UNIVERSITÀ DEGLI STUDI DI MESSINA

DIPARTIMENTO DI SCIENZE MATEMATICHE E INFORMATICHE, SCIENZE FISICHE E SCIENZE DELLA TERRA

Corso di Laurea triennale in Informatica L-31

Sistema di traduzione real-time dell'alfabeto LIS: un approccio geometrico per l'abbattimento delle barriere comunicative

Relatore:

Prof. Andrea Nucita

Candidato:

Luca Ciraolo

Anno Accademico
2023/2024

Indice

Elenco delle figure

Listings

Introduzione

Contesto e Motivazione

La LIS (Lingua dei Segni Italiana) risulta essere molto più che un semplice strumento di comunicazione: è l'espressione che caratterizza ed identifica una comunità che, in Italia, conta oltre 100.000 persone sorde e circa 1 milione con vari gradi di deficit uditivo [1]. Questa lingua visivo-gestuale, fu riconosciuta ufficialmente dal Parlamento italiano nel maggio 2021 [2] e costituisce il principale mezzo di comunicazione per la comunità sorda e muta, permettendo la comunicazione di concetti complessi attraverso una combinazione di movimenti delle mani, espressioni facciali e posture del corpo.

Tuttavia, nonostante la sua notevole rilevanza, la lingua dei segni italiana (LIS) rimane quasi del tutto sconosciuta al di fuori della sua comunità. Questa conoscenza limitata crea delle barriere comunicative tra persone udenti e non, impattando aspetti di una certa importanza nella vita quotidiana: dall'accesso ai servizi pubblici alla pubblica istruzione, dalle relazioni sociali alle opportunità nel mondo lavorativo. In particolare, le statistiche mostrano che:

- In Italia si contano circa 100.000 persone capaci di comprendere la LIS. [3]
- Il 67% delle persone sorde riporta difficoltà nell'accesso ai servizi pubblici. [4]
- Le persone sorde e ipoudenti sperimentano frequenti barriere nella comunicazione medica, spesso dovute alla mancanza di interpreti LIS e strumenti di assistenza adeguati. [5]

- Il tasso di occupazione per le persone con disabilità uditiva è sensibilmente inferiore alla media nazionale. Solo il 26% delle persone con disabilità uditiva dichiara di essere occupato, a fronte di una media nazionale di occupazione che si aggira intorno al 7-8% nel 2023, evidenziando un tasso di disoccupazione nella comunità sorda ben più elevato rispetto alla popolazione generale. [6]

In questo contesto, l'evoluzione tecnologica offre nuove opportunità per abbattere queste barriere. L'avvento di tecnologie come la computer vision e il machine learning ha aperto la strada allo sviluppo di sistemi in grado di riconoscere e interpretare i segni in tempo reale. Questi avanzamenti tecnologici, combinati con la crescente potenza di elaborazione, rendono possibile la creazione di strumenti che possano fungere da "ponte comunicativo" tra le comunità udenti e non.

Il lavoro da me sviluppato si inserisce proprio in questo contesto, proponendo un sistema innovativo che combina il riconoscimento della LIS con tecnologie di autocorrezione e sintesi vocale. L'obiettivo è duplice:

1. Facilitare la comunicazione bi-direzionale tra persone sorde e udenti, fornendo uno strumento di traduzione immediata e il più accurato possibile.

2. Contribuire alla diffusione e alla comprensione della LIS nella società, promuovendo una maggiore inclusione della comunità non udente.

La scelta di sviluppare questo sistema nasce dalla consapevolezza che la tecnologia può e deve essere uno strumento di inclusione sociale. In un'era in cui la comunicazione digitale è sempre più pervasiva, è fondamentale che questi avanzamenti tecnologici siano messi al servizio di tutte le comunità, con particolare attenzione a quelle che affrontano maggiori sfide.

Obiettivi del Progetto

Il progetto sviluppato nasce con l'ambizioso obiettivo di sviluppare un sistema per il riconoscimento e la traduzione della Lingua dei Segni Italiana (LIS). Contribuendo, così, all'abbattimento delle barriere comunicative che ancora oggi rappresentano un ostacolo significativo per la comunità non udente. [2].

Il cuore del progetto risiede nello sviluppo di un sistema di riconoscimento real-time basato sulla libreria open-source offerta da Google: MediaPipe [7]. Questa tecnologia è stata principalmente scelta per la sua capacità di tracciare con precisione i movimenti delle mani, elemento fondamentale per poter riconoscere in maniera accurata i segni dell'alfabeto LIS. L'obiettivo si estende alla creazione di un sistema che possa elaborare i gesti in tempo reale, garantendo una comunicazione naturale.

Un aspetto fondamentale consiste nell' implementazione di un sistema di autocorrezione specializzato. Considerando la natura visiva e gestuale della lingua dei segni, la traduzione dalla LIS al testo scritto può infatti presentare ambiguità, oltre che imprecisioni. Proprio per questo motivo, il sistema integra sia un dizionario specializzato per la lingua italiana, sia algoritmi di correzione ortografica, con l'obiettivo di migliorare significativamente l'accuratezza della traduzione stessa.

La componente di feedback rappresenta un altro elemento cruciale, in quanto il sistema deve fornire un riscontro immediato e chiaro all'utente. Ciò è ottenuto attraverso l'implementazione di un'interfaccia visiva che mostra il riconoscimento in real-time, e mediante un sistema di sintesi vocale che legge le traduzioni elaborate. Questa duplice modalità di feedback è pensata per rendere il sistema accessibile e utilizzabile in diversi contesti comunicativi.

Dal punto di vista tecnico, in termini di accuratezza e tempo di risposta, il progetto si propone di raggiungere elevati standard di performance. La robustezza e l'affidabilità del sistema risultano essere obiettivi imprescindibili del lavoro svolto, in quanto, il sistema deve essere in grado di funzionare andando a considerare le diverse condizioni di illuminazione e le differenze individuali nella gestualità.

Contributi Principali

Il presente progetto di tesi ha portato allo sviluppo di soluzioni innovative nel campo del riconoscimento della LIS, introducendo diversi contributi significativi sia dal punto di vista tecnico che applicativo.

Il contributo principale risiede nella realizzazione di un sistema di riconoscimento gestuale, che utilizza un approccio geometrico originale per l'analisi dei landmark delle mani. Basato sulla tecnologia MediaPipe [7], il sistema implementa una serie di funzioni matematiche ottimizzate per il calcolo delle distanze e degli angoli tra i punti di riferimento delle mani. Questo approccio ha permesso di sviluppare un sistema di tracking efficace, capace di riconoscere i segni dell'alfabeto LIS con tempi di risposta adeguati alle esigenze di una comunicazione in tempo reale.

Un altro contributo significativo è rappresentato dal sistema di autocorrezione, sviluppato specificamente per questo contesto. Il sistema integra un dizionario personalizzato per la lingua italiana, che include non solo il vocabolario comune, ma anche nomi propri e toponimi frequentemente utilizzati. L'algoritmo di correzione è stato progettato considerando le particolarità della traduzione dalla LIS, implementando un sistema di gestione delle ambiguità che propone correzioni multiple quando necessario.

L'interfaccia utente rappresenta un ulteriore elemento innovativo. Il sistema fornisce un feedback multimodale che combina elementi visivi e sonori. L'interfaccia grafica mostra in tempo reale il riconoscimento dei landmark e la conseguente traduzione dei segni, mentre il sistema di sintesi vocale fornisce un feedback audio immediato. Questa combinazione permette una comunicazione più naturale e accessibile.

I test condotti hanno permesso di verificarne l'efficacia pratica. Il sistema richiede 1.5 secondi per il rilevamento iniziale delle mani e lo stesso tempo per il riconoscimento di ogni singola lettera, con un tempo di 1.2 secondi per il rilevamento degli spazi. Questi tempi sono stati calibrati accuratamente per garantire un equilibrio ottimale tra velocità di risposta e accuratezza del riconoscimento.

L'integrazione di queste diverse componenti rappresenta un avanzamento significativo nello stato dell'arte del riconoscimento della LIS. Il progetto non solo dimostra la fattibilità di un approccio integrato al riconoscimento della lingua dei segni, ma fornisce anche una base solida per futuri sviluppi e miglioramenti nel campo dell'accessibilità e della comunicazione inclusiva.

Capitolo 1

Stato dell'Arte

1.1 Riconoscimento dei Gesti

L'evoluzione del riconoscimento automatico dei gesti rappresenta un affascinante percorso nell'ambito dell'interazione uomo-macchina, caratterizzato da continue innovazioni tecnologiche e metodologiche. Questo campo di ricerca, nato dalla necessità di creare interfacce più naturali e intuitive tra esseri umani e computer, ha attraversato diverse fasi evolutive, ognuna delle quali ha contribuito significativamente allo stato attuale della tecnologia.

Nei primi anni '80, i pionieri del settore si confrontarono con la sfida fondamentale di catturare e interpretare i movimenti umani. I primi approcci si basavano su sistemi hardware dedicati, in particolare guanti sensorizzati dotati di accelerometri e giroscopi. Il Data Glove, introdotto nel 1987, rappresentò una pietra miliare in questo campo; questo dispositivo, sebbene tecnologicamente avanzato per l'epoca, evidenziava già le limitazioni intrinseche dell'approccio basato su hardware specializzato: costi elevati, la necessità di calibrazione frequente e, soprattutto, una significativa innaturalezza nell'interazione, poiché l'utente era costretto a indossare dispositivi ingombranti e limitanti nei movimenti.

Gli anni '90 videro l'emergere di un nuovo approccio basato su marker visivi, questi sistemi utilizzavano marcatori riflettenti applicati alle mani dell'utente e telecamere speciali per tracciarne i movimenti. Questa tecnologia, ancora oggi utilizzata in ambiti specifici come la motion capture cinematografica, offriva una precisione notevole ma manteneva molte delle limitazioni pratiche dei guanti sensorizzati, in particolare la necessità di una preparazione accurata dell'ambiente e dell'utente.

L'avvento delle tecniche di computer vision basate su telecamere RGB standard ha aperto la strada a soluzioni più accessibili e naturali, eliminando la necessità di hardware specializzato. I primi sistemi di questo tipo, sviluppati nei primi anni 2000, si basavano principalmente su tecniche di segmentazione del colore e rilevamento dei contorni; questi approcci, sebbene innovativi, mostravano significative limitazioni in termini di robustezza, di fatto, le variazioni di illuminazione, gli sfondi complessi e le occlusioni rappresentavano sfide spesso insormontabili.

Il punto di svolta significativo si è verificato con l'introduzione di sensori di profondità a basso costo, in particolare con il lancio di Microsoft Kinect nel 2010. Questa tecnologia ha permesso di ottenere informazioni tridimensionali della scena in tempo reale, facilitando notevolmente il compito di segmentazione e tracking delle mani. L'approccio basato su sensori di profondità ha dominato il settore per diversi anni, portando allo sviluppo di numerose applicazioni pratiche nel campo della realtà virtuale e dell'interazione gestuale. Un esempio particolarmente innovativo è stato il sistema sviluppato da Zafrulla et al. [8], che ha utilizzato Kinect per creare uno dei primi sistemi di riconoscimento dell'American Sign Language (ASL) basato su sensori di profondità. Il loro lavoro, "American Sign Language Recognition with the Kinect", ha dimostrato come la tecnologia di sensing 3D potesse essere applicata efficacemente nel contesto della lingua dei segni, raggiungendo un'accuratezza dell'86% nel riconoscimento di frasi ASL.

Nonostante le numerose evoluzioni, i punti di svolta e gli avvenimenti storici, la vera rivoluzione nel campo del riconoscimento gestuale è stata segnata dall'avvento di MediaPipe [7], una tecnologia che ha ridefinito completamente lo stato dell'arte nel tracking delle mani. Il suo impatto è stato particolarmente significativo poiché ha saputo combinare l'accessibilità delle comuni telecamere RGB con un livello di accuratezza che in precedenza era ottenibile solo attraverso hardware specializzato.

L'architettura di MediaPipe si distingue per la sua sofisticata pipeline di elaborazione multi-stage, dove ogni fase è stata accuratamente ottimizzata per bilanciare precisione ed efficienza computazionale. Il processo di riconoscimento si articola in tre fasi principali, ognuna delle quali contribuisce all'eccezionale robustezza del sistema. Inizialmente, un modello di detection rapida, implementato attraverso una rete neurale convoluzionale, si occupa di individuare le mani all'interno dell'immagine. Una volta localizzate quest'ultime, entra in gioco un secondo modello, più sofisticato e dettagliato, che si concentra sulla stima precisa di 21 landmark tridimensionali per ciascuna mano. Questi punti di riferimento vengono posizionati strategicamente sulle articolazioni e sulle estremità delle dita, fornendo una rappresentazione dettagliata delle mani. La precisione di questa fase è cruciale, in quanto i landmark costituiscono la base per qualsiasi successiva analisi gestuale. La terza e ultima fase del processo introduce un elemento di particolare innovazione: l'implementazione di tecniche di regolarizzazione temporale nel processo di tracking. Sfruttando la coerenza naturale tra frame consecutivi, MediaPipe riesce a stabilizzare le stime dei landmark e a ridurre significativamente le fluttuazioni nelle previsioni. Tale approccio non solo migliora la qualità complessiva del tracking, ma contribuisce anche a mantenere una coerenza temporale che risulta fondamentale per applicazioni in tempo reale, come il riconoscimento della lingua dei segni.

Questa architettura a cascata rappresenta un compromesso intelligente tra velocità e accuratezza: il primo stadio, più leggero, filtra rapidamente le aree di interesse, mentre il secondo stadio concentra la potenza computazionale dove effettivamente necessario, e il terzo assicura la stabilità temporale delle previsioni. Il sistema implementa inoltre tecniche avanzate di tracking che sfruttano la coerenza tra frame consecutivi, permettendo di mantenere una stima stabile anche in condizioni di movimento rapido o occlusione parziale.

Un aspetto particolarmente rilevante di questa architettura è la sua capacità di operare in tempo reale su hardware standard, un risultato ottenuto attraverso una serie di ottimizzazioni algoritmiche e architetturali mirate. L'utilizzo di modelli quantizzati, l'implementazione di pipeline di elaborazione parallele e l'adozione di tecniche di predizione anticipata contribuiscono a mantenere latenze ridotte pur garantendo un'elevata accuratezza nel riconoscimento. Questa combinazione di caratteristiche rende MediaPipe particolarmente adatto per applicazioni pratiche nel campo del riconoscimento della lingua dei segni, dove la reattività del sistema è tanto importante quanto la precisione del riconoscimento.

Nel contesto specifico del riconoscimento della lingua dei segni, l'approccio geometrico all'analisi dei landmark si è rivelato particolarmente efficace. Questa metodologia offre diversi vantaggi significativi: in primo luogo, l'analisi geometrica è intrinsecamente invariante rispetto a scale e rotazioni, caratteristica fondamentale per gestire la variabilità naturale dei gesti umani. In secondo luogo, questo approccio produce risultati facilmente interpretabili, aspetto cruciale per lo sviluppo e il debugging di sistemi di riconoscimento gestuale.

L'analisi geometrica si basa su un insieme di misure e relazioni spaziali accuratamente selezionate. Le distanze euclidee tra coppie di landmark forniscono informazioni sulla configurazione complessiva della mano, mentre gli angoli tra vettori definiti da triplette di punti catturano informazioni sulla postura delle dita. Queste misure vengono combinate con vincoli topologici che riflettono la struttura anatomica della mano, permettendo di discriminare efficacemente tra diverse configurazioni gestuali.

Nonostante i significativi progressi raggiunti, il campo del riconoscimento gestuale continua a presentare sfide interessanti. La gestione di condizioni di illuminazione variabile rimane un problema aperto, così come il riconoscimento robusto in presenza di occlusioni parziali. La ricerca attuale si sta muovendo verso l'integrazione di tecniche di deep learning con approcci geometrici tradizionali, cercando di combinare la capacità di generalizzazione delle reti neurali con l'interpretabilità e la robustezza dell'analisi geometrica.

Un'altra direzione di ricerca promettente riguarda l'incorporazione di informazioni temporali nel processo di riconoscimento. I gesti umani risultano essere dinamici, e la capacità di analizzare sequenze di configurazioni nel tempo potrebbe portare a sistemi di riconoscimento più accurati e naturali. In questo contesto, tecniche come le reti neurali ricorrenti e i modelli di attenzione temporale stanno mostrando risultati promettenti.

Il presente lavoro si colloca in questo ricco contesto tecnologico, proponendo un approccio che combina la robustezza del tracking fornito da MediaPipe con l'efficacia dell'analisi geometrica per il riconoscimento specifico dei segni dell'alfabeto LIS. Questa combinazione permette di sfruttare i punti di forza di entrambe le tecnologie: la precisione e l'affidabilità del tracking dei landmark fornito da MediaPipe, e la chiarezza interpretativa e l'efficienza computazionale dell'analisi geometrica.

1.2 Lingua dei Segni e Tecnologia

La Lingua dei Segni Italiana (LIS) rappresenta un sistema linguistico completo e strutturato, riconosciuto ufficialmente dalla legislazione italiana nel 2021 [2]. A differenza di quanto comunemente si pensa, la LIS non è una semplice traduzione gestuale della lingua italiana parlata, ma un sistema linguistico autonomo con una propria grammatica, sintassi e un proprio lessico. Questa lingua si è evoluta naturalmente all'interno della comunità sorda italiana, sviluppando caratteristiche uniche che la distinguono sia dalla lingua italiana parlata sia dalle lingue dei segni di altri paesi.

La struttura della LIS si basa su cinque parametri fondamentali: la configurazione della mano, l'orientamento del palmo, il luogo di esecuzione del segno, il movimento e le componenti non manuali. Quest'ultimo elemento, che include espressioni facciali, movimenti del capo e posture del corpo, riveste un ruolo particolarmente importante nella trasmissione di sfumature semantiche e aspetti grammaticali. La complessità di questi elementi rende particolarmente sfidante lo sviluppo di sistemi tecnologici per il suo riconoscimento automatico.

Nel contesto specifico dell'alfabeto manuale LIS, utilizzato principalmente per la dattilogia di nomi propri o termini privi di un segno specifico, ogni lettera viene rappresentata da una particolare configurazione della mano. Questi segni, sebbene più semplici rispetto ai segni lessicali completi, presentano comunque sfide significative per il riconoscimento automatico, in quanto, alcune configurazioni possono apparire molto simili tra loro, e piccole variazioni nella posizione o nell'orientamento della mano possono modificare completamente il significato del segno.

Le prime ricerche nel campo del riconoscimento automatico della LIS risalgono agli anni '90, con un approccio iniziale basato su tecniche di pattern recognition applicate a immagini statiche. Un contributo significativo a questa linea di studio è stato offerto da Volterra et al. [9], i cui lavori hanno posto le basi per una comprensione approfondita della struttura linguistica della LIS, includendo sia gli aspetti manuali che quelli non manuali della lingua. Volterra e colleghi hanno evidenziato come i marcatori non manuali, quali le espressioni facciali e i movimenti della parte superiore del corpo, rivestano un ruolo fondamentale nella grammatica della LIS, in quanto veicolano informazioni sintattiche e grammaticali cruciali [9]. Questi studi hanno quindi fornito un quadro teorico fondamentale che, negli anni successivi, ha facilitato la progettazione di sistemi di riconoscimento automatico sempre più sofisticati, i quali tengono conto della natura complessa e multimodale della comunicazione in lingua dei segni.

Le sfide principali nel riconoscimento automatico della LIS includono la gestione della variabilità nella realizzazione dei segni, la dipendenza dal contesto e l'integrazione delle componenti non manuali. La variabilità è particolarmente rilevante poiché ogni segnante può eseguire lo stesso segno con lievi differenze, mantenendone invariato il significato. Questa caratteristica, naturale nelle lingue dei segni, richiede sistemi di riconoscimento sufficientemente flessibili da gestire queste variazioni mantenendo al contempo un'elevata accuratezza.

La dipendenza dal contesto rappresenta un'altra sfida significativa. Molti segni possono assumere significati diversi in base al contesto in cui vengono utilizzati, e spesso il significato completo emerge solo dalla combinazione di elementi manuali e non manuali. Questa caratteristica rende necessario lo sviluppo di sistemi che non si limitino al riconoscimento di configurazioni statiche, ma che siano in grado di interpretare sequenze di movimenti e di integrare informazioni provenienti da diverse modalità.

I recenti sviluppi tecnologici, in particolare l'avvento di sistemi di tracking delle mani sempre più accurati come MediaPipe, hanno aperto nuove possibilità per il riconoscimento automatico della LIS. La capacità di questi sistemi di fornire informazioni precise sulla posizione e configurazione delle mani in tempo reale ha permesso lo sviluppo di applicazioni più naturali e responsive. Tuttavia, resta ancora aperta la sfida di integrare efficacemente il riconoscimento delle componenti non manuali, aspetto fondamentale per una comprensione completa della comunicazione in LIS.

1.3 Sistemi di Autocorrezione

I sistemi di autocorrezione rappresentano un elemento fondamentale nell'elaborazione del linguaggio naturale, evolvendosi da semplici dizionari di confronto a sistemi sofisticati di analisi linguistica. Il loro sviluppo ha radici profonde nell'informatica, i primi approcci significativi risalgono agli anni '60 con il lavoro pionieristico di Frederick J. Damerau [10], il quale identificò che circa l'80% degli errori di battitura sono riconducibili a quattro operazioni fondamentali: inserimento, eliminazione, sostituzione e trasposizione di caratteri.

La base teorica dei moderni sistemi di autocorrezione si fonda sul concetto di distanza di edit, formalizzato da Levenshtein [11]. Questo approccio matematico definisce la distanza tra due stringhe come il numero minimo di operazioni elementari necessarie per trasformare una stringa nell'altra. L'algoritmo di Levenshtein ha stabilito le fondamenta per lo sviluppo di metodi più sofisticati di correzione automatica, introducendo un approccio quantitativo alla misurazione della similarità tra parole.

Peter Norvig [12] ha successivamente introdotto un approccio probabilistico alla correzione ortografica, combinando la distanza di edit con modelli statistici del linguaggio. Il suo lavoro ha dimostrato come l'integrazione di informazioni sulla frequenza delle parole e sui pattern comuni di errore possa migliorare significativamente l'accuratezza delle correzioni proposte.

Un ulteriore contributo fondamentale nell'evoluzione dei sistemi di autocorrezione è stato apportato da Kernighan et al. [13], che hanno sviluppato un modello di canale rumoroso per la correzione ortografica. Questo approccio considera gli errori di battitura come il risultato di una trasmissione attraverso un canale rumoroso, permettendo una modellazione più accurata dei pattern di errore tipici degli utenti.

I sistemi di autocorrezione moderni si basano su tre componenti principali:

La prima componente consiste nel rilevamento dell'errore, che deve bilanciare precisione e sensibilità per evitare sia falsi positivi che falsi negativi. Kukich [14] ha dimostrato l'importanza di considerare il contesto nella fase di rilevamento, evidenziando come molti errori siano identificabili solo attraverso l'analisi del contesto circostante. La seconda riguarda la generazione dei candidati di correzione. Questa fase utilizza varie tecniche di similarità lessicale e può beneficiare di strutture dati ottimizzate come i trie o gli alberi BK, come dimostrato da Zobel e Dart [15]. La terza, infine, si occupa della classificazione e del ranking dei candidati proposti. L'efficacia di questa fase dipende dalla capacità di integrare diverse fonti di informazione, come la frequenza delle parole nel corpus di riferimento e la plausibilità delle correzioni nel contesto specifico.

L'applicazione di questi sistemi nel contesto del riconoscimento della lingua dei segni presenta sfide uniche. La natura visivo-gestuale della comunicazione introduce una dimensione aggiuntiva di complessità nel processo di correzione, richiedendo l'adattamento degli algoritmi tradizionali per gestire le particolarità della traduzione da gesti a testo.

1.4 Sintesi Vocale e Feedback

La sintesi vocale, o Text-to-Speech (TTS), rappresenta un campo di ricerca in continua evoluzione che ha rivoluzionato il modo in cui i sistemi informatici comunicano con gli utenti. Thierry Dutoit [16] ha fornito una delle prime trattazioni complete delle tecniche di sintesi vocale, definendo i principi fondamentali che ancora oggi influenzano lo sviluppo di questi sistemi.

L'evoluzione del TTS ha attraversato diverse fasi tecnologiche. I primi sistemi, basati sulla concatenazione di fonemi pre-registrati, hanno lasciato il posto a approcci più sofisticati. Paul Taylor [17] ha documentato questa transizione, evidenziando come i sistemi moderni siano passati da un approccio basato su regole a uno basato su modelli statistici e, più recentemente, su reti neurali profonde.

Un contributo fondamentale alla moderna sintesi vocale è stato apportato da Aaron van den Oord et al. [18] con l'introduzione di WaveNet, un modello generativo profondo per la generazione di forme d'onda audio. Questo approccio ha dimostrato come le reti neurali possano produrre voci sintetiche di qualità quasi umana, stabilendo nuovi standard nel campo della sintesi vocale.

Nel contesto dei sistemi di feedback multimodale, la sintesi vocale assume un ruolo cruciale nell'accessibilità delle interfacce utente. Sharon Oviatt [19] ha condotto studi approfonditi sull'efficacia dei sistemi multimodali, dimostrando come l'integrazione di feedback vocale possa migliorare significativamente l'usabilità e l'accessibilità delle interfacce, particolarmente per utenti con diverse abilità.

L'applicazione della sintesi vocale nelle tecnologie assistive ha portato a sviluppi specifici per diverse lingue e contesti culturali. Nell'ambito italiano, Piero Cosi et al. [20] hanno sviluppato e valutato sistemi di sintesi specificatamente ottimizzati per la lingua italiana, affrontando le peculiarità fonetiche e prosodiche che la caratterizzano.

L'integrazione della sintesi vocale con i sistemi di riconoscimento gestuale presenta sfide uniche, particolarmente nel contesto della traduzione dalla lingua dei segni. La necessità di mantenere una latenza minima nel feedback vocale, pur garantendo un'alta qualità della sintesi, richiede un delicato bilanciamento tra prestazioni e qualità. Nicoletta Adamo-Villani et al. [21] hanno esplorato queste problematiche, fornendo linee guida per l'implementazione efficace di sistemi di feedback multimodale nel contesto dell'accessibilità.

L'efficacia del feedback multimodale non dipende solo dalla qualità della sintesi vocale, ma anche dalla sua integrazione temporale con altri elementi dell'interfaccia. La sincronizzazione tra il riconoscimento gestuale e la produzione vocale deve essere accuratamente calibrata per garantire un'esperienza utente fluida e naturale, come evidenziato dalle ricerche di Karpov e Ronzhin [22].

Capitolo 2

Architettura del Sistema

2.1 Visione d'Insieme

Il sistema sviluppato si presenta come un'architettura modulare composta da tre componenti principali che interagiscono tra loro per fornire un'esperienza di traduzione completa della Lingua dei Segni Italiana. Questa architettura è stata progettata per garantire massima flessibilità e manutenibilità, permettendo l'evoluzione indipendente di ciascun componente.

Il cuore del sistema è rappresentato dal modulo di riconoscimento gestuale, basato su MediaPipe, che si occupa del tracking delle mani e dell'analisi dei landmark in tempo reale. Questo componente elabora il flusso video dalla webcam e, attraverso una serie di analisi geometriche, identifica le configurazioni dell'alfabeto LIS. Il modulo implementa una pipeline di elaborazione che include la detection delle mani, l'estrazione dei landmark e l'analisi delle relazioni spaziali tra i punti chiave identificati.

Il secondo componente fondamentale è il sistema di autocorrezione, specificamente progettato per gestire le particolarità della traduzione dalla LIS. Questo modulo riceve le sequenze di lettere riconosciute e applica algoritmi di correzione basati su un dizionario specializzato e regole contestuali. La presenza di questo componente è cruciale per migliorare la robustezza del sistema, compensando eventuali errori di riconoscimento e garantendo una traduzione più accurata.

Il terzo elemento è il sistema di feedback multimodale, che comprende sia la visualizzazione in tempo reale dei risultati che la sintesi vocale. Questo componente è stato progettato per fornire un'esperienza utente completa e accessibile, permettendo sia il monitoraggio visivo del processo di riconoscimento che la comunicazione vocale dei risultati.

L'interazione tra questi componenti segue un flusso di elaborazione ben definito. Il processo inizia con l'acquisizione del video attraverso la webcam, con una risoluzione configurata a 1280x720 pixel per garantire un'adeguata qualità dell'immagine. I frame acquisiti vengono elaborati dal modulo di riconoscimento con una latenza prestabilita di 1.5 secondi per il riconoscimento di ogni lettera, un compromesso che garantisce stabilità e accuratezza nel riconoscimento.

Il sistema implementa anche una gestione intelligente degli spazi tra le parole, utilizzando una configurazione specifica della mano che viene riconosciuta con un ritardo di 1.2 secondi. Questa temporizzazione è stata calibrata empiricamente per ottimizzare l'esperienza utente, riducendo al minimo i falsi positivi mantenendo una buona reattività del sistema.

Un aspetto innovativo risulta essere la gestione del feedback in tempo reale. Il sistema fornisce un riscontro immediato attraverso la visualizzazione dei landmark rilevati, permettendo all'utente di verificare immediatamente la corretta interpretazione dei gesti. Questa caratteristica è particolarmente importante durante la fase di apprendimento del sistema.

La comunicazione tra i moduli è stata progettata per essere efficiente e robusta, con particolare attenzione alla gestione degli errori e alla sincronizzazione dei vari processi. Il sistema implementa un meccanismo di reset che richiede 2 secondi di visualizzazione simultanea di entrambe le mani, permettendo all'utente di ricominciare una nuova sessione di traduzione in modo intuitivo.

L'architettura è stata sviluppata ponendo particolare attenzione alla scalabilità e alla possibilità di future estensioni. Ogni componente è stato progettato come un modulo indipendente, con interfacce ben definite che permettono facili aggiornamenti e modifiche. Questa modularità facilita anche l'aggiunta di nuove funzionalità o il miglioramento di quelle esistenti senza impattare sul funzionamento complessivo del sistema.

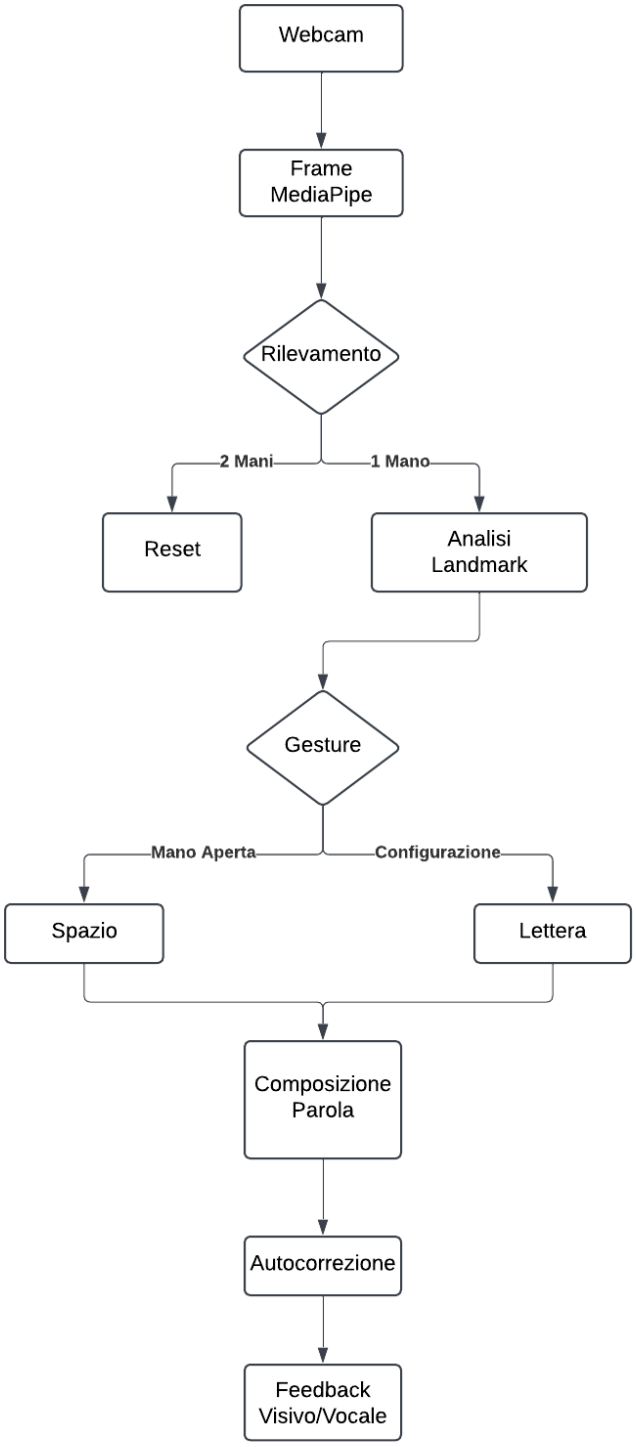

Figura 2.1: Diagramma di Flusso

2.1.1 Sistema di Riconoscimento

Il cuore del sistema, come già espresso nelle sezioni precedenti, si basa sull'analisi geometrica dei landmark forniti da MediaPipe. A differenza dei tradizionali approcci basati su machine learning, il sistema sfrutta relazioni geometriche precise tra i punti di riferimento delle mani per identificare le configurazioni dell'alfabeto LIS.

L'analisi geometrica si fonda su due tipologie principali di misurazioni: le distanze euclidee tra coppie di punti e gli angoli formati da terne di punti. Queste misurazioni fondamentali permettono di caratterizzare in modo univoco le diverse configurazioni dell'alfabeto LIS. La distanza tra la punta di un dito e il polso determina se il dito è piegato o disteso, mentre gli angoli tra le falangi valutano il grado di flessione delle dita.

Il sistema implementa un meccanismo di temporizzazione calibrato empiricamente per bilanciare accuratezza e usabilità:

- 1.5 secondi per il riconoscimento di una lettera.
- 1.2 secondi per il riconoscimento di uno spazio.
- 2.0 secondi per l'attivazione del reset.

La robustezza del riconoscimento è garantita da una serie di accorgimenti implementativi:

- Filtro sulle configurazioni instabili.
- Verifica della coerenza temporale.
- Gestione delle occlusioni parziali.

Un aspetto innovativo è la gestione degli spazi tra le parole, implementata attraverso il riconoscimento di una configurazione specifica con tutte le dita distese. Il sistema verifica simultaneamente l'elevazione delle dita rispetto alla base, la loro separazione e la posizione del pollice, permettendo un'identificazione affidabile dell'intenzione dell'utente di inserire uno spazio.

La scelta di un approccio geometrico, invece di tecniche di machine learning più complesse, garantisce prestazioni costanti senza necessità di training su dataset specifici, rendendo il sistema immediatamente operativo e facilmente adattabile a diverse condizioni d'uso.

2.1.2 Sistema di Autocorrezione

Il sistema di autocorrezione è stato progettato specificamente per gestire le peculiarità della traduzione dalla LIS al testo scritto. A differenza dei tradizionali sistemi di correzione ortografica, questo modulo è stato ottimizzato per gestire le particolari sfide che emergono dal riconoscimento gestuale, dove gli errori non derivano da errori di digitazione ma da potenziali ambiguità nell'interpretazione dei gesti.

L'architettura del sistema di autocorrezione si basa su un dizionario specializzato che include:

- Nomi propri italiani più comuni, selezionati per coprire un'ampia gamma di casi d'uso quotidiani.
- Principali capoluoghi italiani, facilitando la comunicazione relativa a località geografiche.
- Articoli e preposizioni della lingua italiana, fondamentali per la costruzione di frasi grammaticalmente corrette.

Un aspetto distintivo dell'implementazione è l'analisi contestuale delle parole. Il sistema assegna punteggi di confidenza più elevati alle correzioni che rispettano le regole grammaticali italiane. Per esempio, se una parola segue un articolo, il sistema privilegerà le correzioni che rappresentano sostantivi rispetto ad altre categorie grammaticali. Questo approccio contestuale si estende anche alle preposizioni, dove il sistema favorisce i capoluoghi o i nomi propri quando preceduti da preposizioni appropriate.

La valutazione dei candidati di correzione si basa su un sistema di scoring multifattoriale che considera la similarità con la parola originale, calcolata attraverso metriche di distanza tra stringhe, e la coerenza con il contesto grammaticale, valutando attentamente la posizione della parola nella frase. Il sistema considera anche la frequenza d'uso nel linguaggio comune, privilegiando le correzioni più probabili, e implementa una gestione speciale per parole di lunghezza particolare.

Il modulo implementa una gestione intelligente della confidenza, dove il punteggio finale di una correzione viene normalizzato tra 0.5 e 1.0. Questo range, scelto empiricamente, fornisce una chiara indicazione della qualità della correzione e facilita l'integrazione con gli altri componenti del sistema, permettendo una facile interpretazione dei risultati.

Un elemento innovativo è il sistema di feedback che fornisce suggerimenti multipli quando la correzione non è univoca. Per ogni parola corretta, il sistema mantiene una lista dei candidati alternativi, ordinati per punteggio di confidenza. Questa caratteristica risulta particolarmente utile nella gestione delle ambiguità nel riconoscimento dei segni, offrendo alternative quando il contesto non è sufficiente per una decisione univoca.

La scelta di utilizzare un approccio basato su dizionario, invece di tecniche più complesse come il machine learning, garantisce non solo prestazioni prevedibili e facilmente personalizzabili, ma assicura anche rapidità di esecuzione e facilità di manutenzione.

Questo approccio permette un controllo diretto sulle regole di correzione e mantiene il sistema indipendente da specifici dati di training, preservando al contempo il formato originale del testo per rispettare le intenzioni dell'utente nella formattazione finale.

2.1.3 Interfaccia Utente

L'interfaccia utente del sistema è stata progettata con l'obiettivo di fornire un feedback immediato e comprensibile attraverso un approccio multimodale che combina elementi visivi e sonori. Questa scelta progettuale nasce dalla necessità di garantire un'esperienza fluida e intuitiva, fondamentale per un sistema di traduzione in tempo reale.

Il sistema di feedback visivo si compone di tre elementi principali:

- Visualizzazione real-time dei landmark rilevati sulle mani.
- Indicatori di stato per il riconoscimento delle lettere e degli spazi.
- Area dedicata alla visualizzazione del testo tradotto e corretto.

La componente di feedback mostra all'utente i landmark rilevati sulle mani attraverso una rappresentazione grafica sovrapposta all'immagine della webcam. Questa visualizzazione immediata permette all'utente di comprendere come il sistema stia interpretando i suoi gesti e di correggere eventuali posizionamenti non ottimali. Il sistema implementa anche indicatori temporali che mostrano il progresso del riconoscimento, fornendo un riscontro chiaro sui tempi di attesa necessari per il completamento di ogni azione.

Il modulo di sintesi vocale rappresenta un elemento fondamentale per l'accessibilità del sistema. Ogni lettera riconosciuta viene immediatamente vocalizzata, fornendo un feedback sonoro che conferma il corretto riconoscimento. Al completamento di una parola o di una frase, il sistema fornisce una lettura completa del testo tradotto, includendo anche le eventuali correzioni applicate. La voce sintetizzata è stata configurata per garantire una chiara pronuncia della lingua italiana, con particolare attenzione alla naturalezza e alla comprensibilità.

L'interfaccia implementa un sistema di gestione degli errori intuitivo, evidenziando visivamente eventuali ambiguità nel riconoscimento e mostrando le possibili alternative di correzione. Questo approccio permette all'utente di mantenere il controllo sul processo di traduzione, intervenendo quando necessario per selezionare l'interpretazione corretta.

Un aspetto distintivo dell'interfaccia è la sua capacità di adattarsi dinamicamente alle condizioni di utilizzo. Il sistema regola automaticamente parametri come il contrasto e la luminosità della visualizzazione dei landmark per garantire una chiara visibilità in diverse condizioni di illuminazione. La disposizione degli elementi sullo schermo è stata ottimizzata per mantenere sempre visibili le informazioni più rilevanti, facilitando il monitoraggio continuo del processo di traduzione.

L'intero sistema di interfaccia è stato progettato considerando i principi di accessibilità e usabilità, con particolare attenzione alla minimizzazione del carico cognitivo dell'utente durante l'utilizzo. La combinazione di feedback visivo e sonoro, unita a una disposizione intuitiva degli elementi, crea un ambiente operativo che facilita l'apprendimento e l'utilizzo efficace del sistema di traduzione.

Capitolo 3

Implementazione

3.1 Tecnologie Utilizzate

L'implementazione del sistema si basa su un insieme accuratamente selezionato di tecnologie, ognuna scelta per le sue specifiche caratteristiche e capacità. Python è stato scelto come linguaggio principale di sviluppo per la sua versatilità e per l'ampia disponibilità di librerie specializzate nel campo della computer vision e dell'elaborazione del linguaggio naturale.

MediaPipe rappresenta il fulcro tecnologico del sistema di riconoscimento gestuale. Questa libreria, sviluppata da Google, fornisce un framework completo per l'analisi dei landmark delle mani attraverso la sua pipeline Hand Landmarker. L'implementazione utilizza specificamente il modulo mp_hands, che garantisce un tracking preciso e affidabile dei 21 punti chiave di ciascuna mano. La scelta di MediaPipe è stata determinante per il progetto grazie alla sua capacità di operare in tempo reale mantenendo un'elevata accuratezza, anche su hardware standard.

OpenCV (cv2) gestisce l'acquisizione e l'elaborazione del flusso video. La libreria è stata configurata per catturare immagini alla risoluzione di 1280x720 pixel, un compromesso ottimale tra qualità dell'immagine e performance. OpenCV si occupa anche della visualizzazione in tempo reale dei risultati, gestendo il rendering dei landmark e delle informazioni di feedback attraverso le sue funzioni di disegno e visualizzazione.

Il sistema di autocorrezione si basa sulla libreria SpellChecker, specificamente configurata per la lingua italiana. Questa libreria implementa algoritmi di correzione ortografica basati sulla distanza di Levenshtein, ma è stata estesa nel nostro sistema con funzionalità aggiuntive per la gestione del contesto e l'analisi delle relazioni grammaticali. La personalizzazione include l'integrazione di dizionari specializzati per nomi propri, luoghi e termini comuni della lingua italiana.

Per la componente di sintesi vocale, il sistema utilizza pyttsx3, una libreria text-to-speech offline che garantisce una latenza minima nella generazione del feedback vocale. La libreria è stata configurata per utilizzare il sintetizzatore vocale italiano, con parametri ottimizzati per la chiarezza e la naturalezza della pronuncia. L'implementazione gestisce la sintesi vocale in un thread separato per evitare blocchi nell'interfaccia utente durante la generazione dell'audio.

NumPy viene impiegato per l'elaborazione matematica efficiente dei dati dei landmark, fornendo le funzioni necessarie per i calcoli geometrici e l'analisi delle posizioni. La libreria si rivela particolarmente utile nella gestione degli array multidimensionali e nei calcoli vettoriali necessari per l'analisi delle configurazioni delle mani.

Il sistema di threading, implementato attraverso la libreria standard Python, gestisce la concorrenza tra i diversi componenti. Questo approccio garantisce che le operazioni di sintesi vocale e di elaborazione video possano procedere in parallelo senza interferire con la reattività dell'interfaccia utente. Un sistema di code gestisce la comunicazione tra i thread, assicurando un flusso ordinato di informazioni tra i componenti.

L'architettura software si avvale anche del modulo typing per l'implementazione di type hints, migliorando la manutenibilità del codice e facilitando il rilevamento precoce di errori. Questa scelta progettuale contribuisce alla robustezza complessiva del sistema, permettendo un più facile debugging e una documentazione più chiara delle interfacce tra i diversi moduli.

La gestione degli errori è implementata attraverso un sistema di eccezioni personalizzato, che garantisce la resilienza del sistema anche in condizioni non ottimali. Ogni componente implementa meccanismi di fallback appropriati, assicurando che il sistema possa continuare a funzionare anche in caso di problemi con singoli moduli.

L'intero stack tecnologico è stato orchestrato ponendo particolare attenzione all'efficienza e alla scalabilità, permettendo future estensioni e miglioramenti senza necessità di modifiche architetturali significative. Ogni tecnologia è stata integrata nel sistema attraverso interfacce ben definite, facilitando la manutenzione e l'evoluzione del codice.

3.2 Dettagli Implementativi

L'implementazione del sistema si articola in tre componenti principali: il riconoscimento gestuale, l'autocorrezione e il feedback.

Il sistema di riconoscimento gestuale si basa sull'analisi geometrica dei landmark delle mani. Le funzioni fondamentali per questa analisi sono implementate come segue:

```python
def get_distance(p1, p2):
    return np.sqrt((p1.x - p2.x)**2 + (p1.y - p2.y)**2)

def get_angle(p1, p2, p3):
    v1 = np.array([p1.x - p2.x, p1.y - p2.y])
    v2 = np.array([p3.x - p2.x, p3.y - p2.y])
    return np.abs(np.math.atan2(np.linalg.det([v1,v2]),
                                np.dot(v1,v2)))
```

Listing 3.1: Funzioni di base per l'analisi geometrica

Queste funzioni basilari vengono utilizzate per implementare analisi più complesse, come il riconoscimento di specifiche configurazioni delle dita. Per esempio, il rilevamento della chiusura delle dita è implementato come segue:

```python
def finger_is_closed(landmarks, finger_tip, finger_mcp):
    return get_distance(landmarks[finger_tip],
                        landmarks[WRIST]) < \
           get_distance(landmarks[finger_mcp],
                        landmarks[WRIST])

def finger_is_straight(landmarks, finger_mcp,
                       finger_pip, finger_tip):
    angle = get_angle(landmarks[finger_mcp],
                      landmarks[finger_pip],
                      landmarks[finger_tip])
    return angle > 2.8  # Circa 160 gradi
```

Listing 3.2: Rilevamento della posizione delle dita

Il riconoscimento degli spazi tra le parole rappresenta un aspetto critico del sistema. L'implementazione si basa sul rilevamento di una configurazione specifica della mano:

```
def is_hand_open(landmarks_dict):
    tips = [mp_hands.HandLandmark.INDEX_FINGER_TIP,
            mp_hands.HandLandmark.MIDDLE_FINGER_TIP,
            mp_hands.HandLandmark.RING_FINGER_TIP,
            mp_hands.HandLandmark.PINKY_TIP]
    mcps = [mp_hands.HandLandmark.INDEX_FINGER_MCP,
            mp_hands.HandLandmark.MIDDLE_FINGER_MCP,
            mp_hands.HandLandmark.RING_FINGER_MCP,
            mp_hands.HandLandmark.PINKY_MCP]

    fingers_up = all(
        landmarks_dict[tip].y < landmarks_dict[mcp].y - 0.1
        for tip, mcp in zip(tips, mcps))
    fingers_spread = all(
        abs(landmarks_dict[tips[i]].x -
            landmarks_dict[tips[i+1]].x) > 0.04
        for i in range(len(tips)-1))
    thumb_up = landmarks_dict[
        mp_hands.HandLandmark.THUMB_TIP].y < \
        landmarks_dict[mp_hands.HandLandmark.THUMB_MCP].y

    return fingers_up and fingers_spread and thumb_up
```

Listing 3.3: Implementazione del riconoscimento degli spazi

Segue ulteriormente un esempio di come è stata implementata la geometria di MediaPipe per il riconoscimento delle lettere dell'alfabeto LIS:

```
def is_A(landmarks):
    fingers_closed = all(
        finger_is_closed(landmarks, tip, mcp)
        for tip, mcp in [
            (INDEX_FINGER_TIP, INDEX_FINGER_MCP),
            (MIDDLE_FINGER_TIP, MIDDLE_FINGER_MCP),
            (RING_FINGER_TIP, RING_FINGER_MCP),
            (PINKY_TIP, PINKY_MCP)
        ])
    thumb_folded = get_distance(
        landmarks[THUMB_TIP],
        landmarks[INDEX_FINGER_MCP]) < 0.1

    return fingers_closed and thumb_folded
```

Listing 3.4: Implementazione del riconoscimento della lettera A

Il sistema di autocorrezione implementa un approccio basato su dizionario con analisi contestuale. La funzione di ricerca della parola più vicina illustra questo approccio:

```python
def find_closest_word(self, word: str) -> Tuple[str, float]:
    if not word or word.isspace():
        return word, 1.0

    if self.spell.known([word]):
        return word, 1.0

    correction = self.spell.correction(word)

    if correction is None or correction == word:
        return word, 0.0

    candidates = self.spell.candidates(word)
    if candidates:
        confidence = 1.0 / len(candidates)
        confidence = 0.5 + (confidence * 0.5)
    else:
        confidence = 0.0

    return correction, confidence
```

Listing 3.5: Implementazione della ricerca parola più vicina

Il feedback vocale viene gestito attraverso un sistema di code in un thread separato, garantendo che la sintesi vocale non blocchi l'interfaccia utente:

```
def _process_speech_queue(self):
    while self.is_running:
        try:
            text = self.speech_queue.get(timeout=0.1)
            if text:
                self.engine.say(text)
                self.engine.runAndWait()
            self.speech_queue.task_done()
        except queue.Empty:
            continue
        except Exception as e:
            print(f"Errore durante la sintesi vocale: {e}")
```

Listing 3.6: Implementazione del sistema di feedback vocale

La gestione dei tempi di riconoscimento è implementata attraverso un sistema di temporizzazione che bilancia accuratezza e reattività:

```
LETTER_SAVE_DELAY = 1.5
HAND_DETECTION_DELAY = 1.5
RESET_DELAY = 2
SPACE_DELAY = 1.2
```

Listing 3.7: Gestione dei tempi di riconoscimento

Questi valori sono stati calibrati empiricamente per ottimizzare l'esperienza utente, fornendo un compromesso efficace tra la precisione del riconoscimento e la fluidità dell'interazione. L'integrazione di questi componenti avviene nel loop principale del programma, dove ogni frame viene elaborato sequenzialmente attraverso la pipeline di riconoscimento, correzione e feedback. Particolare attenzione è stata posta alla gestione degli errori e alla robustezza del sistema, implementando meccanismi di fallback appropriati per ogni componente critico.

3.3 Gestione degli Eventi

L'architettura del sistema è stata progettata per gestire efficientemente il flusso di eventi e la pipeline di elaborazione. Il cuore di questa gestione è implementato nel loop principale del programma, dove vengono orchestrati i vari componenti del sistema.

La gestione del flusso video e il riconoscimento delle mani sono implementati come segue:

```
while capture.isOpened():
    success, image = capture.read()
    if not success:
        print("Ignoring empty camera frame.")
        break

    image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
    image.flags.writeable = False
    results = hands.process(image)
    image.flags.writeable = True
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

Listing 3.8: Gestione del flusso video e riconoscimento

La gestione degli stati del sistema e il riconoscimento delle configurazioni vengono controllati attraverso un sistema di timer e stati:

```
if results.multi_hand_landmarks and len(results.multi_hand_landmarks) == 2:
    if not is_resetting:
        two_hands_start_time = current_time
        is_resetting = True

    elif current_time - two_hands_start_time >= RESET_DELAY:
        saved_letters = []
        current_phrase = ""
        phrase_shown = False
        is_resetting = False
        corrected_phrase = ""
        corrections = []

        two_hands_start_time = None
        detection_started = False
        hand_detection_start_time = None

        space_start_time = None
        waiting_for_space = False
```

Listing 3.9: Gestione degli stati e temporizzazioni

Ulteriormente è stata implementata una gestione sofisticata degli spazi e della composizione delle parole. Quando viene rilevata la configurazione appropriata, il sistema attiva un timer dedicato:

```
if is_hand_open(landmarks_dict):
    if not waiting_for_space:
        space_start_time = current_time
        waiting_for_space = True

    elif current_time - space_start_time >= SPACE_DELAY:
        if not saved_letters or saved_letters[-1] != " ":
            saved_letters.append(" ")

        waiting_for_space = False
        space_start_time = None
```

Listing 3.10: Gestione degli spazi tra parole

La pipeline di riconoscimento include anche un sistema di feedback visivo in tempo reale che mostra lo stato del riconoscimento e le correzioni proposte:

```
if current_letter:
    time_left = max(0, LETTER_SAVE_DELAY -
                    (current_time - letter_start_time))

    cv2.putText(image, f"Lettera: {current_letter} ({time_left:.1f}s)",
                (10, 50), cv2.FONT_HERSHEY_SIMPLEX,
                1, (255, 255, 255), 2)

    cv2.putText(image, f"Lettere Salvate: {''.join(saved_letters)}",
                (10, 110), cv2.FONT_HERSHEY_SIMPLEX,
                1, (255, 255, 255), 2)
```

Listing 3.11: Implementazione del feedback visivo

Il sistema di feedback vocale viene gestito in modo asincrono attraverso una coda di messaggi, evitando blocchi nell'elaborazione principale:

```
if corrections:
    synthesizer.speak_phrase(f"Ha composto: {current_phrase}")
    time.sleep(0.3)
    synthesizer.speak_phrase(f"Forse intendeva: {corrected_phrase}")
else:
    synthesizer.speak_phrase(current_phrase)
```

Listing 3.12: Gestione del feedback vocale

L'intero sistema di gestione degli eventi è stato progettato per essere resiliente agli errori e per fornire un'esperienza utente fluida. I timer e i ritardi sono stati calibrati empiricamente per bilanciare la reattività del sistema con l'accuratezza del riconoscimento.

La gestione della memoria e delle risorse è implementata attraverso meccanismi di cleanup appropriati, assicurando che il sistema rilasci correttamente le risorse quando necessario:

```
capture.release()
cv2.destroyAllWindows()
synthesizer.cleanup()
```

Listing 3.13: Gestione delle risorse

Questa architettura di gestione degli eventi fornisce un framework robusto e flessibile per l'integrazione dei vari componenti del sistema, mantenendo al contempo prestazioni ottimali e un'elevata reattività nell'interazione con l'utente.

Capitolo 4

Valutazione e Risultati

4.1 Metodologia di Test

Lo sviluppo del sistema è stato guidato da un approccio iterativo e incrementale, dove ogni funzionalità è stata testata attraverso una metodologia di validazione empirica diretta. Questo approccio, sebbene meno formale rispetto ai tradizionali protocolli di test automatizzati, ha permesso una rapida iterazione e ottimizzazione del sistema in base alle esigenze reali di utilizzo.

La metodologia di test adottata si è basata su tre principi fondamentali:

- Verifica immediata delle funzionalità implementate.
- Ottimizzazione iterativa basata sull'esperienza diretta.
- Validazione qualitativa delle prestazioni del sistema.

Il processo di sviluppo e test ha seguito un ciclo continuo di implementazione, verifica e raffinamento. Per ogni componente del sistema, sono stati eseguiti test manuali mirati a verificare specifici aspetti funzionali. Il riconoscimento gestuale, ad esempio, è stato testato eseguendo ripetutamente i segni dell'alfabeto LIS in diverse condizioni di illuminazione e posizionamento, permettendo di identificare e correggere eventuali problemi di riconoscimento.

La calibrazione dei tempi di riconoscimento è stata particolarmente significativa nel processo di test. I valori finali delle costanti temporali (1.5 secondi per il riconoscimento delle lettere, 1.2 secondi per gli spazi e 2 secondi per il reset) sono il risultato di numerose iterazioni di test, dove ogni valore è stato progressivamente raffinato fino a raggiungere un equilibrio ottimale tra accuratezza e usabilità.

Il sistema di autocorrezione è stato validato attraverso una serie di prove pratiche, inserendo deliberatamente sequenze di lettere imprecise e verificando la capacità del sistema di proporre correzioni appropriate. Questo processo ha permesso di affinare i parametri del sistema e di ampliare il dizionario di riferimento con termini frequentemente utilizzati.

L'approccio empirico alla validazione ha permesso di identificare e risolvere rapidamente diverse criticità:

- Problemi di riconoscimento in condizioni di illuminazione non ottimali.
- Difficoltà nel distinguere configurazioni simili delle dita.
- Latenze eccessive nel feedback vocale.

Sebbene non siano stati condotti test formali con un ampio gruppo di utenti, il processo di sviluppo e validazione ha beneficiato di un feedback continuo e diretto, permettendo di ottimizzare il sistema in base all'esperienza pratica di utilizzo. Questo approccio pragmatico ha consentito di creare un sistema che, pur non essendo stato sottoposto a una validazione statistica formale, si è dimostrato funzionale e affidabile nell'uso pratico.

La scelta di questo approccio di test, sebbene non convenzionale in un contesto accademico tradizionale, si è rivelata efficace per lo sviluppo di un prototipo funzionale, fornendo una base solida per future implementazioni più rigorose e test sistematici con un campione più ampio di utenti.

4.2 Limitazioni e Miglioramenti

Il sistema sviluppato, pur raggiungendo gli obiettivi prefissati, presenta alcune limitazioni che aprono la strada a interessanti prospettive di miglioramento. Una delle limitazioni principali riguarda il sistema di autocorrezione, attualmente basato su un dizionario statico e regole predefinite. L'integrazione di tecniche di machine learning potrebbe potenziare significativamente questa componente, permettendo l'implementazione di un modello di linguaggio in grado di considerare non solo le singole parole, ma l'intero contesto della frase.

La gestione delle condizioni di illuminazione rappresenta un'altra area di potenziale miglioramento. Sebbene il sistema funzioni efficacemente in condizioni standard, l'implementazione di tecniche avanzate di pre-elaborazione dell'immagine potrebbe aumentarne la robustezza in diverse condizioni ambientali. Parallelamente, la gestione degli spazi e della punteggiatura potrebbe essere raffinata attraverso l'introduzione di gesti più intuitivi, facilitando la composizione di testi strutturati.

Lo sviluppo futuro del sistema offre numerose prospettive interessanti. Una delle più promettenti riguarda l'integrazione con smart glasses, che aprirebbe nuovi scenari di utilizzo. Questa evoluzione permetterebbe una maggiore portabilità e naturalezza d'uso, consentendo agli utenti di interagire con il sistema in modo più discreto e socialmente integrato. La sovrapposizione di informazioni visive direttamente nel campo visivo dell'utente rappresenterebbe un significativo passo avanti nell'usabilità del sistema.

L'implementazione di modelli di machine learning potrebbe rivoluzionare diversi aspetti del sistema. L'addestramento di reti neurali specifiche per il riconoscimento gestuale e l'analisi del contesto linguistico permetterebbe di sviluppare un sistema più adattivo, capace di apprendere dalle caratteristiche specifiche di ciascun utente. Questo approccio personalizzerebbe l'esperienza d'uso, migliorando progressivamente l'accuratezza del riconoscimento.

Un'evoluzione particolarmente significativa sarebbe l'estensione del sistema al riconoscimento di segni completi della LIS, superando i limiti dell'attuale focus sull'alfabeto manuale. Questo sviluppo richiederebbe l'implementazione di sistemi più sofisticati per il riconoscimento di gesti dinamici e l'interpretazione delle componenti non manuali della lingua dei segni, aprendo la strada a una traduzione più naturale e completa.

La scalabilità del sistema potrebbe essere migliorata attraverso l'implementazione di un'architettura distribuita. Questo approccio permetterebbe di spostare il carico computazionale più pesante su server dedicati, ottimizzando le prestazioni sui dispositivi degli utenti e aprendo la possibilità di elaborazione in tempo reale anche su dispositivi con risorse limitate.

Per garantire l'efficacia di questi sviluppi futuri, sarà fondamentale implementare un rigoroso sistema di validazione, conducendo test sistematici con un ampio campione di utenti in diverse condizioni di utilizzo. Questo processo di validazione fornirà dati empirici preziosi per guidare ulteriori miglioramenti e ottimizzazioni del sistema.

Queste prospettive di sviluppo delineano un percorso evolutivo che potrebbe trasformare il sistema attuale in uno strumento ancora più potente e versatile per l'abbattimento delle barriere comunicative, contribuendo significativamente al campo dell'accessibilità e della comunicazione assistiva.

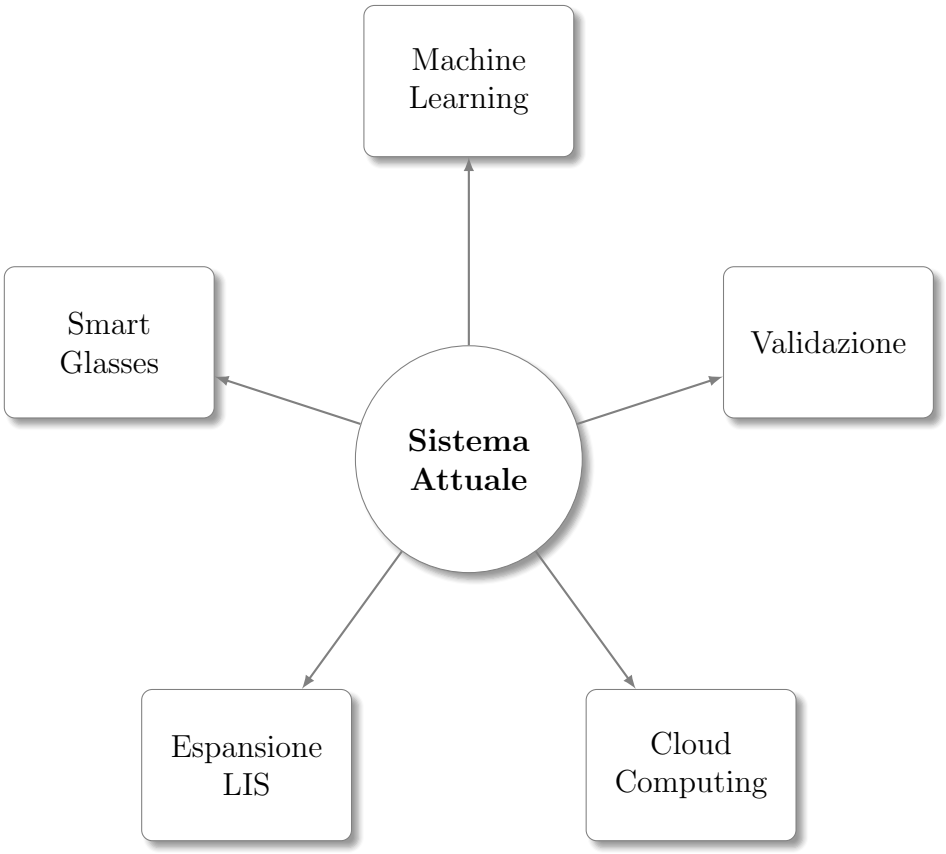

Figura 4.1: Prospettive di sviluppo futuro

Conclusioni

Il lavoro di ricerca e sviluppo presentato in questa tesi ha portato alla realizzazione di un sistema innovativo per il riconoscimento e la traduzione della Lingua dei Segni Italiana, contribuendo in modo significativo al campo delle tecnologie assistive. L'architettura proposta, che integra computer vision, analisi geometrica e sistemi di feedback multimodale, rappresenta un approccio originale alla sfida della comunicazione tra comunità sorda e udente.

L'analisi approfondita delle tecnologie esistenti, presentata nello stato dell'arte, ha guidato le scelte progettuali verso una soluzione che privilegia robustezza e affidabilità. L'utilizzo di MediaPipe come framework di base per il riconoscimento dei landmark, combinato con un'analisi geometrica sofisticata, ha permesso di sviluppare un sistema che non richiede fasi di training preliminari e mantiene prestazioni costanti in diverse condizioni operative.

Dal punto di vista architetturale, il sistema si distingue per la sua modularità e scalabilità. La separazione netta tra il modulo di riconoscimento gestuale, il sistema di autocorrezione e l'interfaccia utente permette una manutenzione efficiente e facilita l'integrazione di futuri miglioramenti. Il sistema di autocorrezione, in particolare, implementa un approccio innovativo che combina analisi contestuale e dizionari specializzati, dimostrando come tecniche relativamente semplici possano produrre risultati significativi quando opportunamente integrate.

L'implementazione del feedback multimodale rappresenta un altro elemento distintivo del progetto. La combinazione di feedback visivo in tempo reale e sintesi vocale crea un'esperienza utente intuitiva e completa, essenziale per un'applicazione destinata all'uso quotidiano. La gestione asincrona della sintesi vocale e l'ottimizzazione delle performance grafiche testimoniano l'attenzione posta agli aspetti pratici dell'utilizzo del sistema.

I risultati ottenuti durante la fase di sviluppo e testing hanno confermato la validità dell'approccio scelto. Il sistema ha dimostrato di poter gestire efficacemente il riconoscimento dell'alfabeto LIS, mantenendo un equilibrio ottimale tra accuratezza e tempi di risposta. La scelta di implementare meccanismi di temporizzazione calibrati empiricamente ha contribuito significativamente alla robustezza del sistema, prevenendo riconoscimenti errati senza compromettere l'usabilità.

La validazione empirica del sistema ha evidenziato non solo i punti di forza dell'implementazione attuale, ma anche le aree che potrebbero beneficiare di ulteriori sviluppi. L'esperienza acquisita durante lo sviluppo suggerisce che l'evoluzione verso un'architettura distribuita e l'implementazione di capacità di apprendimento adattivo rappresenterebbero i prossimi passi naturali nel percorso di miglioramento del sistema.

Dal punto di vista scientifico, questo lavoro contribuisce alla letteratura esistente dimostrando l'efficacia di un approccio geometrico al riconoscimento della LIS. L'architettura proposta può servire come base per futuri sviluppi nel campo delle tecnologie assistive, fornendo un framework modulare e ben documentato per l'implementazione di sistemi simili.

In conclusione, il progetto presentato in questa tesi non rappresenta solo un'implementazione tecnica di successo, ma si configura come un contributo significativo al campo delle tecnologie assistive per la comunità sorda. L'approccio metodologico adottato, le soluzioni tecniche implementate e i risultati ottenuti dimostrano come l'innovazione tecnologica, quando guidata da una chiara comprensione delle esigenze degli utenti, possa produrre strumenti concreti per l'abbattimento delle barriere comunicative. Il sistema sviluppato, con le sue potenzialità di espansione e miglioramento, si propone come punto di partenza per futuri sviluppi nel campo del riconoscimento automatico della lingua dei segni, aprendo nuove prospettive per una comunicazione più inclusiva e accessibile.

Bibliografia

[1] Ente Nazionale Sordi. Rapporto sulla condizione delle persone sorde in italia, 2023.

[2] Sito Ufficiale della Camera dei deputati. Legge 23 maggio 2021, articolo 34-ter: Riconoscimento della lingua dei segni italiana e disposizioni per la tutela delle persone sorde, 2021.

[3] Universita Ca' Foscari Venezia. L'italia riconosce la LIS. *cafoscariNEWS*, 2021.

[4] Lega del Filo d'Oro. Report annuale sulle barriere comunicative, 2023.

[5] Associazione Italiana Sordi. Rapporto sulle barriere sanitarie per la comunità sorda, 2023.

[6] ISTAT. Occupazione e disabilità: Dati e statistiche annuali, 2023.

[7] Google LLC. *MediaPipe Documentation*, 2024.

[8] Zahoor Zafrulla, Helene Brashear, Thad Starner, Harley Hamilton, and Peter Presti. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*. ACM, 2011.

[9] Virginia Volterra, Serena Corazza, and Elena Radutzky. *La Lingua dei Segni Italiana: La comunicazione visivo-gestuale dei sordi*. Il Mulino, 2004.

[10] Fred J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[11] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[12] Peter Norvig. How to write a spelling corrector. 2007.

[13] Mark D. Kernighan, Kenneth W. Church, and William A. Gale. A spelling correction program based on a noisy channel model. In *COLING 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*, 1990.

[14] Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.

[15] Justin Zobel and Philip Dart. Finding approximate matches in large lexicons. *Software: Practice and Experience*, 25(3):331–345, 1995.

[16] Thierry Dutoit. *An Introduction to Text-to-Speech Synthesis*. Springer, 1997.

[17] Paul Taylor. *Text-to-Speech Synthesis*. Cambridge University Press, 2009.

[18] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[19] Sharon Oviatt. Multimodal interfaces. *The Human-Computer Interaction Handbook*, pages 286–304, 2003.

[20] Piero Cosi, Carlo Drioli, and Fabio Tesser. Italian text-to-speech synthesis: Research, development and evaluation. *Language Resources and Evaluation*, 53:917–953, 2019.

[21] Nicoletta Adamo-Villani, Ethan Carpenter, and Laura Arns. An immersive virtual environment for learning sign language mathematics. 2006.

[22] Alexey Karpov and Andrey Ronzhin. A universal assistive technology with multimodal input and multimedia output interfaces. *Communications in Computer and Information Science*, 435:369–378, 2014.