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.

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 ROS2*

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

In questo capitolo saranno presentati i materiali(principalmente software) e i metodi, quindi le tecniche utilizzate, inoltre vengono dettagliate le tecnologie impiegate e le procedure seguite, in modo da permettere la riproducibilità del progetto. In particolare la prima sezione ***Integrazione ROS-Unity*** *approfondisce l'integrazione tra ROS 2 e Unity, analizzando strumenti chiave come ROS-TCP Connector, ROS-TCP Endpoint e URDF Importer (3.1.3). La sezione successiva tratta l'uso di* ***MoveIt*** *per la pianificazione del movimento, mentre la 3.3 descrive l'impiego di* ***Gazebo*** *per la simulazione robotica. Successivamente, la sezione 3.4 introduce* ***Cloth di Unity****, utilizzato per la simulazione di tessuti, e la 3.5 approfondisce la gestione della* ***Camera*** *all'interno dell'ambiente simulato.*

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

Il sistema **Cloth di Unity** offre una soluzione basata sulla fisica per simulare tessuti e materiali flessibili all'interno di ambienti 3D. Sebbene sia stato progettato principalmente per rappresentare abbigliamento su personaggi, può essere utilizzato anche per altri scopi, come bandiere, tende o qualsiasi altro oggetto che richieda una simulazione realistica del comportamento dei tessuti[1]. In questo progetto, il componente Cloth riveste un ruolo central,poiché il compito finale prevede che l'indumento manipolato dal robot presenti una fisica il più possibile realistica, simulando accuratamente il comportamento del tessuto. A tal fine, sono stati forniti tre file principali relativi a una maglietta: un file **OBJ**, un file **FBX** e un fil **Collada**. L'analisi di questi file ha permesso di ottenere informazioni utili riguardo al formato da utilizzare e alle caratteristiche della mesh per applicare efficacemente il componente Cloth.

3.4.1 Compatibilità dei file con Gazebo e Unity

Il formato **.obj** è stato testato con successo sia in Unity che in Gazebo, consentendo l'importazione corretta del modello della t-shirt in entrambi i software. Tuttavia, durante le simulazioni in Unity, l'applicazione del componente Cloth alla t-shirt ha evidenziato alcune

difficoltà: il computer ha riscontrato problemi nel gestire correttamente i comandi impostati e, una volta avviata la simulazione, la t-shirt appariva deformata e non realistica, come è possibile vedere nella figura 3.1.

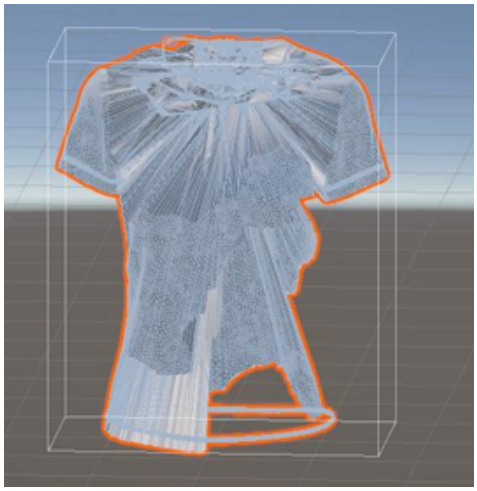

Figure 3.1: *Errore nell'applicazione del Cloth di Unity*

Per risolvere questo problema, l'oggetto è stato importato nel software Blender, dove è stato utilizzato il modificatore Decimate per ridurre il numero di poligoni della mesh. Diminuendo il parametro "Ratio" a un valore che ha portato la mesh a circa 5000 poligoni, è stato possibile reimportare l'oggetto in Unity e applicare correttamente il componente Cloth, ottenendo una simulazione più realistica.

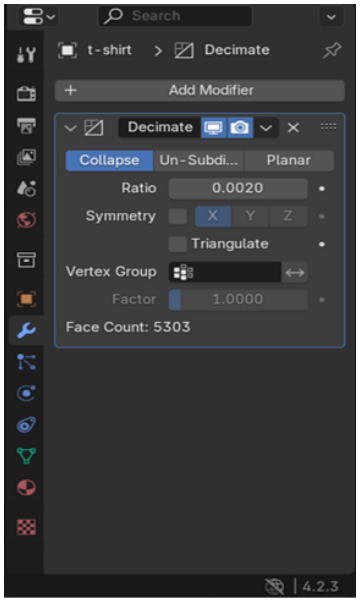

Figure 3.2: *Modifica della Mesh nel software Blender*

Il formato **.fbx** non è supportato da Gazebo; tuttavia, è compatibile con Unity e può

essere convertito in altri formati, come .obj o .dae (Collada), utilizzando Blender. Seguendo lo stesso procedimento adottato per il file .obj, quindi riducendo il numero di poligoni delle mesh tramite il modificatore Decimate in Blender, la simulazione in Unity è stata eseguita correttamente.

Il formato **.dae** (Collada) è supportato da Gazebo e risulta essere ampiamente utilizzato in questo contesto. Importando i file .obj della t-shirt e della scarpa, nonché il file .fbx della t-shirt in Blender, è stato possibile esportarli nel formato .dae. Questa operazione ha permesso di caricare correttamente i vari modelli in Gazebo, facilitando la simulazione degli indumenti nel simulatore.

In sintesi, l'utilizzo combinato di Blender per l'ottimizzazione delle mesh e la conversione dei formati, insieme all'applicazione del componente Cloth in Unity, ha consentito di ottenere simulazioni più realistiche del comportamento dei tessuti, migliorando l'interazione tra il robot e gli indumenti nel contesto del progetto.

3.4.2 Componente Cloth e relativo Manuale

Tra la documentazione di Unity è possibile trovare anche quella relativa al componente Cloth e in seguito verranno riassunti i passaggi fondamentali per utilizzarlo al meglio.

Per implementare una simulazione di tessuto in Unity, è necessario aggiungere il componente Cloth a un oggetto mesh. Ecco i passaggi fondamentali:

- **Preparazione della Mesh**: è importante assicurarsi che l'oggetto a cui si desidera applicare il tessuto abbia una mesh adeguata, e quindi, come visto in precedenza, anche un numero adeguato di poligoni della mesh.
- **Aggiunta del Componente**: una volta importato il file su Unity è sufficiente selezionare l'oggetto nella gerarchia di Unity e, nel pannello Inspector, cliccare su "Add Component", selezionare "Physics" e poi "Cloth" dalla lista dei componenti disponibili.
- **Configurazione**: Una volta aggiunto il componente, sarà possibile modificare vari parametri per ottenere una simulazione personalizzata e accurata.

Affinchè la maglietta abbia una movimentazione guidata dalla fisica, deve chiaramente avere una parte della maglietta fissata e che quindi non è soggetta alla gravità. Questi su Unity vengono chiamati "Cloth Constraint" e quindi ""vincoli del Cloth". Di seguito una descrizione dei passaggi per applicarli correttamente alla maglietta:

1. sul component Cloth si clicca su "Edit Cloth Constraint" come possiamo vedere dalla figura 3.3

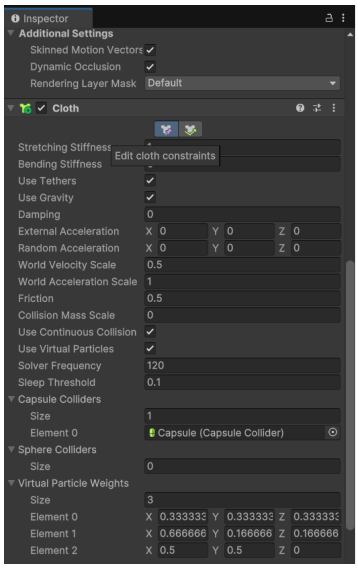

Figure 3.3: *Schermata Unity per l'applicazione dei vincoli*

2. successivamente si aprirà una schermata che permetterà di selezionare (tramite "Select") oppure colorare (tramite "Paint") le parti che saranno vincolate e quindi rimarranno maggiormente rigide. E' importante specificare una distanza minima nel parametro "Max Distance", in figura 3.4 è stato assegnato il valore 0.2 ed ha permesso un buon risultato.

Figure 3.4: *Schermata Unity per la selezione delle parti della mesh a cui applicare i vincoli*

Infine una descrizione delle principali proprietà del componente Cloth e di come sono state utilizzate nell'implementazione del tessuto realistico della maglietta:

- **Stretching Stiffness**: Determina la resistenza del tessuto all'allungamento. Valori più alti rendono il tessuto meno incline a estendersi.
- **Bending Stiffness**: Controlla la rigidità alla flessione del tessuto. Un valore elevato riduce la capacità del tessuto di piegarsi.
- **Use Tethers**: Applica vincoli che aiutano a prevenire che le particelle mobili del tessuto si allontanino troppo da quelle fisse, riducendo l'eccessiva elasticità.
- **Use Gravity**: Indica se la gravità deve influenzare il tessuto.
- **Damping**: Coefficiente che determina quanto velocemente il movimento del tessuto si smorza nel tempo.
- **External Acceleration**: Applica un'accelerazione costante esterna al tessuto, utile per simulare effetti come il vento.
- **Random Acceleration**: Introduce un'accelerazione casuale al tessuto, aggiungendo variazioni imprevedibili nel movimento.
- **World Velocity Scale**: Determina quanto il movimento in spazio globale dell'oggetto influisce sui vertici del tessuto.
- **World Acceleration Scale**: Controlla l'influenza dell'accelerazione globale dell'oggetto sui vertici del tessuto.
- **Friction**: Imposta il coefficiente di attrito del tessuto durante le collisioni.
- **Collision Mass Scale**: Determina l'incremento di massa delle particelle durante le collisioni.
- **Use Continuous Collision**: Abilita la collisione continua per migliorare la stabilità delle interazioni.
- **Use Virtual Particles**: Aggiunge particelle virtuali per migliorare la stabilità delle collisioni.
- **Solver Frequency**: Specifica il numero di iterazioni del solver per secondo, influenzando la precisione della simulazione.
- **Sleep Threshold**: Definisce la soglia sotto la quale il tessuto entra in stato di "sonno", interrompendo la simulazione fino a nuove interazioni.
- **Capsule Colliders**: Array di collisori a capsula con cui il tessuto può interagire.
- **Sphere Colliders**: Array di coppie di collisori sferici con cui il tessuto può interagire.

3.4.3 Componente Cloth e collisioni con altri materiali

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] “Manual cloth unity.” https://docs.unity3d.com/Manual/class-Cloth.html.