INTRODUZIONE

Trainer estimation è un’applicazione pensata per aiutare chi svolge attività fisica, in casa o in palestra a monitorare la propria sessione di allenamento. E’ utile per tenere conto delle ripetizioni svolte di un determinato esercizio e per verificare che lo si abbia eseguito correttamente. Infatti vengono contate solamente le ripetizioni eseguite correttamente.

UTILIZZO

Utilizzare il programma non è per nulla difficile, in quanto l’utente deve solo scegliere quale esercizio eseguire (fra quelli supportati ovviamente), posizionarsi nella posizione stabilita, e al resto ci pensa TrainerEstimation. Infatti, questo assistente di allenamento conta le ripetizioni e aiuta l’utente nel caso esegua una ripetizione sbagliata.

PROGETTAZIONE

Nella fase di progettazione si sono definito inanzitutto quali siano i casi d’uso del programma, eventuali risorse, come ad esempio un modello di esempio per ogni esercizio, e il modo in cui stabilire se l’esercizio viene eseguito correttamente o meno. Per fare ciò, si è deciso inanzitutto di utilizzare come linguaggio di sviluppo, python. In quanto esistono librerie che effettivamente permettono di realizzare il progetto. Abbiamo infatti utilizzato diverse librerie, ognuna con uno scopo ben preciso, e dedita ad una fase specifica dell’esecuzione. Le due librerie principali utilizzate sono opencv e mediapipe.

Opencv viene utilizzata per acquisire le immagini da webcam ed eventualmente modificarle

mediapipe invece è una libreria realizzata da google in python, utile per il rilevamento della posizione di una persona, è in grado di rilevare fino a 33 punti fisici diversi.

Per ricavare la posizione di partenza, è stato creato un video apposito, dove una persona è posizionata in posizione di partenza per diversi secondi. Non abbiamo potuto utilizzare un’immagine perché in queste, la libreria non funziona. Dopo aver scansionato quindi il video, si ricavano le coordinate di ogni landmark(punto corporeo) e confrontate con quelle dell’utente, rilevate in tempo reale. Se quindi l’utente viene rilevato in posizione di partenza, allora si passa alla fase successiva. Il programma inizia, da questo momento, a registrare le azioni dell’utente, ricavare le coordinate dei landmarks frame per frame e creare un dataset, che viene costantemente aggiornato. Una volta che l’utente torna in posizione di partenza, viene effettuato un confronto tra il dataframe creato in ‘live’, e quello preesistente, riferito ad una esecuzione eseguita correttamente. Il confronto viene effettuato questa volta, tramite dtw. Tecnica utile per confrontare delle sequenze temporali. Questo tipo di funzione, restituisce una certa distanza, la quale, se minore della soglia impostata, reputa la ripetizione corretta, e ne aggiorna quindi il numero.

Il codice è stato strutturato in modo da essere il più generale possibile, è infatti molto semplice aggiungere un nuovo esercizio da supportare. Basta creare una nuova cartella contenente i file che servono all’esecuzione. In particolare, servirà inserire un file testuale contenente riga per riga, le coordinate x e y di ogni landmark (una coppia di righe quindi, rappresenterà la posizione di un landmark), un dataframe che rappresenterà l’esempio di esecuzione corretta dell’esercizio e un video di esempio. Aggiunti questi tre file alla cartella delle risorse, il gioco è fatto. Basterà semplicemente definire una soglia di accettazione per la posizione iniziale e per l’esecuzione.

RIFLESSIONI

L’applicazione si dimostra funzionare egregiamente su pc, unica piattaforma per ora supportata dall’app, anche se, come verificato, la maggior parte degli utenti avrebbe preferito una sua implementazione su smartphone, per ovvie ragioni di portabilità. E’ stato inoltre effettuato un test per valutarne le prestazioni in termini di fps, e con una media di 18 frame al secondo, l’app risulta utilizzabile e non particolarmente fastidiosa. Da sottolineare inoltre, che non avendo a disposizione una webcam interna, sono state utilizzate delle webcam esterne, connesse tramite wifi, che forse hanno leggermente appesantito il carico su cpu, già abbastanza grande di suo. Questo è dovuto a diversi fattori, al fatto che la componente principale del programma viene eseguita su thread, in quanto l’altra parte si occupa di disegnare la posizione inziale. Inoltre, la libreria utilizzata, mediapipe, lavora su cpu. E quindi rende molto pesante il carico su questo componente hardware. Senza contare il fatto che il confronto dell’esecuzione, utilizzato il dtw, richiede molte risorse computazionali. In parole povere quindi, tutto il programma pesa sulla cpu