分子振動表示プログラムMOLCATの開発

筒井　祐子*・和佐田裕昭**

（1995年6月30日受理）

The Development of a Program Visualizing Molecular Vibrations

Yuko TSUTSUI* and Hiroaki WASADA**

目　次

* 名古屋大学理学部化学科分析化学研究室，　** 岐阜大学教養部化学教室

1. はじめに

今日，計算科学的手法は，化学反応の制御，分子設計，分子構造の決定などに直接的または間接的に寄与する目的で積極的に使用されている。この際，最も基礎的な情報として対象とする化合物の電子状態や振動状態を何らかの方法で知ることが必要とされる。理論的にこれらを研究するための方法の一つである分子軌道法は，近年の計算機性能の非常な進歩にともなって広く用いられるようになっている[1]。過去においては，X線構造解析などの実験的手段によって得られた分子構造に対して半経験的分子軌道法計算を適用して HOMO-LUMO 相互作用などを調べ，反応性を定性的に議論するといった進め方が多かった。しかし，ここ20年程の方法論的進歩と前述の計算機事情の改善にともない，実験的情報を用いることなく分子の性質を算出する非経験的分子軌道法が一般化した。例えば，H_2O分子のような単純な分子では，非経験的分子軌道法に基づいて算出された分子構造は実験値と同等のものであると考えられる程である。現在の分子軌道法計算の対象は，励起状態分子や生体内反応に関与する複雑な構造を持った分子，高配位錯体，反応の途中で出現する中間体や遷移状態へと広がっている。遷移状態は通常の状態とは異なり，反応座標方向にはエネルギー極大に対応しているので，実験的に明確な構造を把握して研究することが極めて困難であり，理論的手法による解明が期待される。

非経験的分子軌道法計算を行うことにより，分子の最安定構造だけでなく，各構造パラメーターを多様に変化させた点での状態を知ることができる。また，それらの各点で振動解析計算を行うことで分子構造の性格を決定できる。分子を構成する各原子に働く力がゼロになるように対象分子の構造を変化させることで，実験によることなく分子構造を理論的に最適化する。このようにして得られた最適構造がポテンシャルエネルギー曲面上でどのような性格を持っているのかを明らかにするためには振動解析計算が必須である。算出された基準振動数が全て実数ならば最適構造はエネルギー極小に対応し，一つだけ虚数ならば，何らかの反応の遷移状態に対応していることが分かる。この際，エネルギー極小構造ならば各基準振動を詳細に検討することにより，化学反応の開始がどのような原子変位によって引き起こされるかを知ることができる。遷移状態での虚数振動の方向を調べることにより，化学反応がどの方向へ向かっているのかを予測できる。この場合，得られた遷移構造が問題とする化学反応に適切に対応したものかどうかを知る手掛かりとしても重要である。

以上のように，非経験的分子軌道法計算に基づいて得られた分子構造や分子振動を詳細に検討することは，化学反応などを研究する上で最も基本的なことであるが，かつてのように原子数個以内からなる単純な分子系ではなく，より複雑な系が対象になってくると，計算から得られた数値をながめることだけでは，構造や振動のイメージをつかむことが難しい場合が多い。数値に従ってグラフ用紙等に描いてみることが，ある程度は有効なこともある。しかし，奥行きがうまく表現できないこと，数原子以上になると各原子の図が重なるため全体の状況把握が困難になる場合が多いことなどの問題点がある。さらに，分子を様々な方向・角度から検討するたびに分子図を描き直さねばならないので非常に不便である。そこで，計算によって得られた基準振動モードを分子構造とともにスクリーン上にリアルタイムで表示するソフトウェアがあることが望ましい。分子構造表示プログラムとしては現在まで色々な

ものが発表されている。そのなかで，分子軌道法計算の手助けとして最も有用なものは Molecular Editor[2] であろう。その他のプログラムの多くは色々な機能を持ち，それなりに有用な面もあるが，入力が複雑であることなどから，ルーチンワーク的に使用するには難があると思われる。また，最大の欠点は，分子振動を表示する機能を持たないことである。Molecular Editor を使用する際には，図1に示すように，振動ベクトルの先端を原子でなく点で置き換えるなどの工夫をすることで，ベクトル的な表示をすることもできるが，やはり便宜上のものであるため使用しにくい。

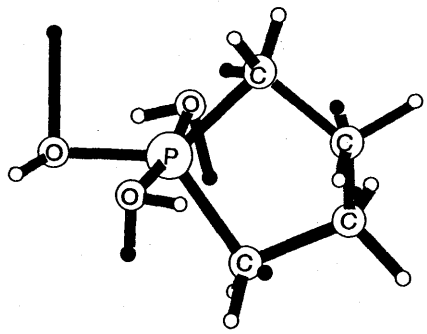


図1　Molecular Editor を用いて描いた分子振動の様子。● は振動ベクトルの先端を表し，黒線は振動ベクトルを表す。

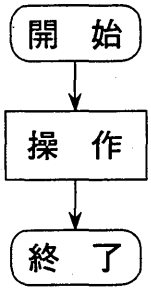
このような困難を本質的に回避するために，我々は非経験的分子軌道法計算の結果をそのまま入力として使用し，振動ベクトルを分子構造とともにリアルタイムで表示するプログラムを開発した[3]。この際，化学系研究室で普及していることや図形処理能力に長けていることを考慮して，Apple 社の Macintosh 用のソフトウェアとすることにした。さらに，フリーウェアとしてこのプログラムを公開した。

2. このソフトウェアに要求される条件

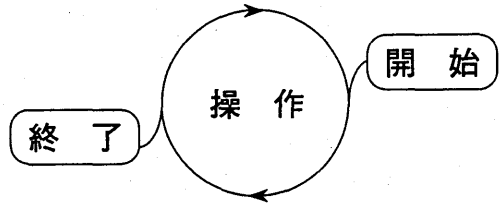
今回の分子振動表示用ソフトウェアの開発において必須とした条件は，主に以下のようなものである。第一に，入力データが単純明解であることである。既存の分子構造表示プログラムでは，特殊な形式で入力しなければならない場合が多く，このため入力データを作成する面倒があった。本プログラムではワークステーション等，他のコンピュータとの互換性が容易であるよう注意を払った。また，フリーフォーマットのテキストデータを基本的入力データとし，入力すべきパラメーターは元素記号と X，Y，Z 直交座標とした。特殊な分子内座標を扱いたい場合は，それを X，Y，Z 直交座標に変換する補助ルーチンで対処することにした。第二に，操作が容易なことである。特に，大型計算機用に開発されたプログラムでは，キーボード入力で分子回転等の数値を入力せねばならないことが多く，余分な操作を修得する努力が必要とされた。今回のプログラムでは，Macintosh のマウス操作等の GUI を最大限利用することで，ユーザーが振動解析計算結果の化学的考察のみに集中できるよう配慮した。第三に，振動解析計算の表示や分子回転を目的としているので計算精度が高く計算が速いことである。この目的のため，数値計算用プログラム言語として信頼性の高い FORTRAN を用い，数値演算コプロセッサーを使用するようにした[4]。FORTRAN では扱いにくいメモリー管理部分だけを C を用いてコーディングした[5]。

3. ソフトウェア設計上の指針

今回のプログラム開発にあたっては，以下に述べるようないくつかの基本的指針をたてて，それに沿ったコーディングを行った。第一は，大型計算機や DOS 上の FORTRAN プログラムにおける基本的な操作の流れが初めから終りへと一方向へ流れるのに対して，このプロ



(a) 一方向的な操作の流れ



(b) 無限ループ的な操作の流れ

グラムでは，無限ループを基本的構造としたことである。プログラムはあたかもオペレーティングシステムのように常に入力待ちの状態にあり，利用者の命令に従って動作するので，ユーザーの要求に柔軟に対応するプログラムを作成できる。第二に，図形の基本単位は，円，鍬形，結合を表す長方形および直線として，これらを作成するサブルーチンを考えた。サブルーチンには，これら基本図形の位置，大きさおよび方向を表すパラメーターを与える。それに従って，基本図形が目的の位置に目的どおりに指定できる。図形を単純な構成要素の組み合わせとして表現することにより，図形処理を専門とした他のプログラムとの図形の相互利用を可能としている。例えば，鍬の色だけを強調したい場合に，他の専門プログラムでそれを行うことが容易となる。第三に，利用者が分子振動等の考察のみに集中できるようにするため，分子の画面表示はできるだけ滑らかに行われることが望まれた。コンピューターのハードウェアの高速化により，解決される問題と考えることもできるが，ここではゲームプログラムの作成などで利用される仮想的なスクリーン（オフスクリーン）をメモリー上に定義して，全描画操作をオフスクリーンに対して行い，操作終了後にディスプレー上に転送する手法を用いることにした。第四に，Macintosh 上でのプログラミングであることを考慮して，システム 7 に対応かつ互換性ガイドラインに準拠するよう注意を払った。

4. MOLCATの概要

MOLCAT は，*ab initio* 計算で得られた分子構造を棒と球モデルで，振動モードを矢印で図示する。分子構造のみの表示も可能である。経験的計算はもちろんのこと，X 線構造解析などを用いた実験結果もまた分子内の各原子の X，Y，Z 座標を入力することによって図示することができる。また，附属の変換プログラム「Z 変換」を用いれば，分子内座標を扱うこともできる。特に，分子軌道法計算プログラム Gaussian シリーズ[6]の出力結果については，付属の変換プログラム「VibAnal」で MOLCAT の振動モード表示用データを，「Moled」で構造表示用データを取り出して簡単に計算結果を図示することができる。通常の入力データの例を図 2 に示す。

原子については，元素記号，X 座標，Y 座標，Z 座標の順にそれぞれを 1 個以上の空白で区切りながら一行で指定する。振動ベクトルの場合には，元素記号の代わりに表現する振動ベクトルの始点となる原子の番号を入力し，ベクトルの座標は矢印の先端の位置座標である（原子の座標）＋（振動による原子の変位の座標）で与える。現在このプログラムでは振動

```
O          -.644760   -1.629981    -.582474
P          -.589295    -.016148    -.040534
C           .535789    -.392881    1.361116
C           .663644     .383026   -1.337251
O         -2.214362    -.066467     .062388
O          -.790865    1.677458     .376484
H         -1.435421   -2.082190    -.322205
H           .254746   -1.311142    1.859650
H           .540910     .414529    2.081534
H           .723573    -.426063   -2.048605
H           .440663    1.303807   -1.859697
H         -2.556589     .778442     .326942
H          -.005138    2.203785     .364248
C          1.903762    -.530857     .651798
C          1.970152     .481215    -.508876
H          1.992576   -1.533921     .251323
H          2.721968    -.380158    1.348723
H          2.835034     .297897   -1.137496
H          2.086923    1.486138    -.108306
1          -.344760   -1.979981     .417526
2          -.589295    -.016148     .159466
3           .835789    -.292881    1.461116
4           .213644     .533026   -1.337251
5         -2.264362     .333533   -1.587612
6          -.790865    1.577458    1.176484
14         1.953762    -.580857     .301798
15         1.770152     .381215    -.958876
```

図2　$P(OH)_3C_4H_8$ の振動モードの入力データの例

ベクトルと原子をあわせて100個まで扱える。データの入力はテキスト形式のファイルから行えるだけではなく，ワープロや通信ソフト上に表示されているテキストデータのコピーをペーストすることによっても行える。図示した分子図は回転させたり，拡大，縮小したりでき，視角，原子球の大きさを変えてPICT形式でファイルに保存したり，コピーしたりすることができる。また，分子図上で原子を指定することにより，原子間距離，結合角，二面角を計算することができる。表示された原子の結合状態や座標，視角，大きさをテキストデータとして保存できる。このデータを編集することも可能である。

MOLCATの操作はマウスやMacintoshの通常のコマンドキー操作で簡単に行うことができる。例えば，視角を変える場合には，「設定」メニューから「視角」を選び，現れたダイアログボックスに視角を入力し，「はい」ボタンをマウスでクリックする。原子間に結合をつくるには，結合したい原子をクリックして，「結合」メニューから「結合をつくる」を選ぶか，"Command" キーと "B" キーを同時に押すと選んだ順に原子を結合していく。座標軸まわりの回転はメニューを使わず，位置指示器のプローブをマウスで移動することによって指示する。ファイル操作や編集操作以外は操作を取り消すことが可能である。実際にMOLCATをMacintosh上で動作させたときの様子を図3に示す。扱っている分子は $P(OH)_3C_4H_8$ である。画面上部に7個のメニューが示されている。右手の長方形と環が分子を回転させるときの角度を指定するための回転指示器である。カラーディスプレイ上では，長方形は緑色，環は青色である。

表1 MOLCATで可能な操作

メニュー	項目	操作内容	キー
アップル	MOLCATについて	MOLCATのロゴを表示する	
ファイル	開く	ファイルを開きデータを読み込む	O
	絵を保存	表示されている分子図をPICT形式で保存する	
	座標を保存	表示されている分子図の数値データを保存する	S
	振動した座標を保存	振動方向に原子を動かした座標を保存する	
	終了	終了条件を発生する	Q
編集	取り消し	直前の操作を取り消す（再実行も可能）	Z
	コピー	表示されている分子図をクリップボードに移す	C
	ペースト	クリップボードのデータを読み込む	V
設定	原子半径	原子の半径の倍率を変更する	B
	分子半径	分子全体の大きさを変更する	D
	視角	視角を設定し，立体感をつける	
結合	結合をつくる	指定された原子を指定された順に結合する	
	結合を切る	指定された原子間の結合を順に切断する	
	官能基を結合する	水素水素間以外の1.6Å以下の原子間を結合する	
測定	原子間距離	指定された2原子間の距離を計算する	
	結合角	指定された3原子間の角度を計算する	
	二面角	指定された4原子間の二面角を計算する	
移動	X軸にあわせる	指定された2原子がX軸上に来るように移動する	
	XY平面にあわせる	指定された3原子がXY平面上に来るように移動する	
	ベクトルを反転する	振動ベクトルの向きを反転する	
マウス操作	回転指示器	X，Y，Z軸周りの回転操作	
	原子選択	原子の強調表示と，番号の表示	

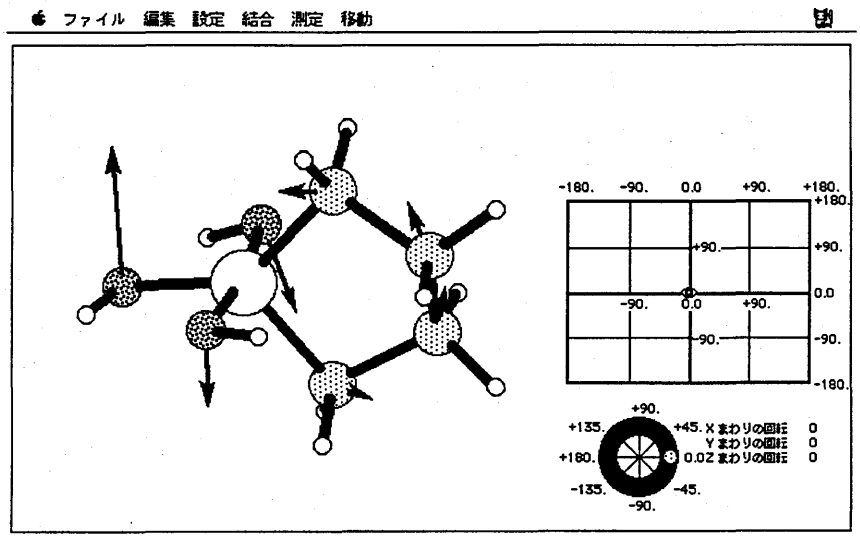


図3 PowerBook170上でMOLCATを動作させたときの画面のハードコピー

5. MOLCAT のプログラム構成

5.1. 変数の宣言

分子の描画のためには，分子内の全原子数，原子番号，原子の空間座標，原子半径，各原子の結合数，それぞれの結合の相手を表す結合表のデータが必要である。これ以外に，MOLCAT では，描画順序，原子の色もデータとしている。また，奥行き感を出すために視角を 0 度以外に指定した場合には，原子の空間座標と原子半径が変換されるので，もとの座標や半径を保存する必要がある。

MOLCAT 内で分子のデータとして定義されている変数を以下に示す。

表 2　分子を定義する座標データ

変数名	役割	型
NATOMS	総原子数	4 バイト整数
NZ(100)	原子番号	4 バイト整数
X(100), Y(100), Z(100)	各原子の描画用空間座標	倍精度実数
XO(100), YO(100), ZO(100)	各原子の実際の空間座標（保存用）	倍精度実数
RATOM(100)	各原子の描画用半径	倍精度実数
RO(100)	各原子の実際の半径	倍精度実数
IB(100)	結合数	2 バイト整数
IBOND(10, 100)	結合の相手の番号	2 バイト整数
NUM(100)	各原子の描画順序	2 バイト整数
ICATOM(100)	原子の色	4 バイト整数

メニュー選択および分子の回転操作の前には描画用変数に実際の空間座標や半径を代入し，操作後の再描画直前に設定された視角に基づいて立体感を出すように描画用変数を操作して計算している。逆に，ファイルを開いたり，ペースト操作等で座標を読み込む場合には，描画用変数に読み込んだ後，実際の変数に値を格納している。

Macintosh では操作の安全性を保証するために操作の取消が可能でなければならない。このために，共通ブロック undo を用意し，以下の変数を用意した。

表 3　共通ブロック undo の内容

変数名	役割	型
iSet	再実行のためのインデックス	2 バイト整数
oldMenu	最後に選択されたメニュー	4 バイト整数
IVAR1, IVAR2	最後の原子半径，視角，回転角	2 バイト整数
XMAX, XMIN, YMAX, YMIN	最後の分子描画領域	倍精度実数

5.2. 主ルーチン

主ルーチンのフローチャートを図 4 に示す。最初のルーチン SystemCheck はシステムのバージョンの確認，コプロセッサの有無の確認，32bitQuickDraw の存在の有無を確認している。これらの確認は Toolbox 関数 Gestalt[7] を用いて行う。各状態について不適切な部分

があればアラートを表示し，ユーザーの選択にしたがってプログラムを続行または終了する。次の EventsInit で各アップルイベントに対する操作を定義したアップルイベントハンドラをとりつける。MenuBarInit ではメニューを構築する。ウィンドウの初期化は Toolbox 関数 NewCWindow を中心とするルーチン群で，ディスプレイの大きさに基づいた PlainDBox タイプのウィンドウを作成する。ウィンドウの大きさから，分子図を描くメモリー上の描画ポート（以下オフスクリーンと呼ぶ）の大きさを決め，MakeGWorld でオフスクリーンを作成する。

パラメーター，座標の初期化では分子データに 0 を代入する。スクラッチファイルとして装置番号31，32に FORM＝'unformatted' のファイル，12に 'formatted' のファイルを割り当てる。'unformatted'ファイルは結合データや座標のデータを取消操作にそなえて保存するのに用いる。'formatted' ファイルは実数，整数，文字が入り交じったデータを画面出力する際に書式を整えるのに用いる。

次の部分はループ構造を持っていて，ループ内で共通ブロック globals の論理型変数 allDone が .true. になるまで無限に動き続ける。Toolbox 関数 WaitNextEvent を用い，定期的にアプリケーションに対する命令が出されていないかを確認する。このような命令をイベントという。このアプリケーションで扱うイベントは，ファインダーから送られるアプリケーションに対する命令であるコア・アップルイベント，マウスボタンの押し下げ（マウスダウンイベント），キーの押し下げ（キーダウンイベント），キーを押し続けること（オートキーイベント），ウィンドウやアプリケーションの切り替え等で生じる画面書き換えの命令（アップデートイベント）である。これらイベントの種類に応じてプログラムの流れを分岐させる。

アップルイベントが実行された場合，Toolbox 関数 AEProcessAppleEven を用い，そのアップルイベントがコア・アップルイベントのどれかを判断し，適切なアップルイベントハ

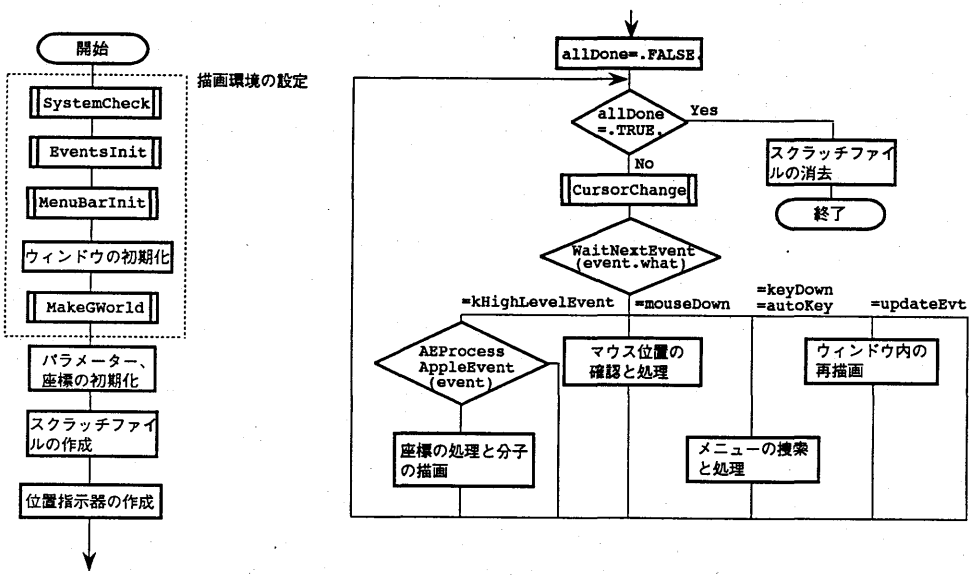


図4 主プログラムの各初期化ルーチン群とイベントループ

ンドラの実行を行う。そのイベントがアプリケーションの開始（OpenApplication）であれば，ファイル選択のダイアログボックスを表示し，書類の開示（OpenDocument）であれば，その書類の名前を調べて OPEN 文を実行する。これら二つの場合では書類を正常に装置番号15番に割り当てることができれば，データの読み込みと分子の描画が行われる。アプリケーションの終了（QuitApplication）であれば allDone を .true. にしてループを脱出する。

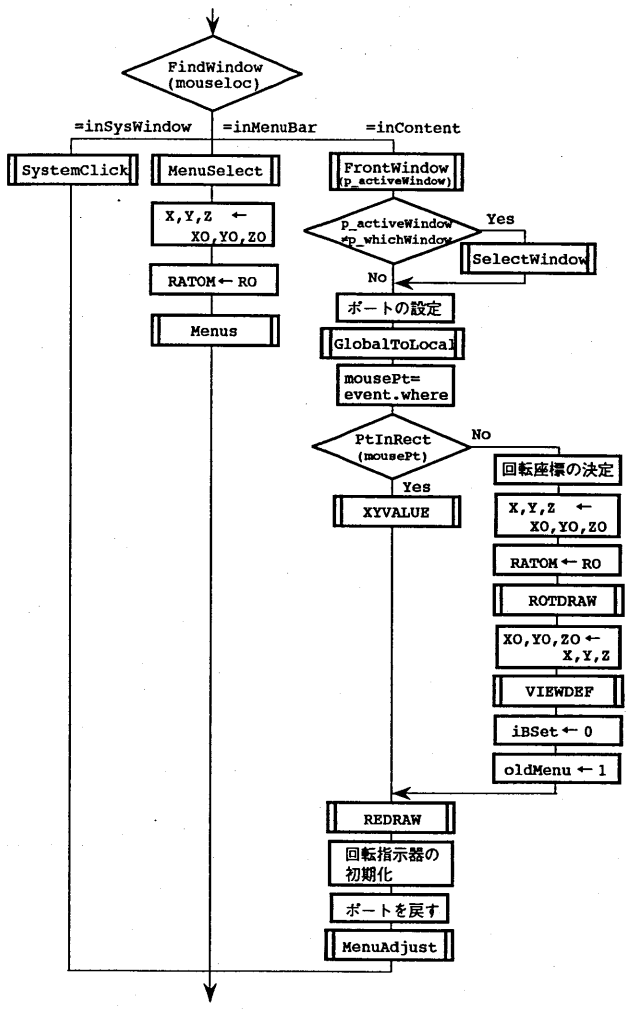


図5　マウスの位置の確認と処理

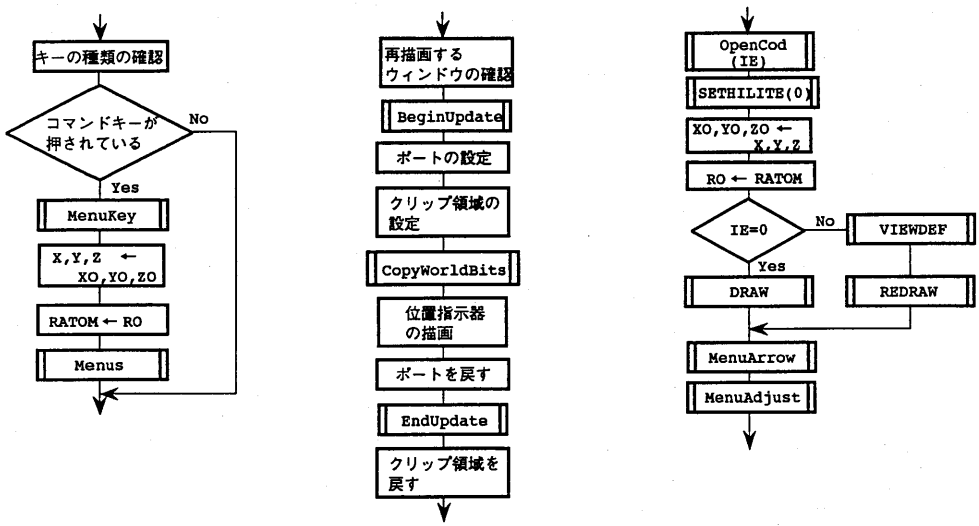
マウスダウンイベントの場合，図5のフローチャートに示したような処理が行われる。マウスボタンが押された場所の位置座標を示す変数 event.what を Toolbox 関数 FindWindow に渡し，座標とウィンドウの位置関係を示す値 mouseloc を得る。mouseloc が inSyswindow，すなわち，マウスボタンが押された場所が MOLCAT のウィンドウやメニューバー以外である場合には，Toolbox 関数 SystemClick を呼び出し，後の処理をファインダーに任せる。inMenuBar である場合，すなわちメニューバーでマウスボタンが押された場合には Toolbox 関数 MenuSelect を呼び出しどの項目が選ばれたかを調べ，その結果を4バイト整数 where に返す。原子の空間座標，半径 XO，YO，ZO，RO を描画用座標 X，Y，Z，RATOM に代入する。項目ごとの操作を決めているルーチン Menus に where を渡し，Menus で選ばれた項目を実行する。mouseloc が inContent すなわちウィンドウ内である場合には，現在最前面にあるウィンドウのポインタ p__activeWindow を調べ，それがマウスボタンの押し下げにより選択されたウィンドウと一致しているかを調べる。一致していない場合には Toolbox 関数 SelectWindow を用いて選択されたウィンドウを最前面に持ってくる。次にポートの設定を行う。描画を行う場合には次のような一連の Toolbox 関数を呼び出す。

```
CALL GetPort(oldPort)
CALL SetPort(VAL4(newPort))
! 描画操作
CALL SetPort(VAL4(oldPort))
```

GetPort は，それまで利用していたポートのポインタを4バイト整数 oldPort に格納する。次の SetPort は，これから描画するポートを描画可能にする。newPort としては，ウィンドウのポインタを用いればよい。描画操作終了後，最初に保存しておいた oldPort にポートを戻す。ポート設定後，マウスの座標を Toolbox ルーチン GlobalToLocal で画面の座標からポートの座標に変換する。次に，マウスが押された位置 mousePt がウィンドウの分子図の領域で押されたかどうかを Toolbox 関数 PtInRect で判断する。分子図の領域で押された場合には，それが，原子を選択しているかどうかを確認するために，ルーチン XYVALUE で選択された原子を探す。それ以外の場合には，回転の指定が行われているかを確認し，指定されている場合には原子座標を描画座標に代入し，描画座標を ROTDRAW で回転したのち描画座標を原子座標に再格納する。立体感をつけるために描画座標をルーチン VIEWDEF で変換する。いずれの場合についてもルーチン REDRAW で分子図を再描画し，回転指示器を初期化したのちポートを戻す。最後にルーチン MenuAdjust でメニューを最適化する。

キーダウンイベント，オートキーイベントの場合，図6a のフローチャートに示したような処理が行われる。入力されたキーの種類を確認し，コマンドキーが押されているかを確認する。コマンドキーが押されている場合には，Toolbox 関数 MenuKey を呼び出し，メニューのどの項目が選ばれたかを調べ，その結果を4バイト整数 where に返す。あとの操作はマウスボタンが押された場合と同じで，選ばれたメニューがルーチン Menu 内で実行される。

アップデートイベントの場合，図6b のフローチャートに示したような処理が行われる。



a メニューの捜索と処理　　b ウィンドウ内の再描画　　c 座標の処理と分子の描画

図6　メニューの捜索と処理，ウィンドウ内の再描画，座標の処理と分子の描画

アップデートの決まった手順に従い，オフスクリーンに保存されている分子図をウィンドウに転送し，回転指示器を再描画する。

5.3. データの読み込み

Macintosh のプログラムでは，ユーザーが指定する限りデータの読み込みが何度でも繰り返される。MOLCAT では「ファイル」メニューから「開く」が選ばれた場合，「編集」メニューから「ペースト」が選ばれた場合，ファインダーで出力ファイルが開かれた場合（OpenDocument イベントが送られた場合）データの読み込みが行われる。これらのデータの読み込みとそれに続く一連の操作を図 6c に示す。データ読み込みルーチン OpenCod は描画に関するすべての値を 0 にしてからデータを読み込み，ファイルの入力形式についての情報を実引数 IE に返す。IE=0 ならば，図 1 のようなフリーフォーマットのテキストデータであるし，IE=1 ならば，MOLCAT の出力座標データである。ルーチン SETHILITE(0) で選択された原子が存在しないようにしたのち，読み込んだ座標データは保存用の変数に代入され，IEの情報にしたがって描画ルーチンが選択実行される。IE=1 の場合にはデータ内で視角が設定されているので，視角を計算するルーチン VIEWDEF を実行してからルーチン REDRAW を実行する。最後に，ルーチン MenuArrow と MenuAdjust を用いてメニューを最適化する。MenuArrow は振動ベクトルがデータ内に存在しない場合に，振動に関するメニューが選択できないようにするルーチンである。これはデータの読み込みの直後のみに行われる。MenuAdjust は，MOLCAT がなんらかの処理を行うたびに呼び出される（図 7）。このルーチンは，原子の数 NATOMS を調べて，0 である場合には「ファイル」メニューの

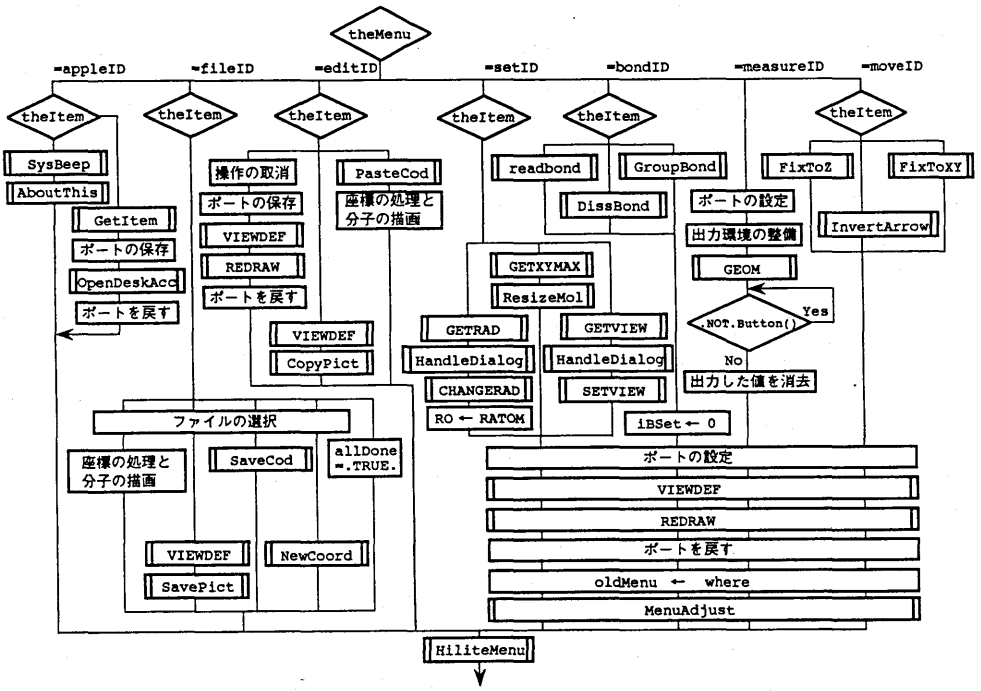


図 7　ルーチン Menus の構成

「開く」と「編集」メニューの「ペースト」以外のメニュー項目が選択できないようにする。そうでない場合には，画面上で選択されている原子の数に応じて，「測定」メニューや「移動」メニューの項目を選択可能にしたり不可能にしたりする。例えば，2個の原子が選択されている場合には「測定」メニューでは「原子間距離」の項目のみを選択可能に，「移動」メニューでは「X軸にあわせる」「XY平面にあわせる」の項目のうち，「X軸にあわせる」を選択可能にする。

5.4. 描画方法

5.4.1. DRAW, REDRAW

ルーチンDRAW，REDRAWは描画環境を整え，ルーチンbonddrawを呼び，画面に分子の図を描く。DRAWはフリーフォーマットデータの読み込みを行った後の描画時に利用され，その他の場合はREDRAWが呼ばれる。DRAWとREDRAWの違いは，DRAWがルーチンWINDOWを呼ぶのに対し，REDRAWはWINDOW2を呼ぶことである。DRAW，WINDOW，WINDOW2のフローチャートを図8のa，b，cにそれぞれ示す。

DRAWの最初のルーチン，FINDMAXXYは，振動ベクトルの先を除く原子のXY座標の最大，最小値を求め，XMAX，YMAX，XMIN，YMINに格納する。これらの変数はルーチンSETXYMAXの中に保存される。次にルーチンSORTが呼ばれる。原子および結合の描画は奥から手前に向かって，すなわちZ座標の小さいものから大きいものに向かって描かねばならないので，SORTは原子を描画する順番に原子の番号を配列NUMに格納する。例えば，2番，3番，1番の順にZ座標が大きくなっていく3原子分子が存在した場合，NUM

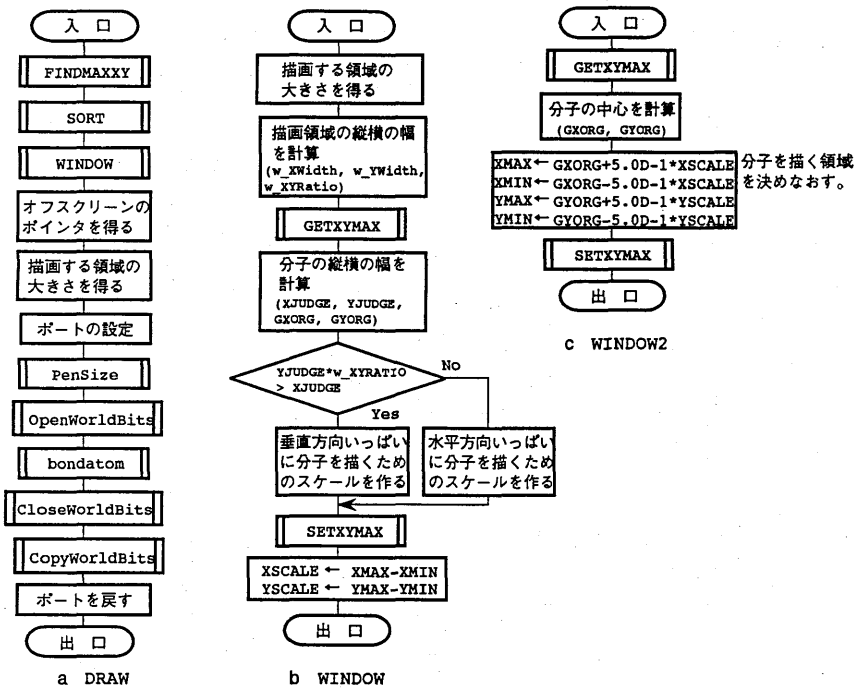
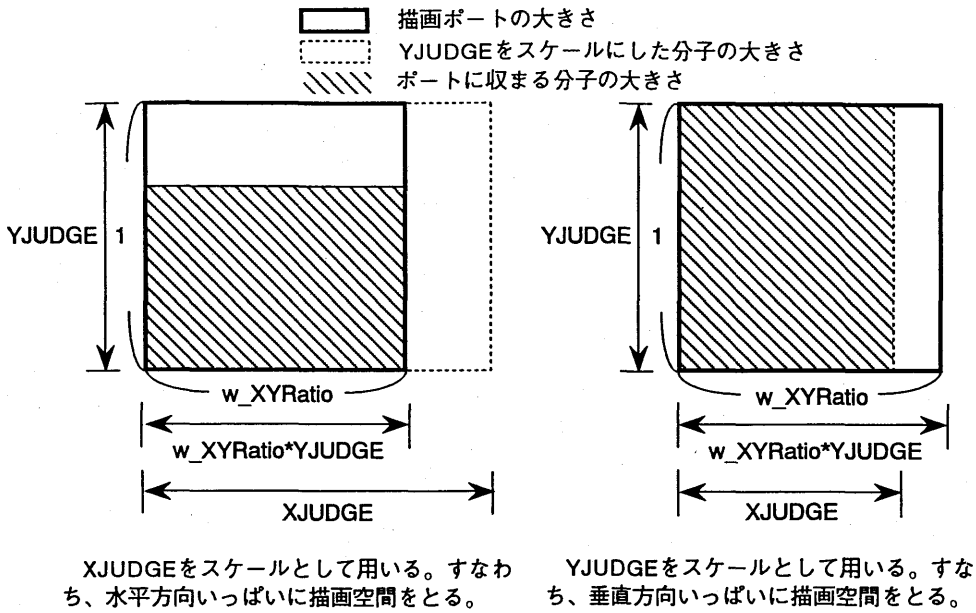


図8 サブルーチンDRAWとDRAWの中のWINDOW，WINDOW2の構成

(1)=2, NUM(2)=3, NUM(3)=1 となる。次のルーチン WINDOW は，SETXYMAX に保存された変数を GETXYMAX で取り出し，描画座標系のスケールを決める。描画ではなるべくウィンドウを有効利用できることが望ましい。このため，上下左右各0.8Åに対応するだけのマージンを確保しつつウィンドウの大きさいっぱいに絵を描くために，以下の計算を行う。

```
w_XYRatio = (w.right - w.left)/(w.bottom - w.top)
XJUDGE = MAX((XMAX-XMIN), 2.0D0) + 1.6D0
YJUDGE = MAX((YMAX-YMIN), 2.0D0) + 1.6D0
```



XJUDGEをスケールとして用いる。すなわち、水平方向いっぱいに描画空間をとる。

YJUDGEをスケールとして用いる。すなわち、垂直方向いっぱいに描画空間をとる。

図9　ルーチン WINDOW による描画スケールの決定の仕方

ここで，w.right，w.left，w.bottom，w.top は，描画部分の右，左，下，上をピクセル単位で示した大きさである。w__XYRatio は描画部分の横と縦の比を示しており，XJUDGE，YJUDGE は分子の X 方向，Y 方向の幅である。MAX は FORTRAN 組み込み関数で，引数の最大値を与える。Z 軸上に原子が並んだ直線分子にみられるように，XMAX-XMIN=YMAX-YMIN=0 になって幅が定義できなくなることを避けるために，分子の幅は2.0Å相当以下にならないとしてある。YJUDGE*w__XYRatio と XJUDGE を比較し，YJUDGE をスケールとして使えるかどうかを判断する。

中心座標 GXORG，GYORG は，次式で決める。

```
GXORG = ( XMAX + XMIN ) * 5.0D-1
GYORG = ( YMAX + YMIN ) * 5.0D-1
```

それぞれのスケールを用いて描画座標の最大，最小値を求める。垂直方向いっぱいにとる場合には，

```
XMAX = GXORG + 5.0D-1*YJUDGE*w_XYRatio
XMIN = GXORG - 5.0D-1*YJUDGE*w_XYRatio
YMAX = GYORG + 5.0D-1*YJUDGE
YMIN = GYORG - 5.0D-1*YJUDGE
```

水平方向いっぱいにとる場合には,

```
XMAX = GXORG + 5.0D-1*XJUDGE
XMIN = GXORG - 5.0D-1*XJUDGE
YMAX = GYORG + 5.0D-1*XJUDGE/w_XYRatio
YMIN = GYORG - 5.0D-1*XJUDGE/w_XYRatio
```

それぞれの値をSETXYMAXに保存し，XSCALE＝XMAX-XMIN，YSCALE＝YMAX-YMINをWINDOWに保存する。WINDOW2は，これらXSCALE，YSCALEを利用し，SCALEを変えることなく分子図が描画ポートの中心にくるように中心を平行移動して再描画する。これは，原点付近に分子の中心が存在しないので回転操作を行ったのちに座標が大きく変わってしまうような場合に図がポートの外に出てしまうのを避けるためである。

スケールを決めたのち，ウィンドウ構造体の要素windowRefConに保存してあるオフスクリーンのポインタをToolbox関数GetWRefConで得る。描画する領域の大きさをルーチンBoundsRectで決定する。Toolbox関数GetGWorldで古いポートを保存したのち，OpenWorldBits[8]でオフスクリーンのメモリー位置をロックし，これをポートに設定，初期化する。後述するbondatomで分子を描画し，CloseWorldBits[8]でメモリーの位置のロックを解除し，CopyWorldBits[8]でオフスクリーンからウィンドウにビットマップを転送する。OpenWorldBits，CloseWorldBitsおよびCopyWorldBitsはCで記述した。

5.4.2. bondatom

bondatomは，座標を元にして，あらかじめ指定されたポートに分子の図を描くルーチンである。bondatomのフローチャートを図10に示す。

bondatomはNUMで指定された番号順に原子を示す円および鍵，結合を描く。I番目の描画についてみる。I番目の原子番号をNUM(I)から得て，JNUMとする。配列IHILITEは，マウスで現在選択されている原子の番号を保存していて，その0番目の要素IHILITE(0)は選択されている原子の個数を示している。IHILITEにJNUMに一致するものがあれば，原子を強調するために線を太くする。Toolbox関数PenSize(VAL2(4)，VAL2(4))でペン幅を水平方向に4ピクセル，垂直方向に4ピクセルと指定する。次に，描画しようとしているものが原子なのか矢印なのかを知るために，原子番号NZ(JNUM)が100以下であるかどうかを調べる。100以下の場合は原子であるので，ルーチンCIRCLEにX，Y座標と半径Rを渡して円を描く。次に，結合を描く。IB(JNUM)にこの原子の結合数（振動ベクトルも結合と数える）が格納されているので結合の相手を示すIBOND(K,JNUM)をK＝1～IB(JNUM)まで検索する。K番目の結合相手の番号をKFINとする。すべての描画は奥から手前に向かって行うという原則に基づき，JNUM番目の原子のZ座標Z(JNUM)とKFIN番目のZ座標を比較し，Z(KFIN)≧Z(JNUM)が成立する，すなわち，KFIN番目の原子または矢印が手前にあるときだけ結合の描画を行う。描画を行う場合にはKFINの原子番号NZ(KFIN)を調べ，これが100以下の場合はbonddrawを呼び出し，そうでない場合には矢

印であるので ARLINE を呼び出す。一方，NZ(JNUM) が100より大きい場合は矢印であるので，JFIN=IBOND(1,JNUM) とし，矢印の頭を描くルーチン ARROW を呼び出す。ARROW は引数として与えられた X(JFIN), Y(JFIN), X(JNUM), Y(JNUM) を利用して矢印の向きを決め，描く。原子のときと同じように，Z(JFIN)，と Z(JNUM) を比較し，鏃を結ぶ線である ARLINE を呼び出す。最後に DrawNum を呼び出し，IHILITE の選択されている原子の番号を表示する。

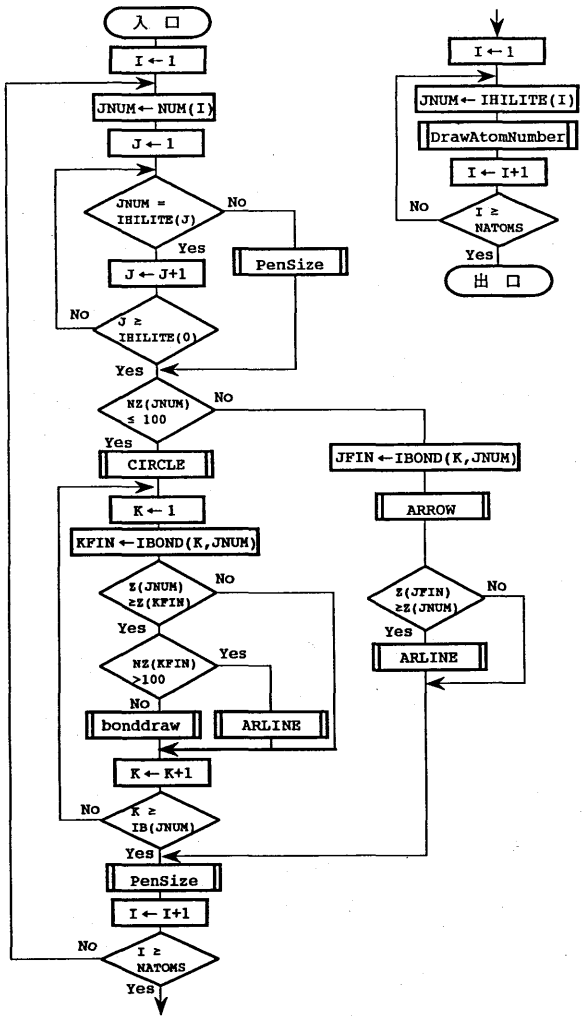


図10　描画サブルーチン bondatom の構成

5.4.3. CIRCLE

ルーチン CIRCLE は与えられた X，Y 座標に半径 R の円を描く。フローチャートを図11a に示す。X，Y 座標を関数 IXCTOB, IYCTOB を用いてウィンドウ，またはオフスクリーンのピクセル単位の座標系に変換する。描画空間が左 w.left，右 w.right，上 w.top，下 w.bottom ピクセルの長方形であるとし，描画座標の X，Y の最大，最小値を XMAX，XMIN，YMAX，YMIN とすると，X 座標の変換値 IXCTOB は，

```
RATIO = DBLE(w.right - w.left)/(XMAX - XMIN)
IXCTOB = NINT( (X - XMIN)*RATIO ) + w.left
```

Y座標の変換値 IYCTOB は，

```
RATIO = DBLE(w.top - w.bottom)/(YMAX - YMIN)
IYCTOB = NINT( (Y - YMIN)*RATIO ) + w.bottom
```

である。これらの値から，円に外接する長方形 elp を決定する。Toolbox 関数 PaintOval (elp) で，ICATOM で指定した色で塗りつぶした円を描いたのち，FrameOval(elp) で黒い輪郭線を描く。

5.4.4. bonddraw

ルーチン bonddraw のフローチャートを図11b に示す。bonddraw は指定された原子の間に結合を描く。結合は KINT 番目の原子から KFIN 番目の原子に向かって引かれる。最初に

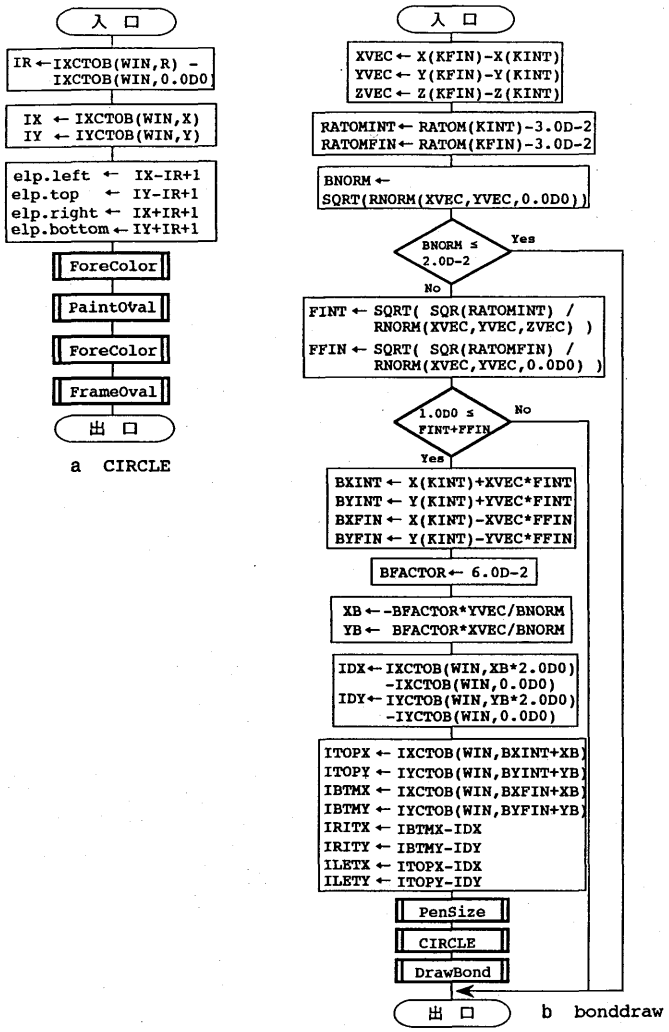


図11 描画サブルーチン CIRCLE と bonddraw

結合の向きを与えるベクトル O_1O_2 (XVEC, YVEC, ZVEC) を計算する。次に，結合の始点と終点になる原子の半径をそれぞれ RATOMINT と RATOMFIN に代入する。RATOM の実際の値よりも0.03Å相当だけ小さくすることで描画する結合が長くなり，描画時に結合が原子から離れてしまうのを避けることができる。次に，XY 平面に投影したときの結合の長さ BNORM を決定する。RNORM は引数X，Y，Z の二乗和を与える文関数である。BNORM が0.02 Å 相当以下の場合には結合は描かない。次に原子を球と，結合を線と考えたときの交点を求める。始点側の原子は奥にあるため描画時に球と線の交点の位置から結合を描画する必要があるのに対し，終点側の原子は手前にあって交点の位置が見えないので描画時に結合を球の縁まで描画すればよい。ベクトル O_1O_2 の長さを1としたとき，それぞれの原子の中心から交点までの長さ O_1P_1，O_2P_2 は FINT，FFIN で与えられる。ここで SQR は引数 X の二乗を与える文関数である。

FINT，FFIN の和が1以上の場合には結合を描くべき部分よりも描かない部分の方が大きくなるので結合を描画する必要がない。1未満の場合についてのみ結合の始点と終点の XY座標 を計算し，BXINT，BYINT，BXFIN，BYFIN とする。

次に結合を太くするために，結合の中心線からの幅 BFACTOR を0.06Å にする。結合線の法線ベクトル（XB, YB）を決め，ポート上でのピクセル単位のベクトル（IDX, IDY）を計算する。これらの値を用いて結合がつくる矩形の頂点の座標 ITOPX，ITOPY，IBTMX，IBTMY，IRITX，IRITY，ILFTX，ILFTY を計算する。これらのデータを元にして結合を描く。結合の始点を中心とする半径 BFACTOR の円をルーチン CIRCLE を呼び出して描く。

次に，ルーチン DrawBond を呼び出して結合の長方形を描く。

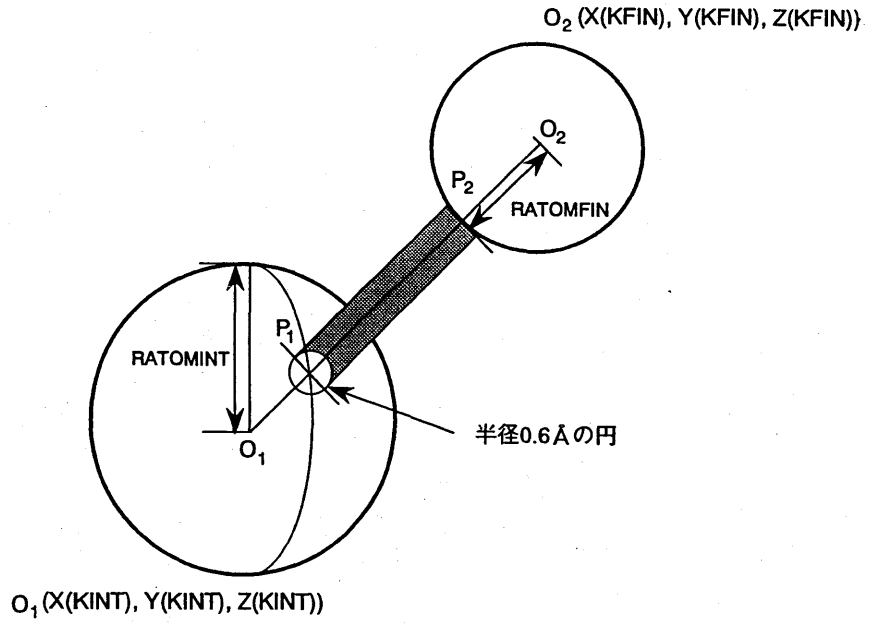


図12　原子間の結合

5.5　描画した分子に対する操作

描画した分子に対して，メニューに登録されている原子半径の変更，分子半径の変更，視角の設定，結合の編集，結合長などの幾何学的数値の計算，分子の移動の各操作と，回転指示器を使用して回転操作を行うことができる。ここでは，これらの操作の一部について説明する。

5.5.1.　原子の選択

結合を編集したり，原子間の幾何学的情報を得たり，ある結合を X 軸に合わせるなどの移動操作を行ったりするためにはどの原子について操作を行うかを指定する必要がある。図2に現れるルーチン XYVALUE はどの原子をマウスで選択したのかを調べるルーチンである。フローチャートを図13a に示す。関数 BITTOXC，BITTOYC は関数 IXCTOB，IYCTOB の逆関数であり，マウスのウィンドウ上での座標 IPOINT(IPOINT.h=IX，IPOINT.v=IY) に対応する X，Y 座標（XC,YC）を次式で計算して与える。X 座標については

```
RATIO = (XMAX - XMIN)/(w.right - w.left)
XC = XMIN + DBLE (IX - w.left)*RATIO
```

Y 座標については

```
RATIO = (YMAX - YMIN)/(w.top - w.bottom)
YC = YMAX + DBLE (IY - w.top)*RATIO
```

ここで使用した変数は，関数 IXCTOB，IYCTOB と同じである。次に各原子とこの座標の位置関係について調べる。ある原子が選択されているとは，その原子がつくる円の領域内に座標（XC,YC）が含まれていることを意味する。すなわち，座標（XC,YC）とその原子の

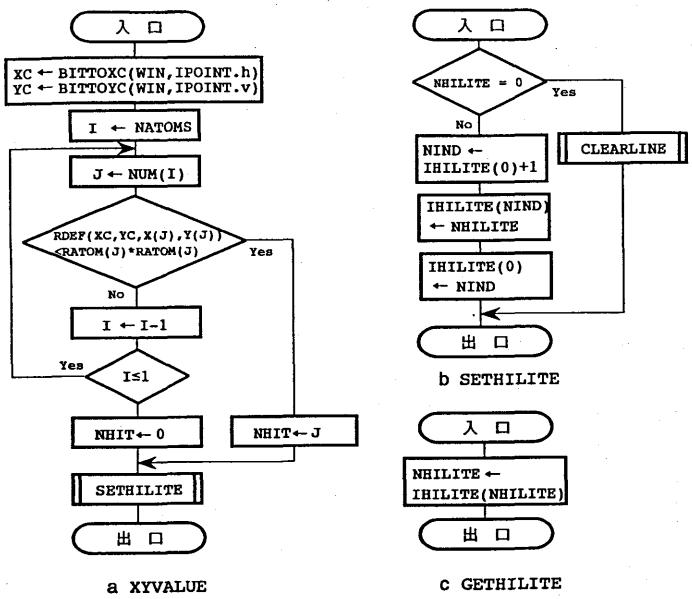


図13 ルーチン XYVALUE, SETHILITE と GETHILITE の構造

中心座標（X,Y）との距離がその原子の半径よりも小さければよい。ただし，原子が互いに重なり合っていて，このような条件を満たす原子が複数個存在する場合には，一番手前にあるものが選択されたと考える。この考え方にしたがって，原子を描く順序と逆の順序で手前から NATOMS 個の原子について調べていく。RDEF は距離の二乗を与える文関数で，距離と半径を比較するかわりに

表4 $P(OH)_3C_4H_8$ の結合表

原子番号	元素記号	IB	IBOND					
1	O	3	20	7	2	0	0	0
2	P	6	21	4	3	6	1	5
3	C	5	22	8	9	14	2	0
4	C	5	23	10	11	15	2	0
5	O	3	24	12	2	0	0	0
6	O	3	25	13	2	0	0	0
7	H	1	1	0	0	0	0	0
8	H	1	3	0	0	0	0	0
9	H	1	3	0	0	0	0	0
10	H	1	4	0	0	0	0	0
11	H	1	4	0	0	0	0	0
12	H	1	5	0	0	0	0	0
13	H	1	6	0	0	0	0	0
14	C	5	26	3	15	16	17	0
15	C	5	27	4	14	18	19	0
16	H	1	14	0	0	0	0	0
17	H	1	14	0	0	0	0	0
18	H	1	15	0	0	0	0	0
19	H	1	15	0	0	0	0	0
20	1	1	1	0	0	0	0	0
21	2	1	2	0	0	0	0	0
22	3	1	3	0	0	0	0	0
23	4	1	4	0	0	0	0	0
24	5	1	5	0	0	0	0	0
25	6	1	6	0	0	0	0	0
26	14	1	14	0	0	0	0	0
27	15	1	15	0	0	0	0	0

おのおのの二乗について比較し，もし条件が満たされれば選択された原子番号 NHIT に J を代入し，ループを脱出する。条件を満たす原子が存在しない場合は選択された原子は存在しないので，NHIT に 0 を代入する。次のルーチン SETHILITE は選択された原子の番号を選択している順に保存している配列 IHILITE に NHIT を保存する。NHIT が 0 のときは，配列 IHILITE の要素をすべて 0 にする。SETHILITE のフローチャートを図13b に示す。IHILITE は 0 番から始まる配列で，0 番目の要素は重複を含む選択されている原子の個数であり，その他の要素は1番から選択された順に原子の番号を保存している。NHIT が 0 のときはルーチン CLEARLINE が配列 IHILITE の要素をすべて0にし，その他の場合には IHILITE(0) の数を 1 増やし，配列の最後尾に NHIT を付け加える。

5.5.2. 結合の編集

各原子にいくつの結合が存在するかは配列 IB に，原子と原子の間の結合がどこに存在するかは二次元配列 IBOND に格納されている。例として，$P(OH)_3C_4H_8$ の MOLCAT 上での結合のデータの様子を表 4 に示す。

これらの結合についての情報を変更することを「結合を編集する」と呼ぶことにする。MOLCAT には選択した順に「結合をつくる」，選択した順に「結合を切る」，選択とは無関係に原子間距離の情報等から結合をつくる「官能基を結合する」の3種類の結合編集のメニューがある。これらの中から「結合をつくる」について説明する。

結合をつくるルーチン readbond のフローチャートを図14に示す。readbond は配列 IHILITE の要素を参照しながら結合を付け加えていくルーチンである。最初に NUMHIT に 0 を代入し，ルーチン GETHILITE に渡す。GETHILITE は図13c のフローチャートに示すように，配列 IHILITE の NHIT（実引数名 NUMHIT）番目の要素を参照し，NHIT にその要素を返す。今回は NHIT が 0 であるので選択されている原子の数 IHILITE(0) が NUMHIT に返る。結合編集操作の取り消しを可能にするために，いままでの結合についてのデータをスクラッチファイルに保存するルーチン BackupBond を実行する。I を 1～NUMHIT-1 番目まで変化させ，N1＝IHILITE(I) 番の原子と N2＝IHILITE(I+1) 番の原子との間に結合をつくる。GETHILITE を用いて N1，N2 を取り出した後，それぞれの結合数が10になっていないかを調べる。10である場合にはこれ以上結合を新たにつくることはできないので何もせず，I+1，I+2 番目の要素に対する操作に移る。次に，N1，N2 間に結合が既に存在しないか確かめるために IBOND(J,N1) を J＝1～IB(N1) について N2 が存在しないかどうかを調べる。N2 が存在した場合には何もせず，I+

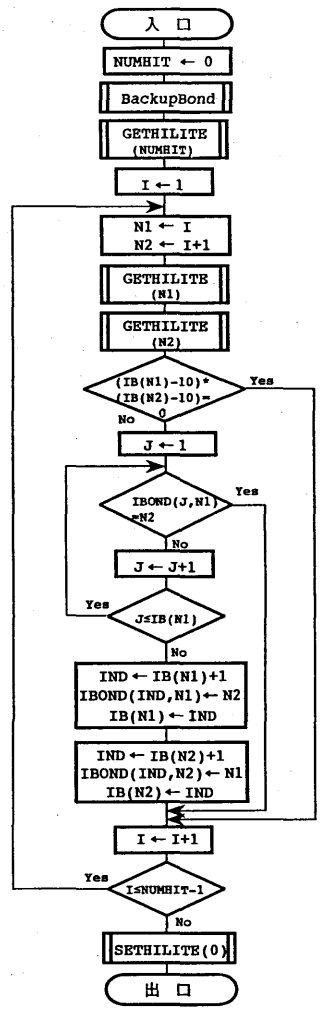


図14　作成する結合の指定

1，I+2番目の要素に対する操作に移る。N2が存在しない場合には，結合をつくるための操作に移る。現在の結合数IB(N1)に1を加え，INDに代入する。IBOND(IND,N1)にN2を代入し，新しい結合数INDをIB(N1)に格納する。N1に対して行った同じ操作をN2に対しても行う。この場合はN1が存在しないことが明らかであるので，INDにIB(N2)を代入する操作から始める。IBOND(IND,N2)にN1を代入し，新しい結合数INDをIB(N2)に格納する。これらの操作が終わった後にI+1，I+2番目の要素に対する操作に移る。IHILITEのすべての要素を参照し終えたのち，SETHILITE(0)でIHILITEのすべての要素を0にする。

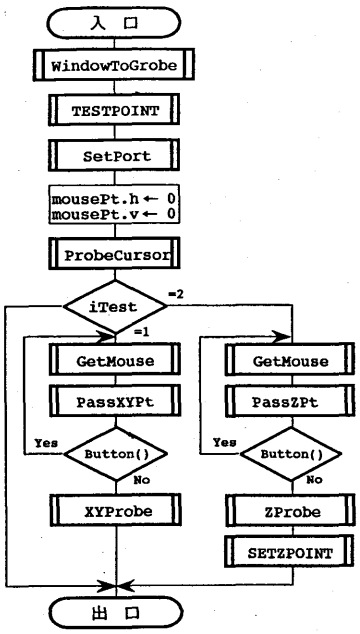


図15 ルーチン GlobePoint

5.5.3. 回転指示器を用いた分子の回転

分子回転を指示するために回転指示器を用意した(図3右下)。プローブが目玉の形をした緑色の長方形の地図状のものでX，Y軸まわりの回転を指示し，プローブが黄色い円形をした青い環状のものでZ軸まわりの回転を指示する。どちらの場合も，プローブをマウスで選択し，プローブを目的の位置に移動することによって回転角を設定することができる。ここでは，特にX，Y座標の回転指示のルーチンについて説明する。図15，16に回転指示のために使う主なルーチンのフローチャートを示す。

図5の「回転座標の決定」で呼ばれるルーチンは

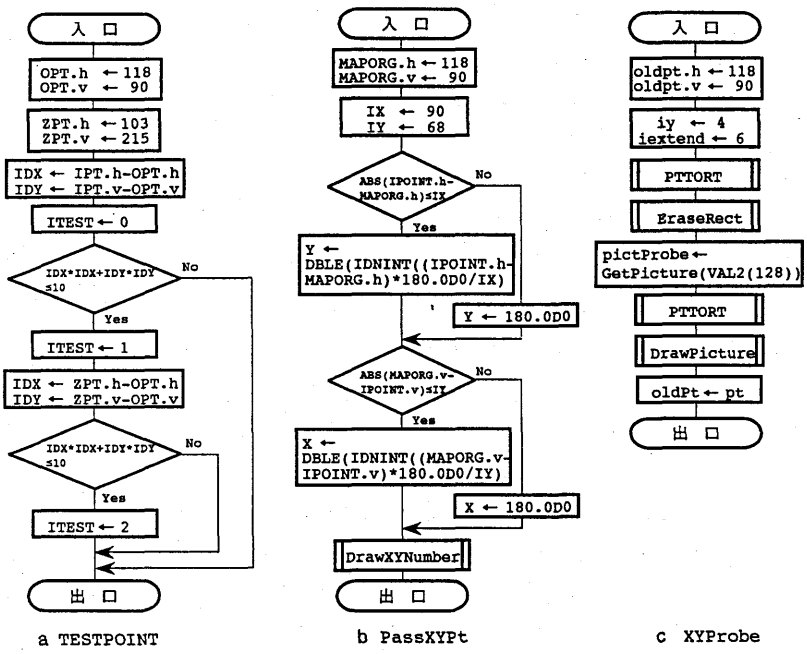


図16 ルーチン TESTPOINT, PassXYPt, XYProbe

SetGlobeOrigin, GlobePoint, SetOrigin の3個である。SetGlobeOrigin はウィンドウのポートの座標の原点を回転指示器の計算がやりやすいように Toolbox 関数 SetOrigin を用いて平行移動するものである。このプログラムでは，ウィンドウのつくる長方形情報を与える変数を w として

```
dh = w.right - w.left -240
dv = w.bottom - w.top - 270
```

で水平，垂直方向の移動の幅 dh と dv を決め，

```
CALL SetOrigin(VAL2(-dh),VAL2(-dv))
```

で原点をウィンドウの右下の端から240ピクセル，左270ピクセル上の位置に移動している。GlobePoint で回転座標を決めた後は，SetOrigin(VAL2(0), VAL2(0)) でウィンドウのポートの左上が原点になる座標系に戻す。

回転座標を決めるルーチン GlobePoint は最初にルーチン WindowToGlobe で現在のマウスの位置を回転指示器用に設定しなおした座標系に変換する。マウス位置 mousePt の変換は dh, dv を用いて

```
mousePt.h = mousePt.h - dh
mousePt.v = mousePt.v - dv
```

で与えられる。このマウス位置が二つの回転指示器のどちらのプローブを捕捉しているかを TESTPOINT で調べ，結果を iTest に返す。iTest の値に応じたカーソルをルーチン Probe Cursor で作成する。iTest の値によってプログラムの流れを分岐させ，XY 軸まわりまたは Z軸まわりの回転を指示する。XY 軸まわりの回転では，Toolbox 関数 GetMouse で現在のマウス位置を受け取り，ルーチン PassXYPt で回転指示器上での座標を計算し，計算した値を表示する。この2個のルーチンはマウスボタンが押されている限り，すなわち，プローブを捕捉している限り繰返し実行される。マウスボタンが放されると，ルーチン XYProbe でプローブの再描画を行う。

ルーチン GrobePoint が呼び出しているルーチンについて以下に説明する。

プローブの捕捉状況を解析するルーチン TESTPOINT は，変数 OPT, ZPT を保存している。

これらの値は他のプログラム単位から SETORGPOINT, SETZPOINT で変更したり，GETORGPOINT, GETZPOINT で参照したりすることができる。OPT は XY 軸の回転指示器の原点（0°, 0°）の座標に，ZPT は Z 軸の回転指示器の原点0°の座標に初期化されている。この値は DATA 文で初期化されているので，このルーチンが一回目に呼ばれた時にのみこの値になり，二回目以降にルーチンが呼ばれた場合にはルーチン内で計算された値が利用される。このルーチンは，ルーチン内に保存している XY と Z の各プローブの古い位置と現在のマウス位置の距離の二乗を計算し，いずれかが10以下の場合にはプローブを捕捉したとみなす。XY のプローブを捕捉している場合には1，Z のプローブを捕捉している場合には2，その他の場合には0を iTest に返す。

回転指示器上での座標を計算するルーチン PassXYPt は，前述したように，マウス位置から回転角を計算し，その角度を画面の「X まわりの回転」と「Y まわりの回転」の後ろに出力する。回転時に実際に原子が動く方向と回転軸は90度ずれているので，マウスの動きと

原子の動きを連動させて直感的に分かりやすくするために，水平方向にY軸まわりの回転，垂直方向にX軸まわりの回転を指定するように指示器の座標を設定した。また，マウス位置が指示器の外に出てしまった場合には，指示した値が±180度を越えないようにする。このルーチンは計算した回転角X，Yを保存していて，その初期値は0.0である。これらの回転角の計算を行うために次の操作を行う。変数MAPORGは回転指示器の原点（0°，0°）の座標を与える。IXはY軸まわりの回転について回転指示器上での0～180度の幅をピクセル単位で与える。IYはX軸まわりの回転について回転指示器上での0～180度の幅をピクセル単位で与える。最初にY軸まわりの回転について，マウスが指示器の左外または右外に出てしまっていないかを確認する。外に出てしまっている場合には180度にする。そうでない場合にはY軸まわりの回転角を計算し，Yに格納する。次にX軸まわりの回転について同様の操作を行い，回転角を計算する。計算した値はルーチンDrawXYNumberを用いて出力する。

ルーチンXYProbeは次のようにしてプローブを再描画する。このルーチンは，変数oldPtを保存している。oldPtは回転指示器の原点（0°，0°）の座標にDATA文で初期化されているので，このルーチンが一回目に呼ばれた時にのみこの値になり，二回目以降にルーチンが呼ばれた場合にはルーチン内で計算された値が利用される。次に描画するプローブの大きさを設定するために，プローブを包む大きさの長方形の幅の半分の値を水平方向についてはiyに，垂直方向についてはiextendに与える。oldPtを中心とする水平方向にiy×2，垂直方向にiextend×2の幅の長方形をルーチンPTTORTで計算し，その長方形内に存在する古いプローブをToolbox関数EraseRectで消去する。次に，新しい位置にプローブを描くために，Toolboxルーチン GetPictureを用いて，あらかじめPICTリソース内にリソースID128番で保存してある目玉の絵を取り出し，4バイト整数pictProbeにこの絵のハンドルを返す。現在のマウス位置を中心とする水平方向にiy×2，垂直方向にiextend×2の幅の長方形をルーチンPTTORTで計算し，得られた長方形内に新しいプローブをToolbox関数DrawPictureで描き，oldPtに現在のマウス位置を代入する。

ここで計算された値を用いてX軸まわり，Y軸まわり，Z軸まわりの順に回転操作をおこなう。

6. MOLCATの応用

前節までに述べてきたMOLCATを実際の分子系に適用する。ここでは，先にMolecular Editorを用いた振動ベクトルの描画との比較のために，同じ$P(OH)_3C_4H_8$の分子振動を取り上げた。入力データは図2に示したものである。結果は図17に示されている。比較しやすいように，図1と同じ角度から分子を表示してある。分子内の各原子がどのような基準振動モードのもとで振動しているかが良く分かる。MOLCATでは振動ベクトルを矢印で表現して

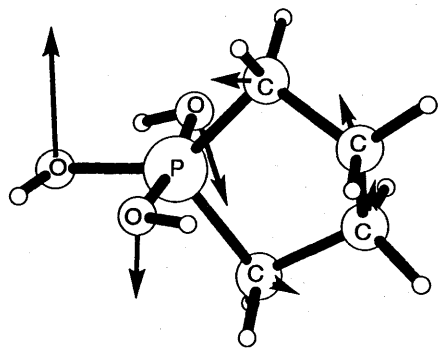


図17 MOLCATを用いて描いた分子振動の様子。矢印は振動ベクトルを表す。

いるので，分子振動を表した図であることが直感的に分かる。これに対して，●と長方形で振動ベクトルを表す方法は，操作している本人が振動ベクトルと認識することによってのみ意味が出てくるものであり，第三者には充分な説明を与えなければ何を表現したものか理解し難いと思われる。本論文の図はカラーでないためにMOLCAT図における結合と矢印がどちらも黒になってしまう。しかし，MOLCAT自体はカラー対応であるため，スクリーン上もしくはカラープリンターでは全く問題ない。入力データを操作するだけで，化学反応の追跡時の各原子に加わる力の表現に矢印を用いることができることなど，単に分子振動の表示だけの目的以外にも広く応用可能なプログラムであると思われる。

7. まとめ

今回我々は，Macintosh上で動作し，分子構造および振動ベクトルをリアルタイムで表示するソフトウェアMOLCATを開発し，アルゴリズムと適用例について解説した。プログラムは主としてFORTRANでコーディングし，Macintoshのシステムが提供するToolbox関数を利用してユーザーが使用しやすいものになるようにした。MOLCATは特に非経験的分子軌道法計算の結果を取り扱うのに便利である。このソフトウェアはフリーウェアとして公開した。

本研究の一部の遂行に際し，文部省科学研究費補助金「重点領域研究　化学反応理論」による援助を受けた。

参考文献

(1) 藤永茂，"分子軌道法"，岩波書店(1980); W. J. Hehre, L. Radom, P. v. R. Schleyer, and J. A. Pople, "Ab initio Molecular Orbital Theory", John Wiley & Sons, Inc (1986); A. Szabo, and N. S. Ostlund, "Modern Quantum Chemistry", McGraw-Hill Publishing Company (1989); "Computational Advances in Organic Chemistry : Molecular Structure and Reactivity", edited by C. Ogretir, and I. G. Csizmadia, Kluwer Academic Publishers (1991); " Lecture Notes in Quantum Chemistry" edited by B. O. Roos, Springer-Verlag (1992); J. B. Foresman, and A. Frisch, "Exploring Chemistry with Electronic Structure Methods", Gaussian, Inc.(1993); " Lecture Notes in Quantum Chemistry II" edited by B. O. Roos, Springer-Verlag (1994)

(2) R. Wargo, A. Smith, and D. McFerren, "Molecular Editor", ver. 1.2, Drexel University (1989)

(3) 筒井祐子，和佐田裕昭，"分子構造表示プログラム「MOLCAT」を利用した分子軌道法計算の効率的な進め方"，名古屋大学大型計算機センターニュース，Vol.26, No.1, P.61(1995).; Yuko Tsutsui, and Hiroaki Wasada, Chem. Lett., 517 (1995)

(4) "MacFortranII", ver. 3.4, Absoft, Corp., Rochester Hills MI (1995)

(5) "MPW C", ver. 3.3.2, Apple Computer, Inc. (1993)

(6) M. J. Frisch, G. W. Trucks, M. Head-Gordon, P. M. W. Gill, M. W. Wong, J. B.

Foresman, B. G. Johnson, H. B. Schlegel, M. A. Robb, E. S. Replogle, R. Gomperts, J. L. Andres, K. Raghavachari, J. S. Binkley, C. Gonzalez, R. L. Martin, D. J. Fox, D. J. Defrees, J. Baker, J. J. P. Stewart, and J. A. Pople, "Gaussian 92", Revision C, Gaussian, Inc., Pittsburgh PA (1992)

(7) Toolbox 関数の詳細については，アップルコンピューター株式会社，"Inside Macintosh（日本語版）"，Vol.1-6，星雲社（1992）を参照.

(8) D. Mark, C. Reed, "Macintosh C プログラミング I, II"，トッパン（1993）

(9) 小池邦人，"改定版 TOOLBOX 100 の定石"，アスキー出版局（1993）