A Basic Digitising Software

SAMIR KUMAR PAL

In many scientific experiments, it becomes necessary to measure the area under the experimentally obtained curve (spectrum). For instance, for measurement of fluorescence quantum yield and differential scanning calorimetry etc., it is often required to store the experimentally obtained hardcopy (spectrum) in the form of a data file for future use so as to regenerate the curve using some curve-tracing software such as Grapher or Microcal Origin, etc.

Typically, in such an experiment, one of the physical quantities is obtained as a function y(x) of another physical quantity x, which is treated as the independent variable. The quantity of interest, however, is not y(x) but rather $I(x, y) = \int_{a}^{b} y(x)dx$, (a ≤ x ≤ b), i.e. the area under the curve.

However, if the experimental output is in the form of a spectrum and the spectrometer is not computer interfaced, we need a scanner to digitise the spectrum, determine the area under the spectrum, and store the corresponding data. This technique, though effective, is at the same time expensive. An alternative cheaper technique has been developed by the author using a GWBASIC program which does away with the scanner.

The experimental spectrum is first traced on a transparent sheet. This sheet is then attached on to the screen of the computer monitor such that the base line of the spectrum coincides with the green coloured baseline of the program-generated screen. The spectral curve is then traced pixel by pixel (---) on the screen beneath the attached transparent sheet by using the four arrow keys. When the entire curve has been thus redrawn on the monitor screen, the paper is taken off and <Esc> key is pressed. If the data file corresponding to the spectrum is required, then the start wavelength, end wavelength, and name of the data file can be specified interactively. The program computes the area under the curve in arbitrary (arb.) units and displays the result as the output.

The actual emission characteristic (spectrum) of a xenon lamp obtained using a Perkin Elmer made fluorimeter, which is not computer interfaced, is shown in Fig. 1(a). Fig. 1(b) is the attenuated version of the same. Figs 2(a) and 2(b) are the program-generated replicas of Figs 1(a) and 1(b) respectively.

![Emission Characteristics of a Xenon Lamp](image)

**Fig. 1:** Actual emission spectrum of xenon lamp obtained using fluorimeter

![Program Generated Replicas of Spectrum Obtained Using Software](image)

**Fig. 2:** Program generated replicas of spectrum obtained using software

It is obvious that the program-generated spectra are true reproduction of the actual spectra. The area under the spectra Figs 2(a) and 2(b) have been calculated to be 23,445 arb. units and 6,142.5 arb. units using the program. From the ratio of the two areas we can easily infer that, within the wavelength range of 400 nm to 500 nm, the total number of photons emitted in the attenuated version is 3.8 times less than that of previous one.

Thanks are due to author's colleagues A. Datta and D. Mandal for their very useful comments and suggestions.

**Digitising Software Program in Basic**

10 DIM X(1000), Y(1000), F(2000), G(2000), A(1000), B(1000)
20 I = 0
30 CLS: SCREEN 8: KEY OFF: X = 1: Y = 195
40 GOSUB 530
50 PSET(X, Y), 15
60 I = I + 1
70 K = POINT(I)
80 C = POINT(1)
90 X$ = ""
100 WHILE X$ = ""; X$ = INKEY$: WEND
110 IF X$ = CHR$(27) THEN GOTO 230
120 IF X$ = CHR$(0) + CHR$(72) THEN Y = Y + 1: N = Y + 1
130 IF X$ = CHR$(0) + CHR$(78) THEN Y = Y - 1: N = Y - 1
140 IF X$ = CHR$(0) + CHR$(75) THEN X = X - 1: M = X + 1
Rubick Cube

SHREE KUMAR

All of us, at some point of time, have tried varying mental exercises to keep ourselves away from boredom and (b) to harness our immense, yet dormant, intellectual capacities. The Rubick Cube offers a mental exercise and is also an efficient means of passing time. And surely, nothing else can be more fascinating than playing the Cube on the computer.

First, the game starts with an introductory screen, after which the user is asked to select a level of difficulty from amongst these: 1. Novice, 2. Expert, and 3. Challenger.

Next, the player sees the isometric projection of the cube on the screen, at the centre. The other faces, as seen from the left, back and bottom, are also shown to the left, right, and below the Cube respectively. Pressing any key jumbles the Cube.

The Cube is divided into nine rotatable parts. To move a part, press the top row to the left, all he has to do is to press the corresponding number (in this case 1), followed by the left arrow key. Finally, the user has to end up with the same colour on each of the faces.

As soon as this happens, the game terminates with a congratulatory message.

Pressing Esc key terminates the program from the game screen and pressing F3 starts a new game.

On execution, the program searches for the files BGAVGA.BGI, GOTH.CHR, and LITT.CHR in the C:\TCP directory. If these files don’t exist there, then these are searched for in the current directory. If not found there too, the program terminates. The user can specify the path to these files in the command line itself. For example, use

`A:\RUBICK C:\TCP\BGI`

if the above files are in the C:\TCP\BGI directory. It’s always preferable to have these files in the hard disk because loading these files from a floppy can take time. You can even create a stand-alone executable file using register bgi driver and register bgi fonts—the standard library functions of C.

This program was initially designed for a monochrome monitor, so the colour selection may not be to your liking. You can change the colours to suit your taste.

Further improvements that the user