Fourth Irish Workshop on Computer Graphics (2003)
Eurographics Irish Chapter

Complementarity Based Multiple Point Collision Resolution

T. Giang,† G. Bradshaw,‡ C. O'Sullivan,§

Department of Computer Science, Image Synthesis Group,
Trinity College Dublin, Dublin, Ireland

Abstract

Collision detection and response is one of the most extensively researched areas in computer graphics but yet, very little information is available on how to deal with multiple collision situations after they have been detected. In general such topics are quickly brushed over and are small parts of a much larger topic. This paper aims to break this trend and to give a thorough and hopefully informative presentation on how to effectively extract appropriate impulses for multiple point collisions using Linear Complementarity Programming techniques.

1. Introduction

The field of physically based modeling in computer graphics research over the last few decades has grown to become quite a substantial subject area. This is partially due to the ever increasing computational speeds of modern day processors, which has facilitated the growth of feasible real time processing of what once was considered quite computationally complex systems. Much research has gone into the reproduction of physically correct behavior of both rigid and non-rigid structures and many dynamics packages have been made available to the general community from such research. Cohen et al [6] and Hudson et al [12] are examples of papers describing the more popular non-commercial packages available at time of writing (these packages are actually collision handling packages, a subset of physically based modeling research). Many major computer games and motion picture companies have now started to realise the potential within the area and have started to utilise this vast research in their productions. This in turn has provided a market for many new commercial ventures which have emerged over the last few years to provide such companies with commercially viable packages.

One of the most extensively researched topics in physically based modeling has been that of collision detection [13] [5] [15] and collision/contact response [2] [1] [14]. This paper aims to address in a clear and concise manner the problem of determining viable contact impulses during multiple point collisions. This is not to be confused with the determination of contact forces which prevent interpenetration. Both approaches are analytically similar and one can be easily extended to incorporate the other. The approach we adopt is analogous to that described by Baraff [1]. However, in this paper we endeavor to provide a more comprehensive treatment of the problem. We first briefly present the basic architecture of how a collision response module fits into the simulation chain and then strive to give an inclusive description from first principles of the underlying solution formulation to the multiple collision point problem.

2. Background

There have been many papers produced in the past years that have in one way or another extensively dealt with the problem of response after a detected collision event [14] [9] [11] [17]. However, many of these papers do not give a thorough study to the problem of multiple contact collisions or else completely neglect the subject matter altogether. We only concern ourselves with the formulation of multiple collision impulses for rigid body collisions and leave flexible body collision problems for future work.

One of the earliest complete treatments of the subject of collision detection and response for physically based simulations can be found in [11] and [17]. Both papers give extensive consideration to the problem of dealing with response for single point collisions and both very briefly suggest solutions to the problem of multiple point collisions. [17] tack-

† email: Thanh.Giang@cs.tcd.ie
‡ email: Gareth.Bradshaw@cs.tcd.ie
§ email: Carol.OSullivan@cs.tcd.ie

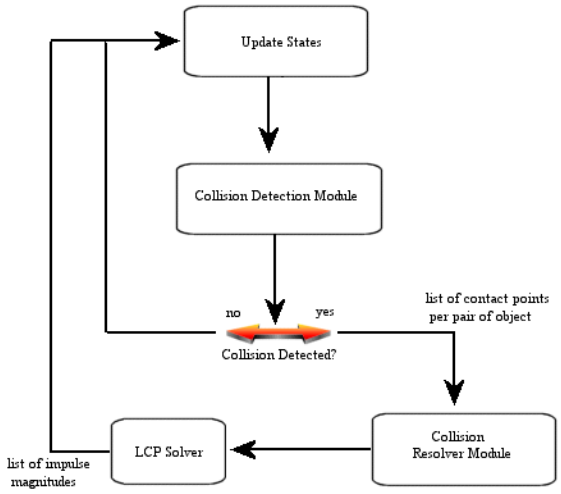

Figure 1: Broad overview of how response module architecture fits into a typical dynamics package. The resolver module is the main module which formulates the final problem description which it then passes into the LCP solver.

les this problem by *"inventing"* a series of individual non-related impulses that act exclusively on each point of contact which is then incorporated into the matrix of 15 linear equations to be solved for. [11] on the other hand suggested the solution of obtaining an impulse for each point individually and then summing up the resultant impulses that acted on the same body. This however was only suggested for situations whereby a body was in simultaneous contact with multiple other bodies at a single point per contact and the idea was not extended to account for multiple contact points for each pair of contacting bodies.

The most comprehensive treatment to date on the issue of tackling the tricky situation of response impulses for multiple point collisions can be found in [1]. The author models simultaneous collisions on each colliding body as a collection of impulses acting at each colliding point, each having an effect on how the other behaves. This method is based very much on the same non-penetration techniques presented for resolving contacting forces to prevent inter-penetration in resting contact situations, which is the main body of the paper. As such, the bulk of derivative details and assumptions for the simultaneous collision impulse section in the paper are omitted and left to the readers' own devices. This however, is far from trivial.

3. Task of the collision resolver

Within a typical dynamics simulator, the collision resolver's job is to determine the outcome of a simulation once a collision has been detected and then to feed back to the simulator the necessary data to resolve this outcome (see Figure 1). It is the job of the collision detection module in the simulator to detect any potential collisions and to determine the necessary contact/collision points along with any other necessary information needed by the resolver. This information is fed into the resolver which then goes ahead and determines the necessary data that will satisfy the final outcome for the situation at hand. Note that here we ignore all other possibilities like object interpenetration and back-stepping and presume that the collision detection module has taken care of all this for us.

The essential primary data that any collision resolver needs may be as follows:

- *the "common" point of contact*
- *a pointer to colliding object A*
- *a pointer to colliding object B*
- *the normal of contact*
- *the coefficient of restitution for this collision*

The above object pointers point to some rigid body structure whereby all necessary state information about the object can be accessed quickly and conveniently. For non-moveable objects, like walls and floors, the above pointers may point to a dummy rigid body whereby the mass of that body is infinite and the normal of contact is set to point away from the non-moveable structural object(s) (see Figure 2). Also, it may be worth noting that by convention, we consider the normal of contact to always point away from the contacting surface of object B. This is a trivial but important point.

When dealing with simultaneous collisions at multiple points on a body we may consider, for structural convenience, this data to be encapsulated within a *contact node* which itself is part of a list of all colliding points. This is dispatched from the collision detection module to the collision response module.

4. Basic collision impulse assumptions

At collision time t_c there may be two or more bodies colliding with one another. If this is the case, then we treat all bodies that are in a state of collision simultaneously and view each point of contact on the colliding bodies and the influences through those points (i.e. *impulses*) as affecting the overall outcome of the collision between that body and the body or bodies which it is colliding. So for each colliding point on the colliding body, that point has an influence on how all other colliding points on that body behave and vice versa. Figure 3 illustrates the relation between body A and B during a collision. For data structural convenience, in each *contact node* that we set up for each collision point, we only ever consider at most two bodies per point. However, in our collision equations we take into account all other possible collision points and the resultant influence produced

from those points. This will become clearer later on. If there are more than two colliding bodies per point we simply produce a separate *contact node* for every permutation of pairs of bodies that collide at that point. At the end of this process we will have a list of *contact nodes* which correspond to points of contact for each pair of colliding objects. This list, as mentioned previously is then fed into our response resolver which then solves for the corresponding *impulses* that act at each collision point. In order to get a better grasp of the final impulse equation formulations, let us initially start from first principles:

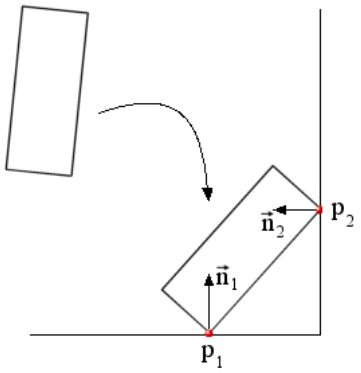

Figure 2: contact normal direction of collision between a moveable and non-moveable object at multiple points

For any pair of bodies, A and B, during a collision at time t_c, the ith point of contact on body A should be such that it is equal to the ith point of contact on body B (see Figure 3). Let p_i denote the ith point of contact at time t_c on a pair of bodies A and B so that

$$p_{A_i}(t_c) = p_{B_i}(t_c) = p_i(t_c) \tag{1}$$

whereby $p_{A_i}(t_c)$ and $p_{B_i}(t_c)$ denote the ith point of contact at collision time t_c of bodies A and B respectively.

Now let us consider the velocity of each body. At any time t, the velocity of any point on an unrestricted moving body ξ may be expressed as

$$\dot{p}_\xi(t) = v_\xi(t) + \omega_\xi(t) \times (p_\xi(t) - x_\xi(t)) \tag{2}$$

where $v_\xi(t)$ is the linear velocity of body ξ at time t, $\omega_\xi(t)$ is the angular velocity at time t and $x_\xi(t)$ is the center of mass of the object at time t. For notational convenience let us denote $(p_\xi(t) - x_\xi(t))$ as the variable $r_\xi(t)$ from now on. Also, let us represent pre- and post-collision events using the superscripts $^-$ and $^+$ respectively. So the pre-collision velocity of point i at time t_c is denoted as $\dot{p}_i^-(t_c)$ and post-collision velocity is $\dot{p}_i^+(t_c)$.

With this in mind, let us consider what happens during a collision event. According to Newton's law of restitution for frictionless collisions, at time t_c:

$$v_{rel}^+(t_c) = -\varepsilon v_{rel}^-(t_c) \tag{3}$$

v_{rel} being the *relative velocity in the normal direction* of the colliding objects which can be expanded to be

$$v_{rel} = \vec{n} \cdot (\dot{p}_A - \dot{p}_B) \tag{4}$$

and ε being what is known as the *coefficient of restitution*. This variable, measured between 1 and 0, determines how *elastic* a resultant collision is and can be viewed as how bouncy the materials from both colliding bodies are. For a perfectly elastic collision, ε = 1, and for a perfectly inelastic collision ε = 0. For a detailed proof please see [8].

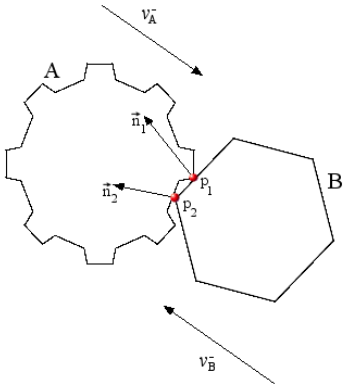

Figure 3: contacting points and normal directions during collision between two objects. Collision at p1 affects how p2 reacts and vice versa. Both points p1 and p2 are the same for both objects A and B at collision.

Before we continue any further, it may be advantageous to clarify the definition of an impulse to prevent any confusion that may arise later on. Informally speaking, the incoming velocity of a body during a collision event is perturbed by an *impulse* Ψ. This differs from constraint forces that enforce non-penetration during continuous contact modeling (i.e. resting or sliding contact). We may simply look at Ψ as a very large force acting over a very small interval of time, applied to two newly colliding objects, in such as way as to instantaneously change the approach velocities of both objects so that they each push the other apart. Impulses relate

to momentum so in fact a more formal definition of an impulse would be *the instantaneous change in momentum over a very small time period*. The direction in which Ψ acts relates to the direction of the normal of contact. Recall that we said by convention we always consider the normal of contact to be pointing away from the surface of object B. All this may be expressed as

$$\Psi = F\Delta t = M\Delta v = f\vec{n} \tag{5}$$

In the above equation, $F\Delta t$ denotes a large force acting over a very small time period, M is simply the mass of the colliding body, Δv being the change in velocity of the colliding body and f the magnitude of the resultant impulse due to the collision. Though the resultant impulse variable Ψ is a vector term, the variable f (the impulse magnitude) is an unknown scalar and is such that $f \geq 0$. This is what our resolver wishes to solve for. So, according to Newton's third law of motion, the final calculated impulse(s) should act positively on one body and equally but oppositely on the other. Let us say that Ψ acts positively on body A and equally but oppositely on body B, thus conforming to Newton's third law of motion. Thus for body A, $\Psi_A = f\vec{n}$ and for body B, $\Psi_B = -f\vec{n}$.

Now, let us deduce the change in velocity due to an applied impulse during a collision at time t_c. It is possible to use the information thus far to derive the change in velocity as an expression in terms of the resultant impulse Ψ. Thus the change in velocity, Δv, at any point on a body ξ due to an imparted impulse Ψ at time t_c is

$$\Delta v_\xi(t_c) = \frac{\Psi_\xi(t_c)}{M_\xi} + I_\xi^{-1}(t_c)\big(r_\xi(t_c) \times \Psi_\xi(t_c)\big) \times r_\xi(t_c) \tag{6}$$

Where $\frac{\Psi_\xi(t_c)}{M_\xi}$ is simply the linear component of the change and the second part, $I_\xi^{-1}(t_c)\big(r_\xi(t_c) \times \Psi_\xi(t_c)\big) \times r_\xi(t_c)$ is the angular component. The variable I^{-1} is the inverse inertia tensor of the object and all other variables are as described previously. Knowing this, the equation for the *post collision* velocity of point i on body ξ at time t_c can be expressed as

$$\dot{p}^+_{\xi_i}(t_c) = \dot{p}^-_{\xi_i}(t_c) + \Delta v_{\xi_i}(t_c) \tag{7}$$

We now have v^+_{rel} as a linear function of f.

5. Simultaneous collision points

Recall that during a collision event, $v^+_{rel}(t_c) = -\varepsilon v^-_{rel}(t_c)$. We will need to extend this assumption to fit our multiple collision point criteria. Let us first make the assumption that the coefficient of restitution, ε, relates to each collision point individually on the colliding body rather than the body as a whole. Since each colliding point on the colliding body influences the other, so too must ε by virtue of relation. So, in short, for each colliding point on the body during a collision event, that point may possess a different elasticity to any other point on that body even though it is part of the same body. Furthermore, the colliding body may be pushed away by a third (or more) body or bodies in a simultaneous, more powerful, collision thus cancelling the effect of the previous collision and breaking the rule in Equation (3). To account for this, let us change the constraint imposed by Equation (3) to one that reflects this situation better:

$$v^+_{rel}(t_c) \geq -\varepsilon v^-_{rel}(t_c) \tag{8}$$

Of course this implies that if $v^+_{rel}(t_c)$ exceeds $-\varepsilon v^-_{rel}(t_c)$, then our impulse magnitude f for that point must be zero as the initial collision contact assumption no longer holds. We can write these constraints down as

$$v^+_{rel}(t_c) + \varepsilon v^-_{rel}(t_c) \geq 0 \tag{9}$$

$$f(t_c) \geq 0 \tag{10}$$

$$f(t_c)\big(v^+_{rel}(t_c) + \varepsilon v^-_{rel}(t_c)\big) = 0 \tag{11}$$

From this point onwards, for the sake of clarity, let us drop the t_c variable from our equations as they are going to get quite messy. Instead, it will be assumed that all equations from here onwards refer to time t_c.

Back to the problem at hand. The above three constraints, (9), (10), (11) formulate what is referred to as a *Linear Complementarity Problem*. More formally, an LCP can be stated as:

Given a known vector $\mathbf{b} \in R^n$ *and a known matrix* $\mathbf{A} \in R^{n \times n}$ *the problem is to find a vector* $\mathbf{f} \in R^n$ *such that*

$$\mathbf{w = Af + b} \geq 0 \tag{12}$$

$$\mathbf{f} \geq 0 \tag{13}$$

$$\mathbf{f}^T(\mathbf{w}) = 0 \tag{14}$$

or to show that no such vector $\mathbf{f}$ *exists.*

Cottle et al. [7] and Murty [16] both give excellent detailed explanations of LCPs and various solution methods.†

† The latter book is unfortunately out of print but is however, freely available in electronic form on the web

Our solver's job is to take constraints (9), (10), (11) and formulate the appropriate A matrix and b vector, giving them to the LCP solver to thus finally solve for vector f, the list of required impulse magnitudes. Let us now extend the *post collision velocity* equation (7) so that it accounts for simultaneous multiple colliding points. Recall that we said each impulse acting on each colliding point on a colliding body will influence all other colliding points on that same body. This must be taken into account in the updated equation formulation. So updating (7) will gives us

$$\dot{p}^+_{\xi_i} = \dot{p}^-_{\xi_i} + \sum_{j=1}^{n} \frac{f_{\xi_j}\vec{n}_j}{M_\xi} + I_\xi^{-1} \sum_{j=1}^{n} (r_{\xi_j} \times f_{\xi_j}\vec{n}_j) \times r_{\xi_i} \quad (15)$$

If we substitute equation (15) into equation (4), the resultant formulation describes the relative velocity at point i in the normal direction after the collision, taking account of all other influences acting through all other contacting points on the colliding objects.

$$v^+_{rel_i} = v^-_{rel_i} + \vec{n}_i \cdot \left(\sum_{j=1}^{n} \frac{f_j\vec{n}_j}{M_{A_i}} + \lambda_{A_i} \sum_{j=1}^{n} (r^*_{A_j} f_j \vec{n}_j) - \sum_{j=1}^{n} \frac{-f_j\vec{n}_j}{M_{B_i}} - \lambda_{B_i} \sum_{j=1}^{n} \left(r^*_{B_j}(-f_j\vec{n}_j)\right) \right) \quad (16)$$

where $\lambda_{A_i} = r^{*T}_{A_i} I^{-1}_{A_i}$, $\lambda_{B_i} = r^{*T}_{B_i} I^{-1}_{B_i}$, $f_{A_j} = f_{B_j} = f_j$ and $v^-_{rel_i}$ is as in Equation (4). Note that we negate the impulse magnitude f acting on object B in the above formulation so as to conform to Newton's third law of motion. To neglect to do so would give us an improper model of the resultant impulses acting on each object.

The above equation (16) has been rearranged for notational convenience and clarity. The reader should have no problem in seeing that it was got by simple substitution. Perhaps the only confusing thing is the variables with the * superscript. Barzel et al [3] name this *"transformation"* as the *dual of a vector* and hence we follow suit here. This formulation is just a convenient way for us to express a cross product of two vectors as a matrix vector multiplication. Equation (17) below might perhaps give a better picture of what is happening. The appendix B in [3], as well as giving a brief description of vector dual properties, also gives a very good overview of point behavior on a rigid body which is highly applicable to the equation formulations presented in this paper. The reader may want to refer to chapter 4 of Goldstein [10] for the proof of why this vector cross product duality relation holds.

$$a \times b = a^* b = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad (17)$$

We now take the above equation (16) and substitute for the left hand side of equation (3). This brings us one step closer to obtaining our required b vector and A matrix in our LCP problem. The last thing for us to do is to bring over the right hand side of equation (3) and further rearrange to give:

$$b_i + \sum_{j=1}^{n} f_j A_{ij} \geq 0 \quad (18)$$

where

$$b_i = v^-_{rel_i}(1 + \varepsilon_i) \quad (19)$$

and

$$A_{ij} = \vec{n}_i \cdot \left(\left(\frac{\vec{n}_j}{M_{A_i}} + \lambda_{A_i}(r^*_{A_j}\vec{n}_j) \right) - \left(\frac{-\vec{n}_j}{M_{B_i}} + \lambda_{B_i}\left(r^*_{B_j}(-\vec{n}_j)\right) \right) \right) \quad (20)$$

Each element ij in the A matrix can be got from the above by straightforward substitution.

The above Equation (20) may still seem a bit overwhelming so let us further simplify the equation by further variable substitution. Let

$A_{A_{ij}} = \frac{\vec{n}_j}{M_{A_i}} + \lambda_{A_i} r^*_{A_j}\vec{n}_j$
$A_{B_{ij}} = \frac{-\vec{n}_j}{M_{B_i}} + \lambda_{B_i} r^*_{B_j}(-\vec{n}_j)$

so that equation (18) can now be expressed as:

$$b_i + \sum_{j=1}^{n} f_j \vec{n}_i \cdot \left(A_{A_{ij}} - A_{B_{ij}}\right) \geq 0 = w_i \quad (21)$$

We now have all the information we need to be able to solve for the impulse magnitude f. The collision resolver module, given the appropriate list of collision nodes, will use the above formulae to determine the appropriate constraint formulation to give to the LCP solver. The LCP solver itself in turn will then solve for vector f, the list of appropriate impulse magnitudes, which it can either return back to the resolver to pass on to the main update module in the simulator or return the information to the main update module directly. If the resolver was given only one collision point, then the response problem simply reduces down to one linear equation in one unknown which can be solved very efficiently and mirrors the more traditional single point impulse resolution methods. The impulse magnitudes calculated through the resolver will then be applied as appropriate to the states of the rigid bodies in the simulation so as to change their momentum to reflect a collision (remember, impulses are related to momentum).

6. Conclusions and Future Work

We have set up many experimental animations using the LCP resolution method for simultaneous collisions as described in this paper and the results to date have been most favorable. Physical plausibility has been maintained, if not improved, for all produced animations. Figure 4 shows stills from an animation of two Stanford bunnies‡ falling onto a ramp using the simultaneous multiple collision point resolution method as described in this paper. Figure 5 shows the same animation but this time with considered collision points shown. We use a sphere-tree collision detection model [9] [4] so the number of collision points at each narrow phase collision is vast (as can be seen by yellow points) but however is reduced down to at most an approximate 4 points (these are marked in red). Informal tests have shown that collision response done on multiple collision points simultaneously rather than on a per point basis in general gives a more pleasing outcome.

Linear Complementarity Programming techniques have been utilised as a solution method in resolving simultaneous contact forces to enforce non-interpenetration constraints [1] [2]. However, very little has been seen of them (to our knowledge) when it comes to utilising them as a solution to resolving simultaneous collision impulses. Baraff's 1989 paper [1] has been the only paper which we are aware of to have touched on the subject.

For all the technique's merits however, the necessity of a specialised linear equation solver along with the seemingly complex maths involved may explain why LCP based resolution methods for simultaneous collision impulse responses have not been explored or mentioned more in the past. It is our hope that people with little maths background will be able to take what we have presented here and just simply "plug" each variable in the final formulation into their code or project the necessary data to enable them to account for more than one contact point in a collision. At the very least, we hope that this paper has improved their understanding of how to deal with impulses for multiple collision points.

We are currently in the process of integrating this technique into a dynamics framework and hope in the future to research further into the possibility of speeding up the method by perhaps making the simulation as well as the collision detection time critical.

References

1. David Baraff. "Analytical methods for dynamic simulation of non-penetrating rigid bodies". *ACM SIGGRAPH*, pp 223–232, 1989.
2. David Baraff. "Non-penetrating rigid body simulation." *State of the Art Reports of EUROGRAPHICS'93, Eurographics Technical Report Series*, 1993.
3. Ronen Barzel and Alan H. Barr. "A modeling system based on dynamic constraints." *ACM SIGGRAPH*, pp 179–188, 1988.
4. Gareth Bradshaw and Carol O'Sullivan. "Sphere-tree construction using medial-axis approximation." *Proceedings of the ACM SIGGRAPH Symposium on Computer Animation SCA 2002. pp 33-40*, 2002.
5. Stephen Cameron. "Enhancing gjk: computing minimum and penetration distances between convex polyhedra." *Int Conf Robotics and Automation*, 1997.
6. Jonathan D. Cohen, Ming C. Lin, Dinesh Manocha, and Madhav K. Ponamgi. "I-COLLIDE: An interactive and exact collision detection system for large-scale environments." *Symposium on Interactive 3D Graphics*, pp 189–196, 1995.
7. Richard W. Cottle, Jong-Shi Pang, and Richard E. Stone. *The Linear Complementarity Problem*. Academic Press, Inc., 1992.
8. Edward A. Desloge. *Classical Mechanics*, volume 1. Wiley-Interscience, 1982.
9. John Dingliana and Carol O'Sullivan. "Graceful degradation of collision handling in physically based animation." *Computer Graphics Forum (EUROGRAPHICS 2000 Proceedings)*, pp 239–247, 2000.
10. Herbert Goldstein. *Classical Mechanics*. Addison-Wesley, Inc., 1971.
11. James K. Hahn. "Realistic animation of rigid bodies." In *ACM SIGGRAPH*, pp 299–308, 1988.
12. Thomas C. Hudson, Ming C. Lin, Jonathan Cohen, Stefan Gottschalk, and Dinesh Manocha. "V-collide: Accelerated collision detection for VRML." *VRML 97: Second Symposium on the Virtual Reality Modeling Language*, 1997.
13. Ming C Lin. *Efficient Collision Detection For Animation and Robotics*. PhD thesis, 1994.
14. Brian Mirtich and John F. Canny. "Impulse-based simulation of rigid bodies." *Symposium on Interactive 3D Graphics*, pp 181–188, 1995.
15. Brian Mirtich. "V-clip: fast and robust polyhedral collision detection." Technical Report TR-97-05, Mitsubishi Electric Research Laboratory, 1997.
16. Katta G. Murty. *Linear Complementarity, Linear and Nonlinear Programming*. Helderman-Verlag, 1988.
17. Mathew Moore and Jane Wilhelms. "Collision detection and response for computer animation." In *ACM SIGGRAPH*, volume 22, pp 289–298, 1988.

‡ The Stanford bunny model was obtained from Stanford's 3D scanning repository : http://graphics.stanford.edu/data/3Dscanrep/

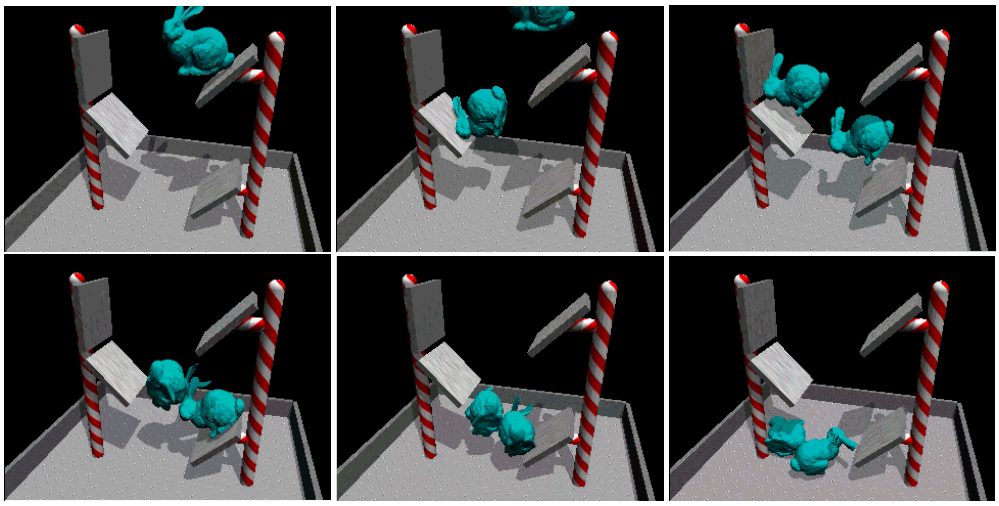

Figure 4: Stanford bunnies falling onto ramps

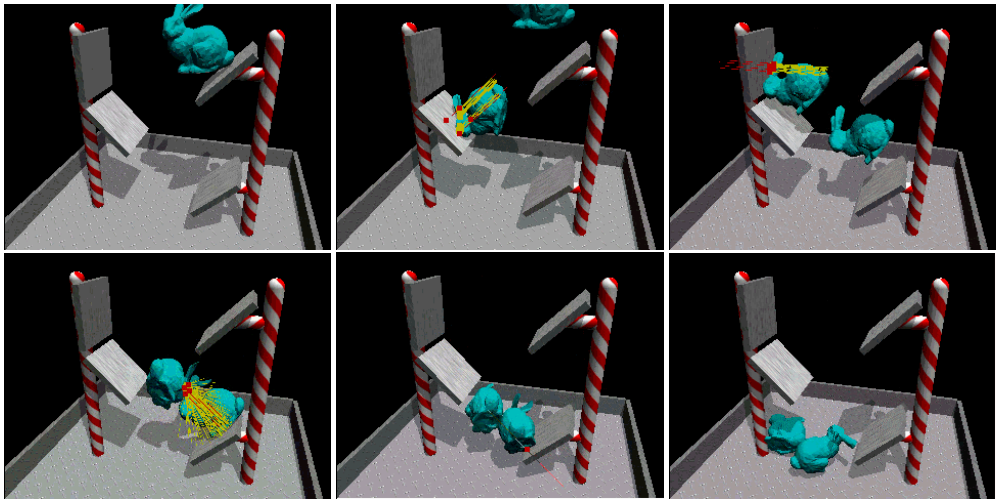

Figure 5: Stanford bunnies falling onto ramps with considered collision points shown