Rectilinear Motion

Homework1

Miguel Tlapa Juárez

31/01/2014

This document describes the system architecture and design about the body controller module, it's have block diagram and flowchart to describe software and hardware architecture.

Revision History

Date	Revision Number	Author/Editor	Modifications
January 2014	0.1	Miguel Tlapa	Created file

Disclaimers

1. Explanation

1.1 Code Main.cpp

I am going to try to explain each of the program covered in class.

a) Define Manipulation Variables Like:
   - Delay = the time that each object in the screen is drawn.
   - Width and height = define the window
   - r,g,b = define the color of the window
   - Angle = rotation angle trough axis x

b) Define 3 objects like:

   Box, Line, Cylinder.

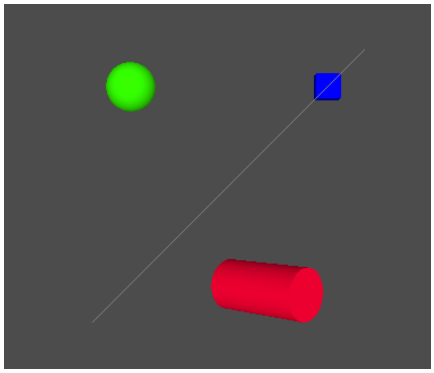
   Each object has color and positions in x,y,z.

   The first define the start and the last the end of the figure.

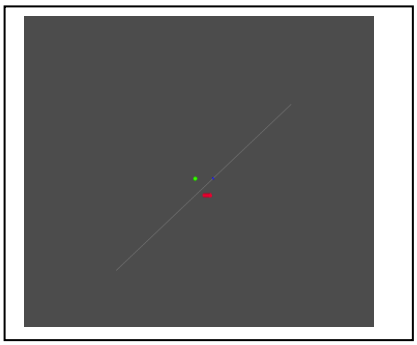
c) Modify the position of Primitive in the axis z.
   if the value of z is in [-15,-4.0] then
   incZ *= -1.0

I obtain the next values. If I change the value of angle, the cylinder rotate 45 grades.

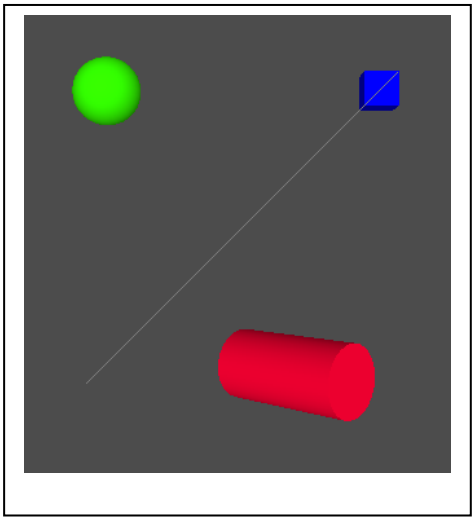
```
Valor de incZ -0.1
Valor de z -10.2
Valor de incZ -0.1
Valor de z -10.3
…
Valor de z -14.9
Valor de incZ -0.1
Valor de z -15
Valor de incZ 0.1
Valor de z -14.9
Valor de incZ 0.1
Valor de z -14.8
Valor de incZ 0.1
Valor de z -14.7
…
Valor de incZ 0.1
Valor de z -4.1
Valor de incZ 0.1
Valor de z -4

Valor de incZ -0.1
Valor de z -4.1
```

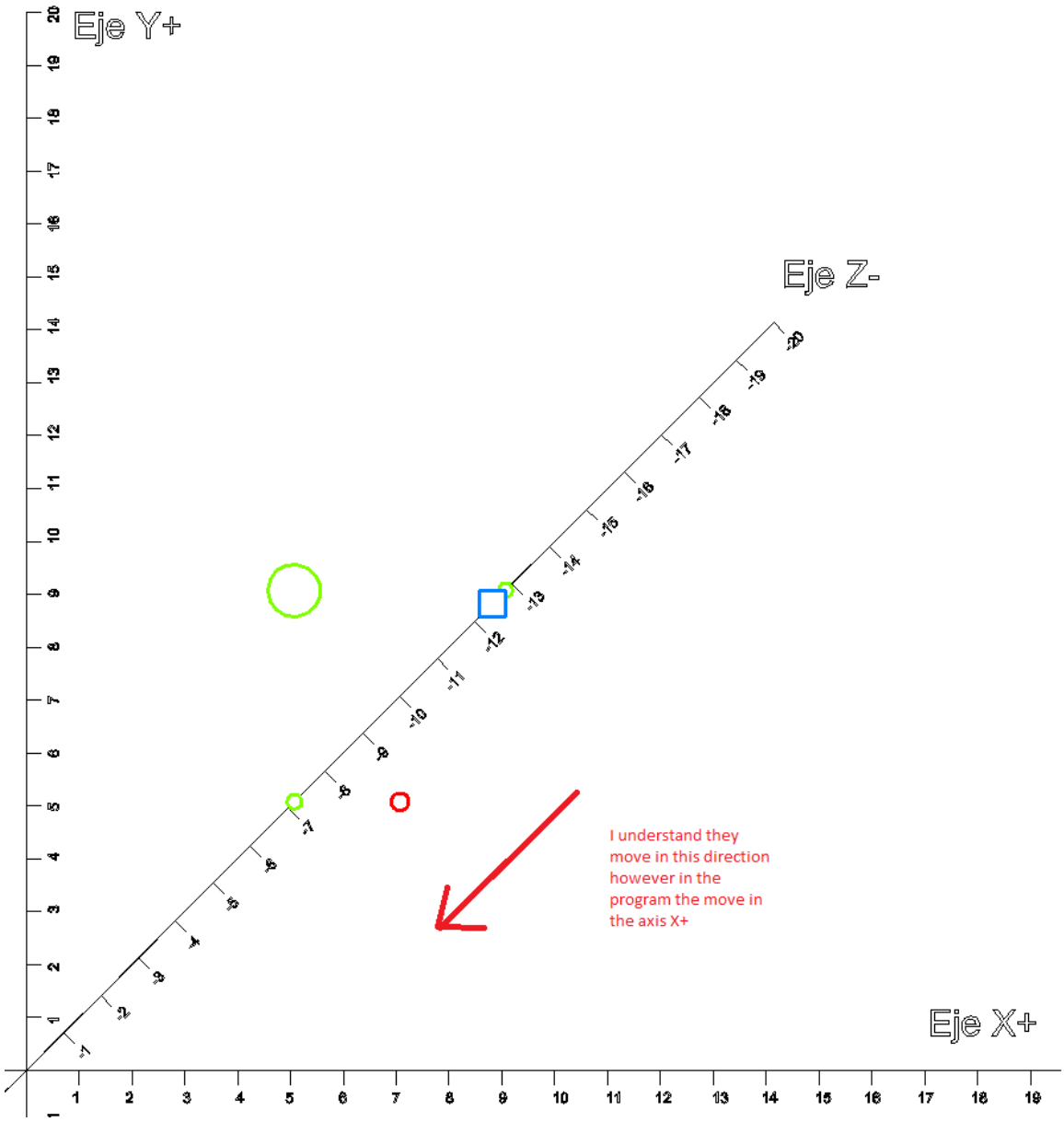


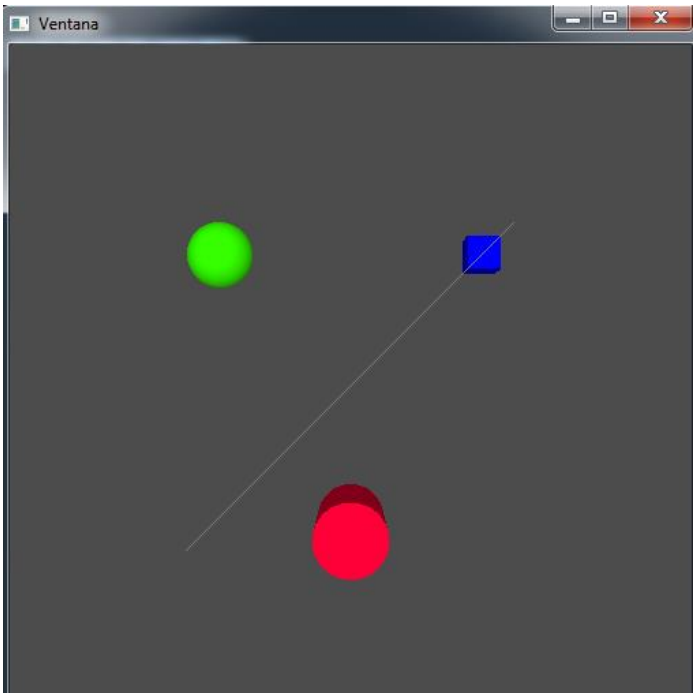
If I change the value of z =-100 I can see the objects very far.



If I change the value of delay =100 I can see the objects move slowly.

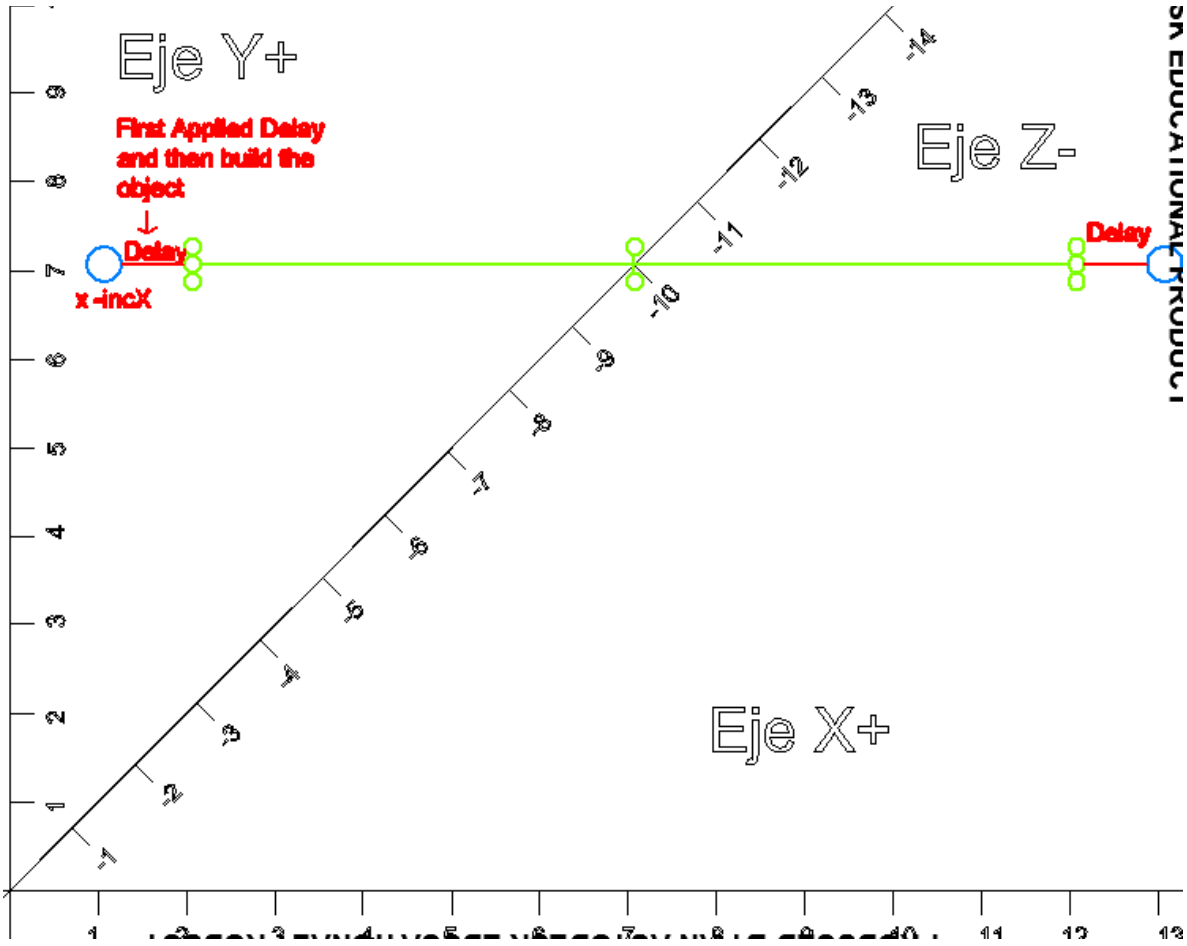



Eje Y+
Eje Z-
Eje X+
I understand they move in this direction however in the program the move in the axis X+


Ventana

Exercise02.cpp

First I draw the lines and elements in the rectangular plane xyz.



I obtained the next values for each iteration.

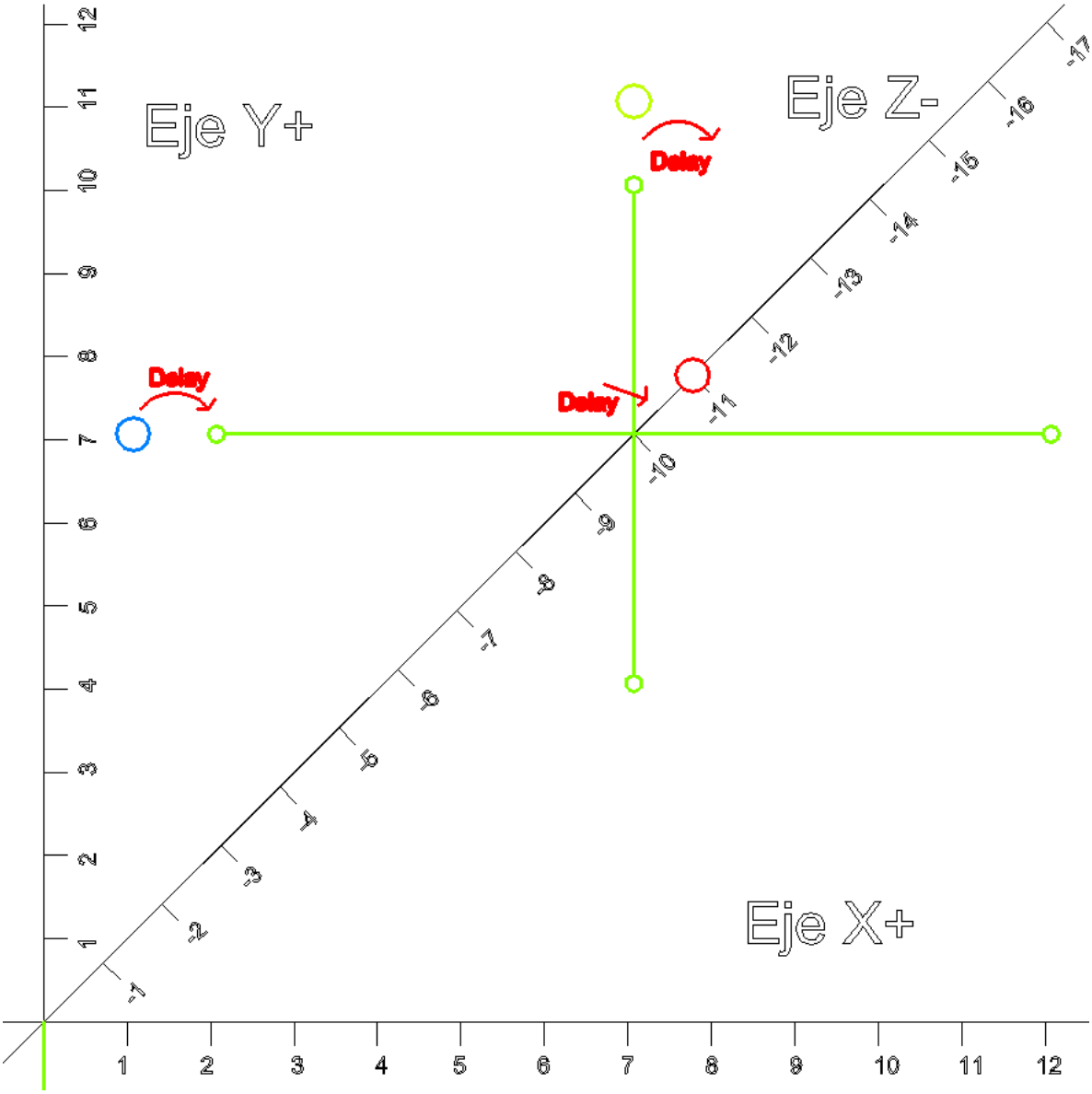
```
value x-4
value x-3
value x-2
value x-1
value x0
value x1
value x2
value x3
value x4
value x5
value x6
value x6
```

Draw each element each time of delay:

Exercise03.cpp

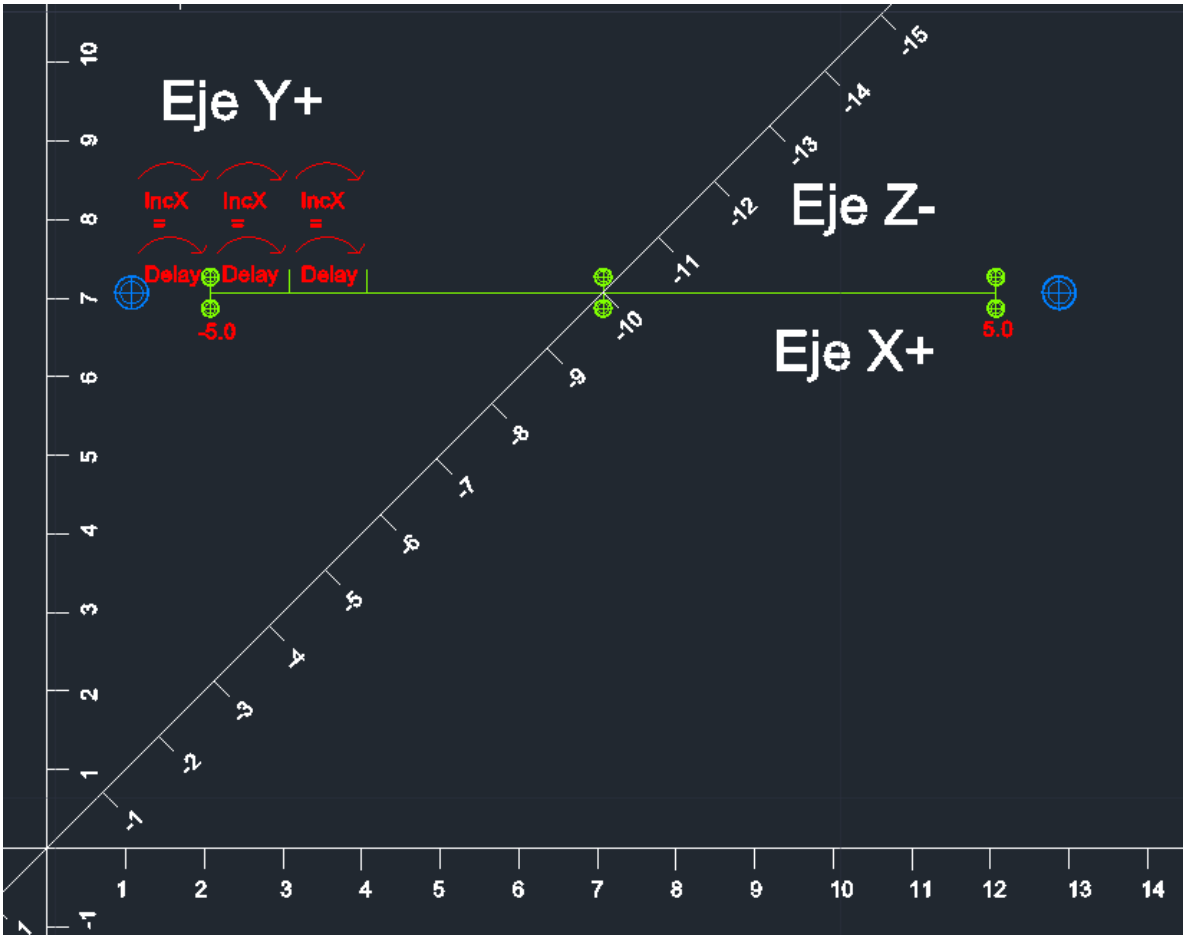
We can see 3 objects like move in x,y and z.

Applied the same principle how the first exercise.



Exercise04.cpp

In this exercise learn the principles to move a particle in one axis with velocity constant.



inc x= Spacial Physics Increment

delay = Drawing Time

Data:

xi = -5.0 = Initial Position

xf = 5.0 = Final Position

x = xi = Change of variable

tx = 10 [seg]

delay = 1000 ms = 1s

Equations

First we need to calculate the distance between 2 points

xi and xf

d = xf - xi = 5.0 - (-5.0) = 10 units

v = d/t = 10units/10seg = 1/seg

Increment x is definede trough velocity of movement.

inc x = d/t = 10units/10seg = 1/seg

Rule of tree

distance ---> time

incx ----> [tseg]

[tseg] = Time of Refresh that use GPU to draw objects.

[tseg] = incx * time/ distance However we need to multiply x 1000 because the time should be in milisecods

[tseg] = incx * time/ distance * 1000

With this relation we can adjust the Time of Refresh with Increment X.

But always we need to define delay.

```
value of x =-3
value of x =-2
value of x =-1
```

```
value of x =0
value of x =1
value of x =2
value of x =3
value of x =4
value of x =5
value of x =6
value of x =6
value of x =6
value of x =6
value of x =6
```

Exercise05.cpp

inc x= Spacial Physics Increment

delay = Drawing Time

Data:

tx = 10 [seg]

xi = -5.0 = Initial Position

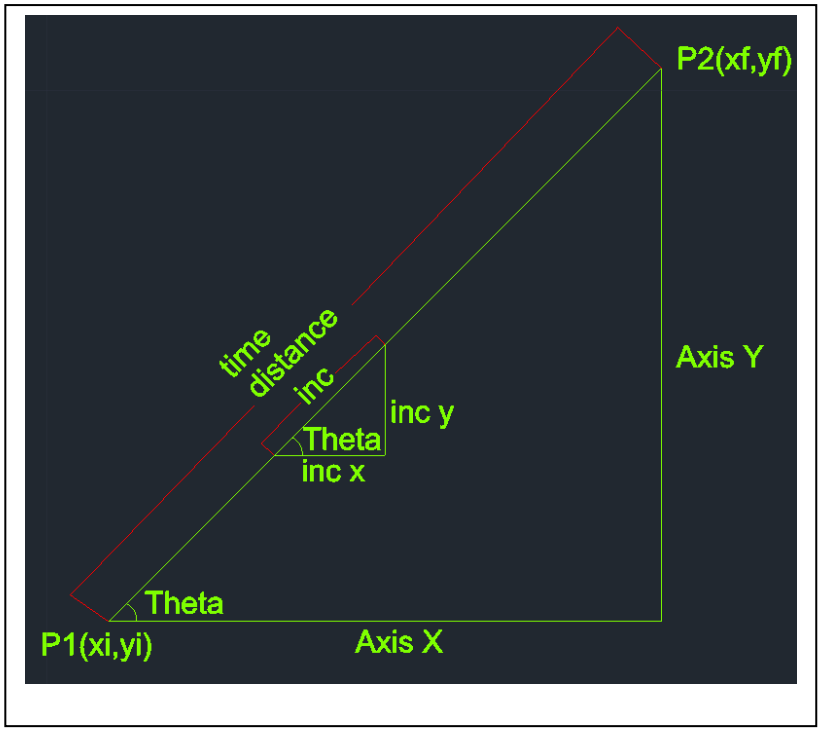
xf = 5.0 = Final Position

x = xi = Change of variable

yi = -5.0 = Initial Position

yf = 5.0 = Final Position

y = yi = Change of variable



Equations

First we need to calculate the distance between 2 points

P2 and P1

d = (sqrt (pow(xf - xi, 2) + pow(yf - yi, 2)))

Note pow(xf-xi) = pow(xi-xf)

Increment x is definede trough velocity of movement.

inc = small increment whith respect theta angle in the plane xy

inc = d/t = 10units/10seg = 1/seg

Theta is defined

Theta = atan ((yf-yi)/(xf-xi))

incx = inc * cos theta

incx = inc * sen theta

Rule of tree

distance ---> time

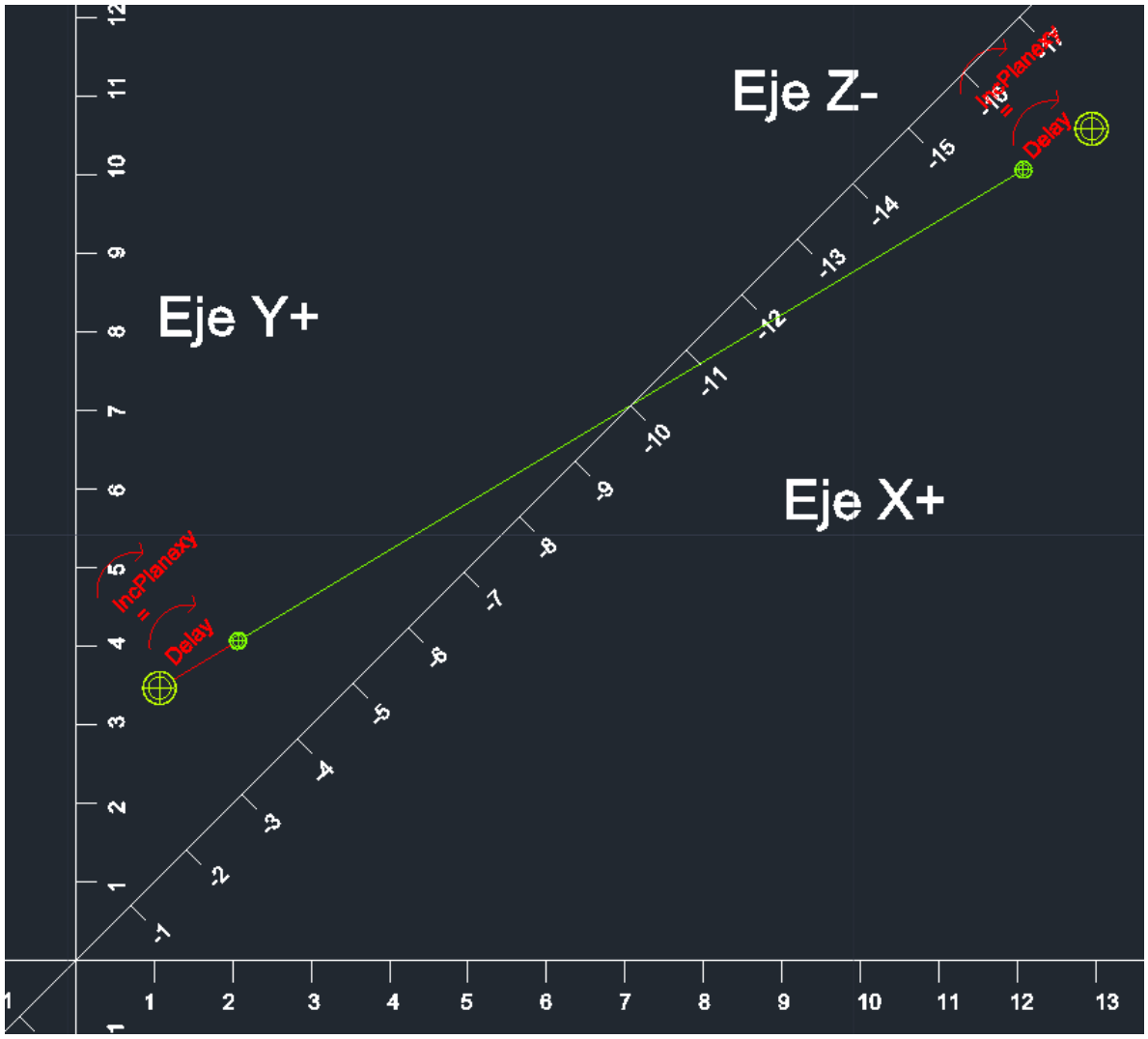
incx ----> [tseg]

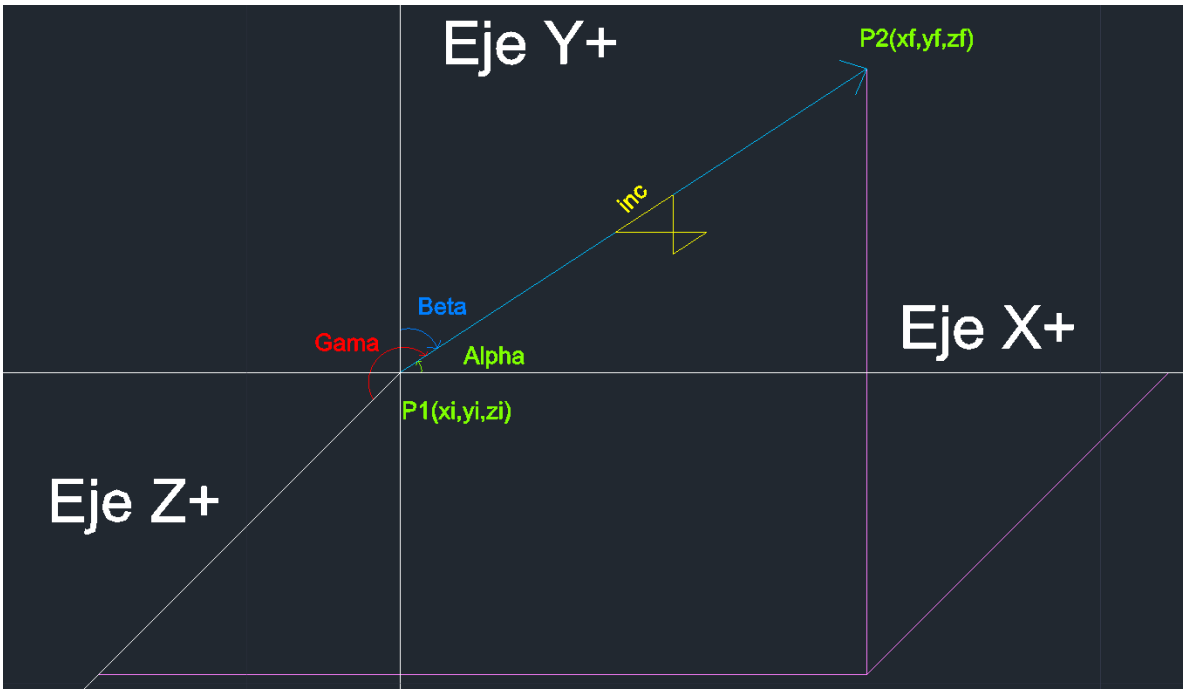
[tseg] = Time of Refresh that use GPU to draw objects.

[tseg] = incx * time/ distance However we need to multiply x 1000 because the time should be in milisecods

[tseg] = incx * time/ distance * 1000

With this relation we can adjust the Time of Refresh with Increment X.


Eje Z-
Eje Y+
Eje X+
IncPlanexy
=
Delay
IncPlanexy
=
Delay



inc x= Spacial Physics Increment

delay = Drawing Time

Data:

tx = 10 [seg]

xi = -5.0 = Initial Position

xf = 5.0 = Final Position

x = xi = Change of variable

//

yi = -5.0 = Initial Position

yf = 5.0 = Final Position

y = yi = Change of variable

//////////////////////////////////////

zi = -5.0 = Initial Position

zf = -10.0 = Final Position

z = zi = Change of variable

Equations

First we need to calculate the distance between 2 points

P2 and P1

d = (sqrt (pow(xf - xi, 2) + pow(yf - yi, 2) + pow(zf - zi, 2)))

Note pow(xf-xi) = pow(xi-xf)

Increment x is definede trough velocity of movement.

inc = small increment whith respect theta angle in the plane xy

inc = d/t = 10units/10seg = 1/seg

Using Direction Cosines

Cos alpha = x/d = xf-xi/d

Cos alpha = y/d = yf-yi/d

Cos gama = z/d = zf-zi/d

Angles are

Rule of tree

distance ---> time

incx ----> [tseg]

[tseg] = Time of Refresh that use GPU to draw objects.

[tseg] = incx * time/ distance However we need to multiply x 1000 because the time should be in milisecods

[tseg] = incx * time/ distance * 1000

With this relation we can adjust the Time of Refresh with Increment X.