Subject: changing contrast and brightness on the fly
Posted by Simon Williams on Sat, 16 Jun 2001 00:13:03 GMT
View Forum Message <> Reply to Message

Greetings,

I'm looking for tips on how to implement an image display feature that's bugging me. I'm new to widget programming, trying to get up to speed with David Fanning's book and other helps, but any short-cuts would be appreciated.

The job is to display one or more grayscale MRI images (mostly small, 256 pixels squared is common) and then to be able to adjust the image brightness and contrast interactively, without going to any special widgets like sliders. The functionality I have in mind is to middle-click the image and then have drag right/left control contrast and drag up/down control brightness.

The plan is to replicate functionality that the end users (radiology folks) are already familiar with from other image viewers. In the longer term plan, the display would also be re-sizable and allow interactive ROI drawing as well.

The functionality I have in mind is to middle-click the image and then have drag right/left control contrast and drag up/down control brightness.

It sounds easy but I can't find anything similar described on the web etc. to use as a suitable starting point. Question: is changing the (familiar to me) contrast and brightness equivalent to re-defining the color table and re-scaling the data (more often discussed with color images). What parameters really define contrast and brightness?

Thanks for any help you can offer,

Simon Williams

Subject: Re: changing contrast and brightness on the fly
Posted by Ken Mankoff on Sat, 16 Jun 2001 02:24:47 GMT
View Forum Message <> Reply to Message

>
> I'm looking for tips on how to implement an image display feature that's
> bugging me. I'm new to widget programming, trying to get up to speed
> with David Fanning's book and other helps, but any short-cuts would be
> appreciated.
>

> The job is to display an MRI image and to be able to adjust the image
> brightness and contrast interactively, without going to any special
> widgets like sliders. The control I have in mind is to
> middle-click the image and then have drag right/left control contrast
> and drag up/down control brightness.
>

I think brightness adjustment is achieved with alpha channels. May require
IDL Object Graphics. There are other ways. Search the groups.google.com
archive for recent threads on this topic.

IDL> ? cursor ; for your
IDL> ? !mouse ; second question.

> It sounds easy but I can't find anything similar described on the web
> etc. to use as a suitable starting point. Question: is changing the
> contrast and brightness to be achieved by re-defining the
> color table and re-scaling the data?

your choice to use data or color scale. use bytscl(), lt & gt to rescale
the data. Probably easier (for debugging too) to do it on data.

-k.

--
Ken Mankoff
LASP://303.492.3264
http://lasp.colorado.edu/~mankoff/

Subject: Re: changing contrast and brightness on the fly
Posted by david[2] on Sat, 16 Jun 2001 02:43:18 GMT
View Forum Message <> Reply to Message

Simon Williams writes:

> I'm looking for tips on how to implement an image display feature that's
> bugging me. I'm new to widget programming, trying to get up to speed
> with David Fanning's book and other helps, but any short-cuts would be
> appreciated.
>
> The job is to display an MRI image and to be able to adjust the image
> brightness and contrast interactively, without going to any special
> widgets like sliders. The control I have in mind is to
> middle-click the image and then have drag right/left control contrast
> and drag up/down control brightness.
>
> The plan is to replicate functionality that the end users (radiology

> folks) are already familiar with from other image viewers. In the
> longer term plan, the display would also be re-sizable and allow
> interactive ROI drawing as well.
>
> It sounds easy but I can't find anything similar described on the web
> etc. to use as a suitable starting point.

Sometimes the things that sound the easiest are really
the hardest. :-)

But in this case, I think you are right. It's
fairly easy. The hard part (it seems to me) is
coming up with the appropriate equation to
go from a movement in pixel space to a change
in contract/brightness. I remember working with
Phil Williams when he was at Children's Hospital
in Cinncinatti and he had a gadget just like this.
I can't recall the equation how, but I seem to
remember it was some kind of high energy particle
field equation, or something. He had chanced upon
it from a previous life as a real physicist. Great
use for it though.

In any case, I think it is likely someone will
offer you something very like this, or at least
a better idea than I can give you. It is a
common feature of medical image processing.
I wouldn't be surprised to see the algorithm in
a medical imaging book.

> Question: is changing the
> contrast and brightness to be achieved by re-defining the
> color table and re-scaling the data?

Yes, I think this is exactly what you need to do.
If you can just find the right scaling equation...

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: changing contrast and brightness on the fly
Posted by david[2] on Sat, 16 Jun 2001 02:55:22 GMT
View Forum Message <> Reply to Message

Ken Mankoff writes:
>
> I think brightness adjustment is achieved with alpha channels. May require
> IDL Object Graphics. There are other ways. Search the groups.google.com
> archive for recent threads on this topic.
>
> IDL> ? cursor ; for your
> IDL> ? !mouse ; second question.

Uh, I don't think I would be recommending object
graphics on the one hand, and the Cursor command (for
God's sake!) on the other!

Stick with direct graphics in a draw widget
without a Cursor command, is my advice. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: changing contrast and brightness on the fly
Posted by Richard Tyc on Mon, 18 Jun 2001 14:42:40 GMT
View Forum Message <> Reply to Message

I do something like this in our medical imaging application which also manipulates MR images retrieved via DICOM and allows the user to interactively adjust image settings. However, I included sliders because I already use the mouse for other things ie. left mouse selects ROI's, middle mouse zooms in out (click, drag up ->zoom in, drag down -> zoom out) and right mouse moves the image around within the view window. I thought adding contrast/brightness would make the mouse too busy and you don't see the values.

Anyways, I implemented it in object graphics. It really becomes a image width/level problem rather than brightness/contrast, at least that is what our Siemens MR console does with the mouse manipulation's you refer to.

This is because the MR data is actually 12 bit (or 0-4095) and the display only shows 256 gray levels so you are simply windowing within the 12 bit scale for the display.

I simply calculate the Image Max and Min values from the Width/Level values and then use bytscl to create a new image from the original 12 bit data.
The level slider has a range of 0-2047 and the width slider 0-4095.

```
ImgMax =  LevVal + WidVal/2
IF ImgMax GT 4095 THEN ImgMax = 4095

ImgMin =  LevVal - WidVal/2
IF ImgMin LT 0 THEN ImgMin = 0

; LgImg is the 12 bit original MR data (INTARR(256,256))
ImgData = BYTSCL( LgImg, MAX= ImgMax, MIN = ImgMin, $
          TOP= !D.TABLE_SIZE-1 )

oSliceImg->SetProperty, data= ImgData
oWindow->Draw, oView
```

To add mouse interactivity, you add motion and button events to your WIDGET_DRAW widget and then in your event
handler for your draw widget, you need something like this case statement note: this is what I do for interactive zooming, you could modify this for width/level

```
'wMainDraw' : BEGIN

  ;Motion Events
    IF (sEvent.type EQ 2) THEN BEGIN
      IF (*pState).btndown EQ 2b THEN BEGIN    ; left mouse 1b,
middle mouse is 2b, right mouse 4b

       ;This is the distance moved since you pressed the middle
mouse and held it during movement
        dataxy =   [sEvent.x - (*pState).ZoomPos[0], $
                   sEvent.y - (*pState).ZoomPos[1] ]

        ; now add your code to adjust window level.width like above
and redraw



        oWindow->Draw, oView

      ENDIF
    ENDIF
```

```
        ; Handle other events.
        ;  Button press.
        IF (sEvent.type EQ 0) THEN BEGIN
          IF sEvent.press EQ 2 THEN BEGIN  ;MIDDLE Mouse
            ;On button press, hold current value of mouse position
            (*pState).ZoomPos = [sEvent.x,sEvent.y]
            (*pState).btndown = 2b
            ; let widget emit motion events while mouse is moving
            WIDGET_CONTROL, (*pState).wDraw, /DRAW_MOTION
          ENDIF
        ENDIF

        ; Button release.
        IF (sEvent.type EQ 1) THEN BEGIN
          (*pState).btndown = 0b
          WIDGET_CONTROL, (*pState).wDraw, DRAW_MOTION=0
        ENDIF

    END
```

Email me if you need any more help.

Richard Tyc
Project Engineer
St. Boniface Hospital Research Center
351 Tache Ave
Winnipeg, MB R2H 2A6
Canada

Tel: 204-237-2557
Fax: 204-231-1164
email: richt@sbrc.ca

Ken Mankoff <mankoff@I.HATE.SPAM.cs.colorado.edu> wrote in message
news:Pine.LNX.4.33.0106152012570.32515-100000@snoe.colorado. edu...
>>
>> I'm looking for tips on how to implement an image display feature that's
>> bugging me. I'm new to widget programming, trying to get up to speed
>> with David Fanning's book and other helps, but any short-cuts would be
>> appreciated.
>>
>> The job is to display an MRI image and to be able to adjust the image
>> brightness and contrast interactively, without going to any special
>> widgets like sliders. The control I have in mind is to

>> middle-click the image and then have drag right/left control contrast
>> and drag up/down control brightness.
>>
>
> I think brightness adjustment is achieved with alpha channels. May require
> IDL Object Graphics. There are other ways. Search the groups.google.com
> archive for recent threads on this topic.
>
> IDL> ? cursor ; for your
> IDL> ? !mouse ; second question.
>
>> It sounds easy but I can't find anything similar described on the web
>> etc. to use as a suitable starting point. Question: is changing the
>> contrast and brightness to be achieved by re-defining the
>> color table and re-scaling the data?
>
> your choice to use data or color scale. use bytscl(), lt & gt to rescale
> the data. Probably easier (for debugging too) to do it on data.
>
> -k.
>
> --
> Ken Mankoff
> LASP://303.492.3264
> http://lasp.colorado.edu/~mankoff/
>
>

Subject: Re: changing contrast and brightness on the fly
Posted by m.hadfield on Tue, 19 Jun 2001 01:41:36 GMT
View Forum Message <> Reply to Message

From: "Simon Williams" <williams.simon@gene.com>

> When you see how the end-users operate, having the slider-less control
makes
> good sense. The images are not being displayed for quantitation but for

> "expert review" by a radiologist who does all the feature recognition,
often
> rather rapidly. They want to be able to stare at the image and change the
> contrast and brightness without looking away from the image, which is when
> slider controls get to be irritating. So, the existing software is, for
its
> purpose, very satisfactory indeed.

I stand corrected.

It was an interesting question.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

--

Subject: Re: changing contrast and brightness on the fly
Posted by Liam E. Gumley on Wed, 20 Jun 2001 14:11:09 GMT
View Forum Message <> Reply to Message

JD Smith wrote:
> A standard feature of astronomical viewers. For a solution which uses
> only colormap fiddling (vs. full image rescaling), see atv:
>
> http://cfa-www.harvard.edu/~abarth/atv/atv.html
>
> This brings up the age old question of how best to dynamically redisplay
> images (brightnest/contrast/etc.). With only 255 colors, making the top
> 100 white to increase brightness is pretty wasteful, and cuts down on
> the dynamic visual range... much better (and slower) is to rescale the
> image range of interest into the full colormap.
>
> Here's how Andrew (and cohorts) did it:
>
> ++
> pro atv_stretchct, brightness, contrast, getmouse = getmouse
>
> ; routine to change color stretch for given values of
> ; brightness and contrast.
> ; Complete rewrite 2000-Sep-21 - Doug Finkbeiner
> ; This routine is now shorter and easier to understand.
>
> common atv_state
> common atv_color
>
> ; if GETMOUSE then assume mouse positoin passed; otherwise ignore
> ; inputs
>
> if (keyword_set(getmouse)) then begin
>   state.brightness = brightness/float(state.draw_window_size[0])

```
>    state.contrast = contrast/float(state.draw_window_size[1])
>  endif
>
>  x = state.brightness*(state.ncolors-1)
>  y = state.contrast*(state.ncolors-1) > 2   ; Minor change by AJB
>  high = x+y & low = x-y
>  diff = (high-low) > 1
>
>  slope = float(state.ncolors-1)/diff ;Scale to range of 0 : nc-1
>  intercept = -slope*low
>  p = long(findgen(state.ncolors)*slope+intercept) ;subscripts to select
>  tvlct, r_vector[p], g_vector[p], b_vector[p], 8
>
>  end
>  ++++++++++++++++++++++++++++++++++++++++++++
```

Is it fair to say that this method will only give satisfactory results
when IDL is running in 8-bit display mode?

Cheers,
Liam.

Subject: Re: changing contrast and brightness on the fly
Posted by John-David T. Smith on Wed, 20 Jun 2001 20:26:08 GMT
View Forum Message <> Reply to Message

"Liam E. Gumley" wrote:
>
> JD Smith wrote:
>> A standard feature of astronomical viewers. For a solution which uses
>> only colormap fiddling (vs. full image rescaling), see atv:
>>
>> http://cfa-www.harvard.edu/~abarth/atv/atv.html
>>
>> This brings up the age old question of how best to dynamically redisplay
>> images (brightnest/contrast/etc.). With only 255 colors, making the top
>> 100 white to increase brightness is pretty wasteful, and cuts down on
>> the dynamic visual range... much better (and slower) is to rescale the
>> image range of interest into the full colormap.
>>
>> Here's how Andrew (and cohorts) did it:
>>
>> ++
>> pro atv_stretchct, brightness, contrast, getmouse = getmouse
>>
>> ; routine to change color stretch for given values of

```
>>  ; brightness and contrast.
>>  ; Complete rewrite 2000-Sep-21 - Doug Finkbeiner
>>  ; This routine is now shorter and easier to understand.
>>
>>  common atv_state
>>  common atv_color
>>
>>  ; if GETMOUSE then assume mouse positoin passed; otherwise ignore
>>  ; inputs
>>
>>  if (keyword_set(getmouse)) then begin
>>     state.brightness = brightness/float(state.draw_window_size[0])
>>     state.contrast = contrast/float(state.draw_window_size[1])
>>  endif
>>
>>  x = state.brightness*(state.ncolors-1)
>>  y = state.contrast*(state.ncolors-1) > 2   ; Minor change by AJB
>>  high = x+y & low = x-y
>>  diff = (high-low) > 1
>>
>>  slope = float(state.ncolors-1)/diff ;Scale to range of 0 : nc-1
>>  intercept = -slope*low
>>  p = long(findgen(state.ncolors)*slope+intercept) ;subscripts to select
>>  tvlct, r_vector[p], g_vector[p], b_vector[p], 8
>>
>>  end
>>  ++++++++++++++++++++++++++++++++++++++++++++
>
>  Is it fair to say that this method will only give satisfactory results
>  when IDL is running in 8-bit display mode?
```

Umm, not necessarily. ATV for instance just issues a redisplay if you're using 24 bit color. The actual breakdown I've discovered (please correct any errors) is:

Psuedo-Color: Shared colormap (usally 8bit), no redisplay necessary. The hardware colormap is read-writeable.

Direct-Color: One large colormap for each of R/G/B, no redisplay necessary. The hardware colormap is read-writeable.

True-Color: No real colormap, colors are expressed in absolute terms. Redisplay necessary (which will usually be fairly much slower than direct manipulation of the hardware color table). IDL will maintain a software translation colormap for you, with decomposed=0. A linear ramp of RGB colors is presumed, and you can't change this directly (the hardware colormaps is read only).

Some machines (usually commercial unices) support multiple visuals at once, a capability termed "overlays". In this case, you can say device,PSUEDO=8, and be up and running, able to write the underlying hardware colormap for fast color-changing. On the PC side, typically you only have one visual class available at a time (and this includes linux).

The pros and cons are:

1. TrueColor:

Pros: Millions of colors, and straightforward to get exactly the color you want, since noone can muck with the underlying colormap (a linear ramp in R,G, and B space)

Cons: Read-only colormap, requiring slow redraws and software-only color translation for normal color table operation.

2. PsuedoColor:

Pros: Lightening fast color manipulation, since you're just loading a table into the display hardware. No redisplay required.

Cons: Usually only 255 colors. Colormap flashing may result (even with IDL widgets... yuck). This is not a problem for overlayed PsuedoColor visuals (which get their own little 8-bit colorspace to play in). Overlays have limited availability.

3. DirectColor:

Pros: Colormaps can be fiddled, separately for RGB. Redisplay probably *not* required.

Cons: Not as fast at color operations as Psuedo-color. It's quite possible that when using the DECOMPOSED=0 flag with DirectColor, IDL performs the exact same color translation as for TrueColor (though it doesn't need to), and then sets the hardware colormap.

JD

Subject: Re: changing contrast and brightness on the fly
Posted by slashell on Thu, 21 Jun 2001 16:55:29 GMT

View Forum Message <> Reply to Message

Perhaps no one has said this because it's obvious, but just in case:

Level and width (as described by whatshisname above) is a more natural

way to write your code. If people want brightness and contrast, the connections are as follows:

Higher contrast means a more narrow window
Brighter image means a lower level

I don't know what the convention is for units of contrast and brightness, but for example:

```
a = congrid(indgen(2^6,2^6),300,300) ; dummy image with range of 2^12

aMin = min(a,max=aMax) ; get max and min values of raw image

; scale all 12 bits of data into 8 bits of display
; one might call this contrast = 0 (out of 100) and brightness = 50 (out of 100)
tv,bytscl(a)

; now, to display some other contrast and brightness
contrast = 50
brightness = 25

Level = (1-brightness/100.)*(aMax - aMin) + aMin
width = (1-contrast/100.)*(aMax - aMin)

displayMax = level + width/2
displayMin = level - width/2

tv,bytscl(a,min=displayMin,max=displayMax)
```

Hope this helps,
Sean La Shell

Massachusetts General Hospital
Radiation Oncology
Northeast Proton Therapy Center
