

Bachelorproef:
Optimaliseren van de lichtcollectie bij
cathodoluminescentie

Emile Segers
emile.segers@ugent.be

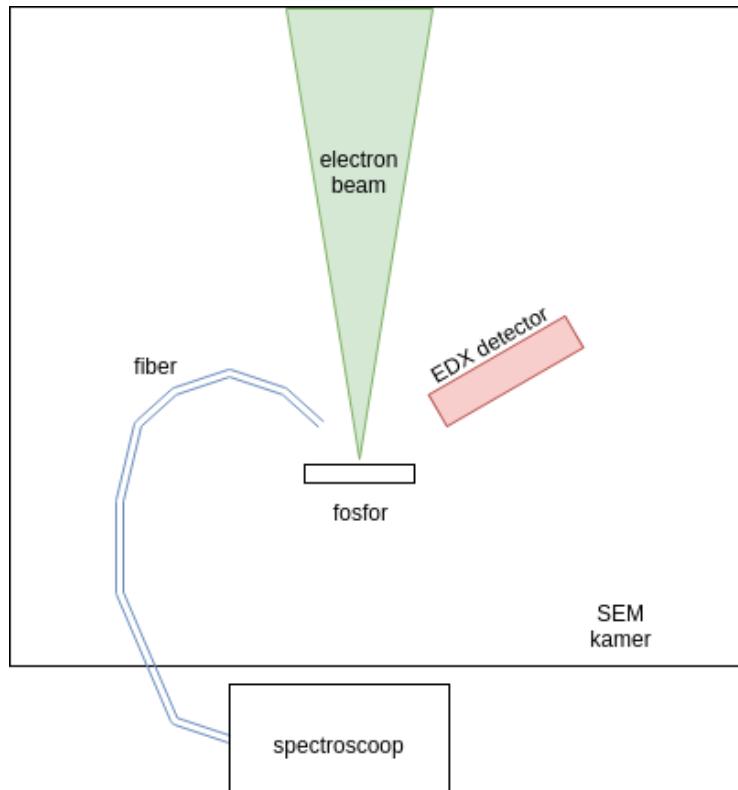
19 juni 2018

Inhoudsopgave

1 Inleiding	1
2 Doel en methode	2
2.1 Doel	2
2.2 Methode	2
3 Experimenten	2
3.1 Basisopstelling	3
3.2 Inzichtsexperimenten	4
3.3 Initiële omstandigheden	5
3.4 Alternatieven	6
3.4.1 Collimerende lens op vezel	6
3.4.2 Spiegel	7
3.5 Vergelijken resultaten	9
4 Simulaties	9
4.1 Initiële toestand	11
4.2 Alternatieven	12
4.2.1 Lens en spiegel	12
4.2.2 Lens	13
5 Maken van houder	13
5.1 Concept	13
5.2 3D model en uiteindelijke houder	15
5.3 Vergelijking van oude en nieuwe methode	16
A Appendix	19

1 Inleiding

Fosforen worden gezien hun eigenschap van luminiscentie in vele toepassingen aangewend, zoals bijvoorbeeld bij LED's en glow in the dark produkten. Om een maximale efficiëntie te verkrijgen is meer inzicht in het werkingsmechanisme en kwaliteit van fosforen aangewezen, dit door onderzoek op microscopische schaal. Dergelijk onderzoek doet men aan de hand van een SEM-CL-EDX. Hierbij vuurt men een bundel elektronen af op een fosfor. Het moederkristal zal energie absorberen en deze doorgeven aan de dopanten die dan licht in het zichtbaar spectrum uitstralen. Dit noemt men cathodoluminescentie. De eigenschappen van een fosfor hangen af van de microscopische structuur van het fosfor. Dit is bijvoorbeeld het samenclusteren van dopanten. Met behulp van de SEM-CL-EDX kan men aan de hand van een spectroscoop en een EDX (= energy dispersive X-ray spectroscopy) de eigenschappen van het fosfor onderzoeken op submicrometer niveau. Zo kan men onder andere thermische quenching [2] onderzoeken om de werking van fosforen zo efficiënt mogelijk te maken.



2 Doel en methode

2.1 Doel

Binnen dit bachelorproject is het de bedoeling om de collectie van cathodoluminescentie m.b.v een optische vezel binnenin een SEM te optimaliseren.

2.2 Methode

Het oorspronkelijke idee was om eenhouder te maken die aan de hand van vacuum doorvoer of vacuumstappenmotoren kan aangestuurd worden, dit om de vezel makkelijk te kunnen manoeuvreren in de kamer. Wegens complicaties bij de constructie van dit concept en het prijskaartje dat we hierbij mogen verwachten, zijn we op zoek gegaan naar een alternatieve manier van aanpak. Het concept dat we in deze proef verder uitwerken bestaat uit het 3D printen van eenhouder die men op de stage van de SEM plaatst om aan de hand van spiegels en lenzen zo veel mogelijk licht op te vangen in de vezel.

Om de collectie zo efficiënt mogelijk te maken, voeren we eerst experimenten uit om de optimale karakteristieken en positie van de vezel, spiegels en lenzen te bepalen. We doen dit a.d.h.v. experimenten met een optische bank en o.b.v. simulaties met raytracing software.

3 Experimenten

Naar aanleiding van onverwachte resultaten in de opgemeten spectra zijn de opstelling en omstandigheden van de omgeving in de loop van de tijd aangepast. Deze wijzigingen gaan van het toevoegen van een filter tot het werken in een verduisterde kamer. De redenen achter deze aanpassingen lichten we in de volgende paragrafen toe.

De experimenten splitsen we op in 3 verschillende groepen: inzichtsexperimenten, de initiële omstandigheden en de mogelijke alternatieven. Elk experiment zal gebruik maken van dezelfde basisopstelling, die hieronder wordt beschreven. Aan deze basisopstelling kunnen, indien gewenst of nodig, elementen toegevoegd worden.

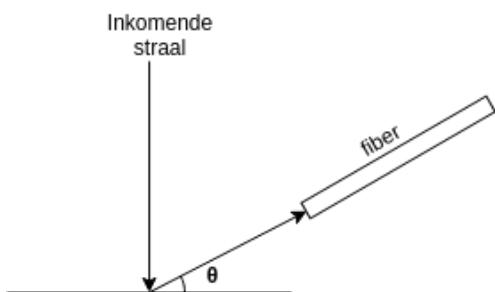
3.1 Basisopstelling

De opstelling die we maken op de optische bank is zo ingericht dat deze zo dicht mogelijk bij de opstelling komt van de SEM. Het fosfor dat we gebruiken in het experiment is $\text{ZnGa}_2\text{S}_4 : \text{Eu}^{2+}$. Gezien we niet kunnen werken met een bundel van electronen om het fosfor te exciteren, vervangen we deze door een laser die blauw licht (450 nm) uitstraalt. Om de intensiteit van de laser te controleren maken we gebruik van een golffunctiegenerator die een blokfunctie genereert met een frekwentie van 60 kHz, een amplitude rond de 5 V, een offset van ongeveer 1,85 V en een duty cycle van 20 %. Indien de afstelling van dit apparaat niet goed gebeurd is, is het mogelijk dat de laser meer en meer zal opladen over de tijd en dus meer licht zal uitstralen. Om dit fenomeen te vermijden, moeten we voor de uitvoering van elk experiment de amplitude en de offset opnieuw zorgvuldig afstellen. Om de experimenten te kunnen vergelijken met elkaar moeten we er tevens voor zorgen dat in eenzelfde positie de intensiteit van het invallende licht identiek is voor elk experiment. Ten dien einde zullen we voor en na elke dag van experimenten een spectrum opnemen waarbij de vezel op een afstand van 3 cm en onder een hoek van 40° van het fosfor is geplaatst. Door vergelijking van de opeenvolgende opnames kunnen we dan bevestigen dat ze met elkaar overeenstemmen.

Om het licht op te vangen maken we gebruik van een optische vezel met een diameter van 1 mm die verbonden is aan een spectrometer (Ocean Optics STS-VIS Miniature spectrometer). Ten laatste plaatsen we nog een filter in de opstelling om zoveel mogelijk van het blauwe licht te elimineren in het spectrum dat we onderzoeken.



(a) afbeelding



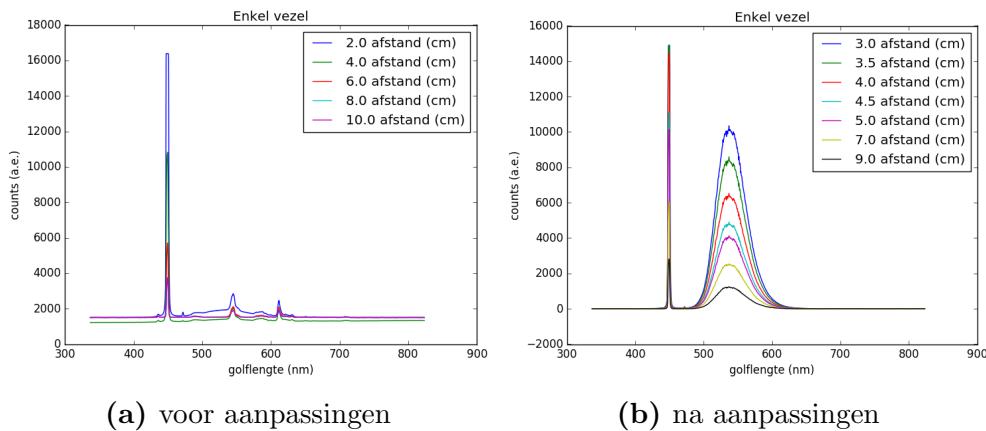
(b) diagram

Figuur 1: basisopstelling

3.2 Inzichtsexperimenten

Dit zijn alle experimenten die data leveren die niet direct informatie geven over hoe de collectie van licht zich precies gedraagt maar eerder over wat men kan aanpassen aan de experimenten of mogelijkse verbanden die men kan leggen tussen de verschillende experimenten.

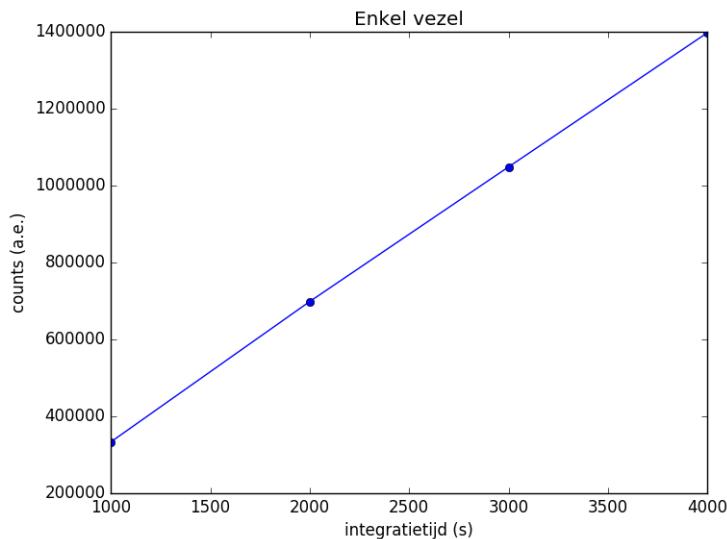
Figuur 2a visualiseert de resultaten van het allereerste experiment dat we uitvoerden.



Figuur 2: eerste experiment

Als we kijken naar het spectrum kunnen we rond 550 nm en 620 nm 2 pieken waarnemen die we niet verwachten in het spectrum van het fosfor. Deze pieken zijn afkomstig van de verlichting van de ruimte waar we de experimenten uitvoeren. We kunnen deze pieken elimineren door in het vervolg experimenten in het donker uit te voeren. Niettegenstaande het toevoegen van een filter aan de opstelling, is er nog steeds een dominerende piek in het spectrum rond 450 nm. Initieel was het de bedoeling het spectrum niet te laten satureren rond deze piek. Als we deze piek niet satureren zijn er bijna geen counts waar te nemen in het groene gebied van het licht. Daarom zullen we op het vlak van saturatie niet kijken naar de piek in het blauw licht maar enkel naar het spectrum van het groen licht. Indien we dit experiment herhalen met deze aanpassingen krijgt men de grafiek in figuur 2b.

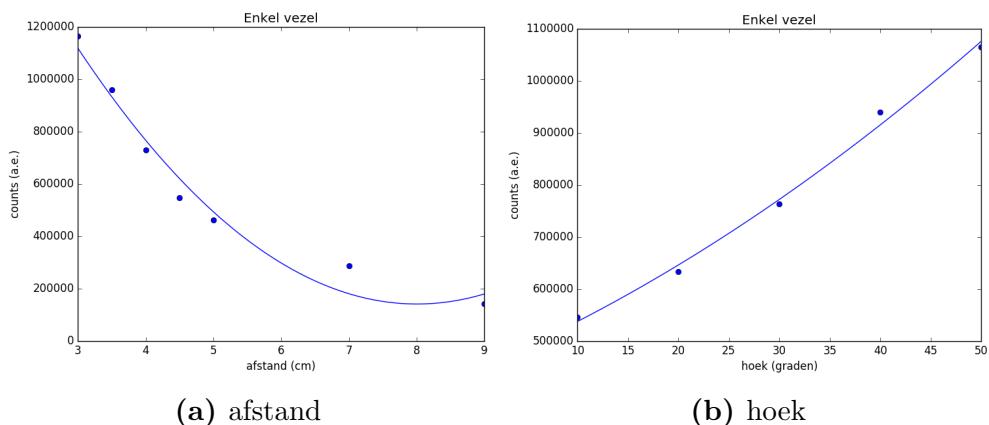
Een tweede inzichtsexperiment behelst de relatie van de integratietijd van de spectroscoop en de hoeveelheid geregistreerde counts. We doen dit door de vezel in de basisopstelling onder een hoek van 40° en op 3 cm van het fosfor te plaatsen en dit te herhalen voor een aantal verschillende integratietijden. Als we voor elk spectrum de counts in het gebied van 480-630 nm optellen krijgen we de grafiek in figuur 3. Hier kan je direct uit opmaken dat de integratietijd wel degelijk evenredig is met het aantal counts dat geregistreerd wordt.



Figuur 3: relatie integratietijd en aantal counts

3.3 Initiële omstandigheden

Deze metingen zijn gedaan met de basisopstelling waarbij we de afstand en de hoek van de vezel ten opzichte van het fosfor aanpassen. Qua integratietijd willen we de situatie vermijden waarin saturatie van de spectroscoop optreedt en we nemen de integratietijd dan ook zo groot mogelijk op dat we geen belangrijke informatie verliezen bij het onderzoeken van de opstelling met lagere lichtintensiteiten.



Figuur 4: Initiële toestand: enkel vezel

Voor dit experiment maken we gebruik van een integratietijd van 3000 ms. Integreren

we weer over het groene gebied van het licht, dan zien we de trend die we verwachten. Het aantal counts zal kwadratisch afnemen naarmate de vezel verder wordt geplaatst van het fosfor. Wat de hoekafhankelijkheid betreft kan men duidelijk zien dat er bij een stijgende hoek een stijgende trend is in lichtintensiteit. Dit komt overeen met de lambertiaanse straler.

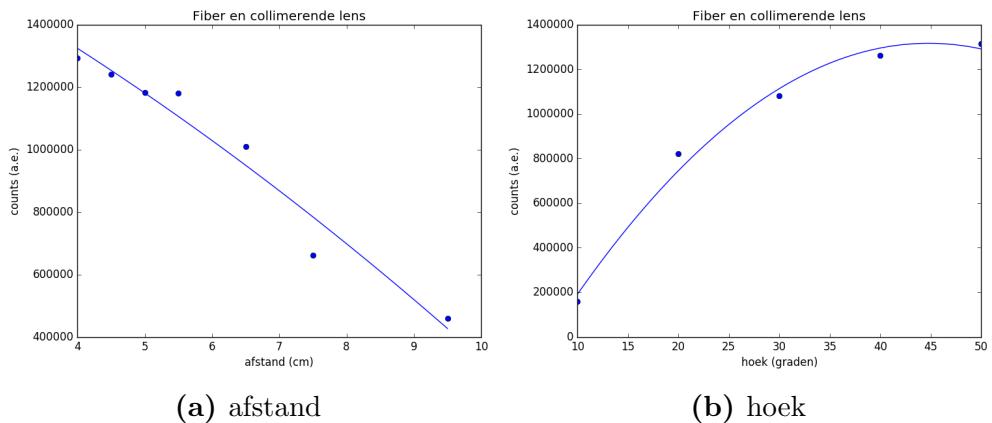
$$I = I_{max} \sin(\theta) \quad (1)$$

Hierbij is I de intensiteit van het geëmitteerde licht, I_{max} de intensiteit van het inkomende licht en θ is deze van figuur 1b.

3.4 Alternatieven

3.4.1 Collimerende lens op vezel

Het meest eenvoudige object om toe te voegen aan de opstelling is een collimerende lens die men op de optische vezel kan schroeven. Dit zal 2 voordelen met zich meebrengen. Enerzijds heeft men nu de mogelijkheid de aperture van het systeem aan te passen. Anderzijds is het oppervlak waar het licht kan op invallen veel groter dan het oppervlak van de vezel alleen. We kunnen dit direct zien aan de experimenten die we uitvoeren. We krijgen



Figuur 5: vezel met collimerende lens

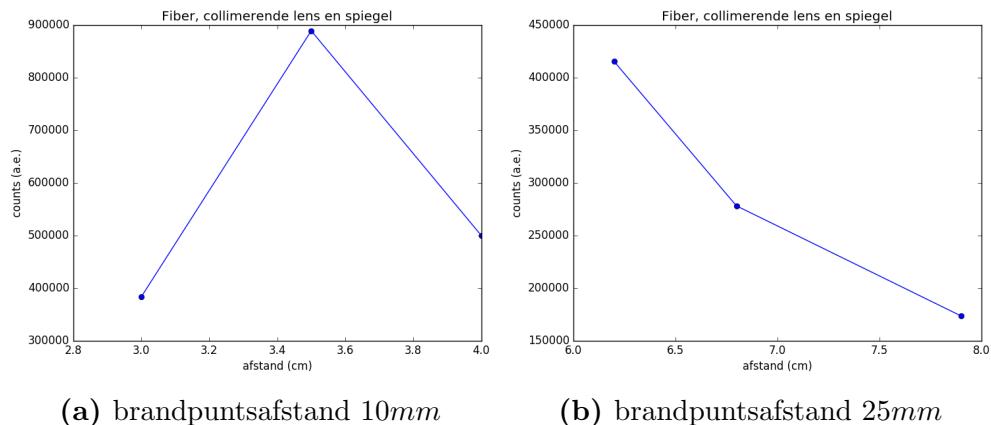
dezelfde trends zoals bij de initiële omstandigheden maar we moeten hier een integratietijd van 1500 ms gebruiken om er voor te zorgen dat de spectroscoop niet satureert. De reden waarom deze soms wat afwijkt van de verwachte trend is omdat deze heel gevoelig is aan kleine verschillen in de opstelling. Dit komt omdat we de aperture voor één experiment constant willen houden waardoor de spot van lichtcollectie op bepaalde afstanden van het

fosfor heel klein wordt. Wat men hier wel mooi kan zien is de hoekafhankelijkheid van de lambertiaanse straler.

3.4.2 Spiegel

Voor een correcte interpretatie van de resultaten, is het belangrijk om te weten dat een spiegel en een lens met dezelfde brandpuntsafstand voor dezelfde resultaten zorgen. We werken hier met spiegels om 2 redenen: enerzijds omdat een spiegel reflecteert en niet gewoon het licht afbuigt, zou men de uiteindelijk opstelling kunnen halveren qua grootte, anderzijds zijn lenzen met een voldoende kleine brandpuntsafstand moeilijk te vinden.

In dit experiment plaatsen we links van het fosfor een spiegel en plaatsen de vezel zo dat deze wijst naar de spiegel. De hoek is zo gekozen dat de vezel zo veel mogelijk licht ontvangt van de spiegel. We onderzoeken 2 spiegels met 2 verschillende brandpuntsafstanden (a. $f = 10 \text{ mm}$, b. $f = 25 \text{ mm}$). Voor deze experimenten gebruiken we een integratietijd van 50 ms om ervoor te zorgen dat de spectroscoop niet satureert. Voor experiment a moeten we

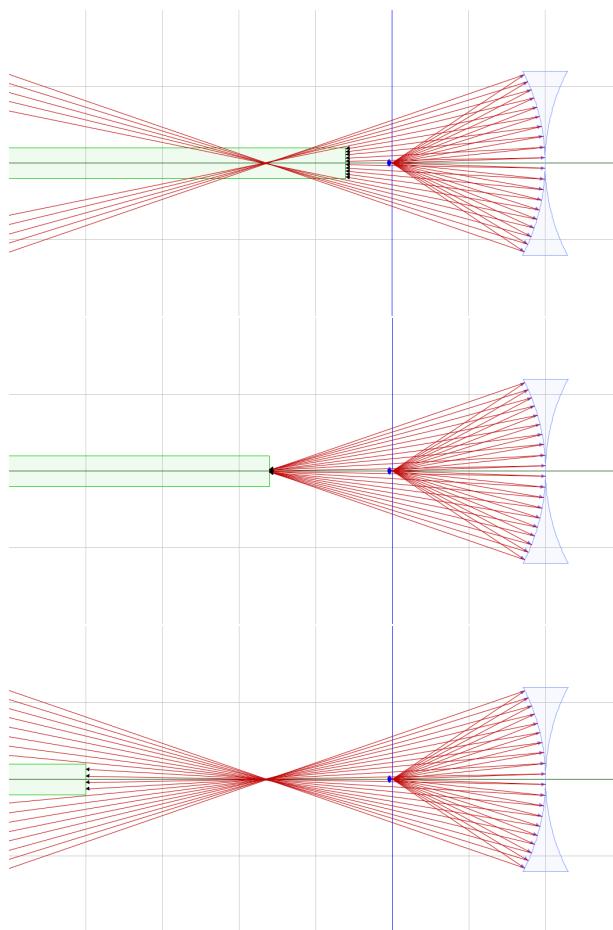


Figuur 6: vezel met spiegel

de spiegel zo dicht mogelijk plaatsen bij het specimen om ervoor te zorgen dat het beeld van het fosfor gemaakt door de spiegel zich voorbij het sample bevindt. Zoniet, krijgen we problemen met de vezel die de stralen tegenhoudt van de excitatiebron. Met deze limitatie hebben we de spiegel op 1,4 cm van het specimen geplaatst. Hieruit kan men opmaken dat voor onze toepassing de geometrische optica van toepassing zijn:

$$\frac{1}{f} = \frac{1}{b} + \frac{1}{v} \quad (2)$$

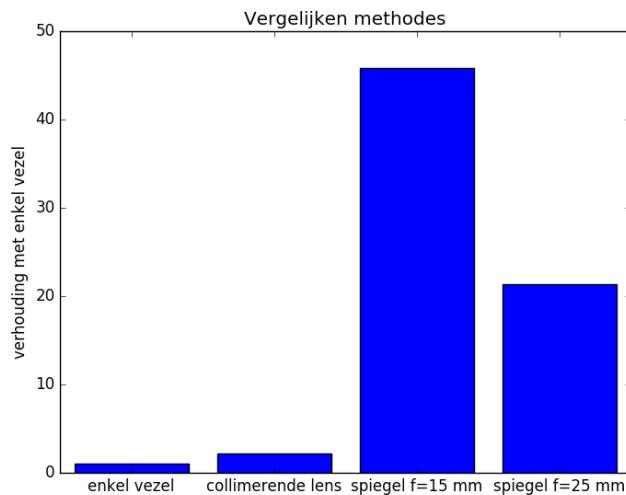
Hierbij is v de voorwerpsafstand en b de beeldafstand. Voor het experiment met de tweede spiegel is de spiegel geplaatst op een afstand van 4,2 cm van het fosfor. Het is duidelijk dat de grafiek naar een maximum aan het gaan is als we dichter bij het sample gaan onderzoeken. Praktisch gezien is het echter onmogelijk om de vezel dichter te positioneren op de optische bank, daar in de opstelling ook nog een ruiter met filter moet geplaatst worden tussen de vezel en de spiegel. De reden waarom we een piek krijgen in het beeld van zo een systeem is aan te tonen aan de hand van de figuur 7. Je kan zien dat als de vezel zich niet in het beeldpunt bevindt een deel van de lichtstralen niet opgevangen zullen worden.



Figuur 7: Illustratie van verminderde lichtcollectie bij mismatch beeldafstand en vezelpositie

3.5 Vergelijken resultaten

Als we gebruik maken van het feit dat het aantal geregistreerde counts evenredig is met de integratietijd van een meting kunnen we de hoogst verkregen waarde van elk experiment vergelijken met elkaar om een idee te krijgen hoe efficiënt de nieuwe methodes zijn tegenover de oude. De bijhorende figuur toont dat de opstellingen met spiegels veel meer licht



Figuur 8: verhouding alternatieve opstellingen en initiële opstelling

opvangen, waar we bij het maken van de houder gebruik van gaan maken.

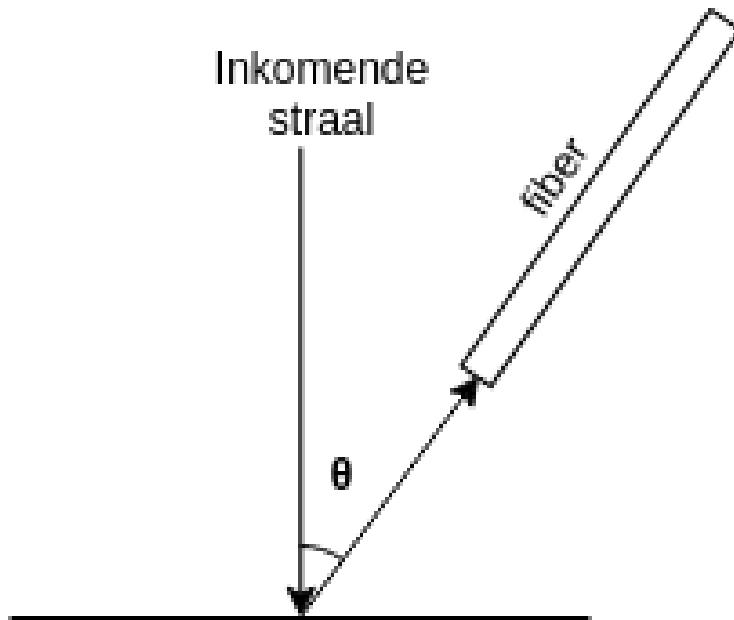
4 Simulaties

Voor de simulaties maken we gebruik van een raytracer programma. Dit is een programma dat in real time lichtpaden kan berekenen. Dit omvat onder andere reflectie, absorptie en afbuiging van lichtstralen. De raytracer software die we gebruiken is de optical raytracer van P. Lutus [1]. De reden waarom we deze gebruiken is omdat deze raytracer een goede combinatie geeft tussen intuïtief gebruik en voldoende functies om alles wat we willen te onderzoeken. Nog een groot voordeel bij dit programma is dat er een mogelijkheid is om het aan te sturen via commando's in de terminal. We kunnen een config-file opstellen die de objecten zoals lenzen of spiegels op de correcte plaats zal zetten voor de simulatie waarna we bij het uitvoeren van de raytracer een tabel terug krijgen met alle informatie

over hoe elke lichtstraal interageert met de objecten die in hun weg voorbij komen.

Bij het initieel opstellen van het virtuele experiment traden er een aantal problemen op.

- Lambertiaanse straler: In het gebruikte pakket bestaat er geen bron die zal stralen als een lambertiaanse straler. Daarom hebben we een lineaire lichtbron laten invallen op een elliptische spiegel. Aan de hand van deze oplossing kan men toch de hoekafhankelijkheid van de lichtintensiteit testen. Een belangrijke opmerking hierbij is dat de hoekafhankelijkheid omgekeerd werkt als in de werkelijkheid. Dit probleem lossen we op door de hoek te veranderen naar deze in de onderstaand figuur.



Figuur 9: diagram opstelling simulatie

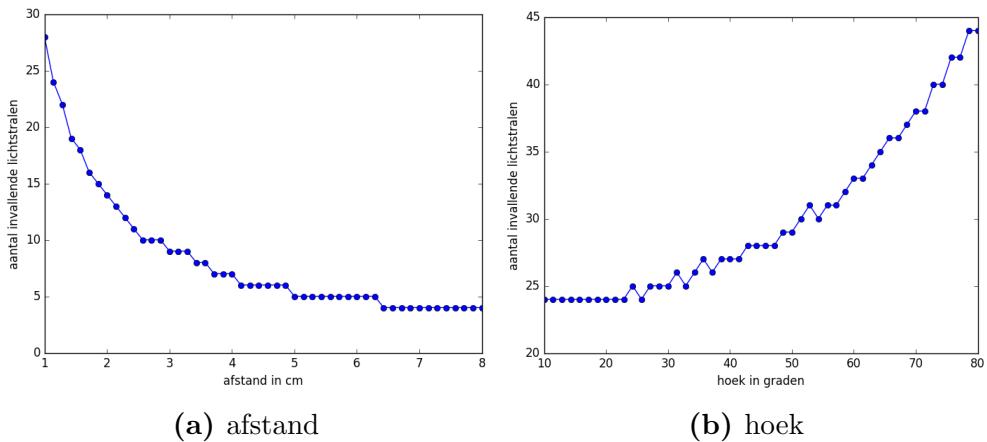
- Positie limitaties: De raytracer zal geen kommagetallen toelaten in het programma waardoor men gelimiteerd wordt in hoe precies men de metingen kan doen. Dit probleem kan deels omzeild worden door te werken op schaal waardoor men toch nog een hogere graad van accuraatheid kan behalen.
- Continuïteit van lichtstralen: In de werkelijkheid weet men natuurlijk dat het licht elke ruimtehoek van het fosfor zal bestrijken. In een simulatie is dat echter niet mogelijk. Hier heeft men een beperkt aantal lichtstralen die men in één keer kan onderzoeken. Voor dit programma in het bijzonder is dit 1000 lichtstralen.

Met behulp van een Python-applicatie is het makkelijk om de resulterende tabellen te interpreteren en te visualiseren. Ook kunnen de simulaties geautomatiseerd worden. De Python-code kan je vinden in appendix A. Deze code is geoptimaliseerd voor gebruik in linux.

Doel van de simulaties is om de trends die men heeft gevonden in de experimenten te verifiëren en om in de werkelijkheid niet te onderzoeken extremen toch eens te bekijken.

4.1 Initiële toestand

Het is natuurlijk interessant om te zien of de meest eenvoudige opstelling dezelfde trend zal weergeven voor de afstandsafhankelijkheid en de hoeksafhankelijkheid van de lichtintensiteit. Voor wat de afstandsafhankelijkheid van de lichtintensiteit betreft, zien we geen



Figuur 10: simulatie van basisopstelling

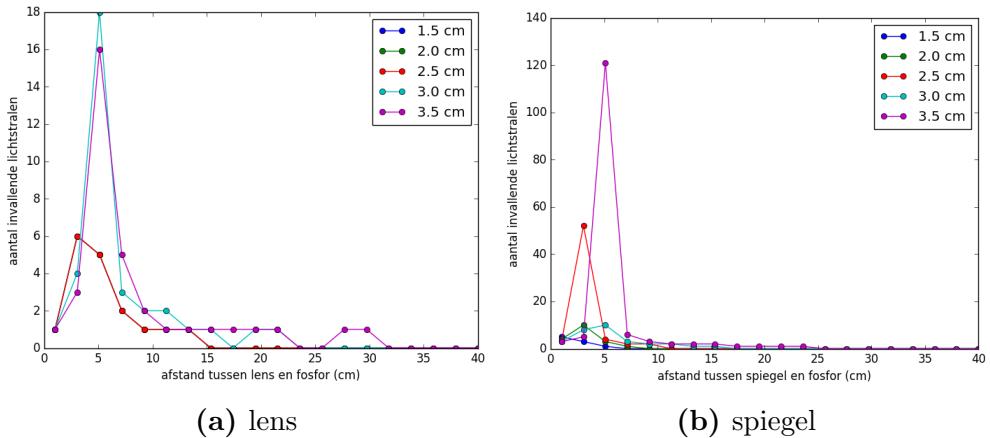
verschil. Daarentegen ziet de grafiek voor de hoekafhankelijkheid er helemaal anders uit dan deze van bij de experimenten. Dit komt door de limitaties op het nabootsen van een lambartiaanse straler in het raytracing programma. Desondanks valt de verkregen data nog mee. Als men even logisch kijkt, kan men zien dat dit gewoon een spiegeling is over de eerste bissectrice. Dit is voor ons uiteindelijk niet zo een groot probleem, een strikt stijgende functie die we spiegelen over de eerste bissectrice zal ook een strikt stijgende functie zijn. De soms onverwachte dip in het aantal invallende lichtstralen is een gevolg van het feit dat in het programma licht wordt voorgesteld door een eindig aantal lichtstralen. Afhankelijk van de verplaatsing van de vezel kan het aantal opgevangen lichtstralen

immers licht gaan variëren: bij de verplaatsing kan langs de ene kant een lichtstraal alreeds weggevallen zijn terwijl er aan de andere kant er nog geen nieuwe is bijgekomen.

4.2 Alternatieven

4.2.1 Lens en spiegel

Een eerste interessepunt is de aannname dat lenzen en spiegels met gelijke brandpuntsafstanden zich gelijk zullen gedragen. Dit kunnen we verifiëren door in een experiment zowel de lens als de spiegel te onderzoeken. Deze simulatie gaat als volgt. Het object dat we onderzoeken, plaatsen we onder een vaste hoek en een variërende afstand van het fosfor. Wat de plaatsing van de vezel betreft, zetten we deze steeds in het midden van de geïnterageerde lichtstralen en op een vaste afstand van het object, i.c. op 10 cm. We herhalen deze simulaties over een aantal verschillende brandpuntsafstanden om de gelijkenis duidelijk te maken. Het fenomeen waarbij de pieken in counts gelijk zijn bij de lens voor de

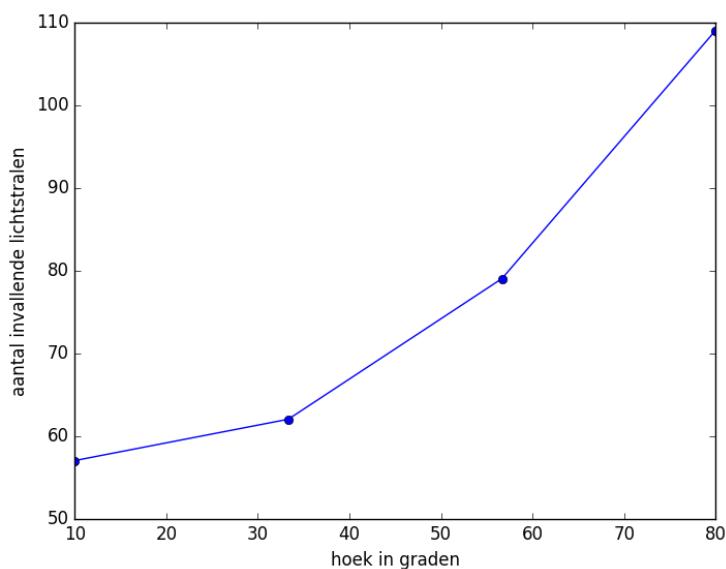


Figuur 11: vergelijken van eigenschappen spiegel en lens

kleinere brandpuntsafstanden komt doordat we in een limitatie van het programma terecht komen. De radius van de spiegel zelf zal kleiner beginnen worden dan de radius van de curve van de spiegel. Dit is natuurlijk niet mogelijk en het programma reduceert de radius van de curve automatisch naar de kleinst mogelijke waarde. Bekijken we de objecten met de grotere brandpuntsafstanden dan zien we een mooie gelijkenis op de plaats waar deze een piek zal vertonen in het aantal inkomende lichtstralen.

4.2.2 Lens

Naast het bekijken van wat de lens en spiegel doen in functie van de afstand is het ook nog handig eens te zien wat deze voorwerpen doen in functie van de hoek. Omdat we juist hebben bewezen dat de spiegel en de lens zich gelijkaardig gedragen, hoeven we maar één van beide te onderzoeken. Het opstellen van een simulatie met een lens is een stuk makkelijker dan deze van een spiegel. Hier plaatsen we de vezel ook altijd in het midden van de gecollimeerde lichtstralen om zoveel mogelijk licht op te vangen. De hoek die men



Figuur 12: simulatie hoekafhankelijkheid lens en spiegel

hier onderzoekt is dezelfde als die in het diagram van de simulatie waarbij men tussen het fosfor en de fiber een lens heeft geplaatst. Zoals je kan zien vertoont deze simulatie juist hetzelfde gedrag als dat van de basisopstelling.

5 Maken van houder

5.1 Concept

Het concept dat we willen uitwerken is dat van een 3D-geprinte houder die we op de stage van de SEM kunnen plaatsen. De vezel en de elementen die we extra willen toevoegen kunnen we aan de hand van blokken toevoegen aan de grondplaat.

Om te beslissen welke opstelling we best gebruiken is het aangewezen om eerst de voor- en

nadelen van elk object te bekijken.

objecten	verhouding counts tov enkel vezel	voordelen	nadelen
enkel vezel	1	<ul style="list-style-type: none"> • makkelijk op te stellen voor maximale lichtopvang • stage kan hoog geplaatst worden (geen obstakels) 	<ul style="list-style-type: none"> • weinig lichtinval
collimerende lens	± 2	<ul style="list-style-type: none"> • makkelijk op te stellen voor maximale lichtopvang • makkelijk toe te voegen • flexibele aperture 	<ul style="list-style-type: none"> • iets groter → stage lager plaatsen
spiegel /lens	± 20 (brandpuntsafstand 25 mm)	<ul style="list-style-type: none"> • hoge lichtcollectie 	<ul style="list-style-type: none"> • complexer te installeren • stage veel lager door obstructie van de SEM kolom

Op het eerste zicht ziet het er naar uit dat het gebruik van een lens of een spiegel te veel nadelen heeft om te integreren in de oplossing. De te verwachten substantiële verbetering in lichtcollectie zet ons er toch toe aan hiervoor een oplossing te zoeken.

Zowel de spiegel als de lens hebben elk hun eigen voor- en nadeel. De lens heeft een heel geconcentreerd beeld waardoor we efficiënter licht kunnen opvangen in de vezel maar deze opstelling zal voor hetzelfde effect van een spiegel veel meer plaats innemen omdat alles sequentieel achter elkaar moet staan. Bij de spiegel nemen we juist het tegenovergestelde waar. Omdat we hier reflecteren kunnen we de spiegel aan de ene kant van het fosfor plaatsen en de vezel aan de andere kant waardoor de omvang van de opstelling potentieel gehalveerd wordt. Het nadeel van de spiegel is dat om ervoor te zorgen dat het weerkaatste licht niet terug op dezelfde plaats terecht komt, we de spiegel een beetje schuiner moeten zetten. Op zijn beurt zorgt dit ervoor dat het beeld uitgesmeerd wordt over een groter oppervlak. We kunnen dit aanpakken door deze hoek minimaal te houden, dit in combinatie met een collimerende lens. Dit wordt duidelijk als we de oude en nieuwe methode van lichtcollectie met elkaar vergelijken.

5.2 3D model en uiteindelijke houder

Bij het maken van een 3D model is het van essentieel belang dat er in het ontwerp geen niet-ondersteunde onderdelen mogen aanwezig zijn. Dit zorgt voor een aantal problemen bij de openingen voor de spiegel en de optische vezel. De manier waarop we dit kunnen omzeilen is door het model op te splitsen in meerdere onderdelen. De basis vormt het hoofdonderdeel: dit is een schijf waar in het centrum een opening is gemaakt waar de houder van het sample juist in past. Voor het bevestigen van de optische elementen gebruiken we blokjes met een voet die we dan kunnen vast zetten in een opening die we hebben gemaakt in de basis van de houder.

Voor de plaatsing van de objecten gebruiken we geometrie om de juiste afstanden tot elkaar te bepalen. Het is gebleken dat de spiegel met een brandpuntsafstand van $f = 25mm$ goed gepositioneerd is op een afstand van $v = 4,0cm$ van het sample. Deze staat dan dicht genoeg bij het sample om zoveel mogelijk licht op te vangen maar ook niet te dicht, wat zou maken dat de combinatie van de optische vezel en lens op tientallen centimeters van de spiegel moet staan. Gebruik makend van geometrische optica kunnen we de afstand tussen de spiegel en vezel bepalen.

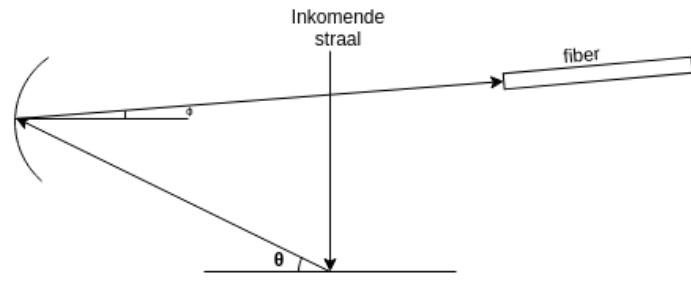
$$b = \frac{1}{\frac{1}{f} - \frac{1}{v}} \approx 66mm \quad (3)$$

Wat de hoeken betreft zetten we de spiegel onder een hoek van $\theta = 15^\circ$ met het sample en de vezel onder een hoek van $\phi = 5^\circ$ met de spiegel. De spiegel zelf zal een kleine offset nodig hebben om ervoor te zorgen dat de bundel licht in de vezel toekomt. Voor de hoek van de spiegel zelf hebben we:

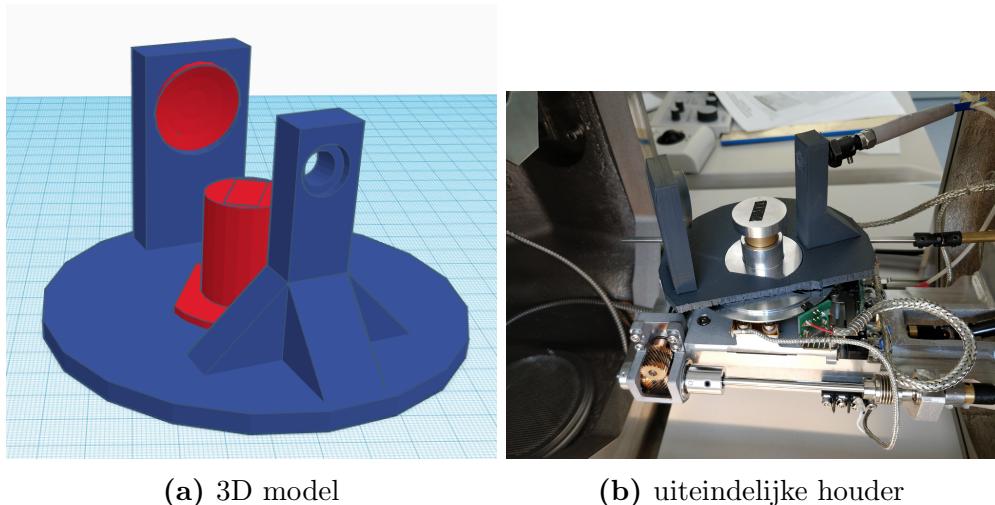
$$\Delta\theta = \frac{\theta + \phi}{2} = 10^\circ \quad (4)$$

Het 3D model maken we op de website [tinkercad.com](https://www.tinkercad.com). Dit is een website waar men met vrij rudimentaire operaties snel aan de slag kan met het ontwerpen van 3D modellen. In figuur 14a zie je een foto van het uiteindelijke model. Na printing van het model zijn er nog een aantal aanpassingen aan de houder uitgevoerd zodat deze precies op de stage past.

1. Bijvijlen van de openingen voor de elementen. Dit moeten we doen omdat cirkels in een 3D model bestaan uit regelmatige veelhoeken. Deze kunnen we met een vijl tot cirkels afronden.



Figuur 13: schets van de geometrie van de opstelling



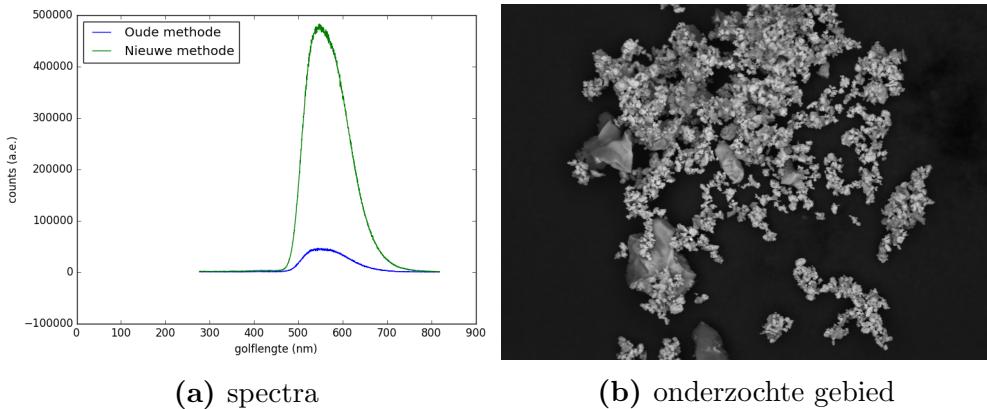
Figuur 14: model vs realiteit

2. Deel van de basis afzagen om rekening te houden met een sensor aan de zijkant van de stage. De weggenomen hoeveelheid stelt ons in staat om de stage veilig in tegenwijzerszin over een hoek van 30° te draaien.

5.3 Vergelijking van oude en nieuwe methode

Om het verschil in lichtopvang tussen beide methodes duidelijk te maken, onderzoeken we voor elk dezelfde spot met een integratietijd van 20 ms en een gain van 100, waarbij we telkens 20 accumulaties uitvoeren. Bij vergelijking van de twee verkregen spectra, kan je zien dat de nieuwe methode een factor 10 meer licht opvangt dan de oude methode met hogere SNR. Deze verbetering aan lichtcollectie zal een aantal gevolgen met zich meebrengen:

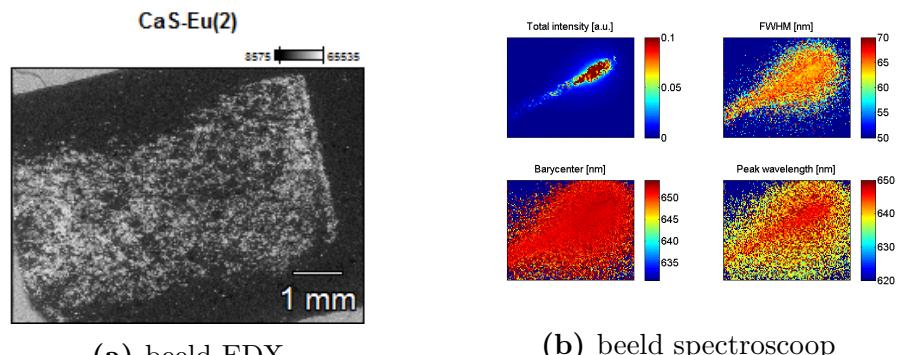
- Experimenten kunnen tot 10 maal sneller uitgevoerd worden



Figuur 15: vergelijking oude en nieuwe methode in de SEM

- Omdat de lichtcollectie met de nieuwe opstelling zo hoog ligt hebben we nu ook de mogelijkheid om te spelen met de slit die de lichttoevoer van de spectroscoop bepaalt. Immers, doordat de resolutie verhoogt zullen in spectra scherpe pieken minder uitgesmeerd worden, zoals bijvoorbeeld bij lijnemissie.

Naast het voordeel van de extra lichtcollectie hebben we ook 2 nadelen bij de nieuwe methode. Het eerste nadeel is dat door de extra voorwerpen op de stage veel lager in de kamer moet geplaatst worden. Dit zal ervoor zorgen dat we niet meer met dezelfde resolutie en vergroting kunnen kijken naar het sample, zowel voor de SE-detector als de EDX. Deze vermindering in resolutie is echter niet zo groot, we kunnen nog steeds individuele korrels herkennen. In het geval er toch een hoge resolutie foto nodig is, is het nog steeds meer rendabel eerst het spectrum op te nemen, vervolgens de houder te verwijderen en dan de hoge resolutie foto te maken. Dit komt omdat het maken van de foto veel minder tijd in beslag neemt dan het maken van CL mappings. Het tweede nadeel van deze opstelling komt omdat de houder zelf mee zal bewegen met de stage in plaats van stil te blijven staan samen met het elektronenkanon. Doordat de houder mee beweegt met het sample kijken we altijd naar dezelfde oppervlakte van het sample. Hierdoor limiteren we onszelf tot een bepaald oppervlak waarin we het spectrum van de cathodoluminescentie kunnen onderzoeken. In figuur 16b ziet men de vorm van de spot die men kan waarnemen. De dimensies van deze spot zijn $1,5\text{mm} \times 3\text{mm}$. Ter vergelijking is de diameter van een typische fosforkorrel $1\text{ }\mu\text{m}$. De spot is met andere woorden groot genoeg waardoor het te onderzoeken fenomeen zich altijd zal manifesteren in deze spot. Voor deze metingen is CaS : Eu^{2+} gebruikt. De reden voor de kleine schaal bij de totale intensiteit bij de beelden van de spectroscoop is



Figuur 16: spotsize nieuwe houder

omdat het fosfor in de spot zelf een fluctuatie in intensiteit heeft. Door deze schaal zo klein te nemen kan men mooi de vorm van de spot bepalen. Binnen het gebied van de rode tot groene tinten kunnen we metingen uitvoeren.

A Appendix

Listing 1: pythoncode simulatie

```
#!/bin/python
from __future__ import division
import subprocess
import numpy as np
import re
import matplotlib.pyplot as plt
import os

directory=os.path.dirname(os.path.realpath(__file__))
print directory

## standard values
#beam: amount, beamwidths
beam = [1000, 5, -5]
#fiber: angle, x, y
fiber = [45, 80, 100]
#lens: angle, radius, curve, x, y
lens = [0, 200, 300, 3000, 0]
#mirror: angle, radius, curve, x, y
mirror = [0, 200, -600, 3000, 0]

def setup_experiment(settings_beam=beam, settings_fiber=fiber,
                     settings_lens=lens, settings_mirror=mirror):
    file1 = open("{{}}/sample.txt".format(directory), "r")
    file2 = open("{{}}/output.txt".format(directory), "w+")
    lines=file1.readlines()
    for line in lines:
        #beam
        line = re.sub('beam1', '{{}}'.format(settings_beam[0]), line.
                      rstrip())
    file2.write(line)
    file1.close()
    file2.close()
```

```

line = re.sub('beam2', '{}'.format(settings_beam[1]), line .
    .rstrip())
line = re.sub('beam3', '{}'.format(settings_beam[2]), line .
    .rstrip())
#fiber
line = re.sub('fiber1', '{}'.format(settings_fiber[0]), line .
    .rstrip())
line = re.sub('fiber2', '{}'.format(settings_fiber[1]), line .
    .rstrip())
line = re.sub('fiber3', '{}'.format(settings_fiber[2]), line .
    .rstrip())
#lens
line = re.sub('lens1', '{}'.format(settings_lens[0]), line .
    .rstrip())
line = re.sub('lens2', '{}'.format(settings_lens[2]), line .
    .rstrip())
line = re.sub('lens3', '{}'.format(settings_lens[1]), line .
    .rstrip())
line = re.sub('lens4', '{}'.format(settings_lens[3]), line .
    .rstrip())
line = re.sub('lens5', '{}'.format(settings_lens[4]), line .
    .rstrip())
#mirror
line = re.sub('mirror1', '{}'.format(settings_mirror[0]), line .
    .rstrip())
line = re.sub('mirror2', '{}'.format(settings_mirror[2]), line .
    .rstrip())
line = re.sub('mirror3', '{}'.format(settings_mirror[1]), line .
    .rstrip())
line = re.sub('mirror4', '{}'.format(settings_mirror[3]), line .
    .rstrip())
line = re.sub('mirror5', '{}'.format(settings_mirror[4]), line .
    .rstrip())

```

```

file2 . write ( line + "\n" )
file1 . close ()
file2 . close ()

def experiment () :
    command = "java -jar {} /OpticalRayTracer.jar -r -t -q < {} /
        output.txt | grep 'optical' | cut -f 13".format(directory ,
    directory)
    exp = subprocess . Popen ( command , shell=True , stdout=subprocess
        . PIPE )
    return exp . stdout . read ()

def beams_in_fiber ( results , angle_fiber ) :
    temp = np . array ([ 180 - float ( i ) for i in results . split ("\n" )
        [ :- 1 ] ])
    return np . count_nonzero (( angle_fiber - 25.4 < temp ) & ( temp <
        angle_fiber + 25.4 ))

## only fiber

def fiber_angle ( start_angle = 10 , stop_angle = 80 , fiber_distance
= 100 , bins = 300 ) :
    angles = np . linspace ( start_angle , stop_angle , bins )
    counts = []
    # settingfiles that do not change
    beam_settings = list ( beam )
    beam_settings [ 2 ] = 0
    # settingsfiles that change
    for angle in angles :
        fiber_settings = [ angle , 20 - fiber_distance * np . cos ( angle
            / 180 * np . pi ) , fiber_distance * np . sin ( angle / 180 * np . pi ) ]

```

```

setup_experiment(settings_beam=beam_settings, settings_fiber=
    fiber_settings)
exp = experiment()
counts.append(beams_in_fiber(exp, angle))
plt.plot(angles, counts, '-o')
plt.xlabel('hoek in graden')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_angle', bbox_inches='tight')
plt.close()

def fiber_distance(start_distance=100, stop_distance=800,
    fiber_angle=45, bins=300):
    distances = np.linspace(start_distance, stop_distance, bins)
    counts = []
    beam_settings = list(beam)
    beam_settings[2] = 0
    for distance in distances:
        fiber_settings = [fiber_angle, 20 - distance * np.cos(
            fiber_angle/180*np.pi), distance * np.sin(fiber_angle/180*
            np.pi)]
        setup_experiment(settings_beam=beam_settings, settings_fiber=
            fiber_settings)
        counts.append(beams_in_fiber(experiment(), fiber_angle))
    plt.plot(distances/100, counts, '-o')
    plt.xlabel('afstand in cm')
    plt.ylabel('aantal invallende lichtstralen')
    plt.savefig('fiber_distance', bbox_inches='tight')
    plt.close()

## fiber and lens
def fiber_angle_lens_angle_distance(start_angle=10, stop_angle
    =80, fiber_distance=800, start_lens_distance=100,
    stop_lens_distance=700, angle_bins=300, distance_bins=300):

```

```

angles = np.linspace(start_angle , stop_angle , angle_bins)
distances = np.linspace(start_lens_distance ,
                       stop_lens_distance , distance_bins)
counts = []
beam_settings = list(beam)
beam_settings [2] = 0
total = []
for angle in angles:
    counts = []
    for distance in distances:
        fiber_settings = [angle , 20 - fiber_distance * np.cos(
            angle/180*np.pi) , fiber_distance * np.sin(angle/180*np
            .pi)]
        lens_settings = list(lens)
        lens_settings [0] = angle
        lens_settings [3] = 20 - distance * np.cos(angle/180*np.pi
            )
        lens_settings [4] = distance * np.sin(angle/180*np.pi)
        setup_experiment(settings_beam=beam_settings ,
                         settings_fiber=fiber_settings , settings_lens=
                         lens_settings)
        counts.append(beams_in_fiber(experiment() , angle))
    plt.plot(distances/100 , counts , '-o' , label='{} graden'.
              format(int(angle)))
    total.append(np.sum(np.array(counts)))
plt.legend()
plt.xlabel('afstand in cm')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_angle_lens_angle_distance' , bbox_inches='
tight')
plt.close()
plt.plot(angles , total , '-o')
plt.xlabel('hoek in graden')

```

```

plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_angle_lens_angle_distance_compair',
            bbox_inches='tight')
plt.close()

def fiber_lens_distance_curve(angle=45, fiber_distance=800,
                               start_lens_distance=100, stop_lens_distance=700,
                               start_lens_curve=200, stop_lens_curve=250, curve_bins=300,
                               distance_bins=300):
    curves = np.linspace(start_lens_curve, stop_lens_curve,
                         curve_bins)
    distances = np.linspace(start_lens_distance,
                           stop_lens_distance, distance_bins)
    counts = []
    beam_settings = list(beam)
    beam_settings[2] = 0
    for curve in curves:
        counts = []
        for distance in distances:
            fiber_settings = [angle, 20 - fiber_distance * np.cos(
                angle/180*np.pi), fiber_distance * np.sin(angle/180*np.pi)]
            lens_settings = list(lens)
            lens_settings[0] = angle
            lens_settings[2] = curve
            lens_settings[3] = 20 - distance * np.cos(angle/180*np.pi)
            lens_settings[4] = distance * np.sin(angle/180*np.pi)
            setup_experiment(settings_beam=beam_settings,
                             settings_fiber=fiber_settings, settings_lens=
                             lens_settings)
            counts.append(beams_in_fiber(experiment(), angle))

```

```

plt.plot(distances/100, counts, '-o', label='{} cm'.format(
    curve/100))
plt.legend()
plt.xlabel('afstand in cm')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_lens_distance_curve', bbox_inches='tight')
plt.close()

def fiber_distance_lens_curve(angle=45, start_fiber_distance=200,
    stop_fiber_distance=800, lens_distance=100, start_lens_curve
    =200, stop_lens_curve=250, curve_bins=300, distance_bins=300):
    curves = np.linspace(start_lens_curve, stop_lens_curve,
        curve_bins)
    distances = np.linspace(start_fiber_distance,
        stop_fiber_distance, distance_bins)
    counts = []
    beam_settings = list(beam)
    beam_settings[2] = 0
    for curve in curves:
        counts = []
        for distance in distances:
            fiber_settings = [angle, 20 - distance * np.cos(angle
                /180*np.pi), distance * np.sin(angle/180*np.pi)]
            lens_settings = list(lens)
            lens_settings[0] = angle
            lens_settings[2] = curve
            lens_settings[3] = 20 - lens_distance * np.cos(angle/180*
                np.pi)
            lens_settings[4] = lens_distance * np.sin(angle/180*np.pi
                )
            setup_experiment(settings_beam=beam_settings,
                settings_fiber=fiber_settings, settings_lens=
                lens_settings)

```

```

counts.append(beams_in_fiber(experiment(), angle))
plt.plot(distances/100, counts, '-o', label='{} cm'.format(
    curve/100))
plt.legend()
plt.xlabel('afstand in cm')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_distance_lens_curve', bbox_inches='tight')
plt.close()

def fiber_lens_distance_curve(fiber_angle = 15, fiber_distance
=1000, lens_angle=15, start_lens_distance=100,
stop_lens_distance=2000, lens_curves=[600], bins=300):
    distances = np.linspace(start_lens_distance,
                            stop_lens_distance, bins)
    counts = []
    beam_settings = list(beam)
    beam_settings[2] = 0
    for curve in lens_curves:
        counts = []
        for distance in distances:
            lens_settings = list(lens)
            lens_settings[0] = lens_angle
            lens_settings[2] = curve
            lens_settings[3] = 20 - distance * np.cos(lens_angle/180*
                np.pi)
            lens_settings[4] = distance * np.sin(lens_angle/180*np.pi
                )
            fiber_settings = [fiber_angle, 20 - (fiber_distance+
                distance) * np.cos(fiber_angle/180*np.pi), (
                fiber_distance+distance) * np.sin(fiber_angle/180*np.
                pi)]
            setup_experiment(settings_beam=beam, settings_fiber=
                fiber_settings, settings_lens=lens_settings)

```

```

counts.append(beams_in_fiber(experiment(), fiber_angle))
plt.plot(distances/100, counts, '-o', label='{} cm'.format(
    curve/100))
plt.legend()
plt.xlabel('afstand tussen lens en fosfor (cm)')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber-lens-distance_{}'.format(len(lens_curves)))
    , bbox_inches='tight')
plt.close()

## fiber en spiegel
def fiber_mirror_distance_curve(fiber_angle=5, fiber_distance
=1000, mirror_angle=15, start_mirror_distance=100,
stop_mirror_distance=2000, mirror_curves=[-600], bins=300):
    distances = np.linspace(start_mirror_distance ,
        stop_mirror_distance , bins)
    counts = []
    beam_settings = list(beam)
    beam_settings[2] = 0
    for curve in mirror_curves:
        counts = []
        for distance in distances:
            mirror_settings = list(mirror)
            mirror_settings[0] = mirror_angle - (mirror_angle +
                fiber_angle) / 2
            mirror_settings[2] = curve
            mirror_settings[3] = 20 - distance * np.cos(mirror_angle
                /180*np.pi)
            mirror_settings[4] = distance * np.sin(mirror_angle/180*
                np.pi)
            print distance , mirror_settings

```

```

fiber_settings = [-fiber_angle , mirror_settings[3] +
    fiber_distance * np.cos( fiber_angle/180*np.pi) ,
    mirror_settings[4] + fiber_distance * np.sin(
    fiber_angle/180*np.pi)]
setup_experiment(settings_beam=beam_settings ,
    settings_fiber=fiber_settings , settings_mirror=
    mirror_settings)
counts.append(beams_in_fiber(experiment()) , 180-
    fiber_angle))
plt.plot(distances/100, counts , '-o' , label='{} cm'.format(-
    curve/200))
plt.legend()
plt.xlabel('afstand tussen spiegel en fosfor (cm)')
plt.ylabel('aantal invallende lichtstralen')
plt.savefig('fiber_mirror_distance_{}'.format(len(
    mirror_curves)) , bbox_inches='tight')
plt.close()

```

```

fiber_angle(bins=50)
fiber_distance(bins=50)
fiber_angle_lens_angle_distance(angle_bins=4, distance_bins=10)
fiber_lens_distance_curve(curve_bins=5, distance_bins=20)
fiber_distance_lens_curve(start_fiber_distance=500,
    stop_fiber_distance=10000, stop_lens_curve=500, lens_distance
    =400, curve_bins=5, distance_bins=20)
fiber_lens_distance_curve(stop_lens_distance=4000, lens_curves=np
    .arange(300,800,100)/2, bins=20)
fiber_mirror_distance_curve(bins=50)
fiber_mirror_distance_curve(fiber_distance=600, mirror_curves
    =[-600], bins=20)

```

```
fiber_mirror_distance_curve(stop_mirror_distance=4000,  
    mirror_curves=(-1)*np.arange(300,800,100), bins=20)
```

Referenties

- [1] P. Lutus. Opticalraytracer. <https://arachnoid.com/OpticalRayTracer/>.
- [2] Lisa Martin, Dirk Poelman, Philippe Smet, and Jonas Joos. Microscopic study of dopant distribution in europium doped srga₂s₄: impact on thermal quenching and phosphor performance. *ECS Journal of Solid State Science and Technology*, 7(1):R3052–R3056, 2018.