心理実験教育用プログラミング言語としての Processing の利用可能性

伊丸岡 俊秀　　金沢工業大学情報学部心理情報学科

代表的な心理実験用ツールとして，実験作成／制御用ソフトとしては E-Prime や Presentation（が，プログラミング言語ないしはプログラミング用ライブラリとしては Matlab 用 toolbox の Psychtoolbox や Cogent が挙げられる．これらのソフトェア／ライブラリは精度の高い実験を容易に作成可能だが，高価なソフトの購入が必要となり教育場面では使いにくい．本発表では，GPL のもとでフリーソフトウェアとして公開されている開発環境である Processing（http://processing.org）での心理実験作成事例と時間精度の計測結果を報告し，心理実験教育用としての利用可能性と限界を論じる．

Keywords: graphic programing, psychological experiment, Java.

問題・目的

代表的な心理実験用ツールとして，実験作成／制御用ソフトとしては E-Prime（http://www.pstnet.com/）や Presentation（http://www.neurobs.com/）が，プログラミング言語ないしはプログラミング用ライブラリとしては Matlab 用 toolbox の Psychtoolbox（Brainard, 1997; Pelli, 1997; http://docs.psychtoolbox.org/Psychtoolbox）や Cogent（http://www.vislab.ucl.ac.uk/cogent.php）が挙げられる．これらのソフトェア／ライブラリは精度の高い実験を容易に作成可能だが，高価なソフトの購入が必要となるため，学部教育のように多数のライセンスが必要となる場面では使用するのは難しい．安価に，あるいは無償で入手可能な環境である Vision Egg（Straw, 2008; http://www.visionegg.org/）や Psychlops（http://psychlops.l.u-tokyo.ac.jp/）といったソフト／ライブラリも開発されているが，これらを使用するためには Python や C++用の開発環境を整える必要があり，プログラミングの初学者にとっては難易度が高い．これらの理由により，現在，学部学生を対象に心理実験作成用のプログラミング教育を行うのは困難な状況にあると思われる．しかし，実験心理学を学ぶ学部学生や大学院修士課程の大学院生にとってプログラミングは必要な技術であり，教育場面で使用しやすい環境が望まれる．本稿では，無償で入手可能で，比較的容易にプログラミング可能である Processing を用いた心理実験作成について報告する．

Processing

Processingは，プログラミングの基礎知識を持たないユーザーがコンピュータ上で“スケッチ”を行うことができることを目的に，MIT Media Labで作成されたJavaベースのプログラミング言語で，GNU GPLとLGPLのデュアルライセンスのもとで配布されているオープンソースソフトウェアである．Windows, Mac OS X, Linuxで動作する開発環境が統合されているため，通常のソフトウェアをインストールする手順を踏むだけで，開発が可能となるため，導入は容易である．また，グラフィック描画に特化された環境であり，さらに基本的な描画関数も用意されていることから，単純なものであれば，数行のプログラムを書くだけで描画可能である（Figure 1）．

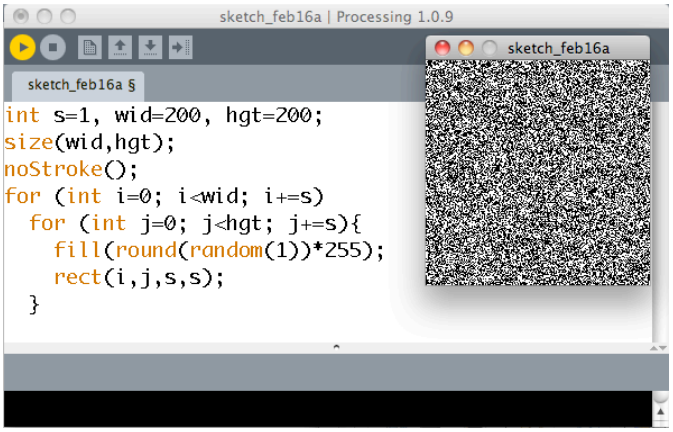

Figure 1. An example of source code for creating a random dot pattern.

心理実験の作成

Processingによるプログラムは，初期設定のためのsetup()とグラフィック描画のためのdraw()という2つの基本的な関数によって構成される．作成したアプリケーションを実行すると，最初のsetup()が一度だけ，その後setup()がアプリケーションの終了まで繰り返される．draw()が実行される頻度はコンピュータ画面のフレームレートを上限に設定可能である．

setup()とdraw()以外の関数として，マウス，キーボードによる入力や，外部ファイルへの書き出しなど，心理実験作成に必要な機能を使用可能である．入力に関わる関数はdraw()のループとは独立に入力装置を監視するため，被験者の反応の取得も容易である．そのため，基本的な実験であればdraw()内で適切な条件分

岐をすることだけで作成できる．Figure. 2は基本的な心理実験プログラム構造の例である．

```
// 試行数などをグローバル変数で設定
int trial=0, maxTrial=100;
void setup(){
        // 画面設定や出力ファイル生成などの初期設定
}
void draw(){
        // 固視点，固視点持続，刺激呈示，反応待ちなどの分岐
}
void keypressed(){
        // キーが押されると実行される関数
        // 反応書き込み
        trial++;
        if (trial==maxTrial){
                // ファイル書き出し
        }
}
```

Figure 2. An example of the program structure for a psychological experiment.

また，Processingは画像の扱いやピクセル単位での描画が容易である上に，Psychtoolboxのtextureにあたるメモリ上でのグラフィック描画も可能なため，例えばGabor patchのような刺激も作成可能である（Figure. 3）．

Figure 3. A gabor patch created by Processing.

時間精度

前述のようにProcessingでは画面描画のためのループであるdraw()の頻度を変更することで，画面表示時間を制御できる単位を変更することができるが，デフォルトのレンダリングシステムを用いると，その精度はあまり高くない．例えば，フレームレートを60に設定して，全画面への描画を300回繰り返したところ，フレーム間間隔の平均が16.7ms, 標準偏差が0.67ms, 範囲が15.69msから27.20msとかなり大きなばらつきを示した．このばらつきは，レンダリングにOpenGLを使用することである程度は抑えることが可能であるが，それでもPsychtoolboxのばらつきと比べると一桁程度大きくなることが分かった（Figure. 4; フレームレート60でほぼ同一の描画を，同一コンピュータで実施した結果）．Processingによる描画では，平均が16.67ms, 標準偏差が0.41msと設定に近い値になっているものの，範囲は15.71msから17.70msとフレームによっては1ms程度のずれが生じてしまう．

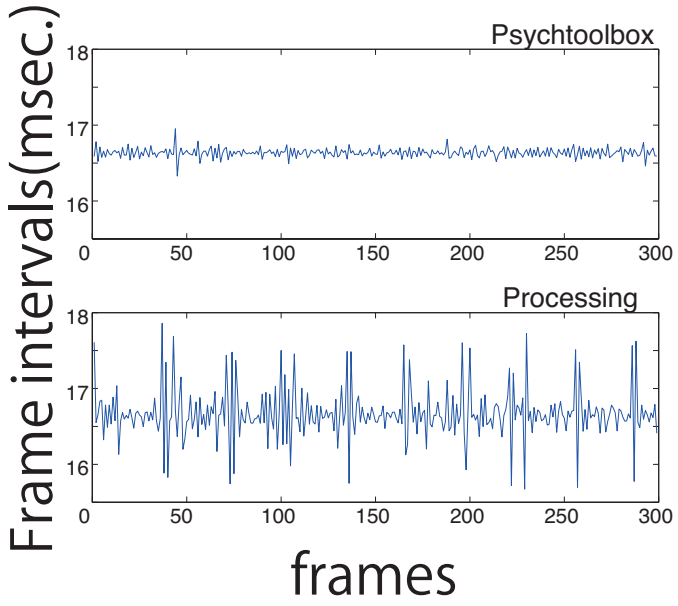

Figure 4. Intervals between display updates controlled by Psychtoolbox and Processing.

結論

ここまで心理実験用プログラミング言語としてProcessingを使うことの可能性について検討し，導入と使用の容易さという観点から，教育用途での有用性について議論した．ただし，同時に高い時間精度での制御な実験への適用における限界も示した．