

**Hands and finger detection to control PC programs****Board:** Zynq board or Raspberry**Target:** Image processing techniques to detect hand and finger movement. Associating gestures to different control operations such as change page, scrolling, drag & drop, minimize, etc.**Requisiti****di****progetto:**

Realizzare un sistema basato su scheda Zynq o Raspberry che possa svolgere semplici operazioni per il controllo di un personal computer (PC) utilizzando i movimenti delle mani rilevati in un video. La parte software deve essere realizzata utilizzando il linguaggio c++.

Specifiche di progetto (definite in parte oralmente):

SdP01: Il progetto verrà sviluppato su una scheda Raspberry pi 4 B.

SdP02: La parte software verrà realizzata utilizzando il linguaggio C++, appoggiandosi alla libreria open-source OpenCV.

SdP03: Sulla scheda verrà installato il sistema operativo Raspbian OS. In questo modo il PC controllato sarà la Raspberry pi stessa.

SdP04: Il video con i movimenti di una mano sarà lo stream di una webcam collegata alla scheda Raspberry pi. In mancanza di questa verrà usato un file video.

SdP05: Il sistema non dovrà necessitare di calibrazione preliminare da parte dell'utente.

SdP06: il sistema controllerà il volume di sistema, la rotella del mouse (scroll) e la gestione delle finestre aperte dall'utente.

Realizzazione:

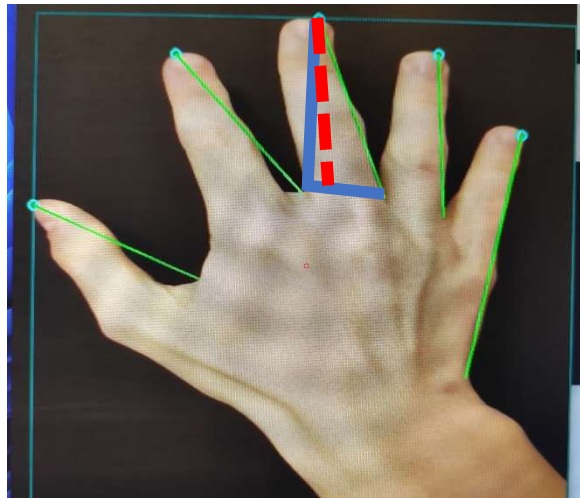
Per realizzare il progetto è stata realizzata prima la parte software: una volta ottenuto un programma in C++ funzionante, è stato caricato lo stesso sulla scheda Raspberry.


Il programma realizzato utilizza la libreria open-source OpenCV per acquisire ed elaborare i frame video. Aperto un file video o lo stream di una webcam, vengono presi i singoli frame e l'immagine viene elaborata come segue: viene ridimensionata (per ridurre la grandezza delle matrici e quindi la complessità dei calcoli), viene convertita in scala di grigi, viene applicato un blur gaussiano (al fine di ridurre la qualità dell'immagine e il rumore) e viene applicata una soglia binaria alla scala di grigi (in questo modo tutto ciò che è più chiaro di un certo colore viene rappresentato da pixel bianchi, tutto il resto da pixel neri). Di default si suppone che lo sfondo sarà più scuro delle mani dell'utente (quindi la mano verrà rappresentata in bianco rispetto allo sfondo), ma viene in automatico effettuato un controllo dopo alcuni frame e, qualora fosse necessario, viene cambiato il tipo di soglia per ottenere sempre la mano bianca su sfondo nero. Viene anche creata una trackbar che consente di cambiare in tempo reale il valore della soglia: a seconda della luce nella stanza, sarà necessario un valore piuttosto che un altro per evidenziare al meglio la mano e le dita.

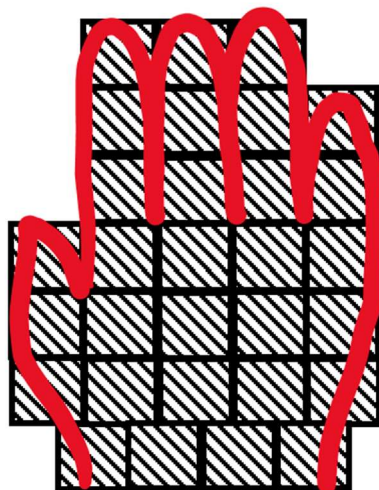
Vengono quindi riconosciuti i contorni di tutti gli oggetti presenti nell'immagine binaria ottenuta, tramite la funzione findContours presente in OpenCV. Supponendo che la mano sia l'oggetto più grande presente nel video, viene calcolata l'area di tutti gli oggetti presenti nel frame e i calcoli successivi vengono fatti solo sull'oggetto con l'area più grande. La funzione findContours funziona bene quando lo sfondo è quanto più possibile uniforme, quindi questo diventerà una limitazione del progetto finale. L'oggetto identificato come "mano" viene quindi riquadrato da un rettangolo.

Per riconoscere le dita della mano vengono cercati punti di convessità all'interno dei contorni dell'oggetto identificato come "mano" (come illustrato, l'oggetto con l'area più grande). Per fare ciò vengono presi i difetti di convessità e, se si supera una certa soglia, viene interpretata la convessità come un dito. Nello specifico vengono presi tre punti susseguenti (p1,p2,p3 nel codice): sono i punti restituiti dalla funzione

convexityDefects, e identificano il punto di inizio, centrale e di fine del difetto di convessità. Se la distanza tra il punto centrale e il punto di fine/inizio è maggiore di un certo valore (lunghezza dito) e, contemporaneamente, l'altezza del triangolo costruito sui tre punti è maggiore di una certa soglia (profondità) allora viene disegnato un segmento verde a identificare la punta del dito.

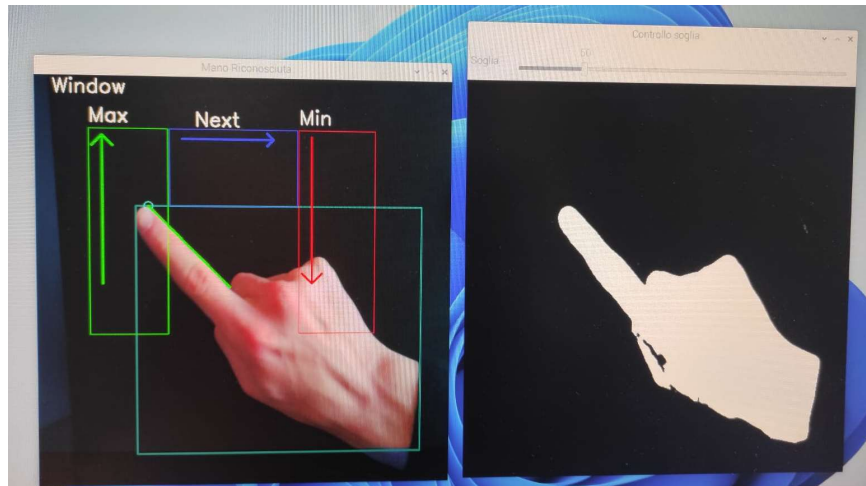


Un dito casuale viene identificato come “indice” per poterlo usare nei controlli che necessitano di uno o due dita: la punta dello stesso viene cerchiata, per permettere un miglior controllo in fase di utilizzo. Queste lunghezze e distanze sono ovviamente dipendenti da quanto è grande la mano nell'immagine: avvicinare la mano o meno alla videocamera avrebbe l'effetto di ingrandire le dita mentre allontanarla avrebbe l'effetto di rimpicciolirle. Per evitare quindi che fosse necessaria una nuova calibrazione, le variabili di lunghezza e profondità sono state rese dipendenti dall'area della mano: considerando il seguente modello (non standardizzato e puramente inventato) della mano, è stata trovata la lunghezza di un dito come circa tre volte il lato di un quadrato . L'area totale della mano equivale a poco più dell'area di 30 quadrati (varia a seconda di includere o meno il polso).

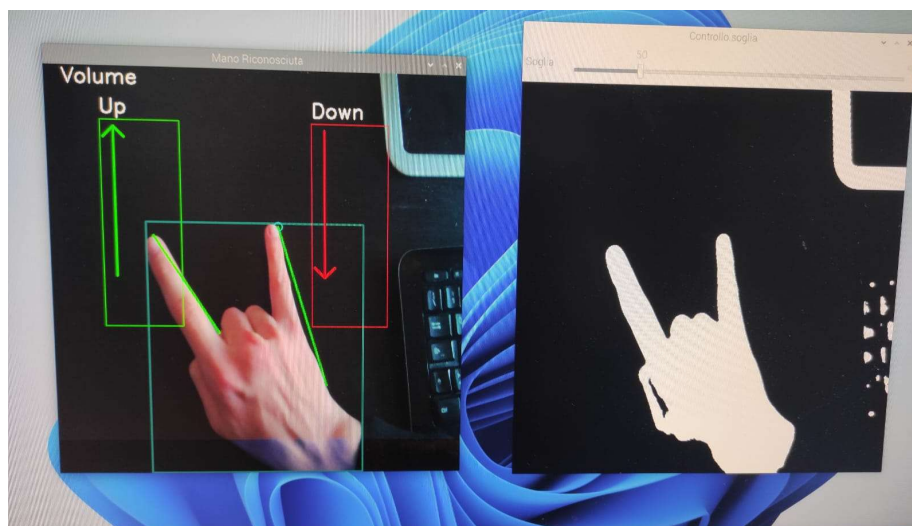
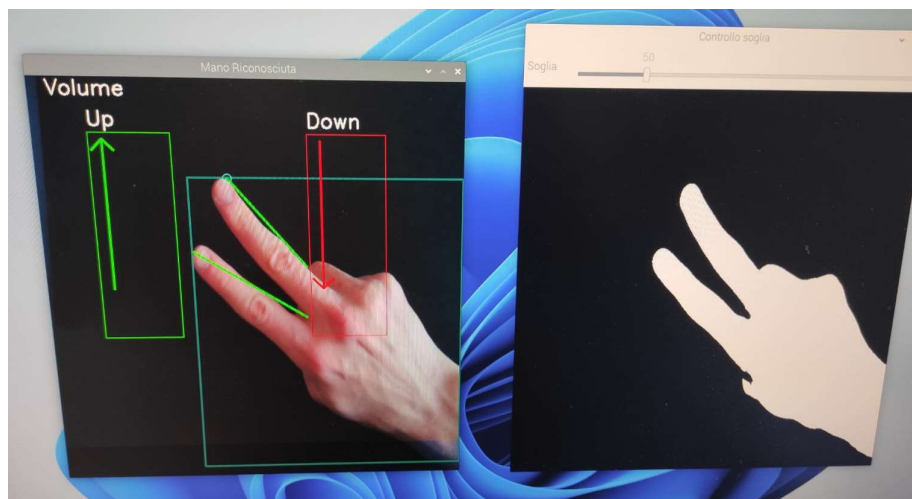


Incrociando il numero delle dita alzate e l'area della mano, è possibile determinare (in maniera abbastanza robusta) se la mano è aperta o chiusa. Per determinare se la mano è aperta o chiusa viene preso un frame ogni dieci e vengono confrontate le dita riconosciute di volta in volta e la variazione di area: è necessario decimare i frame per fare in modo che l'apertura/chiusura non siano troppo veloci. Oltre al controllo sull'apertura o chiusura della mano, sono stati implementati dei controlli basati sul movimento del dito identificato come “indice”: incrociando questo movimento con il numero di dita alzato è possibile realizzare vari controlli. Per facilitare l'utilizzo l'interfaccia utente viene realizzata scrivendo in alto

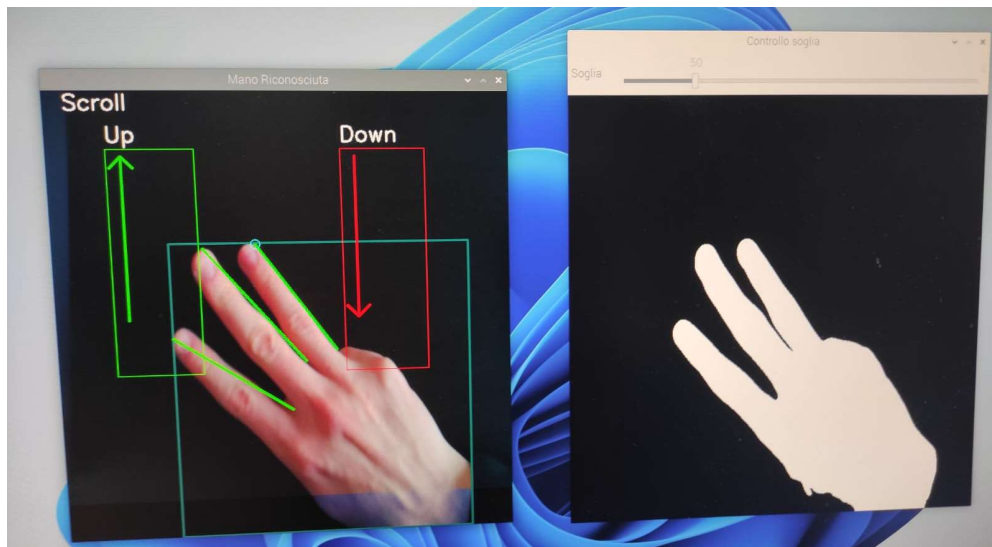
a sinistra la funzione che si sta controllando, e vengono disegnati dei rettangoli colorati con delle frecce per facilitare l'utente ed evitare che dei movimenti "spuri" vengano considerati degli input. Quando una funzione viene attivata (ad esempio lo scroll up) la scritta corrispondente viene evidenziata con un colore differente. Il movimento viene rilevato confrontando la coordinata della punta del dito evidenziato a cinque frame di distanza, fissando al 10% della grandezza del frame la soglia per l'attivazione del movimento. 1 dito – si selezionano e si controllano le finestre attive: scorrendo verso destra il dito si naviga tra le finestre, posizionando il dito nel rettangolo rosso e scorrendo verso il basso si riduce la grandezza della finestra selezionata, scorrendo verso l'alto nel riquadro verde si aumenta la grandezza. Più avanti verrà descritta la modalità di chiusura delle finestre.



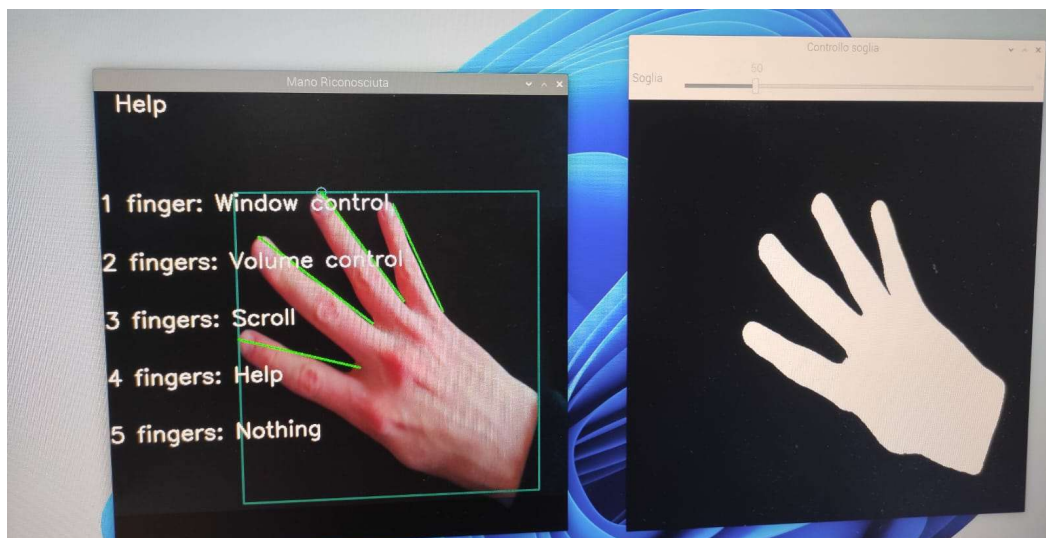
2 dita – si controlla il volume di sistema: anche qui bisogna far scorrere il dito evidenziato nei rettangoli seguendo le frecce. Si sottolinea che non è importante utilizzare due dita particolari, basta che siano due.



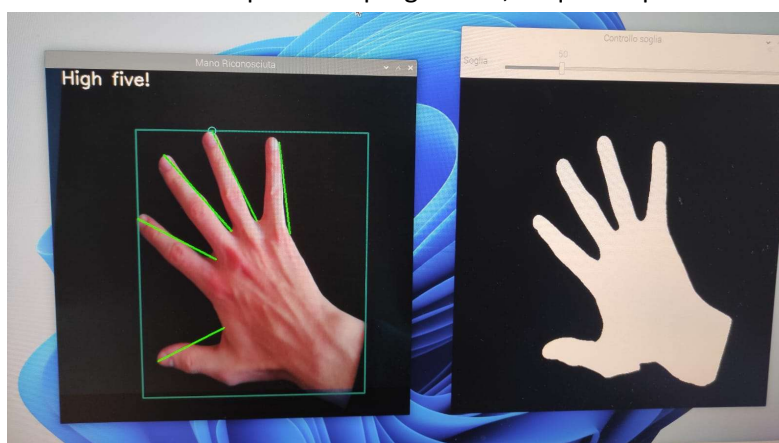
3 dita – si controlla la rotella del mouse (scroll) verso l'alto o verso il basso nella pagina selezionata, muovendo il dito evidenziato verso l'alto o verso il basso nei rettangoli.



4 dita – alzando quattro dita compare un “Help” ovvero un veloce riassunto delle varie funzioni, sottoforma di scritte bianche sulla finestra di controllo. Un manuale esteso viene aperto sul terminale premendo il tasto “h” sulla tastiera



5 dita – Alzare cinque dita di base non è associato a nessun controllo. Se però la mano viene chiusa repentinamente, si termina il processo della finestra selezionata. Questo controllo va utilizzato dopo aver selezionato una finestra differente da quella del programma, in quanto può terminare anche il programma stesso.



Modalità d'uso

Una volta accesa la scheda Raspberry, è necessario aprire un terminale e posizionarsi nella cartella "DSP": è necessario scrivere sul terminale `cd Desktop/All/DSP`. Per lanciare il programma bisogna scrivere sul terminale il comando `./jarvis.exe` seguito dalla sorgente del video ("webcam" o "nome file"). Si apriranno due finestre: una con l'immagine catturata dalla webcam (con eventuali dita alzate riconosciute evidenziate), e una seconda finestra per il controllo della soglia. Se la mano non è riconosciuta o se le dita alzate non vengono contate correttamente, è possibile usare la trackbar presente sulla seconda finestra per modificare la soglia. Una volta in funzione il programma è possibile premere barra spaziatrice per metterlo in pausa (o riprendere l'esecuzione) o il pulsante ESC per chiudere il programma. Premendo "h" viene aperto sul terminale il manuale (manual.txt).

Se fosse necessario modificare il codice del programma, è possibile farlo modificando il file "jarvis.cpp" e lanciando da terminale il comando "make". Se si volesse compilare un altro programma utilizzando sempre OpenCV presente nella cartella e CMake, sarebbe necessario modificare il file CMakeList.txt presente indicando il nome del nuovo programma e il file del codice sorgente. Nella cartella "DSP" sono presenti anche le versioni precedenti del programma (i file chiamati "codice" numerati in ordine crescente risalgono a quando il progetto era in uno stato embrionale, i file "jarvisB", anch'essi numerati in ordine crescente, sono delle versioni precedenti del programma), dei video di prova (in formato mp4, ad esempio "test.mp4") e della musica per poter testare il volume (per ascoltarle usare il comando `./paplay "nome canzone"` da terminale) in formato wav.

