POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Informatica

Tesi di Laurea

Generazione automatica di animazioni 3D da contenuti multimodali

Relatori
Prof. Fabrizio Lamberti

Candidato
Luca Macis

Correlatori
Dr. Alberto Cannavò
Prof. Valentina Gatteschi

Anno Accademico 2020-2021

Sommario

Spesso, la comprensione di informazioni testuali può essere migliorata attraverso adeguati contenuti visivi, quali immagini e video. Un caso d'uso può essere rappresentato dai manuali utente, nei quali le istruzioni testuali sono spesso accompagnate da immagini esemplificative. Contenuti video, come pure, applicazioni grafiche (interattive o meno), ad esempio in realtà virtuale o aumentata, potrebbero aumentare ulteriormente l'efficacia di tale materiale, mostrando in maniera animata lo svolgimento dei passi richiesti. Tuttavia, la generazione di questi contenuti è, ad oggi, un processo manuale e lento, poiché ogni contenuto deve essere specifico, ad esempio nel caso d'uso menzionato, per la particolare procedura e la relativa sequenza di istruzioni. Lo strumento realizzato nel presente lavoro di tesi mira a generare in modo automatico un video in computer grafica, utilizzando in ingresso una sorgente multi-modale, ovvero un testo in linguaggio naturale e una serie di immagini, provenienti da un documento collegato, ed una libreria di modelli 3D animati. Lo strumento utilizza una rete neurale profonda (Mask R-CNN) per il riconoscimento dei componenti delle immagini ed utilizza le caratteristiche geometriche di tali componenti per dedurne posizione e rotazione. Per l'analisi del testo utilizza invece una soluzione di Natural Language Processing (Scene Graph Parser) con il quale estrarre i componenti menzionati nel documento, e collegarli, attraverso analisi semantiche, a quelli rilevati nelle immagini e presenti nella libreria di modelli. Lo strumento è stato validato nel caso d'uso considerato, con diversi manuali di stampanti. I risultati sono promettenti, sebbene siano stati rilevati limiti nell'analisi di frasi complesse e nell'individuazione di determinati componenti nelle immagini. Come detto, lo strumento è stato realizzato con l'obiettivo di generare velocemente ed in maniera automatica un video che possa integrare o sostituire, ad esempio, un manuale di istruzioni. Esso potrebbe essere adoperato anche in altri contesti, ad esempio per l'animazione di sceneggiature cinematografiche. Lavori futuri potrebbero includere la generalizzazione del contenuto da animare e l'utilizzo, durante la fase di analisi del testo, di una fase di semplificazione delle frasi in ingresso in modo da migliorare le capacità di riconoscimento.

Indice

Elenco delle tabelle

Elenco delle figure

Parte I

Prima Parte

Capitolo 1

Introduzione

Un sistema di authoring è un sistema che aiuta nello sviluppo di contenuti digitali come materiale multimediale interattivo. Questo tipo di strumento permette agli utenti, non programmatori, di creare facilmente applicazioni per la manipolazione di oggetti multimediali.
Tra questi sistemi, quelli riguardanti la generazione di animazioni a partire da testo in linguaggio naturale, sono ad oggi ambito di numerose ricerche.
Questo tipo di sistema trova applicazione in numerose aree tra cui l'intrattenimento, la didattica e la formazione del personale per lavori specializzati e non. Alcuni esempi di utilizzo di questo tipo di sistema sono:

- La generazione di un video animato tramite modelli 3D rappresentante una sceneggiatura cinematografica ricevuta in ingresso. Lo scopo dell'utilizzo del sistema consiste nell'avere un'anteprima virtuale generata in automatico ed evitare di perdere tempo e risorse tramite prove in ambienti reali o tramite la generazione manuale di un prototipo virtuale.
- La realizzazione di video didattici tramite l'animazione di oggetti 3D, in cui tramite un testo scritto in linguaggio naturale si ha la possibilità di riprodurre visivamente, tramite il video, il concetto espresso dal testo. I video così generati hanno lo scopo di integrare il testo aumentando la comprensione delle informazioni espresse.
- La creazione di video formativi animati di modelli 3D a partire da testo di manuali di istruzione o manuali di formazione del personale per operazioni tecniche. Il video generato ha lo scopo di illustrare visivamente la procedura da adoperare in modo da aumentare l'efficacia del testo per quanto riguarda la formazione dell'utente.

Lo scopo di questo tipo di sistemi è quello di poter essere utilizzati anche da personale non specializzato, al fine di realizzare un video animato a partire da un

testo in linguaggio naturale che, integrato con il testo stesso, li possa aiutare nella comprensione dei concetti espressi in forma testuale tramite una rappresentazione visiva. Oltre all'incremento della comprensibilità, un ulteriore vantaggio, è quello di velocizzare le fasi di progettazione o di testing di prodotti che richiedano la realizzazione di un video, come la valutazione di una sceneggiatura o la procedura di assemblaggio di una stampante.
Il sistema descritto in questo lavoro di tesi, è stato sviluppato con l'idea di facilitare la comprensione di un manuale di istruzioni, in questo caso di stampanti, tramite la generazione automatica di un video animato con modelli 3D, da integrare o sostituire al testo. In questi manuali il testo e le immagini che descrivono la procedura da eseguire spesso non bastano per la sua comprensione e la sua successiva esecuzione. La realizzazione di un video dimostrativo è quindi un utile strumento, che in aggiunta al testo e alle immagini, può aumentare la comprensibilità della procedura da eseguire. I benefici dell'utilizzo di questo sistema in aggiunta al manuale di istruzioni sono molteplici:

- Aumenta la comprensibilità della procedura di montaggio.
- Aumenta le informazioni che l'utente memorizza, tramite l'utilizzo della memoria visiva, rispetto alle informazioni reperibili e memorizzabili tramite il solo testo in linguaggio naturale.
- Velocizza la fase di realizzazione di un video formativo, riguardante la procedura di assemblaggio o utilizzo descritto nel testo presente all'interno del manuale, tramite l'automatizzazione della generazione del video animato di modelli 3D.

Il sistema descritto in questo lavoro di testi utilizza il testo presente all'interno dei manuali di istruzioni per la selezione dell'animazione degli oggetti 3D da mostrare in relazione ad ogni specifico step della procedura. Il sistema, per la selezione dell'animazione corretta, reperisce tutte le animazioni e i modelli 3D riguardanti la stampante da animare a cui il manuale di istruzione si riferisce presenti in una libreria pre caricata, per effettuare un confronto tra l'azione e il componente citato nel manuale con quelli presenti all'interno della libreria.
Il sistema inoltre utilizza le immagini presenti all'interno del manuale di istruzioni per la generazione automatica delle animazioni dei modelli 3D e per il posizionamento automatico della telecamera rispetto all'inquadratura utilizzata all'interno delle immagini. Per posizionare in modo automatico la telecamera, il sistema identifica i vertici rappresentanti gli angoli della stampante raffigurata nelle immagini presenti sul manuale di istruzioni, e tramite la corrispondenza con i punti rappresentanti gli angoli del modello 3D della stampante, calcola un'approssimazione della posizione che la telecamera deve assumere per ottenere la stessa inquadratura. Questa posizione viene utilizzata dal sistema per selezionare la camera virtuale

opportuna all'interno dell'insieme di camere virtuali posizionate precedentemente nella scena.
Per la generazione automatica delle animazioni dei modelli 3D, il sistema identifica i vertici rappresentanti gli angoli dei componenti della stampante, raffigurati all'interno delle immagini presenti sul manuale di istruzioni, e tramite una conversione in punti 3D all'interno dell'ambiente virtuale e un confronto con i punti 3D, il modello del componente da animare assume la sua posizione di default. Inoltre il sistema riesce a calcolare la posizione e la rotazione che devono essere applicate al modello per generare l'animazione mostrata nell'immagine corrispondente.
I vantaggi del sistema descritto in questo lavoro di tesi rispetto ai sistemi già presenti per la generazione delle animazioni, che utilizzano unicamente il testo come sorgente, riguardano i benefici derivanti dall'elaborazione delle immagini presenti all'interno del manuale di istruzioni, ovvero:

- La possibilità di generare in automatico alcune animazioni di modelli 3D a partire dalle immagini. Il fine è quello di diminuire la quantità di animazioni memorizzate all'interno della libreria pre caricata, contenente le animazioni di default.

- Il posizionamento semi-automatico della camera virtuale all'interno dell'ambiente virtuale allo scopo di realizzare un video avente la stessa inquadratura delle immagini, presenti all'interno del manuale di istruzioni, e quindi più facilmente confrontabile con esse.

Il secondo capitolo di questo lavoro di tesi, tratta lo stato dell'arte dei sistemi odierni, i quali a partire dal testo o dalle immagini generano un video animato tramite modelli 3D. Il capitolo parla dell'evoluzione, nel tempo, dei sistemi di questo tipo, trattando nello specifico dei limiti principali riscontrati nell'utilizzo dei sistemi e delle soluzioni realizzate per ovviare ad alcuni di essi.
Vengono accennati i limiti che sono rimasti anche nei sistemi odierni e vengono descritti le cause che hanno determinato la loro permanenza all'interno dei sistemi.
Infine vengono descritte le scelte implementative attuate all'interno del sistema descritto nel lavoro di tesi proposto, per risolvere o attenuare questi problemi.
All'interno del terzo capitolo vengono introdotti i framework e le librerie utilizzate per la realizzazione del sistema descritto in questo lavoro di tesi. Nello specifico vengono descritte brevemente le caratteristiche principali e il funzionamento ad alto livello dello strumento scelto, insieme alla spiegazione riguardante il motivo della loro scelta a discapito di altri strumenti e librerie presenti online. Nel quarto capitolo viene descritta nei dettagli l'implementazione della fase di analisi del testo in linguaggio naturale all'interno del sistema trattato in questo lavoro di tesi, con dettagli riguardanti la fase di filtraggio e le scelte implementative.

In aggiunta viene descritta la pipeline utilizzata per l'analisi del testo ricevuto in ingresso e ogni fase della proceduta di estrapolazione delle informazioni, riguardanti l'istruzione, presenti all'interno del testo. Queste informazioni verranno poi utilizzate per la scelta dell'animazione all'interno della libreria, ed infine viene descritta la fase di ricerca dei sinonimi per le informazioni estrapolate dal teso.
Il quinto capitolo descrive la fase di analisi delle immagini presente all'interno del sistema sviluppato per questo lavoro di tesi. Viene descritta nei dettagli la procedura e le scelte adoperate per l'addestramento del modello, dei pesi per il riconoscimento dei componenti di una stampante all'interno di un'immagine, ponendo enfasi sulle scelte e le considerazioni adoperate per la configurazione del modello, sulla fase di annotazione dei componenti nelle immagini e sulla fase di generazione dei pesi del modello.
Successivamente viene trattata la fase di rilevamento dei componenti, dandone una definizione dettagliata e descrivendo come questa venga svolta all'interno del sistema descritto in questo lavoro di tesi. Infine viene descritta in modo approfondito la procedura per l'identificazione dei vertici rappresentanti gli angoli di un componente, o della stampante stessa all'interno dell'immagine, così come la procedura per la memorizzazione dei dati raccolti nelle apposite strutture.
Il sesto capitolo è dedicato alla descrizione del processo di generazione del video animato tramite modelli 3D. Il processo si basa sulla selezione delle animazioni all'interno della libreria di animazioni e la generazione automatica delle animazioni dei modelli 3D. All'interno del capitolo troviamo descritto il processo di traduzione dei nomi dei componenti, delle azioni rilevate durante l'analisi del testo e delle immagini da parte del sistema descritto in questo lavoro di tesi, e delle rispettive controparti presenti all'interno della libreria pre caricata contenente i modelli e le animazioni di default. Viene inoltre descritto l'ambiente virtuale utilizzato e il processo di comunicazione adoperato per l'interazione dell'ambiente virtuale con il resto del sistema.
Successivamente viene esposto approfonditamente il processo di calibrazione della camera virtuale e del calcolo della posa. Si conclude infine illustrando la fase di montaggio del video.

Capitolo 2

Stato dell'arte

Questo capitolo, tratta lo studio dello stato dell'arte per quanto riguarda i sistemi di generazione di video animati o immagini statiche realizzate tramite modelli 3D che utilizzino come fonte dalla quale reperire le informazioni, su cui si basa la scelta delle animazioni da riprodurre, un testo in linguaggio naturale. I sistemi qui presentati riguardano diversi contesti applicativi tra cui:

- la generazione di video tramite l'animazione di oggetti 3D, a partire da sceneggiature cinematografiche, ovvero la realizzazione di un prototipo visivo di testi composti da didascalie e dialoghi (1)(2)

- l'automatizzazione del processo di generazione della mappatura tra gesti e parlato, riguardante la lingua dei segni, basato su regole, senza l'intervento di esperti umani (3)

- la creazione di un sistema che consenta agli autori di annotare sceneggiature cinematografiche standard, con eventi, riguardanti l'animazione, rilevanti, come azioni dei personaggi, posizioni della telecamera e sfondi delle scene, per poterli successivamente animare (4)

- la generazione di un video animato tramite modelli 3D di ricette culinarie in forma testuale tramite linguaggio naturale (5)

- la realizzazione di personaggi interattivi che aiutino i bambini con problemi di udito a imparare l'inglese e la lingua dei segni, e che aiutino invece le persone udenti ad imparare la lingua dei segni (6)

Si deduce quindi che il campo applicativo per questo tipo di sistema è ampio e diversificato e può includere vari aspetti e fasi dei ciclo di sviluppo di un progetto.

2.1 Da text-to-scene a text-to-animation

Per iniziare vengono descritti come si sono evoluti i sistemi di conversione tex-to-scene e poi successivamente i sistemi text-to-animation (7).
La comprensione di questa evoluzione pone la base per una comprensione migliore delle problematiche e delle soluzioni che si sono adoperate nel tempo nei vari sistemi di questo tipo. I sistemi inizialmente avevano come scopo quello di realizzare semplici scene statiche in 2D i cui limiti erano la mancanza di animazioni e la necessità di testi con un determinato formato in ingresso.
Esempi di questi sistemi sono NALIG (1993) (8) e PAR (1999) (9) che sono tra i primi sistemi per la traduzione da testo ad animazione, il cui punto debole era la scarsa raccolta di oggetti 3D, l'analisi superficiale del testo e la mancanza di interazione con l'output generato.
Nel corso degli anni molti dei problemi riscontrati in questi primi sistemi sono stati risolti. Il sistema SceneMaker (2010) (10) (11), un sistema collaborativo multi modale sviluppato per la pre visualizzazione di scene di una data sceneggiatura per facilitare il processo di produzione cinematografica, ad esempio utilizza grandi database online da sfruttare nell'analisi del testo il cui scopo è quello di ottenere un analisi più efficiente ed accurata dei database di oggetti 3D più vasti e permette un certo grado di interattività. Oltre a risolvere i problemi pregressi gli ultimi sistemi realizzati hanno implementato alcune nuove caratteristiche come la capacità di analizzare le emozioni e la possibilità di utilizzare testi non formattati o in generale con una flessibilità maggiore sul tipo di formato.
Nel corso degli anni sono stati molti i sistemi che sono stati sviluppati ad esempio il CARSIM (2001) (12), un software di predizione delle prestazioni di un veicolo tramite simulazione in risposta al controllo dell'autista.
Altri sistemi sono lo ScriptViz (2005) (13), un sistema che permette agli utenti di visualizzare le proprie sceneggiature in real time tramite animazioni grafiche, il CONFUCIUS (2005) (14), un sistema di conversione da testo ad animazione multimodale che genera un animazione da una singola frase in ingresso contenente un verbo di azione da cui in oltre prendono spunto altri sistemi come lo SceneMaker (2009) e IVELL (2013) (15) (16), dei sistemi per la realtà virtuale multi-modale che consistono in alcuni agenti conversazionali incorporati con lo scopo di migliorare l'abilità di ascolto e di parlato per utenti non madrelingua inglese.

Tra i sistemi text-to-animation odierni quello sviluppato dalla Disney Research (1), il cui punto di forza consiste nell'avere un'architettura di analisi delle scenografie, si adatta ai diversi formati testuali attraverso una macchina a stati finiti opportunamente pensata e sviluppata.
A differenza di molti altri sistemi infatti quello sviluppato dalla Disney Research si concentra maggiormente sulla fase della semplificazione e dell'analisi del testo, sia

per testi formattati, secondo lo standard delle sceneggiature, con errori di formattazione o non formattati, questo permettere al sistema di avere grande flessibilità in termine di testo in ingresso accettato. I normali sistemi text-to-animation non hanno una fase di semplificazione del testo, spesso questa fase non è necessaria e quando è presente non è molto elaborata, inoltre l'estrazione delle informazioni dal testo in linguaggio naturale, in questi casi, funziona bene solo con frasi semplici che hanno un verbo, un oggetto e un soggetto.
Nel sistema sviluppato dalla Disney Research si utilizzano una serie di regole linguistiche per la fase di semplificazione che manipolano la struttura della frase analizzando le dipendenze sintattiche per ottenere in output una frase semplificata.
Il sistema fa uso dell'algoritmo Support Vector Machine (SVM) di NeuralCoref (17), allenato appositamente per segmentare il testo presente nelle sceneggiature con indentazione inconsistente, e della pipeline di animazione del sistema CARDINAL (2) per generare l'animazione.
NeuralCoref è un'estensione della pipeline per spaCy (18) che annota e risolve i cluster di coreferenza tramite una rete neurale. La pipeline di animazione CARDINAL ha un insieme di animazioni pre-baked limitato (52 animazioni). Nel caso in cui un verbo non trovi riscontro tra le animazioni presenti viene eseguita una ricerca dell'animazione più pertinente attraverso una funzione di similarità di word2vec (19), che trova la più simile escludendo gli opposti tramite wordNet (20).
Nella valutazione BLEU (21), un algoritmo di valutazione della qualità del testo che viene tradotto da una macchina da una lingua naturale ad un'altra, il sistema ha ottenuto un punteggio basso in alcune frasi (38%), questo a causa del fatto che il sistema cerca di scomporre le frasi senza ridurre l'informazione al suo interno. Secondo una valutazione basata sull'esperienza soggettiva degli utenti invece il sistema ha ottenuto risultati accettabili (il 68% degli utenti ha valutato positivamente l'animazione generata). Un altro sistema di authoring è quello utilizzato per la generazione di animazioni da istruzioni di ricette culinarie (5) sviluppato per lavorare con frasi dalla struttura semplice (oggetto e verbo), ovvero con delle istruzioni semplici.
Le animazioni vengono generate tramite delle regole di scomposizione di un verbo in azioni più semplici che devono essere preparate ed inserite manualmente in anticipo dall'utente, questo permette al sistema di non necessitare di grandi database di animazioni limitandosi a contenere unicamente le animazioni dei verbi più semplici.
Sempre secondo una valutazione che si basa sull'esperienza soggettiva degli utenti coinvolti nella sperimentazione, il sistema ha ottenuto il 75% di pareri positivi riguardante la bontà dell'animazione in base al testo fornito.
A differenza del sistema sviluppato dalla Disney Research (1), il sistema per l'automatizzazione della mappatura tra gesti e parlato (2), sfrutta il framework NLP

Core di Stanford (22) per l'analisi del testo e un server esterno per comunicare la tripletta <soggetto, verbo, oggetto>. Per ottenere una visualizzazione temporale della azioni in quest'ultimo sistema si utilizza un'indicizzazione basata sul tempo, che permette di considerare come da simulare contemporaneamente frasi contenenti più azioni al proprio interno, un eccezione viene fatta nel caso in cui all'interno della frase si utilizzino parole per la "temporizzazione esplicita" come "dopo" e "prima".
Come nel sistema della Disney Research, quest'ultimo sistema sfrutta word2vec per la ricerca di parole simili al fine di ridurre la dimensione del database di azioni ed oggetti da caricare in anticipo. Per la previsualizzazione delle animazioni invece il quest'ultimo sistema sfrutta il framework ADAPT(23) per la generazione dell'albero dei componenti necessari all'interno della scena da visualizzare. Il sistema permette due tipi di visualizzazione 2D e 3D, modificabili tramite la modifica dell'albero generato dal framework ADAPT, i due tipi di visualizzazione sono utili rispettivamente per la fase di progettazione e per ottenere un prototipo della scena reale. Il sistema (2) è stato confrontato con un altro sistema open source per la scrittura di sceneggiature (24), il metro di confronto consisteva nella valutazione delle risposte fornite dagli utenti su una storia osservata attraverso i due sistemi. Il sistema Carsim (2) è risultato, tramite questo confronto, come più semplice da utilizzare ed è risultato più facile reperire le informazioni al suo interno.

Nei sistemi odierni la libreria più utilizzata per l'analisi del testo è spaCy unita alla pipeline NeuralCoref per l'espansione della pipeline della libreria.
Per ovviare al problema del numero di animazioni generabili tutti i sistemi hanno sfruttato wordvec per la ricerca di verbi simili, un utile aggiunta utilizzata in alcuni sistemi è wordNet, utilizzato allo scopo di escludere i verbi opposti. Un altro sistema odierno che tratta questa tematica è quello che genera un animazione di un piccolo gruppo di personaggi nell'ambiente tramite l'utilizzo di folle ambientali (25), il sistema genera l'animazione dei gruppi di personaggi/oggetti tramite il linguaggio naturale analizzato tramite Stanford Core NLP, il quale mostra un approccio diverso al problema. In questo caso le informazioni estratte dal testo vengono utilizzate per la generazione di nodi situazionali, e le azioni vengono salvate come coppia azione – azione', dove azione' corrisponde ad una azione passiva o ad una azione che descrive un'emozione. A queste azioni il quest'ultimo sistema associa dei "modi", corrispondenti alle informazioni spaziali riguardanti il singolo personaggio, all'interno di un movimento di gruppo, e definisce il comportamento di tutti i personaggi rispetto al comportamento del personaggio in esame. La fase di animazione di questo sistema sfrutta Unity, la quale impiega un'interfaccia interattiva per consentire all'utente di ripristinare l'animazione e modificare, cancellare o aggiungere nodi situazionali, portando ad una modifica dell'animazione mostrata. I valori difficilmente estraibili dal testo, come tempo, velocità di animazione,

inquadratura della telecamera, vengono invece inseriti manualmente. Anche il sistema descritto in (26) ha come scopo l'animazione dei piccoli gruppi (25) tramite la generazione, la gestione e la simulazione di una folla a partire dal testo scritto in linguaggio naturale. Per ogni modello il sistema crea una mappa di scena con etichette contenenti i campi necessari al sistema per la gestione della folla come la popolazione, gli oggetti e gli ostacoli. La fase di elaborazione del linguaggio naturale viene eseguita da Stanford Core NLP per l'analisi delle dipendenze. Il sistema utilizza in aggiunta al framework la memoria a corto-lungo termine bidirezionale (Long Short Term Memory - LSTM), una architettura di rete ricorrente neurale artificiale che a differenza delle reti neurali feedfoward standard ha connessioni feedback, questa architettura ha la capacità quindi di elaborare intere sequenze di dati invece di elaborarne una sola. In questo sistema viene utilizzato Unity per la generazione dell'animazione, word2vec per l'elaborazione dei sinonimi delle azioni estrapolate, wordToEye per la conversione da testo ad immagine statica ed infine CONFUCIUS per la conversione da scena statica a dinamica.
Per quanto riguarda i sistemi che utilizzano l'elaborazione di immagini o video per la generazione di animazioni corrispondenti, tra i sistemi odierni, quello trattato in (27) è un sistema con lo scopo di generare una mappatura tra gesti e informazioni estrapolate da un video.
In fase di esecuzione l'incorporamento di parole viene utilizzato al fine di trovare regole semantiche significative ed accurate. La fase di mappatura può essere eseguita manualmente da parte dell'utente se la generazione dei gesti è basata su regole o automaticamente tramite un modello addestrato se la generazione dei gesti e basata su dati.
La generazione automatica ha come problemi la dipendenza dalla sorgente, le laboriose annotazioni manuali, la mancanza di sorgenti di dati e i problemi legati al deep learning.
La generazione manuale per contro, ha come problemi la poca flessibilità, la possibilità di avere dati ripetuti e la laboriosità del processo.
La fase di animazione, sempre elaborata su Unity, cerca il token precedentemente estratto tra le regole generate per una riproduzione in sequenza e sfrutta inoltre Glove (28) e word2vec per la ricerca di sinonimi.
L'approccio del sistema TakeToons (4) è incentrato sulla possibilità da parte degli utenti di aggiungere annotazioni rilevanti per la generazione della animazione (es. azioni dei personaggi, posizione della telecamera, sfondi) alla normale sceneggiatura. Lo scopo del sistema è quello di convogliare in un'unica rappresentazione i dialoghi, i cambi di scena, il movimento dei personaggi e le performance vocali in modo che i cambiamenti effettuati in una di questi componenti non comportino una modifica nei componenti successivi con una risultante perdita di tempo.
Il sistema supporta la ripetizione e sovrapposizione di animazioni come la collaborazione nella sua realizzazione. In questo caso non è presente alcuna elaborazione

sintattica della sceneggiatura il sistema traduce le performance dal vivo degli attori in azioni dei personaggi virtuali.
Il sistema SignAR (6) consiste in un'applicazione di realtà aumentata per fornire una mappatura "invisibile" della lingua dei segni di ogni parola all'interno di un testo visualizzato tramite camera.
Anche in questo caso il sistema è rilevante in quanto il suo scopo è quello di analizzare un'immagine al fine di estrapolare informazioni, in questo caso raffigurazioni o parole, e di generare un animazione corrispondente. Le parole e le raffigurazioni sono memorizzate in precedenza all'interno di un database, per il loro rilevamento tramite l'immagine catturata da una camera viene utilizzato Vuforia SDK e le informazioni così estrapolate vengono elaborate ulteriormente e poi sfruttate per la generazione dell'animazione tramite Unity. L'ultimo sistema che cito in questo capitolo, il sistema descritto in (29), non è odierno ma ha comunque dati interessanti che hanno contribuito allo studio dello stato dell'arte per la realizzazione del sistema descritto in questo lavoro di tesi. Infatti il sistema trattato in (29) riceve in input una fonte multi modale, ovvero immagini, testo ed audio, allo scopo di indicizzazione all'interno di un film.
L'utilizzo di input eterogeneo porta come conseguenza a dei problemi, ad esempio l'analisi delle immagini restituisce una quantità grande di informazioni semantiche di basso livello (colore, consistenza, movimento,etc), mentre le sinossi contengono informazioni in quantità ridotta, ma con un contenuto semantico alto.
Per estrarre possibili collegamenti tra le parole e l'attività osservata nel film viene utilizzato un algoritmo di classificazione automatico, l'algoritmo J48 disponibile con Weka (30), questo algoritmo misura il guadagno di informazione, ovvero la differenza di entropia, e considera valida una classificazione se il termine estratto corrisponde al descrittore di attività simbolica previsto fornito dall'analisi dell'immagine. Come in altri sistemi visti precedentemente, l'algoritmo J48, utilizza word2vec e wordNet per la ricerca di sinonimi e contrari.
Si può dedurre che alcuni dei sistemi qui citati prima di analizzare il testo eseguono una semplificazione tramite delle regole linguistiche tramite ad esempio YATS (31), in modo da migliorare l'accuratezza dei risultati restituiti dalla fase dell'analisi del testo, eseguita solitamente dal framework NLP Core Stanford o la libreria SpaCy 2.1+.
I sistemi a questo punto raffinano i risultati ottenuti tramite la ricerca dei sinonimi e dei contrari tramite word2vec e wordNet al fine di ridurre la dimensione dei database contenente le azioni e gli oggetti da animare all'interno della scena. Le limitazioni che rimangono in questo tipo di sistemi sono:

- Mancanza di animazioni facciali
- Mancanza di informazioni a causa di oggetti ed animazioni che non hanno riscontro nel database

- Semplificazione a volte troppo aggressiva, che elimina informazioni utili alla generazione dell'animazione
- Limiti nella comprensione di alcune componenti grammaticali, come avverbi e aggettivi, ad esempio "improvvisamente", o vincoli temporali
- Animazioni non rappresentative dell'azione fornita
- Informazioni utili alla generazione dell'animazione, scartate o deliberatamente ignorate
- Numero limitato di animazioni presenti nel database

Le animazioni facciali rimangono un problema in quanto complesse da realizzare ma di interesse secondario nella visualizzazione di una animazione, infatti l'utente generalmente focalizza la propria attenzione sui macro comportamenti piuttosto che sulle espressioni facciali.
Il numero di animazioni limitato all'interno del database è causato dall'incapacità di generarle in automatico, questo porta ad un ulteriore problema, ovvero l'incapacità di rappresentare ogni azione descritta. Capita infatti di avere un azione non presente nel database e attraverso la ricerca dei sinonimi si trova un sostituto che non rappresenta adeguatamente l'azione richiesta.
La semplificazione del testo, quando effettuata, spesso ha come problema secondario il fatto che delle informazioni rilevanti vengano eliminate o decontestualizzate, infatti in alcuni sistemi di semplificazione la frase viene segmentata in frasi più piccole e semplici ma completamente indipendenti dalle altre, questo può portare ad avere un interpretazione sbagliata delle azioni.

Nel sistema proposto in questo lavoro di tesi, vengono utilizzate le immagini al fine di generare animazioni in automatico, questo approccio non risolve completamente il problema delle animazioni non rappresentative o del limite delle animazioni presenti all'interno del database ma lo attenua aumentando il numero delle animazioni all'interno del database e generando animazioni plausibilmente più pertinenti all'azione descritta in maniera automatica.
Le immagini vengono utilizzate per estrapolare informazioni non ottenibili tramite la sola analisi del testo e che aiutano a generare animazioni più apprezzabili e più comprensibili. Tra queste informazioni c'è la posizione della camera all'interno della scena, le direzioni di movimento e l'ampiezza di rotazione dei componenti. Questo aumenta il numero di informazioni reperite dai nostri dati in ingresso generando potenzialmente un animazione più accurata e comprensibile.
Nel sistema qui descritto non viene attuata alcuna fase di semplificazione del testo, le informazioni vengono filtrate al fine di ottenere solo quelle effettivamente utili alla generazione delle animazioni mentre quelle utili alla comprensione del contesto

che non influiscono sull'animazione finale vengono deliberatamente memorizzate per essere utilizzate in caso di future versioni del sistema. Durante l'elaborazione del testo oltre alle regole già fornite dal framework scene graph parser ne sono state aggiunte ulteriori indispensabili per la gestione di testi provenienti da manuali di istruzioni.
Questi tipi di manuali, come i manuali di istruzioni riguardanti le stampanti, possiedono una sintassi che si differenzia dai manuali tecnici informatici, dai giornali o da testi di discussioni presenti online su forum, testi su cui si è basata la realizzazione del framework scene graph parser. In questo sistema vengono quindi aggiunte regole atte alla gestione delle clausole che cambiano il significato della azione e regole atte a riconoscere e memorizzare le istruzioni espresse in forma imperativa.

2.2 Liberia NLP

Nel campo della ricerca vi è molta confusione per quanto riguarda la scelta della libreria per l'elaborazione del linguaggio naturale (NLP – Natural Language Processing) da utilizzare. A questo scopo alcuni studi si sono soffermati su questo problema al fine di redigere delle linee guida prendendo in considerazione le odierne librerie allo stato dell'arte e determinati fonti di documentazione come sorgente. Gli esperimenti effettuati nello studio trattato in (32) hanno reso evidente che l'accuratezza ottenuta dalle singole librerie è relativa al tipo di testo che il sistema deve elaborare, i dati forniti dagli sviluppatori sull'accuratezza della loro libreria, nella maggior parte dei casi, è relativa a testi non tecnici ovvero simili a quelli del wall street journal,ne consegue che l'accuratezza dei testi di tipi differente come manuali di istruzioni e sceneggiature o in generale testi contenenti tecnicismi non sia la stessa ma inferiore.
La causa del calo dell'accuratezza ottenuta nella fase di elaborazione del testo può essere ricondotta a testi incompleti, contenenti "tecnicismi del mestiere" o scritti da non madre lingua, di conseguenza contenenti errori grammaticali al proprio interno. La scelta della libreria di NLP è quindi un passo di notevole importanza spesso sottovalutato che può portare a drastici cali di prestazioni da parte del sistema che l'utilizza. Nella sperimentazione eseguita nello studio trattato in (32) le librerie tenute in considerazione sono quelle che ad oggi continuano ad essere le librerie NLP allo stato dell'arte, ovvero:

- Google's syntaxNet (33) : La libreria NLP più recente, tra quelle pubblicamente disponibili, definito come "l'analizzatore open source più accurato al mondo"
- Stanford Core NLP (22) :Un framework scritto in Java con l'obbiettivo di facilitare l'applicazione di strumenti di analisi linguistica a una parte di testo.

Il più utilizzato tra i ricercatori di ingegneria del software basato su NLP fino ad oggi, può essere integrato con strumenti di analisi grammaticale e supporta diverse lingue tra cui l'italiano e l'inglese, possiede inoltre una serie di interfacce disponibili per i diversi linguaggi di programmazione.

- NTLK (34) : Una delle librerie storiche e più sperimentate che dispone di diverse interfacce per i diversi linguaggi di programmazione, ha a sua disposizione numerose risorse ed è allenato con una vasta gamma di corpora e risorse lessicali.
- SpaCy (18) : La libreria che ad oggi sta acquistando molta popolarità, scritta in Python e Cython, risulta nei dati forniti dagli sviluppatori essere più veloce delle altre librerie allo stato dell'arte.

I testi utilizzati nella sperimentazioni sono stati reperiti da Stack Overflow, File ReadMe da GitHub e Documentazione API Java, ovvero testi contenenti tecnicismi e non sempre scritti da madre lingua a cui è stato filtrato il codice per semplicità. Lo studio ha mostrato che in base al testo utilizzato la percentuale di corrispondenza dei PoS e token individuati tra le diverse librerie variava dal 89% al 94%, inoltre eseguendo una elaborazione manuale dei testi e confrontandola con l'elaborazione restituita dalle librerie, la libreria NTLK ha ottenuto una percentuale di corrispondenza tra il 98% e il 99% per quanto riguarda l'identificazione dei Token in tutti i testi, le altre librerie hanno comunque ottenuto una corrispondenza sopra il 94% con tutti i testi. Per quanto riguarda l'identificazione di PoS la libreria spaCy ha ottenuto i migliori risultati in tutti i testi con una percentuale di corrispondenza che varia tra il 78% e il 89%. Lo studio pone in luce il fatto che nonostante il largo utilizzo della libreria Stanford Core NLP questa non sia la migliore in termini di identificazione di token o Pos.
La libreria spaCy risulta quindi, con i testi presi per questa sperimentazione e con i testi utilizzati dagli sviluppatori, avere in media delle ottime prestazioni, dato che viene confermato dall'andamento della scelta delle librerie NLP all'interno del mondo della ricerca e del suo sempre maggiore utilizzo nei sistemi odierni.

Capitolo 3

FrameWork e librerie utilizzate

Questo capitolo tratta i framework e le librerie utilizzate dal sistema fornendo una breve descrizione sul loro utilizzo e funzionamento e elencandone le principali caratteristiche. Viene inoltre fornita la motivazione che ha portato alla scelta di un particolare framework o di una particolare libreria a discapito di altre trattate nel capitolo riguardante lo stato dell'arte.

3.1 Scene Graph Parser

SceneGraphParser (35) (36) è un toolkit realizzato in python che si ispira al Stanford Scene Graph Parser, il suo scopo è quello di realizzare grafici di scena, ovvero una rappresentazione semantica basata sui contenuti presenti all'interno dell'immagine o di un testo.
Questo framework analizza gli attributi e le relazioni tra gli oggetti , presenti in un testo, tramite l'elaborazione del linguaggio naturale. Il framework ha la capacità di generare un grafo della scena di un immagine, per fare ciò richiede una descrizione in linguaggio naturale dell'immagine, fornita direttamente dall'utente o ricavata dall'immagine stessa tramite opportune librerie o framework, l'analisi dell'immagine e la sua conversione nel grado di scena risultante si basa sull'analisi delle dipendenze.
Per quanto riguarda l'analisi del linguaggio naturale, il framework, analizza le frasi presenti all'interno del testo e le converte in un grafo di scena dove i nodi sono i sostantivi, ad esempio sostantivi diretti o passivi e i collegamenti tra nodi sono le relazioni che legano un sostantivo ad un altro.

3.1.1 Caratteristiche

Il grafo viene rappresentato tramite l'utilizzo di liste e dizionari (dict). Gli sviluppatori hanno preferito questa struttura di dati, nonostante i problemi legati ad esse, in quanto il framework è ancora in fase di sviluppo e quindi le API sono soggette a modifiche. Quella qui utilizzata è una versione stabile ufficialmente pubblicata. Un'altra caratteristica preponderante del framework consiste nella sua semplicità di integrazione con qualsiasi progetto sviluppato in python, difatti non è obbligatorio attuare ulteriori elaborazioni ai risultati restituiti dall'analizzatore. I dizionari vengono utilizzati per raccogliere i dati riguardanti entità, come soggetti ed oggetti, e relazioni che di fatto rappresentano le entità e le relazioni all'interno del codice, mentre le liste vengono utilizzate come contenitori per contenere tutte le entità e le relazioni che vengono menzionate all'interno del testo (3.1).
L'analizzatore è ispirato allo Scene Graph parser di Stanford ma non è identico, lo scopo degli sviluppatori è stato quello di replicarne solo parte delle funzionalità, concentrandosi sulle prestazioni empiriche ottenibili. Rispetto all'analizzatore di Stanford sono presenti ottimizzazioni euristiche minori e regole aggiuntive per la gestione di :

- nomi composti
- preposizioni frasali come "davanti a"
- verbi frasali
- rilevamento di nomi a livello di scena (ad es. aeroporto)

di contro, non sono presenti alcune funzionalità dell'altro analizzatore, quali:

- modificatori quantificazionali come "un sacco di"
- risoluzione dei pronomi
- sostantivi plurali

3.1.2 Motivo della scelta

A differenza della versione di Stanford (37) questo toolkit è realizzato in python e dispone di una interfaccia dal facile utilizzo, ciò ha reso questo framework un'ottima scelta come framework di partenza per la fase di analisi del linguaggio naturale. La versione utilizzata all'interno del nostro sistema è stata modificata per adattarsi meglio al contesto ed al tipo di dato ricevuto in ingresso, nel caso di studio su cui si basa il sistema descritto in questo lavoro di tesi la maggior parte del testo

```
'entities': [                                   # lista delle entità
        {
            'span':                             # l'intera frase nominale
          'lemma_span'                          # versione lemmatizzata
                                                dello span
            'head':                             # il sostantivo padre
          'lemma_head'                          # la versione lemmatizzata
                                                del sostantivo padre
           'modifiers': [                       # i modificatori
                {
                        'dep':                  # il tipo di dipendenza
                        'span':                 # l'intervallo del modifica-
                                                tore(l'intero modificatore)
                        'lemma_span':           # la versione lemmatizzata
                                                dello span
                }
                ]
        },
        ],

'relations': [                                  # lista delle relazioni
        {
            śubject':                           # l'identificativo del soggetto (entità)
            óbject':                            # l'identificativo dell'oggetto (entità)
            ŕelation':                          # la relazione che lega soggetto ed oggetto
        }
        ]
```

Tabella 3.1. Rappresentazione, tramite dict e list, delle entità e delle relazioni presenti nel testo in ingresso

ricevuto contiene verbi in forma imperativa e quindi in cui il soggetto è implicito. Tali regole insieme ad una gestione personalizzata delle frasi complesse ha portato alla personalizzazione sostanziale del framework che si è comunque rivelato un ottima base di partenza per lo sviluppo dell'algoritmo di analisi del testo.

3.2 Spacy

SpaCy è una libreria open-source scritta in Cython/Python utilizzata per l'analisi del linguaggio naturale e per l'apprendimento automatico che sta avendo sempre più successo all'interno del mondo della ricerca grazie alle sue prestazioni eccellenti nell'estrazione di informazioni da testo, grazie ad una grande varietà di plug-in ed algoritmi di machine learning che la rendono flessibile.

3.2.1 Caratteristiche

La libreria Spacy supporta il flusso di lavoro di deep learning nelle reti neurali convoluzionali per quanto riguarda il PoS tagging, ovvero la marcatura di una parola presente in un testo come corrispondete ad una particolare parte del discorso, e per quanto riguarda l'analisi delle dipendenze e il rilevamento di entità denominate. Le caratteristiche di questa libreria sono quelle riportate nella pagina ufficiale di SpaCy riferite alla libreria nella versione 3.0, la stessa che viene utilizzata all'interno del sistema descritto in questo sistema.

- Supporta più di 64 lingue, tra cui l'italiano e l'inglese
- Apprendimento multi-task con trasformatori pre-addestrati
- Vettori di parole pre-addestrati
- Velocità nell'elaborazione
- realizzato per prodotti commerciali
- Tokenizzazione linguisticamente motivata
- Componenti per il riconoscimento di entità , tagging di PoS, analisi delle dipendenze, segmentazione di frasi, classificazione del testo , lemmatizzazione, analisi morfologica, collegamento di entità
- Facilmente estendibile con componenti e attributi personalizzati
- Supporto per modelli personalizzati in PyTorch , TensorFlow e altri framework

3.2.2 Motivo della scelta

Spacy è una libreria che sta acquisendo notevole successo nel campo della ricerca, molti studi che trattano dell'elaborazione del linguaggio naturale, natural language processing (NLP), utilizzano questa libreria per la fase di NLP, anche il framework Scene Graph Parse citato precedentemente utilizza spacy come libreria per le sue funzionalità di NLP.
Un ulteriore motivazione che ha portato alla scelta di questa libreria consiste nella presenza di vettori di parole pre-addestrati per la fase di ricerca dei sinonimi e nella accuratezza dimostrata in altri studi per la fase di tokenizzazione, fase essenziale in questo lavoro di tesi in quanto fondamenta dell'analisi del testo.

3.3 MASK R-CNN

Mask R-CNN (38) è una rete neurale convoluzionale o convolutional neural network (CNN), ovvero un tipo di rete neurale che prende ispirazione da processi biologici ed progettata in maniera tale da utilizzare al minimo la pre-elaborazione. Mask R-CNN è un estensione di Faster R-CNN, che in aggiunta agli output già presenti ovvero la bounding box e il classificatore, permette di prevedere la maschera di una istanza (Figura 3.1).
Questo framework rappresenta lo stato dell'arte per quanto riguarda la segmentazione dell'immagini, ovvero il processo di suddivisione di un'immagine in segmenti ossia un insieme di pixel che vengono utilizzati per classificare oggetti simili come un'unica classe, e per quanto riguarda la segmentazione delle istanze che si occupa di rilevare, localizzare e classificare tutti gli oggetti presenti in un'immagine.

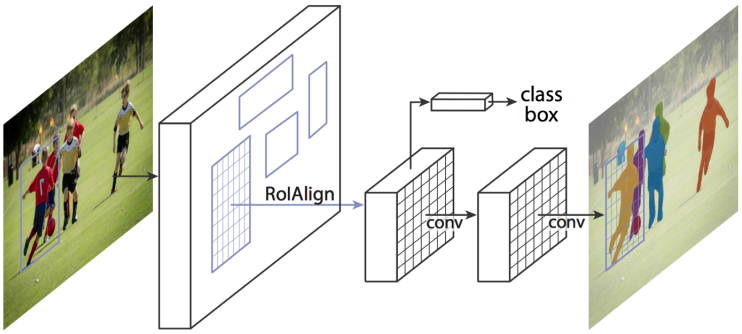

Figura 3.1. Architettura di Mask R-CNN per la segmentazione di un istanza

3.3.1 Caratteristiche

I vantaggi nell'utilizzare Mask R-CNN rispetto ad altre reti neurali come Faster R-CNN sono :

- semplicità nell'addestramento
- prestazioni migliori in ogni task, rispetto a tutti i modelli a singola entry
- efficienza a discapito di un piccolo overhead rispetto a Faster R-CNN
- Velocità nell'elaborazione
- flessibilità, in quanto è semplice da generalizzare per altri compiti.

Il framework a seguito di un'operazione di detection eseguita su un'immagine ricevuta in ingresso restituisce un file contenente le informazioni dell'immagine insieme alla lista delle maschere degli oggetti rilevate e la lista delle regioni di interesse, o region of interest (ROI) rappresentanti i confini dell'area che rappresentano la locazione per ogni componente rilevato all'interno dell'immagine

3.3.2 Motivo scelta ed implementazione nel sistema

La caratteristica principale che lo distingue dalle altre reti neurali è il suo allineamento pixel-to-pixel. Mask R-CNN, in parallelo con la previsione dei bounding box e delle classi, calcola una maschera binaria per ogni ROI, questo approccio è in contrasto con i sistemi più recenti, in cui la classificazione dipende dalle previsioni della maschera. La versione utilizzata in questo sistema è realizzata in Python3, Keras e TensorFlow e si basa su Feature Pyramid Network (FPN) e ResNet101 come backbone, ovvero una rete neurale convoluzionale a 101 strati. La Feature Pyramid Network (FPN) è un estrattore di funzionalità che produce mappe di funzionalità di dimensione proporzionali a più livelli partendo da un immagine ricevuta in input,

3.4 Word2Vec (Gensim)

Word2vec è un'insieme di modelli utilizzati per produrre word embedding, e si avvale di una rete neurale superficiale per l'elaborazione del linguaggio naturale. L'algoritmo, ricevendo in ingresso un corpus, fornisce in output un'insieme di vettori che rappresentano la distribuzione semantica delle parole. Il word embedding, conosciuto come rappresentazione distribuita delle parole, permette di memorizzare le loro informazioni semantiche e sintattiche partendo da un corpus non annotato, e costruendo uno spazio vettoriale in cui la vicinanza di tali vettori

rappresentanti le parole, sono più vicine tanto più queste occorrono nei medesimi contesti linguistici, ovvero quanto più le parole sono semanticamente simili.

3.4.1 Caratteristiche

Con word2vec sono possibili 2 metodi per l'apprendimento delle rappresentazioni delle parole:

- Modello Bag-of-Words : prevede la parola in esame sulla base di parole che la circondano, prima e dopo, ovvero in base al contesto. Questa architettura è chiamata modello bag-of-words poiché l'ordine delle parole nel contesto non è importante.
- Modello Skip-gram: prevede le parole all'interno di un certo intervallo prima e dopo la parola ricevuta come input

All'interno del sistema di questo lavoro di tesi, per sfruttare word2vec in python, si sfrutta spaCy e Gensim (39), una libreria python per la modellazione di argomenti, che fornisce l'accesso a word2vec e ad altri algoritmi di word embedding.

3.4.2 Implementazione nel sistema

In questo sistema si è optato per modelli pre-addestrati al posto di addestrare un modello ad hoc, in quanto esistono, open source, modelli contenenti una grande set di dati, difficile se non impossibile da uguagliare con un modello addestrato personalmente. Lo svantaggio derivante dall'utilizzo di questi tipi di modelli è che, le parole contenute all'interno di questo enorme set di dati, potrebbero non catturare le peculiarità del linguaggio nel dominio specifico dell'applicazione che, nel nostro caso, potrebbero ad esempio non avere dati riguardanti i componenti specifici di una stampante. I modelli utilizzati in questo sistema sono:

- en_core_web_md di spaCy, questo modello è integrato nei modelli necessari al normale utilizzo di spaCy, quindi non implica la necessità di scaricare ed installare ulteriore materiale. en_core_web_md è il modello più grande standard di spaCy, che include 20k vettori unici con 300 dimensioni.
- modello di set di dati di Google News, è un modello pre addestrato in Gensim, contenente incorporamenti a 300 dimensioni per 3 milioni di parole e frasi

I vettori di parole in spaCy sono "statici" nel senso che non vengono appresi parametri dei modelli statistici, inoltre non è presente la possibilità di apprendere tabelle vettoriali di parole. Per ovviare a questo problema, in questo sistema, si utilizza Gensim.

3.5 WordNet

WordNet è un grande database semantico-lessicale per la lingua inglese con lo scopo di descrivere i concetti espressi tramite vocaboli.

3.5.1 Caratteristiche

L'organizzazione lessicale avviene tramite il raggruppamento di sostantivi, verbi, aggettivi e avverbi con un significato affine, in gruppi di sinonimi cognitivi (synsets), dove ognuno dei quali esprime un concetto differente. I synset internamente hanno dei collegamenti in termini di concezione semantica e relazioni lessicali per cui le differenze del significato sono numerate e definite. WordNet, rispetto ad altri database che raggruppano le parole in base al loro significato, collega le parole non unicamente in base alla parola stessa, ovvero stringa di caratteri, ma in base al suo significato. Inoltre WordNet etichetta le relazioni semantiche tra le parole e non si basa quindi semplicemente sulla similarità del significato delle parole.

3.5.2 Motivo della scelta

Queste differenze permettono a WordNet di ottenere degli ottimi risultati per la ricerca di sinonimi appropriati per una data parola, infatti ha portato all'utilizzo di questo database all'interno del sistema. Nello specifico, dato che il sistema è stato sviluppato utilizzando la libreria SpaCy, è stata scelta una versione compatibile con questa libreria, ovvero spaCy WordNet, ossia un componente per utilizzare WordNet con questa libreria, che viene utilizzata all'interno del sistema descritto in questo lavoro di tesi, a seguito della sua aggiunta alla pipeline di SpaCy.

SpaCy WordNet combina l'interfaccia per WordNet di NTLK con domini WordNet, per permettere agli utenti di:

- ottenere tutti i synsets per un token
- ottenere e filtrare i synsets per dominio. Ad esempio, trova il sinonimo del verbo "rotate" all'interno del dominio "geometry"

I domini WordNet sono risorse lessicali create in modo semi-automatico aggiungendo a WordNet delle etichette rappresentanti i domini. Ogni synsets di WordNet è stato annotato con una o più etichette rappresentanti i domini.

3.6 OpenCV

OpenCV (40),o Open Source Computer Vision Library, è una libreria multi piattaforma con licenza gratuita realizzata in python ed utilizzata per computer vision e machine learning. L'utilizzo principale di questa libreria è quella di elaborare immagini e video in tempo reale per operazioni di recognition e detection.

3.6.1 Caratteristiche

La libreria ha più di 2500 algoritmi ottimizzati che comprendono anche algoritmi per le operazioni di computer vision e machine learning allo stato dell'arte. Gli algoritmi presenti all'interno della libreria sono utilizzati soprattutto nel campo del riconoscimento e dell'individuazione di facce ed oggetti, nel tracciare i movimenti di una telecamera o di oggetti in movimento, così come estrarre i modelli 3D di oggetti e in altre operazioni riguardanti l'elaborazione delle immagini e dei video. La grande comunità dietro questa libreria, più di 47mila persone, permette a quest'ultima di rimanere aggiornata e di risolvere tempestivamente i bug a seguito di segnalazioni provenienti anche dagli utenti. La libreria è largamente utilizzata anche da compagnie di alto spessore come Google, Intel, IBM e Microsoft.

3.6.2 Motivo della scelta

Per questo sistema è stato scelta questa libreria in base al largo utilizzo nel mondo della produzione e della ricerca, che ne comprovano l'efficacia ed efficienza, e in base alla sua compatibilità con diversi sistemi operativi e con il linguaggio Python. Un ulteriore motivo che ha portato alla scelta di questa libreria consiste nel fatto che durante i corsi tenutisi al Politecnico di Torino, la libreria è stata illustrata nello specifico, e attraverso i laboratori e i progetti è stato possibile sperimentarne l'utilizzo e l'efficacia.

Parte II

Seconda Parte

Capitolo 4

Il sistema

4.1 Nozioni generali sul sistema

Il sistema riceve in ingresso i testi e le immagini prelevati dai manuali di istruzione delle stampanti in lingua inglese. Il testo viene estratto manualmente dai manuali e riportato all'interno di un file "txt",il testo all'interno rappresenta una lista numerata di istruzioni scritte in linguaggio naturale da eseguire in ordine di scrittura, in cui il numero viene utilizzato come identificativo dell'istruzione mentre la posizione nella lista per la temporizzazione. Le immagini, come il testo, vengono estratte manualmente dal manuale e salvate in formato "jpg", nel caso in cui le immagini siano legate a delle istruzioni presenti all'interno del testo, per legare le immagini alle istruzioni, queste avranno il nome corrispondente al numero dell'istruzione, ad esempio "1.jpg" se l'immagine raffigura l'operazione da eseguire nell'istruzione il cui numero identificativo e "1".
Le immagini vengono inserite in 2 cartelle separate, la cartella delle immagini di calibrazioni, chiamata "images_calibration" conterrà le immagini della stampante con tutti i componenti "chiusi" o inseriti negli appositi alloggi, mentre l'altra, chiamata "images" conterrà le immagini che rappresentano graficamente le istruzioni riportate all'interno del testo. Lo scopo del sistema è di ottenere un video animato tramite modelli 3D che riproduca le animazioni riguardanti le istruzioni riportate all'interno del testo.

4.1.1 Architettura del sistema

Il capitolo illustra l'architettura del sistema descritto in questo lavoro di tesi, tramite una descrizione generale del funzionamento e delle operazioni eseguite da questo, argomenti che verranno trattati più approfonditamente nei capitoli successivi (Figura 4.1).

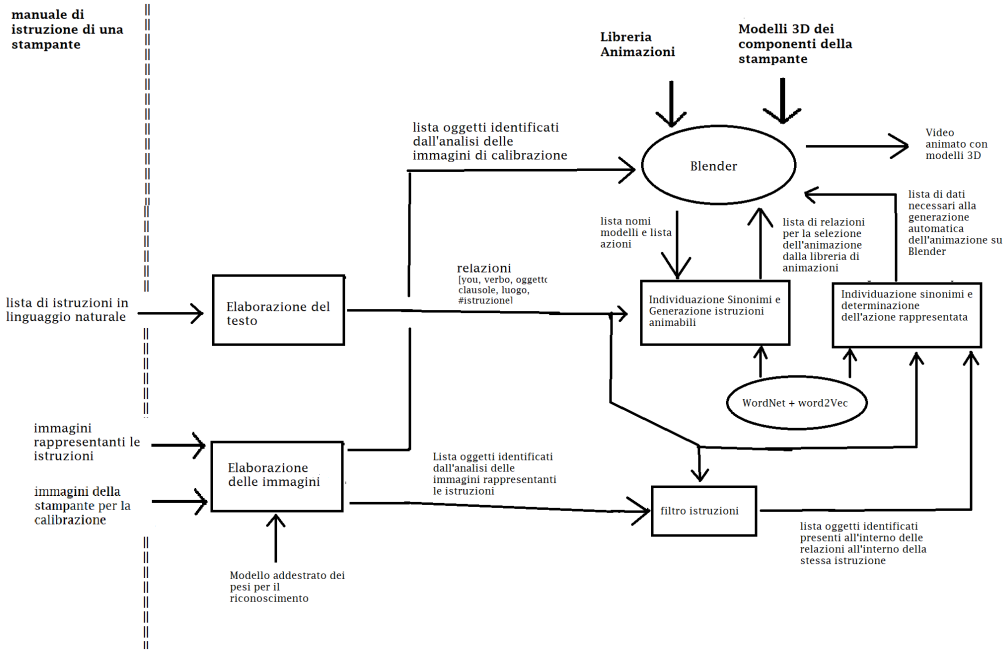

Figura 4.1. Architettura del sistema: a sinistra sono presenti le risorse prelevate dal manuale di istruzioni, in alto sono mostrate le risorse pre caricate su Blender, all'interno dei contenitori rettangolari sono riportate le funzioni eseguite, i box ellittici contengono il software,framework o le librerie esterne utilizzate ed infine i nomi senza contenitore rappresentano i dati ottenuti da una determinata operazione

Il sistema riceve in ingresso il testo contenente le istruzioni da animare scritte in linguaggio naturale e i due insiemi di immagini, uno necessario per la fase di calibrazione della camera virtuale presente all'interno di Blender e uno necessario all'estrapolazione dei dati utili alla generazione automatica delle animazioni, gli insiemi di immagini vengono selezionati in base alla cartella in cui si trovano, quindi rispettivamente "images_calibration" e "images"
Il sistema estrapola dal testo in ingresso le relazioni presenti al suo interno. Queste relazioni rappresentano una singola istruzione semplice da animare, ovvero un'istruzione in cui è eseguita una sola azione. Per eseguire questa operazione il sistema utilizza il framework scene graph parser opportunamente modificato per il sistema, le modifiche riguardano la fase di analisi in cui sono state aggiunte delle regole aggiuntive al fine di analizzare correttamente le frasi in forma imperativa.
Le relazioni così ottenute avranno i seguenti dati:

- numero dell'istruzione
- soggetto

- verbo dell'azione eseguita
- oggetto su cui avviene l'azione
- possibili clausole che possono modificare il significato dell'azione
- possibili luoghi rilevanti a cui si riferisce l'istruzione ad esempio "put the cartridge into the slot" dove "slot" rappresenta il luogo

Entrambe le liste di immagini vengono analizzate dal sistema al fine di rilevare i componenti presenti all'interno delle immagini ed ottenere attraverso l'elaborazione dei dati, i punti rappresentanti i 4 angoli del componente. Il riconoscimento dei componenti viene svolto tramite il framework Mask R-CNN che riceve oltre alle immagini da analizzare anche il modello dei pesi per il riconoscimento delle immagini addestrato per questo scopo.
In uscita il sistema otterrà per tutti i componenti rilevati all'interno delle immagini provenienti da entrambe le liste un dizionario contenente i dati necessari a descrivere il componente, ovvero:

- il nome dell'immagine in cui il componente è stato rilevato, privato della sua estensione (corrispondente al numero dell'istruzione)
- nome del componente (determinato durante la fase di riconoscimento)
- posizione del componente, ovvero i 4 angoli ricavati tramite l'analisi
- il centro del componente
- la ROI del componente

Per ogni istruzione Il sistema confronta i nomi dei componenti presenti all'interno delle relazioni con quelli all'interno della lista dei componenti. Il confronto viene svolto per eliminare dalla lista dei componenti gli elementi che non hanno una corrispondenza all'interno della lista delle relazioni.
Durante la fase di rilevamento dei componenti è plausibile che più componenti vengano riconosciuti all'interno della stessa immagine e quindi associati alla stessa istruzione, con questo filtro il sistema si assicura di eliminare tutti i componenti che in realtà non sono citati dall'istruzione. Ad esempio all'interno della immagine "4.jpg" possono essere rilevati i seguenti componenti ["printer", "front cover", "ink cartridge"] poichè tutti presenti all'interno dell'immagine. Il testo per all'istruzione identificata dal numero 4 cita "insert the ink cartridge", la relazione corrispondente generata sarà ["4","you","insert","ink cartridge"]. Il sistema confronterà i nomi dei componenti rilevati, ovvero "printer", "front cover" e "ink cartridge" con il nome all'interno della relazione, ovvero "ink cartridge" eliminando gli elementi associati

a "printer" e "front cover" mantenendo unicamente l'elemento associato a "ink cartridge"
In aggiunta durante l'operazione di filtraggio, a tutti i componenti che sono citati all'interno dell'istruzione a cui sono associati, il sistema assegnerà come verbo l'azione estrapolata dal testo, ottenendo in questo modo una possibile istruzione a partire dal riconoscimento dei componenti, riprendendo l'esempio sopra, il sistema una volta eliminati gli elementi "printer" e "front cover" assegnerà come azione all'elemento associato a "ink cartridge" il verbo "put" inserendolo all'interno del dizionario rappresentante l'elemento, il dizionario quindi da {"name":"ink cartridge", "roi":[...], "vertex":[...], "image":"4"} diventerà {"name":"ink cartridge", "roi":[...], "vertex":[...], "image":"4", "action":"put"}
Il sistema a questo punto, sia per la lista di relazioni che per la lista di istruzioni in uscita dall'operazione di filtraggio, otterrà una lista di istruzioni animabili e una lista di istruzioni generabili automaticamente su Blender.
Per ottenere quest'ultime liste di istruzioni il sistema data la lista di relazioni/istruzioni provenienti dall'analisi dell'immagine e data la lista ricevuta da Blender dei nomi dei modelli e delle azioni presenti al suo interno, identificherà tutti i possibili sinonimi per i nomi presenti nelle liste.
Questi sinonimi vengono ottenuti tramite l'utilizzo di wordNet e una lista di sinonimi aggiuntivi che caratterizzano il contesto della stampante in quanto non riconosciuti come tali all'interno di wordNet.

Il sistema per ottenere l'istruzione animabile o generabile, data un'istruzione (descritta dalla relazione o proveniente dall'analisi dell'immagine) confronterà i nomi dei componenti e dell'azione, e i rispettivi sinonimi, con tutti i nomi presenti all'interno della lista dei nomi dei modelli e delle azioni presenti su Blender, insieme a tutti i loro sinonimi.
Tramite questa operazione di confronto, eseguita tramite l'utilizzo di word2vec, il sistema identificherà i nomi dei modelli e delle azioni a cui si riferisce l'istruzione e otterrà l'istruzione animabile o generabile automaticamente su Blender.
Il sistema una volta ottenuta la lista delle istruzioni comunica con Blender inviando, tramite richiesta POST, le liste calcolate e la lista di oggetti identificati dalle immagini di calibrazione.
Su Blender il sistema utilizza la lista degli oggetti identificati per la calibrazione e la successiva posa della camera mentre utilizza la lista di istruzioni animabili per selezionare un'animazione da mostrare a video, quando non è possibile per la stessa istruzione generarne una in modo automatico tramite l'utilizzo dei dati presenti all'interno della lista delle istruzioni generabili in modo automatico.
Il sistema infine assegna dei sottotitoli al video per una maggiore comprensibilità e genera in automatico un video in formato AVI

Capitolo 5

Analisi del linguaggio naturale

Questo capitolo tratta nel dettaglio della prima fase del sistema, ovvero della analisi ed estrapolazione di informazioni rilevanti che legano le entità presenti all'interno del testo. Le informazioni che vengono estrapolate sono:

- Le entità presenti all'interno di una istruzione
- Il ruolo che queste entità ricoprono all'interno di una istruzione (ovvero se una determinata entità è un soggetto, oggetto, clausola che cambia il significato della frase o luogo/oggetto a cui si riferisce la frase)
- La relazione che coinvolge una o più entità, ovvero i verbi.

Questo sistema si specializza principalmente sull'analisi di testi relativi a manuali di istruzioni di stampanti, ma può essere generalizzato anche ad altri oggetti o contesti con le dovute modifiche.

5.1 L'analisi attraverso spaCy

L'analisi del testo viene eseguita tramite il framework, Scene Graph Parser, che sfrutta spaCy 3.0 per l'analisi del linguaggio naturale. Il framework permette di analizzare le frasi tramite l'esplorazione del grafo delle dipendenze. Le regole definite al suo interno sono pensate per lavorare unicamente con testi il cui soggetto è esplicito all'interno della frase. In questo lavoro di tesi sono state quindi effettuate delle modifiche con lo scopo di adattare il framework al caso di studio. Nel caso in cui il soggetto fosse sottinteso, ad esempio in "open the door", il sistema considerava come soggetto della frase "you" in quanto all'interno dei manuali di

istruzione tutte le azioni sono intese come da eseguire da parte dell'utente. All'interno del framework non erano inoltre gestite le frasi negative come "don't open the door" e i nomi di azioni composti come ad esempio "push down". Sono state inoltre modificate alcune regole, in particolare la parte della memorizzazione dei dati, tramite la gestione dell'inserimento dei campi riguardanti le clausole e il luogo. Le regole inoltre non premettevano di gestire frasi in cui erano presenti più oggetti per lo stesso verbo ad esempio "open the door or the window", in questi casi all'interno del sistema è stato sostituito l'oggetto con una lista di oggetti. All'interno del sistema sono stati introdotti i meccanismi per la modifica delle relazioni a seguito dell'analisi di altre entità che si riferivano alla stessa azione, ad esempio "put the cartridge into the slot" creava la relazione quando analizzava l'entità "cartridge" che doveva però essere modificata con l'aggiunta dell'informazione del luogo quando analizzava l'entità "slot". SpaCy è stato utilizzato con la pipeline "en_core_web_sm" in quanto è ottimizzata per la CPU, e contenente i seguenti componenti:

- modello "token to vector", utile per condividere una singola sotto rete tra più componenti ad esempio per condividere una rete di incorporamento e CNN (rete neurale convoluzionale) tra l'analizzatore delle dipendenze, il tagger e il riconoscitore di entità.
- Tagger, componente per l'etichettatura delle parti del discorso (PoS)
- parser, componente dell'analizzatore delle dipendenze, che apprende la segmentazione della frase e l'analisi delle dipendenze etichettate
- senter, componente per la segmentazione delle frasi
- attribute ruler, componente necessaria all'assegnazione, basandosi su regole, degli attributi del token
- lemmatizzatore, componente per la lemmatizzazione, assegna moduli di base ai token utilizzando regole sui tag della parte del discorso o tabelle di ricerca
- NER, componente per il riconoscimento di entità denominate basato sulla transizione che identifica intervalli non sovrapposti di token

Il sistema descritto in questo lavoro di tesi riceve in ingresso un testo generico in linguaggio naturale e per ogni riga all'interno del testo separa il numero rappresentante l'indice dal resto della frase. Per effettuare questa separazione il sistema controlla se la prima parola presente all'interno della riga è un numero, nel caso lo fosse salva il numero e lo rimpiazza all'interno della riga con un carattere vuoto (""). Il numero salvato e la riga modificata vengono inseriti dal sistema all'interno di un dizionario che viene aggiunto ad una lista contenente tutti

i dizionari rappresentanti le righe. Ogni frase contenuta nel dizionario viene analizzata dal framework scene graph parser indipendentemente dalle altre. Le linee singole vengono elaborate indipendentemente dal framework Scene Graph Parser, che analizza la frase e genera grafi di scena, ovvero dei grafi che rappresentano le relazioni tra le entità citate all'interno dell'istruzione. Il grafo viene generato tramite l'utilizzo dell'analizzatore delle dipendenze di spaCy. La fase analisi delle dipendenze è composta da 3 fasi :

1. Ricerca di tutte le entità tra le "frasi nominali di base", ovvero frasi che hanno un nome come testa della loro struttura, e risoluzione dei loro modificatori.

2. Individuazione del soggetto dei verbi, nel caso in cui una frase sia in forma imperativa, ovvero senza soggetto esplicito, il sistema qui descritto memorizza come soggetto di una azione estrapolata all'interno della frase, il soggetto "you", in quanto l'azione deve essere eseguita presumibilmente da chi sta leggendo il manuale.

3. Definizione delle relazioni tra le entità trovate

5.2 Ricerca delle entità

5.2.1 Identificazione dell'indice

Le frasi riguardanti istruzioni, all'interno dei manuali di istruzioni, solitamente hanno una struttura determinata, composto da un indice, che ne definisce l'ordine, e dalla istruzione in sé, che dichiara le azioni da eseguire in quel determinato passo, Sotto è presente un esempio di un'ipotetica frase che rispetta il formato descritto poc'anzi.

1 Make sure that the printer is turned on

Per evitare che l'indice all'interno dell'istruzione venga dichiarato come un'entità invece che come indice, prima di passare alla fase di analisi delle dipendenze, genericamente descritta nel paragrafo precedente, il sistema controlla la prima parola della frase, ignorando l'indentazione, se questa parola è un numero, essa viene rimossa dalla frase, e posta all'interno di una lista di dizionari rappresentanti la lista di istruzioni. I dizionari presenti all'interno della lista avranno un campo che identifica l'indice ed un altro che identifica l'istruzione, secondo la struttura della frase precedentemente descritta. Il sistema mette in questo modo in relazione l'indice con la frase. La scelta di un dizionario piuttosto che di un array, in cui utilizzare l'indice dell'istruzione anche come indice dell'array stesso, è dovuta alla possibilità di rilevare non solo indici numerici ma anche sotto forma letterale (A,B,C. . . ..) ed alla possibilità che un'istruzione possa essere definita

tramite sotto-istruzioni (Tabella 5.1), che ne specificano in modo più accurato la realizzazione.

istruzione	SottoIstruzione
4 Remove the empty FINE cartridge.	
	1 Press down on the FINE cartridge until it clicks into place
	2 Remove the FINE cartridge.

Tabella 5.1. Istruzione contenente altre sotto istruzioni per una migliore definizione

Il dizionario creato ha quindi un elemento che rappresenta l'indice, uno per la frase e uno opzionale per la lista di sotto istruzioni con la stessa struttura (5.2).

istruzione	Sotto Istruzione
instruction = dict (	
	number = indice,
	sentence = frase,
	instructions = [sotto istruzioni]
)	

Tabella 5.2. Struttura del dizionario di istruzioni dove instruction è il nome del dizionario mentre number, sentence e instructions sono i nomi delle chiavi ed infine indice,frase e la lista di sotto istruzioni sono i valori

5.2.2 Ricerca delle entità

Il sistema descritto in questo lavoro di tesi, nella prima fase dell'analisi delle dipendenze, ovvero la ricerca di entità tra le "frasi nominali di base" (Noun Chunks), ricava per ogni frase un contenitore specifico, chiamato "doc", al cui interno sono presenti e sono accessibili le annotazioni, o entità denominate, riguardanti la frase analizzata, ovvero una sequenza di token (ad esempio una parola, la punteggiatura, spazi bianchi etc). Questo contenitore permette l'accesso alle entità denominate, ovvero le informazioni chiave presenti all'interno del testo. Per l'identificazione dell'entità, in questa fase il sistema qui descritto analizza i tag della parte del discorso (POS) e le dipendenze (DEP) di ogni token presente all'interno del contenitore. Il sistema descritto in questo lavoro di tesi, durante l'analisi pone maggiore attenzione, tramite delle regole aggiuntive, ad alcuni token che non rappresentano l'entità

della frase ma che hanno proprietà simili e che quindi verrebbero erroneamente identificati come tali. Nello specifico il sistema durante l'analisi traduce i pronomi come "it" nell'entità precedentemente identificata ed analizza ulteriormente i nomi propri al fine di individuare e classificare correttamente le apposizioni, ovvero nomi accessori posti innanzi ad un altro nome al fine di cambiarne il significato. I restanti token, che non rientrano nei casi specifici, quindi né pronomi né nomi propri, vengono identificai come entità, e vengono memorizzate le seguenti proprietà:

- la parola associata al token
- l'etichetta
- la parola lemmatizzata
- il responsabile del token, ovvero l'elemento radice del token
- la lemmatizzazione della testa
- l'intervallo in cui appare l'entità all'interno della frase

I modificatori associati ad un token vengono inseriti successivamente dal sistema, al fine di memorizzarli correttamente il sistema controlla la loro dipendenza (dep). I modificatori memorizzati sono:

- i determinanti, come "the"
- i modificatori numerici, che modificano il significato di un sostantivo in termini di quantità
- i composti nominali, ad esempio "book" in "phone book"
- i modificatori aggettivali, ovvero una qualsiasi parola aggettivale che modifica il significato del sostantivo a cui è associato
- le clausole che modificano il significato di verbi o oggetti

Una volta terminata questa ulteriore analisi, il sistema qui descritto, memorizza le entità rilevate e raffinate all'interno dell'opportuna lista di entità. Ad esempio nella frase "Make sure the light is on, but not flashing." si analizza il token associato a "light" all'interno dei noun chunks. Al token viene associata, da spaCy, la dipendenza "nsubj" e il pos "NOUN" poiché non rientra nei casi speciali si procede alla normale memorizzazione dei suoi dati. Il sistema genera quindi un dizionario con la seguente forma {"span":"the light","laber":"NP", "lemma_span":"the light", "head":"light", "lemma_head":"light", "span_bounds":(2,5) }. Quando deve inserire i modificatori del token, il sistema controlla tra gli elementi figlio della radice del token, ovvero gli elementi figlio di "light". In questo esempio l'unico modificatore trovato e memorizzato è il determinante "the", di cui si salva il testo ("the"), lo span ("the") e il lemma ("the").

5.3 Definizione del soggetto

Per la definizione del soggetto, il sistema descritto in questo lavoro di tesi, analizza nuovamente i token presenti nel contenitore “Doc”. I token vengono inizialmente filtrati in modo da escludere componenti privi di significato per quanto riguarda la definizione della istruzione, ovvero spazi bianchi, punteggiatura, parentesi e virgolette. I token candidati ad essere soggetto, vengono quindi identificati e salvati opportunamente in base al tipo di dipendenza:

- i soggetti nominali, vengono memorizzati insieme al verbo a cui sono associati nel caso in cui il verbo risulti essere il genitore del token all’interno del grafo e inoltre lo stesso non risulti essere una clausola di un altro verbo/oggetto.
- i modificatori di clausola del sostantivo, come “playing” in “A woman playing the piano”, vengono memorizzati insieme al sostantivo a cui sono associati che risulta essere il genitore del token.
- gli oggetti di una preposizione, nel caso in cui siano associati ad un agente (ovvero il complemento di un verbo in forma passiva che viene introdotto dalla preposizione “by” e che esegue l’azione) che a sua volta è associato ad un token che rappresenta il verbo da eseguire, vengono memorizzati in relazione al verbo nel caso in cui non sia una clausola di un altro verbo, ad esempio The piano is played by a woman
- I verbi che ricoprono il ruolo di modificatori di clausola degli avverbi, in questo caso se è presente un ausiliare associato si ignora il token non memorizzandolo in quanto esprime concetti come l’attesa o le ripetizioni, emozioni aspetto etc, ovvero categorie non animabili in questa versione del sistema.

Poiché il sistema è pensato per ricevere in ingresso dei testi provenienti da manuali di istruzione, e poiché le frasi in questo tipo di testi, sono in forma imperativa, ovvero una forma in cui il soggetto della frase è spesso sottinteso, per includerli all’interno dei casi riconosciuti all’interno dell’analisi, il sistema memorizza il verbo accomunato al soggetto “you” nel caso in cui il token sia etichettato come verbo (nel tag della parte del discorso) e non sia presente un soggetto a cui associarlo. In questo caso se il token non è una negazione e non è già presente all’interno dei soggetti delle relazioni identificate, il sistema controlla se il token rappresenti effettivamente un verbo di un istruzione animabile, controlla quindi se il token ha altri verbi come token figli ma nessun oggetto. Nel caso in cui fossero presenti altri verbi il sistema inserirà, all’interno della lista dei possibili soggetti, il soggetto "you" associato ai verbi trovati. Ad esempio nella frase “controlla che la stampante sia accesa”, il verbo "Controlla" avrà come token figlio "sia accesa"(essere accesa) ma nessuno oggetto, il verbo "essere accesa" invece è legato all’oggetto "stampante",

verrà quindi memorizzato all'interno dei soggetti "you" legato ad "essere accesa". Il sistema infine pone ulteriore attenzione alle frasi contenenti token con ruolo di negazione, infatti poiché in questa versione del sistema descritto in questo lavoro di tesi, si animano unicamente istruzioni da eseguire, e non quelle da evitare, il sistema esclude e non memorizza tali token tra quelli riconosciuti ma memorizza il verbo tra quelli da non animare.

5.4 Identificazione delle relazioni

Nell'ultima fase dell'analisi delle dipendenze il sistema analizza le istruzioni ottenendo le relazioni che divide in relazioni imperative, ovvero quelle utili per la generazione delle animazioni, e relazioni tra gli oggetti ad esempio "sheets of paper", non utilizzati per la generazione dell'animazione in questa versione del sistema. Per procedere con l'identificazione delle relazioni, il sistema estrapola le entità presenti all'interno dei noun chunks, ovvero le frasi nominali base ed analizza in modo differente le entità in base alla loro dipendenza (dep), parte del discorso rappresentata (pos) e presenza o assenza tra i soggetti identificati nella fase precedente dal sistema. In base a queste caratteristiche l'entità assume un ruolo particolare all'interno della frase. Vengono quindi definite manualmente una serie di regole euristiche che il sistema segue al fine di analizzare correttamente le frasi passate. Per prima cosa il sistema segue regole diverse in base alla complessità della frase, per identificare la complessità il sistema controlla in che relazione è il token in esame, rappresentante l'entità da analizzare, rispetto al token rappresentante la radice della frase stessa. Per individuare il token radice, il sistema controlla le parentele dell'entità, in particolare controlla il genitore (token.head) e il genitore del genitore(token.head.head). L'identificazione della posizione della radice della frase può rappresentare un'informazione importante che può portare alla semplificazione dell'analisi da parte del sistema. Nel caso in cui la radice sia individuata in uno dei due token prima citati, il sistema eviterà di controllare ulteriori parentele superiori (ad esempio token.head.head.head) in cerca di informazioni utili per la comprensione del ruolo dell'entità all'interno della frase, ad esempio nella frase "Remove the new ink cartridge from its package." l'entità corrispondente alla parola "the new ink cartridge" ha come token.head "Remove" che è la radice della frase, il sistema non controlla quindi le parentele di ordine superiore (token.head.head e superiore) in quanto inutile, il padre della radice infatti sarà uguale a se stesso. Nel caso cui la radice venga individuata dal sistema, ed inoltre il genitore dell'entità sia un verbo, il sistema applica le seguenti regole:

- se l'entità è un soggetto nominale il sistema controlla se il verbo associato (il genitore) è una clausola di avverbio e nel caso se ha un marker collegato, ovvero una parola che introduce una clausola subordinata ad un altra. Il sistema

nel caso in cui sia già presente una relazione, identifica il verbo come clausola della relazione precedente, altrimenti il sistema crea una nuova relazione in cui il verbo è il verbo principale dell'azione.

- se l'entità è un soggetto nominale il sistema nel caso in cui il verbo è la radice o un verbo di congiunzione, genera la relazione con il verbo in esame come verbo principale dell'azione
- se l'entità è un oggetto diretto o un attributo, il sistema controlla se questo sia un nome alternativo per una relazione già memorizzata, in quel caso aggiunge l'oggetto come possibile nome dell'oggetto tramite l'utilizzo di una lista. Se non è già presente una relazione che si riferisce a quell'azione il sistema ne genera una nuova.
- Se l'entità è un soggetto nominale in forma passiva, il sistema cerca di ottenere dal soggetto in forma passiva l'oggetto a cui si riferisce
- Se l'entità è una congiunzione il sistema l'ignora e non genera alcuna relazione

Nel caso in cui il sistema individui la radice ma il genitore (token.head) non ricopra il ruolo di verbo si controlla se sia il genitore del genitore (token.head.head) a ricoprire il ruolo del verbo all'interno della frase, ad esempio nella frase "move to position" l'entità è "position", il genitore(token.head) è "to" mentre il genitore del genitore (token.head.head) è "move" ovvero il verbo. In quel caso si applicano le seguenti regole euristiche:

- se l'entità è un oggetto diretto o un attributo il sistema controllerà se è associabile al verbo (token.head.head), controllerà quindi se il verbo è in forma negativa e se l'entità è l'unico oggetto plausibile su cui si può svolgere l'azione, ad esempio nella frase "Apri la porta dell'armadio o della camera" l'entità "camera" è un possibile oggetto mentre "armadio" un altro, il token "porta" è il token.head dell'oggetto mentre "Apri" è il token.head.head ed è il verbo.
- Se il genitore dell'entità è una preposizione, ad esempio “press on the button” il sistema interpreterà l'entità come il luogo dell'azione “the button” nel caso sopracitato per la relazione precedente, mentre “on” ovvero il genitore come un'ulteriore indicazione che specifica maggiormente l'azione da eseguire. Nel caso in cui non fosse presente una relazione precedente il sistema genera una relazione in cui l'entità è l'oggetto della relazione.
- Se sia il genitore dell'entità che l'entità stessa sono sostantivi, il sistema controlla se è presente un verbo li lega, ad esempio “apri la porta e la finestra”, per questo controlla se tra i figli del genitore dell'entità è presente un modificatore di coordinamento (ad es “and”), in quel caso genera una relazione per entrambi i token, legati allo stesso verbo.

- Se il genitore dell'entità in esame è un aggettivo che definisce l'entità, il sistema salva il nome dell'oggetto modificato opportunamente
- se il genitore dell'entità in esame è una clausola complementare il sistema genera una relazione con l'entità come oggetto, il genitore come clausola e il genitore del genitore dell'entità come verbo.

Nel caso in cui la radice venga individuata dal sistema, ma nessuno dei due token sopracitati rappresenti il verbo all'interno della frase, il sistema non prosegue oltre con l'analisi della frase, e passa alla frase successiva. Un'ulteriore possibilità gestita prevede il caso in cui il sistema non individui la radice, in questo caso l'analisi delle relazioni continua oltre il secondo livello di "parentela" all'interno del grafo delle dipendenze, ovvero oltre token.head.head., un esempio può essere una la frase "open the output tray and than pull out the tray extender" dove "pull out" non sarà la radice della frase per "tray extender", occorrerà gestire quindi questo caso in modo da ottenere la relazione "you pull out tray extender". Le regole qui applicate dal sistema sono dunque diverse rispetto ai precedenti casi:

- se il token genitore dell'entità in esame è un verbo il sistema identifica la frase come una frase semplice all'interno di una frase più complessa, è genera quindi una relazione composta dall'entità come oggetto e dal token genitore come verbo, come nell'esempio sopra citato "pull out the tray extender".
- Nel caso in cui infine il sistema individui il token.head.head.head come il token rappresentante il verbo ed inoltre il token.head.head come il token rappresentante un aggettivo ed il token.head come sostantivo, il sistema genererà una relazione in cui i due sostantivi saranno elementi di una lista passata come possibili nomi dell'oggetto della relazione, ad esempio "Handle part of front cover or printer cover" dove l'entità è "printer", il token.head è "front" un sostantivo, il token.head.head è "part" un aggettivo e il verbo è il token.head.head.head ovvero "Handle".
- se il verbo è rappresentato dal token genitore del genitore dell'entità (token.head.head), il sistema cerca di capire il ruolo ricoperto dal genitore dell'entità (token.head) per generare correttamente la relazione. Nel caso in cui il token genitore dell'entità ricopra il ruolo di preposizione, se non è stata generata ancora alcuna relazione, il sistema genererà la relazione usando come oggetto l'entità. Se è già stata generata una relazione, il sistema controllerà se l'istruzione si riferisce alla stessa azione, e nel caso aggiorna la relazione precedente con il campo rappresentante il luogo.
- Come controllo più estremo, per frasi ancora più complesse, il sistema verifica se il token.head.head.head, ovvero il padre del padre del padre dell'entità in

esame, sia un verbo. Nell'eventualità che lo fosse il sistema esegue dei controlli sul token padre (token.head) e sul padre del token padre (token.head.head) cercando di individuare il ruolo che questi ricoprono all'interno della frase. Nello specifico il sistema :

- se il token.head è una preposizione tra due sostantivi, ovvero se l'entità (token) è un sostantivo, se il token.head è una preposizione e se il token.head.head è anch'esso un sostantivo. In questo caso l'oggetto sarà composto dalla composizione dei due token.
- Se il token.head è una preposizione ma il token.head.head non è un sostantivo, in questo caso il sistema aggiorna se presente la relazione precedente inserendo un campo riguardante il luogo

In entrambi i casi il testo del token rappresentante la preposizione viene confrontato con i testi memorizzati all'interno di una lista, definita all'interno del sistema, contenente la maggior parte delle preposizioni di locazione e posizione in lingua inglese.

Il sistema indifferentemente dal caso, tra i casi precedentemente descritti, controlla se l'entità si riferisce ad un'istruzione il cui verbo è in forma negata, in quel caso passa alla prossima entità da analizzare senza generare alcuna relazione. Inoltre controlla se il verbo è composto da più parti "es. push down" memorizzando la sua forma completa nel caso fosse necessario ed infine prima di generare una nuova relazione controlla se l'oggetto è un nome alternativo per un oggetto in una determinata azione ad esempio "front cover or printer cover", in quel caso invece di generare una nuova relazione, aggiunge l'oggetto alla lista degli oggetti alternativi per la relazione a cui si riferisce. I soggetti in questa fase vengono identificati tramite la loro posizione all'interno della frase, poiché un soggetto potrebbe essere un soggetto valido per un azione ma non in un'altra. Il sistema utilizza quindi gli indici al posto del testo per evitare che il soggetto risulti sempre valido all'interno di un'istruzione. Per un motivo differente il sistema utilizza gli indici anche per gli oggetti, in questo caso la motivazione risale al modo in cui spaCy memorizza e gestisce i token, spaCy spezzetta le entità in token elementari non ulteriormente scomponibili, la risoluzione delle relazione viene eseguita su questi token, memorizzare il testo di uno di questi token porterebbe alla perdita delle informazioni riguardanti il resto del nome del componente, il sistema qui descritto utilizza quindi l'indice e la lista delle entità, per recuperare il nome completo del token e all'occorrenza le proprietà associate, memorizzate durante la prima fase dell'analisi delle dipendenze. Il sistema traduce infine i soggetti e gli oggetti dalla loro posizione nel loro nome effettivo e raffina i restanti campi quando presenti, come luogo e clausole, eliminando componenti non necessari come ad esempio il determinante (es "the") Alla fine delle tre fasi, il sistema memorizza la lista delle

entità, delle relazioni delle frasi in forma imperativa, delle relazioni tra oggetti e la lista delle correlazioni tra entità. Il sistema descritto in questo lavoro di tesi riformatta i dati raccolti per essere meglio interpretabili, eliminando alcuni dati salvati come lo span, lo span lemmatizzato, l'etichetta e il range all'interno della frase in cui è presente l'entità ed aggiungendo altri dati come l'istruzione in cui compare e le clausole che modificano il significato del verbo.

5.5 Conversione dei nomi e delle azioni del testo

5.5.1 Ricerca dei sinonimi

Il sistema qui descritto per la generazione del video animato tramite gli oggetti 3D sfrutta un set di animazione pre caricate, queste animazioni coinvolgono dei modelli e sono identificate tramite dei nomi. Il sistema al fine di identificare a quale di queste azioni e modelli l'istruzione si riferisca esegue un confronto tra i nomi presenti all'interno di ogni istruzione, nomi che identificano i componenti e le azioni, con i nomi dei modelli e delle animazioni al fine di convertire i nomi presenti nell'istruzione in nomi presenti su Blender.
Diverse stampanti, di diverse case produttrici e modelli, infatti possono riferirsi ad un determinato componente con altri nomi, e possono riferirsi all'azione da eseguire in altri termini. Il sistema quindi reperisce i sinonimi per queste parole, in modo da "convertire" le azioni e i nomi, ricevuti in input ed identificati durante la fase di identificazione delle entità e delle relazioni, in azioni e nomi utilizzabili durante la fase di generazione del video animato.
Lo scopo della ricerca dei sinonimi, all'interno del sistema descritto in questo lavoro di tesi, è quello di ottenere tutti i possibili nomi di un componente o di una azione al fine di aumentare le possibilità di riscontrare un elevata somiglianza tra il nome rilevato, o un suo sinonimo, e il nome presente all'interno dell'ambiente di animazione, o un suo sinonimo.
Per la fase finale dell'analisi del linguaggio naturale passato come input, il sistema ricerca dei sinonimi per i nomi dei componenti e delle azioni tramite WordNet e tramite un dizionario contenenti i sinonimi riguardanti il contesto applicativo delle stampanti (Figura 5.1).
WordNet non funziona bene per la ricerca dei sinonimi per i nomi dei componenti quando sono nomi composti a causa dell'elevato tecnicismo di questi nomi che non hanno riscontro all'interno del database, i nomi dei componenti vengono quindi elaborati diversamente al fine di ottenere dei plausibili sinonimi.
Nel caso dei nomi dei componenti il sistema scompone la parola composta nelle sue singole parole ed esegue la ricerca dei sinonimi per queste parole, una volta ottenuta la lista di tutti i sinonimi il sistema procede alla concatenazione dei

lista sinonimi di contesto	
parola	sinonimi
light	[power, button]
power	[light, button]
button	[power, light]
printer	[light, button, power,front]
front	[printer]
paper	[letter, document, sheet]
sheet	[paper, document, letter]
letter	[paper, document,sheet]
document	[paper,sheet,letter]
cover	tray
tray	cover

Figura 5.1. Tabella in cui vengono elencati i sinonimi per il contesto applicativo delle stampanti aggiunte manualmente

sinonimi al fine di ottenere una parola composta, con la stessa struttura della parola di partenza, rappresentante un plausibile sinonimo della parola iniziale, ad esempio la parola "paper tray" viene scomposta in "paper" e "tray", per ognuna di essere si ricercano dei sinonimi ottenendo per esempio "newspaper" e "drawer", le parole vengono combinate per ottenere possibili sinonimi della parola iniziale come "newspaper tray", "paper drawer" e "newspaper drawer". Per quanto riguarda invece i nomi delle azioni per ogni azione confrontata, sia quelle provenienti dalle relazioni che quelli provenienti dall'ambiente virtuale di animazione, si ottiene la lista dei sinonimi e il sistema memorizza la versione lemmatizzata del sinonimo. Per entrambe le liste ogni sinonimo viene legato al nome dell'azione da cui deriva tramite un dizionario, questo procedimento rende semplice determinare la parola di partenza a cui il sinonimo si riferisce.
Le animazioni all'interno della libreria di animazioni sono memorizzate con un nome in uno specifico formato, ovvero il nome del componente seguito dalla azione che sta eseguendo separate da il carattere "_", ad esempio "front cover_open".
Il formato rende più facile le operazioni durante la fase di conversione, infatti, prima di eseguire il confronto con ogni azione memorizzata all'interno delle relazioni ricavate dall'analisi del testo, il nome dell'animazione proveniente da Blender, viene diviso nel nome del componente e nel nome dell'azione associata, in questo modo si evita di dover eseguire delle operazioni per associare questi due elementi successivamente.
Il sistema confronta ogni nome del componente presente all'interno delle relazioni, e i relativi sinonimi, con il nome dei modelli e i loro sinonimi. Nel caso in cui due nomi abbiano una percentuale di similarità superiore al 70% il sistema esegue la stessa operazione di confronto anche sull'azione e su i suoi sinonimi. Poiché wordNet riesce a generare più efficacemente sinonimi sulle azioni piuttosto che sui

nomi dei componenti, per questo confronto la percentuale di similarità richiesta come soglia sale al'80% in modo da avere una conversione più accurata. Il sistema per effettuare il confronto, calcola la percentuale di similarità tra i nomi tramite word2vec nella pipelinne di spaCy congiuntamente con il modello pre-addestrato con il set di dati di Google News di Gensim. Quest'ultimo modello, contenente 3 milioni di parole e frasi, contiene parole in ambito generico, non tratta quindi parole specifiche nel campo delle stampanti, come i suoi componenti. Il calcolo di similarità effettuata tramite Gensim quindi fallisce nel caso in cui le parole di cui si cerca un sinonimo non siano presenti all'interno del suo set di parole, per questo motivo in questo sistema, in aggiunta a Gensim si utilizza il modello pre-addestrato di spacy. Il modello pre-addestrato di spacy, presente nella pipeline "en_core_web_md" contiene 20 mila parole e a differenza del modello di Gensim, basa il suo confronto anche sulle caratteristiche della parola, come ad esempio la parte del discorso e le dipendenze, e sul suo ruolo all'interno della frase, ovvero il contesto.
I valori delle soglie qui citati sono stati ottenuti tramite l'osservazione dei risultati ottenuti dal confronto con molteplici test. (Figura 5.2).

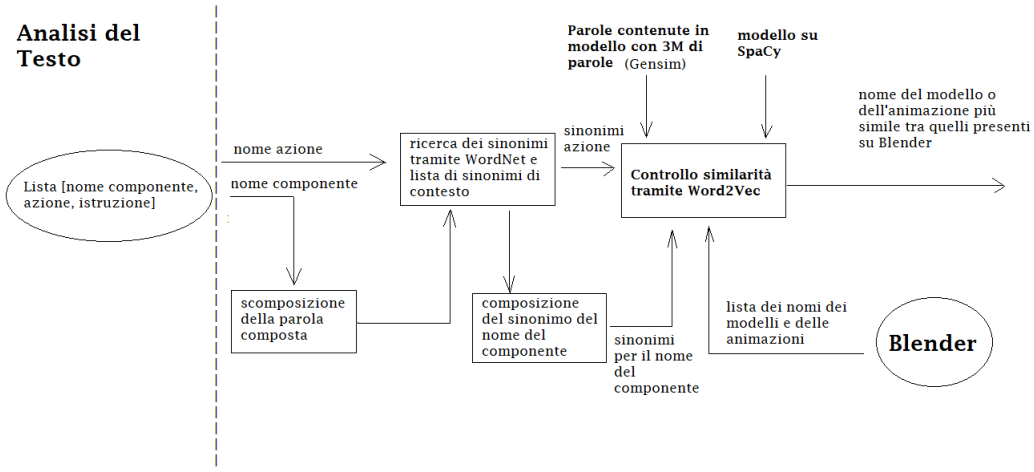

Figura 5.2. Architettura del metodo per la conversione di azioni e nomi di componenti provenienti dall'analisi del testo (sinistra) in nome di modelli e animazioni riconoscibili da Blender (destra), all'interno dei contenitori rettangolari è presente l'operazione eseguita dal sistema

Prima di concludere questa fase il sistema elimina dalla lista delle relazioni, le istruzioni contenenti delle sotto-istruzioni. Le sotto-istruzioni contengono la stessa procedura descritta dall'istruzione principale, ma in questo caso i passaggi vengono ulteriormente suddivisi per aumentare la comprensibilità dell'operazione.

Per questa ragione, l'istruzione genitore sarebbe solamente una ripetizione delle istruzioni figlio ma più generica, e quindi viene eliminata dal sistema, ad esempio in questa sequenza di istruzioni:

1. "Rimuovi la cartuccia dal pacchetto"
    - 1 "apri il pacchetto"
    - 2 "estrai la cartuccia dal pacchetto"

le istruzioni "apri il pacchetto" e "estrai la cartuccia dal pacchetto" risultano essere sotto-istruzioni di "Rimuovi la cartuccia dal pacchetto" che descrivono esplicitamente e più in dettaglio l'operazione da eseguire. Il sistema quindi non considera l'istruzione "Rimuovi la cartuccia dal pacchetto" come da animare ma le sue due sotto istruzioni, in quanto plausibilmente possono generare un'animazione più accurata dell'istruzione descritta.

Capitolo 6

Analisi delle immagini

Questo capito tratta la fase di analisi delle immagini, dove il sistema partendo dai set distinti di immagini, il primo per contenere le immagini utili alla fase di calibrazione e il secondo con le immagini riguardanti le istruzioni da animare, elabora tutte le immagini ricevute restituendo in output il nome dell'oggetto riconosciuto associato ai vertici che ne definiscono la forma e il numero dell'istruzione in cui compare, numero che corrisponde al nome del file.
In questa fase il sistema utilizza un modello addestrato precedentemente di pesi necessario per il riconoscimento automatico dei componenti presenti all'interno di un'immagine. I dati ottenuti dal riconoscimento vengono utilizzati dal sistema per il calcolo di informazioni utili, quali i vertici rappresentanti gli angoli dei componenti e il centro del componente.
I dati ottenuti in questo modo non sono ancora completi, in quanto alcune informazioni, tra cui l'azione da eseguire, non sono estrapolabili dalla sola immagine, il sistema qui descritto utilizzerà quindi il risultato dell'analisi del testo, ovvero la lista delle relazioni, per coprire questi tipi di informazioni ed ottenere in uscita una lista di oggetti rilevati contenenti tutte le informazioni necessarie per la successiva fase di animazione.

6.1 Allenamento del modello

Per la fase di analisi dell'immagini e la conseguente estrapolazione delle informazioni, il sistema descritto in questo lavoro di tesi, si appoggia al framework Mask R-CNN (38).
I modelli pre-addestrati dei pesi, ovvero i modelli precedentemente sviluppati di parametri, all'interno della rete neurale, con la capacità di trasformare i dati in ingresso all'interno dei livelli nascosti della rete, non sono efficienti per l'analisi di immagini riguardanti le stampanti e i suoi componenti.

Le immagini analizzate nel sistema qui descritto, sono specifiche di un determinato campo, di uso non comune, e questo porta ad avere difficoltà nella ricerca di un modello pre-addestrato, in quanto i modelli disponibili pubblicamente e con licenza gratuita, sono ad oggi, addestrati solo per campi di uso comune, come può essere il rilevamento di oggetti in generale (macchine, bottiglie, anche stampanti), ma non dei loro componenti, o il rilevamento di oggetti specifici e dei loro componenti nel caso di alcuni oggetti particolari, come le macchine.
L'utilizzo di uno di questi modelli pre-addestrati, conduce all'ottenimento di scarsi risultati durante l'object detection, all'interno delle immagini di stampanti.
Un altro problema riguarda il fatto che il sistema si propone di identificare le stampanti e i loro componenti in base ad immagini provenienti da manuali di istruzioni. Questo porta a dover lavorare con immagini, in prevalenza, in bianco e nero, e con una qualità inferiore rispetto alle foto di stampanti reperibili online tramite gli appositi siti o all'interno di grandi database come ImageNet (41) o VisualGenome (42). A questo proposito, il sistema qua descritto, utilizza un modello addestrato specificatamente per il rilevamento delle stampanti e dei suoi componenti, basato su un dataset di 240 immagini provenienti da manuali di stampanti di diverse marche e serie.

6.1.1 La configurazione

La configurazione per l'addestramento del modello è:

- IMAGE_PER_GPU = 2, questo parametro permette di lavorare con 2 campioni, in questo caso immagini, alla volta sfruttando la GPU, il parametro influisce implicitamente anche sul batch_size, in quanto batch_size = IMAGE_PER_GPU * # GPU. Il batch_size è il numero di campioni utilizzati per addestrare il modello prima di aggiornare le sue variabili, ovvero i pesi e le distorsioni. In ogni fase di addestramento un batch di campioni viene propagato in avanti e poi indietro per calcolare il gradiente dei campioni, questo la media dei gradienti viene utilizzata per l'aggiornamento dei pesi e delle distorsioni.
- NUM_CLASSES : 34, ovvero il numero di classi su cui il modello si addestra e che saranno successivamente riconoscibili, sono 34 ovvero 1 per il background e 33 per i componenti della stampante (6.1)
- STEPS_PER_EPOCH = 20, ovvero il numero di iterazioni del batch di campioni, prima che un epoca sia considerata completata.
- DETECTION_MIN_CONFIDENCE = 0.9, la percentuale minima per considerare un oggetto, correttamente identificato

- EPOCH = 20, un epoca rappresenta un ciclo di scansione in avanti e indietro attraverso la rete neurale di un intero dataset, il numero qui specificato rappresenta il numero di cicli che vogliamo eseguire per l'addestramento del modello.

La scelta del numero degli steps per epoca è importante per il corretto rilevamento degli oggetti nell'immagine, infatti, poiché il nostro sistema si basa sul rilevamento di 33 classi diverse, 1 solo ciclo di scansione non è sufficiente per la corretta impostazione dei pesi e delle distorsioni all'interno della rete, inoltre il set di dati utilizzato per l'addestramento di questo modello è limitato, questo porta alla necessità della scelta ponderata del numero di epoche (Figura 6.2).

```
# Add classes
self.add_class("object", 1, "Operation Panel")
self.add_class("object", 2, "Platen")
self.add_class("object", 3, "Document Cover")
self.add_class("object", 4, "Paper Guides")
self.add_class("object", 5, "Cassette")
self.add_class("object", 6, "Paper Output Tray")
self.add_class("object", 7, "Paper Output Support")
self.add_class("object", 8, "Output Tray Extension")
self.add_class("object", 9, "Paper Output Cover")
self.add_class("object", 10, "ON button")
self.add_class("object", 11, "ON lamp")
self.add_class("object", 12, "FAX Memory lamp")
self.add_class("object", 13, "Alarm lamp")
self.add_class("object", 14, "ADF")
self.add_class("object", 15, "Document Feeder Cover")
self.add_class("object", 16, "Document Guide")
self.add_class("object", 17, "Document Tray")
self.add_class("object", 18, "Document Output Slot")
self.add_class("object", 19, "Front Cover")
self.add_class("object", 20, "Power Cord Connector")
self.add_class("object", 21, "Rear Cover")
self.add_class("object", 22, "Telephone Line Jack")
self.add_class("object", 23, "External Device Jack")
self.add_class("object", 24, "USB Port")
self.add_class("object", 25, "Transport Unit Cover")
self.add_class("object", 26, "ink cartridges")
self.add_class("object", 27, "Cartridge Holder")
self.add_class("object", 28, "printer")
self.add_class("object", 29, "ink cover")
self.add_class("object", 30, "paper support")
self.add_class("object", 31, "cassette cover")
self.add_class("object", 32, "scanning cover")
self.add_class("object", 33, "document feeder cover")
```

Figura 6.1. Classi impostate per il rilevamento dei componenti della stampante

La scelta del numero di steps per epoca troppo elevato, porta all'overfitting o sovra-campionamento, ovvero ad un'interruzione prematura a seguito del raggiungimento dell'errore minimo prefissato, in modo da evitare che il modello si adatti eccessivamente, ovvero che il modello non apprenda i dati ma li memorizzi.
La scelta del numero di step per epoca troppo basso, porta all'underfitting o sotto-campionamento , ovvero porta alle conseguenze opposte all'overfitting, dove il modello non ha avuto abbastanza cicli per adattarsi che porta alla conseguente scarsa prestazione nel rilevamento degli oggetti nelle immagini.

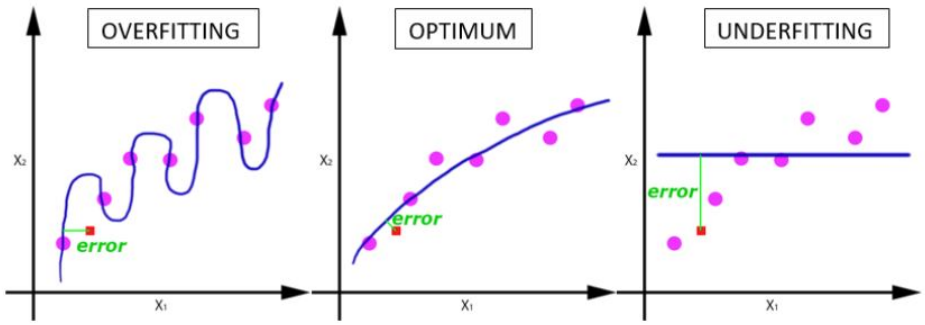

Figura 6.2. Variazione del tasso di errore in relazione ad un caso di overfitting, underfitting e nell'uso del numero di epoche adatto (43)

Per la scelta del numero di cicli corretto, non esiste una formula, il numero varia in base alla diversità dei dati all'interno del dataset, è comunque ricavabile tramite diversi esperimenti, per questo sistema è stato provato l'addestramento con 10 – 20 – 30 - 50 - 100 steps per epoca, mantenendo le rimanenti impostazioni della configurazione inalterate.
Per verificare l'efficacia dell'addestramento sul set di addestramento e su quello di valutazione, in questo lavoro di tesi, si utilizza TensorBoard, un toolkit di TensorFlow (44) per la visualizzazione e la sperimentazione del machine learning. Tensorboard permette, tra le sue molteplici funzionalità, di tracciare e visualizzare alcune metriche come perdita ed accuratezza, quindi questo toolkit è particolarmente utile quando si vuole verificare una situazione di overfitting o underfitting. Nel primo caso avremo un valore crescente di perdita nei cicli riferiti al set di valutazione, associato ad un valore in continuo calo, tendente allo 0, nei cicli riferiti al set di addestramento. (Figura 6.3) Questo è indice di un sistema che memorizza e non si addestra, infatti il valore in aumento della perdita riferito al set di valutazione, indica che il modello non riesce a rilevare correttamente i componenti all'interno dell'immagine, componenti che sono stati precedentemente annotati.

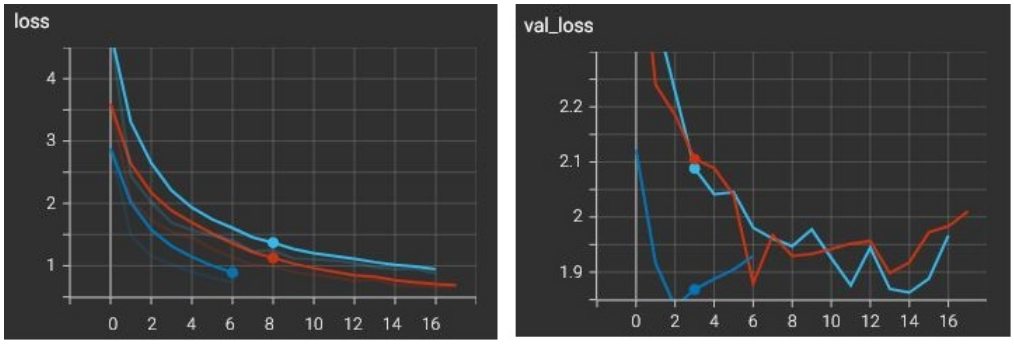

Figura 6.3. A sinistra il grafico della perdita riferito al set di addestramento, che è in continuo calo e tendente allo 0, a destra il grafico della perdita riferito al set di valutazione che dopo un iniziale fase di decrescita inizia a crescere, segnando l'inizio dello stato di overfitting, le tre linee rappresentano i modelli addestrati con 100 (blu) 50 (rosso) 30 (azzurrino) steps per epoca

Nel secondo caso, i valori di perdita riferiti ad entrambi i set, addestramento e validazione, rimarranno alti in quanto il sistema non ha abbastanza tempo per addestrarsi, e ne consegue che il riconoscimento dei componenti all'interno dell'immagine abbia una scarsa accuratezza.
La scelta della dimensione del batch di campioni dipende dalla quantità di GPU disponibile e al numero di GPU utilizzabili, questa scelta influisce criticamente sulla convergenza del processo di addestramento del modello, nonché sulla sua relativa accuratezza. La scelta di una dimensione grande porta ad una cattiva generalizzazione, ovvero la rete neurale, avrà delle prestazioni non accettabili nel rilevamento degli oggetti non appartenenti al training set, e del set di immagini utilizzato per l'addestramento del modello. La scelta di una dimensione piccola porta invece ad una lenta convergenza dell'algoritmo di apprendimento. Poiché durante l'addestramento i campioni verranno scelti casualmente dal training set, se il numero di campioni è basso, la stima del gradiente necessario per il calcolo dei pesi e delle distorsioni sarà rumoroso e poco accurato.
Poiché il sistema qui trattato, è stato sviluppato su una macchina che ha 8GB di RAM per la GPU e poiché il trainig set è limitato, per questo sistema è stato scelto di utilizzare una dimensione piccola, che fosse quindi gestibile dal calcolatore, e che portasse ad una accuratezza accettabile a discapito del tempo necessario per l'addestramento. Le configurazioni della dimensione del batch provate sono 1-2-3, mantenendo le altre impostazioni inalterate.

6.1.2 Annotazione delle immagini

Per l'annotazione delle immagini che costituiscono il training set e il validation set, ovvero il set di immagini utilizzati per la fase di validazione del database per il sistema qui descritto, è stato quindi utilizzato VGG Image Annotator (VIA) (45), un software autonomo, leggero e offline, per l'annotazione manuale delle immagini. La scelta è ricaduta su questo prodotto, in quanto facile da utilizzare, disponibile per uso gratuito online senza la necessità di installare software sulla propria macchina, ed inoltre questo software online dispone di dettagliati tutorial su l'utilizzo, in associazione con mask R CNN.
Il set di dati utilizzato per effettuarne l'annotazione con questo software, include 240 immagini di stampanti, provenienti da manuali di stampanti di marca Canon per il training set, e 50 immagini di stampanti, provenienti da manuali di altre marche (Epson e HP), non presenti all'interno del training set, per il validation set. Il software ha un interfaccia grafica intuitiva che permette di selezionare la forma delle regioni, in modo da facilitare il compito di selezione della sagoma del componente, permettendo di scegliere la forma che più si adatta agli oggetti da annotare.
È inoltre presente una sezione per l'inserimento di tutte le tipologie di attributi, ovvero le classi utilizzate all'interno del sistema per la generazione del modello (Figura 6.4). Il software assegna ad ogni attributo specificato un id univoco, che identificherà la classe di appartenenza di un componente. A seguito della selezione dei contorni di un componente, sarà infatti necessario attribuire alle forma generate, l'attributo corrispondente al componente in esame.

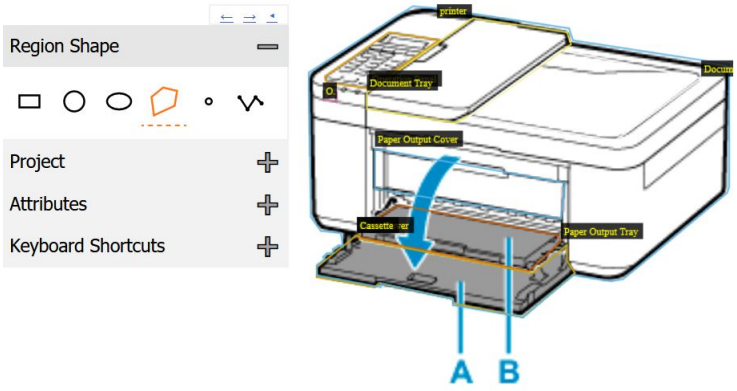

Figura 6.4. interfaccia di VIA, a SX le tipologia di regioni, il progetto e gli attributi, a DX l'immagine da annotare, con presenti alcune annotazioni realizzate manualmente

Le annotazioni realizzate tramite il software, verranno infine esportate in formato JSON. L'annotatore VIA salverà le immagini associando al nome del file, le regioni presenti all'interno dell'immagine, elencando i nomi degli attributi che rappresentano la determinata regione, e la forma della regione selezionata, ovvero tutti i punti x e tutti i punti y presenti nei contorni delle regioni, rappresentanti i punti selezionati manualmente tramite software, ed infine la dimensione dell'immagine (6.1).

```
, [‘filename’:’1.jpg’],
        ‘regions’:{
                ‘0’:{
                        ‘region_attributes’:{
                                ‘Paper Output Cover’,
                                ‘Document Tray’,
                        }
                        ‘shape_attributes’:{
                                ‘all_points_x’:{
                                01,
                                05,
                                96,
                                ..., }
                                ‘all_points_y’:{
                                16,
                                11,
                                12,
                                ..., }
                                ‘name’: ‘ polygon’
                        }
                }
            altre regioni ...
    ‘size’: 211194
        }
```

Tabella 6.1. Formato dei dati salvati tramite VIA, riguardanti l'immagine e le regione in essa annotate

6.1.3 Generazione dei pesi

L'ultima fase riguarda l'utilizzo delle annotazioni precedentemente realizzate, al fine di generare i pesi utili al corretto rilevamento degli oggetti, all'interno delle immagini che il sistema riceverà in ingresso.
Per la generazione dei pesi, il sistema partiva inizialmente con dei pesi provenienti dal set COCO e quindi non pertinenti con il nostro caso di studio,per poi utilizzare a seguire, per la realizzazione dei modelli successivi, i pesi precedentemente realizzati. Il sistema dopo aver caricato i dataset, training set e validation set, recupera le annotazioni eseguite tramite il software VIA, per associarle alle immagini.
Il sistema utilizza il framework Mask RCNN per la generazione dei pesi, ottenendo in questo modo un file per ogni epoca, contenente i pesi generati fino a quel momento. Questo approccio permette di poter rincominciare l'addestramento del modello dall'epoca in cui il sistema si ferma, in caso di malfunzionamenti o altre cause che portano all'interruzione del processo di addestramento. Il processo si interrompe nel momento in cui l'ultima epoca viene scansionata, e il relativo file viene generato. Questo file verrà utilizzato al momento del processo di rilevamento dei componenti all'interno dell'immagine.

6.2 Rilevamento dei componenti

La fase di rilevamento dei componenti, come dice il nome stesso, consiste nel riconoscere i componenti della stampante all'interno delle immagini che il sistema riceve in ingresso, utilizzando il modello pre addestrato dei pesi generato precedentemente. Il rilevamento degli oggetti, chiamato in inglese object detection, è una tecnica di computer vision, utilizzata per identificare e localizzare un determinato oggetto all'interno di un immagine o di un flusso video.
Nello specifico l'object detection disegna una regione di delimitazione, o bounding box in inglese, attorno all'oggetto, con lo scopo di localizzarlo all'interno della scena fornita. Il rilevamento degli oggetti viene spesso confuso con il riconoscimento degli oggetti, la differenza sostanziale tra le due tecniche consiste nel fatto che, il riconoscimento delle immagini attribuisce un etichetta ad un immagine, un immagine di una stampante riceve l'etichetta "printer" così come un immagine con due o più stampanti, mentre il rilevamento degli oggetti fornisce oltre a questo dato un bounding box che localizza l'oggetto all'interno della scena, e attribuisce l'etichetta a quel bounding box e non all'intera scena (Figura 6.5).
Nella image recognition non riceviamo quindi nessuna informazione riguardante la posizione della stampante o delle stampanti all'interno della scena, ma otteniamo solamente l'informazione riguardante la loro presenza all'interno di essa. L'object detection all'interno del framework Mask R-CNN è eseguita tramite le tecniche

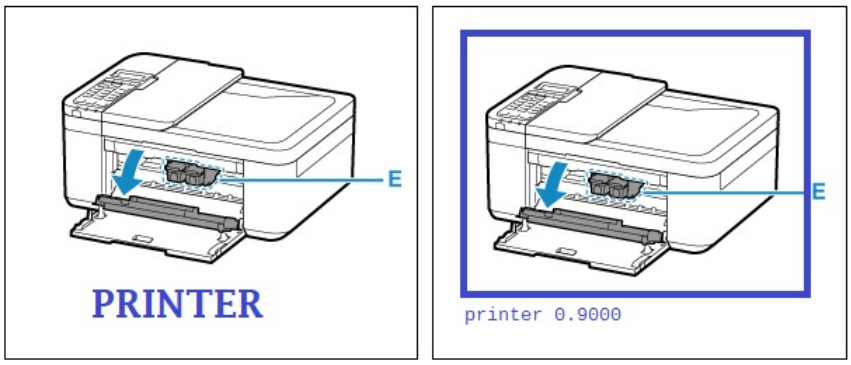

Figura 6.5. A sinistra il risultato della image recognition, a destra il risultato delle object detection, eseguiti sull'immagine di una stampante

di image segmentation, o segmentazione dell'immagine, e instance segmentation, o segmentazione dell'istanza. La tecnica di image segmentation, identifica la presenza di una determinata istanza di una classe, ad esempio "printer", all'interno della scena, se all'interno della scena sono presenti più stampanti, questa tecnica li salverà come un unica maschera associata alla classe "printer", e non sarà quindi possibile distinguere un istanza della classe "printer" da un altra (Figura 6.6).

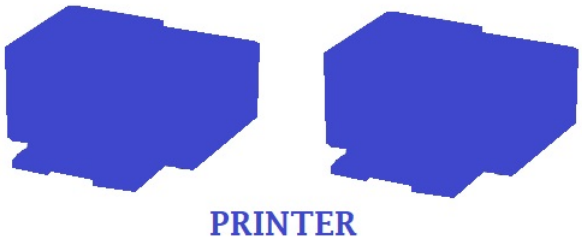

Figura 6.6. Esempio di image segmentation con 2 stampanti presenti contemporaneamente nella scena

L'object detection una volta ottenuta la maschera in uscita dalla fase di image segmentation, esegue l'operazione di instance segmentation al fine di ottenere le maschere appartenenti ad ogni istanza della classe, all'interno della scena, nel caso di 2 stampanti le due maschere saranno quindi separate, ed associate ad una etichetta che richiama la classe di appartenenza e il numero dell'istanza (Figura 6.7). L'operazione di instance segmentation identificherà, quindi, ogni oggetto presente all'interno della scena, e assocerà ad ognuno di essi, la maschera corrispondente.
A partire dalle maschere generate dal processo di instance segmentation, verranno

infine calcolate dal sistema, i bounding boxper ovvero ogni oggetto presente nella scena, associandoli alle maschere, all'etichetta e al nome del file in sui gli oggetti sono stati rilevati.
Il sistema, come prima operazione, crea il modello in "inference mode", ovvero il modello processa i punti ricevuti in tempo reale in un algoritmo di apprendimento automatico, per ottenere un output, come un punteggio numerico che non contribuisce all'aggiornamento dei pesi e dei gradienti, come nel caso del "training mode" utilizzato durante la fase di addestramento, ma che riguarda le prestazioni del modello posto in stato di "produzione".

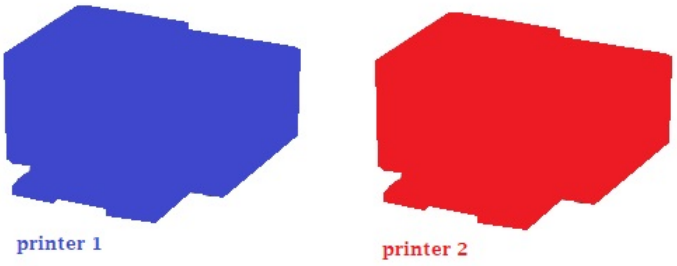

Figura 6.7. Esempio di instance segmentation con 2 stampanti all'interno della scena, ad ogni maschera viene associata un etichetta composta dal nome della classe più il numero dell'istanza, ovvero "printer 1" e "printer 2"

Una che il sistema carica il modello e i pesi generati nella fase precedente, procede con la fase di detection degli oggetti, all'interno di ogni immagine che riceve in ingresso (Figura 6.8).

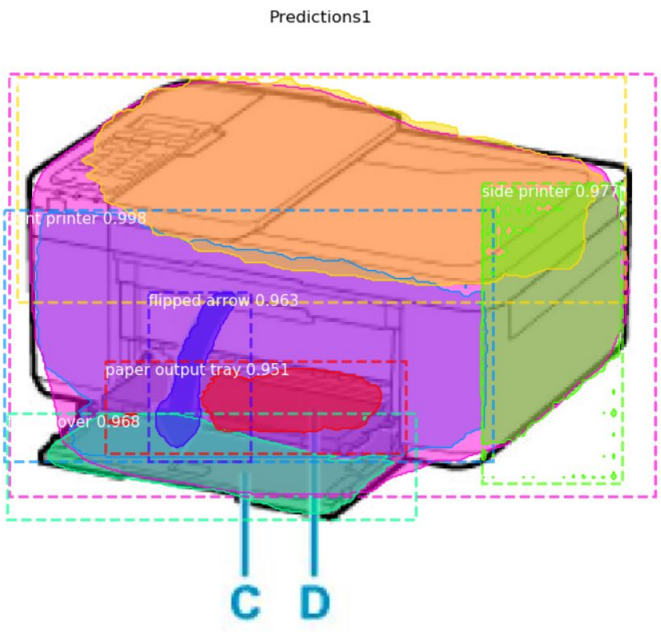

Figura 6.8. Esempio visivo del risultato di detection dei componenti all'interno dell'immagine di una stampante, ad ogni ROI, raffigurato da un rettangolo, è associato il nome e il grado di affidabilità, il ROI oltre a localizzare l'oggetto inoltre delimita la maschera dello stesso

Per questa operazione si sfrutta il metodo detect() che utilizza la pipeline di detection del framework Mask R-CNN, il metodo riceve in ingresso una lista di immagini, la cui dimensione non influisce sul processo, e restituisce in output una lista di dizionari, dove ognuno di essi è associato ad un'immagine presente all'interno della lista passata in ingresso. I dizionari generati contengono:

- Le regioni di interesse, o regions of interest (ROI), di tutti i componenti rilevati all'interno dell'immagine, il ROI è un poligono composto da 4 vertici che contiene al suo interno il componente, ed è rappresentato tramite una lista di 4 punti, [y1,x1,y2,x2] riferiti a 2 dei suoi vertici e necessari per la rappresentazione del poligono.

- L'ID della classe, ovvero il numero identificativo che rappresenta la classe del componente riconosciuto. L'id identifica la classe all'interno di una lista precedentemente caricata con le coppie [id: 'nome classe'], e sarà successivamente utilizzato per la traduzione nel nome corrispettivo.

- I punteggi, o scores, ovvero la probabilità che gli oggetti identificati corrispondano effettivamente all'id attribuitogli

- La maschera, ovvero una matrice N x M della stessa dimensione dell'immagine corrispondente, contenente i valori booleani False e True, dove True rappresenta un pixel in cui il componente è stato riconosciuto all'interno dell'immagine (Figura 6.9)

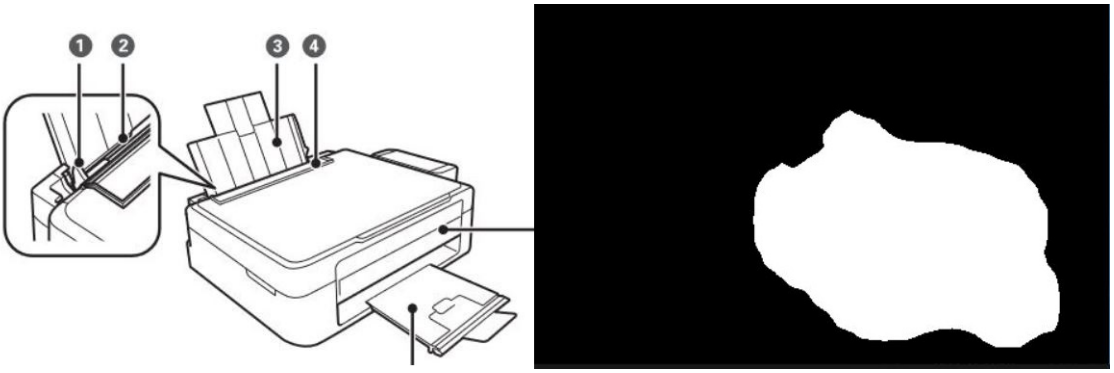

Figura 6.9. A sinistra l'immagine ricevuta in ingresso dalla funzione detect di MASK RCNN, a destra la maschera risultante, associata all'oggetto "printer"

6.3 Calcolo degli angoli dei componenti

Una volta completata la fase di detection, il sistema possiede tutte le informazioni, ovvero il ROI e la maschera di ogni componente rilevato, necessarie al calcolo degli angoli dei componenti, all'interno di una determinata immagine. Questo calcolo viene eseguito in due fasi:

- La prima consiste nell'utilizzare metodi noti, ovvero il metodo Canny seguito dal metodo goodFeaturesToTrack per le stampanti e il metodo cornerHarris per il resto dei componenti, per individuare dei possibili angoli all'interno dell'immagine

- La seconda consiste nella raffinazione dei risultati ottenuti dal precedente metodo, tramite un metodo sviluppato appositamente per il sistema descritto in questa tesi di laurea, in modo da ottenere un quartetto di vertici rappresentanti il componente

6.3.1 Corner Detection

La fase di detection dei componenti presenti all'interno dell'immagine i-esima, rappresentante la i-esima istruzione del manuale, fornisce al sistema i vertici rappresentanti la ROI del componente e una matrice di valori booleani, False e True, rappresentanti la maschera che localizza il componente. Il sistema descritto in questo lavoro di tesi, utilizza le coordinate x e y, che rappresentano la ROI, o bounding box, dell'immagine per eseguire un ritaglio della maschera associata al componente rilevato all'interno dell'immagine di partenza (Figura 6.10).

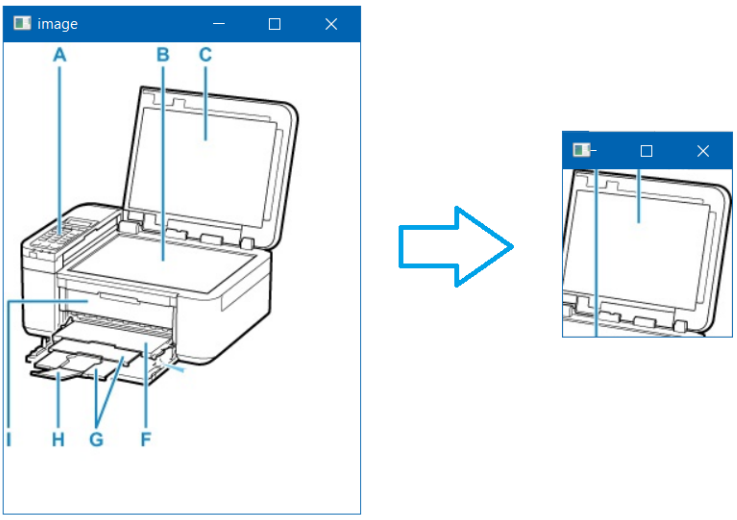

Figura 6.10. Esempio di ritaglio dell'immagine al solo componente identificato dal bounding box dell'oggetto in esame

Il ritaglio viene eseguito selezionando i pixel dell'immagine compresi tra le due coordinate x e le due coordinate y, che identificano il bounding box. Il ritaglio dell'immagine è una procedura necessaria in quanto per la ricerca dei punti che rappresentano il contorno dell'oggetto, si utilizza il metodo di rilevamento degli angoli noto come Harris Corner Detection, nel caso dei componenti dell'oggetto o i metodi Canny ed goodFeaturesToTrack nel caso della stampante.

Il metodo Canny

Il metodo di rilevamento dei bordi di Canny, è un algoritmo composto da più stage di elaborazione che riceve in ingresso un'immagine in scala grigi, che all'interno del sistema descritto in questo lavoro di testi viene anche sfocata in modo da ridurre la variazione del gradiente nei bordi e ridurre in questo modo il numero di punti

che il metodo rileva lungo i lati dell'oggetto.
Gli stage di elaborazione che vengono eseguiti dal metodo sono:

- Riduzione del rumore tramite un filtro gaussiano 5x5
- Ricerca dei gradienti di intensità all'interno dell'immagine, tramite un filtro Sobel verticale ed orizzontatale
- Filtro dei valori non massimi locali, ovvero all'interno di un'area locale il metodo elimina quei pixel che non rappresentano il massimo locale
- Applicazione di una soglia di isteresi tramite due valori, ovvero il gradiente minimo e massimo accettati come possibili pixel di bordo

Il metodo goodFeaturesToTrack

Il metodo goodFeaturesToTrack di OpenCV, è un metodo che rileva i bordi "forti",ovvero gli angoli più prominenti in un immagine o in una regione di interesse. Il sistema descritto in questo lavoro di tesi utilizza questo metodo specificando la necessità di utilizzare il metodo cornerHarris al suo interno, piuttosto che il metodo cornerMinEigenVal per la misura della qualità dell'angolo per ogni pixel. Il metodo esegue anche in questo caso una serie di step per l'elaborazione:

- la soppressione dei valori non massimi locali nell'intorno 3x3
- l'applicazione di una soglia per gli autovalori, rispetto a due valori minimo e massimo
- riordinamento degli angoli rispetto alla loro qualità
- eliminazione degli angoli la qui qualità non è massimo in un distanza specifica "distance"

Il metodo CornerHarris

Il metodo cornerHarris, sfrutta, per il rilevamento degli angoli, la conoscenza del fatto che gli angoli sono regioni dell'immagine in cui sono presenti grandi variazioni di intensità in tutte le direzioni, ovvero saranno zone dell'immagine in cui sarà presente una varianza del gradiente elevata in tutte le direzioni.
Il metodo calcolerà, per ogni pixel, la covarianza del gradiente, in una finestra 2x2, sposterà questa finestra nei blocchi vicini, all'interno di una finestra NxN e identificherà gli angoli come quei pixel che produrranno una variazione del gradiente significativa in ogni direzione.
Il sistema qui descritto, ricava i possibili vertici dell'oggetto, tramite il metodo,

fornito dalla libreria openCV, cornerHarris, questo metodo riceve come parametro un'immagine, nel nostro caso l'immagine di partenza, ritagliata, convertita in scala di grigi e leggermente sfocata, al fine di smussare le variazioni di gradiente e ridurre in questo modo i falsi positivi.
L'immagine ha la necessità di essere ritagliata in modo da limitare l'area di applicazione del metodo, alla sola zona dell'immagine raffigurante il componente da noi preso in esame.
Il metodo fornisce in uscita dei valori del gradiente, convertibili in una matrice di punti, rappresentanti i "presunti" angoli del componente rappresentato all'interno dell'immagine passata.

6.3.2 Raffinazione dei vertici e calcolo degli angoli

La lista di vertici ottenuti contiene numerosi punti, a causa della complessità dell'immagine passata e all'impossibilità di eseguire un ritaglio preciso che isoli il componente dal resto della stampante e non tutti i vertici all'interno di essa rappresentano gli angoli effettivi del componente. É necessario quindi un'elaborazione della lista dei vertici ottenuta atta ad effettuare una scrematura col fine di ottenere i soli 4 vertici che plausibilmente rappresentano gli angoli che definiscono la forma del componente.
Per prima cosa il sistema elimina dalla lista dei possibili angoli tutti i pixel che non sono presenti all'interno della maschera del componente attraverso un operazione di AND bit to bit, tra la matrice della maschera e la matrice ricavata dai dati in uscita del metodo di rilevamento degli angoli.
Per questo scopo è necessario eseguire il ritaglio anche della maschera, in modo da ottenere una matrice della stessa dimensione di quella derivante dalla ricerca degli angoli effettuata sull'immagine ritagliata.
Durante l'esplorazione della maschera per l'operazione di AND bit to bit viene anche calcolata la dimensione della maschera stessa, ovvero viene calcolata l'ipotetica dimensione in termini di numero di pixel del componente costituente la maschera derivante dal rilevamento degli oggetti.
Prima di effettuare la ricerca degli angoli all'interno della lista dei vertici filtrati, i vertici vengono traslati in modo da rappresentare correttamente il punto all'interno dell'immagine di partenza e non all'interno dell'immagine ritagliata, per eseguire questa procedura vengono semplicemente sommati ad ogni vertice la rispettiva componente x e y, ricavata nuovamente dalla regione di interesse (ROI), associata al componente in esame.
Queste procedure rappresentano le operazioni preliminari necessarie al corretto funzionamento dell'algoritmo di individuazione degli angoli definito per il sistema

descritto in questo lavoro di tesi.

La prima operazione eseguita dal metodo per il calcolo approssimativo degli angoli è quella di estrapolare i valori massimi per entrambe le coordinate, x e y, dalla lista dei punti. I valori ottenuti permettono di generare i punti ideali, chiamati d'ora in avanti punti vincolanti, nelle quali vicinanze si dovrebbero trovare gli angoli del componente (Figura 6.11).

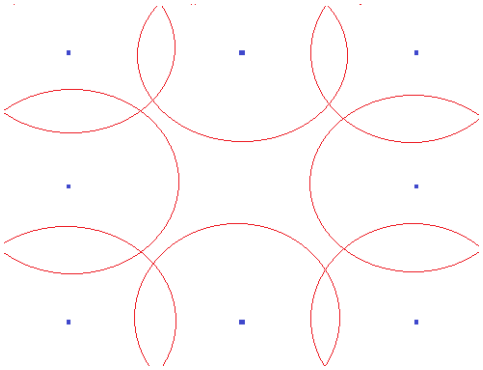

Figura 6.11. I punti blu rappresentano i punti vincolanti, le aree delimitate da un cerchio rosso, rappresentano l'area di ricerca di angoli rispetto ad un determinato punto vincolante

Nell'output dei metodi di individuazione degli angoli, i punti sono raggruppati in località ristrette o sono presenti lungo i bordi dell'oggetto identificato.
All'interno delle località di punti, nel migliore dei casi, solo uno dei punti rappresenta un ipotetico angolo, nel peggiore dei casi nessuno di quei punti,è quindi necessario eliminare tutti i punti spuri non utili all'identificazione dell'angolo che rappresentano una ripetizione dello stesso punto traslato di qualche pixel.
Vengono quindi selezionati unicamente i punti più vicini ai punti vincolanti che rispettino la seguente regola: un nuovo punto aggiunto alla lista dei possibili angoli non può essere eccessivamente vicino, in proporzione alla grandezza del roi dell'immagine, ai possibili angoli precedentemente identificati.
Questa ulteriore regola ha lo scopo di evitare che i possibili angoli si addensino o addirittura che un punto venga considerato come possibile angolo per più di una volta.Si ottiene in questo modo una lista di 8 ipotetici angoli identificati dalle loro coordinate, x e y, in associazione con le rispettive distanze.
Una volta ottenuto la lista degli 8 angoli per la selezione degli angoli approssimati del componente vengono utilizzate 2 diverse metodologie, una per il riconoscimento dei punti della stampante e una per il riconoscimento dei punti dei diversi componenti che la costituiscono.

Sono necessarie due metodologie diverse in quanto le immagini raffiguranti le stampanti all'interno dei manuali di stampanti hanno delle caratteristiche ripetitive dettate dalla forma tipica delle stampanti e dalla necessità di illustrare un comportamento o i componenti, che può essere sfruttato per ottenere rilevamenti degli angoli più precisi.
Queste caratteristiche, tra cui il fatto che si predilige una visione frontale della stampante per una visualizzazione comoda delle operazioni da eseguire ,pongono dei limiti alle possibili zone dove è plausibile trovare gli angoli.
I componenti della stampante che invece possono avere pressoché ogni forma e qualsiasi orientamento, necessitano di una metodologia più generale in modo da evitare, quanto più possibile, l'identificazione di angoli completamente erronei.
La metodologia riguardante la stampante limita inizialmente la scelta dei possibili punti a quelli vicini a 4 punti vincolanti piuttosto che ad 8,precisamente ai 4 punti vincolanti corrispondenti ai punti diagonali (Nord-Est, Nord-Ovest, Sud-Est, Sud-Ovest)(Figura 6.12) La differenza rispetto all'utilizzo di 8 punti consiste nel fatto

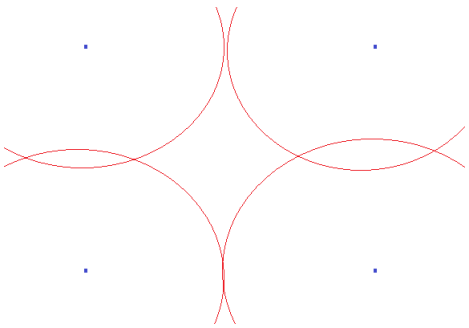

Figura 6.12. Esempio di utilizzo di 4 zone al posto di 8, i restanti 4 punti vincolanti vengono ignorati. I punti blu rappresentano i punti vincolanti, le aree delimitate da un cerchio rosso, rappresentano l'area di ricerca di angoli rispetto ad un determinato punto vincolante

che, utilizzando 4 punti vincolanti, le distanze verranno calcolate solamente rispetto a questi 4 punti ignorando gli altri. Questa limitazione è attuabile solamente poiché si suppone che i punti rappresentanti gli angoli effettivi della forma della stampante passata non possano trovarsi eccessivamente lontani rispetto ai 4 punti vincolanti e la presenza degli altri 4 punti potrebbe portare alla individuazione di punti intermedi più vicini rispetto agli angoli effettivi che rappresentino un bordo.
La seconda metodologia, quella riguardante ogni componente della stampante è simile alla precedente metodologia ma in questo caso le zone associate ai punti vincolanti, in cui viene effettuata la ricerca, sono tutte e 8, anche in questo caso si ottengono 4 zone di ricerca maggiori composte però da 3 zone di ricerca ciascuna piuttosto che una sola come nel caso precedente (Figura 6.13). In questo

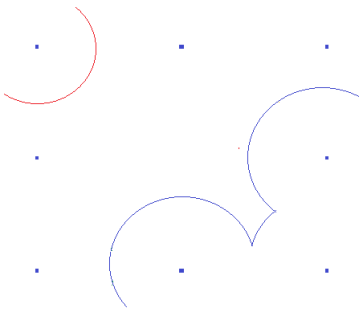

Figura 6.13. Illustrazione della differenza di una zona composta da 1 micro-zona (in alto a sinistra in rosso) rispetto ad una zona composta da 3 micro-zone (in basso a destra in blu)

caso infatti il componente può avere qualsiasi forma ed orientamento in quanto non ci sono regole generali per quanto riguarda la sua posizione all'interno della stampante.
Durante la ricerca dei possibili angoli per entrambe le metodologie è possibile che il sistema identifichi dei punti contesi tra due zone, in questo caso, il sistema, per ogni punto conteso ricerca un possibile sostituto per entrambe le zone che lo contendono. Durante la ricerca del sostituto, il sistema, controlla inizialmente il prossimo punto più vicino per ognuna delle zone, quindi nel caso della stampante la distanza sarà calcolata rispetto al punto vincolante rappresentante il centro della singola zona di ricerca, mentre nel caso dei componenti si prenderà la minore tra le distanze calcolate tra i sostituti del punto e i 3 rispettivi punti vincolanti che rappresentano le zone di ricerca componente la zona di ricerca maggiore.
I sostituti del punto saranno selezionati tra quelli presenti all'interno della lista degli 8 possibili angoli. Successivamente il sistema si assicura ricorsivamente che quell'angolo non sia stato già individuato e memorizzato da un'altra zona di ricerca.
A questo punto:

- Nel caso in cui la distanza del punto non superi una certa soglia per entrambi i punti trovati, il sistema, mette a confronto le due distanze ottenute, ed assegna alla zona con il sostituto del punto più distante il punto conteso, mentre all'altra zona assegna il proprio sostituto del punto precedentemente trovato.

- Nel caso in cui una zona riesca a trovare un secondo punto mentre l'altra no, il punto conteso viene assegnato a quello che non l'ha trovato, mentre all'altra zona viene assegnato il secondo punto

- Nel caso in cui in nessuna delle due zone si riesca a trovare un secondo punto, il processo di ricerca degli angoli viene interrotto, e il sistema informa l'utente che non è stato possibile trovare degli angoli adeguati a rappresentare l'oggetto

Il processo di ricerca termina, nel caso in cui si riesca a trovare 4 punti differenti per rappresentare la figura, il sistema in questo modo ottiene la lista dei 4 angoli rappresentanti la forma del componente o della stampante (Figura 6.14). Nel caso in cui non si trovino 4 punti il metodo restituisce il valore 0, ad indicare un fallimento nella ricerca degli angoli per la forma del componente o della stampante passata.

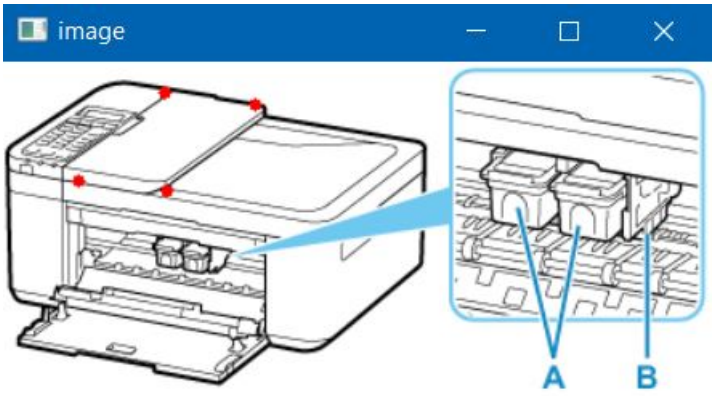

Figura 6.14. Esempio di un output, visualizzato tramite la libreria openCV, di 4 possibili punti individuati come rappresentanti di un componente della stampante

6.4 Memorizzazione dei dati rilevati

La procedura di identificazione e di salvataggio degli angoli rappresentanti i componenti e la stampante viene eseguita su 2 set distinti di immagini, un set con incluse le immagini per la calibrazione, ovvero contenente la posizione di default di tutti i componenti, e 1 set che comprende le immagini raffiguranti la rappresentazione grafica delle istruzioni.
Per quanto riguarda la lista delle posizioni di default, poiché all'interno della cartella con le immagini per la calibrazione possono essere presenti diverse immagini, ed un componente potrebbe essere rilevato in più di una immagine il sistema confronta le dimensioni della maschera del componente rilevato con quello memorizzato precedentemente all'interno della lista e memorizza quello nuovo unicamente nel caso in cui la dimensione dell'area della sua maschera sia maggiore rispetto a quella già presente all'interno della lista. Questa procedura non viene eseguita per le immagini rappresentanti le istruzioni, in questo caso al sistema sono necessarie tutte le posizioni rilevate nelle differenti immagini, fatta eccezione per la posizione

della stampante, che non si dovrebbe poter muovere tra un istruzione ed un altra. Per entrambe le liste ottenute, il sistema memorizza per ogni elemento della lista la posizione, il nome del componente rilevato, la sua ROI, ed il nome dell'immagine in cui sono stati rilevati, nome che corrisponde al numero dell'istruzione.
Il sistema ottenuta la lista dei componenti con i rispettivi angoli che ne definiscono la forma, esegue un'operazione di filtraggio atta a eliminare tutti gli elementi della lista il cui componente rappresentato non è il componente che sta effettuando l'azione descritta dall'istruzione rappresentata dall'immagine in cui sono stati rilevati.
Il filtro viene eseguito confrontando reperendo all'interno delle relazioni derivanti dall'analisi del testo la stessa istruzione a cui il componente si riferisce. Il nome del componente individuato tramite l'analisi dell'immagine viene così confrontato con il nome trovato all'interno del testo, anche in questo caso i nomi possono essere diversi in quanto i nomi individuati dall'analisi delle immagini dipendono dai nomi attribuiti alle classi durante l'addestramento del modello dei pesi e durante l'annotazione delle immagini. Il sistema attua anche in questo caso la stessa procedura di creazione dei sinonimi per i nomi composti tramite wordNet e tramite la lista di sinonimi di contesto, in questo caso però vengono confrontati tramite word2vet solo i due nomi in esame, ovvero il nome del componente derivante dall'analisi dell'immagine (e i suoi sinonimi) e il nome del componente presente all'interno della relazione (e i suoi sinonimi). Nel caso la percentuale di similarità sia inferiore al'80% il sistema elimina l'elemento dalla lista, mantenendo unicamente quei nomi di componente che risultano simili e quindi probabilmente lo stesso componente.

6.5 Completamento e conversione delle informazioni

Il sistema non è capace di ottenere dall'immagine tutti i dati necessari per la successiva fase di animazione, informazioni come l'azione da eseguire non sono estrapolabili. Il sistema esegue quindi un'ulteriore elaborazione dei dati allo scopo di trovare ed inserire queste informazioni mancanti.
Il sistema infine deve convertire l'azione reperita in un'azione facilmente gestibile da Blender per la generazione automatica dell'animazione.

6.5.1 Completamento delle informazioni dal testo

Queste informazioni vengono reperite, dal sistema, all'interno delle relazioni durante l'operazione di filtraggio precedentemente descritta, in questa fase infatti quando un nome proveniente dall'analisi dell'immagine risulta avere una percentuale di similarità superiore al'80% il sistema, oltre a salvare l'elemento, memorizza

l'azione presente nella relazione all'interno dell'elemento all'interno della lista realizzata durante l'analisi dell'immagine. Un ulteriore informazione che può essere memorizzata dal sistema riguarda il "luogo" salvato all'interno della relazione, da questo dato il sistema calcola l'oggetto verso cui l'azione dovrà essere svolta la direzione verso cui si dovrà movere ovvero verso il luogo o lontano dal luogo.

6.5.2 Conversione dei nomi rilevati all'interno dell'immagine

La lista delle azioni all'interno dell'immagine ha un fine differente rispetto alla lista proveniente dall'analisi del testo, in questo caso i nomi delle azioni rilevati serviranno a generare un'animazione e non a trovare corrispondenza con le animazioni di default presenti all'interno di Blender, le azioni devono essere quindi tradotte in animazioni generabili. Le azioni generabili, che in questo sistema vengono tenute in considerazioni, sono le operazioni elementari di sola traslazione e di sola rotazione, non vengono quindi generati comportamenti misti o riguardanti la procedura di scale. Per questo sistema si è preferito questo limite in quanto le istruzioni riguardano operazioni elementari, riguardanti nella maggior parte dei casi, la sola traslazione o la sola rotazione, e l'aggiunta di comportamenti misti avrebbe aggiunto ulteriore complessità al sistema e avrebbe diminuito le sue prestazioni, senza d'altra parte aumentare in modo considerevole l'accuratezza.
Le azioni verranno convertite rispetto alle azioni di "move" e "rotate" ovvero le azioni elementari che si possono eseguire all'interno di Blender. La conversione delle azioni avviene come nei casi precedenti, vengono trovati i sinonimi tramite WordNet sia per le due azioni di "move" e "rotate" che per l'azione in esame e viene calcolato la percentuale di similarità tra le parole e i loro sinonimi, in questo caso non esiste alcuna soglia poiché l'intento è quello di convertire il verbo nella rispettiva azione più simile. La ricerca dei sinonimi viene effettuata tramite wordNet, il sistema per limitare la ricerca ai soli sinonimi interessanti per la generazione dell'animazione, specifica i vari tipi di dominio in cui ricercare i sinonimi, ovvero "geometry" e "mathematics" in quanto le azioni che dobbiamo convertire sono limitate a questi casi. Per il calcolo della similarità ogni sinonimo viene legato tramite un dizionario alla parola di provenienza in modo da non perdere l'informazione riguardante l'azione da convertire facilmente generabile su Blender. (Figura 6.15).

Una volta filtrata la lista dalle istruzioni ripetute, il sistema genera un file JSON da inviare il server contenente:

- La lista delle istruzioni proveniente dall'analisi del testo, gli elementi di tale lista saranno nella seguente forma: ["numero istruzione", "soggetto", "azione","oggetto","clausola","luogo"]

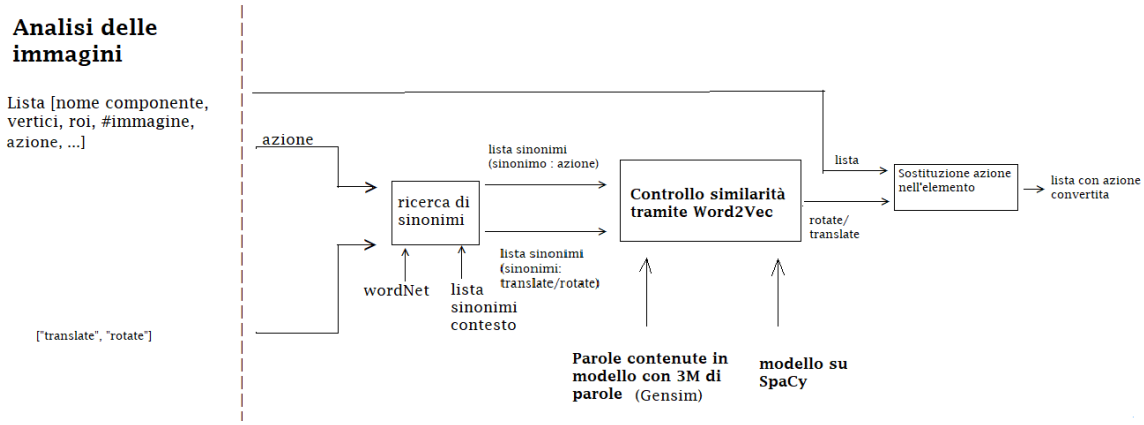

Figura 6.15. Schema della procedura di conversione del nome presente all'interno della lista associata ai dati dell'analisi dell'immagine in un'azione basilare tra rotate e translate

- Lista degli oggetti individuati dall'analisi delle immagini, gli elementi al suo interno saranno nella seguente forma: ["nome del componente", "roi", "vertici forma", "image","place", "center","direction"]
- Lista degli oggetti di default, individuati tramite l'analisi delle immagini, gli elementi di questa lista saranno nella seguente forma: ["nome componente", "roi", "vertici forma", "dimensione","center"]

Capitolo 7

Generazione del video in grafica 3D

Questo capitolo tratta dell'ultima fase del sistema descritto in questo lavoro di tesi, ovvero la fase di generazione del video animato tramite modelli 3D dall'output generato dalla fase di analisi della lista di istruzioni in linguaggio naturale e dall'output generato dalla fase di analisi delle immagini raffiguranti le rappresentazioni grafiche delle istruzioni. Dopo una breve descrizione dell'ambiente virtuale utilizzato, ovvero Blender, il capitolo descrive come avviene la comunicazione quest'ultimo e il resto del sistema al fine di passare le informazioni necessarie per l'elaborazione durante la fase di analisi del linguaggio naturale e delle immagini, e per l'invio dei risultati ottenuti durante queste fasi. Infine il capitolo descrive come il sistema utilizza i dati ricevuti al fine di posizionare correttamente la camera all'interno dell'ambiente virtuale e come genera il video animato finale.

7.1 Blender

7.1.1 Cos'è Blender

Blender (46) è un software open source e multi piattaforma per la modellazione, il rigging, la composizione, il rendering, l'animazione e il montaggio video per oggetti 3D e 2D. Permette l'animazione lineare e non lineare e la creazione di applicazioni 3D. É disponibile per diversi sistemi operativi come Microsoft Windows e GNU/Linux. Blender possiede un robusto set di funzionalità paragonabili, per caratteristica e complessità, ad altri programmi noti per la modellazione 3D come Cinema 4D, 3D Studio Max e Maya. Tra le funzionalità di Blender vi è anche l'utilizzo di script in Python. Blender dispone di diverse tecniche e funzionalità

per la gestione delle animazioni come la cinematica inversa, le armature, la gestione dei fotogrammi chiave, le animazioni non lineari, i vincoli e altro. Queste tecniche e funzionalità unite all'editor per lo scripting in python per automatizzare e controllare i numerosi aspetti del programma e della scena, hanno reso Blender un ottimo strumento, per la realizzazione del video in grafica 3D, tramite l'utilizzo di animazioni dei modelli 3D. Per questo sistema è stata utilizzata la versione 2.92 del software, poiché in questa versione oltre alla risoluzione dei bug della versione precedente, è stata migliorata l'interpolazione dei frame, sono stati aggiunti dei comandi per la creazione rapida di primitive di base ed è possibile un controllo maggiore delle proprietà di alcuni componenti dell'editor video, per esempio è stata aggiunta la possibilità di aggiungere e impostare il colore di sfondo per i sottotitoli. Per quanto riguarda la API python, è stata aggiornata sostituendo, eliminando o modificando alcuni metodi già presenti, obsoleti o non performanti.

7.1.2 Operazioni Preliminari

Il software Blender, viene utilizzato per la fase di generazione delle animazioni. Questo software a partire da oggetti 3D ha la possibilità di creare delle animazioni sfruttando alcune delle sue funzionalità e dei suoi strumenti. Il sistema in questo caso, sfrutta il metodo basato sulla gestione dei fotogrammi chiave per la generazione delle animazioni a partire dalle immagini, utilizza l'editor NLA per la concatenazione delle azioni. È necessario, prima che il software interagisca con il resto del sistema, che all'interno di Blender, venga caricato il modello della stampante da animare, ovvero tutti i componenti della stampante. Nello specifico ognuno di questi componenti deve avere già impostato i modificatori, i vincoli, la parentela con l'oggetto "printer" e con altri componenti quando necessario e il punto d'origine nel punto di perno, nel caso di oggetti soggetti ad animazioni che coinvolgono una componente rotatoria. I vincoli sull'oggetto sono necessari nel caso delle camere per ottenere un inquadratura in cui la stampante sia sempre al centro. La parentela degli oggetti è necessaria quando dei componenti sono collegati o montati assieme, in questo caso anche i rispettivi modelli dovranno avere lo stesso comportamento. Quando l'oggetto padre si muove tutti gli oggetti figli devono eseguire la stessa trasformazione. Questa parentela potrebbe essere eseguita anche tramite python tramite il controllo delle proprietà dei vincoli opportuni, come "copy transformation", per ridurre il numero di operazioni che il sistema deve eseguire, non apportando grossi vantaggi, in questa versione del sistema si presume che i modelli abbiano la parentela già impostata, può essere comunque fonte di lavori futuri per le prossime versioni, considerare la possibilità di automatizzare anche questo passaggio. Oltre al modello della stampante, è importante caricare anche la lista delle azioni di default, ovvero la lista delle animazioni che i componenti possono eseguire normalmente.

7.2 Il server HTTP

La parte di interazione con il sistema, presente su Blender, è gestita tramite uno script in Python. All'interno dello script si crea un server HTTP, tramite il modulo che definisce le classi per la sua implementazione, ovvero http.server. Il server comunica con il resto del sistema tramite la gestione di richieste GET (Figura 7.1). e POST (Figura 7.2).

```
def do_GET(self):
    print("thread inside get handling:",  threading.current_thread().ident)
    self.send_response(200)
    self.send_header('Content-type','text/html')
    self.end_headers()
    action_list = []
    obj_list = []
    jsonData = {}
    for action in bpy.data.actions:
        action_list.append(action.name)
    for obj in bpy.data.collections["Printer"].objects:
        obj_list.append(obj.name)
    jsonData['obj'] =  obj_list
    jsonData['actions'] = action_list
    message = json.dumps(jsonData)
    self.wfile.write(bytes(message, "utf8"))

    return
```

Figura 7.1. Lo script python presente su Blender una volta ricevuta la richiesta GET reperisce la lista dei nomi dei modelli presenti nella collezione "Printer" e delle azioni e le invia in formato json

```
def do_POST(self):
    print("thread inside post handling:",  threading.current_thread().ident)
    content_length = int(self.headers['Content-Length'])
    post_data = self.rfile.read(content_length)
    data = post_data.decode('utf-8')
    print("extraction of data ...")

    jsonData = json.loads(data)
    Text_Relations = jsonData['text']
    Default_objects = jsonData['default']
    Image_objects = jsonData['image']
    instr_list = jsonData['instructions']
    create_animation(Text_Relations,Image_objects,Default_objects,instr_list)
    self.send_response(200)
    self.send_header('Content-type', 'text/html')
    self.end_headers()

    return
```

Figura 7.2. Lo script python di Blender estrae i dati presenti all'interno del messaggio ricevuto tramite la richiesta POST, incapsula i dati all'interno dei contenitori e richiama il metodo per la generazione del video passando come parametri i contenitori creati

7.3 testHTTPServer_RequestHandler

Per la gestione delle richieste provenienti dal sistema, all'interno dello script presente su Blender, è stato creata ed instanziata la classe "testHTTPServer_RequestHandler"(Figura 7.3),

che deriva dalla classe "BaseHTTPRequestHandler" (47). Quest'ultima classe è

```
def do_POST(self):
    print("thread inside post handling:",  threading.current_thread().ident)
    content_length = int(self.headers['Content-Length'])
    post_data = self.rfile.read(content_length)
    data = post_data.decode('utf-8')
    print("extraction of data ...")

    jsonData = json.loads(data)
    Text_Relations = jsonData['text']
    Default_objects = jsonData['default']
    Image_objects = jsonData['image']
    instr_list = jsonData['instructions']
    create_animation(Text_Relations,Image_objects,Default_objects,instr_list)
    self.send_response(200)
    self.send_header('Content-type', 'text/html')
    self.end_headers()

    return
```

Figura 7.3. La classe testHTTPServer_RequestHandler deriva le sue funzionalità dalla classe BaseHTTPRequestHandler e specifica il comportamento per la gestione delle richieste POST e GET

utilizzata infatti per la gestione delle richieste HTTP che arrivano dal sistema al server qui implementato. Questa classe non ha la capacità di rispondere da sola ad alcuna richiesta HTTP, per fare ciò è necessario implementare un'ulteriore classe che funga da sottoclasse, ovvero la classe "testHTTPServer_RequestHandler" a cui vengono forniti le variabili e i metodi necessari per la gestione delle richieste. Il server HTTP su Blender, è stato sviluppato per gestire sia richieste "GET" che richieste "POST".

7.4 Richieste GET

Il server, realizzato all'interno di Blender, nel caso di ricezione di richieste "GET", ovvero delle richieste, effettuate dal sistema, in cui il sistema chiede le liste dei nomi delle azioni (Figura 7.4) e i nomi dei modelli rappresentanti i nomi dei componenti della stampante (Figura 7.5, il server svolge 2 compiti principali:

Figura 7.4. Esempio di lista di azioni presenti all'interno di Blender, in cui le azioni si trovano all'interno della cartella "azione" automaticamente creata da Blender alla creazione del progetto

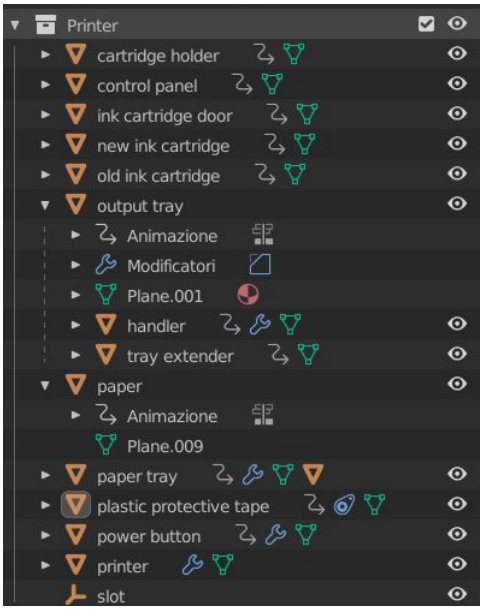

Figura 7.5. Esempio di lista dei modelli presenti all'interno della scena e all'interno della collezione "printer"

- Fornire la lista dei nomi dei modelli caricati rappresentanti i componenti della stampante, presenti in una collezione separata, rispetto ad altri oggetti, come la camera e le luci, e la lista delle animazioni di default. La lista viene creata tramite la scansione di tutti gli oggetti presenti all'interno della collezione contenente il modello pre caricato, contenente tutti i componenti

- Fornire la lista delle azioni di default. Le azioni sono presenti in una cartella apposita di Blender che può essere scansionata quando necessario.

Una volta ottenute entrambe le liste, il sistema genera un file JSON da inviare come risposta alla richiesta (Figura 7.6), con il messaggio di stato 200 nel caso in cui le operazioni eseguite siano tutte riuscite senza lanciare eccezioni, il codice 200 indica la buona riuscita della richiesta.

{"obj": ["paper tray", "printer", "output tray", "tray extender", "ink cartridge door", "cartridge holder", "old ink cartridge", "new ink cartridge", "plastic protective tape", "control panel", "power button", "paper width guide", "paper", "handler", "slot"], "actions": ["cartridge holder_default", "control panel_default", "handler_default", "ink cartridge door_close", "ink cartridge door_default", "ink cartridge door_open", "new ink cartridge_align", "new ink cartridge_default", "new ink cartridge_remove", "new ink cartridge_shake", "new ink cartridge_slide", "old ink cartridge_default", "old ink cartridge_pull out", "old ink cartridge_push down", "output tray_close", "output tray_default", "output tray_open", "paper tray_close", "paper tray_default", "paper tray_default.001", "paper tray_lift", "paper width guide_default", "paper width guide_slide left", "paper width guide_slide right", "paper_default", "paper_load", "plastic protective tape_default", "plastic protective tape_remove", "power button_default", "power button_press", "tray extender_close", "tray extender_default", "tray extender_pull out"]}

Figura 7.6. Esempio di file json inviato in risposta alla richiesta GET, il file json è strutturato come un dizionario in cui per la chiave "obj" sono elencati i nomi di tutti i modelli e per la chiave "actions" sono elencati i nomi di tutte le azioni. Le parole rappresentanti le chiavi sono evidenziate in giallo, mentre la lista dei nomi dei modelli e delle azioni sono evidenziate rispettivamente in rosso e in blu

7.4.1 Richiesta POST

Il server, realizzato all'interno di Blender, nel caso di ricezione di richieste "POST", ovvero delle richieste in cui il sistema invia al server delle liste necessarie a Blender per la generazione automatica delle animazioni, il server riceve un file JSON contenente:

- La lista delle relazioni, ovvero la lista delle istruzioni proveniente dall'analisi del testo in linguaggio naturale
- La lista degli oggetti di default, contenente la posizione, la locazione, il ROI di default di tutti i componenti rilevati all'interno delle immagini di calibrazione
- La lista degli oggetti rilevati all'interno delle istruzioni rappresentate in forma grafica.

Una volta estrapolate le liste dal file JSON, Blender procederà con la generazione o selezione dell'animazione corrispondente sfruttando i dati presenti al loro interno. I procedimenti di generazione e di selezione delle animazioni vengono descritti rispettivamente nella sezione 7.8 e nella sezione 7.9

7.5 Calibrazione delle camera principale

La fase di calibrazione della camera principale (schematizzata in Figura 7.5), è la fase iniziale dello script presente all'interno di Blender, a seguito della comunicazione col sistema tramite server HTTP. Consiste nella generazione della matrice

intrinseca rappresentante le caratteristiche della camera denominata "MainCamera". Questa matrice viene calcolata a partire dalle proprietà della camera virtuale. La matrice intrinseca è un componente fondamentale, richiesto infatti dal metodo solvePnP, per il calcolo della posa della camera in quanto fornisce informazioni riguardanti la camera di visualizzazione.

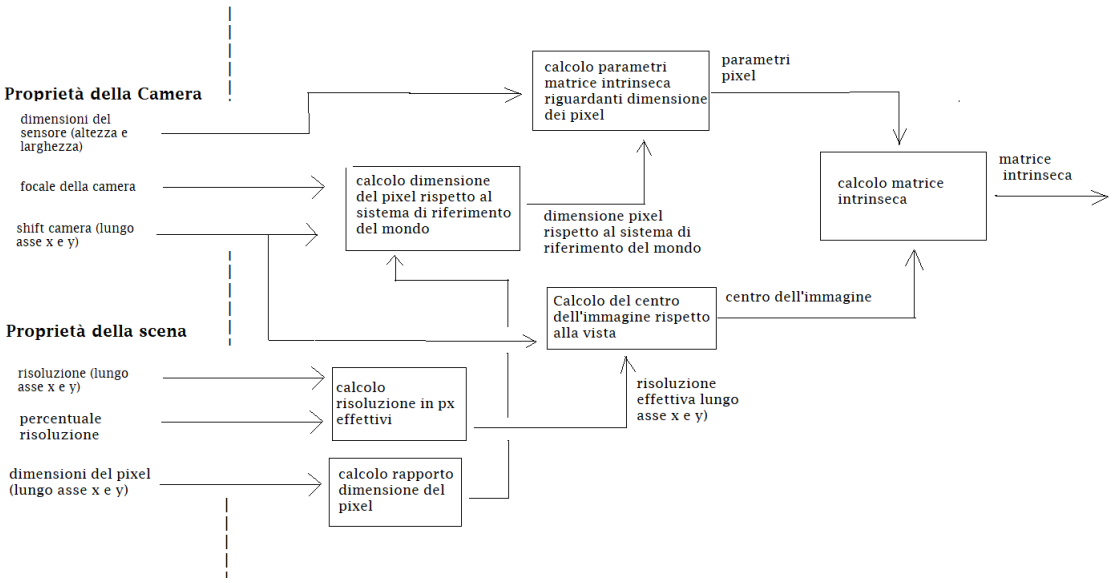

Figura 7.7. Nella figura è rappresentata l'architettura della fase di calibrazione della camera, a sinistra sono presenti le risorse utilizzate durante questa fase, mentre all'estrema destra la risorsa ottenuta. Gli elementi rappresentati da box rettangolari rappresentano le operazioni svolte dal sistema mentre gli elementi senza box rappresentano le risorse ricevute e ottenute dalle varie operazioni

7.5.1 La camera in Blender

Su Blender è presente una camera, chiamata "Main Camera" che rappresenta la camera principale all'interno di Blender. La camera all'interno di questo software è un oggetto necessario per la renderizzazione delle immagini, o dei video, definendo quale parte della scena è visibile all'intero dell'immagine. Da un oggetto camera ne deriva una "Camera View", la visuale dal punto di vista della camera attualmente attiva (Figura 7.8) all'interno del nostro sistema, in cui è presente un unica telecamera che raffigura, di conseguenza, la sola telecamera attiva. Ad un oggetto camera, all'interno di Blender, sono associati diversi parametri, alcuni dei quali necessari durante la fase di calibrazione. I parametri associati alla telecamera riguardano anche parametri di telecamere reali, questi tipi di parametri verranno

simulati dal software in modo da ottenere una rappresentazione comparabile con quella ottenibile da una telecamera reale.

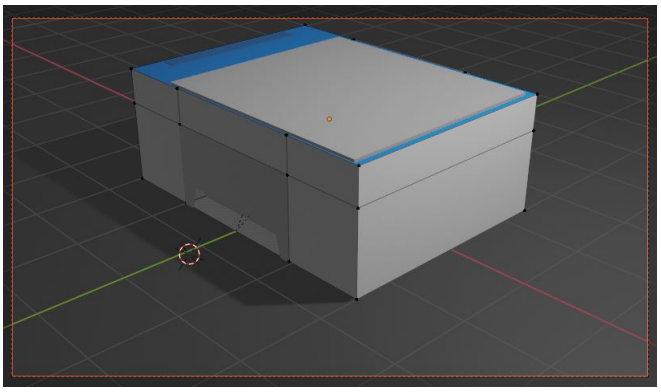

Figura 7.8. Esempio di vista della camera principale presente su Blender, La vista delimita l'area che verrà renderizzata all'interno del video che si otterrà in uscita

7.5.2 La matrice intrinseca

Durante questa fase è necessario ricavare la matrice intrinseca, rappresentante la camera utilizzata. I componenti necessari per ricavare questa matrice intrinseca sono:

- La lunghezza focale della lente, espressa in millimetri
- La larghezza del sensore, espressa in millimetri
- L'altezza del sensore, espressa in millimetri
- Adattamento del sensore, o sensor fit, ovvero l'opzione per controllare quale dimensione (Verticale o orizzontale) lungo l'angolo del campo visivo si adatta, nel nostro caso è stato selezionato "AUTO", per un'opzione scelta in modo automatico.
- Lo scostamento X, o shift x, utilizzato per regolare i punti di fuga nell'asse orizzontale
- Lo scostamento Y, o shift y, utilizzato per la regolazione dei punti di fuga per l'asse verticale

I soli parametri della camera sarebbero sufficienti nel caso di una telecamera reale, poiché invece su Blender la telecamera è simulata e si basa sull'utilizzo dei pixel come metro di misura, bisogna reperire anche i dati riguardanti la scena virtuale, ovvero:

- La Scala, ovvero la percentuale della risoluzione, necessaria a ridurre o diminuire le dimensioni dell'immagine renderizzata rispetto ai valori della risoluzione X e Y.
- La risoluzione Y, quantificata in numero di pixel lungo l'asse verticale dell'immagine
- La risoluzione X, quantificata in numero di pixel lungo l'asse orizzontale dell'immagine
- L'aspetto X/Y dei pixel, o pixel aspect ratio in inglese, in quanto i pixel potrebbero non essere sempre di forma quadrata, quindi è necessario controllare la dimensione dei pixel lungo i due assi. È necessario utilizzare l'aspect ratio corretto durante la fase di rendering per evitare il ridimensionamento con una conseguente perdita di qualità dell'immagine ottenuta

La matrice intrinseca è una matrice 3x3 (equazione 7.1) contenente i parametri della camera e della scena caratterizzante l'ambiente di Blender utilizzato, all'interno della matrice sono presenti informazioni riguardanti la lunghezza focale della camera, il formato del suo sensore, la distorsione, o skew, e il punto centrale.

$$M = \begin{bmatrix} \alpha_u & skew & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.1}$$

Dove, all'interno della matrice intrinseca M, a_u (equazione 7.2) e a_v (equazione 7.3) corrispondono alla focale della telecamera opportunamente scalata in base alla dimensione, espressa in millimetri, del pixel in coordinate del mondo, rispettivamente nell'asse orizzontale e verticale.

$$\alpha_u = \frac{\frac{focale}{sensor_width}}{AspectRatio_x} \tag{7.2}$$

$$\alpha_v = \frac{\frac{focale}{sensor_heigth}}{AspectRatio_y} \tag{7.3}$$

La dimensione del pixel in coordinate del mondo, si ottiene tramite il rapporto tra la dimensione del sensore, verticale o orizzontale, rispetto all' aspect ratio nel medesimo asse. Il valore di skew, o distorsione, rappresenta la distorsione del pixel, un valore pari a 0 implica che il pixel all'interno dell'ambiente di Blender è rettangolare in quanto non subisce distorsioni. Infine i valori di u_0 e v_0 rappresentano

il punto principale della camera. Per le immagini prese in considerazione all'interno della trattazione di questo lavoro di tesi, il punto principale è posizionato in alto a sinistra, e corrisponde al punto 0,0 delle immagini. Poiché all'interno di Blender il punto principale è invece al centro dell'immagine, bisogna convertire il centro dell'immagine in una posizione coerente per Blender, a questo proposito ai valori u_0 e v_0 viene assegnata metà dell'altezza dell'immagine, ovvero metà della risoluzione y della scena, e metà della sua larghezza, ovvero metà della risoluzione x della scena. La matrice intrinseca ha lo scopo di mappare le coordinate omogenee in tre dimensioni, espresse all'interno dello spazio della camera e non del mondo, di un qualsiasi punto all'interno dell'ambiente 3D di Blender, nelle coordinate in due dimensioni (equazione 7.4), rappresentante la sua proiezione nel piano dell'immagine.

$$P_{2D} = M P_{3D} \tag{7.4}$$

7.6 Posa della camera principale

La posa di una telecamera consiste nella derivazione ed impostazione dei valori di 6 gradi di libertà, 3 per la rotazione nei vari assi, chiamati anche rollio, beccheggio ed imbardata, ovvero roll, pitch e yaw in inglese, e 3 per la traslazione lungo i vari assi (x, y, z) rispetto all'origine del mondo 3D. La posa della camera è la fase successiva alla calibrazione della stessa, ed una delle sue applicazioni, per cui è stata scelta per il sistema trattato in questo lavoro di tesi, è la posa degli oggetti 3D a partire da immagini. Per calcolare la posa della telecamera il sistema utilizza il metodo SolvePnP, un metodo di openCV creato appositamente per la posa di oggetti e della telecamera. In questa fase vengono utilizzati i punti memorizzati all'interno della lista delle posizioni di default degli oggetti, ricevuta tramite server HTTP dal sistema.

7.6.1 Liste dei punti oggetto e immagine

All'interno del sistema descritto in questa tesi, si è scelto di utilizzare 4 punti, per identificare e rappresentare un oggetto, sia nell'immagine 2D che nel modello 3D pre caricato su Blender, le cui coordinate 3D sono recuperate dai vari oggetti costituenti il modello all'interno di Blender. Il sistema per ogni modello presente all'interno della collezione "Printer" reperisce la lista dei vertici che lo definiscono, e li converte in coordinate rispetto al sistema di riferimento del mondo. Infine calcola il valore massimo e minimo per i diversi assi. Quest'ultimi valori vengono utilizzati dal sistema per identificare i 4 punti che rappresentano gli angoli del modello. Poiché un oggetto deve essere definito in base alla corrispondenza dei

punti 3D e di quelli 2D, ne consegue che questi punti devono essere sempre presenti e identificabili in ogni immagine, indifferentemente dall'angolazione con cui l'oggetto è inquadrato. Per questo motivo si è scelto di prendere i soli 4 punti che sono sempre visibili e identificabili, nello specifico i punti presi sono i due angoli inferiori nella parte frontale della stampante e i due angoli superiori nella sua parte posteriore (Figura 7.9).

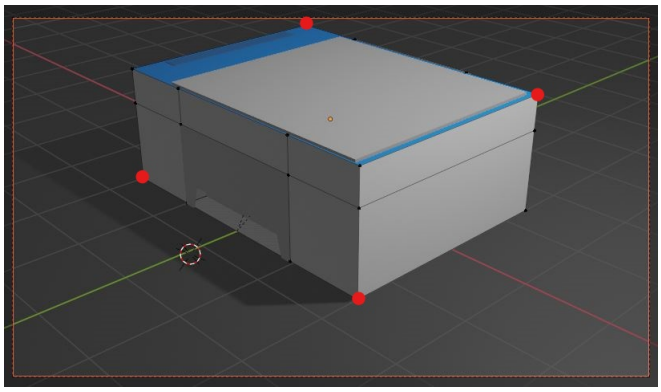

Figura 7.9. Esempio di selezione dei 4 punti trai i vertici caratterizzanti il modello della stampante pre caricato, 2 punti rappresentano gli angoli inferiori del lato frontale e 2 rappresentano gli angoli superiori del lato posteriore

Per reperire la lista delle coordinate 2D, rappresentanti le proiezioni dei punti 3D interni dell'immagine, si controlla che all'interno della lista delle coordinate 2D di default, inviata dal sistema, sia presente un elemento con il nome "printer" associato, nel caso in cui questo elemento fosse disponibile, si recupera la lista dei punti ad esso legato.

7.7 SolvePnP

SolvePnP è un metodo della libreria OpenCV realizzato allo scopo di trovare la posa di un oggetto dalla corrispondenza tra i punti 2D e quelli 3D (Figura 7.10). La funzione restituisce in uscita i vettori di rotazione e quelli di traslazione che permettono di convertire un punto 3D espresso nel sistema di coordinate dell'oggetto in un punto espresso nel sistema di coordinate della telecamera.

La Prospettiva-n-Punto, o Perspective-n-Point (PnP) in inglese, è il problema che riguarda la stima di una telecamera dato un insieme di n punti 3D nelle coordinate del mondo e un insieme di n punti 2D corrispondenti alla loro proiezione all'interno dell'immagine. SolvePnP in base al numero di punti in ingresso che utilizza ed alle proprietà di questi punti, ad esempio sei i punti sono complanari o meno, viene utilizzato un metodo differente, chiamato da SolvePnP. Nel caso

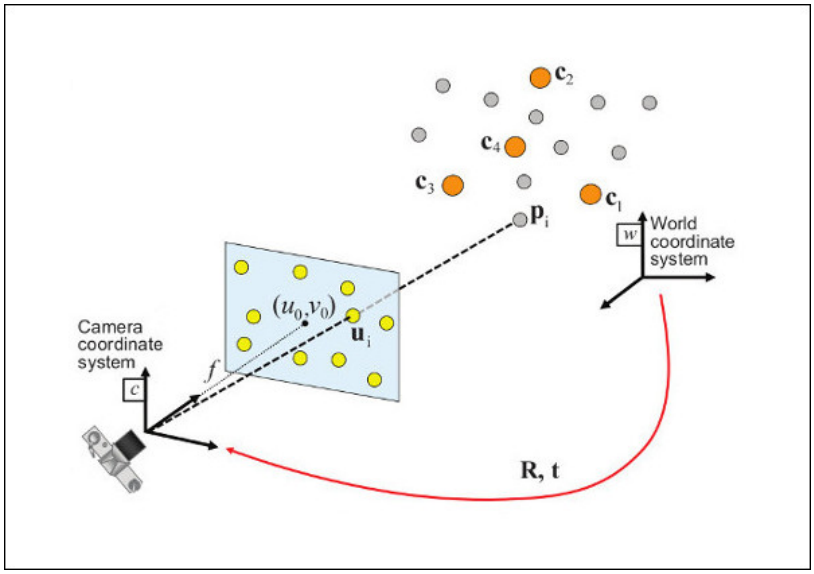

Figura 7.10. La figura reperita dalla documentazione ufficiale di OpenCV (48) mostra il funzionamento della conversione di un punto espresso nelle coordinate del mondo, ovvero dell'oggetto, in coordinate della telecamera dove i punti ci rappresentano i punti passati per la corrispondenza, mentre i punti pi rappresenta il punto da convertire, infine R e t sono rispettivamente i vettori di rotazione e traslazione utilizzati per la conversione in modo da ottenere i punti sul piano di vista della telecamera

descritto in questo lavoro di tesi, poiché i punti utilizzati per descrivere un oggetto sono 4, e questi punti sono quelli che verranno passati in ingresso a solvePnP, il metodo chiamerà i metodi P3P, ovvero SOLVEPNP_P3P o SOLVEPNP_AP3P, in quanto i 4 punti 3D non sono complanari. Il metodo riceve in ingresso:

- l'insieme dei punti oggetto 3D, ovvero una matrice o un vettore di punti
- l'insieme dei punti immagine 2D corrispondenti, ovvero una matrice o un vettore di punti in 2 dimensioni
- la matrice intrinseca della camera principale, calcolata durante la fase di calibrazione
- i coefficienti di distorsione, ovvero un vettore contenente i valori per la correzione della distorsione radiale e tangenziale, un tipo di distorsione che viene, a volte, introdotto dalle lenti

Il metodo infine restituisce in uscita il vettore di rotazione, quello di traslazione e un valore booleano per indicare il successo o meno dell'operazione di conversione.

Lo script dopo di ché converte il vettore di rotazione nella sua rappresentazione matriciale al fine di poter essere utilizzato per la stima della posizione. Per eseguire questa conversione, lo script chiama il metodo, sempre della libreria OpenCV, Rodriguez, (49) che permette la conversione da vettore a matrice e viceversa.

7.7.1 Cambio di coordinate da OpenCV a Blender

Il metodo SolvePnP non è pensato per mettere in posa una camera virtuale presente all'interno di Blender, ma per mettere in posa una camera reale definita, all'interno di un sistema, tramite la funzione di calibrazione fornita da OpenCV. Per questo motivo sorgono dei problemi che riguardano l'orientamento del sistema di coordinate utilizzato in OpenCV rispetto a quello utilizzato su Blender. SolvePnP infatti essendo un metodo di OpenCV restituisce i suoi vettori in base al sistema di coordinate della libreria da cui proviene, i suoi risultati quindi necessitano di un ulteriore elaborazione prima di essere utilizzabili su Blender per il posizionamento della camera virtuale. Per comprendere come applicare la giusta rotazione, in modo da compensare il cambio di coordinate bisogna innanzi tutto osservare l'orientamento degli assi nei due rispettivi ambienti. Per l'ottenimento di queste informazioni, lo studio qui descritto si basa sulla documentazione di Blender (50) e di OpenCV (51) (Figura 7.11). La documentazione ci informa che in Blender gli assi sono:

- da sinistra a destra per la X
- da giù a su per la Y
- verso lo schermo per la Z

per quanto riguarda invece l'orientamento degli assi nell'ambiente di OpenCV:

- da sinistra a destra per la X
- da su a giù per la Y
- lontano dallo schermo per la Z

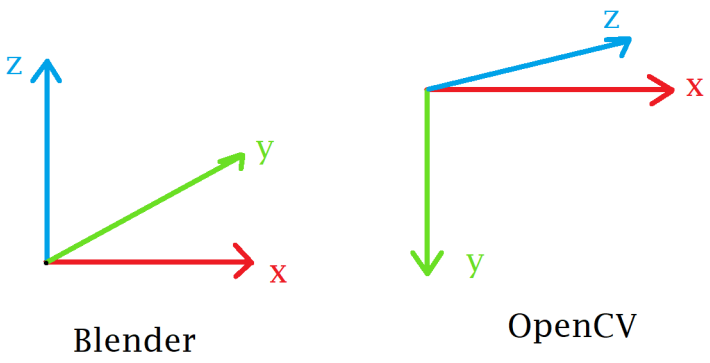

Figura 7.11. Rappresentazione grafica degli assi utilizzati nei due rispettivi ambienti, Blender e OpenCV

Entrambi i sistemi hanno due diversi orientamenti del sistema di coordinate in base al tipo di coordinate, ovvero quello globale e quello dello schermo. Per la traduzione corretta delle coordinate bisogna convertire le coordinate dal sistema di coordinate dello schermo di OpenCV, ovvero quelle rappresentate nella vista della camera virtuale che il metodo solvePnP si aspetta, nelle coordinate globali di Blender, ovvero quelle con cui è localizzata la telecamera virtuale e gli altri oggetti nell'ambiente di simulazione. Per la conversione, seguendo la regola della mano destra, bisogna effettuare una rotazione lungo l'asse X di -90 gradi. A tal fine una volta ottenuta la matrice di rotazione a partire dal vettore di rotazione ricevuto dal metodo solvePnP, viene generata la matrice rotazionale (equazione 7.5) necessaria per la conversione.

$$M_{rotX} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(-90) & sin(-90) \\ 0 & sin(-90) & cos(-90) \end{bmatrix} \tag{7.5}$$

La matrice viene quindi moltiplicata per la matrice di rotazione ricavata da metodo al fine di attuare la conversione delle coordinate rotazionali.

7.7.2 Calcolo della posizione e della rotazione

Una volta ottenuta la matrice di rotazione corretta per il nostro sistema, il sistema calcola la posizione e la rotazione che la telecamera dovrà assumere per inquadrare la scena. SolvePnP fornisce la posizione dell'origine rispetto alla camera, invece al sistema occorre conoscere la posizione della camera rispetto all'origine, a questo

proposito, per calcolare la posizione il sistema moltiplica il vettore di traslazione per il negativo dell'inverso della matrice di rotazione (equazione 7.6)

$$Cam_{pos} = -M'_{rot} * V_{trasl} \tag{7.6}$$

Per quanto riguarda i valori della rotazione della camera, ovvero i valori di roll, pitch e yaw, il sistema li calcola a partire dalla matrice di rotazione, utilizzando le funzioni geometriche di arcoseno, arcotangente e coseno.

La fase di posa della camera è una fase legata alla bontà del rilevamento della maschera della stampante e della successiva identificazione degli angoli. Entrambe le fasi sono fonte di errore rispetto alla posizione effettiva degli angoli della stampante nell'immagine, uno dovuto ad una identificazione non perfetta della maschera ed uno dovuto al fatto che il calcolo degli angoli è una approssimazione e scrematura degli angoli ottenuti tramite i metodi di corners detection. Questo errore durante la posa della camera porta a risultati a volte bizzarri, tanto più che l'errore è marcato, infatti l'errore della posa, aumenta per ogni punto 2D fuori posizione rispetto al punto effettivo, in quanto l'inquadratura viene aggiornata punto per punto durante l'esecuzione del metodo solvePnP.

7.7.3 Scelta della camera

Per ottenere un'inquadratura attinente all'immagine di partenza e godibile, ovvero senza inclinazioni della camera che porta a non comprendere la scena mostrata, si utilizzano una serie di camere pre caricate. Queste camere mostrano le differenti inquadrature che sono accettate per visualizzare l'animazione, il sistema descritto in questo lavoro di tesi, calcola la distanza tra la posizione della camera ottenuta nella fase di posa con le posizioni delle camere pre impostate, selezionando e rendendo attiva per il rendering del video, ovvero per la generazione del video, la camera la cui distanza è inferiore, quindi la camera che più si avvicina alla posa calcolata ottenendo però un inquadratura godibile in quanto l'inclinazione è stata precedentemente impostata.

7.8 Generazione automatica delle animazioni

Una volta posizionata la camera virtuale all'interno dell'ambiente virtuale il sistema descritto in questo lavoro di laurea, se possibile, genera in automatico le animazioni che riproducono il comportamento descritto dalle istruzioni, utilizzando i dati ricevuti, provenienti dalla fase di analisi delle immagini e del testo, e memorizzati nelle apposite liste. Il sistema, a seguito di controlli e calcoli preliminari, può generare in automatico le animazioni di due tipi di comportamento, la

rotazione o la traslazione. Il tipo di comportamento viene stabilito durante la conversione dei nomi delle azioni delle istruzioni all'interno della lista dei componenti identificati all'interno delle immagini (sezione 6.5.2).

7.8.1 Controlli e calcoli preliminari

Il sistema descritto in questo lavoro di tesi per ogni istruzione, presente all'interno della lista delle relazioni proveniente dall'analisi del testo, controlla se all'interno della lista contenente i dati dei componenti animabili ci sia corrispondenza, ovvero se all'interno dell'altra lista sia presente un elemento il cui numero di istruzione a cui si riferisce sia il medesimo. Inoltre il sistema controlla che il componente a cui gli elementi selezionati provenienti dalle due liste, ovvero lista delle relazioni e lista degli oggetti animabili, si riferiscono sia lo stesso. Può infatti capitare che all'interno di una sola istruzione i componenti che debbano essere animati siano molteplici, il sistema quindi si assicura che i due elementi si riferiscano alla stessa istruzione ed allo stesso componente. Questi controlli sono necessari al fine di selezionare correttamente:

- Le istruzioni per cui è possibile generare in automatico l'animazione corrispondente, tramite la rotazione o traslazione del componente.
- Le istruzioni per cui è possibile selezionare l'animazione pre caricata all'interno della libreria. Le istruzioni vengono selezionate nel caso non fosse possibile generare in automatico l'animazione.

Una volta che il sistema qui descritto reperisce i dati dell'oggetto da animare provenienti dall'analisi dell'immagini, se presenti, estrapola i dati all'interno dell'elemento e li incapsula in opportune strutture, in modo da essere facilmente usufruibili, ed estrapola le posizioni di default, definite rispetto all'ambiente virtuale, del componente a cui l'istruzione si riferisce, presenti all'interno della rispettiva lista. Successivamente calcola i dati necessari alla generazione automatica delle animazioni, sfruttando i dati estrapolati e i dati provenienti dalle precedenti fasi. Nello specifico calcola:

- La matrice R|t, formata dalla matrice rotazionale 3x3 ed il vettore traslazionale 3x1 provenienti dalla fase di posa della camera, e la sua inversa.
- L'inverso della matrice intrinseca della camera virtuale
- Le coordinate correnti degli angoli del componente a cui l'istruzione si riferisce, nell'ambiente 3D
- La lunghezza del lato lungo l'asse verticale del componente a cui l'istruzione si riferisce, a partire dalle coordinate del componente in posizione di default nell'ambiente virtuale.

- Una approssimazione del valore di profondità del componente, ovvero la differenza di valore delle coordinate degli angoli del componente lungo l'asse ortogonale che esce dallo schermo per l'immagine 2D. Per approssimare questo valore, oltre alle coordinate degli angoli del componente da animare, nell'ambiente virtuale, si utilizza anche la lunghezza del lato precedentemente calcolata.

- La proiezione prospettica dei punti 2D rappresentanti gli angoli del componente a cui l'istruzione si riferisce, il sistema quindi dai punti 2D ottiene i punti 3D corrispondenti all'interno dell'ambiente virtuale.

Per il calcolo dei punti 3D corrispondenti ai punti 2D individuati dall'immagine, il sistema descritto in questo lavoro di tesi, inizialmente converte i punti espressi in coordinate 2D rispetto agli assi x e y, in punti 3D aggiungendo l'informazione della profondità calcolata precedentemente. I punti 3D ottenuti dalla conversione vengono tradotti ulteriormente rispetto al sistema di riferimento della vista della camera virtuale utilizzata su Blender. Per ottenere questa traduzione si fa uso dell'inverso della matrice intrinseca, matrice che era stata calcolata durante la fase di calibrazione della camera virtuale. Per ottenere infine le coordinate rispetto al sistema di riferimento del mondo virtuale, le coordinate vengono convertite ulteriormente tramite la moltiplicazione per la matrice R|t precedentemente calcolata.

7.8.2 Animazione di rotazione o traslazione

Il sistema descritto in questo lavoro di tesi, dopo essersi assicurato di poter animare il componente citato dall'istruzione e dopo aver calcolato i valori necessari per la generazione automatica dell'animazione, esegue operazioni differenti in base al tipo di animazione da generare. Le animazioni generabili sono la rotazione e la traslazione del componente. Per la rotazione il sistema si assicura che questa avvenga lungo un solo asse e che il modello da animare abbia predefinito un punto di pivot da usare come riferimento per eseguire la rotazione. Per la traslazione il sistema assume che questa avvenga lungo un solo asse e che coinvolga tutti i punti appartenenti al componente da animare. Nel caso in cui l'immagine raffiguri solo la parte finale o iniziale dell'azione eseguita all'interno dell'istruzione. Un esempio può essere un'immagine che raffigura il momento subito dopo l'inizio dell'azione con l'aggiunta della freccia per indicare la direzione di traslazione. L'immagine non fornisce informazioni riguardanti la posizione finale che il componente deve assumere, il sistema quindi assume come posizione finale quella mostrata all'interno dell'immagine (Figura 7.12).

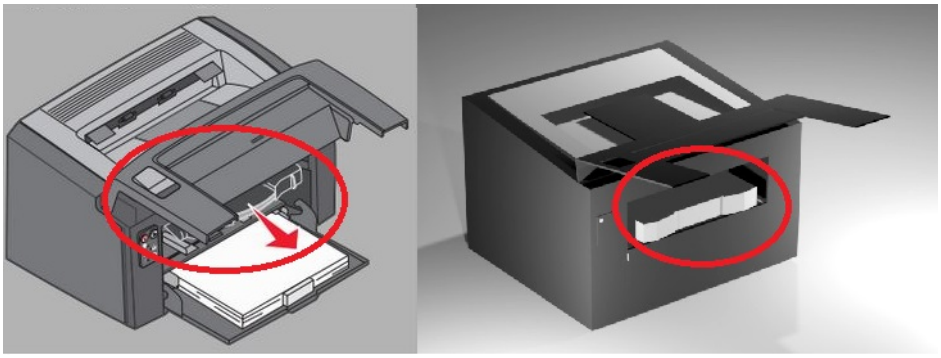

Figura 7.12. L'esempio mostra a sinistra l'immagine di partenza in cui il toner da sostituire è leggermente fuori e rappresenta la fase subito dopo l'inizio dell'azione descritta nell'istruzione. A destra viene invece mostrato il risultato della generazione automatica dell'animazione in cui il componente invece che uscire del tutto rimane nella posizione mostrata dall'immagine

Per la rotazione, il sistema qui descritto, utilizza le proiezioni prospettiche dei punti 2D, rappresentanti gli angoli dei componenti all'interno dell'immagini provenienti dai manuali di istruzioni e reperisce, all'interno della lista delle posizione di default dei componenti rispetto all'ambiente virtuale, le posizioni di default del componente citato dall'istruzione. Vengono calcolati i centroidi dei due insieme di punti 3D nell'ambiente virtuale e si ottiene una matrice 3x3 tramite la moltiplicazione matriciale dei due centroidi. La matrice così ottenuta viene fattorizzata e i valori ottenuti vengono utilizzati per la generazione della matrice rotazionale, dalla matrice rotazionale infine si ottengono i valori di roll, pitch e yaw da applicare al componente.
La procedura di proiezione prospettica da punti 2D e 3D utilizza valori approssimati di profondità, matrice intrinseca della telecamera virtuale e di vettori di rotazione e traslazione, che derivano dalla matrice intrinseca. Il valore che viene calcolato tramite questa procedura sarà quindi anch'esso una approssimazione delle coordinate reali che i punti 2D dovrebbero assumere all'interno dell'ambiente virtuale. Poiché le coordinate sono approssimate i valori di roll, pitch e yaw sono anch'essi approssimati, a questo proposito per eliminare le componenti rotazionali che non dovrebbero essere presenti, in quanto il componente è vincolato alla stampante da perni ed altri componenti, si sfruttano i vincoli impostati sulle rotazioni applicabili al componente, nelle specifico si controllano i lock che bloccano le rotazioni lungo determinati assi per selezionare lungo quale asse eseguire la rotazione.

Per quanto riguarda la traslazione, si presume che lo spostamento coinvolga tutti i vertici del componente a cui l'istruzione si riferisce. Quindi piuttosto che calcolare la proiezione e la posizione di tutti gli angoli, si calcola il vettore di traslazione

da applicare alle coordinate del punto 2D rappresentante il centro del componente raffigurato all'interno dell'immagine dell'istruzione da animare, per poi raggiungere le coordinate del punto 2D rappresentante la proiezione delle coordinate 3D attuali rappresentanti il centro del modello relativo al componente da animare. Le coordinate 3D del centro del modello rappresentano l'origine del modello in Blender ovvero punto, non appartenente alla lista dei vertici del modello, posizionato di default al centro del modello da Blender durante la creazione del modello stesso. Queste coordinate vengono proiettate eseguendo l'inverso della procedura eseguita, nel caso della rotazione, per eseguire la proiezione prospettica da punti 2D a quelli 3D. Quindi dai punti 3D nelle coordinate del mondo si ottiene tramite la moltiplicazione con la matrice R|t i punti 3D rispetto al sistema di riferimento della vista della camera. Questi punti vengono successivamente convertiti nelle coordinate 3D dell'immagine, contenente ancora l'informazione riguardante la profondità ed infine, tramite la moltiplicazione matriciale con la camera intrinseca, dai punti 3D (x,y,w) si ottengono i punti 2D (x,y) nell'immagine, ossia eliminando la componente di profondità. Per calcolare il movimento che il componente dovrà compiere viene calcolata la distanza euclidea tra i due punti 2D, il punto nell'istante precedente e quello nell'istante corrente. Il valore viene ridimensionato in base al valore di profondità, al rapporto di grandezza tra il lato del componente all'interno dell'immagine e il lato del modello 3D del componente.

7.8.3 Memorizzazione dell'animazione generata

Una volta identificate le coordinate che gli angoli del componente dovranno raggiungere a seguito dell'azione illustrata nell'immagine proveniente dal manuale di istruzione, il sistema descritto in questo lavoro di tesi, tramite Blender, genera delle actions, ovvero il contenitore utilizzato da Blender per contenere e memorizzare i dati riguardanti l'animazione. L'animazione viene generata creando una action che memorizza la posizione/rotazione che il componente assume al frame 1, ovvero prima che l'azione illustrata nell'immagine avvenga, generando un keyframe in quel frame. Successivamente applica la trasformazione sul modello, modificando la posizione/rotazione e la memorizza al frame 20 generando un keyframe in quel frame. La posizioni nell'intervallo di tempo compreso tra il frame 1 e 20 verranno calcolate tramite interpolazione da Blender (Figura 7.13).

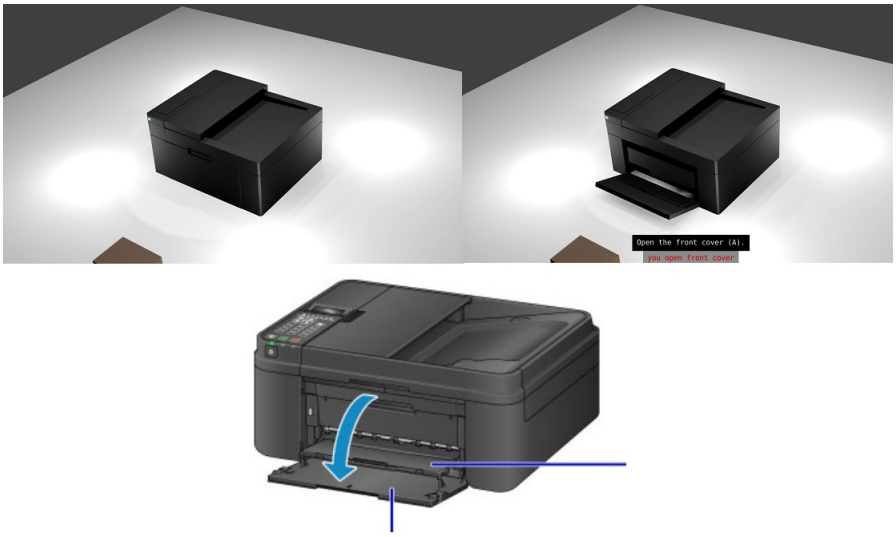

Figura 7.13. Nella figura sono rappresentati in alto i 2 fotogrammi chiave catturati all'interno dell'azione mentre in basso è riportata l'immagine di partenza

Per distinguere le azioni generate in automatico da quelle presenti di default, il nome dell'azione generata in automatico avrà il seguente formato:

nome del componente + "_" + azione svolta + "_AT"

dove l'azione svolta sarà il nome del movimento eseguito, ovvero rotazione o traslazione, mentre i caratteri finali AT indicano che la azione è stata generata in modo automatico.

7.9 Selezione delle animazioni dalla libreria

Non è sempre possibile generare delle animazioni in automatico utilizzando le immagini provenienti dal manuale di istruzione, è quindi necessario avere una libreria contenente delle animazioni di default per ogni componente. Per ogni modello vengono create manualmente in media 25 animazioni di default, il numero può variare in base al numero dei componenti che compongono il modello. Tra le animazioni più comuni abbiamo:

- "front cover_open"
- "front cover_close"
- "output tray_open"

- "output tray_close"
- "new ink cartridge_insert"
- "new ink cartridge_remove"
- "old ink cartridge_insert"
- "old ink cartridge_remove"

Le animazioni di default sono animazioni generate in fase di creazione del modello prima dell'utilizzo del sistema descritto in questo lavoro di tesi, è necessario che le animazioni di default seguano il seguente formato per la scelta del proprio nome:

nome del componente + "_" + azione eseguita

Per la selezione dell'animazione all'interno della libreria di animazione dei modelli 3D, il sistema descritto in questo lavoro di tesi, utilizza i dati estrapolati dall'analisi del testo e memorizzati all'interno della rispettiva lista su Blender. Come prima operazione il sistema controlla se sia generabile in automatico l'animazione descritta nel testo, seguendo la procedura descritta nel paragrafo precedente. Nel caso in cui l'azione memorizzata all'interno della lista non fosse animabile automaticamente, il sistema procede con la selezione dell'animazione all'interno della libreria di animazione dei modelli 3D pre definita e pre caricata su Blender. Per identificare quale delle animazioni presenti all'interno della libreria occorra mostrare, il sistema controlla inizialmente se il componente specificato dall'istruzione è presente tra i modelli dei componenti all'interno di Blender. Se così fosse il sistema compone il nome dell'azione da cercare utilizzando il nome del componente e il nome dell'azione memorizzata nella relazione, seguendo il formato prima citato. Il nome dell'azione composto viene confrontato con i nomi delle azioni presenti all'interno della libreria della animazioni pre caricate, nel caso in cui ci fosse corrispondenza il sistema seleziona l'animazione corrispondente. La corrispondenza in questo caso deve essere perfetta in quanto i nomi dei componenti e delle azioni che riceve Blender sono stati già convertiti in base ai nomi dei modelli e delle azioni presenti al suo interno, sarà possibile quindi comporre esattamente lo stesso nome delle azioni presenti all'interno della libreria di animazioni.

Il sistema è stato valutato con modelli generati a partire da

7.9.1 Generazione del video

Finita la fase di selezione e generazione dell'animazione il sistema descritto in questo lavoro di tesi procede con il montaggio del video animato tramite l'utilizzo dell'editor di animazioni non lineari (NonLinear Animation Editor - NLA Editor)

presente su Blender. Questo editor permette di manipolare e riutilizzare le azioni senza gestire manualmente i fotogrammi chiave e per concatenare facilmente una sequenza di azioni stratificate. Il sistema inizialmente, per ogni modello, genera all'interno dell'editor NLA le tracce dove inserire le animazioni, nel caso in cui queste fossero già presenti le seleziona. Per ogni modello da animare, memorizza all'interno della traccia una azione al secondo 0 della durata di 1 secondo con un unico keyframe contenente la posizione del modello iniziale, questo procedimento assicura che l'animazione partirà con tutti i componenti nelle loro posizioni iniziali. Successivamente il sistema inserisce un'azione ogni 25 fotogrammi per ogni relazione presente nella lista delle relazioni elaborate dall'analisi del testo. Se è stato possibile generare automaticamente l'animazione il sistema inserisce quella altrimenti seleziona l'animazione corrispondente presente all'interno della libreria delle animazioni dei modelli 3D. Viene scelto come temporizzazione 25 fotogrammi in quanto ogni azione dura 20 fotogrammi, all'interno del sistema tra una azione ed un altra il sistema descritto in questo lavoro di tesi, aggiunge ulteriori 5 fotogrammi al fine di rendere le azioni distinte e chiare da visualizzare. Il sistema inserisce le azioni all'interno della traccia appartenente al modello del componente da animare. Il sistema aggiunge alla scena così realizzata i sottotitoli riportanti l'istruzione di partenza presente all'interno del manuale di istruzioni con testo bianco e sfondo nero, mentre riporta in rosso con sfondo grigio l'istruzione estrapolata animabile su Blender (Figura 7.14). Questi sottotitoli vengono aggiunti all'interno dell'editor video di Blender (Figura 7.15) contemporaneamente a quando le azioni vengono inserite all'interno dell'NLA editor . Il video viene infine renderizzato in formato AVI, un formato contenitore standard.

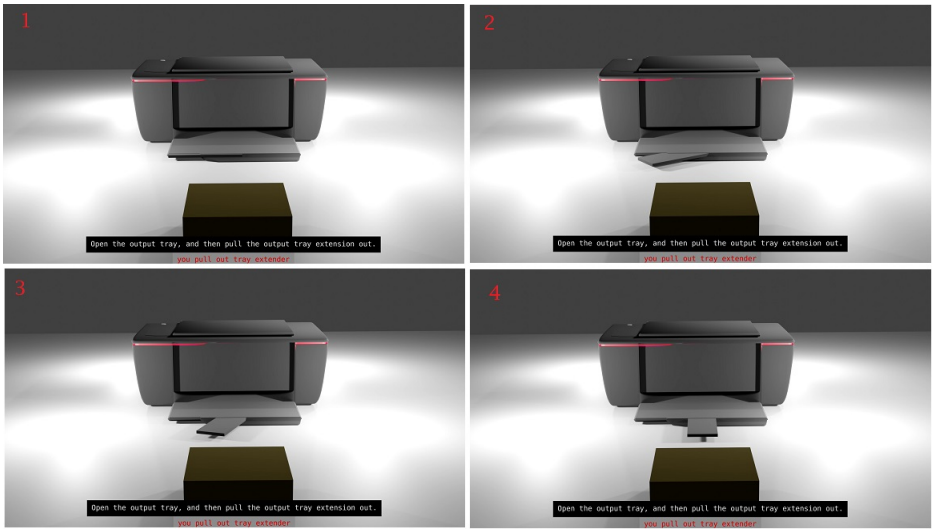

Figura 7.14. Nella figura sono rappresentati 4 fotogrammi di una animazione, dove il 1 ed il 4 sono fotogrammi chiave mentre il 2 ed il 3 sono stati interpolati

Figura 7.15. Nella figura è rappresentato lo stato finale dell'editor

Parte III

Terza Parte

Capitolo 8

Sperimentazione

In questo capitolo si espongono i risultati ottenuti valutando il video animato dei modelli 3D realizzato a partire da un manuale di istruzioni di una stampante ed esaminando i risultati derivanti da test riguardanti:

- la capacità del sistema di estrapolare le istruzioni da eseguire dal testo
- la capacità del sistema di convertire le istruzioni precedentemente estratte in istruzioni utilizzabili all'interno di Blender
- la capacità del sistema di generare animazioni in modo automatico

8.1 Valutazione del video

Per la fase di valutazione del video generato automaticamente dal sistema si è scelto di utilizzare come manuale di istruzioni quello realizzato dagli specialisti per la stampante Epson WF-7010, un manuale liberamente consultabile online. Le immagini presenti all'interno del manuale di istruzione di questa stampante non sono state utilizzate per l'addestramento del modello dei pesi per il riconoscimento delle immagini al fine di poter valutare coerentemente il funzionamento del sistema di analisi delle immagini. Il sistema utilizza 2 set di immagini uno per la calibrazione della camera ed uno per l'identificazione dei componenti e la successiva deduzione dell'azione eseguita e genera in automatico le loro rappresentazioni in bianco e nero. La lista di istruzioni riportate all'interno del manuale sono le seguenti:

1. Make sure the light is on, but not flashing.
2. Open the printer cover.
3. For best results, shake the new ink cartridge four or five times before opening the package.

4. Remove the new ink cartridge from the package.

5. Remove the yellow tape from the bottom of the ink cartridge.

6. Open the cartridge cover.

7. Squeeze the tab at the back of the ink cartridge that you want to replace. Lift the cartridge straight up and out of the printer, you dispose of it properly.

8. Place the new ink cartridge into the cartridge holder with the bottom down. Then push down the new ink cartridge until it clicks into place.

9. When you are finished replacing cartridges, close the cartridge cover and the printer cover.

8.1.1 Preparazione ambiente di Blender

Il sistema ottiene un'applicazione grafica che consiste in un video generato tramite l'animazione di modelli 3D rappresentanti i componenti caratterizzanti la stampante Epson. I modelli 3D sono stati realizzati manualmente a partire unicamente dalle immagini della stampante presenti online cercando di conservare il rapporto di grandezza dei componenti. Il modello non ambisce ad essere una copia perfetta della stampante presente sui manuali in quanto in questo lavoro di tesi si preferisce concentrare il lavoro sull'adempimento dei compiti di analisi del testo e delle immagini. I nomi dei componenti assegnati ai modelli 3D presenti su Blender e i nomi delle azioni presenti all'interno della libreria di animazioni, sono stati selezionati ed attribuiti in base ai nomi presenti all'interno del manuale di istruzioni preso in esame. Oltre ai modelli sono stati create e posizionate in anticipo l'insieme di camere virtuali che rappresentano le possibili inquadrature utilizzabili per la realizzazione del video tra cui il sistema quindi dovrà scegliere. All'interno di Blender i modelli della stampante risiedono tutti all'interno della collezione "Printer", mentre l'insieme di camere virtuali risiedono tutte all'interno della collezione "Cameras"(Figura 8.1).

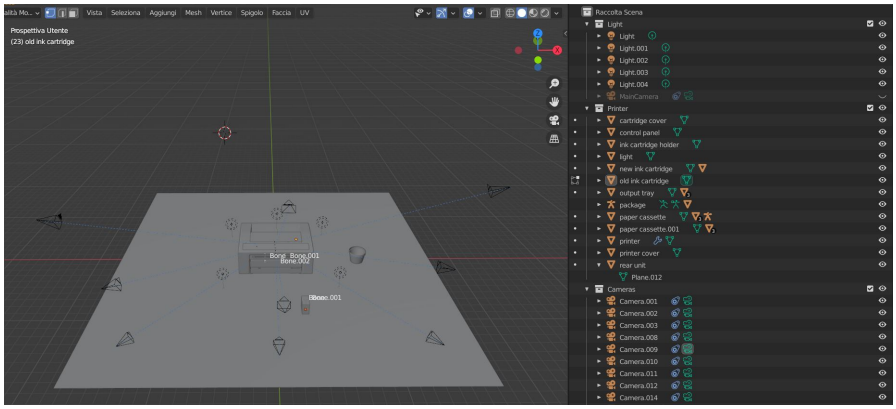

Figura 8.1. Ambiente virtuale di partenza di Blender in cui i modelli si trovano nella loro posizione iniziale, le inquadrature delle camere sono state impostate e sono state posizionate le luci per una migliore visualizzazione, a sinistra si possono vedere le collezioni utilizzate e gli oggetti che contengono

Al momento dell'avvio dello script python presente su Blender, il sistema imposterà lo stato di partenza dell'Editor non lineare presente all'interno di Blender (Figura 8.2), memorizzando le posizioni di partenza di ogni componente e generando delle azioni contenenti un keyframe di posizione e rotazione di 1 frame di durata, posizionandoli al frame 0 della traccia di ogni componente. Solo gli oggetti presenti all'interno della collezione "Printer" subiranno questo procedimento.

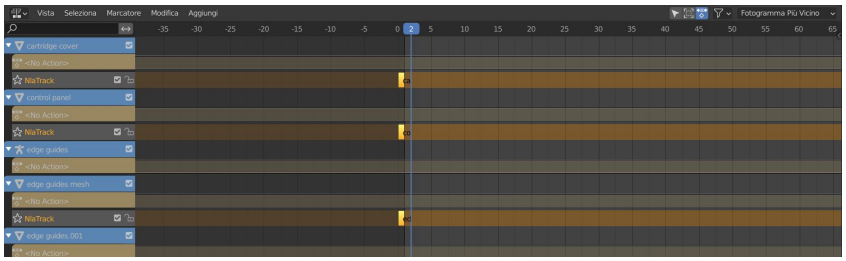

Figura 8.2. Editor per le animazioni non lineari in cui sono stati inseriti come operazioni preliminari delle azioni con le posizioni di default dei modelli, i tasselli evidenziati in giallo prima della linea blu raffigurante il frame corrente

8.1.2 Realizzazione del video

Una volta ottenuto lo stato di partenza dell'editor non lineare il sistema procede con la comunicazione dei nomi dei componenti e delle azioni ed attende la ricezione della lista di istruzioni da eseguire, la lista delle azioni generabili a partire dalle

immagini e le posizioni di default dedotte dal sistema tramite l'analisi dell'immagine. Il sistema quindi a partire dalle immagini presenti all'interno della cartella "images calibration", ovvero la cartella contenente tutte le immagini necessarie alla fase di calibrazione della camera, otterrà la posa di una camera virtuale (Figura 8.3) . La posizione così ottenuta viene utilizzata per selezionare la camera virtuale, presente all'interno della collezione "Cameras", la cui inquadratura si avvicina maggiormente a quella mostrata nell'immagine.

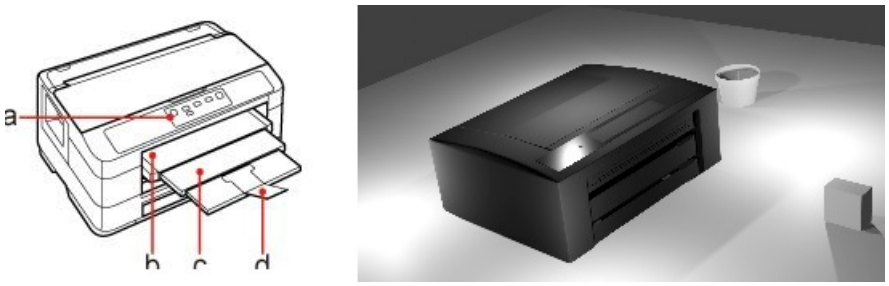

Figura 8.3. Posa della camera per la stampante in esame, a sinistra l'immagine utilizzata per la calibrazione e a destra l'inquadratura della camera virtuale ottenuta

L'inquadratura risulta come mostrato nella figura antecedente, simile all'immagine di partenza, anche se non uguale, questo avviene in quanto le posizioni delle camere sono pensate per essere standard per ogni stampante. Come lavoro futuro citato anche successivamente, infatti il sistema necessiterà di un'interfaccia grafica che permetta di inserire il modello della stampante all'interno dell'ambiente di Blender pre impostato, in questo modo l'utilizzatore avrà bisogno solamente di avere il modello già realizzato e la lista delle animazioni senza aver la necessita di dover impostare anche il resto degli oggetti.
Questo test mira ad osservare l'efficacia di questo algoritmo, inquadrature migliori possono essere ottenute tramite il posizionamento più consono e accurato delle camere standard. Il sistema a questo punto cerca la corrispondenza tra le istruzioni presenti all'interno della lista di istruzioni precedentemente ricevuta tramite server. Per ogni istruzione presente all'interno della lista, il sistema, controlla se sono presenti i dati necessari alla realizzazione automatica della stessa. Nel caso questi fossero presenti il sistema genera l'animazione corrispondente ed inserisce questa all'interno dell'editor al frame opportuno (Figura 8.4).

Figura 8.4. Sopra sono presenti 2 catture di un'animazione generata in automatico, catture corrispondenti ai framekey inseriti dal sistema in automatico per l'animazione "place new ink cartridge", sotto viene mostrato l'editor a seguito dell'inserimento dell'azione generata in automatico riconoscibile dalle lettere "AT" finali

Nel caso in cui invece non sia possibile generare in modo automatico l'animazione, il sistema cerca una corrispondenza tra i dati presenti all'interno della lista di istruzioni rappresentanti il componente da animare e l'animazione da mostrare, e le animazioni presenti all'interno della libreria dell'animazioni. Quando la libreria trova una corrispondenza inserisce l'animazione all'interno dell'editor delle animazioni non lineari nella posizione opportuna (Figura 8.5)

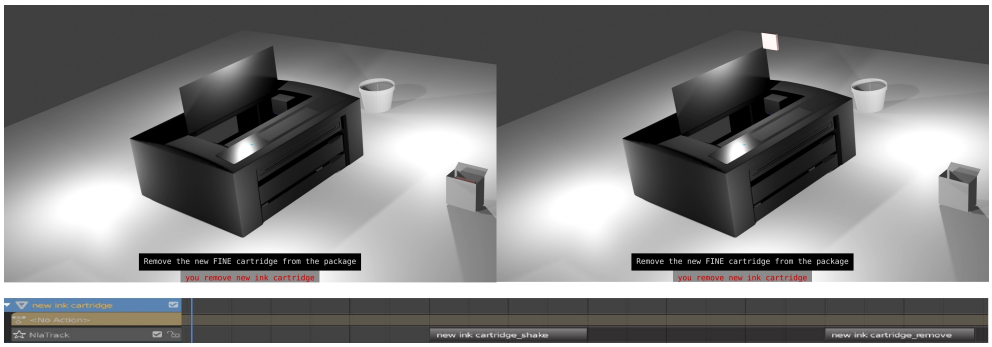

Figura 8.5. Sopra sono presenti 2 catture di un'animazione selezionata, le catture corrispondono ai framekey iniziale e finale per l'animazione "remove new ink cartridge", sotto viene mostrato l'editor a seguito dell'inserimento nel frame opportuno dell'azione selezionata dove si può vedere la presenza di un animazione precedentemente inserita

La sequenza di azioni così disposte all'interno delle tracce di ogni modello di

componente, generano l'intera scena da animare. Questa scena mostra l'animazione corretta per tutte le istruzioni come riportato all'interno della tabella contenete i dati del test.

Istruzione Manuale	istruzione presente	Istruzione estrapolata [azione] [componente]	Animazione generata Automaticamente	Animazione Selezionata dal DB	instruction image	Istruzione Correttamente Mostrata
Make sure the light is on, but not flashing	Make sure light is on	you [turn on] [light]		x		x
Open the printer cover	open printer cover	you [open] [printer cover]		x		x
For best results, shake the new ink cartridge four or five times before opening The package	shake new ink cartridge	you [shake] [new ink cartridge]		x	presente	x
	open the package	you [open] [package]		x		x
Remove the new ink cartridge from the Package	remove new ink cartridge from package	you [remove] [new ink cartridge]		x		x
Remove the yellow tape from the bottom Of the ink cartridge	Remove yellow tape	you [remove] [yellow tape]		x	presente	x
Open the cartridge cover	open cartridge cover	you [open] [cartridge cover]		x	presente	x
Squeeze the tab at the back of the ink cartridge that you want to replace. Lift the cartridge straight up and out of the Printer. You dispose of it properly	lift cartridge	you [lift] [ink cartridge]		x		x
	dispose cartridge properly	you [dispose] [ink cartridge]		x		x
Place the new ink cartridge into the cartridge holder with the bottom down. Then push down The new ink cartridge until it clicks into place	place new ink cartridge	you [place] [new ink cartridge]	x		presente	x
	push down new ink cartridge	you [push] [new ink cartridge]		x		x
When you are finished replacing cartridges, close the cartridge cover and the printer cove	close cartridge cover	you [close] [cartridge cover]		x	presente	x
	close printer cover	you [close] [printer cover]		x		x
9	13	13	1	13	5	13

Figura 8.6. Da sinistra verso destra si vedono gli output dell'elaborazione del testo , in particolare vengono mostrate le frasi iniziali, le istruzioni estrapolate con un'analisi manuale e le istruzioni inviate a Blender alla fine della fase di analisi e conversione delle frasi. Proseguendo verso destra vengono segnate tramite una "x" le istruzioni per cui è stata generata in automatico l'animazione e per cui l'animazione è stata selezionata. La colonna "instruction image" tiene traccia delle istruzioni per cui è presente una rappresentazione grafica ed infine l'ultima colonna segnala con una "x" tutte le istruzioni per cui è stata generata l'animazione corretta

La scena da animare viene infine utilizzata dall'editor video (Figura 8.7) per l'aggiunta dei sottotitoli riportanti l'istruzione presente all'interno del testo in bianco con sfondo nero e l'azione eseguita su Blender in rosso con sfondo grigio.

Figura 8.7. Esempio di prodotto finale ottenuto a seguito della creazione della scena e dell'inserimento dei sottotitoli opportuni all'interno dell'editor video presente su Blendere

Come ultima azione il sistema genera il video in formato AVI.

8.2 Test analisi testo e generazione animazioni

Al fine di poter valutare la capacità di analisi del testo in linguaggio naturale e al fine di poter valutare l'impatto della generazione delle animazioni in automatico, sono stati eseguiti dei test che consistono nell'estrapolazione del testo proveniente dallo stesso manuale di istruzioni di quello utilizzato per la creazione del modello 3D, in questo caso i nomi presenti all'interno del manuale sono gli stessi che sono stati utilizzati come nome per i componenti del modello su Blender. In questo test lo scopo è quello di mostrare la capacità del sistema di estrapolare le istruzioni necessarie dal testo e la capacità di generare in alcuni casi un animazione automatica adeguata. Al tal fine sono state create delle tabelle per memorizzare i dati raccolti, in particolare sono stati contati il numero di istruzioni animabili, provenienti da un'analisi manuale del testo, il numero di istruzioni animabili ritornate dalla fase di estrapolazione delle istruzioni dal testo, il numero di animazioni generate in automatico e di quelle selezionate ed il numero delle animazioni animate correttamente. Per quanto riguarda la capacità di estrapolare la lista di istruzioni presenti all'interno del testo ricevuto in ingresso dal sistema, i test mostrano che il sistema ha un'ottima capacità di estrapolazione, riuscendo ad estrarre le istruzioni anche quando la frase di partenza è complessa, ovvero formata da più istruzioni semplici. Un possibile esempio è la frase "Squeeze the tab at the back of the ink cartridge that you want to replace. Lift the cartridge straight up and out of the Printer, You dispose of it properly" che viene convertita in 3 istruzioni semplici :

1. you squeeze tab at back
2. you lift cartridge
3. you dispose cartridge

e una relazione tra oggetti, ovvero "back of ink cartridge". Il test effettuato sulle 4 stampanti ha dimostrato che il 98,33% delle istruzioni veniva correttamente estrapolata dalle frasi di partenza (Figura 8.8).

Il sistema ha delle difficoltà nell'estrapolare la frase corretta nel caso in cui una frase contenga più clausole legate tra loro, nel caso in cui l'oggetto su cui l'azione del verbo deve essere eseguita è sottinteso o compare in altre istruzioni.

Per quanto riguarda la generazione delle animazioni in modo automatico come risultato deducibile dai test è attesa una bassa capacità del sistema di generare in modo automatico le animazioni dovuto all'utilizzo di un database di immagini annotate per la fase di training del modello scarno. Dal test è risultato che su 4 stampanti diverse il sistema è riuscito a generare in media il 10,95% delle animazioni mentre ha selezionato il restante 89,05%. Ovvero circa 1 animazione su 10 è stata generata in modo automatico (Figura 8.9). Le istruzioni per cui è

Figura 8.8. Il grafo riporta in arancione la percentuale di istruzioni per cui è stata estrapolata correttamente l'istruzione 98,33% mentre in blu la percentuale di istruzioni per cui non è stato possibile farlo ovvero 1,67%

possibile generare automaticamente un'animazione sono quelle che hanno un immagine che le rappresenti graficamente. Non tutte per tutte le istruzioni quindi è possibile generare in automatico le animazioni. Tra le istruzioni che dispongono di un'immagine che le rappresenti, la possibilità da parte del sistema di generare in automatico l'animazione dipende dalla capacità di individuare tramite il riconoscimento degli oggetti il componente corretto all'interno dell'immagine. La generazione automatica è quindi strettamente legata con l'efficacia del modello dei pesi per il riconoscimento delle immagini. L'algoritmo generato nell'attuale stato

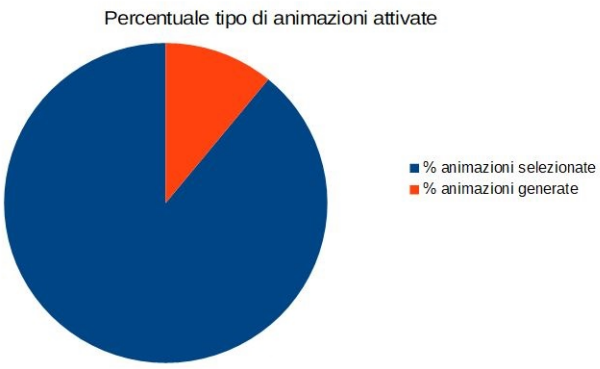

Figura 8.9. Il grafo riporta in arancione la percentuale di istruzioni per cui è stata generata un'animazione ovvero il 10,95% mentre in blu la percentuale di istruzioni per cui non è stato possibile generarla ovvero il 89,05%

del sistema e con l'attuale modello dei pesi è quindi capace di ridurre il numero di animazioni da pre caricare all'interno della libreria di animazioni di p 1/10,

risultato promettente che applicato su grandi database di animazioni può portare ad un notevole risparmio si spazio e di lavoro da parte dei professionisti. È necessario rapportare questi risultati insieme al numero di immagini rappresentanti le istruzioni fornite al sistema, immagini da cui deriva la generazione automatica dell'animazione. Infatti ne consegue che per quelle istruzioni che non posseggono un'immagine rappresentativa la probabilità di generare in automatico un'animazione è nulla. Risulta infatti che in media solo il 61,14% delle istruzioni ha una sua rappresentazione grafica, quindi per il restante 38,86% non vi è possibilità di generare in automatico un'animazione (Figura 8.10).

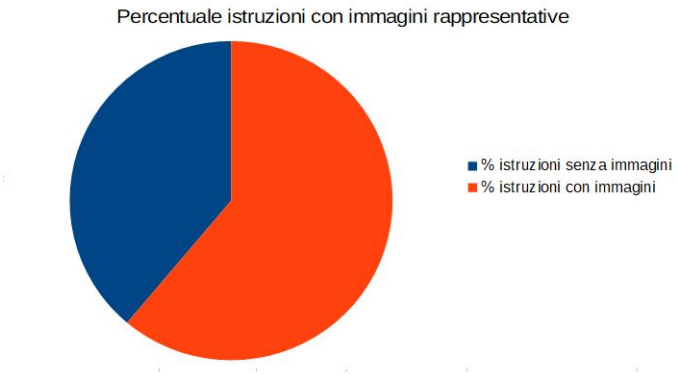

Figura 8.10. Il grafo riporta in arancione la percentuale di istruzioni per cui è disponibile un'immagine che la rappresenti 61,14% mentre in blu la percentuale di istruzioni per cui non è presente alcuna immagine ovvero il 38,86%

È importante inoltre notare che per la generazione di queste animazioni non è richiesta alcuna competenza tecnica da parte dell'utente, infatti quest'ultimo ha il solo compito di inserire il testo contenente il set di istruzioni da animare e le immagini ad esse associate.

8.3 Test generalizzazione dei testi

Per testare la capacità del sistema di convertire i nomi dei componenti nel caso in cui il modelli e il manuale fossero realizzati da due specialisti distinti sono stati utilizzati diversi manuali o sono state cambiate parole all'interno dello stesso al fine di ottenere una versione del manuale di istruzioni nella quale i nomi dei componenti e possibilmente anche delle azioni eseguite fossero differenti rispetto a quelli utilizzati per la realizzazione dei modelli. Questi particolari test permettono

di valutare la capacità del sistema di trovare corrispondenza tra parole simili e i loro sinonimi.

Lo scopo dei test è quello di mostrare la capacità del sistema di riconoscere ed associare nomi di componenti con nomi di modelli differenti ma rappresentanti lo stesso componente e di non considerare un nome di componente o azione quando questo non ha rispettivamente il modello o l'animazione pre caricata. I test sono stati eseguiti su 3 stampanti dove il sistema per ogni stampante ha utilizzato tutti e 3 i manuali disponibili, adattandoli quando necessario. Per valutare la capacità di analisi del testo per ogni test, quindi per ogni coppia manuale-stampante, sono state annotate in una tabella:

- le frasi utilizzate
- le frasi corrispondenti alle istruzioni che necessitavano di essere animate ottenute tramite un'analisi manuale del testo
- le frasi ottenute dal sistema a seguito della fase di estrapolazione, elaborazione e conversione del testo, ovvero l'istruzione inviata a Blender.
- le percentuali di similarità tra il nome del componente o l'azione con i nomi dei modelli o delle animazioni presenti su Blender e i loro rispettivi sinonimi.

Di seguito vengono riportate le tabelle ottenute dai test effettuati con le stampanti Canon PIXMA-MX495 (Figura 8.11), Epson WF 7010 (Figura 8.12) e HP Deskjet 3000 (Figura 8.13) le tabelle sono organizzate per mostrare dall'alto verso il basso: le frasi all'interno del manuale, le istruzioni derivanti dall'analisi manuale, le istruzioni derivanti dall'analisi del testo, le istruzioni convertite per essere utilizzate da Blender, i nomi dei modelli/animazioni presenti in Blender che corrispondono ai nomi dei componenti/azioni estrapolati dal testo (salvati nella categoria successiva), la percentuale di similarità ottenuta dal confronto, le istruzioni per cui è stato possibile generare un'animazione e quelle per cui è stata selezionata, le istruzioni per cui era presente l'immagine e le istruzione rappresentate correttamente dall'animazione.

Vengono riportate sotto le tabelle contenenti le statistiche estrapolate dalle tabelle sopra citate, per ogni stampante:

HP Deskjet 3000:

% frasi scartate	0,00%
# frasi scartate	0
% istruzioni corrette	90,91%
#istruzioni correttamente mostrate	10
% immagini/istruzioni	63,64%
Media similarità:	89,75%
# istruzioni convertite:	11
# istruzioni testo:	11
% frasi convertite:	100,00%
numero animazioni in Blender	30
numero animazioni attivate	10
% animazioni attivate	33,33%
% animazioni selezionate	90,91%
numero animazioni generate	1
% istruzioni generate	9,09%

Tabella 8.1. La tabella riporta a destra le statistiche esplorate per il test con il primo manuale per la stampante HP Deskjet 3000, mentre a sinistra le percentuali e i valori relativi, si può notare che in questo test sono state animate correttamente la maggior parte delle istruzioni senza scartarne alcuna

% frasi scartate	45,45%
# frasi scartate	5
% istruzioni corrette	100,00%
#istruzioni correttamente mostrate	6
% immagini/istruzioni	27,27%
Media similarità:	84,47%
# istruzioni convertite:	6
# istruzioni testo:	11
% frasi convertite:	54,55%
numero animazioni in Blender	30
numero animazioni attivate	6
% animazioni attivate	20,00%
% animazioni selezionate	100,00%
numero animazioni generate	0
% istruzioni generate	0,00%

Tabella 8.2. La tabella riporta a destra le statistiche esplorate per il test con il secondo manuale per la stampante HP Deskjet 3000, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente tutte le istruzioni che non sono state scartate, ovvero poco più della metà che sono state convertite

% frasi scartate	11,11%
# frasi scartate	1
% istruzioni corrette	100,00%
#istruzioni correttamente mostrate	8
% immagini/istruzioni	55,56%
Media similarità:	88,93%
# istruzioni convertite:	8
# istruzioni testo:	9
% frasi convertite:	88,89%
numero animazioni in Blender	30
numero animazioni attivate	7
% animazioni attivate	23,33%
% animazioni selezionate	87,50%
numero animazioni generate	1
% istruzioni generate	12,50%

Tabella 8.3. La tabella riporta a destra le statistiche esplorate per il test con il terzo manuale per la stampante HP Deskjet 3000, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente tutte le istruzioni e che sono state scartate poco più del 10% delle istruzioni

espson WF-7010:

% frasi scartate	9,09%
# frasi scartate	1
% istruzioni corrette	100,00%
#istruzioni correttamente mostrate	10
% immagini/istruzioni	36,36%
Media similarità:	88,18%
# istruzioni convertite:	10
# istruzioni testo:	11
% frasi convertite:	90,91%
numero animazioni in Blender	30
numero animazioni attivate	9
% animazioni attivate	30,00%
% animazioni selezionate	90,00%
numero animazioni generate	1
% istruzioni generate	10,00%

Tabella 8.4. La tabella riporta a destra le statistiche esplorate per il test con il primo manuale per la stampante Epson WF-7010, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente tutte le istruzioni e che sono state scartate meno del 10% delle istruzioni

% frasi scartate	28,57%
# frasi scartate	4
% istruzioni corrette	90,00%
#istruzioni correttamente mostrate	9
% immagini/istruzioni	42,86%
Media similarità:	85,18%
# istruzioni convertite:	10
# istruzioni testo:	14
% frasi convertite:	71,43%
numero animazioni in Blender	30
numero animazioni attivate	10
% animazioni attivate	33,33%
% animazioni selezionate	100,00%
numero animazioni generate	0
% istruzioni generate	0,00%

Tabella 8.5. La tabella riporta a destra le statistiche esplorate per il test con il secondo manuale per la stampante espson WF-7010, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente il 90% delle istruzioni che sono state scartate circa 1 terzo delle istruzioni

Canon PIXMA-MX495 :

% frasi scartate	12,50%
# frasi scartate	1
% istruzioni corrette	85,71%
#istruzioni correttamente mostrate	6
% immagini/istruzioni	62,50%
Media similarità:	90,77%
# istruzioni convertite:	7
# istruzioni testo:	8
% frasi convertite:	87,50%
numero animazioni in Blender	30
numero animazioni attivate	6
% animazioni attivate	20,00%
% animazioni selezionate	85,71%
numero animazioni generate	1
% istruzioni generate	14,29%

Tabella 8.6. la tabella riporta a destra le statistiche esplorate per il test con il primo manuale per la stampante Canon PIXMA-MX495, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente l'85% delle istruzioni che sono state scartate poco più del 10 %

% frasi scartate	40,00%
# frasi scartate	4
% istruzioni corrette	83,33%
#istruzioni correttamente mostrate	5
% immagini/istruzioni	20,00%
Media similarità:	80,73%
# istruzioni convertite:	6
# istruzioni testo:	10
% frasi convertite:	60,00%
numero animazioni in Blender	30
numero animazioni attivate	5
% animazioni attivate	16,67%
% animazioni selezionate	83,33%
numero animazioni generate	1
% istruzioni generate	16,67%

Tabella 8.7. la tabella riporta a destra le statistiche esplorate per il test con il secondo manuale per la stampante Canon PIXMA-MX495, mentre a sinistra le percentuali e i valori relativi. Si può notare che in questo test sono state animate correttamente circa l'80 % delle istruzioni ma che sono state scartate poco meno della metà delle istruzioni

Questi dati sono stati utilizzati per contare il numero di occorrenze per ogni categoria e per calcolare la media della percentuale di similarità ottenuta nel confronto dei nomi e delle azioni. Sono state in questo modo generate diverse tabelle contenente i dati rilevanti ottenuti tramite i test, questi dati sono stati infine raccolti per generare una media generale che potesse rappresentare la bontà dell'algoritmo di ricerca di similitudine presente nel sistema. I test hanno dimostrato che i nomi dei componenti e delle azioni sono stati riconosciuti con una percentuale di similarità del 86,86% dove per la media sono stati considerati sia i riconoscimenti positivi che negativi. Questo dato va rapportato insieme alla percentuale di frasi scartate tra quelle rilevate durante la fase di analisi del testo e alla percentuale di frasi correttamente convertite tra quelle non scartate, ovvero rispettivamente il 20,96% (Figura 8.14) e 92,85% (Figura 8.15).

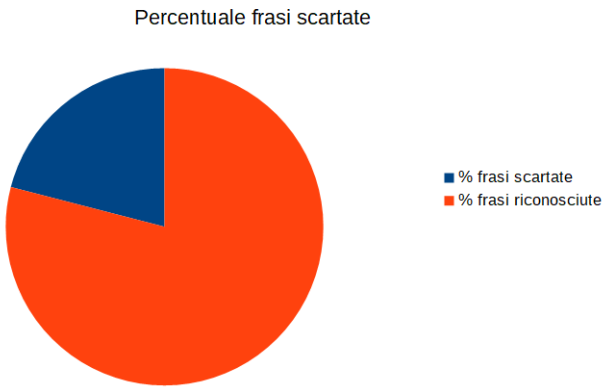

Figura 8.14. Il grafo riporta in arancione la percentuale di istruzioni per cui è stata trovata una frase utilizzabile su Blender per l'animazione dei modelli che il sistema ritenesse accettabile 79,04% mentre in blu la percentuale di istruzioni scartate in quanto non è stato trovato una corretta rappresentazione ovvero il 20,96%

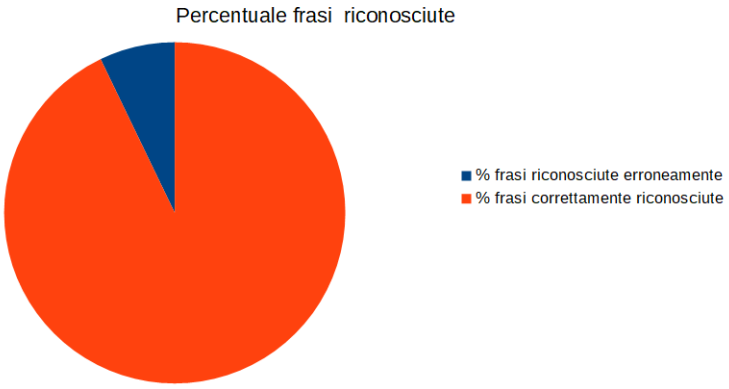

Figura 8.15. Il grafo riporta in arancione la percentuale di istruzioni, tra quelle non scartate, che si sono rivelate corrette per la generazione dell'opportuna animazioni ovvero il 92,85% mentre in blu la percentuale di frasi che si sono rivelate non rappresentati correttamente l'istruzione descritta nel testo ovvero 7,15%

La percentuale di frasi correttamente interpretate viene calcolata come la media delle percentuali di frasi correttamente interpretate per ogni test manuale-stampante. Le singole percentuali poc'anzi citate vengono calcolate come il rapporto tra le animazioni corrette e le animazioni identificate dal sistema. Anche la percentuale di frasi scartate viene calcolata come la media delle singole percentuali ottenute in ogni test manuale-stampante. In questo caso la singola percentuale è

calcolata come il rapporto tra il numero di frasi scartate tra quelle rilevate tramite l'analisi manuale delle frasi. I risultati possono essere interpretati complessivamente positivamente, infatti nella maggior parte dei casi il sistema è capace di ottenere nonostante nomi di componenti e di azioni diversi la corretta animazione da mostrare scartando quando necessario le frasi che non potrebbe animare o per mancanza dei componenti o a causa di una scarsa sicurezza nella scelta del sinonimo. Inoltre in media la percentuale di similarità si mantiene sopra l'85% ciò implica che i sostituti vengono scelti dal sistemi con sicurezza.

Infine per valutare il comportamento generale del sistema nella generazione di un video pertinente alla lista di istruzioni elencate nel manuale di istruzioni, sono stati contati il numero di animazioni che raffiguravano correttamente l'istruzione elencate. Questo numero rapportato con il numero di istruzioni animabili all'interno del testo in ingresso, derivante dall'analisi manuale, premette di ottenere per ogni test la percentuale di animazioni correttamente mostrate. La media delle percentuali ottenute permette di valutare il comportamento generale del sistema, risulta quindi che il 91,43% delle istruzioni da animare presenti all'interno del testo venga correttamente animato. I casi di non riconoscimento sono dovuti ad animazioni non animabili o inclusi in altre animazioni, come nel caso di "Grasp the old toner cartridge and pull it out" che viene semplificata in un'unica azione "pull out the old toner cartridge" e a frasi complesse che sono state semplificate in modo erroneo. In base a quanto evidenziato con i vari test e attraverso l'analisi manuale delle istruzioni, si può concludere che il sistema sviluppato riesca nell'intento di generare un video animato tramite modelli 3D in modo automatico. Una volta ottenuti i modelli 3D e la libreria di animazioni non è necessario da parte dell'utilizzatore del sistema avere conoscenze informatiche per ottenere il video corrispondente alle istruzioni da animare. Inoltre il sistema ricevendo in ingresso un insieme di testi arbitrari riportanti diverse sequenze di istruzioni permette di ottenere con minimo sforzo tutti i video necessari. Lo stesso lavoro, se eseguito manualmente occuperebbe più tempo rispetto a quello necessario tramite l'utilizzo del sistema. Un altro dato rilevante deducibile dai test riguarda la capacità del sistema di utilizzare le immagini al fine di posizionare la camera ed ottenere l'inquadratura riportata nelle immagini e la capacità del sistema di generare in automatico le animazioni. Anche se entrambi ancora limitati dalla qualità del modello dei pesi generato derivante dal database di immagini, mostrano la possibilità da parte del sistema di ridurre il numero delle animazioni da caricare all'interno della libreria e alla possibilità di ottenere viste multiple a partire dalle immagini per ottenere per ogni istruzione l'inquadratura più efficace.

Marca	Serie Stampante	Istruzione Manuale	istruzione presente	Istruzione estrapolata [azione] [componente]	Percentuale Similarità			Animazione generata Automaticamente	Animazione Selezionata dal DB	instruction image	Istruzione Correttamente Mostrata
					Blender OBJ	Text Component	Similarity %				
canon	PIXMA-MX495	Make sure the light is on	Make sure light is on	you [turn on] [on button]	on button	light	85,56%				
					push	is on	84,28%		x		x
		Open the printer cover	open printer cover	you [open] [front cover]	front cover	printer cover	100,00%				
					open	open	100,00%	x		presente	x
		Shake the newcartridge before opening it	shake new cartridge	you [shake] [newfine cartridge]	newfine cartridge	newcartridge	91,61%				
					shake	shake	100,00%		x	presente	x
			open newcartridge		newfine cartridge	newcartridge	91,61%				
					insert	open	53,16%				
		Remove the cartridge from the package	remove cartridge	you [remove] [empty fine cartridge]	empty fine cartridge	cartridge	75,57%				
					remove	remove	100,00%		x	presente	x
		Open the cartridge cover	Open cartridge cover	you [open] [front cover]	front cover	cartridge cover	78,91%				
					open	open	100,00%		x	presente	
		Place the newcartridge into the cartridge holder	Place newcartridge	you [insert] [newfine cartridge]	newfine cartridge	newcartridge	91,61%				
					insert	place	100,00%		x		x
		When you are finished, close the printer cover	close printer cover	you [close] [front cover]	front cover	printer cover	100,00%				
					close	close	100,00%		x	presente	x
		7	8	7			90,77%	1	6	5	6
canon	PIXMA-MX495	Open the printer cover	open front cover	you [open] [front cover]	front cover	printer cover	84,75%				
					open	open	100,00%	x		presente	x
		open the packet	open packet	you [open] [package]	package	packet	100,00%				
					open	open	100,00%		x		x
		Unpack the newtoner cartridge	unpack newtoner cartridge		newfine cartridge	newtoner cartridge	86,27%				
					take	shake	40,81%				
		Shake the newtoner cartridge	*shake newtoner cartridge*	you [shake] [newfine cartridge]	newfine cartridge	newtoner cartridge	86,27%		x		x
					shake	shake	100,00%				
		Open the cartridge door	Open cartridge cover		newfine cartridge	cartridge door	70,02%				
					insert	open	48,55%				
		grasp the old toner cartridge and pull it out from its slot	grasp old toner cartridge		empty ink cartridge	old toner cartridge	82,22%				
					push down	grasp	45,61%				
			pull out old toner cartridge		empty ink cartridge	old toner cartridge	82,22%			presente	
					remove	pull out	76,35%				
		push the newtoner cartridge into its slot	push newtoner cartridge	you [insert] [newfine cartridge]	newfine cartridge	newtoner cartridge	86,27%				
					insert	push	84,85%		x		x
		Close the cartridge cover	Close cartridge cover	you [close] [front cover]	front cover	cartridge cover	78,91%				
					close	close	100,00%		x		
		Close the printer cover	Close front cover	you [close] [front cover]	front cover	printer cover	84,75%				
					close	close	100,00%		x		x
		9	10	6			81,89%	1	5	2	5

Figura 8.11. La figura mostra la tabella realizzata tramite i test con 2 diversi manuali sul modello realizzato per la stampante Canon PIXMA-MX495. In ogni test il modello conserva le stesse azioni e gli stessi nomi dei componenti

Marca	Serie Stampante	Istruzione Manuale	istruzione presente	Istruzione estrapolata [azione] [componente]	Percentuale Similarità			Animazione generata Automaticamente	Animazione Selezionata dal DB	instruction image	Istruzione Correttamente Mostrata
					Blender OBJ	Text Component	Similarity %				
epson	WF-7010	Make sure the power is turned on	Make sure power is turned on	you {turn on} {light}	light	power	100,00%				
					turn on	turned on	100,00%		x		x
		Open the front cover (A).	Open front cover	You [open] [printer cover]	printer cover	front cover	84,75%				
					open	open	100,00%		x		x
		Shake the new FINE cartridge before Opening the packet	shake new FINE cartridge	You [shake] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%		x	presente	x
					shake	shake	100,00%				
			open packet	You [open] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%		x		x
					open	opening	100,00%				
		Remove the new FINE cartridge from the package	remove new FINE cartridge from package	You [remove] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%		x		x
					remove	remove	100,00%				
		Open the cartridge door	open cartridge door	you {open} {cartridge cover}	cartridge cover	cartridge door	76,13%		x	presente	x
					open	open	100,00%				
		Remove the old FINE cartridge and discard it	*remove old FINE cartridge*		old ink cartridge	Old FINE cartridge	74,60%				
					lift	remove	76,78%				
			discard old FINE cartridge	You [dispose] [old ink cartridge]	old ink cartridge	Old FINE cartridge	74,60%		x		x
					dispose	discard	100,00%				
		Place the new FINE cartridge into the cartridge holder	place new FINE cartridge Into cartridge holder	You [place] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%	x		presente	x
					place	place	100,00%				
		When you are finished, close the cartridge door And the front cover	close cartridge door	You [close] [cartridge cover]	cartridge cover	cartridge door	76,13%		x	presente	x
					close	close	100,00%				
			close front cover	You [close] [printer cover]	printer cover	front cover	84,75%		x		x
					close	close	100,00%				
		8	11	10			88,18%	1	9	4	10
epson	WF-7010	If the product is not already on, press The Power button to turn it on.	press the power button to turn it on	you {turn on} {light}	light	power button	100,00%				
					turn on	turned on	100,00%		x		x
		Open the front cover	Open front cover	You [open] [printer cover]	printer cover	front cover	84,75%				
					open	open	100,00%		x		x
		Open the output tray, and then Pull the tray extender out.	open output tray		output tray	output tray	100,00%				x
					slide out	open	41,87%				
			pull out the tray extender		output tray extender	tray extender	75,66%				x
					close	pull out	62,42%				
		shake the new cartridge four or five times Before opening the packet	shake new cartridge	You {shake} {new ink cartridge}	new ink cartridge	new cartridge	71,55%		x	presente	x
					shake	shake	100,00%				
			open the packet	you {open} {package}	package	packet	100,00%		x		
					open	open	100,00%				
		Remove the new cartridge from its packet	Remove new cartridge	You [remove] [*new ink cartridge*]	new ink cartridge	new cartridge	71,55%		x		x
					remove	remove	100,00%				
		Remove the plastic protective tape from the new cartridge	*remove plastic protective tape*	you {remove} {yellow tape}	plastic protective tape	yellow tape	73,38%		x	presente	
					remove	remove	100,00%				
		Open the ink cartridge door	open ink cartridge door	you {open} {cartridge cover}	cartridge cover	ink cartridge door	79,62%		x	presente	x
					open	open	100,00%				
		Push down lightly on the old cartridge that you want to rep To release it, and then pull it toward you out of its slot	push down old cartridge		old ink cartridge	old cartridge	72,60%				
					lift	push down	78,49%				
			pull old cartridge out its slot		old ink cartridge	old cartridge	72,60%			presente	
					dispose	pull	68,27%				
		Place the new cartridge into its slot until it snaps firmly into pl	put new cartridge into slot	You [place] [new ink cartridge]	new ink cartridge	new cartridge	71,55%		x	presente	
					place	put	100,00%				
		After replacing cartridges, close the ink cartridge door a	*ink cartridge door*	You [close] [cartridge cover]	cartridge cover	cartridge door	76,13%		x	presente	x
					close	close	100,00%				
			close front cover	You [close] [printer cover]	printer cover	front cover	84,75%		x		x
					close	close	100,00%				
		10	14	10			85,16%	0	10	6	9

Figura 8.12. La figura mostra la tabella realizzata tramite i test con 2 diversi manuali sul modello realizzato per la stampante Epson WF 7010. In ogni test il modello conserva le stesse azioni e gli stessi nomi dei componenti

Marca	Serie Stampante	Istruzione Manuale	istruzione presente	Istruzione estrapolata [azione] [componente]	Percentuale Similarità			Animazione generata Automaticamente	Animazione Selezionata dal DB	instruction image	Istruzione Correttamente Mostrata
					Blender OBJ	Text Component	Similarity %				
HP	HP Deskjet 3000	Make sure the light is on	turn on the light	you [press] [power button]	power button	light	100,00%				
					press	is on	100,00%		x		x
		open the document tray	open document tray	you [open] [output tray]	output tray	document tray	78,24%				
					open	open	100,00%		x	presente	
		Open the paper output tray, and then Pull the output tray extension out.	Open the output tray,	you [open] [output tray]	output tray	paper output tray	73,31%				
					open	open	100,00%		x		x
			Pull the output tray extension out	you [push down] [old ink cartridge]	tray extender	output tray extension	73,47%				
					pull down	pull down	100,00%		x		x
		Load the document in the tray	Load document	you [load] [paper]	paper	document	100,00%				
					load	load	100,00%		x	presente	x
		Open the cartridge cover	open cartridge cover	you [pull out] [old ink cartridge]	ink cartridge door	cartridge cover	80,27%				
					open	open	100,00%		x	presente	x
		remove the old cartridge out of its slot	remove old cartridge	you [remove] [new ink cartridge]	old ink cartridge	old cartridge	72,52%				
					remove	remove	100,00%		x		x
		Remove the new cartridge from its package	Remove new cartridge	you [remove] [plastic protective tape]	new ink cartridge	new cartridge	71,55%				
					remove	remove	100,00%		x	presente	x
		Remove the yellow tape from the new ink cartridge	Remove yellow tape	you [slide] [new ink cartridge]	plastic protective tape	yellow tape	73,38%				
					remove	remove	100,00%	x		presente	x
		Place the new cartridge into its slot	Place new cartridge	you [close] [output tray]	new ink cartridge	new cartridge	71,55%				
					slide	place	100,00%		x	presente	x
		Close the cartridge cover	close cartridge cover	you [close] [output tray]	ink cartridge door	cartridge cover	80,27%				
					close	close	100,00%		x	presente	x
		10	11	11			89,75%	1	10	7	10
HP	HP Deskjet 3000	lift the front cover	lift front cover		ink cartridge door	front cover	59,36%				
						lift					
		Unpack the new toner cartridge	Unpack new toner cartridge		new ink cartridge	new toner cartridge	91,96%				
					remove	unpack	35,37%			presente	
		Gently shake the new toner cartridge from side to side and front to back to distribute toner	shake new toner cartridge	you [shake] [new ink cartridge]	new ink cartridge	new toner cartridge	91,96%				
					shake	shake	100,00%		x		x
		Open the cartridge cover	Open cartridge cover	you [open] [ink cartridge door]	ink cartridge door	cartridge cover	80,27%				
					open	open	100,00%		x		x
		grasp the old toner cartridge and pull it out from its slot	grasp old toner cartridge		old ink cartridge	old toner cartridge	92,46%				
					pull out	grasp	50,56%				
			pull out old toner cartridge	you [pull out] [old ink cartridge]	old ink cartridge	old toner cartridge	92,46%				
					pull out	pull out	100,00%		x		x
		Remove the tape from the new toner cartridge	Remove tape	you [remove] [plastic protective tape]	plastic protective tape	tape	75,35%				
					remove	remove	100,00%		x	presente	x
		align the sides of the new toner cartridge With the guides in the printer	align sides of the new toner cartridge	you [align] [new ink cartridge]	new ink cartridge	new toner cartridge	91,96%				
					align	align	100,00%		x		x
		push the new toner cartridge into its slot	push new toner cartridge		new ink cartridge	new toner cartridge	91,96%				
					slide	push	71,01%			presente	
		Close the cartridge cover	Close cartridge cover	you [close] [ink cartridge door]	ink cartridge door	cartridge cover	80,27%				
					close	close	100,00%		x		x
		Close the front cover	Close front cover		ink cartridge door	front cover	59,36%				
					close	close	100,00%				
		10	11	6			84,02%	0	6	3	6
HP	HP Deskjet 3000	Make sure that the power is turned on.	Make sure power is turned on	you [press] [power button]	power button	power	100,00%				
					press	turned on	100,00%		x		x
		Open the front cover (A).	Open front cover		ink cartridge door	front cover	59,36%				
					open	open	100,00%				
		Open the paper output cover	Open paper output cover	you [open] [output tray]	output tray	paper output cover	73,31%				
					open	open	100,00%		x	presente	x
		push down the empty FINE cartridge Until it clicks	push down empty fine cartridge	you [push down] [old ink cartridge]	old ink cartridge	empty FINE cartridge	81,85%				
					push down	push down	100,00%		x	presente	x
		Remove the empty FINE cartridge	Remove empty fine cartridge	you [pull out] [old ink cartridge]	old ink cartridge	empty FINE cartridge	81,85%				
					pull out	remove	91,81%		x		x
		Take a new FINE cartridge out of its Package	Take new fine cartridge out its package	you [remove] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%				
					remove	take	100,00%		x	presente	x
		remove the protective tape (D) gently	remove gently protective tape	you [remove] [plastic protective tape]	plastic protective tape	protective tape	93,20%				
					remove	remove	100,00%		x	presente	x
		Insert the new FINE cartridge at a slant Into the FINE cartridge holder	Insert new fine cartridge Into fine cartridge holder	you [slide] [new ink cartridge]	new ink cartridge	new FINE cartridge	73,05%				
					slide	insert	100,00%	x		presente	x
		Close the paper output cover	Close paper output cover	you [close] [output tray]	output tray	paper output cover	73,31%				
					close	close	100,00%		x		x
		9	9	8			88,93%	1	7	5	8

Figura 8.13. La figura mostra la tabella realizzata tramite i test con 3 diversi manuali sul modello realizzato per la stampante HP Deskjet 3000. In ogni test il modello conserva le stesse azioni e gli stessi nomi dei componenti

Capitolo 9

Conclusione e lavori futuri

In questo lavoro di tesi è stato sviluppato un sistema che permettesse di generare in modo automatico un'animazione 3D sulla base di testo espresso in linguaggio naturale e immagini. In particolare, il caso c'uso oggetto del test del sistema ha riguardato la generazione di istruzioni per sostituire le cartucce di una stampante. Le istruzioni testuali e le immagini date in input al sistema erano quelle contenute nel manuale della stampante.

In particolare, per la generazione del video il sistema sfrutta in modo congiunto una libreria contenente un numero limitato di animazioni dei modelli 3D e un algoritmo per la generazione automatica delle animazioni a partire dalle immagini. I vantaggi di questo sistema consistono nel fatto che, basandosi su un dizionario semantico, è in grado di associare ai modelli 3D e alle relative animazioni anche quegli elementi, contenuti nel manuale di istruzioni, che sono stati riportati con nomi diversi da quelli utilizzati per etichettare i relativi modelli 3D. Inoltre, la generazione automatica delle animazioni a partire dalle immagini fa sì che venga mostrata all'utente un'animazione anche qualora questa non fosse presente all'interno del database.

Per il confronto semantico dei nomi dei componenti, il sistema si basa sulla similarità in termini di vettore di parola, dipendenza, parte del discorso e contesto per ogni nome/azione e per i suoi sinonimi.

Dato che i nomi utilizzati all'interno dei manuali sono nomi tecnici, che non trovano spesso corrispettivi all'interno di dizionari semantici, è stato adottato il seguente approccio, per massimizzare le possibilità di trovare un match corretto: laddove possibile, i nomi dei componenti sono stati inizialmente scomposti per ottenere dei nomi semplici per i quali è possibile trovare dei sinonimi. In un secondo tempo, i sinonimi sono stati ricomposti per ottenere delle parole con la medesima composizione della parola di partenza in modo da creare un plausibile sinonimo della parola iniziale.

Per quel che riguarda la generazione dell'animazione in modo automatico e la posa della camera all'interno dell'ambiente di Blender, esse si basano sulla bontà del riconoscimento dei componenti all'interno dell'immagine. Al momento della realizzazione di questo studio non è presente online e liberamente accessibile un database contenente immagini di stampanti annotate. E' stato quindi necessario procedere alla sua creazione manuale sfruttando immagini presenti online provenienti da diversi manuali di istruzioni. In particolare, il database realizzato contiene circa 500 immagini annotate. Visto l'esiguo numero di immagini, ne consegue che durante la fase di analisi il sistema non è detto che identifichi tutti i componenti presenti all'interno dell'immagine e che le maschere identificate non risultino accurate. Ciò causa uno scarso utilizzo dell'algoritmo di generazione delle animazioni in automatico e errori nella posa della camera all'interno dell'ambiente.

Per ovviare al problema degli errori nella posa della camera, il sistema, una volta ottenuto una ipotetica posizione della camera, identifica tra le camere la cui inquadratura è stata definita in anticipo quella più vicina alla posizione della camera calcolata precedentemente e la seleziona come camera attiva la cui inquadratura verrà utilizzata per la realizzazione del video. Il sistema riesce a estrarre dal testo la maggior parte delle istruzioni a convertirle ottenendo una buona percentuale di similarità e riesce a produrre l'animazione corretta nella maggior parte dei casi.

Come lavoro futuro si può considerare la possibilità di applicare applicare tecniche di machine learning al testo e di effettuare test con gli utenti per valutare l'usabilità e verificare se il sistema permette di migliorare la capacità di comprensione delle informazioni contenute in un testo. Un ulteriore lavoro futuro consiste nell'implementare un'interfaccia grafica che permetta all'utente non esperto di interagire col sistema inserendo facilmente le risorse richieste (testo, immagini, modelli e libreria di animazioni).

Bibliografia

[1] Y. Zhang, E. Tsipidi, S. Schriber, M. Kapadia, M. Gross, A. Modi, Generating animations from screenplays, 2019.

[2] M. Marti, J. Vieli, W. Witoń, R. Sanghrajka, D. Inversini, D. Wotruba, I. Simo, S. Schriber, M. Kapadia, M. Gross, Cardinal: Computer assisted authoring of movie scripts, in: 23rd International Conference on Intelligent User Interfaces, 2018, pp. 509–519.

[3] G. Ali, M. Lee, J.-I. Hwang, Automatic text-to-gesture rule generation for embodied conversational agents, Computer Animation and Virtual Worlds 31 (4-5) (2020) e1944.

[4] H. Subramonyam, W. Li, E. Adar, M. Dontcheva, Taketoons: Script-driven performance animation, in: Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology, 2018, pp. 663–674.

[5] P. Yadav, K. Sathe, M. Chandak, Generating animations from instructional text, International Journal 9 (3) (2020).

[6] M. H. Joseph, et al., Signar: A sign language translator application with augmented reality using text and image recognition, in: 2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), IEEE, 2019, pp. 1–5.

[7] K. Hassani, W.-S. Lee, Visualizing natural language descriptions: A survey, ACM Computing Surveys (CSUR) 49 (1) (2016) 1–34.

[8] A. Armando, P. Pecchiari, Nalig: A cad system for interior design with high level interaction capabilities, in: Proceedings of 1993 IEEE Conference on Tools with Al (TAI-93), IEEE, 1993, pp. 446–447.

[9] N. I. Badler, R. Bindiganavale, J. C. Bourne, M. S. Palmer, J. Shi, W. Schuler, A parameterized action representation for virtual human agents (2000).

[10] E. Hanser, P. Mc Kevitt, T. Lunney, J. Condell, Scenemaker: automatic visualisation of screenplays, in: Annual Conference on Artificial Intelligence, Springer, 2009, pp. 265–272.

[11] E. Hanser, P. Mc Kevitt, T. Lunney, J. Condell, M. Ma, Scenemaker: multimodal visualisation of natural language film scripts, in: International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, Springer, 2010, pp. 430–439.

[12] O. Åkerberg, H. Svensson, B. Schulz, P. Nugues, Carsim: an automatic 3d text-to-scene conversion system applied to road accident reports, in: Demonstrations, 2003.

[13] Z.-Q. Liu, K.-M. Leung, Script visualization (scriptviz): a smart system that makes writing fun, Soft Computing 10 (1) (2006) 34–40.

[14] M. Ma, Automatic conversion of natural language to 3d animation, Ph.D. thesis, University of Ulster (2006).

[15] K. Hassani, A. Nahvi, A. Ahmadi, Architectural design and implementation of intelligent embodied conversational agents using fuzzy knowledge base, Journal of Intelligent & Fuzzy Systems 25 (3) (2013) 811–823.

[16] K. Hassani, A. Nahvi, A. Ahmadi, Design and implementation of an intelligent virtual environment for improving speaking and listening skills, Interactive Learning Environments 24 (1) (2016) 252–271.

[17] huggingface /neuralcoref, https://github.com/huggingface/neuralcoref (2021).

[18] spacy.io, https://spacy.io (2021).

[19] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).

[20] G. A. Miller, Wordnet: a lexical database for english, Communications of the ACM 38 (11) (1995) 39–41.

[21] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th annual meeting of the Association for Computational Linguistics, 2002, pp. 311–318.

[22] Stanford corenlp: a suite of core nlp tools, http://stanfordnlp.githubjo/CoreNLP/. (2021).

[23] A. Shoulson, N. Marshak, M. Kapadia, N. I. Badler, Adapt: the agent developmentand prototyping testbed, IEEE Transactions on Visualization and Computer Graphics 20 (7) (2013) 1035–1047.

[24] Trelby the free, multiplatform, feature-rich screenwriting program!, https://www.trelby.org/ (2016).

[25] C.-Y. Chen, S.-K. Wong, W.-Y. Liu, Generation of small groups with rich behaviors from natural language interface, Computer Animation and Virtual Worlds 31 (4-5) (2020) e1960.

[26] W.-Y. Liu, S.-K. Wong, C.-Y. Chen, A natural language interface with casual users for crowd animation, Computer Animation and Virtual Worlds 31 (4-5) (2020) e1965.

[27] G. Ali, M. Lee, J.-I. Hwang, Automatic text-to-gesture rule generation for embodied conversational agents, Computer Animation and Virtual Worlds 31 (4-5) (2020) e1944.

[28] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[29] G. Pais, F. Deloule, D. Beauchêne, P. Lambert, Animated movie activity characterization by image and text information fusion, in: 2009 12th International Conference on Information Fusion, IEEE, 2009, pp. 1068–1075.

[30] R. Dimov, M. Feld, D. M. Kipp, D. A. Ndiaye, D. D. Heckmann, Weka: Practical machine learning tools and techniques with java implementations, AI Tools SeminarUniversity of Saarland, WS 6 (07) (2007).

[31] D. Ferrés, M. Marimon, H. Saggion, A. AbuRa'ed, Yats: yet another text simplifier, in: International Conference on Applications of Natural Language to Information Systems, Springer, 2016, pp. 335–342.

[32] F. N. A. Al Omran, C. Treude, Choosing an nlp library for analyzing software documentation: a systematic literature review and a series of experiments, in: 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), IEEE, 2017, pp. 187–197.

[33] S. Petrov, Announcing syntaxnet: The world's most accurate parser goes open source, Google Research Blog 12 (2016).

[34] Natural language toolkit nltk 3.0 documentation, http://www.nltk.org (2021).

[35] Scenegraphparser, https://github.com/vacancy/SceneGraphParser (2020).

[36] H. Wu, J. Mao, Y. Zhang, Y. Jiang, L. Li, W. Sun, W.-Y. Ma, Unified visual-semantic embeddings: Bridging vision and language with structured meaning representations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6609–6618.

[37] Nlp scene graph parser, https://nlp.stanford.edu/software/scenegraph-parser.shtml (2016).

[38] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[39] R. Rehurek, P. Sojka, Software framework for topic modelling with large corpora, in: In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks, Citeseer, 2010.

[40] G. Bradski, The opencv library., Dr. Dobb's Journal: Software Tools for the Professional Programmer 25 (11) (2000) 120–123.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[42] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al., Visual genome: Connecting language and vision using crowdsourced dense image annotations, International journal of computer vision 123 (1) (2017) 32–73.

[43] Epoch vs batch size vs iterations, https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9 (2021).

[44] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467 (2016).

[45] A. Dutta, A. Gupta, A. Zissermann, Vgg image annotator (via), URL: http://www. robots. ox. ac. uk/˜ vgg/software/via (2016).

[46] Blender, https://www.blender.org/ (2021).

[47] Basehttprequesthandler, https://docs.python.org/3/library/http.server.html#http.server.BaseHTTPRequestHandler (2021).

[48] Camera calibration and 3d reconstruction, https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html (2021).

[49] Metodo rodrigues, https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga61585db663d9da06b68e70cfbf6a1eac (2021).

[50] Blender orientation, https://docs.blender.org/manual/en/2.92/editors/3dview/controls/orientation.html (2021).

[51] Open cv tutoria real time pose, https://docs.opencv.org/4.5.1/dc/d2c/tutorial_real_time_pose.html (2021).