1 Introducción

1.1 Descripción del problema

El problema que aborda nuestro proyecto es el analisis de una campaña militar en la cual se enfrentan dos bandos, el bando defensor y el atacante. Cada uno de ellos tiene su propia condición de victoria; El defensor deberá acabar con el ejército atacante antes de que este tome todas las posiciones estratégicas, mientras que el objetivo de los tcntes es precismente conquistar todas esas posiciones.

Cada bando cuenta con un ejército dividito en pelotones independientes, cada uno de estos pelotones tiene sus propias carcterísticas y se mueven por un mapa con distintos tipos de terreno, estos terrenos también presentan su propio conjunto de características. Tanto las características de cada pelotón como las del terreno en cada área del mapa pueden ser conocidas o no por los ejércitos de cada bando.

1.2 Objetivos

El objetivo de nuestro proyecto es analizar los resultados de los enfrentamientos para poder deducir que condiciones pueden asegurar la victoria de alguno de los bandos, así como las mejores estratégias para enfrentarse a distintos escenarios.

2 Simulación

2.1 Detalles de la simulación

Para simular el problema tomamos como agentes a los pelotones de cada bando, y como ambiente al terreno y las variables asociadas a este y a los agentes de las cuales habaremos más adelante.

Decidimos utilizar ademas un modelo de simulación discreta en el que las acciones de los agentes así como la actualización del ambiente ocurre en intervalos discretos llamados turnos, cada turno se divide a su vez en tres faces inicio, ejecución y final. Al inicio se actualizan las estadísticas del pelotón tales como la moral, los suministros, el número de heridos y la cantidad de soldados; en la ejecución el pelotón realizará una acción de su elección; en la etapa final se vuelven a actualizar las variables del pelotón ademas de la de las de otro pelotón al que haya atacado o el de un terreno tipo castillo que haya conquistado.

La decisión de utilizar un enfoque por turnos es porque permite simplificar mucho la ocurrencia de eventos complejos que involucran muchos parámetros, y esta fue a su vez la que impulsó la decisión de utilizar un modelo de simulación discreta, ya que es el que mejor se adapta a un sistema de turnos.

2.2 Generación de terreno

El mapa es tratado como una matriz de hexágonos donde cada hexágono es un uno de los 5 tipos de terrenos (Llanura, Montaña, Bosque, Agua y Castillo) El ancho y la altura del mapa es seleccionada previamente y se ejecuta un algoritmo de varios pasos para determinar que tipo de terreno se le asigna a cada casilla.

En la mayoría de los pasos que explicaremos a continuación nos apoyamos en el algoritmo de generación de mapa de ruido Perlin el cual genera una matriz de valores aleatorios entre 1 y -1 pero que a diferencia de un mapa usual de ruido, en este los valores no dan cambios bruscos, en su lugar queda un mapa de ruido donde se pueden apreciar grupos de valores similares. Elegimos utilizar este tipo de mapa de ruido para apoyar la generación de alturas, bosques y cadenas montañosas ya que en la naturaleza este tipo de terrenos suelen encontrarse de esa forma, con lo cual generamos mapas mas similares a un posible mapa real que si usaramos en su lugar un mapa de ruido convencional.

El primer paso del algoritmo es generar las alturas, para ello generamos una matriz con el algoritmo de ruido Perlin que llamaremos matriz de altura (H), una vez obtenida sumamos una constante positiva predeterminada a la matriz para obtener una mayoria de valores positivos y luego lo multiplicamos por otra constante positiva llamada escala. Para las celdas de la matriz con valor positivo se le asignará un terreno de tipo llanura con valor de altura igual al valor de la matriz, mientras que a las que tengan valor negativo se les asignará un terreno de tipo agua con altura 0.

El segundo paso es la generación de los bosques. Para ello calculamos primeramente dos nuevas matrices. Una es la matriz de ruido Perlin de los bosques (F) y la segunda es una matriz de cercanía al agua (W) la cual tiene valores entre 0 y 1 en dependencia de cuan cercana es cada celda a una celda de tipo agua, donde el valor es 1 si la celda es de tipo agua y 0 si es la celda con la diatancia más grande posible a la celda de tipo agua más cercana, esta matriz se calcula utilizando un algoritmo de BFS. Luego calculamos la matriz F + W y a cada celda que no sea de tipo agua y que tenga un valor por encima de un umbral prefijado se le asigna el tipo bosque. Al sumar la matriz W a la matriz de ruido Perlin damos una mayor probabilidad a la generación de bosques cerca del agua.

El tercer paso del algoritmo es la generación de montañas la cual se genera de forma muy similar a la de los bosques, calculando una nueva matriz de ruido Perlin para las montañas (M) pero en lugar de usar W utilizamos la matriz H. Luego calculamos la matriz M + H y a cada celda que no sea de tipo agua y que tenga un valor por encima de un umbral prefijado se le asigna el tipo montaña. Al sumar la matriz H a la matriz de ruido Perlin damos una mayor probabilidad a la generación de montañas en celdas altas.

El cuarto y último paso en la generación de terreno es generar los castillos. El número de castillos que se generan es una constante predefinida n. En este paso se realizan n iteraciones donde en cada iteración se genera un castillo. En este paso tenemos 3 matrices: la matriz de ruido de los castillos (R), en este caso es una matriz usual de ruido y no una de ruido de Perlin ya que no nos interesa que los castillos se generen agrupados, la matriz W y la matriz de cercanía a un castillo (C) que en la primera iteración es ina matriz de ceros pero que se irá recalculando al inicio de cada iteración posterior con un BFS de manera similar a la matriz W. En cada iteración se calcula la matriz R + W - C y se hacen 0 los valores de las filas de los 2 primeros tercios de la matriz. Luego se ordenan las filas empezando por la de mayor valor y en la primera fila de este orden que no sea de tipo agua se coloca un castillo y se realiza la siguiente iteración. De esta manera generamos n castillos en el tercio inferior del mapa que tienden a estar separados entre ellos y cercanos al agua.

2.3 Variables del terreno

Como se pudo apreciar en la explicación anterior, al finalizar la generación del mapa cada celda de este tendrá asignado uno de los 5 tipos de terreno (Llanura, Montaña, Bosque, Agua o Castillo). Cada uno de estos terrenos tienen una serie de características que afectan el como los agentes interactuan con ellos.

- Reducción de Movimiento: Esta característica define cuanto movimiento le va a costar a un agente atravesar esa casilla.
- Visibilidad: Similar a la reducción de movimiento, este parámetro define cuanto se verá obstaculizada la visión de un pelotón al ver atravez del terreno.
- Ventaja: El valor de esta variable está asociado a cuanta ventaja ofrece ese terreno en batalla a los pelotónes que se encuentren sobre ella.
- Bando: Exclusivo de los terrenos de tipo castilo, este parametro indica a que bando pertenece la fortaleza ubicada en esa casilla.
- Contador de Conquista: También exclusivo de los castillos, indica cuantos turnos se necesitan para ser conquistado por un pelotón.

2.4 Agentes

Los agentes de nuestra simulación como mencionamos anteriormente son los pelotones de cada ejército. Estos estan implementados como agentes de deductivos los cuales se basan en las condiciones del ambiente para decidir que acción realizar en ese turno. Estos a su vez pueden ser de uno de cuatro tipos distintos cada uno con unas reglas distintas que definiran comportamientos distintos.

- Vanguardias: Un tipo de pelotón de los atacantes. Avanzan por el mapa como exploradores y tienden más a provocar batallas y tomar fortalezas.
- Refuerzos: El segundo tipo de pelotón atacante, se mantiene detrás de los vanguardias y prioriza las batallas donde estén participando sus aliados. Su avance es más moderado y tienden menos a iniciar los enfrentamientos.
- Guardianes: Un tipo de pelotón defensor, tienden a mantenerse cerca o ocupando castillos, no suelen realizar acciones que les hagan alejarse mucho de estos.
- Emboscadores: El otro tipo de pelotón defensor. Se mueve por el mapa para interceptar a los enemigos aprovechando al máxime posible las ventajas del terreno y debilitando a los ejércitos atacantes.

2.5 Variables de los agentes

Cada uno de los agentes tienen una serie de características que varian a lo largo de la simulación y que influyen tanto en sus decisiones como en la de otros agentes, ademas de influir entre ellas para su modificación.

- Número de Soldados: La cantidad total de unidades del pelotón que determina su tamaño (incluye soldados sanos y heridos). Este parámetro influye enormemente en los resultados de las batallas y en otras variables como la Visibilidad y el Grado de Vigilancia.
- Número de Heridos: Esta estadística determina el número de soldados subconjutos del total que se encuentran heridos y por ende incapacitados para participar en las batallas. Esta estadística se modifica al inicio de cada turno restando aquellos heridos que sanaron o fallecieron y al final de cada asalto de una batalla añadiendo aquellos que resultaron heridos en él.
- Moral: Esta variable influye en el rendimiento de los soldados en la batalla y en la ocurrencia de eventos que disminuyen el número de estos (deserción). Esta estadística disminuye a cada turno en base a la cantidad de suministros restantes y al tamaño del ejército, también puede aumentar o disminuir como consecuencia de eventos como perder o ganar una batalla o conquistar una fortaleza.
- Suministros: Este parámetro disminuye cada turno, influye directamente en la moral y es la variable que garantiza que la campaña se realiza en tiempo finito.
- Distancia de Movimiento: Esta variable depende del número de tropas, la moral y el número de heridos y Determina el maximo de casillas que puede moverse una tropa en un turno (sin contar las condiciones del terreno).

- Visibilidad: Esta estdística influye positivamente en la provabilidad que tiene un pelotón de ser detectado por otro al estar dentro de su alcance de visión.
- Grado de Vigilancia: Parámetro que influye de manera positiva en la facilidad que tienen los pelotones de detectar a otros que estén en su rango de visión.
- Estado de Detección: Un booleano que determina si un pelotón está siendo detectado por algún pelotón del ejército enemigo.

2.6 Movimiento

Una de las acciones que pueden tomar los agentes en su turno es moverse a otra casilla. Para calcular todas las casillas válidas a las que puede moverse el pelotón en su turno se utiliza el algoritmo de Djistra en un grafo donde los vértices son las casillas y existe la arista (x,y) si x y y son adyasentes y su peso será la reducción de movimiento de x. El algoritmo no recorre necesariamente todo el grafo ya que utiliza como contador el movimiento del agente al cual le va restando el peso de las aristas por las que va pasando y se detiene cuando este llega a cero.

2.7 Visión

Un pelotón puede o no tener visión de un enemigo que esté en su cercanía. Para determinar esto los pelotones tienen un radio de visión, para cada cada casilla se determinan las casillas que se encuentran entre esta y la casilla del pelotón creando un camino por el que pasa la visión para cada una. En caso de haber un pelotón enemigo dentro del radio de contrastarán su visibilidad y la vigilancia del enemigo con el factor de ocultación que le ofrecen los terrenos del camino hacia su casilla. Con esto hay una probabilidad basada en esos factores que determina si el pelotón es detectado o no.

2.8 Batallas

Una de las principales acciones que se lleban a cabo en la simulación es atacar a un pelotón del equipo contrario. Al efectuarse esta acción se produce una batalla entre ambos pelotones la cual puede durar varios turnos. En cada turno se calcula el daño que cada pelotón le causa a su rival, este se calcula multiplicando el ratio de baja (0.3) por el número de soldados, el factor de moral y la ventaja del terreno, este valor sería el valor esperado que se utilizará para obtener el valor final que es una variable aleatoria con distribución normal. Los heridos se calculan de igual manera pero con ratio 0.1. Una vez calculado las bajas de cada bando finaliza el turno, y en el turno del otro pelotón este puede elegir si continuar con la batalla o intentar escapar. Cuando los soldados sanos de una de los dos contrincantes llegue a cero la batalla finaliza automáticamente y dicho pelotón desaparece.

3 Componentes de IA

3.1 Búsqueda

Al realizar una acción en casi la totalidad de las ocasiones el agente tendrá que desplazarse a una ubicación del mapa. Para encontrar en estos casos la mejor ruta en un un tiempo aceptable empleamos el algoritmo de búsqueda de A*, en el cual nuestra función g(x) es la cantidad de turnos que te toma alcanzar la casilla y h(x) es la distancia euclidiana al destino. Además la función h está multiplicada por una constante de peso w ¡ 1 para equlibrar el algoritmo.

3.2 Conocimiento

La toma de decisiones de los agentes está modelada con un sistema de Creencia, Deseo, Intención. Las creencias son deducciones tomadas en base a la información que el agente tiene del ambiente; un enemigo cerca de un castillo, un pelotón más fuerte que el mio cerca de mi posición, ect; a partir de estas creencias se decide que acciones tomar. El deseo está dado en gran medida por el tipo de agente por un lado y por las creencias que este tenga, este actua como un filtro de entre todas las acciones posibles quedandose con un conjunto reducido de estas. La intención es la elección final de una acción concreta en base a las posibles acciones del deseo; mantener posición en un castillo para defenterlo, alejarse de la posición de un enemigo, ect.

3.3 Procesamiento de Lenguaje Natural

El usuario puede controlar el comportamiento de un agente mediante un chat que interpreta los mensajes del usuario y los traduce en acciones para ese pelotón. Para ello nos apoyamos en la IA Gemini a la cual se le da a conocer una lista con todoslos tipos de acciones que existen; atacar a (pelotón), moverse a (casilla), Conquistar el castillo de (casilla), ect; Gemini interpretará el mensaje y decidirá cual de las acciones es a la que el usuario hace referncia y el agente procederá a realizarla.

4 Resultdos

5 Conclusiones

5.1 Recomendaciones

Para la continuación de este proyecto se pueden añadir más factores que complejicen la simulación con el objetivo de volverla más fiel a la realidad commo podría ser un sistema de climas, agregar varios tipos de unidades (caballería, arqueros, asedio, ect), añadir comportamientos para los agentes, ect. Es viable además agregar interfaces que permitan al usuario modificar los parámetros de manera más sencilla para poder analizar diferentes escenarios con mayor facilidad.♠