Fast Implementation of Lemke's Algorithm for Rigid Body Contact Simulation

John E. Lloyd
Computer Science Dept., University of British Columbia
Vancouver, Canada   http://www.cs.ubc.ca/spider/lloyd

***Abstract*— We present a fast method for solving rigid body contact problems with friction, based on optimizations incorporated into Lemke's algorithm for solving linear complementarity problems. These optimizations improve computation time in general and reduce the expected solution complexity from $O(n^3)$ to nearly $O(nm + m^3)$, where n and m are the number of contacts and rigid bodies. For a fixed number of bodies the expected complexity is therefore close to $O(n)$. Our method also improves numerical robustness, and removes the need to explicitly compute the large matrices associated with rigid body contact problems.**

***Index Terms*— Haptics, contact, simulation, virtual reality**

I. Introduction

Computing the reaction forces that arise between rigid bodies in contact is important in robotics, computer graphics, animation, haptics, and mechanical simulation.

This calculation is generally expensive whenever there are multiple contacts, which can occur even in simple situations involving extended contact between two bodies (Fig. 1). For example, contact between two polygonal faces can be modeled by placing contacts at the vertices of the convex hull of their intersection (Fig. 2), but even for two squares this may produce up to eight contacts. Some of these contacts may be redundant, but eliminating redundant contacts in advance involves calculations equivalent to solving the contact problem itself.

Rigid body contact can be formulated as a linear complementarity problem (or *LCP*, described below), which can then be solved by either indirect (iterative) methods, or direct (pivoting) methods [1], [2]. Iterative methods include impulse-based techniques [3], [4], and are useful for handling collisions and producing visually plausible behaviors for large numbers of objects, but may suffer from low accuracy or slow convergence [5]. Convergence is also hard to prove, particularly with friction, which makes the associated linear system asymmetric. For direct methods, the only one which has been proven to work in the presence of friction [6], [7] is Lemke's algorithm, which produces an exact answer but has a nominal expected time complexity of $O(n^3)$ in the number of contacts. Lemke's algorithm is also numerically sensitive, particularly in the presence of redundant contacts, and requires working with large matrices: for example, the LCP for a problem with 16 contacts and an 8-sided polygonal friction cone has a matrix size of 160.

We present here an implementation of Lemke's algorithm for contact problems that removes some of these difficulties. First, the method is fast: in tests described

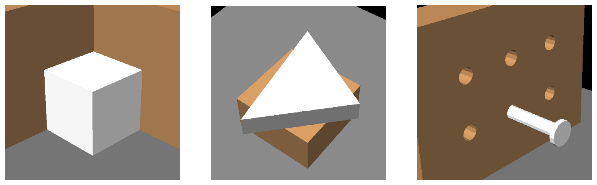
Fig. 1. Extended contacts with many contact points. From left: block-in-corner (12 contacts), face-on-face (6 contacts), peg-in-hole (arbitrary).

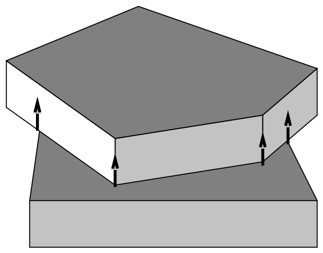
Fig. 2. Contacts (arrows) modeling an extended face-on-face contact.

below, the average solution time for a 16 contact peg-in-hole problem was reduced from 30 msec to 2 msec. The expected complexity also appears to reduce from $O(n^3)$ to nearly $O(nm + m^3)$, where n and m are (respectively) the number of contacts and rigid bodies; for fixed m this becomes $O(n)$. This increases the practicality of rigid body simulation for real-time interactive applications, such as haptics. Furthermore, there is no need to explicitly calculate the (large) LCP matrix, and numerical robustness is improved because of internal reductions in problem size. Our method has been coded in Java and is available from http://www.cs.ubc.ca/spider/lloyd/fastContact.html.

Sections II, III, and IV give background on contact problems, the solution of LCPs by pivoting methods, and Lemke's algorithm. Section V will show how the contact problem's structure can be used to reduce the complexity of the Lemke calculations, and Section VI will show how to streamline things further by reducing the size of the LCP that needs to be solved. Performance tests for these methods are described in Section VII.

II. Problem Formulation

Rigid body contact has been described in many publications (e.g., [8], [9], [10], [11], [7]) and so will only be summarized here.

The dynamics equation for a rigid body system with n

contact constraints and m bodies can be expressed as

$$\mathcal{M}\dot{\mathbf{v}} - \tilde{\mathbf{N}}\boldsymbol{\theta} - \tilde{\mathbf{D}}\boldsymbol{\phi} = \mathbf{f}_x \qquad (1)$$

where $\mathcal{M} \in \mathbb{R}^{6m\times 6m}$ is the block-diagonal system mass matrix containing the mass matrices $\mathcal{M}_k \in \mathbb{R}^{6\times 6}$ for each body, $\mathbf{v}$, $\mathbf{f}_x \in \mathbb{R}^{6m}$ describe the spatial velocities and external forces (including coriolis forces) for each body, $\tilde{\mathbf{N}}$ and $\tilde{\mathbf{D}}$ are the constraint matrices for normal forces and friction, and $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are the normal and friction force components that act along these constraints. Bilateral constraints can also be included in this formulation, but we will omit this for brevity.

Each contact i is associated with one column of $\tilde{\mathbf{N}}$ and d columns of $\tilde{\mathbf{D}}$, corresponding to the contact normal $\hat{\mathbf{n}}_i$ and d directions spanning the plane perpendicular to $\hat{\mathbf{n}}_i$ which form a polyhedral approximation to the friction cone[1] (see [12], [6], [9]). Because each contact affects at most two bodies, each column of $\tilde{\mathbf{N}}$ and $\tilde{\mathbf{D}}$ has at most 12 non-zero entries. Size-wise, we have $\tilde{\mathbf{N}} \in \mathbb{R}^{6m\times n}$ and $\tilde{\mathbf{D}} \in \mathbb{R}^{6m\times nd}$.

It is common to combine the solution of (1) with integration over a time step h, in order to ensure the existence of solutions [9], [11]. Using an explicit Euler step with $\mathbf{v}_0$ denoting the initial velocities, we get

$$\mathbf{v} = \mathcal{M}^{-1}(\tilde{\mathbf{N}}\boldsymbol{\theta} + \tilde{\mathbf{D}}\boldsymbol{\phi} + \tilde{\mathbf{k}}_x) \qquad (2)$$

where $\tilde{\mathbf{k}}_x \equiv \mathbf{f}_x h + \mathcal{M}\mathbf{v}_0$ and $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are now impulse components.

Although (2) will be used in the sequel, other formulations (using $\dot{\mathbf{v}}$, different integration schemes [7], or first order physics) all result in an LCP with the same structure as system (6) below and so may benefit from the results of this paper. System (6) can also be modified to incorporate numeric stabilization [13] and restitution for elastic collisions [11], [9].

The computation of $\mathbf{v}$ is also subject to constraints. Quickly summarized, these are[2]

$$\boldsymbol{\nu} \equiv \tilde{\mathbf{N}}^T\mathbf{v} \ge 0,\ \boldsymbol{\nu}^T\boldsymbol{\theta} = 0,\ \boldsymbol{\theta} \ge 0, \qquad (3)$$
$$\boldsymbol{\gamma} \equiv \boldsymbol{\mu}\boldsymbol{\theta} - \mathbf{E}^T\boldsymbol{\phi} \ge 0,\ \boldsymbol{\gamma}^T\boldsymbol{\lambda} = 0,\ \boldsymbol{\lambda} \ge 0, \qquad (4)$$
$$\boldsymbol{\sigma} \equiv \tilde{\mathbf{D}}^T\mathbf{v} + \mathbf{E}\boldsymbol{\lambda} \ge 0,\ \boldsymbol{\sigma}^T\boldsymbol{\phi} = 0,\ \boldsymbol{\phi} \ge 0, \qquad (5)$$

where $\boldsymbol{\lambda} \equiv (\lambda_1, \dots, \lambda_n)$ is a vector of Lagrange multipliers, $\boldsymbol{\mu} \in \mathbb{R}^{n\times n}$ is a diagonal matrix of friction coefficients, and $\mathbf{E} \in \mathbb{R}^{nd\times n}$ with the form

$$\mathbf{E} = \begin{pmatrix} \mathbf{1} & & \\ & \ddots & \\ & & \mathbf{1} \end{pmatrix},\ \mathbf{1} \in \mathbb{R}^d,\ \mathbf{1} = (1, \dots, 1)^T.$$

Respectively, these enforce non-penetration at the contact (3), friction cone constraints (4), and keeping friction as opposite as possible to tangential velocity (5).

To simplify notation in the remainder of this paper, we use the fact that $\mathcal{M}$ is symmetric positive definite and so can be factored as $\mathcal{M} = GG^T$. If we then define

$$\mathbf{N} \equiv G^{-1}\tilde{\mathbf{N}}, \quad \mathbf{D} \equiv G^{-1}\tilde{\mathbf{D}}, \quad \mathbf{k}_x \equiv G^T\tilde{\mathbf{k}}_x,$$

[1]Larger d creates a more accurate friction cone approximation.

[2]Relational operators on vectors imply element-wise satisfaction.

we can form (2)-(5) into a single constrained system:

$$\begin{pmatrix} \boldsymbol{\nu} \\ \boldsymbol{\sigma} \\ \boldsymbol{\gamma} \end{pmatrix} = \begin{pmatrix} \mathbf{N}^T\mathbf{N} & \mathbf{N}^T\mathbf{D} & 0 \\ \mathbf{D}^T\mathbf{N} & \mathbf{D}^T\mathbf{D} & \mathbf{E} \\ \boldsymbol{\mu} & -\mathbf{E}^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\phi} \\ \boldsymbol{\lambda} \end{pmatrix} + \begin{pmatrix} \mathbf{N}^T\mathbf{k}_x \\ \mathbf{D}^T\mathbf{k}_x \\ 0 \end{pmatrix}, \qquad (6)$$
$$\boldsymbol{\nu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\lambda} \ge 0,\ \boldsymbol{\nu}^T\boldsymbol{\theta} = \boldsymbol{\sigma}^T\boldsymbol{\phi} = \boldsymbol{\gamma}^T\boldsymbol{\lambda} = 0.$$

Solving this for $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ gives the reaction forces (or impulses) from which $\mathbf{v}$ can then be determined via (2).

System (6) is an example of a *linear complementarity problem* (LCP). It has been shown [6], [9], [7] that the contact LCP can always be solved by a pivoting technique known as Lemke's algorithm.

III. Solution of LCPs by Pivoting Methods

Formally stated, an LCP is defined by a square matrix $\mathbf{M}$ and an associated vector $\mathbf{q}$, and entails solving

$$\mathbf{w} = \mathbf{M}\mathbf{z} + \mathbf{q} \qquad (7)$$

for the variables $\mathbf{w}$ and $\mathbf{z}$, subject to the constraints

$$\mathbf{w} \ge 0,\ \mathbf{z} \ge 0,\ \mathbf{w}^T\mathbf{z} = 0.$$

If $\mathbf{q} \ge 0$ then a solution is immediately given by $\mathbf{z} = 0$ and $\mathbf{w} = \mathbf{q}$. Otherwise, one can search for a solution by setting other combinations of $\mathbf{z}$ and $\mathbf{w}$ variables to zero. In particular, let $\mathbf{z}_\alpha$ and $\mathbf{w}_{\tilde{\alpha}}$ be equally-sized subsets of the variables $\mathbf{z}$ and $\mathbf{w}$, formed from the index sets α and $\tilde{\alpha}$, and let the remaining variables by given by $\mathbf{z}_\beta$ and $\mathbf{w}_{\tilde{\beta}}$ (with index sets β and $\tilde{\beta}$). Under a suitable row/column rearrangement, (7) can then be partitioned as

$$\begin{pmatrix} \mathbf{w}_{\tilde{\alpha}} \\ \mathbf{w}_{\tilde{\beta}} \end{pmatrix} = \begin{pmatrix} \mathbf{M}_{\tilde{\alpha}\alpha} & \mathbf{M}_{\tilde{\alpha}\beta} \\ \mathbf{M}_{\tilde{\beta}\alpha} & \mathbf{M}_{\tilde{\beta}\beta} \end{pmatrix} \begin{pmatrix} \mathbf{z}_\alpha \\ \mathbf{z}_\beta \end{pmatrix} + \begin{pmatrix} \mathbf{q}_{\tilde{\alpha}} \\ \mathbf{q}_{\tilde{\beta}} \end{pmatrix}. \qquad (8)$$

If $\mathbf{M}_{\tilde{\alpha}\alpha}$ is non-singular, we can exchange $\mathbf{w}_{\tilde{\alpha}}$ and $\mathbf{z}_\alpha$ to get a *pivoted* system

$$\begin{pmatrix} \mathbf{z}_\alpha \\ \mathbf{w}_{\tilde{\beta}} \end{pmatrix} = \mathbf{M}' \begin{pmatrix} \mathbf{w}_{\tilde{\alpha}} \\ \mathbf{z}_\beta \end{pmatrix} + \mathbf{q}', \qquad (9)$$

where

$$\mathbf{M}' = \begin{pmatrix} \mathbf{M}_{\tilde{\alpha}\alpha}^{-1} & -\mathbf{M}_{\tilde{\alpha}\alpha}^{-1}\mathbf{M}_{\tilde{\alpha}\beta} \\ \mathbf{M}_{\tilde{\beta}\alpha}\mathbf{M}_{\tilde{\alpha}\alpha}^{-1} & \mathbf{M}_{\tilde{\beta}\beta} - \mathbf{M}_{\tilde{\beta}\alpha}\mathbf{M}_{\tilde{\alpha}\alpha}^{-1}\mathbf{M}_{\tilde{\alpha}\beta} \end{pmatrix},$$
$$\mathbf{q}' = \begin{pmatrix} \mathbf{q}'_{\tilde{\alpha}} \\ \mathbf{q}'_{\tilde{\beta}} \end{pmatrix} = \begin{pmatrix} -\mathbf{M}_{\tilde{\alpha}\alpha}^{-1}\mathbf{q}_{\tilde{\alpha}} \\ \mathbf{q}_{\tilde{\beta}} - \mathbf{M}_{\tilde{\beta}\alpha}\mathbf{M}_{\tilde{\alpha}\alpha}^{-1}\mathbf{q}_{\tilde{\alpha}} \end{pmatrix}. \qquad (10)$$

The variable sets $\{\mathbf{z}_\alpha, \mathbf{w}_{\tilde{\beta}}\}$ and $\{\mathbf{w}_{\tilde{\alpha}}, \mathbf{z}_\beta\}$ are called the *basic* and *non-basic* variables, respectively, and the set of basic variables defines a *basis* for the pivoted system. $\mathbf{w}$ and $\mathbf{z}$ variables whose indices match are called *complementary*, with each being the *complement* of the other. If the index sets α and $\tilde{\alpha}$ are identical, then $\mathbf{z}_\alpha$ is complementary to $\mathbf{w}_{\tilde{\alpha}}$ and the basis is a *complementary basis*.

If the basis for the pivoted system (9) is complementary and $\mathbf{q}' \ge 0$, then we have a solution to the LCP given by $\mathbf{z}_\alpha = \mathbf{q}'_{\tilde{\alpha}}$, $\mathbf{w}_{\tilde{\beta}} = \mathbf{q}'_{\tilde{\beta}}$, and $\mathbf{w}_{\tilde{\alpha}} = \mathbf{z}_\beta = 0$.

Generally speaking, a pivoting method solves an LCP by incrementally exchanging, or *pivoting*, variables (usually one pair at a time) in order to find a complementary basis for which $\mathbf{q}' \ge 0$.

IV. Lemke's Algorithm

Lemke's algorithm [1], [2] is a pivoting method where the search is facilitated by augmenting the LCP (7) with an auxiliary variable z_0 and a *covering vector* $\mathbf{c} \geq 0$:

$$\mathbf{w} = \bar{\mathbf{M}} \begin{pmatrix} \mathbf{z} \\ z_0 \end{pmatrix} + \mathbf{q}, \quad \text{with} \quad \bar{\mathbf{M}} \equiv \begin{pmatrix} \mathbf{M} & \mathbf{c} \end{pmatrix}$$

This augmented system can be partitioned and pivoted exactly as shown in (8) and (9), with $\mathbf{c}$ now included in the appropriate partitions $\bar{\mathbf{M}}_{\tilde{\alpha}\alpha}$, $\bar{\mathbf{M}}_{\tilde{\beta}\alpha}$, etc.

The algorithm works as follows:

- **Step 0.** If $\mathbf{q} \geq 0$, stop; $\mathbf{z} = 0$ solves the system. Otherwise, choose variable w_r in $\mathbf{w}$ for which $r = \arg\min\{q_i/c_i\}$, and pivot z_0 with w_r. Set the *driving variable* y_r to z_r.
- **Step 1.** Let $\mathbf{m}'$ be the column of the pivoted matrix $\bar{\mathbf{M}}'$ corresponding to y_r. If $\mathbf{m}' \geq 0$, stop: the LCP has no solution or is unsolvable by Lemke's algorithm. Otherwise, let y_s be the (basic) variable indexed by
$$s = \arg\min\{-q'_i/m'_i : m'_i < 0\} \tag{11}$$
- **Step 2.** If $y_s = z_0$, pivot z_0 with y_r and stop: the resulting $\mathbf{q}'$ solves the LCP. Otherwise, pivot y_s with y_r, set the new driving variable y_r to the complement of y_s, and return to step 1.

If test (11) results in a tie, the LCP is *degenerate*, and additional columns of $\bar{\mathbf{M}}'$ may be needed to resolve the tie; see section 4.9 in [1] or 2.2.7 of [2].

Generally, each pivot step requires computing $\mathbf{q}'$ and a single column $\mathbf{m}'$ from $\bar{\mathbf{M}}'$, using the formulae (10). $\mathbf{q}'$ takes the recursive form

$$\mathbf{q}' = \begin{pmatrix} \mathbf{q}'_{\tilde{\alpha}} \\ \mathbf{q}'_{\tilde{\beta}} \end{pmatrix} = \begin{pmatrix} -\bar{\mathbf{M}}_{\tilde{\alpha}\alpha}^{-1} \mathbf{q}_{\tilde{\alpha}} \\ \mathbf{q}_{\tilde{\beta}} + \bar{\mathbf{M}}_{\tilde{\beta}\alpha} \mathbf{q}'_{\tilde{\alpha}} \end{pmatrix}. \tag{12}$$

For $\mathbf{m}'$, if the driving variable y_r is a $\mathbf{z}$ variable, then

$$\mathbf{m}' = \begin{pmatrix} \mathbf{m}'_{\tilde{\alpha}} \\ \mathbf{m}'_{\tilde{\beta}} \end{pmatrix} = \begin{pmatrix} -\bar{\mathbf{M}}_{\tilde{\alpha}\alpha}^{-1} \mathbf{m}_{\tilde{\alpha}} \\ \mathbf{m}_{\tilde{\beta}} + \bar{\mathbf{M}}_{\tilde{\beta}\alpha} \mathbf{m}'_{\tilde{\alpha}} \end{pmatrix} \tag{13}$$

where $\mathbf{m}_{\tilde{\alpha}}$ and $\mathbf{m}_{\tilde{\beta}}$ are the $\tilde{\alpha}$ and $\tilde{\beta}$ partitions of $\mathbf{m}_{\tilde{\alpha}}$, which is the column of $\bar{\mathbf{M}}$ corresponding to y_r. Otherwise, if y_r is a $\mathbf{w}$ variable,

$$\mathbf{m}' = \begin{pmatrix} \mathbf{m}'_{\tilde{\alpha}} \\ \mathbf{m}'_{\tilde{\beta}} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{M}}_{\tilde{\alpha}\alpha}^{-1} \mathbf{e}_r \\ \bar{\mathbf{M}}_{\tilde{\beta}\alpha} \mathbf{m}'_{\tilde{\alpha}} \end{pmatrix} \tag{14}$$

where $\mathbf{e}_r$ is the $\tilde{\alpha}$ partition of the r-th column of the identity matrix.

Calculations (12)-(14) each entail solving a system

$$\bar{\mathbf{M}}_{\tilde{\alpha}\alpha} \mathbf{x} = \mathbf{b} \tag{15}$$

where $\mathbf{b}$ is $\mathbf{q}_{\tilde{\alpha}}$, $\mathbf{m}_{\tilde{\alpha}}$, or $\mathbf{e}_r$. If $\bar{\mathbf{M}}_{\tilde{\alpha}\alpha}$ is available in factored form, this, plus the other calculations in (12)-(14), can be done in $O(n^2)$ time. Each pivot makes a rank-1 change to $\bar{\mathbf{M}}_{\tilde{\alpha}\alpha}$, and so its factorization can then be updated in $O(n^2)$ time [1], [14] (although this must be done carefully to limit numerical errors [15], [16]). Then, since Lemke's algorithm typically requires $O(n)$ pivots [1], its overall expected complexity is $O(n^3)$. It should be noted, however, that pathological problems exist for which Lemke's algorithm has a worst-case complexity of $O(2^n)$ ([2], Chapter 6).

V. Simplification using Matrix Structure

We now show how the structure of (6) allows us to compute $\mathbf{q}'$ and $\mathbf{m}'$ efficiently and without having to explicitly form the LCP matrix.

To simplify the presentation, we will ignore the covering vector $\mathbf{c}$ (so that $\bar{\mathbf{M}} = \mathbf{M}$), and assume a complementary basis (so that $\tilde{\alpha} = \alpha$ and $\tilde{\beta} = \beta$). Full details on the complete calculations required by Lemke's algorithm, which must consider the covering vector, are given in the appendix of [12].

A. Simplification for Complementary Bases

Comparing (7) with (6), we see that $\mathbf{M}$ and $\mathbf{q}$ have a size of $n(2+d)$, with $\mathbf{w}^T = \begin{pmatrix} \boldsymbol{\nu}^T \boldsymbol{\sigma}^T \boldsymbol{\gamma}^T \end{pmatrix}$ and $\mathbf{z}^T = \begin{pmatrix} \boldsymbol{\theta}^T \boldsymbol{\phi}^T \boldsymbol{\lambda}^T \end{pmatrix}$.

With respect to the partitioned system (8), denote the basic $\mathbf{z}$ variables $\mathbf{z}_\alpha$ by $\boldsymbol{\theta}_\alpha$, $\boldsymbol{\phi}_\alpha$, and $\boldsymbol{\lambda}_\alpha$, and let $\mathbf{N}_\alpha$ and $\mathbf{D}_\alpha$ be the submatrices of $\mathbf{N}$ and $\mathbf{D}$ corresponding to $\boldsymbol{\theta}_\alpha$ and $\boldsymbol{\phi}_\alpha$. Because we are assuming a complementary basis, we have $\bar{\mathbf{M}}_{\tilde{\alpha}\alpha} = \mathbf{M}_{\alpha\alpha}$, and if we let $\begin{pmatrix} \boldsymbol{\theta}_\alpha^T \boldsymbol{\phi}_\alpha^T \boldsymbol{\lambda}_\alpha^T \end{pmatrix}$ correspond to $\mathbf{x}$ in (15), this relation takes the form

$$\begin{pmatrix} \mathbf{N}_\alpha^T \mathbf{N}_\alpha & \mathbf{N}_\alpha^T \mathbf{D}_\alpha & 0 \\ \mathbf{D}_\alpha^T \mathbf{N}_\alpha & \mathbf{D}_\alpha^T \mathbf{D}_\alpha & \mathbf{E}_{\alpha\alpha} \\ \boldsymbol{\mu}_{\alpha\alpha} & -\mathbf{E}_{\alpha\alpha}^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}_\alpha \\ \boldsymbol{\phi}_\alpha \\ \boldsymbol{\lambda}_\alpha \end{pmatrix} = \mathbf{b} \tag{16}$$

where $\boldsymbol{\mu}_{\alpha\alpha}$ and $\mathbf{E}_{\alpha\alpha}$ are α-indexed submatrices of $\boldsymbol{\mu}$ and $\mathbf{E}$.

Since each column of $\mathbf{E}_{\alpha\alpha}$ has at least one non-zero unit entry (proven in [12]), it can, with a suitable rearrangement, be partitioned into

$$\mathbf{E}_{\alpha\alpha} = \begin{pmatrix} \mathbf{E}_{\kappa\kappa} \\ \mathbf{I} \end{pmatrix}.$$

Noting that each row of $\mathbf{E}_{\alpha\alpha}$ is associated with an element of $\boldsymbol{\phi}_\alpha$, let $\boldsymbol{\phi}_\kappa$ and $\boldsymbol{\phi}_x$ be those elements of $\boldsymbol{\phi}_\alpha$ associated with $\mathbf{E}_{\kappa\kappa}$ and $\mathbf{I}$, respectively, and let $\mathbf{D}_\kappa$ and $\mathbf{D}_x$ be the corresponding submatrices of $\mathbf{D}_\alpha$. This allows (16) to be further partitioned into

$$\begin{pmatrix} \mathbf{N}_\alpha^T \mathbf{N}_\alpha & \mathbf{N}_\alpha^T \mathbf{D}_\kappa & \mathbf{N}_\alpha^T \mathbf{D}_x & 0 \\ \mathbf{D}_\kappa^T \mathbf{N}_\alpha & \mathbf{D}_\kappa^T \mathbf{D}_\kappa & \mathbf{D}_\kappa^T \mathbf{D}_x & \mathbf{E}_{\kappa\kappa} \\ \mathbf{D}_x^T \mathbf{N}_\alpha & \mathbf{D}_x^T \mathbf{D}_\kappa & \mathbf{D}_x^T \mathbf{D}_x & \mathbf{I} \\ \boldsymbol{\mu}_{\alpha\alpha} & -\mathbf{E}_{\kappa\kappa}^T & -\mathbf{I} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}_\alpha \\ \boldsymbol{\phi}_\kappa \\ \boldsymbol{\phi}_x \\ \boldsymbol{\lambda}_\alpha \end{pmatrix} = \begin{pmatrix} \mathbf{b}_\nu \\ \mathbf{b}_\kappa \\ \mathbf{b}_x \\ \mathbf{b}_\gamma \end{pmatrix}.$$

Now we can solve for $\boldsymbol{\phi}_x$:

$$\boldsymbol{\phi}_x = \boldsymbol{\mu}_{\alpha\alpha} \boldsymbol{\theta}_\alpha - \mathbf{E}_{\kappa\kappa}^T \boldsymbol{\phi}_\kappa - \mathbf{b}_\gamma. \tag{17}$$

Then, using a block elementary operation of the form

$$\begin{pmatrix} A & B \\ C & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{I} & 0 \\ C & \mathbf{I} \end{pmatrix} = \begin{pmatrix} A + BC & B \\ 0 & -\mathbf{I} \end{pmatrix}$$

and defining

$$\mathbf{N}_* \equiv \mathbf{N}_\alpha + \mathbf{D}_x \boldsymbol{\mu}_{\alpha\alpha}, \; \mathbf{D}_* \equiv \mathbf{D}_\kappa - \mathbf{D}_x \mathbf{E}_{\kappa\kappa}^T, \; \mathbf{b}_* \equiv \mathbf{D}_x \mathbf{b}_\gamma \tag{18}$$

we can eliminate $\boldsymbol{\phi}_x$ and reduce the system to

$$\begin{pmatrix} \mathbf{N}_\alpha^T \mathbf{N}_* & \mathbf{N}_\alpha^T \mathbf{D}_* & 0 \\ \mathbf{D}_\kappa^T \mathbf{N}_* & \mathbf{D}_\kappa^T \mathbf{D}_* & \mathbf{E}_{\kappa\kappa} \\ \mathbf{D}_x^T \mathbf{N}_* & \mathbf{D}_x^T \mathbf{D}_* & \mathbf{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}_\alpha \\ \boldsymbol{\phi}_\kappa \\ \boldsymbol{\lambda}_\alpha \end{pmatrix} = \begin{pmatrix} \mathbf{b}_\nu \\ \mathbf{b}_\kappa \\ \mathbf{b}_x \end{pmatrix} + \begin{pmatrix} \mathbf{N}_\alpha^T \\ \mathbf{D}_\kappa^T \\ \mathbf{D}_x^T \end{pmatrix} \mathbf{b}_*.$$

Next, we can define

$$\mathbf{v}_c \equiv \mathbf{N}_* \boldsymbol{\theta}_\alpha + \mathbf{D}_* \boldsymbol{\phi}_\kappa - \mathbf{b}_*, \tag{19}$$

solve for $\boldsymbol{\lambda}_\alpha$:

$$\boldsymbol{\lambda}_\alpha = -\mathbf{D}_x^T \mathbf{v}_c + \mathbf{b}_x, \tag{20}$$

and then use a block elementary operation to eliminate $\boldsymbol{\lambda}_\alpha$ and further reduce the system to

$$\begin{pmatrix} \mathbf{N}_\alpha^T \mathbf{N}_* & \mathbf{N}_\alpha^T \mathbf{D}_* \\ \mathbf{D}_*^T \mathbf{N}_* & \mathbf{D}_*^T \mathbf{D}_* \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}_\alpha \\ \boldsymbol{\phi}_\kappa \end{pmatrix} = \begin{pmatrix} \mathbf{b}_\nu \\ \mathbf{b}'_\kappa \end{pmatrix} + \begin{pmatrix} \mathbf{N}_\alpha^T \\ \mathbf{D}_*^T \end{pmatrix} \mathbf{b}_* \tag{21}$$

where $\mathbf{b}'_\kappa \equiv \mathbf{b}_\kappa - \mathbf{E}_{\kappa\kappa} \mathbf{b}_x$. This smaller system is solved for $\boldsymbol{\theta}_\alpha$ and $\boldsymbol{\phi}_\kappa$, after which we back-solve for $\boldsymbol{\phi}_x$ and $\boldsymbol{\lambda}_\alpha$ using (17) and (20).

Computations (12)-(14) also involve calculating $\bar{\mathbf{M}}_{\tilde{\beta}\alpha}\mathbf{x}$. Again, because we are assuming $\tilde{\beta} = \beta$, we have

$$\bar{\mathbf{M}}_{\tilde{\beta}\alpha} = \mathbf{M}_{\beta\alpha} = \begin{pmatrix} \mathbf{N}_\beta^T \mathbf{N}_\alpha & \mathbf{N}_\beta^T \mathbf{D}_\alpha & 0 \\ \mathbf{D}_\beta^T \mathbf{N}_\alpha & \mathbf{D}_\beta^T \mathbf{D}_\alpha & \mathbf{E}_{\beta\alpha} \\ \boldsymbol{\mu}_{\beta\alpha} & -\mathbf{E}_{\alpha\beta}^T & 0 \end{pmatrix}$$

where $\mathbf{N}_\beta$ and $\mathbf{D}_\beta$ are submatrices of $\mathbf{N}$ and $\mathbf{D}$ corresponding to the basic elements of $\boldsymbol{\nu}$ and $\boldsymbol{\sigma}$, and $\mathbf{E}_{\beta\alpha}$, $\mathbf{E}_{\alpha\beta}$, and $\boldsymbol{\mu}_{\beta\alpha}$ are submatrices of $\mathbf{E}$ and μ indexed by α and β. Now, combining the definition of $\mathbf{v}_c$ in (19) with (18) and (17), we get

$$\begin{aligned} \mathbf{v}_c &= (\mathbf{N}_\alpha + \mathbf{D}_x \boldsymbol{\mu}_{\alpha\alpha}) \boldsymbol{\theta}_\alpha + (\mathbf{D}_\kappa - \mathbf{D}_x \mathbf{E}_{\kappa\kappa}^T) \boldsymbol{\phi}_\kappa - \mathbf{D}_x \mathbf{b}_\gamma \\ &= \mathbf{N}_\alpha \boldsymbol{\theta}_\alpha + \mathbf{D}_\alpha \boldsymbol{\phi}_\alpha, \end{aligned}$$

so that $\mathbf{M}_{\beta\alpha}\mathbf{x}$ becomes

$$\mathbf{M}_{\beta\alpha}\mathbf{x} = \begin{pmatrix} \mathbf{N}_\beta^T \\ \mathbf{D}_\beta^T \\ 0 \end{pmatrix} \mathbf{v}_c + \begin{pmatrix} 0 \\ \mathbf{E}_{\beta\alpha} \boldsymbol{\lambda}_\alpha \\ \boldsymbol{\mu}_{\beta\alpha} \boldsymbol{\theta}_\alpha - \mathbf{E}_{\alpha\beta}^T \boldsymbol{\phi}_\alpha \end{pmatrix}. \tag{22}$$

B. Expected Complexity

The above calculations can be done quite rapidly, since (1) each column of $\mathbf{E}_{\alpha\alpha}$ has at most 2 non-zero unit entries, (2) each column of $\mathbf{N}$ and $\mathbf{D}$ (and hence $\mathbf{N}_\alpha$, $\mathbf{N}_*$, $\mathbf{D}_\kappa$, and $\mathbf{D}_*$) has at most 12 non-zero entries, and (3) each row and column of $\boldsymbol{\mu}_{\alpha\alpha}$ has at most one non-zero entry.

Moreover, if n and m are the number of contacts and bodies, and r is the size of the matrix in the reduced system (21), then it is shown in [12] that the complexity of computing $\mathbf{q}'$ or $\mathbf{m}'$ is $O(m + n + r^2)$, assuming the matrix in (21) is kept in factored form if necessary.

To bound r, we note that the matrix in (21) factors as

$$\begin{pmatrix} \mathbf{N}_\alpha^T \\ \mathbf{D}_*^T \end{pmatrix} \begin{pmatrix} \mathbf{N}_* & \mathbf{D}_* \end{pmatrix}.$$

The non-singularity of $\mathbf{M}_{\alpha\alpha}$ ensures that this matrix has full rank, and so its size r must be bounded by the rank of each factor. Each factor has one dimension equal to $6m$, and hence a rank $\leq 6m$, and so we obtain $r \leq 6m$. The per-pivot complexity of $O(m + n + r^2)$ is hence bounded by $O(n + m^2)$. Since the expected number of pivots in Lemke's algorithm is of the same order as the size of $\mathbf{M}$, which is $n(2+d)$, we obtain an overall expected complexity of $O(n^2 + nm^2)$.

Generally speaking, we can expect $m \leq n$, since otherwise there will be more bodies than contacts and the problem will likely decompose into subproblems. If m is noticeably smaller than n, we should get a large improvement over the standard Lemke algorithm, for which the expected complexity is $O(n^3)$. If m is fixed, the complexity improves still further to $O(n^2)$, which is consistent with the tests labeled *structural* in section VII.

In summary, what we have done is reduce the calculations to a form that is partly constrained by the number of degrees of freedom in the system, rather than the number of contacts.

VI. Reducing the Size of the LCP

We now show how to reduce the complexity still further by reducing the expected number of pivots.

The number of pivots is bounded from below by the number of basic $\mathbf{z}$ variables in the final solution. From (6), we can determine that a $\boldsymbol{\lambda}$ variable will be non-zero (and hence basic) for every contact i exhibiting tangential motion. Since this is possible at all contacts, we can expect $O(n)$ $\boldsymbol{\lambda}$ variables to be basic, implying $O(n)$ pivots.

To reduce this, we observe that we don't need all the values of $\boldsymbol{\lambda}$ in order to solve the contact problem. In particular, we needn't consider values of $\boldsymbol{\lambda}$ (or $\boldsymbol{\phi}$) for any contact i which is inactive (i.e., for which θ_i is non-basic): if θ_i is non-basic, then $\theta_i = 0$, and, by the friction cone constraints (4), all the associated $\boldsymbol{\phi}$ variables must also be zero. Inactive contacts therefore make no contribution to $\mathbf{v}$.

Put another way, we need not be concerned about the frictional aspect of a contact until that contact becomes active. We can therefore begin the solution of a contact problem using a reduced, frictionless, LCP of the form

$$\boldsymbol{\nu} \ = \ \mathbf{N}^T \mathbf{N} \, \boldsymbol{\theta} + \mathbf{N}^T \mathbf{k}_x,$$

and then incrementally expand the system into the form of (6) by adding appropriate $\boldsymbol{\phi}$, $\boldsymbol{\lambda}$, $\boldsymbol{\sigma}$, and $\boldsymbol{\gamma}$ variables (and the corresponding matrix rows and columns) as contacts become active. This process of expanding an LCP online is described in Section 4.6 of [1], and specific details for the contact problem are given in [12].

Ideally, most contacts which become active will remain active in the final solution, so that few pivots will be expended for contacts which are later deactivated. Since the number of active contacts equals the number of basic $\boldsymbol{\theta}$ variables, it is bounded by the size r of the matrix in (21), and since r is $O(m)$, we can, in the best case, expect the number of pivots to be reduced from $O(n)$ to $O(m)$.

Combined with the per-pivot complexity of $O(n + m^2)$ described in Section V-B, this would give an overall expected complexity of $O(nm + m^3)$, or $O(n)$ when m is fixed. We do in fact observe behavior close to $O(n)$ in the tests labeled *reduced* in section VII.

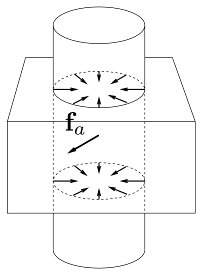

Fig. 3. Peg-in-hole, with 16 contacts and an applied wrench $\mathbf{f}_a$.

VII. Experimental Results

We now present some tests to demonstrate the utility of these methods. The tests were implemented in Java 1.4.2 (with HotSpot) and executed on a 1 GHz Pentium III. Each computation was done with $d = 8$ and friction coefficients μ_i in the range 0.2 to 0.3. Each test measured expected solution times for three different methods:

- *Standard:* Used a standard, efficient implementation of Lemke's algorithm as described in [14].
- *Structural:* Used the structural simplifications described in Section V.
- *Reduced:* Used Section V plus the problem reduction described in Section VI.

Expected solution times were measured by solving each problem 20 times with randomly generated external forces, and averaging the computation times.

The first test was for a single body ($m = 1$) and involved computing the reaction forces on a peg passing through a hole in a fixed block (Fig. 3), in response to a random wrench applied at the center. Contacts were arranged around the hole's top and bottom, and computation times were measured for different numbers of contacts ranging from 8 to 32. The results for the three different methods are shown in 4, and our *structural* and *reduced* methods (shown close-up in Fig. 5) can be seen to be significantly faster. As the number of contacts n varied from 8 to 32, the average compute times (in msec) varied from 6.18 to 400.75 (*standard*), 2.55 to 26.63 (*structural*), and 1.08 to 4.58 (*reduced*), corresponding (roughly) to expected complexities of $O(n^3)$, $O(n^2)$, and $O(n)$, respectively.

The next test involved a variety of ten different single body contact situations (again $m = 1$), similar to those shown in Fig. 1: block in corner, ball resting in the rim of a hole, block in groove, face on surface, etc., with the number of contacts varying from 1 to 16. Results are shown in Fig. 6, with Fig. 7 showing a close-up scatter plot of the results for the *reduced* method. The complexity of the *reduced* method appears to be just a little greater than $O(n)$: as n varied from 8 to 16, the computation time varied from 0.57 to 1.34. Over the same range, the other methods varied from 5.2 to 50.9 (*standard*) and 2.1 to 12.5 (*structural*), again loosely compatible with $O(n^3)$ and $O(n^2)$.

The last test involved multiple bodies, and computed the reaction forces for a stack of blocks, subject to gravity, with a random wrench applied to the top block (Fig. 8).

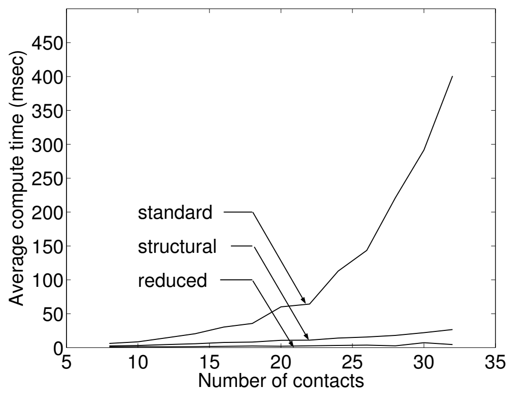

Fig. 4. Average compute times for peg-in-hole by three different methods.

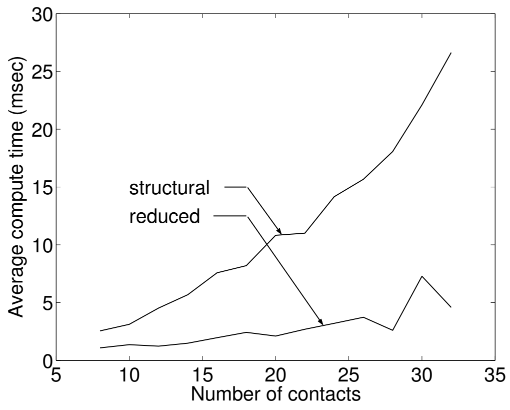

Fig. 5. Close-up of our *structural* and *reduced* results shown in Fig. 4.

Ten configurations were tested, with the number of stacked blocks m ranging from 1 to 5, and the number of contacts n from 4 to 32, depending on the positioning of the blocks with respect to each other. The compute times, shown in Fig. 10, varied from 2.1 to 2136 (*standard*), 1.45 to 224 (*structural*), and 1.32 to 96 (*reduced*). These ranges are roughly compatible, modulo a constant multiplier, with expected complexities of $O(n^3)$, $O(n^2 + nm^2)$, and $O(nm + m^3)$.

VIII. Conclusion

We have developed a fast method for solving rigid body contact problems, with friction, based on Lemke's algorithm for solving LCPs. In essence, the method exploits the sparsity of the factors of $\mathbf{M}$ to reduce the problem in size to one that is largely bounded by the number of degrees of freedom in the system. This amounts to a reduction in expected complexity from $O(n^3)$ down to nearly $O(nm + m^3)$, where n and m are the number of contacts and bodies in the system. When m is fixed, the expected complexity is then close to $O(n)$. These results should improve the utility of rigid body simulation in real-time and interactive applications.

Our method also improves numerical robustness by decreasing the size of the primary linear system that must be solved during each pivot step. This system, (21), is bounded

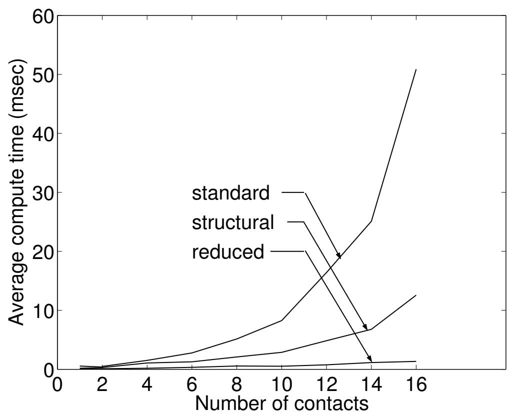

Fig. 6. Average compute times for 10 different single body problems with different numbers of contacts.

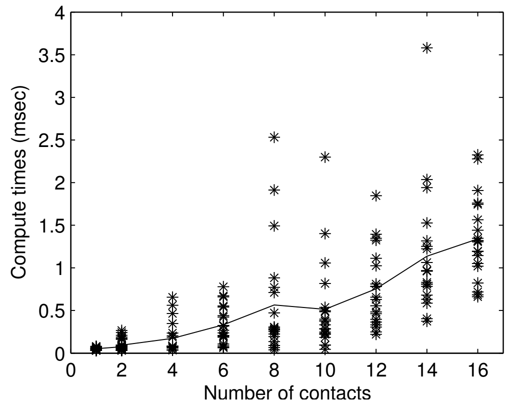

Fig. 7. Close up scatter plot of the *reduced* results shown in Fig. 6.

in size by $6m$, and so if m is small enough (e.g., 3 or 4) it may be solved directly in real-time using numerically robust methods, without having to do factorization updates in which round-off error can accumulate. Our method also eliminates the need to calculate the LCP matrix $\mathbf{M}$, whose size can be quite large even for simple problems.

A Java implementation can be downloaded from http://www.cs.ubc.ca/spider/lloyd/fastContact.html.

ACKNOWLEDGMENT

This work was made possible with funding from IRIS, the Canadian Institute for Robotics and Intelligent Systems.

REFERENCES

[1] R. E. S. Richard W. Cottle, Jong-Shi Pang, *The Linear Complementarity Problem.* Academic Press, 1992.

[2] K. G. Murty, *Linear Complementarity, Linear and Nonlinear Programming*. Helderman-Verlag, 1988.

[3] B. Mirtich and J. Canny, "Impulse-based simulation of rigid bodies," in *Proceedings of the 1995 symposium on Interactive 3D graphics*, (Monterey, California), pp. 181–188, Apr. 1995.

[4] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," in *Proceedings of SIGGRAPH 2003*, pp. 871–878, July 2003.

[5] C. Lacoursiere, "Splitting methods for dry frictional contact problems in rigid multibody systems: Preliminary performance results," in *SIGRAD2003, The Annual SIGRAD Conference*, (Umea, Sweden), Nov. 2003.

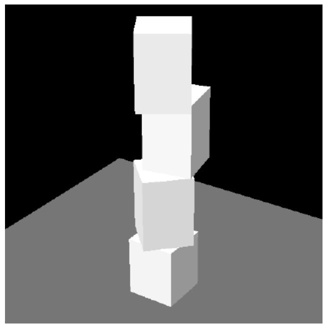

Fig. 8. Stacked blocks used for the multi-body test.

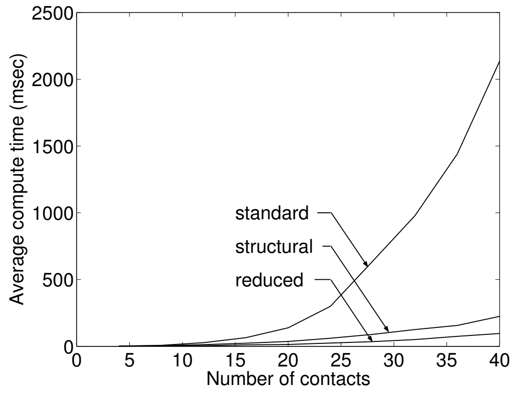

Fig. 9. Average compute times for 10 different stacked-block problems with different numbers of contacts.

[6] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction," *International Journal for Numerical Methods in Engineering*, vol. 39, pp. 2673–2691, Aug. 1996.

[7] M. Anitescu and F. A. Potra, "A time-stepping method for stiff multibody dynamics with contact and friction," *International Journal for Numerical Methods in Engineering*, vol. 55, pp. 753–784, Nov. 2002.

[8] D. Baraff, "Fast contact force computation for nonpentrating rigid bodies," in *Proceedings of SIGGRAPH 94*, pp. 23–34, July 1994.

[9] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, pp. 231–247, Nov. 1997.

[10] J. Sauer and E. Schoemer, "A constraint-based approach to rigid body dynamics for virtual reality applications," in *Proceedings of ACM Symposium on VR Software and Technology 98*, (Taipai, Taiwan), pp. 153–161, Nov. 1998.

[11] D. E. Stewart, "Rigid body dynamics with friction and impact," *SIAM Review*, vol. 42, pp. 3–39, Mar. 2000.

[12] J. E. Lloyd, "Fast implementation of Lemke's algorithm for rigid body contact simulation," tech. rep., Department of Computer Science, University of British Columbia, Jan. 2005, http://www.cs.ubc.ca/spider/lloyd/papers/fastContactLemke.pdf.

[13] M. B. Cline and D. K. Pai, "Post-stabilization for rigid body simulation with contact and constraints," in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Taipei, Taiwain), pp. 3744–3751, Sept. 2003.

[14] R. W. H. Sargent, "An efficient implementation of the Lemke algorithm and its extension to deal with upper and lower bounds," *Mathematical Programming Study*, vol. 7, pp. 36–54, 1978.

[15] J. A. Tomlin, "Robust implementation of Lemke's method for the linear complementarity problem," *Mathematical Programming Study*, vol. 7, pp. 55–60, 1978.

[16] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, "Maintaining LU factors of a general sparse matrix," *Linear Algebra and its Applications*, vol. 88/89, pp. 239–270, 1987.