Optimización de estructuras lego

Búsqueda de Soluciones e Inferencia Bayesiana (enero–junio 2025)

Entregable para segundo parcial

IA

Salgado Flores Irving
Rafael

Descripción del Proyecto

Contexto

El problema se enmarca en la simulación computacional y modelización de estructuras con bloques de lego. Se centra en analizar el comportamiento de un puente al ser atravesado por un carrito, considerando efectos físicos como gravedad, peso y velocidad.

Problema

Dentro de este proyecto se busca encontrar la configuración óptima de piezas LEGO para construir una estructura 3D que conecte dos puntos definidos en el espacio, utilizando un algoritmo evolutivo implementado en Godot.

Cada individuo, representado mediante pilin datos que incluyen la posición, rotación, velocidades y estado (activo o inactivo) de sus componentes, se evalúa en función de la estabilidad de la estructura, donde se busca minimizar el contador de activación y asegurar que la prueba (pj) se complete satisfactoriamente.

La optimización se orienta a obtener la solución que use la menor cantidad de piezas posibles sin comprometer la integridad estructural, considerando las coordenadas de los puntos inicial y final, y las características específicas de cada tipo de pieza LEGO (dimensiones, formas de conexión, y ubicación en el espacio).

La salida del algoritmo es, por tanto, una secuencia ordenada de bloques que especifica la ubicación, el tipo y la orientación de cada pieza, garantizando que el vehículo de prueba (en este caso, un carro de bomberos) logre atravesar la estructura de manera exitosa.

Planteamiento del problema

Individuos

Cada individuo es una entidad única que agrupa información de una pieza específica y de sus componentes, donde cada componente registra su estado (activo o inactivo) y parámetros como posición, rotación y velocidades. Formalmente, podemos definir al individuo como:

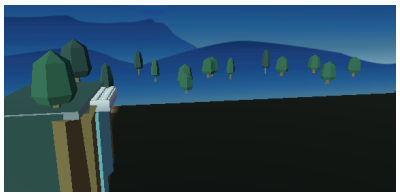
$$I_i = [id_i,\ N,\ \{(P_j,\ R_j,\ V_{lj}, V_{aj}, a_j, c_j, p_j)\}\ _{j=1}^{n}]$$

- ➔ Donde id_i es su identificador único. N es el nombre de la pieza $P_j = (x_j, y_j, z_j)$ representa la posición de la pieza (dentro del rango $pos_{min} \leq P_j \leq pos_{max}$).
- ➔ R_j es la rotación de la pieza.
- ➔ a_j es si la pieza actual está activa o no.

- ➔ c_j es el contador de activación
- ➔ p_j indica si la prueba a sido completada, para cada componente j con $j = 1, 2, ..., n.$

Metodología

Todo será realizado dentro del motor de juegos Godot, ya que permite el uso de físicas esenciales para nuestro proyecto, así mismo nos permite observar nuestro código de una forma más visual.

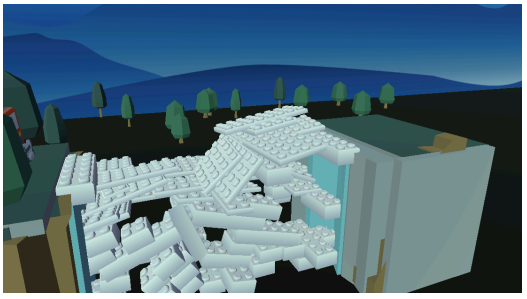


Para este proyecto usaremos el algoritmo evolutivo. Dado que vamos a usar este tipo de algoritmo se tiene que mencionar el cómo se va a evaluar cada individuo, para ello tenemos c_j y p_j, siendo uno más importante que el otro, los individuos serán calificados dependiendo de su c_j entre menor sea el valor que tiene c_j será un individuo mejor valorado. También tenemos p_j la cual es una variable true or false que se encargará de determinar si el individuo pasa la prueba o no, y en caso de que si lo haya pasado ese sin importar su c_j será el mejor individuo, ahora cuando se tienen más individuos que contengan p_j true, entonces se ordenan dependiendo de su c_j.

Prueba

Es importante mencionar que para nuestro p_j, como se mencionaba anteriormente nuestro individuo debe de ofrecer una estructura estable, por lo que se le pasara un carro, que en este caso es un carro de bomberos, que se evaluará si logra llegar a la otra montaña pasando por la pieza objetivo.

Resultados.



Durante las simulaciones se logró conseguir diversas estructuras que se completaron con el objetivo de la creación de un puente, sin la ruptura de la estructura de la estructura.

Cabe recalcar que la “fitness function” fue modificada un poco, ya que solo busca minimizar la cantidad de piezas cuando, al menos ⅓ de la población de individuos a logrado completar con éxito la prueba.

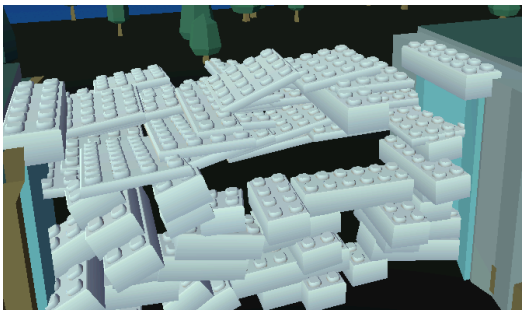
Lo que provoca que tengamos una función que va aumentando la cantidad de piezas dentro del tablero.

“Mejor individuo de la generación1: ID =9, Prueba completada =false, Cantidad de piezas =78”

“Mejor individuo de la generación45: ID =10, Prueba completada =true, Cantidad de piezas =175”

“Mejor individuo de la generación121: ID =18, Prueba completada =true, Cantidad de piezas =56”

Pero lo curioso es que cuando se cumple la condición anterior se logran estructuras con una mejor capacidad de carga, con menores cantidades de piezas, lo que significa que se va cumpliendo el objetivo de manera satisfactoria.



En conclusión, tenemos un conflicto debido a que como se puede ver en las capturas de la simulación, no se logra encontrar alguna estructura típica de un puente, esto se debe a la complejidad en el tiempo que se lleva, debido a que nuestro simulador requiere que se visualice el código y estos necesitan un tiempo para funcionar, entonces tomaría mucho tiempo real para poder encontrar una solución global.