**Report del 3° Progetto di 3D Perception**

Authors: Francesco De Patre, Davide Faroldi Lo Presti

**Rilevamento e Localizzazione di Persone tramite Kinect, GPS, YOLO e bussola, con Data Fusion tramite rete neurale**

1. **Introduzione**

Il progetto si è concentrato sul migliorare e integrare il progetto precedente, aggiungendo ai sensori già presenti (quello RGB, di profondità del Kinect e il segnale GPS) una bussola per poter calcolare la posizione globale dei rilevamenti. Tutti i dati vengono poi uniti ed elaborati attraverso un neural network con implementazione di LSTM.

L’obiettivo principale è quello di sviluppare un sistema di rilevamento dei volti e di misurazione della distanza in tempo reale che applica data fusion per determinare la possibile posizione globale dei volti individuati.

1. **Librerie, algoritmi e framework utilizzati**

* **YOLO11**: L’ultimo modello di deep learning di Ultralytics per il rilevamento rapido di oggetti, addestrato su un dataset personalizzato
* **MEGAFACE**: Dataset contente le immagini di volti/persone per il training
* **PyTorch**: Framework per il training di YOLO e del neural network custom
* **Freenect**: Libreria per interfacciarsi con il Kinect
* **OpenCV**: Utilizzato per il pre-processing delle immagini e per l’output
* **NumPy**: Utilizzato per la gestione di matrici e operazioni numeriche
* **Folium**: Libreria per la creazione di mappe interattive

1. **Struttura del sistema**

Il sistema è composto da diverse fasi e supporta sia l’elaborazione real-time che la post-elaborazione.

L’elaborazione **real-time** è composta dalle seguenti fasi:

* **Calibrazione**: : Il processo di calibrazione delle telecamere del sensore Kinect è basato sull’acquisizione di immagini di una scacchiera posta in diverse posizioni davanti al dispositivo. La scacchiera viene utilizzata dal Kinect come riferimento fisso in quanto i punti d’angolo sono disposti in un pattern regolare. Durante la calibrazione, il sistema rileva gli angoli della scacchiera nelle immagini RGB e di depth, associando ogni punto rilevato alle coordinate corrispondenti nello spazio. Una volta raccolte le immagini, un algoritmo calcola le matrici intrinseche delle telecamere e i loro coefficienti di distorsione, necessari per correggere l’immagine e migliorare la precisione della percezione spaziale. I parametri ottenuti vengono poi salvati e possono essere usati successivamente per migliorare l’allineamento tra immagini e dati di profondità.
* **Acquisizione delle Immagini RGB, della Mappa di Profondità, del segnale GPS e dei dati della bussola**: In questa fase i sensori del Kinect catturano i dati RGB e di profondità, mentre il GPS i dati di posizione globali e la bussola quelli di direzione
* **Rilevamento degli Oggetti con YOLO**: Una volta acquisita l’immagine , questa viene preparata per essere processata dal modello YOLO. Nel codice viene caricato il modello “last.pt” e le classi degli oggetti “obj.names”.
* **Passaggio dei dati alla rete e elaborazione:** I dati vengono trasformati in tensori, messi in una lista frame per frame (massimo 10 frame) e passati al modello, che ne estrae le feature frame per frame, poi le concatena e restituisce la sua stima
* **Visualizzazione e Feedback**: Gli oggetti rilevati vengono visualizzati sull’immagine RGB con un bounding box, il loro ID e la distanza stimata dalla rete. In aggiunta ad ogni frame viene generato e riscritto un file html contenente una mappa centrata nel segnale gps, e tanti punti di interesse quanti gli oggetti rilevati, nella loro possibile posizione nello spazio.

La **post-elaborazione** invece verte sulle seguenti fasi:

* **Acquisizione delle immagini RGB, della Mappa di Profondità, del segnale GPS e i dati della bussola**: In questa fase il programma carica il video RGB, il video di profondità, il segnale GPS (contenuto in un file) e i dati della bussola (anch’essi contenuti in un file).

Il **Rilevamento degli oggetti con YOLO**, il **Passaggio dei dati alla rete e elaborazione** e la **Visualizzazione e Feedback** sono i medesimi dell’elaborazione real-time

1. **Conclusioni**

Questo progetto integra con successo il progetto precedente, aggiungendogli dati da una bussola e un meccanismo di data fusion.