Realtime Ray Tracing and Interactive Global Illumination

Ingo Wald

Max-Planck Institut für Informatik, Saarbrücken
wald@mpi-sb.mpg.de

Abstract: Interaktive 3D-Computergraphik basiert heutzutage fast ausschliesslich auf dem Rasterisierungsverfahren. Dieses ist heute sehr effizient und günstig realisierbar, stösst aber zunehmend an seine Grenzen hinsichtlich unterstützter Szenenkomplexität und erreichbarer Darstellungsqualität. Eine Alternative zum Rasterisierungsverfahren ist das Ray Tracing Verfahren, welches zwar allgemein für bessere Bildqualität bekannt ist, aber aufgrund hoher Rechenanforderungen bisher als unvereinbar mit interaktiver Performanz galt. In dieser Arbeit geht es um die Entwicklung der Echtzeit Ray Tracing Technologie, mit der das Ray Tracing Verfahren auch für interaktive Anwendungen ermöglicht wird. Die im Rahmen dieser Dissertation entwickelten Techniken ermöglichen nun erstmalig interaktive Ray Tracing Performanz selbst auf Standard PCs. Die dazu entwickelten Methoden beinhalten unter anderem einen extrem schnellen Ray Tracing Kern, ein effizientes Parallelisierungsframework, die Unterstützung dynamischer und extrem komplexer Modelle, sowie eine geeignete Programmierschnittstelle. Darauf aufbauend wurden dann weiterhin Verfahren entwickelt, die es erstmalig ermöglichen, auch die globale Lichtausbreitung interaktiv zu simulieren. Die im Rahmen der Dissertation entwickelten Techniken formen ein komplettes Framework für Echtzeit-Ray Tracing und interaktive Beleuchtungssimulation, welches interaktive Performanz selbst für massiv komplexe Szenen liefert, sowie erstmalig physikalisch basierte Computergraphik unter Echtzeitbedingungen ermöglicht.

1 Einführung

Computergenerierte Bilder sind heutzutage in fast allen Bereichen sowohl des industriellen als auch des alltäglichen Lebens präsent, z.B. in Form von Computerspielen, computergenerierten und -bearbeiteten Kinofilmen und Animationen, industriell eingesetzter "Virtual-Reality" (VR), oder CAD/CAM/CAE[1] - Anwendungen. Für den praktischen Einsatz spielt insbesondere die *Echtzeit*-Computergraphik eine immer wichtigere Rolle. So wird heute fast jedes Industrieprodukt mittels Techniken der Echtzeit-3D-Graphik virtuell am Computer geplant und entworfen.

Dabei ist ein immer stärkerer Bedarf nach mehr und mehr "Realismus" zu beobachten. Dies betrifft nicht nur Computerspiele, sondern auch den industriellen Bereich: Um qualitative und quantitative Aussagen über ein am Computer entworfenes Objekt machen zu

[1] Computer Aided Design / Computer Aided Manufacturing / Computer Aided Engineering

können, ist es unerlässlich, dass die computergenerierte Darstellung des virtuellen Objekts auch mit dem Aussehen seines realen Pendants übereinstimmt, die Darstellung also "photorealitisch" ist.

Aufgrund der wirtschaftlichen Bedeutung hat sich die 3D-Computergraphik rasant entwickelt, sodass heutzutage sogar schon Desktop-PCs mit extrem leistungsfähigen 3D-Graphikprozessoren ausgestattet sind. Trotz einer Vielzahl von Anbietern (z.B. ATI, NVidia, oder SGI) basieren jedoch *alle* dieser Graphikprozessoren – und damit auch alle darauf aufbauenden Anwendungsprogramme – auf der gleichen Technologie, dem sogenannten Rasterisierungsverfahren [AMH02].

1.1 Das Rasterisierungsverfahren

In diesem Verfahren wird jedes Polygon der darzustellenden 3D-Szene einzeln verarbeitet, beleuchtet, und auf den Bildschirm "gerastert", wobei dahinterliegende Polygone "übermalt" werden. Die Einfachheit dieses Verfahrens ermöglicht eine sehr einfache und extrem effiziente Umsetzung des Verfahrens in Spezial-Hardware, was die Entwicklung der heutzutage sehr preisgünstigen und leistungsfähigen Graphikprozessoren erst ermöglichte.

Allerdings hat dieses Verfahren auch einige gravierende Nachteile: So müssen für jedes Bild alle Polygone neu gerastert werden, was für komplexe Daten selbst auf den modernstens Chips schnell ineffizient wird. Daher können heutzutage zwar typische Spiele-Szenen aus mehreren hunderttausend Polygonen interaktiv dargestellt werden, industrielle CAD-Modelle enthalten aber oft mehrere Millionen bis Milliarden an Dreiecken, welche für Echtzeitanforderungen erst aufwändig und fehlerträchtig vereinfacht werden müssen.

Ein noch gravierender Nachteil der Rasterisierung ist, dass sie für hochqualitative Darstellungen nur bedingt geeignet ist: Da jedes Polygon einzeln und für sich alleine gerastert wird, können indirekte Effekte wie z.B. Schattenwurf oder Reflektion – welche grundlegend auf dem direkten Zusammenspiel *mehrerer* Objekte basieren – nicht direkt berechnet werden. Um diese Limitierung zu umgehen wurden – speziell für Spieleanwendungen – eine Vielzahl an Algorithmen und Verfahren entwickelt, um solche Effekte wirklichkeits*ähnlich* vortäuschen zu können. Dies erfordert allerdings einen hohen manuellen Aufwand von speziell für das jeweilige Spiel bzw. Projekt eingestellten Künstlern und Programmierern, welcher für jede Anwendung von neuem geleistet werden muss.

Desweiteren erheben diese Verfahren keinen Anspruch auf Korrektheit: Typischerweise erzeugen sie keine *simulierten* Bilder eines realen Effekts, sondern nur ein realistisch *wirkendes*, aber dennoch falsches Bild. So kann man z.B. ein Auto mit einem spiegelnd "aussehenden" Lack darstellen, allerdings zeigt die so "berechnete" Spiegelung meist nicht das korrekte Bild der Umgebung, und wird daher nicht mit dem Aussehen des realen Autos übereinstimmen. Somit sind die mit diesem Verfahren generierten Ergebnisse gänzlich ungeeignet, qualitative oder quantitative Aussagen über das reale, zu simulierende Objekt zu treffen, was aber für praktische Anwendungen in der Industrie unerläßlich ist.

1.2 Ray-Tracing und Photorealistische Computergraphik

Eine Alternative zum Rasterisierungsverfahren ist das ca. 20 Jahre alte "Ray-Tracing"-Verfahren (Strahlverfolgungs-Verfahren), welches die physikalischen Gesetze der Lichtausbreitung verwendet, um das in die virtuelle Kamera einfallende Licht naturgetreu zu *simulieren*. Im Gegensatz zum Rasterisierungsverfahren berücksichtigt das Ray-Tracing-Verfahren automatisch *alle* Polygone der Szene, wodurch globale Effekte wie Schatten und Reflektionen *automatisch und exakt* berechnet werden. Insbesondere können damit physikalisch korrekte, "photorealistische" Bilder generiert werden, welche zur verlässlichen Visualisierung und zur *Vorhersage* des optischen Eindrucks eines Produktes verwendet werden können. Darüber hinaus greift das Ray-Tracing automatisch nur auf Polygone zu, die auch wirklich zum jeweiligen Bild beitragen, wodurch es sich speziell für komplexe Geometrien eignet.

Während das Ray-Tracing-Verfahren also einen extrem hohen Grad an Genauigkeit, Flexibilität und Realismus aufweist, so war es bisher leider auch bekannt für seinen notorisch hohen Berechnungsaufwand. Selbst auf schnellen Rechnern waren bisher oft mehrere Minuten bis sogar mehrere Stunden *pro einzelnem Bild* erforderlich. Dadruch bedingt konnten photorealistische Techniken bisher ausschließlich in nicht-interaktiven Anwendungen verwendet werden.

1.3 Photorealismus in Echtzeit

Zusammenfassend kann man sagen, dass rasterisierungsbasierte Echtzeit-Graphik zwar interaktive Anwendungen erlaubt, dafür aber starken Beschränkungen hinsichtlich Szenenkomplexität und erreichbarer Bildqualität unterliegt. Im Gegensatz dazu konnten bisher Ray-Tracing-basierte, photorealistische Techniken zwar quasi beliebig realistische Bilder auch für komplexe Geometrien erzeugen, jedoch im Allgemeinen nur für nicht-interaktive Anwendungen.

Um diese Lücke zu schließen, bieten sich nur zwei Möglichkeiten an – entweder das Ray-Tracing-Verfahren auf Echtzeit-Performanz zu beschleunigen, oder aber das Rasterisierungsverfahren auf photorealistiche Qualität zu erweitern. Dabei wurde bisher allgemein angenommen, dass eine derartige Beschleunigung des Ray-Tracing-Verfahrens grundsätzlich nicht realisierbar wäre, sodass nahezu alle Forschungsanstrengungen zur interaktiven hochqualitativen Computergraphik ausschließlich den Ansatz verfolgten, Photorealismus mittels verbesserter Rasterisierungsverfahren erreichen zu wollen. Obwohl immense Forschungstätigkeit auf diese Fragestellung verwendet wurde, ist man vom endgültigen Ziel – Photorealismus in Echtzeit – auch heute noch weit entfernt.

2 Echtzeit-Ray Tracing (ERT)

Im Gegensatz zum "Standardansatz" der Computergraphik – die Qualität der Rasterisierungsverfahren inkrementell zu verbessern – wurde im Rahmen der eingereichten Dissertation der grundlegend neue, unkonventionelle Ansatz des "Echtzeit-Ray-Tracing" (ERT) verfolgt. Dabei wurden mehrere Teilaspekte berücksichtigt, insbesondere die Entwicklung eines hochoptimierten ERT-Kerns, ein Framework zum parallelen Ray Tracing auf PC

Clustern, eine Methode zur Unterstützung dynamischer Szenen, sowie eine eigens entwickelte Programmierschnittstelle.

2.1 Entwicklung eines Echtzeit Ray Tracing Kerns

Um einen hochgradig optimierten Ray Tracing Kern zu erhalten, müssen insbesondere die Eigenschaften moderner CPUs beachtet werden: So erreichen moderne CPUs zwar extrem hohe Fließkomma-Performanz, allerdings typischerweise nur, wenn diese durch SIMD-Erweiterungen effizient ausgenutzt wird. Desweiteren sind die extrem langen Pipelines moderner Prozessoren sehr anfällig für komplexen Programmcode und schwer vorhersagbare Speicherzugriffsmuster, da beide sehr schnell zu CPU-Leerlaufzeiten durch "branch mispredictions" bzw. Speicherzugriffslatenzen ("cache misses") führen.

Zu diesem Zweck wurde der ERT-Kern speziell so entworfen, dass eine effiziente Verwendung von SIMD-Erweiterungen möglich wurde [WSBW01], indem die Berechnungen so umgeordnet wurden, dass komplette *Pakete* von Strahlen parallel traversiert wurden. Zusätzlich ermöglicht dies eine signifikante Reduktion der Speicherbandbreite, was auch für Hardwarelösungen von großem Interesse ist [SWS02]. Desweiteren wurden Programmcode und Datenorganisation explizit so ausgelegt, dass die Cacheausnutzung maximiert, sowie Speicherzugriffe, Sprünge und Abhängigkeiten minimiert werden.

Zusätzlich zu diesen "low-level" Optimierungen wurde darauf geachtet, auch aus algorithmischer Sicht nur die effizientesten Algorithmen zu verwenden. So werden insbesondere die kd-Bäume mit den derzeit besten bekannten Techniken konstruiert, indem erweiterte Kostenabschätzungsfunktionen (die "Surface Area Heuristic" [Wal04, Hav01]) benutzt werden, um die zu erwartende Anzahl an Traversierungs- und Schnittberechnungsoperationen zu minimieren. In ihrer Gesamtheit erlauben diese Techniken, eine bis zu 30-fach höhere Ausführungsgeschwindigkeit als bei herkömmlichen Ray-Tracern zu erreichen [WSBW01], womit interaktive Performanz selbst auf Standard-PCs möglich wird.

2.2 Ein Framework zur effizienten Parallelisierung auf PC Clustern

Während ein extrem optimierter Kern für ein ERT System unabdingbar ist, so ist es für viele reale Anwendungen – bei denen oft mehrere Millionen Strahlen pro Bild berechnet werden müssen – doch nötig, die Gesamtperformanz durch Parallelisierung weiter zu steigern. Zu diesem Zweck wurde der ERT-Kern um ein Modul erweitert, welches die parallele Berechnung auf mehreren vernetzten PCs ermöglicht. Dabei musste insbesondere das Problem gelöst werden, selbst auf Standardnetzwerken und unter Echtzeitbedingungen eine effiziente Parallelisierung und gute Skalierbarkeit zu erreichen.

Dies erforderte unter anderem eine effiziente dynamische Lastbalancierung, Techniken zur Bandbreitenreduktion und zum Verstecken der Netzwerklatenz, sowie einen hochgradig asynchronen und nebenläufigen Aufbau des Gesamtsystems. Durch die effiziente Nutzung von PC Clustern wurde es möglich, die Gesamt-Performanz um weitere ein bis zwei Größenordnungen zu steigern, und selbst unter extremen Anforderungen Echtzeitperformanz zu erreichen [WSBW01, WBDS03] (siehe Abb. 2 und 3).

Insbesondere wurde dieses verteilte Ray Tracing System auch um ein Modul erweitert, um

selbst solche Szenen interaktiv darstellen zu können, die nicht im Speicher eines einzelnen PCs gehalten hätten werden können. Dazu wurde ein System entwickelt, auf dem nicht jeder Rechner alle Daten haben muss, sondern jeder Rechner für sich die ihm fehlenden Daten "on demand" über das Netz nachläd. Um Leerlaufzeiten beim Nachladen über das Netz zu minimieren, wurden dazu insbesondere die Berechnungen so umgeordnet, dass parallel zum Ladeprozess bereits anderen Strahlen weiterberechnet werden [WSB01].

2.3 Ray-Tracing dynamischer Szenen

Eine der Grundvoraussetzungen für eine hohe Ray Tracing Performanz ist die Verwendnung effizienter Datenstrukturen ("Beschleunigungsstrukturen") wie kd-Bäume, Octrees, Gitter, oder Bounding Volume Hierarchies, da erst diese Verfahren es ermöglichen, die Anzahl der Strahl-Objekt-Schnitttests pro Strahl zu minimieren [Hav01]. Während diese Datenstrukturen für schnelles Ray Tracing also unabdingbar sind, so stellen sie doch ein großes Problem für Echtzeit Ray Tracing dar: Solange die Geometrie der Szene selbst statisch ist, kann man zwar die Beschleunigungsstruktur vorberechnen und in jedem späteren Frame wiederverwenden; für "dynamische" Szenen mit veränderlicher Geometrie jedoch ist dieser Ansatz nicht mehr anwendbar. Eine komplette Neuberechnung der Beschleunigungsstruktur pro Frame ist i.A. ebenfalls nicht durchführbar.

Für "interaktives" Ray Tracing ist es jedoch unerlässlich, auch eine Interaktion des Nutzers mit der Szene zu ermöglichen. Zu diesem Zweck wurde eine hierarchische Methode entwickelt, in der die Szene in in sich statische "Objekte" zerlegt wird, die je eine eigenen Beschleunigungsstruktur enthalten. Diese Objekte werden dann in einer weiteren, darüberliegenen Hiearchiestufe verwaltet, welche zumindest affine Transformationen der Objekte erlaubt, indem statt der Objekte selbst die Strahlen transformiert werden [WBS03a]. Obwohl affine Transformationen nur einen Teilbereich dynamischer Szenen abdecken, reicht dies für fast alle praktischen Anwendungen bereits aus (siehe Abb. 1). Durch Neudefinition eines Teils der Objekte pro Frame können auch nichtaffine, freie Bewegungen zu einem gewissen Grad unterstützt werden (siehe Abb. 1b). Als Seiteneffekt erlaubt diese Methode auch die "Mehrfachinstantiierung" von Objekten, bei der die Geometrie eines Objektes mehrfach verwendet werden kann, womit auch massivst komplexe Szenen mit geringem Speicherbedarf modelliert werden können (siehe Abb. 2). Die Performanz der Methode hängt dabei nur von der *Anzahl* der bewegten Objekte ab (insbesondere nicht ihrer Grösse), und ist i.A. unabhängig von der Gesamt-Szenengrösse.

Abbildung 1: Ray Tracing dynamischer Szenen: a) Eine typische Spiel-Szene, in der sich Roboter hierarchisch animiert durch eine Stadt bewegen (mit Schatten und Reflektionen). b) Eine Animationssequenz, bei der sich das goldene Objekt in der Mitte komplex verformt. c) und d) Ein komplexes Modell ("UNC PowerPlant") aus 12.5 Millionen Dreiecken wird interaktiv zerlegt und editiert.

2.4 Entwicklung einer ERT-Programmierschnittstelle

Eine Grundvoraussetzung, um das Potential der ERT-Technologie einer breiten Nutzerschicht zur erschliessen, sowie deren einfache Verwendung und Erweiterbarkeit zu ermöglichen, ist das Vorhandensein einer geeigneten Programmierschnittstelle (API). Da existierende APIs nicht für die ERT-Technologie geeignet waren, musste ein solches API komplett neu entworfen und entwickelt werden [Wal04]. Dieses "OpenRT" API bildet bereits heute einen Quasi-Standard für die ERT-Technologie.

2.5 Entwurf von ERT-Hardware

Basierend auf den Erkenntnissen des Softwarekerns wurde die weltweit erste Hardware-Architektur für Echtzeit Ray Tracing entworfen, indem die im Rahmen der Dissertation entwickelten Konzepte in Spezial-Hardware realisiert wurden [SWS02, WSS05]. Diese mittlerweile patentierte Hardware wurde bereits prototypisch realisiert, an einer kommerziellen Realisierung wird bereits gearbeitet.

2.6 Effiziente Unterstützung hochgradig komplexer Modelle

Eine bekannte Eigenschaft des Ray Tracing ist, dass die darin verwendeten Beschleunigungsstrukturen typischerweise zu einer logarithmischen Kostenfunktion bzgl. der Modellkomplexität führen. Durch diese Eigenschaft ist Ray Tracing hervorragend dafür geeignet, auch massivst komplexe Modelle effizient handhaben zu können.

Zu diesem Zweck wurden im Rahmen der Disseration – und darauf aufbauend auch noch nach Abschluss der Dissertation – neue Verfahren entwickelt, die auch massivst komplexe Datensätze (insbesondere hochgradig komplexe CAD-Modelle) in Echtzeit darstellen zu können. Dies beinhaltet unter anderem hierarchische Datenstrukturen, sowie spezielle Streaming- und Out-of-core-Verfahren [WSB01, WDS04]. Mit diesen Verfahren können sogar aus mehreren Millionen bis Milliarden Polygonen bestehende Modelle in Echtzeit und *ohne Simplifizierungen* dargestellt werden [WBDS03, WDS04] (siehe Abb. 2).

Abbildung 2: Massiv komplexe Datensätze, interaktiv dargestellt mittels Echtzeit Ray Tracing: a) Drei "UNC Powerplant"s zu insgesamt 37 Millionen Polygonen. b) und c) Eine extrem detailgenaue Landschaftsszene, bestehend aus 28.000 Instanzen von 10 Arten Sonnenblumen, sowie mehreren hochkomplexen Bäumen (siehe c.). Die gesamte Szene besteht aus ca 1 *Milliarde* Polygonen, und wird – auf einem Cluster – interaktiv mit Transparenz und komplexem Schattenwurf dargestellt. d) Ein Modell einer "Boeing 777", bestehend aus 350 Millionen individuellen Polygonen (ca 40GB Daten), interaktiv visualisiert (mit Schatten) auf einem einzigen PC.

2.7 ERT-Anwendungen in VR und Industriellen Design Reviews

Da die Dissertation von Beginn an real existierenden Probleme der interaktiven Computergraphik aufgriff, ergeben sich automatisch direkte Anwendungsfälle in der Praxis. Daher wurde von Anfang an darauf Wert gelegt, das System auch an praktische Anwendungen einzusetzen und zu testen. Zu diesem Grunde wurde ein Virtual Reality (VR) Framework entwickelt, welches Funktionalitäten eines typischen VR-Programms bietet (z.B. Materialzuweisungen, Beleuchtungseinstellungen, Variantenschaltung, Schnitte, Messen, ...), dies aber *komplett* auf der Basis der in der Dissertation entwickelten ERT-Technologie realisiert, und damit auch physikalisch korrekte Visualisierung in industriellen Design-Reviews ermöglicht [BWDS02, WS05a, WBE+05] (siehe auch Abb. 3).

Dieses im Rahmen der Dissertation entworfene – und mittlerweile weiterentwickelte – ERT-basierte VR-System wird mittlerweile kommerziell vertrieben [inT], und bereits in der industriellen Praxis verwendet. So wird dieses System u.A. bei Volkswagen benutzt, um mittels eines dedizierten Ray Tracing Clusters eine 3200×1200 Pixel große "Power-Wall" VR-Installationen für hochqualitative Design-Reviews mit ERT-Technologie zu betreiben. Alle der bisher beschriebenen Konzepte der Dissertation – schnelles Ray Tracing, Parallelisierung, komplexe Modelle, OpenRT Framework, und hochqualitative Schader – werden in diesem Rahmen praktisch angewandt.

Abbildung 3: Photorealistische Visualisierung in industriellen Anwendungen. a) Modell eines "Hella" Autoscheinwerfers. Mit ERT können auch die hoch komplexen Brechungs- und Spiegelungseffekte im Glas exakt – und interaktiv – simuliert werden. b) Modell einer Mercedes-Benz C Klasse Limousine, mit korrekten Glas- und Lack-Effekten, sowie weichen Schatten aus Umgebungsbeleuchtung. c) Interaktive Visualisierung des Innenraums mit vorberechneter globaler Beleuchtung. d) Auch mittels digitaler Materialaquisition vermessene reale Materialien (hier z.B. Holz und strukturiertes Aluminium) können exakt simuliert werden.

3 Interaktive Globale Beleuchtungssimulation

Obwohl Ray Tracing eines der wichtigsten Hilfmittel für die Globale Beleuchtungsberechnung ist, sind nicht alle Ray Tracing Verfahren automatisch "globale" Beleuchtungsverfahren. Insbesondere weiche Schatten und indirekte Beleuchtung werden typischerweise von den meisten Ray Tracing Verfahren nicht unterstützt. Um solche globalen Effeke zu berechnen sind eine Vielzahl von Verfahren bekannt, jedoch ist der Berechnungsaufwand typischerweise nochmals hoeher als bei "normalem" Ray Tracing.

Dennoch eröffnet die Verfügbarkeit der ERT-Technologie erstmals das Potential, auch die globale Lichtausbreitung interaktiv zu berechnen. Zu diesem Zweck wurden im letzten Teil der Dissertation mehrere Verfahren entwickelt, die die ERT-Technologie zur interaktiven Beleuchtungssimulation verwenden (siehe auch Abb. 4).

Instant Global Illumination: Das “Instant Global Illumination” Verfahren [WKB+02, BWS03] wurde in Kooperation mit der Universitaet Kaiserslautern entwickelt, und kombiniert schnelles Ray Tracing, effektive numerische Integration mittels Niederdiskrepanzmustern, ein effizientes Parallelisierungsframework, sowie ein Bildbasiertes, kantenerhaltendes Filterverfahren, um mittels eines Clusters von PCs interaktive Beleuchtungssimulation zu verwirklichen. Dabei werden so gut wie alle globalen Beleuchtungseffekte – direkte und indirekte Beleuchtungs, weiche und harte Schatten, simple Kaustiken, Antialiasing, etc. – unterstützt. Mittels geeigneter Erweiterungen [WBS03b] wurde es sogar möglich, die Methode auf hoch komplexe Modelle zu erweitern.

Echtzeit Photon Mapping: In neuerer Zeit wurde das Verfahren weiterhin auch um ein Photon Mapping Model erweitert [GWS04], mit Hilfe dessen auch die interaktive Simulation von Kaustiken – also durch Brechung oder Spiegelung fokussiertem Licht – ermöglicht wird. Letzteres ist insbesondere erforderlich, um Objekte wie Glaeser, Scheinwerfer, Lichtleiter, etc. physikalisch korrekt simulieren zu können.

Abbildung 4: Interaktive Globale Beleuchtungssimulation: a) Eine einfache Szene, die mit 20+ Bildern pro Sekunde berechnet werden kann. b) Globale Beleuchtung im “UNC PowerPlant” (12.5 Million Dreiecke). c) ”Office” Szene, mit stark indirekter Beleuchtung und Kaustiken. d) ”Conference Room”, mit interaktiver Gobaler Beleuchtung. Da die komplette Beleuchtung pro Bild vollständig neu berechnet wird, können auch die Objekte der Szenen (Tische, Stühle, ...) interaktiv bewegt werden. Für industrielle Anwendungsbeispiele siehe auch Abb. 3b+c.

4 Zusammenfassung

Im Rahmen der vorgestellten wurden zahlreiche neue Verfahren und Methoden entwickelt, die es erstmalig ermöglichen, Ray Tracing mit Echtzeit-Performanz auf moderner PC-Hardware zu realisieren. Die dabei entwickelten Techniken beinhalten unter anderem einen extrem schnellen, auf moderne CPUs optimierten Ray Tracing Kern, ein skalierbares Parallelisierungsframework, eine Methode zur effizienten Unterstützung dynamischer Szenen, sowie eine eigens entworfene, flexible und mächtige Programmierschnittstelle. In ihrer Gesamtheit formen diese Methoden das OpenRT Echtzeit Ray Tracing System, mit welchem es erstmals möglich ist, selbst massivst komplexe Szenen von Milliarden von Polygonen interaktiv zu visualisieren, sowie hochqualitative, teils photorealistische Bildqualität in Echtzeit zu erzeugen. Darauf aufbauend wurde ein Framework zur interaktiven globalen Beleuchtungssimulatoin entworfen und entwickelt, welches es erlaubt, auch globale Beleuchtungseffekte – z.B. weiche Schatten, indirekte Beleuchtung, oder Kaustiken – interaktiv zu simulieren.

Das im Rahmen der Dissertation entwickelte System wird bereits in der Praxis eingesetzt, und erlaubt erstmals den interaktiven Einsatz von Graphiktechniken, die bisher nur für Offline-Anwendungen denkbar waren.

4.1 Neuere Entwicklungen

Wie bereits oben beschrieben erlaubt das in der Dissertation entwickelte System bereits ein breites Spektrum an Anwendungen. Dennoch konnte es – durch seine von Beginn an verfolgte Modularität und Erweiterbarkeit – auch nach Abschluss der Dissertation noch einfach um neue Anwendungen erweitert werden. Diese Erweiterungen beinhalten unter anderem auch Module für das Echtzeit Ray Tracing von komplexen Freiformflächen [WBE+05], Iso-Flächen hochgradig komplexer Volumendatensätze [WFM+05], sowie Ray Tracing punktbasierter Modelle [WS05b] (siehe Abb. 5, auf die aber hier nicht weiter eingegangen werden soll.

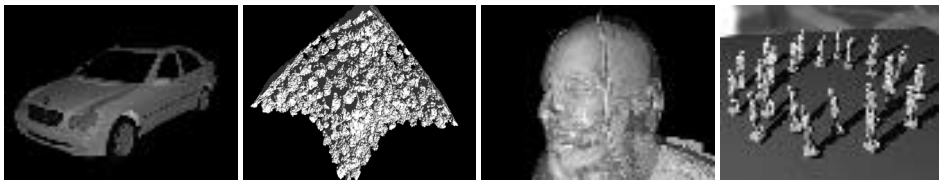

Abbildung 5: Aktuelle Erweiterungen des OpenRT Systems: a) Ray Tracing von Freiformflächen, an einem C-Klasse Modell aus 200,000 Patches, b) Ein Isofläche des 8GB "Richtmyer-Meshkov" Datensatzes, c) Die "Visible Female", mit mehreren semi-transparenten Haut- und Knochenschichten, sowie d) Ein interaktiv geraytracetes Modell aus 24 Millionen Punkten, inklusive Schatten.

Literatur

[AMH02] Tomas Akenine-Möller und Eric Haines. *Realtime Rendering (2nd edition)*. A K Peters, July 2002. ISBN: 1568811829.

[BWDS02] Carsten Benthin, Ingo Wald, Tim Dahmen und Philipp Slusallek. Interactive Headlight Simulation – A Case Study of Distributed Interactive Ray Tracing. In *Proceedings of the 4th Eurographics Workshop on Parallel Graphics and Visualization (PGV)*, Seiten 81–88, 2002.

[BWS03] Carsten Benthin, Ingo Wald und Philipp Slusallek. A Scalable Approach to Interactive Global Illumination. *Computer Graphics Forum*, 22(3):621–630, 2003. (Proceedings of Eurographics).

[GWS04] Johannes Günther, Ingo Wald und Philipp Slusallek. Realtime Caustics using Distributed Photon Mapping. In *Rendering Techniques 2004, Proceedings of the Eurographics Symposium on Rendering*, Seiten 111–121, June 2004.

[Hav01] Vlastimil Havran. *Heuristic Ray Shooting Algorithms*. Dissertation, Faculty of Electrical Engineering, Czech Technical University in Prague, 2001.

[inT] inTrace GmbH. inTrace Realtime Ray Tracing GmbH. http://www.intrace.com.

[SWS02] Jörg Schmittler, Ingo Wald und Philipp Slusallek. SaarCOR – A Hardware Architecture for Ray Tracing. In *Proceedings of the ACM SIGGRAPH/Eurographics Conference on Graphics Hardware*, Seiten 27–36, 2002.

[Wal04] Ingo Wald. *Realtime Ray Tracing and Interactive Global Illumination*. Dissertation, Computer Graphics Group, Saarland University, 2004. Available at http://www.mpi-sb.mpg.de/~wald/PhD/.

[WBDS03] Ingo Wald, Carsten Benthin, Andreas Dietrich und Philipp Slusallek. Interactive Ray Tracing on Commodity PC Clusters – State of the Art and Practical Applications. In Harald Kosch, László Böszörményi und Hermann Hellwagner, Hrsg., *Euro-Par*, Jgg. 2790 of *Lecture Notes in Computer Science*, Seiten 499–508. Springer, 2003.

[WBE+05] Ingo Wald, Carsten Benthin, Alexander Efremov, Tim Dahmen, Johannes Guenther, Andreas Dietrich, Vlastimil Havran, Philipp Slusallek und Hans-Peter Seidel. A Ray Tracing based Framework for High-Quality Virtual Reality in Industrial Design Applications. (submitted for publication), 2005.

[WBS03a] Ingo Wald, Carsten Benthin und Philipp Slusallek. Distributed Interactive Ray Tracing of Dynamic Scenes. In *Proceedings of the IEEE Symposium on Parallel and Large-Data Visualization and Graphics (PVG)*, 2003.

[WBS03b] Ingo Wald, Carsten Benthin und Philipp Slusallek. Interactive Global Illumination in Complex and Highly Occluded Environments. In *Proceedings of the 2003 Eurographics Symposium on Rendering*, Seiten 74–81, 2003.

[WDS04] Ingo Wald, Andreas Dietrich und Philipp Slusallek. An Interactive Out-of-Core Rendering Framework for Visualizing Massively Complex Models. In *Rendering Techniques 2004, Proceedings of the Eurographics Symposium on Rendering*, Seiten 81–92, 2004.

[WFM+05] Ingo Wald, Heiko Friedrich, Gerd Marmitt, Philipp Slusallek und Hans-Peter Seidel. Faster Isosurface Ray Tracing using Implicit KD-Trees. *IEEE Transactions on Visualization and Computer Graphics*, (to appear), 2005.

[WKB+02] Ingo Wald, Thomas Kollig, Carsten Benthin, Alexander Keller und Philipp Slusallek. Interactive Global Illumination using Fast Ray Tracing. *Rendering Techniques*, Seiten 15–24, 2002. (Proceedings of the 13th Eurographics Workshop on Rendering).

[WS05a] Ingo Wald und Hans-Peter Seidel. High-Quality Global Illumination Walkthroughs using Discretized Incident Radiance Maps. (in preparation), 2005.

[WS05b] Ingo Wald und Hans-Peter Seidel. Interactive Ray Tracing of Point Based Models. (submitted for publication), 2005.

[WSB01] Ingo Wald, Philipp Slusallek und Carsten Benthin. Interactive Distributed Ray Tracing of Highly Complex Models. In *Rendering Techniques*, Seiten 274–285, 2001. (Proceedings of Eurographics Workshop on Rendering).

[WSBW01] Ingo Wald, Philipp Slusallek, Carsten Benthin und Markus Wagner. Interactive Rendering with Coherent Ray Tracing. *Computer Graphics Forum*, 20(3):153–164, 2001. (Proceedings of Eurographics).

[WSS05] Sven Woop, Joerg Schmittler und Philipp Slusallek. RPU: A Programmable Ray Processing Unit for Realtime Ray Tracing. *Proceedings of ACM SIGGRAPH*, (to appear), 2005.

Ingo Wald wurde am 29.3.1974 in Bad Kreuznach geboren. Nach dem Abitur 1993 studierte er von 1993-1999 an der Universität Kaiserslautern Informatik, wo er 1999 einen Abschluss als als Diplom-Informatiker erhielt. Danach arbeitete er von 2000-2004 als Doktorand am Lehrstuhl für Computergraphik an der Universität des Saarlandes, wo er das OpenRT Realtime Ray Tracing Projekt leitete. Im Mai 2004 promovierte er mit dem Thema “Realtime Ray Tracing and Interactive Global Illumination”, und arbeitet seitdem als Post-Doc am Max-Planck-Institut für Informatik in Saarbrücken. Seine Forschungsaktiviäten konzentrieren sich auf Echtzeit Ray Tracing auf PCs und PC Clustern, photorealistische Computergraphik, effizientes paralleles Rendering, punktbasiertes und Isoflächen-Rendering, sowie interaktive Visualisierung massiv komplexer Datensätze. Neben seinen wissenschaftlichen Aktivitäten hat Ingo Wald zahlreiche Industrieerfahrung gesammelt (z.B. bei DASA und Intel), ist Mitinhaber eines Patentes auf Ray Tracing Hardware, sowie Mitbegründer und Geschäftsführer der *in*Trace Realtime Ray Tracing GmbH. Ab Juli 2005 übernimmt er eine Stelle als “Visiting Assistant Professor” an der Universtity of Utah.