
Subject: Re: List of Points --> ROI via IDLVECTOR data type for iimage itool
Posted by [David Alexander](#) on Thu, 13 Oct 2005 20:15:29 GMT

[View Forum Message](#) <> [Reply to Message](#)

Edward,

You want to display the pixels from the image after you run your sampling routine, right?

After your sampling routine, do you end up with 3 vectors (x, y, pixels)? If so you could either pass them directly into iimage (iimage can take 3 vectors as input, but it will force you to grid the data), or you can generate an image array with an alpha channel (RGBA) based on your pixel data, and set the Alpha channel value for all the unused pixels to 0. So, something like this:

```
pro test
  imagePixels = randomu(s,1000)
  x = bytscl(randomu(s,1000))
  y = bytscl(randomu(s,1000))

  newImage=fltarr(max(x)+1,max(y)+1)
  newAlpha=fltarr(max(x)+1,max(y)+1)
  newImage[x,y]=imagePixels
  newAlpha[x,y]=255
  newImage=[[[newImage]],[[newImage]],[[newImage]],[[newAlpha] ]]

  iimage,newImage
end
```

My sample data is not RGB, so I triplicated the newImage variable. If you already have RGB pixels, you won't need to do that.

Dave

Subject: Re: List of Points --> ROI via IDLVECTOR data type for iimage itool
Posted by [MarioIncandenza](#) on Thu, 13 Oct 2005 22:08:55 GMT

[View Forum Message](#) <> [Reply to Message](#)

Dave,

That's an interesting trick, but not what I'm after right now. I don't need to transform the sampled points into an image, I simply need to display them on top of the image so that I can see where they fall. So my problem is more about trying to pass these XY points into an image handler (iimage, IMap, or really I could use ENVI too). This should be easy, but for the life of me I can't figure it out.

--Edward H.

Subject: Re: List of Points --> ROI via IDLVECTOR data type for iimage itool
Posted by [David Alexander](#) on Fri, 14 Oct 2005 17:40:18 GMT
[View Forum Message](#) <> [Reply to Message](#)

Edward,

OK, I see what you're trying to do.

There's no point visualization that I know of in iTools. However, you could create a second image containing only the points, and overlay it on top of the original. This is based on the same idea as my last example. I'm setting the points to red in the overlay image, and making all other pixels transparent.

```
pro test
  x = bytscl(randomu(s,1000))
  y = bytscl(randomu(s,1000))

  newImage=bytarr(256,256)
  newImage[x,y]=255

  newImage=[[[[newImage]],[[bytarr(256,256)]],[[bytarr(256,256) ]],[[newImage]]]]

  iimage,dist(256)
  iimage,newImage,/OVERPLOT
end
```

If you don't want to mark individual pixels, but would rather put some kind of symbol annotation in the location of each point, you could overlay a 2D plot with symbols on the image, then turn off the plot line. Something like this:

```
pro test
  x = bytscl(randomu(s,100))
  y = bytscl(randomu(s,100))

  idx=SORT(x)
  x=x[idx]
  y=y[idx]

  iimage,dist(256)
  iplot,x,y,/OVERPLOT,LINESTYLE=6,SYM_INDEX=2
```

```
;Remove the axes  
void=itGetCurrent(TOOL=oTool)  
id=oTool->FindIdentifiers('*plot',/VISUALIZATIONS)  
oPlot=oTool->GetByIdentifier(id)  
oPlot->SetAxesStyleRequest,0
```

```
;Unselect the plot to force the axes to disappear  
oPlot->Select,0  
end
```

iPlot will always add axes, so the last few lines of my example show how to turn those off. iplot also has many other keywords to set the appearance (color, size, type, etc.) of the symbols.

Dave
