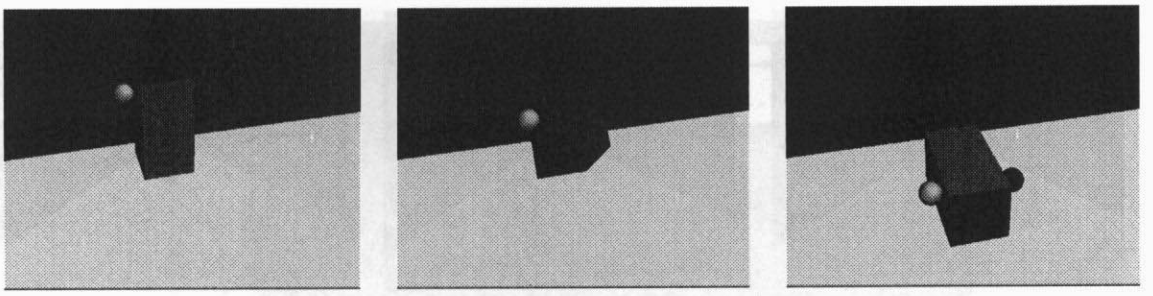


(a) start (b) intermediate (45[deg]) (c) goal

Fig. 6.27: Tumbling 1': Planned Operation

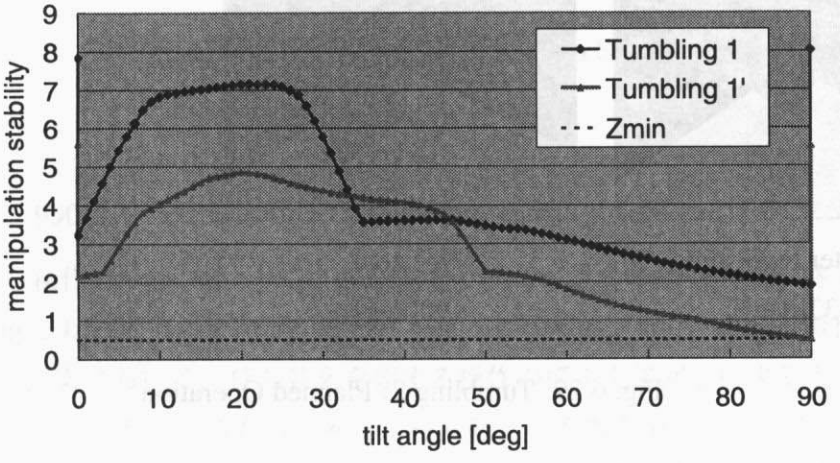


Fig. 6.28: Tumbling 1: Manipulation Stability

直方体の転がし操作 (2)

一方，対象物の背後だけでなく片方の側面にも障害物がある場合は，対象物を両側からつまむようには指を配置することができない．この場合について計画を行った（問題 Tumbling 2）ところ，Fig. 6.29 のように，途中で持ち替えを 1 回行って，対象物を転がす操作が生成された．このときの対象物上の指位置は，最初は指 1 が $[0,\ -1.25,\ 5]^T$，指 2 が最初 $[0,\ 1.875,\ 5]^T$ であり，33[deg] まで対象物を傾け，そこで持ち替えにより指 2 が $[0,\ -2.5,\ -2.5]^T$ へと移動する．また，各指の指令力，および操作の確実性の指標の値の推移を Fig. 6.30, 6.31 に示す．この場合は，各指の制御モードを数回切り替えるような計画結果が得られた．Fig. 6.31 の 33[deg] のところで一瞬操作の確実性の指標値が下がっていると

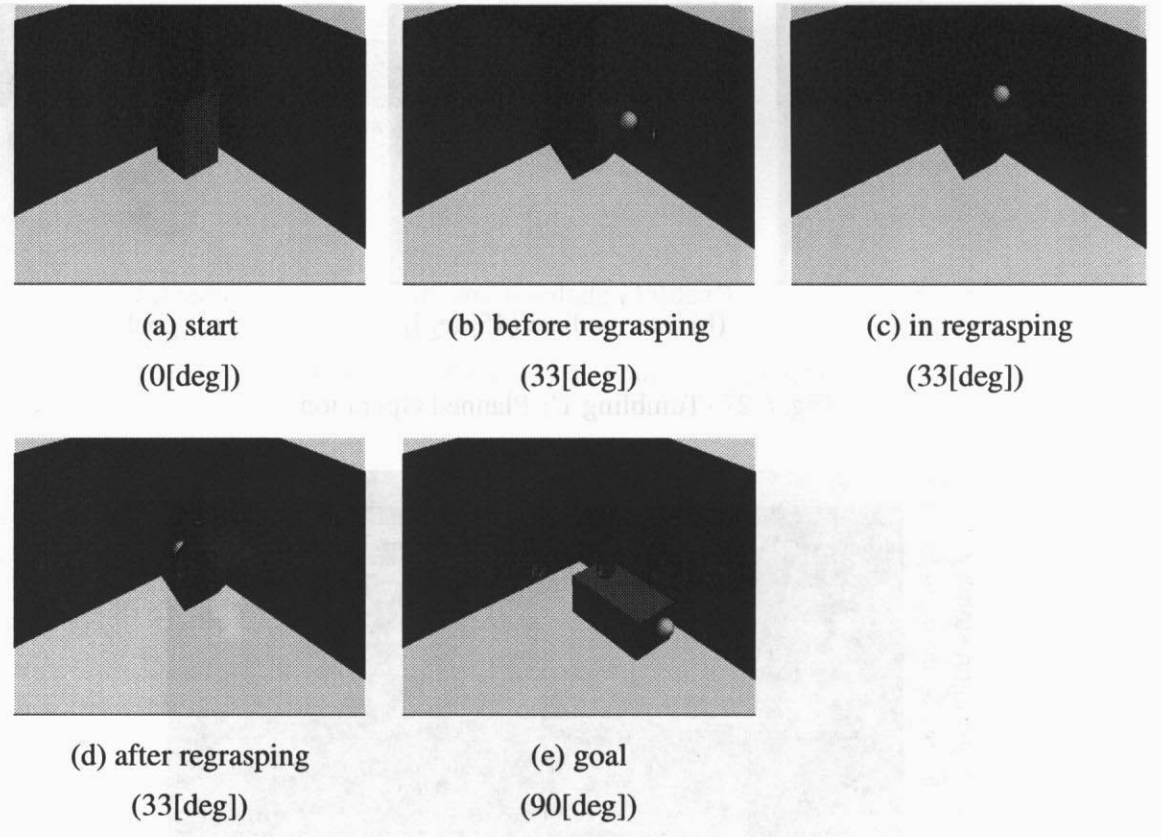


Fig. 6.29: Tumbling 2: Planned Operation

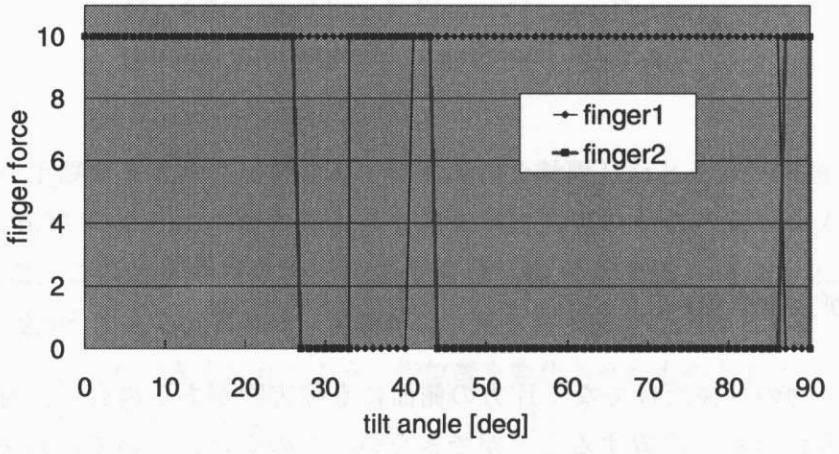


Fig. 6.30: Tumbling 2: Planned Finger Forces

ころは，持ち替え中で1本の指だけで対象物を支えているときに対応する．

なお，この計画において展開されたノードの数は74,028個で，有効分岐数は1.362，計画

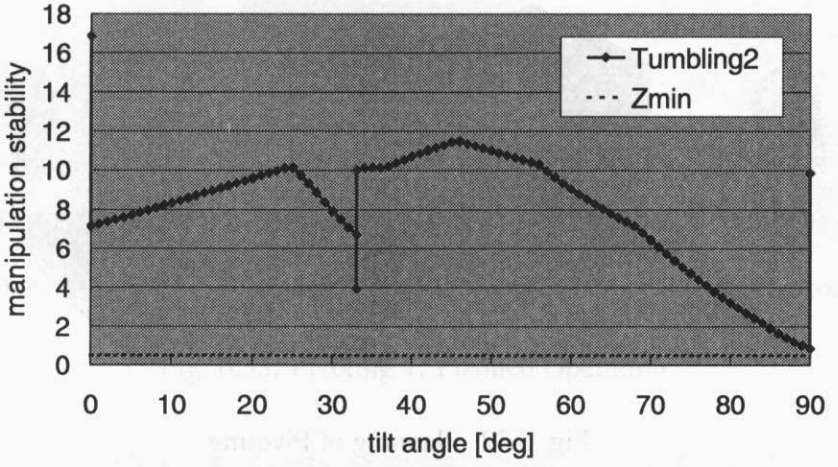


Fig. 6.31: Tumbling 2: Manipulation Stability

に要した時間は 990 CPU 分であった.

これら転がし操作の計画において，Tumbling 1 は比較的短時間で計画が終了したのに対して，Tumbling 2 はかなり長い時間を要している．この差の一因は，本章の計画アルゴリズムで用いる A^* 探索において，持ち替えを要する操作の探索を後回しにするようなヒューリスティック関数が使われているためである．持ち替えの不要な前者では，A^* 探索が有効に機能して，比較的高速に探索が行える．これに対して持ち替えの必要な後者の操作では，持ち替えなしの操作の探索に失敗した後で，持ち替えを伴う操作の探索を始めるために，必然的に探索時間が長くなる．ただし，この場合でも，持ち替えを 2 回以上行う操作の探索は後回しにされるので，その意味では A^* 探索自体は有効に機能している．

6.7.3 ピボット操作の計画

次に，ここでは平面上での直方体および立方体のピボット操作を考える．対象物を初期位置から目標位置までピボット操作で旋回させてあやつるためのロボット指の動作を求める問題である．

ここでは，対象物が ψ 傾いて辺 AB_1 で地面と接触している状態からスタートし，ここから頂点 A をピボットとして旋回させ，再び ψ 傾いて辺 AB_2 で地面と接触している状態へと遷移させる (Fig. 6.32)．このとき，$\angle B_1AB_2 = \theta_0$ とする．そして，ピボット操作中の対象物のコンフィギュレーションを θ で表し ($0 \leq \theta \leq \theta_0$)，頂点 B の地面からの高さ h が $h = \phi_0 \sin(\pi\theta/\theta_0)$ となるように動かすとする．ここでは $\theta_0 = 45$[deg], $\psi = 10$[deg],

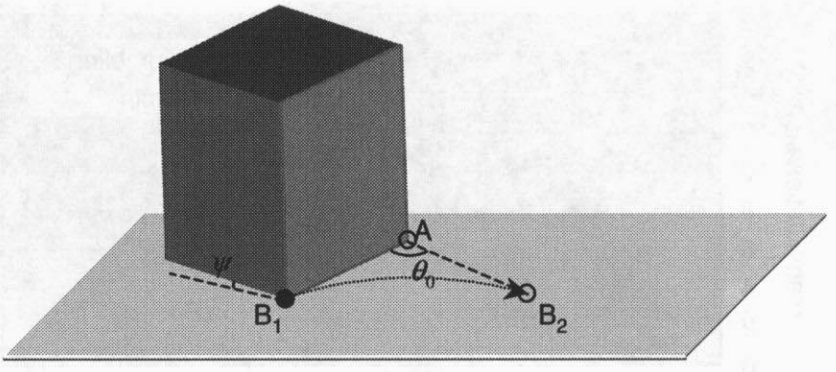


Fig. 6.32: Planning of Pivoting

$\phi_0 = 5$[deg] と設定した.

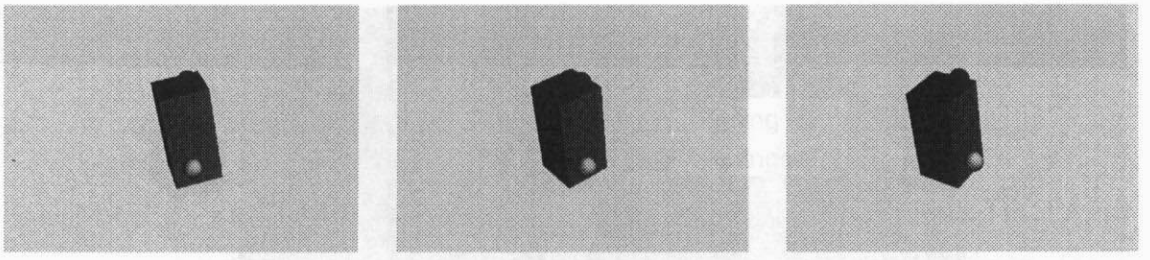
コンフィギュレーション空間の離散化については，前項と同様，対象物上の各面を 7×7 のグリッドに切ることによって，指の配置可能点を設定し，また，対象物の自由度 θ についても同じく 31 段階に離散化した.

直方体のピボット操作

まず，$5 \times 5 \times 10$ の大きさの直方体のピボット操作を考える．この対象物の質量は 1 ($M_o = 1$) で均質であるとする.

この場合，Fig. 6.33 のように，対象物の側面に片方の指（指 1）を位置制御で添えて，上面をもう片方の指（指 2）が力制御で押さえながら，対象物を旋回させる動作が生成された．このときの対象物上の指位置は，指 1 が $[-2.5,\ 0.625,\ -2.5]^T$，指 2 が $[-1.875,\ -0.625,\ 5]^T$ である．また，各指の指令力，および操作の確実性の指標の値の推移を Fig. 6.34，6.35 に示す．ピボット操作が始まると，対象物と環境との接触が辺接触から一点接触に移行するので，操作の確実性の指標値は一気に下がる．ピボット操作が終わると指標値は元に戻る.

なお，この計画において展開されたノードの数は，1,335,201 個のうち 837 個で，有効分岐数は 1.1674，計画に要した時間は 6.3 CPU 分であった.



(a) start (b) intermediate (c) goal

Fig. 6.33: Pivoting 1: Planned Operation

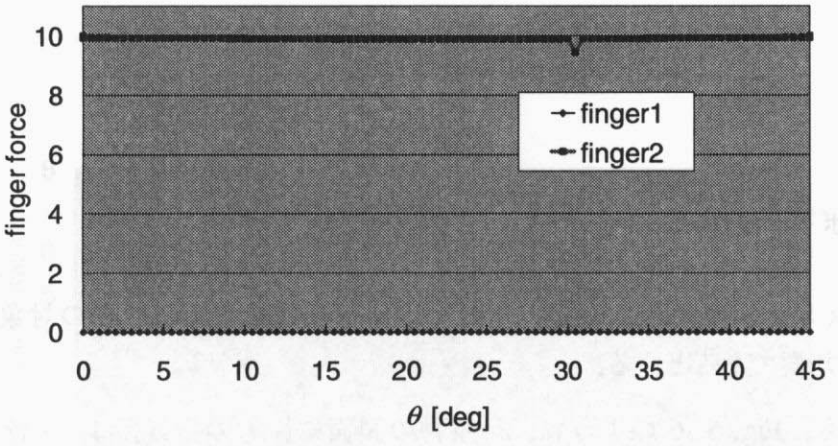


Fig. 6.34: Pivoting 1: Planned Finger Forces

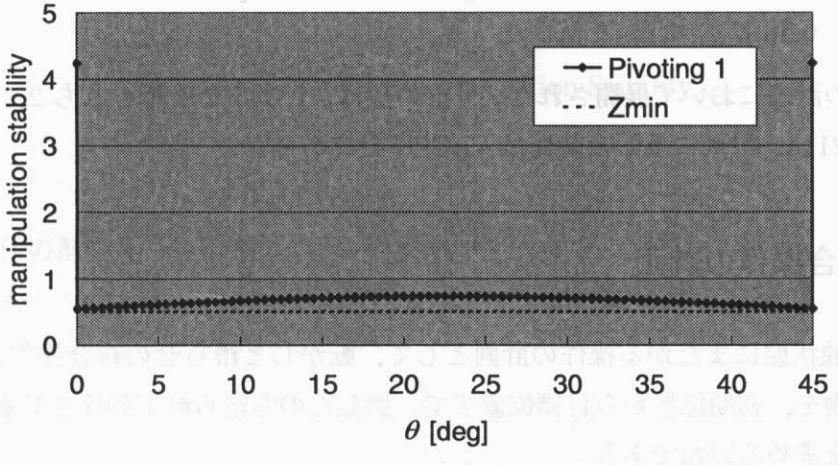
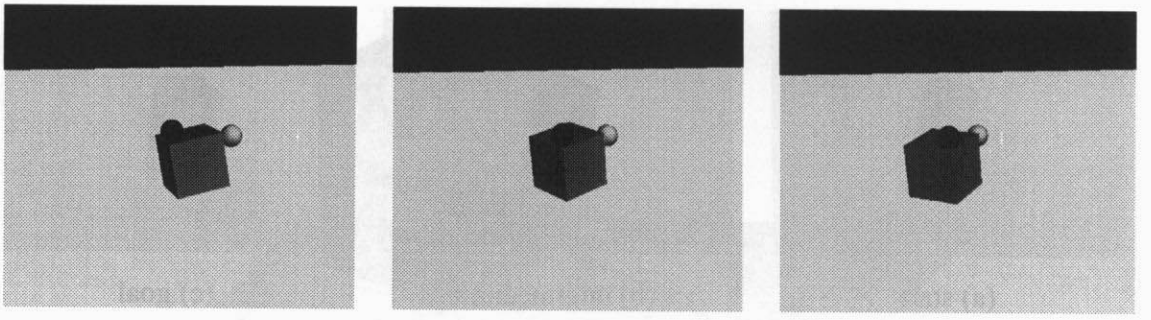


Fig. 6.35: Pivoting 1: Manipulation Stability



(a) start　　(b) intermediate　　(c) goal

Fig. 6.36: Pivoting 2: Planned Operation

立方体のピボット操作

次に，$5 \times 5 \times 5$ の大きさの立方体のピボット操作を考える．この対象物の質量は 1 ($M_o = 1$) で均質であるとする．

この場合も，Fig. 6.36 のように，対象物の側面に片方の指（指 1）を位置制御で添えて，上面をもう片方の指（指 2）が力制御で押さえながら，対象物を旋回させる動作が生成された．このときの対象物上の指位置は，指 1 が $[-1.875,\ -2.5,\ 1.875]^T$，指 2 が $[-1.875,\ 1.875,\ 2.5]^T$ である．また，各指の指令力，および操作の確実性の指標の値の推移を Fig. 6.37，6.38 に示す．

なお，この計画において展開されたノードの数は，1,335,201 個のうち 2,182 個で，有効分岐数は 1.2114，計画に要した時間は 47 CPU 分であった．

6.7.4　複合操作の計画

複数の接触状態にまたがる操作の計画として，転がしと滑らせの複合操作を取り上げる．直方体対象物を，初期位置から目標位置まで，倒したのち滑らせて到達させるためのロボット指の動作を求める問題である．

直方体の大きさは $5 \times 5 \times 10$，質量は 1 ($M_o = 1$) で均質であるとする．

コンフィギュレーション空間の離散化については，前項と同様，対象物上の各面を 7×7 のグリッドに切ることによって，指の配置可能点を設定し，また，対象物の自由度について

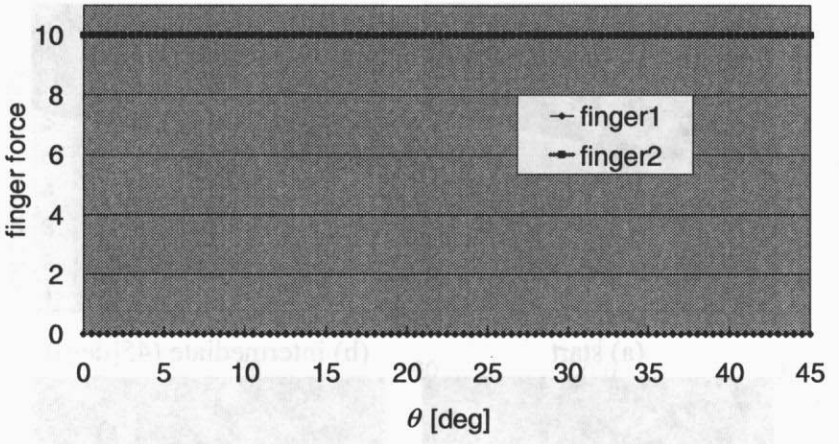


Fig. 6.37: Pivoting 2: Planned Finger Forces

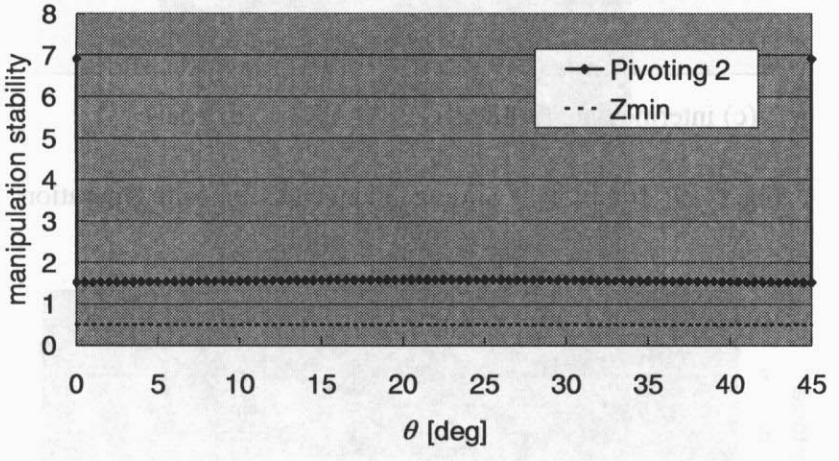


Fig. 6.38: Pivoting 2: Manipulation Stability

は，転がし操作の部分を 31 段階に，滑らせ操作の区間を 10 段階に分割した．これにより，理論的には最大

$$_{7\times7\times6}C_2 \times (31+10) = 1,765,911$$

のノードから成る操作可能性グラフを探索することになる．

平面上に立てた対象物を倒してさらに滑らせる操作を計画した（問題 Tumbling + Sliding）ところ，Fig. 6.39 のように，両側から物体をつまんで倒し，そのまま持ち替え無しで滑らせる操作が生成された．このとき，片方の指（指 1）が位置制御，もう片方の指（指 2）が力制御となり，対象物上の指位置は，指 1 が $[2.5,\ 1.25,\ 1.25]^T$，指 2 が $[2.5,\ 1.25,\ 1.25]^T$ である（問題 Tumbling 1 の結果と同じ）．また，各指の指令力，および操作の確実性の指標の

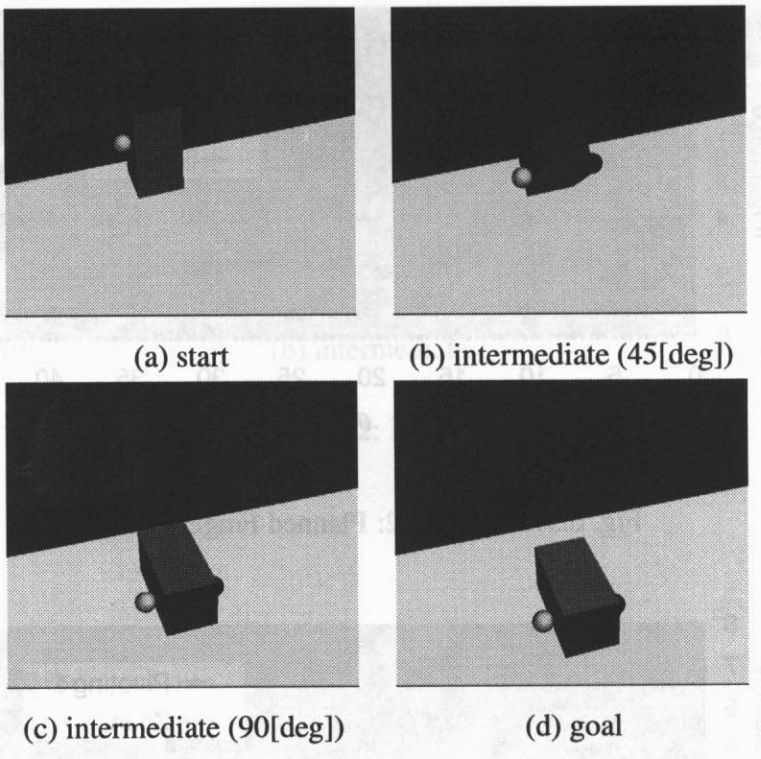


Fig. 6.39: Tumbling + Sliding: Planned Composite Operation

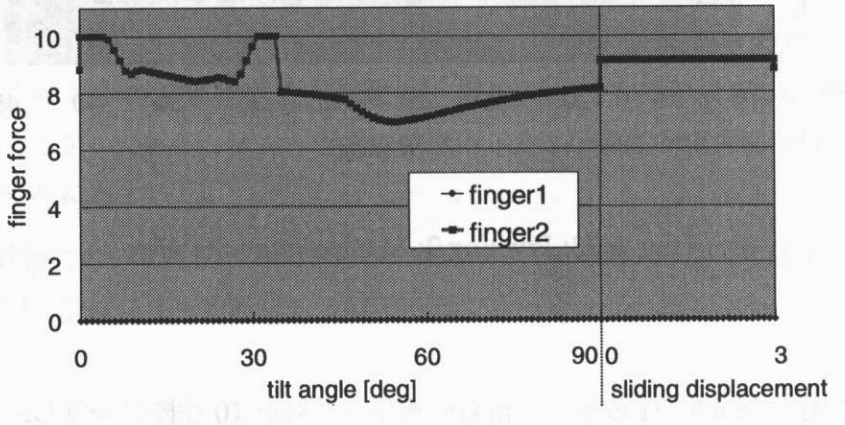


Fig. 6.40: Tumbling + Sliding: Planned Finger Forces

値の推移を Fig. 6.40, 6.41 に示す.

なお, この計画において展開されたノードの数は, 3,133 個で, 有効分岐数は 1.1595, 計画に要した時間は 120 CPU 分であった.

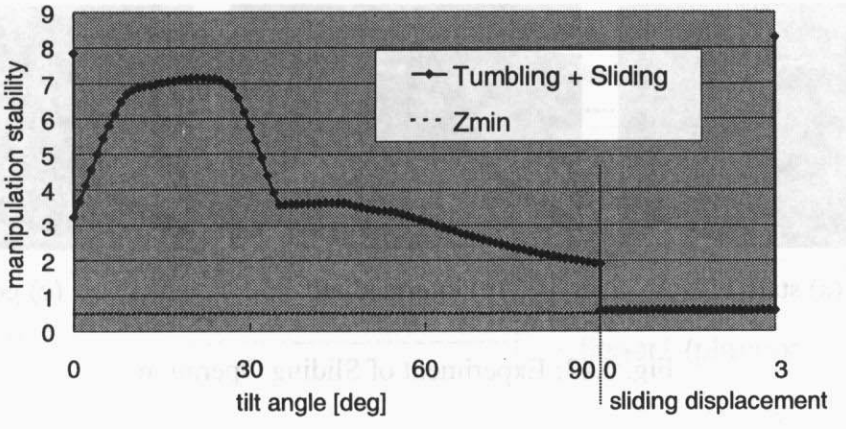


Fig. 6.41: Tumbling + Sliding: Manipulation Stability

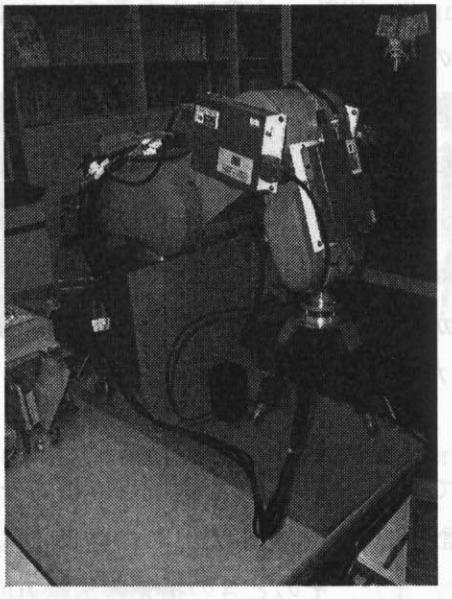
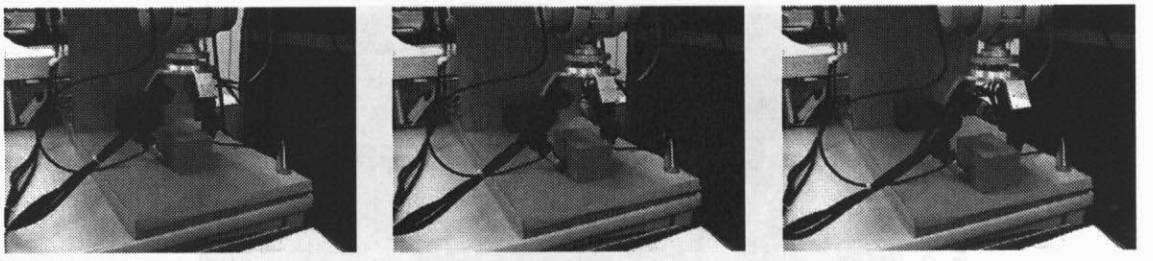


Fig. 6.42: Arm-Hand Mechanism

6.7.5 アーム・ハンド機構による計画結果の実行例

計画アルゴリズムによって生成された計画結果を，Fig. 6.42 のアーム・ハンド機構によって実行することを試みた．この機構は，垂直多関節型の 5 自由度アーム（三菱電機・RV-M2）の手先に，3 本指 3 関節ハンド（安川電機製）を装着した構成になっている．また，指先には 6 軸の力覚センサ（BL オートテック・NANO 2.5/2）が装着されている．システム全体



(a) start　　(b) intermediate　　(c) goal

Fig. 6.43: Experiment of Sliding Operation

は，ART-Linux [石綿] を用いて PC から制御される．

ここでは 60[mm] × 60[mm] × 100[mm] のコルク製の直方体（質量 0.092 [kg]）を対象物とし，ハンドの 3 本のうちの 2 本の指を使ってグラスプレス・マニピュレーションを行うことを考える．予備実験の結果から，対象物と指先との間の摩擦係数を 0.15，対象物と環境との間の摩擦係数を 1.2 と設定して動作計画を行い，得られた計画結果からアーム・ハンドの制御を行った．

平面上で対象物を並進させる操作について計画を行ったところ，指 1 を力制御，指 2 を位置制御として対象物を両側からはさんで滑らせる，という計画結果が得られた．その結果をもとに滑らせ操作を行わせたときの様子を Fig. 6.43 に，そのときの指先の法線方向の力を Fig. 6.44 に示す．

また，転がし操作についても，同様に指 1 を力制御，指 2 を位置制御として対象物を両側からはさんで倒す，という計画結果が得られた．この結果に基づいて転がし操作を行ったときの様子を Fig. 6.45 に示す．また，そのときの指先の法線方向の力を Fig. 6.46 に示す．

どちらの場合も，若干の誤差はあるものの，おおむね計画どおりに制御が行われ，操作を実現することができている．指の制御モードまで含めて計画が行われているので，その結果を実機に適用して実行させるのは比較的容易である．

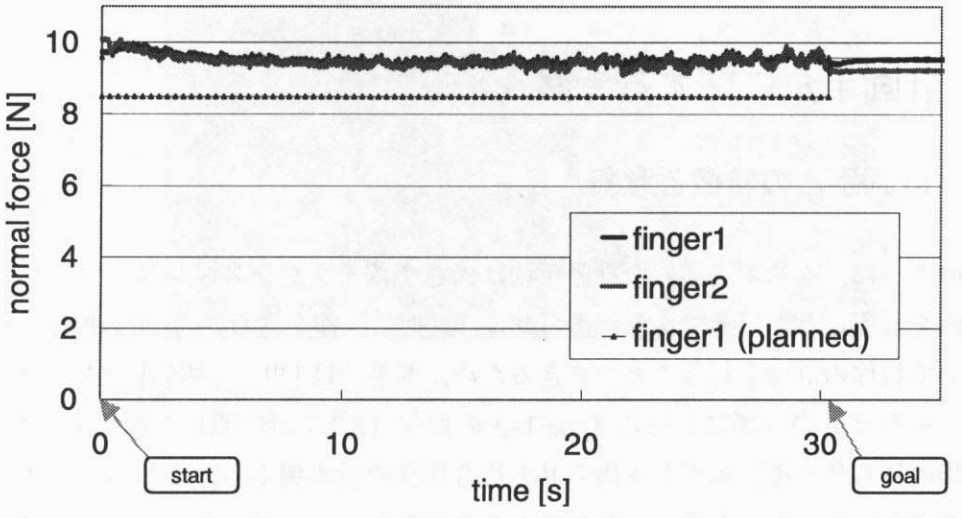
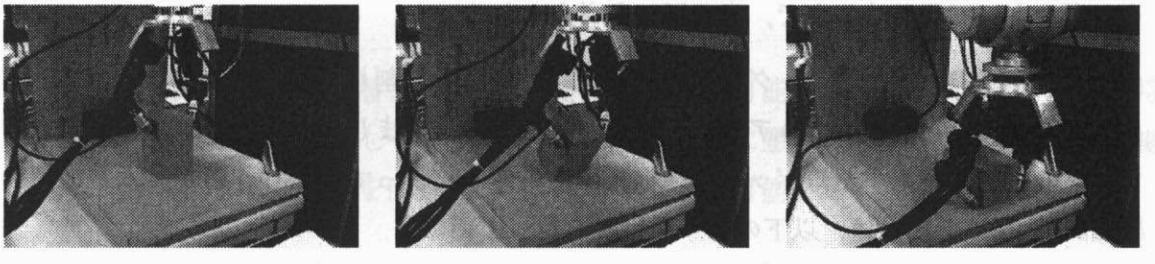


Fig. 6.44: Finger Forces in Sliding



(a) start (b) intermediate (c) goal

Fig. 6.45: Experiment of Tumbling Operation

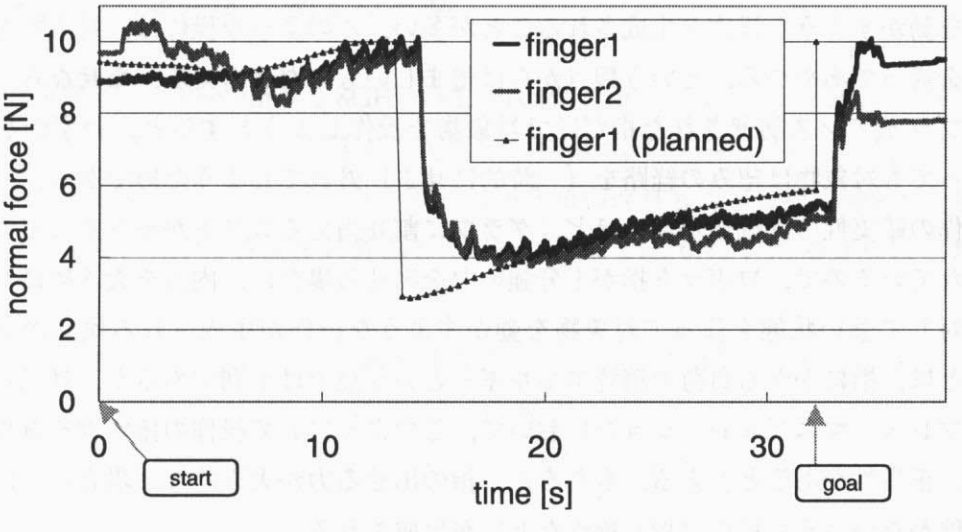


Fig. 6.46: Finger Forces in Tumbling

6.8　計画手法に関する考察

6.8.1　計画手法の特徴と意義

以上のように，本章で提案した計画手法は，さまざまなグラスプレス・マニピュレーションの計画を行うことが可能である．滑らせ操作（押し操作を含む）・転がし操作・ピボット操作について操作の計画を行うことができるため，本手法は [相山 1996a] のグラスプレス・マニピュレーションの分類における Type 1–3 の操作（2.2.3 項参照）を網羅できたと言える．また，指の持ち替えや，転がし＋滑らせの複合操作の計画例から，これらをさまざまに組み合わせることによって，多様かつ複雑なグラスプレス・マニピュレーションの計画の実現の可能性を示すことができた．このように，特定の操作手法に限定されない，一般的なグラスプレス・マニピュレーションの計画の枠組みを構築したことは，器用なマニピュレーションをロボットに実現させる上で，意義深いと考えられる．

また，本章の計画手法は，グラフのアークに評価関数を割り当て，グラフ探索によってマニピュレーション計画を行うアプローチをとっている．望ましい評価関数は，マニピュレーションの目的や使用するロボットの仕様，環境条件等によって変わりうるが，本章で用いた評価関数（6.6.1 項）は，以下のような性質を持っている．

- 5.6.1 項でも触れたように，できるだけ位置制御を使用する操作が計画される傾向がある．具体的には，位置制御された指もしくは環境を，マニピュレーションのガイドとして用い，それに対して力制御された指もしくは重力によって予圧をかけた状態で，対象物を動かすような操作が生成されることが多い．このような操作は，対象物を望みの経路を通ってあやつる，という観点からは望ましいものだと言える．なぜなら，例えばコンプライアンス制御された指だけで対象物を操作しようとすると，わずかな外乱力によっても対象物は望みの経路を（一時的にせよ）外れてしまうためである．
- 操作の確実性の評価値が高いほど，グラフに割り当てるコストが小さくなるように設定されているので，ロボット指が十分強い力を出せる場合は，内力を大きめにして，外乱に対して強い状態を作って対象物を動かすような操作が生成される傾向がある．このことは，指にかかる負荷や消費エネルギーという点では不利であるが，外乱に弱いグラスプレス・マニピュレーションにおいて，このようにして操作の確実性を確保することは，正当性があると言える．もちろん，指の出せる力が大きくない場合は，内力の発生を伴わないような操作（押し操作など）が生成される．
- 指の制御モードの決定は，瞬時的な最適化をベースに行っているので，場合によっては

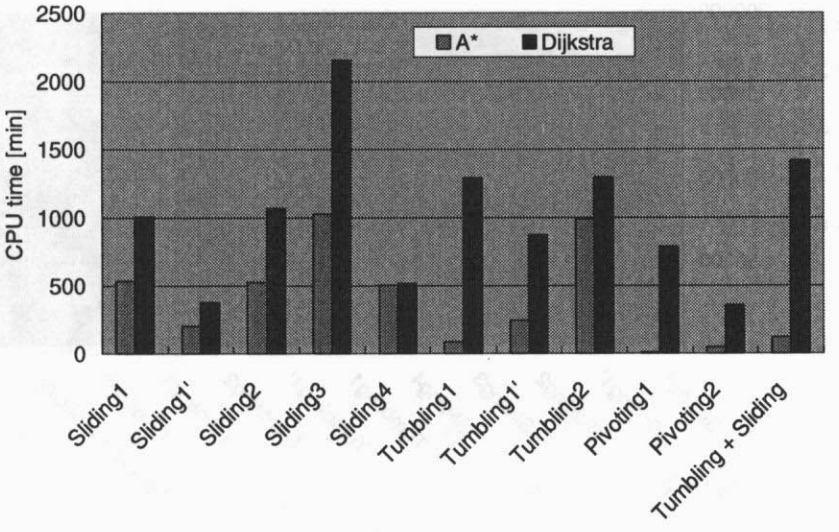


Fig. 6.47: CPU Time for Planning

頻繁に制御モードを切り替えるような解が生成される可能性がある．制御モードを瞬時的に切り替えることは難しく，その切替の過程が操作に悪影響を与える可能性がある場合は，各指の制御モードを固定して計画を行うことも可能である．

以上のことから，本章で使用した評価関数は，万能ではないが多くの場合で妥当な操作を生成すると考えられる．

6.8.2 実際の計算時間に関する考察

前節の計画問題を，Pentium4–2.8GHz の Linux PC で実行した際にかかった計算時間を，Fig. 6.47 に示す．グラフには，A^* 探索の代わりに Dijkstra のアルゴリズムを使った場合（すなわち，ヒューリスティック関数を利用せず，6.6.3 項の改良も適用しない場合）の結果も併せて載せた．また，グラフ探索中に展開したノード数を Fig. 6.48，有効分岐数を Fig. 6.49 に示す．

グラフより，探索中の展開ノード数と計算時間がおおよそ対応していることが見てとれる．そして，ヒューリスティクスを利用した A^* 探索によって，すべての場合について計算時間が改善されていることがわかる．しかし，その改善の度合いについてはばらつきがかなりある．

比較的障害物が少ない状態での転がし操作やピボット操作では，A^* 探索によって計算時

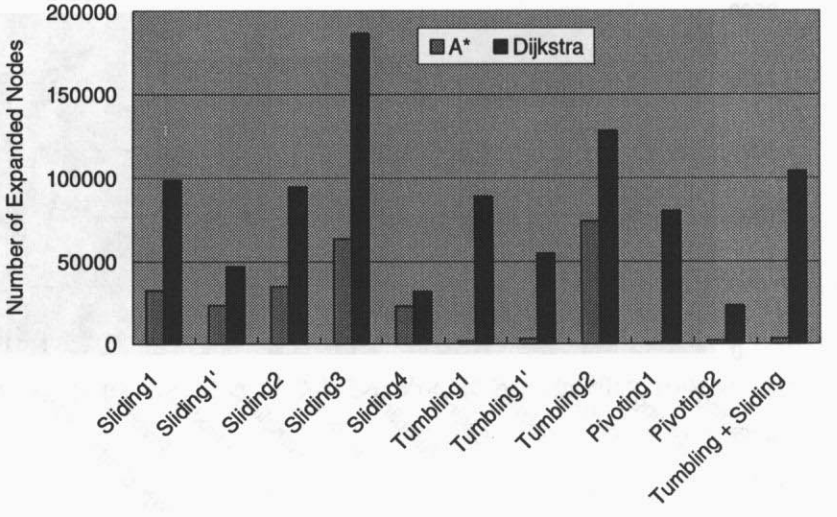


Fig. 6.48: Number of Expanded Nodes in Planning

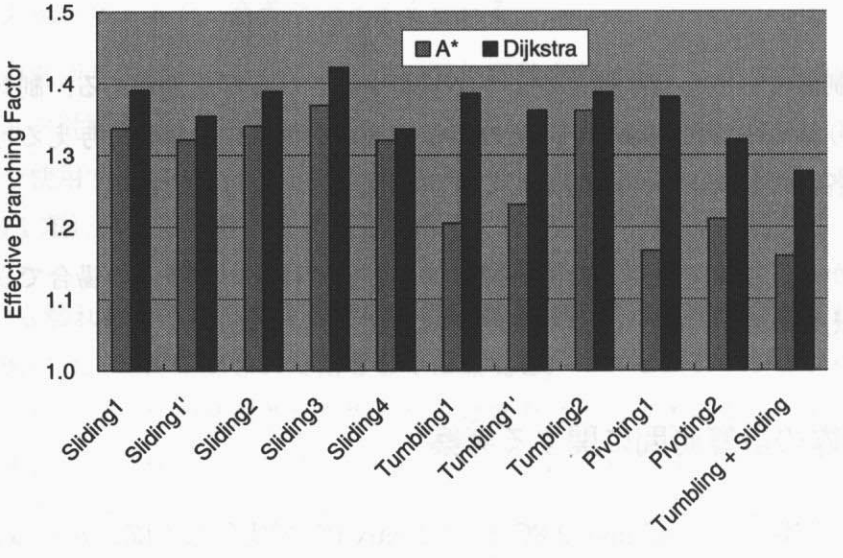


Fig. 6.49: Effective Branching Factors in Planning

間が数分の 1 (Pivoting 2)～100 分の 1 以下 (Pivoting 1) と大幅に短縮されている．これらの場合には，ヒューリスティック関数によるコストの予測が有効に機能している．

一方，障害物が多く持ち替えが必要な場合の転がし操作 (Tumbling 2) ではそれほど改善が大きくないが，これは計画問題そのものが難しい（実行可能解を見つけること自体が難しい）ことに起因するものと思われる．このような問題における計画の高速化は容易でない．

また，全般に滑らせ操作の場合の改善は小さいが，こちらは最適解および最適解に近い解が多数あるためだと考えられる．つまり，多数の候補の中から一つの解を選ぶのに時間がかかっているためである．これは，言わば，計画問題が易しい（実行可能解を見つけるのは簡単だが，多数見つかってしまう）ために計算時間がかかっていると言える．このような場合は，解の最適性を諦めれば，欲張り探索を用いることで計画の高速化が見込める．

6.8.3 理論的な計算量

本項では，提案した計画アルゴリズムの計算量について考察する．ただし，A^* 探索におけるヒューリスティック関数（6.6.2 項）の効果を理論的な計算量の観点から一般的に議論するのは難しく，また 6.6.3 項の改良の効果についても同様である．そこでここでは，A^* 探索ではなく Dijkstra のアルゴリズムを使った場合の計算量を考えることにする．

Dijkstra のアルゴリズムは A^* 探索で $h = 0$ とした場合（ヒューリスティック関数を利用しない場合）に相当する．したがって，Dijkstra のアルゴリズムの理論的な計算量は，本章で示した計画アルゴリズムの計算量の上限を与える，と考えてよい．

同一接触状態内での計画

まず，1 つの接触状態内での操作計画の計算量を考える．対象物の自由度は $d(\leq 6)$，指は n 本だから，コンフィギュレーション空間の次元は $(d+2n)$ 次元になる．このコンフィギュレーション空間 の各次元を X 分割して離散的なノードを作ることとすると，ノードの数は $O(X^{d+2n})$ 個になる

一方アークについては，対象物の自由度方向のアークと持ち替え方向のアークに分けて考える．対象物の自由度の方向には，各ノードは $O(d)$ 個の隣接ノードとアークで接続される．したがって，対象物の自由度方向のアークは，$O(d) \times O(X^{d+2n}) \times 2 = O(dX^{d+2n})$ 本になる（最後に 2 倍するのは双方向にアークを張るため）．持ち替えの方向には，各ノードは $n \times O(X^2) \times 2 = O(nX^2)$ 個のノードとアークで接続されるため，アークの数は $O(nX^2) \times O(X^{d+2n}) = O(nX^{d+2n+2})$ 本になる．したがって，アークの総数は $O(dX^{d+2n}) + O(nX^{d+2n+2}) = O((d+nX^2)X^{d+2n})$ 本となるが，ここで $X \gg d$, $X \gg n$ であることを考慮すると，結局アークの総数は $O(nX^{d+2n+2})$ と見てよい．

一般に Dijkstra のアルゴリズムは，アークの数 n_{arc} とノードの数 n_{node} に対して

$O(n_{\mathrm{arc}} \log n_{\mathrm{node}})$ の計算量である [エイホ 1987]. したがって，この場合の計算量は

$$\begin{aligned} & O(O(nX^{d+2n+2}) \times \log O(X^{d+2n})) \\ & = O(n(d+2n)X^{d+2n+2} \log X) \end{aligned} \tag{6.9}$$

となる.

しかし，本章で提案した計画手法においては，グラフ探索中に用いられる指の制御モードの決定部分（5章）の計算量がかなり大きい．したがって，計画手法の計算量としては，その分を含めた計算量で評価すべきである.

1回の指の制御モード決定にかかる計算量は，5.6.2項で調べたように

$$O(2^n N_{\mathrm{vert}}^3 s n^2 (m+n))$$

程度と見積もられる．ここで N_{vert} は，6次元超球を超多面体近似するときの頂点数，s は摩擦円錐を多角錐近似する際の稜の数，m は環境との接触点数であった.

グラフ探索中のコスト更新部分でこの計算が行われるため，全体での計算量は，グラフ探索の計算量と制御モード決定の計算量の積であると考えられる．1回のコスト更新について，制御モード決定の計算は，最大で P 回行われる場合があるので，全体の計算量は

$$\begin{aligned} & O(n(d+2n)X^{d+2n+2} \log X \times P \times O(2^n N_{\mathrm{vert}}^3 s n^2 (m+n)) \\ & = O(sPN_{\mathrm{vert}}^3 n^3 (m+n)(d+2n) 2^n X^{d+2n+2} \log X) \end{aligned} \tag{6.10}$$

程度と考えられる.

上記の計算量の式の中のパラメータのうち，s, P, N_{vert} は近似計算の精度に関係するパラメータであり，d, m, n, X は直接計画問題の規模を表すパラメータである．後者のうち，非常に大きな値をとりうるのは X のみであることを考えると，この計画アルゴリズムの計算量は X に対する多項式オーダであると言える．ただし，その次数はかなり高い.

複数の接触状態にまたがる計画

次に，複数の接触状態にまたがる計画の計算量を考える．A 個の接触状態と B 個の接触状態遷移からなる，簡略化された接触状態ネットワークが与えられているとする．このグラフのノードの総数は $O(AX^{d+2n})$ である．アークに関しては，同一接触状態内のアークの数の和は $O(AnX^{d+2n+2})$ 本となる．また接触状態の遷移を表すアークの数の和は $O(2BX^{d+2n}) = O(BX^{d+2n})$ 本となる．したがって，アークの総数は $O(AnX^{d+2n+2}) + O(BX^{d+2n}) = O(AnX^{d+2n+2})$ となる.

よって，Dijkstra のアルゴリズムによる探索の計算量は

$$\begin{aligned} & O(O(AnX^{d+2n+2}) \times \log O(AX^{d+2n})) \\ & = O(AnX^{d+2n+2}(\log A + (d+2n)\log X)) \\ & = O(An(d+2n)X^{d+2n+2}\log X)\ (\because X \gg A) \end{aligned} \tag{6.11}$$

となる．

前項と同様にして，操作の確実性の評価の部分を含めて計算量を考えると，

$$\begin{aligned} & O(An(d+2n)X^{d+2n+2}\log X \times P \times O(2^n N_{\text{vert}}^3 sn^2(m+n)) \\ & = O(sPN_{\text{vert}}^3 An^3(m+n)(d+2n)2^n X^{d+2n+2}\log X) \end{aligned} \tag{6.12}$$

となる．これは X の多項式オーダであるが，その次数はかなり大きい．ただし，提案した計画手法では，ヒューリスティック関数（6.6.2 項）を利用した A^* 探索の利用，および本計画問題の特徴を利用したその改良（6.6.3 項）などの対策が盛り込まれている．また，操作の確実性の計算の部分では，制御モードの組み合わせを調べる回数を減らす工夫（5.4.4 項）がされている．(6.12) 式にはこれらの効果を無視して求めたものであるので，実際的な計算量はそれより小さくなる可能性がある．

6.8.4 計算時間の改善策

以上のように，提案した計画アルゴリズムの計算量はかなり多い．計算時間を短縮するための工夫として考えられるものを以下に挙げる．

1. 双方向探索を用いる．

   ここで扱う計画問題では，目標状態がはっきりしているので，双方向探索を利用することも可能である．目標状態付近では物理的に取りうる状態が少ないようなマニピュレーションの計画では，計算時間を大きく短縮できる可能性がある．

2. 欲張り探索に近づける．

   最適解の保証を諦め，ヒューリスティック関数の重みを大きくして欲張り探索に近づけることによって，計算時間を短縮できる可能性がある．この場合，具体的には式 (6.3) の代わりに $f(n) = w_g g(n) + w_h h(n)$ （w_g, w_h は適当な重み係数で $w_g < w_h$ とする）に基づいて探索を行うことになる．特に，実行可能なマニピュレーション方法が多数存在する場合（例えば，どのような指配置でもマニピュレーションが可能な場合）には，すべての可能性を調べることなく探索を終了することで計算時間が改善されることが予想される．ただし，適切な重み係数を決定するのが難しいこと，また問題によっては

却って計算時間が悪化する可能性があることから，最適性が失われることの影響も含め，導入には注意を要する．

3. 力学的に等価なコンフィギュレーションを検出し，計算結果を再利用する．

   提案した計画アルゴリズムの手続きの中で時間がかかるのは，指の制御モードを決定するために線形計画問題を解く部分である．現在の実装では，各コンフィギュレーションごとにこの計算を行っている．したがって，平面上での滑らせ操作の計画などで，周囲に障害物がない場合は，力学的には同じ状態を表すノードが多数生成され，それぞれに対してまったく同様な計算を繰り返していることになる．そこで，探索中の指の制御モードの計算結果を記憶しておき，すでに調べたノードと力学的に等価なノードを検出したら，過去の計算結果を再利用することによって，計算の重複をなくすことが考えられる．ハッシュを利用するなどして，等価なコンフィギュレーションの検出を高速に行うことができれば，大幅な計算時間の短縮が見込める．

4. ノードのサンプリングを適応的に行う．

   現在はコンフィギュレーション空間を固定的な刻み幅で離散化することによってノードのサンプリングを行っている．しかし，この方法では，刻み幅を不必要に細かくし過ぎて計画が非効率的になったり，粗い刻み幅のために計画が失敗する，などの事態が起こりうる．そこで，例えばProbablistic Roadmap (PRM) [Kavraki 1996] のような方法を利用することが考えられる．コンフィギュレーションのサンプリングを適応的に行うことにより，生成されるグラフの大きさを抑え，計算時間を短縮できる可能性がある．もちろんこの場合は解の最適性は犠牲となる．本章で扱う計画問題のコンフィギュレーション空間は素直な空間ではないため，必ずしもPRMの実装は容易ではないが，接触動作の計画にPRMを適用した例 [Ji 2001] が参考になると思われる．

6.9 おわりに

本章ではグラスプレス・マニピュレーションの計画問題を扱い，ロボット指の動作計画手法を提案した．滑らせ操作（押し操作を含む）・転がし操作・ピボット操作のそれぞれの計画例を示したことで，この計画アルゴリズムが，[相山 1996a] のグラスプレス・マニピュレーションの分類における Type 1–3 の操作すべてを扱うことができることを確認した．また，持ち替えや，複数の操作を組み合わせた複合的なマニピュレーションの計画例も併せて示した．これらの組み合わせによって，多様かつ複雑なグラスプレス・マニピュレーションの計画が可能になるものと考えられる．また，計画結果の実機ロボットへの適用例も示した．これにより，一般的なグラスプレス・マニピュレーションの計画を行うための，一つの基本的な枠組みを示すことができたと言える．

現状の計画手法の欠点としては，計算量が多いことが大きな問題である．提案したアルゴリズムには改良の余地が多数残っているが，6.8.4 項でも触れた Probablistic Roadmap (PRM) [Kavraki 1996] や Rapidly-Exploring Random Trees (RRT) [LaValle 2001] などのランダム探索の手法を取り入れることも有効であると考えられる．

また，指と対象物の間の滑り・転がりを扱えない点も問題である．特に指と対象物の間の転がりの利用は，あやつりの能力を高める上で重要であることが指摘されており [吉川 1996b, 黄 1999]，これを併せて扱えるようにすることも大きな課題の 1 つである．