Touwbruggen gemaakt door een Drone

Tom Carpay (10508368)
Tyler Cools (11004851)
Mara Fennema (11020164)
Kyle Molenaar (10779027)
Tessa Wagenaar (11021047)

24-06-2016

Inleiding

In de laatste week van het vak Zoeken, Sturen en Bewegen zal er een drone worden geprogrammeerd om kabels te spannen, en daartussen een soort brug te maken. De gebruikte software hiervoor is *node.js* op een Mac met OS X El Capitan 10.11.2. De gekozen quadcopter-drone is een *Parrot rolling-spider*.

De hoofdvraag luidt als volgt: Kan een quadcopter-drone geprogrammeerd worden zodat deze volledig zelfstandig een primitieve touwbrug kan bouwen?

Om de hoofdvraag op een zo efficiënt mogelijke manier te beantwoorden, zullen er deelvragen worden opgesteld.

1. Hoe kan de drone geprogrammeerd worden zodat deze vanaf een laptop bestuurd kan worden?
2. Hoe kan de drone geprogrammeerd worden zodat hij zelfstandig een route kan vliegen?
3. Kan er met deze mate van zelfstandigheid een vliegpatroon gecreëerd worden zodat de drone een primitieve brug kan bouwen?
4. Kan de camera van de drone worden gebruikt om een pad te volgen?

Als eerste stap zal gekeken worden naar de besturing van de drone. Deze zal geprogrammeerd worden zodat deze vanaf een laptop volledig te besturen is. Hiervoor zal gekeken worden naar software die al toegankelijk is vanaf het internet en hoe zulke software zelf ontwikkeld zou kunnen worden.

Wanneer de drone volledig bestuurbaar is vanaf een laptop, kan naar geavanceerdere taken worden gekeken om de drone zelfstandig te laten vliegen. Hierbij kan de zelf geprogrammeerde software gebruikt worden om een pad uit te stippelen die de drone zal gaan volgen.

Als de drone in staat is om zelfstandig te vliegen, zal als laatste stap worden onderzocht hoe de drone geprogrammeerd kan worden dat deze volledig zelfstandig een primitieve touwbrug kan bouwen. Hiervoor kan de route gebruikt worden die bij de vorige deelvraag is gecodeerd.

Als er toegang tot de camera kan worden verkregen, zal worden gekeken of de drone een afgebakend pad kan volgen. Het volgen van een pad kan van toepassing zijn op het volgen van een kabel. Hierdoor kan de drone extra accuraat zijn in het maken van de touwbrug.

Voortgangsreportage

Maandag 20 juni 2016

Op de eerste dag van het project moest onderzocht worden doe de *Parrot Rolling Spider* vanaf de computer bestuurd zou kunnen worden. Na vele uren zoeken en proberen is hier echter geen oplossing voor gevonden. Doordat deze drone dit jaar voor het eerst gebruikt werd, kon Arnoud helaas ook geen oplossing vinden om de drone werkend te krijgen. Er is onderzocht hoe een eigen *SDK class* gemaakt kon worden voor Linux.

Dinsdag 21 juni 2016

Door de problemen die maandag ondervonden werden, werd er besloten om op dinsdag twee dingen te proberen. Er werd verder onderzocht hoe de *Parrot Rolling Spider* vanaf de computer geprogrammerd zou kunnen worden en ondertussen werd er ook gekeken of de *iRobot Roomba 500 series* geprogrammeerd zou kunnen worden. Deze robot was meegenomen vanuit huis door Mara. Dit

bleek echter al snel een doodlopend eind te zijn, aangezien er een snoer nodig was om de robot aan de computer aan te sluiten die zo zeldzaam is dat deze niet op het Science Park te vinden is. Dit snoer werd ook niet meegeleverd met de *Roomba* omdat de robot gekocht werd om te stofzuigen en niet om hem zelf te kunnen programmeren.

Ondertussen werd er een oplossing gevonden voor het probleem met de drone. Uiteindelijk zijn er JavaScript bestanden gevonden waarmee een bluetooth connectie tussen de drone en de computer aangelegd kan worden. Via deze connectie kunnen dan gegevens doorgestuurd worden die de drone verder kan uitvoeren. De code die ervoor zorgt dat de drone verbinding kan maken met de computer heet *node.js*. Dit is een *JavaScript runtime built*. De code die met dit programma geschreven kon worden was in deze stijl:

function TEMPORAL.QUEUE :
    wacht 5 seconden
    stijg op
    vlieg naar voren
end function

Deze eerste versie van deze *node.js* code bevatte echter erg veel fouten. Deze fouten zorgden ervoor dat het commando voor opstijgen wel altijd werkte, maar dat hij ongeveer 70% van de tijd na het opstijgen de verbinding met de drone verbrak waardoor deze in de lucht bleef hangen zonder nieuwe commando's te verkrijgen.

Om deze reden is er gezocht naar een andere versie van de *node.js* code. Toen er een nieuwe versie gevonden was van deze *node.js* code bleek dat deze code een functie *setTimeOut* bevatte die de code erg onoverzichtelijk maakte wanneer er een pauze tussen twee elementen werd ingevoegd. Bovendien vergde deze functie veel meer rekenkracht van de drone dan de *delay* functie van de eerste code.

function SETTIMEOUT(int miliseconds)
    vlieg naar voren op halve snelheid na de gegeven tijd
end function

Om deze redenen is uiteindelijk besloten om het beste uit beide *node.js* codes te halen en hier een combinatie van te maken. Dit resulteerde in een code met de volgende stijl:

function TEMPORAL.QUEUE :
    wacht 5 seconden
    vlieg naar voren op halve snelheid na de gegeven tijd
    draai naar links met een gegeven hoek
end function

Woensdag 26 juni 2016

Nu alle code goed werkte kon er pas echt goed geprogrammeerd worden. Als eerste werd er geprobeerd om de drone in een vierkant te laten vliegen. Toen dit werkte, werd er gekeken of er ook in de hoogte veranderingen aangebracht kunnen worden. Dit resulteerde in een patroon waarbij de drone in het vierkant op de eerste hoek omhoog ging en bij de derde hoek weer omlaag

ging. Toen dit werkte, werd er geprobeerd of dit patroon ook om een touw gevlogen kon worden. Met een paar kleine aanpassingen in de waarden lukte dit ook. Toen dit lukte, is het patroon zo aangepast dat de drone om een denkbeeldige paal die loodrecht kruist met het gespannen touw heen een kruisje vliegt. Dit zou ervoor zorgen dat de denkbeeldige balk aan het gespannen touw vast wordt geknoopt. Toen dit werkte, werd er gekeken of er een stuk touw aan de drone bevestigd zou kunnen worden, zodat het patroon te zien zou zijn wanneer de drone het patroon vliegt om het gespannen touw en de denkbeeldige paal. Een van de mogelijkheden waarom dit niet goed ging, was dat de constructie die nodig was om het touwtje aan de drone te kunnen bevestigen onder de hoogtesensor bleek te zitten. Hierdoor kon de drone het patroon niet meer normaal vliegen, omdat hij dacht dat hij vlak bij de grond was. Een andere mogelijkheid is dat de door het verschil in gewicht, de ingebouwde software de drone niet meer stabiel kon krijgen. Het touwtje kon echter ook niet op een andere plek op de drone bevestigd worden, want dan was de kans te groot dat het touwtje in de propellers zou belanden en dit zou de drone kapot maken.

Figure 1: De drone bij het touw.

Vervolgens is de groep in twee delen gesplitst. Het ene deel (Tom, Tessa en Tyler) van de groep is verder gegaan met de drone zodat deze vanaf een laptop te besturen is. Het andere deel (Mara en Kyle) is begonnen aan de implementatie van de *Motive OptiTrack*, voor het volgen van de locatie van de robot.

De groep met de drone

Voor het op afstand besturen van de drone, is gekeken naar hoe de *keyboard events* geïmplementeerd kunnen worden. De code hiervoor had een stijl zoals deze:

function FUNCTION

```
    if w is ingedrukt then
        vlieg naar voren
    end if
end function
```

De letters worden geïdentificeerd aan de hand van *utf8 codes*. Zo ziet letter 'a' er in *utf8 code* uit als *\u0061*. De code kan dan aan de hand van deze *utf8 codes* zien welke letter er is ingedrukt en door middel van *if-statements* de juiste actie aan de letter koppelen.

De groep met *Motive OptiTrack*

De *Motive OptiTrack* is een syteem dat in het robolab staat. In het robolab bestaat dit uit een set van zes infrarood-camera's, die een specifiek object kunnen vinden. Op basis van de relatieve afstanden en hoeken tussen drie witte bolletjes op de drone kan een computer de exacte locatie van de drone vinden. Vorig jaar is er ook een groep geweest die de *Motive OptiTrack* gebruikt heeft om hun eigen robot te volgen. Eerst is hun website gelezen, en na overleg met Arnoud Visser bleek de computer in het robolab al verbonden te zijn met de camera's. Het stappenplan is vervolgens opgevolgd, en er is begonnen met de kalibratie. Eerst gaf de computer de error dat er minstens twee camera's gelinkt met de computer moesten zijn voor een succesvolle kalibratie, terwijl alle zes de camera's via usb-snoeren met de computer gelinkt waren. Vervolgens is een al bestaande file geopend van de computer, en toen kon hij wel kalibreren. De resultaten van de kalibratie lijken echter niet correct, aangezien deze de impressie geven dat camera's in de grond zitten, in plaats van dat ze in de lucht hangen. Later bleek het tracking systeem niet aan te staan, en de kalibratie bleek ook niet nodig te zijn.
De volgende stap was drie van de wit-grijze bolletjes aan de drone vastmaken. Hiervoor is gekozen voor het object in figure 2 en 3, want deze past op de drone. Vervolgens is een korte test uitgevoerd om te kijken of de drone wel kon vliegen met dit object eraan vast gemaakt, wat het geval bleek te zijn.

Figure 2: Het object met de bolletjes.

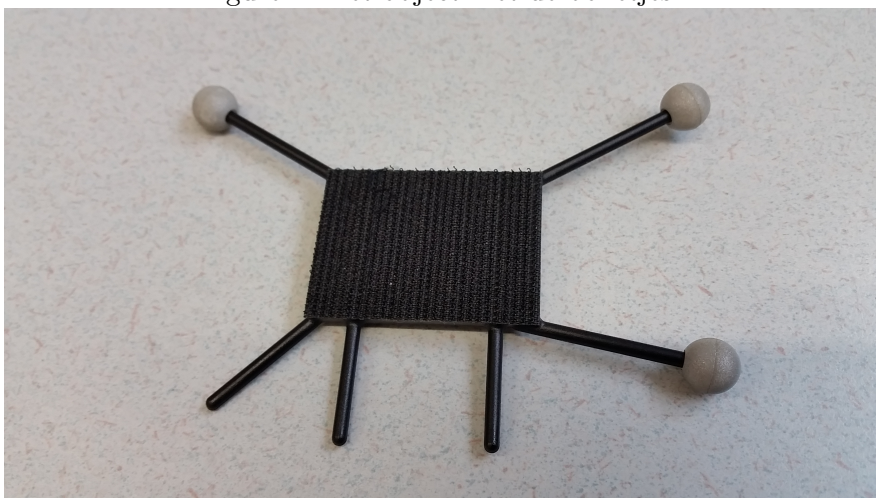

Figure 3: De drone met de bolletjes.

Donderdag 23 juni 2016

De groep met *Motive OptiTrack*

Er is verder gegaan met het project om de *Motive Optitrack* te laten werken, door gebruik te maken van het 3D-beeld op de computer. Hoewel elke camera los de bolletjes wel zag, was het niet mogelijk ze in het 3D-beeld te krijgen. Dus toen is opnieuw aan de kalibratie begonnen.

Het kalibreren bracht veel obstakels met zich mee. Ten eerste duurde het zeer lang voordat ontdekt werd dat de constructie die gebruikt werd als *OptiWand*, niet de correcte constructie was. Hoewel de stok wel klopte, was het opzetstuk de oorsprong, en niet de echte *OptiWand*. Nadat deze gevonden was ging het kalibreren veel beter.

De volgende stap was de oorsprong, en dit is waar alles fout ging. Hoewel de het computerprogramma nu wel bolletjes kon plaatsen in het 3D-veld, moest er ook nog een grond gedefinieerd zijn. Als de oorsprong echter op de grond geplaatst werd, waren de bolletjes niet te zien op de computer. Na veel gepruts is het uiteindelijk gelukt om de drie bolletjes in het 3D beeld te krijgen. Toen moesten deze drie bolletjes geselecteerd worden, en op de knop drukken om hier het grondvlak van te maken. Hoewel alle drie de bolletjes geselecteerd waren, liet de computer dezelfde error zien. Ook het internet zei dat het grondvlak gedefinieerd moest worden op de manier zodat dat hier gedaan is. Toen dit dus niet werkte is dit onderdeel van het project opgegeven, want de tijd die er nog over was om aan het project te werken was niet lang genoeg om het *OptiTrack* systeem te laten werken.

De groep met de drone

Om de functies van de drone uit te breiden, is er onderzoek gedaan naar de functionaliteiten van de camera van de drone. De drone heeft een camera en gebruikt deze ook actief. Wanneer er een bewegend object is op de grond, dan volgt de drone dit object. Er is code gevonden om toegang te krijgen tot de camera, maar toen deze niet werkte, bleek dat deze code niet geschikt was voor onze drone. Toegang krijgen tot de camera is dus uiteindelijk niet gelukt.

Verder is er een interface gemaakt voor de besturing van de drone met het toetsenbord. Hiervoor is een scherm ontwikkeld in ascii code waarin alle mogelijke functies van de drone beschreven staan en met welke letters deze functies aangeroepen kunnen worden.

Discussie

Uit de resultaten is te concluderen dat de gebruikte drone niet helemaal voldeed aan de eisen die in eerste instantie gesteld waren. Zo was deze drone niet sterk genoeg om een touwtje op veilige constructie te dragen. Deze veilige constructie was nodig, omdat het touwtje anders in de propellers van de drone terecht zouden kunnen komen, wat niet de bedoeling is. Hierdoor was het niet mogelijk om een om de drone een touwbrug te laten bouwen.

Ook is er te concluderen dat de drone te besturen is vanaf de computer, je deze ook een voorgeprogrammeerde route kan laten vliegen, maar dat er geen enkele mogelijkheid is om enige feedback terug te krijgen van de drone (zoals informatie van de camera).

Interessante vervolgstappen zouden zijn om dit zelfde plan uit te voeren met een drone waar wel feedback van verkregen van kan worden. Ook zou het interessant zijn om meerder drones samen een brug te laten bouwen (op een wijze van een *swarm*, want dan kunnen betere knopen gelegd worden, en worden er meerder touwen gebruikt.

Door het doen van dit project is veel geleerd. Ten eerste is er geleerd dat wanneer het tracking systeem in het robolab gebruikt wil worden, men hier meer dan twee dagen van te voren mee moet beginnen. Het vergt veel geduld en moeite, dus bij grote projecten zoals dat moet ook veel tijd ingepland worden.

Ten tweede is er geleerd dat zelfs wanneer een project tegen zit, er doorgezet moet worden. Op de eerste dag is er weinig bereikt, en zat alles tegen, maar op de tweede dag is de drone toch aan de praat gekregen. Hoewel dit niet zonder obstakels ging, is het uiteindelijk toch gelukt om de drone te besturen, en als het opgegeven was, was het eindproduct er niet geweest.

Appendix

De functie van de drone dat er een patroon gevlogen wordt om een kabel heen voor het maken van een touwbrug.

```
var RollingSpider = require("parrot-rolling-spider");
var temporal = require('temporal');
var yourDrone = new RollingSpider();

var distX = 35; // needs to be dividable by 2
var distY = 20;
var height = 35;
var heightCorrect = 10;
var delayBetween = 1500;

yourDrone.connect(function () {

   temporal.queue([
      {
        delay: 1000,
        task: function () {
          yourDrone.trim();
          yourDrone.takeOff();
        }
      },
      {
        delay: delayBetween, // HOVER
        task: function () {
```

```
      yourDrone.hover();
    }
  },
  /* ------- BEGIN OF RUN -------*/
    {
    delay: delayBetween, // DOWN - to correct for takeoff
    task: function () {
      yourDrone.down(heightCorrect);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
  {
    delay: delayBetween, // FORWARD - times two to correct for drift
    task: function () {
      yourDrone.forward(distX+distX/2);
    }
  },
  {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
    {
    delay: delayBetween, // UP - to wrap around
    task: function () {
      yourDrone.up(height);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
     {
    delay: delayBetween, // RIGHT
    task: function () {
      yourDrone.right(distY);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
  {
    delay: delayBetween, // BACKWARDS
```

```
    task: function () {
      yourDrone.backward(distX);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
  {
    delay: delayBetween, // DOWN - to wrap around
    task: function () {
      yourDrone.down(height);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
   {
     delay: delayBetween, // FORWARDS
    task: function () {
      yourDrone.forward(distX);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
    {
    delay: delayBetween, // UP - to wrap around
    task: function () {
      yourDrone.up(height);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
    {
    delay: delayBetween, // LEFT
    task: function () {
      yourDrone.left(distY);
    }
  },
    {
    delay: delayBetween, // HOVER
```

```
    task: function () {
      yourDrone.hover();
    }
  },
  {
    delay: delayBetween, // BACKWARDS
    task: function () {
      yourDrone.backward(distX);
    }
  },
  {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
    {
    delay: delayBetween, // DOWN - to wrap around
    task: function () {
      yourDrone.down(height);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
    {
     delay: delayBetween, // FORWARDS
    task: function () {
      yourDrone.forward(distX);
    }
  },
    {
    delay: delayBetween, // HOVER
    task: function () {
      yourDrone.hover();
    }
  },
  /* ------- END OF RUN -------*/
  {
    delay: delayBetween,
    task: function () {
      yourDrone.land();
    }
  },
  {
    delay: 2000,
    task: function () {
      temporal.clear();
      process.exit(0);
    }
  }
```

```
    ]);

});
```

De functie voor de noodlanding, mocht alles misgaan, dan kan deze code aangeroepen worden voor het maken van een noodlanding.

```
var RollingSpider = require("parrot-rolling-spider");

var yourDrone = new RollingSpider();

yourDrone.connect(function () {

   yourDrone.emergencyLand();

});
```

De functie voor het besturen van de drone met de laptop. De interface vertelt hoe de drone bestuurd kan worden

```
var RollingSpider = require("parrot-rolling-spider");
var temporal = require('temporal');
var yourDrone = new RollingSpider();


var dist = 15;
var height = 25;
var delayBetween = 1500;

var stdin = process.stdin;
stdin.setRawMode(true);
stdin.resume();
stdin.setEncoding('utf8');
var beginGame =
"+------------------------------------------------------------+\n\
|                                                            |\n\
|                   WELCOME TO OUR DRONE APP                 |\n\
|               FEEL FREE TO MAKE A FLIGHT WITH IT           |\n\
|                                                            |\n\
|                   KEYS TO CONTROL THE DRONE                |\n\
|                                                            |\n\
|  w -- forwards     q-- turn left    z-- up       v-- take off|\n\
|  s -- backwards    e-- turn right   x-- down     b-- land    |\n\
|  a -- left         f-- hover        g -- frontflip          |\n\
|  d -- right                         h -- backflip           |\n\
|                                                            |\n\
|                      n-- emergency land                    |\n\
|                      p-- quit                              |\n\
|                                                            |\n\
+--------------------+--------------+------------------------+\n\
|                  This appliction was provided by           |\n\
|                                                            |\n\
|          Tom Carpay, Tyler Cools, Mara Fennema, Kyle Molenaar, |\n\
|                            Tessa Wagenaar                  |\n\
```

```
|                                                                    |\n\
+--------------------+--------------+------------------------+";

process.stdout.write(beginGame);

yourDrone.connect(function () {
   stdin.on('data', function(key){
      if (key == '\u0071') {
         process.stdout.write(turnLeft);
         yourDrone.counterClockwise(dist);
         }
       if (key == '\u0065' ) {
           process.stdout.write(turnRight);
           yourDrone.clockwise(dist);
       }
       if (key == '\u0066' ) { //e
           process.stdout.write(hover);
           yourDrone.hover();
       }
       if (key == '\u0061' ) { //a
           process.stdout.write(goingLeft);
           yourDrone.left(dist);
       }
       if (key == '\u0077') { //w
           process.stdout.write(goingForward);
           yourDrone.forward(dist);
       }
       if (key == '\u0073') { //s
           process.stdout.write(goingBackwards);
           yourDrone.backward(dist);
       }
       if (key == '\u0064') { //d
           process.stdout.write(goingRight);
           yourDrone.right(dist);
       }
      if (key == '\u007A') { //q
           process.stdout.write(goingUp);
           yourDrone.up(height);
       }
       if (key == '\u0078') { // t
           process.stdout.write(goingDown);
           yourDrone.down(height);
       }
       if (key == '\u0076') { // v
           process.stdout.write(takeOff);
           yourDrone.trim();
            yourDrone.takeOff();
       }
       if (key == '\u0062') { // x
           process.stdout.write(landing);
           yourDrone.land();
       }
       if (key == '\u0067') { // n
           process.stdout.write(frontFlip);
```

```
            yourDrone.frontFlip();
        }
        if (key == '\u0068') { // z
            process.stdout.write(backFlip);
            yourDrone.backFlip();
       }
        if (key == '\u006E') { // n
            process.stdout.write(emergencyLanding);
            yourDrone.emergencyLand();
        }
        if (key == '\u0070') { //p
          process.exit();
          }
   });
});
```