|  |  |
| --- | --- |
| Descripción: tempimage1 | **FACULTAD DE CIENCIAS NATURALES Y EXACTAS**  **DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN**  **Trabajo Final**  **Arquitectura de Computadoras**  **SEGUNDO AÑO LICENCIATURA EN CIENCIA DE LA COMPUTACIÓN**  **Autores**:  Juana Iris Pérez Salas  Alejandro Fonseca Cuza  Alvaro de Jesus Reyes  Carlos Antonio Velazques  **Tutor**: Wilkie  **2023** |

**Resumen**

El presente trabajo tiene como objetivo aplicar la programación a bajo nivel para recrear el juego de Pocket Tanks . Como herramientas para la creación del juego se utilizó MIPS arquitectura de microprocesador basada en la arquitectura RISC . El desarrollo fue realizado en el MARS. Se obtuvo como resultado un juego similar a Pocket Tanks que posee un mapa aleatorio para cada inicio del juego, dos tanques, y 5 tipos de balas distintas para crear diversas estrategias y acabar con el oponente.

**Abstract**

The present work has as objective to apply the programming at low level to recreate the game of Pocket Tanks. As tools for the creation of the game you uses MIPS microprocessor architecture based on the architecture RISC. The development was carried out in the MARS. It was obtained a very similar game as a result to Pocket Tanks that possesses a random map for each beginning of the game, two tank, and 5 types of different bullets to create diverse strategies and to put an end to the opponent.Contenido

[1. Introducción 6](#_Toc22463)

[1.1 Enunciado del problema a resolver. 6](#_Toc4843)

[1.2 Análisis del problema. 7](#_Toc12175)

[1.3 Objetivo del trabajo. 7](#_Toc31321)

[2. Desarrollo 9](#_Toc22092)

[2.1 Modelos matemáticos de la solución. 9](#_Toc29231)

[2.2 Descripción de los algoritmos elaborados. 10](#_Toc8515)

[2.3 Implementación 11](#_Toc5633)

[3. Conclusiones 14](#_Toc3573)

[4. Recomendaciones 15](#_Toc29172)

[Bibliografía 16](#_Toc1654)

# Introducción

El estudio y uso de la programación en lenguaje ensamblador puede ser una de las tareas más desafiantes de un Ing. de Software; pero, aunque sus características les hacen un lenguaje difícil, programando en ensamblador nos convertiremos prácticamente en artesanos del desarrollo de programas, ocupándonos de todas las tareas en las que deseemos intervenir personalmente.

Al programar en ensamblador no sólo se utiliza el conjunto de instrucciones y registros de un cierto microprocesador sino que, además, se usarán dichas instrucciones para acceder a elementos hardware, como el teclado o los buses de comunicaciones de una cierta arquitectura de ordenador.

MIPS (Microprocessor without Interlocked Pipeline Stages) es una arquitectura de microprocesador basada en la arquitectura RISC (Reduced Instruction Set Computing). En la actualidad, ha perdido gran parte de su popularidad en el mercado de los procesadores de propósito general, pero sigue siendo ampliamente utilizada en sistemas integrados y en la industria de la electrónica de consumo. Además, MIPS ha sido utilizado en la creación de procesadores para dispositivos móviles, routers de red y otros dispositivos especializados.

## Enunciado del problema a resolver.

Utilizando MARS, simulador de la arquitectura MIPS programar un juego basado en Pocket Tanks, juego de artillería multijugador basado en turnos.

Este juego tiene que cumplir las siguientes funcionalidades:

* Crear un terreno en 2d con elevaciones y obstáculos.
* Determinar cuándo un jugador derrota al tanque enemigo y anunciar el ganador.
* Crear diferentes tipos de proyectiles con diferentes efectos.
* Crear un sistema de turno para los jugadores.
* Dar la posibilidad de ajustar el ángulo, y la potencia del disparo.

## Análisis del problema.

El juego basado en Pocket Tanks y programado en MARS plantea el desafío de implementar una versión en lenguaje ensamblador MIPS de un juego de artillería multijugador basado en turnos.

El objetivo principal del juego es derrotar al tanque enemigo alcanzándolo con diferentes proyectiles lanzados desde un tanque propio. Para lograr esto, será necesario programar el sistema de turnos, donde cada jugador podrá ajustar el ángulo y la potencia del disparo, así como seleccionar la munición adecuada, y luego lanzar el proyectil hacia el tanque enemigo.

La implementación del juego requerirá tener en cuenta las reglas y mecánicas específicas de Pocket Tanks, como el uso del terreno y los obstáculos para protegerse o lanzar ataques estratégicos, y la variedad de armamentos disponibles, cada uno con diferentes efectos y propiedades.

Uno de los desafíos técnicos más importantes será la gestión de las físicas de los proyectiles y su interacción con el terreno y los obstáculos. Será necesario calcular la trayectoria y la colisión de los proyectiles teniendo en cuenta la potencia y el ángulo. Además, deberá implementarse un sistema de detección de colisiones para determinar si un proyectil ha alcanzado con éxito al tanque enemigo.

La planificación y estructura del código serán aspectos clave para abordar el desarrollo del juego en MIPS. Será necesario definir funciones y subrutinas que permitan gestionar las interacciones entre los jugadores, los proyectiles y los elementos del entorno del juego.

Problema concreto: Complejidad y variedad de las funcionalidades que dificultan la implementación de un juego basado en Pocket Tanks programado en MIPS en Mars.

## Objetivo del trabajo.

Partiendo de los elementos antes expuesto se propone como objetivo del trabajo: Desarrollar un juego usando MIPS que permita las funcionalidades básicas del juego Pocket Tanks.

El juego propuesto permitirá a los jugadores disfrutar de un terreno diferente en cada partida, diversos tipos de balas para armar estrategias, un dominio parcial de la potencia y ángulo de disparo, y una pequeña influencia del viento que hará necesario pensar un poco más para realizar un disparo acertado.

Para dar cumplimiento al objetivo se desarrollarán las siguientes tareas

1. Modelación del juego
2. Diseño gráfico del juego
3. El estudio de las herramientas de MARS y del funcionamiento de la arquitectura MIPS
4. Implementación del juego

# Desarrollo

En este acápite se abordará lo relativo a la modelación matemática de la solución del problema, se describe los algoritmos para los procesos más relevantes, además de una breve descripción de sus funcionalidades.

## Modelos matemáticos de la solución.

La representación de la solución dada al problema de la complejidad y variedad de las funcionalidades se aborda desde la teoría de funciones y matemática:

Movimiento de la bala:

x = ( x0 + v0x\* t + (Fx / m)\* t^2 ) /2

y = ( y0 + v0y\* t + (Fy / m - g )\* t^2 ) /2

Siendo x: posición de la bala en x y: posición de la bala en y

x0: posición inicial de la bala en x y0: posición inicial de la bala en y

m: masa de la bala v: velocidad(depende del ángulo de disparo) t: tiempo……ciclo

g: gravedad 9.8

Fx: componente horizontal de la fuerza del viento(depende del ángulo y velocidad del viento)

Fy: componente vertical de la fuerza del viento(depende del ángulo y velocidad del viento)

Generación aleatoria del mapa:

h = (h1 + h2)/2 + f

f = f/2

Siendo h : altura de la elevaciones un punto, h1 y h2 las alturas de los extremos que en inicio se toman random y luego se actualizan y f un factor aleatorio, este proceso se repite por todo el mapa.

## Descripción de los algoritmos elaborados.

Para dar solución al problema se diseñaron diferentes algoritmos entre ellos:

Función para calcular donde se debe pintar la bala en x

Función: CalcularX(t) calcula la posición en x sin tener en cuenta el viento(se le agrega la influencia del viento después)

Entrada: Se carga la posición inicial del arma, y se entra el tiempo

Salida: posición en x en $a0

CalcularX(%t)

Cargar posición del arma en $a0

Cargar ángulo del arma en $a1

Velocidad($a0,$a1) calcular la velocidad en x, se guarda en $t0

Multiplicar $t0 \* %t y guardar en $t0

Adicionar $t0 + $a0 y guardar en $a0

Retornar a donde se hizo el llamado

Función para escuchar las entradas del teclado y mover el tanque

Función: escuchador

Entrada: un evento en el teclado

Salida: una orden para mover el tanque

escuchador:

Si $t0 = 0 ir a fin\_escuchador

if\_a: # para la letra a

Cargar en $t7 el código de la letra presionada

Si $t7 = 65 ir a NCa

Si $t7 != 97 ir a fin\_a

NCa:

Cargar en $a0 posicion\_del\_tanque1\_x

Cargar en $a1 posicion\_del\_tanque1\_y

Cargar en $t8 posicion\_del\_arma\_tanque1#Posicion para pintar el arma [0-8]

Llamar a BorrarTanque

Cargar en $s5 posicion\_del\_tanque1\_x

Restar a $s5 -1 y guardar en $s5

Salvar $s5 en posicion\_del\_tanque1\_x

fin\_a:

Este algoritmo permite mover el tanque un paso a la izquierda al presionar la a

## 2.3 Implementación

MIPS (Microprocessor without Interlocked Pipeline Stages) es una arquitectura de microprocesador basada en la arquitectura RISC (Reduced Instruction Set Computing). Fue desarrollada por MIPS Computer Systems Inc. en la década de 1980 y se utilizó ampliamente en sistemas integrados, estaciones de trabajo y servidores de gama alta.

La arquitectura MIPS se caracteriza por su conjunto de instrucciones simple y eficiente, su estructura de registro de 32 bits y su enfoque en la ejecución de instrucciones en paralelo. Esto permite una alta velocidad de procesamiento y una eficiencia energética mejorada.

La aplicación implementada tiene los siguientes requerimientos funcionales

1. Manipulación de la interfaz gráfica: El juego tiene elementos visuales que representan los tanques, los proyectiles y el terreno.

2. Implementación de la física: Los proyectiles lanzados en el juego siguen trayectorias balísticas y están sujetos a las fuerzas de la gravedad y la resistencia del aire.

3. Gestión de los turnos: El juego se basa en turnos, lo que implica que cada jugador tiene la oportunidad de ajustar el ángulo y la potencia del disparo antes de lanzar el proyectil.

4. Selección y uso de armamentos: El juego cuenta con una amplia variedad de armamentos, cada uno con diferentes atributos y efectos.

5. Detección de colisiones: El juego es capas de detectar las colisiones con los tanque y aplicar el efecto correspondiente de la bala

# bandicam 2023-05-31 20-33-22-648

Trayectoria de la bala según su potencia.

# 

# La figura muestra la interfaz del juego

# 1: Vidas del tanque rojo

# 2: Tipo de bala

# 3: Potencia de la bala

# 4: Angulo del viento

# 5: Potencia del viento

Controles del juego:

Tanque uno:

Letra A: Retroceder el tanque

Letra D: Avanzar el tanque

Letra W: Mover el arma del tanque hacia arriba

Letra S: Mover el arma del tanque hacia abajo

Letra E: Bajar la potencia de la bala a disparar

Letra R: Subir la potencia de la bala a disparar

Letra Q: Cambiar el tipo de bala

Tanque dos:

Letra L: Retroceder el tanque

Letra J: Avanzar el tanque

Letra I: Mover el arma del tanque hacia arriba

Letra K: Mover el arma del tanque hacia abajo

Letra U: Bajar la potencia de la bala a disparar

Letra Y: Subir la potencia de la bala a disparar

Letra O: Cambiar el tipo de bala

# Conclusiones

* El juego desarrollado cumple con los requisitos funcionales previsto para parecerse al Pocket Tanks original.
* A partir del estudio realizado sobre el MARS y MIPS se pudo hacer uso de herramientas como el simulador del teclado y de la pantalla, permitiendo que el juego sea cómodo de jugar.
* Se utilizo la pila del sistema y el espacio de datos para solucionar la insuficiencia de registros para guardar todos los datos del juego, permitiendo que el juego tenga todas las variables requeridas para su correcto funcionamiento.

# Recomendaciones

* Incluir nuevas funcionalidades al juego.
* Hacer un ejecutable del juego.

# Bibliografía

1. Luis Rincón,ESTRUCTURA DE COMPUTADORES.
2. https://mcs.utm.utornto.ca/258/doodlejump.html.