The 6th Asian Conference on Multibody Dynamics
August 26-30, 2012, Shanghai, China

Simulation of the anchor lifting maneuver of a ship using contact detection techniques and continuous force models

Daniel Dopico Dopico
Laboratorio de Ingeniería Mecánica
Universidade da Coruña
Ferrol, A Coruña, 15403, Spain
Email: ddopico@udc.es

Alberto Luaces Fernández
Laboratorio de Ingeniería Mecánica
Universidade da Coruña
Ferrol, A Coruña, 15403, Spain
Email: aluaces@udc.es

Florian Michaud
Dep. Ergonomy, Design and Mech. Eng.
University of Tech. Belfort-Montbeliard
Belfort, 90010, France
Email: florian.michaud@utbm.fr

Javier Cuadrado
Laboratorio de Ingeniería Mecánica
Universidade da Coruña
Ferrol, A Coruña, 15403, Spain
Email: javicuad@cdf.udc.es

ABSTRACT

To develop the geometric design of a ship's hull that guarantees a correct anchor maneuver is not an easy task. The engineer responsible for the design has to make sure that the anchor does not jam up during the lifting process, and that the position adopted by the anchor on the hull is acceptable when completely lifted. Nowadays, the design process is based on wooden scale models of the hull, anchor and chain links, which are expensive and time consuming, and do not offer the required precision. For these reasons, having a computational tool to simulate the multibody system would be very helpful for the designers.

In this work the anchor lifting maneuver of a ship is simulated, taking into account the behavior of the anchor and the chain. To consider the contact forces between them and the hull of the ship, a general contact algorithm for rigid bodies and a particular contact algorithm for the chain links is described.

1 INTRODUCTION

To design the part of the hull of the ship in which the anchor will be placed when completely lifted, is a geometrical problem. The correct geometrical design has to guarantee that the anchor does not jam up during the lifting process and the position adopted by the anchor on the hull is acceptable when completely lifted, independently of initial position and velocity of the anchor when the lifting maneuver begins. Another condition to ensure is that the anchor does not keep hitting the hull or it slips if the sea conditions are not good.

Nowadays, to check all this design conditions the only tools at the designers disposal, are: a) the engineer's experience, of course; b) wooden scale models of the hull, anchor and chain links, which are expensive, time consuming and, more important, they do not offer the required precision. Up to the authors knowledge it does not exist neither a software nor a complete simulation of the anchor lifting maneuver of a ship. For these reasons, having a multibody code to simulate the system would be very helpful and valuable for the designers.

The dynamics of the system is governed by the large amount of contacts that take place between its bodies. Because of this reason, the contact models and contact detection algorithms play a key role in this application. Two aspects must be taken into consideration to successfully simulate the behavior of multibody systems with contacting bodies. First, an adequate contact model must be chosen for the application: it has to be accurate, stable and robust enough for the applications to tackle. Second, the geometrical detection of the contact events is necessary to feed the contact model, independently of the contact model chosen. The treatment of contact is still a challenging topic in multibody dynamics.

In a broad sense there are two families of methods to solve the normal contact problem in multibody systems composed of rigid bodies, see [11, 6]: the discontinuous and the continuous approaches. The discontinuous approaches are impulsive and for this reason suited to impact forces, while the continuous approaches are better suited to applications in which it is expected to occur permanent contacts or at least contacts of a significant duration; the normal force model described in this work is continuous and based on regularized forces. Moreover, it is worth to mention that friction is a complex nonlinear phenomenon. Up to these days, there is not a universally accepted model to calculate the friction force between bodies that accurately predicts all

the experimentally observed behaviors. under dry conditions. In this work, a tangential friction model developed by the authors in a previous work is described [5]: the model includes dry friction, stiction at low velocities and a viscous component.

Independently of the contact model chosen, it is necessary to detect the contacts to feed the contact model, which turns to be a geometrical problem. In many cases it is possible to make some assumptions, for example when the exact geometry of the bodies are known, it is possible to simplify the task of detecting the contacts between bodies by using the analytical equations of the geometry [12, 1], or by replacing the actual one by surrounding primitives like spheres or boxes. Nevertheless, when the exact geometries of the bodies are not known (because they are read from CAD models) or when the geometries are very complex, more general strategies have to be developed in order to detect the contacts and carry out all the calculations necessary to feed the contact model [2, 10]. The algorithms described in this work fall within both categories: the first algorithm described, is a general algorithm developed to detect contacts between bodies with complex 3D geometries given by CAD models in triangular mesh format; the second algorithm, is a specific algorithm to detect contact between chain links and complex 3D geometries given by CAD models in triangular mesh format.

Between the large number of formulations of the equations of motion existent (see e.g. [8]), the penalty and augmented Lagrangian formulations, [3] [4], are characterized by transforming the constraints into forces proportional to the constraints violation. This technique, used along this work, is similar and compatible to that of the continuous force models for normal contact,which relate the force and deformation of the bodies in contact to avoid the penetration between them.

2 MULTIBODY FORMULATION

The multibody formulation chosen for this work is an index-3 Augmented Lagrangian in mixed coordinates (natural plus relative), with projections of velocities and accelerations onto the constraints manifolds. The integration scheme adopted was the the implicit single-step trapezoidal rule. The mentioned formulation was described in [3].

3 DESCRIPTION OF THE CONTACT MODEL

The contact forces approach proposed for this work comprises two different models: the normal force model and the tangential force model. The two sub-models are presented separately in subsequent sections. A more detailed description of the contact model is given in [5].

For simplicity reasons, the contact model is going to be explained for the collision of a spherical and a flat body but it is easily generalized to bodies with arbitrary shapes.

3.1 Normal force model

The normal force model chosen for this work was the Flores model [7]. The model is very similar to the Hunt-Crossley model described in [5] with a little difference in the dissipation term, therefore the detailed description given there is valid here with little changes. The expression for the normal force has the following form.

$$\mathbf{F}_n = k_n \delta^n \left(1 + \frac{8\,(1-\epsilon)}{5\epsilon} \frac{\dot{\delta}}{\dot{\delta}_0} \right) \mathbf{n} \tag{1}$$

where k_n is the equivalent stiffness of the contact and depends on the shape and material properties of the colliding bodies, n is the Hertz's exponent, $\delta = R_{sph} - |\mathbf{p}_{center} - \mathbf{p}_{contact}|$ is the indentation, $\dot{\delta}$ its temporal derivative, $\dot{\delta}_0$ is the relative normal velocity between the colliding bodies when the contact is detected, ϵ is the coefficient of restitution, and $\mathbf{n}$ is the direction of the force. The subscript "*n*" comes from "*normal*".

The value of k_n depends on the shape and materials of the colliding bodies.

3.2 Tangential force model

The tangential force model for the friction force was developed in detail in [5] and it is based on Coulomb's law including stiction plus a viscous friction term. The general form of this force is the following,

$$\mathbf{F}_t = \kappa \mathbf{F}_{stick} + (1-\kappa)\,\mathbf{F}_{slide} - \mu_{visc}\mathbf{v}_t \tag{2}$$

In the previous expression, the first two terms constitute the dry friction, while the third term accounts for the viscous friction. For the smooth transition between sticking and slipping the dry friction force is divided in two components coupled by a smooth function, following the ideas proposed in [9]. The subscript "*t*" comes from "*tangential*".

In (2), μ_{visc} is the viscous damping coefficient, $\mathbf{F}_{stick}$ and $\mathbf{F}_{slide}$ are the components of the stiction and slipping forces, κ is a smooth function of the tangential velocity, $\mathbf{v}_t$, which is defined in terms of the central point of the contact region, $\mathbf{p}_{contact}$, and the normal vector at the contact,$\mathbf{n}$, as follows.

$$\mathbf{v}_t = \dot{\mathbf{p}}_{contact} - \left(\mathbf{n}^{\mathrm{T}}\,\dot{\mathbf{p}}_{contact}\right)\mathbf{n} \tag{3}$$

The mentioned function, κ, was taken from [9] and has the following form.

$$\kappa = \mathrm{e}^{-\left(\mathbf{v}_t^{\mathrm{T}}\mathbf{v}_t\right)/v_{stick}^2} \tag{4}$$

Equation (2) showed that the total force is composed of three contributions: the sliding dry friction force at high velocities, the stiction force at low velocities and the viscous friction force. The sliding force is given by the classical Coulomb expression, while the stiction force is considered by means of viscoelastic elements acting between the colliding bodies. To see the detailed expressions of the sliding and stiction forces see [5].

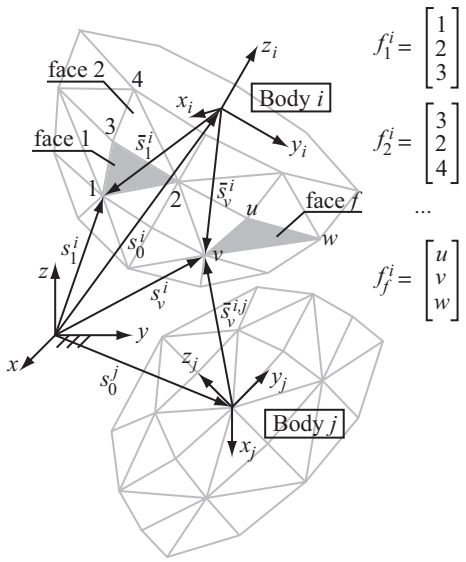

Figure 1. Body mesh.

4 CONTACT DETECTION ALGORITHMS I: GENERAL DETECTION ALGORITHM

The algorithms described in this section perform the detection of the contacts existing between pairs of bodies with arbitrary shapes. Moreover, for each one of the contacts found, they perform the calculation of the necessary data to feed the contact model described in section 3. The colliding bodies considered in this section have complex 3D geometries given by CAD models in triangular mesh format and their geometries are read from CAD files. Since the geometry is not known in advance, the detection algorithm has to be completely general. All the existent CAD packages have translators from the native format to triangular mesh formats like *obj* or *stl*.

The mathematical representation of the triangular mesh of the body *i*, with n_v vertexes and n_f triangular faces, expressed in the local framework of the body, is the following (see Figure 1).

$$\bar{\mathbf{s}}_v^i;\ 1 \leq v \leq n_v;\ n_v \geq 3 \text{ (list of vertexes)} \tag{5}$$

$$\mathbf{f}_f^i = \begin{bmatrix} f_{f1}^i \\ f_{f2}^i \\ f_{f3}^i \end{bmatrix};\ 1 \leq f \leq n_f;\ n_f \geq 1 \text{ (list of triangles)} \tag{6}$$

Thus, the three vertexes that compose the face *f* of the body are obtained by replacing each one of the indexes from (6) in (5): $\bar{\mathbf{s}}_{f_{f1}^i}^i$, $\bar{\mathbf{s}}_{f_{f2}^i}^i$ and $\bar{\mathbf{s}}_{f_{f3}^i}^i$.

The vertexes list of the mesh in global coordinates is obtained by means of the body transformation matrix $\mathbf{A}^i$, using homogeneous coordinates.

$$\mathbf{s}_v^{*i} = \mathbf{A}^i \bar{\mathbf{s}}_v^{*i} \Rightarrow \begin{bmatrix} \mathbf{s}_v^i \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^i & \mathbf{s}_0^i \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \bar{\mathbf{s}}_v^i \\ 1 \end{bmatrix} \tag{7}$$

It will be necessary, for some of the algorithms of this section, to express the vertexes of a body in the local frame of other body. Thus, the vertexes of body *i* in the local frame of body *j*.

$$\bar{\mathbf{s}}_v^{*i,j} = \left(\mathbf{A}^j\right)^{-1} \mathbf{s}_v^{*i} = \left(\mathbf{A}^j\right)^{-1} \mathbf{A}^i \bar{\mathbf{s}}_v^{*i} = \mathbf{A}^{i,j} \bar{\mathbf{s}}_v^{*i} \tag{8}$$

where,

$$\left(\mathbf{A}^j\right)^{-1} = \begin{bmatrix} \left(\mathbf{R}^j\right)^{\mathrm{T}} & -\left(\mathbf{R}^j\right)^{\mathrm{T}} \mathbf{s}_0^j \\ 0 & 1 \end{bmatrix} \tag{9}$$

$$\mathbf{A}^{i,j} = \begin{bmatrix} \left(\mathbf{R}^j\right)^{\mathrm{T}} \mathbf{R}^i & \left(\mathbf{R}^j\right)^{\mathrm{T}} \left(\mathbf{s}_0^i - \mathbf{s}_0^j\right) \\ 0 & 1 \end{bmatrix} \tag{10}$$

Replacing equation (10) in equation (8).

$$\bar{\mathbf{s}}_v^{i,j} = \left(\mathbf{R}^j\right)^{\mathrm{T}} \left(\mathbf{s}_0^i - \mathbf{s}_0^j\right) + \left(\mathbf{R}^j\right)^{\mathrm{T}} \mathbf{R}^i \bar{\mathbf{s}}_v^i \tag{11}$$

4.1 CAD models preprocessing

This task is performed only once at the beginning of the simulation and carries out the preprocessing of the CAD models, which includes basically the calculation of the neighbors of each face and the creation of the data structures to calculate the pairs of triangles in collision. As was mentioned before, the geometry of each body, is given by a mesh of triangles.

4.2 Calculation of the collision pairs

For this task the open source library OPCODE (Optimized Collision Detection) was partially used. The tree structures to order and subdivide the faces were taken from Opcode, but the algorithms to detect if two triangles collide were completely reprogrammed, since the original algorithms of Opcode offer inaccurate (and sometimes completely incorrect) results, not valid for multibody dynamics. The algorithms described in this section, return a list of n_p disordered colliding triangle pairs between bodies *i* and *j* (see Figure 2) and a list of n_p disordered intersection segments between the pairs, expressed in the local frame of body *j*.

$$\mathbf{pairs}^{ij} = \left\{ f_k^i, f_k^j \right\};\ 0 \leq k \leq n_p \text{ (list of pairs)} \tag{12}$$

$$\overline{\mathbf{isects}}^{ij,j} = \left\{ \overline{\mathbf{i}_{k,1}\mathbf{i}_{k,2}} \right\};\ 0 \leq k \leq n_p \text{ (list of segments)} \tag{13}$$

In equations (12) and (13) the super index ^{ij}, indicates collision between bodies *i* and *j*, while in equation (13) the over line along with the superindex ^{,j} indicates local coordinates of body *j*.

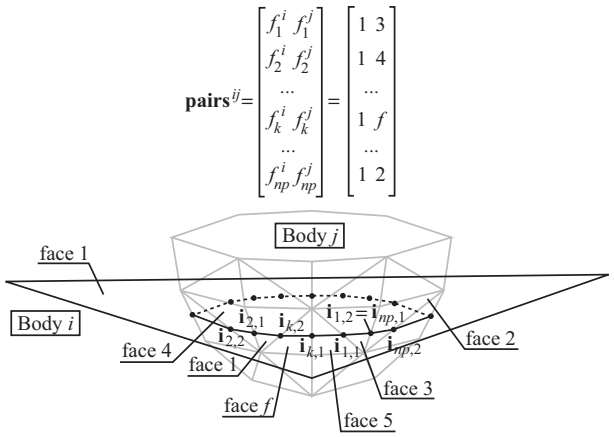

Figure 2. Collision pairs and segments between bodies *i* and *j*

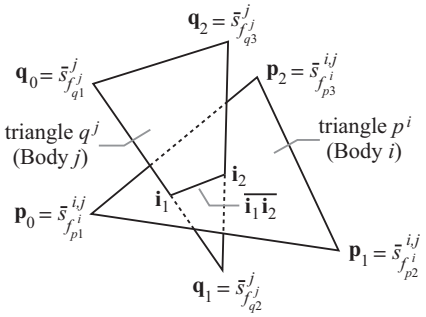

Figure 3. Triangle-triangle intersection

4.2.1 Box-Box overlap algorithm

The original Opcode algorithm to test AABB (Axis Aligned Bounding Box) with AABB collisions offered incorrect results for local aligned or almost local aligned boxes. The new algorithm checks this alignment.

4.2.2 Triangle-Triangle overlap algorithm

The original Opcode algorithm was based on projections while the new programmed algorithm is based on the direct solution of edge-triangle intersections which is much more robust. Moreover the new algorithm was programmed in double precision. The intersection of the triangles is typically a straight line segment, the new algorithm includes also the calculation of the extreme points of the segment.

In Figure 3 the triangles p^i (from body *i*) and q^j (from body *j*), are intersecting. The triangles are composed of the vertexes $\mathbf{f}_p^i$ and $\mathbf{f}_q^j$ respectively. To check the intersection between them, it is enough to check each edge of triangle p^i against q^j and vice versa. If the triangles overlap, two edge-triangle intersections exist. To illustrate the edge-triangle test, the intersection between edge $\overline{f_{q1}^j f_{q2}^j}$ and triangle p^i is calculated here.

Let's call $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$ the vertexes of triangle p^i and $\mathbf{q}_0$, $\mathbf{q}_1$, $\mathbf{q}_2$ the vertexes of triangle q^j. Using equation (11), it is possible to express all the vertexes in the local frame of body *j*.

$$\begin{aligned}
\mathbf{p}_0 = \bar{\mathbf{s}}^{i,j}_{f^i_{p1}} &= \left(\mathbf{R}^j\right)^{\mathrm{T}}\left(\mathbf{s}_0^i - \mathbf{s}_0^j\right) + \left(\mathbf{R}^j\right)^{\mathrm{T}}\mathbf{R}^i\bar{\mathbf{s}}^i_{f^i_{p1}} \\
\mathbf{p}_1 = \bar{\mathbf{s}}^{i,j}_{f^i_{p2}} &= \left(\mathbf{R}^j\right)^{\mathrm{T}}\left(\mathbf{s}_0^i - \mathbf{s}_0^j\right) + \left(\mathbf{R}^j\right)^{\mathrm{T}}\mathbf{R}^i\bar{\mathbf{s}}^i_{f^i_{p2}} \\
\mathbf{p}_2 = \bar{\mathbf{s}}^{i,j}_{f^i_{p3}} &= \left(\mathbf{R}^j\right)^{\mathrm{T}}\left(\mathbf{s}_0^i - \mathbf{s}_0^j\right) + \left(\mathbf{R}^j\right)^{\mathrm{T}}\mathbf{R}^i\bar{\mathbf{s}}^i_{f^i_{p3}}
\end{aligned} \tag{14}$$

$$\begin{aligned}
\mathbf{q}_0 &= \bar{\mathbf{s}}^j_{f^j_{q1}} \\
\mathbf{q}_1 &= \bar{\mathbf{s}}^j_{f^j_{q2}} \\
\mathbf{q}_2 &= \bar{\mathbf{s}}^j_{f^j_{q3}}
\end{aligned} \tag{15}$$

The equations of triangle p^i.

$$\begin{aligned}
\mathbf{r}_t = \mathbf{p}_0 + \mu_1\mathbf{u}_1 + \mu_2\mathbf{u}_2 &\left\{ \begin{array}{l} \dfrac{\mu_1}{l_1} + \dfrac{\mu_2}{l_2} \le 1 \\ \mu_1 \ge 0 \\ \mu_2 \ge 0 \end{array} \right\} \\
\mathbf{u}_1 = \frac{\mathbf{p}_1 - \mathbf{p}_0}{l_1};\ \mathbf{u}_2 &= \frac{\mathbf{p}_2 - \mathbf{p}_0}{l_2} \\
l_1 = |\mathbf{p}_1 - \mathbf{p}_0|\,;\ l_2 &= |\mathbf{p}_2 - \mathbf{p}_0|
\end{aligned} \tag{16}$$

The equations of edge $\overline{f_{q1}^j f_{q2}^j}$.

$$\begin{aligned}
\mathbf{r}_e = \mathbf{q}_0 + \eta\mathbf{v};\ 0 \le \eta \le d \\
\mathbf{v} = \frac{\mathbf{q}_1 - \mathbf{q}_0}{d};\ d = |\mathbf{q}_1 - \mathbf{q}_0|
\end{aligned} \tag{17}$$

Making $\mathbf{r}_t = \mathbf{r}_e$.

$$\begin{aligned}
\mathbf{p}_0 + \mu_1\mathbf{u}_1 + \mu_2\mathbf{u}_2 = \mathbf{q}_0 + \eta\mathbf{v} \Rightarrow \\
[-\mathbf{v}\ \mathbf{u}_1\ \mathbf{u}_2]\begin{bmatrix} \eta \\ \mu_1 \\ \mu_2 \end{bmatrix} = [\mathbf{q}_0 - \mathbf{p}_0] \Rightarrow \\
\mathbf{A}\mathbf{x} = \mathbf{b}
\end{aligned} \tag{18}$$

There are 3 possible situations.

1. $\mathrm{rank}(\mathbf{A}) = 3 = \mathrm{rank}([\mathbf{A}|\mathbf{b}])$. The edge intersects the plane which contains the triangle. In case $\frac{\mu_1}{l_1} + \frac{\mu_2}{l_2} \le 1$ with, $\mu_1, \mu_2 \ge 0$ and $0 \le \eta \le d$ the intersection lays into the triangle, otherwise the edge is discarded. In case of intersection, the intersection point can be easily calculated replacing η in (17).
2. $\mathrm{rank}(\mathbf{A}) = 2 = \mathrm{rank}([\mathbf{A}|\mathbf{b}])$. The edge is contained in the plane. The triangles might be coplanar or adjacent. The edge is discarded.
3. $\mathrm{rank}(\mathbf{A}) = 2 \neq \mathrm{rank}([\mathbf{A}|\mathbf{b}]) = 3$. The edge is parallel to the plane which contains the triangle. The edge is discarded.

The three edges of triangle q^j are successively checked against triangle p^i and after, the three edges of p^i against triangle q^j. In case two intersections are obtained, the triangles overlap and the intersection, $\overline{\mathbf{i}_1\mathbf{i}_2}$, is given by the segment composed of the intersection points.

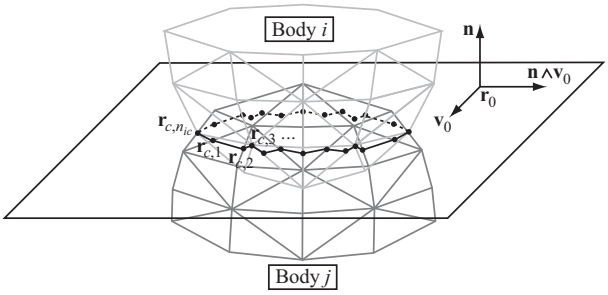

Figure 4. Contact plane calculation

4.3 Contact regions contour closure algorithm

From the disordered colliding triangles list (12) of section 4.2, this algorithm performs the closure of the contours of the different contact regions, grouping together the collision pairs by regions and ordering the collision pairs of each region (and its contour segments) in a way that permits to follow the contours from a segment to the adjacent one. The algorithm uses the topological information about the neighbors, calculated in section 4.1, to find out which collision pairs belong to the same region and to order the segments inside each region.

Once the segments are grouped by regions and the segments of each region are ordered, the algorithm merges the adjacent segments removing the coincident vertexes in (13), by means of a simple numerical procedure. Finally, the algorithm returns a list with the existent n_c 3D contours given by their ordered vertexes.

$$\begin{aligned} \overline{\mathbf{c}}_c^{ij,j} &= \left[\begin{array}{cccc} \mathbf{r}_{c,1} & \mathbf{r}_{c,2} & \dots & \mathbf{r}_{c,n_{ic}} \end{array} \right]; \\ & 0 \leq c \leq n_c \text{ (list of contours)} \end{aligned} \tag{19}$$

In (19), n_{ic} is the number of vertexes of the contour *c*, the super index ^{ij}, indicates collision between bodies *i* and *j*, and the over line along with the superindex ^{,j} indicates local coordinates of body *j*.

4.4 Contact plane calculation algorithm

For each one of the contact regions identified in section 4.3, the algorithm calculates the equations of the contact plane that better fits the 3D contour, (19), of the region (see Figure 4).

Replacing the vertexes of the contour given by (19) in the equations of the contact plane.

$$\left[\begin{array}{cc} \mathbf{r}_{c,1}^{\mathrm{T}} & 1 \\ \mathbf{r}_{c,2}^{\mathrm{T}} & 1 \\ \dots & \dots \\ \mathbf{r}_{c,n_{ic}}^{\mathrm{T}} & 1 \end{array} \right] \left[\begin{array}{c} \overline{\mathbf{n}} \\ d \end{array} \right] = \left[\begin{array}{c} 0 \\ 0 \\ \dots \\ 0 \end{array} \right] \Rightarrow \mathbf{A}\mathbf{x} = \mathbf{0} \tag{20}$$

Where $\overline{\mathbf{n}}$ is a vector normal to the contact plane and $d = -\mathbf{r}_c^{\mathrm{T}}\overline{\mathbf{n}}$ being $\mathbf{r}_c$ a point that belongs to the contact plane.

In general the system of equations (20) has the only solution $\overline{\mathbf{n}} = \mathbf{0}; d = 0$, which obviously is not the desired solution. It is necessary to impose the condition $|\overline{\mathbf{n}}| = 1$ to obtain an incompatible system of equations that can be solved by least squares.

Writing the least squares system from (20).

$$\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)\mathbf{x} = \mathbf{0} \tag{21}$$

Factoring the matrix $\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)$ and imposing the value (for example equal to 1) of the component of $\mathbf{x}$ corresponding to the minimum pivot of the factorization, the equations of the contact plane are obtained. Finally $\mathbf{x}$ has to be scaled to fulfill the condition $|\overline{\mathbf{n}}| = 1$, obtaining the final equations of the contact plane.

$$\overline{\mathbf{n}}^{\mathrm{T}}\mathbf{r} + d = 0 \tag{22}$$

Where $\overline{\mathbf{n}}$ is the unit normal vector to the plane and *d* is the distance from the plane to the origin measured along the normal vector.

All the calculations described in this section were performed with the contour of equation (19) expressed in the local reference frame of body *j*. The normal vector transformed to the global reference frame is obtained by means of the rotation matrix of body *j*.

$$\mathbf{n} = \mathbf{R}^j\overline{\mathbf{n}} \tag{23}$$

4.5 Contact region centroid algorithm

For each one of the contact regions identified in section 4.3, the algorithm described in this section, calculates the centroid of the projection of the contact region into the contact plane.

The centroid of a general 2D polygon of *N* vertexes, contained in the *X*-*Y* plane has the following expression.

$$\begin{aligned} \overline{\mathbf{r}}_c^{\triangle} &= \frac{1}{6A}\sum_{i=0}^{N-1}\left[\begin{array}{c} (x_i + x_{i+1})\,(x_i y_{i+1} - x_{i+1} y_i) \\ (y_i + y_{i+1})\,(x_i y_{i+1} - x_{i+1} y_i) \\ 0 \end{array} \right] \\ A &= \frac{1}{2}\sum_{i=0}^{N-1}(x_i y_{i+1} - x_{i+1} y_i) \end{aligned} \tag{24}$$

Nevertheless the contour, *c*, of equation (19) does not constitute a 2D polygon since its vertexes do not belong, in general, to the same plane. To assimilate the contour to a 2D polygon, the vertexes can be projected into the contact plane calculated in section 4.4. Moreover, the resulting 2D polygon has to be contained in the *X*-*Y* plane, what can be achieved by means of the transformation matrix $\mathbf{M}_t$, which transforms the *X*-*Y* plane into the contact plane of equation (22).

The mentioned transformation matrix has the following expression.

$$\begin{aligned} \mathbf{M}_t &= \left[\begin{array}{cc} \mathbf{M}_r & \overline{\mathbf{r}}_0 \\ 0 & 1 \end{array} \right] \\ \mathbf{M}_r &= \left[\begin{array}{ccc} \overline{\mathbf{v}}_0 & \overline{\mathbf{n}} \wedge \overline{\mathbf{v}}_0 & \overline{\mathbf{n}} \end{array} \right] \end{aligned} \tag{25}$$

Where $\overline{\mathbf{r}}_0$ and $\overline{\mathbf{v}}_0$ are a point and a vector contained in the contact plane, respectively, that can be chosen like follows.

$$\overline{\mathbf{r}}_0 = \begin{bmatrix} -d/n_x \\ 0 \\ 0 \end{bmatrix}; \overline{\mathbf{v}}_0 = \begin{bmatrix} -n_y \\ n_x \\ 0 \end{bmatrix}; \quad n_x = \max(n_x, n_y, n_z)$$
$$\overline{\mathbf{r}}_0 = \begin{bmatrix} 0 \\ -d/n_y \\ 0 \end{bmatrix}; \overline{\mathbf{v}}_0 = \begin{bmatrix} n_y \\ -n_x \\ 0 \end{bmatrix}; \quad n_y = \max(n_x, n_y, n_z)$$
$$\overline{\mathbf{r}}_0 = \begin{bmatrix} 0 \\ 0 \\ -d/n_z \end{bmatrix}; \overline{\mathbf{v}}_0 = \begin{bmatrix} n_z \\ 0 \\ -n_x \end{bmatrix}; \quad n_z = \max(n_x, n_y, n_z) \tag{26}$$

Being n_x, n_y, n_z the components of $\overline{\mathbf{n}}$. Expressing the contour in the local frame of the plane.

$$\overline{\mathbf{c}}_c^{ij,\triangle} = \mathbf{M}_r^{\mathrm{T}} \left(\overline{\mathbf{c}}_c^{ij,j} - \begin{bmatrix} \overline{\mathbf{r}}_0 & \overline{\mathbf{r}}_0 & \dots & \overline{\mathbf{r}}_0 \end{bmatrix} \right) \tag{27}$$

Replacing the *x* and *y* components of (27) in (24), the centroid $\overline{\mathbf{r}}_c^{\triangle}$, in local coordinates of the plane, is obtained.

Finally, the centroid expressed in global coordinates has the following expression.

$$\mathbf{r}_c^{ij} = \mathbf{s}_0^j + \mathbf{R}^j \left(\overline{\mathbf{r}}_0 + \mathbf{M}_r \overline{\mathbf{r}}_c^{\triangle} \right) \tag{28}$$

4.6 Maximum indentation calculation algorithm

For each one of the contact regions identified in section 4.3, the algorithm calculates the maximum indentation (or inter-penetration), δ. This algorithm travels along the colliding triangles (equation (12)) and their neighbors looking for the maximum indentation. In order to distinguish the faces of the first body that are inter-penetrating the second, it is necessary to check, for each neighbor not belonging to the colliding triangles list, two conditions: 1) the distance of each vertex of the triangle to the contact plane, calculated in section 4.4, is negative and 2) the projection each one of the vertex of the triangle into the contact plane lies inside the projected contact region contour of section 4.3 (see Figure 4). Between the lists of colliding and internal triangles of each body, the algorithm looks for the maximum indentation.

The checking of the condition 1) is straightforward while the condition 2) is checked by means of a ray casting algorithm: the number of intersections of a ray departing from a point is an even number if the point is outside the polygon, and it is odd if the point is inside the polygon.

5 CONTACT DETECTION ALGORITHMS II: CHAIN-HULL DETECTION ALGORITHM

The algorithms described in this section perform the detection of the contacts existing between pairs of bodies when one of them is a chain link and the other one has an arbitrary shape. Moreover, for each one of the contacts found, they perform the calculation of the necessary data to feed the contact model described in section 3. Here the geometry of the link is known and the geometry of the arbitrary body is given in triangular mesh format and read from a CAD file like in section 4.

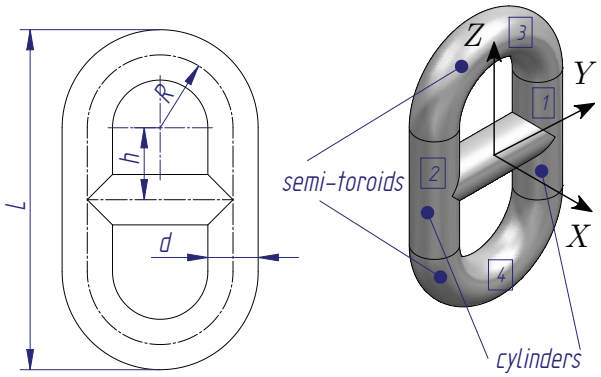

Figure 5. Chain link: approximative geometries.

The geometry of each link is composed of primitives: two finite cylinders (quadric surfaces limited by two planes) and two semi-toroids (superquadric surfaces limited by one plane), as shown in Figure 5. The algorithms take advantage of the analytical expressions to make the detection faster and more precise.

The bodies with arbitrary geometries are preprocessed like in section 4.1 and the calculation of the collision pairs is quite similar to 4.2 with the different that the triangle-triangle overlap algorithm of 4.2.2 has to be replaced by a link-triangle overlap algorithm that will be described next. As before, to avoid unnecessary calculations, each link and each primitive inside the link have a box surrounding them.

5.1 Link-Triangle overlap and maximum indentation algorithms

As indicated in Figure 5, the geometry of the link is divided into four primitives: two cylinders (primitives 1 and 2) and two torus (primitives 3 and 4) limited by planes. The stud in the center of the link, if exists, it is not considered for overlap calculations. The algorithm calculates if the triangle overlaps each one of the primitives, therefore is divided in two different overlap tests: cylinder-triangle overlap algorithm and torus-triangle overlap algorithm.

The way to calculate if a triangle intersects a primitive is done calculating the minimum distance between them. If the distance is negative, the triangle intersects the primitive with an indentation equal to the minimum distance.

5.1.1 Cylinder-Triangle overlap algorithm

Let's suppose the overlap test for the triangle p^i, given by equations (16) and one of the cylinders (primitives 1 and 2). The coordinates of the triangle are transformed to the local frame of

the cylinder using (14).

The minimum distance between the triangle and the cylinder is given by the following constrained optimization problem with inequality constraints.

$$\begin{aligned} \text{min.} \quad & r_p^2 = p_x^2 + (p_y \mp R)^2 && (29)\\ st. \quad & \\ \phi_1 = \quad & 1 - \frac{\mu_1}{l_1} - \frac{\mu_2}{l_2} \geq 0 && (30)\\ \phi_2 = \quad & h - p_z \geq 0 && (31)\\ \phi_3 = \quad & h + p_z \geq 0 && (32)\\ \phi_4 = \quad & \mu_1 \geq 0 && (33)\\ \phi_5 = \quad & \mu_2 \geq 0 && (34) \end{aligned}$$

From (16)

$$\begin{aligned} p_x &= p_{0x} + \mu_1 u_x + \mu_2 v_x && (35)\\ p_y &= p_{0y} + \mu_1 u_y + \mu_2 v_y && (36)\\ p_z &= p_{0z} + \mu_1 u_z + \mu_2 v_z && (37) \end{aligned}$$

Equation (29) is the distance between a point of the plane and the cylinder generatrix, in (29) the "$-$" sign is for the primitive 1 and the "$+$" sign for the primitive 2; equations (31) and (32) are the boundaries of the cylinder while (30) (33) and (34) are the boundaries of the triangle. Note that the unknowns of the problem are $\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$.

The Lagrangian of this problem is like follows.

$$L(\boldsymbol{\mu}, \boldsymbol{\lambda}) = r_p^2 - \lambda_1\phi_1 - \lambda_2\phi_2 - \lambda_3\phi_3 - \lambda_4\phi_4 - \lambda_5\phi_5 = r_p^2 - \boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{\lambda} \quad (38)$$

The Karush-Kuhn-Tucker conditions provide the necessary conditions for the optimum.

$$\begin{aligned} \boldsymbol{\nabla}_{\boldsymbol{\mu}} L(\boldsymbol{\mu}, \boldsymbol{\lambda}) &= 0 && (39)\\ \boldsymbol{\nabla}_{\boldsymbol{\lambda}} L(\boldsymbol{\mu}, \boldsymbol{\lambda}) &\geq 0 && (40)\\ \lambda_i \geq 0; \quad & i = 1, ..., 5 && (41)\\ \lambda_i \phi_i = 0; \quad & i = 1, ..., 5 && (42) \end{aligned}$$

The previous conditions are not easy to manage directly but they allow to transform the problem to several problems with equality constraints, instead of the inequality ones. Equations (42) say that the slack constraints have null multipliers and therefore they may be removed, solving only for the active constraints. After solving is necessary to check the conditions (40) and (41) to discover if it is necessary to activate or deactivate new constraints.

Since (29) is a quadratic function of the variables and the constraints are linear, the problem can be stated as a quadratic programming problem.

$$\left.\begin{array}{ll} \text{min.} & \frac{1}{2}\boldsymbol{\mu}^{\mathrm{T}}\mathbf{G}\boldsymbol{\mu} + \boldsymbol{\mu}^{\mathrm{T}}\mathbf{d} \\ st. & \mathbf{A}^*\boldsymbol{\mu} = \mathbf{b}^* \end{array}\right\} \Rightarrow$$

$$\begin{bmatrix} \mathbf{G} & -\mathbf{A}^{*\mathrm{T}} \\ \mathbf{A}^* & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\lambda}^* \end{bmatrix} = \begin{bmatrix} -\mathbf{d} \\ \mathbf{b}^* \end{bmatrix} \quad (43)$$

Where.

$$\mathbf{G} = \begin{bmatrix} 2\left(u_x^2 + u_y^2\right) & 2\left(u_x v_x + u_y v_y\right) \\ 2\left(u_x v_x + u_y v_y\right) & 2\left(v_x^2 + v_y^2\right) \end{bmatrix} \quad (44)$$

$$\mathbf{d} = \begin{bmatrix} 2p_{0x}u_x + 2\left(p_{0y} \mp R\right)u_y \\ 2p_{0x}v_x + 2\left(p_{0y} \mp R\right)v_y \end{bmatrix} \quad (45)$$

$$\mathbf{A} = \begin{bmatrix} -\frac{1}{l_1} & -\frac{1}{l_2} \\ -u_z & -v_z \\ u_z & v_z \\ 1 & 0 \\ 0 & 1 \end{bmatrix}; \ \mathbf{b} = \begin{bmatrix} -1 \\ -h + p_{0z} \\ -h - p_{0z} \\ 0 \\ 0 \end{bmatrix} \quad (46)$$

Note that $\mathbf{A}$ and $\mathbf{b}$ are the complete sets of constraints, while $\mathbf{A}^*$ and $\mathbf{b}^*$ are only the active sets. It was said before that after solving each problem with the current active set, it is necessary to check the validity of the set, adding or removing constraints if necessary. For this purpose active-set methods were employed (see, for example [13]).

The solution to the problem is the vector $\boldsymbol{\mu}$, which determines the point inside the triangle at the minimum distance to the cylinder. The distance is calculated like follows.

$$\delta = r_p - \frac{d}{2} \quad (47)$$

5.1.2 Toroid-Triangle overlap algorithm

Let's suppose the overlap test for the same triangle p^i, of section 5.1.1, and one of the semi-toroids (primitives 3 and 4) of Figure 5. The minimum distance between the triangle and the semi-toroid is given by the following constrained optimization problem with inequality constraints.

$$\begin{aligned} \text{min.} \quad & r_p^2 = p_x^2 + \left(\sqrt{p_y^2 + (p_z \mp h)^2} - R\right)^2 && (48)\\ st. \quad & \\ \phi_1 = \quad & 1 - \frac{\mu_1}{l_1} - \frac{\mu_2}{l_2} \geq 0 && (49)\\ \phi_2 = \quad & \pm(p_z \mp h) \geq 0 && (50)\\ \phi_4 = \quad & \mu_1 \geq 0 && (51)\\ \phi_5 = \quad & \mu_2 \geq 0 && (52) \end{aligned}$$

Equation (48) is the distance between a point of the triangle and the toroid generatrix, equation (31) is the boundary of the semi-toroid while (30) (33) and (34) are the boundaries of the triangle. The unknowns of the problem are again $\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$. In equations (48) and (50), when $\pm$ or $\mp$ appear, the upper sign is for the upper semi-toroid (primitive 3) and the lower sign for the lower semi-toroid (primitive 4).

The problem here is very similar to the cylinder overlap, with the difference that the objective function is not quadratic

anymore. The technique is the same, to transform the inequality constraints to equality constraints and to use the active set method (see [13]).

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}} L\left(\boldsymbol{\mu}, \boldsymbol{\lambda}\right) = \boldsymbol{\nabla}_{\boldsymbol{\mu}}\left(r_p^2\right) - \boldsymbol{\phi}_{\boldsymbol{\mu}}^{\mathrm{T}} \boldsymbol{\lambda} = \mathbf{0} \tag{53}$$

$$\boldsymbol{\phi}^* = \mathbf{0} \tag{54}$$

The problem can be solved using the Augmented Lagrangian technique.

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}}\left(r_p^2\right) - \boldsymbol{\phi}_{\boldsymbol{\mu}}^{\mathrm{T}}\left(\boldsymbol{\lambda} - \alpha\boldsymbol{\phi}\right) = \mathbf{0} \tag{55}$$

Since $\boldsymbol{\nabla}_{\boldsymbol{\mu}}\left(r_p^2\right)$ is non-linear, equation (55) can be solved by means of the Newton-Raphson iteration.

$$\left[\boldsymbol{\nabla}_{\boldsymbol{\mu}\boldsymbol{\mu}}\left(r_p^2\right) + \boldsymbol{\phi}_{\boldsymbol{\mu}}^{\mathrm{T}} \alpha \boldsymbol{\phi}_{\boldsymbol{\mu}}\right]_j \Delta\boldsymbol{\mu}_{j+1} = -\left[\boldsymbol{\nabla}_{\boldsymbol{\mu}}\left(r_p^2\right) - \boldsymbol{\phi}_{\boldsymbol{\mu}}^{\mathrm{T}}\left(\boldsymbol{\lambda}_i - \alpha\boldsymbol{\phi}\right)\right]_j \tag{56}$$

The system has to be solved iteratively keeping constant the values of $\boldsymbol{\lambda}_i$ until convergence. Once the convergence is attained, the outer iteration for the multipliers is performed.

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i - \alpha\boldsymbol{\phi} \tag{57}$$

Taking derivatives to the equations of the objective function and constraints.

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}}\left(r_p^2\right) = \left[\frac{\partial r_p^2}{\partial \mathbf{p}} \frac{\partial \mathbf{p}}{\partial \boldsymbol{\mu}}\right]^{\mathrm{T}} = \begin{bmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \end{bmatrix} \begin{bmatrix} 2p_x \\ 2k_1 p_y \\ 2k_1\left(p_z - h\right) \end{bmatrix} = \frac{\partial \mathbf{p}}{\partial \boldsymbol{\mu}}^{\mathrm{T}} \frac{\partial r_p^2}{\partial \mathbf{p}}^{\mathrm{T}} \tag{58}$$

$$k_1 = 1 - \frac{R}{\sqrt{p_y^2 + \left(p_z \mp h\right)}} \tag{59}$$

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}\boldsymbol{\mu}}\left(r_p^2\right) = \left[\frac{\partial^2 r_p^2}{\partial \mathbf{p} \partial \boldsymbol{\mu}} \frac{\partial \mathbf{p}}{\partial \boldsymbol{\mu}} + \frac{\partial r_p^2}{\partial \mathbf{p}} \cancel{\frac{\partial^2 \mathbf{p}}{\partial \boldsymbol{\mu}^2}}\right]^{\mathrm{T}} = \begin{bmatrix} \frac{\partial \mathbf{p}}{\partial \boldsymbol{\mu}}^{\mathrm{T}} \frac{\partial^2 r_p^2}{\partial \mathbf{p} \partial \mu_1}^{\mathrm{T}} & \frac{\partial \mathbf{p}}{\partial \boldsymbol{\mu}}^{\mathrm{T}} \frac{\partial^2 r_p^2}{\partial \mathbf{p} \partial \mu_2}^{\mathrm{T}} \end{bmatrix} \tag{60}$$

$$\frac{\partial^2 r_p^2}{\partial \mathbf{p} \partial \mu_1}^{\mathrm{T}} = 2\begin{bmatrix} u_x \\ R\left(\dfrac{p_y u_y + \left(p_z \mp h\right) u_z}{k_2^3}\right) p_y + k_1 u_y \\ R\left(\dfrac{p_y u_y + \left(p_z \mp h\right) u_z}{k_2^3}\right)\left(p_z \mp h\right) + k_1 u_z \end{bmatrix} \tag{61}$$

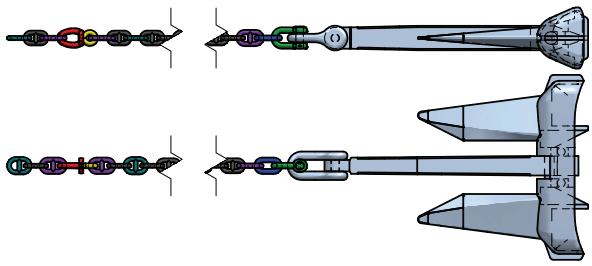

Figure 6. Anchor with chain

$$\frac{\partial^2 r_p^2}{\partial \mathbf{p} \partial \mu_2}^{\mathrm{T}} = 2\begin{bmatrix} v_x \\ R\left(\dfrac{p_y v_y + \left(p_z \mp h\right) v_z}{k_2^3}\right) p_y + k_1 v_y \\ R\left(\dfrac{p_y v_y + \left(p_z \mp h\right) v_z}{k_2^3}\right)\left(p_z \mp h\right) + k_1 v_z \end{bmatrix} \tag{62}$$

$$k_2 = \sqrt{p_y^2 + \left(p_z \mp h\right)} \tag{63}$$

$$\boldsymbol{\phi}_{\boldsymbol{\mu}} = \begin{bmatrix} \dfrac{-1}{l_1} & \dfrac{-1}{l_2} \\ u_z & v_z \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{64}$$

Note that $\boldsymbol{\phi}$ and $\boldsymbol{\phi}_{\boldsymbol{\mu}}$ are the complete sets of constraints and jacobian matrix, while $\boldsymbol{\phi}^*$ and $\boldsymbol{\phi}_{\boldsymbol{\mu}}^*$ are only the active sets.

The solution to the problem is the vector $\boldsymbol{\mu}$, which determines the point inside the triangle at the minimum distance to the semi-toroid. The distance is calculated using (47).

6 NUMERICAL SIMULATION: ANCHOR MANEUVER OF A WARSHIP

The system is shown in Figure 6, it is composed of the chain and the anchor. The chain has a large number of links of different types, parametrized in terms of the diameter d; the anchor is composed of two bodies linked with a revolute joint. The contacts between the different links of the chain is considered by means of universal joints with friction. The hull of the warship is static. The system is subjected to the tension force on the chain, the gravity forces, the constraint forces and the contact forces with the hull of the warship. The contacts between anchor and hull use the general algorithms described in section 4, while the contacts between links and hull use the specific algorithms of section 5, except by some kind of special links with shapes that cannot be approximated by cylindrical and toroidal surfaces and use the general algorithm also.

A graphical user interface was designed to control the simulation. The user can select parameters like: total simulation

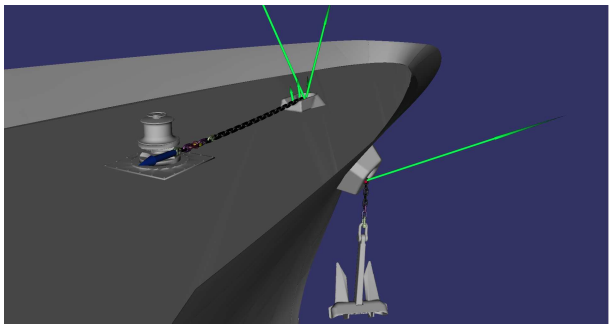

Figure 7. Simulation of the maneuver of a ship.

time, CAD models for hull and anchor, number and type of links, maximum tension force in the chain, etc.

The results of the simulations are shown in Figure 7.

7 Conclusions

All the algorithms necessary to simulate the anchor maneuver of a ship were developed. The description of the algorithm includes: a contact force model composed of normal and tangential forces; a general contact detection algorithm for bodies with complex non-conforming 3D geometries given by CAD models in triangular mesh format and a specific contact algorithms for the chain links. Furthermore, the performance of the described algorithms was tested with that realistic simulation.

ACKNOWLEDGMENT

The authors want to acknowledge the support of the Spanish company Navantia in this research.

REFERENCES

[1] Al Bedah A.a and Uicker J.J.b . Contact prediction between moving objects bounded by curved surfaces. *Journal of Computing and Information Science in Engineering*, 12(1), 2012. ISSN 15309827. doi: 10.1115/1.4005453.

[2] Choi Juhwan ; Ryu Han Sik ; Kim Chang Wan ; and Choi Jin Hwan . An efficient and robust contact algorithm for a compliant contact force model between bodies of complex geometry. *Multibody System Dynamics*, 23(1):99–120, JAN 2010. doi: 10.1007/s11044-009-9173-3.

[3] Cuadrado J ; Gutierrez R ; Naya MA ; and Morer P . A comparison in terms of accuracy and efficiency between a mbs dynamic formulation with stress analysis and a nonlinear fea code. *International Journal for Numerical Methods in Engineering*, 51(9):1033–1052, 07 2001.

[4] Cuadrado J ; Gutierrez R ; Naya MA ; and Gonzalez M . Experimental validation of a flexible mbs dynamic formulation through comparison between measured and calculated stresses on a prototype car. *Multibody System Dynamics*, 11(2):147–166, 03 2004.

[5] Dopico D. ; Luaces A. ; Gonzalez M. ; and Cuadrado J. . Dealing with multiple contacts in a human-in-the-loop application. *Multibody System Dynamics*, 25(2):167–183, 2011. ISSN 13845640. doi: 10.1007/s11044-010-9230-y.

[6] Flores P ; Ambrosio J ; Claro JCP ; and Lankarani HM . *Kinematics and Dynamics of Multibody Systems with Imperfect Joints*. Springer-Verlag, 2008. ISBN 978-3-540-74359-0.

[7] Flores Paulo ; Machado Margarida ; Silva Miguel ; and Martins Jorge . On the continuous contact force models for soft materials in multibody dynamics. *Multibody System Dynamics*, 25:357–375, 2011. ISSN 1384-5640.

[8] Jaln Garca de J and Bayo E . *Kinematic and dynamic simulation of multibody systems: The real-time challenge*. Springer-Verlag, New York (USA), 1994. ISBN 0-387-94096-0 / 3-540-94096-0.

[9] Gonthier Y ; McPhee J ; Lange C ; and Piedb?uf J-C . A regularized contact model with asymmetric damping and dwell-time dependent friction. *Multibody System Dynamics*, 11(3):209–233, 2004.

[10] Gonthier Yves ; Lange Christian ; and McPhee John . A volumetric contact model implemented using polynomial geometry. In *ECCOMAS Thematic Conference Multibody Dynamics 2009*, number CD, paper 230, pages 144–145, Warsaw, Poland, 06 2009.

[11] Lankarani HM and Nikravesh PE . Contact force model with hysteresis damping for impact analysis of multibody systems. *Journal of Mechanical Design*, 112(3):369–376, 1990.

[12] Lopes Daniel ; Silva Miguel ; Ambrosio Jorge ; and Flores Paulo . A mathematical framework for rigid contact detection between quadric and superquadric surfaces. *Multibody System Dynamics*, 24:255–280, 2010. ISSN 1384-5640.

[13] Nocedal J and Wright SJ . *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, 1999. ISBN 0-387-98793-2.