**Alma Mater Studiorum – Università di Bologna**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

**TESI DI LAUREA**

In

Calcolatori Elettronici – T

Creazione applicazione per il tracking

di persone e oscuramento volti.

CANDIDATO: Relatore:

Francesco Luzzi Prof. Stefano Mattoccia

Correlatore:

Alessio Mingozzi

Sessione

Anno Accademico 2020/21

Sommario

[1 Introduzione 3](#_Toc86830681)

[2 Strumenti utilizzati 3](#_Toc86830682)

[2.1 CMake 3](#_Toc86830683)

[2.2 OpenPose 3](#_Toc86830684)

[2.3 OpenCv 3](#_Toc86830685)

[2.4 NumPy 3](#_Toc86830686)

[3 Sviluppo 4](#_Toc86830687)

[3.1 Input 4](#_Toc86830688)

[3.2 Logica 4](#_Toc86830689)

[3.3 Primi output 4](#_Toc86830690)

[3.4 Modifiche dei threshold 4](#_Toc86830691)

[3.5 Nuovi output 4](#_Toc86830692)

[4 Conclusione 4](#_Toc86830693)

# Introduzione

Alla base di questo progetto vi è la necessità di nascondere l’identità, in modo automatico, delle persone presenti all’interno di una fotografia o di un video.   
Questa rete non riconosce le persone, semplicemente ne riconosce la posa di ognuna di esse concentrandosi maggiormente sulla posizione del volto (naso-collo) così da poterlo oscurare.

Bla bla bla approfondimento di questo problema mi ha spinto a creare questa applicazione…

Bla bla bla la presentazione sarà suddivisa in 5 capitoli:  
 -…

# Strumenti utilizzati

Gli strumenti utilizzati per questo progetto sono stati:

## CMake

CMake is an extensible, open-source system that manages the build process in an operating system and in a compiler-independent manner. Unlike many cross-platform systems, CMake is designed to be used in conjunction with the native build environment. Simple configuration files placed in each source directory (called CMakeLists.txt files) are used to generate standard build files (e.g., makefiles on Unix and projects/workspaces in Windows MSVC) which are used in the usual way. CMake can generate a native build environment that will compile source code, create libraries, generate wrappers and build executables in arbitrary combinations. CMake supports in-place and out-of-place builds, and can therefore support multiple builds from a single source tree. CMake also supports static and dynamic library builds. Another nice feature of CMake is that it generates a cache file that is designed to be used with a graphical editor. For example, when CMake runs, it locates files, libraries, and executables, and may encounter optional build directives. This information is gathered into the cache, which may be changed by the user prior to the generation of the native build files.

CMake is designed to support complex directory hierarchies and applications dependent on several libraries.

## OpenPose

Sfruttando il machine learning è stata creata una rete neurale che passata come input un’immagine restituirà una mappa di probabilità per diverse parti del corpo.

Questa specifica rete utilizza un formato di output delle articolazioni chiamato MPI, questa reta ritorna anche mappe di probabilità per le diverse “connessioni” tra le diverse articolazioni, molto utili per riconoscere di chi è il preciso keypoint.

## OpenCv

Libreria open source praticamente essenziale nel panorama della computer vision, sviluppata in C++ e resa disponibile tramite un wrapper anche su python. L’implementazione da me utilizzata è leggermente modificata, è stata compilata con CMake utilizzando il codice della versione “originale” di opencv ed il codice della versione “contrib”, tutto ciò per abilitare/sfruttare l’accelerazione GPU data da CUDA (difatti la mia gpu è di marca Nvidia) che è molto efficiente per questo tipo di workload di tipo algerico/matriciale.

## NumPy

Libreria OpenSource indispensabile per fare operazioni algebriche in python con overhead e durata dell’esecuzione minori, di fatto è una libreria in C con un wrapper ed è molto veloce e performante.

# Sviluppo

Immagini del codice, con spiegazione semplicistica di ciò che sta succedendo

## Input

Immagini utilizzate per il testing e problemi con questa rete neurale.

## Logica

Immagini del codice con immagini esplicatorie della logica dietro la probMap, il ritrovamento dei keypoints e l’utilizzo dei PAF per legare e collegare i diversi Keypoints, poi da collegare alla stessa persona. (spiegazione per quale motivo usare un algoritmo che lega diversi keypoint a seconda della distanza è sbagliato)

## Primi output

Immagini di output degli input con spiegazione della problematica su immagine 000586

## Modifiche dei threshold

Spiegazione della modifica dei valori di threshold per la rilevazione dei keypoints, specificando che solitamente i keypoint meglio riconosciuti sono quelli del collo e del naso/viso, quindi anche perdere qualche keypoint dei piendi non ci interessa allo scopo dell’applicazione

## Nuovi output

Nuove immagini di output degli input con commento sui cambiamenti dell’output e sull’aumento delle prestazioni (dati i minori keypoints “inutili” registrati)

# Conclusione

Possibili ambiti di utilizzo e possibili migliorie, utilità di questa applicazione.

Ringraziamenti ed addio.