SIMULACIÓN Y VISUALIZACIÓN DE LA DINÁMICA DEL COMPORTAMIENTO DE MULTITUDES USANDO ACELERADORES GRÁFICOS

DAVID LEONARDO BURGOS GRANADOS

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO-MECANICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2015

SIMULACIÓN Y VISUALIZACIÓN DE LA DINÁMICA DEL COMPORTAMIENTO DE MULTITUDES USANDO ACELERADORES GRÁFICOS

AUTOR:
DAVID LEONARDO BURGOS GRANADOS

TRABAJO DE GRADO PARA OPTAR EL TÍTULO DE INGENIERO DE SISTEMAS E INFORMÁTICA

DIRECTOR:
Ph D. CARLOS JAIME BARRIOS HERNÁNDEZ

CO-DIRECTOR:
MSc. HUGO ANDRADE SOSA.

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO-MECANICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2015

AGRADECIMIENTOS

Agradezco a dios por darme la estabilidad y tranquilidad suficiente para lograr cumplir los objetivos planteados durante todo este proceso de estudio y aprendizaje.

A mi papá Ricardo Burgos Castillo y mi mamá Morelia Granados Álvarez, que siempre han estado ahí para apoyarme en cada una de las decisiones que he tomado, me han dado siempre todas las herramientas necesarias para cumplir mis objetivos y han sido mi soporte y gran respaldo en cada uno de los proyectos que en toda mi vida he emprendido, a los cuales les debo todo lo que soy y les agradeceré toda mi vida.

A mi hermano Ricardo Andrés Burgos Granados y mi hermana Jeimmy Johana Burgos Granados, que son parte fundamental de mi núcleo familiar, he contado siempre con su respaldo y confianza, y son el motor para hacer las cosas bien, logrando grandes éxitos en mi vida personal y profesional.

Al profesor Carlos Jaime Barrios, por dirigir mi proyecto de grado y brindarme la confianza, colaboración y continuo apoyo en todas las etapas del proyecto.

Al profesor Hugo Andrade Sosa por su tiempo, respaldo y dedicación aceptando conformar el equipo de trabajo siendo mi codirector de proyecto de grado.

TABLA DE CONTENIDO

TABLA DE IMÁGENES

GLOSARIO

Default Quality Rendering: Cuando no se necesita una alta calidad hacen, pero desea reducir el tiempo de empate en la vista de escena y aumentar la eficiencia.

Fps: Imágenes por segundo, también conocido como Tasa de Refrescamiento, es la velocidad (tasa) a la cual un dispositivo de imagen muestra imágenes llamadas cuadros o fotogramas. El termino aplica igualmente a películas y cámaras de video, gráficos computacionales y sistemas de captura de movimiento. La tasa de refrescamiento se expresa en Cuadros Por Segundos o Frames Per Second (FPS) en Inglés.

Lógica difusa: El término es usado habitualmente para hacer referencia a la interacción entre el hombre y lo que lo rodea, más específicamente a la interacción hombre-máquina[1].

Render: Proceso de generar una imagen o vídeo mediante el cálculo de iluminación GI partiendo de un modelo en 3D.

Spot: Es una (posición) punto en el espacio 3D[2].

Viewport 2.0: Para una vista de la escena de alto rendimiento que optimiza grandes escenas. Se le permite interactuar con escenas complejas con muchos objetos, así como grandes objetos con geometría pesada.

[1] http://www.biblioteca.udep.edu.pe/bibvirudep/tesis/pdf/1_185_184_133_1746.pdf
[2] https://basefount.atlassian.net/wiki/display/MDE/Spot

Cuando esta opción está activada, los puntos de vista de escena se dibujan con ajustes de baja calidad por el procesador de hardware.

RESUMEN

TÍTULO: SIMULACIÓN Y VISUALIZACIÓN DE LA DINÁMICA DEL COMPORTAMIENTO DE MULTITUDES USANDO ACELERADORES GRÁFICOS.*

AUTOR: DAVID LEONARDO BURGOS GRANADOS.**

PALABRAS CLAVES: Simulación, multitudes, GPU, Arquitecturas aceleradas, Visualización, Modelado, GeForce, Nvidia, graficos 3d, rendimiento computacional.

DESCRIPCIÓN:

En este proyecto se propone una metodologia básica para orientar el realizar simulación de multitudes y se ilustra con un modelo sencillo a modo de ilustración aplicadola a un caso real, con ayuda de motores que facilitan la simulación y visualización y que actualmente son utilizados a nivel profesional en la insdustria de los video juegos, cine e investigacion.

Se describe el uso de las herramientas software en el proceso de construir un modelo matemático para simular el comportamiento de multidudes, y basado en éste proceso se desarrolla un modelo básico especifico para simular un caso plateado previamente.

Se proponen lineamientos básicos para realizar simulación y visualización de multitudes usando modelos y herramientas profesionales que hacen uso de arquitecturas aceleradas.

Además se evalúa el rendimiento computacional teniendo en cuenta la cantidad de frame o imágenes generadas por segundo, la carga porcentual del procesador y la carga porcentual de GPU y se ilustra la importancia del uso de las GPU en la visualización de geometrías 3D animadas simulando multitudes definiendo densidades, reglas y obstáculos.

Se ilustra la necesidad de realizar simulación y modelado de multitudes para la ayuda de toma de decisiones, video juego, publicidad e investigación principalmente, y se pone a disposición de los interesados en este tipo de fenómenos las herramientas y planteamientos básicos para su realización.

---

* Trabajo de grado.

** Universidad Industrial de Santander, Facultad de Físico Mecánicas, Escuela de Ingeniería de Sistemas, PhD Carlos Jaime Barrios, Msc Hugo Andrade Sosa.

ABSTRACT

TITLE: SIMULATION AND DISPLAY OF DYNAMIC CROWD BEHAVIOR USING GRAPHIC ACCELERATORS.*

AUTHOR: DAVID LEONARDO BURGOS GRANADOS.**

KEY WORDS: Simulation, crowd, GPU, accelerated architecture, display, Modeling, GeForce, Nvidia, 3d graphics, computational performance.

DESCRIPTION:

This project proposes a basic methodology to guide the conduct crowd simulation and it is illustrated by applying it with a simple model by way of illustration to a real case, using engines that facilitate the simulation and visualization and they are currently used professional level in video games industry, movies and research.

The use of software tools in the process of building a model mathematical to simulate the behavior of crowd is described, and based on this process a specific model basic is developed to simulate a case reised previously.

Basic guidelines are proposed for simulation and visualization of crowds using models and professional tools that use accelerated architectures.
Besides it is evaluated the performance and the importance of using the GPU on display animated 3D geometries simulating crowds defining densities, rules and constraints.

Also the computational performance is assessed taking into account the amount of frame or images generated per second , the percentage load of the processor and GPU load percentage and the importance of using the GPU to display animated 3D geometries illustrated defining simulating crowds densities , rules and obstacles.

The need for simulation and modeling multitude to help decision making , video game, advertising and research mainly illustrated, and are available to those interested in this type of phenomena the basic tools and approaches to its realization .

---

* Graduation Project

** Santander Industrial University , Faculty of Physical Mechanics , School of Systems Engineering , PhD Carlos Jaime Barrios, MSc Sosa Hugo Andrade

INTRODUCCIÓN

La simulación de multitudes ha sido utilizada en diversas áreas de la industria e investigación; entes gubernamentales hacen uso de estas simulaciones para la planificación urbana, tráfico, crecimiento poblacional entre otros campos de aplicación, buscando prever fallas en la dinámica de este tipo de fenómenos y plantear, por ejemplo, planes de emergencia en caso de caos por terremotos, tsunamis, atentados, incendios y demás casos en los cuales se ven implicadas grandes cantidades de personas, vehículos, animales, partículas etc.

Gracias a la simulación de multitudes ha podido entender mejor el comportamiento de algunos animales que viven en multitud y naturalmente migran de un lugar a otro, o simplemente conviven, como por ejemplos aves, hormigas, peces, pingüinos etc. pero también observar las consecuencias y comportamientos de éstos en caso de desplazamientos de sus lugares de hábitat natural sea por tala indiscriminada de árboles, pesca, edificaciones, explotación minero energética etc.

Además hay que resaltar la importancia de determinar el tipo entorno, contexto y situación en el cual se está realizando el estudio, así como las acciones y reacciones en la dinámica del comportamiento ya que éstos juegan un papel primordial en la toma de decisiones en el modelo.

La simulación de altas densidades de agentes, geometrías complejas y cálculo de decisiones que conforman la multitud, implican una alta exigencia computacional debido a la gran cantidad de datos que se deben procesar y proyectar como imagen en un entorno gráfico, para esto, se hace uso de arquitecturas gráficas,

con el fin de acelerar los procesos de cálculo y mejorar la dinámica de visualización, rende rizado y texturizado.

En el capítulo numero 1 denominado “Presentación del proyecto” se define el “Planteamiento y la justificación del problema”, donde se define la necesidad de las diferentes industrias en generar simulación de multitudes y definir lineamientos que permitan la implementación fácil y portable de modelos. Además se establecieron los objetivos planteados en este proyecto.

En el capítulo 2, titulado “Marco de Referencia”, se ilustra los usos e interesados en la simulación de multitudes basados en teoría de agentes, se establecen las características que define a los agentes y el uso de arquitecturas aceleradas en el procesamiento grafico para la visualización de la dinámica.

En el capítulo 3, se define una metodologia básica para realizar simulación de multitudes aplicado a problemas tipo plateado en este proyecto.

En el capítulo 4 titulado “Diseño e implementación” se presentan dos simulaciones como caso de estudio aplicado, siguiendo la metodologia planteada anteriormente.

En el capítulo 5 se define la especificación de arquitectura usada en el desarrollo de la simulación y visitación de este proyecto.

Posteriormente en el capítulo 6, titulado “Evaluación de desempeño” se establecen los parámetro sobre los cuales se llevara a cabo el análisis de rendimiento y las herramientas usadas para la captura de datos para el análisis e ilustración grafica de rendimiento.

En el capítulo 7 se realiza el análisis de los resultados obtenidos en el capítulo anterior.

Se definen conclusiones en el capítulo 8 basados en el proceso llevado durante todo el proyecto y posteriormente se plantean recomendaciones y posibles trabajos futuros en el capítulo 9.

1. PRESENTACIÓN DEL PROYECTO

1.1 PLANTEAMIENTO Y JUSTIFICACIÓN DEL PROBLEMA

La visualización de multitudes plantea un reto en torno al modelamiento de los datos que representan cada elemento de la multitud, el comportamiento de todos hasta la interacción de los mismos. Diferentes modelos matemáticos se han propuesto para analizarlas, debido a los usos que puede tener: recreación (como en cine, televisión, juegos de video), entendimiento de fenómenos naturales y sociales, planificación de ambientes en los que interactúan multitudes (como carreteras, puentes peatonales, estadios), entre otros.

Si bien, existen modelos matemáticos coherentes con el fenómeno a simular, la visualización plantea un reto técnico y metodológico, que hoy en día busca tratarse con el uso de arquitecturas con aceleradores, teniendo en cuenta aspectos como el realismo, la cantidad de componentes o elementos a simular, complejidad en las interacciones e interacción con las mismas. Sin embargo, existen muy pocos lineamientos que permitan la implementación fácil y portable de esos modelos aprovechando las características arquitecturales que ofrecen las computadoras con aceleradores gráficos.

El proyecto plantea visualizar multitudes a partir de modelos que permitan tener en cuenta la mayoría de aspectos técnicos, metodológicos y temáticos para la simulación de multitudes, en un caso aplicado a planificación.

1.2 OBJETIVOS

1.2.1 Objetivo General. Establecer lineamientos para la visualización de Multitudes a partir de modelos que permiten la simulación dinámica sobre arquitecturas computacionales con aceleradores gráficos.

1.2.2 Objetivos Específicos. Identificar opciones de modelos y herramientas que faciliten definir elementos, componentes y relaciones para visualizar multitudes sobre arquitecturas computacionales aceleradas y seleccionar una para ilustrar su uso con un caso práctico.

- Implementar sobre arquitecturas computacionales que tengan aceleradores gráficos, el modelamiento de multitudes aplicados a un problema que facilite ilustrar la visualización contemplando densidad, reglas y obstáculos.

- Evaluar la implementación sobre aceleradores gráficos para establecer patrones de rendimiento, de acuerdo a la complejidad, cantidad de datos y arquitectura de hardware específica usada.

- Establecer los lineamientos básicos para la visualización de Multitudes, con el uso de modelos y herramientas operables sobre arquitecturas computacionales con aceleradores gráficos.

2. MARCO DE REFERENCIA

2.1 USOS E INTERESADOS EN LA SIMULACIÓN DE MULTITUDES

Cada vez son más los interesados en el uso de aplicaciones para la simulación de multitudes, entes gubernamentales buscan estudiar fenómenos y a partir de estos estudios tomar decisiones en cuanto a la planificación urbana y la seguridad pública en donde se ve involucrada gran cantidad de personas que buscan prevenir situaciones de desastre en casos de emergencia.

En la industria cinematográfica, películas como *"El Señor de los Anillos", "Avatar" y "Guerra Mundial Z",* han logrado dar vida y realismo a diferentes escenas, usando técnicas desarrolladas en la simulaciones de multitudes con personajes complejos que interactúan entre sí y con su entorno.[1]

La industria de los videojuegos también está haciendo uso de la simulación de multitudes logrando mayor realismo e inmersión para el usuario.

"Gobiernos alrededor del mundo están tratando de utilizar este tipo de simulaciones para planificar sus ciudades, asegurando que la infraestructura sea suficiente para las necesidades de la población. Prototipos de nuevos avances tecnológicos, como las ciudades inteligentes donde coches autónomos interactúan con los peatones en un ambiente seguro son desarrollados y probados utilizando aplicaciones para la simulación de multitudes [2]."

La simulación de multitudes también es usada por investigadores en todo el mundo, usando patrones de comportamiento animal que se mueven en manada, buscando entender su comportamiento, a raíz del cambio climático, la

contaminación ambiental, desastres ambientales, desplazamiento forzado, migración natural etc. y poder tomar decisiones a partir de estas simulaciones.

Es importante revisar qué tipo de multitud se pretende simular, ya que existen factores que generan diversos patrones de comportamiento en la multitud, ya sean peces, anfibios, reptiles, aves o mamíferos, en qué lugar o espacio interactúan, cual es el contexto en el cual se encuentran inmersos y cuál es su objetivo como tal.

En cuanto a la simulación de multitudes en ambientes universitarios se han hecho algunas investigaciones basado en grabaciones de vídeo, con algunos parámetros específicos como el lugar, la hora, época del semestre, clima etc., ya que estas condiciones logran influir de gran manera en el comportamiento de la multitud.

Un estudio realizado en la Universidad de Ciudad de México. Se tomaron grabaciones de video en 6 momentos diferentes durante un día normal de clase en la universidad. En esta investigación se observó el comportamiento de 150 personas caminando en diferentes direcciones, de las cuales el 57% pertenece a un pequeño grupo. [3]

Según Moussaid [4], hasta el 70% de los peatones que ellos observaron caminaban en grupos, es decir que la cantidad de peatones que pertenecen a un grupo es mayor a las personas que caminan solas lo cual se corrobora con el estudio hecho en la universidad.

Como resultado del estudio también se pudo establecer que los grupos de dos a cuatro miembros son los más comunes, mientras que los grupos con cantidad

superior de miembros se vuelven raros; en el estudio se observó solo un grupo con más de cuatro miembros [3].

Se han realizado diferentes estudios acerca de multitudes a nivel macroscópico, es decir, estudiar el comportamiento de la multitud como un conjunto de individuos, y a nivel microscópico que le da un enfoque individual, su comportamiento y relación con el resto de individuos, una manera de simular estos individuos es usando motores gráficos basados en el modelado de agentes inteligentes, lo cual le da a la simulación, algunas facultades para que el comportamiento humano sea lo más parecido a la realidad.

En el mercado actual, se pueden conseguir diferentes motores generadores de multitudes, dada la importancia y la relevancia otorgada, así como a la necesidad de la industria, por estudiar, investigar y proponer nuevas soluciones, a partir de dichas simulaciones, aprovechando la gran oferta en cuanto a arquitecturas del alto rendimiento que facilitan y agilizan los procesos de simulación y a la facilidad de uso que tiene el software desarrollado, logrando resultados en la parte gráfica, muy semejantes a la realidad; algunos de estos simuladores están basados en la teoría de agentes inteligentes, con el fin de darle un comportamiento al personaje que lo asemeje, al comportamiento que tendría el miembro en la vida real.

Comúnmente la dinámica de comportamiento se modela con teoria de agentes, el comportamiento de la multitud es el resultado de la dinámica de cada elemento simulado en términos de agentes, cada agente está bajo algunos términos y condiciones que lo conyeban a operar de alguna manera y éste interactúa con su entorno y con otros agentes.

La simulación de multitudes basada en la teoría de modelado de agentes inteligentes busca lograr un comportamiento del individuo muy semejante al comportamiento real del ser humano en cuanto a la toma de decisiones, interacción con objetos y con otros individuos.

2.2 CONCEPTO DE AGENTE. (MODELAMIENTO DEL COMPORTAMIENTO DINÁMICO DE MULTITUDES)

“Revisando la documentación existente se puede ver que no existe una definición precisa de lo que es un agente, aunque pueden establecerse criterios que permitan distinguir lo que es un agente de lo que no.

El término agente viene del latín “agere" que significa hacer. Agente deriva del participio “agens". Expresa la capacidad de acción o de actuación de una entidad. [Mas, 2005][5]
Según [Huhns and Singh, 1998][6] el término agente se usa habitualmente para describir aquellos programas autónomos que pueden controlar las propias acciones basándose en sus percepciones de su entorno operativo.

Según [Russell and Norvig, 2004][7] un agente es cualquier cosa capaz de percibir el entorno con ayuda de sensores y actuar en ese medio con ayuda de actuadores. La secuencia de percepciones de un agente refleja el historial de percepciones que ha recibido. Un agente tomará una decisión en un momento dado dependiendo de la secuencia completa de percepciones hasta ese instante.

Una de las definiciones más citadas es la de [Wooldridge, 1997][8]: \Un agente es un sistema informático situado en un entorno y que es capaz de realizar acciones de forma autónoma para conseguir sus objetivos de diseño".

Aunque esta definición identifica algunas de las características de un agente, como la autonomía, no permite distinguirlo claramente de un sistema distribuido convencional. [Mas, 2005][5] [Wooldridge and Jennings, 1995][9] destacan que los agentes informáticos típicamente tienen las siguientes propiedades:

- Autonomía. Los agentes operan sin que otros tengan control directo de sus acciones y de su estado interno.

- Habilidad social. Los agentes interactúan con otros agentes.

- Reactividad. Los agentes pueden percibir su entorno y pueden responder al mismo.

- Proactividad. Los agentes también son capaces de tomar la iniciativa, involucrándose en un comportamiento dirigido a un objetivo propio.

Aunque no necesariamente deben tenerlas todas. Además, a los agentes se les suele atribuir un cierto grado de intencionalidad.

Sin embargo, estas características no son muy útiles para diseñar un agente, y además un agente no tiene por qué poseer necesariamente todas estas características. Una forma más útil de describir agentes es decir que tienen las siguientes características: [Gilbert, 2008] [10]

- Percepción. Pueden percibir el entorno, posiblemente incluyendo la presencia de otros agentes de su alrededor. En términos de programación, esto significa que los agentes tienen los medios para determinar qué objetos y agentes están localizados en su vecindad.

- Representación. Tienen un conjunto de comportamientos que son capaces de representar. Con frecuencia, dentro se incluyen:

- Movimiento. Se pueden mover dentro de un espacio (ambiente).

- Comunicación. Pueden enviar mensajes y recibir mensajes de otros agentes.

- Acción. Pueden interactuar con el entorno, por ejemplo, recogiendo “comida".

- Memoria. Tienen una memoria, que almacena sus percepciones de anteriores estados y acciones.

- Políticas. Tienen un conjunto de reglas, heurísticas, o estrategias que determinan, dada su situación actual y su historia, qué comportamientos no deben llevar a cabo.” [11].

2.3 USO DE GPU’S EN PROCESAMIENTO GRÁFICO.

Gracias a las GPU (Graphics Processing Unit), unidad de procesamiento gráfico por sus siglas en inglés, se pueden manejar grandes cantidades de agentes,

realizando la simulación en el procesador gráfico, teniendo el cuidado de reducir la cantidad de información que se comunica entre la CPU y GPU, ya que ésta será enviada por cada cuadro de la simulación [3].

"A mediados de la década de 1990, la demanda de gráficos 3D por parte de los usuarios aumentó vertiginosamente a partir de la aparición de juegos inmersitos en primera persona, como Doom, Duke Nukem 3D o Quake, que acercaban al sector de los videojuegos para ordenadores personales entornos 3D cada vez más realistas. Al mismo tiempo, empresas como Nvidia, ATI Technologies y 3dfx Interactive empezaron a comercializar aceleradores gráficos que eran suficientemente económicos para los mercados de gran consumo. Estos primeros desarrollos representaron el principio de una nueva era de gráficos 3D que ha llevado a una constante progresión en las prestaciones y capacidad computacional del procesamiento gráfico". [12]

Comercialmente, Nvidia ofrece diferentes productos, divididos en tres familias principalmente:

- GeForce®: orientada al gran mercado de consumo multimedia (videojuegos, edición de vídeo, fotografía digital, entre otros).

- Quadro®: orientada a soluciones profesionales que requieren modelos 3D, como los sectores de la ingeniería o la arquitectura.

- Tesla®: orientada a la computación de altas prestaciones, como el procesamiento de información sísmica, simulaciones de bioquímica, modelos meteorológicos y de cambio climático, computación financiera o análisis de datos.

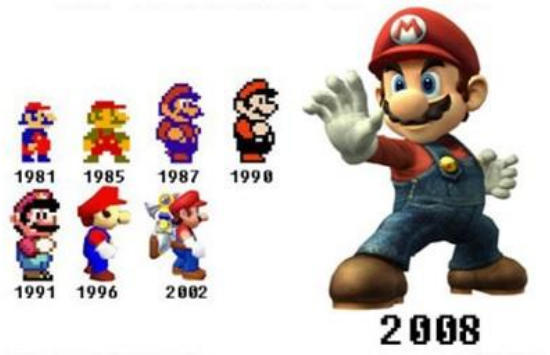
En el proceso de rende rizado (proceso de transformación del modelo 3D hasta conseguir la imagen representada en la pantalla), cada instrucción a ejecutar debe realizarse numerosas veces, para cada vértice o píxel, sin dependencia entre ellos, lo cual permite directamente paralelizar entre ellos. Este tipo de procesado se conoce como SIMD (Single Instruction Multiple Data), es decir, ejecutar una instrucción a un conjunto más o menos grande. [13]

Desde el 2000 hasta la actualidad, con la mejora de OpenGL y DirextX, las GPUs han ido añadiendo funcionalidad a sus capacidades. Ahora cada píxel puede procesarse con un pequeño programa e incluso pintarlo con texturas, así como cada vértice puede modificarse con pequeñas tareas antes de proyectarse sobre la pantalla. La Nvidia GeForce 3 (NV20) fue la primera GPU en introducir funcionalidad programable. La ATI Radeon 9700 fue la primera en introducir soporte para bucles y operaciones en coma flotante complejas. [13]

Coma flotante: En el procesado de vértices, existen transformaciones entre el modelo 3D y su proyección sobre el plano de visión. Recordando trigonometría esto implica multiplicación por senos y cosenos, los cuales trabajan con muchos decimales siendo necesario el uso de este tipo de operaciones. En el caso de píxeles ocurre igual con los efectos de luminosidad. [13]

En el caso de la simulación de multitudes en 3D, se realiza con modelos conformados por figuras geométricas, estas figuras geométricas a mayor cantidad de vértices generan mayor calidad de definición de los modelos y mayor realismo del modelo. [13]

Imagen 1. Aumento de vertices en la Imagen con el tiempo.



[3]

Actualmente se pueden conseguir en el mercado diferentes software que permiten realizar simulación de multitudes con gran cantidad de individuos y reglas, basados en el modelado de agentes; herramientas tales como Massmotion[4], Unity[5], Massive[6], Flamegpu[7], Golaem Crowd[8], Miarmy[9] entre otros, la mayoría requieren de arquitecturas gráficas para el procesamiento de gran cantidad de datos que se genera en dicha simulación, ya sea en la simulación de la gran cantidad de agentes que conforman la multitud o en el rende rizado.

Algunos de este software buscan facilitar el proceso de creación de simulación de multitudes evitando que el usuario ingrese directamente código, ofreciendo reglas para los agentes preestablecidos, las cuales generan el comportamiento e interacción del agente con el entorno.

[3] Imagen tomada de: http://www.iuma.ulpgc.es/~nunez/clases-micros-para-com/mpc1011-trabajos/mpc1011-Graphics%20Processing%20Units%20Omar%20Espino%20Santana.pdf
[4] http://www.oasys-software.com/products/engineering/massmotion.html
[5] https://unity3d.com/es
[6] http://www.massivesoftware.com/
[7] http://www.flamegpu.com/
[8] http://golaem.com/
[9] http://www.basefount.com/miarmy.html

3. METODOLOGÍA PARA LA SIMULACIÓN DE MULTITUDES

Identificar el problema para así poder definir el tipo de simulación que se desea realizar, el entorno con el cual van a interactuar los agentes, densidades de agentes, causas y consecuencias del fenómeno a simular, reglas, obstáculos y condiciones, el nivel de realismo que se pretende generar en la escena con geometrías, textura, materiales y dinamismo de los agentes .

Una vez identificado el problema, se procede a establecer las necesidades de software, licenciamiento, limitaciones para modelar, características, etc. con el fin de identificar la mejor herramienta para simular el caso particular.

Basados en las herramientas de software seleccionadas, se definen los requerimientos de arquitectura computacional para realizar dicha simulación.

Si la simulación de multitudes busca tener una representación gráfica con modelos complejos, grandes densidades de agentes y alto nivel de rendimiento computacional, es importante tener en cuenta, hacer uso de arquitectura gráfica que agilice el proceso de cálculo.

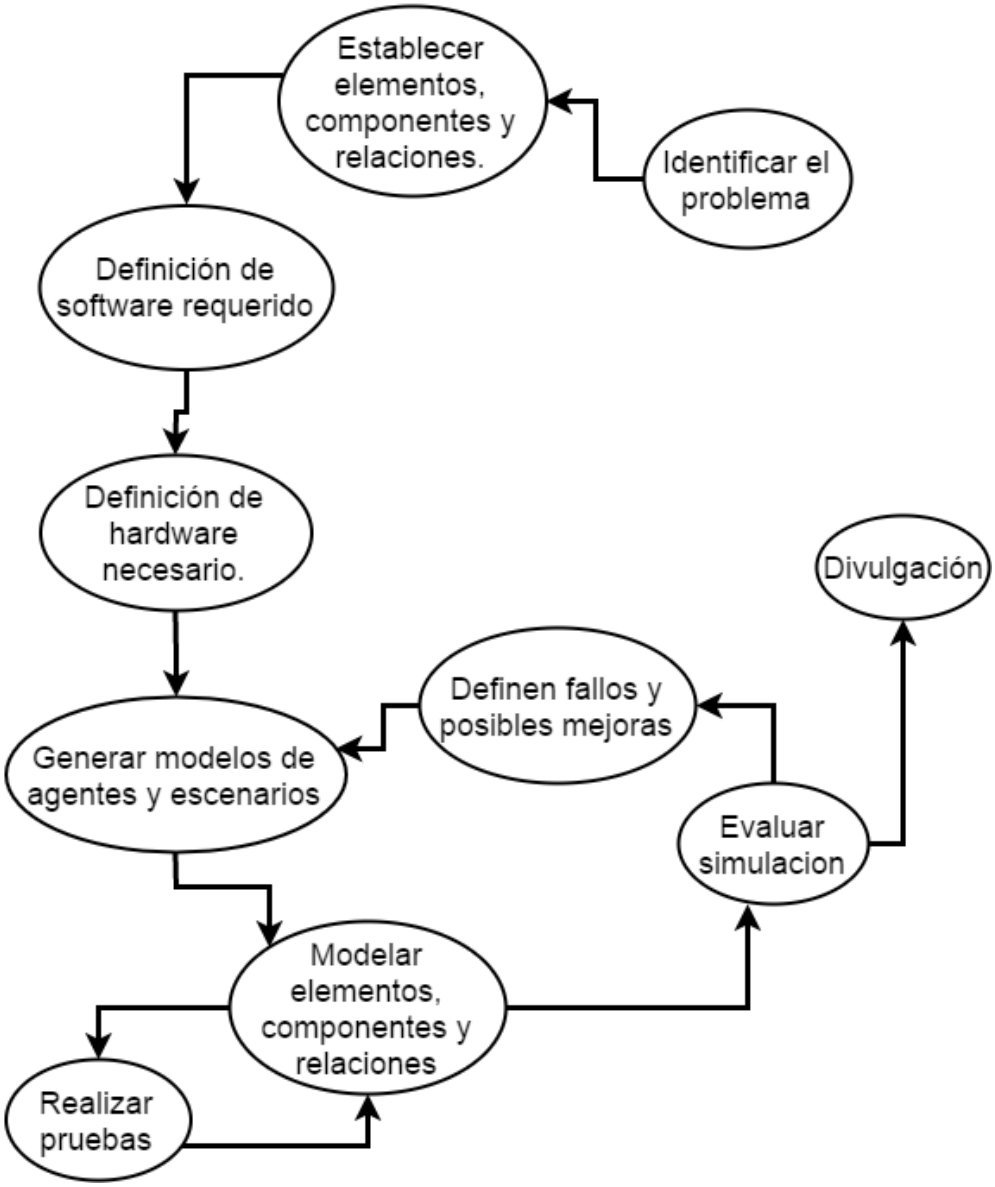
Se procede generar los modelos gráficos de agentes y el escenario sobre el cual se realizará la simulación; Se modela la dinámica de comportamiento de la multitud teniendo en cuenta acciones, reglas, obstáculos y efectos.

Se realizan pruebas, cambios y mejoras al modelo y finalmente se evalúa la dinámica del comportamiento de los agentes en la simulación.

En la imagen 2 ilustrada a continuación, se puede observar el flujo de trabajo.

3.1 DIAGRAMA PARA LA SIMULACIÓN DE MULTITUDES

Imagen 2. Flujo de trabajo.



Como se puede observar en el diagrama anterior, inicialmente se identifica el problema, basado en este, se establecen elementos componente y relaciones necesarias para recrear la dinámica, y así escoger el software y arquitectura necesaria para modelar el caso particular planteado.

Posteriormente se generan los modelos de agentes y escenarios. Se definen los componente y relaciones que darán lugar al comportamiento propio de la multitud, se realizaran pruebas con el fin de ajustar el modelo de comportamiento y posteriormente se evaluara la simulación. En caso de encontrar posibles fallos o mejoras se volverá a evaluar modelos, escenarios elementos y relaciones con el fin de solucionar los fallos o realizar mejoras siguiendo el ciclo indicado, evaluando la simulación hasta dar parte de conformidad para posteriormente divulgarlo en caso de ser necesario. En este punto se da por terminado el flujo.

4. DISEÑO E IMPLEMENTACIÓN

4.1 PRIMER CASO DE ESTUDIO

4.1.1 Identificación del Problema. Se tomó como caso de estudio la dinámica de comportamiento de la multitud generada en la portería principal de la Universidad Industrial de Santander teniendo en cuenta una gran afluencia de personas entrando y saliendo de la universidad como sucede en horas de intercambio de clase un día de normalidad académica.

En horas de intercambio de clase hay mayor flujo de personas en la portería, ya que la gran mayoría de estudiantes reciben clase en el transcurso del día, entendiendo que no hay jornada única continua; la actividad académica inicia a las 6 de la mañana y finaliza a las 10 de la noche, por lo general las horas de intercambio de clases se da en horas pares, ya que la mayoría están establecidas en bloques de dos horas por asignatura y esto hace que al finalizar los bloques de dos horas de clase se cree un mayor tráfico de personas en el campus universitario.

Hay que resaltar la importancia de definir los obstáculos observados en dicho entorno representado en la siguiente imagen, ya que estos juegan un papel fundamental en la toma de decisiones para cada uno de los individuos.

Imagen 3. Entrada principal de la Universidad.

Se establecieron como principales obstáculos:

- Las barandas separadoras.
- La caseta para los celadores.
- Las columnas de concreto.
- Las barandas metálicas.

Éstos son algunos elementos que conforman la portería principal de la Universidad.

En la entrada principal de la Universidad Industrial de Santander se implementó un sistema de ingreso con separadores de personas fijos, con el fin de dar mayor orden al tránsito peatonal generado por la comunidad universitaria. Se estableció que las cuatro primeras filas conformadas por personas entrando a la universidad de derecha a izquierda se tomarían como filas de ingreso , y las filas siguientes, como filas de egreso de la misma, esto ocasiona que se genere un efecto embudo de personas reunidas en esta zona, que evitan tropezar con las barandas separadoras, y posteriormente buscan dirigirse en cualquier dirección siguiendo su trayecto dentro y fuera de la universidad.

Como caso complementario se planteó simular una eventual emergencia por atentado con artefacto explosivo dentro del claustro universitario en la zona próxima a la portería principal y evaluar el posible comportamiento de los individuos, establecido de manera intuitiva en tal eventualidad, basados en los antecedentes registrados en el claustro.

Se determinó que los individuos que se encuentren en un radio muy próximo al lugar de la explosión, quedarían heridos tendidos en el piso, los individuos ubicados a una distancia media al lugar de la explosión correrían en direcciones contrarias al lugar para alejarse del punto de explosión hasta llegar a una distancia donde se sientan a salvo, algunas personas que están lejos llegaran al sitio y conformarán un círculo alrededor del lugar de la explosión para observar lo sucedido.

En este punto se da por terminada la simulación del caso en estudio.
Este caso de estudio simulado puede ayudar a la toma de decisiones y así establecer de manera más precisa por ejemplo la cantidad de celadores requeridos debido a la magnitud del flujo de personas transitando por esta partería, definir los planes de evacuación en caso de emergencia, analizar el tráfico de personas en determinadas horas y condiciones entre otras.

En el siguiente capítulo se definen los requerimientos necesarios para la realización de la simulación teniendo en cuenta los parámetros establecidos anteriormente.

4.1.2 Definición de Requerimientos. Una vez establecidos los componentes necesarios para la realización de la simulación, se concluyó que el desarrollo del proyecto se realizaría con Maya Autodesk 2014 y Miarmy en su versión 4.2 como motor para la simulación de multitudes , ya que es un software que cuenta con más de diez años de trascendencia en el mercado, ha sido usado por grandes corporaciones como Dexter Digital, Disney Televisión Animation, Square Enix,Prasad EFX, CGCG Inc., Digital Frontiers[10], que sirven de evidencia de la calidad y grandes resultados que se pueden obtener con esta herramienta.

Miarmy es un plugin que debe ser instalado sobre Maya de Autodesk, el cual es un software de animación, modelado, simulación en 3d, cuenta con una amplia gamas de herramientas creativas[11], compatible con varios software de diseño 3d usados en la industria como Blender, Unity, Houdini etc, y trabaja con varios motores de rende rizado brindando excelente calidad gráfica y alto nivel de realismo.

Maya Autodesk cuenta con una versión de total funcionamiento para estudiante con licencia libre por tres años[12], mientras que Miarmy permite rende rizar máximo 100 agentes como única restricción en su versión libre[13], vienen con un pack de personajes listos para su uso que se pueden descargar de su página

[10] https://en.wikipedia.org/wiki/Miarmy

[11] http://www.autodesk.es/products/maya/overview

[12] http://www.autodesk.com/education/free-software/maya

[13] https://basefount.atlassian.net/wiki/display/MDE/Windows

oficial[14], la versión de pago permite generar el render de más de 100 agentes y tiene un costo de 2.650 USD por un año con actualizaciones y soporte[15].

4.1.2.1 Requisitos Mínimos.

4.1.2.1.1 Requisitos de Maya Autodesk16:

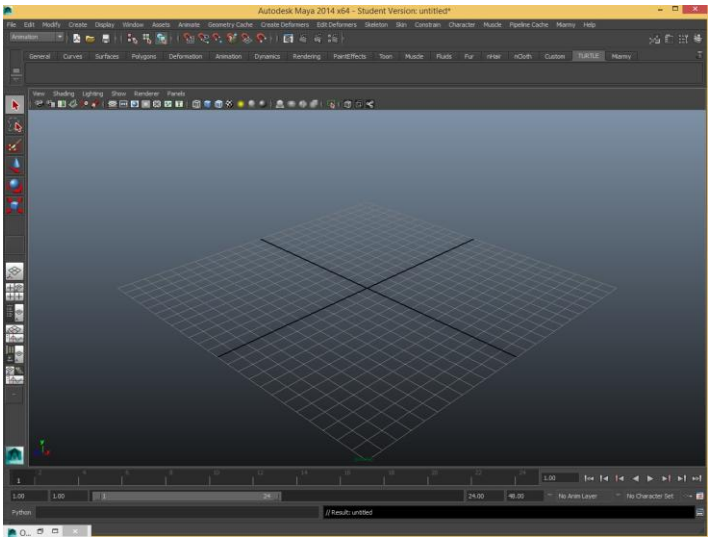
- Windows® 8 Professional edition, Windows® 7 Professional edition, Apple® Mac OS® X 10.7.x or 10.8.x, Red Hat® Enterprise Linux® 6.2 W S, or Fedora™ 14 Linux operating system
- 64-bit Intel or AMD multi-core processor
- 4 GB of RAM minimum (8 GB recommended)
- 2 GB of free disk space for installation
- Microsoft® Internet Explorer®, Apple® Safari®, or Mozilla® Firefox® web browser
- 3-button mouse.

[14] http://www.basefount.com/agents-library.html

[15] http://www.basefount.com/buy-now.html

[16] http://knowledge.autodesk.com/support/maya/troubleshooting/caas/sfdcarticles/sfdcarticles/System-requirements-for-Autodesk-Maya-2014.html

Imagen 4. Espacio de trabajo Maya.



4.1.2.1.2 Requisitos de Miarmy [17]:

SISTEMA OPERATIVO:

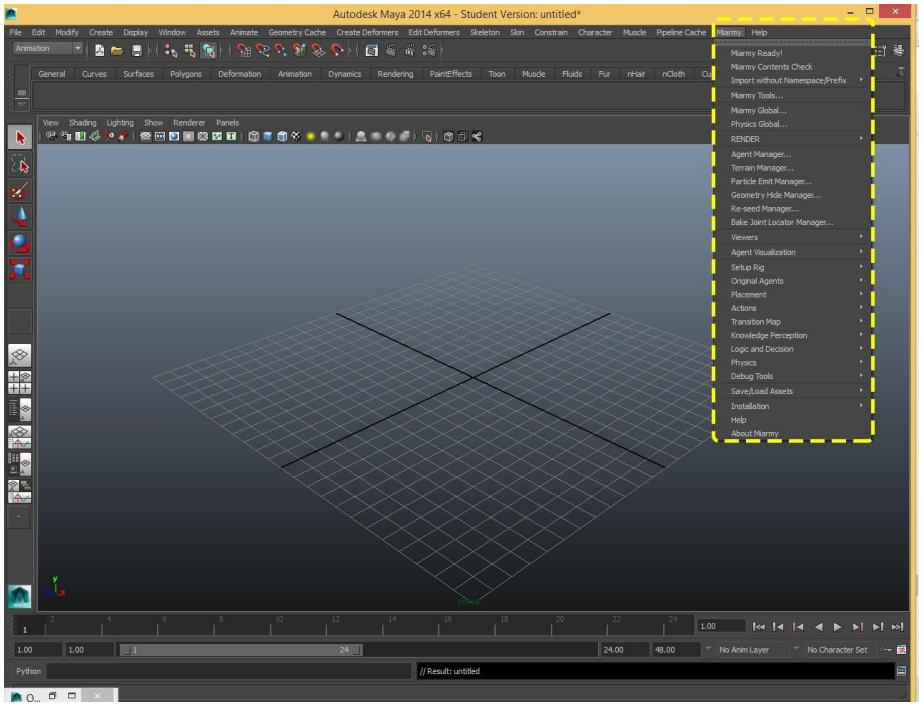
- Windows 7, (XP SP3)
- Red hat Linux 5.5 +
- Apple Snow Leopard +

PLATAFORMA:

- Autodesk Maya 2013
- Autodesk Maya 2014
- Autodesk Maya 2015
- Autodesk Maya 2016

[17] https://basefount.atlassian.net/wiki/display/MDE/Windows

Imagen 4. Menú principal Miarmy.



4.1.3 Modelado del Escenario. Se modelo en Maya Autodesk un plano que asemeja la entrada principal de la Universidad Industrial de Santander sobre el cual se realizará la simulación de multitudes teniendo en cuenta los principales obstáculos con los cuales interactúa la multitud.

Imagen 5. Modelo del primer escenario.

Con el fin de darle un aspecto más real al escenario, se agregaron materiales, texturas e iluminación a las geometrías que conforman el escenario.

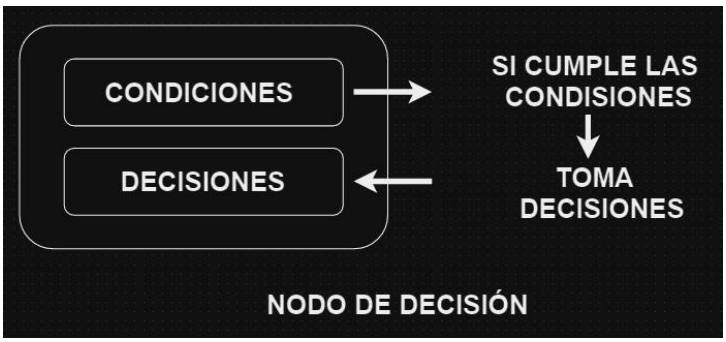
A continuación se explica el proceso de modelado planteado para este caso particular usando Miarmy.

4.1.4 Modelado y Simulación con Miarmy en Maya Autodesk.

Miarmy utiliza la lógica de control basado el lenguaje humano, que trata de asemejar el pensamiento humano ya que las personas constantemente están tomando decisiones a partir de lo que les está sucediendo, es decir si A sucede, entonces B.

En Miarmy las “condiciones” están denominadas como “sentencias de entrada” y las “decisiones” como “salida de decisiones” y son evaluadas con lógica difusa de forma automática.

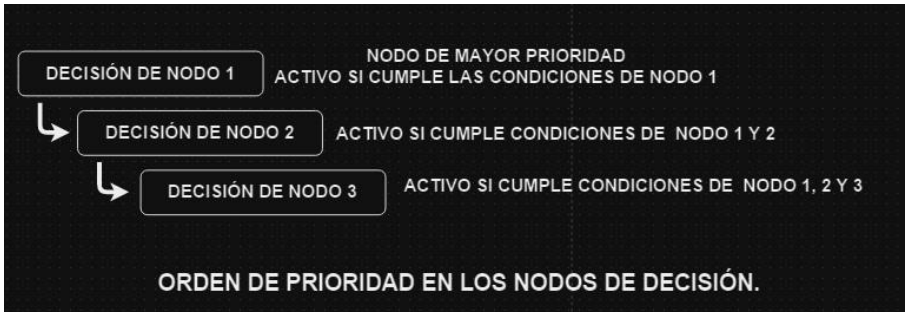
Imagen 7. Nodo decisión.



La decisión activa es el resultado de la prueba de la condición de nodo de decisión.

Para la toma de decisiones también se toma en cuenta la prioridad que tenga cada uno de los nodos de decisión, ya que estos cuentan con una jerarquía representada de la siguiente manera.

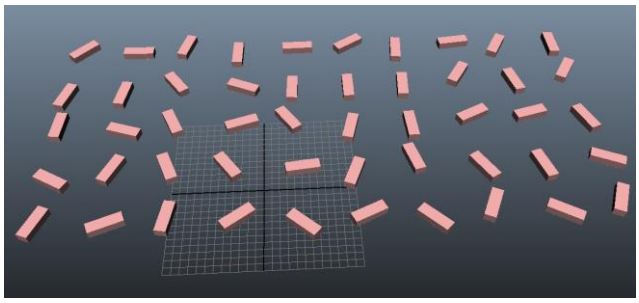
Imagen 8. Prioridad del nodo decisión.



Con Miarmy se pueden crear multitudes de diferentes modelos, dichos modelos deben contar con una estructura ósea que se diseña con la ayuda de la

herramienta “Joint Tool” de Maya Autodesk, una vez ésta figura ósea se integra en Miarmy y se crea el placement node[18], Miarmy genera un prisma rectangular para cada uno de los Skin, el cual se puede modificar en sus dimensiones como se muestra en la siguiente imagen.

Imagen 6. Formación de agentes.

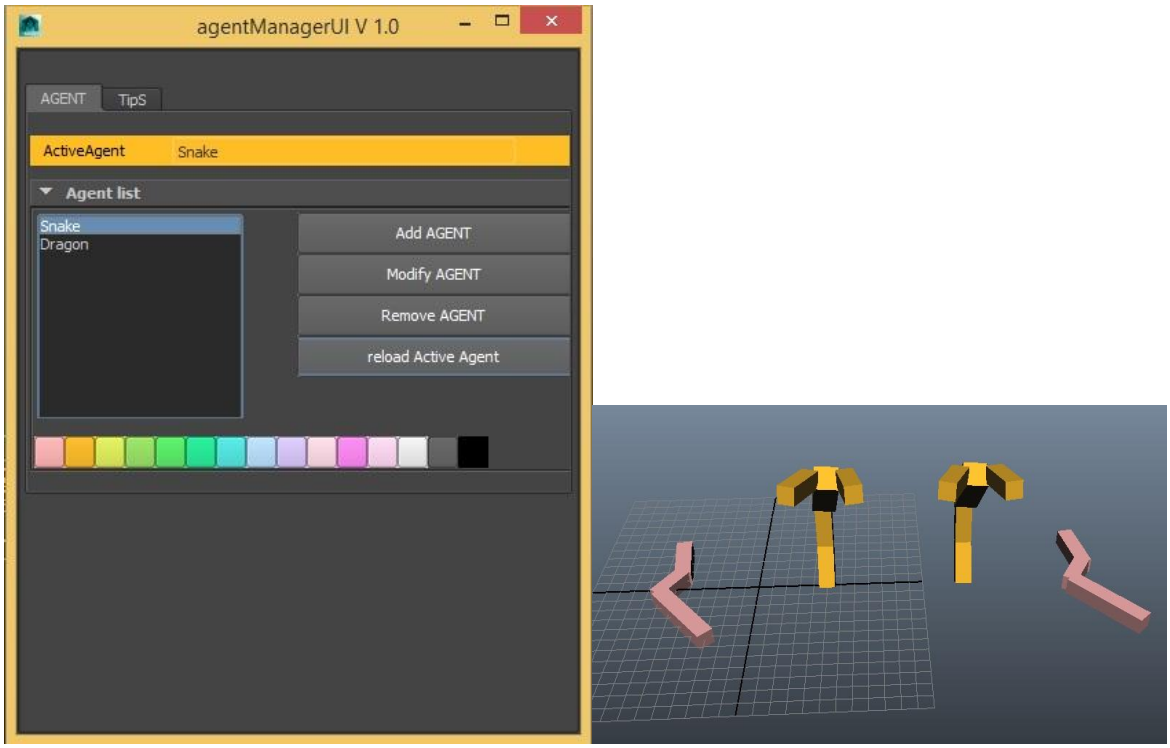


En Miarmy, se establecen reglas para cada agente que se desea modelar, se puede multiplicar la cantidad de veces necesaria, darle ubicación y orientación inicial por separado.

El “AgenteManager” lista los agentes generados en el motor de Miarmy, permite modificar algunos parámetros e indicar el color característico para cada tipo de agente.

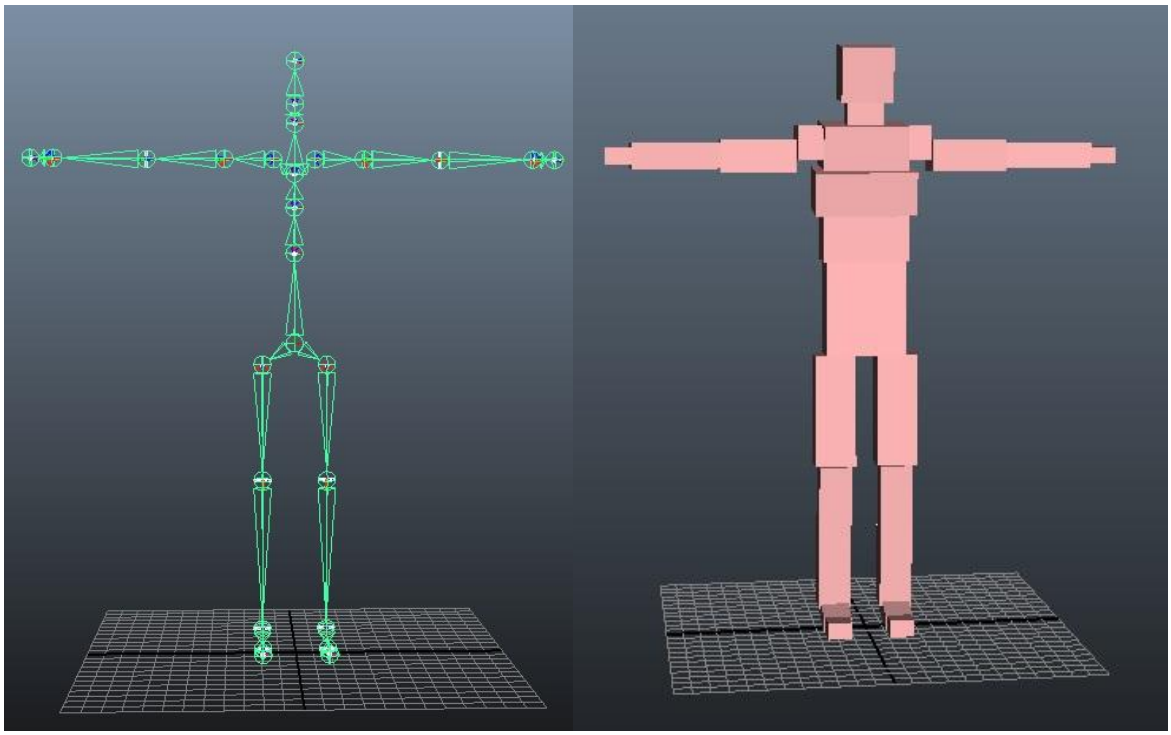
[18] Punto de referencia que será ubicado sobre el espacio con dirección, sentido.

Imagen 7. Menú para administrar agentes.



El desarrollo de esta simulación se realizó con modelos que están compuestos por "huesos y articulaciones" que asemejan una figura ósea general del cuerpo humano como se muestra en la siguiente imagen.

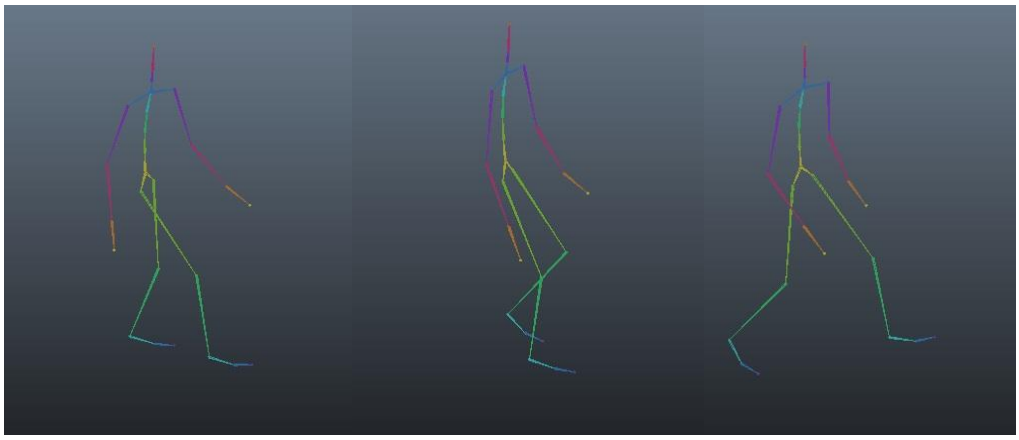
Imagen 8. Agente humano.



Una vez creada la estructura ósea guardando la proporción de escala con respecto al plano y los objetos con los cuales los agentes interactúan, se realiza la animación de los movimientos que se desean modelar, esto con el fin de crear "acciones", por ejemplo.

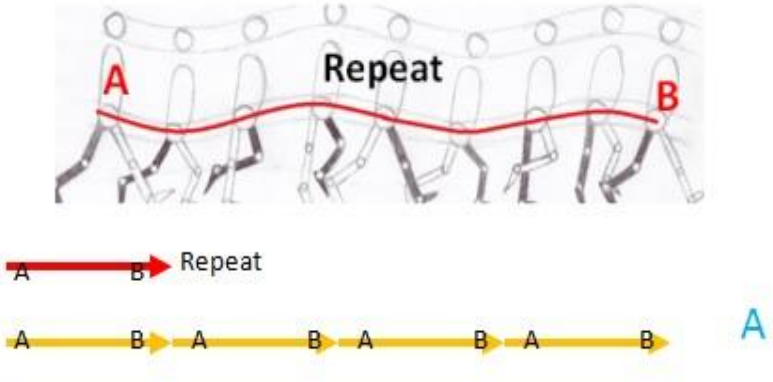
Para lograr que el agente camine indefinidamente, se debe generar la animación que caracteriza dicho movimiento. Esto se logra cambiando la ubicación de cada uno de los huesos del modelo, asemejando la acción de caminar con respecto a la línea de tiempo de Maya hasta cumplir un ciclo completo.

Imagen 9. Secuencia de animación.



Una vez generada la animación sobre el esqueleto completo y cumplida la secuencia desde A hasta B, se crea la acción propiamente desde Miarmy generando un ciclo repetitivo de A hasta B y volver a A, como se ilustra a continuación.

Imagen 10. Ciclo de animación.

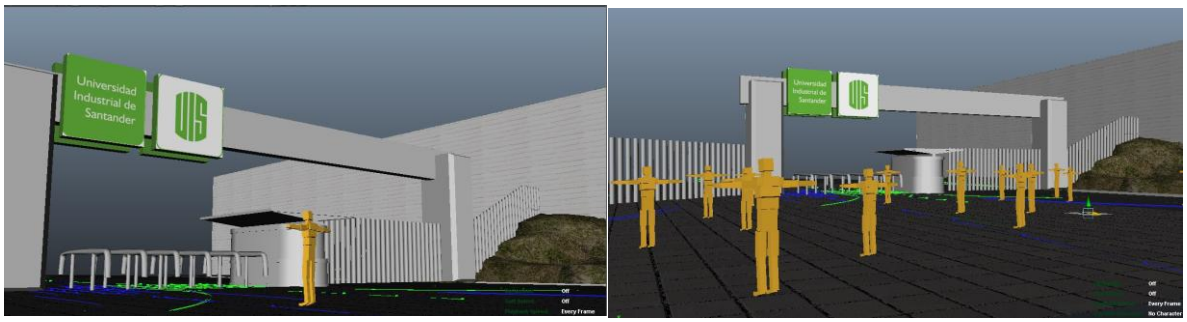


De esta manera se obtiene y guarda la acción de caminar identificada con el nombre de (walkA) que se usará en la simulación del caso aplicado planteado anteriormente, y ésta se podrán aplicar en Miarmy al modelo sobre el cual se realizó sin importar el tipo de agente.

Se creó el primer agente llamado “Estudiante 1” con la geometría del esqueleto humano sobre el cual se trabajó, se le asignó la acción llamada “walkA” y fue ubicado sobre el plano entrando a la universidad como lo ilustra la siguiente imagen.

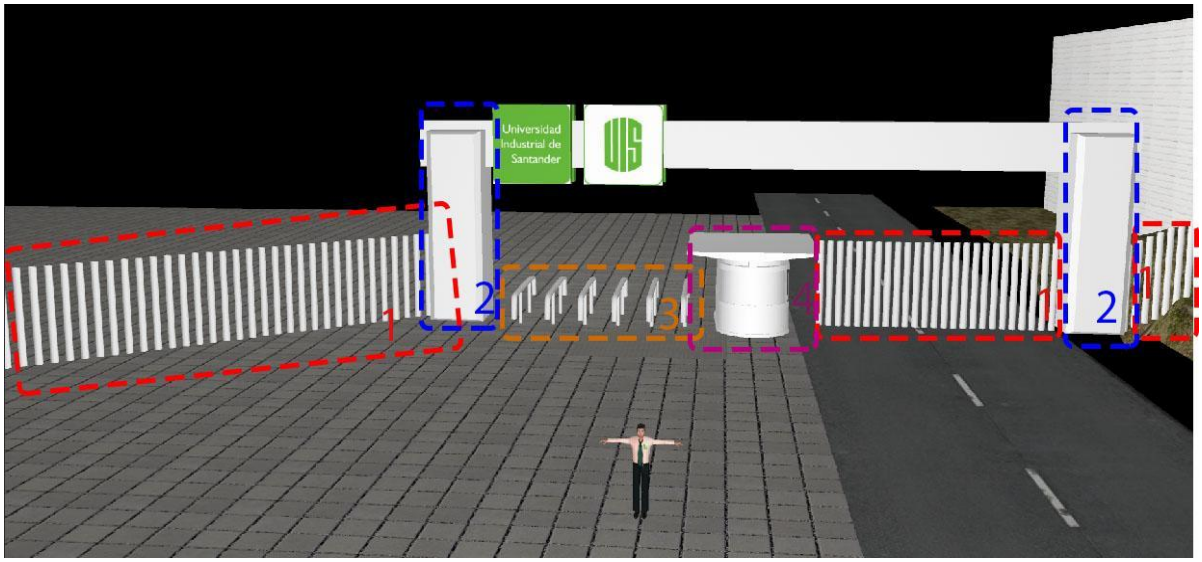
Una vez creado el “Agente 1” se puede variar la cantidad de agentes que se desee en la simulación, los cuales interactúan bajo las mismas reglas con el entorno y con otros agentes.

Imagen 11. Agentes sobre el plano.



En la siguiente imagen se pueden observar los principales obstáculos con los cuales interactúa de alguna manera la multitud y el terreno sobre el cual se van a desplazar cuando desean ingresar o abandonar la universidad por la portería principal.

Imagen 12. Representación gráfica entrada principal Universidad Industrial de Santander.



La zona encerrada con líneas punteadas de color rojo (1) son barandas metálicas que impiden el ingreso al interior del campus.

La zona encerrada con líneas punteadas de color azul (2) son edificaciones que impiden el ingreso al interior de campus.

La zona encerrada con líneas punteadas de color naranja (3) son separadores metálico los cuales permiten el ingreso al campus, y hacen efecto embudo con la multitud, generando filas de personas.

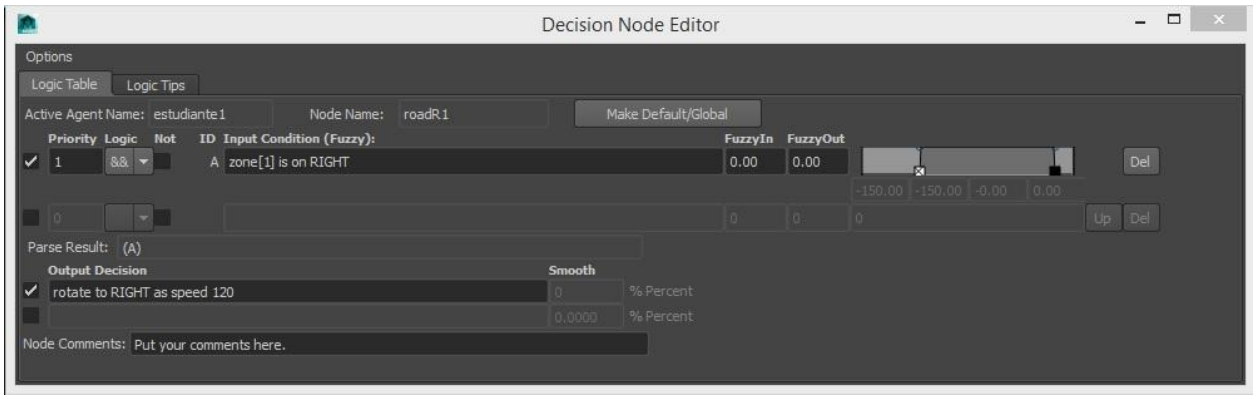
La zona encerrada con líneas punteadas de color fucsia (4) es la caseta para personal de seguridad privada de la universidad.

El objetivo inicial es lograr que los agentes que van a ingresar a la universidad, lo hagan por el lugar indicado, es decir por las cuatro filas de la derecha; para esto,

se creó una “zona 1” a la cual todos los agentes “Estudiante 1” deben dirigirse cumpliendo dos reglas especiales.

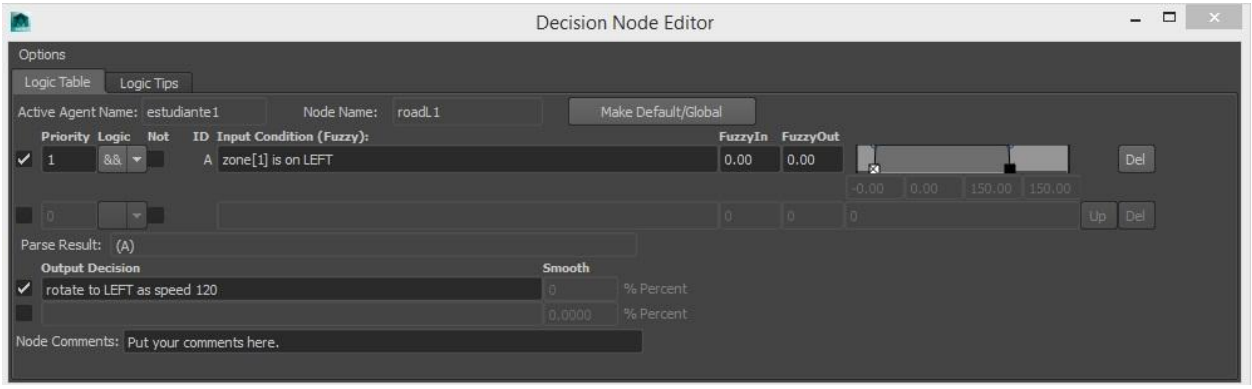
- Si la zona 1 está a la derecha del agente en estudio, éste debe girar hacia la derecha con una velocidad indicada buscando llegar a la zona objetivo (zona 1) como se ilustra en la siguiente imagen.

Imagen 13. Nodo decisión, rotar a la derech.



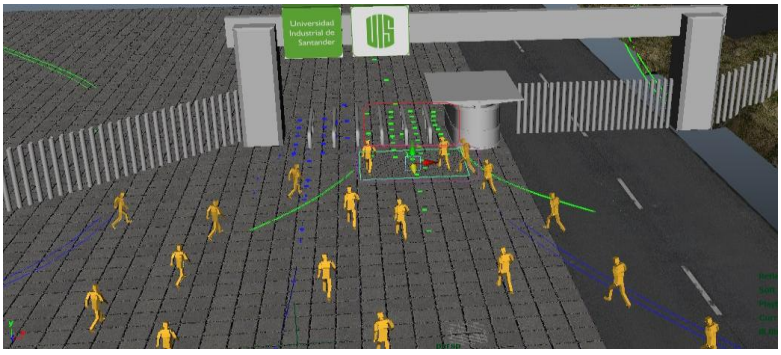
- Si la zona 1 está a la izquierda del agente en estudio, éste debe girar hacia la izquierda con una velocidad indicada buscando llegar a la zona objetivo (zona 1) como se ilustra en la siguiente imagen.

Imagen 14. Nodo decisión, rotar a la izquierda.



La zona señalada con el recuadro de color púrpura indica cual es la "zona 1" nombrada anteriormente y la zona señalada con el recuadro de color rojo, el camino por el cual los agentes debe ingresar, entendiendo cuales son los obstáculos que se deben evitar.

Imagen 15. Identificación zona 1.

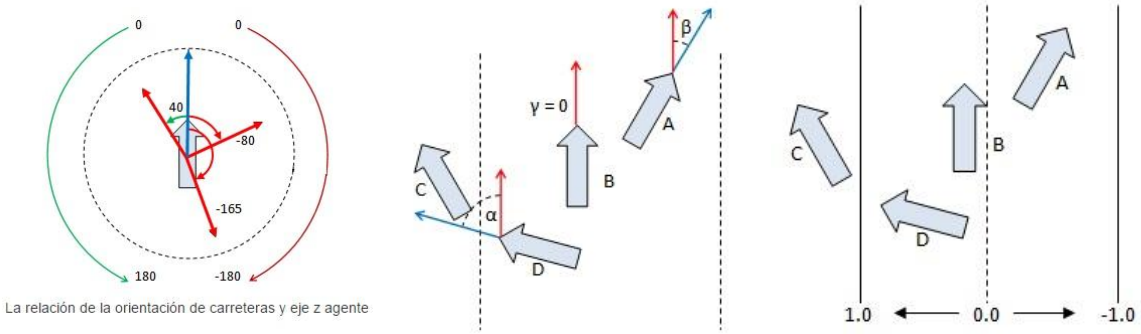


Una vez el agente "Estudiante 1" está ubicado en la "zona 1", percibirá en su radio próximo cuatro caminos identificados con el ID 2, los cuales le indican al agente seguir un camino estos por donde debe dirigirse para ingresar al claustro universitario.

Cada “camino”, se crea a partir de un segmento de recta con la función “Road” de Miarmy teniendo en cuenta el sentido en el cual se crea el segmento de recta ya que éste influirá al momento de definir las reglas para los agentes.

Para los agentes ubicados en la carretera, la frase devolverá el ángulo entre el eje Z (flecha azul) de agente y la dirección del camino (flecha roja).

Imagen 16. Orientación sobre road.



La relación de la orientación de carreteras y eje z agente

En el ejemplo ilustrado en la imagen anterior las flechas de color gris representan los agentes A,B,C y D, para el caso del agente C, la frase devolverá falso ya éste se encuentra fuera del camino, para el caso de los agentes A, B y C, devolverá verdadero, porque éstos están dentro del camino. El ángulo conformado por el vector Z (azul) que indica la dirección del agente y el vector rojo la dirección del camino, determina si el camino está ubicado a la derecha del agente en caso de ser positivo; en caso contrario si el ángulo es negativo, el camino está a la izquierda[19].

Para Miarmy no es necesario recordar tanta teoría ya que esto se puede manejar con lenguaje humano como se ilustra en la siguiente imagen, es decir, indicando

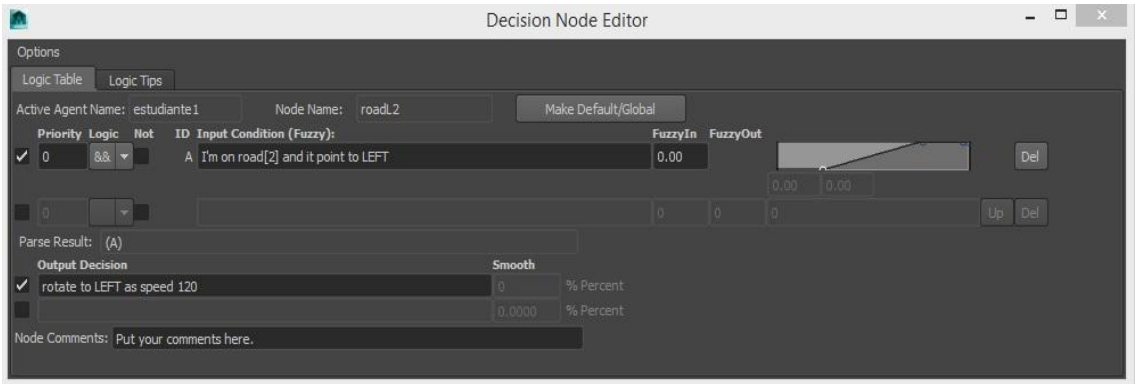
[19] https://basefount.atlassian.net/wiki/display/MDE/Road

las reglas en el nodo decisión dependiendo si está a la derecha o izquierda entendiendo el concepto principal.

Se generaron cada uno de los caminos y establecieron reglas para los agentes “Estudiante 1” que se encuentren en el camino identificado con el ID 2, buscando conformar filas de agentes simulando el comportamiento real de la multitud de la siguiente manera:

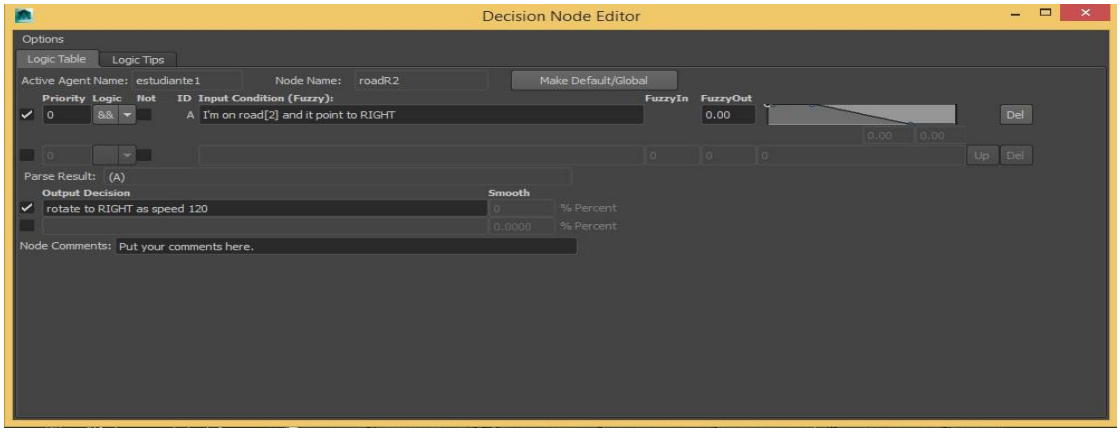
- Si el agente está sobre el camino de ID 2 y el vector dirección del camino está a la izquierda, entonces el agente rotará a la izquierda a una velocidad de definida 120.

Imagen 17. Nodo decisión rotar a la izquierda cuando este en el road 2.



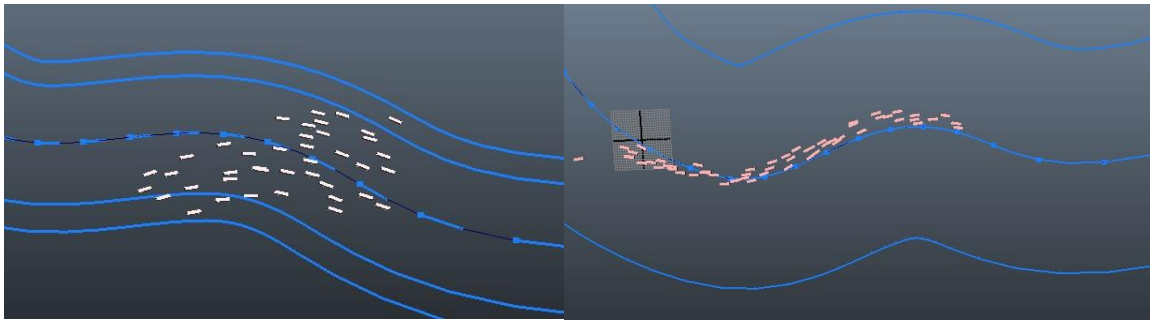
- Si el agente está sobre el camino de ID 2 y el vector dirección del camino está a la derecha, entonces el agente rotará a la derecha a una velocidad de 120.

Imagen 18. Nodo decisión rotar a la derecha cuando este en el road 2.



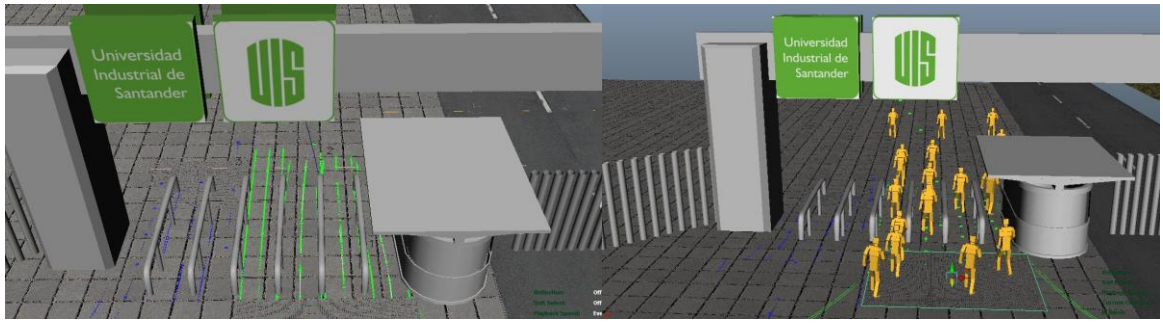
Se establece además, que los agentes deben dirigirse todos justo a la mitad del camino conformando así una fila de agentes como se indica en la siguiente imagen.

Imagen 19. Conformar fila de agentes.



A modo de resultado se puede observar que los agentes en estudio llegan a la zona 1, ahí, los agentes reconocen cuatro caminos identificados con el ID 2 de color verde en su radio, y siguiendo las reglas indicadas anteriormente, conformarán las filas correspondientes simulando el correcto ingreso al claustro universitario como se ilustra a continuación.

Imagen 20. Agentes ingresando al claustro.



Una vez los agentes terminan de recorrer dichos caminos, éstos reconocen en su radio próximo un camino verde identificado con el ID 2 con forma de embudo que indica a los agentes seguir el camino en el mismo sentido y se desplacen en la diferentes direcciones.

Simulación Estudiante 2

Bajo el modelo de esqueleto óseo similar al del ser humano que se usó en el agente “Estudiante 1”, se creó un segundo agente llamado “Estudiante 2”, el cual se ubicó en dirección contraria al claustro universitario, ya que éstos agentes simulan el comportamiento de las personas que están saliendo de la universidad.

Los agentes “Estudiante 2” deben acercarse primero a la zona previa de salida en la portería, por la cual es permitida la salida de personas, para esto se creó una zona denominada “zona 2”.

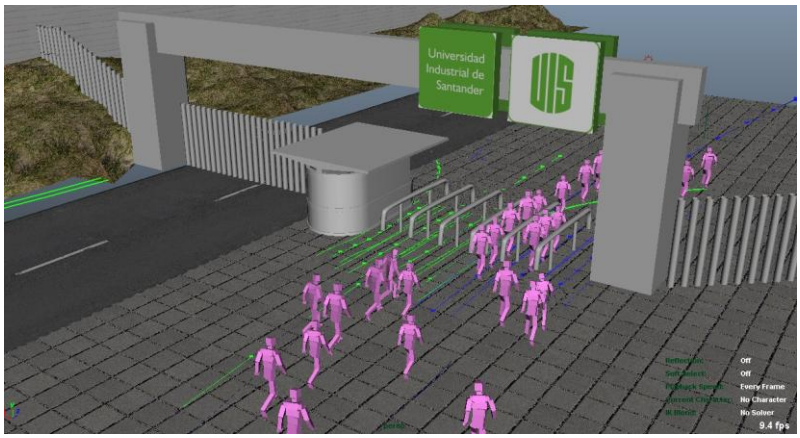
Para indicar a los agentes como llegar a dicha zona se establecieron las siguientes reglas de comportamiento.

- Si la zona 2 está a la izquierda del agente en estudio, éste debe girar hacia la izquierda con una velocidad de 120 buscando llegar a la zona objetivo (zona 2).

- Si la zona 2 está a la derecha del agente en estudio, éste debe girar hacia la derecha con una velocidad de 120 buscando llegar a la zona objetivo (zona 2).

Cuando el agente en estudio está ubicado en la zona 2, éste reconoce cuatro caminos de color azul en su radio, identificados con el ID 3. Siguiendo las reglas indicadas conformarán una fila de agentes simulando el correcto flujo de personas al abandonar el claustro universitario como se ilustra con la siguiente imagen.

Imagen 21. Agentes saliendo del claustro.



Una vez los agentes terminan de recorrer dichos caminos, éstos reconocen en su radio próximo un camino verde identificado con el ID 3 con forma de embudo que indica a los agentes seguir el camino en el mismo sentido y se desplacen en la diferentes direcciones.

Posteriormente se hicieron pruebas y mejoras realizando la simulación de los diferentes agentes interactuando al tiempo en un mismo plano para observar su comportamiento como se muestra en la siguiente imagen.

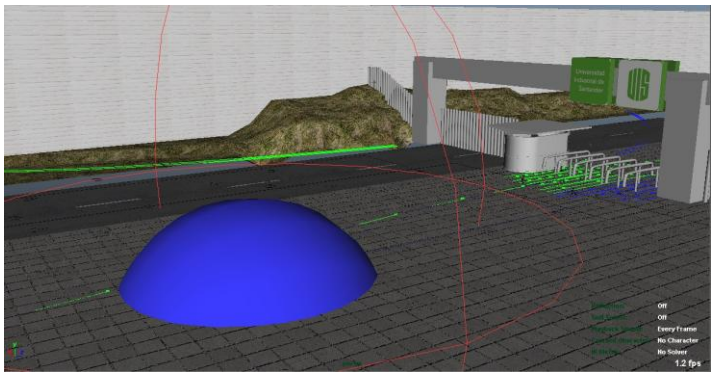
Imagen 22. Agentes entrando y saliendo del claustro.

Una vez realizada la simulación de los agentes desplazándose con normalidad en el escenario planteado, se procedió a generar el efecto de explosión buscando recrear la dinámica de comportamiento de la multitud a manera de prueba, ya que registros históricos muestran la necesidad de evaluar el comportamiento de la multitud en acontecimientos sucedidos en el claustro como se ilustra a continuación por la comunidad universitaria.

Imagen 26. Incidentes registrados en el claustro.

Se estableció que el efecto explosión perjudica a los agentes próximos al lugar del siniestro en ese instante como se ilustra a continuación.

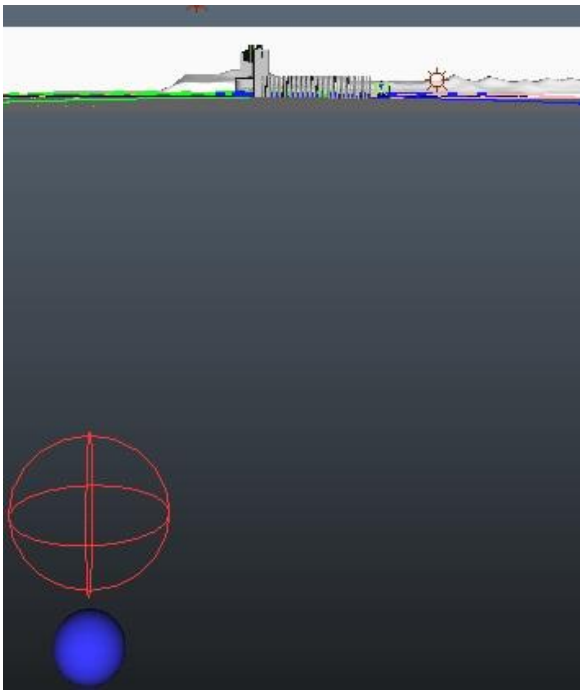
Imagen 23. Spot usado en el efecto explosión.



En la imagen anterior se puede observar el spot identificado con el “ID 1” de color rojo, con el cual se logrará generar un efecto de explosión en el que los agentes ubicados en un radio próximo “saldrán volando” de una manera controlada. Para lograr esto, se hizo uso de funciones de Miarmy tal que:

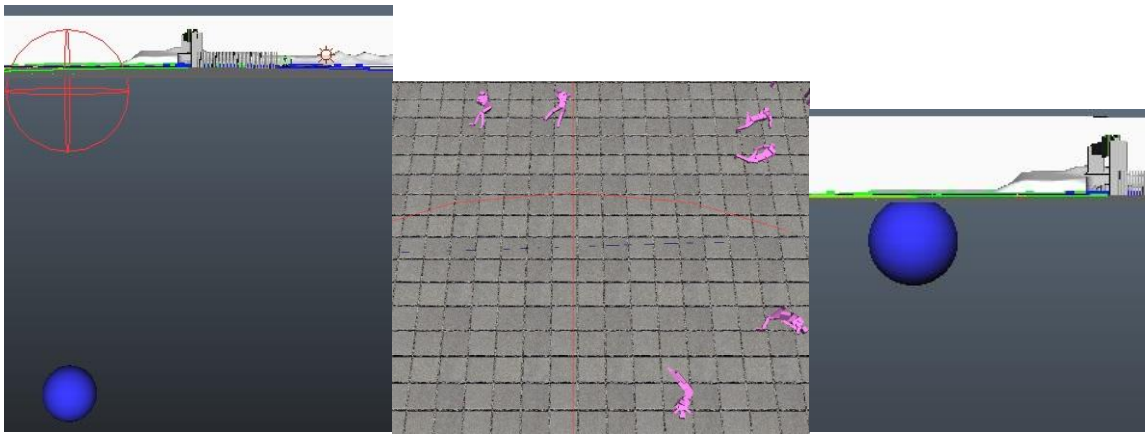
Los agentes que estén a una distancia en ese instante menor de 90 del spot 1 son expulsados del radio de influencia, éstos agentes caen simulando ser personas heridas, con una tolerancia pequeña con el fin de que no salgan volando hasta el infinito y se deslizan sin parar. Este spot 1 inicialmente está a una distancia muy superior a 90 con el fin de que ningún agente lo perciba, después de un determinado tiempo, el spot se dirige al punto sobre el plano donde se desea generar el efecto.

Imagen 24. Imagen posición inicial del Spot 1.



En el instante en que es percibido por los agentes, se genera el efecto de explosión con los agentes que se encuentren en ese instante a una distancia menor o igual a 90 y de inmediato se aleja tanto como para que el resto agentes no lo perciba como se ilustra en la siguiente imagen.

Imagen 29. Cambio de porción a través del tiempo.



Una vez los agentes perciben el efecto explosión, entra en acción el spot identificado con el ID 2 de color azul que se encontraba a una distancia imperceptible por los agentes, el cual se ubicará inmediatamente después en el lugar del siniestro que ocupó anteriormente el spot 1 , e indicará a los agentes que estén cerca pero que no fueron víctimas de la explosión, que deben alejarse a un radio mayor o igual a 300 y al tiempo cambiar de acción caminar (walkA) con la cual venían a la acción correr (runA).

Las reglas establecidas para los agentes que no fueron heridos ya que no fueron víctimas directas en el efecto explosión fueron:

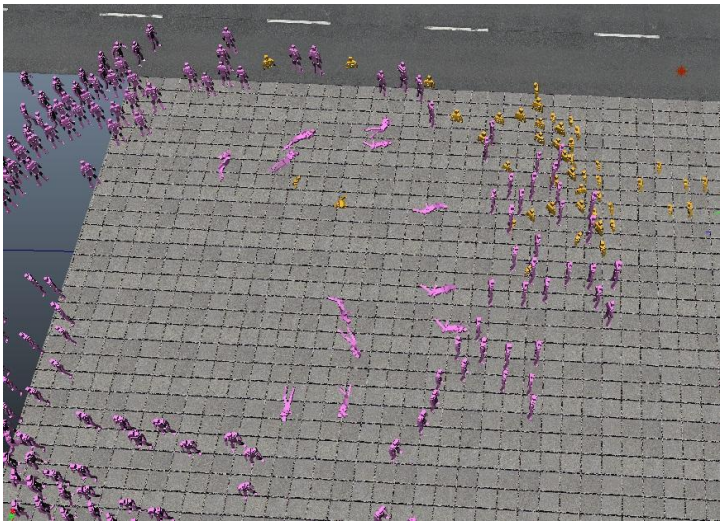
- Todos los agentes que se encuentren a una distancia menor de 300 y a la derecha del spot 2, rotaran a la izquierda a una velocidad de 120 y huirán corriendo.

- Todos los agentes que se encuentren a una distancia menor de 300 y a la izquierda del spot 2, rotaran a la derecha a una velocidad de 120 y huirán corriendo.

Una vez los agentes lleguen a una distancia no menor de 300 optarán por quedarse a mirar qué fue lo que sucedió mientras que otros decidirán alejarse del lugar lo más pronto posible.

El alcance del efecto explosión se maneja de acuerdo con el radio de impacto deseado. Los agentes que fueron alcanzados por el impacto de la explosión quedarán tendidos en el plano como víctimas.

Imagen 25. Momento despues de la explosión.



En este punto termina la simulación cumpliendo con el objetivo inicial planteado.

En el siguiente link se puede visualizar la dinámica planteada anteriormente.
https://www.youtube.com/watch?v=c55FBZJbkWU

Algunas de las ventajas identificadas de usar Miarmy son:

Es más parametrizado, por tanto se tiene un mejor control la definición de reglas para los agentes.

Tiene como restricción rende rizar más de 100 agentes, pero se puede visualizar la dinámica de comportamiento de la multitud con geometrías básicas de más de 100 agentes, si el rende rizado final no representa importancia en el cumplimento de los objetivos del proyecto planteado.

La edición y manipulación del placement, o grupos de agentes, se puede realizar de manera simple variando la taza de agentes tipos que se requiere o la cantidad de agentes, así como la ubicación y orientación de cada uno de los agentes.

Cuenta con documentación que ilustra el funcionamiento de la herramienta, y cuanta con un canal en YouTube con más de 100 video tutoriales donde se enseña la implementación de la herramienta.

El video tutoriales se pueden visualizar de manera libre en el siguiente link:

https://www.youtube.com/watch?v=5W5rqzzyARM&list=PLalQe7BG7XcL8YXywnLD6QSnlN-dV2VZe

Algunas desventajas de Miarmy son:

La no explotación de arquitecturas con aceleradores gráficos genera un aumento en el tiempo de ejecución de la simulación ya que todos los cálculos son realizados en el procesador.

Al generar una acción con un esqueleto y asignarle dicha acción a un esqueleto en diferente escala, se genera error y altera la geometría del modelo.

Como no genera una vista previa de los modelos al final del rende rizado, no se puede tener control preciso de los modelos geométricos de cada uno de los agentes.

A continuación se ilustra un segundo caso en donde se realiza la simulación con otra herramienta para la simulación de multitudes llamada Golaem.

4.2 SEGUNDO CASO DE ESTUDIO.

4.2.1 Identificación del Problema. Se tomó como caso de estudio la dinámica de comportamiento de la multitud en la plazoleta Che Guevara, de la Universidad Industrial de Santander teniendo en cuenta una gran afluencia de personas evacuando el claustro universitario por amenaza de atentado en un día de actividad normal académica**.**

Ya que la plazoleta Che Guevara se encuentra ubicada en medio de la salida principal de la universidad y el edificio de Ingeniería Industrial, el Instituto de lenguas, Bienestar Universitario, Comedores, INSED y frente al auditorio mayor Luis A. Calvo, se convierte en un corredor peatonal para una gran cantidad de personas.

Tiempo después de activada la alarma de evacuación y en medio de la multitud que camina tranquilamente hacia la salida principal para evacuar la universidad, se detona un artefacto explosivo de mediano poder. Se definió que la ubicación

del artefacto explosivo sería justo enfrente del auditorio Luis A. Calvo, en medio de la plazoleta Che Guevara.

Imagen 26. Incidentes registrados en la plazoleta Che Guevara UIS

Una vez el artefacto detona, las personas más cercanas al artefacto se ven afectadas por la explosión y caen heridas en el lugar; mientras que las personas que no queda heridas corren inmediatamente para alejarse del lugar del siniestro.

El comportamiento de la multitud un momento después de huir del lugar se ve reflejado de la siguiente manera:

- Algunas personas se detendrán en el lugar donde se encuentren en ese instante para tranquilizarse, observar y pensar ya sea por miedo o curiosidad todo lo sucedido.

- Un grupo con mayor cantidad de personas intentará llegar rápidamente a la portería de la universidad para alejarse del peligro.

- Otro grupo con mayor cantidad de personas que el anterior, tratará de refugiarse rápidamente en la universidad, dando un giro de 180 grados para

devolverse por donde venía y llegar a un punto de encuentro donde se sientan a salvo.

- También llegarán algunas personas y conforman un círculo alrededor del lugar de siniestro para observar lo sucedido, pedir auxilio y/o ayudar a los heridos.

En este punto se da como terminado el caso de estudio.

A continuación se define los requerimientos necesarios para realizar la simulación del caso planteado anteriormente usando una herramienta diferente para evaluar el modelo planteado.

4.2.2 Definición de Requerimientos. Una vez establecidos los componentes necesarios para la realización de la simulación, se concluyó que el desarrollo del proyecto se realizaría con Golaem desarrolla herramientas para artistas que animan personajes digitales. Integrado en Autodesk Maya, Golaem Crowd hace que sea fácil y asequible para poblar mundos con personajes digitales orientables, desde unos pocos hasta miles. Artistas de todo el mundo utilizan Golaem Multitud para dar vida a los comerciales, producciones episódicas, largometrajes y juegos.

Golaem Crowd es un plug-in para Autodesk Maya que permite la simulación de multitudes de personajes controlables basado en agentes independientes. Es desarrollado por Golaem, una compañía de software con sede en Francia (creada en Rennes en 2009).

Es una herramienta para generar caracteres en las zonas accesibles de la escena (por ejemplo, para colocar un ejército en un campo evitando los árboles y las rocas) Mientras los animadores pueden utilizar herramientas de Maya para hacer los personajes de multitudes (representado como partícula) se mueven en la escena, el plugin proporciona comportamientos de navegación para hacer que se vayan de A a B de forma autónoma. Se permite el cálculo automático de navegación en la malla, a base de hoja de ruta de planificación de trayectoria y configuración de comportamientos de dirección (incluyendo prevención de colisiones reactiva).

El motor de animación incluido permite reproducir el movimiento creado previamente, incluso en personajes de diferentes morfologías. Golaem Multitud puede adaptarse a los movimientos del terreno y calcular automáticamente las transiciones entre los movimientos, movimiento con dos patas o cuatro patas. Proporciona: mapeo automático y editables esqueleto, bípedo y cuadrúpedo motor de animación dedicado, retargeting movimiento automático, mezcla y adaptación al terreno. Las animaciones se pueden activar y mezclar mediante la definición de los comportamientos asociados con las condiciones de arranque / parada de disparo[20].

Golaem Crowd cuenta con una licencia de evaluación que permite hacer exactamente lo mismo que la versión final, pero con tiempo limitado por 30 días.

Permiten rende rizar en un número ilimitado de nodos de forma gratuita, y vienen con un contemporáneo pack de personajes listos para su uso que se pueden descargar de su página oficial.

[20] https://en.wikipedia.org/wiki/Golaem_Crowd

También cuenta con diferentes versiones de pago.

La licencia por 3 meses de alquiler tiene un costo de 1.500 Euros mientras que la licencia permanente por un año tiene un costo de 4.999 Euros con servicios de soporte y actualizaciones.

Hay una versión paga para académicos y tiene un costo de 249 Euros de alquiler por un año[21].

4.2.2.1 Requisitos de Golaem [22]:

SISTEMA OPERATIVO[23]:

- Microsoft Windows XP SP3, Microsoft Windows 7 SP1, Microsoft Windows 8 or 8.1, 64 bit
- Red Hat 6 WS / CentOS 6, Fedora 14 and OpenSUSE 12, 64 bit
- Maya 2016, 64 bit
- Maya 2015, 64 bit
- Maya 2014, 64 bit
- Maya 2013, 64 bit

4.2.3 Modelado del Escenario. El desarrollo de esta simulación se realizó con la última versión disponible de Golaem Crowd en la página oficial, versión 4.2.0.1, instalada en Maya Autodesk 2014.

Inicialmente se realizó un modelo tridimensional con ayuda de las herramientas de diseño de Maya Autodesk del entorno sobre el cual se realizaría la simulación,

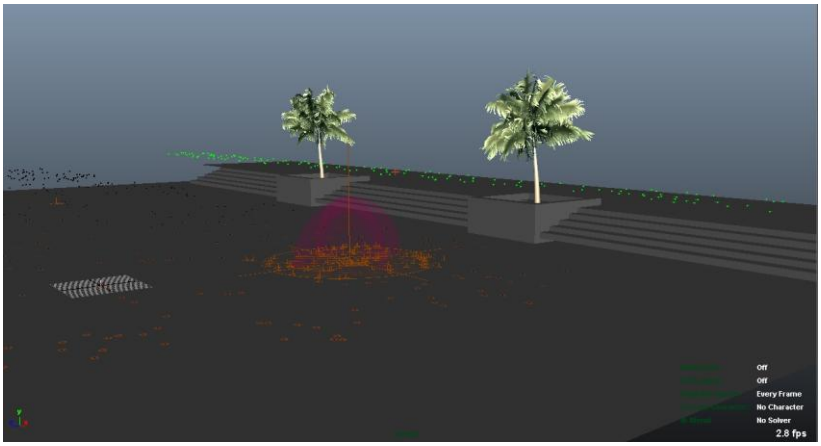
[21] http://golaem.com/content/buy

[22] https://basefount.atlassian.net/wiki/display/MDE/Windows

[23] http://golaem.com/content/doc/golaem-crowd-documentation/platform-requirements

teniendo en cuenta escalas, obstáculos y rutas de evacuación ilustrado a continuación.

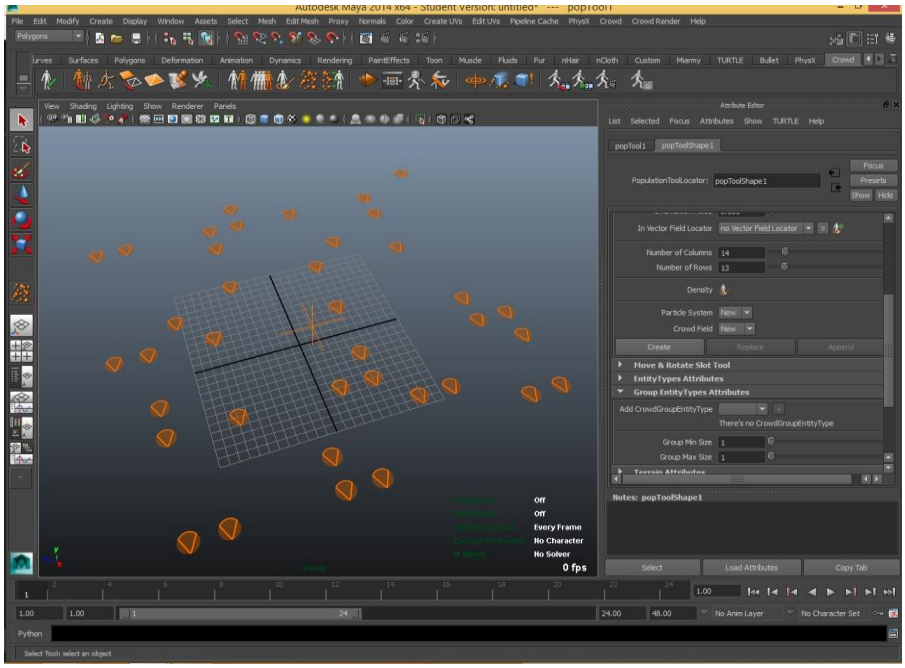
Imagen 32. Modelo grafico escenario, Plazoleta Che Guevara, Universidad Industrial de Santander.



4.2.4 Modelado y Simulación con Golaem en Maya Autodesk.

Para crear una simulación de multitudes con Goleam Crowd, primero se debe generar con la herramienta “Population Tool” los puntos de referencia en los cuales van a ir los modelos, es decir la ubicación, orientación, cantidad de agentes, distancia, cantidad mínima y máxima de grupos de agentes etc.

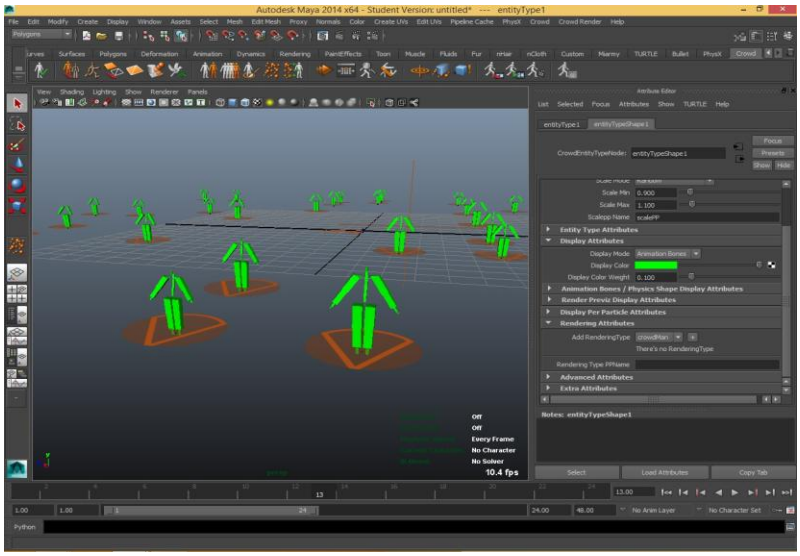
Imagen 27. Ubicaciones en Golaem.



Se crean los puntos definiendo ubicaciones y cantidades, posteriormente se procede a generar el tipo de entidad, en donde se le indica al motor de multitudes cuáles van a ser los modelos que serán ubicados en dichos puntos, escala mínima y máxima en que desea visualizar los modelos, color distintivo del grupo de agentes, etc.

Gracias a los modelos oficiales que vienen listos para trabajar ofrecidos por Golaem, se importaron los modelos y geometrías, éstos cuentan con un esqueleto que asemeja al cuerpo humano y son rende rizados como se muestra en la siguiente imagen.

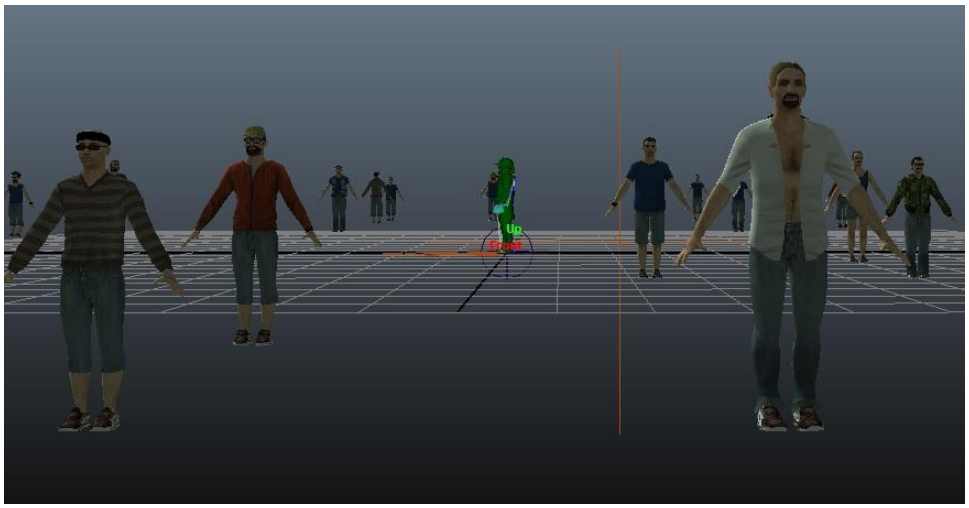
Imagen 28. Render basico de agentes.



Una vez cargados los modelos geométricos y las texturas, serán cargados en la posición de cada partícula y tendrán la orientación anteriormente definida.

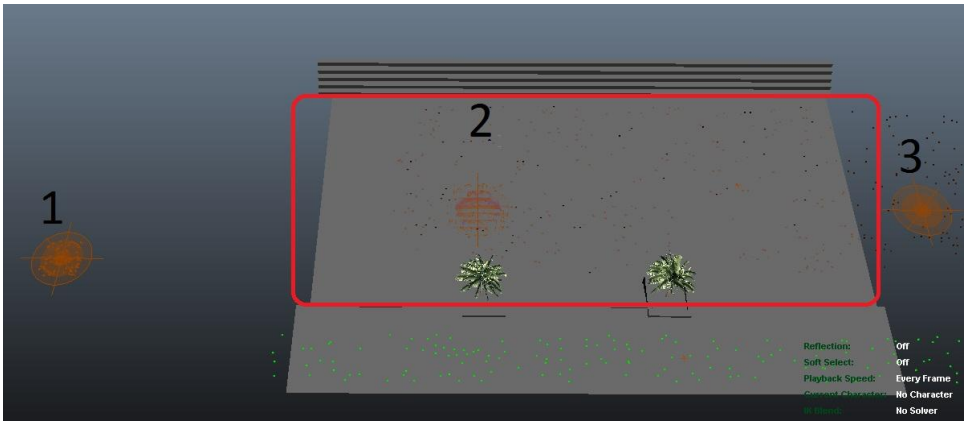
Con el fin de visualizar las geometrías reales de los agentes, se asignó “Render Previz”; éstas geometrías se asignan aleatoriamente a cada personaje con el fin de mostrar diferentes modelos variando geometrías, tamaños y texturas como se ilustra a continuación; valores que pueden ser variados fácilmente y visualizados de inmediato.

Imagen 29. Agentes con geometrias y texturas.



Una vez ubicados los puntos de partida de los agentes sobre el plano de la plazoleta Che Guevara, se estableció que habrán grupos de una a cuatro personas como máximo; todas iniciarán caminando a diferente ritmo en el mismo sentido, buscando llegara la salida principal para evacuar la Universidad.

Imagen 30. Puntos de refecncia sobre el plano.



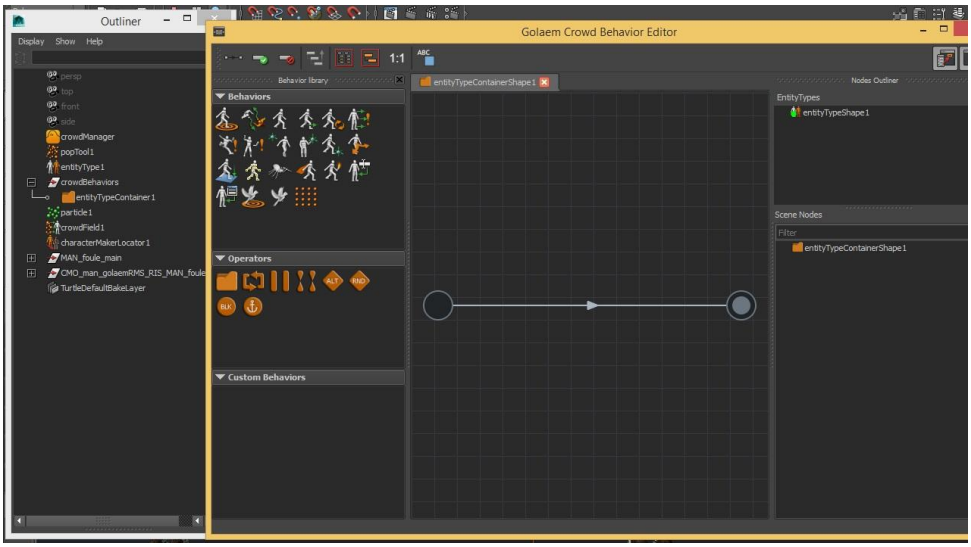
En la imagen anterior se puede observar un área de color anaranjado etiquetado con el número 2, en dicha zona se generará una explosión, y los agentes que se

encuentren ubicados en ese instante de tiempo en un radio de 25, sufrirán los efectos físicos que recrean dicha dinámica.

Los puntos de color anaranjado indicados en la figura anterior con el número 1 y 3, son puntos de referencia para los agentes, el área 1, está ubicada en dirección a la portería principal de la Universidad y el área 3 indica un punto de encuentro dentro de la misma.

Para la administración de estados y reglas, Golaem cuenta con un menú en el cual se listan los grupos conformados por agentes, las conductas que se le pueden establecer a los agentes, operadores para las conductas y carpetas donde almacenar las mismas por grupos, con una secuencia lógica establecida con un diagrama de estados, el cual se activa desde un nodo inicial, y va cambiando de estados a medida que el agente interactúa con el entorno.

Imagen 31. Menú administrador de comportamientos.



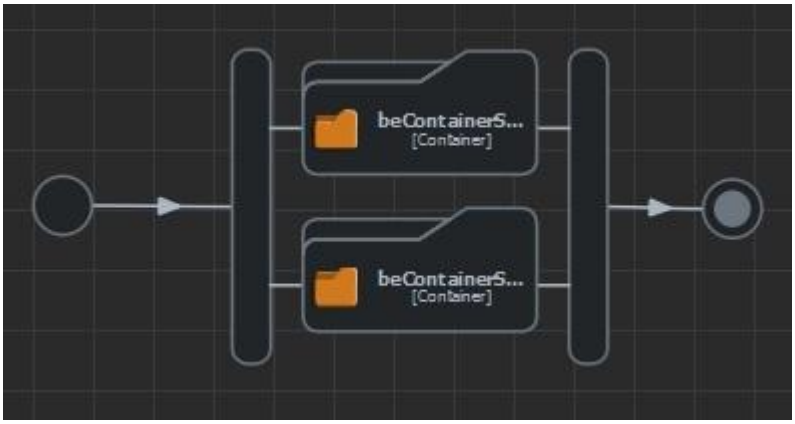
Para agregar operadores o comportamientos, se debe contar con un diagrama de estados en donde sean ubicarlos de forma lógica, éste diagrama se obtiene

creando una carpeta; se pueden crear carpetas dentro de otras carpetas y cada una éstas contendrá un diagrama de estados, esto será de gran ayuda al momento de organizar procesos y prioridades.

Se creó una carpeta llamada “entityTypeContainershape1” la cual contendrá todos los diagramas de estado, conexiones y operadores como se describe a continuación.

4.2.4.1 CARPETA 1 (entityTypeContainershape1). Al diagrama de estado de esta carpeta se agregó el operador “Parallel” el cual procesa varios comportamientos al mismo tiempo y se incluyeron dentro de este operador dos carpetas llamadas beContainershape1 y beContainershape2 con el fin de ser procesadas tan pronto inicie la simulación, y están organizadas de la siguiente manera:

Imagen 32. Diagrama 1.



4.2.4.1.1 CARPETA 2. (beContainershape1). A esta carpeta se agregó el comportamiento llamado “Physicalize” y será procesado tan pronto se active el operador “Parallel” que contiene esta carpeta, el cual activa / desactiva la simulación física de una Entidad.

Una vez el comportamiento deje de ser procesado, éste activará el tipo de física deseado, y se desactivará cuando otro comportamiento “Physicalice” lo indique con la función “Dephysicalize”.

Las condiciones para que este comportamiento inicie son:

- Que el operador “Parallel” esté siendo procesado.

Teniendo en cuenta que el efecto físico surgirá efecto cuando el comportamiento “Physicalize” deje de ser procesado, las condiciones para que este comportamiento termine son:

- Haber transcurrido 30 frames.

- Se indicó, que este comportamiento se aplicará a los agentes que estén ubicados a una distancia menor o igual a 20 con respecto al punto anaranjado etiquetado con el número 1, llamado “crowdTarget1”.

Si se ejecuta la simulación en este punto, los personajes se van a caer al piso después de cumplir las reglas anteriores y se va seguir corriendo simulación física para siempre.

Imagen 33. Agentes caidos efecto Physicalice.

Al lado derecho del comportamiento anterior siguiendo la secuencia lógica, se agregó un comportamiento llamado “Force” el cual, aplicará una fuerza en este caso de expulsión que simula la explosión y afectará a los agentes que están bajo el efecto físico teniendo en cuenta la magnitud de expulsión de los agentes afectados.

Las condiciones para que este comportamiento inicie son:

- Que el comportamiento anterior haya finalizado.

Es decir que se agregan las propiedades físicas a los agentes con el comportamiento “Physicalize” pasados 30 frames, e inmediatamente se procesa el comportamiento “Force”, que aplica a estos agentes una fuerza de expulsión que genera el efecto explosión antes de que los agentes caigan desmayados por el efecto “Physicalize”.

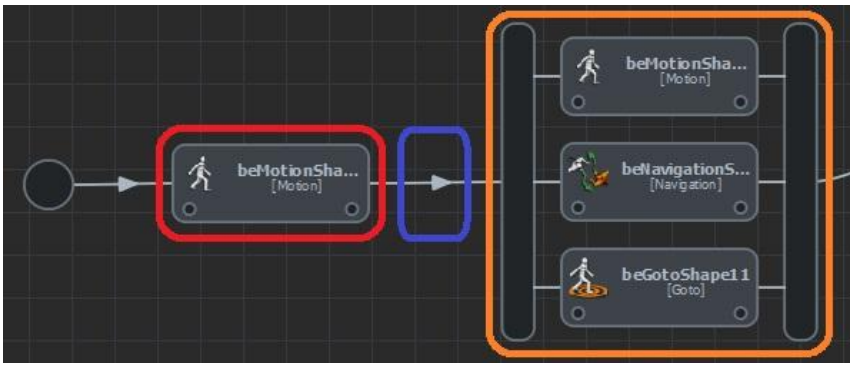
Las condiciones para que este comportamiento termine son:

- El comportamiento haya iniciado.

En este punto se da por terminada la ejecución de comportamientos de esta carpeta.

4.2.4.1.2 CARPETA 3. (beContainershape2). Al diagrama de estados de la carpeta “beContainershape2”, se agregó el comportamiento “Motion” como se ilustra en la siguiente imagen, al cual se le importaron los archivos “WalkNormal”, “WalkFast” y “RunNormal”, que recrean las animaciones de caminar, caminar rápido y correr respectivamente; cabe recalcar que en las propiedades de modelado de Golaem, se puede variar el tiempo de inicio de acción para cada agente, éste efecto ayudará a dar un mayor realismo a la simulación.

Imagen 34. Diagrama de estados 2.



Este estado indicará al agente que ejecute una de esas tres acciones.

Las condiciones para que este comportamiento inicie son:

- Que la carpeta contenedora esté siendo procesada.

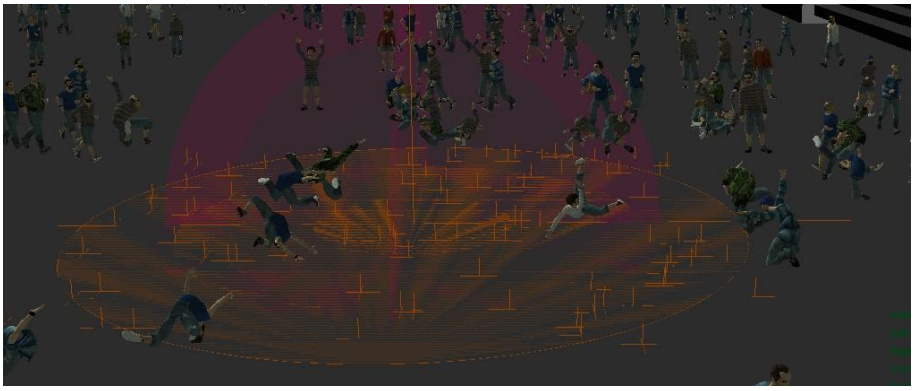
Las condiciones para que este comportamiento termine son:

- Haberse ejecutado la simulación hasta el frame 30.

Imagen 35. Simulación de la multitud con geometrias y texturas.

Una vez generados 30 frames a partir del momento en que el operador “Parallel” que contiene las dos carpetas ha sido procesado, se recrea la simulación de explosión sobre el plano como se ilustra a continuación e inmediatamente da paso para que se inicie el siguiente comportamiento.

Imagen 36. Efecto explosión.



Al lado derecho del comportamiento anterior siguiendo la secuencia lógica, se agregó un operador “Parallel” enmarcado de color anaranjado, el cual contiene los siguientes comportamientos como se evidencia en la imagen anterior:

“beMotionShape14” al cual se le importó la acción de “RunNormal” e indicará a los agentes que no fueron afectados por el efecto explosión, que deben ejecutar la acción “RunNormal” es decir que deben correr.

“beGotoShape11” en donde se indicó, que los agentes que no fueron víctimas de las explosión deberán alejarse del lugar de la explosión (“crowdTarget1”).

“beNavigationShape6” indica a los agentes que deben evitarse unos con otros con el fin de no colisionar entre ellos.

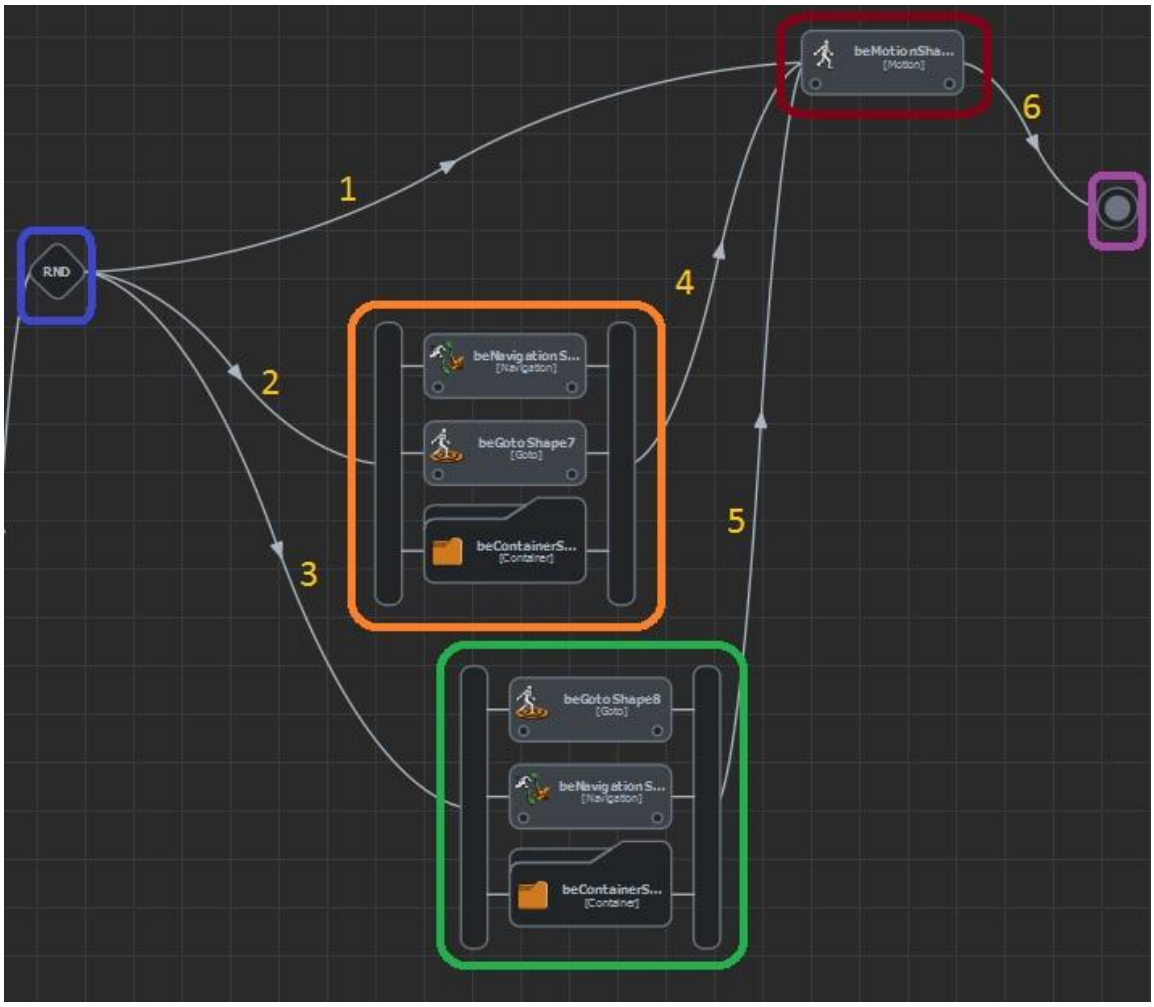
En resumen hasta este instante: La simulación inicia, se procesa el comportamiento enmarcado en la imagen anterior de color rojo, el cual hace que los agentes caminen, troten o corran.

Transcurridos 30 frames se genera el efecto explosión en el instante indicado con el recuadro de color azul, e inmediatamente se procesa el contenedor “Parallel” que indica a los agentes que deben correr, alejarse del lugar de la explosión y evitar colisionar entre ellos gracias a los comportamientos respectivos.

Se estableció como condición que el contenedor “Parallel” termina de ser procesado una vez se ejecute durante 4 segundos el comportamiento “beMotionShape14”.

Al lado derecho del comportamiento anterior siguiendo la secuencia lógica, se agregó un operador “RND” (random) del cual surgen tres posibles caminos que se dirigen a diferentes operadores como se describe a continuación:

Imagen 37. Diagrama de comportamientos.



El camino etiquetado con el número “1” se dirige al comportamiento “beMotionShape8” enmarcado de color vino tinto al cual se le importaron las siguientes animaciones:

StandVictoryShape1: Recrea movimientos de euforia levantando los brazos
StandWatchingShape1: Recrea el estado de observación tranquilo
StandAngry: Recrea el estado de enojado con movimientos en los brazos.

Este estado recreará alguno de los tres comportamientos nombrados anteriormente.
Como se ilustra en la imagen Imagen 43, el camino etiquetado con el número "2" conduce a un operador "Parallel" llamado "beOpParallelShape12" enmarcado de color anaranjado el cual contiene dos compartimentos y un operador.

El operador "Go To" llamado "beGotoShape7" se configuró de tal manera que los agentes buscan llegar al punto objetivo llamado "crowdTarget2", ilustrado en la imagen Imagen 36 etiquetado con el número "1".

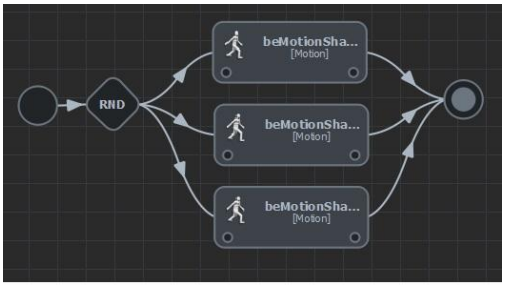
El objetivo de esta conducta es simular el comportamiento de las personas que buscan llegar a la portería principal de la Universidad huyendo del lugar de la explosión.

El operador "Navigation" llamado "beNavigationShape4" indica a los agentes que deben evitarse unos con otros con el fin de no colisionar entre ellos.

4.2.4.1.2.1 CARPETA 4 (beContainerShape5). El diagrama de estados de la carpeta llamada "beContainerShape5" está conformado dela siguiente manera:

Un operador "RND" (Random) llamado "beOpRandomShape3" del cual surgen tres posibles caminos que se dirigen a diferentes operadores:

Imagen 38. Diagrama de Estados



Un comportamiento “Motion” llamado “beMotionShape11” al cual se le agregó la animación “RunNormal” que recrea la dinámica de correr.

Las condiciones para que este comportamiento inicie son:

- Que la carpeta contenedora esté siendo procesada.

Las condiciones para que este comportamiento termine son:

- Estar a una distancia menor o igual a 70 del “crowdTargetShape1”, es decir del lugar de la explosión.

Un comportamiento “Motion” llamado “beMotionShape16” al cual se le agregó la animación “WalkFast” que recrea la dinámica de trotar.

Las condiciones para que este comportamiento inicie son:

- Que la carpeta contenedora esté siendo procesada.

Las condiciones para que este comportamiento termine son:

- Estar a una distancia menor o igual a 75 del “crowdTargetShape1”, es decir del lugar de la explosión.

Un comportamiento “Motion” llamado “beMotionShape17” al cual se le agregó la animación “WalkNormal” que recrea la dinámica de caminar.

Las condiciones para que este comportamiento inicie son:

- Que la carpeta contenedora esté siendo procesada.

Las condiciones para que este comportamiento termine son:

- Estar a una distancia menor o igual a 80 del “crowdTargetShape1”, es decir del lugar de la explosión.

En las propiedades del operador random que precede estos tres comportamientos, se configuro para darle mayor peso a la dinámica de correr, peso medio a la dinámica de trotar y bajo peso a la dinámica de caminar, entendiendo que a mayor peso, mayor cantidad de agentes serán enviados a procesar dicho comportamiento.

Este operador “Parallel” terminará de ser procesado cuando el comportamiento procesado contenido en la carpeta anterior llamada “beContainerShape5” cumpla la condición para terminar.

Como se ilustra en la imagen anterior, el camino etiquetado con el número “3” conduce a un operador “Parallel” llamado “beOpParallelShape13” enmarcado de color verde el cual contiene dos compartimentos y un operador.

La estructura de comportamiento de este operador “Parallel” es similar al explicado anteriormente, la diferencia es que se tomó como punto de referencia el localizador etiquetado con el número “3” en la Imagen 36 llamado “crowdTarget3”, es decir que los agentes buscarán refugiarse en un punto dentro de la Universidad una vez generada la explosión.

En las propiedades del operador random que precede estos tres comportamientos, se configuró para darle mayor peso al operador “Parallel” llamado “beOpParallelShape13” enmarcado de color anaranjado, peso medio al operador “Parallel” llamado “beOpParallelShape12” enmarcado de color verde y bajo peso al comportamiento “Motion” llamado “beMotionShape8”, ilustrado en la imagen 43.

Las conexiones “4” y “5” evidenciadas en la imagen 43 tienen como destino el comportamiento “Motion” llamado “beMotionShape8” ya que una vez termina de procesar cada “Parallel” los agentes se detendrán en algún momento y tomarán las animaciones correspondientes a este comportamiento.

En este punto se da como finalizada la simulación.

Algunas ventajas de Golaem son:

El uso de arquitecturas aceleradas, lo que aumenta la cantidad de frames generados por segundo e ilustra una dinámica fluida de la simulación como se ilustra en la imagen 55.

Se pueden pre visualizar los modelos geométricos de los agentes y variar fácilmente las geometrías.

Gracias a los comportamientos ya establecidos por Goleam, se agiliza el trabajo de modelado del comportamiento.

Se pueden generar grupos de agentes fácilmente variando el tamaño de máximo y mínimo, lo cual genera mayor realismo a la simulación.

Algunas desventajas de Golaem son:

La licencia de prueba vence en 30 días, no permite visualizar el modelo de simulación ni interactuar con agentes una vez se vence el tiempo de prueba.

La creación y manipulación de nuevos agentes dentro de un mismo grupo requiere volver a realizar el proceso de creación del grupo para poderlo modificar.

La documentación para el estudio de la herramienta no es muy clara, se basa en explicar cada herramienta pero no en la implementación de las herramientas en conjunto para realizar el modelo.

5. ESPECIFICACIÓN DE ARQUITECTURA

- Procesador: Intel(R) Core(™) i7-4790 CPU @ 3.60 GHz
- Memoria Instalada (RAM): 8.00 GB
- Tipo de sistema: Sistema operativo Windows 8 de 64 bits.
- Tarjeta Gráfica: Nvidia GeForce GT 610.

24

25

[24] http://www.nvidia.es/object/geforce-gt-610-es.html#pdpContent=2

6. EVALUACIÓN DE DESEMPEÑO

Con el fin de evaluar el rendimiento de la tarjeta gráfica Nvidia Geforce GT 610 usada en este proyecto, se planteó generar la simulación con tres densidades diferentes de agentes para cada uno los casos simulados.

Maya Autodesk cuenta con un indicador en tiempo de simulación que mide los frames por segundo que se están generando, la cantidad de frames por segundo son importantes para la fluidez de la dinámica, con una cantidad de 24 frames por segundo se logrará generar la ilusión de una dinámica realista.

Todas pruebas de rendimientos se realizaron con una configuración de 24 fps ya que es un estándar en la producción de cine y televisión.

El índice que muestra la cantidad de frames por segundo generada por el pre visualizador, se muestra en la parte inferior derecha de la pantalla.

MSI AfterBurner[26] es la herramienta de software utilizada para medir el rendimiento computacional, la cual indica con gráficas y valores numéricos la tasa de trabajo en tiempo de ejecución de la GPU, la memoria RAM, el procesador, cantidad de frames por segundo generados, entre otros.

Se harán las pruebas de pre visualizado con Viewport 2.0 que cuenta con características que agilizan procesos por medio de GPU y Default Quality Rendering para el caso de Golaem.

---

[25] http://digitaltechnology.com.mx/wp-content/uploads/2015/09/T2eC16NyUE9s6NDMWdBQj1zGYBw-60_35.jpg

[26] http://gaming.msi.com/es/features/afterburner

Miarmy no cuenta con pre visualizado en Viewport 2.0 por tanto se harán las pruebas solo en Default Quiality Rendering.

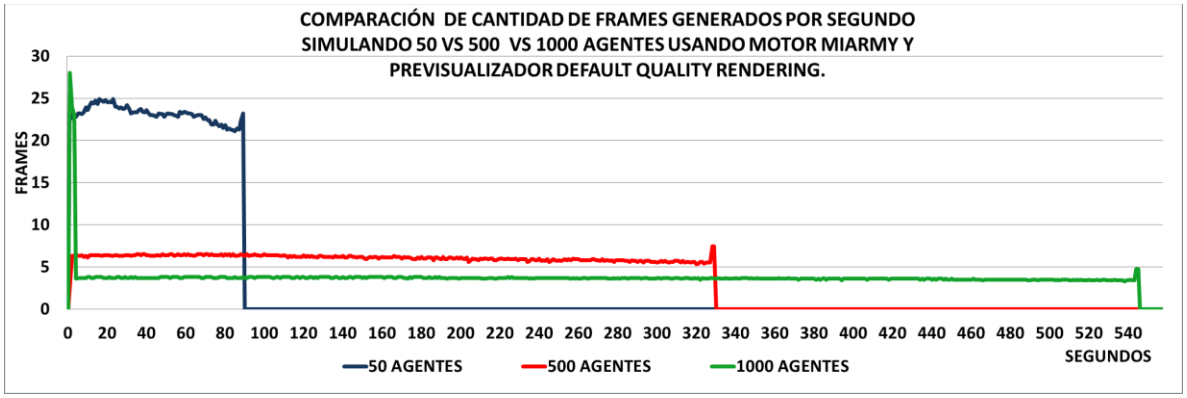
6.1 RENDIMIENTO COMPUTACIONAL USANDO MIARMY

Para las pruebas realizadas con Miarmy, se usó el pre visualizador Default Quality Rendering.

En el siguiente capítulo “Análisis de resultados” se evaluara con mayor detenimiento los resultados presentadas a continuación.

El siguiente gráfico ilustra la cantidad de frames generados por segundo variando la cantidad de agentes. La línea que indica la cantidad ideal de frames por segundo, se muestran como guía, representando el tiempo que debería tardar de simulación en condiciones Ideales de rendimiento.

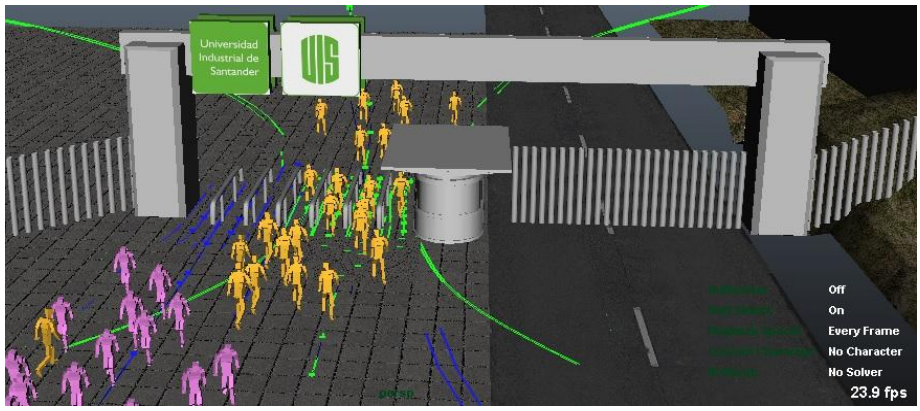
Imagen 39. Fps simulando 50 vs 500 vs 1000 agentes.



En la imagen anterior se puede observar el tiempo requerido por el visualizador para generar 2000 frames simulando 50, 500, 1000 agentes.

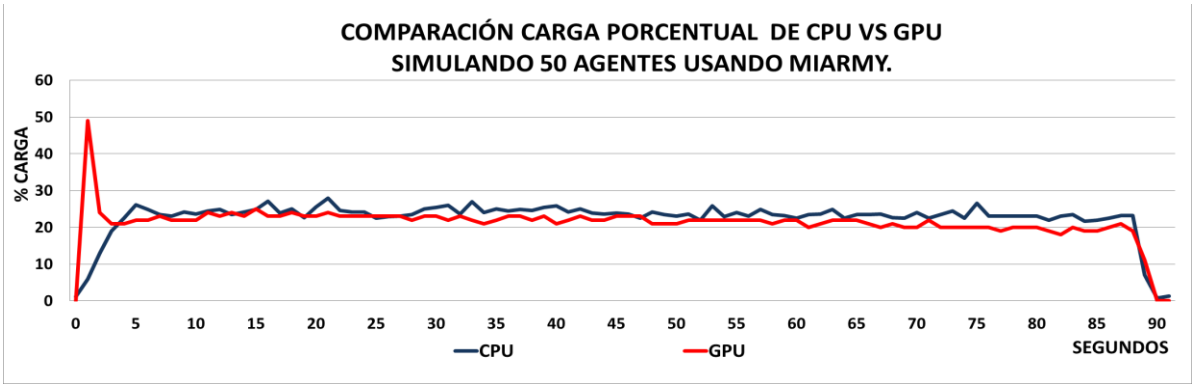
Como primera muestra, se realizó la simulación con 25 agentes ingresando y 25 agentes saliendo de la Universidad.

Imagen 40. 50 Agentes simulados.



En la siguiente imagen se puede observar la cantidad de carga porcentual de la GPU y el promedio de la carga porcentual de las CPU del procesador.

Imagen 41 Carga porcentual de GPU vs CPU simulando 50 agentes.



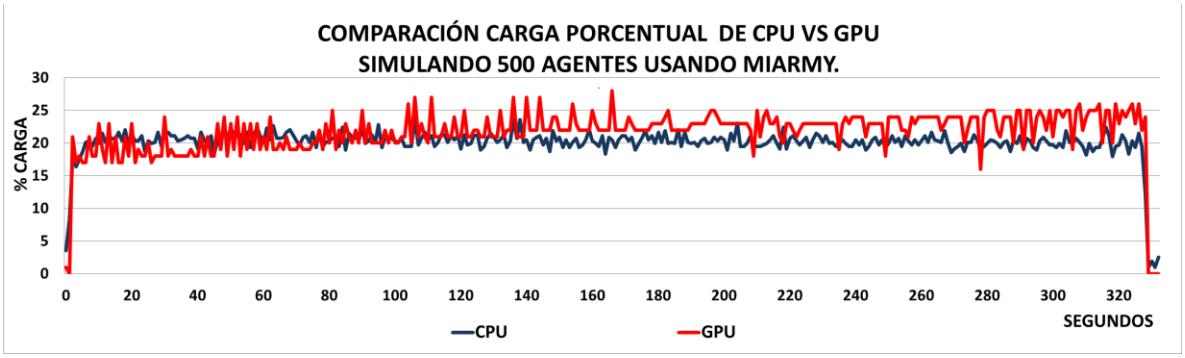
Se puede observar en el grafico anterior que no hay mayor diferencia entre en la carga para procesador y la arquitectura gráfica y demora alrededor de 10 segundos más que el tiempo ideal.

Como segunda muestra, se realizó la simulación con 250 agentes ingresando y 250 agentes saliendo de la Universidad.

Imagen 42. 500 Agentes simulados.

En la siguiente imagen se puede observar la cantidad de carga porcentual de la GPU y el promedio de la carga porcentual de las CPU del procesador.

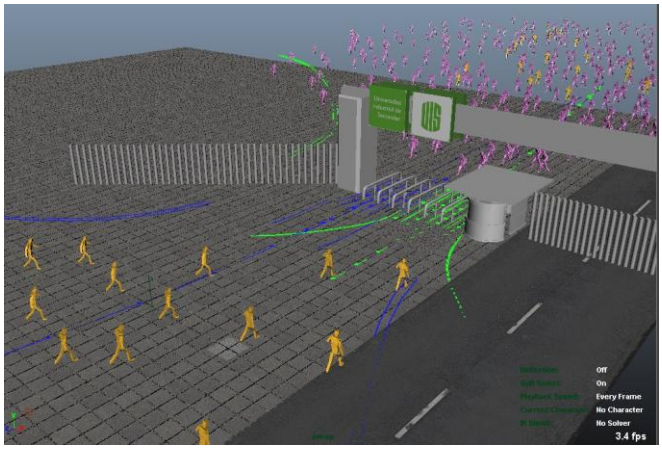
Imagen 43 Carga porcentual de GPU vs CPU.



En el grafico anterior se puede observar que la carga porcentual del procesador permanece constante en el tiempo, mientras que la carga porcentual de la GPU va aumentando, además del aumento de tiempo para generar los frames requeridos.

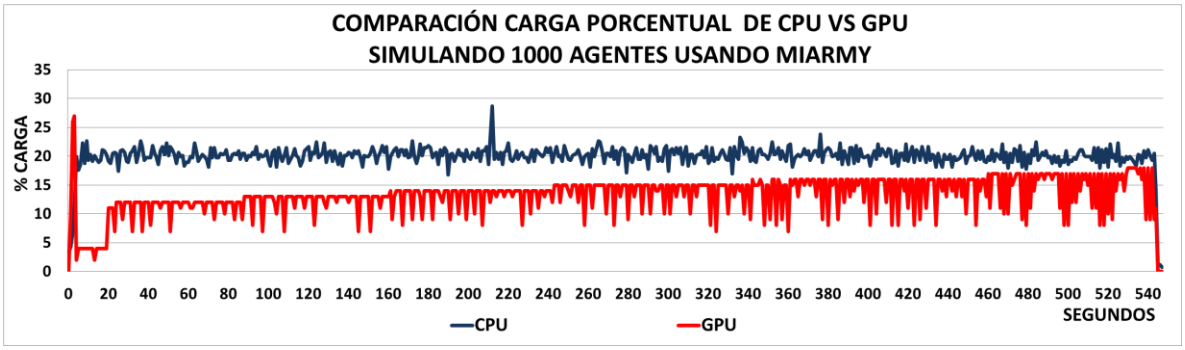
Como tercera muestra, se realizó la simulación con 500 agentes ingresando y 500 agentes saliendo de la Universidad.

Imagen 44. Simulación 1000 agentes.



En la siguiente imagen se puede observar la cantidad de carga porcentual de la GPU y el promedio de la carga porcentual de las CPU del procesador.

Imagen 45. Carga de GPU vs CPU



En el grafico anterior se puede observar que la carga porcentual del procesador es constante en el tiempo mientras que la carga porcentual de la GPU va aumentando y el tiempo en generar la simulación aumento aún más que el caso anterior.

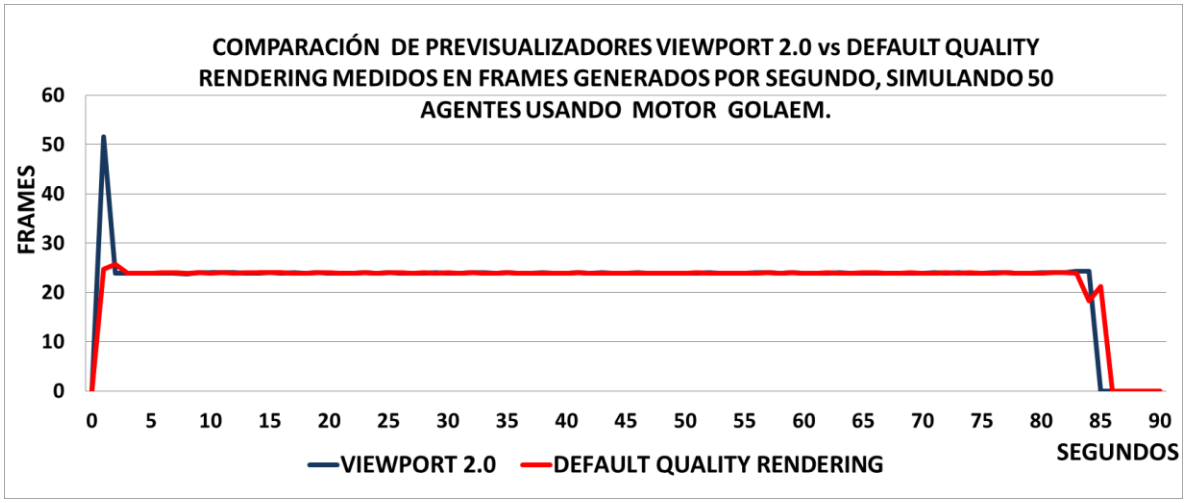
6.2 RENDIMIENTO COMPUTACIONAL USANDO GOLAEM

Para las pruebas realizadas con Golaem, se usó el pre visualizador Default Quality Rendering y Viewport 2.0, La línea que indica la cantidad Ideal de frames por segundo, se muestran como guía, representando el tiempo que debería tardar de simulación en condiciones óptimas de rendimiento.

En el siguiente capítulo “Análisis de resultados” se evaluara con mayor detenimiento los resultados presentadas a continuación.

Como primera muestra, se realizó la simulación con 50 agentes.

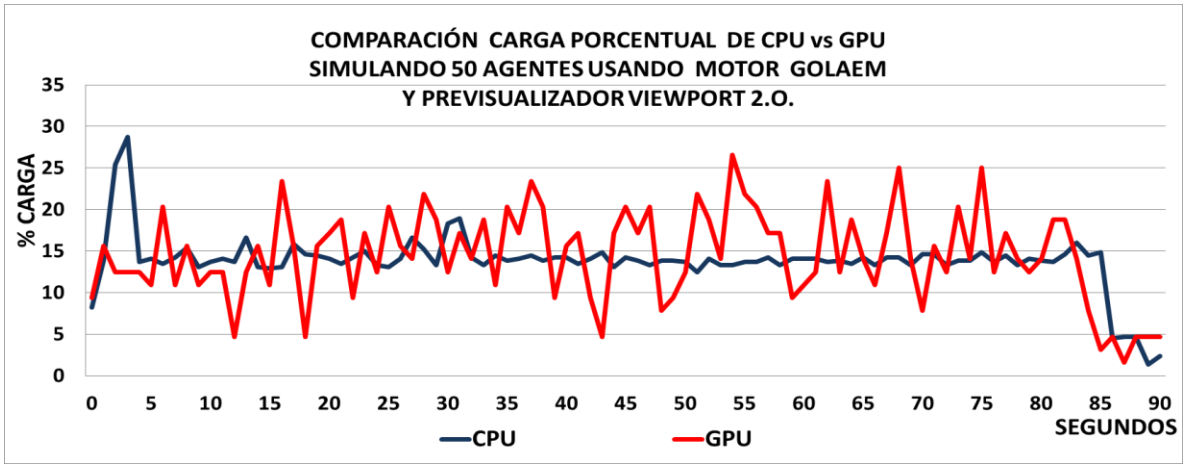
Imagen 46. Rendimiento de renderizado DQR vs viewport 2.0



Como se puede observar en el grafico anterior los frames requeridos fueron generados en el tiempo indicado por el motor de rende rizado Default Quality Rendering y Viewport 2.0.

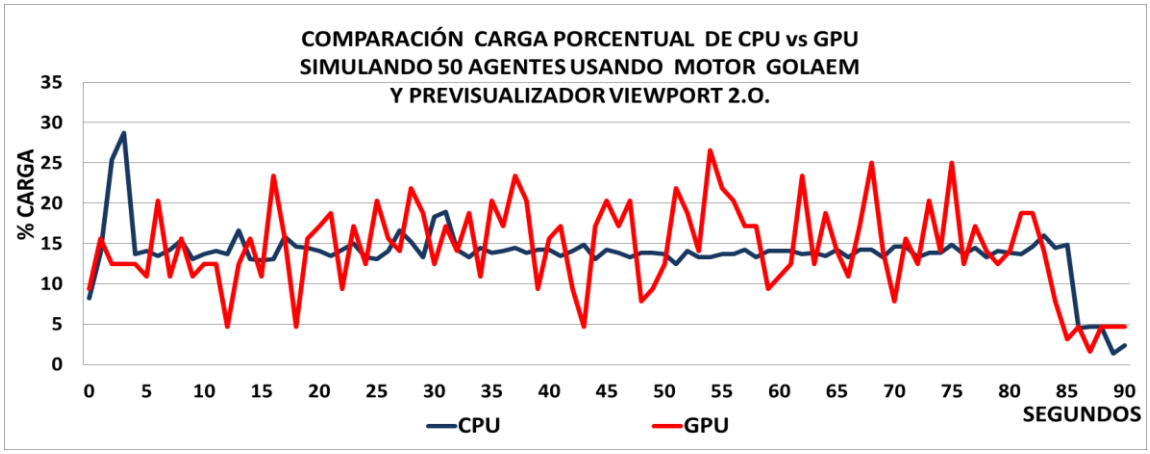
Usando como pre visualizador Viewport 2.0.

Imagen 47. Carga GPU vs CPU.



Se puede observar en el grafico anterior que los frames requeridos fueron generados en el tiempo indicado.
Usando como pre visualizador Default Quality Rendering.

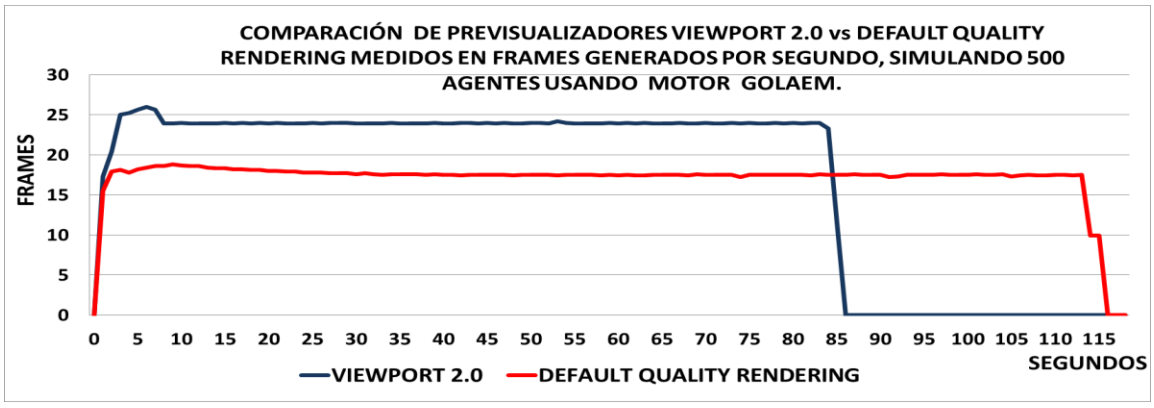
Imagen 48 Carga GPU vs CPU



Como se puede observar en el grafico anterior los frames requeridos fueron generados en el tiempo indicado.

Como segunda muestra, se realizó la simulación con 500 agentes para cada uno de los motores de rende rizado.

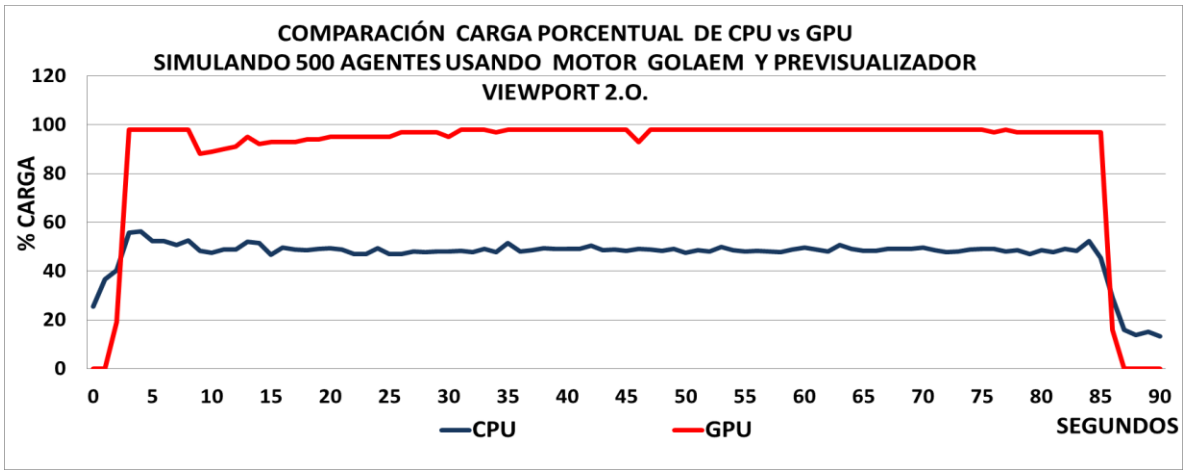
Imagen 49. Frames DQR vs Viewport 2.0



Como se puede observar en el grafico anterior los frames requeridos fueron generados en el tiempo indicado con Viewport, mientras que Default Quality Rendering se tardó más tiempo del ideal.

Usando como pre visualizador Viewport 2.0.

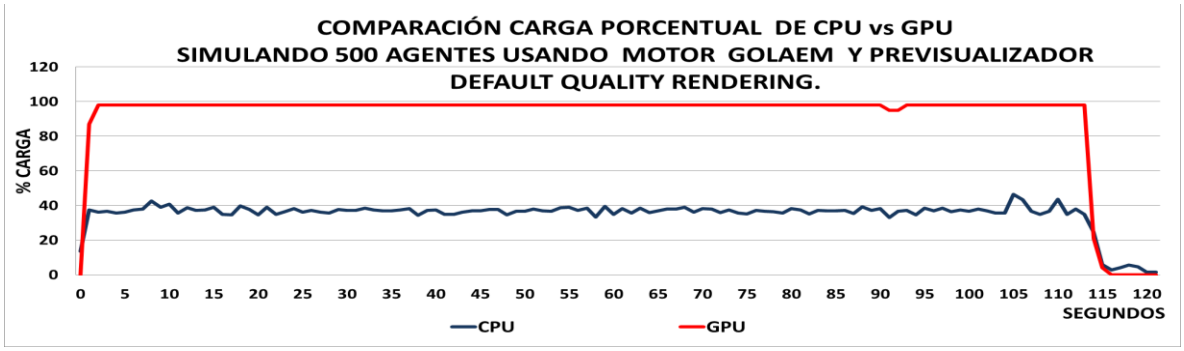
Imagen 50 Carga de GPU vs CPU



Se puede observar en el grafico anterior que los frames requeridos fueron generados en el tiempo indicado con una pequeña diferencia.

Usando como pre visualizador Default Quality Rendering.

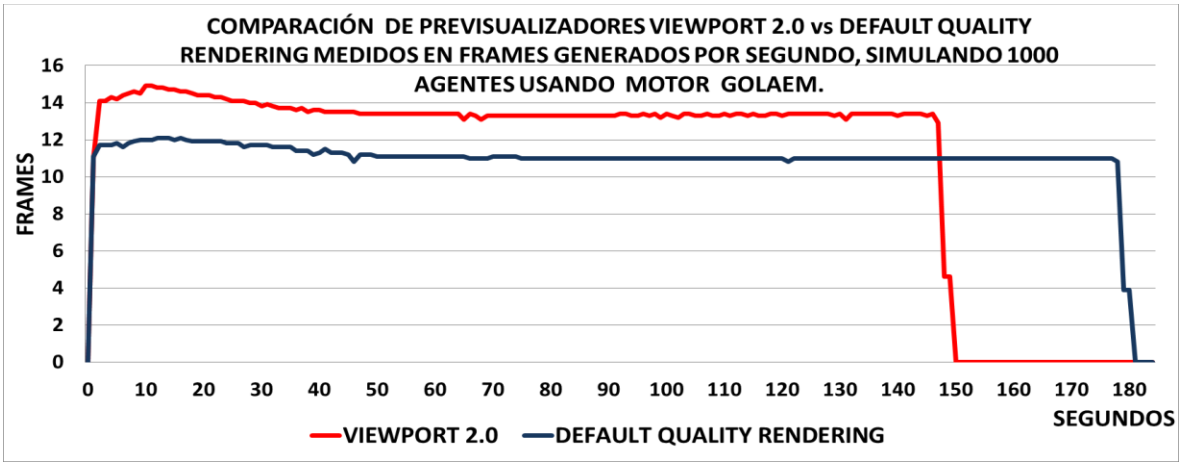
Imagen 51. Carga porcentual GPU vs CPU



Se puede observar en el grafico anterior que los frames requeridos no fueron generados en el tiempo indicado ya que hay una gran diferencia entre el tiempo ideal y el tiempo demorado.

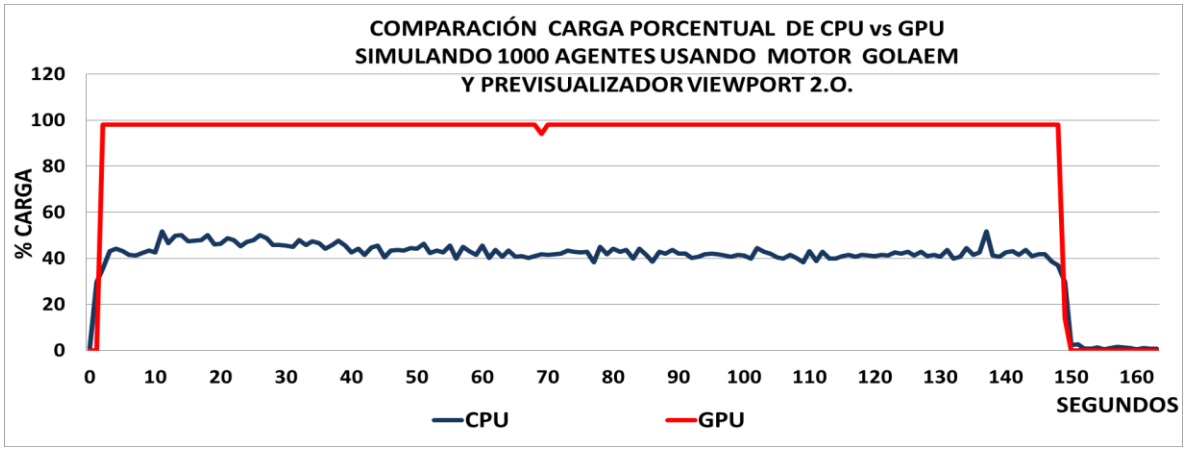
Como tercera muestra, se realizó la simulación con 1000 agentes para cada uno de los motores de rende rizado.

Imagen 52 DQR vs Viewport 2.0



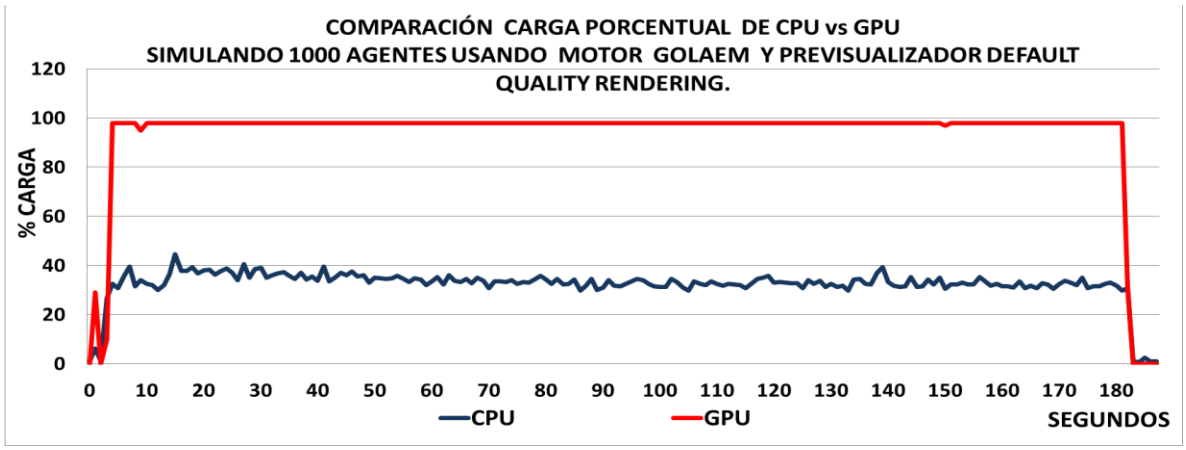
Se puede observar en el grafico anterior que los frames requeridos no fueron generados en el tiempo indicado ya que hay una gran diferencia entre el tiempo ideal y el tiempo generado, pero se puede establecer que viewport 2.0 tiene mejor rendimiento que DQR ya que los realiza en menos tiempo.
Usando como pre visualizador Viewport 2.0.

Imagen 53 Carga porcentual de CPU vs CPU



La carga porcentual de la GPU esta en 100% durante el tiempo de ejecución mientras que la carga de la CPU se mantiene constante alrededor del 50%. Usando como pre visualizador Default Quality Rendering.

Imagen 54 Carga de GPU vs CPU



La carga porcentual de la GPU esta en 100% durante el tiempo de ejecución mientras que la carga de la CPU se mantiene constante alrededor del 40%.

En el siguiente capítulo se hará el análisis de los resultados obtenidos, ilustrados anteriormente.

7. ANÁLISIS DE RESULTADOS

7.1 SIMULACIÓN CON MIARMY

Pre visualizando la simulación con Default Quality Rendering según el gráfico ilustrado en la imagen 45 se puede deducir que:

Simulando el caso de estudio número uno con 50 agentes, la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es pequeña, se demora casi el mismo tiempo en generar 2000 frames, lo que indica que se pudo generar la simulación a una velocidad óptima.

Simulando el caso de estudio numero uno, el porcentaje de carga de CPU y GPU simulando 50 agentes fue variando entre el 20% y 30 % de su capacidad total como se ilustra en la Imagen 47, La GPU mostró una leve tendencia a disminuir la carga, ya que a razón del tiempo la cantidad de agentes a visualizar disminuye debido a la velocidad de paso por el escenario.

Simulando el caso de estudio número uno con 500 agentes, la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es mayor, se demora más de tres veces veses lo que demora el caso ideal generando 2000 frames, lo que indica que el flujo de la dinámica se visualiza de manera deficiente.

Como se ilustra en la imagen 49 el porcentaje de carga de la GPU simulando 500 agentes tuvo mayor variación incrementando la cantidad de picos y su longitud, variando entre el 15 % y 25 % de su capacidad total en la mayor parte de la simulación. Se pudo observar una tendencia a aumentar el porcentaje de carga a

razón del tiempo ya que la cantidad de agentes a visualizar va aumentando debido a la velocidad de paso por el escenario.

El porcentaje promedio de carga de la CPU disminuyó levemente pero se mantuvo constante en el tiempo con una mayor cantidad de picos que el caso anterior.

Simulando el caso de estudio número uno con 1000 agentes, la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es mayor que el caso anterior, se demora más de seis veces de lo que demora el caso ideal generando 2000 frames, esto indica que el flujo de la dinámica se visualiza de manera muy deficiente.

Como se ilustra en la imagen 51, el porcentaje de carga de la GPU simulando 1000 agentes tuvo mayor variación que el caso anterior incrementando la cantidad de picos y su longitud. Se pudo observar una tendencia a aumentar el porcentaje de carga a razón del tiempo ya que la cantidad de agentes a visualizar va aumentando debido a la velocidad de paso por el escenario.

El porcentaje promedio de carga de la CPU disminuyó levemente pero se mantuvo constante en el tiempo con una mayor cantidad de picos que el caso anterior.

7.2 SIMULACIÓN CON GOLAEM

Simulando el caso de estudio número dos con 50 agentes pre visualizando con Viewport 2.0.

Como se ilustra en la imagen 52 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es nula, se demora el mismo tiempo en generar 2000 frames, lo que indica que se pudo generar la simulación a una velocidad óptima.

También se puede observar que el porcentaje promedio de carga de la GPU es de aproximadamente el 15% con picos de gran magnitud, y el porcentaje del promedio de carga de la CPU esta al rededor del 15 % con picos de pequeña magnitud.

Simulando el caso de estudio número dos con 50 agentes pre visualizando con Default Quality Rendering.

Como se ilustra en la imagen 52 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es mínima, se demora casi el mismo tiempo en generar 2000 frames, lo que indica que se pudo generar la simulación a una velocidad óptima.

El porcentaje de carga de la GPU es alrededor del 13% y se mantiene constante con pocos picos y de pequeña magnitud. La carga de CPU se mantuvo constante en el tiempo con una carga aproximada al 17% y picos de pequeña magnitud (ver imagen 54).

Simulando el caso de estudio número dos con 500 agentes pre visualizando con Viewport 2.0.

Como se ilustra en la imagen 55 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados es mínima, se demora casi el mismo tiempo en generar 2000 frames, lo que indica que se pudo generar la simulación a una velocidad óptima.

En la imagen 56 se puede observar que el porcentaje de carga de la GPU es máximo pero se mantiene constante en el tiempo con pocos picos de poco tamaño y que el porcentaje de carga de la CPU es de aproximadamente un 50% y se mantiene constante.

Simulando el caso de estudio número dos con 500 agentes pre visualizando con Default Quality Rendering.

Como se ilustra en la imagen 51 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados grande, se demora alrededor de 30 segundos más en generar 2000 frames, lo que indica que se pudo generar la simulación ralentizada.

En cuanto al porcentaje de carga de uso de la CPU disminuyó a menos del 40%, y aumentó la cantidad de picos con pequeña magnitud, pero el porcentaje de carga de la GPU se mantuvo al máximo constante en la mayoría del flujo (Imagen 57).

Simulando el caso de estudio número dos con 1000 agentes pre visualizando con Viewport 2.0.

Como se ilustra en la imagen 58 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados grande, se demora más de 60 segundos aproximadamente en generar 2000 frames, lo que indica que se generó la pre visualización bastante lenta.

En la imagen 59 se evidencia que el porcentaje de carga de la GPU es máximo y constante en el tiempo, mientras que el porcentaje de carga de la CPU se mantiene alrededor del 50%.

Simulando el caso de estudio número dos con 1000 agentes pre visualizando con Default Quality Rendering.

Como se ilustra en la imagen 58 la diferencia entre la línea que representa el flujo ideal de fps y el flujo de fps generados mayor que el caso anterior, se demora más de 90 segundos aproximadamente en generar 2000 frames, lo que indica que se generó la pre visualización más lenta que el caso anterior.

El porcentaje de carga de la GPU es máximo y constante en el tiempo, mientras que el porcentaje de carga de la CPU es aproximadamente de 35% y se mantiene constante con picos de pequeña magnitud (Imagen 60).

8. CONCLUSIONES

Se definieron modelos descriptivos para implementar sobre herramientas como Miarmy y Golaem, que facilitan utilizar la matemática para construir el modelado matematico necesario que genera la simulación de multitudes con la interacción de gran cantidad de agentes.

Tanto Miarmy como Golaem tienen más de (5) cinco años en la industria y han requerido de un equipo de profesionales especialistas en diferentes áreas para lograr su desarrollo, esto muestra la no viabilidad de desarrollar un proyecto, con tales características, en un proyecto académico. Es más útil definir lineamientos y mecanismos metodológicos de implementación o construir librerías asociadas a infraestructuras, métodos numéricos no existentes o comportamientos en las simulaciones no incluidos. Igualmente, es importante considerar trabajar de la mano con software de diseño tridimensional que involucre ambientes, desarrollando "plugins" especializados que generen la dinámica de comportamiento de las multitudes.

Se evaluó la implementación sobre aceleradores gráficos, de la dinámica del comportamiento de la multitud en un caso aplicado y se evidenció la importancia del uso de estos aceleradores (no solamente las tarjetas gráficas tipo GPU de NVIDIA utilizadas en el proyecto, sino también tarjetas integradas a la tarjeta madre que permiten la visualización de las simulaciones y su respectivo rasterizado), en la simulaciones de la multitud, como se ilustra en la imagen55, Cabe resaltar que la arquitectura aceleradora debe tener características de rende rizado como lo hacen la línea GeForce de Nvidia, usada en este proyecto. Es

importante resaltar que las tarjetas de tipo TESLA (Fermi o Kepler) permitirán calcular las geometrías pero no permiten rasterizar.

Se estableció un flujo (workflow) con lineamientos para simular y visualizar la dinámica de comportamiento de la multitud, usando herramientas profesionales que hacen uso de arquitecturas con aceleradores gráficos. Este flujo involucra: (listar los pasos)

La importancia de este tipo de simulaciones radica en el hecho de permitir observar el comportamiento de multitudes, para apoyar la toma de decisiones. Por ejemplo, en la operación de sistemas de transporte masivo (portales del transporte integrado de servicio en Bucaramanga), estadios (planes de evacuación o de acceso, asonadas), tráfico vehicular (observación de horas pico, contraflujos vehiculares, congestión por accidentes), seguridad (choques con la fuerza pública, atentados terroristas), investigación aplicada a ciencias de la vida (migraciones, epidemias, colonias de hormigas y abejas) y el desarrollo de animaciones asociadas a video juegos, cine y publicidad.

Observando la importancia de este tipo de simulaciones es conveniente fortalecer esta línea de desarrollo, para apoyar las necesidades académicas, industriales y sociales.

9. LIMITACIONES

Este proyecto se limitó al uso de las dos herramientas (Miarmy y Golaem) pero las mismas fueron suficientes para recrear la dinámica de grandes multitudes y apreciar la necesidad de aceleradores gráficos para obtener simulaciones útiles en la toma de decisiones

EL objetivo general del proyecto estuvo limitado a identificar la necesidad de usar arquitecturas con aceleradoras, mas no la selección de las mismas para casos específicos.

Los modelos y simulaciones no pretenden identificar gran variedad de comportamientos de los agentes, sino simular los comportamientos básicos que determinan la dinámica de una gran multitud, frente a un hecho especifico.

Los modelos de comportamiento de las multitudes ilustradas en este proyecto fueron concebidos de manera intuitiva, para el desarrollo de modelos confiables y útiles para la toma de decisiones debe contarse con la participación de expertos.

Se usaron dos visualizadores a modo de prueba que viene instalados por defecto en Maya Autodesk con el fin de comparar rendimiento computacional ya que Default Quiality Rendering no hace uso de arquitecturas aceleradoras mientras de Viewport 2.0 sí.

Los planos sobre los cuales se realizó la simulaciones respectivas fueron elaborados sin tener en cuenta escalas. Estos escenarios fueron diseñados para el desarrollo del proyecto.

La cantidad de agentes simulados, orientaciones, comportamientos y flujo de la multitud se establecieron a modo de prueba sin tener en cuenta la dinámica real, ya que el principal objetivo de este proyecto es evaluar la necesidad de usar arquitecturas graficas en este tipo de simulaciones.

10. RECOMENDACIONES PARA PROYECTOS FUTUROS

La realización de una interfaz fuera del software de diseño en donde se puedan variar los parámetros de simulación de manera rápida e intuitiva para la toma de decisiones.

Realizar la investigación del comportamiento de la dinámica de la multitud usando la GPU en el cálculo de la dinámica del comportamiento, además del rende rizado, tratando por ejemplo de soportar el cálculo con tarjetas tipo TESLA y la rasterización con otro hardware o software integrado. Esto implicaría observar el rendimiento general en una nueva arquitectura en el que los dos componentes (cálculo de las geometrías y sus interacciones, rasterizado) son diferentes pero relacionados.

Realizar plugin sobre Miarmy con el fin de dar solución a las limitaciones planteadas anteriormente de esta herramienta y plantear nuevas reglas con el fin de mejorar el modelo. Esto igualmente involucra definir más allá del uso del openGL otro tipo de lenguajes de programación.

11. REFERENCIAS BIBLIOGRÁFICAS

[1] Oriam De Gyves, Leonel Toledo, Isaac Rudomín,, Comportamientos en simulación de multitudes: revisión del estado del arte.

[2] Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. Human mobility, social ties, and link prediction. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11, page 1100, New York, New York, USA, 2011. ACM Press.

[3] Oriam De Gyves, Leonel Toledo, Isaac Rudomín, Simulación de grandes multitudes con dinámica de grupos, Research in Computing Science. 72.

[4] Moussa• d, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PloS one 5(4), e10047 (Jan 2010)

[5] [Mas, 2005] Mas, A. (2005). Agentes Software y Sistemas Multi-Agente. Conceptos, arquitecturas y aplicaciones. Prentice Hall.

[6] [Huhns and Singh, 1998] Huhns, M. and Singh, M. (1998). Reading in Agents. Morgan Kaufmann Publishers, Inc.

[7] [Russell and Norvig, 2004] Russell, S. and Norvig, P. (2004). Inteligencia Artificial. Un enfoque moderno. Prentice Hall, second edition.

[8] [Wooldridge, 1997] Wooldridge, M. (1997). Agent-based software engineering. IEEE Proc. on Software Engineering.

[9] [Wooldridge and Jennings, 1995] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: theory and practice. Knowledge Engineering Review, pages 115–152.

[10] [Gilbert, 2008] Gilbert, N. (2008). Agent-Based Models. SAGE Publications.

[11] Elena Núñez González, 8 de Septiembre de 2011, Modelado de Evacuación de Multitudes Mediante Agentes Y Transcripción de Comportamientos.

[12] Francesc Guim, Ivan Rodero, Arquitecturas basadas en computación gráfica (GPU).

[13] Omar Espino Santana, 2010. Graphics Processing Units (GPUs).

BIBLIOGRAFIA.

Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. Human mobility, social ties, and link prediction. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11, page 1100, New York, New York, USA, 2011. ACM Press.

Francesc Guim, Ivan Rodero, Arquitecturas basadas en computación gráfica (GPU).

Gilbert, N. (2008). Agent-Based Models. SAGE Publications.

Huhns, M. and Singh, M. (1998). Reading in Agents. Morgan Kaufmann Publishers, Inc.

Mas, A. (2005). Agentes Software y Sistemas Multi-Agente. Conceptos, arquitecturas y aplicaciones. Prentice Hall.

Moussa• d, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PloS one 5(4), e10047 (Jan 2010)

Núñez González Elena, 8 de Septiembre de 2011, Modelado de Evacuación de Multitudes Mediante Agentes Y Transcripción de Comportamientos.

Omar Espino Santana, 2010. Graphics Processing Units (GPUs).

Oriam De Gyves, Leonel Toledo, Isaac Rudomín, Simulación de grandes multitudes con dinámica de grupos, Research in Computing Science. 72.

Oriam De Gyves, Leonel Toledo, Isaac Rudomín,, Comportamientos en simulación de multitudes: revisión del estado del arte.

Russell, S. and Norvig, P. (2004). Inteligencia Artificial. Un enfoque moderno. Prentice Hall, second edition.

Wooldridge, M. (1997). Agent-based software engineering. IEEE Proc. on Software Engineering.

Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: theory and practice. Knowledge Engineering Review, pages 115–152.