TVPR: Top View Person Re-identification

Adel Massimo Ramadan, adel.massimo.ramadan@gmail.com
Riccardo Reali, finokkio@gmail.com
Ciolini Alberto, albeciolo@stud.unifi.it

X agosto 2017

Contents

List of Algorithms

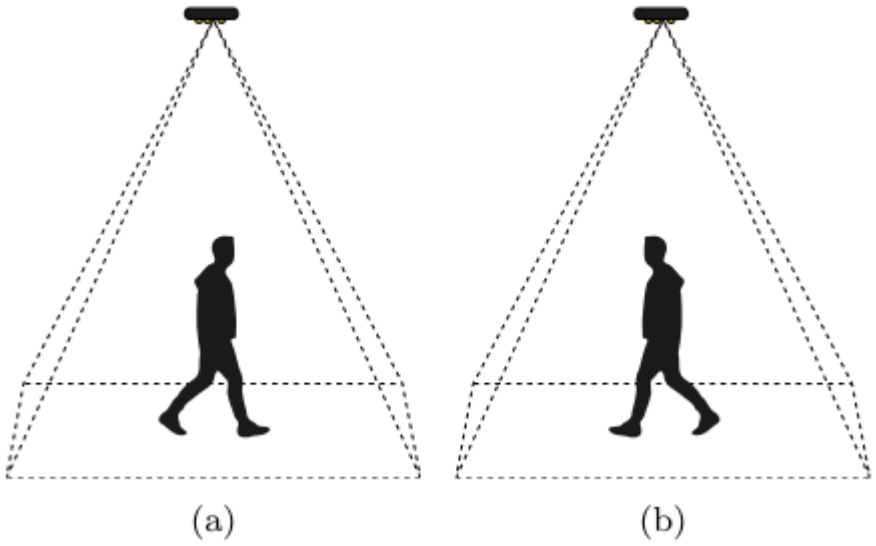


Figure 1: Mockup di un possiblie contesto: la camera, in alto, riprende la scena. La stessa persona passa nella scena (a) e nella (b).

1 Introduzione

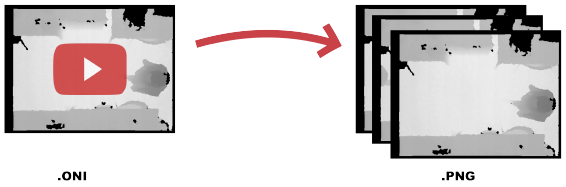
Un importante compito per sistemi muti-camera distribuiti è il riconoscimento di un individuo in scene diverse, a tempi diversi: questo è noto come problema di *Person Re-identification*.
La Person Re-identification è un'importante disciplina di settori come *Human Computer Interaction*, *Screen Monitoring*, *Ambient Assisted Living* e molti altri rami di ricerca della *Computer Vision*. Si possono avere casi d'uso *Online* o *Offline*: nel primo caso l'analisi sul soggetto è immediata; mentre nel secondo caso si analizza la scena in differita (spesso avendo a disposizione una quantità di informazioni maggiore).
Nel caso studiato il contesto è quello di un sistema offline, basato su un sistema di *depht cameras*[1]: le osservazioni di questo sistema vanno a comporre il dataset *TVPR*[2]. In questo dataset sono presenti 23 video, ognuno nel formato 640x480, a un frame rate prossimo a 30fps, e disponibile in formato non compresso (*.ONI*); in questi video compaiono 100 persone nel loro normale modo di fare quotidiano.

[1]Asus Xtion Pro Live | https://www.asus.com/3D-Sensor
[2]Università Politecnica delle Marche: http://vrai.dii.univpm.it/re-id-dataset

2 Obiettivi

Il principale obiettivo di questo progetto consiste nella realizzazione di un sistema in grado di stabilire se, dato un soggetto osservato al tempo t_0, questo sia presente o meno in un altra osservazione effettuata al tempo t_1. Il sistema dovrà essere in grado di effettuare questi confronti avvalendosi unicamente delle immagini di profondità, si tratta quindi di un argomento ancora poco studiato dato che numerosi studi condotti sin'ora si sono avvalsi anche del dato relativo ai frame a colori.
Diventa quindi obiettivo anche il creare una buona base di partenza per un problema del genere, ed a prescindere dai risultati sarà un traguardo anche una consistente e strutturata analisi preliminare del problema.
La metrica per determinare la similarità tra due persone sarà scelta a piacere tra *3D - Shape Context*[3] e *3D - Shape Context*[4].

[3]–riferimento
[4]–riferimento



3 Analisi

In questa sezione il tema affrontato sarà la trattazione dei dati a disposizione: l'elaborazione di questi e la definizione dei modelli a cui sarà applicata la strategia risolutiva.

3.1 OpenNI file reader

Il dataset a disposizione comprende, come già introdotto, 23 filmati: ognuno di questi è presente sia in formato *.avi* che in formato *.oni*. Il primo come sappiamo comporta una compressione di tipo lossy, mentre il secondo è un tipo di dato raw, quindi diventa preferibile lavorare con quest'ultimo per non subire perdite di informazioni dovute alla compressione.
Questo formato conserva informazioni relative al frame di profondità (depth frame), RGB e IR (InfraRed frame). Il dato che risulta di specifico interesse per il nostro scopo è quello relativo alla profondità. Per leggere questo formato sono disponibili delle API disponibili su *Structure.io*[5]. Grazie a queste è possibile scrivere il codice, riportato in Algorithm2 che consente la lettura dei frame di profondità dal dataset e la scrittura di questi come PNG.

```
import numpy as np
from openni import openni2
import png


path = "/Volumes/Volume/Top-view-TVPR/"
prefix = "g0"

for i in range(1, 23):
```

[5]https://structure.io/openni

```
filename = prefix + str(i)
openni2.initialize()
dev = openni2.Device(path+filename+".oni")
print dev.get_device_info()

depth_stream = dev.create_depth_stream()
total_frames = depth_stream.get_number_of_frames()
print "total frames: "+str(total_frames)

for i in range(total_frames):
   depth_stream.start()
   frame = depth_stream.read_frame()
   depth_stream.stop()
   frame_data = frame.get_buffer_as_uint16()
   save_frame(frame_data, i)

openni2.unload()
```

Algorithm 1: Oni to PNG converter

È importante sottolineare che i depth frame sono codificati in scala di grigi a 16 bit, ma nonostante ciò il range effettivo dei colori è circa 7-3500. Non si usano perciò 16 bit effettivi (ne basterebbero 12): questo è dovuto al fatto che il sensore utilizza 1 livello per ogni millimetro, ed essendo il campo raggiungibile tra 7mm e 3,5m ne segue che la banda utilizzata sia molto ridotta.
Es: un punto a 2 metri dal sensore sarà rappresentato con il valore 2000.

3.2 Clasificare le Persone

```
path = '../img/g001';

person_folder2 = '';
frame_names = dir(strcat(path,'/*.png')); %salvo qui tutti i frame

bg_frame = imread( strcat(path,'/frame00000.png') ); %assegno frame di backGround
bg_sum = sum(bg_frame(:));%sommo tutti i valori dei pixel di background

bound = bg_sum * 0.99;
max_interruption = 9;
person_count = 0;
same_person = false;
```

```
empty_frames = 1;
min_frames_per_person = 20;

for frame_name = frame_names'
    path_to_frame = strcat(frame_name.folder,'/', frame_name.name);
    frame = imread(path_to_frame);

    frame_sum = sum(frame(:));
    if(frame_sum < bound)
        empty_frames = 0;

        if(~same_person)

            person_count = person_count + 1;

            person_folder = strcat('/person', num2str(person_count));
            mkdir(path, person_folder);

            same_person = true;
            person_folder2 = strcat( strcat(path, person_folder), '/');
            imwrite( bg_frame, strcat(person_folder2, 'background.png') );
        end

        imwrite( frame, strcat(person_folder2, frame_name.name) );

    else
        if(empty_frames < max_interruption)
            empty_frames = empty_frames + 1;

            if(same_person)
                imwrite( frame, strcat(person_folder2, frame_name.name) );
            end
        else
            frames_detected = max(size(dir(strcat(person_folder2,'*.png'))));
            if (frames_detected < min_frames_per_person && same_person)
                rmdir(person_folder2, 's');
                person_count = person_count-1;
            end
            same_person = false;
        end
    end
end
```

Algorithm 2: Oni to PNG converter

3.3 Central Frame Detection

3.4 Normalizzare le Persone