Subject: Object boundaries
Posted by Michael Wallace on Tue, 27 Jul 2004 17:56:33 GMT
View Forum Message <> Reply to Message

Is it possible to determine the boundaries of an object in an object graphics view? For example, say that I wanted the coordinates of an IDLgrAxis with it's annotations included. I'd like to get the coordinates which describe the bounding box of the axis and everything else associated with it (title, tick labels, tick marks, etc).

I have some other annotations in this view, and I want to make sure that there is sufficient spacing between these annotations and the annotations automatically created for the axis. I am currently positioning the other annotations at exact locations and most of the time they're fine, but if my axis gets some long tick labels which causes the axis title to get pushed out, the title can collide with the other annotations. I'm just searching for a way to position my other stuff relative to boundary of the axis annotation to ensure that there won't be any overlap. Ideas?

-Mike

Subject: Re: Object boundaries
Posted by David Fanning on Mon, 02 Aug 2004 19:44:10 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> While this works, is there any way to remove the call to xobjview? When
> I remove that call and try the code, I get back 0.0 for all of the tick
> text ranges. Is this because IDL doesn't know where it is before it's
> drawn?

Probably. The text objects really don't know how big they are until their dimensions are computed. I presume this occurs just before they are displayed the first time.

> If so, can I just create a temporary IDLgrBuffer or something,
> and draw to it and grab the numbers from that? If this is the solution,
> that seems pretty ugly to have to create a random IDLgrBuffer in the
> middle of my code, but that's the way RSI works, I suppose. ;-)

Well, you *could* do it this way. I've seen worse. :-)

But I don't think you have to. The IDLexInscribingView object (found somewhere in the IDL distribution) has a GetBounds method. I notice that method only looks at the models in

the graphics hierarchy. I don't know if this is because
the person who wrote that code always scales models and
not graphics primitives, or whether this is a better way to
do it. Have to so some experimenting, I guess.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object boundaries
Posted by Michael Wallace on Mon, 02 Aug 2004 20:19:42 GMT
View Forum Message <> Reply to Message

>> While this works, is there any way to remove the call to xobjview? When
>> I remove that call and try the code, I get back 0.0 for all of the tick
>> text ranges. Is this because IDL doesn't know where it is before it's
>> drawn?
>
>
> Probably. The text objects really don't know how big they are
> until their dimensions are computed. I presume this occurs just
> before they are displayed the first time.

It seems to me that it should be a fairly common problem then... you
want to do some smart positioning of text objects. You have text object
1 which is placed somewhere. It needs to get drawn, and then text
object 2 can be placed smartly (since text object one now has correct
dimension numbers). Now, to see text object 2, you have to draw again.
So, in order to get everything positioned just right, it appears that
you have to go through multiple draw operations.

Maybe there's something preventing this from being done, but why
couldn't the text object at least calculate ahead of time where it falls
on the view plane? Of course, I'd want that calculation to be lazy so
that it doesn't get called every time you apply a transformation or
twiddle some parameter, but there could be a method which would
calculate those numbers given the current setup of the view. In essence
this would be a Draw command, except nothing would really get drawn and
it'd only apply to the text object (and any related object such as an
axis that it's a property of). Then whenever I need to know those
numbers, I could just call that method on the object and then get the
[XYZ]RANGE properties which will now be filled in correctly.

That's enough IDL theory for now. Time to get back to making pretty plots.

-Mike

Subject: Re: Object boundaries
Posted by David Fanning on Mon, 02 Aug 2004 20:32:11 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> It seems to me that it should be a fairly common problem then... you
> want to do some smart positioning of text objects. You have text object
> 1 which is placed somewhere. It needs to get drawn, and then text
> object 2 can be placed smartly (since text object one now has correct
> dimension numbers). Now, to see text object 2, you have to draw again.
> So, in order to get everything positioned just right, it appears that
> you have to go through multiple draw operations.

Well, of course, the alternative is not to let IDL sort
it out, but create the text object yourself with all the
properties you expect it to have. IDL is trying to be *nice*
to you here by creating the tick text object for you if you
haven't bothered. It is one of the few things they *will* do
for you in object graphics, so I think we have to be
grateful. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object boundaries
Posted by Michael Wallace on Mon, 02 Aug 2004 20:46:29 GMT
View Forum Message <> Reply to Message

> Well, of course, the alternative is not to let IDL sort
> it out, but create the text object yourself with all the
> properties you expect it to have. IDL is trying to be *nice*
> to you here by creating the tick text object for you if you
> haven't bothered. It is one of the few things they *will* do

> for you in object graphics, so I think we have to be
> grateful. :-)

But, IDL could be nicer by giving us more control over primitive layout, if we are insane enough (such as myself) to actually want such control. ;-) I am grateful IDL does create tick text objects for us, but that still does not remove my general feeling that IDL could be nicer, much nicer.

As I continue my IDL education, I see more and more that IDL is a *product* rather than a *programming language*.

-Mike

Subject: Re: Object boundaries
Posted by David Fanning on Mon, 02 Aug 2004 21:11:42 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> As I continue my IDL education, I see more and more that IDL is a
> *product* rather than a *programming language*.

I think the original concept was a "programming language for scientists". Which meant, I suppose, a dumbed down language, maybe part "product" and part "language" if you like. But certainly just that part of a language scientists could be expected to know (or learn in a relatively short time).

I'm not sure where IDL is going now. I have heard of people outside of RSI writing programs with iTools, but they have spent the past six months writing their own documentation so they can figure it out. I'm pretty sure no "scientist" is going to program in iTools (or object graphics either, for that matter), so I presume the concept now is "tools for scientists".

I think the jury is still out on whether this will be a successful strategy in the long run, but I'm not too excited about it in the near term. :-(

Cheers,

David

P.S. Have you ever wondered if it's just the Luddites

who hang out here on this newsgroup. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object boundaries
Posted by Rick Towler on Mon, 02 Aug 2004 21:38:57 GMT
View Forum Message <> Reply to Message

Michael Wallace wrote:

>>> While this works, is there any way to remove the call to xobjview?
>>> When I remove that call and try the code, I get back 0.0 for all of
>>> the tick text ranges. Is this because IDL doesn't know where it is
>>> before it's drawn?
>>
>>
>>
>> Probably. The text objects really don't know how big they are
>> until their dimensions are computed. I presume this occurs just
>> before they are displayed the first time.
>
>
> It seems to me that it should be a fairly common problem then... you
> want to do some smart positioning of text objects. You have text object
> 1 which is placed somewhere. It needs to get drawn, and then text
> object 2 can be placed smartly (since text object one now has correct
> dimension numbers). Now, to see text object 2, you have to draw again.
> So, in order to get everything positioned just right, it appears that
> you have to go through multiple draw operations.
>
> Maybe there's something preventing this from being done, but why
> couldn't the text object at least calculate ahead of time where it falls
> on the view plane? Of course, I'd want that calculation to be lazy so
> that it doesn't get called every time you apply a transformation or
> twiddle some parameter, but there could be a method which would
> calculate those numbers given the current setup of the view. In essence
> this would be a Draw command, except nothing would really get drawn and
> it'd only apply to the text object (and any related object such as an
> axis that it's a property of). Then whenever I need to know those
> numbers, I could just call that method on the object and then get the
> [XYZ]RANGE properties which will now be filled in correctly.
>
> That's enough IDL theory for now. Time to get back to making pretty plots.

You're almost there... Don't be afraid to experiment.

Destination devices have a Draw method, but so does every atom. RSI doesn't provide us with the calling sequence but that is easy enough to figure out:

```
IDL> odest=obj_new('idlgrwindow')
IDL> oview=obj_new('idlgrview')
IDL> oaxis=obj_new('idlgraxis')
IDL> oaxis->getproperty, ticktext=ott
IDL> ott->getproperty, xrange=xr, yrange=yr, zrange=zr
IDL> print, xr,yr,zr
      0.00000000      0.00000000
      0.00000000      0.00000000
      0.00000000      0.00000000

;  The reason you are reading this...
IDL> oaxis->draw, odest, oview

IDL> ott->getproperty, xrange=xr, yrange=yr, zrange=zr
IDL> print, xr,yr,zr
     -0.033318131       1.0333181
     -0.076304187    -0.018814731
      0.00000000      0.00000000
```

So you can do exactly what you want to do. And don't worry, calling an atom's draw method will not actually cause it to be drawn in the destination device.

-Rick

Subject: Re: Object boundaries
Posted by Paul Van Delst[1] on Mon, 02 Aug 2004 21:54:21 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Michael Wallace writes:
>
>
>> As I continue my IDL education, I see more and more that IDL is a
>> *product* rather than a *programming language*.
>
>
> I think the original concept was a "programming language

> for scientists". Which meant, I suppose, a dumbed down
> language, maybe part "product" and part "language" if you
> like. But certainly just that part of a language scientists
> could be expected to know (or learn in a relatively short
> time).
>
> I'm not sure where IDL is going now. I have heard of
> people outside of RSI writing programs with iTools,
> but they have spent the past six months writing their
> own documentation so they can figure it out. I'm pretty
> sure no "scientist" is going to program in iTools (or
> object graphics either, for that matter), so
> I presume the concept now is "tools for scientists".

Hmm. I think "toolbox for scientists' graduate students" is more accurate. ;o) All the bits and pieces are there.. which you can hand off to your grad student (or freshly minted Masters grad now working for a contractor :o) and ask him/her to make pretty pictures for the next budget review/conference meeting/paper. For people that aren't into programming, the IDL object stuff is too complicated, or too slow/unresponsive (via the various iTools).

I am quite boggled by the fact that people are still having to futz with tick mark objects (or whatever) in creating plots.

> I think the jury is still out on whether this will
> be a successful strategy in the long run, but I'm not
> too excited about it in the near term. :-(

I'll second that.

But, just in case people think I'm being overly crotchety, I still haven't found a peer to the simplicity of reading in and direct-graphics PLOT-ing data in IDL. At least in Linux. Dunno about Windows (although there is this company in Golden, CO that makes a *sweet* suite of windows-based graphics tools, Grapher/Surfer, etc.).

paulv

> P.S. Have you ever wondered if it's just the Luddites
> who hang out here on this newsgroup. :-)

We're the only ones who need the help as we're dragged kicking and screaming into the world of objects, no? :o)

Subject: Re: Object boundaries
Posted by David Fanning on Mon, 02 Aug 2004 21:58:09 GMT
View Forum Message <> Reply to Message

Rick Towler writes:

> ; The reason you are reading this...
> IDL> oaxis->draw, odest, oview

How did you figure this out, Rick? I tried the DRAW
method, too, but could only vaguely remember how to
find parameters to unknown procedures and functions.
And I didn't have time to figure it out again. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Object boundaries
Posted by Karl Schultz on Mon, 02 Aug 2004 22:44:04 GMT
View Forum Message <> Reply to Message

Look into the GetTextDimensions method in the Window object. It will tell
you the size of the text string before it is rendered.

Karl

"Michael Wallace" <mwallace.no.spam@no.spam.swri.edu.invalid> wrote in
message news:10gt8evodbiac7e@corp.supernews.com...
>>> While this works, is there any way to remove the call to xobjview? When
>>> I remove that call and try the code, I get back 0.0 for all of the tick
>>> text ranges. Is this because IDL doesn't know where it is before it's
>>> drawn?
>>
>>
>> Probably. The text objects really don't know how big they are
>> until their dimensions are computed. I presume this occurs just
>> before they are displayed the first time.
>
> It seems to me that it should be a fairly common problem then... you
> want to do some smart positioning of text objects. You have text object
> 1 which is placed somewhere. It needs to get drawn, and then text
> object 2 can be placed smartly (since text object one now has correct
> dimension numbers). Now, to see text object 2, you have to draw again.
>   So, in order to get everything positioned just right, it appears that
> you have to go through multiple draw operations.
>
> Maybe there's something preventing this from being done, but why

> couldn't the text object at least calculate ahead of time where it falls
> on the view plane? Of course, I'd want that calculation to be lazy so
> that it doesn't get called every time you apply a transformation or
> twiddle some parameter, but there could be a method which would
> calculate those numbers given the current setup of the view. In essence
> this would be a Draw command, except nothing would really get drawn and
> it'd only apply to the text object (and any related object such as an
> axis that it's a property of). Then whenever I need to know those
> numbers, I could just call that method on the object and then get the
> [XYZ]RANGE properties which will now be filled in correctly.
>
> That's enough IDL theory for now. Time to get back to making pretty
plots.
>
> -Mike

Subject: Re: Object boundaries
Posted by Rick Towler on Mon, 02 Aug 2004 23:05:01 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

> Rick Towler writes:
>
>
>> ; The reason you are reading this...
>> IDL> oaxis->draw, odest, oview
>
>
> How did you figure this out, Rick? I tried the DRAW
> method, too, but could only vaguely remember how to
> find parameters to unknown procedures and functions.
> And I didn't have time to figure it out again. :-)

I wish I could say I divine the parameters, like the great Carnac
foretold the question to the answer written on the envelope, by simply
lifting the keyboard to my forehead. Alas, my methods are far less exotic.

IDLgrGraphic::Draw is pretty easy if you think about it. Passing one,
then two dummy arguments led IDL to tell me it took two parameters.
Thinking about it for a second, I knew that the view was probably one,
and the destination the other. Trial and error got the order.

Obviously this approach has its limitations but RSI doesn't make their
API overly obtuse (well, IMO) so you can do this for quite a number of
the undocumented parent objects/methods. For example,
IDLgrModel::GetXYZRange. The name says function method. No arguments

fails. One fails. Two fails. Three works. O.K. Then the arguments
are (x,y,z).

-Rick

Subject: Re: Object boundaries
Posted by James Kuyper on Tue, 03 Aug 2004 14:46:07 GMT
View Forum Message <> Reply to Message

Paul Van Delst wrote:
...
> But, just in case people think I'm being overly crotchety, I still
> haven't found a peer to the simplicity of reading in and
> direct-graphics PLOT-ing data in IDL. At least in Linux. Dunno about

As far as simplicity of reading in and plotting data, it's hard to beat
gnuplot. It's available on the IRIX systems I have access to, but not on
the Linux ones, which seems odd for a program with "gnu" in it's name.
I'm sure there's a fascinating history behind that fact.

I prefer IDL for it's greater power; but when I'm in a hurry I use
gnuplot, if it's feasible to do so.

Subject: Re: Object boundaries
Posted by mperrin+news on Wed, 04 Aug 2004 21:41:34 GMT
View Forum Message <> Reply to Message

James Kuyper <kuyper@saicmodis.com> wrote:
> Paul Van Delst wrote:
> ...
>> But, just in case people think I'm being overly crotchety, I still
>> haven't found a peer to the simplicity of reading in and
>> direct-graphics PLOT-ing data in IDL. At least in Linux. Dunno about
>
> As far as simplicity of reading in and plotting data, it's hard to beat
> gnuplot. It's available on the IRIX systems I have access to, but not on
> the Linux ones, which seems odd for a program with "gnu" in it's name.
> I'm sure there's a fascinating history behind that fact.

That history reflects more on your particular systems administrators than on
any more global phenomena. :-) Gnuplot is readily available for Linux, and is
included out-of-the-box in a number of distributions...

 - Marshall
