KU Leuven
Departement Computerwetenschappen

P&O: Computerwetenschappen

Tussentijds verslag

Team:
Zilver

Bram Vandendriessche (Coördinator)
Arne Vlietinck (Secretaris)
Matthias Van der Heyden
Jef Versyck
Vincent Vliegen
Laura Vranken

Academiejaar 2016-2017

Samenvatting

Auteur:

Inhoudsopgave

1 Inleiding

Auteur: Laura Vranken & Arne Vlietinck

Drones zijn de laatste jaren enorm in populariteit toegenomen en blijven hierdoor ook in positieve zin evolueren. Ze worden tegenwoordig gebruikt voor talloze toepassingen maar evenzeer voor ontspanningsdoeleinden. De bekendste toepassing bevindt zich binnen defensie die zo informatie kan winnen over vijandelijk gebied zonder mensenlevens te moeten riskeren. Daarnaast hebben ook grote bedrijven (e.g.: Amazon) de weg naar deze technologie gevonden. De toekomst brengt echter nog veel meer voordelen. Enkele voorbeelden zijn veiligheidsinspectie van windturbines of elektriciteitslijnen, luchtsteun bij zoek- en reddingsoperaties, bewaking, luchtfotografie etc. Om de verscheidenheid aan toepassingen te kunnen verwezenlijken, is natuurlijk een goede aansturing van belang. De drone kan aangestuurd worden via een persoon op de grond met een controller ofwel volledig autonoom. ([**?**])
Wanneer een drone autonoom functioneert, is een betrouwbare aansturing door de Autopilot van levensbelang. Hij moet namelijk bestand zijn tegen alle (eventueel extreme) weersomstandigheden. De twee basis doelen van de Autopilot zijn: het doel bereiken op de meest efficiënte manier en de drone veilig laten landen op een aangegeven plaats.

Aangezien een autonome drone zijn informatie haalt uit de beelden die hij produceert, is de kwaliteit van de camera's van groot belang. Indien geen goed beeld verkregen wordt, is de relatieve positiebepaling moeilijker en minder precies te bepalen. Bovendien moet de Autopilot ook rekening houden met het verschil in intensiteit van de kleuren door bijvoorbeeld invloed van schaduw, felle zon en regendruppels op de camera. Door deze twee oorzaken kunnen toepassingen, waar nauwkeurigheid van belang is, hun nut verliezen.
In dit verslag gaan we verder in op hoe de autonome aansturing van een drone (en meer bepaald een quadcopter) gebeurt. We gaan uit van een drone waarop twee camera's vast bevestigd zijn op een zekere afstand van elkaar. Beiden zijn ze voorwaarts gericht. Op basis van deze beelden moeten afstand en positie van het doel ingeschat worden en nieuwe bewegingsopdrachten voor de drone gegenereerd worden. Aangezien er geen hardware ter beschikking was, moet er een Virtual Testbed simulator ontworpen worden. Dit is een softwaresysteem dat een fysieke opstelling van een drone en camera's simuleert. ([**?**]) De simulator genereert beelden van de drone in verschillende standpunten a.d.h.v. de verkregen bewegingsopdrachten van de Autopilot.

De Autopilot en Virtual Testbed moeten zo ontworpen worden dat de drone in staat is om zijn doel, een rode bol, te lokaliseren en er naar toe te vliegen. Dit eventueel onder lichte invloed van wind in willekeurige richtingen. Bovendien moet ook voor beiden een GUI ontworpen worden die ter beschikking staat van de gebruiker. De GUI toont de vooruitgang en laat de gebruiker toe allerlei informatie (snelheid, positie en verschillende camerastandpunten) op te vragen.

De tekst is als volgt opgebouwd. Eerst wordt het ontwerp van de Autopilot en Virtual Testbed verder uitgediept in sectie 2. Vervolgens wordt er ingegaan op de gebruikte algoritmen (sectie 3) en wordt de opbouw van onze software verduidelijkt (sectie 4). Ook wijden we uit over de toepassingen van de GUI in sectie 5. Tenslotte sluiten we in sectie 6 af met de uitgevoerde testen te bespreken, het besluit is te vinden in sectie 8.

2 Ontwerp

Auteurs: ; redactie: Arne Vlietinck

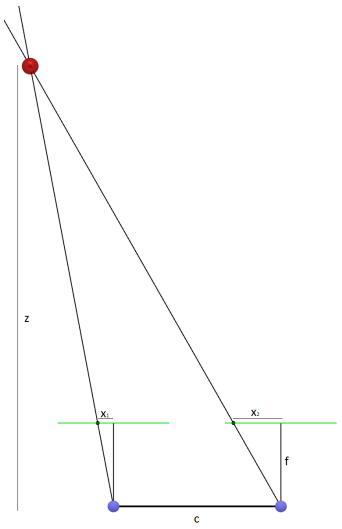
2.1 Drone Autopilot

Auteur: Laura Vranken

De Drone Autopilot zorgt voor de besturing van de drone. Het bepaalt zijn positie relatief ten opzichte van zijn doel a.d.h.v. twee beelden. Deze beelden zijn gemaakt door twee camera's die op de drone gemonteerd staan op een vaste afstand van elkaar. Bovendien kunnen ze opgevraagd worden vanuit de GUI van het Virtual Testbed. Vervolgens zorgt de Autopilot ervoor dat de drone juist geöriënteerd staat en naar zijn doel toe vliegt. Wanneer de drone zijn doel (in deze eerste fase is dit een rode bol) bereikt, moet hij daarin blijven zweven. Tenslotte moet de Autopilot ook rekening houden met een mogelijke invloed van wind die de drone van zijn koers doet afwijken.

Ten eerste moeten dus de beelden die de Autopilot van de Virtual Testbed binnen krijgt, geanalyseerd worden. Dit gebeurt door iteratief de integer waarden van elke pixel te vergelijken met de waarde van het kleur rood. Alle rode pixels worden bijgehouden door hun positie ten opzichte van het beeld, uitgedrukt in rij en kolom, op te slaan. We baseren onze komende berekeningen op het midden van de bol. Dat midden kan bepaald worden door het zwaartepunt van de rode pixels te berekenen via het gemiddelde van de opgeslagen coördinaten.
Indien de Autopilot geen rode pixels detecteert, zal de drone 360 graden ronddraaien of met andere woorden een yaw beweging uitvoeren, totdat in beide beelden rode pixels verschijnen. In het geval dat de Autopilot slechts rode pixels in een van de twee beelden opmerkt, zal het zichzelf door middel van een yaw beweging in de juiste richting bijsturen. Wanneer uiteindelijk in beide schermen rode pixels gevonden zijn, zal de Autopilot stoppen met draaien en zijn positie tegenover het doel berekenen.

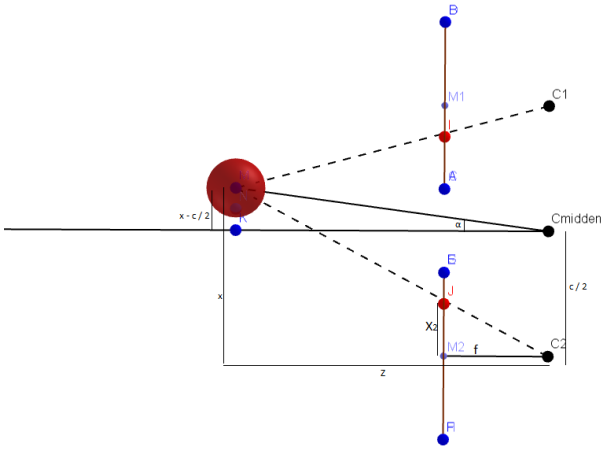
Om zijn positie tegenover het doel te berekenen, bepalen we eerste de diepte. Dit kan met behulp van de formule van stereo vision [**?**] uitgewerkt worden. Zie figuur 1 voor een grafische weergave van de berekening.



Figuur 1: Diepteberekening tussen Drone en doel.

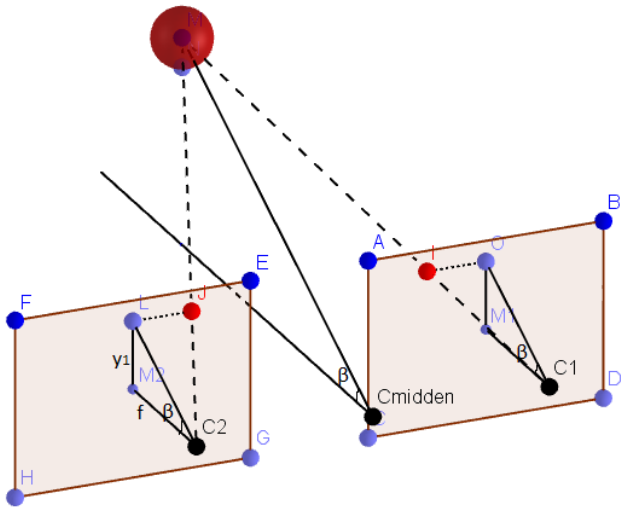
Vervolgens bepalen we de hoek waaronder de drone een yaw beweging moet uitvoeren om recht naar het doel gericht te zijn. Deze formule kon afgeleid worden uit de goniometrie en wordt weer-

gegeven in figuur 2.



Figuur 2: Relatieve horizontale hoek tussen drone en doel.

Om tenslotte naar het doel te kunnen vliegen, moet een evenwicht gevonden worden tussen pitch en thrust. De pitch hoek wordt gekozen zodat het zwaartepunt van het doel nog juist in beeld blijft. Die hoek is gelijk aan de helft van de verticale hoek die het beeld overspant min de verticale hoek waaronder de bol zich tegenover de drone bevindt, zie figuur 3.



Figuur 3: Relatieve verticale hoek tussen drone en doel.

Wanneer de pitch hoek vastligt, kan de hoeveelheid thrust berekend worden zodat de drone in rechte lijn naar het doel kan vliegen. (uitwerking formule)

Tenslotte moet dit proces herhaaldelijk worden uitgevoerd ten gevolge van de invloed van wind. De wind kan de drone namelijk uit koers brengen. Hierdoor zal de drone telkens zijn positie moeten herberekenen en zich opnieuw moeten heroriënteren. Ook kan de wind ervoor zorgen dat de drone

een roll uitvoert. Deze moet eerst gecompenseerd worden, vooraleer we verder onze berekeningen kunnen uitvoeren.
De drone bereikt zijn doel wanneer de Autopilot niks anders dan rode pixels opvangt. De drone zal dan de opdracht krijgen om zijn pitch te compenseren en vervolgens enkel via thrust de zwaartekracht tegen te werken.
Het effectief laten vliegen van de drone gebeurt in de Virtual Testbed waar de motoren worden aangestuurd. De Autopilot zendt enkel de verhouding in graden per seconden door waaronder pitch, yaw en roll moeten worden uitgevoerd en thrust in Newton. Het is slechts door herhaaldelijk te controleren hoe ver nog gedraaid moet worden, dat kan besloten worden wanneer de beweging volledig uitgevoerd is en wanneer er gestopt mag worden.

Voor de volgende deadline is het de bedoeling om de beeldverwerking via *OpenCV* uit te voeren. Deze bibliotheek kan op een betere en gemakkelijkere manier vormen en kleuren herkennen. Dit speelt in ons voordeel wanneer er meerdere doelen met verschillende vormen en kleuren in beeld zijn. Hierdoor zal zowel het iteratief zoeken als het oplsaan van de pixels vermeden worden. Aangezien *OpenCV* op een bepaalde range in RGB waarden kan zoeken, kan ook het probleem van lichtinval en schaduw opgelost worden. Ook wordt er gezocht naar een betere manier om de roll hoek in de berekeningen te trekken. Zo zal er tijd gewonnen worden aangezien die beweging niet altijd eerst moet gecompenseerd worden voordat de berekeningen kunnen worden uitgevoerd.

2.2 Virtual Testbed

Auteur:

3 Algoritmes

Auteurs: ; redactie: Arne Vlietinck

3.1 Drone Autopilot

Auteurs:

3.2 Virtual Testbed

Auteurs:

4 Software

Auteurs: ; redactie: Arne Vlietinck

4.1 Drone Autopilot

Auteurs: Laura Vranken
Tijdens het programmeren, is het belangrijk om een duidelijke structuur voor ogen te houden. Dit leidt immers tot goede leesbaarheid voor buitenstaanders en bewaart overzichtelijkheid bij lange code. Ook werd er geprogrammeerd met het doel om later makkelijker delen te kunnen hergebruiken, mits enkele aanpassingen. Om deze tekst beter te kunnen volgen, staat (in bijlage: Klassendiagram Autopilot) een afbeelding die dit visueler maakt.

Het deel van de Autopilot begint bij het creëren van een Autopilot in de DroneAutopilotFactory class, geimplementeerd met de interface AutopilotFactory. Hierin wordt een Autopilot aangemaakt en worden de beginwaarden voor yaw, roll, pitch en thrust direct ingesteld.

De Autopilot wordt aangemaakt door middel van een nieuw object DroneAutopilot, geimplementeerd met de interface Autopilot, aan te roepen. De DroneAutopilot class bestaat uit één functie, namelijk timeHasPassed die herhaardelijk door de simulator opgeroepen wordt. Vanuit deze methode wordt de class MoveToTarget aangeroepen met de opdracht naar de rode bol te vliegen.

De class MoveToTarget bepaalt dan de bewegingen en algehele aansturing van de Drone en zorgt ervoor dat de rates doorgegeven worden aan de simulator. MoveToTarget steunt zowel op de informatie van de class ImageCalculations als van PhysicsCalculations om zijn beweging te bepalen. Bovendien wordt hierin ook telkens de GUI geüpdatet wanneer een nieuwe waarde voor de diepte bepaald is.

ImageCalculations is verantwoordelijk voor alles omtrent de beelden die de Autopilot binnen krijgt, m.a.w. de rode pixels zoeken en het zwaartepunt bepalen. In de class PhysicsCalculations, worden tenslotte alle fysische berekening van hoeken en afstanden uitgevoerd.

4.2 Virtual Testbed

Auteurs:

5 GUI

Auteurs: ; redactie: Arne Vlietinck

5.1 Drone Autopilot

Auteur: Matthias Van der Heyden
De GUI van de drone autopilot heeft tot nu toe twee functies: de gebruiker de mogelijk geven een opdracht voor de drone te selecteren en de voortgang van de voltooiing van deze opdracht weergeven. Voor de layout van de GUI wordt Java's Abstract Window Toolkit (java.awt) gebruikt.

Voor het selecteren van een opdracht is er een drop-down menu voorzien dat gebruikt maakt van JComboBox uit de Swing library van Java. Wanneer de gebruiker een optie aanduidt, verandert de boolean van deze optie naar true. Dat zorgt er voor dat in de timeHasPassed functie de commando's voor deze opdracht uitgevoerd worden. Op dit moment is de enige optie in het menu de opdracht om naar de rode bol te vliegen maar een uitbreiding van mogelijke opdrachten is relatief eenvoudig.

Een JProgressBar uit de Swing library van Java geeft in de GUI de voltooiing van de geselecteerde opdracht weer. Bij elke berekening van de afstand tot het doel wordt deze geüpdatet. Is de afstand groter dan de laatste grootste afstand tot het doel dan stelt de progress bar deze nieuwe afstand in als het nieuwe maximum en is de voltooiingsgraad weer 0. Is de afstand kleiner dan is de nieuwe voltooiingsgraad gelijk aan 100 procent vermindert met de verhouding van de grootste afstand en de huidige afstand.

5.2 Virtual Testbed

Auteurs:

6 Testen

Auteurs: ; redactie: Arne Vlietinck

6.1 Drone Autopilot

Auteurs: Vincent Vliegen

De autopilot is voorzien van enkele JUnit testclasses. Deze testen de verschillende gebruikte methods op hun nauwkeurigheid en correctheid. Zo is het eenvoudiger de capaciteiten van de autopilot aan te tonen en de beperkingen duidelijk af te bakenen.

De ImageCalculationsTest is uitgerust met tests voor elke method in de ImageCalculations class. Er is gebruik gemaakt van een anonieme class die de Camera interface implementeert, opdat er afbeeldingen naar keuze gegenereerd kunnen worden, die het testen praktisch vereenvoudigen.

De PhysicsCalculationsTest verschaft testmethodes voor de PhysicsCalculations class. Ook hier zijn anonieme classes geïmplementeerd, namelijk voor de Camera en Drone interfaces.

6.2 Virtual Testbed

Auteurs:

7 ...

8 Besluit

Auteurs: ; redactie: Arne Vlietinck

Referenties

De volgende informatie wordt na de finale demonstratie apart ingediend.

A Beschrijving van het proces

- Welke moeilijkheden heb je ondervonden tijdens de uitwerking?
- Welke lessen heb je getrokken uit de manier waarop je het project hebt aangepakt?
- Hoe verliep het werken in team? Op welke manier werd de teamcoördinatie en planning aangepakt?

B Beschrijving van de werkverdeling

- Geef voor elk van de groepsleden aan aan welke delen ze hebben meegewerkt en welke andere taken ze op zich hebben genomen.
- Rapporteer in tabelvorm hoeveel uur elk groepslid elke week aan het project gewerkt heeft, zowel tijdens als buiten de begeleide sessies. Geef ook totalen per groepslid voor het volledige semester.

C Kritische analyse

- Maak een analyse van de sterke en zwakke punten van het project. Welke punten zijn vatbaar voor verbetering. Wat zou je, met je huidige kennis, anders aangepakt hebben?