Counter Strike 2D

TP Final

Objetivos	● Afianzar los conocimientos adquiridos durante la cursada. ● Poner en práctica la coordinación de tareas dentro de un grupo de trabajo. ● Realizar un aplicativo de complejidad media con niveles aceptables de calidad y usabilidad.
Instancias de Entrega	**Entrega 1:** clase 13 (06/07/2021). **Entrega 2:** clase 15 (20/07/2021).
Temas de Repaso	● Aplicaciones Cliente-Servidor multi-threading. ● Interfaces gráficas ● Manejo de errores en C++
Criterios de Evaluación	● Criterios de ejercicios anteriores. ● Construcción de un sistema Cliente-Servidor de complejidad media. ● Empleo de buenas prácticas de programación en C++. ● Coordinación de trabajo grupal. ● Planificación y distribución de tareas para cumplir con los plazos de entrega pautados. ● Cumplimiento de todos los requerimientos técnicos y funcionales. ● Facilidad de instalación y ejecución del sistema final. ● Calidad de la documentación técnica y manuales entregados. ● Buena presentación del trabajo práctico y cumplimiento de las normas de entrega establecidas por la cátedra (revisar criterios en sitio de la materia).

El trabajo es grupal*: debe ser de autoría completamente del grupo. Cualquier forma de* ***plagio es inaceptable****: copia de otros trabajos, copias de ejemplos de internet o copias de tus trabajos anteriores (self-plagiarism).*
Si usas material de la cátedra deberás dejar en claro la fuente y dar crédito al autor (a la materia).

Índice

Introducción

El presente trabajo consiste en diseñar e implementar una recreación 2D del clásico juego *Counter Strike* [1]. En este, los jugadores se agruparán al bando de los *terroristas (terrorist)* o de los *anti-terroristas (counter terrorist)* en una pelea armada “n versus m” (configurable).

Esta variante del conocido multijugador competitivo tendrá una vista de águila, centrada en el jugador.

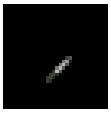
Descripción

El juego consiste en una batalla armada entre dos bandos, donde los terroristas querrán plantar una bomba para ganar la partida, mientras que los anti-terroristas harán lo posible para impedirlo. La partida terminará luego de jugadas diez rondas (configurable). Cada ronda finaliza cuando todos los integrantes de un bando hayan sido eliminados, cuando los terroristas planten la bomba y ésta explote luego de un tiempo, o luego de que los anti-terroristas desactiven dicha bomba.

Armas

El juego dispondrá de 5 armas:

 **Bomba**: Arma exclusiva de los terroristas. Solo hay una por partida y la recibe al empezar el juego de forma aleatoria un integrante del bando terrorista. Si este muere, esta aparece en forma de drop en el piso, para que otro integrante pueda tomarla y activarla. Se debe de activar la bomba en uno de los puntos indicados del mapa (ver sección mapa). Una vez activada, la bomba tarda unos segundos en detonar.

 **Cuchillo**: es el arma por default cuando el jugador se queda sin balas. El ataque se realiza cada vez que el jugador presiona la tecla de disparo. A diferencia del resto de las armas está solo produce daño si el jugador está al lado del enemigo.

 **Glock:** Solo puede disparar de a una bala cada vez que el jugador presione el click izquierdo. Tiene un daño mayor que el cuchillo a una distancia adecuada.

 **AK-47**: Dispara de a 3 balas por rafaga. El jugador puede seguir disparando ráfagas si mantiene presionado el click izquierdo. En este caso las rafagas tendrán una frecuencia de 0.4 balas/sec (no hay un disparo perfectamente contiguo). El daño de las balas impactadas es menor que la pistola.

 **M3:** Hace daño en área, en forma de cono (los proyectiles se dispersan). Tiene una distancia de alcance menor a las anteriores. Solo se puede disparar una vez que el jugador presione el click izquierdo. Su daño es muy grande si el disparo se realiza a corta distancia del objetivo.

 **AWP**: Arma con el mayor alcance. Su disparo, cuando acierta causa el máximo impacto. Tiene un tiempo de retardo entre disparo y disparo alto. Su daño no se ve afectado por la distancia del objetivo. Es el arma más cara del juego.

Tanto el cuchillo como la pistola (y la bomba para los terroristas) vienen en el equipo default de cada jugador al empezar la partida. Las demás pueden ser compradas en la etapa de preparación inicial (ver sección juego).

Cada arma (excepto el cuchillo) tiene una precisión, una probabilidad de acertar en el blanco y producir un daño en el enemigo. La precisión también es modificada según la distancia con el enemigo. Además cuentan con un daño inversamente proporcional a la distancia del enemigo.

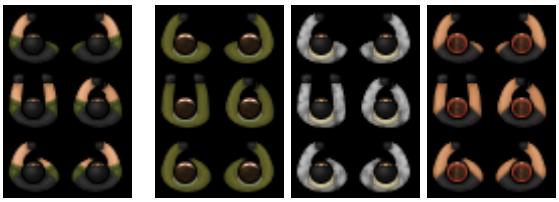
El daño producido tendrá una variación aleatoria por cada impacto de bala. En el caso del cuchillo este causara daño siempre que se esté en el rango de alcance y su daño también será aleatorio entre un rango determinado por el arma (aleatorio entre el mínimo y el máximo de daño).

Cuando la ronda termine, el jugador conservará sus armas y balas, hasta que haya un cambio de bandos (o hasta que termine la partida). Sin embargo, si el jugador es asesinado en una ronda, pierde todas sus armas en forma de drop (ver sección juego).

El jugador puede tener equipado 3 armas: el cuchillo y la pistola (arma secundaria) que vienen por default, y una de las otras tres alternativas (desde ahora, el arma principal). En el caso de los terroristas, a aquel que le toque la bomba, cuenta con 4 slots para armas (el cuchillo, la pistola, un arma mas y la bomba). Estas podrán ser intercambiadas durante el juego con atajos del teclado. El arma optativa puede ser cambiada por otra en la tienda, o tomando un drop que haya dejado un personaje muerto, desequipandosé el arma optativa que tenga (si es que posee una).

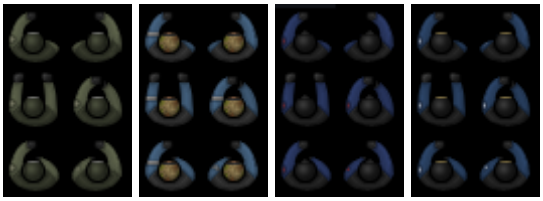
Personajes

Terrorists



Los terroristas solo quieren ver el mundo arder. Su objetivo es la destrucción del sistema capitalista y no pararán hasta destruir los distintos monumentos de las distintas regiones. El jugador podrá seleccionar al comenzar la partida que skin usar (*Pheonix, L337 Krew, Artic avenger o Guerrilla*) Estos personajes no tienen diferencia funcional entre si, solo la diferencia estética.

Counter terrorists



Son el brazo de la ley. Su misión es detener a los rebeldes a toda costa. El jugador tiene la posibilidad de elegir cuál protector utilizar (*Seal force, German GSG-9, UK SAS o French GIGN*).

Todos los personajes cuentan con una vida inicial, que irá disminuyendo a medida que vayan recibiendo impactos. El jugador podrá moverlo usando atajos del teclado y apuntará usando el mouse.

Partida

Las partidas son multijugador. Cada nivel o escenario tendrá un número de jugadores posibles aunque se podrá comenzar la partida con menos de ellos. Cada jugador se conectará al servidor y deberá poder elegir entre crear una partida nueva (donde elegirá el nivel) o unirse a una ya existente.

Una vez que los jugadores se hayan unido, se podrá dar por comienzo la partida. Una vez que la partida comenzó no se puede sumar nadie más.

Juego

El juego consiste en 10 rondas, 5 en cada bando. El equipo que arranque como terrorista, luego de 5 rondas se pasará automáticamente al bando de los anti-terroristas y viceversa. Al finalizar las 10 rondas, el equipo que haya tenido más victorias será el ganador.

Cada ronda cuenta con distintas fases. La fase inicial, o de preparación, es una etapa donde nadie puede moverse, y cuentan con una cantidad de segundos para comprar el armamento necesario y las balas de estas armas para la ronda.

Las armas tienen un precio. Un jugador puede tener 3 armas en su equipo (4 si son terroristas con la bomba): el cuchillo, la pistola y un arma principal extra. El cuchillo y la pistola son el equipamiento por default del personaje (no necesita comprarlo). Todo el equipamiento que posea el jugador lo conserva entre las distintas rondas, salvo que haya sido asesinado, en cuyo caso pierde todo. Si un jugador ya posee un arma principal, y se compra otra, deja la anterior en forma de drop en el mapa.

Las armas cuestan dinero, cada jugador cuenta con un saldo, que va aumentando en cada ronda con una suma inicial, y también cuenta con bonificaciones. Estas bonificaciones son obtenidas al ganar una ronda, o al matar a un rival.

Pasado el tiempo de preparación, el equipo podrá moverse libremente en la fase final de la ronda. Esta fase terminará si:

- Todo el equipo rival muere
- Los terroristas activaron la bomba en los monumentos, y esta explota.
- Los antiterroristas desactivan la bomba.

Si los terroristas plantan la bomba, pero son eliminados por la fuerza de la ley, todavía pueden ganar la ronda si es que no logran desactivar la bomba a tiempo.

Luego de cinco rondas, se invierten los roles de cada equipo (los terroristas pasan a ser anti-terroristas y viceversa). Luego de cinco rondas más (10 en total) la partida finaliza, y aquel equipo con más rondas ganas es declarado el campeón. Al finalizar la partida se mostrará un tablero de puntuaciones, con la cantidad de asesinatos, muertes y dinero recolectado de cada jugador, en un ranking por cada equipo.

Items

En el mapa puede haber armas en el suelo, para que cualquier usuario lo pueda recolectar al pasar por encima de él. Estos ítems en el suelo pueden provenir:

- Un drop de un personaje muerto.
- Un drop debido a que un personaje compró otra arma principal.
- El mapa contaba con items desperdigados por ahí.

Visión de campo (field of view)

El juego original, al ser en primera persona, te imposibilitaba ver a los enemigos que se encuentren a tus espaldas. Para simular esto, se implementará un *field of view* del jugador. Esta visión de campo consta de un cono de visión, de ángulo configurable, donde el jugador puede ver. Para evitar cualquier tipo de diseño con luces y sombras (temas que se verán en otra materia: Sistemas Gráficos), se pedirá que implementen un *stencil buffer[2]*.
Un stencil es una técnica de pintura donde se superpone una lámina sobre el papel a pintar. El pintor empieza a dibujar, típicamente con un aerosol, y se genera un negativo del stencil en el lienzo.

Nosotros haremos una simplificación de esto. Se creará una textura de forma dinámica (dibujando un triangulo de angulo configurable de color blanco sobre un fondo negro) para soperponerla al mapa usando la técnica de alpha blending [3][4]. Esta zona de visión deberá rotar en función de a donde está apuntando el

jugador.

Nótese que en la imagen, el FOV solo “blurea” la zona detrás del jugador. No se hace ninguna lógica de dibujado condicional (primero se dibujará el mapa y los elementos dinámicos en su totalidad, luego se lo filtrará con el stencil para oscurecer la zona que el personaje no está viendo). Tanto la opacidad como el ángulo de FOV deben ser configurables (cuanto más opaco, menos transparente será la zona detrás del jugador). El FOV afecta al jugador, no se acumulará el FOV de los aliados.

Escenario

Cada mapa tendrá un terreno característico. Este podrá ser un campo en el desierto, un pueblito azteca o una zona de entrenamiento.
El terreno está compuesto de campo abierto, para que los jugadores puedan batallar y moverse libremente.

El mapa cuenta también con una serie de estructuras sólidas, que servirán de protección contra los disparos y por las cuales los jugadores no podrán traspasar (ver sección Física del juego).

Además, cada mapa cuenta con dos zonas donde los terroristas pueden activar las bombas, estas zonas

estarán debidamente indicadas.

Física del juego

El juego tendrá un motor físico que se encargue de simular el movimiento, la detección de colisiones y el lanzamiento de proyectiles. Será requisito obligatorio que este motor no sea implementado por ustedes. Se debe usar una librería de terceros. Se recomienda el uso del *framework* Box2D [5].

Interfaz de usuario

El juego muestra el escenario, objetos y a otros jugadores desde una perspectiva de *vista de águila*. El escenario siempre mostrará al jugador en la zona central del mapa. El jugador puede moverse usando el teclado y apuntar y disparar usando el mouse.

La interfaz debe mostrar también el arma equipada, los puntos de vida actuales, el dinero, las balas, una mira para poder apuntar y el tiempo que queda para la detonación de la bomba (si esta fue activada)

El cliente debe poder mostrar el juego en fullscreen y en modo ventana siendo el modo definido en el archivo de configuración. En ambos casos la resolución de la pantalla también será determinada por archivo de configuración y serán valores típicos como 320x200 o 640x400

La imagen está a efectos ilustrativos. No se requiere que la UI sea exactamente igual a esta.

Configuración

Todos los valores numéricos (constantes numéricas) deben venir de un archivo de configuración de texto YAML[6]: cantidad de balas máxima, vida, frecuencia de disparo, etc.

Tener los parámetros en un solo lugar les permitirán hacer modificaciones al juego si este está desbalanceado y hace que el juego no sea divertido.

Un archivo de configuración le permitirá a ustedes hacer *ajustes* más finos sin recompilar y al docente le permitirá cambiar ciertos valores para facilitar la corrección (por ejemplo se puede poner vida infinita).

Aclaración: No se debe implementar un parser YAML, sino que se debe investigar y utilizar uno ya existente.

Sonidos

Como todo juego se debe reproducir sonidos para darle realismo a los eventos y acciones que suceden[7]:

- Cuando hay un disparo se debe emitir el sonido característico.
- Cuando hay una muerte se debe emitir un sonido acorde.
- Cuando se active la bomba

Si la cantidad de eventos que suceden es muy grande, algunos sonidos deben ser evitados para no saturar al jugador con tanta información.

El volumen de los ruidos deberá estar modulado por la distancia: eventos cercanos producirán sonidos o ruidos más fuertes.

Musica

Se debe reproducir una música de fondo.

Animaciones

Las explosiones, los disparos y todo que sea dinámico tendrá que ser animado [7].

Aplicaciones Requeridas

Servidor

El juego a implementar es multijugador con una arquitectura cliente-servidor. El servidor deberá recibir por parámetro la ruta a un archivo de configuración que contendrá:

- Puerto donde escuchar
- Parámetros del juego: todos los valores numéricos descritos en el presente enunciado deben ser configurables desde un archivo de texto. Esto es ***esencial*** para optimizar la jugabilidad y balancear las fortalezas y debilidades de las unidades sin tener que recompilar el código.

La definición del protocolo de comunicación entre el cliente y el servidor queda a libre elección. Se requiere sin embargo que sea binaria (no puede ser texto) y no pueden usarse librerías de terceros.

Se debe soportar múltiples partidas a la vez.

Cliente

Es el elemento central que interactúa con el jugador. Debe poder mostrarle una pantalla de login para definir a qué servidor quiere conectarse, si quiere crear una partida o unirse a una.
Ya iniciado el juego, debe ofrecer una interfaz rica de acuerdo a los requerimientos pedidos.

Editor

Junto con la aplicación cliente-servidor se deberá entregar una tercer aplicación: un editor de niveles o escenarios.

Se deberá poder diseñar los mapas colocando terreno, objetos, armas. Además se deberá poder indicar en donde arrancará cada equipo, y cuales son las zonas para plantar bombas. El editor deberá validar que se tengan dos zonas de inicio para los equipos, y que cuente con al menos 1 zona para plantar la bomba.

El editor deberá poder abrir y guardar los mapas. Estos deberán estar en formato YAML.

Los mapas pueden ser más grandes de los que el editor puede mostrar en pantalla por lo que se tendrá que soportar scrolling.

Además deberá implementarse point and click y drag and drop como formas para editar el mapa (donde mejor convenga para el usuario).

Distribución de Tareas Propuesta

Con el objetivo de organizar el desarrollo de las tareas y distribuir la carga de trabajo, es necesario planificar las actividades y sus responsables durante la ejecución del proyecto. La siguiente tabla plantea una posible división de tareas de alto nivel que puede ser tomada como punto de partida para la planificación final del trabajo:

	Alumno 1 **Servidor - Modelo**	**Alumno 2** **Modelo - Cliente**	**Alumno 3** **Editor - Cliente**
Semana 1 (08/06/2021)	- Carga de mapas. - Lógica de movimiento de los personajes (colisión con las paredes y otros objetos).	Mostrar una imagen. Mostrar una animación. Mostrar ambas en un lugar fijo o desplazándose por la pantalla (movimiento).	- Dibujar el stencil - Reproducir sonidos, música. - Mostrar texto en pantalla.

Semana 2 (15/06/2021)	- Lógica de las partidas. - Lógica de ataque.	Mostrar todo el mapa, incluyendo varios tipos de terrenos distintos, objetos y unidades	- Integrar el stencil al game-loop - Draft del editor.
Semana 3 (22/06/2021)	- Sistema de comunicación (cliente - servidor)	Mostrar los objetos (drop). Interacción por parte del usuario.	- Editor básico
Semana 4 (29/06/2021)	- Servidor completo. - Configuración.	- Cliente completo.	- Editor completo. - Pantalla de login y de partidas.
Semana 5 (06/07/2021)	- Pruebas y corrección sobre estabilidad del servidor. - Detalles finales y documentación preliminar	- Pruebas y corrección sobre estabilidad del cliente. - Detalles finales y documentación preliminar	- Pruebas y corrección sobre estabilidad del cliente. - Detalles finales y documentación preliminar
Primera Entrega el día 06/07/2021			
Semana 6 (13/07/2021)	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación
Semana 7 (20/07/2021)	- Testing - Documentación - Armado del entregable	- Testing - Documentación - Armado del entregable	- Testing - Documentación - Armado del entregable
Entrega Final el día 20/07/2021			

Restricciones

La siguiente es una lista de restricciones técnicas exigidas por el cliente:

1. El sistema se debe realizar en POSIX 2008 C++11 utilizando librerías *SDL y/o Qt*.
2. Con el objetivo de facilitar el desarrollo de las interfaces de usuario, se permite el uso de *QtDesigner* u otro editor de interfaces gráficas.
3. Se debe usar una biblioteca de parsing YAML. No se puede implementar un parser propio.
4. Es condición necesaria para la aprobación del trabajo práctico la entrega de la documentación mínima exigida (consultar sitio de la cátedra). Es importante recordar que cualquier elemento faltante o de dudosa calidad pone en riesgo la aprobación del ejercicio.
5. El trabajo deberá contar con un instalador.
6. De forma opcional, se sugiere la utilización de alguna librería del estilo xUnit. Si bien existen varias librerías disponibles en lenguaje C++, se recomienda optar por CxxTest [8] o CppUnit [9]
7. Todo socket utilizado en este TP debe ser bloqueante (es el comportamiento por defecto).
8. El protocolo de comunicación tiene que ser binario (no texto) y no puede usarse una librería de terceros.

Referencias

[1] https://en.wikipedia.org/wiki/Counter-Strike
[2] https://en.wikipedia.org/wiki/Stencil_buffer
[3] https://lazyfoo.net/SDL_tutorials/lesson27/index.php
[4] https://github.com/Taller-de-Programacion/clases/blob/master/bibliotecas-gui/sdl/src/mainSdlAlphaBlending.cpp
[5] https://box2d.org/documentation/index.html
[6] https://yaml.org/
[7] https://mega.nz/file/WoJCwTZK#8DJUTQBxLEiU-ejM_iF0NPpeuTYY6Yndo1ORgtJwh1g
[8] https://cxxtest.com/
[9] https://en.wikipedia.org/wiki/CppUnit
[10] https://cxxtest.com/
[11] https://en.wikipedia.org/wiki/CppUnit
[12] https://lazyfoo.net/tutorials/SDL/08_geometry_rendering/index.php