Tutorial: Control procedural del movimiento en Blender

Introducción

Realiza los siguientes ejercicios usando blender y el lenguaje de programación python. Para seguir el tutorial os resultará de utilidad el material sobre Blender y python disponible en Aula Virtual.

Primeros pasos

En un entorno de animación por ordenador, la posición de cualquier objeto en la escena viene dada por sus coordenadas cartesianas, $p=(x,y,z)$. Por tanto, conseguir que un objeto se mueva consistirá en construir una secuencia de posiciones a medida que avanza el tiempo,

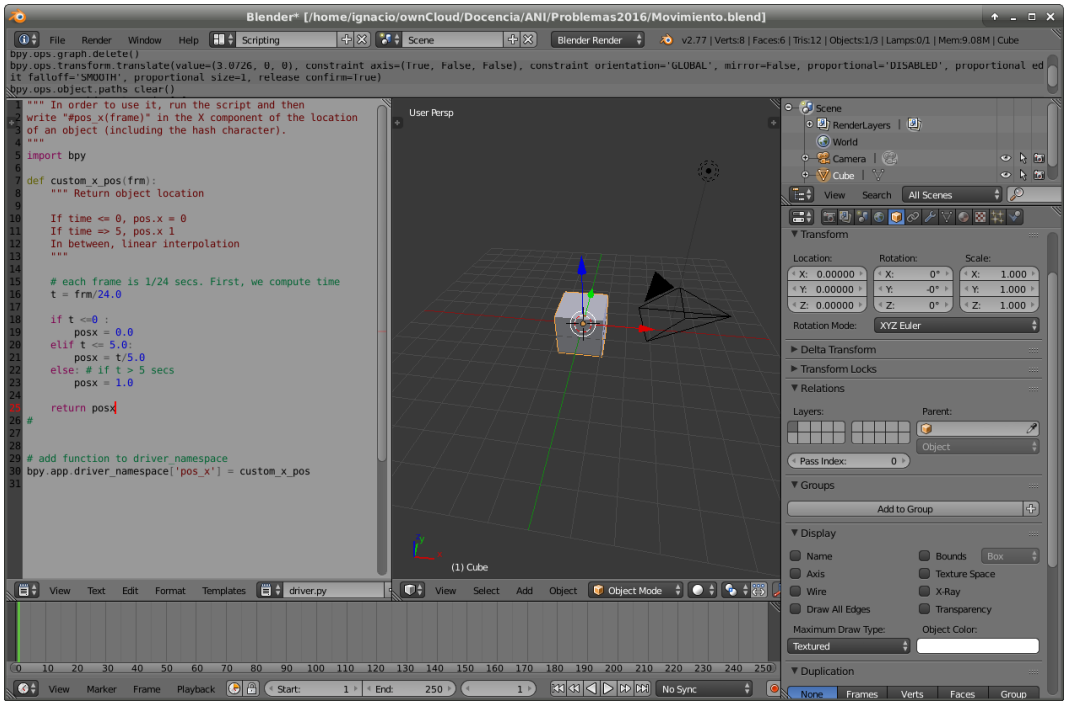
$$p(t) = (x(t),y(t),z(t))$$

En este ejercicio, vamos a conseguir un movimiento que consista en desplazar un objeto sólo en la dirección del eje x. Por medio de blender, debemos conseguir que un objeto pase de la posición $p(0) = (0,0,0)$ a la posición $p(5)=(1,0,0)$. Es decir, debe avanzar un metro en 5 segundos.

Para ello, la primera coordenada debe ser una función del tiempo, $p(t) = (f(t),0,0)$. La forma más sencilla de conseguir esto es usando interpolación lineal, lo que nos dará una velocidad constante durante todo el recorrido. La fórmula sería

$$f(t) = 0,\ si\ t \leq 0; \quad f(t) = \tfrac{t}{5},\ si\ t \in [0,1]; \quad f(t) = 1,\ si\ t > 1.$$

El fichero `Movimiento.blend`, que encontraréis en Aula Virtual, contiene la implementación de esta función en un script de python. La siguiente figura muestra el programa Blender cuando abrimos este fichero.



Para conseguir que un objeto utilice una función de python para calcular el valor de una de sus coordenadas vamos a utilizar Controladores (*Drivers* en inglés). Un Controlador de Blender permite asignar el valor de una propiedad (por ejemplo, la coordenada x de la posición de un objeto) utilizando una función matemática, un script en python o el valor de otra propiedad en la escena. Nosotros usaremos scripts en python.

Para conseguirlo debemos seguir tres pasos. El primero es implementar una función que devuelva, con return, el valor de la propiedad que queremos controlar. Este paso lo encontraréis resuelto en el fichero blend, por medio de la función `custom_x_pos`, entre la línea 7 y la línea 26.

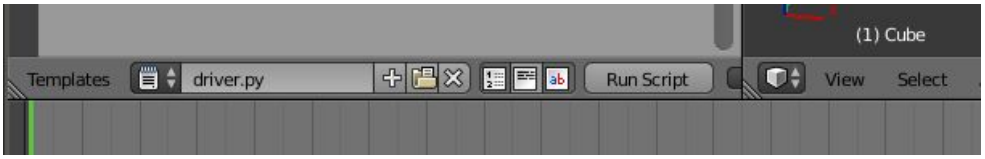
```
def custom_x_pos(frm):
    """ Return object location for frame 'frm'

    If time < 0, pos.x = 0
    If time > 5, pos.x = 10
    In between, linear interpolation
    """

    # each frame is 1/24 secs. First, we compute time
    t = frm/24.0

    if t <=0 :
        posx = 0.0
    elif t <= 5.0:
        posx = 10.0*t/5.0
    else: # if t > 5 secs
        posx = 10.0

    return posx
#
```

El segundo es dar de alta la función como Controlador. Blender mantiene un registro de los controladores existentes, y sólo aquellas funciones que hayan sido registradas pueden usarse como controladores. Para registrar el controlador, es necesario darle un nombre, e insertarlo en una tabla de la escena. En el ejemplo, esta acción se realiza en la línea 30 del script.

```
bpy.app.driver_namespace['pos_x'] = custom_x_pos
```

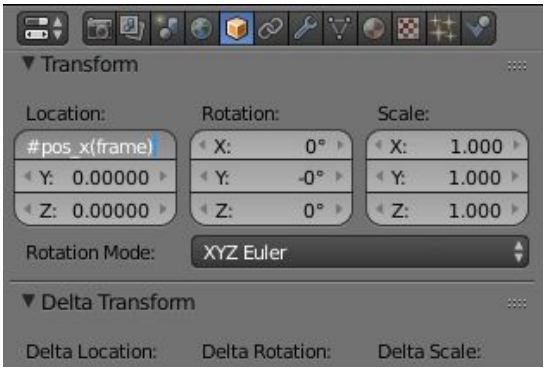
En esa línea se está indicando que el Controlador se va a llamar `pos_x`, y que Blender invocará la función `custom_x_pos` cuando se use este controlador. Por tanto, a partir de ahora sólo necesitamos saber el nombre `pos_x`. El controlador y la función pueden llamarse igual.

Para ejecutar el script, declarando la función y registrándola como Controlador, debemos pulsar el botón "Run Script" que aparece en la parte inferior del editor de texto. Si posteriormente cambiamos la función, podemos volver a pulsar el botón, y el Controlador se actualizará.



Por último, debemos usar el Controlador para modificar la posición del cubo.

- Seleccionamos el cubo haciendo clic sobre él con el botón derecho del ratón,
- Editamos la coordenada X de la propiedad location, que encontraremos en el panel derecho,
- Escribimos "#pos_x(frame)", incluyendo la almohadilla, y pulsamos Intro.



Con estos pasos, al usar la almohadilla, hemos asociado un Controlador a la propiedad. La propiedad a la que hemos asignado un Controlador aparecerá de color morado. En cada fotograma, Blender invoca nuestra función, pasándole la variable frame, que contiene siempre el número de fotograma actual. Esta variable ya está definida en Blender, y no es necesario que la declaremos previamente. En el manual de Blender podemos encontrar una lista de las variables y funciones que ya están definidas y podemos usar (véase el punto *Driver Namespace*).

Para comprobar el funcionamiento del controlador, basta con que lancemos la animación (Alt+A) o que seleccionemos diferentes fotogramas en la línea de tiempo.

Reproducir un movimiento

La función `custom_x_pos` traslada el cubo desde la posición 0 hasta la posición 10 en 5 segundos a velocidad constante. En animación, buena parte del éxito depende de conseguir que los movimientos se hagan al ritmo correcto. A continuación se proponen varios ejercicios que consisten en modificar la velocidad a la que se desplaza el objeto entre la posición inicial y final.

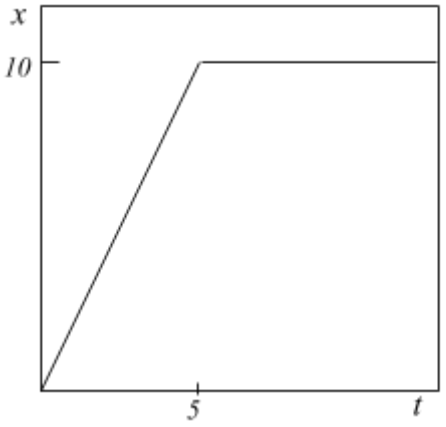
Reproducción de un movimiento dado

En primer lugar, vamos a intentar reproducir dos movimientos a partir de vídeos. Observa cada uno de los vídeos siguientes (vídeo A, vídeo B). En los vídeos, las barras horizontales están separadas por 1m.

Ejercicio 1: Construye dos nuevas funciones, `custom_x_A` y `custom_x_B`, que devuelvan la posición del cubo en cada fotograma, de manera que el movimiento sea lo más parecido posible al del vídeo.

Representación de posición frente a tiempo

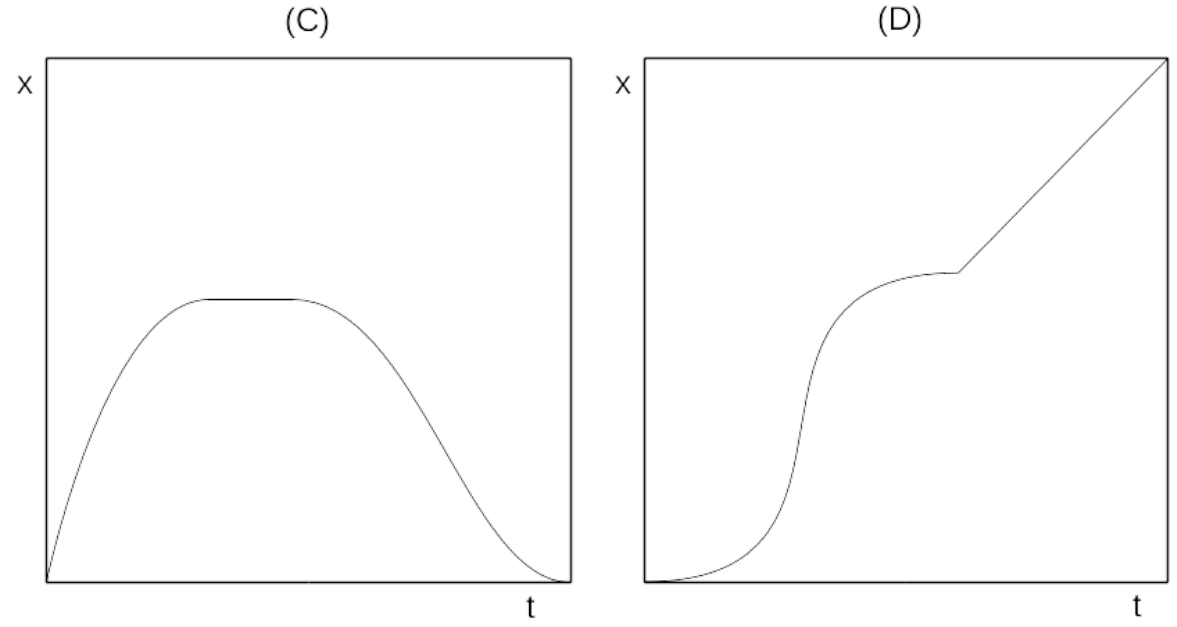
Como en este ejercicio sólo nos movemos a lo largo de una recta, otra forma de especificar el movimiento de un objeto es dibujar una gráfica que represente su distancia a la posición original, a lo largo del tiempo que dura el movimiento. Por ejemplo, para el movimiento de ejemplo inicial, el valor de la posición en el eje x a lo largo del tiempo se puede representar con la gráfica:



Para ver las gráficas correspondientes a un movimiento en Blender, puedes grabar el desplazamiento del objeto usando fotogramas clave, con la orden *Bake Action* (busca en google la forma de ejecutarla, o consulta al profesor). Esto generará fotogramas clave que puedes visualizar en el *Graph Editor*.

Ejercicio 2: Construye y visualiza las gráficas para los movimientos que has generado.

Ejercicio 3: Construye dos nuevas funciones `custom_x_C` y `custom_x_D`, que generen el movimiento correspondiente a las siguientes gráficas. Comprueba, visualizando los fotogramas clave correspondientes, que la gráfica de posición frente a movimiento obtenida coincide con las gráficas del enunciado.



Ejercicio 4: Consigue que el cubo describa un círculo cambiando de posición en el eje *x* y en el eje *y*. Para ello, debes dibujar (puedes hacerlo a mano) las funciones de cada coordenada frente al tiempo y construir dos funciones, `custom_x_loop` y `custom_y_loop`, y usarlas como drivers para esas coordenadas.