Universidad Nacional de Moquegua
Escuela Profesional de Ingeniería de Sistemas e Informática

Paralelizacion de pixeles

ALGORITMOS PARALELOS

Estudiante: Carlos Eduardo Patina Rojas
Docente: Honorio Apaza Alanoca

15 de julio de 2025

INTRODUCCIÓN

En el ámbito de la programación, las animaciones no solo representan un componente visual atractivo, sino también una herramienta poderosa para comprender y visualizar procesos computacionales. Este proyecto implementa una animación paralelizada de un dinosaurio pixelado que se desplaza horizontalmente sobre un escenario generado proceduralmente.

La implementación utiliza tres componentes principales:

- **Matrices NumPy** para representar los objetos gráficos (dinosaurio, nubes, árboles y piedras)
- **Threading** para paralelizar la generación de los 60 frames de animación
- **Matplotlib** para la visualización final con `FuncAnimation`

La animación resultante muestra un dinosaurio de 17x14 píxeles cuyas patas alternan movimiento mientras avanza, sobre un escenario de 20x50 píxeles que incluye elementos decorativos posicionados estratégicamente. La generación paralela con 4 hilos permite precargar eficientemente todos los frames antes de la visualización.

Este enfoque demuestra cómo técnicas fundamentales de programación (matrices, paralelismo y visualización) pueden combinarse para crear animaciones eficientes sin requerir motores gráficos complejos, siendo particularmente útil en contextos educativos y prototipado rápido.

OBJETIVOS

Objetivo general:

Implementar una animación paralelizada de dinosaurio pixelado usando matrices NumPy y threading, demostrando la eficacia de estas técnicas para procesamiento gráfico.

Objetivos específicos:

- Modelar gráficos pixelados mediante matrices binarias (0/1) para dinosaurio y elementos del escenario
- Implementar movimiento animado alternando las patas del dinosaurio cada frame
- Generar proceduralmente un escenario con distribución aleatoria de nubes (parte superior), árboles y piedras (anclados al suelo)
- Paralelizar la generación de frames dividiendo el trabajo en 4 hilos
- Optimizar la composición gráfica mediante operaciones matriciales con `np.maximum`
- Visualizar la animación a 10 FPS (100ms por frame) usando `FuncAnimation`
- Garantizar sincronización segura entre hilos mediante `threading.Lock`
- Documentar el proceso como referencia para proyectos educativos de gráficos y paralelismo

MARCO TEÓRICO

El proyecto combina tres áreas fundamentales:

1. Gráficos por Matrices

- Cada objeto se define como matriz binaria:
  - Dinosaurio: 17x14 píxeles
  - Nube: 5x7 píxeles
  - Piedra: 4x6 píxeles
  - Árbol: 10x8 píxeles
- Composición mediante `np.maximum` para superposición
- Coordenadas:
  - Nubes: y = 0-3
  - Objetos terrestres: y = ALTO - altura

2. Paralelización con Threading

- 60 frames divididos en 4 hilos (15 frames/hilo)
- Mecanismo de sincronización:
  - `Lock` para acceso seguro al array compartido
  - División estática del trabajo
- Limitaciones por GIL (Global Interpreter Lock)

3. Animación con Matplotlib

- `FuncAnimation` para ciclo de renderizado
- Configuración:
  - Intervalo: 100ms (10 FPS)
  - Modo `blit=True` para optimización
- Visualización binaria (cmap='binary')

RESULTADOS

Implementación Funcional

El sistema completo consta de:

- 3 objetos gráficos definidos como matrices NumPy
- Generación procedural de escenario
- Animación de dinosaurio con:
  - Movimiento horizontal progresivo
  - Alternancia de patas cada frame
- Pipeline paralelizado:
  - Precálculo de frames
  - Renderizado suave

Especificaciones Técnicas

Parámetro	**Valor**
Dimensión escenario	20x50 píxeles
Total frames	60
Hilos	4
Frames/hilo	15
Intervalo animación	100ms
Tamaño dinosaurio	17x14 píxeles
Posición vertical dinosaurio	y=3

Código Clave

```

hilos = []
por_hilo = N_FRAMES // N_HILOS

for i in range(N_HILOS):
    ini = i * por_hilo
    fin = (i + 1) * por_hilo if i != N_HILOS - 1 else N_FRAMES
    h = threading.Thread(target=generar_frames, args=(ini, fin))
    hilos.append(h)
    h.start()
```

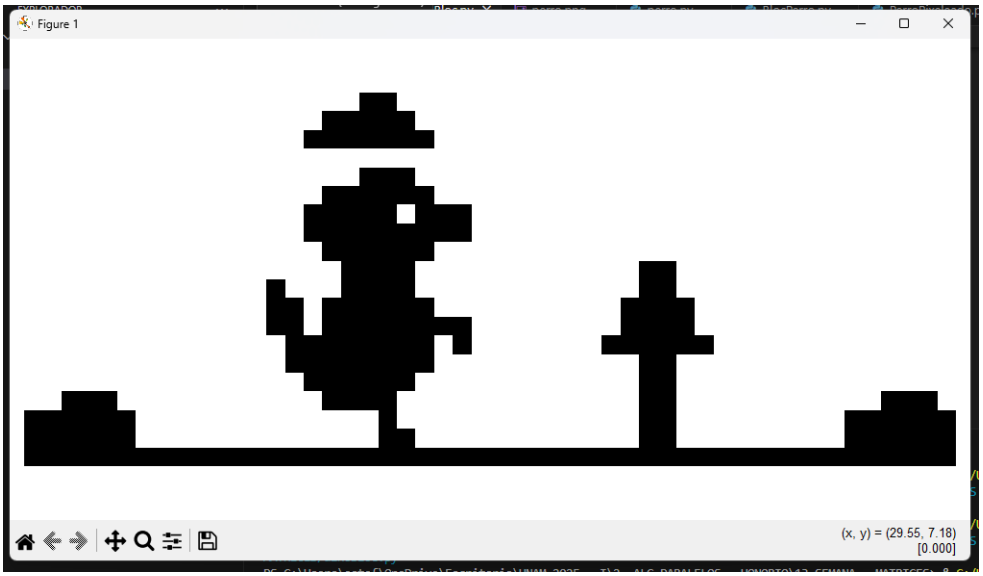

Figura 1: Frame representativo mostrando: 1) Dinosaurio, 2) Nube, 3) Árbol, 4) Piedra

CONCLUSIONES

Logros Principales

- Implementación exitosa de animación paralelizada
- Composición eficiente mediante operaciones matriciales
- Distribución balanceada de carga entre hilos
- Visualización fluida a 10 FPS

Aprendizajes Clave

- Las matrices NumPy son ideales para gráficos pixelados
- `np.maximum` simplifica la composición gráfica
- Threading acelera tareas de generación de contenido
- La sincronización con `Lock` es crucial en paralelismo

Mejoras Futuras

- Implementar movimiento vertical del dinosaurio
- Añadir sistema de colisiones
- Probar con `multiprocessing` para evitar limitaciones del GIL
- Optimizar renderizado con técnicas de double buffering

Este proyecto constituye una base sólida para explorar técnicas más avanzadas de gráficos y paralelismo, demostrando que con componentes básicos se pueden lograr resultados visuales efectivos y didácticos.