All You Need Is Force: a constraint-based approach for rigid body dynamics in computer animation

Kees van Overveld and Bart Barenbrug

Department of Mathematics and Computing Science, Eindhoven University of Technology
PO Box 513 ; 5600 MB Eindhoven, the Netherlands

Abstract. Over the last few years, simulating the motion of linked articulated rigid bodies based on classical rigid body dynamics has become a valuable paradigm for making realistic 3-D computer animations. Although several operational methods for dynamical simulation have been developed, in general these are both conceptually and computationally complex. To inspire further research in devising alternative and possibly simpler schemes for dealing with articulated rigid bodies, this paper discusses an alternative approach to rigid body dynamics which is based on (conceptually much simpler) point mechanics. Geometric constraints, e.g. the requirement that the distance between two points should be conserved, take the form of additional algebraic equations. We propose to solve these algebraic constraints in concert with the numerical integration. First, we give a general formulation of such a scheme. Next, we describe a preliminar implementation on the basis of a very naive numerical solver for ODE's (ordinary differential equations).

1 Introduction

(The first few paragraphs of this introduction have appeared earlier, in a different context, in [8])

The mathematical theory of rigid body dynamics has been developed over the past 200 years by Euler, Lagrange, Poincelet and others. One of the main applications of this theory was the quantitative description of motions of celestial bodies. This meant that the theory had to be formulated such that numerical results could be obtained without having to rely on elaborate numerical calculations: sophisticated analytical methods were used instead.

What if computers and numerical methods would have existed in the 18th century?

An interesting question then is whether this would have changed the appearance of the theory, i.e. if the theory would emphasise numerical methods rather than analytical techniques. As a consequence, would the collection of canonical applications of the theory comprise more examples in the context of the motion of composite mechanisms and articulated systems, rather then (symmetrical) tops and celestial bodies?

Let us elaborate a bit on this speculation.

Much of the mathematical complexity of rigid body dynamics is due to the introduction of angle coordinates rather than Cartesian coordinates: the geometry of $O(3)$ (the space of three-dimensional rotations) is much more complicated than the geometry of $E(3)$ (the three dimensional Euclidean space). This reflects itself in the occurrence of torques and angular momenta, resulting in coupled nonlinear ODE's in the angular velocities[1].

Now suppose instead that rigid objects were described as a collection of N point masses, whose locations were represented by points in a $3N$-dimensional Euclidean space (like a gas consisting of N molecules). Its phase space, which also contains the velocities of the points, therefore would be $6N$-dimensional. The rigidity of the object then might be assured by extending this system with a sufficient number of geometrical constraints. Since the state vector of a rigid object has 12 components (3 coordinates of the centre of gravity, 3 components of the translation vector, 3 Euler angles and 3 components of the angular velocity vector), the number of scalar constraints would be $6N$-12. These constraints could e.g. state that the distances between appropriate point-pairs has to be invariant and hence that the relative velocity component in that direction vanishes. (We will call these constraints length constraints; one length constraint therefore adds effectively two scalar constraints, so in this case $3N$-6 length constraints are needed).

By this manipulation, we have replaced a problem of differential geometry in $E(3) \times O(3)$ by a simpler problem in $E(3N)$ together with $3N$-6 length constraints. The differential equations of the latter problem are trivial when compared with the differential equations of the former problem: they take the form $F = m\ddot{x}$ for each of the point masses, whereas the original problem gives rise to the Euler equations of a rotating rigid body ([2]).

In other words, we have exchanged a great deal of analytical complexity by numerical complexity. Of course, such a manipulation would not have made sense in the 18th century, without the availability of numerical methods and computers to implement these, so it is obvious that the classical textbook theory has not been developed along these lines.

Nowadays, however, computers and efficient numerical algorithms are available, so an alternative development of rigid body dynamics, with a larger emphasis on these numerical algorithms is possible.

Now the major motivation of the current work can be stated as follows: we want to investigate the applicability of the alternative formulation of rigid body dynamics, based on point-mass mechanics enhanced with additional constraints, such as the length constraints introduced above.

Even though a definitive implementation should use more sophisiticated numerical methods, the results of our preliminar implementation already show the potential merits of our scheme. One of the consequences of our scheme is that, to keep the structure of the ODE's as simple as possible, rigidity is also treated as a constraint, thus avoiding the necessity of introducing angular momentum and torque. As a result, we arrive at a particular transparent representation where both linked rigid objects, collision handling, point and line hinges and point-to-curve (PTC) constraints all are dealt with in

[1] See [2] for a treatment of classical rigid body mechanics, and [1], [9], [5], [4], [10], [6] for the application of classical rigid body mechanics to computer animation.

the same formal context. We will also study the qualitative and some quantitative merits of such methods.We stress, however, that the primary claim of this paper is NOT that point-mass based mechanics is in some way a better way to do rigid body mechanics than the classical Euler-type mechanics; it rather aims at provoking a discussion on the issue of whether computer based simulations should be founded on the same theoretical formulations as developed in the pre-computer era, OR that some aspects of classical theories might profit from minor re-structuring in order to provide a smoother transition from theory to computer simulation.

In section 2, the numerical scheme in its general setting is presented which will be used throughout this paper. Next a toy implementation of this scheme is introduced which consists of taking the Euler midpoint method as ODE solver, since this gives rise to straigthforward mathemetical manipulations. In 2.1, the scheme is introduced for the examples of a point mass which collides with a plane, and for a swinging pendulum. The constraints here arise from the requirement that the point mass does not penetrate the collision plane, and the conservation of the length of the pendulum, respectively. In 2.2, we study constraints that express the rigidity of an object; this gives rise to our formulation of rigid body dynamics. In 2.3 we study the coupling of two rigid bodies via a PTP constraint. This constraint gives rise to additional forces, and in the canonical formulation, to additional torques as well. We show how such torques can be eliminated, so that our original representation, based on forces only, still holds. A slightly more complicated interface between two rigid objects is the line hinge; it is discussed in 2.4. Section 2.5 discusses collision response based on constraint forces for articulated rigid objects. Point-to-curve constraints, e.g. needed for modelling moving beads on a curved string, or roller coasters, are the topic of 2.6. In section 3 we assess some quantitative properties of our algorithm which allow comparison with the standard methods. Section 4 summarises our results and concludes the paper.

2 Rigid Body Dynamics = Point Dynamics + Constraints

Consider a dynamical system with state vector $\phi = \phi(t)$, given by the evolution equation $\dot{\phi} = f(\phi; F)$ where $F = F(t)$ is the combined vector of all (*a priori* unknown) internal reaction forces in the system. The system should satisfy the algebraic constraints $c(\phi) = 0$ at all times. Suppose we have a numerical method N to give an estimate for $\phi(t+h)$, given $\phi(t)$ and $F(t)$ (and possibly the values of $\phi(t_i)$ at earlier time points $t_i < t$). Since $F(t)$ is *a priori* unknown we cannot compute $\phi(t+h) = N(\phi(t); F(t))$ right away. Therefore we adopt the assumption that the reaction forces $F(t)$ change slowly over time. That is $F(t) = F(t-h) + \delta F$ with δF in some sense small. We introduce an iterative scheme where in each step the estimate for δF, and hence for $F(t)$ (and hence the estimate for $\phi(t+h)$) will be improved. The values for $F(t)$ and $\phi(t+h)$ will be labeled by superscript k to distinguish the several subsequent estimates, so $\phi^k = N(\phi; F^k)$; and $\phi^{k+1} = N(\phi; F^{k+1})$ is supposed to be a better estimation for $\phi(t+h)$ than ϕ^k. Without a superscript, ϕ is short for $\phi(t)$. We start the iteration by taking for δF the value 0, so $F^0(t) = F^\infty(t-h)$. Next the algorithm looks as follows:

$$
\begin{array}{l}
F^0(t) := F^\infty(t-h);\\
\delta F(t) := 0;\\
\phi^0 := N(\phi; F^0);\\
k := 0;\\
\textbf{do}\\
\quad \textbf{compute improved estimate for } \delta F\\
\quad F^{k+1} := F^0 + \delta F;\\
\quad \phi^{k+1} := N(\phi; F^{k+1});\\
\quad k := k+1;\\
\textbf{until convergence}
\end{array}
\tag{1}
$$

Notice that we have not explained how $\delta F = \delta F(\phi^k)$ should be obtained. It goes without saying that here the algebraic constraint $c(\phi) = 0$ comes into play. Indeed, assume that $c(\phi) = 0$ holds for $\phi(t)$. It should hold as well for $\phi(t+h)$. If we demand it to hold for ϕ^k then we get

$$
\begin{aligned}
0 = c(\phi^k) &= c(N(\phi; F^k))\\
&= c(N(\phi; F^{k-1} + \delta F))\\
&= c(N(\phi; F^{k-1}) + \frac{\partial N}{\partial F}\delta F + \cdots)\\
&= c(N(\phi; F^{k-1})) + \frac{\partial c}{\partial N}\frac{\partial N}{\partial F}\delta F + \cdots\\
&= c(N(\phi; F^{k-1})) + \frac{\partial c}{\partial F}\delta F + \cdots
\end{aligned}
\tag{2}
$$

This gives rise to the Newton-like method for computing δF^k:

$$\delta F^k = -(\frac{\partial c}{\partial F})^{-1} c(N(\phi; F^{k-1})$$

. This is the suggested scheme in the most general setting. In order to demonstrate how it works, we choose a naive numerical method N, namely the Euler midpoint method. Also, we will make use of additional knowledge of the structure of the constraints $c = 0$ to avoid the explicit computation[2] of the Jacobians $\frac{\partial c}{\partial F}$.

A differential equation of the form

$$F(t) = m\ddot{x}(t) \tag{3}$$

can be solved numerically by observing that

$$\ddot{x}(t) = \frac{x(t+h) + x(t-h) - 2x(t)}{h^2} + O(h^2)$$

[2] Observe, however, that an explicit computation of $\frac{\partial c}{\partial F}$ would not even be that expensive since every constraint component of c typically depends on very few components of F, so the Jacobian consists of small blocks.

so

$$x(t+h) = 2x(t) - x(t-h) + h^2\,\ddot{x} + O(h^4). \tag{4}$$

Therefore an assignment to $x(t+h)$ should be:

$$x(t+h) = 2x(t) - x(t-h) + h^2 F(t)/m.$$

Given $x(t)$ and $x(t-h)$ and $F(t)$, the function N in our scheme, using this Euler method, is $N(x(t), x(t-h), F(t)) = 2x(t) - x(t-h) + h^2 F(t)/m$.

2.1 Mechanical Systems without Internal Structure: a Colliding Mass Point and a Swinging Pendulum

Consider a point mass moving in the positive x-direction with location $x(t)$ with uniform velocity. For $x = x(t_0)$, an impenetrable wall, perpendicular to the x-axis, blocks the motion of the point mass, so for $t = t_0$ a collision takes place. In a fully *in*elastic case, the point mass would come to a full stop due to an infinitely large force that works during an infinitely short time interval. In our discretised model, we assume that a finite, constant force works during the entire time interval of length h from t_0 to $t_0 + h$. So

$$x(t_0) = x(t_0 + h) = 2x(t_0) - x(t_0 - h) + h^2 F/m$$

and hence

$$F = m\frac{x(t_0 - h) - x(t_0)}{h^2}.$$

In this (trivial) example we see how evaluating a geometric constraint of the form $c(x(t+h)) = 0$ may serve to compute a constraint force at time t.

A less trivial example is a physical pendulum. Here $x(t)$ is the location of a point mass in $\Re^3$ that moves while keeping its distance to a given point, say the origin, constant. The motion equation reads

$$F(t) + G = m\ddot{x}(t),$$

and the numerical scheme is

$$x(t+h) = 2x(t) - x(t-h) + h^2(F(t) + G)/m.$$

The force G is a known gravity force; $F(t)$ is an *a priori* unknown reaction force that is induced by the constraint $c(x) = (x(t), x(t)) - l^2 = 0$. Rather than computing $(\frac{\partial c}{\partial F})^{-1}$, we proceed as follows: we can evaluate this constraint at $t + h$ which gives a quadratic equation in the magnitude of F. Since the direction of F is known ($F(t)$ is parallel to $x(t)$), this completely defines the reaction force.

2.2 Rigidity is a constraint.

We consider a 3-dimensional rigid object with total mass M and inertia tensor I. Assume that this object is composed of a large number of point masses, labeled with i, with locations $x_i(t)$ and masses m_i. These point masses have fixed relative positions with respect to each other since the object is rigid, so they can be written as $x_i = c + \sum_j \rho_{ij}\, B_j$

where c is the centre of gravity; B_j, with $j = 0, 1, 2$, are basis vectors for some body fixed frame, and ρ_{ij} are fixed coefficients expressing the relative position of x_i with respect to the frame $\{B_j\}$. Assume for the moment that c is in rest, so the object is only rotating round c. The kinetic energy then can be written either as $E_{kin} = \frac{1}{2} I\omega^2$ or $E_{kin} = \sum_i \frac{1}{2} m_i \dot{x}_i^2$. Elaborating the second expression gives

$$\begin{aligned} E_{kin} &= \sum_i \frac{1}{2} m_i (\dot{x}_i, \dot{x}_i) \\ &= \sum_i \frac{1}{2} m_i \sum_{j\,k} \rho_{ij}\, \rho_{ik} (\dot{B}_j, \dot{B}_k) \\ &= \frac{1}{2} \sum_{j\,k} \gamma_{j\,k} (\dot{B}_j, \dot{B}_k), \end{aligned} \tag{5}$$

$$\tag{6}$$

where $\gamma_{j\,k} = \sum_i m_i \rho_{ij}\, \rho_{ik}$. Now suppose for the moment that γ is diagonal, then

$$E_{kin} = \frac{1}{2} \sum_j \gamma_{jj} (\dot{B}_j, \dot{B}_j). \tag{7}$$

This means that if we consider the $B_j = B_j(t)$ as generalised coordinates, the associated Euler equation takes the form (see [2]):

$$F_j = \gamma_{jj} \ddot{B}_j. \tag{8}$$

The force F_j in the left hand part is not defined yet; it will follow from the rigidity constraints to be discussed below. We see that for coordinates B_j, the motion equation 8 takes exactly the form of 3 when we consider γ_{jj} as generalised mass parameter. Also note that, due to the assumption of diagonality of γ, the motion equations for each of the three B_j are un-coupled. From now on we will assume that rigid bodies posses a body fixed frame with base vectors B_j such that γ is diagonal. This base can be found by diagonalising the inertia tensor of the object. For convenience we scale each of the B_j such that the corresponding γ_{jj} reduces to unity, say $|B_j| = \beta_j$, so the remaining motion equations are

$$F_j = \ddot{B}_j. \tag{9}$$

Forward integration, ignoring F yields

$$B_j(t+h) = 2B_j(t) - B_j(t-h) \tag{10}$$

and similarly for the centre of gravity:

$$c(t+h) = 2c(t) - c(t-h). \tag{11}$$

A rigid object has six degrees of freedom (location and orientation). Our representation has twelve (the four vectors c, B_0, B_1 and B_2; assuming an object of dimension 3). So we have to impose some additional constraints that ensure that the generalised coordinates remain an orthogonal base. Using j, j' and j'' to distinguish the three base vectors we must take care that:

- $(B_{j'}, B_{j''}) = 0$, for all j
- $|B_j| = \beta_j$, for all j

The forward integration 10 of the generalised coordinates might violate these constraints, so we need to compute the values of the forces F_j from 9. We derive the F_j using Lagrange multipliers. The directions of these forces are defined beforehand: for the orthogonality, the correction on B_j has a component in the direction of $B_{(j+1)mod3}$ and one in the direction of $B_{(j+2)mod3}$. For the length constraint, the correction forces are directed along the base vector that is to be corrected. So again we can avoid to compute the Jacobian $\frac{\partial c}{\partial F}$. The *a priori* unknown magnitudes of the correction forces are represented by Lagrange multipliers. For each orthogonality constraint, we introduce the multiplier Λ_j (enforcing the constraint $(B_{j'}, B_{j''}) = 0$; note the antisymmetric naming convention for the indices. For each length constraint, we introduce the multiplier λ_j (enforcing $|B_j| = \beta_j$). Adding the appropriate terms to the motion equations of the generalised coordinates yields for B_j: (we assume the factors h^2 to be included within the values of the multipliers)

$$B_j(t+h) = 2B_j(t) - B_j(t-h) + (\Lambda_{j'} B_{j''} + \Lambda_{j''} B_{j'} + \lambda_j B_j)(t) \qquad (12)$$

From the condition that the rigidity constraint should also be satisfied for time $t + h$, we can derive equations for the values of the multipliers. In the case of the pendulum in section 2.1, we found a quadratic equation for the multiplier that could be solved without iteration; here we find six coupled quadratic equations. This means that we now have to resort to the iterative method as explained in section 2, but rather than computing and inverting the Jacobians $\frac{\partial c}{\partial F}$ we make use of the geometric meaning of the reaction forces to find linearised approximations of their magnitudes in a straightforward manner.

In the absence of any other constraint (see sections 2.3, 2.4, 2.5, 2.6) this iterative method converges extremely fast. In practice, the number of iterations is fixed to about 3 or 4.

2.3 Coupled Rigid Bodies Require Dealing with Point-to-point Constraints

In case a rigid objects moves subject to an external force F (scaled such that the factor h^2 is taken into account), we have to have a device for translating F into a set of equivalent forces on the centre of gravity and the B_j, since these are the only coordinates of our representation. The effect of F on the centre of gravity c is trivial: $F = m\ddot{c}(t)$. If F works on a given point p_i, one of the points x_i of the rigid object, it causes a torque, $(p_i - c) \times F$. The effect of this torque is going to be accounted for by introducing 3 forces, F_{Bj} on the B_j. So we have

$$\sum_j F_{Bj} = 0 \qquad (13)$$

and

$$\sum_j B_j \times F_{Bj} = (p_i - c) \times F \qquad (14)$$

This does not completely determine the F_{Bj}: we have room for additional requirements on the F_{Bj}. We may demand the F_{Bj} to be such that the rigidity constraints are still satisfied:

$$|B_j + F_{Bj}| = \beta_j; \tag{15}$$

$$(B_{j''} + F_{Bj''}, B_{j'} + F_{Bj'}) = 0. \tag{16}$$

Choosing the F_{Bj} such that they don't affect the rigidity constraints up to first order in the F_{Bj} means that the rigidity correction from section 2.2. converges fast.

If we linearize these equations, assuming the the F_{Bj} to be small we get

$$(B_j, B_j) + 2(B_j, F_{Bj}) = {\beta_j}^2; \tag{17}$$

$$(B_{j''}, B_{j'}) + (F_{Bj'}, B_{j''}) + (F_{Bj''}, B_{j'}) = 0. \tag{18}$$

Using 13, 14, 17, and 18, we can compute matrices A_j such that $F_{Bj} = A_j F$. The computation of the A_j is explained in [3].

(The linearity condition is not necessarily true. In the case of F being the force due to a PTP constraint, however, we again use the argument that F varies gradually, and a iterative approach is used to find *increments* in F_{Bj} between the previous and the current value, that *are* expected to be small. If F is an external force that is allowed to vary significantly over time, it is split in a number of small components that are accounted for subsequently within one time step.)

Applying a force F to an object means that for that object we have to do the following assignments:

$$c := c + \frac{1}{m}F \tag{19}$$

and for each j:

$$B_j := B_j + A_j F. \tag{20}$$

Note that in the derivation of the torque distribution matrices A_j, the rigidity constraints were linearised. The rigidity constraint mechanism, which works in parallel with the PTP constraint mechanism, thanks to the iterative approach, is used to restore the rigid shape of the object again, thus correcting for the ignored non-linearity.

Using these formulae, also the effect (=the displacement) of a force on one of the vertices of that object can be calculated. Using the definition of the ρ_{ij}, we have:

$$\begin{aligned} p_i &:= (c + \frac{1}{m}F) + \sum_j \rho_{ij}(B_j + A_j F) \\ &:= p_i + MF, \end{aligned} \tag{21}$$

where $M = (\frac{1}{m}I + \sum_j \rho_{ij} A_j)$.

So also the displacement of an arbitrary vertex due to the application of a force to a given location of a rigid body can be expressed using a 3×3 matrix.

So far we assumed that F was a *known* force. The same observations, however, hold for an *a priori* unknown force, such as the force in a PTP hinge (constraint). Let the two vertices that are connected via a PTP constraint be distinguished by the superscripts k and l: they belong to two different rigid objects. The matrices M for the corresponding

vertices of these objects are M^k and M^l, respectively. The location of a vertex after application of the correcting PTP constraint force, F_{ptp} is denoted by a tilde ($\tilde{}$). We then have:

$$\begin{aligned}
&\tilde{p}^k = \tilde{p}^l, \text{ so} \\
&p^k + M^k F_{ptp} = p^l - M^l F_{ptp}, \text{ and hence} \\
&p^l - p^k = (M^k + M^l) F_{ptp}
\end{aligned} \tag{22}$$

The point to point error in the absence of F_{ptp}, $p^l - p^k$ is known, as is matrix $M^k + M^l$, so the reaction force F_{ptp} can be calculated from 22 as $F_{ptp} = (M^k + M^l)^{-1}(p^l - p^k)$.

Although we didn't compute $M^k + M^l$ via differentiation of the constraint equation ($p^l - p^k = 0$) with respect to F_{PTP}, this matrix does play the role of the Jacobian $\frac{\partial c}{\partial F}$.

Note that in the presence of more than two coupled rigid objects, we arrive at a coupled set of constraint force equations. As previously, the combined effect of several of such forces may be accounted for by means of iteration. Alternatively, all linearised equations may be grouped together and be solved simultaneously by means of a standard LU-decomposition method.

2.4 A Pair of Point-to-point Constraints Removes Two Degrees of Freedom

Consider two rigid objects. They are coupled with two PTP constraints, so effectively they share a line hinge. This means that (1) they have only one relative DOF left (rotation round the line through the two point pairs), as opposed to 3 relative DOF in the case of one PTP constraint; and (2) the two PTP reaction forces may contain components along this line of arbitrary, mutually opposite, magnitudes. This means that the approach from 2.3 does not work. We have to be careful to avoid the undetermined force components growing beyond boundaries, thus making the computation instable.

Let the point to point reaction forces be denoted by F_1 and F_2. If the the pair (F_1, F_2) satisfies the line hinge constraint, then so will the pair $(F_1 + \lambda e, F_2 - \lambda e)$ for any λ with e being the hinge vector, since the effects of the added terms are exactly opposite. To avoid growth of such components, we must find a way to minimize them. This can be done by forcing the components of both reaction forces in the direction of e to be the same. This yields:

$$\begin{aligned}
&(F_1 + \lambda e, e) = (F_2 - \lambda e, e), \text{ so} \\
&\lambda = \frac{(F_2 - F_1, e)}{2(e, e)}.
\end{aligned} \tag{23}$$

Replacing F_1 by $F_1 + \lambda e$ and F_2 by $F_2 - \lambda e$ with λ from 23 solves the problem.

2.5 Collision Forces Result from Constraints

When a vertex of a rigid object collides with a collision plane, this plane exerts a reaction force onto that vertex that prevents the vertex from crossing the collision plane (and perhaps causes the vertex to bounce off the surface). In case of a completely inelastic

collision, the result of the reaction force (which we assume to be directed perpendicular to the collision plane) is that the velocity of the vertex perpendicular to the collision plane is zero after the collision. This observation can be used to calculate the reaction force.

Let n be the normal of the collision plane, and the reaction force λn (for a yet unknown λ). Let $p(t)$ be the location of the colliding vertex at the current time stamp, and $p(t+h)$ the location at the next time stamp as it would be without application of a reaction force (so $p(t)$ and $p(t+h)$ lie at opposite sides of the collision plane). The position of the vertex after application of the reaction force will be denoted by $\tilde{p}(t+h)$ ($= p(t+h) + M\lambda n$, using 21). The (discrete) new velocity of the vertex is then given by $\tilde{p}(t+h) - p(t)$. Let, for vectors v and w, $[v]_w$ denote the component of v in the direction of w. The observation that the new velocity of the vertex perpendicular to the collision plane is zero after the collision force is applied then yields:

$$\begin{aligned} 0 &= [\tilde{p}(t+h) - p(t)]_n \\ &= [p(t+h) + M\lambda n - p(t)]_n \\ &= [\frac{(p(t+h) + M\lambda n - p(t), n)}{(n,n)}]_n \\ &= (p(t+h) - p(t) + M\lambda n, n) \\ &= (p(t+h) - p(t), n) + \lambda(Mn, n), \end{aligned} \tag{24}$$

and hence

$$\lambda = -\frac{(p(t+h) - p(t), n)}{(Mn, n)}, \tag{25}$$

which again was obtained without explicitly computing $(\frac{\partial c}{\partial F})^{-1}$.

So using this constraint we can find the reaction force λn for a completely inelastic collision. This force operates upon the point p, and the method from section 2.3 is used to account for the effect of this force on the entire rigid object. For an elastic collision, we only have to enlarge the reaction force (to cause the vertex' speed to be 'reflected' by the collision plane), so for an arbitrary collision we have a reaction force of $(1+\gamma)\lambda n$ for an elasticity coefficient γ which is zero for complete inelasticity and one for a completely elastic collision. Note that a more sophisticated approach should take the precise time point for the collision into account. We make a mistake here of at most h, which may cause some aliasing effects if h is relatively large.

2.6 Computing the Constraint Forces for a Point-to-curve Constraint

Consider a point p, which may be a point from a rigid object, which is constrained to move along a curve g, $g : \Re \rightarrow \Re^3$. The force between p and g may be modeled by a PTP constraint where one of the two points is an *a priori* unknown point on the curve, say $g(s)$. To account for the movement along the curve, s should be modified each frame.

Let the change in s be δ. Then δ represents the velocity along the curve. If p would move uniformly, the assignment $s := s + \delta$ would suffice to describe p's motion.

After the new position $g(s + \delta)$ is calculated, the required reaction force F_{ptp} that pulls p to $g(s + \delta)$ can be computed based on the PTP constraint.

(Notice that we cannot simply write that the direction of F_{ptp} is directed along $g(s + \delta) - (2p(t) - p(t - h))$, since pulling on $p(t)$ also causes a torque, and hence an angular acceleration to operate on the object. Instead we have to apply the method from section 2.3, using the M-matrix.)

Assume a given value for δ. Then in general the resulting F_{ptp} will contain both a normal component and a tangential component. Let the latter component be $\lambda \frac{d}{ds} g(s)$. Assume the case of a friction-less contact between p and g. Then F_{ptp} can only be perpendicular to the curve. So if we find a F_{ptp} with $\lambda \neq 0$, the starting value for δ was wrong, and a correction is e.g. $\delta := \delta - \kappa\lambda$ for some positive κ. In this manner, again an iterative algorithm is obtained to find the appropriate value for δ. This iterative process is executed in parallel with the algorithms for rigidity, PTP constraints, line hinges and collision response as explained in sections 2.2-2.5: all these algorithms work by improving the estimates for the corresponding reaction forces, and when they all converge the total collection of all reaction forces is known such that all constraints are being met.

The curve parameter s should stay within the domain of g. The current implementation in our animation system is such that, when this isn't the case, the PTC constraint is cancelled automatically. This causes e.g. a roller coaster vehicle to fly off the track in a physically correct way when it reaches the end. An animator can always ensure that the vertex doesn't leave the curve by placing collision planes on its extremes.

3 Some Quantitative Properties of The Algorithm

The algorithms dealing with constraints as outlined above have been implemented in the Eindhoven computer animation system WALT ([7]. To asses the numerical properties of the algorithm, some experiments were performed.

Experiment 1

In the first experiment the configuration consisted of one object (a cube with sides of length 2), zero gravity, and two equal forces (working in opposite directions) working on two different point-masses of the cube. These forces cause the cube to rotate with precession. During the first 1000 frames (approximately 50 full rotations) the rotational energy of the cube was measured. It follows that after the forces have accelerated the cube, all energy in the system is preserved within 4 digits of accuracy. No energy is lost during the orthogonalisation and preservation of the length of the base vectors. For this experiment there were only two iterations for the rigidity constraints per frame.

Experiment 2

Next, a PTP constraint was added to keep one vertex of a cube fixed in space. Gravity is introduced to cause a swinging motion of the cube. Again, there were only two iterations for the rigidity constraints per frame and now also only two iterations per frame for the PTP constraints. For this configuration, we found that the value of the orthogonality-preserving multiplier oscillates in a completely periodic fashion between + and - 3.1×10^{-6} and the value of the length-preserving multiplier varies in the same phase between

0 and 2.3×10^{-3}.

The magnitude of the PTP force consists of a constant contribution of about 0.09 due to gravity and a varying centripetal force with magnitude between 0 and 0.1 which depends on the instantaneous angular velocity. The error in the PTP constraint does not exceed 4.1×10^{-12}, which is probably due to numerical quantisation in the internal representation of real numbers. We also measured the total energy of the system. The observed dissipation of about 0.4% per 1000 iterations is chiefly due to the truncation error of the assignment 4.

To further explore this energy-loss, the loss of energy was also measured as a function of the number of iterations for the PTP constraints (with this number varying between 1 and 100). The value of the energy-loss turned out to be independent of the number of iterations. Also the maximum PTP-error was measured as a function of the number of iterations. Only in case of one iteration this value is significant (0.003). For larger numbers, the error is masked by the errors of the internal representation of the reals (it is of order 10^{-12} or smaller).

Experiment 3

Next, the configuration of experiment two was expanded with another cube connected to the first cube by a PTP constraint. Again, gravity causes a (chaotic) swinging motion of the two coupled cubes.

For this configuration, the maximal PTP error as a function of the number of iterations decreases exponentially from 3.6×10^{-4} for 7 iterations to $< 10^{-6}$ for 15 iterations.

The energy-loss of the system is also measured as a function of the number of applied iterations. It turns out that after approximately 15 iterations no further accuracy gain is obtained.

In figures 1 and 2 two animation fragments can be seen of somewhat more complex systems. Figure 1 models an articulated beach chair tumbling of the stairs which demonstrates the interplay between rigidity constraints, line hinges and collision response. An example of the combination of rigidity, PTC constraints and line hinges can be seen in 2 where a constellation of two linked rigid objects slides from a curved rail. Note that the topmost object is connected via two PTC constraints to the rail. The colour plates show some stills from the tumbling articulated beach chair sequence.

4 Discussion; Summary

We propose a method to implement rigid body dynamics for computer animation based on two central ingredients:

rigidity is a constraint: several approaches for rigid body simulation are based on the notion of constraints to emulate e.g. the couplings between rigid components and hinges. We take this idea one step further to consider even rigidity proper as a constraint. This reduces the formal framework for the motion equations to point mass dynamics plus algebraic constraints.

use forces only: rigidness does not occur in our motion equations proper. This means that there is no need for angular coordinates, and hence angular momentum and

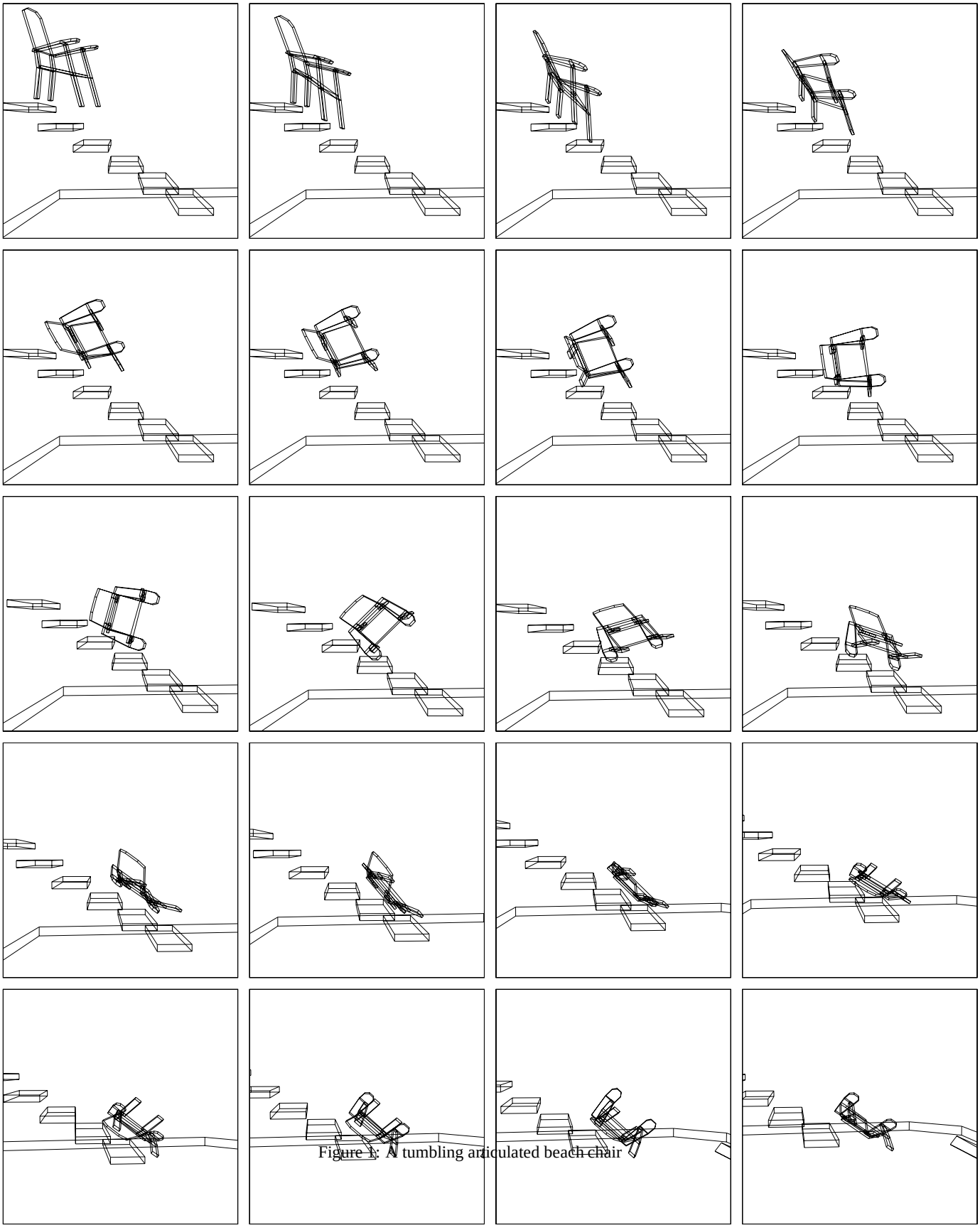

Figure 1: A tumbling articulated beach chair

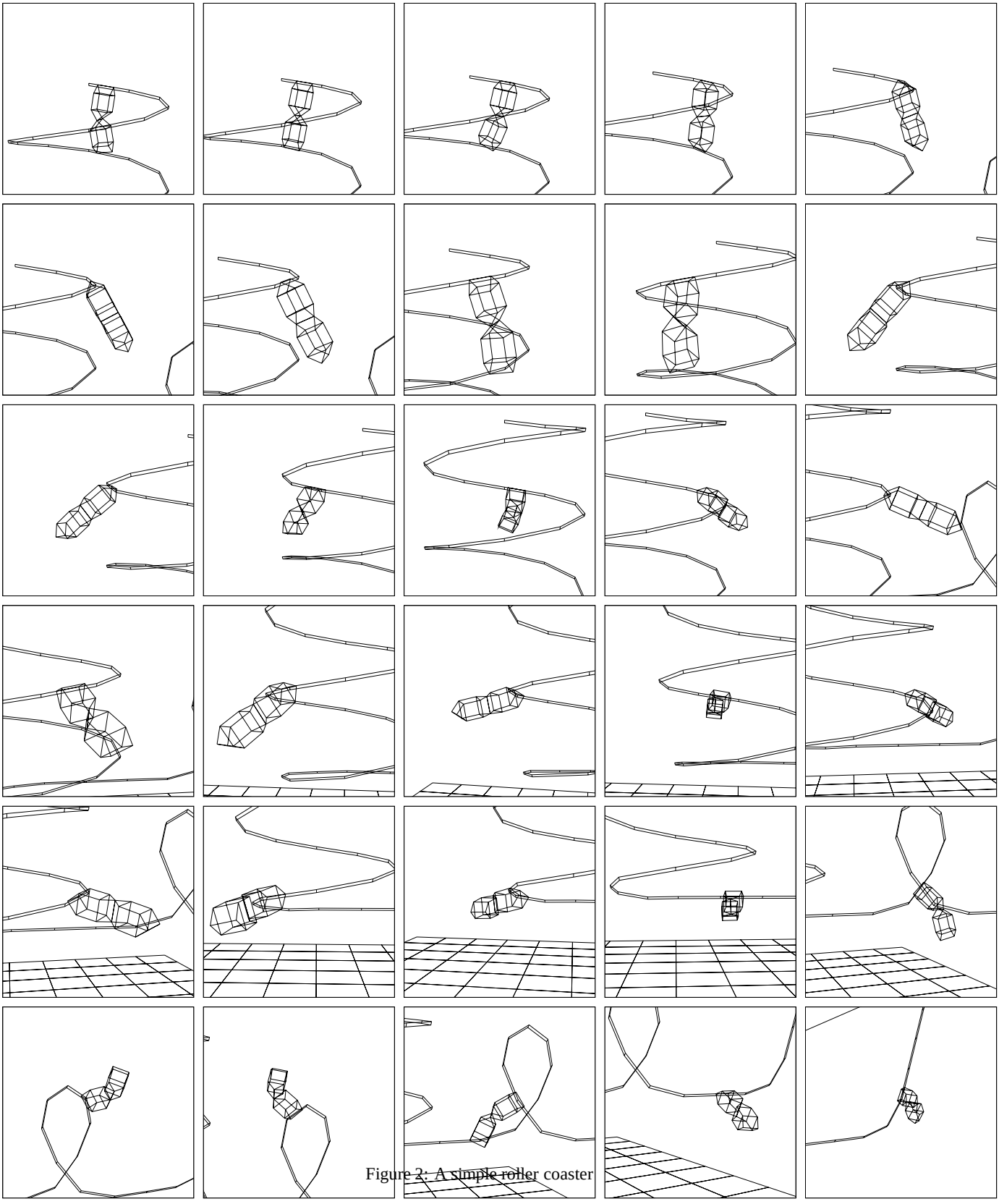

Figure 2: A simple roller coaster

torque need not to be represented in the model. This results both in a straightforward mathematical treatment and a transparent implementation. Moreover, these forces can be simply visualised for instructive purposes. Also, if we impose a threshold value on the applied PTP-forces, it is straightforward to have objects break apart if internal reaction forces become too large.

A less central ingredient is

use iteration: The iterative method with about 10 iterations gives sufficient performance to achieve a frame update rate of about 20 frames/second in on a sun SPARC station 10 for systems with several tens of constraints. This number of iterations is sufficient to preserve all rigidity constraints, PTP constraints and PTC constraints within visible accuracy (equivalent to about 10^{-2} relative accuracy) in all cases except fierce collisions where temporary deformations may be visible.

References

1. William Armstrong and Mark Green. The dynamics of articulated rigid bodies for purposes of animation. *The Visual Computer*, 1:231–240, 1985.
2. V.I. Arnold. *Mathematical Methods in Classical Mechanics* . Springer Verlag, New York, 1989.
3. B. Barenbrug. Using a constraint driven approach to dynamic simulation of rigid body movements in computer animation. Master's thesis, Eindhoven University of Technology, April 1994.
4. Ronen Barzel and Alan H. Barr. A Modeling System Based On Dynamic Constraints. *Computer Graphics (Proc. SIGGRAPH 88)*, 22(4):179–188, August 1988.
5. Paul M. Isaacs and Michael F. Cohen. Mixed methods for complex kinematic constraints in dynamic figure animation. *The Visual Computer*, 4(6):296–305, 1988.
6. Mathew Moore and Jane Wilhelms. Collision Detection and Response for Computer Animation. *Computer Graphics (Proc. SIGGRAPH 88)*, 22(5):289–298, August 1988.
7. C.W.A.M. van Overveld. The generalised display processor as an approach to real time interactive 3-D computer animation. *The Journal of Visualisation and Computer Animation*, 2:16–25, 1991.
8. C.W.A.M. van Overveld. A simple approximation to Rigid Body Dynamics for Computer Animation . *The Journal of Visualisation and Computer Animation*, 5:17–36, 1994.
9. Jane Wilhelms, Matthew Moore, and Robert Skinner. Dynamic animation: interaction and control. *The Visual Computer*, 4(6):283–295, 1988.
10. A. Witkin, M Gleicher, and W Welch. Interactive dynamics. *Computer Graphics ACM SIGGRAPH; special issue on 1990 symposium on interactive 3D graphics*, 24(2):11–21, March 1990.