# Grafica

*Anno Accademico 2018-2019*

# Esercitazione 2

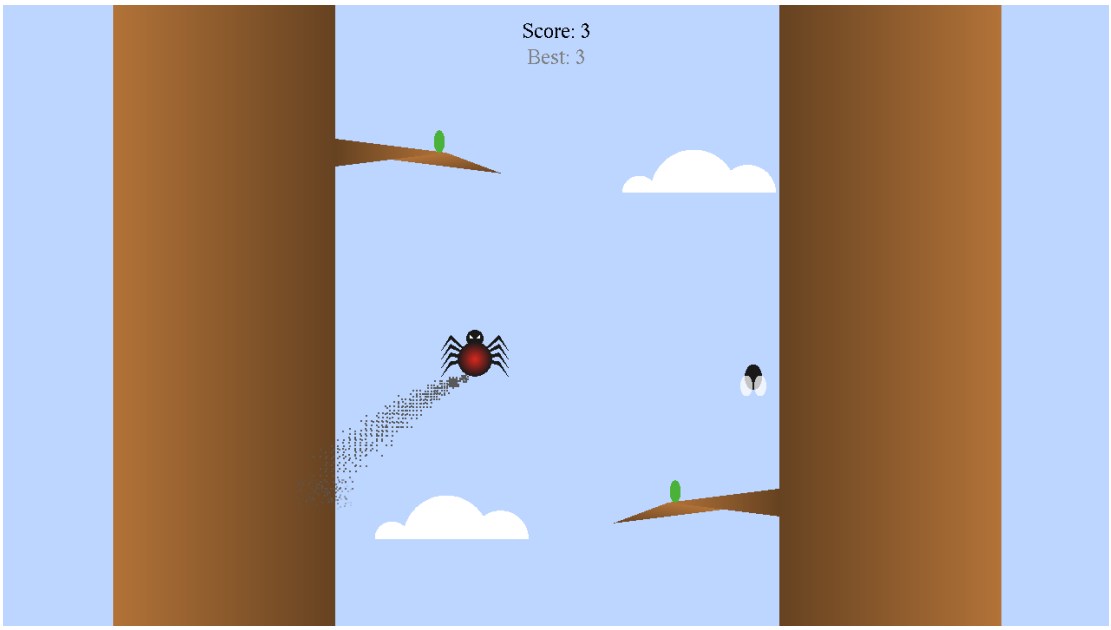
Matteo Berti, Matricola 889889

25 giugno 2019

**Comandi da tastiera**

| **Tasto** | **Azione** |
|---|---|
| Esc | Terminazione dell'esecuzione |
| Space | Salto / Restart |

**1. Impegno artistico**
La scena contiene vari elementi che si muovono in modo coordinato generando un effetto **parallasse** che crea un effetto "scalata" verso l'alto. Le diverse velocità di discesa delle nuvole, rispetto ai rami danno l'impressione della lontananza. Il ragno stesso oscilla nella scalata per creare un effetto più realistico. Oltre a questo sono presenti sfumature e trasparenze nei colori per un maggior impatto visivo.




**2. Animazioni basate su simulazioni fisiche**
Il salto del ragno da un tronco all'altro avviene con le stesse regole presenti nel mondo fisico, quindi tenendo conto della gravità, della velocità e della differenza in questultima nei vari periodi

del salto (subito vi è uno scatto più rapito poi si rallenta).

**3. Elementi di gameplay**
Il giocatore è sconfitto quando entra in collisione con un ramo, mentre aumenta il proprio punteggio se mangia una mosca. Non solo il punteggio della partita attuale è costantemente visualizzato ma lo è anche il miglior punteggio realizzato nell'attuale sessione di gioco.
L'apparizione di rami e mosche non è prevedibile in quanto generati in modo del tutto casuale, ma facendo si che ogni mosca sia sufficientemente distanziata dai rami per permettere al giocatore di prenderla.
Infine man mano che il gioco procede la velocità di "scalata" aumenta linearmente, aumentando la difficoltà e i punti ottenuti per ogni mosca.

**4. Presenza di particellari**
I particellari simulano una tela dietro al ragno. Quando è fermo vengono generate particelle casuali che formano un alone sotto al ragno, mentre durante il salto creano una scia che segue la traiettoria.

**5. Esecuzione indipendente delle animazioni**
Tramite la funzione `glutTimerFunc(5, updateSpider, 0)` il moto del ragno è gestito in modo autonomo ed avendo una frequenza più veloce rispetto alla scena, il movimento è molto più fluido.

**Struttura generale**
In generale le funzioni `build*()` generano e posizionano le figure in scena, mentre le funzioni `draw*()` utilizzano i dati nei *VBO* per disegnare effettivamente sullo schermo le figure. La gestione della collisione (sia ragno-ramo che ragno-mosca) avviene tramite **Axis-Aligned Bounding Box** calcolate sia in *OCS* che in *WCS*. La generazione casuale dei rami tiene conto che vi sia una distanza minima sia tra due rami sullo stesso tronco che tra due rami su tronchi diversi, questo per permettere sempre la possibilità di saltare. Per quanto riguarda la generazione casuale di mosche si tene conto che vi sia una distanza minima dai tronchi per evitare sovrapposizioni.