Subject: Re: XINTERANIMATE problem in IDL 5.02
Posted by thompson on Tue, 16 Dec 1997 08:00:00 GMT
View Forum Message <> Reply to Message

bowman@null.edu (Kenneth P. Bowman) writes:

> I have a program that animates using XINTERANIMATE. It works fine under
> IDL 4, but fails under 5.02. The symptoms are:

> 1. The program realizes the animate widget correctly. (SET = [ni, nj, nframes])

> 2. It loads all the frames. (FRAME = frame)

> 3. It then activates the animator (no keywords) and the labels become
> 'ungrayed', but the cursor remains an hourglass, none of the controls
> work, and the animation does not run. The program then 'falls through'
> the XINTERANIMATE call and executes the subsequent IDL code.

> I haven't seen any reports of similar problems in this newsgroup. Anyone
> else having this problem? Any cures?

> Thanks, Ken Bowman

> E-mail replies in addition to postings would be appreciated. My address
> is bowman at csrp dot tamu dot edu. (Please translate the address in the
> obvious way, I am sick of getting spam.)

I'm seeing something similar. It seems that under IDL/v5, XINTERANIMATE is able to run independently of the command line. I tried running a short piece of code which ran two movies back-to-back. Under IDL/v5, the first one loaded, but before it could start running, the second one tried to load. Only after I got past all the error messages about not running two copies of xinteranimate at once, did the first movie actually start. I tried putting a read from the keyboard statement between the two movies to force a wait between the two movies, but this did not have any effect--the first movie wouldn't start until after it tried to execute the second movie. I suspect that it was waiting for the entire procedure to execute.

Upon investigation, I discovered that xinteranimate.pro had been modified to include the keyword /NO_BLOCK in the XMANAGER statement. This is problematic for two reasons:

1. This should have been a keyword in the XINTERANIMATE procedure, so that it could be under the control of the user.

2. The behavior is different from previous versions of XINTERANIMATE. I'm all in favor of new features, but existing routines should always maintain their default behavior through version changes. New features should be

incorporated as options.

I found that if I used the xinteranimate.pro from IDL/v4, it seemed to work just fine in IDL/v5. That's my recommendation until RSI can fix the routine in their library. For your convenience, I include the IDL/v4 version of xinteranimate.pro below.

My recommendation to RSI is to change the IDL/v5 version of xinteranimate.pro to include NO_BLOCK as a keyword in the xinteranimate procedure definition, and change the XMANAGER line to read NO_BLOCK=NO_BLOCK. That way, the default behavior will be the same as we've come to expect, but the user can select /NO_BLOCK in the XINTERANIMATE call if desired. RSI must take backwards compatibility seriously.

William Thompson

```
 ============================================================
==================
; $Id: xinteranimate.pro,v 1.3 1993/10/18 19:46:19 dave Exp $

; Copyright (c) 1992-1993, Research Systems, Inc.  All rights reserved.
; Unauthorized reproduction prohibited.

;+
; NAME:
;  XINTERANIMATE
;
; PURPOSE:
; Display an animated sequence of images using X-windows Pixmaps.
; The speed and direction of the display can be adjusted using
; the widget interface.
;
; CATEGORY:
; Image display, widgets.
;
; CALLING SEQUENCE:
; To initialize:
;  XINTERANIMATE, SET = [Sizex, Sizey, Nframes]
;
; To load a single image:
;  XINTERANIMATE, IMAGE = Image, FRAME = Frame_Index
;
; To load a single image that is already displayed in an existing window:
;  XINTERANIMATE, FRAME = Frame_index, $
;   WINDOW = [Window_Number [, X0, Y0, Sx, Sy]]
; (This technique is much faster than reading back from the window.)
;
; To display the animation after all the images have been loaded:
```

```
;  XINTERANIMATE [, Rate]
;
; To close and deallocate the pixmap/buffer (which also takes place
; automatically when the user presses the "Done With Animation"
; button or closes the window with the window manager):
;  XINTERANIMATE, /CLOSE
;
; OPTIONAL INPUTS:
; Rate: A value between 0 and 100 that represents the speed of the
;  animation as a percentage of the maximum display rate.
;  The fastest animation is with a value of 100 and the slowest
;  is with a value of 0.  The default animation rate is 100.
;  The animation must be initialized using the SET
;  keyword before calling XINTERANIMATE with a rate value.
;
; KEYWORD PARAMETERS:
; CLOSE: Set this keyword to delete the offscreen pixwins and Widget,
;  freeing storage.
;
; CYCLE: If set, cycle.  Normally, frames are displayed going either
;  forward or backwards.  If CYCLE is set, reverse direction
;  after the last frame in either direction is displayed.
;  Provide this keyword with the SET keyword.
;
; FRAME: The frame number when loading frames.  This keyword only has
;  an effect when used in conjunction with the SET keyword.
;  FRAME must be set to a number in the range 0 to Nframes-1.
;
; GROUP: The widget ID of the widget that calls XINTERANIMATE.  When
;  this ID is specified, the death of the caller results in the
;  death of XINTERANIMATE.
;
; IMAGE: A single image to be loaded at the animation position given
;  by FRAME.  The keyword parameter FRAME must also be specified.
;
; KEEP_PIXMAPS: If TRUE, XINTERANIMATE doesn't destroy the animation
;  pixmaps when it is killed. Calling it again without
;  going through the SET and LOAD steps will cause the same
;  animation to play without the overhead of creating
;  the pixmaps.
; ORDER: Set this keyword to display images from the top down instead
;  of the default bottom up.  This keyword is only used when
;  loading images.
;
;     SHOWLOAD: Set this keyword (in conjunction with the SET keyword) to
;  display each frame and update the frame slider as frames are
;  loaded.
;
```

```
; SET: This keyword initializes XINTERANIMATE.  SET should be equated
;  to a 3-element integer vector containing the following
;  parameters:
;    Sizex, Sizey: The width and height of the images to be
;    displayed, in pixels.
;
;    Nframes: The number of frames in the animated sequence
;    (since XINTERANIMATE is an animation routine,
;    Nframes must be at least 2 frames).
;
; TITLE: A string to be used as the title of the widget.  If this
;  keyword is not specified, the title is set to "XInterAnimate"
;  This keyword has an effect only when used in conjunction with
;  the SET keyword).
;
; TRACK: If set, the frame slider tracks the current frame.  Default
;  is not to track.  Provide this keyword with the SET keyword.
;
; WINDOW: When this keyword is specified, an image is copied from an
;  existing window to the animation pixmap.  When using X
;  windows, this technique is much faster than reading
;  from the display and then calling XINTERANIMATE with a 2D
;  array.
;
;  The value of this parameter is either an IDL window
;  number (in which case the entire window is copied),
;  or a vector containing the window index and the rectangular
;  bounds of the area to be copied, for example:
;  WINDOW = [Window_Number, X0, Y0, Sx, Sy]
;
;     XOFFSET: The horizontal offset, in pixels from the left of the frame,
;  of the image in the destination window.
;
;     YOFFSET: The vertical offset, in pixels from the bottom of the frame,
;  of the image in the destination window.
;
; OUTPUTS:
; No explicit outputs.
;
; COMMON BLOCKS:
; XINTERANIMATE_COM: a private common block.
;
; SIDE EFFECTS:
; A pixmap and widget are created.
;
; RESTRICTIONS:
; Only a single copy of XINTERANIMATE can run at a time.
;
```

```
; PROCEDURE:
; When initialized, this procedure creates an approximately square
; pixmap or memory buffer, large enough to contain Nframes of
; the requested size.  Once the images are loaded, using the
; IMAGE and FRAME keywords, they are displayed by copying the images
; from the pixmap or buffer to the visible draw widget.
;
; EXAMPLE:
; Enter the following commands to open the file ABNORM.DAT (a series
; of images of a human heart) and animate the images it contains using
; XINTERANIMATE.  For a more detailed example of using XINTERANIMATE,
; see the example in the "Using IDL Widgets" chapter of "IDL Basics".
; Read the images into the variable H by entering:
;
;  OPENR, 1, FILEPATH('abnorm.dat', SUBDIR = 'images')
;  H = BYTARR(64, 64, 16)
;  READU, 1, H
;  CLOSE, 1
;  H = REBIN(H, 128, 128, 16)
;
; Initailize XINTERANIMATE with the command:
;
;  XINTERANIMATE, SET=[128, 128, 16], /SHOWLOAD
;
; Load the images into XINTERANIMATE and play the animation by entering:
;
;  FOR I=0,15 DO XINTERANIMATE, FRAME = I, IMAGE = H(*,*,I)
;  XINTERANIMATE
;
; MODIFICATION HISTORY:
; DMS, April, 1990.
; SMR, December, 1990.   Modified the XANIMATE code to work
;    interactively with widgets.
;
; DMS, March, 1991. Modified the routine to use individual pixmaps
;    for each frame of the animation.  Also added
;    the ability to read in from current IDL
;    windows directly into offscreen bitmap.
;
; SMR, March, 1991. Modified to use new XMANAGER keyword CLEANUP
;    to clean up the offscreen pixmaps when dying.
;
; SMR, Jan, 1992.  Modified the /CLOSE portion to check for a
;    valid widget before using WIDGET_CONTROL
;    and /DESTROY.
;
; AB, June 1992  Rewrite using the new CW_ANIMATE compound
;    widget. Added the KEEP_PIXMAPS keyword.
```

```
;-

PRO xintanim_kill_pix
; If there are pixmaps currently open, free them.

  COMMON XInterAnimate_com, topbase, animatebase, pwin

  i = size(pwin)
  if (i(0) ne 0) then begin  ; Not scalar, so contains valid IDs
    i = i(i(0) + 2)   ; # of elements in pwin
    FOR j=0, i-1 DO IF pwin(j) GE 0 THEN WDELETE, pwin(j) ;Delete the windows
  pwin = -1  ;Show nothing there by setting to scalar value
  endif

end




PRO xintanim_event, ev
; The only event that can be seen by this application is the "DONE"
; event from the CW_ANIMATION cluster.

  widget_control, /destroy, ev.top
END




PRO XInterAnimate, RATE, SET = SET, IMAGE = IMAGE, FRAME = FRAME, $
 ORDER = ORDER, CLOSE = CLOSE, TITLE = TITLE, $
 SHOWLOAD = SHOWLOAD, GROUP = GROUP, WINDOW = WINDOW, $
 XOFFSET = XOFFSET, YOFFSET = YOFFSET, KEEP_PIXMAPS=KEEP_PIXMAPS, $
 CYCLE = cycle, TRACK = track

COMMON XInterAnimate_com, topbase, animatebase, pwin


;---------------------- CLOSE Portion of Xinteranimate -------------------------

  IF KEYWORD_SET(CLOSE) THEN BEGIN
```

```
   if (widget_info(topbase, /valid)) THEN $
  WIDGET_CONTROL, topbase, /DESTROY
   xintanim_kill_pix
   RETURN
  ENDIF

  ;Don't allow two copies of xinternimate to run at once
  IF xregistered("XInterAnimate") THEN begin
   XANNOUNCE, 'XINTERANIMATE', 'Only one animation at a time is allowed.'
   return
  endif



;----------------------  SET Portion of Xinteranimate -----------------------

  IF KEYWORD_SET(SET) THEN BEGIN

   ;This is the first call to xinteranimate.  Here the pixmap is
   ; created and the widgets are initialized.

   xintanim_kill_pix    ;If old pixmap exists, delete

   IF NOT(KEYWORD_SET(TITLE)) THEN TITLE = "XInterAnimate"
   topbase = WIDGET_BASE(TITLE = TITLE)
   animatebase = CW_ANIMATE(topbase, set(0), set(1), set(2), $
  TRACK = KEYWORD_SET(track), CYCLE=KEYWORD_SET(cycle))

   ; If the SHOWLOAD keyword is set, realize things now so the load is seen
   IF KEYWORD_SET(SHOWLOAD) THEN $
    WIDGET_CONTROL, topbase, /REALIZE, /HOURGLASS

   RETURN
  ENDIF

;-----------------  IMAGE Loading Portion of Xinteranimate --------------------

  nwindow = N_ELEMENTS(WINDOW)
  nimage = N_ELEMENTS(image)
  if (nwindow gt 0) or (nimage gt 0) then begin
   old_window = !D.WINDOW    ;Save old window

   ; Make sure a widget has been created before trying to load it.
   if (not widget_info(topbase, /valid)) then MESSAGE, 'Not initialized'

   IF (N_ELEMENTS(YOFFSET) EQ 0) THEN YOFFSET = 0
   IF (N_ELEMENTS(XOFFSET) EQ 0) THEN XOFFSET = 0
```

```
    if (N_ELEMENTS(WINDOW) gt 0) then begin
      CW_ANIMATE_LOAD, animatebase, frame=frame, window=window, $
   XOFFSET = XOFFSET, YOFFSET = YOFFSET
    endif else begin
      IF (N_ELEMENTS(ORDER) EQ 0) THEN ORDER = 0
      CW_ANIMATE_LOAD, animatebase, frame=frame, image=image, $
   XOFFSET = XOFFSET, YOFFSET = YOFFSET, ORDER = ORDER
    endelse
    IF (old_window GE 0) THEN WSET, old_window
    RETURN
   ENDIF

;---------------  Register and Run Portion of Xinteranimate ------------------

   ; If the base is not valid, it means that we have skipped
   ; calling this routine with the SET keyword. In this case, we can restart
   ; the last animation if the KEEP_PIXMAP keyword was used to preserve them
   ; in the previous call.

   if (not widget_info(topbase, /valid)) then begin
     s = size(pwin)
     if (s(0) eq 0) then message, 'No image frames loaded'
     ; Scan the pixmaps to figure out the image size
     n = s(s(0) + 2)
     xs = 0
     for i = 0, n-1 do begin
       if pwin(i) ne -1 then begin
         old_window = !D.WINDOW
  wset, pwin(i)
         xs = !d.x_vsize
         ys = !d.y_vsize
  IF (old_window GE 0) THEN WSET, old_window
         goto, found  ; Like a "break" in C
       endif
     endfor
   found:
     if (xs ne 0) then begin
       IF NOT(KEYWORD_SET(TITLE)) THEN TITLE = 'XInterAnimate'
       topbase = WIDGET_BASE(TITLE = TITLE)
       animatebase = CW_ANIMATE(topbase, xs, ys, 0, PIXMAPS=pwin)
       ; The pixmaps are no longer our responsibility. They will get destroyed
       ; by CW_ANIMATE as usual unless this invocation of XINTERANIMATE
       ; specifies the KEEP_PIXMAP keyword.
       pwin = 0   ; Indicate that we aren't saving them anymore.
     endif else message, 'No image frames loaded'
   endif
```

```
  ; At this point, the application must be realized if it isn't already
  if (not widget_info(topbase, /realized)) then $
    WIDGET_CONTROL, topbase, /REALIZE

  ; Save the pixmaps for later restart
  if keyword_set(keep_pixmaps) then CW_ANIMATE_GETP, animatebase, pwin

  if N_ELEMENTS(RATE) EQ 0 THEN RATE = 100
  cw_animate_run, animatebase, rate

  Xmanager, "XInterAnimate", topbase, EVENT_HANDLER = "xintanim_event", $
    GROUP_LEADER = GROUP
END
```
