# Labboek Project ZSB

Pieter Kronemeijer (11064838)
Douwe van der Wal (11042206)
Shankara van de Ven (11000791)
Laurens Weitkamp

23 juni 2016

# 1 Inleiding

Voor de vierde week van het ZSB project hebben wij de opdracht gekregen om iets te bedenken en te implementeren wat vorige jaren niet gemaakt was("go, where no has gone before"). In dit project zullen wij een drone een toverhoed laten volgen. Deze toverhoed representeert een kenmerk dat de dader van een winkeldiefstal bij zich kan dragen. Door deze toverhoed te kunnen volgen met een drone kunnen daders gemakkelijk gevolgd en onderschept worden door de politie. De hoofdvraag die wij stellen luidt als volgt: "Hoe kan een quadcopter met camera's een object blijven volgen in een begrensd gebied?"Wij verwachten dat wij hier 'tracking software' voor nodig hebben dat gebruik maakt van 'computer vision'. Om deze hoofdvraag te beantwoorden is er gekozen voor de volgende drie deelvragen, ieder met zijn eigen hypothese.
De eerste deelvraag, en wellicht de kern van dit onderzoek luidt als volgt:"Hoe kan de drone een specifiek object volgen, met behulp van een camera?"Wij veronderstellen dat een tracking systeem al bestaat dat wij kunnen implementeren in onze software.
Maar wat als de drone het object uit het oog verliest? Het volgen van een object wordt dan vanzelfsprekend onmogelijk. Daarom stellen wij de volgende deelvraag: "Hoe kan de drone een specifiek object vinden als deze uit zicht is?"De verwachting is dat de drone simpel rondjes kan draaien tot hij het object gevonden heeft.
Een laatste toevoeging kan essentieel zijn als de drone in de 'echte' wereld gebruikt gaat worden. Een drone kan namelijk niet voor eeuwig blijven vliegen. Er zit een batterij in die na verloop van tijd leeg is, wat zorgt voor een beperkte radius vanaf de vliegbasis. De volgende deelvraag volgt uit dit probleem: "Hoe kan de drone in een specifieke omgeving blijven, en deze niet verlaten?"Wij verwachten dat de drone bijhoudt hoe veel meter hij heeft afgelegd en op basis daarvan in een bepaalde radius van de oorsprong blijft.

# 2 Project week 4

### 2.0.1 Maandag

Wij hebben ervoor gekozen om gebruik te maken van de Parrot AR.Drone 2.0 [1]. Dit is een quadcopter (een drone met vier propellers) die aangestuurd kan worden door middel van een smartphone app genaamd freeflight 2.4.10 (Android/iOS) of via Linux. Ons project richt zich op het autonoom vliegen van de quadcopter dus deze app zullen wij niet gebruiken. De keuze voor deze drone volgt uit het feit dat er een camera op zit, in tegenstelling tot de kleinere drones. De eerste poging in dit project was het node-ar-drone project van github [2]. Dit gebruikt node.js [3] om de drone te besturen. Dit lijkt te werken, echter is de videofeed niet toegankelijk terwijl dit wel essentieel is voor het volgen van een object. Daarom is er voor gekozen om dit project voort te laten bouwen op een bestaand project genaamd CV Drone [4]. Dit is een combinatie van OpenCV [5] en software specifiek voor de AR.Drone geschreven in c++. OpenCV staat voor 'Open Source Computer Vision' en is ontwikkeld om 'computer vision' mogelijk te maken. Met CV Drone is het mogelijk om bewegingen, zoals naar rechts en omhoog te programmeren. Ook zitten er verschillende voorbeelden bij de software, zoals een tracking module die vierkante objecten kan volgen mits er genoeg contrast is met omliggende kleuren. Deze tracking module is goed te gebruiken in dit project. De software werkt echter niet op alle laptops. Het lijkt erop dat de connectie soms wegvalt, en vervolgens bevriest het beeld. De connectie lijkt zich, aan de hoeveelheid internetverkeer te zien, te herstellen, maar dit zorgt niet voor een herstelde videofeed.
Door aanpassing van de KVH waardes (Kleurtoon, Verzadiging, Helderheid) en gebruik makend van de tracking module volgt de drone nu een geel doekje. Deze KVH waardes geven aan in welk bereik de waargenomen kleuren moeten liggen. Zo geeft de helderheid aan hoe donker het geel mag zijn. Deze waarde kan verschillen door verschillend lichtinval wat veroorzaakt kan worden

door het doekje schuiner of minder schuin te houden. De drone houdt echter geen rekening met de afstand tot het doekje en focust zich nog vaak op erg kleine gele objecten.

Tabel 1: KVH waardes gele kleur doekje

| | min | max |
|---|---|---|
| Kleur | 33 | 61 |
| Verzadiging | 129 | 255 |
| Helderheid | 69 | 255 |

### 2.0.2 Dinsdag

Het probleem met de tracking module van CV Drone is dat de drone te dicht naar het object dat hij trackt toe vliegt. Om een specifieke afstand tot het object te bewaren moet de grootte van het object berekent worden en constant gehouden worden. Immers, wanneer het object zich op een grotere afstand bevindt zijn zijn afmetingen kleiner dan wanneer hij dichtbij staat. OpenCV zet een contour om het gevonden voorwerp. Dit contour wordt opgeslagen als een vector en bevat veel aaneengesloten punten met allemaal hun x en y waarde. De afmetingen van dit contour probeerden wij eerst te verkrijgen door het kleinste punt en het grootste punt uit de vector te pakken en het verschil in x en y coördinaten te berekenen, zie deze functie:

**Vind oppervlakte countour**
**for** *aantal punten in vector* **do**
  initialiseer grootste dx+dy;
  initialiseer kleinste dx+dy;
  **if** *dx+dy is groter dan oude dx+dy* **then**
    grootste dx+dy is nieuwe dx+dy;
  **end**
  **if** *dx+dy is kleiner dan oude dx+dy* **then**
    kleinste dx+dy is nieuwe dx+dy;
  **end**
**end**
oppervlak is grootste min kleinste dx+dy;
geef oppervlak;

Dit werkte helaas niet goed, een gedraaid voorwerp wordt soms groter gezien dan een recht voorwerp omdat bij een draaiïng de x of y soms groter wordt dan dat de y of x verkleint. Uiteindelijk is gekozen voor een ingebouwde functie, namelijk contourArea. De tijd die besteed is aan het schrijven van een eigen functie is echter niet verspild. Het opvragen van de coördinaten kon later nog gebruikt worden. Nu het volgen op een gespecificeerde afstand werkt is deelvraag 1 beantwoord. Maar wat als de drone het object uit het oog verliest? Het project richt zich nu op deelvraag 2. Een simpele maar doeltreffende oplossing is om de drone een rondje te laten draaien. Dit werkt en de volledige code voor tracking via de camera aan de voorkant ziet er als volgt uit:

**Volgen via de camera aan de voorkant**

```
if volgen en camera aan de voorkant staat aan then
    if oppervlakte contour groter dan 200 then
        if oppervlakte contour groter dan 10000 then
            drone achteruit;
        else
            if oppervlakte contour kleiner dan 4000 then
                drone vooruit;
            end
        end
    else
        roteren;
    end
end
```

Er is gekozen voor de vereiste dat contour oppervlakte groter dan 200 moet zijn. Dit zorgt er voor dat zeer kleine plekjes geel over worden geslagen en de drone zich hier niet op fixeert.

### 2.0.3 Woensdag

Nu de deelvragen 1 en 2 beantwoord zijn kan er gekeken worden naar het vliegen in een begrensd gebied. Dit gebied is groen en wordt begrensd door witte lijnen. Deze witte lijnen blijken het best detecteerbaar met de volgende KVH-waardes:

Tabel 2: KVH waardes witte lijn

| | min | max |
|---|---|---|
| Kleur | 32 | 156 |
| Verzadiging | 3 | 36 |
| Helderheid | 164 | 255 |

De drone bevat behalve zijn camera aan de voorkant ook een camera aan de onderkant. Het plan was om de drone naar achter te laten vliegen wanneer er een witte lijn in beeld kwam. Met de bovenstaande KVH-waardes en een juiste contour afmeting zou dit hetzelfde moeten werken als het gele doekje detecteren. Dit was inderdaad het geval, echter trad er een probleem op. Door de instabiliteit van de drone ging hij vaak over de lijn heen en kon de drone niet meer bepalen welke kant hij op moest vliegen. Daarom is er voor gekozen om een rood gebied te nemen met de groene vloer als grenzen. Omdat deze grenzen breder zijn dan de witte was het probleem opgelost en ging de drone netjes naar achteren zodra hij het groen detecteerde. Het oppervlakte van het contour mocht eerst 2000 worden, hierbij vloog de drone echter al buiten het veld. Daarom is er gekozen voor een contour waarde van 1000. Het groen van de vloer had de volgende KVH-waardes:

Tabel 3: KVH waardes groene achtergrond

| | min | max |
|---|---|---|
| Kleur | 94 | 136 |
| Verzadiging | 100 | 255 |
| Helderheid | 45 | 255 |

### 2.0.4 Donderdag

De drone kon nu een geel doekje volgen en binnen de grenzen van het veld blijven. Echter nog niet tegelijkertijd. Er moet geswitch worden tussen de twee camera's waardoor telkens maar

één van de twee implementaties (het tracken en in het begrensde gebied blijven) kan worden uitgevoerd. Door de datastroom van de drone richting de laptop te controleren werd duidelijk dat de drone nooit meer dan één camera beeld aan het uitzenden is. Zonder camera aan is de output 3 Mbps, met onder camera 6 Mbps en met de camera aan de voorkant 8 Mbps. Het eerste wat wij probeerden was het continu switchen tussen de camera's. Dit bleek geen goede oplossing. Het OpenCV kon dit simpel niet aan en OpenCV paste soms het detecteren van grenzen toe op de camera aan de voorkant en probeerde het gele doekje te vinden met de camera aan de onderkant. Om dit op te lossen is er uiteindelijk gekozen om het wisselen tussen de camera's minder vaak te doen.
Door deze aanpassing werkte het programma erg goed. Echter werd soms nog een frame van de verkeerde camera geanalyseerd, omdat er te snel het laatst opgenomen frame werd opgevraagd. Daarom probeerden we in eerste instantie een pauze in te lassen na het wisselen van de camera. Deze pauze stond op 50 milliseconden wat tot op een zekere hoogte werkte. Het aantal keer dat er een frame van een verkeerde camera voorkwam was nu ongeveer 20 keer minder. Met 55 milliseconde pauze zijn ze niet meer waargenomen.
Dit programma wisselt iedere 5 frames tussen de camera aan de onderkant en de camera aan de bovenkant. Wanneer onder camera aan staat kijkt hij of groen in beeld is, en zo ja, of dat bovenin het scherm zit of juist onderin. Als de camera aan de voorkant aan staat gebruikt hij deze informatie om te bepalen of hij vooruit of achteruit moet of dat hij door kan gaan met het volgen van het gele doekje.
**Kies juiste camera en check waar het groen zich bevindt**

```
if teller kleiner dan 5 then
    gebruik KVH waardes voor geel;
    if volgen en camera aan de voorkant staat aan then
        if oppervlakte contour groter dan 200 then
            if oppervlakte contour groter dan 11000 then
                if groen voor zich then
                    drone achteruit;
                else
                    drone stil staan;
                end
            else
                if oppervlakte contour kleiner dan 5000 then
                    if groen achter zich then
                        drone vooruit;
                    else
                        drone stil staan;
                    end
                end
            end
        else
            roteren;
        end
    end
else
    if teller kleiner dan 10 then
        gebruik KVH waardes voor groen;
        if groen in beeld then
            if groen in bovenste helft beeld then
                groen voor zich;
            else
                if groen in onderste helft beeld then
                    groen achter zich;
                end
            end
        end
    else
        teller is 0
    end
end
teller plus 1;
```

#### 2.0.5 Vrijdag

## 3 Discussie

geen videofeed tegelijkertijd

### 3.1 Conclusies

### 3.2 Vervolgvragen

### 3.3 Open issues

1. Het grootste probleem is dat er een vertraging zit in het wisselen van camera. De drone zou de videofeeds van de onder- en voorcamera simultaan moeten versturen. Als dit mogelijk is kan de drone soepeler en sneller bewegen en is er daarnaast minder kans dat deze te ver van het tapijt vliegt, omdat er dan altijd naar het beeld van een camera wordt gekeken in plaats van maar 50 procent van de tijd.

2. groen zijkant

3. c

### 3.4 Wat hebben wij geleerd tijdens dit project?

## 4 Appendix

https://www.youtube.com/watch?v=hRhjeULyCvk (alleen tracking van geel)
https://www.youtube.com/watch?v=IcS6q6wB8oQ (tracking van geel en niet van het tapijt)
https://www.youtube.com/watch?v=noRpkdN0IhI (drone vliegt toch van het tapijt)

## Referenties

[1] AR.Drone 2.0, http://www.parrot.com/nl/producten/ardrone-2/.

[2] Felix Geisendörfer, *node-ar-drone*, https://github.com/felixge/node-ar-drone.

[3] nodejs.

[4] puko0x, *cvdrone*, https://github.com/puku0x/cvdrone.

[5] opencv.