制約条件に基づく機構モデルの迅速生成と対象形状の対話的操作

吉川浩一

(1)

平成4年度 博士論文

制約条件に基づく機構モデルの迅速生成と対象形状の対話的操作

指導教官 木村文彦 教授

東京大学大学院 工学系研究科
精密機械工学専攻

吉川 浩一

概要

本研究では，機構設計における部品の形状設計を支援するために，機構全体としての整合性を維持しながら容易な形状の操作を可能とする，機構設計に適した対象表現法を提案する．機構設計においては，部品形状を設計していく際に，その部品が満足すべき機構部品としての条件を考慮しながら形状を決定していく必要がある．したがって，設計対象の形状情報と機構情報を密接に関連づけることが可能な対象表現を必要とする．本研究では，部品形状は線分と円弧からなるセグメントで表す．また機構情報を基本的な設計変数をパラメータとする代数式（制約式）により表現する．さらに両者を媒介する対象表現法として骨格形状モデルを新たに導入することで，形状情報と機構情報を共に扱える対象表現法を実現する．これにより，従来では設計者が管理していた部品形状と機構情報の整合性を，計算機上で扱うことが可能となる．

骨格形状モデルとは「部品形状を針金状に縮退させた形状を点と直線分で表したモデル」である．部品形状と骨格形状間では，部品形状のセグメントを骨格の点や線分に対応づけ，骨格形状の変更を２つの変形操作 (1) 骨格による形状要素の回転移動，(2) 形状要素の骨格軸方向への伸縮，により部品形状に展開する．これにより大域的な部品形状の変更が可能となる．機構情報と骨格形状間では，制約式中のパラメータ値を骨格のノード位置に対応させ，制約式を解いて得られたパラメータ値を骨格形状に反映させる．制約評価手法としては，運動による制約変化をともなう機構を表現するために成立条件つきの制約（区間制約）を導入し，さらに設計変数の決定に対して制約が不足している場合でも評価可能な方法を提案する．

また本手法に基づく機構部品の形状設計支援システムを実際に試作し，機構設計における部品形状設計に適用することでその有用性を確認した．

もくじ

1 序論 **1**

1.1 はじめに 1

1.2 既存の対象表現法 2

1.2.1 設計対象の抽象度 3

1.2.2 設計対象の種類 5

1.2.3 モデルの表現力 6

1.2.4 モデルの操作性 7

1.3 機構の設計プロセス 7

1.4 本研究の目的と実現手法 11

1.4.1 研究の目的 11

1.4.2 本研究で提案する対象表現 12

1.5 本論文の構成 15

2 機構モデルの表現法 **18**

2.1 機構モデル 18

2.2 従来の対象表現と問題点 19

2.2.1 組立品の表現法 19

2.2.2 挙動の表現法 21

2.2.3 配位空間による挙動の表現 23

2.2.4 形状の表現法 28

2.2.5 従来の対象表現のまとめ 29

2.3 機構モデルの構成要素 31

2.3.1 属性モデル 33
2.3.2 制約モデル 35
2.3.3 骨格形状モデル 39
2.3.4 概略形状モデル 42
2.4 構成要素間の対応関係 44
2.4.1 属性モデルと制約モデルの対応関係 44
2.4.2 制約モデルと骨格形状モデルの対応関係 44
2.4.3 骨格形状モデルと概略形状モデルの対応関係 45

3 機構モデルにおける制約条件の評価 47
3.1 制約モデルの表現 47
3.1.1 制約表現とデフォルト値 47
3.1.2 制約条件の分類と定義方法 50
3.2 デフォルト値を含む制約条件の解法 55
3.2.1 デフォルト値の指定方法 55
3.2.2 パラメータの分類 58
3.2.3 デフォルト値を用いた制約評価の方法 60
3.3 区間制約を含む制約条件の解法 64
3.3.1 切換えを必要とする制約条件の表現 66
3.3.2 制約条件の組合せ 67
3.3.3 評価手順の AND/OR 木表現 69
3.3.4 AND/OR 木による区間制約の評価 71

4 制約に基づく形状操作 73
4.1 形状変形操作の要件 73
4.2 骨格形状モデルによる形状変形 74
4.2.1 骨格形状モデルの操作 74
4.2.2 概略形状モデルの操作 81
4.2.3 骨格形状モデルと概略形状モデルの対応関係 82
4.2.4 概略形状モデルによる骨格形状モデルの変更 82

4.2.5 骨格形状モデルによる概略形状モデルの変更 83
4.3 非干渉条件に基づく形状変形 92
4.3.1 ゆるやかな拘束条件に基づく形状生成 93
4.3.2 骨格形状を利用した非干渉形状の生成 95

5 機構モデリングシステムの試作 **98**
5.1 システム構成 98
5.1.1 2次元スケッチャ 98
5.1.2 対偶制約ライブラリ 108
5.1.3 属性・対偶モデル操作ユニット 109
5.1.4 骨格形状モデル操作ユニット 110
5.1.5 概略形状モデル操作ユニット 110
5.1.6 骨格形状／概略形状モデル変換ユニット 111
5.1.7 対偶制約評価ユニット 111
5.1.8 非干渉制約評価ユニット 111
5.2 実行例 111
5.2.1 ストロボのポップアップ機構の設計例 111
5.2.2 制約条件式の追加による形状変更 122
5.2.3 部品の干渉回避形状の生成例 128
5.2.4 制約条件の切り換えを含む機構の評価 136

6 結論と展望 **143**
6.1 結論 143
6.2 展望 145

図一覧

1.1 機構の設計プロセス 8
1.2 ゴム変形モデルによる部品形状の変形 15

2.1 仮想リンクを用いた組立品の表現 20
2.2 接触関係グラフによる組立品の表現 21
2.3 MATHPAK における制約の代数表現 22
2.4 Concept Modeler で用意されている機械要素 24
2.5 機構の運動に関する仕様の変更と形状変更 25
2.6 配位空間による挙動の表現 26
2.7 形状要素間の依存関係による形状表現 29
2.8 形状特徴による形状表現 30
2.9 機構モデルと構成要素の対応関係 34
2.10 歯車の属性モデル 35
2.11 歯車機構 37
2.12 制約モジュールのアイコンの例 38
2.13 制約モデルのアイコンと骨格形状 40
2.14 骨格変形操作の例 41
2.15 ゴム変形モデルによる部品形状の変形（再掲） 45
2.16 骨格接続部におけるゴム変形モデル 46

3.1 接続関係の制約表現 52
3.2 概略形状を用いた初期デフォルト値の設定 57
3.3 VTR テープローディング機構 65

3.4 制約条件評価のための AND/OR 木 69
3.5 AND/OR 木の構成過程 71

4.1 骨格ノードの生成・消去操作 75
4.2 骨格アークの生成・消去操作 75
4.3 骨格ノードが消去できない例 (1) 77
4.4 骨格ノードが消去できない例 (2) 78
4.5 骨格ノードの移動操作 79
4.6 不適当な骨格ノードの移動 80
4.7 骨格アークの移動操作 81
4.8 骨格の勢力圏とセグメントの種類 83
4.9 単位骨格による形状変更 86
4.10 分岐ノードの勢力圏にあるセグメントの変形例 89
4.11 S^J の強制挿入が必要な形状例 90
4.12 概略形状の骨格に対する写像 92
4.13 構造モデルによる形状生成 94
4.14 構造モデルによる形状生成 95

5.1 試作システムの構成図 99
5.2 システムの初期画面 100
5.3 一眼レフカメラのストロボ - ポップアップ機構 112
5.4 ストロボ - ポップアップ機構のスケッチ 112
5.5 試作システムによるスケッチ例 113
5.6 スライダ - リンク機構の定義例 114
5.7 制約条件に基づく機構部品の位置の算出例 115
5.8 ストロボ部分の移動経路の表示 116
5.9 4 節リンク機構の定義例 117
5.10 制約条件に基づく機構部品位置の算出例 (2) 118
5.11 4 節リンク機構を用いた場合の移動経路表示 119
5.12 リンク形状の定義例 120

5.13 リンク形状を含む機構部品の移動経路表示 121
5.14 スライダ・リンク機構のスケッチ例 123
5.15 骨格形状モデルによるスライダの変形 124
5.16 デフォルト値を用いたリンクの変形結果 125
5.17 設計者定義の制約条件によるリンクの変形結果 126
5.18 スライダ-リンク機構の動作シミュレーション(1) 127
5.19 テープヘッド，歯車とスライダの干渉 128
5.20 スライダの非干渉形状の生成 129
5.21 制約条件による部品位置の導出 130
5.22 VTR テープ・ローディング機構のスケッチ 131
5.23 スライダの非干渉形状(2) 132
5.24 リンクとテープヘッドの干渉 133
5.25 リンクの非干渉形状 134
5.26 骨格形状を用いた部品形状の変更 135
5.27 不等式条件を含む制約条件の評価 137
5.28 不等式条件を含む制約条件の評価例(2) 138
5.29 制約条件の切り換えを含む機構の動作シミュレーション 139
5.30 2個のスライダを含む機構の評価(1) 140
5.31 2個のスライダを含む機構の評価(2) 141
5.32 VTR テープ・ローディング機構の動作シミュレーション . . . 142

表一覧

1.1 対象表現法の分類 2
1.2 設計対象の抽象度による対象表現法の整理 3
1.3 設計対象の種類による対象表現法の整理 5
1.4 モデルの表現力による対象表現法の整理 7
1.5 本研究で提案する対象表現の特徴 13

2.1 機構モデルの構成要素 32

3.1 制約条件の分類 55
3.2 制約条件の評価におけるパラメータの分類 60
3.3 パラメータタイプの変更 60

4.1 骨格形状モデルに対する操作 74
4.2 頂点の分類 91

第1章

序論

1.1 はじめに

機械製品の設計・生産プロセスを計算機支援することは，近年ますます重要になってきている．従来から，CAD（Computer Aided Design）やCAM（Computre Aided Manufacturing）の研究として，設計・生産過程を計算機によって支援する技術が研究されているが，対象となる問題が広範囲に及んでいるので研究方針もさまざまである [Finger 89a, Finger 89b]．たとえば，設計に関連するものだけでも次のようなものが挙げられる．設計をよりよく理解するために，実際に行われている設計に関する知識をまず獲得しようとする方針をとる設計プロトコルの研究．設計のプロセスを一般化し，効率的な設計のための指針を示す設計方法論や，設計の本質を解明することを目的とする設計学 [吉川 77, 吉川 79]．また設計対象物をより洗練されたものにするために，設計結果をいろいろな方法で評価することを研究対象とすれば，設計を進める上で欠くことのできない各種の解析の研究となる．

このように，計算機による設計支援の研究方針は多岐にわたるが，上記以外の計算機支援を実現する一つの有効な考え方として「モデリング」に基づく方法論がある．これは設計・生産の対象となっている機械製品のモデルを，計算機が利用できる方法で計算機内部に構成し，設計・生産過程において必要な情報を生成していく方法である [PR 90]．設計者は，設計対象を詳細化していく過程で計算機内に表現されたモデルを種々の方法で評価する．した

がって，対象モデルが表現しなければならない情報の種類は非常に多く，さらにそれらの情報が全体として整合していなければならない．しかし，そうしたあらゆる種類の情報を扱えるモデルを表現する方法は現状ではない．

1.2 既存の対象表現法

前述したように，あらゆる種類の情報を扱える万能な対象表現法はない．換言すれば，これまでに研究されてきた対象表現法はいずれも扱える情報の範囲が限られていることになる．本研究では，機構を扱うのに適切な対象表現法を提案することを目的としているが，本研究で提案する対象表現法もまた万能な表現方法ではない．そこでその特徴を明らかにするために，まず既存の対象表現法の特徴とその問題点を以下で整理する．

ここでは機械設計を対象とし，これまでに提案されている対象表現法を次のような三つの視点で整理する．

1. 設計対象の抽象度
2. 設計対象の種類
3. モデルの表現力

対象表現法の特徴を整理する際の視点としてはこれらの視点以外にも考えることができるが，本研究で示す対象表現法の特徴を明確にする視点として上記のような視点を選ぶ．表 1.1に，対象表現法を視点ごとに分類した例を示す．

表 1.1: 対象表現法の分類

分類の視点	分類例
設計対象の抽象度	概念設計，基本設計，詳細設計
設計対象の種類	個別部品／組立品，運動有／無
モデルの表現力	幾何情報，機能，挙動，設計者の意図

1.2.1 設計対象の抽象度

設計対象は設計が進むにつれて次第に具体化されていく．たとえば，設計の初期段階では，設計対象は機能的な要求仕様によって表現されている．次に，機能的な表現から各機能を実現する基本的な機構に具体化される．このレベルでの対象表現では，部品の支持方法は考慮されていないので，各部品が空中に浮んでいるような表現になる．そこで，さらに各部品を支持する部品や全体を収納するケーシングなどまで具体化され，実際に製造可能なレベルの対象表現へと展開していく．

このように設計対象の抽象度は，機能レベルから製造が可能となるような具体化されたレベルまで変化していくが，対象表現法を分類する名称として，抽象度の違いを明示できる適切なものがない．そこで以下では設計過程を分類する際によく用いられる，概念設計段階，基本設計段階，詳細設計段階という名称を代りに用いる．つまり，概念設計段階には機能的な対象表現が，基本設計段階には設計対象の基本的な構造の表現が，そして詳細設計段階には製造可能なレベルまでの対象表現が，それぞれ対応していると考える．この分類にしたがって，設計対象の抽象度別に対象表現の問題点を整理し表 1.2に示す．

表 1.2: 設計対象の抽象度による対象表現法の整理

抽象度	問題点（現状）
概念設計段階	機能的表現された対象の操作が困難，実体化（基本設計段階）への展開が困難，等.
基本設計段階	未定義部分を含む対象の表現が困難，付加的な条件（設計者の意図）の表現が困難，等.
詳細設計段階	問題領域ごとに適当な対象表現が存在．より精緻な対象表現法の開発などが課題.

まず概念設計段階の対象表現については，機能的な仕様に基づく対象の表

現が困難である点が問題点として挙げられる．より正確には，機能的な表現による対象記述は可能だが，計算機で操作可能な表現が困難な点に問題がある．設計の初期段階においては，設計仕様が機能的な条件によって実際に表現されることから，機能的な対象表現は可能であるといえる．しかし，通常そうした設計仕様は自然言語によって表現されていると考えられるので，それをそのまま計算機内に入力しても，計算機によって操作することが困難である．また仮にそのような情報が操作可能であっても，機能とそれを実現する実体との関係は明らかではない．したがって，概念設計段階で表現された対象を発展させる方法でその対象を実体化していくことは困難である．

次に基本設計段階の対象表現に関しては，まず未定義部分を含む対象が扱えない点に問題があるといえる．基本設計段階では，概念設計段階での結果を受けて要求機能を実現する機構を決定していく．この過程で，一度に設計対象全体を決定することができるわけではない．したがって，具体化できずにあいまいなまま残された部分が存在している状態の設計対象を表現する必要がある．しかし，従来の対象表現法では，未定義部分を含んだ不完全な状態の対象を扱うことができない．また，概念設計段階で得られた条件だけから機構の構造を決定して行くことはできない．実際には，さらに条件を追加することで実体化していく．こうした付加的な条件は設計者の意図を表していると言ってもよく，設計対象に対する条件として表現することは重要である．しかし，設計者が任意に与える条件を扱うことのできる対象表現法は確立されていない．

最後に詳細設計段階の対象表現に関しては，既に多くの対象表現法が提案されまた実際に利用されている．たとえば，設計対象の幾何情報を表現する方法としては，境界表現法（boundary representation）やCSG（constructive solid geometry）などが広く用いられている．さらに最近では，曲面形状の表現方法が発達してきており，自動車の車体設計などに応用されている．また，強度計算などの解析を行うための対象表現としては，有限要素法や境界要素法などが用いられている．このように，詳細設計段階の対象表現としては問題領域ごとに定着している表現法が多い．

しかし，現状の対象表現法では詳細度が不足している部分があり，実物で生じる可能性のある問題を検討するためには，より精緻な表現方法が必要である．たとえば，幾何情報に関して言えば，実際の部品形状は誤差を含むため，部品の寸法には公差を指定しなければならない．ところが，従来の方法では理想的な形状を表現することしかできないという問題がある．

1.2.2 設計対象の種類

既に述べたように，今のところあらゆる種類の情報を扱うことのできる対象表現法はない．たとえば，個別部品を対象としている場合もあれば，複数の部品で構成される製品全体すなわち組立品を対象としている場合もある．また組立品の中でも，各部品の位置関係が変らない対象と，機構のように相対的な位置関係が変化する対象がある．そこで，対象表現法を分類する視点として「対象の種類」をとりあげる．対象の種類によって扱わなければならない情報が異なり，その結果，問題となる点もまた対象の種類によって異なるからである．表 1.3に設計対象の種類と各々の問題点を示す．

表 1.3: 設計対象の種類による対象表現法の整理

設計対象の種類	問題点（現状）
個別部品	他部品との不整合
組立品	部品間の関係指定が煩雑
可動部品	挙動に対する条件指定が困難，対象が限定

従来の対象表現では，個々の部品を対象としているものが多い．部品単位での対象表現の問題点は，ある部品の形状などを変更する場合，他の部品との整合性が保証できない点である．一般に機械を構成している部品は，それぞれ単独で設計することはできない．しかし，部品単位の対象表現では部品間の関係を扱えず，部品どうしの整合性は設計者が管理しなければならないという問題がある．これに対し，組立品を表現する方法についても研究され

ている．しかし，従来の組立品の表現では，部品間の関係を指定するのが煩雑であるという欠点をもつ．

部品が運動するような場合，組立品の場合と同様に部品間の接続関係を表現しなければならない．したがって，組立品の対象表現の問題と同じように，部品間の関係指定が煩わしい欠点がある．さらに，要求された挙動を実現するためには，部品形状など部品間の接続関係以外の条件を指定しなければならない．しかし，これまでの対象表現では，こうした挙動に関する条件を指定するのが難しいと言う問題がある．また対象となる機構を限定して扱う表現法も多く，対象の自由度の点で問題がある．

1.2.3 モデルの表現力

対象表現法が扱わなければならない情報としては，部品形状といった幾何情報をはじめ，設計対象の機能や挙動，また設計者の意図などさまざまである．したがって，こうした多様な情報のなかでどのような情報が表現できるかという，対象表現法の表現力は重要である．ただし，いろいろな情報を単に表現することができるだけでは多様な情報を扱えるとはいえず，さらに，そうしたさまざまな種類の情報を相互に関連づけて扱えることができなければならないことは言うまでもない．なぜなら，どのような種類の情報であっても単独で決定することはできず，各々の情報が全体として整合していなければならないからである．

この点に関しては，幾何情報，機能，挙動など，それぞれの問題領域ごとに対象表現法が多く研究されてきた．しかし，そうした従来の対象表現法は，複数の問題領域をカバーすることができないという問題点があり，これまでの対象表現法を統合するような表現方法については，これからの研究が待たれている．また，設計の進行にともなって，表現される設計対象の抽象度は具体的なものへと変化して行くが，概念設計から基本設計，詳細設計へと連続的に展開して行ける対象表現がない．つまり，従来の対象表現法では，モデルの発展性という観点からも表現力が不足しているといえる．表1.4に，これまでの対象表現法におけるモデルの表現力に関する問題点をまとめて示す．

表 1.4: モデルの表現力による対象表現法の整理

表現力	問題点
統合性	対象表現が問題領域ごとに孤立
発展性	設計過程にともなう対象表現の連続的な展開が困難

1.2.4 モデルの操作性

これまで述べてきた対象表現法を分類する観点ではないが，またモデルを構築していくことを含めて，モデルを容易に操作できることも重要である．なぜなら，モデル操作があまりにも繁雑な場合，設計そのものよりもモデルを構築したり変更することに多くの時間を費やすことになり，設計を支援するのではなく逆に設計を阻害してしまうことも考えられるからである．

たとえば，詳細設計段階での幾何情報の表現法として境界表現法や CSG 表現を示したが，これらの表現法に基づいた形状モデルを構成したり変形するためには複雑な操作が必要であり，これを実行することは容易ではない．また，前述したように組立品を表現する場合にも，部品間の関係を指定するのが煩雑なことから，対象物のモデルを構成していくことは容易ではない．このように操作性の問題点は，その対象表現法が設計者にとってかなり詳細な情報を必要とすることに主な原因があると考えられる．

1.3 機構の設計プロセス

1.2では，設計過程をおおまかに概念設計段階，基本設計段階，詳細設計段階とに分類した．ここでは，本研究で仮定している機構の設計プロセスについてより明確にしておく．本研究では機構の設計過程として，図 1.1に示すような5つの段階からなる設計プロセスを考えている．以下では各段階ごとに，その段階で行われる設計の内容を説明する．

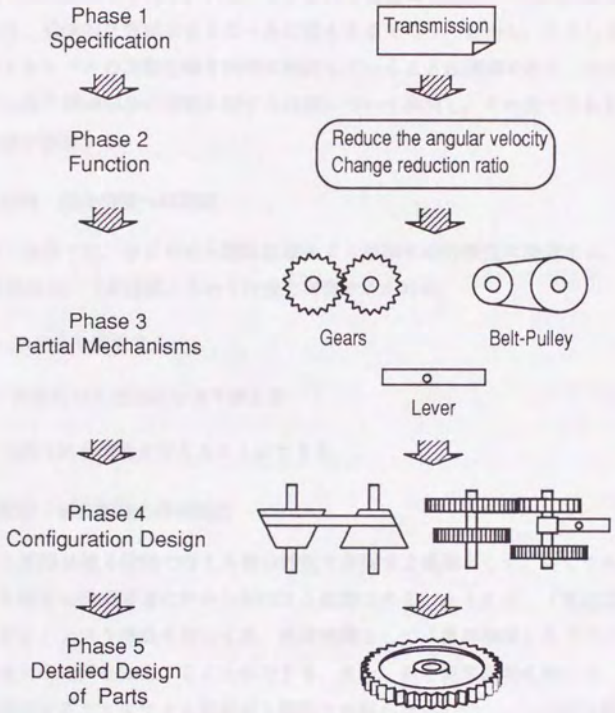


図 1.1: 機構の設計プロセス

第 1 段階：設計仕様の決定

まず第 1 段階で設計仕様が与えられる．図 1.1の場合，「変速機」という仕様が与えられている．本研究は設計対象として機構（または機構部品）を考えているので，このように「運動」に関する仕様に着目する．

もちろん部品の運動以外に関する条件が，設計仕様としてより詳細な仕様

が与えられることも考えられる．たとえば「重量は 200 グラム以内」などといった，具体的な数値をともなった仕様もあるだろう．しかし，こうしたさまざまなレベルの設計仕様を同時に検討していくことは困難である．この段階では特に機構部品の挙動に関する仕様について検討し，その後でそれ以外の仕様を検討する．

第２段階：部分機能への展開

第２段階では，与えられた設計仕様をより詳細な部分機能に展開する．図 1.1の例では，「変速機」という仕様を実現するために，

- 角速度を変更する
- 角速度の入出力比を切り換える

などの部分的な機能を考えることができる．

第３段階：部分機構の候補抽出

第３段階は第２段階で考えた部分機能を実現する機構として，いくつかの機構を既存の機構要素の中から抽出する段階である．たとえば，「角速度を変更する」という機能に対しては，実現機構として「歯車機構」や「ベルト車」などの機構を抽出することができる．また，ある要求機能に対して，それを実現することができる機構が１種類とは限らないので，一つの部分機能に対して複数の機構を候補として抽出する．

これに対して，部分機能を実現する機構が既存の機構のなかに存在しない場合は，その機構設計は発明的な新規設計となるが，本研究ではそうした発明的な設計は扱わない．なぜならば多くの場合，既によく知られている機構要素を用いた設計が行われていると考えるからである．ただし，機構要素として新規なものを扱わないことは，必ずしもある既存の設計例をもとに部分的な改良を行なう設計だけを対象としていることを意味するわけではない．むしろさまざまな機構要素を用いて，自由に機構を構成していく設計の支援を本研究では考える．

第4段階：機構の基本設計

第4段階では，次のようなことが行なわれる．

1. 機構の構造の決定

2. 機構要素の基本的な設計変数の決定

本論文ではこの段階を「基本設計段階」と呼ぶことにする．

まず，第3段階で抽出された機構要素間の関係を定義することによって，目的とする機構の構造を設計する．機構の構造は，機構の構成要素とその要素間の位置関係によって表される．つまりこの段階では，要求仕様として与えられた運動を実現するために，第3段階で得られた機構要素の中からどの要素を用いるかを決定する．さらに，利用することが決まった機構要素間の接続方法を検討し，その接続関係に従ってどのように配置するかを決定する．

このようにして設計対象である機構の構造を決定していくが，この段階で扱われる機構の構造は必ずしも一つに限らない．第3段階で抽出される機構要素には複数の候補があり得るし，同じ機構要素を利用する場合でも，それらを組み合わせる方法にはいくつかの候補があり得る．したがってこの段階では，単に機構要素を組み合わせるという意味で機構の構造を決定するだけではない．機構全体としてどれだけ要求仕様を満足しているかを評価し，その評価に基づいて，複数の候補の中から適当な機構の構造を取捨選択することもまた行われる．

また，機構の構造の設計と並行して，その機構を構成する機構要素の基本的な設計変数を決定する必要がある．図1.1の例では，歯車の歯数比やベルト車の半径などが決まらなければ，それぞれの機構で実現される運動を評価することはできない．機構要素の配置に関しても，歯数比や半径などが決まらなければ歯車やベルト車の配置を決定していくことは困難である．このように「減速機構」を設計する上で，歯数比やベルト車の半径などは基本的な設計変数であるといえる．これに対し，歯車の軸径やベルトの厚さなどはこの段階ではそれほど重要な設計変数ではない．

こうした基本的な設計変数の値によっても，その機構によって実現される運動が決定される．そのため，機構の構造を決定する場合と同じように，どれだけ要求仕様を満足しているかを評価しながら適切な設計変数の値を決めていかなければならない．

第5段階：部品の詳細設計

第4段階（基本設計段階）で決定された機構の構造にしたがって，第5段階では部品単位のより詳細な設計が行われる．たとえば図1.1に示した例では，歯車のモジュールや転位量，さらに軽量化や強度を考慮した歯車の詳細形状などの検討が行われる．このように第5段階は，第1段階で述べたような設計対象の運動に関する要求仕様以外の仕様を検討する段階であるといえる．

また，第4段階では，いくつかの機構構造の候補から適当なものを選択することを目的として設計対象の評価が行われるのに対して，この段階では，主に部品単位での最適化を目的とした対象の評価が行われる．したがって，第4段階同様この段階においても設計対象はさまざまな方法で評価されるが，その評価方法は第4段階で行われるものと比較して，マスプロパティの計算，動力学的な解析，強度解析などのように，設計対象に関する詳細な情報を必要とするものが多い．

1.4 本研究の目的と実現手法

1.4.1 研究の目的

本研究は，平面機構を設計対象とし，前述した機構設計の第4段階（基本設計段階）において，次に示す二つの条件を満足する機構部品形状の設計支援を目的とする．

1. 機構の運動学的条件
2. 部品間の非干渉条件

そのために設計対象の幾何情報と機構の動作の両方を表現することが可能な対象表現法を提案する．さらにその表現法に基づいて設計対象である機構のモデルを構築し，上にあげた条件を満足するようにモデルを操作することができるシステムを開発する．

ここで1.2で述べた対象表現の分類に従えば，本研究で扱う設計対象を表現するためには次のような特徴が必要である．

- 基本設計段階の対象表現
- 組立品で可動部品を含む
- 幾何情報と機構の挙動を扱う

まず本研究で扱う設計対象の抽象度は，図1.1で示した基本設計段階に対応する．また設計対象が機構であることから，組立品でかつ部品間の相対運動がある対象を表現する必要がある．モデルの表現力に関しては，機構部品の幾何情報と機構の動作を表現できなければならない．この時，幾何情報と動作に関する属性値を連動して扱えなければならないことは既に述べた．つまり，幾何情報と動作を単に表現できるだけでは不十分であり，設計対象の動作に関する条件や，部品間の接続関係などを満足するように部品形状を決めることができなければならない．

1.4.2 本研究で提案する対象表現

本研究で提案する対象表現法では，幾何情報を，線分と円弧からなるセグメントの列として表現する．機構の動作に関する条件は，機構を構成する部品の位置や姿勢をあらわす設計変数をパラメータとする代数式で表現する．この機構の動作をあらわした代数式を以下では「制約式」と呼ぶ．したがって，本研究で扱う「機構」とは制約式表現できるものを指す．また，制約式で表現できれば他の条件と同様に扱うことができるので，付加的な条件としての設計者の意図も代数式表現できる範囲で扱える．そして，幾何情報と設計変数値の変更を相互に伝達するために，設計変数の値によって複数のセグ

メントを変形することができる変形手法を提案し，これにより制約式を満足する設計変数の値と幾何情報を連動させる．このような対象表現をとることにより得られる利点を，既に述べた従来の対象表現の問題点と対応させて表1.5にまとめる．

表 1.5: 本研究で提案する対象表現の特徴

評価項目	特徴
基本設計段階	未定義部分を含む対象の表現，付加的な条件（設計者の意図）の表現が可能
組立品	部品間の関係指定が容易，対話的な整合性管理
可動部品	表現対象の自由度大
統合性	幾何情報，機構の動作，付加的条件を扱う

本研究では，幾何情報の表現方法として線分と円弧の列を用いる．これにより，従来の対象表現法に対して対象モデルの操作性がよく，迅速なモデル生成が容易になるという利点がある．また従来の方法では扱うのが難しい，未定義部分を含む対象の幾何情報を扱える点でも本研究の方法は有利である．

たとえば，CSG 表現に基づいた幾何情報の表現法を用いることも可能だが，既に述べたように形状モデルを生成し操作することが簡単でない．これに対し本研究の方法では，設計対象のスケッチを描くように容易に幾何情報を扱うことができる．また，途中段階の形状として概略の部品形状や稜線が閉じていない部品形状の一部のような不完全な形状が生じるが，厳密な位相構造を必要とする表現法では，このような形状が生じると表現が破綻してしまい扱えない．この点でも単にセグメント列で形状を表現する利点がある．

次に，機構の動作を表現する方法としては代数式による制約表現をもちいる．これにより従来の表現法に対して (1) 部品間の対偶関係の表現が容易になる，(2) 未定義部分を含む対象を扱える，(3) 表現可能な機構の種類が多い，

(4) 付加的な条件を扱える，などの利点がある [Kikkawa 91c, Kikkawa 93].

たとえば，部品間の関係を表現するこれまでの方法として，変換行列（transformation matrix）による表現がある．しかし，変換行列を直接入力することは困難であり，制約式のほうが設計者にとってより扱いやすいといえる．また，制約式は一度に入力する必要はなく，制約式を追加していくことにより徐々に設計対象を詳細化していけばよい．つまり，制約が与えられていない部分として未定義部分を扱うことができる．そして本研究における対象表現法においては，制約式表現されるものを「機構」として設計の対象としている．したがって，制約式表現可能な範囲においてどのような機構でも扱うことができる．さらに，対偶関係以外の条件も制約式として表現できれば他の条件と同様に扱えるので，設計者が付加的な条件を与えることも可能である．たとえば，設計対象に対して「リンクの長さが等しい」といった条件を加えることができる．

以上で幾何情報の表現方法と，対偶関係や設計者の意図を表現することができる制約表現について述べた．しかし，既に述べたように，これらの表現が孤立していたのでは本研究で示す対象表現法の表現能力は機構を扱う表現法として十分でない．そこで幾何情報と制約を連動して扱えるようにするために，機構部品の変形手法として「ゴム変形モデル」を導入し，制約式を構成する設計変数と幾何情報とを関係付ける．

本研究では，「ゴム変形モデル」として次のような形状変形モデルを考える．まず，部品形状をあらわすセグメントが伸縮するゴム板の上に描かれているものとする．また，ゴム板上のある2点の座標が設計変数の値として与えられる．この時，2点間の距離が変化すると，図1.2に示したように2点間のゴムの伸縮によってゴム板上のセグメントも変形する．このような変形モデルを用いて，設計変数の値と幾何情報を連動させる．この変形モデルを「ゴム変形モデル」と呼ぶことにする．ゴム変形モデルを用いることにより，2点の座標という少数の設計変数値を使って，その2点の間にある複数のセグメントを変形できるところが利点である．ゴム変形モデルについては2.4.3で詳細に述べる．

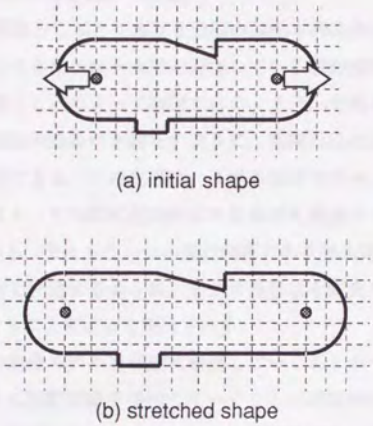


図 1.2: ゴム変形モデルによる部品形状の変形

1.5 本論文の構成

以上で述べた表現法の利点から結果として，モデル操作が容易なので，効率良く迅速に設計対象のモデルを構築していくことが可能となるといえる．またモデリングシステムに，制約を解きながら設計変数を決定する機能を付加することにより，幾何情報と制約の整合性を計算機によって管理できるようになる．その結果，設計者による形状変更や設計変数の値の変更に対して，計算機が制約を満足する状態のモデルを示し，さらにそれを用いて詳細化していくというように，システムと対話的にモデルを構築，操作することが可能になる．

ただし，本研究で提案する対象表現法では，概念設計から基本設計，詳細設計へと連続的に展開していくといったモデルの発展性は実現されない．これは，幾何情報と機構の動作を統一的に表現する方法ではなく，幾何表現と制約表現という二つの異なる表現法と両者の関係によってモデルの表現力を

確保していることからくる限界である．

本研究で設計問題としてとりあげる機構の運動学的条件は，まず，対偶関係や設計者によって与えられた条件を表現している設計変数のパラメータからなる方程式を解くことによって検討される．また，それらの方程式を解くことによって，運動学的条件を満たすように，機構部品の基本的な設計変数を決定することができる．その結果は，「ゴム変形モデル」にしたがって実体形状に反映させる．その際に部品形状を針金状に縮退させた「骨格形状」を媒介として用いる．またもう一つの設計問題である部品間の非干渉条件は，与えられた障害物を回避するように，まず骨格形状を変更し，その結果を実体形状に反映させることによって実現される．

こうした機構の対象モデルを迅速に構築していくことができ，また設計条件を満足するように設計対象を操作していくことが対話的に実行できるインタフェースを持つ機構のモデリングシステムを試作する．そして，実際にいくつかの機構のモデルを構築することによってその有効性を確認する．

以下では第2章において，本研究で提案する対象表現である「機構モデル」について述べる．まずこれまでに提案されている対象表現法を概観し，その問題点を明らかにする．次に機構モデルとして4つの対象表現を組み合わせて用いることについて説明し，その後で各表現間の関係について述べる．特に幾何情報と制約式を連動させる変形モデルとしてのゴム変形モデルについて詳説する．

次に第3章では，対偶関係や設計者による付加的な条件の表現方法として設計変数による制約式表現を用いることを述べ，その定義方法を説明する．そして，それらの制約式を解く方法として，条件式がすべての設計変数を決定するのに十分でない場合でも，適当な値を用いてすべての設計変数に対して一時的な値を決定する方法について述べる．また，動作にともなって対偶関係の条件式が変化する機構を扱うことができるように制約表現を拡張し，制約式として不等式を導入する．さらに不等式を含む場合の制約条件を解く手法について説明する．

第4章では，骨格形状によって実体形状を操作する方法について述べる．

そのためにまず骨格形状に対する操作を定義し，次に実体形状との対応関係を定義する．そして骨格形状での形状変更の結果を実体形状に反映させる手法を説明する．さらに，障害物の形状が与えられた時，干渉を回避するように骨格形状を変更する方法について述べ，さらに実体形状を骨格形状にあわせて変形することで部品の干渉回避形状が生成できることを示す．

第５章では，これらの手法をもとに，対話的なユーザインタフェースを備えた機構のモデリングシステムを試作し，実行例とともにその機能を紹介する．

最後に第６章で本研究の結論と展望を述べる．

第2章

機構モデルの表現法

2.1 機構モデル

本研究では，機構を実現するのに必要な情報を表現したもの全体を「機構モデル」と呼ぶ．具体的には機構モデルが表現すべき情報として，部品間の対偶関係，部品の配置（位置），部品形状，精度，材質，強度などいくつか考えられるが，本研究では，運動学的な条件と部品間の非干渉条件に着目した部品形状設計の支援が目的なので，以下のような情報を表現できなければならない．

1. 部品間の対偶関係
2. 部品の配置（位置）
3. 部品形状
4. 付加的条件

4番目に挙げてある付加的条件とは，設計者が設計対象に対して与える条件を指す．本研究では基本設計段階における設計対象を扱うが，この段階では設計対象を確定するだけの条件は与えられていないのが普通である．つまり，1～3番目の条件だけでは設計解をただ一つに決定することができない．実際には，設計者がさまざまな条件を追加していくことによって，設計対象

が詳細化されていく．そのため，機構モデルが表現すべき情報として，4番目に挙げた設計者による付加的条件を表現できなければならない．

また，上には示されていないが，上記の情報を相互に関係付けることが可能であることも必要である．1.2でも述べたように，上に示した情報をそれぞれ単独で表現するだけでは対象表現として不十分であり，各情報が整合性を保持していなければならないからである．さらに，これらの情報によって表現される機構モデルの操作性も重要である．モデル操作があまりにも繁雑になると，モデルの操作に多くの時間を費やすことになり，設計を支援するのではなく逆に設計を阻害してしまうことも考えられる．

2.2 従来の対象表現と問題点

本研究で提案する対象表現法について述べる前に，まずこれまでに提案されている対象表現の特徴と問題点をまとめる．ここでは，後で述べる本研究の手法との相違を明確にするために，次の三つの観点で分類する．

1. 組立品の表現法
2. 挙動の表現法
3. 形状の表現法

2.2.1 組立品の表現法

組立品の表現法は，部品間の位置関係を表現するために変換行列（transformation matrix）を利用しているものが多い．変換行列とは，各部品ごとに設定してある座標系間の座標変換を 4×4 の行列によって表現したものである．これによりある部品の位置・姿勢は，基準となる座標系からその部品へ至る変換行列によって求めることができる．たとえば [Lee 85b] では，各部品の面間の接触関係から変換行列による連立方程式を導出し，Newton-Raphson法や最小二乗法を用いて各部品の位置・姿勢を求める手法について述べている．

これに対し，直接変換行列を指定することは困難であるということから，[Lee 85a] では仮想リンク（virtual link）をもちいた対象表現法を提案している．仮想リンクを用いることで，設計対象の構成部品間の接触関係を階層的に表現できるとしている．この方法でも，部品間の位置関係を最終的には変換行列に展開するが，仮想リンクで表された構造から変換行列が導出できるため，変換行列を直接入力するのに対して扱いやすいといえる．

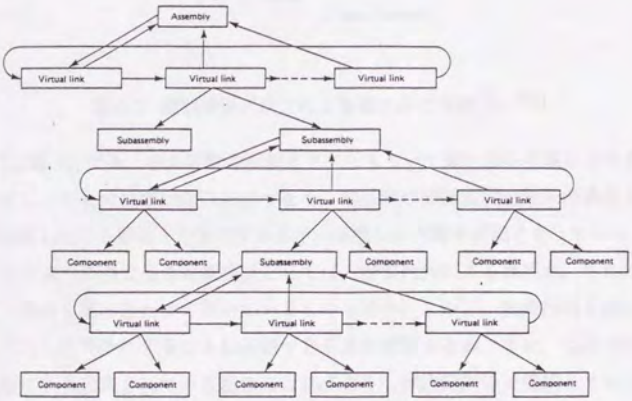


図 2.1: 仮想リンクを用いた組立品の表現 [Lee 85a]

また [Ko 87] では，部品どうしの面の接触関係を 4 種類に分類したうえで，部品間の接触関係を部品をノードとし，部品間のアークに接触関係の種類を付け加えたグラフ構造（mating graph）で表現している．この方法においても，部品間の位置関係は接触関係のグラフから変換行列に展開される．同様

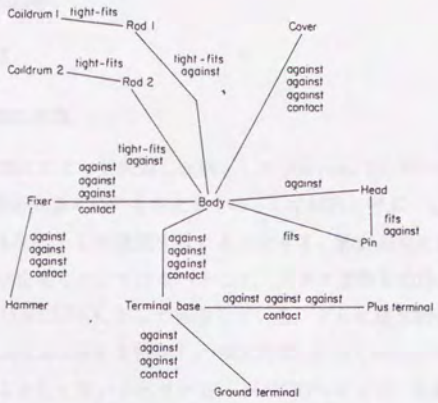


図 2.2: 接触関係グラフによる組立品の表現 [Ko 87]

に [北嶋 81] でも，部品面間の接続グラフをもちいて組立品を表現している．ただし，ここでは接続関係だけでなく，部品間の関係が固定関係の場合と運動拘束している場合とで区別するなど，実際には複雑な表現となっている．

このように組立品の対象表現としては，変換行列による表現か，さらにグラフ表現を組み合わせて用いているものが多い．しかし，変換行列を設計者が入力したり操作することは困難である点に問題がある．また，これらの表現法により表現されている組立品は部品どうしが固定関係で接続しているものが多く，[北嶋 81] のように機構部品を対象としている場合は，表現可能な機構が制限されている点に問題がある．

2.2.2 挙動の表現法

ここでは，挙動を表現する方法として，

- 代数式による表現
- 配位空間による表現

- 定性推論の応用

について検討する.

代数式による挙動の表現

設計対象を代数式によって表現した例として [Serrano 88] がある. このような代数式は, 設計対象に条件を与えるものとして制約と呼ぶ. したがって, 設計対象の挙動も制約として表現することができる. 設計過程において制約を管理することは自明なことではないとして, グラフ理論を応用した制約管理手法を提案し MATHPAK として実現している. さらに概念設計支援システムとして Concept Modeler を開発し, MATHPAK は Concept Modeler の制約管理システムとして用いられている. MATHPAK では, 制約条件を設計変数の代数式として表現する. 図 2.3に MATHPAK 上に表現された制約条件の例を示す.

図 2.3: MATHPAK における制約の代数表現 [Serrano 88]

MATHPAKにおける制約条件の評価方法としては，これらの制約条件に含まれるパラメータ（設計変数）を入力パラメータと出力パラメータに分類し，入力パラメータの値から出力パラメータの値を求めることが可能である．また定義された制約条件式に基づいて，パラメータ間の関係をグラフ表現し，その関係グラフを用いて冗長な制約条件や干渉する条件式の抽出を行うことができる．さらに制約条件の定性的な評価方法として，入力パラメータの増減が，出力パラメータの増減にどのように影響するかを求めることができるとしている．

こうした制約条件に基づいて設計対象をを定義する方法として，Concept Modelerでは機械設計でよく用いられる要素をあらかじめ用意し，それらを組み立てることによって設計対象を構成していく．またそのような機械要素は簡単な図形で表示され，設計対象はそれらの図形の集合体として表現される．Concept Modelerでは機械要素を表示する図形を「アイコン」と呼び，各アイコンごとにその機械要素を設計するのに必要な計算式を用意している．図2.4にConcept Modelerで用意されている機械要素とそのアイコンの例を示す．設計者はこれらの機械要素を抽象化したアイコンを使って，対話的に対象モデルを構築できる．

2.2.3 配位空間による挙動の表現

配位空間（Configuration Space）とは，物体の位置・姿勢をあらわすパラメータを軸とする多次元空間である．この配位空間上で物体どうしが干渉する領域を求めることにより，障害物回避経路の生成などにも応用される．[Joskowicz 88]では設計対象に対する挙動の仕様を，このような配位空間によって表現している．

たとえば，図2.5(a)の機構は固定軸O_1のまわりに回転する部品Aと，固定された軸O_2にそって平行移動する部品Bで構成されている．この機構に対して，部品Bの位置によって部品Aの回転が，回転角$=0$と$\pi/2$の位置でロックされるように運動に関する要求仕様を変更したと仮定する．その要求仕様を満足するためには，部品Aの形状は図2.5(b)のように変更されなけれ

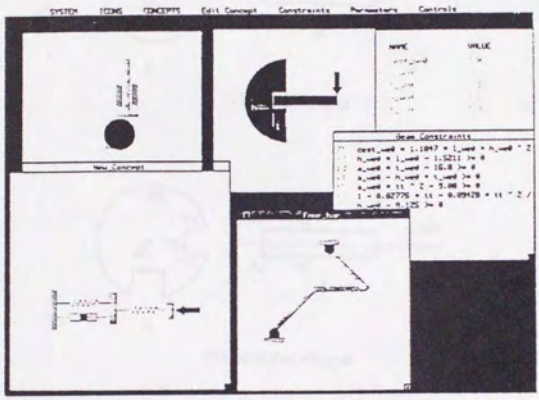
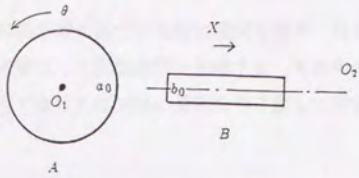
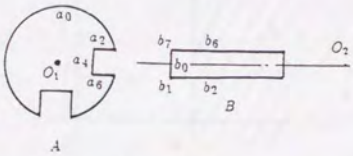


図 2.4: Concept Modeler で用意されている機械要素 [Serrano 88]

ばならない．このような機構の運動に関する要求仕様の変更は，配位空間上



(a) Initial Shapes



(b) Modified Shapes

図 2.5: 機構の運動に関する仕様の変更と形状変更 [Joskowicz 88]

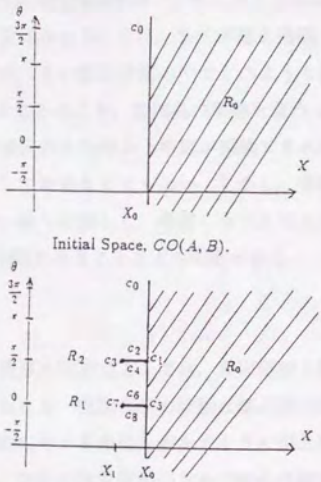
では図 2.6の (a) から (b) の変化として表される．そこで [Joskowicz 88] では，設計対象の運動に関する条件を図 2.6のように配位空間上で表すことができれば，逆に配位空間の変化から以下に示すような手順にしたがって，対応する部品の形状を変更できるとしている．ただし，扱う機構部品は固定軸のまわりに回転するものと，固定軸にそって平行移動するものに限定している．

1. 初期状態の配位空間を求め，挙動の条件を満たすように配位空間を変更する．
2. 初期状態の配位空間と変更後の配位空間を比較し，消去された部分と，追加された部分を明らかにする．

3. 配位空間において消去された部分があれば，対応する図形を消去する．

4. 配位空間上で追加された部分がある場合は，その配位空間に対応するような対偶を選択し，選ばれた対偶の種類に応じて形状を変更する．

5. 変更された部品形状に基づいて配位空間を求め，はじめに挙動の条件を満たすように変更した配位空間と比較する．その条件を満足していれば終了し，そうでなければ対偶の選択をやり直して形状変更を繰り返す．



Initial Space, $CO(A, B)$.

Space After Modification, $R(A, B)$.

図 2.6: 配位空間による挙動の表現 [Joskowicz 88]

機構設計においては，配位空間中の自由空間（部品間の非干渉領域）として，機構部品の可動範囲を表すことができる．また，自由空間と干渉領域の境界は，部品が接触している状態に対応する．したがって，配位空間上の自

由空間の形状を変更することは，機構部品の可動範囲や，部品間の接触状態を変更することに対応する．つまり，自由空間の形状によって，機構の動作に関する条件を表現することができる．さらに，配位空間における自由空間と干渉領域の境界の形状は，その境界に対応して接触している機構部品の形状に依存しているので，境界の形状によって逆に機構部品の形状を規定することができるといえる．

定性推論による挙動の表現

また，部品の挙動を直接表現せずに，部品形状や配置などの幾何的な条件から挙動を導出する試みに定性推論が用いられている [Faltings 88, Faltings 90, Forbus 91]．たとえば [Faltings 90] では，まず挙動を推論しようとしている部品間の配位空間を求め，その配位空間の中でどのような接触関係の変化が起こり得るかを推論することにより，各部品の挙動を求める．

このように各部品の幾何的な条件から挙動が推論できれば，部品の動作に関する条件を直接表現しておかなくてもよい．しかし，実際には部品間の配位空間を求めることが一般には難しく，考慮しなければならない部品点数が増えると極端に推論が困難になることなどの問題がある．

挙動の表現の問題点

これまでの挙動の表現法の問題点としては，幾何情報と関係付けて扱うことができない点が挙げられる．機構部品の挙動は部品間の接続関係と形状によって決まるので，挙動に対する条件を満たすように部品形状も変更しなければならない．しかし，従来の対象表現のなかで特に代数式表現によるものは，形状との関係付けがほとんど考慮されておらず，非常に単純な形状を扱うのみである．

これに対して配位空間を用いる方法は，動作に関する条件と形状をともに配位空間上で扱うことができる点で有利である．しかし，配位空間を算出すること自体が難しい問題であることと，配位空間中の障害物領域の形状変化を実際の部品形状へ変換することが困難であることなどから，やはり挙動の

表現と形状との関係付けが問題であると言える.

2.2.4 形状の表現法

形状の表現法としては，境界表現やCSG表現が既に広く応用されている．しかし，これらの表現法は，形状を定義したり変更するために操作が複雑であるという問題点をもつ．そこで以下では，設計者にとってさらに扱いやすくすることを目的とした表現法について検討する．

形状要素間の依存関係による形状表現

形状要素間の依存関係による形状の表現法は，線分や円弧などの形状要素間の幾何学的な関係を形状決定のための制約条件とする方法である．たとえば，「平行」「直交」「接線」などの関係を形状要素間の依存関係として与えておく．そのうえで，平行直線間の距離や互いに交わる線分のなす角度などを決めていくことにより形状を決定していくことができる [Ando 89, Nelson 85].

このような表現法をとることにより，より容易に形状を表現することができる．たとえば，図2.7ではアルファベットのAの文字が描かれているが，始めに入力する図形は平行直線などでなくてもよく，大雑把にAの文字を入力すればよい．その後で，図に列挙されてあるような依存関係を入力することによって，正確な平行直線や水平直線などを得ることができる．つまり，概略の形状を入力しておいてから，形状要素間の依存関係を用いて整形すればよいのである．

形状特徴による形状表現

「溝」や「穴」などといった良く用いられる部分的な形状をまとめて扱うことができれば，形状を表現していく際に有効である．そのような部分的な形状を「形状特徴」とし，形状特徴を用いて対象の形状を表現するさまざま研究が行われている．たとえば [Mäntylä 90] では，形状特徴の組合せによって図2.8のような形状を表現している．このように形状特徴として適当な部分形状を用意しておけば，それらを組み合わせて用いることで複雑な形状を比

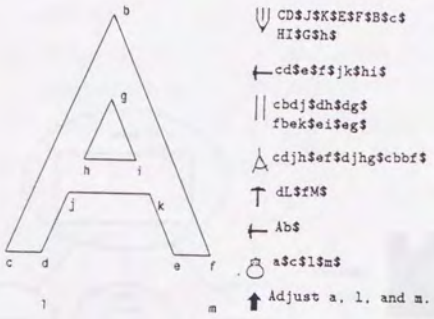


図 2.7: 形状要素間の依存関係による形状表現 [Nelson 85]

較的容易に表現することが可能になる.

形状表現の問題点

上記のように入力や操作の複雑さの点に関していえば，形状要素間の依存関係や形状特徴を用いることによってある程度問題を回避することができる. しかし，機構を表現する場合には，組立品の表現が難しい，挙動との関係を扱うことが困難である，などの問題がある. また，基本設計段階における対象を表現するためには，部分的に定義されていないような不完全な状態を表現する必要がある. この点でも従来の方法では対応できない問題がある.

2.2.5 従来の対象表現のまとめ

以上で述べたように，従来からの対象表現において，機構部品間の対偶関係や部品形状をそれぞれ単独で表現するのに適した表現は存在する. しかし，それらを統合して表現できる表現法は今のところない. また，両者の関係を表現可能なものとして配位空間による表現方法が提案されているが，この表現方法を利用して機構を表現することは現段階では困難である. したがって，

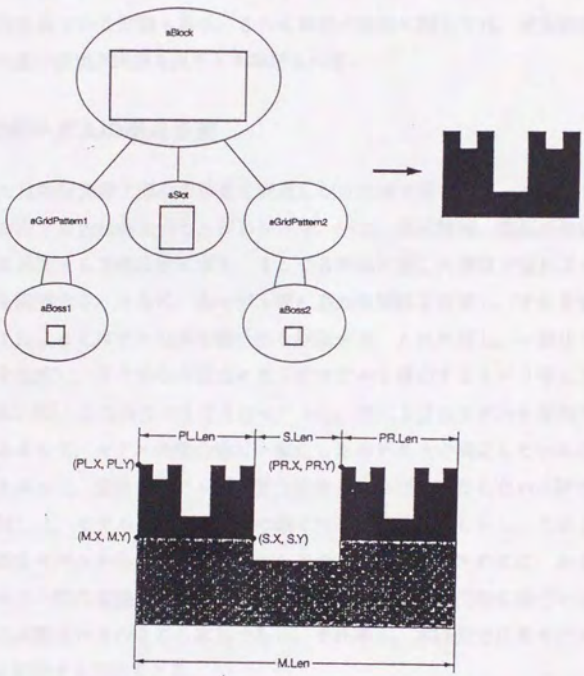


図 2.8: 形状特徴による形状表現 [Mäntylä 90]

従来の対象表現では部品の動作の表現と部品形状とを関係づけて扱うことができない点に大きな問題がある.

対象表現法の表現力以外の問題点としては，組立品の表現に関しては，部品間の関係指定が煩雑である点が挙げられる．また挙動の表現に関しては，対象が限定されている問題もある．さらに形状の表現に関しては，未定義部分を含む対象の表現が困難な点などが挙げられる.

2.3 機構モデルの構成要素

本研究では幾何情報と部品の挙動を表現しなければならないが，これらを統一的に表現する方法がないことがわかった．一方，幾何情報，部品の挙動とも個別に表現する方法は既にある．そこで各情報に適した表現方法によってモデルを構築する．さらに，各モデル間に適当な関係を定義し，それを管理することによってモデル全体の整合性を保証する．これに対し，一般化したモデルを生成し，そこから各視点にそってモデルを導出するという考え方もある [桐山 92]．この場合の各視点のモデルは，常に上位のモデルを参照することになるので，モデルの整合性は一般化したモデル上で満足していればよい．したがって，複数のモデル間で整合性をとらなければならない本研究の方法に対して，モデルの整合性管理の面では有利である．しかし，そのような一般的なモデルから各視点に対応したモデルを導出するためには，かなり高度なモデル間の変換規則が必要であり，そのような変換規則に基づいた推論手法の実現はいまのところ容易でない．それゆえ，本研究では各モデル間の関係を管理する方法をとる.

具体的には，表 2.1のように 4 つのモデルにわけて機構を表現する．「構成要素」は，各モデルで表現される情報の内容を表すような名前をつけてある．「モデル表現法」の項目は，各モデルを具体的に計算機内で表現する方法を示している．「ユーザインタフェース」の項目には，各モデルの表示方法と操作方法が示してある．すなわち，各モデルの持つ情報をユーザに対して示す方法と，モデルを操作する際のデータの入力方法が示されている．これら

のモデルを用いて，機構設計における，動作実現のための幾何学的な条件と部品間の非干渉条件を扱っていく．幾何学的な条件は，主として制約モデルによって表現される．また非干渉条件は主として骨格形状モデル上で扱われる．

表 2.1: 機構モデルの構成要素

構成要素	モデル表現法	ユーザインタフェース
概略形状モデル	セグメントを基本要素とする幾何情報	部品形状のスケッチ
骨格形状モデル	グラフ構造＋幾何情報	骨格形状のスケッチ
制約モデル	パラメータ間の関係式	アイコン，モジュール名，(関係式)
属性モデル	属性名，属性値，属性値のタイプの組	パラメータ名，パラメータ値，パラメータタイプ

設計者にとって理解しやすい方法で情報を提示し，また設計者にとって扱いやすい情報によってモデルを定義できるようなインタフェースを提供することは本質的である．計算機での処理に適した情報と，設計者にとって扱いやすい情報は必ずしも一致しないため，適当なインタフェースによって両者の隔たりを埋めなければ有効な支援が行えない．本研究では，設計の初期段階から支援することを目的としているので，特に，対話的な方法や非数値的なインタフェースが重要である．

部品形状の入力インタフェースを例にとると，初期段階ゆえに形状の寸法が未確定状態だとしても，それを計算機で扱うためには，その形状の大きさや位置をあらわす数値が必要となる．したがって，計算機システムの側から見れば，どのような方法でユーザ・インタフェースを実現しても，最終的に数値が得られるという点で同じである．しかし，ユーザの側からすると，数

値を直接入力しなければならないようなインタフェースでモデルを構成していくことは，不可能ではないだろうが必要以上の労力を必要とする．これに対し，ユーザにとって扱いが容易なのは作図など図形を利用した方法であろう．このようなことから，表2.1の概略形状モデルに示したように，本研究ではこの場合のユーザインタフェースとして，作図（スケッチ）により間接的に数値を指定する方法を採用している [吉川 90]．

また，属性モデルでの変更は，最終的に概略形状モデルに反映されなければならないし，逆に形状レベルでの変更結果は，モデル全体の整合性がとれるように属性モデルまでその結果を伝播しなければならない．そのため，各モデル間には一定の対応関係を定義し，その関係にしたがって各モデルの情報が他のモデルに伝達されるようにする．図2.9にモデル間の関係を示す．以下では各モデルについて説明する．

2.3.1 属性モデル

属性モデルとは，「対象とする設計において必要となる設計属性を抽出し，その属性集合を設計対象表現（モデル）としたもの」である [伊藤 90]．属性モデルは（属性名，属性値）の組により対象を表現し，属性値がスカラー値や文字列などの非構造的なデータであることを特徴としている [鈴木 89a]．本研究では，さらにその属性値がどのように決定されたかを示すものとして「属性値のタイプ」を加え，（属性名，属性値，属性値のタイプ）を組として設計対象を表現する．図2.10に歯車の属性表現を示す．歯車の形状や位置，さらに運動に関する回転角などがパラメータ化されている．

こうした属性モデルを操作する手段としては，パラメータ名によって適当なパラメータを選択し，パラメータの値やそのタイプを直接指定する方法がある．しかし，本研究では特に寸法を明示的に扱わない段階での形状操作を対象としているため，実際にはパラメータ値を直接与えることは難しい．また，モデルが複雑になるにつれて，パラメータ名とその値から設計対象を把握することは困難になる．そこで，属性モデルは設計者に対して可能な限り隠蔽し，原則として個々のパラメータ値やパラメータタイプはインタフェー

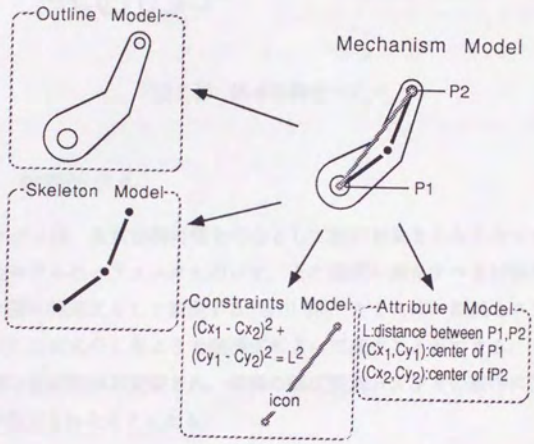
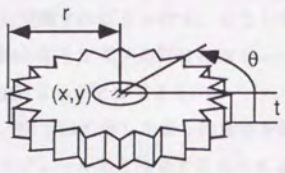


図 2.9: 機構モデルと構成要素の対応関係

スとして用いない．そのかわりに，制約モデルと関係付けることにより，制約モデルから間接的に指定する方法をとる．



Attributes
t: face width
r: pitch radius
z: number of teeth
(x,y): center of gear
θ: rotation angle

図 2.10: 歯車の属性モデル

2.3.2 制約モデル

制約モデルは，主に機構対偶を中心として設計対象をとらえたモデルである．属性モデルのパラメータを用いて，その機構が満たすべき対偶関係をパラメータ間の関係式として表現する [吉川 89]．たとえば，単純なスライダは (2.1) ～ (2.3) 式に示したような関係式によって表すことができる．これにより部品間の接続関係が定義され，機構の構成要素およびその動作に関する制約条件が指定されたことになる．

$$(x_p - x_1)(y_2 - y_1) = (y_p - y_1)(x_2 - x_1) \tag{2.1}$$

$$x_1 \leq x_p \leq x_2 \tag{2.2}$$

$$y_1 \leq y_p \leq y_2 \tag{2.3}$$

こうした制約条件を定義する方法として，表 2.1のインタフェースの項目にあるように，モジュール名を指定し，ある程度まとめて条件式を定義する方法をとる．モジュール名とは，モジュール化された関係式につけられた名前（ラベル）である．パラメータ間の関係式をひとつひとつ設計者が入力するのでは，モデルを構成していく作業が繁雑になる．また，対偶を定義する際には，その対偶を表現するのに必要なすべての条件式を定義しなければ意味

がない．そのため，良く用いられる対偶または単純な機構を表現する条件式は，あらかじめまとめてモジュール化しておく．ある対偶を定義するのに必要な制約条件が場合によってさまざまに変化することはなく，数個のきまった制約条件を定義すればよいので，こうしたモジュール化は可能である．その結果，制約条件を定義する際にはモジュール単位でパラメータ間の関係式をまとめて生成することができる．

たとえば，図2.11に示したように歯車を構成要素として用いる場合，「歯車」というモジュール名を指定することによって，(2.4)～(2.7)式を生成する．ただし，このような方法をとるとまったく新しい機構対偶を開発する場合には対応できない．しかしそれ以外の場合では，すでに知られている機構対偶を組み合わせることによって機構を構成していくと考えられ，このような方法をとっても十分有効であると考えられる．

$$r_1\theta_1 + r_2\theta_2 = 0 \tag{2.4}$$

$$Ra = r_1/r_2 \tag{2.5}$$

$$D = r_1 + r_2 \tag{2.6}$$

$$D^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2 \tag{2.7}$$

r_i：$Gear_i$ のピッチ円半径，θ_i：$Gear_i$ の回転角，(x_i, y_i)：$Gear_i$ の中心座標，Ra：ギア比，D：軸間距離

制約モデルにおいては，制約条件を付加していくことにより設計対象を詳細化していくことができる．上記の(2.4)～(2.7)式は，二つの歯車が満たすべき位置関係や回転角の関係といった，歯車機構の基本的な条件を表現しているにすぎない．さらに設計を進め歯車機構を詳細化していくためには，この機構に対する制約条件を付加していけばよい．たとえば，図2.11のように歯車の回転軸を水平方向にならべるためにはさらに(2.8)式を制約条件として定義する必要がある．また，仮にギア比に対する要求値が与えられているならば，(2.9)式を制約条件として付加することによって，ギア比を指定することができる．このようにして，設計要求をあらたな制約条件として定義す

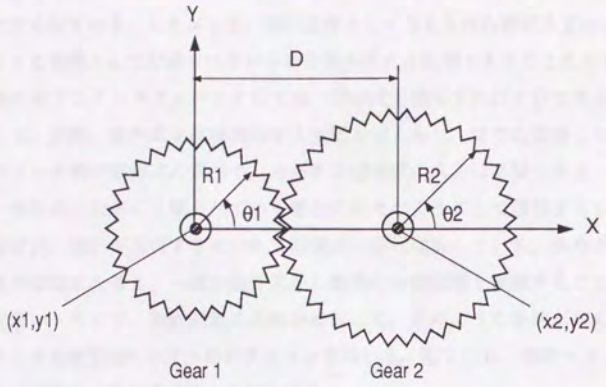


図 2.11: 歯車機構

ることによって，次第に設計対象が詳細化されていく．

$$y_1 = y_2 \tag{2.8}$$

$$Ra = 2/3 \tag{2.9}$$

またもう一つのインタフェースとしてあげてあるアイコンとは，制約の各モジュールごとに定義する図形表示のことである．アイコンは，線分，円，正方形の組合せで表示する[1]．制約モデルにおけるアイコンは，次のような2つの役割をもつ．

- 対偶関係の視覚化
- パラメータ値操作のハンドル

[1]ここでいうアイコンは，一般に「機構のスケルトン図」と呼ばれるものに近い．しかし，つぎに示すモデルを「骨格形状モデル」とするため，これを「スケルトン図」とすると紛らわしくなる．そこで本研究では視覚化という機能に着目して「アイコン」と呼ぶことにする．

対偶関係の視覚化　上で述べたように，機構の対偶関係はパラメータ間の関係式で表されている．したがって，制約条件として与えられた数式を見れば，どのような機構として定義されているかを基本的には理解できることになり，設計者に対するインタフェースとしては，関係式を提示すれば十分であるともいえる．実際，条件式を直接操作する状況も考えられ，すでに定義してあるパラメータ間の関係式の表示や，あらたな関係式の入力は必要である．しかし，条件式の数がごく限られている場合にはそのようにして理解することも可能だが，設計が進行するにつれて関係式の数は増加していき，条件がある程度の個数になると，一連の条件式から機構の対偶関係を理解することは容易でない．そこで，設計対象の各部分に対して，どのような条件が与えられているかを視覚的に示すためにアイコンを用いる．図 2.12に，制約モジュールごとに定義してあるアイコンの例を示す．

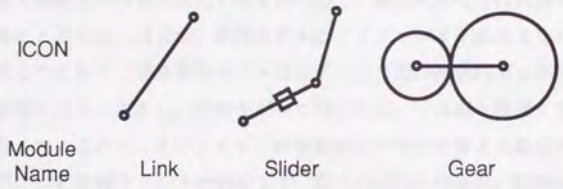


図 2.12: 制約モジュールのアイコンの例

パラメータ値操作のハンドル　2.3.1でも述べたように，設計初期段階においてはパラメータ値として直接数値を与えることは難しい．そこでアイコンと属性モデルのパラメータ表現を対応づけることによって，図形情報を利用したパラメータ値の指定方法を提案する．これにより，アイコンを操作することによって対話的なパラメータ値の決定が可能となる．

2.3.3 骨格形状モデル

骨格形状モデルは，部品形状の全体的な特徴を表すモデルであり，ループを含まないグラフによって表現される．骨格形状という名称が示すように，部品形状そのものを表すモデルではなく，全体的な凹凸や長手方向の軸などといった形状の特徴を部品形状の骨組として表現するモデルである [図法研究 90, 図法研究 91]．したがって，グラフ構造によってあらわされる骨格形状のトポロジだけでは不十分であり，部品形状に対応したアークの長さやノードの位置という幾何情報も必要である．骨格形状モデルの表示方法としては，グラフのノードには点（小円）を，アークには線分をそれぞれ対応づけた図形表示を用いる．

骨格形状モデルは，部品形状を抽象化したモデルといえる．また，制約モデルのアイコンも対偶部分の形状を抽象化した表現とみなせる．いずれも部品形状を単純な形状に抽象化したものなので，場合によっては両者の表示が重なることがある．通常は，制約モデルのアイコンが 2 部品にまたがって表示されるのに対し，骨格形状モデルは必ず一つの部品形状内部に存在するという相違がある．ただし，制約モデルに関しては，「単純な機構」を範囲として含んでいるので，そのアイコンが骨格形状モデルと重なる場合が生じ得る．図 2.13に単純なリンクの例を示す．図 2.13の (a) の場合，制約モデルのアイコンと骨格形状モデルの表示は一致するが，(b) の場合のように形状が変化すると骨格形状もそれに対応して変化するが，機構としての機能は変化しないのでアイコンは変化しない．その結果，制約モデルのアイコンと骨格形状は異なってくる．

こうした骨格形状モデルを導入する理由として，つぎに示す 3 点があげられる．

1. 大域的な形状操作のハンドル
2. 属性表現と形状表現の媒介
3. 機構の構造の視覚化

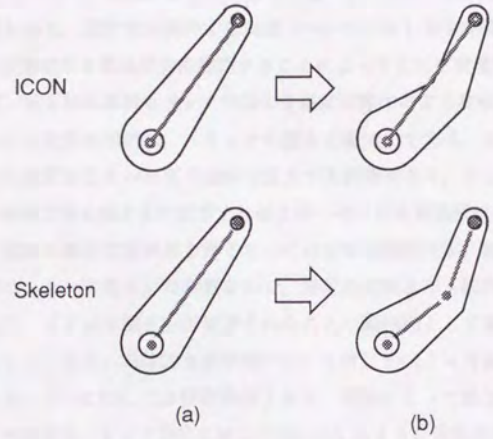


図 2.13: 制約モデルのアイコンと骨格形状

以下，それぞれについて説明する．

大域的な形状操作のハンドル　機構を設計していく過程で必要な形状操作として，局所的な変形操作と，大域的な変形操作が考えられる．ここで，局所的な形状操作を「個々の幾何要素（形状を表現している線分や円弧などのセグメント）に対して行われる操作」，大域的な形状操作を「一度の操作が複数の幾何要素に及ぶもの」とする．また，骨格形状による変形操作を略して「骨格変形操作」と呼ぶことにする．局所的操作には，幾何要素に対する一般的な操作（生成，消去，移動，変形）が含まれる．また，大域的操作の例としては，図形の移動，拡大縮小，そして骨格変形操作などがあげられる．

本研究で骨格変形操作をとりあげる理由は，機構の動作を設計していくという観点からは，幾何要素を直接操作することが目的とならないこと．また，結果としてある軸方向に変形することが多く，そのような変形が，図形の移動や拡大縮小のように，対象とするすべての幾何要素に対して一つの変換規

則を適用したのでは，実現できないからである．そのため，部品形状に対して骨格形状を与え，設計者が操作する対象（＝ハンドル）は骨格形状とし，骨格形状の変形結果を部品形状に展開することによってこれを実現する [吉川 91b].

たとえば，図 2.14は単純なリンク形状の骨格変形操作による変形例である．この例における変形の目的は，「リンクの長さの変更」である．この場合，よほど極端な変更をしないかぎり全体を拡大する必要はなく，リンクの幅はそのままで長さ方向を軸として変形すればよい．そうした部品形状の変更は，実体として意味のある変更結果を得るためには必要な操作だが，設計者の意図した操作は「リンク長さ」の変更なので，形状の変更までも設計者に任せるのではなく，「リンク長さ」が変更されたことの副作用として実現されることが望ましい．また，具体的な変形操作としては，L1,L2 には拡大操作が適用されるが，C1 に対しては移動操作となり，要素によって処理内容が違う．こうした変形を，リンク形状に対して図に示したような骨格形状を与え，骨格形状と部品形状を表している形状要素間にある一定の対応関係を付けることにより実現する．

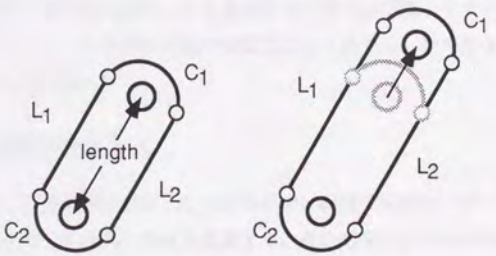


図 2.14: 骨格変形操作の例

属性表現と形状表現の媒介　属性表現と形状表現をつなぐためには，その中間に情報を媒介するものが必要である．骨格形状モデルは，その媒介となり得る．属性表現におけるパラメータによって，形状を完全に記述することは

困難であり，また仮にできたとしても，属性モデルの目的からもはずれる．したがって，属性レベルでの決定を形状表現に反映させることはそのままでは難しい．

これに対し，制約モデルのアイコンと骨格形状モデルが重なる場合のあることからもわかるように，制約モデルでの形状の扱いと骨格形状モデルで扱う形状のレベルは比較的近い．したがって，制約モデルのアイコンと骨格形状モデルの親和性は高く，制約モデルと骨格形状モデルを接続することは可能である．一方，骨格形状モデルによって部品形状を操作することも可能なので，結果として，骨格形状モデルを媒介とすることにより，属性表現を形状表現に展開することができると考えられる．

機構の構造の視覚化　3次元的に配置された機構部品の構成を認識する場合や，また本研究のように平面機構であっても上下に部品が重なっている場合などは，骨格形状のように部品形状の骨組だけを表示することによって，機構の構造の認識をより容易にすることが期待できる [松尾 91a],[松尾 91b]．また，設計対象の動作のみに着目し部品間の干渉は特には問題としないような場合，複雑な実際の部品形状よりも骨格形状のほうが認識しやすいという利点がある．このように機構の構造の視覚化という点でも，骨格形状による表現が有利となる局面がある．

2.3.4　概略形状モデル

本研究では平面機構を対象としているので，線分と円弧をセグメントとする2次元形状により部品の形状を表現する．その結果，板カムの外形などの複雑な曲線形状を表現するのは困難になる．しかし，カム曲線以外の多くの機構部品の形状は線分と円弧の組合せで表現可能であり，また複雑な曲線形状の正確な表現が困難であっても，設計初期段階では線分と円弧による近似形状で十分な場合が多いと考えられる．

部品形状の定義方法は，設計の早い段階での定義を容易にするために，その寸法値などを明示的に扱うことのない方法が必要である．さらにセグメン

ト単位で形状を扱えることも重要である．設計の初期段階で扱われる形状はおおよその形状でよく，寸法値は必ずしも要求されない．この段階で形状を定義するために，寸法値を決定することは困難であると思われる．そのためこのモデルで表現する形状を「概略形状」と呼ぶ．詳細な形状が未決定の段階でも，機構の構成を決定したり，部品どうしが干渉しないような配置や概略形状を決めることは可能である．またセグメント単位での形状操作が必要となる理由は，部品形状を一度に決定することが困難なので，形状変更が容易でなければならないからである．部品形状は，すべての部品の形状を一度に決定することが難しいだけでなく，ひとつひとつの部品の形状についても，初期形状は簡単なものであり次第に複雑な形状に変形されていくという意味で，一度に決定することは難しい．そこで表2.1のユーザインタフェースの項目にあるように，入力方法として作図に自由度のあるスケッチを用いる．ただしここでは，「寸法値を扱わない」という意味で「スケッチ」と呼ぶこととし，必ずしも手書き入力といったフリーハンドの入力を意味しない．

入力方法としてスケッチを用いる結果，部品の形状表現として不完全な状態も生じ得る．その意味でも，このモデルで扱う形状は概略形状である．たとえば，部品の外形はセグメントの閉ループとして表現されなければ，実体の表現として完全とはいえないが，設計者の意図としては接続しているセグメントをスケッチしても，それぞれのセグメントの端点が完全に一致しているとは限らない．また意図的にある一部分の形状しか考慮しない場合も考えられる．そのため概略形状モデルでは，立体モデルの winged-edge 構造や half-edge 構造などに対応するような，幾何要素間の厳密な構造をとることができない．したがって，線分を ls，円弧を cs とすると，概略形状モデル Rf は，

$$\mathrm{Rf} = \{\mathrm{ls}, \mathrm{cs}\} \tag{2.10}$$

と表される．

2.4 構成要素間の対応関係

機構モデルの構成要素となる各モデルを，それぞれ単独のモデルとして扱ったのでは機構モデル全体としての整合性が維持できない．そこで各構成要素間に対応関係をつけ，各モデルでの操作を他のモデルへ伝播させる必要がある．以下では，そのような各モデル間の対応関係について述べる．

2.4.1 属性モデルと制約モデルの対応関係

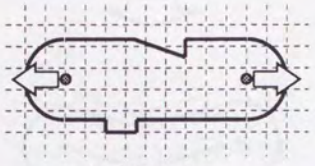
制約式は，属性モデルにおける属性値をパラメータとした数式によって表現されるので，属性モデルと制約モデル間の対応づけを行うための特別なしくみは必要ない．設計者によって値が指定されている属性については，制約式を解く時に，対応するパラメータをすべて数値に置き換えることで制約モデルに反映される．また，制約式を解くことによって求められたパラメータの値は，そのまま対応する属性の値として用いられる．このようにして，属性モデルでの変更を制約モデルへ伝達し，また制約モデルの評価結果を属性モデルに反映させる．

2.4.2 制約モデルと骨格形状モデルの対応関係

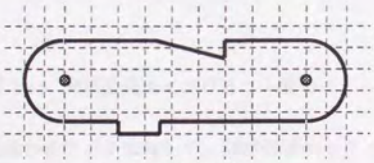
骨格形状モデルのもつ情報は，骨格のノード位置とアークの長さである．したがって，制約式中にノードの座標とアークの長さに対応する変数が含まれる場合，骨格のノード座標とアークの長さが常に対応する変数の値を指すように対応づける．このようにして，制約モデルの評価結果を骨格形状モデルに反映させる．また制約式を生成する際に，骨格形状モデルのノード位置をパラメータとして含む場合がある．このような場合，骨格形状モデルを参照しながら制約式を生成することで，そのパラメータの初期値を設定することができる．

2.4.3 骨格形状モデルと概略形状モデルの対応関係

骨格形状モデルと概略形状モデルの対応関係は，1.4.2で述べたようにゴム変形モデルにしたがう．概略形状モデルはゴム板上に描かれていると考え，骨格形状モデルはゴム板を伸縮させるとき力を加える２点を結ぶものであるとみなすことができる．図2.15では，両端に示してある２点の間に骨格があると考えることができる．つまり，骨格はゴム板を変形させるアクチュエータの役割を果たしているといえる．



(a) initial shape



(b) stretched shape

図 2.15: ゴム変形モデルによる部品形状の変形（再掲）

また，骨格どうしの接続部分に関しても同様に，骨格のなす角度が増加する方向の変形に対しては接続部分のゴムが伸び，逆に骨格のなす角度が減少する方向の変形に対しては接続部分のゴムが縮む．ただし，図2.16に示すように，接続部分の変形に関しては骨格の接続部を中心とする扇形に含まれる領域のみ変形し，それ以外の部分には影響がないとする．すなわち，骨格に対して直角方向には変形しないことを前提としている．またゴム板は無限に

伸縮することができると考え，縮む場合にゴム板がたるむ可能性などは考慮しない．

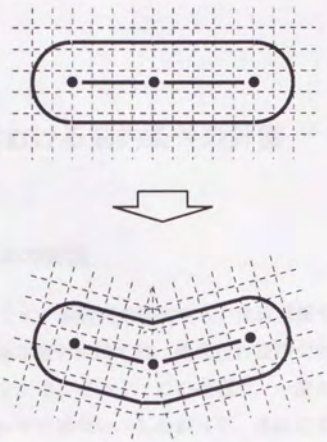


図 2.16: 骨格接続部におけるゴム変形モデル

実際には骨格形状の各アークに対して，概略形状のセグメントを対応づけることにより，こうしたゴム変形モデルによる形状変形を実現する．ただし，概略形状の変形をすべてゴム変形モデルで扱うわけではない．換言すれば，形状操作法をゴム変形に限定しない．たとえば，概略形状の部分的な変更はゴム板上の概略形状を描き直すことに相当し，この場合はゴム変形モデルとは無関係に変形することができる．ゴム変形モデルによる形状変形は，制約モデルを評価した結果を形状に反映させるために利用される．

第3章

機構モデルにおける制約条件の評価

3.1 制約モデルの表現

1.4.2で述べたように，本研究で提案する対象表現法では基本設計段階における対象を扱う．基本設計段階では，概念設計段階での結果を受けて要求機能を実現する機構を決定していく．この過程で，一度に設計対象全体を決定することができるわけではない．したがって，具体化できずにあいまいなまま残された部分が存在している状態の設計対象を表現する必要がある．つまり，基本設計段階では一般に設計対象に対する制約が不足しており，そのままでは設計対象をあらわす属性値をすべて決定することができない．そこで本研究ではデフォルト値を導入して，そのような場合でも，制約を解くことができる手法を提案する．

3.1.1 制約表現とデフォルト値

デフォルト値とは，ある時点で，制約条件式を解いても決定できないパラメータに対して与える，ある適当な値である．適切なパラメータにデフォルト値を与えることで，条件式が不足している状態でもパラメータ値を全部決定できるようになる．これにより，パラメータ値を参照することによって，その時点での設計結果を確認することができる，また，制約条件の評価結果を反映した骨格形状モデルや概略形状モデルによっても，具体的に設計結果

を確認できるようになる.

制約モデルでは，主に機構対偶を中心に設計対象をとらえ，機構が満たすべき幾何学的な条件をパラメータの関係式で表現することについては，すでに 2.3.2で述べた．また，設計対象が必ず満足していなければならない条件という意味で，そのようなパラメータ間の関係式を「対偶制約」または単に「制約条件」と呼ぶ．たとえば，2.3.2でも述べたように，簡単な歯車機構に関する制約条件を再掲すると，(3.1) ～ (3.4) 式のようになる.

$$r_1\theta_1 + r_2\theta_2 = 0 \tag{3.1}$$

$$Ra = r_1/r_2 \tag{3.2}$$

$$D = r_1 + r_2 \tag{3.3}$$

$$D^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2 \tag{3.4}$$

ただし,

r_i : $Gear_i$ のピッチ円半径

θ_i : $Gear_i$ の回転角

(x_i, y_i) : $Gear_i$ の中心座標

Ra : ギア比，D : 軸間距離

ところで，(3.1) ～ (3.4) 式は，二つの歯車が満たすべき位置関係や回転角の関係といった，歯車機構の基本的な条件を表現している点では十分であるといえるが，これらの条件式だけからパラメータ値を決定することはできない．さらに，設計の進行にともなって，(3.5) 式や (3.6) 式が与えられたと仮定する．(3.6) 式は，ギア比 Ra の値を 2/3 に指定する制約条件式である.

$$y_1 = y_2 \tag{3.5}$$

$$Ra = 2/3 \tag{3.6}$$

しかし，この段階においても，直接その値を指定した Ra 以外のパラメータに関しては，(3.1) ～ (3.6) 式を解いてもパラメータ値を求めることはできな

い．そのため，たとえばピッチ円半径に対する条件は，(3.2),(3.6) 式などの制約条件式から判断するしかない．

このように，制約表現を用いると，設計初期段階では一般に条件不足のため，すべてのパラメータに関してその値を算出することはできない．その結果，ある時点での設計結果を確認するためには，その時点までに与えられている制約条件式を参照し，それらの条件式から設計結果を判断しなければならないことになる．制約モデルとして与えられている条件式の個数が少ない場合は，制約条件から設計対象の状態を判断することは可能であろう．しかし，条件式の数が増えるにしたがって，それらの条件式によって表現されている機構のモデルを認識することは困難になる．こうした問題を避けるために機構モデルを階層化し，設計者にとってより扱いやすいモデルとして，骨格形状モデル層や概略形状モデル層を導入している．したがって，設計者によって与えられた制約条件が不足している状態でも，すべてのパラメータに関して，その時点での制約条件を満足しているパラメータ値を求め，それらのパラメータ値を反映した骨格形状モデルや概略形状モデルを設計者に示す必要がある．

そこで，始めにも述べたようにデフォルト値を導入する．制約条件が不足している状態でも，適当なパラメータにデフォルト値を与えて条件式を解くことにより，すべてのパラメータに関して値を求めることができるようになる．たとえば (3.7) 式のように歯車の軸間距離 D に対してデフォルト値を与えれば，$r_1 = 20, r_2 = 30$ と求められ，パラメータ値から具体的にピッチ円半径の比が 2/3 であることを確認できる．さらに，各歯車のピッチ円半径がそれぞれ 20 と 30 になるように，骨格形状モデルや概略形状モデルを定義することが可能になる．その結果，歯車に対応する図形を適当な大きさで表示することによって，(3.1) 〜 (3.6) 式を反映した概略形状モデルを設計者に対して示すことが可能になる．

$$D = 50 \tag{3.7}$$

3.1.2 制約条件の分類と定義方法

制約モデルを表現する制約条件は5種類に分類して扱う．制約条件の定義方法や条件式を解く際の扱いが，それぞれの制約条件を定義する目的によって異るためである．例えば，(3.1)式は対偶関係の表現を目的としているし，(3.6)式はパラメータ値の指定を目的とした制約条件といえる．そこで，その制約条件を定義する目的にしたがって条件式を5種類に分類し，各種類ごとに定義方法や制約条件の評価における取り扱いかたを示す．

単純な機構をあらわす制約条件

対偶関係や単純な機構の表現を目的とする制約条件が，一番目の種類としてあげられる．このような制約条件をC^M(Constraints of Mechanism)と表すことにする．ここで，対偶関係だけでなく，「単純な機構」を定義する制約条件としたのは，「リンク」などの，複数の対偶を含む機構要素を表現する制約条件も対象とするためである．本来，機構を表現するためには，機構を構成する部品間の対偶関係と一つの部品に含まれる対偶間の位置関係を定義すればよい．たとえば，リンクの場合には，二つの回転対偶とその回転対偶間の距離を定義すれば，ある一つのリンクを表すことができる．しかし，よく用いる機構はあらかじめ用意しておけば，より簡単に目的とする機構を定義することができるので，対偶に限らず機構単位でも制約条件を扱うことができるようにする．

C^Mに含まれる制約条件は，単純な機構ごとにまとめてモジュール化して扱う．ある一つの対偶や単純な機構を表現するためには，複数の条件式を必要とするのが普通である．また，対偶や単純な機構を表現するために定義すべき条件式が，場合によって変化することはない．つまり，C^Mに含まれる制約条件は，対偶や機構に対応する複数の条件式を必ずセットで定義しなければならない．そこで，対偶や単純な機構ごとに必要な制約条件をまとめ，条件式をモジュール化する．

C^Mを定義する方法としては，制約条件の各モジュールに名前を付け，そ

の名前を指定することによって必要な条件式を生成する方法をとる．目的とする機構を表現するのに必要な制約条件を，ひとつひとつ定義して行くのでは煩雑な作業となる．また，属性モデルや制約条件などの，設計者にとって扱いにくい情報を隠蔽するという目的からも，条件式のレベルで定義して行くことは望ましくない．これに対し，制約条件をモジュール化しておけば，必要な条件式をまとめて定義することが可能である．さらに条件式を直接扱うのに比較して，モジュール名というレベルで条件式を扱うことができる点で有利である．

先に示した歯車機構の例では，(3.1)～(3.4)式がこの種類に属する．これらの制約条件は，単純な歯車機構を使用する際にかならず定義しなければならない制約条件である．場合によって，たとえば(3.2)式のみ定義すれば十分である，ということはない．したがって(3.1)～(3.4)式をひとつのモジュールとすることができる．さらにこのモジュールに対して「歯車機構」という名前を付けておけば，以後「歯車機構」というモジュール名を指定をすることで，歯車機構を表現するのに必要な制約条件式を得ることができる．

接続に関する制約条件

機構要素どうしの接続関係の表現を目的とする制約条件が，次の種類としてあげられる．このような制約条件をC^C(Constraints of Connection)と表すことにする．

設計対象は，モジュール化された制約条件を組合せていくことでそのモデルが構成されていく．その時，各モジュールで表現される機構どうしの関係を定義する必要がある．つまり，モジュール間の位置関係や姿勢に関するパラメータ間の関係を条件式として与えることになる．基本的にはモジュール間の固定関係をあらわす条件式となる．なぜならば，モジュール間で相対的な運動がある場合には対偶関係が生じていることになるので，そうした制約条件は，むしろC^Mに含まれる条件式と考えられるからである．

たとえば，図3.1のように歯車機構のモジュールを接続する場合，図中にも示したように，(3.8)～(3.10)式を制約条件として付加すればよい．また，C^M

と同様に (3.8) ～ (3.10) 式をひとまとめにして，「同軸固定」のようにモジュール名をつけておけば，(3.8) ～ (3.10) 式のような制約条件式を直接扱わずに，モジュール名を指定することで対応する条件式を得ることができる．

$$\theta_2 = \theta_3 \tag{3.8}$$

$$x_2 = x_p \tag{3.9}$$

$$y_2 = y_p \tag{3.10}$$

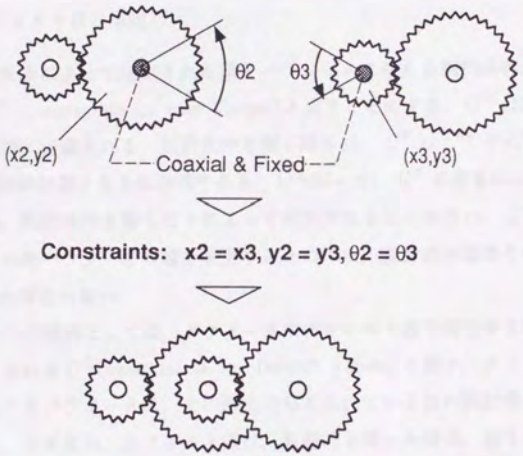


図 3.1: 接続関係の制約表現

パラメータ値を指定する制約条件

制約条件式に含まれるパラメータに対して，値を設定するための条件式である．パラメータ値を指定するためには，「パラメータ＝数値」という形式

の等式によって表現される制約条件を用いる．この「パラメータ＝数値」という式が単に数値の代入を表していると考えると，このような関係式を，C^M や C^C と同様に制約条件とみなすことには違和感がある．しかし，属性モデルにおけるパラメータの値を直接規定していると考えられるので，制約条件として扱うことに問題はない．前述した歯車機構の例では，(3.6) 式や (3.7) 式がこの種類の制約条件に含まれる．

また，パラメータに対して与えられる数値がどのようにして決定されたかによって，さらにこの種類の制約条件を 2 種類に分類する．

- 設計者による指定 (C^V)
- デフォルト値の指定 (C^D)

まず，設計者によって指定された値をパラメータに与える制約条件がある．これを C^V(Constraints for user Values) と表すことにする．C^V は設計者によって個別に定義される．制約条件を解く際には，C^V は C^M や C^C とともに常に評価の対象となる条件式である．したがって，C^V に含まれるパラメータの値が，制約条件を解くことによって変更されることはない．C^V によって設定されたパラメータの値を変更するためには，設計者が直接その値を変更しなければならない．

もう一つの種類としては，パラメータにデフォルト値を指定する制約条件がある．これを C^D(Constraints for Default values) と表す．デフォルト値を与えるべきパラメータは，その時点で与えられている他の制約条件によって決まる．なぜなら，デフォルト値は，条件式を解いた結果，値を計算することができなかったパラメータに対して与えられるものだからである．したがって，C^D は制約条件を評価する過程で生成されるものであり，設計者が直接定義する必要はない．具体的な C^D の生成方法については後で述べることにする．

このように，あるパラメータの値を指定する方法として等式を用いているが，単に入力の形式や表示形式として等式を用いるというのではなく，C^M や C^C と同様に，制約条件として扱うことが重要である．パラメータ値の設

定も条件式とすることにより，計算機による処理の面からパラメータ値の変更が容易になるという利点が得られる．

これに対して，パラメータ値の指定を制約条件として扱うのではなく，たとえば条件式中に含まれるパラメータを，すべて与えられた数値に置き換える方法も考えられる．この場合，数値化された部分の計算は比較的容易であるし，条件式全体を解かなくても局所的に計算を行える場合が多い．また，数値部分の計算を進めることにより，条件式中に含まれるパラメータの数が少なくなるので，制約条件が簡単な式で表され，扱いやすくなるという点で有利である．

しかし，本研究では制約条件をできる限り隠蔽し，直接操作しなくてもすむようにすることを目的としているので，制約条件が簡単な式に変換されなくてもそれほど問題とならない．むしろ，設計においては，パラメータの値がしばしば変更されるので，その変更を容易におこなえることが重要である．パラメータ値の変更に際して，パラメータを数値に置き換える方法では，数値部分の計算を最初からやり直さなければならない．一方，パラメータ値の指定を条件式として与えるのであれば，その条件式中の値の部分だけを変更すればよい．

ユーザ定義の制約条件

設計者が定義した制約条件がこの種類にあてはまる．このような制約条件を C^U
(Constraints by User definition) と表すことにする．「設計者による定義」を広く解釈すれば，C^M, C^C, C^V もユーザ定義の制約条件といえる．しかしここでは，これまでに述べてきた C^M, C^C, C^V, C^D 以外のユーザが定義した制約条件を C^U とする．C^V 同様，C^U も設計者が個別に定義しなければならない．先の歯車機構の例では (3.5) 式がこれにあたる．

C^U の定義に関しては，属性モデルや制約条件を設計者が直接操作しなくてもすむようにするという目的からはずれてしまう．しかし，事前にすべての場合を想定して制約条件を用意しておくことは不可能である．設計者の必

要とする制約条件が，あらかじめ用意してあるとは限らないので，あらたな制約条件を定義する手段を残しておくことは必要である．ただし，設計者が制約条件を一度定義してしまえば，その条件式を再利用することのできるような方法も用意する．一度定義した条件式を C^M, C^C 同様にモジュール化して利用する手段があれば，設計者に対する負担はそれほど大きなものとはならないだろう．

最後に制約条件の分類を表 3.1にまとめる．

表 3.1: 制約条件の分類

制約条件の種類	条件式が表す内容
C^M	単純な機構が満足すべき条件
C^C	接続に関する条件
C^V	設計者によるパラメータ値の指定
C^D	デフォルト値の指定
C^U	設計者による一般的な制約条件

3.2 デフォルト値を含む制約条件の解法

前述したように，本研究における制約条件の評価方法は，適当なデフォルト値を用いながら条件式を解くことにより，条件式が不足している状態であっても，設計過程の任意の時点ですべてのパラメータに関してその値を求めることができる点に特徴がある．以下では具体的にデフォルト値を利用しながら制約条件を解く方法について述べる．ただし，ここで制約条件として扱うことができるのは，多項式によって表された条件式である．

3.2.1 デフォルト値の指定方法

すべてのパラメータは初期値としてデフォルト値を持つ．このデフォルト値を「初期デフォルト値」と呼ぶ．すべてのパラメータに対して初期デフォ

ルト値を設定するのは，設計の初期段階においてまず定義されるのは，対偶関係などを表現する制約条件であること，また本研究では，寸法などを明示的に扱わない段階からの支援を目的としているため，早い段階でのパラメータ値の指定が困難なことなどから，制約条件の定義が先行し，そのあとで適宜パラメータに対する値を指定すると考えられるからである．

初期デフォルト値として適当な値を求めることは容易ではない．そこで，概略形状モデルや骨格形状モデルがスケッチされており，それらのモデルを参照することができる場合には，概略形状や骨格と制約モデルのアイコンとの対応関係から初期デフォルト値を求める．一方，概略形状モデルや骨格形状モデルが参照できない場合は，対偶や単純な機構を表す制約条件 C^M の各モジュールごとに，そのモジュールに含まれる制約条件を満足するように，あらかじめ初期デフォルト値を決めておく．

C^M の各モジュールごとに決められる初期デフォルト値の例として，たとえば「歯車機構」モジュールの場合について以下に示す．制約条件の「歯車機構」モジュールに対しては，(3.1) ～ (3.4) 式を満たす値として，

$$r_1 = 10, r_2 = 20, D = 30, Ra = 1/2$$

などのように初期デフォルト値をあらかじめ決めておく．ただし，各モジュールを構成する制約条件をみたしさえすればよいので，初期デフォルト値の与え方は一意ではない．そして，概略形状や骨格モデルが与えられていない場合は，これらの値を初期デフォルト値として採用する．しかし実際には，こうした初期デフォルト値が用いられることはあまりない．なぜならば通常は，概略形状や骨格形状は先に与えられており，それらのモデルに対して制約条件を対応づけることにより，機構としての意味付けをおこなうからである．

概略形状モデルや骨格形状モデルがすでに定義されている場合は，それらの概略形状や骨格を制約モデルのアイコンと対応づけることにより，初期デフォルト値を求めることができる．概略形状などのスケッチが，実際に目標としている機構と掛け離れているとは考えられないので，事前に用意しておくモジュールごとの初期デフォルト値よりも，スケッチとの対応づけによっ

て得られる初期デフォルト値のほうが適当である.

たとえば,「リンク」をあらわす制約条件のデフォルト値は,図3.2に示したような概略形状が与えられているとすると,スケッチされた部品形状の両端に描いてある円 C_1, C_2 と対応づけることにより,二つの円の中心をそれぞれリンクの端点とし,二つの円の中心間の距離をリンク長さとすることができる.つまり,リンクは(3.11)式によって表されるから,図3.2に示したように,概略形状との対応から,(3.11)式を満足する値として,たとえば(3.12)式のような初期デフォルト値を得ることができる.

$$(x_2 - x_1)^2 + (y_2 - y_1)^2 = L^2 \tag{3.11}$$

$$x_1 = 0, y_1 = 0, x_2 = 40, y_2 = 30, L = 50 \tag{3.12}$$

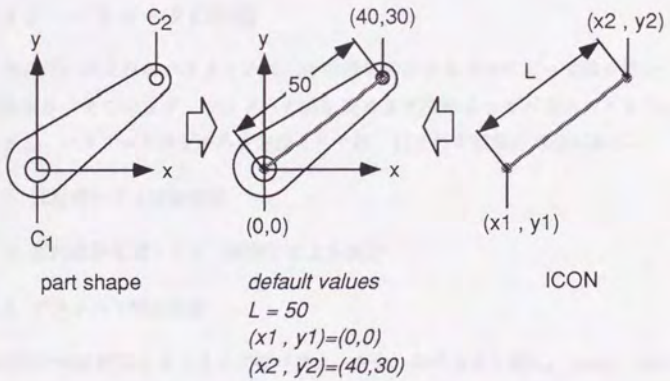


図 3.2: 概略形状を用いた初期デフォルト値の設定

このようにして与えられらた初期デフォルト値は,設計者がパラメータ値を明示的に設定したような効果がある.なぜならば,概略形状のスケッチは,目標としている機構の部品形状を描いたものなので,概略形状との対応によ

り設定されたパラメータ値は，概略形状を介して間接的にではあるが，設計者があたえたパラメータ値であるとみなせるからである．その意味では，初期デフォルト値というよりは，設計者が指定したパラメータ値 C^V に近いといえる．しかし，概略形状が正確に描かれているとは限らないので，場合によっては，制約条件を解くことによって初期デフォルト値は他の値に変更され得る．したがって，設計者が直接与えたパラメータ値とは異なり，スケッチと対応づけることによって得られたパラメータ値を C^V のように扱うことはできない．

また，この方法で初期デフォルト値を得るために参照する図形は，制約条件の各モジュールごとに経験的に決める．したがって，どのような図形が，アイコンのどこに対応するのかといった，アイコンと図形との対応関係をあらかじめ把握しておかなければならない．

3.2.2 パラメータの分類

条件式に含まれるパラメータは，その値を決定する方法によって取り扱いが異なる．そこでまず，パラメータ値を決める方法によってパラメータを分類する．パラメータ値を決める方法としては，以下の3種類の方法がある．

1. 設計者による直接指定
2. 制約条件を解くこと（評価）による決定
3. デフォルト値の指定

各決定方法に対応した3タイプのパラメータを，各々上から順に，*user*，*solved*，*default* というタイプ名で呼ぶことにする．また，各パラメータをあらわす記号として，タイプ別にそれぞれ p^U，p^S，p^D と表すことにする．これらの記号とタイプ名，さらに各パラメータ値を決定した方法の対応をまとめて表3.2に示す．

次に制約条件を解いていく過程における，各パラメータの扱いかたについて述べる．

- p^U の処理方法：パラメータ p_i^U の値が a_i（＝定数）である時，$p_i^U = a_i$ なる条件式を生成し，この制約条件式の集合を C^V とする．設計者が直接指定した値に関しては，C^V としてかならず C^M や C^C とともに解かなければならないことは既に述べた．制約条件の分類のところでは，「パラメータ＝数値」という形式の条件式で C^V を定義するとしたが，実際にはパラメータのタイプを *user* に変更することによって C^V を定義する．つまり，「パラメータ＝数値」形式の条件式は設計者が入力する必要はなく，パラメータのタイプにしたがってシステムが自動的に生成する．先にも述べたように，p^U の値は制約条件を解いても変更されない．p^U の値を変更するためには，設計者が直接パラメータ値を指定する必要がある．

- p^S の処理方法：p^S は制約条件を解く過程で，その値を算出することができたパラメータである．したがって，制約条件を解かずにこのパラメータの値を変更することは許されない．制約条件と無関係にパラメータ値を変更すると，一般には条件式を満たさなくなるからである．

- p^D の処理方法：パラメータ p_i^D の値が b_i（＝定数）である時，$p_i^D = b_i$ なる条件式を生成し，この制約条件式の集合を C^D とする．p^U と同様，「パラメータ＝数値」の形式の条件式を設計者が定義する必要はない．パラメータのタイプにしたがってシステムが自動的に生成する．デフォルト値を指定する制約条件 C^D の具体的な利用方法は 3.2.3で述べる．

こうしたパラメータのタイプは，設計を進めて行く過程で変更することができる．しかし，すべての変更の仕方が可能なわけではない．たとえば，いずれのタイプも *"solved"* に変更することはできない．p^S はかならず制約条件を解く過程で決定されるパラメータである．また，p^S に含まれるパラメータのタイプを *"user"* に変更することは可能ではあるが，制約条件としては冗長

になる[1]．たとえば，ある p_i^S の値が α であるとき，このパラメータを *"user"* タイプに変更すると，

$$p_i^S = \alpha \tag{3.13}$$

なる式を加えて制約条件を解くことになるが，それ以前に与えられていた制約条件からも (3.13) 式が導出されるので，(3.13) 式が冗長に与えられていることになる．これらをまとめて表 3.3に示す．表の左の列に示してあるタイプのパラメータから，上の行に示してあるパラメータタイプへの変更が可能な場合は○，不可能な場合を ×，望ましくない場合を△で示してある．

表 3.2: 制約条件の評価におけるパラメータの分類

記号	タイプ名	値の決定方法
p^U	*user*	設計者による指定
p^S	*solved*	制約条件の計算結果
p^D	*default*	デフォルト値の指定

表 3.3: パラメータタイプの変更

From/To	p^U	p^S	p^D
p^U	–	×	○
p^S	△	–	○
p^D	○	×	–

3.2.3 デフォルト値を用いた制約評価の方法

制約条件は，デフォルト値を用いながら以下のような手順で評価して行く．

[1] p^U に含まれるパラメータのタイプを変更した場合は，かならずしも冗長になるとはかぎらない．

Step1　この段階で定義されている制約条件式に含まれているパラメータのうち，すべての p^S のタイプを *default* に変更する．

この時点での p^S は，直前の制約評価によってタイプを *solved* に設定されたパラメータである．したがって，その後，パラメータ値を指定する条件式も含めて新たな条件式が追加されたり削除されたりした場合，現段階で p^S であるパラメータが再び *solved* になるとは限らない．

また，制約条件の評価プロセスの終了条件として，C^V や C^D として値を代入したパラメータ以外のパラメータがすべて *solved* になった時，プロセスを終了させる．このためにも *solved* というタイプは，条件式を解くプロセスを始める前にリセットしておく必要がある．

Step2　5種類の制約条件のうち，デフォルト値を設定するための条件式を除いたすべての条件式，すなわち C^M，C^C，C^V，C^U をまとめてブフバーガ (Buchberger)・アルゴリズム[2]により簡略化する．

与えられた制約条件を簡略化した結果 $\{1\}$ を導いた時には，矛盾する式を含んでいたとして異常終了する．それ以外の場合，簡略化した結果の式の中から1変数の方程式を抽出し，その解を求め，他の方程式に結果を伝播させる．また，解が算出できたパラメータのタイプを *solved* にする．このような操作の結果，すべてのパラメータについて値を求めることができていれば完了．値の決まらないパラメータがある場合には，それ以上解けない条件式が残るので，それらの条件式を解くために次のステップに進む．

条件式が不足しているため，この段階ではまだ値を算出することのできないパラメータが，通常残っていることは既に述べた．そこで，次のステップでデフォルト値を用いることにより，さらに計算を進めてすべてのパラメー

[2]ブフバーガ・アルゴリズムは，連立方程式をある評価基準に基づいて式変形するアルゴリズムである．したがって，このアルゴリズムによって得られる結果は変形された式であり，2次以上の方程式に関しては，方程式の解まで求められるわけではない．また，矛盾する式を含む時，与えられた式がどのようなものであっても結果として $\{1\}$ を導く．このアルゴリズムについては後で詳細を述べる．

タに関してその値を求めていく.

Step3 C^D の中から適当に一つの条件式を取りだして, Step2 で残った条件式と合せて, ブフバーガ・アルゴリズムで再び簡略化する.

値が求まらずに残っていたパラメータすべてについて, その値を求めることができていれば完了. 簡略化の結果が $\{1\}$ の時には異常終了する. それ以外の場合次のステップに進む.

Step4 解の算出が可能な方程式を解きながら, その結果を他の方程式に伝播し, 潜在的に決定されているパラメータの値をすべて求める.

簡略化の結果得られた条件式の中から, 1 変数の方程式を抽出しその解を求める. 値を決定することができたパラメータのタイプを *solved* に変える. また残りの式に含まれているそのパラメータを, すべて算出されたパラメータ値に置き換える. この操作を一つの方程式を解くたびに行い, 1 変数方程式がなくなるまで繰り返す.

すべてのパラメータについて値を求めることができていれば完了. 値を決めることができないパラメータがある場合には, かならずそのままではそれ以上解けない条件式が残るので, それらの条件式をさらにデフォルト値を利用して解くために, Step3 に戻る.

Step4 の操作が必要な理由は, ブフバーガ・アルゴリズムは式変形をするアルゴリズムであり, 2 次以上の方程式の解を求めることができないことによる. たとえば, (3.14) 式と (3.15) 式がブフバーガ・アルゴリズムによって得られたとする.

$$x^2 = 1 \tag{3.14}$$

$$x + y = 3 \tag{3.15}$$

まず (3.14) 式は 1 変数方程式なので, x の値を求めることができることがわかる. 一方 (3.15) 式は, この式だけ単独でみれば 2 変数を含んでいるので, これ以上解くことができない. しかし, (3.14) 式を解いて x の値を $x = \pm 1$ と求めれば, たとえば $x = 1$ を用いて $y = 2$ と求めることができる. した

がって，この場合 y の値は潜在的に決定されていたとみなせる．このように，単独では解を求めることができない式であっても，他の式で求められた解を用いることで，最終的に計算可能となる場合がある．こうした方程式をもれなく計算するのが Step4 の役割である．また，以下にはデフォルト値を利用した制約条件の評価方法を疑似プログラム形式で示す．

Procedure **Main**

begin

forall p^S *do* $type(p^S) \leftarrow default$;

$C^{(0)} \leftarrow C^M \cup C^C \cup C^U \cup C^V$;

$C^{(0)} \leftarrow Buch(C^{(0)})$;

if $C^{(0)} = \{1\}$ *then* 異常終了 *else* *Calc&Prop*;

if $C^D = \phi$ *then* 完了;

$n \leftarrow 0$;

while $C^D \neq \phi$ *do*

  begin

  $c^D \leftarrow select(C^D)$; $C^D \leftarrow C^D - \{c^D\}$;

  $C^{(n+1)} \leftarrow C^{(n)} \cup c^D$; $n \leftarrow n + 1$;

  $C^{(n)} \leftarrow Buch(C^{(n)})$;

  if $C^{(n)} = \{1\}$ *then* 異常終了 *else* *Calc&Prop*;

  if $C^D = \phi$ *then* 完了;

  end

end

Procedure **Calc&Prop**

forall $c \in C^{(n)}$ *do*

  if cが 1 変数(p)の方程式 *then*

    begin

    $value(p) \leftarrow Calc(c)$; $type(p) \leftarrow solved$;

$C^D \leftarrow C^D - \{c\}$; $C^{(n)} \leftarrow C^{(n)} - \{c\}$;
$C^{(n)}$中の p をその値 *value(p)* に置き換える
end

Main は条件式を解くメインループである．*while* 以前の部分が Step1 と Step2 にあたる．全ての p^S に対しタイプを *default* にリセットし，最初の一回はデフォルト値を用いずに条件式を解いている．次に *while* 内部が Step3 に対応する．デフォルト値を一個ずつ使いながら処理を進める．**Calc&Prop** は Step4 に対応する，計算可能な条件式を探し，その結果を他の条件式に伝播させる手続きである．*Calc* は 1 変数方程式の解を返す関数である．また *Buch* はブフバーガ・アルゴリズム [Buchberger 85] によって簡略化された方程式を返す関数である．

Buchberger アルゴリズム Buchberger（ブフバーガ）アルゴリズムは，変数の重みづけにしたがって多項式の各項の重みを計算し，全体として重みが小さくなるように項を書き変える数式処理アルゴリズムである．例えば，連立方程式 (3.16) は，x,y の順に重みづけをすると (3.17) 式のように簡略化される．

$$y^2 + x^2 - 4 = 0, y - x = 0 \tag{3.16}$$

$$x^2 - 2 = 0, x - y = 0 \tag{3.17}$$

なお矛盾する方程式を含む場合，簡略化の結果は常に $\{1\}$ になる．**Main** における異常終了がこれにあたる．

3.3 区間制約を含む制約条件の解法

機構の中には運動にともなって制約条件が変化するものがある．そうした機構を表現するために，区間制約を導入し，同時にその評価方法を示す．

機構の中には，運動にともなって制約条件が変化するものがある．その例を図 3.3に示す．この機構は，VTR テープ・ローディング機構の一部である．

Gear1,2 を駆動すると Link1 ～4 が動き，Slider1,2 にそって動くスライドピン P1,P2 に取り付けられたテープ引き出し体によってテープがカセットから引き出され，ヘッドに巻き付けられる．この機構において，Slider1 は Slider1-1,1-2,1-3 の各部分ごとに異なる制約条件で表現され，ピンの位置によって条件式を切替える必要がある [吉川 91a].

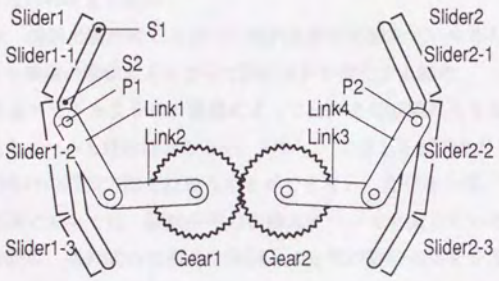


図 3.3: VTR テープローディング機構

例えば S1(x_1, y_1) を Slider1-1 の端点，S2(x_2, y_2) を Slider1-1 と 1-2 の接続点，(x_p, y_p) を P1 の位置とすると，Slider1-1 は (3.18) ～ (3.20) 式に示す制約条件式で表現される．

$$(x_p - x_1)(y_2 - y_1) = (y_p - y_1)(x_2 - x_1) \tag{3.18}$$

$$x_2 \leq x_p \leq x_1 \tag{3.19}$$

$$y_2 \leq y_p \leq y_1 \tag{3.20}$$

(3.18) 式は S1,S2 を通る直線上に P1 があるという条件を表している．また (3.19) 式と (3.20) 式は S1 と S2 の間に P1 があるという条件をそれぞれ表している．同様にして S3(x_3, y_3) を Slider1-2 と Slider1-3 の接続点とすると，Slider1-2 は (3.21) ～ (3.23) 式であらわされる．

$$(x_p - x_2)(y_3 - y_2) = (y_p - y_2)(x_3 - x_2) \tag{3.21}$$

$$x_2 \leq x_p \leq x_3 \tag{3.22}$$

$$y_3 \leq y_p \leq y_2 \tag{3.23}$$

このように，Slider1-1 と Slider1-2 はそれぞれ異なる制約条件で表される．したがって，スライドピン P1 が，Slider1-1 から Slider1-2 へ移動するのにともなって，Slider1 を表す制約条件式も (3.18) ～ (3.20) 式から (3.21) ～ (3.23) 式へ切換えなければならない．

こうした，機構の動作にしたがって制約条件を切換えていかなければならないこうした機構の運動にともなって制約条件が変化する場合，

これまで述べてきたように，機構によっては，その運動にともなって制約条件を切換えていかなければならない．しかし，こうした制約条件は，3.2.3で示した制約条件の評価方法では扱うことができない．なぜならば，3.2.3で示した評価方法においては，制約条件の切換えについて考慮していないからである．また仮に，条件式の切換えを無視して全部の制約条件を一度に解くとしても，三つの部分スライダ Slider1-1,1-2,1-3 上に，スライドピンが同時に存在することはありえないので，制約条件の連立方程式は常に「解なし」となり，パラメータ値を計算することはできない．さらに，3.2.3で示した評価方法で扱うことのできる条件式は，等式で表現された制約条件のみであり，区間制約を評価することはできない．

以下では，制約条件の切換えに対応し，区間制約も含めた条件式の評価方法について 4 段階にわけて述べる．また 3.2.3で述べた，デフォルト値を利用した制約条件の評価方法のことを「基本評価法」と呼ぶ．

3.3.1 切換えを必要とする制約条件の表現

制約変化をともなう機構を表す制約条件を C^{SW} とする．制約条件を切換える必要があるとしても，機構を表現する条件式なので，

$$C^{SW} \subset C^{M} \tag{3.24}$$

である．また (3.19) 式と (3.20) 式がともに成り立つときのみ (3.18) 式が意味を持つことから，これらをまとめて，(3.25) 式のように表す．つまり (3.25)

式は，前半部分の不等式を満足する時のみ後半部分の関係式を用いることができる制約条件を表している．このような成立条件つきの制約を C^L と表すことにする．

$$x_2 \leq x_p \leq x_1,\ y_2 \leq y_p \leq y_1 \Rightarrow$$
$$(x_p - x_1)(y_2 - y_1) = (y_p - y_1)(x_2 - x_1) \tag{3.25}$$

C^{SW} は C^L を用いて表される．たとえば，図3.3に示した機構の中で，屈曲した経路をもつスライダ Slider1 をあらわす制約条件を c_1^{SW} とし，Slider1 を構成する部分スライダ Slider1-1，Slider1-2，Slider1-3 をあらわす制約条件をそれぞれ $c_{1,1}^L$，$c_{1,2}^L$，$c_{1,3}^L$ とすると，c_1^{SW} は (3.26) 式のように定義できる．

$$c_1^{SW} = c_{1,1}^L \vee c_{1,2}^L \vee c_{1,3}^L \tag{3.26}$$

つまり Slider1 をあらわす制約条件は，各部分スライダをあらわす条件式をひとまとめにしたものとして表現される．このように表現すると，$c_{1,1}^L$，$c_{1,2}^L$，$c_{1,3}^L$ のうち，実際に成立する条件式はどれか一つしかないことから，矛盾を生じるように見える．しかし C^L は成立条件つきなので，c_1^{SW} は (3.26) 式のように各部分スライダをあらわす条件式の選言でなければならない．

以上のことから，一般に k 通りの場合に制約条件を切替える必要のある機構は (3.27) 式のように表される．

$$c_i^{SW} = \bigvee_{j=1\ldots k} c_{i,j}^L \tag{3.27}$$

また以下では説明のために，C^{SW} 以外の場合わけを必要としない制約条件をまとめて C^{Rest} とする．図3.3の例では，歯車やリンク機構の制約条件などがこれに含まれる．

3.3.2 制約条件の組合せ

前項の結果を用いると，機構全体は形式的に $C^{Rest} \cup C^{SW}$ と表せる．しかし，実際には C^L のような成立条件つきの制約式と，C^{Rest} に含まれる等式

表現された通常の制約式とをあわせて解く方法はない．したがって，C^{SW} の構成要素である c^L を適当に選び，その後半部分の関係式のみを用いて機構全体の制約条件を解く必要がある．

基本的には全ての場合について制約条件式を解く必要がある．制約条件の切換えを必要とする対偶（c^{SW}）が n 個存在し，各々の対偶について k_i 個の場合わけが必要ならば，すべての場合を網羅すると $k_1 \times k_2 \times \ldots \times k_n$ 通りの組合せができる．たとえば図 3.3の機構では，各スライダとも 3 通りに場合分けされるので，$3 \times 3 = 9$ 通りの制約条件の組合せを必要とする．

しかし，機構によっては組合せ数を減らすことができる．例えば図 3.3では，一度にすべての制約条件を解くのではなく，まず Link4,P2 間を切り離し Slider2 は無関係としたうえで，左から Link4 までの機構を評価する．これにより Slider1, Link1, Link2, Gear1, Gear2, Link3 の位置・姿勢を求めることができる[3]．その後，Slider1 から Link3 までの位置・姿勢はそのままとして，残りの機構（Link4, Slider2）を評価する．この場合，解かなければならない制約条件の組合せは (3.28) 式のようになり，組合せの数は $3 + 3 = 6$ 通りに減る．ただし $c^L_{i,j}$ は Slider i における j 番目の部分スライダを表す制約条件である．

$$C^{Rest} \cup \{c^L_{i,m}\},\ i = 1, 2,\ m = 1, 2, 3 \tag{3.28}$$

このように制約条件を分割して解くことができる理由は，次のように考えることができる．図 3.3において，制約条件を解いた結果 Slider1 と Slider2 の両方を同時に満足する解が得られなければ，機構全体として制約条件を満たしたことにならないのは明らかである．つまり，Slider1 のみ成立する解が仮にあったとしても，機構全体としてみるとそのような状態はありえない．逆に，Slider1 と Slider2 のいずれか一方でも，制約条件を満足する解が求められない場合は，その時点で機構全体として制約条件を満足する解がないと判断してよい．したがって Slider1 をまず調べ，そこで制約条件を満たす状態を

[3]この状態では Link4 の位置・姿勢を一意に決定することはできない．Link4 はデフォルト値から求められた適当な位置・姿勢をとる．

求めてからSlider2について調べても，すべての組合せを解いた場合と同じ結果を導くことができる．

3.3.3 評価手順のAND/OR木表現

制約条件の組合せ方法と，各組合せの計算結果を機構全体として評価する方法の両方をAND/OR木で表現する．木構造中のノードはC^{Rest}やC^{L}などに対応し，これを制約ノードと呼ぶ．図3.4は図3.3に対応するAND/OR木[4]である．

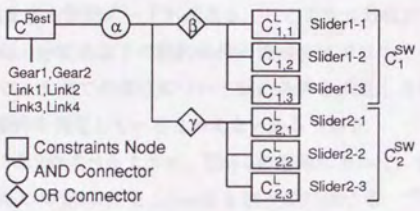


図 3.4: 制約条件評価のためのAND/OR木

制約条件の組合せは，AND/OR木の根から末端への経路に含まれる制約ノードの和集合として表される．たとえば，図3.4の一番上の経路をとると，その経路中にはC^{Rest}と$c^{L}_{1,1}$の二つの制約ノードが含まれるので，制約条件の組合せは

$$C^{Rest} \cup \{c^{L}_{1,1}\}$$

と表される．

[4]図に示したANDコネクタとORコネクタからは，それぞれ1本のアークしかでていないように描かれているが，これは単に作図上の理由からである．実際にはANDコネクタ，ORコネクタから直接，複数のアークが分岐して出ていると考えてよい．

この例からもわかるように，制約ノードの集合を求める際にANDコネクタやORコネクタは無関係である．したがって，制約条件の組合せ方法を記述する手段としては，実際には木構造を用いれば十分である．しかし，次に述べるように，機構全体を評価する時に用いるAND/OR木は，制約条件の組合せをあらわす木構造と構造が同じなので，両者を統一してAND/OR木で表現している．

AND/OR木の分岐におけるコネクタ種類の選択方法について次に述べる．ANDコネクタは，制約条件の組合せ数を減らすために，制約条件を分割して解く場合に用いる．したがって，ANDコネクタから分岐するアークには，C^{SW} の各要素を含む制約ノードが連なる．このような分岐がANDコネクタに対応するのは，分岐点以下の制約条件が機構全体ではなくその一部分を表現しているので，すべての部分について制約条件を満足しなければ，機構全体として制約条件を満足しているといえないからである．

たとえば，3.3.2で述べたように，図3.3の機構において，制約条件の組合せ数を減らすためにSlider1とSlider2を別々に評価する．Slider1，Slider2をあらわす制約条件式をそれぞれ c_1^{SW}，c_2^{SW} とすると，解くべき制約条件の組合せは，$C^{Rest} \cup c_1^{SW}$ と $C^{Rest} \cup c_2^{SW}$ となり，両方の組合せについて解が存在しなければ機構全体として制約条件を満足したことにならない．そのため，図3.5に示したように，ANDコネクタ（α）から分岐したアークに c_1^{SW} と c_2^{SW} がそれぞれ接続される．

ORコネクタは，個々の c^{SW} の場合わけを目的とした分岐に用いる．したがって，ORコネクタ以下には C^L を含む制約ノードが連なる．この場合は，各 c^{SW} が c^L の選言として定義されていることから，コネクタ以下のどれか一つの場合について解が存在すればよい．したがって，このような分岐にはORコネクタを用いる．

たとえば，図3.5に示したように，c_1^{SW} はORコネクタ β とそのコネクタに接続する $\{c_{1,1}^L, c_{1,2}^L, c_{1,3}^L\}$ の各制約条件に置き換える．また c_2^{SW} についても同様にORコネクタ γ と $\{c_{2,1}^L, c_{2,2}^L, c_{2,3}^L\}$ の各制約条件に置き換える．その結果図3.4に示したAND/OR木が構成される．このようにして構成されたAND/OR

構造が機構全体としての制約条件の評価方法を表す.

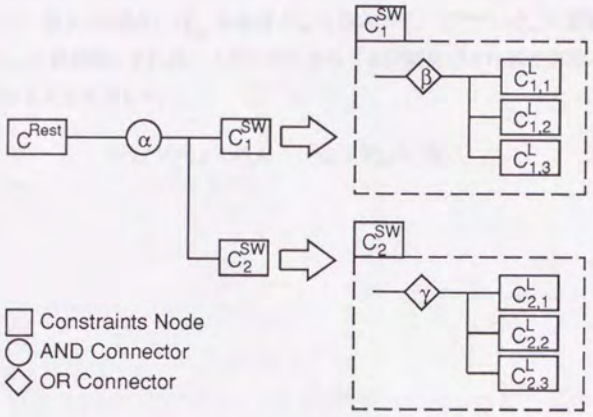


図 3.5: AND/OR 木の構成過程

3.3.4 AND/OR 木による区間制約の評価

不等式はスラック変数によって等式化する方法 [Kapur 88] もあるが，ここでは等式条件とは別扱いにし，まず AND/OR 木により制約条件の組合せを決め，等式条件のみ 3.2.3で述べた基本評価法を用いて解いた後に不等式を評価する．ここで扱う不等式は，パラメータ値の存在範囲を与えるものなので，条件つき最適化問題のように他の条件式と同時に処理していく必要はなく，基本評価の結果得られたパラメータ値の検証に用いればよいためである．ここで与えられたすべての不等式を満足するとき，「成功」と呼び，それ以外を「失敗」と呼ぶ.

また末端ノードを，その制約ノードを含む制約条件の組合せを解いた結果と置き換えていく．したがって，木構造によって表されている制約条件の組合せをすべて解けば，すべての末端ノードに「成功」か「失敗」の結果が割

り当てられる．そして，AND/OR木にしたがって，末端から根のノードへ評価を進め，根のノードが成功になれば機構全体で条件を満足していると判断する．図3.4の場合，$c^L_{i,m}$ を命題 $P_{i,m}$ と読みかえ，$C^{Rest} \cup c^L_{i,m}$ の評価結果を $P_{i,m}$ の真偽値とすれば，AND/OR木による評価は(3.29)式の成功／失敗を求めることに等しい．

$$(P_{1,1} \vee P_{1,2} \vee P_{1,3}) \wedge (P_{2,1} \vee P_{2,2} \vee P_{2,3}) \tag{3.29}$$

第4章

制約に基づく形状操作

4.1 形状変形操作の要件

設計初期段階で用いるためには，寸法を導入しないレベルでの形状を，機構に関する制約条件を満足するように変形操作する必要がある．従って

1. 対話的変形
2. 寸法値不要

などの特徴を持つ形状変形操作が必要である．

これらの要件を満たす形状操作法として，骨格形状モデルを用いた概略形状の変形方法を提案する．この方法ではセグメント単位で操作するのではなく，複数のセグメントをまとめて操作するので，変形に対するユーザのコストが少なくて済む．形状要素（線分，円弧）を直接ユーザが変形するのは，特にある制約条件を満たすように変形しなければならない場合，かなり繁雑な操作であるといえる．これに対し，骨格形状モデルと対偶モデルのアイコンとの対応関係がよいことから，アイコンと骨格形状を経由することによって，機構に関する条件を満たすように概略形状を変形することができる．

ただし，骨格形状モデルの操作から概略形状を決定するためには，何らかの前提をを必要とする．なぜなら，骨格形状モデルの持つ情報は，概略形状モデルの持つ情報に1対1に対応するとは限らず，一般に骨格形状モデルの持つ情報のほうが少ないと考えられるからである．そこで本研究では，骨格

形状モデルにおける操作を，概略形状モデルに反映させる変換法則をあらかじめ定義しておく．したがって，実際に行なえる概略形状の変形操作は，変換法則により規定される．

4.2 骨格形状モデルによる形状変形

4.2.1 骨格形状モデルの操作

骨格形状は先に示したように，その位相構造をグラフ構造によって表現する．そこで，骨格形状モデルの構成要素を示すために，「骨格アーク」（または単に「アーク」）と「骨格ノード」（あるいは「ノード」）という名称を用いる．ただし，骨格の幾何情報も重要なので，「骨格アーク」は単にノード間の接続関係を示すだけでなく，ノード間の距離も表す．また，「骨格ノード」はアークとアークの接続点としての意味を持つだけでなく，自分自身の位置（座標値）も情報として持つものとする．

骨格形状モデルの操作としては，表4.1に示すように，ノードとアークそれぞれに対して3種類，合計6種類の操作が可能である．これらの操作を図4.1〜図4.7に示す．「ノードの生成」とは，アーク上にあらたなノードを生成する操作，「ノードの消去」はその逆操作である．図4.1にノードの生成・消去の例を示す．また，「アークの生成」とは，すでにあるノードとあらたなノードの間にアークを張る操作，「アークの消去」は，グラフの末端のアークを取り除く操作である．図4.2にアークの生成・消去の例を示す．

表 4.1: 骨格形状モデルに対する操作

操作の対象	ノード	アーク
可能な操作	生成	生成
	消去	消去
	移動	移動

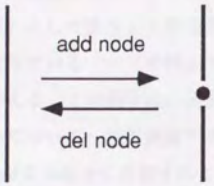


図 4.1: 骨格ノードの生成・消去操作

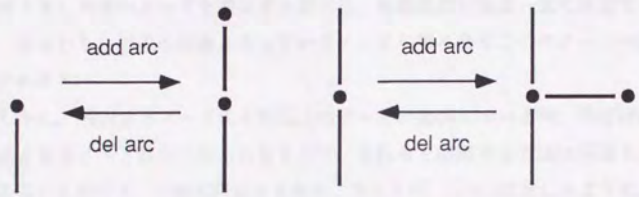


図 4.2: 骨格アークの生成・消去操作

設計者は，骨格アークや骨格ノードの生成・消去操作によって骨格形状モデルを構成していく．アークやノードの生成操作によって，次第に骨格形状モデルを詳細なものにしたり，また逆に，消去操作によって骨格形状モデルの不必要な枝を刈り，モデルを簡潔な構造にすることなどを通じて，適当な骨格形状モデルを生成する．こうした，骨格形状モデルを構築していく過程では，生成操作，消去操作ともその実行に対して特別な制限はない．しかし，概略形状モデルと対応づけた後では，アークやノードを消去する操作の実行は制限され，いかなる場合でも実行できるわけではない．たとえば，ノード消去の実行には，以下に示す二つの条件を満たしていなければならない．

1. ノードに接続しているアークが2本であること

2. 2本のアークのなす角度が180度であること

1番目の条件を満たさない場合として，アークが1本の場合が考えられるが，これは「アークの消去」として扱うことができ，特に問題はない．ここでは，消去操作の対象となっているノードに対して，3本以上のアークが接続している場合について考える．この場合は，ノードを消去した後アークをどのように接続しなおすかについて一意に決定できない点と，互いに接続しないとすると，グラフ構造が2つ以上に分割されてしまう点に問題がある．

まず，ノード消去後のアークの再接続に関する問題について述べる．ノードの消去にともなって，通常アークの本数も減少する．その結果残ったノード間をあらたなアーク接続しなおすことになる．ここで，接続しているアークが2本しかないノードを消去する場合は，再接続の方法は一意に決定できる．すなわち，消去の対象となっているノードに隣り合う二つのノードを接続すればよい．

しかし，消去するノードに3本以上のアークが接続している時，再接続の対象となるノードは三つ以上になるので，それらを接続する方法は何通りか考えることができ，一意に決定できない．たとえば，図4.3に示したように，3本のアークが一つのノードを共有するように接続している場合，共有されているノードを消去すると，残りの三つのノードを接続する方法はループを除けば3通りある．そのうちのどの接続方法を選択すべきか決定する方法はない．

また，ノード消去後の最接続の方法が一意に決定できないことから，再接続せずに，消去対象となっているノードを含むアークをすべて消去してしまう方法も考えられる．しかしこの場合，上述したように骨格形状モデルのグラフ構造が二つ以上に分裂してしまう問題を生じる．これについては，アーク消去の問題点として詳しく述べることにする．

次に2番目の条件である，2本のアークのなす角が180度を満たさない場合について述べる．この場合，ノードの消去は実現可能ではある．しかし，ノードを消去した後のアークの方向が，消去前の方向と異なってしまう点で問題となる．なぜなら，アークの方向は対応する図形の軸を表しているので，その方向を変更する場合はユーザが指定すべきである．それに対し，ノード

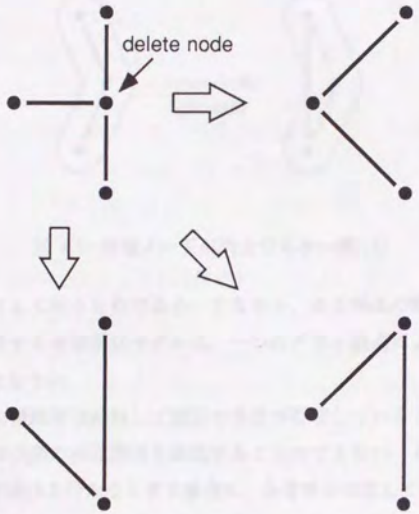


図 4.3: 骨格ノードが消去できない例 (1)

消去の結果，操作の始めに指定されていた方向と異なるアークを自動的に生成することは望ましくない．図 4.4 2番目の条件を満足していない例を示す．仮にノードを消去してしまうと図の右のようになり，概略形状との対応が明らかに不適当である．

アーク消去の実行に関しては，

- 末端のノードに接続するアークであること

が条件となる．すでにノード消去の問題点として指摘したように，この条件を満足しない場合，骨格形状モデルのグラフ構造が二つ以上に分離してしまう点に問題がある．二つ以上の骨格が同一部品内に存在することが問題となるのは，骨格形状モデルと概略形状モデルは，部品単位で 1 対 1 対応してい

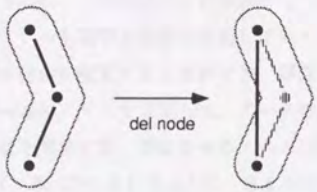


図 4.4: 骨格ノードが消去できない例 (2)

ることを前提としているためである．すなわち，ある部品の概略形状モデルの「軸」を表現する骨格形状モデルは，一つのグラフ構造によって表現されていなければならない．

ある一部品の概略形状に対して複数の骨格が存在していると仮定すると，それらの骨格形状間の相互関係を表現することはできない．その結果，骨格による形状変形操作を行おうとする場合に，各骨格が独立していることから，部品全体の形状の整合性を管理できなくなる．さらに，骨格が分散して位置しているために，いずれの骨格にも対応づけることができない形状部分が生じてしまい，そのような部分の形状変形操作を定義することができなくなることからも，一つの部品形状に対応する骨格形状モデルは，一つのグラフ構造で表現されていなければならない．

ただし，この条件を満足していてもアークの消去が適当であるとは限らない．なぜなら，アークを消去するということは，そのアークの軸方向への形状変形手段がなくなるということを意味するからである．もともとアークが存在していたからには，そのアークに対応する「軸」を想定できるような形状が，そこに存在しているはずである．そうした特徴的な形状の操作が必要でなくなることは，形状自体が変更されてしまわない限り，通常ではあまりないことであると考えられる．つまり，アークの消去は，対応する概略形状の変更なしに実行されることはほとんど無いといってよい．

以上で述べた生成・消去操作は，骨格形状モデルのグラフ構造を構築して

いくことができる．さらに，「骨格ノードの移動」と「骨格アークの移動」によって，骨格形状モデルを適当な形状に修正していくことができる．たとえば，ノードの位置は自由に変更することができ，移動したノードに接続しているアークのもう一方のノードを固定して，アークの長さを伸び縮みさせることにより骨格形状を変更する．移動させるノードは末端のノードに限定されない．したがって，図4.5に示したように，3本のアークが交差しているノードを移動することも可能である．図4.5の場合，位置を変えないノードは n_1, n_2, n_3 である．

ただし，図4.6に示したような移動はできない．図4.6(a)は，骨格のトポロジが変化してしまう移動の例である．(b)はアークどうしが交差する点に問題がある．いずれの場合においても，このような骨格形状モデルの変形結果は，骨格として無意味とは必ずしもいえない．しかし，この骨格に対応づけられる概略形状を考慮すると，骨格のトポロジが変化したり，骨格が交差するような形状変形はありえない．その結果として図4.6に示したような骨格の変形は許されないことになる．

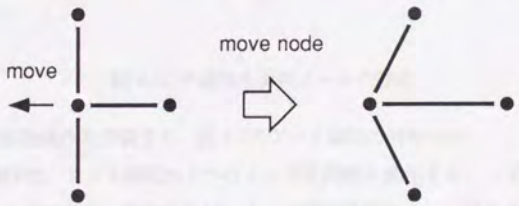


図 4.5: 骨格ノードの移動操作

アークの移動はノード移動の組合わせによって実現可能なので，操作の定義としては冗長である．このような移動操作を定義するのは，アークの姿勢を保ちながら位置を変更する操作を容易にするためである．すなわち，移動するアークの長さと姿勢を変更しないように平行移動のみを許すものとして，

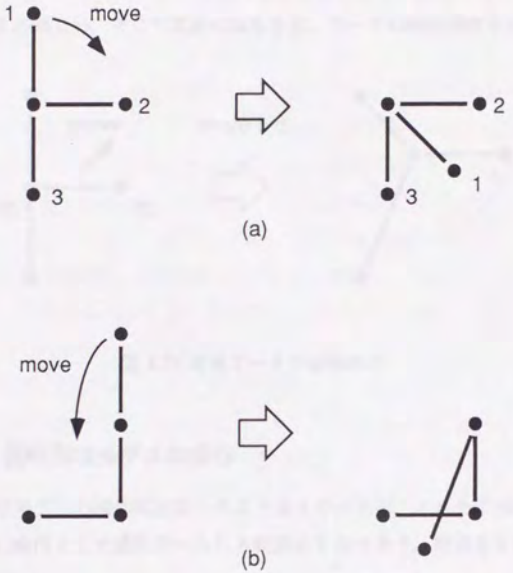


図 4.6: 不適当な骨格ノードの移動

アークの移動操作を定義する．図 4.7にアーク移動の例を示す．

この操作は，アーク両端の 2 つのノードを同時に操作するノードの移動とみなすことができる．換言すれば，ノード移動操作によって図の n_0 と n_2 を別々に動かすことによっても，同じ結果を得ることができる．それにもかかわらず，このような骨格アークの移動操作を定義するのは，骨格形状モデルを概略形状モデルに対応づけた時に，骨格のアークの方向が，対応している概略形状の「軸」という重要な意味をもつからである．たとえば，あるアークに対応する形状部分の位置を変更する時に，その位置のみを変更し「軸」の方向は保持したままであることが要求される場合が多い．このような場合，

ノードの移動操作のみでアークの方向を変えないように変形することは，可能ではあるが難しい．そこで冗長にはなるが，アークの移動操作を定義する．

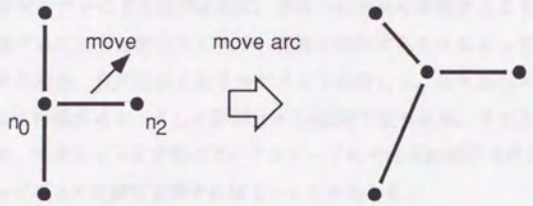


図 4.7: 骨格アークの移動操作

4.2.2 概略形状モデルの操作

概略形状モデルの操作には以下のようなものがある．これらの操作は，2次元形状の操作として通常用いられる範囲のものであり，特別なものは含まれていない．

1. セグメント単位の操作
   (a) 生成
   (b) 移動
   (c) 変形
   (d) 消去
2. 複数セグメントに対する操作
   (a) 移動
   (b) 拡大・縮小
   (c) 消去

4.2.3 骨格形状モデルと概略形状モデルの対応関係

対応関係の定義

骨格形状モデルによる形状変形は，骨格の軸方向に伸縮するような形状変形を実現することを目的としている．骨格を操作することによって概略形状を変更する場合，変更対象となるセグメントに対して，各々適用する処理が異なる点に特徴がある．そして詳細は4.2.5以降で述べるが，そのような処理の相違が，セグメントを骨格のアークとノードにそれぞれ対応づけると，アークとノードごとに処理を定義すればよいことがわかる．

そこで，その対応づけをノードやアークの影響がおよぶ勢力圏を考え，各勢力圏に含まれるセグメントをそれぞれの勢力圏に影響を及ぼすノードやアークに対応づける．ここで，ノードの勢力圏に含まれるセグメントをS^N，アークの勢力圏に含まれるセグメントをS^Aとする．また，S^Nのうち骨格の分岐点にあるノードに対応するものは，特別な処理を行う必要があるので，特にS^Jと表すことにする．たとえば図4.8に示した部品形状と骨格形状の場合，S^NとS^Aはそれぞれ(4.1), (4.3)式のようになる．

$$S^N = \{s1, s4\} \tag{4.1}$$

$$S^J = \{s6\} \tag{4.2}$$

$$S^A = \{s2, s3, s5, s7\} \tag{4.3}$$

4.2.4 概略形状モデルによる骨格形状モデルの変更

ここでは，概略形状の操作を骨格形状に伝播する方法について述べる．概略形状モデルに対する操作についてはすでに述べたが，まず複数のセグメントに対する操作（拡大・縮小，平行移動，回転など）については，その概略形状に含まれる骨格に対しても同じ操作を適用する．平行移動や回転移動の場合は，形状の位置と姿勢が変化するだけであり形状自体は変化しないので，骨格形状も変形することなく位置姿勢を概略形状と同じように変更すればよ

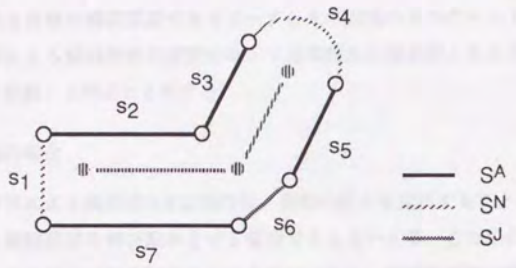


図 4.8: 骨格の勢力圏とセグメントの種類

い．また，拡大・縮小に関しても大きさが変化するだけなので，骨格形状も同じ比率で拡大・縮小すればよい．

一方，個々のセグメント単位で行われる操作は，骨格形状に反映させることが難しい．さらに，骨格形状は，セグメント単位の操作のような局所的な変形操作ではなく，複数のセグメントをまとめて操作することを目的としているので，セグメント単位での操作結果により骨格形状モデルを直接変更するのではなく，必要に応じて概略形状と骨格形状の対応づけを再度行うことにする．

4.2.5 骨格形状モデルによる概略形状モデルの変更

骨格形状を変形することにより概略形状を操作する手法について述べる．まずセグメント，頂点レベルで変形方法を説明する．実際には線分と円弧をセグメントとして概略形状を表現するが，線分と円弧とでは処理方法が多少異なり，特に円弧について特別に考慮しなければならない場合がある．そこで，説明が複雑になるのを避けるために以下では「セグメント＝線分」とし，円弧を含む場合の処理方法については後でまとめて述べる．また骨格形状についても，まず一番単純な 1 本のアークからなる場合について述べ，その後

で骨格がループを含まないグラフ構造となる一般の場合について説明する．一番単純な骨格の構成要素であるアークとその両端の２つのノードは，骨格形状操作による概略形状の変形において基本的な処理単位となるので，これを「単位骨格」と呼ぶことにする．

単位骨格の場合

単位骨格による典型的な形状操作は，骨格の長さを変更することによって，対応する概略形状を伸び縮みさせる変形であるといえる．この場合，骨格の軸方向にのみ拡大（縮小）するというだけでなく，形状の両端は変形しない点に特徴がある．つまり，操作対象となる形状の全てに対して同様の処理を行うのではなく，部分ごとに異なる処理を行わなければならない．このような処理の相違は，操作対象となっているセグメントが，骨格のノードとアークのどちらの影響を受けるかの違いとみなすことができる．すなわち骨格による形状操作は，「骨格形状のノードやアークの勢力圏を考え，ノードの勢力圏にあるセグメントは変形操作を適用せず，アークの勢力圏にあるセグメントのみを軸方向に変形する操作」とすることによって実現可能となる．

骨格形状モデルに対する操作としては，4.2.1で述べたように６種類の操作がある．ここではそのうち，ノードの移動操作とそれにともなう概略形状の変更方法を説明する．単位骨格におけるアークの移動操作は，対応する形状全体を移動すればよいので，ここでは変形方法の説明を省略する．また，それ以外のノードやアークの生成・消去によって概略形状が変更されることはない．

ノードの移動操作による骨格の変形は，もう一方のノードの位置が変更されないことを考慮すると，次の二つの骨格形状操作に分離できる．

1. アークの回転

2. アークの伸縮

この二つの操作は順不同で，どちらを先に行っても結果は等しい．説明の便宜上，アークの回転，伸縮の順に段階的に形状操作を行うとする．また，移

動させるノードを「移動ノード」，もう一方のノードを「固定ノード」，移動ノードが最終的に到達すべき位置を「目標位置」とそれぞれ呼ぶことにする．

ノードを移動することによって，一般には骨格の方向も変化する．アークの回転は，まずその方向のみを変更する操作である．この操作は，移動ノードと目標位置が同一直線上に並ぶように，固定ノードを中心としてアークを回転させる．これにともなって，S^N と S^A の両方のセグメント，すなわちその骨格の勢力圏に含まれるすべてのセグメントに対して，固定ノードを中心とし，アークの回転角とおなじだけ回転させる回転操作を行う．

次に，移動ノードと同一直線上に並んだ目標位置とが一致するように，アークの長さを変更する．この操作においては，S^N と S^A で行うべき処理が異なる．まず S^N のうち，固定ノードの勢力圏にあるセグメントはその位置を変えないので，さらに操作を加える必要がない．S^N の移動ノードの勢力圏内にあるセグメントは，移動ノードとの相対的な位置関係を保持するように，形状自体は変更せず平行移動させる．すなわち，アーク回転後の移動ノードを目標位置に一致させる方向に，移動量として移動ノードと目標位置間の距離だけセグメントを平行移動させる．アークの勢力圏に含まれる S^A は，各セグメントのアーク方向の長さの比が常に一定であるような変形処理を行う．これにより，S^A はアーク方向に関して弾性的に変形することになる．このような変形をアークの方向を軸とする「一軸変形」と呼ぶことにする．

単位骨格による形状変形例を図 4.9に示す．固定ノード n1，移動ノード n2，目標位置 p1 である．この例では，S^N，S^A に含まれるセグメントはそれぞれ (4.4) 式，(4.5) 式のようになる．

$$S^N = \{s1, s2, s3, s8\} \tag{4.4}$$

$$S^A = \{s4, s5, s6, s7, s9\} \tag{4.5}$$

図 4.9の (a) を初期状態とし，(b) のような形状を得ることを目的として変形する．(c) は，固定ノード n1 を中心とした骨格形状の回転，(d) ではさらに骨格の長さを伸ばし，その結果として $s8$ は，骨格軸方向への平行移動により

変形後の位置が決定されている．またS^Aは，各セグメントの軸方向長さの比が常に一定となるように，アーク方向へ長さが伸びている．

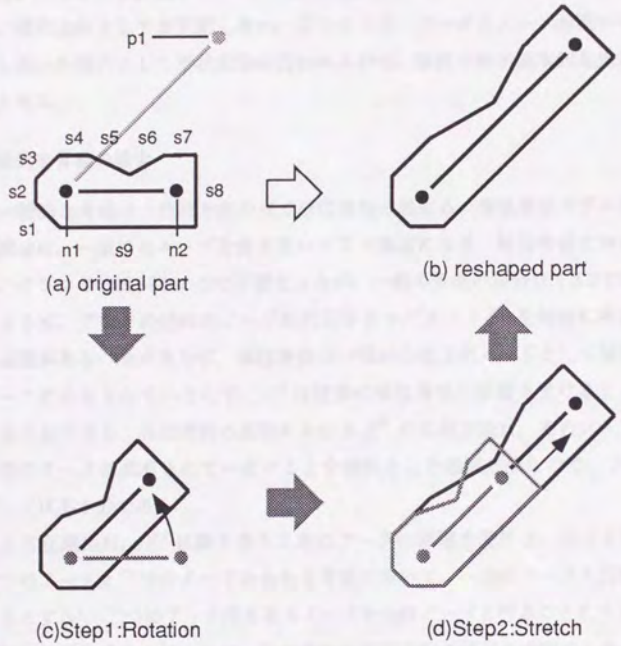


図 4.9: 単位骨格による形状変更

単位骨格は，骨格形状操作による概略形状の変形において基本的な処理単位となると述べたが，これは，S^Nの形状操作を，ノードに関する操作として定義することができないことによる．骨格形状の観点からは，ノードは位置情報を持てば十分であり，本来「ノードの方向」のような情報は定義できない．したがって上記のように，S^Nに対して回転操作を行う場合，この操作に必要な回転角をノードの持つ情報からは得られない．この場合，回転角

はノードに接続しているアークの方向から得られるので，アークに依存した操作であるといえる．一方 S^A の形状操作はアークのみに依存しているといえるが，かならず関連するノードの勢力圏にあるセグメントも操作しなければ，操作全体としては完結しない．このように，アークとノードが互いに依存しあった操作として形状変形が行われるので，単位骨格が基本的な処理単位となる．

一般的な骨格の場合

一般的な骨格は，分岐を含む点で単位骨格と異なる．骨格形状モデルの位相構造は，一般にはループを含まないグラフ構造になる．単位骨格においては，グラフの分岐がないので不要だったが，一般の骨格の場合は4.2.3で述べたように，グラフの分岐のノードに対応するセグメント S^J を特別に考慮する必要がある．なぜならば，単位骨格の一端が分岐点のノードとして複数のアークに共有されているので，S^J は複数の単位骨格の影響を受けることになるからである．単位骨格の変形における S^N の処理方法は，そのノードが複数のアークに共有されていないことを前提とした処理方法なので，S^J に対しては不十分である．

より正確には，S^J は隣り合う2本のアークの影響を受ける．たとえば，二つのアークと三つのノードからなる骨格において，一方のアークを回転させるとする[1]．二つのアーク間にあるノードを分岐ノードと呼ぶことにすると，回転した単位骨格の側からは，単位骨格の変形方法で述べた手順にしたがって，分岐ノードの勢力圏にあるセグメント S^J もその分岐ノードを中心に回転させることになる．しかし，分岐ノードはもう一つの単位骨格にも含まれているので，その移動していない単位骨格の側から見ると，S^J を回転させる必要がない．このように，分岐ノードの勢力圏に含まれるセグメントは，一方の単位骨格側からは回転することが要請され，もう一方の単位骨格に関してはその位置が変化するとかえって不都合を生じる．ここに，S^J を S^N の

[1]単位骨格による変形方法で示したように，S^N の変形はアークの長さ変化には依存しないので，アークの回転のみを考えればよい．

ように単純に処理できない理由がある.

以上のことから，隣り合う2本のアークによって挟まれる領域に含まれる S^J は，その2本のアークのなす角度に影響を受けると考えられる．そこで，S^A がアークの伸縮に対応して弾性的に変形したのと同様に，S^J を回転方向に対して弾性的に変形させる．すなわち，S^A における軸方向長さに対応するものとして，セグメントの両端点とノードを結ぶ直線のなす角度を考え，その角度の比が一定となるように変形する.

図4.10に，S^J の変形例を示す．ただし，概略形状，骨格形状とも一部分だけを示してある．この図において，S^J と S^A はそれぞれ(4.6)式，(4.7)式となる.

$$S^J = \{s2, s3\} \tag{4.6}$$

$$S^A = \{s1, s4\} \tag{4.7}$$

はじめ(a)の状態にあった骨格において，n_1 を固定し n_2 を回転させた場合の変形例を(b)に示す．図のように，変形前の *s2,s3* をみこむ角度をそれぞれ α,β とし，変形後の角度を α', β' とすると，変形前後で(4.8)式が成り立つようにセグメント *s2,s3* を変形する.

実際には，α,β などの角度の比だけではセグメントの変形方法は一意に決まらない．そこで，各セグメントの両端点と骨格のノードの距離は変化しないものとしてセグメントを変形する．図では右端の頂点とノードとの距離についてのみ l_1 として図示してあるが，他の頂点についても同様に，ノードとの距離は変形前後で同じである.

$$\alpha : \beta = \alpha' : \beta' \tag{4.8}$$

上記のように，単位骨格どうしの接続部分の形状変更方法において，対象となるセグメント数は任意である．しかし実際には，特に設計者が指定しない限り，S^J は一つの接続領域当たりに1セグメントしか含まれないものとする．ここで接続領域とは，となりあう二つのアークによって分割された，

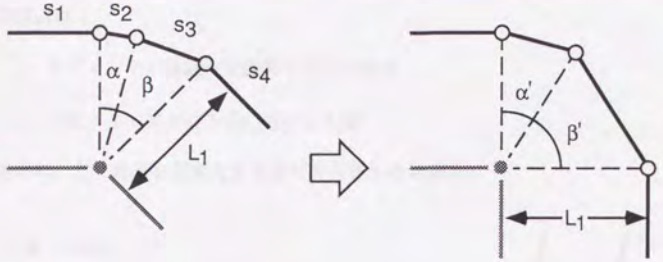


図 4.10: 分岐ノードの勢力圏にあるセグメントの変形例

分岐ノードまわりの領域を指す．すなわち，ある分岐ノードに接続しているアークの本数が k 本の時，その分岐ノードに関する接続領域の個数は k 個である．このように接続領域における S^J の本数を制限するのは，「概略形状レベルで接続部分を詳細に決定することはなく，連続したセグメント列として形状が表現されていることを保証すればよい」ということを仮定しているためである．本研究では，「応力集中を避けるためになめらかにつなぐ」などといった条件を扱うことは目的としていないので，接続部分を必要以上に複雑化することにはあまり意味がない．さらにこのような制限を加えることで，S^N との処理の相違がほとんどなくなり，骨格形状による形状変形処理の単純化という利点もある．

接続部分の処理の単純化という点に関して，S^J として，長さがゼロのセグメントを挿入して扱う場合がある．図 4.11 に示すような概略形状が与えられている場合，S^J に相当するセグメントが存在しない．これまでの説明では S^J が存在していた部分で，異なるアークの勢力圏にあるセグメントどうしが直接連結している．しかし，直接連結しているとするのではなく，長さがゼロのために表示されていない S^J が存在しているとみなし，実際に長さゼロのセグメントを挿入して変形処理を行うという方法をとる．図 4.11では，s0 が挿入されたセグメントである．「S^J は存在しない」としたままで処理を

考えることも可能だが，このようにセグメントを強制的に挿入する方法をとる理由は，

- セグメントの連続性を保証するのが容易
- 分岐ノードにおける場合分けが不要

などの，主に処理を単純化する点で有利なためである．

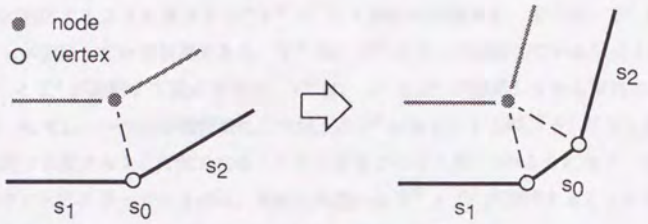


図 4.11: S^J の強制挿入が必要な形状例

この変形方法の問題点としては，概略形状の自己干渉があげられる．これを避けるためには，干渉部分での交点を計算し干渉部分を取り除く方法が考えられる．しかし本研究では，形状を操作している過程ではそのような処理は行わず，ある程度形状が固定された段階で行うものとする．なぜなら，自己干渉部分を取り除くと，それによって消去されてしまうセグメントが生じ，逆の操作を行った時にもとの形状にもどらなくなるのを避けるためである．

骨格形状操作のアルゴリズム

これまで述べてきたように，骨格形状モデルによる概略形状の変形操作（これを骨格形状操作と呼ぶことにする）は，骨格のアークやノードの勢力圏に含まれるセグメントに対する操作として説明してきた．しかし実際には，セグメントに対してさまざまな処理を行うのではなく，頂点の位置を求める処理によって骨格形状操作を実現する．セグメント単位で処理を行うと，とな

りあうセグメント間で共有する頂点を重複して扱う可能性があり，セグメントの連続性を保証するために，適切に処理されているかどうかを検証する必要がある．しかし，頂点ごとに処理を行えば，そのような処理の重複の問題が回避される．さらに，S^A を軸方向に弾性的に変形する処理においても，「セグメントを変形する」と考えるのではなく，「頂点の位置を変更する」としたほうが扱いやすいためである．

そこで，セグメントを S^A,S^N,S^J と分類したのと同様に，セグメントの種類に対応するように頂点を V^A,V^N,V^J の3種類に分類する．V^A は，S^A どうしが連結している頂点である．V^N は，S^N どうしが連結している頂点と，S^N と S^A が連結する頂点を含む．V^J は，S^J と S^A が連結している頂点である．ただし，一つの接続領域に二つ以上の S^J があるとする時，S^J どうしが連結する頂点も V^J に含まれる．これらをまとめると表4.2のようになる．表の中で×印になっているのは，骨格の構造から S^N と S^J が連続することはないためである．S^N と S^J の間にはかならず S^A が存在する．

表 4.2: 頂点の分類

セグメントの種類	S^A	S^N	S^J
S^A	V^A	V^N	V^J
S^N	V^N	V^N	×
S^J	V^J	×	(V^J)

つぎに，各頂点を骨格に写像するベクトルと，写像された骨格上の点を定義する．V^N と V^J は対応するノードに写像する．この時，そのノードを始点とし，V^N,V^J をそれぞれ終点とするベクトルを $\vec{m^N}$ とする．V^A は，対応するアークに対して正射影し，射影された点を P^M とする．また P^M を始点とし，V^A を終点とするベクトルを $\vec{m^A}$ とする．図4.12に例を示す．

骨格形状操作による，頂点位置の導出アルゴリズムを以下に示す．

1. 移動した単位骨格の回転角を求める．これを α とする．

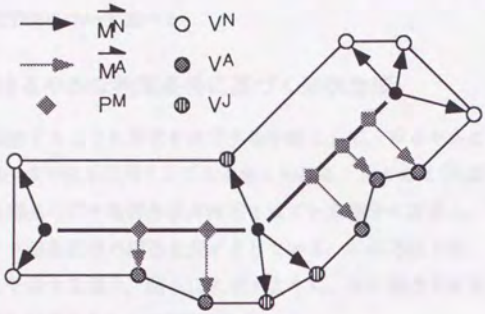


図 4.12: 概略形状の骨格に対する写像

2. その単位骨格に含まれる $\vec{m^N}$,$\vec{m^A}$ をすべて α だけ回転する.

3. あらたな単位骨格の長さを L とする.

4. その単位骨格に含まれる p_i^M について，固定ノードからの距離が $k_i \times L$ となるように骨格上の位置を変更する.

5. $\vec{m^N}$ の始点を，それぞれ対応する固定ノード，移動ノードとして，あらたな V^N,V^J を求める.

6. $\vec{m^A}$ の始点を，対応する P^M としてあらたな V^A を求める.

4.3 非干渉条件に基づく形状変形

本研究ではさらに骨格形状モデルによる概略形状モデルの操作手法を利用して，機構部品の非干渉形状を生成する．これにより，機構部品の形状設計において考慮すべき条件として先に示した，部品間の非干渉条件を満足するような部品形状を決定することができる．以下では，まずこのような非干渉形状を生成する手法として応用できる，ゆるやかな拘束条件に基づく形状生

成手法について述べ，その問題点を示す．続いて，本研究で提案する非干渉形状の生成手法について述べる．

4.3.1 ゆるやかな拘束条件に基づく形状生成

干渉を回避するように形状を決定する手法として，ゆるやかな拘束条件に基づく形状生成手法を応用することが考えられる．たとえば [佐藤 87] や [鈴木 88] では，板金部品の穴や輪郭形状が指定されている部分に着目し，それらを結ぶ直線により部品形状の構造を表すとしている．この方法では，ある部品が他の部品と干渉する場合，図 4.13に示すように，その構造を変更することによって干渉を回避することが可能である．

しかし，部品形状の構造は板金部品の穴の位置を利用して決定しており，また穴どうしがかならず直線で接続されるため，構造の変更方法は板金部品中の穴の位置に依存してしまう．たとえばその部品に穴が二つしか存在しない場合は，構造を変更することができないという問題が生じる．つまり，「穴」といった特徴的な形状が部品形状にあまり含まれていない場合，構造の変更方法が限定されてしまう点で問題がある．また，干渉を回避するために穴の位置を変更することは通常行なわれないと考えられるので，部品形状の構造が穴の位置を基準として決定されるこの手法では，形状変更の柔軟性に難点があるといえる．

また [Shimada 92] では，部品形状中の穴と設計者によって指定されている輪郭形状，さらにそれらを結ぶ直線のまわりにポテンシャル場を作り，図 4.14に示すような外形に対して定義されているエネルギを最小とするなめらかな輪郭形状を生成する．この手法の場合，障害物領域のまわりにポテンシャル場をつくれば，その障害物を回避する形状を生成することになる．

しかし [Shimada 92] では，部品形状中の穴やそれらを結ぶ直線によって作られるポテンシャル場の形状そのものを変更することについては述べていない．したがって，部品形状と障害物との干渉領域が大きく，ポテンシャル場を生成する際に用いる直線まで障害物と干渉している場合，障害物を回避する輪郭形状を求めることは困難である．

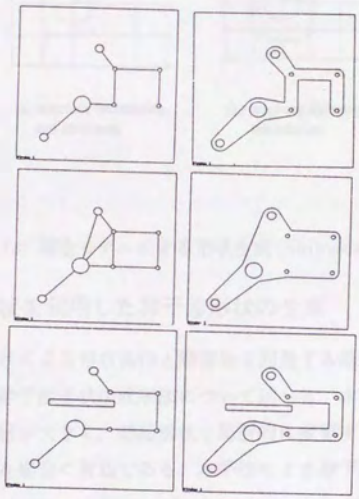
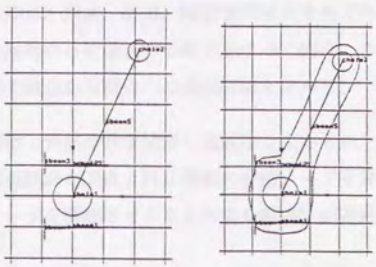


図 4.13: 構造モデルによる形状生成 [鈴木 88]



(a) interface boundaries and skeletons　　(b) resulting skinning boundaries

図 4.14: 構造モデルによる形状生成 [Shimada 92]

4.3.2　骨格形状を利用した非干渉形状の生成

ここでは骨格形状による形状操作と障害物を回避する経路生成手法を応用した，機構部品の非干渉形状生成手法について述べる．本手法は，機構部品と障害物の干渉領域が大きく，部品形状を局所的に変更するだけでは干渉を回避できないような場合に有効である．本手法による非干渉形状の生成手順を次に示す．

1. 障害物領域のオフセット形状算出：本手法では障害物を回避する経路生成手法として，膨張・収縮法を用いる [杉原 87]．したがって，まず障害物領域のオフセット形状を求める．オフセット量は設計者があたえたクリアランスと変形対象である機構部品の骨格形状と概略形状間の距離を加えたものとする．障害物回避経路の生成手法としては膨張・収縮法以外にも提案されており，本手法でこの方法を用いる理由については後で述べる．

2. 障害物回避経路の算出：変形対象である骨格形状の両端点をそれぞれ始点と終点とし，障害物領域を回避して始点から終点に至る最短経路を求

める．そのために，始点，終点，障害物領域のそれぞれと端点で接し，障害物領域の外部のみを通過する線分をすべて描く．さらにそれらが作るグラフの上で始点から終点への最短経路を求める．

3. 骨格形状の変形：骨格形状を変形し前段階で求められた経路に一致させる．障害物回避経路の節点と同じ個数の骨格ノードを骨格形状上に生成し，各骨格ノードを移動させることにより障害物回避経路と骨格形状を一致させる．

4. 概略形状の変形：4.2.5で述べた方法により，骨格形状の変更結果を概略形状に展開する．

本手法では，骨格形状による概略形状の変形手法を応用していることから，概略形状の大域的な変形が可能である．その結果，干渉領域が大きく部品形状を局所的に変更するだけでは干渉を回避できないような場合についても対応することができる．

膨張・収縮法を用いる理由

障害物を回避する経路の生成手法としては，膨張・収縮法以外にもポテンシャル場を利用する方法がある．たとえば，[Khosla 88]では，障害物回避経路を探索する際にLocal Minimumに陥らないようにするため，ポテンシャルの形状をsuperquadircで表現する方法を提案している．また[Barraquand 91]では，パスを生成する際にLocal Minimumから脱出する手段を与える方法を提案している．さらにポテンシャル場において移動経路を最適化することは通常計算コストが大きいので，[Hwang 88]では，アルゴリズムが簡単で計算量が少なくてすむアルゴリズムを考案している．

こうしたポテンシャル場を利用した方法によって求められる障害物回避経路は，一般に曲線の経路となる．しかし，本研究で提案する非干渉形状の生成手法では，概略形状の変形操作法として骨格形状による方法を用いるので，骨格形状が点と直線によって表現されることから，干渉を回避する骨格形状として曲線経路を直接利用することができない．つまり，本手法では障害物

回避経路として直線経路を導出する方法が適している．そこでポテンシャルによる障害物回避経路の生成手法は利用しない．また [Jacobs 89] で提案されている手法も同様に，導出する経路を円弧と直線からなる経路としているので利用できない．

第5章

機構モデリングシステムの試作

5.1 システム構成

これまで述べてきたモデリング手法に基づいた，機構のモデリングシステムを試作した．試作システムは Common Lisp と対象指向言語 Flavor， CLOS を用いて， Lisp マシン Mac-Ivory 上で開発されている [Keene 89]．図 5.1に試作システムの構成図を示す．また図 5.2にシステムの初期画面を示す．以下では，試作システムを構成する各ユニットについて説明する．

5.1.1 2次元スケッチャ

骨格形状モデルや概略形状モデルの入力インタフェースとして，2次元スケッチャ（作画ツール）を用意している．対話的な機構のモデリング環境を提供することが本研究の目的の一つなので，多くの操作が2次元スケッチャをインタフェースとして実行される．また，寸法を明示的に扱わないという点でも，画面上に表示された概略形状や骨格形状を見ながら操作することが必要となる．すでに述べたように，ここで描かれる図形のことを，寸法を明示的に扱わないという意味で「スケッチ」と呼ぶ．したがって，フリーハンドの入力を意味するわけではない．

2次元スケッチャの具体的な機能としては，マウスによる線分，円弧の入力と図形編集機能が基本的なものである．図 5.2のシステムの初期画面に示し

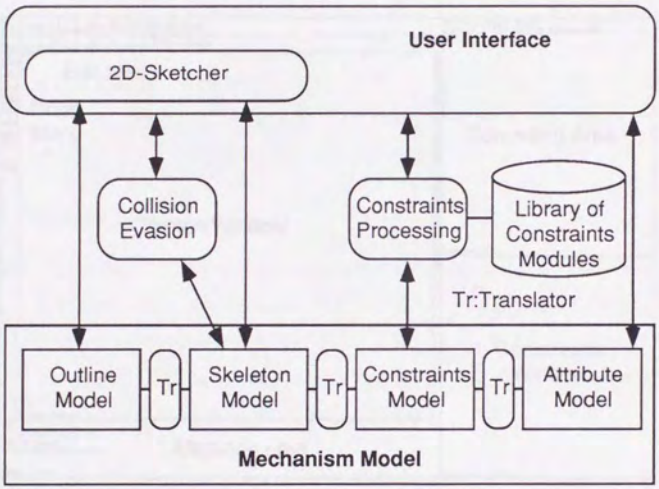


図 5.1: 試作システムの構成図

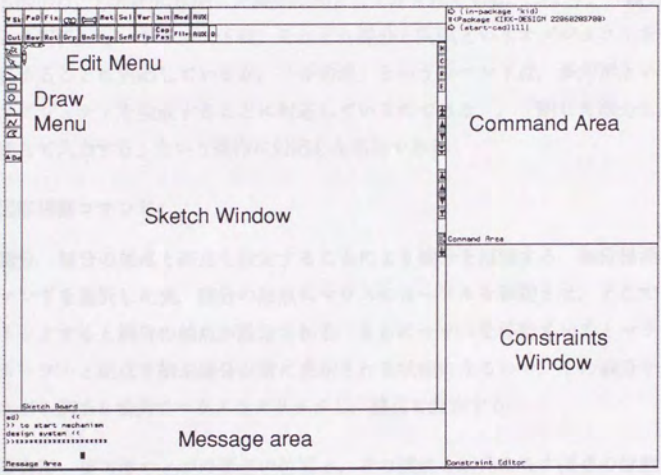


図 5.2: システムの初期画面

たように，機構の図は中央の画面上に描かれる．作画のためのコマンド群は，アイコン表示されたボタンとして左端に並んでいる．これらのコマンドを用いて，線分／長方形／円／円弧／折れ線／多角形などを作図することができる．

ただし本システムでは，セグメント（線分と円弧）のみを図形の基本要素としている．したがって，それ以外の，たとえば「多角形」などは基本要素として扱われているのではない．「多角形」というコマンドにより生成される図形は，内部的には閉じた線分の列として表されている．つまり，「線分」と「円弧」というコマンドは，それぞれ線分と円弧というオブジェクトを生成することに対応しているが，「多角形」というコマンドは，多角形というオブジェクトを生成することに対応しているのではなく，「閉じた線分を連続して入力する」という操作に対応した名称である．

図形描画コマンド

線分　線分の始点と終点を設定することにより線分を描画する．線分描画コマンドを選択した後，線分の始点にマウスのカーソルを移動させ，そこでクリックすると線分の始点が設定される．さらにマウスを移動させるとマウスカーソルと始点を結ぶ線分が常に表示される状態になるので，その線分を見ながら適当な場所でマウスをクリックし，終点を設定する．

長方形　長方形の一つの頂点の位置と，その頂点と対角をなす頂点の位置を指定することにより，長方形を描画する．長方形描画コマンドを選択した後，マウスのカーソルを適当な場所に移動させ，クリックすることにより長方形の頂点位置（この頂点を「頂点１」とする）を設定する．さらにカーソルを移動させると，カーソル位置が頂点１と対角をなす頂点になるような長方形を，常に表示するようになるので，それを見ながら適当な場所でマウスをクリックし，もう一つの頂点位置を決定する．

これにより生成されるオブジェクトは，「長方形」という種類のオブジェクトではなく，「線分のループ」というオブジェクトになる．したがって，

システム内部では，多角形描画コマンドによって生成されるオブジェクトと同一のものである．ただし，このコマンドでは，かならず長方形を構成するように生成される線分のパラメータを決定する．

折れ線　頂点の位置を連続して指定することにより，それらの頂点を順番に接続する折れ線を描画する．折れ線描画コマンドを選択した後，マウスカーソルを適当な場所に移動させ，クリックすることにより折れ線の各頂点位置を設定していく．頂点を設定していく方法は線分描画コマンドと同様で，頂点を設定すると，その頂点とカーソルをむすぶ線分が常に表示される状態になるので，その線分を見ながら適当な次の頂点位置を設定する．

設定された頂点が順番に $\{v_1, v_2, \ldots, v_n,\}$ のとき，生成される折れ線は，(5.1)式で定義される．ここで $s(v_i, v_{i+1})$ は，頂点 v_i と v_{i+1} をむすぶセグメントをあらわす．また，このように頂点を共有した一連の線分を「線分列」と呼ぶ．

$$s_i = s(v_i, v_{i+1}) \tag{5.1}$$

このコマンドと線分描画コマンドとの相違は，線分を連続的に生成するということだけではない．生成されるオブジェクトの相違が重要である．線分描画コマンドで生成された線分の端点は，線分ごとに必ず異なるが，折れ線描画コマンドで生成された一連の線分群は，隣り合う線分どうしで頂点を共有している．つまり，どちらのコマンドを利用しても折れ線を描画することは可能だが，線分描画コマンドで折れ線を作図した場合は，各線分はそれぞれ独立しており，位置的に連続しているように作図されているだけである．

多角形　頂点の位置を連続して指定することにより，それらの頂点を順番に接続する多角形を描画する．多角形描画コマンドを選択した後，マウスカーソルを適当な場所に移動させ，クリックすることにより多角形の各頂点位置を設定していく．作図方法は，折れ線描画コマンドと同様である．ただし，頂点を二つ以上設定した後では，直前に設定した頂点と，一番最初に設定した頂点の両方とカーソルをむすぶ線分が常に表示される状態になる点が折れ線描画コマンドと異なる．

また生成される線分列も，設定された頂点が順番に $\{v_1, v_2, \ldots, v_n,\}$ のとき，(5.2),(5.3) 式で定義される．折れ線描画コマンドで生成されるオブジェクトに対して，(5.3) 式が加えられていることから，多角形描画コマンドではかならず閉じた線分列を生成することになる．

$$s_i = s(v_i, v_{i+1}) \tag{5.2}$$

$$s_n = s(v_n, v_1) \tag{5.3}$$

円　中心位置と半径を指定することにより円を描画する．円描画コマンドを選択した後，適当な場所にカーソルを移動させ，クリックすることによって円の中心位置を設定する．その後カーソルを移動させると，線分描画コマンドを実行した時のように，円の中心位置とカーソルを結ぶ直線が常に表示されるとともに，その直線を半径とする円も同時に表示される状態になる．これらの半径や円を見ながら適当な場所でマウスをクリックすることにより，円の半径を決定する．

これにより生成されるオブジェクトは，「円」という種類のオブジェクトではなく，「円弧」オブジェクトの始点と終点が一致している特別な状態として生成される．また始点と終点の一致を単に位置の一致としているだけでなく，始点と終点が一つの点で表されているという意味でも一致しているので，円は，もっとも単純な閉じたセグメントであるといえる．

円弧　中心位置，円弧の半径，始点，終点を設定することにより円弧を描画する．円弧描画コマンドを選択した後，適当な場所にカーソルを移動させ，クリックすることによってまず円弧の中心位置を指定する．次にカーソルを移動させると，円描画コマンドの実行時と同様に，カーソルに合せて円と半径が常に表示される状態になるので，適当な場所でマウスをクリックすることにより，円弧の半径と始点を設定する．その後カーソルと移動させると，カーソルと円弧の中心をむすぶ直線にのる半径が常に表示される状態になるので，その半径を見ながら適当な場所でクリックすることにより，円弧の終点を設定する．

図形編集コマンド

また，画面上部のメニューバーの下段に，図形の消去／複写／平行移動／回転／変形などの，図形編集コマンドが表示されている．これらの編集操作は，セグメント単位で行うことが可能なだけではなく，セグメントをグループ化したオブジェクトに対しても有効である．たとえば，ある部品の概略形状が複数の連続していないセグメントによって作図されている場合，それらのセグメントを「グループ」として指定しておけば，グループに含まれるセグメントを同時に平行移動したり，変形することができる．

本来，各部品の概略形状は，連続したセグメントのループとして表現される．しかし，スケッチの段階では，それらのセグメントの連続性をチェックして一連のセグメント列とすることは，必ずしも必要ない．このような場合，直接連結していない複数のセグメントをグループ化して扱うと便利である．ただし，グループ化したオブジェクトに対する変形操作は，グループに含まれるすべてのセグメントに対する，縦横比を変えない単純な拡大・縮小操作のみ可能である．

選択　画面上に表示されている図形オブジェクトを選択する．２次元スケッチャは，普通つねに選択コマンドを実行できる状態になっている．マウスカーソルを図形オブジェクトの上に移動させ，そこでクリックすることによりオブジェクトを選択する．選択コマンドを実行すると，そのオブジェクトを内包する長方形領域の各頂点が表示され，ユーザに対してオブジェクトが選択されている状態であることを示す．ただし，線分を選択した場合に表示されるのはその両端点であり，長方形の領域ではない．

消去　指定された図形オブジェクトを消去する．はじめに選択コマンドによって図形オブジェクトを選択する．そのあとで消去コマンドを実行すると，選択されていた図形が画面上から消去される．

消去コマンド実行直後ならば，消去された図形オブジェクトは一時的に保存されている．その結果，消去コマンドを実行した直後に貼込みコマンドを

実行すると，ふたたび消去された図形が表示される．ただし，消去された図形は一時的に保存されているだけなので，あらたに消去コマンドや複製コマンドを実行すると，それ以前に保存されていた図形は完全に廃棄されてしまう．

複製　指定された図形オブジェクトの複製を生成し，一時的に保存する．はじめに選択コマンドによって図形を選択した後，複製コマンドを実行すると，選択されていた図形の複製を自動的に生成し，その図形オブジェクトを一時的に保存する．

このコマンドは，指定された図形の複製を生成するだけなので，複製コマンドを実行してもあたらしく図形が表示されるわけではない．複製した図形を表示するためには，「貼込み」コマンドを用いる．また，指定された図形を一時的に保存する点では，消去コマンドと類似したコマンドといえる．しかし，消去コマンドを実行した場合には，消去対象となった図形オブジェクトそのものを一時的に保存するが，複製コマンドを実行した場合には，まず指定された図形の複製を生成し，その複製を保存する点に相違がある．したがって，複製コマンドを実行しても，複製対象となった図形はそれ以前とまったく変化がなく，画面上に表示されたままである．

貼込み　一時的に保存されている図形オブジェクトを再表示する．消去コマンドや複製コマンドによって，図形を一時的に保存した後，貼込みコマンドを実行すると，その時点で保存されている図形をふたたび表示する．

平行移動　指定された図形を動的に平行移動させる．まず選択コマンドで移動させる図形を選んだ後，平行移動コマンドを実行し，適当な場所にマウスカーソルを移動させてクリックする．この時，必ずしも選択されている図形の上でクリックする必要はない．次にカーソルを移動させると，移動対象となっている図形もカーソルの移動にともなって平行移動する状態になる．この状態で，カーソルとともに移動している図形を適当な位置まで移動させ，そこでもう一度クリックすることにより平行移動コマンドを終了させる．

最初にクリックした地点を p_1 とし，次にクリックした地点を p_2 とすると，結果的に，指定された図形を $\overrightarrow{p_1p_2}$ だけ平行移動させたことになる．ここで，p_1 と p_2 が与えられた後に平行移動を実行するのではなく，カーソルとの相対的な位置関係を常に保ったまま図形を移動させていくことから，「動的な平行移動」と呼ぶ．

回転移動　指定された図形を動的に回転移動させる．まず選択コマンドにより回転移動させる図形を指定する．その結果，前述したように，その図形のまわりには選択されていることを示す点が表示される．次に回転移動コマンドを選択したのち，図形のまわりに表示されている点のいずれかをクリックすると，その点と対角をなす点が固定され，カーソルの移動にともなって固定点のまわりに回転できる状態になる．この状態で，カーソルとともに回転する図形を適当な姿勢になるまで回転移動させ，ふたたびクリックすることにより回転移動コマンドを終了させる．

最初にクリックした点を p_1，最後にクリックした地点を p_2，選択された図形の長方形領域において p_1 と対角をなす点を p_0 とする．p_0 を中心として p_0,p_1 をむすぶ直線と，p_0,p_2 をむすぶ直線がなす角度を α とすると，指定された図形を p_0 のまわりに α だけ回転移動させたことになる．こうした回転中心や回転角を与えてから，その後に図形を回転移動させるのではなく，回転移動コマンドの場合も平行移動コマンドと同様に，カーソルの動きにしたがって図形を回転移動させていくことから，このような回転移動の方法を「動的な回転移動」と呼ぶ．

拡大・縮小　指定された図形の縦横比を変えずに拡大（縮小）する．まず選択コマンドにより拡大する図形を指定する．この時点で選択された図形のまわりには，その図形を内包する長方形の四つの頂点が表示されている．次に拡大コマンドを選択し，四つの頂点のうちいずれかをマウスでクリックする．その後カーソルを移動させると，図形を内包する領域の長方形が，カーソルの動きに合せて動的に拡大（縮小）される．

この時，最初にクリックした頂点と対角をなす頂点は固定点となり，表示

される長方形の縦横比は，はじめに図形が選択された時の領域の縦横比と同じである．このようにして動的に拡大される領域を見ながら適当な大きさにし，もう一度マウスをクリックすることによって，拡大コマンドを終了する．

ただし，線分に対してはこの操作は行えない．なぜならば，線分の場合には「縦横比」を定義することができないからである．線分の大きさを変更するためには，変形操作を用いればよいので問題はない．

変形　指定された図形の頂点を移動することにより図形を変形する．変形する図形を選択したのち，変形コマンドを実行すると，操作可能な頂点が表示される．次にそれらの頂点の中から移動させる頂点までカーソルを移動させクリックすると，マウスによってその頂点を移動させることができる状態になる．

線分，折れ線，多角形が変形の対象となっている場合は，カーソルを動かすことによって頂点を任意の場所に移動させることができる．一方，円弧が変形の対象となっている場合は，その円弧を一部分とする円周上での移動だけが許される．すなわち，円弧の中心位置や半径の大きさを変更するような変形は行えず，円弧の始点と終点の位置を変更することができる．円弧の中心位置を移動する場合には「平行移動」コマンドを，また，半径の大きさを変更する場合には「拡大」コマンドを用いればよい．いずれの場合にも，適当な位置に頂点を移動した後，ふたたびマウスをクリックすることにより変形コマンドを終了する．

ただし，長方形描画コマンドによって描かれた図形に対しては，変形コマンドは適用できない．すでに述べたように，長方形描画コマンドで描かれた図形もシステム内部では線分列として表現されているので，多角形などと同様に変形操作を適用することは可能である．しかし，長方形描画コマンドで作図された図形は，編集操作を行ったあとでも長方形であるべきだと考え，変形コマンドの適用を不可とする．なぜなら，変形コマンドを実行した後では，四角形が長方形となるための条件である「すべての内角が９０度」という条件を，一般的には満足しないからである．

5.1.2 対偶制約ライブラリ

対偶モデルにおける制約条件のモジュール群は，ライブラリとしてあらかじめ用意されている．「リンク」「歯車」などの簡単な機構名や，「同軸接続」のような接続方法を指定することにより，対応する対偶制約条件式 C^M, C^C が生成される．

実際には，ライブラリに保存されている制約条件式はそのまま利用するのではなく，具体的な条件式を生成する際のテンプレートとして利用する．これは単に，同じモジュールを数箇所で使用する場合に，そのままでは条件式中のパラメータが干渉してしまうという問題があるためだけではない．属性モデルと対偶モデルを対応づけておくためには，属性モデルにおけるパラメータを用いて制約条件を記述しなければならないからである．

たとえば，歯車機構を表現する制約条件式を再掲すると (5.4) ～ (5.7) 式のようになるが，この式中の変数は，それぞれ対応する属性モデルのパラメータでもなければならない．つまり，この場合はそれぞれの歯車が，$\{r_1, \theta_1, x_1, y_1, \ldots\}$ や $\{r_2, \theta_2, x_2, y_2, \ldots\}$ などのパラメータで表現されていなければならない．したがって，属性モデルにおいて一方の歯車の中心座標が (Cx_1, Cy_1) のように表現されている場合，(5.7) 式中の (x_1, y_1) を (Cx_1, Cy_1) に置き換えた条件式である (5.8) 式を生成する必要がある．

$$r_1\theta_1 + r_2\theta_2 = 0 \tag{5.4}$$

$$Ra = r_1/r_2 \tag{5.5}$$

$$D = r_1 + r_2 \tag{5.6}$$

$$D^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2 \tag{5.7}$$

r_i : $Gear_i$ のピッチ円半径，θ_i : $Gear_i$ の回転角，(x_i, y_i) : $Gear_i$ の中心座標，Ra : ギア比，D : 軸間距離

$$D^2 = (x_2 - Cx_1)^2 + (y_2 - Cy_1)^2 \tag{5.8}$$

こうしたことから，対偶制約ライブラリには，制約条件式を生成するため

のテンプレートとともに，それらのテンプレートごとに，具体的なパラメータを埋め込んで制約条件式を生成する手続きも保存されている．

5.1.3 属性・対偶モデル操作ユニット

属性モデルと対偶モデルは，表現方法としてパラメータを互いに共有し，密接な関係があるので，システム内では分離せずに両方を合せて扱う．属性モデルと対偶モデルを操作する機能として以下のような機能があげられる．

- パラメータのデフォルト値／タイプ指定
- 設計者による制約条件式の定義

対偶制約の条件式を生成するためには，制約条件のモジュール名を指定すればよいが，条件式に含まれるパラメータに対するデフォルト値を設定するためには，スケッチャ上に描かれている概略形状モデルや骨格形状モデルと，生成する制約条件式との対応を指定する必要がある．すでに述べたように，制約条件を生成する時にスケッチャ上の図形を指定することによって，概略形状モデルや骨格形状モデルを反映したデフォルト値を設定することができる．また，パラメータのタイプは対偶制約を解く際に重要な意味をもつ．特に値を固定しておく必要のあるパラメータ C^V は，制約条件式を解く状況に応じて変更する場合がある．そのため，パラメータのタイプを指定する機能も実現されている．

機構対偶に関する条件式は対偶制約ライブラリに登録してあるが，それ以外の制約条件式を定義する手段を提供する．具体的にはキーボードから条件式を直接入力する．これによってユーザは，多項式で表現できる限り自由に条件式（C^U）を定義することができる．しかし，属性モデルの情報を直接操作せずに，機構のモデルを構成していくことができる点に本システムの特徴がある．そのために通常，属性モデルや対偶モデルの内容は設計者に対して隠蔽されている．したがって実際には，条件式を定義するために必要なパラメータを属性モデルから得ることが難しいので，設計者が制約条件式を直接定義することは容易ではない．

5.1.4 骨格形状モデル操作ユニット

骨格形状モデルによって表現される情報は，幾何情報と位相情報に分けられる．幾何情報の管理は，2D スケッチャの機能を利用することでほとんど実現できる．したがって主に位相情報の管理を行う．たとえば，4.2.1で述べたように，位相構造を変えてしまうようなノードの移動は禁止する．そのために，ノード移動の実行後，図4.6に示したような，位相構造の変化や骨格アークの自己干渉などを検証し，問題点が見つかれば，その操作前の状態に強制的に復帰させる．

5.1.5 概略形状モデル操作ユニット

概略形状モデルを操作するほとんどの機能は，2D スケッチャによって実現されている．ここでは，次のような機能が実現されている．

1. セグメントのループ抽出

2. セグメント種類の指定

セグメントのループ抽出が必要となるのは，骨格形状モデルと対応づける前に，部品の概略形状が，セグメントのループとして表現されていなければならないからである．なぜなら，骨格による概略形状の変形手法の前提条件として，となりあうセグメントが頂点を共有している必要がある．しかし，概略形状をスケッチしていく時には，必ずしも連続した線分で形状を入力していくとは限らず，とくに円弧を作図する場合は，かならず1セグメントごとの入力となる．

したがって，ある微小距離 δ として適当な値を決めておき，異なるセグメントに所属している頂点 v_1 と v_2 の距離を $dist(v_1, v_2)$ とする時，(5.9) 式が成立すれば二つの頂点 v_1, v_2 は同一の頂点であるとみなし，v_1, v_2 を一端とするセグメントどうしを連結する．

$$dist(v_1, v_2) \leq \delta \tag{5.9}$$

セグメント種類の決定方法についてはすでに述べたように，基本的にはシステムが自動的にセグメントの種類を判別する．しかし，システムが決定したセグメントの種類を，場合によっては設計者が変更することを考慮して，骨格ノードとセグメントの対応をシステムの画面上で指定する手段も提供する．たとえば，単位骨格の接続部分にあたる S^J の判別結果に関しては，長さがゼロのセグメントを導入するか，すでにあるセグメントを S^J とするかの修正が必要な場合がある．

5.1.6 骨格形状 / 概略形状モデル変換ユニット

5.1.7 対偶制約評価ユニット

すでに述べた手法にしたがって，自動的に制約条件を評価していく．評価結果により，部品間の接続関係を維持した形状変更や機構の動作シミュレーションが行える．

5.1.8 非干渉制約評価ユニット

与えられた障害物領域と，非干渉条件を検討する部品の機構モデルから，干渉を回避するような部品形状を決定する．

5.2 実行例

5.2.1 ストロボのポップアップ機構の設計例

まず始めにストロボのポップアップ機構の設計を例にして，試作システムを用いた設計の基本的な手順と，その特徴について述べる．

ここでの設計対象は，図5.3に示したような一眼レフカメラのペンタプリズム上部に内蔵されているストロボを，撮影の時に飛出させる機構である．この機構を設計する際に実際に用いられたスケッチとして図5.4が報告されている [写真工業 87]．このように，図面といった詳細なレベルではなく，寸法な

どについては十分な情報を扱わないレベルでの図において機構を検討することが行われている．

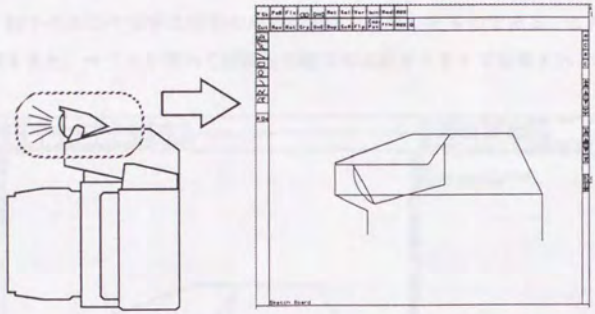


図 5.3: 一眼レフカメラのストロボ - ポップアップ機構

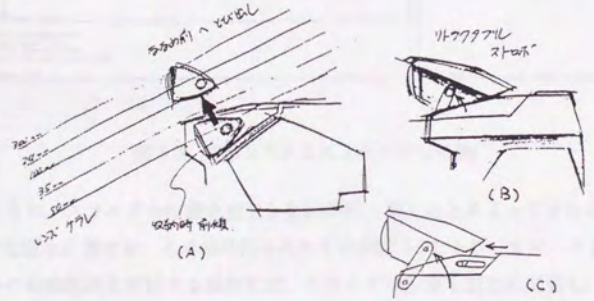


図 5.4: ストロボ - ポップアップ機構のスケッチ [写真工業 87]

そこで試作システムのスケッチャを用いて，まず図 5.5に示したようなスケッチを描く．すでに述べたように，一般の 2 次元作画ツールと同様な機能を用

いてこのような図を描く．次に，ポップアップ機構としてまずスライダ-リンク機構を採用し，実現の可能性を検討する．そこで，図5.6のようにスケッチ上にリンク(Link1, Link2)とスライダ(Slider)をそれぞれ定義する．ただし，図中の矢印や文字は説明のために後から書込んだものである．これらの機構もまた，マウスを用いて画面上で適当な位置や大きさで定義される．

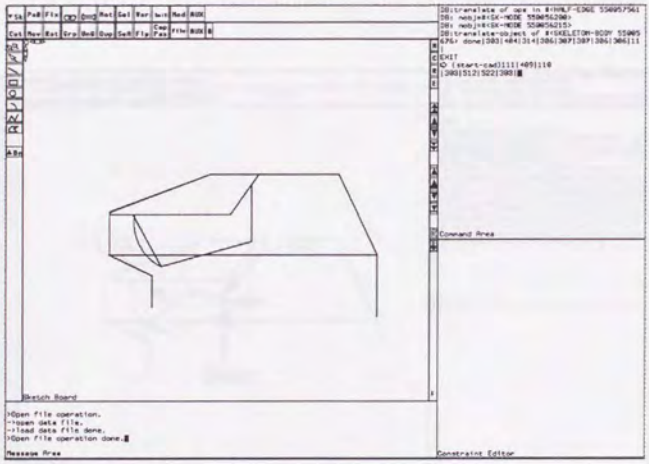


図 5.5: 試作システムによるスケッチ例

さらに，スライダの位置を変更し制約条件を解くことによってあらたな部品の位置を計算する．その結果得られたものが図5.7である．また，ストロボ部分の移動経路を検討する場合には，スライダの位置を適当に変更し，部品位置を算出する．数箇所について部品位置を計算し，その結果を重ねて表示したものを図5.8に示す．

図5.7からわかるように，この場合にはストロボ部分を十分突出させることができていない．そこで次に代替案として4節リンクを採用し，スケッチを変更する．図5.9はスライダ部分を消去し，新たにリンクを定義した状態を示

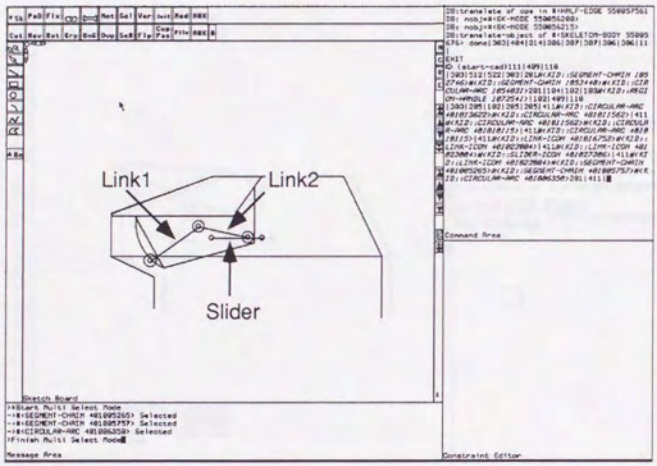


図 5.6: スライダ - リンク機構の定義例

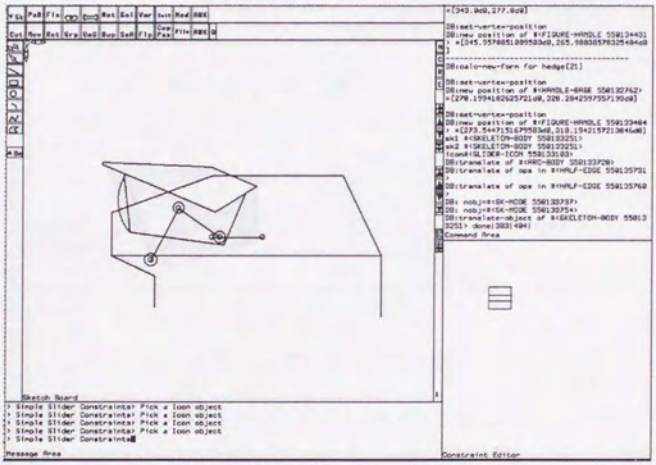


図 5.7: 制約条件に基づく機構部品の位置の算出例

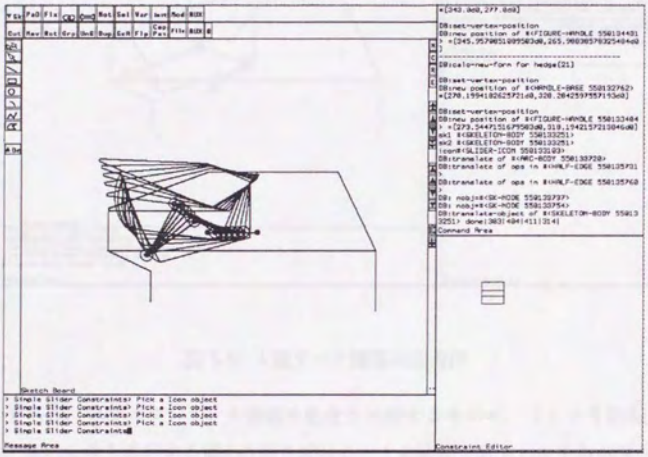


図 5.8: ストロボ部分の移動経路の表示

す．画面上には三つのリンクしか表示されていないが，これは固定節が表示されていないためである．

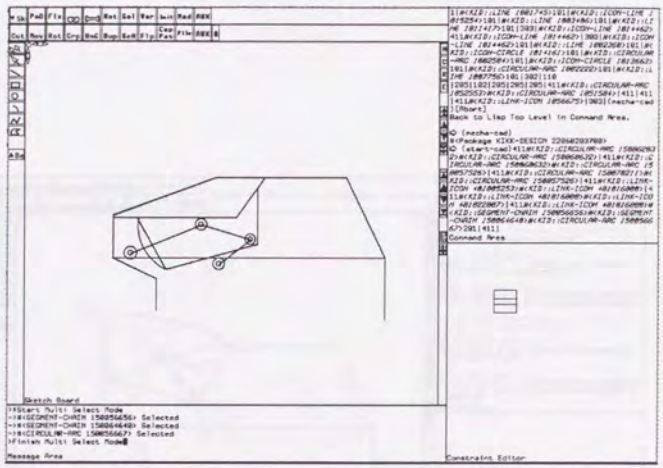


図 5.9: 4 節リンク機構の定義例

さらに，スライダ - リンク機構の場合と比較するために，リンクを回転させ，制約条件を満足する部品位置を求める．その結果を表示したものが図 5.10である．また，複数の適当な回転角について制約条件を解き，それらの結果をかさねて表示した例を図 5.11に示す．

これまで示してきた図 5.6〜図 5.11では，リンクやスライダをアイコンのみで表示してある．これは試作システムにおいて，スケッチャによって定義された概略形状レベルの情報と，対偶モデルのアイコンで示されるような制約条件レベルでの情報を共存させて用いることができることを示すためである．これに対し，リンクなどについても部品形状を与えた状態で用いることもできる．図 5.12に，リンクに対しても概略形状を定義した例を示す．さらにこの状態での部品の移動経路を図 5.13に示す．

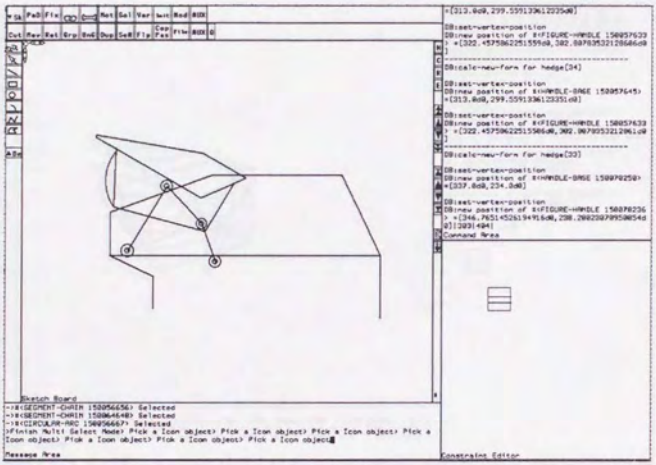


図 5.10: 制約条件に基づく機構部品位置の算出例 (2)

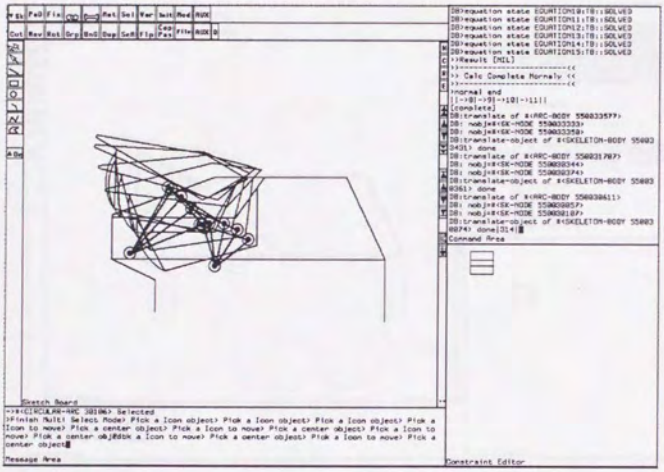


図 5.11: 4節リンク機構を用いた場合の移動経路表示

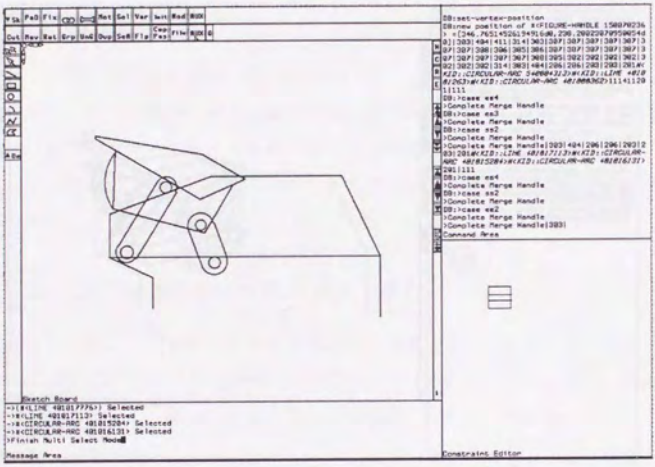


図 5.12: リンク形状の定義例

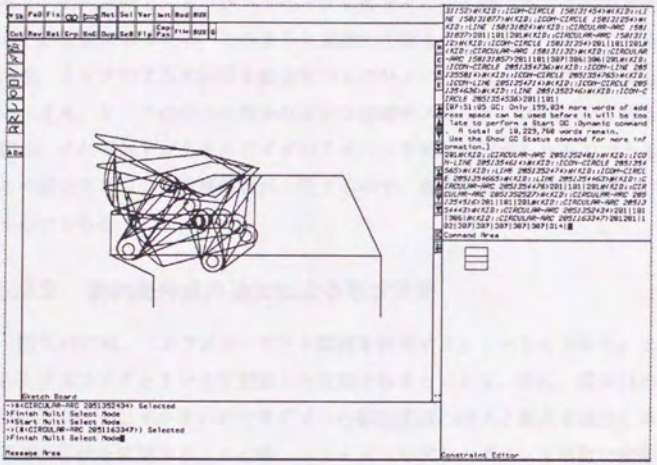


図 5.13: リンク形状を含む機構部品の移動経路表示

図3.3で示したVTRのテープ・ローディング機構の設計を例にとり，そのモデリング過程における部品の干渉回避形状の生成例を示す．まずリンクやスライダの概略形状をスケッチする．たとえば図5.14のようなスケッチを描く．リンクはそれぞれ円弧や線分を用いて描かれている．スライダに関しては，ここでは形状の描画手順をマクロ化し，ピンを表す円を描きその円を通るスライダの経路を指定すれば自動的に形状が生成される．スライダの場合はスケッチとしてのバリエーションがあまりないと考えられ，図5.14のようにピンを円で表し，ピンがスライドする溝をその円に接するように図を描くことが普通であるため，このように描画の手順をマクロ化して用いている．当然，リンクのように円弧と線分をひとつひとつ入力した図形を用いてもよい．また，リンクの中心に描かれている直線やスライダの溝の中心を通る直線は，それぞれリンクとスライダのアイコンを同時に表示したものである．この場合アイコンと骨格形状が一致するので，各部品の骨格形状を表示したものでもある．

5.2.2 制約条件式の追加による形状変更

図5.14では，「スライダ-リンク機構を利用する」ということから，とりあえずスライダとリンクを定義した状態であるといえる．次に，図5.14の状態では全体として小さいのでまずピンの移動経路の始点と終点を指定しスライダの形状を変更する．この時，スライダの形状をセグメント単位で変形するのではなく，スライダの骨格形状を変更することで，概略形状も変更する．したがって，骨格の両端点の位置をマウスなどで指定するだけでよい．その結果を図5.15に示す．

図5.15の状態では，ピンP1をLink2によってスライダの上端に移動させるためには，明らかにリンクの長さが不足している．そこでピンP1の位置をスライダ上端に固定した上で，制約条件式を解くことによってリンク長さを決定する．ただし，この段階ではリンク長さに関しては制約条件が定義されていないので，図5.16のような結果を導くことがある．

図5.16の状態では，Link1とLink2のリンク長の差が大きすぎるので，設

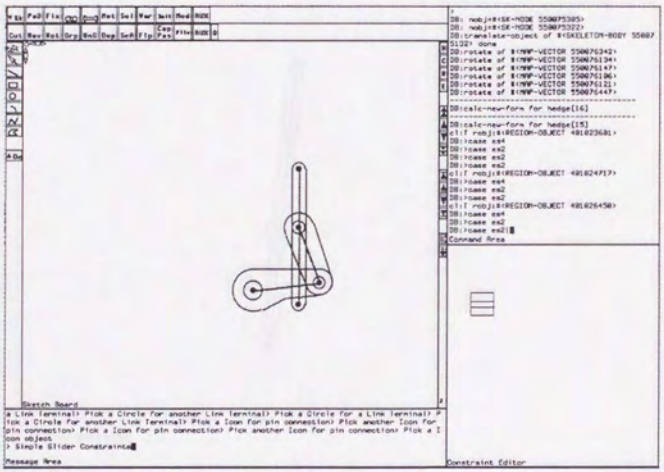


図 5.14: スライダ・リンク機構のスケッチ例

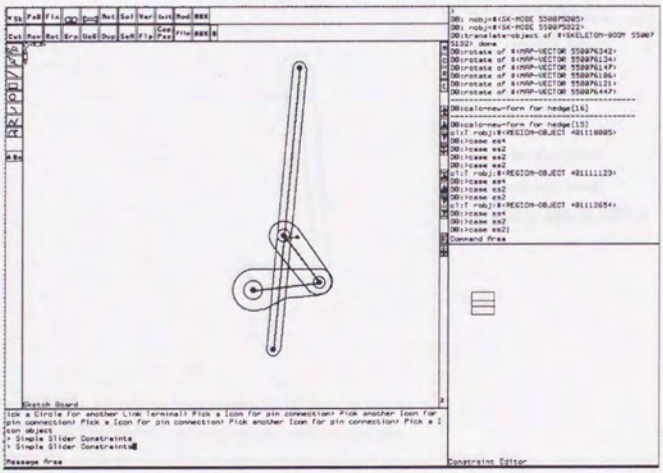


図 5.15: 骨格形状モデルによるスライダの変形

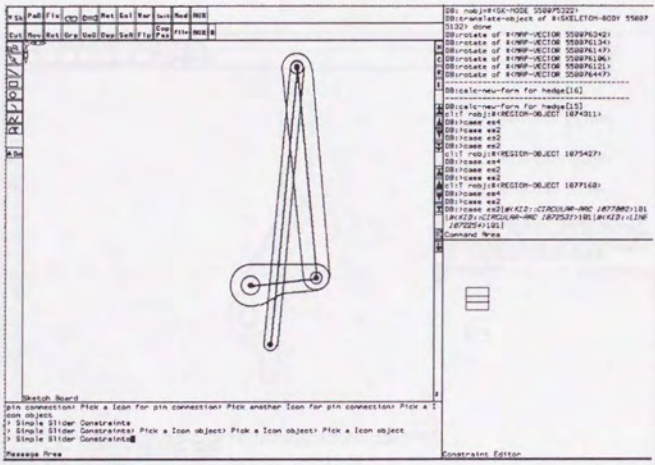


図 5.16: デフォルト値を用いたリンクの変形結果

計者が「Link1 と Link2 の長さは等しい」といったような条件を与え，Link1 か Link2 のどちらかに適当な長さを指定しておけば，図 5.17のような結果を得ることができる．実際には「Link1 と Link2 の長さは等しい」という制約条件は，(5.10) 式のように表現される．

$$L_1 - L_2 = 0 \tag{5.10}$$

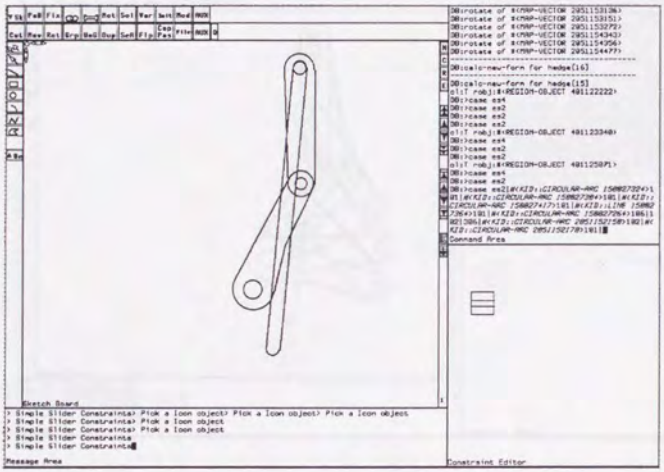


図 5.17: 設計者定義の制約条件によるリンクの変形結果

このような変形操作後も Link1 や Link2 に対する制約条件は有効であり，Slider1 の移動経路が拡大されたことに付随する制約条件の更新も行われているので，Link1 の姿勢を変えながら制約条件式を解くことによって，図 5.18のようなスライダ - リンク機構の動作確認が可能である．

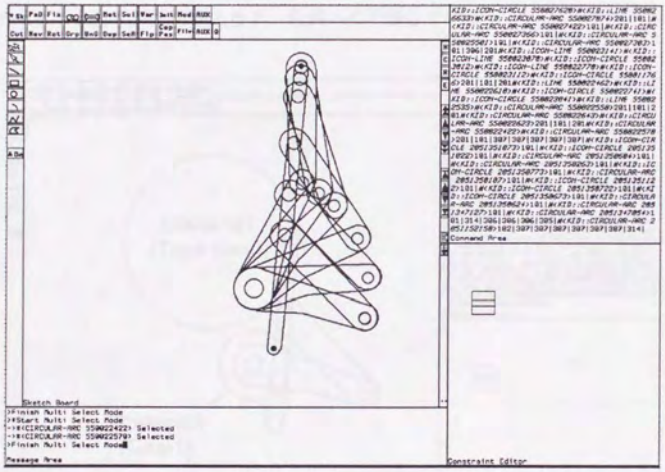


図 5.18: スライダ - リンク機構の動作シミュレーション (1)

5.2.3 部品の干渉回避形状の生成例

つづいて，部品間の非干渉条件を満たすために，部品形状を変形していく過程を示す．図5.17までに，ピンP1が移動するべき始点と終点を指定し，それに対応したリンク形状の変形が終了している．ここでリンク(Link1)を駆動するために必要な歯車(Gear1)と，VTRのテープヘッドを図5.17に追加する．たとえば図5.19のようにテープヘッドを置き，歯車をLink1の固定点に接続するようにスケッチを描く．この状態では，スライダに対してテープヘッドと歯車が障害物となり，たがいに干渉している．

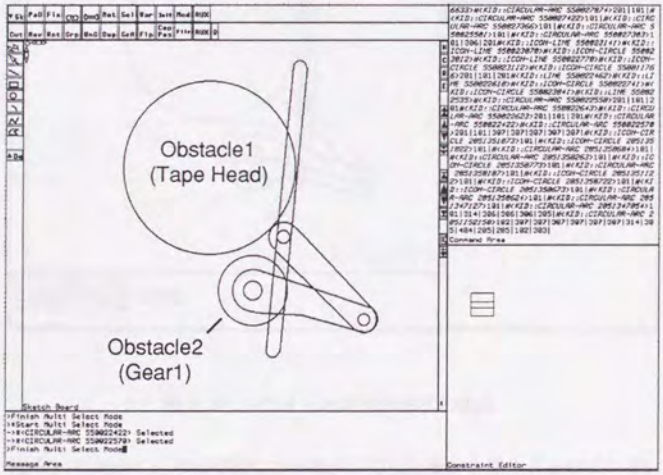


図 5.19: テープヘッド，歯車とスライダの干渉

ここで，図5.19に示したようなテープヘッドや歯車とスライダの干渉を回避するために，スライダの形状を変更する．一般にヘッドや歯車の形状は変更できないので，テープヘッドや歯車の位置が変更できないとすると，スライダの経路を変更することによって干渉を回避することになる．試作システ

ムでは，形状変更の対象としてスライダを選択し，障害物としてテープヘッドと歯車をあわわしている円を選択することにより，図5.20に示すような結果を得ることができる．図にも示したように，もともとは1本の直線経路であったスライダが，Slider1-1，Slider1-2，Slider1-3の各部分スライダに分割されている．

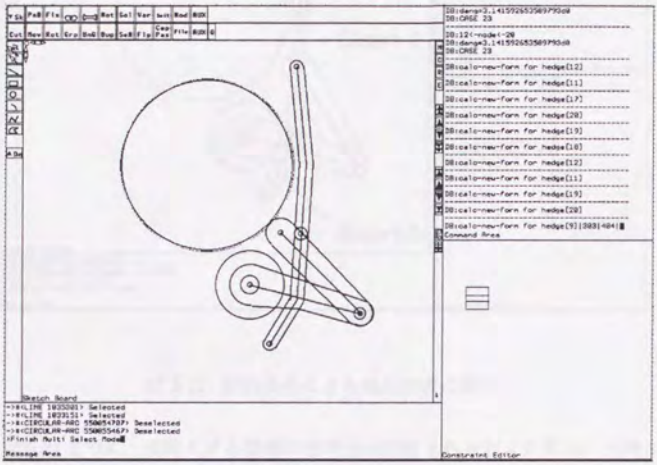


図 5.20: スライダの非干渉形状の生成

図5.20において，本来接続していなければならないスライダ上のピンP1とリンクLink2が分離しているのは，非干渉形状の生成過程では，指定された骨格形状以外は考慮していないためである．したがって非干渉形状生成の後処理として，制約条件式を評価することにより，接続関係を満足する各部品位置を計算する必要がある．非干渉形状を生成する際に，Slider1-1，Slider1-2，Slider1-3の各部分スライダに対応した制約条件式も生成されている．そ

の結果，図5.21に示したように，Slider1-2 上にピン P1 があると求められる[1].

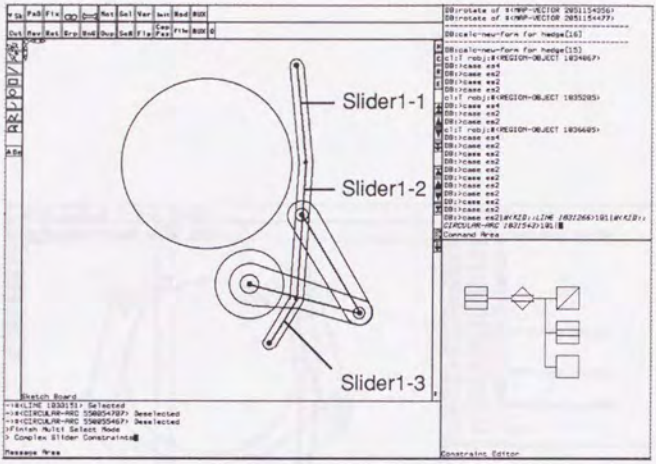


図 5.21: 制約条件による部品位置の導出

図5.21までに，目的とする機構の右半分が定義されたことになる．引続き残りの左半分を設計するために，すでにスケッチされている Link1 や Link2 を複写するなどして，図5.22に示すスケッチを描いたものとする．

この状態でも，スライダがテーブヘッドとリンクを駆動する歯車と干渉する．そこで図5.19の時と同様に，非干渉形状を生成する対象としてスライダを選び，干渉している障害物としてテーブヘッドと歯車を指定する．その結果システムが求めたスライダの非干渉形状を図5.23に示す．

またこの状態で，Link4 が Slider2 の上端まで移動することが可能なようにリンク長を求める．その結果が図5.24である．図から明らかなように，リンクがテーブヘッドに干渉している．そこで今度は，Link4 の非干渉条件を考

[1]この場合の制約条件の評価方法については 5.2.4で述べる

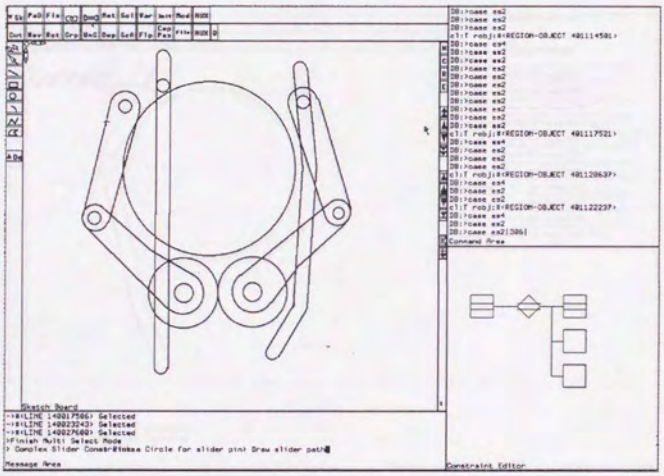


図 5.22: VTR テープ・ローディング機構のスケッチ

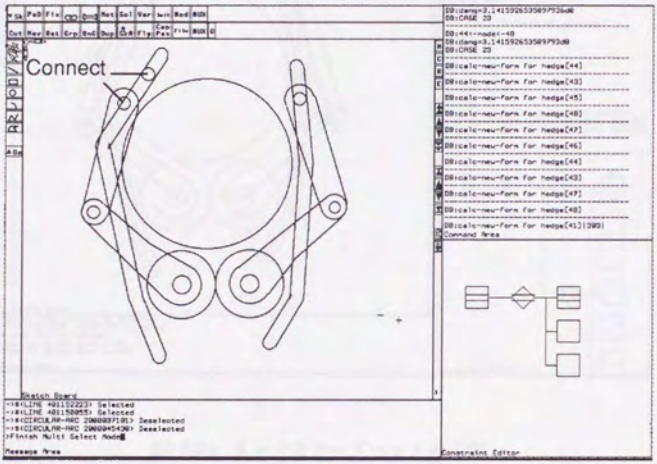


図 5.23: スライダの非干渉形状 (2)

慮する必要がある．スライダの時と同様に，干渉回避のために形状を変更するLink4と，障害物であるテーブヘッドを指定することにより，図5.25に示す結果を得る．ただし，図5.25では，そのままでは煩雑な図になるので説明のためにスライダのスケッチを消去してある．

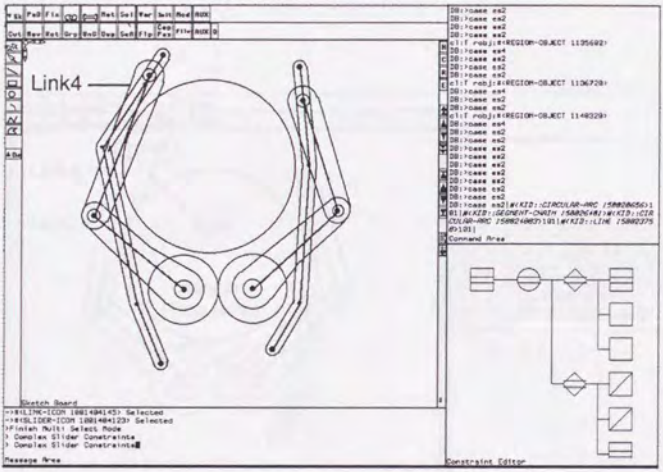


図 5.24: リンクとテーブヘッドの干渉

図5.25に示したように，リンクの場合には，骨格形状と対偶モデルのアイコンが異なる形状となる場合がある．これは，アイコンが対応する対偶モデルの制約条件を視覚化するものだからである．リンクの場合は「リンク両端点の距離が一定」という条件を表すため，実際の部品形状によらず，他の部品と接続している2点の距離を表すようなアイコンとなる．一方骨格形状は，部品形状の「軸」を表すものであり常に概略形状の内部に存在するので，図5.25からわかるように，概略形状にしたがって途中で屈曲した形状となる．

これに対し，スライダの場合は，ピンが移動するべき範囲を表すという意

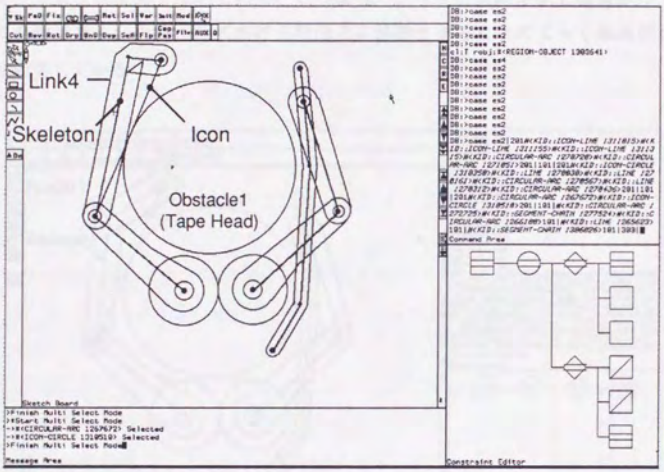


図 5.25: リンクの非干渉形状

味でスライダ両端の位置が重要であるだけでなく，どのような経路を通過してピンが移動すべきかも重要なので，スライダ溝の骨格形状とそのアイコンの形状は完全に一致している．そのため，図5.19や図5.22において，干渉を回避するためにスライダの形状を変更しても骨格形状とアイコンの分離が生じない．

また，システムが自動的に生成した部品形状に満足できない場合は，設計者が直接形状を変更することも可能である．図5.26は，リンク形状の屈曲点を中央よりに変更した例である．この場合，リンクの概略形状を直接操作する必要はなく，骨格形状における屈曲点を移動させることによって部品形状を変更している．

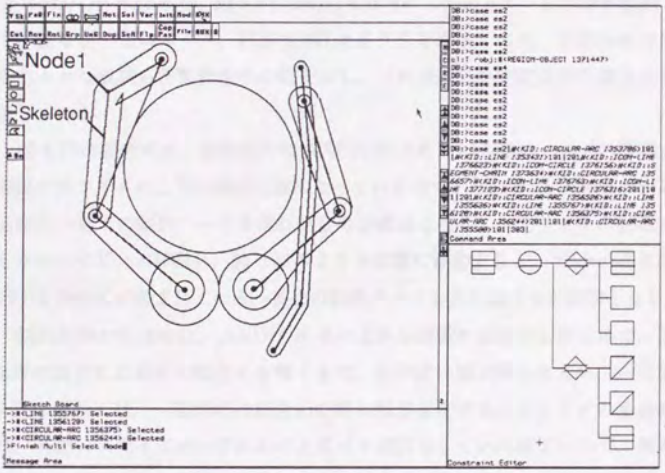


図 5.26: 骨格形状を用いた部品形状の変更

5.2.4 制約条件の切り換えを含む機構の評価

図5.26までに，目的としているVTRテープ・ローディング機構が設計されたことになる．この例ではスライダの経路が屈曲しているため，各部分スライダごとに制約条件式を切り換える必要がある．そこで最後に，制約条件の切り換えを必要とする機構の評価について例を示す．

3.3で述べたように，運動にともなって制約条件が変化する場合，制約条件式を切り換えながら解く必要がある．たとえば図5.27では，スライダを表す制約条件は，三つの部分スライダごとにことなる条件式で表さなければならない．そこで条件式を解く際にも，3通りの条件式の組合せについて解く．図5.27の右下に表示されているのが3.3.3で述べた，制約条件の評価手順を示すAND/OR木である．図3.4で説明した記号にしたがって，正方形が制約ノード，菱形がORコネクタ，円がANDコネクタを表す．また，各記号に付けられた＝は成功した制約条件の組を表し，／は失敗した制約条件の組合せを表している．

図5.27の場合には，制約条件のAND/OR木における制約ノードの順番が実際のスライダの上下の順番と逆になっているので，Slider1-1上にピンがある時に一番下の制約ノードを含む組合せが成功となる．一方リンクの回転にしたがってピンが移動し，図5.28のような位置に移動すると，Slider1-3に対応する条件式が含まれている一番上の制約ノードを含む組合せが成功となる．

制約条件の組合せは，AND/OR木の上から順番に組合せて解くので，図5.27の場合には最後の組合せを解くまで，条件式の解が得られない．一方図5.28の場合には，一番始めの組合せの時に解が存在する．スライダの場合には，各部分スライダのいずれかの上にピンが存在していればよいので，解が存在する制約条件の組合せが見つかれば，それ以外の組合せについて解く必要がない．そこで図5.28の場合には，一番上の条件式の組合せを解くだけで制約条件の評価を終了している．

制約条件の組合せについては制約評価ユニットが自動的に行うので，設計者が条件式の切り換えを操作したり，意識する必要はない．したがって，切

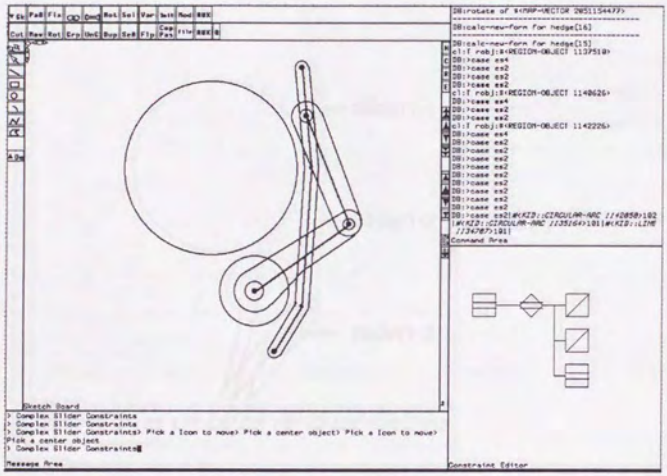


図 5.27: 不等式条件を含む制約条件の評価

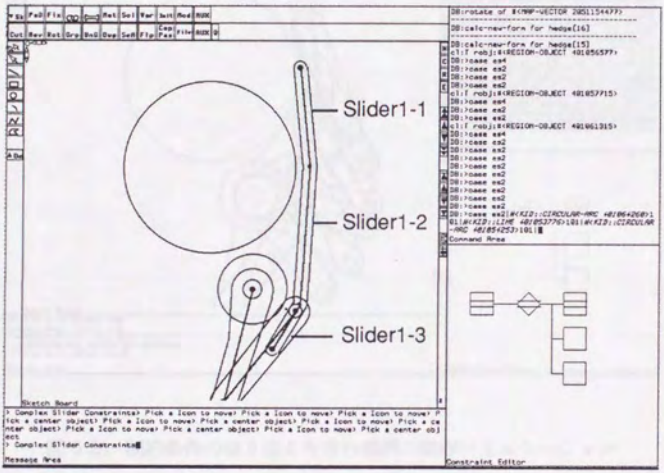


図 5.28: 不等式条件を含む制約条件の評価例 (2)

り換えを必要とする条件式も他の通常の条件式と同様に扱うことができ，リンクに対して適当な姿勢を指定し，その状態での各部品の位置を求めることを繰り返すことで，こうした機構の動作を検討することも可能である．図 5.29に複数の姿勢での計算結果を重ねて表示した例を示す．

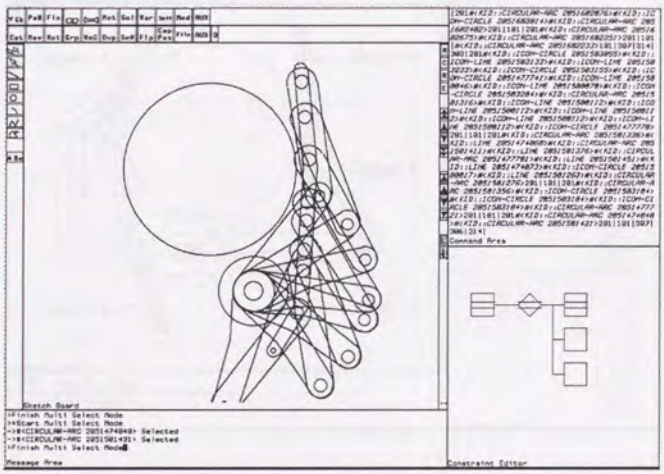


図 5.29: 制約条件の切り換えを含む機構の動作シミュレーション

図 5.26に示したような，スライダを 2 個含む場合でも同様に処理を行う．ただし，3.3.2で述べたように，制約条件の組合せの数を減らすために，右半分の機構と左半分の機構を分離して条件式を解く．そのため図の右下に示されている AND/OR 木は，AND コネクタを用いた木構造となっている．

図 5.30では，P1 が Slider1-2 上に，P2 が Slider2-1 上にそれぞれ存在しているので，制約条件を解いた結果，解が得られる条件式の組合せは図の右下に＝が示されている組合せとなる．つまりまず P1 の位置を決定する条件式の組合せとして，Slider1-2 に対応する条件式を含む上から 2 番目の組合せが

成功となり，つづいてP2の位置を決定する条件式としては，Slider2-1に対応する制約条件が一番下の制約ノードに含まれているので，一番最後の組合せが成功となる．

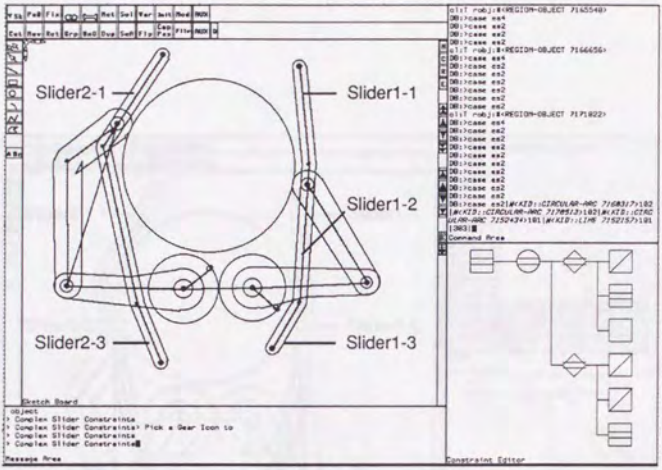


図 5.30: 2個のスライダを含む機構の評価 (1)

さらに歯車を回転させていくと，P1がSlider1-2上にあり，P2はSlider2-2上に存在する状態となり，図5.31に示すようにAND/OR木の2番目と5番目の制約条件の組合せの時成功となる．こうして歯車の回転角を変えながら，リンクやスライダのピン位置などを求めることで図5.32に示すように機構の動作シミュレーションが行える．

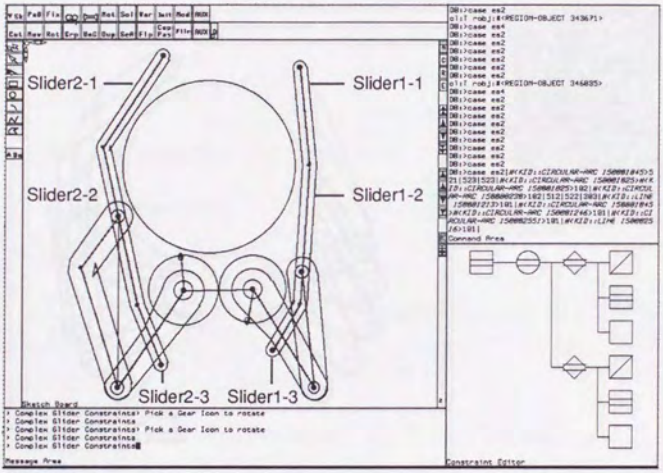


図 5.31: 2個のスライダを含む機構の評価 (2)

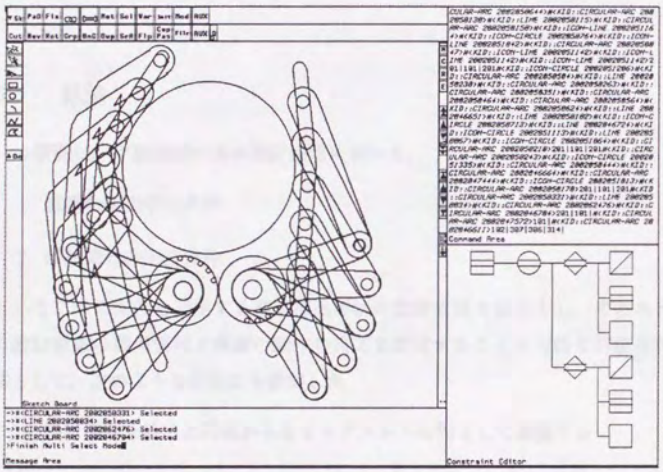


図 5.32: VTR テープ・ローディング機構の動作シミュレーション

第6章

結論と展望

6.1 結論

本研究は，平面機構の基本設計段階において，

1. 機構の運動学的条件
2. 部品間の非干渉条件

という二つの条件を満足する機構部品形状の設計支援を目的とし，そのために設計対象の幾何情報と機構の動作の両方を表現することが可能な対象表現法として，次のような表現法を提案した．

- 幾何情報を線分と円弧からなるセグメントの列として表現する
- 機構の運動学的条件を，機構を構成する部品の位置や姿勢をあらわす設計変数をパラメータとする制約式で表現する
- ゴム変形モデルに基づく骨格形状を利用した変形手法により，制約式を満足する設計変数の値と幾何情報を連動させる

このような対象表現をとることにより，従来の対象表現法にみられない以下のような利点が得られた．

1. 未定義部分を含む対象の表現：概略の部品形状や稜線が閉じていない部品形状の一部のような，不完全な形状を扱うことができるようになった．

また設計対象の動作に関しては，制約が与えられていない部分を未定義部分として扱うことができるようになった．

2. 付加的な条件（設計者の意図）の表現が可能：制約式で表現できれば他の条件と同様に処理できるので，付加的な条件としての設計者の意図も扱えるようになった．

3. 整合性を維持した対話的な設計：制約を解きながら設計変数を決定する機能を付加することにより，幾何情報と制約の整合性を計算機によって管理できるようになった．その結果，設計者による形状変更や設計変数の値の変更に対して，計算機が制約を満足する状態のモデルを示しさらにそれを用いて詳細化していくというように，システムと対話的にモデルを構築，操作することが可能になった．

4. 表現対象の自由度大：制約式表現できる範囲であらゆる機構を扱うことができるようになった．

5. モデル表現力の拡張：変形手法として「ゴム変形モデル」を導入し，制約式を構成する設計変数と幾何情報とを骨格形状を介して関係付けることにより，幾何情報，機構の動作，付加的条件を連動して扱うことができるようになった．

こうした対象表現法を具体的に実現する方法として，第2章において，機構モデルとして4つの対象表現を組み合わせて用いることについて説明し，その後で各表現間の関係について述べた．第3章では，制約式の定義方法について述べ，それらの制約式を解く方法として，条件式がすべての設計変数を決定するのに十分でない場合でも，デフォルト値を用いて解く方法について述べた．また，動作にともなって対偶関係の条件式が変化する機構を扱うために，不等式を含む場合の制約条件を解く手法を示した．第4章では，ゴム変形モデルに基づいた，骨格形状による概略形状の変形方法について述べ，さらに，この手法を用いて部品の干渉回避形状が生成できることを示した．

最後に，本研究で提案した対象表現法に基づいて，設計対象である機構のモデルを構築することのできる，対話的なユーザインタフェースを備えた機構のモデリングシステムを試作した．また，実際に本システムを用いて機構をモデリングした結果，設計対象のモデルを比較的容易に構成することができ，また部品形状と対偶関係との整合性をシステムが管理することにより，機構としての整合性を維持したモデルの修正も可能であることなど，本研究において提案した手法の有効性が確認できた．

6.2 展望

本研究で提案した対象表現では，制約条件の付加が設計対象を詳細化することにあたる．制約条件を解く際にデフォルト値を導入したことで，条件式が不足している状態でも制約を評価することができるので，設計者は制約条件を一度に定義する必要はなく，必要なところから与えればよい．このように設計対象を徐々に詳細化していくことができる．また，基本的に条件式を付加する順序の制限はないので，設計者は任意の部分を詳細化することができる．すなわち，設計対象を部分的に詳細化していくことが可能であるといえる．このような特徴から，トップダウン的な設計支援の要素技術として応用することが考えられる．

謝辞

東京大学工学部・木村文彦教授には，修士課程の学生として研究室に配属されて以来，指導教官として適切な研究テーマと数々の貴重な御指導を頂いた．機械系 CAD およびファクトリーオートメーションの分野で国際的に活躍されている木村教授に師事することで，常に最先端の研究にふれることができた．また国内のみならず海外における学会発表の機会を与えて下さり，多くの研究者と交流する貴重な経験を得ることができた．心から感謝する次第である．

東京大学教養学部・鈴木宏正助教授には，本研究の全般にわたり常に的確な御助言を頂いた．しばしば暗礁に乗り上げそうになる研究をまとめることができたのも，その進むべき方向をこの分野に関する幅広い視点から示して頂いたおかげである．ここに心からお礼を申し上げる。

東京大学工学部・乾正知講師には，研究グループにおける議論を通して研究の進め方などさまざまな御意見を頂いた．また本論文をまとめる上でその構成や内容について非常に多くの貴重な助言を頂いた．心から感謝の意を表する．

吉田良正助手をはじめとする東京大学工学部精密機械工学科木村研究室の諸氏には，研究生活全般において大変お世話になった．山口泰氏 (現・東京電機大学)，小林一也氏 (現・富山県立大学) には形状処理に関しての助言を頂き，さらに両氏から研究に取り組む姿勢を学ばせて頂いた．安藤英俊，田中一郎 (現・日本学術振興会特別研究員)，寺沢幹雄，高橋究の諸氏とは日頃の話し合いによりお互いの研究に対して多くの意見交換ができ，本研究を進める上で非常に有益であった．特に安藤氏には，3.2.3におけるブフバーガ・ア

ルゴリズムを実現したプログラムを提供して頂いた．ここに改めて感謝する次第である．

東京大学工学部・中島尚正教授，大園成夫教授，新井民夫教授，佐々木健助教授には本論文の査読をして頂き，多くの貴重な御意見を頂戴した．ここに感謝の意を表する．

研究を進めるにあたり，精密工学会産学協同研究協議会「高度生産自動化のためのプロダクト・モデリング・システムの開発」研究協力分科会の会員の方々には様々な御意見や御協力をいただいた．心からお礼を申し上げる．

最後に，長い間大学生活を支援し続けてくれた家族に深く感謝する．

1992 年 12 月

吉川 浩一

参考文献

[伊藤 90] 伊藤 公俊: “知的 CAD のための属性モデリング方法論”, 人工知能学会 SIG-KBS-9002, 1990.

[北嶋 81] 北嶋 克寛, 吉川 弘之: “階層的ネットワークモデルに基づく対話型機械設計システム HIMADES-1 の開発”, 精密機械, Vol.47, No.12, pp.50-57, 1981

[吉川 90a] 吉川 浩一: “制約表現に基づく対話的機構モデリングシステムの開発”, 1989 年度東京大学大学院修士論文, 1990.

[桐山 92] 桐山 考司: “設計対象知識の体系化 (第 1 報) – 基礎理論 –”, 1992 年度精密工学会秋季大会学術講演会講演論文集, pp.53-54, 1992.

[佐藤 87] 佐藤達志, 鈴木宏正, 木村文彦 : “緩やかな拘束条件のもとでの板金部品形状の生成”, 第 5 回設計自動化講演会講演論文集, 精密工学会, 1987.

[写真工業 87] “カメラテスト・テクニカルレポート”, 写真工業, Vol.6, pp.84-96, 1987.

[杉原 87] 杉原厚吉 : “図形の移動と変形に関する推論技術”, 情報処理 Vol.28, No.11, pp.1485-1492, 1987.

[図法研究 90] 概念図画法研究会: “特許図面の新作図法 (上)– メカニズムの迅速な理解を目指す –”, 機械設計, Vol.34, No.16, pp.83-88, 1990.

[鈴木 88] 鈴木宏正, 安藤英俊, 原田雅之, 木村文彦：“緩やかな拘束条件のもとでの板金部品形状の自動生成”, 昭和 63 年度精密工学会秋季大会学術講演会講演論文集, pp.771-772, 1988.

[鈴木 89a] 鈴木 宏正: “属性モデルの問題点”, PIXEL, No.86, pp.153-155, 1989.

[鈴木 89b] 鈴木 宏正: “プロダクトモデルとインテリジェント CAD”, コンピュートロール, No.25, pp.91-97, コロナ社, 1989.

[PR 90] “製品の機能や生産性の事前評価にもとづく機械生産の高精度化” 精密工学会産学協同研究協議会, 研究協力分科会第 2 年次報告書, pp.24-30, 1990.

[図法研究 91] 概念図画法研究会: “特許図面の新作図法 (下)– メカニズムの迅速な理解を目指す –”, 機械設計, Vol.35, No.1, pp.82-88, 1991.

[松尾 91a] 松尾 英治: “機構の仕組みを表現する概念図, 技術情報の蓄積伝達が容易に”, 日経メカニカル, 6-10, No.351, pp.42-50, 1991.

[松尾 91b] 松尾 英治: “特許図面の新作図法と CAD 化”, 機械設計, Vol.35, No.9, pp.79-84, 1991.

[吉川 77] 吉川 弘之: “設計学研究”, 精密機械, Vol.43, No.1, pp.21-26, 1977.

[吉川 79] 吉川 弘之: “一般設計学序説”, 精密機械, Vol.45, No.8, pp.20-26, 1979.

[Ando 89] Hidetoshi Ando, Hiromasa Suzuki and Fumihiko Kimura : “A Geometric Reasoning System For Mechanical Product Design”, Proceedings of CAPE '89 - Third International Conference on Computer Applications in Production & Engineering, pp.131-139 (1989).

[Barraquand 91] J.Barraquand and J-C.Latombe: "Robot Motion Planning: A Distributed Representation Approach", The International Journal of Robotics Research, Vol.10 Num. 6 pp.628-649 1991.

[Buchberger 85] B.Buchberger: "Grobner bases: an alogrithmic method in polynomial ideal theory", in N.K.Bose ed., Recent Trends in Multidimensional Systems, pp.184-232 1985.

[Kapur 88] D.Kapur and J.L.Mundy: "Wu's Method and Its Application to Perspective Viewing", Artificial Intelligence, Vol.37, pp.15-36.

[Faltings 88] Boi Faltings: "Qualitative Kinematics and Computer-Aided Design" Proc. IFIP WG5.2, Workshop on Intelligent CAD, pp.145-164. 1988.

[Faltings 90] Boi Faltings: "Qualitative Kinematics in Mechanisms" Artificial Intelligence Vol.44, Num.1-2, pp.89-119, 1990.

[Finger 89a] Susan Finger and John R. Dixon: "A Review of Research in Mechanical Engineering Design. Part I: Descriptive, Prescriptive, and Compputer-Based Models of Design Processes", Research in Engineering Design, Vol.1, pp.51-67, 1989.

[Finger 89b] Susan Finger and John R. Dixon: "A Review of Research in Mechanical Engineering Design. Part II: Representations, Analysis, and Design for the Life Cycle", Research in Engineering Design, Vol.1, pp.121-137, 1989.

[Forbus 91] K.D.Forbus, P.Nielsen and B.Faltings: "Qualitative spatial reasoning: the CLOCK project" Artificial Intelligence Vol.51, Num.1-3, pp.417-471, 1991.

[Gossard 88] D.C.Gossard, R.P.Zuffante and H.Sakurai: "Representing Dimensions, Tolerances, and Features in MCAE Systems", IEEE Computer Graphics and Applications, March, pp.51-59, 1988.

[Hwang 88] Y.K.Hwang and N.Ahuja 'Path Planning Using a Potential Fieled Representation', Proc. of IEEE International Conference on Robotics and Automation, Vol.1 pp.648-649 1988.

[Jacobs 89] P.Jacobs and J.Canny 'Planning Smooth Paths for Mobile Robots', Proc. of IEEE International Conference on Robotics and Automation, Vol.1 pp.2-7 1989.

[Joskowicz 88] L.Joskowicz and S.Addanki: "From Kinematics to Shape: An Approach to Innovative Design", Proc. AAAI-88, pp.347-352 1988.

[Joskowicz 91a] L.Joskowicz and E.Sacks: "Computational kinematics", Artificial Intelligence Vol.51, Num.1-3, pp.381-416, 1991.

[Joskowicz 91b] L.Joskowicz and E.Sacks: "Incremental Configuration Space Construction for Mechanism Analysis", Proc. AAAI-91, pp.888-893 1991.

[Keene 89] S. Keene : "Object-Oriented Programming in Common Lisp (A Programmer's Guide to CLOS)", Addison-Wesley Publishing Company,(1989).

[Khosla 88] P.Khosla and R.Volpe: "Superquadric Artificial Potentials for Obstacle Avoidance and Approach" Proc. of IEEE International Conference on Robotics and Automation, Vol.3 pp.1778-1784 1988.

[Ko 87] Heedong Ko and Kunwoo Lee: "Automatic assembling procedure generation from mating conditions", Computer-Aided Design, Vol.19, No.1, pp.3-10, 1987.

[Krause 89] F.-L. Krause, M. Bienert, F.H. Vosgerau and N. Yaramanoglu: "Feature Oriented System Design for Geometric Modeling", Memorandum, Technical University Berlin, 1989.

[Lee 85a] K.Lee and D.C.Gossard: "A Hierarchical Data Structure for Representing Assemblies:Part I", Computer-Aided Design, Vol.17, No.1, pp.15-19, 1985.

[Lee 85b] K.Lee and G.Andrews: "Inference of the positions of components in an assembly", Computer-Aided Design, Vol.17, No.1, pp.20-24, 1985.

[Mäntylä 90] M.Mäntylä: "A modeling system for top-down design of assembled products", IBM Journal of Research and Development, Vol.34, No.5, pp.636-659 1990.

[Nelson 85] G.Nelson: "Juno, a constraint-based graphics system", ACM SIGGRAPH '85, Vol.19, No.3, pp.235-243, 1985.

[Orlandea 77] N.Orlandrea, M.A.Chace and D.A.Calahan: "A sparsity-oriented approach to the dynamic analysis and design of mechanical systems", Trans. ASME Journal of Engineering for Industry, Vol.99-3, pp.773-784, 1977.

[Rehg 88] J.Rehg, A.Elfes, S.Talukdar, R.Woodbury, M.Eisenberger and R.Edahl: "CASE:Computer-Aided Simultaneous Engineering", Technical Report, Engineering Design Research Center, Carnegie Mellon University, EDRC-18-05-88, 1988.

[Serrano 88] D.Serrano and D.Gossard: "Constraint management in MCAE", Artificial Intelligence in Engineering:Design, D.Sriram, R.A.Adey ed, Computational Mechanics Publications, pp.217-240, 1988.

[Sheth 72] P.N.Sheth and J.J.Uicker Jr. "IMP(Integrated Mechanism Program) – A computer aided design analysis system for mechanism and linkages", Trans. ASME Journal of Engineering for Industry, Vol.94-2, pp.454-464, 1972.

[Shimada 92] K.Shimada and D.Gossard: "Automated Shape Generation of Components in Mechanical Assemblies", Advances in Design Automation, ASME, 1992.

[Ulrich 88] K.T.Ulrich and W.P.Seering: "Function Sharing in Mechanical Design", Proc. AAAI-88, vol.1, pp.342-346, 1988.

[Wu 88] L.Wu and A.P.Pisano: "The Development and Application of Kinemateic Icon and Inactive-Joint Concepts in Automated Mechanism Sketching", Transactions of the ASME, Journal of Mechanisms,Transmissions and Automation in Design, Vol.110, pp.73-80, MARCH 1988.

発表論文一覧

投稿論文

[Kikkawa 93] K. Kikkawa, H. Suzuki, H. Ando and F. Kimura: "A Product Modeling System for Top-down Desing of Machine Assembly with Kinematic Motion", Robotics & Computer-Integrated Manufacturing, Vol.10, No.1/2, pp.49-55, 1993.

口頭発表

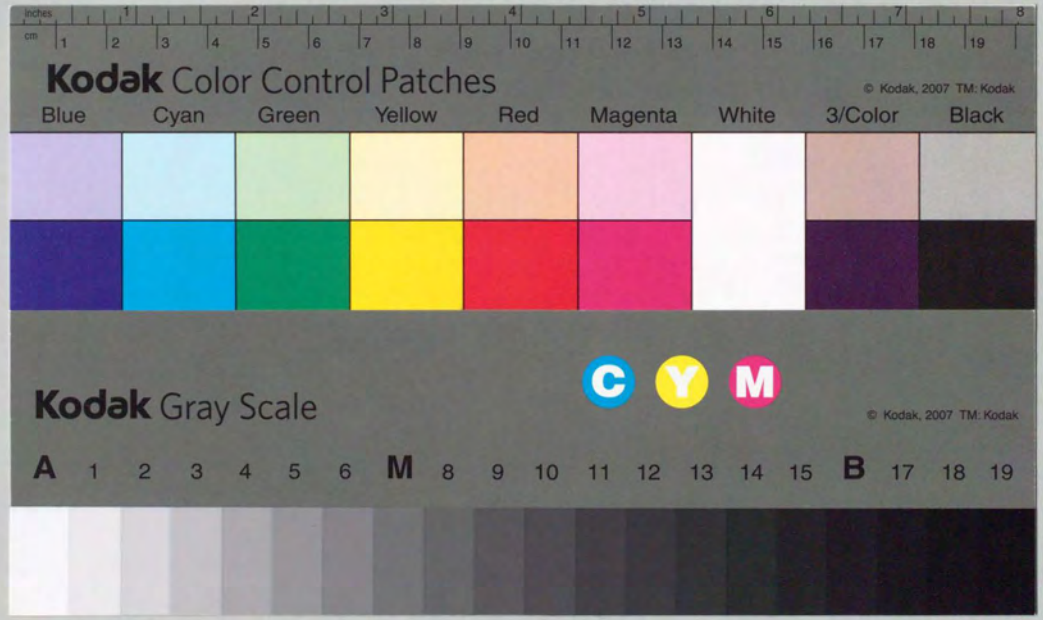
[吉川 89] 吉川 浩一, 鈴木 宏正, 木村 文彦: "プロダクトモデルに基づく機構の表現", 1989 年度精密工学会秋季大会学術講演会講演論文集, pp.245-246, 1989.

[吉川 90] 吉川 浩一, 鈴木 宏正, 木村 文彦: "部品形状に着目した機構のモデリング", 1990 年度精密工学会秋季大会学術講演会講演論文集, pp.905-906, 1990.

[吉川 91a] 吉川 浩一, 鈴木 宏正, 木村 文彦: "プロダクトモデルに基づく機構の表現 (第 2 報) – 接触の変化をともなう機構の処理 –", 1991 年度精密工学会春季大会学術講演会講演論文集, pp.533-534, 1991.

[吉川 91b] 吉川 浩一, 鈴木 宏正, 木村 文彦: "プロダクトモデルに基づく機構の表現 (第 3 報) – スケルトンによる形状の操作 –", 1991 年度精密工学会秋季大会学術講演会講演論文集, pp.811-812, 1991.

[Kikkawa 91c] K. Kikkawa, H. Suzuki, H. Ando and F. Kimura: "A Product Modeling System for Top-down Desing of Machine Assembly with Kinematic Motion", MSTF '91, 5th International Conference on the Manufacturing Science and Technology of the Future, Preprints, 1991.


inches
cm
Kodak Color Control Patches
© Kodak, 2007 TM: Kodak
Blue
Cyan
Green
Yellow
Red
Magenta
White
3/Color
Black
Kodak Gray Scale
C
Y
M
© Kodak, 2007 TM: Kodak
A 1 2 3 4 5 6 M 8 9 10 11 12 13 14 15 B 17 18 19