Visualizzazione Volumetrica Diretta Interattiva con effetti di illuminazione mediante Register Combiner OpenGL

Antonio Zorcolo, Marco Agus, Enrico Gobbetti
CRS4
VI Strada Ovest, Z.I. Macchiareddu, I-09010 Uta (CA), Italy
{zarco,magus,gobbetti}@crs4.it –
http://www.crs4.it

Sommario

In diversi ambiti scientifici e industriali, la sempre più frequente acquisizione di dati volumetrici fa crescere la domanda di potenti tool per la loro visualizzazione, analisi e manipolazione interattiva.

In questo rapporto presentiamo una tecnica di visualizzazione volumetrica diretta che consente l'applicazione del modello di illuminazione di Phong in tempo reale. La tecnica implementa il modello ottico di assorbimento + emissione eseguendo in tempo reale campionamento, mapping e integrazione delle proprietà ottiche esclusivamente attraverso funzionalità base di OpenGL 1.2. L'implementazione del modello di Phong, basato sul calcolo del gradiente di opacità, è eseguito in tempo reale utilizzando i Register Combiner. I vincoli del real-time sono soddisfatti anche grazie alla applicazione di una tecnica di accelerazione multipass basata sulla copia rapida di informazioni dal framebuffer alla memoria di texture, offerta già dalla versione 1.1 di OpenGL.

1 Introduzione

L'esplorazione e la manipolazione di grossi volumi di dati prodotti da scanner 3D (raggi X, risonanza magnetica, ultra suoni ecc.) sta diventando sempre più comune in diversi ambiti scientifici e industriali. Quanto più il progresso rende potente ed economico l'hardware di acquisizione e di processazione dei dati, tanto più la prospettiva di operare su tali volumi di dati in modo "naturale", fà crescere la domanda di potenti tool per la loro visualizzazione, analisi e manipolazione interattiva.

In campi particolarmente delicati quale quello medico, l'altissimo rischio associato all'applicazione di tecniche chirurgiche invasive, sia per l'incertezza sulla reale entità della patologia, che può essere accertata solo al momento dell'intervento, sia per la possibilità di fare esperienza solo "sul campo" da parte dei giovani chirurghi, si verifica ormai una richiesta pressante di sistemi di analisi preoperatoria che consentano al giovane chirurgo di incrementare la propria sicurezza senza alcun rischio e al chirurgo più esperto di avere un quadro più completo della patologia, di pianificare con maggior accuratezza l'intervento e di eseguire una stima più realistica dei rischi e della possibilità di successo dell'intervento stesso.

Il processo di visualizzazione dei dati volumetrici comporta un elevato sfruttamento delle risorse di calcolo a causa della elevata quantità di informazioni da elaborare, mentre i pesanti vincoli temporali imposti dalla interattività mettono alla frusta anche l'hardware più potente. In aggiunta, l'opportunità di lasciare al personale esperto la massima libertà nella scelta dei parametri di visualizzazione ottimali impone il vincolo aggiuntivo che sia preservata il più possibile l'integrità dei dati originali, eseguendo in tempo reale, sulla base dei parametri definiti dall'utente, tutti i passi necessari a produrre limmagine richiesta direttamente dai dati "grezzi".

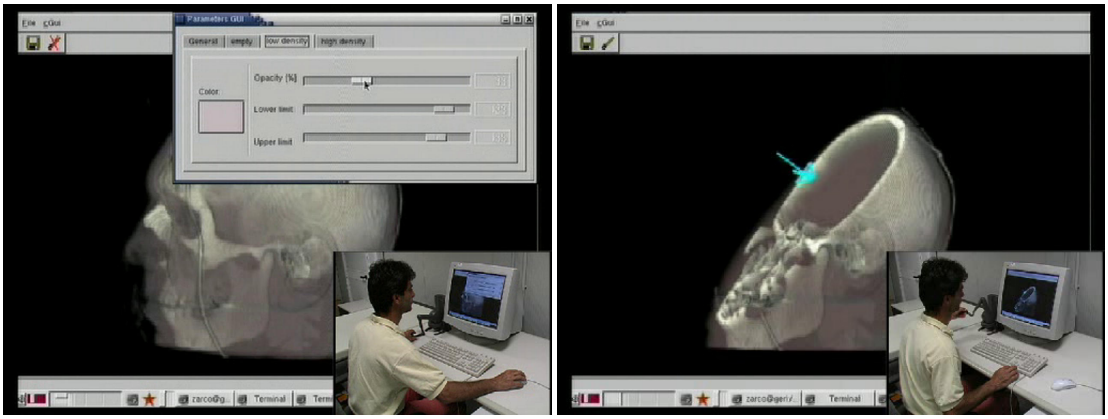
L'approccio più generale alla visualizzazione di dati volumetrici è la visualizzazione diretta, una tecnica che non richiede la preventiva ricostruzione di una rappresentazione geometrica, ma genera una immagine del volume direttamente dai dati in esso contenuti. La possibilità di ridefinire in tempo reale la funzione che mappa sui dati originali le desiderate proprietà ottiche, offre una grande flessibilità consentendo il raggiungimento di svariati effetti, compresa la visualizzazione di isosuperfici e l'inclusione nel volume di oggetti opachi.

La principale espressione del potenziale della visualizzazione volumetrica diretta è la possibilità offerta all'utente di esaminare ed interagire col volume interattivamente ed in modo assolutamente naturale. La possibilità di generare viste prospettiche in tempo reale consente di incrementare sensibilmente la percezione della profondità consentendo, anche grazie all'effetto del parallasse di movimento, di implementare tecniche di interazione diretta per l'esecuzione di complesse azioni tridimensionali, quali la ricerca del punto di vista ottimale ed il posizionamento di piani di sezionamento del volume, con assoluta naturalezza. Sotto questo aspetto la visualizzazione volumetrica diretta è sicuramente la tecnica più promettente nella realizzazione di applicazioni di simulazione. Per contro, sopratutto se vincolata ai tempi di risposta di tali applicazioni, essa presenta il limite di consentire l'adozione di modelli ottici molto semplificati, con una evidente limitata qualità delle immagini.

In questo rapporto presentiamo una tecnica di visualizzatore diretta di dati volumetrici che, pur soddisfacendo i rigidi vincoli temporali imposti da applicazioni time-critical, garantisce una fedeltà di riproduzione visiva normalmente non raggiungibile da un renderer volumetrico interattivo. Tale fedeltà è raggiunta grazie alla applicazione del modello di illuminazione di Phong, resa possibile dalle altissime prestazioni raggiunte dai moderni processori grafici nell'esecuzione del multitexturing, che consentono di stimare in tempo reale le normali alle strutture rilevanti presenti nel volume attraverso la valutazione del gradiente di opacità.

Per meglio evidenziarne le peculiarità il resto del rapporto è strutturato come segue:

Nella sezione 2 defininiremo cosa si intenda per visualizzazione volumetrica diretta, quali siano i principali vincoli temporali nelle applicazioni interattive e quale sia il modello ottico implementato; nella sezione 3 descriveremo in dettaglio le caratteristiche del visualizzatore in relazione all'hardware impiegato; nella sezione 4 descriveremo infine la piattaforma sulla quale è stato testato il sistema e forniremo una valutazione delle sue prestazioni; nell'appendice A riportiamo inoltre alcune considerazioni sulle prestazioni raggiungibili dall'hardware grafico, ed alcuni valori di riferimento.



(a) (b)

Figura 1: Classificazione e sezionamento di un dataset medico.

2 Modello ottico per la visualizzazione volumetrica diretta

Per visualizzazione volumetrica diretta si intende la generazione di una immagine del dataset direttamente dai dati volumetrici, senza passare per ricostruzioni intermedie, mappando direttamente (transfer function) le richieste proprietà ottiche (colore, coefficienti di emissione e di assorbimento) sulla grandezza fisica rilevata.

Il volume viene visto come costituito da innumerevoli particelle in sospensione (una nube di pulviscolo o una sorta di nebbia), o come una massa gelatinosa. L'aspetto della sua proiezione sul piano di vista è ottenuta integrando lungo il percorso dei raggi di luce le funzioni di assorbimento/emissione/riflessione che definiscono, con modelli matematici più o meno complessi, la risposta alla luce delle particelle sospese o della massa gelatinosa.

La complessità computazionale dell'algoritmo di visualizzazione è quindi molto elevata e dipende fortemente sia dal livello di approssimazione dell'integrale che dalla densità dei raggi di luce lungo i quali esso è valutato, nonché dall'accuratezza del modello matematico che descrive il comportamento ottico delle particelle.

In generale, per applicazioni interattive si sacrificherà la qualità delle immagini, accettando approssimazioni più grossolane dell'integrale ed adottando modelli matematici semplificati, a favore della rapidità di esecuzione. Infatti, una delle principali esigenze percettive dell'essere umano nella interazione con ambienti virtuali tridimensionali è la *fedeltà temporale*, i cui parametri di valutazione sono essenzialmente la frequenza di presentazione dei fotogrammi ed il tempo di latenza. La prima è l'inverso del tempo necessario al sistema per calcolare e visualizzare un nuovo fotogramma che dipende, a sua volta, dalla complessità del modello ottico, dalle dimensioni del volume di dati e dalla risoluzione del fotogramma. La minima frequenza di fotogramma che l'utente può accettare è molto variabile col tipo di compito che deve svolgere e può variare da un minimo di 10 fps fino anche a 50 fps. Il secondo è il tempo complessivo intercorrente fra l'azione dell'utente e la reazione del sistema. Pur non essendo indipendente dal tempo di fotogramma, il tempo di latenza dipende da molti altri fattori e può essere anche considerevolmente maggiore del primo. Ad esempio, il ricorso a strutture parallele a pipeline per ridurre il tempo di fotogramma ha effetto negativo sul tempo di latenza. Esperimenti condotti su sistemi virtuali cooperativi distribuiti [8] e su sistemi virtuali con tracciamento del punto di vista [1] hanno mostrato come la difficoltà nel manipolare oggetti virtuali sia proporzionale alla latenza del sistema già a partire dai 40 ms, e che latenze superiori ai 300 ms rendono quasi impossibile il completamento del compito. In [4, 5] si evidenzia inoltre la sensibilità dell'utente a variazioni di latenza anche inferiori ai 30 ms.

Fra i vari modelli ottici proposti per la visualizzazione volumetrica diretta [7], quello generalmente adottato per sistemi interattivi è il modello di "assorbimento + emissione" per il quale la variazione di intensità di un raggio di luce che attraversa una sezione *S* del volume per un tratto di lunghezza infinitesima ds è definito dalla equazione differenziale:

$$\frac{ds}{Is} = g(s) - \tau(s)I(s) = C(s)\tau(s) - \tau(s)I(s) \tag{1}$$

con $\tau(s)$, $I(s)$, $C(s)$ rispettivamente coefficiente di estinzione, intensità luminosa e intensità riflessa/emessa specifica in *s*, che integrata fra *0* ed *s* definisce la intensità luminosa risultante in *s*.

L'integrale della (1), assumendo un passo d'integrazione pari allo spessore del voxel, coefficiente di estinzione $\tau(n)$ e intensità riflessa (colore del voxel) $C(n)$ costanti nel voxel stesso, è approssimato con la (*particle model* [7]):

$$\widetilde{C}_N = \sum_{n=1}^{N} C[n]\alpha[n] \prod_{m=1}^{n-1} (1 - \alpha[m]), \tag{2}$$

dove $\widetilde{C}_N$ è il colore associato [2] risultante da *N* slice del volume, mentre la quantità:

$$T_N = \prod_{n=1}^{N} (1 - \alpha[n]) \tag{3}$$

rappresenta la trasparenza equivalente degli n strati attraversati dal raggio di luce.

Utilizzando esclusivamente colori associati, la (2) diviene:

$$\widetilde{C}_N = \sum_{n=1}^{N} \widetilde{C}[n] \prod_{m=1}^{n-1} (1 - \alpha[m]) \tag{4}$$

che, indicando con $t[m]$ la trasparenza della slice definita dalla differenza $(1 - \alpha[m])$, diviene ancora [9]:

$$\widetilde{C}_N = \sum_{n=1}^{N} \widetilde{C}[n] \prod_{m=1}^{n-1} t[m]. \tag{5}$$

Alla nesima iterazione del processo di integrazione, l'apporto della n-esima slice al colore totale $\widetilde{C}_N$ è dato da:

$$\widetilde{C}_N = \widetilde{C}[n] + (1 - \alpha[n])\widetilde{C}_{N-1}, \tag{6}$$

dove $\widetilde{C}_{N-1}$ è il colore associato accumulato nelle iterazioni precedenti, se l'accumulazione avviene in back-to-front, oppure da:

$$\widetilde{C}_N = \widetilde{C}_{N-1} + (1 - \alpha_{N-1})\widetilde{C}[n] = \widetilde{C}_{N-1} + T_{N-1}\widetilde{C}[n], \tag{7}$$

dove α_{N-1} è l'opacità complessiva dei primi N-1 strati valutata con la (3), se l'accumulazione avviene in front-to-back.

Le caratteristiche del termine $g(s)$, così come compare nella (1), sono però tali che l'immagine prodotta dal processo d'integrazione risulti povera di dettagli, particolamente importanti per l'uso al quale il visualizzatore è destinato. La percezione dei dettagli dipende infatti dall'andamento della normale alla superficie delle strutture presenti nel volume, non valutata nella (1). Per questo motivo si preferisce utilizzare al posto di $g(s)$ la funzione $g(s, \bar{\omega})$ (*scattering* [7]) che tiene conto, oltre che del già considerato termine non direzionale $C(s)$ anche di un termine dipendente dalla direzione di illuminazione. In generale si pone:

$$g(s, \bar{\omega}) = E(s) + L(s, \bar{\omega}) \tag{8}$$

dove $E(s)$ è la componente di radiazione non direzionale (emissiva o riflessiva) e $L(s, \bar{\omega})$ è la componente direzionale, funzione della direzione di vista $\bar{\omega}$.

Nell'ipotesi che il generico voxel del volume sia raggiunto, senza alcuna azione di oscuramento/assorbimento da parte dei restanti voxel e con direzione $\bar{\omega}'$, dalla radiazione $i(s, \bar{\omega}')$ emessa da una sorgente direzionale, e indicata con $p(s, \bar{\omega}, \bar{\omega}')$ la *funzione di fase* in *s* che determina il fattore di riflessione nella direzione di vista $\bar{\omega}$, il termine $L(s, \bar{\omega})$ della (8) può essere valutato con:

$$L(s, \bar{\omega}) = A(s)\tau(s)p(s, \bar{\omega}, \bar{\omega}')i(s, \bar{\omega}') \tag{9}$$

in cui $\tau(s)$ è il già citato coefficiente di estinzione e $A(s)$ il fattore di riflessione (*particle albedo*).

Se si considera la sola riflessione Lambertiana , la funzione di fase dipende solo dalla direzione di illuminazione $\bar{\omega}'$ e dalla normale $\bar{n}$ alla superficie in *s*, che, a sua volta, può essere supposta uguale alla direzione del gradiente della funzione *f* codificata nel volume:

$$L(s, \bar{\omega}) = i(s, \bar{\omega}')A(s)\tau(s)max\left(\frac{\nabla f(s)}{\|\nabla f(s)\|} \cdot \bar{\omega}', 0\right). \tag{10}$$

Sostituendo dunque nella (8) il valore di $L(s, \bar{\omega})$ dato dalla (10) e assumendo che il termine $E(s)$ della (8) sia di tipo riflessivo piuttosto che emissivo, il colore associato $\widetilde{C}[n]$ che compare nella (6) può essere espresso come:

$$\begin{aligned} \widetilde{C}[n] &= \widetilde{C}_a[n]l_a + \widetilde{C}_d[n]max\left(\frac{\nabla f[n]}{\|\nabla f[n]\|} \cdot \bar{l}_d, 0\right) \\ &= \alpha[n]C_a[n]l_a + \alpha[n]C_d[n]\left\|\bar{l}_d\right\| max\left(\frac{\nabla f[n]}{\|\nabla f[n]\|} \cdot \frac{\bar{l}_d}{\left\|\bar{l}_d\right\|}, 0\right) \end{aligned} \tag{11}$$

in cui $C_a[n]$, $C_d[n]$ e $\alpha[n]$ sono il fattore di riflessione ambiente (non direzionale), il fattore di riflessione diffusa (direzionale) e l'opacità del voxel campionato sulla n-esima slice, mentre l_a e $\bar{l}_d$ sono l'intensità della luce ambiente e l'intensità luminosa generata dalla sorgente direzionale che investono il voxel.

Per evidenziare i dettagli superficiali, si può ricorrere all'artificio di pesare l'opacità $\alpha[n]$ con lo *strength* della superficie, valutato, in generale, come funzione della f e del suo gradiente: $S = h\left(f(s), \nabla f(s)\right)$ [3, 6]. Utilizzando come *strength* il modulo del gradiente la (11) diviene:

$$\widetilde{C}[n] = l_a \left\|\nabla f[n]\right\| \alpha[n] C_a[n] + \left\|\bar{l}_d\right\| \alpha[n] C_d[n] max\left(\nabla f[n] \cdot \frac{\bar{l}_d}{\left\|\bar{l}_d\right\|}, 0\right), \tag{12}$$

il cui effetto è di visualizzare esclusivamente le superfici di separazione dei tessuti, rendendo completamente trasparenti le zone a gradiente nullo.

3 Visualizzazione interattiva su hardware COTS

I volumi di dati generati da scanners medicali sono costituiti da griglie regolari di punti di campionamento, con spaziature che possono variare fra qualche decimo di millimetro a qualche millimetro, per ciascuno dei quali viene restituito un valore scalare con una precisione variabile fra gli 8 e i 16 bit (di norma 12 per la tomografia computerizzata). Questa caratteristica li rende facilmente utilizzabili, senza importanti manipolazioni, come texture scalari: direttamente come texture 3D, se l'hardware lo consente, oppure come pile di texture 2D allineate, ricavate dai piani della griglia con normale prossima alla direzione di vista.

Il processo di integrazione viene eseguito, indifferentemente in back-to-front o front-to-back, campionando il volume (sui piani delle slice, nel caso di texture 2D) per mezzo dell'hardware di texturing. Il processo equivale a "lanciare", dal punto di osservazione, tanti raggi quanti sono i pixel ricoperti dalla finestra di vista, uno per pixel, ed eseguire un passo di integrazione su ciascun raggio ad ogni iterazione implementando le equazioni (6) e (7) con l'opportuna funzione di blending OpenGL. Ad ogni passo e per tutti i pixel l'hardware esegue un accesso alla texture e ne estrae il valore scalare. Il valore campionato viene convertito, tramite una opportuna funzione di trasferimento, in una terna di colore ed un valore di opacità che vengono salvati nei registri d'ingresso dei combiner. I combiner, opportunamente programmati per implementare l'equazione (12), restituiscono il colore $\widetilde{C}_N$ atteso dalla funzione di blending. La necessità di disporre nei combiner anche del gradiente $\nabla f[n]$ per il calcolo di $\widetilde{C}_N$ viene soddisfatta grazie alla capacità dell'hardware di texturing di eseguire più di un campionamento della texture per pixel (*multitexturing*).

Di seguito diamo una descrizione più dettagliata delle varie fasi menzionate.

3.1 Preprocessazione del volume

L'unico trattamento preventivo che può risultare necessario è l'adattamento del volume al formato imposto dall'hardware grafico.

Il primo vincolo è relativo alla risoluzione delle texture. Sebbene possano essere definite texutre di dimensioni arbitrarie (estensione EXT_texture_rectangle) ciò non è supportato da tutte schede grafiche ed è inoltre soggetto a limitazioni di vario tipo (mancanza dei bordi, limitazione alle texture bidimensionali ecc). Il formato generale è invece quello delle potenze di 2, distintamente per ciascuna dimensione.

Altri vincoli sulle dimensioni delle textures sono riportati, nell'appendice A. In generale, al fine di soddisfare i vincoli di tempo, potrà essere necessario eseguire un ridimensionamento del volume in base alle tabelle riportate nella stessa sezione. Anche la coppia dimensioni-finestra-di-vista/risoluzione-schermo dovrà essere "calibrata" sui vincoli di tempo e sulla risoluzione del volume. A questo proposito si noti che il rapporto ottimale fra la risoluzione del volume e quella della finestra di vista, definiti il punto di vista medio ed il tipo di proiezione, è quello che mediamente proietta un voxel su un pixel dello schermo. Visualizzare il volume su una finestra con risoluzione troppo alta (l'area della proiezione del voxel ricopre più di un pixel) non aggiunge informazione all'immagine ma appesantisce notevolmente il processo di *filling*; viceversa una risoluzione del volume troppo alta rispetto a quella della finestra implica un inutile

sovraccarico del canale di comunicazione e della memoria video senza un apprezzabile incremento della qualità.

Un'altro trattamento preventivo è finalizzato a ridurre al minimo la dimensione del texel. In molti casi, come ad esempio per volumi CT a 12 bit, è possibile ricodificare i dati con soli 8 bit senza una significativa perdita di precisione. Infatti, dato che la distribuzione di densità dei tessuti di interesse è normalmente contenuta entro un subrange della intera gamma CT, tale ricodifica si riduce sostanzialmente ad una traslazione con, eventualmente, una scalatura molto modesta.

Algorithm 1 Campionamento mediante texture mapping

```
// definisce la slice per l'unità di texture

  glActiveTextureARB( GL_TEXTURE0_ARB);
    glBindTexture (GL_TEXTURE_2D, tex(z) );

  glActiveTextureARB( GL_TEXTURE1_ARB);
    glBindTexture (GL_TEXTURE_2D, tex(z) );

  glActiveTextureARB( GL_TEXTURE3_ARB);
    glBindTexture (GL_TEXTURE_2D, tex(z) );

  glActiveTextureARB( GL_TEXTURE3_ARB);
    glBindTexture (GL_TEXTURE_2D, tex(z+dz) );


// Posiziona le slices sul poligono con gli offsets corretti.
// L'hardware fa il resto

  glMultiTexCoord2fARB(GL_TEXTURE0_ARB, smin, tmin);
  glMultiTexCoord2fARB(GL_TEXTURE1_ARB, smin+ds, tmin);
  glMultiTexCoord2fARB(GL_TEXTURE2_ARB, smin, tmin+dt);
  glMultiTexCoord2fARB(GL_TEXTURE3_ARB, smin, tmin);
    glVertex3f( xmin, ymin, zpos );

  glMultiTexCoord2fARB(GL_TEXTURE0_ARB, smax, tmin);
  glMultiTexCoord2fARB(GL_TEXTURE1_ARB, smax+ds , tmin);
  glMultiTexCoord2fARB(GL_TEXTURE2_ARB, smax, tmin+dt);
  glMultiTexCoord2fARB(GL_TEXTURE3_ARB, smax, tmin);
    glVertex3f( xmax, ymin, zpos);

  glMultiTexCoord2fARB(GL_TEXTURE0_ARB, smax, tmax);
  glMultiTexCoord2fARB(GL_TEXTURE1_ARB, smax+ds, tmax);
  glMultiTexCoord2fARB(GL_TEXTURE2_ARB, smax, tmax+dt);
  glMultiTexCoord2fARB(GL_TEXTURE3_ARB, smax, tmax);
    glVertex3f( xmax, ymax, zpos);

  glMultiTexCoord2fARB(GL_TEXTURE0_ARB, smin, tmax);
  glMultiTexCoord2fARB(GL_TEXTURE1_ARB, smin+ds, tmax);
  glMultiTexCoord2fARB(GL_TEXTURE2_ARB, smin, tmax+dt);
  glMultiTexCoord2fARB(GL_TEXTURE3_ARB, smin, tmax);
    glVertex3f( xmin, ymax, zpos);
```

3.2 Campionamento mediante texture mapping

Dopo aver caricato nella RAM video una texture 2D per ogni slice del volume, la ricostruzione dell'intero volume avviene "agganciando" ogni texture ad un poligono di supporto avente dimensioni pari alla dimensione fisica della slice e distanziato dai poligoni adiacenti della stessa distanza di acquisizione delle slice. Il numero di campionamenti per ogni raggio di vista è dunque pari al numero di slice del volume. L'hardware di rasterizzazione determina, per ciascun pixel ricoperto dalla proiezione del poligono sulla finestra di vista, il punto di campionamento sulla texture e l'hardware di texturing ne estrae, per interpolazione lineare

(bi o trilineare rispettivamente per texture 2D o 3D) dei texel viciniori, il relativo valore di densità; tramite la *funzione di trasferimento* prescelta, il valore campionato viene quindi mappato su una quaterna RGBA.

Per disporre dei tre valori di densità necessari, insieme al valore "centrale", a determinarne il gradiente, si abilitano tre unità di texture "ausiliarie" che saranno utilizzate per campionare il volume con offset dx, dy, dz, prefissati rispetto al punto "centrale" (Algoritmo 1). Le prime due unità utilizzeranno la stessa texture utilizzata dall'unità principale, la terza utilizzerà la texture successiva secondo l'ordine di accumulazione stabilito.

3.3 Funzione di trasferimento

Il mapping densita→colore-opacità è eseguito sfruttando la funzionalità "*glColorTable*" implementata in hardware dalla scheda grafica. Utilizzando la funzione di trasferimento definita in [3] si compila una tavola di look-up di 256 valori e la si installa nella memoria della scheda grafica. Contesualmente al campionamento l'hardware di texturing esegue la conversione dal valore campionato nella corrispondente quadrupla RGBA contenuta nella tavola. Nella tavola sono memorizzati *colori associati* invece che colori puri (vedi [10, 2]) per controllare l'errore d'interpolazione del colore. L'uso dei colori associati riduce inoltre, a beneficio dei passi successivi, la complessità della funzione di accumulazione del colore (cfr. 2 e 4).

L'uso della tavola di look-up consente inoltre di fornire all'utente la funzionalità di calibrazione della funzione di trasferimento in tempo reale. Dato il numero limitato di valori contenuti, la tavola può essere ricalcolata e ricaricata in un tempo trascurabile ogni volta che l'utente apporta delle variazioni ai paramentri della funzione stessa.

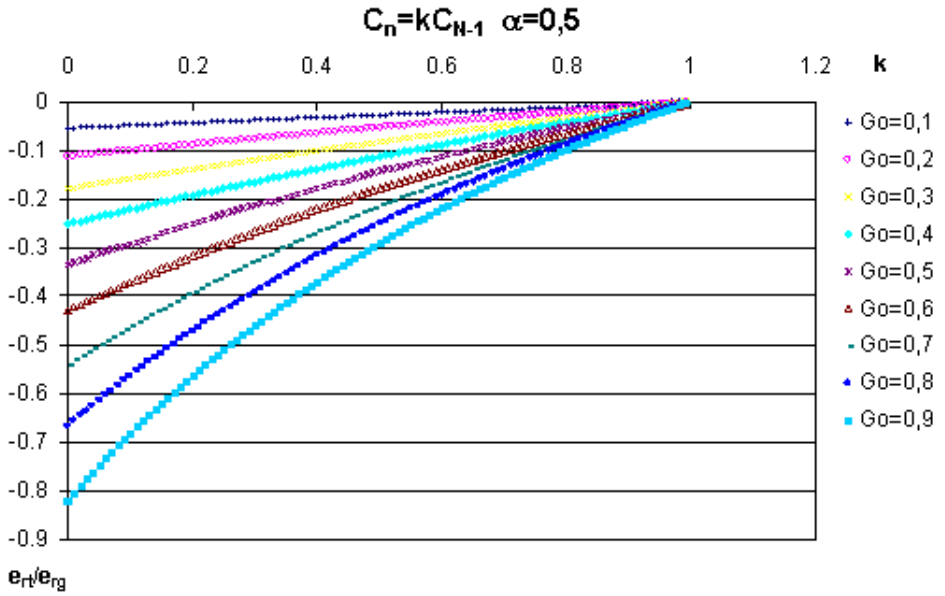


Figura 2: Andamento del rapporto $\varepsilon_{rt}/\varepsilon_{rg}$ al variare di $k = C[n]/\widetilde{C}_{N-1}$ per diversi valori di G_o e $\alpha = 0,5$.

3.4 Calcolo del gradiente di opacità

Nella equazione di illuminazione (12) la normale nel punto di campionamento viene desunta dal gradiente $\nabla f[n]$, il cui modulo è anche interpretato come *strength* della superficie. Utilizzando come $f[n]$ l'opacità, invece che la densità, si ha il vantaggio di poter modificare a piacere e in tempo reale l'apparenza delle superfici (opacità, spessore e consistenza) modificando opportunamente la funzione di trasferimento.

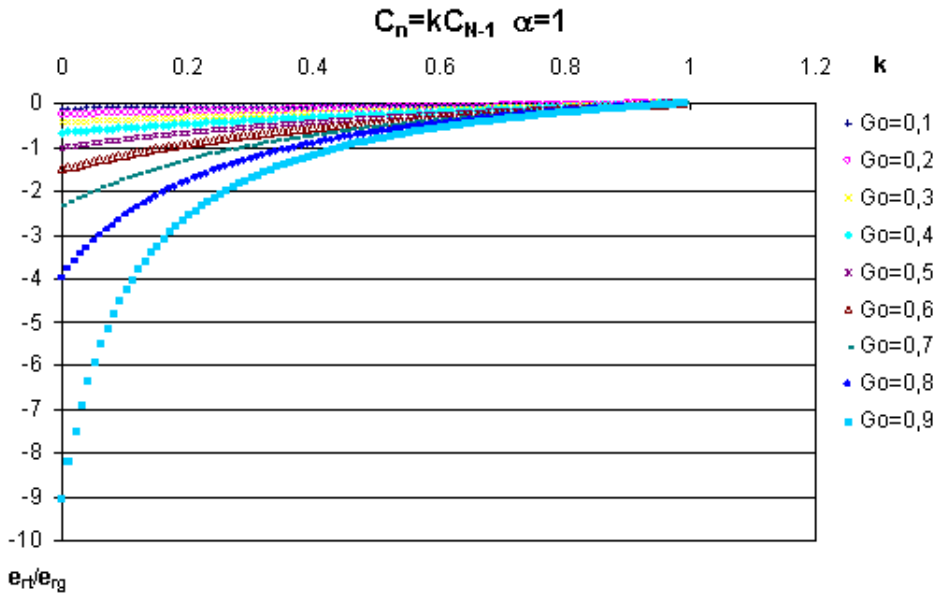


Figura 3: Andamento del rapporto $\varepsilon_{rt}/\varepsilon_{rg}$ al variare di $k = C[n]/\widetilde{C}_{N-1}$ per diversi valori di G_o e $\alpha = 1$.

Ricevendo dall'hardware di texturing i 4 valori di opacità $\alpha(p)$, $\alpha(p+dx)$, $\alpha(p+dy)$, $\alpha(p+dz)$, i combiner possono approssimare il gradiente di opacità con le differenze in avanti:

$$\begin{aligned} \nabla_x &= \frac{\alpha(p) - \alpha(p+dx)}{dx} \\ \nabla_y &= \frac{\alpha(p) - \alpha(p+dy)}{dy} \\ \nabla_z &= \frac{\alpha(p) - \alpha(p+dz)}{dz}. \end{aligned}$$

I combiner sono organi aritmetici SIMD in grado di eseguire delle combinazioni lineari (somme e prodotti) delle quadruple presenti nei registri d'ingresso, con canali differenziati per le terne di colore e i valori di opacità. Nei combiner è quindi agevole calcolare le differenze in avanti e il quadrato del modulo del gradiente mentre è praticamente impossibile eseguire operazioni di divisione (es. normalizzazione dei vettori) e di estrazione della radice quadrata. La limitazione della divisione non pone particolari problemi: l'utilizzazione del modulo di gradiente come misura dello strength delle superfici elimina infatti la necessità di normalizzazione del gradiente stesso (unico caso di divisione per una variabile), mentre la divisione per valori costanti viene aggirata moltiplicando per il reciproco delle stesse, precalcolato via software e memorizzato in opportuni registri d'ingresso. Più delicata è invece la soluzione del problema di estrazione della radice per la determinazione della norma del gradiente. Questa deve essere approssimata mediante un polinomio, il cui errore di approssimazione dipende sia dall'ordine che dai coefficienti del polinomio stesso. Dato che il numero di combiner in cascata è esiguo e la maggior parte degli stadi sono impegnati per il calcolo di altre grandezze (componenti del gradiente, equazione di illuminazione ecc.), non è possibile andare oltre il secondo ordine. C'è invece ampia scelta per quanto riguarda il valore dei coefficienti. Ad esempio il polinomio può discendere dallo sviluppo in serie di Taylor della radice, arrestato alla derivata seconda e valutato nell'intorno di un punto x_0 qualunque dell'intervallo]0, 1], nel quale, avendo assunto valori unitari degli offset, cade il gradiente di opacità. La scelta di x_0 determina l'andamento dell'errore di approssimazione nell'intervallo e deve essere scelto accuratamente. A questo proposito si consideri che la norma del gradiente viene utilizzata per pesare l'opacità del voxel e che questa determina l'incidenza del

colore dello stesso sul colore accumulato complessivamente secondo (cfr. 6) la:

$$\widetilde{C}'_N = \alpha'[n]C[n] + (1 - \alpha'[n])\widetilde{C}_{N-1} \tag{13}$$

con:

$$\alpha'[n] = \|\nabla\alpha[n]\|\,\alpha[n]. \tag{14}$$

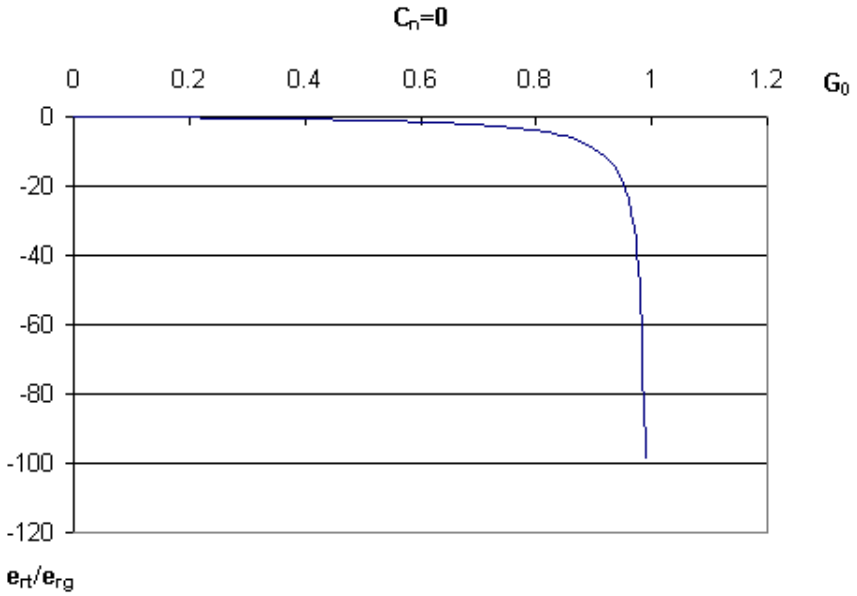


Figura 4: Andamento del rapporto $\varepsilon_{rt}/\varepsilon_{rg}$ al variare di G_o per $\alpha[n] = 1$ e $C[n] = 0$.

Valutiamo l'andamento dell'errore sulla (13) al variare della approssimazione di $\|\nabla\alpha[n]\|$, di cui indicheremo in seguito, per brevità, con G_o il valore esatto e con G'_o il valore approssimato.

Indicando con:

$$\varepsilon_{rg} = \frac{G'_o - G_o}{G'_o}$$

$$\varepsilon_{rt} = \frac{\widetilde{C}'_N - \widetilde{C}_N}{\widetilde{C}_N}$$

rispettivamente l'errore relativo di approssimazione su G_o ed il conseguente errore relativo su $\widetilde{C}_N$ (di cui $\widetilde{C}'_N$ è il valore approssimato) possiamo scrivere:

$$\begin{aligned} \widetilde{C}'_N &= \widetilde{C}_N\,(1 + \varepsilon_{rt}) \\ &= \left[G_o\alpha[n]C[n] + (1 - G_o\alpha[n])\,\widetilde{C}_{N-1}\right](1 + \varepsilon_{rt}) \end{aligned} \tag{15}$$

oppure anche:

$$\begin{aligned} \widetilde{C}'_N &= G'_o\alpha[n]C[n] + (1 - G'_o\alpha[n])\,\widetilde{C}_{N-1} \\ &= G_o\,(1 + \varepsilon_{rg})\,\alpha[n]C[n] + [1 - G_o\,(1 + \varepsilon_{rg})\,\alpha[n]]\,\widetilde{C}_{N-1} \end{aligned} \tag{16}$$

eguagliando le quali si può ricavare il rapporto fra i due errori relativi:

$$\frac{\varepsilon_{rt}}{\varepsilon_{rg}} = \frac{G_o\alpha[n]\left(C[n] - \widetilde{C}_{N-1}\right)}{G_o\alpha[n]\left(C[n] - \widetilde{C}_{N-1}\right) + \widetilde{C}_{N-1}}. \tag{17}$$

La (17) mostra la dipendenza dell'errore di colore dai valori di G_o, $\alpha[n]$ e dalla differenza ($C[n] - \widetilde{C}_{N-1}$). In particolare si può notare che l'errore tende ad annullarsi se (con $\widetilde{C}_{N-1}$ non nullo) è nullo il contributo di opacità ($G_o = 0$ o $\alpha[n] = 0$) della texture corrente o se il contributo di colore della stessa tende al colore già accumulato nelle iterazioni precedenti. Al contrario, il rapporto d'errore diviene massimo quando $G_o = \alpha[n] = 1$ e tende ad ∞ al tendere a 0 di $C[n]$.

L'andamento del rapporto d'errore è stato diagrammato nelle figure 2 e 3 per diversi valori di G_o e $\alpha[n]$ e per $0 \leq C[n] \leq \widetilde{C}_{N-1}$, esprimendo la differenza ($C[n] - \widetilde{C}_{N-1}$) in funzione del rapporto $k = C[n]/\widetilde{C}_{N-1}$, con $0 \leq k \leq 1$.

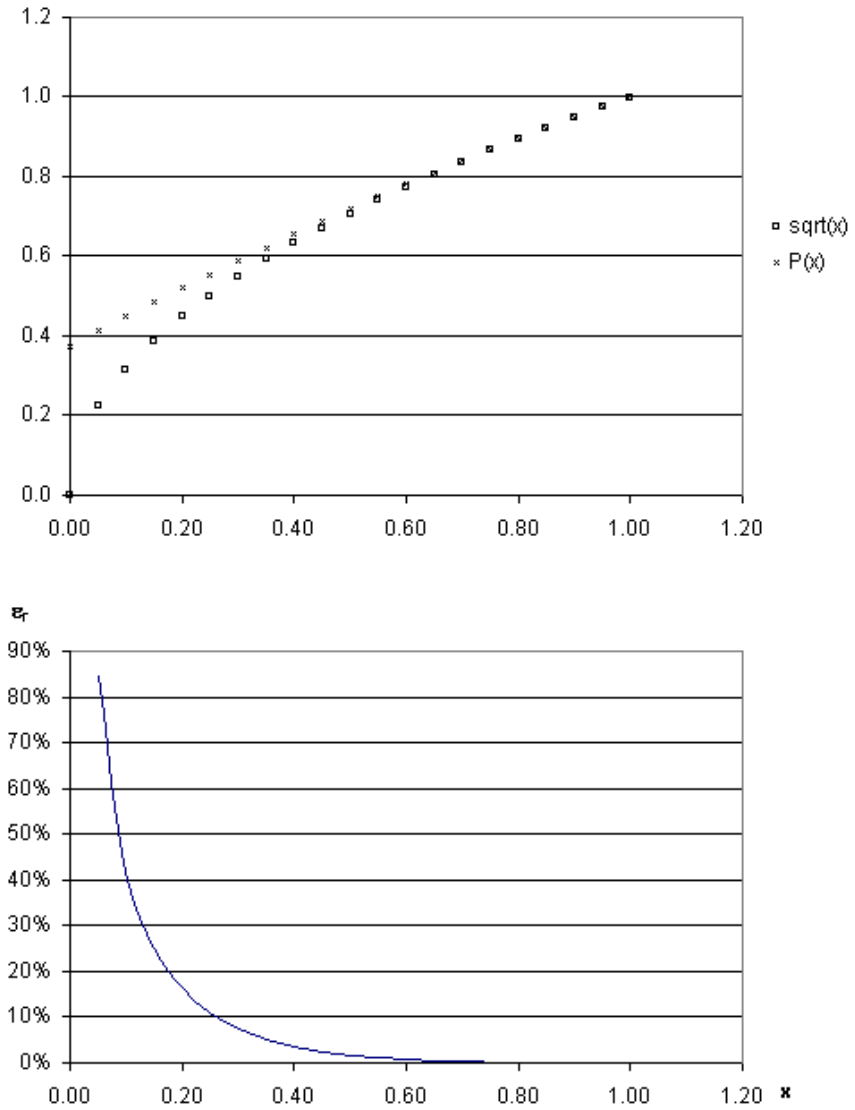


Figura 5: Andamento del polinomio approssimante $P(x)$ rispetto alla funzione $\sqrt{x}$ e andamento dell'errore di approssimazione.

Si può notare l'andamento monotono delle curve d'errore, il cui valore assoluto presenta sempre un

massimo per $k = 0$, e la proporzionalità diretta fra l'errore di approssimazione, il modulo del gradiente di opacità e l'opacità stessa.

Valutandola nelle condizioni più sfavorevoli di $\alpha[n] = 1$ e $C[n] = 0$ la (17) diviene:

$$\frac{\varepsilon_{rt}}{\varepsilon_{rg}} \quad = \quad f(G_o) = \frac{G_o}{G_o - 1} \tag{18}$$

che mostra l'influenza sull'errore di colore del termine G_o, approssimato nei combiner (fig 4).

La migliore approssimazione per il colore risultante si ha quindi se il polinomio che approssima $\sqrt{x}$ è scelto in modo da restituire il valore esatto in $x = 1$ ($G_o = 1$).
Imponendo ai coefficienti del polinomio di eguagliare in $x = 1$ sia la funzione che le prime due derivate si ottiene il polinomio interpolante:

$$\sqrt{x} \approx \frac{3}{8} + \frac{3}{4}x - \frac{1}{8}x^2,$$

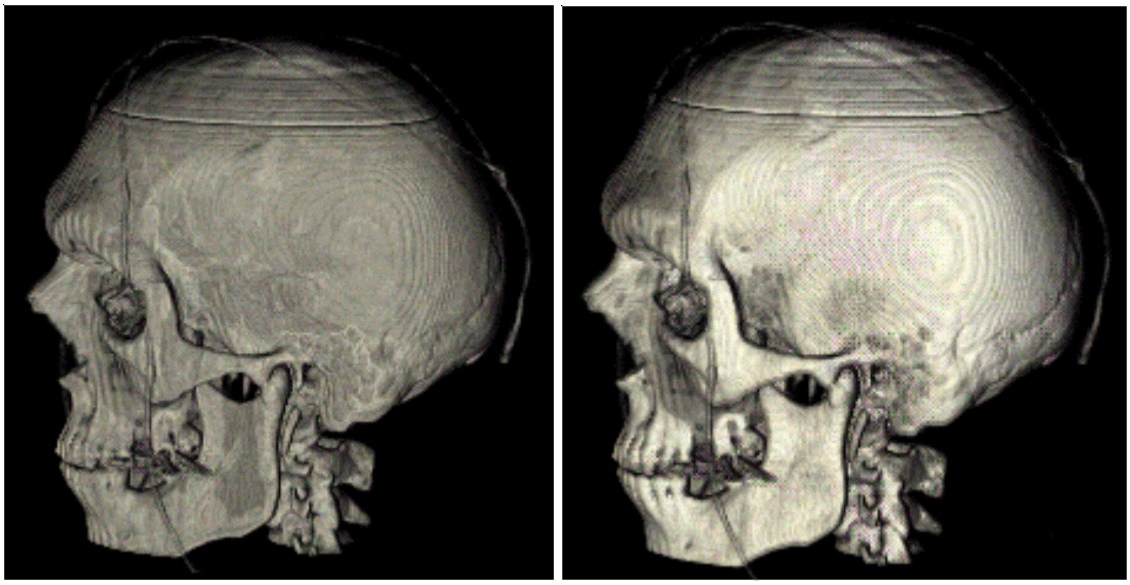
il cui andamento è diagrammato in figura 5 rispetto alla funzione approssimata. La stessa figura mostra infine l'errore relativo di approssimazione.

3.5 Calcolo del colore del voxel

In accordo con la equazione (12) l'illuminazione del volume è stata supposta costituita dall'insieme di una componente ambiente e di una componente direzionale emessa da una sorgente posta all'∞ sull'asse *z* del volume (normale alle slices). La particolare direzione di illuminazione è tale che il suo prodotto scalare per il gradiente di opacità dipenda esclusivamente dalla componente $\nabla_z f[n]$. Questo consente di semplificare la (12) come segue:

$$\widetilde{C}[n] = l_a \left\| \nabla f[n] \right\| \alpha[n] C_a[n] + l_d \alpha[n] C_d[n] max\left(\nabla_z f[n], 0\right), \tag{19}$$

risparmiando un prodotto scalare e liberando uno stadio dei combiner.



(a) (b)

Figura 6: Confronto risultati visivi equazioni (19) e (20).

Il risultato visivo prodotto dalla semplice applicazione della (19) non risulta però pienamente soddisfacente. Il suo effetto è infatti quello che solo i voxel di superficie contribuiscano al colore complessivo del

pixel per effetto dell'annullamento dello strength nelle zone interne ai tessuti. Questo sarebbe irrilevante se le superfici avessero una consistenza tale da saturare l'opacità mascherando completamente il colore dei voxel interni. Tuttavia, sia per la presenza di superfici con basso strength (ad es. le superfici di separazione fra tessuti molli), sia per i limiti stessi del processo di acquisizione che, mediando la densità su tutto il volume del voxel, tende a "disgregare" le pareti di spessore esiguo, accade spesso che le superfici siano tutt'altro che opache lasciando intravedere le "cavità" prodotte dalla (19) (fig. 6a).

Per questo motivo il modello ottico adottato per le zone interne è differnte da quello descritto dalla (19) impiegato esclusivamente nelle zone a gradiente non nullo. Il modello complessivo è pertanto definito da:

$$\widetilde{C}[n] = \begin{cases} l_a \left\| \nabla f[n] \right\| \alpha[n] C_a[n] + l_d \alpha[n] C_d[n] max\left(\nabla_z f[n], 0\right) & se \quad \left\| \nabla f[n] \right\| > 0 \\ \left(l_a + l_d\right) \alpha[n] C_a[n] & se \quad \left\| \nabla f[n] \right\| = 0 \end{cases} . \tag{20}$$

In questo modo la qualità della immagine migliora considerevolmente durante la visualizzazione dei tessuti a bassa densità o nel caso di sottili strati di tessuti si elevata densità (es. osso), come mostrato nella figura 6b.

3.6 Accumulazione di colore e opacità

Le equazioni di accumulazione (6 e 7), possono essere implementate efficacemente su tutte le piattaforme grafiche OpenGL.

Ciò può essere fatto abilitando la funzione di alpha blending (GL_BLEND) e impostando i fattori di blending, *source* e *destination*, a (ONE, ONE_MINUS_SRC_ALPHA) per la composizione back-to-front e (ONE_MINUS_DST_ALPHA, ONE) per la composizione front-to-back.

Infatti, al momento della scrittura sul framebuffer, con l'alpha blending abilitato, il valore del fragment viene "miscelato" col precedente colore del pixel secondo la:

$$C_{d_{new}} = C_s F_s + C_d F_d \tag{21}$$

dove C_s (source color) è la generica componente di colore del fragment, C_d (destination color) l'omonima componente di colore del pixel, F_s e F_d rispettivamente i fattori *source* e *destination* precedentemente definiti. Per le due impostazioni citate e per ciascuna componente di colore, la (21) diviene:

$$back - to - front : \begin{cases} R_{d_{new}} = R_s + R_d \left(1 - \alpha_s\right) \\ G_{d_{new}} = G_s + G_d \left(1 - \alpha_s\right) \\ B_{d_{new}} = B_s + B_d \left(1 - \alpha_s\right) \\ \alpha_{d_{new}} = \alpha_s + \alpha_d \left(1 - \alpha_s\right) \end{cases}$$

$$front - to - back : \begin{cases} R_{d_{new}} = R_s \left(1 - \alpha_d\right) + R_d \\ G_{d_{new}} = G_s \left(1 - \alpha_d\right) + G_d \\ B_{d_{new}} = B_s \left(1 - \alpha_d\right) + B_d \\ \alpha_{d_{new}} = \alpha_s \left(1 - \alpha_d\right) + \alpha_d \end{cases} .$$

In particolare, per l'opacità risultante, tenendo conto che all'n-esima iterazione il framebuffer contiene l'opacità eccumulata nelle precedenti *N-1* iterazioni, si ha:

$$\begin{aligned} \alpha_N &= \alpha_n + \alpha_{N-1} \left(1 - \alpha_n\right) \\ &= 1 - 1 + \alpha_n + \alpha_{N-1} - \alpha_{N-1} \alpha_n \\ &= 1 - \left(1 - \alpha_n\right) \left(1 - \alpha_{N-1}\right) \\ &= 1 - T_N \end{aligned} \tag{22}$$

per l'accumulazione back-to-front e:

$$\begin{aligned} \alpha_N &= \alpha_n (1 - \alpha_{N-1}) + \alpha_{N-1} \\ &= 1 - 1 + \alpha_n - \alpha_n \alpha_{N-1} + \alpha_{N-1} \\ &= 1 - (1 - \alpha_n)(1 - \alpha_{N-1}) \\ &= 1 - T_N \end{aligned} \tag{23}$$

per l'accumulazione front-to-back, in accordo con la (3).

È importante osservare che le (22) e (23) hanno valore solo nel caso in cui siano stati impiegati colori associati. È anche possibile, impostando i fattori di blending a (SRC_ALPHA, ONE_MINUS_SRC_ALPHA), eseguire l'accumulazione back-to-front con colori puri, in quanto la terna $(R_{d_{new}}, G_{d_{new}}, B_{d_{new}})$ non dipende dal valore di opacità accumulato nel framebuffer α_d, che, in questo caso, non può essere cacolato conformemente alla (3).

3.7 Accelerazione mediante texture zoom

Come si può evincere dai dati riportati nell'appendice A, il fill-rate è un importante collo di bottiglia delle architetture grafiche, sopratutto quando si esegue la visualizzazione su finestre molto grandi e con un elevato numero di slice. Il limite è accentuato, sulle schede NVIDIA, dal massiccio uso dei combiner che può portare una riduzione del frame rate a valori inaccettabili. Il problema può essere superato con un algoritmo multipass che consiste nel visualizzare il volume su una porzione ridotta della finestra e poi eseguire lo zoom della immagine ridotta sulla intera finestra. Grazie alla efficienza dell'hardware grafico nell'eseguire il trasferimento di dati dal framebuffer alla memoria di texture (*glCopyTexSubImage2D*), è possibile implementare lo zoom a costi trascurabili copiando l'immagine in una texture 2D e visualizzando poi quest'ultima su un poligono che ricopre l'intera finestra di vista. Oltre alla elevata efficienza, questa tecnica presenta l'ulteriore vantaggio che l'immagine finale è generata dall'hardware di texturing per interpolazione bilineare, con un evidente beneficio sulla qualità della stessa.

4 Implementazione e risultati

Abbiamo realizzato due versioni (desktop e workbench) del visualizzatore che sono state testate su un personal computer così configurato:

- 2 processori P3 600 MHz, cache 512 KB
- 256 MB di memoria RAM PC100
- scheda grafica NVIDIA GeForce 4 Ti 4600, 128 MB, AGP 2x
- sistema operativo Linux Red Hat 7.2 (kernel 2.4.18)
- un dispositivo aptico Phantom 1.0 per il sezionamento del volume

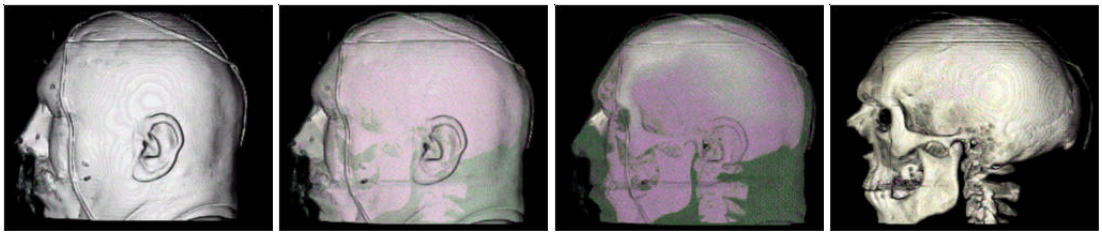


Figura 7: Sequenza di manipolazione dell'opacità dei tessuti di un dataset medico.

Il volume visualizzato è un estratto della scansione CT del maschio del progetto *Visible Human*, delle dimensioni di 256x256x205 voxel e 8 bit di codifica, per un totale di ˜13 MB di occupazione di memoria. Per entrambe le versioni sono stati visualizzati due ricampionamenti del volume di 128 e 256 slice rispettivamente, in finestra da 640x480 impiegando la tecnica di texture zoom (immagine ridotta: 320x240), entrambi precaricati in memoria di texture per eliminare il collo di bottiglia della AGP 2x (vedi valori del *transfer rate* in Appendice A).

All'indirizzo web http://www.crs4.it/vic/multimedia/ sono disponibili i filmati delle sessioni di lavoro sulle due versioni del visualizzatore.

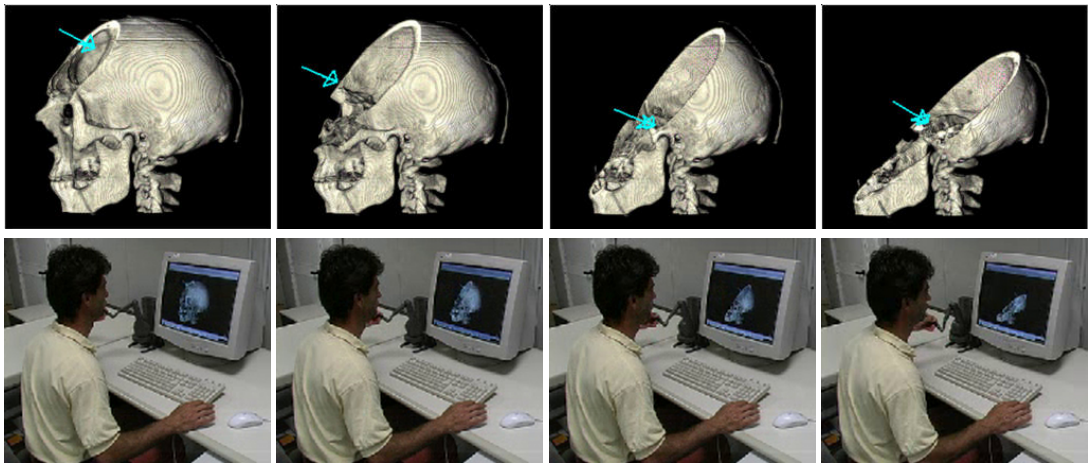


Figura 8: Sequenza di sezionamento di un dataset medico.

4.1 Applicazione desktop

L'applicazione desktop consente la rapida ispezione e manipolazione del volume su una comune postazione di lavoro. L'utente controlla la posizione del piano di taglio agendo sul braccetto del device aptico con la mano sinistra e abilita il sezionamento mediante lo switch posizionato sul braccetto stesso. L'orientazione del volume e la classificazione dei tessuti vengono invece controllati con la mano destra tramite il mouse, agendo, rispettivamente, direttamente sul volume e su una apposita interfaccia grafica di classificazione (vedi fig. 1a).

Due tipiche sequenze della sessione di lavoro sono visibili nelle figg. 7 e 8. La fig. 7 mostra la progressiva riduzione della opacità assegnata ai tessuti molli, fino alla completa esposizione dell'osso, mentre la fig. 8 mostra il progressivo sezionamento della scatola cranica medinate l'uso del Phantom.

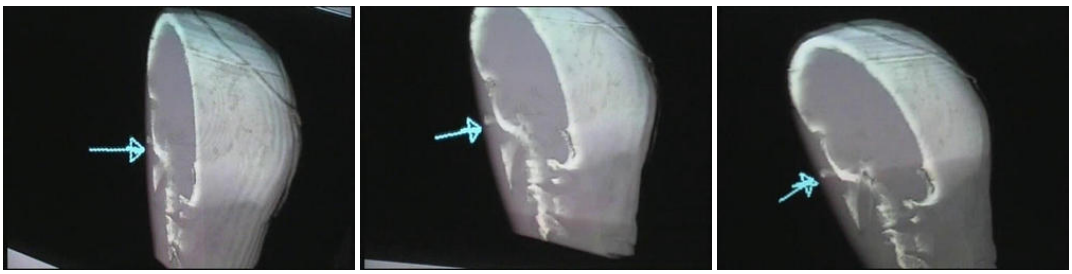
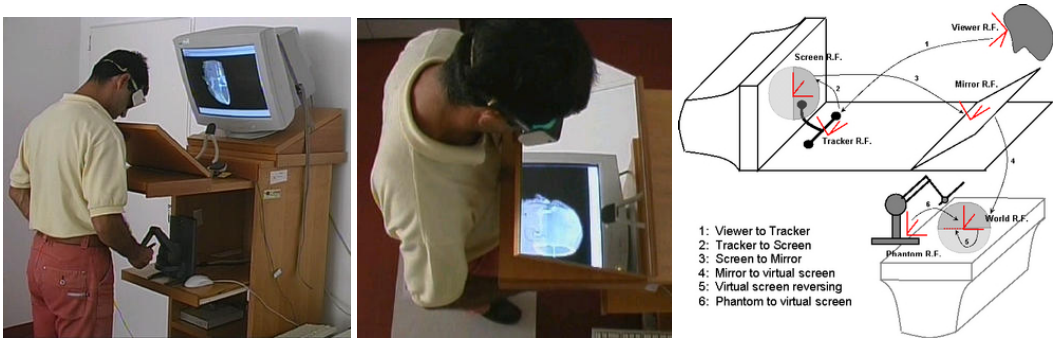


Figura 9: Adattamento della vista prospettica al punto di vista corrente.

4.2 Applicazione workbench

L'applicazione workbench fornisce l'opportunità di osservare e manipolare il volume di dati come se fosse fisicamente appoggiato su un tavolo di lavoro davanti all'utente. Il realismo della interazione è raggiunto sia attraverso l'impiego di un sistema di head tracking a 6 gradi di libertà che attraverso la collimazione dello spazio virtuale (sistema di riferimento di visualizzazione) con lo spazio fisico (sistema di riferimento del Phantom). In questo modo, aggiornando costantemente la proiezione prospettica alla effettiva posizione di vista dell'utente (fig. 9) e facendo coincidere la posizione visiva del cursore di sezionamento con quella fisica del braccetto del Phantom, (fig. 10b) il sistema fornisce l'impressione di operare su un oggetto posizionato in una ben precisa posizione fisica, agendo con la mano direttamente sul cursore di sezionamento. La collimazione degli spazi fisico e virtuale è ottenuta impiegando la postazione di lavoro visibile nella fig. 10a, che assicura una corretta collocazione spaziale delle varie periferiche (monitor, Phantom, tracker), e attraverso l'opportuna composizione, via software, delle relative matrici di trasformazione, schematizzate nella figura 10c.



(a) (b) (c)

Figura 10: Applicazione workbench.

4.3 Prestazioni

Nonostante la piattaforma sia costituita soltanto da componenti commerciali di largo consumo, le prestazioni del sistema sono risultate assolutamente adeguate ai vincoli temporali imposti dalla interattività. Per i due volumi di 128 e 256 slice sono stati misurati rispettivamente 30 e 15 fps circa per il frame rate. Nel caso della versione desktop i tempi di latenza sono contenuti entro il tempo di generazione del fotogramma (˜33 ms e ˜67 ms rispettivamente), mentre per la versione workbench sono stati riscontrati tempi di latenza superiori di circa 20 ms. L'incremento di latenza è dovuto essenzialmente alla presenza del device di tracciamento che richiede appunto tale intervallo di tempo per eseguire la rilevazione.

5 Conclusioni e lavoro futuro

Abbiamo descritto e implementato una evoluzione della tecnica del texture mapping per la visualizzazione diretta di dati volumentrici che, grazie all'uso dei register combiners, riesce a raggiungere prestazioni adeguate ad applicazioni interattive pur implementando un modello di illuminazione più sofisticato rispetto a quello comunemente impiegato. È significativo osservare che le prestazioni raggiungibili da piattaforme più attuali sono, come risulta dai dati in appendice, di gran lunga superiori a quelle misurate sulla piattaforma di test, prestazioni che fino a non molti anni fa erano raggiungibili solamente da workstation grafiche di costi di almeno 2 ordini di grandezza superiori.

È grazie alla maggior potenza e alla maggiore programmabilità delle schede grafiche di più recente produzione che riteniamo di arrivare ad implementare nel prossimo futuro modelli ottici ancora più generali ed accurati pur raggiungendo prestazioni, in termini di frame rate di tempi di latenza, ancora superiori.

Per incrementare la percezione della profondità, elemento fondamentale durante l'esecuzione di compiti tridimensionali, stiamo sviluppando il sistema per consentire la visione stereoscopica del volume. Parallelamente stiamo lavorando per ridurre la latenza introdotta dal sistema di tracciamento che, a causa del maggior tempo di calcolo richesto dalla generazione di mmagini binoculari, può portare la latenza complessiva a livelli inaccettabili.

L'evoluzione dell'hardware nella gestione delle texture tridimensionali e della sostituzione di loro porzioni fà annoverare fra i nostri obbiettivi una più accurata valutazione del gradiente ed una migliore gestione del volume di dati.

6 Ringraziamenti

Questo lavoro è parzialmente supportato dal progetto MIUR: "Laboratorio Avanzato per la Progettazione e la Simulazione Assistita al Calcolatore".

Riferimenti bibliografici

[1] Robert S. Allison, Laurence R. Harris, Michael Jenkin, Urszula Jasiobedzka, and James E. Zacher. Tolerance of temporal delay in virtual environments. In *Proceedings of the IEEE Virtual Reality 2001 International Conference*, volume 2, pages 247–254, March 2001.

[2] James F. Blinn. Jim Blinn's corner: Compositing. 1. Theory. *IEEE Computer Graphics and Applications*, 14(5):83–87, September 1994.

[3] B. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. In Rosalee Wolfe, editor, *Significant Seminal Papers of Computer Graphics: Pioneering Efforts that shaped the Field*, pages 363–372, N.Y., 1998. ACM Press.

[4] S. Ellis, M. Young, B. D. Adelstein, and S. M. Ehrlich. Discrimination of changes in latency during head movement. In *Proceedings of the Eighth International Conference on Human-Computer Interaction*, volume 2, pages 1129–1136, 1999.

[5] Adelstein D. Bernard Ellis R. Stephen, Young J. Mark and Ehrlich M. Sheryl. Discrimination of changes of latency during voluntary hand movement of virtual objects. In *Proceedings of the 43rd Annual Meeting of the Human Factors and Ergonomics Society*, pages 1182–1186, 1999.

[6] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, May 1988.

[7] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995. ISSN 1077-2626.

[8] Kyoung Shin Park and Robert V. Kenyon. Effects of network characteristics on human performance in a collaborative virtual environment. In *Proceedings of IEEE VR '99*, pages 104–111, 1999.

[9] Paolo Sabella. A rendering algorithm for visualizing 3D scalar fields. volume 22, pages 51–58, August 1988.

[10] Craig M. Wittenbrink, Thomas Malzbender, and Michael E. Goss. Opacity-weighted color interpolation for volume sampling. In *IEEE Symposium on Volume Visualization*, pages 135–142. IEEE, ACM SIGGRAPH, 1998.

A Appendice: Vincoli hardware e prestazioni limite

Nelle applicazioni di realtà virtuale i colli di bottiglia sono localizzati essenzialmente nell'hardware grafico e nel canale di comunicazione fra la scheda madre e la scheda grafica (porta AGP).

I limiti alle prestazioni, o meglio al rapporto fra prestazioni e qualità, del sistema possono essere senz'altro attribuiti, per quanto riguarda l'hardware grafico, ai valori di:

- *fill rate*
- *vertex rate*
- *texel rate*

e, per quanto riguarda il canale di comunicazione, al:

- *transfer rate*

L'importanza del transfer rate è inoltre enfatizzata da altri fattori che concorrono a determinare lo sfruttamento del canale di comunicazione:

- la quantità di memoria video disponibile per le texture
- la dimensione del texel

Assumendo come valori di riferimento quelli dello stato dell'arte dell'hardware grafico per PC, riportati in tabella 1, diamo di seguito una valutazione delle prestazioni teoriche raggiungibili relativamente a ciascuno dei fattori vincolanti precedentemente citati.

Tipo hardware	Transfer rate	Pixel rate	Texel rate	Vertex rate	Memoria max
GeForce4 Ti 4800	2,1 GB/s	1.200 M/s	4.800 M/s	136 M/s	128 MB
ATI Radeon 9800 Pro	2,1 GB/s	3050 M/s	18.200 M/s	380 M/s	256 MB
GeForce FX 5800 Ultra	2,1 GB/s	4.000 M/s	16.000 M/s	350 M/s	256 MB

Tabella 1: Prestazioni teoriche hardware grafico

A.1 Fill Rate

Definisce il numero di pixel dello schermo che il sistema riesce a "riempire" nell'unità di tempo.

La tabella 2 riporta le prestazioni, in fotogrammi al secondo, raggiungibili teoricamente con la proiezione parallela di un dataset con punto di vista assiale (totale sovrapposizione delle proiezioni delle slices), riferite alle seguenti due situazioni di riferimento:

1. *rapporto voxels/pixel ottimale di 1:1* :
   il tempo di frame dipende dalla risoluzione del volume;
2. *finestra di vista interamente ricoperta da tutte le slice del volume* :
   Il tempo di frame è indipendente dalla risoluzione della slice e varia linearmente col numero di slice e con l'area della finestra

È importante sottolineare che le prestazioni indicate hanno validità solo per una processazione *standard* del fragment, ovvero se non vengono utilizzati i combiners per particolari elaborazioni dello stesso. L'abilitazione dei combiners comporta invece un incremento del tempo di visualizzazione pari al tempo base per ogni coppia di combiner abilitati oltre la prima. In altri termini, mentre l'utilizzazione di due combiner non comporta una riduzione delle prestazioni, le stesse si riducono alla metà abilitando quattro combiner e ad un quarto abilitandone otto.

Rapporto voxel/pixel 1:1			
Scheda Grafica	**Risoluzione Volume**		
	128^3	256^3	512^3
GeForce4 Ti 4800	600	75	9
ATI Radeon 9800 Pro	1525	191	24
GeForce FX 5800 Ultra	2000	250	31

Copertura totale finestra 320×240			
Scheda Grafica	**N. di Slice**		
	128	256	512
ATI GeForce4 Ti 4800	128	64	32
ATI Radeon 9800 Pro	325	163	81
GeForce FX 5800 Ultra	427	214	107

Copertura totale finestra 640×480			
Scheda Grafica	**N. di Slice**		
	128	256	512
GeForce4 Ti 4800	32.0	16.0	8.0
ATI Radeon 9800 Pro	81.3	40.8	20.3
GeForce FX 5800 Ultra	106.8	53.5	26.8

Copertura totale finestra 1280×1024			
Scheda Grafica	**N. di Slice**		
	128	256	512
GeForce4 Ti 4800	7.50	3.75	1.88
ATI Radeon 9800 Pro	19.0	9.55	4.75
GeForce FX 5800 Ultra	25.0	12.5	6.27

Tabella 2: Limiti imposti al frame rate dal fill rate

A.2 Texel rate

Definisce il numero di texel che possono essere prodotti per unità di tempo.

Se l'hardware supporta il multitexturing questo dato è un multiplo intero del fill rate. Il valore del fill rate non è di norma aumentato da un elevato texel rate perché le texture multiple concorrono alla determinazione del colore di uno stesso pixel.

Per il rendering volumentrico è comunque possibile utilizzare il multitexturing per ottenere importanti benefici in termini di frame rate, anche se a scapito della qualità. Applicando opportunamente le texture al poligono e sfruttando opportunamente i combiner per accumulare i colori dei texel, si può pensare di "proiettare" 4 slice per volta incrementando di 4 volte le prestazioni riportate in tabella 2.

A.3 Vertex rate

Definisce il massimo numero di vertici che possono essere elaborati nell'unità di tempo.

Il vertex rate non rapresenta normalmente un vincolo nel caso di rendering volumetrico diretto per il quale il numero di primitive geometriche è assolutamente trascurabile rispetto alle possibilità dell'hardware grafico.

Valori Teorici (AGP 8x)			
Scheda Grafica	**Risoluzione Volume**		
	128^3	256^3	512^3
GeForce4 Ti 4800 Ultra	1024	128	16
ATI Radeon 9800 Pro	1024	128	16
GeForce FX 5800 Ultra	1024	128	16

Valori Rilevati (AGP 2x)			
Scheda Grafica	**Risoluzione Volume**		
	128^3	256^3	512^3
GeForce4 Ti 4800 Ultra	64.0	9.40	1.28

Tabella 3: Limiti imposti al frame rate dal transfer rate

A.4 Transfer rate

Ci si riferisce alla quantità di byte che possono essere scambiati, attraverso la porta AGP, con il chipset della scheda madre.

Il transfer rate influenza fortemente le prestazioni complessive se il tipo di visualizzazione eseguita comporta una massiccia rigenerazione delle informazioni di stato della scheda, mentre può non influire assolutamente sulle prestazioni se le informazioni di rendering sono caricate una volta per tutte nella memoria video e, eventualmente, rigenerate con un frequenza molto bassa. In questo senso l'influenza del transfer rate è strettamente legata anche alla quantità di memoria con cui la scheda è equipaggiata, che determina quanta parte delle informazioni può essere mantenuta a bordo della scheda.

Considerando la condizione di lavoro più gravosa, ovvero ipotizzando la completa rigenerazione dei dati volumetrici ad ogni frame, e assumendo il massimo sfruttamento del canale AGP si ricavano i valori massimi di frame rate riportati in tabella 3, relativi ad una AGP 8x. Nella stessa tabella sono riportati anche i valori reali rilevati sulla piattaforma di test del visualizzatore, che dispone di una AGP 2x. È evidente la differenza fra questi e i precedenti a conferma del fatto che numerosissimi fattori, quali il formato dei dati, la dimensione dei blocchi di trasferimento ecc., intervengono a limitare l'effettiva banda passante.

A.5 Considerazioni

Se si esclude l'utilizzo di tecniche di compressione delle texture che comportano inevitabilmente un ulteriore degrado della qualità della riproduzione, le prestazioni reali riportate in tabella 3, ottenute con dati volumetrici scalari codificati con la minima precisione (8 bit), debbono considerarsi le massime raggiungibili. I nostri benchmark hanno infatti indicato che nel caso di codifiche a 16 bit o con colori RGBA di 8 bit/componente si ha un degrado delle prestazioni di almeno un fattore 2.

Se l'applicazione non richiede la rigenerazione dei dati volumetrici (applicazioni di manipolazione del volume), l'intero dataset può essere caricato in memoria video una sola volta in fase di inizializzazione dell'applicazione consentendo di eliminare il collo di bottiglia costituito dal canale di comunicazione. Perché ciò sia possibile è però necessario disporre di una quantità di RAM video sufficiente ad ospitare, oltre alle necessarie risorse di rendering, l'intero volume di dati. Se questa condizione non è verificata il sistema grafico manterrà a bordo della scheda solo la porzione di volume di volta in volta processata, utilizzando più o meno intensamente la porta AGP per l'operazione di swap.