■解説 (VISION Vol. 8, 1996)

視覚研究のための実験環境：Shell & MacGlib

塩入　諭

千葉大学 工学部 画像工学科
〒263 千葉市稲毛区弥生町 1-33

1．はじめに

Shell & MacGlib（Shell と MacGlib は独立したサブルーチンのパッケージであるが，ここでは両者を含めて Shell と呼ぶことにする）は，Cavanagh 教授（ハーバード大）の研究室において開発された，Macintosh 用の視覚実験プログラムの開発支援ソフトである．開発者は Raynald Comtois であるが，その原型は Cavanagh 教授自身のミニコンでの画像処理システムの制御用プログラムにある．制御用のコンピュータを Macintosh に代えた時点で Comtois 氏が Shell の最初のバージョンを作ったことになる．その時は，刺激呈示はまだ Macintosh ではなくそれまで使われていた画像処理システムであった．これは，ちょうど私がモントリオール大学で Cavanagh 教授のポスドクをしていた時であったが，Macintosh でプログラムすることができる用になりエディタの使いやすさのため，プログラミングにかかる時間が大分短縮された（少なくともそう感じた）．その後刺激も Macintosh のディスプレイとなり，Macintosh 用の実験プログラム開発のためのソフトウエアとして提供されることとなった．

私自身は千葉大に職を得てからこのシステムを使用し始めたが（モントリオール大では，Macintosh のディスプレイは使用していなかった），それはなじみがあったからということもあるが，むしろ比較的安価に，自由度の高く十分な精度の実験刺激を，しかも比較的手軽に作ることができるということを考えてであった．現在はコンピュータを取り巻く環境は大きく変わっているが，いまでも非常に便利な道具のひとつであることはかわりはない．以下，Shell について簡単な紹介をするが，コンピュータをある程度知っている人を対象にして書いている．詳しい人には物足りなく，全くなじみのない方には説明不足になっていると思うのだが，これはこのソフトの性格上避けにくいものであることをご理解いただき，ご容赦願いたい．

2．Shell の概略

このソフトウエアの働きは，大きく２つに分けられる．ひとつは，実験の各種の変数（例えば，グレーティングの周波数，コントラスト，色など）をマウスで（あるいは数値を入力して）自由に変化することを可能にするなどの環境を提供することで，もうひとつは，Macintosh のシステムを通さずに直接ビデオカードの制御を行い刺激画像の作成や処理速度，また時間の制御を簡単にすることである．前者はShellと呼ばれるプログラムが担当し，後者は Macglib という関数ライブラリーによって実現される．使用者はこの２つの機能を，利用しながら，自身で実験プログラムを作るわけである．このシステムでは視覚実験用のプログラム環境を提供しているだけであるので，そのまま，なにかに使えるものではない．したがってどのような刺激が使え，どのような実験ができるかについては，使用者のプログラムとハードウエアしだいであるといえる．

使用環境はマック II 以降，システム 6.0.5 以降で使える．また，Think C でプログラムを書くので，Shell とは別に Think C を持っている

必要がある．実験環境としては，制御用のディスプレイと刺激呈示用のディスプレイの2台を用い刺激呈示は256色のルックアップテーブル方式（RGB各色は8ビットの解像度）での使用を想定している．もっとも便利な環境はいまのところApple社の8・24ビデオカードを使うことである．このカードを使うと256色を3ページ分取れるのでかなり自由度の高い操作が可能となる．しかし，ディスプレイ1台であっても使用は可能であるし，プログラム次第では，さほど不自由なく使えるようにもできる．

前述のようにShellは，Cavanagh教授の研究室のプログラマー Raynald Comtois 氏が，Cavanagh教授の実験を支援するするために開発したものである．したがって，ハーバード大Cavanagh教授の実験のほとんどはShellを用いてプログラムしたものである．千葉大での私の実験の多くもまたShellを使った実験であるし，その他東工大，ヨーク大などでも使用されている．その他にもいくつかの研究室で使用していると思うが，使用者は100人を越えていることはないと思う．Comtois氏によると，現在のところShellで彼が得られる利益はARVOへの旅費がでる程度であるという．

Comtois氏はCavanagh教授に雇われているわけであるが，Shellについては，すべての権利をもっていてMicro ML inc.（4875 Frontenac, Saint-Hyacinthe, Quebec, Canada J2S 2E3; Phone +1-514-774-2604, Fax+1-514-771-4857）という会社を通して発売している．今のところ直接そこへ問い合わせるしかないが，対応は早いとは言えないようである．技術的な内容についてはComtois氏に電子メールで問い合わせると，すぐに分かることであれば返事は早い．

価格は現在のバージョンはUS$350であるが，次ぎのバージョンはそれよりも高くなる予定だという．ソフトと一緒に，比較的よくできたマニュアルとサンプルプログラムがついてくる．しかし，全くCを知らない人にとっては，まずCをある程度使えるようになる必要がある

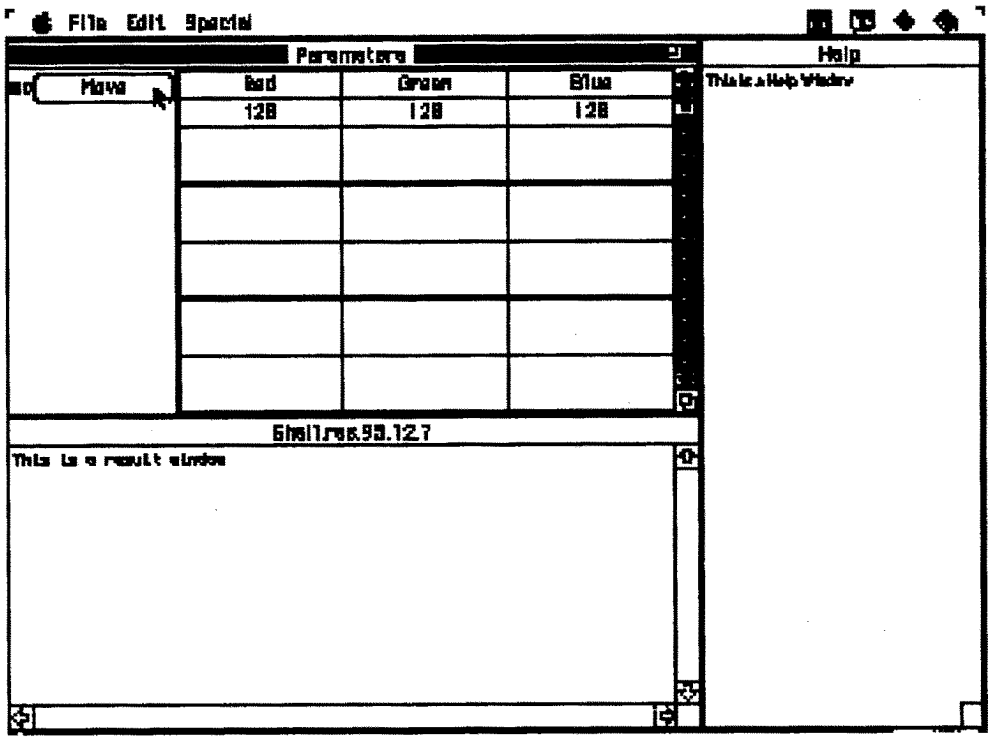

図1　3つのパラメータが設定されている．

し，また初めて視覚実験用のプログラムを作ろうとしている人は，やはりそれなりの時間が必要となるであろう．

現在市販されているバージョンは 2.2.1 であるが，Cavanagh 教授の研究室では現在 3.0 以上になっているのではないかと思う．私自身は，昨年彼の研究室に滞在した時に使わせてもらった 2.8.1 を，Comtois 氏好意によって使っている．現在我々の研究室では Mac IIcx, Mac IIfx, Quadra 800, Quadra 950, PowerMac7100 で使用していて，ビデオカードはほとんど 8・24 ビデオカードである．PowerMac 7100 については，使い始めたばかりであるので不明な点も多いが，処理速度はあまり期待できないようである．Shell の PowerPC の native code 化は次の次のバージョンの予定だという．

3．基本的な使用法

Shell を用いたプログラムは，制御用モニター上に表示されたいくつかのパラメーターを選択して，自由にその値を変化することができる．例えば，刺激の色をディスプレイのR，G，Bの3色で表しているとき，そのR，G，Bのそれぞれの輝度をパラメーターとして設定しておくと（図1），マウスでその値を変更することができるようになる．この場合，使用者が書くプログラムは，

```
addparameter(0,0,"Red",0,128,0,255,0L,
changecolor,0L,0L);
addparameter(0,0,"Green",0,128,0,255,0L,
changecolor,0L,0L);
addparameter(0,0,"Blue",0,128,0,255,0L,
changecolor,0L,0L);
```

の各行と，マウストラッキングモードが選択された時に何をするかを決める関数 changecolor を含む必要がある．Shell がなにをするかと言えば，addparameter というサブルーチンによって指定されたパラメーターをマトリックスの中に作り，そのパラメーターが選択されたときに指定されたサブルーチンを実行するということである．サブルーチンで，マウスによって変更された値を用いて刺激の色を変更するという操作をプログラムしておけば，パラメータの値の変更とともに刺激の色が変化する．サブルーチンの指定は上述のトラッキングモード時のほかに，パラメータが選択された時に実行す

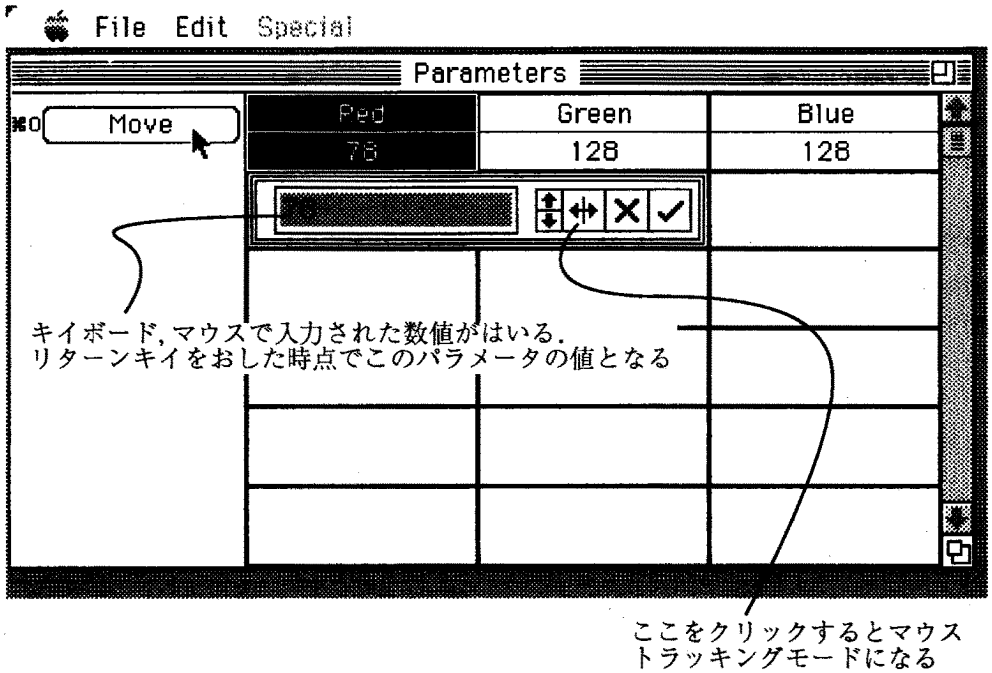

図2　パラメータ Red が選択された様子

るもの，クリックされたときに実行されるもの，選択が終了したときに実行するものの４種類が設定できる．パラメータの選択はその部分をマウスでクリックして行い，選択されるとキーボードから数値を入力することができるモードのなる（図２）．その状態でマウストラッキングモードの選択ボタンも表示されるので，数値を入力してリターンキイを押すか，マウストラッキングモードを選択する．上の例ではマウストラッキングモードの時以外はサブルーチンを指定していないのでなにも起こらない．また，パラメータを選択していない時には，常に mainloop という名前のルーチンを繰り返し実行しているので，定期的に色を変化するなどの時間的な制御は，このルーチンの中で実行可能である．

マウスで色を変えながら，クリックしたときにその数値を記録したい時には，例えば，値を出力する writeresult というルーチンを作り，それを以下の例のように指定しておけば，マウスがクリックされるたびにその時の値を記録してくれる．

```
addparameter(0,0,"Red",0,128,0,255,0L,
changecolor,writeresult,0L);
```

結果の出力などは，resultwindow と呼ばれるウインドウ（図１）に書き出すようになっている．このウインドウのなかにはコメントなども直接書き込むこともできる．resultwindow の内容は，Macintosh の標準の出力と同様に名前をつけて保存ができるようになっている．

addparameter での他の設定についての簡単な説明を付け加えておく．最初の２つのゼロは，パラメータの場所を意味する．この場合は一番左上となり，（1,0）が左から２番目，（2,0）が３番目で，（0,1）が左の上から２番目と続く．その次の文字列は，パラメータのタイトル，0,128,0,255 は順に，表示の小数点以下の桁数，初期値，扱うパラメータの最小値と最大値である．Shell では，パラメータの選択によって実行するサブルーチンを指定する他ににボタンとメニュー（Special という項目の下に加わる）による指定もできる（図１）．それらは addparameter の変わりに addbutton や addmenu という命令で指定する．例えば，ボタンを押すことで刺激を動かしたり止めたりするなどの使い方ができるわけである．その他，コメントを書いておくための helpwindow も使える．

４．Shell の特徴

Shell の機能中で，特に重要と考えられるものは３点ある．まず，描画のプログラムが作りやすい点である．上述の色を変える例では，当然刺激図形の色を変えたいのであるが，その図形は色を変えるなどの操作の前に書かれていなければならない．それについて，Shell はなにもしてくれない．使用者が位置を決め，四角なり丸なりを描く必要がある．そこでは，Macintosh の持っている Quickdraw のルーチンを使うか，Macglib の点を描くルーチンを使うことになる（あるいは，別のソフトで描いたものを読み込むこともできる）．この点では Shell を使うメリットが余りないと思うかも知れない．しかし，実はひとつのありがたい点がここにはある．Macintosh で図形を描くプログラムを（Shell なしで）作ろうとするとウインドウを開く操作などの Macintosh のシステムの持つ多くのルーチンを理解することから始めないといけない．なれない人にとってはかなりの時間を必要になる．もちろん出来上がったプログラムは，Macintosh-like となって使いやすくなるというメリットはあるが，Shell を用いた場合でも同様なものが得られる．

次の特徴は刺激に加える時間変化についての機能である．運動など刺激に時間的な変化を加えるためには，ルックアップテーブルによるアニメーションを使うか，ページを切り替えて画像に変化を与えるかのいずれかである．いずれにしても，重要なことは画像変化のタイミングであり，Shell のルーチンは簡単にこの問題を解決してくれる．ビデオディスプレイを使う以上フレーム周波数が時間制御の限界となるので，刺激に変化を与える場合はフレーム単位で

（つまりビデオ信号の垂直同期信号に同期して）変化を与えたい．このような操作はシステムが複雑になった現在のパソコンでは，なかなか面倒なこととなっている．市販のムービー編集用のほとんどのソフトウエアは，そのような同期についてはそれほど注意をはらっていない．通常のムービーでそれが問題となることは少ないし，現在ではおそらくその他の問題点の方が大きいからであろう．視覚実験においては，装置の限界の呈示時間での使用が望まれることが多いし，画像の変化が垂直同期と同期が取れていない場合は，画像の途中に画像の切り替えによる線が走って見えるなどの不自然は避けたい．

Shell でできることは単純なことである．macwait というルーチンを入れておくと次の垂直同期信号を待ってくれ，それを検出したあとその次の命令に進む．単純なことではあるが，これを一行挿入することにより，すべての時間はフレーム単位で制御できることとなる．また，macwait と次の macwait の間に多くの処理をした場合に，その間に１フレームを越える時間が経過してしまうかも知れない．その場合は時間が正確に計れなくなる．これをチェックしたい場合には，前の macwait から経過した時間をフレーム数で取り出すこともできる．経過時間がわかれば，少なくとも呈示時間などについて事実と異なる認識をする可能性はなくなる．

さて，この場合に次問題となるのは，ルックアップテーブルの入れ替えやページの入れ替えの方法である．こういった操作は Macintosh にかぎらず，現在多くのコンピュータではシステムが管理しているため，命令をだしてもその後いつ実際に変更されるか不明である，時間がかかるなどの問題がある．Shell ではこれらのグラフィック関連の多くの操作を直接ビデオボードにアクセスすることで，可能なかぎり短時間で行うことを可能としている．ただし，これは，Comtois 氏がチェックしその中身を検討したビデオカードに限られるため，その他のカードではシステムのルーチンを用いて操作を実行する．我々の研究室では，実験には Apple 社の 8・24 ビデオカードを使用するようにしているが，Shell の次のバージョンでは PCI バス対応のビデオカードにも最適化が予定されているので，今後は選択の幅が広がると期待している．

もうひとつの重要な特徴は，多くのパラメータを自由に変化することができる点である．前節の例では，RGB の色の変化を設定しているが，3 × 20 のマトリックスすべてを使えば，60 の変数を使って，刺激を操作することができる．この 60 という数は，非常に多いのであるが，それでも足りなくなってしまうことがないわけではない（これは，使用する変数の整理／統合がうまくなためであることが多いのではあるが）．実験用に限らずプログラムをメニュー形式にして変数を変えられるようにすることは多いと思うが，Shell では，それがマウスでできてしかもその変化した値をすぐに使って刺激を制御できる点は非常に便利である．

以上述べた点は，私が重要であると判断する Shell の特徴である．繰り返すと，１）描画の容易さ，２）画像変化のタイミング，３）パラメータマトリックスである．１）は，重要性から言えばこの３つの中ではもっとも低いが，実験刺激が素直に作れるという点で便利である．２）は，Macintosh を使うという条件では，Shell なしではかなり面倒となるし速度の問題では他により良い方法があるとは思えないため，Shell のもっとも重要な特徴であると思っている．３）は，プログラム上で色々な変数をいじってみて，そこで何が起こっているか観察する場合に便利であり，実験そのものというより予備実験や現象を観察する場合に効力を発揮する．私の場合，パラメータの数が限りなく増加する傾向があり，その場合，プログラムの使用を通して一度もあるいは一度だけしか変化されないパラメータがでてきてしまったりするが，それも 60 という多数のパラメータを扱えるためである（新しいバージョンではこの数はさらに増えている）．

5．その他の機能

Shell には数多くのルーチンがあるがすべてについて解説する余裕はないし，また，我々が使っていない機能も多い．以下では，個人的に便利だと思う機能を並べてみたい．もちろん，これらは，Shell だけに特有な機能ではなく，他のソフトでももっている．つまり，上記の理由で Shell を使いたい時にそれと一緒に以下の機能が使えるという点で，便利な機能であると言うべきかも知れない．

・メモリー上に仮想的にビデオ画面を作る．たくさんの画像が必要なときそれをメモリー上に作って（あるいは読み込んでおいて）随時ビデオ画面にコピーして使う．
・刺激画像の保存．スライドなどにするときに使える．
・画像の値の読み取り．FFT などの処理をしたいときに使える．
・複数のディスプレイを簡単に制御できる．selectcard という命令で，どのディスプレイの画像を変更したいかを指定するだけでそれができる．
・標準フォーマット（PICT）の任意の画像を読み込める．他のソフトで作成した，あるいはスキャナーで取り込んだ画像を刺激として使える．
・パラメータマトリックスの他にボタンとメニューを使うことができる．使用法，パラメータマトリックスの場合と同様に，選択されたときにどこのサブルーチンを実行するかを指定する．

現在 Shell は，市販用の新しいバージョン3に向けて開発が進んでいて，そのバージョンでは，いくつか新しい機能が追加される予定であるという．そのなかで興味深いものを以下に列記しておく．

・パラメータマトリックスの中の値を保存できる．今までは，そうしたい場合，自分でその部分のプログラムを書かなければならなかったが，多くのパラメータを使う場合には厄介な作業になったし，それをおこたると実験に際してパラメータの設定を間違える可能性もあった．この機能は，そのような煩わしさから開放される．
・サイン波グレーティング，ランダムドットパターンを作ってくれる．我々はすでにそのようなルーチンを何度も使っているので，それほど便利とも思わないが，初めて使う場合は便利な機能であろう．
・オーバーレイ機能が追加される．これによって，例えば画像の一部のみを切り取って表示するなどの手続きが簡略化されると思われる．
・AppleScript に対応する．AppleScript を書けるのであれば，自動処理が便利になる．
・PCI 対応のビデオカードの最適化（最短時間での命令の実行を可能にする）予定．
・QuickTime ムービーファイルの読み込みを可能にする予定．
・マニュアルの電子化の予定．

新しいバージョンについて問い合わせると，いつでもあと2，3か月かかるとの返事が返ってくるので，いつできるのかの予測は難しいが，今回は半年以内には新しくなると思われる．

6．おわりに

Shell を使うことで，視覚実験用のプログラムの開発時間はかなり短縮されると思うし，Shell のもっている機能は便利なものが多い．もちろん使用者は，常に新しい他の研究者が十分検討していない実験条件を求める傾向にあるので，必要な機能が Shell に十分そなわっていないことは頻繁にあると思う．そのような場合には，使用者が解決方法を考えなければならないわけであるが，それでも Shell の環境をそのまま使いながら解決方法を考えることができるであろう．そのような自由度も含めて考えると，Shell は多くの Macintosh 使用者にとって使ってみて損のないソフトであるといえる．