Constraint Jacobians for Constant-Time Inverse Kinematics and Assembly Optimization

Donald Nelson

UUCS-98-004

Department of Computer Science
University of Utah
50 S Central Campus Dr Rm 3190
Salt Lake City, UT 84112-9205

17 February 1998

An algorithm for the constant-time solution of systems of geometric constraint equations is presented in this work. Constraint equations and their Jacobians may be used in conjunction with other numerical methods to solve for a variety of kinematics, dynamics, and assembly optimization problems. The use of constraint equations for these purpose is an under-utilized method in this area. The use of quaternions for coordinates in these constraint equations is shown to be a key choice in the optimization problem for avoiding local minima.

Constraint Jacobians for Constant-Time Inverse Kinematics and Assembly Optimization

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

Abstract

An algorithm for the constant-time solution of systems of geometric constraint equations is presented in this work. Constraint equations and their Jacobians may be used in conjunction with other numerical methods to solve for a variety of kinematics, dynamics, and assembly optimization problems. The use of constraint equations for these purpose is an under-utilized method in this area. The use of quaternions for coordinates in these constraint equations is shown to be a key choice in the optimization problem for avoiding local minima.

1 Introduction

The use of the constraint style of programming for the solution of kinematics, dynamics and automatic interactive re-assembly problems is a versatile and extensible framework in which to operate in. These equations express geometric relationships between bodies, such as the requirement that two joints stay together.

We will denote our coordinates for these geometric constraints with $\mathbf{q}$, our augmented Cartesian generalized coordinates. An alternative is to use joint-space (i.e. joint angle) coordinates, but such reduced coordinates limit our expressivity of the constraint equations and prohibit non-holonomic and other irregular constraints from being formulated [2].

The *kinematic* constraint equation, is expressed in general as a function of the coordinates, higher derivatives of the coordinates, and of time.

$$\mathbf{C}(\mathbf{q},t) = \mathbf{0}$$

An example is the spherical joint constraint equation (which is not time dependent):

$$\mathbf{R}_i + \mathbf{A}_i(q)\overline{u}_i - \mathbf{R}_j - \mathbf{A}_j(q)\overline{u}_j = \mathbf{0}$$

which says that the endpoints of the links of each half of the spherical joint must meet. $\mathbf{R}$ is translation, $\overline{u}$ is the link length vector in body coordinates, and $\mathbf{A}$ is orientation. $\mathbf{R}$ and $\mathbf{A}$ are in the *world, or absolute*, frame. The generalized coordinates for 3D are thus 3 components of position ($\mathbf{R}$) and 3 components of orientation ($\mathbf{A}$, the ZXZ Euler angles). A black box can be used to compute all joint constraint Jacobians, which we have found to be a particularly useful characteristic for modular, maintainable software development.

2 Background

To solve for kinematics and assembly optimization problems, we need the Jacobian of the constraint equation; that is, the partial derivative of the constraint equation with respect to the generalized coordinates. The constraint Jacobian is denoted as $\mathbf{C}_q$.

For the dynamics problem, we take 2 time derivatives of the constraint equation as in [1] so that we can use it in conjunction with the equation of motion in the next section.

$$\mathbf{C}_q\ddot{q} = -\mathbf{C}_{tt} - \mathbf{C}_{qq}\dot{q}\dot{q} - 2\mathbf{C}_{qt}\dot{q}$$

Two derivatives of the constraint equation can be used in conjunction with the equations of motion to allow a dynamics solution [1]. Various dynamics formulations such as the Lagrange or Newton-Euler equations (see Eq. 1) can be augmented with the term $C_q^T\lambda$ to allow for the joint constraint force. This is known as the Lagrange multiplier technique [1].

$$\mathbf{M}_i\ddot{q}_i + \mathbf{C}_q^T\lambda = Q_e + Q_v \tag{1}$$

where Q_e is the external applied (gravity or push) forces, and Q_v wraps inertial forces and terms quadratic in velocity corresponding to $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$ from the original equation of motion. Baraff uses F_{ext} to represent $Q_e + Q_v$. $\mathbf{C}_q^T\lambda$ is the joint constraint force. These equations are coupled to a system that has joint constraint and external forces.

[1] solves the constraint equation and motion equation simultaneously as a system for $\ddot{q}$ and λ, while [2] finds that there is an always sparse solution to obtaining the Lagrange multipliers first. The by-product of Lagrange multipliers are used in this paper for joint reaction force in haptics operator interaction.

3 Assembly Optimization

Automatic assembly and interactive re-assembly optimization for finding mechanical configurations satisfying joint constraints $\mathbf{C}(q) = 0$, that is by Newton's optimization method,

$$\mathbf{C}_q\Delta q = -\mathbf{C}(q) \tag{2}$$

is very useful in assembly analysis or for simply eliminating the need to piece together assemblies by hand. The solution with a stable, always converging solution given by Eqn.3, used as an initial guess finder for the quadratic converging solution, Eqn. 4.

$$\Delta q = k\mathbf{C}_q^T(-\mathbf{C}(q)) \tag{3}$$

$$\Delta q = \mathbf{C}_q^{-1}(-\mathbf{C}(q)) \tag{4}$$

The evaluation of $\mathbf{C}_q^{-1}$ makes use of a pseudo-inverse.

This paper makes use of the change in rotation of a vector with respect to the change in the coordinates defining that rotation. [1] derives such a term $\frac{\partial A(q_{4..6})v}{\partial q_{4..6}}$ for a set of Euler angles, which works fine for dynamics analysis. Optimization problems with local minima arise with a set of Euler angles in most configurations.

To circumvent local minima problems, a quaternion 4-tuple will be used to represent orientation. Slanting $\boldsymbol{q} = q_0 + \mathbf{q}$ will denote the quaternion. Methods of obtaining the term $\frac{\partial(qvq^*)}{\partial q}$ in $\mathbf{C_q}$ are obtained using quaternion differentiation.

Modes of a mechanism that cause frequent local minima with the Euler angle representation are avoided with the use of quaternions. Optimization of the constraint equations of a mechanism by gradient descent is essentially linearly interpolating between position and quaternion orientations.

Linear quaternion interpolation and the more accurate "slerping" [13] provide smooth, physically based interpolation of the intermediate mechanism orientation. Quaternions make use of the fact that getting from one orientation to another can be done by rotating around a constant vector. This has physical basis, which can also be seen in the angular velocity (ω) relation with quaternion rates.

$$\dot{q} = q \cdot (0, \omega)$$

The gradient used in descending towards a more optimal orientation does not therefore become stuck in modes that may occur due to strange orientations from the use of Euler angles.

Pursuing the physical basis further, we may think of Eqn 3 as a kind of spring. When one uses

Lagrange multipliers to enforce constraints, multipliers λ give rise to a force in configuration space of $J^T\lambda$.

If we think of springs trying to enforce the constraints, and make the spring force be proportional to error, we would have in general

$$spring - force = -kJ^TC \tag{5}$$

The maximum step size during iteration over Δq can be determined using Lyapunov functions as in [8]. A choice of k=0.1 in practice is found experimentally to be a high yet stable value across all mechanisms. This size is a limit on how fast the assembly will converge to satisfaction. Because there is no direct dependence on the size of the mechanism and k, the constant time algorithm in the next section holds.

The physically based explanation may now be used to see the advantage of quaternion configuration representation. Springs can be constructed to apply torque in configuration space. The torque would move a vector about a single axis, just like quaternion linear or slerping interpolation. No similar physically based torque for a change in Euler angles would hold.

3.1 Massively Parallel Optimization

Eqn. 3 is a sparse matrix multiplied by a column vector. The locations of the dense areas in the sparse matrix are known and generated from the constraint relation between bodies. Each element in Δq may be evaluated on a separate processor by sending each row of $\mathbf{C}_q^T$ and a copy of $\mathbf{C}$ to a different place. Since each processor has at most 2 or 3 non-zero areas in $\mathbf{C}_q^T$, only 2 or 3 areas of multiplication need to occur between $\mathbf{C}_q^T$ and $\mathbf{C}$ in practice because bodies have at most 2 or 3 joints with which to connect to others. The interdependence of data is not a problem; data is simply copied or farmed out to each processor and independent results collected on a host processor. The algorithm therefore has the valuable property in that it will run in constant time given enough processors.

The algorithm can be summarized in the following pseudocode.

```
Pick initial q, send to processors 1..n, n=# bodies.
Iterate until ||C||<small epsilon,
  for i=1..n, compute constraint Jacobians for all joints of body i.
  for i=1..n,
     for i=j,2,..k  where k=2,3 for practical mechanisms
        On processor i, multiply transpose of constraint j for body i
           by elements of C that are affected by body i.
        Add result of multiplications into shared memory vector q.
```

4 Constraint Jacobians for Assemblies with Operator-in-the-Loop

Operator interaction with virtual mechanical assemblies is key to the analysis of the kinematic and inertial characteristics of mechanical assemblies within a virtual prototyping environment. The operator grasps or pushes a place on the mechanism to determine kinematic range and feel inertial effects.

One method is to pose the problem as a robotics calibration optimization problem as in [3], whereby the configuration of the mechanism that minimizes the distance between the grasped point and the operator's hand is optimized using *dense manipulator Jacobians*. The resulting distance causes a constraint wrench to push the operator back to a reachable point on the mechanism. In addition, the operator's movements cause acceleration of the mechanism that cause inertial force that are solved for in a typical inverse dynamics problem. This inertial force is the dynamics wrench, which is also rendered to the operator. However, the technique is not scalable as $O(N_{bodies}^3)$ cost is required to perform an inverse of a dense manipulator Jacobian.

Another method that makes use of O(N) forward dynamics is is the use of a spring contact model between the mechanism and the operator. The distance between the hand and the designated grasped point on the virtual mechanism will cause the spring to push on the virtual mechanism. The force applied to the mechanism will be the same force returned to the operator. However, in practice, there is a limit to the amount of spring stiffness due to integration stability and servo rate. An attractive option is to model the operator's hand as a joint contact, and look at joint constraint force, which is presented here.

One of the results of the Hamiltonian, Lagrangian, and other dynamics formulations is that with the use of Lagrange multipliers, the term $\mathbf{C}_q^T\lambda$ in the equation of motion is the total force of constraint which keeps joints attached in a mechanical assembly.

An easy method of presenting dynamic forces to an operator who has grasped a part of the mechanical assembly, possibly with more than one hand, is to make use of the joint constraint force $\mathbf{C}_q^T\lambda$. The operator's hand can be modeled as a "spherical joint" or ground joint with either 3 or 6 degrees of freedom constrained. The force of constraint for that ground equation can be reflected to the operator so that dynamics effects of the mechanism will be felt. In this way, the forward dynamics equations can be used to return force to the operator as a by-product of computing the Lagrange multipliers and constraint Jacobians.

5 Results

The constraint optimization techniques have been developed within a prototypical Matlab environment and are being incorporated into Utah's Alpha_1 modeling environment. Local minima are not a problem when quaternions are used to represent orientation. It is notable that the constraint equations and constraint Jacobians are used both for optimization and dynamics, due to the modularity inherent in the constraints as separate boxes. On a 2 processor R10000 SGI Octane, the speedup for the optimization re-assembly is very nearly a factor of two. When the operator is a part of the assembly optimization through a grounded finger, a single iteration is assumed sufficient for the interactive re-assembly, and an update of 12kHz is achieved for small examples with simple joints.

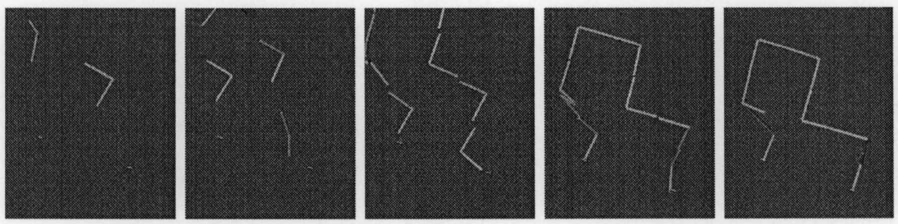

Figure 1: Steps of auto-assembly during optimization for a mechanism with bendy, 90 degree pieces and 2 ground constraints.

Data in the comparison between spring and constraint Jacobian methods of operator prescribed motion of virtual mechanical assemblies is being collected at this time and is a subject of future work.

6 Conclusion

The approach advocated in this paper for constraint satisfaction, inverse kinematics, and similar assembly problems is the use of constraint equations and their Jacobian, along with optimization, to solve a large class of three dimensional problems. The advantage of this approach is a tiny implementation, re-use of constraints for dynamics, computation expense linear in the number of constraint equations, and massively parallelizable for constant asymptotic running time.

The technique of constraint Jacobians for use in optimization, dynamics, and kinematics has been shown to be an effective tool in mechanical analysis and design. Constraint Jacobians have also been shown to be useful for haptics force feedback.

7 Acknowledgments

The author would like to thank Ali Nahvi for his help with references to related work. 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] Shabana, A., *Computational Dynamics*, John Wiley & Sons, New York, NY, 1994.

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

[3] Nahvi, A., Nelson, D., Hollerbach, J., Johnson, D., "Haptic Manipulation of Virtual Mechanisms from Mechanical CAD Designs," To appear, International Conference on Robotics and Automation, 1998.

[7] Funda, J., Taylor, R.H., and Paul, R.P., "On homogeneous transforms, quaternions, and computational efficiency," in *IEEE Trans. Automation and Robotics,* vol 6, pp. 382-388, 1990.

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

[9] Barzel, R., Barr, A., "A Modeling System Based On Dynamic Constraints," in in *Computer Graphics Proceedings, SIGGRAPH 1988,* pp. 179-187, August, 1988.

[12] Watt, A., Watt, M., Advanced Animation and Rendering Techniques, Addison-Wesley, 1992, pp. 357-362.

[13] Shoemake, D., "Animating rotation with quaternion curves," in *Computer Graphics Proceedings, SIGGRAPH 1985,* pp. 245-254, July, 1985.

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

[15] Barr, A., Currin, G., Gabriel, S., Hughes, J., "Smooth Interpolation of Orientations with Angular Velocity Constraints using Quaternions," in *Computer Graphics Proceedings, SIGGRAPH 1992,* Chicago, July 26-31, pp. 313-320, 1992.

[16] Baumgarte, J., "Stabilization of constraints and integrals of motion in dynamical systems," in *Comp. Meth. in Appl Mech. and Eng.,*, vol 1, pp 1-16, 1972.

[17] Evans, D. J., "On the representation of orientation space, in *Molecular Physics*, 34 (1977), no. 2, 317-325.

[18] Du Val, P., "Homographies, quaternions and rotations," Oxford Mathematical Monograph, 1964.

[19] Baraff, D., "Fast contact force computation for nonpenetrating rigid bodies," in *Computer Graphics,* pp. 23-24, ACM, July, 1994.

[20] Lotstedt, P., "Mechanical systems of rigid bodies subject to unilateral constraints," in *SIAM Journal of Applied Math.,* vol. 42, pp. 281-296, April 1992.

[21] Angeles, J., Rational Kinematics, Springer Tracts in Natural Philosophy, Vol. 34, Springer-Verlag, 1988.

[22] Bouma, W., Fudos, I., Hoffmann, C.M., "A Geometric Constraint Solver," in *CAD* 27, 1995, pp. 487-501.

[23] Dixon, L, Spedicato, E., Szego, G., Nonlinear Optimization, Theory and Algorithms, Birkhauser, 1980.

[24] Ratschek, G., Rokne, J., New Computer Methods for Global Optimization, Ellis Horwood, 1988.