
Subject: Re: Creating a new image from an image input in IDL
Posted by [bcubeb3](#) on Wed, 05 May 2010 13:37:22 GMT
[View Forum Message](#) <> [Reply to Message](#)

On May 5, 8:01 am, Jeremy Bailin <astroco...@gmail.com> wrote:
> On May 5, 2:45 am, bcubeb3 <barry.brian.barr...@gmail.com> wrote:
>
>
>
>> After typing this line of code:
>> IMAGE=READ_TIFF(FILEPATH('/bin/butterfly.tiff'))
>> help, IMAGE
>
>> I get the output
>> IMAGE BYTE = Array[3, 4800, 6000]
>
>> Now I want to write a computer program to systematically loop through
>> each of the $n \times n$ pixels of the image and to use a coordinate system
>> in pixel units to compute new coordinates based on the formula θ_s
>> = $\theta - (\text{size parameter of your choosing in units of}$
>> $\text{pixels}) * \theta_{\text{hat}}$.
>
>> The vector θ_s tells me where to look in the original image to
>> extract intensity information which will then store in my image array.
>> I will use a bilinear scheme when assigning new intensity values that
>> will be stored for my newly created image array θ . Now I have no
>> idea how to even begin. I was looking for stuff online and I was
>> looking at help manuals but all efforts proved futile. Let me know of
>> your suggestions and I greatly appreciate your help on this.
>
>> -Barry
>
> So I think what you're saying (please correct me if I've
> misunderstood!) is that you have a simple transformation between the
> pixel coordinates of the new image and the pixel coordinates of the
> old image. So there's two steps here:
>
> 1) Calculate the transformation for each of the pixel locations of the
> new image
> 2) Copy the image values over
>
> For 1) I would do something like this:
>
> ; size of image:
> imagesize = size(image,/dimen)
> nchan=imagesize[0]
> nx=imagesize[1]
> ny=imagesize[2]

```

> npix = nx*ny
>
> ; get a 2D list of all pixel coordinates. This is going to take a LOT
> of memory
> ; for a 4800x6000 image!
> newcoords_2d = array_indices([nx,ny], lindgen(npix), /dimen)
>
> ; apply the transformation. I don't really understand your formula,
> ; but hopefully this example will help you.
> ; reform is needed to flatten it instead of being 1xNPIX
> ; X coordinate in old image, as a linear combination of X and Y
> ; coordinates of new image (oldX = A*newX + B*newY)
> oldcoordsX = reform( A * newcoords[0,*] + B * newcoords[1,*], npix)
> ; Y coordinates in new image, as a linear combination of X and Y
> ; coordinates of new image (oldY = C*newX + D*newY)
> oldcoordsY = reform( C * newcoords[0,*] + D * newcoords[1,*], npix)
>
> Once you've done that, copying it over is easy (though again, very
> memory-intensive):
>
> newimage = bytarr(imagesize)
> for chan=0,2 do newimage[chan,*,*] = oldimage[replicate(chan,npix),
> oldcoordsX, oldcoordsY]
>
> (NOTE: COMPLETELY UNTESTED)
>
> -Jeremy.

```

THIS IS RIGHT you understood my problem. I ran your code and it works.

The only question I have is how would you use a bilinear interpolation then. It is important that I take that into account for the after picture after the transformation. I am reading in wikipedia what it is. I am assuming bilinear interpolation is needed since the transformation move pixels to fractional coordinates ??? But in any case how would that be done. I can tweak the transformation, but I think you gave me a great start. I am new so it took a while to understand what your code did. I am still not clear what this line does:

```
newcoords = array_indices([nx,ny], lindgen(npix), /dimen)
```

It initializes an array to lindgen(npix) of the same size of the array of original pic???

Also for the transformation,

i.e.,

```
oldcoordsX = reform( A * newcoords[0,*] + B * newcoords[1,*], npix)
```

is oldcoordsX a known or is newcoords a known. From this fragment of code, it leads me to believe that newcoords is given and oldcoordsX is being solved by the transformation which to me means that oldcoords is really the new coordinates after the transformation. Not entirely sure if I am understanding your code, but it ran and I saw a before and after picture.

best,
Barry
