Visualisering af lampers belysning

Gruppe B2-28

Morten Rask Andersen

Anton Christensen

Lasse Fribo Gadegaard

Christian Mønsted Grünberg

Mathias Ibsen

Mathias Rohde Pihl

18/12-2015

Aalborg Universitet

Software, 1. semester

AALBORG UNIVERSITET
STUDENTERRAPPORT

Første Studieår
Software
Strandvejen 12-14
9000 Aalborg
http://tnb.aau.dk

Titel: Visualisering af lampers belysning

Tema: Raytracing
Projektperiode: P1, Efterårssemesteret 2015

Projektgruppe: B2-28
Deltagere:
Morten Rask Andersen
Anton Christensen
Lasse Fribo Gadegaard
Christian Mønsted Grünberg
Mathias Ibsen
Mathias Rohde Pihl

Vejledere:
Hovedvejleder:
Benjamin Bjerre Krogh

Bivejleder:
Anette Grunwald

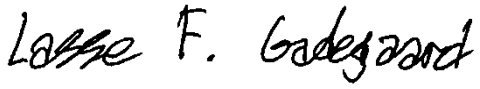
Synopsis:

Med udgangspunkt i det initierende problem "Kunden kan ikke visualisere, hvordan lys udbreder sig fra en lampe uden først at købe og installere lampen", analyseres relevansen, begreberne, interessenterne og det initierende problems placering. Ud fra dette afgrænses der i problemformuleringen til det overordnede spørgsmål "Hvordan kan vi lave et værktøj til e-butikker, som visualiserer indendørslampers belysning for kunderne". Herefter præsenteres et løsningsforslag, hvoraf krav til løsningen opstilles. I teoridiskussionen beskrives rotationsmatricer, konvertering af farvetemperatur til RGB, samt dele af teorien til udvikling og optimering af en raytracing algoritme. Denne teori anvendes til at udvikle et værktøj som vha. raytracing renderer et billede af en lampe og dens belysning med brugerbestemt synsvinkel og farvetemperatur. Koden bag værktøjet dokumenteres og testes. På baggrund af diskussion konkluderes der, at der i projektet er taget de første skridt mod at udvikle et værktøj, der visualiserer en lampe og dens belysning, når kunder handler på en e-butiks hjemmeside.

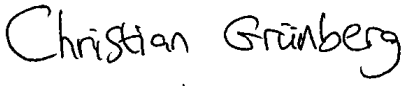
Oplagstal: 9
Sideantal: 78
Bilagsantal og -art: 5 x papir
Afsluttet den: 18/12-2015

Morten Rask Andersen

Anton Christensen



Lasse Fribo Gadegaard



Christian Grünberg

Mathias Ibsen

Mathias Pihl

1 Forord

Denne rapport er udarbejdet af gruppe B2-28, bestående af software-studerende, som P1-rapport på Aalborg Universitet.

Rapporten tager udgangspunkt i Aalborg-modellen for problembaseret læring. Denne læringsproces har givet gruppen mulighed for at undersøge en given problemstilling og derudfra tilegne sig viden, og på baggrund af denne viden udarbejde en problemanalyse. Derudover gør rapporten brug af den kvalitative metode til korrespondance med interessenterne. Den kvalitative metode er fordelagtig at bruge, når man vil undersøge forhold, som er svære at iagttage eller måle [1]. I forbindelse med problemanalysen har vi haft kontakt med to lampedesignere og en belysningskonsulent for en dansk lampebutik. Belysningskonsulenten vil efter eget ønske fremgå anonymt.

Tak til vejledere Benjamin Bjerre Krogh og Annette Grunwald samt den medvirkende belysningskonsulent og de medvirkende designere.

Rapporten er blevet afleveret på papirform og online på Digital Eksamen, hvor programmet er blevet vedhæftet. Koden kan derfor findes online og betragtes som elektronisk bilag. Farvede billeder kan derfor også ses på online udgaven af rapporten.

Som en del af projektet er der i rapportens appendiks udarbejdet et projektforslag til næste års P1-forløb.

1.1 Læsevejledning

Hvert afsnit X.X har sin egen indledning og opsummering hhv. først og sidst i afsnittet. Kilder og kodeuddrag er angivet i rapporten, som beskrevet i nedenstående afsnit.

1.1.1 Kildehenvisning

Rapportens brug af kildehenvisninger er baseret på nummermetoden [2]. I nummermetoden anføres kilderne i fortløbende nummerorden, svarende til hvilket nummer, de har i teksten. To identiske kilder har samme nummer. Herunder ses et eksempel på henvisninger til hhv. internetkilder og bøger:

Interneteksempel med kilde[1].

[1] Titel på emne eller kort forklaring på emnet, hjemmesidenavn. Set DD-MM-YYYY. URL på hjemmeside.

Bogeksempel med kilde[2].

[2] Titel på bog, udgavenummer, forfatter(e), udgivelsesår. ISBN/ISSN-nummer.

Hvis en kilde har yderligere relevante informationer (såsom sidetal, ophavsret mm. angives disse også i kilden).

Figurhenvisning foregår på samme måde som med andre kilder, dog med en forklaring under selve figuren. Hvis en figur ingen kilde har, er figuren fremstillet af gruppen.

1.1.2 Kodeuddrag

Flere steder i rapporten, vil der blive vist dele af gruppens kode. Et eksempel på hvordan dette vil blive vist er herunder:

```
1 #include <stdio.h>
2
3 int main(void){
4    printf("Hello, world!\n");
5    return 0;
6 }
```

Kodeuddrag 1: Kodeeksempel i C

Indholdsfortegnelse

Figurliste

2 Indledning

Selvom vi ikke tænker på det særligt ofte, er lamper en stor del af vores hverdag. De står i vores hjem, på vores gade og på vores arbejdspladser - de er stort set overalt. Men hvorfor er lamper så udbredte? Det er de, fordi lamper bliver brugt til at skabe lys. Belysning kan bidrage til mange ting som at læse i mørke omgivelser, arbejde mere koncentreret eller til at skabe hygge og stemning i et rum, der ellers ville have været koldt og kedeligt.

Lamper findes i mange forskellige typer og mange forskellige steder. Der er læselamper, arbejdslamper, loftlamper, udendørslamper m.fl. og de tjener alle forskellige formål, men fælles for dem er, at de skaber lys, hvor der ellers ikke ville have været lys.

Da belysning fra lamper fylder en stor del i rigtig mange menneskers hverdag, er det derfor interessant at undersøge de konsekvenser, der kan opstå, når en lampe ikke passer ind der, hvor den bruges, samt at udvikle en løsning på dette problem. Problemet kan være irritation over en blændende lampe, men endnu værre kan der opstå sundhedsmæssige konsekvenser, af belysning, fra lamper der ikke er optimale i forhold til konteksten (Omtalt i afsnit 3.1.2).

Da vi mere eller mindre, kan spore disse konsekvenser tilbage til købet af lampen, er det netop købet af lampen, som er udgangspunktet for vores undersøgelse og løsning.

Med dette udgangspunkt har vi, som studerende på AAU Software, taget kontakt til 10 lampebutikker (Bilag D) for, at få mere viden inden for området. Herefter afgjorde vi, at det er mangel på visualisering af lys fra lamper, som er et stort problem i forhold til købet. Dette afgør nemlig, om forbrugeren kan se hvordan lyset breder sig fra en lampe, og det er derfor vigtigt, når brugeren skal vælge en lampe. På baggrund af vores viden om lys fra lamper, samt de erfaringer og diskussioner vi har foretaget os som gruppe, har vi valgt følgende initierende problem:

2.1 Initierende problem

Kunden kan ikke visualisere, hvordan lyset udbreder sig fra en lampe uden først at købe og installere lampen.

2.2 Rapportens struktur og metodiske overvejelser

Rapportstrukturen afspejler den metodiske tilgang til projektet. Rapportens afsnit er struktureret i kronologisk rækkefølge fra problemanalysen til perspektiveringen (punkt 3 - 9). Rapportstrukturen ses nedenfor:

3. Problemanalyse: En analyse af problemfeltet for, at få en dybere forståelse for det initierende problem. Afsnittet indeholder følgende underafsnit:
    - Relevans: Argumenterer for relevansen af det initierende problem.
    - Begrebsliggørelse: Redegører for begreber til forståelse af det initierende problem.
    - Interessentanalyse: Undersøger interessenterne og hvilken målgruppe rapporten vil arbejde videre med. I afsnittet har gruppen anvendt den kvalitative undersøgelse til bl.a. at samarbejde med erhvervslivet heraf designere og en belysningskonsulent for en dansk lampebutik. Formålet med interessentanalysen er at finde ud af hvilke interessenter, denne rapport vil løse problemet for.
    - Problemets placering: Undersøger det initierende problems placering, samt hvor rapporten vil fokusere på at løse problemet.
4. Problemformulering: Formulerer det problem, som skal løses.
5. Løsningsdesign: Gennemgår gruppens løsningsforslag af problemet samt hvilke teknologier der er til visualisering. Afsnittet indeholder følgende underafsnit:
    - Løsningsforslag: Skitserer et løsningsforslag og opstiller krav til løsning.
    - Teknologier til visualisering: Beskriver hvilke teknologier, der er til at visualisere en lampe og dens belysning.
6. Problemløsning: Undersøger hvilke teorier, der skal til for at løse problemet, samt implementationen og test af løsningen. Afsnittet indeholder følgende underafsnit:
    - Teoridiskussion: Undersøger den teori, som skal til for at lave løsningen.
    - Udvikling: Dokumenterer udviklingen af løsningen.
    - Test af programmet: Tester om løsningen lever op til de krav som blev sat til løsningsforslaget.
7. Diskussion: Diskuterer hvorvidt løsningen opfylder de krav, som blev sat til løsningsforslaget
8. Konklusion: Opsummerer hvordan den endelige problemformulering blev udarbejdet. Derudover konkluderes der hvorvidt den udviklede løsning besvarer den endelige problemformulering.
9. Perspektivering: En diskussion af hvad der skal til for at arbejde videre med løsningen, samt alternative anvendelsesmuligheder for løsningen.

3 Problemanalyse

I dette afsnit er formålet, at få en dybere forståelse for det initierende problem. Dette gøres ved at foretage en analyse af problemfeltet, hvor der ønskes svar på en række hv-spørgsmål. Hvorfor er problemet relevant? Hvilke begreber indgår i problemet? Hvilke interessenter er relevante ift. problemet? Hvor eksisterer problemet? Svaret på disse spørgsmål danner til sidst fundamentet for den endelige problemformulering, og er med til at give en forståelse for problemfeltet senere i rapporten.

3.1 Relevans

Rapporten vil i dette afsnit besvare om problemet er relevant og i så fald, hvorfor det er relevant. I denne sammenhæng lægges der blandt andet fokus på hvordan lyset påvirker mennesker, samt hvilke konsekvenser dårlig belysning kan medføre.

3.1.1 Lysets påvirkning på mennesket

Med udgangspunkt i udvalgte artikler og bogen "Human Factors in Lighting" af Peter R. Boyce, undersøges der hvilke konsekvenser dårlig belysning kan have på mennesker.

De fleste problemer opstår, hvis man sidder for længe i lys som øjnene opfatter som værende ubehageligt eller forstyrrende [3]. Den mest normale konsekvens af dette er overanstrengelse af øjnene [3]. Symptomerne på overanstrengelse af øjnene er: irritation af øjnene som viser sig som betændelse omkring øjne og øjenlåg [3]. Et andet symptom er forringet syn, som kan opleves som dobbeltsyn og sløring af synet [3]. Derudover kan der forekomme bivirkninger som bl.a. hovedpine, forstoppelse og svimmelhed [3].

Konsekvenser ved dårlig belysning samt de efterfølgende bivirkninger ved det opfattes forskelligt fra menneske til menneske [3]. Nogle mennesker er særligt sårbare overfor dårlig belysning, heraf den gruppe som har fotoepilepsi [3]. Her kan ubehageligt og flimrende lys forårsage anfald [3].

Et problem for især ældre mennesker er, at de risikerer at vælte, da deres syn er blevet forringet [3]. Dette har formindsket deres balance og evne til at vurdere afstande [3]. God belysning kan være med til at styrke de sanser som holder kroppen i balance. Heraf kan en passende natlampe bl.a. være en løsning om natten [3].

Dårlig belysning kan have negative konsekvenser for mange forskellige faktorer i en kontorarbejders hverdag. Det kan f.eks. føre til overanstrengelse af øjet, hovedpine og træthed [4]. Det fysiske og psykiske ubehag kan formindske kontorarbejderens produktivitet, og det er derfor i virksomhedens bedste interesse at komme dette problem til livs. Ifølge OSHA [4], estimerer nogle undersøgelser, at op til 90% af de 70 millioner amerikanske arbejdere, der benytter en computer i deres arbejdstid, i mere end tre timer, oplever 'computer vision syndrome', som er betegnelsen for synsproblemer forudsaget af belysningen fra en computerskærm [5]. Ifølge artiklen "The Er-

gonomics of Light" kan den rigtige arbejdslampe ift. arbejdskonteksten øge bekvemmeligheden, produktiviteten og moralen [4].

3.1.2 Argumentation for problemets relevans

For at svare på, hvorfor problemet er relevant antages nu, at det initierende problem eksisterer. Det er svært at bevise, at denne antagelse er korrekt, men antagelsen er lavet efter en diskussion med Lars Peter Jensen, lektor på AAU, som netop havde erfaret, at han havde svært ved at visualisere hvordan lys fra en bestemt lampe ville se ud i sit hus, før han havde installeret lampen. Antagelsen understøttes af en udtalelse fra en belysningskonsulent for en dansk lampebutik som siger følgende: "Der er overraskende mange, der gerne vil se lyset inden de køber lamper"(se bilag D). Dette er en erfaring som flere i gruppen også har gjort sig. Ud fra en diskussion, har gruppen valgt at arbejde videre med antagelsen, da det formodes at andre mennesker har haft lignende problem. Denne formodning ønskede gruppen at teste yderligere ved en spørgeskemaundersøgelse i IKEA Aalborg, hvor tanken var at spørge kunderne om de kunne visualisere hvordan lys udbredte sig fra de lamper som de så i butikken. Gruppen henvendte sig derfor til IKEA i Aalborg for at høre om det var muligt at møde op i deres butik for at uddele spørgeskemaer, men gruppens henvendelse blev afvist (Se bilag C). Spørgeskemaet er vedhæftet i bilag E.

På baggrund af antagelsen formoder vi, at mennesker har svært ved at visualisere hvordan lys udbreder sig fra en lampe. Hvis kunden mangler visualisering af lampen og dens belysning, kan der opstå fejlkøb af lamper. Hvis kunden oplever fejlkøb af en lampe, kan det medføre økonomiske konsekvenser, hvis lampen ikke kan returneres. Hvis fejlkøb resulterer i købet af en lampe med dårlig belysning, kan det medføre konsekvenserne nævnt i afsnit 3.1.1.

Opsummering

I dette afsnit er der på baggrund af en antagelse, blevet argumenteret for, at manglende visualisering af lamper kan føre til fejlkøb, samt de konsekvenser dette kan have. Da de nævnte konsekvenser i afsnit 3.1.1, kan skyldes manglende visualisering af en lampe og dens belysning, ved købet af lampen, så må vi formode at det initierende problem er relevant.

Rapporten kan nu beskæftige sig med at undersøge andre dele af problemfeltet, men for først, at få en forståelse for det initierende problem, er der i næste afsnit en beskrivelse af de centrale begreber i det initierende problem.

3.2 Begrebsliggørelse

Der er indtil videre blevet argumenteret for relevansen af det initierende problem, og det er i den sammenhæng nødvendigt, at redegøre for nogle vigtige emner og ord indenfor problemfeltet.

Formålet med dette afsnit er at beskrive vigtige begreber samt kort at give en beskrivelse af, hvordan de forskellige ord, og begreber skal forstås i den videre rapport. Begreberne visualisering, lys, farvetemperatur og lampe, som fremgår i følgende afsnit, danner grundlag for forståelsen af det initierende problem.

3.2.1 Visualisering

At visualisere betyder at skabe et billede på baggrund af noget [6]. Dette kan til dels være tanker, som omsættes til billeder for det indre øje. Det kan også være en række data, som omsættes til billeder, så de er nemmere at forstå. Visualisering kan være et værktøj til at skabe en forståelse for det, der visualiseres. Dette kan f.eks. være *prototyper af lamper*, der kan give en forståelse for hvordan lyset udbreder sig fra en lampe. Derudover er der inden for computergrafik metoder til at skabe billeder på baggrund af *3D-modeller*, så man f.eks. kan lave et realistisk billede af en lampe. Dette billede kan hjælpe med at få en forståelse for, hvordan lampen ser ud i virkeligheden og hvordan dens lys udbredes. Forskellige teknologier til visualisering er uddybet senere i rapporten under afsnit 5.2.

3.2.2 Lys

Formålet med dette afsnit er at forsøge at definere lys, og beskrive hvilken type af lys, som rapporten vil tage udgangspunkt i. Derudover redegøres der for farvetemperatur samt den optimale placering af lyset ift. anvendelsen.

Der er forskellige opfattelser af hvad lys indebærer. Hvis vi tager udgangspunkt i Karsten Rottwitt, som er professor ved DTU fotonik, definerer han lys som:

"Lys er andet end synligt lys. For mig er lys et elektromagnetisk felt, som har en høj frekvens" - Karsten Rottwitt [7].

Han mener også, at der er en hårfin grænse for hvornår lys kan betegnes som lys, denne grænse er dog først i spil når vi snakker om UV-lys og infrarødt lys [7]. Andre er ikke enige med Karsten Rottwitt om hvordan lys defineres. Tager vi nu udgangspunkt i Britannica [8], så betegnes lys, som magnetiske stråler, som det menneskelige øje kan opfange, også kaldet synligt lys.

Det er denne definition, som rapporten vil tage udgangspunkt i. Dette er valgt, da det er oplagt at kombinere synligt lys og lamper.

Farvetemperatur Det lys, som kommer fra en lampe, kan have forskellige farvetemperaturer. Ordet beskriver hvilken farve lyset fra et sort legeme, med en bestemt temperatur vil have [9]. Dette er f.eks. om det er varmt eller koldt. Farvetemperaturen måles i kelvin, og der er forskel på, hvor man bør benytte pærer med forskellige farvetemperaturer. Integral-LED er et firma med over 25 års erfaring [10], som har opstillet nogle foretrukne steder at bruge de forskellige typer af lys:

1. Varm(2700 Kelvin) til varm hvid(3000 Kelvin): anbefaler de at bruge steder som i stuen, soveværelset og i entréen [11].
2. Hvid(4000 Kelvin) til kold hvid(5000 Kelvin): anbefaler de at bruge steder som i køkkenet, kontoret, badeværelset og i større skabe [11].

Denne rapport vil altså tage udgangspunkt i synligt lys, hvor farven af dette lys kan beskrives med en farvetemperatur. Ud fra ovenstående er det nu også beskrevet hvordan lysets farvetemperatur er en faktor, når der skal afgøres hvilket lys der passer bedst ind i et bestemt rum.

3.2.3 Lamper

Formålet med dette afsnit er at konkretisere definitionen af hvad en lampe er i vores kontekst, og hvordan begrebet skal forstås i rapporten. Der findes mange forskellige definitioner på hvad en lampe er, og det viser sig ifølge American Heritage® Dictionary of the English Language [12], at begrebet 'lampe' dækker over mange forskellige ting.

American Heritage definerer en lampe som værende én eller flere af følgende:

En af flere forskellige enheder, der genererer lys og ofte varme, især:

1. En elektrisk anordning, der har en sokkel til en pære, især et fritstående stykke møbel.
2. En anordning, der afgiver ultraviolet, infrarød, eller anden stråling, som kan anvendes til terapeutiske formål.
3. En pære: en projektør/et spotlight, udstyret med metalhalogenlampe.
4. En lanterne eller armatur, der afgiver lys ved afbrænding af gas/brændbare væsker, ofte ved brug af en kappe.

Idet der er så mange forskellige definitioner på en lampe, er vi i konteksten af vores projekt nødsaget til at afgrænse begrebet. Da vi vil hjælpe forbrugeren med at visualisere lampen i et givent rum, tages der udgangspunkt i en indendørs lampe, og idet vi ønsker at tage udgangspunkt i en mere 'normal' lampe, afgrænses vores definition af en lampe også til den førstnævnte definition, altså "en elektrisk anordning, der har en sokkel til en pære, især et fritstående stykke møbel".

Opsummering

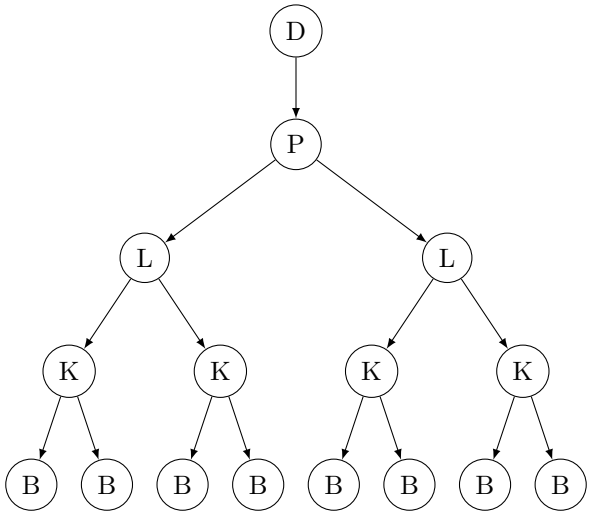
Ud fra de ovenstående afsnit i begrebsliggørelsen, kan der nu kortfattes, at der senere i denne rapport anvendes de omtalte begreber med følgende betydning:

1. Visualisering: Skabelsen af et billede på baggrund af noget, der evt. ønskes lettere forståeligt.
2. Lys: Den elektromagnetiske stråling, der er synligt for øjet (Synligt lys).
3. Farvetemperatur: Om et lys ser varmt eller koldt ud. Måles i kelvin.
4. Lampe: En indendørs anordning hvor der kan isættes en pære, som udsender lys der evt. afskærmes af anordningen.

Ud fra de ovenstående begreber skulle der nu være en entydig forståelse af det initierende problem, som gør at problemet nu kan analyseres videre i de kommende afsnit.

3.3 Interessentanalyse

I dette afsnit undersøges der hvilke interessenter der er, og hvilken interesse de har i problemet samt en eventuel løsning af problemet. Vi har udvalgt fem interessenter ud fra emnet lamper, da belysning fra lamper er det centrale i det initierende problem. Formålet med interessentanalysen er, at finde ud af hvilke interessenter som denne rapport vil løse problemet for. De fem interessenter er: designer, producent, lampebutik, kunde og bruger, som vist på nedenstående figur.



Figur 1: Illustrerer princippet bag, hvordan lampen overføres mellem de fem interessenter designer(D), producent(P), lampebutik(L), kunde(K) og bruger(B).

3.3.1 Designere

Designere er interesseret i at deres design bliver solgt og er derfor sandsynligvis interesseret i et program, der kan hjælpe dem med at gøre deres design mere populært.

Vi har haft kontakt med to forskellige lampedesignere, danske Erik Mortensen, og svenske David Wahl fra IKEA, for at undersøge hvilke værktøjer de bruger til visualisering, når de designer lamper. Erik Mortensen sagde følgende:

"I alle mine lamper er valg af lyskilde og placering sket på grundlag af test via prototyper. De fleste af mine lamper er prototyper".

"Jeg har i en del år arbejdet med lampedesign. og har derfor mest været optaget af armaturets/lampens skulpturelle udtryk, men da det jo er en lampe skal den selvfølgelig også opfylde det belysningsmæssige." Mailen kan ses i bilag A.

På baggrund af dette kan man altså uddrage at Erik Mortensen hovedsagligt benytter sig af

prototyper, til at visualisere lampens lys på.

David Wahl sagde:

"Apart from hand sketching and physical prototypes, we use the 3D modeling application Solid Works in IKEA of Sweden. And for renderings we use either the built in renderer, or photo works, which is also part of solid works." Mailen kan ses i bilag B.

David Wahl bruger altså, ligesom Erik Mortensen, prototyper, men derudover benytter han sig også af computerprogrammet Solid Works, som består af produkter til 3D-modellering, simulering og visualisering [13].

Ud fra de to udtagelser fra Erik Mortensen og David Wahl, er der på den ene side *prototyper*, som en teknik til at visualisere lampen, og på den anden side *computersoftware*, som f.eks. Solid Works, som visualisere en 3D-model af lampen. Derfor tyder det ikke på at designere mangler værktøjer til visualisering, når de designer lampen, men at problemet om manglende visualisering af lamper og deres belysning må opstå et andet sted.

3.3.2 Producenter

Producenten samarbejder med designeren om at udvikle lampen. Producentens rolle er at fremstille lampen på baggrund af designet. Når lamperne er produceret sendes de ud til lampebutikkerne. Producenten kan have en interesse i, at kunderne køber deres produkter i lampebutikkerne, da der er mulighed for, at dette vil medføre at lampebutikkerne bestiller flere af deres produkter hjem. Vi har haft kontakt med en belysningskonsulent, som her ønsker at være anonym. Belysningskonsulenten sagde følgende om producenternes interesse i problemet:

"Det er blevet en kompliceret proces at producere en lampe ift. EU lovgivning i dag så jeg har svært ved at se at producenterne vil koste endnu flere penge til produkter til privatmarkedet som måske kun køber en lampe til 3000 kr. som ofte kun interesserer sig for den laveste pris og ikke den bedste service og rådgivning. Så producenters incitament til ligge investeringer hos privatkunder er meget begrænset." - Mailen kan ses i bilag D.

Belysningskonsulenten mener altså at producenterne ikke vil smide en masse penge ind i privatmarkedet, da det ikke gavner dem økonomisk.

Producenterne får sandsynligvis flere midler til større produktion af en given lampe, hvis lampen er populær. desto flere solgte lamper til en lampebutik, desto flere penge tjener producenten. Det er derfor et problem for producenterne, hvis en lampe returneres, grundet en utilfreds kunde. Dog er producenterne, som nævnt i citatet fra belysningskonsulenten, ikke nødvendigvis interesseret i at investere penge i rådgivning, og dermed også visualisering af en lampe og dens belysning for kunder. I stedet er dette en opgave, som i større grad ligger hos lampebutikken, som beskrevet i næste afsnit.

3.3.3 Lampebutikker

Lampebutikken er interesseret i at sælge flest mulige lamper, da dette giver større indkomst for butikken. Derudover er butikken også interesseret i, at kunden køber den 'rigtige' lampe første gang, da butikken på denne måde undgår utilfredse kunder, som vil returnere lamperne.

Derudover er det lampebutikken, som er den primære rådgiver for kunderne, når der skal købes en ny lampe, derfor vil det være oplagt, at lave et værktøj, som lampebutikker kan stille til rådighed for sine kunder, så det bliver lettere at visualisere en lampe og dens belysning.

3.3.4 Kunder

Kunden er interesseret i at visualisere hvordan lys udbreder sig fra en lampe, da dette vil hjælpe kunden med at afgøre hvordan lampen passer ind i en kontekst, og kunden kan derfor undgå fejlkøb. Dette er kunden interesseret i, da det i sidste ende vil gavne brugeren af lampen, hvis kunden er i stand til at købe en lampe, som passer ind i konteksten. Kundens interesse i visualisering af lampen og dens belysning underbygges af følgende citat fra den førnævnte belysningskonsulent:

"Der er overraskende mange der gerne vil se lyset inden de køber lamper." Mailen kan ses i bilag D

Kundernes interesse vil også være at undgå de økonomiske konsekvenser, hvis der købes en lampe, som opfylder de forventninger, som kunden forestillede at lampen opfyldte, men først indses efter lampen er installeret og emballagen er brudt, og lampen dermed ikke kan returneres.

3.3.5 Brugere

Det er brugerne der i sidste ende benytter sig af lamperne i deres hjem eller på deres arbejde. Dette gør brugerne til den gruppe af interessenter, som påvirkes direkte af problemet, da de må leve med konsekvenserne, som belysningen fra en lampe kan medføre, som omtalt i afsnit 3.1.1. Et eksempel på brugere, er hjemme i privaten, hvor der kan være mange forskellige typer lamper, som skal passe ind i hjemmet. Hvis en person i hjemmet bruger en lampe, som har en belysning, der ikke passer ind i hjemmet pga. manglende visualisering ved købet, vil dette påvirke brugerne i hjemmet, da dårlig belysning kan have sundhedsmæssige konsekvenser som omtalt i afsnit 3.1.1.

3.3.6 Målgruppen

Som beskrevet i forrige afsnit, er det både designere, producenter, butikker, kunder og brugere, der påvirkes af problemet. Det er nu relevant at afgøre hvem problemløsningen retter sig mod, da dette danner grundlag for, hvordan løsningen skal udvikles og hvem, der kan indrages i løsningen og udarbejdelsen af løsningsforslaget.

Som illustreret på figur 1 er det lampebutikkerne, som har den direkte kontakt til kunderne og via kunderne en forbindelse til brugerne. Designere er fravalgt, da vi ud fra mails fra Wahl og Mortensen kan uddrage, at de allerede har værktøjer til at visualisere lys fra lamper, og som der ses på figur 1 har designeren ikke nogle direkte kontakt til kunden eller brugeren. Man kunne forestille sig, at problemet kunne løses allerede fra producentens side. Ud fra udtagelsen fra belysningskonsulenten, er vi dog blevet informeret om, at producenterne ikke er interesseret i at bruge ressourcer på at løse problemet, da det ikke gavner producenterne direkte. Derudover er det ikke producentens opgave at vejlede kunder til det bedste køb af lamper, dette er derimod lampebutikkens opgave.

Hvis man retter problemløsningen mod lampebutikker, og laver en løsning, der gør det muligt for kunder at visualisere lamperne bedre, er det sandsynligt at kunderne vil være mere tilfredse med deres lamper, da de har mulighed for at se lampens belysning inden købet. For lampebutikker kan dette betyde, at kunderne ikke returnerer lige så mange lamper, og dette vil bidrage til øget kundetilfredshed, som i sidste ende gavner både lampebutikker og kunderne. Hvis kunden er i stand til at købe en lampe som passer ind i den korrekte kontekst vil dette også tilfredsstille brugeren.

Opsummering

I afsnittet er der blevet argumenteret for, at det påvirker brugeren af lampen hvis kunden ikke kan visualisere hvordan lyset udspreder sig fra en lampe, og dette vil i sidste ende også påvirke lampebutikkerne. Derudover er der blevet argumenteret for, at det er mest fordelagtigt at rette løsningen mod lampebutikker, da det er lampebutikkerne, der rådgiver kunderne, som køber lampen for brugerne.

Dette afsnit er relevant i forhold til den senere problemformulering, da der er blevet argumenteret for hvem det er, som har problemet, samt hvilken målgruppe det senere produkt skal udvikles til.

3.4 Problemets placering

I dette afsnit undersøges der hvor, og i hvilke situationer problemet opstår. Da problemet tager udgangspunkt i købet af lampen, beskrives der hvordan købssituationen er, ved hhv. handel i fysiske butikker(detailhandel) og handel i internetbaserede butikker(e-handel). Ud fra disse undersøgelser sammenlignes detailhandel og e-handel, for at afgøre hvor problemet er størst og derudfra vælge hvilken type af handel, som denne rapport ønsker at løse problemet indenfor.

3.4.1 Detailhandel

En fysisk butik er et sted, hvor kunderne selv skal komme hen, når de vil købe eller kigge på butikkens varer. En fysisk butik har et personale, som tager sig af butikkens kunder, og som besvarer deres mulige spørgsmål til butikkens varer. En fysisk butik er, grundet det ansatte personale og andre udgifter, dyr i omkostning. Dog viser en amerikansk undersøgelse, at 92% af de udspurgte 1029 personer i undersøgelsen foretrækker de fysiske butikker, da man kan se varen selv og få direkte assistance om varen gennem en medarbejder [14]. Det var ikke muligt at finde en lignende rapport, der viste noget om danskernes meninger mht. om de foretrækker at se den fysiske vare, før de køber den, men grundet kulturelle ligheder mellem USA og Danmark, har vi valgt at antage at en lignende tendens er gældende i Danmark.

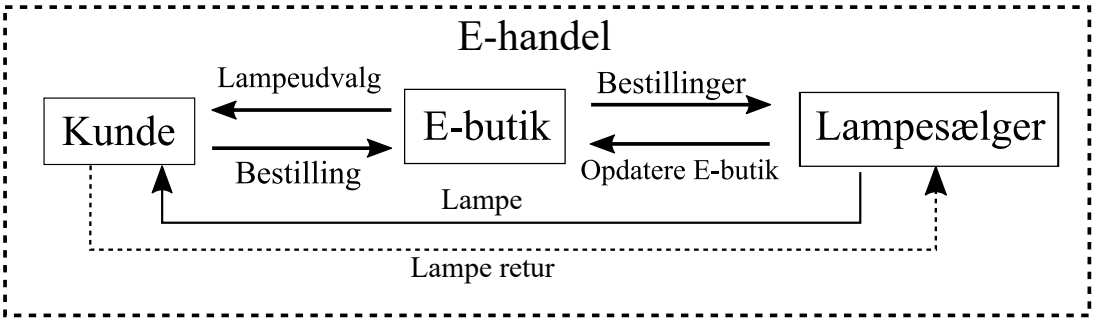
I vores kontekst snakker vi om en, hvilken som helst, fysisk butik, der har med lampesalg at gøre. Den lampeinteresserede kunde kommer ud i butikken og leder f.eks. efter en ny væglampe til stuen. Problemet heri kan opstå ved, at der er adskillige forskellige lamper at vælge imellem, men ikke alle lamperne er tilsluttet og man har derfor ikke mulighed for at se lampens belysning i rummet.

Ved køb af lamper i den fysiske butik, er det ikke altid, at de lamper som stilles frem til udstilling giver et realistisk billede af, hvordan lyset udbreder sig, da der ofte er mange lamper og lyskilder tæt samlet. Et andet problem er returretten. Returretten er ikke obligatorisk at have for fysiske butikker, så det er altså op til den enkelte butik/butikskæde, om de vælger at gøre det muligt at returnere en vare selvom den er uåbnet og stadig i original emballage [15]. Hvis kunden beslutter sig for en lampe, som ikke er tilsluttet, men regner med at den vil se godt ud på væggen i stuen, hvorefter det så viser sig, at lyset falder helt forkert, er det for sent, da lampen er pakket ud. Lampen kan altså ikke byttes, og er ikke optimal i forhold til kundens stue. Der er mange butikker og butikskæder, som vælger at gøre det muligt for kunden at bytte en vare, hvis den stadig er i original emballage og med kvittering [16], men det er ikke noget, som butikker er tvunget til at gøre, og med en lampe som eksempel kan man ikke tage den med hjem og 'afprøve', da dette giver afkald på returretten.

3.4.2 E-handel

E-handel er, i modsætning til detailhandel, elektronisk handel via internettet [17]. På internettet kan lampebutikker have såkaldte e-butikker, hvor kunder kan købe varer [18]. E-butikker er ofte udformet således, at kunden kan se billeder og informationer omkring lampebutikkens varer og derudfra kan kunden vælge at lægge varerne i en virtuel indkøbskurv, hvor kunden til sidst indtaster de nødvendige oplysninger for at købe og modtage varerne.

Blandt de mange forskellige varer, der sælges via e-butikker, er det her relevant at tale om e-handel med lamper. Nedenstående figur 2 illustrerer princippet bag en lampesælgers salg af lampe til en kunde via en e-butik.



Figur 2: Princippet bag handel af en lampe via en e-butik.

På figur 2 er det vist, hvordan e-handlen starter med, at kunden får et udvalg af lamper fra e-butikken. Kunden sender så en bestilling, som via e-butikken sendes videre til lampebutikken, og til sidst sendes lampen til kunden. Dog ender handlen ikke nødvendigvis her, da kunden kan sende lampen retur såfremt at de gældende lovgivninger og købsbetingelser muliggør dette. For at undersøge lovgivningen nærmere kan man tage udgangspunkt i den danske lov om forbruger-aftaler, nærmere bestemt ved: LOV nr 1457 af 17/12/2013 Gældende (Forbrugeraftaleloven), Offentliggørelsesdato: 18-12-2013 Justitsministeriet [19].

I lovens kapitel 1, § 1, stk. 2, nr. 1 fremgår det, at lovens bestemmelser for fortrydelsesret gælder for aftaler, som er indgået ved fjernsalg. For en fjernsalgsaftale gælder der, at aftalen om varer, er indgået gennem fjernkommunikation, hvor den erhvervsdrivende og forbrugeren ikke mødes fysisk (jf. kap. 1, § 3, nr. 1).

Ser man nu på loven i forbindelse med e-handel, foregår fjernkommunikationen gennem internettet via e-butikken, hvor fjernsalgsaftalen udføres i form af brugerens bestilling af f.eks. en lampe. Dette gør at fortrydelsesretten gælder ved e-handel.

Fortrydelsesretten er en kundes mulighed for at melde sig ud af en aftale, herunder køb af lamper ved e-handel. Hvis en kunde eksempelvis køber en lampe via en e-butik, har kunden mulighed for at fortryde købet inden 14 dage ved at meddele dette til den erhvervsdrivende (jf. kap. 4, § 19). Herefter har kunden 14 dage til at returnere varen (jf. kap. 4, § 24). Hvis varens værdi er

forringet som følge af kundens unødvendige håndtering af varen for at inspicere denne, så hæfter kunden for værdiforringelsen (jf. kap. 4, § 24, stk. 5). Dvs. at hvis en bruger installerer og bruger lampen, hvor der f.eks. tilpasses ledninger, så kan lampens værdi forringes og kunden skal hæfte for dette.

3.4.3 Sammenligning af detail- og e-handel

Ud fra ovenstående redegørelse af de to typer for handel, analyseres disse nu med henblik på at finde ligheder og forskelle, hvoraf det kan afgøres i hvilken af de to typer af handel, at problemet er størst.

Da det initierende problem er, at kunden ikke kan visualisere lampen uden at købe den, er det derfor relevant, at se på i hvor høj grad dette er tilfældet ved de to typer handler.

Fordelen ved detailhandel er, at kunden ofte kan se lampen i butikken, og ud fra dette, vurdere hvilken lampe der opfylder de behov som kunden har. Dog er problemet stadig, at kunden ikke ser lampen i den rette kontekst, dvs. i sit eget hjem. Dette kan gøre, at kunden får et godt indtryk af lampen i den kontekst, som butikken præsenterer den i, men at den ikke passer ind i den kontekst, som kunden køber lampen til.

Ved e-handel har kunden ikke muligheden for, at se en fysisk udgave af lampen, men ofte kun billeder. Dette gør at kunden alene kan tage valg ud fra de billeder og informationer som e-butikken præsenterer. Problemet er så, at billederne til dels ikke er interaktive, dvs. brugeren ikke kan se lampen fra flere vinkler end dem som billederne er taget i, samt at billederne ikke er taget af lampen i den kontekst, som kunden ønsker at købe lampen til.

Med hensyn til konteksten er fordelen ved e-handel, at kunden kan sidde derhjemme, i den kontekst, hvor lampen skal indgå, og sammenligne med de informationer, der er tilgængelige på e-butikken. I modsætning til dette er detailbutikker, hvor kunden står i butikken, og måske har problemer med at huske eller blot forestille sig alle detaljerne ved den kontekst, som lampen skal indgå i.

Ud fra denne sammenligning, er der på den ene side detailhandel, hvor det er svært at visualisere konteksten, men hvor man kan se lampen. På den anden side er e-handel, hvor man kan sidde derhjemme i konteksten, men har svært ved at visualisere lampen.

For at afgøre hvilken type handel denne rapport vil fokusere på, skal der derfor svares på om det er mangel på visualisering af lampen i kontekst ved detailhandel eller mangel på visualisering af lampen ved e-handel, som er det største problem.

Da kunden omgås og ser den kontekst, som lampen skal indgå i f.eks. et kontor, køkken og bad, må man kunne antage at kunden har en forestilling om, hvordan denne kontekst ser ud selvom kunden ikke står i den, når der handles i en fysisk butik. Derfor er dette ikke et lige så stort problem, som hvis kunden ikke kan visualisere lampen når der handles via e-handel. Derfor vil fokus i denne rapport være at forbedre kundens evne til at visualisere lamper under e-handel.

Opsummering

Ud fra ovenstående kan der opsummeres følgende iagttagelser for problemet i forhold til detail- og e-handel:

- Detailhandel: Fysisk lampebutik, hvor det kan være svært at visualisere lampen i den kontekst, som kunden vil købe lampen til, da kunden ikke er til stede derhjemme i konteksten.
- E-handel: Internetbaseret lampebutik, hvor det kan være svært at visualisere lampen, som kunden vil købe, da kunden ikke kan se en fysisk udgave af lampen.

Ud fra ovenstående er valget faldet på visualisering af en lampe og dens belysning ved e-handel, da visualisering af selve lampen og dens belysning er højere prioritet end visualiseringen af den korrekte kontekst.

Dermed er der nu igennem problemanalysen afgrænset til, at det initierende problem skal løses ved sælgeren, herunder internetbaserede lampebutikker, hvor det er et problem at kunden mangler visualisering af lampen og dens belysning. Denne afgræsning samt problemanalysens resultater er opsummeret i den endelige problemformulering i næste afsnit.

4 Problemformulering

Ud fra problemanalysen er vi blevet opmærksomme på følgende problem indenfor problemfeltet: Det er et problem at kunden i en købssituation på e-butikker ikke kan visualisere, hvordan lys udbreder sig fra en lampe. Dette kan føre til fejlkøb af lamper, som påvirker både kunden og e-butikken. For brugeren kan dette betyde irritation og i værste tilfælde kan det have sundhedsmæssige konsekvenser. For lampebutikker kan fejlkøb medføre utilfredse kunder og heraf dårlig omtale.

Dette fører os til det overordnede spørgsmål som vi ønsker besvaret i dette projekt:

Hvordan kan vi lave et værktøj til e-butikker, som visualiserer belysningen fra indendørslamper for kunderne?

Herunder er der tre underspørgsmål, som ønskes besvaret:

1. *Hvordan visualiseres lyset fra en given indendørslampe?*
2. *Hvordan kan lampen og dens belysning visualiseres fra flere vinkler?*
3. *Hvordan visualiseres forskellige pærers lys?*

5 Løsningsdesign

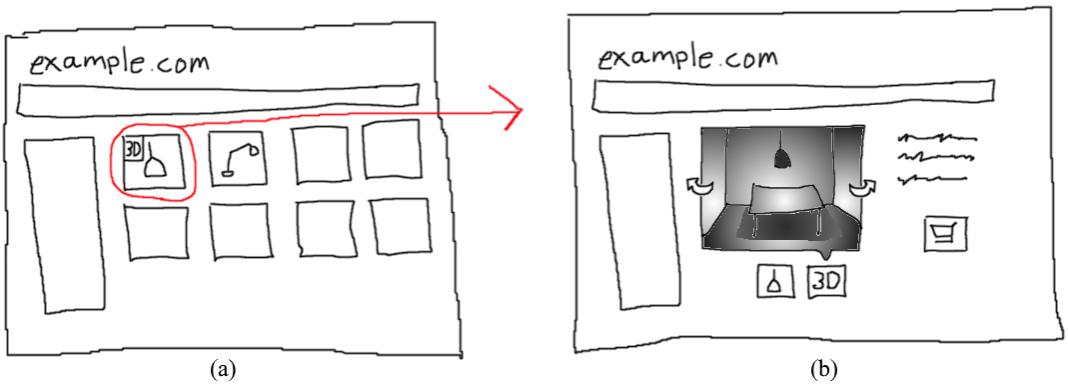
I dette afsnit er formålet, at finde og beskrive løsningsforslag til den endelige problemformulering. Først er den overordnede ide bag løsningen beskrevet, hvorefter der undersøges hvilken tilgang til udviklingen af løsningsforslaget, som er mest oplagt. Dette gøres ved at se på de teknologier, som kan anvendes til at visualisere en lampe og dens belysning.

5.1 Løsningsforslag

Dette afsnit beskriver en idé til løsningen af den endelige problemformulering. Idéen er resultatet af en diskussion i gruppen, hvor vi kom med idéer til løsninger, og herefter diskuterede fordele og ulemper ved de forskellige idéer. Formålet med afsnittet er at skitsere gruppens bud på den optimale løsning på problemet. I slutningen af afsnittet vil der være en afgrænsning af løsningen som vil danne grundlag for den senere udvikling af en løsning.

Som beskrevet i problemanalysen har gruppen valgt at løse problemet når kunder handler på e-butikker. Tanken er derfor at implementere software på en e-butikshjemmeside med henblik på at visualisere lamper og deres belysning for kunderne. I næste afsnit er en skitse af gruppens løsningsforslag.

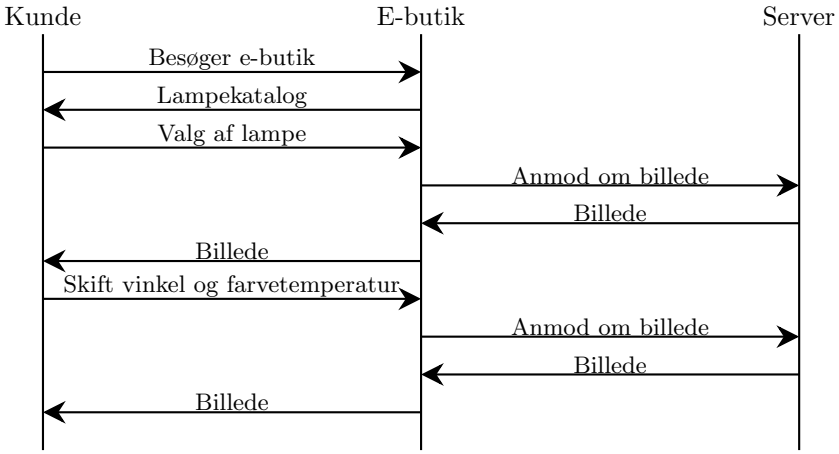
5.1.1 Skitse af løsning



Figur 3: Skitse af idé til løsning.

På figur 3 er der illusteret en skitse af en e-butik, som sælger lamper. Figuren illustrerer hvordan systemet kan integreres på en hjemmeside. Skitse 3a viser et online katalog over e-butikkens udvalg af lamper. Som det fremgår af skitsen vil nogle lamper være markeret med et '3D-ikon' og dette indikerer, at kunden har mulighed for, at se lampen fra flere vinkler. Kunden tilgår billedet ved at klikke på ikonet. Når kunden trykker på ikonet bliver kunden omdirigeret til en

anden menu. Som det fremgår af skitse 3b, så har kunden her mulighed for at se et billede af lampen. De to pile på skitsen indikerer, at kunden har mulighed for at rotere billedet, og se hvordan belysningen fra lampen er, set fra forskellige vinkler. Derudover har kunden mulighed for, at justere lampens farvetemperatur i kelvin. Meningen med denne feature er, at kunden har mulighed for at visualisere hvordan forskellige pærer vil se ud i lampen. De forskellige billeder for lampen fra forskellige synsvinkler og farvetemperaturer, vil ligge til rådighed på en ekstern server og vil derfor ikke gøre e-butikkernes hjemmeside betydeligt langsommere.



Figur 4: Sekvensdiagram af løsningsidéen.

Figur 4 tager udgangspunkt i en hjemmeside, som har fået implementeret vores løsning, og viser et sekvensdiagram, som illustrerer processen når en kunde vil købe en lampe.

I forbindelse med implementeringen af softwaren på en hjemmeside har der været forskellige ting, som skulle overvejes. Da problemet omhandler visualisering af lys fra lamper, er det primære mål, at kunden kan se billeder af lampen og dens belysning fra forskellige synsvinkler og med forskellige farvetemperaturer. Derfor har vi afgrænset således at valg af konteksten ikke prioriteres.

5.1.2 Krav til løsningen

I forbindelse med vores projekt har vi fået nogle krav fra universitetets side. Disse er, at programmet skal skrives i programmeringssproget C, derudover er der også tidsmæssigt pres da hele projektet kun varer ca. to en halv måned (Svarende til 10 ECTS).

Ud fra afsnittet 5.1.1 og med baggrund i underspørgsmålene i problemformuleringen (afsnit 4) har gruppen opstillet følgende krav til løsningen:

1. Løsningen skal gøre det muligt at vise et billede af en lampe og dens belysning.
2. Det skal være muligt at ændre, hvilken synsvinkel lampen ses fra.

3. Det skal være muligt at bestemme hvilken farvetemperatur, som pæren i lampen visualiseres ud fra.
4. Løsningen skal fremskaffe et billede på en tid så den er praktisk anvendelig.
5. Løsningen skal kunne implementeres sådan at billederne af lampen kan vises for kunderne på en e-butiks hjemmeside.

Kravene hører alle under det overordnede spørgsmål i problemformulering i afsnit 4, men hvor ovenstående krav 1-3 specifikt knytter sig til hhv. underspørgsmål 1-3 i problemformulering.

Opsummering

Da den primære udfordring rent datalogisk set ligger i krav 1-4, som omhandler selve visualiseringen af lampen og dens belysning, er disse krav det primære fokus, hvor krav 5 er et sekundært krav, som er nødvendigt, hvis løsningen skulle kunne anvendes i praksis. Derfor vil vi lave et program, der gør det muligt, at visualisere hvordan lys spreder sig fra en given lampe beskrevet som en 3D-model. Derudover skal det være muligt at justere pærens farvetemperatur, samt se lampens belysning fra flere forskellige vinkler.

5.2 Teknologier til visualisering

Vi ser en tydelig mulighed for at assistere kunden med at træffe et valg, når det kommer til køb af varer på nettet. Formålet med afsnittet er, at få en forståelse af hvilke teknologier der allerede eksisterer inden for visualisering, og finde ud af hvilken teknologi, der er bedst i forhold til visualisering af lamper for kunder, der besøger en e-butiks hjemmeside. De enkelte teknologier vurderes på baggrund af de krav, der er sat til løsningsforslaget i afsnit 5.1.

5.2.1 Digitale billeder taget med et fysisk kamera

Som beskrevet under afsnit 3.4.2, benytter e-butikker, sig ofte af billeder til at vise kunden deres varer over internettet.

Fordelen ved denne type af visualisering er, at den giver et virkelighedstro billede af, hvordan lampen ser ud i den kontekst, som billedet er taget i. Ulempen er, at der ofte kun er et begrænset antal billeder til rådighed, hvilket kan medføre, at køberen ikke kan se lampen fra alle vinkler og på den måde ikke kan visualisere lampen for sig. Det er dog en mulighed at tage mange billeder, men med forskellige pæretyper, vinkler og lamper, giver dette hurtigt mange kombinationer, og dermed mange billeder, der skal tages, hvilket gør det til en tidkrævende proces. Manglende billeder kan gøre det svært for kunden, at se hvordan lyset udbreder sig fra lampen, da dette til dels afhænger af hvilken vinkel man ser lampen fra.

Herudfra kan man kortfattet sige, at visualisering af lamper gennem billeder, taget med et fysisk kamera, giver et realistisk billede af lampen, men kun i den kontekst og vinkel billedet er taget i.

5.2.2 Computergrafik

I computergrafik, er en 3D model, en beskrivelse af objekters form og materiale [20]. Computergrafiske metoder kan bruges til at simulere, hvordan lys interagerer med modellen og på den måde tegne et billede af modellen. Der eksisterer forskellige computergrafiske metoder, flere af hvilke, kan bruges sammen med andre for, at opnå et mere realistisk eller effektivt resultat. Der er allerede værktøjer, som kan visualisere produkter til salg på e-butikker som f.eks. Cylindo [21]. Vi har ikke kendskab til at andre specialiserer sig, eller markedsfører sig på nuværende tidspunkt med deres kompetencer med fokus på visualisering af lampers belysning. Derfor er der herunder beskrivelse af to af de mest anvendte metoder inden for computergrafik.

Rasterisering er en metode til at visualisere miljøer med høj aktiv brugerinteraktion som f.eks. computerspil [22]. Metoden virker ved, rent matematisk, at projektere modellen på et billedplan som repræsenterer skærmen [22]. Fordelen ved rasterisering er, at disse projektioner, kan foretages meget hurtigt af computerens grafikkort, som er bygget specielt til formålet [22]. Ulempen ved rasterisering er, at mere avancerede lysfænomener ikke nødvendigvis, passer ind i den proces (graphics pipeline [22]), som de enkelte grafikkort danner billeder ud fra.

Raytracing forsøger nøjagtigt at simulere lys i et virtuelt miljø, i modsætning til rasterisering, hvor hastighed er den primære faktor. Raytracing bygger fundamentalt på at følge lysstråler og bygge en model for, hvordan lysstrålerne interagerer med forskellige objekter og materialer [23].

På grund af den mere detaljerede simulering af forskellige lysfænomener, tager det, for raytracing, ofte længere tid at lave et billede af en 3D-model, end det gør ved rasterisering. Fordelen ved raytracing er, at den gør det muligt at beskrive mere komplekse lysfænomener, og kan dermed også opnå en mere realistisk præcision end rasterisering [24]. Nogle fænomener som bløde skygger kan også beskrives [25]. Desto flere lysfænomener der kræves, des længere tid tager det oftest at rendere et billede, men raytracing tillader stor fleksibilitet.

5.2.3 Augmented Reality

Der er nu blevet gennemgået visualisering af virkelige objekter ved hjælp af digitale billeder og virtuelle objekter ved hjælp af computergrafik. I dette afsnit beskrives augmented reality, som er en teknologi, der kombinerer virkelige og virtuelle objekter [26].

Augmented reality fungerer ved at tage et billede med et normalt kamera, og herefter ændrer billedet ved at indsætte computergrafik på billedet [26].

Et eksempel på anvendelsen af augmented reality er Artemides Augmented Reality App. Denne app gør det muligt, at visualisere udvalgte lamper i en kontekst som brugeren selv vælger [27].

Fordelen ved augmented reality, i forhold til løsningsforslaget, er, at den muliggør at se lampen fra flere forskellige vinkler i den kontekst som kunden ønsker. Hvis der ses bort fra tekniske udfordringer ville det også være en fordel hvis kunden kunne visualisere en lampe og dens belysning i den kontekst som kunden ønsker at købe lampen til.
Ulempen ved augmented reality i forhold til løsningsforslaget er, at det er en teknisk udfordring, at få en rummelig forståelse for den virkelige kontekst så det virtuelle, der indsættes får en tilstrækkelig realisme. Det vil derfor være svært at simulere lampens belysning hvis man ikke kender til dimensioner eller materialerne i den virkelige kontekst. Hvis det lykkedes, at få en forståelse for de virkelige objekter i billedet, vil det stadig være raytracing eller en anden teknik inden for computergrafik som ville være at foretrække, når man skal visualisere lampens belysning. Derfor er udfordringen ved augmented reality dobbelt, da den både skal få en rummelig forståelse for de virkelige objekter i billedet, samt lave en realistisk computergrafik som passer ind i billedet.

Opsummering

Ud fra ovenstående afsnit er der udledt følgende fordele og ulemper for de enkelte teknologier.

Teknologi	Fordele	Ulemper
Kamerabilleder	Realistisk visualisering.	Tidskrævende. Begrænset kombinationer af lampen, synsvinkel, farvetemperatur og kontekst.
Computergrafik	Nemt at integrere på lampebutikkers hjemmeside. Kan opnå høj realisme af lampen og dens belysning. Nemt at ændre vinkel, farvetemperatur og kontekst hvorfra lampen visualiseres.	Kræver 3D-model af lampen og konteksten. Kræver meget computerkraft ved høj realisme.
Augmented Reality	Lampen kan visualiseres fra flere vinkler og i den kontekst som kunden ønsker at købe lampen til.	Det er en teknisk udfordring at virtuelle objekter med den virkelige kontekst, så der stadigvæk opnås en hvis realisme.

Tabel 1: Viser fordele og ulemper ved de tre teknologier til visualisering.

På baggrund af fordele og ulemper vist i tabel 1 sammenlignes de tre teknologier nu med de krav, der er opstillet til løsningsforslaget i afsnit 5.1.2.

Billeder af fysiske lamper taget med et kamera, kan bidrage til udviklingen af løsningsforslaget, da den delvist opfylder krav 1-3 og 5. Da det er muligt at tage et billede af lampen og dens belysning med en bestemt farvetemperatur og fra en bestemt vinkel. Derudover kan løsningen implementeres på en e-butiks hjemmeside, ved blot at bruge de billeder, der tages med kameraet. Ulempen er, at det kan være ressourcekrævende, da der skal tages billeder med forskellige lamper, farvetemperaturer og vinkler som giver mange kombinationer og dermed mange billeder.

Hvordan computergrafik kan biddrage med at løse kravene i afsnit 5.1.2, afhænger af hvilken teknik man benytter. Hvis man benytter rasterisering, vil det være svært at opfylde krav 1, da visualisering af lampens belysning kræver, at der anvendes lysfænomener, som ikke passer ind i modellen for rasterisering. Rasterisering vil dog være oplagt til at opfylde krav 2 og 4, da den som nævnt er bygget til at rendere et billede med høj brugerinteraktion og kort renderingstid. Raytracing vil derimod være oplagt til at opfylde krav 1-3, da det er muligt at simulere lampen og dens belysning med forskellige farvetemperaturer og vinkler. Derudover kan raytracing implementeres i løsningen, der renderer et billede, der kan vises på en e-butiks hjemmeside, hvilket gør at den opfylder krav 5. Ulempen ved raytracing er, at det kan være svært at opfylde krav 4, da simulering af lampens belysning kan være en tidskrævende proces.

Augmented reality kan bidrage til udviklingen af løsningsforslaget ved at den har potentialet for at opfylde de samme krav, som både visualisering vha. kamerabilleder og computergrafik (krav 1-3). Dog er dette en teknisk udfordring, da den rummelige forståelse af den virkelige kontekst af billedet er nødvendig for at kunne opnå en realistisk visualisering, når computergrafikken indsættes. I forhold til krav 4 har augmented reality den samme udfordring som f.eks. raytracing, da simulering af lampens belysning kan være en tidskrævende proces. Derudover kan det være svært at implementere augmented reality på en e-butiks hjemmeside, da det kræver adgang til kundens kamera.

Ud fra ovenstående har vi valgt raytracing som teknologi til visualisering, da den har potentialet for at opfylde kravene opstillet i afsnit 5.1.2, uden at være en lige så stor teknisk udfordring som augmented reality.

6 Problemløsning

I dette afsnit vil der blive redegjort for den teori, der er nødvendig for at udvikle en løsning der opfylder kravene opstillet i afsnit 5.1.2. Herefter vil strukturen og udviklingen af løsningen blive beskrevet. Til sidst i afsnittet vil løsningen blive testet med henblik på de forskellige krav i afsnit 5.1.2.

6.1 Teoridiskussion

Dette afsnit omhandler de teorier samt videnskabelige principper, som er relevante i forhold til at lave den løsning, som er beskrevet i afsnit 5.1. Der vil derfor, i afsnittet, være fokus på de principper, formler og algoritmer, der bruges i udviklingsfasen. Teoridiskussionen vil primært fokusere på at beskrive og undersøge tekniske og matematiske principper som ellers ville være svære at forstå og arbejde med. Ud fra kravene til løsningen i afsnit 5.1.2 har gruppen valgt at undersøge og diskutere følgende fænomener og principper: Rotationsmatricer i afsnit 6.1.1, konvertering af farvetemperatur til RGB-værdier i afsnit 6.1.2, fra 3D-model til billede i afsnit 6.1.3 herunder 3D-model, kamera, perspektiv projektion, backwards raytracing, phong-modellen og til sidst skæring mellem linje og trekant i rummet i afsnit 6.1.4. I slutningen af afsnittet vil der fremgå, hvordan de enkelte teorier kan bidrage til udviklingen af løsningen.

6.1.1 Rotationsmatricer

Hvis vi vil rotere et punkt eller en vektor omkring en akse i et koordinatsystem kan vi bruge en rotationsmatrix [28].

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \tag{1}$$

$$R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \tag{2}$$

$$R_z(\theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Vi indsætter den vinkel som vi vil dreje vektoren med i radianer og multiplicerer dem sammen som angivet i udtryk 1. Vektoren bliver drejet omkring nul-punktet med netop den mængde radianer, som er angivet. Nedenstående eksempel illustrerer princippet ved at dreje en vektor i

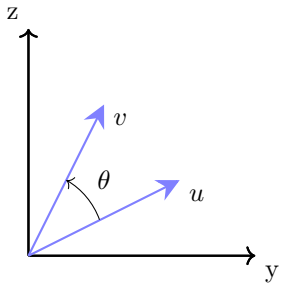
rummet.

$$\vec{u} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \tag{4}$$

Den roterede vektor $\vec{u}$ kan nu beskrives som set i udtryk 5

$$\vec{v} = R_x(\theta) \cdot \vec{u} = \begin{bmatrix} u_x \\ cos(\theta) \cdot u_y - sin(\theta) \cdot u_z \\ sin(\theta) \cdot u_y + cos(\theta) \cdot u_z \end{bmatrix} \tag{5}$$

Vektor $\vec{u}$ og $\vec{v}$ er illustreret i nedenstående figur 5.



Figur 5: Eksempel på rotation af en vektor om x-aksen

6.1.2 Konvertering fra farvetemperatur til RGB

Farvetemperatur, som også er beskrevet nederst i 3.2.2, er temperaturen af et udsendt lys og måles i kelvin. Denne temperatur kan bruges til at finde ud af, om et lys er varmt eller koldt. RGB-værdien er en værdi for en given farves indhold af rød, grøn og blå. Værdien angives normalt ved et tal mellem 0 og 255, altså er RBG-værdien [0, 128, 255] ingen del rød, en del grøn og fuld blå, hvilket, blandet sammen, giver en blålig farve. Der findes ingen direkte og 100% præcis formel for at 'oversætte' en kelvintemperaturværdi til en RGB-værdi, derfor har rapporten taget udgangspunkt i en algoritme, som er lavet ud fra 400 målinger, men som stadig ikke er præcis nok til videnskabelig brug [29]. Måden hvorpå algoritmen er lavet, er ved at tage disse 400 målinger, og lave en funktion ud fra dem. Der er lavet én måling per 100 kelvin, der starter ved 1000 kelvin og slutter ved 40.000 kelvin [30]. Ved at kigge på funktionen [31] har Tanner Helland kunne konkludere tre ting:

- Røde værdier under 6600 kelvin er altid 255.
- Blå værdier under 2000 kelvin er altid 0.
- Blå værdier over 6500 kelvin er altid 255.

Disse tre, forholdsvis simple, konklusioner har hjulpet med at gøre algoritmen meget kortere og mere simpel. Herunder kan udregningerne for hhv. rød-, grøn- og blå-værdierne ses matematisk, før de er skrevet om til kode. Matematikken er vist gennem gaffelfunktioner, altså funktioner med forskellige funktionsudtryk for bestemte intervaller.

$$R(k) = \begin{cases} 255 & 1000 <= \text{k} \wedge \text{k} <= 6600 \\ 329.698727446 * (k - 60^{-0.1332047592}) & 6600 < \text{k} \wedge \text{k} <= 40000 \end{cases}$$

$$G(k) = \begin{cases} 99.4708025861 * \ln(k) - 161.1195681661 & 1000 <= \text{k} \wedge \text{k} <= 6600 \\ 288.1221695283 * (k - 60^{-0.0755148492}) & 6600 < \text{k} \wedge \text{k} <= 40000 \end{cases}$$

$$B(k) = \begin{cases} 255 & 6600 <= \text{k} \wedge \text{k} <= 40000 \\ 0 & 1000 <= \text{k} \wedge \text{k} <= 1900 \\ 138.5177312231 * \ln(k - 10) - 305.0447927307 & 1900 < \text{k} \wedge \text{k} < 6600 \end{cases}$$

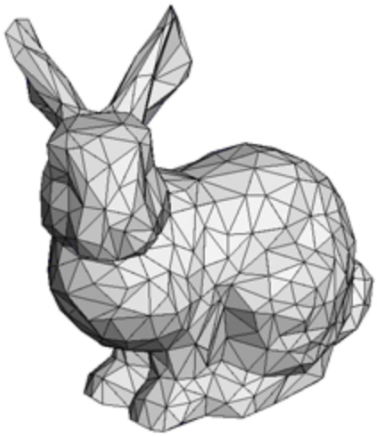
Grunden til at der oversættes fra farvetemperatur til RGB-værdi, er for at kunne visualisere farverne på en computer. Et billede vist på en computer består af pixels, som alle har en RGB-værdi, derfor kan lyset fra en lampe med en given farvetemperatur visualiseres, hvis farvetemperaturen oversættes til RGB-værdi.

Opsummering Vi har nu set hvordan man kan konvertere en farvetemperatur i Kelvin til RGB-værdier. Derudover har vi set, hvordan man kan rotere en vektor ved hjælp af matricer. Denne viden er nødvendig for at kunne opfylde kravene om at løsningen skal gøre det muligt at vise lampen med forskellige farvetemperaturer og vinkler.

6.1.3 Fra 3D-model til billede

I dette afsnit er formålet nu at vise en model for, hvordan billeddannelsen af objekter i rummet, også kaldt rendering, kan konstrueres. Dette er essentielt, da billeddannelsen danner grundlag for, hvordan 3D-modellen for en lampe omdannes til et billede, der kan vises for kunderne på e-butikken. Til sidst i afsnittet udledes en model for, hvordan belysningen fra en lampe kan simuleres og visualiseres vha. raytracing.

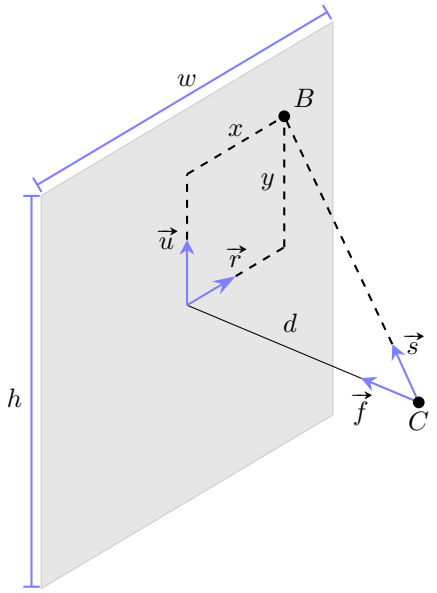
3D-model En 3D-model er en matematisk beskrivelse af et tre dimensionelt objekt. For at beskrive et 3D-objekt opdeler man ofte objektet i trekanter. Dette er illustreret på nedenstående figur [32].



Figur 6: Eksempel hvor trekanter bruges til at repræsentere et objekt.

For at rendere et billede af en 3D-model bestående af trekanter, er man nødt til at have en model for det kamera, som billedet renderes ud fra. Hvordan kameraet kan modelleres er beskrevet i næste afsnit.

Kamera Et kamera i forbindelse med rendering af en 3D-model, er typisk en beskrivelse af position, orientation og synsfelt. Andre informationer kan tilknyttes som beskriver opløsning af det renderede billede, samt perspektivet af det resulterende billede. Orientationen fastsættes ved hjælp af et sæt vektorer som beskriver hvilken retning der f.eks. er højre og op for kameraret.



Figur 7: Visualisering af et punkt på billedplanen som en kombination af tre vektorer $\vec{r}$, $\vec{u}$ og $\vec{f}$, der repræsenterer hhv. højre, op og frem retninger for kameraet.

Et punkt B på billedplanen (Se figur 7) kan repræsentere en pixel og kan beskrives som en lineær kombination af en op- og en højre-orientationsvektor, samt en vektor, der beskriver afstanden til billedplanen relativt til kameraet.

En stråles retning gennem et pixelkoordinat B relativt til kameraets udgangspunkt C kan altså beskrives på følgende måde:

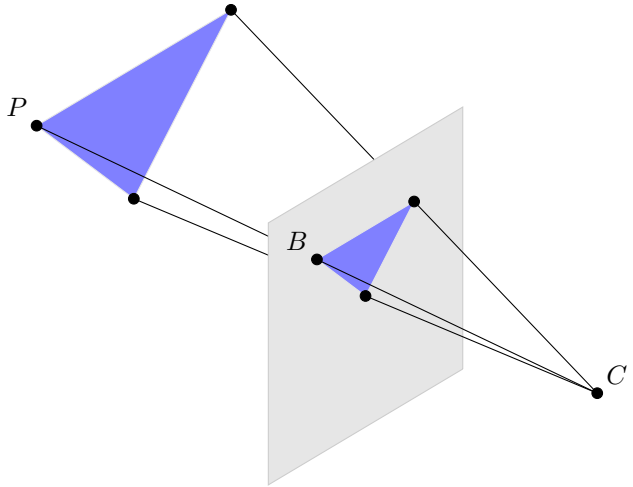
$$\vec{B} = C + \vec{f} \cdot d + \vec{u} \cdot \left(y - \frac{h}{2}\right) + \vec{r} \cdot \left(x - \frac{w}{2}\right) \tag{6}$$

$$\vec{s} = \frac{\vec{B} - \vec{C}}{||\vec{B} - \vec{C}||} \tag{7}$$

Hvor (x, y) er pixelkoordinatet på billedet mellem $(0, 0)$ og (w, h).

Afstanden d samt størrelsen af billedplanen bestemmer synsvinklen.

Perspektiv projektion For at udlede en model for billeddannelsen, tages der udgangspunkt i en perspektiv projektion. Perspektiv projektion er en måde at danne et billede af 3D-objekter ved at projektere objekterne hen på et plan mod et kameras position [33]. Princippet bag perspektiv projektion er vist på figur 8.



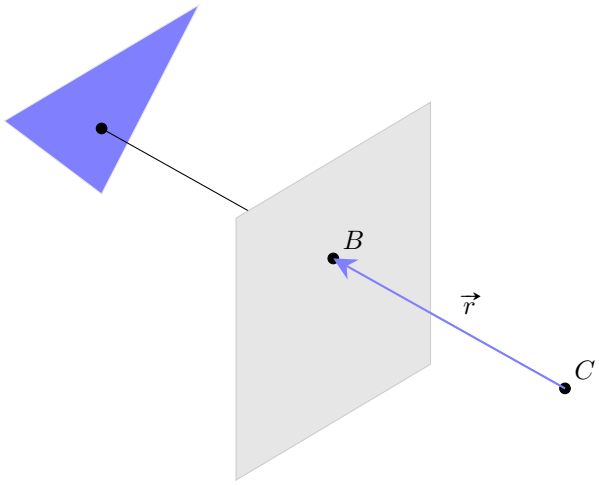
Figur 8: Princippet bag perspektiv projektion af et punkt på et billedplan.

Som vist på figur 8 kan et punkt $P \in \mathbb{R}^3$ projekteres ned på billedplanen α, ved at finde skæringspunktet B mellem billedplanen α og en lysstråle L, som går fra punktet P mod kameraets position C. Gør man nu dette for alle punkter på et objekt i rummet, og tegner skæringspunkterne på billedplanen, dannes et billede af objektet, ved at oversætte punktet på billedplanen, til (x,y)-pixelkoordinater, på det resulterende billede, med udtryk 6 beskrevet under afsnittet om kameraet.

Udfordringen er så, at afgøre hvilken farve punkterne på billedplanen skal have, da dette afhænger af objektets egenskaber, samt hvilket udefrakommende lys der rammer objektet.

For at løse denne udfordring, benytter vi i dette projekt raytracing, der, som beskrevet under afsnit 5.2.2, bygger på at simulere lysstrålers interaktion med forskellige objekter i rummet. Hvordan dette fungerer er beskrevet i næste afsnit, hvor der er beskrevet en model for backwards raytracing.

Backwards raytracing I modsætning til en perspektiv projektion af et punkt på et plan, står backwards raytracing, hvor man i stedet for punktet i rummet, tager udgangspunkt i de lysstråler (rays), der danner billedet. Ved backwards raytracing følger man lysstrålerne baglæns og ser på, hvor stor en lysintensitet den pågældende lysstråle har efter den har interageret med objekterne i rummet. Ud fra dette farves det tilhørende punkt på billedet, og på den måde kan man rendere et helt billede. På figur 9 er det vist hvordan man kan konstruere en lysstråle ud fra et bestemt punkt på billedplanen, hvor lysstrålen er beskrevet ved en retningsvektor og et startpunkt.



Figur 9: Viser hvordan der kan opstilles retningsvektor mellem kameraets position C og punktet P på billedplanen, som sammen med startpunktet C beskriver lysstrålen fra trekanten i omvendt retning.

Retningsvektoren $\vec{r}$ for lysstrålen kan heraf beskrives som følgende.

$$\vec{r} = \vec{B} - \vec{C}$$

Hvor $\vec{B}$ og $\vec{C}$ er stedvektorer for hhv. punktet på billedplanen B og kameraets position C.

Lysstrålen kan på den måde beskrives ved følgende vektorfunktion.

$$\vec{l}\,(t) = \vec{r}t + \vec{C}$$

Hvor t er en skalar i $\mathbb{R}$.

For at finde ud af hvilken farve punktet på billedplanen B skal have, ser man hvordan lysstrålen rammer de forskellige objekter der skal renderes.

Der findes flere forskellige modeller for hvordan lysintensiteten for en lysstråle beregnes. En simpel model, er Phong-modellen, som opdeler lys i forskellige kategorier: ambient, diffuse og specular.

Phong-modellen Phong-modellen er en såkaldt *shading* funktion, som beskriver lyset fra punkter på et objekt på baggrund af lyskilden, objektet og kameraets synsvinkel [34]. Der findes flere forskellige variationer af phong-modellen. Da vi som vist i afsnit 6.1.2 kan arbejde med farvetemperaturer via RGB-værdier, så har vi valgt en variation af phong-modellen, som beskriver lys via RGB-værdier.

Ud fra modellen [35], kan der skrives følgende overordnede *shading* funktion.

$$\rho = \rho_a + \sum_{lights} (\rho_l + \rho_s) \tag{8}$$

Hvor ρ_a, ρ_l og ρ_s er hhv. *ambient*, *diffuse* og *specular* lys der er beskrevet kort herunder.

***Ambient* lys** repræsenterer det lys, som reflekteres rundt i rummet og rammer objekter ligeligt fra alle sider [35]. Formlen til at beregne dette er følgende [35].

$$\rho_a = m_a C A \tag{9}$$

Hvor m_a er den ambiente konstant, C er overfladens farve som RGB-værdi og A er den ambiente lysintensitet.

***Diffuse* lys** er det lys, som reflekteres ifølge *Lambert's Cosinuslov*. Ud fra loven kan følgende model anvendes til at udregne *diffuse* lys [35].

$$\rho_l = m_l C I max(\vec{I} \bullet \vec{n}, 0) \tag{10}$$

Hvor m_l er den Lambertianske konstant, I er det lyskildens lysintensitet, $\vec{I}$ og $\vec{n}$, er normaliserede vektorer, hhv. vektor fra overfladepunktet mod lyskilden og normalvektoren til overfladepunktet.

***Specular* lys** er det lys der spejles i objektets overflade, givet ved nedenstående formel [35].

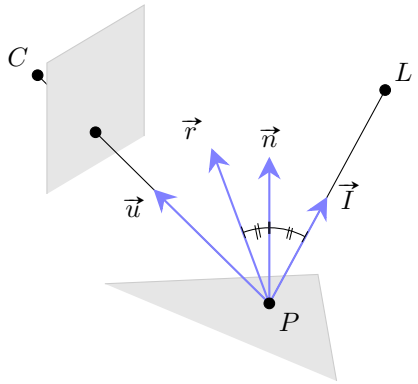
$$\rho_s = m_s S I max(\vec{r} \bullet \vec{u}, 0)^{m_{sp}} \tag{11}$$

$$S = m_{sm} C + (1 - m_{sm})(1, 1, 1) \tag{12}$$

Hvor m_s er den speculare konstant, m_{sp} er den speculare eksponent som beskriver størrelsen af lys der spejles i objektets overflade, S er en lineær interpolation mellem objektets farve og hvid, afhængig af objektets *metalness* m_{sm}. $\vec{u}$ og $\vec{r}$ er normaliserede vektorer, for hhv. vektor fra overfladepunktet mod kameraet og retningsvektor for det reflekterede lys beregnet som følgende.

$$\vec{r} = -\vec{I} + 2(\vec{I} \bullet \vec{n})\vec{n} \tag{13}$$

De anvendte vektorer til phong *shading*-funktionen er vist på figuren herunder.



Figur 10: Viser vektorer der anvendes i phong-modellen. $\vec{u}$ er vektor mod kameraet med position C, $\vec{n}$ er normalvektoren til objektet i punktet P, $\vec{I}$ er vektor mod lyskildens position L og $\vec{r}$ er vektor for det reflekterede lys.

6.1.4 Skæring mellem linje og trekant i rummet

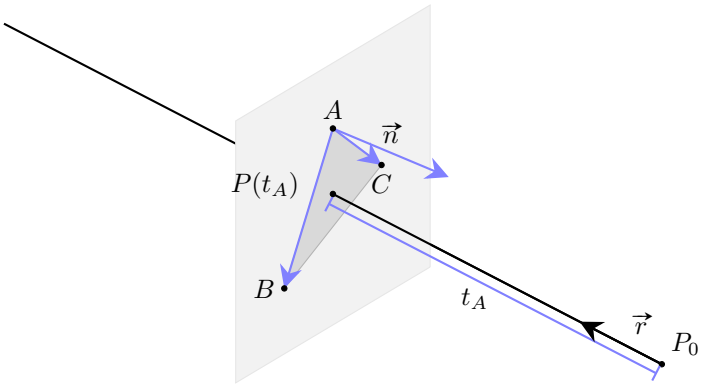
En essentiel del af raytracing er at bestemme skæring mellem en ray og en trekant i 3D-modellen. For at finde ud af om der er en skæring mellem en ray og en trekant, følger her en beskrivelse af først at finde strålens skæring med trekantens plan, og derefter at bestemme om punktet er indenfor trekantens tre sider.

Hvis rayen er parallel med planen, skærer det enten ikke planen noget sted, ellers ligger alle punkter i planen. En ray er parallel med en plan, hvis rayens retning er 90 grader relativt til planens normalvektor, dette er sandt hvis ligning 14 er opfyldt.[36]

$$\vec{r} \bullet \vec{n} = 0 \tag{14}$$

Planens normalvektor findes ved at krydse to vektorer fra en af trekantens hjørner til de to andre (se ligning 15).

$$\vec{n} = (B - A) \times (C - A) \tag{15}$$



Figur 11: Viser princippet bag bestemmelsen af en linjes skæring med planen for en trekant.

Hvis strålen ikke er parallel med planen kan der nu findes et punkt i planen, som strålen skærer. Alle vektorer i planen er ortogonale på normalvektoren, dermed kan der opstilles en ligning 17 til at beskrive betingelsen for et punkt strålen som også ligger i planen. Ved at substituere linjens ligning (ligning 16) ind og isolere t_A, som bruges til at finde punktet på strålens linje. Er $t_A > 0$, så er punktet også foran rayens udgangspunkt P_0 [36]

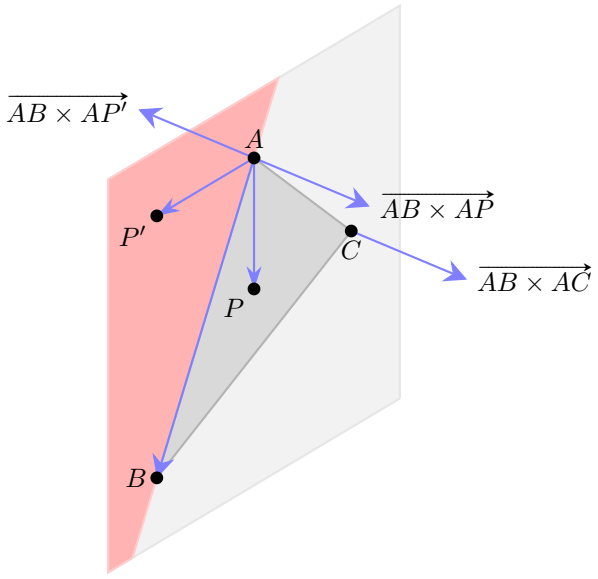
$$P(t) = \vec{r} \cdot t + \overrightarrow{P_0} \tag{16}$$

$$(P(t_A) - A) \bullet \vec{n} = 0 \tag{17}$$

$$(\vec{r} \cdot t_A + \overrightarrow{P_0} - A) \bullet \vec{n} = 0 \tag{18}$$

$$t_A \cdot \vec{r} \bullet \vec{n} + (\overrightarrow{P_0} - A) \bullet \vec{n} = 0 \tag{19}$$

$$t_A = -\frac{(\vec{A} - \overrightarrow{P_0}) \bullet \vec{n}}{\vec{r} \bullet \vec{n}} \tag{20}$$

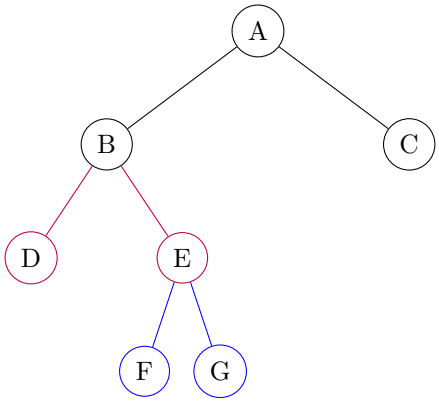


Figur 12: Viser krydsproduktet $\vec{n_a}$ af en af trekantens sider $\vec{a}$ og $\overrightarrow{AP}$.

For at et punkt i trekantens plan også er indenfor trekanten skal det ligge på den rigtige side af hver af trekantens sider $\{\overrightarrow{AB}, \overrightarrow{BC}, \overrightarrow{CA}\}$. Krydsproduktet af en side og en vektor til punktet i planen fra en af trekantens hjørner $\overrightarrow{AB} \times \overrightarrow{AP}$ vil være parralel med trekantens normal vektor (enten i samme retning eller direkte modsat). Siden af $\overrightarrow{AB}$ hvoraf punktet P befinder sig bestemmer retningen af krydsproduktsvektoren, Så derved kan vi vælge et andet punkt på den ønskede side af $\overrightarrow{AB}$ at sammenligne med. Derved skal retningen af $\overrightarrow{AB} \times \overrightarrow{AP}$ være lig $\overrightarrow{AB} \times \overrightarrow{AC}$. Samme metode anvendt på de to andre sider af trekanten ($\overrightarrow{BC}$, $\overrightarrow{CA}$) og hvis punktet P er på 'indersiden' af dem alle, da er P i trekanten.

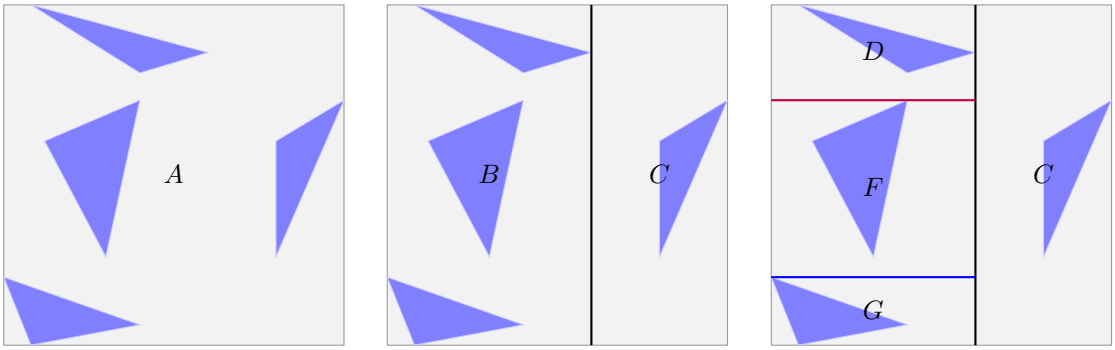
6.1.5 KD-træer

Et K-dimensionelt træ (KD-træ), er en abstrakt datastruktur som bl.a. andvendes til opdeling af 'rum'. Et KD-træ er repræsenteret som et binært træ, dvs. at hver gren i træet har yderligere to grene indtil bladene nåes (se figur 13) herunder.



Figur 13: Binært træ

Med et KD-træ opdeles et rum efter værdier, højere eller lavere end et taktisk udvalgt delingspunkt (se figur 14b), således at alle værdier i en gren, enten er højere eller lavere end den gren de udspringer fra. Denne opdeling fortsætter indtil værdierne i en gren er trivielle at behandle, herved kaldes en sådan gren for et blad. [37]



(a) Rod *(b) Første itteration* *(c) Færdigbygget KD-træ*

Figur 14: KD-træ

Et eksempel på opbygningen af et KD-træ ses i figur 14. Et KD-træ starter med roden (figur 14a), som omkranser hele data-mængden. Herefter opdeles alle non-trivielle grene i yderligere to grene (figur 14b), dette gentages indtil alle grene/blade er et simpelt tilfælde (figur 14c).

Opsummering

Ud fra teoridiskussionen kan der nu kortfattes hvordan teorierne bidrager til at opfylde de fem krav til løsningen i afsnit 5.1.2.

Rotationsmatricer giver den viden, som skal til for at rotere vektorer i rummet med en vilkårlig

vinkel. Dette er relevant når man skal flytte kameraets position, som hører til krav 2 i afsnit 5.1.2 om at lampens belysning skal kunne visualiseres fra flere forskellige vinkler.

Teorien om konvertering fra farvetemperatur til RGB giver os den algoritme, som skal til for at lave en farvetemperatur i kelvin om til en RGB-værdi. Dette er relevant i forhold til krav 3 i afsnit 5.1.2 om at kunne ændre pærens farvetemperatur.

Teorien fra 3D-model til billede omhandler matematiske modeller og beskrivelser af relevante teknologier og algoritmer. Teorien i dette er relevant for, at forstå nogle af grundprincipperne i en raytracer, heriblandt backwards raytracing og Phong-modellen. Som bruges til at opfylde krav 1 i afsnit 5.1.2.

Teorien om skæringen mellem linje og trekant i rummet omhandler en matematisk model, som kan anvendes til at udvikle en raytracer, hvor rays er beskrevet ved linjer i rummet og 3D-modellen er beskrevet med trekanter. Dette er også med til at opfylde krav 1 i afsnit 5.1.2.

Som afslutning på teoridiskussionen opstilles følgende nye krav til programudviklingen i forhold til kravene i afsnit 5.1.2:

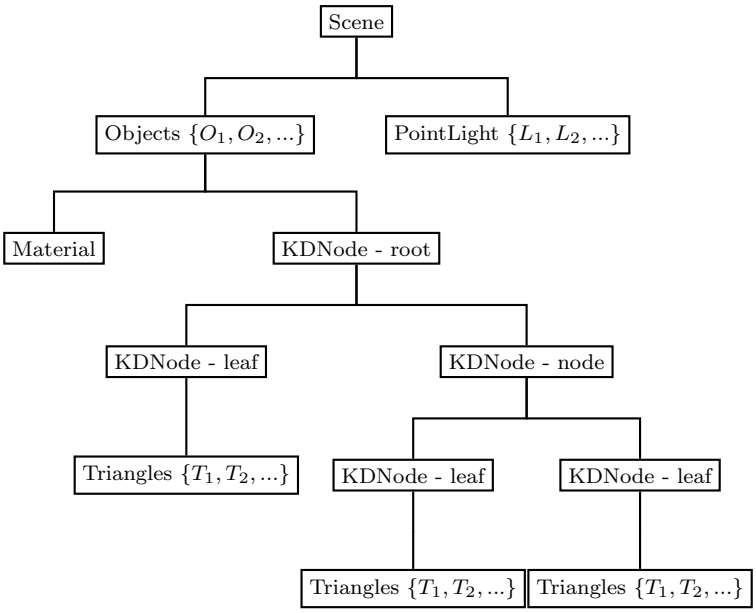
1. Programmet skal kunne modtage følgende input fra brugeren: 3D-fil af lampen og dens kontekst, billedets opløsning, pærens farvetemperatur og hvilken synsvinkel man ønsker at visualisere lampen og dens belysning fra.
2. Programmet skal ved brug af raytracing kunne fremstille et billede af lampen og dens belysning på baggrund af brugerinputtet. Derudover skal raytracerens renderingstid optimeres ved brug af KD-træer.
3. Programmet skal kunne gemme det renderede billede.

6.2 Udvikling

Formålet med dette afsnit er, at give en beskrivelse af programmets opbygning og struktur, samt ved brug af kodeeksempler og teori beskrevet i 6.1, at forklare hvordan funktionerne i programmet virker og hvad de gør. I afsnittet vil der være kodeeksempler på prototyperne til funktionerne og kodeeksempler på selve funktionen. Hele programmet kan findes online som elektronisk bilag.

6.2.1 Datastrukturer

I programmet bliver der brugt datastrukturer for at abstrahere over data. Nedenstående figur viser data-relationerne i programmet.



Figur 15: Viser relationer mellem sammenhængende datastrukturer.

Vektor En stor del af opgaven bygger på vektorer i rummet. Nedenstående kodeuddrag viser hvordan der er abstraheret over en vektor i programmet.

```
1 typedef struct _vector {
2   double x, y, z;
3 } Vector;
```

Kodeuddrag 2: Struct til vektor

På linje 2 i ovenstående kodeuddrag er det vist hvordan en 3D-vektors koordinater er beskrevet med typen double.

Ray For at benytte sig af raytracing-metoden er det nødvendigt at have en ray som man kan følge.

```
1 typedef struct _ray {
2   Vector initial_point, direction;
3 } Ray;
```

Kodeuddrag 3: Struct til ray

På linje 2 i ovenstående kodeuddrag kan det ses at en ray består af et startpunkt (initial_point) og en retning. Begge af disse er af typen Vector, som tidligere er beskrevet til at have et x-, y- og z-koordinat.

Triangle Enhver figur i programmet består af mange sammensatte trekanter. Jo flere trekanter billedet består af, des mere glat er overfladen.

En trekant udspændes af flere vektorer af tre hjørner hvor hver hjørne kaldes en vertex, der har en position og en normal.

```
1 typedef struct _vertex {
2   Vector position;
3   Vector normal;
4 } Vertex;
```

Kodeuddrag 4: Struct til vertex

I ovenstående kodeuddrag kan det ses at en vertex består af et stedvektor (linje 2), og en normalvektor (linje 3). Vertex normalerne kan bruges i forbindelse med implementationen af phong-modellen.

Som nævnt for ovenstående struct for en vertex, er alle objekter i programmet bestående af trekanter. Nedenstående kodeuddrag viser hvordan trekanter er implementeret i programmet.

```
1 #define VERTICES_IN_TRIANGLE 3
2
3 typedef struct _triangle {
4   Vertex *verticies[VERTICES_IN_TRIANGLE];
5   Vector edges[VERTICES_IN_TRIANGLE];
6 } Triangle;
```

Kodeuddrag 5: Struct til triangle

Denne struct viser, kort sagt, at en trekant består af 3 hjørner, altså 3 vertices, af typen Vertex, samt 3 edges, der er vektorer til trekantens 3 sider.

Point_lights En raytracers formål er at følge lysstrålerne, som udskydes fra et bestemt punkt, som i vores tilfælde er fra en pære.

```
1 typedef struct _pointlight {
2   Vector position;
3   Pixel color;
4   double intensity;
5   double radius;
6   int sampling_rate;
7 } PointLight;
```

Kodeuddrag 6: Struct til light

På linje to i ovenstående kodeuddrag kan det ses, at et lys har en given position, som bestemmes af ét 3D-koordinat. Dette lys har en farve, bestående af en RGB-værdi. Derudover har lyset også en intensitet og radius af typen double, samt sampling_rate af typen int. sampling_rate og radius bruges til at danne bløde skygger.

Materiale I virkeligheden har vi mange forskellige materialer, som har forskelligt udseende. En helt ny bil skinner når man lyser på den, mens en murstensvæg er helt mat. For at illustrere dette i programmet anvendes der forskellige konstanter til at beskrive et materiale. Konstanterne bruges i phong-modellen se afsnit 6.1.

```
1 typedef struct _material {
2   double ambient_coefficient;
3   double diffuse_coefficient;
4   double specular_coefficient;
5   int smoothness;
6   double metalness;
7 } Material;
```

Kodeuddrag 7: Struct til Material

Ambient_coefficient er værdien der fortæller hvor lyst objektet er, selvom der ikke er direkte lys på det. Den bevirker at hvis objektet er i skygge er det ikke helt sort. Diffuse_coefficient er skygge-værdien, der viser hvor stor indflydelse lys har på objektet. Specular_coefficient er værdien, der viser hvor meget objektet spejler igen. Denne værdi er høj for en nypoleret bil, men næsten 0 hvis det er en murstensvæg.

AABB AABB står for *axis aligned bounding box*, og er en kasse, hvis sider er parallelle med akserne. En boks består af to vektorer, som indeholder det laveste og største koordinatsæt for boksen. Den ene vektor er en stedvektor til det hjørne på boksen, som har de største x-,y- og z-værdier, hvor den anden vektor er en stedvektor til det hjørne på boksen med de laveste x-, y-, z-værdier.

Nedenstående kodeuddrag viser hvordan dette er gjort i programmet.

```
1 typedef struct _AABB {
2   Vector low, high;
3 } AABB;
```

Kodeuddrag 8: Struct til bounding boxes

De to punkter til boksen er angivet som stedvektorer af typen Vector (linje 2).

Plan En plan i rummet er beskrevet ved et punkt og en normalvektor til planen. Hvordan dette er gjort i programmet er vist i nedenstående kodeuddrag.

```
1 typedef struct _plane {
2   Vector normal;
3   Vector point;
4 } Plane;
```

Kodeuddrag 9: Struct til plan

Planens normalvektor er beskrevet med typen Vector (linje 2). Et punkt til planen er angivet som en stedvektor af typen Vector (linje 3).

Skæring For at abstrahere over de data der er relevante for en rays skæring med et objekt er der blevet lavet nedenstående datastruktur.

```
1 typedef struct _intersection {
2   Vector normal;
3   Material material;
4   Pixel color;
5   Triangle *triangle;
6   Ray ray;
7   double t;
8 } Intersection;
```

Kodeuddrag 10: Struct til intersection

På ovenstående struct kan det ses, at hver intersection har en mængde data, som bruges senere til phong. Hver intersection har en normalvektor, et materiale, en farve beskrevet ved typen Pixel. Den har derudover en 'triangle' til at finde hvilken trekant i træet, som den skærer i, en 'ray', som beskriver lysstrålen, der skærer i det givne punkt samt en tid t af typen double, der beskriver afstanden fra startpunktet.

Billede Vi har en Image-struct for at simplificere måden vi udskriver pixels på og forbedre læseligheden.

```
1 typedef struct _image {
2   unsigned int width, height;
3   Pixel **pixels;
4 } Image;
```

Kodeuddrag 11: Struct til Image

Vores Image-struct består af to doubles, der beskriver højden og bredden på vores billede i pixels. Samt et 2D-array med Pixels, der beskriver vores billede.

Kamera Som vist på figur 7 i afsnit 6.1.3 er kameraet beskrevet som dens position og orientering i rummet, samt en afstand, d, som er afstanden fra kameraets position til billedplanen. Derudover har kameraet en højde og en bredde, som definerer billedets opløsning. Nedenstående kodeuddrag viser hvordan dette er gjort i programmet.

```
1 typedef struct _camera {
2   Vector up;
3   Vector right;
4   Vector forward;
5   Vector position;
6   unsigned int width, height;
7   double distance;
8 } Camera;
```

Kodeuddrag 12: Struct til kamera

Kameraets orientering kan beskrives som tre vektorer af typen Vector (linje 2-4). Kameraets position kan angives som en stedvektor af typen Vector (linje 5). Opløsningen er angivet som positive heltal af typen unsigned int (linje 6). Afstanden mellem kameraets position og billedplanen er beskrevet vha. distance (linje 7).

Scene Scenen er den virtuelle verden, som skal visualiseres. I programmet er scenen beskrevet på følgende måde.

```
1 typedef struct _scene {
2   Object **objects;
3   unsigned int n_objects;
4   PointLight **lights;
5   unsigned int n_lights;
6   Pixel ambient_intensity;
7 } Scene;
```

Kodeuddrag 13: Struct til scene

En scene består af en liste af objekter (linje 2-3), en liste af lys (linje 4-5), samt den ambiente lysintensitet, der bruges i Phong-modellen.

Objekt Objektet er det man kan se, når billedet bliver renderet af raytraceren. I programmet er objektet beskrevet på følgende måde.

```
1 typedef struct _object {
2   Vertex *verticies;
3   int n_verticies;
4   Triangle *triangles;
5   int n_triangles;
6   Pixel color;
7   Material material;
8   KDNode root;
9 } Object;
```

Kodeuddrag 14: Structs til objektet

Et objekt består af en liste af verticies (linje 2-3), en liste af trekanter (linje 4-5), samt en farve på objektet, koefficienterne til det materiale den er lavet af og en KD-rod.

K-dimensionelt træ Da programmet er bygget op efter at skulle kunne implementeres på en hjemmeside, er renderingen nødt til at være hurtig. Dette opnås ved hjælp af optimering, hvor der i programmet er brugt KD-træer.

```
1 typedef struct _KDNode {
2   struct _KDNode *low, *high;
3   AABB box;
4   Triangle **triangles;
5   int n_triangles;
6 } KDNode;
```

Kodeuddrag 15: Struct til KDNode

Et KD-træ består af to nodes (linje 2), hvilket er forgreninger, den består af en AABB box (linje 3) og en liste af trekanter (linje 4-5).

6.2.2 Overordnet programstruktur

Main-funktionen er den første funktion i programkoden der eksekveres. Det er således i funktionen main, at programmets andre funktioner kaldes og eksekveres. Programmets main funktion er vist nedenfor:

```
1  int main(int argc, char* argv[]) {
2    Scene *scene;
3    Camera *camera;
4    Image *image;
5    Configuration conf;
6
7    unsigned long t0 = time(NULL);
8    conf = create_configuration();
9
10   if(input_parse(argc, argv, &scene, &camera, &conf) == 0) {
11     return -1;
12   }
13
14   image = raytracer_render(scene, camera);
15
16   image_write(image, conf.out_file);
17
18   free_camera(camera);
19   free_scene(scene);
20
21   printf("%lus\n", time(NULL) - t0);
22   return 0;
23 }
```

Kodeuddrag 16: Main

På linje 1 i ovenstående kodeuddrag, kaldes main, med parametrene argc og argv. argv indeholder en liste af programparametre som f.eks. favetemperaturen, som ønskes anvendt af programmet. argc er blot antallet af programparametre som argv indeholder. På linje 10 behandles programparametrene og der testes om der er fejl i inputtet, hvilket kunne være en manglende 3D-model. Dette gøres gennem input_parse-funktionen i input.c. På Linje 14 kaldes funktionen raytracer_render som renderer billedet. Linje 7 og 21 bruges udelukkende til teknisk anvendelse, for at kunne se hvor lang tid programmet har taget om at fuldføre.

Input Som input kræver programmet filnavnet på en 3D-model i PLY formatet, derudover modtager programmet en række valgfrie input-parametre:

- -h [INT]: Et heltal, som angiver højden af det resulterende billede i pixel.
- -w [INT]: Et heltal, som angiver breden af det resulterende billede i pixel.
- -t [INT]: Et heltal, som angiver farvetemperaturen af lyskilder i modellen hvis farve er angivet som sort.

- -H [FLOAT]: Et decimaltal som, i radianer, angiver en horisontal rotation om en sort lyskilde.
- -V [FLOAT]: Et decimaltal som, i radianer, angiver en vertikal rotation om en sort lyskilde.
- -o [STRING]: En tekststreng, som angiver placering og navnet på det resulterende billede.

Filstilen til 3D-modellen skal være den første programparameter og skal være i et tilpasset PLY format. Data-input har ikke været fokus for opgaven og et tilpasset PLY var blot den nemmeste måde, hurtigt at få et system op at køre, så der kunne udvikles forskellige modeller til test. Af samme grund har input-parseren heller ikke været et fokus. Input-parseren kan findes i input.c, se elektronisk bilag.

6.2.3 Beskrivelse af raytracer

I dette afsnit vil de væsentlige funktioner til raytraceren blive beskrevet. Formålet med afsnittet er, at få en forståelse for hvordan teorien i afsnit 6.1, er anvendt og håndteret i programudviklingen.

Rendering For at rendere et billede af lampen og dens belysning, er der lavet en funktion raytracer_render, der modtager scenen, dvs. samlingen af alle 3D-objekter og lys, samt modellen for det kamera, som billedet skal dannes ud fra. Funktionen er vist herunder.

```
1  Image *raytracer_render(Scene *scene, Camera *camera) {
2    int x, y;
3    Image *image;
4    Ray ray;
5
6    image = new_image(camera->width, camera->height);
7
8    /* For each pixel */
9    for(x = 0; x < camera->width; x++) {
10     for(y = 0; y < camera->height; y++) {
11       ray = raytracer_calculate_ray(x, y, camera);
12
13       /* Trace ray and assign result to pixel */
14       image->pixels[x][y] = raytracer_trace(ray, scene);
15     }
16     /* print progress percentage */
17     printf("%.1f\n", ((double)x + 1) / camera->width * 100);
18   }
19
```

```
20   return image;
21 }
```

Kodeuddrag 17: Funktionen, der rendere billedet af scenen med et kameras perspektiv

Funktionen vist herover, danner en ray for hver pixel i billedet. Rayen sendes videre sammen med scenen til funktionen raytracer_trace, som returnerer, hvilken farve den pågældende pixel på billedet skal have. Til sidst returneres det endelige billede.

Tracer Funktionen raytracer_trace er den funktion som starter raytraceren samt returnerer en pixelfarve, hvis en ray skærer med et objekt i scenen. Funktionen er vist herunder.

```
1  Pixel raytracer_trace(Ray ray, Scene *scene) {
2    Intersection intersection = create_intersection();
3    Pixel pixel;
4
5    pixel = (Pixel){0, 0, 0};
6
7    /* If ray intersects with scene: */
8    if(raytracer_scene_intersection(ray, scene, &intersection)) {
9      /* Shade pixel */
10     pixel = raytracer_phong(intersection, scene);
11   }
12
13   return pixel;
14 }
```

Kodeuddrag 18: Funktionen raytracer_trace

Funktionen initialiserer en skæring med værdien -1 ved at kalde funktionen create_intersection (linje 2), og en pixel initialiseres til at indeholde RGB-værdien for farven sort (linje 5). Der tjekkes efterfølgende om den pågældende ray skærer med et objekt i scenen (linje 8), hvis den gør det så tildeles der en RGB-værdi (linje 10), som returneres til sidst i funktionen.

Skæring med scene Funktionen raytracer_scene_intersection tjekker om en ray skærer med et objekt.

```
1  int raytracer_scene_intersection(Ray ray, Scene *scene,
2                                   Intersection *intersection) {
3    int i;
4    Intersection temporary_intersection;
5
6    temporary_intersection = create_intersection();
```

```
7
8   /* For each object in scene: */
9   for(i = 0; i < scene->n_objects; i++) {
10    /* If ray intersects with object: */
11    if(raytracer_object_intersection(ray, scene->objects[i],
12       &temporary_intersection))
13      /* Reassign intersection if current intersection is closer */
14      if(temporary_intersection.t < intersection->t || intersection->
      t == -1)
15        *intersection = temporary_intersection;
16  }
17  return temporary_intersection.t > 0;
18 }
```

Kodeuddrag 19: Funktionen raytracer_scene_intersection

I funktionen tjekkes der om en ray skærer et objekt, hvis dette er tilfældet så returneres der information om denne skæring (linje 11). Der tjekkes efterfølgende om den nye skæring er tættere på udgangspunktet end de førhenværende skæringer og den tætteste skæring gemmes (linje 14 - 15). Herefter returnerer funktionen sandt, hvis tiden, som beskriver afstanden fra startpunktet til skæringspunktet, er over 0, og falsk hvis den ikke er.

Skæring med objekt Funktionen raytracer_object_intersection er en funktion, der undersøger om der er en skæring mellem en ray og et objekt. Funktionen ses herunder:

```
1 int raytracer_object_intersection(Ray ray, Object *object,
    Intersection *intersection) {
2   double i, j;
3   Intersection temporary_intersection;
4
5   temporary_intersection = create_intersection();
6
7   /* if ray intersects with object's aabb: */
8   if(intersection_ray_aabb(ray, object->root.box, &i, &j)){
9     /* if ray intersects with triangle in object's kd-tree */
10    if(raytracer_kdtree_intersection(ray, &(object->root), &
    temporary_intersection)){
11      temporary_intersection.color = object->color;
12      temporary_intersection.material = object->material;
13      if(temporary_intersection.t < intersection->t || intersection->
    t == -1)
14        *intersection = temporary_intersection;
```

```
15      }
16    }
17    return temporary_intersection.t > 0;
18 }
```

Kodeuddrag 20: raytracer_object_intersection

På linje 8 i ovenstående kodeuddrag kan man se, at funktionen først tjekker om en ray skærer med AABB, og hvis dette er sandt tjekker den om ray'en skærer i træet. Hvis begge af disse tilfælde er sande, får pixlen i skæringspunktet en farve bestemt af, hvor og hvordan rayen rammer objektet. Derudover bliver værdierne af materialet gemt, som kan bruges senere. Herefter returnerer funktionen sandt, hvis tiden, som beskriver afstanden fra startpunktet til skæringspunktet, er over 0, og falsk hvis den ikke er.

Skæring med KD-træ Funktionen raytracer_kdtree_intersection bruges til at undersøge om rayen skærer med KD-træet og i så fald hvor. Formålet med denne funktion er at optimere programmet, og undgå at lave unødvendige udregninger.

Hele funktionen kan findes på elektronisk bilag i filen kdnode.c

```
1 if(kdnode_is_leaf(node)) {
2   /* test intersection with triangles */
3   for(i = 0; i < node->n_triangles; i++) {
4     if(raytracer_triangle_intersection(ray, node->triangles[i], &
    temporary_intersection)) {
5       /* if first intersection or closer than current intersection:
    */
6       if(temporary_intersection.t < intersection->t || intersection->
    t == -1) {
7         *intersection = temporary_intersection;
8         intersection->triangle = node->triangles[i];
9       }
10    }
11  }
12 }
13 }
```

Kodeuddrag 21: Uddrag af funktionen raytracer_kdtree_intersection

På ovenstående kodeuddrag undersøger funktionen, om skæringen med træet sker i et af bladene, dette ses på linje 1. Et blad er 'bunden' af træet, altså det punkt, hvor en kasse ikke længere kan inddeles i flere kasser. Der er tre krav til, hvornår vi ender vores inddeling af flere kasser, og istedet slutter og kalder det for et blad. Første krav er, at hvis antallet af trekanter i en kasse er mindre end eller lig med to er noden et blad. En kasse med to eller færre trekanter er nemt og

hurtigt for computeren at udregne, og derfor behøves den ikke at inddeles yderligere. Det andet krav er, hvis over halvdelen af to kassers trekanter ligger i begge kasser. Hvis dette er tilfældet kaldes begge kasser for blade, da mindst halvdelen af begge kassers fælles trekanter er i begge kasser og der derfor ikke er brug for at inddele i flere yderligere kasser. Det tredje, og sidste, krav er, hvis der er mere end 30 niveauer. Dette krav er nødvendigt for at programmet ikke kører i en evighedsløkke, og bliver ved med at tjekke, selvom det ikke er nødvendigt. En tilfælde, hvor det tredje punkt kunne være nødvendigt er, hvis ét sted i kassen indeholder 10 punkter, og dermed ikke kan opdeles, så ville programmet uden tredje krav blive ved med at lave den samme opdeling igen og igen.

Hvis roden (hovedkassen) er et blad, findes de nærmeste skæringer først som vist på linjerne 6-8 i kodeuddrag 21, og der findes skæring med samtlige trekanter som vist på linje 3 i kodeuddrag 21, medmindre nogle trekanter ligger helt i skygge for andre.

```
1  else {
2      /* test intersection recursively */
3      retl = intersection_ray_aabb(ray, node->low->box, &t_minl, &
       t_maxl);
4      reth = intersection_ray_aabb(ray, node->high->box, &t_minh, &
       t_maxh);
5
6      if(retl && reth) { /* Intersecting both sub-nodes */
7        if(t_minh < t_minl) { /* low node is hit first */
8          if(!raytracer_kdtree_intersection(ray, node->high,
       intersection)) {
9            raytracer_kdtree_intersection(ray, node->low, intersection)
       ;
10         }
11       } else if(t_minh == t_minl) {
12         raytracer_kdtree_intersection(ray, node->low, intersection);
13       }
14     } else if(t_minh == t_minl) {
15       raytracer_kdtree_intersection(ray, node->low, intersection);
16       raytracer_kdtree_intersection(ray, node->high, intersection);
17     } else { /* high node is hit first */
18       if(!raytracer_kdtree_intersection(ray, node->low, intersection)
       ) {
19         raytracer_kdtree_intersection(ray, node->high, intersection);
20       }
21     }
22   } else if(retl) { /* only low node is hit */
23       raytracer_kdtree_intersection(ray, node->low, intersection);
```

```
24    } else { /* only high node is hit */
25        raytracer_kdtree_intersection(ray, node->high, intersection);
26    }
27 }
```

Kodeuddrag 22: Fortsættelse af uddraget fra funktionen raytracer_kdtree_intersection

Hvis roden ikke er et blad og hvis ray skærer i begge kasser, undersøges hvilken kasse, der er nærmest og funktionen kaldes rekursivt for denne kasse, dette kan ses på linje 6-18 i ovenstående kodeuddrag. Hvis rayen går gennem den nærmeste kasse uden at ramme et objekt tjekkes for den fjerneste kasse. Hvis rayen kun skærer én af kasserne, tjekkes simpelt kun for den kasse, som set på linje 19-24.

Funktionen returnerer tiden for skæringen med trekanten.

Skæring med trekant Funktionen raytracer_triangle_intersection bruges til at undersøge om der er skæring mellem en ray og en trekant. Funktionen ses herunder

```
1  int raytracer_triangle_intersection(Ray ray, Triangle *triangle,
       Intersection *intersection) {
2    double denominator, time;
3    Plane plane;
4    Vector triangle_normal;
5
6    time = -1;
7    triangle_normal = vector_normalize(vector_cross(triangle->edges[0],
8                                                    triangle->edges[1])
       );
9    plane = create_plane(triangle->verticies[0]->position,
       triangle_normal);
10
11   /* get ray's intersection time with triangle plane: */
12   if(intersection_ray_plane(ray, plane, &time) && time > 0) {
13     int i;
14     Vector point;
15     point = ray_get_point(ray, time);
16
17     /* if intersection point inside triangle: */
18     for(i = 0; i < VERTICES_IN_TRIANGLE; i++)
19       if(vector_dot(triangle_normal, vector_cross(triangle->edges[i],
20           vector_subtract(point, triangle->verticies[i]->position))) <
       0)
21         return 0;
```

```
22      intersection->t = time;
23      intersection->ray = ray;
24      if(vector_dot(ray.direction, triangle_normal) > 0)
25        triangle_normal = vector_scale(triangle_normal, -1);
26      intersection->normal = triangle_normal;
27      return 1;
28    }
29
30    return 0;
31  }
```

Kodeuddrag 23: Funktionen, der finder skæring med trekant

Funktionen finder en normaliseret normalvektor ud fra to vektorer, som beskriver siderne i trekanten (linje 7-8). Derefter findes trekantens plan ud fra en stedvektor og en normalvektor (linje 9). Der undersøges nu, om der er skæring mellem trekant og plan (linje 13), hvis der er skæring tjekkes der om punktet ligger i trekanten (linje 13 - 22). Hvis punktet ligger indenfor trekanten (linje 25) skaleres der med -1 for, at få normalvektoren til trekanten til at vende ud mod kameraet (linje 26) og værdien gemmes i en output parameter (linje 27).

Phong pixelfarve Når skæringen er fundet ved ovenstående funktioner, returneres skæringen og tilhørende data gemt i en Intersection til funktionen raytracer_trace, som kalder raytracer_phong med scenen og en Intersection, som parametre. Prototypen for funktionen er vist herunder.

```
1 Pixel raytracer_phong(Intersection intersection, Scene *scene);
```

Kodeuddrag 24: prototypen til funktionen der beregner pixelfarven på baggrund af data fra skæring med scenen.

I funktionen raytracer_phong initialiseres først de variable til phong-modellen, som er beskrevet i afsnit 6.1.3.

```
1 m_a = intersection.material.ambient_coefficient;
2 m_l = intersection.material.diffuse_coefficient;
3 m_s = intersection.material.specular_coefficient;
4 m_sp = intersection.material.smoothness;
5 m_sm = intersection.material.metalness;
6 vN = intersection.normal;
7 pC = intersection.color;
8 pA = scene->ambient_intensity;
9 diffuse = create_pixel(0.0, 0.0, 0.0);
10 specular = create_pixel(0.0, 0.0, 0.0);
```

Kodeuddrag 25: Initialisering af variabler i raytracer_phong.

Herefter beregnes *ambient* lys efter formlen i udtryk 9 under afsnit 6.1.3.

```
1 ambient = pixel_multiply(pixel_scale(pC, m_a), pA);
```

Kodeuddrag 26: Beregning af ambient lys i raytracer_phong.

Efter det *ambient* lys, beregnes vektorer til beregning af *diffuse* og *specular* lys.

```
1 intersection_point = ray_get_point(intersection.ray, intersection.t);
2 vU = vector_scale(intersection.ray.direction, -1.0);
3 pS = pixel_add(pixel_scale(pC, m_sm), pixel_scale(create_pixel(1.0,
      1.0, 1.0), (1 - m_sm)));
```

Kodeuddrag 27: Beregning af skæring, vektor $\vec{U}$ og pixel S i raytracer_phong.

Via en for-løkke summeres *diffuse* og *specular* lys fra alle lys i scenen, som vist herunder.

```
1  for(i = 0; i < scene->n_lights; i++) {
2    sampled_light_intensity = raytracer_get_light_intensity(scene->
       lights[i],
3                                                      intersection_point,
       scene);
4
5    pI = pixel_scale(scene->lights[i]->color, sampled_light_intensity);
6    vI = vector_normalize(vector_subtract(scene->lights[i]->position,
7                          intersection_point));
8    vR = vector_normalize(vector_add(vector_scale(vI, -1),
9                          vector_scale(vN, vector_dot(vI, vN) * 2)));
10
11   /* diffuse light =  m_l * MAX(vI * vN, 0) * pC * pI*/
12   diffuse = pixel_add(diffuse, pixel_multiply(pixel_scale(pC,
13                       m_l * MAX(vector_dot(vI, vN), 0)), pI));
14
15   /* specular light = m_s * MAX(-vR * vU, 0) ^ m_sp * pI * pS */
16   specular = pixel_add(specular, pixel_multiply(pS, pixel_scale(pI,
17                        m_s * pow(MAX(vector_dot(vR, vU), 0), m_sp))))
       ;
18 }
```

Kodeuddrag 28: Beregning og summering af diffuse og specular lys fra scenens lys.

I kodeuddrag 28 linje 2 indlæses intensiteten fra den pågældende lyskilde. Lysintensiteten beregnes i funktionen raytracer_get_light_intensity, ved at teste, hvor stor en procentdel af et bestemt antal tilfældige rays fra lyskilden, som ikke afskærmes på vej mod skæringspunktet ved objektet. Dette muliggør bløde skygger, da hver lyskilde kan have en radius, i stedet for blot et bestemt

punkt. Ud fra lysintensiteten og lyskildens farve beregnes I (Linje 4). Herefter beregnes $\vec{I}$ og $\vec{R}$, som er hhv. vektor mod lyset og refleksionsvektoren. I linje 12-17, lægges de pågældende *diffuse* og *specular* lys til de samlede summer. Til sidst i raytracer_phong returneres summen af *ambient*, *diffuse* og *specular* lys, som er den farve det pågældende skæringspunkt på objektet får, når billedet renderes.

```
1 /* return ambient + diffuse + specular */
2 return pixel_add(ambient, pixel_add(diffuse, specular));
```

Kodeuddrag 29: summen af ambient, diffuse og specular lys returneres fra raytracer_phong.

Placering af kamera Funktionen camera_look_at_point er en funktion, der sørger for, at vi nemt og hurtigt kan bestemme en position og vinkel til kameraet hvis vi vil kigge på et bestemt objekt. For at kigge på et punkt skal vi vide hvor langt væk kameraet skal være fra punktet, hvilken vinkel vi vil se punktet fra og punktets position. Disse variabler er givet ved en vektor, der indeholder punktets position og tre doubles, der definerer de to vinkler og afstanden fra punktet til kameraet. Vi sætter kameraets position til at være vores afstand ud af y aksen, og 0 ud af x og z aksen. Så roterer vi punktet ved hjælp af vores camera_set_angle funktion som drejer kameraet om sig selv for at vi kigger den rigtige vej. Så drejer vi kameraet de angivne radianer om x og z aksen og til sidst lægger vi punktet vi vil se på, til kameraets position.

```
1 void camera_look_at_point(Camera *camera, Vector point, double
      distance, double vertical_angle, double horizontal_angle) {
2   /* Resetting camera position to prepare rotation */
3   camera->position = (Vector){0, -distance, 0};
4
5   /* Rotate camera around itself */
6   camera_set_angle(camera, -vertical_angle, horizontal_angle);
7
8   /* Rotate camera around x and z axis and add to point we wanna look
        at */
9   camera->position = vector_rotate_around_xz(camera->position, -
      vertical_angle, horizontal_angle);
10  camera->position = vector_add(camera->position, point);
11 }
```

Kodeuddrag 30: Kode-uddrag fra camera.c: camera_look_at_point

Opsummering

I afsnittet er der, ved brug af udvalgte kodeuddrag, blevet dokumenteret hvordan programmet er udviklet. Ud fra kravene opstillet i slutningen af teoridiskussionen (afsnit 6.1 forventer vi nu, at programmet kan:

1. Modtage input fra brugeren heraf en 3D-fil, billedets opløsning, en farvetemperatur for pæren og en kameraposition.
2. Rendere et billede ud fra den angivne input.
3. Gemme billedet.

For at teste om programmet opfylder dette, er der i næste afsnit en beskrivelse af en række test af programmet.

6.3 Test af programmet

Som en del af problemløsningen udføres der test på det udviklede program. Formålet med testafsnittet er, at undersøge om programmet lever op til de programkrav, som blev opstillet i slutningen af teoridiskussion, 6.1. Testene vil være med til at belyse eventuelle fejl og mangler i programmet.

Først testes det om programmet kan rendere et billede af en lampe og dens belysning. Dette gøres ved at sammenligne et billede taget med et fysisk kamera, med et billede renderet af programmet.

På figur 16a er der vist et billede taget med et mobilkamera (Samsung Galaxy S3), hvorimod billedet på figur 16b er renderet af programmet på baggrund af en tilnærmet model af lampen, med nedenstående programparametre.

```
./trace model.ply -V 0.785 -t 2700 -w 540 -h 960
```

(a) Fotografi

(b) Renderet billede

Figur 16: Viser billede taget med mobilkamera (a) og billede renderet af programmet (b). For de to billeder gælder at pærens farvetemperatur er angivet som 2700K.

For at teste de forskellige programparametre, som kan indtastes i programmet, er der herunder billeder af test med forskellige programparametre.

6.3.1 Test af farvetemperatur

(a) Farvetemperatur: 2700K

(b) Farvetemperatur: 5000K

Figur 17: Viser billede renderet af programmet med to forskellige farvetemperaturer.

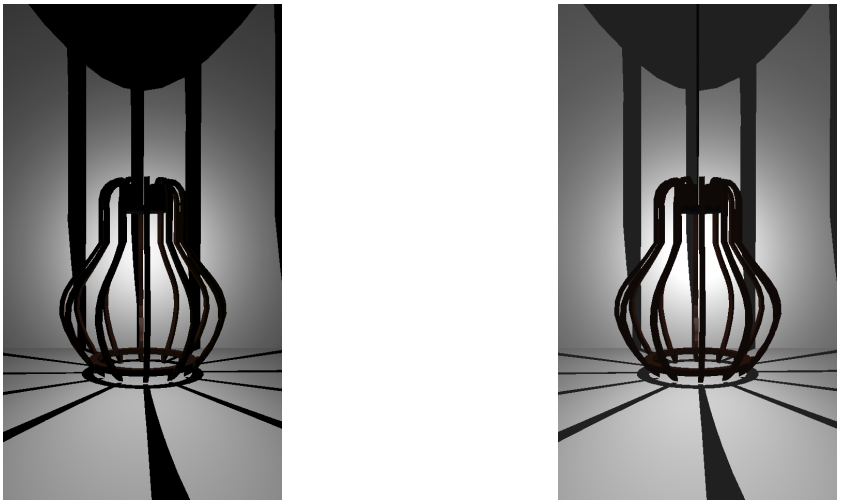
6.3.2 Test af synsvinkler

(a) Horisontal vinkel: 0^o
Vertikal vinkel: 45^o

(b) Horisontal vinkel: 45^o
Vertikal vinkel: 0^o

Figur 18: Viser billeder renderet af programmet, med to forskellige vinkler.

6.3.3 Test af optimering



(a) Før optimering: 658 sek. *(b) Efter optimering: 53 sek.*

Figur 19: Viser et billede angivet med renderingstider for programmet henholdsvis før og efter implementationen af KD-træer. Begge billeder er renderet uden bløde skygger.

Med den gamle version af programmet opnås der en renderingstid på 658 sekunder, og med den nye version af programet opnås der en renderingstid på 53 sekunder. Der er altså efter implementationen af KD-træer, i dette tilfælde, opnået en speedup på $658sek./53sek. \approx 12$. Renderingen af de to billeder er foretaget på samme hardware, da renderingstiden bla. er afhængig af hvilken hardware programmet bliver kørt på. Hvis billederne renderes på anden hardware vil renderingstiden derfor ændres, men speedup skulle gerne være det samme.

Opsummering

Som vist på figur 16b opfylder programmet krav 1-3 fra afsnit 6.2.3, da den kan rendere og gemme et billede af en lampe og dens belysning på baggrund af indtastede input der indeholder 3D-fil, farvetemperatur, synsvinkel og opløsningen af billedet. Derudover er der, som vist, sket en optimering af programmets renderingstid efter implementeringen af KD-træer. Som det er vist på figur 16 er der en afvigelse mellem billedet af den rigtige lampe og det renderede billede af en model af lampen. Denne afvigelse diskuteres i næste afsnit sammen med rapportens resterende resultater.

7 Diskussion

Formålet med dette afsnit er, at diskutere hvorvidt vores løsning opfylder de krav, der blev opstillet til løsningen i afsnit 5.1.

Det første krav var, at programmet skulle kunne vise et billede af en lampe og dens belysning. Dette krav er som vist under afsnit 6.3 opfyldt, da programmet kan modtage en 3D-fil af en lampe samt en kontekst, og ud fra dette rendere og gemme et billede af lampens belysning i konteksten. På figur 16 kan det ses, at der er en afvigelse mellem billedet af den virkelige lampe og billedet af den renderede 3D-model for lampen. Dette kan til dels skyldes præcisionen af de mål, der blev foretaget for at konstruere 3D-modellen af lampen. Derudover er det en simpel model (phong-modellen) som, i vores program, ikke tager højde for alle lysfænomener såsom refleksion og refraktion. Dette gør, at programmet har en begrænsning i forhold til den realisme, billederne af lampen kan have. Et eksempel som programmet ikke kan rendere, er en lampe, som har en reflekterende overflade, som gør at man kan se en spejling af andre objekter i lampens overflade. I stedet for phong-modellen, kunne det tænkes at der var andre modeller, som også kunne anvendes. Derudover kunne modellen udvides, så den kunne håndtere refleksioner og refraktioner, og på den måde hæve kvaliteten af det renderede billede. Dette er dog ikke nået i dette projekt grundet tidsmangel. Derudover vil højere realisme sandsynligvis også øge renderingstiden, hvilket vil kræve at programmet optimeres yderligere. Der er derfor en balancegang mellem realisme og renderingstid, som skal tages højde for, hvis programmet skulle videreudvikles.

Det andet krav var, at det i programmet skulle være muligt at ændre synsvinklen hvorfra lampen og dens belysning visualiseres fra. Dette krav er løst som vist på figur 18 under afsnit 6.3 ved at modtage to vinkler, der indikerer hvorfra lampen skal ses. I den 3D-model, som programmet indlæser, er lampens lyskilde indikeret med et sort lys. Det er positionen af dette sorte lys, som bestemmer hvorhen det virtuelle kamera er rettet mod. Dette kan være problematisk, hvis man f.eks. ønsker at visualisere en lyskæde, hvor det er svært at afgøre hvilket lys kameraet skal rettes mod. Dette kunne eventuelt løses ved at finde et gennemsnitspunkt af alle sorte lys positioner.

Det tredje krav var, at det i programmet skulle være muligt at ændre farvetemperaturen for pæren i lampen. I programmet er pæren beskrevet ved en radius, en lysintensitet og en farvetemperatur. Dette er blot en tilnærmet model af en pære. Der tages ikke højde for pærens form eller type. Derudover visualiseres selve pæren ikke i programmet, men kun selve dens lys. For at øge realismen vil dette være nødvendigt at tage højde for pærens form og type. Farvetemperaturen for pærens lys er implementeret i programmet ved at omdanne farvetemperaturens værdi i kelvin til RGB-værdier ud fra algoritmen, som er beskrevet i afsnit 6.1.2. Når 3D-filen indlæses overskrives alle sorte lyskilder med den pågældende farvetemperatur. Ulempen ved denne løsning er, at algoritmen der konverterer farvetemperatur i kelvin til RGB-værdier kun bygger på regression af ét datasæt, og der kan derfor være usikkerheder som gør, at billedet af lampen og dens belysning med en bestemt farvetemperatur ikke stemmer overens med den reelle farvetemperatur for lampen. Afvigelsen kan ses på figur 16, hvor der er en forskel mellem de to billeders farvenuancer. Dette kan enten være en fejl i programmet eller en uoverensstemmelse

med den farvetemperatur, som er angivet på pæren. For at afgøre hvor fejlen ligger, vil det være nødvendigt at foretage flere test og målinger, hvis der ønskes en højere overenstemmelse med den reelle farvetemperatur.

Det fjerde krav var, at programmet skulle kunne rendere et billede af en lampe og dens belysning på en tid der er praktisk anvendelig. Som dokumenteret i afsnit 6.3 ses der, at selv efter løsningen er blevet optimeret, så kan det tage adskillige minutter at rendere et billede. Det tog f.eks. 5305s at rendere figur 16b. Dette udelukker, at vores løsning er praktisk anvendelig i form af, at der skal renderes et nyt billede så snart brugeren ændrer synsvinklen. Dog kunne man benytte programmet til, at forudrendere billeder af lampen fra forskellige synsvinkler, som lagres på en server, der herefter kan vises på en e-butikshjemmeside. Dette begrænser dog antallet af synsvinkler som lampen kan visualiseres fra, da mange synsvinkler og farvetemperaturer giver mange kombinationer og dermed lang renderingstid.

Det femte krav var, at programmet skulle kunne implementeres på en e-butikshjemmeside således at billederne af lampen og dens belysning kan visualiseres for kunderne. Dette krav blev ikke opfyldt da fokus har været at visualisere belysningen fra lamper, og ikke selve brugerfladen for visningen af de renderede billeder på en e-butikkenshjemmeside.

Opsummering

Ud fra ovenstående kan der nu kort opsummeres, at krav 1-3 er løst, da programmet kan rendere et billede af en lampe og dens belysning, fra forskellige synsvinkler og med forskellige farvetemperaturer. Krav 4 er delvist løst, da det stadigvæk er muligt for simple renderinger at fremstille billeder på en tid, som gør at programmet er praktisk anvendelig til renderinger af billeder til en e-butikshjemmeside. Krav 5 er ikke løst, da fokus har været at lave den del der renderer billedet, og der mangler derfor den brugerflade, som gør at billederne kan vises på en e-butiks hjemmeside.

8 Konklusion

Formålet med dette afsnit er, at opsummere hvordan den endelige problemformulering blev udarbejdet. Derudover konkluderes der hvorvidt den udviklede løsning besvarer den endelige problemformulering.

I problemanalysen, blev der argumenteret for det initierende problems relevans. I denne forbindelse blev der antaget, at det initierende problem eksisterede, på baggrund af egne erfaringer, diskussion med lektor Lars Peter Jensen og en udtagelse fra en belysningskonsulent for en dansk lampebutik. Ønsket var at bekræfte antagelsen ved at udføre en spørgeskemaundersøgelse. Da denne undersøgelse aldrig blev fuldført, er det ikke bevist, at det initierende problem eksisterer.

Efter argumentationen for problemets relevant blev begreber, interessenter og placering af det initierende problem undersøgt. Resultatet af disse undersøgelser dannede grundlaget for den endelige problemformulering.

Det første underspørgsmål i problemformuleringen var: *"Hvordan visualiseres lyset fra en given indendørslampe?"*. Dette underspørgsmål blev besvaret ved at repræsentere lampen, som en 3D-model, som sammen med brugerinput om den ønskede visualisering, indlæses af programmet og renderes vha. raytracing med brug af phong-modellen beskrevet i afsnit 6.1.

Det andet underspørgsmål var: *"Hvordan kan lampen og dens belysning visualiseres fra flere vinkler?"*. Dette underspørgsmål blev besvaret ved at anvende teorien om rotationsmatricer i afsnit 6.1, til at positionere det virtuelle kamera, som billedet renderes ud fra.

Det tredje underspørgsmål var: *"Hvordan visualiseres forskellige pærers lys?"*. Her blev der anvendt en algoritme, som konverterer en farvetemperatur for pæren til en RGB-værdi, som kunne benyttes i phong-modellen.

Problemformuleringens overordnede spørgsmål var *Hvordan kan vi lave et værktøj til e-butikker, som visualiserer belysningen fra indendørslamper for kunderne?"*. Ud fra ovenstående kan det nu konkluderes, at problemformuleringen er delvist løst, da der er udviklet et værktøj, der kan visualisere en lampe og dens belysning med forskellige synsvinkler og farvetemperaturer. Problemformuleringen er kun delvist løst, da der stadigvæk er mangler i forhold til implementeringen af værktøjet på en e-butiks hjemmeside. Programmet mangler følgende:

- Bedre overenstemmelse af farvetemperaturen for pæren i den rigtige lampe i forhold farvetemperaturen på det renderede billede.
- Lavere renderingstid for, at løsningen er praktisk anvendelig.
- Brugerflade til visning af billeder.

Vi kan nu samlet set konkludere, at vi gennem projektet har taget de første skridt mod at udvikle et værktøj, der visualiserer en lampe og dens belysning når kunder handler på en e-butiks hjemmeside.

9 Perspektivering

I dette afsnit forklares der hvad, der skal til for at arbejde videre på løsningen samt alternative anvendelsesmuligheder.

For at arbejde videre på løsningen vil det være relevant at udvikle følgende:

- En brugerflade til visning af billeder på e-butiks hjemmeside.
- Optimering af raytraceren, så renderingstiden formindskes.
- Viderudvikling af raytaceren med henblik på højere realisme.
- Højere overenstemmelse med reel farvetemperatur for pærer.
- At lave en backend for kommunikationen mellem serveren, hvor billederne renderes samt en e-butiks hjemmeside.

Selvom løsningsforslaget i denne rapport er målrettet mod kunder, som handler lamper på en e-butiks hjemmeside, er der andre anvendelsesmuligheder for løsningsforslaget. Det kunne tænkes at løsningsforslaget også kunne visualisere lamper og deres belysning for kunder, der handler i en detailbutik ved at have en computer eller tablet, som har implementeret løsningen.

I rapporten er det beskrevet, hvordan man kan konstruere et program, der renderer et billede på baggrund af en 3D-fil. Denne løsningsmodel kan nødvendigvis ikke kun bruges på lamper, men er også anvendelig til visualisering af andre produkter, som f.eks. elektronikprodukter, møbler mm.

10 Referencer

[1] Forklaring af kvalitativ metode, Den Store Danske. Set 25-11-2015.
http://www.denstoredanske.dk/Samfund,_jura_og_politik/Sociologi/Sociologisk_metodologi/kvalitative_metoder

[2] Beskrivelse af nummermetoden, Københavns Universitet. set 17-12-2015.
http://iva.ku.dk/refererkorrekt/tekshenvisninger/#Nummermetoden

[3] Human Factors in Lighting, third edition, Peter R. Boyce, 2014. Sider 532-536.
ISBN: 9781439874882.

[4] Konsekvenser ved dårlig belysning på arbejdsplads mm., ebscohost. Set 2-12-2015.
http://web.b.ebscohost.com/ehost/detail/detail?sid=2898a5ec-e3ec-4bf2-b3b7-f4eb754cd767%40sessionmgr115&vid=0&hid=101&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#anchor=toc&db=buh&AN=7531667

[5] Computer vision syndrom, GMJ. Set 2-12-2015.
http://gmj.sljol.info/article/10.4038/gmj.v11i1.1115/galley/1023/download/

[6] Definition af visualisering, Den Danske Ordbog. Set 27-10-2015.
http://ordnet.dk/ddo/ordbog?query=visualisere

[7] Definition af lys, Videnskab dk. set 27-10-2015.
http://videnskab.dk/sporg-videnskaben/hvad-er-lys

[8] Definition af lys, Britannica. Set 27-10-2015.
http://global.britannica.com/science/light

[9] Beskrivelse af farvetemperatur, Den Store Danske. Set 13-12-2015.
http://www.denstoredanske.dk/It,_teknik_og_naturvidenskab/Elektricitet/Belysning/farvetemperatur

[10] Om integral-led, integral-LED. Set 2-11-2015.
http://www.integral-led.com/about-integral-led

[11] Definition af varm og kold lys, integral-LED. Set 27-10-2015.
http://www.integral-led.com/education/warm-white-or-cool-white

[12] American Heritage, The free dictionary. set 27-10-2015
http://www.thefreedictionary.com/lamp

[13] Liste af solidworks produkter, solidworks. Set 13-12-2015.
http://www.solidworks.com/sw/3d-cad-design-software.htm

[14] The State of Retail 2015, Timetrade Sarah Wallace. Rapport udgivet i 2015. Side 22, figur 14. Copyright © 2015 by TimeTrade Systems, Inc. Set 3-11-2015.
http://www.timetrade.com/system/files/surveys/State_of_Retail_Report_Final_June15.pdf

[15] Fortrydelse og returret, forbrug.dk. Set 28-10-2015.
http://www.forbrug.dk/Raad-og-rettigheder/Forbrugerleksikon/Fortrydelsesret

[16] Ikeas returret, Ikea. Set 27-10-2015.
http://www.ikea.com/ms/da_DK/kundeservice/kundeservice_sporgsmaal_svar_kontakt_os.html?ICID=DKFOO_KONTAKT_230315

[17] Definition af e-handel, Den Danske Ordbog. Set 26-10-2015.
http://ordnet.dk/ddo/ordbog?query=ehandel

[18] Definition af e-butik, Den Danske Ordbog. Set 26-10-2015.
http://ordnet.dk/ddo/ordbog?query=ebutik

[19] Lov om forbrugeraftaler, Retsinformationen. Set 26-10-2015.
https://www.retsinformation.dk/forms/r0710.aspx?id=160666

[20] A Short Introduction to Computer Graphics, Frédo Durand - MIT Laboratory for Computer Science. CSAIL. Set 5-11-2015.
http://people.csail.mit.edu/fredo/Depiction/1_Introduction/reviewGraphics.pdf

[21] Visual content and software as a service, Cylindo. Set 9-11-2015.
http://www.cylindo.com/

[22] Real-Time Massive Model Rendering, first edition, Sung-Eui Yoon, 2008. Side 31.
ISBN: 9781598297928.

[23] Ray Tracing: Graphics for the Masses, Paul Rademacher of Department for Computer Science at the University of North Carolina at Chapel Hill. Department of Computer Science, University of North Carolina at Chapel Hill. Set 5-11-2015.
https://www.cs.unc.edu/~rademach/xroads-RT/RTarticle.html

[24] Fordele ved raytracing frem for rasterisering, set 16-12-2015,
http://www.tomshardware.com/reviews/ray-tracing-rasterization,2351-3.html

[25] Beskrivelse af bløde skygger i raytracing, set 16-12-2015,
https://graphics.ethz.ch/teaching/former/seminar/handouts/Lang_SoftShadowVolumes.pdf

[26] Beskrivelse af augmented reality, Augmented Reality. Soha Maad Chapter 1, section 4
ISBN 978-953-7619-69-5

[27] Artemides Augmented Reality App, Artemide. Set 14-12-2015.
http://www.artemide.com/blog/portfolio/the-new-version-of-artemide-augmented-real

[28] Elementary Linear Algebra 2015, Chapther 6.9: Rotations of R^3 and Computer Graphics, Olav Geil
ISBN: 978-1-78448-372-2

[29] How to Convert Temperature (K) to RGB: Algorithm and Sample Code, Tanner Helland. Set 04-12-2015.
http://www.tannerhelland.com/4435/convert-temperature-rgb-algorithm-code/

[30] Blackbody color datafile, Mitchell Charity. S et 04-12-2015.
http://www.vendian.org/mncharity/dir3/blackbody/UnstableURLs/bbr_color.htmle/

[31] Raw temperature vs RGB chart, Tanner Helland. Set 09-12-2015.
http://www.tannerhelland.com/4435/convert-temperature-rgb-algorithm-code/raw_temperature_vs_rgb_chart/

[32] 3D-bilede af kanin opdelt i trekanter, Rensselaer Computer Science. Set 17-12-2015.
http://www.cs.rpi.edu/~cutler/classes/advancedgraphics/S09/hw1_meshes.html

[33] Essential mathematics for games and interactive applications : a programmer's guide, second edition, James Van Verth & Lars M Bishop, 2008. Sider 212-236.
ISBN: 9780080878614.

[34] Illumination for computer generated pictures, Volume 18 Issue 6, Bui Tuong Phong & Robert Ashenhurst, 1975. Sider 311-317.
ISSN: 0001-0782.

[35] Raytracing, CS148 AS3, Stanford University. Set 01-12-2015.
http://graphics.stanford.edu/courses/cs148-10-summer/as3/instructions/as3.pdf

[36] Computational Geometry in C, Cambridge Tracts in Theoretical Computer Science 2nd edition.
ISBN: 0521649765.

[37] Kd-træer, Computer Science - University of Maryland (kd-trees CMSC 420). Set 17-12-2015
https://www.cs.umd.edu/class/spring2008/cmsc420/L19.kd-trees.pdf

11 Appendiks

Projektforslag

IT og læring i folkeskolen

Mange folkeskoler bruger i dag computeren i undervisningen. Hvordan kan man anvende computeren til at fremme indlæring i folkeskolen?

Problemstilling

Folkeskolen er grundlaget for den videre uddannelse af den nye generation. Da mange folkeskoler anvender computere i undervisning, er det oplagt at undersøge hvordan der kan udvikles et værktøj til at styrke indlæringen i undervisningen ved brug af computere.
Hvordan udvikles et værktøj til computeren, der kombinerer IT og læring i folkeskolen?

Mål

Målet er at udvikle et program, der kombinere IT og læring. Er der nogle problemer ved nuværende læringsværktøjer i folkeskolens undervisning, som kan løses ved hjælp af en datalogisk løsning?

Ekspempler på datalogiske problemstillinger

Datastrukturer, algoritmer, udvikling af grafisk brugerflade (GUI).

Ekspempler på kontekstuelle problemstillinger

Hvem er relevante at kontakte, når der skal udvikles et nyt læringsværktøj? Hvem skal læringsværktøjet udvikles til? Hvilke krav er der til et læringsværktøj?

Forslagsstiller

Gruppe B2-28 (SW1b2-28@student.aau.dk)

Bilag

A Mail fra lampedesigner Erik

Kære Mathias

Det er en meget komplex opgave i der er igang med, der er mange faktorer i spil når det handler om lys, både de fysiske, men ikke mindst de mentale. Jeg har i en del år arbejdet med lampedesign. og har derfor mest været optaget af armaturets/lampens skulpturelle udtryk, men da det jo er en lampe skal den selvfølgelig også opfylde det belysningsmæssige behov. Jeg har arbejdet med mange lyskilder, lige fra glødepæren til det nyeste LED.I alle mine lamper er valg af lyskilde og placering sket på grundlag af test via prototyper. de fleste af mine lamper er prototyper. En af de mest krævende lamper, har været Gedserlampen, der har et specielt designet armatur, der kan sammensættes til forskellige højder. Gedserlampen er en reflektorlampe, og lyskilden er LED. det krævede utallige målinger, og det foregik såmænd kl 12 om natten, en tommestok på jorden og et luxmeter. Jeg vil nok foretrække prototype test, da de jo er tæt på virkeligheden, men måske kunne jeres software være en god hjælp i den indledende fase af et projekt? Du er velkommen til at kontakte mig igen hvis du tror jeg kan bidrage med noget.

Held og Lykke med projektet.

Venlig hilsen

Erik Mortensen

B Mail fra lampedesigner David fra IKEA Sverige

Hi! Apart from hand sketching and physical prototypes, we use the 3D modeling application Solid Works in IKEA of Sweden. And for renderings we use either the built in renderer, or photo works, which is also part of solid works.

Regards

//David

On 9 October 2015 15:19:41 +08:00, Lasse Fribo Gadegaard ¡lgadeg15@student.aau.dk¿ wrote:

Hello

We are seven students from Aalborg university, that are writing a project on lamps and how they are design. And so we would like to ask you if you use any software to visualize your design, before they are made as a prototype or finished product. If you use any software feel free to tell us so we could research the software and get a better insight in the industry. I really hope you can help us. Thanks

Regards, 7 Students from Aalborg university

C Mail fra John Sørensen fra IKEA

From: XX
To: lasse-fribo@hotmail.com
Subject: Lys og lamper - IKEA
Date: Mon, 26 Oct 2015 15:25:23

Hej Lasse.

Tak for din mail

Jeg har lidt svært ved at forstå, ved at læse vedhæftede spørgeskema, hvad jeres problemformulering er.

Jeres spørgeskema har en ordlyd og formulering som kan opleves som negativt mod IKEAs produkter.

Jeg forstår, at jeres projekt ikke handler om IKEAs produkter, men er en generel undersøgelse om lamper og lys. Udfordringen er at vores kunder vil læse spørgsmålene og relatere det til IKEA, da I henvender jer til vore kunder i varehuset. Da spørgsmålene handler om utilfredshed, er vi bekymrede for denne kobling. På den baggrund, kan vi desværre ikke tilbyde jer at gennemføre undersøgelsen her i varehuset.

Med venlig hilsen
John Kristian Sørensen
Store Manager
IKEA Aalborg A/S

From: lasse fribo gadegaard
Sent: 30. oktober 2015 14:04
To: John Sørensen
Subject: RE: Lys og lamper - IKEA

Hej igen,

Jeg vedhæfter vores spørgeskema og vi vil meget gerne komme enten mandag, tirsdag efter kl 12 eller torsdag efter kl 14.

Venlig hilsen
Lasse Fribo Gadegaard og resten af gruppe B2-28
From: XX
To: lasse-fribo@hotmail.com
Subject: Lys og lamper - IKEA
Date: Mon, 26 Oct 2015 12:43:25

Hej Lasse.

Som aftalt sender jeg mine kontaktoplysninger.

I må gerne sende mig de spørgsmål I ønsker at stille vore kunder og forslag til dato I kommer til IKEA.

Ønsker dig en god dag.

Med venlig hilsen
John Kristian Sørensen
Store Manager
IKEA Aalborg A/S

D Mail fra belysningskonsulent

Fra: XX
Sendt: 6. november 2015 13:36
Til: Lasse Fribo Gadegaard
Emne: SV: Semesterprojekt om lamper - AAU
Hej

Det at se en lampe i 3D gør ikke at man ser lyset. Nogle af vores producenter laver allerede 3D modeller af deres lamper og endda sådan at man kan se lampen med lys i. Jeg har lige vedhæftet Artemides udgave af fremgangsmåden.

Men derfra til at se hvordan lyset er i et konkret rum hvor farver, højde mv. har indflydelse på lyset gør at det bliver en ekstrem kompleks størrelse der kræver komplicerede belysningsberegningsprogrammer som f.eks. DIALux. Lysberegning handler primært om lysmængde og ikke om lyseffekt.

Kunder har svært ved at forstå er hvilken lyseffekt lampen giver. Det er jo to-siddet. Dels vil de

se hvordan lampen ser ud i rummet og dels vil de se hvordan lyseffekten er. Første del klarer flere producenter. Hvis man så bagefter at man har sat lampen ind som 3D model burde man lave et lag over billedet hvor producenten så har taget et billede af et rum hvor man kan se skygger mv. Eks Tom Dixon. Det er det jo ikke lampe man køber men ofte lyseffekt og lampe handler jo om at forme lyset og lave skygger. At sætte en Tom Dixon lampe ind i et rum gør ikke at du kan se skyggerne.

Det er blevet en kompliceret proces at producere en lampe ift. EU lovgivning i dag så jeg har svært ved at se at producenterne vil koste endnu flere penge til produkter til privatmarkedet som måske kun køber en lampe til 3000 kr. som ofte kun interesserer sig for den laveste pris og ikke den bedste service og rådgivning. Så producenters incitament til ligge investeringer hos privatkunder er meget begrænset. Mange laver end ikke et fritskravet billede af deres lampe. Og igen Tom Dixon anvender en klar halogenlyskilde. Hvis du sætter en klar kultrådslyskilde hvor filamentet er længere giver forsat skygger men på en blødere måde da lyset er delt ud på en større overflade. Hvis du sætter en mat lyskilde i forsvinder skyggerne næsten helt. Pludselig er løsningen bare komplekst og når en producent så har 4000 varenumre. Samme lyskilder laves så i 3 farvetemperaturer. Med vores adgang til varesortiment giver det 2 mio. billeder dokumenter som skal indhentes. Så er vi der hvor det begynder ikke at hænge sammen tidsmæssigt når nethandlen handler om at være først ift. googlesøgninger mv. Hvem har lyst til at give bedste rådgivning ift. lyseffekter hvis man ender på side 20 når folk søger på google?

Løsningsforslag modtages derfor med kyshånd da kompleksitet er desværre nem at se.

I må selvfølgelig også gerne vores udtagelser.

Med venlig hilsen / best regards

XX
Belysninskonsulent

Fra: Lasse Fribo Gadegaard [mailto:lgadeg15@student.aau.dk]
Sendt: 6. november 2015 11:19
Til: XX
Emne: SV: Semesterprojekt om lamper - AAU

Hej

Vi er meget glade for at I vil bidrage til projektet. Indtil videre har vi analyseret problemet: Forbrugeren kan ikke visualisere, hvordan lyset udbreder sig fra en lampe uden at købe og installere lampen.

I problemanalysen har vi undersøgt interessenter, begreber, placering og teknologier til problemet.

Ud fra dette har vi valgt at fokusere på e-butikker, der sælger indendørs lamper til brug i erhverv

eller private hjem.

Vi har netop udarbejdet den endelige problemformulering, hvor udkastet lyder som følgende: Hvordan kan man lave et værktøj til e-butikker som vha. raytracing*, visualiserer belysningen fra indendørs lamper for kunderne?

* En teknik til at simulere lys og lave et billede af en 3D-model.

Vi skal nu til at udvikle en løsning til problemet, og hertil har vi lavet en simpel skitse (Se vedlagt billede) af den ide vi har på nuværende tidspunkt.

Som vist på skitsen, er ideen at lave et produkt som gør det muligt for kunderne at se nogle lamper og deres belysning i et interaktivt 3D-billede på e-butikken.

Vi tænker at forbrugeren skal kunne gøre følgende:
- Vende og dreje billede, så de kan se lampen og belysningen fra flere vinkler.
- Se lampen med forskellige pærer (evt. angive farvetemperatur i Kelvin)
- Skifte den kontekst som lampen visualiseres i (f.eks. forskellige rum/møbler)

Pga. tidsbegrænsning forventer vi ikke at lave hele løsning, som produkt, men blot implementere de mest studierelevante dele.

Dog skal vi stadigvæk præsenterer en færdig løsning i rapporten.

Det vi nu ønsker jeres respons på, er følgende - Jeres tanker omkring ideen, som løsning på problemet.
- Forslag og ønsker til forbedringer af ideen.
- Jeres accept til, at vi i rapporten må inddrage jeres udtagelser anonymt.

Med venlig hilsen,
Lasse Gadegaard

På vegne af
Gruppe B2-28
Software, AAU

Fra: XX
Sendt: 5. november 2015 15:43
Til: Lasse Fribo Gadegaard
Emne: SV: Semesterprojekt om lamper - AAU
Hej Lasse

Det lyder til at være et meget spændende projekt.

Som primær detailforretning med projektafdeling lever vi af konsulentarbejde ved at give rådgivning omkring hvordan lys forandre sig ift. til lofthøjde, farver, armatur, lyskilde foruden at der er en subjektiv mening om hvad godt lys er.

Der er overraskende mange der gerne vil se lyset inden de køber lamper. Man kan dog undre sig ovre at samme kunde køber både køleskabe, vaskemaskiner mv. uden at stille krav til at prøve tingene før de køber varerne selv om disse produkter koster lige så meget som de lamper vi sælger. Kunder har åbenbart et specielt forhold til lys.

Har i allerede valgt teori, metode og empiri?

Jeg tror godt vi kan hjælpe jer. Eneste krav er at data herfra bliver anonymiseret og vi får et eksemplar af opgaven når den er skrevet.

God dag

Med venlig hilsen / best regards
XX
Belysningskonsulent

Fra: Lasse Fribo Gadegaard [mailto:lgadeg15@student.aau.dk]
Sendt: 5. november 2015 15:06
Til: XX
Emne: Semesterprojekt om lamper - AAU

Hej XX

Vi er en gruppe på Aalborg Universitet, som er i gang med et projekt om visualisering af lamper.

Vi arbejder med følgende problemstilling: "Forbrugeren kan ikke visualisere, hvordan lyset udbreder sig fra en lampe uden at købe og installere lampen."

Vi har fokus på e-handel, og ønsker at tilbyde e-butikken et værktøj som gør det nemmere for kunderne at visualisere, hvordan lyset breder sig ud fra en lampe (f.eks. hvilke skygger, mønstre og farver som lampen udsender).

Derfor søger vi nu e-butikker, som ønsker at bidrage med viden og informationer omkring e-handel med lamper.

Hvis I er interesserede i at medvirke i projektet, så skriv venligst tilbage på mail: lgadeg15@student.aau.dk

Med venlig hilsen,
Lasse Gadegaard
På vegne af
Gruppe B2-28
AAU, Software

E Spørgeskema

Spørgeskema omhandlende lamper

Mand: __ Kvinde: __

Alder: _____

Har du oplevet, at du har været utilfreds med et lampekøb?

- ☐ Ja
- ☐ Nej

Hvis ja, hvilke af følgende punkter var årsag til din utilfredshed? (Sæt gerne flere kryds/flueben).

- ☐ Lampen havde et irriterende lys
- ☐ Lampen passede ikke ind i rummet
- ☐ Andet:_____

Afgør om du er enig i følgende udsagn:

Jeg har tidligere oplevet at lyset fra en lampe var væsentligt anderledes end jeg forestillede mig på købstidspunktet.

□ Meget enig □ Lidt enig □ Lidt uenig □ Meget uenig □ Ved ikke

Tak for svaret

Hilsen gruppe B2-28

AAU - Software