3D Turtle Assisted Toolbox (ver 3.01)

Yuji Suda ysuda@smccake.com

We live in a 3D world. But we can not fly and play freely in space. So in order to play in 3D space, I wrote a 3D turtle system in J language (version J602a).

What can we do with the system?

You can move around in a 3D coordinate system with full freedom!

And once you have 3D voxel world with a property, you can creat a number of tools for themes in life.

In the following documents, you will learn the basic movements of a 3D turtle and some tools I have figured out for my themes.

After you reviewed and confirmed the examples, you may creat tools of your own.
Please enjoy and utilize this wonderful mechanism for the rest of your life!

Please check the link of screenshot of my experiments at
http://www.smccake.com/j602/index.htmol

<<< 3D turtle basics >>>

Start J602 and File->Open c:\3d-turtle-assisted-toolbox\3d_turtle_assisted_toolbox_300.ijs and Run->Window.

Now you are ready to go. Imaginary 3D turtles can travel anywhere in a 3D world with the following commands. long names and short names in parenthesis

```
make_new_turtle (mt)
setpos (sp)
forward (fd)
backward (bk)
right (rt)
left (lt)
up_pitch (up)
down_pitch (dp)
right_roll (rr)
left_roll (lr)
towards (td)
```

Let's play in a 3D world ON a 3D turtle!

Firstly, we have to creat a 3D turtle to ride on with 'make_new_turtle' command

```
mama =. make_new_turtle 0
```

To check the property of mama, type the turtle name and hit enter key.

```
        mama
0 0 0 1
1 0 0 1
0 1 0 1
0 0 1 1
```

J interpreter returns the properties of mama. The first row contains x y z values in 3 axes coordinate system. The last item of the row 1 is used for another property in the future. So you can just ignore the 4th item for the time being. The following three rows represent heading information of the turtle. These values are changed according to the motion of the turtle.

Let's travel in 3D world on the turtle.

```
        mama =. mama forward 1
```

To see the result, type mama.

```
        mama
1 0 0 1
1 0 0 1
0 1 0 1
0 0 1 1
```

Now the first row of the turtle information has changed from 0 0 0 1 to 1 0 0 1. x y z coordinate shows new position of 1 0 0.

Try once again the same command

```
        mama =. mama forward 1
        mama
2 0 0 1
1 0 0 1
0 1 0 1
0 0 1 1
```

Now x value is 2.

Then let's turn to right 90 degrees and move forward 1.

```
        mama =. mama right 90
        mama =. mama forward 1
        mama
          2          _1 0 1
6.12303e_17          _1 0 1
          1 6.12303e_17 0 1
          0           0 1 1
```

Now you see, x y z coordinate has been changed from 2 0 0 to 2 _1 0 and the heading values has been changed! The turtle's heading has changed to minus direction of y-axis. So the result of forward 1 made its position 2 _1 0.

OK. Then let's warp to a new position with the same heading.

```
      mama =. mama setpos 10 10 0
      mama
         10           10 0 1
6.12303e_17           _1 0 1
          1  6.12303e_17 0 1
          0            0 1 1
```

You see the x y z value are now 10 10 0 with the same heading, so

```
      mama =. mama forward 1
      mama
         10            9 0 1
6.12303e_17           _1 0 1
          1  6.12303e_17 0 1
          0            0 1 1
```

Now the x y z value is 10 9 0.

You can move back and foth, change yaw with right or left, change pitch with up_or down_pitch, and roll right or left. You can also change heading toward a specific coordinate with towards command. Usages are as follows.

```
      foo_turtle =. make_new_turtle 0
      foo_turtle =. foo_turtle setpos x_y_z_vector
      foo_turtle =. foo_turtle forrward a_distance
      foo_turtle =. foo_turtle backward a_distance
      foo_turtle =. foo_turtle right an_angle_in_degree
      foo_turtle =. foo_turtle left an_angle_in_degree
      foo_turtle =. foo_turtle up_pitch an_angle_in_degree
      foo_turtle =. foo_turtle down_pitch an_angle_in_degree
      foo_turtle =. foo_turtle right_roll an_angle_in_degree
      foo_turtle =. foo_turtle left_roll an_angle_in_degree
      foo_turtle =. foo_turtle towards target_x_y_z_vector
```

With combination of movement commands you can travel anywhere in a 3D world!

<<< 3D turtle graphics >>>

So far, we travel in a 3D coordinate world. And you might have noticed that there has been no graphics, turtle graphics.

Now it's time to prepare a concrete 3D world and a pen functionality.

Let's make a 3D voxel world, for example, 200 x 200 x 200.

Let's make this voxel 3D world in ppm full color file. Please confirm the definition of ppm file in http://www.smccake.com/j602/ppm_man_page.pdf

It is defined as a full color image file that consists of a header and a series of image RGB byte sets.

A 200x200 pixel ppm full color file header is defined as a series of character bytes as follows:

'P','6',LF,'2','0','0',LF,'2','0','0',LF,'2','5','5',LF

where LF is line feed control code, which is 10 in decimal 0A in hexadecimal.

We can define this ppm header as follows.

```
        PPMHEADER =: 'P6',LF,'200',LF,'200',LF,'255',LF
```

Let's check the definition

```
        PPMHEADER
P6
200
200
255
```

OK. Then let's add pixel data set. Since a plane consists of 200x200 pixels. One pixel needs RGB three bytes, so a total of 120000 pixel bytes are needed.

White color in full color scale is 255, 255,255. Black is 0,0,0. So the definition of a black canvas is 200 x 200 x 3 characters of the 1st character from ASCII character set.

BLACK_PIXEL_RGB =: (0 { a.),(0 { a.),(0 { a.) NB. take the first character from ASCII set as a black pixel with use of from primitive '{'.
BLACK_PLANE_RGB =: 40000 3 $ BLACK_PIXEL NB. Define 200x200 pixel plane, i.e. 40000 RGB pixels, with shape primitive '$'.
BLACK_PLANE_DATA_RGB =: , BLACK_PLANE_RGB NB. make the plane pixels in series with lavel primitive ','.
BLACK_PLANE_PPM =: PPMHEADER , BLACK_PLANE_DATA_RGB NB. And use append primitive ',' to combine header and data to form a ppm file.

Let's check how it appears. Save the image data to a test file test.ppm with use of write_str_to_file primitive, as follows,

```
        BLACK_PLANE_PPM write_str_to_file 'test.ppm'  NB. save the file to test.ppm
```

Now we have created test ppm black plane canvas. Let's view this image file with ImageJ. ImageJ is a freeware for versatile image processing and analysis. Please download it at http://rsbweb.nih.gov/ij/download.html
Run ImageJ and open the ppm file we have created. You will see black 200x200 RGB image.

Here we need a 3D canvas. So let's make a 200x200x200 3D canvas. How? By piling 200 sheets of a 200x200 size plane! When 200 planes are stacked, we have 200x200x200 full color voxel canvas. Z axis is assigned to file number as in ppm000.ppm, ppm001.ppm, ppm002.ppm ... ppm199.ppm.

Let's make 200 series BLACK_PLANE_ppm files with use of make_ppm_stack command,

```
        BLACK_PLANE_PPM make_ppm_stack 200
```
NB. BLACK_PLANE_PPM ppm file 000..199 are created with this function.

Now we get 200 ppm files in ppm directory and we are ready to play 3d graphics!

Oh! We need a pen, too. Here it is! long names and short names in parenthesis

```
        paint_here_color (phc)
        forward_pen_down_color (fdpdc)
```

Usages are

```
        foo_turtle paint_here_color a_RGB_vector
        foo_turtle forward_pen_down_color a_distance
```

RGB color code should be set at COLOR_RGB, default is 255 255 255.

That is all we need to do graphics. Every painting tool is derived from these two primitives!

At this point, we have a black 3D voxel world with 200x200x200 full color elements and a pen mechanism with default white color.

Now it's time to think and try!

Please check the link of screenshot of my experiments and related ijs source files at http://www.smccake.com/j602/index.htmol My experimetns will continue as I face theme in my life. You may see how I live with this tool and I hope you will find how it is good for you to solve your themes.

<< Acknowledgement >>

I thank Mr. Toshihiro Anzai for the public domain software (TURBO PASCAL source fils) on 3D turtle graphics. My J scripts of 3D turtle movements except the primitive 'towards' totaly depended on the program by him published in 1986. Please see the file at http://www.smccake.com/j602/JUG023.zip