Elaborazione di dati tridimensionali: homework 2

Alberto Cenzato 1134707

March 28, 2018

Abstract

Scopo di questo secondo homework era di familiarizzare con la libreria PCL. Sono stati svolti quattro esercizi di complessità crescente per approfondire diverse funzionalità di PCL: lettura di point cloud da file, lettura e visualizzazione dei valori contenuti nella point cloud, downsampling, calcolo di normali, keypoint e descrittori, registrazione di point cloud, people detection.

1 Lab 1

La prima esperienza era suddivisa in due esercizi. Nel primo lo scopo era di rimuovere dalla point cloud tutti i punti con $X > 0$ e colorare i rimanenti in blu. Nel secondo era necessario filtrare la point cloud facendo di fatto un downsample dell'immagine 3D con risoluzioni diverse per ogni quadrante del piano X-Y.

1.1 Algortimi utilizzati

Per quanto riguarda il primo esercizio non c'è molto da dire sugli algoritmi utilizzati. Infatti, per modificare il colore dei punti, è sufficiente iterare su tutti i punti, verificare se $X > 0$ e in caso affermativo copiare il punto in una nuova point cloud modificando il suo colore in blu. La nuova cloud sarà quindi definita $C' = \{p \in C | p.x > 0, p.b = 255\}$. Per filtrare la point cloud invece è stata usata la classe di PCL `VoxelGrid` che divide lo spazio 3D in voxel: tutti i punti all'interno di un voxel vengono approssimati col loro centroide. Modificando la dimensione degli spigoli dei voxel si ottiene un donwsample maggiore o minore.

1.2 Considerazioni

L'implementazione è stata piuttosto diretta a partire dai file di esempio forniti, sono state necessarie solo alcune modifiche. L'output del secondo esercizio è mostrato in figura 1

2 Lab 2

I due esercizi della seconda esperienza puntavano a calcolare le normali ad una superficie definita da una point cloud e ad utilizzare queste normali per il calcolo di features che possano essere caratteristiche e univoche per ogni punto.

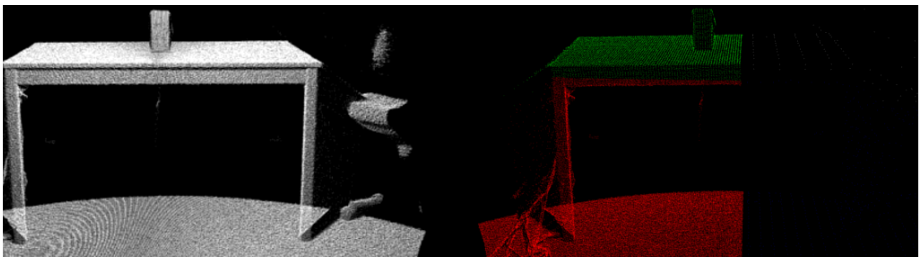

Figure 1: Esempio di point cloud filtrata e colorata: a sinistra l'originale, a destra l'output.

2.1 Algortimi utilizzati

Il calcolo delle normali è stato fatto attraverso la classe di PCL `NormalEstimationOMP`. `NormalEstimationOMP` per ogni punto p cerca i suoi vicini all'interno di una sfera di raggio r, questi vengono usati per stimare il piano π^* che meglio approssima la superficie nel punto p; cioè si calcola

$$\pi^* = \arg\min \sum_{x_i \in kNN(p)} dist(x_i, \pi) \tag{1}$$

`NormalEstimationOMP` utilizza OpenMP per fornire un'implementazione multithread multipiattaforma.
Vengono poi identificati i keypoint della point cloud, punti che per caratteristiche di posizione e/o colore rispetto ai loro vicini possono essere considerati salienti. Un buon keypoint detector dovrebbe essere invariante per traslazione, rotazione, scala e lumiosità, identificando sempre gli stessi keypoint indipendentemente dalle trasformazioni a cui viene sottoposta la point cloud. Per questo esercizio è stato scelto `SIFT3D`, un'estensione di `SIFT` in tre dimensioni, implementato dalla classe PCL `SIFTKeypoint`.
Una volta identificati i keypoint questi devono poter essere descritti o comparati in qualche modo. A questo scopo si utilizzano dei feature vector, cioè ad ogni keypoint viene assegnato un vettore caratteristico che identifica in modo univoco quel punto. Il contenuto dal vettore dipende dal feature descriptor utilizzato; per questo esercizio è stato usato `Fast Point Feature Histogram`. L'algoritmo considera il keypoint e tutti i suoi vicini entro un raggio r; per ogni coppia dell'insieme vengono calcolati i tre angoli che definiscono l'orientazione reciproca delle due normali; infine crea un istogramma a partire dalle orientazioni di tutte le coppie. Il feature vector così estratto è invariante per rotazione e traslazione della point cloud.

2.2 Considerazioni

Il calcolo delle normali e l'estrazione delle features è piuttosto semplice grazie alle classi di PCL. La parte più lunga, come spesso accade con questi algoritmi, è la taratura dei parametri migliori per ottenere il risultato desiderato. Per `SIFTKeypoint` sono stati usati `min_scale = 0.01`, `n_octaves = 3`, `n_scales_per_octave = 4`, `min_contrast = 0.001`. In figura 2 si possono vedere due esempi di descrittori calcolati con `FPFHEstimationOMP`.

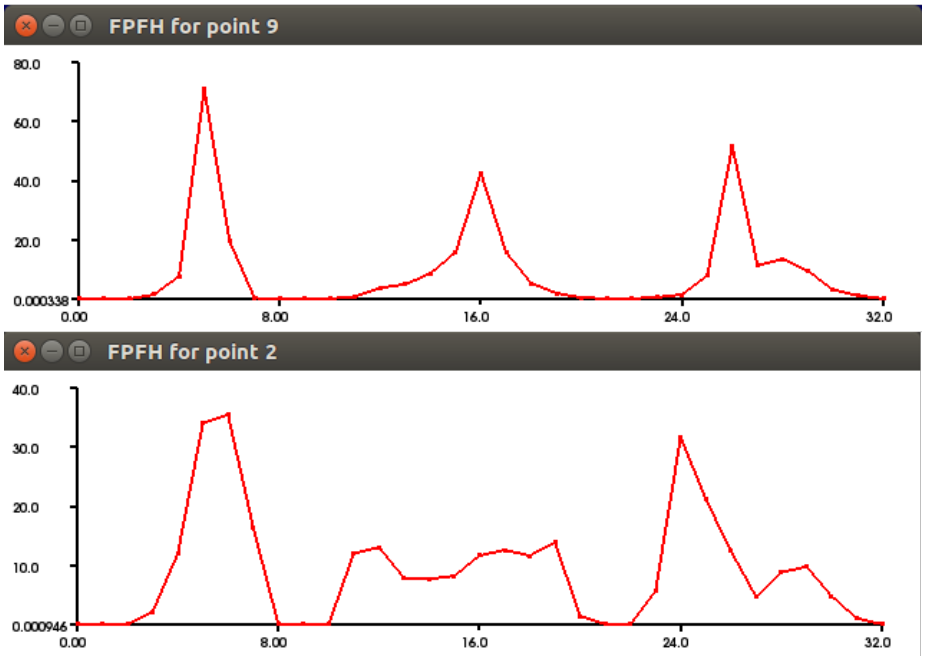

Figure 2: Due esempi di descrittori FPFH per due keypoint diversi.

3 Lab 3

Descrizione bla bla

3.1 Algortimi utilizzati (teoria)

3.2 Considerazioni

4 Lab 4

Descrizione bla bla

4.1 Algortimi utilizzati (teoria)

4.2 Considerazioni

References

[1] K. Grove-Rasmussen og Jesper Nygård, *Kvantefænomener i Nanosystemer*. Niels Bohr Institute & Nano-Science Center, Københavns Universitet