Subject: 24 bit colors in IDL
Posted by Richard Penrose on Tue, 03 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

I am working on a piece of software written in IDL that has always assumed the color resolution to be 256 colours. I would like to be able to use the same piece of software on a machine which has a flashy graphics card and only allows a colour resolution of 24 bits (True Colour).

At the moment when I change the colour resolution it messes up all of the software's colours. The piece of code that we use to set up the colours is as follows:

```
  iaR
=                                                $
   [255,1,192,128,255,142,255,213,  0,  0,  0,  0,  0,
0,255,197,248,247]
  iaG
=                                                $
   [255,1,192,128,  0,  0,255,213,255,176,255,171,  0,
0,192,138,208,167]
  iaB
=                                                $
   [255,1,192,128,  0,  0,  0,  0,  0,  0,255,171,255,185,  0,
0,198,195]

  tvlct, iaR, iaG, iaB

  st_color = { $
     i_white              :     0                          ,
$
     i_black              :     1                          ,
$
     i_light_grey         :     2,
$
     i_dark_grey          :     3,
$
     i_light_red          :     4,
$
     i_red                :     4,
$
     i_dark_red           :     5,
$
     i_light_yellow       :     6,
$
     i_yellow             :     6,
```

```
$
    i_dark_yellow        :    7,
$
    i_light_green        :    8,
$
    i_dark_green         :    9,
$
    i_light_cyan         :   10,
$
    i_dark_cyan          :   11,
$
    i_light_blue         :   12,
$
    i_dark_blue          :   13,
$
    i_light_orange       :   14,
$
    i_orange             :   14,
$
    i_dark_orange        :   15,
$
    i_light_pink         :   16,
$
    i_dark_pink          :   17
$
    }
```

We then use the st_color structure to access the iaR, iaG, iaB arrays and get the desired color, e.g. [1,1,1] should give black and [255,255,255] should give white.

Now it seemed obvious to me that to upgrade to 24 bit colours all I needed to do was to
multiply each element of the iaR, iaG, iaB arrays by 65536 and subtract 1. Disappointingly this does not seem to work!

Can anyone help.

Cheers Richard

Subject: Re: 24 bit colors in IDL
Posted by davidf7203 on Fri, 06 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Philip Aldis <teal@dera.gov.uk> writes in response to
a 24-bit color question:

> I hope I've explained how the 24 bit colours work, it's a bit hard to
> get your head round. If what I've said solves the problem, then please post
> another reply, just to let me know, because I only just a beginner and it
> helps if what I think is right is confirmed to be right (or quite probably
> wrong!!)

No, no, lad. You are spot on!

Cheers,

David

Subject: Re: 24 bit colors in IDL
Posted by philip aldis on Fri, 06 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Richard Penrose wrote:

> I am working on a piece of software written in IDL that has always
> assumed the color resolution to be 256 colours. I would like to be able
> to use the same piece of software on a machine which has a flashy
> graphics card and only allows a colour resolution of 24 bits (True
> Colour).
>
> At the moment when I change the colour resolution it messes up all of
> the software's colours. The piece of code that we use to set up the
> colours is as follows:
>
> -----code deleted----------

>
> Now it seemed obvious to me that to upgrade to 24 bit colours all I
> needed to do was to
> multiply each element of the iaR, iaG, iaB arrays by 65536 and subtract
> 1. Disappointingly this does not seem to work!
>
> Can anyone help.
>
> Cheers Richard

Before I start, check out, http://www.dfanning.com/tips/colors24.html,
which has some tips on 24 bit colour.
I don't know whether you know the difference between 24 bit colour and
ordinary colour, but I will attempt to give a brief explanation. With the

sort of colour that you have been using in the past, colours are not specified as the exact colour, i.e. a pixel does not say, I have 255 parts red, 50 parts green and 70 parts blue. Instead it has a colour index which ranges from 0 to 255. This number is referring to a colour map, a table which has 256 colours and for each colour index the table gives the RGB triple to describe that colour. The pixel is impervious to its actual colour, it only knows its index. If you made every index in the colour map read RED, 255, GREEN, 0, BLUE, 0 (by using tvlct)and then displayed an image, it would be entirely red. If you then loaded all GREENS into the colour map, the image displayed on your screen would become all green. This is because, as I said, the pixels don't know what colour they are, they simply know what index in the colour map tells them what colour to be, so when you change the actual values in the colour map, by loading in all greens, the pixels also change colour.

Here lies the fundamental difference between 24 bit colour, and the colour maps - 8 bit colour. With 24 bit colour each pixel can tell you exactly what colour it is, and no amount of fiddling can get it to change it. This is because each pixel, instead of now having an 8 bit number, 0 to 255, which is an index in a colour map, it stores a 24 bit value, 0 - 16.7 million (recognise that number), which tells the pixel exactly what colour to be. The number is a long and contains 8 bits of red, 8 bits of green and 8 bits of blue. This type of colour is called decomposed colour.

Now as far as I can see there are two options of how you can deal with this. Decomposed colour can be switched on or off using device, decomposed = 0 or device, decomposed = 1. So, you can make a 24 bit system still use colour maps. If you put device, decomposed=0 at the start of your program then I'm pretty sure that everything would then work as normal. The advantage though now is that although you are using colour maps, you are not using them in quite the same way as on an 8 bit system. With the 8 bit system, as I said, when you load in a new colour table what's on the screen changes. However if you are using 24 bit colour but with a colour map, the map is merely a translator and what is stored for the pixels is still unchangeable. This is of course a great advantage because you can have different colour tables for each window. This would mean that all the old programs would work but at the same time you would be reaping the rewards of a 24 bit system.

The second option is to go decomposed, and specify all your colours using these 24 bit longs to access the colours. e.g. plot, findgen(100), color=65000 which gives a yellowey colour, no matter what machine you're on and there are no colour table dependencies. David Fanning has a program which converts an RGB triple to the 24 bit long - on the web site I gave at the start. So you could take your arrays and pass them to this program and your st_color array would contain 24 bit longs which were the right colours. I personally think that if your just using colours on plots, or in fact anything except a great deal of image processing then decomposed colour is not really worth, although I'm sure that a programmer with a lot more experience than me will tell you that I'm wrong - so don't take my advice as

gospel.

I hope I've explained how the 24 bit colours work, it's a bit hard to get your head round. If what I've said solves the problem, then please post another reply, just to let me know, because I only just a beginner and it helps if what I think is right is confirmed to be right (or quite probably wrong!!)

As a postscript, if there's anyone out there who does animations please e-mail me or post a message to let me know if you would be interested in a replacement for xinteranimate which vastly reduces on pixmap memory usage and enables custom animations to be built very easily. In fact I would boast that there isn't any animation that you couldn't create in under half an hour although for most cases it would be less. You could even have animations where one image was moving every frame and then the other was moving every 3 frames, and for the second animation you would only store pixmaps for when it was moving, and also any black space in your animations are not stored either.

cheers,
Phil Aldis

Subject: Re: 24 bit colors in IDL
Posted by Richard Penrose on Wed, 11 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In reply to my own initial problem, I found out that by putting DEVICE, DECOMPOSED=0 all was resolved, amazing!!!!!

I noticed that this was what philip suggested, cheers!

Regards

Richard

Subject: Re: 24 bit colors in IDL
Posted by davidf on Thu, 12 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Richard Penrose (rpenrose@hhhh.ibm.com) writes:

> In reply to my own initial problem, I found out that by putting DEVICE,
> DECOMPOSED=0 all was resolved, amazing!!!!!

Indeed.

So here is a question that is sure to stump you.
How come this isn't the default when IDL starts up?

Cheers,

David

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

Subject: Re: 24 bit colors in IDL
Posted by Andy Bristow on Tue, 17 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Richard Penrose wrote:
>
> In reply to my own initial problem, I found out that by putting DEVICE,
> DECOMPOSED=0 all was resolved, amazing!!!!!
>
> I noticed that this was what philip suggested, cheers!
>
> Regards
>
> Richard

I was just getting my head round 24-bit, having been prompted by this
thread. So, at Philip's suggestion I went to

http://www.dfanning.com/tips/colors24.html

All well and good. Sounds relatively straightforward.

So on my SGI (IRIX 6.5.1, IDL 5.1) I tried some of the
suggested code (immediatley after starting IDL up):

```
device,decomposed=0
tvlct,[[255],[255],[0]],100
plot,randomu(10,10),color=100
```

expecting the plot to be yellow, as advertised. Except no, _I_
get a medium-light shade of grey!

Any suggestions?

Andy

--
Andy Bristow

The views expressed above are entirely those of the writer
and do not represent the views, policy, or understanding of
any other person or official body.

Subject: Re: 24 bit colors in IDL
Posted by davidf on Tue, 17 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Andy Bristow (ajbristow@dera.gov.uk) writes:

> I was just getting my head round 24-bit, having been prompted by this
> thread. So, at Philip's suggestion I went to
>
> http://www.dfanning.com/tips/colors24.html
>
> All well and good. Sounds relatively straightforward.
>
> So on my SGI (IRIX 6.5.1, IDL 5.1) I tried some of the
> suggested code (immediatley after starting IDL up):
>
> device,decomposed=0
> tvlct,[[255],[255],[0]],100
> plot,randomu(10,10),color=100
>
> expecting the plot to be yellow, as advertised. Except no, _I_
> get a medium-light shade of grey!
>
> Any suggestions?

Oh, dear. I think this is one of those TrueColor/DirectColor
tricks that SGI engineers like to play on people. This is
a complication that I don't *even* want to know about.

Try this. Type this:

```
Device, Get_Visual_Name=thisName
Print, thisName
```

If thisName is "TrueColor" change it to "DirectColor":

Device, DirectColor=24

If thisName is "DirectColor" change it to "TrueColor":

Device, TrueColor=24

Do this *before* you open any graphics windows in IDL.
Try the code above again. Different? The same?

Please let us know. :-(

Cheers,

David

Note: A copy of this article was e-mailed to the original poster.

Subject: Re: 24 bit colors in IDL
Posted by davidf on Wed, 18 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Nigel Wade (nmw@ion.le.ac.uk) writes:

> Why do you want to blame SGI?
>
> If IDL asks for a DirectColor visual and gets what's wrong
> with that?

Geez, Nigel. Bad day? You must be the systems
administrator for a bunch of SGIs. :-)

My apologies. IDL does in fact look for a DirectColor
visual first. The problem with a DirectColor visual is
that it is not writeable. That is to say, you can't load
a color table. Thus, the *only* way you can express
a yellow color in a DirectColor visual is to specify
its color triple (usually as a 24-bit integer).
Similarly, the only way you can get a color image
is to use a 24-bit image. I suppose this visual is
looked for first because it is the "purest" form of
24-bit color.

The real problem with SGIs is that those guys tend
to use color correctly, but they are the only ones
to do so. If you get brought up on a machine that
is confused, then you tend to think the SGI is at

fault. :-)

Cheers,

David

Subject: Re: 24 bit colors in IDL
Posted by Nigel Wade on Wed, 18 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> Andy Bristow (ajbristow@dera.gov.uk) writes:
>
>> I was just getting my head round 24-bit, having been prompted by this
>> thread. So, at Philip's suggestion I went to
>>
>> http://www.dfanning.com/tips/colors24.html
>>
>> All well and good. Sounds relatively straightforward.
>>
>> So on my SGI (IRIX 6.5.1, IDL 5.1) I tried some of the
>> suggested code (immediatley after starting IDL up):
>>
>> device,decomposed=0
>> tvlct,[[255],[255],[0]],100
>> plot,randomu(10,10),color=100
>>
>> expecting the plot to be yellow, as advertised. Except no, _I_
>> get a medium-light shade of grey!
>>
>> Any suggestions?
>
> Oh, dear. I think this is one of those TrueColor/DirectColor
> tricks that SGI engineers like to play on people.

Why do you want to blame SGI?

If IDL asks for a DirectColor visual and gets what's wrong
with that?

If you look at the idlhelp under "The X Windows Device" you will
find it says:

"How IDL Selects a Visual Class
When opening the display, IDL asks the display for the following
visuals,

in order, until a supported visual class is found:
DirectColor, 24-bit
TrueColor, 24-bit
PseudoColor, 8-bit, then 4-bit
StaticColor, 8-bit, then 4-bit
GrayScale, any depth
StaticGray, any depth"

So, if IDL first asks for DirectColor, and on an SGI it gets it because the display supports it. You shouldn't assume that becuase the default visual someone else gets isn't the same as yours that it is the fault of the system. In this case it is probably because your display doesn't support DirectColor or because it can only handle one visual at a time. An SGI O2 display provides 26 visuals.

If you want to change the default visual which IDL asks for then set idl.gr_visual in your X defaults file.

--

Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523568, Fax : +44 (0)116 2523555

Subject: Re: 24 bit colors in IDL
Posted by Andy Bristow on Wed, 18 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Andy Bristow (ajbristow@dera.gov.uk) writes:
<...>
>> device,decomposed=0
>> tvlct,[[255],[255],[0]],100
>> plot,randomu(10,10),color=100
>>
>> expecting the plot to be yellow, as advertised. Except no, _I_
>> get a medium-light shade of grey!
>>
>> Any suggestions?
>
> Oh, dear. I think this is one of those TrueColor/DirectColor
> tricks that SGI engineers like to play on people. This is
> a complication that I don't *even* want to know about.
>
<...>
>

> If thisName is "DirectColor" change it to "TrueColor":
>
> Device, TrueColor=24

Of course this sould be True_Color=24, but whatever, it worked!

Thanks David!

Andy

--
Andy Bristow

The views expressed above are entirely those of the writer
and do not represent the views, policy, or understanding of
any other person or official body.

Subject: Re: 24 bit colors in IDL
Posted by Nigel Wade on Thu, 19 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
;>
;> Nigel Wade (nmw@ion.le.ac.uk) writes:
;>
;> > Why do you want to blame SGI?
;> >
;> > If IDL asks for a DirectColor visual and gets what's wrong
;> > with that?
;>
;> Geez, Nigel. Bad day? You must be the systems
;> administrator for a bunch of SGIs. :-)

How did you guess? (I could say every day is a bad day
when you administer SGI machines, but I won't).

Sorry, I didn't mean it to sound harsh. Words without
intonation can be hard to control.

;>
;> My apologies. IDL does in fact look for a DirectColor
;> visual first. The problem with a DirectColor visual is
;> that it is not writeable. That is to say, you can't load
;> a color table. Thus, the *only* way you can express
;> a yellow color in a DirectColor visual is to specify
;> its color triple (usually as a 24-bit integer).
;> Similarly, the only way you can get a color image

;> is to use a 24-bit image. I suppose this visual is
;> looked for first because it is the "purest" form of
;> 24-bit color.
;>
;> The real problem with SGIs is that those guys tend
;> to use color correctly, but they are the only ones
;> to do so. If you get brought up on a machine that
;> is confused, then you tend to think the SGI is at
;> fault. :-)
;>
;> Cheers,
;>
;> David

Yeah, isn't 24 bit colour fun?

We've recently gone from 8 bit to 24 bit displays here and all the attendant problems of getting our old 8-bit programs to display in colours other than various shades of red.

--

Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523568, Fax : +44 (0)116 2523555

Subject: Re: 24 bit colors in IDL
Posted by davidf on Thu, 19 Nov 1998 08:00:00 GMT

View Forum Message <> Reply to Message

William Daffer (I suspect) writes:

> The help file says...
>
> The colormap/visual class combination is chosen when IDL first
> connects with the X Window server. Note that if you connect with the X
> server by creating a window or
>
> using the DEVICE keyword to the HELP procedure,
> ^^^
>
> the visual class will be set; it then cannot be changed
> until IDL is restarted. If you wish to use a visual class other than
> the default, be sure to set it with a call to the DEVICE procedure

> before creating windows or otherwise connecting with the X Window
> server.
>
>
> I suspect that the same mechanism is used by
> 'device,get_visual_name=name' as is used in help,/device. In order to
> answer the question, a connection is made to the X-server, the die is
> cast, the matter is settled.
>
> If a connection to the X-server has not been made, I would rather see
> a "don't know" message returned,rather than have IDL decide on the
> visual. At least that way, when some routine started up, I'd know that
> I still have the option of pseudo color or true color available. This
> way, just asking the question decides the answer.

Oh, this is s-o-o-o-o ugly. And news to me too.
This whole business about never being able to set
your visual class after opening a window is extremely
inconvenient.

I would say that if you have no graphics windows
currently open you ought to be able to change your
visual class. After all, PV-Wave can do it. :-) :-)

Cheers,

David

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

Subject: Re: 24 bit colors in IDL
Posted by Vapuser on Thu, 19 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

davidf@dfanning.com (David Fanning) writes:

Dave;

I don't think this will work. On my SGI running Irix 6.5 and idl
5.1.1 just making the 'Device, Get_Visual_Name=thisName' call set's
the visual class. I issued the following statements as the first

statements after starting IDL.

```
IDL> device,get_visual_name=name & print,name
DirectColor
IDL> device,pseudo=8
    ^^^^^^^^^^^^^^^

IDL> device,get_visual_name=name & print,name
DirectColor
^^^^^^^^^^^

IDL> window
IDL> print,!d.n_colors
   16777216

IDL>exit
Process idl finished
```

Similarly ...

```
IDL> device,pseudo=8
IDL> device,get_visual_name=name & print,name
PseudoColor
IDL> device,true=24
IDL> device,get_visual_name=name & print,name
PseudoColor
IDL> window
IDL> print,!d.n_colors
        41
IDL> exit
Process idl finished
```

The help file says...

The colormap/visual class combination is chosen when IDL first
connects with the X Window server. Note that if you connect with the X
server by creating a window or

using the DEVICE keyword to the HELP procedure,
^^^

the visual class will be set; it then cannot be changed
until IDL is restarted. If you wish to use a visual class other than
the default, be sure to set it with a call to the DEVICE procedure

before creating windows or otherwise connecting with the X Window server.

I suspect that the same mechanism is used by 'device,get_visual_name=name' as is used in help,/device. In order to answer the question, a connection is made to the X-server, the die is cast, the matter is settled.

If a connection to the X-server has not been made, I would rather see a "don't know" message returned,rather than have IDL decide on the visual. At least that way, when some routine started up, I'd know that I still have the option of pseudo color or true color available. This way, just asking the question decides the answer.

>
> Andy Bristow (ajbristow@dera.gov.uk) writes:
>
>> I was just getting my head round 24-bit, having been prompted by this
>> thread. So, at Philip's suggestion I went to
>>
>> http://www.dfanning.com/tips/colors24.html
>>
>> All well and good. Sounds relatively straightforward.
>>
>> So on my SGI (IRIX 6.5.1, IDL 5.1) I tried some of the
>> suggested code (immediatley after starting IDL up):
>>
>> device,decomposed=0
>> tvlct,[[255],[255],[0]],100
>> plot,randomu(10,10),color=100
>>
>> expecting the plot to be yellow, as advertised. Except no, _I_
>> get a medium-light shade of grey!
>>
>> Any suggestions?
>
> Oh, dear. I think this is one of those TrueColor/DirectColor
> tricks that SGI engineers like to play on people. This is
> a complication that I don't *even* want to know about.
>
> Try this. Type this:
>
> Device, Get_Visual_Name=thisName
> Print, thisName
>
> If thisName is "TrueColor" change it to "DirectColor":
>

> Device, DirectColor=24
>
> If thisName is "DirectColor" change it to "TrueColor":
>
> Device, TrueColor=24
>
> Do this *before* you open any graphics windows in IDL.
> Try the code above again. Different? The same?
>
> Please let us know. :-(
>
> Cheers,
>
> David
>
> Note: A copy of this article was e-mailed to the original poster.
>

Subject: Re: 24 bit colors in IDL
Posted by davidf on Fri, 20 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

William Daffer (vapuser@catspaw.jpl.nasa.gov) writes:

> On Unix machines, I suspect this is a limitation of X. Don't know
> about Window's machines. Also, is it the same in the
> Windows version? Does calling device,get_visual_name=name set the
> visual class?

There is no such thing as "visual classes" really on
Windows machines. I guess you could say you change
"visual classes" by selecting properties of your
video adapter. This is usually done through a
Control Panel interface. The Get_Visual_Name
keyword returns "class names" that are consistent
with UNIX machines, I think. (I'm going to get
into some trouble with this answer, I can tell
already.)

> Can PV-Wave change the visual class? >

I may have been mistaken about this. I do
remember that if you had no windows open you could
change the number of colors in your IDL session. I'm
not sure about changing visual classes. At the time
I worked with PV-Wave there only was one visual class
for all intents and purposes. :-)

> Yes, cheers. I've just started working in 24bit color, and alot of
> my code is implicitly 8-bit. It would be nice to be able to determine
> whether the visual class has been set yet, without actually setting
> it. Unless I've missed something, it doesn't look like that is
> possible.

Apparently not. Have you brought this to RSI's attention?

Cheers,

David

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

Subject: Re: 24 bit colors in IDL
Posted by Vapuser on Fri, 20 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

davidf@dfanning.com (David Fanning) writes:

(whd snips quote from IDL help file, but leaves in...)
>> (whd)
>> I suspect that the same mechanism is used by
>> 'device,get_visual_name=name' as is used in help,/device. In order to
>> answer the question, a connection is made to the X-server, the die is
>> cast, the matter is settled.
>>
>> If a connection to the X-server has not been made, I would rather see
>> a "don't know" message returned,rather than have IDL decide on the
>> visual. At least that way, when some routine started up, I'd know that
>> I still have the option of pseudo color or true color available. This
>> way, just asking the question decides the answer.
>
> Oh, this is s-o-o-o-o ugly. And news to me too.
> This whole business about never being able to set
> your visual class after opening a window is extremely
> inconvenient.
>

On Unix machines, I suspect this is a limitation of X. Don't know
about Window's machines.

> I would say that if you have no graphics windows
> currently open you ought to be able to change your
> visual class. After all, PV-Wave can do it. :-) :-)
>

Can PV-Wave change the visual class? Also, is it the same in the Windows version? Does calling device,get_visual_name=name set the visual class?

> Cheers,
>
> David
>

Yes, cheers. I've just started working in 24bit color, and alot of my code is implicitly 8-bit. It would be nice to be able to determine whether the visual class has been set yet, without actually setting it. Unless I've missed something, it doesn't look like that is possible.
