

---

Subject: XOR graphics

Posted by [your name](#) on Wed, 15 Dec 1999 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Hi all,

I was intrigued by someone's comments about using XOR graphics mode (I think it was DWF).

I am using XOR to plot a moving object. It works OK on a terminal using TRUECOLOR but screws up the colours on a terminal using PSEUDOCOLOR.

I am still trying to work this out, but I thought I'd try using a PIXMAP window followed by a copy to the viewable window. However, this approach, although worked OK on both terminals, was 10 times slower than using XOR.

Since my application is time critical, it looks as if I'm stuck with XOR.

I'm using IDL 5.2 under OpenVMS using DECWindows (X windows). Below is a copy of my test software. I'd welcome any comments (it may not be the tidiest of code but it works).

```
FUNCTION Plot_rotated_shape, X, Y, Shape, Clr
  Point_1 = CONVERT_COORD(X[0], Y[0], /TO_NORMAL) ; Values in norm
coords
  Point_2 = CONVERT_COORD(X[1], Y[1], /TO_NORMAL) ; Values in norm
coords
  Ang = ATAN(Point_2[1] - Point_1[1], $
Point_2[0] - Point_1[0]) * !RADEG ; DRAWN angle of line
  T3D, /RESET, $ ; Create transform matrix
  ROTATE = [0, 0, Ang] ; ...rotated to fit line
  T3D, TRANSLATE = Point_2 ; ...and moved to end point
  POLYFILL, Shape, COLOR = Clr, /NORMAL, /T3D ; Plot shape
  RETURN, !P.T ; Return transformation
END
```

```
PRO Initialise, Obj_1, Obj_2
```

```
DEVICE, DECOMPOSED = 0, RETAIN = 2 ; Screen setup
TVLCT, [255, 155], [0, 200], [0, 255], 1 ; Define colours
```

```
Obj_1.X_coord = FINDGEN(200) ; Create some data
Obj_1.Y_coord = Obj_1.X_coord ^ 2
Obj_2.X_coord = FINDGEN(200)
Obj_2.Y_coord = 40000 - Obj_2.X_coord ^ 2
```

```
PLOT, Obj_1.X_coord, Obj_1.Y_coord, TICKLEN = 1, /NODATA ; Plot axes
```

```
Shape_x = [-15.0, -15.0, 15.0, 15.0, -15.0] ; Define shape
Shape_y = [-5.0, 5.0, 5.0, -5.0, -5.0]
```

```
Obj_1.Shape = CONVERT_COORD(Shape_x, Shape_y, $ ; ...in normalised
/DEVICE, /TO_NORMAL) ; coords
Obj_2.Shape = Obj_1.Shape
```

```
END
```

```
PRO Exor
```

```
; This program draws 2 moving objects along the object trajectories.
```

```
; The objects are drawn and then redrawn which effectively erases them
; (because the XOR graphics mode is selected). No underlying info is
lost.
```

```
Obj_1 = {X_coord:FLTARR(200), Y_coord:FLTARR(200), Shape:FLTARR(3, 5)}
Obj_2 = Obj_1
```

```
Initialise, Obj_1, Obj_2 ; Create some data
```

```
Start = SYSTIME(1)
```

```
FOR J = 1, 199 DO BEGIN ; Work thru all data
    PLOTS, Obj_1.X_coord[J - 1:J], Obj_1.Y_coord[J - 1:J], COLOR = 1 ;
Add track
    PLOTS, Obj_2.X_coord[J - 1:J], Obj_2.Y_coord[J - 1:J], COLOR = 2 ;
Add track
    DEVICE, SET_GRAPHICS_FUNCTION = 6 ; Set XOR
    Obj_1_T3 = Plot_rotated_shape(Obj_1.X_coord[J - 1:J], $ ; Add shape
Obj_1.Y_coord[J - 1:J], Obj_1.Shape, 1)
    Obj_2_T3 = Plot_rotated_shape(Obj_2.X_coord[J - 1:J], $ ; Add shape
Obj_2.Y_coord[J - 1:J], Obj_2.Shape, 2)
; WAIT, 0.05 ; Uncomment to watch plot
    POLYFILL, Obj_2.Shape, COLOR = 2, /NORMAL, /T3D ; Re-plot shapes
(delete them)
    !P.T = Obj_1_T3 ; Transform to other object
    POLYFILL, Obj_1.Shape, COLOR = 1, /NORMAL, /T3D
    DEVICE, SET_GRAPHICS_FUNCTION = 3 ; Reset the graphics mode
ENDFOR
```

```
PRINT, SYSTIME(1) - Start
```

```
END
```

```
PRO Pix
```

; This program draws 2 moving objects along the object trajectories.

; Objects are PLOTted to a pixmap, which is subsequently copied to the  
; current window.

```
Obj_1 = {X_coord:FLTARR(200), Y_coord:FLTARR(200), Shape:FLTARR(3, 5)}  
Obj_2 = Obj_1
```

WINDOW, 1, XSIZE = 640, YSIZE = 512 ; Visible window

WINDOW, 0, XSIZE = 640, YSIZE = 512, /PIXMAP ; Define pixmap window

Initialise, Obj\_1, Obj\_2 ; Create some data

Start = SYSTIME(1)

FOR J = 1, 199 DO BEGIN ; Work thru all data

WSET, 0 ; Use pixmap

PLOT, Obj\_1.X\_coord, Obj\_1.Y\_coord, TICKLEN = 1, /NODATA ; Add axes  
(white)

OPLOT, Obj\_1.X\_coord[0:J], Obj\_1.Y\_coord[0:J], COLOR = 1 ; Add track

OPLOT, Obj\_2.X\_coord[0:J], Obj\_2.Y\_coord[0:J], COLOR = 2 ; Add track

Obj\_1\_T3 = Plot\_rotated\_shape(Obj\_1.X\_coord[J - 1:J], \$ ; Add shape  
Obj\_1.Y\_coord[J - 1:J], Obj\_1.Shape, 1)

Obj\_2\_T3 = Plot\_rotated\_shape(Obj\_2.X\_coord[J - 1:J], \$ ; Add shape  
Obj\_2.Y\_coord[J - 1:J], Obj\_2.Shape, 2)

; WAIT, 0.05 ; Uncomment to watch plot  
WSET, 1

DEVICE, COPY = [0, 0, 640, 512, 0, 0, 0] ; Place on visible window  
ENDFOR

PRINT, SYSTIME(1) - Start

END

> EXOR

0.601

> PIX

6.000

Ian Dean

Ian.Dean@GECM.COM