The Psychophysics Toolbox

David H. Brainard
Department of Psychology
UC Santa Barbara
Santa Barbara, CA 93106

Short Title: The Psychophysics Toolbox

Corresponding Author

David H. Brainard
Department of Psychology
UC Santa Barbara
Santa Barbara, CA 93106
brainard@psych.ucsb.edu

This paper appeared in *Spatial Vision*, 10, 433-436. This version has been slightly modified from the published version to reflect changes in the toolbox software and website.

http://color.psych.ucsb.edu/psychtoolbox

Abstract

The Psychophysics Toolbox is a software package that supports visual psychophysics. Its routines provide an interface between a high-level interpreted language (MATLAB on the Macintosh) and the video display hardware. A set of example programs is included with the toolbox distribution.

Introduction

The attraction of using computer displays for visual psychophysics is that they allow software specification of the stimulus. To provide the required precision, experimental programs are usually written in a low-level language (e.g. C or Pascal). Although these languages provide power and flexibility, they are not conducive to rapid program development. Interpreted languages (e.g. LISP, Mathematica, or MATLAB) are abstracted from hardware details and provide friendlier development environments. Most interpreted languages are not designed to support psychophysics, however, and do not allow precise stimulus specification. The Psychophysics Toolbox is a software package that adds this capability to the MATLAB application on Apple Macintosh computers.

MATLAB is a high level interpreted language with extensive support for numerical calculations (The MathWorks, 1993). The Psychophysics Toolbox interfaces between MATLAB and the Macintosh hardware. The toolbox's core routines provide access to the display frame buffer and color lookup table, allow synchronization with the vertical retrace, support millisecond timing, and facilitate the collection of observer responses. Ancilliary routines support commonly used procedures such as color space transformations (Brainard, 1995) and the QUEST (Watson and Pelli, 1983; Pelli and Farell, 1995) threshold seeking algorithm.

Examples and Uses

Table 1 lists a simple MATLAB program that uses the Psychophysics Toolbox to display a flickering counterphase grating. Steps 1 and 2 of the example use standard MATLAB functions to compute a the grating and a series of color lookup tables. Steps 3-5 use Psychophysics Toolbox

routines to flicker the grating at the center of the screen. The program is intended only as an illustrative example. An extended version of this example that incorporates gamma correction is provided in the distribution.

Other example programs provided in the distribution implement visual search, detection of dot symmetry and motion, lightness matching, magnitude estimation of line length, and measurement of contrast thresholds in static noise. Many of these were designed as experimental modules for an undergraduate laboratory course on perception and may be useful for teaching.

[Table 1 about here.]

An email poll of users revealed that the toolbox has been used to measure a variety of psychophysical thresholds (e.g. detection of gratings and letters in noise), to display stimuli for functional MRI and electrophysiogical experiments, to measure asymmetric color matches, to evaluate image compression algorithms, and to study categorization, perceptual learning, visual search, and visual object recognition.

Implementation

The core routines of the Psychophysics Toolbox are MATLAB Extension (MEX) files. They are written in C and compiled into an object format recognized by MATLAB. As such, they may be called directly from MATLAB programs. The MEX files run on both 680x0 and PowerPC Macintoshes. The core routines rely on the subroutines in Pelli's VideoToolbox (Pelli and Zhang, 1991; Pelli, 1996) which provide low-level control of the Macintosh display hardware. The toolbox distribution includes the C source for the MEX files.

Documentation and Availability

The Psychophysics Toolbox is available at the web site. Installation consists of adding a single folder (approximately 2.5 megabytes) to MATLAB's collection of toolboxes. The distribution includes example programs and source code. The software may be used freely for teaching or research. It may not used for commercial gain without permission of the author.

Acknowledgments

Many people have contributed to the development of the Psychophysics Toolbox. Full acknowledgments are provided at the web site. Denis Pelli wrote the VideoToolbox routines (Pelli, 1996) on which the Psychophysics Toolbox relies and modified these routines for use in MEX files. More recently he has contributed substantially to the Psychophysics Toolbox itself.

Macintosh is a trademark of Apple Computer Inc. Mathematica is a trademark of Wolfram Research Inc. MATLAB is a trademark of The MathWorks Inc. PowerPC is a trademark of International Business Machines Corporation.

Development of the Psychophysics Toolbox was supported in part by NSF DUE9350868 and by NEI EY 10016.

References

Brainard, D. H. (1995). Colorimetry. In*Handbook of Optics: Volume 1. Fundamentals, Techniques, and Design*. M. Bass (ed.). McGraw-Hill., New York, 26.1-26.54.

The MathWorks. (1993). *MATLAB User's Guide*. The MathWorks, Inc., Natick, MA.

Pelli, D. G. (1997). The VideoToolbox software for visual psychophysics. *Spatial Vision,* **10**, 437-442.

Pelli, D. G.& Farrell, B. (1995). Psychophysical methods. In*Handbook of Optics: Volume 1. Fundamentals, Techniques, and Design*. M. Bass (ed.). McGraw-Hill., New York, 29.1-29.13.

Pelli, D. G.& Zhang, L. (1991). Accurate control of contrast on microcomputer displays. *Vision Research* **31**, 1337-1350.

Watson, A. B.& Pelli, D. G. (1983). Quest: a Bayesian adaptive psychometric method. *Perception and Psycophysics* **33**, 113-120.

Table Caption

Table 1. Listing of a simple MATLAB program that uses the Psychophysics Toolbox to display a flickering counterphase grating. Steps 1 and 2 use standard MATLAB functions to compute a

sinusoidal grating and a series of color lookup tables. Steps 3-5 use Psychophysics Toolbox routines to display the flickering grating.

[Note: This demo program is now out of date, although its flavor is still representative. Up-to-date demos may be found in the toolbox itself or on the web site.]

```
% STEP 1:  Create a sinusoidal image in a MATLAB matrix.  The
% grating is scaled between 1 and 255.  The creation
% happens in two steps.  First create a sinusoidal vector,
% then replicate this vector to produce a sinusoidal image.
% The replication is done by an outer product.  This is not
% the fastest way to do it in MATLAB, but it is a little
% clearer.
nPixels = 256;
cyclesPerImage = 4;
sinVector = 1+ 254*(1 + sin(2*pi*cyclesPerImage*(1:nPixels)/nPixels))/2;
sinImage = ones(nPixels,1)*sinVector;

% STEP 2:  Create a set of color lookup tables (cluts) for animating the
% grating.  Each clut is a 256 by three matrix.  We compute a set of
% cluts that will produce one cycle of counterphase flicker.  These
% are all stored in a single large matrix (theCluts).  We also create
% a uniform clut (offClut) to initialize the display.  The variable
% nCluts determines the number of temporal quantization intervals.
nCluts = 80;
theCluts = zeros(256,3*nCluts);
theContrasts = sin(2*pi*(0:nCluts-1)/nCluts);
for i = 1:nCluts
      contrast = theContrasts(i);
      lowVal = 127.5 - contrast*127.5;
      highVal = 127.5 + contrast*127.5;
      clutEntries = [round(linspace(lowVal,highVal,256)')];
      theCluts(:,(i-1)*3+1:i*3) = clutEntries*ones(1,3);
end
offClut = 128*ones(256,3);

% STEP 3: Open up the a window on the main screen, initialize the clut,
% and draw a grating onto the screen.  This is done by calling the
% SCREEN function of the Psychophysics Toolbox.  The SCREEN function
% performs a variety of display tasks according to its text argument.
SCREEN('OpenScreen',128,0);
SCREEN('SetClut',offClut);
SCREEN('PutImage',sinImage);

% STEP 4: Animate the clut.  The clut animation is handled by a
% SCREEN function which synchronizes clut writes to the vertical
% blank of the display.
firstClutEntry = 0;
framesPerClut = 3;
nCycles = 2;
HideCursor;
SCREEN('ClutMovie',theCluts,firstClutEntry,framesPerClut,nCluts*nCycles);
SCREEN('SetClut',offClut);
ShowCursor;

% STEP 5: Close up the screen and exit.
SCREEN('CloseScreen');
```

Table 1