Subject: Re: animated gif's with IDL
Posted by Roger A. Moncrief on Tue, 24 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz wrote:
>
> Hi,
>
> I am trying to create a little gif animation using IDL (for those who
> don't know it: a powerful data analysis package). IDL has a WRITE_GIF
> routine and it allows storage of multiple gif images in one file. So far
> so good. Only, if I want to watch my animation with a web browser
> (Netscape), it runs through exactly once, then stops. Can anyone tell me
> how to change a gif file so that it loops ad infinitum (or even better:
> does anyone have an IDL routine that does this) ?
>
> Thanks,
> Martin.
>
> --
> -- -------
> Dr. Martin Schultz
> Department for Earth&Planetary Sciences, Harvard University
> 186 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA
>
> phone: (617)-496-8318
> fax : (617)-495-4551
>
> e-mail: mgs@io.harvard.edu
> IDL-homepage: http://www-as.harvard.edu/people/staff/mgs/idl/
> -- -------

Martin,
I don't know anything about your IDL software but there are over 60 .gif animation applications available. I would be very surprised any of them lacked a check box for looping the animation. Many people thing Ulead has one of the better if not the best .gif animation application around. You can download a fully functional demo at http://www.ulead.com. I seem to recall it works for either a certain number of uses or a certain time period. Its called Ulead GIF Animator 2.0. Load the gif animations from your IDL software into it, check the looping infinite check box, run the optimization wizard and save it. It will loop then. You can also control the animation speed by setting the image delay for each image. Hope this helps. Good luck.
--
Roger A. Moncrief
http://www.indrev.com

Subject: Re: animated gif's with IDL
Posted by bowman on Tue, 24 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In article <351830D2.41C6@io.harvard.edu>, Martin Schultz <mgs@io.harvard.edu> wrote:

> I am trying to create a little gif animation using IDL (for those who
> don't know it: a powerful data analysis package). IDL has a WRITE_GIF
> routine and it allows storage of multiple gif images in one file. So far
> so good. Only, if I want to watch my animation with a web browser
> (Netscape), it runs through exactly once, then stops. Can anyone tell me
> how to change a gif file so that it loops ad infinitum (or even better:
> does anyone have an IDL routine that does this) ?

For those with a Mac available, I offer the following solution. There must be similar products (to GraphicConverter) for the PC and Unix.

The following creates a set of raw image files called frame.0000, frame.0001, etc.

```
FOR frame = 0L, nframes-1L DO BEGIN
  ;Draw your graphic here

  image =TVRD()                          ;Read image from window
  OPENW, ounit, STRING(frame,FORMAT="(I4.4)"),$;Open new file for writing
    /GET_LUN
  image = ROTATE(image,7)                ;Flip image right side up
  WRITEU, ounit, image                   ;Write image to file
  FREE_LUN, ounit                        ;Release out unit and file
ENDFOR
```

Move the files to a Macintosh. Use GraphicConverter (highly-rated shareware) to convert the images to a QuickTime movie. Import the figures as filetype RAW. QuickTime lets you set the frame rate, the compressor, the image quality, etc. You can display it with a web browser, and you get a slider, so you can go backwards, forwards, jump to any frame, etc.

Regards, Ken

Subject: Re: animated gif's with IDL
Posted by thompson on Wed, 25 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In article <351830D2.41C6@io.harvard.edu>, Martin Schultz
<mgs@io.harvard.edu> wrote:

> I am trying to create a little gif animation using IDL (for those who
> don't know it: a powerful data analysis package). IDL has a WRITE_GIF
> routine and it allows storage of multiple gif images in one file. So far
> so good. Only, if I want to watch my animation with a web browser
> (Netscape), it runs through exactly once, then stops. Can anyone tell me
> how to change a gif file so that it loops ad infinitum (or even better:
> does anyone have an IDL routine that does this) ?

Apparently, the /MULTIPLE keyword was added in IDL/v5. In UNIX, we've been
using a program called whirlgif to combine a series of GIF stills written by
the older version of WRITE_GIF in IDL into a movie. This allows you to control
how many times the GIF movie loops through, or you can let it loop
indefinitely.

You should be able to find whirlgif on the Web. Here is the IDL code we use as
a front-end to whirlgif.

Bill Thompson

```
 ============================================================
==================
 PRO WR_MOVIE, FILENAME, IMAGE_ARRAY, IFRAME, NFRAMES, MPEG=MPEG, $
  GIF=K_GIF, PICT=PICT, DELETE=DELETE, FRAMEDELAY=FRAMEDELAY, $
  LOOPCNT=LOOPCNT, NOSCALE=NOSCALE, JAVASCRIPT=JAVASCRIPT, $
  INCREMENT=INCREMENT, URL=URL, TITLE=TITLE, TOP=TOP
;+
; Project     : SOHO - CDS
;
; Name        : WR_MOVIE
;
; Purpose     : Convert an array of frames to a MPEG or GIF movie.
;
; Explanation : This procedure takes a three dimensional array of images
;  in the form (X-dim, Y-dim, Frames) and converts it to a
;  series of GIF files, where each GIF contains one image
;  frame.  The procedure then will convert the GIF files to
;  a GIF movie (viewable with Netscape >2.0) and/or a MPEG
;  movie.  The procedure uses the program mpeg_encode to
;  create the MPEG movie and the program whirlgif to create the
;  GIF movie.
;
;  This procedure creates a subdirectory /mpegframes to store the
```

```
;  individual GIF image frames.  If the procedure is called with
;  the /DELETE keyword, then this directory and its contents will
;  be removed before the procedure exits.  Otherwise it will be
;  left intact.
;
;  An alternative way to use this routine is to pass it one frame
;  at a time.  This is useful when the total amount of data is too
;  large to have in memory at the same time.
;
;
; Use         : IDL> WR_MOVIE, FILENAME, MOVIE_ARRAY, /MPEG, /GIF, /DELETE
;
;  IDL> WR_MOVIE, FILENAME, MOVIE_FRAME, IFRAME, NFRAME, ...
;
; Inputs      : FILENAME: Base filename to use when creating MPEG and
;    GIF movies.  If a GIF movie is created, then
;    it will be named filename.gif.  If a MPEG
;    movie is created, then it will be called
;    filename.mpg.
;
;  IMAGE_ARRAY: Three dimensional array of images in the form
;    (X-dim, Y-dim, Frames) to be converted to
;    a movie.  This is the same form of input that
;    is used by XMOVIE.
;
; Opt. Inputs : An alternative way of calling this routine is one frame at a
;  time.  When using this option, two additional parameters are
;  required:
;
;   IFRAME:  The frame number, from 0 to N-1.
;   NFRAMES: The total number of frames.
;
;  Also, in this case IMAGE_ARRAY would be a two-dimensional frame
;  rather than a three-dimensional array of frames.
;
;  When using this approach, the following restrictions apply:
;
;  * All the frames must be exactly the same size.
;
;  * All the calls to WR_MOVIE must be with the same value of
;    NFRAMES.
;
;  * The frames must be passed in order, from 0 to NFRAMES-1.  No
;    frames can be omitted.
;
;  * If the /PICT option is desired, then it must be passed in
;    each call to WR_MOVIE.
;
```

```
;  * If the /GIF or /MPEG option is desired, then the appropriate
;    keywords must be present in the last call to WR_MOVIE.  (In
;    the preceeding calls, these keywords are ignored--the safest
;    thing to do is to use the same format for all the calls.)
;
; Outputs     : None.
;
; Opt. Outputs: None.
;
; Keywords    : The following keywords are used to determine what kind of movie
;  is produced.  If neither are passed, then the default format is
;  a GIF movie.  If both are passed, then both a GIF and an MPEG
;  movie is created.
;
;  MPEG:  If set, then a MPEG movie is created.
;
;  GIF:  If set, then a GIF movie is created.
;
;  Additional keywords are:
;
;  PICT:  If set, then a series of PICT frames are
;    written out.  This is suitable for converting
;    to a VHS tape.  The filenames will be written
;    such that the extensions are the frame number,
;    e.g. "filename.00", "filename.01", etc.
;
;  JAVASCRIPT: If set, then an Java Script HTML file is
;    written out.
;
;  DELETE:  If set, then the temporary directory
;    /mpegframes will be deleted before this
;    procedure exits.
;
;  FRAMEDELAY: The delay time between frames, in 1/100 of a
;    second.  The default value of 10 gives a movie
;    rate of approximately 10 frames/second.  The
;    bigger the number, the slower the movie will
;    run.  Not applicable to MPEG movies.
;
;  LOOPCNT: The number of times to loop through the GIF
;    movie.  The default value is 0, which
;    represents an infinite number of loops.
;
;  NOSCALE: If set, then the routine BYTSCL will not be
;    called.  Use this keyword when the movie frames
;    have already been scaled into the color table.
;
;              INCREMENT: Percent increment for speed control for Java
```

```
;    Script movies.  [def= 10]
;
;              URL:  Optional URL path to GIF images for Java Script
;    movies.  The default is that the GIF frames
;    will be in the subdirectory FILENAME.
;
;              TITLE:  Optional HTML title for Java Script movies.
;
;  TOP   The maximum value of the scaled image array, as
;    used by BYTSCL.  The default value is
;    !D.TABLE_SIZE-1.
;
; Calls       : None.
;
; Common      : The internal common block WR_MOVIE is used to keep information
;  between calls.
;
; Restrictions: This process that is running IDL when this procedure
;  executes must have write priveleges to the working directory
;  or this procedure will fail.
;
;  If a MPEG movie is being created, then the following programs
;  must be in the current path:
;
;    mpeg_encode
;    giftoppm
;
;  If a GIF movie is being created, then the following program
;  must be in the current path:
;
;    whirlgif
;
; Side effects: Files and subdirectory are created in the working directory.
;
;  When called with the single frame option, the temporary file
;  giflist is left open between calls, and is only closed on the
;  final call.
;
; Category    : Display
;
; Prev. Hist. : This procedure is based on WRITE_MPEG by
;   A. Scott Denning  scott@abyss.Atmos.ColoState.edu
;  Dept. of Atmospheric Science Phone (970)491-2134
;  Colorado State University
;
; Written     : Ron Yurow, GSFC
;
; Modified    : Version 1, 19 September 1996
```

```
;  Version 2, 6 December 1996, William Thompson, GSFC
;   Made slight modifications--better error handling.
;  Version 3, 9 December 1996, William Thompson, GSFC
;   Corrected bug writing GIF movies with LE 10 frames.
;   Corrected bug writing GIF movies with LE 10 frames.
;  Version 4, 30 December 1996, Zarro, GSFC
;                   Added FRAMEDELAY and LOOPCNT keywords
;                   Added '-f' to spawn,'rm '
;  Version 5, 8 January 1997, William Thompson, GSFC
;   Added single frame option.
;   Added keywords PICT and NOSCALE.
;  Version 6, 10 January 1997, William Thompson, GSFC
;   Changed file naming convention for PICT files
;  Version 7, 16 July 1997, William Thompson, GSFC
;   Added keywords JAVASCRIPT, INCREMENT, URL, AND TITLE
;  Version 8, 23 July 1997, William Thompson, GSFC
;   Create directory for GIF files when /JAVASCRIPT keyword
;   is used.
;  Version 9, 08-Dec-1997, William Thompson, GSFC
;   Scale image to top color.  Added keyword TOP
;
; Version     : Version 9, 08-Dec-1997
;-
;
 ON_ERROR, 2
 COMMON WR_MOVIE, GIFLST, TMPDIR
;
; Check the input parameters.
;
 CASE N_PARAMS() OF
    2:  BEGIN
  SINGLE_FRAME = 0
  IFRAME = 0
  END
    4:  BEGIN
  IF N_ELEMENTS(IFRAME) NE 1 THEN MESSAGE, $
   'IFRAME must be a scalar'
  IF N_ELEMENTS(NFRAMES) NE 1 THEN MESSAGE, $
   'NFRAMES must be a scalar'
  SINGLE_FRAME = 1
  END
    ELSE: MESSAGE, 'Syntax:  WR_MOVIE, FILENAME, ARRAY'
 ENDCASE
 IF DATATYPE(FILENAME,1) NE 'String' THEN MESSAGE, $
  'Input parameter FILENAME must be a character string'
 IF N_ELEMENTS(FILENAME) NE 1 THEN MESSAGE,  $
  'FILENAME must be a scalar value'
;
```

```
; Determine the correct setting for the GIF keyword.  If neither the GIF nor
; MPEG keyword was passed, then write out a GIF movie file.
;
 GIF = KEYWORD_SET(K_GIF)
 IF (NOT GIF) AND (NOT KEYWORD_SET(MPEG)) THEN GIF = 1
;
; Call the SIZE function in order to find the dimensions of the image array.
;
 MOVIESIZE = SIZE (IMAGE_ARRAY)
 IF SINGLE_FRAME THEN BEGIN
    IF MOVIESIZE(0) NE 2 THEN MESSAGE, $
     'Input array must be two-dimensional'
 END ELSE BEGIN
    IF MOVIESIZE(0) NE 3 THEN MESSAGE, $
     'Input array must be three-dimensional'
 ENDELSE
;
; Set the X dimension and Y dimension of each frame as well as the number of
; frames in the movie based on the size of image_array.
;
 XSIZE = MOVIESIZE(1)
 YSIZE = MOVIESIZE(2)
 IF NOT SINGLE_FRAME THEN NFRAMES = MOVIESIZE(3)
;
; Set NDIGITS to the minimum field length required to display largest frame
; number.  Can't be less than 2, or whirlgif won't work.
;
 NDIGITS = (1 + FIX (ALOG10 (NFRAMES))) > 2
;
; Set FRMT to a format string that will result in the frame number of each
; frame using the same number of characters with 0's padding left side as
; needed.
;
 FRMT = '(i' + STRING (NDIGITS) + '.' + STRING (NDIGITS) + ')'
 FRMT = STRCOMPRESS(FRMT, /REMOVE_ALL)
;
; If we screw up writing a frame, we will just have to give up and go home.
;
 ON_IOERROR, BADWRITE
;
; Make a temporary directory to hold the individual frames of the the movie
; with each frame stored as GIF file.  Begin by setting TMPDIR to the name
; of the tempory directory to create or clear.
;
 IF (NOT SINGLE_FRAME) OR (IFRAME EQ 0) THEN BEGIN
    TMPDIR = 'mpegframes'
;
; Now make a command to clear the temporary directory (RMCMD) and a command to
```

```
; create the temporary directory (CRCMD).  We will decide which one to use
; latter.
;
    RMCMD = 'rm -f ' + TMPDIR + '/*'
    CRCMD = 'mkdir ' + TMPDIR
;
; Spawn a command to determine if the tempory command already exists.
;
    SPAWN, 'if (-d ' + TMPDIR + ') echo "exists"', RESULT
;
; Check the contents of the array RESULT.  If the temporary directory already
; exists, then RESULT (0) will be set to the string 'exists'.
;
    DIREXISTS = RESULT (0) EQ 'exists'
;
; Clear or add the directory as needed.
;
    IF DIREXISTS THEN SPAWN, RMCMD ELSE SPAWN, CRCMD
;
; Open a file for recording which GIFs we created.
;
    OPENW, GIFLST, TMPDIR + "/giflist", /GET_LUN
 ENDIF
;
; Write each frame into TMPDIR as a gif image file
;
 IF SINGLE_FRAME THEN BEGIN
    FRAME1 = IFRAME
    FRAME2 = IFRAME
 END ELSE BEGIN
    FRAME1 = 0
    FRAME2 = NFRAMES - 1
 ENDELSE
;
; If autoscaling is to be used, then get the minimum and maximum values to use
; for scaling the images.
;
 IF NOT KEYWORD_SET(NOSCALE) THEN IMIN = MIN(IMAGE_ARRAY, MAX=IMAX)
;
 FOR FRAME = FRAME1,FRAME2 DO BEGIN
;
; Let FILENAME be the filename of the GIF we are createing.
;
    FRAMENAME = 'frame' + STRING (FRAME, FORMAT = FRMT) + '.gif'
;
; Let PATHNAME be the entire file and path of the GIF we are createing.
;
    PATHNAME = TMPDIR + '/' + FRAMENAME
```

```
    IF SINGLE_FRAME THEN IMAGE = IMAGE_ARRAY ELSE $
          IMAGE =  REFORM (IMAGE_ARRAY (*, *, FRAME))
    IF N_ELEMENTS(TOP) NE 1 THEN TOP = !D.TABLE_SIZE-1
    IF NOT KEYWORD_SET(NOSCALE) THEN $
     IMAGE = BYTSCL(IMAGE, TOP=TOP, MIN=IMIN, MAX=IMAX)
         WRITE_GIF, PATHNAME, IMAGE
;
; Write the file name into our list of GIFs.
;
      PRINTF, GIFLST, FRAMENAME
      PRINT, 'Wrote temporary GIF file for frame ', FRAME + 1
;
;  If the PICT keyword was passed, then also write the PICT file.
;
    IF KEYWORD_SET(PICT) THEN WRITE_PICT, FILENAME + '.' + $
     STRING(FRAME, FORMAT=FRMT), IMAGE
 ENDFOR
;
; If only processing a single frame, and it's not the last frame, then we can
; return now.
;
 IF SINGLE_FRAME AND ((IFRAME+1) NE NFRAMES) THEN RETURN
;
; Close the file contianing our list of GIFs
;
 FREE_LUN, GIFLST
;
; Set FRAMEDELAY and LOOPCNT to determine how the GIF will be displayed by
; netscape.
;
      IF NOT EXIST(FRAMEDELAY) THEN FRAMEDELAY = 10 ; 10 frames per second
 IF NOT EXIST(LOOPCNT) THEN LOOPCNT    = 0  ; Infinite repititions
;
; Check if we should create a GIF movie.  If we do, then we will have to
; create a command to SPAWN whirlgif.
;
 IF KEYWORD_SET (GIF) THEN BEGIN
;
; Create a unix command to make GIF movie from all of our gif frames
;
   CMD = "cd " + TMPDIR + " ; "
   CMD = CMD + "whirlgif"
         CMD = CMD + " -loop " + STRING (LOOPCNT, FORMAT = FRMT)
   CMD = CMD + " -time " + STRING (FRAMEDELAY, FORMAT = FRMT)
   FNAME = FORM_FILENAME(FILENAME,".gif")
   CMD = CMD + " -o " + FNAME + " -i giflist"
;
; Spawn the command.
```

```
;
   SPAWN, CMD
;
; Move the resulting GIF file out of the temporary frames directory.
;
   SPAWN, 'mv ' + TMPDIR + '/' + FNAME + ' .'
 ENDIF
;
; Check if we should create MPEG movie.  If we do, we will have
; set up parameter file for mpeg_encode and then SPAWN mpeg_encode.
;
 IF KEYWORD_SET (MPEG) THEN BEGIN
;
; Set up the name and path to the mpeg parameter file
;
   PARAMFILE = TMPDIR + '/params.mpeg'
;
; Open the mpeg parameter file for writing.  Store logical unit number in
; MPRM.
;
   OPENW, MPRM, PARAMFILE, /GET_LUN
;
; Write out the mpeg parameter file.
;
   PRINTF, MPRM, 'PATTERN         IBBBBBBBBBBP'
   PRINTF, MPRM, 'OUTPUT          ' + FORM_FILENAME(FILENAME,'.mpg')
   PRINTF, MPRM, 'GOP_SIZE 12'
   PRINTF, MPRM, 'SLICES_PER_FRAME 5'
   PRINTF, MPRM, 'BASE_FILE_FORMAT PPM'
   PRINTF, MPRM, 'INPUT_CONVERT giftoppm *'
   PRINTF, MPRM, 'INPUT_DIR       ' + TMPDIR
   PRINTF, MPRM, 'INPUT'
   PRINTF, MPRM, 'frame*.gif [' + STRING(0,FORMAT=FRMT) + '-' + $
  STRING(NFRAMES-1,FORMAT=FRMT) + ']'
   PRINTF, MPRM, 'END_INPUT'
   PRINTF, MPRM, 'PIXEL           FULL'
   PRINTF, MPRM, 'RANGE           5'
   PRINTF, MPRM, 'PSEARCH_ALG     LOGARITHMIC'
   PRINTF, MPRM, 'BSEARCH_ALG     SIMPLE'
   PRINTF, MPRM, 'IQSCALE         6'
   PRINTF, MPRM, 'PQSCALE         6'
   PRINTF, MPRM, 'BQSCALE         6'
   PRINTF, MPRM, 'REFERENCE_FRAME  ORIGINAL'
   PRINTF, MPRM, 'FORCE_ENCODE_LAST_FRAME'
;
; Close the mpeg parameter file.
;
   FREE_LUN, MPRM
```

```
;
; Spawn a shell to process the mpeg_encode command.
;
   SPAWN, 'mpeg_encode ' + PARAMFILE
 ENDIF
;
; Check to see if the Java Script HTML file should be written.
;
 IF KEYWORD_SET(JAVASCRIPT) THEN BEGIN
    NAMES = 'frame' + STRING(INDGEN(NFRAMES), FORMAT=FRMT) + '.gif'
    IF N_ELEMENTS(FRAMEDELAY) EQ 1 THEN DELAY = FRAMEDELAY*10.
    IF N_ELEMENTS(URL) EQ 0 THEN URL = FILENAME
    JSMOVIE, FILENAME, NAMES, DELAY=DELAY, TITLE=TITLE, URL=URL, $
     SIZE=[XSIZE,YSIZE], INCREMENT=INCREMENT
;
; Create the directory to hold the GIF files, and move the files to that
; directory.
;
    SPAWN, 'if (-d ' + FILENAME + ') echo "exists"', RESULT
    IF RESULT(0) EQ 'exists' THEN SPAWN, 'rm -f ' + FILENAME + '/*' $
     ELSE SPAWN, 'mkdir ' + FILENAME
    IF KEYWORD_SET(DELETE) THEN     $
     SPAWN, 'mv ' + TMPDIR + '/frame*.gif ' + FILENAME ELSE $
     SPAWN, 'cp ' + TMPDIR + '/frame*.gif ' + FILENAME
 ENDIF
;
; Check if we should remove the temporary frames directory that we just
; created.
;
 IF KEYWORD_SET(DELETE) THEN SPAWN, 'rm -rf ' + TMPDIR
 RETURN
;
BADWRITE:
 MESSAGE, 'Unable to write MPEG and/or GIF file!'
 END
```