Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
Animación por Computadora
Otoño 2024
Elaboró: Josué Alejo Carbente Gómez - 202272637

Proyecto final de animación por computadora

El objetivo de este script es hacer una simulación del sistema solar en 3D utilizando VPython, mostrando las órbitas planetarias, la rotación de los planetas, y una cámara que sigue el movimiento de la escena.

Componentes clave:

CANVAS DE VPYTHON (SE USA EN SCENE):

Definición de parámetros del sistema solar
Este espacio es donde se renderiza la simulación

```
scene = canvas(title="Sistema Solar 3D", width=1400, height=700, background=color.black)
scene.lights = []
distant_light(direction=vector(1, 0, 0), color=color.white)
```

```
planetas = ['mercurio', 'venus', 'tierra', 'marte', 'jupiter', 'saturno', 'urano', 'neptuno
radios_orbita = [57.9, 108.2, 149.6, 227.9, 778.3, 1427, 2871, 4497]
radios_planeta = [2.4, 6.0, 6.3, 3.4, 69, 58, 25, 24]
periodos_orbitales = [88, 225, 365, 687, 4333, 10759, 30687, 60190]
retrogrados = [False, True, False, False, False, True, False, False]
inclinaciones = [0, 3.39, 0, 1.85, 0, 2.49, 82.2, 1.77]
```

De igual manera es donde configura el tamaño, color de fondo y la iluminación.

PLANETAS Y SOL:

El Sol es un objeto sphere con emisión de luz (emissive=True).
Los planetas son esferas con texturas específicas asignadas, que orbitan al Sol.
Se generan etiquetas flotantes para cada planeta.

```
sol = sphere(pos=vector(0, 0, 0), radius=10, color=color.yellow, emissive=True)
num_estrellas = 1000
for _ in range(num_estrellas):
    sphere(
        pos=vector(random.uniform(-1000, 1000), random.uniform(-1000, 1000), random.uniform
        radius=random.uniform(0.1, 0.3),
        color=color.white,
        emissive=True
    )
```

TEXTURAS Y ESTRELLAS:

Se aplican texturas desde archivos locales.
Se crea un fondo de estrellas y estrellas aleatorias para ambientar.

```
num_estrellas = 1000
for _ in range(num_estrellas):
    sphere(
        pos=vector(random.uniform(-1000, 1000), random.uniform(-1000, 1000), random.uniform(-1000, 1000)),
        radius=random.uniform(0.1, 0.3),
        color=color.white,
        emissive=True
    )
```

MOVIMIENTO Y ANIMACIÓN:
Los planetas giran alrededor del Sol en órbitas elípticas.
La cámara rota continuamente alrededor del sistema solar simulando una vista dinámica.
Los planetas rotan sobre su propio eje.

```
while True:
    rate(60)  # Controlar la velocidad de la animación
    t += factor_velocidad

    # Movimiento de la cámara
    cam_angle = 0.0005 * t  # Ángulo de rotación de la cámara
    scene.camera.pos = vector(
        cam_radius * np.cos(cam_angle),
        100,  # Altura de la cámara
        cam_radius * np.sin(cam_angle)
    )
    scene.camera.axis = vector(0, 0, 0) - scene.camera.pos  # La cámara siempre apunta al Sol

    # Actualización de la posición de los planetas
    for i in range(len(planetas)):
        angulo_orbital = 2 * np.pi * (t / periodos_orbitales[i])  # Cálculo del ángulo orbital
        if retrogrados[i]:
            angulo_orbital = -angulo_orbital  # Ajustar el sentido de rotación si es retrógrado

        inclinacion = np.radians(inclinaciones[i])  # cambiamos a radianes
        a = radios_orbita[i] * factor_distancia  # Distancia de la órbita
        b = a * 0.9  # Forma de la órbita (elipsoide porque no son círculos)

        x = a * np.cos(angulo_orbital)  # Coordenada X del planeta
        y = b * np.sin(angulo_orbital)  # Coordenada Y del planeta
        z = y * np.sin(inclinacion)  # Coordenada Z del planeta

        planetas_obj[i].pos = vector(x, y, z)  # posición de los planetas
        etiquetas[i].pos = planetas_obj[i].pos + vector(0, 0.1, 0)  # Actualizamos la posición de la etiqueta
        planetas_obj[i].rotate(angle=factor_rotacion, axis=vector(0, 1, 0))  # Rotación del planeta
```

DIAGRAMAS: DISEÑO DE FUNCIONES Y FLUJO