Echtzeitvisualisierung von SAR-Effekten mittels programmierbarer Grafikhardware

Von der Fakultät Luft- und Raumfahrttechnik und Geodäsie
der Universität Stuttgart zur Erlangung der Würde eines Doktors
der Ingenieurwissenschaften (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von
Timo Balz
aus Stuttgart

Hauptberichter:	Prof. Dr.-Ing. habil. Dieter Fritsch
Mitberichter:	Prof. Dr.-Ing. Uwe Stilla
Mitberichter:	Prof. Dr.-Ing. Alfred Kleusberg

Tag der mündlichen Prüfung: 3. Dezember 2007

Institut für Photogrammetrie
der Universität Stuttgart

2007

Inhaltsverzeichnis

Liste der Abkürzungen und Symbole

API	Application Programming Interface (Programmierschnittstelle)
CPU	Central Processing Unit (Hauptprozessor)
DGM	Digitales Geländemodell
DHM	Digitales Höhenmodell
DOM	Digitales Oberflächenmodell
GPU	Graphics Processing Unit (Grafikprozessor)
GPGPU	General-Purpose Computation on Graphics Processing Units
IEEE	Institute of Electrical and Electronics Engineers, Inc.
LIDAR	Light Detection and Ranging
RADAR	Radio Detection and Ranging
SAR	Synthetic Aperture Radar
SNR	Signal to Noise Ratio
SRTM	Shuttle Radar Topography Mission
UAV	Unmanned aerial vehicle

λ	Radarwellenlänge
$c \approx 3 \cdot 10^8$ m/s	Lichtgeschwindigkeit
$\varphi_{ra}, \varphi_{sa}$	Winkelauflösung der realen und der synthetischen Apertur
l_{ra}, l_{sa}	Länge der realen und der synthetischen Apertur
Δt	Zeitspanne
r_O	Abstand zwischen Sensor und Objekt
δ_r, δ_{rg}	Bodenauflösung in Slant-Range und Ground-Range
δ_{raa}, δ_{saa}	Auflösung der realen und der synthetischen Apertur in Azimut
ϕ	Azimutwinkel
θ_{off}	Off-Nadirwinkel
θ	Lokaler Inzidenzwinkel
ω	Squintwinkel
β	Bistatischer Winkel
P_S	Sendeleistung
P_R	Am Sensor empfangene Leistung
A_{eff}	Effektive Antennenfläche
G	Antennengewinn
σ	Radarrückstreuquerschnitt
σ°	Radarrückstreukoeffizient
ε	Dielektrizitätskonstante
$[S_0]$	Streumatrix
HH, VV, HV, VH	Polarisation der ausgesendeten und empfangenen Welle. H = horizontal; V = vertikal
k	Position der virtuellen Kamera
s	Sensorposition
p	Beobachtungspunkt / Nadirpunkt
$\|\vec{v}\|$	Länge des Vektors $\vec{v}$
$v_{x,y,z,w}, \vec{v}_{x,y,z,w}$	Die *x*-, *y*-, *z*- und *w*-Werte des Punktes bzw. Vektors

Zusammenfassung der Arbeit

Für die Entwicklung neuer SAR-Systeme sowie bei der Interpretation und Auswertung von SAR-Daten sind SAR-Simulatoren wichtige Hilfsmittel. Die Berechnungszeit der SAR-Simulation spielt dabei, sowohl für viele wissenschaftliche Anwendungen als auch für die Sensorentwicklung eine eher untergeordnete Rolle. Diese Anwendungen benötigen möglichst realistische SAR-Simulationsergebnisse. Das andere Extrem stellt die SAR-Echtzeitvisualisierung dar. Hierbei spielt die exakte Simulation eine Nebenrolle, stattdessen besteht das Ziel darin, SAR-Effekte in Echtzeit zu visualisieren. Jedoch sind damit weder neue Sensoren noch neue Konfigurationen zu simulieren bzw. zu entwickeln. Die Anwendungsbereiche liegen in der interaktiven Anwendung, z.B. bei der SAR-Auswertung, oder in der Bereitstellung simulierter Daten für automatische Mustererkennungsverfahren. Auch für die Missionsplanung oder für die Lehre und Ausbildung ist die Rechenzeit ein entscheidender Faktor. In diesen Fällen können Abstriche bei der Simulationsgenauigkeit hingenommen werden.

Ziel der vorliegenden Arbeit ist die Echtzeitvisualisierung von SAR-Effekten für die oben genannten Anwendungen. Denn mittels der immer leistungsfähigeren und flexibler programmierbaren Grafikkarten können SAR-Effekte in Echtzeit visualisiert werden. Dies ist beispielsweise in der Missionsplanung ein entscheidender Vorteil. Durch die hohe Berechnungsgeschwindigkeit können eine Vielzahl möglicher Sensorparameter und Flugbahnen vor der eigentlichen Befliegung getestet werden, um so die optimalen Einstellungen für den jeweiligen Flug zu ermitteln. Die SAR-Effektvisualisierung in Echtzeit erlaubt die schnelle Verfügbarkeit von Sichtbarkeitsanalysen und damit höchste Effizienz bei der Missionsplanung. Dadurch können unnötige und teure Fehler bei der Datenaufnahme vermieden werden.

Bei der Lehre und Ausbildung ist die Interaktivität von enormer Bedeutung. Die Echtzeitvisualisierung erlaubt die verständlichere Präsentation unterschiedlicher SAR-Effekte. Durch die Möglichkeit, bestimmte Effekte in der Simulation an- oder abzuschalten, können die unterschiedlichen Effekte separat oder im Zusammenspiel betrachtet werden. Dies verspricht einen größeren Lernerfolg.

Radarbilder unterscheiden sich in vielerlei Hinsicht von den Bildern passiver Fernerkundungssysteme. Die Abbildungsgeometrie eines Radarbildes ist in Flug- und Entfernungsrichtung unterschiedlich. In Entfernungsrichtung basiert die Abbildung auf einer Laufzeit- bzw. Distanzgeometrie. In Flugrichtung basiert die Abbildung hingegen auf dem Dopplereffekt. Bei der Simulation von SAR-Bildern müssen diese Unterschiede berücksichtigt werden. Für die Berechnung der Rückstreuung von Mikrowellen gibt es in der Radarsimulation unterschiedliche Konzepte. Es gibt Ansätze, die auf der Auswertung von Labormessungen, auf physikalischen Modellrechnungen oder auf Ableitungen von Methoden der Computergrafik beruhen. Daneben sind die durch die Abbildungsgeometrie auftretenden radiometrischen Mehrdeutigkeiten sowie das Speckling zu berücksichtigen.

Die rasante Entwicklung auf dem Gebiet der Computergrafik erlaubt eine immer realistischere Darstellung virtueller Welten. Vor allem Computerspiele nutzen 3D-Grafikkarten zur Visualisierung komplexer Szenen. Zudem wird die Rechenleistung der modernen Grafikhardware zur Datenvisualisierung, aber auch zur Lösung mathematischer Probleme verwendet. Die programmierbaren Grafikkarten der neuesten Generation liefern ein Vielfaches der Rechenleistung handelsüblicher CPUs. Für wissenschaftliche Anwendungen ist dies, vor allem auf Grund des sehr günstigen Verhältnisses von Preis zu Rechenleistung, lukrativ. Die massiv parallel arbeitenden Grafikkarten sind für einige rechenintensive Anwendungen besonders geeignet und lassen sich dank ihrer flexiblen Programmiermöglichkeit für eine Vielzahl von Anwendungen nutzen.

Diese Technik kann auch zur Darstellung von Radardaten benutzt werden. Hierfür sind allerdings einige Anpassungen nötig. Diese sollen dynamisch und in Echtzeit erfolgen. Für die Geometrie übernimmt der sogenannte Vertex-Shader der Grafikkarte diese Aufgabe. Die dynamisch veränderten Geometrien werden an den Pixel-Shader weitergereicht. Dieser ist für die Berechnung der Radiometrie zuständig. Durch die flexible Programmierbarkeit der Pixel-Shader können unterschiedliche Methoden zur Berechnung der Rückstreuung elektromagnetischer Wellen implementiert werden. Um die realistische Darstellung in Echtzeit zu ermöglichen, werden Methoden der Computergrafik adaptiert. Eine Anpassung an die SAR-Geometrie und Radio-

metrie ist erforderlich. Dadurch lassen sich auch komplexe Szenen, bistatische Sensorkonfigurationen oder die Auswirkungen der Bewegungen von Objekten in der Szene visualisieren.

Die Echtzeitvisualisierung hat jedoch auch ihre Grenzen. Im Gegensatz zu der üblicherweise in SAR-Simulatoren verwendeten Ray-Tracing-Methode werden bei der auf der Grafikkarte berechneten Rasterisierung die Strahlen nicht verfolgt. Deswegen können keine Mehrfachreflexionen dargestellt werden. Für die Missionsplanung z.B. spielen die vorhandenen Einschränkungen jedoch nur eine untergeordnete Rolle. Ein Vergleich zwischen den Ergebnissen der SAR-Effektvisualisierung in Echtzeit und Ergebnissen von anderen Simulatoren zeigt, dass die Unterschiede gering sind. Dies gilt insbesondere für natürliche Objekte. Künstliche, vor allem metallische, Objekte weisen jedoch eine Vielzahl von Mehrfachreflexionen auf. Die Unterschiede in den Simulationsergebnissen sind in diesen Fällen weit ausgeprägter.

SAR-Effektvisualisierung in Echtzeit eröffnet dem Nutzer eine Vielzahl neuer Möglichkeiten. Die Interaktivität der Simulation führt zu Verbesserungen in der Missionsplanung, der Lehre und bei der Auswertung von SAR-Daten. Die fehlenden Mehrfachreflexionen sind für viele Anwendungen irrelevant. Von der rasanten Entwicklung im Bereich der Computergrafik kann die SAR-Simulation profitieren. In Zukunft kann durch Fortschritte in der Hardwareentwicklung mit noch realistischeren Ergebnissen gerechnet werden.

Summary

SAR simulators are important tools for developing new SAR systems as well as for analyzing acquired SAR data. The time needed to calculate the simulation is not crucial for most scientific applications or in sensor design. For these applications, the SAR simulation has to be as realistic as possible. The other extreme is the real time visualisation of SAR effects. Here the realism of the simulation is not crucial, but the visualisation must be in real time. It is not possible to develop new sensors using the visualisation of SAR effects only. Real time SAR simulations are useful for interactive applications, like assisting the analysis of SAR data and creating simulated data for pattern recognition applications. Furthermore, applications like mission planning, education and training require fast calculation and interactivity. However, these applications accept less accurate simulation results.

This thesis will discuss the development of the real time visualisation of SAR effects for those applications. Using the more and more powerful and flexible graphics processing units (GPU), it is possible to visualise SAR effects in real time. For mission planning as an example, real time visualization is crucial. With faster visualisations, many different sensor properties and flight paths can be tested to evaluate the optimal parameters, before flights actually take place. The fast availability of shadow and layover analysis guarantees the highest efficiency in mission planning. This reduces unnecessary and expensive errors during data acquisition.

In training and education users should be able to interactively change simulation parameters and see the results in real time. This improves the user's experience and learning results. Furthermore, the ability to switch particular effects on or off (e.g. the speckle noise) guarantees a deeper understanding of each single effect and the complete processes during SAR data interpretation.

Radar images differ in many ways from images acquired by passive sensor systems. The imaging geometry of a SAR system is different in azimuth and range directions. In range direction the placement of objects depends on the distance between objects and the sensor. In azimuth, the geometry is based on the Doppler Effect. All radar simulations must consider the Doppler Effect. There are different methods for calculating the microwave backscattering. Some are based on lab measurements, others base on physical models, and further methods are derived from computer graphics. Furthermore, the radiometric ambiguities, as well as speckling, have to be considered.

Rapid development in computer graphics enables greater and more realistic visualisation of virtual worlds. Computer games, a widely popular application, use the power of modern 3D graphics cards for rapid visualisation of complex scenes. GPUs are also used for visualizing scientific data and solving mathematical problems. GPUs of the latest generation have more computational power than standard CPUs. Due to their comparably cheap prices, programmable graphic cards are especially lucrative for scientific applications. Massive parallel graphic processing units are particularly suitable for the calculation of intense applications and are useable for a variety of applications, due to their flexible programmability.

GPUs can also be used for visualizing radar images. However, some adjustments are necessary. For instance, the geometry of SAR images is different, because in range direction the geometry is based on the time of flight (TOF) of the signal. The radiometry is also different due to the different wavelengths of the signal. Geometry and radiometry must be adapted in real time. Geometry is adapted in the vertex stage of the graphics card using so called vertex shaders. The dynamically adapted geometry is piped into the fragment stage where pixel shaders are used to calculate the radiometry. Because of the programmability of these shaders, different models for backscattering electromagnetic waves can be implemented. Different approaches to computer graphics used in a variety of computer games are adapted to realise visualisation of SAR data in real time. These approaches, however, must be modified in order to use them for SAR geometry and radiometry. By further modifying these approaches, complex scenes, bi-static missions, or effects of moving objects can be visualised.

Real time visualisation used on graphics cards does have some limits. Current GPUs are based on rasterization. Unlike in ray tracing, the paths of the rays are unknown. Therefore multiple reflections cannot be visu-

alised. However, these limitations are not very important in most applications. For instance, mission planning does not require visualization of multi-bouncing. By comparing the results with SAR simulators based on ray tracing, only small differences are visible when natural landscapes are simulated. In contrast, by simulating man made objects, especially metallic objects, huge differences occur.

Visualisation of SAR effects in real-time offers a huge variety of new possibilities to users. Interactivity improves mission planning tools, as well as the analysis of SAR data. Moreover, fast and interactive visualisations offer many training and education benefits. Compared with the possibilities from the real time visualisation capability, the absence of multi-bouncing is only a minor problem. SAR simulations can greatly benefit from the tremendous development in computer graphics. Due to ongoing hardware development, even more realistic results are expected in the near future.

1 Einleitung

1.1 Motivation

Die Fernerkundung und die aus der Fernerkundung gewonnenen Daten liefern entscheidende Grundlagen für eine Vielzahl wissenschaftlicher Untersuchungen und politischer Entscheidungen. Hochaufgelöste Satellitendaten sind heute, dank Google Earth (siehe www.earth.google.com), für jedermann schnell und kostenlos verfügbar. Neben der wachsenden Anzahl verfügbarer hochauflösender Daten im optischen Bereich spielen Radarsysteme, die im Mikrowellenbereich arbeiten, eine zunehmende Rolle bei der Datengewinnung.

1.1.1 Einsatzbereiche und Vorteile der Radarfernerkundung

Radarsysteme sind unabhängig von der Beleuchtung und den Witterungseinflüssen, da die Systeme aktiv elektromagnetische Wellen im Mikrowellenbereich ausstrahlen, die Wolken und Wasserdampf durchdringen können. Diese Unabhängigkeit von äußeren Einflüssen ist für eine Vielzahl von Anwendungen, vor allem im Militär- und Grenzsicherungsbereich, aber auch im Katastrophenmanagement und beim Umweltmonitoring, ein entscheidender Vorteil. Fernerkundungsanwendungen in Klimazonen mit nahezu ständiger Wolkenbedeckung sind auf Systeme angewiesen, die unabhängig von der Bewölkung arbeiten. Die zurückgestreute Energie in ein aussagekräftiges Bild umzuwandeln ist der erste Schritt zur Interpretation der Daten. Dies geschieht im Normalfall durch das SAR-Verfahren (Synthetic Aperture Radar). Radare mit synthetischer Apertur haben den Vorteil, dass die erzielbare Auflösung am Boden unabhängig von der Entfernung des Sensors zum beobachteten Objekt ist. Dies ist insbesondere für Aufklärungs- und Überwachungsanwendungen von großer Bedeutung.

Zusätzlich zur Wetterunabhängigkeit ermöglichen SAR-Systeme eine schnelle Datenaufnahme, da sie einen vergleichsweise breiten Geländestreifen erfassen. Deswegen, und vor allem wegen ihrer Wetterunabhängigkeit, sind sie wichtige Datenquelle für die kartographische Datenerhebung insbesondere im tropischen Bereich, wie verschiedene Kartographiemissionen, z.B. in Brasilien (Moreira et al, 2001), Indonesien (Hoffmann & Fischer, 2002) oder die weltweite SRTM-Mission (Bamler, 1999) beweisen. Bei diesen Missionen wird zusätzlich die Fähigkeit von SAR-Systemen genutzt mittels Interferometrie Höhendaten zu erfassen. Durch die Verwendung zweier Empfangsantennen, deren relative räumliche Lage zueinander sehr genau bekannt ist, können hochgenaue Höhendaten erfasst werden. Die hohe Auflösung und Genauigkeit bei gleichzeitig hoher Flächenleistung bei der Datenaufnahme senken die Kosten für die Erstellung von Höhendaten im Vergleich zu LIDAR-Systemen oder der photogrammetrischen Erfassung. Allerdings ist die vertikale Genauigkeit sowohl von LIDAR-Daten als auch bei der photogrammetrischen Auswertung von Luftbildern deutlich höher (Mercer, 2001).

1.1.2 Schwierigkeiten bei der Radarfernerkundung

Entscheidend für den Einsatz von Fernerkundungsdaten ist deren Interpretation. Durch die Interpretation, entweder manuell oder automatisch, werden die Daten zur Information. Die manuelle Auswertung optischer Fernerkundungsdaten ist vergleichsweise einfach. Die Bilder können auch von Laien interpretiert werden. Es gibt zahlreiche automatische Verfahren zur Klassifikation und Interpretation von optischen Fernerkundungsdaten, die je nach verwendeten Daten und gesuchten Objekten mehr oder weniger genau und erfolgreich sind. Die Interpretation von SAR-Daten, sei sie automatisch, halbautomatisch oder manuell, ist komplizierter. Dies liegt hauptsächlich an den längeren Wellenlängen und der unterschiedlichen Bildgeometrie. Eine erfolgreiche Bildinterpretation stützt sich auf drei wichtige Punkte:

1. Ausreichende Auflösung (geometrisch, radiometrisch, temporär, usw.) der Daten, um die spezifische Interpretationsaufgabe zu lösen.
2. Kenntnisse des Auswerters über das zu erkennende und auszuwertende Objekt, d.h. Objekte können nur dann erkannt und interpretiert werden, wenn der Operator das Objekt kennt und seine Interpretation auf Zusatzinformation und Wissen stützen kann.
3. Kenntnisse des Operators über die Abbildungseigenschaften des Sensors.

Der Erfolg einer Interpretation ist eine Frage der verfügbaren Sensorik. Die Verfügbarkeit eines für die Aufgabe geeigneten Sensors ist die Grundlage für die weitere Auswertung. Des Weiteren ist die Kenntnis über das zu interpretierende Objekt eine Vorraussetzung für die Interpretation. Dies ist Aufgabe der fachlichen Ausbildung eines Auswerters und hängt vom Aufgabenziel ab. Der dritte Punkt ist bei der Auswertung von Radardaten besonders zu beachten. Kenntnisse über die Abbildungseigenschaften von Radardaten sind entscheidend, da sich Radardaten sowohl in ihrer Abbildungsgeometrie als auch in der Radiometrie stark von optischen Fernerkundungsdaten unterscheiden.

Für die Interpretation von Einzelobjekten in Fernerkundungsdaten kann man zwischen vier Interpretationsstufen unterscheiden:

Detektion:	Die Präsenz eines Objektes wird festgestellt, ohne dass die Objektart erkannt wird
Erkennung:	Die Objektklasse wird erkannt (z.B. Brücke, Fahrzeug, usw.)
Identifikation:	Der Objekttyp wird erkannt (z.B. VW Käfer)
Technische Analyse:	Spezielle Details des Objekts werden erkannt, z.B. bauliche Veränderungen

Die einzelnen Stufen erfordern ein unterschiedlich hohes Auflösungsvermögen des Systems und ein unterschiedlich hohes Wissen des Operators. Dabei nimmt sowohl das geforderte Auflösungsvermögen des Sensors als auch der nötige Wissenstand des Anwenders von der Detektion zur Technischen Analyse hin ständig zu. Die nötige räumliche Auflösung hängt entscheidend vom Sensortyp ab. Mit Hilfe eines multispektralen Sensors kann mehr erkannt werden, als mit einem panchromatischen Sensor gleicher Auflösung, welcher ebenfalls deutlich mehr Interpretationen zulässt als ein SAR-Sensor, wie in Tabelle 1-1 erläutert und in Abbildung 1-1 erkennbar ist.

Tabelle 1-1: Interpretierbarkeit von verschiedenen Sensoren und unterschiedlichen Auflösungen (FAS, 1998)

Auflösung	***Sichtbares Licht***	***SAR***	***Multispektral***
≈ 1m	Unterscheidung zwischen LKW mit und ohne Anhänger	Lücken zwischen Zugwaggons sind detektierbar	Auto kann auf einem Parkplatz detektiert werden
≈ 0,5m	Unterscheidung unterschiedlicher Helikoptermodelle	Unterscheidung von feststehenden/variablen Flugzeugflügeln	Fußspuren im Gras und frische Erdarbeiten
≈ 0,1m	Identifizierbarkeit einiger LKW-Aufbauten	Helikoptermodelle sind unterscheidbar	Fuchslöcher können entdeckt werden

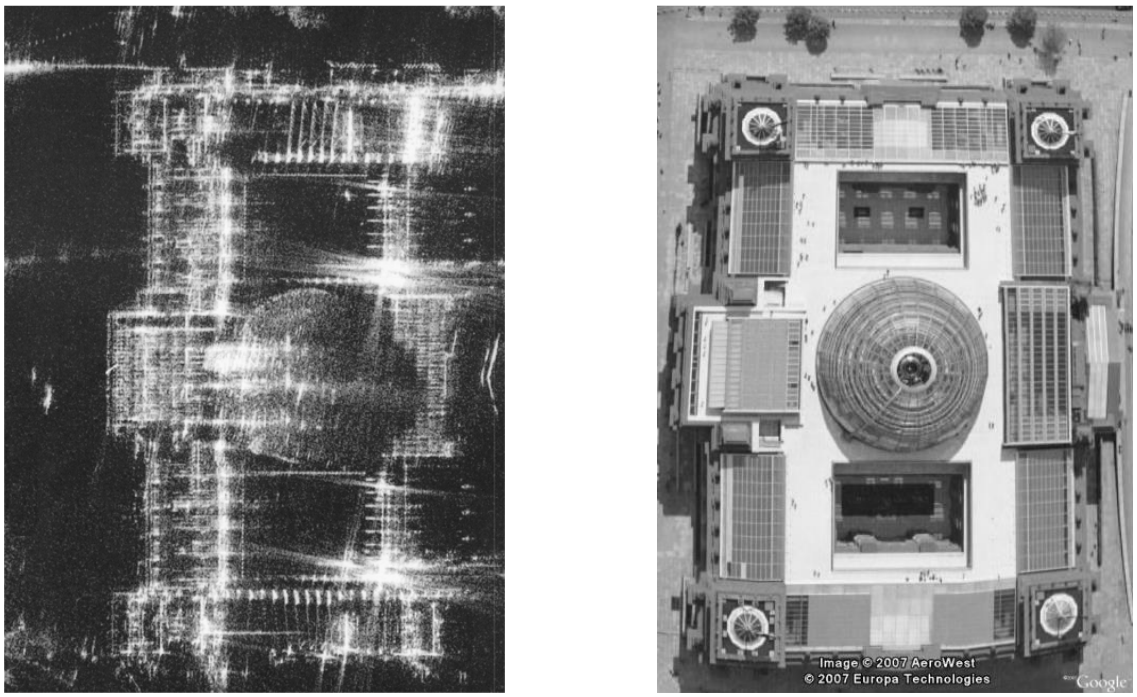

Abbildung 1-1: Vergleich zwischen hochauflösenden SAR-Daten (links) (PAMIR-System von FGAN/FHR) (Ender & Brenner, 2003) und optischen Daten (rechts) (Quelle: Google Earth)

Bei vergleichbarer geometrischer Auflösung der Aufnahmen des Reichstages in Abbildung 1-1 ist das optische Bild, trotz der geometrischen Verzerrungen durch die fehlerhafte Orthobildgenerierung, nicht nur einfacher zu interpretieren, es können ferner weit mehr Aussagen über Details des Gebäudes getroffen werden. SAR-Bilder sind deutlich schwieriger zu interpretieren, da zahlreiche Störeffekte auftreten. Der Einsatz von SAR-Simulatoren ist eine Möglichkeit, die Auswertung von SAR-Bildern zu unterstützen.

1.1.3 Echtzeitsimulationen zur Unterstützung der SAR-Datenauswertung

Unter einem Echtzeitsimulator wird in dieser Arbeit ein Simulator verstanden der die Echtzeitbedingungen der Computergrafik erfüllt und zwischen 20 und 30 Ergebnisbilder pro Sekunde berechnen und darstellen kann. Echtzeitsimulatoren können die Auswertung in mehreren Aspekten unterstützen. Zum einen dienen Simulatoren der Ausbildung und können helfen, die SAR-Effekte zu verstehen und zu verinnerlichen. Des Weiteren können Simulatoren die Auswertung direkt stützen. Die Frage, ob ein Gebäude durch ein Erdbeben beschädigt wurde, kann mit Hilfe eines Vergleiches zwischen SAR-Aufnahmen vor und nach der Katastrophe vergleichsweise leicht beantwortet werden. Allerdings sind oftmals keine SAR-Daten aus der Zeit vor der Katastrophe vorhanden. Falls jedoch LIDAR-Daten oder 3D-Stadtmodelle vorhanden sind, können die fehlenden SAR-Daten simuliert werden, so dass ein Vergleich zwischen einem simulierten unbeschädigten Modell und der realen SAR-Aufnahme erfolgen kann.

Auch die Objekterkennung kann, sowohl bei der manuellen als auch bei der automatischen Bildauswertung, durch die Simulation unterstützt werden. Der Vergleich zwischen dem Objekt und simulierten Modellen kann helfen, den Objekttyp zu bestimmen. Für die genannten Aufgaben ist jedoch eine schnelle Reaktionszeit der Simulation wichtig. Minuten auf ein Simulationsergebnis zu warten ist weder bei der Ausbildung noch bei der simulationsunterstützten Bildauswertung akzeptabel. Erst eine Darstellung der Simulationsergebnisse in Echtzeit erlaubt den sinnvollen Einsatz für interaktive Anwendungen.

Um komplexe Modelle und Szenen in Echtzeit zu visualisieren, bedarf es einer neuen Vorgehensweise. Hierfür werden Methoden aus der Computergrafik und der Spieleindustrie übernommen und adaptiert. Computerspiele sind ein Milliardenmarkt und das Budget für die Entwicklung eines Spieles kann heutzutage mehrere Millionen US-Dollar betragen. Die hohen Umsätze in diesem Bereich sorgen für eine schnelle technische Entwicklung bei der Erstellung immer realistischerer Visualisierungen. Die erzielten Fortschritte in der Hard- und Softwareentwicklung können in der Photogrammetrie, z.B. zur Visualisierung von Geodaten (Fritsch, 2003), bei der terrestrischen Photogrammetrie (Kada, 2004) oder für die Visualisierung von SAR-Daten genutzt werden. Speziell für Computerspiele entwickelte Hardware sowie die existierenden Methoden und Bibliotheken können verwendet werden, so dass neben der preiswerten aber leistungsstarken Hardware auch umfangreiche Programmierschnittstellen zur Verfügung stehen.

Diese Methoden erlauben die Entwicklung einer neuen Generation von SAR-Simulatoren für die Echtzeitvisualisierung von SAR-Daten. Die Echtzeitfähigkeit eröffnet neue Anwendungen für die SAR-Simulation, SAR-Auswertung, SAR-Schulung und SAR-Missionsplanung. Die Verbesserung der Interpretation und der Ausbau der möglichen Anwendungsbereiche, sowie die Einbeziehung neuer Anwender sind mitentscheidend für den Erfolg der aktuellen und zukünftigen SAR-Projekte. Satellitengestützte hochauflösende SAR-Daten, wie sie z.B. TerraSAR-X liefern wird (Werninghaus et al, 2004), verbreitern das Datenangebot und die Auswertung dieser Daten kann durch die SAR-Effektvisualisierung in Echtzeit unterstützt werden.

1.2 Einteilung der unterschiedlichen SAR-Simulationsprinzipien

Die SAR-Simulation leistet schon lange einen hilfreichen Beitrag zur Radarfernerkundung. *La Prade* (1963) präsentierte schon in den sechziger Jahren die Anwendung der Radarsimulation für die Planung von Stereobefliegungen. *Holtzmann et al* (1978) präsentierten Ende der siebziger Jahre ihre Arbeit zur „Radar Image Simulation“. Bei der SAR-Simulation kann man *Marconi* (1984) (nach Leberl, 1990) zufolge zwischen drei Simulationstypen unterschieden:

1. Die „Systemsimulation“, die das komplexe Radarsignal mittels eines DHM und Informationen über die Geländebeschaffenheit berechnet.

2. Die „Bildsimulation“, welche nicht die komplexen Radarsignale simuliert, sondern nur die zur Erzeugung eines realistischen Bildes nötigen Signalanteile. Das Speckling wird statistisch berechnet und ist kein integraler Bestandteil der Simulation. Auch die Frequenzfilterung und andere Effekte bleiben unberücksichtigt.
3. Eine Simulation von SAR-Bildern basierend auf echten SAR-Daten einer anderen Szene oder eines anderen Geländeausschnittes.

Franceschetti et al (1995) unterscheiden, analog zu den ersten beiden Typen nach *Marconi* (1984), zwischen SAR-Bildsimulatoren und SAR-Rohdatensimulatoren. Bei der Bildsimulation ist das gewünschte Endprodukt ein simuliertes SAR-Bild, im anderen Fall sind es die Rohdaten, welche erst prozessiert werden müssen, um ein Bild zu erzeugen. Eine weitere Unterteilung erfolgt in Punktsimulatoren und Szenensimulatoren (Franceschetti et al, 1995). Punktsimulatoren konzentrieren sich auf die Simulation des Sensors und dessen Charakteristika, wohingegen Szenensimulatoren sich auf die realistische Darstellung von Bild- bzw. Szeneninhalten fokussieren und der Schwerpunkt hier auf der Simulation der Rückstreuung liegt. Tabelle 1-2 zeigt eine Zusammenfassung der unterschiedlichen Simulationsprinzipien.

Tabelle 1-2: Simulationsprinzipien

Simulationsprinzip		***Simulator***	***Literatur***
Rohdatensimulator	Punktsimulation	SARSIM	(Pike, 1985)
	Szenensimulation	SARAS SETES	(Franceschetti et al, 1992) (Speck et al, 2002)
Bildsimulator	Szenensimulation	PIRDIS	(Meyer-Hilberg, 2006)
	Szenenvisualisierung	SARViz	(Balz, 2006b)

Die Berechnung des Rückstreukoeffizienten $\sigma°$ (siehe Kapitel 4.4.1 und Anhang A.3) erfolgt als Funktion aus der Geländebeschaffenheit und dem lokalen Inzidenzwinkel. Hierfür gibt es unterschiedliche Methoden (vgl. Leberl, 1990):

- Die Bestimmung von $\sigma°$ erfolgt anhand von Tabellen. Diese werden aus Messwerten, Laborwerten und statistischen Auswertungen erzeugt, so dass pro simuliertem Geländetyp und lokalem Inzidenzwinkel der Rückstreukoeffizient $\sigma°$ ermittelt wird. Beispiel hierfür ist die Arbeit von *Holtzman et al* (1978), aber auch die in dieser Arbeit verwendete Berechnung nach den Tabellen von *Ulaby & Dobson* (1989) (siehe Kapitel 4.4.1.1).
- Der Rückstreukoeffizient wird aus einem DHM und einem realen Radarbild abgeleitet. Bei dieser z.B. von *Muhleman* (1964) oder *Domik & Leberl* (1987) angewendeten Methode wird die Rückstreuung eines realen Bildes mit einem DHM des Gebietes untersucht. Die Rückstreuung eines Geländetyps ohne den Einfluss des lokalen Inzidenzwinkels, wird mit Hilfe eines DHM bestimmt. Für das simulierte Bild wird dann dieser Rückstreuungswert unter Berücksichtigung des lokalen Inzidenzwinkels im simulierten Bild verwendet.
- Bei der direkten Berechnung des Rückstreukoeffizienten werden die Rückstreukoeffizienten basierend auf physikalischen Modellen berechnet. Hierfür existieren eine Reihe von Ansätzen und Modellen. Ein häufig verwendetes Modell ist das „Integral Equation Model (IEM)“ nach *Fung* (1994). In dieser Arbeit wird die Berechnung des Rückstreukoeffizienten nach dem Modell von *Zribi* (2006) implementiert (siehe 4.4.1.3).

1.3 Entwicklung der SAR-Simulation

Probleme bei der Akquisition von stereotauglichen Radardaten veranlassten *La Prade* (1963) zur Entwicklung einer Radarsimulation zur Missionsplanung. Ziel war die Optimierung der Flugparameter für die Erfassung stereotauglicher Radarbilder. Hierfür wurde auf den simulierten mathematischen Oberflächen überprüft, ob und mit welcher Genauigkeit Parallaxenmessungen durchgeführt werden können. *Muhleman* (1964) entwickelte eine Gleichung zur Berechnung der Radarrückstreuung der Venus und des Mondes. Die Rückstreuung berechnet *Muhleman* aus dem lokalen Inzidenzwinkel und der Muhlemanschen Konstante *M*, die mittels realer SAR-Bilder bestimmt wurde.

Die frühen Arbeiten zur SAR-Simulation von *Holtzman et al* (1978) bilden die Grundlagen vieler späterer SAR-Simulationen. So ist das sogenannte „Point Scattering Model“ heute noch Grundlage vieler Simulationen. Die Eingangsdaten werden in Auflösungszellen unterteilt. Eine Auflösungszelle dehnt sich dabei in Azimut- und Entfernungsrichtung entsprechend der geometrischen Auflösung des zu simulierenden Systems aus. Die Auflösungszelle wird als Punkt repräsentiert. Sie enthält Informationen über die Reflektivität und die Höhe der Daten pro Zelle, also ein räumliches Reflexionsmodell und ein Höhenmodell. Daraus berechnet sich das endgültige SAR-Bild, wobei das Speckling in Form einer Rayleighstreuung simuliert wird. Das Ergebnis wurde, wie bei den damaligen SAR-Systemen üblich, auf Film ausgegeben.

Wohlers et al (1980) simulierten die Abbildung von Flugzeugmodellen auf dem Computer, um die simulierten Bilder für die Entwicklung und den Test von Mustererkennungsalgorithmen zu verwenden. *Wohlers et al* setzen nicht wie *Holtzmann et al* auf das „Point Scattering Model“, sondern auf eine kontinuierliche Abbildung vergleichbar mit der Visualisierung in der Computergrafik. Die Besonderheiten der SAR-Geometrie und Reflektivität werden berücksichtigt. Das Bild wird anschließend an die Auflösung des SAR-Systems angepasst. Die vorher kontinuierliche Abbildung wird in das zu simulierende Auflösungsraster des Bildes digitalisiert.

Zur Unterstützung der Analyse und Interpretation von Radarbildern benutzten *Domik et al* (1984) Radarsimulationen auf Basis von DHM-Daten. Die Simulation dient dabei der geometrischen Entzerrung der realen Radarbilder und der Zuordnung zwischen Radarbildern und anderen Datensätzen. Die Berechnung der Rückstreuung erfolgt, unter anderem, auf Basis der Funktion von *Muhleman* (1964). Diese vereinfachende Berechnung ist nach *Domik et al* für die geplanten Anwendungen, wie der geometrischen Entzerrung und der radiometrischen Analyse, ausreichend.

Pike (1985) beschreibt ein Softwaresimulationsmodell, welches die Rohdaten pulsweise simuliert. SARSIM konzentriert sich auf die Sensormodellierung und generiert Rohdaten. Das System wurde auf einem CRAY-1/S Supercomputer im Wert von mehreren Millionen US-Dollar implementiert und simulierte kleinere Szenenausschnitte in wenigen Minuten. Die zu simulierende Szene wird in eine Reihe von Punktzielen unterteilt, deren Pulsantwort einzeln berechnet wird. Die simulierten Rohdaten müssen dann im SAR-Prozessor prozessiert werden. SARSIM eignet sich damit sowohl zum Sensordesign als auch zur Entwicklung von SAR-Prozessoren und wurde zur Simulation von ERS-Daten eingesetzt.

Mit SARAS stellen *Franceschetti et al* (1992) einen Rohdatensimulator für umfangreiche Szenen vor. Statt der Unterteilung der Szene in einzelne Punktstreuer und einer simulierten Rückstreuung basierend auf Datenbankabgleichen wird die Rückstreuung von SARAS anhand eines kontinuierlichen Höhenmodells und durch Anwendung der elektromagnetischen Theorie berechnet. Die Rückstreuung ist eine Funktion aus dem lokalen Einfallswinkel, der Strahlung und den elektromagnetischen Eigenschaften des Materials. Diese Methode erfordert weniger Parameter und ist flexibler. Das kontinuierliche Höhenmodell wird in einzelne Ebenen zerlegt und pro Ebene wird der lokale Einfallswinkel berechnet. Dabei berücksichtigt das physikalische Modell der Reflektion das Speckling, wodurch die Realitätsnähe des Simulators verbessert wird. In weiteren Arbeiten vergleichen *Franceschetti et al* die Ergebnisse von SARAS mit realen ERS-1 Daten (Franceschetti et al, 1994). Diese zeigen eine insgesamt gute Übereinstimmung. Fehlende Übereinstimmungen sind oftmals auf fehlerhafte Eingangsdaten der Simulation zurückzuführen.

Eine Vielzahl weiterer SAR-Simulatoren wird in den folgenden Jahren entwickelt und vorgestellt. So entwickeln z.B. *Armand und Vidal-Madjar* (1992) einen weiteren SAR-Simulator für die Rohdatensimulation. *Horell und Inggs* (1993) stellen einen SAR-Simulator, basierend auf der kommerziellen Software zur Radarsimulation CS-IRSS von „CAE SOFT“, vor. *Yang et al* (1994) präsentieren EMSARS, einen Simulator der auf GIS-Basis die Höhendaten und Landoberflächen bereithält. Die Rückstreuungen der Landoberflächen werden in einem Vorverarbeitungsschritt anhand einer Rückstreuungsdatenbank berechnet. Das simulierte SAR-Bild wird daraus in Abhängigkeit des digitalen Geländemodells berechnet. Sowohl *Song et al* (1997) als auch *Yong et al* (1999) stellen ihre Untersuchungen zum Design eines Punktsimulators, basierend auf Modellen der elektromagnetischen Rückstreuung, vor.

Speck et al (2002) präsentieren mit SETES einen Simulator für satellitengestützte SAR-Systeme, der auf Vorarbeiten von *Gierull und Ruppel* (1996) beruht. Vergleichbar mit SARAS von *Franceschetti et al* erzeugt SETES Rohdaten. In die virtuellen Landschaften lassen sich 3D-Modelle, z.B. von künstlichen Objekten,

platzieren. Der Simulator dient zum Entwurf und zur Entwicklung von neuen SAR-Sensoren und SAR-Systemen. Für die Simulation von SAR-Systemen in UAVs (unmanned aerial vehicle) wurde das System von *Vandewal et al* (2006) angepasst. *Brand et al* (2006) verwenden SETES als SAR-Simulator für die Missionsplanung.

Einen Bildsimulator entwickelte *Meyer-Hilberg* (2006) mit PIRDIS, welches zum Teil auf SARView, aber auch auf Vorarbeiten von *Bickert et al* (2002) aufbaut. SARView ist ein Bildsimulator und wurde von der DaimlerChrysler Forschung im Auftrag der EADS Dornier entwickelt. PIRDIS simuliert keine Rohdaten, sondern erzeugt direkt ein SAR-Bild. SARView unterscheidet zwischen der Simulation von Landschaften und von künstlichen Objekten. Die einzelnen Objekte werden separat simuliert, d.h. es werden keine Reflexionen zwischen unterschiedlichen Objekten simuliert. Dadurch wird die Gesamtdauer der Simulation gesenkt, wodurch die SAR-Simulation auch für Anwendungsgebiete außerhalb des Sensordesigns interessant wird.

Der kommerzielle Simulator von „OKTAL Synthetic Environment“ bietet neben der Radar-Simulation auch die Darstellung der Szenen in anderen Wellenlängen, z.B. in Infrarot, an. Er basiert auf der Ray-Tracing Technologie (Mametsa et al, 2002). Die Echtzeitvisualisierung im Infrarot wurde mit dem SE-FAST-IR Paket in das Produkt integriert (Cathala et al, 2006). Das auf OpenGL basierende SE-FAST-SAR Paket ist in der Entwicklung.

1.4 Entwicklungen in der Visualisierung nicht-photorealistischer Szenen

Im visuellen Bereich ist die Simulation von Szenen Aufgabe der Computergrafik. Die Entwicklung der Computergrafik begann bereits kurz nach Ende des Zweiten Weltkrieges in unterschiedlichen Forschungseinrichtungen. Meilensteine bildeten unter anderem *Sutherlands* (1963) Sketchpad, die Entwicklung des Scan-line Algorithmus (Bouknight, 1970) zur Darstellung von 3D-Szenen, das Z-Buffering (Catmull, 1978), die Beleuchtungsberechnung nach *Phong* (1975), das Environment-Mapping (Blinn & Newell, 1976), die Entwicklung der Shadow Volumes (Crow, 1977) und der Shadow Maps (Williams, 1978), das Ray-Tracing (Whitted, 1980), bis hin zur Radiosity (Goral et al, 1984). Eine gute Übersicht über die Computergrafik leistet *Foley* (1996). *Kent* (2001) bietet eine Übersicht über die Entwicklung der Spieleindustrie, die zunehmend zum Taktgeber der Entwicklungen in der Computergrafik wurde und sein wird.

Die Computerspieleindustrie legte den Grundstein für die starke Verbreitung der 3D-Grafik und der 3D-Grafikkarten. Für eine möglichst schnelle und realistische Darstellung der komplexer werdenden Spielewelten wird immer mehr auf die Verwendung von 3D-Beschleunigerkarten gesetzt. Heutzutage kommt kaum ein Spiel mehr ohne diese Grafikkarten aus. Dadurch förderte die Spieleindustrie die Grafikkartenhersteller, deren Produkte in immer mehr PCs zum Einsatz kommen. Die Entwicklung der Grafikkarten ging, bedingt durch die hohen Umsätze der Spieleindustrie, schnell voran. Die immer größer werdende Verbreitung und das über dem Moore’schen Gesetz liegende hohe Wachstum der Rechenleistung von Grafikprozessoren macht die Nutzung dieser Leistung auch außerhalb der Spiele und der 3D-Visualisierung lukrativ. Die GPGPU (general-purpose computation on GPUs) ist heutzutage eine Disziplin mit vielen Anwendungsmöglichkeiten. *Owens et al* (2007) bieten eine gute Übersicht über die Entwicklungen und die Anwendungen dieser Disziplin. Unter GPGPU versteht man unter anderem die hardwarebeschleunigte Darstellung von nicht-photorealistischen Szenen, wie in dieser Arbeit anhand von SAR-Daten beschrieben, hauptsächlich aber die Berechnung von Gleichungssystemen oder Simulationen mittels Graphics Processing Units (GPU).

Fischer (2006) präsentiert Methoden, die Umwelt und virtuelle Objekte in Anwendungen der Augmented-Reality besser zu verschmelzen. Dabei wird nicht das virtuelle Objekt realistischer dargestellt, stattdessen wird das per Kamera aufgezeichnete Bild der Umgebung verändert. Die Darstellung der realen Welt wird der virtuellen Welt angepasst, in dem z.B. die Umgebung „cartoon-like“, also vergleichbar einem Zeichentrickfilm, dargestellt wird. Diese Veränderungen werden auf der Grafikhardware berechnet, um die gewünschten kurzen Berechnungszeiten für eine „flüssige“ Visualisierung zu gewährleisten. Die Veränderungen in eine nicht-photorealistische Darstellung sollen dem Benutzer ein tieferes Eintauchen in die Augmented Reality ermöglichen, da es zu keinem Stilbruch zwischen der Darstellung der Umgebung und der visualisierten Objekte kommt.

Die Darstellung von Ergebnissen der Computertomographie ist eine Anwendung für die Visualisierung von Sensordaten im nicht-optischen Wellenlängenbereich. *Cabral et al* (1994) präsentieren einen Ansatz auf Basis von spezieller Grafikhardware, *Xu und Mueller* (2005) zeigen die Beschleunigung der Computertomographie auf Standardhardware. *Sherbondy et al* (2003) unterstützen die Segmentierung von Volumendaten, wie sie bei der Computertomographie und der Magnetresonanztomographie benötigt werden, auf Basis programmierbarer Grafikhardware. Gerade in der medizinischen Bildverarbeitung gibt es eine Reihe von Arbeiten zur beschleunigten Visualisierung von nicht-optischen Daten auf herkömmlicher Grafikhardware.

Ein ganz anderes Einsatzgebiet, die hardwarebeschleunigte Darstellung der Relativitätstheorie zu Lehr- und Verständniszwecken, zeigt *Borchers* (2005). Die Visualisierung der Effekte der speziellen Relativitätstheorie erfordert eine Anpassung der Geometrie und der Radiometrie. Die darzustellende Geometrie verändert sich dabei umso mehr, je mehr sich der Beobachter der Lichtgeschwindigkeit nähert. Die Geometrie wird auf der Grafikkarte dynamisch in Abhängigkeit von der Geschwindigkeit des Betrachters visualisiert. Neben der Geometrie verändert sich durch den Dopplereffekt auch die Radiometrie der betrachteten Szene. Da der Anwender diese Effekte in der Realität nicht erleben kann, ist die Simulation und Visualisierung am Computer der einzige Weg, die Welt unter den Bedingungen der Lichtgeschwindigkeit zu erleben.

1.5 Zielsetzung der Arbeit

In dieser Arbeit werden die Methoden der Computergrafik für die SAR-Simulation angewendet. Ziel ist es, 3D-Modelle von Objekten in Echtzeit so zu visualisieren, dass das Ergebnis möglichst einem realen SAR-Bild des modellierten Objektes entspricht. Die Echtzeitvisualisierung wird mittels programmierbarer Grafikhardware ermöglicht. Die verwendeten Techniken basieren einerseits auf Methoden der Computergrafik, die ursprünglich für Visualisierungsanwendungen entwickelt wurden sowie deren Weiterentwicklung für nicht-photorealistische Anwendungen, andererseits auch auf den Entwicklungen der SAR-Simulationen. Die Echtzeitvisualisierung von SAR-Abbildungseffekten mittels Grafikhardware wurde erstmals 2006 (Balz, 2006b) vorgestellt. Diese Arbeit beschreibt das zwischenzeitlich weiterentwickelte System SARViz und dessen Anwendungen.

Obwohl SARViz in Tabelle 1-2 als Bildsimulator geführt wird, handelt es sich um keinen Simulator im eigentlichen Sinne. Die SAR-Effekte werden nicht simuliert, sondern bekannte Effekte der SAR-Abbildung werden visualisiert. Es ist eine SAR-Effektvisualisierung, weswegen sich SARViz nicht für das Sensordesign eignet, da das Verhalten neuer und unbekannter Sensoren nicht modelliert wird. Nichtsdestotrotz eröffnet die Echtzeitfähigkeit des Systems eine Palette weiterer Anwendungen, für die andere Simulatoren ungeeignet sind. SARViz ist somit eine konsequente Fortentwicklung des Weges von SARView und PIRDIS (Meyer-Hilberg, 2006), die bereits schnelle Simulationsergebnisse zum Ziel hatten. Für Ergebnisse in Echtzeit müssen Einschränkungen bei der Simulationsgenauigkeit hingenommen werden. Es können jedoch mittels Methoden, wie sie auch in Computerspielen Anwendung finden, realistische SAR-Effektvisualisierungen erzeugt werden.

1.6 Überblick über die Arbeit

Die Grundlagen der SAR-Bilderzeugung werden in **Kapitel 2** beschrieben. Zunächst wird die grundsätzliche Funktionsweise eines SAR-Systems erläutert, insbesondere die zum Verständnis der SAR-Effektvisualisierung und deren Anwendung nötigen Grundlagen und Theorien. Der Schwerpunkt liegt auf der SAR-Geometrie und Radiometrie. In **Kapitel 3** werden die theoretischen Grundlagen der Grafikprogrammierung erläutert. Neben einer kurzen Einführung in die Entwicklung der 3D-Computergrafik werden die Möglichkeiten der neuen programmierbaren Grafikprozessoren vorgestellt, die eine Vorraussetzung für die Implementierung der SAR-Effektvisualisierung in Echtzeit sind.

Die Implementierung der SAR-Effektvisualisierung wird ausführlich in **Kapitel 4** besprochen. Die einzelnen Implementierungsschritte werden detailliert vorgestellt. Die Schritte für die geometrische und radiometrische Prozessierung der Daten werden beschrieben. Zudem werden die Zusammenarbeit und Abstimmung dieser Prozessierungsschritte samt Implementierungsdetails erläutert. In **Kapitel 5** wird die SAR-Effektvisualisierung mit Beispielen illustriert. Weitere Einflüsse auf das Endergebnis, z.B. durch die Repräsentation der Modelle, werden erläutert und die Ergebnisse mit anderen SAR-Simulationen verglichen.

In **Kapitel 6** werden mögliche Anwendungen der Echtzeitvisualisierung von SAR-Effekten diskutiert. Da die Anwendungsgebiete zahlreich sind, kann in dieser Arbeit nur ein kurzer Überblick über mögliche, beispielhafte Anwendungen erfolgen. Hierfür werden neben der SAR-Missionsplanung, der Lehre und Ausbildung auch Anwendungsmöglichkeiten in der automatischen Objekterkennung und in der Georeferenzierung aufgezeigt. Eine Zusammenfassung und Bewertung der Ergebnisse samt einem Ausblick auf zukünftige Entwicklungen folgt in **Kapitel 7**.

2 Radarsysteme mit synthetischer Apertur

In diesem Kapitel werden zunächst die Grundlagen des Radars und der abbildenden Radarsysteme erläutert. Dann wird auf die Besonderheiten der Radare mit synthetischer Apertur (SAR) eingegangen. Es wird die Abbildungsgeometrie von SAR-Systemen erläutert und schließlich der störende Einfluss von Umgebungsfaktoren auf die Auswertung von SAR-Bildern aufgezeigt.

2.1 Grundlagen der Radarsysteme mit synthetischer Apertur

Bereits im Jahr 1904 stellte der Düsseldorfer Ingenieur Christian Hülsmeyer sein patentiertes „Telemobiloskop“ auf einem Schifffahrtkongress vor. Die Vorstellung seines „Gerät[es] zur Feststellung und Entfernungsbestimmung bewegter metallischer Gegenstände im Nebel“ (Hülsmeyer, 1904) kann als Geburtsstunde des Radars verstanden werden, auch wenn die Entwicklungen Hülsmeyers zunächst nicht vermarktet wurden. Erst 1922/23, mit dem Bau von Höhenmessern für Luftfahrzeuge durch den deutschen Heinrich Löwy, ging die Radarentwicklung in Deutschland weiter (Trenkle, 1979).

In England wurde bereits 1936/37 mit der Erprobung von Radargeräten in Jagdflugzeugen begonnen. Diese sollten die Detektion von Flugzeugen im Luftraum auch bei schlechter Sicht ermöglichen. Allerdings zeigten die Systeme stattdessen Reflexionen von Schiffen an (Godt, 2003). Im Herbst 1941 wurde in England, vermutlich durch Zufall, die Fähigkeit zur Bodenbeobachtung der Radarsysteme entdeckt. Die Küstenlinie und bebaute Bereiche konnten erkannt werden. Im Januar 1943 wurde das System dann erstmals operationell eingesetzt (Godt, 2003). Dies kann als die Geburtsstunde der Radarfernerkundung verstanden werden.

Abbildende Radarsysteme sind für die militärische Aufklärung von besonderem Interesse, da sie einen Einsatz auch bei Wolkenbedeckung und bei Nacht erlauben. Darüber hinaus handelt es sich um Schrägsicht-Systeme, d.h. sie strahlen die Radarkeule seitlich aus, weswegen kein direktes Überfliegen des Aufnahmegebietes nötig ist.

2.1.1 Abbildende Radarsysteme mit realer Apertur

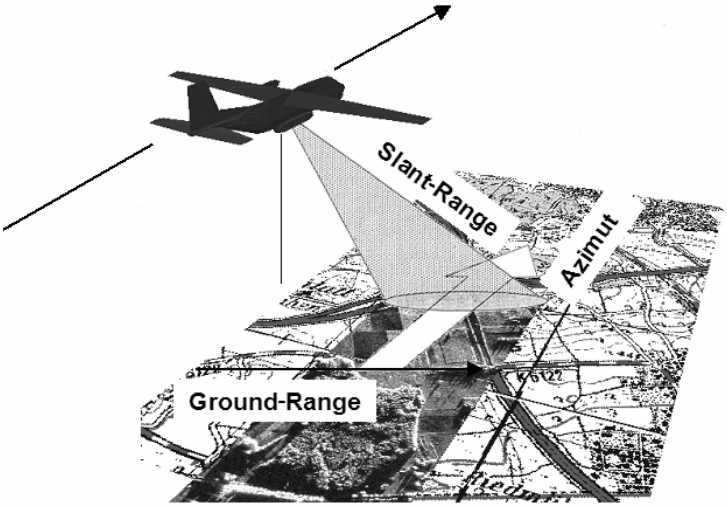

Abbildung 2-1: Abbildende Radarsysteme

Die Bewegungsrichtung des Sensors wird als Azimutrichtung bezeichnet. Die direkte Entfernung vom Sensor zum aufgenommenen Objekt bezeichnet man als Slant-Range. Die Entfernung vom Nadirpunkt des Sensors zum Objekt als Ground-Range. Radarbilder, welche die Schrägentfernung abbilden, werden deswegen als Slant-Range Bilder bezeichnet. Slant-Range Bilder sind die Originalbilder des Systems, jedoch für die Interpretation der Daten von geringerem Interesse, weswegen die Bilder in Ground-Range projiziert werden. Diese Projektion kann auf unterschiedlichen Annahmen beruhen: flache Erdoberfläche, gekrümmte Erdoberfläche oder digitales Höhenmodell.

Die geometrische Auflösung eines realen Radarsystems hängt von der Entfernung zwischen Sensor und Ziel ab (siehe Anhang A.1). Wie in Abbildung 2-2 schematisch dargestellt, bieten, auf Grund der Breite der Radarkeule, Radarsysteme mit realer Apertur nicht die gewünschte hohe Auflösung in großer Entfernung.

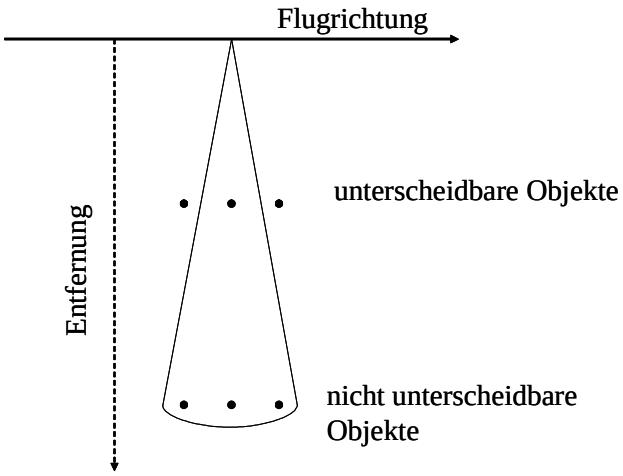

Abbildung 2-2: Entfernungsabhängigkeit der Azimutauflösung bei realer Apertur

2.1.2 Systeme mit synthetischer Apertur

Wiley (1954) löste mit dem „Synthetic Aperture Radar (SAR)“ dieses Problem. Beim SAR-System wird die Antenne künstlich verlängert, indem die Flugbahn des Sensorträgers ausgenutzt wird. Wenn die Flugbahn der sensortragenden Plattform bekannt ist, bzw. als konstant und gleichmäßig angenommen wird, dann kann die Apertur des Systems künstlich verlängert werden. Vereinfacht ausgedrückt nimmt das System mehrere Aufnahmen entlang der Flugbahn auf und speichert diese ab. Aus den Einzelaufnahmen der kleinen realen Apertur wird eine Aufnahme einer künstlich vergrößerten synthetischen Apertur berechnet.

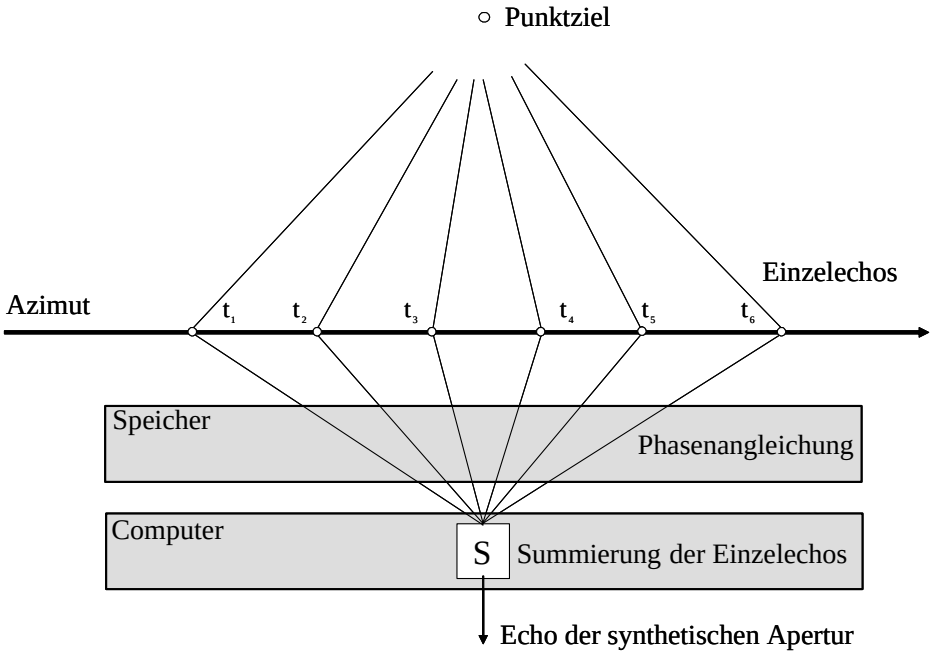

Abbildung 2-3: Schematische Darstellung der synthetischen Apertur

Vergrößert sich der Abstand zwischen Sensor und Objekt, verlängert sich automatisch die synthetische Apertur, was die Auflösungsverschlechterung durch die größere Distanz aufwiegt. Deswegen ist die erreichbare geometrische Auflösung eines Systems mit synthetischer Apertur unabhängig von der Distanz des Sensors zum Objekt (siehe Anhang A.2). Dies ist ein enormer Vorteil von SAR-Systemen und insbesondere beim

Einsatz in der Satellitenfernerkundung von großer Bedeutung. In der technischen Umsetzung nutzen die meisten SAR-Systeme den Dopplereffekt aus. Alle Objekte mit gleicher Entfernung zum Sensor, die sich innerhalb der Radarkeule befinden, werden gleichzeitig zurückreflektiert. Die einzelnen Reflexionen dieser unterschiedlichen Objekte können also nicht voneinander getrennt werden. Die Auflösung hängt somit, im Fall der realen Apertur, von der Keulenbreite ab. Diese ist wiederum abhängig von der Entfernung des Objekts zum Sensor und von der Länge der Apertur.

Durch Ausnutzung des Dopplereffektes gelingt es jedoch, die einzelnen Objektreflexionen auch innerhalb einer Radarkeule voneinander zu trennen. Durch die Frequenzverschiebung, ausgelöst durch den Dopplereffekt, kann unterschieden werden, ob sich der Sensor auf das Objekt zu bewegt, ob sich das Objekt neben dem Sensor befindet oder ob sich der Sensor vom Objekt wegbewegt. Ist die Flugbahn des Sensors bekannt, kann so die exakte Position des Objektes in Azimutrichtung bestimmt werden. Die Entfernung wird durch die Laufzeit des Signals bestimmt.

2.1.3 Pulskompression

Das SAR-Prinzip verbessert die Auflösung in Azimutrichtung, in Rangerichtung hängt die erreichbare Auflösung von der Bandbreite des ausgestrahlten Signals ab. Um hier eine möglichst hohe geometrische Auflösung zu erreichen, muss ein möglichst kurzes Signal mit möglichst großer Stärke gesendet werden. Sehr starke und gleichzeitig sehr kurze Signale sind technisch nicht einfach zu erzeugen und diese technischen Probleme limitieren die Auflösung in Rangerichtung.

Anstatt eine Vielzahl einzelner kurzer Signale zu senden, gibt es die Möglichkeit, ein längeres frequenzmoduliertes Signal zu senden. Dies ist technisch einfacher zu realisieren. Die „einzelnen“ Signale unterscheiden sich durch die Veränderungen der Frequenz. Dadurch können höhere Bandbreiten und somit höhere Auflösungen in Rangerichtung erreicht werden. Da diese Signale dem Zwitschern eines Vogels ähneln, nennt man sie Chirp-Signale.

2.1.4 SAR-Prozessierung

Am Ende der SAR-Datengewinnung steht ein Rohdatensatz. Aus diesen Signaldaten wird durch die SAR-Prozessierung ein SAR-Bild berechnet. Ein Bild einer synthetischen Apertur zu prozessieren ist weit komplizierter als die Prozessierung eines Bildes einer realen Apertur. Bei der Prozessierung werden die Daten einer entfernungsabhängigen zweidimensionalen Filterung unterworfen. Dabei wird das Signal eines Punktstreuers, das im Rohdatensatz in Chirpform über zwei Dimensionen auseinander gezogen wurde, wieder in die Form eines Punktes transformiert.

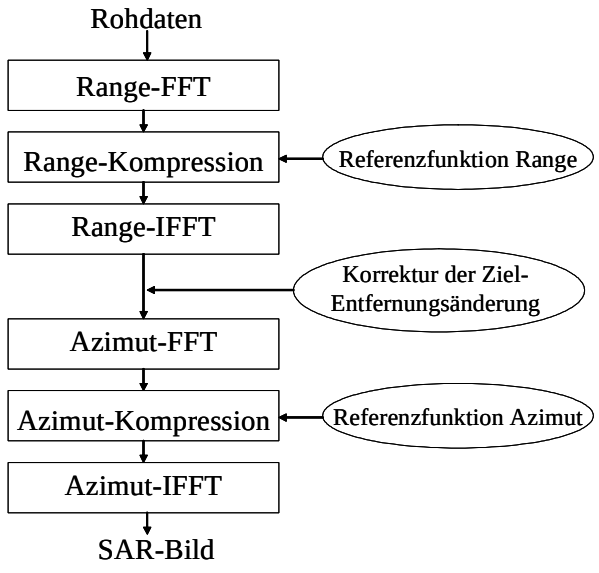

Abbildung 2-4: SAR-Prozessierung (Ulbricht, 2000)

In der Praxis reduziert man den Prozess auf eine Reihe von nacheinander durchgeführten Multiplikationen der einzelnen Range- bzw. Azimutzeilen mit der jeweiligen Referenzfunktion. Die Reflexion eines Punktziels kann durch eine zweidimensionale sinc-Funktion angenähert werden. Die geometrische Auflösung eines Bildes kann über die 3dB-Breite der Hauptkeule der sinc-Funktion einer Punktzielantwort ermittelt werden. Hierfür werden links und rechts des Maximums die Positionen bestimmt, an denen die Leistung um die Hälfte gefallen ist. Die Anzahl der dazwischen liegenden Punkte multipliziert mit der Größe der Punkte ergibt die geometrische Auflösung (Mittermayer, 2000).

2.2 Abbildungsgeometrie der SAR-Daten

Die Bilder eines SAR-Systems haben andere geometrische Eigenschaften als z.B. Kamerabilder mit zentralprojektiver Abbildungsgeometrie. Die Geometrie eines SAR-Bildes unterscheidet sich in Azimut- und Entfernungsrichtung. In der Entfernungsrichtung ist die Geometrie eines SAR-Bildes eine Laufzeitgeometrie, d.h. dass die Position eines Objektes im Bild durch die Laufzeit des Signals bestimmt wird. In Azimut wird die Position eines Objektes durch die Auswertung des Dopplereffekts in der SAR-Prozessierung bestimmt. Diese Unterschiede sind bei der Implementierung eines SAR-Simulators entscheidend, wie in Kapitel 4.3 beschrieben wird.

2.2.1 Abbildungsgeometrie in der Entfernungsrichtung

2.2.1.1 Abbildung dreidimensionaler Objekte

Die, im Vergleich zu Photographien, unterschiedliche Abbildungsgeometrie von SAR-Bildern führt bei der Interpretation zu Schwierigkeiten, da der Operator Bilder mit Zentralprojektion gewöhnt ist und diese Abbildungseigenschaften zunächst auch in SAR-Bildern erwartet. Wie in Abbildung 2-5 schematisch dargestellt, wird im Fall der Zentralprojektion (links) die Spitze des Hauses hinter der Geländeoberfläche abgebildet. Bei der Laufzeitgeometrie wird der obere Teil des Hauses vor dem unteren Teil abgebildet. Die Laufzeit, bzw. die Länge des Strahls entscheidet über die Positionierung im Bild. Bei der Berechnung der Position wird normalerweise von einem flachen oder gekrümmten Untergrund ausgegangen. Die Strecke vom Sensor zur Dachspitze C wird auf der Geländeoberfläche abgebildet und entspricht der Position C'. In Abbildung 2-5 (rechts) liegt C' zwischen B' und A', d.h. es gerät bei der laufzeitgeometrischen Abbildung dieses Gebäudes sogar die Reihenfolge der Punkte, im Vergleich zur gewohnten perspektivischen Abbildung, durcheinander. Einzig Punkt A ist richtig auf A' abgebildet. Die Abbildung von B' vor A' nennt man Layover. Der nicht beleuchtete Bereich ist der Radarschatten. Dieser erstreckt sich von A/A' bis D.

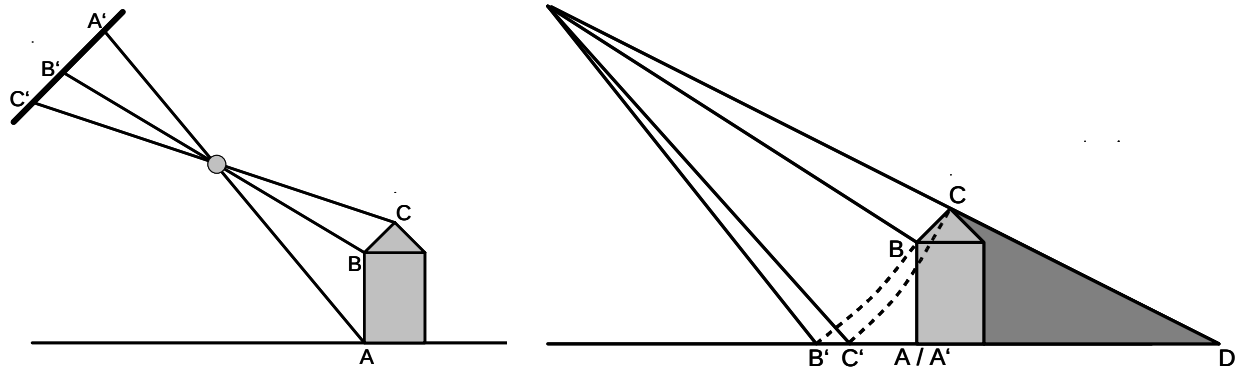

Abbildung 2-5: Unterschiedliche Abbildungsgeometrien

2.2.1.2 Darstellung als „Nadiraufnahme"

Ein weiterer wichtiger Unterschied ist die Abbildung der Daten des Seitensichtradars als „Nadirbild". Das System ist ein Seitensichtsystem. Dies wird allerdings in der resultierenden Abbildung nicht deutlich. Betrachtet man eine optische Schrägaufnahme wird schnell klar, in welche Bereiche auf Grund von Verdeckungen nicht eingesehen werden kann. Im SAR-Bild ist dies nicht sofort erkennbar, da die Daten als eine Art Nadirbild präsentiert werden. Oftmals können die verdeckten Bereiche durch den Radarschattenwurf erkannt werden, jedoch ist der Radarschatten nicht immer vom Hintergrund unterscheidbar.

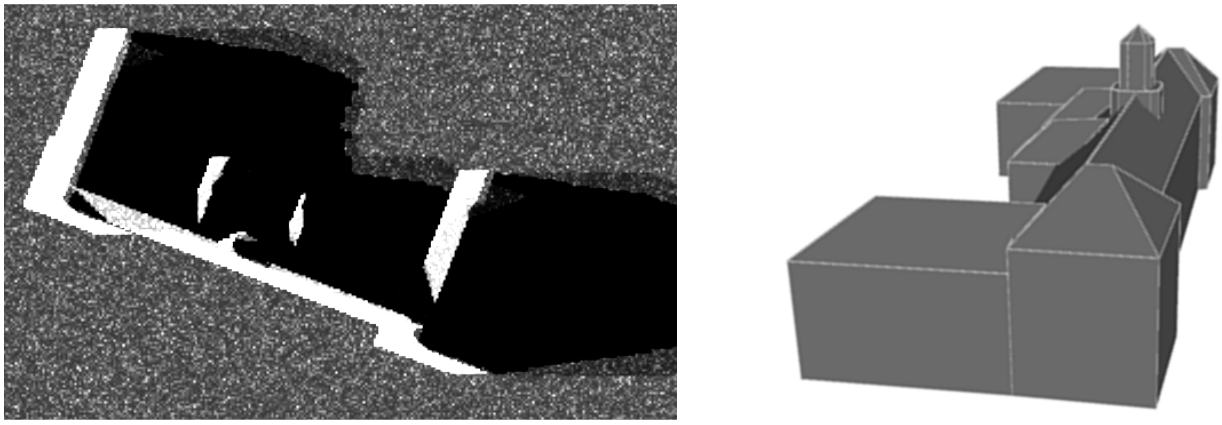

Abbildung 2-6: Vergleich zwischen einem simulierten SAR-Bild und der Schrägsicht aus der Position des Sensors entlang der Entfernungsrichtung auf das simulierte Gebäudemodell

In Abbildung 2-6 ist links ein simuliertes SAR-Bild eines Gebäudemodells abgebildet. Die Darstellung des rechten Modells zeigt den Blickwinkel unter dem das simulierte SAR-System das Gebäudemodell betrachtet. Rechts erkennt der Betrachter sofort: einige Teile des Gebäudes sind auf Grund der Schrägsicht verdeckt. Im Radarbild sind diese Bereiche ebenfalls verdeckt, doch sind diese Verdeckungen nicht sofort offensichtlich, da das Bild als eine Art Nadiraufnahme dargestellt wird.

2.2.1.3 Mehrdeutigkeiten in Entfernungsrichtung

In topographisch bewegtem Gelände und insbesondere durch hochragende Einzelobjekte wie Häuser oder Bäume kommt es in SAR-Bildern häufiger zu Mehrdeutigkeiten. Einzelne Pixel bestehen aus einer Mehrzahl von Streuern, so dass z.B. eine Baumspitze und die davor liegende Wiese gleichzeitig erfasst werden.

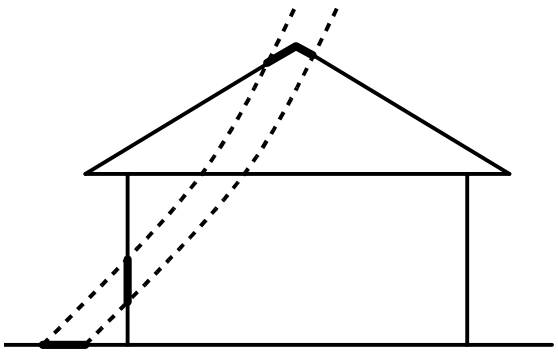

Abbildung 2-7: Mehrdeutigkeiten in Entfernungsrichtung

In Abbildung 2-7 ist diese Mehrdeutigkeit zu erkennen. Das Bildelement, zwischen den abgebildeten Linien gleicher Laufzeit, besteht aus mehreren Teilen: dem Dach, der Hauswand und der Fläche vor dem Haus. Diese Teile können nicht voneinander getrennt werden und addieren sich im resultierenden Bild zum Mischpixel.

2.2.2 Abbildungsgeometrie in Azimutrichtung

In Azimutrichtung unterscheidet sich die Abbildungsgeometrie von der Laufzeitgeometrie in Entfernungsrichtung. Die Azimutposition eines Streuers im resultierenden SAR-Bild wird durch die Dopplerverschiebung festgelegt. Die Abbildung in Azimut kann durch Eigenbewegungen des Streuers gestört werden. Wird z.B. ein Zug abgebildet und dieser Zug bewegt sich während der Aufnahme auf den Sensor zu oder vom Sensor weg, so verschiebt sich die Azimut-Position des abgebildeten Objekts. Die Eigenbewegung des Objektes führt zu einer zusätzlichen Frequenzverschiebung, so dass die Positionsbestimmung über den Dopplereffekt verfälscht wird.

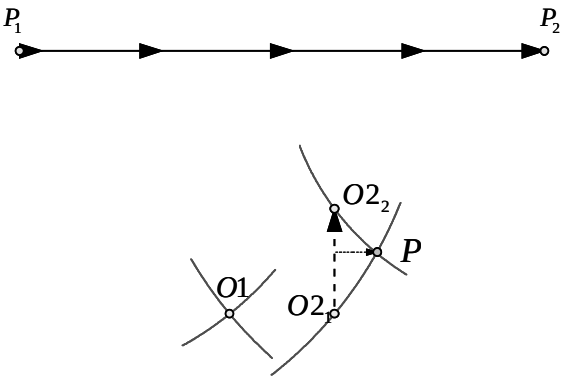

Abbildung 2-8: Verschiebung der Azimut-Position durch Eigenbewegung in Entfernungsrichtung

In Abbildung 2-8 wird die Verschiebung der abgebildeten Azimut-Position eines sich nur in Entfernungsrichtung bewegenden Objektes schematisch und vereinfachend über die synthetische Apertur erklärt. Während des Fluges werden in P_1 und P_2 zwei Einzelaufnahmen durch das System gemacht. Objekt *O1* bleibt in der Zeit zwischen den Aufnahmen in P_1 und P_2 stehen, während sich *O2* gleichmäßig auf den Sensor zu bewegt und zweimal an verschiedenen Positionen registriert wird: an $O2_1$ und $O2_2$. Die Schnittkreise gleicher Entfernung um P_1 und P_2 schneiden sich auf Grund der Eigenbewegung von *O2* aber nicht auf der Azimutposition von *O2* sondern in Azimutrichtung versetzt im Punkt *P*. Diese vereinfachte Darstellung ist jedoch für Objekte mit Bewegungskomponenten in Flugrichtung ungültig.

Die Größe des Versatzes Δy in Azimut hängt von der Geschwindigkeit des Sensors v_S, der Geschwindigkeit des Objektes v_O und der Entfernung vom Sensor zum Objekt d_s ab (siehe auch die Implementierung in Kapitel 4.3.4).

$$\Delta y = \frac{d_S \cdot v_O}{v_S} \tag{2.1}$$

(Schreier, 1993; Elachi, 1988)

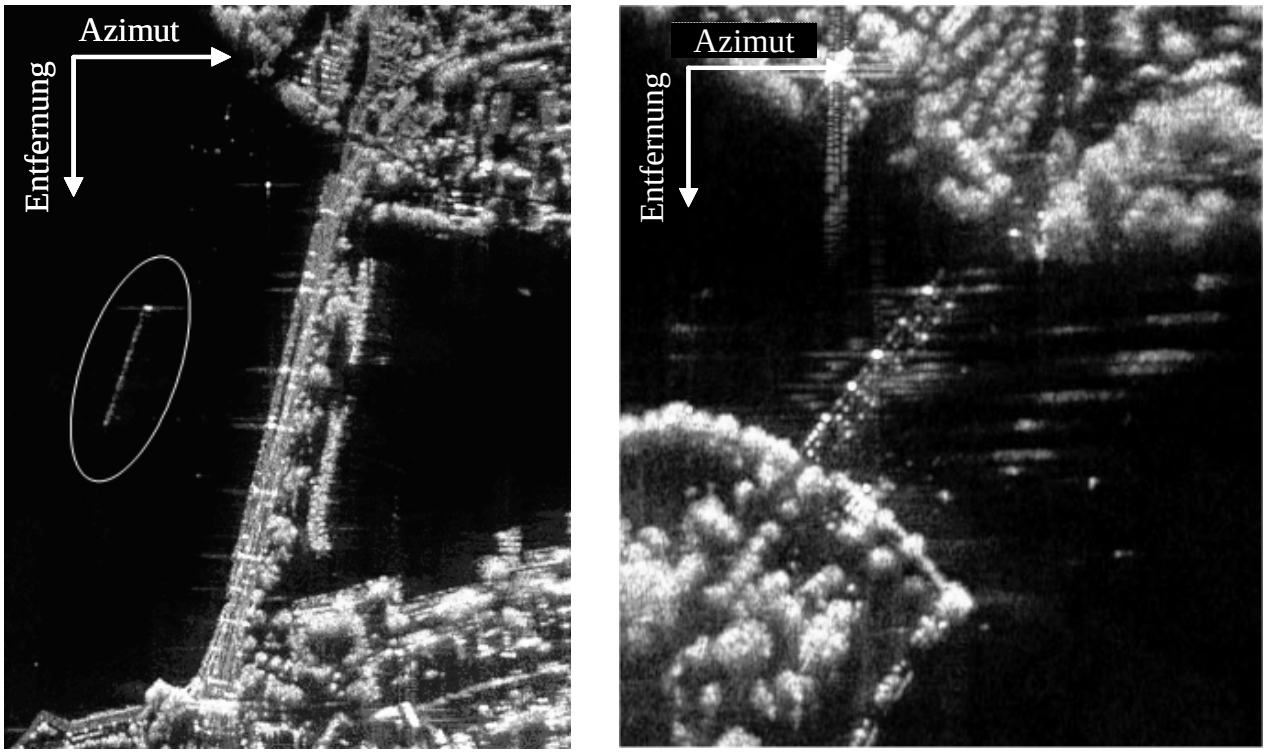

Abbildung 2-9: Verschiebung in Azimut durch Eigenbewegung der Objekte. Bild: DOSAR (EADS Deutschland)

In Abbildung 2-9 ist links deutlich die Verschiebung eines Zuges auf der Brücke von Lindau zu sehen. Der Zug bewegt sich vom Sensor weg und wird entgegen der Azimutrichtung verschoben. Die wirkliche Position des Objektes im Bild bleibt auf der Brücke leer und somit schwarz. Im rechten Bild sind verschiedene Autos zu sehen, deren Positionen im SAR-Bild ebenfalls in Azimutrichtung versetzt sind. Das Ausmaß des Versatzes ist, wie in Gleichung (2.1) gezeigt, abhängig von der Geschwindigkeit des Fahrzeuges.

Auf Grund der manchmal sehr langen Integrationszeit einer SAR-Aufnahme von mehreren Sekunden spielt die Eigenbewegung von Objekten eine große Rolle. Dies betrifft nicht nur sich fortbewegende Objekte wie Autos, Züge, usw., sondern z.B. auch Bäume, die sich im Wind bewegen. Durch die Eigenbewegungen dieser Objekte während der Aufnahme kommt es zu Verschiebungen, Verzerrungen und Verwischungen in Azimutrichtung. Je größer die Entfernung vom Sensor zum Objekt ist und je langsamer die Sensorplattform sich bewegt, desto höher ist die Integrationszeit und desto größer ist dieser Effekt.

In Abbildung 2-10 ist die Verzerrung der Bäume in Azimut durch die Eigenbewegung der Bäume im Wind zu erkennen. An Hand des fokussierten Feldes im rechten Bildbereich und der Straßenmitte ist klar erkennbar, dass das Bild scharf ist. Der Waldbereich ist allerdings unscharf und es sind deutlich Verwischungen erkennbar, die durch die Eigenbewegung der Bäume verursacht werden.

Abbildung 2-10: Verzerrung in Azimut durch Eigenbewegung der Bäume. Bild: DOSAR (EADS Deutschland)

2.2.3 Zwei- und Mehrfachreflexionen

Wird das Signal auf dem Weg zwischen Sensor-Objekt-Sensor mehrfach reflektiert, so ändert sich die Laufzeit des Signals und damit die im SAR-Bild abgebildete Position des Objektes. Typisches Beispiel hierfür ist die Zweifachreflexion. Bei zwei Ebenen die sich im 90° Winkel schneiden, spricht man von einem dihedralen Corner-Reflektor. Durch die senkrechte Anordnung kommt es zu Zweifachreflexionen. Schneiden sich drei Ebenen, spricht man von einem trihedralen Corner-Reflektor. Durch die starke Rückstreuung dieser Reflektoren bilden sich Nebenkeulen, die die benachbarten Bereiche sternförmig überstrahlen können. Diese Cornerreflektoren lassen sich, auf Grund dieser starken Reflexion, selbst dann detektieren, wenn die Größe des Objektes deutlich unter der geometrischen Auflösung des SAR-Systems liegt. Deswegen werden bei Flug- und Messkampagnen Corner-Reflektoren zur radiometrischen Kalibrierung (Zink, 1993) und zur Georeferenzierung verwendet.

Mehrfachreflexionen sind seltener zu beobachten. Bei Flussbrücken treten, auf Grund des stark spiegelnden Effektes der Wasseroberfläche, Zwei-, Drei-, und mehr Reflexionen auf (siehe Lee et al, 2006). Bei großen metallischen Objekten, wie z.B. Schiffen, treten oftmals Mehrfachreflexionen auf. Die Zwei- und Mehrfachreflexionen sind nicht nur ein Grund für geometrische Verschiebungen im SAR-Bild, sie beeinflussen auch massiv die Radiometrie der SAR-Bilder.

2.3 Radiometrie der SAR-Daten

Die für die Radarsimulation wichtige Radargleichung dient der Berechnung der Reflexion flächenhafter Ziele (siehe Anhang A.3):

$$P_R = \iint_F \frac{P_S \cdot G^2 \cdot \lambda^2}{(4\pi)^3 \cdot r_O^4} \cdot \sigma^\circ \qquad (2.2)$$

Die vom Radar empfangene Energie P_R hängt von der ausgestrahlten Leistung P_S, dem Antennengewinn G, der Entfernung zwischen Sensor und Ziel r_O, der Wellenlänge λ und dem Radarrückstreukoeffizienten σ° ab. Der Dynamikbereich des Radarrückstreukoeffizienten σ° ist sehr groß. Typische Szenen können, wie in Abbildung 2-11 zu sehen, 50 dB umfassen. Dieser Dynamikbereich kann mit 32-bit Daten abgedeckt werden. Typischerweise werden für die Bildauswertung im SAR-Bereich 16-bit Daten benutzt, was ohne entscheidenden Informationsverlust möglich ist.

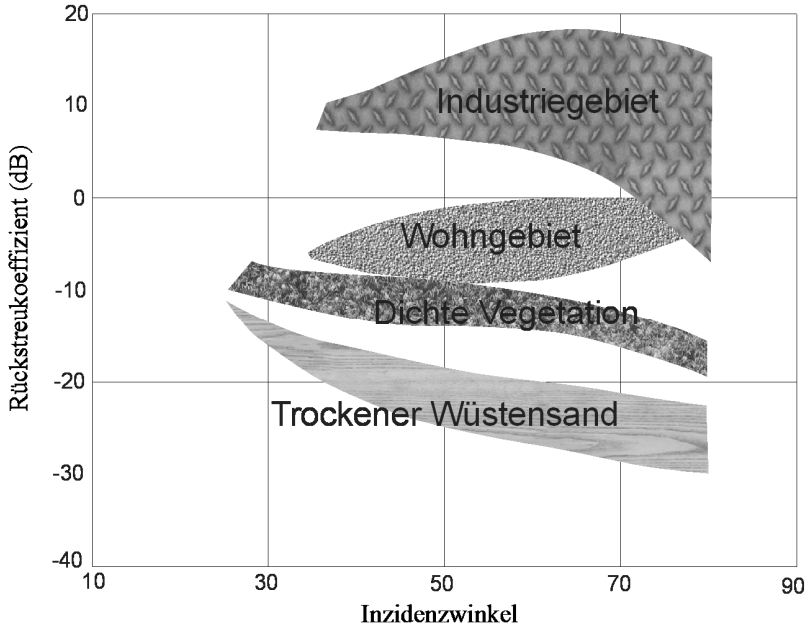

Abbildung 2-11: Reflektivität verschiedener Oberflächen (nach Newbry, 1960)

2.3.1 Einflüsse der Streueigenschaften auf die Radiometrie

Die Helligkeit eines Objektes im Bild wird, neben den vom Sensor vorgegebenen Faktoren und der Entfernung zwischen Sensor und Ziel, durch den Radarrückstreuquerschnitt σ bestimmt. Dieser ist für die Berechnung der Rückstreuung in der SAR-Simulation eine sehr wichtige Größe (siehe Kapitel 4.4.1). Der Radarrückstreuquerschnitt hängt ab vom:

- Blickwinkel des Systems
- der Rauigkeit der Objektoberfläche im Vergleich zur verwendeten Wellenlänge
- den elektrischen Eigenschaften des Objektes
- dem Rückstreuverhalten des Objektes
    - Oberflächenstreuer
    - Volumenstreuer
- der Polarimetrie der gesendeten und empfangenen Welle
- der Neigung des Objektes im Vergleich zur Blickrichtung des Systems

Die Wellenlänge bestimmt, ob ein Objekt als Oberflächenstreuer oder als Volumenstreuer auftritt. Oberflächenstreuer reflektieren die Energie an ihrer Oberfläche, während Volumenstreuer die Energie im Objektkörper reflektieren.

2.3.1.1 Einfluss der Oberflächenrauigkeit

Die Rauigkeit der Oberfläche hat große Auswirkungen auf die Rückstreuung und damit auf die Helligkeit eines Objektes im Bild. Sie wird in Abhängigkeit von der Wellenlänge betrachtet. Bei kleineren Wellenlängen, wie z.B. im X-Band (3cm), ist ein Acker sehr rau, während derselbe Acker für ein Signal im langwelligen P-Band (60 cm) glatt ist und wie ein Spiegel wirkt. Nach einer Faustregel gilt eine Fläche als eben, wenn für Δh gilt (Klausing & Holpp, 2000):

$$\Delta h < \frac{\lambda}{8 \cdot \cos\theta} \tag{2.3}$$

Δh ist die Höhenvariation und θ ist der Winkel zwischen der Flächennormalen und der Beleuchtungsrichtung. Spiegelnde Flächen strahlen auf Grund der Schrägsicht nichts an den Sensor zurück, sondern reflektieren den Radarstrahl vom Sensor weg, weswegen diese Bereiche in SAR-Bildern schwarz sind. Bei sehr rauen Oberflächen entspricht die Rückstreuung dem Lambertschen Kosinusgesetz, d.h. das Objekt reflektiert in alle Richtungen und die empfangene Signalstärke hängt vom Kosinus des Einfallswinkels θ ab.

$$P_R = P \cdot \cos\theta \tag{2.4}$$

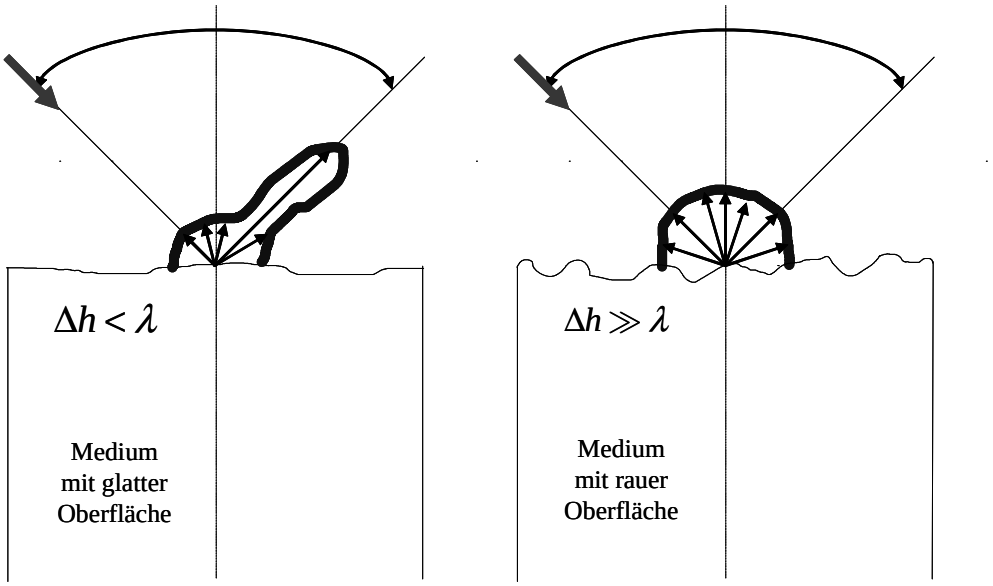

Abbildung 2-12: Schematische Darstellung von Oberflächenstreuern unterschiedlicher Rauigkeit (nach Klausing & Holpp, 2000)

2.3.1.2 Die elektrische Leitfähigkeit

Die dielektrische Konstante ε ist ein Maß für die elektrische Leitfähigkeit eines Objektes und bestimmt damit auch das Verhältnis von Transmission und Reflexion. Niedrige dielektrische Konstanten führen zu einer stärkeren Transmission und somit zu einer schwächeren Reflexion, während höhere dielektrische Konstanten zu einer stärkeren Reflexion bei geringerer Transmission führen. Oberflächen mit einer hohen dielektrischen Konstante reflektieren die Energie vollständig, was je nach Einfallswinkel zu einer vollständigen Rückstreuung der Energie an den Sensor oder zu keiner Rückstreuung zum Sensor hin führt.

2.3.1.3 Einfluss der Umgebung

Die Umgebung eines Punktes hat Auswirkungen auf die von diesem Punkt zurückgestreute Energiemenge. So können Schattenwürfe durch umgebende Objekte die Rückstreuung des Punktes verhindern. Objekte, die größer als eine Auflösungszelle, d.h. größer als die Auflösung des Systems sind, können über ihre Textur im SAR-Bild identifiziert werden. So können z.B. Waldflächen in SAR-Bildern auf Grund ihrer ausgeprägten variablen Textur erkannt werden. In sehr hoch aufgelösten SAR-Bildern sind selbst einzelne Bäume anhand ihrer speziellen Textur erkennbar.

Auch die Topographie hat einen starken Einfluss auf die Radiometrie. Ein Bereich, der zum Sensor hin geneigt ist, ist deutlich heller als ein vom Sensor abgeneigter Bereich. In Abbildung 2-13 ist dies dargestellt. Je mehr Hangfläche innerhalb des Bereiches gleicher Laufzeit Δr beleuchtet wird, desto größer ist die von dieser Auflösungszelle zurückgestreute Energie. Das SAR-Bild ist dementsprechend heller in Hangbereichen die dem Sensor zugewandt sind, als in ebenen Bereichen.

Dieses Wissen wird in der Radarklinometrie verwendet. Ausgehend von der Annahme, dass alle Pixel eines Bildes die gleichen Reflexionseigenschaften haben und nur die Topographie für die Helligkeitsunterschiede im SAR-Bild verantwortlich ist, kann aus den Pixelwerten die Steigung berechnet werden. Zwar ist die Annahme, dass alle Bereiche eines Bildes die gleichen Reflexionseigenschaften aufweisen, stark vereinfacht, in einigen Fällen kann diese Annahme jedoch getroffen werden. Diese Methode wurde z.B. in der Weltraumfernerkundung bei der Kartierung der Venus angewandt (Wildey, 1986).

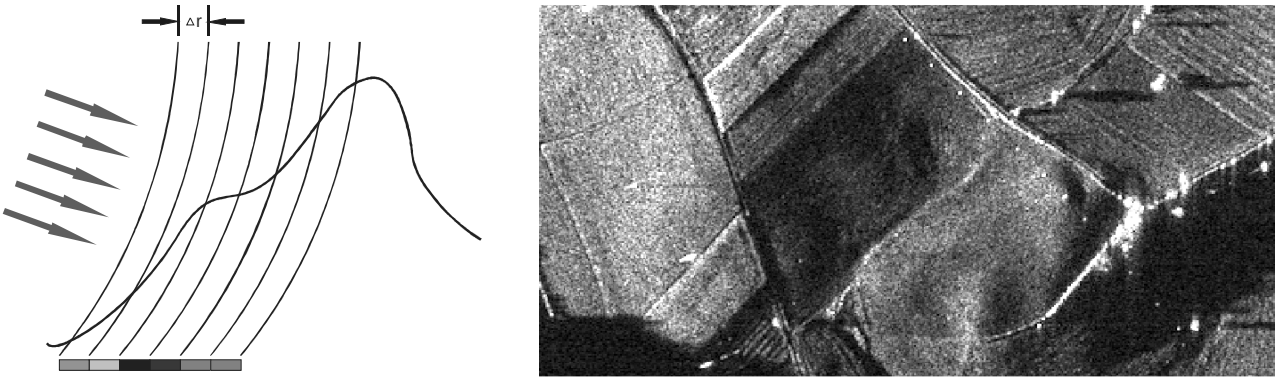

Abbildung 2-13: Links: Radiometrische Veränderungen durch die Topographie. Rechts: DOSAR-Beispielbild (EADS Deutschland)

2.3.1.4 Volumenstreuung

Neben der Oberflächenstreuung kommt es, in Abhängigkeit von der Wellenlänge und den Materialeigenschaften, zu einer Volumenstreuung. Langwellige Strahlung kann tiefer in Objekte eindringen oder diese sogar durchdringen. So wird eine Welle im X-Band größtenteils an der Oberfläche eines trockenen Bodens gestreut, wohingegen eine Welle im P-Band tief in den Boden eindringen kann und dort der Volumenstreuung unterliegt. Im Wald wird das kurzwelligere X-Band im oberen Bereich der Baumkronen reflektiert, während P-Band-Wellen diesen Bereich größtenteils durchdringen und erst am Boden oder von den Stämmen reflektiert werden. Dieser Effekt kann bei der Multifrequenzinterferometrie ausgenutzt werden, um den Höhenunterschied zwischen der Baumkrone und dem Boden zu messen.

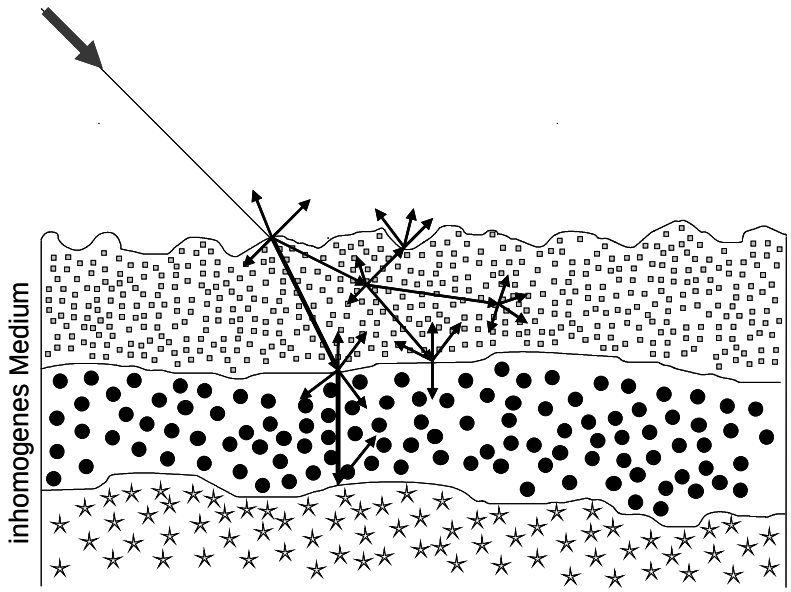

Abbildung 2-14: Schematische Darstellung eines Volumenstreuers

Volumenstreuung wird hauptsächlich durch Ungleichheiten der Dielektrizität im Volumen ausgelöst. Natürliche Oberflächen sind normalerweise inhomogen, obwohl sie in Abhängigkeit von Wellenlänge und Blickrichtung als homogen betrachtet werden können. So ist Wasser im Mikrowellenbereich homogen und unterliegt nahezu ausschließlich der Oberflächenstreuung, während es im optischen Bereich, in Abhängigkeit der Blickrichtung auf die Wasseroberfläche, der Volumenstreuung unterliegt. Eine wichtige Objektart mit Volumenstreuung ist die Vegetation. Bewachsene Flächen unterliegen im Mikrowellenbereich fast immer der Volumenstreuung, im langwelligen Bereich mehr, im kurzwelligen Bereich weniger stark.

2.3.2 Radiometrische Mehrdeutigkeiten

Eine Schwierigkeit bei der Auswertung von Radardaten ist die Mehrdeutigkeit und die komplexen Zusammenhänge, die zu einer bestimmten Rückstreuintensität im Bild führen können. Eine gemessene Rückstreuintensität lässt sich nicht auf eine einzige Ursache zurückführen, sondern ist das Ergebnis eines komplexen Zusammenspiels verschiedener Ursachen.

Eine Reihe von Mehrdeutigkeiten bestimmt die Radiometrie von SAR-Daten. Die Rückstreuung kleiner metallischer Objekte kann z.B. weit stärker sein, als die von großen natürlichen Objekten. Die Form und das Material sind, neben der Ausrichtung des Objektes zum Sensor, ausschlaggebend. Bei kleinen, kantigen metallischen Objekten kann bereits eine kleine Änderung der Ausrichtung eine große Änderung in der Rückstreuintensität bedeuten. Dies kann im Extremfall von der höchsten Rückstreuintensität im Falle einer zum Sensor hin spiegelnden Reflexion, bis zum Ausbleiben jedweder Rückstreuung durch eine Spiegelung weg vom Sensor, reichen. Extrem starke Rückstreuungen von einzelnen Objekten können durch ihre Nebenkeulen die Rückstreuung benachbarter Objekte überlagern.

2.3.3 Speckling

SAR-Bilder sind meist gekennzeichnet durch ein Specklemuster, das SAR-Bilder verrauscht erscheinen lässt. Bei kohärenter Beleuchtung einer rauen Oberfläche entstehen Interferenzen, die im Fernfeld als körnige Struktur sichtbar werden. Das Speckling entsteht durch statistische Fluktuation und Interferenzen der Rückstreuung der einzelnen Streuer innerhalb einer Auflösungszelle. In der Fläche *A* einer Auflösungszelle befinden sich *N* Streuer, mit:

$$N \approx \frac{A}{\lambda^2} \tag{2.5}$$

Die Rückstreuung der Zelle ist eine Mischung aus der Rückstreuung der *N* Streuer. Das Verhalten eines Streuers kann als Punktzielantwort aufgefasst werden. Eine solche Punktzielantwort ist Abbildung 2-15 auf der linken Seite dargestellt. Die Echos vieler einzelner Streuer können sich gegenseitig überlagern und sich gegenseitig verstärken oder auslöschen. Dies ist in Abbildung 2-15 schematisch dargestellt. Die simulierte Fläche in Abbildung 2-15 ist unterteilt in 25 Auflösungszellen. Jede dieser Auflösungszellen enthält 20 gleichstarke Streuer, die zufällig innerhalb der Zelle verstreut sind. Jeder einzelne Streuer muss einen Mindestabstand zu seinen Nachbarn einhalten. Durch die gegenseitigen Überlagerungen und Verstärkungen reflektieren einzelne Auflösungszellen im Mittel mehr Energie zurück als andere.

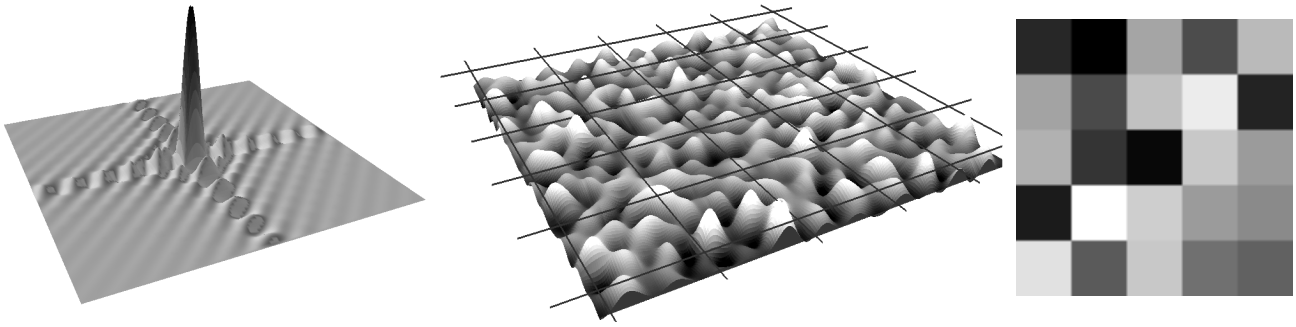

Abbildung 2-15: Darstellung einer zweidimensionalen sinc-Funktion (links) und des Speckling (Mitte und rechts)

Diese Schwankungen sind mehr oder weniger zufällig und daher kann das Speckling vereinfachend als Zufallsrauschen betrachtet werden. Dies wird in Kapitel 4.4.6 näher erläutert. Besteht eine Auflösungszelle aus nur einem oder wenigen dominanten Streuer, dann kommt es zum Speckling mit dominanten Streuern. *N* ist dann nicht mehr groß, sondern wird selbst zufällig. Dieser Effekt tritt vermehrt bei sehr hoch aufgelösten SAR-Aufnahmen auf.

2.3.3.1 Multilook-Aufnahmen

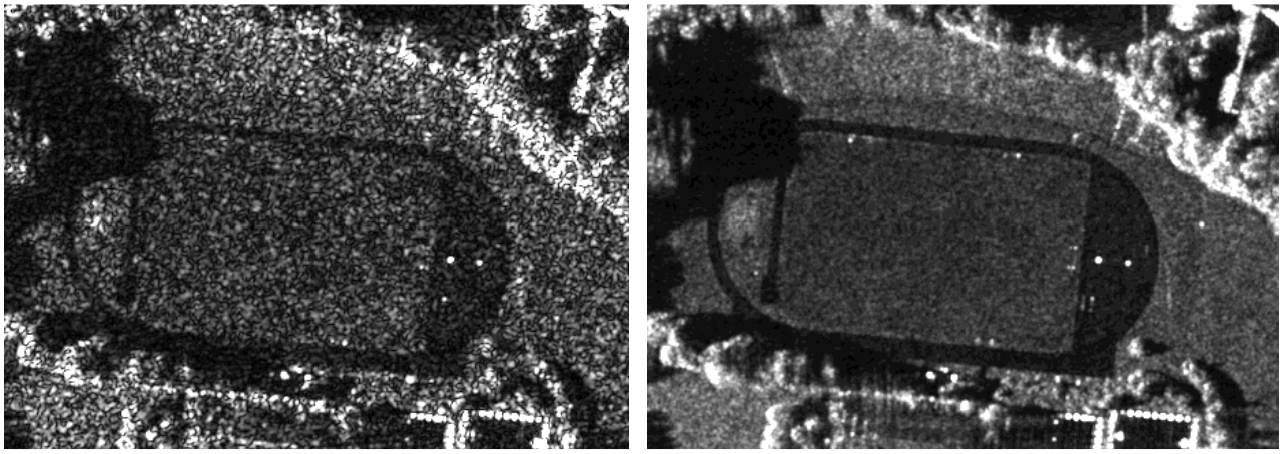

Abbildung 2-16: Singlelook (links) im Vergleich zu Multilook (rechts). Bilder: DOSAR (EADS Deutschland)

Wie in Abbildung 2-16 zu sehen, lässt sich das Speckling durch Aufnahmen im Multilook-Verfahren reduzieren. Dazu wird die Gesamtlänge der synthetischen Apertur in mehrere kleine Aperturteile unterteilt, wie z.B. in Abbildung 2-16 in 51 Teile. Das Multilookbild rechts hat daher 51 sogenannte Looks und entspricht einem Mittelwertsbild der einzelnen Aufnahmen.

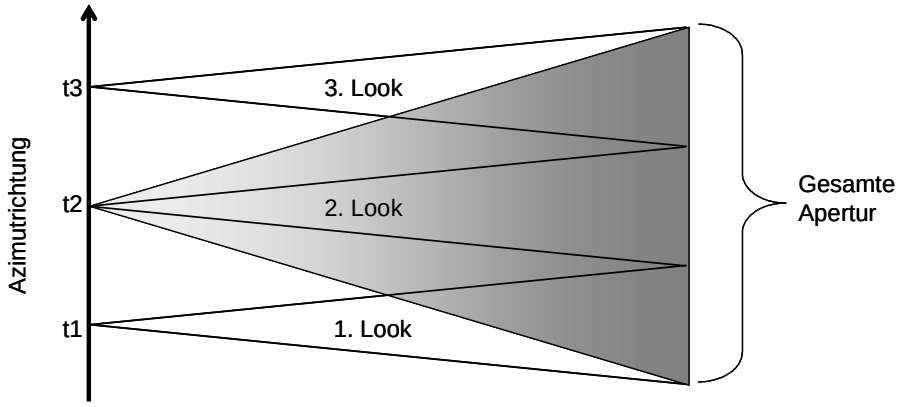

Abbildung 2-17: Schematische Darstellung des Multilook

Bei Multilook-Aufnahmen wird zwar die radiometrische Auflösung erhöht, man erkauft sich dies jedoch mit einer geringeren räumlichen Auflösung in Azimutrichtung, da nicht die komplette Länge der synthetischen Apertur ausgenutzt wird. Die mögliche Gesamtlänge wird auf die einzelnen Looks aufgeteilt, so dass jedem Look nur eine verkürzte Aperturlänge zur Verfügung steht. Die Länge der synthetischen Apertur wird um den Faktor *1/M* kleiner, wobei *M* für die Anzahl der unabhängigen Looks steht. Erfahrungsgemäß liegt für die Auswertung die optimale Anzahl unabhängiger Looks bei *M=3* (Ulaby et al, 1982).

2.4 Radarpolarimetrie

Die Ausrichtung eines Objektes zum Sensor oder auch dessen Materialeigenschaften beeinflussen die Polarisation der elektrischen Welle. Polarimetrische Radarsysteme sind in der Lage, elektromagnetische Wellen entweder mit horizontaler (H) oder vertikaler (V) Polarisation zu senden oder zu empfangen. In der Indexschreibweise steht die ausgestrahlte Polarisation einer Welle an erster, die empfangene Polarisation einer

Welle an zweiter Stelle. HV bedeutet, dass die Welle in horizontaler Polarisation ausgestrahlt, aber in vertikaler Polarisation empfangen wird. In einigen Veröffentlichungen ist auch die umgekehrte Schreibweise zu finden. Bei der Auswertung betrachtet man die komplette Streumatrix:

$$[S] = \begin{bmatrix} S_{HH} & S_{HV} \\ S_{VH} & S_{VV} \end{bmatrix} \tag{2.6}$$

Die unterschiedlichen Streuarten (geradzahlige Streuung, ungeradzahlige Streuung, diffuse Streuung) können zur Klassifikation von Bodennutzungsarten verwendet werden (van Zyl, 1989). Die Analyse der Polarimetrie kann auf unterschiedliche Arten erfolgen (Cloude & Pottier, 1996) und bietet eine Vielzahl weiterer Anwendungsmöglichkeiten in der Fernerkundung. Mit sehr hochauflösenden voll-polarimetrischen SAR-Systemen kann in Zukunft eine genauere Analysemöglichkeit der einzelnen Objekte und ihrer polarimetrischen Eigenschaften erfolgen, wodurch die Analyse- und Auswertemöglichkeiten der SAR-Systeme erweitert werden.

2.5 Bistatisches SAR

Unter bistatischem Radar versteht man ein Radarsystem, dessen Antennen sich für die Ausstrahlung und den Empfang sich an unterschiedlichen Positionen befinden. Die Winkel und Entfernungen der Antennen zum Ziel müssen sich signifikant unterscheiden (IEEE, 1998). Die Entfernung, die von der IEEE nicht näher definiert wurde, sollte nach Skolnik (2001) „vergleichbar“, nach Blake (1986) „ein signifikanter Bruchteil“ entweder der Entfernung vom Sender zum Ziel, oder der Entfernung vom Empfänger zum Ziel, sein. Die Interferometrie fällt nach dieser Definition nicht unter den Begriff des bistatischen SAR.

In Abbildung 2-18 ist das bistatische Dreieck dargestellt. Dieses liegt zwischen dem Sender *S*, Ziel *T* und dem Empfänger *R* im Raum. Der Abstand zwischen Sender und Empfänger ist die Baseline *L*, der Winkel ist der bistatische Winkel β.

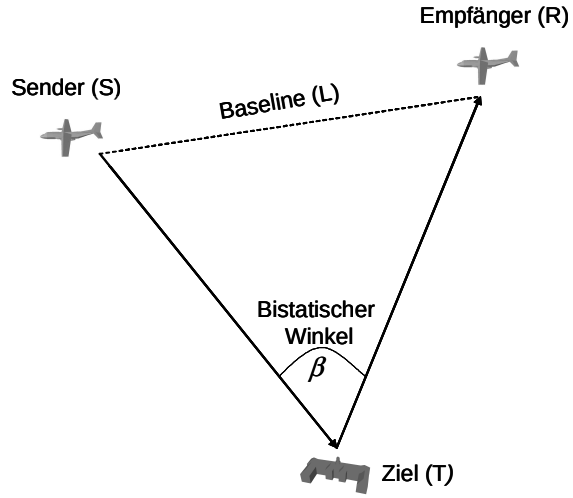

Abbildung 2-18: Bistatische SAR-Geometrie

Bistatisches Radar hat eine lange Geschichte und dient vor allem militärischen Anwendungen, z.B. der Tarnung des Empfängers oder der parasitären Ausnutzung unkooperativer oder feindlicher Sender zur eigenen Radaraufklärung. Bistatisches Radar war und ist jedoch nur ein Nischenprodukt im Vergleich zum monostatischen Radar. Dies liegt nach *Willis* (1991) vor allem an:

1. der hohen Komplexität des bistatischen Radars
2. den hohen Kosten
3. den anderen Lösungen, die mit im Wettkampf um die knappen Finanzierungsmittel stehen
4. den schlechteren Leistungen
5. der geringen Anzahl von Tests und verfügbaren Daten
6. der Unterschätzung oder der Unterbewertung möglicher Gefahren für die monostatischen Systeme
7. der monostatischen Denkweise und Mentalität der Entscheidungsträger

Neuerdings gewinnen bistatische Radarsysteme wieder an Bedeutung (Krieger & Moreira, 2006), insbesondere als Gegenmaßnahme zu elektronischen Störmaßnahmen oder den Stealth-Technologien. Stealth-Flugzeuge oder Schiffe versuchen ihre Radarrückstreuung gering zu halten und die Reflexion vom Sender wegzustreuen, wodurch monostatische Systeme nichts mehr empfangen können. Diese Maßnahme kann bei bi- und multistatischen Systemen jedoch fehlschlagen. Bistatisches SAR soll sowohl die Empfindlichkeit der Aufklärungssysteme gegen elektronische Gegenmaßnahmen verringern, als auch die Bekämpfung des SAR-Systems erschweren (Yates et al, 2006). Hierfür kann z.B. der SAR-Sender in großem Abstand zum Ziel positioniert werden, wohingegen der oder die SAR-Empfänger nahe am Aufklärungsziel operieren können. Da der Empfänger selbst passiv ist, ist dessen Bekämpfung schwierig.

Für die zivile Anwendung bieten bistatische SAR-Systeme ebenso Vorteile. Die ausgestrahlten Mikrowellen eines großen Sendersatelliten, evtl. geostationär, könnten von einer Reihe passiver Kleinstsatelliten empfangen und ausgewertet werden. Dies könnte bei sinkenden Gesamtkosten eine Vielzahl von Satellitenkonstellationen und Auswertungsmöglichkeiten bieten (Keydel, 2003; Krieger et al, 2004). Die Entwicklung der bistatischen SAR-Systeme ist bereits weit fortgeschritten. Erste Beispiele wurden von *Auterman* (1984) publiziert. In den letzten Jahren haben in Europa z.B. ONERA/DLR (Dubois-Fernandez et al, 2004; Cantalloube et al, 2004), FGAN-FHR (Walterscheid et al, 2004; Ender et al, 2006b) und QinetiQ (Yates et al, 2006) Testflüge mit bistatischen Systemen unternommen. FGAN, ZESS und FOMAAS planen ein bistatisches Experiment (TerraPAMIR) mit dem raumgestützten TerraSAR-X als Sender und dem flugzeuggetragenen PAMIR als Empfänger (Ender et al, 2006a; Gebhardt et al, 2006).

Die bistatische SAR-Geometrie und Radiometrie unterscheiden sich vom monostatischen Fall (siehe auch Kapitel 4.5). Auffällig ist das vermehrte Auftreten von Schattenbereichen im bistatischen Bild, da jedes beleuchtete Objekt sowohl zum Sender als auch vom Empfänger unverdeckt sein muss. Weiterhin fallen die verminderten Cornerreflexionen auf. Auch die Bildgeometrie verändert sich. So hängt z.B. der Layovereffekt im bistatischen Fall sowohl von der Blickrichtung des Senders als auch von der des Empfängers ab.

2.6 Verfügbare und zukünftige SAR-Sensorsysteme

SAR-Systeme sind sowohl kommerziell als auch militärisch, sowohl flugzeug- als auch satellitengestützt im Einsatz. Gerade die satellitengestützten Systeme werden in sehr naher Zukunft Daten mit einer stark verbesserten Auflösung sowohl für militärische (SAR-Lupe) als auch für kommerzielle (Radarsat-2, TerraSAR-X) Zwecke liefern. Bei der von DLR, EADS Astrium und Infoterra geplante Partnermission TanDEM-X (Moreira et al, 2004) ist der Start eines zweiten SAR-Satelliten geplant. Dieser soll mit TerraSAR-X zusammenarbeiten und gemeinsam innerhalb von drei Jahren ein hoch aufgelöstes digitales Geländemodell der Erdoberfläche erzeugen.

Eine bemerkenswerte neue Entwicklung im Bereich der flugzeuggestützten Systeme ist das Experimentalsystem PAMIR (Ender & Brenner, 2003) der FGAN-FHR. Damit werden Auflösungen im Dezimeterbereich erreicht. SAR-Bilder dieser hohen Auflösung bieten eine Vielzahl von neuen Möglichkeiten aber auch von neuen Problemen bei der Auswertung (Soergel et al, 2006). Insbesondere im militärischen Bereich finden unbenannte Drohnen bei der Aufklärung immer weitergehende Verwendung. Der Einsatz dieser UAVs (unmanned aerial vehicle) kann dazu beitragen, die Aufklärungsdauer im Einsatzgebiet zu verlängern und die zur Informationsgewinnung benötigte Zeitspanne zu verringern. Das Global-Hawk-System der Firma Northrop Grumman kann z.B. bis zu 32 Stunden nonstop im Einsatzgebiet verbleiben und dank direktem Datendownlink in dieser Zeit nahezu kontinuierlich Daten aus dem Aufklärungsgebiet senden (Stacy et al, 2002).

In Zukunft ist eine weitere Zunahme an verfügbaren SAR-Daten und an Anwendern die auf die Auswertung dieser Daten angewiesen sind, zu erwarten. Hierfür werden neue Werkzeuge benötigt. In dieser Arbeit wird ein solches Werkzeug entworfen. Die SAR-Effektvisualisierung basiert auf Methoden der Computergrafik und wendet diese für die Radarfernerkundung an. Durch diese Verknüpfung entsteht eine neue Anwendung, die es erstmals erlaubt SAR-Effekte in Echtzeit darzustellen.

3 3D-Computergrafik

Seit dem ersten kommerziellen Computerspiel Pong von 1972, bis hin zu den modernen, nahezu fotorealistischen Spielen in komplexen Welten, hat die Computerspielindustrie und mit ihr die Computergrafikindustrie eine stürmische Entwicklung beschritten. Dabei spielt die Hardwareentwicklung eine entscheidende Rolle. Hardwarebeschleunigte Grafikkarten erreichen heute ein Vielfaches der Leistung von professionellen Grafik-Workstations vergangener Tage zu einem Bruchteil des Preises.

Daneben wurde auch die Software weiter verbessert und ermöglicht es, komplexe 3D-Berechnungen und Darstellungen vergleichsweise einfach, schnell und flexibel zu programmieren. Realistische Szenen auf dem Computerbildschirm darzustellen gehört zu den rechenintensiven Anwendungen. Eine Vielzahl von Methoden wurden und werden entwickelt, um die Darstellung sowohl schneller als auch realistischer zu machen.

Abbildung 3-1: "The Parthenon". Visualisierung des virtuellen Modells des Parthenons in Griechenland (Debevec, 2004)

Moderne Grafikkarten erlauben die nahezu photorealistische Visualisierung von Daten in Echtzeit. Die in Abbildung 3-1 gezeigte Visualisierung des Parthenon basiert auf einem virtuellen Modell, welches aus Laserscandaten und der Oberflächenreflektivität gewonnen wurde (Debevec, 2004). Dieses Modell dient der Firma ATI zur Demonstration der Leistungsfähigkeit ihrer Grafikkarten.

In diesem Kapitel werden die für die Visualisierung von Daten und damit die für die SAR-Simulation in Echtzeit bedeutenden Techniken und Entwicklungen in der Computergrafik aufgezeigt. In der 3D-Computergrafik werden 3D-Modelle entsprechend den Reflexionseigenschaften ihrer Oberfläche und dem Abbildungsmodell visualisiert. Dies entspricht der Vorgehensweise bei der SAR-Simulation, jedoch unterscheiden sich die Reflexionseigenschaften und das Abbildungsmodell im Mikrowellenbereich fundamental. Deswegen müssen die Methoden angepasst werden. Im Folgenden wird zunächst das in der SAR-Simulation üblicherweise verwendete Ray-Tracing kurz beschrieben. Danach folgt ein kurzer Überblick über die Entwicklung der Grafikkarten, gefolgt von der für die Echtzeitvisualisierung in der Computergrafik üblicherweise verwendeten Rasterisierung. Abschließend werden einige für die SAR-Simulation in Echtzeit wichtige Neuerungen und Entwicklungen im Bereich der Hard- und Software beschrieben.

3.1 Ray-Tracing

Durch Ray-Tracing können, auf Grund der nahen Anlehnung des Verfahrens an den realen Abbildungsprozeß, Objekte realistisch visualisiert werden. Neben den zu visualisierenden Daten und ihren Eigenschaften, wie z.B. der Reflexionseigenschaft, werden eine oder mehrere Lichtquellen definiert. „Alle dadurch entstehenden optischen Phänomene, wie z.B. Schattenwurf oder die Spiegelung der Umgebung in einer glatten Oberfläche, werden durch eine einzige Methode bzw. Funktion berechnet, deren verblüffend einfache Struktur aber mit hohem Rechenaufwand erkauft ist.“ (Haenselmann, 1996)

Beim Versuch der Abbildung der Realität durch das Ray-Tracing, geht man davon aus, dass Lichtquellen Lichtstrahlen aussenden, die dann im Auge abgebildet werden. Diese Interpretation des Phänomens Licht wird beim Ray-Tracing simuliert. Um die Rechenzeit möglichst gering zu halten, sollten möglichst wenig Strahlen betrachtet werden, also möglichst nur die zur Visualisierung der Szene wichtigen Strahlen. In den meisten Implementierungen wird daher die Strahlrichtung umgedreht und der Strahl wird vom Auge ausgehend verfolgt, wodurch wirklich nur die Strahlen berechnet werden, die im Blickfeld des virtuellen Betrachters liegen. Für jeden Strahl wird der am nächsten liegende Schnittpunkt des Strahls mit den Objekten der Szene berechnet. Der Schnittpunkt ist beleuchtet, wenn der Strahl zu einer der Lichtquellen nicht von einem anderen Objekt blockiert wird. Vom Schnittpunkt können auch weitere Strahlen reflektiert werden, die ihrerseits wieder Schnittpunkte mit Objekten haben können

Ray-Tracing eignet sich besonders für die Beleuchtungsberechnung von spiegelnden Oberflächen. Für die Berechnung von diffusen Oberflächen wird zumeist die Radiosity (Goral et al, 1984; Ashdown, 1994) berechnet. Sinnvoll ist die Kombination beider Methoden (Heckbert, 1990). Daneben gibt es noch eine Vielzahl weiterer Methoden und Verbesserungen, deren vollständige Aufzählung den Rahmen der Arbeit sprengen würde. Einen tieferen Einblick geben *Shirley und Morley* (2003).

Ray-Tracing gilt für Echtzeitanwendungen als zu langsam, insbesondere da Ray-Tracer nur in begrenztem Umfang von der Entwicklung der 3D-Grafikkarten profitieren können. Allerdings gibt es Entwicklungen hin zu Ray-Tracing-Hardware (Wald et al, 2001; Schmittler et al, 2002). Diese könnte in Zukunft eine noch realistischere Visualisierung bieten und verspricht insbesondere die Echtzeitvisualisierung von spiegelnden Oberflächen zu verbessern.

3.2 Entwicklung der Grafikkarten

Grafikkarten traten in der Serienproduktion erstmals mit dem „Apple II Mikrocomputer“ auf. Durch zusätzliche Erweiterungskarten konnten damit die Grafikfähigkeiten des Computers verbessert werden. Die erste Grafikkarte für IBM-PCs brachte 1982 die Firma Hercules auf den Markt. Die „Hercules Graphics Card“ (HGC) bot einen Text- und einen Grafikmodus bei einer Auflösung von 720x348 Punkten mit zwei Farben an. Farbige Grafikkarten setzten sich insbesondere ab 1989 mit der VGA-Grafikkarte durch. Der VGA-Standard bietet 640x480 Punkten in 16 Farben. Die Entwicklung von 3D-Grafikkarten, welche die zur Berechnung der Darstellung von dreidimensionalen Szenen benötigten Rechenoperationen selbst ausführen und so die CPU des Computers entlasten, setzte Mitte der 1990er Jahre ein. Die „3Dfx Voodoo 1“ Grafikkarte läutete 1996 diese Ära, auch für den Massenmarkt, endgültig ein (thedodgegarage.com, 2007). Lange boten Grafikkarten nur einen begrenzten Befehlssatz und die Daten wurden in einer festgelegten Befehlsfolge (fixed function pipeline) bearbeitet. Mit dem RenderMan legte die Firma Pixar (Upstill, 1990) den Grundstein für die Entwicklung hin zu flexibleren und programmierbaren Grafikkarten, welche heute Standard sind.

Insbesondere grafikintensive Spiele heizten die Produktentwicklung auf dem Massenmarkt an, was zu einer weiten Verbreitung führte und damit starke Preisnachlässe bei gleichzeitiger Leistungssteigerung ermöglichte. Eine ATI „Radeon X1900 XTX“ Grafikkarte zum Beispiel bietet eine mittels GPUBench (Buck et al, 2004) gemessene Rechenleistung von ca. 240 GFLOPS, verglichen mit der theoretischen Leistung von ca. 25,6 GFLOPS für die SSE-Einheiten eines „dual-core 3,7 GHz Intel Pentium Extreme Edition 965“ (Owens et al, 2007). Die vergleichbare NVIDIA „GeForce 7900 GTX“ Grafikkarte verfügt über eine Bandbreite von ca. 51,2 GB/sec verglichen mit den ca. 8,5 GB/sec der Pentium CPU (Owens et al, 2007). Erreicht werden diese hohen Rechenleistungen durch Spezialisierung und extreme Parallelisierung.

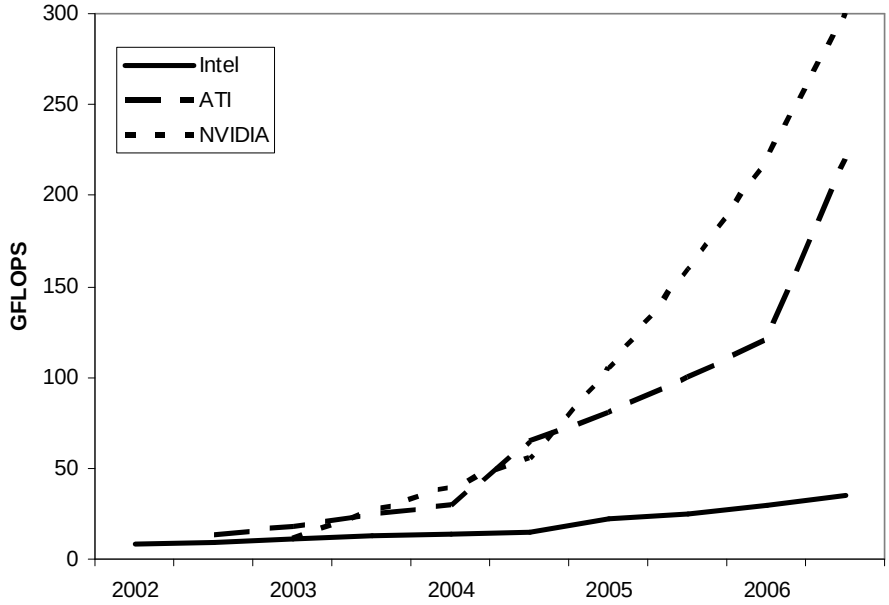

Abbildung 3-2: Zunahme der Rechengeschwindigkeit. Dargestellt sind floating-point Multiplikationen in GFLOPS (nach Owens et al, 2007; und Buck, 2004)

Der Zuwachs an Rechenleistung pro Jahr ist derzeit bei Grafikkarten deutlich höher als bei den CPUs. Die Rechenleistung der Grafikkarten erhöht sich mit einer jährlichen Rate von ca. 1,7 (Pixel / Sekunde) zu 2,3 (Vertices / Sekunde) (Owens et al, 2007). Diese Zuwachsrate unterscheidet sich deutlich von den Entwicklungen bei den CPUs, die, entsprechend dem „Moore'schen Gesetz", jährlich um das 1,4-fache schneller werden. *Moore* sagte bereits 1965 eine Verdopplung der Transistorzahldichte auf integrierten Chipsätzen alle zwei Jahre voraus (Moore, 1965). Seit ungefähr 1996 schwächt sich das Wachstum der CPU-Geschwindigkeit ab (Ekman et al, 2005), so dass die Wachstumsunterschiede zwischen der GPU-Geschwindigkeit und CPU-Geschwindigkeit noch bedeutender sind.

Die Ziele bei der Entwicklung einer GPU unterscheiden sich grundsätzlich von den Anforderungen an eine CPU. CPUs arbeiten weniger stark parallel, haben komplexe Steuerungsanforderungen und niedrigere Leistungsanforderungen als GPUs. Wie in Abbildung 3-3 illustriert, dient ein größerer Teil der CPU der Steuerung des Kontrollflusses, wodurch vergleichsweise weniger Transistoren für die arithmetischen Berechnung zur Verfügung stehen (Owens, 2005). Eine GPU ist auf das Rendern spezialisiert. Durch die Hardwarebeschleunigung erfüllt die GPU diese Aufgabe deutlich schneller als jeder softwarebasierte Ansatz. Der Speicher der Grafikkarte ist auf hohen Durchsatz optimiert, wohingegen der CPU-Hauptspeicher auf geringe Latenzzeiten ausgelegt ist. CPUs sind für die serielle Programmierung optimiert, GPUs für die Stromprogrammierung.

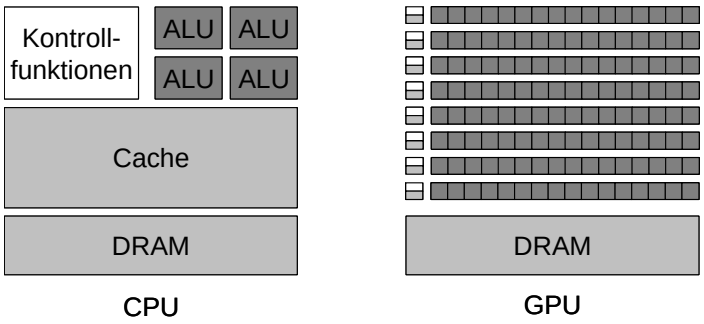

Abbildung 3-3: Eine GPU setzt verhältnismäßig viele Transistoren für die „Arithmetic Logic Units (ALU)" ein (nach NVIDIA, 2007)

Ein Strom ist ein geordneter Datensatz des gleichen Datentyps. Bei der Stromprogrammierung sind alle Daten ein Strom beliebiger Länge. Einen langen Strom zu prozessieren ist dabei effizienter als die Prozessie-

rung einer Reihe kleinerer Ströme (Owens, 2005). Die Berechnung der Ströme wird von sogenannten Kernels durchgeführt. Diese Kernel arbeiten parallel und beeinflussen sich nicht gegenseitig. Die Ausgabe eines Kernel kann die Eingabe eines oder mehrerer nachfolgender Kernel sein.

3.3 Rasterisierung

Bei der Rasterisierung werden die Bildprimitive, normalerweise Dreiecke, zunächst vom dreidimensionalen Weltkoordinatensystem auf das zweidimensionale Bildschirmkoordinatensystem transformiert. Dann wird für jedes Pixel dieser Dreiecke der Farbwert und, falls gewünscht, der Tiefenwert im Bildspeicher (framebuffer) gespeichert. Durch einen Vergleich des neu berechneten Tiefenwertes mit dem bereits im Bildspeicher gespeicherten Tiefenwert wird festgestellt, ob das Pixel sichtbar ist, d.h. ob der Punkt näher an der virtuellen Kamera liegt als der vorher gespeicherte Wert. Durch dieses sogenannte Z-Buffering (Catmull, 1978) werden nur die „sichtbaren“ Pixel dargestellt, nicht jedoch die für den Betrachter verdeckten Pixel. Dazu wird im Bildspeicher ein Z-Puffer angelegt. In diesem Puffer wird an jeder Pixelposition eine Tiefeninformation gespeichert.

In Abbildung 3-4 ist die Funktionsweise des Z-Puffers abgebildet. Für jedes gezeichnete Dreieck wird neben der Bildschirmausgabe ein Z-Puffer angelegt. In diesem wird die Entfernung des gezeichneten Wertes zur virtuellen Kamera gespeichert. Vor der Ausgabe eines Pixels, wie in Abbildung 3-4 beispielhaft durch das weiße Quadrat dargestellt, wird die bereits im Z-Puffer hinterlegte Tiefeninformation mit der Tiefe des aktuell zu zeichnenden Pixels verglichen. Liegt das zu zeichnende Pixel näher an der virtuellen Kamera als der im Z-Puffer hinterlegte Wert, dann wird das Pixel dargestellt und der neue Tiefenwert an der entsprechenden Stelle im Z-Puffer gespeichert.

Die Speichertiefe des verwendeten Z-Puffers hat direkten Einfluss auf die Qualität des Ergebnisses. Ein Z-Puffer mit 8-bit ist deutlich ungenauer und es kommt zu weit mehr Artefakten als bei der Verwendung eines Z-Puffers mit 32-bit. Die Angabe von minimalen und maximalen Abständen der darzustellenden Objekte von der virtuellen Kamera hilft, den für Z-Puffer verwendbaren Speicherraum besser auszunutzen und damit eng beieinander liegende Objekte selbst bei geringer Speichertiefe voneinander unterscheiden zu können.

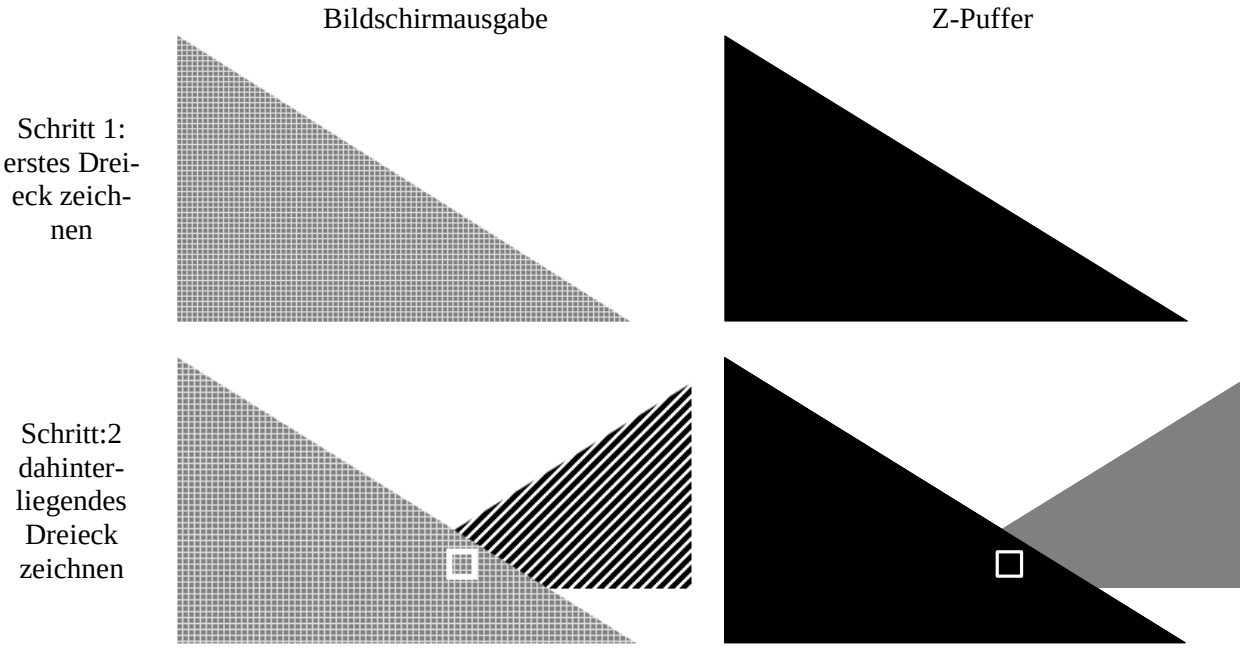

Abbildung 3-4: Funktionsweise des Z-Puffers

Rasterisierung ist normalerweise in der Grafikhardware implementiert. Sie kann allerdings auch softwaremäßig implementiert sein. Bei der Rasterisierung wird über alle Bildprimitive iteriert und diese werden dann in die Bildschirmkoordinaten transformiert und dargestellt. Bei der Rasterisierung wird, im Gegensatz zur Berechnung der globalen Beleuchtung wie beim Ray-Tracing, die Beleuchtungsstärke nur lokal, d.h. pro

Polygon, berechnet. Im Gegensatz zum Pixel-für-Pixel Ansatz des Ray-Tracing werden bei der Rasterisierung Primitive verwendet. Dadurch wird die Berechnung weniger aufwendig und somit schneller. Die hardwarenahe Implementierung der Rasterisierung auf 3D-Beschleunigerkarten ermöglicht die hohe Darstellungsgeschwindigkeit komplexer Szenen.

3.3.1 Hardware Rendering der Fixed-Function Pipeline

Das hardwarebasierte Rendering wird durch moderne Grafikkarten unterstützt. Die 3D-Beschleunigung wird über entsprechende Programmbibliotheken angesprochen. Weit verbreitet sind OpenGL (Shreiner, 2000) und DirectX (Microsoft, 2005), welches im Rahmen dieser Arbeit in der Version 9 verwendet wird. DirectX ist eine umfassende Softwareschnittstelle und schließt, neben der Direct3D genannten Programmierschnittstelle für 3D-Grafik, noch Schnittstellen für 2D-Grafik, Audio- und Videounterstützung sowie die Unterstützung unterschiedlicher Eingabegeräte ein. Die Aufgabe von Direct3D ist das 3D-Rendering. Darunter versteht man die Erzeugung eines Bildes aus einem oder mehreren dreidimensionalen Modellen. Die dreidimensionalen Modelle werden transformiert und die virtuelle Kamera wird mit Standpunkt, Blickrichtung und Zoomfaktor definiert. Die Lichtquellen werden positioniert und evtl. vorhandene Texturen werden geladen. Aus diesen Daten wird ein Bild berechnet. Diese Berechnung erfolgt entweder softwarebasiert oder hardwarebeschleunigt, d.h. eine für diese Aufgabe spezialisierte Grafikhardware führt diese Berechnungen aus.

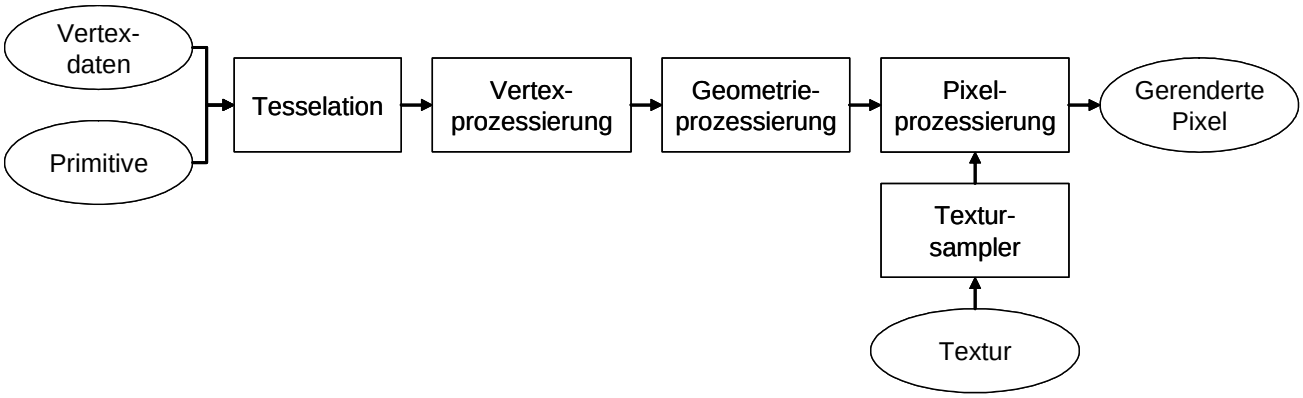

Abbildung 3-5: Direct3D Grafikpipeline (nach Microsoft, 2005)

Die darzustellenden Daten werden von der in Abbildung 3-5 dargestellten Direct3D-Grafikpipeline bearbeitet. Die Szene wird durch sogenannte Vertexe beschrieben. Vertexe sind Eckpunkte im dreidimensionalen Raum. Diese Punkte werden zu Dreiecken verbunden. Ein Vertex kann Positionsinformation, Farbinformation, Texturkoordinaten und die Normalenrichtung beinhalten. Bei der „Tesselation“ werden höherwertige Primitive, wie z.B. Displacement-Maps, in Vertexe umgewandelt, um sie in der folgenden Vertexprozessierung bearbeiten zu können. Die Koordinaten der Vertexe werden in das Bildschirmkoordinatensystem transformiert. In der anschließenden Geometrieprozessierung wird unter anderem das sogenannte „back face culling“ durchgeführt. Beim „back face culling“ werden sämtliche von der virtuellen Kamera abgewandte Dreiecke ignoriert. Die Ausrichtung der Dreiecke wird anhand ihrer Drehrichtung überprüft. Darzustellende Texturen werden für die Bildschirmdarstellung vorbereitet. Danach werden die Pixel prozessiert und schließlich auf dem Bildschirm dargestellt.

In der oben beschriebenen Verarbeitungskette werden die Vertexdaten mehrfach transformiert, um letztlich im Bildschirmkoordinatensystem dargestellt zu werden. Diese Transformationen erfolgen über passende 4x4 Matrizen. Im Gegensatz zu OpenGL sind die Matrixmultiplikationen in DirectX von links nach rechts verkettet und die Koordinatensysteme sind links-händisch (Gray, 2003). Die Transformationen werden immer in homogenen Koordinaten berechnet. Hierfür werden die Punkte und Vektoren um den Wert *w* erweitert. Für Punkte gilt *w=1* für Vektoren *w=0*. Mittels einer Division durch *w*, kann nach einer Transformation ein Punkt auf kartesische Koordinaten zurückgerechnet werden.

$$[x, y, z, 1] \cdot \begin{bmatrix} \text{Welt-} \\ \text{transformation} \end{bmatrix} \cdot \begin{bmatrix} \text{Kamera-} \\ \text{transformation} \end{bmatrix} \cdot \begin{bmatrix} \text{Projektions-} \\ \text{transformation} \end{bmatrix} = [x', y', z', w] \qquad (3.1)$$

3.3.2 Die programmierbare Grafik-Pipeline

Programmierung unter der in Kapitel 3.3.1 vorgestellten „fixed-function pipeline“ besteht hauptsächlich darin, die Einstellungen der Pipeline entsprechend den Anforderungen der Anwendung vorzunehmen und die Daten durch die Grafikpipeline zu leiten. Bis zur Entwicklung von DirectX 8 gab es unter DirectX nur die Möglichkeit, die „fixed-function pipeline“ zur Darstellung von 3D-Grafiken zu verwenden. Dies ermöglichte zwar bereits die Darstellung grafisch aufwendiger Szenen, bot jedoch nicht genügend Flexibilität. Beginnend mit DirectX 8 boten, inspiriert von Pixars RenderMan, sowohl die Grafik-API als auch die Grafikkarten die Möglichkeit, das Rendering auf der Grafikkarte zu programmieren. In zukünftigen Versionen von DirectX wird es, beginnend mit DirectX 10, keine fixed-function pipeline mehr geben.

Über die programmierbare Grafik-Pipeline, schematisch in Abbildung 3-6 dargestellt, kann direkt Einfluss auf den Renderingprozess genommen werden. Neben der Möglichkeit, die Beleuchtung pro Pixel zu berechnen, anstatt der sonst üblichen Berechnung pro Vertex, erlaubt dies ferner eine direkte Manipulation der Vertexdaten während des Rendering. Damit kann die Abbildungsgeometrie in Echtzeit so verändert werden, wie es zur SAR-Simulation nötig ist.

Unter DirectX 8 erfolgte die Programmierung der Grafikhardware noch in Maschinensprache. Dank der Software-API konnten zwar bereits die entwickelten Maschinencodes auf den gängigen Grafikkarten angewendet werden, doch war die Programmierung mühselig und nur schwer verständlich. Inzwischen bietet NVIDIA mit der Grafikprogrammiersprache Cg (NVIDIA, 2005a) eine C-ähnliche Hochsprache zur Grafikkartenprogrammierung an. Microsoft® zog mit DirectX 9 und der Programmiersprache HLSL (High-Level Shading Language) nach und erlaubt nun ebenfalls die Programmierung der Grafikkarten in einer höheren, C-ähnlichen und damit leichter verständlichen Sprache. OpenGL bietet mit der OpenGL Shading Language (GLSL) ebenfalls eine Hochsprache zur Shaderprogrammierung an (Kesenvich, 2006).

Mit HLSL können unter DirectX 9 drei verschiedene Shadertypen programmiert werden. Vertex-Shader sind für die Vertexprozessierung und damit für die Transformationen zuständig. Pixel-Shader berechnen die Farbe und Beleuchtungswerte jedes Pixels und Textur-Shader können zur Erstellung von dynamischen Texturen verwendet werden. Mit DirectX 10 werden, neben Vertex-, Pixel- und Textur-Shader, zusätzlich die Geometry-Shader eingeführt (Blythe, 2006).

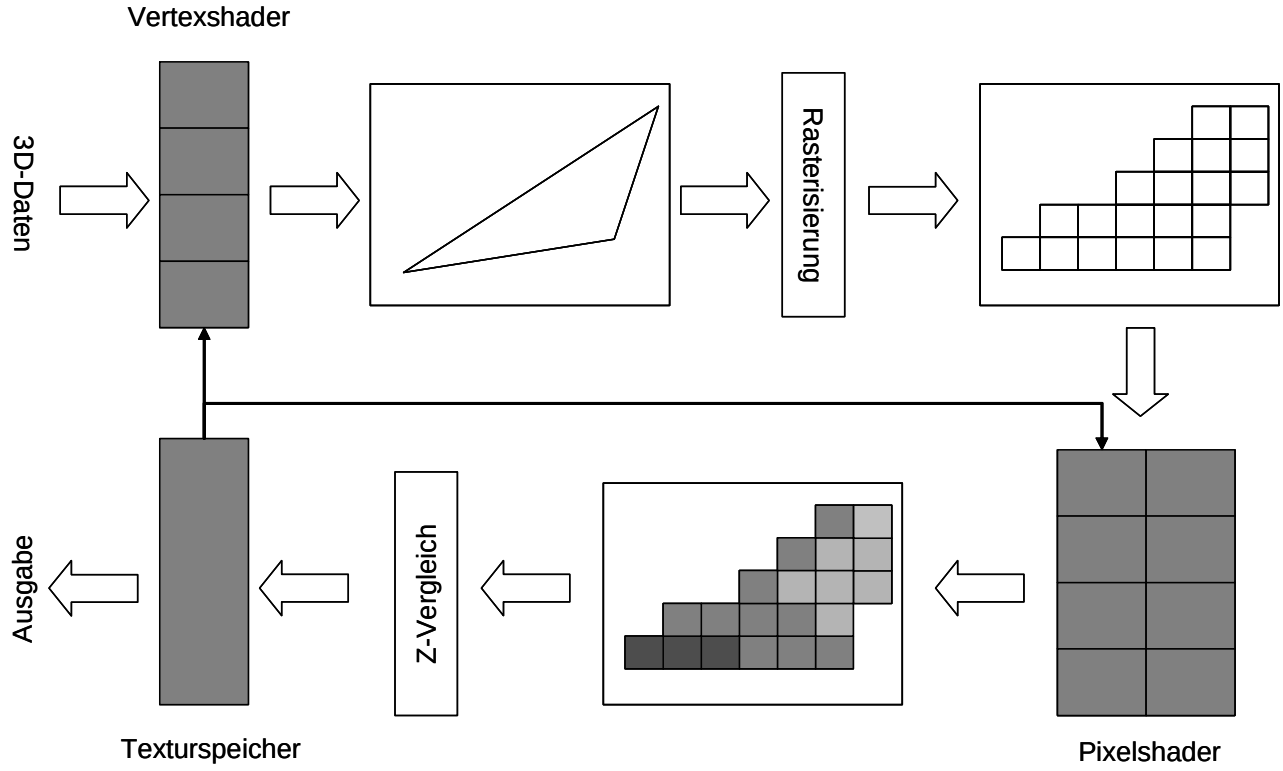

Abbildung 3-6: Die programmierbare Grafikpipeline moderner Grafikkarten

3.3.3 Die Shader

Ein Shader ist ein spezialisiertes Programm. Auf programmierbaren Grafikkarten können unter DirectX Vertex-Shader, Pixel-Shader, Textur-Shader und ab Shaderversion 4.0 auch Geometry-Shader programmiert werden. Die Programmierung moderner Grafikkarten folgt dem Modell der Stromprogrammierung. Das Ergebnis des Vertex-Shaders ist der Eingangswert für den Pixel-Shader. Die Ergebnisse von Textur-Shadern können sowohl als Eingangswerte für Vertex- als auch für Pixel-Shader dienen. Im Vertex-Shader werden die einzelnen Punkte transformiert und letztlich in Bildschirmkoordinaten an den Pixel-Shader weitergereicht. Neben den Koordinaten können noch zusätzliche Informationen, wie z.B. Farbwerte, Texturkoordinaten, Normalenvektoren, usw. weitergereicht werden. Die Pixelkoordinaten sind im Vertex-Shader auf unterschiedliche und flexible Art zu berechnen. Ebenso können Farbinformationen abgeändert werden. Frühe Vertex-Shader erlaubten nur eine geringe Anzahl von Befehlen und keine Verzweigungen, wohingegen spätere Versionen einen immer komplexeren Funktionskatalog aufweisen (siehe Tabelle 3-1).

Pixel-Shader erlauben die Berechnung der Beleuchtung pro Bildpunkt. Sie bieten Funktionen zum Zugriff auf Texturen, Environmental-Maps oder Farboperationen. Vom Vertex-Shader weitergeleitete Werte, wie z.B. Normalenvektoren, können ebenfalls zur Berechnung der Farbwerte verwendet werden. Mittels Textur-Shader können prozedurale Texturen, z.B. für Rauscheffekte, erzeugt werden. Im Gegensatz zu Vertex- und Pixel-Shadern werden Textur-Shader normalerweise nicht für jedes erzeugte Bild aufgerufen, sondern nur wenn eine Textur neu berechnet wird.

Tabelle 3-1: Zusammenfassung der Eigenschaften unterschiedlicher Shaderversionen (Blythe, 2006)

	Version 1.1 (2001)	Version 2.0 (2002)	Version 3.0 (2004)	Version 4.0 (2006)
instruction slots	128	256	≥ 512	≥ 64K
	4 + 8	32 + 64	≥ 512	
constant registers	≥ 96	≥ 256	≥ 256	16x4096
	8	32	224	
tmp registers	12	12	32	4096
	2	12	32	
render targets	1	4	4	8
samplers	8	16	16	16
textures			4	128
	8	8	16	
2D tex size			2Kx2K	8Kx8K
integer ops				✓
load op				✓
sample offsets				✓
transcendental ops	✓	✓	✓	✓
		✓		
derivative ops			✓	✓
flow control		static	stat / dyn	Dynamic
			stat / dyn	

Die Entwicklung der Shader beginnt im Jahre 2001 mit der Version 1.1. Aktuell ist die Version 3.0 gebräuchlich, deren Spezifikation im Jahre 2002 verabschiedet wurde. Shader 3.0 fähige Hardware ist seit 2004 erhältlich. Die Spezifikation von Shader 4.0 ist seit 2006 verfügbar. Seit Ende 2006 sind die ersten Shader 4.0 fähigen Grafikkarten auf dem Markt. Die Möglichkeiten haben sich seit Version 1.1 stark verbessert. Insbesondere die geringe Anzahl möglicher Befehle schränkte in frühen Shaderversionen die Nutzbarkeit stark ein. Die fehlende dynamische Ablaufkontrolle ließ nur einfache Anweisungen zu. Durch die Einführung einer dynamischen Ablaufkontrolle wurde es möglich, vermehrt Programmlogik auf die Grafikkarte auszulagern.

3.4 Radiometrisch hochauflösendes Rendering

Die Unterschiede in der Beleuchtungsstärke zwischen hellen und dunklen Bereichen sind in der Realität um Größenordnungen größer, als dies auf dem Bildschirm darstellbar ist. Das menschliche Auge passt sich dynamisch an die aktuellen Lichtverhältnisse an. Deswegen kann man, nach einer kurzen Anpassungszeit, selbst in dunklen Räumen Objekte erkennen. Tritt man jedoch aus einem dunklen Raum in einen helleren Raum, kann es sein, dass man nichts erkennen kann, solange sich das Auge noch nicht angepasst hat.

Mittels „high dynamic range (HDR) lightning" kann die Berechnung der Beleuchtungsintensität mit einer höheren radiometrischen Auflösung erfolgen. Zwar können auf dem Bildschirm die Farben weiterhin mit nur 8-bit pro Farbkanal dargestellt werden, doch werden bis zu 32-bit pro Farbkanal berechnet. Diese hohe radiometrische Auflösung ermöglicht die korrekte Berechnung der Intensität und die danach erfolgende Anpassung der dargestellten Intensität. Beleuchtungsintensität und -situation können dadurch aneinander angepasst werden. Ebenso ermöglicht dies die Berechnung von Spezialeffekten auf Grund von Über- oder Unterbelichtung. Beim HDR-Rendering wird das Ergebnis zunächst auf eine Textur mit 16- oder 32-bit Farbtiefe gerendert. Diese Textur ist am Bildschirm nicht direkt darstellbar. Danach wird die Beleuchtung im Bild gemessen und das Bild entsprechend radiometrisch skaliert, damit eine sinnvolle und möglichst informationsreiche Darstellung auf 8-bit möglich ist.

3.5 Allgemeine Berechnungen auf der Grafikkarte (GPGPU)

Die Entwicklung der programmierbaren Grafikpipeline erlaubt es, die hohe Rechenleistung moderner Grafikkarten auch für andere Anwendungen zu nutzen. Dieser Teilbereich der Grafikprogrammierung wird als „general-purpose computation on GPUs" (siehe www.gpgpu.org) bezeichnet. Für ein begrenztes Anwendungsfeld bietet die ausgeprägte Spezialisierung und extreme Parallelisierung der GPUs eine deutliche Leistungssteigerung. Die Lösung von nicht grafischen Aufgaben auf spezialisierten Grafikprozessoren ist seit vielen Jahren Forschungsgegenstand. In letzter Zeit entwickelte sich dieses Randgebiet der Forschung zu einem bedeutsameren Wissenschaftsbereich. Die schnelle Entwicklung der Rechenleistung der GPUs und deren weite Verbreitung führen zu einem stetig wachsenden Interesse an diesem Gebiet. Obwohl die GPUs bislang ausschließlich zur Grafikprozessierung entwickelt werden, sind sie ebenso für andere Berechnungen einzusetzen.

Die Programmierung der Grafikpipeline entspricht der Stromprogrammierung. Diese erlaubt eine effiziente und massive Parallelisierung, ist allerdings nicht für jede Aufgabe geeignet. Grafik- und Multimediaanwendungen profitieren am meisten von diesem Modell. Um Multimediaanwendungen zu unterstützen, beinhalten einige moderne CPUs, wie z.B. Intel mit der SSE Technologie, Bereiche zur Stromprozessierung. Grafikkarten sind auf die Stromprozessierung optimiert. Jedes auf einer Grafikkarte zu lösende Problem sollte daher in das Stromprogrammiermodell passen.

Tarditi et al (2006) zeigen in ihrer Arbeit, dass Implementierungen auf der GPU bis zu 18-fach schneller sind als auf der CPU. Je nach Problem und Parallelisierbarkeit kann eine CPU-Implementierung durchaus schneller sein. Bei Aufgaben wie der Matrix-Matrix Multiplikation ist eine GPU-Implementierung nur wenig leistungsfähiger als eine optimierte CPU-Implementierung (Fatahalian et al, 2004). Darüber hinaus ist die Implementierung auf der GPU deutlich schwieriger zu realisieren und weniger flexibel. Dank abstrahierender Hochsprachen bei der GPU-Programmierung (Cg, HLSL, GLSL) ist immerhin eine gewisse Zukunftssicherheit und Weiterverwendbarkeit des Quellcodes gegeben. Abstraktionen und Bibliotheken wie „Accelerator" (Tarditi et al, 2006), „Brook" (Buck et al, 2004) oder „CGis" (Lucas et al, 2006) sind beim Erstellen allgemeiner GPU-Programme hilfreich und senken die Einstiegsschwelle für Programmierer, die nicht auf Grafikprogrammierung spezialisiert sind.

Die Grafikkartenhersteller haben den Trend zu GPGPU erkannt und unterstützen ihn. So bietet z.B. NVIDIA zusammen mit seiner neuesten Generation „GeForce 8800" Grafikkarten die Möglichkeit über die sogenannte „Compute Unified Device Architecture (CUDA)" die Grafikkarte über C-Sprachelemente anzusteuern (NVIDIA, 2007) . Spezielle Bibliotheken zur Berechnung der Fast-Fourier Transformation (CUFFT) und zur Linearen Algebra (CUBLAS) erlauben die schnelle und effiziente Entwicklung entsprechender GPGPU-Ap-

plikationen mit CUDA. Dies trägt zur Vereinfachung der Programmierung von allgemeinen Berechnungen auf der Grafikkarte bei.

Die Rechengenauigkeit von GPUs ist momentan allerdings auf die IEEE 32-bit floating-point Genauigkeit beschränkt. Dabei ist zu beachten, dass die arithmetischen Berechnungen nicht immer IEEE konform sind (Owens et al, 2007). 64-bit Werte sind derzeit nicht verfügbar, was den Einsatz für Berechnungen mit hohen Genauigkeitsanforderungen einschränkt. Das Fehlen von Integer-Werten und Bit-Operationen verhindert bislang den Einsatz von GPUs für viele rechenintensive Anwendungen, wie z.B. die Kryptographie. Beginnend mit DirectX 10 werden Integer-Werte unterstützt (Blythe, 2006), was die Zahl möglicher Anwendungen für die Grafikkarten weiter erhöhen wird. Trotz dieser Einschränkungen gibt es eine Vielzahl von Einsatzfeldern. *Owens et al* (2005; 2007) liefern eine Übersicht über Anwendungen auf GPUs. Darüber hinaus finden sich beispielhafte Publikationen und Anwendungen im Bereich der Strömungsanalysen (Krüger et al, 2005), der Linearen Algebra (Krüger & Westermann, 2003; Galoppo et al, 2005) oder der Datenabfragen (Govindaraju et al, 2004). Weitere Einsatzfelder sind die Signal- und Bildverarbeitung (Sumanaweera & Liu, 2005; Lefohn et al, 2004; Fung & Mann, 2005) oder das Ray-Tracing (Purcell et al, 2002; Purcell, 2004).

Die Menge an Anwendungen ist groß. Das darf jedoch nicht darüber hinwegtäuschen, dass sowohl die Grafikkartenprogrammierung als auch die zu berechnende Methode vom Programmierer beherrscht werden müssen. Darüber hinaus muss das Problem bzw. die Aufgabe sinnvoll auf die Grafikhardware abbildbar und parallelisierbar sein. Die Simulation von SAR-Daten ist eine Visualisierungsaufgabe und eignet sich daher optimal für die Berechnung auf moderner Grafikhardware. Die besonderen Reflexions- und Abbildungseigenschaften von Radardaten erfordern eine umfangreiche Anpassung der aus der Computergrafik stammenden Verfahren. Da die modernen programmierbaren Grafikkarten die dynamische Anpassung des Abbildungs- und Reflexionsmodells erlauben, ist die SAR-Effektvisualisierung in Echtzeit möglich.

4 Grafikhardwarebasierte Echtzeitvisualisierung von SAR-Effekten

Die im Kapitel 2 erläuterten SAR-Abbildungseigenschaften bilden die Grundlagen der SAR-Simulation. Die Nutzung von Methoden aus der der 3D-Computergrafik ermöglicht eine sehr schnelle und realistische Visualisierung von SAR-Effekten. In diesem Kapitel werden der Aufbau und die Implementierung der SAR-Simulation auf Basis von „Graphic Processor Units“ erläutert. Zunächst wird auf die Besonderheiten der Simulation unter DirectX eingegangen und es werden die benötigten Eingangsinformationen erläutert. Die Beschreibung der eigentlichen Visualisierung ist in die Berechnung der Geometrie und der Radiometrie unterteilt. Die Probleme bei der Visualisierung von Mehrfachreflexionen werden gesondert erläutert. Abschließend wird der gesamte Ablauf der Visualisierung dargestellt.

4.1 SAR-Visualisierung in DirectX

DirectX ist eine Sammlung von Programmierschnittstellen zur Spiele- und Multimediaanwendungsentwicklung. DirectX unterstützt die hardwarebeschleunigte 2D- und 3D-Grafikentwicklung (Direct3D) sowie Toneffekte und eine Reihe unterschiedlicher Eingabegeräte. Im Unterschied zu OpenGL von SGI sind die auf Basis von DirectX entwickelten Programme nur unter Microsoft® Windows lauffähig. Im Rahmen dieser Arbeit wird unter DirectX und HLSL (High-Level Shading Language) entwickelt.

Bei der Visualisierung von SAR-Effekten unter DirectX wird zwischen der Sensorposition und der Position der virtuellen Kamera unterschieden. Zwar betrachtet der Sensor die Szene aus der Sensorposition, allerdings wird das berechnete SAR-Bild als eine Art Nadirbild dargestellt. Deswegen ist die Position der virtuellen Kamera von DirectX in Nadirrichtung zu sehen. Dabei ist zu beachten, dass es sich bei SAR-Daten nicht um eine projektive Abbildung handelt. Die Projektionsstrahlen sind bei der SAR-Abbildung parallel. Die Position in x- und y-Richtung des Bildes ist unabhängig von der Höhe des abzubildenden Punktes. Um diese Unabhängigkeit zu erreichen muss die Kameramatrix (siehe Anhang B.1.2) leicht abgeändert werden:

$$\begin{bmatrix} \vec{x}_x & \vec{y}_x & 0 & 0 \\ \vec{x}_y & \vec{y}_y & 0 & 0 \\ 0 & 0 & \vec{z}_z & 0 \\ \langle \vec{x}, k \rangle & \langle \vec{y}, k \rangle & \langle \vec{z}, k \rangle & 1 \end{bmatrix} \tag{4.1}$$

Die Position des Sensors ist wichtig für die Berechnung der Reflexionsstärke und des Schattenwurfs. Die Reflexionsstärke und der Schattenwurf sind unabhängig von der Azimutposition, da die SAR-Aufnahme eine Integration über den Flugweg entlang Azimut darstellt. Daher ist für jede Bildzeile die Sensorposition in Azimut gleich der Position der Bildzeile in Azimut. Im Squintwinkelfall und im Spotlightmodus ist die Situation allerdings anders (siehe Kapitel 4.3.2 und 4.6).

4.2 Eingangsdaten für die Simulation

4.2.1 3D-Objekte als DirectX-Mesh

Um 3D-Koordinaten und Objekte zu speichern, gibt es eine Vielzahl unterschiedlicher Formate. Eine wichtige Bedingung für die Verarbeitung der Objektdaten in DirectX ist die Zerlegung der Objekte in Dreiecke, da die Shader nur Dreiecke verarbeiten können. Die Speicherung der 3D-Objektbeschreibung im DirectX-Mesh-Datenformat bietet den Vorteil, dass die Daten darin für die Verwendung unter DirectX optimiert sind. Das Format erlaubt neben der Speicherung der Punktkoordinaten auch die Speicherung von Normalen, Farb-, Textur- und Nachbarschaftsinformationen. Daneben können Animationen und kleine Programme zur Darstellung der Daten im Format enthalten sein.

DirectX-Mesh-Daten können von verschiedenen bekannten 3D-Grafikprogrammen erzeugt werden. Für die SAR-Visualisierung fehlen in den Meshdaten jedoch Informationen über die Reflexionseigenschaften des

Materials. Da für die Speicherung der Reflexionseigenschaften der verwendeten Materialien eine Vielzahl von Parametern gespeichert werden müssen, werden diese Informationen in einer zusätzlichen Datei abgelegt. Die Trennung von Modell- und Reflexionsdatei erlaubt eine flexible Verwendung der 3D-Modelle auch außerhalb des Simulators und eine einfache Bearbeitung der Modelle in gängigen 3D-Softwarepaketen. Darüber hinaus wird die flexible und modulare Entwicklung unterschiedlicher Beleuchtungsmodelle ermöglicht. DirectX-Meshes können Verweise auf Texturen enthalten. Diese Texturen können, wie in Kapitel 4.4.5 beschrieben, die zur Berechnung der Reflexionsstärke nötigen Parameter berücksichtigen.

4.2.2 Koordinatenreduktion

Die Koordinaten werden auf der Grafikkarte als 32-bit floating-point Daten verarbeitet. Dies ist vergleichsweise ungenau und kann, insbesondere bei der Verwendung von Weltkoordinatensystemen, wie z.B. Gauss-Krüger oder UTM, auf Grund der hohen Koordinatenwerte zu Problemen führen. Aus diesem Grund werden die Koordinaten vor ihrer Verarbeitung reduziert. Dazu wird eine Referenzkoordinate als 64-bit-double-Wert gespeichert. Die restlichen Koordinaten beziehen sich auf diesen Wert.

4.2.3 Sensordaten

Die Sensordaten bilden, neben den Modellen, den maßgeblichen Input für die Visualisierung. Die Beschreibung der Sensordaten umfasst die Position, Geschwindigkeit und Flugrichtung der Plattform, die Blickrichtung des Sensors als Off-Nadirwinkel, die Bodenauflösung, den Squintwinkel, das Radarband, die Polarisation der ausgesendeten und empfangenen Welle und die Position des linkeren unteren Punktes des Bildes in Weltkoordinaten. Diese Daten bilden die Grundlage für die Transformation der Weltkoordinaten, die Berechnung der Reflexionsstärke, des Schattenwurfes und der Position der virtuellen Kamera.

Die Auflösung des simulierten Bildes hängt von der Position der virtuellen Kamera und dem gewählten Blickfeld („field of view“) der Kamera ab. Bei stabilem Zoomfaktor wird eine geringere Pixelauflösung durch eine Erhöhung der Kameraposition erreicht. Die Höhe der virtuellen Kamera k_z hängt von der gewünschten Bodenauflösung δ_{rg}, dem „field of view“ *fov* und der Größe des simulierten Bildes in Pixel w_{pix} ab. Bei konstantem *fov* verändert sich die Höhe der virtuellen Kamera bei Veränderungen der Szenengröße oder der Auflösung.

$$k_z = \frac{\delta_{rg} \cdot w_{pix}}{fov} \tag{4.2}$$

4.3 Berechnung der SAR-Geometrie

Die Abbildungsgeometrie eines SAR-Bildes unterscheidet sich fundamental von der perspektivischen Geometrie, wie sie standardmäßig von den 3D-Entwicklungsbibliotheken unterstützt wird. So hängt die Position eines Punktes in Rangerichtung (also in Richtung des ausgesendeten Signals) von der Laufzeit des Signals ab. Ein Objekt A, welches höher als ein Objekt B liegt, allerdings in Ground-Rangerichtung gleich weit entfernt ist, erscheint daher im SAR-Bild näher (siehe Kapitel 2.2). Die Position in Rangerichtung des SAR-Bildes hängt von der Höhenlage des Objektes ab. In Azimutrichtung tritt dieser Effekt nicht auf. Die Position eines Objektes in Azimut wird nicht durch die Laufzeit eines Strahls bestimmt, sondern durch den Dopplereffekt. Störende Effekte und Falschpositionierungen von Objekten in Azimut treten durch Störungen des Dopplereffektes, vor allem durch Eigenbewegungen der beobachteten Objekte, auf (siehe Kapitel 2.2.2 und 4.3.4). Bedingt durch die Laufzeitgeometrie kann es dazu kommen, dass die Reflexionen unterschiedlicher Objekte gleichzeitig zum Sensor zurückkommen. Die Entfernung dieser Objekte zum Sensor ist identisch und die einzelnen Signale addieren sich im Sensor auf.

Ein SAR-Bild in Ground-Range stellt die beobachtete Szene als „Nadiraufnahme“ dar, obwohl die eigentliche Aufnahme schräg erfolgt. Diese Darstellung ist bei der Auswertung sowohl hilfreich, da dem Anwender die Navigation im Bild leichter fällt, als auch störend, da die eigentliche Schrägsicht des Systems nicht direkt erkennbar ist. Bei der Auswertung von SAR-Bildern unterscheidet man zwischen Slant-Range-Bildern, welche die Laufzeitgeometrie direkt abbilden und Ground-Range-Bildern. Normalerweise werden nur Ground-Range-Bilder ausgewertet, weswegen sich diese Arbeit auf deren Visualisierung beschränkt.

Es ergeben sich für die die Visualisierung von SAR-Effekten mittels 3D-Grafikkarten folgende Vorgaben:

1. Die Darstellung der Daten soll in Ground-Range erfolgen
2. Die Geometrie in Rangerichtung muss der Laufzeitgeometrie entsprechen
3. Die Darstellung der Daten erfolgt aus der Nadirposition
4. Verdeckungen und Abschattungen beziehen sich auf die Sensorposition
5. Die Reflexion unterschiedlicher Punkte mit gleicher Entfernung zum Sensor addieren sich auf

4.3.1 Transformation in das Simulationskoordinatensystem

Die zu simulierenden Daten müssen in einem einheitlichen Weltkoordinatensystem vorliegen. Diese werden dann in das Simulationskoordinatensystem transformiert. Der Ursprung dieses Koordinatensystems ist die linke untere Ecke des simulierten Bildes *b*. Die y-Achse verläuft entlang der Flugbahn des Sensors. Die Flugbahn wird als Gerade angenommen. Die x-Achse verläuft orthogonal zur Flugbahn und entspricht, abgesehen vom Squintwinkelfall (siehe Abschnitt 4.3.2), der SAR-Signalrichtung. Die Koordinaten sind in Meter angegeben. Ausgehend von einem auf Nord ausgerichteten Weltkoordinatensystem mit Angaben in Metern und einer Azimutrichtung ϕ berechnet sich die Transformationsmatrix aus der Translation und der Rotation um die Z-Achse folgendermaßen:

$$[x,y,z,1]\cdot\left(\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\-b_x&-b_y&0&1\end{bmatrix}\cdot\begin{bmatrix}\cos\phi&\sin\phi&0&0\\-\sin\phi&\cos\phi&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\right)=[x',y',z,1] \qquad (4.3)$$

Die Festlegung der Azimutrichtung als y-Achse ist willkürlich. Die Transformation der Daten in das Simulationskoordinatensystem erfolgt anhand der geraden Flugbahn der Plattform. Der Nullwert auf der z-Achse entspricht der angenommenen Ebene, welche die Erdoberfläche repräsentiert. Die Höhe dieser Ebene in Weltkoordinaten ist einstellbar. Dieses Verhalten entspricht der bei realen SAR-Systemen einstellbaren angenommenen Flughöhe über Grund.

Eine umfangreichere Transformation der Weltkoordinaten in Simulationskoordinaten, z.B. unter Einbeziehung gekrümmter Flugbahnen oder einer gekrümmten Erdoberfläche, ist denkbar und je nach Einsatzgebiet, z.B. für den Fall der Satellitenradarfernerkundung, sinnvoll. Im weiteren Verlauf werden Koordinaten im Simulationskoordinatensystem vorausgesetzt.

4.3.2 Transformation der Squintwinkelgeometrie

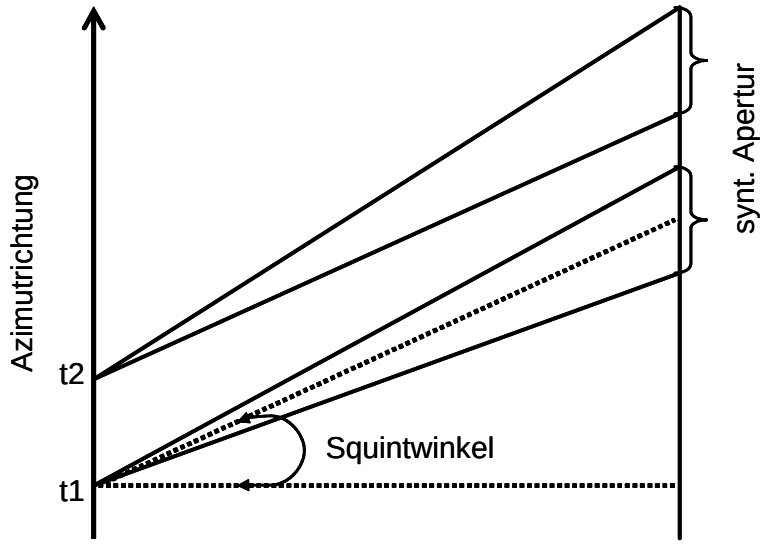

Abbildung 4-1: Schematische Darstellung Squintwinkel

SAR-Systeme können nicht nur in 90° zur Azimutrichtung senden, sondern auch schräg nach vorne oder hinten. Der Squintwinkel ω ist der Winkel zwischen der Blickrichtung des Sensors nach schräg vorne oder hinten und der Achse im rechten Winkel zur Azimutrichtung (siehe Abbildung 4-1).

Die Squintwinkelgeometrie kann, vereinfacht gesehen, durch eine Drehung der Flugrichtung des Sensors, d.h. durch eine Veränderung der Azimutrichtung, visualisiert werden. Diese Drehung muss vor der Darstellung des Bildes rückgängig gemacht werden. Damit die Drehung sich auf den richtigen Drehpunkt, nämlich den Nadirpunkt *p*, bezieht, wird zunächst eine Translation durchgeführt. Danach wird um den Squintwinkel ω rotiert und schließlich wird die Translation zurückgenommen.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}^T \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -p_x & -p_y & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos-\omega & \sin-\omega & 0 & 0 \\ -\sin-\omega & \cos-\omega & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p_x & p_y & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z \\ 1 \end{bmatrix}^T \tag{4.4}$$

Die Rückdrehung erfolgt dann vor der Ausgabe der Textur:

$$[tu, tv] \cdot \begin{bmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{bmatrix} = [tu', tv'] \tag{4.5}$$

Diese Visualisierung ist vereinfachend, da einige SAR-Effekte, insbesondere Multipath und Dopplerverschiebungen, dadurch im Squintwinkelfall nicht richtig simuliert werden. Außerhalb dieser Fälle ist die vereinfachte Betrachtung der Squintwinkelgeometrie ausreichend.

4.3.3 Anpassung der Geometrie in Rangerichtung

Liegen die Daten im Simulationskoordinatensystem vor, so betreffen geometrische Änderungen durch die Laufzeitgeometrie nur die x-Achse. Die x-Position eines Punktes wird durch die Entfernung zum Sensor und der Ebene, auf welche die SAR-Daten bei der Prozessierung projiziert werden, bestimmt. Diese Ebene entspricht in unserem Fall dem Nullwert der z-Achse im Simulationskoordinatensystem. Punkte, deren z-Werte im Simulationskoordinatensystem gleich null sind, werden auf ihrem x-Wert im Simulationskoordinatensystem abgebildet. Werden die z-Werte verändert, dann bedeutet dies eine Verschiebung der Punktprojektion in x-Richtung. Punkte mit höheren z-Werten liegen näher zum Sensor und haben entsprechend niedrigere x'-Werte.

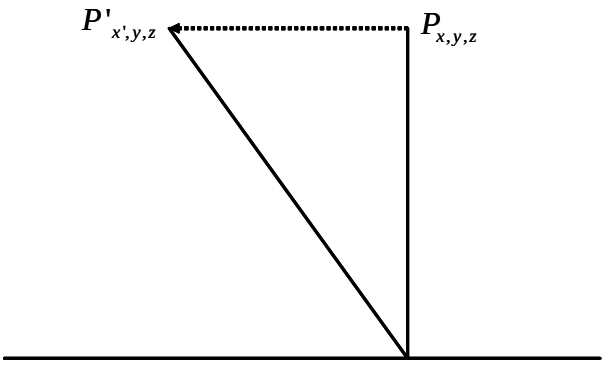

Abbildung 4-2: Abbildung des Punktes P in P' durch die Laufzeitgeometrie

Bei der Transformation des Punktes P von der Weltgeometrie zum Punkt P' in Laufzeitgeometrie ist x' des Punktes P' eine Funktion aus x, z und dem Off-Nadirwinkel θ_{off}. Die Koordinaten berechnen sich aus:

$$\begin{aligned} x' &= x - z \cdot \tan\theta_{off} \\ y' &= y \end{aligned} \tag{4.6}$$

Am Einfachsten lässt sich dies durch eine Anpassung der Kameramatrix aus (4.1) erreichen:

$$\begin{bmatrix} \vec{x}_x & \vec{y}_x & 0 & 0 \\ \vec{x}_y & \vec{y}_y & 0 & 0 \\ -\tan\theta_{off} & 0 & \vec{z}_z & 0 \\ \langle \vec{x}, k \rangle & \langle \vec{y}, k \rangle & \langle \vec{z}, k \rangle & 1 \end{bmatrix} \tag{4.7}$$

Die Transformation der Koordinaten in die SAR-Geometrie ist damit für sich nicht bewegende Objekte erfolgt. Für Objekte, die sich während der SAR-Aufnahme bewegen, ist die Annahme von *y' = y* ungültig. Diese Objekte werden in Abhängigkeit ihrer Eigengeschwindigkeit und Bewegungsrichtung in y-Richtung verschoben (siehe Kapitel 4.3.4).

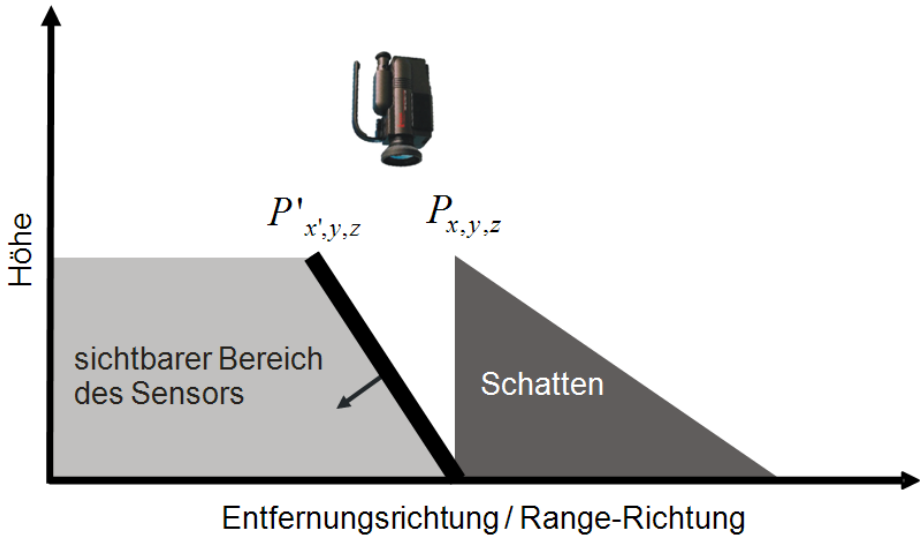

Abbildung 4-3: Transformation in die Laufzeitgeometrie

Die Simulation einer Wand soll im Folgenden als Beispiel dienen. Die Wand wird bei der Simulation aus einer Reihe einzelner Dreiecke (mindestens zwei) beschrieben. Für die schematische Darstellung der Abbildung in Rangerichtung sind in Abbildung 4-3 nur zwei Punkte bedeutend: der Fußpunkt der Wand und die Spitze. Die Wand wird durch die Linie zwischen den beiden Punkten beschrieben. In Abbildung 4-3 liegt der Fußpunkt der Wand auf *z=0* im Simulationskoordinatensystem und wird daher nicht verschoben. Die restliche Wand wird in Richtung zum Sensor verschoben, so dass die Wand in der laufzeitgeometrischen Abbildung nicht mehr senkrecht, sondern schräg steht (siehe auch Abbildung 4-2).

Da SAR-Bilder als Nadiraufnahmen dargestellt werden, wird bei der Visualisierung die Position der virtuellen Kamera von DirectX in Nadir gewählt. Durch die veränderte Geometrie betrachtet die virtuelle Kamera die Rückseite der Wand. Dank der Verschiebung wird zwar das Layover korrekt abgebildet, allerdings würde die Reflexion der Wandrückseite berechnet werden, nicht der Wandvorderseite. Um Darstellung von nicht sichtbaren Dreiecken zu vermeiden, wird bei der 3D-Visualisierung oftmals das sogenannte „back face culling“ angewendet. Diese Technik kann hier jedoch keine Anwendung finden, da ansonsten die Dreiecke, welche die Wand darstellen, ignoriert werden würden, bzw. falls die Wand als Quader modelliert wurde, dann würde fälschlicherweise die Vorderseite ignoriert und die Rückseite abgebildet werden.

4.3.4 Veränderung der Geometrie in Azimut durch Eigenbewegung der Objekte

Objekte mit einer Eigenbewegung während des Aufnahmezeitraums werden durch die Dopplerverschiebung in Azimutrichtung verschoben (siehe Kapitel 2.2.2). Dieser Versatz lässt sich durch eine Anpassung der Kameramatrix aus Gleichung (4.7) erreichen. In Anlehnung an die Gleichung (2.1) definieren wir die relative Eigengeschwindigkeit v_r als das Verhältnis der Geschwindigkeit des Objektes v_o in Entfernungsrichtung zur Geschwindigkeit der Sensorplattform v_s:

$$v_r = \frac{v_o}{v_s} \tag{4.8}$$

Die Entfernung zwischen Objekt und Sensor r_O könnte man statisch für die gesamte Szene festsetzen oder für jede Dreieckskoordinate im Vertex-Shader berechnen. Die Kameramatrix unter Berücksichtigung der Eigenbewegung eines Objektes entspricht dann:

$$\begin{bmatrix} \vec{x}_x & \vec{y}_x & 0 & 0 \\ \vec{x}_y & \vec{y}_y & 0 & 0 \\ -\tan\theta_{off} & 0 & \vec{z}_z & 0 \\ \langle \vec{x}, k \rangle & \langle \vec{y}, k \rangle - r_O \cdot v_r & \langle \vec{z}, k \rangle & 1 \end{bmatrix} \tag{4.9}$$

Neben der Bewegung in Entfernungsrichtung beeinflusst auch eine Bewegung in Azimutrichtung die Visualisierung eines Objektes. Wird das Objekt durch eine Bewegung in Entfernungsrichtung verschoben, so führt die Bewegung in Entfernungsrichtung zu einem „Verwischen“ des Objektes im SAR-Bild. Gleichzeitig wird das Objekt gedehnt, wenn es sich in Azimutrichtung bewegt und gestaucht, wenn es sich entgegen der Azimutrichtung bewegt. Hierzu wird das Objekt um $1+v_r$ skaliert, wobei v_r in diesem Fall aus der Eigengeschwindigkeit des Objektes in Azimutrichtung berechnet wird. Diese Visualisierung ist vereinfachend, da eine Reihe von Effekten unberücksichtigt bleiben (Schreier, 1993). So wird unter anderem die Beschleunigung von Objekten nicht berücksichtigt.

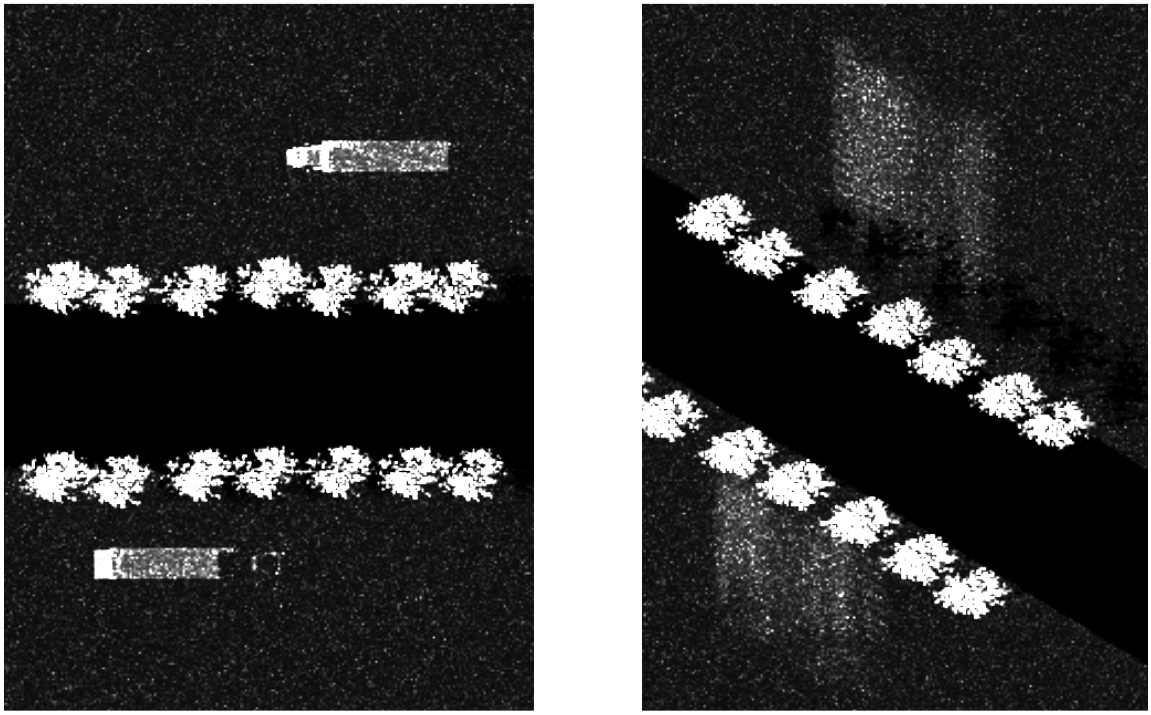

Abbildung 4-4: Beispielsszene mit sich in Entfernungsrichtung bewegenden Fahrzeugen (links) und mit Fahrzeugen die sich in Entfernungs- und Azimutrichtung bewegen (rechts)

Die Skalierung wird in der Weltmatrix (4.3) vorgenommen:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1+v_r & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -b_x & -b_y & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & \sin\phi & 0 & 0 \\ -\sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

Durch die Bewegung des Objektes in Azimutrichtung wird das Objekt verwischt. Das Ausmaß der Verwischung hängt von der Distanz ab, die das Objekt zurücklegt, während es beleuchtet wird. Die Zeitdauer der Beleuchtung Δt hängt von der Länge der synthetischen Apertur aus Gleichung (A.5) und damit von der Auflösung der realen Apertur ab und berechnet sich wie folgt:

$$\Delta t = \frac{\lambda \cdot r_O}{v_s \cdot l_{ra}} \tag{4.11}$$

Um den Verwischungseffekt zu visualisieren, wird das Objekt *n*-mal jeweils um einen Pixel versetzt gezeichnet, wobei bei jedem Durchlauf die gezeichnete Pixelstärke mit *1/n* multipliziert wird. Dabei hängt *n* von der Strecke, die das Objekt in Δt in Azimutrichtung zurücklegt und der Pixelauflösung ab. *n* berechnet sich aus:

$$n = \frac{\Delta t \cdot v_o}{\delta_{rg}} = \frac{\lambda \cdot r_O}{l_{ra} \cdot \delta_{rg}} \cdot v_r \tag{4.12}$$

Das Ergebnis dieser Verschiebung und Verwischung ist in Abbildung 4-4 zu sehen. Die Fahrzeuge werden, wenn sie sich in Entfernungsrichtung bewegen, in Abhängigkeit ihrer Geschwindigkeit in Entfernungsrichtung verschoben. Bewegen sie sich außerdem in Azimutrichtung, werden sie zusätzlich gestreckt bzw. gestaucht und verwischt (vgl. Abbildung 2-9).

4.3.5 Sichtbarkeitsanalyse

Zur Analyse der Sichtbarkeit wird die Normalenrichtung $\vec{n}$ eines Dreiecks und die Position des Sensors *s* betrachtet. Es wird dabei stets die Normale in der Weltgeometrie und nicht in der Laufzeitgeometrie betrachtet. Der Kosinus des lokalen Inzidenzwinkels wird berechnet aus:

$$\cos\theta = \left\langle \frac{\vec{n}}{\|\vec{n}\|}, \frac{s}{\|s\|} \right\rangle \tag{4.13}$$

Wenn $\cos\theta < 0$ dann zeigt der Normalenvektor in eine vom Sensor abgewandte Richtung, d.h. die Fläche ist nicht dem Sensor zugewandt.

Durch die Betrachtung der Normalen können zwar die Ausrichtung der Flächen zum Sensor analysiert werden, nicht jedoch deren Verdeckungen. Wie in Abbildung 4-3 zu erkennen, liegt ein Teil der Erdoberfläche durchaus im Sichtbereich des Sensors, wird allerdings durch die laufzeitgeometrische Veränderung der Wandgeometrie von der Wand überdeckt und ist daher von der virtuellen Kamera aus nicht zu sehen.

Bei der Rasterisierung werden alle Dreiecke in der Reihenfolge dargestellt, in der sie an die Grafikpipeline überreicht werden. Dabei wird die Sichtbarkeit nicht überprüft. Näher an der virtuellen Kamera liegende Dreiecke können von entfernter liegenden verdeckt werden. Um dies zu verhindern, könnten die Dreiecke zunächst sortiert werden. Dann werden zuerst die vom Sensor am weitesten entfernten Dreiecke dargestellt. Diese werden später, von den sich näher am Sensor befindenden Dreiecke, überschrieben. Die Sortierung der Dreiecke ist allerdings zeitaufwendig und im Einzelfall problematisch.

Besser geeignet ist der von *Catmull* (1978) entwickelte Z-Puffer-Algorithmus (siehe Kapitel 3.3). Der Einsatz eines Z-Puffers kommt bei der SAR-Effektvisualisierung jedoch nicht in Frage, bzw. nur in abgewandelter Form (siehe 4.4.2), da ein Z-Puffer, der sich auf die virtuelle Kamera bezieht, die Verdeckungen falsch berechnen würde.

4.4 Berechnung der Radiometrie

Neben der Abbildung der SAR-Geometrie ist die SAR-Radiometrie von entscheidender Bedeutung für die realistische Darstellung von SAR-Bildern. Unterschiede in der Radiometrie von optischen und SAR-Bildern entstehen hauptsächlich durch die unterschiedlichen Wellenlängen. Die Rauigkeit eines Objektes ist bei der

Visualisierung immer im Vergleich zur verwendeten Wellenlänge zu sehen. Deswegen müssen bei den längeren Wellenlängen der SAR-Daten weit mehr Objekte als spiegelnd betrachtet werden, als dies bei Wellenlängen im optischen Bereich der Fall wäre. Daneben ist die Addition gleichzeitig am Sensor eintreffender Reflexionen (siehe Abschnitt 4.4.4) ein bedeutender Unterschied bei der Visualisierung.

4.4.1 Berechnung des Rückstreukoeffizienten $\sigma°$

Entscheidend für die Berechnung der Radiometrie ist der Rückstreukoeffizient $\sigma°$ (siehe Anhang A.3). Für eine bestimmte Polarisation kann σ^o_{pp} als Produkt zweier Funktionen ausgedrückt werden: der Funktion zur Beschreibung der Oberflächenrauigkeit $f_s(\theta)$ und der Funktion der dielektrischen Materialeigenschaft $f_{pp}(\varepsilon_s\theta)$, mit ε_r als der relativen dielektrischen Konstante des Materials.

$$\sigma^o_{pp}(\theta) = f_{pp}(\varepsilon_r\theta) \cdot f_s(\theta) \tag{4.14}$$

(nach Weimann et al, 1998)

Prinzipiell unterteilt sich der Kurvenverlauf der Rückstreuung in Abhängigkeit des Inzidenzwinkels in drei Regionen: dem spiegelnden oder quasi-spiegelnden Bereich, dem Plateaubereich und dem Schattenbereich (siehe Abbildung 4-5). Bei unterschiedlichen Materialien, Wellenlängen und Polarisationen sind die einzelnen Bereiche unterschiedlich stark ausgeprägt.

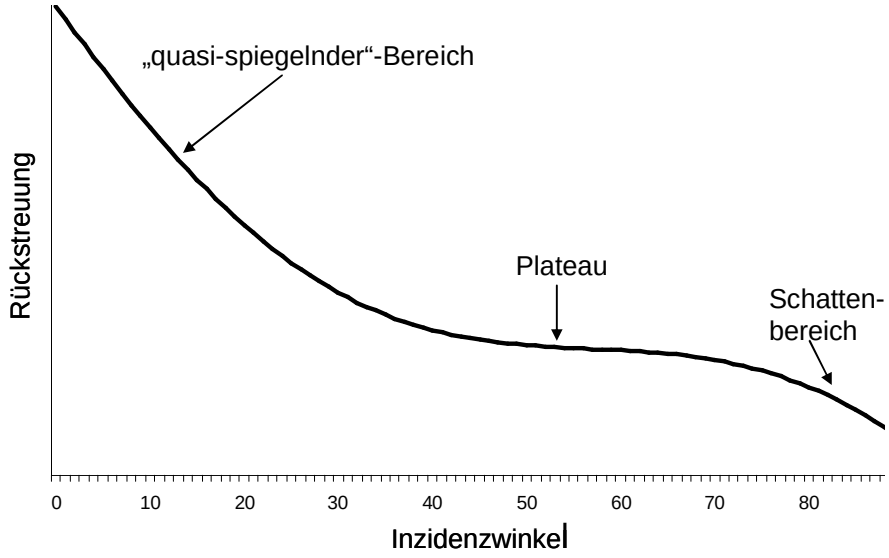

Abbildung 4-5: Genereller Kurvenverlauf der Rückstreuung in Abhängigkeit des Inzidenzwinkels (nach Ulaby & Dobson, 1989)

Zur Berechnung der Rückstreuung werden in den unterschiedlichen Simulationen und Studien verschiedene Methoden angewandt. Dabei kann man drei Haupttypen unterscheiden (siehe 1.2):

1. Bestimmung der Rückstreuung mittels Datenbanken bzw. Tabellen
2. Berechnung der Rückstreuung auf Basis von realen Radardaten oder durch inkohärente Modelle
3. Berechnung der Rückstreuung basierend auf einem mathematisch/physikalischen Modell

Im Folgenden werden die unterschiedlichen Typen beispielhaft implementiert. So wird in Kapitel 4.4.1.1 die Rückstreuung anhand von Rückstreuungsdatenbanken und ihrer statistischen Auswertung bestimmt. In Kapitel 4.4.1.3 wird die Rückstreuung basierend auf einem mathematisch/physikalischen Modell berechnet. In der SAR-Simulation wird neben dem Kirchhoff Modell und dem „small pertubation Model (SDM)" (Ulaby et al, 1986) vor allem das „Integral Equation Model (IEM)" (Fung et al, 1992; Fung, 1994) verwendet. Da das IEM auf Grund der vergleichsweise komplexen Berechnung für die grafikhardwarebasierte Berechnung

in Echtzeit ungeeignet ist, wird das Modell zur Berechnung des Reflexionsverhaltens nach *Zribi et al* (2006) implementiert. Dieses Modell kommt mit nur wenigen Parametern aus. Die Berechnung in Kapitel 4.4.1.2 basiert nicht auf der Rückstreuung von realen Radardaten, sondern auf inkohärenten Modellen, abgeleitet aus den Methoden der Computergrafik. Die zur Berechnung der Rückstreuung in Kapitel 4.4.1.2 notwendigen Variablen sind dimensionslos. Realistische Werte für diese Variablen können entweder durch einen Vergleich der Daten mit realen Radarbildern oder durch einen Rückgriff auf eine Rückstreuungsdatenbank gefunden werden.

4.4.1.1 Datenbank mit unterschiedlichen Reflexionseigenschaften

Das Reflexionsverhalten einzelner Objekte hängt von der lokalen Blickrichtung, der Wellenlänge und der Polarisation der elektromagnetischen Welle ab. *Ulaby und Dobson* (1989) haben eine umfangreiche Sammlung von Messwerten unterschiedlicher Oberflächenklassen veröffentlicht. Diese umfangreiche Sammlung dient als Grundlage für den Aufbau einer Rückstreuungsdatenbank. Mit den in einer solchen Datenbank gespeicherten Werten lässt sich bei einer Simulation für ein bestimmtes Material die Rückstreuung unter einem bestimmten Inzidenzwinkel ermitteln.

Bei der Programmierung von Grafikkarten sind solche Datenbankabfragen nur sehr umständlich und langsam zu implementieren. Die Grafikkarten sind schnell in der Berechnung, jedoch vergleichsweise langsam bei wahlfreien Datenzugriffen. Da *Ulaby und Dobson* nicht nur Daten erhoben haben, sondern die gesammelten Daten auch statistisch auswerteten, berechneten sie für jedes Material, Radarband und Polarisation die Werte für P_1 bis P_6 der Gleichung (4.15).

$$\sigma^\circ = P_1 + P_2^{(-P_3\theta)} + P_4 \cos(P_5\theta + P_6) \tag{4.15}$$

Die Werte haben keinen physikalischen Bezug und neue Materialien lassen sich nur über umfangreiche Messungen in die Datenbank einführen. Sind die Werte jedoch bekannt, so lässt sich über die sechs Parameter die Rückstreuung berechnen. Im Gegensatz zu einer Datenbankabfrage lassen sich so die Rückstreuung von in der Datenbank vorhandenen Materialien realistisch und einfach auf der Grafikhardware berechnen. Die Werte von P_1 bis P_6 werden für jedes Material definiert, der lokale Inzidenzwinkel θ wird nach (4.13) berechnet.

4.4.1.2 Anpassung von Modellen der Computergrafik an die SAR-Rückstreuung

Die Computergrafik bietet unterschiedliche Methoden zur Beleuchtungsberechnung. Diese lassen sich auch für die SAR-Visualisierung anpassen. Die Grafikhardware ist für einige dieser Methoden optimiert, so dass diese für die Reflexionsberechnung auf der Grafikhardware prädestiniert sind.

Objektoberflächen, deren Höhenvarianz Δh weit größer als die Wellenlänge λ im Verhältnis zum Einfallswinkel ist, verhalten sich, wie in Kapitel 2.3.1.1 beschrieben, wie Lambertsche Streuer (Klausing & Holpp, 2000). Lambertsche Streuer reflektieren in alle Richtungen gleich stark. Die Reflexionsstärke hängt von den Materialeigenschaften und der effektiv sichtbaren Fläche des Objektes ab. In der Computergrafik berechnet man den Reflexionswert r_D eines Lambertschen oder diffusen Streuers für einen Farbkanal aus dem Skalarprodukt des Normalenvektors $\vec{n}$ und des normalisierten Vektors vom Objekt zur Lichtquelle $\vec{l}$ multipliziert mit der diffusen Farbstärke des betrachteten Materials m_D und der diffusen Farbstärke des Lichtes l_D (Gray, 2003)

$$r_D = m_D l_D \cdot \left\langle \vec{n}, \vec{l} \right\rangle \tag{4.16}$$

Für den SAR-Fall betrachtet man anstatt der Lichtquelle den normalisierten Positionsvektor des Sensors $\vec{s}$ und anstatt den Farbstärken des Materials und des Lichtes den Reflexionswert *r*.

$$\sigma_D^o = r \cdot \left\langle \vec{n}, \vec{s} \right\rangle \tag{4.17}$$

Der Reflexionswert *r* ist immer positiv oder null. Er beinhaltet die für die Reflexionseigenschaft des Materials relevanten Werte, einschließlich der dielektischen Konstante und hängt von der Polarisation der ausgesendeten und empfangenen Welle ab. *r* wird durch den Anwender der Simulation festgelegt. Dabei ist *r* nicht näher definiert und die Wahl eines realistischen Wertes für *r* bleibt dem Anwender überlassen.

Das Skalarprodukt des Normalenvektors $\vec{n}$ und der normalisierten Sensorposition $\vec{s}$ sind für vom Sensor abgeneigte Flächen negativ. Vom Sensor abgeneigte Flächen reflektieren nicht zum Sensor zurück, bzw. können von diesem nicht angestrahlt werden. In diesem Fall gilt $\sigma^\circ = 0$.

Die Reflexionsstärke einer spiegelnden Oberfläche ist stark von der Beleuchtungs- und Blickrichtung abhängig. Sie berechnet sich aus den Farbwerten des Materials m_s und des Lichtes l_s sowie aus der Normalenrichtung und dem normalisierten Vektor $\vec{h}$, der genau zwischen dem Vektor vom betrachteten Objektpunkt zur Lichtquelle und dem Vektor vom betrachteten Objektpunkt zur virtuellen Kamera liegt (Blinn, 1977; Gray, 2003).

$$r_S = m_S l_S \cdot \left\langle \vec{n}, \vec{h} \right\rangle^p \tag{4.18}$$

Im monostatischen Fall entspricht die Beleuchtungsrichtung der Blickrichtung, so dass der normalisierte Vektor $\vec{h}$ dem normalisierten Vektor $\vec{s}$ entspricht. Dadurch vereinfacht sich die Berechnung von σ° einer spiegelnden Oberfläche:

$$\sigma_S^o = r \cdot \left\langle \vec{n}, \vec{s} \right\rangle^p \tag{4.19}$$

Die Potenz *p* ist variabel. Je höher *p* gewählt wird, desto stärker ist die Bündelung der Reflexion, d.h. dementsprechend stärker ist die Reflexion im Reflexionszentrum und desto schneller nimmt sie nach außen hin ab. Typische Werte für *p* liegen bei $p \approx 32$.

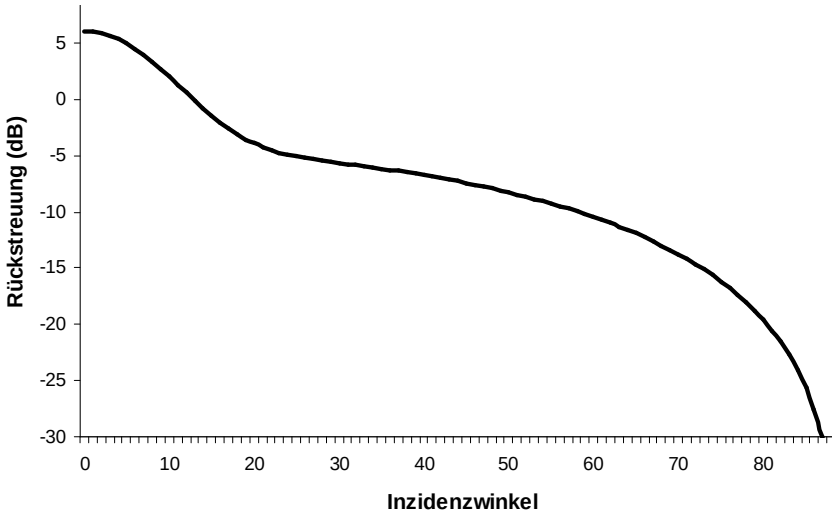

Abbildung 4-6: Beispielhafter Kurvenverlauf der Rückstreuung von (4.20)

Viele natürliche Oberflächen sind weder absolute Lambertsche Streuer noch rein spiegelnde Oberflächen. Die Gleichung des Lambertschen Streuers (4.17) entspricht der einer spiegelnden Oberfläche (4.19), wenn *p = 1* gilt. Eine Gleichung, die in Abhängigkeit der Rauigkeit einer Materialoberfläche den Wert *p* bestimmt, wäre eine einfache Möglichkeit, die Gesamtreflexion zu berechnen. Allerdings wird dadurch der spiegelnde Anteil zu stark überbetont. Ein etwas realistischeres Ergebnis erhält man, wenn die Gesamtreflexion sich anteilig aus Lambertscher und spiegelnder Reflexion zusammensetzt. In Gleichung (4.20) ist dies dargestellt. Die Gesamtreflexion σ° setzt sich anteilig, in Abhängigkeit des Rauhigkeitswertes *z* mit $0 \le z \le 1$, zusammen aus der diffusen Reflexion σ_D^o und der spiegelnden Reflexion σ_S^o. *p* bleibt konstant.

$$\sigma^\circ = (z \cdot \sigma_D) + ((1-z) \cdot \sigma_S) \tag{4.20}$$

Der Kurvenverlauf von (4.20) unterscheidet sich, wie in Abbildung 4-6 zu sehen, vom idealisierten Kurvenbild aus Abbildung 4-5. Der Schattenbereich ist zu deutlich ausgeprägt, so dass bei hohen Inzidenzwinkeln die berechnete Rückstreuung zu niedrig ist. Durch einen Vergleich der Daten von *Ulaby und Dobson* mit dem berechneten Wert von σ° kann man versuchen, realistische Werte für *r* und *z* zu erhalten. Da der Kurvenverlauf der Gleichung dies nicht in jedem Fall ermöglicht, wird die Gleichung (4.20) um den Multiplikator *m* und den Potenzwert *q* erweitert.

$$\sigma^\circ = (((z \cdot \sigma_D) + ((1-z) \cdot \sigma_S)) \cdot m)^q \tag{4.21}$$

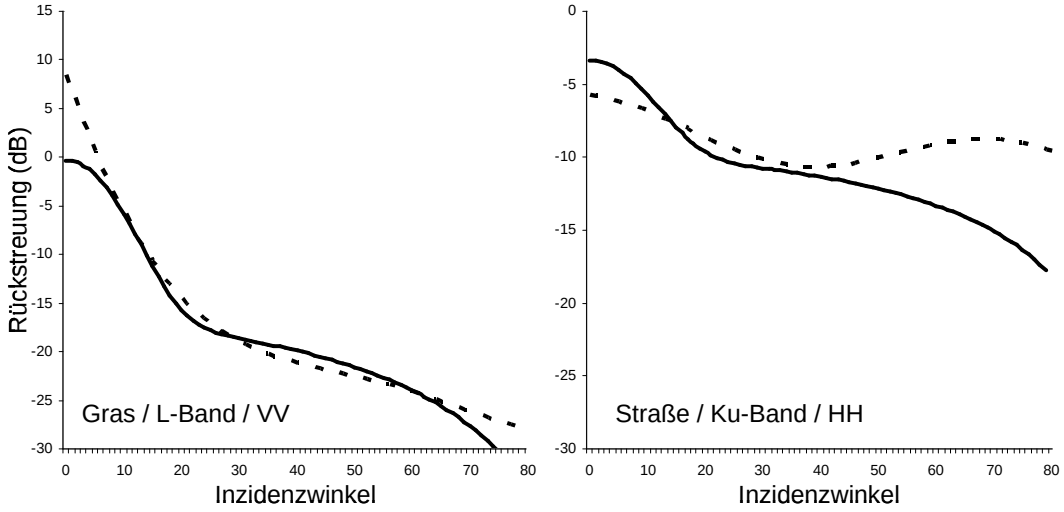

Abbildung 4-7: Reflexionsverhalten einzelner Materialien. Die durchgezogene Linie entspricht (4.21), die gestrichelte Linie entspricht (4.15)

Durch diese Änderung kann der Kurvenverlauf, wie in Abbildung 4-7 dargestellt, so angepasst werden, dass bei richtiger Wahl von *r, z, m* und *q* eine realistische und den statistischen Auswertungen entsprechende Funktion für σ° gefunden werden kann. Es können allerdings nicht alle Materialeigenschaften und Kurvenverläufe exakt abgebildet werden. Ein weiterer Nachteil dieser Vorgehensweise ist es, dass *m* und *q* keine physikalische Bedeutung haben und es ohne einen Vergleich mit Datenbanken kaum möglich ist, vernünftige Werte für diese Parameter festzulegen. Zwar sind *r* und *z* in ihrer Funktion als Reflexionsstärke und Rauigkeit verständlich, doch bleiben auch sie abstrakt und repräsentieren keinen realen, messbaren Wert. Auch müssen, entsprechend der Methode aus Kapitel 4.4.1.1, für jede Radarfrequenz und jede Polarisation eigene Werte bestimmt werden.

4.4.1.3 Zribis Modell zur Berechnung des Rückstreuquotienten

Zribi et al (2006) präsentieren ein Modell zur Berechnung der Rückstreuung. Dieses Modell benötigt nur wenige Parameter, die aus physikalischen Größen und Messungen abgeleitet werden können. Das Modell ist allerdings nur gültig für glatte bis leicht raue Oberflächen mit $ks<1{,}2$ und für große Inzidenzwinkel mit $\theta>35°$. Die Rückstreuung in der HH-Polarisation wird folgendermaßen berechnet (nach Zribi et al, 2006):

$$\begin{aligned}
\sigma^o_{hh} &= 3{,}2110^{p(\theta)} \cdot (kZs)^{q(\theta)} \cdot \frac{\cos\theta^4}{\sin\theta^3} \cdot {\alpha_{hh}}^2 \\
\alpha_{hh} &= \frac{\cos\theta - \sqrt{\varepsilon_r - \sin^2\theta}}{\cos\theta + \sqrt{\varepsilon_r - \sin^2\theta}} \\
q(\theta) &= 2{,}6289\theta^2 - 3{,}2561\theta + 1{,}969 \\
p(\theta) &= 2{,}303\theta^2 - 2{,}3217\theta
\end{aligned} \tag{4.22}$$

Die Rückstreuung in der VV- Polarisation berechnet sich aus:

$$\sigma_{vv}^{o} = 0,5 \cdot (kZs)^{0,84} \cdot \frac{\cos\theta^4}{\sin\theta^3} \cdot \alpha_{vv}{}^2$$
$$\alpha_{vv} = (\varepsilon_r - 1) \cdot \frac{\sin^2\theta - \varepsilon_r(1+\sin^2\theta)}{(\varepsilon_r \cos\theta + \sqrt{\varepsilon_r - \sin^2\theta})^2} \tag{4.23}$$

(nach Zribi et al, 2006)

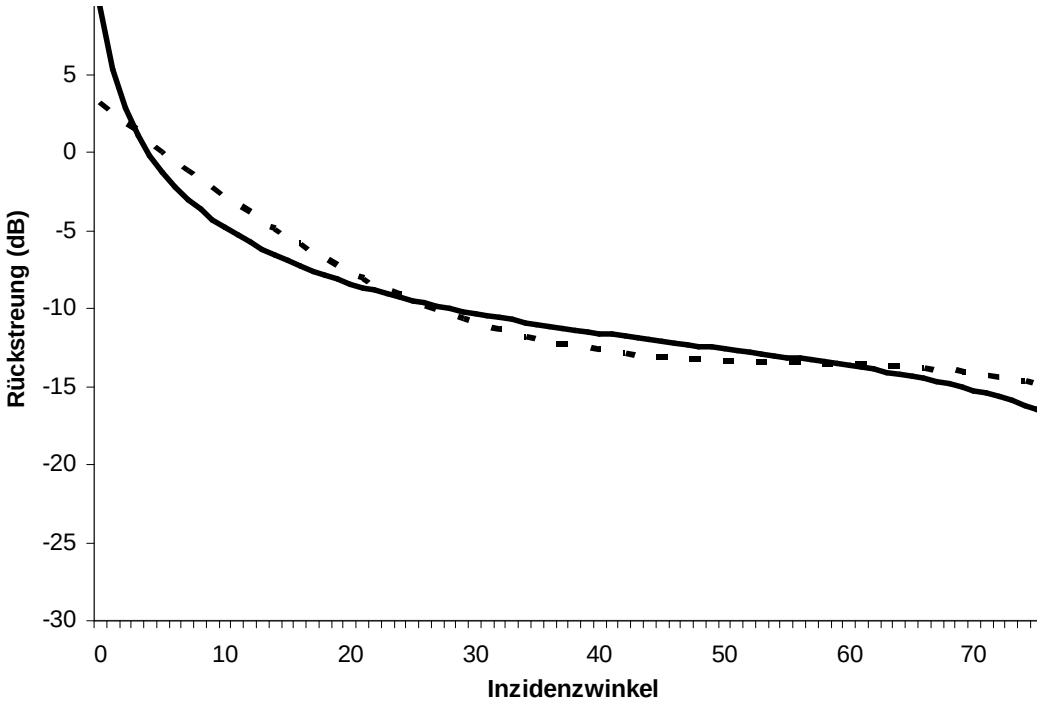

Abbildung 4-8: Verlauf der Rückstreuungsfunktion von Gras im C-Band (VV) nach Zribi et al (durchgezogene Linie) und nach Ulaby und Dobson (gestrichelt)

Die Wellenzahl *k* berechnet sich aus der Wellenlänge λ:

$$k = \frac{2\pi}{\lambda} \tag{4.24}$$

ε_r ist die relative dielektrische Konstante des Materials. *Zs* ist der Parameter, der die Rauigkeit der Materialoberfläche bestimmt. *Zs* wird aus dem quadratischen Mittelwert der Höhe *s* und der korrelierenden Länge *l* berechnet (Zribi & Dechambre, 2002).

$$Zs = \frac{s^2}{l} \tag{4.25}$$

Bei einer beispielhaft angenommenen Grasoberfläche ist, wie in Abbildung 4-8 zu sehen, der Funktionsverlauf vergleichbar mit den statistischen Ergebnissen der Gleichung (4.15). Das Model von *Zribi et al* bietet eine Reihe von Vorteilen. Für gegebene Materialeigenschaften kann die Rückstreuung für jede Radarfrequenz berechnet werden. Es sind nur wenige und physikalisch messbare Parameter notwendig. Obwohl das Modell nur eine begrenzte Gültigkeit hat, lässt es sich, wenn auch nur mit Einschränkungen bei der Simulationsgenauigkeit, für die Simulation verwenden.

Welches der in Kapitel 4.4.1 gezeigten Modelle das geeignetste ist, hängt von der Anwendung und den vorhandenen Ausgangsdaten ab. Neben den oben aufgeführten Rückstreumodellen lassen sich prinzipiell noch weitere Modelle implementieren.

4.4.1.4 Performanceunterschiede der unterschiedlichen Verfahren

Die Verfahren werden im Pixel-Shader unterschiedlich schnell berechnet. Das aufwendigste ist, wie in Tabelle 4-1 zu sehen, das Verfahren nach *Zribi*. Die Berechnung nach *Ulaby* ist zwar schneller, aber immer noch langsamer als die Berechnung der Gleichung (4.21). Zur Berechnung des Inzidenzwinkels θ, welcher bei der Berechnung nach *Ulaby* benötigt wird, muss ein *acos* berechnet werden. Dessen Berechnung ist relativ rechenzeitaufwendig. Bei der Gleichung (4.21) wird nur das Skalarprodukt verwendet und der tatsächliche Winkel muss nicht berechnet werden. Da sich die Werte in Tabelle 4-1 nur auf eine von mehreren Funktionen beziehen, ist der tatsächliche Unterschied der Gesamtperformance deutlich geringer.

Tabelle 4-1: Performanceunterschiede für eine NVIDIA GeForce 7900 GTX gemessen mittels NVShaderPerf (NVIDIA, 2005b)

	Ulaby (4.4.1.1)	***Computergrafik (4.4.1.2)***	***Zribi VV (4.4.1.3)***	***Zribi HH (4.4.1.3)***
Arith.Operationen	88	69	97	99
Pixeldurchsatz	400 MPix/s	457 MPix/s	369 MPix/s	355 MPix/s

4.4.2 Die Berechnung von Schatten

Beim Ray-Tracing werden Schattenwürfe automatisch mitberechnet, da Bereiche, die von den verfolgten Strahlen nicht erreicht werden, dementsprechend auch nicht beleuchtet werden. Bei der Rasterisierung hingegen müssen diese extra berechnet werden. Dazu gibt es zwei weit verbreitete Verfahren: Shadow Volume und Shadow Map.

4.4.2.1 Shadow Volume

Bereits 1977 publizierte *Crow* (1977) den Shadow Volume Algorithmus. Die sogenannten Shadow Volumes definieren den Bereich der, ausgehend von einer bestimmten Lichtquelle, im Schatten eines Objektes liegt. Um ein Shadow Volume zu berechnen, wird für jeden Punkt der äußeren Silhouette des schattenwerfenden Objektes eine Linie entlang der Geraden, die ausgehend von der Lichtquelle durch diesen Punkt führt, projiziert. Aus den Randpunkten des Objektes und den projizierten Punkten wird dann das Shadow Volume berechnet. Punkte innerhalb dieses Bereiches liegen im Schatten und sind unbeleuchtet, Punkte außerhalb sind beleuchtet.

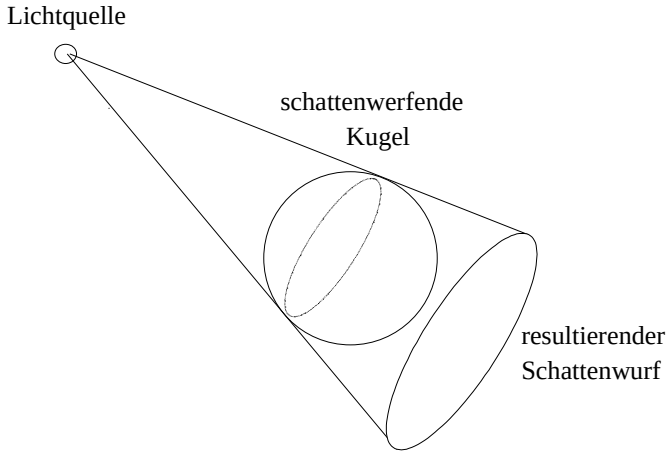

Abbildung 4-9: Verlängerung der Silhouette des verdeckenden Objektes

Unter Ausnutzung des heutzutage auf nahezu allen Grafikkarten implementierten Stencil-Puffers konnte *Heidmann* (1991) diesen Ansatz hardwarebeschleunigt implementieren. Unter dem Stencil-Puffer versteht man einen, neben Farb-Puffer und Z-Puffer, zusätzlichen dritten Speicherbereich im Bildspeicher moderner Grafikkarten. Stenciled Shadow Volumes können auf moderner Hardware schnell und robust berechnet werden (Everitt & Kilgard, 2002) und werden in einer Reihe von Anwendungen, unter anderem im kommerziell erfolgreichen Computerspiel „Doom 3“ (McGuire, 2004), eingesetzt.

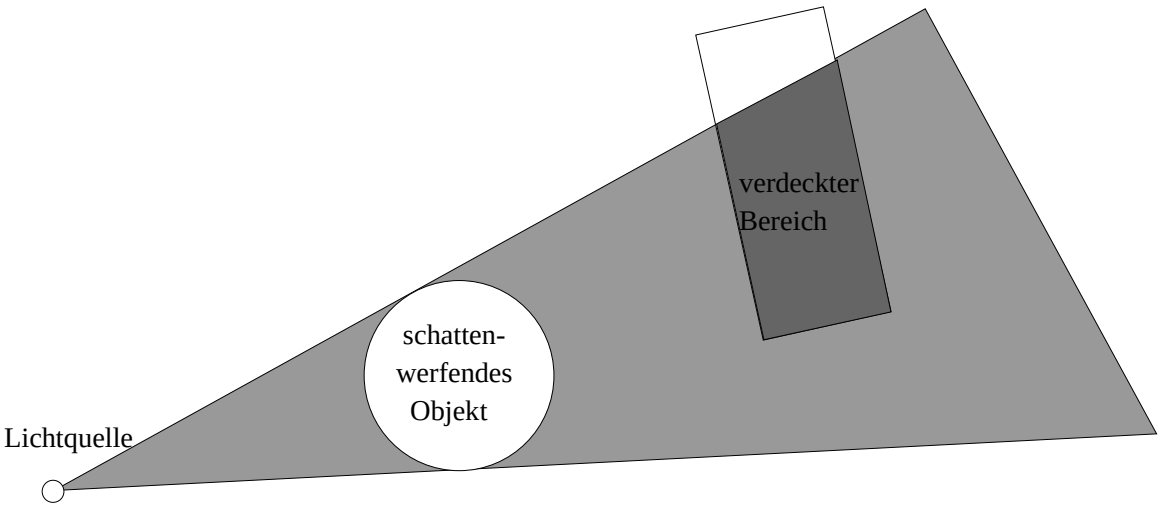

Abbildung 4-10: Schematische Darstellung eines Shadow Volume

4.4.2.2 Shadow Map

Shadow Map ist eine bildbasierte Technik, die von *Williams* (1978) publiziert wurde. Shadow Maps sind hardwarenah implementierbar, da sie, im Gegensatz zu Shadow Volumes, keine geometrische Prozessierung benötigen, sondern bildbasiert sind. Beim Shadow-Mapping wird, wie in Abbildung 4-11 illustriert, das Bild zunächst aus Sicht der Lichtquelle gerendert, d.h. die virtuelle Kamera wird auf der Lichtquelle positioniert und blickt in Richtung des Lichtstrahls. Statt des gerenderten Bildes interessieren in der weiteren Prozessierung jedoch nur die im Z-Puffer gespeicherten Tiefeninformationen. Der Z-Puffer entspricht bereits einem Sichtbarkeitstest und wird als Shadow Map bezeichnet. Die Shadow Map wird daraufhin in die virtuelle Szene transformiert und für jedes zu zeichnende Pixel wird, anhand der projizierten Shadow Map, überprüft, ob dieses Pixel im Schattenbereich liegt oder nicht.

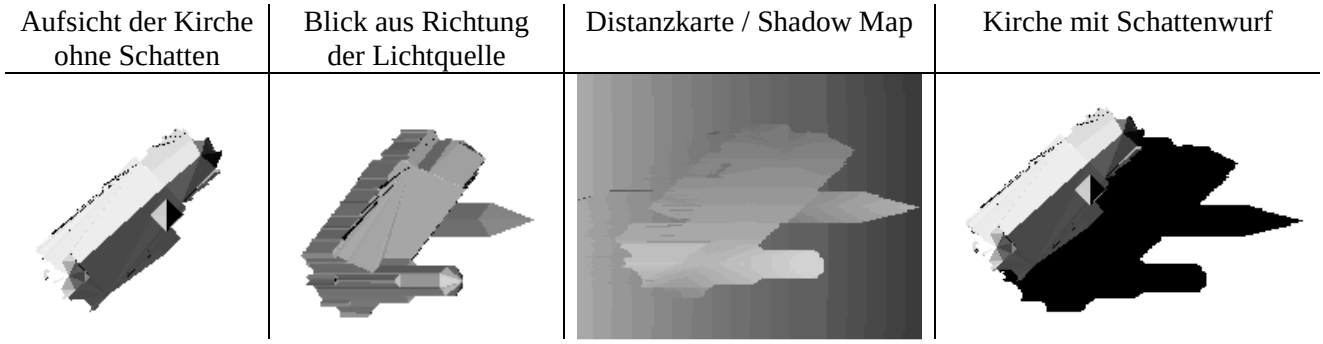

Abbildung 4-11: Schattenberechnung mittels Shadow Map

Auf Grund der begrenzten Genauigkeit bei der Berechnung der Shadow Maps kann es zu Ungenauigkeiten und Artefakten bei der Schattenbildung kommen. Die Shadow Map sollte daher mit einer 32-bit floating-point Genauigkeit berechnet werden, um die Abstände der Punkte genauer abspeichern zu können und die Fehler zu reduzieren. Des Weiteren ist es von Vorteil, wenn bei der Sichtbarkeitsüberprüfung ein Schwellwert eingeführt wird, bzw. die Polygone versetzt werden (polygon offset). Die Ungenauigkeit der floating-point Variablen und die dadurch bedingten Schwierigkeiten, zwei floating-point Werte direkt miteinander zu vergleichen, machen diesen Schritt notwendig.

Shadow Maps funktionieren optimal mit gerichteten Strahlungsquellen, können jedoch nicht ohne weiteres die Schatten von Rundstrahlern abbilden, was wiederum mit Shadow Volumes problemlos möglich ist. Shadow Maps funktionieren gut, wenn die Lichtquelle und die virtuelle Kamera ungefähr die gleiche Blickrichtung und Position haben (Everitt et al, 2001) und die Lichtquelle nicht zu weit von der Szene entfernt liegt. Es gibt eine Reihe von Verbesserungsvorschlägen für die unterschiedlichen Probleme der Shadow Map. So bieten sowohl *Fernando et al* (2001), *Stamminger und Drettakis* (2002) als auch *Martin und Tan* (2004) Vor-

schläge zur Lösung bzw. Verbesserung des Auflösungsproblems der Shadow Maps. Verbesserungen des Polygon-Offset Problems liefern *Wang und Molnar* (1994). Shadow Maps sind einfach und hardwarenah zu implementieren und erlauben es, Schatten schnell zu berechnen.

Mit Hilfe der Shadow Map ist es nicht nur möglich, die Schattenwürfe zu berechnen. Die Shadow Map kann auch zur Sichtbarkeitsanalyse eingesetzt werden und löst damit das Sichtbarkeitsproblem bei der SAR-Visualisierung (vgl. Kapitel 4.3.3). Im ersten Schritt wird, genauso wie bei der Shadow Map, die Szene mit der virtuellen Kamera auf der Sensorposition in die Sensor Map gerendert. In die Sensor Map wird der Abstand des gerenderten Punktes zum Sensor eingetragen, vorausgesetzt die Sensor Map enthält an der betreffenden Stelle nicht bereits einen niedrigeren Wert (vgl. Z-Pufferung in Kapitel 3.3). Dadurch enthält die Sensor Map am Ende des Rendering die Entfernungen, die vom Sensor aus gesehen am Nächsten zum Sensor liegen. Die Sensor Map wird als 32-bit floating-point Textur gespeichert. Um den vollen Werteraum der Textur auszunutzen, wird nicht der absolute Abstand in die Sensor Map geschrieben, sondern ein normierter Wert, der sich folgendermaßen berechnet:

$$d = \frac{|position - sensorPosition| - nearRange}{farRange - nearRange} \tag{4.26}$$

Im zweiten Schritt wird die Szene erneut gerendert, diesmal mit der virtuellen Kamera in Nadirposition, da direkt ein Ground-Range-Bild erzeugt werden soll. Für jeden zu rendernden Punkt wird überprüft, ob die Entfernung dieses Punktes zum Sensor größer ist, als die in der Sensor Map gespeicherte Entfernung. Die Transformation der Sensor Map in die veränderte Projektion erfolgt direkt in den Shadern, d.h. auf der Grafikkarte. Die Punkte mit einer größeren Entfernung sind verdeckt und werden nicht gerendert.

Die Ungenauigkeiten bei der Entfernungsbestimmung verlangen, wie oben beschrieben, einen Polygon-Offset, bzw. einen Schwellwert. Ein ungünstiger Schwellwert führt dabei entweder zu fehlendem Schattenwurf bei Objekten mit geringer Höhe oder zu Artefaktbildungen bei Flächen, welche beinahe im Schattenbereich liegen, wie in Abbildung 4-12 illustriert.

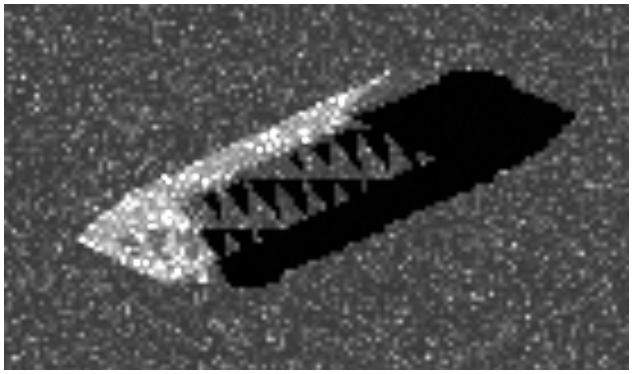

Abbildung 4-12: Haus mit Schattenartefakten im Bereich des Daches

4.4.3 Realistischere Visualisierungen durch „weiche Schatten“

Die Schatten, die mit der unter Kapitel 4.4.2 beschriebenen Methode erzeugt werden, sind an ihren Rändern zu hart. Die Ränder sind sozusagen binär, d.h. entweder ein Pixel liegt völlig im Schatten oder überhaupt nicht. Das spiegelt jedoch die Realität nicht wieder. Durch die Keulenform des Radarstrahls werden die Schattenränder zumindest zeitweise während des Vorbeifluges beleuchtet, so dass der Schattenübergang fließend ist. Um dies zu visualisieren, werden sogenannte „weichen Schatten“ benötigt.

Um weichere Radarschatten darzustellen, werden drei Sensor Maps erzeugt. Eine Sensor Map wird mit der Azimutposition des Sensors in der Bildmitte erzeugt, zwei weitere jeweils am Bildanfang und Bildende. Die exakte Position für eine realistische Visualisierung hängt von der Radarkeulenbreite ab. Bei der Schattenvisualisierung wird dreimal überprüft, ob ein Punkt im Schatten liegt (vgl. Kapitel 4.4.4). Liegt ein Punkt nur in einem der Schattenbereiche, so wird er schwach beleuchtet dargestellt, d.h. seine Reflexionsstärke wird reduziert. Dadurch werden, wie in Abbildung 4-13 zu sehen, die Ränder weicher und die Schatten abgerundet.

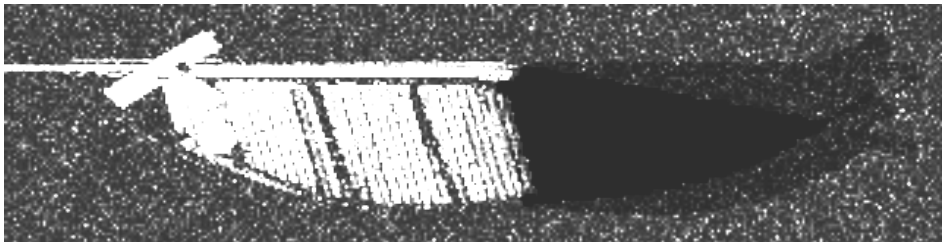

Abbildung 4-13: Visualisierung eines Modells des „Burj-Al-Arab Dubai“ mit weichen Schatten

4.4.4 Addition der Reflexionen gleicher Laufzeit

Die Reflexionen der vom Sensor einsehbaren Punkte, welche die gleiche Entfernung zum Sensor in Entfernungsrichtung haben, addieren sich. Punkte, die vom Sensor abgeneigt oder verdeckt sind, reflektieren nicht, weswegen sich die Reflexionsstärke in diesem Fall nicht weiter erhöht. Zusätzlich kann es noch in jedem einzelnen Punkt zu thermischem Rauschen kommen, welches zur Gesamtreflexion addiert werden muss. Da die einzelnen Schritte aufeinander aufbauen und die Ergebnisse der vorhergehenden Schritte die nachfolgenden beeinflussen, werden die Ergebnisse der Einzelschritte nicht direkt auf dem Bildschirm ausgegeben, sondern zunächst in einem Speicherbereich der Grafikkarte, dem sogenannten Texturspeicher, zwischengespeichert.

Die Addition von Werten auf der gleichen Pixelposition lässt sich auf zweierlei Art realisieren. Die beste Lösung ist das Alpha-Blending. Hier werden die unterschiedlichen Pixel in Abhängigkeit ihres sogenannten Alphawertes überlagert. Leider unterstützen zurzeit nicht alle Grafikkarten das Alpha-Blending auf 16-bit Texturen. Noch weniger unterstützen das Alpha-Blending auf 32-bit Texturen. Verwendet man Texturen mit drei Farbwerte, also rot, grün und blau, mit jeweils 16-bit, kann der Werteraum eines einzelnen 32-bit Wertes leicht abgedeckt werden.

Beim Alpha-Blending berechnet sich die resultierende Reflexion σ aus dem sich bereits im Texturspeicher befindlichen Reflexionswert σ_z, dem hinzuzuaddierenden Reflexionswert σ_q, sowie den zugehörigen Blendwerten *b*.

$$\sigma = (\sigma_q \cdot b_q) + (\sigma_z \cdot b_z) \tag{4.27}$$

Im Normalfall werden b_q und b_z aus dem Alphawert α_q von $\sigma_q s$ gebildet, so dass gilt:

$$\sigma = (\alpha_q \cdot \sigma_q) + ((1-\alpha_q) \cdot \sigma_z) \tag{4.28}$$

Bei der 3D-Visualisierung entstehen hierbei Probleme, da die Reihenfolge, in der die Objekte gezeichnet werden die endgültige Farbe mitbestimmt. Die Reihenfolge ist jedoch im Normalfall unbekannt. Zur Lösung dieses Problems wird das sogenannte additive Alpha-Blending verwendet. Dies kann mit $b_z = 1$ einfach realisiert werden:

$$\sigma = (\alpha_q \cdot \sigma_q) + \sigma_z \tag{4.29}$$

Hierbei ist zu beachten, dass negative Beleuchtungsstärken, die bei der Berechnung der Reflexionsstärke von vom Sensor abgeneigten Flächen entstehen, nicht berücksichtigt werden. Ein Pixel nicht zu zeichnen, kann man in HLSL mit dem Befehl „Clip“ erreichen. Dieser Befehl ist jedoch vergleichsweise langsam. Die dynamischen Verzweigungen sind in der Ausführung ebenfalls vergleichsweise langsam. Beim additiven Alpha-Blending kann man durch ein Setzen der Beleuchtungsstärke $\sigma_q = 0$ dasselbe erreichen. Um alle negativen Werte auf *0* zu setzen, kann man den Befehl:

$$\sigma = \frac{\sigma + |\sigma|}{2} \tag{4.30}$$

einsetzen.

Für Punkte, die im Schatten liegen, soll ebenfalls $\sigma=0$ gelten. Um Verzweigungen im Pixel-Shader zu vermeiden, wird die in der Shadow Map gespeicherte Distanz von der Distanz aus Nadirrichtung abgezogen. Bei negativen Werten liegt der Punkt im Schatten, bei positiven nicht. Die resultierende Reflexion berechnet sich aus:

$$\begin{aligned} d &= d_{Nadir} - d_{ShadowMap} - c_{Threshold} \\ v_s &= (d - |d|)^{(d+|d|)} \\ \sigma_{erg} &= \sigma \cdot v_s \;\Big|\; \text{mit } v_s = 0 \;;\; 1 \end{aligned} \tag{4.31}$$

Durch Anwendung von (4.30) und (4.31) lässt sich die Ausführungsgeschwindigkeit gegenüber dynamischen Verzweigungen um ca. 20% steigern. Dieser Wert bezieht sich auf die Analyse einer Funktion durch NVIDIAs „NvShaderPerf“ (NVIDIA, 2005b). Da jedoch bei der Rasterisierung eine Reihe von Funktionen aufgerufen werden, ist die Gesamtperformance weit weniger betroffen.

4.4.5 Pixelbasierte Reflexionsberechnung mittels Objekttexturen

Neben der Möglichkeit pro Bildprimitiv, d.h. pro Dreieck, ein Material mit entsprechenden Reflexionswerten zu definieren, können mittels Texturen auch mehrere Materialien pro Dreieck definiert werden. Vergleichbar mit der Darstellung virtueller Szenen mittels Texture-Mapping, wie z.B. in Computerspielen, können so weit detailliertere Informationen pro Modell dargestellt werden. Anstatt Farbinformationen beinhalten die Texturen jedoch Reflexionswerte. Um diese Werte exakt zu speichern, sollten Texturen mit 16-bit oder 32-bit Werten pro Farbkanal verwendet werden.

Um die zur Berechnung der Reflexionsstärke benötigten zwei (nach *Zribi*) bis sechs (nach *Ulaby und Dobson*) Parameter zu repräsentieren, müssen in Abhängigkeit der gewählten Berechnungsmethode ein oder zwei Texturen verwendet werden. Die Festlegung der Reflexionsparameter per Textur ermöglicht z.B. die Verwendung eines grob aufgelösten Dreiecknetzes zur Darstellung des Untergrundes. Dieses Modell wird dann, wie in Abbildung 4-14 zu sehen, durch die Texturinformation ergänzt. Dies ist mit der 3D-Visualisierung von Landschaftsmodellen durch ein mit Orthophotos texturiertes DHM vergleichbar. Dadurch können auf einem groben Modell sehr feine Strukturen wie z.B. Straßen abgebildet werden. Grafikhardware ist auf diese Form der Datenrepräsentation spezialisiert und kann die für die jeweilige Position gültigen Parameterwerte sehr schnell aus der Textur bzw. aus den Texturen lesen.

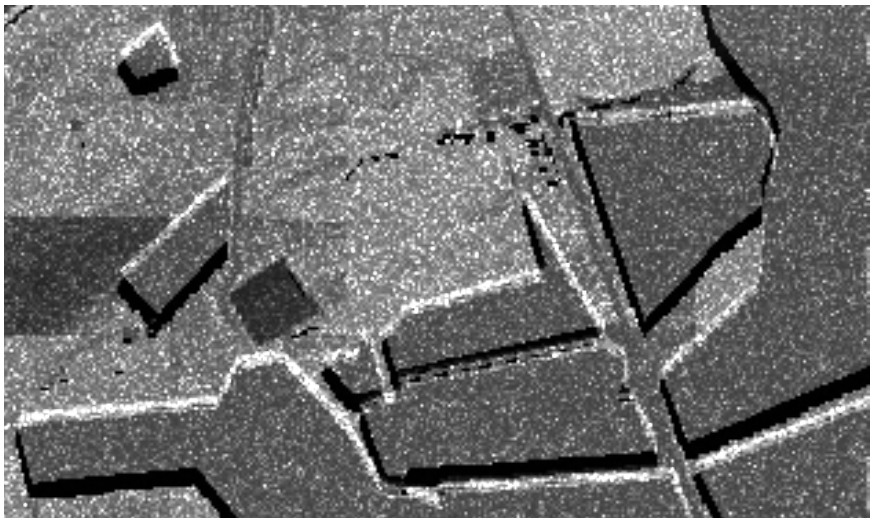

Abbildung 4-14: SAR-Simulation einer texturierten Szene

Neben radiometrischer Information kann die Textur auch Informationen zur Objektform enthalten. Diese sogenannten „Billboards“ dienen dazu, komplizierte Geometrien schnell darzustellen (McReynolds & Blythe, 2005). Die komplexe Geometrie wird durch eine teilweise transparente Textur visualisiert. In Abbildung

4-15 wird diese Möglichkeit exemplarisch dargestellt. Die Geometrie des links abgebildeten Baumes besteht nur aus einem Viereck, auf das die Baumtextur aus Abbildung 4-15 (rechts) aufgebracht wurde. Der helle Bereich der Textur ist transparent. Dies muss bei der Berechnung der Reflexion und des Schattenwurfes beachtet werden. Für eine realistische Darstellung ist das Billboard immer von vorne zu betrachten. Bei der Darstellung von Bäumen werden normalerweise Axiale Billboards verwendet (Akenine-Möller & Haines, 2002), d.h. die Billboards rotieren um eine Achse im Weltkoordinatensystem und richten sich zum Sensor aus. Durch diese Drehung in Sensorrichtung und die Transparenz der Textur entsteht der Eindruck einer komplexeren Geometrie.

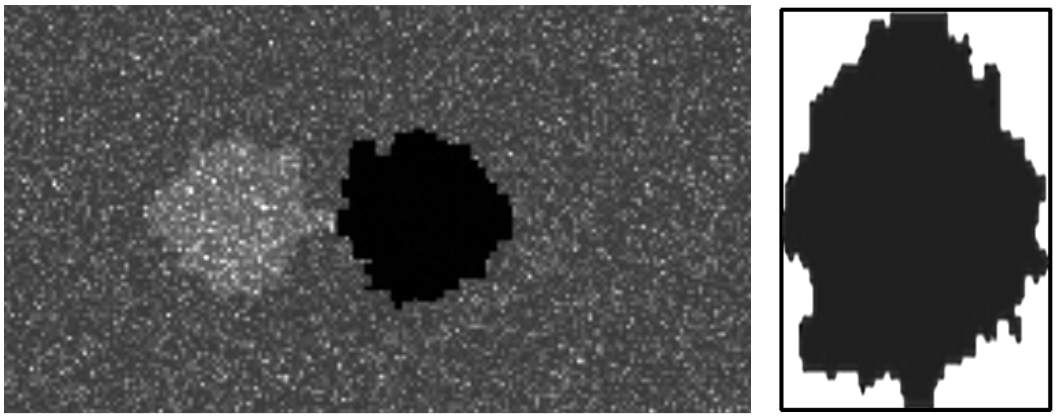

Abbildung 4-15: Formrepräsentation durch transparente Texturen

Die Verwendung von Texturen mit hoher radiometrischer Auflösung kann allerdings schnell zu Speicherproblemen führen. Ausgehend von einer zu visualisierenden Szene mit einer Ausdehnung von einem Quadratkilometer und einer Texturauflösung von einem halben Meter, werden ein bis zwei Texturen mit jeweils 2000 mal 2000 Pixel und vier Mal 32-bit pro Pixel gespeichert. Dies entspricht 61 bzw. 122 Megabyte.

Durch Verwendung von 16-bit pro Pixel kann der Speicherbedarf ohne großen Informationsverlust halbiert werden. Eine weitere Möglichkeit ist die Verwendung von Materialdatenbanken. Diese können in Texturen mit 32-bit Auflösung hinterlegt werden. Eine kleine Textur mit 256 mal 256 Werten kann bereits 65536 unterschiedliche Materialien speichern und benötigt nur 0,5 Megabyte Speicherplatz. Die Textur eines Objektes enthält dann nicht mehr die Reflexionsparameter, sondern die Texturposition des Materials in der Texturdatenbank. Die Texturposition eines Materials kann, bei einer Texturdatenbankgröße von 256 mal 256 Werten, in einer 8-bit Textur gespeichert werden. Im obigen Beispiel verbraucht die 2000 Pixel mal 2000 Pixel große Textur dann nur noch 15,3 Megabyte plus der 0,5 Megabyte für die Materialdatenbank. Allerdings verlangsamt dieses Vorgehen die Visualisierung der Daten, da statt ein bzw. zwei Texturzugriffe nun zwei bzw. drei Texturzugriffe pro Pixel erfolgen müssen.

4.4.6 Die Berechnung des Speckling

Das Speckling kann als ein multiplikatives Rauschen betrachtet werden. Zwar ist diese Sichtweise vereinfachend, jedoch für die SAR-Effektvisualisierung hinreichend realistisch. Wenn eine große und nahezu gleichmäßige Zahl von Streuzentren pro Auflösungszelle N vorausgesetzt wird, dann entspricht das reflektierte elektrische Feld einem Gauß'schen Feld. Die beobachtbare Amplitude entspricht der Rayleigh-Verteilung (Delignon & Pieczynski, 2002). Eine stetige Zufallsvariable x ist Rayleighverteilt, wenn für sie gilt:

$$f(x|\sigma) = \frac{x}{\sigma^2} e^{\left(\frac{-x^2}{2\sigma^2}\right)} \tag{4.32}$$

Ein gleichförmig verteiltes Feld von Zufallszahlen z, wie es vom Zufallsgenerator der CPU erzeugt wird, kann in ein Feld, welches der Rayleighverteilung entspricht, umgewandelt werden (Bolter et al, 1996). Wenn für die Verteilungsdichte $\sigma = 1$ gilt, dann gilt für die Rayleighverteilten Specklewerte s:

$$s = -\ln(1 - z) \tag{4.33}$$

Da *z* eine Zufallszahl mit $0 \le z \le 1$ ist, kann (4.33) vereinfacht werden zu:

$$s = -\ln(z) \tag{4.34}$$

4.4.7 Berechnung von Multilooks

Im Multilookverfahren setzt sich der Reflexionswert aus *M*–Looks, also aus *M* einzelnen SAR-Aufnahmen zusammen. Der resultierende Specklewert im Multilook s_M berechnet sich somit aus:

$$s_M = \frac{1}{M} \sum_{i=1}^{M} s_i \tag{4.35}$$

Der Specklewert *s* beschreibt jedoch nicht den Reflexionswert einer Auflösungszelle. Dieser hängt von der unter 4.4.1 beschriebenen Berechnung für $\sigma°$ ab. Der endgültige Reflexionswert σ_s^o einer Auflösungszelle wird berechnet aus:

$$\sigma_s^o = \sigma^o + \sigma^o \cdot s_M \tag{4.36}$$

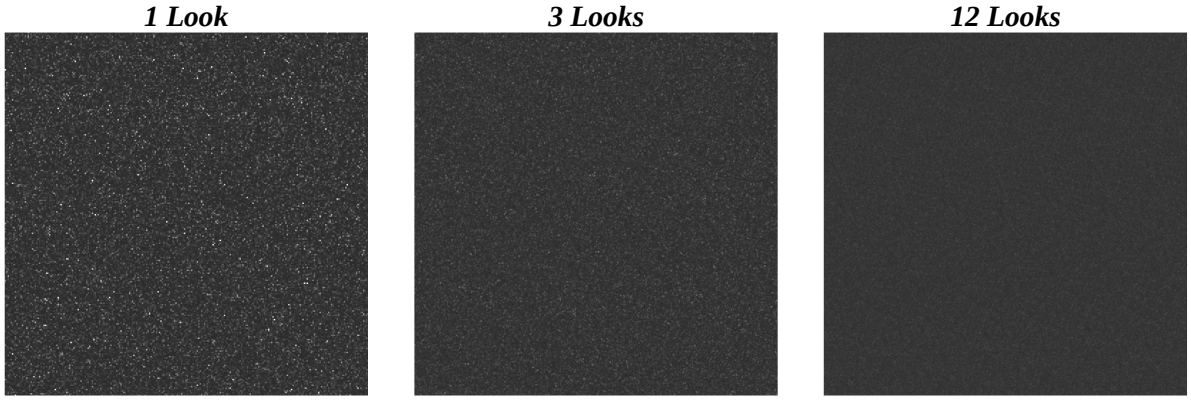

Abbildung 4-16: Auswirkungen von Looks auf das Speckling

Multilookszenen durch bloße Veränderung des Specklewertes zu visualisieren ist allerdings eine starke Vereinfachung. Zwar ist diese Herangehensweise für viele Anwendungen ausreichend genau, allerdings lässt sich die Realitätstreue der Visualisierung verbessern, wenn die einzelnen Looks komplett simuliert und visualisiert werden. Im Multilookbild wird die synthetische Apertur in einzelne Teilaperturen unterteilt, aus denen ein räumlich schwächer aufgelöstes Teilbild berechnet wird. Das Mittel aus diesen Teilbildern ist ein Multilookbild, welches durch die Verminderung des Specklings eine höhere radiometrische Auflösung hat.

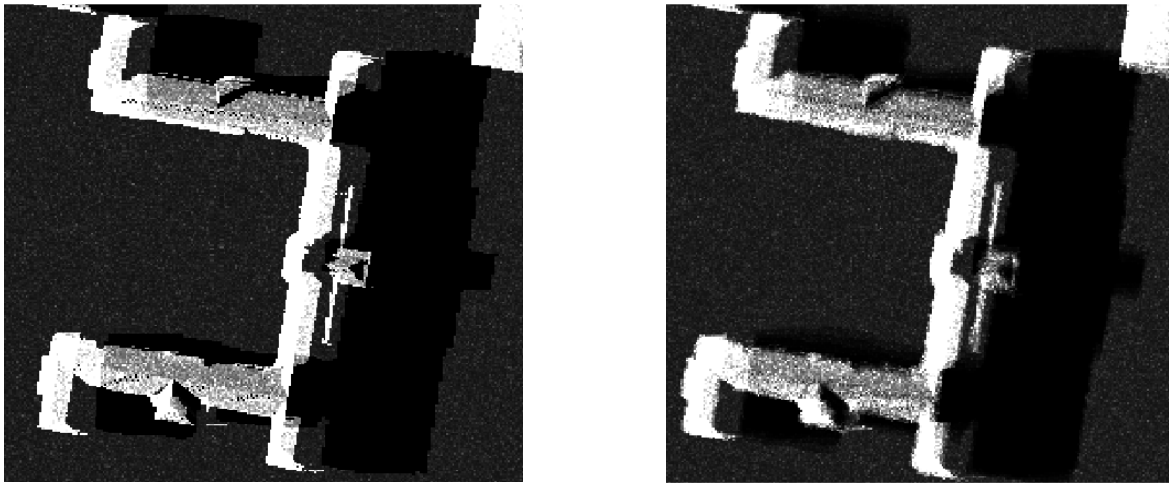

Abbildung 4-17: Bild mit vereinfachtem Multilook (links) und separat berechnetes Multilookbild (rechts)

Die einzelnen Looks unterscheiden sich durch ihren Squintwinkel. Die Simulation der einzelnen Looks mit leicht unterschiedlichen Squintwinkel und anschließender Mittelwertsbildung erlaubt die Visualisierung der Multilookeffekte für Schattenwurf und Layoverbildung und daher eine realistischere Visualisierung von Multilookbildern. Wie in Abbildung 4-17 dargestellt, unterscheiden sich separat berechnete Multilookbilder von vereinfachten Multilookbildern insbesondere hinsichtlich der Unschärfe an den Schattenrändern und im Layoverbereich. Schattenränder verwischen, da einzelne Looks seitlich den Schattenrand beleuchten. Die Layoverbereiche verwischen sich, wie in Abschnitt 2.3.3.1 beschrieben, durch die unterschiedlichen Layoverrichtungen in Abhängigkeit der einzelnen Squintwinkel.

4.4.8 Nebenkeulen und deren Visualisierung

Normalerweise wird nur ein Bruchteil der vom SAR-System ausgestrahlten Energie zum Sensor zurück gestreut. In Ausnahmefällen kann es jedoch vorkommen, dass ein großer Teil der Energie zurück zum Sensor reflektiert wird. In diesem Fall entstehen Nebenkeulen in Azimut- und Rangerichtung, welche für SAR-Bilder typisch sind (siehe Abbildung 4-18 und Abbildung 2-15).

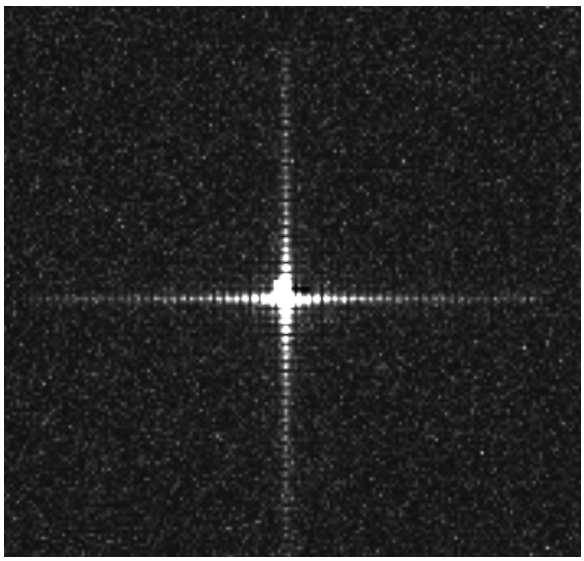

Abbildung 4-18: Überstrahler mit Nebenkeulen

Um Details in schwach und stark belichteten Bereichen erkennen zu können, sind auch bei digitalen Bildern die üblichen 256 Graustufen pro Farbkanal nicht ausreichend. Deswegen unterstützen moderne Grafikkarten das „high dynamic range (HDR) lighting“ (siehe Kapitel 3.4). Die Berechnungen der Beleuchtungsintensität wird in 16-bit oder 32-bit Auflösung vorgenommen. Das Ergebnis wird dann für die Ausgabe auf dem Monitor auf 8-bit optimiert. Dies ermöglicht es, Beleuchtungen realistisch darzustellen und vermeidet schwache Kontraste im dargestellten Bild. Dadurch ist es z.B. in einem Computerspiel möglich, eine allmähliche Anpassung des Auges an helle oder dunkle Beleuchtungsverhältnisse zu visualisieren.

Die Entwicklung des HDR ist entscheidend für die SAR-Simulation, da erst die Einführung von 32-bit Texturen die nötige dynamische Bandbreite liefert. Das Ergebnis der SAR-Simulation wird zunächst auf eine 32-bit Textur gerendert. Zur Ausgabe auf den Bildschirm wird die 32-bit Farbinformation auf 8-bit angepasst, wobei hierfür sowohl lineare als auch logarithmische Verfahren angewendet werden können.

Bevor die endgültige Ausgabe auf dem Bildschirm erfolgt, werden in einem auf die Simulation folgenden Bearbeitungsschritt die Überstrahlungen visualisiert. Überschreitet ein Wert im Simulationsergebnis einen angegebenen Schwellwert, dann wird von einer Überstrahlung ausgegangen. Für die gefundenen Überstrahler werden, in Abhängigkeit der Reflexionsstärke, Texturen von Nebenkeulen erzeugt und an den entsprechenden Positionen gerendert. Die Visualisierung von Überstrahlungen muss auch den Squintwinkelfall beachten. Die Nebenkeule in Rangerichtung ist in diesem Fall gedreht, wohingegen die Nebenkeule in Azimutrichtung unverändert bleibt.

4.5 Visualisierung bistatischer SAR-Szenen

Die Visualisierung bistatischer SAR-Szenen erfordert die Einführung einer weiteren Sensorposition (siehe Kapitel 2.5). Der bistatische Empfänger entspricht dem bisherigen Sensor. Der zusätzliche bistatische Sender wird in Abhängigkeit zum Empfänger positioniert, d.h. der simulierte bistatische Sender ist nicht frei positionierbar. Dadurch ist nur eine beschränkte Bandbreite bistatischer Missionen darstellbar. Diese werden allerdings korrekt abgebildet und kritische Sonderfälle werden dadurch vermieden. Die Position des Senders wird beschrieben durch die bistatische Winkelabweichung von der Rangerichtung und der Abweichung des Off-Nadirwinkels.

Im bistatischen Fall verändern sich sowohl Geometrie als auch Radiometrie der Szene. Da beobachtete Objekte sowohl vom Sender als auch vom Empfänger aus sichtbar sein müssen, kommt es zu vermehrtem Schattenwurf. Um dies zu visualisieren, werden zwei Shadow Maps berechnet. Bei der Berechnung der Radiometrie äquivalent zum optischen Fall (siehe Kapitel 4.4.1.2) muss die Position des Senders ebenso beachtet werden. In diesem Fall entspricht der Sender der Lichtquelle und der Empfänger der virtuellen Kamera. Für die Berechnung der spiegelnden Reflexion gilt jetzt nicht mehr, dass Sender und Empfänger identisch sind. Die Vereinfachung der Gleichung (4.18) zu (4.19) in Kapitel 4.4.1.2 ist daher ungültig. Stattdessen muss für die Berechnung der Reflexionsstärke zunächst der Halbvektor $\vec{h}$ berechnet werden.

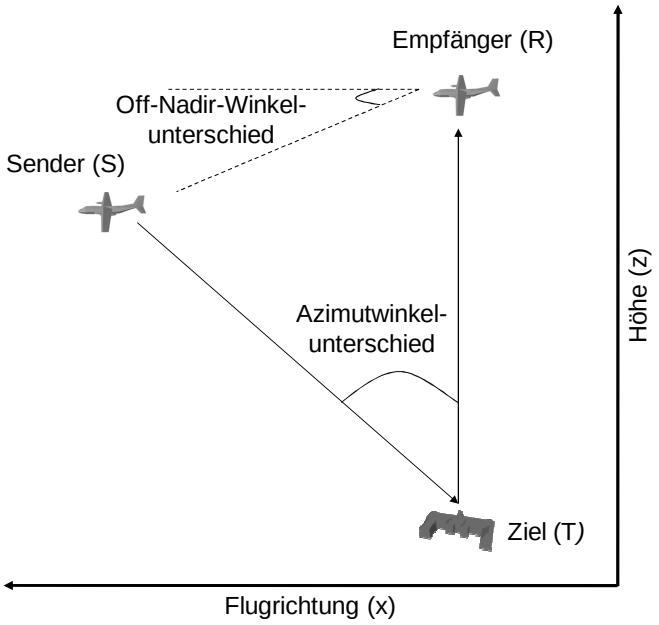

Abbildung 4-19: Parametrisierung des bistatischen Senders

Die bistatische SAR-Geometrie unterscheidet sich von der monostatischen unter anderem durch die unterschiedlichen Off-Nadirwinkel von Sender und Empfänger. Das Ausmaß der Verschiebung in Rangerichtung in Abhängigkeit von der Objekthöhe hängt vom Off-Nadirwinkel ab. Im bistatischen Fall setzt sich die Laufzeit zusammen aus der Entfernung vom Sender zum Objekt plus der Entfernung vom Objekt zum Empfänger. Bei der Berechnung der Verschiebung in Rangerichtung sind daher die Off-Nadirwinkel von Sender und Empfänger zu berücksichtigen.

In Abbildung 4-20 sind drei Beispiele einer bistatischen SAR-Effektvisualisierung abgebildet. Im linken Fall befindet sich der SAR-Sender hinter dem Empfänger, im mittleren Fall knapp unterhalb des Empfängers und rechts ist der Sender vor dem Empfänger. Stark unterschiedlich sind insbesondere die Schattenregionen. Bei der Objekterkennung und Interpretation kann dieser zusätzliche Schatten hilfreich sein. In vielen Fällen ist die Zunahme an Schattenbereichen hinderlich, da insgesamt weniger Bildbereiche sichtbar sind. Die bessere Interpretierbarkeit wird in der Praxis oftmals dadurch erschwert, dass die zu erkennenden Objekte auf spiegelnden Oberflächen stehen und die Schattenbereiche kaum von der dunklen Umgebung zu unterscheiden sind.

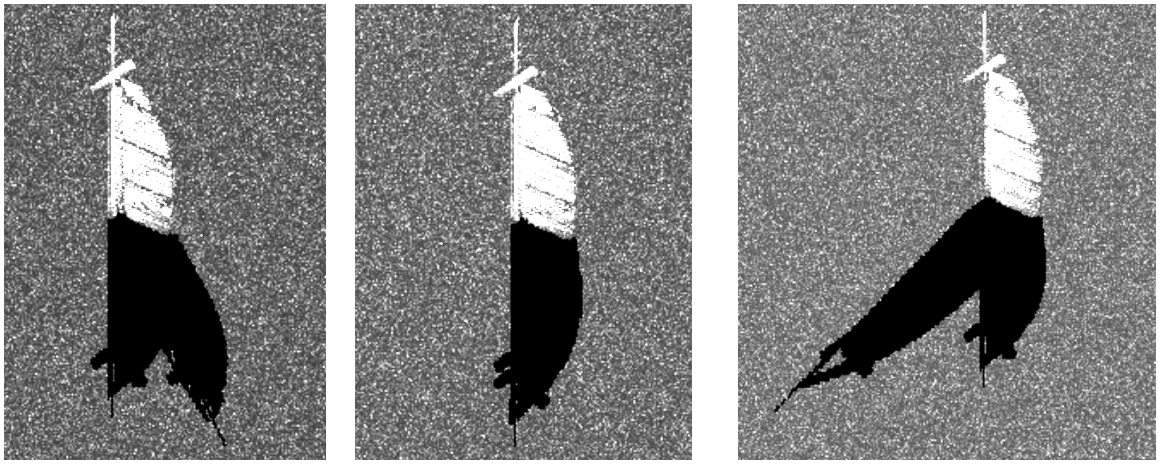

Abbildung 4-20: Bistatische Simulation des Burj-Al-Arab-Hotels mit dem Sender hinter (links), neben (Mitte) und vor (rechts) dem Empfänger

4.6 Besonderheiten des Spotlightmodus

Im Spotlightmodus wird die SAR-Antenne dynamisch nachgeführt, um einen bestimmten Bereich länger beobachten zu können. Dadurch wird die synthetische Apertur weiter vergrößert und eine höhere Bodenauflösung ist möglich. Der Squintwinkel der Antenne ändert sich dynamisch im Verlauf der Aufnahme. Bei der Visualisierung kann daher kein fixer Squintwinkel angegeben werden. Darüber hinaus ändert sich die Beleuchtungsrichtung in Abhängigkeit von der Azimutposition.

Um diesen Modus zu visualisieren, muss sowohl die Erzeugung der Sensor Map als auch die Berechnung des Layovers angepasst werden. Die Beleuchtungsrichtung ändert sich dynamisch in Abhängigkeit der Azimutposition im Bild. Hierfür werden die unter Kapitel 4.4.3 beschriebenen weichen Schatten erweitert. Es werden drei Sensor Maps erzeugt. Die Sensorposition für jede Sensor Map richtet sich nach den Start- und End-Squintwinkel des Spotlightmodus. Im Gegensatz zu den weichen Schatten sind jedoch nicht alle Sensor Maps gleichberechtigt. Die Bedeutung der einzelnen Sensor Maps zur Bestimmung des Schattens ändert sich über den Verlauf des Bildes. In der Bildmitte ist die mittlere Sensor Map und an den Bildendpunkten die jeweilige Start- oder End-Sensor-Map ausschlaggebend.

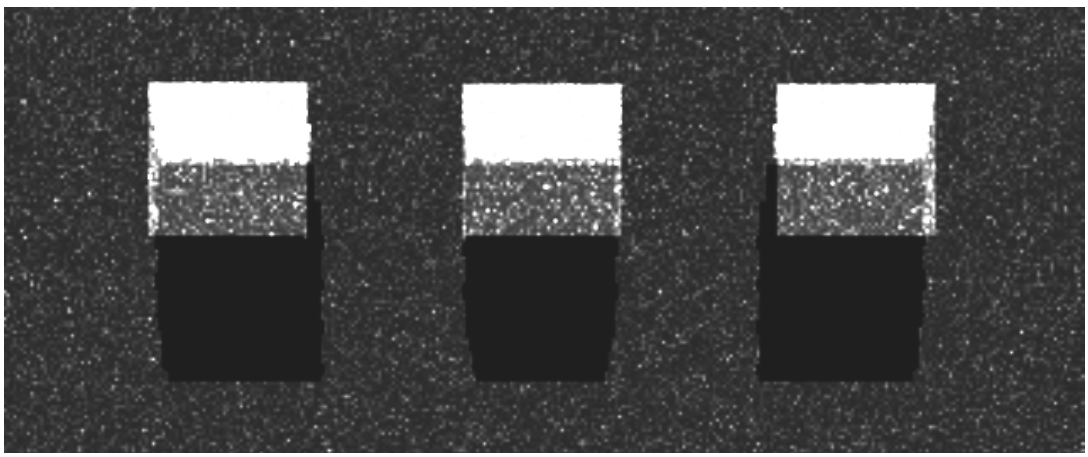

Abbildung 4-21: Visualisierung dreier Flachdachhausmodelle im Spotlight-Modus

Neben dem Schattenwurf ändert sich, bedingt durch den sich dynamisch im Bildverlauf ändernden Squintwinkel, auch der Layovereffekt. Der dynamische Squintwinkel ω_d kann, in Abhängigkeit der y-Position y im Bildraum (mit $0 \le y \le 1$) und dem Spotlightwinkel α_{spot}, berechnet werden aus:

$$\omega_d = 2 \cdot y \cdot \sin \alpha_{spot} - \sin \alpha_{spot} \tag{4.37}$$

Durch Anpassung der Kameramatrix aus der Gleichung (4.7) lässt sich das durch den dynamischen Squintwinkel verursachte Layover in Azimut im Vertex-Shader leicht berechnen:

$$\begin{bmatrix} \vec{x}_x & \vec{y}_x & 0 & 0 \\ \vec{x}_y & \vec{y}_y & 0 & 0 \\ -\tan\theta_{off} & -\tan\theta_{off}\cdot\tan\omega_d & \vec{z}_z & 0 \\ \langle\vec{x},k\rangle & \langle\vec{y},k\rangle & \langle\vec{z},k\rangle & 1 \end{bmatrix} \tag{4.38}$$

In Abbildung 4-21 ist das Ergebnis der Visualisierung des Spotlightmodus abgebildet. Der Schattenwurf und der Layovereffekt ändern sich im Bildverlauf dynamisch.

4.7 Mehrfachreflexionen

Beim Ray-Tracing-Verfahren wird, wie in Kapitel 3.1 beschrieben, der Verlauf der Lichtstrahlen, oder im Fall der SAR-Simulation der Verlauf der vom Radar ausgesendeten Strahlung, verfolgt. Interaktionen, insbesondere Reflexionen, mit oder an Materialoberflächen werden simuliert. Der weitere Verlauf des Strahls wird bis zu seiner Ankunft am Sensor verfolgt. Dieses Verfahren erlaubt damit die Simulation und Visualisierung von Mehrfachreflexionen. Bei der Rasterisierung ist dies nicht möglich, da die Strahlen nicht verfolgt werden. Eine Visualisierung von Mehrfachreflexionen ist bei der Rasterisierung nicht vorgesehen, weswegen Ray-Tracing hierfür das geeignetere Verfahren ist.

Mehrfachreflexionen treten, wie in Kapitel 2.2.3 erläutert, bei SAR-Daten häufig auf und sind ein wichtiger Bestandteil der SAR-Simulation. Wie groß der Unterschied zwischen simulierten Bildern mit und ohne Mehrfachreflexionen ist, hängt vom Bildinhalt ab. Bei natürlichen Oberflächen mit größtenteils Lambertschem Streuverhalten treten nur geringe Unterschiede auf, wohingegen metallische, kantige und eng beieinander stehende Objekte eine Vielzahl von Reflexionen erzeugen. In diesem Fall sind Mehrfachreflexionen für ein realistisches Simulationsergebnis wichtig (vgl. Beispiele in Kapitel 5.3).

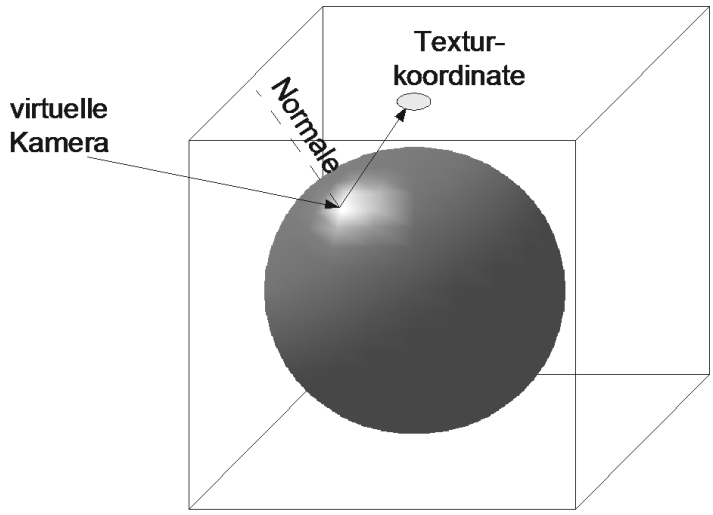

Abbildung 4-22. Schematische Darstellung des „Cube-Mapping"

Die Computergrafik bietet auch für die Rasterisierung eine Möglichkeit, spiegelnde Oberflächen darzustellen. Mittels dem sogenannten „Environmental Mapping“ (Blinn & Newell, 1976) lassen sich Spiegelungen visualisieren. Die verbreitetste Methode ist das „Cubic Environment Mapping“ nach *Greene* (1986). Bei diesem Verfahren wird um jedes Objekt mit spiegelnder Oberfläche eine sogenannte Cube-Map erzeugt. Die Szene wird dann aus Sicht der Position des Objektes sechsmal gerendert, jeweils mit einer anderen Blickrichtung, allerdings ohne das Objekt. Entsprechend der Blickrichtung werden die Seiten des virtuellen Kubus mit den

gerenderten Szenen texturiert. Im abschließenden letzten Renderschritt wird die Szene aus Richtung der virtuellen Kamera gerendert. Die Texturkoordinaten der darzustellenden Textur aus der Cube-Map bestimmen sich aus der Blickrichtung und der Normalenrichtung des abgebildeten Punktes, wie in Abbildung 4-22 illustriert. Anhand dieser Informationen wird die entsprechende Koordinate auf der Cube-Map berechnet und der dort abgespeicherte Texturwert dargestellt. Der Vorteil dieses Verfahrens ist, dass die Cube-Map mittels der vorhandenen Render-Engine erzeugt werden kann (Akenine-Möller & Haines, 2002), also z.B. auch mit der SAR-Geometrie.

Das Verfahren ist vergleichsweise aufwendig und erfordert ein Mehrfaches an Berechnungsaufwand, da die Szene pro erstellter Cube-Map sechsmal gerendert werden muss. Oftmals ist es allerdings nicht nötig, alle sechs Seiten der Cube-Map zu rendern. Um z.B. in einem Computerspiel einen Wandspiegel zu visualisieren, ist es nur nötig, die zum Zimmer gerichtete Seite der Cube-Map zu füllen. Diese wird außerdem in einer stark verringerten Auflösung berechnet, wodurch die Berechnung weiter beschleunigt wird.

Die Technik kann auch bei der SAR-Simulation für die Simulation von Mehrfachreflexionen eingesetzt werden. Mehrere Einschränkungen sind zu beachten. Das Cube-Mapping wird nur zur Visualisierung von Zweifachreflexionen verwendet. Dreifach- oder Vierfachreflexionen von SAR-Daten werden nicht visualisiert. Eine weitere Einschränkung ist die fehlende Visualisierung von Intra-Objektreflexionen. Zwar können die Reflexionen umgebender Objekte visualisiert werden, jedoch nicht Spiegelungen und Reflexionen, die innerhalb eines Objektes auftreten, da die Cube-Map um das Objekt gelegt wird und das Objekt bei der Erstellung der Cube-Map nicht dargestellt wird. Da für eine realistische Visualisierung zwischen dem spiegelnden Objekt und den auf der Cube-Map zu rendernden Objekten ein gewisser Mindestabstand bestehen muss, kann es zu Problemen bei sehr eng aufeinander stehenden Objekten kommen.

Genauso wie moderne Computerspiele realistische Landschaften in Echtzeit visualisieren, können SAR-Szenen realistisch simuliert werden. Die fehlenden Mehrfachreflexionen beeinträchtigen zwar die Realitätstreue der SAR-Visualisierung, jedoch nur in einem für viele Anwendungen akzeptablem Umfang. Mehrfachreflexionen in auf Ray-Tracing basierenden SAR-Simulatoren unterliegen ebenso umfangreichen Einschränkungen, da mit der Zahl der zu berechnenden Reflexionen die Rechenzeit extrem zunimmt. Zweifachreflexionen sind besonders bei der Objektidentifikation bedeutsam. Hier ist insbesondere die „Sensor–Boden–Objekt–Sensor“-Reflexion ein häufig auftretendes Phänomen. Um Rechenzeit zu sparen und dadurch die Echtzeitfähigkeit des Simulators zu bewahren, wird nur diese Zweifachreflexion simuliert. Diese rechenzeitsparende Simplifizierung erlaubt die Simulation einiger für die Visualisierung besonders bedeutsamer Zweifachreflexionen unter Beibehaltung der Echtzeitfähigkeit. Statt sechs Oberflächen der Cube-Textur pro Objekt zu visualisieren, wird nur eine Textur mit der Zweifachreflexion befüllt.

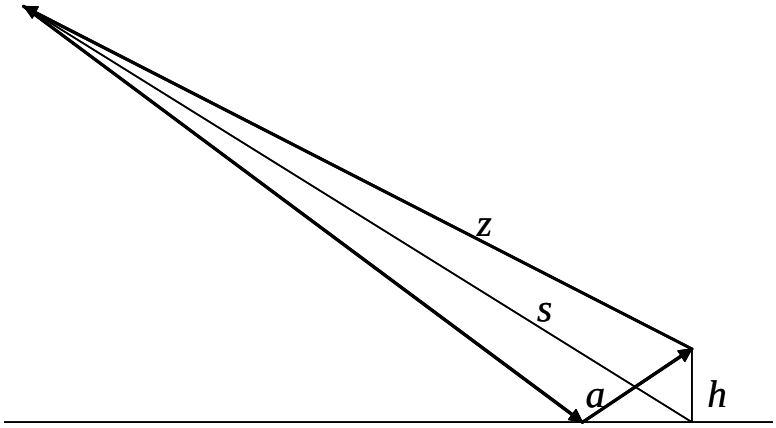

Abbildung 4-23: Schematische und vereinfachte Darstellung der Geometrie der Zweifachreflexionen

Die Simulation der Zweifachreflexion unterliegt einer anderen Abbildungsgeometrie als die der Einfachreflexion. In der Simulation wird die Geometrie der Zweifachreflexion etwas vereinfacht abgebildet. Wie in Abbildung 4-23 dargestellt, setzt sich die Laufzeit des Signals zusammen aus der Laufzeit zwischen Sensor – Boden – Objekt – Sensor.

In der Abbildung der Einfachreflexion entspricht die Laufzeit der Distanz *z* zwischen Sensor und Objekt. In der SAR-Abbildung wird dieser Punkt am Boden abgebildet. Die Linie *a* ist die Linie zwischen dem direkten Reflexionspunkt und dem Punkt gleichen Abstandes auf der Referenzebene. Die im Fall der Zweifachreflexion zurückgelegte Signalstrecke entspricht:

$$2s = 2z + a \tag{4.39}$$

und damit gilt:

$$s = z + \frac{a}{2} \tag{4.40}$$

Daher findet bei der Zweifachreflexion keine Verschiebung in Rangerichtung statt, die Textur der Zweifachreflexionen wird deswegen ohne eine Verschiebung erzeugt. Die Position der virtuellen Kamera und des Sensors wird entsprechend der Reflexionsrichtung angepasst, d.h. der Sensor beleuchtet die Szene von unten. Die spiegelnde Oberfläche wird nicht visualisiert. Die Bereiche der Oberfläche, welche selbst im Schatten liegen, können jedoch keine Strahlung reflektieren. Die Oberfläche wirft, von diesen Bereichen ausgehend, Schatten bei der Berechnung der Zweifachreflexion. Im abschließenden Rendering wird die Textur der Zweifachreflexionen erneut ausgelesen. In Abhängigkeit von den spiegelnden Eigenschaften des Bodens werden die Werte aus dieser Textur zu den Einfachreflexionen addiert.

In Abbildung 4-24 ist ein simuliertes Modell eines LKW abgebildet, wobei die SAR-Effektvisualisierung einmal mit und einmal ohne die Zweifachreflexionen durchgeführt wurde. Das Modell steht auf einem spiegelnden Untergrund, vergleichbar mit einem Asphaltbelag. Die Zweifachreflexionen sind daher vergleichsweise stark, erkennbar an Hand der starken Reflexionen im Bodenbereich des linken LKW.

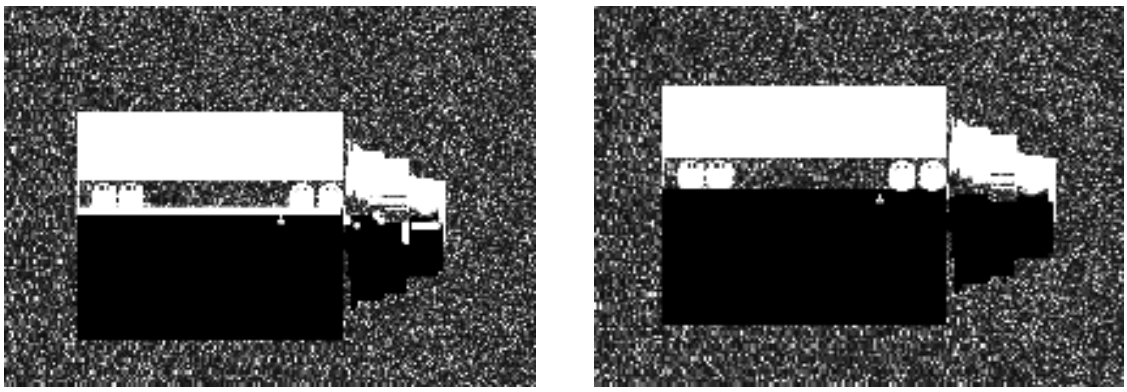

Abbildung 4-24: SAR-Simulation eines LKW-Modells mit (links) und ohne (rechts) Zweifachreflexion

Ein weiteres Problem bei der Visualisierung sind die sogenannten Cornerreflektoren. Diese sehr starken Reflexionen werden bei der vorgestellten Methode nicht stark genug visualisiert. Eine mögliche Lösung wäre die Vorverarbeitung der Modelle. Durch die Vorberechnung von Ecken in den Modellen könnten diese bei der Visualisierung gesondert behandelt und somit besser dargestellt werden.

Letztendlich bietet die Simulation der Zweifachreflexion mittels Cube-Mapping nur eine geringe Verbesserung. Mehrfachreflexionen fehlen ebenso wie Intra-Objektreflexionen. Die prinzipielle Möglichkeit, Reflexionen zu simulieren, darf nicht über diese Einschränkungen hinwegtäuschen. Je nach Anwendung und der Bedeutung der Mehrfachreflexionen für die Anwendung, ist der Einsatz eines Ray-Tracing-Simulators mit voller Unterstützung der Mehrfachreflexionen nötig. Dabei könnten für die Echtzeitanwendungen sowohl Softwaretechniken wie Ray-Tracing mit GPU-Unterstützung (Purcell et al, 2002; Purcell, 2004) als auch hardwarebeschleunigtes Ray-Tracing (Schmittler et al, 2002) zum Einsatz kommen.

4.8 Zusammenfassung des Ablaufs

Der Programmablauf der Visualisierung kann unterteilt werden in Verwaltungsfunktionen und die eigentliche Visualisierung. Unter Verwaltungsfunktionen werden alle Programmteile verstanden, die der Verwal-

tung von Benutzereingaben dienen und welche der eigentlichen Simulation die Daten zur Verfügung stellen. Darüber hinaus fallen der Datenimport und der Datenexport unter die Verwaltungsfunktionen. Die Darstellung der Daten auf dem Bildschirm bzw. die Erzeugung von Texturdaten für den späteren Datenexport sind Funktionen der Visualisierung. Auf die Verwaltungsfunktionen wird im Folgenden nicht näher eingegangen, sie bilden allerdings das nötige Grundgerüst der Visualisierung.

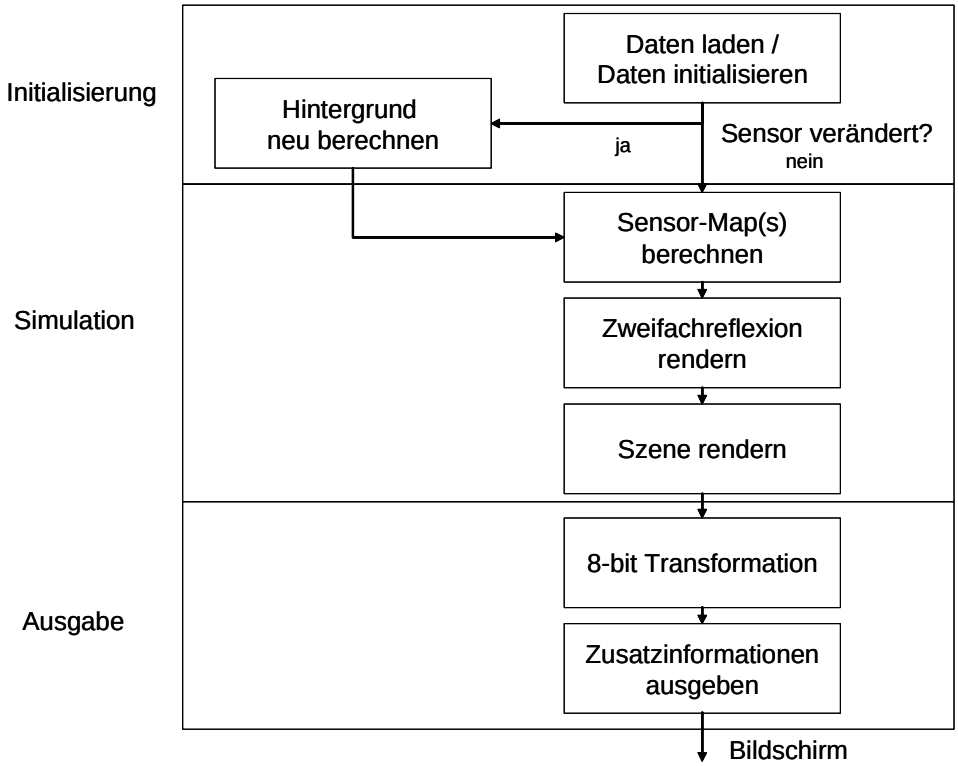

Abbildung 4-25: Schematische Darstellung des Simulationsablauf

In Abbildung 4-25 ist der gesamte Ablauf der Simulation schematisch dargestellt. Zunächst werden die zu visualisierenden Daten geladen und initialisiert. Die Initialisierung umfasst auch Schritte zur Vorbereitung der Hardware und der Ausgabe. Dann wird überprüft, ob sich die Sensoreinstellungen verändert haben. Die Sensoreinstellungen umfassen in diesem Fall, neben den direkt den Sensor betreffenden Daten, wie z.B. den Blickwinkel oder die räumliche Auflösung, ebenso Informationen über den anzuzeigenden Bildausschnitt. Ändern sich diese Daten, so müssen evtl. angezeigte Bildhintergründe neu berechnet bzw. neu ausgeschnitten werden. In Abhängigkeit der Einstellungen wird die für das Speckling benötigte mit Zufallszahlen gefüllte Textur neu berechnet oder die Startwerte der Zufallszahlen neu gesetzt.

Dann wird die Sensor Map berechnet, welche sowohl für die Berechnung der Textur der Zweifachreflexionen als auch für die Sichtbarkeitsanalyse während der Visualisierung der SAR-Szene benötigt wird. Danach wird die Textur der Zweifachreflexionen berechnet, sofern die Simulation von Zweifachreflexionen gewünscht wird. Das folgende Rendering der SAR-Szene wird in Abbildung 4-26 schematisch dargestellt und weiter unten ausführlich beschrieben. Die Szene wird in eine 32-bit Textur gerendert. Diese Textur lässt sich nicht auf dem Bildschirm darstellen, so dass eine Anpassung der Textur auf die 8-bit Bildschirmausgabe erfolgt. Die 32-bit Werte der Simulation werden entsprechend den Anforderungen des Benutzers linear oder logarithmisch transformiert. Eventuell darzustellende Hintergrundbilder werden eingeblendet. Im letzten Schritt können dann Zusatzinformationen, wie z.B. die aktuelle Mausposition, auf dem Bildschirm ausgegeben werden.

Der Kern der Visualisierung ist das Rendern der Szene. In Abbildung 4-26 ist diese Prozedur schematisch dargestellt. Die vorhergehende Erzeugung der Textur der Zweifachreflexionen ist vergleichbar, es fallen allerdings Schritte wie z.B. das thermale Rauschen weg und die verwendeten Transformationen sind entsprechend an die Zweifachreflexion angepasst.

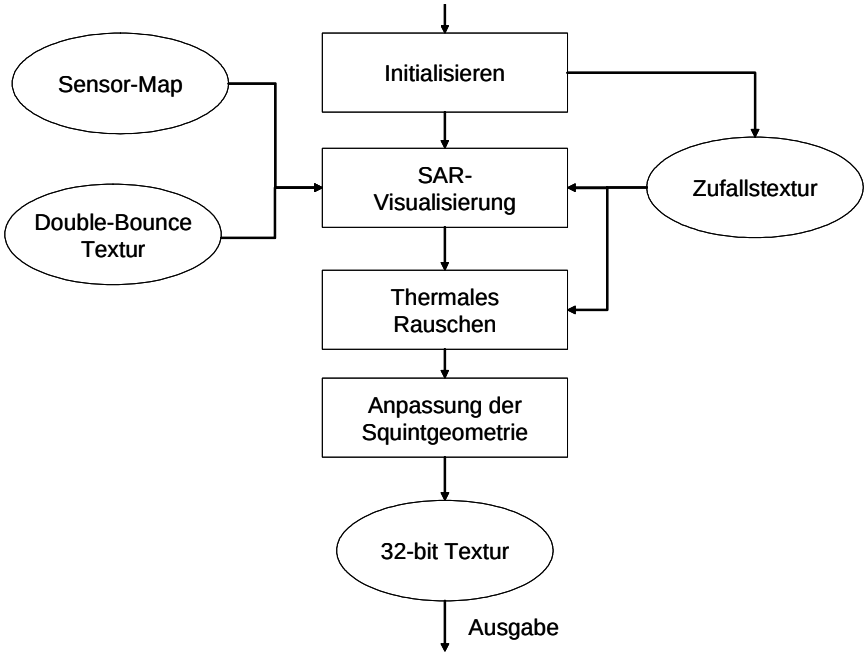

Abbildung 4-26: Schematische Darstellung der Prozedur: „Szene rendern“

Nach der Initialisierung der benötigten Texturen und Transformationen wird die Szene gerendert. Danach wird zur Ausgabe ein thermales Rauschen addiert, um ein additives Rauschen zu erzeugen. Dadurch werden absolute Nullwerte in Schattenbereichen vermieden. Schließlich werden die Drehungen des Squintwinkels wieder zurückgedreht.

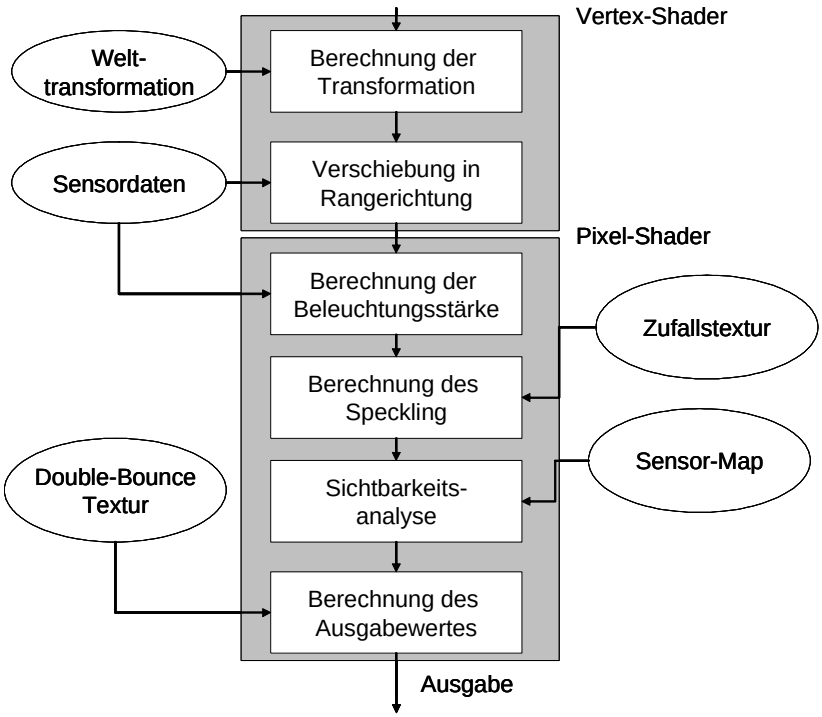

Abbildung 4-27: Schematische Darstellung der SAR-Simulation auf der GPU

In Abbildung 4-27 ist der Ablauf der Simulation auf der GPU schematisch dargestellt. Im Vertex-Shader findet die Transformation der Geometrie von Weltkoordinaten in Bildschirm- oder Bildkoordinaten statt. Dabei wird die laufzeitgeometrisch bedingte Verschiebung der Koordinaten berechnet. Im Pixel-Shader wird die Radiometrie berechnet. Abhängig von der Normalenrichtung, den Materialeigenschaften und den Sensorparametern wird die Reflexion pro Pixel berechnet. Danach wird der Specklewert berechnet und schließlich

die Sichtbarkeitsanalyse anhand der Sensor Map durchgeführt. Falls der Bildpunkt sichtbar ist, wird der berechnete Wert ausgegeben. Nicht sichtbare Werte werden transparent ausgegeben. Falls vorhanden, werden die Werte der Zweifachreflexion an der entsprechenden Stelle hinzuaddiert.

Um die Nebenkeulen zu visualisieren, muss das in den vorherigen Schritten berechnete Ergebnis untersucht werden. Reflexionsstärken, welche einen Schwellwert überschreiten, werden als mögliche Kandidaten für Überstrahlungen behandelt und in einer Ergebnistextur gespeichert. Die Stärke der Reflektion und ihre Position im Bild werden unter Ausnutzung von drei Farbkanälen gespeichert. Dabei wird in einem Farbkanal die Stärke gespeichert, die anderen beiden enthalten die Position der Überstrahler in Texturkoordinaten. Dieses Ergebnisbild wird dann reduziert (siehe auch Kapitel 6.4), so dass nur die stärksten Überstrahlungen weiter bearbeitet werden. Für diese Überstrahler wird dann eine in Abhängigkeit der Reflexionsstärke passende Textur berechnet. Diese Texturen werden schließlich auf das Ergebnisbild addiert, so dass die durch die Nebenkeulen entstehenden Effekte in Echtzeit visualisiert werden können.

5 Beispiele und Anwendungen der SAR-Effektvisualisierung

Im Folgenden wird die unter Kapitel 4 vorgestellte Methode mit Beispielen illustriert und die Ergebnisse der vorgestellten Methode der SAR-Effektvisualisierung werden mit den Ergebnissen anderer SAR-Simulatoren verglichen. Dabei werden die Einflüsse der verwendeten Modelle auf die Simulation untersucht und die Fähigkeit des implementierten Ansatzes hochauflösende SAR-Daten zu visualisieren wird demonstriert.

5.1 Beispiele der Echtzeitvisualisierung von SAR-Effekten

Die Visualisierung von SAR-Daten mittels Rasterisierung unterliegt einigen Einschränkungen. Trotzdem erlaubt diese Methode die realistische SAR-Visualisierung in Echtzeit, wie im Folgenden anhand einiger Bildbeispiele dargestellt wird.

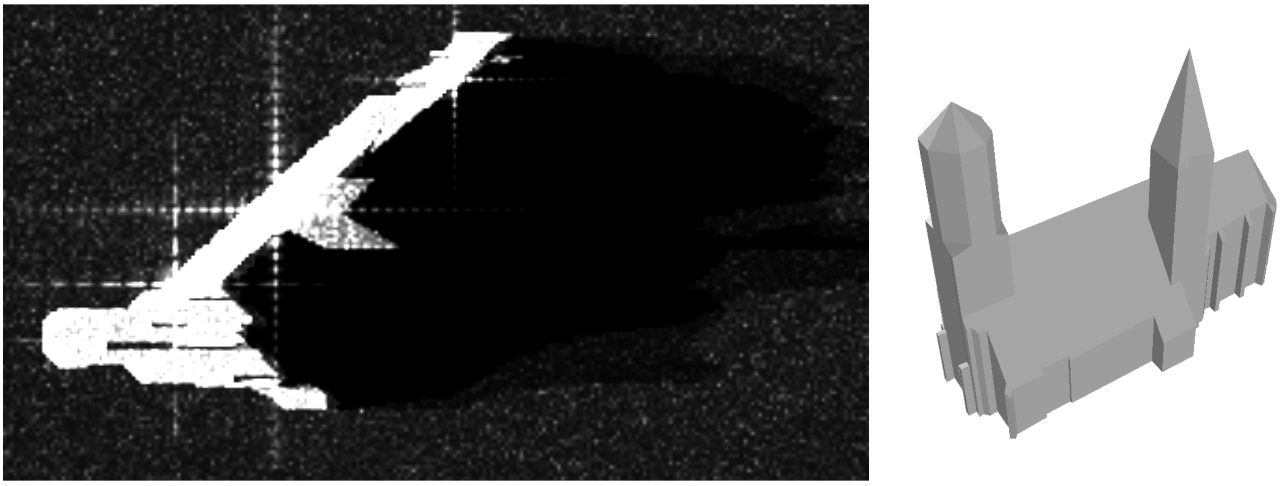

Abbildung 5-1: SAR-Visualisierung (links) eines Modells (rechts) der „Stiftskirche“ in Stuttgart

Abbildung 5-1 zeigt die Visualisierung eines einfachen Gebäudemodells. Dieses vergleichsweise einfache Modell stellt keine großen Anforderungen an die Leistungsfähigkeit der Visualisierung. Die Visualisierung erfolgt mit einer Pixelauflösung von 30 cm und einem Off-Nadirwinkel des Sensors von 55°.

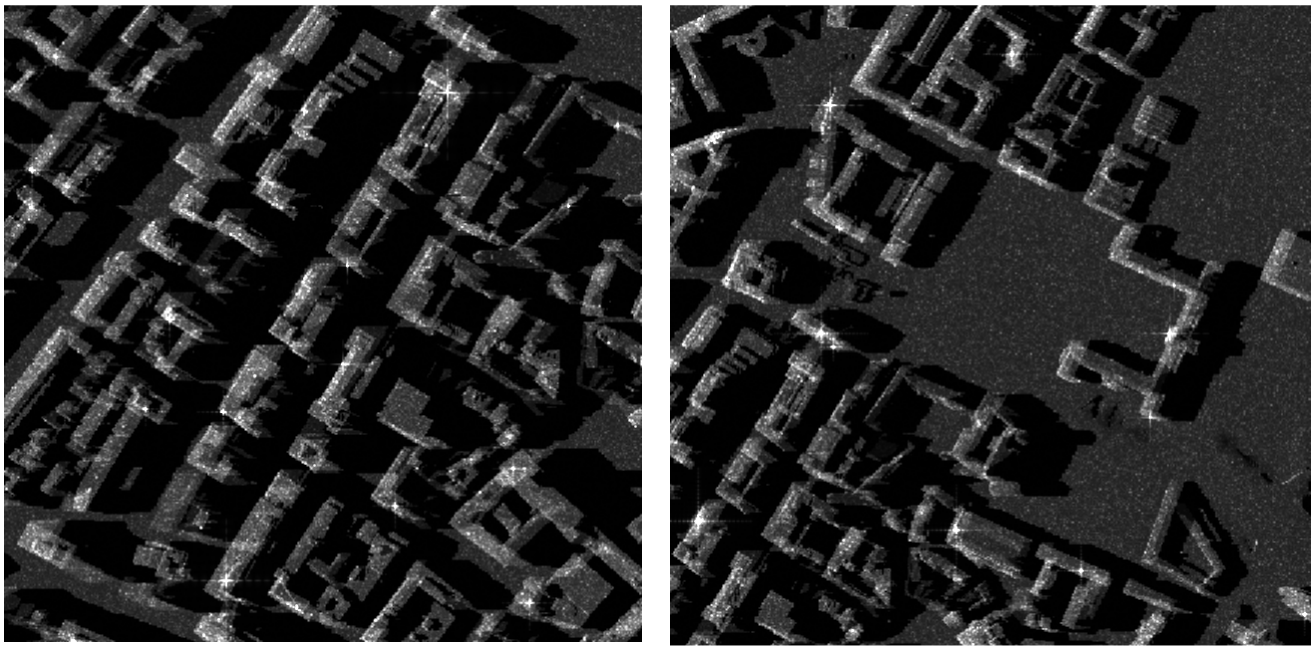

Abbildung 5-2: SAR-Visualisierung des Stadtmodells von Stuttgart mit hoher Pixelauflösung

Bei der Visualisierung wird das Gebäude als Lambertscher Streuer dargestellt. Überstrahlungen werden visualisiert, nicht jedoch die Zweifachreflexionen. Die Visualisierung komplexer Stadtmodelle ist für eine Reihe von Anwendungen bedeutend. Gerade bei der Missionsplanung im urbanen Raum ist eine Echtzeitvisualisierung eine große Hilfe und erlaubt es, eine Reihe von unterschiedlichen Flug- und Sensorparametern in kürzester Zeit zu testen, um die für den jeweiligen Auftrag passende Konfiguration zu ermitteln.

In Abbildung 5-2 sind zwei hochauflösende SAR-Visualisierungen des 3D-Stadtmodells von Stuttgart abgebildet. Das Modell wurde vom Stadtmessungsamt Stuttgart erzeugt. Es enthält nur die Geometrien und keine Informationen über die Materialeigenschaften. Dies ist bei den meisten verfügbaren Datensätzen der Fall, so dass diese Informationen für die meisten Simulationen nicht zur Verfügung stehen. In Abbildung 5-2 sind zwei Ausschnitte aus dem Stadtzentrum von Stuttgart zu sehen, in Abbildung 5-3 ist das ganze Modell abgebildet. Zusätzlich zum Stadtmodell, welches mehr als 32.000 Gebäude umfasst, wird ein Digitales Geländemodell des Talkessels von Stuttgart visualisiert (siehe Abbildung 5-4). Insgesamt besteht die Szene aus 1.273.499 Dreiecken und wird von einer „NVIDIA GeForce 7900 GTX“ Grafikkarte in ca. 50 ms visualisiert. Diese Zeitangabe enthält weder die Zeit die benötigt wird, um die Modelle zu laden, noch die Zeit, die benötigt würde die Daten vom Videospeicher in den Hauptspeicher zu kopieren, um dort die Bilder weiter zu verarbeiten oder auf der Festplatte abzuspeichern.

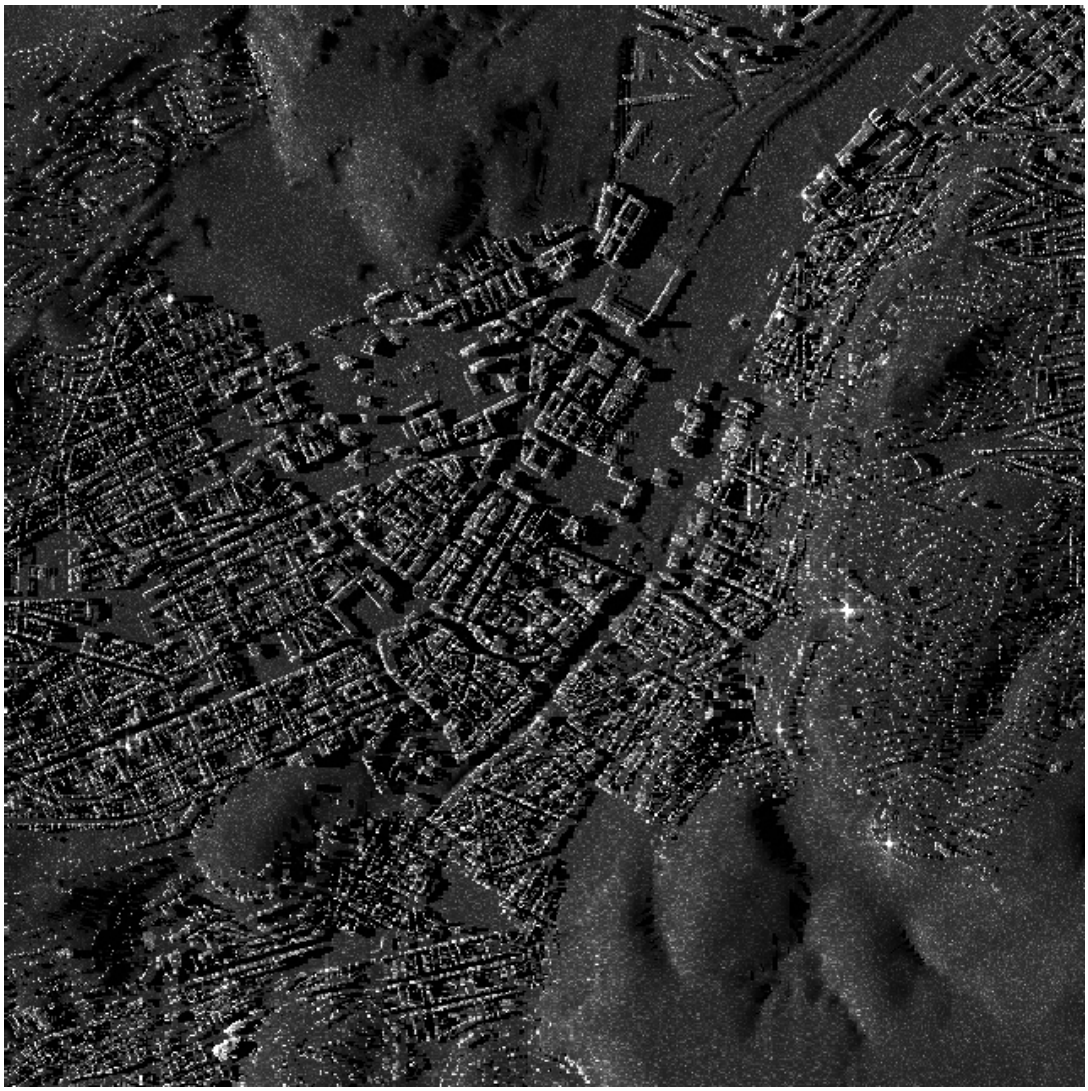

Abbildung 5-3: SAR-Visualisierung des Stadtmodells Stuttgart inklusive DTM mit 7m Pixelauflösung

Schnellere Ergebnisse lassen sich, insbesondere in niedrigeren Auflösungen, durch Vereinfachungen der Geometrien und der dadurch erreichten Verringerung der Anzahl abzubildender Dreiecke, erreichen. Speziell zur Generalisierung von 3D-Gebäudemodellen gibt es verschiedene Verfahren, wie z.B. Verfahren basierend auf der Erweiterung der Generalisierung mit Maßstabsräumen (Forberg & Mayer, 2002; Forberg, 2004),

Verfahren mittels Zerlegung von Gebäudemodellen in 3D-Primitive (Thiemann, 2002) oder Verfahren basierend auf der Halbraummodellierung (Kada, 2006). Diese oder andere Formen der Optimierung sind zukünftig implementierbar, womit die Visualisierung weiter beschleunigt werden könnte.

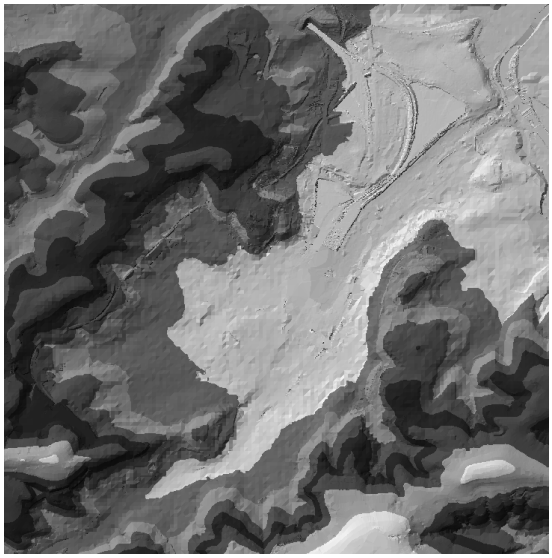

Abbildung 5-4: Digitales Geländemodell des Talkessels von Stuttgart

Sämtliche unter Abbildung 5-1 bis Abbildung 5-3 visualisierten Objekte wurden mit Lambertschen Reflexionseigenschaften simuliert. Für realistische Simulationsergebnisse ist es sehr bedeutsam, dass unterschiedliche Objekte unterschiedliche Reflexionseigenschaften aufweisen können. Bei der Visualisierung von Stadtmodellen sind die Unterschiede im Reflexionsverhalten der einzelnen Gebäude jedoch im Normalfall nicht bekannt und obwohl deren Simulation möglich wäre, scheitert dies in der Praxis häufig an fehlenden Informationen.

Die Reflexionseigenschaft metallischer Objekte unterscheidet sich stark von den Eigenschaften der Lambertschen Reflexionen. Metallische Reflexionen spiegeln stärker und sind insbesondere an Kanten, welche die Energie zum Sensor zurückreflektieren, besonders stark. In Abbildung 5-5 ist dies an den Modellkanten deutlich zu erkennen.

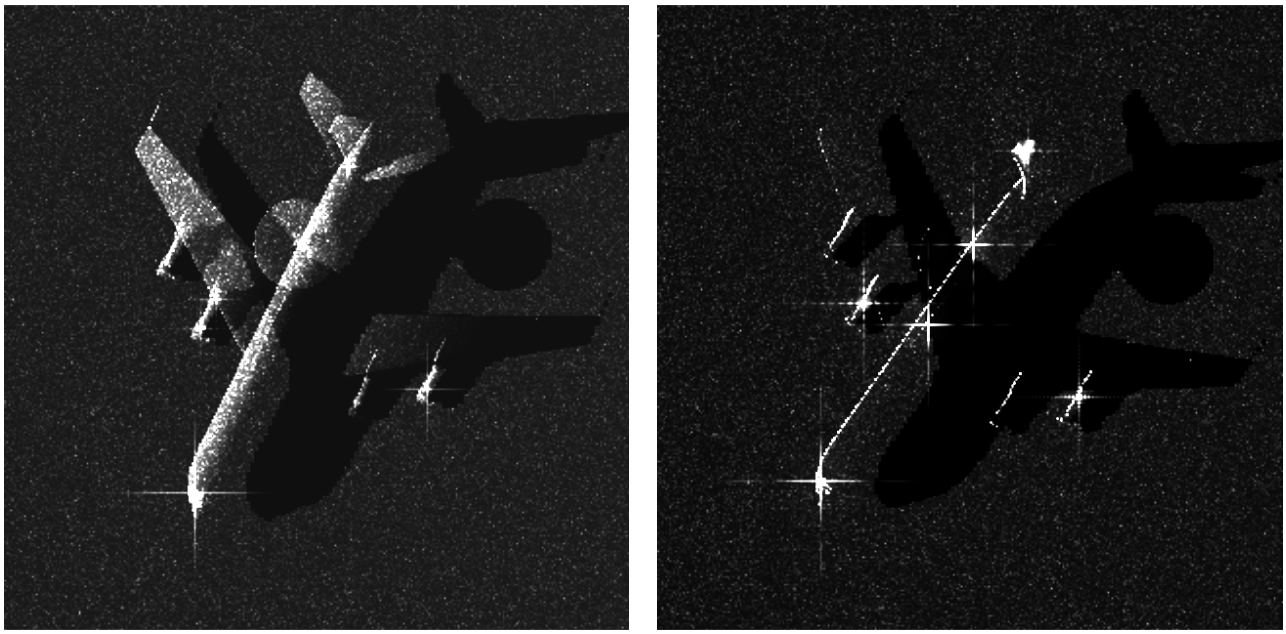

Abbildung 5-5: SAR-Visualisierung eines Modells der E3-Sentry (Awacs) mit Lambertschen Reflexionseigenschaften (links) und metallischen Reflexionseigenschaften (rechts)

Die Auswirkungen von Eigenbewegungen dargestellter Objekte in realen und simulierten SAR-Bildern sind in Abbildung 5-6 zu sehen. Die Verwischung in Azimut ist deutlich erkennbar. Bei dem Flugzeug im linken Teil von Abbildung 5-6 handelt es sich vermutlich um eine rollende Maschine unbekannten Typs nach ihrer Landung (siehe Kapitel 2.2.2 und 4.3.4).

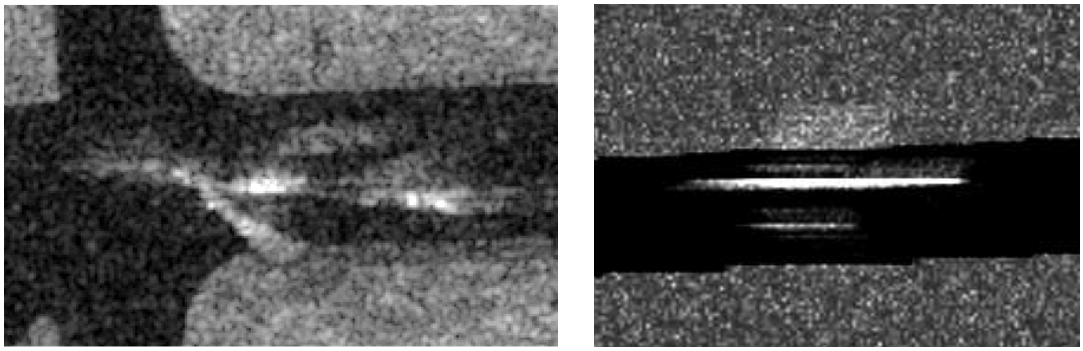

Abbildung 5-6: DOSAR-Bild (links) (Quelle: EADS Deutschland) und SAR-Simulation (rechts) eines sich bewegenden Flugzeuges

5.2 Einfluss der Modelle auf die Visualisierung

Die Qualität und Detailtreue der verfügbaren Modelle beeinflusst den Realismus der Visualisierung entscheidend. Allerdings kann es sehr aufwendig und kostenintensiv sein, solche Modelle zu erstellen. Auch lässt sich, selbst durch sehr detaillierte Modelle, der Realitätsgrad einer Simulation nicht beliebig steigern. Kein Modell entspricht der Wirklichkeit und neben Fehlern oder Ungenauigkeiten in den Modellen gibt es noch eine Reihe weiterer Fehlerquellen bei Simulationen. Neben den Modellen an sich ist die Repräsentation der Modelle im Speicher und bei ihrer Prozessierung mitentscheidend für eine realistische Visualisierung.

5.2.1 Einfluss der Modellrepräsentation auf die Visualisierung

In Abbildung 5-7 ist sowohl links als auch rechts dasselbe Modell einer E3-Sentry abgebildet. Rechts sind die Normalenrichtungen für jeden Punkt im Dreieck gleich. Im linken Beispiel setzt sich die Normalenrichtung eines Vertex aus den Normalen der angrenzenden Dreiecke zusammen. Somit sind die Normalenrichtungen Mittelwerte über die an den Punkt angrenzenden Dreiecke. Dies ist insbesondere für die Visualisierung von Kurven und gekrümmten Oberflächen von Vorteil. Bei der Berechnung der Beleuchtungsintensität auf dem Pixel-Shader werden die Normalenrichtungen über das Dreieck interpoliert und pro Pixel die Beleuchtungsintensität in Abhängigkeit der interpolierten Normalenrichtung berechnet. Dies entspricht dem aus der Computergrafik bekannten Phong-Shading. Dadurch wirken die Flächen gekrümmt und insbesondere die sehr winkelabhängigen spiegelnden Reflexionen realistischer.

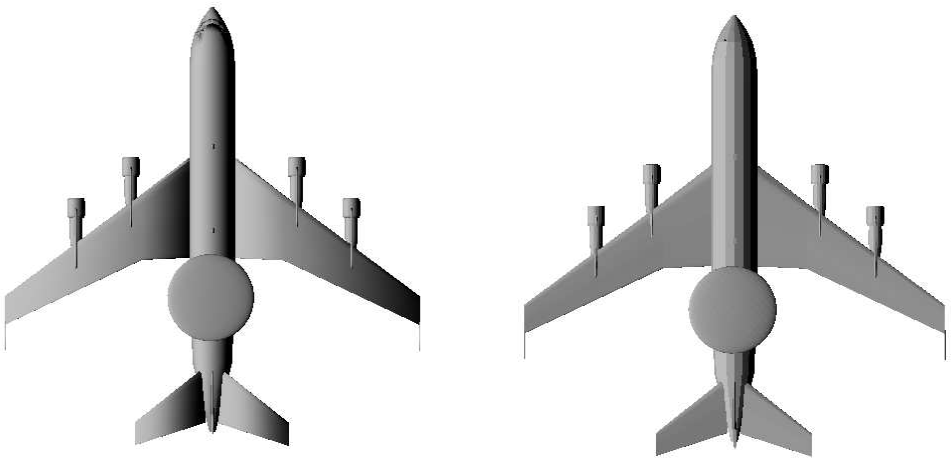

Abbildung 5-7: Modelldarstellung mit gerundeter Oberfläche mittels Phong-Shading (links) und kantiger Oberfläche (rechts)

In Abbildung 5-8 ist die Auswirkung der unterschiedlichen Modellrepräsentationen auf die SAR-Visualisierung zu sehen. Bei gekrümmten Flächen kommt es im linken Beispiel zu einer sehr starken Reflexion, da es auf Grund der Interpolation der Normalenrichtungen über die Krümmung an mindestens einer Stelle zu einer direkten spiegelnden Reflexion kommt, wohingegen dies bei der eher kantigen Repräsentation nur zufällig, dann aber für die ganze Fläche, vorkommt.

Die Interpolation der Normalenrichtung ist besonders geeignet zur Darstellung von gekrümmten Flächen. Für die Darstellung von Kanten ist dieses Verhalten nicht optimal. Eine Kombination der beiden Repräsentationen ist möglich, so dass sowohl die Kanten als auch die gekrümmten Oberflächen realistisch abgebildet werden.

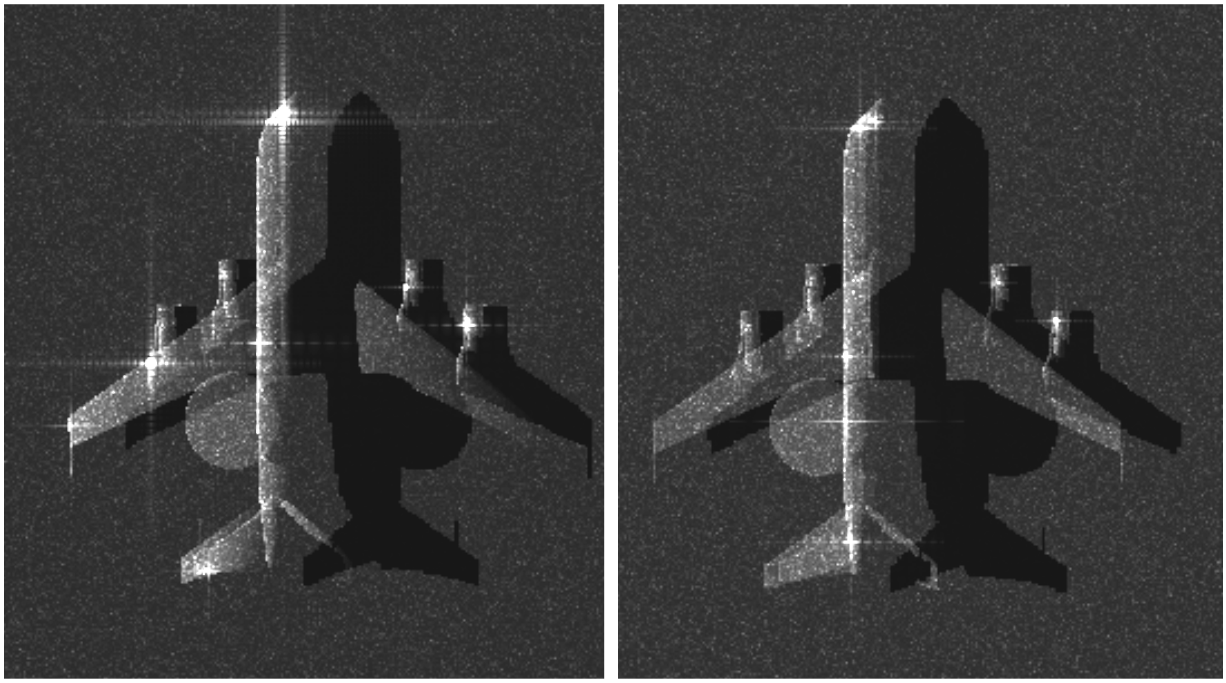

Abbildung 5-8: SAR-Visualisierung der Modelle aus Abbildung 5-7 mit stark Lambertschem Streuverhalten

5.2.2 Probleme der Subpixelgenauigkeit bei der SAR-Visualisierung

Bei der Modellvisualisierung kann es zu Maßstabsproblemen kommen. Bei sehr kleinen Maßstäben und sehr kleinen Modellen werden einzelne Bildprimitive, Objektteile oder ganze Modelle nicht mehr dargestellt. Sollte das Modell oder Teile des Modells im Bildraum kleiner als ein Pixel sein, dann wird dies überhaupt nicht dargestellt. Doch selbst wenn Modellteile noch dargestellt werden, kann es bei kleinen Maßstäben zu Problemen bei der Visualisierung von gekrümmten Flächen kommen. Streckt sich die Krümmung über nur wenige Pixel, fehlt durch die Diskretisierung, trotz Interpolation der Normalen über die Krümmung hinweg, unter Umständen die direkte Reflexion. Im Fall einer spiegelnden Fläche wäre die Gesamtreflexion dann zu niedrig, da die sehr starke direkte Reflexion nicht berechnet werden würde. Die Vermeidung der Darstellung von Modellen oder von Modellteilen die kleiner als ein Pixel sind, bietet Geschwindigkeitsvorteile bei der Visualisierung in Computerspielen. Da sich bei der SAR-Visualisierung einzelne Reflexionen addieren und bereits kleine Flächen, die direkt zum Sensor zurückreflektieren einen großen Teil zur Gesamtreflexion beitragen, führen die Diskretisierungsfehler zu fehlerhaften Simulationsergebnissen.

Eine mögliche Lösung wäre es, sämtliche Szenen in einer fixen Auflösung zu berechnen, beispielsweise in halber Wellenlänge des verwendeten Radarbandes. Für eine Beispielszene von einem Quadratkilometer entspräche dies, bei einer angenommenen Wellenlänge von 5cm im C-Band, 20.000 mal 20.000 Bildpixel. Ausgehend von einer Bildgröße von 1.000 x 1.000 Pixel pro Renderschritt, müssten 400 Bilder gerendert und zur endgültigen Szene zusammengefasst werden. Da der Bildinhalt pro Renderschritt nicht sehr komplex sein dürfte, kann man von einer Berechnungszeit von 50 ms pro Renderschritt ausgehen, so dass es ca. 20 Sekunden dauert, bis sämtliche Teilbilder berechnet sind und voraussichtlich weitere 10 bis 20 Sekunden, bis die Teilbilder zu einem geringer aufgelösten Gesamtbild zusammengefasst werden. Somit wäre eine von Maß-

stabseffekten unabhängige Berechnung zwar nicht mehr echtzeitfähig, jedoch mit einer Berechnungszeit von unter einer Minute weiterhin sehr schnell.

5.2.3 Modellierung komplexer Szeneninhalte

Die Darstellung komplexer Umgebungen, wie z.B. Waldgebiete, kann auf zwei unterschiedliche Arten erfolgen. Die komplexen Umgebungen können entweder durch detaillierte Modelle oder durch Texturen repräsentiert werden. Die Repräsentation der Umgebung durch detaillierte Modelle ist die genauere Methode. Dabei wird z.B. ein Wald durch eine Vielzahl an Baummodellen dargestellt. Jedes Baummodell ist im Idealfall einzigartig. Zur Darstellung der Äste und der Blätter besteht jedes Baummodell aus Zehntausenden von Dreiecken, so dass zur detaillierten Darstellung eines Waldes, Millionen von Dreiecken dargestellt werden müssen.

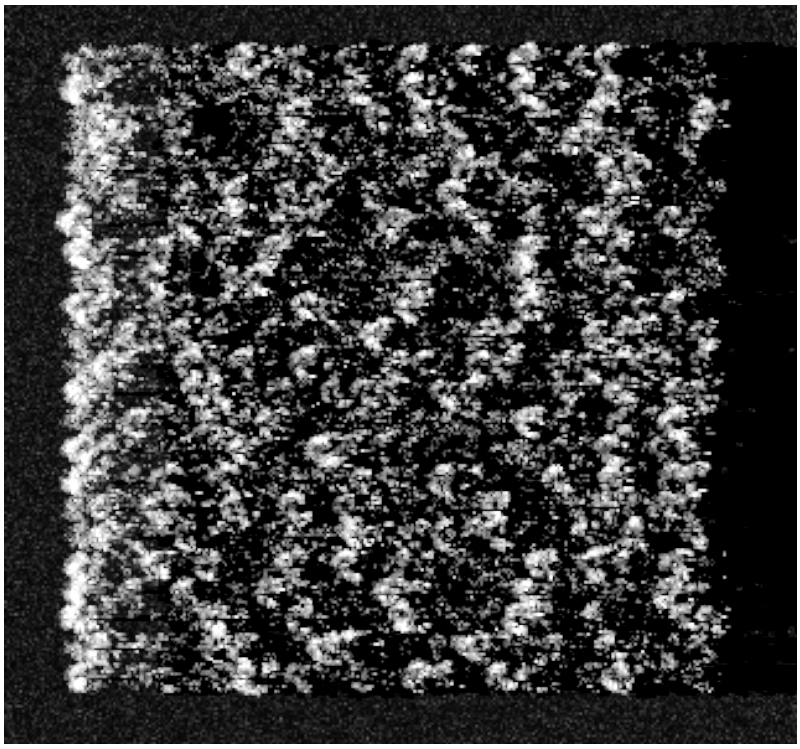

Abbildung 5-9: Simulierter Wald bestehend aus 536 individuellen Bäumen mit insgesamt 17.666.955 Dreiecken

In Abbildung 5-9 ist ein so simulierter Waldbereich dargestellt. Das Waldmodell besteht aus einer Vielzahl von Modellen einzelner Bäume, die unterschiedlich hoch sind und in ungleichmäßigen Abständen zueinander platziert wurden. Das dargestellte Modell eines Waldes besteht aus mehr als 17 Millionen Dreiecken. Die Darstellung einer solch hohen Zahl von Dreiecken verlangsamt die Visualisierung. Die Visualisierung eines größeren und komplexeren Waldgebietes würde selbst die Fähigkeiten moderner Grafikkarten überschreiten, so dass hierfür andere Methoden angewendet werden sollten. So könnte ein Ausschnitt des Waldes als Impostor verwendet werden. Impostors sind Billboards (siehe Kapitel 4.4.5) die während der Programmausführung erzeugt werden (Forsyth, 2001). In diesem Falle würde ein Teil des Waldes gerendert und die erzeugte Textur als Billboard für den restlichen Wald verwendet werden.

In einem Waldgebiet sind nicht die unterschiedlichen Reflexionsparameter ausschlaggebend für die im SAR-Bild erscheinende Textur, sondern die starken Höhendifferenzen und die dadurch bedingten Unterschiede in den Normalenrichtungen. Eine vergleichsweise einfache Möglichkeit, dies zu visualisieren, ist die Erzeugung einer Normalentextur. Diese sogenannte Bump-Map (Blinn, 1978) beinhaltet keine Reflexionsparameter, sondern pro Pixel eine Normalenrichtung. Diese Normalenrichtung ersetzt bei der Berechnung der Reflexionsstärke die Normalenrichtung des Dreiecks und ermöglicht so die einfache Repräsentation von geometrisch sehr komplexen Szeneninhalten wie z.B. Waldflächen.

5.3 Visueller Vergleich mit den Ergebnissen anderer SAR-Simulatoren

Um die Leistungsfähigkeit der SAR-Effektvisualisierung in Echtzeit einschätzen zu können, ist es wichtig, die Resultate mit anderen SAR-Simulatoren zu vergleichen. Daher werden die Simulationsergebnisse von zwei unterschiedlichen Bildsimulatoren mit den Ergebnissen der in dieser Arbeit vorgestellten SAR-Effektvisualisierung (SARViz) verglichen, um sowohl die Leistungsfähigkeit, als auch die Schwierigkeiten und Probleme des gewählten Ansatzes aufzuzeigen. „SE-Workbench-EM“[1] ist ein kommerzieller Simulator der Firma „Oktal Synthetic-Environment“ (www.oktal-se.fr). Für die SAR-Simulationen mit SE-Workbench-EM wird das SE-RAY-EM-Modul verwendet. Der Ray-Tracing Simulator des FGAN-FOM[2] befindet sich noch in der Entwicklung, so dass die hier erstmals präsentierten Ergebnisse als vorläufig zu bewerten sind. Beide Simulatoren sind in der Lage, Mehrfachreflexionen auch zwischen unterschiedlichen Objekten in einer Szene darzustellen.

5.3.1 Simulation von Gebäudemodellen

Abbildung 5-10: Das Gebäude des FGAN-Forschungsinstitut für Optronik und Mustererkennung FGAN-FOM (links) und das für die Simulationen verwendete 3D-Gebäudemodell (rechts)

Die Simulation einzelner Gebäude ist eine vergleichsweise einfache Aufgabe. Um die Simulationsergebnisse vergleichen zu können, befindet sich der simulierte Sensor in einer Bodenentfernung von 3000 Meter zum Ziel und in einer Flughöhe von 3000 Metern. Zunächst wird das Modell aus Abbildung 5-10 aus zwei verschiedenen Blickrichtungen simuliert mit einer Auflösung von 40 cm simuliert. Bei der Simulation wird von einer einheitlichen Materialoberfläche für das Modell ausgegangen. Die unterschiedlichen Reflexionseigenschaften von Dachziegeln, Fenster und Türen werden nicht berücksichtigt. Vielmehr besteht das Gebäude für die Simulatoren komplett aus Stein und es befindet sich auf einer Grasfläche.

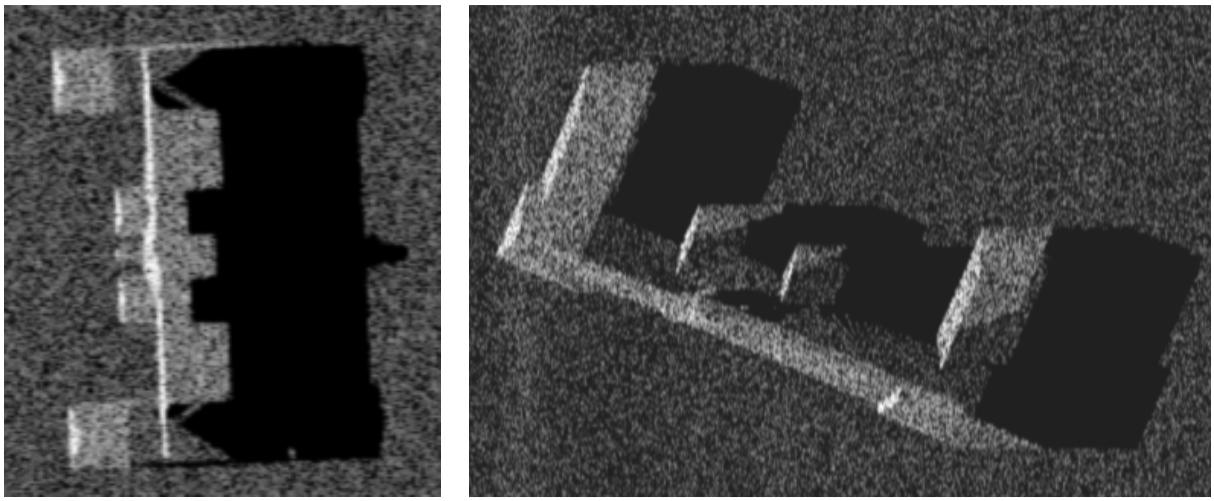

Abbildung 5-11: SAR-Simulation des Gebäudemodells aus Abbildung 5-10 mit SE-RAY-EM

[1] Die SE-Workbench-EM sowie einige der unten gezeigten 3D-Modelle wurden freundlicherweise vom FGAN-Forschungsinstitut für Optronik und Mustererkennung (FGAN-FOM) zur Verfügung gestellt.
[2] Der von Dr. Hammer entwickelte SAR-Simulator befindet sich noch in einem sehr frühen Entwicklungsstadium, so dass es noch keine zitierfähige Veröffentlichung gibt.

Die mit SE-RAY-EM durchgeführte Simulation in Abbildung 5-11 dient als Referenz für die weiteren Simulationen. Die Simulationen mit SE-RAY-EM und dem FOM-Simulator wirken, auf Grund der bei beiden vorgenommen Punkfaltung, realistischer als die SAR-Effektvisualisierung, die aus Performancegründen darauf verzichtet. Auffällig bei der Frontalsicht auf das Gebäude ist, dass nur in Abbildung 5-12 die Doppelreflexion an der Kante zwischen dem Turm und der Dachfläche deutlich zu erkennen ist. Bei der Visualisierung mittels Grafikhardware in Abbildung 5-13 tritt dieser Effekt nicht auf, da die Abbildung nur mit einer Reflexion visualisiert wurde. Obwohl bei der Simulation mit SE-RAY-EM acht Reflexionen simuliert wurden, tritt diese Doppelreflexion in Abbildung 5-11 ebenfalls nicht auf. Dies liegt an dem bei der Modellierung verwendeten Material, welches nicht spiegelt. Wird das Gebäude mit spiegelnden oder teilweise spiegelnden Materialen simuliert, kommt es zu diesen Mehrfachreflexionen. Aus Gründen der Vergleichbarkeit wurde jedoch auf eine umfangreiche Modellierung der Materialeigenschaften verzichtet.

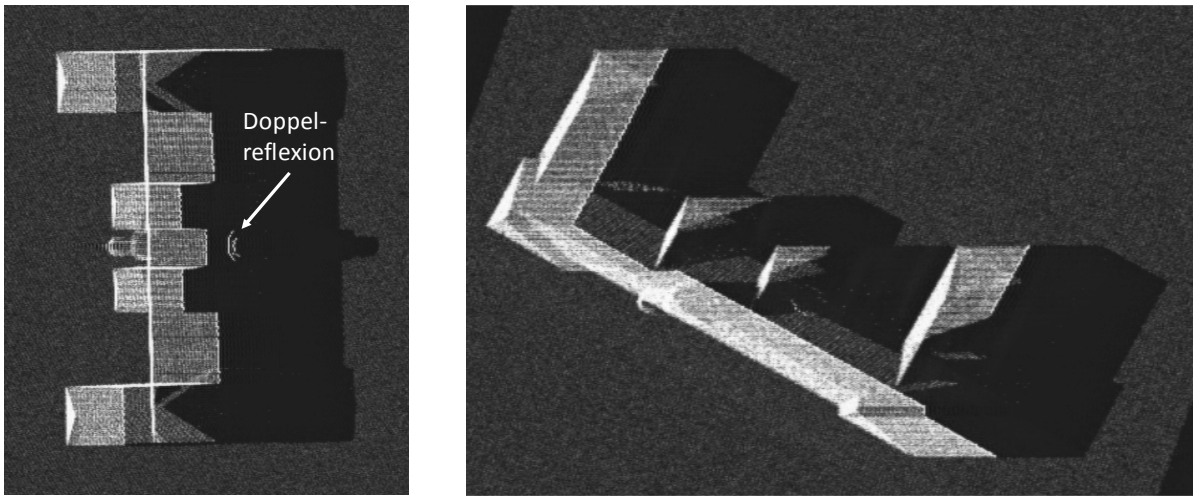

Abbildung 5-12: SAR-Simulation des Gebäudemodells mit dem Simulator von FGAN-FOM

Die größten Unterschiede entstehen bei der Simulation von Schatten. Die Ungenauigkeiten der Shadow-Map-Methode (siehe Kapitel 4.4.2.2) führt in Abbildung 5-13 dazu, dass einige schräge Dachflächen im Schatten liegen, obwohl dies in Abbildung 5-11 nicht der Fall ist. Außerdem unterscheidet sich der Kontrast aller Simulationsergebnisse, was nicht auf die Simulationseigenschaften zurückzuführen ist, sondern einzig auf Unterschiede bei der Wandlung der simulierten SAR-Daten in darstellbare 8-bit Bilder.

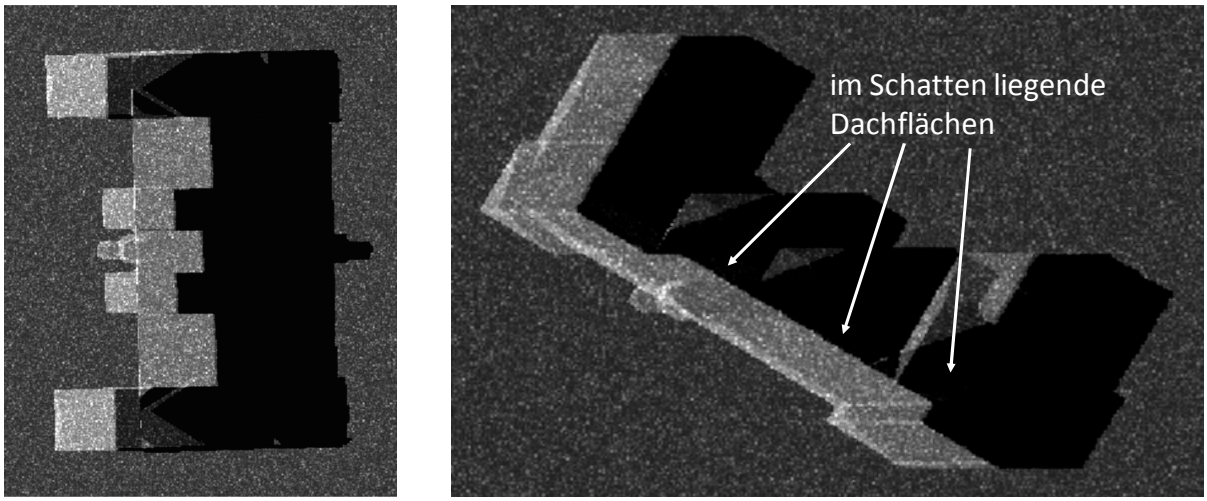

Abbildung 5-13: SAR-Effektvisualisierung des Gebäudemodells aus Abbildung 5-10 mit SARViz

Bei der Simulation eines vergleichsweise einfachen Gebäudemodells sind die Ergebnisse ähnlich und die Echtzeitvisualisierung liefert realistische Ergebnisse. Im Detail treten, wie in Abbildung 5-14 zu sehen, jedoch auch bei einfachen Gebäudemodellen Unterschiede zwischen den Simulatoren auf. Das in Abbildung 5-14 abgebildete Modell besteht komplett aus Stein und steht auf einer Grasfläche. Schatten, Layover und Kantenreflexionen sind bei allen drei Simulationsergebnissen vorhanden, unterschiedlich ist allerdings die

Reflexion des Daches und der Wand. Auf Grund der Materialrepräsentation simuliert SE-RAY-EM keine Reflexion zwischen den Wänden. Am realistischsten scheint das Ergebnis des Simulators von FGAN-FOM, jedoch treten Artefakte auf der Rückseite des Gebäudemodells auf. Die Echtzeitvisualisierung mit SARViz kann die Mehrfachreflexionen nicht darstellen. Auch die Reflexionen des Daches werden unterschiedlich simuliert. So treten bei der FOM-Simulation dreieckige Strukturen auf, die bei der Simulation durch SE-RAY-EM nur sehr schwach ausgeprägt sind und die Dachstruktur hier sehr einheitlich wirkt. Die SARViz Simulation liefert keine einheitliche sondern eine stufige Dachstruktur.

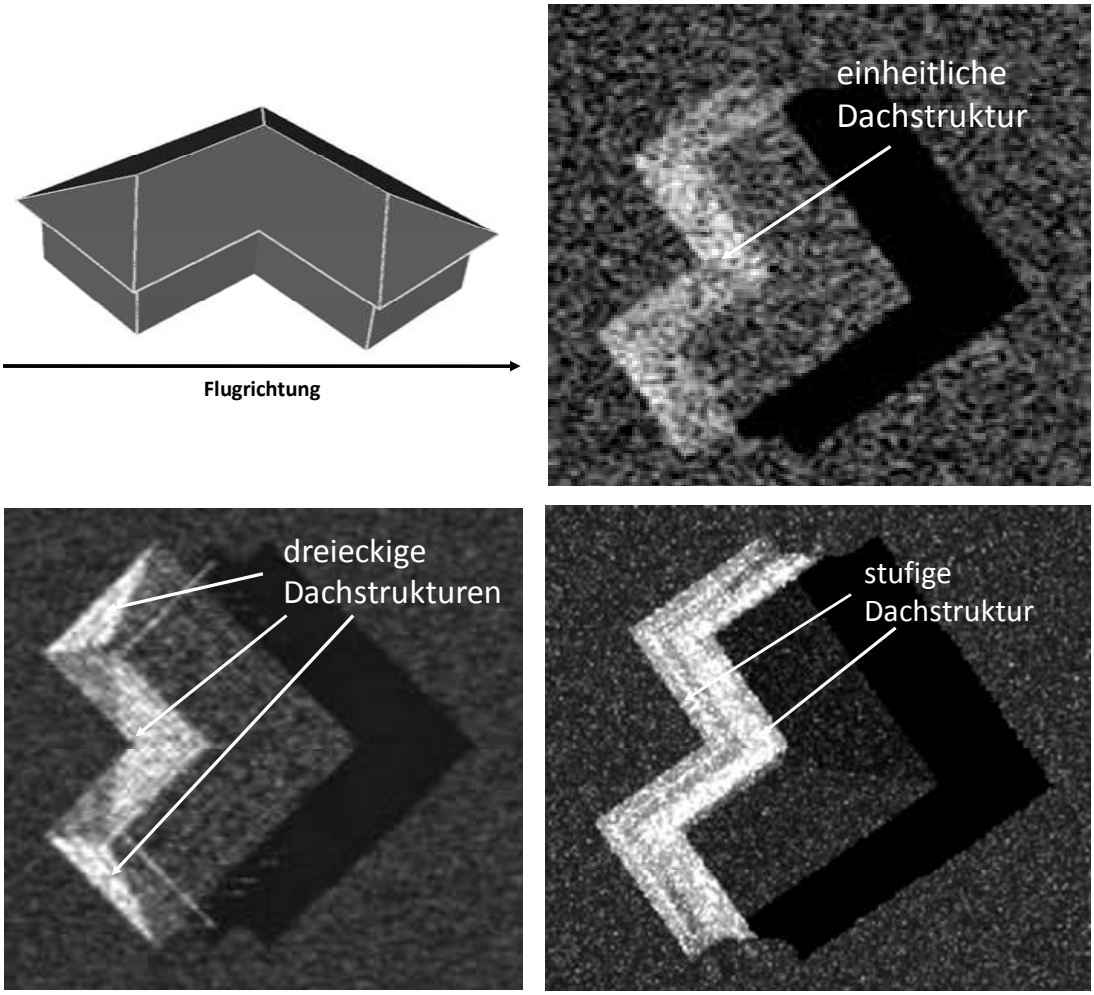

Abbildung 5-14: Simulation des Gebäudemodells (links oben). SE-RAY-EM (rechts oben), FOM-Simulator (links unten) und SARViz (rechts unten)

5.3.2 Simulation von Objekten mit spiegelnden Oberflächen

Die Simulation von Objekten mit metallischen bzw. spiegelnden Oberflächen ist komplizierter, da eine Reihe von Mehrfachreflexionen auftreten. Die SAR-Effektvisualisierung in Echtzeit ist nicht in der Lage, Mehrfachreflexionen korrekt darzustellen. Ein Vergleich der Simulationsergebnisse von komplexen metallischen Objekten zeigt daher die Einschränkungen bei der Simulationsgenauigkeit, die auf Grund der fehlenden Visualisierung von Mehrfachreflexionen entstehen. In Abbildung 5-15 sind die Ergebnisse der Simulation eines Modells der E-3 Sentry (Awacs) zu sehen. Das Modell wurde mit 40° Off-Nadirwinkel simuliert. Es besteht komplett aus Metall und befindet sich auf einer Grasfläche.

Das Ergebnis der SAR-Effektvisualisierung ist erwartungsgemäß vergleichbar mit der SE-RAY-EM Simulation mit einer Reflexion. Die bei der Simulation von Mehrfachreflexionen auftretenden Effekte zwischen dem Flugzeug und dem Radom können von SARViz nicht visualisiert werden. Solche Szenen sind in dieser Form in der Realität nur vergleichsweise selten zu beobachten, denn nur in Ausnahmefällen befinden sich Flugzeuge auf Grasflächen, normalerweise stehen sie auf Asphalt- oder Betonflächen. Diese Oberflächen wirken in Radarbildern jedoch wie Spiegel, so dass weit mehr Mehrfachreflexionen auftreten.

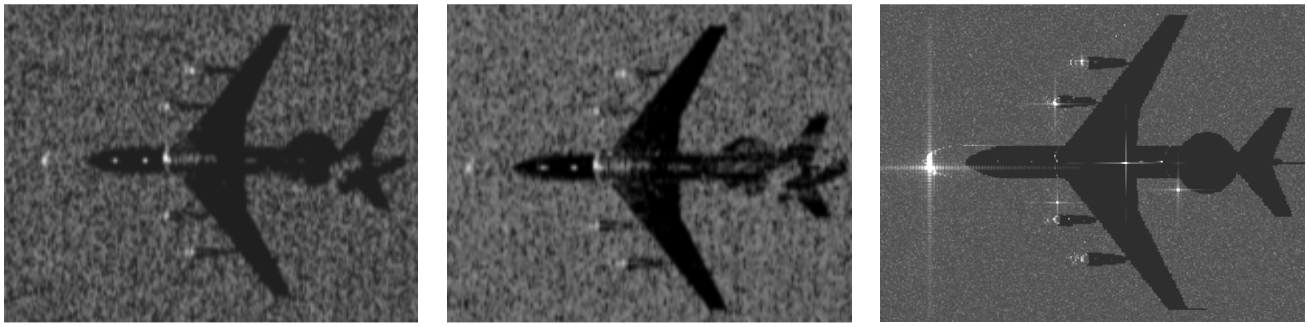

Abbildung 5-15: SAR-Simulation eines E-3 Sentry (Awacs) Modells durch SE-RAY-EM mit einer Reflexion (links), mit 16 Reflexionen (Mitte) und durch SARViz mit einer Reflexion und Nebenkeulen (rechts)

In Abbildung 5-16 ist das Modell einer E-3 Sentry auf einer Betonfläche stehend simuliert. Der Sensor hat einen Off-Nadirwinkel von 67° und das simulierte Flugzeugmodell besteht komplett aus Metall. Dadurch kommt es zu einer Vielzahl von Reflexionen zwischen dem Boden und dem Flugzeug. SE-RAY-EM simuliert diese Reflexionen, weswegen das Simulationsergebnis sehr realistisch ist. SARViz simuliert nur die einfache Reflexion sowie die Zweifachreflexion zwischen Boden und Flugzeugmodell. Das Ergebnis ist daher, wie in Abbildung 5-16 zu sehen, weit weniger realistisch. Zwar sind einige Effekte richtig abgebildet, auf Grund der fehlenden Mehrfachreflexionen und der nur rudimentären Unterstützung der Zweifachreflexion, ist das Simulationsergebnis allerdings weniger befriedigend.

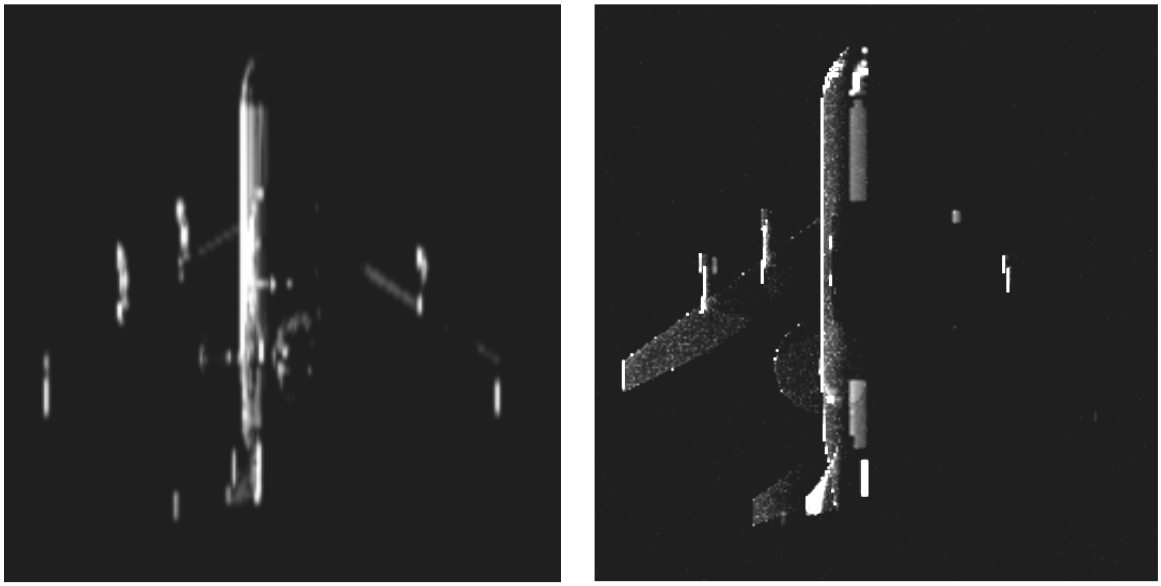

Abbildung 5-16: SAR-Simulation eines E-3 Sentry (Awacs) Modells auf einer Betonfläche durch SE-RAY-EM mit 16 Reflexionen (links) und durch SARViz und zwei Reflexionen ohne Nebenkeulen (rechts)

Auffällig sind die unterschiedliche Darstellung des Flugzeugrumpfes sowie die unterschiedliche Positionierung des Radoms im Bild. Grund für die unterschiedliche Position des Radoms ist, dass bei der Simulation durch SE-RAY-EM die Einfachreflexion des Radoms nur schwach ist, die stärke Mehrfachreflexion aber wird auf Grund der längeren Laufzeit des Signals weiter vom Sender entfernt abgebildet. Alles in allem ist die SAR-Effektvisualisierung bei der Abbildung von Objekten, die eine Vielzahl von Reflexionen aufweisen qualitativ deutlich schlechter.

Wirklichkeitsgetreue Szenen bestehen nicht nur aus einem, sondern aus einer Vielzahl von Objekten mit zum Teil komplexen Rückstreueigenschaften. SE-Workbench-EM ist in der Lage, auch diese komplexen Szenen zu simulieren, wie z.B. die in Abbildung 5-17 dargestellte Tankstelle. Die Tankstelle befindet sich auf einer Betonfläche und besteht zu großen Teilen aus ebenen Oberflächen, so dass es zu einer Reihe komplexer Rückstreuungen kommt.

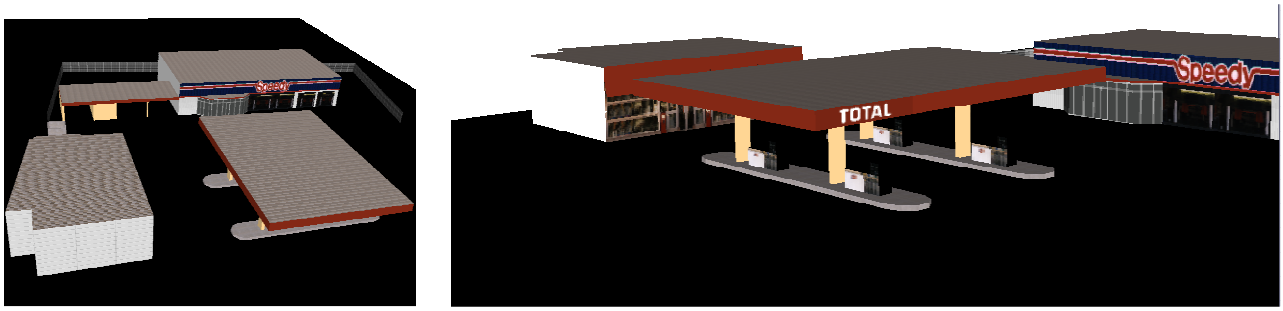

Abbildung 5-17: Mit SE-Workbench-EM modellierte Szene einer Tankstelle (Quelle: FGAN-FOM)

Zur besseren Visualisierung des komplexen Streuverhaltens wird bei den folgenden Simulationen auf ein Speckling verzichtet. Der Sensor blickt mit einem Off-Nadirwinkel von 63° auf die Szene. Die Simulationsergebnisse aus Abbildung 5-18 zeigen, dass die Qualitätsunterschiede der Simulationen bei Szenen mit zahlreichen Mehrfachreflexionen sehr groß sind. Bedingt durch den flachen Blinkwinkel des Sensors kommt es unterhalb der Überdachung des Tankbereiches zu einer Vielzahl von Mehrfachreflexionen. Da diese bei der SAR-Effektvisualisierung nicht berücksichtigt werden, ist das Ergebnis weniger realistisch.

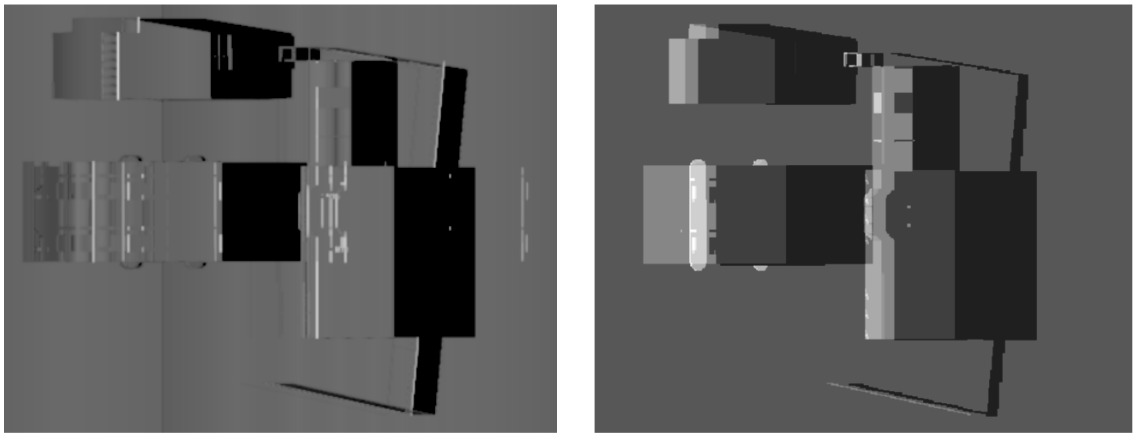

Abbildung 5-18: SAR-Simulation der Szene aus Abbildung 5-17. SE-RAY-EM mit 64 Reflexionen (links) und SARViz mit Doppelreflexion (rechts)

Für die Simulation von Mehrfachreflexionen ist die SAR-Effektvisualisierung in Echtzeit nicht geeignet. Dies liegt, wie in Kapitel 4.7 dargelegt, an der Rasterisierung, die keine Verfolgung der Strahlen und ihrer Reflexionen erlaubt. Für Szenen, in denen Einfachreflexionen überwiegen, liefert die SAR-Effektvisualisierung in Echtzeit jedoch realistische Ergebnisse in weit kürzerer Zeit. Die Rechenzeit für die in diesem Kapitel simulierten Szenen beträgt mit SE-RAY-EM mehrere Sekunden bis wenige Minuten, wohingegen die SAR-Effektvisualisierung weniger als 50 ms für die Visualisierung benötigt.

5.4 Die Simulation hochauflösender Daten

Neben der Qualität und der Geschwindigkeit der Simulation spielt die Fähigkeit, komplexe Szenen zu simulieren, eine wichtige Rolle bei der Beurteilung eines Simulators. Auf Grund der begrenzten Menge an zu berechenbaren Reflexionen sind alle Simulatoren in ihrer Komplexität begrenzt. Je aufwendiger ein Simulator arbeitet, desto geringer ist die maximale Anzahl an Objekten pro Simulation.

Hochauflösende SAR-Systeme, wie z.B. das PAMIR-System (Ender & Brenner, 2003; Brenner & Ender, 2004), erschließen neue Einsatzmöglichkeiten. Die hochauflösenden Bilder dieser Systeme sind jedoch in ihrer Interpretation oft schwierig. Kleine bauliche Besonderheiten, z.B. Gitter, können das Reflexionsverhalten eines Gebäudes stark verändern (Soergel et al, 2004; Soergel et al, 2006). Für die Simulation hochauflösender Daten bedeutet dies, dass die zu simulierenden Modelle eine hohe Qualität und Auflösung aufweisen müssen. Bei einer geometrischen Auflösung der Simulation von ca. 10 cm in Azimut- und Entfernungsrich-

tung sollte das verwendete Modell nach Möglichkeit ungefähr doppelt so hoch aufgelöst sein. Bei der Simulation von PAMIR-Daten würde dies einer Dreiecksseitenlänge von ungefähr 5 cm entsprechen.

Daten dieser Qualität sind flächendeckend jedoch nicht erhältlich. Sollten diese Daten verfügbar sein, ist mit einem Umfang von mehreren hunderttausend Dreiecken pro Gebäude zu rechnen. Dies könnte von einer Vielzahl von SAR-Simulatoren überhaupt nicht mehr verarbeitet werden. Der Umfang und die Komplexität der Modelle wären für die meisten Ray-Tracing-Simulatoren zu groß, um in akzeptabler Zeit visualisiert zu werden. Eine Möglichkeit, den Umfang der Daten zu begrenzen, ist es, die Modelle mit Texturen und Bump-Maps zu versehen. Dadurch kann die Zahl der zu visualisierenden Dreiecke begrenzt werden. Doch ist die Erzeugung solcher Texturen sehr zeitaufwendig.

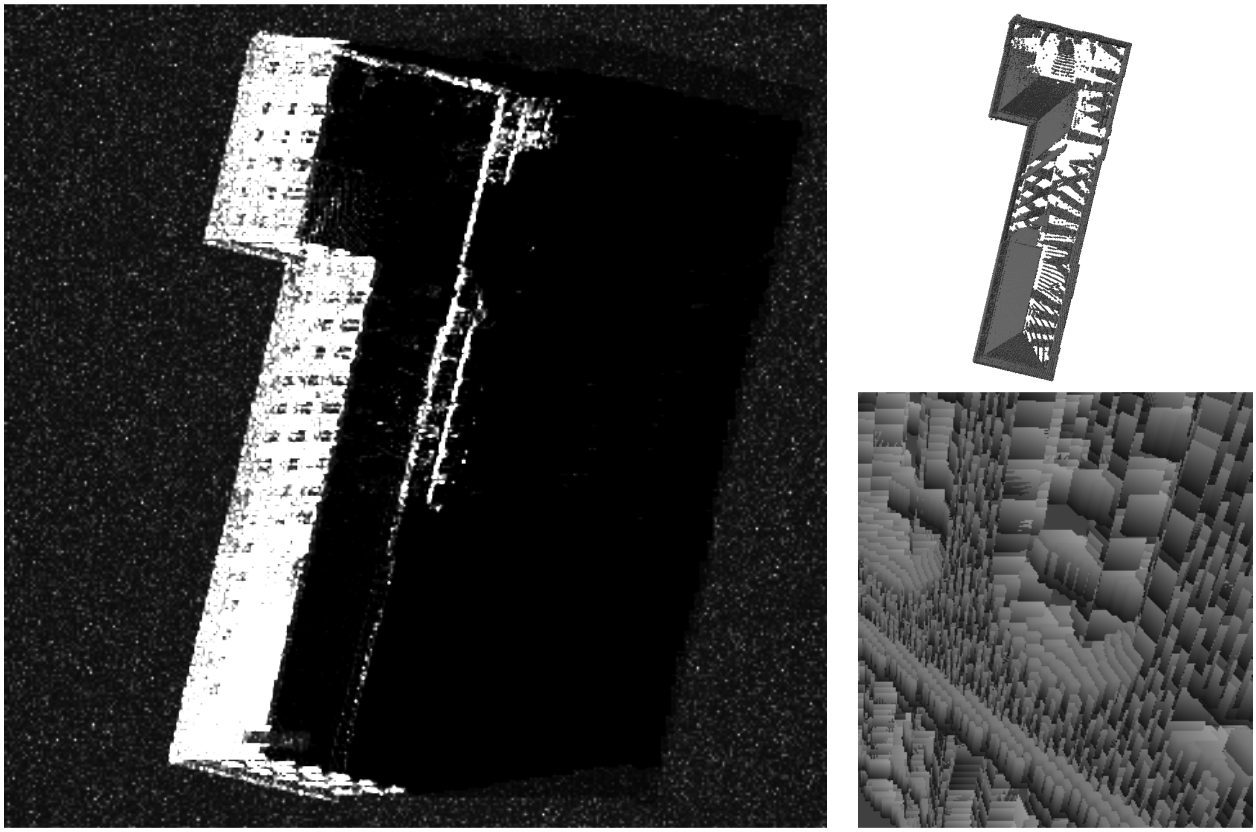

Abbildung 5-19: Visualisierung basierend auf terrestrisch erfasster 3D-Punktwolke des Rektorats der Universität Stuttgart. Links: SAR-Effektvisualisierung. Rechts: Darstellung der Punktwolke aus Nadirsicht (oben) und Ausschnitt aus der Nahansicht der Gebäudefassade (unten)

Bereits die Erzeugung der 3D-Modelle ohne Texturen oder Bump-Maps ist aufwendig. Modelle mit dieser hohen Auflösung können von terrestrischen Laserscannern erfasst werden. Um jedoch die mit terrestrischen Laserscannerdaten erfassten Modelle simulieren zu können, müssen die vom Laserscanner gelieferten Punktwolken zunächst trianguliert werden. Dieser Prozess ist aufwendig und die Ergebnisse sind ohne manuelle Nacharbeit meist nicht zufriedenstellend. Eine Alternative bei der Darstellung von Punktwolken ist das punktbasierte Rendering (siehe z.B. Gross, 2001; oder Sainz & Pajarola, 2004). Hierfür werden die Punkte nicht trianguliert, sondern sie werden direkt dargestellt.

In Abbildung 5-19 ist eine vergleichbare Methode dargestellt. Die aus den Laserscanningdaten erzeugte Punktwolke wurde nicht trianguliert, stattdessen werden die Punkte durch Quader visualisiert. Die Größe der Quader wird abhängig von der Punktdichte auf der abgetasteten Häuserfassade gewählt und beträgt in diesem Beispiel 20x20x20 cm^3. Die Überlagerungen der einzelnen Quader lassen sich bei näherer Betrachtung der Fassade in Abbildung 5-19 (rechts unten) erkennen. Diese Überschneidungen beeinträchtigen die Visualisierung jedoch nicht. Durch diesen einfachen Ansatz lassen sich komplexe Punktwolken einfach und ohne manuelle Nacharbeit für die SAR-Visualisierung nutzen.

Ein Hauptproblem sind die Dachflächen, die von terrestrischen Laserscannern oft gar nicht oder nur mit großen Lücken erfasst werden. In Abbildung 5-19 reflektiert, durch die Lücken im Dach, auch die Rückseite des Gebäudes teilweise zurück. Durch manuelle Nacharbeit könnten nach einer Triangulation diese Lücken im Modell geschlossen werden. Trotz dieser Einschränkung ermöglicht die Punktvisualisierung eine schnelle und realistische hochauflösende Simulation von komplexen Gebäuden, ohne dass eine manuelle Nachbearbeitung der Daten nötig wäre. Bemerkenswert sind die klar erkennbaren Fenster und Wandstrukturen, die entscheidend sind für die realistische Darstellung von hochauflösenden Gebäuden.

6 Einsatzmöglichkeiten der SAR-Effektvisualisierung

Die Zahl möglicher Anwendungen für die Echtzeitvisualisierung von SAR-Effekten ist hoch. Im Folgenden soll daher ein kurzer Überblick über einige mögliche Anwendungen gegeben werden. SAR-Simulationen werden im Sensordesign und beim Entwickeln neuer Verfahren zur SAR-Datenverarbeitung eingesetzt. Die auf Grafikkarten basierte SAR-Echtzeitvisualisierung ist für diese Anwendungen nicht geeignet. Der Einsatz von Simulatoren beim Entwurf von Sensoren oder bei der Entwicklung von Verfahren zur SAR-Datenverarbeitung erfordert eine detaillierte Simulation der physikalischen Abläufe. Die hierfür eingesetzten Simulatoren sind vergleichsweise langsam aber genau. Weder beim Sensordesign noch bei der Erstellung von Testdaten für die Entwicklung von SAR-Datenverarbeitungsverfahren ist eine Echtzeitfähigkeit von Bedeutung.

Anwendungen, die von der SAR-Echtzeitvisualisierung profitieren können, sind bisherige Nischenanwendungen der SAR-Simulation. Obwohl es sich um wichtige Anwendungen handelt, wird für diese die SAR-Simulation erst durch die schnelle Verfügbarkeit von Simulationsergebnissen interessant. Die SAR-Effektvisualisierung in Echtzeit kann ein wichtiges Hilfsmittel bei der SAR-Missionsplanung sein. Die Ausbildung der Nutzer von SAR-Daten und die Lehre an der Hochschule werden durch die SAR-Visualisierung verbessert, da die SAR-Effekte erlebbar werden. Neben der verbesserten Ausbildung profitiert die SAR-Auswertung von schnell verfügbaren Simulationsergebnissen. Der Vergleich zwischen Simulation und realen SAR-Daten ist ein Vergleich zwischen Erwartung und Realität. Dadurch kann sowohl die automatische, als auch die manuelle Objekterkennung und Veränderungsdetektion unterstützt werden.

6.1 Missionsplanung

Bereits die erste Radarsimulation von *La Prade* (1963) diente der Missionsplanung für die Erfassung stereoskopisch auswertbarer Radardaten. Auch *Kaupp et al* (1983) und *Guindon* (1993) entwickelten SAR-Simulationen für die Missionsplanung. Früh wurde erkannt, dass die Planung einer SAR-Mission den Grundstein für die erfolgreiche SAR-Datengewinnung legt. Dabei sind insbesondere Effizienz und Kostenminimierung wichtig. In Abhängigkeit der Anwendung, für die Daten erhoben werden sollen, spielt die Zeitplanung eine gewichtige Rolle. Ziel ist es, ein bestimmtes Gebiet zu einem bestimmten Zeitpunkt möglichst kostengünstig zu beobachten. Die Faktoren Zeit und Kosten sind in Abhängigkeit der Anwendung unterschiedlich gewichtet. Für militärische Anwendungen, Anwendungen im Katastrophenmanagement oder bei großräumigen Überwachungen ist der Faktor des Zeitpunktes oft wichtiger als die Kostenminimierung. Bei der großflächigen Datengewinnung, z.B. für die Geodatenerstellung, sind die Kosten entscheidend, der Zeitpunkt der Datenerhebung ist weniger wichtig. Daneben gibt es oft noch weitere Einschränkungen, die zu berücksichtigen sind, wie z.B. die Bahnparameter der Sensoren, Überflugsrechte, Sensoreigenschaften, Gefährdungslage, usw. All diese Faktoren müssen bei der Missionsplanung berücksichtigt werden. Ein sehr wichtiger, wenn nicht gar der wichtigste, Punkt bleibt oft unbeachtet, nämlich die entscheidende Frage: wird das zu beobachtende Gebiet überhaupt beobachtet?

Durch Verdeckungen und Abschattungen kann das aufzunehmende Gebiet evtl. nicht oder nur lückenhaft erfasst werden, wodurch die SAR-Aufnahme wertlos wird. Die verdeckungsfreie Beobachtung kann, vorhandene Vorabinformationen z.B. in Form von Höhen- oder Stadtmodellen vorausgesetzt, durch eine SAR-Simulation bereits vor dem Flug überprüft werden. Eine Schatten- und Overlayanalyse ist hierfür ausreichend (Soergel et al, 2003). Dabei ist zu beachten, dass Stadtmodelle oftmals keine Bäume enthalten, obwohl Bäume einen erheblichen Teil der Oberfläche in urbanen Räumen bedecken können (Stilla et al, 2003). Daher sind oftmals digitale Oberflächenmodelle, z.B. aus LIDAR-Daten, den Stadtmodellen vorzuziehen. Bei teilweise verdeckten Gebieten ist es evtl. nötig, das Gebiet aus mehreren Blickwinkeln zu betrachten. Dies kann unter Zuhilfenahme von SAR-Simulatoren vorab untersucht werden. Die SAR-Effektvisualisierung ermöglicht die Durchführung von Sichtbarkeitsanalysen in Echtzeit und steigert so die Effizienz bei der Missionsplanung.

Die Missionsplanung ist angesichts der wachsenden Zahl unterschiedlicher Sensoren und Plattformen ein komplexer Vorgang von zunehmender Bedeutung für die Erdbeobachtung. Die Integration unterschiedlicher Systeme und Sensoren in die Missionsplanung wird immer wichtiger, wobei für die Planung auch die Sichtbarkeitsanalyse mittels Simulationen entscheidend sein kann (Brand et al, 2006). Die Verdeckungsanalyse

hilft bei der Auswahl des richtigen Sensors und der passenden Sensoreinstellungen. Die Missionsplanung für bistatische oder multi-aspect Missionen ist komplizierter und verlangt eine noch gründlichere Planung.

Verdeckungen treten insbesondere im urbanen Raum häufig auf und die gewünschte Beobachtung bestimmter Gebiete innerhalb eines städtischen Raumes bedarf einer besonders sorgfältigen Analyse der Verdeckungen. Die Bedeutung des urbanen Raumes nimmt in der Fernerkundung, auch in der SAR-Fernerkundung, ständig zu. Dies liegt zum einen an der wachsenden Bedeutung dieser Räume für die Menschheit insgesamt, als auch an der zunehmenden Auflösung der verfügbaren (SAR-)Sensoren, was deren sinnvollen Einsatz im urbanen Raum erst ermöglicht. Bereits jetzt leben mehr Menschen in den Städten als außerhalb und der Anteil der städtischen Bevölkerung wächst weiterhin (UN-HABITAT, 2006).

Fernerkundung in urbanen Räumen erfordert eine hohe geometrische Auflösung der beobachtenden Sensoren, damit die oft eng beieinander stehenden Objekte innerhalb der verdichteten Raumstrukturen klar voneinander unterschieden werden können. In Abhängigkeit von der Anwendung und der Bebauungsdichte sollte die erzielte räumliche Auflösung bei einem Meter oder darunter liegen. Moderne flugzeuggestützte Systeme, wie das PAMIR-System, erreicheichen eine Auflösung im Dezimeterbereich (Ender & Brenner, 2003), zukünftige Satellitensysteme, wie z.B. TerraSAR-X, im Meterbereich (Werninghaus et al, 2004). Diese Systeme erfüllen damit die Anforderungen für die Fernerkundung in städtischen Räumen.

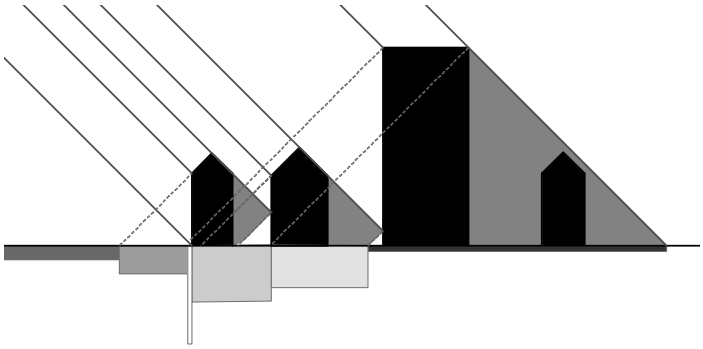

Abbildung 6-1: Verdeckungen und Mehrdeutigkeiten von SAR-Bildern aus urbanen Räumen

Doch neben dem Sensor müssen die Missionsplanung und die Auswertung den Anforderungen gerecht werden. Wie in Abbildung 6-1 dargestellt, sind bei SAR-Aufnahmen aus urbanen Räumen weite Bereiche verdeckt. Liegt das zu beobachtende Objekt in einem verdeckten Bereich, so ist eine Aufnahme sinnlos. Um Zeit und Kosten zu sparen, sollte dies bereits bei der Missionsplanung erkannt werden.

Neben der manuellen Missionsplanung und der interaktiven Suche nach den besten Missionsparametern, kann der Anteil der Verdeckung automatisch ermittelt und durch Brute-Force-Techniken optimiert werden. Unter Brute-Force versteht man das Ausprobieren aller möglichen Fälle zur Ermittlung des besten Ergebnisses. Ausgehend von einer Analyse in 1° Schritten gibt es 360 mögliche Azimutwinkel. Bei angenommenen 50 sinnvollen Off-Nadirwinkeln (30°-80°) gibt es somit 18.000 mögliche Kombinationen. Diese können bei einer Simulationsgeschwindigkeit eines komplexen Stadtmodels von angenommenen 50 ms, in nur 15 Minuten simuliert werden. Für diese Sichtbarkeitsanalyse ist keine komplette Simulation nötig, sondern es genügt die Erstellung der Sensor Map. Dies spart weitere Rechenzeit. Die auftretenden Schattenbereiche werden innerhalb des zu beobachtenden Bereiches markiert. Durch Reduktion (Harris, 2005) (siehe auch Kapitel 6.4) wird auf der Grafikkarte der Anteil des Schattenbereiches berechnet, wodurch die Winkelkombination mit der geringsten Verdeckung allein durch Brute-Force in weniger als 15 Minuten berechnet werden kann.

Um die Verdeckungen und Abschattungen berechnen zu können, wird ein Höhenmodell des Untersuchungsgebietes benötigt. Bei geplanten SAR-Aufnahmen mit hoher Auflösung in bebauten Gebieten ist zusätzlich ein Stadtmodell bzw. ein DOM mit hoher Auflösung erforderlich. Höhenmodelle mit geringer räumlicher Auflösung sind, z.B. aus SRTM-Daten, nahezu weltweit verfügbar. Das aus den SRTM-Daten abgeleitete Höhenmodell eignet sich zur großräumigen Verdeckungsanalyse außerhalb bebauter Gebiete. Verdeckungen durch Berge oder innerhalb von Tälern lassen sich durch die Verwendung dieser Daten bereits bei der Missionsplanung vorherberechnen, wie in Abbildung 6-2 beispielhaft dargestellt.

Abbildung 6-2: SAR-Visualisierung von SRTM-Daten eines Tals auf der Schwäbischen Alb in der Nähe von Bärenthal nahe Tuttlingen (links SAR-Visualisierung, mittig Visualisierung ohne SAR-Geometrie und Speckling, rechts maskierte Schattenbereiche)

Die Schattenbereiche werden direkt maskiert, so dass diese schnell lokalisierbar sind bzw. ihr Anteil im Bild einfach berechnet werden kann. Die gleiche Methodik funktioniert auch in urbanen Räumen, allerdings ist die Auflösung der frei verfügbaren SRTM-Daten dort zu schlecht. Stadtmodelle oder digitale Oberflächenmodelle, z.B. aus LIDAR-Daten, müssen hier als Datengrundlage für die Ermittlung der verdeckten Bereiche dienen. Die üblicherweise erhältlichen flächendeckenden LIDAR-Datensätze verfügen über eine Punktdichte von einem Punkt pro m². Damit ist eine grobe Analyse der Verdeckungen möglich. In Abbildung 6-3 ist die weite Verbreitung der Schattenflächen im urbanen Raum zu erkennen. Der Off-Nadirwinkel beträgt 55°, bei flacheren Blickwinkeln oder bistatischen Aufnahmen ist eine weitere Zunahme der Schattenflächen zu erwarten. Im Extremfall besteht eine SAR-Aufnahme im urbanen Raum nahezu ausschließlich aus Schatten und Layover. Es kann sinnvoll sein, zusätzlich die Layoverbereiche zu ermitteln und zu maskieren. Gebiete, die durch Layover überdeckt sind, werden zwar erfasst, doch lassen sich Objektreflexionen und Layoverreflexionen kaum mehr unterscheiden. Dies erschwert die Auswertung der Objekte.

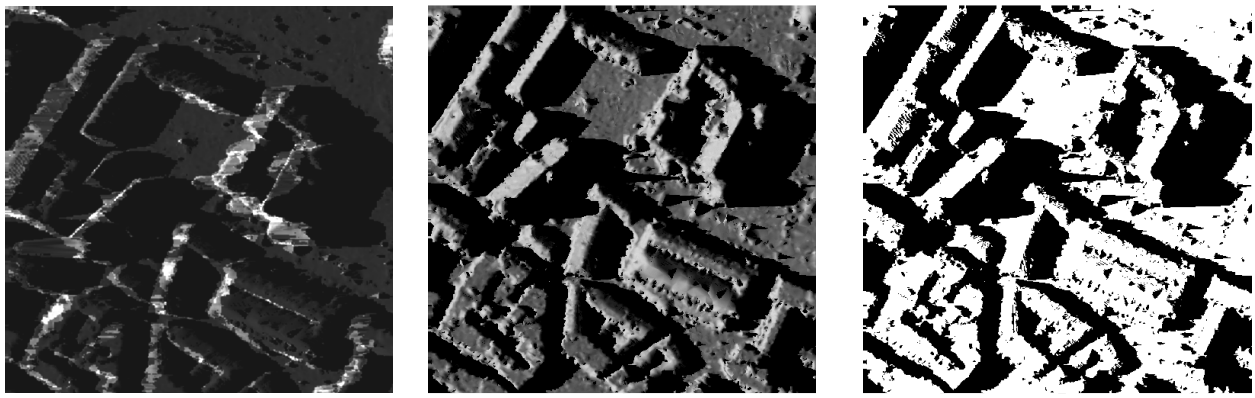

Abbildung 6-3: SAR-Visualisierung von LIDAR-Daten aus dem Stadtzentrum von Stuttgart (links ohne Speckling, mittig ohne SAR-Geometrie, rechts maskierte Schattenbereiche)

Schatten und Layover sind aber nicht nur störend, sondern können die Auswertung auch unterstützen. So können durch eine gezielte Schatten- und Layoveranalyse Objekthöhen bestimmt werden (Kirscht & Rinke, 1998). Auf eine gute Sichtbarkeit des Schattens und des Layoverbereiches ist dabei zu achten. Beide Bereiche dürfen hierfür nicht verdeckt sein. Dies sollte, falls eine solche Analyse gewünscht ist, bereits bei der Missionsplanung berücksichtigt werden. In urbanen Räumen ist es oftmals unmöglich, unverdeckte und störungsfreie Schatten- und Layoverbereiche zu erhalten. Aufschlussreicher, aber noch planungsintensiver, ist die Analyse von bistatischen Schattenwürfen. Ohne eine vorhergehende umfangreiche Planung können kaum die bistatischen Winkel, die eine störungsfreie Auswertung erlauben, bestimmt werden. Wegen der Zunahme freier Parameter durch den zweiten Sensor wird die Planung komplizierter und eine schnelle Schatten- und Verdeckungsanalyse in der Missionsplanung umso bedeutender.

Die Missionsplanung ist eine komplexe Aufgabe und die Simulation oder die Visualisierung möglicher Ergebnisse ist nur ein Teil dieser Aufgabe. Die Simulation muss daher ein solches System ergänzen und sich in ein vorhandenes System nahtlos einfügen lassen. Daher sind ein modulares Design sowie die Verwendung verbreiteter und möglichst freier Datenformate unerlässlich für die Integration. Die Visualisierung von Verdeckungen ist kein Selbstzweck, sondern nur im Verbund mit weiteren Planungstools sinnvoll. Die Vorteile einer Simulation in Echtzeit bei der Planung schwieriger Missionen, insbesondere im urbanen oder alpinen Raum, sind nicht zu übersehen.

6.2 Lehre und Ausbildung

Die Auswertung von SAR-Daten ist schwierig. Menschen, die zum ersten Mal mit SAR-Daten konfrontiert werden, können meist nichts erkennen und selbst erfahrene Anwender haben manchmal Schwierigkeiten, bestimmte SAR-Effekte zu erklären. Der zunehmende Einsatz von hochauflösenden SAR-Sensoren im militärischen (z.B. SAR-Lupe) oder zivilen (z.B. TerraSAR-X) Bereich erfordert die Ausbildung von Experten. Es werden aber nicht nur Experten mit SAR-Daten konfrontiert, auch Laien brauchen Hilfsmittel, um die Daten zu verstehen und um die erkannten Effekte richtig einordnen zu können.

Die SAR-Simulation kann ein wichtiges Hilfsmittel bei der Lehre und in der Ausbildung sein. Neben den theoretischen Erläuterungen der SAR-Prinzipien ist die Betrachtung von unterschiedlichen SAR-Daten ein wichtiger Teil der Ausbildung, denn nur dadurch werden die SAR-Daten verständlich und können weitergehend interpretiert werden. Die SAR-Simulation leistet hierfür einen wichtigen Beitrag, da sie eine Vielzahl von SAR-Bildern kostengünstig liefern kann. Darüber hinaus lassen sich bei der Simulation genau definierte Szenen und Objekte darstellen, was den Lerneffekt erhöhen kann. Die Ausbildung für die Auswertung von SAR-Daten für noch nicht vorhandene Sensoren, also z.B. die Ausbildung der Auswerter der SAR-Lupe, ist nur mit simulierten Daten möglich. Dabei wird normalerweise auf reale vorhandene Daten aus Befliegungen zurückgegriffen. Diese Daten werden dann entsprechend den Parametern der Satellitenmission prozessiert, um Satellitendaten zu simulieren.

Die echtzeitfähige Simulation eröffnet neue Möglichkeiten bei der Ausbildung. Die Auszubildenden können die Szenen interaktiv ändern und sehen sofort die Auswirkungen. Das veränderte Erscheinungsbild eines Objektes bei unterschiedlichen Blickwinkeln kann direkt nachvollzogen, sozusagen erlebt werden. Die stärkere Interaktivität erhöht den Lerneffekt. Daneben können bei der Simulation einzelne Effekte ein- bzw. ausgeschaltet werden, was bei realen Bildern nicht möglich ist. SAR-Daten ohne Speckling können ebenso dargestellt werden wie nicht spiegelnde Flugzeuge. Dadurch können die einzelnen Effekte separiert werden und ihre Auswirkungen werden einzeln erfahrbar.

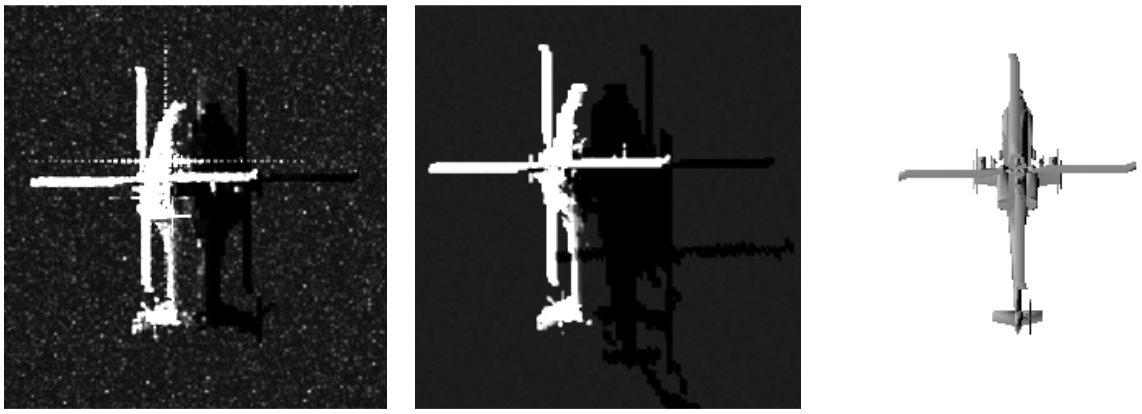

Abbildung 6-4: Mögliche Visualisierungen eines Helikoptermodells. Mit Zweifachreflexion (links), bistatisch ohne Zweifachreflexion, Nebenkeulen und Speckling (Mitte) und optisch (rechts)

Die begrenzte Anzahl darstellbarer Effekte ist bei der Ausbildung, insbesondere am Anfang, von Vorteil. Die grundlegenden Effekte können visualisiert werden und durch den direkten Vergleich mit der dargestellten Szene wird die unterschiedliche SAR-Geometrie und Radiometrie begreifbar. Die Visualisierung erlaubt es z.B. SAR-Daten ohne SAR-Geometrie darzustellen, d.h. ohne Layover, aber mit SAR-Radiometrie und Speckling.

Ein weiterer bedeutender Vorteil ist der mögliche direkte Vergleich zwischen der SAR-Visualisierung und der optischen Visualisierung eines Modells. Einem unerfahrenen Anwender ist die Geometrie der Szene in Abbildung 6-5 vielleicht zunächst nicht klar. Durch die Darstellung der Szene aus der Sensorrichtung wie in Abbildung 6-6 (rechts) dargestellt, kann die Geometrie des Bildes schneller verstanden werden. Die zusätzliche Möglichkeit, den Einfluss der Eigenbewegung der Objekte zu visualisieren, erhöht den Praxisbezug und den Lerneffekt.

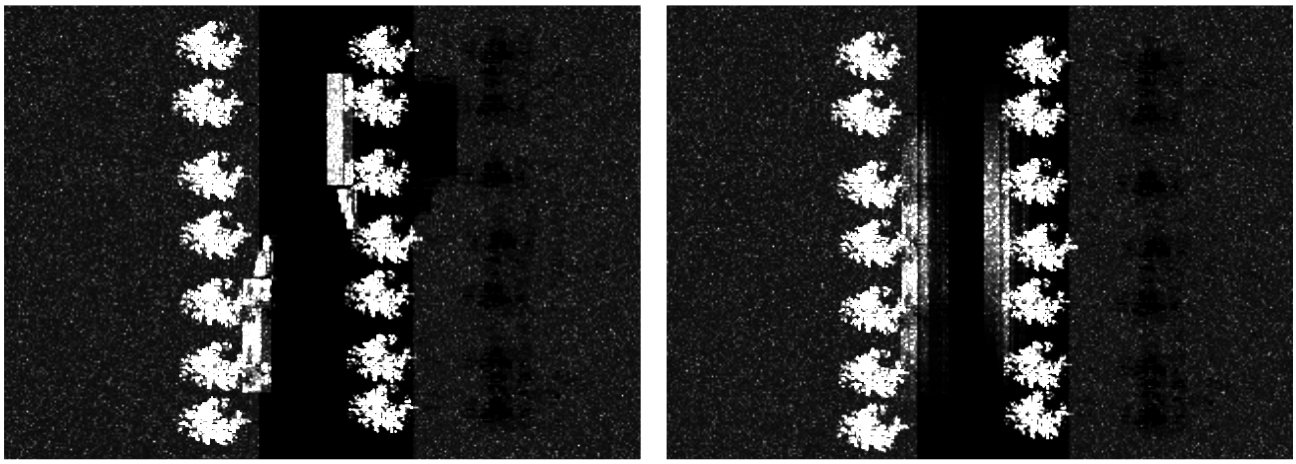

Abbildung 6-5: SAR-Simulation zweier stehender (links) bzw. fahrender LKW (rechts)

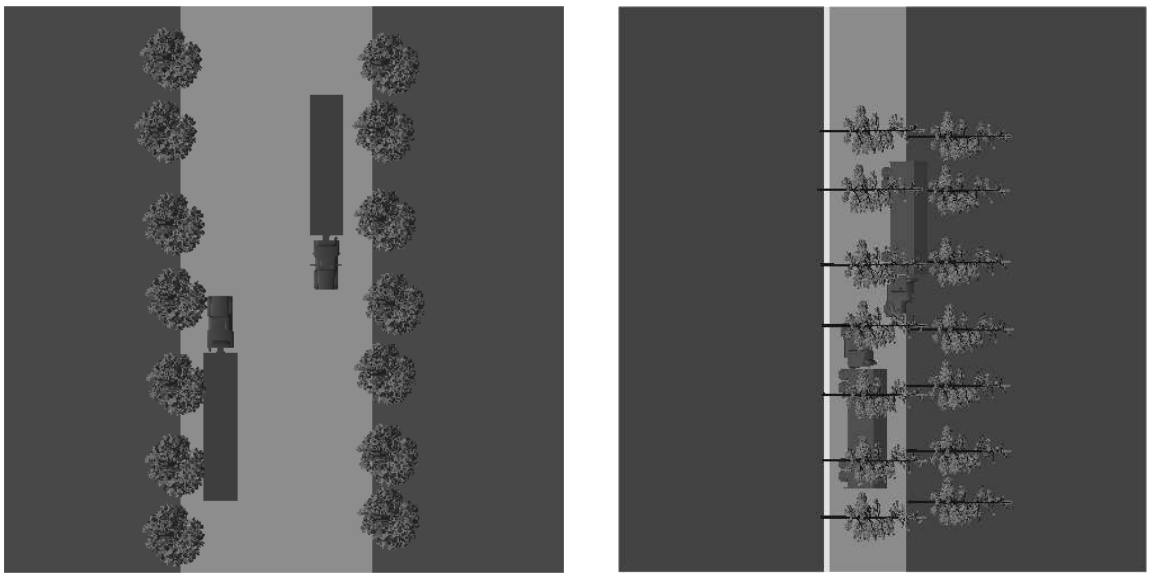

Abbildung 6-6: Visualisierung der unter Abbildung 6-5 gezeigten Szene aus Nadir (links) und aus Blickrichtung des Sensors (rechts)

In Abbildung 6-6 ist die in Abbildung 6-5 dargestellte SAR-Szene als optische Szene visualisiert. Ein typisches Problem bei der Analyse von SAR-Daten ist die Schrägsicht des Sensors. Ein Vergleich zwischen Nadirsicht und Sensorsicht auf die Szene und dem visualisierten SAR-Bild macht dies deutlich. Hierbei sind SAR-Simulatoren ein nicht zu unterschätzendes Hilfsmittel bei der Ausbildung, da sie diesen Effekt erfahrbar machen.

So hilfreich die Echtzeitvisualisierung auch ist, die Simulationsergebnisse sind, insbesondere hinsichtlich der Mehrfachreflexionen, ungenau bzw. fehlerhaft. Hier kann der Einsatz von simulierten Daten von z.B. Rohdatensimulatoren sinnvoll sein. Im Idealfall können beide Simulatoren die gleiche Szene simulieren und immer mehr „Features“ zuschalten. Die Ergebnisse können letztlich mit einem realen SAR-Bild verglichen werden. Dadurch können die einzelnen Effekte, wie z.B. die Zweifachreflexion, einzeln sichtbar gemacht werden und gleichzeitig wird der Unterschied zwischen Simulation und Realität deutlich. Durch diese Kombination kann ein tieferes Verständnis für die SAR-Datenauswertung erzeugt werden.

6.3 Auswertung und Objekterkennung

Die Auswertung von SAR-Daten erfordert Expertise, sowohl in der Sache z.B. in Geologie, Gewässerkunde oder der militärischen Objekterkennung, als auch in der SAR-Theorie. Viele Details aus SAR-Bildern können bei der Bewertung wichtig sein und es erfordert eine gewisse Erfahrung im Umgang mit SAR-Daten, um die für die jeweilige Aufgabe entscheidende Information in den Daten zu erkennen.

Bei der Erkennung von Einzelobjekten, insbesondere für die militärische Auswertung von Aufklärungsdaten, ist ein großes Wissen über die aufzuklärenden Objekte erforderlich. Auswerter sind daher spezialisiert auf verschiedene Teilbereiche möglicher Objekte, also z.B. auf Flugzeuge, Schiffe oder strategische Objekte und Gebäude. Für die Objekterkennung werden die eigene Erfahrung und Nachschlagewerke benötigt. Digitale Hilfsmittel gibt es z.B. in Form des RecceMan (Geisler et al, 2000; Geisler et al, 2004) des Fraunhofer Instituts für Informations- und Datenverarbeitung. RecceMan unterstützt den Bildauswerter in dem durch Auswahl verschiedener Merkmale die Anzahl möglicher Kandidaten bei der Auswertung eingegrenzt wird. Soll z.B. der Typ eines Flugzeugs erkannt werden, kann die Zahl möglicher Kandidaten durch die erkannte Anzahl an Triebwerken, die Flügelform und die Spannweite sehr schnell soweit reduziert werden, bis letztlich das Objekt klassifiziert werden kann. Die Datenbank enthält darüber hinaus mehrere Ansichten und Photos des Objektes und präsentiert somit dem Auswerter eine Fülle von Daten, anhand derer eine qualifizierte Entscheidung getroffen werden kann.

In Abbildung 6-7 ist beispielhaft dargestellt, wie selbst eine sehr vereinfachte SAR-Simulation bei der Auswertung helfen kann. Über das Originalbild (links) wird in der Mitte und auf der rechten Seite von Abbildung 6-7 je ein simuliertes Modell einer Transportmaschine vom Typ C-160 Transall bzw. C-130 Hercules eingeblendet. Beide Maschinen sind in ihren Bemaßungen ähnlich, die Hercules hat allerdings vier Triebwerke, die Transall nur zwei. Die Zahl der Triebwerke ist ein deutliches Unterscheidungskriterium, aber in SAR-Daten sind die Triebwerke nicht immer klar zu identifizieren, wie in Abbildung 6-7 (links) zu erkennen ist.

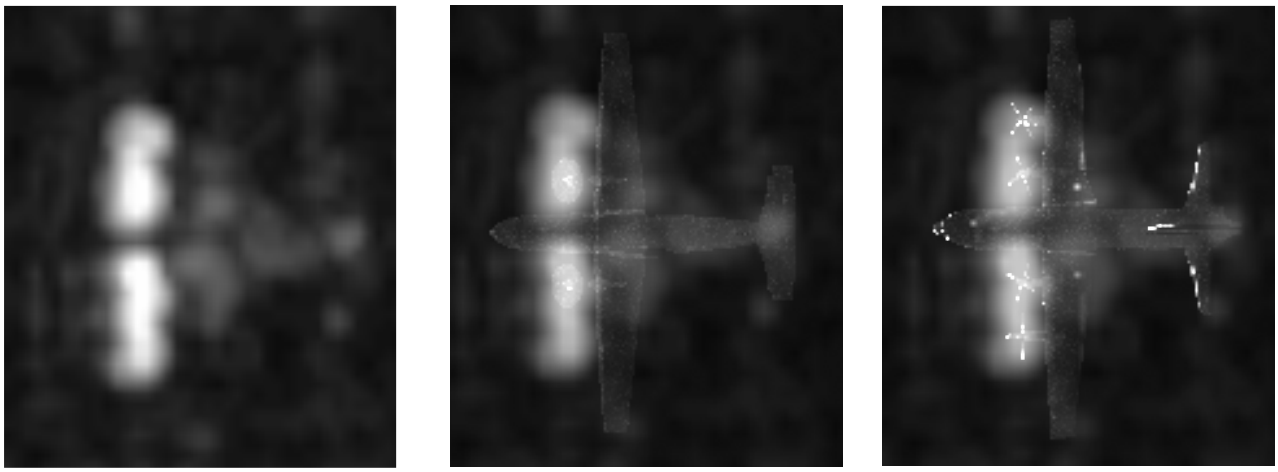

Abbildung 6-7: Übereinanderblenden von Original DOSAR-Bild (links ; EADS Deutschland) und Simulation eines Transallmodells (Mitte) und eines Herculesmodells (rechts) als Auswertehilfe

Bei dem in Abbildung 6-7 abgebildeten SAR-Bild handelt es sich tatsächlich um eine „C-130 Hercules“. Der Vergleich zwischen der Simulation und dem weit schlechter aufgelösten Original zeigt die Übereinstimmung zwischen dem simulierten Modell und dem Originalbild, besonders an den sehr stark reflektierenden Bereichen. Die Reflexion der vier Triebwerke im Originalbild zeigt deutlich, dass es sich nicht um eine „C-160 Transall“ handeln kann. Beim Vergleich der Simulation des Modells einer „C-130 Hercules“ fällt auf, dass die Positionen der vier simulierten Triebwerksreflexionen identisch mit den vier Reflexionen im Originalbild sind. Auffällig sind jedoch die fehlende bzw. schwach ausgeprägte Reflexion des Leitwerkes und die schwache Zweifachreflexion im Originalbild, so dass die Identifikation nur über die Triebwerke möglich ist.

Ein weiteres Einsatzgebiet ist die halbautomatische Rekonstruktion von Gebäudemodellen. Bereits mit einfachen Mitteln kann eine halbautomatische Gebäuderekonstruktion, die auf dem Vergleich von simulierten Da-

ten mit realen Daten beruht, realisiert werden (Balz & Haala, 2003). Durch den Einsatz einer Echtzeitsimulation können komplexe Gebäude- oder Objektmodelle mit dem realen SAR-Bild verglichen werden. Der Operator setzt das zu rekonstruierende Gebäudemodell auf die Position des Gebäudes im SAR-Bild und passt das Gebäude interaktiv solange an, bis die Simulation der Realität in weiten Teilen entspricht. Durch Überblendung kann die Simulation ständig mit dem realen SAR-Bild verglichen werden.

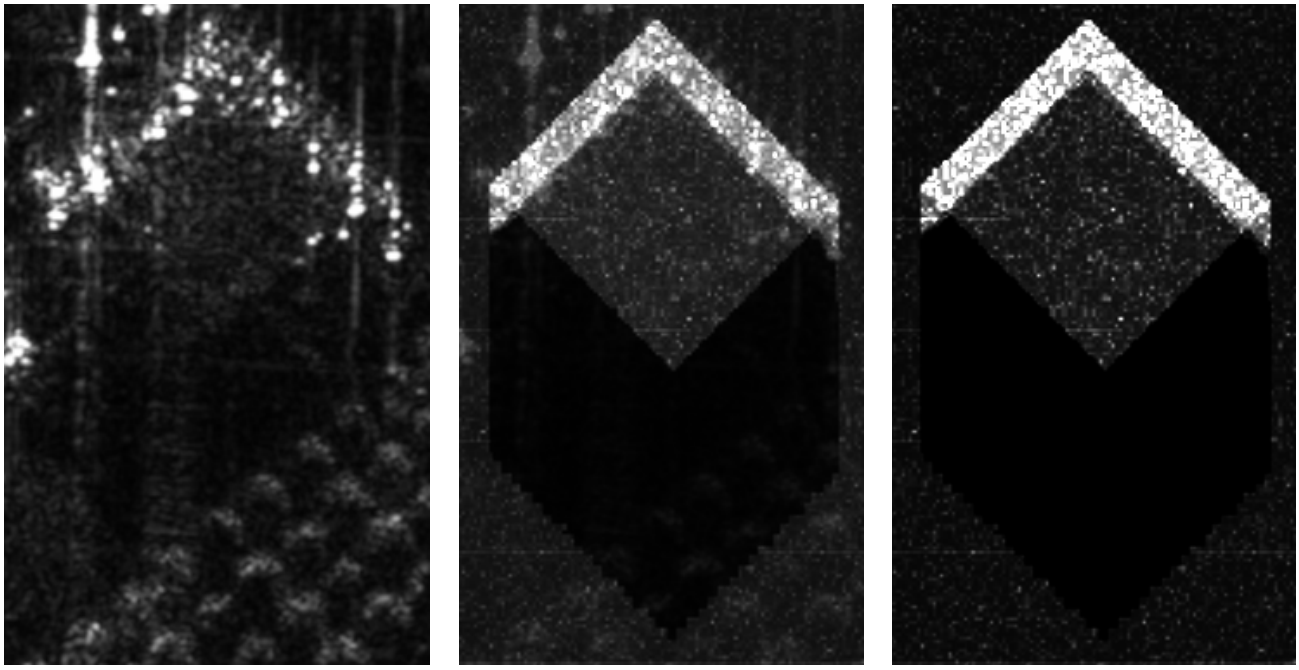

Abbildung 6-8: Halbautomatische Rekonstruktion von Gebäuden durch Vergleich zwischen Simulation und SAR-Bild. Original DOSAR-Bild (links), Überlagerung mit Simulation (mitte), Simulation des rekonstruierten Gebäudes (rechts)

In eng bebauten und komplexen urbanen Räumen müssen die Verdeckungen und Überlappungen durch die Umgebung in die Simulation einbezogen werden. Komplexe Stadtmodelle sollten in Echtzeit dargestellt werden können. Dann können durch die interaktive Anpassung der Gebäudeform und Höhe sehr schnell einfache Gebäudemodelle erstellt werden. Leider ist, insbesondere in Räumen mit vielen Verdeckungen und Überlappungen, oftmals keine eindeutige Rekonstruktion möglich. Meistens lassen sich aber, insbesondere wenn die Umgebung bekannt ist und in die Visualisierung mit einbezogen wird, die Gebäudeform und Höhe ausreichend genau bestimmen.

Noch besser lassen sich Veränderungen an Gebäuden feststellen. Sollte die Simulation eines Gebäudemodells stark von der tatsächlichen Erscheinung im SAR-Bild abweichen, so kann eine Veränderung angenommen werden. Der Auswerter muss beachten, dass nicht im Modell vorhandene Objekte, wie z.B. vor dem Gebäude stehende Bäume, die Form eines Objektes im SAR-Bild sehr verändern können. Große bauliche Veränderungen oder starke Beschädigungen lassen sich jedoch auch ohne Berücksichtigung dieser Objekte oft erkennen. Durch den Vergleich der Simulation von Gebäudemodellen mit dem realen SAR-Bild lassen sich bei freistehenden Objekten selbst geringe Unterschiede in Höhe und Form feststellen. In eng bebauten Räumen ist dies, auf Grund der Verdeckungen und Überlappungen, oft nicht hinreichend genau möglich. Hier lassen sich Beschädigungen oder Veränderungen nur feststellen, wenn diese hinreichend groß sind.

6.4 Automatische Objekterkennung

Objekte automatisch in Bilddaten zu erkennen und zuverlässig zu klassifizieren, ist seit langer Zeit ein Ziel zahlreicher Forschungsprojekte. Bislang ist jeder Algorithmus in punkto Zuverlässigkeit und Fehlerrate dem menschlichen Operator unterlegen, doch automatische Verfahren ermüden nicht und sind in der Lage große Datenmengen schnell zu verarbeiten. Zur Bearbeitung und Auswertung der immer größer werdenden Datenmengen sind automatische Verfahren daher immer wichtiger.

Die Qualität der automatischen Objekterkennung steht in einem direkten Zusammenhang mit der Qualität der Daten. Bei einem schlechten Signal-zu-Rauschverhältnis ist auch die Identifizierungsrate automatischer Verfahren schlecht. Die Fehlerrate hängt ebenfalls vom Bildinhalt ab. Ist der Bildinhalt einfach und sind die zu erkennenden Objekte klar von ihrer Umgebung zu unterscheiden, dann sind die Erkennungsraten automatischer Systeme hoch und ihre Fehlerrate ist niedrig.

Systeme zur automatischen Objekt- oder Zielerkennung arbeiteten in den frühen 80er Jahren häufig mit bestimmten Schwellwerten, um die Objekte vom Hintergrund unterscheiden zu können (Ratches et al, 1997). Eine Vielzahl dieser Verfahren zur Segmentierung des Bildinhaltes wurde bereits in den 80er Jahren entwickelt (Markham, 1989). Durch die Entwicklung von wissensbasierten Verfahren in den 90er Jahren verbesserte sich die Erkennungsrate der Systeme. Die Identifizierung basiert bei diesen Verfahren auf bereits vorhandenem Wissen über das Erscheinungsbild der zu erkennenden Objekte. Die im Bild segmentierten Objekte werden mit bekannten Objekten verglichen.

Bei diesen Methoden zur automatischen Objekterkennung kann die Echtzeitsimulation einen wichtigen Beitrag leisten. Anstatt die segmentierten Objekte mit Vergleichsmustern aus einer Datenbank zu vergleichen, können sie direkt mit Simulationsergebnissen verglichen werden. Die reduzierte Darstellungsgenauigkeit der in Echtzeit visualisierten SAR-Effekte spielt nur eine untergeordnete Rolle, da Simulationsergebnisse ohnehin nicht zu hundert Prozent mit der Realität übereinstimmen und dies bei den Verfahren berücksichtigt werden muss. Echtzeitsimulationsergebnisse können die Umgebung in ihren Vergleich mit einbeziehen und so auch Objekte bearbeiten, die teilweise verdeckt sind. Dafür muss allerdings das verdeckende Objekt bekannt und modellierbar sein.

Um den Geschwindigkeitsvorteil der Echtzeitvisualisierung bei der automatischen Objekterkennung besser ausnutzen zu können, sollten die Objekterkennungsverfahren ebenfalls auf der GPU implementiert sein. Dadurch wird der rechenzeitaufwendige Transfer der Daten vom Grafikkartenspeicher in den Hauptspeicher vermieden. Die Programmierbarkeit der GPU ist eingeschränkt. Verfahren zur automatischen Bilderkennung, die auf der GPU implementiert werden, müssen dies berücksichtigen.

Auf die Segmentierung der Daten wird im Weiteren nicht näher eingegangen. Eine Reihe von Bildverarbeitungsverfahren zur Bildsegmentierung wurden in der Vergangenheit entwickelt. Eine Übersicht über diese Verfahren geben z.B. *Morel und Solimini* (1995) oder *Pal und Pal* (1993). Bei den meisten Verfahren wird abhängig vom Bildinhalt eine Vielzahl von falschen Objekten segmentiert. Je nach gewünschter Anwendung kann durch den Einsatz von Geoinformationen die Anzahl der segmentierten Objekte stark reduziert werden. Sollen z.B. Flugzeuge klassifiziert werden, so sind nur die Objekte, die sich auf einem Flughafen befinden, zu segmentieren. Die Anzahl der zu segmentierenden Objekte verringert sich weiter, wenn nur noch Objekte, die sich auf den Verkehrswegen oder Abstellflächen befinden, betrachtet werden. Durch eine solche wissensbasierte Beschränkung der Segmentauswahl wird nicht nur die Zahl der zu segmentierenden Objekte gesenkt, der Bildinhalt wird deutlich einfacher, da Flugzeuge, insbesondere im SAR-Bild, sich stark vom Asphalthintergrund unterscheiden. Die Bildsegmentierung ist, obwohl sie hier nicht weiter betrachtet wird, keinesfalls trivial. Sie ist im Gegenteil eines der Hauptprobleme bei der automatischen Objekterkennung. Eine nähere Betrachtung der Segmentierung würde allerdings den Rahmen dieser Arbeit sprengen. Im Weiteren wird vorausgesetzt, dass interessante Bildsegmente entweder automatisch erkannt oder vom Anwender ausgewählt wurden.

Diese Bildausschnitte werden mit simulierten Bildern korreliert. Das simulierte Bild mit der höchsten Korrelation zum realen Bildausschnitt entspricht dann am wahrscheinlichsten dem realen Objekt. Nicht nur die Simulation, sondern auch die Berechnung der Korrelation erfolgt auf der Grafikhardware. Erstens ist die Grafikhardware gut geeignet zur Korrelationsberechnung und Zweitens ist der Kopiervorgang zwischen Grafikkartenspeicher und Hauptspeicher vergleichsweise zeitaufwendig, so dass es insgesamt deutlich schneller ist, die Berechnung auf der Grafikhardware durchzuführen. Die Größe des für die Bildkorrelation gewählten Fensters sollte fix sein. Zwar erlauben moderne Grafikkarten auch dynamische Befehle und somit auch Schleifen, doch ist die Performance deutlich schlechter. Aus dem Ergebnisbild der Korrelation soll ein Wert für die Gesamtkorrelation des Bildes ermittelt werden. Dies geschieht durch Reduktion.

In Abbildung 6-9 ist schematisch die Vorgehensweise bei der Reduktion abgebildet. Das ursprüngliche Bild wird in mehreren Schritten immer weiter reduziert, bis schließlich nur noch ein Wert übrig bleibt, in diesem

Fall das arithmetische Mittel. Bei jedem Schritt wird der Mittelwert in einem Fenster bestimmt und im Ergebnisbild gespeichert. Dieses Ergebnisbild dient dann im nächsten Durchlauf als Eingangsbild. Die Reduktion funktioniert am besten bei Bildern, die um ein Vielfaches des Fensters größer sind. Bei einer Fenstergröße von 2x2 sollte das zu reduzierende Bild also eine Größe von z.B. 128x128 oder 512x512 Pixel haben. Normalerweise erfolgt die Reduktion nicht bis zum letzten Pixel auf der Grafikhardware. Meist ist es schneller, das Bild bereits vorher, also z.B. bei einer Größe von 16x16 Pixel, in den Hauptspeicher zu kopieren und die restliche Berechnung dort vorzunehmen. Ansonsten sind für die Reduktion der letzten Pixel vergleichsweise viele Renderschritte nötig.

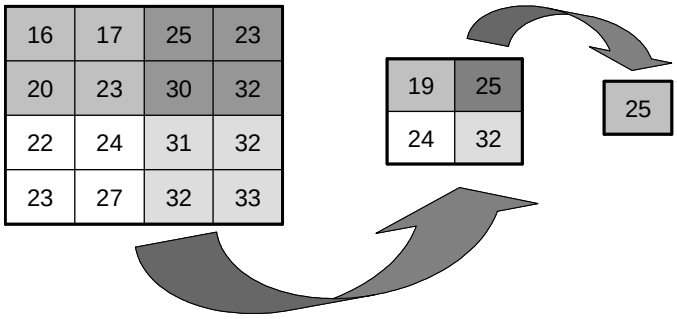

Abbildung 6-9: Reduktion zur Berechnung eines Mittelwertes auf Grafikhardware (nach Harris, 2005)

Da sowohl der Objekttyp als auch die Ausrichtung des Objektes unbekannt sind, müssen auf diese Art sehr viele Simulationen und Korrelationen berechnet werden. Pro Simulation und anschließender Korrelation werden, in Abhängigkeit von der Größe des Bildausschnittes, ungefähr 25ms benötigt. Daher ist die Zahl der durchzuführenden Vergleiche pro unbekanntem Objekt möglichst zu reduzieren. Wenn man bei den Simulationen das Objektmodell in 5° Schritten dreht, so sind pro Objektmodell 72 Simulationen und Korrelationen mit einer Gesamtdauer von 1800ms nötig. Wenn die Objektdatenbank beispielsweise aus 100 Objekten besteht, würde es dann pro unbekanntem Bildsegment 180 Sekunden dauern, alle Möglichkeiten zu testen. Durch die Einbeziehung zusätzlicher Informationen kann die benötigte Zeit weiter reduziert werden. Auf Landflächen müssen z.B. keine Schiffsmodelle in die Untersuchung miteinbezogen werden. Auf Flugplätzen kann sich die Untersuchung auf Flugzeuge und evtl. Fahrzeuge beschränken. Die Frage, ob sich das zu untersuchende Objekt auf einem Flugplatz befindet, kann durch die Einbeziehung von Geodaten beantwortet werden.

Durch eine Eingrenzung der möglichen Objektausrichtungen kann die Zahl der zu berechnenden Korrelationen weiter stark reduziert werden. In vielen Fällen kann eine Objektausrichtung entweder manuell oder durch Methoden der automatischen Bildverarbeitung bestimmt werden. Schiffe sind prinzipiell länger als breit. Untersucht man nun das Bildsegment hinsichtlich Länge und Breite und der Ausrichtung der langen Seite, so können die zu simulierenden Objektausrichtungen stark eingegrenzt werden. Anstatt 72 unterschiedlicher Richtungen sind dann nur noch zwei Richtungen zu untersuchen. Selbst wenn man eine gewisse Unsicherheit annimmt und von den zwei Richtungen jeweils noch eine Untersuchung 5° links und 5° rechts durchführt, so bleiben trotzdem nur noch sechs zu berechnende Korrelationen übrig.

Ähnliches gilt für Fahrzeuge und Flugzeuge, wobei Flugzeuge vier mögliche Richtungen haben, da sich im SAR-Bild die Tragflächen kaum vom Flugzeugrumpf unterscheiden lassen. Inklusive Unsicherheiten blieben dann noch 12 durchzuführende Korrelationen. Geht man von 50 Flugzeugtypen in der Objektdatenbank aus, lassen sich die nötigen 600 Simulationen und Korrelationen in 15 Sekunden durchführen.

Ein weiteres Ausschlusskriterium ist die Objektgröße. Die Länge und Breite der zu simulierenden Objekte ist bekannt und es macht keinen Sinn, einen Flugzeugträger mit einem Objekt von nur wenigen Metern Durchmesser zu vergleichen. Ein Vergleich der Größen erlaubt eine weitere drastische Reduzierung der zu vergleichenden Objekte. Allerdings ist die Messung der Objektgröße in realen SAR-Bildern vergleichsweise schwierig, da ein einzelnes Objekt sich im SAR-Bild aus verschiedenen Bereichen starker Reflexion zusammensetzen kann. Diese einzelnen Bereiche starker Reflexionen sind voneinander getrennte Bildsegmente. Jedes Segment einzeln zu detektieren und einzugrenzen ist einfach. Festzulegen, welche Segmente von ein und demselben Objekt reflektiert werden, ist dagegen schwierig.

Abbildung 6-10: Flugzeuge im SAR-Bild setzen sich aus mehreren Reflexionszentren zusammen. Bild: DOSAR (EADS Deutschland)

In Abbildung 6-10 ist ein Flugzeug im Parkbereich eines Flughafens abgebildet. Das Flugzeug setzt sich aus mehreren hellen Bildbereichen zusammen. Im Falle dieses einzeln stehenden Objektes wäre die Zuordnung der einzelnen Segmente zu einem Objekt vergleichsweise einfach, doch befinden sich meist neben dem Flugzeug weitere Objekte, wie z.B. Tankfahrzeuge. Deren einzelne Rückstreuzentren würden, insbesondere von automatischen Verfahren, als zu dem Objekt zugehörig betrachtet werden. Die Vielzahl möglicher Probleme, z.B. durch Flugzeuge die eng nebeneinander geparkt wurden, lässt die Grenzen automatischer Verfahren erahnen.

Nicht nur die zu einem Objekt zugehörigen Bildsegmente sind ein Problem, sondern auch der unbekannte Standpunkt des realen Objektes. Durch den Layovereffekt ist der genaue Standpunkt des Objekts im Bild nicht exakt zu ermitteln. Die Positionen der Bildsegmente sind bekannt und der Mittelpunkt des Objekts im SAR-Bild lässt sich, eine erfolgreiche Bildsegmentzuordnung vorausgesetzt, anhand der Bildsegmente berechnen. Unbekannt ist allerdings die genaue reale Objektposition, denn die Positionen der Reflexionen hängen von der Höhe des Reflexionszentrums ab.

Der Mittelpunkt kann als unscharf angenommen werden und kann z.B. durch eine Vielzahl weiterer Simulationen und Korrelationen bestimmt werden. Aber bereits eine Unschärfe von drei Pixeln in jede Richtung führt zur 49-fachen Menge an benötigten Simulationen. Da die Berechnung des wahren Objektpunktes vom Objekt abhängt, könnten die vorhandenen Objektmodelle benutzt werden, um einen möglichen Objektmittelpunkt im Bild in Abhängigkeit des Objektmodells zu bestimmen. Diese Vorgehensweise ist jedoch nicht trivial und erfordert komplizierte Berechnungen für jedes zu testende Objektmodell, einschließlich der Gefahr von Fehlern und Mehrdeutigkeiten.

Die automatische Objekterkennung in SAR-Bildern ist sehr problematisch. Echtzeit-SAR-Simulationen können einen Beitrag zur Objekterkennung leisten. Die schnelle Simulation und Korrelation der simulierten Modelle mit dem realen Bild erlauben es, eine Vielzahl möglicher Optionen zu testen. Die schiere Menge der Optionen bei der automatischen Auswertung realer SAR-Bilder erfordert allerdings eine Vorauswahl. Dies kann durch einen Operator oder durch vorgeschaltete, automatische Verfahren geschehen. Automatische Verfahren sind jedoch ungenau und fehleranfällig. Ein sinnvoller Einsatz bei der automatischen Objekterkennung hängt daher stark von den Fähigkeiten und Einschränkungen dieser Verfahren ab.

Neben dem Einsatz von Simulatoren zur automatischen Objekterkennung, profitiert auch die automatische Gebäuderekonstruktion von der SAR-Simulation in Echtzeit. Nach *Stilla & Sörgel* (2006) kann unter Ausnutzung eines Hypothesen-Simulations-Verifikations-Schemas eine automatisierte Gebäuderekonstruktion realisiert werden. Dabei werden iterativ Gebäudehypothesen SAR-simuliert. Dadurch werden die Layoverbereiche, die Schatten und Corner-Reflektoren ermittelt. Zusätzlich wird ein interferometrisches Höhenmodell simuliert. Diese simulierten Daten werden mit den realen Daten verglichen. Das Ergebnis dieses Vergleiches führt dann zu neuen Gebäudehypothesen, die dann ihrerseits wieder simuliert und verifiziert werden.

6.5 Automatische Georeferenzierung

Vergleiche zwischen Simulation und SAR-Bild können zur automatischen Georeferenzierung von SAR-Daten verwendet werden. SAR-Bilder sind normalerweise bereits vorab georeferenziert, allerdings oft nicht ausreichend genau. Um die Georeferenzierung zu verbessern, kann beispielsweise auf simulierte Straßendaten oder auf simulierte 3D-Stadtmodelle zurückgegriffen werden (Balz, 2006a). Diese automatische Methode profitiert von der schnellen Verfügbarkeit von SAR-Simulationsergebnissen. Auf Grund der bei den Vergleichsoperationen zwischen Simulation und realem SAR-Bild vorgenommenen Vereinfachungen und Generalisierungen ist die physikalische Korrektheit der Simulation zweitrangig. Die Simulation muss der Realität nur Nahe kommen.

Für die Georefenrenzierung werden korrespondierende Punkte zwischen dem SAR-Bild und einem Vektordatensatz gesucht. Soll das SAR-Bild, z.B. auf Grundlage von Straßendaten, georeferenziert werden, so wird in den Straßendaten nach markanten Bereichen gesucht. Markant ist ein Bereich, wenn er viele Kreuzungen enthält. Durch einfache GIS-Abfragen lassen sich schnell markante Bereiche mit vielen Kreuzungen finden. Allerdings sind diese Bereiche selten gleichmäßig über das Bild verteilt. Für eine erfolgreiche Georeferenzierung ist jedoch eine möglichst gleichmäßige Verteilung der Referenzpunkte über das Bild notwendig. Für das Verfahren zur Auswahl der Bildregionen wird das Bild in mehrere Subregionen unterteilt und in jeder Subregion müssen sich mindestens eine, besser zwei, Referenzregionen befinden. Im Beispiel aus der Region Karlsruhe, in Abbildung 6-11, ist die Lage des zu referenzierenden SAR-Bildes als graue Fläche dargestellt. Die Straßenvektoren sind als schwarze Linien und die ausgewählten Referenzregionen als schwarze Punkte dargestellt.

Abbildung 6-11: Auswahl von Referenzregionen

Die Regionen sind nicht gleichmäßig verteilt, sondern konzentrieren sich auf das Stadtgebiet im linken Bilddrittel. In den weniger urban geprägten Räumen in der Bildmitte und im rechten Bildbereich finden sich weniger Kreuzungen und daher weniger geeignete Regionen. Die gewählten Regionen liegen, auf Grund der geringen Streifenbreite der Szene, alle nahezu auf einer Linie. Dies erschwert die exakte Georeferenzierung.

Die ausgewählten Regionen werden anschließend simuliert. Im weiteren Verlauf wird angenommen, dass die Straßen nur schwach reflektieren und die umgebenden Bereiche stärker. Das Bild wird in ein binäres Bild, bestehend aus den Klassen *Straße* und *Sonstiges*, klassifiziert. Für jedes Pixel im Suchbereich wird dann der Suchwert *v* berechnet. Unter der Annahme, dass die Straßen dunkel, die anderen Bereiche jedoch hell sind, wird die Summe aller Pixelwerte der Klasse *Straße* g_{str} durch die Summe aller Pixelwerte der Klasse *Sonstiges* $g_{sonstiges}$ geteilt. Das Pixel mit dem kleinsten Wert *v* entspricht der gesuchten Pixelposition.

$$v = \frac{\sum_{x=1}^{x}\sum_{y=1}^{y}\left[g_{str}(x,y)\right]}{\sum_{x=1}^{x}\sum_{y=1}^{y}\left[g_{sonstiges}(x,y)\right]} \tag{5.1}$$

Im Umkreis jeder Referenzregion wird nach dem kleinsten Wert für *v* gesucht. Die Größe des Suchumkreises richtet sich nach dem angenommenen Fehler in der vorhergehenden Referenzierung des SAR-Bildes. Auf Grund des hohen Rechenaufwandes sollte der ursprüngliche Positionsfehler, und damit der Suchbereich, nicht allzu groß sein. Um Rechenzeit zu sparen, wird die Suche zunächst auf einer geringeren Auflösung durchgeführt und erst nachträglich verfeinert. Der pro Suchbereich gefundene Referenzpunkt geht dann in

eine Ausgleichungsrechung ein. Die Ausgleichung sollte überbestimmt sein, um fehlerhafte Referenzpunkte auszuschließen.

Die Zahl möglicher Fehlerquellen ist groß. Die Suche nach einem korrespondierenden Punkt kann durch mangelnden Kontrast zwischen Straße und Umgebung, sowie durch eine zu geringe Anzahl von Kreuzungen bzw. durch mehrdeutige Situationen, beeinträchtigt werden. Abbildung 6-12 zeigt zwei Beispiele für eine fehlerhafte Bildzuordnung. Die Straßen aus dem Referenzdatensatz sind weiß abgebildet, um die Zuordnung zu erkennen. In beiden Fällen ist der Kontrast zwischen Straße und Umgebung gering. Im linken Fall stört der Schattenwurf der Baumreihe in der Bildmitte die Auswertung zusätzlich.

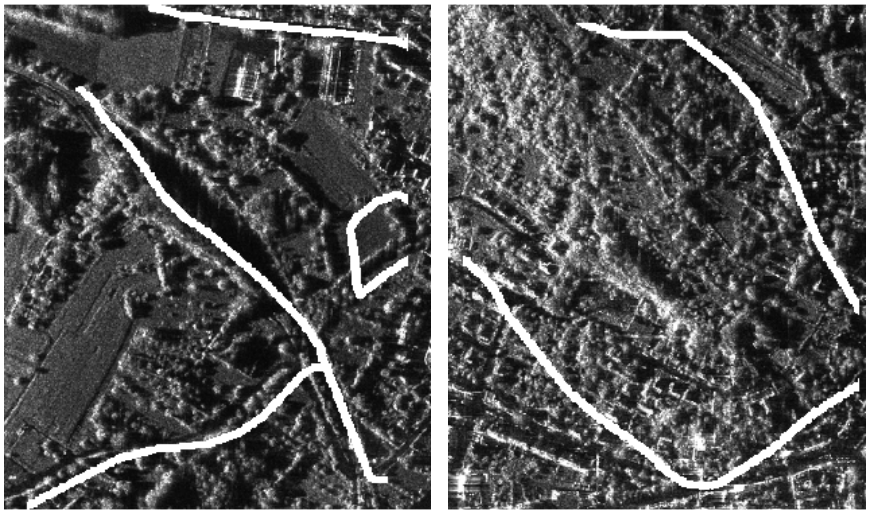

Abbildung 6-12: Beispiele für eine fehlerhafte Bildzuordnung

Abbildung 6-13: Beispiele für eine richtige Bildzuordnung

In Abbildung 6-13 sind richtige Bildzuordnungen abgebildet. Bei ausreichend hoher Kreuzungsdichte, bzw. bei einem eindeutigen und dichten Straßennetz und hohem Kontrast kommt es zu einer richtigen Bildzuordnung. Hoher Kontrast wird vor allem in urbanen Räumen erreicht, für die das Verfahren konzipiert wurde. Für gleichmäßige, schachbrettartige Stadtstrukturen ist das Verfahren auf Grund der auftretenden Mehrdeutigkeiten jedoch nicht geeignet.

Gerade in urbanen Räumen tritt ein weiteres Problem auf: die Zuordnung wird in Richtung des Sensors verschoben. Diese Verschiebung ist bedingt durch die umliegenden Häuser und hängt in ihrem Ausmaß von der Gebäudehöhe ab. Die Verschiebung ist schematisch in Abbildung 6-14 dargestellt. Bei der Suche nach korrespondierenden Punkten zwischen Straßendaten und SAR-Bild wird davon ausgegangen, dass die Straßen im Bild dunkel abgebildet werden. Schattenbereiche sind allerdings ebenfalls dunkel und kaum von Straßen zu unterscheiden, so dass sich im SAR-Bild Straßen und Gebäudeschatten vermischen. Der Layoverbereich

von den Häusern in Far-Range überdeckt die Straße, so dass dieser Bereich im SAR-Bild hell erscheint. Sind die Gebäude in Near- und Far-Range der Straße gleich hoch, so bleibt die abgebildete Straßenbreite erhalten, allerdings „verschiebt" sich die abgebildete Straße hin zum Sensor. Tatsächlich wird der nur von der Straße zurückreflektierte und nicht durch das Layover beeinflusste Bereich kleiner. Dieser Bereich vermischt sich aber mit dem Schattenbereich, so dass der Eindruck entsteht, die Straße würde sich verschieben.

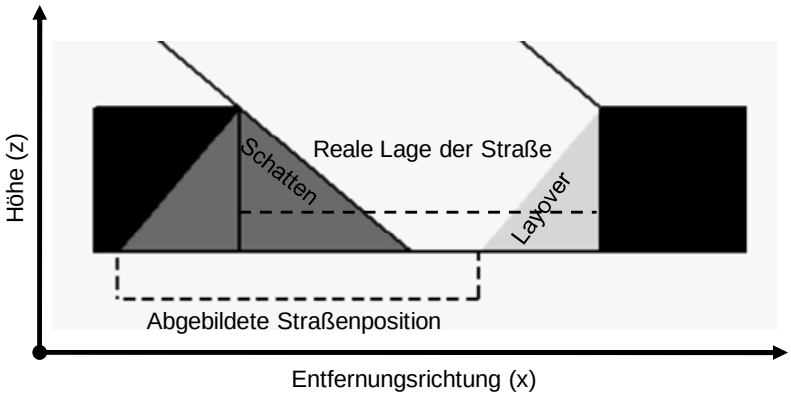

Abbildung 6-14: Verschiebung der abgebildeten Straßenposition durch Schatten und Layover

Für die automatische Georeferenzierung ist dies ein Problem, da das Ausmaß der Verschiebung von der Höhe der umliegenden Gebäude und dem Inzidenzwinkel abhängt. Die Höhe der Gebäude ist jedoch im Normalfall unbekannt, so dass das Ausmaß der Verschiebung nicht bestimmt werden kann. Da bei dem gezeigten Verfahren insbesondere die Positionierung in Entfernungsrichtung exakt sein sollte, um der ungünstigen Verteilung der Referenzpunkte wie in Abbildung 6-11 dargestellt entgegenzuwirken, ist diese Verschiebung in Entfernungsrichtung ein großes Problem.

Trotz dieser Probleme lässt sich die Georeferenzierung eines SAR-Bildes durch diese Methode deutlich verbessern. Mittels der in Autonavigationssystemen verbreiteten GDF-Straßendaten, die selbst nur eine horizontale Genauigkeit von ±3 Metern haben (Walter, 1997), lässt sich im obigen Beispiel von Karlsruhe, die räumliche Genauigkeit der SAR-Daten, die ursprünglich um ca. 150m verschoben waren, auf ±6m bis ±10m verbessern. Auf Grund der Vielzahl der oben beschriebenen Probleme ist die Methode kaum allgemein gültig und nur bedingt vollautomatisch anwendbar.

Durch die Simulation von 3D-Stadtmodellen und dem automatischen pixelweisen Vergleich zwischen Simulation und realem SAR-Bild in einem definierten Suchraum lässt sich, die Verfügbarkeit aktueller 3D-Stadtmodelle vorausgesetzt, die Genauigkeit weiter erhöhen. Dazu werden anstatt Straßendaten 3D-Gebäudemodelle simuliert. Die simulierten Bilder werden mit dem realen SAR-Bild verglichen, um korrespondierende Punkte zu finden. Die Lagegenauigkeit der 3D-Stadtmodelle ist normalerweise höher als die der GDF-Straßendaten. Des Weiteren kann durch die Einbeziehung der dritten Dimension die Genauigkeit verbessert werden.

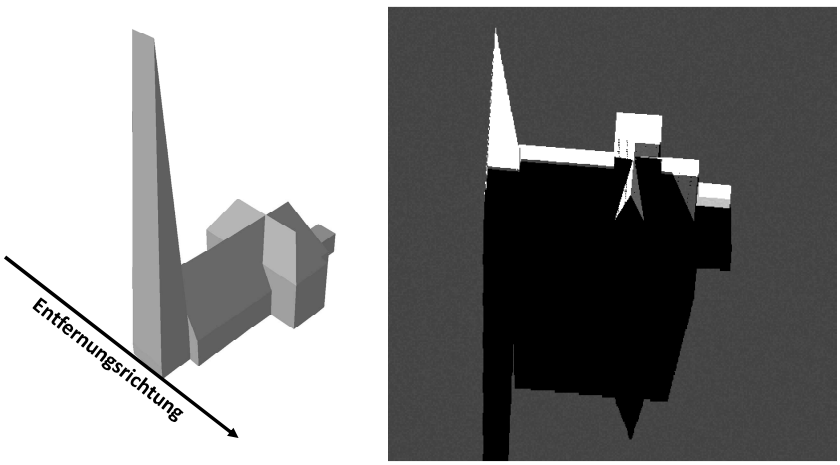

Abbildung 6-15: 3D-Modell einer Kirche (links) und klassifiziertes Simulationsergebnis (rechts)

Das simulierte Bild wird in drei Klassen unterteilt: Schatten, starke Reflektion und Sonstiges. Es wird dann das Pixel mit der höchsten Korrelation zwischen simuliertem Bild und realem SAR-Bild gesucht, wobei nur die Schattenbereiche und die starken Reflektionen korreliert werden. Um korrespondierende Punkte ausgewählter Gebäude zu ermitteln, muss dieses Alleinstellungsmerkmale haben, um nicht mit benachbarten Gebäuden verwechselt zu werden. Außerdem sollte das Gebäude möglichst freistehend sein, um Überlappungen und Verdeckungen weitestgehend zu vermeiden. In Abbildung 6-15 ist am Beispiel eines aus LIDAR-Daten rekonstruierten Modells der St.-Bernharduskirche in Karlsruhe das klassifizierte Simulationsergebnis dargestellt.

Die pixelweise durchgeführte Korrelation ist sehr rechenzeitaufwendig. Es gelingt dadurch jedoch die Genauigkeit der Georeferenzierung auf ±2 m zu verbessern (Balz, 2006a), was ungefähr dem Vierfachen der 3dB-Auflösung entspricht. Der Prozess lässt sich durch eine Berechnung auf der Grafikkarte beschleunigen. Für eine Suche in einem 40x40 Pixel großen Bildraum benötigt eine NVIDIA GeForce 6600 Grafikkarte ungefähr 2,7 Sekunden (Balz, 2006a). 40x40 Pixel entsprechen in diesem Beispiel einem Suchfenster von ca. 10m x 10m.

Für die automatische Georeferenzierung von SAR-Daten ist ein zweistufiges Vorgehen empfehlenswert. Zunächst wird die bereits vorhandene, aber sehr ungenaue Georeferenzierung durch einen automatischen Abgleich mit SAR-simulierten GDF-Straßenvektoren verbessert. In urbanen Gebieten kann eine Genauigkeit von ca. ±10 Metern erreicht werden. Diese Georeferenzierung kann dann durch den Vergleich zwischen simulierten 3D-Gebäudemodellen und dem realen SAR-Bild weiter verbessert werden.

7 Zusammenfassung und Ausblick

Die SAR-Datengewinnung ist tageszeit- und wetterunabhängig. Daher sind SAR-Daten von besonderem Interesse für Aufgaben, welche eine dauernde Überwachung eines Gebietes oder die Aufklärung zu einem bestimmten Zeitpunkt erfordern. Dies ist beispielsweise bei militärischen Anwendungen oder im Katastrophenmanagement notwendig. Moderne SAR-Systeme erreichen eine hohe geometrische Auflösung, allerdings sind die Daten oft nur schwer zu interpretieren. Die SAR-Simulation unterstützt die Entwicklung der SAR-Technik und die Anwendung von SAR-Daten seit langer Zeit. SAR-Simulatoren sind für die Entwicklung neuer Sensoren und Prozessierungsmethoden nahezu unverzichtbar. Für einige Anwendungen der SAR-Simulation sind die vorhandenen SAR-Simulatoren jedoch zu langsam. Die SAR-Effektvisualisierung in Echtzeit bietet neue Möglichkeiten.

Die Nutzung der sich rapide entwickelnden Grafikhardware zur Visualisierung von SAR-Effekten ist ein logischer Schritt zur Steigerung der Simulationsgeschwindigkeit. Im Gegensatz zu frühen Versionen von Microsoft® Direct3D, lässt sich, seit Version 8, direkt Einfluss auf die von der Grafikkarte durchgeführten Berechnungen nehmen. Dadurch können z.B. die Beleuchtungen individuell berechnet werden, wodurch die Visualisierungen flexibler und realistischer werden. Diese freie Programmierbarkeit ermöglicht nicht nur die Verbesserung der Visualisierung virtueller Szenen, sondern auch die Nutzung der Grafikkarten für andere rechenintensive Anwendungen. Eine dieser Anwendungen ist die SAR-Effektvisualisierung. Die flexible Programmierbarkeit der Grafikhardware erlaubt die Berechnung der SAR-Geometrie und -Radiometrie und damit die Echtzeitvisualisierung von SAR-Effekten auf handelsüblichen Grafikkarten. Die Berechnung und die Darstellung der SAR-Effekte basiert dabei auf Technologien und Methoden, die ansonsten bei der Programmierung von Visualisierungsanwendungen, wie z.B. Computerspielen, zum Einsatz kommen. Die größte Herausforderung stellen die geometrischen Unterschiede dar. SAR-Daten unterliegen in Entfernungsrichtung der Laufzeitgeometrie. In Flugrichtung wird die Geometrie durch den Dopplereffekt bestimmt. Für die SAR-Effektvisualisierung muss deswegen die Bildgeometrie angepasst werden.

Dies erfolgt, wie in Kapitel 4 beschrieben, durch eine Änderung der Transformationsmatrizen auf der Grafikkarte. Die Sichtbarkeits- und Schattenwurfanalyse muss eigenständig erfolgen. Sie basiert auf dem Shadow-Map-Verfahren. Neben der Geometrie muss auch die Radiometrie angepasst werden. Drei Verfahren zur Berechnung des Rückstreukoeffizienten σ° wurden implementiert. Die drei Verfahren stehen für unterschiedliche Methoden und repräsentieren, sowohl die auf Messungen und statistische Auswertung beruhende Simulation als auch die physikalische Modellierung der Rückstreuung. Dadurch wird die hohe Flexibilität bei der Berechnung der Rückstreuung auf der programmierbaren Grafikhardware demonstriert. Die unterschiedlichsten Methoden und Modelle können implementiert werden. Die Addition der Rückstreuung gleicher Laufzeit lässt sich über das additive Alpha-Blending einfach realisieren. Durch die Verwendung von Texturen lassen sich auch komplexe Modelle einfach und schnell visualisieren. Die Verwendung von weichen Schatten und die dynamische Anpassung der Beleuchtungsrichtung im Spotlightmodus zeigen die vielfältigen Möglichkeiten durch die dynamische Prozessierung der Daten auf der Grafikhardware. Die Fähigkeit, die Eigenbewegungen von Objekten in die Prozessierung mit einzubeziehen, ist zwar nur rudimentär, da z.B. Beschleunigungen fehlen, zeigt jedoch das hohe Potential und die enormen Erweiterungsmöglichkeiten der SAR-Effektvisualisierung.

Die in Kapitel 5 gezeigten Visualisierungen umfangreicher Stadt- und anderer 3D-Modelle beweisen die hohe Leistungsfähigkeit des Ansatzes. Durch den Vergleich mit anderen Simulatoren wird die erreichbare Qualität deutlich. Bei Einfachreflexionen, wie sie vor allem bei natürlichen Objekten auftreten, sind die Simulationsergebnisse vergleichbar mit den Resultaten herkömmlicher und kommerzieller Simulatoren. Es können weit umfangreichere und komplexere Modelle verarbeitet werden, als dies bei Ray-Tracing-Simulatoren in akzeptabler Zeit möglich ist. Die Fähigkeit Modelle mit Millionen von Dreiecken in Echtzeit zu simulieren ist insbesondere für urbane Anwendungen ein entscheidender Vorteil. Trotzdem bleiben einige Probleme ungelöst. So kann die Zweifachreflexion zwar durch die Verwendung von Environmental-Maps teilweise visualisiert werden, doch bleiben die Ergebnisse unbefriedigend und die Darstellung der Mehrfachreflexionen weiterhin unmöglich.

Einige Anwendungen für die Echtzeitsimulation von SAR-Daten wurden in Kapitel 6 aufgeführt. Allen diesen Anwendungen ist gemein, dass die Genauigkeit des Simulationsergebnisses eine untergeordnete Rolle spielt. Eine Fokussierung auf hochgradig physikalisch exakte Simulationsergebnisse ist für viele Anwendungen irrelevant. Stattdessen legen diese vielmehr Wert auf eine hohe Geschwindigkeit. Hochpräzise Simulationsergebnisse verlangen hochgenaue Modelle und Eingangsdaten, die oftmals nicht vorhanden sind. Eine hochpräzise Simulation mit ungenauen Daten führt zu ungenauen Simulationsergebnissen. Die Erzeugung hochgenauer Daten, welche eine präzise Simulation ermöglichen, ist zumeist nur in ausgewählten Testgebieten möglich und in der praktischen Anwendung meist undenkbar. In Anbetracht der meist dürftigen Datenlage ist der Glaube an Simulationsergebnisse, die exakt die realen Ergebnisse widerspiegeln, ein Irrglaube. Eine Simulation in Echtzeit mit eingeschränktem Realismus ist daher für viele Anwendungen nützlicher. Die Simulierbarkeit umfangreicher Szenen mit hochkomplexen Modellen ist vor allem für Anwendungen mit hochauflösenden Daten im urbanen Raum notwendig. Komplexe und umfangreiche Szenen können schnell mehrere Zehnmillionen Dreiecke umfassen. Diese Datenmenge in absehbarer Zeit zu verarbeiten, stellt viele Simulatoren vor eine große Herausforderung. Der Einsatz von Grafikhardware für die SAR-Simulation ist ein neuer Ansatz, der dies ermöglicht.

Die Anwendung der Rechenleistung von Grafikkarten für verschiedene Anwendungen wird immer bedeutender. Einmal mehr ist die Spieleindustrie Motor dieser Entwicklung. Neben der realistischen Visualisierung der Spiellandschaften wird eine realistische Spielphysik immer gefragter. Eine Reihe von Hardware- und Softwarelösungen werden momentan hierfür angeboten. Eine eigene Hardware zur Berechnung der Spielphysik bietet die Firma AGEIA an. „AGEIA PhysX“ ist eine Hardwareerweiterung für PCs mit einem spezialisierten Prozessor. Andere Anbieter berechnen die Physik auf der Grafikkarte, wodurch keine spezielle Physikhardware nötig ist. So bietet die Firma Havok mit „Havok FX“ die Möglichkeit an, die Flugbahn von Partikeln auf der Grafikkarte zu berechnen. NVIDIA (2007) bietet mit CUDA eine eigene Bibliothek zur Berechnung von nicht-grafischen Problemen auf der Grafikkarte. CUDA eignet sich für die Berechnung der Spielphysik, aber auch für andere Anwendungen. Durch die Möglichkeit, zwei Grafikkarten zusammenzuschließen, erhöht sich die mögliche Leistung enorm. Eine Grafikkarte kann für die Visualisierung, eine zweite zur Berechnung der Physik verwendet werden.

Für die SAR-Simulation bieten diese Technologien neue Möglichkeiten. Die Berechnung der Mehrfachreflexionen mittels Physikhardware oder Physikbibliotheken könnte die Qualität der Echtzeitsimulation weiter verbessern. Neben den Mehrfachreflexionen lassen sich diese Physikbibliotheken auch zur Berechnung atmosphärischer Effekte einsetzen. Auch eine verbesserte Berechnung der Rückstreuung und der Volumenreflexionen sind denkbar. Eine weit realistischere Echtzeitsimulation wird dadurch in Zukunft möglich sein.

Anhang A – Abbildende Radarsysteme

A.1 Geometrische Auflösung des Radars mit realen Apertur (RAR)

Die Winkelauflösung der realen Apertur in Azimut φ_{ra} hängt von der Wellenlänge λ und der Länge der realen Apertur l_{ra} ab. Typische Wellenlängenbereiche λ in der Radarfernerkundung liegen zwischen 60cm (P-Band) und 1cm (Ka-Band).

$$\varphi_{ra} \approx \frac{\lambda}{l_{ra}} \tag{A.1}$$

Die tatsächliche Auflösung in Azimut berechnet sich durch Multiplikation der Winkelauflösung mit der Entfernung zum Objekt r_O.

$$\delta_{raa} = r_O \cdot \varphi_{ra} \tag{A.2}$$

In Entfernungsrichtung, auch Rangerichtung genannt, strahlt das Radarsystem eine schnelle Folge einzelner Radarimpulse aus. Diese werden am Boden reflektiert und erreichen nach einer bestimmten Zeit Δt wieder den Sensor. Δt hängt von der Entfernung zwischen Sensor und Objekt ab.

$$\Delta t = \frac{2 r_O}{c} \tag{A.3}$$

Die erreichbare Auflösung in Slant-Range δ_r hängt daher von der Pulslänge τ des Radarstrahls ab. Die Bodenauflösung in Ground-Range hängt zusätzlich noch vom Blickwinkel θ_{off} ab.

$$\begin{aligned} \delta_r &= \frac{c \cdot \tau}{2} = \frac{c}{2B_r} \\ \delta_{rg} &= \frac{c \cdot \tau}{2 \cdot \sin \theta_{off}} \end{aligned} \tag{A.4}$$

Soll beispielsweise eine Aufnahme mit 2m Auflösung in Azimut- und Entfernungsrichtung im X-Band aus einem Flugzeug in sechs Kilometer Entfernung erfolgen, so muss das System eine Bandbreite von ca. 75 MHz haben, um die gewünschte Auflösung in Slant-Range zu erhalten. Zusätzlich muss die reale Apertur, um die gewünschte Auflösung in Azimut erreichen zu können, eine Länge von 90 Metern haben. Dies ist technisch kaum zu realisieren.

A.2 Geometrische Auflösung des Radars mit synthetischer Apertur (SAR)

Unter Ausnutzung des Dopplereffektes kann die Azimutauflösung eines Radars durch die synthetische Apertur verbessert werden. Die Länge der synthetischen Apertur l_{sa} entspricht dabei der Strecke, welche die Sensorplattform zwischen dem ersten und letzten Signal, die ein Ziel in Entfernung r_O treffen, zurücklegt.

$$l_{sa} = \varphi_{ra} \cdot r_O = \frac{\lambda}{l_{ra}} \cdot r_0 \tag{A.5}$$

Die Länge der synthetischen Apertur l_{sa} entspricht der Azimutauflösung der realen Apertur δ_{raa}. Die Winkelauflösung eines SAR in Azimut φ_{sa} berechnet sich analog zu der Gleichung (A.1), wobei zu beachten ist, dass die erfolgte Zusammenfassung der Einzelimpulse zu einer etwa halb so schmalen Antennenkeule führt und daher für die Winkelauflösung φ_{sa} gilt:

$$\varphi_{sa} \approx \frac{\lambda}{2 \cdot l_{sa}} \tag{A.6}$$

Somit ergibt sich:

$$\varphi_{sa} \approx \frac{\lambda \cdot l_{ra}}{2 \cdot \lambda \cdot r_O} \approx \frac{l_{ra}}{2 \cdot r_O} \tag{A.7}$$

Die Bodenauflösung wird analog zu (A.2) berechnet:

$$\delta_{sa} = \varphi_{sa} \cdot r_O = \frac{l_{ra}}{2} \tag{A.8}$$

Die erreichbare Auflösung eines SAR ist von der Entfernung vom Sensor zum Ziel unabhängig. Wie aus (A.8) ersichtlich, ist die Länge der realen Antenne auch beim Radar mit synthetischer Apertur eine wichtige Größe, allerdings umgekehrt proportional zu Radarsystemen mit realer Apertur. Je kleiner die erreichbare Bodenauflösung sein soll, desto kleiner muss die reale Apertur sein. Dies führt zu einem Zielkonflikt, da eine kleinere Antenne ein schlechteres Signal-zu-Rauschverhältnis aufweist.

A.3 Die Radargleichung

Die Radargleichung beschreibt die Beziehung zwischen dem Radar, dem Ziel und dem empfangenen Signal (Ulaby et al, 1982). Wenn das Radar mit der Leistung P_S in Richtung des Ziels ausstrahlt und die ausgestrahlte Leistung um den Antennengewinn *G* erhöht, dann ist die Sendeleistungsdichte S_T am Ziel in Entfernung r_O :

$$S_T = \frac{P_S \cdot G}{4\pi \cdot r_O^{\,2}} \tag{A.9}$$

Die von einem Punktziel zurückgestreute Leistung P_{RS} ist abhängig vom Radarrückstreuquerschnitt σ. Dieser beschreibt das Reflexionsverhalten des Ziels und ist definiert als das Verhältnis der pro Einheitsraumwinkel vom Ziel zum Empfänger gestreuten Leistung zu der auf das Ziel einfallenden Flächenleistungsdichte (Klausing & Holpp, 2000). Es gilt:

$$P_{RS} = S_T \cdot \sigma \tag{A.10}$$

Ein Teil der am Ziel auftreffenden Energie wird absorbiert, der Rest wird wieder in verschiedene Richtung reflektiert. Die Stärke der am Ziel, diesmal der Antenne, ankommenden Energie kann mit der Gleichung (A.9) beschrieben werden. Die an der Antenne empfangene Leistung kann mit der Gleichung (A.10) berechnet werden, wobei σ in diesem Fall für die effektive Antennenfläche A_{eff} steht. Sie hängt von der verwendeten Wellenlänge λ und dem Antennengewinn *G* ab.

$$A_{eff} = \frac{G \cdot \lambda^2}{4\pi} \tag{A.11}$$

Davon ausgehend, dass der Antennengewinn, die effektive Antennenfläche und die Entfernung zwischen Radar und Ziel, sowohl beim Senden als auch beim Empfang gleich bleiben, kann man die am Radar empfangene Energie P_R berechnen aus:

$$P_R = \frac{P_S \cdot G^2 \cdot \lambda^2}{(4\pi)^3 \cdot r_O^{\,4}} \cdot \sigma \tag{A.12}$$

Systemverluste und Atmosphärendämpfung finden hier keine Berücksichtigung. Die empfangene Leistung fällt mit der vierten Potenz des Abstandes. Die Möglichkeit, schwach empfangene Signale zu detektieren hängt von der Rauschleistung im Empfänger ab. Kennt man das zur Zielerfassung mindestens notwendige Signal-zu-Rauschverhältnis $SNR_{\min}$, dann kann man die maximale erzielbare Reichweite $r_{\max}$ berechnen:

$$r_{\max} = \sqrt[4]{\frac{P_S \cdot G^2 \cdot \lambda^2 \cdot \sigma \cdot \tau_p}{(4\pi)^3 \cdot k \cdot T_{eff} \cdot SNR_{\min} \cdot L_{ges}}} \tag{A.13}$$

Dabei steht *k* für die Boltzmann-Konstante, T_{eff} für die effektive Rauschtemperatur, L_{ges} für den Gesamtverlust des Systems und τ_p für die Dauer des rechteckförmigen Sendeimpulses (Klausing & Holpp, 2000).

In (A.12) hängen alle Variablen außer σ und r_O vom Sensor ab. Für die Bildsimulation ist daher die Berechnung des Rückstreuquerschnitts σ entscheidend, da die Berechnung von r_O trivial ist und die restlichen Variablen vom Sensor abhängen und über das Bild identisch bleiben. Der Rückstreuquerschnitts σ ist definiert als:

$$\sigma = 4\pi {r_O}^2 \cdot \frac{\left|\vec{E}_s\right|^2}{\left|\vec{E}_i\right|^2} \tag{A.14}$$

Dabei steht $\vec{E}_i$ für die am Ziel einfallende Feldstärke und $\vec{E}_s$ für die zurückgestreute Feldstärke (nach Knott et al, 2004). σ hat die Dimension einer Fläche. Der Rückstreuquerschnitt σ gilt für Punktziele. Flächenhafte Ziele werden durch den dimensionslosen Rückstreukoeffizient $\sigma°$ beschrieben. Dieser ist definiert als der Radarrückstreuquerschnitt eines Ziels mit einer horizontalen Flächenausbreitung *F*, normalisiert mit *F*:

$$\sigma° = \frac{\sigma}{F} \tag{A.15}$$

(nach Ulaby & Dobson, 1989)

Für die Berechnung von flächenhaften Zielen muss die Gleichung (A.12) erweitert werden:

$$P_R = \iint_F \frac{P_S \cdot G^2 \cdot \lambda^2}{(4\pi)^3 \cdot {r_O}^4} \cdot \sigma° \tag{A.16}$$

Anhang B – Grafikprozessierung

B.1 Die Vertex-Prozessierung

B.1.1 Welttransformation

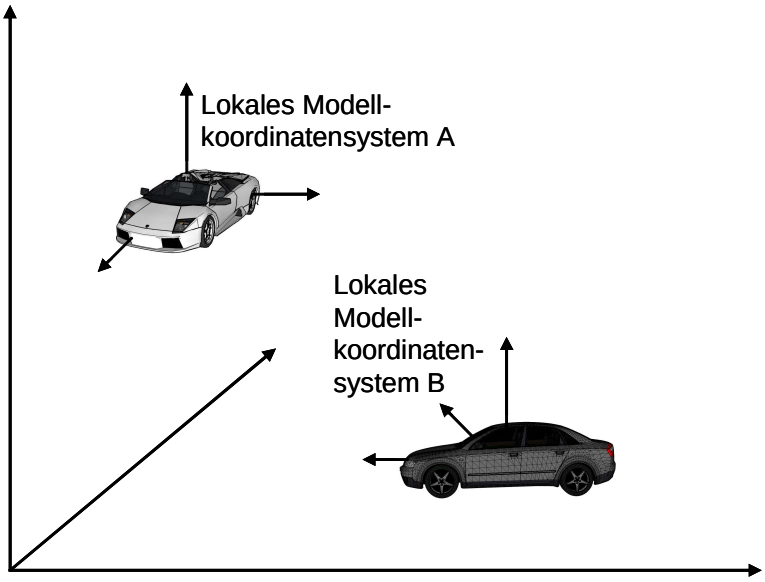

Abbildung B-1: Das Weltkoordinatensystem

Die darzustellenden Modelle liegen, wie in Abbildung B-1 illustriert, normalerweise in Modellkoordinaten vor und müssen zunächst in ein einheitliches Weltkoordinatensystem transformiert werden. Die Modelle können dabei rotiert, skaliert und verschoben werden. Das Weltkoordinatensystem ist gültig für alle Objekte in der Szene und beschreibt die Lage der Objekte zueinander. Die Welttransformationsmatrix setzt sich aus einer Translation (T), Skalierungen (S) entlang den Achsenrichtungen und Rotationen (R) entlang den Achsen zusammen (Gray, 2003).

$$\begin{bmatrix} \text{Welt-} \\ \text{transformation} \end{bmatrix} = [\mathrm{S_x}] \cdot [\mathrm{S_y}] \cdot [\mathrm{S_z}] \cdot [R_z] \cdot [R_y] \cdot [R_x] \cdot [\mathrm{T}] \qquad \text{(B.1)}$$

In DirectX erfolgt, im Gegensatz zu OpenGL, die Verkettung von links nach rechts, so dass die Objekte zunächst skaliert, dann rotiert und zuletzt verschoben werden.

B.1.2 Kameratransformation

Im sogenannten Kameraraum liegt die Kamera im Ursprung des Koordinatensystems. mit Blick in die positive Z-Richtung. Der Kameraraum wird gebildet aus:

1. Kameraposition im Weltkoordinatensystem *k*
2. Der Beobachtungspunkte *p*
3. Dem sogenannten „up-vector“ der „oben“ definiert $\vec{o}$

Hieraus werden die Achsen des Koordinatensystems wie folgt berechnet:

$$\vec{z} = \frac{p-k}{\|p-k\|}$$

$$\vec{x} = \frac{\vec{o} \times \vec{z}}{\left\| \vec{o} \times \vec{z} \right\|}$$
$$\vec{y} = \vec{z} \times \vec{x} \tag{B.2}$$

Die Kameramatrix setzt sich folgendermaßen zusammen:

$$\begin{bmatrix} \vec{x}_x & \vec{y}_x & \vec{z}_x & 0 \\ \vec{x}_y & \vec{y}_y & \vec{z}_y & 0 \\ \vec{x}_z & \vec{y}_z & \vec{z}_z & 0 \\ \langle \vec{x}, k \rangle & \langle \vec{y}, k \rangle & \langle \vec{z}, k \rangle & 1 \end{bmatrix} \tag{B.3}$$

(Microsoft, 2005)

B.1.3 Projektionstransformation

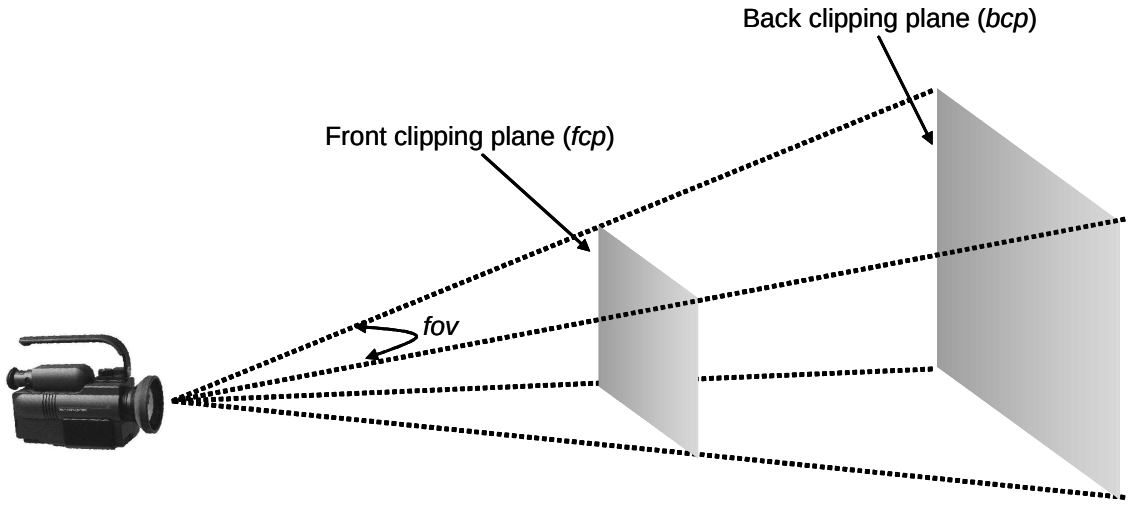

Abbildung B-2: Die Projektionstransformation

Bei der Projektionstransformation werden die Daten entsprechend ihrer Lage zur Kamera transformiert. Zur Berechnung der Ausgabekoordinaten werden das Bildseitenverhältnis (*ar*) und der Sehbereich (*fov*), sowie die sogenannte „front clipping plane“ (*fcp*) und „back clipping plane“ (*bcp*) definiert (siehe Abbildung B-2). Objekte, die näher an der Kamera als die „front clipping plane“ oder weiter von der Kamera entfernt als die „back clipping plane“ liegen, werden nicht dargestellt. Nur Objekte, die räumlich zwischen den beiden Ebenen liegen, werden dargestellt. Danach werden die Daten noch entsprechend der Größe des Ausgabebereiches skaliert und auf die zweidimensionalen Bildschirmkoordinaten transformiert. Die Projektionsmatrix setzt sich zusammen aus:

$$\begin{bmatrix} \cot(\frac{fov}{2}) \cdot ar & 0 & 0 & 0 \\ 0 & \cot(\frac{fov}{2}) & 0 & 0 \\ 0 & 0 & \frac{bcp}{bcp - fcp} & 1 \\ 0 & 0 & \frac{-fcp \cdot bcp}{bcp - fcp} & 0 \end{bmatrix} \tag{B.4}$$

(Microsoft, 2005)

B.2 Die High-Level Shading Language (HLSL)

Die High-Level Shading Language (HLSL) ist eine C-ähnliche Hochsprache, die zur Programmierung von sogenannten Shader-Programmen auf Grafikkarten entwickelt wurde. HLSL ist speziell auf die Bedürfnisse der Computergrafik zugeschnitten und besitzt daher einfache und leistungsstarke Befehle zur Matrixmultiplikation. Außerdem werden spezielle Datentypen wie z.B. float4, welches einen Array aus vier float-Werten repräsentiert, unterstützt. Ein float4 kann sowohl für die Darstellung von homogenen Koordinaten als auch von Farbwerten benutzt werden.

In HLSL können Vertex-, Pixel-, Textur-, und ab DirectX10 auch Geometry-Shader programmiert werden. Vertex-Shader bearbeiten die Vertexe und sind für die Transformation der Geometrien zuständig. Die Transformation eines Modells von den lokalen Koordinaten in Bildschirmkoordinaten erfolgt über Matrixmultiplikationen mit den unter Kapitel B.1 beschriebenen Matrizen. Ein einfacher Vertex-Shader kann in HLSL beispielsweise folgendermaßen implementiert werden:

```
void VertexShader (float4 Pos : POSITION,
                   float4 Col : COLOR,
                   out float4 oPos : POSITION,
                   out float4 oCol : COLOR0)
{
      float4 worldPos = mul(Pos, worldMatrix);
      float4 viewPos  = mul(worldPos, viewMatrix);

      oPos = mul(viewPos, projectionMatrix);
      oCol = Col;
}
```

Dieser einfache Vertex-Shader implementiert die fixed-function Pipeline neu. Die Position des Punktes wird mit der `worldMatrix`, der `viewMatrix` und schließlich der `projectionMatrix` multipliziert und die Bildschirmkoordinaten dann weiter- bzw. ausgegeben. Der Farbwert wird nicht verändert, sondern direkt weitergegeben. Im Pixel-Shader wird dann pro auszugebendem Pixel ein Farbwert berechnet.

```
float4 PixelShader(in float4 iCol : COLOR0) : COLOR
{
      float4 inverseColor = 1.0f - iCol;
      return inverseColor;
}
```

Dieser einfache Pixel-Shader invertiert die vom Vertex-Shader übergebene Farbinformation `iCol` und gibt die invertierte Farbe aus.

Literatur

Akenine-Möller, T., Haines, E. (2002): *Real-Time Rendering. Second Edition*, Natick, Massachusetts.

Armand, P., Vidal-Madjar, D. (1992): Synthetic Aperture Radar Simulation on Radar Terrain Clutter. In: *Proceedings of the Geoscience and Remote Sensing Symposium, 1992. IGARSS '92*, Houston.

Ashdown, I. (1994): *Radiosity: a programmer's perspective*, New York.

Auterman, J.L. (1984): Phase stability requirements for bistatic SAR. In: *Proceedings of the IEEE National Radar Conference 1984*, Atlanta.

Balz, T. (2006a): Automated CAD model-based geo-referencing for high-resolution SAR data in urban environment. In: *IEE Proceedings - Radar, Sonar and Navigation*, 153, S. 289-293.

Balz, T. (2006b): Real Time SAR-Simulation on Graphics Processing Units. In: *Proceedings of the 6th European Conference on Synthetic Aperture Radar (EUSAR 2006)*, Dresden.

Balz, T., Haala, N. (2003): SAR-based 3D-reconstruction of complex urban environments. In: *IAPRS Vol. 34, Part 3/W13*, Dresden.

Bamler, R. (1999): The SRTM Mission: A World-Wide 30 m Resolution DEM from SAR-Interferometry in 11 Days. In: Fritsch, D. & Spiller, R.: *Photogrammetric Week '99*, Heidelberg, S. 145-154.

Bickert, B., Meyer-Hilberg, J., Schmid, J. (2002): A radar raw data generator and modular algorithm simulation system for MTI development. In: *Proceedings of the 4th European Conference on Synthetic Aperture Radar (EUSAR 2002)*, Köln.

Blake, L.V. (1986): *Radar range-performance analysis*, Norwood, Massachusetts.

Blinn, J.F. (1977): Models of light reflection for computer synthesized pictures. In: *Proceedings of the 4th annual conference on Computer graphics and interactive techniques (SIGGRAPH '77)*, San Jose.

Blinn, J.F. (1978): Simulation of wrinkled surfaces. In: *Proceedings of the 5th annual conference on Computer graphics and interactive techniques (SIGGRAPH '78)*, New York

Blinn, J.F., Newell, M.E. (1976): Texture and reflection in computer generated images. In: *Communications of the ACM*, 19, S. 542-547.

Blythe, D. (2006): The Direct3D 10 System. In: *ACM Transactions on Graphics (TOG), Proceedings of the 33th annual conference on Computer graphics and interactive techniques SIGGRAPH '06*, Boston.

Bolter, R., Gelautz, M., Leberl, F. (1996): SAR Speckle Simulation. In: *Proceedings of the ISPRS Congress 1996. IAPRS XXXI-B2*, Wien.

Borchers, M.P. (2005): *Interaktive und stereoskopische Visualisierung in der speziellen Relativitätstheorie*, Tübingen.

Bouknight, W.J. (1970): A Procedure for Generation of Three-dimensional Half-toned Computer Graphics Presentation. In: *Communications of the ACM*, 13, S. 527-536.

Brand, B., De Florio, S., Neff, T., Speck, R., Zehetbauer, T. (2006): Simulation Tool for SAR Missions. In: *Proceedings of the 6th European Conference on Synthetic Aperture Radar (EUSAR 2006)*, Dresden.

Brenner, A., Ender, J. (2004): Airborne SAR Imaging with Subdecimeter Resolution. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Buck, I., Fatahalian, K., Hanrahan, P. (2004): GPUBench: Evaluating GPU performance for numerical and scientific applications. In: *2004 ACM Workshop on General-Purpose Computing on Graphics Processors*, Los Angeles.

Buck, I. (2004): GPU computation strategies and tips. In: *Proceedings of the conference on SIGGRAPH 2004 course notes GRAPH '04*, Los Angeles.

Buck, I., Foley, T., Horn, D., Sugerman, J., Fatahalian, K., Houston, M., Hanrahan, P. (2004): Brook for GPUs: stream computing on graphics hardware. In: *ACM Transactions on Graphics*, 23, S. 777-786.

Cabral, B., Cam, N., Foran, J. (1994): Accelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware. In: *Proceedings of the 1994 symposium on Volume visualization*, Tysons Corner, Virginia.

Cantalloube, H., Wendler, M., Giroux, V., Dubois-Fernandez, P., Krieger, G. (2004): Challenges in SAR processing for airborne bistatic acquisition. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Cathala, T., Le Goff, A., Gozard, P., Latger, J. (2006): Real time simulation tools in the CHORALE workshop. In: *Targets and Backgrounds XII: Characterization and Representation. Proceedings of the SPIE, Volume 6239*, Orlando, Florida.

Catmull, E. (1978): A hidden-surface algorithm with anti-aliasing. In: *ACM SIGGRAPH Computer Graphics, Proceedings of the 5th annual conference on Computer graphics and interactive techniques SIGGRAPH '78*, Atlanta.

Cloude, S.R., Pottier, E. (1996): A Review of Target Decomposition Theorems in Radar Polarimetry. In: *IEEE Transactions on Geoscience and Remote Sensing*, 34, S. 498-518.

Crow, F.C. (1977): Shadow algorithms for computer graphics. In: *ACM SIGGRAPH Computer Graphics, Proceedings of the 4th annual conference on Computer graphics and interactive techniques SIGGRAPH '77*, San Jose.

Debevec, P.E. (2004): Making "The Parthenon". In: *SIGGRAPH 2004 Electronic Theater*, Los Angeles.

Delignon, Y., Pieczynski, W. (2002): Modelling Non-Rayleigh Speckle Distribution in SAR images. In: *IEEE Transactions on Geoscience and Remote Sensing*, 40, S. 1430-1435.

Domik, G., Kobrick, M., Leberl, F. (1984): Analyse von Radarbildern mittels Digitaler Höhenmodelle. In: *Bildmessung und Luftbildwesen*, 52, S. 249-263.

Domik, G., Leberl, F.W. (1987): Image Based SAR Product Simulation for Analysis. In: *Proceedings of the American Society for Photogrammetry and Remote Sensing and ACSM Annual Convention*, Baltimore.

Dubois-Fernandez, P., Cantalloube, H., Ruault du Plessis, O., Wendler, M., Vaizan, B., Coulombeix, C., Heuzé, D., Krieger, G. (2004): Analysis of bistatic scattering behaviour of natural surfaces. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Ekman, M., Wark, F., Nilsson, J. (2005): An in-depth look at computer performance growth. In: *ACM SIGARCH Computer Architecture News*, 33, S. 144-147.

Elachi, C. (1988): *Spaceborne radar remote sensing: Applications and techniques*, New York.

Ender, J.H.G., Brenner, A.R. (2003): PAMIR - a wideband phased array SAR/MTI system. In: *IEE Proceedings - Radar, Sonar and Navigation*, 150, S. 165-172.

Ender, J.H.G., Klare, J., Walterscheid, I., Brenner, A.R., Weiß, M., Kirchner C., Wilden, H., Loffeld, O., Kolb, A., Wiechert, W., Kalkuhl, M., Knedlik, S., Gebhardt, U., Nies, H., Natroshvili, K., Ige, S., Medrano Ortiz, A., Amankwah, A. (2006a): Bistatic exploration using spaceborne and airborne SAR sensors: A close collaboration between FGAN, ZESS, and FOMAAS. In: *Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium and 27th Canadian Symposium on Remote Sensing (IGARSS 2006)*, Denver, Colorado.

Ender, J.H.G., Walterscheid, I., Brenner, A.R. (2006b): Bistatic SAR - translational invariant processing and experimental results. In: *IEE Proceedings - Radar, Sonar and Navigation*, 153, S. 177-183.

Everitt, C., Kilgard, M.J. (2002): Practical and Robust Stenciled Shadow Volumes for Hardware-Accelerated Rendering. In: *http://developer.nvidia.com (Stand 16.08.2006)*

Everitt, C., Rege, A., Cebenoyan, C. (2001): Hardware Shadow Mapping. In: *http://developers.nvidia.com (Stand 10.07.2006)*

FAS (1998): National Image Interpretability Rating Scales. In: *http://www.fas.org/irp/imint/niirs.htm (Stand 28.11.2006)*

Fatahalian, K., Sugerman, J., Hanrahan, P. (2004): Understanding the efficiency of GPU algorithms for matrix-matrix multiplikation. In: *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, Grenoble.

Fernando, R., Fernandez, S., Bala, K., Greenberg, D.P. (2001): Adaptive shadow maps. In: *Proceedings of the ACM SIGGRAPH 2001: 28th International Conference on Computer Graphics and Interactive Techniques*, Los Angeles.

Fischer, J.T. (2006): *Rendering Methods for Augmented Reality*, Tübingen.

Foley, J.D. (1996): *Computer graphics: principles and practices*, Reading, Massachusetts.

Forberg, A. (2004): Generalization of 3D Building Data Based on Scale-Space Approach. In: *IAPRS Vol. 35, Part B*, Istanbul.

Forberg, A., Mayer, H. (2002): Generalization of 3D Building Data Based on Scale-Spaces. In: *Proceedings of the Joint International Symposium on Geospatial Theory, Processing and Applications*, Ottawa.

Forsyth, T. (2001): Impostors: Adding Clutter. In: DeLoura, M.: *Game Programming Gems 2*, Hingham, Massachusetts, S. 488-497.

Franceschetti, G., Migliaccio, M., Riccio, D. (1994): SAR Raw Signal Simulation of Actual Ground Sites in Terms of Sparse Input Data. In: *IEEE Transactions on Geoscience and Remote Sensing*, 32, S. 1160-1169.

Franceschetti, G., Migliaccio, M., Riccio, D. (1995): The SAR simulation: an overview. In: *Proceedings of IGARSS '95. Quantitative Remote Sensing for Science and Application*, Florenz.

Franceschetti, G., Migliaccio, M., Riccio, D., Schirinzi, G. (1992): SARAS: A Synthetic Aperture Radar (SAR) Raw Signal Simulator. In: *IEEE Transactions on Geoscience and Remote Sensing*, 30, S. 110-123.

Fritsch, D. (2003): 3D Building Visualisation - Outdoor and Indoor Applications. In: Fritsch, D.: *Photogrammetric Week '03*, Heidelberg, S. 281-290.

Fung, A.K. (1994): *Microwave Scattering and Emission Models and Their Applications*, Boston.

Fung, A.K., Li, Z., Chen, K.S. (1992): Backscattering from a Randomly Rough Dielectric Surface. In: *IEEE Transactions on Geoscience and Remote Sensing*, 30, S. 356-369.

Fung, J., Mann, S. (2005): OpenVIDIA: parallel GPU computer vision. In: *Proceedings of the 13th annual ACM international conference on Multimedia*, Singapore.

Galoppo, N., Govindaraju Naga K., Henson, M., Manocha, D. (2005): LU-GPU: Efficient Algorithms for Solving Dense Linear Systems on Graphics Hardware. In: *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, Washington DC, USA.

Gebhardt, U., Loffeld, H., Nies, K., Natroshvili, K., Knedlik, S. (2006): Bistatic Space Borne / Airborne Experiment: Geometrical Modeling and Simulation. In: *Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium and 27th Canadian Symposium on Remote Sensing (IGARSS 2006)*, Denver, Colorado.

Geisler, J., Pallmer, D., Littfaß, M., Kerker, R., Schoenbein, R., Schumacher, W. (2000): RecceMan – Interactive Interface for Target Recognition Support and Training. In: *Proceedings of the 4th Joint International Military Sensing Symposium (MSS)*, Palaiseau, France.

Geisler, J., Peinsipp-Byma, E., Littfaß, M., Angele, S. (2004): Smart recognition assistance for multisensor-image-based reconnaisance. In: *Proceedings of the 6th Joint International Military Sensing Symposium (MSS)*, Dresden.

Gierull, C., Ruppel, M. (1996): An End-to-End Synthetic Aperture Radar Simulator. In: *Proceedings of the European Conference on Synthetic Aperture Radar (EUSAR '96)*, Königswinter.

Godt, B. (2003): *Aspekte der Radarentwicklung und -anwendung im Zweiten Weltkrieg*, Konstanz.

Goral, C.M., Torrance, K.E., Greenberg, D.P., Battaile, B. (1984): Modeling the interaction of light between diffuse surfaces. In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques (SIGGRAPH 1984)*, Minneapolis.

Govindaraju, N.K., Lloyd, B., Wang, W., Lin, M., Manocha, D. (2004): Fast computation of database operations using graphics processors. In: *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, Paris.

Gray, K. (2003): *Microsoft DirectX 9 Programmable Graphics Pipeline*, Redmond, Washington.

Greene, N. (1986): Environment Mapping and Other Applications of World Projections. In: *IEEE Computer Graphics and Applications*, 6, S. 21-29.

Gross, M. (2001): Are Points the Better Graphics Primitives? In: *Computer Graphics Forum*, 20, S. 17.

Guindon, B. (1993): Development of a SAR data acquisition planning tool (SARPLAN) based on image simulation. In: *International Journal of Remote Sensing*, 14, S. 333-344.

Haenselmann, T. (1996): *Raytracing*, Bonn.

Harris, M. (2005): Mapping Computational Concepts to GPUs. In: Pharr, M.: *GPU Gems 2. Programming Techniques for High-Performance Graphics and General-Purpose Computation*, Boston, S. 493-508.

Heckbert, P.S. (1990): Adaptive Radiosity Textures for Bidirectional Ray Tracing. In: *Computer Graphics*, 24, S. 145-154.

Heidmann, T. (1991): Real Shadows Real Time. In: *IRIS Universe: The Magazine of Visual Computing*, 18, S. 28-31.

Hoffmann, K., Fischer, P. (2002): DOSAR: A Multifrquency Polarimetric and Interferometric Airborne SAR-System. In: *2002 International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing*, Toronto.

Holtzman, J.C., Frost, V.S., Abbott, J.L., Kaupp, V.H. (1978): Radar Image Simulation. In: *IEEE Transactions on Geoscience Electronics*, 16, S. 297-303.

Horell, J.M., Inggs, M.R. (1993): Satellite and airborne SAR simulator. In: *Proceedings of the 1993 IEEE South African Symposium on Communications and Signal Processing (COMSIG 93)*, Johannesburg.

Hülsmeyer, C. (1904): Verfahren, um entfernte metallische Gegenstände mittels elektrischer Wellen einem Beobachter zu melden, Patent DRP 165546 und DRP 169154.

IEEE (1998): *IEEE Standard Radar Definitions. IEEE Std 686-1997*, New York.

Kada, M. (2004): Hardware-Based Texture Extraction for Building Facades. In: *Proceedings of the XXth Congress of the ISPRS, Vol. 35, Part B*, Istanbul.

Kada, M. (2006): 3D Building Generalization Based on Half-Space Modelling. In: *Proceedings of the ISPRS Workshop on Multiple Representation and Interoperability of Spatial Data. IAPRS, Vol. 34, Part XXX*, Hannover.

Kaupp, V.H., Bridges, L.C., Pisaruck, M.A., MacDonald, H.C., Waite, W.P. (1983): Simulation of Spaceborne Stereo Radar Imagery: Experimental Results. In: *IEEE Transactions on Geoscience and Remote Sensing*, 21, S. 400-405.

Kent, S.L. (2001): *The ultimate history of video games: from Pong to Pokémon and beyond – the story behind the craze that touched our lives and changed the world*, New York.

Kesenvich, J. (2006): The OpenGL Shading Language. Language Version: 1.20. In: *http://www.opengl.org/documentation/glsl.html (Stand 28.11.2006)*

Keydel, W. (2003): Perspectives and visions for future SAR systems. In: *IEE Proceedings - Radar, Sonar and Navigation*, 150, S. 97-103.

Kirscht, M., Rinke, K. (1998): 3D Reconstruction of Buildings and Vegetation from Synthetic Aperture Radar (SAR) images. In: *MVA, IAPR Workshop*, Chiba, Japan.

Klausing, H., Holpp, W. (2000): *Radar mit realer und synthetischer Apertur*, München.

Knott, E.F., Shaeffer, J.F., Tuley, M.T. (2004): *Radar Cross Section. Second Edition.*, Raleigh, North Carolina.

Krieger, G., Moreira, A. (2006): Spaceborne bi- and multistatic SAR: Potential and challenges. In: *IEE Proceedings - Radar, Sonar and Navigation*, 153, S. 184-198.

Krieger, G., Fiedler, H., Moreira, A. (2004): Bi- and Multistatic SAR: Potentials and Challenges. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Krüger, J., Kipfer, P., Kondratieva, P., Westermann, R. (2005): A Particle System for Interactive Visualization of 3D Flows. In: *IEEE Transactions on Visualization and Computer Graphics*, 11, S. 744-756.

Krüger, J., Westermann, R. (2003): Linear algebra operators for GPU implementation of numerical algorithm. In: *ACM Transactions on Graphics*, 22, S. 908-916.

La Prade, G.L. (1963): An Analytical and Experimental Study of Stereo for Radar. In: *Photogrammetric Engineering*, 29, S. 294-300.

Leberl, F.W. (1990): *Radargrammetric image processing*, Norwood, Massachusetts.

Lee, J.-S., Ainsworth, T.L., Krogagor, E., Boerner, W.-M. (2006): Polarimetric Analysis of Radar Signature of a Manmade Structure. In: *Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium and 27th Canadian Symposium on Remote Sensing (IGARSS 2006)*, Denver, Colorado.

Lefohn, A.E., Kniss, J.M., Hansen, C.D., Whitaker, R.T. (2004): A Streaming Narrow-Band Algorithm: Interactive Computation and Visualization of Level Sets. In: *IEEE Transactions on Visualization and Computer Graphics*, 10, S. 422-433.

Lucas, P., Fritz, N., Wilhelm, R. (2006): The CGiS Compiler - A Tool Demonstration. In: Mycroft, I. & Zeller, A.: *Proceedings of the 15th International Conference on Compiler Construction*, Berlin, S. 105-108.

Mametsa, H.-J., Rouas, F., Berges, A., Latger, J. (2002): Imaging Radar Simulation in Realistic Environment Using Shooting and Bouncing Rays Technique. In: *Proceedings of SPIE: 4543. SAR Image Analysis, Modeling and Techniques IV*, Toulouse.

Marconi (1984): SAR Simulation Concept and Tools, Final Report. In: *Report MTR 84/34*, Marconi Research Centre, United Kingdom.

Markham, K.C. (1989): Comparisson of segmentation processes for object acquisition in infrared images. In: *IEE Proceedings Radar and Signal Processing*, 136, S. 13-21.

Martin, T., Tan, T.-S. (2004): Anti-aliasing and Continuity with Trapezoidal Shadow Maps. In: *Proceedings of the Eurographics Symposium on Rendering*, Norrköpping, Schweden.

McGuire, M. (2004): Efficient Shadow Volume Rendering. In: Fernando, R.: *GPU Gems. Programming Techniques, Tips, and Tricks for Real-Time Graphics*, Boston, S. 137-166.

McReynolds, T., Blythe, D. (2005): *Advanced Graphics Programming Using OpenGL*, San Francisco.

Mercer, B. (2001): Combining LIDAR and IfSAR: What can you expect? In: Fritsch, D. & Spiller, R.: *Photogrammetric Week '01*, Heidelberg, S. 227-237 .

Meyer-Hilberg, J. (2006): PIRDIS: A New Versatile Tool for SAR/MTI Systems Simulation. In: *Proceedings of the 6th European Conference on Synthetic Aperture Radar (EUSAR 2006)*, Dresden.

Microsoft (2005): DirectX 9.0c Software Development Kit. In: *http://www.microsoft.com/directx (Stand 11.07.2006)*

Mittermayer, J. (2000): *Hochauflösende Verarbeitung von Radardaten mit synthetischer Apertur*, Köln.

Moore, G.E. (1965): Cramming more components onto integrated circuits. In: *Electronics*, 38, S. 114-117.

Moreira, A., Krieger, G., Hajnsek, I., Hounman, D., Werner, M., Riegger, S., Settelmeyer, E. (2004): TanDEM-X: A TerraSAR-X Add-On Satellite for Single-Pass Interferometry. In: *Proceedings of the 2004 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2004)*, Anchorage.

Moreira, J., Schwäbisch, M., Wimmer, C., Rombach, M., Mura, J. (2001): Surface and Ground Topography Determination in Tropical Rainforest Areas Using Airborne Interferometric SAR. In: Fritsch, D. & Spiller, R.: *Photogrammetric Week '01*, Heidelberg, S. 167-173.

Morel, J.-M., Solimini, S. (1995): *Variational methods in image segmentation*, Boston.

Muhleman, D.O. (1964): Radar Scattering from Venus and the Moon. In: *The Astronomical Journal*, 69, S. 34-41.

Newbry, L.E. (1960): Terrain Radar Reflectance Study. In: *Photogrammetric Engineering*, 26, S. 630-637.

NVIDIA (2005a): NVIDIA GPU Programming Guide. Version 2.4.0. In: *http://developer.nvidia.com (Stand 28.11.2006)*

NVIDIA (2005b): NVShaderPerf. In: *http://developer.nvidia.com/object/nvshaderperf_home.html (Stand 17.01.2007)*

NVIDIA (2007): CUDA Programming Guide Version 0.8. In: *http://developer.nvidia.com (Stand 16.02.2007)*

Owens, J.D. (2005): Streaming Architectures and Technology Trends. In: Pharr, M.: *GPU Gems 2. Programming Techniques for High-Performance Graphics and General-Purpose Computation*, Boston, S. 457-470.

Owens, J.D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A.E., Purcell, T.J. (2005): A Survey of General-Purpose Computation on Graphics Hardware. In: *Eurographics 2005, State of the Art Reports*, Dublin.

Owens, J.D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A.E., Purcell, T.J. (2007): A Survey of General-Purpose Computation on Graphics Hardware. In: *Computer Graphics Forum*, 26, S. 80-113.

Pal, N.R., Pal, S.K. (1993): A review on image segmentation techniques. In: *Pattern Recognition*, 26, S. 1277-1294.

Phong, B.T. (1975): Illumination for computer generated pictures. In: *Communications of the ACM*, 18, S. 311-317.

Pike, T.K. (1985): SARSIM: A Synthetic Aperture Radar System Simulation Model. In: *DFVLR-Mitt, 85-11*, Oberpfaffenhofen.

Purcell, T.J. (2004): *Ray Tracing on a Stream Processor*, PhD thesis, Stanford University.

Purcell, T.J., Buck, I., Mark, W.R., Hanrahan, P. (2002): Ray tracing on programmable graphics hardware. In: *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, San Antonio, Texas.

Ratches, J.A., Walters, C.P., Buser, R.G., Guenther, B.D. (1997): Aided and Automatic Target Recognition Based Upon Sensory Inputs From Image Forming Systems. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, S. 1004-1019.

Sainz, M., Pajarola, R. (2004): Point-based rendering techniques. In: *Computer & Graphics*, 28, S. 869-879.

Schmittler, J., Wald, I., Slusallek, P. (2002): SaarCOR - A Hardware Architecture for Ray Tracing. In: Ertl, T., Heidrich, W., Doggett, M.: *Graphics Hardware* , Saarbrücken, Germany, S. 1-11.

Schreier, G. (1993): Geometrical Properties of SAR Images. In: Schreier, G.: *SAR Geocoding: Data and Systems*, Karlsruhe, S. 103-134.

Sherbondy, A., Houston, M., Napel, S. (2003): Fast Volume Segmentation With Simultaneous Visualization Using Programmable Graphics Hardware. In: *Proceedings of the 14th IEEE Visualization Conference 2003 (VIS 2003)*, Seattle, Washington.

Shirley, P., Morley, R.K. (2003): *Realistic Ray Tracing*, Wellesley, Massachusetts.

Shreiner, D. (2000): *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.2*, Reading, Massachusetts.

Skolnik, M.I. (2001): *Introduction to radar systems*, New York.

Soergel, U., Schulz, K., Thoennessen, U., Stilla, U. (2003): Event-driven SAR Data Acquisition in Urban Areas Using GIS. In: *GIS*, 16, S. 32-37.

Soergel, U., Thoenessen, U., Stilla, U., Brenner, A. (2004): New opportunities and challenges for analyiss of urban areas in high resolution SAR data. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Soergel, U., Thoennessen, U., Brenner, A., Stilla, U. (2006): High-resolution SAR data: new opportunities and challenges for the analysis of urban areas. In: *IEE Proceedings - Radar, Sonar and Navigation*, 153, S. 294-300.

Song, H.J., Zhu, M.H., Bai, Y.T. (1997): Design of General Purpose Simulation Package of SAR System. In: *Proceedings of Radar 97*, Edinburgh.

Speck, R., Hager, M., Süß, H. (2002): An End-to-End Simulator for Spaceborne SAR-Systems. In: *Proceedings of the 4th European Conference on Synthetic Aperture Radar (EUSAR 2002)*, Köln.

Stacy, N.J.S., Craig, D.W., Staromlynska, J., Smith, R.B. (2002): The Global Hawk UAV Australian Deployment: Imaging Radar Sensor Modifications and Employment for Maritime Surveillance. In: *Proceedings of the 2002 International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing*, Toronto.

Stamminger, M., Drettakis, G. (2002): Perspective shadow maps. In: *ACM Transactions on Graphics (TOG), Proceedings of the 29th annual conference on Computer graphics and interactive techniques SIGGRAPH '02*, San Antonio, Texas.

Stilla, U., Soergel, U., Thoennessen, U. (2003): Potential and limits of InSAR data for building reconstruction in built-up areas. In: *ISPRS Journal of Photogrammetry & Remote Sensing*, 58, S. 113-123.

Stilla, U., Sorgel, U. (2006): Reconstruction of buildings in SAR imagery of urban areas. In: Weng, Q. & Quattrochi, D.A.*: Urban Remote Sensing*, Boca Raton, FL, USA, S. 47-67.

Sumanaweera, T., Liu, D. (2005): Medical image reconstruction with the FFT. In: Pharr, M.: *GPU Gems 2*, S. 765-784.

Sutherland, I.E. (1963): *Sketchpad, a man-machine graphical communication system*, Massachusetts Institute of Technology.

Tarditi, D., Puri, S., Oglesby, J. (2006): Accelerator: using data parallelism to program GPUs for general-purpose uses. In: *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, San Jose.

thedodgegarage.com (2007): The Dodge Garage 3dfx Collection. In: *http://www.thedodgegarage.com/3dfx/ (Stand 15.01.2007)*

Thiemann, F. (2002): Generalization of 3D Building Data. In: *Proceedings of the Joint International Symposium on Geospatial Theory, Processing and Applications*, Ottawa, Kanada.

Trenkle, F. (1979): *Die deutschen Funkmeßverfahren bis 1945*, Stuttgart.

Ulaby, F.T., Moore, R.K., Fung, A.K. (1982): *Microwave Remote Sensing. Volume II: Radar Remote Sensing and Surface Scattering and Emission Theory*, London.

Ulaby, F.T., Moore, R.K., Fung, A.K. (1986): *Microwave Remote Sensing. Volume III: From Theory to Applications*, London.

Ulaby, F.T., Dobson, M.C. (1989): *Handbook of Radar Scattering Statistics for Terrain*, Norwood, Massachusetts.

Ulbricht, A. (2000): *Flugzeuggetragene Repeat-Pass-Interferometrie mit dem E-SAR*, Köln.

UN-HABITAT (2006): *State of the Worlds Cities 2006/07: The Millenium Goals and Urban Sustainability*, Nairobi, Kenya.

Upstill, S. (1990): *The RenderMan Companion: A Programmer's Guide to Realistic Computer Graphics*, Reading, Massachusetts.

van Zyl, J.J. (1989): Unsupervised Classification of Scattering Behavior Using Radar Polarimetry Data. In: *IEEE Transactions on Geoscience and Remote Sensing*, 27, S. 37-45.

Vandewal, M., Speck, R., Süß, H. (2006): A UAV-Based SAR Raw Data Simulator for Complex Scenes. In: *Proceedings of the 6th European Conference on Synthetic Aperture Radar (EUSAR 2006)*, Dresden.

Wald, I., Slusallek, P., Benthin, C., Wagner, M. (2001): Interactive Rendering with Coherent Ray Tracing. In: *Computer Graphics Forum*, 20, S. 153-164.

Walter, V. (1997): *Zuordnung von raumbezogenen Daten - am Beispiel ATKIS und GDF*, München.

Walterscheid, I., Brenner, A.R., Ender, J.H.G. (2004): Geometry and System Aspects for a Bistatic Airborne SAR-Experiment. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Wang, Y., Molnar, S. (1994): Second-Depth Shadow Mapping. In: *UNC-CS Technical Report TR94-019*, Chapel Hill, North Carolina.

Weimann, A., von Schönermark, M., Schumann, A., Jörn, P., Gunther, R. (1998): Soil moisture estimation with ERS-1 SAR data in East-German loess soil area. In: *International Journal of Remote Sensing*, 19, S. 237-243.

Werninghaus, R., Balzer, W., Buckreuss, J.M., Mühlbauer, P., Pitz, W. (2004): The TerraSAR-X Mission. In: *Proceedings of the 5th European Conference on Synthetic Aperture Radar (EUSAR 2004)*, Ulm.

Whitted, T. (1980): An improved illumination model for shaded display. In: *Communications of the ACM*, 23, S. 343-349.

Wildey, R.L. (1986): Radarclinometry for the Venus Radar Mapper. In: *Photogrammetric Engineering and Remote Sensing*, 52, S. 41-50.

Wiley, C. (1954): Pulsed Doppler Radar Method and Mean, Patent US 3.196.436.

Williams, L. (1978): Casting curved shadows on curved surfaces. In: *ACM SIGGRAPH Computer Graphics, Proceedings of the 5th annual conference on Computer graphics and interactive techniques SIGGRAPH '78*, Atlanta.

Willis, N.J. (1991): *Bistatic Radar*, Norwood, Massachusetts.

Wohlers, M.R., Hsiao, S., Mendelsohn, J., Gardner, G. (1980): Computer Simulation of Synthetic Aperture Radar Images of Three-Dimensional Objects. In: *IEEE Transactions on Aerospace and Electronic Systems*, 16, S. 258-271.

Xu, F., Mueller, K. (2005): Accelerating Popular Tomographic Reconstruction Algorithms on Commodity PC Graphics Hardware. In: *IEEE Transactions on Nuclear Science*, 52, S. 654-663.

Yang, Y.E., Ewe, H.T., Wang, L., Shih, S.E., Ding, K.H., Kong, J.A., Shin, R.T., Nguyen, H., Nguyen, T., Ho, J.Y., O'Neill, K. (1994): A GIS-Driven Interactive Radar Image Simulation Using EMSARS Model. In: *Proceedings of IGARSS '94. Surface and Atmospheric Remote Sensing: Technologies, Data Analysis and Interpretation*, Pasadena, California.

Yates, G., Horne, A.P., Blake, A.P., Middleton, R. (2006): Bistatic SAR image formation. In: *IEE Proceedings - Radar, Sonar and Navigation*, 153, S. 208-213.

Yong, G., Wang, J., Lou, X. (1999): Research on the Framework of Computer 3D Simulation SAR Integrated System. In: *Proceedings of the 20th Asian Conference on Remote Sensing*, Hong Kong.

Zink, M. (1993): *Kalibration von SAR-Systemen*, Oberpfaffenhofen.

Zribi, M., Baghdadi, N., Guérin, C. (2006): A new Semi-empirical model for the analysis of surface roughness heterogenity. In: *Proceedings of the 2006 IEEE International Geoscience and Remote Sensing Symposium and 27th Canadian Symposium on Remote Sensing (IGARSS 2006)*, Denver, Colorado.

Zribi, M., Dechambre, M. (2002): A new empirical model to inverse soil moisture and roughness using two radar configurations. In: *Proceedings of the 2002 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2002)*, Toronto.

Dank

Mein allergrößter Dank gilt meiner geliebten Frau Jingjin Wang. Ohne ihre Geduld und Unterstützung wäre die Promotion kaum möglich gewesen. Weiterhin gilt mein Dank meiner Familie. Meine Mutter Waltraud und mein Vater Hans-Peter Balz (†) sowie meine Brüder Alexander und Peter haben mich immer unterstützt und mir letztlich diese Arbeit erst ermöglicht.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. habil. Dieter Fritsch, der mir die Möglichkeit gab am Institut für Photogrammetrie (ifp) der Universität Stuttgart zu arbeiten und der meine Arbeit als Hauptreferent betreute. Herrn Prof. Dr.-Ing. Uwe Stilla, der als Mitberichter die Arbeit betreute, gilt mein Dank für die fruchtbaren Diskussionen und hilfreichen Ratschläge während dieser Zeit. Herrn Prof. Dr.-Ing. Alfred Kleusberg danke ich für die Begutachtung der Arbeit als Mitberichter.

Dem gesamten Kollegium am Institut für Photogrammetrie gebührt ebenfalls mein Dank, denn die kollegiale und freundschaftliche Atmosphäre hat mir seit Beginn meiner Tätigkeit am ifp ein angenehmes und produktives Arbeiten ermöglicht. Dabei bedanke ich mich besonders bei Herrn PD Dr.-Ing. Norbert Haala, durch den ich zum Institut für Photogrammetrie kam und der meinen gesamten akademischen Werdegang nicht nur begleitet sondern ihn erst ermöglicht hat.

Weiterhin gilt mein Dank Peter Fischer, der mir die wesentlichen Grundlagen des Radars mit synthetischer Apertur beibrachte und der mir half die Welt mit „Radaraugen“ zu sehen. Den Kolleginnen und Kollegen des FGAN-FOM danke ich für die freundliche Zusammenarbeit die maßgeblich zum Entstehen dieser Arbeit geführt hat. Ein ganz besonderer Dank gebührt Herrn Prof. Dr.-Ing. Uwe Sörgel, der meine Arbeit von Anfang an unterstützte und der mir bei der Lösung zahlreicher Probleme geholfen hat.

Prof. Dr. Deren Li, Prof. Dr. Jianya Gong und Prof. Dr. Mingsheng Liao danke ich für ihre Unterstützung und für die freundliche Aufnahme an der Universität Wuhan. Ihre Hilfe ermöglichte mir die Weiterarbeit an meiner Promotion während meines Aufenthaltes am „State Key Lab of Information Engineering in Surveying Mapping and Remote Sensing (LIESMARS)“ der Universität Wuhan.

Lebenslauf

Timo Balz
geboren am 13. November 1974 in Stuttgart

Schulbildung

07/1981 – 02/1983	Grundschule in Stuttgart - Feuerbach
02/1983 – 07/1985	Grundschule in Steinheim a.d. Murr
07/1985 – 06/1994	Friedrich-Schiller Gymnasium in Marbach am Neckar
06/1994	Abitur

Wehrdienst

07/1994 – 07/1995	Grundwehrdienst

Studium

10/1995 – 10/2001	Studium der Geographie an der Universität Stuttgart
10/2001	Abschluss als Diplom-Geograph
09/2004 – 09/2005	Studium der Sinologie und Visiting Scholar am „State Key Lab of Information Engineering in Surveying Mapping and Remote Sensing (LIESMARS)” der Universität Wuhan (VR China)

Beruflicher Werdegang

seit 11/2001	Wissenschaftlicher Mitarbeiter am Institut für Photogrammetrie der Universität Stuttgart