Sliding contact between freeform surfaces

A. Tasora, P. Righettini

Politecnico di Milano, Dipartimento di Ingegneria Strutturale - STM
P.zza L. da Vinci 32, 20133, Milano, Italy
tasora@mech.polimi.it righettini@mech.polimi.it

Abstract

This paper deals with the sliding-contact constraint equations describing the relative motion of two freeform surfaces, assuming that the surfaces can have arbitrary curvature in 3D space. The sliding-contact equations are developed either for the non-penetration condition and for the surface-tangency condition. Both are differentiated twice in time in order to allow a straightforward application to dynamic and kinematic multibody simulation within the context of an augmented lagrangian approach . This formulation represents the contact constraint by means of a *sliding tangent plane*, hence exploiting the advantageous optimizations of the so called *lock formulation*.

1 Introduction

Modern multibody software may be requested to perform simulations involving the contact between arbitrarly-shaped surfaces: these are the higher-pair joints which are extensively used in applied mechanics. Cam-follower mechanisms are notable examples of such joints, where the contact doesn't happen between cylindrical or prismatic surfaces, as in lower-pair joints, but happens instead along a line or a point.

Several methods for lower-pair joints (cylindrical joints, revolute joints, etc.) have been proposed and studied in multibody dynamics literature, but not so many methods have been discussed for a general-purpose approach to the problem of the contact between freeform surfaces.

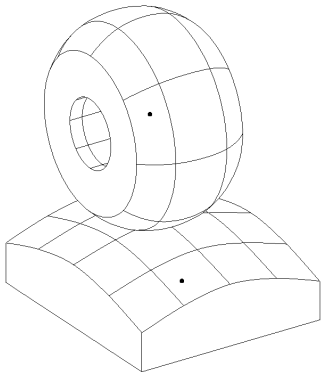

Figure 1: *Example of contact between free-form rigid bodies.*

Unfortunately, the analytical description of the kinematics of higher-pair joints may require complex formalisms, expecially whenever the contact happens between two freeform surfaces in

three-dimensional space. This may be the case of the contact between wheel and railway, or in spatial cams.

The kinematics of two rigid bodies subject to sliding contact is complicated by the fact that the curvature of their surfaces is liable of mutual accelerations; moreover these curvatures could be non uniform as in the example of cams.

This problem has been already investigated by some researchers, for example [2] recently suggested a method which fits well into whatever multibody formalism which is based on joint-coordinates: in that paper the contact point becomes a joint of the kinematical chain of rigid bodies, and its coordinates are the four *u,v* parameters of the two surfaces.

Instead, our method is rather targeted at multibody software based on the lagrangian approach, where the constraints are added by means of lagrange multipliers, and the coordinates of the equations are the natural coordinates of the rigid bodies. This, of course, implies that a fast and efficient formalism must be defined in order to compute the contact constraint equations as well as their derivatives and their jacobians (which will be used to solve the DAE differential-algebraic system, as explained in Shabana [14]).

Given that a fast and efficient way to handle the "point on a flat plane" basic constraint has already been developed and tested within the framework of the *lock formulation* [1], we managed to extend its capabilities to the case of contact between surfaces.

In fact, we can represent the contact constraint by introducing an auxiliary tangent plane which moves between the two bodies as they slide. If one manages to compute the exact position, alignment speed etc. of the tangent plane as function of body states during the motion, the sliding-contact constraint can be expressed by means of a simple "point on a flat plane" constraint, where the flat reference plane belongs to one of the two bodies, and the point belongs to the other body. Note that both the point and the plane must have specific relative motions respect to their bodies, because the plane must stay tangent to the shifting contact point, and these kinematical contributions can be easily applied to the "point on plane" constraint as described in the *lock formulation* (where arbitrary speed/accelerations can be freely imposed to the references used to describe the plane and the sliding point).

2 The sliding plane approach

Lets consider two indeformable rigid bodies being in contact, under the simplificative hypothesis of existence and non-singularity of the point of contact (multi-point contacts and degenerate situations of the type surface-surface or line-surface are not taken into consideration)

Note that, for the moment, the hypothesis of unilateral constraint is not imperative, therefore we will deal with bilateral contact for sake of semplicity -here we won't discuss the problems of non-smooth dynamics and impacts, which are investigated for example in [8] or [9]-.

It can be shown that the contact is geometrically correct if two conditions are satisfied at once: the two surfaces must have a point in common, and the tangent planes in that point must be the same.

Say $\vec{P}_{Po1-W,W}$ is the point of contact on surface O1 expressed in absolute coordinate system (W), and $\vec{S}_{So2-W,W}$ is the point of contact on surface O2 expressed in absolute coordinate system (W). These vectors will be later referenced as $\vec{P}_{o1}$ and $\vec{S}_{o2}$ for a more compact notation.

The first constraint equation implies that $\vec{P}_{o1}$ and $\vec{S}_{o2}$ must coincide in space, that is:

$$\vec{C}_{ps} = \vec{P}_{o1} - \vec{S}_{o2} = \vec{0}. \tag{1}$$

Now, say $\vec{n}_{no1-W,W}$ is the unit-lenght normal to the surface O1 at the point of contact $\vec{P}_{o1}$, and $\vec{n}_{no2-W,W}$ is the unit-lenght normal to the surface o2 on the point of contact $\vec{S}_{o2}$. These vectors will be later referenced as $\vec{n}_{o1}$ and $\vec{n}_{o2}$.

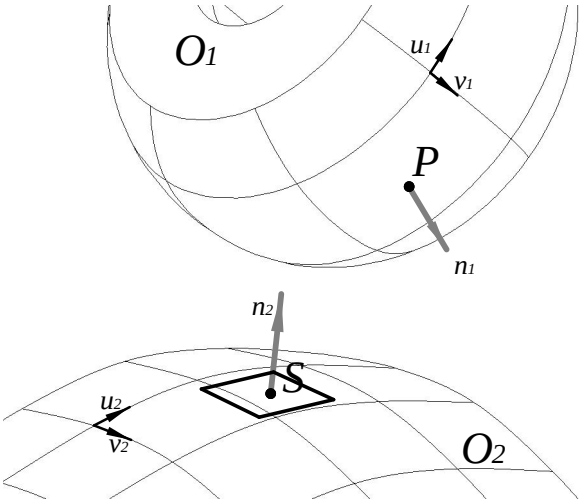

Figure 2: *The bodies are taken apart to show the contact point on body O1 and the sliding plane on body O2. Contact happens for coincident P and S, and for aligned normals* $\vec{n}_{o2}$ *and* $\vec{n}_{o1}$.

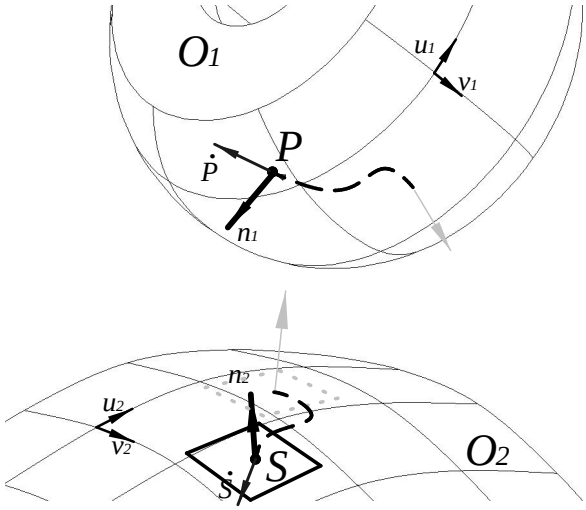

Figure 3: *Exploded view. The contact point* $\vec{P}_{o1}$ *and the sliding plane must move on their surfaces during relative body motion, so that* $\vec{P}_{o1}$ *and* $\vec{S}_{o2}$ *share the same speed and position.*

This lead us to the second constraint equation, which requires that the two surfaces must be tangent at the contact point, hence the normals must be aligned:

$$\vec{C}_n = \vec{n}_{o1} + \vec{n}_{o2} = \vec{0}. \tag{2}$$

We assume that the position of point $\vec{P}_{o1}$ on surface of body O1 can be expressed (at least, locally) as a function of two curvilinear coordinates u_{o1}, v_{o1}, and the same for point $\vec{P}_{o1}$, whose position on surface can be a function of two curvilinear coordinates u_{o2}, v_{o2}. Hence, satisfying the equations 1 and 2 implies a system of nonlinear equations

$$\vec{C}_{ps,n} = \vec{C}\left(q_{o1}, q_{o2}, u_{o1}, v_{o1}, u_{o2}, v_{o2}, t)\right) = \vec{0}. \tag{3}$$

which must be solved either for the positions of the two bodies (the coordinates q_{o1}, q_{o2}), either for the auxiliary variables $u_{o1}, v_{o1}, u_{o2}, v_{o2}$.

Note that one of the three scalar constraints of eq. 2 is redundant (because unit norm of normals is implied, $\parallel \vec{n}_{o1} \parallel = \parallel \vec{n}_{o1} \parallel = 1$), therefore the complete system of constraints eq.3 has $3+(3-1) = 5$ scalar equations. Meanwhile, four auxiliary variables $u_{o1}, v_{o1}, u_{o2}, v_{o2}$ were added: hence the contact effectively subtracts $5-4 = 1$ degree of freedom from the mechanical system, an intuitive result which is also confirmed by many authors dealing with classical mechanical problems [11].

The introduction of four auxiliary variables in the state vector our system, as well as the description of the contact by way of the 5-dimensional equation 3, of course adds unwanted complication into our multibody formalism and may have a negative impact on the performance of the simulation code.

Therefore one may want to reduce the system to a more handy formulation, where only a single scalar constraint equation is added, and the four auxiliary variables can be recovered afterward as dependent variables (i.e. only rigid body coordinates are introduced in state system, while $u_{o1}, v_{o1}, u_{o2}, v_{o2}$ variables -and their derivatives- are computed separately).

An effective way to accomplish this task may be represented by the *sliding plane* approach, which we discuss in this paper. Such method introduces a "point on plane" constraint between the two contacting bodies, which is responsible of reducing the degrees of freedom of the system

by one unit. During the multibody simulation, the position of the reference plane is continuously moved tangentially to the surface of a body (as well as the reference point continuously moves on the surface of the other), thus updating the auxiliary variables $u_{o1}, v_{o1}, u_{o2}, v_{o2}$ and their derivatives as dependent variables (fig.2,3).

Given that there's no need to add the variables $u_{o1}, v_{o1}, u_{o2}, v_{o2}$ in system's state vector, the solution of kinematic and dynamic problems is somewhat simple: it just requires the implementation of a holonomic constraint of the type "point on a plane", where the position/speed/acceleration of both the reference point and reference plane can be imposed. This is easily achieved, for example, through the *lock formulation* approach, formulated in [1] and briefly discussed in the next paragraph.

Furthermore, as a positive side effect of this approach, the orthogonal contact force is effortlessly recovered from the lagrangian multiplier of the "point on plane" constraint.

However, special attention must be paid to the problem of computing the rheonomic terms which are required by the *lock formulation*, as they will be responsible of the acceleration effects caused by surface curvature. In other words, one must know not only the position but also the speed of the contact point as a function of body states, in order to set proper values for position/speed/acceleration of both the reference point and reference plane. The paragraph "Kinematics of contact plane" will deal with this problem.

3 Basic point-plane constraint via "lock formulation"

The so called "lock formulation" relies heavily on quaternion algebra and offers a compact yet efficient way to implement the derivations of constraint equations, where most common holonomic and rheonomic constraints can be inherited from a single formalism. Moreover, the jacobians are obtained analytically, and this has a positive impact on the performance of multibody simulations based on the lagrangian approach.

Let consider two generic rigid bodies O1 ans O2, both with two auxiliary coordinate systems P and S (the so called "markers") whose body-relative positions and body-relative rotations can be constant or imposed via time-functions (fig. 4).

One can impose a translation constraint on the relative position of P respect to the coordinate system of S: this is the "positional" constraint. Also, one can impose a rotation constraint on the relative rotation of P respect to S, in the coordinate system of S, and this is the "rotational" constraint.

If needed, both the positional and the rotational constraints can be expressed with time-dependent functions, as well as the relative positions and relative alignments of markers respect to their rigid bodies.

The effect of these positional and rotational constraints between P and S is a kind of "glue" between the two bodies, hence the name *lock formulation*. This is expressed by 6 scalar equations. However, if one suppresses one or more scalar conditions, the constraint gets some degrees of freedom and turns into specific holonomic constraints.

For example, a spherical joint is obtained by suppressing all the three scalar components of the rotational constraints, and a cylindrical joint is obtained by suppressing -for example- the Z positional component and the Z rotational component. In the same way we can obtain lot of other holonomic constraints, for instance the prismatic guide, the point-on-line condition, the point-on-plane condition (used extensively in this article), the Cardano joint, the revolute joint, the parallelism condition, etc. Also, by setting adequate time-dependant functions in the rheonomic terms of the equations, one can get whatever kind of actuators, motors, motion laws, imposed trajectories, etc.

A simplified and compact version of the "lock formulation" is described below, as a quick reference, but advanced details and implementation issues are described extensively in [1].

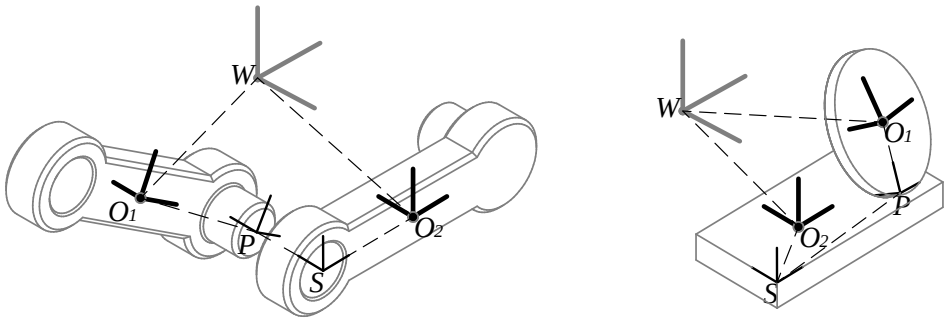

Figure 4: *Reference frames which are used to build constraints with the "lock formulation" method (two examples)*

Let's introduce the following notation:

- $\vec{q_x}_{o1}$, $\vec{q_x}_{o2}$, $\vec{q_\theta}_{o1}$, $\vec{q_\theta}_{o2}$, are the position-coordinates and rotation-coordinates of bodies O1 and O2, where rotations are expressed as unit quaternions $\vec{q_\theta}$,
- $\vec{q_x}_P$, $\vec{q_x}_S$, $\vec{q_\theta}_P$, $\vec{q_\theta}_S$ are the coordinates (positions and rotations) of markers P and S respect to their bodies, O1 and O2, and may be user-imposed functions of time.
- $[\Lambda_r] = [\Lambda_r(\vec{q}_{\theta_r})]$ is a generic rotation matrix, function of a quaternion $\vec{q_\theta}$,
- $[Gl_{o1}]$ is a 3x4 rectangular matrix, function of the quaternion $\vec{q}_{\theta_{o1}}$, so that $[Gl_{o1}]\dot{\vec{q}}_{\theta_{o1}} = \omega_{o1}$, as described in [14]
- $[\tilde{a}]$ is a skew symmetric matrix such that $[\tilde{a}]\vec{b} = \vec{a} \wedge \vec{b}$
- $\vec{q_x}_\Delta = \vec{f}(t)$ and $\vec{q_\theta}_\Delta = \vec{f}(t)$ are the imposed translation and rotation between P and S, in coordinates of S.

Hence the positional constraint can be written as:

$$\vec{C}_x = \vec{q}_{x_{P-S,S}} - \vec{q}_{x_\Delta} = \vec{0} \tag{4}$$

$$\vec{C}_x = [\Lambda_S]^T [\Lambda_{o2}]^T \left((\vec{q}_{x_{o1}} + [\Lambda_{o1}]\vec{q}_{x_P}) (\vec{q}_{x_{o2}} + [\Lambda_{o2}]\vec{q}_{x_S}) \right) - \vec{q}_{x_\Delta} = \vec{0} \tag{5}$$

In order to obtain the jacobian matrix $[C_q]$ and the vector $[Q_c]$, which are extensively used in the equation $\ddot{\vec{C}}_x = [C_q]\ddot{\vec{q}} - \vec{Q} = \vec{0}$ of lagrangian dynamics, one can differentiate twice the eq. 5. Then, after some algebraic manipulations, the analytical expression for the jacobian of the *lock formulation* constraint (translational part) can be written in the following fashion:

$$[C_{q_x}] = \left[[C_{q_x}]_{xo1} \, [C_{q_x}]_{\theta o1} \, [C_{q_x}]_{xo2} \, [C_{q_x}]_{\theta o2} \right] \tag{6}$$

where:

$$\begin{aligned}
[C_{q_x}]_{xo1} &= +[\Lambda_S]^T [\Lambda_{o2}]^T \\
[C_{q_x}]_{\theta o1} &= -[\Lambda_S]^T [\Lambda_{o2}]^T [\Lambda_{o1}] [\tilde{\vec{q}_{x_P}}] [Gl_{o1}] \\
[C_{q_x}]_{xo2} &= -[\Lambda_S]^T [\Lambda_{o2}]^T \\
[C_{q_x}]_{\theta o2} &= +[\Lambda_S]^T [\Lambda_{o2}]^T [\Lambda_{o1}] [\tilde{\vec{q}_{x_P}}] [Gl_{o1}] \\
&\quad +[\Lambda_S]^T \left[[\Lambda_{o2}]^T \tilde{\vec{q}}_{x_{P-S,W}} \right] [Gl_{o2}]
\end{aligned} \tag{7}$$

Also, the $\vec{Q}_x$ vector can be expressed as:

$$\begin{aligned}\vec{Q}_x = \quad & +[\Lambda_S]^T[\Lambda_{o2}]^T\left([\Lambda_{o1}][\tilde{\omega}_{o1}][\tilde{\omega}_{o1}] + 2[\dot{\Lambda}_{o1}]\dot{\vec{q}}_{x_P}[\Lambda_{o1}]\ddot{\vec{q}}_{x_P}\right) + \\ & -[\Lambda_S]^T[\Lambda_{o2}]^T\left([\Lambda_{o2}][\tilde{\omega}_{o2}][\tilde{\omega}_{o2}] + 2[\dot{\Lambda}_{o2}]\dot{\vec{q}}_{x_S}[\Lambda_{o2}]\ddot{\vec{q}}_{x_S}\right) + \\ & +2[\dot{\Lambda}_S]^T[\dot{\Lambda}_{o2}]^T\vec{q}_{x_{P-S,W}} + 2[\dot{\Lambda}_S]^T[\Lambda_{o2}]^T\dot{\vec{q}}_{x_{P-S,W}} + 2[\Lambda_S]^T[\dot{\Lambda}_{o2}]^T\dot{\vec{q}}_{x_{P-S,W}} + \\ & +[\Lambda_S]^T\left[[\Lambda_{o2}][\tilde{\omega}_{o2}][\tilde{\omega}_{o2}]\right]^T\vec{q}_{x_{P-S,W}} + [\ddot{\Lambda}_S]^T[\Lambda_{o2}]^T\vec{q}_{x_{P-S,W}} - \ddot{\vec{q}}_{x_\Delta} \end{aligned} \tag{8}$$

On the other hand, the rotational constraint can be written in quaternion algebra as follows:

$$\vec{C}_\theta = \vec{q}_{\theta_\Delta}^{-1}\vec{q}_{\theta_{P-S,S}} - \vec{q}_{\theta_\Re} = \vec{0} \tag{9}$$

where we introduced the real quaternion $\vec{q}_{\theta_\Re} = [1,0,0,0]^T$. However we won't enter into details for the rotational constraint, because this part of the *lock formulation* isn't necessary for the sliding contact theory presented in this paper (interested readers can find the expressions for jacobian $[C_{q_\theta}]$ and vector Q_θ in [1]).

At this point, many holonomic and rheonomic constraints can be inherited from the formulation above, simply by suppressing some scalar constraints among the three equations for the traslation and/or the three equations for the rotation.

For example, the constraint "point on plane" where the plane moves about one of the two bodies following some prescribed position/speed/acceleration, can be easily recovered from eq. 4 where only the 3rd scalar equation is taken into consideration. Hence the reference P (whose motion on O1 surface can be set via $\vec{q}_{x_P}$, $\dot{\vec{q}}_{x_P}$, $\ddot{\vec{q}}_{x_P}$, $\vec{q}_{\theta_P}$, $\dot{\vec{q}}_{\theta_P}$, $\ddot{\vec{q}}_{\theta_P}$) cannot move orthogonally to the XY plane described by the reference S (whose motion on O2 surface can be set via the terms $\vec{q}_{x_S}$, $\dot{\vec{q}}_{x_S}$, $\ddot{\vec{q}}_{x_S}$, $\vec{q}_{\theta_S}$, $\dot{\vec{q}}_{\theta_S}$, $\ddot{\vec{q}}_{\theta_S}$).

4 Kinematics of contact plane

Since we have introduced the *lock formulation* (see equations 7 and 8) which allows the representation of a simple point-plane constraint, now we must find the position and speed of the references which define this kind of constraint in case the plane and the point are moving on curved surfaces.

4.1 Contact plane: the position problem

As seen before, the contact constraint is expressed by two vectorial equations, the former saying that the contact points on both the surfaces must have the same position in space, and the latter implying that the two surfaces must be tangent (that is, the normals must be aligned).

$$\vec{C}_{ps} = \vec{P}_{o1} - \vec{S}_{o2} = \vec{0}. \tag{10}$$

$$\vec{C}_n = \vec{n}_{o1} + \vec{n}_{o2} = \vec{0}. \tag{11}$$

The solution of the equations above for $\{u_{o1}, v_{o1}, u_{o2}, v_{o2}\}$ is an highly non-linear problem, and the solutions can be multiple or even infinite in degenerate situations (for example, if contact points are multiple, or along a line such as in the case of the contact between two parallel cylinders). In this paper we will focus our attention on the situation of a single contact. Also, we won't deal the details of the procedures which can be used to solve the non-linear problem of finding globally the contact point(s), because this topic is extensively discussed in

other scientific areas, for instance in computational geometry, computer graphics and modelling ([3], [4], and [5]).

We just remark that collision-detection is a very time-consuming process, and efficient procedures must be set up for this task, for example [4] one can pre-process a rough tesselated approximation of the NURBS surface, then he can perform polygon-polygon test very quickly, expecially if an hierarchical tree structure of bounding boxes has been computed off-line as an optimization.

We experienced that after the "global" rough approximation of the contact point has been obtained with such polygon-proximity test, a more precise "local" solution can be refined by using local non-linear programming methods, such as the gradient or Newton methods [10], which operate on the true -non tesselated- parametric surface.

Also, we further improved the efficiency of this geometric problem by performing the "global" contact point search with a less dense time sample than the faster "local" refinement. In fact the local position must be refined at each step of the simulation in order to get smooth and accurate results, but the position-jumps of the contact point (which are detected by the global method) may happen seldom. Think about a cam and a follower: after the initial contact point between cam and follower have been found once, the point may be updated during cam rotation just by using the local optimization method, whose convergency can be fast because the previous positions can be used as approximate guesses. Then, the global collision algorithm can be invoked less frequently, just to check if there are sudden jumps in contact points (for example, the cam has a dinch, or the follower touches another object, etc.) [1]

4.2 Contact plane: the speed problem

We can differentiate both equations 1 and 2 respect to time. The variables of equation 1 are the four surface parameters $\{u_{o1}, v_{o1}, u_{o2}, v_{o2}\}$ and the coordinates of the two rigid bodies $\{\vec{q}_{o1}, \vec{q}_{o2}\}$, therefore:

$$\dot{\vec{C}}_{ps} = \frac{\mathrm{d}}{\mathrm{d}t}\left[\vec{P}_{o1}(u_{o1}, v_{o1}, \vec{q}_{o1}) - \vec{S}_{o2}(u_{o2}, v_{o2}, \vec{q}_{o2})\right] = \vec{0}. \tag{13}$$

Applying the chain rule of differentiation:

$$\begin{aligned}\dot{\vec{C}}_{ps} = \quad & \left(\frac{\partial \vec{P}_{o1}}{\partial u_{o1}}\frac{\partial u_{o1}}{\partial t} + \frac{\partial \vec{P}_{o1}}{\partial v_{o1}}\frac{\partial v_{o1}}{\partial t} + \frac{\partial \vec{P}_{o1}}{\partial \vec{q}_{o1}}\frac{\partial \vec{q}_{o1}}{\partial t} + \frac{\partial \vec{P}_{o1}}{\partial t}\right) - \\ & \left(\frac{\partial \vec{S}_{o2}}{\partial u_{o2}}\frac{\partial u_{o2}}{\partial t} + \frac{\partial \vec{S}_{o2}}{\partial v_{o2}}\frac{\partial v_{o2}}{\partial t} + \frac{\partial \vec{S}_{o2}}{\partial \vec{q}_{o2}}\frac{\partial \vec{q}_{o2}}{\partial t} + \frac{\partial \vec{S}_{o2}}{\partial t}\right)\end{aligned} \tag{14}$$

The terms $\frac{\partial \vec{P}_{o1}}{\partial \vec{q}_{o1}}$ and $\frac{\partial \vec{S}_{o2}}{\partial \vec{q}_{o2}}$ are matrices, with 3 rows and 6 columns (given that each rigid body has 6 d.o.f.) and will be written $[P_{q_{o1}}]$ and $[S_{q_{o2}}]$ hereafter. Simplifying the null terms, we can write the more compact form:

[1] Note: in general, because of nonlinearities, the position problem is approached iteratively: here two operations are required for each iteration. The first stage involves a single step on Newton method for solving the typical position problem of inverse kinematics:

$$\begin{aligned}&\tfrac{\partial \vec{C}_n}{\partial \vec{q}}\Delta\vec{q}_i = [C_q]_i\,\Delta\vec{q}_i = -\vec{C}_i \\ &\vec{q}_{i+1} = \vec{q}_i + \Delta\vec{q}_i\end{aligned} \tag{12}$$

where the constraints $\vec{C}(\vec{q}, t)$ contain all joints of the mechanical system including the "point on plane" constraint, and the second operation consists in updating the the position of the point of contact, using the global/local optimization methods discussed above. The two stages are repeated until convergence -if any- is reached.

$$\dot{\vec{C}}_{ps} = \left(\frac{\partial \vec{P}_{o1}}{\partial u_{o1}}\dot{u}_{o1} + \frac{\partial \vec{P}_{o1}}{\partial v_{o1}}\dot{v}_{o1} + [P_{q_{o1}}]\dot{\vec{q}}_{o1}\right) - \left(\frac{\partial \vec{S}_{o2}}{\partial u_{o2}}\dot{u}_{o2} + \frac{\partial \vec{S}_{o2}}{\partial v_{o2}}\dot{v}_{o2} + [S_{q_{o2}}]\dot{\vec{q}}_{o2}\right) \tag{15}$$

Given that the state of the two bodies is known, the speeds $\dot{\vec{q}}_{o1}$ and $\dot{\vec{q}}_{o2}$ are known as well, so we can collecting the unknown terms $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$ in order to obtain the following system:

$$\left[\begin{array}{c|c|c|c} \frac{\partial \vec{P}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{P}_{o1}}{\partial v_{o1}} & -\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} & -\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \end{array}\right] \left\{\begin{array}{c} \dot{u}_{o1} \\ \dot{v}_{o1} \\ \dot{u}_{o2} \\ \dot{v}_{o2} \end{array}\right\} = -[P_{q_{o1}}]\dot{\vec{q}}_{o1} + [S_{q_{o2}}]\dot{\vec{q}}_{o2} \tag{16}$$

A quick glance at the system of eq.16, which has four unknowns and three scalar equations, tells that some other equations must be written to get rid of the indetermination and finally compute the parameters $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$. In fact now we must take into consideration the abovementioned constraint on surface tangency, eq.2, which can be differentiated as follows:

$$\dot{\vec{C}}_n = \frac{\mathrm{d}}{\mathrm{d}t}\left[\vec{n}_{o1}(u_{o1}, v_{o1}, \vec{q}_{o1}) + \vec{n}_{o2}(u_{o2}, v_{o2}, \vec{q}_{o2})\right] = \vec{0}. \tag{17}$$

that is, using the same algebraic manipulations and differentiation rules used for 16:

$$\begin{aligned} \dot{\vec{C}}_n = \quad & \left(\frac{\partial \vec{n}_{o1}}{\partial u_{o1}}\frac{\partial u_{o1}}{\partial t} + \frac{\partial \vec{n}_{o1}}{\partial v_{o1}}\frac{\partial v_{o1}}{\partial t} + \frac{\partial \vec{n}_{o1}}{\partial \vec{q}_{o1}}\frac{\partial \vec{q}_{o1}}{\partial t} + \frac{\partial \vec{n}_{o1}}{\partial t}\right) + \\ & \left(\frac{\partial \vec{n}_{o2}}{\partial u_{o2}}\frac{\partial u_{o2}}{\partial t} + \frac{\partial \vec{n}_{o2}}{\partial v_{o2}}\frac{\partial v_{o2}}{\partial t} + \frac{\partial \vec{n}_{o2}}{\partial \vec{q}_{o2}}\frac{\partial \vec{q}_{o2}}{\partial t} + \frac{\partial \vec{n}_{o2}}{\partial t}\right) \end{aligned} \tag{18}$$

$$\dot{\vec{C}}_n = \left(\frac{\partial \vec{n}_{o1}}{\partial u_{o1}}\dot{u}_{o1} + \frac{\partial \vec{n}_{o1}}{\partial v_{o1}}\dot{v}_{o1} + [n_{q_{o1}}]\dot{\vec{q}}_{o1}\right) + \left(\frac{\partial \vec{n}_{o2}}{\partial u_{o2}}\dot{u}_{o2} + \frac{\partial \vec{n}_{o2}}{\partial v_{o2}}\dot{v}_{o2} + [n_{q_{o2}}]\dot{\vec{q}}_{o2}\right) \tag{19}$$

The following system has three scalar equations and the same unknowns $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$ of system 16:

$$\left[\begin{array}{c|c|c|c} \frac{\partial \vec{n}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{n}_{o1}}{\partial v_{o1}} & \frac{\partial \vec{n}_{o2}}{\partial u_{o2}} & \frac{\partial \vec{n}_{o2}}{\partial v_{o2}} \end{array}\right] \left\{\begin{array}{c} \dot{u}_{o1} \\ \dot{v}_{o1} \\ \dot{u}_{o2} \\ \dot{v}_{o2} \end{array}\right\} = -[n_{q_{o1}}]\dot{\vec{q}}_{o1} - [n_{q_{o2}}]\dot{\vec{q}}_{o2} \tag{20}$$

We can put together the two systems 16 and 20 to get the following:

$$\left[\begin{array}{cccc} \frac{\partial \vec{P}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{P}_{o1}}{\partial v_{o1}} & -\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} & -\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \\ \frac{\partial \vec{n}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{n}_{o1}}{\partial v_{o1}} & \frac{\partial \vec{n}_{o2}}{\partial u_{o2}} & \frac{\partial \vec{n}_{o2}}{\partial v_{o2}} \end{array}\right] \left\{\begin{array}{c} \dot{u}_{o1} \\ \dot{v}_{o1} \\ \dot{u}_{o2} \\ \dot{v}_{o2} \end{array}\right\} = \left\{\begin{array}{c} -[P_{q_{o1}}]\dot{\vec{q}}_{o1} + [S_{q_{o2}}]\dot{\vec{q}}_{o2} \\ -[n_{q_{o1}}]\dot{\vec{q}}_{o1} - [n_{q_{o2}}]\dot{\vec{q}}_{o2} \end{array}\right\} \tag{21}$$

The system above has four unknowns and six scalar equations. However two equations are redundant, to be more precise one of the first three equations and one of the last three equations can be eliminated, to obtain a 4x4 system which can be readily solved for $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$.

Let's proof this. The tangent plane in P and S is the same for both the surfaces o1 and o2 because of equation 2, and we can build the rotation matrix $[\Lambda_n]$ which is aligned to such

tangent plane. We can rewrite the system equations in that space, that is, after a coordinate projection:

$$\begin{bmatrix} [\Lambda_n]^T & [0]^T \\ [0]^T & [\Lambda_n]^T \end{bmatrix} \begin{bmatrix} \frac{\partial \vec{P}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{P}_{o1}}{\partial v_{o1}} & -\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} & -\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \\ \frac{\partial \vec{n}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{n}_{o1}}{\partial v_{o1}} & \frac{\partial \vec{n}_{o2}}{\partial u_{o2}} & \frac{\partial \vec{n}_{o2}}{\partial v_{o2}} \end{bmatrix} \begin{Bmatrix} \dot{u}_{o1} \\ \dot{v}_{o1} \\ \dot{u}_{o2} \\ \dot{v}_{o2} \end{Bmatrix} = \\ \begin{bmatrix} [\Lambda_n]^T & [0]^T \\ [0]^T & [\Lambda_n]^T \end{bmatrix} \begin{Bmatrix} -[P_{q_{o1}}]\dot{\vec{q}}_{o1} + [S_{q_{o2}}]\dot{\vec{q}}_{o2} \\ -[n_{q_{o1}}]\dot{\vec{q}}_{o1} - [n_{q_{o2}}]\dot{\vec{q}}_{o2} \end{Bmatrix} \tag{22}$$

One can easily verify that, if $[\Lambda_n]$ has been built with the Z axis parallel to the surface normals (that is, if $[\Lambda_n]^T \vec{n}_{o1} = \{0, 0, 1\}^T$) then the third and sixth row of system 22 have null coefficients. In other words, either vectorspaces of eq. 13 and eq. 17 spawn the tangent spaces of $\vec{P}(u, v)$ and $\vec{S}(u, v)$ when eq. 1 and eq. 2 are satisfied and the states $\vec{q}_{o1}$, $\vec{q}_{o2}$, $\dot{\vec{q}}_{o1}$, $\dot{\vec{q}}_{o2}$ are coherent with the point-plane constraint equation (see later).

Therefore we can get rid of the 3rd and 6th row of system 22 simply by using the first two rows of the 3x3 rotation matrix $[\Lambda_n]^T$ when performing the projection in tangent coordinates. This means that we can introduce the 3x2 matrix $[\lambda_{uv}]$ which is like $[\Lambda_n]$ except it hasn't the 3rd column, which represents the orthogonal versor, i.e. the surface normal. This means that the two columns of $[\lambda_{uv}]$ are simply obtained as two generic orthogonal versors contained in the tangent plane. This lead to the following system in "surface tangent coordinates":

$$\begin{bmatrix} [\lambda_{uv}]^T & [0]^T \\ [0]^T & [\lambda_{uv}]^T \end{bmatrix} \begin{bmatrix} \frac{\partial \vec{P}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{P}_{o1}}{\partial v_{o1}} & -\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} & -\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \\ \frac{\partial \vec{n}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{n}_{o1}}{\partial v_{o1}} & \frac{\partial \vec{n}_{o2}}{\partial u_{o2}} & \frac{\partial \vec{n}_{o2}}{\partial v_{o2}} \end{bmatrix} \begin{Bmatrix} \dot{u}_{o1} \\ \dot{v}_{o1} \\ \dot{u}_{o2} \\ \dot{v}_{o2} \end{Bmatrix} = \\ \begin{bmatrix} [\lambda_{uv}]^T & [0]^T \\ [0]^T & [\lambda_{uv}]^T \end{bmatrix} \begin{Bmatrix} -[P_{q_{o1}}]\dot{\vec{q}}_{o1} + [S_{q_{o2}}]\dot{\vec{q}}_{o2} \\ -[n_{q_{o1}}]\dot{\vec{q}}_{o1} - [n_{q_{o2}}]\dot{\vec{q}}_{o2} \end{Bmatrix} \tag{23}$$

Then, the coefficient matrix of the system has 4 rows and 4 columns, and the straightforward Gauss solution scheme can be applied in order to obtain $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$ as desired.

4.3 Contact plane: the acceleration problem

Now we can perform a further differentiation of eq.13 and eq.17 in order to obtain the unknown accelerations of the points of contact on the two surfaces, expressed in parametric coordinates as $\{\ddot{u}_{o1}, \ddot{v}_{o1}, \ddot{u}_{o2}, \ddot{v}_{o2}\}$.

For eq. 13 we have:

$$\ddot{\vec{C}}_{ps} = \frac{\mathrm{d}}{\mathrm{d}t}\left[\left(\frac{\partial \vec{P}_{o1}}{\partial u_{o1}}\dot{u}_{o1} + \frac{\partial \vec{P}_{o1}}{\partial v_{o1}}\dot{v}_{o1} + [P_{q_{o1}}]\dot{\vec{q}}_{o1}\right) - \left(\frac{\partial \vec{S}_{o2}}{\partial u_{o2}}\dot{u}_{o2} + \frac{\partial \vec{S}_{o2}}{\partial v_{o2}}\dot{v}_{o2} + [S_{q_{o2}}]\dot{\vec{q}}_{o2}\right)\right] \tag{24}$$

$$\begin{aligned} \ddot{\vec{C}}_{ps} = \quad & \frac{\partial \vec{P}_{o1}}{\partial u_{o1}}\ddot{u}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial u_{o1}}\dot{u}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial v_{o1}}\dot{u}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial \vec{q}_{o1}}\dot{u}_{o1}\dot{\vec{q}}_{o1} \\ & + \frac{\partial \vec{P}_{o1}}{\partial v_{o1}}\ddot{v}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial v_{o1}\partial u_{o1}}\dot{v}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial v_{o1}\partial v_{o1}}\dot{v}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial v_{o1}\partial \vec{q}_{o1}}\dot{v}_{o1}\dot{\vec{q}}_{o1} \\ & + \frac{\partial \vec{P}_{o1}}{\partial \vec{q}_{o1}}\ddot{\vec{q}}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial \vec{q}_{o1}\partial u_{o1}}\dot{\vec{q}}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial \vec{q}_{o1}\partial v_{o1}}\dot{\vec{q}}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{P}_{o1}}{\partial \vec{q}_{o1}\partial \vec{q}_{o1}}\dot{\vec{q}}_{o1}\dot{\vec{q}}_{o1} \\ & - \frac{\partial \vec{S}_{o2}}{\partial u_{o2}}\ddot{u}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial u_{o2}}\dot{u}_{o2}\dot{u}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial v_{o2}}\dot{u}_{o2}\dot{v}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial \vec{q}_{o2}}\dot{u}_{o2}\dot{\vec{q}}_{o2} \end{aligned}$$

$$
\begin{aligned}
&-\frac{\partial \vec{S}_{o2}}{\partial v_{o2}}\ddot{v}_{o1} - \frac{\partial^2 \vec{S}_{o2}}{\partial v_{o2}\partial u_{o2}}\dot{v}_{o2}\dot{u}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial v_{o2}\partial v_{o2}}\dot{v}_{o2}\dot{v}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial v_{o2}\partial \vec{q}_{o2}}\dot{v}_{o2}\dot{\vec{q}}_{o2} \\
&-\frac{\partial \vec{S}_{o2}}{\partial \vec{q}_{o2}}\ddot{\vec{q}}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial \vec{q}_{o2}\partial u_{o2}}\dot{\vec{q}}_{o2}\dot{u}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial \vec{q}_{o2}\partial v_{o2}}\dot{\vec{q}}_{o2}\dot{v}_{o2} - \frac{\partial^2 \vec{S}_{o2}}{\partial \vec{q}_{o2}\partial \vec{q}_{o2}}\dot{\vec{q}}_{o2}\dot{\vec{q}}_{o2}
\end{aligned} \tag{25}
$$

Similarly, for eq. 17 we have: n

$$
\ddot{\vec{C}}_n = \frac{\mathrm{d}}{\mathrm{d}t}\left[\left(\frac{\partial \vec{n}_{o1}}{\partial u_{o1}}\dot{u}_{o1} + \frac{\partial \vec{n}_{o1}}{\partial v_{o1}}\dot{v}_{o1} + [n_{q_{o1}}]\dot{\vec{q}}_{o1}\right) + \left(\frac{\partial \vec{n}_{o2}}{\partial u_{o2}}\dot{u}_{o2} + \frac{\partial \vec{n}_{o2}}{\partial v_{o2}}\dot{v}_{o2} + [n_{q_{o2}}]\dot{\vec{q}}_{o2}\right)\right] \tag{26}
$$

$$
\begin{aligned}
\ddot{\vec{C}}_n = \quad & \frac{\partial \vec{n}_{o1}}{\partial u_{o1}}\ddot{u}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial u_{o1}\partial u_{o1}}\dot{u}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial u_{o1}\partial v_{o1}}\dot{u}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial u_{o1}\partial \vec{q}_{o1}}\dot{u}_{o1}\dot{\vec{q}}_{o1} \\
&+\frac{\partial \vec{n}_{o1}}{\partial v_{o1}}\ddot{v}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial v_{o1}\partial u_{o1}}\dot{v}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial v_{o1}\partial v_{o1}}\dot{v}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial v_{o1}\partial \vec{q}_{o1}}\dot{v}_{o1}\dot{\vec{q}}_{o1} \\
&+\frac{\partial \vec{n}_{o1}}{\partial \vec{q}_{o1}}\ddot{\vec{q}}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial \vec{q}_{o1}\partial u_{o1}}\dot{\vec{q}}_{o1}\dot{u}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial \vec{q}_{o1}\partial v_{o1}}\dot{\vec{q}}_{o1}\dot{v}_{o1} + \frac{\partial^2 \vec{n}_{o1}}{\partial \vec{q}_{o1}\partial \vec{q}_{o1}}\dot{\vec{q}}_{o1}\dot{\vec{q}}_{o1} \\
&+\frac{\partial \vec{n}_{o2}}{\partial u_{o2}}\ddot{u}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial u_{o2}\partial u_{o2}}\dot{u}_{o2}\dot{u}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial u_{o2}\partial v_{o2}}\dot{u}_{o2}\dot{v}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial u_{o2}\partial \vec{q}_{o2}}\dot{u}_{o2}\dot{\vec{q}}_{o2} \\
&+\frac{\partial \vec{n}_{o2}}{\partial v_{o2}}\ddot{v}_{o1} + \frac{\partial^2 \vec{n}_{o2}}{\partial v_{o2}\partial u_{o2}}\dot{v}_{o2}\dot{u}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial v_{o2}\partial v_{o2}}\dot{v}_{o2}\dot{v}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial v_{o2}\partial \vec{q}_{o2}}\dot{v}_{o2}\dot{\vec{q}}_{o2} \\
&+\frac{\partial \vec{n}_{o2}}{\partial \vec{q}_{o2}}\ddot{\vec{q}}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial \vec{q}_{o2}\partial u_{o2}}\dot{\vec{q}}_{o2}\dot{u}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial \vec{q}_{o2}\partial v_{o2}}\dot{\vec{q}}_{o2}\dot{v}_{o2} + \frac{\partial^2 \vec{n}_{o2}}{\partial \vec{q}_{o2}\partial \vec{q}_{o2}}\dot{\vec{q}}_{o2}\dot{\vec{q}}_{o2}
\end{aligned} \tag{27}
$$

For the same reasons which lead to eq.22 and eq.23, the previous equations 27 and 25 can be projected in tangent coordinates (discarding the orthogonal coordinate) obtaining a linear system with 4 unknowns and 4 equations:

$$
\begin{aligned}
&\begin{bmatrix} [\lambda_{uv}]^T & [0]^T \\ [0]^T & [\lambda_{uv}]^T \end{bmatrix} \begin{bmatrix} \frac{\partial \vec{P}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{P}_{o1}}{\partial v_{o1}} & -\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} & -\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \\ \frac{\partial \vec{n}_{o1}}{\partial u_{o1}} & \frac{\partial \vec{n}_{o1}}{\partial v_{o1}} & \frac{\partial \vec{n}_{o2}}{\partial u_{o2}} & \frac{\partial \vec{n}_{o2}}{\partial v_{o2}} \end{bmatrix} \begin{Bmatrix} \ddot{u}_{o1} \\ \ddot{v}_{o1} \\ \ddot{u}_{o2} \\ \ddot{v}_{o2} \end{Bmatrix} = \\
&\begin{bmatrix} [\lambda_{uv}]^T & [0]^T \\ [0]^T & [\lambda_{uv}]^T \end{bmatrix} \begin{Bmatrix} \vec{Q}_{ps} \\ \\ \vec{Q}_n \end{Bmatrix}
\end{aligned} \tag{28}
$$

where $\vec{Q}_{ps}$ and $\vec{Q}_n$ are the vectors of the known terms of eq. 27 and 25.

By means of the above formulation, one can solve 23 and 28 to get, respectively, the parametric speeds and parametric accelerations $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$ and $\{\ddot{u}_{o1}, \ddot{v}_{o1}, \ddot{u}_{o2}, \ddot{v}_{o2}\}$.

In fact this works well as far as the contact-on-point doesn't degenerate into the contact-on-line or contact-on-surface situations, where the matrices of coefficients become ill conditioned. This happens, for example, when a shaft is inserted into a cylindric hole with exactly the same diameter. Hence singular situations must be carefully monitored and handled.

Once one has obtained $\{\dot{u}_{o1}, \dot{v}_{o1}, \dot{u}_{o2}, \dot{v}_{o2}\}$ and $\{\ddot{u}_{o1}, \ddot{v}_{o1}, \ddot{u}_{o2}, \ddot{v}_{o2}\}$, it's easy to update the equations of the *point-on-plane* holonomic constraint. As described in the introduction, such condition consists in a plane (whose position, speed and acceleration about body O1 are known) which constraints a point belonging to object O2 (also position, speed and acceleration of this point about o2 must be known).

Note that eq. 23 and 28 provide speeds and accelerations in parametric coordinates, while the *point-on-plane* constraint formulation needs body-relative carthesian speeds and accelerations,

like $\dot{\vec{P}}_{o1}$, $\ddot{\vec{P}}_{o1}$ etc. It's easy, however, to compute these terms as functions of parametric speeds and accelerations: given the expression of the parametric surfaces

$$\vec{P}_{o1} = \vec{P}_{o1}\left(u_{o1}, v_{o1}\right) \tag{29}$$

$$\vec{S}_{o1} = \vec{S}_{o1}\left(u_{o2}, v_{o2}\right) \tag{30}$$

it follows:

$$\dot{\vec{P}}_{o1} = \dot{u}_{o1}\frac{\partial \vec{P}_{o1}}{\partial u_{o1}} + \dot{v}_{o1}\frac{\partial \vec{P}_{o1}}{\partial v_{o1}} \tag{31}$$

$$\dot{\vec{S}}_{o2} = \dot{u}_{o2}\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} + \dot{v}_{o2}\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \tag{32}$$

and similarly, for the accelerations:

$$\ddot{\vec{P}}_{o1} = \ddot{u}_{o1}\frac{\partial \vec{P}_{o1}}{\partial u_{o1}} + \ddot{v}_{o1}\frac{\partial \vec{P}_{o1}}{\partial v_{o1}} + \dot{u}_{o1}^2\frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial u_{o1}} + \dot{v}_{o1}^2\frac{\partial^2 \vec{P}_{o1}}{\partial v_{o1}\partial v_{o1}} + \dot{u}_{o1}\dot{v}_{o1}\frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial v_{o1}} \tag{33}$$

$$\ddot{\vec{S}}_{o2} = \ddot{u}_{o2}\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} + \ddot{v}_{o2}\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} + \dot{u}_{o2}^2\frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial u_{o2}} + \dot{v}_{o2}^2\frac{\partial^2 \vec{S}_{o2}}{\partial v_{o2}\partial v_{o2}} + \dot{u}_{o2}\dot{v}_{o2}\frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial v_{o2}} \tag{34}$$

The partial derivatives in eq. 31, 32, 33 and 34, can be obtained by straightforward numerical differentiation of equations 29 and 30.

Note that we can distinguish two components for the P,S accelerations of eq.33 and 34:

$$\begin{aligned} \ddot{\vec{P}}_{o1} &= \ddot{\vec{P}}_{o1,\parallel} + \ddot{\vec{P}}_{o1,\perp} \\ \ddot{\vec{S}}_{o2} &= \ddot{\vec{S}}_{o2,\parallel} + \ddot{\vec{S}}_{o2,\perp} \end{aligned} \tag{35}$$

The terms $\ddot{\vec{S}}_{o2,\parallel}$ and $\ddot{\vec{P}}_{o1,\parallel}$, depending on parametric tangential accelerations $\ddot{u}, \ddot{v}$ only, can be called 'tangential components' since from eq.33 and 34 it is easy to see that these vectors are always directed tangentially to the contact surfaces.

$$\ddot{\vec{P}}_{o1,\parallel} = \ddot{u}_{o1}\frac{\partial \vec{P}_{o1}}{\partial u_{o1}} + \ddot{v}_{o1}\frac{\partial \vec{P}_{o1}}{\partial v_{o1}} \tag{36}$$

$$\ddot{\vec{S}}_{o2,\parallel} = \ddot{u}_{o2}\frac{\partial \vec{S}_{o2}}{\partial u_{o2}} + \ddot{v}_{o2}\frac{\partial \vec{S}_{o2}}{\partial v_{o2}} \tag{37}$$

The terms $\ddot{\vec{S}}_{o2,\perp}$ and $\ddot{\vec{P}}_{o1,\perp}$, depending on parametric tangential speeds $\dot{u}, \dot{v}$ only, can be called 'centripetal components' (note some analogies with tangential and centripetal accelerations for classical 2D mechanics).

$$\ddot{\vec{P}}_{o1,\perp} = \dot{u}_{o1}^2\frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial u_{o1}} + \dot{v}_{o1}^2\frac{\partial^2 \vec{P}_{o1}}{\partial v_{o1}\partial v_{o1}} + \dot{u}_{o1}\dot{v}_{o1}\frac{\partial^2 \vec{P}_{o1}}{\partial u_{o1}\partial v_{o1}} \tag{38}$$

$$\ddot{\vec{S}}_{o2,\perp} = \dot{u}_{o2}^2\frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial u_{o2}} + \dot{v}_{o2}^2\frac{\partial^2 \vec{S}_{o2}}{\partial v_{o2}\partial v_{o2}} + \dot{u}_{o2}\dot{v}_{o2}\frac{\partial^2 \vec{S}_{o2}}{\partial u_{o2}\partial v_{o2}} \tag{39}$$

Now, one can see that the 'tangential' terms *do no affect at all* the computation of acceleration terms of the *point on plane* constraint (that is, inserting $\ddot{\vec{S}}_{o2,\parallel}$ and $\ddot{\vec{P}}_{o1,\parallel}$ in eq.8 gives always a null vector, because $\ddot{\vec{P}}_{o1,\parallel} \in ker([C_{q_x}])$ and $\ddot{\vec{S}}_{o2,\parallel} \in ker([C_{q_x}])$ when position constraint

is satisfied). This means that only the 'centripetal' terms have true significance for the *point-on plane* constraint.

In detail, this is very important for the practical implementation of the *sliding plane* method in a dynamical simulator: in fact the computation of the constraint vector $\vec{Q}_x$ as in eq.8 can take $\ddot{\vec{q}}_{x_P} = \ddot{\vec{P}}_{o1,\perp}$ and $\ddot{\vec{q}}_{x_S} = \ddot{\vec{S}}_{o2,\perp}$ instead of $\ddot{\vec{q}}_{x_P} = \ddot{\vec{P}}_{o1}$ and $\ddot{\vec{q}}_{x_S} = \ddot{\vec{S}}_{o2}$, with identical results. Since 'tangential' terms eq.36 and eq.37 aren't needed, there's no need to solve the system 28 for parametric accelerations: this has a positive impact on computation speed and -most important- on the ability to solve for unknown rigid body accelerations during dynamics [2].

4.4 Contact plane kinematics: optimizations

Some notes about noteworthy ways to optimize the *sliding plane* method.

Sometimes the computation of the linear systems 23 and 28 may be difficult: one may want to avoid the many parametric differentiations which are needed to recover the coefficients, not only for speed reasons but mostly because the computation of such differentiations could be numerically ill-conditioned or hard to perform. For example, numerical differentiation of surfaces can be troublesome in proximity of trimmed patches, across singularities caused by neighbouring faces in B-rep topology, when using Catmull-Clark limit surfaces, and so on.

Therefore an easier (though approximate) way to compute the $\dot{\vec{P}}_{o1}$, $\ddot{\vec{P}}_{o1}$, $\dot{\vec{S}}_{o2}$, $\ddot{\vec{S}}_{o2}$ vectors can be the following. At each update during integration, one performs only the position search of contact points $\vec{P}_{o1}$ and $\vec{S}_{o2}$, as in eq.3, then the body relative speed and accelerations could be simply obtained by backward numerical differentiation, knowing the previous position of the points. For example, to get the speeds:

$$\dot{\vec{P}}_{o1|_t} = \frac{\vec{P}_{o1|_t} - \vec{P}_{o1|_{t-\Delta t}}}{\Delta t} \qquad \dot{\vec{S}}_{o2|_t} = \frac{\vec{S}_{o2|_t} - \vec{S}_{o2|_{t-\Delta t}}}{\Delta t} \tag{40}$$

and to get the accelerations:

$$\ddot{\vec{P}}_{o1|_t} = \frac{\dot{\vec{P}}_{o1|_t} - \dot{\vec{P}}_{o1|_{t-\Delta t}}}{\Delta t} \qquad \ddot{\vec{S}}_{o2|_t} = \frac{\dot{\vec{S}}_{o2|_t} - \dot{\vec{S}}_{o2|_{t-\Delta t}}}{\Delta t} \tag{41}$$

At a cost of obvious approximations, like the one-step delay and the low precision (which anyway can be affordable when the step is very low), this trick completely avoids the linear systems 23 and 28. As a positive side effect, surface details with very low scale (bumps, dinches, scratches, small creases) are somewhat 'filtered' and smoothed out, while on the other hand the systems 23 and 28 may be affected by numerical noise when the contact point passes across surfaces with little bumps (where the curvature would range between very high values, even if the surface is smooth on a larger scale and has an average low curvature).

Note, anyway, that the same 'bump filtering' and 'surface smoothing' effect can be obtained with the original (exact) equations 23 and eq.28, if one just manages to compute the partial derivatives of surfaces with some expedients. For example in case of singularities, topology jumps, corners, creases, etc, it may be possible to get rid of original surface parametrization $P(u, v)$ in order to create a new local parametrization by use of a tangent plane with parameters u^*, v^*. In this way, an orthogonal projection from plane to underlying surface(s), for example

[2] This consideration saves us from a potential tautology: "the multibody dynamical solution includes the rheonomic constraint of eq.8 in order to solve for unknown bodies'accelerations, but the term eq.8 itself contains motion laws of references P,S which, among all other things, seem to be functions of body accelerations in their turn, as given in eq.28..." However, this 'deadlock' situation is resolved by the abovementioned consideration, that only the 'centripetal' (speed-dependent) part of P,S references has effect on the term eq.6 of the point-on plane constraint. That is, we don't need the a-priori knowledge of body accelerations in order to compute all the terms of the *sliding plane* constraint in the dynamical solution problem: speed knowledge is enough.

obtainable by a general-purpose and robust *ray-tracing* algorithm, defines a new 'singularity free' parametrization $P(u^*, v^*)$.

This means that no restrictive assumptions must be made on the $C-n$ and $G-n$ continuity of surfaces, and also surfaces whose derivatives are hard to compute can be used now (tesselated meshes, implicit surfaces, Loop subdivision surfaces, etc.)

5 Implementation

An efficient approach to the solution of the DAE (differential algebraic) problem of constrained lagrangian dynamics is discussed in [13]. The solution scheme exposed in fig. 5 relies on that method, which includes two constraint-stabilizing steps per each integration step of the underlying ODE problem. Note that, among all the constraints equations, there is always the contact constraint expressed as a *point on plane* condition.

During the iterative N-Raphson procedure, the problem of updating the position of the "sliding plane" (step A2) is uncoupled from the constraint closure problem (step A1), and both are executed at each iteration one after the other. This causes a simple implementation, while convergence of the method is still good as if A1 and A2 problems were coupled. Later, having

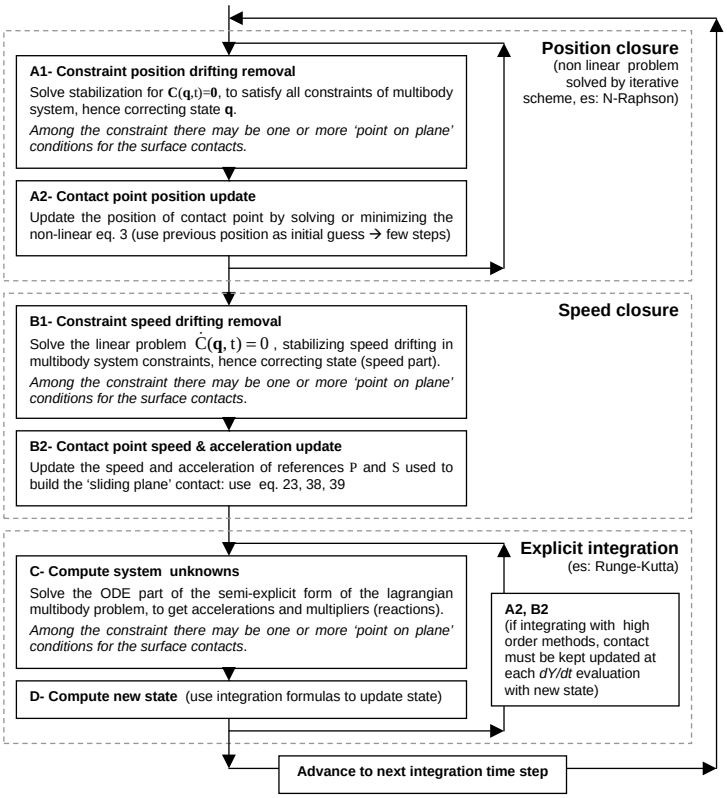

Figure 5: *Integration scheme (simplified flowchart)*

obtained the correct orientation and position for the sliding plane constraint, the speed closure of constraints can be easily solved too (step B1).

We remark that, once positions and speeds of multibody state are correct, one can compute eq. 23, 31 and 32, as well as 38 and 39 without problems (step B2), then obtaining all the informations about the speed and centripetal acceleration of the references $\vec{P}_{o1}$ and $\vec{S}_{o2}$ which are used to represent the *sliding plane* constraint.

In fact the computation of unknowns accelerations for a given state (step C) can take place

only if the kinematics of $\vec{P}_{o1}$ and $\vec{S}_{o2}$ is correct in terms of both positions, speeds and accelerations: again we stress the point that only the 'centripetal' parts of references'accelerations (depending only on parametric speeds eq.23) is required in eq.8 of the *lock formulation*, while the 'tangential' part of acceleration (which depends from eq.28) has no effect at all on that constraint.

Therefore, only after step C, one may compute also eq. 28, eq.33 and eq.34 in order to get also the complete accelerations of contact points (i.e. including tangential effect), if needed.

6 Examples

To validate the model of contact, we built a simple cam-follower mechanism using the three dimensional modeling environment of our multibody software. The cam and the follower are made of NURBS bi-parametric surfaces (fig. 6).

In detail, the shape of the cam has been created with a procedural modeling tool which uses the formulas in [11], where one gets the profile as a function of the motion law imposed to the follower. Therefore, using a test motion law, we built the cam surface for that motion, using 200x4 control points (the more the samples, the less the approximation).

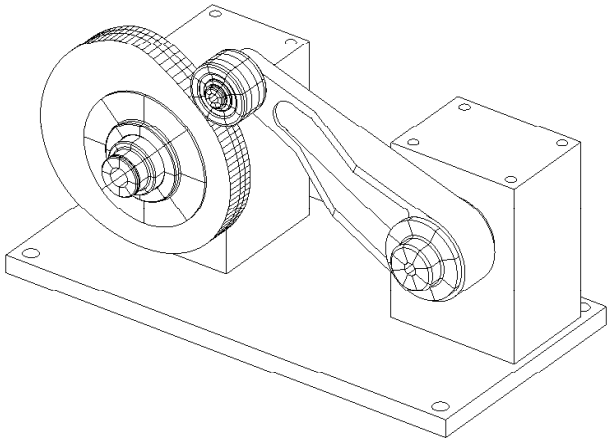

Figure 6: *Cam-follower benchmark for contact between freeform surfaces*

We performed the simulation of the mechanism, and compared the resulting motion of the follower (moved only by contact) with the hypothetical "exact" motion law that we used to build the cam. We observed little or no differences in position and speed of the follower, but sometimes a small noise can affect acceleration (fig.8) mostly because the cam hasn't an analytical shape, but it is approximated by 5th-degree Nurbs).

7 Conclusions

An approach has been proposed for the multibody simulation of sliding contact between freeform surfaces. The geometric constraint has been represented by means of a tangential plane which moves between the contact bodies, hence only a simple "point on plane" constraint had to be added to the system of motion-equations. On the other side, the problem of computing the auxiliary variables of the contact constraint (position, speed and acceleration of contact point) could be solved separately, mostly for sake of better performance. The theoretical result have been implemented into our general-purpose multibody software and have been successfully tested with real world examples. Future developements may embrace the application of these results to non-parametric surfaces and the problem of high-performance collision detection.

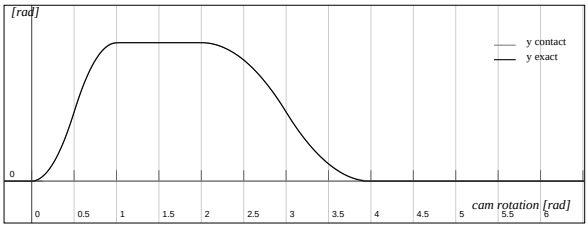

Figure 7: *Comparing the two motion laws: exact (analytical original) and simulated motion of follower (moved by contact): they overlap perfectly.*

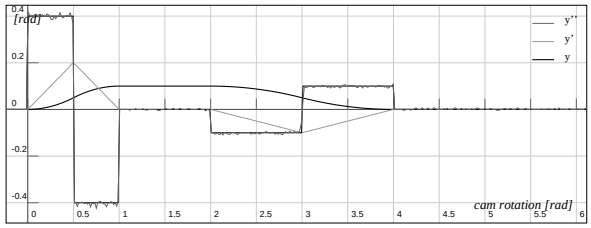

Figure 8: *Speed and acceleration comparison. The speed profile of simulated motion coincides exactly with the analytical profile. Note some high frequency numerical noise on acceleration.*

References

[1] A.Tasora and P.Righettini, *Application of quaternion algebra to the efficient computation of jacobians for holonomic-rheonomic constraints*, EUROMECH 404, Lisboa, 1999.

[2] C.Balling, *Formulation of a class of higher-pair joints in multibody systems using joint coordinates*, Multibody System Dynamics vol.3, Ed. Kluwer Academic Publishers, The Netherlands, 1999, 21-45.

[3] J.Cohen, M.C.Lin, D.Manocha and M.Ponamgi, *I-collide: an interactive and exact collision detection system for large-scale environments*, Proc. of ACM Interactive 3D Graphics Conference, 1995, 189-196.

[4] S.Gottschalk, M.C.Lin, D.Manocha, *OBB tree: a hierarchical structure for rapid interference detection*, Proc. of ACM Siggraph'96, 1996, 171-180.

[5] S.Krishnan, A.Pattekar, M.Lin and D.Manocha, *Spherical shell: a higher order bounding volume for fast proximity queries*, Proc. of 3rd International Workshop on Algorithmic Foundations of Robotics, 1998.

[6] D.Baraff, *Curved Surfaces and Coherence for Non-penetrating Rigid Body Simulation*, Computer Graphics (Proc. ACM Siggraph'90, Dallas) Volume24, Number 4, August 1990.

[7] D.Baraff, *Linear-Time Dynamics using Lagrange Multipliers*, Computer Graphics (Proc. ACM Siggraph'96, New Orleans) August 1996.

[8] Ch.Glocker and F.Pfeiffer, *Dynamical systems with unilateral contacts*, in Nonlinear Dynamics 9(3), 1992, 245-259.

[9] H.M.Lankarani and M.Pereira, *Treatment of impact with friction in multibody mechanical systems*, EUROMECH 404, Lisboa, 1999.

[10] S. Rao, *Engineering Optimization: Theory and Practice*, Wiley and Sons.

[11] G.Ruggieri, Magnani, *Progettazione meccanica funzionale* ed. UTET, Torino, 1990

[12] R.Roberson and R.Schwertassek, *Dynamics of multibody systems*, Ed. Springer Verlag, Berlin 1988.

[13] A.Tasora, *An optimized lagrangian multiplier approach for interactive multibody simulation in kinematic and dynamical digital prototyping*, VII ISCSB, Milano, 2001.

[14] A.Shabana, *Multibody Systems*, Ed. John Wiley Sons, New York 1989.