Progetto Highway per il corso di informatica e sistemi in tempo reale

Rahela Daila, matricola 603044, r.daila@studenti.unipi.it

1 Descrizione del progetto

La relazione illustrerà l'implementazione di una simulazione che rappresenta una sezione di autostrada, dove l'utente ha la possibilità di creare diversi tipi di veicoli, tra cui motociclette, automobili e camion, considerandoli come task interattivi. Ogni categoria di veicolo è caratterizzata da un comportamento autonomo e risponde a regole di guida specifiche; ad esempio, i camion sono vincolati a non superare gli altri veicoli e a rispettare limiti di velocità inferiori rispetto agli altri. Per garantire una guida realistica, ogni veicolo è dotato di sensori in grado di rilevare le corsie e gli altri veicoli presenti sulla carreggiata. Inoltre, l'interfaccia consente all'utente di visualizzare l'autostrada dall'alto e di zoomare o seguire un veicolo specifico per monitorarne il movimento e le interazioni con gli altri utenti.

2 Modello fisico utilizzato

Quando un veicolo viene inizializzato nel modello, vengono configurati diversi parametri iniziali, inclusi tipo di veicolo, velocità iniziale e corsia di partenza. La selezione del tipo di veicolo può avvenire casualmente o essere specificata dall'utente, ognuno con proprie caratteristiche di velocità e accelerazione. La posizione iniziale del veicolo sulla strada è calcolata considerando larghezza, altezza del veicolo e la corsia assegnata.

I veicoli nel modello possono assumere vari stati come "inattivo", "accelerazione", "decelerazione", "cambio corsia", etc., ognuno con una logica di gestione specifica basata sulle condizioni della strada e sugli altri veicoli nelle vicinanze. Ad esempio, se un veicolo si avvicina troppo a un altro, potrebbe decidere di rallentare o di cambiare corsia.

La logica di guida è basata sulla valutazione delle distanze dai veicoli circostanti e sulla gestione degli stati del veicolo di conseguenza. La presenza di sensori su ogni veicolo consente di rilevare altri veicoli nelle vicinanze e di prendere decisioni di guida informate in base alle condizioni del traffico.

Il modello fisico utilizzato per descrivere le funzioni di movimento dei veicoli nel codice fornito si basa su un modello di moto rettilineo uniformemente accelerato (MRUA) combinato con un modello di movimento bidimensionale. Questa combinazione consente di simulare il movimento dei veicoli su una strada bidimensionale, tenendo conto dell'accelerazione e dell'angolo di sterzata.

Equazioni di aggiornamento della velocità e della posizione del veicolo:
Nel contesto del nostro modello, la velocità del veicolo viene aggiornata utilizzando l'equazione del moto rettilineo uniformemente accelerato:

$$v = v_0 + a \cdot t$$

Dove:

- v è la velocità finale del veicolo.
- v_0 è la velocità iniziale del veicolo.
- a rappresenta l'accelerazione del veicolo.
- t è il tempo trascorso dall'ultimo aggiornamento della velocità, ottenuto tramite `task_get_period(ti)`.

Inoltre, la posizione lungo l'asse x del veicolo (State.pos.x) viene aggiornata utilizzando l'equazione del moto rettilineo uniforme:

$$x = x_0 + v \cdot t$$

Dove:

- x è la posizione finale del veicolo lungo l'asse x.
- x_0 è la posizione iniziale del veicolo lungo l'asse x.
- v è la velocità appena calcolata.
- t è il tempo trascorso dall'ultimo aggiornamento della posizione.

Queste equazioni consentono di determinare la posizione e la velocità del veicolo in base alla sua accelerazione e al tempo trascorso. L'aggiornamento continuo di queste variabili consente al modello di simulare in modo accurato il movimento dei veicoli lungo la strada.

La funzione di sensore di prossimità è fondamentale per la simulazione realistica dei veicoli nell'ambiente virtuale. Utilizzando le coordinate del veicolo, la portata del sensore, l'angolo di rilevamento e il bitmap che rappresenta la scena, fornisce informazioni sulla distanza dal veicolo più vicino lungo una determinata traiettoria. Ciò contribuisce a modellare il comportamento realistico dei veicoli nell'ambiente simulato.

3 Interfaccia utente

L'interfaccia utente è formata da due parti: nella parte superiore vengono rappresentate graficamente le corsie e si vedono i vari veicoli e il loro movimento. Inoltre clickando su un veicolo è possibile anche visualizzare i sensori. Nella parte inferiore vengono mostrate a sinistra le informazioni riguardanti le deadline missed, nel centro tutte le informazioni riguardanti il veicolo che viene clickato, a destra l'elenco dei vari comandi che possono essere scelti dall'utente.

4 Scelte design del codice

Per la realizzazione dell'applicazione è stata creata la libreria ptask e una lista per la condivisioni dei dati tra le varie funzioni. In seguito è stato implementato il codice sorgente.

- ***Ptask***: la libreria "ptask" serve come interfaccia alla libreria standard "pthread", fornendo le strutture e le funzioni necessarie per la gestione di task concorrenti. Queste funzioni includono la creazione e l'attivazione di task, la selezione della politica dello scheduler, l'attesa del prossimo periodo e il monitoraggio delle deadline mancate. Inoltre, fornisce funzioni per semplificare la gestione del tempo e per estrarre e modificare vari parametri dei singoli task, come il periodo o la deadline.
- ***Struttura dati condivisi***: Nel contesto del progetto di simulazione autostradale, è stata sviluppata e implementata una struttura dati condivisa per gestire le informazioni relative ai veicoli all'interno dell'ambiente simulato. Questa struttura è stata progettata per agevolare l'interazione tra i diversi moduli del sistema, garantendo un accesso sicuro e concorrente da parte di più thread.

  La struttura dati principale consiste in una lista concatenata, creata mediante l'utilizzo della funzione `create_shared_list()`. Ogni veicolo presente nella simulazione è rappresentato da un nodo all'interno di questa lista. Sono state implementate funzioni specifiche per l'aggiunta, la rimozione, l'impostazione e l'ottenimento dello stato dei veicoli all'interno della lista. Inoltre, è stata inclusa una funzione per determinare la dimensione attuale della lista.

  Oltre alla struttura dati principale, è stata sviluppata una struttura di supporto nota come "support list", utilizzata per memorizzare temporaneamente informazioni specifiche sui veicoli durante determinate operazioni. Anche questa struttura è stata progettata per garantire un accesso sicuro tramite l'utilizzo di un mutex.

  L'implementazione della struttura dati condivisa ha consentito una gestione efficiente e affidabile delle informazioni relative ai veicoli all'interno della simulazione autostradale. Tale implementazione ha reso possibile un accesso sicuro e affidabile da parte di più componenti del sistema, assicurando al contempo una coerenza dei dati e una gestione ottimizzata delle risorse del sistema.

5 Task coinvolti

I task principali di questo progetto sono tre: *Veicle task*, *user task* e *graphic task*.

1. **Veicle task**: La funzione *veicle task* implementa il comportamento di un veicolo all'interno di un sistema simulato. Inizialmente, essa ottiene gli argomenti necessari al task e attende l'attivazione del suo esecutore. Una volta attivato, procede all'inizializzazione dello stato del veicolo, delle statistiche associate e delle variabili relative alle distanze rilevate dai sensori.

   Successivamente, il veicolo viene aggiunto ad una lista condivisa, fornendo così un meccanismo di sincronizzazione con altre entità del sistema. Durante l'esecuzione del task, il veicolo entra in un ciclo principale che si ripete fino alla sua terminazione. All'interno di questo ciclo, vengono eseguite diverse operazioni.

   Innanzitutto, vengono calcolate la velocità e la posizione del veicolo in base all'accelerazione e all'angolo di sterzata. Questi calcoli avvengono nel contesto delle unità di misura del Sistema Internazionale (MKS). Successivamente, il veicolo esegue le misurazioni dei sensori per rilevare eventuali ostacoli o variazioni nell'ambiente circostante.

   La logica di guida del veicolo viene quindi applicata per interpretare le misurazioni dei sensori e adattare il comportamento del veicolo di conseguenza. Questo include la gestione di situazioni di pausa nel gioco, la modifica della velocità e l'aggiornamento dello stato del veicolo in base alle informazioni rilevate.

   Il veicolo periodicamente verifica se è uscito dallo schermo e, in caso affermativo, si rimuove dalla lista condivisa. Questo garantisce una corretta gestione delle entità presenti nell'ambiente di simulazione. Inoltre, il veicolo controlla se è stata rispettata la scadenza temporale del task, segnalando un eventuale errore in caso di mancata osservanza.

   Infine, al termine dell'esecuzione del ciclo, il task viene pulito e il veicolo termina la sua esecuzione.

2. **User task**: La funzione user task gestisce l'interazione dell'utente con il sistema di simulazione. All'inizio, essa ottiene gli argomenti necessari al task e attende l'attivazione del suo esecutore. Una volta attivato, il task entra in un ciclo continuo che si ripete fino alla sua terminazione.

   Durante l'esecuzione del ciclo, la funzione monitora costantemente l'input dell'utente, sia da tastiera che da mouse. Se l'utente preme un tasto sulla tastiera, la funzione reagisce di conseguenza eseguendo azioni come la creazione di nuovi veicoli, la gestione dello stato di pausa del gioco, lo zoom sulla scena o la regolazione della velocità di simulazione.

   Inoltre, se l'utente preme il pulsante del mouse, la funzione identifica l'elemento della scena su cui è stato effettuato il clic e ne gestisce l'interazione di conseguenza. Ad esempio, se l'utente seleziona un veicolo, viene visualizzato un messaggio di conferma e il veicolo viene marcato come selezionato. Lo stesso avviene per altri elementi come i pulsanti e le strade.

   La funzione gestisce anche la logica di auto-generazione dei veicoli, se abilitata dall'utente. In questo caso, vengono calcolati i tempi di generazione dei veicoli in base alla frequenza impostata dall'utente e vengono creati nuovi veicoli quando scade il tempo stabilito.

   Infine, la funzione controlla se è stata rispettata la scadenza temporale del task e attende il prossimo periodo di attivazione. Quando il ciclo termina, la funzione chiude il gioco, disattivando i task rimanenti e liberando le risorse allocate.

3. **Graphics task**: La funzione graphics task gestisce l'aggiornamento periodico della schermata di gioco, visualizzando la scena di gioco, i veicoli presenti e altre informazioni utili per l'utente.

   Dopo aver ottenuto gli argomenti necessari per il task e aver atteso l'attivazione, la funzione procede creando e inizializzando le bitmap necessarie per la gestione dei buffer grafici.

   Successivamente, all'interno di un ciclo continuo, la funzione si occupa di disegnare la scena di gioco sul buffer grafico dedicato alla scena stessa. Questo include il disegno dello sfondo, dei veicoli presenti nella lista condivisa e dei punti di generazione delle corsie. Tutti questi elementi vengono resi sul buffer grafico della scena.

   Dopo aver disegnato la scena, la funzione determina quale buffer grafico visualizzare sulla schermata principale in base allo stato corrente del gioco. Possono essere visualizzati tre tipi di buffer: la scena principale, una versione ingrandita della scena o una versione ingrandita di un singolo veicolo. A seconda di quale buffer

grafico viene selezionato, la funzione disegna le informazioni aggiuntive corrispondenti, come i dettagli dei veicoli selezionati o le strade ingrandite.

Infine, la funzione si occupa di disegnare altre informazioni come la posizione del mouse e le istruzioni per l'utente sul buffer grafico della schermata principale e trasferisce il contenuto del buffer grafico della schermata sulla schermata reale.

La funzione continua a eseguire questo ciclo fino a quando il flag running è impostato su 1. Quando il ciclo termina, la funzione rilascia le risorse allocate e termina.

6 Diagrammi task risorse

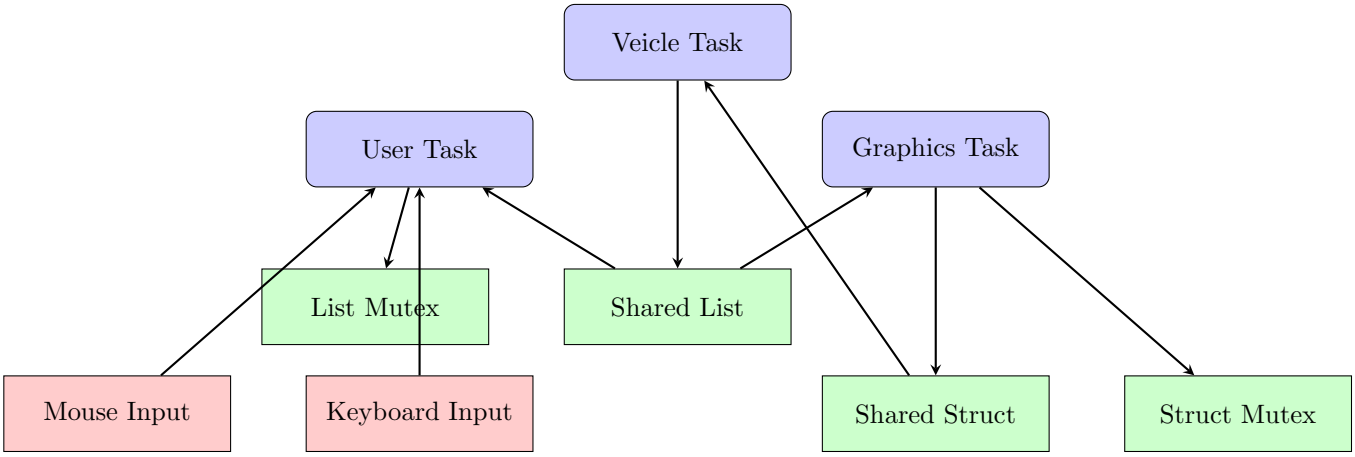

Figure 1: Interazioni tra i task, i buffer e gli input

Elenco delle Interazioni tra Task

- **Veicle Task:**

  Interagisce con:

  Graphics Task per aggiornare la posizione dei veicoli sullo schermo.

  Utilizza i seguenti buffer:

  Shared List per condividere lo stato del veicolo.

  List Mutex per garantire l'accesso sicuro alla lista condivisa dei veicoli.

  Riceve input:

  Non riceve input direttamente da tastiera o mouse.

- **User Task:**

  Interagisce con:

  Veicle Task per creare nuovi veicoli.

  Graphics Task per gestire l'interfaccia utente e il gioco.

  Utilizza i seguenti buffer:

  Shared Struct per gestire lo stato globale del gioco.

  Riceve input:

  Riceve input da tastiera per comandi come spawn, pausa, zoom, ecc.

  Riceve input dal mouse per selezionare veicoli o interagire con l'interfaccia utente.

- **Graphics Task:**

  Interagisce con:

  Veicle Task per ottenere lo stato dei veicoli da visualizzare sullo schermo.

  Utilizza i seguenti buffer:

  Shared List per ottenere lo stato dei veicoli da visualizzare.

  Shared Struct per gestire lo stato globale del gioco e le informazioni sull'interfaccia utente.

  Riceve input:

  Non riceve input direttamente da tastiera o mouse.

7 Parametri dei task

8 Dati sperimentali