

**ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA**

---

**SCUOLA DI INGEGNERIA E ARCHITETTURA**

**DIPARTIMENTO DISI**

*CORSO DI LAUREA  
IN  
INGEGNERIA INFORMATICA*

**TESI DI LAUREA**

In

Calcolatori Elettronici T

**Metodologie di Machine Learning Applicate alla Visione  
Stereo**

CANDIDATO:  
Nigro Simone

RELATORE:  
Prof. Stefano Mattoccia

CORRELATORE:  
Dott. Matteo Poggi

Anno Accademico 2014/2015

**Sessione II**



# **Indice**

## **Indice**

**Elenco delle figure**

**Elenco delle tabelle**

**Elenco dei grafici**

**CAPITOLO 1 – Introduzione** 1

**CAPITOLO 2 - La Visone Stereo** 3

**2.1. Algoritmi di Matching Stereo** 4

**2.2. L'Algoritmo SGM** 6

**2.2.1. Calcolo dei Costi di Matching**

**2.2.2. Aggregazione dei Costi**

**2.2.3. Calcolo della Disparità**

**CAPITOLO 3 - Il Machine Learning**

**3.1. Classificazione dei Diversi Algoritmi**

**3.2. Framework Proposto**

**3.2.1. Alberi Decisionali**

**3.2.2. Random Forest**

**CAPITOLO 4 - Modifiche Proposte all'Algoritmo SGM**

**4.1. Random Forest**

**4.1.1. Parametri Random Forest**

**4.1.1.1. Ricerca dei Parametri Ottimali**

**4.1.2. Training**

**4.2. Modifiche Apportate all'Algoritmo SGM**

**CAPITOLO 5 – Risultati Sperimentali**

**5.1. Metodologie di Confronto**

**5.2. Risultati**

**5.2.1. Errore Percentuale Medio**

**5.2.2. Differenza Media dell'Errore Percentuale**

**5.2.3. Riduzione Percentuale dell'Errore**

**5.3. Confronto Mappe di Disparità**

**5.3.1. Scelta del Contributo Proveniente dalla Scanline Migliore**

**5.3.1.1.** Risultato Migliore

**5.3.1.2.** Risultato Peggio

**5.3.2.** Somma dei Contributi delle 4 Scanline più Affidabili

**5.3.2.1.** Risultato Migliore

**5.3.2.2.** Risultato Peggio

**5.3.3.** Somma Pesata dei Contributi delle 4 Scanline più  
Affidabili

**5.3.3.1.** Risultato Migliore

**5.3.3.2.** Risultato Peggio

**5.3.4.** Somma dei Contributi delle 8 Scanline

**5.3.4.1.** Risultato Migliore

**5.3.4.2.** Risultato Peggio

## **CAPITOLO 6 - Riflessioni Conclusive**

### **Bibliografia**

# **Elenco delle figure**

## **CAPITOLO 2 - La Visone Stereo**

- 2.1 Calcolo della profondità mediante triangolazione (sistema stereo ideale)
- 2.2 Un esempio di mappa di disparità
- 2.3 Algoritmo di matching basico
- 2.4 Strategia Winner-Takes-All
- 2.5 Approccio ad una singola scanline vs. approccio con 8 scanline
- 2.6 Costo globale del punto p
- 2.7 Matrice DSI

## **CAPITOLO 3 - Il Machine Learning**

- 3.1. Un esempio di albero decisionale

## **CAPITOLO 4 - Modifiche proposte all'algoritmo SGM**

- 4.1. Le 8 mappe di disparità ottenute
- 4.2. Confronto tra immagine di partenza, ground truth, e mappa di disparità densa

## **CAPITOLO 5 – Presentazione dei Risultati Sperimentali**

- 5.1. Una delle immagini della coppia 88
- 5.2. Mappa di disparità della coppia 88, algoritmo SGM
- 5.3. Mappa di disparità della coppia 88, selezione del contributo migliore
- 5.4. ground truth della coppia 88
- 5.5. Mappa di disparità della coppia 88, algoritmo SGM, nei punti validi
- 5.6. Mappa di disparità della coppia 88, selezione del contributo migliore, nei punti validi
- 5.7. Una delle due immagini della coppia 95
- 5.8. Mappa di disparità della coppia 95, algoritmo SGM
- 5.9. Mappa di disparità della coppia 95, selezione del contributo migliore
- 5.10. ground truth della coppia 95
- 5.11. Mappa di disparità della coppia 95, algoritmo SGM, nei

punti validi

- 5.12. Mappa di disparità della coppia 95, selezione del contributo migliore, nei punti validi
- 5.13. Una delle immagini della coppia 88
- 5.14. Mappa di disparità della coppia 88, algoritmo SGM
- 5.15. Mappa di disparità della coppia 88, somma 4 scanline
- 5.16. ground truth della coppia 88
- 5.17. Mappa di disparità della coppia 88, algoritmo SGM, nei punti validi
- 5.18. Mappa di disparità della coppia 88, somma 4 scanline, nei punti validi
- 5.19. Una delle due immagini della coppia 95
- 5.20. Mappa di disparità della coppia 95, algoritmo SGM
- 5.21. Mappa di disparità della coppia 95, somma 4 scanline
- 5.22. ground truth della coppia 95
- 5.23. Mappa di disparità della coppia 95, algoritmo SGM, nei punti validi
- 5.24. Mappa di disparità della coppia 95, somma 4 scanline, nei punti validi
- 5.25 Una delle due immagini della coppia 191
- 5.26. Mappa di disparità della coppia 191 algoritmo SGM
- 5.27. Mappa di disparità della coppia 191, somma pesata 4 scanline
- 5.28. ground truth della coppia 191
- 5.29. Mappa di disparità della coppia 191, algoritmo SGM, nei punti validi
- 5.30. Mappa di disparità della coppia 191, somma pesata 4 scanline
- 5.31. Una delle due immagini della coppia 24
- 5.32. Mappa di disparità della coppia 24, algoritmo SGM
- 5.33. Mappa di disparità della coppia 24, somma pesata 4 scanline
- 5.34. ground truth della coppia 24
- 5.35. Mappa di disparità della coppia 24, algoritmo SGM, nei punti validi
- 5.36. Mappa di disparità della coppia 24, somma pesata 4 scanline, nei punti validi
- 5.37. Una delle due immagini della coppia 105
- 5.38. Mappa di disparità della coppia 105 algoritmo SGM
- 5.39. Mappa di disparità della coppia 105, somma pesata 8 scanline
- 5.40. ground truth della coppia 105

- 5.41. Mappa di disparità della coppia 105, algoritmo SGM, nei punti validi
- 5.42. Mappa di disparità della coppia 105, somma pesata 8 scanline
- 5.43. Una delle due immagini della coppia 95
- 5.44. Mappa di disparità della coppia 95, algoritmo SGM
- 5.45. Mappa di disparità della coppia 95, somma pesata 8 scanline
- 5.46. ground truth della coppia 95
- 5.47. Mappa di disparità della coppia 95, algoritmo SGM, nei punti validi
- 5.48. Mappa di disparità della coppia 95, somma pesata 8 scanline, nei punti validi

# **Elenco delle tabelle**

## **CAPITOLO 5 –Risultati Sperimentali**

- 5.1. Errore percentuale medio commesso a soglie di tolleranza differenti
- 5.2. Differenza media tra l'errore commesso da SGM e quello commesso dalle varie modifiche proposte
- 5.3. Riduzione percentuale dell'errore media

# **Elenco dei grafici**

## **CAPITOLO 5 – Risultati Sperimentali**

- 5.1. Istogramma della riduzione percentuale dell'errore, scelta del contributo proveniente dalla scanline migliore
- 5.2. Istogramma della riduzione percentuale dell'errore, somma dei contributi delle 4
- 5.3. Istogramma della riduzione percentuale dell'errore, somma pesata dei contributi delle 4 scanline più affidabili
- 5.4. Istogramma della riduzione percentuale dell'errore, somma dei contributi delle 4 scanline più affidabili



## CAPITOLO 1. Introduzione

La visione stereo è un insieme di tecniche che, data una scena, mirano a stimare la profondità basandosi su due o più immagini acquisite da punti di vista differenti [1]. Tali tecniche permettono quindi di ottenere una rappresentazione tridimensionale dello spazio pur avendo a disposizione soltanto delle semplici immagini in due dimensioni.

I risultati ottenibili implementando gli algoritmi attualmente a disposizione non sono però esenti da errori o da imprecisioni, e di conseguenza si è sempre alla ricerca di metodi in grado di migliorarne le prestazioni.

In un mondo in cui l'intelligenza artificiale sta entrando sempre di più a far parte della vita quotidiana sotto forma di algoritmi per le ricerche web sempre più accurati, assistenti intelligenti presenti ormai in ogni smartphone [2], e sistemi di riconoscimento di volti e oggetti sempre più efficaci [3], viene spontaneo chiedersi se tale approccio può essere utilizzato anche nell'ambito della ricostruzione 3D.

Il Machine Learning in particolare è una branca dell'intelligenza artificiale, e si occupa dello sviluppo di algoritmi e tecniche che consentono ai computer di “imparare” e di utilizzare la conoscenza appresa per la sintesi di nuova conoscenza, sia essa sotto forma di una classificazione o della previsione di un risultato.

L'obiettivo di questo lavoro è proprio quello di implementare tecniche di Machine Learning al fine di provare a migliorare le prestazioni dei normali algoritmi di corrispondenza stereo.

In particolare, si cercherà di incrementare l'accuratezza dell'algoritmo di corrispondenza stereo SGM (Semi Global Matching)[4], un algoritmo stereo di tipo semi-globale in grado di

offrire un ottimo compromesso tra qualità dei risultati e velocità di esecuzione.

A partire dall'algoritmo SGM originale si cercherà di sfruttare le capacità di previsione di una Random Forest [5] al fine di ottenere mappe di disparità risultanti migliori rispetto a quelle ottenibili tramite l'algoritmo SGM standard.

## CAPITOLO 2. La Visione Stereo

La teoria della Visione Stereo è basata sull'utilizzo di due o più camere che inquadrano una stessa scena da differenti punti di vista. Nel caso ideale di *Fig.1.1* le due camere sono perfettamente allineate e poste alla stessa altezza, a una determinata distanza  $T$  tra di loro, e osservano entrambe lo stesso punto  $P$  rispettivamente dai centri di proiezione  $O_l$  e  $O_r$ . Per ciascuna di esse,  $P$  è proiettato sul rispettivo piano di proiezione nei punti  $p^l$  e  $p^r$ , aventi medesime coordinate verticali.

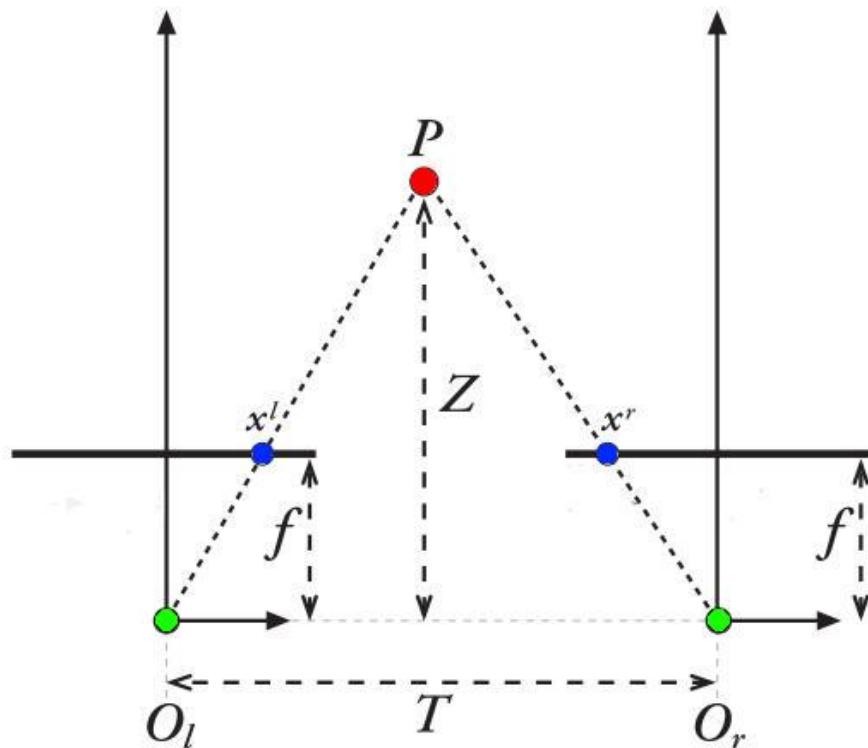


Fig.2.1 Calcolo della profondità mediante triangolazione (sistema stereo ideale)

Dal confronto della coppia di immagini ottenuta da un sistema in questa configurazione è possibile ottenere informazioni sulla profondità della scena stessa. Infatti, sfruttando la particolare geometria del sistema stereoscopico (geometria *epipolare*), è possibile valutare lo scostamento orizzontale tra i pixel omologhi delle due immagini. Chiamando  $x^l$  e  $x^r$  le coordinate orizzontali di  $p^l$  e  $p^r$  sui rispettivi piani, tale scostamento  $x^l - x^r$  (denominato *disparità*) permette di determinare, tramite una procedura di *triangolazione* e la conoscenza di opportuni parametri del sistema stereoscopico, la posizione del punto nello spazio: più i due punti sono lontani, minore è la disparità e viceversa.

L'esatta misura di profondità  $Z$  è calcolata con la formula

$$\frac{T-(x^l-x^r)}{z-f} = \frac{T}{z} \Rightarrow Z = \frac{fT}{(x^l-x^r)}$$

Dove  $T$  è la distanza tra i due centri di proiezione e  $f$  è la lunghezza focale delle due camere.

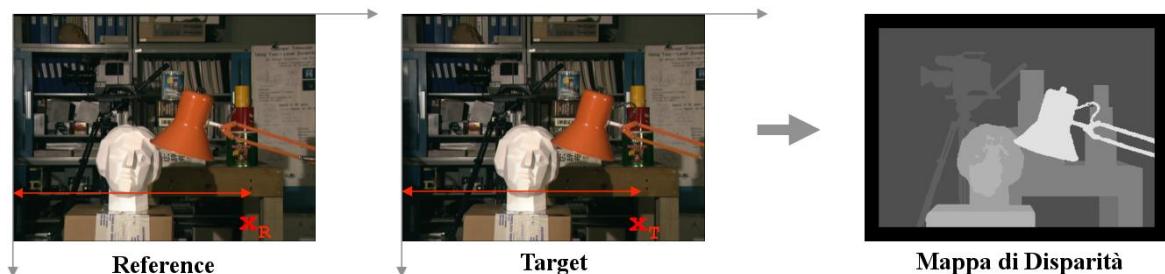


Fig. 2.2 Un esempio di mappa di disparità

Individuando tutte le coppie di pixel *omologhi* delle immagini (*matching stereo*) e iterando il calcolo della disparità per tutte le coppie di pixel ottenute, si ottiene la cosiddetta mappa di disparità,

ossia una immagine in scala di grigi dove l'intensità di ciascun pixel è proporzionale alla disparità in quel punto *Fig. 2.2*.

## 2.1. Algoritmi di Matching Stereo

Gli algoritmi di matching stereo si occupano di trovare le coppie di pixel omologhi (ovvero, la proiezione dello stesso punto reale nelle due immagini), denominate, rispettivamente, *Reference* e *Target* e di calcolare le disparità.

L'idea è quella di confrontare ciascun pixel dell'immagine Reference con i pixel dell'immagine Target sino a trovare il pixel corrispondente allo stesso punto della scena.

L'approccio più semplice è quello di confrontare l'intensità di ciascun pixel di Reference con le intensità dei pixel dell'immagine Target posti alla stessa altezza e di coordinate orizzontali  $x + d$ , dove  $d$  spazia da 0 a  $d_{max}$  e rappresenta la disparità cercata.

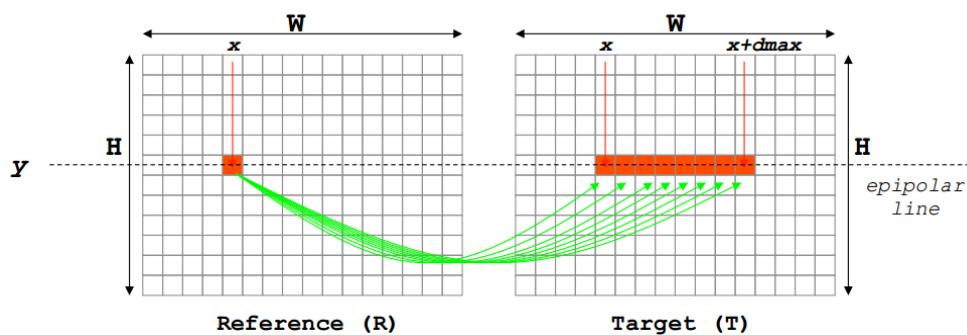


Fig. 2.3 Algoritmo di matching basico

A ciascuna coppia di pixel presa in esame viene assegnato un valore detto costo che indica quanto sono dissimili i due pixel in questione. Nel caso più semplice, il costo indica di quanto l'intensità del pixel di Reference si discosta dal pixel di Target preso in esame. Al termine dei confronti ciascun punto di Reference si troverà quindi associati

$d_{max}$  costi, uno per ciascun confronto effettuato con i pixel di reference posti da  $x$  a  $x+d_{max}$ .

La disparità in quel punto sarà il valore di  $d$  in cui il costo è, quindi, la differenza tra le intensità dei due pixel, è minore (*Strategia Winner-Takes-All*).

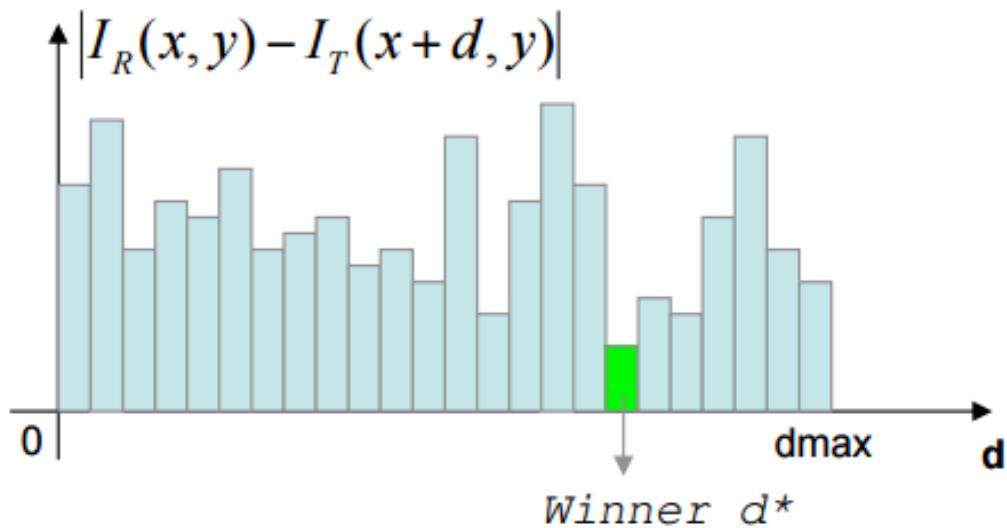


Fig. 2.4 Strategia *Winner-Takes-All*

Questo approccio non porta però a risultati particolarmente soddisfacenti.

Di base, esistono due strategie differenti per tentare di migliorare questo approccio: gli algoritmi locali e gli algoritmi globali.

Gli algoritmi locali usano la stessa strategia WTA ma riducono le ambiguità calcolando i costi su di una finestra che circonda il pixel di interesse invece che sul singolo pixel.

Gli algoritmi globali, invece, cercano la disparità corretta minimizzando una funzione di energia sull'intera immagine, calcolando quindi la disparità di ogni singolo punto in relazione a

tutti i restanti pixel dell'immagine.

Ciascun tipo di approccio ha i suoi pregi e i suoi difetti. I metodi locali risultano essere veloci ma soffrono molto di una forte ambiguità delle corrispondenze in aree caratterizzate da poca tessitura. I metodi globali, invece, producono risultati molto accurati ma al tempo stesso risultano essere piuttosto onerosi in termini di risorse computazionali e memoria, rendendoli così difficilmente implementabili su dispositivi con risorse hardware limitate.

Alcuni algoritmi cercano un compromesso fra la velocità degli approcci locali e la robustezza dei globali: in essi la minimizzazione della funzione energia è effettuata lungo un sottoinsieme delle immagini sfruttando metodi euristici. Per questo motivo sono in grado di produrre risultati più accurati rispetto agli algoritmi locali pur senza gli elevati costi computazionali dei globali, rendendoli idonei anche per applicazioni *real-time* e su piattaforme *embedded*.

## 1.2. L'Algoritmo SGM

Per i motivi elencati nel capitolo precedente, per il presente lavoro si è scelto di utilizzare l'algoritmo SGM. Come suggerisce il nome stesso, questo metodo semplifica l'approccio globale applicando una soluzione euristica.

Come tutti gli algoritmi stereo, l'azione di SGM può essere schematizzata fondamentali passaggi:

- **Calcolo dei costi di matching**

I costi possono essere calcolati seguendo criteri di confronto differenti che vanno dalla semplice differenza tra le intensità di due pixel omologhi a metodi ben più avanzati.

Nell'implementazione di SGM scelta è utilizzata la differenza di Hamming calcolata in ogni punto delle trasformate census della coppia di immagini.[1]

- **Aggregazione dei Costi**

Estrarre le disparità basandosi soltanto sui valori dei costi calcolati pixel per pixel, come ricordato in precedenza, può produrre risultati ambigui. Per questo motivo SGM effettua un'aggregazione dei costi provenienti da più punti dell'immagine e introduce una funzione di energia.

- **Calcolo della disparità**

Al fine di calcolare il valore della disparità in ogni punto, sarà minimizzata la funzione di energia introdotta al punto precedente.

### 2.2.1 Calcolo dei Costi di Matching

Il primo passaggio dell'implementazione dell'algoritmo SGM che utilizzeremo consiste nell'applicare alle immagini la trasformata census [6], un'operazione che ci permette di ottenere, per ciascun pixel, una stringa di bit.

La trasformata census si basa sul seguente operatore di confronto:

$$\xi(I, p, p') = \begin{cases} 1 & se I(p) < I(p') \\ 0 & altrimenti \end{cases}$$

dove  $I(p)$  e  $I(p')$  sono, rispettivamente, i valori dell'intensità dei pixel

$p$  e  $p'$ .

La trasformata Census per un pixel  $p$  nell'immagine  $I$  è la stringa di bit:

$$C[I(p)] = \bigodot_{p' \in S(p, \beta)} \xi(I, p, p')$$

dove  $S(p, \beta)$  rappresenta una finestra chiamata finestra di trasformazione, di raggio  $\beta$  centrata in  $p$ .

Ciò che fa la trasformata Census, è circondare ciascun pixel con una finestra di dimensione fissa, nel nostro caso 9x9, e comporre una stringa di bit avente tanti bit quanti sono i pixel della finestra meno il pixel preso in esame posto al centro. A ciascun bit è assegnato il valore 0 se l'intensità del corrispondente pixel della finestra è maggiore dell'intensità del pixel al centro, 1 altrimenti.

Una volta applicata la trasformata ad entrambe le immagini, per calcolare il costo di matching tra due punti verrà utilizzata la distanza di Hamming, una semplice operazione tra stringhe di bit che restituisce il numero di bit nei quali esse differiscono.

Rispetto al semplice confronto tra le intensità dei pixel utilizzato nel caso più semplice, questo criterio di somiglianza è preferibile in quanto è:

- Invariante nei confronti di forti distorsioni fotometriche che conservano *l'ordine reciproco* tra le intensità.
- Robusto (tollerante nei confronti degli errori dovuti a occlusioni).
- Veloce (le operazioni sono semplici confronti di numeri interi).

## 2.2.2 Aggregazione dei Costi

La disparità non è più calcolata affidandosi soltanto alla differenza tra un pixel ed il presunto pixel omologo, ma è introdotta una funzione di energia più complessa che verrà minimizzata alla ricerca della disparità d migliore:

$$E(d) = E_{data}(d) + E_{smooth}(d)$$

Dove il primo termine indica la funzione dei costi di matching da minimizzare lungo l'intera immagine, ed il secondo termine serve a penalizzare in modo più o meno marcato eventuali variazioni di disparità tra punti vicini.

Più precisamente, nel caso di SGM, la funzione dei costi globale si può esprimere come:

$$E(D) = \sum_p (C(p, D_p) + \sum_{q \in N_p} P_1 T[|D_p - D_q| = 1] + \sum_{q \in N_p} P_2 T[|D_p - D_q| > 1]). \quad ()$$

Questa funzione è calcolata lungo una linea in una particolare direzione (*scanline*, da cui prende il nome questa euristica, *scanline optimization*).

Il primo termine indica la somma dei costi di matching per le disparità dell'intera immagine. Il secondo termine aggiunge una penalità costante  $P_1$  per tutti i pixel  $q$  appartenenti al vicinato di  $N_p$  di  $p$  per i quali la disparità varia di poco (in questo caso di 1 pixel). Il terzo termine aggiunge una penalità maggiore  $P_2$  per tutte le variazioni di disparità maggiori di 1 pixel.

Lo scopo di  $P_1$  e  $P_2$  (detti anche *smoothness penalties*) è quello di penalizzare i cambi di disparità tra punti vicini, garantendo la continuità (smoothness) dell'immagine e permettendo quindi all'algoritmo di adattarsi alle superfici curve o inclinate grazie a  $P_1$

(negli approcci locali, i pixel appartenenti a superfici inclinate spesso sono assegnati alla stessa ipotesi di disparità, pur essendo a distanze diverse), ma anche di preservare eventuali discontinuità presenti nella scena grazie alla penalità maggiore  $P_2$ . Un eventuale cambio di disparità molto accentuato, infatti, verrebbe considerato valido nel caso in cui il costo in quel punto sia talmente basso da essere il minimo nonostante la penalità introdotta. [4]

Un'altra differenza fondamentale tra SGM e il caso base analizzato precedentemente è che la ricerca dei pixel omologhi non è effettuata soltanto lungo una direzione ma lungo ben 8 (o 16) differenti direzioni denominate, appunto, scanline.

La funzione dei costi appena introdotta è quindi applicata lungo ciascuna delle 8 o 16 direzioni differenti. Nel nostro caso 8.



Fig. 2.5 Approccio a una singola scanline vs. approccio con 8 scanline

Per ogni direzione  $\mathbf{r}$ , se chiamo  $L'_{\mathbf{r}}(\mathbf{p}, d)$  il costo globale,  $C(\mathbf{p}, d)$  il costo puntuale relativo al punto  $\mathbf{p}$  e alla disparità  $d$  in esame, e per  $i$  si intendono tutti i valori di  $d$  da  $d_{\min}$  a  $d-1$  e da  $d+1$  a  $d_{\max}$ , si definisce ricorsivamente:

$$\begin{aligned} L'_{\mathbf{r}}(\mathbf{p}, d) = & C(\mathbf{p}, d) + \min(L'_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d), \\ & L'_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d - 1) + P_1, \\ & L'_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d + 1) + P_1, \\ & \min_i L'_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + P_2). \end{aligned}$$

Dove per  $\mathbf{p}-\mathbf{r}$  si intende il punto precedente, che a sua volta riporta le informazioni di tutti i punti percorsi nel cammino  $\mathbf{r}$  fino al punto  $\mathbf{p}$ . Siccome il valore  $L'_{\mathbf{r}}(\mathbf{p}, d)$  incrementa costantemente durante l'avanzamento della scansione in una direzione, occorre sottrarre alla formula appena introdotta il minimo costo globale calcolato nel punto precedente  $\mathbf{p} - \mathbf{r}$ .

$$\begin{aligned} L_{\mathbf{r}}(\mathbf{p}, d) = & C(\mathbf{p}, d) + \min(L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d), \\ & L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d - 1) + P_1, \\ & L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d + 1) + P_1, \\ & \min_i L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + P_2) - \min_k L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, k) \end{aligned}$$

Si noti che, sottraendo uno stesso valore a tutti i fattori, non è modificata la posizione del minimo e che, di conseguenza, la disparità non subisce variazioni.

Graficamente, immaginando di avere 8 possibili disparità, il costo globale nel punto  $p$  alla disparità 4 può essere rappresentato come:

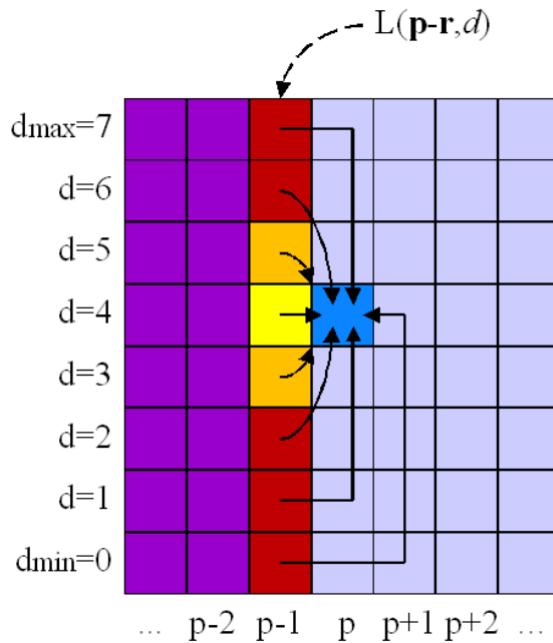


Fig. 2.6 Costo globale del punto  $p$

A titolo di esempio, supponiamo di conoscere il costo globale relativo alla disparità selezionata nel punto  $p-1$ , pari a  $L'(\mathbf{p}-1, k)_{\min}$ , oltre al costo puntuale del punto  $\mathbf{p}$  alla disparità  $d=4$   $C(\mathbf{p}, 4)$ . Posso allora calcolare il relativo contributo globale  $L'(p, 4)$ :

$$L'(p, 4) = C(p, 4) + \min \left\{ \begin{array}{l} L'(p - 1,4) \\ L'(p - 1,5 + P_1) \\ L'(p - 1,3 + P_1) \\ L'(p - 1,7 + P_2) \\ L'(p - 1,6 + P_2) \\ L'(p - 1,2 + P_2) \\ L'(p - 1,1 + P_2) \\ L'(p - 1,0 + P_2) \end{array} \right\} - L'(p - 1, k)_{\min}$$

Una volta valutati i contributi  $L'(p, d)$  relativi ai pixel dell'intera immagine per ogni valore di  $d$ , occorrerà considerare i costi globali ottenuti da ogni direzione.

In SGM questo è ottenuto mediante la seguente sommatoria  $S$ :

$$S(\mathbf{p}, d) = \sum_{\mathbf{r}} L_{\mathbf{r}}(\mathbf{p}, d)$$

Dove  $\mathbf{r}$  può assumere il valore 8 o 16 a seconda che si sia deciso di utilizzare 8 o 16 differenti scanline.

I valori  $S(\mathbf{p}, d)$  vengono quindi memorizzati all'interno di una matrice tridimensionale denominata **DSI** (**D**isparity **S**pace **I**mage).

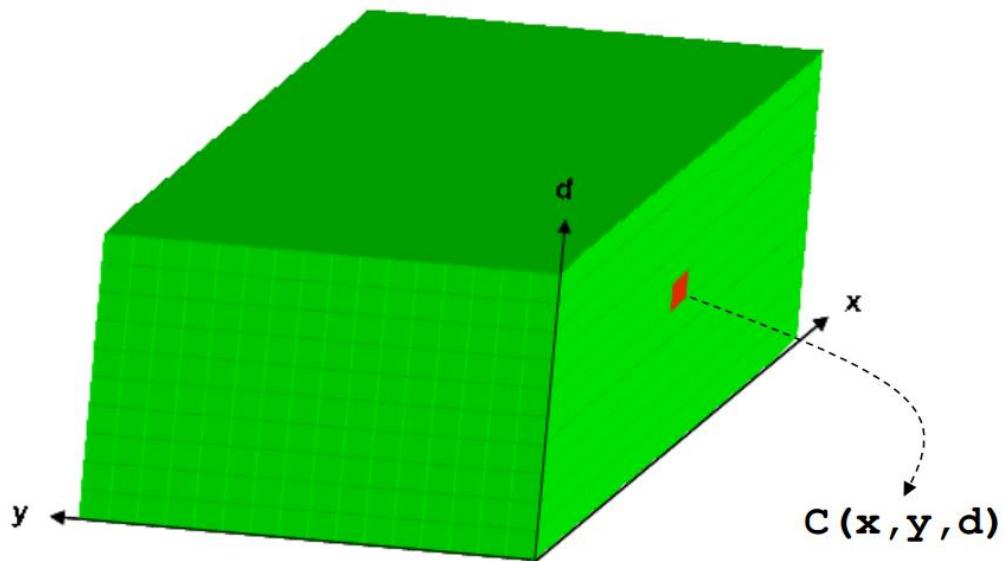


Fig.2.7 Matrice DSI. Ciascun elemento  $C(x,y,d)$  del **DSI** rappresenta il costo della corrispondenza tra il pixel in posizione  $(x,y)$  dell'immagine Reference ed il pixel in posizione  $(x,y)$  dell'immagine Target alla disparità  $d$ .

### 2.2.3 Calcolo della Disparità

Una volta raggruppati all'interno della **DSI** tutti i costi aggregati ottenuti, si procede cercando il costo dal valore minimo per ogni pixel.

Una volta individuato il minimo per ciascun punto  $(x,y)$ , al valore della disparità nel punto  $(x,y)$  della mappa di disparità risultante verrà assegnato l'indice corrispondente al costo minimo.

## CAPITOLO 3. Il Machine Learning

Come già accennato in precedenza, il Machine Learning si occupa di algoritmi e sistemi in grado di “apprendere”. Dove “apprendere” può significare imparare a svolgere un determinato compito, a prevedere dei risultati, o a classificare dei dati. In generale, un sistema basato sul Machine Learning è in grado di apprendere come migliorare il proprio comportamento basandosi sull'esperienza passata.

Spesso si ha la necessità di risolvere problemi che risulterebbero di difficile soluzione seguendo un approccio algoritmico tradizionale, pur essendo talvolta i medesimi compiti banali se svolti da un essere umano. Riconoscere un volto all'interno di una serie di immagini, o individuare un ostacolo durante la guida, sono due cose che facciamo senza il minimo sforzo, ma si tratta di due azioni non facilmente realizzabili mediante un algoritmo tradizionale che esamina determinate caratteristiche delle immagini o delle altre strutture dati inviate ad un calcolatore.

È qui che entra in gioco il Machine Learning. Piuttosto che studiare un algoritmo in grado di risolvere il problema direttamente, l'idea alla base del Machine Learning è quella di far sì che sia il computer a trovare una soluzione al problema, senza la necessità di codificare un algoritmo modellato attorno al problema specifico.

Il Machine Learning può essere visto come un tipo di programmazione “per esempi”, dal momento che l'unica cosa sulla quale si basa il computer per riuscire a fornire una soluzione al problema è un set più o meno ampio di esempi con relativa soluzione ottimale.

### **3.1. Classificazione dei Diversi Algoritmi**

Esistono differenti tipi di approcci al Machine Learning, una prima distinzione tra i diversi algoritmi si può effettuare in base al meccanismo utilizzato per l'apprendimento:

- **Apprendimento Supervisionato**

Gli algoritmi di apprendimento supervisionato sono addestrati attraverso un set di esempi caratterizzati da più attributi (*features*) e a cui è associato un certo risultato (o un'*etichetta/label*), tale set di esempi prende il nome di *training set*.

L'algoritmo cerca quindi un modello che colleghi l'andamento degli attributi a quel determinato risultato o etichetta (*fase di training*).

Dopo che l'algoritmo ha trovato il modello migliore possibile, questo è utilizzato per eseguire stime per i dati di prova privi di un risultato. (*fase di testing*)

- **Apprendimento Non Supervisionato**

L'algoritmo cerca apprendere a partire da un set di esempi privi di risultati o etichette, non ha quindi alcuna indicazione su cosa cercare. Ciò fa sì che il modello stesso riclassifichi e organizzi gli input in base a caratteristiche comuni individuate autonomamente nei dati, effettuando ragionamenti e previsioni su input successivi. Gli algoritmi non supervisionati procedono quindi confrontando i dati e cercando analogie e differenze.

- **Apprendimento Per Rinforzo**

L'apprendimento per rinforzo punta a realizzare algoritmi in grado di apprendere e adattarsi alle mutazioni dell'ambiente. Questi tipi di algoritmi si basano sul presupposto di potere ricevere degli stimoli dall'esterno a seconda delle scelte fatte. Quindi una scelta corretta comporterà un premio mentre una scelta scorretta porterà ad una penalizzazione del sistema. L'obiettivo del sistema è il raggiungimento del maggior premio possibile e di conseguenza del migliore risultato possibile.

### **3.2. Framework proposto**

Il Machine Learning può aiutarci a risolvere tutta una serie di problemi differenti, ma di base tali algoritmi possono essere impiegati o al fine di classificare dei dati ricevuti in ingresso (ad esempio, ricevute delle immagini, il classificatore è in grado di suddividerle in base al contenuto), o al fine di produrre un risultato numerico (regressione).

Per quanto concerne il presente lavoro, il Machine Learning è impiegato al fine di ottenere un risultato numerico compreso tra 0 e 1 che indichi il grado di correttezza stimato per i diversi contributi provenienti dalle varie scanline. Abbiamo quindi bisogno di un *regressore*, ovvero di un modello in grado di restituirci un valore numerico e non un'etichetta “giusto” o “sbagliato” come potrebbe fare un *classificatore*.

La scelta è ricaduta su di un regressore di tipo Random Forest, un particolare classificatore ottenuto dalla composizione di più

classificatori semplici detti *alberi decisionali*.

### 3.2.1. Alberi Decisionali

Un albero decisionale [7] è un *modello predittivo* che può essere immaginato come una struttura gerarchica formata da un insieme di nodi correlati da archi (rami) orientati e "etichettati".

In un albero decisionale ogni nodo effettua un test su un attributo, ogni ramo uscente da un nodo corrisponde ad uno dei possibili valori che l'attributo può assumere, ed ogni foglia rappresenta il risultato predetto a partire dai valori dei vari attributi. Ciascuna regola di predizione è rappresentata dal cammino (path) dal nodo radice (root) al nodo foglia.

In fase di costruzione dell'albero, nel caso della classificazione, per ciascun nodo sarà scelto come attributo da valutare quello che in quel punto ha la capacità di discriminare maggiormente i dati ricevuti in ingresso. Nel caso della regressione, invece, a ogni nodo è provato uno split in base a ciascun singolo attributo, poi è scelto l'attributo che ha prodotto il risultato che si discosta meno dal risultato reale, e così via in maniera ricorsiva.

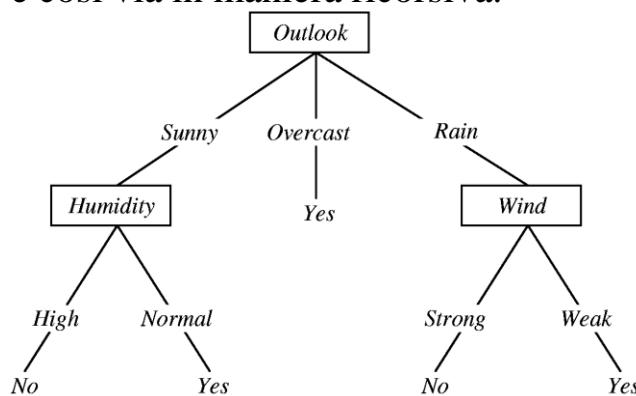


Fig. 3.1. Un esempio di albero decisionale in grado di stimare se una particolare giornata è adatta per giocare a tennis in base a diverse condizioni meteorologiche

### **3.2.2. Random Forest**

Le Random Forest rientrano nella sfera dell'*ensemble learning*, dal momento che altro non sono se non un insieme di alberi decisionali. L'idea alla base dell'*ensemble learning* è quella di ottenere un classificatore/regressore *forte* (ovvero molto efficace) combinando assieme più classificatori/regressori *deboli* (ovvero non necessariamente molto efficaci).

A differenza dei normali alberi decisionali, ciascun albero presente all'interno di una Random Forest è costruito a partire da un sottoinsieme casuale dei dati presenti nel training set, gli alberi non utilizzano quindi il set completo, e ad ogni nodo non viene più selezionato l'attributo migliore in assoluto, ma viene scelto l'attributo migliore tra un set di attributi selezionati casualmente. Un tale approccio, che all'apparenza potrebbe sembrare controproducente, produce in realtà risultati spesso migliori rispetto ad altri tipi di classificatori/regressori.

Il risultato finale restituito dalla Random Forest altro non è che la media del risultato numerico restituito dai diversi alberi nel caso di una regressione, o la classe restituita dal maggior numero di alberi nel caso la Random Forest sia stata utilizzata per effettuare una classificazione.

## CAPITOLO 4. Modifiche Proposte ad SGM

Lo scopo della presente tesi è quello di sfruttare le potenzialità di una Random Forest per migliorare la modulazione dei costi effettuata dall'algoritmo SGM e tentare quindi di ottenere un'accuratezza maggiore dei risultati.

Tutte le immagini utilizzate provengono dal dataset KITTI [8], scelto perché si tratta di un dataset creato a partire da scenari reali (contiene quindi scene particolarmente difficili da analizzare), e non da scenari creati ad hoc in laboratorio, che fornisce per ciascuna coppia di immagini la corrispondente *ground truth*, ovvero una mappa di disparità in cui le profondità sono sicuramente corrette, dal momento che sono state rilevate tramite scansioni laser [8].

### 4.1. Random Forest

Il punto cardine attorno al quale ruota l'intero lavoro è la Random Forest. Abbiamo deciso di utilizzare il componente CvRTrees fornito da OpenCV, una libreria *opensource* incentrata sulla Computer Vision e dotata di moduli per il Machine Learning. [9]

#### 4.1.1. Parametri Random Forest

Il componente CvRTrees che abbiamo deciso di utilizzare come implementazione di una Random Forest permette di settare tutti i diversi parametri che possono caratterizzare una foresta. Andiamo a vederli nel dettaglio:

- **max\_depth**

La profondità massima degli alberi decisionali contenuti al suo interno. Un valore troppo basso rischia di far sì che la foresta non riesca a rispecchiare correttamente il trend dei dati (underfitting), ma al contempo un valore troppo alto potrebbe rendere il modello troppo specifico e troppo sensibile al rumore, e ciò andrebbe a minare la qualità delle predizioni della foresta (overfitting)

- **min\_sample\_count**

Il numero minimo di campioni necessari affinché un nodo sia ramificato

- **max\_categories**

E' un parametro che è utilizzato per raggruppare più categorie in una durante la ricerca dell'attributo su cui eseguire lo split nei casi di classificazione in cui il numero di categorie è maggiore di 2. Nel nostro caso non è considerato, dal momento che ci occupiamo di regressione.

- **calc\_var\_importance**

Se impostato a true, la foresta calcolerà l'importanza attribuita a ciascuna feature durante la fase di creazione del modello

- **nactive\_vars**

Permette di specificare il numero di attributi selezionati casualmente all'altezza di ciascun nodo tra cui calcolare lo split migliore.

- **max\_num\_of\_trees\_in\_the\_forest**

Il massimo numero di alberi che possono essere presenti nella Foresta. Tipicamente, maggiore è il numero di alberi e maggiore sarà l'accuratezza delle previsioni. Tuttavia, una volta raggiunto un certo numero di alberi, il miglioramento nella qualità delle previsioni tenderà a calare, e la qualità si assesterà ad un valore limite. Una cosa importante di cui tenere conto è che i tempi necessari per costruire il modello e per eseguire le previsioni aumenteranno in maniera direttamente proporzionale al numero di alberi.

- **forest\_accuracy**

Permette di specificare il livello di accuratezza desiderato (errore OOB).

- **termcrit\_type**

Permette di specificare il criterio da valutare per decidere quanto interrompere la creazione del modello, può assumere i seguenti valori:

- **CV\_TERMCRIT\_ITER**

Termina quando è stato raggiunto il numero massimo di alberi

- **CV\_TERMCRIT\_EPS**

Termina quando è stata raggiunta l'accuratezza desiderata

- **CV\_TERMCRIT\_ITER | CV\_TERMCRIT\_EPS**

Usa entrambi i criteri

### **4.1.1. Training**

La prima, fondamentale, operazione da compiere quando si lavora con un regressore/classificatore è il training.

E' molto importante scegliere con attenzione degli attributi significativi come features da fornire alla Random Forest.

In questo caso si è deciso di utilizzare come features la *mediana*, la *deviazione standard*, e la *deviazione mediana assoluta* calcolate su finestre di dimensioni 5x5, 7x7, 9x9, e 11x11 su ciascuna delle 8 mappe di disparità ottenibili prendendo in esame i contributi provenienti da una singola scanline per volta.

Per evitare di creare una Random Forest in grado di lavorare al meglio soltanto in situazioni specifiche, le immagini utilizzate per il training sono state scelte in modo tale da rappresentare tutte le differenti situazioni in cui gli algoritmi stereo incontrano delle difficoltà: aree senza textures, baseline molto ampie, pattern ripetitivi, trasparenze, e riflessi. Le coppie di immagini scelte sono la 43, 71, 82, 87, 94, 120, 122, 180 del traing set di KITTI [11].

Sfruttando l'algoritmo SGM, per ciascuna coppia sono quindi state generate le 8 mappe di disparità delle singole scanline visibili nella *Fig.4.1.*

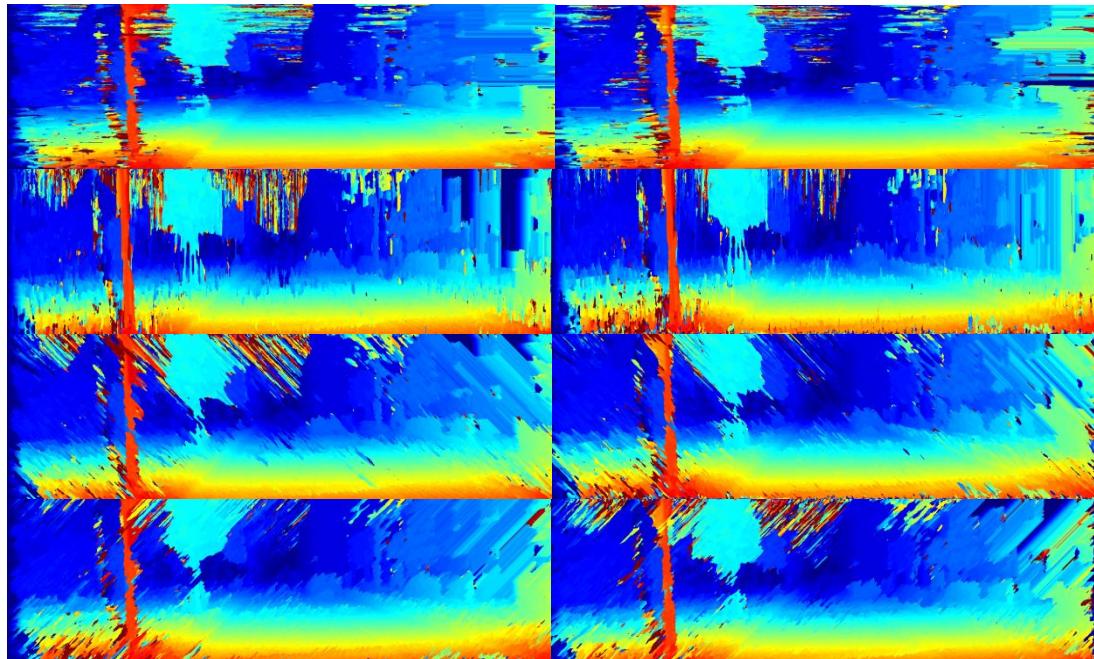


Fig.4.1. Le 8 mappe di disparità ottenute

Una volta ottenute le 8 mappe, in ciascun punto di esse sono state calcolate mediana, deviazione standard, e deviazione mediana assoluta su finestre di dimensioni 5x5, 7x7, 9x9, e 11x11.

A questo punto è stato generato un file di training in formato csv composto da tanti record quanti sono i pixel validi delle 64 mappe ottenute.

Ogni record è composto dalla sequenza dei valori delle 12 features in quel punto e da un risultato numerico che può assumere i valori 0 o 1.

Il risultato attribuito a ciascun record è stato ottenuto confrontando l'intensità del pixel in questione con quella del corrispondente pixel della ground truth qualora tale intensità della ground truth in quel punto fosse diversa da 0 (a ciò corrisponderebbe un punto valido della ground truth).

Le ground truth presenti nel dataset KITTI non contengono infatti informazioni dense sulla profondità, ma sono definite per punti. Soltanto il 50% dei pixel delle ground truth è definito, gli altri hanno intensità 0 [8].

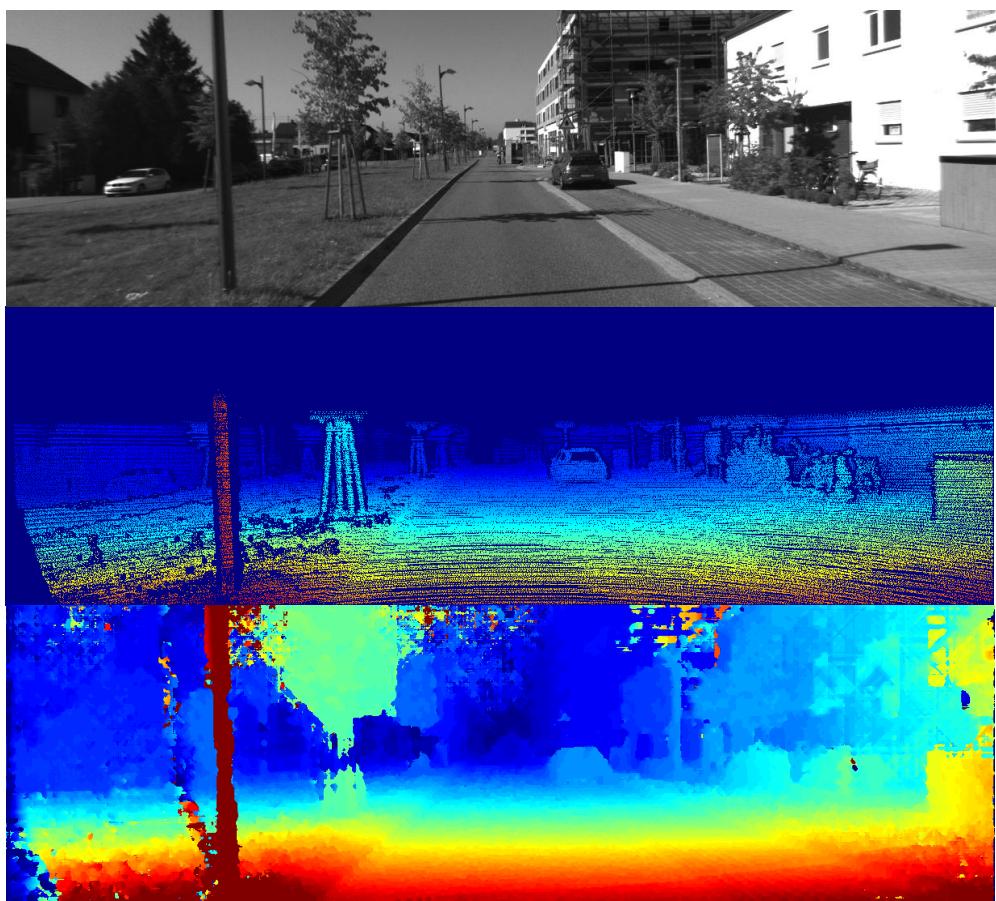


Fig.4.2. Confronto tra immagine di partenza, ground truth, e mappa di disparità densa

Se l'intensità del pixel analizzato si discosta di 3 o meno da quella del corrispondente pixel della ground truth, il risultato assegnato al record sarà 1, 0 altrimenti. Tale soglia è necessaria in virtù dei possibili errori di calibrazione o di misura degli strumenti laser utilizzati nella creazione del dataset KITTI [8].

#### **4.1.1.1. Ricerca dei Parametri Ottimali**

Al fine di individuare i parametri ottimali, sono state effettuate svariate prove a partire dallo stesso file di training ed utilizzando un file di test generato allo stesso modo del file di training a partire dalle immagini 10-20 del training set di KITTI.

I risultati migliori sono stati ottenuti con:

<b>max_depth</b>	25,
<b>min_sample_count</b>	1000,
<b>regression_accuracy</b>	0.0001f,
<b>compute_surrogate_split</b>	false,
<b>nactive_vars</b>	12,
<b>max_number_of_trees_in_the_forest</b>	10,
<b>forrest_accuracy</b>	0.0001f,
<b>termcrit_type</b>	CV_TERMCRIT_ITER  CV_TERMCRIT_EPS.

Come si può notare, i risultati migliori sono stati ottenuti facendo in modo che ad ogni split l'attributo migliore fosse scelto tra tutti i 12 attributi presenti. Si è inoltre deciso di utilizzare un numero limitato di alberi, dal momento che un numero maggiore avrebbe incrementato considerevolmente i tempi di elaborazione senza che la qualità dei risultati ne traesse particolare beneficio [10].

#### **4.2. Modifiche Apportate all'Algoritmo SGM**

La prima parte di SGM resterà immutata. Le differenze tra pixel

saranno quindi calcolate tramite la distanza di Hamming applicata alle trasformate Census delle coppie di immagini Reference e Target come accade nell'implementazione di SGM scelta come base di partenza [12][13]. Inoltre, anche i costi lungo le singole scanline saranno trovati nella maniera originaria.

Ciò che cambia è il modo in cui saranno aggregati i costi calcolati lungo le 8 differenti scanline.

Anziché sommarli e memorizzare i risultati all'interno di una DSI come eseguito dall'approccio tradizionale, infatti, sono stati studiati quattro nuovi approcci che potessero sfruttare le informazioni ottenute dal *framework di learning* introdotto precedentemente.

In tutti e quattro i casi l'algoritmo creato riceve in ingresso una coppia di immagini ed utilizza il classico algoritmo SGM per calcolare i costi in ciascun punto lungo le 8 direzioni previste.

A questo punto, anziché sommare gli 8 contributi provenienti dalle diverse scanline, analogamente a quanto fatto per generare il file di train, utilizza i contributi di ciascuna per generare 8 mappe di disparità diverse, ognuna delle quali ottenuta a partire dai contributi di una singola scanline, per poi calcolare mediana, deviazione standard, e deviazione mediana assoluta su finestre di dimensioni 5x5, 7x7, 9x9, e 11x11 per ciascuna di esse.

A ogni punto delle 8 mappe sono quindi associati i valori delle 12 features calcolate in quel punto.

I valori così ottenuti sono passati come vettore di features alla Random Forest che, a partire da essi, effettuerà una predizione e genererà quindi un risultato numerico compreso tra 0 e 1 per ciascun punto delle 8 mappe.

Per come è stata addestrata, è facile intuire che la foresta assocerà valori bassi ai punti che crede essere sbagliati, e valori prossimi all'1 per i punti che reputa corretti.

A questo punto le strade si dividono, e ciascun metodo utilizzerà i risultati forniti dalla Random Forest in modi differenti.

Ecco le 4 proposte di modifiche sperimentate:

- **Scelta del contributo proveniente dalla scanline migliore**

Per ogni pixel, è scelta l'ipotesi di disparità corrispondente al minimo costo estratto dalla scanline più affidabile, cioè avente il valore in uscita dalla Random Forest più alto.

- **Somma dei contributi delle 4 scanline più affidabili**

Il DSI è creato memorizzando in ogni suo punto il risultato della somma dei contributi provenienti dalle 4 scanline ritenute più affidabili, cioè aventi lo score più alto formulato dalla Random Forest.

- **Somma pesata dei contributi delle 4 scanline più affidabili**

Il DSI è creato memorizzando in ogni suo punto il risultato della somma pesata dei contributi provenienti dalle 4 scanline ritenute più affidabili. Il peso assegnato a ciascuna di esse altro non è che il risultato restituito dalla Random Forest relativo all'affidabilità della scanline in questione nel punto analizzato.

- **Somma pesata dei contributi delle 8 scanline**

Il DSI è creato memorizzando in ogni suo punto il risultato della somma pesata dei contributi provenienti dalla totalità delle

scanline. Come per la proposta precedente, il peso assegnato a ciascuna di esse altro non è che il risultato restituito dalla Random Forest relativo all'affidabilità della scanline in questione nel punto analizzato.

## CAPITOLO 5. Presentazione Dei Risultati Ottenuti

Per valutare l'efficacia dei diversi interventi effettuati si è deciso di generare con SGM e con ciascuno dei 4 metodi proposti le mappe di disparità per tutte le coppie di immagini presenti nel training set KITTI. Avere utilizzato il training set KITTI significa avere a disposizione le mappe di disparità ground truth per ogni coppia di immagini. Andremo quindi a confrontare le mappe di disparità ottenute con le rispettive ground truth al fine di calcolare la percentuale di pixel errati. Ciò ci permetterà di confrontare i risultati ottenuti con quelli ottenibili utilizzando l'algoritmo SGM.

### 5.1 Metodologie di Confronto

Per ovvi motivi, sono stati analizzati soltanto i punti in cui la ground truth è definita, e si ricorda che essi ammontano a circa il 50% del totale [8].

L'intensità di ciascun pixel delle mappe di disparità ottenute è stata confrontata con l'intensità dell'omologo pixel della ground truth che rappresenta il valore di disparità corretto e sono considerati errati i punti con una differenza superiore a una certa soglia.

I confronti sono stati eseguiti scegliendo come valori di soglia 0, 1, 2, 3, e 4.

Si ricorda che, in virtù dei possibili errori di calibrazione o di misura degli strumenti laser utilizzati nella creazione del dataset KITTI, i test effettuati a partire dalle immagini contenute in esso solitamente utilizzano una soglia pari a 3.

## 5.2 Risultati

### 5.2.1. Errore percentuale medio

Soglia	SGM Standard	Punti Scelti	Somma 4	Somma 4 Pesata	Somma 8 Pesata
4	6,1529	7,882	7,8884	6,8406	5,7071
3	8,0859	11,054	11,054	9,2205	7,5977
2	12,738	17,856	17,856	14,662	12,162
1	29,792	34,695	34,697	31,445	28,999
0	74,953	74,239	74,238	73,984	74,279

Tab.5.1. Errore percentuale medio commesso a soglie di tolleranza differenti (0,1,2,3 e 4)

Come si evince dalla *Tab.5.1*, tre dei quattro metodi sperimentati non sono riuscite a migliorare l'accuratezza dell'algoritmo SGM se non nel caso dei confronti effettuati con soglia di tolleranza pari a 0, una soglia poco significativa alla luce del fatto che l'accuratezza delle ground truth presenti nel dataset KITTI non è garantita a soglie così ridotte.

Tre delle quattro metodologie non sono quindi state in grado di ridurre l'errore sull'output di SGM dopo averne modificato la formulazione.

Notiamo però un importante risultato: introducendo dei pesi nella somma dei contributi provenienti dalle 8 scanline siamo riusciti ad ottenere un errore significativamente inferiore, considerando il numero ridotto di errori commessi da SGM standard, e per ogni soglia di tolleranza rispetto a quello commesso dall'implementazione di SGM originale.

### **5.2.2. Differenza media dell'errore percentuale rispetto a SGM**

Soglia	Punti Scelti	Somma 4	Somma 4 Pesata	Somma 8 Pesata
4	-1,73531	-1,73553	-0,688	0,445811
3	-2,9683	-2,96823	-1,13458	0,488201
2	-5,11764	-5,11802	-1,92405	0,576039
1	-4,90313	-4,9051	-1,65286	0,793031
0	0,713963	0,714524	0,969036	0,679017

Tab.5.2. Differenza media tra l'errore commesso da SGM e quello commesso dalle varie modifiche proposte. Un valore positivo indica un miglioramento.

La Tab.5.2 evidenzia la differenza, in media, tra l'errore ottenuto utilizzando SGM e l'errore ottenuto sperimentando le diverse metodologie proposte. Si può notare in maniera ancor più evidente come, contrariamente a Somma 8 Pesata, che migliora per ogni soglia di tolleranza, due delle altre tre modifiche proposte riescano a migliorare i risultati dell'algoritmo SGM soltanto con una soglia di tolleranza pari a 0, peggiorando il risultato negli altri casi.

Il margine di miglioramento ottenuto grazie all'introduzione dei pesi nella somma delle 8 scanline potrebbe apparire poco significativo, ma va tenuto presente che si tratta di incrementi prestazionali relativi ad un algoritmo, SGM, già estremamente accurato nella sua versione base.

### 5.2.3. Riduzione percentuale dell'errore

Soglia	Punti Scelti	Somma 4	Somma 4 Pesata	Somma 8 Pesata
4	-38,1783	-38,1817	-16,9698	6,740291
3	-46,2704	-46,2687	-19,0797	5,612246
2	-46,8729	-46,8762	-18,534	4,290641
1	-17,226	-17,2326	-6,03556	2,612001
0	0,919442	0,920132	1,27388	0,906101

Tab.5.3. Riduzione percentuale dell'errore medio relativo

La Tab.5.3 è forse quella che rende più giustizia ai risultati ottenuti. Notiamo subito come una differenza media dell'errore percentuale pari a 0,488201 ottenuta nel caso della somma a 8 pesata con soglia 3 si trasformi in una riduzione del 5,612264% dell'errore reattivo commesso rispetto a SGM.

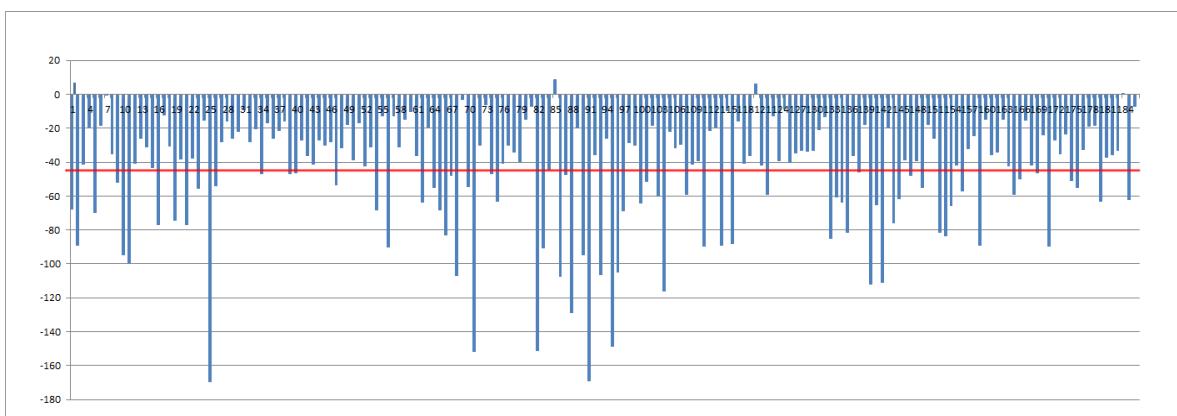
Ciò non deve sorprenderci in virtù del fatto che siamo partiti con un algoritmo in grado di fornire già ottimi risultati in termini di accuratezza, ed è per questo motivo che ad un margine di miglioramento apparentemente minimo in termini assoluti corrisponda in realtà un incremento prestazionale significativo in termini relativi.

L'altra faccia della medaglia sono le percentuali mostrate dalle altre modifiche sperimentate, che mostrano in maniera ancor più evidente gli scarsi risultati ottenibili attraverso tali modifiche.

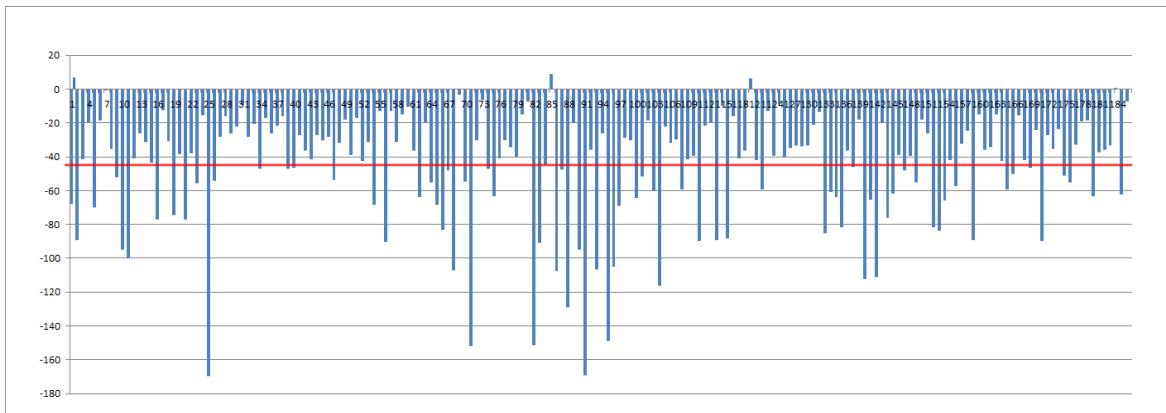
Al fine di riuscire a comprendere meglio i numeri presentati, è utile visualizzare la riduzione percentuale dell'errore rispetto a SGM per ciascuna coppia di immagini su cui sono stati effettuati i test.

Sulle ascisse di ciascun grafico sono rappresentate le coppie di immagini presenti nel training set di KITTI utilizzate per i test, sull'asse delle ordinate è rappresentata la percentuale di riduzione dell'errore per ciascuna di esse.

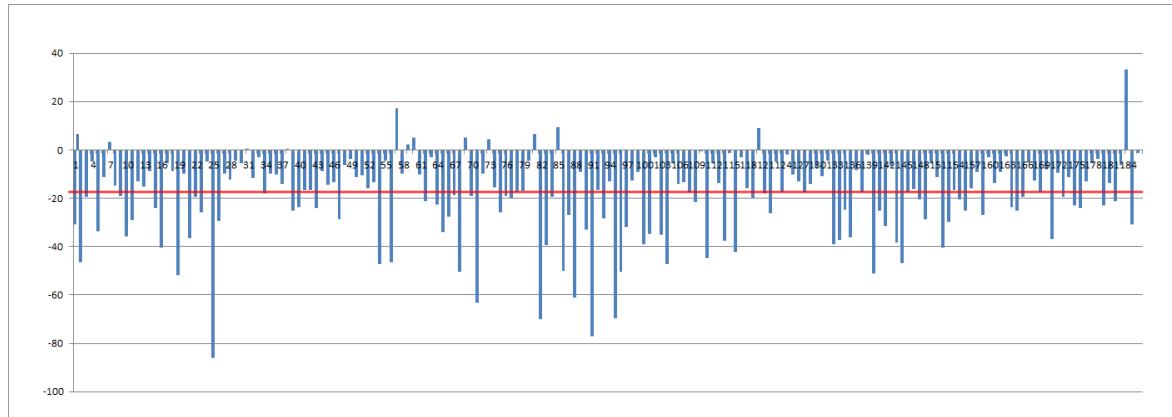
La linea rossa rappresenta la riduzione percentuale dell'errore media per la metodologia di intervento presa in analisi.



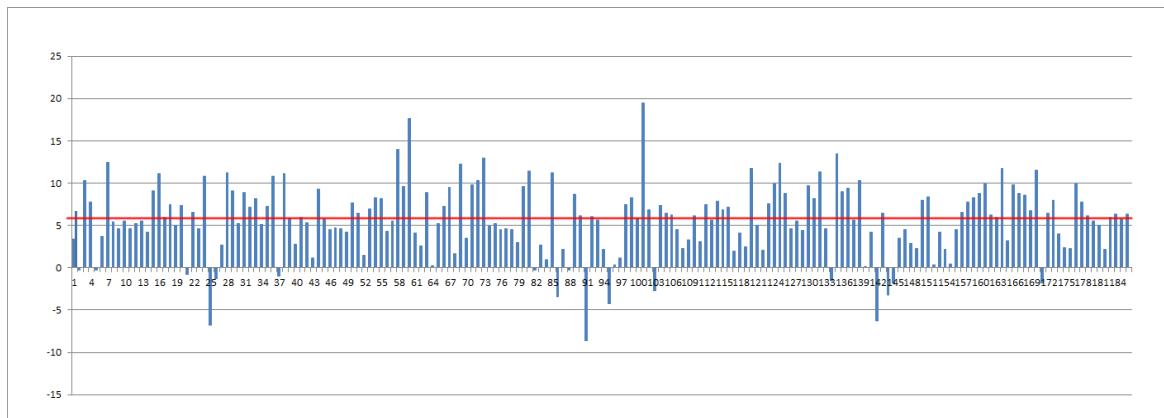
Graf.5.1. Istogramma della riduzione percentuale dell'errore, scelta del contributo proveniente dalla scanline migliore



Graf.5.2. Istogramma della riduzione percentuale dell'errore, somma dei contributi delle 4 scanline più affidabili



Graf.5.3. Istogramma della riduzione percentuale dell'errore, somma pesata dei contributi delle 4 scanline più affidabili



Graf.5.4. Istogramma della riduzione percentuale dell'errore, somma dei contributi delle scanline

Possiamo osservare come i primi tre approcci mostrino una variazione media negativa della percentuale di errore, rispecchiando globalmente un peggioramento sull'intero dataset, come già discusso in precedenza. La quarta metodologia mostra un miglioramento dell'accuratezza sull'intero dataset.

### **5.3 Confronto qualitativo con mappe di disparità**

Per ciascuna proposta di intervento proposta confronteremo la mappa di disparità migliore e peggiore ottenute (valutate con soglia 3), con le mappe di disparità corrispondenti generate utilizzando l'algoritmo SGM originale.

Questo confronto qualitativo può mettere in evidenza, in alcuni casi particolarmente favorevoli, una notevole riduzione degli errori.

#### **5.3.1. Scelta del contributo proveniente dalla scanline migliore**

##### **5.3.1.1. Risultato Migliore**

In questo caso la coppia di immagini che ha permesso di ridurre, in percentuale, l'errore di SGM della quantità maggiore è stata la coppia n°88 del training set di KITTI.



Fig.4.1. Una delle due immagini della coppia 88

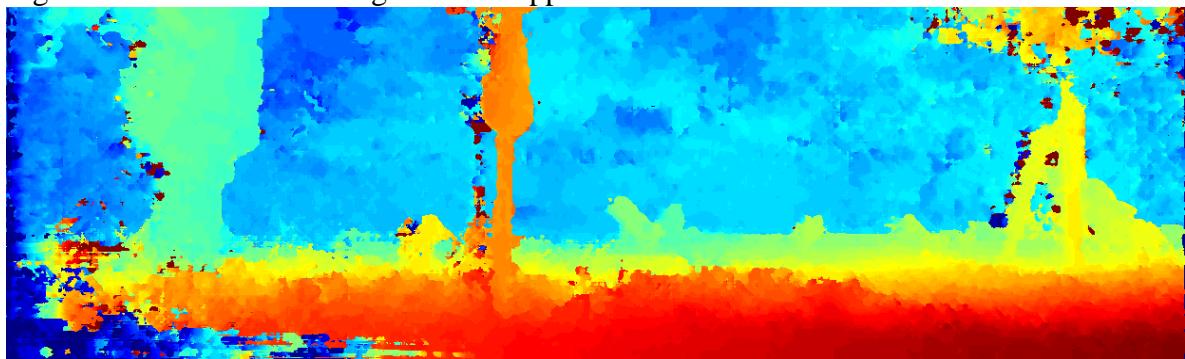


Fig.4.2. Mappa di disparità della coppia 88, algoritmo SGM

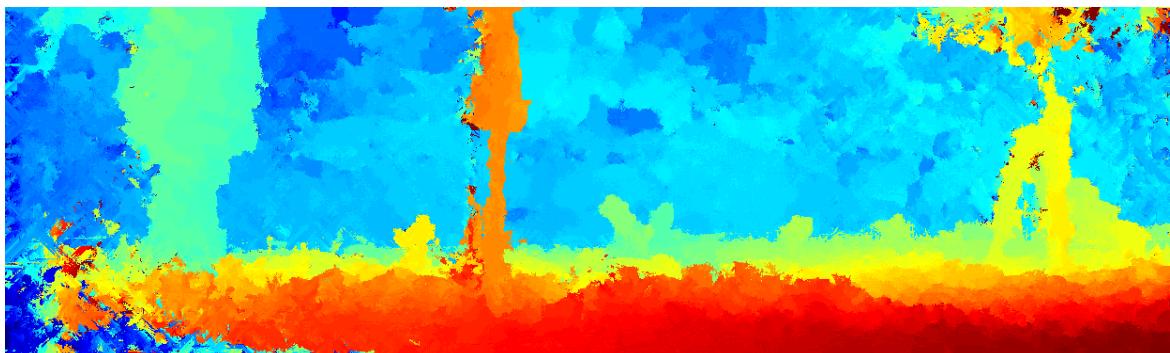


Fig.5.3. Mappa di disparità della coppia 88, selezione del contributo migliore

Dal confronto tra la *Fig.5.2* e la *Fig.5.1* appare evidente come in questo caso la modifica proposta sia riuscita a ridurre in maniera significativa alcuni artefatti presenti nella mappa di disparità ottenibile attraverso l'algoritmo SGM, è inoltre riuscita a riempire parzialmente la parte in basso a sinistra dell'immagine.

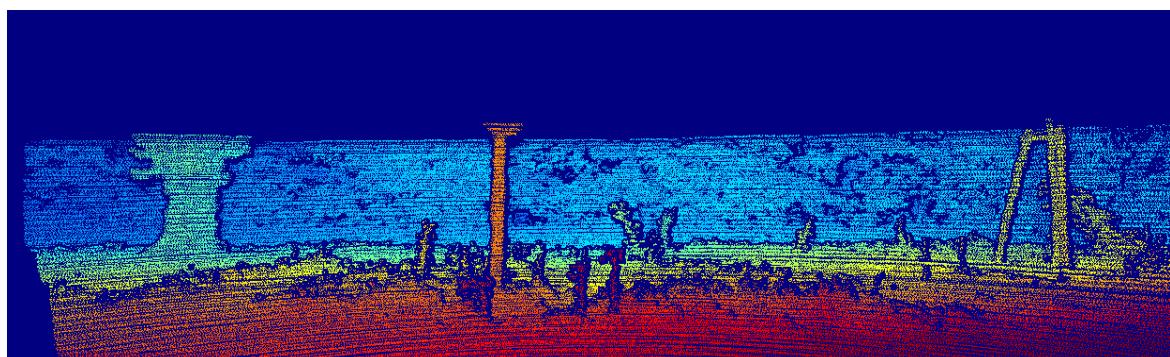


Fig.5.5. ground truth della coppia 88

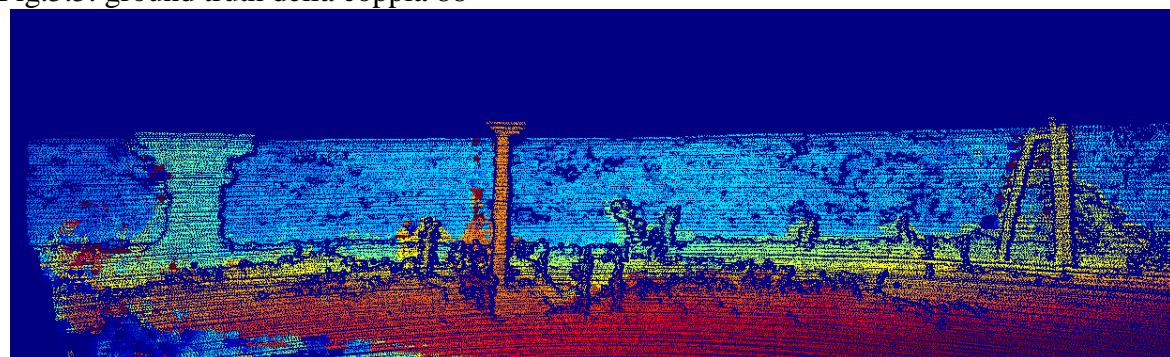


Fig.5.5 Mappa di disparità della coppia 88, algoritmo SGM, nei punti validi

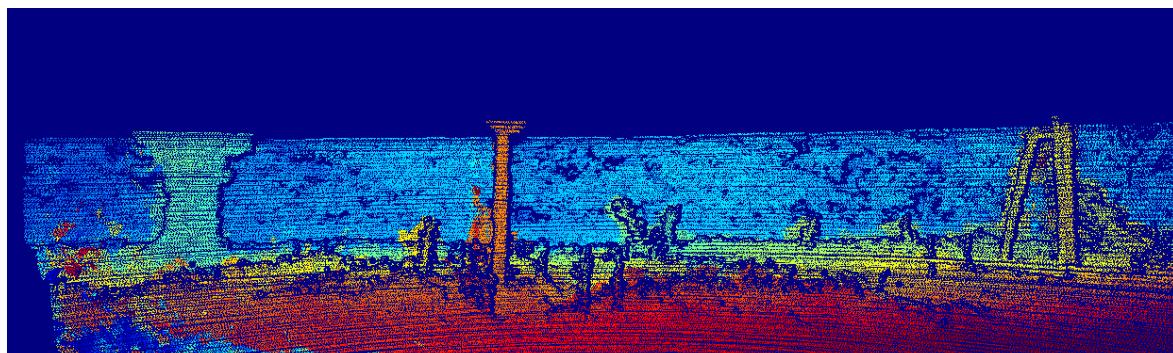


Fig.5.6 Mappa di disparità della coppia 88, selezione del contributo migliore, nei punti validi

Confrontando le figure 5.5, 5.5 e 5.6, si nota quanto i disturbi presenti nella mappa ottenuta tramite l'algoritmo SGM possano aver influito in maniera negativa sulla valutazione dell'errore percentuale.

Il metodo sperimentato è riuscito ad incrementare l'efficacia di SGM dell'8.90181% con soglia 3, commettendo un errore del 12.565% contro un errore di SGM pari al 13.793%.

### 5.3.1.2. Risultato Peggio

In questo caso la coppia di immagini con qui questo metodo ha ottenuto le prestazioni peggiori è stata la coppia n°95 del training set di KITTI.



Fig.5.7. Una delle due immagini della coppia 95

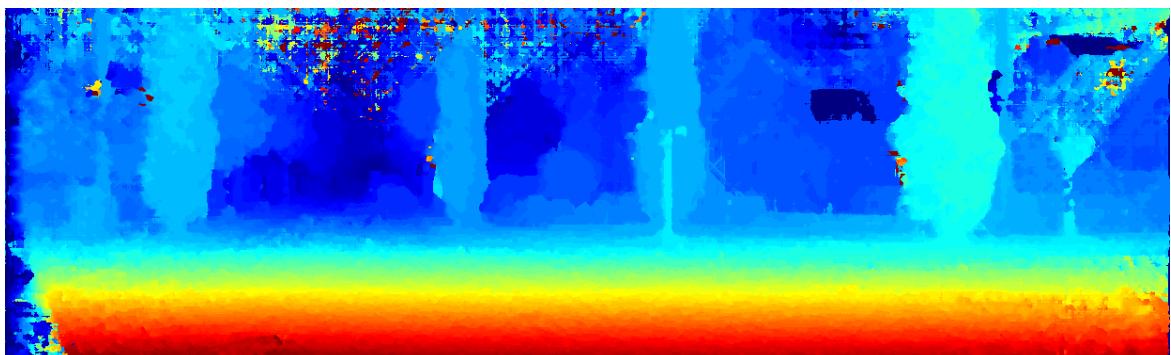


Fig.5.8. Mappa di disparità della coppia 95, algoritmo SGM

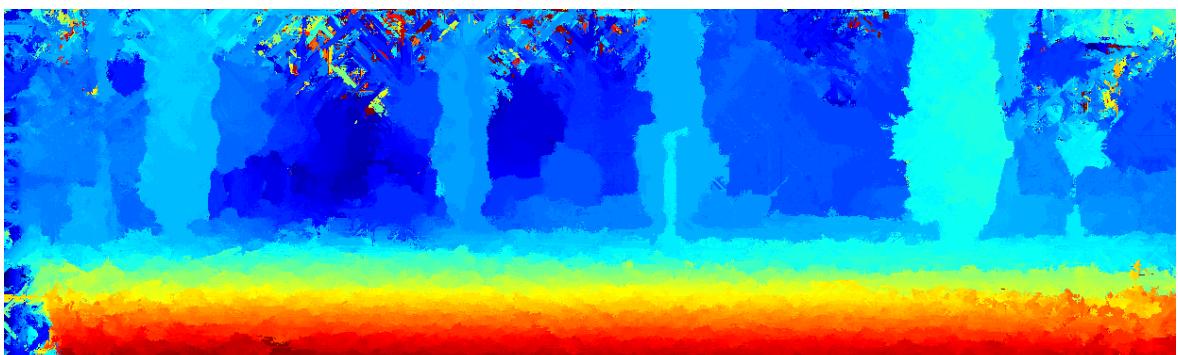


Fig.5.9. Mappa di disparità della coppia 95, selezione del contributo migliore

In questo caso le differenze sono meno evidenti, ma possiamo notare che la parte bassa dell'immagine di Fig.5.8 , al contrario della parte bassa dell'immagine di Fig.5.9, non presenta increspature né discrepanze di intensità.

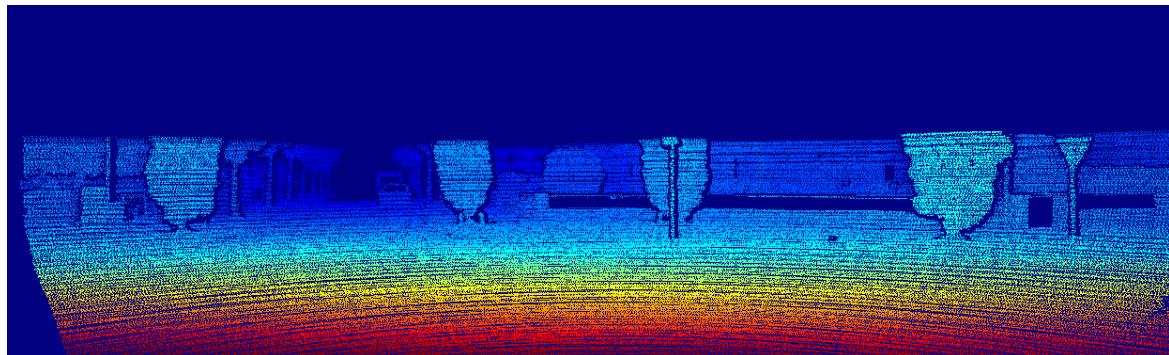


Fig.5.10 ground truth della coppia 95

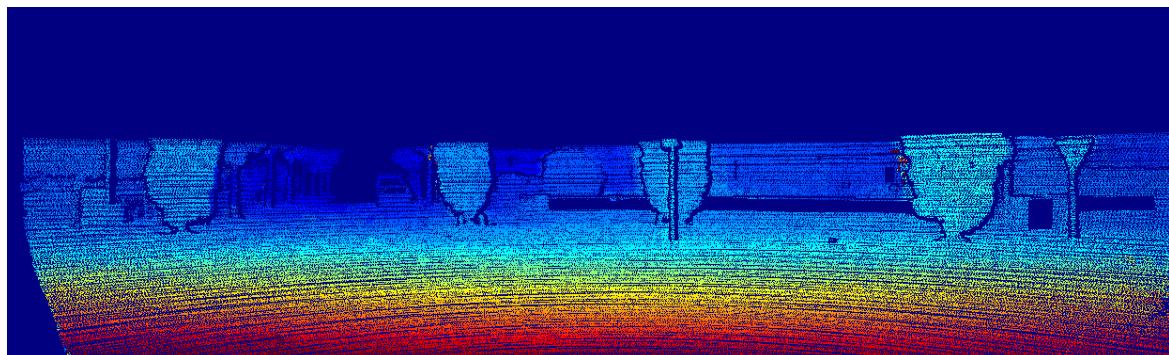


Fig.5.11 Mappa di disparità della coppia 95, algoritmo SGM, nei punti validi

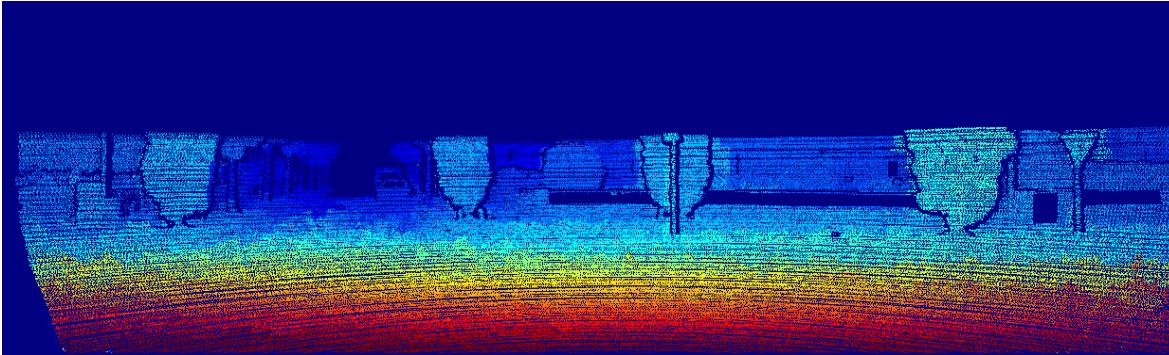


Fig.5.12 Mappa di disparità della coppia 95, selezione del contributo migliore, nei punti validi

Confrontando le figure 5.10 e 5.12, si notano differenze nell'intensità di gran parte dei pixel presenti nell'immagine di Fig.5.12, in più, il risultato ottenuto grazie ad SGM è già molto buono e ciò rende ancora più evidenti le non ottimali prestazioni del metodo proposto con questa immagine.

Il metodo sperimentato ha peggiorato il risultato ottenuto da SGM del 169.104% con soglia 3, commettendo un errore dell'8.8246% contro un errore di SGM pari al 3.2793%.

### 5.3.2. Somma dei contributi delle 5 scanline più affidabili

#### 5.3.2.1. Risultato Migliore

Anche questo caso la coppia di immagini che ha permesso di ridurre, in percentuale, l'errore di SGM della quantità maggiore è stata la coppia n°88 del training set di KITTI.



Fig.5.13. Una delle due immagini della coppia 88

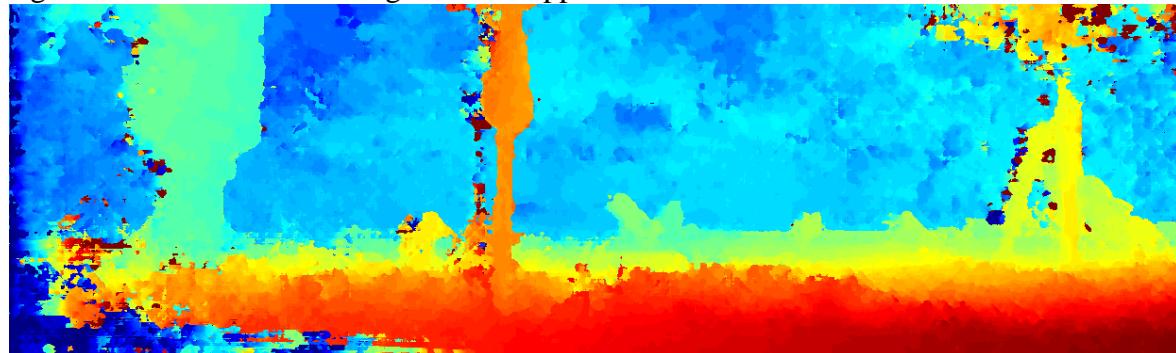


Fig.5.15. Mappa di disparità della coppia 88, algoritmo SGM

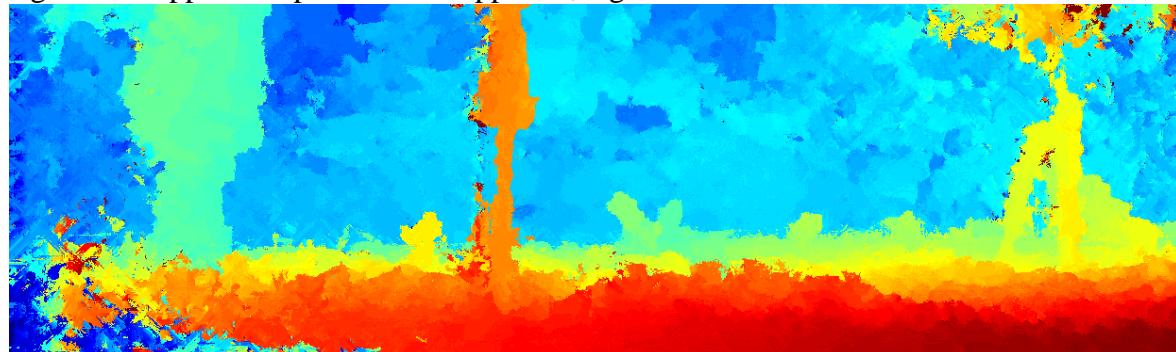


Fig.5.15. Mappa di disparità della coppia 88, somma 5 scanline

Dal momento che i risultati ottenuti sono pressoché indistinguibili, valgono le affermazioni riportate per il caso precedente:  
Appare evidente come in questo caso la modifica proposta sia riuscita a ridurre in maniera significativa gli artefatti presenti nella mappa di disparità ottenibile attraverso l'algoritmo SGM, è inoltre riuscita a riempire parzialmente la parte in basso a sinistra dell'immagine.

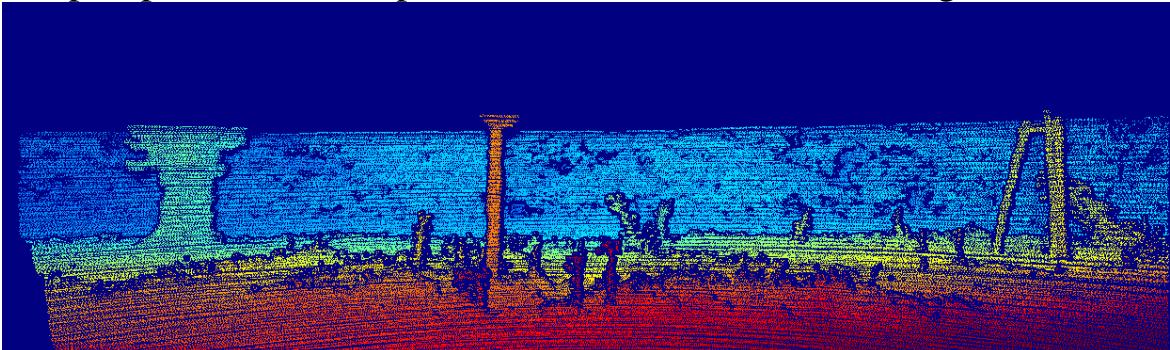


Fig.5.16 ground truth coppia 88

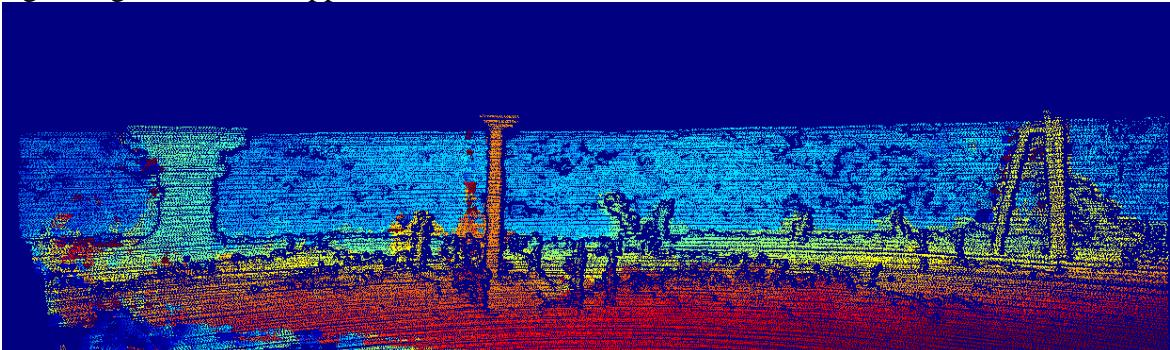


Fig.5.17 Mappa di disparità della coppia 88, algoritmo SGM, nei punti validi

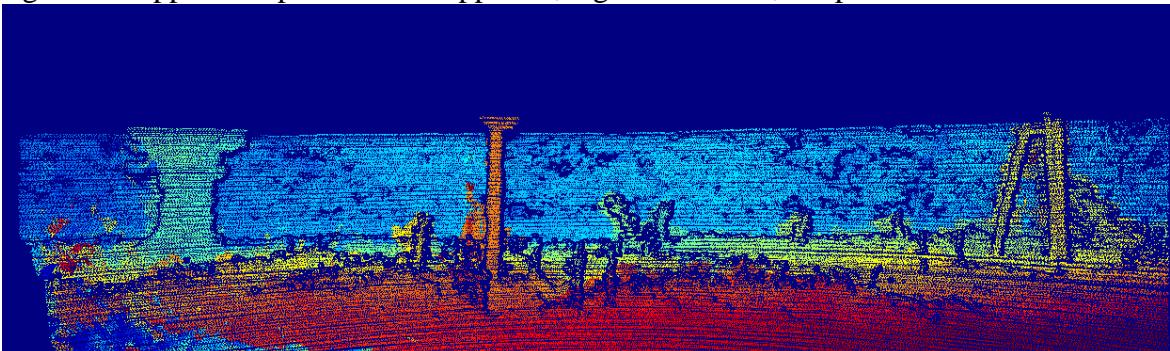


Fig.5.18 Mappa di disparità della coppia 88, somma 5 scanline, nei punti validi

Confrontando le figure 5.16, 5.17 e 5.18, si nota quanto i disturbi presenti nella mappa ottenuta tramite l'algoritmo SGM possano aver influito in maniera negativa sulla valutazione dell'errore percentuale.

Il metodo sperimentato è riuscito ad incrementare l'efficacia di SGM dell'8.90181% con soglia 3, commettendo un errore del 12.566% contro un errore di SGM pari al 13.793%.

### 5.3.2.2. Risultato Peggio

Anche questo caso in questo caso la coppia di immagini con qui questo metodo ha ottenuto le prestazioni peggiori è stata la coppia n°95 del training set di KITTI.



Fig.5.19. Una delle due immagini della coppia 95

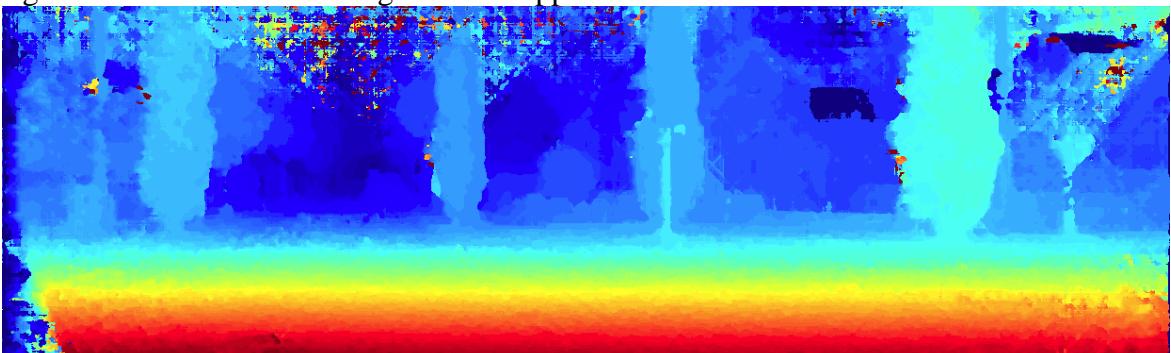


Fig.5.20. Mappa di disparità della coppia 95, algoritmo SGM

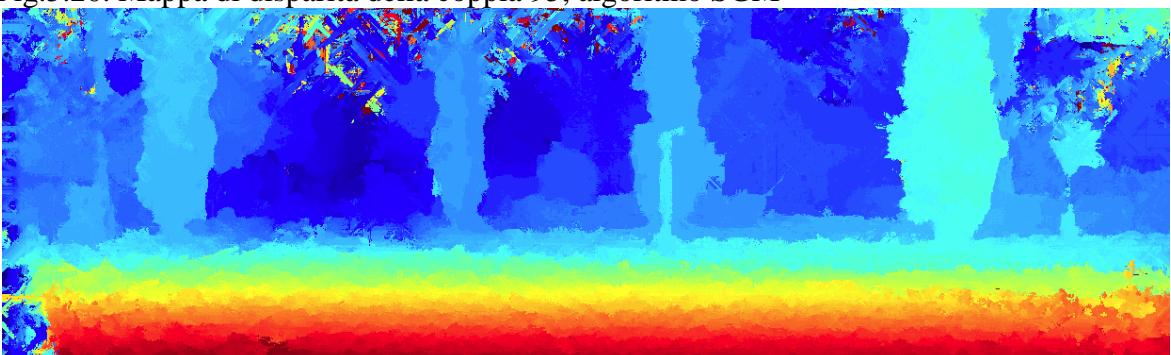


Fig.5.21. Mappa di disparità della coppia 95, somma 5 scanline

Dal momento che i risultati ottenuti sono pressoché indistinguibili, anche in questo caso valgono le affermazioni riportate per il caso precedente: In questo caso le differenze sono meno evidenti, ma possiamo notare che la parte bassa dell'immagine di Fig.5.20 , al contrario della parte bassa dell'immagine di Fig.5.21, non presenta increspature né discrepanze di intensità.

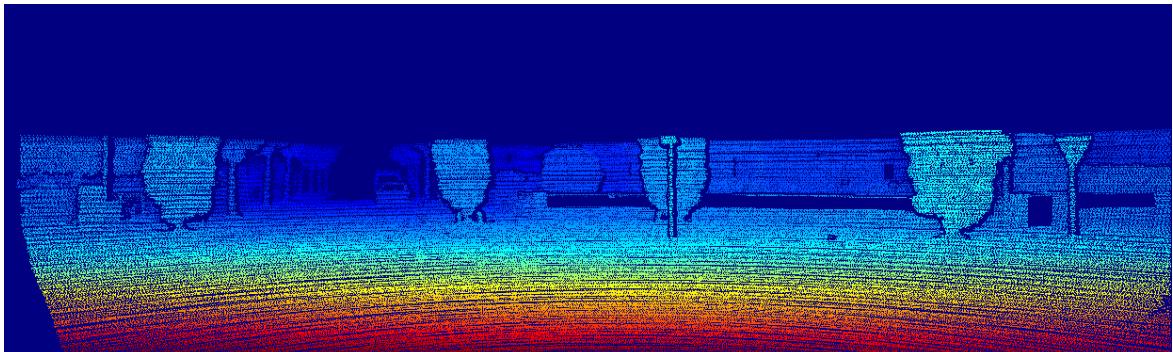


Fig.5.22 ground truth coppia 95

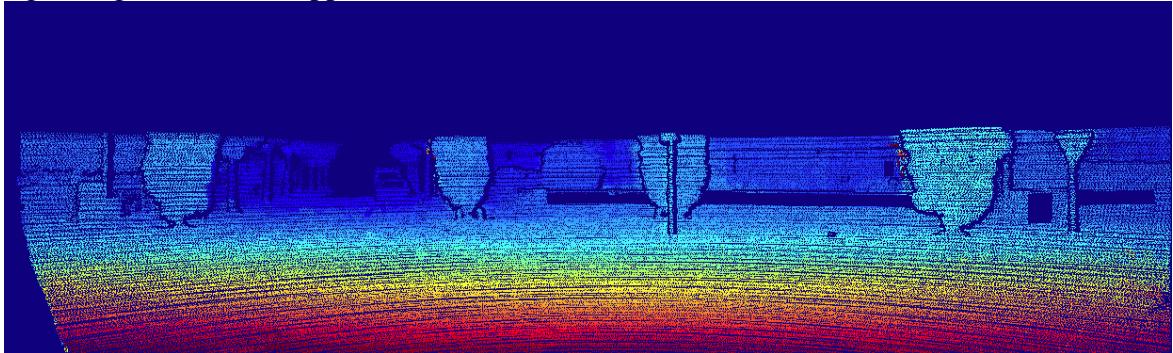


Fig.5.23 Mappa di disparità della coppia 95, algoritmo SGM, nei punti validi

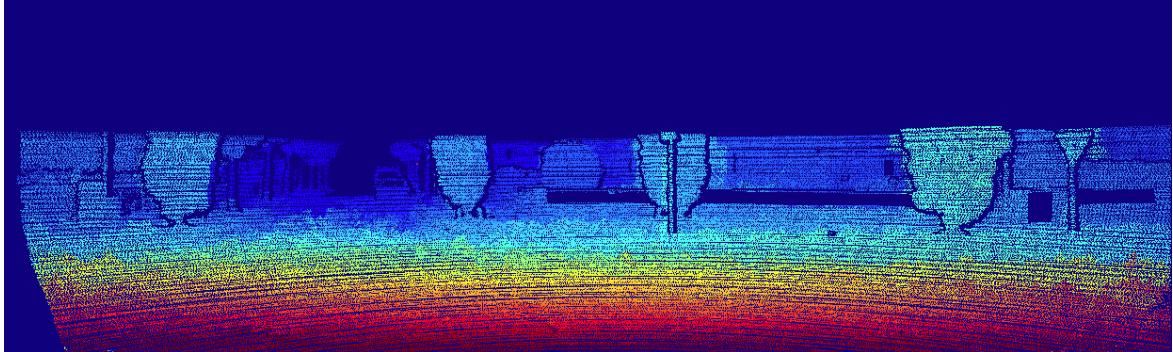


Fig.5.25 Mappa di disparità della coppia 95, somma 5 scanline, nei punti validi

Confrontando le figure 5.23 e 5.25, si notano differenze nell'intensità di gran parte dei pixel presenti nell'immagine di *Fig.5.25*, in più, il risultato ottenuto grazie ad SGM è già molto buono, e valgono le considerazioni fatte nel caso precedente.

Il metodo sperimentato ha peggiorato il risultato ottenuto da SGM del 169.089% con soglia 3, commettendo un errore del 8.8248% contro un errore di SGM pari al 3.2793%.

### 5.3.3. Somma pesata dei contributi delle 5 scanline più affidabili

#### 5.3.3.1. Risultato Migliore

In questo caso la coppia di immagini che ha permesso di ridurre, in percentuale, l'errore di SGM della quantità maggiore è stata la coppia n°191 del training set di KITTI.



Fig.5.25. Una delle due immagini della coppia 191

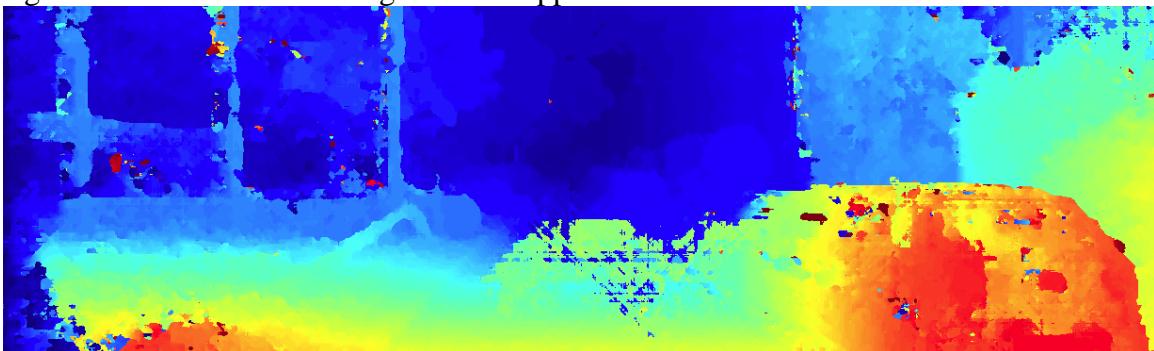


Fig.5.26. Mappa di disparità della coppia 191 algoritmo SGM

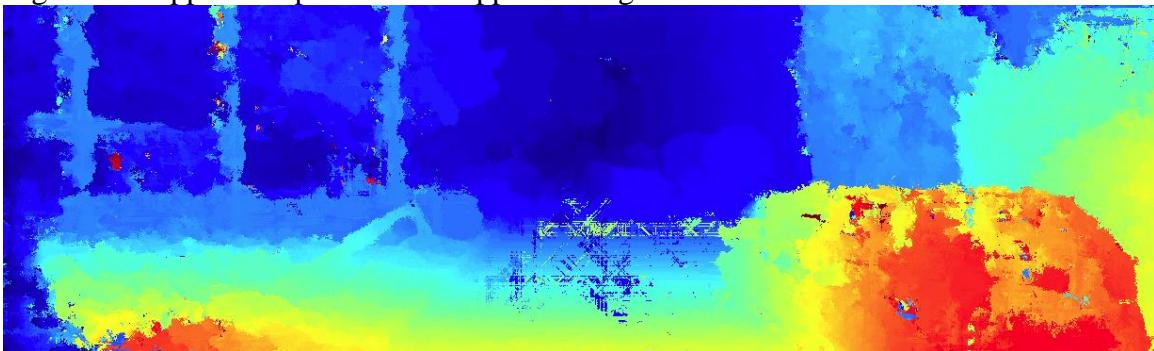


Fig.5.27. Mappa di disparità della coppia 191, somma pesata 4 scanline

Dal confronto tra la *Fig.5.26* e la *Fig.5.27* appare evidente come la miglioria che ha contribuito in maggior modo al risultato sia stata il fatto che la somma a 4 pesata è riuscita ad eliminare un evidente errore commesso dall'algoritmo SGM nella valutazione della disparità al centro della strada, dovuto probabilmente al forte riflesso della luce.

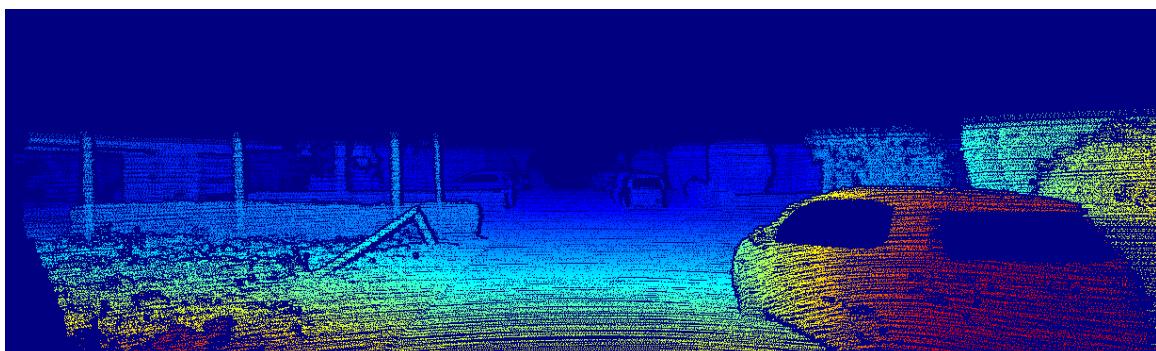


Fig.5.28 ground truth della coppia 191

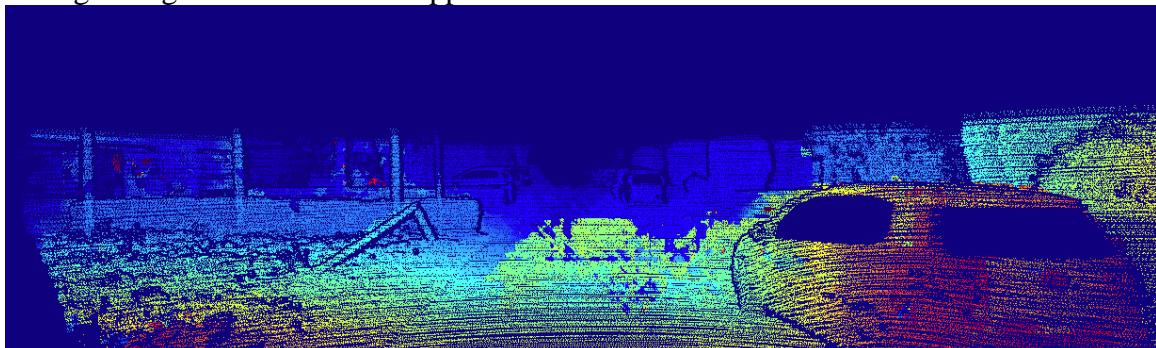


Fig.5.29 Mappa di disparità della coppia 191, algoritmo SGM, nei punti validi

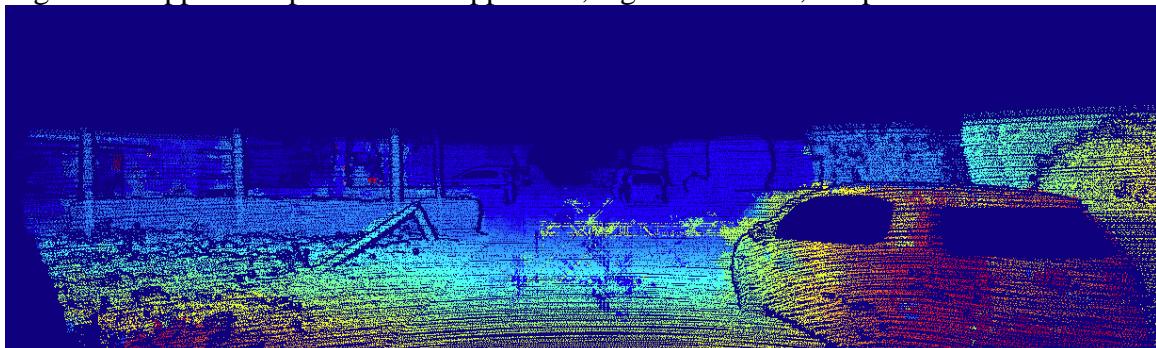


Fig.5.30 Mappa di disparità della coppia 191, somma pesata 4 scanline

Dal confronto tra le figure 5.28, 5.29 e 5.30, si nota una seconda volta come la somma a 5 pesata sia riuscita ad eliminare un evidente errore di valutazione dell'algoritmo SGM, migliorandolo quindi del 32,96518%.

Il metodo sperimentato è riuscito ad incrementare l'efficacia di SGM del 32.96518% con soglia 3, commettendo un errore del 16.54% contro un errore di SGM pari al 24.673%.

### 5.3.3.2. Risultato Peggio

In questo caso la coppia di immagini con qui questo metodo ha ottenuto le prestazioni peggiori è stata la coppia n°24 del training set di KITTI.



Fig.5.31. Una delle due immagini della coppia 24

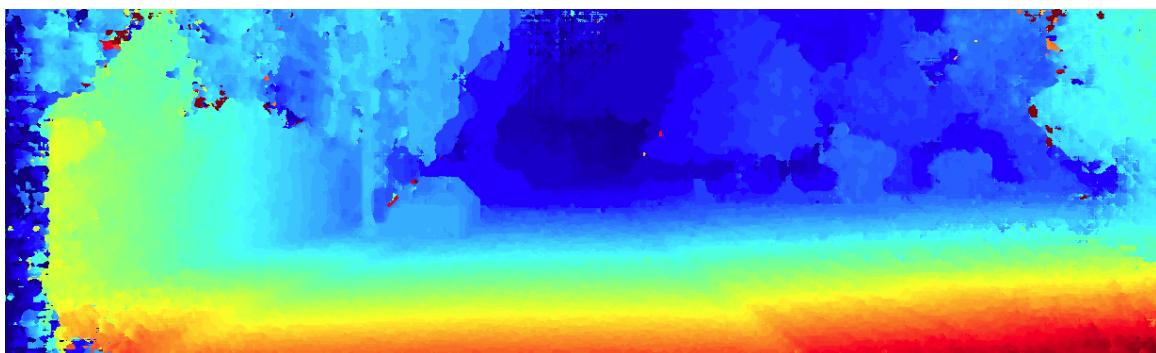


Fig.5.32. Mappa di disparità della coppia 24, algoritmo SGM

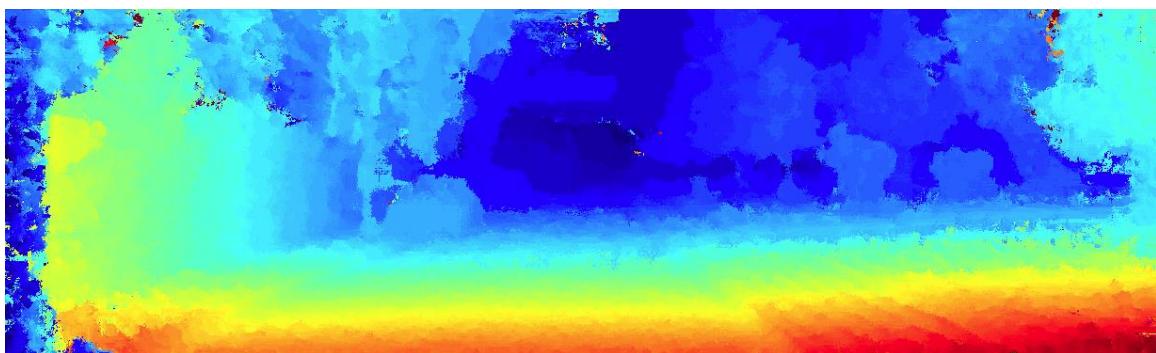


Fig.5.33. Mappa di disparità della coppia 24, somma pesata 4 scanline

Anche questo caso le differenze non sono molto evidenti, ma possiamo notare come anche questa volta la mappa ottenuta utilizzando SGM risulti più “liscia” e regolare rispetto all'altra, che presenta alcuni cenni di rumore ed increspature presenti ad ogni cambio di disparità nella parte bassa della scena.

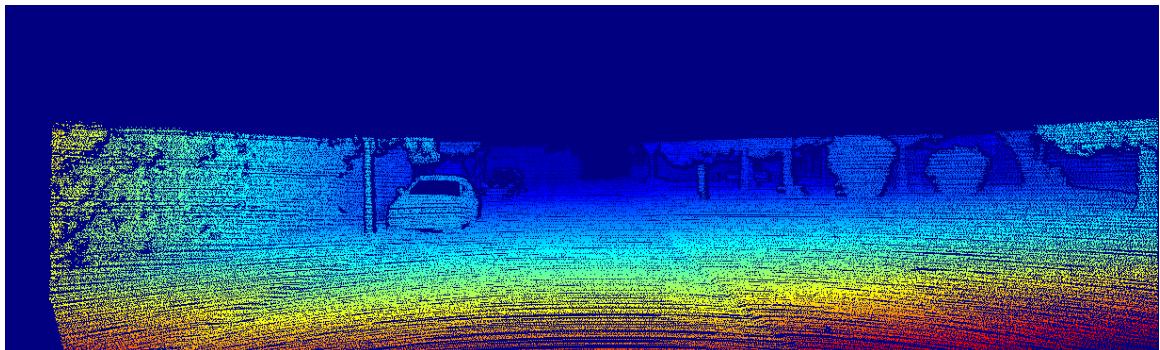


Fig.5.34 ground truth della coppia 24

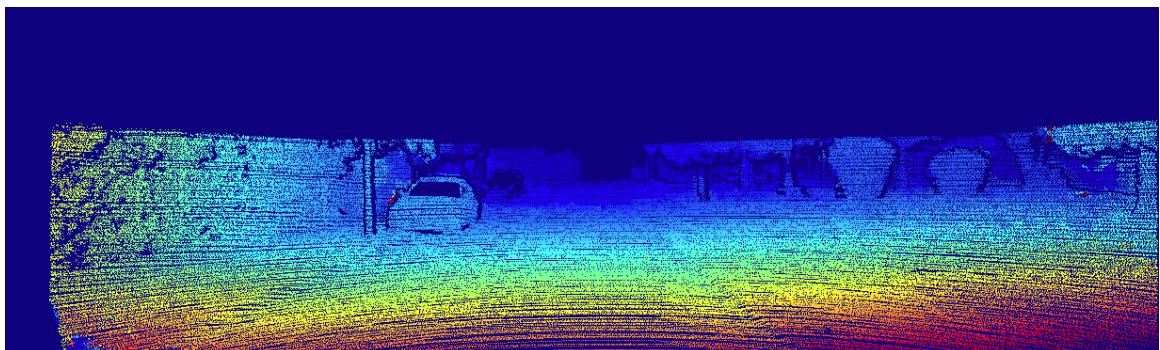


Fig.5.35 Mappa di disparità della coppia 24, algoritmo SGM, nei punti validi

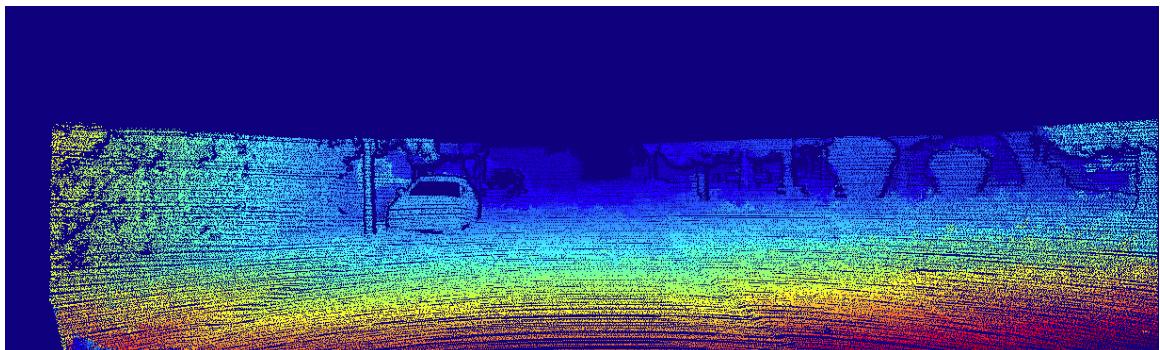


Fig.5.36 Mappa di disparità della coppia 24, somma pesata 5 scanline, nei punti validi

Confrontando le figure 5.34 e 5.36, si notano differenze nell'intensità di gran parte dei pixel presenti nella parte bassa della scena, in corrispondenza delle aree increspatte a cui si è fatto cenno precedentemente. Si nota inoltre la presenza di un accenno di rumore nella parte destra della scena, in corrispondenza dell'albero, in posizione diversa rispetto a quanto riscontrato nell'immagine di *Fig.5.35*.

Il metodo sperimentato ha peggiorato il risultato ottenuto da SGM dell'85.9376% con soglia 3, commettendo un errore del 6.8947% contro un errore di SGM pari al 3.7081%.

### 5.3.5 Somma pesata dei contributi delle 8 scanline

#### 5.3.5.1. Risultato Migliore

In questo caso la coppia di immagini che ha permesso di ridurre, in percentuale, l'errore di SGM della quantità maggiore è stata la coppia n°105 del training set di KITTI.



Fig.5.37. Una delle due immagini della coppia 105

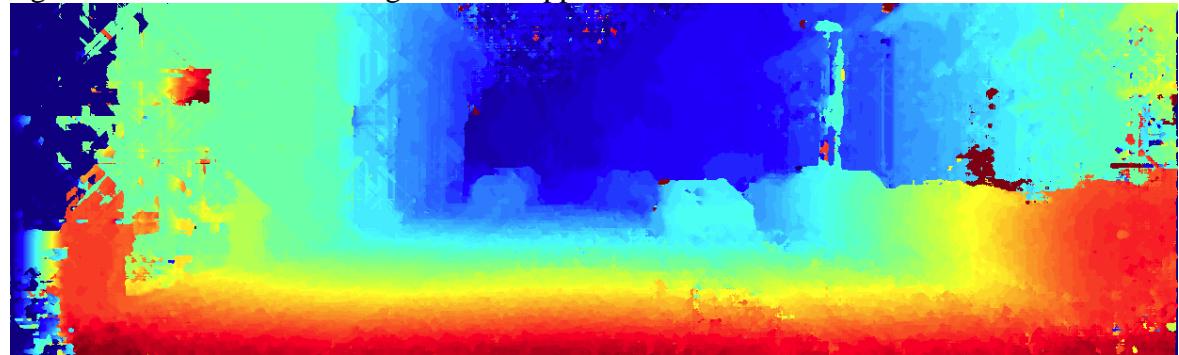


Fig.5.38. Mappa di disparità della coppia 105 algoritmo SGM

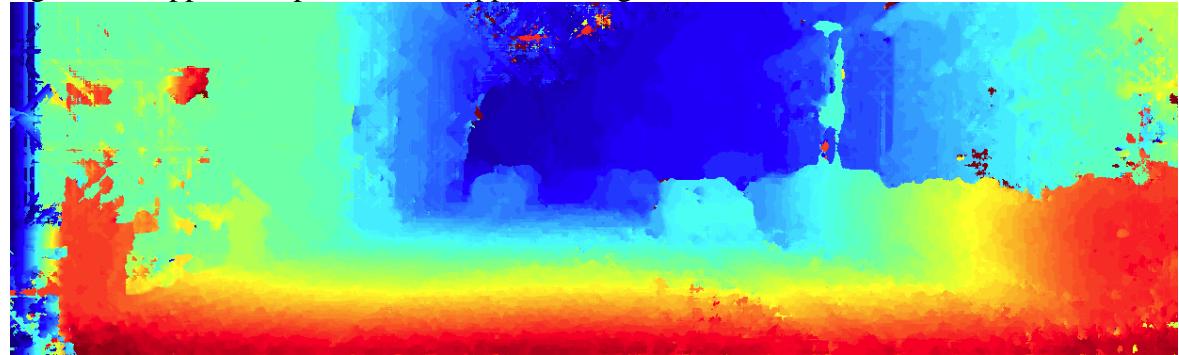


Fig.5.39. Mappa di disparità della coppia 105, somma pesata 8 scanline

Dal confronto tra la *Fig.5.38* e la *Fig.5.39* notiamo immediatamente come grazie all'utilizzo della somma pesata delle 8 scanline sia stato possibile ottenere informazioni sulla profondità della parte sinistra della scena che sarebbero state perse utilizzando il normale algoritmo SGM.

Ci accorgiamo inoltre che è stata ridotta l'entità di un disturbo presente nella parte destra dell'immagine di *Fig.5.38*, a scapito dell'introduzione di un lieve disturbo nella parte alta dell'immagine.

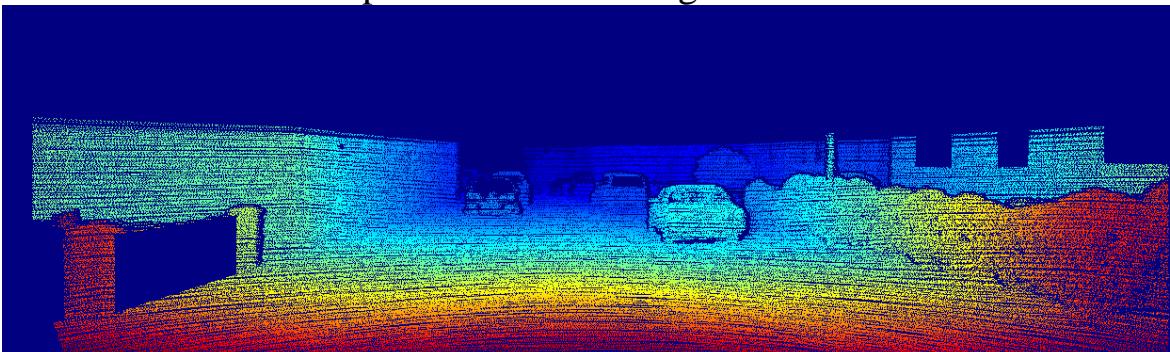


Fig.5.50 ground truth della coppia 105

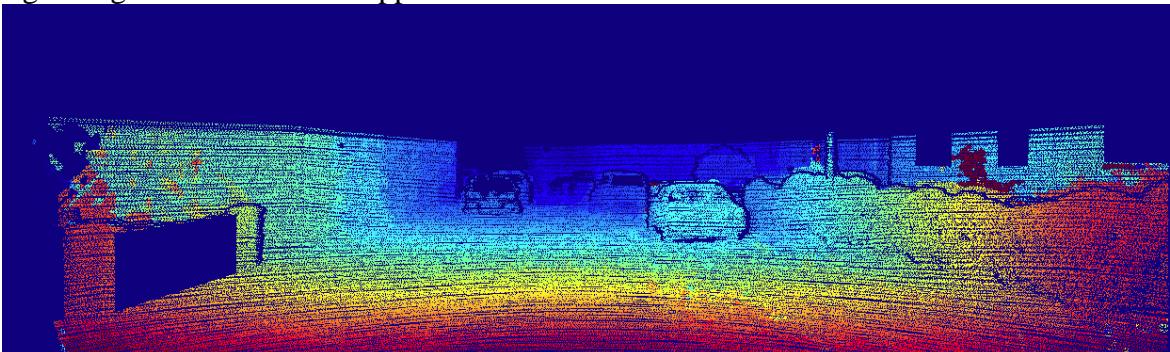


Fig.5.51 Mappa di disparità della coppia 105, algoritmo SGM, nei punti validi

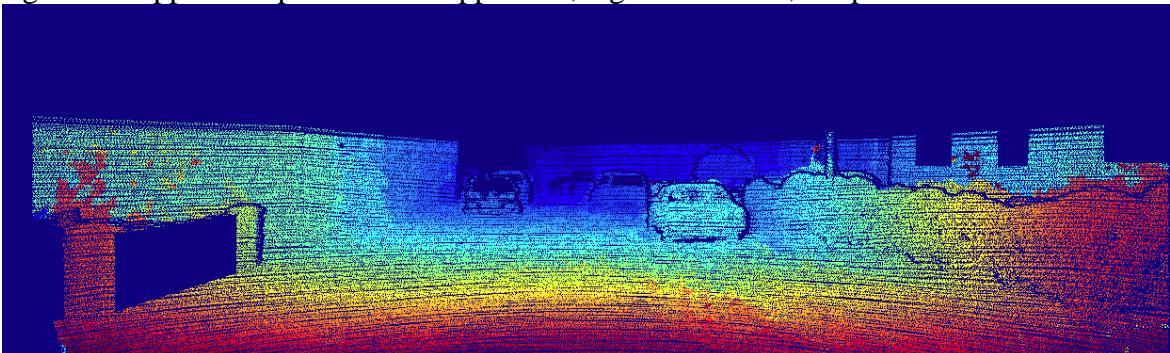


Fig.5.52 Mappa di disparità della coppia 105, somma pesata 8 scanline

Dal confronto tra le figure 5.50, 5.51 e 5.52, si nota come il contributo più importante apportato dall'introduzione dei pesi nella somma dei contributi provenienti dalle 8 scanline sia stato quello di riempire un'area di cui altrimenti avremmo perso il contenuto informativo, ha permesso inoltre di ridurre i disturbi presenti nell'immagine ottenuta tramite l'utilizzo dell'algoritmo SGM. Il disturbo introdotto nella parte alta dell'immagine non ha avuto effetti sulla qualità del risultato, dal momento che tale area non rientrava tra i punti valutabili.

Il metodo sperimentato è riuscito ad incrementare l'efficacia di SGM del 19.49303% con soglia 3, commettendo un errore del 6.03444% contro un errore di SGM pari al 7.49555%.

### 5.3.5.2. Risultato Peggio

In questo caso la coppia di immagini con cui questo metodo ha ottenuto le prestazioni peggiori è stata la coppia n°95 del training set di KITTI.



Fig.5.53. Una delle due immagini della coppia 95

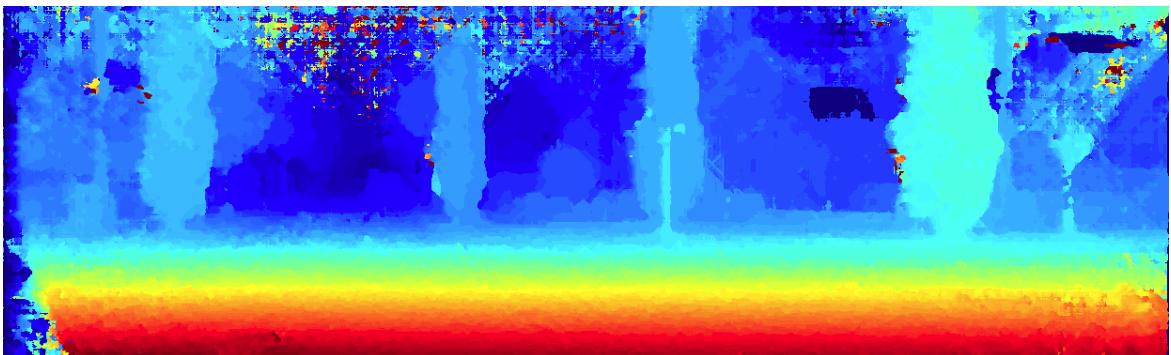


Fig.5.54. Mappa di disparità della coppia 95, algoritmo SGM

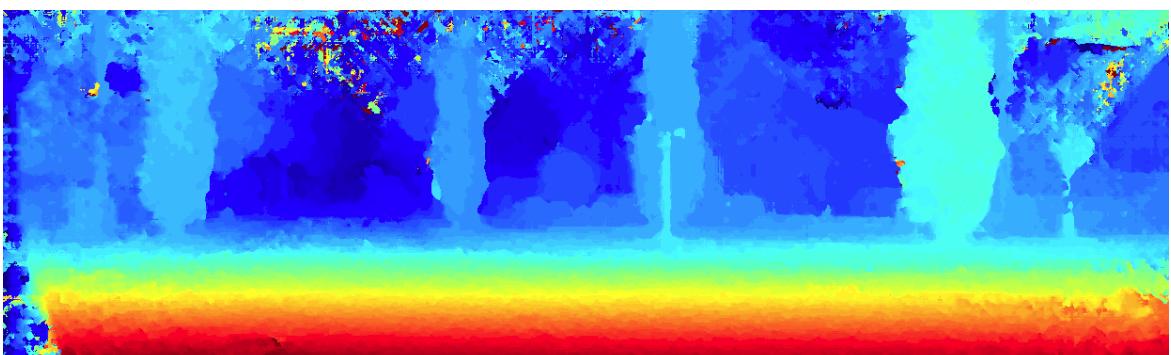


Fig.5.55. Mappa di disparità della coppia 95, somma pesata 8 scanline

Anche in questo caso le differenze non sono molto evidenti, anzi, la mappa ottenuta grazie alla somma pesata dei contributi delle 8 scanline sembra aver restituito una mappa migliore rispetto a quella ottenuta utilizzando l'algoritmo SGM, dal momento che riesce a riempire una mancanza di informazione a metà altezza ed a ridurre il rumore presente nella parte alta di *Fig.5.54*.

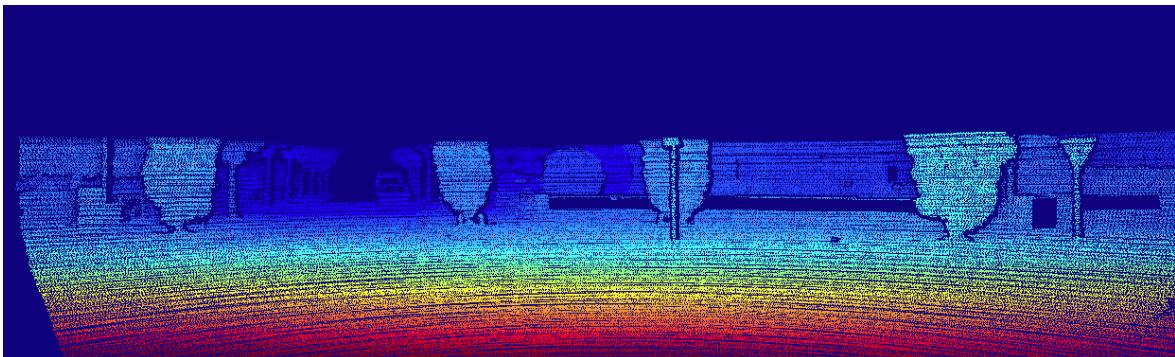


Fig.5.56 ground truth della coppia 95

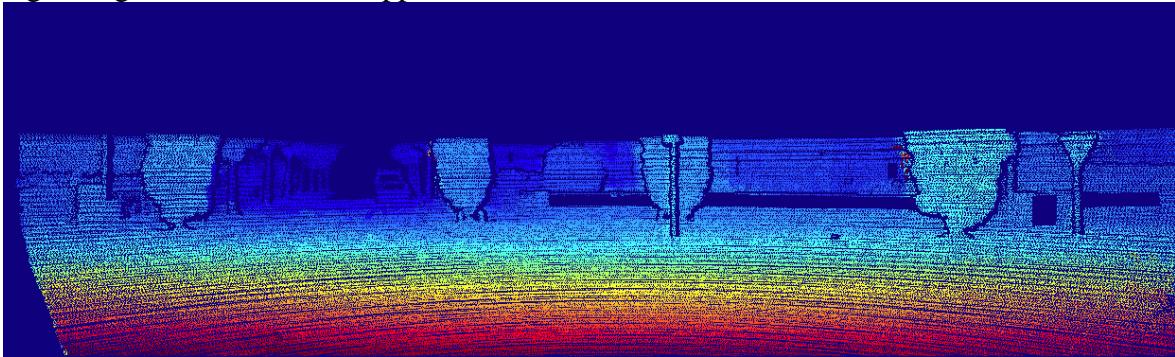


Fig.5.57 Mappa di disparità della coppia 95, algoritmo SGM, nei punti validi

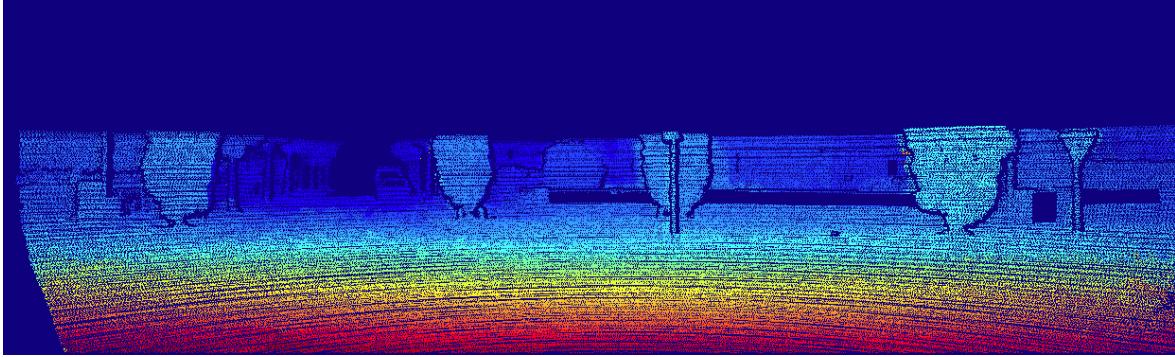


Fig.5.58 Mappa di disparità della coppia 95, somma pesata 8 scanline, nei punti validi

Osservando le figure 5.56, 5.57, e 5.58, ci accorgiamo tuttavia che il confronto non è riuscito a tener conto del miglioramento nella parte centrale e nella parte alta dell'immagine perché i punti dove la somma pesata delle 8 scanline ha fatto la differenza non sono stati valutati in quanto non presenti nella ground truth, e pertanto la mappa di disparità ottenuta è risultata peggiore a causa di lievi differenze di intensità nella parte in basso a destra della scena.

Il metodo sperimentato ha peggiorato il risultato ottenuto da SGM dell'8.64564% con soglia 3, commettendo un errore del 3.56286% contro un errore di SGM pari al 3.2793%.

## 6. Conclusioni

Dall'analisi dei risultati ottenuti si evince che tentare di perfezionare un algoritmo stereo già estremamente efficace sia un compito alquanto arduo, e nel farlo si rischia di peggiorarne le prestazioni.

Tuttavia, ciò non significa che si tratti di un compito impossibile; lo abbiamo dimostrato ottenendo risultati decisamente positivi con uno dei quattro possibili approcci tentati.

Inoltre, se guardiamo oltre ai numeri, inoltre, ci accorgiamo che forse i risultati ottenuti sono migliori di quanto i numeri stessi possano far credere.

In molte mappe di disparità generate scegliendo il contribuito indicato come migliore dalla Random Forest, è infatti possibile notare come l'intervento del Machine Learning sia stato in grado di colmare ampie mancanze di informazione o a mitigare, se non a eliminare completamente, la presenza di artefatti. Questi problemi si riscontrano sempre quando i normali algoritmi di matching stereo incontrano situazioni difficili nella scena.

In alcuni casi, le mappe di disparità nelle quali tale effetto è evidente, sono state svantaggiate probabilmente dal fatto che il confronto è eseguito soltanto rispetto ai punti validi delle ground truth, che tendono ad escludere i bordi delle scene e la parte alta che spesso contiene il cielo.

L'effetto appena descritto è riscontrato anche nelle mappe di disparità generate eseguendo la somma pesata dei contributi provenienti dalle 8 scanline, seppur in maniera lievemente minore.

Tuttavia, questo approccio introduce un significativo miglioramento in termini di accuratezza sull'intero dataset rispetto all'algoritmo SGM standard.

## Bibliografia

- [1] S. Mattoccia, “*Stereo Vision: Algorithms and Applications*”, <http://vision.deis.unibo.it/~smatt/Seminars/StereoVision.pdf>
- [2] Jianfeng Gao, “*Deep Learning for Web Search and Natural Language Processing*”, Deep Learning Technology Center (DLTC), Microsoft Research, WSDM 2015
- [3] Chuck Rosenberg, “*Improving Photo Search: A Step Across the Semantic Gap*”, <http://googleresearch.blogspot.it/2013/06/improving-photo-search-step-across.html>
- [4] Heiko Hirschmüller (2008), “*Stereo Processing by Semi-Global Matching and Mutual Information*”, in IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 30(2), February 2008, pp. 328-351.
- [5] Breiman L, “*Random Forests. Machine Learning*”, 55 (1), pp 5-32.
- [6] Zabih R. and Woodfill J., ”Non-parametric local transforms for computing visual correspondence”. (1995) In Proceedings of the third European conference on Computer Vision (ECCV). Secaucus, NJ, USA: Springer-Verlag New York, Inc., pp. 151-158.
- [7] Tom M. Mitchell. “*Machine Learning*”. McGraw Hill. 1997.
- [8] Andreas Geiger, Philip Lenz and Raquel Urtasun, ”*Are we ready*

*for Autonomous Driving? The KITTI Vision Benchmark Suite”,*  
Conference on Computer Vision and Pattern Recognition (CVPR).  
2012

- [9] Bradski, G., Dr. Dobb's Journal of Software Tools, 2000.
- [10] Min-Gyu Park and Kuk-Jin Yoon, “*Leveraging Stereo Matching with Learning-based Confidence Measures*”, Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference.
- [11] R. Haeusler, R. Nair, and D. Kondermann. “*Ensemble learning for confidence measures in stereo vision*”. In CVPR, pages 305–312, 2013.
- [12] Tesi di Laurea di Antonio Benincasa, “*Implementazione di algoritmi di matching stereo*”, AA 2015/15
- [13] Tesi di Laurea di Pasquale Presutti, “*Implementazione di algoritmi per interpolazione subpixel*”, AA 2015/15

