EUROGRAPHICS '99 / P. Brunet and R. Scopigno
(Guest Editors)

Volume 18 (*1999*), *Number 3*

Interactive Mechanical Design Variation for Haptics and CAD

Donald D. Nelson Elaine Cohen

University of Utah Computer Science Dept.
50 S Central Campus Dr Rm 3190
Salt Lake City, UT 84112-9205
Email: {dnelson,cohen}@cs.utah.edu

Abstract

A fast design variation technique for mechanical systems is presented. It is used to interactively optimize mechanical characteristics while "self-assembling" or satisfying large systems of mechanical constraints. The high speed method is central to providing inverse dynamics force feedback in haptics and control applications. Performance advantages with the use of augmented coordinates for inverse dynamics of closed loop topologies are also noted. The interaction framework allows manipulation of complex assemblies while maintaining kinematically admissible configurations though linkage and joint limit constraints. Furthermore, design variables such as link length can be treated as free variables and optimized to meet design criteria such as assembly dexterity. Assemblies with flexible bodies fit naturally within this framework. Thus, the contribution of this paper is the advancement of techniques in augmented coordinates for the kinematic and force feedback interaction with virtual mechanical assembly design optimization at force control rates.

1. Introduction

Mechanical design variation, known as kinematic and dynamic sensitivity analysis in the mechanical engineering literature, is the optimization of a mechanical design objective with respect to any design parameter such that additional constraints, typically assembly constraints, are met. More generally, kinematics is the study of motions of bodies without considering mass and force, while dynamics is the general study of motion. Forward kinematics maps from a body's configuration variables to positions in Cartesian space and forward dynamics maps forces to body acceleration. Inverse kinematics and dynamics are projections in the other direction. The variational mechanical design methodology presented here allows interactive manipulation and optimization of an assembly. A designer might concurrently change assembly geometry and see the effects, for example, on the rest of the assembly. The framework has similarities to differential manipulation[1], self-assembly[2], and to the kinematic manipulation of open and closed loop systems in reduced coordinates[3, 4, 5, 6]. However, previous special-case assembly interaction techniques are generalized here for all aspects of variational mechanical design so that any mechanical parameter can be a manipulation or optimization "free" variable.

This paper focuses on high performance kinematic sensitivity analysis for assemblies and inverse dynamics force feedback for haptic interaction. A primary concern is scalability for large systems. By using augmented coordinates to describe a system of geometric design objectives and constraints, and using the solution methods presented in this paper, force control interaction update rates can occur. Such kilohertz rate updates of kinematic and dynamic analyses make it possible to have more natural manipulation of assemblies in CAD, animation, and virtual prototyping applications. A general form of the interactive geometric design optimization problem is solved within this framework. An example is the perturbation of joint geometry for maximal dexterity such that the assembly constraints and additional mechanical joint limit constraints are met. The placement of an object in a scene which is to be picked up by an armature might be optimized so that it is in a highly dextrous area.

This approach provides an intuitive, natural setting for assembly design. The method includes elements of an immersive virtual environment:

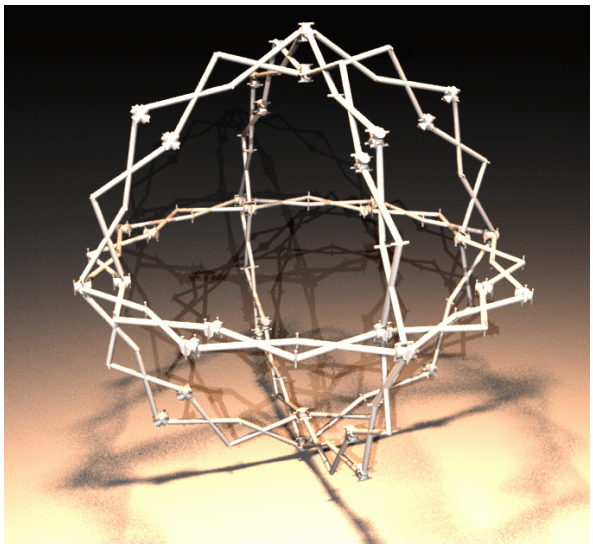


Figure 1: *A spherical toy with 132 bodies (graph nodes) and 192 constraints (graph edges) forms many closed loops. A fast algorithm is required to interact with this mechanism.*

- Concurrent viewing, manipulation, optimization, and force feedback,
- Navigation through the configuration space of a mechanism (Fig.1),
- 6 DOF input extraction from a mouse (Figs.2,14,15) and use of a $PHANToM^{TM}$ haptics device.

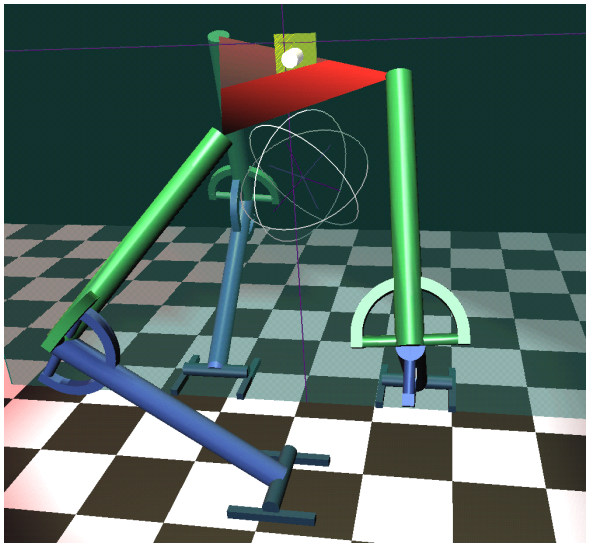


Figure 2: *Manipulation of the geometric parameters of a 6 DOF Stewart platform. The best platform position and orientation such that assembly constraints and additional constraints, like mechanical joint limit stops, are met fits naturally within this framework.*

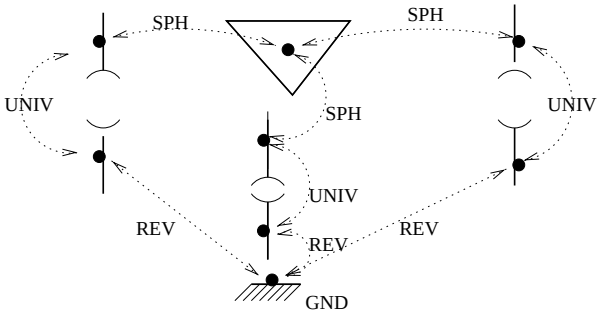


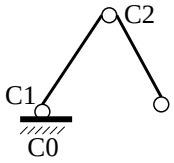
Figure 3: *Joints are edges and bodies are nodes for the graph of a mechanism. This Stewart platform has 3 revolute, 3 spherical, 3 universal joints, and one constraint on a ground body holding it immobile.*

2. Systems of Geometric Constraints

One representation for describing the kinematics of mechanisms for a rigid or flexible body is augmented coordinates. Each rigid or flexible body is defined as a vector of world frame position, orientation, and other variables such as its deformation state. An alternative is to use a minimal set of coordinates, called reduced or joint space coordinates (as used in robotics[3, 4, 5, 6, 7, 8]). We will prefer augmented coordinates due to the generality of expression of nonholonomic constraints, flexible body deformations, and efficient inverse dynamics computations in the closed loop case for haptics (Section 5). Joints in augmented coordinates are defined as a system of constraint equations, $\mathbf{C}$, that are a function of coordinates and design variables, concatenated in $\mathbf{q}$. $\mathbf{C_q}$ refers to the partials of the constraints with respect to the variables in the vector $\mathbf{q}$.

Each constraint (joint) between bodies is represented by a constraint equation ${}^{i}\mathbf{C}$. For example, a two link open loop mechanism that has ground, link_1, and link_2 bodies with constraints ${}^{1}\mathbf{C}$ between ground and link_1, ${}^{2}\mathbf{C}$ between link_1 and link_2, and ${}^{0}\mathbf{C}$ holding ground immobile, is described kinematically by

$$\left.\begin{array}{l} {}^{0}\mathbf{C}(\mathbf{q}_g) \\ {}^{1}\mathbf{C}(\mathbf{q}_g, \mathbf{q}_{l1}) \\ {}^{2}\mathbf{C}(\mathbf{q}_{l1}, \mathbf{q}_{l2}). \end{array}\right\} = \mathbf{0}.$$



This system is a function of three 7×1 vectors $\mathbf{q}$ containing components of a world frame 3×1 Cartesian position vector and 4×1 quaternion orientation vector. We demonstrate a specialized method to augment such systems with design objective functionals to solve kinematic systems in the manner of a generic constrained optimization problem:

minimize **f** subject to $\mathbf{C} = \mathbf{0}$.

Example objectives f include maximizing a mechanical character's dexterity or maximal collision avoidance (Appendices C,D). A fast solution (to follow) will be used for mechanisms which are assembled and do not suffer local minima problems during manipulation, i.e. in the steady state. Accelerations derived from the assembly manipulation create inertial forces and other effects for use in haptics force feedback to the operator.

3. Interactive Design Variation

To solve or self-assemble the system using augmented coordinates, we construct a software constraint object[1, 11, 19] of symbolic equations that do not change form once they are formulated. A new system of equations is not needed for each different set of geometric design parameters that an animator or designer may want to choose. At a high level, we require the following constraint information:

$\mathbf{C}$	$\mathbf{C_q}, \mathbf{C}_t$	$(\mathbf{C_q}\dot{\mathbf{q}})_\mathbf{q}\dot{\mathbf{q}}, \mathbf{C_{qt}}, \mathbf{C}_{tt}$

The last grouping is required only for dynamic analysis, not kinematic design variation, but is included for completeness (see Section 5.1). The rows of **C** are the constraint equations, which when stacked form a constraint manifold. The graph topology indicates which constraints are used.

In this work, elements of the design parameters **q** may be removed and treated as constants. Which parameters are variable may be set by the user by "clicking" on a particular piece of geometry. The constants set by the user will be manipulation parameter constants. We have found it intuitive to render the non-constant parameters (except for part positions), such as variable link length, in wireframe to provide a clear indication of what the optimization algorithm is doing. The user is effectively interacting with the graph of the mechanism by selecting which parameters are variable and which are constant.

The system includes appropriate columns $\mathbf{C_q}$, the partial of **C** with respect to each design parameter **q**, depending on which parameters the user wants to optimize (rather than manipulate or hold constant). Design parameters, and therefore Jacobian columns, can be removed or turned off with flags without reformulating the system equations. The elements of **q** and $\Delta\mathbf{q}$ are adjusted to correspond to the columns in the constraint Jacobian.

3.1. Sparse Matrix Characteristics in Augmented Coordinates

The gradient of the constraint system **C**, that is, the constraint Jacobian $\partial\mathbf{C}/\partial\mathbf{q}$, has sparse structure[11, 19, 20]. Each constraint such as a universal joint constraint in Appendix A or surface-surface "joint" in Appendix E relates two (or a constant small number) of bodies when augmented coordinates are used (Fig.4). The universal joint constraint Jacobian will be of size $4 \times (w_i + w_j)$, for w design variables associated with body i, j. For n_c constraints, $\mathbf{C_q}$ will contain $2n_c$ small, dense submatrices and **0** elsewhere.

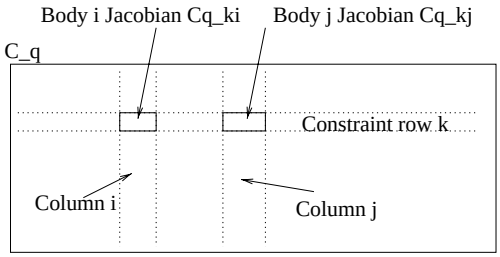


Figure 4: *$2n_c$ dense constraint Jacobian subblocks are placed in a sparse matrix $\mathbf{C_q}$.*

3.2. Interactive Numerical Optimization

A fast method of solving the system equations is absolutely critical to enable interactive editing. Even more serious is the computational requirement with kilohertz rate force feedback for haptics. Our approach has been to abandon the more sophisticated, global optimization methods in favor of a simple, fast algorithm, with reasonably good convergence properties. We may maintain the assembly constraints by looking at the Jacobian of the constraint manifold, which is the direction of the joint constraint force. Scaling this direction by the constraint violation, as in Eq.3, we have a means by which to satisfy the joint constraints within some radius of convergence at extremely high update rates.

Starting from the augmented Lagrangian method[35, 36] for optimizing f subject to constraints **C**, we have

$$\dot{\mathbf{q}}_i = -f_{\mathbf{q}_i} - \sum_j (\lambda_j \frac{\partial \mathbf{C}_j}{\partial \mathbf{q}_i} + k_{1_i}\mathbf{C}_j \frac{\partial \mathbf{C}_j}{\partial \mathbf{q}_i}), \tag{1}$$

$$\dot{\lambda} = \mathbf{k}_2\mathbf{C}. \tag{2}$$

Now we conveniently restack Eq.1 into a concise form for use with the sparse Jacobian $\mathbf{C_q}$,

$$\Delta\mathbf{q} = -f_\mathbf{q}^T - \mathbf{C}_\mathbf{q}^T(\lambda + \mathbf{kC}), \tag{3}$$

and integrate $\dot{\lambda}$, $\dot{\mathbf{q}}$ to maintain the constraints and minimize f.

In practice, assembly constraints are easily maintained once an admissible configuration is found. Quaternions are

used as elements of orientation in $\mathbf{q}$ to help linearize the optimization search space. It is notable that the use of Euler angles causes Eq.3 to fail in most circumstances. The effect of quaternion renormalization is a concern in the iterative algorithm; we address this issue by adding the unit length quaternion condition $\mathbf{q}_{rot}^T\mathbf{q}_{rot} - 1 = 0$ as an additional constraint, one for each body. The constant $\mathbf{k}$ in Eq.3 is a diagonal matrix of weights derived following the robotics calibration literature on scaling and rank.

This method requires only a sparse matrix times column vector operation and therefore one iteration executes several orders of magnitude faster than higher order optimization methods[33, 37]. It is trivially parallelizable. The convergence properties are in general dependent on the nonlinearity of the constraints and other factors. Excellent results have been obtained for large systems with design variables including flexible body coordinates, body positions, joint locations, and ground positions. For a 4 body, spatial mechanism, 100 kilohertz updates are achieved on a 2 processor SGI workstation. For simple planar systems, 1 MHz updates are achieved since fewer than a hundred multiplications are required for each iteration. A second processor helps because the sparse matrix times column vector operation is an easily distributed task. Eq.3 would run in constant time given enough processors since there is no data dependency in the multiplication operation.

Since the user's hand does not move very far in this short update time, so far only a single iteration has been required. More sophisticated techniques are used to get the system to converge initially. Graphics updates are run remotely at a much slower rate.

In practice, the use of a Jacobian pseudo-inverse in Eq.3 to perform Newton-like steps does not work very well at all in augmented coordinates for common examples. It is necessary to be very close to a solution for the Newton-like method to work[10, 12]. Reduced coordinate representations appear to work much better with a Jacobian pseudo-inverse[5, 17]. Again, we cannot afford the $O(n^3)$ cost of matrix inverse operations (which are also required in Fletcher-Powell[33] and other more advanced techniques). Also, we prefer the use of augmented coordinates for use in the inverse dynamics in Section 5.

3.3. Radius of Convergence during Manipulation

A concern with local methods for optimization solutions is that they can get stuck in a minima somewhere away from the global solution. Our use of a local method is due to a severe performance requirement for force interaction and the fact that we can take tremendous advantage of inter-frame coherence during continuous manipulation. The configuration at the previous timestep will be extremely close to the solution we require at the current timestep.

To gain some insight into the radius of convergence within this coherent environment, we turn to some empirical evidence in representative cases. We assume that some admissible configuration satisfies the assembly and objective requirements initially. The rate of change in slope in Eq.3 is measured due to a range of input changes. As long as a user does not move so quickly as to escape some locally convex region, then the constraints will always be maintained in the steady state. Because the local method runs at kilohertz rates, the "escape velocity" for the manipulation of a node's position, for example, in practice would be on the order of one kilometer per second.

4. Generalized Inverse Kinematics

Manipulation through inverse kinematics is often useful in animation operations and kinematic workspace visualization. Some solutions from the literature do not use numerical means and are quite robust[23, 24] but not very general in scope.

A largely neglected inverse kinematics problem is the manipulation of closed loop mechanisms (Fig.2). When a part is constrained to be attached to a finger, but no configuration can reach the position or orientation of the finger, some "best" or optimal solution is required. Closed chain inverse kinematics are found from the geometric satisfaction solution given above. Using objectives (f) that indicate the norm of the point-finger distance should be minimized, or by adding finger constraints to the assembly, we have an effective means for manipulating a virtual mechanism with multiple fingers and hands.

General inverse kinematics solutions for flexible body coordinates are also found with our geometric optimization (Fig.9). See the references[25, 26] for a description of coordinates of flexible bodies $\mathbf{q}_f$. A design objective f for flexible coordinates is the minimization of strain energy

$$U = \frac{1}{2}\mathbf{q}_f^T\mathbf{K}_f\mathbf{q}_f \tag{4}$$

for stiffness matrix $\mathbf{K}_f$. For our purposes, minimizing the strain energy $\mathbf{q}_f^T\mathbf{K}_f\mathbf{q}_f$ will prevent large deformations in the flexible body inverse kinematics solution.

5. Graph Theory and Inverse Dynamics

Inverse dynamics computations in reduced coordinates make use of open chain analysis methods, such as the recursive Newton-Euler formulation[22]. For closed loops, others[5, 9, 15, 16] use graph theory for cutting closed loop systems into open chains, at which point the recursive methods can be used. However, it is necessary to solve for the cut forces, a cubic cost $O(n^3)$ operation in general.

A mechanical topology can be represented by a graph. Bodies are nodes, and joints (or other constraints) are edges

in this context. Optimal spanning trees can be constructed so that edges left out are cut, e.g., joints/constraints are cut, in the least computationally expensive way. The graph of the mechanism from Fig.2 is shown in Fig.3. Our method is organized around graph theory. However, we do not require cut joints because our coordinates describe joint constraints at a local scope (by using local frame body geometry to define joints as constraints) and the operator's fingers are presumed to be involved in only one constraint (each) on the mechanism.

5.1. Inverse Dynamics

The general constrained equations of motion are frequently used to solve forward and inverse dynamics problems. If the Lagrange multiplier technique is used, two derivatives of the constraint equations are coupled with the equations of motion, for mass tensor $\mathbf{M}$, undetermined multipliers λ, external forces $\mathbf{Q}_e$, and forces quadratic in velocity $\mathbf{Q}_v$[11, 20].

$$\mathbf{C_q}(\mathbf{q},t)\ddot{\mathbf{q}} = -(\mathbf{C_q}\dot{\mathbf{q}})_\mathbf{q}\dot{\mathbf{q}} - 2\mathbf{C_{qt}}\dot{\mathbf{q}} - \mathbf{C}_{tt} \tag{5}$$

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}_\mathbf{q}^T\lambda = \mathbf{Q}_e(\mathbf{q},\dot{\mathbf{q}},t) + \mathbf{Q}_v(\mathbf{q},\dot{\mathbf{q}},t) \tag{6}$$

Inverse dynamics computations are needed to compute joint loads. For bodies that are involved in only one constraint in the mechanism (such as a finger-to-assembly attachment) the loads can be evaluated using only Eq.6. In that case, the joint constraint force $\mathbf{C}_\mathbf{q}^T\lambda$ will be the unknown and can be obtained, for open or closed loops, in augmented coordinates as,

$$\mathbf{C}_\mathbf{q}^T\lambda = -\mathbf{M}\ddot{\mathbf{q}} + \mathbf{Q}_e + \mathbf{Q}_v. \tag{7}$$

This form can be computed quickly. Propogation of accelerations and forces along a chain is not required because we have the world frame generalized accelerations $\ddot{\mathbf{q}}$ from the user's movements of the assembly and the augmented formulation establishes joints at a local scope. Inverse dynamics formulations in reduced coordinates are also possible, such as the recursive Newton-Euler formulation[22], which can be extended for closed loops[5, 16]. In general, to compute forces applied to a body that is part of a loop (has more than one constraint on it), the augmented coordinate formulation requires solving for the Lagrange multipliers and accelerations in Eqs.5,6.

5.2. Inverse Dynamics for Design through Haptics

Whether the multipliers λ and Jacobian $\mathbf{C_q}$ are obtained separately or not, the joint constraint force $\mathbf{W}_c = \mathbf{C}_\mathbf{q}^T\lambda$ in the configuration space of the mechanism may be expressed in terms of body space force and torque $\mathbf{W}$ through standard transformation operators $\mathbf{G}$[26, 29],

$$\mathbf{W} = \frac{1}{4}\mathbf{G}(\mathbf{q})\mathbf{W}_c \tag{8}$$

This force and torque may be projected in the direction of a joint axis to provide the amount of torque required by a controls system to produce a motion. For our interactive haptics force feedback application, $\mathbf{W}$ can be transmitted to the user, where the user's fingers or hands are point or rigid constraints $\mathbf{C}_{fing}$ at various places on the assembly. Which constraint is used will correspond to whether point or grasping contact is assumed.

The joint constraint force will now serve two purposes; it will be the constraint force required to keep the user on the assembly, and it will be the force felt by the user from inertial and quadratic velocity forces from the mechanism while manipulated (accelerated by the user).

$\ddot{\mathbf{q}}$ and $\dot{\mathbf{q}}$ in the equations of motion may be obtained by observing how $\mathbf{q}$ from the haptics inverse kinematics changes over time. For a PHANToM^{TM} or other device with encoders, there is no noise and no filtering of position is required to obtain $\ddot{\mathbf{q}}$ numerically. For devices with potentiometers, the situation is more problematic, but filtering with some delay works well.

When flexible body coordinates are present, Eq.6 can be extended with elastic forces $\mathbf{K}_f\mathbf{q}_f$ and elastic inertial terms. The addition of flexible force feedback effects during manipulation gives further insight into the characteristics of the virtual mechanism.

5.3. Multi-finger, Multi-hand Manipulation

When multiple PHANToM^{TM}s or a MotionStar^{TM} is available, it may useful to incorporate more than one point contact or grasping contact on the assembly. In fact, these constraints are added as part of building the assembly initially and may be removed or included with flags as previously mentioned. When not all of the finger constraints may be met, as will usually be the case with low degree of freedom virtual mechanisms, the best solution is selected.

5.4. Forward Dynamics Manipulation

Forward dynamics is an inherently more difficult and expensive computation than inverse dynamics. Both Eqns.5 and 6 are required for the Lagrange Multiplier method. An approach to assembly manipulation using forward dynamics would be to "push" assemblies around or manipulate them by attaching stiff springs from the finger to the assembly. The need for finger constraints is removed with springs, but stiff springs are needed to approach a firm grasp. Due to computational costs associated with closed loop forward dynamics and the focus of our research on design through kinematic sensitivity analysis, the inverse kinematics and inverse

dynamics method has been used so far. A full forward dynamics approach fits into our framework and would have the advantage of accounting for free swinging parts that are not directly prescribed by the user.

6. Summary

Our work attains interactive force control rates due to a sparse matrix method and the assumption that global optimization concerns will not be a problem in the steady state in practice for assembly manipulation. The optimization method has been organized around:

- Modular symbolic constraint objects for design parameter selection,
- Graph theory organization of the mechanical topology, constraint framework and design parameters,
- Local scope constraints in augmented coordinates that avoid expensive computations with closed loop inverse kinematics and inverse dynamics analysis for haptics.

Appendix A: Simple Joint Constraints

This paper presents a modular framework in which simple joints and more complex joints can be used together for assembly and objective optimization. Constraint equations $\mathbf{C}$ and their Jacobians $\mathbf{C_q}$ are required. The constraints are written in the manner of the spherical, revolute, translational, universal, and other geometric joint constraints[13, 10, 29]. We will derive the constraint equation for a spherical and universal joint for reference here. The universal joint included here incorporates a spherical joint constraint for the first three constraints and a perpendicular constraint for the fourth equation.

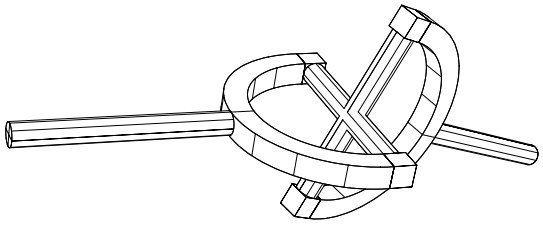


Figure 5: *A universal joint is a spherical joint that can rotate in any way as long as both axes are perpendicular.*

The universal joint features of the two bodies must meet, which is enough for a spherical joint constraint, given by

$$\mathbf{C}^{sph}(\mathbf{q}) = \left[\ \mathbf{A}(\mathbf{q}_{i,rot})\overline{\mathbf{k}}_i + \mathbf{q}_{i,disp} - \mathbf{A}(\mathbf{q}_{j,rot})\overline{\mathbf{k}}_j - \mathbf{q}_{j,disp}\ \right]_{3x1} \tag{9}$$

$\mathbf{A}$ is a rotation matrix, $\mathbf{q}_{i,disp}$ is a translation vector, and $\mathbf{q}_{i,rot}$ are the rotation coordinates, whether Euler angles, quaternions, Rodriguez parameters, etc., and $\overline{\mathbf{k}}$ is a constant local body frame vector indicating the position of the universal joint feature.

The perpendicular condition enforces the constraint that the local body axes, $\overline{\mathbf{h}}_i$, $\overline{\mathbf{h}}_j$ of each piece remain at 90 degrees,

$$\mathbf{C}^{perp}(\mathbf{q}) = \left[\ (\mathbf{A}_i\overline{\mathbf{h}}_i)^T(\mathbf{A}_j\overline{\mathbf{h}}_j)\ \right]_{1x1}. \tag{10}$$

Four degrees of freedom are removed by the universal joint. The Jacobian of this constraint has 14 columns, i.e. the partial derivative with respect to the translation and quaternion coordinates.

$$\mathbf{C}^{univ}(\mathbf{q}) = \left[\begin{array}{c} \mathbf{C}^{sph} \\ \mathbf{C}^{perp} \end{array}\right]_{4x1} \tag{11}$$

Appendix B: Alpha_1 CAD Specification of Virtual Mechanisms

The Alpha_1 modeling environment[27] has been extended to interface assemblies with the high performance design variation framework. Joint features are defined as local body frame vectors in the Alpha_1 SCL modeling language. For example, the quarter-torus in Fig.6 has two revolute joint features, located by $\overline{\mathbf{u}}_1$ and $\overline{\mathbf{u}}_2$, with directions $\overline{\mathbf{v}}_1$ and $\overline{\mathbf{v}}_2$ in the local frame of the part. A body, or graph node, associates the model geometry *surfaces* with the node in the SCL construct

```
b : body(surfaces);
```

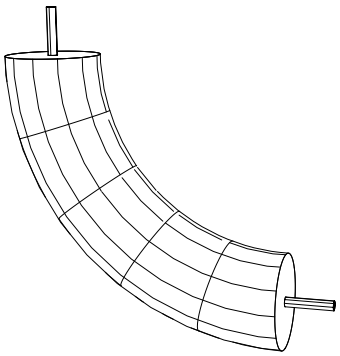


Figure 6: *Quarter torus local body geometry. Two revolute joints are attachment features of this part.*

Parts are assembled with connections, or graph edges indicating joint constraints, between bodies (graph nodes), such as

```
c : connection(b1,b2,array(b1_u1,b1_v1),
    array(b2_u1,b2_v1),"REVOLUTE");
```

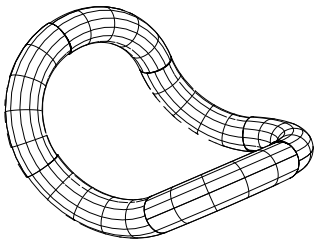


Figure 7: *The assembled mechanism, or satisfaction of the geometric constraints. The ends of the linkage are attached, forming one closed loop.*

The joint feature vectors can be constants or may be derived though inheritance or instantiation like any SCL modeling variable. The entire assembly is defined with a list of connections

```
a : assembly(c1,c2,c3,...);
```

Special finger (manipulation) objectives or constraints are also added with connections in a similar way. The interaction thread spawns from the Alpha_1 modeling process to run the high performance haptics design variation. Alpha_1 models are updated with the results of this interactive optimization.

Appendix C: Dexterity Objectives

The manipulability ellipsoid defined by the eigenvectors of the manipulator Jacobian **J** is the measure of the dexterity of a mechanism or a character's hands; it is an interesting design objective. A good review of dexterity measure is in the references[30], including appendages with closed loops[41]. In differential kinematics[8], one manipulability measure is

$$f(\mathbf{q}) = \sqrt{det(\mathbf{J}(\mathbf{q})\mathbf{J}^T(\mathbf{q}))}. \tag{12}$$

It uses the geometric velocity manipulator Jacobian **J**. We will use the cross product form for spatial manipulators. For example, for revolute joints, for joint coordinate axes $\mathbf{z}$, Denavit-Hartenburg coordinate origin $\mathbf{o}$, and body point $\mathbf{x}$

$$\mathbf{J}_i = \begin{bmatrix} \mathbf{z}_i \times (\mathbf{x} - \mathbf{o}_i) \\ \mathbf{z}_i \end{bmatrix}. \tag{13}$$

Other joint constraints have similar forms for **J**[31]. The algebra for $f(\mathbf{q})$ can easily be reworked in augmented coordinates, noting that a constant vector in that body frame $\overline{\mathbf{u}}$ locates $\mathbf{o}$ (which can be obtained initially as in the references[28]). Casting the maximization as a minimization, we obtain a new equation for f.

$$f = det(\mathbf{J}\mathbf{J}^T)^{-1/2}. \tag{14}$$

Appendix D: Dodging Bullets

Distance measures, such as those used in avoiding obstacles (see Fig.13), are easily written. The generic distance measure objective for a point $\mathbf{p}$ avoiding obstacle $\mathbf{o}$ for a suitably redundant open or closed kinematic chain is

$$f(\mathbf{q}) = min_{p,o} 1/||\mathbf{p} - \mathbf{o}||. \tag{15}$$

Other objectives such as maximization of a configuration for joint limit distance or maximizing workspace volume are similarly written for mechanical design.

Appendix E: Surface Constraints

More complicated joint constraints such as nonholonomic, rolling contact constraints between two surfaces, such as in a spatial cam-follower mechanism, also fit naturally into this design variation framework. The gradient of such constraints can be obtained through expensive numerical finite difference methods[2], but a fast, closed form solution also exists[43]. The rate of change of the parametric contact coordinates $\dot{\mathbf{u}}$ with respect to the surface body coordinates $\mathbf{q}$ in the context of floating contact[2] of the form

$$\dot{\mathbf{u}} = \mathbf{D}\dot{\mathbf{q}} \tag{16}$$

is given in the references[43]. This gradient $\mathbf{u_q} = \mathbf{D}$ of the parametric contact coordinates is the key to obtaining the gradient of the surface contact constraint and to nonholonomic, unilateral surface rolling contact velocity constraints. A preliminary implementation of the surface constraints has been incorporated into the Alpha_1 modeling system.

Acknowledgments

The author would like to thank David Johnson and Ali Nahvi for their recommendations. Thanks also go to the students and staff of the *Alpha_1* project, within which this work was developed. Support for this research was provided by NSF Grant MIP-9420352, by DARPA grant F33615-96-C-5621, and by the NSF and DARPA Science and Technology Center for Computer Graphics and Scientific Visualization (ASC-89-20219).

References

1. Gleicher, M., Witkin, A., "Differential Manipulation," in *Proceedings of Graphics Interface*, 1991.

2. Witkin, A., Fleischer, K., Barr, A., "Energy Constraints on Parameterized models," in *Computer Graphics Proceedings, SIGGRAPH*, pp. 225-232, 1987.

3. SDFast user's manual, Symbolic Dynamics, 1990.

4. IGRIP (Interactive Graphics Robot Instruction Program), Deneb Robotics Inc.

5. Nahvi, A., Nelson, D., Hollerbach, H., Johnson, D., "Haptic Manipulation of Virtual Mechanisms from Mechanical CAD Designs," *International Conference on Robotics and Automation*, 1998.

6. Zhou, J., Badler, N., "Inverse Kinematics Positioning Using Nonlinear Programming for Highly Articulated Figures," in *Trans. on Graphics*, 13, 4, pp. 313-336, October 1994.

7. Denavit, J., Hartenberg, R., "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices," *ASME J. of Applied Mechanisms*, vol. 22E, 1955.

8. Sciavocco, L., Siciliano, B., Modeling and Control of Robot Manipulators, McGraw-Hill, 1996.

9. Zou, H., Abdel-Malek. K., Wang, J., "A Variational Approach for the Design of Spatial Four-Bar Mechanism," *Mech. Struct. & Mach.*, Vol. 25., pp. 41-59, 1997.

10. Shabana, A., *Computational Dynamics*, John Wiley & Sons, New York, NY, 1994.

11. Serban, Haug, E.J., "Kinematic and Kinetic Derivatives in Multibody System Analysis," Mechanics of Structures and Machines, Vol 26, No. 2, 1998 and in 1997 ASME Design Engineering Technical Conferences.

12. Johnson, A., "An Open Architecture Approach to Kinematic Analysis for Computer-Aided Embodiment Design," in *Computer Aided Design*, Vol 30. no. 3, March 1998, pp. 169-178.

13. Yang, K., Lee, S., Bae, D., Suh, M., "Use of Joint Geometric Conditions in Formulating Cartesian Constraint Equations," *Mech. Struct. & Mach.*, Vol. 23, No 3, pp. 395-417, 1995.

14. Ashrafiuon, H., Mani, N., "Analysis and Optimal Design of Spatial Mechanical Systems," in J. of Mechanical Design, Trans. of the ASME, Vol. 112, No 2., 1990.

15. Tsai, F., Haug, E., "Real-time multibody system dynamic simulation: Part I. A modified recursive formulation and topological analysis," *Mech. Struct. & Mach.*, Vol. 19., pp. 99-127, 1991.

16. Lin, S. "Dynamics of the manipulator with closed chains," IEEE Trans. Automation and Robotics, Vol. 6, pp. 496-501, 1990.

17. Lenarcic, J., "Alternative Computational Scheme of Manipulator Inverse Kinematics", in *International Conference on Robotics and Automation*, pp. 3235-3240. 1998.

18. Barzel, R., Barr, A., "A Modeling System Based On Dynamic Constraints," in *Computer Graphics*, pp. 179-187, ACM, August 1988.

19. Witkin, A., Gleicher, M., Welch, W., "Interactive Dynamics," *1990 Symposium on Interactive 3D Graphics*, pp. 11-21, ACM, March, 1990.

20. Baraff, D., "Linear Time Dynamics Using Lagrange Multipliers," in *Computer Graphics Proceedings, SIGGRAPH 1996*, New Orleans, August 4-9, 1996.

21. Metaxas, D., Terzopoulos, D., "Dynamic Deformation of Solid Primitives with Constraints," in *Computer Graphics Proceedings, SIGGRAPH 1992*, Chicago, July 26-31, 1992.

22. Luh, J., Walker, M, Paul, R., "On-line computational scheme for Mechanical Manipulators," ASME *J. Dynamic Systems, Measurement and Control*, vol. 102, pp. 69-76, 1980.

23. Kramer, G. A., Solving geometric constraint systems : a case study in kinematics, McGraw Hill, 1992.

24. Fudos, I., Hoffmann, C., "A Graph-Constructive Approach to Solving Systems of Geometric Constraints," in *ACM Trans. Graphics*, Vol. 16, No. 2, April 1997, pp. 179-216.

25. Shabana, A., "Computer Implementation of Flexible Multibody Equations," *Computer-Aided Analysis of Rigid and Flexible Mechanical Systems*, Pereira, Ambrosio, eds., Kluwer Academic Publishers, pp. 325-349, 1994.

26. Shabana, A., Dynamics of Multibody Systems, Cambridge University Press, 1998.

27. Cohen, E., Lyche, T., and Riesenfeld, R., "Discrete B-Splines And Subdivision Techniques In Computer Aided Geometric Design And Computer Graphics," *Computer Graphics and Image Processing*, Vol 14, Number 2, October 1980.

28. Sklar, M., "Geometric calibration of industrial manipulators by circle point analysis," Proc., 2nd Conf. on Recent Advances in Robotics, FAU, Boca Raton, FL, May 18-19, pp. 178-202.

29. Haug, E., Intermediate Dynamics, Prentice Hall, 1992.

30. Soper, R., Canfield, S., Reinholtz, C., Mook, D., "New Matrix-Theory-Based Definitions for Manipulator Dexterity," 1997 ASME Design Engineering Technical Conferences.

31. Spong, M., Vidyasagar, M., Robot Dynamics and Control, John Wiley & Sons, 1989.

32. McCarthy, M., Introduction to Theoretical Kinematics, MIT Press, 1990.

33. Fletcher, R., Powell, M., "A rapidly convergent descent method for minimization," in *J. of Computing*, 1963.

34. Shigley, J. E., Mitchell, L. D., *Mechanical Engineering Design*, McGraw-Hill, 1983.

35. Platt, J., Barr, A., "Constraint Methods for Flexible Models," in *Computer Graphics Proceedings, SIGGRAPH 1988*, pp. 279-288.

36. Conn, A., Gould, N., Toint, P., "A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds," SIAM Journal on Numerical Analysis, Volume 28, Number 2, April, 1992.

37. Cohen, M., "Interactive Spacetime Control for Animation", in *Computer Graphics Proceedings, SIGGRAPH 1988*, pp. 279-288.

38. Szalavari, Z., Gervautz, M., "The Personal Interaction Panel–A Two Handed Interface for Augmented Reality," in *Eurographics* 16(3), pp.C:335-346, 1997

39. Yeh, T., Vance, J., "Applying Virtual Reality Techniques to Sensitivity-Based Structural Shape Design," in *J. Mech. Des.*, Vol. 120, Dec. 1998.

40. Harada, M., Witkin, A., Baraff, D., "Interactive Physically-Based Manipulation of Discrete/Continuous Models," in *Computer Graphics Proceedings, SIGGRAPH*, 1995.

41. Park, F., Kim, J., "Manipulability of Closed Kinematic Chains," in *J. Mech. Des.*, Vol. 120, Dec. 1998.

42. Ruspini, Diego, Oussama Khatib, "Dynamic Models for Haptic Rendering Systems." in *Advances in Robot Kinematics: ARK'98*, June 1998, Strobl/Salzburg, Austria, pp 523-532.

43. Nelson, D., Differential Surface Kinematics for Arbitrary Surface Parameterizations, Technical Report, University of Utah, 1999, available at http://www.cs.utah.edu/~dnelson/communication.ps.

44. Jayaram, S., Connacher, H., Lyons, K., "Virtual assembly using virtual reality techiques," in *Computer-Aided Design*, Vol.. 29, No. 8, pp.575-584, 1997.

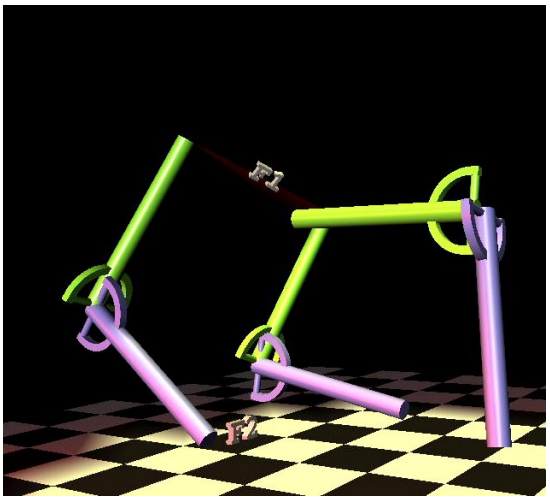


Figure 8: *Two-handed MotionStar^{TM} manipulation of an assembly at grasping points F1 and F2. The platform and ground position are attached to "fingers" with rigid constraints.*

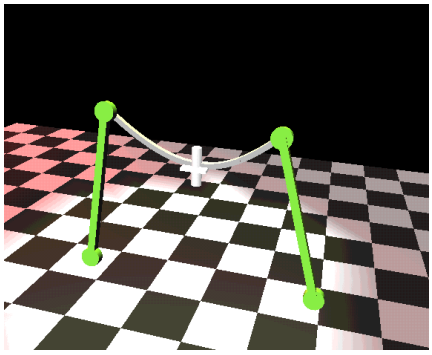


Figure 9: *Inverse kinematics with flexible body degrees of freedom.*

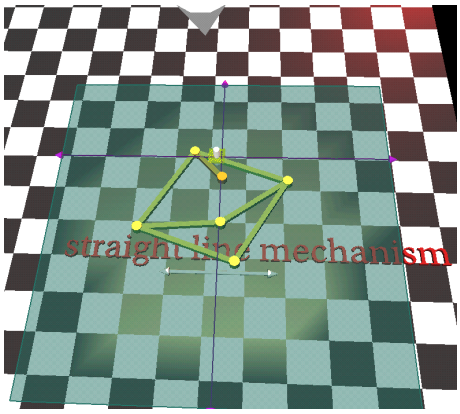


Figure 10: *Manipulation of the orange link of the Peucellier mechanism. The objective function is that the mechanism follow the straight line throughout the manipulation, which affects the link length and joint location design variables.*

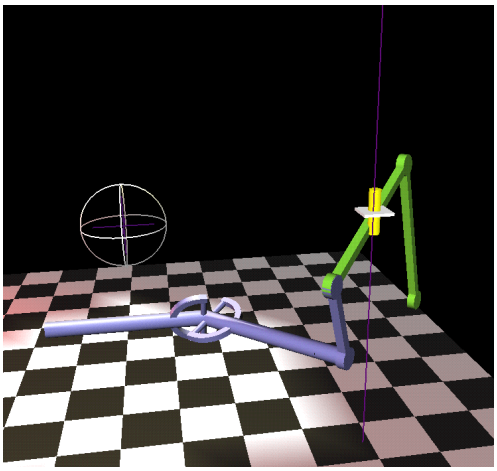


Figure 11: *Manipulation of 1 DOF 4 bar driving a universal joint mechanism. The mechanism is attached to ground in 3 locations, requiring a high performance closed loop manipulation solution.*

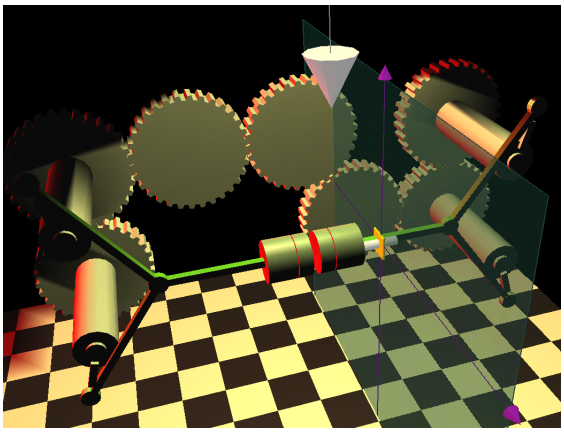


Figure 12: *Interactive perturbation of an* Alpha_1 *piston assembly. The mechanism moves with 1 DOF due to constraints from gears and four slider-crank subassemblies.*

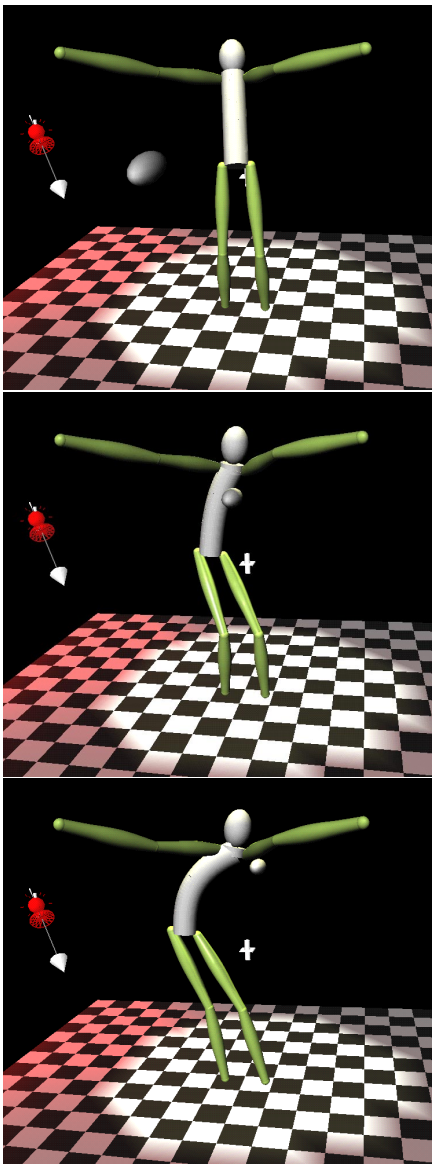
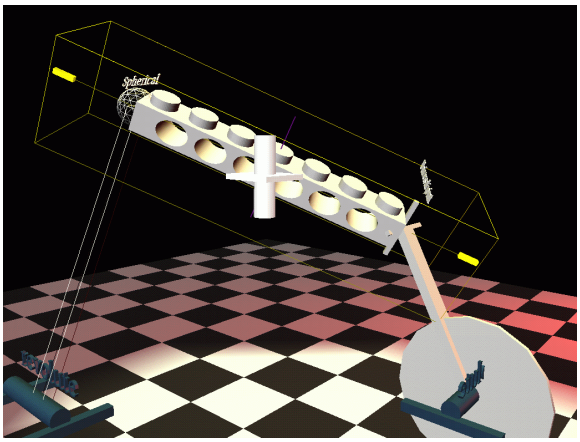
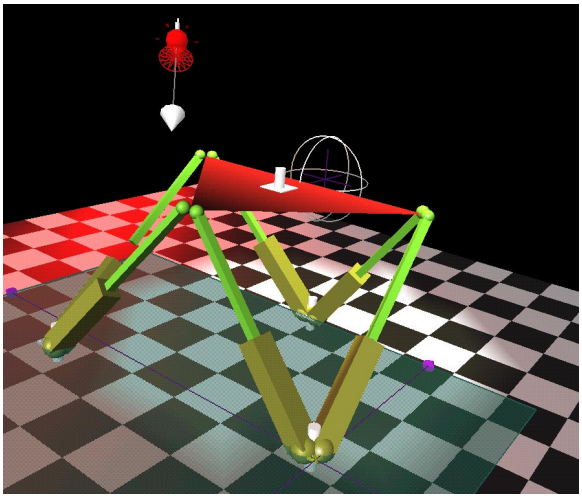


Figure 13: *The creature moves its flexible belly redundant legs (spherical hip and ankle joints, revolute knee joints) to avoid an approaching bullet.*



Nelson and Cohen, Design Variation, Figure 14: *Manipulation of joint location in a 1 DOF 4 bar spatial mechanism with a spherical, a universal and 2 revolute joints. The length of the left link is variable, indicated by the wireframe view. The objective f is to find the best configuration so that the middle of the "leggo" link is as close as possible to the user's "hand" indicated by the manipulation widget.*



Nelson and Cohen, Design Variation, Figure 15: *Manipulation of the linear actuator Stewart platform mechanism. The ground locations or "feet" can be "dragged" because they are set to be manipulation variables.*