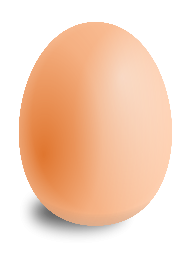
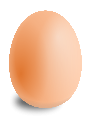
**Eieren tellen**

**met**

**Computer Vision**





Door Alexander de Graaff

27 maart 2021

**Inhoud**

[**1. Inleiding** 2](#_Toc67821083)

[**2. Experiment 1** 2](#_Toc67821084)

[**2.1. Versie 1** 3](#_Toc67821085)

[**2.2. Versie 2** 4](#_Toc67821086)

[**2.3. Uitkomst** 5](#_Toc67821087)

[**3. Experiment 2** 5](#_Toc67821088)

[**4. Conclusie** 6](#_Toc67821089)

[**5. Bronnen** 7](#_Toc67821090)

# **1. Inleiding**

Een lekker eitje bij het ontbijt, wie wil dat niet. Door de UN wordt voorspeld dat we in 2050 met 9.7 miljard mensen op aarde zijn (Elferink & Schierhorn, 2016) en die lusten allemaal wel een eitje. Wegens deze groei voorspelt de UN ook dat de vraag naar eten zal stijgen met 59% tot wel 98%. De verwachtte bevolkingsgroei zal onder andere leiden tot de vraag naar meer kippen(boerderijen). Dit blijkt nu al in de praktijk te zien. Zo is het aantal kippen in de wereld tussen 2000 en 2017 met 37% gestegen tot wel 22.85 miljard (Shahbandeh, 2020).

22.85 miljard kippen produceren dagelijks een gigantische hoeveelheid eieren. Deze eieren worden aan de lopende band geproduceerd wat het voor de boer lastig kan maken om zicht te houden op het officiële productie aantal. Dit probleem is gelukkig op te lossen met computer vision. Je kan boven een lopende band die de eieren transporteert een camera monteren die telt hoeveel eieren er worden geproduceerd.

Het klinkt eenvoudig om de kippenboerderijen gebruik te laten maken van een dergelijke techniek, het is alleen makkelijker gezegd dan gedaan. Zo kan een camera niet uit zichzelf bepalen hoeveel eieren hij voor zich heeft liggen. Daarvoor maak je gebruik van computer vision. Computer vision is het gebruik maken van programma’s om afbeeldingen en video’s te verwerken en begrijpen. In het geval van eieren tellen voer je de afbeeldingen die je maakt van eieren op de lopende band aan een programma dat de eieren voor je telt. Maar hoe doet het programma dat? Het gebeurt globaal in twee stappen:

* De eerste stap focust zich op het simpeler maken van de afbeelding zodat er makkelijker mee gewerkt kan worden.
* De tweede stap werkt verder met die versimpelde afbeelding. Deze stap rekent van een set afbeeldingen uit hoe groot een ei gemiddeld is en gebruikt vervolgens dit gemiddelde om de andere afbeeldingen te analyseren en te bepalen hoeveel eieren zich op de afbeelding bevinden.

Is het in praktijk mogelijk met deze twee stappen om van een gegeven afbeeldingen het aantal eieren te tellen?

# **2. Experiment 1**

Deze eerste stap, hierna genoemd als experiment 1, is gebaseerd op het werk dat gedaan is door een team van het College of Engineering (hierna CoE), Laboratory of Modern Facility Agriculture Technology and Equipment Engineering of Jiangsu Province, Nanjing Agricultural University, Jiangsu, China, gepubliceerd in Journal of Food Engineering (JoFE, 2020). Zij concluderen dat het makkelijker is te werken met een versimpelde versie van een foto. Deze versimpeling doen zij met behulp van een diepte foto van de eieren, waarna ze kijken naar “distance intensities of the depth images” (JoFE, 2020) om zo het verschil te zien tussen het ei en de achtergrond. Hoewel dit een uitstekende manier is voor het onderscheiden van de achtergrond is een diepte camera een gespecialiseerde gadget die niet elke boer in zijn schuur heeft liggen. Om dit programma meer toegankelijk te maken voor een boer is het handig om het bruikbaar te maken voor een eenvoudigere camera. In plaats van te kijken naar de diepte wordt gekeken naar de kleur van de achtergrond. Zo wordt het ei makkelijk te onderscheiden van de anders gekleurde achtergrond. Nadat we hebben vastgesteld wat deel uitmaakt van de achtergrond en wat niet, kleuren we de achtergrond en de eieren in met een aparte contrasterende kleur om zo in de toekomst makkelijk onderscheid tussen de twee te kunnen maken.

Experiment 1 is essentieel voor de goede werking van de tweede stap en moet als resultaat daarom zo exact mogelijk onderscheid maken tussen de eieren en de achtergrond. Het experiment wordt op twee verschillende manieren benaderd, om te kijken welke aanpak de beste resultaten geeft.

## **2.1. Versie 1**

De eerste versie van dit experiment probeert hetzelfde te bereiken als het werk van the CoE. Er wordt gebruik gemaakt van dezelfde dataset. Hoewel zij deze niet in zijn totaliteit vrijgeven laten ze wel een aantal voorbeelden zien. Het nadeel van deze voorbeelden is dat de foto’s van lage kwaliteit zijn, wat slechte gevolgen had voor de uitkomt van het experiment. The CoE maakt deze versimpelde afbeeldingen met de eerdergenoemde diepte foto’s, helaas geven zij de originelen niet vrij waardoor hun 3 stappenplan niet gevolgd kan worden. De manier waarop the CoE het deed was als volgt:

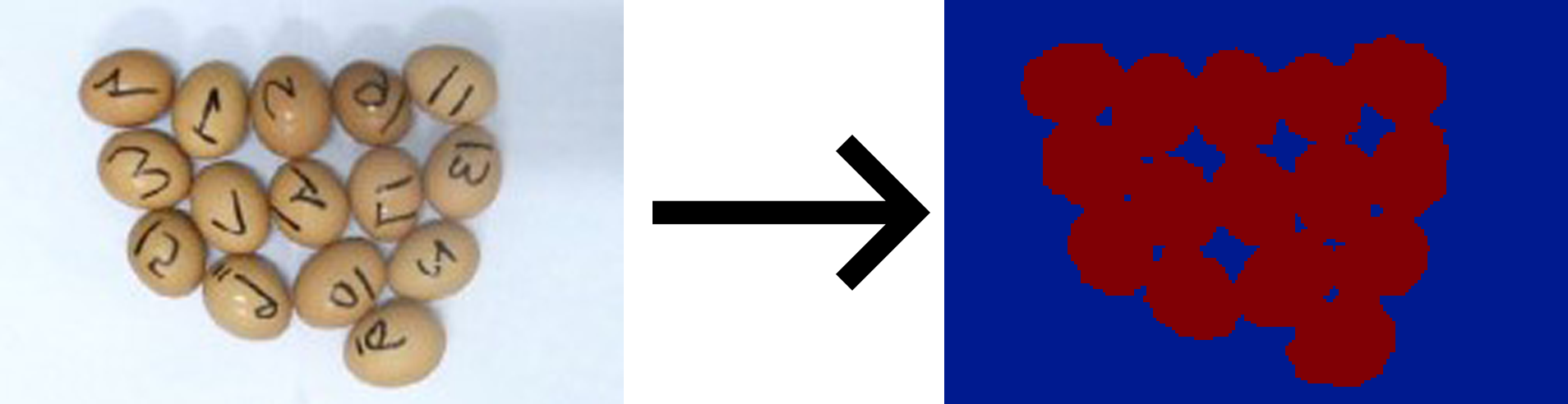
* “Firstly, based on distance intensities of the depth images, minimum and maximum thresholds were set to segment the Region of Interest”
* “Secondly, a smoothening operation was performed baby a 5x5 zero-mean Gaussian Kernel Filter”
* “Thirdly, the morphological opening operation was performed by a disk structural element of size 6 pixels to remove holes in the image object.“

Helaas is de eerste stap onmogelijk omdat de diepte foto’s niet vrijgegeven zijn wat stap 2 en 3 niet uit te voeren maakt.

Hoewel de diepte foto’s niet vrijgegeven werden is zoals eerder genoemd er wel een kleine dataset aan foto’s gegeven. Versie 1 werkt verder met deze foto’s. Deze versie probeert onderscheid te maken tussen de eieren en de achtergrond door de foto om te zetten naar een HSV (Howard, 2019) en vervolgens te kijken in de eerste channel, de hue channel. Het programma maakt gebruik van deze channel wegens een simpele reden: “Hue is the color portion of the model” (Howard, 2019). We kunnen daarmee tegen het programma zeggen dat we van elke pixel in de foto willen checken of de hue waarde gelijk is aan dat van een witte pixel, zo ja teken het blauw, zo nee teken het rood (Deze kleuren komen overeen met de aanpak van the CoE).

De reden dat hier gekozen is voor HSV voor het zoeken naar de kleur in plaats van de meer bekende RGB is omdat je bij HSV maar één channel hoeft te bekijken om te checken of iets wit is terwijl bij RGB alle drie de channels een bepaalde waarden moeten overtreffen.

Hoewel het gebruiken van HSV goed werkt in deze situatie is de uitkomst van het programma verre van perfect, zoals te zien is in de onderstaande afbeelding. De afbeelding heeft erg blokkerige randen en representeert de vormen van de eieren niet goed. De voornaamste redenen hiervoor is de slechte van de kwaliteit van de afbeelding (200x131 pixels). Dit zorgt ervoor dat de randen van de eieren maar een paar pixels zijn waardoor het programma moeite heeft onderscheid te maken rond deze randen.



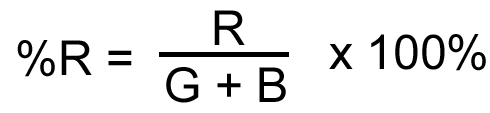
Het resultaat van de eerste versie van experiment 1 is dus geen succes. Gelukkig is er een andere manier die geprobeerd kan worden.

## **2.2. Versie 2**

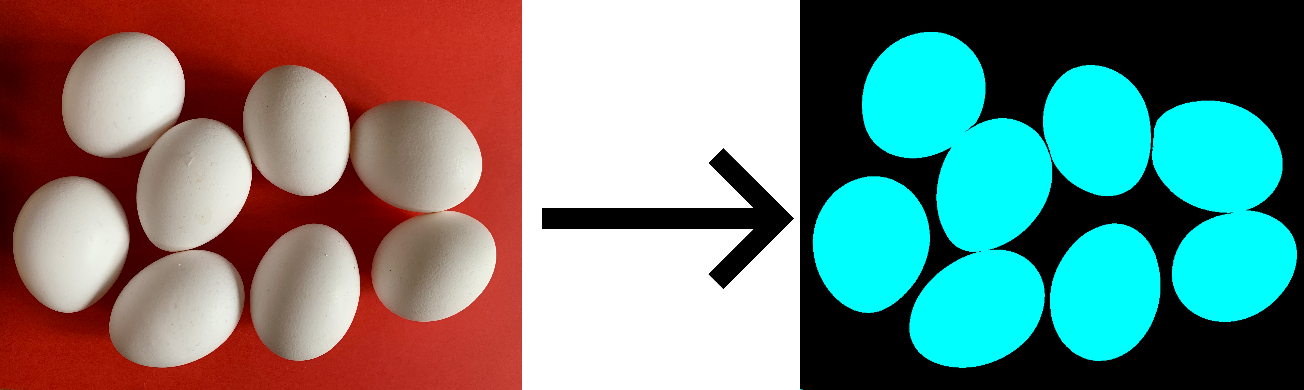
De kleine dataset die the |CoE heeft vrijgegeven is dus moeilijk te gebruiken, vandaar is het beste om een nieuwe dataset te gebruiken. Waar moet deze dataset aan voldoen? Het moet in ieder geval uit meerdere afbeeldingen bestaan waar eieren worden afgebeeld in verschillende formaties. Het moet van bovenaf worden gefotografeerd om zoveel mogelijk van het ei te kunnen zien. Dit is tot nu toe allemaal hetzelfde zoals the CoE het heeft gedaan, met één groot verschil. In plaats van de eieren af te beelden op een witte achtergrond is het beter om dat te doen op een rode, groene of blauwe achtergrond. De reden hiervoor is dat met RGB makkelijker onderscheid is te maken tussen één van deze kleuren dan met wit.

Laten we willekeurig kiezen voor een rode achtergrond. We plaatsen op de achtergrond een set eieren en maken een foto. Het resultaat geven we door aan het nieuwe programma dat daar vervolgens mee aan de slag gaat. Dit programma kijkt net zoals die van versie 1 naar elke pixel van de gegeven afbeelding, maar in plaats van dat hij de afbeelding omzet naar een HSV en kijkt naar de hue waarde houdt hij de afbeelding als een RGB-afbeelding en kijkt naar de relatie tussen de 3 kleurkanalen.

Aangezien we voor een rode achtergrond hebben gekozen kijken we naar de relatie tussen het R (rood) kanaal met de G (groen) en B (blauw) kanaal. Waar we achter willen komen is of de pixel rood is en dus deel uitmaakt van de achtergrond. Dat wordt gedaan met de volgende formule:



Deze formule berekent welk percentage rood is in vergelijking tot het groen en blauw. In het programma wordt een pixel als rood geteld als %R meer of gelijk is aan 110%. Nu we weten hoe we rood herkennen, kunnen we het programma vertellen dat zodra een pixel rood is deze zwart gemaakt moet worden. Is de pixel niet rood dan maken we hem cyaan. Deze methode heeft veel meer succes dan de voorgaande, zoals te zien is in de onderstaande afbeelding:



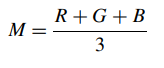
## 

## **2.3. Uitkomst**

Het resultaat van de tweede versie ziet er zonder enige twijfel beter uit en heeft meer gedefinieerde randen. Er is helaas wel een addertje onder het gras. De input afbeelding van de eerste was ongeveer 200\*131 pixels. Dit is erg klein waardoor veel detail verloren gaat, maar zorgt er wel voor dat het programma snel klaar is. Dit in tegenstelling tot de input van de tweede versie, deze input afbeelding is 3.024\*4.032. Dit zijn ongeveer 46500% meer pixels. Zoals voor te stellen is heeft dit een impact op de snelheid van het programma. Om deze methode bruikbaar te maken moet de afbeelding worden verkleind voordat alle pixels bekeken worden.

# **3. Experiment 2**

Nu experiment 1 geoptimaliseerd is en een goede output afbeelding geeft kan begonnen worden met experiment 2; het tellen van de eieren. De eerste stap is om de afbeelding die uit experiment 1 komt om te zetten naar een zwart wit afbeelding om het makkelijker te maken om mee te werken. Waarom is dit makkelijker? De afbeelding uit experiment 1 bevat nog steeds kleur die in RGB of HSV zijn opgeslagen. Beide kleur opslagmethodes hebben 3 kanalen (Pedamkar, P) (Howard, 2019). Dat is erg handig als je daar gebruik van wil maken, maar als je alleen het verschil wil zien tussen twee verschillende vlakken is het erg overbodig om daar 3 kleuren kanalen voor te gebruiken. Dit probleem bestaat niet bij een zwart wit afbeelding, zo`n afbeelding wordt ook wel *grayscale* (Scikit-Image) genoemd. Grayscale heeft maar één datachannel zoals makkelijk te zien is in deze formule (Klette, 2014):



Zoals te zien is worden alle drie de channels van de RGB-format samengevoegd tot maar één channel. Nu de afbeelding omgezet is naar grayscale kan er een begin gemaakt worden met het echte werk.

De volgende stap is om het programma te trainen eieren te tellen. Voor deze stap zijn er meerdere afbeeldingen nodig. Hoe meer hoe beter. De nieuwe afbeeldingen worden aan het programma gevoed samen met het correcte aantal eieren dat op die afbeelding wordt afgebeeld. Het programma telt vervolgens per afbeelding hoeveel van de pixels cyaan zijn en deelt dat door het aantal eieren dat afgebeeld is. Het resultaat is per afbeelding een gemiddelde grote van een ei. Als dat voor een groot aantal afbeeldingen wordt gedaan geeft dat een hele collectie aan gemiddelden. Deze gemiddelden worden bij elkaar opgeteld en gedeeld door het totaal aantal afbeeldingen dat werd ingevoerd. Dit geeft de gemiddelde grote van een ei.

Nu de gemiddelde grootte bekend is, kan dit toegepast worden op andere afbeeldingen. Het programma doet nu als volgt: het programma krijgt een versimpelde afbeelding uit experiment 1, telt vervolgens daar het aantal pixels dat cyaan is, deelt dit door de gemiddelde grote van een ei, rond deze uitkomt af met als resultaat het aantal afgebeelde eieren.

Natuurlijk hebben niet alle eieren exact dezelfde grootte. Daarom moet je rekening houden met een marge. Bij grote aantallen, zoals met de eieren het geval is zal het gemiddelde redelijk kloppen. Maar een enkel ei kan natuurlijk groter of kleiner zijn. Het programma zal bij een foto met alleen maar grote eieren denken dat er meer eieren zijn dan er werkelijkheid zijn afgebeeld. Hierom is het belangrijk dat er een groot aantal foto’s van verschillende eieren aan het programma gegeven wordt tijdens het train proces zodat het gemiddelde goed de verschillende groottes van de eieren representeert. Als het gemiddelde goed is zal de telling op een groot aantal foto’s perfect zijn.

# **4. Conclusie**

Is het door gebruik te maken van computer vision mogelijk om van een gegeven afbeeldingen het aantal eieren te tellen?

Ja, ik heb uitgevonden dat het mogelijk is om via computer vision op een efficiënte manier eieren te tellen. Het enige wat ervoor nodig is een camera die foto’s kan maken van een aantal eieren en deze vervolgens aan het programma kan voeren. Dit programma kan dus relatief makkelijk gebruikt worden door een boer die de telling van zijn eieren wil automatiseren. Het programma kan goed bepalen hoeveel eieren zich op een afbeelding bevinden. Enige nadeel is dat er een marge bestaat waardoor er fouten kunnen ontstaan bij het tellen als de eieren extreem afwijken van het gemiddelde en dus erg groot of klein zijn. Dit probleem is wellicht te ondervangen door een voorselectie op de lopende band op gewicht of grootte, maar dat heb ik niet onderzocht.

Daarnaast is het essentieel dat het beeld uit voldoende resolutie bestaat om zo de grens tussen het ei en de achtergrond goed te kunnen onderscheiden.

Of het werken met “depth images” net zo goed of beter werkt kan ik niet beoordelen, omdat het niet mogelijk was hier een experiment mee uit te voeren. Wel kan worden gezegd dat deze werkwijze complexer is dan mijn methode.

# **5. Bronnen**

1. Elferink, M & Schierhorn, F. (April 07, 2016), “Global Demand for Food Is Rising. Can We Meet It?”, [Global Demand for Food Is Rising. Can We Meet It? (hbr.org)](https://hbr.org/2016/04/global-demand-for-food-is-rising-can-we-meet-it)

2. Shahbandeh, M. (Maart 07, 2020), “Broiler meat production worldwide in 2020”,  
[• Broiler meat production worldwide by country, 2020 | Statista](https://www.statista.com/statistics/237597/leading-10-countries-worldwide-in-poultry-meat-production-in-2007/)

3. Journal of Food Engineering. (Oktober 2020), “Egg volume estimation based on image processing and computer vision”, <https://www.sciencedirect.com/science/article/pii/S0260877420301394>

4. Klette, R. (2014), “Concise Computer Vision”, London: Springer-Verslag

5. Pedamkar, P. “RGB Color Model”,  
[RGB Color Model | How It Work | Uses & Example | Advantages (educba.com)](https://www.educba.com/rgb-color-model/)

6. Howard, J. (November 11, 2019), “The HSV Color Model in Graphic Design”,  
[What Is the HSV (Hue, Saturation, Value) Color Model? (lifewire.com)](https://www.lifewire.com/what-is-hsv-in-design-1078068)

7. Scikit-Image. “RGB to grayscale”  
[RGB to grayscale — skimage v0.19.0.dev0 docs (scikit-image.org)](https://scikit-image.org/docs/dev/auto_examples/color_exposure/plot_rgb_to_gray.html)