11. Visualisierungstechniken

"***Visualisieren***" bedeutet soviel wie "sichtbar machen", oder auch "darstellen". Die Aufgabe der graphischen Datenverarbeitung beschränkt sich dabei jedoch nicht (wie die eines Photoapparates) auf die Sichtbarmachung von Objekten, die in unserer Alltagswelt materiell vorhanden sind bzw. sein könnten (vgl. Kapitel 1). Das folgende Kapitel beschäftigt sich zunächst mit der Visualisierung von Objekten dieser Art, und geht anschließend auf einige Techniken ein, die die graphische Darstellung von Objekten behandelt, die keinerlei oder nur in geringfügigem Maße geometrische Informationen beinhalten.

11.1. Visualisierung geometrischer Modelle

Wenn wir davon ausgehen, dass die Modelle die darzustellen sind 3-dimensionaler Natur sind, können wir zunächst eine einfache Unterscheidung treffen: Sollen bei der Visualisierung der Objekte die Gesetze der Optik (also die Einwirkung von Licht (Art, Farbe, Position der Lichtquelle) und die Eigenschaften des Materials (spiegelnd, brechend, transparent) eine Rolle spielen oder nicht. Lassen wir diese Einflüsse außer Betracht, wird auch eine komplexe Szene schneller darzustellen sein, als wenn Lichteinwirkungen eine Rolle spielen.

11.1.1. Visualisierung ohne Berücksichtigung von Licht

Diese Visualisierungmethode wurde bereits im Abschnitt 9.1 verwendet. Das Prinzip ist sehr einfach: Für eine Objektoberfläche (repräsentiert durch graphische Primitive) werden eine oder mehrere Farben definiert. Dies geschieht im Allgemeinen durch Angabe der RGB-Werte. In OpenGL erfolgt die Zuordnung der Farben zu Eckpunkten (vertices) (s. Abb. 11.1):

```
Beginn der Primitivdefinition
  Setzen einer aktuellen Farbe
  Definition des 1. Vertex
  .....
  Definition des n-ten Vertex
Ende der Primitivdefinition
```

Abb. 11.1: OpenGL - Primitiv mit einheitlicher Farbe

Handelt es sich bei dem Primitiv in Abb. 11.1 um GL_LINE_STRIP, so hat der gesamte Streckenzug eine einheitliche Farbe. Handelt es sich um GL_POINTS, so erhalten alle Punkte der Punktmenge die selbe Farbe. Ist das Primitiv z.B. GL_POLYGON, so wird die gesamte Fläche mit der einheitlichen Farbe ausgefüllt.

Nun wird die aktuelle Farbe aber den nachfolgenden Eckpunkten zugeordnet (s. oben). Was also, wenn das Primitiv 2 Vertices enthält, dem ersten davon die Farbe ROT, dem 2. die Farbe BLAU zugeordnet wird?

11.1.1.1. Lineare Interpolation

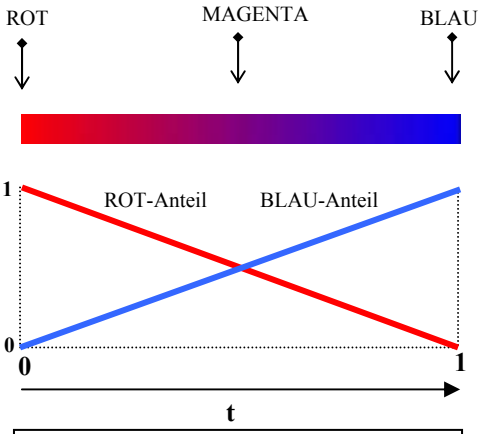


Abb. 11.2: Lineare Interpolation der Farbe

Die Antwort, die OpenGL auf die obige Frage gibt, lautet: ***Lineare Interpolation***. Dieses Prinzip soll am Beispiel der Abbildung 11.2 erläutert werden. Ausgehend vom 1. Eckpunkt nimmt der Rot-Anteil der Farbe zum 2. Eckpunkt hin linear ab. Gleichzeitig steigt der Blau-Anteil der Linie (von 0 beginnend) linear auf seinen Endwert 1.

Mathematisch wird dieser Sachverhalt wie folgt formuliert: $F(t) = (1-t) * F_0 + t * F_1$

Hierbei ist F(t) der Wert, der sich für eine Farbe an der Stelle t ergibt. Der Parameter t steht für die normierte Länge der Strecke und geht von t=0 bis t=1.

Beispiel: Die Farbe F_0 im ersten Punkt sei [R_0=1, G_0=0, B_0=0] und F_1des 2. Punktes (R_1=0, G_1=0, B_1=1) (vgl. Abb. 11.2), so ergibt sich

wie zu erwarten für den Startpunkt: (t=0): $F_0 = (1-0)*[1\ 0\ 0] + 0*[0\ 0\ 1] = [1\ 0\ 0]$ (rot),

ebenso für den Endpunkt: (t=1): $F_1 = (1-1)*[1\ 0\ 0] + 1*[0\ 0\ 1] = [0\ 0\ 1]$ (blau),

und für die Mitte der Strecke: (t=½): $F_{½} = (1-½)*[1\ 0\ 0] + ½*[0\ 0\ 1] = [\ ½\ 0\ ½]$
(Dies ist ein dunkles Magenta.)

Beispiel: Die Farbe im ersten Punkt sei [R_0=1, G_0=1, B_0=0] und die des 2. Punktes (R_1=0, G_1=1, B_1=1) (vgl. Abb. 11.2), so ergibt sich

wie zu erwarten für den Startpunkt: (t=0): $F_0 = (1-0)*[1\ 1\ 0] + 0*[0\ 1\ 1] = [1\ 1\ 0]$ (gelb),

ebenso für den Endpunkt: (t=1): $F_1 = (1-1)*[\ 1\ 1\ 0] + 1*[0\ 1\ 1] = [0\ 1\ 1]$ (cyan)

und für ein Viertel der Strecke: (t=¼): $F_{¼} = (1-¼)*[1\ 1\ 0] + ¼*[0\ 1\ 1] = [\ ¾\ 1\ ¼]$

Dies ist ein ... (ja was wohl? Probieren Sie es aus!)

11.1.1.2. Bilineare Interpolation

Als nächstes stellt sich die Frage, wie die Farbe im Inneren eines Polygons ermittelt wird, wenn sich die Farben der Eckpunkte unterscheiden. Erinnern wir uns daran, dass Graphiksysteme Flächen zum Zweck der Darstellung i.a. in Dreiecke zerlegen. Wir werden daher die Lösung des Problems am Beispiel eines Dreiecks betrachten. Hierbei ist es nützlich sich an die Vektor-Rasterkonvertierung von Dreiecken zu erinnern.

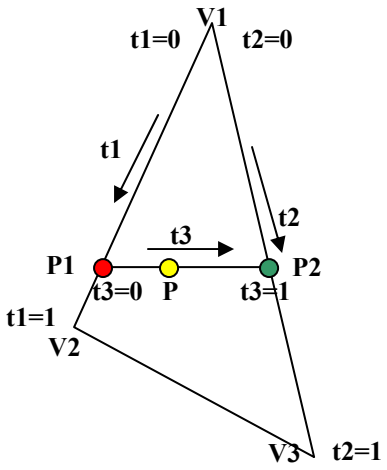


Abb. 11.3: Bilineare Interpolation der Farbe

Die Berechnung der Farbe eines Pixels im Dreieck erfolgt in 2 Stufen (Abb. 11.3):

1. Stufe: Bei der Konvertierung wird das Dreieck zeilenweise (beginnend mit kleinen y-Werten) bearbeitet. Durch lineare Interpolation wird die Farbe in den Punkten P1 (Parameter t1 zwischen den Vertices V1 und V2) und P2 (Parameter t2) ermittelt (s.o.). Hierzu sind also 2 lineare Interpolationen entlang zweier Dreiecksseiten erforderlich.

2. Stufe: P1 und P2 (deren Farben ja nun bekannt sind) werden linear interpoliert (Parameter t3). Hieraus ergibt sich die Farbe jedes Punktes P auf der Scanzeile.

Man nennt dies eine ***bilineare Interpolation***, da nach der Interpolation auf der Kontur (lineare Interpolation) ein Übergang in das Innere der Fläche der erfolgt

11.1.2. Rendering

Im Folgenden soll kurz über Techniken gesprochen werden, bei denen die Beleuchtungsverhältnisse und die Art und Weise wie Materialien darauf reagieren eine Rolle spielen. Die (möglichst realistisch wirkende) Visualisierung einer derartigen Szene wird als ***Rendern*** bezeichnet. Unter einem ***Renderer*** versteht man dann die Software oder Hardware, die die Ausführung übernimmt. Die Erstellung qualitativ hochwertiger Bilder für die Produktion kann bis zu mehreren Stunden dauern. Echtzeit - Renderer erzeugen Bilder in Bruchteilen von Sekunden. Als ***Rendering Pipeline*** bezeichnet man die Folge der Einzelschritte, die erforderlich ist, um ein realistisches Bild zu erzeugen. Dies beinhaltet die Umwandlung von Weltkoordinaten

in Bildschirmkoordinaten, die Berechnung der Beleuchtung, Aufbringung von Texturen und die Vektor/Rasterkonvertierung.

Der Eindruck, den ein Betrachter vom Aussehen einer bestimmten Stelle einer Objektoberfläche gewinnt, hängt von den verschiedensten Faktoren ab. Zum einen spielt dabei die relative Lage und Orientierung von Betrachter und Objekt eine Rolle, zum anderen ist die Art der Beleuchtung und die Beschaffenheit der Objektoberfläche entscheidend. Um realistische Darstellungen zu erzielen, müssen diese Gegebenheiten modelliert werden (optische Modellierung). Weiterhin spielt für den visuellen Eindruck auf der Bildschirmoberfläche das Verfahren eine Rolle, das zum Rendern der Objekte eingesetzt wird.

In diesem Rahmen kann über optische Modellierung nicht gesprochen werden. Dennoch ist es möglich, die wichtigsten Rendering - Techniken in ihrer prinzipiellen Arbeitsweise vorzustellen. Um von einem Objekt - im Folgenden ist damit immer ein Teil seiner Oberfläche (i.a. eine dreieckige ***Facette***) gemeint - einen realistischen Eindruck zu erzeugen, wollen wir voraussetzen, dass folgende Einflussgrößen bekannt sind:

Objekt: Position, Ausrichtung, Materialeigenschaften,...
Lichtquellen: Lage, Strahlrichtung, Lichtfarbe und Intensität,...
Betrachter: Position, Blickrichtung

Man kann sich vorstellen, dass die Vielfalt dieser Daten, in teilweise umfangreichen Formeln miteinander verknüpft, zu aufwendigen Berechnungen führen kann - ein Grund, weshalb trotz schneller Hardware, nach wie vor nach neuen effizienten Verfahren zum Rendern komplexer Szenen gesucht wird. Die erste Frage beim Rendern sollte daher immer sein: Welches Verfahren ist anwendbar, um mit dem geringsten Aufwand ein qualitativ akzeptables Ergebnis zu erzielen? Eine Möglichkeit den Aufwand von vornherein zu begrenzen besteht in der Anwendung sogenannter ***lokaler Verfahren***:

Diese Verfahren berücksichtigen bei der Berechnung des von einer Objektoberfläche reflektierten Lichtes ausschließlich a) das von den definierten Lichtquellen einfallende Licht und b) die Eigenschaften dieser Oberfläche. Die einzelnen Objekte in der Szene stehen hierbei nicht in Wechselwirkung miteinander. Dies bedeutet, dass keine gegenseitige Beeinflussung der Objekte durch Spiegelung, Schattenwurf und Lichtreflexionen stattfindet.

Im Gegensatz hierzu stehen die ***globalen Verfahren***, bei denen sich die Objekte in der Szene gegenseitig beeinflussen. Hier können Spiegelungen, Schattenwurf und Reflexionen von Licht direkt umgesetzt werden. Man kann diesen Sachverhalt auch dadurch ausdrücken, dass neben den sogenannten ***primären Lichtquellen*** (direktes Licht) auch ***sekundäre Lichtquellen*** (also durch Reflexionen von Licht an anderen Objekten verursacht) berücksichtigt werden. Sollen bei Verwendung lokaler Verfahren derartige Effekte erzielt werden, ist dies allerdings über "Tricks" und Umwege auch möglich.

Die lokalen Verfahren sind historisch älter, da sie weniger rechenaufwendig sind. Zu ihnen gehören Schattierungsverfahren wie Gouraud Shading und Phong Shading. Zu den globalen Verfahren zählt das rekursive Ray Tracing und das Radiosity Verfahren.

.

11.1.2.1. Lokale Verfahren

Im Folgenden wird von ***Shading - Verfahren*** die Rede sein. Der Begriff Shading wird im Deutschen mit Schattierung übersetzt (also: ***Schattierungsverfahren***) und hat nichts mit dem ähnlich klingenden Begriff shadow = Schatten zu tun. Mit Schattierung meint man das Einfärben einer Facettenoberfläche entsprechend den Material-, Beleuchtungs- und Betrachtungsverhältnissen.

Flat Shading

Flat Shading, auch ***Constant Shading*** genannt**,** ist das einfachste Verfahren dieser Art. Die Facette erhält dabei eine einheitliche Farbe. Diese wird z.B. für eine Ecke der Facette ermittelt und dann auf alle Punkte des Inneren der Facette übertragen. Da gekrümmte Oberflächen i.a. durch ebene Facetten approximiert werden (s. Abb. 11.4) liefert dieses einfache Verfahren nur bedingt qualitativ befriedigende Ergebnisse. Die Voraussetzungen für brauchbare Ergebnisse sind gegeben, wenn

- die Lichtquelle sehr weit entfernt ist,
- die Betrachterin von der Facette weit entfernt ist,
- die Oberfläche eben ist.

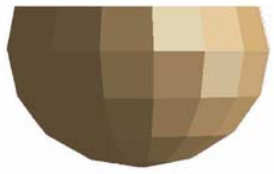


Abb. 11.4: Flat Shading

Gouraud Shading

Das weitverbreitete ***Gouraud Shading*** (sprich: Gur'ooh), ist nicht viel aufwendiger als Flat Shading, liefert jedoch schon erheblich bessere Resultate (s. Abb. 11.5).

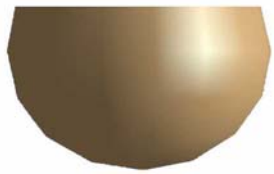


Abb. 11.5: Gouraud Shading

Das Verfahren arbeitet wie folgt:
Die Farbe (Helligkeit) wird in den Eckpunkten der Facette (des Dreiecks) berechnet. Dies erfolgt (wie beim Flat Shading) unter Anwendung der Formeln der optischen Modellierung. Der einzige Unterschied besteht darin, dass statt eines (richtigen) Wertes für die Facette jetzt an 3 Stellen (bei einem Dreieck) "exakte" Werte errechnet werden. Damit diese Aufgabe erfüllbar ist, müssen in den Ecken Normalenvektoren vorgegeben werden. Man bezeichnet diese als ***Eckennormalen***. Zum Unterschied zwischen Eckennormalen und einer ***Flächennormalen*** siehe Abbildung 11.6.

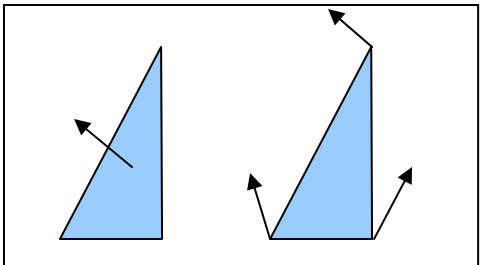


Abb. 11.6: Ecken- und Flächennormalen

Sind die Farben (Helligkeiten) in den Ecken berechnet, so werden die Werte für die übrigen Pixel durch bilineare Interpolation errechnet. Nehmen die Eckennormalen unterschiedliche Werte an (sinnvollerweise sind das die Normalen der Tangentialebenen in diesen Punkten, vgl. Abb. 11.7.), so ergibt sich ein erstaunlich gut glättender Effekt (s. Abb. 11.5.) Die Geometrie weist ja wirklich nur eine geringe Anzahl von Facetten auf - vgl. Abb. 11.4). Beachten Sie, dass die Geometrie unverändert bleibt, also nicht geglättet wird.

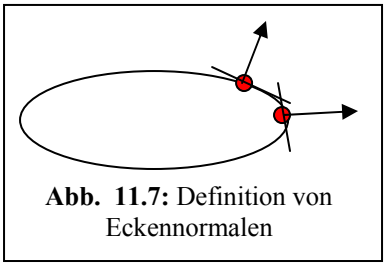


Abb. 11.7: Definition von Eckennormalen

Die Grenzen des Gouraud Shadings werden klar, wenn man sich. überlegt welchen Effekt z.B. ein stark gebündeltes Licht, das auf das Innere der Facette gerichtet ist, hervorruft. Wenn die 3 Ecken kein Licht erhalten ist ihre Helligkeit 0. Werden diese 3 Helligkeiten interpoliert, so bleibt natürlich auch das Innere der Facette dunkel. Eine übliche Abhilfe ist, die gesamte Objektoberfläche feiner zu unterteilen. Damit steigt natürlich auch entsprechend der Rechenaufwand.

Phong Shading

Ein wesentlich rechenaufwendigeres Verfahren stellt das ***Phong Shading*** dar. Der Unterschied zum Gouraud Shading besteht darin, dass dabei für jedes Pixel der Facette die optischen Gesetze Anwendung finden. Hierzu ist es erforderlich, für jedes der Pixel die Normale zu kennen. Diese wird aus den Eckennormalen der Facette ermittelt - wer hätte das für möglich gehalten? Dies geschieht durch bilineare Interpolation der Eckennormalen! Mit diesem Verfahren sind auch sogenannte Glanzlichter darstellbar. OpenGL unterstützt Flat - und Gouraud Shading.

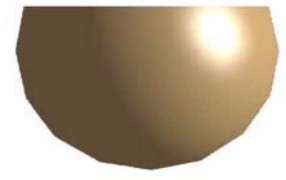


Abb. 11.8: Phong Shading

11.1.2.2. Globale Verfahren

Strahlverfolgung (Ray Tracing)

Mit diesem Verfahren können 'relativ einfach' der Schattenwurf von Objekten, sowie spiegelnd reflektierende und transparente Oberflächen behandelt werden. Die Strahlverfolgung stellt eine aufwendige 'Geradeausmethode' zur Erzeugung sehr realistisch wirkender Darstellungen dar.

Die Grundidee hierbei ist, den Wegen der Lichtstrahlen zu folgen, die die Szene erhellen. Dabei wird berechnet, wie die Strahlen an Oberflächen absorbiert, reflektiert und gebrochen werden. Einige dieser Lichtstrahlen werden am Ende am Blickpunkt (COP) eintreffen und das endgültige Bild erzeugen. Da die meisten Lichtstrahlen die Betrachterin niemals erreichen, wäre es ein unnötig hoher Berechnungsaufwand, die physikalischen Strahlen von der Lichtquelle aus zum Auge hin zu verfolgen (***Forward Ray Tracing***). In der Praxis verfolgen wir in umgekehrter Richtung, ausgehend vom COP, den Sehstrahl auf seinem Weg durch jedes Pixel der Projektionsfläche hindurch. Irgendwann trifft dieser Strahl auf das erste Objekt oder er verschwindet im Hintergrund (***Backward Ray Tracing***).

Abb. 11.9 zeigt dieses Prinzip am Beispiel dreier vom COP ausgehender Sehstrahlen. Beachte, dass hier noch keine gegenseitige Beeinflussung der Objekte statt findet. Andererseits wird das Problem der gegenseitigen Verdeckung von Objekten bereits auf sehr natürliche Art gelöst. Der vom Auge ausgehende Sehstrahl schneidet zwar möglicherweise mehrere Objekte in der Szene, maßgeblich für die Sichtbarkeit ist jedoch (bei nichttransparenten Objekten) ausschließlich der erste dieser Schnittpunkte. Mathematisch kann dieser sehr einfach ermittelt werden.

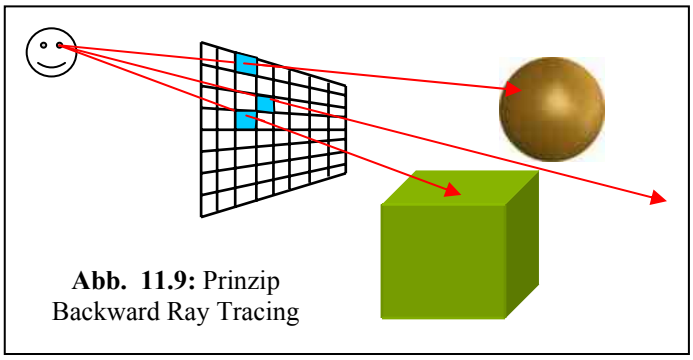


Abb. 11.9: Prinzip Backward Ray Tracing

Die Helligkeit eines Oberflächenpunktes (Schnittpunkt des Sehstrahls mit Objekt) hängt nicht nur von direkten Lichtquellen ab, die diesen Punkt beleuchten, sondern auch von anderen Objekten, die sich möglicherweise in diesem Punkt spiegeln oder, bei Transparenz des Objekts, durchscheinen. Re-

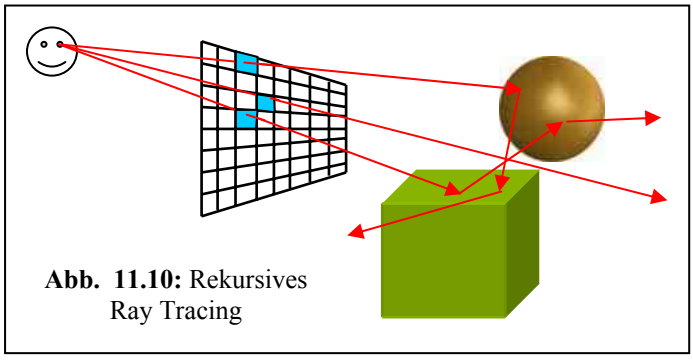


Abb. 11.10: Rekursives Ray Tracing

flektierte bzw. gebrochene Strahlen werden im Gegensatz zu den vom COP ausgehenden ***Primärstrahlen*** als ***Sekundärstrahlen*** bezeichnet. Die Verfolgung von Sekundärstrahlen führt zum sog. ***rekursiven Ray Tracing***. Abb. 11.10 zeigt den prinzipiellen Unterschied zum so genannten ***Ray Casting***, dem nicht rekursiv arbeitenden Ray Tracing (Abb. 11.9).

Gegenüber den Shading Verfahren lassen sich folgende Vorteile des Verfahrens ausmachen:
- Lichteffekte können fast alle realistisch simuliert werden.
- Das Verdeckungsproblem wird elegant (nebenbei) gelöst. Dies gilt auch in Fällen, wo dies bei den Schattierungs - Algorithmen nur mit Schwierigkeiten möglich ist.

Folgende Nachteile des Verfahrens bestehen gegenüber den Shading Verfahren:
Es ergeben sich (vor allem für das rekursive Ray Tracing) hohe Rechenzeiten, die abhängig sind von
- der Bildauflösung (es handelt sich um ein bildraumorientiertes Verfahren - vgl Abb. 11.9!)
- der Komplexität der Szene,
- der gewünschten Bildqualität

Der durchschnittlicher Aufwand für eine einfache 3D - Szene verteilt sich etwa wie folgt:
- Verwaltung: 13 %
- Schnittpunktberechnung: 75 % (bei komplexen Szenen bis 95 %)
- Helligkeitsberechnungen: 12 %

Eine Optimierung setzt daher sinnvoll bei der Schnittpunktberechnung an. Das bedeutet u.a., die Anzahl der erforderlichen Schnittpunktberechnungen zu minimieren und Schnittpunktberechnungen mit komplizierten Oberflächen durch solche mit einfachen Oberflächen zu ersetzen. Hierzu dient z.B. die Definition geeigneter Begrenzungskörper (***bounding boxes*** s. Kapitel 2.2.2. und Abb. 2.9) und entsprechender Hierarchien.

Radiosity

Das ***Radiosity*** Verfahren zur Schattierung von Objekten ist noch jünger als das Ray Tracing. Im ursprünglichen Ansatz konnten nur diffus reflektierende Oberflächen bearbeitet werden. Heute sind hybride Verfahren entwickelt worden, die Ray Tracing und Radiosity verbinden.

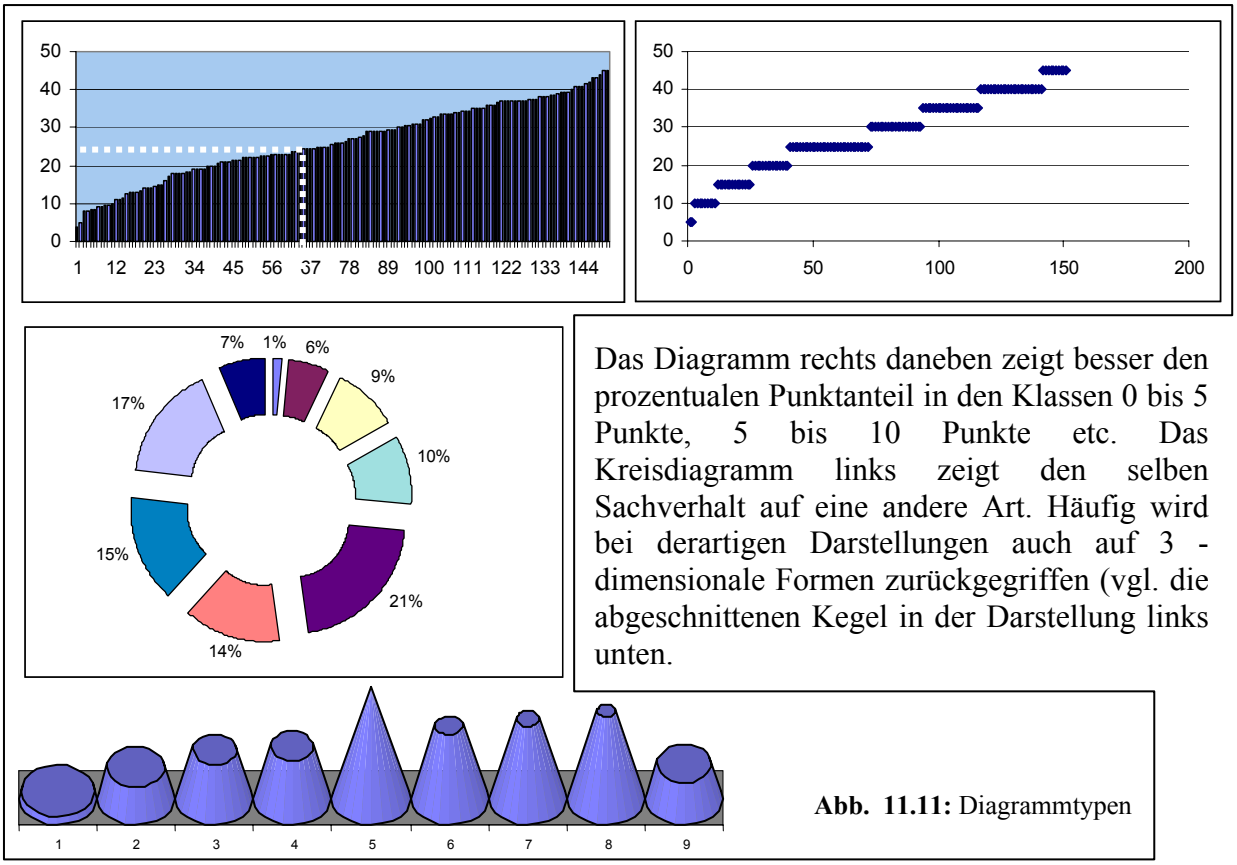
Prinzip: Als Annahme und Einschränkung gilt, dass alle Oberflächen ideale diffuse Reflektoren sind. Das bedeutet, dass die reflektierte Lichtintensität in allen Richtungen gleich ist. Das physikalische Verhalten des Lichtes wird mit Hilfe der Gesetze der Thermodynamik beschrieben (Energietransport und Energieerhaltung). Damit wird die Interaktion von Licht zwischen diffusen Oberflächen berücksichtigt.

Soll die auf eine Oberfläche auftreffende Lichtenergie berechnet werden, so muss hierzu die Strahlung aus allen Richtungen des Raumes berücksichtigt werden, die die Oberfläche erreicht. Jede Fläche wird in Teilflächen zerlegt, die sich bzgl. ihrer Eigenschaften einheitlich verhalten:
- die Teilfläche ist ein idealer diffuser Reflektor, oder
- die Teilfläche ist eine ideale diffuse Lichtquelle, oder
- die Teilfläche ist eine Kombination aus beiden.

Bei Verwendung einer direkten Lichtquelle werden die Flächen bestimmt, die durch diese beleuchtet werden. Das durch die Flächen reflektierte Licht wird als diffuse Lichtquelle behandelt (sekundäre Lichtquelle).

11.2. Visualisierung nichtgeometrischer Modelle

Dieser Abschnitt soll mit einem einfachen Beispiel beginnen. Abbildung 11.11 zeigt die Statistik über die erzielten Punkte im GDVI - Kurs für Bachelor - Studierende im WS 2000/2001. An der Klausur haben 152 Studierende teilgenommen. Es konnten 45 Punkte erworben werden. Die zu Grunde liegenden Daten beinhalten keinerlei geometrische Informationen. Gleichzeitig zeigt das Beispiel den Vorteil einer graphischen Visualisierung. Auf einen Blick kann dem Diagramm links oben entnommen werden, dass etwa 66 der 152 Studierenden ihr Ziel, einen Schein zu erhalten vermutlich nicht erreichen werden.



Das Diagramm rechts daneben zeigt besser den prozentualen Punktanteil in den Klassen 0 bis 5 Punkte, 5 bis 10 Punkte etc. Das Kreisdiagramm links zeigt den selben Sachverhalt auf eine andere Art. Häufig wird bei derartigen Darstellungen auch auf 3 - dimensionale Formen zurückgegriffen (vgl. die abgeschnittenen Kegel in der Darstellung links unten.

Abb. 11.11: Diagrammtypen

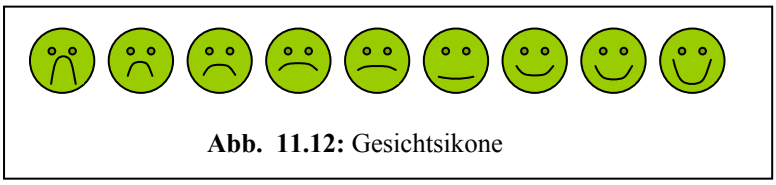


Abb. 11.12: Gesichtsikone

Abbildung 11.12 zeigt den selben Sachverhalt auf andere Art. Man spricht hier von einer ikonenbasierten Visualisierung von Daten. Hierunter versteht man die Abbildung von Datenwerten auf graphische Primitive. Letztere bezeichnet man als "***Ikonen***" oder "***Glyphen***". Wie aus der Darstellung hervorgeht, werden dabei die zu visualisierenden Werte auf einzelne Elememte des Primitivs abgebildet. Im obigen Fall sind es die "Wertigkeit" der einzelnen Klassen auf die Form des Mundes).

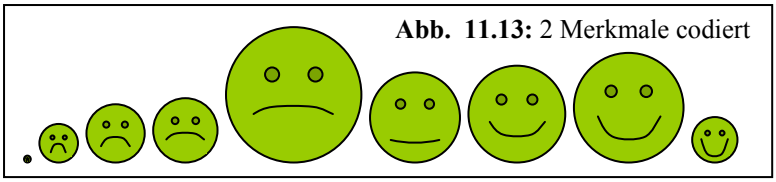


Abb. 11.13: 2 Merkmale codiert

Ikonenbasierte Techniken eignen sich auch zur Visualisierung mehrerer Parameterwerte auf einmal. Dies ist am Beispiel der Abbildung 11.13 zu ersehen. Hier wurde die Anzahl der Elemete pro Klasse in die Größe der Ikone codiert. Offensichtlich eignet sich eine derartige Darstellung recht gut, um Zusammenhänge zu verdeutlichen. Vergleiche hierzu die Aussagekraft der Darstellungen in Abbildung 11.11 und 11.13.

Multiparameterdaten (also mehr als ein Parameter) werden auch als ***mehrdimensionale Daten*** oder auch ***multivariate Daten*** bezeichnet. Die geeignete, also anschauliche Visualisierung solcher Daten ist ein eigener Wissenschaftszweig. Ohne auf die einzelnen Techniken einzugehen, folgen an dieser Stelle noch einige Beispiele:

Abb. 11.14 zeigt eine sog. ***Chernoff-Ikone***. Dies ist ein häufig verwendeter Glyph in dem bis zu 12 Merkmale eines menschlichen Gesichts verwendet und codiert werden.

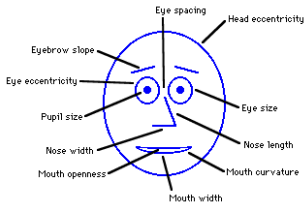


Abb. 11.14: Chernoff - Ikone mit 11 Merkmalen

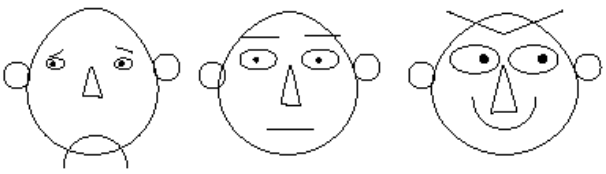
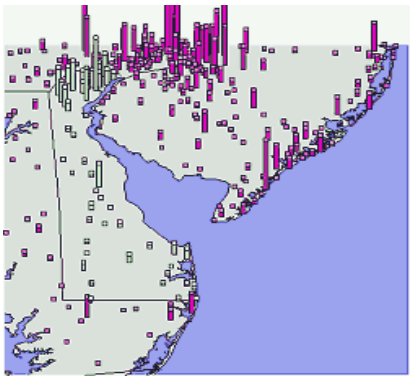
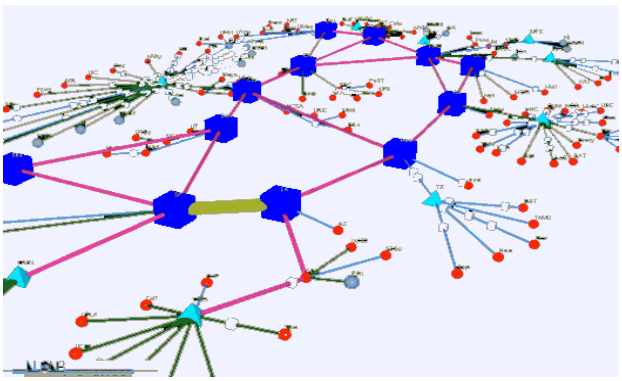
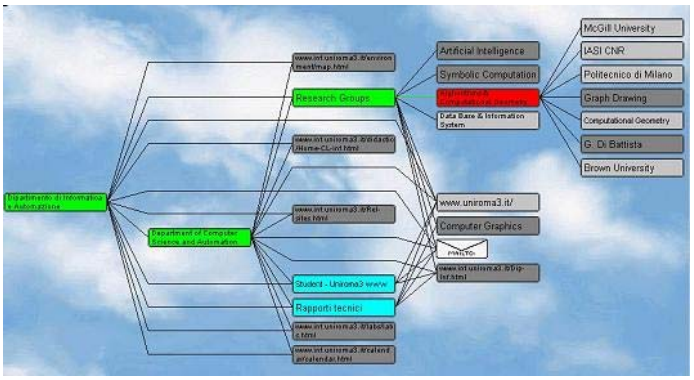


Abb. 11.15: Chernoff Gesichter mit 4 Merkmalen

Abb. 11.15 zeigt dazu ein Beispiel, bei dem 4 verschiedene Merkmale codiert wurden. Die Abbildungen 11.16 bis 11.19 zeigen abschließend noch 4 Beispiele von Visualisierungstechniken aus dem Bereich "Cybergeographie"







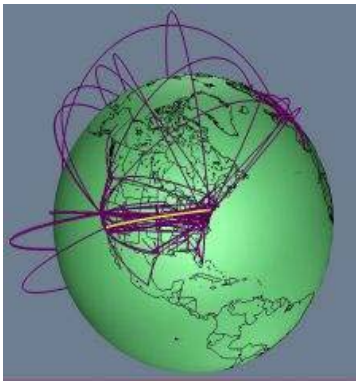


Abb. 11.16-20 Beispiele: Cybergeographie