Università degli Studi di Napoli
Parthenope

Dipartimento di Scienze e Tecnologie
Corso di Laurea in Informatica

Progetto Reti di Calcolatori
Network Pong

Giuseppe Fusco
Mat. 0124002696

Anno Accademico 2023/2024

Indice

1 Traccia

Implementare un gioco basato su pong. Il gioco è basato su comunicazione in tempo reale usando UDP.

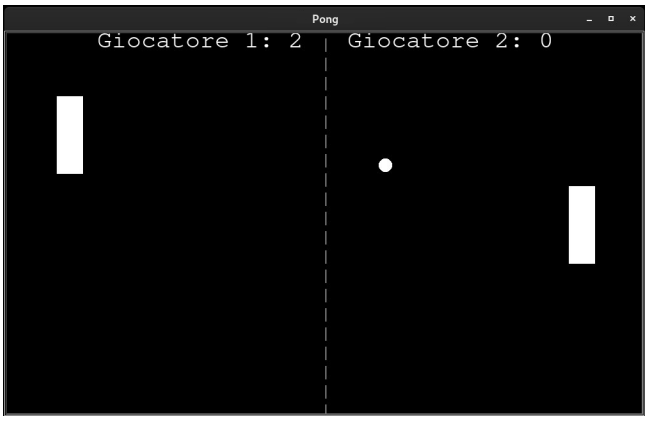
Rappresentare la griglia di gioco tramite una matrice 2D, non è richiesta fluidità nei movimenti. La pallina si sposta di una cella ogni X millisecondi, lo spostamento è determinato dalla macchina dell'host che ha toccato per ultimo la pallina che invia dati sulla nuova posizione.

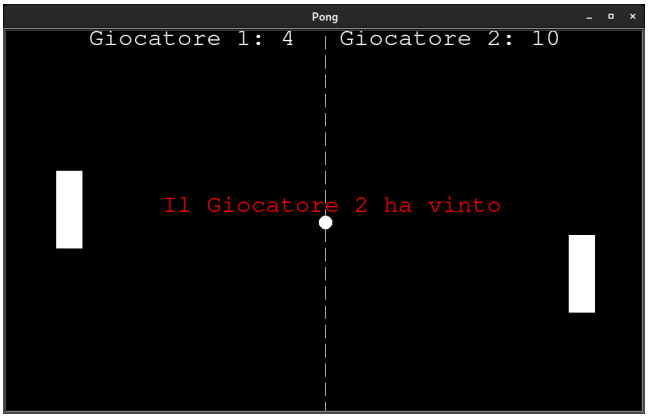
Gli spostamenti delle pedine devono essere notificati immediatamente alla controparte.

Il giocatore totalizza 1 punto se l'avversario non riesce ad intercettare la pallina.

2 Descrizione del progetto

Il progetto è basato sul gioco Pong multigiocatore. Il progetto è stato sviluppato in Python e le comunicazioni tra client e server avvengono su socket UDP. Per la visualizzazione del gioco è stata usata la libreria grafica Turtle inclusa in Python. Il progetto è composto da due file Python, un client ed un server. Per poter giocare occorre avviare un server e due client. Nel punto 6 vengono approfonditi i passaggi per l'esecuzione.
Il gioco inizia quando entrambi i giocatori sono collegati correttamente al server. Il giocatore che controlla la racchetta sinistra ha il controllo iniziale della pallina che viene lanciata verso la racchetta avversaria. Il controllo della pallina viene ceduto al giocatore avversario solo quando quest'ultimo tocca la pallina con la sua racchetta; se egli non intercetta in tempo la pallina, viene assegnato un punto al giocatore in controllo ad essa. Dopo aver segnato un punto il giocatore che ha il controllo della pallina lo mantiene portando la pallina al centro dello schermo e lanciandola verso l'avversario. La partita finisce quando un giocatore raggiunge 10 punti



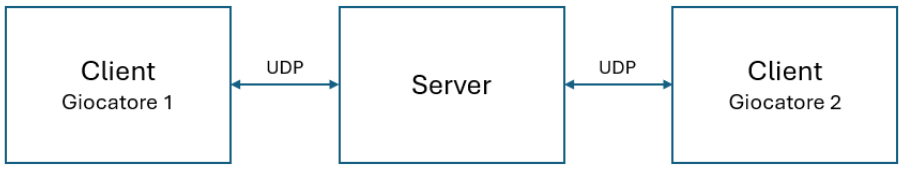


3 Descrizione e schema dell'architettura

Il progetto è basato su un'architettura client-server

Il server gestisce le connessioni con i client e si occupa di inviare costantemente lo stato del gioco tra i due client. Il server possiede due code ed un dizionario. Il dizionario contiene gli IP dei due giocatori collegati. Le code servono per memorizzare le posizioni degli utenti (la posizione della racchetta) e la posizione della pallina. Il server esegue due thread, uno per la ricezione dei messaggi dai client ed uno per l'invio dei messaggi ai client.

Il client si occupa della visualizzazione del campo di gioco (attraverso la libreria Turtle), del movimento della pallina e delle racchette dei giocatori. Il client possiede due code, una per le posizioni della pallina (usata quando la pallina viene controllata dall'avversario) ed una per la posizione della racchetta avversaria. Durante la partita, i giocatori inviano costantemente le posizioni delle racchette, il server inserisce le informazioni ricevute dai client nella coda delle posizioni (positions_queue) con l'IP del giocatore avversario, che dovrà ricevere la posizione aggiornata della racchetta.
Quando il giocatore che ha il controllo della pallina calcola l'urto della pallina con la racchetta avversaria, invia un messaggio di *"passo"* al server, il server informa il giocatore che ha toccato la pallina che può prendere il controllo della stessa. Il client esegue due thread, uno per l'invio della posizione della propria racchetta (e della pallina, se l'utente ha il controllo) ed uno per la ricezione delle posizioni di racchetta, pallina ed eventuali messaggi di *"passo"* o *"reset"*.



4 Dettagli implementativi dei client/server

4.1 Funzioni del server

handle_client(): la funzione per la gestione dei dati ricevuti dai client, inserisce l'IP dei giocatori nel dizionario e decodifica il messaggio ricevuto dal client e lo gestisce.
Il client può inviare al server quattro differenti tipi di messaggi:
-*"reset"*: il client che ha il controllo della pallina comunica all'avversario che ha segnato un punto. Il server attraverso il dizionario recupera l'IP del client avversario e manda il segnale di reset attraverso la funzione reset_players(ip_to_reset)
-*"passo"*: il client che ha il controllo della pallina comunica all'avversario che quest'ultimo ha toccato la pallina con la propria racchetta, cedendo così il controllo della pallina. Il server attraverso il dizionario recupera l'IP del client avversario e manda il segnale di passo attraverso la funzione next_players(next_ip)
-*"ball"*: il client invia un messaggio contenente la parola ball del tipo: *[ball x y]* con ball testo, x ed y valori interi rappresentanti le coordinate x ed y della pallina. Il server recupera le coordinate della pallina e le inserisce nella coda delle posizioni della pallina (ball_queue) con l'IP del giocatore che deve ricevere la posizione della pallina
-*"float"*: il client invia un messaggio contenente il valore float che rappresenta la posizione della propria racchetta, il server inserisce la coordinata della racchetta nella coda delle posizione delle racchette (positions_queue) con l'IP del giocatore che deve ricevere la posizione della racchetta avversaria

send_positions(): funzione per l'invio delle posizioni delle racchette e della pallina ai client, se la coda delle posizioni delle racchette non è vuota prende dalla coda la posizione della racchetta e l'IP del destinatario ed invia la posizione, se la coda delle posizioni della pallina non è vuota prende dalla coda la posizione della pallina e l'IP del destinatario ed invia la posizione

reset_players(ip_to_reset): funzione per l'invio del messaggio di reset all'utente che che non ha preso in tempo la pallina (utente identificato dal suo IP)

next_players(next_ip): funzione per l'invio del messaggio di passo all'utente che ha toccato la pallina e ne deve prendere il controllo

4.2 Funzioni del client

send_position(): la funzione invia al server la posizione della propria racchetta e, se l'utente ha anche il controllo della pallina, invia al server un messaggio nel formato [ball x y] con x ed y le componenti della posizione della pallina

receive_position(): la funzione riceve i dati dal server e li gestisce in modo opportuno, può ricevere:

-la posizione della racchetta avversaria: informazione necessaria per visualizzare i movimenti avversari, la posizione ricevuta viene inserita nella coda delle posizioni avversarie (avv_pad_queue)
-le coordinate della pallina: la funzione riceve la posizione calcolata dall'avversario e la inserisce nella coda delle posizioni della pallina (ball_queue)
-il messaggio di reset: se l'avversario ha segnato un punto viene ceduto il controllo (se per un errore di sincronizzazione è stata erroneamente toccata la palla ed assunto il controllo) e viene incrementato il punteggio avversario
-il messaggio di passo: quando un giocatore riceve correttamente la pallina viene informato dall'avversario di ciò, per cui può prendere il controllo della stessa. Di conseguenza viene svuotata la coda relativa alle posizioni precedentemente ricevute

update_opponent_paddle(): funzione usata per aggiornare la posizione della racchetta avversaria. Viene prelevato il valore relativo alla posizione dalla coda "avv_pad_queue" ed attraverso la funzione "avv_pad.sety(avv_pad_position)" viene visualizzata

update_ball(): funzione per la gestione del movimento della pallina, se l'utente non ha il controllo della pallina preleva la posizione dalla coda relativa alle posizioni della pallina (ball_queue) e la stampa con ball.goto. Se l'utente ha il controllo della pallina aggiorna la posizione attuale della pallina aggiungendo lo spostamento dx e dy alla posizione attuale della pallina e calcola le collisioni con il bordo superiore ed inferiore. Se la pallina collide con la racchetta avversaria viene inviato il messaggio di "passo" all'avversario e viene ceduto il controllo della pallina. Se la pallina supera la racchetta avversaria senza essere presa viene inviato il messaggio di reset e la pallina viene portata alle coordinate [0 0] al centro del campo di gioco

check_win(): funzione usata per controllare se un client ha raggiunto i 10 punti. Quando un giocatore vince la partita viene stampato il numero del giocatore vincitore e vengono terminati i thread del client

5 Parti rilevanti del codice sviluppato

5.1 Socket UDP

La comunicazione tra client e server avviene tramite le socket UDP.
Nel server viene creata una socket chiamata *socket_server* con:

```
socket_server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

Il server associa alla socket *socket_server* indirizzo e porta (in questo caso è presente l'indirizzo localhost, per poter eseguire il server ed i client sulla stessa macchina)

```
server_IP = "127.0.0.1"
server_port = 5005
socket_server.bind((server_IP, server_port))
```

Il server riceve sulla socket *socket_server* i dati dal client, e l'IP del mittente

```
data, client_address = socket_server.recvfrom(1024)
```

Successivamente il server decodifica il dato ricevuto sulla socket presente nella variabile *data*, in questo esempio viene decodificata la posizione della racchetta

```
position = data.decode()
```

Nel client viene creata una socket chiamata *socket_client* con:

```
socket_client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

Il client invia dati al server, in questo esempio viene inviata la posizione della pallina, la posizione prima di essere inviata viene codificata

```
server_IP = "127.0.0.1"
server_port = 5005
socket_client.sendto(ball_position_str.encode(), (server_IP, server_port))
```

5.2 Movimento della pallina

Quando un client ha il controllo della pallina calcola lo spostamento della pallina nella funzione *update_ball* con:

```
ball.goto(ball.xcor() + ball.dx, ball.ycor() + ball.dy)
```

-*ball* è l'oggetto che rappresenta la pallina nel gioco
-*goto()* è un metodo di Turtle che sposta la pallina alle coordinate specificate
-*ball.xcor()* e *ball.ycor()* restituiscono le coordinate correnti della pallina lungo gli assi x e y
-*ball.dx* e *ball.dy* sono rispettivamente la velocità lungo l'asse orizzontale e verticale della pallina, variando i valori di dx e dy è possibile variare la velocità della pallina. Ad ogni iterazione, i valori di dx e dy vengono sommati alle coordinate attuali della pallina, consentendo il movimento della stessa attraverso il campo di gioco.

6 Manuale utente con le istruzioni su compilazione ed esecuzione

Il progetto è stato sviluppato in Python, in particolare viene usata il modulo Python "Tkinter" che include la libreria grafica *Turtle*, su Debian il modulo Tkinter è stato installato con python3-tk.
Vengono usate le seguenti librerie incluse in python3: turtle, socket, time, threading, queue e sys.

Per l'esecuzione del gioco per entrambi i giocatori sulla stessa macchina basta avviare i file client.py e server.py in quanto entrambi i file hanno la variabile *server_IP* impostata su 127.0.0.1, localhost, quindi il server ed i due giocatori lavorano in locale. Per poter giocare nella stessa rete lan basta modificare nel client e nel server la variabile *server_IP* con l'IP locale della macchina dove viene eseguito il file server.py.
Il file client.py riceve un valore dall'avvio tramite riga di comando, il valore determina il lato del giocatore, passando il numero 0 il giocatore controlla la racchetta sinistra, inserendo il numero 1 il giocatore controlla la racchetta destra. Il controllo iniziale della pallina lo ha sempre il giocatore 0.

Guida per poter giocare:
1) eseguire il file server.py per avviare il server
2) eseguire il file main.ph 1 per avviare il giocatore che controlla la racchetta destra (rimane in attesa del giocatore 0)
3) eseguire il file main.ph 0 per avviare il giocatore che controlla la racchetta sinistra e la pallina

```
python3 server.py

python3 main.py 1

python3 main.py 0
```