Universidad de las Ciencias Informáticas

Facultad 5

Módulo de animación de personajes para Simuladores y Juegos

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Autor: Karel Pérez Ramírez

Tutores: Ing. Fernando Jiménez López

Ing. Yulier Casas Estrada

Ciudad de la Habana

Mayo del 2007

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al *Proyecto Herramientas de Desarrollo para Sistemas de Realidad Virtual* de la Facultad 5 de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Karel Pérez Ramírez	Fernando Jiménez López
_____	_____
Firma del Autor	Firma del Tutor

Dedicatoria

A mi madre y mi hermana por su confianza y amor.

A mis abuelos por ser especiales.

A mi padre por su ejemplo y a mis hermanos.

A mi familia.

A mi novia y amigos de siempre.

Agradecimientos

A mis compañeros y amigos del proyecto.

A Leticia, Fernando y Yanoski.

A todos los que me han ayudado a cultivar conocimientos.

A quienes han hecho más fácil estos años de estudio.

Resumen

El desarrollo de las técnicas de la Realidad Virtual (RV) ha alcanzado niveles de realismo impresionantes por la capacidad de representar conceptos de forma muy rápida mediante la gráfica. Estas se hacen cada vez más populares por su gran acercamiento a la realidad. En las aplicaciones de RV se pueden encontrar cualquier tipo de objetos. Es muy frecuente ver objetos complejos -como humanos o animales- que necesitan reproducir animaciones para ser más reales. El objetivo de este trabajo es desarrollar un módulo para el manejo de colecciones de animaciones.

En la investigación realizada como parte de este trabajo se abordan los conceptos necesarios y técnicas más usadas en la manipulación de animaciones de personajes en espacios tridimensionales (3D) en tiempo real. En este documento se muestra cómo definir el comportamiento de personajes a partir de colecciones de animaciones predefinidas y también algoritmos eficientes para la transición y mezcla de animaciones.

Se expone, a continuación de la fase de investigación, el diseño e implementación de un conjunto de clases que permiten la animación de personajes. Como resultado de este proceso se obtuvo un módulo que cuenta con las características necesarias para su acople a la herramienta de representación de escenas 3D *SceneToolkit* (STK). Con el uso del módulo desarrollado se facilita la construcción de Sistemas de Realidad Virtual (SRV) con prestaciones para la manipulación del comportamiento de los personajes insertados en las escenas representadas por la STK, reduciendo además el tiempo de desarrollo de estas aplicaciones.

Palabras claves: Realidad Virtual, 3D, animación, personajes, colecciones, roles, transiciones, interpolación.

Contenido

Introducción

La gráfica por computadora es de las ramas de las ciencias de la computación que gozan de mayor popularidad y su acelerado y reciente desarrollo tiene un marcado impacto en el avance de las ciencias. Debido a este desarrollo también se han alcanzado sustanciales logros en los sistemas de Realidad Virtual (RV).

Las aplicaciones de la RV se han convertido en sistemas muy comunes y populares en nuestro tiempo; están en numerosas esferas de la vida debido a su amplia área de aplicación. Estos sistemas permiten representar situaciones de nuestro mundo con gran realismo por los niveles de detalles, de inmersión e interactividad que se han obtenido.

Los aportes de la RV se hacen muy notables en la educación con el empleo de las multimedias; en las investigaciones científicas con la visualización de fenómenos complejos; en la industria del entretenimiento, donde resaltan el cine y los videos-juegos; y en la simulación, donde se conocen en la actualidad numerosos tipos de simuladores desde la conducción de autos hasta los simuladores más complejos de aviación. [1]

Las crecientes prestaciones de los ordenadores permiten representar entornos sintéticos, complejos en cuanto al número de objetos que en él se pueden representar y la complejidad de estos; por lo cual los usuarios de los mismos esperan cada vez más realismo. Con mucha frecuencia en las escenas representadas mediante aplicaciones de RV, se encuentran objetos como plantas, lluvia, polvo, niebla, objetos blandos, articulados y fluidos que logran un acercamiento mayor a la realidad.

Los objetos animados son de especial importancia en los entornos sintéticos debido a que le agregan un gran dinamismo y los hacen más reales. Entre los objetos animados se distinguen autos, aviones, barcos, vehículos de combate, plantas, animales y figuras humanas u otras con características similares.

Tanto en juegos como en simuladores se hace muy necesaria la presencia de personajes; fundamentalmente en los entornos urbanos y juegos de deporte o combate donde presentan un comportamiento que los distingue unos de otros y está definida su interactividad.

Figura 1: Juegos donde aparecen modelos animados para representar figuras humanos.

También los personajes poseen gran importancia porque pueden realizar acciones que para un humano resultarían muy difíciles o imposibles. Así, también aumentan los aportes de las aplicaciones de RV, contribuyendo a incrementar los beneficios económicos y a reducir el riesgo para el hombre en actividades tales como entrenamientos, etc.

Cuba ha tenido un reciente desarrollo de las técnicas de RV. Apoyando este avance, en la "Universidad de las Ciencias Informáticas" se desarrolla la herramienta: *SceneToolkit* (STK) a cargo del grupo "Herramienta de desarrollo de aplicaciones de Realidad Virtual". Esta es capaz de manipular escenas tridimensionales en tiempo real, haciendo mucho más fácil y por tanto más rápida la construcción de aplicaciones de RV como son simuladores y juegos (proyectos productivos de la UCI). Dicha herramienta cuenta con módulos especializados para cada una de sus funciones, que permiten el control de los elementos de las escenas.

En particular, brinda varias clases que permiten la reproducción de animaciones de objetos rígidos y articulados. Estos últimos, mediante deformación de malla por esqueletos. Pero la STK carece de un módulo integrado de clases que manipule colecciones de animaciones aplicadas a los personajes, haciendo combinaciones de animaciones predefinidas.

Este trabajo parte del siguiente **problema**: la STK no es capaz de manejar colecciones de animaciones. Esto hace que los productos desarrollados -utilizando la misma- no tengan la calidad requerida en materia de animaciones.

El **objeto de estudio** son las animaciones en tiempo real en los entornos virtuales y como **campo de acción** tiene las técnicas realistas de manipulación de animaciones en tiempo real.

Como **objetivo** se propone desarrollar un módulo para el manejo de colecciones de animaciones.

Con el fin de dar cumplimiento a los objetivos planteados se hace indispensable realizar varias **tareas** resumidas en las siguientes:

- ✓ Investigar las características de las animaciones en los entornos sintéticos.
- ✓ Estudiar las técnicas actuales utilizadas para la obtención de animaciones, con el fin de lograr un alto nivel de realismo.
- ✓ Estudio de la herramienta *SceneToolKit.*
- ✓ Desarrollar soluciones técnicas para alcanzar los objetivos propuestos.
- ✓ Analizar y diseñar un módulo que de solución a los problemas planteados.
- ✓ Implementar los algoritmos definidos para darle solución al problema.

El desarrollo de este trabajo estará organizado de la siguiente manera:

En un primer capítulo, **Fundamentación Teórica**, se hace un análisis bibliográfico donde se investigan las características de las animaciones, los algoritmos, tendencias actuales y las técnicas necesarias para llevarlas a los entornos virtuales. En el segundo capítulo **Soluciones Técnicas**, se exponen las características que presentará el sistema como solución a los problemas planteados. En el tecero, **Descripción de la solución propuesta**, se analiza con mayor profundidad el objeto de estudio, se crea el modelo del dominio, se hace la captura de requisitos y se definen los modelos de casos de uso del sistema. En el cuarto, **Diseño e Implementación del sistema**, se presentan los diagramas de clases de diseño, de secuencia y los diagramas de componentes. Finalmente, se ofrece un glosario de abreviaturas y otro de términos para ayudar a la comprensión del lenguaje técnico utilizado en el trabajo.

1

Capítulo 1: Fundamentación Teórica

Introducción

Un gráfico es capaz de representar y sintetizar conceptos que por escrito necesitarían mucho texto y quizás aún así quedase poco claro. Por extensión, se puede añadir que una imagen en movimiento vale por cientos de palabras y permite representar conceptos que necesitan de la variable tiempo para ser explicados y entendidos.

Las animaciones en la informática gráfica juegan un papel fundamental en las aplicaciones más modernas de realidad virtual. Estas son estudiadas por gran cantidad de especialistas con el propósito de perfeccionarlas y conseguir un realismo mayor en las diferentes ramas en que se utiliza.

En el presente capítulo se brinda una visión general de los aspectos relacionados con las animaciones, los conceptos necesarios para el estudio y clasificación de los mismos, así como la descripción de los principales conceptos asociados al dominio del problema y que son necesarios para entender la propuesta de solución.

1.1 La Animación gráfica.

Un concepto básico del cual se debe partir para comprender las nuevas tendencias de la gráfica computacional es la animación. Esta palabra puede interpretarse literalmente como ***dar vida*** (del griego “anemos” y del latín “animus”, vida). En el campo de la informática gráfica, la animación se define como la simulación de movimientos a través de secuencias de imágenes o fotogramas. [1].

Con ella (la animción) se consigue dar vida a las escenas, por tanto hace referencia a cualquier cambio que se produzca en la escena que se está dibujando y que tenga consecuencias visuales. [2] En el contexto de esta investigación se trabajará con una definición más específica:

“La animación es la generación, almacenamiento y presentación de imágenes que en sucesión rápida, producen sensación de movimiento”. [3]

Figura 2: Principio de la animación.

1.1.1 Desarrollo histórico

Con el avance del conocimiento humano, a través de la historia han resaltado momentos que marcan hitos en el desarrollo de la simulación de animaciones. Un estudio de persistencia de la visión en 1824 marcó el inicio de las investigaciones en este campo. Luego se inventó la "película fotográfica" en 1889 y más tarde apareció el "cine" en 1895. Algunos años más tarde se realizó la primera película de animación: "Humorous phases of funny faces" en 1906 y el personaje animado "Félix el Gato" irrumpe 1913, aún en el cine mudo. Mickey Mouse en 1928 fue una película que contó con animación, y ya en 1964 se empleó por primera vez un ordenador en animación. [1]

Evolución de la animación por ordenadores

Los Primeros experimentos tuvieron lugar en la década de 1960 y principios de los 70 del pasado siglo, utilizándose sobre todo gráficos vectoriales (años 60-principio de los 70). Luego siguieron tres etapas delimitadas de la siguiente forma:

Comienzos (década de los 70)

- ✓ Primeras animaciones en universidades (Utah, UPenns., Ohio, NYIT).
- ✓ Hunger: 1ª película de animación por ordenador propuesta para un Oscar (Peter Foldes, Canadá, 1974): animación 21/2D, utilizando extensivamente técnicas de interpolación.
- ✓ Primeros usos en publicidad.

Desarrollo (años 80)

- ✓ Daniel y Nadia Magnenat-Thalmann: actores sintéticos.
- ✓ Evolución de hardware (VAX, Raster T., PCs) y software (Alias, Wavefront, TDI) permite la difusión de la animación por ordenador.
- ✓ Comienzan a aparecer películas de animación: Tron (82); Luxo (86); Tin Toy (88, ganadora de un Oscar).
- ✓ Comienzan a usarse efectos especiales realistas generados por ordenador (The last starfighter; 84: entornos espaciales).

Madurez (desde finales de los 80)

- ✓ Fomentado por el desarrollo de las técnicas de modelado y animación y por el abaratamiento del hardware y el software.
- ✓ Utilización generalizada (especialmente a partir de Parque Jurásico, 1993).
- ✓ Gran difusión en otros medios (publicidad, TV, etc.).
- ✓ Actualmente existen dos extremos: animaciones en tiempo real y las animaciones hechas en preproceso.

1.1.2 La ilusión óptica

La animación se consigue gracias al fenómeno fisiológico de la persistencia de la visión. Los estímulos de una imagen permanecen en la retina entre 100 y 200 milisegundos. Si las imágenes que estimulan la retina son muy parecidas entre sí, podemos aprovechar este fenómeno para generar la ilusión del movimiento. [4].

Según su cadencia, cinco imágenes por segundo se considera una sucesión de imágenes estáticas, esto quiere decir que el ojo humano no es capaz de percibir un movimiento fluido. Cuando la sucesión cuenta con una cadencia de 16 imágenes por segundo, se puede apreciar un movimiento contínuo. Se considera una buena cantidad de cuadros por segundos generalmente los valores superiores a 20. En el cine usualmente se utilizan 24, mientras que en la televisión se utilizan 30. [4]

Figura 3: Secuencia que provocan sensación de movimiento.

1.1.3 Principios de la animación

Las animaciones presentan principios de los que se valen los creadores para lograr efectos con el propósito de hacer las animaciones más fluidas y llamar la atención de los que las observan. Entre los principios de la animación se encuentran los siguientes:

Difuminado del movimiento: Si las imágenes no son refrescadas con la suficiente velocidad se produce lo que se conoce como efecto estroboscópico, o sea: las imágenes parecen que se suceden a saltos. Para evitar este fenómeno cada fotograma es una interpolación entre una imagen y la siguiente. [5]

Anticipación: Preparación de la siguiente acción (por ejemplo, antes de salir corriendo, forzar una postura en el sentido contrario). La anticipación permite preparar los músculos para la acción, preparar al espectador y llamar la atención hacia la acción principal e indicar la velocidad de la acción. [5]

Estiramiento y compresión: Para conseguir movimientos fluidos y sensación de elasticidad se usan deformaciones como el estiramiento y la compresión. Los objetos se deforman en la dirección de desplazamiento para dar sensación de peso y gravedad. La deformación es perpendicular a la trayectoria en los impactos. La regla básica consiste en mantener el volumen constante de los objetos. Con estas deformaciones se evita el efecto estroboscópico de forma semejante al difuminado por movimiento. [5]

Solapamiento y continuación de las acciones: El solapamiento consiste en comenzar la siguiente acción antes de terminar la anterior (por ejemplo, para abrir una puerta, el personaje al acercarse va estirando la mano antes de llegar). La continuación plantea que los movimientos no se detienen bruscamente, sino que continúan más allá de su posición final (por ejemplo, al golpear un pelota con una raqueta, esta continúa su movimiento por inercia mucho después de haber golpeado a la pelota). [5]

1.2 Tipos de animación.

Existen varios tipos de animación según su origen, forma de realización o producto final que se desea. A lo largo de su evolución se han desarrollado muchas técnicas y métodos que marcan las especificidades de cada una de ellas.

1.2.1 Animación clásica.

Consiste en la generación de una secuencia de imágenes por métodos pictóricos, formada a través de píxeles, asignada manualmente o semiautomáticamente por mecanismos sencillos guiados manualmente. [2]. Estas técnicas requieren de mucho tiempo de desarrollo para lograr animaciones fluidas.

1.2.2 Animación por ordenadores.

Es la creación por medio de un proceso automático, a partir de una representación de los objetos que forman parte de la escena y de su movimiento.

La animación por ordenadores permite representar objetos que evolucionan; los cuales en el tiempo cambian su posición, rotación, forma, textura, iluminación, etc. Los objetos pueden estar representados en una escena bidimensional lo cual tiene un resultado similar a la animación clásica, o pueden representarse en escenas tridimensionales en la cuales se pueden aplicar métodos realistas de representación.

La animación por ordenadores se basa en una unidad llamada *frame* (marco o fotograma) que no es más que cada imagen estática que forma la secuencia animada. [4]

1.2.3 Animación 3D por ordenadores

Lo realmente novedoso de la animación 3D por ordenadores es que permite la creación de escenas tridimensionales y visualizar la misma escena desde diferentes puntos de observación.

En este tipo de animación son posibles adicionar elementos y métodos que hacen las escenas mucho más reales como son: el enfoque de las cámaras, la presencia de luces, sombras y la incorporación simulación física, etc. [2]

La animación 3D por ordenadores generalmente requiere de computadoras potentes por el alto consumo de memoria gráfica y cantidad de cálculos necesaria para representar las escenas. Por tanto los métodos utilizados deben ser lo suficientemente eficientes para garantizar la velocidad del procesamiento y lograr animaciones fluidas.

1.2.4 Animación en tiempo real.

La animación en tiempo real es aquella en la que se van generando los fotogramas a medida que son necesarios, o sea, no se tiene previamente la información de cada fotograma, sólo se tiene la de los fotogramas necesarios para, mediante cálculos, generar aquellos cuadros de los cuales no se tiene la información completa. Los cambios en la escena pueden estar dados por la interacción con elementos externos, por ejemplo cuando se indica a través de algún periférico la dirección del movimiento u otra acción a realizar. [2]

1.3 Transformaciones geométricas.

Las transformaciones geométricas permiten modificar atributos de los objetos representados en entornos sintéticos; como son la traslación, la rotación y la escala. Estas transformaciones están regidas por ejes de coordenadas, en el caso de los entornos 3D se utilizan 3 ejes (x, y, z).

1.3.1 Representación vectorial de las transformaciones.

Los vectores en la programación gráfica son estructuras muy utilizadas por sus facilidades para el cálculo geométrico. En un espacio tridimensional se usan vectores de tres componentes para la manipulación de los objetos o de sus vértices. Los vectores son usados para representar magnitudes físicas y matemáticas como posición, rotación y escala, entre otras. Estos (los vectores) pueden estar contenidos en matrices que agrupan datos similares.

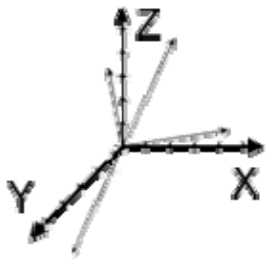


Figura 4: Vectores en un espacio tridimencional.

$$\begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figura 5: Vectores de traslación y escala contenidos en matrices de transformación.

Representación matricial de las transformaciones.

Las matrices son usadas en los sistemas de RV para el trabajo con las transformaciones geométricas, permiten agruparlas y combinarlas.

Las transformaciones de un cuerpo son el resultado de las transformaciones de cada uno de sus vértices. Esta operación se hace multiplicando las matrices de transformación por cada uno de los vértices representados como matrices columnas, obteniendo así los valores del los nuevos vértices. [6]

Combinación de transformaciones.

Las transformaciones más complejas se logran a través de concatenación de otras más sencillas con lo que se obtiene una matriz única llamada SRT-transform (escalar-rotar-trasladar). [7].

Existen transformaciones que son conmutativas y otras que no. A continuación se muestran clasificadas en grupos.

Entre las transformaciones conmutativas encontramos: [7]

- ✓ Traslación – traslación.
- ✓ Rotación – rotación.
- ✓ Escalado – escalado.
- ✓ Escalado uniforme - rotación.

Transformaciones no conmutativas:

- ✓ Traslación – escalado.
- ✓ Traslación – rotación.
- ✓ Escalado no uniforme – rotación.

1.3.2 Representación de las transformaciones mediante Quaternions.

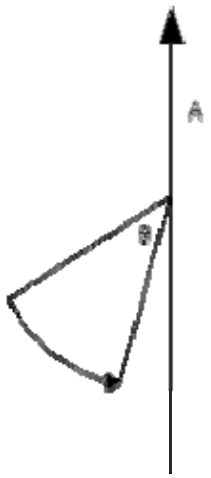
Los *Quaternions* pueden estar formados por un vector de 4 dimensiones (x, y, z, w) o un escalar y un vector 3D (w, v) donde v = (x, y, z) y w es un escalar. [8]

Los *Quaternions* son estructuras muy convenientes para representar rotaciones 3D. Estos posibilitan multiplicaciones más rápidas que las matrices y la interpolación de *Quaternions* permite rotaciones suavizadas. [8]

Las matrices de 3 filas por 3 columnas (3x3) pueden ser representadas a través de conversiones simples a un *Quaternions*. También un *Quaternions* y un vector de traslación pueden ser representados por una matriz de 4 por 4, agrupando la traslación y la rotación sin pérdida de información en ninguna de estas conversiones. [8]

Significado de las componentes del *Quaternions*:

Los valores x, y, z determinan el eje donde tiene lugar la rotación y w determina de cuánto es la rotación en dicho eje.



Q = (s XA, s YA, s ZA, C)

S = sen (Θ/2)

C = cos (Θ/2)

Θ es el ángulo que se desea rotar en sentido contrario a las manecillas del reloj.

Figura 6: Rotación sobre un eje.

Concatenación de las rotaciones mediante *Quaternions*.

Para representar más de una rotación con un único *Quaternions* se debe hacer una multiplicación de todas las rotaciones. En el caso que se quiera aplicar las rotaciones q1 y q2 (*Quaternions* unitarios). Para que el resultado de la rotación q1 le sea añadida la rotación q2 se debe multiplicar en el orden q2 * q1 (la multiplicación de *Quaternions* da como resultado otro Quaternions), esta operación no es conmutativa por lo que no podemos cambiar su orden (q1 * q2 ≠ q2 * q1). [9]

Aplicaciones de los *Quaternions.*

Sus aplicaciones incluyen animaciones de esqueletos, de cinemática inversa y generalmente está presente en los sistemas de RV relacionados con la física. Su uso es muy importante en el campo de las animaciones porque permite implementar el suavizado de movimientos por sus fáciles y económicas formas de interpolación. Los *Quaternions* pueden ser interpolados de varios modos según el resultado que se desee. Existen fundamentalmente cuatro tipos de interpolaciones, la Lineal. (*Lerp*), Esférica lineal (*Slerp*)**,** Esférica cúbica (*Squad*) y por *spline*. [8]

1.4 Leyes Físicas.

Para lograr altos niveles de realismo en las animaciones debemos acercar estas a la naturaleza y a sus leyes. Mediante estas leyes se agregan parámetros reales a los movimientos particularmente sobre el campo de la dinámica y la cinemática.

1.4.1 Cinemática directa.

Es la posibilidad de mover algunas de las "piezas" de un personaje o montaje 3D, actuando sobre un punto y produciendo un movimiento sobre su eje o centro de rotación (por ejemplo, mover el brazo fijada la rotación sobre el hombro. El programa de animación 3D genera con fórmulas geométricas simples, todos los movimientos necesarios de las partes ligadas a ella. En este caso, en la jerarquía de movimientos o giros definida, se parte de un eje fijo más importante (por ejemplo, el hombro) para mover elementos más sencillos (por ejemplo, el brazo). [5]

1.4.2 Cinemática inversa.

Es la posibilidad de que, moviendo elementos más sencillos en la jerarquía, el programa interpola el resto de articulaciones o puntos de giro, que pueden ser configurados por el animador, para conseguir que se muevan acorde a eso. Este tipo de movimiento es mucho más interesante pero a la vez más complejo, ya que en general no hay un sólo modo de rotar los elementos entre sí para conseguir seguir el movimiento final que pretende el usuario. Por ejemplo, un codo puede girar en un sentido, pero no en otro. Por ello pueden configurarse márgenes de rotación que indiquen al software qué límites tiene a la hora de elegir entre unos movimientos u otros. [5]

1.4.3 La Dinámica.

La dinámica estudia el movimiento teniendo en cuenta las fuerzas que lo producen. Se puede obtener gran realismo, pero resulta difícil especificar la animación. Hay que tomar en consideración masas, aceleraciones, grados de libertad, restricciones al movimiento, movimientos prioritarios y otras propiedades físicas. La dinámica de los cuerpos rígidos articulados es más sencilla que la de los cuerpos deformables. Se distinguen:

Dinámica directa: A partir de las masas y fuerzas aplicadas, se calculan las aceleraciones. [3]

Dinámica inversa: A partir de las masas y aceleraciones, se calculan las fuerzas que hay que aplicar. [3]

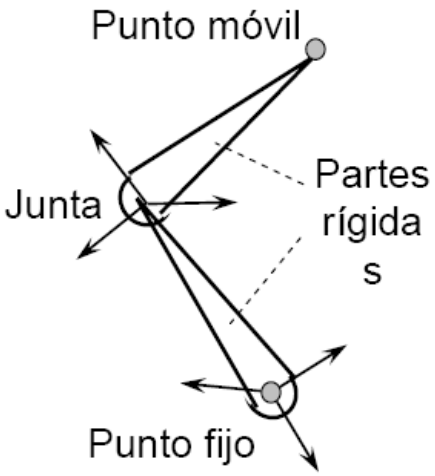


Figura 7: Cuerpo articulado para aplicar la Dinámica directa e inversa.

1.5 Formas de animación por ordenadores.

Se puede apreciar una animación como resultado de cambios de diferentes tipos de un fotograma a otro, los cambios en los objetos pueden ser producto de que su geometría tome otra forma, sus texturas o posición cambien. Por ejemplo animación por deformación, animación de textura y *morphing*.

1.5.1 Animación a lo largo de trayectorias.

Son de gran utilidad los recorridos a través de caminos para objetos móviles tanto sobre una superficie (ejemplo: vehículos de transporte terrestre) o para objetos que se encuentren sumergidos o sin apoyo (ejemplo: aviones, proyectiles, torpedos, etc.). Las trayectorias que describen estos objetos pueden estar predefinidas en su totalidad y ser seguidas estrictamente o pueden ser modificadas cambiando la información espacial de los objetos en los distintos puntos de la trayectoria. Son apreciables estas variaciones de trayectoria en objetos que posean inteligencia artificial (ejemplo: autos dentro de una ciudad). [3]

Los cambios en las animaciones predefinidas pueden estar dados por modelos matemáticos asociados a los objetos que se están animando. Estos provocan a través de las modificaciones en parámetros físicos -como la velocidad u otros-, las desviaciones de la trayectoria original. [2]

1.5.2 Animación por deformación de geométrica.

Los métodos basados en la deformación geométrica modifican la forma de los objetos, haciendo cambios en sus vértices indirectamente, a través de puntos de control o ajustando parámetros en la función que define la superficie. Estos métodos son imprescindibles para aplicaciones donde generalmente se representen objetos blandos como en los simuladores quirúrgicos. [10]

1.5.3 Animación de objetos articulados.

La animación de objetos articulados es de las más usadas en las modernas aplicaciones de realidad virtual. Ella es utilizada en las animaciones donde se apliquen leyes físicas como la dinámica y cinemática, y actúa sobre cuerpos que están conformados por elementos conectados jerárquicamente como esqueletos o brazos mecánicos. [2]

1.6 Técnicas de animación.

Animación basada en fotogramas.

La animación basada en fotogramas es una de las más utilizadas. Para hacer una secuencia, se van filmando las imágenes fotograma por fotograma y luego estos se unen para formar la animación. Es posible formar bibliotecas de movimientos de cada parte del cuerpo de la animación para de esta forma combinarlas y hacer animaciones diferentes. [11]

KeyFramming.

El *keyframming* se refiere a establecer posiciones en puntos específicos de tiempo en una animación y la parte intermedia la obtiene la computadora por medio de la interpolación matemática. Es necesario hacer un *keyframming* para cada control en cada nivel de la jerarquía del modelo. [11]

Esta técnica conocida también como "animación por cotas" consiste en basar el movimiento en unos fotogramas fundamentales o claves ("*keyframes*") y luego dejar que el sistema genere automáticamente los fotogramas intermedios mediante métodos de interpolación. Es importante que las cotas sean representativas del movimiento para que la interpolación tenga suficiente información. Esta técnica está basada en los métodos de trabajo de la animación tradicional en la que los animadores más expertos dibujan los momentos fundamentales del movimiento (cotas o keyframes) y los animadores principiantes dibujan los fotogramas intermedios ("*inbetweens*"). [11]

Rotoscopiado

El rotoscopiado consiste en una forma más elaborada de keyframming, donde se captura un movimiento real, y se utiliza esa información para mover un diseño generado por ordenador. [3]

Wavelets

Significa "pequeñas ondulaciones". Esta técnica permite que en una sola imagen se compriman una gran cantidad de datos para que al acercarse a ella, se vayan viendo los detalles sin distorsión. [3]

Animación procedural.

Consiste en describir el movimiento de forma algorítmica. Hay una serie de reglas que controlan cómo se van modificando los distintos parámetros (como la posición o la forma) a lo largo del tiempo. Para movimientos sencillos (un péndulo o una rueda que gira) es una buena solución, pero para movimientos más complejos (una persona caminando, o una moneda que cae al suelo), resulta difícil conseguir buenos resultados. Hay algunas técnicas con resultados interesantes, como los sistemas de partículas o la simulación de movimientos grupales. [3]

Animación por esqueleto.

Consiste en deformar la malla de un cuerpo a través de una estructura jerárquica de huesos, a los que se les asocia un grupo de vértices. Es una técnica sumamente utilizada actualmente por los programadores, dada su rapidez para procesar y obtener excelentes resultados, y la posibilidad de incluir una gran cantidad de detalles en la animación de un personaje, desde las arrugas de la piel hasta la definición de los músculos de su cuerpo. [12]

Es usada en muchos juegos y cada vez se hacen más perfectas. Uno de los primeros juegos en usar esta técnica fue el Half-Life donde las criaturas tienen un movimiento más fluido y realista que en juegos anteriores. [8]

1.7 Sistema de animación por esqueleto.

Con el sistema de animación por esqueleto, para representar a un personaje y su animación, se necesitan junto a los datos de una malla 3D, una jerarquía de "huesos". Estos últimos son una serie de transformaciones jerárquicas que, como los huesos en el cuerpo humano, permiten la deformación de este. Usualmente, las estructuras huesos son representadas básicamente como matrices de transformación. Estas eliminan la necesidad de almacenar la posición de los vértices por cada fotograma de la animación, superando ya desde este punto a la animación basada en vértices. Otras ventaja de la animación por esqueleto es la suavidad del cambio de una animación a otra, además, como las animaciones son independientes a las mallas, en una escena una misma animación puede aplicársele a diferentes mallas. [12]

La deformación puede ocurrir dado que cada hueso según su posición lógica con respecto a la mallas, tiene influencia sobre una serie de vértices de la misma. Las últimas dos maneras que surgieron con el tiempo para mejorar la técnica de deformación que dan lugar a lo que podemos ver actualmente en los videojuegos y simuladores más avanzados del mundo son las que siguen, enunciadas en orden de su surgimiento: [13]

Stitching

Con esta técnica los vértices correspondientes a cada hueso son transformados aplicándoles la matriz de transformación del hueso para cada fotograma a mostrar. La deficiencia de esta técnica es que pierde suavidad la malla en las intersecciones de los huesos, pues al flexionarse una articulación del cuerpo los vértices más cercanos a esta -y que pertenecían a huesos diferentes- se alejan demasiado, lo que crea ángulos abruptos en la zona de la articulación y una sensación de rigidez. [13]

Skinning

Esta técnica es básicamente como la anterior pero soluciona su deficiencia. Los vértices pueden ser influenciados por más de un hueso. Para estos casos cada hueso tiene un valor entre 0 y 1 llamado *weight* (peso), que indica cuánta influencia tiene este hueso sobre el vértice. La sumatoria de los pesos de

diferentes huesos para un vértice es igual a 1. En este caso, para obtener un vértice transformado sería: [13]

VérticeFinal = Vértice * Matriz1 * Peso1 + Vértice * Matriz2 * Peso2 + …

+ Vértice * MatrizN * PesoN

Donde las matrices y los pesos pertenecen a cada hueso con influencia sobre el vértice. [14]

Figura 8: Malla de un brazo asociada a una estructura de huesos.

1.7.2 Jerarquía de huesos.

Los esqueletos utilizados para las animaciones poseen una jerarquía determinada que posibilita que la cinemática directa pueda ser utilizada correctamente. Todas las transformaciones que se le apliquen a un elemento de la jerarquía tendrán efecto también en sus hijos. [15]

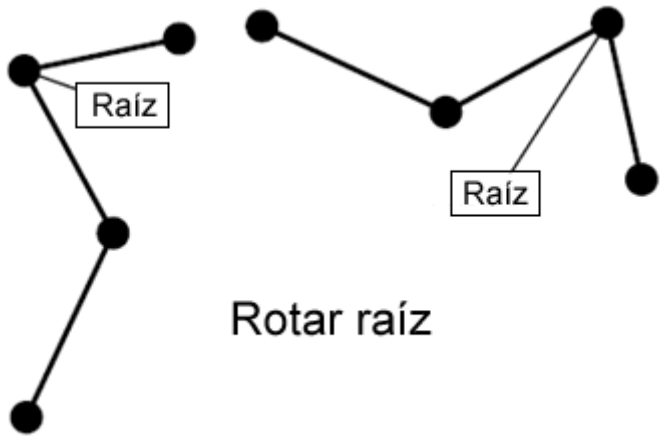


Figura 9: Jerarquía de elementos.

Para la representación de las figuras humanas, es un estándar utilizar una jerarquía de huesos similar a la del cuerpo humano real. En estas, las caderas son la raíz de la jerarquía y las extremidades los elementos más simples. [16]

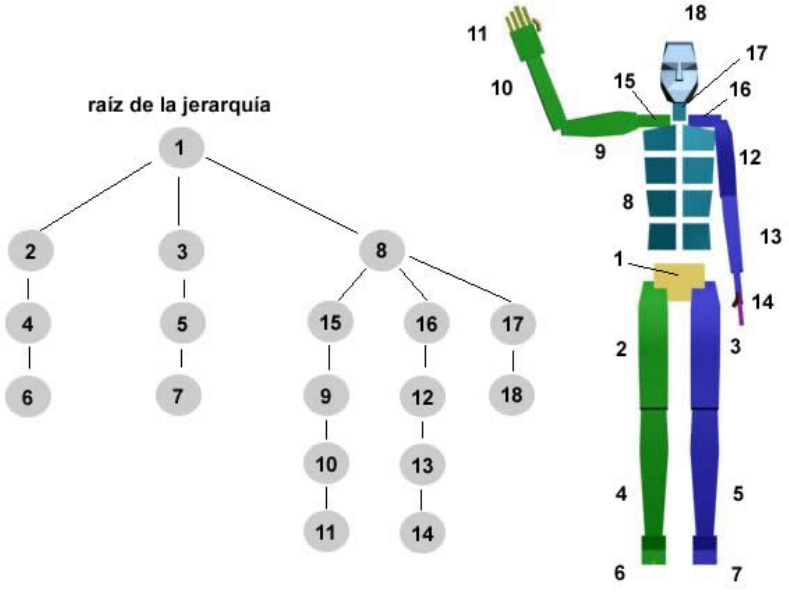


Figura 10: Estructura de huesos de un modelo de esqueleto modelo humano. [16]

1.7.3 Animaciones predefinidas

Las animaciones predefinidas son aquellas que se cargan desde un fichero, o sea, son las que están construidas con antelación por medio de otros softwares. Dichas animaciones son muy prácticas debido a su costo -relativamente bajo- en cuanto a cálculos. Además permiten evaluar con antelación el realismo de los movimientos. [17]

Animaciones desde el diseño

Existen varios softwares que tienen como función crear o editar animaciones y que soportan las animaciones por huesos. Dichos softwares les permiten a los diseñadores crear animaciones a su gusto, combinar otras y almacenarlas de forma que hagan mucho más rápida la realización de futuros trabajos donde se puedan reutilizar las mismas.

Entre los softwares que permiten el trabajo con animaciones utilizando la técnica de huesos se pueden encontrar algunos como: 3D Studio Max [16], LightWave [18], Maya [19], Kaydara Motion Builder [20], Blender [21], entre otros.

Animaciones desde la captura.

Se busca cada vez más realismo en los movimientos, y la forma de hacer esto con mejores resultados es a través de la Captura de Movimiento (MOCA). Estas técnicas consisten en atrapar los movimientos de actores mediante herramientas que pueden ser diversas, dependiendo de la técnica que se utilice para obtener la animación final que sea aplicable a un modelo 3D. [22]

Tipos de sistemas de captura.

Sistemas ópticos de captura de movimiento: en ellos se utilizan cámaras para rastrear el movimiento de marcadores acoplados a las articulaciones del cuerpo. El movimiento captado por las diferentes cámaras es combinado por el método de triangulación para calcular sus posiciones fotograma a fotograma en el espacio 3D. Los sistemas sencillos o duales de cámaras son adecuados para la captura facial, en tanto, un sistema de 3 a 16 o más cámaras son los utilizados para realizar la animación del cuerpo completo.

Sistemas magnéticos de captura de movimiento: Consisten en el uso de un transmisor centralmente localizado y un grupo de receptores adheridos a partes del cuerpo del actor. Estos receptores son capaces de medir su relación espacial con el transmisor central. Cada recibidor es a su vez conectado a una interfaz que puede ser sincronizada. [22]

Sistemas Electro-Mecánicos de captura de movimiento: El actor se coloca una armadura humana de piezas metálicas (similar a un esqueleto). Esta armadura es enganchada hacia la parte trasera de las articulaciones del actor, la armadura posee sensores que van exactamente en las articulaciones del actor para de esta forma sentir las rotaciones que realizan las mismas. Otros tipos de sistemas mecánicos utilizan guantes o modelos articulados. [22]

Figura 11: Trajes para sistemas captura de movimiento.

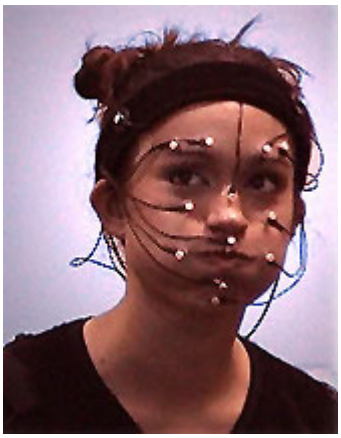


Figura 12: Sensores para la captura de movimientos faciales.

Formatos de captura de movimiento.

Entre los muchos formatos utilizados para almacenar datos de las animaciones se encuentran (siendo los más utilizados) el BVH, CSM y el BIP. Estos formatos son los utilizados por las herramientas más potentes conocidas por los especialistas de las animaciones. Son fáciles comprender y esto los hace muy populares. [22]

BVH: Es un formato de fichero ASCII que es usado para importar los datos de la rotación de las articulaciones desde varios sistemas de captura de movimiento. Este fue desarrollado originalmente por Biovision, una compañía que se dedica a los servicios de captura de movimiento. Es un excelente formato, pero carece de información explícita sobre cómo dibujar los segmentos, pero esto no entorpece la definición del movimiento. [20]

BIP: Es el formato nativo de *Character Studi*o**.** Este contiene la información de la longitud de cada hueso así como su rotación. [16]

CSM: Es un formato defichero ASCII utilizado para importar el dato de la posición de marcadores desde varios sistemas de captura de movimiento. [16]

1.8 Manipulación de Animaciones.

En ocasiones se necesita representar determinados movimientos de los cuales no se posee la animación correcta, lo cual hace necesario convertir las animaciones existentes en las necesitadas. Para lograr obtener las animaciones deseadas se pueden aplicar varias técnicas que permiten hacerle cambios a las animaciones predefinidas. Dichas animaciones se pueden trabajar por separado haciendo variaciones a las transformaciones de los elementos del esqueleto, basadas en parámetros ajenos a la animación. También se pueden transformar las animaciones que vienen desde formatos de capturas u otros ficheros combinando más de una animación. [2]

1.8.1 Transiciones de animaciones.

Los distintos personajes que intervienen en escenas virtuales reproducen varias animaciones dependiendo de las actividades que se quieran representar a través de estos. Con el propósito de que las animaciones puedan tener un orden variable, no sean específicas para un sólo modelo, sean reutilizables y más cortas entre otras necesidades, se emplean las transiciones de animaciones. Esta técnica posibilita cambiar las animaciones que están reproduciéndose todas las veces que se necesite, permitiendo con animaciones cortas, lograr largas secuencias concatenando en tiempo real dichas animaciones. También muchos SRV implementan grafos de secuencias de animación los que determinan a cuáles animaciones se pueden cambiar después de la que se está utilizando. [16]

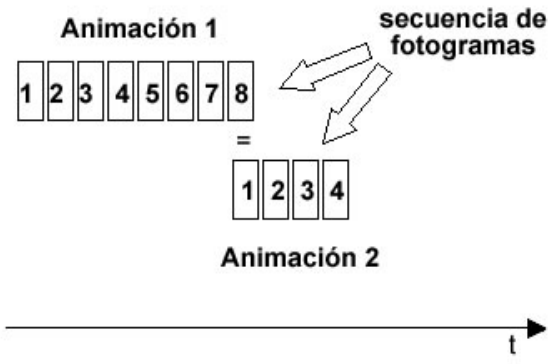


Figura 13: Proceso de transición de animaciones donde el final de la primera debe coincidir en el principio de la segunda.

1.8.2 Interpolación de animaciones.

La interpolación es un método utilizado en muchas ramas de las ciencias de la computación con el fin de obtener valores numéricos intermedios de cualquier tipo, ya sean de temperatura, altitud, vectores de velocidad o posiciones 3D. [2]

En las animaciones, la interpolación es utilizada para suavizar movimientos. Se puede suavizar el cambio de la posición o rotación a un modelo y también de cada uno de sus elementos. En el caso de los personajes, estos elementos son los huesos.

Existen varios tipos de interpolaciones según el resultado que se espere y los datos que se posean. En específico, para las animaciones se utilizan las interpolaciones de los *Quaternions* por ser estructuras que permiten representar matrices. Los *Quaternions* permiten interpolar de diferentes formas útiles para la animación, como son la interpolación lineal, la interpolación esférica lineal y la interpolación esférica cúbica. [8]

Cuando se combinan las transiciones de animaciones con la interpolación de las mismas, se logran transiciones suavizadas. De este modo se aprecian transiciones sin saltos aunque la animación a la que se le haga la transición no comience igual a como termina la que se este reproduciendo. De otra manera, seria necesario tener todas las combinaciones intermedias entre las dos animaciones que se quieran reproducir consecutivamente. Otra ventaja es que no es necesario llegar al final de la animación para comenzar la siguiente, ya que no es preciso que los fotogramas coincidan para evitar el salto (último fotograma de la primera animación con el que comienza la segunda animación). [16]

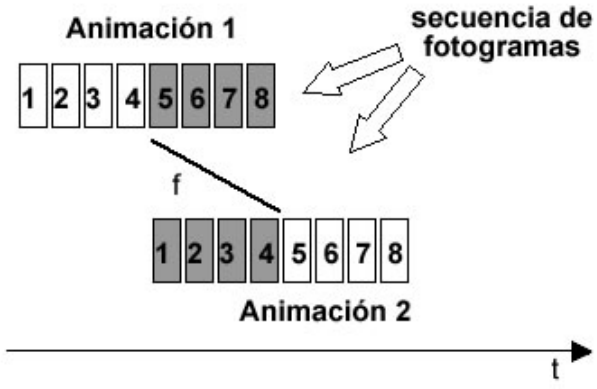


Figura 14: Proceso de transición con interpolación de animaciones. Se interpolan los últimos fotogramas de la primera animación con los primeros de la segunda.

1.8.3 Animaciones por capas.

Las animaciones por capas son implementadas por varios softwares de edición de animaciones. Esta utilidad posibilita adicionarle a una animación los movimientos de otra animación o de varias. La animación resultante es la suma de las transformaciones almacenadas en cada animación previamente cargada. Las animaciones por capas permiten que de movimientos sencillos se obtengan otros más complejos y específicos necesarios en muchas escenas, sin tener cada una de estas animaciones previamente generadas desde el diseño. [16]

1.8.4 Mezcla de animaciones.

La mezcla de animaciones es muy similar a las animaciones por capas, porque también se obtiene una animación final como resultado de la suma de las transformaciones almacenadas en otras más sencillas. En esta se permite adicionar parámetros que especifiquen cuáles partes del esqueleto se afectarán por cada animación. [16]

Al mezclar animaciones se obtienen buenos resultados si las predeterminadas que se eligen y la forma de mezclarlas son las correctas. Por ejemplo: a determinado personaje se le aplica una animación donde esté corriendo y otra donde esté disparando, el resultado sería que corra y dispare al mismo tiempo sin tener esta animación en un formato de captura.

1.9 Herramienta “*SceneToolkit*”.

La STK es la primera herramienta y una de las producidas por el proyecto de Herramientas de Desarrollo para Sistemas de Realidad Virtual. Es una herramienta para la visualización de escenas tridimensionales en tiempo real. La misma está en fase de desarrollo, pero actualmente permite la producción de simuladores y juegos con relativa facilidad. Esta es capaz de incorporar distintas funcionalidades a los objetos insertados en las escenas que representa, como autos, árboles, terrenos, etc. [23]

La herramienta está dividida en 3 capas fundamentales: una capa *Engine* donde se encuentra todo el procesamiento importante de carga y manejo de objetos; una capa *Renderer* donde se define con qué librería gráfica dibujar los entornos (OpenGL y DirectX por el momento), y que permite añadir nuevas librerías por parte de los usuarios de la herramienta en caso de que lo necesiten; y una capa *Application* donde se hace el manejo de eventos del SO específico, y con la que el usuario de la herramienta puede desarrollar su propia interfaz para cualquier SO. [23]

La STK tiene como estructuras fundamentales Nodos y Controladores, esto se muestra gráficamente en la siguiente imagen.

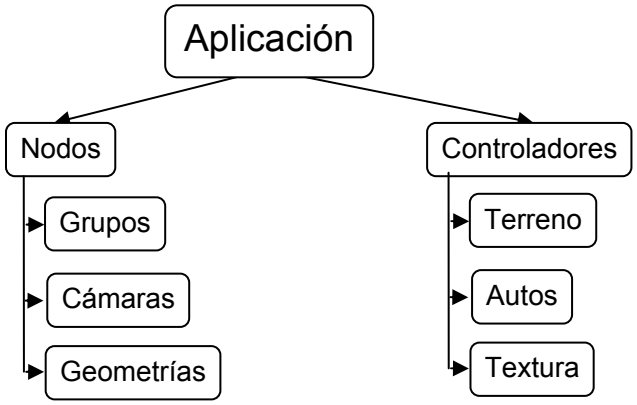


Figura 15: Estructuras fundamentales de la STK.

Conclusiones.

En el capítulo anteriormente presentado se hizo un estudio sobre el objeto de la investigación tratada, y quedaron enunciados importantes conceptos que están estrechamente relacionados con las animaciones en tiempo real. Tambien se expusieron algunas de las técnicas utilizadas para lograr animaciones con altos niveles de realismo.

En el caso especifico de la animación por ordenadores, se aplicó el método histórico-lógico para abordar las diferentes etapas de su evolución. En el estudio aplicado a las técnicas, la investigación se concentró en las animaciones por esqueletos dados sus aportes a las animaciones de personajes.

Al final del capítulo se explican algunas de las formas de manipulación de animaciones que brindan los esqueletos.

2

Capítulo 2: Soluciones Técnicas

Introducción

A continuación, se exponen las soluciones técnicas con el proposito de satisfacer los objetivos del trabajo. Para ello se ha realizado un amplio análisis del los métodos vistos en primer capítulo, seleccionándose los óptimos para resolver el problema.

2.1 Manipulación de animaciones.

A partir de los conceptos y técnicas estudiadas en la fundamentación teórica, se propone como solución al problema planteado el desarrollo de un módulo que facilite el manejo de colecciones de animaciones en tiempo real. Se propone realizar este módulo mediante la modelación e implementación de las siguientes funciones:

Animación a través de trayectorias: esta función permite a los personajes recorrer caminos en tres dimensiones. Además debe permitir definir con qué nivel suavizado se desea recorrer los vértices de dichas trayectorias. Para lograrlo se necesitará de caminos definidos para los elementos que recorrerán los mismos. Los caminos pueden contar con especializaciones que posibilitarán definir el rango de velocidad con el que se pueden transitar en su totalidad o por partes de él.

Transición de animaciones: Está basada en las técnicas de animación por esqueleto. Permite el manejo de colecciones de animaciones predefinidas de forma que permita transitar de una animación a otra, posibilitando que un esqueleto reproduzca distintas animaciones mientras es visualizado. Así los personajes podrán representar distintos comportamientos según se desee, haciéndose estos cambios en tiempo real.

Interpolación de animaciones: Permite que -utilizando colecciones de animaciones y principios de las transiciones- se obtengan transiciones suavizadas mediante interpolación, haciendo que los cambios de animaciones de los personajes no tengan saltos perceptibles. Para que las transiciones sean suavizadas se utilizarán *quaternions* por la capacidad de estos para interpolarse de varias formas, siendo este un método muy efectivo para hacer transiciones suavizadas con pocas animaciones intermedias. A las transiciones con interpolación se le podrán definir parámetros como con qué velocidad se quiere que ocurra el cambio de animación o cuántos fotogramas interpolar de las animaciones que intervienen en el proceso.

Mezclas de animaciones: esta técnica es una solución factible para hacer combinaciones de animaciones refinadas. Dichas mezclas permiten obtener animaciones finales como producto de otras predefinidas, las cuales pueden aportar en cierto grado a la animación resultante según se defina.

Posibilita hacer más específicas las combinaciones, definiendo cuáles regiones del cuerpo se desean afectar con nuevas transformaciones de otras animaciones. De esta forma se obtiene gran variedad de animaciones que no se tienen desde el diseño, sólo en tiempo real.

Cálculo de velocidad de traslación: Cuando los personajes corren o caminan deben hacerlo a la velocidad adecuada en correspondencia con la animación que reproducen para evitar la impresión de patinaje sobre la superficie en que se encuentran. Esto sugiere que el módulo a desarrollar debe hacer los cálculos necesarios para determinar la velocidad a la que se moverán los personajes al reproducir este tipo de animaciones.

Observar objetivos: En muchas actividades los personajes deben mirar a un lugar determinado de la escena, independientemente de la animación que reproduzcan. Dicho problema debe ser solucionado por el módulo a desarrollar, para que se pueda además especificar los grados de libertad de la cabeza.

Roles: Posibilitan definir cuáles animaciones pertenecen a cada personaje así como qué animación reproducirá cuando está inactivo. De esta forma puede definirse el comportamiento del personaje diferenciándolo por las animaciones que le corresponden a cada uno y garantizando así que no reproduzca movimientos no deseados. Ejemplo de la utilidad de esto se ve en aplicaciones de RV donde existen personajes de diferentes edades, habilidades u otros atributos que influyan sobre el comportamiento de los mismos y por lo cual deban reproducir sólo una serie de movimientos.

2.2 Colección de animaciones.

Para obtener un módulo con las funciones anteriormente enunciadas, como parte de la solución propuesta es necesario que en este se implementen una colección de animaciones. Esta característica es importante para las funcionalidades en las que intervienen más de una animación ya que posibilita que en el momento en que se utiliza otra animación no sea necesario cargarla y a la vez tener todas las animaciones que se quieran para obtener una resultante.

2.3 Grafo de transiciones.

Con las transiciones interpoladas se pueden hacer cambios de una animación a otra, de manera que no ocurran saltos en la transición aunque estas no estén previamente preparadas. Sin embargo, hay casos en que las animaciones no permiten la transición con movimientos naturales. Para estos casos es necesario conocer de cuáles animaciones se pueden pasar a otras. Este es un problema que el módulo debe solucionar implementando un grafo de animaciones donde, de cada una, se conozca a cuáles se pueden realizar transiciones. También el grafo puede contener otras informaciones importantes para las mezclas, para recorrer caminos y para las propias transiciones.

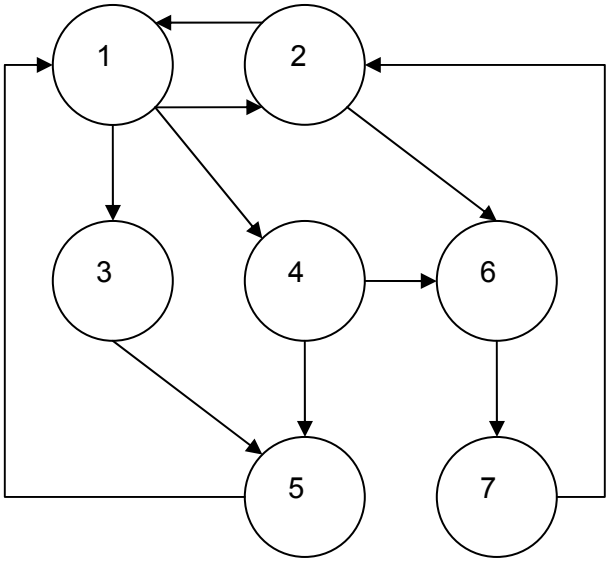


Figura 16: Grafo de animaciones.

En el grafo representado los nodos son las animaciones y las aristas significan las posibles transiciones además de contener la información de cuál es la velocidad apropiada para el cambio.

Conclusiones

Como resultado de las soluciones propuestas, el módulo a desarrollar debe cumplir con los objetivos trazados en el comienzo de la investigación. Debe ser capaz de manipular adecuadamente animaciones y colecciones de estas con el fin de obtener secuencias de animaciones de alto realismo para que los personajes puedan reproducir las adecuadas, según se les defina en tiempo real. Dichas soluciones harán que el módulo facilite la construcción de aplicaciones -como juegos y simuladores profesionales- donde los personajes sean figuras representadas constantemente en las escenas y tengan diversos comportamientos.

Con el capítulo presentado quedan trazadas las guías que servirán para realizar las siguientes fases del trabajo, teniendo las bases técnicas por las cuales se regirá el módulo encargado de las animaciones.

3

Capítulo 3: Descripción de la Solución Propuesta

Introducción

El propósito del presente capítulo es obtener una descripción de la solución propuesta de forma conceptual. En él se muestran los conceptos necesarios para el desarrollo de la solución, se describen las restricciones a considerar para modelar la misma, así como las capacidades que tendrá el módulo de animaciones, descritas en forma de casos de uso.

3.1 Reglas del Negocio

Con el propósito de definir las condiciones que deben satisfacerse para el correcto funcionamiento del módulo de animación se establecen como reglas del negocio las siguientes:

- ✓ Los personajes deben estar ubicados en el origen de coordenadas y con los pies sobre el cero del eje vertical o eje Z.
- ✓ La velocidad estándar debe ser 30 *fps*.
- ✓ Todos los huesos deben tener asociados al menos un vértice de la malla.
- ✓ Los huesos deben estar jerarquizados correctamente.
- ✓ Las animaciones deben ser compatibles con los esqueletos.
- ✓ Las animaciones cíclicas deben tener el último fotograma igual al primero.
- ✓ Las animaciones cíclicas para movimientos de caminar o correr u otros similares donde el esqueleto deba trasladarse a una velocidad determinada, deben ser sin traslación predefinida.
- ✓ Las relaciones entre los huesos y los vértices de la malla deben estar en el fichero PWX que es el utilizado por la herramienta.
- ✓ Los ficheros de animación BVH, BIP o CSM deben ser convertidos a ANX para cargarse adecuadamente.
- ✓ El personaje debe estar siempre animado. Si no está la animación por defecto no se carga como personaje.
- ✓ Debe existir la animación por defecto con el mismo nombre del personaje.

3.2 Modelo de Dominio

El modelo de dominio representado a continuación es un acercamiento a la solución propuesta, donde se modelan los principales conceptos así como sus relaciones con los que se trabajarán en el módulo a obtener.

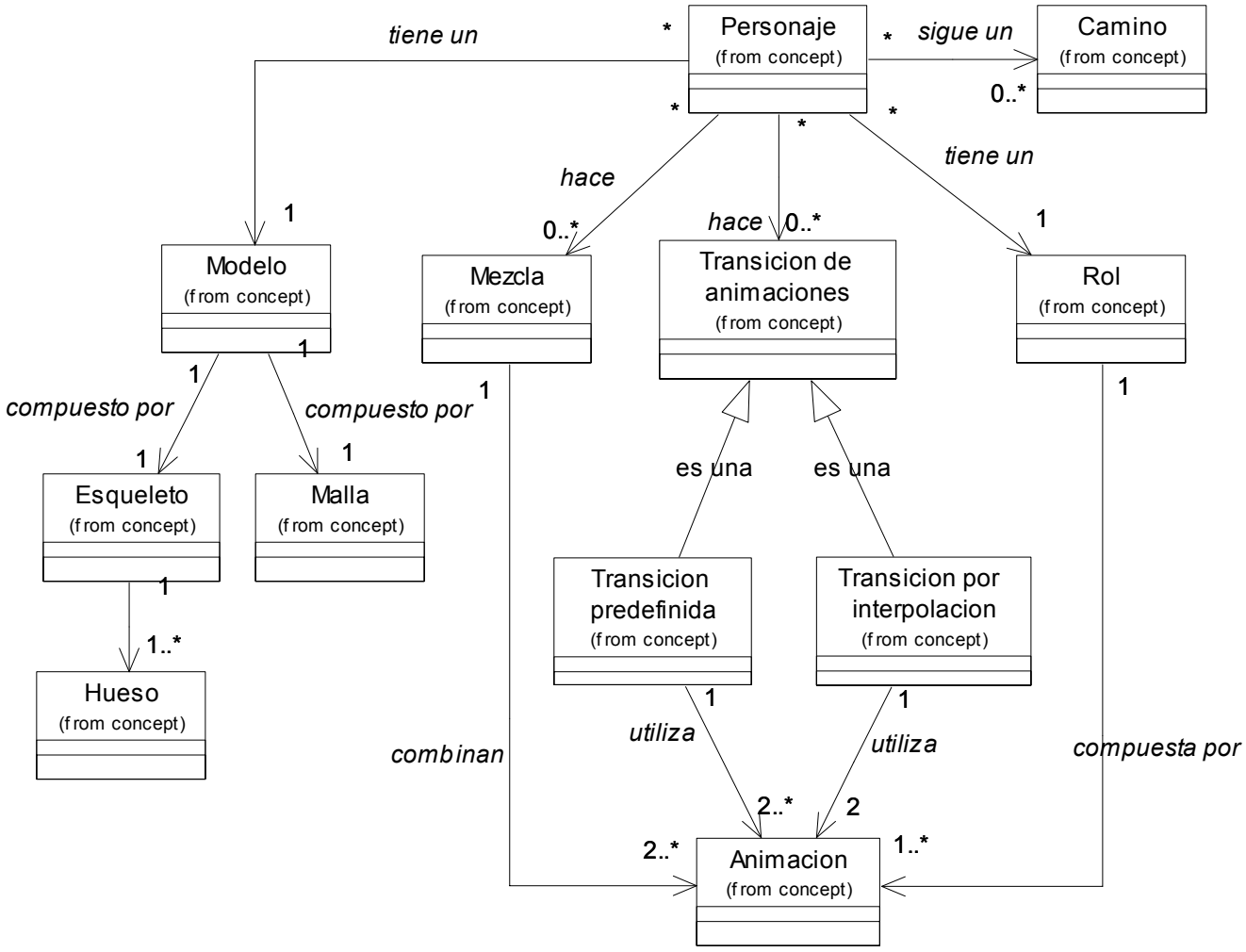


Figura 17: Modelo de Domino para módulo de animación de personajes.

3.3 Glosario de términos.

Es importante conocer qué significan los conceptos relacionados en el modelo, anterior por lo cual a continuación estos se especifican para facilitar la compresión del modelo dominio.

Personaje: actor de la escena de un mundo de realidad virtual, que soporta acciones (como un tipo de comportamiento), y que tiene entre sus atributos, cualidades físicas y emocionales que serán usadas a la hora de ejecutar las acciones, así como determinados roles

Modelo: Prototipo para la animación que relaciona la malla con el esqueleto.

Malla: Forma de representar un modelo a partir de polígonos. Colección de vértices, aristas y polígonos conectados de forma que cada arista es compartida como máximo por dos polígonos.

Esqueleto: Estructura jerárquica que agrupa los huesos de un modelo.

Hueso: Estructura con influencia sobre una determinada cantidad de vértices de la malla determinando en estos la posición a partir de la posición y rotación del hueso.

Camino: Colección de puntos a recorrer por el personaje.

Animación: Secuencia de transformaciones de cada uno de los huesos del esqueleto en una cantidad de fotogramas determinados a una velocidad dada.

Rol: Papel que trae consigo un grupo de acciones y que formará parte de los atributos de un personaje animado.

Transición: Cambio de la animación que el personaje reproduce.

Transición Predefinida: Es el cambio de animaciones donde no se modifican las transformaciones predefinidas almacenadas en las animaciones que intervienen en el proceso.

Transición por interpolación: Transición donde se hace un cambio de animación suavizando, modificando las animaciones predefinidas mediante métodos de interpolación.

Mezcla: Combinación de animaciones donde intervienen dos o más de estas, aportando en un determinado grado sus transformaciones a la animación resultante.

3.4 Captura de requisitos.

El módulo de animación de personajes tendrá que cumplir con una serie de condiciones y capacidades para dar solución a las deficiencias que presenta la herramienta para animar personajes. Se definen dichas condiciones y funcionalidades en los siguientes requisitos clasificados como funcionales y no funcionales.

3.4.1 Requisitos funcionales

Entre los requisitos funcionales que debe cumplir el módulo de animación encontramos:

1. **Gestionar roles**.
   1.1. Crear roles.
   1.2. Adicionar roles a la colección.
   1.3. Adicionar animaciones a un determinado rol.
   1.4. Buscar animaciones dentro de roles.
   1.5. Buscar roles en la colección.
2. **Configurar Personaje**.
   2.1. Crear estados para el control de las animaciones que reproduce el personaje.
   2.2. Establecer animación inactiva.
   2.3. Asignar Rol al personaje.
3. **Configurar transición de animación**.
   3.1. Especificar animación a la cual se va a cambiar.
   3.2. Definir si el cambio será suavizado o no.
   3.3. Definir si se esperará al fin de la primera animación para realizar el cambio
   3.4. Definir a qué fotograma de la próxima animación pasar.
4. **Configurar mezcla de animación**.
   4.1. Especificar animaciones a mezclar.
   4.2. Definir fotograma en que comenzará la mezcla.
   4.3. Definir cuánto aporta cada animación a la resultante.
   4.4. Habilitar mezcla.
   4.5. Deshabilitar mezcla.

5. **Configurar recorrido**.
   5.1. Definir recorrido.
   5.2. Definir velocidad.
   5.3. Definir posición inicial en el camino.
   5.4. Definir sentido en el camino.
   5.5. Nivel de suavizado de la trayectoria.
6. **Configurar capacidad de visión**.
   6.1. Definir punto a mirar.
   6.2. Configurar el rango de visión (por distancia al objetivo y grados de libertad de la cabeza).
   6.3. Habilitar orientación de la cabeza al objetivo.
   6.4. Deshabilitar orientación de la cabeza al objetivo.
7. **Calcular velocidad de traslación**.
   7.1. Buscar animación a la cual se le calculará la traslación.
   7.2. Buscar amplitud del paso promedio.
   7.3. Calcular distancia a trasladar por fotograma.
8. **Actualizar transición**.
   8.1. Reproducir dos animaciones simultáneamente.
   8.2. Interpolar transformaciones de dos animaciones.
   8.3. Aplicar para cada fotograma la transformación interpolada correspondiente a cada hueso.
9. **Actualizar mezcla**.
   9.1. Comprobar que la mezcla esté activada.
   9.2. Actualizar transformaciones de cada hueso según la animación asociada a ellos.
10. **Actualizar recorrido**.
    10.1. Comprobar que esté habilitado el camino.
    10.2. Calcular distancia a trasladarse.
    10.3. Identificar vector por el cual trasladarse.
    10.4. Hallar el próximo vector por el cual trasladarse.
    10.5. Calcular vector de la trayectoria suavizada.
    10.6. Calcular posición del personaje sobre el vector de la trayectoria suavizada.
    10.7. Calcular rotación del personaje sobre el vector de la trayectoria suavizada.

10.8. Ubicar personaje en la posición calculada.

10.9. Orientar personaje con la orientación calculada.

11. **Mirar objetivo**.

11.1. Comprobar propiedad de mirar objetivo esté activada.

11.2. Calcular dirección a la que se encuentra el punto a mirar respecto a la cabeza del personaje.

11.3. Comprobar que esté dentro del rango de visión definido por el personaje.

11.4. Rotar el hueso cabeza en dirección del punto a mirar.

11.5. Mover cabeza hacia el punto definido.

11.6. Determinar si está el objetivo dentro del campo de visión.

11.7. Restablecer orientación de la cabeza por defecto si el objetivo sale del rango.

3.4.2 Requisitos no funcionales

Entre los requisitos no funcionales que debe tener el módulo a obtener a partir de investigación realizada, la modelación e implementación se encuentran los siguientes:

Rendimiento: Como aplicación de tiempo real, debe tener alto grado de velocidad de procesamiento o cálculo, tiempo de respuesta y de recuperación, y disponibilidad.

Soporte: En una versión inicial deberá ser compatible con la plataforma Windows.

Legales: Se regirá por las normas ISO 9000.

Software: Sistema operativo Windows.

Hardware: Compatibilidad con tarjetas gráficas de la familia NVIDIA (Geforce 3, Geforce 4, FX 5200).

Diseño e implementación: Debe implementada en el Leguaje C++ estándar. Se regirá por la filosofía de Programación Orientada a Objetos.

Usabilidad: La aplicación debe ser implementada con el objetivo de ser reutilizado por otra aplicación similar.

3.5 Casos de uso del sistema

A partir de las funcionalidades que deben ser implementadas en el módulo de animación, definidas en el epígrafe “Captura de requisitos”, se hace el agrupamiento de las mismas y se llega como resultado a los casos de uso del sistema representados en las siguientes figuras.

Los casos de usos definidos son divididos en dos vistas fundamentales. Una primera vista muestra los casos de uso encargados de la inicialización y configuración y otra vista donde aparecen los casos de uso encargados de la actualización.

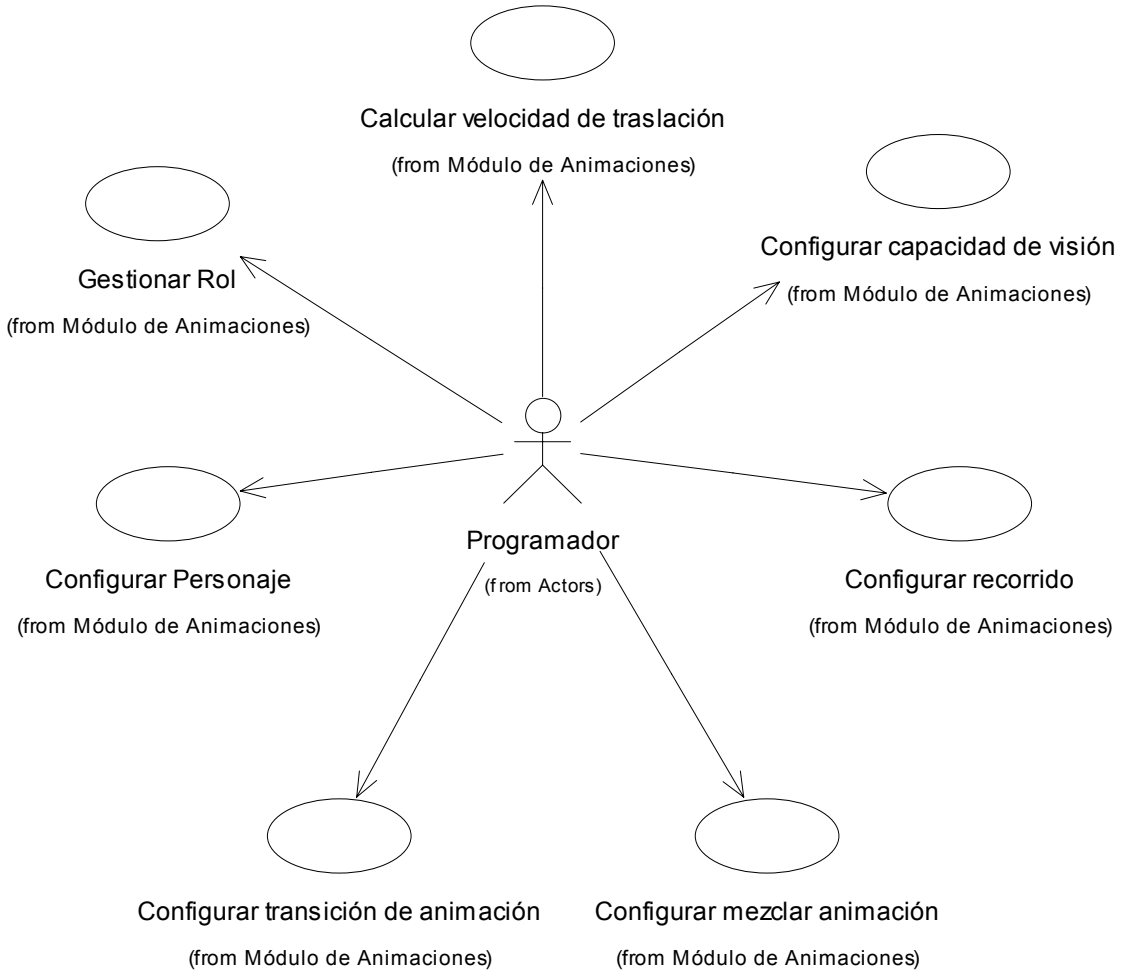


Figura 18: Casos de uso del sistema encargados de la Inicialización y Configuración.

En los casos de uso del sistema a implementar, representados en el diagrama anterior el programador inicializa y configura los personajes y las animaciones que ellos reproducirán.

Los casos de uso representados en el diagrama anterior son los encargados de la actualización de los estados geométricos del personaje a partir de animaciones. Los casos de uso ***Inicializar y actualizar escena*** *y* ***Actualizar animación*** pertenecen a la SceneToolKit y utilizan los demás casos de uso como extensiones, porque no son siempre utilizadas las animaciones en las escenas y en las que son usadas pueden no utilizarse todas al mismo tiempo.

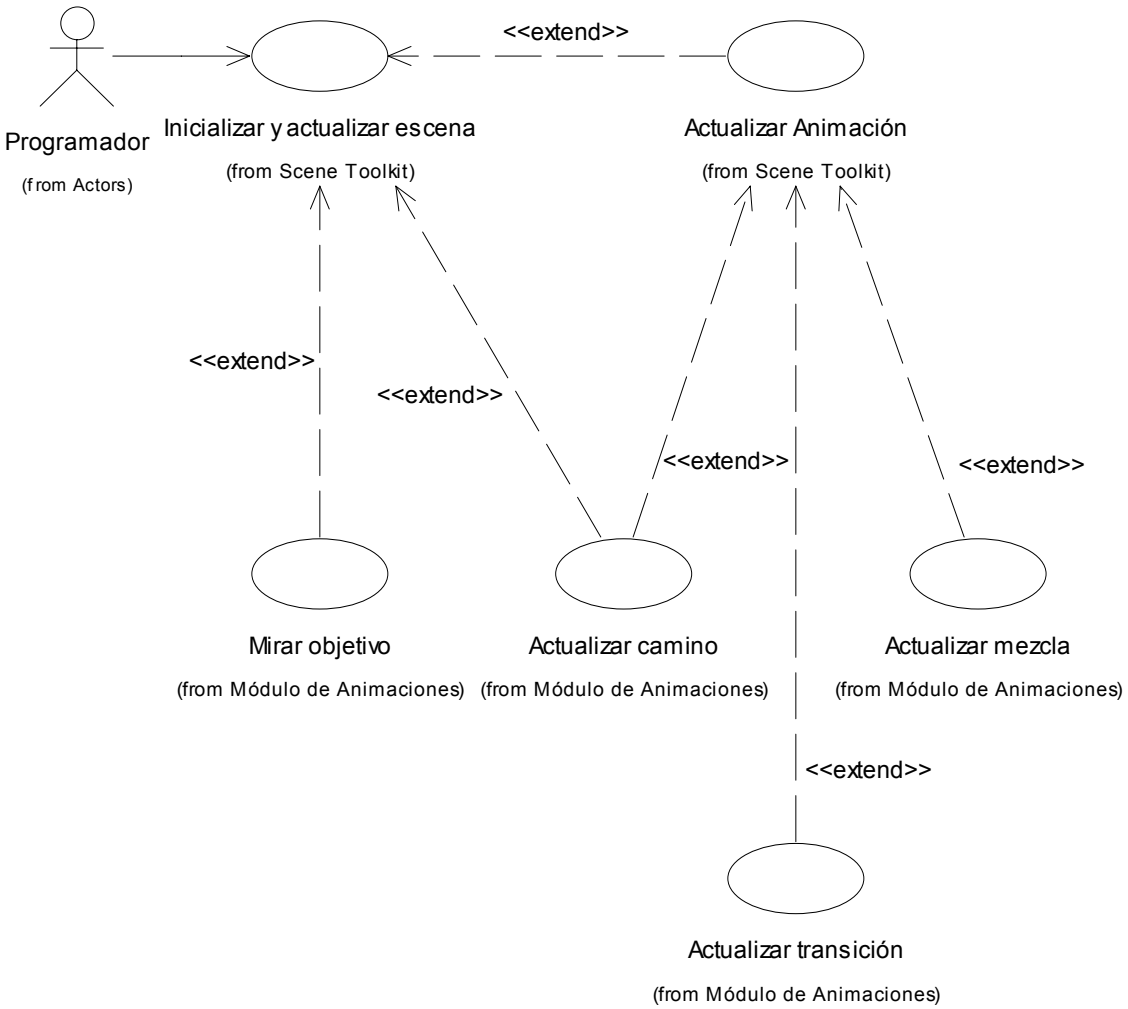


Figura 19: Casos de uso del sistema encargados de la actualización.

3.5.1 Definición del actor del sistema.

Actores	Justificación
Programador	Es quién se beneficia con las funcionalidades que brinda el módulo de animaciones e interactúa con todo el sistema.

Tabla 1: Definición de actores del sistema.

3.5.2 Casos de uso por ciclos de desarrollo

Primer ciclo de desarrollo

Entre las funcionalidades con las que el módulo de animaciones contará, se distinguen algunas como las más importantes, por lo que estas deben ser desarrolladas y optimizadas para el correcto funcionamiento de la herramienta. Esto hace que el trabajo se divida fundamentalmente en dos ciclos de desarrollo. Como resultado del primer ciclo se obtendrán las funcionalidades más importantes y aquellas que permitirán continuar al segundo ciclo posteriormente.

Tabla 2: Casos de uso correspondientes al primer ciclo de desarrollo.

Código	Nombre de caso de uso	Paquete	Justificación de la selección.
CU_1	Gestionar rol	Inicializar y Configurar Animación	Necesario para las funcionalidades básicas del módulo.
CU_2	Configurar personaje	Inicializar y Configurar Animación	Necesario para las funcionalidades básicas del módulo.
CU_3	Configurar transición de animación	Inicializar y Configurar Animación	Necesario para las funcionalidades básicas del módulo.
CU_4	Configurar mezcla animación	Inicializar y Configurar Animación	Necesario para las funcionalidades básicas del módulo.
CU_5	Configurar recorrido	Inicializar y Configurar Animación	Necesario para las funcionalidades básicas del módulo.
CU_6	Configurar capacidad de visión	Inicializar y Configurar Animación	Necesario para la posterior implementación del caso de uso mirar objetivo.

CU_8	Actualizar transición	Actualizar animación	Necesario para las funcionalidades básicas del módulo.
CU_9	Actualizar mezcla	Actualizar animación	Necesario para las funcionalidades básicas del módulo.
CU_10	Actualizar camino	Actualizar animación	Necesario para las funcionalidades básicas del módulo.

Segundo ciclo de desarrollo

Tabla 3: Casos de uso correspondientes al segundo ciclo de desarrollo.

Código	Nombre de caso de uso	Paquete	Justificación de la selección.
CU_7	Calcular velocidad de traslación	Inicializar y Configurar Animación	A pesar de que el módulo no cuente con esta funcionalidad es posible cambiar la velocidad de traslación de un personaje y así calibrar la misma.
CU_11	Mirar objetivo		Esta funcionalidad no es prioritaria para los productos que utilizan actualmente la herramienta.

3.5.3 Listado de casos de uso.

A continuación se describen brevemente los casos de uso, se especifica la prioridad y el autor de los mismos, así como el código y la relación con los requisitos funcionales para posteriormente describir con más detalles aquellos que forman parte del primer ciclo de desarrollo.

Tabla 4: Descripción del caso de uso Gestionar Rol.

CU_1	Gestionar Rol
Actor	Programador
Descripción	El caso de uso es iniciado por el actor, el cual puede invocar crear un rol especificando el personaje al cual pertenecerá y animación que tendrá el mismo como inactiva; o puede adicionarle una animación a un rol previamente creado.
Prioridad	Crítico
Referencia	1.1, 1.2, 1.3, 1.4, 1.5

Tabla 5: Descripción del caso de uso Configurar Personaje.

CU_2	Configurar Personaje
Actor	Programador
Descripción	El caso de uso es iniciado por el actor y en él se define cual animación reproducirá el personaje como animación inactiva o su rol.
Prioridad	Crítico
Referencia	2.1, 2.2, 2.3

Tabla 6: Descripción del caso de uso Configurar transición de animación.

CU_3	Configurar transición de animación.
Actor	Programador
Descripción	El caso de uso es inicializado por el autor y en él se especifica como será el cambio y a cual animación se pasará, validando los parámetros que definen la transición.
Prioridad	Crítico
Referencia	3.1, 3.2, 3.3, 3.4

Tabla 7: Descripción del caso de uso Configurar mezcla de animación

CU_4	Configurar mezcla de animación.
Actor	Programador
Descripción	El caso de uso es iniciado por el programador y en él se define como será la mezcla de las animaciones que el personaje reproducirá.
Prioridad	Crítico
Referencia	4.1, 4.2, 4.3, 4.4, 4.5

Tabla 8: Descripción del caso de uso Configurar recorrido.

CU_5	Configurar recorrido.
Actor	Programador
Descripción	El caso de uso es iniciado por el programador y permite especificar características de la trayectoria que recorrerá el personaje.
Prioridad	Crítico
Referencia	5.1, 5.2, 5.3, 5.4, 5.5

Tabla 9: Descripción del caso de uso Configurar capacidad de visión.

CU_6	Configurar capacidad de visión.
Actor	Programador
Descripción	En el caso de uso se definir cual es el punto de la escena donde el personaje mirará dentro de un área determinada especificando cuanto puede girar la cabeza.
Prioridad	Secundario
Referencia	6.1, 6.2, 6.3, 6.4

Tabla 10: Descripción del caso de uso Calcular velocidad de traslación.

CU_7	Calcular velocidad de traslación.
Actor	Programador
Descripción	El caso de uso es iniciado por el actor y en él se calcula velocidad de traslación idónea para las animaciones que requieran recorrer un camino u otro tipo de traslación para que no se perciban los pies resbalando.
Prioridad	Opcional
Referencia	7.1, 7.2, 7.3

Tabla 11: Descripción del caso de uso Actualizar transición.

CU_8	Actualizar transición. <Extensión>
Actor	
Descripción	Se inicia cuando se actualiza la escena, en él se hacen los cálculos necesarios para representar las transformaciones intermedias entre las animaciones que se hace el cambio y se transforman los parámetros configurados en al CU_6.
Prioridad	Crítico
Referencia	8.1, 8.2, 8.3

Tabla 12: Descripción del caso de uso Actualizar mezcla.

CU_9	Actualizar mezcla. < Extensión >
Actor	
Descripción	Se inicia cuando se actualiza la escena, y en él se hacen los cálculos necesarios para obtener las animaciones del fotograma que se está representando como producto de la combinación de la información de las transformaciones de cada hueso en las animaciones predefinidas incluidas en la mezcla.
Prioridad	Crítico
Referencia	9.1, 9.2

Tabla 13: Descripción del caso de uso Actualizar trayectoria.

CU_10	Actualizar trayectoria. < Extensión >
Actor	
Descripción	Es iniciado cuando se actualiza la escena, el sistema calcula la posición en la que se ubicará el personaje a partir de las actualizaciones anteriores y los parámetros definidos en el caso de uso CU_4.
Prioridad	Crítico
Referencia	10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.7, 10.8, 10.9

Tabla 14: Descripción del caso de uso Mirar objetivo.

CU_11	Mirar objetivo < Extensión >
Actor	
Descripción	Se inicia cuando se actualiza la escena, en él se hacen los cálculos necesarios para orientar la cabeza hacia un punto especificado anteriormente por el caso de uso CU_5 si el punto esta dentro del rango de visión.

Prioridad	Secundario
Referencia	11.1, 11.2, 11.3, 11.4, 11.5, 11.6, 11.7

3.5.4 Expansión de casos de uso.

Tabla 15: Descripción expandida del caso de uso Gestionar Rol.

Caso de uso		
CU_1	Gestionar Rol	
Propósito	Gestionar roles de forma que sea posible crearlos, adicionarles animaciones y adicionarlos a la colección.	
Actores: Programador		
Resumen: El caso de uso es iniciado por el actor, el cual puede invocar crear un rol especificando la animación que tendrá el mismo como inactiva; o puede adicionarle una animación a un rol previamente creado.		
Referencias	1.1, 1.2, 1.3, 1.4, 1.5	
Sesión Crear Rol.		
Acción del actor		Respuesta del sistema
1. Invoca crear rol especificando: ✓ Nombre del Rol ✓ Animación que reproducirá cuando esté inactivo el personaje.		1.1– Buscar animación especificada. 1.2– Si existe la animación crear la estructura rol. 1.3– Comprobar que no exista un rol con el mismo nombre. 1.4– Si no existe un Rol con el mismo nombre crear el nuevo Rol con el nombre y la animación especificada. 1.5– Adicionar Rol a la colección de roles y termina el caso de uso.
Flujo alternativo: Animación inexistente.		
Acción del actor		Respuesta del sistema

	1.2– Si la animación no existe se termina el caso de uso.
Flujo alternativo: Rol repetido.	
Acción del actor	Respuesta del sistema
	1.4– Si existe un Rol con el mismo nombre termina el caso de uso.
Sesión adicionar animación al rol.	
Acción del actor	Respuesta del sistema
1- Invoca adicionar animación especificando: ✓ Rol que al que se le desea agrega la animación. ✓ Animación que se desea agregar.	1.1 – Buscar la animación especificada. 1.2– Si existe animación buscar el Rol especificado. 1.3– Si existe el Rol comprobar que la animación sea compatible con él. 1.4– Si es compatible la animación adicionar la misma y termina el caso de uso.
Flujo alternativo: Animación inexistente.	
Acción del actor	Respuesta del sistema
	1.2– Si la animación no existe se termina el caso de uso.
Flujo alternativo: Rol inexistente.	
Acción del actor	Respuesta del sistema
	1.3– Si el Rol no existe termina el caso de uso.
Flujo alternativo: Animación incompatible.	
Acción del actor	Respuesta del sistema
	1.3– Si la animación no es compatible termina el caso de uso.
Sesión Buscar animación en el rol.	

Acción del actor	Respuesta del sistema
1- Invoca buscar animación especificando: ✓ Nombre de la animación.	1.1– Se busca la animación especificada en la colección. 1.2– Si existe la animación, retornar el índice que ocupa en la colección y se termina el caso de uso.
Flujo alternativo: animación inexistente.	
Acción del actor	Respuesta del sistema
	1.2– Si la animación no pertenece al rol se informa y se termina el caso de uso.
Precondiciones:	Personaje cargado, colecciones de animaciones, roles y caminos creadas.

Tabla 16: Descripción expandida del caso de uso Configurar Personaje.

Caso de uso		
CU_2	Configurar Personaje	
Propósito	Configurar el personaje de forma que se le asigne o se le cree un Rol con las animaciones que puede reproducir y la animación inactiva.	
Actores: Programador		
Resumen: El caso de uso es iniciado por el actor, el cual invoca configurar un personaje especificando la animación que tendrá el personaje como inactiva o un Rol previamente creado. En caso de invocar configurar personaje especificando animación inactiva se crea un Rol con el nombre de la misma. Si se invoca configurar personaje especificando el Rol que este poseerá.		
Referencias	2.1, 2.2, 2.3	
Acción del actor		Respuesta del sistema
1- Invoca Configurar Personaje especificando: ✓ Animación que tendrá el personaje como inactiva. (ver sesión Configurar Personaje con Animación) ✓ Rol que tendrá el personaje. (ver sesión Configurar Personaje con Rol)		
Sesión: Configurar Personaje con Animación		
Acción del actor		Respuesta del sistema

	1.1– Se crea un Rol con la animación especificada 1.2– Se adiciona Rol a la colección de roles. 1.3– Se adiciona la animación inactiva del Rol a la lista de animaciones que reproduce el personaje. 1.4– Se crea un estado de animación para la animación inactiva del rol para que el personaje la reproduzca.
Sesión: Configurar Personaje con Rol	
Acción del actor	Respuesta del sistema
	1.1– Se adiciona la animación inactiva del Rol a la lista de animaciones que reproduce el personaje. 1.2– Se crea un estado de animación para la animación inactiva del rol para que el personaje la reproduzca.
Precondiciones:	Personaje cargado, colecciones de animaciones y roles.

Tabla 17: Descripción expandida del caso de uso Configurar transición de animación.

Caso de uso		
CU_3	Configurar transición de animación.	
Propósito	Especificar animación a la que se hará el cambio y características que tendrá la transición.	
Actores: Programador		
Resumen: El caso de uso es inicializado por el autor y en él se especifica como será el cambio, validando los parámetros que definen la transición como son el fotograma en el cual se iniciará el cambio y a cual animación se pasará.		
Referencias	3.1, 3.2, 3.3, 3.4	
Acción del actor		Respuesta del sistema
1- Invoca configurar transición especificando: ✓ Animación a la que se pasará. ✓ Fotograma de la animación en el que se iniciará el cambio. ✓ Velocidad en que se hará la transición.		1.1 – Se busca si la animación pertenece al rol del personaje. 1.2 – Si pertenece al rol, inicializar el factor de mezcla poniéndole todo el peso a la animación que se está reproduciendo. 1.3 – Si el fotograma a iniciar el cambio de la primera animación esta en rango, actualizar su valor con el valor especificado. 1.4 – Actualizar velocidad de la transición. 1.5 – Adicionar animación a la lista de animaciones que reproduce el personaje. 1.6 – Crear un estado para la nueva animación que reproducirá el personaje. 1.7 – Adicionar el nuevo estado a los estados de

	animación del personaje. 1.8 – Habilitar la transición y termina el caso de uso.
Flujo alternativo: Animación incorrecta.	
Acción del actor	Respuesta del sistema
	1.2 – Si no pertenece la animación al rol del personaje se termina el caso de uso.
Flujo alternativo: Fotograma fuera de rango.	
Acción del actor	Respuesta del sistema
	1.3 – Si no el fotograma definido para el inicio de la transición es mayor que la cantidad de fotogramas de la primera animación, actualizar con primer fotograma de esta.
Precondiciones:	Personaje y las animaciones cargadas. Colecciones de animaciones, roles y caminos creadas.
Poscondiciones:	

Tabla 18: Descripción expandida del caso de uso Configurar mezcla de animación.

Caso de uso		
CU_4	Configurar mezcla de animación.	
Propósito	Configurar características que tendrá la mezcla de dos animaciones sobre un personaje.	
Actores: Programador		
Resumen: El caso de uso es iniciado por el programador y en él se define como será la mezcla de las animaciones que el personaje reproducirá, se define la animación con la cual se mezclará, cuanto aportará cada animación a la resultante y en que fotograma comienza la mezcla.		
Referencias	4.1, 4.2, 4.3, 4.4, 4.5	
Acción del actor		Respuesta del sistema
1- Invoca configurar mezcla especificando: ✓ Animación con la cual se hará la mezcla. ✓ Hueso a partir del cual se aplicará la nueva animación. ✓ Fotograma a partir del cual se iniciará la mezcla.		1.1 – Comprobar si la animación pertenece al rol del personaje. 1.2 – Si pertenece al Rol, adiciona la animación a la lista de animaciones que reproduce el personaje. 1.3 – Crea un estado para esta animación. 1.4 – Actualiza el fotograma a partir del cual iniciará la mezcla. 1.5 – Adiciona el estado a la lista de estados de las animaciones que se están reproduciendo. 1.6 – Marca huesos que involucrados en la nueva animación. 1.7 – Habilita la mezcla y termina el caso de uso.

Flujo alternativo: Animación incorrecta.	
Acción del actor	Respuesta del sistema
	1.2 – Si no pertenece la animación al rol del personaje se termina el caso de uso.
Sesión Deshabilitar mezcla	
Acción del actor	Respuesta del sistema
1- Invoca deshabilitar mezcla de animación especificando: ✓ Animación que reproducirá el personaje después de terminar la mezcla.	1.1 – Si está habilitada la mezcla, deshabilita la mezcla. 1.2 – Quita de la lista de animaciones la animación que no reproducirá el personaje. 1.4 – Quita de la lista de estados de animaciones el estado correspondiente a la animación que no reproducirá el personaje y termina el caso de uso.
Flujo alternativo: mezcla no habilitada.	
Acción del actor	Respuesta del sistema
	1.1 – Si no está habilitada la mezcla se termina el caso de uso.
Precondiciones:	Personaje y las animaciones cargadas. Colecciones de animaciones, roles y caminos creadas.

Tabla 19: Descripción expandida del caso de uso Configurar recorrido.

<table>
<tr><td colspan="3">Caso de uso</td></tr>
<tr><td>CU_5</td><td colspan="2">Configurar recorrido.</td></tr>
<tr><td>Propósito</td><td colspan="2">Actualizar los valores de los parámetros que definen el recorrido.</td></tr>
<tr><td colspan="3">Actores: Programador</td></tr>
<tr><td colspan="3">Resumen: El caso de uso es iniciado por el programador y permite especificar características de la trayectoria que recorrerá el personaje. Permite definir el camino, la posición donde comenzará el recorrido en el camino, sentido que tomará personaje en el camino, orientación del personaje, velocidad de traslación, suavizado de los vértices de la trayectoria.</td></tr>
<tr><td>Referencias</td><td colspan="2">5.1, 5.2, 5.3, 5.4, 5.5</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1- Invoca configurar recorrido especificando:

✓ Camino a recorrer.
✓ Posición inicial en el camino.
✓ Sentido de la trayectoria.
✓ Orientación del personaje.
✓ Velocidad de traslación del personaje sobre en el camino.
✓ Nivel de suavizado sobre los vértices de la trayectoria.</td><td>1.1– Asignar camino al personaje.

1.2– Actualizar posición inicial especificada en el camino.

1.3 – Actualizar sentido de la trayectoria especificada.

1.4– Actualizar orientación especificada del personaje.

1.5– Actualizar velocidad de traslación especificada del personaje sobre el camino.

1.6– Actualizar nivel de suavizado especificado de los vértices de la trayectoria.

1.7– Habilitar recorrer camino y termina el caso de uso.</td></tr>
<tr><td colspan="3">Sesión Habilitar</td></tr>
</table>

Acción del actor	Respuesta del sistema
Invoca habilitar recorrido	1.1– Comprobar si está configurado recorrer trayectoria. 1.2– Si está configurado, habilitar recorrer camino y termina el caso de uso.
Flujo alternativo: Camino no configurado	
Acción del actor	Respuesta del sistema
	1.1 – Si no está configurado el camino se termina el caso de uso.
Sesión Deshabilitar	
Acción del actor	Respuesta del sistema
Invoca deshabilitar recorrido	1.1– Si está habilitado, deshabilitar propiedad recorrer camino y se termina el caso de uso.
Flujo alternativo: Camino no habilitado	
Acción del actor	Respuesta del sistema
	1.1– Si no está habilitado se termina el caso de uso.
Sesión Reiniciar	
Acción del actor	Respuesta del sistema

1- Invoca reiniciar recorrido	1.1– Actualizar posición en el camino con el valor inicial. 1.2– Actualizar sentido de la trayectoria con el valor inicial. 1.3– Actualizar orientación del personaje con el valor inicial.
	1.5– Actualizar velocidad de traslación del personaje sobre el camino con el valor inicial. 1.6– Actualizar nivel de suavizado de los vértices de la trayectoria con el valor inicial. 1.7– Habilitar recorrer camino y termina el caso de uso.
Precondiciones:	Personaje cargado. Colecciones de animaciones, roles y caminos creadas.

Tabla 20: Descripción expandida del caso de uso Configurar capacidad de visión.

<table>
<tr><td colspan="3">Caso de uso</td></tr>
<tr><td>CU_6</td><td colspan="2">Configurar capacidad de visión.</td></tr>
<tr><td>Propósito</td><td colspan="2">Configurar los parámetros que definen el objetivo a mirar por el personaje, la distancia en la que lo hace y los grados de libertad de la cabeza.</td></tr>
<tr><td colspan="3">Actores: Programador</td></tr>
<tr><td colspan="3">Resumen: En el caso de uso se definen cual es el punto de la escena donde el personaje mirará dentro de un área determinada especificando, cuanto puede girar la cabeza definiendo los grados de libertad de la misma tanto horizontales como verticales.</td></tr>
<tr><td>Referencias</td><td colspan="2">6.1, 6.2, 6.3, 6.4</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1- Invoca configurar mirar a un punto, especificando:
✓ Punto a mirar en la escena.
✓ Grados de libertad verticales de la cabeza.
✓ Grados de libertad horizontales de la cabeza.
✓ Distancia a la que se comienza a mirar.</td><td>1.1– Actualizar punto a mirar en la escena.

1.2– Actualizar grados de libertad verticales de la cabeza.

1.3– Actualizar grados de libertad horizontales de la cabeza.

1.4– Actualizar distancia en la que se comienza a mirar.

1.5– Habilitar al personaje propiedad de mirar objetivo y termina el caso de uso.</td></tr>
<tr><td colspan="3">Sesión Habilitar capacidad de visión</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1- Invoca habilitar mirar objetivo</td><td>1.1– Si está configurado el personaje para mirar objetivo habilita esta propiedad y termina el caso de uso.</td></tr>
<tr><td colspan="3">Flujo alternativo: Personaje no configurado.</td></tr>
</table>

Acción del actor	Respuesta del sistema
	1.1– Si no está configurado el personaje para mirar hacia un punto se termina el caso de uso.
Sesión Deshabilitar	
Acción del actor	Respuesta del sistema
1- Invoca deshabilitar mirar objetivo	1.1– Si está configurado el personaje para mirar objetivo inhabilita esta propiedad y termina el caso de uso.
Precondiciones:	Personaje cargado.

Tabla 21: Descripción expandida del caso de uso Calcular velocidad de traslación.

Caso de uso	
CU_7	Calcular velocidad de traslación.
Propósito	Calcular velocidad de traslación de las animaciones de personajes que lo requieran.
Actores: Programador	
Resumen: El caso de uso es iniciado por el actor y en él se calcula velocidad de traslación idónea para las animaciones que requieran recorrer un camino u otro tipo de traslación hallando la amplitud de los pasos para que no se perciban los pies resbalando. La amplitud del paso se halla calculando la posición que toman los pies en toda la animación y buscando la máxima separación que alcanzan.	
Referencias	7.1, 7.2, 7.3

Acción del actor	Respuesta del sistema
1- Invoca calcular velocidad de traslación una animación especificando: ✓ La animación a la que se le va a calcular la velocidad.	1.1– Buscar si la animación pertenece al rol del personaje. 1.2– Si pertenece al rol, calcular la posición de los huesos de los pies en cada fotograma. 1.3– Hallar la amplitud del paso promedio. 1.4– Calcular cuanto desplazar en cada fotograma. 1.5– Retorna el valor de la velocidad y termina el caso de uso.
Flujo alternativo: Animación incorrecta.	
Acción del actor	Respuesta del sistema
	1.2– Si no pertenece la animación al rol del personaje se termina el caso de uso.

Precondiciones:	Personaje y la animación cargada. Colecciones de animaciones, roles y caminos creadas.

Tabla 22: Descripción expandida del caso de uso Actualizar transición.

Caso de uso		
CU_8	Actualizar transición. < Extensión >	
Propósito	Actualizar transición del personaje dentro del ciclo de la escena.	
Actores:		
Resumen: Se inicia cuando se actualiza la animación, en él se hacen los cálculos necesarios para representar las transformaciones intermedias entre las animaciones que se hace el cambio.		
Referencias	8.1, 8.2, 8.3	
Acción del actor		Respuesta del sistema
		1.1– Si está habilitada la transición se comprueba que no se ha alcanzado el valor de mezcla máximo. 1.2– Si no se ha alcanzado el valor máximo de mezcla este se incrementa según la velocidad especificada. 1.3– Interpolar las animaciones con el factor de mezcla actualizado. 1.4– Pasar la animación resultante a la actualización del esqueleto.
Flujo alternativo: Transición no habilitada.		
Acción del actor		Respuesta del sistema

	1.1– Si no está habilitada la transición termina el caso de uso.
Flujo alternativo: Factor de mezcla máximo.	
Acción del actor	Respuesta del sistema
	1.2– Si se ha alcanzado el valor máximo, detener la primera animación 1.3– Deshabilitar transición. 1.3– Remover la animación que se dejó de reproducir de las animaciones que reproduce el personaje. 1.4– Borrar el estado de esta animación asociada a la animación que se dejó de reproducir y termina el caso de uso.
Precondiciones:	Personaje y las animaciones cargadas. Colecciones de animaciones, roles y caminos creadas.

Tabla 23: Descripción expandida del caso de uso Actualizar mezcla.

Caso de uso		
CU_9	Actualizar mezcla. < Extensión >	
Propósito	Obtener una animación como producto de la combinación de los fotogramas de dos animaciones previamente definidas.	
Actores:		
Resumen: Se inicia cuando se actualiza la escena, en él se actualizan cada hueso del esqueleto con la animación asociada a ellos configurada previamente en el CU_4.		
Referencias	9.1, 9.2	
Acción del actor		Respuesta del sistema
		1.1– Comprobar si la mezcla está activada. 1.2– Si la mezcla está activada, comprobar que las animaciones asociadas a los huesos estén reproduciéndose. 1.3– Para cada hueso se le aplica la transformación del *frame* que se está reproduciendo de la animación asociada a él y termina el caso de uso.
Flujo alternativo: Mezcla no activada		
Acción del actor		Respuesta del sistema
		1.2– Si la mezcla no está activada se termina el caso de uso.
Precondiciones:	Personaje y las animaciones cargadas. Colecciones de animaciones, roles y caminos creadas.	

Tabla 24: Descripción expandida del caso de uso Actualizar recorrido.

Caso de uso		
CU_10	Actualizar recorrido. < Extensión >	
Propósito	Ubicar el personaje en la posición y con orientación correcta según el camino y la configuración predefinida para recorrer el mismo.	
Actores:		
Resumen: Es iniciado cuando se actualiza la escena, el sistema calcula la posición en la que se ubicará el personaje a partir de las actualizaciones anteriores y los parámetros definidos en el caso de uso CU_4.		
Referencias	10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.7, 10.8, 10.9	
Acción del actor		Respuesta del sistema
		1.1– Comprobar que esté habilitada la propiedad en el personaje de recorrer camino. 1.2– Si está habilitado recorrer camino, localizar el vector por el cual se traslada el personaje. 1.3– Calcular la posición del personaje sobre el vector. 1.4– Calcular la orientación del personaje sobre el vector. 1.5– Calcular la posición que debe tener el personaje para que la trayectoria sea suavizada. 1.5- Calcular la orientación que debe tener el personaje para que la trayectoria sea suavizada. 1.6– Actualizar los valores de posición y orientación a tener en cuanta en el siguiente ciclo de escena. 1.7– Ubicar el personaje en la posición calculada con la orientación resultante y termina el caso de uso.

Flujo alternativo: Camino inhabilitado	
Acción del actor	Respuesta del sistema
	1.2– Si no está habilitado el camino se termina el caso de uso.
Precondiciones:	Personaje cargado. Colecciones de animaciones, roles y caminos creadas.

Conclusiones

El capítulo presentado anteriormente, describió lo que el sistema debe ser capaz de ofrecer a sus usuarios. Con este fin se establecieron los requisitos funcionales, se agruparon en casos de uso y se describieron para un mejor entendimiento de los procesos que tendrán lugar en el módulo a desarrollar aquellos que serán incluidos en el primer ciclo de desarrollo.

4

Capítulo 4: Diseño e Implementación del Sistema

Introducción

En este capítulo se muestran los diagramas de clases del módulo de animación propuesto, así como los diagramas de secuencia de la realización de los casos de uso correspondientes al primer ciclo de desarrollo. También se encuentran los diagramas de componentes en los que se definen cómo se harán físicas las clases de diseño en el lenguaje especificado y de forma compatible con el *SceneToolkit,* sistema del cual formará parte este módulo.

Además de lo antes mencionado, se especifican otros aspectos necesarios para la comprensión de los diagramas representados y se detallan aspectos relacionados con la herramienta *SceneToolkit*.

4.1 Diagrama de Paquetes de clases de diseño.

Las clases necesarias para el funcionamiento del módulo a desarrollar tienen responsabilidades que las distinguen a unas de otras por lo que se pueden clasificar y agrupar por paquetes. El propio módulo como formará parte de otra aplicación, debe mantener la estructura y organización de la misma.

A continuación se muestran los principales paquetes en los que se agrupan las clases y las dependencias entre ellos. Los mismos formaban parte de la herramienta anterior al diseño del nuevo módulo y las nuevas estructuras se ajustan a estos paquetes.

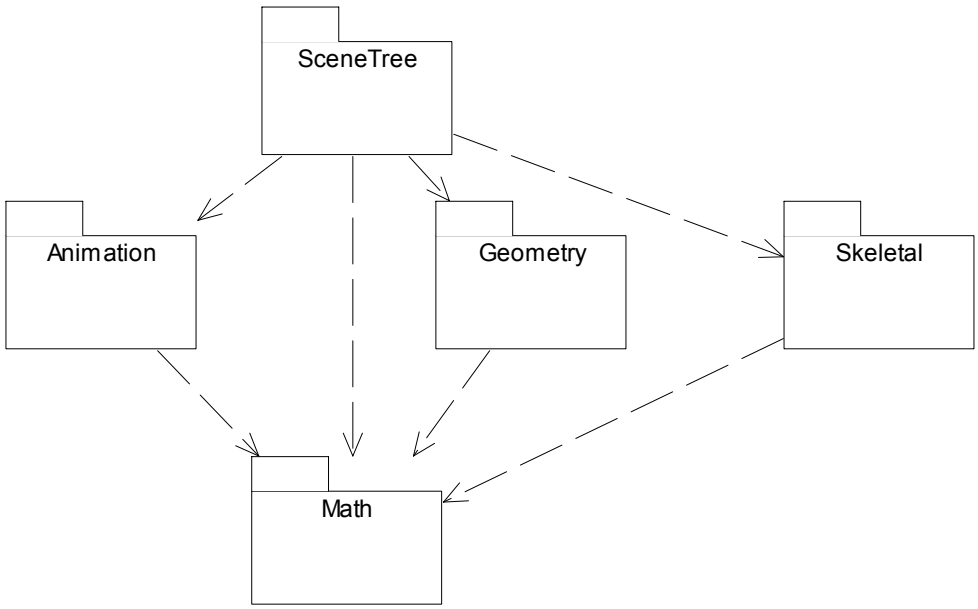


Figura 20: Diagrama de subsistemas del diseño.

Dentro de los subsistemas representados se encuentran todas las clases necesarias para las funcionalidades del módulo. Cada uno de los paquetes contiene las clases especificadas a continuación:

Tabla 25: Relación entre clases y subsistemas.

Subsistemas	**Clases de interés contenidas**
SceneTree	*CNodo* *CGroupNode* *CSkeletalNode* *CBoneNode*
Animation	*CPath* *CPathController* *CAnimCollection* *CBoneFrame* *CSkeletalAnimation* *CSkeletalAnimState* *CRole* *CSkeletalController*
Geometry	*CTriMesh*
Sekeletal	*CBone* *CVertexWeight* *CSkeleton* *CSkeletalModel*
Math	*CVector* *CMatrix3* *CQuaternion*

4.2 Estándares de codificación.

El código del módulo a implementar está en correspondencia con los estándares presentes en la STK. El idioma utilizado para la nomenclatura será el inglés, debido que las palabras son más simples, no se acentúan y es un idioma muy difundido en el mundo informático.

El conocimiento de los estándares seguidos para el desarrollo de la misma permitirá un mayor entendimiento del código, y es una exigencia de los autores de la misma que cualquier módulo que se añada debe estar codificado siguiendo estos estándares.

Nombre de los ficheros:

Se nombrarán los ficheros .h y .cpp de la siguiente manera:

STKNameOfUnits.cpp

Se usará STK para identificar el nombre de la herramienta.

Constantes:

Las constantes se nombrarán con mayúsculas, utilizándose el "_" para separar las palabras: MY_CONST_ZERO = 0;

Tipos de datos:

Los tipos se nombrarán siguiendo el siguiente patrón:

Enumerados: enum EMyEnum {ME_VALUE, ME_OTHER_VALUE};

Indicando con "E" que es de tipo enumerado. Nótese que las primeras letras de las constantes de enumerados son las iniciales del nombre del enumerado. Véase otro ejemplo:

enum ENodeType {NT_GEOMETRYNODE,…};

Estructuras: struct SMyStruct {…};

Indicando con “S” que es una estructura. Las variables miembros de la estructura se nombrarán igual que en las clases, leer más adelante.

Clases: class CClassName;

Indicando con “C” que es una clase

Interfaces: IMyInterface

Indicando con “I” que es una interfaz.

Listas e iteradores STD:

vector<> TNameList;

TNameList::iterator TNameListIter;

map<> TNameMap;

TNameMap::iterator TNameMapIter;

multimap<> TNameMultiMap;

TNameMultiMap::iterator TNameMultiMapIter;

Declaración de variables:

Los nombres de las variables comenzarán con un identificado del tipo de dato al que correspondan, como se muestra a continuación. En el caso de que sean variables miembros de una clase, se le antepondrá el

identificador “” (en minúscula), si son globales se les antepondrá la letra “g”, y en caso de ser argumentos de algún método, se les antepondrá el prefijo “arg_”.

Tipos simples:

bool bVarName;

int iName;

unsigned int uiName;

float fName;

char cName;

char* acName; // arreglo de caracteres

char* pcName; // puntero a un char

char** aacName; // bidimensional

char** apcName; // arreglo de punteros

bool m_bMemberVarName; //variable miembro

char gcGlobalVarName; //variable global, no se le antepone “”

short sName;

Instancias de tipos creados:

EMyEnumerated eName;

SMyStructure kName;

CClassName kObjectName;

CClassName* pkName; //puntero a objeto

CClassName* akName; //arreglo de objetos

CClassName* akName; // variable miembro de clase

IMyInterface* pIName; //puntero interfaces

Métodos:

En el caso de los métodos, se les antepondrá el identificador del tipo de dato de devolución, y en caso de no tenerlo (void), no se les antepondrá nada. Solamente los constructores y destructores comenzarán con "".En el caso de los argumentos se les antepone el prefijo "arg_"

Constructor y destructor:

CClassName (bool arg_bVarName, float& arg_fVarName);

~CClassName ();

Funciones:

bool bFunction1 (…);

int* piFunction2 (…);

CClassName* pkFunction3 (…);

Procedimientos:

void Procedure4 (…);

Métodos de acceso a miembros:

Los métodos de acceso a los miembros de las clases no se nombrarán “Gets” y “Sets”, sino como los demás métodos, pero con el nombre de la variable a la que se accede y sin “m_”:

```
int iMyVar;  //variable
```

Obtención del valor:

```
int iMyVar();
{
   return iMyVar;
}
```

Establecimiento del valor:

```
void MyVar(char* arg_iMyVar)
{
  iMyVar = arg_iMyVar;
}
```

Obtención y establecimiento del valor:

```
int& iMyVar();
{
   return iMyVar;
}
```

4.3 Diagrama de clases de diseño.

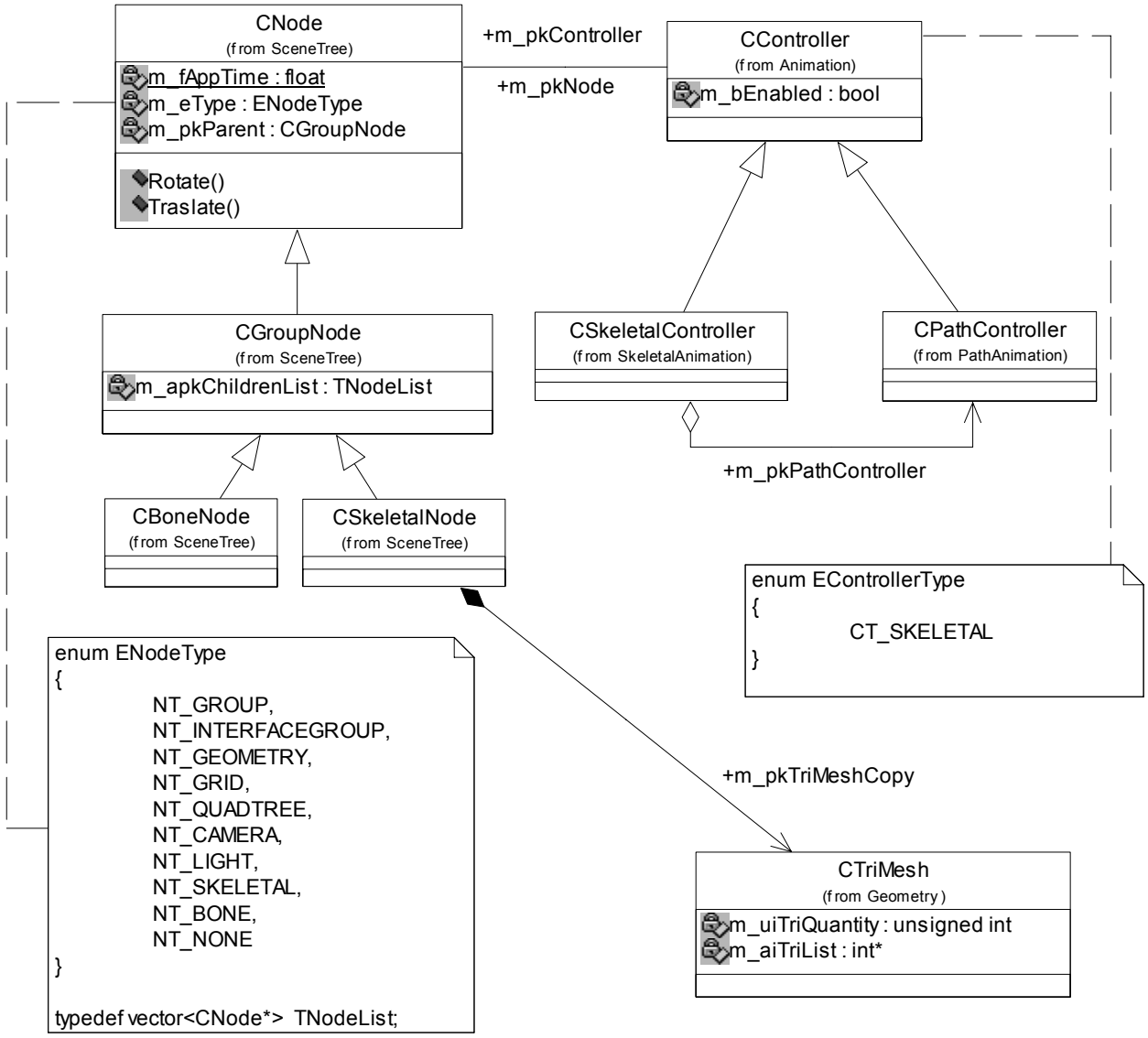


Figura 21: Diagrama de clases de diseño de nodos y controladores.

En el anterior diagrama se aprecian las relaciones entre los nodos y los controladores de acuerdo con la arquitectura que presenta la herramienta. Las clases *CSkeletalController*, *CSkeletalNode*, *CBoneNode*, *CPathController* pertenecen al nuevo módulo de animación, las otras clases existían en la herramienta anteriormente al modelado de dicho módulo. Se incluyen como notas en el diagrama enumerados con tipos definidos para estas estructuras. Los atributos y métodos de mayor interés de las nuevas clases se pueden ver en los diagramas que se verán más adelante.

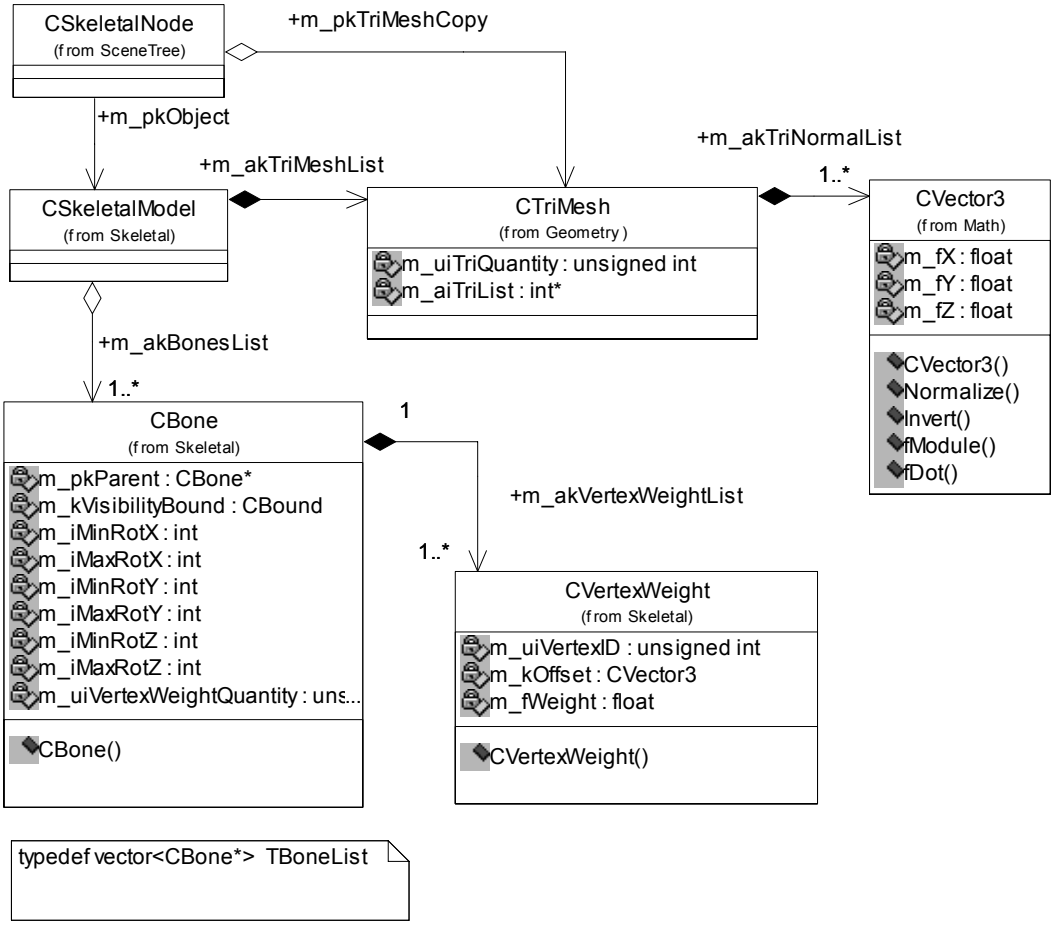


Figura 22: Diagrama de clases de diseño para conformar un personaje con esqueleto.

En el anterior diagrama se representan las clases que definen un personaje con las características necesarias para ser animado mediante la técnica de animación por esqueletos.

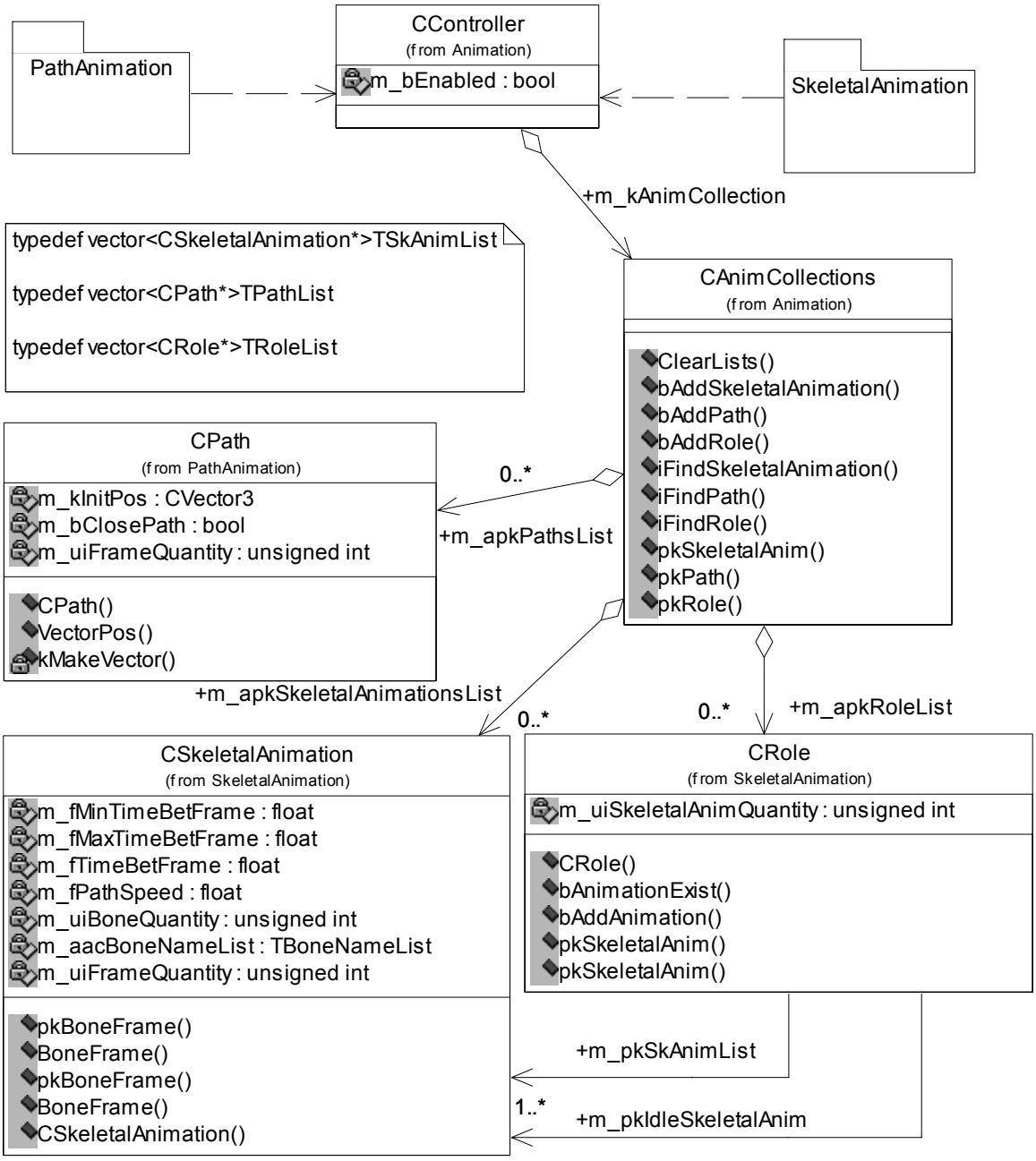


Figura 23: Diagrama de clases de diseño correspondiente a las colecciones de animaciones caminos y roles.

El diagrama de clases anterior muestra las relaciones entre las clases tanto entidades como controladoras que permiten almacenar animaciones, roles y caminos en forma de colecciones. De esta manera se tiene en memoria los datos necesarios para animar los personajes cargados.

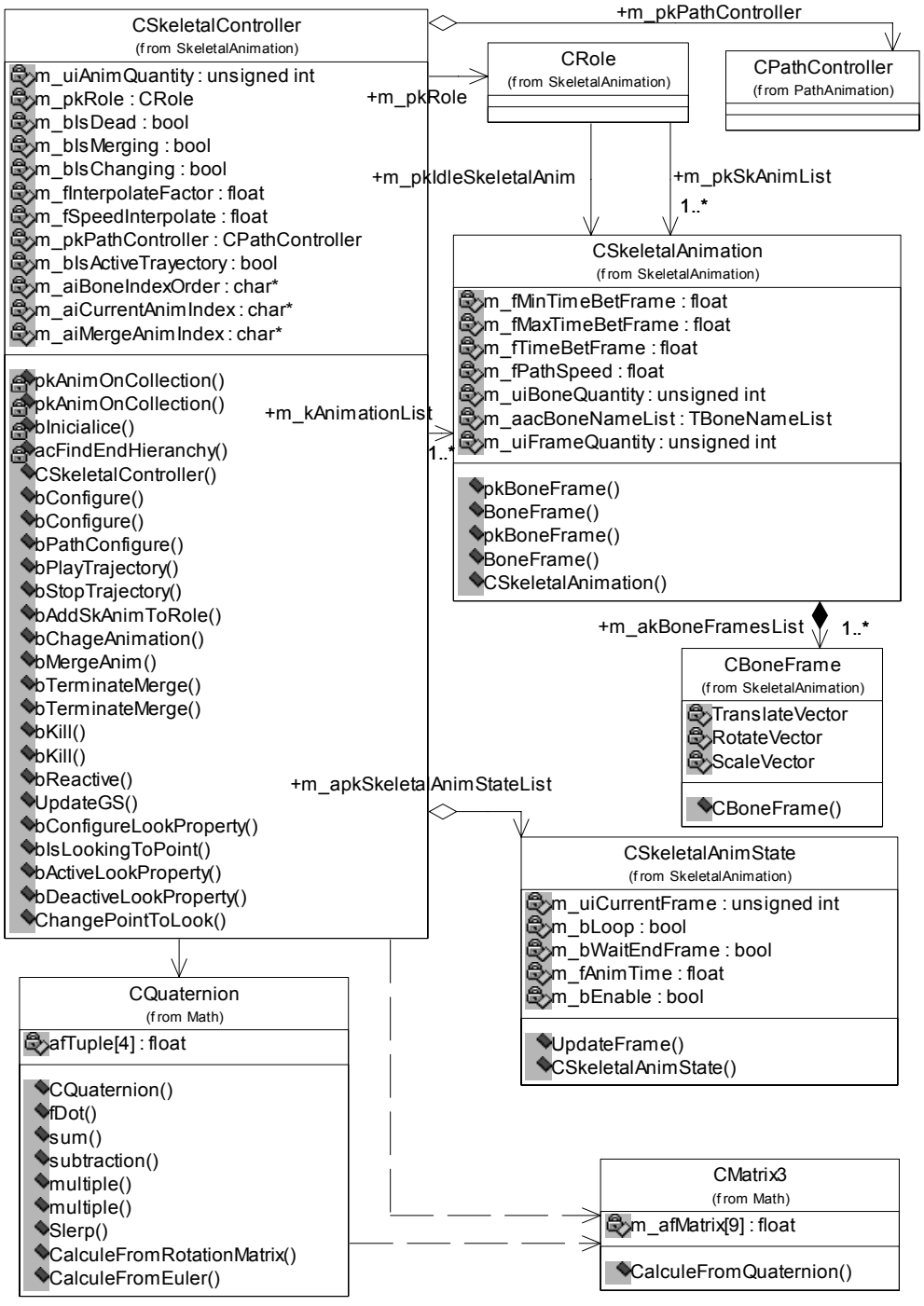


Figura 24: Diagrama de clases de diseño para el manejo de animaciones y las colecciones de estas.

Las clases y relaciones presentes en diagrama anterior permiten tener el control de las animaciones que reproduce el personaje, los caminos que recorre y el su rol.

El siguiente diagrama muestra las clases necesarias y las relaciones entre ellas para el control del desplazamiento a través de un camino definido para el personaje mediante el controlador de esqueleto. Estas clases también pueden controlar el movimiento de otros objetos en las escenas mediante el uso directo del controlador de camino.

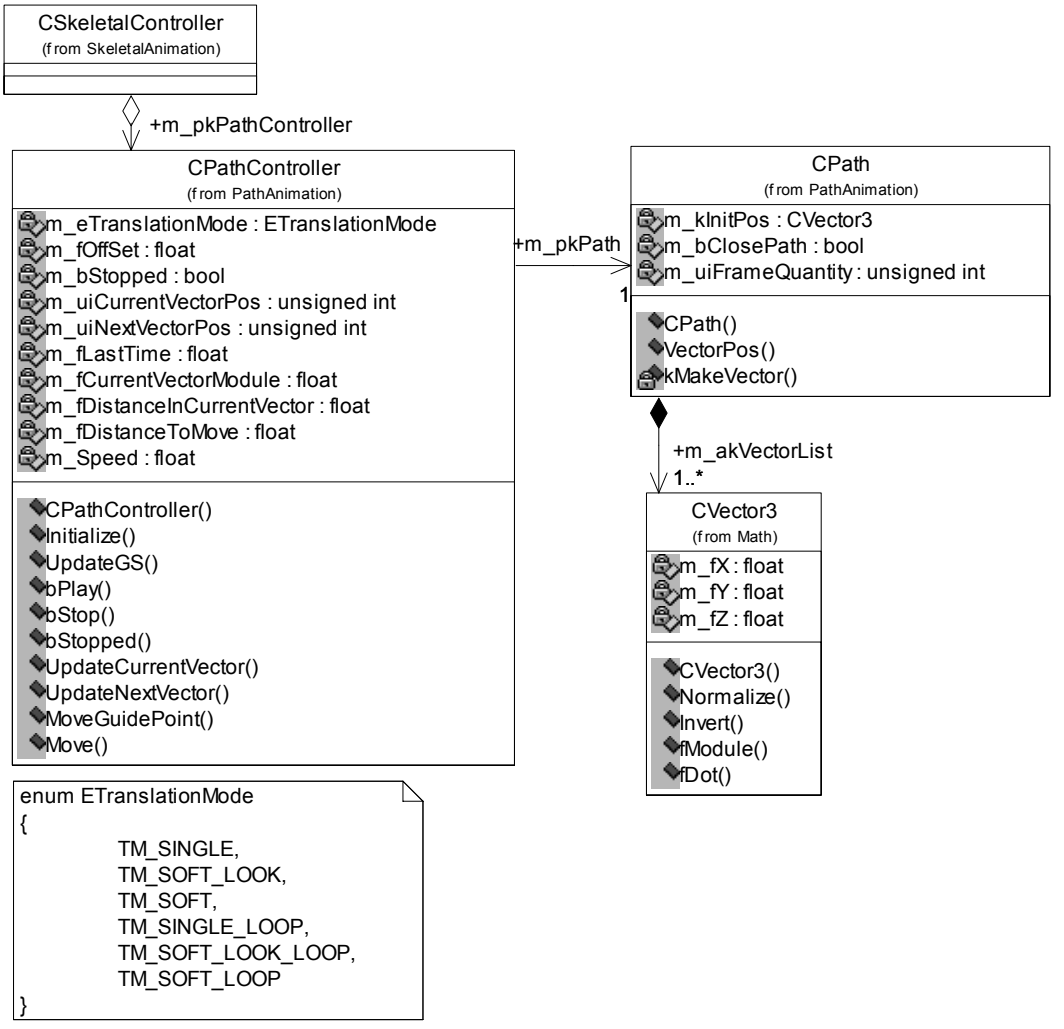


Figura 25: Diagrama de clases de diseño para la manipulación de la trayectoria de un personaje u otro modelo.

4.3.1 Descripción de clases de diseño.

Después de haber visto los diagramas de clases es necesario especificar en alguna de ellas sus características. En las tablas que aparecen a continuación se describen los atributos y métodos más importantes de las clases, fundamentales incluidas en el módulo estas son: *CSkeletalController, CPathController, CAnimCollections, CSkeletalAnimation, CPath, CRole* y *CSkeletalAnimState.*

Tabla 26: Descripción de la clase *CSkeletalController.*

Nombre de la Clase: CSkeletalController	
Tipo de clase: Controladora	
Atributo	**Tipo**
m_uiAnimQuantity	unsigned int
m_pkRole	CRole
m_bIsDead	bool
m_bIsMerging	bool
m_bIsChanging	bool
m_fInterpolateFactor	float
m_fSpeedInterpolate	float
m_pkPathController	CPathController
m_bIsActiveTrayectory	bool
m_aiBoneIndexOrder	char*
m_aiCurrentAnimIndex	char*
m_aiCurrentAnimIndex	char*
m_aiMergeAnimIndex	char*
Responsabilidades de la clase:	
Nombre:	pkAnimOnCollection
Descripción:	Busca una animación en la colección especificando el nombre o por el índice que ocupa en la misma.

Nombre:	bInicialice
Descripción:	Inicializa el esqueleto del personaje definiendo el orden de los índices de los huesos para optimizar el recorrido de estos.
Nombre:	acFindEndHieranchy
Descripción:	Busca el fin de una jerarquía de huesos especificando el hueso padre.
Nombre:	bConfigure
Descripción:	Configura el personaje especificando la animación o el nombre de esta y las características que tendrá esta animación que será la inactiva.
Nombre:	bPathConfigure
Descripción:	Configura el camino especificando el mismo y como será recorrido
Nombre:	bPlayTrajectory
Descripción:	Hace que el camino se recorra si está configurado
Nombre:	bStopTrajectory
Descripción:	Hace que se detenga el recorrido por el camino si este está configurado
Nombre:	bAddSkAnimToRole
Descripción:	Adiciona una animación al Rol del personaje especificando la animación el nombre de la misma
Nombre:	bChageAnimation
Descripción:	Configura la transición de animación especificando la animación a la que se cambiará, las características de esta para reproducirse y la velocidad de cambio.
Nombre:	bMergeAnim
Descripción:	Configura la mezcla de animación especificado con que animación se mezclará a partir de que hueso y la velocidad con la que se pasará a la mezcla.
Nombre:	bTerminateMerge
Descripción:	Termina la mezcla dando la posibilidad de especificar que animación reproducir después de terminada la misma.
Nombre:	bKill
Descripción:	Le quita la vida al personaje dando la posibilidad de especificar con que animación matar al mismo.

Nombre:	bReactive
Descripción:	Le da vida al personaje dando la posibilidad de especificar con que animación se quiere que se realice la acción, de no especificarse se comienza a reproducir la animación inactiva.
Nombre:	UpdateGS
Descripción:	Se encarga de actualizar las transformaciones que se aplicaran a los huesos y si se esta en proceso de mezcla o transición se calculan las transformación finales que se le aplicarán a cada hueso en el *frame* correspondiente.
Nombre:	bConfigureLookProperty
Descripción:	Configura la capacidad de visión del personaje, donde se especifica el punto al cual se observará, los grados de libertad de la cabeza y la distancia en la cual se mirará.
Nombre:	bIsLookingToPoint
Descripción:	Necesaria para comprobar si el personaje está mirando el objetivo.
Nombre:	bDeactiveLookProperty
Descripción:	Desactiva la propiedad de mirar para el personaje.
Nombre:	ChangePointToLook
Descripción:	Cambia el punto al que el personaje mira especifican el punto.

Tabla 27: Descripción de la clase *CPathController*.

Nombre de la Clase: CPathController	
Tipo de clase: Controladora	
Atributo	**Tipo**
m_eTranslationMode	ETranslationMode
m_fOffSet	float
m_bStopped	bool
m_uiCurrentVectorPos	unsigned int

m_uiNextVectorPos	unsigned int
m_fLastTime	float
m_fCurrentVectorModule	float
m_fDistanceInCurrentVector	float
m_fDistanceInCurrentVector	float
m_fDistanceToMove	float
m_Speed	float

Responsabilidades de la clase:	
Nombre:	Initialize
Descripción:	Ubica el nodo en la posición inicial en la dirección del vector que le corresponde
Nombre:	UpdateGS
Descripción:	Actualiza la posición y orientación del nodo
Nombre:	bPlay
Descripción:	Si el camino está configurado activa el controlador para que este lo recorra.
Nombre:	bStop
Descripción:	Si el nodo está en movimiento hace que el nodo se detenga en la posición que esté.
Nombre:	bStopeed
Descripción:	Retorna verdadero si el nodo está detenido.
Nombre:	UpdateCurrentVector
Descripción:	Pasa al próximo vector del camino.
Nombre:	UpdateNextVector
Descripción:	Actualiza el vector siguiente al que está recorriendo.
Nombre:	MoveGuidePoint
Descripción:	Calcula la posición de la guía que recorre el camino.

Tabla 28: Descripción de la clase *CAnimCollections*.

Nombre: CAnimCollections	
Tipo de clase: Entidad	
Atributo	**Tipo**
m_apkSkeletalAnimationsList	TSkAnimList
m_apkPathsList	TPathList
m_apkRolesList	TRoleList
Responsabilidades de la clase:	
Nombre:	bAddSkeletalAnimation
Descripción:	Adiciona una animación a la colección.
Nombre:	iFindSkeletalAnimation
Descripción:	Busca una animación especificada por su nombre en la colección y devuelve su índice.
Nombre:	pkSkeletalAnim
Descripción:	Busca una animación especificada por su nombre en la colección y devuelve una referencia a ella.
Nombre:	bAddPath
Descripción:	Adiciona un camino a la colección.
Nombre:	iFindPath
Descripción:	Busca un camino especificado por su nombre en la colección y devuelve su índice.
Nombre:	pkPath
Descripción:	Busca un camino especificado por su nombre en la colección y devuelve una referencia a él.
Nombre:	bAddRole
Descripción:	Adiciona un Rol a la colección.
Nombre:	iFindRole
Descripción:	Busca un Rol especificado por su nombre en la colección y devuelve su índice.
Nombre:	pkRole

Descripción:	Busca un Rol especificado por su nombre en la colección y devuelve una referencia a él.

Tabla 29: Descripción de la clase *CSkeletalAnimation*.

Nombre de la Clase: CSkeletalAnimation	
Tipo de clase: Entidad	
Atributo	**Tipo**
m_fMinTimeBetFrame	float
m_fMaxTimeBetFrame	float
m_fTimeBetFrame	float
m_fPathSpeed	float
m_uiBoneQuantity	unsigned int
m_aacBoneNameList	TBoneNameList
m_uiFrameQuantity	unsigned int
Responsabilidades de la clase:	
Nombre:	pkBoneFrame
Descripción:	Especificando el índice del hueso y del *frame,* retorna las transformaciones que el hueso indicado debe adoptar en el *frame* especificado.

Tabla 30: Descripción de la clase *CPath.*

Nombre de la Clase: CPath	
Tipo de clase: Entidad	
Atributo	**Tipo**
m_kInitPos	CVector3
m_bClosedPath	bool

m_uiFrameQuantity	unsigned int
m_akVectorList	CVector3*
Responsabilidades de la clase:	
Nombre:	AddVector
Descripción:	Adiciona un vector a la lista.
Nombre:	kMakeVector
Descripción:	Convierte dos puntos en un vector de traslación.

Tabla 31: Descripción de la clase *CRole.*

Nombre: CRole	
Tipo de clase: Entidad	
Atributo	**Tipo**
m_pkSkAnimList	TSkAnimList
m_uiSkeletalAnimQuantity	unsigned int
m_pkIdleSkeletalAnim	CSkeletalAnimation
Responsabilidades de la clase	
Nombre:	bAnimationExist
Descripción:	Retorna si la animación especificada pertenece al Rol.
Nombre:	bAddAnimation
Descripción:	Adiciona una animación especificada por el nombre al Rol.
Nombre:	pkSkeletalAnim
Descripción:	Retorna una referencia de animación especificada por el nombre si esta pertenece al Rol.

Tabla 32: Descripción de la clase *CSkeletalAnimState*.

Nombre de la Clase: CSkeletalAnimState	
Tipo de clase: Entidad	
Atributo	**Tipo**
m_pkSkAnim	CSkeletalAnimation*
m_uiCurrentFrame	unsigned int
m_bLoop	bool
m_bWaitEndFrame	bool
m_fAnimTime	float
m_bEnabled	bool
Responsabilidades de la clase	
Nombre:	UpdateFrame
Descripción:	Actualiza el *frame* que se debe reproducir según el tiempo que ha acumulado la animación y el *frame* por el que haya comenzado.

4.4 Diagramas de secuencia.

Los diagramas siguientes corresponden a las relaciones que se establecen de forma secuencial entre los objetos en las principales funcionalidades descritas en los escenarios de los casos de uso del sistema.

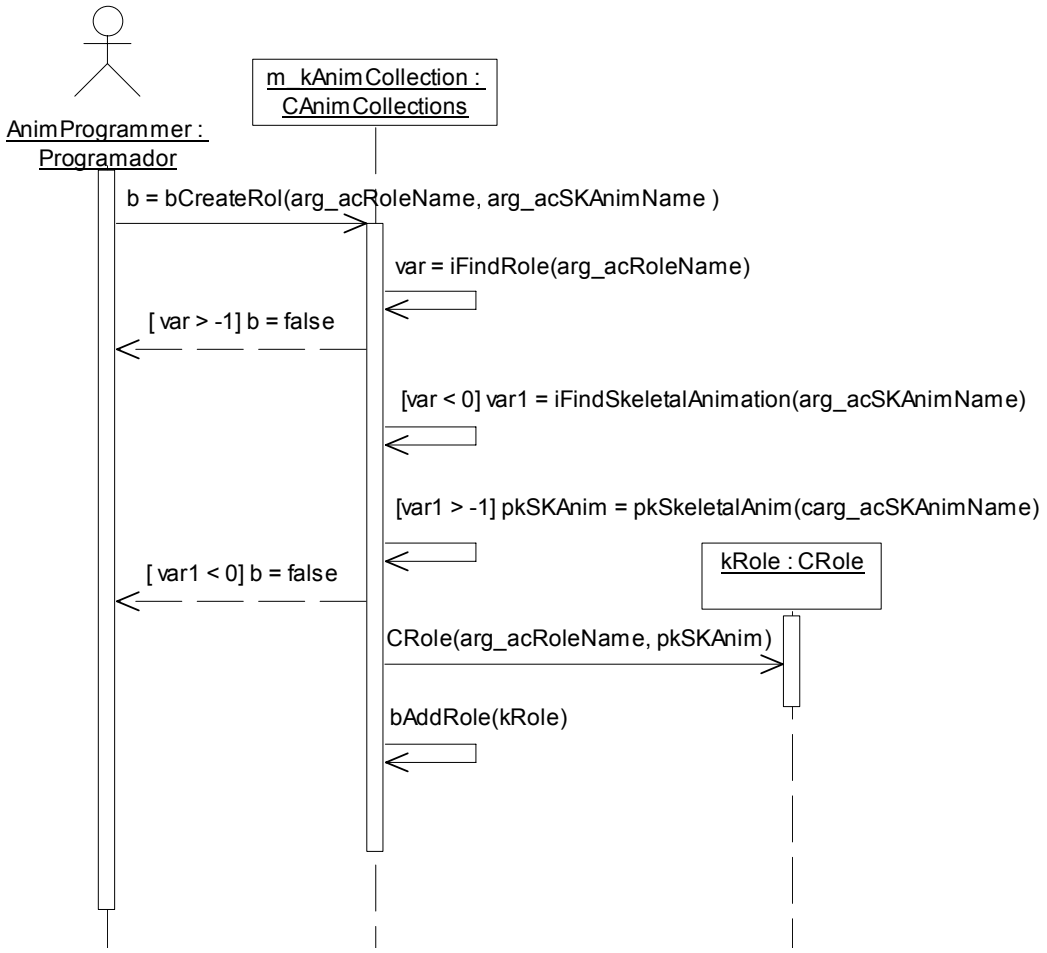


Figura 26: Diagrama de secuencia Crear Rol.

El diagrama anterior corresponde a la sesión *Crear Rol* del caso de uso *Gestionar Rol*. En él se especifica el nombre del rol y la animación inactiva que tendrá el mismo. Si el Rol existe se retorna falso, de lo contrario se busca la animación y si la animación no está en la colección se retorna falso, de encontrarse la animación se crea el Rol y se adiciona en la colección de roles.

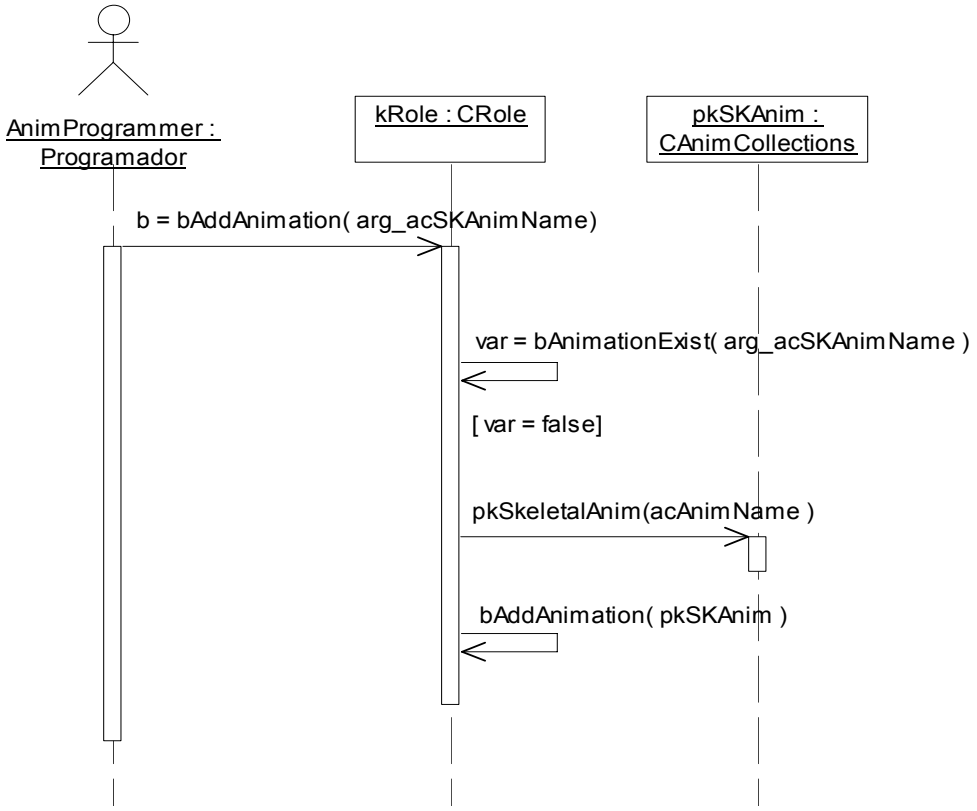


Figura 27: Diagrama de secuencia Adicionar Animación al Rol.

El diagrama anterior corresponde a la sesión *Adicionar animación al Rol* del caso de uso *Gestionar Rol*. En él se especifia la animación que se le quiere adicionar al Rol y si esta se encuentra en la colección de animaciones se adiciona al Rol y se retorna verdadero.

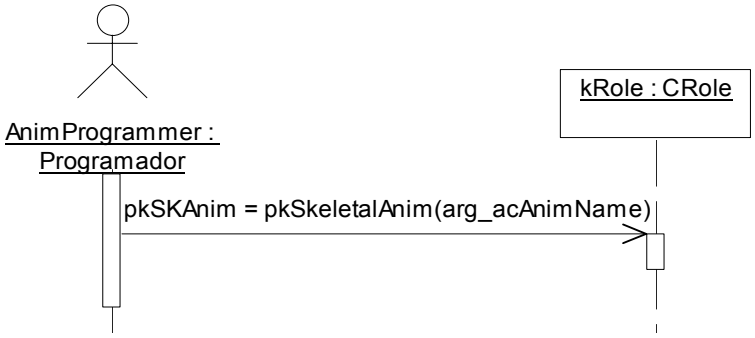


Figura 28: Diagrama de secuencia Buscar animación en el Rol.

El diagrama anterior corresponde a la sesión *Buscar animación en el rol* del caso de uso *Gestionar Rol*. En él se especifica el nombre de la animación que se está buscando, la misma se retorna si aparece asociada al Rol.

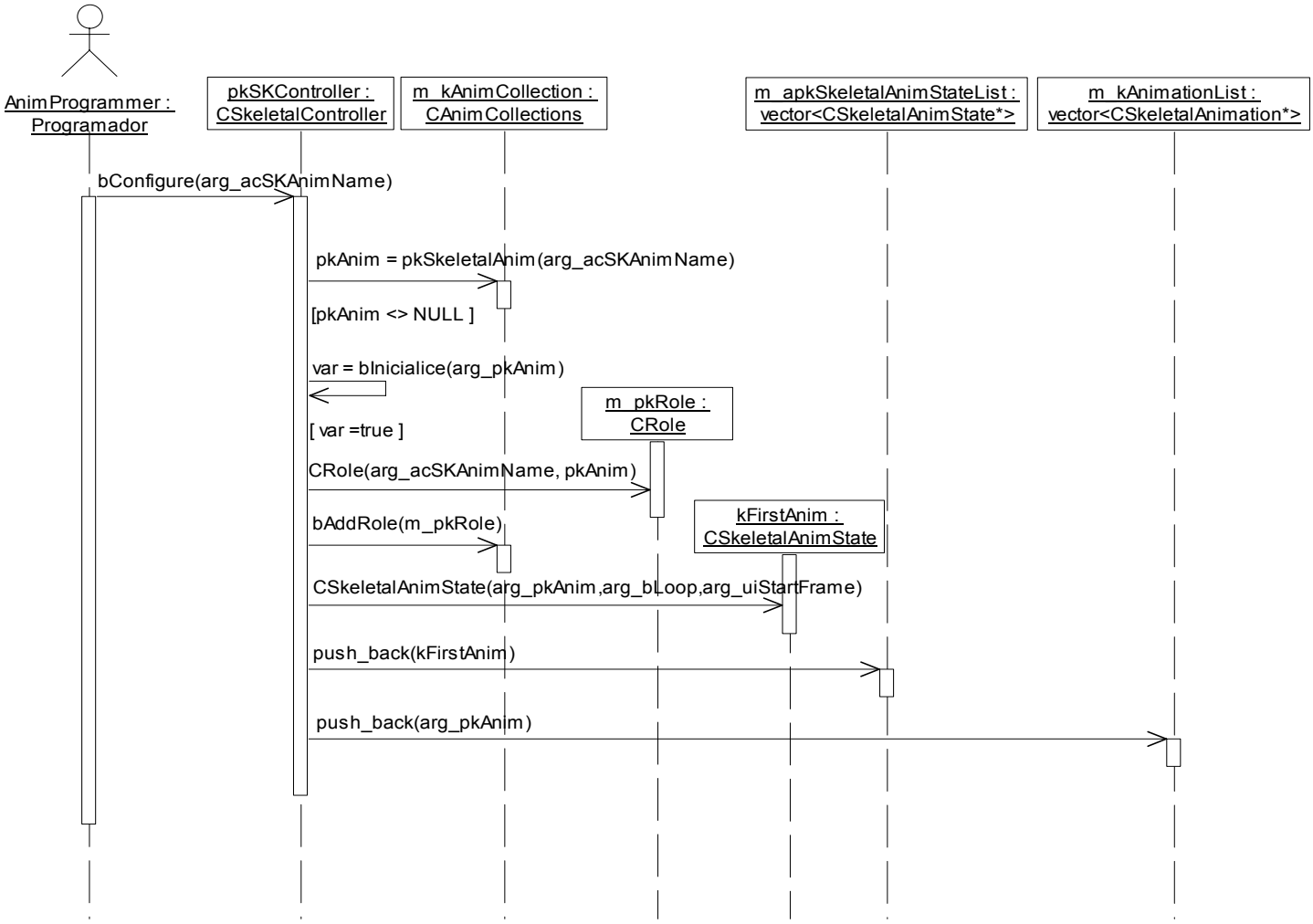


Figura 29: Diagrama de secuencia Configurar personaje.

El diagrama anterior corresponde a la sesión *Configurar Personaje con Animación* del caso de uso *Configurar Personaje*. En él se especifica el nombre de la animación con la cual se configurará el personaje. Luego se busca la animación, se crea un Rol con la misma, se adiciona el Rol a la lista de Roles, se crea un estado de animación para la animación y se adiciona a la lista de animaciones que reproduce el personaje.

Figura 30: Diagrama de secuencia Configura Mezcla.

El diagrama anterior corresponde al caso de uso *Configurar Mezcla*. En él se especifica el nombre de la animación que se mezclará, el hueso a partir del cual comenzará a reproducirse la nueva animación, el fotograma por el cual empezará la nueva animación a ser reproducida y la velocidad con la que se pasará a mostrar la mezcla de las dos animaciones. Luego se busca la nueva animación, se crea un estado para esta y se comienza a reproducir la misma.

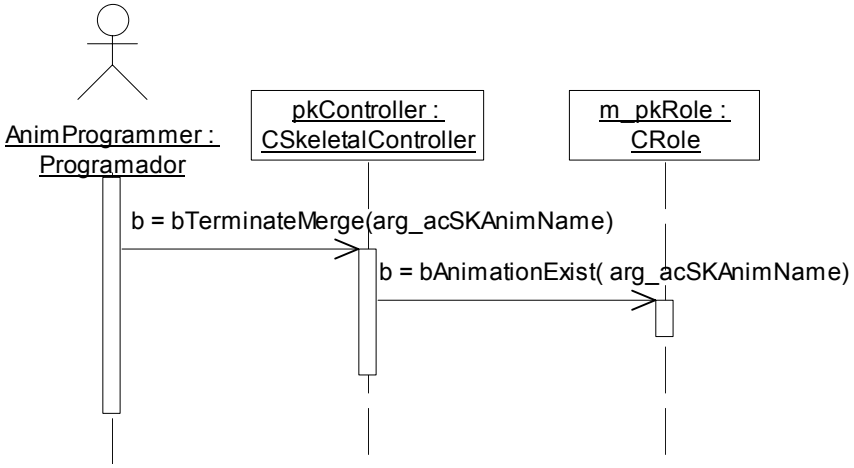


Figura 31: Diagrama de secuencia Terminar Mezcla.

El diagrama anterior corresponde a la sesión *Deshabilitar mezcla* del caso de uso *Configurar Mezcla*. En él se especifica la animación a la que se pasará después de terminar la mezcla, si esta animación se encuentra en el RoI se pasa a la misma, de lo contrario se pasa a la primera animación de las que reproduce el personaje.

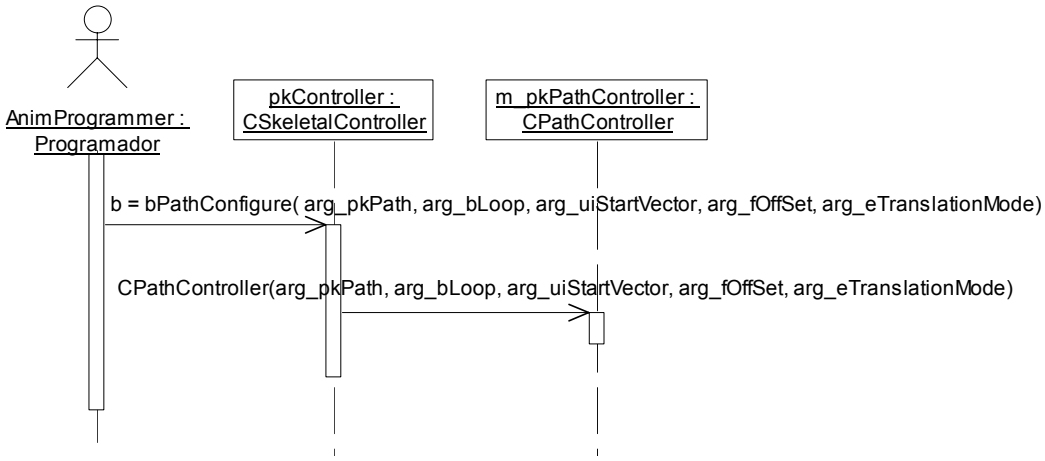


Figura 32: Diagrama de secuencia Configurar trayectoria.

El diagrama anterior corresponde al caso de uso *Configurar recorrido.* En él se especifican características de la trayectoria como son: camino a recorrer, posición inicial en el camino, sentido de la trayectoria, orientación del personaje, velocidad de traslación del personaje sobre en el camino, nivel de suavizado sobre los vértices de la trayectoria.

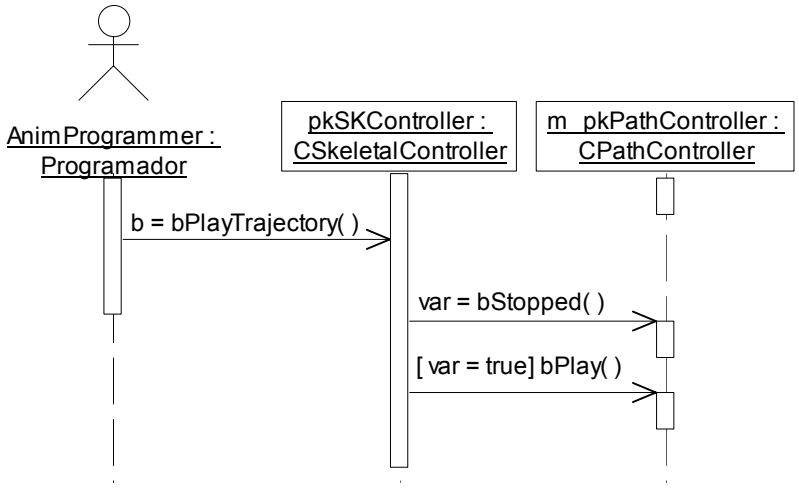


Figura 33: Diagrama de secuencia Habilitar trayectoria.

El diagrama anterior corresponde a la sesión *Habilitar recorrido* del caso de uso *Configurar recorrido.* En él se chequea que esté deshabilitado recorrer camino y de ser así se habilita mediante el controlador de camino asociado al personaje.

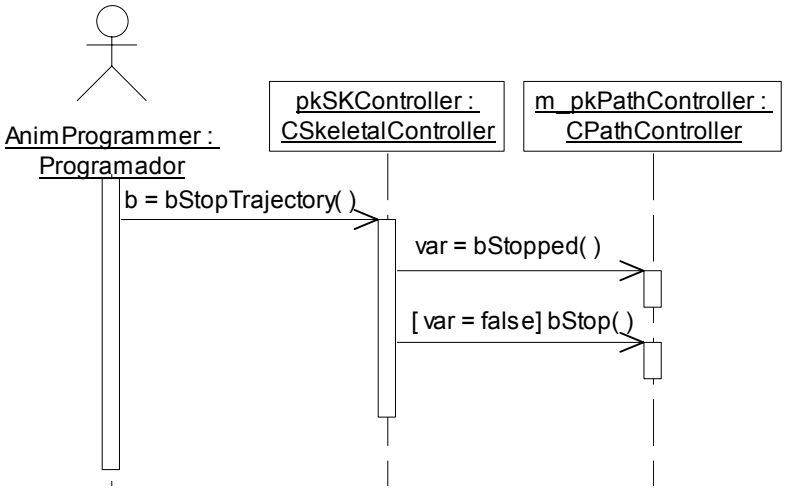


Figura 34: Diagrama de secuencia Deshabilitar trayectoria.

El diagrama anterior corresponde a la sesión *Deshabilitar recorrido* del caso de uso *Configurar recorrido.* En él se chequea que esté habilitado recorrer camino y de ser así se deshabilita mediante el controlador de camino asociado al personaje.

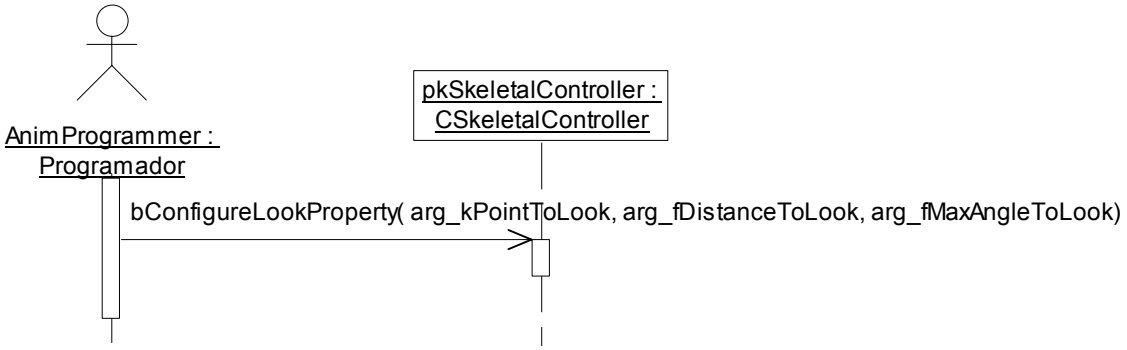


Figura 35: Diagrama de secuencia Configurar capacidad de visión.

El diagrama anterior corresponde al caso de uso *Configurar capacidad de visión.* En él se especifica el punto a donde mirará el personaje, la distancia a la que mirará y los grados de libertad de la cabeza. Luego se guardan estos datos para actualizar la orientación de la cabeza en el ciclo de actualización.

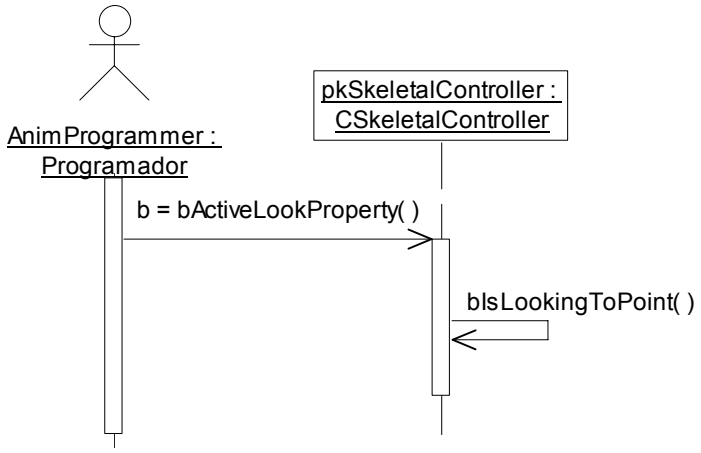


Figura 36: Diagrama de secuencia Habilitar Propiedad Visión.

El diagrama anterior corresponde a la sesión *Habilitar capacidad de visión* del caso de uso *Configurar capacidad de visión.* En él si no se está mirando un objetivo se activa esta propiedad.

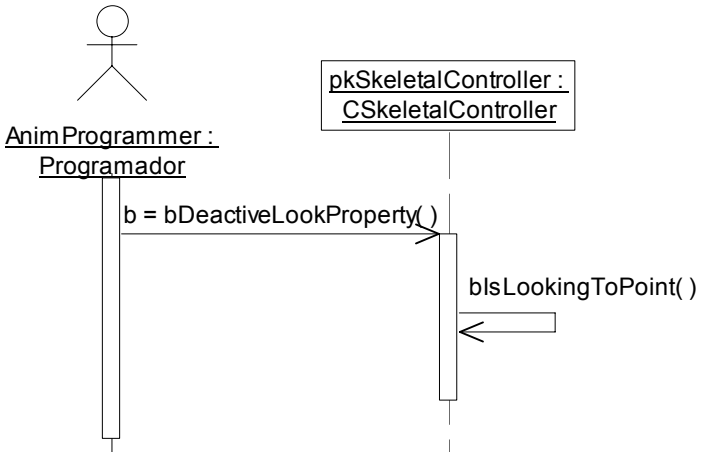


Figura 37: Diagrama de secuencia Desactivar Propiedad Visión.

El diagrama anterior corresponde a la sesión *Deshabilitar capacidad de visión* del caso de uso *Configurar capacidad de visión.* En él si se está mirando un objetivo se desactiva esta propiedad.

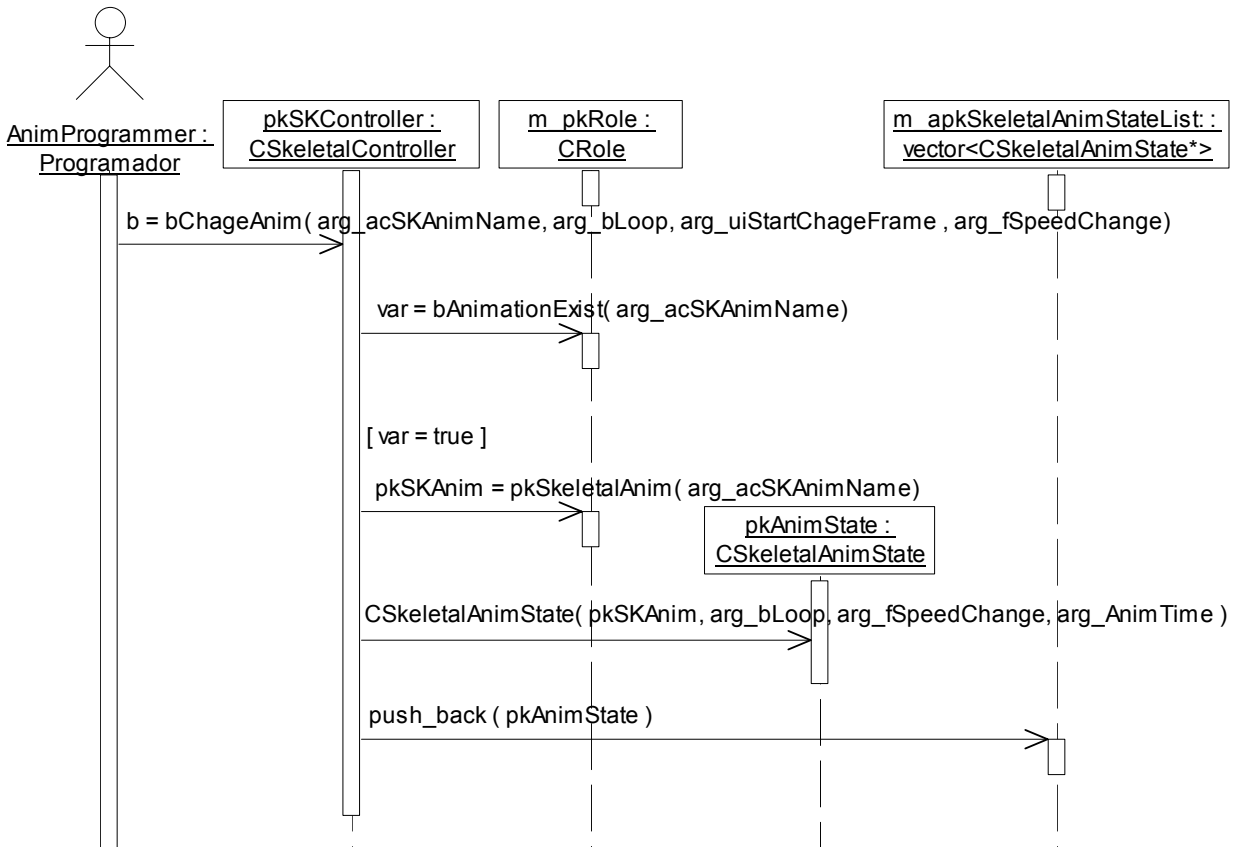


Figura 38: Diagrama de secuencia Configurar Transición.

El diagrama anterior corresponde al caso de uso *Configurar transición de animación.* En él se especifica la animación a la que se hará el cambio, fotograma en el cual comenzará la mezcla y velocidad en la que ocurrirá el proceso de interpolación. Luego se busca la animación, se crea un estado para ella y se agrega a las animaciones que el personaje está reproduciendo.

Figura 39: Diagrama de secuencia Actualizar Recorrido.

El diagrama anterior corresponde al caso de uso *Actualizar Recorrido.* En él se relacionan los objetos que controlan el camino. Primeramente se comprueba si es la primera animación, si es así se inicializa el

controlador. Después se actualizan los vectores por el que se está recorriendo y el siguiente, se mueve la guía que recorre el camino, se mueve al personaje la posición calculada y se orienta según la posición calculada.

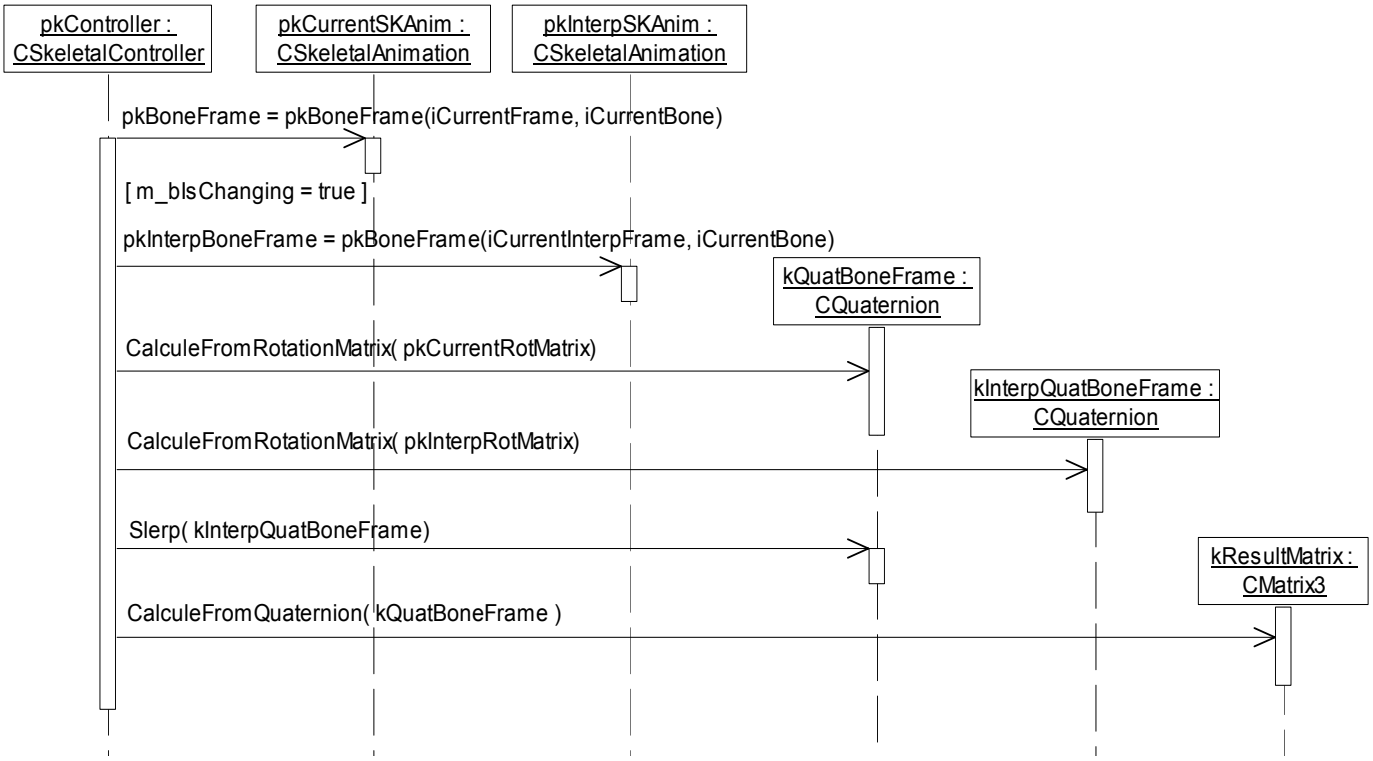


Figura 40: Diagrama de secuencia Actualizar Transición.

El diagrama anterior corresponde al caso de uso *Actualizar Transición.* En él se relacionan los objetos que controlan las animaciones. En forma secuencial se comprueba que el factor de mezcla sea menor que el máximo permitido (debe ser 1.0) luego se buscan las transformaciones de cada hueso en las animaciones que intervienen en el cambio, después se convierten estas transformaciones de matrices a *quaternions* para en el siguiente paso realizar la interpolación utilizando un factor, este es incrementado en la velocidad definida y se convierte la transformación resultante a matriz para que los huesos del personaje sean actualizados. Este proceso se repite en cada actualización de la escena hasta que el factor alcanza el máximo valor.

4.5 Diagramas de componentes.

Las clases necesarias para el módulo de animación se hacen físicas mediante componentes. En este epígrafe se muestran como están agrupados en paquetes los componentes según las clases que estos contienen. Los paquetes mostrados pertenecen a la *SceneToolkit* y a estos se le agregan nuevos componentes con nuevas clases.

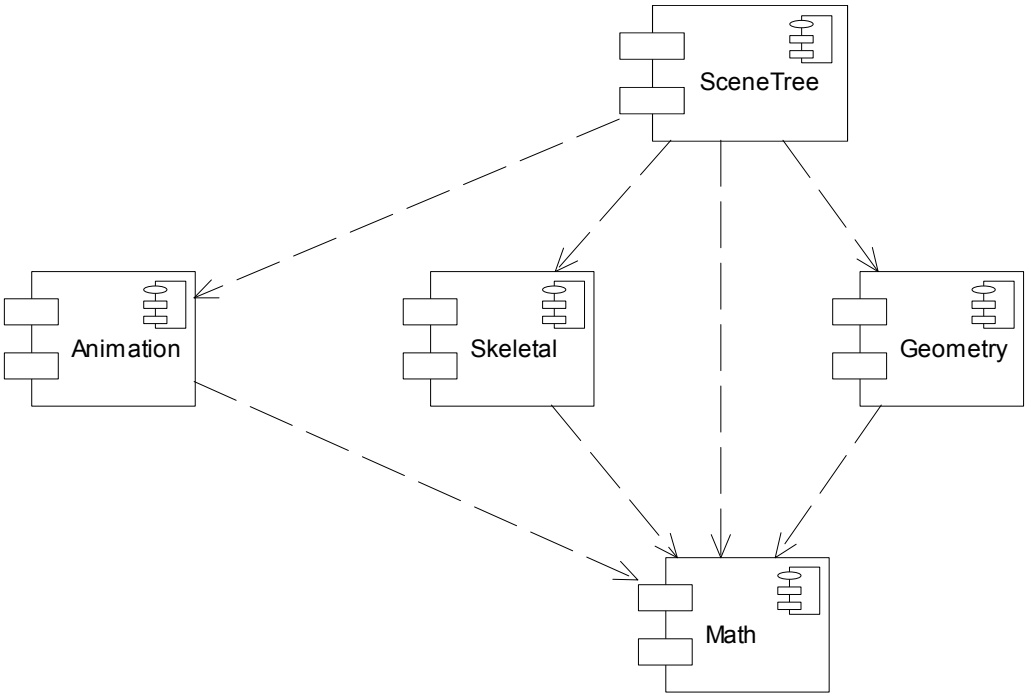


Figura 41: Diagrama de paquetes de componentes de la *SceneToolKit*.

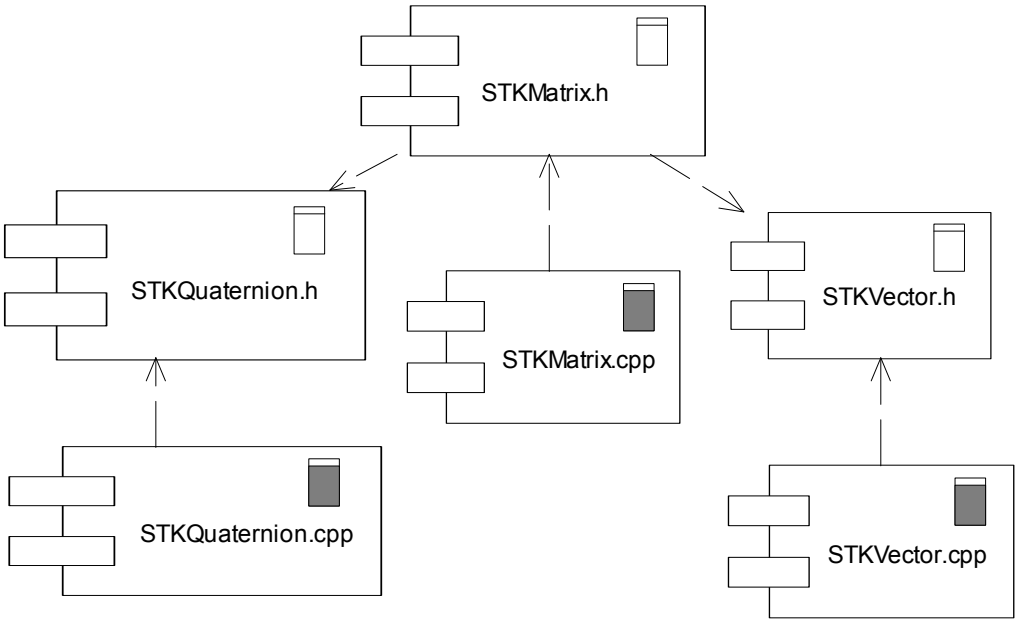


Figura 42: Diagrama de componentes del paquete matemático (paquete Math).

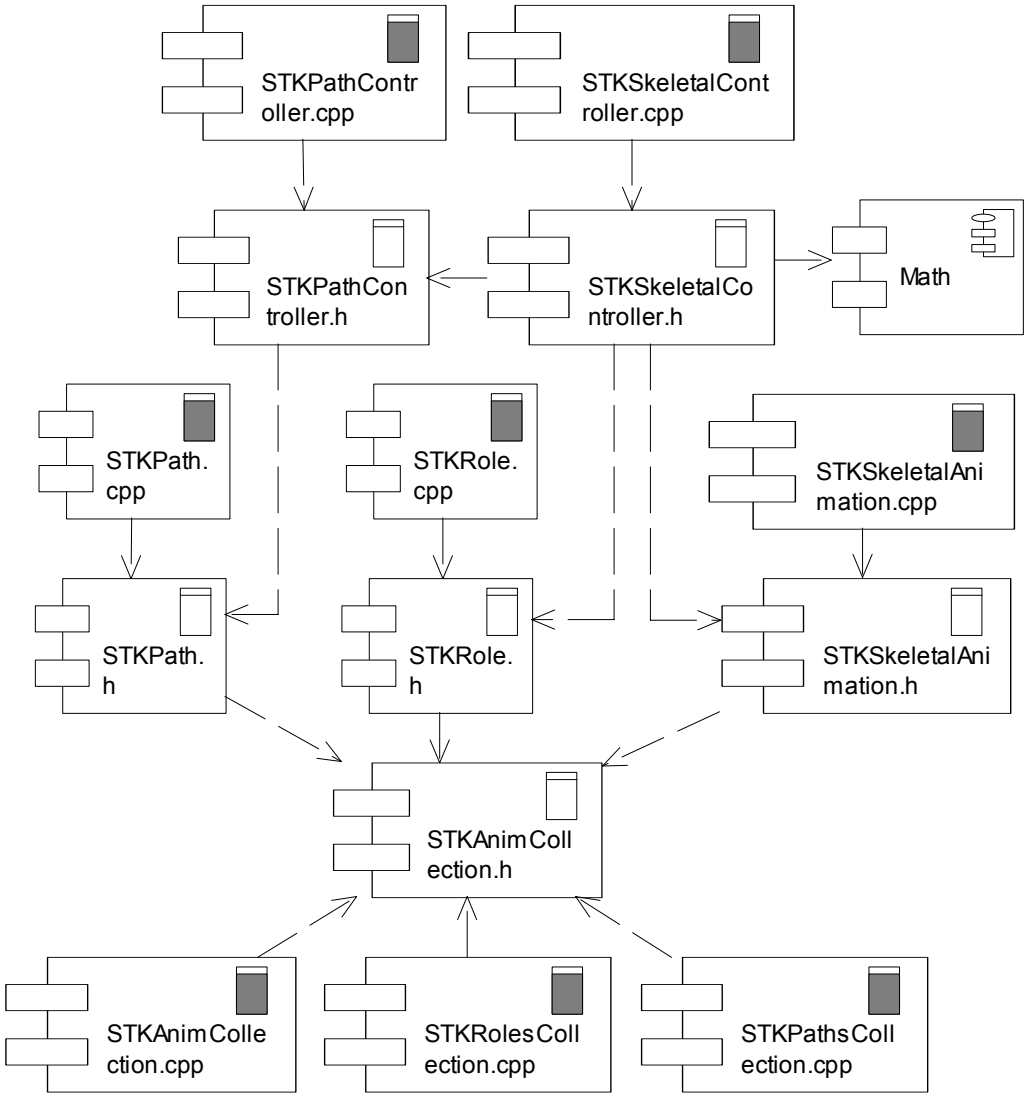


Figura 43: Diagrama de componentes para el manejo de colecciones de animaciones, roles y caminos.

En el anterior diagrama se muestran los componentes correspondientes a varios paquetes especificados en la siguiente tabla. También en la misma se muestran otros componentes de la Herramienta definidos anterior al módulo de animación pero necesarios para este.

Tabla 33: Relación entre paquetes y componentes.

Paquetes	**Componentes incluidos**
SceneTree	*STKNode.h* *STKNode.cpp* *STKGroupNode.h* *STKGroupNode.cpp* *STKSkeletalNode.h* *STKSkeletalNode.cpp* *STKBoneNode.h* *STKBoneNode.cpp*
Animation	*STKRole.h* *STKRole.cpp* *STKPath.h* *STKPath.cpp* *STKSkeletalAnimation.h* *STKSkeletalAnimation.cpp* *STKAnimCollection.h* *STKRoleCollection.cpp* *STKPathCollection.cpp* *STKAnimCollection.cpp* *STKSkeletalController.h* *STKSkeletalController.cpp* *STKPathController.h* *STKPathController.cpp*
Skeletal	*STKSkeletalModel.h*

	STKSkeletalModel.cpp *STKVertexWeight.h* *STKVertexWeight.cpp* *STKBone.h* *STKBone.cpp*
Geometry	*STKTriMesh.h* *STKTriMesh.cpp*
Math	*STKVector.h* *STKVector.cpp* *STKMatrix3.h* *STKMatrix3.cpp* *STKQuaternion.h* *STKQuaternion.cpp*

Conclusiones.

A lo largo del este capítulo se mostraron los artefactos necesarios para el desarrollo del módulo de animación, pertenecientes a las etapas de diseño e implementación. Fundamentalmente se definió una arquitectura compatible con la *SceneToolkit* y la mensajería entre las clases en forma secuencial.

Después de ser concebido y detallado el diseño se pasó a detallar cómo se harán físicas las clases de diseño, consolidando el proceso de desarrollo para pasar a la programación de los casos de uso correspondientes al primer ciclo de desarrollo. Con los diagramas de clases y de componentes ofrece la posibilidad al programador de generar el código en el lenguaje definido mediante alguna herramienta *case* como lo es el “Rational Rose”.

Conclusiones.

Una vez concluida la investigación previa al estudio de las animaciones, se comprobó en la práctica que la STK presentaba problemas para el manejo de colecciones de animaciones y la reprentación de estas de forma realista. Con el propósito de ofrecer una solución a esta problemática se desarrolló un módulo capaz de hacer un manejo eficiente de las animaciones tratadas.

El módulo resultante presenta características que lo hacen el complemento idóneo para que la herramienta (STK) sea funcional en cuanto a la manipulación de animaciones. Algunas de estas características son:

- ✓ Es capaz de manipular animaciones realistas cumpliendo con el procesamiento en tiempo real.
- ✓ Brinda una interfaz de fácil uso por parte de los programadores que lo utilicen.
- ✓ Es flexible a futuras actualizaciones posibilitando incorporar mejoras a la STK.

Por tanto, se puede asegurar que como resultado de todo el proceso realizado se obtuvo un producto totalmente funcional y en correspondencia a los requisitos planteados, que permite la manipulación de las animaciones reproducidas por los personajes con amplias facilidades para el programador.

Recomendaciones.

Se recomienda:

- ✓ Implementar el segundo ciclo de trabajo propuesto.
- ✓ Diseñar e implementar grafo de transiciones de animaciones.
- ✓ Diseñar e implementar herramienta para facilitar la construcción y edición de grafos de transiciones de animaciones.
- ✓ Diseñar e implementar especializaciones para los tipos de personajes más comunes en los SRV.
- ✓ Diseñar e implementar interfases para facilitar la comunicación con módulos de inteligencia artificial (IA).
- ✓ Diseñar e implementar módulo para el trabajo con la cinemática inversa.

Referencias bibliográficas.

[1] MENDOZA, C. Animación.

http://www.mendozajullia.com/papers/Animacion15marzo.pdf

[2] VALÉNCIA, D. I. U. D. Ampliación de Informática Gráfica. Tema 5: Animación 3D, 2007.

http://informatica.uv.es/iiguia/AIG/web_teoria/tema5.pdf

[3] RUÍZ, D.; A. SANSANO, et al. Animación en Art of Illusion, 2004.

http://www.dccia.ua.es/dccia/inf/asignaturas/RG/trabajos/trabajo-david-ruiz.pdf

[4] VELA, J. L. Introducción a la Informática Gráfica.Animación, 2005.

http://wwwdi.ujaen.es/~juanjo/download/animacion.pdf

[5] JAUME, U. Curso de Multimedia. Capitulo 9: Animación, 2004.

http://www4.uji.es/~belfern/IS34/

[6] ROMÁN, Y. R. C. y F. J. LÓPEZ. *Biblioteca Gráfica Para Sistemas de Realidad Virtual*. Ciudad de la Habana, 2004. 196 p.

[7] CHOVER, M. Informática Gráfica. Capítulo 3. Transformaciones. 2004. p.

http://www3.uji.es/~jromero/grafica/Documentos/3-TransformacionesIG35.pdf

[8] AUTORES, R. D. Game Programming Gems. Charles River Media. Rockland Massachusetts, 2000. 614 p.

[9] THE SINGER COMPANY, L. F. S. D. Animating rotation with quaternion curves, 2004.

http://doi.acm.org/10.1145/325334.325242

[10] Drägner Ch. “A ChainMail Algorithm for Direct Volumen Deformation in Virtual Endoscopic Simulation”. Vienna University of Technology. 2005

http://www.cg.tuwien.ac.at/research/publications/2005/draeger-2005-chain/draeger-2005-chain-PDF.pdf

[11] SUÁREZ, P. Animación y Visualización de Fenómenos Naturales, 2004.

http://www.pue.udlap.mx/~tesis/lis/suarez_r_pk/capitulo3.pdf

[12] MAESTRI, G. *Character Animation 2*. New Riders. 1999. 300 p.

[13] PUTZ, M. y K. HUFNAGL Character Animation for Real-time Applications, 2004.

http://www.cg.tuwien.ac.at/studentwork/CESCG/CESCG-2002/

[14] LANDER, J. Skin Them Bones: Game Programming for the Web Generation., 1998.

http://www.darwin3d.com/gamedev/articles/col0598.pdf

[15] PIPHO, E. y A. LAMOTHE. *3D Model, Game Development Series*. Cincinnati, Ohio, 2003. 200 p

[16] 3DMAX8. Character studio user reference, 2005

[17] MULTON, F.; L. FRANCE*, et al.* Computer animation of human walking *The Journal of Visualization and Computer Animation*, 1999, 10(1): 39 – 54

http://www3.interscience.wiley.com/cgi-bin/abstract/60501346/ABSTRACT

[18] ALBEE, T. LIGHTWAVE 3D (8) CHARACTER ANIMATION BOOK. 2004. 478 p.

http://www.agapea.com/Lightwave-3D-8-Character-Animation-Book-CD-Package-n237603i.htm

[19] MARAFFI, C. *MAYA CHARACTER DEVELOPMENT*. New Riders. 2003. 408 p.

[20] ALIAS Alias MotionBuilder 6 User's Guide 2004.

[21] ANDAUER, C.; M. BASTIONI*, et al. Blender Documentation - User Guide*. WILEY 2004. 478 p

http://www.blender.org/documentation/htmlI/

[22] Casas Y. Sistema de captura de movimiento. 2006

[23] ROMÁN, Y. R. C. Manual de programadores para Proyecto de Herramientas de desarrollo para Sistemas de Realidad Virtual, 2007, 81p

Bibliografía consultada.

1. Lifshitz., L., Mecánica: Mecánica del sólido rígido. . Reverté ed. 1991, Barcelona.
2. Curello, P.A., Simulación de la Dinámica de los Cuerpos Rígidos en Tiempo Real: Instituto Tecnológico de Buenos Aires.
3. Microsoft, MSDN Library. Quaternion Structure. 2007.
4. Fiedler, G., Game Physics: Integration Basics. Gaffer on Games, 2006.
5. List of games using physics engines.
6. Parejo, J.C., Introducción a la Animación Asistida por Ordenador. 2004.
7. CHRISTOPHER, T., Mathematics for game developers
8. Larman, C., UML y Patrones. Introducción al análisis y diseño orientado a objetos Editorial Félix Varela ed. 2004, La Habana.

Apéndices.

Glosario de abreviaturas

3D: Tres dimensiones o tridimensional.

BVH: Formato de fichero usado para importar los datos de la rotación de las articulaciones desde varios sistemas de captura de movimiento.

BIP: Formato nativo de Character Studio

C++: Lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos (POO). C++ está considerado por muchos como uno de los lenguajes más potentes, debido a que permite trabajar tanto a alto como a bajo nivel.

CSM: Es un formato de fichero ASCII utilizado para importar el dato de la posición de marcadores desde varios sistemas de captura de movimiento.

RV: Realidad Virtual

RUP: Proceso Unificado de Software

STK: *SeceneToolkit*, herramienta para el manejo y representación de escenas tridimensionales.

SRV: Sistemas de Realidad Virtual.

Glosario de términos.

A:

Animación: Simulación de un movimiento creada por la muestra de una serie de imágenes o fotogramas

Animaciones en preproceso o predefinidas: Animaciones en las que el realizador define los diferentes movimientos y luego genera una animación. Ejemplo: dibujos animados 3D.

Animaciones en tiempo real: La animación en tiempo real es aquella en la que se van generando los fotogramas a medida que son necesarios y permite la interactividad.

Animaciones fluidas: Animaciones donde no se perciben saltos en las transformaciones geométricas.

C:

Cuadro: (ver frame)

Cuadros por segundos: velocidad de secuencia de imágenes.

Centro de rotación: punto respecto al cual se le aplica la rotación a un cuerpo.

Comportamiento (*behavior*): cambio de atributos o características de los objetos visibles y no visibles del mundo virtual y que cambian el aspecto visual de la escena logrando una animación

E:

Entorno sintético: mundo virtual

Esqueletos: Estructura jerárquica de huesos.

F:

Frame, Fotogramas o marco: Cada imagen estática que forma la secuencia animada.

G:

Gráfico vectorial: Gráficos en los que su representación se realiza por medio de descripción de "trazos" (líneas, círculos, curvas...)

H:

Huesos: estructura que permite la deformación de un segmento de malla asociado a él.

I:

Interpolación: algoritmo matemático que a partir de varios puntos en el espacio, describe una función que contiene a los puntos intermedios.

Informática gráfica: Rama de la informática que se encarga de representación gráfica.

Inmersión: Sensación que producen los SRV sobre los usuarios cuando se adentran en el entorno virtual.

Información espacial: Datos que especifican posición y rotación de los objetos en las escenas.

M:

Malla: forma de representar un modelo a partir de polígonos. Colección de vértices, aristas y polígonos conectados de forma que cada arista es compartida como máximo por dos polígonos.

Matrices de transformación: matrices definidas para calcular nuevas coordenadas a partir de coordenadas existentes según una determinada transformación gráfica (rotación, trasladación, escalado y reflexión).

P:

Path: Lista de vectores que definen una trayectoria a través de la escena.

Personajes: Objeto animado que presenta un comportamiento definido.

Peso o *Weight*: Indica cuanta influencia tiene un hueso sobre los vértices asociados a él.

Píxel: abreviatura de “picture element”. Es la menor unidad de información de una imagen digital.

Propiedades físicas: propiedades de los objetos del mundo real que se simularán con los objetos virtuales a través de la manera de ejecutar sus tareas, por ejemplo, la masa.

Puntos de observación: Lugar donde ubica la cámara desde donde se observa la escena.

POO: Programación Orientada a Objetos.

Q:

Quaternion: Extensión de los números imaginarios en forma de vector de cuatro componente que permiten la interpolación y la representación de rotaciones.

R:

Realidad Virtual: Término futurista el cuál pretende describir la interacción de los seres humanos en mundos virtuales o simulados.

Rotaciones suavizadas: Rotaciones donde no se perciben saltos, el cambio entre un ángulo y otro es gradual.

Rol: papel que trae consigo un grupo de acciones y que formará parte de los atributos de un personaje animado.

S:

Simulador: Programa computacional basado en cálculos y modelos estadísticos, usados para representar un escenario determinado.

STK: *SeceneToolkit*, herramienta para el manejo y representación de escenas tridimensionales.

T:

Transición de animación: cambio de una animación a otra.

Transición suavizada de animación: cambio de una animación a otra donde se mezclan los fotogramas finales de la primera animación con los del principio de la animación a la que se pasa.

V:

Vector: Estructura que expresa magnitud y dirección.

Vértice: Punto común entre dos arista de la malla.

Índice de figuras y tablas.

Índice de figuras.

Diagramas de caso de uso.

Diagramas de Diseño

Diagramas de Secuencia.

Índice de tablas