal{B}_2}_{\mathcal{B}_1} \left\{\frac{\partial\,(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial \dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_1} + \\
& m_3 \left\{\frac{\partial(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_2} \left\{\frac{\partial(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^B_A + \boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1} \wedge \mathbf{r}^B_A)}{\partial \dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_2}
\end{aligned} \tag{5.85}$$

En la ecuación anterior los términos primero y último se han de realizar sin cambio de base. En los términos segundo y tercero sí que hay que hacer un

[6]Por simplicidad se ha suprimido el signo negativo.

[7]Nótese el carácter tensorial de los elementos. A la hora de realizar las operaciones, dichos tensores se deberían proyectar a un *base común*.

cambio de base. Daría igual en qué base realizar la proyección, siempre que sea las base $\mathcal{B}_2$ o la base $\mathcal{B}_3$. Solo de esta manera se evitarán la aparición de expresiones que se puedan simplificar.

Siendo el resultado final:

$$m_3 \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{v}^B_{RI\dot{\boldsymbol{q}}}]\!] = \begin{bmatrix} m_3(2l_2\cos(\theta_2)l_1 + l_1^2 + l_2^2) & m_3(l_2\cos(\theta_2)l_1 + l_1^2) & 0 \\ m_3(l_2\cos(\theta_2)l_1 + l_1^2) & m_3 l_2^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.86}$$

5.10.1.2. Velocidad angular

en este caso se presenta un producto que implica al tensor de inercia y la velocidad angular. Para este ejemplo se ha tomado la contribución del sólido dos.

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^2_A]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] \tag{5.87}$$

Donde $[\![\mathbf{I}^2_A]\!]$ se define de forma natural en la base $\mathcal{B}_2$.

$$[\![\mathbf{I}^2_A]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \mathbf{0} & \mathbf{I}^2_A & \mathbf{0} \end{array}\right] \tag{5.88}$$

La componente del tensor inercial, al tratarse de un mecanismo plano, no cambia cuando se cambia de base de proyección y por lo tanto ha se ser cambiada a la base que se encuentre más abajo en la cadena de bases, es decir:

$$[\![\mathbf{I}^2_A]\!] = \left[\begin{array}{c|c|c|c} \mathbf{I}^2_A & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \tag{5.89}$$

Si se desarrollan las estructuras extendidas de este producto:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^2_A]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] = \left(\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}+\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2}\right)}{\partial\dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \mathbf{I}^2_A \left(\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}+\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2}\right)}{\partial\dot{\boldsymbol{q}}}\right) = \left\{\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}+\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2}\right)}{\partial\dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_{RI}} \mathbf{I}^2_A \left\{\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}+\boldsymbol{\omega}^{\mathcal{B}_3}_{\mathcal{B}_2}\right)}{\partial\dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_{RI}} \tag{5.90}$$

Siendo el resultado final:

$$[\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\mathbf{I}^2_A]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] = \begin{bmatrix} I_2 & I_2 & 0 \\ I_2 & I_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.91}$$

5.10.1.3. Vector centro de masa

Por último, se estudia el producto donde está involucrado en vector del centro de gravedad de sólido. Para este ejemplo, por simplicidad de ha tomado el referido al sólido dos:

$$[\![\mathbf{v}^A_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot m_2 \cdot [\![\widetilde{\mathbf{r}^{G_2}_A}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] \tag{5.92}$$

Donde:

$$[\![\mathbf{r}^{G_2}_A]\!] = \left[\begin{array}{c|c|c|c} \mathbf{0} & \mathbf{0} & \widetilde{\mathbf{r}^{G_2}_A} & \mathbf{0} \end{array}\right] \tag{5.93}$$

Si se desarrollan las estructuras extendidas de este producto:

$$m_2 \cdot [\![\mathbf{v}^A_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\widetilde{\mathbf{r}^{G_2}_A}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] = m_2 \left(\frac{\partial(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial\dot{\boldsymbol{q}}}\right)^{\mathsf{T}} \widetilde{\mathbf{r}^{G_2}_A} \left(\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}\right)}{\partial\dot{\boldsymbol{q}}}\right) = m_2 \left\{\frac{\partial\ (\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}} \wedge \mathbf{r}^A_O)}{\partial\dot{\boldsymbol{q}}}\right\}^{\mathsf{T}}_{\mathcal{B}_1} \mathbf{R}^{\mathcal{B}_2}_{\mathcal{B}_1} \left\{\widetilde{\mathbf{r}^{G_2}_A}\right\}_{\mathcal{B}_2} \mathbf{R}^{\mathcal{B}_{RI}}_{\mathcal{B}_2} \cdot \left\{\frac{\partial\left(\boldsymbol{\omega}^{\mathcal{B}_1}_{\mathcal{B}_{RI}}+\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_1}\right)}{\partial\dot{\boldsymbol{q}}}\right\}_{\mathcal{B}_{RI}} \tag{5.94}$$

Siendo el resultado final:

$$m_2 \cdot [\![\mathbf{v}^A_{RI\dot{\boldsymbol{q}}}]\!]^{\mathsf{T}} \cdot [\![\widetilde{\mathbf{r}^{G_2}_A}]\!] \cdot [\![\boldsymbol{\omega}^{\mathcal{B}_2}_{\mathcal{B}_{RI}\dot{\boldsymbol{q}}}]\!] = \begin{bmatrix} m_2(CG_2^x\cos(\theta_2)+\sin(\theta_2)CG_2^z)l_1 & m_2(CG_2^x\cos(\theta_2)+\sin(\theta_2)CG_2^z)l_1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (5.95)$$

5.10.1.4. Comparación

Para demostrar la potencia de esta técnica, se muestra el primer elemento de la matriz de masa en dos casos: cuando se evita la aparición de ESST y cuando no se evita.

La expresiones se muestran en extenso y las simplificaciones asociadas a "suma de ángulos" no se han realizado.

La primera ecuación corresponde al citado elemento cuando se utilizan técnicas que evitan la aparición de ESST.

$$\begin{aligned} \mathbf{M}(1,1) = & -2\, l_1\, m_2\, (\sin(\theta_2)\, CG_2^z + CG_2^x\, \cos(\theta_2)) \\ & + 2\, l_1\, (CG_3^x\, (\sin(\theta_2)\, \sin(\theta_3) - \cos(\theta_3)\, \cos(\theta_2)) \\ & - (\cos(\theta_2)\, \sin(\theta_3) + \sin(\theta_2)\, \cos(\theta_3))\, cg3z)\, m_3 - l_1^2\, m_2 \\ & - I_3 - I_2 - I_1 - 2\, (CG_3^x\, \cos(\theta_3) + \sin(\theta_3)\, CG_3^z)\, m_3\, l_2 \\ & - (l_2^2 + 2\, l_1\, \cos(\theta_2)\, l_2 + l_1^2)\, m_3 + l_2\, \cos(\theta_2)) + l_2\, \sin(\theta_2)\, \cos(\theta_1)) \end{aligned}$$

La siguiente muestra el citado elemento cuando no se han utilizado técnicas para evitar las ESST. En algunas partes, se pueden ver elementos (senos y cosenos) elevados al cuadrado, estos elementos, entre otros, son los susceptibles de simplificación trigonométrica. En otras partes de la misma ecuación, no aparecen de forma tan explicita.

$$
\begin{aligned}
\mathbf{M}(1,1) = & -m_2\ (l_1^2\ \cos(\theta_1)^2 + l_1^2\ \sin(\theta_1)^2) - m_3\ ((l_2\ \sin(\theta_1)\ \sin(\theta_2) \\
& - (l_1 + l_2\ \cos(\theta_2))\ \cos(\theta_1))^2 + (\sin(\theta_1)\ (l_1 + l_2\ \cos(\theta_2)) \\
& + l_2\ \sin(\theta_2)\ \cos(\theta_1))^2) - I_3 - I_2 \\
& - 2\ (CG_2^z\ m_2\ (\sin(\theta_1)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_1)) \\
& + (\sin(\theta_2)\ \cos(\theta_1) + \cos(\theta_2)\ \sin(\theta_1))\ m_2\ CG_2^x)\ l_1\ \sin(\theta_1) \\
& - 2\ l_1\ ((\sin(\theta_2)\ \cos(\theta_1) + \cos(\theta_2)\ \sin(\theta_1))\ CG_2^z\ m_2 \\
& - m_2\ CG_2^x\ (\sin(\theta_1)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_1)))\ \cos(\theta_1) \\
& + 2\ (m_3\ (\cos(\theta_1)\ (\cos(\theta_2)\ \sin(\theta_3) + \sin(\theta_2)\ \cos(\theta_3)) \\
& - \sin(\theta_1)\ (\sin(\theta_3)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_3)))\ CG_3^z \\
& - m_3\ CG_3^x\ (\sin(\theta_1)\ (\cos(\theta_2)\ \sin(\theta_3) + \sin(\theta_2)\ \cos(\theta_3)) \\
& + \cos(\theta_1)\ (\sin(\theta_3)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_3))))) \\
& (l_2\ \sin(\theta_1)\ \sin(\theta_2) - (l_1 + l_2\ \cos(\theta_2))\ \cos(\theta_1)) - I_1 \\
& - 2\ (m_3\ CG_3^x\ (\cos(\theta_1)\ (\cos(\theta_2)\ \sin(\theta_3) + \sin(\theta_2)\ \cos(\theta_3)) \\
& - \sin(\theta_1)\ (\sin(\theta_3)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_3))) \\
& + m_3\ (\sin(\theta_1)\ (\cos(\theta_2)\ \sin(\theta_3) + \sin(\theta_2)\ \cos(\theta_3)) \\
& + \cos(\theta_1)\ (\sin(\theta_3)\ \sin(\theta_2) - \cos(\theta_2)\ \cos(\theta_3)))\ CG_3^z) \\
& (\sin(\theta_1)\ (l_1 + l_2\ \cos(\theta_2)) + l_2\ \sin(\theta_2)\ \cos(\theta_1))
\end{aligned}
$$

CAPÍTULO 6

RESULTADOS Y ANÁLISIS

Con el objetivo de probar y de validar los algoritmos y técnicas presentados en esta tesis, en este capítulo se presentan los resultados de los análisis realizados para el conjunto de sistemas multicuerpo descritos en detalle en el Anexo A.

Se trata de cinco sistemas: Bloque-Péndulo (A.1), Cuatro Barras (A.2), Biela-Manivela-Disco (A.3), Plataforma Stewart (A.4) y la Locomotora FEVE (A.5). Los sistemas modelados presentan un creciente grado de complejidad.

Todos los algoritmos y técnicas presentados en esta tesis han sido implementados en la librería simbólica lib_3D_MEC-GiNaC.

6.1. Algoritmos genéricos: atomización, sustitución y derivación

En esta sección se analizan los algoritmos de tipo genérico, es decir, las técnicas de atomización basadas en tablas hash (Sección 3.4), el algoritmo de sustitución recursiva (Sección 3.5) y el algoritmo diferenciación recursiva (Sección

3.6).

A este fin, las técnicas de atomización desarrolladas se ha comparado con la versión previa a la realización de esta tesis existente en la librería lib_3D_MEC-GiNaC y que no estaba basada en una búsqueda con tablas hash, si no que estaba basada en un proceso de sustitución de átomos mediante "fuerza bruta" (Sub-sección 3.2.1).

De igual manera, los algoritmos de sustitución y derivación recursivos se han comparado con sus respectivas versiones no recursivas y no optimizadas para trabajar con expresiones atomizadas. En las versiones implementadas, previas a esta tesis, estos algoritmos requieren un pre-proceso de desatomización y un post-proceso de atomización.

Nótese que al trabajar con expresiones atomizadas, los algoritmos de sustitución y derivación tienen una fuerte dependencia con las técnicas de atomización que se utilicen.

Se han modelado simbólicamente los cinco ejemplos citados previamente y se han estudiado las siguientes tres combinaciones:

Caso A Atomización **sin** búsqueda basada en tablas hash. Algoritmos de sustitución y diferenciación **no recursivos**.

Caso B Atomización **con** búsqueda basada en tablas hash. Algoritmos de sustitución y diferenciación **no recursivos**.

Caso C Atomización **con** búsqueda basada en tablas hash. Algoritmos de sustitución y diferenciación **recursivos** y **optimizados** para trabajar con expresiones atomizadas.

Para cada uno de estos tres casos, se ha contado el número total de operaciones algebraicas necesarias para la evaluación numérica de las matrices simbólicas ($\mathbf{M}$ y $\dot{\mathbf{\Phi}}_{\dot{q}}$) y vectores simbólicos (δ y γ) necesarios para la resolución del problema dinámico. La matriz $\mathbf{M}$ y vector δ se han calculado por medio de las ecuaciones presentadas en esta tesis (Ecuaciones 4.54 y 4.62) y el código ha sido generado por medio de los algoritmos protestos en la sección 3.7. Todos los cálculos se han realiza con el modificador "Gravedad UP". La tabla 6.1 muestra los datos obtenidos.

En la tabla 6.1 se puede ver como en general, la introducción de los nuevos algoritmos y técnicas de atomización hace que el número de operaciones

	Caso A				Caso B				Caso C			
Modelo	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ
Bloque-Péndulo	8	16	0	0	8	16	0	0	8	16	0	0
Cuatro Barras	64	250	48	87	66	253	46	80	63	212	42	76
Biela-Manivela-Disco	37	123	121	279	137	128	90	152	37	123	123	226
Stewart	496	2703	393	1052	492	2793	410	1175	486	2667	688	917
Locomotora	-	-	-	-	13825	48007	10002	47727	12782	26626	8449	28202

Tabla 6.1.: Evaluación de algoritmos genéricos. Operaciones algebraicas

necesarias para evaluar las diferentes matrices sea menor que cuando no se introducen estos.

En general, en todos los modelos a excepción del primero (debido a su sencillez), la evaluación numérica de la matriz $\mathbf{M}$ y del vector δ requiere menos operaciones en el **Caso C**, que en el caso **Caso B**. Se observa también, que en los modelos "sencillos" (todos exceptuando el modelo **Locomotora**) la diferencia en operaciones algebraicas entre los casos **Caso A** y **Caso C** es muy pequeña. Esto es debido a que la atomización "por fuerza bruta" funciona bien únicamente para modelos de tamaño pequeño.

Aun así, en los modelos **Biela-Manivela-Disco** y **Stewart** se observa una diferencia en el número de operaciones necesarias para evaluar $\dot{\mathbf{\Phi}}_{\dot{q}}$ y γ entre el caso **Caso C** y los dos otros casos. Esto es debido a que el vector $\dot{\mathbf{\Phi}}$ se ha obtenido mediante derivación a partir del vector $\mathbf{\Phi}$ y esta no es la forma óptima de hacerlo. En los casos **Caso A** y **Caso B** las técnicas de derivación y sustitución no recursivas funcionan bien, ya que estas se delegan al motor simbólico. Cuando las expresiones simbólicas son sencillas el motor simbólico es capaz de realizar ciertas optimizaciones que los algoritmos aquí presentados no realizan.

En la tabla 6.2 se recoge, para los tres mismos casos anteriores, el tiempo total necesario para calcular y exportar las matrices y vectores simbólicos necesarios para la simulación dinámica, según se describen en la sección 2.6.

En la tabla 6.2 se observa que el tiempo de cálculo es mayor en el **Caso A** llegando a ser prácticamente infinito (más de cuatro días) para el modelo **Locomotora**. La introducción de las técnicas de atomización basadas en búsqueda con tabla hash (**Caso B**) hace que el tiempo de cálculo sea notablemente menor, llegando a ser, para el modelo **Stewart** hasta un 0.5 % del tiempo necesario en el **Caso A**. Para el **Caso B** el modelo **Locomotora** sí que puede ser

Modelo	Caso A	Caso B	Caso C
Bloque-Péndulo	0.067	0.035	0.026
Cuatro Barras	3.896	0.176	0.116
Biela-Manivela-Disco	1.8230	0.210	0.151
Stewart	564.317	3.225	2.522
Locomotora	∞	1651.685	849.11

Tabla 6.2.: Evaluación de algoritmos genéricos: Tiempo

realizado. La introducción de los algoritmos de sustitución y derivación recursivas (**Caso C**) mejora también el tiempo total de cálculo. Por ejemplo, para el modelo **Locomotora** en el **Caso C** pasa a ser, aproximadamente, un 40 % del tiempo necesario en el **Caso B**.

6.2. Atomización "sobre la marcha".Generación de código

En las siguientes tablas se hace un análisis de la combinación de la técnicas de atomización "sobre la marcha" y los algoritmos de generación de código propuestos en la sección 3.7.

A este fin se modelan los cinco ejemplos descritos anteriormente en los dos siguientes casos:

Caso C.1 Generación simbólica **sin** atomización. Atomización y generación de código realizada "a posteriori" mediante en el software Maple.

Caso C.2 Generación simbólica **con** atomización "sobre la marcha" basada en tablas hash. Generación de código mediante los algoritmos propuestos en la sección 3.7.

En ambos casos se utilizan los algoritmos de sustitución y de derivación recursivos propuestos en esta tesis. De igual manera, en ambos casos, el cálculo de la matriz **M** y del vector δ se ha realizado según las ecuaciones 4.54 y 4.62.

En la tabla 6.3 se realiza la comparación, para los dos casos anteriores, del tiempo de cálculo y de generación de código de los cinco sistemas a modelar.

Modelo	Caso C.1	Caso C.2
Bloque-Péndulo	1.082	0.026
Cuatro Barras	1.1520	0.158
Biela-Manivela-Disco	1.529	0.151
Stewart	5.191	2.522
Locomotora	7658.896	849.11

Tabla 6.3.: Tiempos de ejecución en segundos

En la tabla 6.3 puede observarse que la combinación de técnicas de atomización "sobre la marcha" y los algoritmos de exportación presentados en esta tesis (**Caso C.2**) es mucho más rápida que las basadas en un post-proceso con un software simbólico tipo Maple (**Caso C.1**) y , en general, con cualquier otro software simbólico que se desee usar. Por ejemplo, para el modelo **Locomotora** el tiempo necesario en el **Caso C.2** es aproximadamente un 11 % del necesario en el **Caso C.1**.

	Caso C.1				Caso C.2			
Modelo	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ
Bloque-Péndulo	1	3	0	0	1	3	0	0
Cuatro Barras	17	50	17	25	13	51	16	24
Biela-Manivela-Disco	10	27	23	35	4	30	23	41
Stewart	52	503	78	203	66	455	114	214
Locomotora	2396	5563	1470	7040	2076	4740	1144	5195

Tabla 6.4.: Número de átomos

	Caso C.1				Caso C.2			
Modelo	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ	**M**	δ	$\dot{\mathbf{\Phi}}_{\dot{q}}$	γ
Bloque-Péndulo	8	16	0	0	8	16	0	0
Cuatro Barras	72	244	42	76	63	212	42	76
Biela-Manivela-Disco	33	111	73	137	37	123	123	226
Stewart	473	2857	368	927	486	2667	618	917
Locomotora	13922	31174	7252	34818	12782	26626	8449	28202

Tabla 6.5.: Número de operaciones

En las tablas 6.4 y 6.5 se comparan respectivamente, para los dos casos an-

teriores, el número de átomos y el número de operaciones algebraicas que requiere la evaluación de la matriz **M** y del vector δ. Se observa que para ciertos modelos sencillos (**Biela-Manivela-Disco** y **Stewart**), en ciertos elementos (por ejemplo la matriz $\dot{\mathbf{\Phi}}_{\dot{q}}$ en el modelo **Stewart**) el software simbólico Maple (**Caso C.1**) es capaz de optimizar mejor el código generado. En el modelo **Locomotora**, debido a su tamaño, este hecho no ocurre y el número de operaciones necesarias para evaluar los diferentes elementos es sustancialmente menor en el **Caso C.2** que en **Caso C.1**. Por ejemplo, para el vector δ el número de operaciones el casi un 20 % en el **Caso C.2** que en textbfCaso C.1.

6.2.1. Exportación conjunta

La tabla 6.6 presenta para los mismos cinco ejemplos el número de operaciones algebraicas necesarias para evaluar numéricamente la matriz **M** y del vector δ en dos casos:

Caso C.2.1 La matriz **M** y del vector δ se exportan por separado.

Caso C.2.2 La matriz **M** y del vector δ se exportan de forma conjunta, tal y como se propone en la sub-sección 3.7.5.

En ambos casos, se han utilizado las técnicas de atomización "sobre la marcha" presentadas en esta tesis. Así como los algoritmos de sustitución y de derivación recursivos también propuestos en esta tesis. De igual manera, en ambos casos, el cálculo de la matriz **M** y del vector δ se ha realizado según las ecuaciones 4.54 y 4.62.

	Caso C.2.1			Caso C.2.2
Modelo	**M**	δ	**Total**	**M**δ
Bloque-Péndulo	8	16	24	21
Cuatro Barras	63	212	275	264
Biela-Manivela-Disco	37	123	160	150
Stewart	486	2667	3153	3106
Locomotora	12782	26626	39408	36695

Tabla 6.6.: Número de operaciones necesarias para la evaluación de **M** y δ

Por ejemplo, según la tabla 6.6 para evaluar la matriz $\mathbf{M}$ y el vector $\boldsymbol{\delta}$ en el modelo **Locomotora** se han de realizar un total de $12782 + 26626 = 39408$ operaciones, frente a 36695 operaciones que se han de realizar si se exportasen de forma conjunta. Es decir, casi un 7 % menos de operaciones. Lo que justifica por sí solo la necesidad de exportar estos dos elementos de forma conjunta.

6.2.1.1. Comparación con Robotran

Para comparar el código generado con las técnica propuestas en esta tesis con el generado por el software Robotran [11] se ha generado para el mecanismo de **Cuatro Barras** el código de la matriz $[\boldsymbol{\Phi}_q|\boldsymbol{\Phi}]$, formada por $\boldsymbol{\Phi}_q$ y $\boldsymbol{\Phi}$ según se explica en la sub-sección 3.7.5.

Se han analizado dos casos:

Caso C.2 Mediante la técnicas de atomización "sobre la marcha", el operador "Vector de Posición", para el cálculo $\boldsymbol{\Phi}$, y los algoritmos de derivación recursiva, para el cálculo del $\boldsymbol{\Phi}_q$.

Caso Robotran Mediante Robotran.

En la tabla 6.7 se ve que el código generado en el **Caso C.2** tiene menos número de átomos y requiere menos operaciones para ser evaluado numéricamente que el generado por Robotran.

	Átomos	Operaciones
Caso C.2	18	43
Caso Robotran	22	47

Tabla 6.7.: Comparación del tamaño del código de $\boldsymbol{\Phi}_q$ y $\boldsymbol{\Phi}$ exportados conjuntamente

En al anexo B.3 se puede ver el código generado por Robotran y el generado con la librería lib_3D_MEC-GiNaC con los algoritmos y técnicas presentados en esta tesis.

Como se verá, en secciones posteriores se realiza una comparación de análoga con la matriz $\mathbf{M}$ y el vector $\boldsymbol{\delta}$.

6.3. Evaluación numérica

En esta sección se presenta el tiempo en segundos de evaluación numérica de la matriz **M** y el vector δ generados para los cinco ejemplos descritos al inicio de esta capítulo. La evaluación numérica ha sido realizada en MATLAB.

Se plantean diversos casos:

Caso D Generación simbólica **sin** atomización.

Caso C.2.1 Generación simbólica **con** atomización "sobre la marcha" basada en tablas hash. Generación de código mediante los algoritmos propuestos en la sección 3.7.

Caso C.2.2 Generación simbólica **con** atomización "sobre la marcha" basada en tablas hash. Generación de código mediante los algoritmos propuestos en la sección 3.7. **Exportación conjunta** de **M** y δ.

En los tres casos se utilizan los algoritmos de sustitución y de derivación recursivos propuestos en esta tesis. De igual manera, en los tres casos, el cálculo de la matriz **M** y del vector δ se ha realizado según las ecuaciones 4.54 y 4.62.

Modelo	Caso D	Caso C.2.1	Caso C.2.2
Bloque-Péndulo	12.504	12.221	5.999
Cuatro barras	33.441	18.636	13.052
Biela-Manivela-Disco	24.863	17.514	10.004
Stewart	176.65	106.23	95.804
Locomotora	-	1004.9	908.083

Tabla 6.8.: Tiempo de evaluación de **M** y δ en milisegundos

La tabla 6.8 por sí sola justifica la necesidad de generar código simbólico altamente optimizado. Como se ve en la misma tabla, el tiempo necesario para la evaluación numérica de **M** y δ es muy superior en el caso **Caso D** que en los otros dos casos. Por ejemplo, en el modelo **Stewart** el tiempo de evaluación numérica es un 47 % inferior al caso en que no se utilizan técnicas atomización (**Caso D**). Para el modelo **Locomotora** el software MATLAB no ha sido capaz de realizar la evaluación numérica de la matriz cuando ha sido generada sin atomizar. El tamaño del fichero es aproximadamente de 250 *MB* y el software no es capaz de cargarlo en memoria.

Igualmente, la misma tabla justifica por sí sola la necesidad de exportar conjuntamente los elementos **M** y δ ya que este hecho permite una más rápida evaluación de los elementos. Como ya se ha visto en la tabla 6.6 estos dos elementos pueden llegar a compartir, en el modelo **Locomotora**, hasta un 7 % de operaciones. Esto directamente se traduce un ahorro de tiempo de evaluación del mismo orden. Así, para este mismo modelo, el tiempo de evaluación numérica es aproximadamente un 10 % inferior cuando se exportan de forma conjunta (**Caso C.2.2**) que cuando no (**Caso C.2.1**). Pese a que el número de operaciones es un 7 % inferior, el tiempo de evaluación es un 10 % inferior, esto es debido a que en el **Caso C.2.1** se ha evaluar dos funciones y en el **Caso C.2.2** únicamente una. El propio hecho de cargar en memoria una función consume tiempo adicional. Esta es otra de la razones que justifican la exportación conjunta de estos elementos.

6.4. Cálculos cinemáticos. Parentesización

En esta sección se analizan los operadores cinemáticos con la intención de ver cómo estos maximizan la parentesización.

A tal fin, se calcula el vector de posición $\mathbf{r}_O^C$ en el modelo **Cuatro Barras** descrito en A.2.

Con el modificador "Gravedad DOWN" se obtienen las componentes del resultado proyectadas en la base absoluta ($\mathcal{B}_{xyz}$). En cambio, con el modificador "Gravedad UP" se obtienen las componentes del resultado proyectadas en la base del tercer brazo ($\mathcal{B}_3$).

El vector ha sido calculado mediante dos métodos:

- Mediante la suma de los vectores $\mathbf{r}_O^C = \mathbf{r}_O^A + \mathbf{r}_A^B + \mathbf{r}_B^C$.
- Mediante el operador "Vector de Posición".

Modelo	DOWN	UP
Suma de vectores	64	21
Operador "Vector de Posición"	54	21

Tabla 6.9.: Operaciones necesarias para la evaluación

Como se ve en la tabla 6.9 cuando se han obtenido las componentes proyectadas en la base $\mathcal{B}_{xyz}$, el operador "Vector de Posición" genera expresiones simbólicas que requieren un 20 % menos de operaciones para su evaluación. En sistemas más grandes, esta diferencia es mayor.

En este caso en particular, el modificador "Gravedad UP" no tiene influencia, y ya en ambos casos (mediante suma de vectores o mediante el operador "Vector de Posición") se realizan las mismas operaciones en el mismo orden. Por lo tanto el resultado es el mismo.

En la sección B.1 se muestran las tres componentes de dicho vector proyectadas en la base $\mathcal{B}_{xyz}$ y en la base $\mathcal{B}_3$, cuando han sido calculadas mediante los dos métodos.

6.5. Modificador "Gravedad". Reciclabilidad de átomos

Para analizar la influencia del modificador "Gravedad" se ha calculado la matriz $\mathbf{M}\boldsymbol{\delta}$ ($\mathbf{M}$ y $\boldsymbol{\delta}$ exportados de forma conjunta) teniendo en cuenta las dos posibilidades de este modificador: "Gravedad UP" y "Gravedad DOWN".

La tabla 6.10 presenta para los cinco ejemplos el número de operaciones algebraicas necesarias para evaluar numéricamente la matriz $\mathbf{M}\boldsymbol{\delta}$ en dos casos:

Caso C.2.DOWN Modificador "Gravedad UP"

Caso C.2.UP Modificador "Gravedad DOWN"

En ambos casos, se han utilizado las técnicas de atomización "sobre la marcha" presentadas en esta tesis. Así como los algoritmos de sustitución y de derivación recursivos también propuestos en esta tesis. De igual manera, en ambos casos, el cálculo de la matriz $\mathbf{M}$ y del vector $\boldsymbol{\delta}$ se ha realizado según las ecuaciones 4.54 y 4.62.

Como se ve en la tabla, en general, el número de operaciones necesarias para evaluar numéricamente la matriz $\mathbf{M}\boldsymbol{\delta}$ es inferior cundo ha sido generada utilizado modificador "Gravedad UP" (**Caso C.2.UP**). La diferencia entre los dos métodos depende de la topología del modelo. Por ejemplo, para el caso del modelo **Stewart**, el número de operaciones es aproximadamente un 37 % inferior, en cambio, para el modelo **Locomotora** esta diferencia es menor, un

Modelo	Caso C.2.DOWN	Caso C.2.UP
Bloque-Péndulo	21	21
Cuatro barras	303	264
Biela-Manivela-Disco	164	150
Stewart	4923	3106
Locomotora	39170	36695

Tabla 6.10.: Número de operaciones necesarias para evaluar numéricamente $\mathbf{M}\delta$

7 % menos de operaciones entre el **Caso C.2.UP** y el **Caso C.2.DOWN**. Estas diferencias entre los dos casos son debidas a que el modificador "Gravedad UP" maximiza la reciclabilidad de átomos.

6.6. Matriz M y vector δ

Otros de los objetivos de esta tesis ha sido el de aplicar el PPV sistemas de mediano y gran tamaño.

En esta sección, se ha procedido al cálculo de la matriz **M** y del vector δ de los cinco ejemplos descritos al inicio del capítulo en los siguientes dos casos:

Caso C.2.2.1 La matriz **M** y vector δ se han calculado mediante las ecuaciones prpopuestas en esta tesis (Ecuaciones 4.54 y 4.62).

Caso C.2.2.2 La matriz **M** y vector δ se calcular a partir de las ecuaciones de la dinámica obtenidas mediante el PPV (Sub-sección: 2.3.1.4). Es decir, primero se han de calcular los torsores actuantes sobre el sólido (Sub-sección: 2.3.1.3). A continuación, por medio del PPV, se ha de calcular las ecuaciones de la dinámica. Y por último, derivar estas ecuaciones respecto a las aceleraciones generalizadas para calcular **M** y sustituir en las ecuaciones de la dinámica las aceleraciones generalizadas por un valor nulo para calcular δ tal y como se describe en 2.7.2.

En ambos casos se utilizan los algoritmos de sustitución y de derivación recursivos propuestos en esta tesis. Los cálculos cinemáticos necesarios han sido realizados por medio de los operadores cinemáticos con el modificador "Gravedad UP". De igual manera, en ambos casos, se utilizan técnicas de ato-

mización "sobre la marcha" y el código ha sido generado por medio de los algoritmos protestos en la sección 3.7. La matriz $\mathbf{M}$ y el vector δ han sido exportados de forma conjunta.

Modelo	Caso C.2.2.1	Caso C.2.2.2
Bloque-Péndulo	0.017	0.017
Cuatro Barras	0.050	0.055
Biela-Manivela-Disco	0.030	0.037
Stewart	0.697	1.7020
Locomotora	132.213	4709.2

Tabla 6.11.: Tiempo en segundos de cálculo de $\mathbf{M}$ y δ

En la tabla 6.11 se muestra, para los cinco modelos, el tiempo en segundos necesario para calcular la matriz $\mathbf{M}$ y el vector δ en los dos casos descritos.

Se observa que el cálculo en el **Caso C.2.2.1** es más rápido que en el **Caso C.2.2.2** y que conforme crece la complejidad del modelo, mayor es la diferencia. Por ejemplo, para el modelo **Locomotora**, el tiempo de cálculo en el **Caso C.2.2.1** es aproximadamente 3 % del tiempo necesario en el **Caso C.2.2.2**. Este hecho justifica por sí solo el método presentado en esta tesis. De igual manera, justifica por qué el PPV (**Caso C.2.2.2**) no es adecuado para sistemas de mediano y gran tamaño, si no se aplica de una manera optimizada.

Modelo	Caso C.2.2.1	Caso C.2.2.2
Bloque-Péndulo	22	23
Cuatro Barras	264	353
Biela-Manivela-Disco	150	197
Stewart	3106	3375
Locomotora	36695	60474

Tabla 6.12.: Número de operaciones para evaluar numéricamente $\mathbf{M}\delta$

La tabla 6.12 muestra el número de operaciones algebraicas necesarias para la evaluación numérica la matriz de masa $\mathbf{M}$ y el vector δ exportados conjuntamente cuando han sido generados según se propone en **Caso C.2.2.1** y **Caso C.2.2.2**. Se puede ver que en el **Caso C.2.2.2** el número de operaciones es superior a cuando se calcula según el **Caso C.2.2.1**. Por ejemplo, en el **Caso C.2.2.1** en el modelo **Stewart** el número de operaciones es aproximadamente

un 8 % inferior al necesario en el **Caso C.2.2.2**. Para el modelo **Locomotora** esta diferencia se hace todavía mayor, siendo el número de operaciones en el **Caso C.2.2.1** aproximadamente un 40 % inferior al necesario en el **Caso C.2.2.2**.

6.6.0.2. Simetría

Utilizando el método del **Caso C.2.2.2**, además de generar elementos que requieres más operaciones algebraicas para su evaluación, aparece el problema de que la matriz de masa atomizada no es simétrica[1].

El siguiente listado de código muestra la matriz de masa del modelo **Cuatro Barras** obtenida según el **Caso C.2.2.2**.

```
M = [atom320 , atom341 , atom351;
     atom355 , atom359 , atom363;
     atom292 , atom323 , -I3yy]
```

Listado 6.1: Matriz de masa no simétrica. Cuatro barras

El siguiente listado de código muestra la matriz de masa del modelo **Cuatro Barras** obtenida según el **Caso C.2.2.1**.

```
M = [atom229 , atom230 , atom231;
     atom230 , atom232 , atom233;
     atom231 , atom233 , -I3yy]
```

Listado 6.2: Matriz de masa simétrica. Cuatro barras

Se puede observar que la primera matriz no es simétrica en átomos, en cambio la segunda sí.

[1] Si se obtuvieran las expresiones desatomizadas, sí que se vería la simetría.

6.6.1. Coordenadas sin inercia asociada. M y δ.

Los modelos **Biela-Manivela-Disco** y **Locomotora** son los únicos que presentan lo que en esta tesis se ha denominado coordenadas sin inercia asociada (Sección 2.8). Esto es debido a que ambos ejemplos presentan un problema de contacto.

En el modelo **Bilela-manivela** hay dos coordenadas de esta naturaleza que se usan para definir el punto de contacto entre el disco y el plano inclinado. En el modelo **Locomotora** para definir el punto de contacto entre cada rueda y el raíl hacen falta cuatro coordenadas. Ya que el modelo dispone de ocho ruedas, han sido necesarias 32 coordenadas sin inercia asociada.

La técnica propuesta en 4.5.4 permite el cálculo de el cálculo de **M** y δ de forma más rápida teniendo en consideración la naturaleza de estas coordenadas.

Se ha realizado el cálculo de **M** y δ en los dos casos siguientes:

C.2.2.3 Cuando **no** se tiene en cuenta la existencia de coordenadas sin inercia asociada.

C.2.2.4 Cuando **sí** se tiene en cuenta la existencia de coordenadas sin inercia asociada.

Modelo	C.2.2.3	C.2.2.4
Biela- manivela	0.030	0.012
Locomotora	132.213	79.703

Tabla 6.13.: Tiempo en segundos de cálculo de **M** y δ

En la tabla 6.13 muestra el tiempo necesario para obtener la matriz **M** y el vector δ en los dos citados casos. Se observa que el tiempo de cálculo de **M** y δ es más rápido cuando se da a este tipo de coordenadas un trato especial. En el modelo **Locomotora** se reduce casi un 35 %, hecho que justifica, por sí solo, la definición de coordenadas de esta naturaleza.

Se ha de remarcar que en ambos casos las matrices tienen el mismo número de átomos y es necesario el mismo número de operaciones para su evaluación numérica.

6.7. Expresiones susceptibles de ser simplificables trigonométricamente

En la sección 5.10 se realiza un análisis de cómo se trabaja con vectores y tensores extendidos para el cálculo de la matriz **M** y vector δ del mecanismo de cuatro barras del ejemplo A.2.

En esta sección se compara el tamaño del código generado en dos casos:

Con ESST El código generado contiene ESST.

Sin ESST El código generado no contiene ESST. Esto se realiza mediante el uso de vectores y tensores extendidos.

En ambos casos, se han utilizado las técnicas de atomización "sobre la marcha", así como los algoritmos de sustitución y de derivación recursivos propuestos en esta tesis. De igual manera, en ambos casos, el cálculo de la matriz **M** y del vector δ se ha realizado según las ecuaciones 4.54 y 4.62. Los torsores debidos a las fuerzas constitutivas no se han tenido en cuenta, únicamente los debidos a la fuerza de la gravedad. En el caso **Con ESST** todos los cálculos se han realiza con el modificador "Gravedad UP".

La tabla 6.14 muestra al número de átomos y de operaciones algebraicas en el código generado para la matriz **M** y el vector δ en los dos citados casos.

	M		δ	
	Átomos	**Operaciones**	**Átomos**	**Operaciones**
Con ESST	13	63	51	212
sin ESST	9	39	16	97

Tabla 6.14.: Código generado para el mecanismo de cuatro barras

En la tabla 6.14 se puede ver que el número de operaciones se reduce aproximadamente un 40 % para **M** y un 55 % para δ cuando se evita la aparición de ESST mediante el uso de vectores y tensores extendidos.

La Tabla 6.15 muestra el número de átomos y de operaciones algebraicas del código correspondiente a la matriz **M** y del vector δ cuando se exportan de forma conjunta. Se presenta para tres casos:

Con ESST El código generado contiene ESST.

Sin ESST El código generado no contiene ESST. Esto se realiza mediante el uso de vectores y tensores extendidos.

Robotran Código generado mediante el software Robotran.

En el tercer caso, con Robotran, se ha delegado en este programa la generación del código.

	Átomos	Operaciones
Con ESST	60	264
Sin ESST	25	135
Robotran	35	182

Tabla 6.15.: Comparación del tamaño del código de $\mathbf{M}\boldsymbol{\delta}$ (exportados conjuntamente)

Se puede observar que la combinación de las técnicas de atomización "sobre la marcha", derivación y sustitución recursivas y las técnicas de trabajo con vectores y tensores extendido producen un código un código más optimizado que el producido por Robotran. Se ha de hacer notar, que Robotran trabaja con una formulación recursiva y en esta tesis los cálculos se realizan basados en el PPV.

En al anexo B.2 se pueden ver los códigos generados para los tres casos.

CAPÍTULO 7

CONCLUSIONES

En la sección 1.3 se citan los distintos objetivos específicos de esta tesis. Estos están dirigidos a proponer una serie de algoritmos y métodos para mejorar y ampliar la aplicabilidad de las técnicas del modelado simbólico de sistemas multicuerpo.

Los distintos experimentos descritos en el capítulo 6 permiten afirmar que los objetivos han sido alcanzados satisfactoriamente.

7.1. Parentesización y reciclabilidad de átomos

La combinación de las técnicas de atomización "sobre la marcha" junto con los operadores cinemáticos (que aprovechan la recursividad subyacente en la estructuras de puntos y bases de forma independiente) y el modificador "gravedad UP" asegura que las expresiones simbólicas generadas estén óptimamente parentesizadas y que la reciclabilidad de átomos sea máxima. Esta combinación permite un modo de trabajo que es independiente de la formulación y de la parametrización propuestas, permitiendo superar algunas de las

limitaciones existentes en las DSM simbólica.

7.1.1. Atomización

Si bien ya se conocía que las técnicas de atomización son esenciales para la generación óptima de código simbólico, esta puede verse limitada por el coste computacional. Por ejemplo, en la tabla 6.1 puede verse que para un sistema con un gran número de grados de libertad (Locomotora) no es posible obtener el modelo cuando el proceso de atomización no está suficientemente optimizada. Las técnicas de atomización "sobre la marcha" descritas en [2] (Sección 5.5) sí que permiten obtener el modelo de sistemas de este tamaño, pero únicamente cuando se utilizan formulaciones de tipo recursivo.

En esta tesis se ha visto que el uso de tablas "hash" disminuye el tiempo necesario para obtener el modelo del sistema. Por ejemplo, en la misma tabla 6.1 se ve que para el sistema de la locomotora sí que es posible obtener el modelo, siempre y cuando se use la atomización basada en tablas "hash".

En la tabla 6.8 se ver el tiempo de necesario para la evaluación numérica de $\mathbf{M}$ y de δ. Se observa en los sistemas en que se han usado técnicas de atomización, el tiempo de evaluación numérica es notablemente inferior. Se puede observar que incluso hay un modelo en que el software numérico no puede llevar a cabo la evaluación, debido al tamaño de las funciones generadas. Se concluye que para poder simular los sistemas en tiempo real, una óptima atomización es esencial ya que la evaluación numérica puede llegar a ser el "cuello de botella" de las simulación.

7.1.2. Operadores cinemáticos

Separar el sistema multicuerpo en un árbol de bases y en un árbol de puntos ha hecho que puedan definir un conjunto de operadores cinemáticos (Sección 4.3) que saquen provecho de la topología de estos árboles. Estos operadores han sido definidos de tal manera que las expresiones obtenidas estén parentesizadas de forma óptima. En la sección 6.4 puede verse la influencia del operador "Vector de Posición" y de como su uso maximiza la parentesización.

Es importante hacer notar que el diseñador, al modelar el sistema, única-

mente se ha de preocupar en plantear el formalismo dinámico que más cómodo le sea, delegando en los operadores cinemáticos, de forma transparente, los cálculos pesados de la DSM.

7.1.3. Gravedad

En la sección 4.2 se ha justificado que el modificador "gravedad UP" asegura, en combinación con las técnicas de atomización, una máxima reciclabilidad de átomos en el sistema a modelar.

Este modificador, por un lado, asegura que las operaciones simbólicas se realicen siempre en el mismo orden, lo que genera idénticos átomos. Por otro lado, asegura que el orden en el que los operadores cinemáticos recorren el correspondiente árbol sea siempre desde la raíz del árbol hacia las hojas. Esto hace que elementos que comunes a dos ramas sean atomizados, y por lo tanto parentesizados, de idéntica forma. En la sección 6.5 se ha visto la influencia de este modificador en cuanto a la reciclabilidad de átomos.

Se ha de destacar que como se ha visto en el apartado 4.5.3.1, el modificador "gravedad UP" contribuye a que las operaciones que involucran al tensor de inercia sean realizadas en la base óptima. El único caso (Sub-sección 4.6.2.4) en que se ha de utilizar el modificador "gravedad DOWN" es en el cálculo de la contribución de las fuerzas y momentos exteriores (gravedad inclusive) al vector δ, si se quiere que el tamaño de las expresiones sea óptimo.

7.2. Algoritmos de sustitución y derivación recursivos

Dentro del contexto de la generación simbólica de sistemas, en las secciones 3.5 y 3.6 se proponen un algoritmo de sustitución de símbolos en expresiones y otro de derivación de expresiones respecto a un símbolo. Ambos algoritmos están diseñados para trabajar con sistemas simbólicos multicuerpo independientemente del tipo de formulación y/o parametrización utilizadas.

Los dos algoritmos han sido optimizados de tal manera que la desatomización ya no es necesaria cuando se está realizando la operación deseada. Los resultados presentados la tabla 6.2 la influencia en el coste computacional de

estos algoritmos.

En la tabla 6.1 se ve que, en general, el número de operaciones necesarias para evaluar los elementos generados por medio de estos algoritmos es menor que cuando no se han usado. Aun así, se ha de hacer notar, que en sistemas de pequeño tamaño, cuando las expresiones simbólicas son sencillas, los algoritmos no recursivos pueden llegar funcionar mejor. Esto es debido a que el motor simbólico es capaz de realizar optimizaciones y simplificaciones que los algoritmos presentados en esta tesis no realizan.

7.3. Generación de código

La atomización no solo juega un papel importante en la agilización de los algoritmos simbólicos, sino que también en la calidad del código generado y, por lo tanto, del coste computacional asociado a su evaluación numérica.

En la generación y exportación de código simbólico existen dos posibles estrategias. La primera es generar las expresiones simbólicas en extenso y, a continuación, utilizar un paquete simbólico para atomizarlas. La segunda estrategia es ir atomizando las expresiones "sobre la marcha".

En la tabla 6.3 se ve se ve la gran diferencia de tiempos entre las dos estrategias. El tiempo de generación de código mediante la atomización "a posteriori" (mediante Maple) es aproximadamente un orden de magnitud superior al de atomización "sobre la marcha".

En las tablas 6.5 y 6.4 se muestra el número de átomos y de operaciones necesarias para evaluar numéricamente los códigos generados mediante las dos estrategias. De nuevo, la primera estrategia, en general, genera un código "peor" (su evaluación requiere más operaciones algebraicas) que el generado mediante la estrategia propuesta en esta tesis.

La atomización "a posteriori", en general, no es la estrategia adecuada ya que no es capaz sacar partido de la recursividad subyacente en los sistemas multicuerpo (problema computacionalmente complejo). En cambio, la combinación de las técnicas de atomización "sobre la marcha" junto con los operadores cinemáticos saca ventaja de la recursividad subyacente en las estructuras de puntos y bases. Se ha de hacer notar que en sistemas sencillos el motor

simbólico Maple, en casos particulares, es capaz de generar código algo más optimizado que el generado con la atomización sobre la marcha.

7.3.1. Exportación conjunta

Para la resolución del problema dinámico, ambos elementos, la matriz $\mathbf{M}$ y el vector δ han de ser evaluados numéricamente. Además, en la sección 4.8 ha quedado demostrado que $\mathbf{M}$ y δ comparten una serie de elementos, por ejemplo, $\mathbf{v}_{RI}^{B_i}$ y $\boldsymbol{\omega}_{RI}^{S_i}$. Estos hechos justifican la necesidad de exportarlos conjuntamente, tal y como se propone en 3.7.5.

En la tabla 6.6 se observa que un aproximadamente 7 % de las operaciones algebraicas necesarias para evaluar ambos elementos es común. Lo que directamente se traduce en una reducción en el tiempo de evaluación numérica. Como ya ha se ha comentado, la evaluación numérica puede ser uno de los "cuellos de botella" de la simulación en tiempo real. Así pues este ahorro producido por el hecho de exportar estos elementos de forma conjunta justifica el hacerlo así.

Se ha de hacer notar que se puede proceder de forma análoga con la matriz $\boldsymbol{\Phi}_q$ y el vector $\boldsymbol{\Phi}$.

7.4. Cálculo de $\mathbf{M}$ y δ

Los dos métodos, basados en el PPV, presentados en esta tesis permiten obtener directamente la matriz $\mathbf{M}$ (Sección 4.5) y el vector δ (Sección 4.6), necesarios para la simulación dinámica, atomizados y con una parentesización óptima. Para ello, estos métodos utilizan los operadores cinemáticos, las técnicas de atomización y el algoritmo de derivación presentados en esta misma tesis. La combinación de los dos métodos en un único bucle (sección 4.8) maximiza la reciblabilidad de los átomos, mejora el tiempo de computación y hace que la expresiones simbólicas obtenidas estén maximizadas de forma óptima.

Como se puede ver en la tabla 6.11, el tiempo de cálculo de $\mathbf{M}$ y δ crece exponencialmente con el tamaño del problema. Cuando se obtienen $\mathbf{M}$ y δ por medio de la metodología presentada en esta tesis el tiempo de cálculo

es mucho menor (Ver el modelo **Locomotora** en la tabla 6.11) que cuando se obtienen mediante al PPV tal y como está definido en 2.3.1.4.

De igual manera, en la tabla 6.12 se ve que el número de operaciones algebraicas necesarias para la evaluación numérica $\mathbf{M}$ y δ exportados conjuntamente es inferior en el caso en que se utilicen los métodos para el cálculo de estos elementos definidos en esta tesis. Esto es debido a la óptima parentesización que estos métodos producen.

Así mismo, en la tabla 6.13 se puede ver que la definición de "coordenadas sin inercia asociada" hace que el calculo de $\mathbf{M}$ y δ se haga todavía de una manera más rápida. Para el modelo **Locomotora** llega a hacerse hasta un 40 % más rápido. Esto es gracias a que las matrices jacobianas necesarias se han de realizar únicamente respecto a un subconjunto de coordenadas y velocidades y no respecto al total. Este hecho justifica por sí solo la definición de este tipo de coordenadas.

Por último, se ha destacar que, si bien en [2] (Sección 5.7.3) se indica que el PPV es inapropiado para sistema de mediano y gran tamaño, en esta tesis se ha logrado definir una metodología que, en combinación con otras técnicas, permite trabajar con este principio en sistemas multicuerpo de gran tamaño, generando resultados igual de óptimos que las formulaciones recursivas.

7.5. Expresiones susceptibles de simplificación

El capítulo 5 se ha centrado en el estudio de las ESST y se ha propuesto un método de trabajo mediante vectores y tensores extendidos que evita su aparición. Evitar la aparición de ESST se traduce en la generación de un código más optimizado y por lo tanto que se evalúa numéricamente más rápido.

En la sección 5.8 se analiza el uso de estos elementos extendidos en el campo de la DSM y los problemas que pueden surgir. Así mismo, se propone un conjunto de soluciones enfocadas a superar estos problemas. El conjunto formado por los vectores y tensores extendido, las técnicas de atomización y los operadores cinemáticos permite generar código altamente optimizado.

Se ha destacar, que las técnicas de trabajo con vectores y tensores extendidos son aplicables independientemente de la formulación y parametrización.

7.6. Otros

Por último, se quiere destacar que todas la técnicas aquí presentadas pueden ser implementadas en cualquier lenguaje de programación que tenga la posibilidad de realizar cálculos simbólicos y en paquetes comerciales para el cálculo simbólico.

Algunas de las metodologías propuestas son independientes del tipo de sistema simbólico a estudio, es decir, no están enfocadas únicamente al contexto de la DSM. Por ejemplo, las técnicas de atomización y los algoritmos recursivos de derivación, de sustitución y de generación de código exportable, son perfectamente aplicables en otros contextos en los que el cálculo simbólico sea una herramienta.

CAPÍTULO 8

LÍNEAS FUTURAS

En este capítulo se presentan los diferentes aspectos teóricos relacionados con la modelización simbólica de sistemas multicuerpo que se pretender abordar en un futuro próximo.

8.1. Algoritmos de sustitución y diferenciación

Se ha visto en el capítulo 6 que en ciertos casos particulares los algoritmos recursivos de sustitución y diferenciación presentados en esta tesis no presentaban resultados tan óptimos como las versiones anteriores de los mismos.

Esto es debido a dos causas principales. En primer lugar, es debido al hecho que el vector $\dot{\mathbf{\Phi}}$ se ha obtenido mediante derivación a partir del vector $\mathbf{\Phi}$ y este cálculo no es óptimo. En segundo lugar, es debido a que para sistemas pequeños, los algoritmos no recursivos funcionan mejor, debido a que las operaciones se delegan al motor simbólico.

Por este motivo, se propone estudiar los mismos casos cuando el vector $\dot{\mathbf{\Phi}}$ es planteado directamente mediante ecuaciones cinemáticas de velocidad y

comparar los resultados con los obtenidos previamente.

8.2. Cálculo de la matriz de observación

Las ecuaciones dinámicas de un modelo multicuerpo, evitando utilizar modelos de fricción no lineales en los parámetros, son lineales en los parámetros dinámicos:

$$\mathbf{K}(q, \dot{q}, \ddot{q})\boldsymbol{\phi} = \boldsymbol{\tau} \tag{8.1}$$

donde $\mathbf{K}$ se denomina "matriz de observación para un instante" y contiene la información del movimiento del sistema, $(q, \dot{q}, \ddot{q})$ representan las coordenadas, velocidades y aceleraciones generalizadas y $\boldsymbol{\tau}$ representa el vector de fuerzas y momentos externos que actúan sobre el sistema. El vector $\boldsymbol{\phi}$ representa los parámetros dinámicos del sistema.

Al ser las ecuaciones dinámicas, $\mathrm{e}(q, \dot{q}, \ddot{q}, \boldsymbol{\phi}, \boldsymbol{\tau})$, lineales en los parámetros, la matriz de observación se obtiene como:

$$\frac{\partial \mathrm{e}(q, \dot{q}, \ddot{q}, \boldsymbol{\phi}, \boldsymbol{\tau})}{\partial \boldsymbol{\phi}^{\mathsf{T}}} = \mathbf{K}(q, \dot{q}, \ddot{q}) \tag{8.2}$$

El elevado conste computacional del cálculo simbólico de esta matriz hace que sea uno de los cuellos de botella del modelado simbólico.

Esta matriz puede ser obtenida mediante un procedimiento análogo al descrito en 4.5, es decir, puede ser obtenida sin necesidad de calcular las ecuaciones dinámicas por medio del PPV . Además, el cálculo de esta matriz puede ser realizado en el mismo bucle de cálculo (Sección: 4.8) en el que se calculan $\mathbf{M}$ y δ, lo que permite "reciclar" una gran cantidad de los cálculos ya realizados.

Se plantea analizar cómo se realiza dicho cálculo y cómo extender el bucle de cálculo de $\mathbf{M}$ y δ para que se realice a la vez el cálculo de la matriz de observación.

8.3. Transferencias de inercia

Puede que un modelo dinámico multicuerpo no dependa de todos y cada uno de los parámetros inerciales que lo constituyen, ya que puede ser independiente de algunos parámetros o depender de combinaciones lineales de otros. Debido a esto, es posible reducir el modelo para escribirlo en términos de los denominados "parámetros base", en función de los cuales la cinética queda unívocamente determinada.

Estos parámetros base pueden ser obtenidos utilizando tanto métodos numéricos como métodos simbólicos. En la referencia [44], se propone un método que permite obtener simbólicamente estos parámetros base de forma automática basado en el concepto denominado "transferencias de inercia" [45].

Así pues, el hecho de que toda la información dinámica dependa únicamente de este conjunto de parámetros base puede ser aprovechado a la hora de reducir el tamaño del modelo [46] y por lo tanto el coste computacional del modelado simbólico del mismo.

Se plantea estudiar cómo integrar este método automático de obtención de los parámetros base en las técnicas aquí presentadas con la intención de definir una metodología que permita reducir y modelar el sistema de forma óptima y automática. Esta metodología ha de ser válida incluso para sistemas de cadena cerrada, ya que el método presentado en [44] así lo permite.

8.4. Vectores de aceleración

De igual manera que existen dos operadores para el cálculo de la velocidad de un punto con respecto a una referencia y de la velocidad angular de una base con respecto a otra, es necesario definir otros dos operadores para el cálculo de las aceleraciones, de un punto y de una base.

Se plantea implementar el operador para el cálculo de la aceleración de un punto respecto a una referencia que aproveche la estructura de puntos y que realice este cálculo por medio de la ecuación de composición de aceleraciones (Ecuación 2.10). El vector resultado ha de tener las componentes proyectadas de forma óptima en la base que determine el modificador "gravedad".

De forma análoga, se plantea definir otro operador que, aprovechando la estructura de bases, calcule la aceleración angular de una base respecto a otra por medio de la ecuación 2.7. El vector resultado ha de tener las componentes proyectadas de forma óptima en la base que determine el modificador "gravedad".

8.5. Modelado de sistemas flexibles

Esta tesis se ha centrado únicamente en la modelización simbólica de sistemas multicuerpo rígidos. Las técnicas de atomización, derivación y sustitución son perfectamente válidas en el contexto de la dinámica flexible. De igual manera, pueden aparecer ESST que si se tratan como en esta tesis se propone pueden ser eliminadas sin mayor esfuerzo computacional.

Así pues, se plantea realizar un análisis de la aplicación de las técnicas presentadas en esta tesis en el contexto de la dinámica de sistemas multicuerpo flexibles.

8.6. Exportación conjunta de todo el sistema

En 3.7.5 se ha propuesto un método para exportar la matriz de masa y el vector de fuerzas generalizadas deforma conjunta. Se propone proceder de forma análoga con el vector de ecuaciones geométricas y la matriz jacobiana para evitar la re-evaluación de expresiones que puedan ser comunes en ambos elementos.

Además, se propone analizar si el planteamiento de las ecuaciones cinemáticas basado en operadores (en vez de en la derivación de las ecuaciones geométricas) permite un mayor reciclado de átomos.

Se analizarán los algoritmos numéricos de simulación para ver si es posible optimizar más la evaluación numérica. Se ha observado que pueden aparecer expresiones que son constantes a lo largo de toda la simulación, o que son constantes a lo largo de cada paso de integración. Se propone analizar cuándo ocurre este efecto y buscar un método de exportación que lo tenga en cuenta.

De igual manera, se propone modificar los algoritmos numéricos de integración para que también lo tengan en cuenta.

8.7. Estructura de la matriz de masa. Descomposición LDL

Se ha visto que en la matriz de masa está implícita la topología del sistema multicuerpo. Por ejemplo, la figura 8.1 muestra la topología de la matriz de masa del ejemplo A.5, donde cada uno de los puntos indica un elemento no nulo. En la figura se puede ver la alta "esparsidad" que presenta la matriz donde un 83.57 % de los elementos son cero.

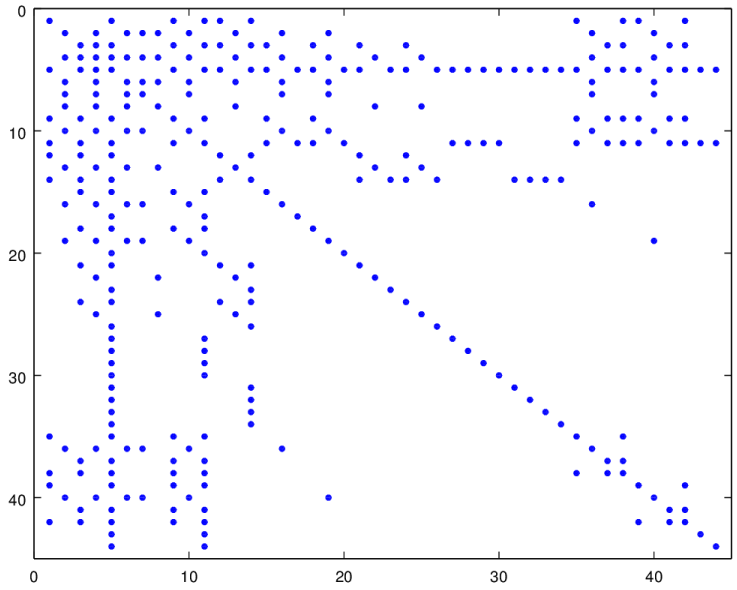


Figura 8.1.: Matriz de de masa del ejemplo A.5

Se plantea la implementación de una descomposición LDL^{T} simbólica que, aprovechando la gran "esparsidad" asociada a la topología del sistema, per-

mita la realización de dicho cálculo de manera más rápida.

APÉNDICE A

EJEMPLOS

En este anexo se describen los distintos ejemplos presentes en la tesis. Se muestra un esquema de los mecanismo, su acotación, definición de coordenadas, puntos y bases.

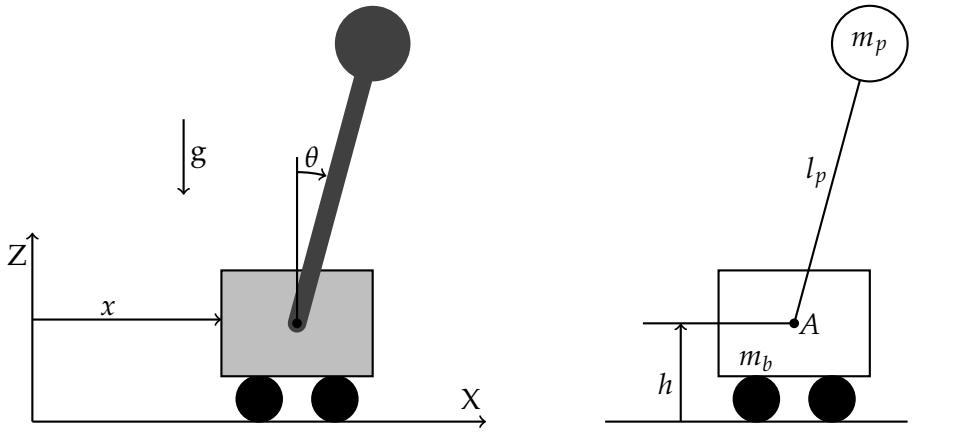
Al final de cada ejemplo se añade una nota en la que se refleja en número de coordenadas, el número de grados de libertad y el número de ecuaciones geométricas y de ecuaciones no-holónomas.

A.1. Bloque-peńdulo

El mecanismo A.1 está formado por un péndulo unido a un sólido (denominado "bloque") que se desplaza a lo largo de un eje.

El péndulo tiene una masa m_p y una longitud l_p; el bloque (que se ha dibujado con dos ruedas aunque en realidad éstas no están en el mecanismo) tiene una masa m_b y una altura h hasta el punto de unión con el péndulo, el punto A.

Además hay una fuerza exterior F_b actuante sobre el bloque y un momento



(a) Péndulo invertido **(b)** Parámetros del mecanismo

Figura A.1.: Ejemplo 1. Péndulo invertido

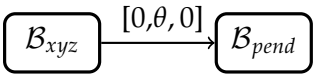


Figura A.2.: Diagrama de bases

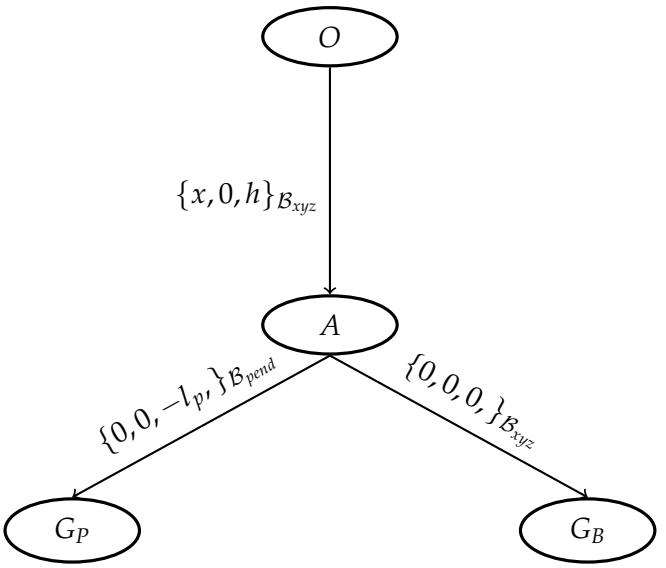


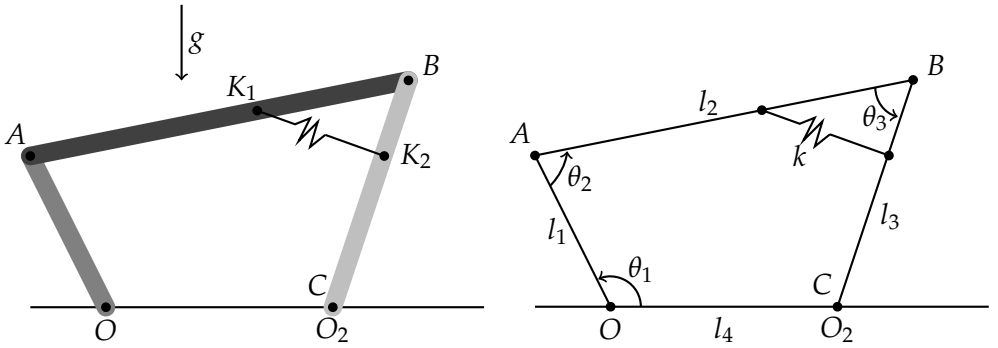
Figura A.3.: Diagrama de puntos

exterior M_p actuante sobre el péndulo. Hay rozamiento viscoso c_{vis} entre el bloque y el suelo y además existe un muelle torsional de constante k entre el péndulo y el carro. El mecanismo está bajo la acción de la gravedad g.

Se trata de un mecanismo con dos coordenadas, dos grados de libertad y sin ecuaciones geométricas.

A.2. Cuadrilátero articulado

El mecanismo es análogo al que se presenta en [2] (Capítulo 5). Se trata de un cuadrilátero articulado, o mecanismo de cuatro barras, que tiene entre dos de sus eslabones un muelle.



(a) Cuadrilátero articulado **(b)** Parámetros del mecanismo

Figura A.4.: Ejemplo 2. Cuadrilátero articulado

El mecanismo ha sido acotado mediante tres coordenadas relativas: θ_1, θ_2 y θ_3. Cada una de las barras tiene una base asociada definida a partir de la base anterior:

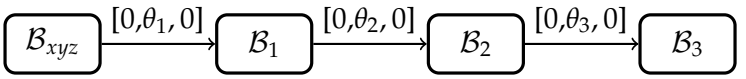


Figura A.5.: Diagrama de bases

Los puntos O, A y B ha sido usado para definir las referencias de las barras uno, dos y tres respectivamente.

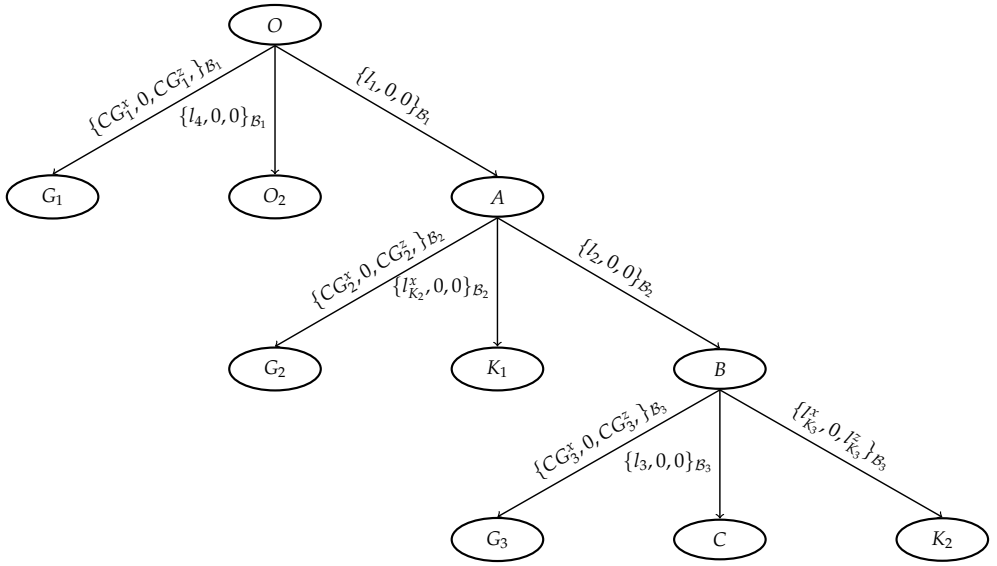


Figura A.6.: Diagrama de puntos

El punto C pertenece a la barra tres, el punto O_2 pertenece al sólido suelo, aunque en los esquemas aparezcan en el mismo lugar geométrico, son dos puntos diferentes.

En la referencia de cada barra se ha definido la posición del centro de masas y su tensor de inercia. La posición del centro de masa se da , por ejemplo, para el sólido dos, mediante el vector:

$$\mathbf{r}_A^{G_2} = \begin{Bmatrix} CG_2^x \\ CG_2^y \\ CG_2^z \end{Bmatrix}_{\mathcal{B}_2} \tag{A.1}$$

Además, en la referencia de la barra número dos se ha definido el punto K_1 y en la referencia de la barra tres el punto K_2, estos son los puntos de unión del muelle. Este muelle tiene una constante elástica de valor k.

Las barras dos y tres tienen cada una un momento exterior y una fuerza exterior que actúan sobre ellas. Estas parejas de fuerza y momentos están aplicadas sobre los puntos A y B respectivamente . Además, el mecanismo está bajo el efecto de al gravedad.

La ecuación vectorial geométrica de cierre del mecanismo es:

$$\mathbf{r}_O^A + \mathbf{r}_A^B + \mathbf{r}_B^C = \mathbf{r}_O^{O_2} \tag{A.2}$$

Se trata de un mecanismo con un grado de libertad, tres coordenadas y dos ecuaciones geométricas.

A.3. Biela, manivela, disco

La figura representa un esquema del mecanismo en su posición de funcionamiento. La manivela (*Arm*1) impulsa a la biela (*Arm*2), que a su vez impulsa el disco (*Disco*). Dicho disco está en contacto con el plano inclinado y rueda sin deslizar. Además, a la manivela se aplica un momento exterior y existe rozamiento viscoso entre la manivela y el suelo. El mecanismo está bajo la acción de la gravedad.

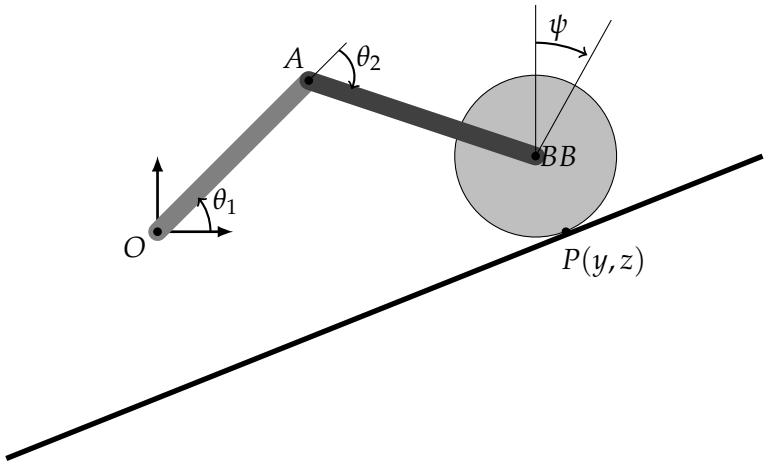


Figura A.7.: Mecanismo Biela, manivela, elipse

El mecanismo está acotado mediante 5 coordenadas:

θ_1 Es el ángulo de la manivela respecto de la referencia absoluta.

θ_2 Es el ángulo de la biela respecto de la manivela.

ψ Es el ángulo girado por el disco respecto de la referencia absoluta.

y Coordenada horizontal del punto auxiliar P definida en la referencia absoluta.

z Coordenada vertical del punto auxiliar P definida en la referencia absoluta.

De estas cinco coordenadas tres son necesarias para la definición de la dinámica del mecanismo y otras dos son necesarias para definir el punto de contacto.

El punto de contacto P está definido mediante tres ecuaciones geométricas. Además, al no haber deslizamiento existe una ecuación no-holónoma y hay una acotación mixta, se usan tanto coordenadas absolutas como relativas.

Hay cuatro bases definidas (Figura: A.8), una en cada brazo, otra en el disco y otra auxiliar para definir la inclinación del plano.

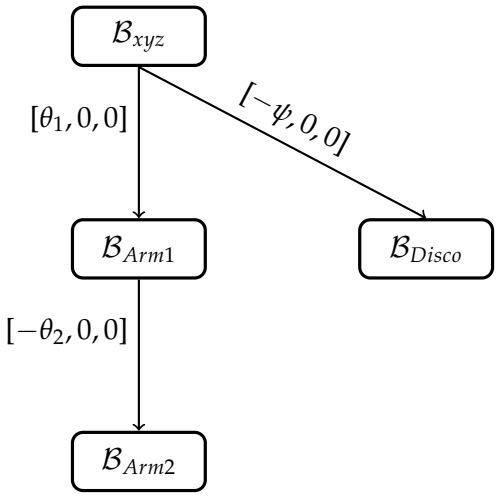


Figura A.8.: Diagrama de bases

La figura A.9 muestra el esquema de puntos de este mecanismo.

Cada una de las dos barras tiene definido su centro de gravedad (G_{Arm1} y G_{Arm2} respectivamente) en el centro de la propia barra. Las longitudes de las barras son l_1 y l_2 respectivamente. El disco tiene definido el centro de gravedad en el centro del mismo y su radio es r.

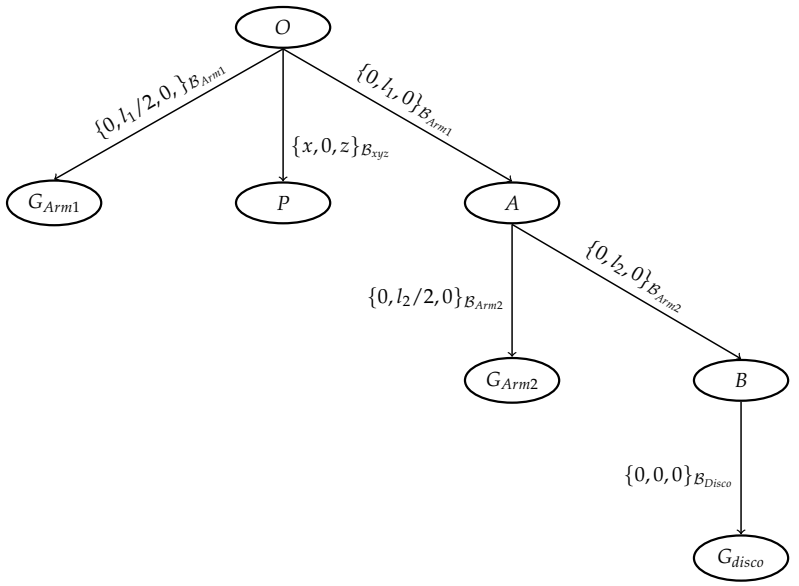


Figura A.9.: Diagrama de puntos

Por último se ha de definir el contacto entre el suelo y el disco y con este fin se han de definir tres ecuaciones geométricas. Además se ha de añadir una cuarta ecuación no-holónoma para definir la rodadura del disco. En total, hay cuatro ecuaciones de velocidad y por lo tanto cuatro multiplicadores de Lagrange.

Las condiciones geométricas se definen de la siguiente forma:

- El punto P está en la periferia del disco (ecuación de la circunferencia)
- El punto P está en el plano inclinado (ecuación del plano inclinado)
- El vector normal a la circunferencia es perpendicular al plano inclinado

La ecuación no-holónoma de la condición de no deslizamiento se plantea imponiendo la velocidad del punto P perteneciente al disco sea nula.

El mecanismo tienen 5 coordenadas, cuatro ecuaciones de restricción y un grado de libertad. Dos de las coordenadas, x y z no tienen una inercia asociada.

A.4. Manipulador paralelo "Stewart"

La figura A.10 muestra un manipulador paralelo de tipo Stewart. Esta formado por una plataforma unida mediante seis actuadores lineales a suelo. Estos actuadores están montados por parejas en tres puntos de la base del manipulador y se cruzan para unirse por parejas a la plataforma en otros tres puntos.

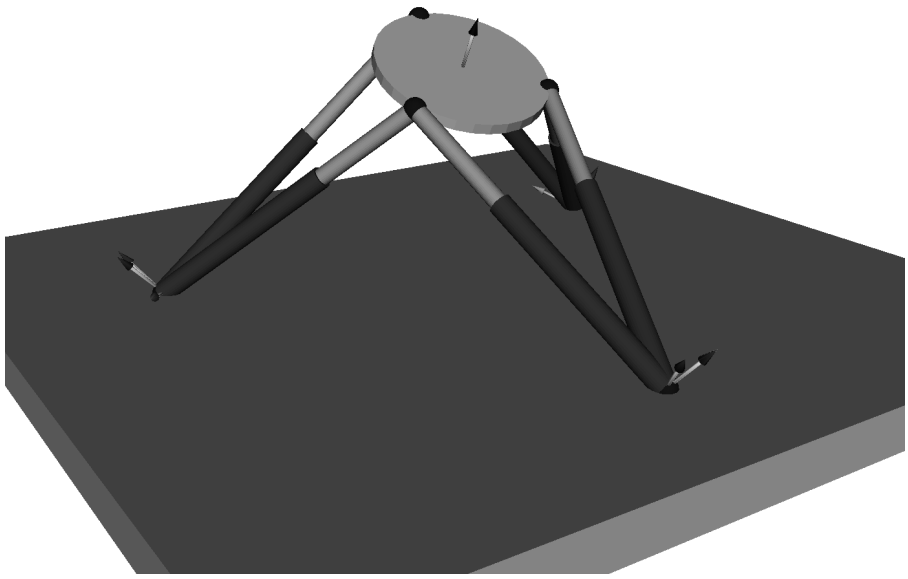


Figura A.10.: Modelo en CAD de la plataforma

La referencia de la plataforma está formada por el punto P y la base $\mathcal{B}_P$. Cada uno de los actuadores está formado por sólidos. El sólido con la referencia formada por la pareja Ax, $\mathcal{B}_x$ (x es el número de sólido) es el que está más cerca de las base. El sólido con la referencia formada por la pareja Cx, $\mathcal{B}_x$ es el que está más cerca de la plataforma.

La plataforma tiene seis grados de libertad (tres desplazamientos y tres giros) y cada uno de los actuadores tres (dos giros y su longitud). La unión de los actuados con la plataforma y el suelo se ha realizado mediante ecuaciones geométricas.

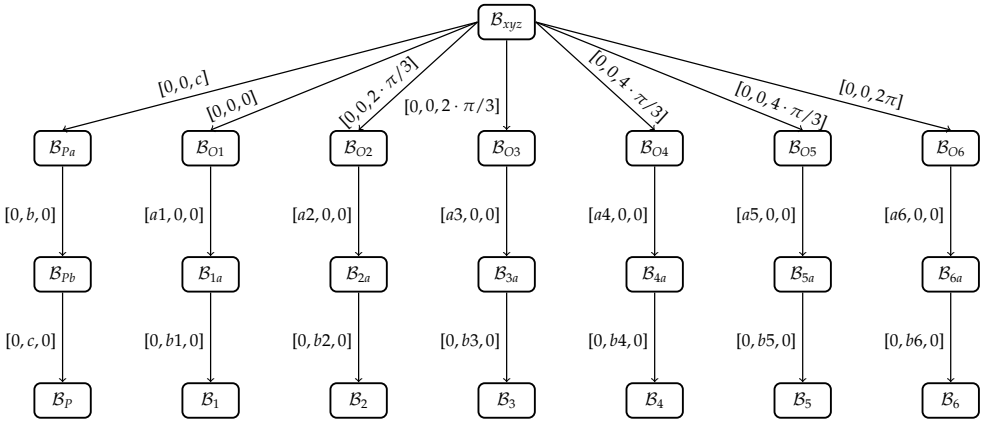


Figura A.11.: Diagrama de bases

En total el mecanismo tienen un total de 24 coordenadas, seis grados de libertad y 18 ecuaciones geométricas.

A.5. Locomotora FEVE 3800

En este ejemplo se ha desarrollado el modelo multicuerpo de una locomotora tipo FEVE 3800 con suficiente detalle como para capturar los efectos dinámicos que puedan afectar a la adherencia[1].

Para ello se ha realizado un modelo tridimensional multicuerpo de toda la locomotora, formada por un bogie tractor y otro no tractor, sobre los que se aplica el peso del resto de la locomotora. Se ha definido además un modelo de contacto detallado rueda raíl, que tiene en cuenta la geometría de los perfiles, con el modelo de fricción tipo Coulomb dependiente de la velocidad de deslizamiento comentado anteriormente.

[1]Uno de los problemas más importantes en el funcionamiento de una locomotora es el de la adherencia rueda raíl. Esto es debido a que la adherencia controla la tracción máxima que puede utilizarse efectivamente para propulsar la locomotora. Frecuentemente la adherencia se modela como una fuerza de rozamiento de tipo Coulomb –por tanto dependiente de la fuerza normal de contacto– cuyo coeficiente de rozamiento dinámico depende de la velocidad de deslizamiento. Debe notarse que este coeficiente de rozamiento disminuye apreciablemente cuando el contacto rueda raíl está mojado, u otros elementos –como hojas– se interponen en el contacto.

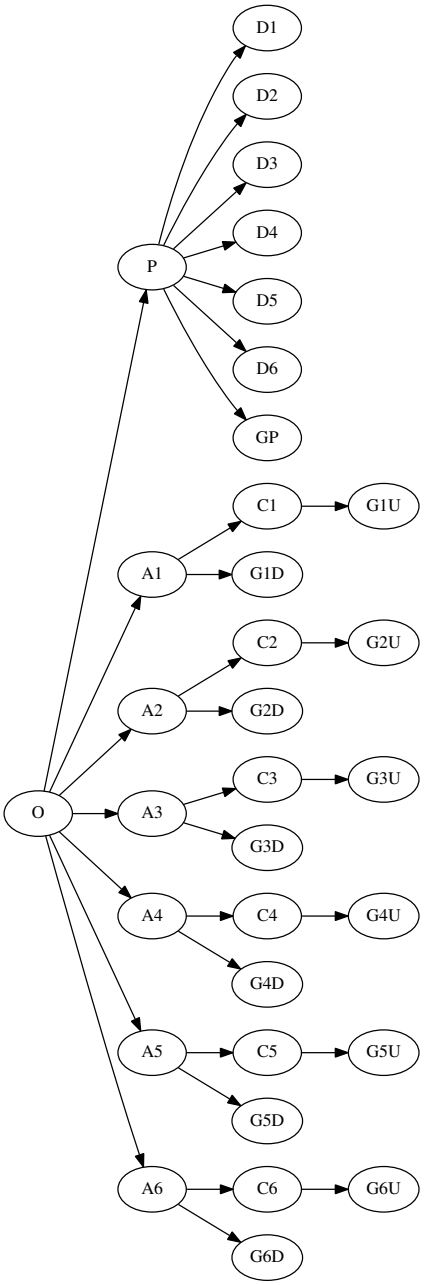


Figura A.12.: Diagrama de puntos

La figura A.13 representa la locomotora de forma esquemática. Como puede verse ésta está compuesta de un cuerpo principal, unido a los bogies delantero y trasero. Los bogies están compuestos por una doble suspensión: la primaria entre las ruedas y el chasis del bogie, y la secundaria entre este último y la biela bailadora o *Slider*. El cuerpo principal de la locomotora está soportado por las suspensiones secundarias de sendos bogies; para ello locomotora se une a los *Slider*s de cada bogie por medio de una junta de pasador vertical que es modelada como un par de revolución.

Figura A.13.: Modelo esquemático de la locomotora FEVE 3800

El cuerpo principal de la locomotora es el denominado WAG y desde él están definidos los dos bogies. Este cuerpo principal tienen seis grados de libertad (tres desplazamientos y tres giros). Los bogies delantero y trasero son iguales salvo por el hecho de que el delantero no presenta motorización.

La figura A.14 representa esquemáticamente el modelo del bogie trasero. Dos "conjuntos rueda" o *Wheelsets* (WHSR y WHSF) están unidos al chasis principal del bogie (BOG) a través de la suspensión primaria. La suspensión primaria para cada *Wheelset* se realiza mediante la colocación de la cajas de grasa cada una de las cuales es unida al chasis por dos grupos muelle amortiguador.

En la Figura A.14 se presenta un esquema de modelo de Bogie, determinando los sólidos, sus referencias, los puntos de interés.

Los distintos sólidos por los que está formado este modelo de bogie tractor y su nomenclatura se pueden ver en la siguiente lista:

- **Sólido BOG**: Bogie o bastidor principal.
- **Sólido MOTF**: Motor delantero.
- **Sólido MOTR**: Motor trasero.
- **Sólido ROTF**: Rotor del motor delantero.

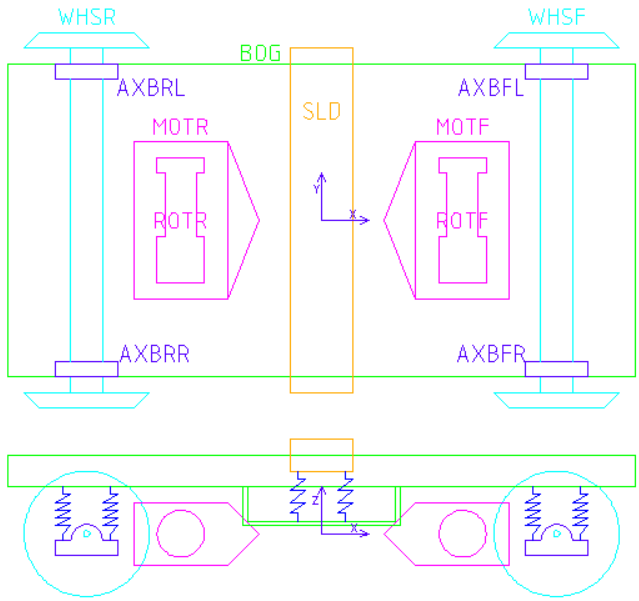


Figura A.14.: Modelo en CAD.

- **Sólido ROTR**: Rotor del motor trasero.
- **Sólido WHSF**: WheelSet (eje + ruedas) delantero.
- **Sólido WHSR**: WheelSet (eje + ruedas) trasero.
- **Sólido AXBFL**: AxleBox delantera izquierda.
- **Sólido AXBFR**: AxleBox delantera derecha.
- **Sólido AXBRL**: AxleBox trasera izquierda.
- **Sólido AXBRR**: AxleBox trasera derecha
- **SLD**: Slider. Traviesa bailadora

El bogie no tractor es análogo a este salvo por la diferencia, como ya se ha comentado, de la ausencia de motores. Así pues la nomenclatura es igual.

El modelo contempla las dos suspensiones de las que dispone el sistema real, una suspensión primaria y una secundara. La primaria se sitúa entre el conjunto WheelSet y el Bogie, la segunda entre el Bogie y el Slider (Car-

Body). En ambos casos la suspensión está formada por muelle y amortiguador. Además, el modelo dispone de un total de 2 SilentBlocks, que son las uniones entre los motores y el bogie. Cada uno permite 3 desplazamientos y 3 giros diferenciales.

Se ha creído conveniente enseñar los esquemas tanto de bases como de puntos de el bogie tractor para que el lector se haga una idea de la magnitud del problema. Es esquema de bases parte desde la base del cuerpo de la locomotora (B_WAG) la cual ha sido definida, como ya se ha comentado, mediante tres giros desde la referencia inercial. De igual manera, la definición del bogie tractor parte desde el punto O_WAG, punto central de la locomotora, el cual ha sido definido a través de tres coordenadas desde el punto origen.

Las bases están definidas por la letra B y el número o nombre del sólido al que hacen referencia. Por ejemplo, la base B_BOG hace referencia a la orientación del sólido BOG (bogie). El índice *abc* (ejemplo B_BOGa, B_BOGb, B_BOGc) significa que de una base a la siguiente se pasa mediante un giro elemental. Éstos están ordenados así: B,C y A. En las flechas que en el diagrama unen los diferentes círculos con los nombres de las bases se ha indicado cómo se pasa de una a la otra, así: Para pasar de la base *B_BOGa* a la *B_BOG* en el diagrama indicia *[0;1;0;bBOG]*. Esto quiere decir que de una base a la otra se pasa girando sobre el eje Y (segunda, donde aparece un 1) un ángulo bBOG.

La figura A.15 corresponde al diagrama de bases del modelo del bogie tractor. El esquema del bogie no tractor es análogo pero sin los tres giros correspondientes a cada motor.

Los puntos que empiezan con la letra "O" son orígenes de referencia. Y el número o nombre a continuación es el nombre del sólido al que es solidario a esa referencia. Lo puntos que empieza por "P" son puntos definidos en los distintos sólidos y que tienen como apellido el sólido al que pertenecen, es decir,"P" + "número o nombre correlativo de punto" + "solido". En la figura A.16 se representan los puntos únicamente para el bogie tractor. El bogie no tractor es análogo pero sin los puntos usados para definir los motores.

En cada una de las uniones entre los puntos se ha detallado el número coordenadas hay implicadas. Los puntos en los que no se indica serán puntos cuya posición con respecto al punto que se define es fija. El conjunto total de coordenadas del sistema dinámica será el número total de coordenadas para los

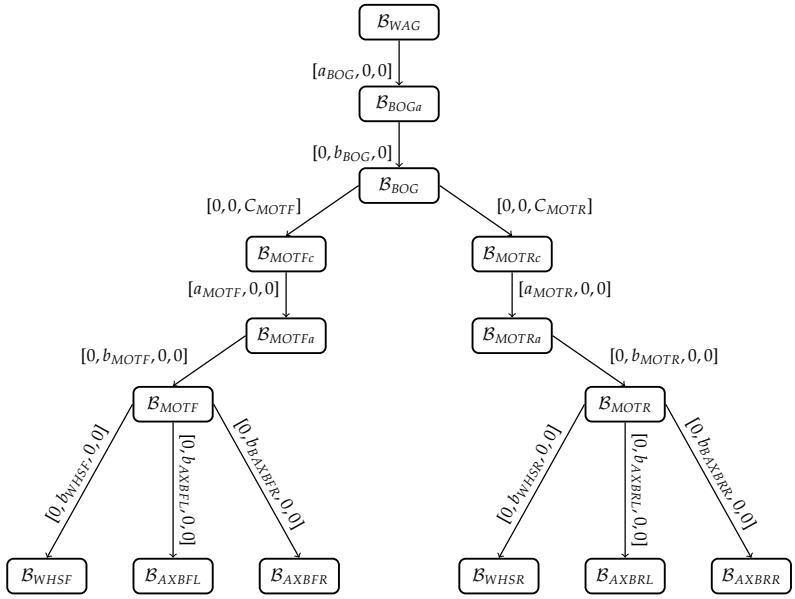


Figura A.15.: Diagrama de bases del bogie tractor

cambios de base, de la locomotora, del bogie tractor y del bogie no tractor, más el número total de coordenadas para definir los puntos de la locomotora, bogie tractor y bogie no tractor. El número total de coordenadas asociadas a las ecuaciones dinámicas es de 44.

A.5.1. Ecuaciones geométricas

La ecuaciones geométricas son necesarias para la definición del contacto entre la rueda y la vía teórico, se ha supuesto que tanto las ruedas como la vía tienen un perfil genérico definido mediante una curva "spline cúbica".

La parametrización del contacto real entre la vía y la rueda se ha de considerar dos superficies, cada una definida por dos parámetros. Para un sólido P su superficie queda definida por:

$$\mathbf{r}^P = \mathbf{f}(s_1, s_1) \tag{A.3}$$

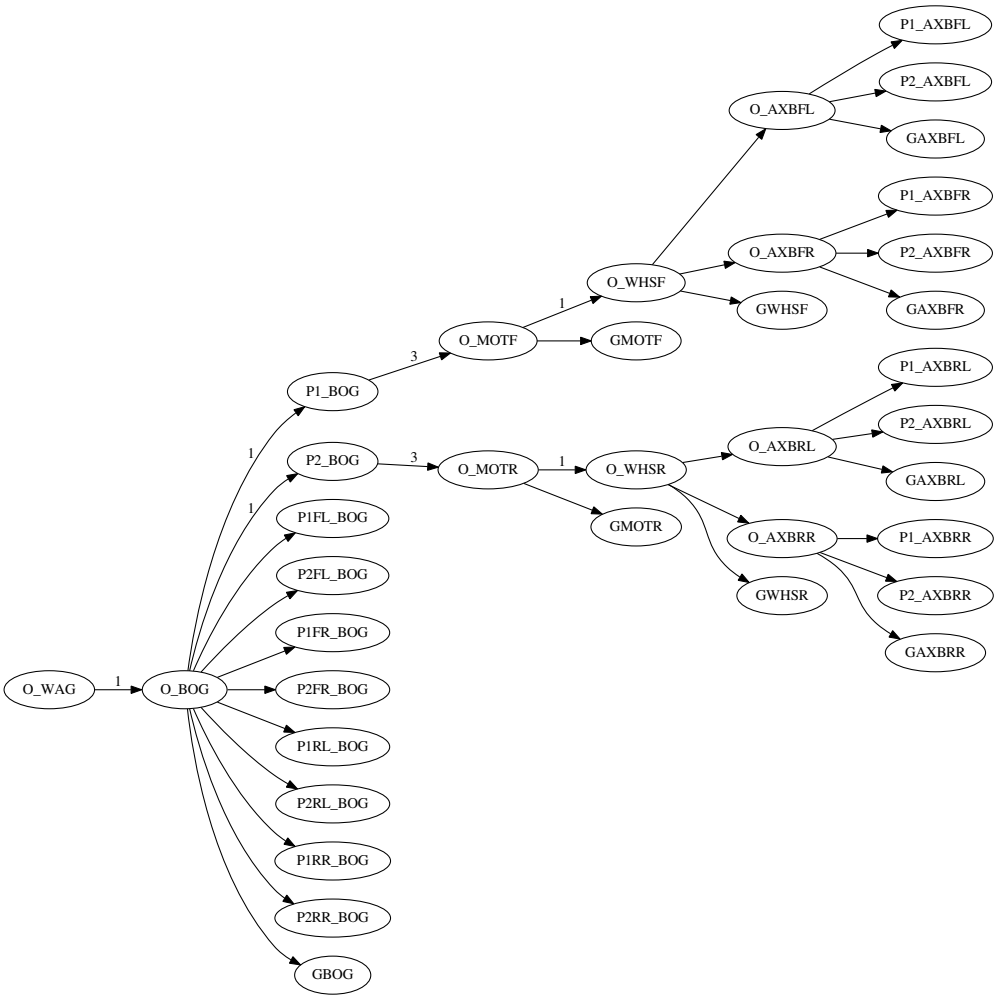


Figura A.16.: diagrama de puntos

Para la descripción matemática del contacto se necesitarán dos vectores tangentes a la superficie y uno normal:

$$\begin{aligned} \mathbf{t}_1^P &= \frac{\partial \mathbf{r}^P}{\partial s_1} \\ \mathbf{t}_2^P &= \frac{\partial \mathbf{r}^P}{\partial s_2} \\ \mathbf{n}^P &= \mathbf{t}_1^P \wedge \mathbf{t}_2^P \end{aligned} \tag{A.4}$$

El contacto entre dos superficies queda definido por las siguientes coordenadas asociadas a cada superficie:

$$\mathbf{q} = \begin{bmatrix} s_{1P}^i & s_{2P}^i & s_{1P}^j & s_{2P}^j \end{bmatrix}^T \tag{A.5}$$

y las ecuaciones que describen el contacto para las dos superficies serían:

$$\begin{aligned} \mathbf{r}_i^p - \mathbf{r}_j^p &= \mathbf{0} \\ \mathbf{n}_p^{j^T} \mathbf{t}_{lp}^j &= 0 \\ \mathbf{n}_p^{j^T} \mathbf{t}_{tp}^j &= 0 \end{aligned} \tag{A.6}$$

Es decir, para definir cada uno de los contactos entre una rueda y la vía se necesitan 4 coordenadas nuevas y 5 ecuaciones de restricción.

A.5.2. Coordenadas totales del modelo

Así pues, ya que el modelo tiene dos bogies y cada uno tiene 4 ruedas, para el contacto de las ruedas se han de definir un total de 32 coordenadas y 40 ecuaciones de restricción. Como para definir la dinámica del modelo han sido necesarias 44 coordenadas, el número total de coordenadas de este modelo asciende a un total de 76. El modelo tiene 36 grados de libertad.

Además, por cada ecuación dinámica, según la formulación aquí elegida, se ha de definir un multiplicador de Lagrange, es decir, un total de 40 multipli-

cadores.

Por último, se ha de hacer notar que las coordenadas utilizadas para definir el contacto no participan en las ecuaciones dinámicas, es decir, que no tienen inercia asociada.

APÉNDICE B

CÓDIGO SIMBÓLICOS GENERADO. MECANISMO DE CUATRO BARRAS

B.1. Componentes del vector $\mathbf{r}_O^C$ calculado según distintos métodos

```
-sin(theta1)*(sin(theta2)*(l3*cos(theta3)+l2)
+l3*sin(theta3)*cos(theta2))-cos(theta1)*
(l3*sin(theta2)*sin(theta3)-l1-cos(theta2)
*(l3*cos(theta3)+l2));
0;
-cos(theta1)*(sin(theta2)*(l3*cos(theta3)+l2)
+l3*sin(theta3)*cos(theta2))+(l3*sin(theta2)*sin(theta3)
-l1-cos(theta2)*(l3*cos(theta3)+l2))*sin(theta1)
```

Listado B.1: Operador "Vector de Posición". Proyección en $\mathcal{B}_{xyz}$

```
l2*(cos(theta1)*cos(theta2)-sin(theta2)*sin(theta1))
+cos(theta1)*l1-l3*((cos(theta3)*sin(theta2)
+sin(theta3)*cos(theta2))*sin(theta1)
-(cos(theta3)*cos(theta2)-sin(theta2)*sin(theta3))
*cos(theta1));
0;
-(sin(theta1)*cos(theta2)+cos(theta1)*sin(theta2))*l2
-l3*((cos(theta3)*cos(theta2)-sin(theta2)*sin(theta3))
*sin(theta1)+cos(theta1)*(cos(theta3)*sin(theta2)
+sin(theta3)*cos(theta2)))-l1*sin(theta1)
```

Listado B.2: Suma de vectores. Proyección en $\mathcal{B}_{xyz}$

```
[l3+(cos(theta2)*l1+l2)*cos(theta3)
-sin(theta3)*l1*sin(theta2);
0;
sin(theta3)*(cos(theta2)*l1+l2)
+l1*cos(theta3)*sin(theta2)]
```

Listado B.3: Operador "Vector de Posición". Proyección en $\mathcal{B}_3$

```
l3+(cos(theta2)*l1+l2)*cos(theta3)
-sin(theta3)*l1*sin(theta2);
0;
sin(theta3)*(cos(theta2)*l1+l2)
+l1*cos(theta3)*sin(theta2)
```

Listado B.4: Suma de vectores. Proyección en $\mathcal{B}_3$

B.2. Matriz M y vector δ juntos

```
atom26 = cos(theta2);
atom75 = l1*atom26;
atom77 =  l2+atom75;
```

```
atom27 = sin(theta2);
atom76 = atom27*l1;
atom24 = m2*cg2x;
atom25 = m2*cg2z;
atom333 =  atom25*atom76+atom75*atom24;
atom1 = sin(theta3);
atom0 = cos(theta3);
atom43 = cg3z*m3;
atom42 = cg3x*m3;
atom387 =  atom43*( atom0*atom76+atom77*atom1)+atom42*(
   atom77*atom0-atom1*atom76);
atom390 =  l2*atom42*atom0+l2*atom43*atom1;
atom15 = sin(theta1);
atom14 = cos(theta1);
atom33 =  atom27*atom14+atom15*atom26;
atom30 =  atom14*atom26-atom15*atom27;
atom49 =  atom30*atom1+atom0*atom33;
atom46 =  atom30*atom0-atom1*atom33;
atom52 =  atom43*atom49+atom46*atom42;
atom57 = -m3*g;
atom98 = -l2*atom27;
atom97 = l2*atom26;
atom462 = atom57*atom14;
atom99 =  atom97+l1;
atom463 = -atom57*atom15*atom98;
atom36 =  atom25*atom33+atom30*atom24;
atom96 = l2*dtheta2;
atom104 =  dtheta1*atom98-atom27*atom96;
atom103 = -atom96*atom26;
atom377 = -atom97*dtheta1+atom103;
atom370 = atom104*atom14;
atom381 = -atom370-atom15*atom377;
atom105 = -dtheta1*atom99+atom103;
atom372 =  atom15*atom105+atom370;
atom373 = -atom15*atom104;
atom375 =  atom105*atom14+atom373;
atom379 =  atom377*atom14+atom373;
atom419 =  dtheta2*atom381*m3-atom372*dtheta1*m3;
atom416 =  dtheta1*m3*atom375+atom379*dtheta2*m3;
atom85 =  dtheta2+dtheta1;
atom92 =  dtheta3+atom85;
```

```
atom429 = -(atom92*atom92)*atom1;
atom428 = -(atom92*atom92)*atom0;
atom446 =  atom419*atom30+atom416*atom33-atom42*atom429+
   atom43*atom428;
atom93 = dtheta1*l1;
atom329 = -atom15*atom93;
atom330 = -atom14*atom93;
atom362 = -atom330*( atom24*atom33-atom30*atom25)*dtheta1+
   atom329*dtheta1*atom36;
atom444 = -atom52*( dtheta2*atom381-atom372*dtheta1)-(
   dtheta1*atom375+atom379*dtheta2)*( atom42*atom49-atom46*
   atom43);
atom40 = atom36*g;
atom56 = atom52*g;
atom491 = -Fx3*atom49+atom46*Fz3;
atom488 =  Fz3*atom49+atom46*Fx3;
atom497 =  atom491*atom14+atom15*atom488;
atom498 = ( atom488*atom14-atom15*atom491)*atom98;
atom406 = -atom333-I2yy-atom390-atom387-I3yy-l2*atom77*m3;
atom407 = -atom387-I3yy;
atom409 = -atom390-I3yy;

_MDelta[0] = -m2*(l1*l1)+-2.0*atom333-I2yy+-2.0*atom387-
   I3yy-( (atom77*atom77)+(atom76*atom76))*m3-I1yy;
_MDelta[1] = atom406;
_MDelta[2] = atom407;
_MDelta[3] = atom406;
_MDelta[4] = -I2yy+-2.0*atom390-(l2*l2)*m3-I3yy;
_MDelta[5] = atom409;
_MDelta[6] = atom407;
_MDelta[7] = atom409;
_MDelta[8] = -I3yy;
_MDelta[9] = -My2-atom40+atom497*atom99-atom77*atom446+l1*(
    ( Fz2*atom30-Fx2*atom33)*atom14+atom15*( Fz2*atom33+
   atom30*Fx2))+atom362-( atom27*(atom85*atom85)*atom24-
   atom25*(atom85*atom85)*atom26-atom329*dtheta1*m2*atom14+
   atom330*atom15*dtheta1*m2)*l1-My3-atom56+atom444-(
   atom419*atom33-atom43*atom429-atom42*atom428-atom416*
   atom30)*atom76-m2*g*l1*atom14-( m1*atom15*cg1z+cg1x*m1*
   atom14)*g-atom463-atom498+atom462*atom99;
_MDelta[10] = -My2-atom40+atom362-My3-atom56+atom444+atom97
```

```
   *atom497-atom463+atom97*atom462-atom498-l2*atom446;
_MDelta[11] = -My3-atom56+atom444;
```

Listado B.5: Código con ESST generado con lib_3D_MEC-GiNaC

```
atom26 = cos(theta2);
atom144 = l1*atom26*l2;
atom108 = (l1*l1);
atom145 = (l2*l2);
atom156 = l2*cg3x*m3;
atom157 =  atom156+l1*cg3x*m3;
atom111 = l1*m2*cg2x;
atom1 = sin(theta3);
atom27 = sin(theta2);
atom0 = cos(theta3);
atom88 = -atom1*atom27+atom26*atom0;
atom14 = cos(theta1);
atom85 =  dtheta2+dtheta1;
atom185 = -atom85*( l2*dtheta2+l2*dtheta1);
atom91 =  atom1*atom26+atom27*atom0;
atom92 =  dtheta3+atom85;
atom119 = -g*atom14;
atom178 = (atom92*atom92)*( cg3z*atom88-cg3x*atom91);
atom138 = -l1*(dtheta1*dtheta1);
atom195 = ( Fz3*atom0-atom1*Fx3+( (atom92*atom92)*( cg3x*
   atom88+atom91*cg3z)*atom27+atom26*atom178-atom138*atom27
   +g*( sin(theta1)*atom27-atom26*atom14))*m3)*l2;
atom227 = -My3+( atom185*cg3z+atom138*cg3z-g*cg3x)*m3;
atom142 =  ( cg2z*atom138-g*cg2x)*m2-My2;
atom245 =  ( atom144+atom145)*m3+atom157+atom156+atom111+
   I2yy+I3yy;
atom246 =  atom157+I3yy;
atom248 =  atom156+I3yy;

_MDelta[0] =  I1yy+2.0*atom157+2.0*atom111+I2yy+I3yy+(
   atom108+2.0*atom144+atom145)*m3+atom108*m2;
_MDelta[1] = atom245;
_MDelta[2] = atom246;
_MDelta[3] = atom245;
_MDelta[4] =  2.0*atom156+atom145*m3+I2yy+I3yy;
```

```
_MDelta[5] = atom248;
_MDelta[6] = atom246;
_MDelta[7] = atom248;
_MDelta[8] = I3yy;
_MDelta[9] = -atom195-atom227-atom142-l1*( ( ( cg2z*atom26-
   atom27*cg2x)*(atom85*atom85)+atom119)*m2-Fx2*atom27+
   atom26*Fz2)+g*cg1x*m1-l1*( Fz3*atom88-atom91*Fx3+(
   atom185*atom27+atom119+atom178)*m3);
_MDelta[10] = -atom195-atom227-atom142;
_MDelta[11] = -atom227;
```

Listado B.6: Código sin ESST generado con lib_3D_MEC-GiNaC

```
C1 = cos(q(1));
S1 = sin(q(1));
C2 = cos(q(2));
S2 = sin(q(2));
C3 = cos(q(3));
S3 = sin(q(3));

% = = Block_0_1_0_0_0_1 = =

% Forward Kinematics

BS11 = -qd(1)*qd(1);
AlF11 = s.g(3)*S1;
AlF31 = -s.g(3)*C1;
OM22 = qd(1)+qd(2);
BS12 = -OM22*OM22;
AlF12 = -(AlF31*S2-C2*(AlF11+BS11*s.dpt(1,1)));
AlF32 = AlF31*C2+S2*(AlF11+BS11*s.dpt(1,1));
AlM12_1 = s.dpt(1,1)*S2;
AlM32_1 = -s.dpt(1,1)*C2;
OM23 = qd(3)+OM22;

% = = Block_0_2_0_1_0_1 = =

% Backward Dynamics

FA13 =-(s.frc(1,3)+s.m(3)*(AlF32*S3+OM23*OM23*s.l(1,3)
```

```
      -C3*(A1F12+BS12*s.dpt(1,2))));
FA33 =-(s.frc(3,3)-s.m(3)*(A1F32*C3-OM23*OM23*s.l(3,3)
      +S3*(A1F12+BS12*s.dpt(1,2))));
CF23 =-(s.trq(2,3)-FA13*s.l(3,3)+FA33*s.l(1,3));
FB13_1 =s.m(3)*(s.l(3,3)+A1M12_1*C3
        -S3*(A1M32_1-s.dpt(1,2)));
FB33_1 =s.m(3)*(A1M12_1*S3+C3*(A1M32_1-s.dpt(1,2))
        -s.l(1,3));
CM23_1 =s.In(5,3)+FB13_1*s.l(3,3)-FB33_1*s.l(1,3);
FB13_2 =s.m(3)*(s.l(3,3)+s.dpt(1,2)*S3);
FB33_2 =-s.m(3)*(s.l(1,3)+s.dpt(1,2)*C3);
CM23_2 =s.In(5,3)+FB13_2*s.l(3,3)-FB33_2*s.l(1,3);
CM23_3 =s.In(5,3)+s.m(3)*s.l(1,3)*s.l(1,3)+s.m(3)
        *s.l(3,3)*s.l(3,3);
FA12 =-(s.frc(1,2)-s.m(2)*(A1F12+BS12*s.l(1,2)));
FA32 =-(s.frc(3,2)-s.m(2)*(A1F32-OM22*OM22*s.l(3,2)));
CF22 =-(s.trq(2,2)-CF23-FA12*s.l(3,2)+FA32*s.l(1,2)
      -s.dpt(1,2)*(FA13*S3-FA33*C3));
FB12_1 =s.m(2)*(A1M12_1+s.l(3,2));
FB32_1 =s.m(2)*(A1M32_1-s.l(1,2));
CM22_1 =s.In(5,2)+CM23_1+FB12_1*s.l(3,2)-FB32_1*s.l(1,2)
        +s.dpt(1,2)*(FB13_1*S3-FB33_1*C3);
CM22_2 =s.In(5,2)+CM23_2+s.m(2)*s.l(1,2)*s.l(1,2)
        +s.m(2)*s.l(3,2)*s.l(3,2)+s.dpt(1,2)
        *(FB13_2*S3-FB33_2*C3);
CF21 =-(s.trq(2,1)-CF22+s.dpt(1,1)*(C2*(FA32-FA13*S3
      +FA33*C3)-S2*(FA12+FA13*C3+FA33*S3))-s.l(1,1)
      *(s.frc(3,1)-s.m(1)*(A1F31-qd(1)*qd(1)*s.l(3,1)))
      +s.l(3,1)*(s.frc(1,1)-s.m(1)*(A1F11+BS11*s.l(1,1))));
CM21_1 =s.In(5,1)+CM22_1+s.m(1)*s.l(1,1)*s.l(1,1)
        +s.m(1)*s.l(3,1)*s.l(3,1)-s.dpt(1,1)*(C2*
        (FB32_1-FB13_1*S3+FB33_1*C3)-S2*(FB12_1+FB13_1
        *C3+ FB33_1*S3));

% = = Block_0_3_0_0_0_0 = =

% Symbolic Outputs

  M(1,1) = CM21_1;
  M(1,2) = CM22_1;
  M(1,3) = CM23_1;
```

```
M(2,1) = CM22_1;
M(2,2) = CM22_2;
M(2,3) = CM23_2;
M(3,1) = CM23_1;
M(3,2) = CM23_2;
M(3,3) = CM23_3;
c(1) = CF21;
c(2) = CF22;
c(3) = CF23;
```

Listado B.7: Código generado con ROBOTRAN

B.3. Vector Φ y matriz Φ_q juntos

```
atom26 = cos(theta2);
atom64 = l3*cos(theta_3);
atom66 =  l2+atom64;
atom65 = -sin(theta_3)*l3;
atom27 = sin(theta2);
atom68 = atom65*atom27;
atom69 =  atom26*atom66+atom68;
atom73 =  atom69+l1;
atom14 = cos(theta1);
atom74 = atom73*atom14;
atom15 = sin(theta1);
atom71 = atom26*atom65;
atom72 = -atom27*atom66+atom71;
atom75 = atom15*atom72;
atom78 = atom14*atom72;
atom79 =  atom78-atom73*atom15;
atom308 =  atom71-atom27*atom64;
atom311 = -atom26*atom64-atom68;

_PhiPhiq[0] =  atom74+atom75-l4;
_PhiPhiq[1] = atom79;
_PhiPhiq[2] = atom79;
_PhiPhiq[3] = -atom74-atom75;
_PhiPhiq[4] =  atom78-atom15*atom69;
```

```
_PhiPhiq[5] = -atom14*atom69-atom75;
_PhiPhiq[6] =  atom15*atom311+atom14*atom308;
_PhiPhiq[7] =  atom14*atom311-atom15*atom308;
```

Listado B.8: Código generado con lib_3D_MEC-GiNaC

```
q = s.q;
qd = s.qd;
qdd = s.qdd;
frc = s.frc;
trq = s.trq;

% === begin imp_aux ===

% === end imp_aux ===

% ===== BEGIN task 0 =====

% = = Block_0_0_0_0_0_1 = =

% Trigonometric Variables

  C1 = cos(q(1));
  S1 = sin(q(1));
  C2 = cos(q(2));
  S2 = sin(q(2));
  C3 = cos(q(3));
  S3 = sin(q(3));

% = = Block_0_1_0_0_0_1 = =

% Trigonometric Variables

%
  S1p2 = C1*S2+S1*C2;
  C1p2 = C1*C2-S1*S2;
  S3p1p2 = C3*S1p2+S3*C1p2;
  C3p1p2 = C3*C1p2-S3*S1p2;

% Constraints and Constraints Jacobian
```

```
  RL_1_12 = s.dpt(1,2)*C1;
  RL_1_32 = -s.dpt(1,2)*S1;
  RL_1_13 = s.dpt(1,3)*C1p2;
  RL_1_33 = -s.dpt(1,3)*S1p2;
  RL_1_15 = s.dpt(1,4)*C3p1p2;
  RL_1_35 = -s.dpt(1,4)*S3p1p2;
  PO_1_35 = RL_1_32+RL_1_33+RL_1_35;
  JT_1_15_1 = RL_1_32+RL_1_33+RL_1_35;
  JT_1_35_1 = -(RL_1_12+RL_1_13+RL_1_15);
  JT_1_15_2 = RL_1_33+RL_1_35;
  JT_1_35_2 = -(RL_1_13+RL_1_15);

% = = Block_0_1_0_0_1_0 = =

% Constraints and Constraints Jacobian

%
  h_1 = -(RL_1_12+RL_1_13+RL_1_15-s.dpt(1,1));

% = = Block_0_3_0_0_0_0 = =

% Symbolic Outputs

  h(1) = h_1;
  h(2) = -PO_1_35;
  Jac(1,1) = -JT_1_15_1;
  Jac(1,2) = -JT_1_15_2;
  Jac(1,3) = -RL_1_35;
  Jac(2,1) = -JT_1_35_1;
  Jac(2,2) = -JT_1_35_2;
  Jac(2,3) = RL_1_15;
```

Listado B.9: Código generado con ROBOTRAN

BIBLIOGRAFÍA

[1] Javier Garcia de Jalon and Eduardo Bayo. *Kinematic and Dynamic Simulation of Multibody Systems. The Real-Time Challenge*. Springer-Verlag, 1994.

[2] J.C. Samin and P. Fisette. *Symbolic Modeling of Multibody Systems*. Solid Mechanics and Its Applications. Springer, 2003.

[3] Javier Gil Javier Ros, Luis Arrondo and Xabier Iriarte. lib_3d_mec-ginac, a library for symbolic multibody dynamics. *EUROMECH Colloquium. Ferrol*, 2007.

[4] Javier Gil. *Preprocesador para la simulación dinámica de sistemas multicuerpo basado en álgebra simbólica*. Ph.D. thesis. Universidad Pública de Navarra, 2005.

[5] Ch. Schmitke P. Gossens. Symbolic computation techniques for multibody model development and dynamic analysis. *ECCOMAS Multibody Dynamics, Brussels*, 2011.

[6] Matlab symbolic toolbox. www.mathworks.com/products/symbolic/.

[7] MapleSim. www.maplesoft.com/products/maplesim.

[8] Sympy. www.sympy.org/.

[9] Giac/Xcas. http://www-fourier.ujf-grenoble.fr/~parisse/giac.

html.

[10] 3d_mec. www.imem.unavarra.es/3d_mec/.

[11] Robotran. www.robotran.be/.

[12] Neweul-M^2. www.itm.uni-stuttgart.de/research/neweul/neweul_en.php.

[13] PyDy: Multibody Dynamics with Python. www.pydy.org/.

[14] Dymola. www.3ds.com/products-services/catia/capabilities/systems-engineering/modelica-systems-simulation/dymola.

[15] Wolfram SystemModeler. www.wolfram.com/system-modeler.

[16] T. Kurz, P. Eberhard, C. Henninger, and W. Schiehlen. From neweul to neweul-m2: symbolical equations of motion for multibody system analysis and synthesis. *Multibody System Dynamics*, 24(1):25–41, 2010.

[17] Sven Erik Mattson, Hilding Elmqvist, and Jan F Broenink. Modelica: An international effort to design the next generation modelling language. *Journal A*, 38(3):16–19, 1997.

[18] Wolfram systemmodeler. http://www.wolfram.com/system-modeler/.

[19] N. Docquier, A. Poncelet, and P. Fisette. Robotran: a powerful symbolic gnerator of multibody models. *Mechanical Sciences*, 4(1):199–219, 2013.

[20] Thomas R. Kane and David A. Levinson. Dynamics, theory and applications. 1985.

[21] Nenad Kirćanski, Miomir Vukobratović, Aleksandar Timčenko, and Manja Kirćanski. Symbolic modeling in robotics: Genesis, applicaton, and future prospects. *Journal of Intelligent and Robotic Systems*, 8(1):1–19, 1993.

[22] Wisama Khalil and Denis Creusot. Symoro+: A system for the symbolic modelling of robots. *Robotica*, 15:153–161, 3 1997.

[23] P. Corke. A robotics toolbox for matlab. *Robotics Automation Magazine, IEEE*, 3(1):24–32, 1996.

[24] Peter I. Corke. *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer, 2011.

[25] Roy Featherstone. The calculation of robot dynamics using articulated-

body inertias. *The International Journal of Robotics Research*, 2(1):13–30, 1983.

[26] A. Kecskeméthy, T. Krupp, and M. Hiller. Symbolic processing of multiloop mechanism dynamics using closed-form kinematics solutions. *Multibody System Dynamics*, 1(1):23–45, 1997.

[27] X. Chenut, P. Fisette, and J.-Cl. Samin. Recursive formalism with a minimal dynamic parameterization for the identification and simulation of multibody systems. application to the human body. *Multibody System Dynamics*, 8(2), 2002.

[28] KurtS. Anderson and YuHung Hsu. Analytical fully-recursive sensitivity analysis for multibody dynamic chain systems. *Multibody System Dynamics*, 8(1):1–27, 2002.

[29] Javier Ros, Roberto Yoldi, Aitor Plaza, and Xabier Iriarte. Real-time hardware-in-the-loop simulation of a hexaglide type parallel manipulator on a real machine controller. In *New Trends in Mechanism and Machine Science*, volume 7 of *Mechanisms and Machine Science*, pages 587–597. Springer Netherlands, 2013.

[30] Stefano Falomi, Monica Malvezzi, and Enrico Meli. Multibody modeling of railway vehicles: Innovative algorithms for the detection of wheel–rail contact points. *Wear*, 271(1–2):453 – 461, 2011. Proceedings of the 8th International Conference on Contact Mechanics and Wear of Rail / Wheel Systems, Florence, 2009.

[31] Evaluation of driver's behavior with multibody-based driving simulator. *Multibody System Dynamics*, 17(2-3):195–208, 2007.

[32] J. Cuadrado, J. Cardenal, and E. Bayo. Modeling and solution methods for efficient real-time simulation of multibody dynamics. *Multibody System Dynamics*, 1(3):259–280, 1997.

[33] J. Agulló. *Mecánica de la partícula y del sólido rígido*. Publicaciones OK PUNT, 1996.

[34] A. A. Shabana and J. R. Sany. An augmented formulation for mechanical systems with non-generalized coordinates: Application to rigid body contact problems. *Nonlinear Dynamics*, 24:183–204, 2001.

[35] A. A. Shabana, K. E. Zaazaa, J. L. Escalona, and J. R. Sany. Development of elastic force model for wheel/rail contact problems. *Journal of Sound and Vibration*, 269(1-2):295–325, jan 2004.

[36] AhmedA. Shabana, Mahmoud Tobaa, Hiroyuki Sugiyama, and KhaledE. Zaazaa. On the computer formulations of the wheel/rail contact problem. *Nonlinear Dynamics*, 40(2):169–193, 2005.

[37] João Pombo, Jorge Ambrósio, and Miguel Silva. A new wheel–rail contact model for railway dynamics. *Vehicle System Dynamics*, 45(2):165–189, 2007.

[38] Jury Auciello, Enrico Meli, Stefano Falomi, and Monica Malvezzi. Dynamic simulation of railway vehicles: wheel/rail contact analysis. *Vehicle System Dynamics*, 47(7):867–899, 2009.

[39] Martin Arnold. Simulation algorithms in vehicle system dynamics. 2004.

[40] Alessandro Tasora and Paolo Righettini. Sliding contact between free-form surfaces. 2003.

[41] P. Fisette, J.M. Péterkenne, B. Vaneghem, and J.C. Samin. A multibody loop constraints approach for modelling cam/follower devices. *Nonlinear Dynamics*, 22(4):335–359, 2000.

[42] T.H. Cormen. *Introduction to Algorithms*. MIT Press, 2009.

[43] Javier Gil Javier Ros and Xabier Iriarte. A symbolic algorithm in real time simulation of multibody systems. *MULTIBODY DYNAMICS 2006, ECCOMAS*, 2007.

[44] Javier Ros, Aitor Plaza, Xabier Iriarte, and Jokin Aginaga. Inertia transfer concept based general method for the determination of the base inertial parameters. *Multibody System Dynamics*, pages 1–21, 2015.

[45] Javier Ros, Xabier Iriarte, and Vicente Mata. 3d inertia transfer concept and symbolic determination of the base inertial parameters. *Mechanism and Machine Theory*, 49(0):284 – 297, 2012.

[46] W. Khalil and J.-F. Kleinfinger. Minimum operations and minimum parameters of the dynamic models of tree structure robots. *Robotics and Automation, IEEE Journal of*, 3(6):517–526, 1987.

ÍNDICE DE FIGURAS

2.1. Resolución numérica sin integración 35
2.2. Integración numérica para cinemática (Euler explícito) 37
2.3. Integración numérica para cinemática con proyección 39
2.4. Integración numérica para dinámica 40
2.5. Matriz de de masa del ejemplo A.5 46

3.1. Expresión expandida en forma de árbol 60
3.2. recursive_tree($ExpIN$) 61
3.3. Recorrido del árbol según el algoritmo recursivo 62
3.4. atomize_ex($ExpIn$) 64
3.5. unatomize_ex($ExpIn$) 67
3.6. recurvive_expression_substitution($ExpIn, Sym, Val$) 69
3.7. recurvive_expression_substitution($ExpIn, Vsym, Val$) 71
3.8. recurvive_differentiation($ExpIn, Sym$) 74
3.9. atoms_in_exp(ExpIn, ListIn, AtomTableIn) 78
3.10. atom_list(Mat) 80
3.11. opt_atom_list(Mat) 84
3.12. opt_unatomize_ex 88
3.13. Integración numérica. Evaluación de funciones 89

4.1. Ejemplo de estructura de bases 96
4.2. Ejemplo de estructura de puntos 101
4.3. Sólido genérico 102
4.4. Algoritmo Rotation_Matrix 108
4.5. Algoritmo Position_Vector 111
4.6. Algoritmo Angular_Velocity_Vector 115
4.7. Velocity_Vector(R_A, P_B,P_{O1},$\mathbf{v}_{IN}$) 118
4.8. Velocity_Vector(R_A, P_B, S_C) 119
4.9. Cálculo de la matriz auxiliar $\mathbf{M}_{\omega\omega}$ 133
4.10. Cálculo de $\mathbf{M}$ y δ 142

5.1. Giros consecutivos entorno al mismo eje 169
5.2. Nuevo esquema de bases donde se han simplificado los giros . 170
5.3. Esquema del mecanismo 171

8.1. Matriz de de masa del ejemplo A.5 209

A.1. Ejemplo 1. Péndulo invertido 212
A.2. Diagrama de bases 212
A.3. Diagrama de puntos 212
A.4. Ejemplo 2. Cuadrilátero articulado 213
A.5. Diagrama de bases 213
A.6. Diagrama de puntos 214
A.7. Mecanismo Biela, manivela, elipse 215
A.8. Diagrama de bases 216
A.9. Diagrama de puntos 217
A.10. Modelo en CAD de la plataforma 218
A.11. Diagrama de bases 219
A.12. Diagrama de puntos 220
A.13. Modelo esquemático de la locomotora FEVE 3800 221
A.14. Modelo en CAD. 222
A.15. Diagrama de bases del bogie tractor 224
A.16. diagrama de puntos 225

ÍNDICE DE TABLAS

3.1. Tabla hash T 55
3.2. Tabla hash T 57
3.3. Tabla hash T 58
3.4. Prioridad de los operadores 59

6.1. Evaluación de algoritmos genéricos. Operaciones algebraicas . . 183
6.2. Evaluación de algoritmos genéricos: Tiempo 184
6.3. Tiempos de ejecución en segundos 185
6.4. Número de átomos 185
6.5. Número de operaciones 185
6.6. Número de operaciones necesarias para la evaluación de $\mathbf{M}$ y δ 186
6.7. Comparación del tamaño del código de $\mathbf{\Phi}_q$ y $\mathbf{\Phi}$ exportados conjuntamente 187
6.8. Tiempo de evaluación de $\mathbf{M}$ y δ en milisegundos 188
6.9. Operaciones necesarias para la evaluación 189
6.10. Número de operaciones necesarias para evaluar numéricamente $\mathbf{M}\delta$ 191
6.11. Tiempo en segundos de cálculo de $\mathbf{M}$ y δ 192
6.12. Número de operaciones para evaluar numéricamente $\mathbf{M}\delta$ . . . 192
6.13. Tiempo en segundos de cálculo de $\mathbf{M}$ y δ 194

6.14. Código generado para el mecanismo de cuatro barras 195
6.15. Comparación del tamaño del código de $\mathbf{M}\delta$ (exportados conjuntamente) 196

NOTACIÓN

Nomenclatura

$\boldsymbol{u}$ Vector $\boldsymbol{u}$

$\mathbf{T}$ Tensor $\mathbf{T}$

$\mathbf{A}$ Matriz $\mathbf{A}$

$\mathbf{1}$ Matriz identidad

$\{\boldsymbol{u}\}_{\mathcal{B}}$ Componentes del vector $\boldsymbol{u}$ proyectadas en la base $\mathcal{B}$

S_i Sólido i-ésimo

R_i Referencia i-ésima

$\mathcal{B}_i$ Base i-ésima

B_i Punto de B del sólido S_i

P_i Punto genérico

m_i Masa del sólido S_i

$\mathbf{I}_{B_i}^{S_i}$ Tensor de inercia del sólido S_i en el punto B_i

$\mathbf{r}_{P_0}^{P_1}$ Vector de posición del punto P_1 respecto al punto P_0

$\mathbf{v}_{R_0}^{P_1}$ Vector de velocidad del punto P_1 con respecto a la referencia R_0

$\mathbf{a}_{R_0}^{P_1}$ Vector de aceleración del punto P_1 con respecto a la referencia R_0

$\mathbf{R}_{\mathcal{B}_0}^{\mathcal{B}_1}$ Matriz de cambio de base. De la base $\mathcal{B}_0$ a la base $\mathcal{B}_1$

$\boldsymbol{\omega}_{\mathcal{B}_0}^{\mathcal{B}_1}$ Velocidad Angular de la base $\mathcal{B}_1$ respecto de la base $\mathcal{B}_0$

$\dot{\boldsymbol{\omega}}_{\mathcal{B}_0}^{\mathcal{B}_1}$ Aceleración Angular de la base $\mathcal{B}_1$ respecto de la base $\mathcal{B}_0$

$\mathbf{h}_{B_i}^{S_i}$ Momento angular del sólido S_i en el punto B_i

$\mathbf{f}$ Vector de Fuerzas

$\mathcal{F}^{S_i}$ Vector de Fuerzas de Inercia del sólido S_i

$\mathbf{m}$ Vector de Momento

$\mathcal{M}_{B_i}^{S_i}$ Vector de Momentos de Inercia del sólido S_i en el punto B_i

w_{B_i} Torsor aplicado en el punto B_i

$\mathcal{W}_{B_i}^{S_i}$ Torsor de inercia del sólido S_i en el punto B_i

$\mathbf{e}$ Vector de ecuaciones dinámicas

$\boldsymbol{q}$ Vector de Coordenadas Generalizadas

$\dot{\boldsymbol{q}}$ Vector de Velocidades Generalizadas

$\ddot{\boldsymbol{q}}$ Vector de Aceleraciones Generalizadas

$\boldsymbol{\Phi}$ Vector de ecuaciones cinemáticas para las coordenadas generalizadas

$\dot{\boldsymbol{\Phi}}$ Vector de ecuaciones cinemáticas para las velocidades generalizadas

$\boldsymbol{\gamma}$ Término independiente del problema de aceleración

$\boldsymbol{\beta}$ Término independiente del problema de velocidad

$\boldsymbol{\Phi}_q$ Matriz jacobiana de $\boldsymbol{\Phi}$ respecto de las coordenadas generalizadas

$\dot{\mathbf{\Phi}}_{\dot{q}}$ Matriz jacobiana de $\dot{\mathbf{\Phi}}$ respecto de las velocidades generalizadas

$\mathbf{\Phi}_{\mathbf{t}}$ Matriz jacobiana de $\mathbf{\Phi}$ respecto del tiempo

δ Vector de fuerzas generalizadas

$\mathbf{M}$ Matriz de masa

λ Vector de multiplicadores de Lagrange

$[\![\boldsymbol{u}]\!]$ Vector extendido $\boldsymbol{u}$

$[\![\mathbf{T}]\!]$ Tensor extendido $\mathbf{T}$

Acrónimos

DSM Dinámica de sistemas multicuerpo

PPV Principio de las Potencias Virtuales

ESST Expresión susceptible de simplificación trigonométrica