

Idee sul futuro di Cassata

Fulvio Satta

27 luglio 2007

Questo documento contiene solo delle idee, e come tali vanno prese, sullo sviluppo del progetto di un renderer unbiased di nome Cassata. Il contenuto di questo documento ha lo scopo di chiarire le idee a chi s'interessa del progetto, e per l'autore per comprendere meglio alcuni punti e non dimenticare nulla man mano si va avanti col progetto. Sebbene sia tutto da prendere come idee chiunque può contattarmi e darmi qualunque genere di consiglio. Se non sapesse come può postare un commento sul [blog](#) del progetto.

1 Cos'è Cassata

Cassata è concepito come renderer unbiased. Tutte le tecniche di base saranno unbiased in tutti i casi in cui è possibile farlo. Tuttavia (come sarà indicato più avanti) potrebbe essere carino mettere anche alcune tecniche bias, a scelta dell'utente.

Cassata verrà studiato attorno al fotorealismo più sfrenato. I compromessi verranno presi solo in situazioni limite, con approssimazioni nel modello da renderizzare che siano davvero piccolissime nelle situazioni normali. Per situazioni limite s'intendono situazioni che una persona probabilmente non vedrà mai nella sua vita, eccetto che al limite in riviste scientifiche o cose simili. Per esempio saranno implementati gli effetti che servono per simulare miraggi, o la diffrazione che causa il colore sul fondo dei CD tramite materiali, ma potrebbero non essere implementate cose come ad esempio un modello di diffrazione completo.

Come detto il fotorealismo è la priorità, ed il renderer sarà imperniato su questo, tuttavia qualche tecnica non fotorealistica potrebbe essere interessante da aggiungere, per gli artisti.

1.1 Cosa non è Cassata

Questa domanda è più difficile.

Innanzitutto Cassata è un renderer, non un simulatore fisico. Può darsi che in futuro qualche simulazione verrà eseguita dal renderer, ed in ogni caso tramite shader qualcos'altro si può fare, ma Cassata non è un simulatore. Le elaborazioni che eventualmente si vorranno fare vanno fatte esternamente, eventualmente tramite shader se si vogliono compiere in parallelo, ma non da Cassata.

Cassata inoltre non è un renderer fatto per essere il più veloce possibile. Lavorerò molto per cercare di renderlo veloce più che potrò, ma la velocità sarà sempre sacrificata in favore della correttezza del rendering.

Inoltre, anche se come detto sarà possibile svolgere numerose operazioni non fotorealistiche il renderer non è pensato per queste operazioni. Saranno possibili molti "trucchetti" che nella realtà non si vedranno mai, ma difficilmente si riusciranno ad ottenere stili pittorici o cose di questo genere. Per questi scopi si consigliano altri renderer scritti appositamente.

2 Caratteristiche implementate nel renderer

2.1 Caratteristiche unbiased

Cassata implementerà un normale ray tracer stocastico, con molte ottimizzazioni. Si potrà definire a piacimento sia il BSDF che l'EDF, potendo così definire qualunque tipo di materiale.

Supporterà i media di partecipazione, il che vuol dire che qualunque effetto dovuto ai media sarà possibile implementarlo.

Inoltre supporterà camere complesse, motion blur e DoF reali, la possibilità di avere materiali che supportino caratteristiche come fluorescenza e fosforescenza, la possibilità di cambiare a piacere l'indice di rifrazione all'interno di un media e via dicendo.

Oltre a questo alcune tecniche di uso comune che richiederebbero (coi metodi sopracitati) troppo tempo verranno ottimizzate tramite tecniche (eventualmente approssimate) fatte appositamente. Un esempio è l'SSS. Notare che queste tecniche rimangono unbiased, pur non essendo fisicamente corrette. Tuttavia l'approssimazione compiuta è generalmente molto bassa, ed in molti casi non notevole ad occhio nudo. Nel caso in cui non fosse così è possibile ricorrere ad altre tecniche più lente, tuttavia è raramente necessario.

2.2 Caratteristiche biased

Saranno implementate caratteristiche biased in 2 casi.

Il primo caso si ha quando non si può avere un effetto in modo unbiased. Fino a questo momento non ho trovato nessuna tecnica utile di questo tipo, tuttavia potrei trovarne. Farò in modo il più possibile che sia usato uno stimatore consistente (che converge all'immagine corretta).

Il secondo caso è sempre e comunque a scelta dell'utente. Si tratta di tecniche a basso impatto visivo, che servono per accelerare il rendering. In pratica si rinuncia a poco qualitativamente per ridurre di molto i tempi di rendering. La scelta che concerne l'usare queste tecniche, come detto prima, dipende sempre dall'utente.

2.3 Caratteristiche non fotorealistiche

Un'idea potrebbe essere quella di interpretare ogni raggio diversamente a seconda di cosa colpisce, magari con un sistema di parametri aggiungibile al raggio. Con questo sistema ogni raggio può essere completamente cambiato a seconda delle superfici che incontra. Ad esempio è possibile rendere un oggetto senza ombre, o solo ombreggiato, o solo riflesso, e via dicendo, ma ovviamente è anche possibile fare tante altre cose. Tanto per dirne una si possono fare 2 specchi, ognuno che riflette oggetti di materiale diverso.

Un'altra idea potrebbe essere quella di fare postprocesso, o processare i raggi della camera, in maniera particolare, eseguendo ad esempio degli shader bidimensionali. Questo potrebbe valutare anche parametri salvati coi raggi stessi, eventualmente personalizzati. Tramite questo è ad esempio possibile fare bordi tipici della grafica toon, ma anche cose più fantasiose, oltre a manipolare l'immagine fotorealistica od aggiungere scritte e via dicendo.

Ovviamente poi è possibile mostrare effetti non fotorealistici cambiando il materiale. Ad esempio sono possibili grafiche di tipo toon, oppure materiali particolari, e via dicendo, in questo modo.

3 Implementazione

3.1 Struttura interna

Come sarà il renderer internamente è ancora un mistero.

3.2 Estensibilità

Il renderer sarà molto estensibile tramite shader. Sicuramente i materiali saranno programmabili, probabilmente anche le camere ed effetti di postprocesso, è ancora incerta la programmabilità delle geometrie.

Bisogna comunque fare attenzione all'equilibrio sulla programmabilità. Molto dev'essere programmabile, ma se gli shader possono fare troppo si lede la possibilità di cambiare internamente il renderer.

4 Futuro

È difficile dire cosa in futuro s'implementerà, tuttavia mi piacerebbe riuscire a rendere il render tanto veloce da poter renderizzare filmati in produzione con risorse non troppo elevate ed offrire la rappresentazione di tutti i fenomeni fisicamente visibili.