1H-5

最短時間動作を目指す衝突回避経路生成法

岡野彰　古畑智武
日本アイビーエム　東京基礎研究所

1　はじめに

ロボットオフラインプログラミングシステムに衝突チェック機能が取り入れられつつある。グラフィックスターミナル上の目視では衝突を見逃す可能性があるからだが、人間にとってそれでもなお、プログラミングにおけるロボットの衝突回避の問題は難しい。しかも単に衝突を避ければ良いというものではない。ロボットのタクトタイムは１秒を争う。目標の位置まで最短時間で動いてほしい。しかしロボットの動作時間はロボットの関節角の変化量に依存し、３次元空間における移動距離に単純には比例しないため、最短時間動作を簡単には教示できない。本稿ではロボットが目標の位置まで極力最短時間で移動できるような衝突回避パスを生成するアルゴリズムを提案し、実験結果を示す。

2　衝突回避問題

これまでにPerez[1]と近藤ら[2]が、障害物が静止物体の場合に最短移動距離の経路を生成しうる衝突回避法を発表している。また筆者も移動物体をロボットが最短移動距離で回避するアルゴリズム[3]を提案した。これらはコンフィギュレーション空間[4]を用いているため、計算量と記憶量がロボットの自由度の指数のオーダとなる。これはロボットの台数が２台あるいは３台になると事実上計算が不可能であることを意味している。またロボット１台としても、実用的な計算時間に納めるためには、はるかに高速な手法を用いる必要がある。

衝突回避が重要となるグロスモーションをプログラムする環境とはどんなものだろうか。ユーザがグラフィックスの画面を見ながらロボットの動作を教示していくばあい、ロボットの目標点を指示すればあとはシステムが自動的に衝突回避経路を算出してくれるのが理想である。しかしそれでは計算時間がかかり、またその経路が最短時間のものとは限らない。それよりはユーザとのなんらかのインタラクションによって人間が直接教示するものより動作時間の短い経路を生成したほうがよいのではないだろうか。例えば衝突チェックの結果から衝突時のリンクと障害物との位置関係、衝突速度等がわかるので、ユーザがおおまかな衝突回避の方向を与えれば、ロボットの動作時間が極力最短になるような回避経路をプログラムが生成してはくれないものだろうか。そのためにはインタラクションに堪えられなければならないので、計算量がロボットの自由度の多項式になって欲しい。次節ではそのための基礎的なアイデアについて述べる。

3　関節－時間空間における可動範囲

ロボットの動作時間は関節角度の変位量に依存する。加減速度を無視し角速度が一定とすれば、関節角度の変位量をおのおの角速度で割ったもののうち、一番遅いものが動作時間となる。まず、動作終了時刻までにどこまで経路を変更できるかを考えてみよう。

ロボットの自由度の数に時間軸を加えた次元数を持つ空間において、初期姿勢を頂点とし動作完了時刻を表す平面(時間軸に垂直)を底面とする超角すいを考える（図１）。底面のエッジは時間軸以外のいずれかの軸に平行である。角すいの広がり方はそれぞれの軸に対応するロボットの関節の速度を表している。この角すいに障害物が交わっているばあい、障害物を回避したところからまた角すいが始まる。この角すいの内部は与えられた時刻までに移動できる範囲である。

さて考えやすいように、この角すいをロボットの全ての軸について一軸ずつと時間軸とを含む平面で切ってみよう。動作完了時刻まで目標の値に戻ってこられる範囲を図２に示す。境界線の傾きはその関節の速度を表す。ロボットが目標点まで移動するばあい、かならずある関節について目標値が境界線上に乗っている。他のものは領域内である。この領域の中であれば、どこを通ろうと動作時間に影響を与えず、規定の時間内に目標点に到着する。したがってロボットが衝突を避けつつ最短時間で動くためには、境界線上にない関節、言い換えれば余裕のある関節の値を変えればよい(図３)。もしどの関節の値を変えても避けられないばあいは動作終了時刻を最小限延ばしその時間内に同様のことを行えばよいことになる(図４)。次節ではこの考え方に基づいていかに関節の回避量を計算するかについて述べる。

4　アルゴリズムの概要

関節角度を変位させるのに必要なデータは、変位させる時刻と回避の方向、距離である。それらから動作に余裕のある関節の変位量を決めていく。

ロボットが衝突を回避するといってもその方法は一通りではない。図５に示したように数通り存在する。これらの

Motion-time-minimum Oriented Collision Avoidance Algorithm
OKANO Akira, FURUHATA Tomotake
IBM Research, Tokyo Research Laboratory

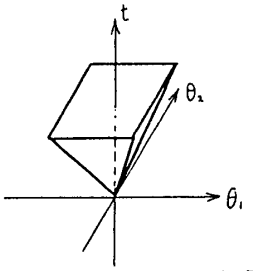

図１　関節時間空間での超角すい

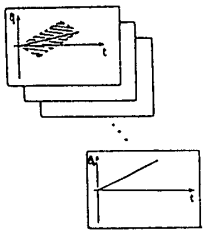
図２　可動範囲

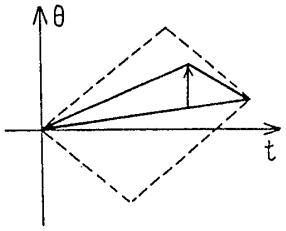

図３　可動範囲内の衝突回避

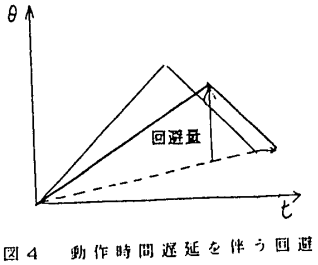

図４　動作時間遅延を伴う回避

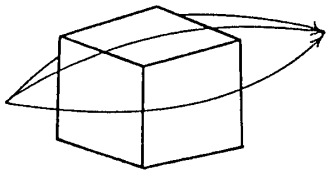
図５　大局的衝突回避

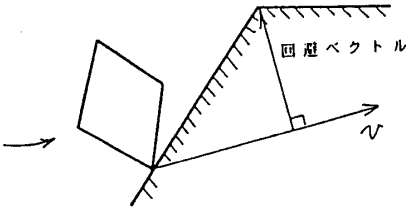

図６　回避ベクトル

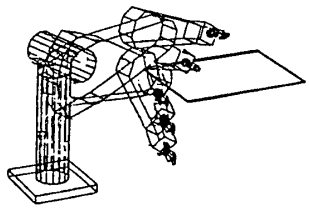
図７　衝突回避例

うちどれが最適かを求めるためにはローカルにせよコンフィギュレーション空間を計算しなければならない。それを避けるためにユーザがどの回避法を取るのかを対話的にグラフィックスターミナルから指示することにする。指示する内容としてはロボット側の頂点と障害物側のエッジ、またはその逆とする。以下の説明ではロボット側の頂点と障害物側のエッジを指示したと仮定する。

そして関節の値を変位させるための時刻と、その時刻での回避する方向と距離を求める。ここでは衝突チェック[5]を行い、上で指示した頂点の衝突時の位置と速度を計算する。そして、頂点をとおり速度を方向とする直線と、やはり上で指示したエッジとの最短距離・方向とその距離を与える時刻を計算することにする(図６)。

次にその回避ベクトルを達成するための関節角度を計算する。回避する時刻において本来ロボットがいる位置の近傍で、関節の微少な単位時間あたりの変位量から、指示された頂点の３次元空間での微少変位量を求める。そして回避方向への各関節変位の寄与率を計算する。寄与率と回避量から関節における変位量を求める。このさい余裕のある関節から変化させていき、回避量に達したら回避経路を出力し、アルゴリズムは終了する。すべての関節がその時刻における最大値または最小値に達したら、すべての関節を動員した場合における残りの回避量を達成するための時間を見積もる。そして各関節の変位量を再計算する。あきらかに、このアルゴリズムのオーダはロボットの関節の数に比例している。

以上の衝突回避法をインプリメントし、実験を行った。例を図７に示す。現在種々の動作・衝突パターンについて評価を行っている。勿論本来非線形なロボットのキネマティックスを線形近似しているため誤差が生じる。そのためアルゴリズムの詳細はまだ手直しの余地がある。しかし基本的な

・大局的な衝突回避の方向はユーザが指示し、
・衝突回避の時刻を計算し、
・その時刻における３次元空間での回避ベクトルを算出し、
・その時刻での各関節の最大値、最小値を計算し、
・回避ベクトルを達成するために動作に余裕のある関節へ変位量を割り振る

ことは最短動作時間を指向し、かつ高速な衝突回避経路生成の枠組みとして有効である。実際我々が直接教える衝突回避経路よりも動作時間が平均して20%から30%早い経路を算出している。

5　おわりに

最短時間動作に近い経路を生成し、計算量がロボットの自由度の数に比例する衝突回避法を提案した。この手法によって複数のロボットが協調的に動いているような場合でもお互いに衝突を回避するための方法の展望が開けた。

参考文献

[1]T. Lozano-Perez, "A Simple Motion-Planning Algorithm for General Robot Manipulators," Journal of Robotics and Automation, IEEE, 1987, June, Vol RA-3, No3, pp 224-238

[2]近藤、木村、"迷路法に基づく自由空間産出法を用いた障害物回避動作計画、" 日本ロボット学会誌、5巻4号、1987, pp 11-20

[3]"Automatic Generation of collision-free path for a robot," IBM Technical Disclosure Bulletin, vol. 31, no. 12, pp 161-162, May, 1988

[4]T. Lozano-Perez,"Automatic Planning of Manipulator Transfer Movements,"IEEE Transactions on Systems, Man, and Cybernetics, Vol SMC-11, No 10, 1981

[5]岡野，川辺，嶋田"シミュレーションによる移動物体間の衝突検出"，日本ロボット学会誌、６巻１号、pp 35-41, 1988