Università Politecnica delle Marche

Ingegneria Informatica e dell'Automazione

Titolo significativo

Corso di

Laboratorio di Automazione

Anno accademico 2024-2025

Studenti:
Pizzuto Andrea
Meloccaro Lorenzo
Percipalle Noemi

Professore:
Andrea Bonci

Dottorando:
Serafini Andrea
Pellicani Ilaria

Dipartimento di Ingegneria dell'Informazione

Indice

1 Introduzione

Il capitolo fornisce una panoramica del progetto, descrivendo l'obiettivo principale di sviluppare e simulare la movimentazione di un robot TM-900 con un tessuto, attraverso l'integrazione di ROS2 e Unity. Viene fornita una panoramica delle motivazioni per l'uso di questi software e delle principali fasi del progetto.

Introduzione

Il presente progetto ha come obiettivo lo sviluppo e la simulazione della movimentazione di un robot TM-900 con un tessuto (cloth), utilizzando l'integrazione di software avanzati come ROS2 e Unity. Il contesto applicativo del progetto si colloca nell'ambito industriale e robotico, ponendo particolare attenzione sull'ottimizzazione dei processi di smontaggio di oggetti, come abiti e tessuti, mediante l'interazione con il robot. L'azienda coinvolta utilizza il software Cloth 3D per la modellazione degli abiti, mentre Unity e Gazebo sono impiegati per simulare la movimentazione del robot e l'interazione con gli oggetti. Uno degli obiettivi di questo progetto è quello di analizzare in dettaglio l'integrazione tra ROS2 e Unity, scelta motivata dalle potenzialità di questi due strumenti. ROS2 offre una gestione efficiente della comunicazione tra il robot e il sistema di simulazione, mentre Unity fornisce una grafica avanzata che consente di visualizzare e simulare i movimenti del robot in tempo reale, sfruttando ROS2. L'integrazione di questi due software è stata scelta per soddisfare la necessità di creare un ambiente simulativo realistico, utile non solo per eseguire test simulativi, ma anche per implementare successivamente il sistema su un robot reale. In questo modo, l'integrazione tra ROS2 e Unity consente di gestire il robot e simularlo in ambiente virtuale, migliorando l'efficienza e la precisione nelle operazioni robotiche. Il progetto affronta diversi task, tra cui la valutazione e la compatibilità dei tessuti con i vari software, la simulazione della movimentazione del robot e dell'oggetto da smontare in un ambiente condiviso, l'acquisizione e la definizione di una sequenza di fasi di smontaggio, l'esecuzione delle fasi in simulazione con il robot e, infine, l'esecuzione delle stesse fasi su un robot reale.

2 Software

In questo capitolo verranno analizzati i software principali utilizzati per il progetto: ROS 2, Unity e Gazebo. Verranno fornite una panoramica generale, informazioni sul loro utilizzo e sulle versioni adottate, evidenziando eventuali problematiche riscontrate e le soluzioni adottate.

2.1 ROS2

Figure 2.1: *Logo Unity*

ROS 2 (Robot Operating System 2) è un framework open-source per lo sviluppo di applicazioni robotiche. Fornisce una serie di strumenti, librerie e convenzioni per facilitare la comunicazione tra i componenti software di un sistema robotico. ROS 2 è progettato per migliorare la scalabilità, la sicurezza e la compatibilità con i sistemi distribuiti rispetto al suo predecessore, ROS 1. Il suo utilizzo è diffuso in ricerca e industria per il controllo di robot mobili, bracci robotici e veicoli autonomi. In questo progetto, ROS 2 è stato utilizzato per gestire la comunicazione tra il robot simulato in Unity e i componenti di controllo. ROS 2 permette lo scambio di messaggi tra i nodi, abilitando il controllo del robot da Unity e viceversa. In particolare, il framework ha permesso l'integrazione con il sistema di simulazione, la gestione delle traiettorie e il controllo dei giunti del manipolatore. Inizialmente, il progetto è stato avviato con ROS 2 Jazzy, ma si sono riscontrati diversi problemi di compatibilità e instabilità, in particolare con l'integrazione del ROS-TCP Connector per Unity. Per questo motivo, è stato deciso di passare a ROS 2 Humble, una versione più stabile e ampiamente supportata, che ha garantito una migliore compatibilità con gli strumenti di sviluppo utilizzati.

2.2 Unity

Unity è un motore di gioco e simulazione ampiamente utilizzato per la creazione di ambienti interattivi in tempo reale. Sebbene sia nato per lo sviluppo di videogiochi, il suo utilizzo si è

Figure 2.2: *Logo Unity*

esteso ad applicazioni di simulazione, robotica, realtà virtuale e aumentata. Unity permette di creare ambienti 3D realistici e interattivi grazie al suo motore grafico avanzato e alle sue funzionalità di scripting basate su C#. In questo progetto, Unity è stato utilizzato come ambiente di simulazione per il robot. Grazie alla sua compatibilità con ROS2 tramite il ROS-TCP Connector, Unity ha permesso di visualizzare il robot, simulare i suoi movimenti e interagire con l'ambiente circostante. Il motore fisico di Unity è stato sfruttato soprattutto per replicare le dinamiche fisiche della maglietta, garantendo una simulazione più realistica delle interazioni tra un indumento e gli oggetti della scena. Per lo sviluppo del progetto, abbiamo utilizzato Unity 6, la versione più recente al momento di sviluppo del progetto.

3 Materiali e Metodi

Inserire sempre un capitolo "Hardware" dove si descrivono tutti i componenti hardware utilizzati (schede, sensori, attuatori..). Dedicare un paragrafo ad ogni componente. Per ogni componente riportare le seguenti informazioni: modello specifico, riferimento bibliografico con link ad un sito web con la documentazione del componente (es [1]), immagine, pinout, solo le informazioni rilevanti per lo svolgimento del task.

3.1 Integrazione ROS-Unity

3.1.1 ROS-TCP Connector

3.1.2 ROS-TCP Endpoint

3.1.3 URDF Importer

3.2 Moveit

3.3 Gazebo

3.4 Cloth di Unity

3.5 Camera

4 Risultati ottenuti e discussione

5 Conclusioni e prospettive future

A Appendice1

Se necessario ricorrete alle appendici per spiegare le parti "di contorno" dell'attività svolta e/o ciò che non riuscite ad inserire nello schema generale dei capitoli della relazione (es acquisizione dei dati con Matlab).

Bibliografia

[1] “MD10C DCdriver, url = https://www.cytron.io/p-10amp-5v-30v-dc-motor-driver.”