LAMK Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

Parrot AR.Drone 2.0

Quadrokopterin käyttö ja ohjelmointi

LAHDEN
AMMATTIKORKEAKOULU
Tekniikan ala
Tietotekniikan koulutusohjelma
Tietokone-elektroniikka
Opinnäytetyö
Kevät 2016
Juuso Savolainen

Lahden ammattikorkeakoulu
Tietotekniikan koulutusohjelma

SAVOLAINEN, JUUSO:

Parrot AR.Drone 2.0
Quadrokopterin käyttö ja ohjelmointi

Tietokone-elektroniikan opinnäytetyö, 40 sivua, 4 liitesivua

Kevät 2016

TIIVISTELMÄ

Opinnäytetyön tarkoituksena oli tutkia Parrot AR.Drone 2.0 quadrokopteria. Työssä selvitettiin kopterin toimintaa ja sitä, onko kenen tahansa mahdollista käyttää ja ohjelmoida laitetta. Työn alussa käydään läpi kopterin ominaisuudet, laitteisto ja käyttäminen. Tutkimalla kopterin sisältämää laitteistoa voidaan selvittää sen käyttömahdollisuuksia.

Opinnäytetyössä esitellään muutama kopterin ohjelmointiin tarvittava sovellus ja annetaan niiden asennusohjeet. Tärkein sovelluksista on Node.js ja siihen asennettavat moduulit. Moduuleilla pystytään lisäämään ominaisuuksia ja toimintoja kopterille. Käytettävät moduulit ovat dronekehittäjien tekemiä kirjastoja kopterille.

Opinnäytetyössä tutustuttiin kopterin käskykantaan ja sitä kautta ohjelmointiin. Työ sisältää myös ohjelmoinnin käytännön esimerkkejä, joiden avulla kuka tahansa voi ohjelmoida opinnäytetyössä käytettyä kopteria. Käytettävät esimerkit ovat kopterin automaattinen liike, objektin seuraaminen sekä videon lähettäminen selaimeen. Lopuksi käytännön esimerkit yhdistetään yhtenäiseksi kokonaisuudeksi.

Asiasanat: Drone, JavaScript, Node.js, Ohjelmointi, Quadrokopteri

Lahti University of Applied Sciences
Degree Programme in Information Technology

SAVOLAINEN, JUUSO:

Parrot AR.Drone 2.0
Quadcopter usage and programming

Bachelor's Thesis in computer electronics, 40 pages, 4 pages of appendices

Spring 2016

ABSTRACT

The purpose of this thesis was to investigate the Parrot AR.Drone 2.0 quadrcopter. The purpose was to study how the drone operates and if it is possible for anyone to use and program the drone. First the drone's features, hardware and the usage of the drone were studied. By investigating the drone's hardware it is possible to find different ways of using it.

The thesis introduces a few applications that are needed in the programming process. The installation instructions for these applications are given as well. The most important of these applications is Node.js and its modules. With the modules the user can add features and functions to the drone. The modules that are used in this thesis are programming libraries created by the drone developers.

The thesis investigates programming through the drone's instruction set. The thesis also includes practical examples so anyone can program the drone used in the thesis. These examples are the automatic movement of the drone, following an object and video stream to a web browser. The examples are then combined into one consistent whole.

Key words: drone, JavaScript, Node.js, programming, quadrcopter

SISÄLLYS

1 JOHDANTO

Opinnäytetyön tarkoituksena on tutkia Parrot AR.Drone 2.0:aa. Dronen ja UAV:n (Unmanned Aerial Vehicle) määritelmä on ilmassa, maalla tai vedessä liikkuva miehittämätön alus, jota ohjataan langattomasti tai laitteen omalla tekoälyllä. Opinnäytetyössä käytetty drone on ilmassa liikkuva quadrokopteri, joka on hyvin yleiskäyttöinen ja kuluttajille suunnattu malli.

Tutkimus käsittää kopterin käyttämisen, laitteiston, ominaisuudet ja ohjelmoinnin. Tavoitteena on tehdä kopterille käyttöohje, jossa tutkitaan kopterin laitteistoa, ominaisuuksia ja mahdollisia lisälaitteita. Laitteiston ja ominaisuuksien tutkimisen pohjalta selvitetään kopterin toimintaperiaatetta ja sitä, mitä kaikkea kopterilla on mahdollista tehdä, kuten ohjelmointi-mahdollisuudet.

Ensiksi on tarkoitus tutkia, mikä on drone ja milloin niitä on alettu käyttämään. Lisäksi pyritään selvittämään dronejen käyttösovelluksia nykypäivänä. Tämän jälkeen tutkitaan opinnäytetyöhön valittua kopteria. Työssä pyritään selvittämään Parrot AR.Drone 2.0:n laitteisto ja toiminnot, jotka vaikuttavat kopterin lentämiseen ja käyttäytymiseen. Tutkimalla kopteria pystytään miettimään sen käyttömahdollisuuksia kuluttajalle.

Kopterin tutkimisen jälkeen on tarkoitus tutkia kopterin ohjelmointi-mahdollisuuksia. Työssä tutkitaan, mitä sovelluksia kopterin ohjelmointiin on saatavilla, ja perehdytään niihin. Ohjelmointisovelluksiin perehtymisen yhteydessä niille annetaan tarvittaessa asennusohjeet. Ohjelmointiin perehtymisellä tutkitaan, onko kopterille mahdollista luoda tekoälyä tai automaattista liikettä ja voisiko kuka tahansa ohjelmoida kopteria. Opinnäytetyössä käytettävää kopteria pitäisi pystyä ohjelmoimaan, silloin tarvitaan käskykanta ja kirjasto kopterin ohjelmointiin. Työssä tutkitaan saatavilla olevia kirjastoja ja kopterin käskykantoja, joita käytetään myöhemmin esimerkkiohjelmien tekemiseen. Esimerkeistä tulisi lopuksi muodostua yksi kokonaisuus, jolloin kopterille on määritetty suoritettavia toimintoja.

2 UAV HISTORIA JA KÄYTTÖTARKOITUS

Tässä luvussa käsitellään UAV-historiaa ja kehitystä UAV:n niiden alkuajoista lähtien. Tarkoituksena on saada lukija ymmärtämään UAV:n alkuperäinen käyttötarkoitus ja sen jälkeiset kehitysvaiheet. Luvun lopussa esitellään nykypäivän käyttötarkoituksia.

2.1 Miehittämättömien aluksien historiaa ja kehitysvaiheita

UAV:n historia on läheisesti sidoksissa sotateollisuuteen jo vuodesta 1849 lähtien. Ennen 1900-luvun puoliväliä laitteet tunnettiin paremmin nimellä UAV, joka tarkoittaa miehittämätöntä lentävää laitetta. Ensimmäisiä miehittämättömiä lentoja toteutettiin jo vuonna 1849, jolloin välineenä oli kuumailmapalloja lastattuna räjähteillä. Niistä ruvettiin käyttämään nimitystä Itävallan pallot, kun Itävalta hyökkäsi Venetsiaan. Myöhemmin samankaltaisia kuumailmapalloja käytettiin Japanin armeijan toimesta vuonna 1945 Fu-Gon pommituksissa. (Nesta 2016.)

Isona käänteenä UAV-kehitykselle voidaan pitää Nikola Teslan keksimää kauko-ohjainta vuonna 1898, joka oli ensimmäinen laatuaan. Teslalle myönnettiin samana vuonna radiotaajuksilla toimivasta kauko-ohjaimesta Yhdysvaltojen-patentti ja keksinnön nimeksi annettiin Teleautomaton. Keksintöään Tesla esitteli julkisesti samana vuonna veneen ja ison vesitankin avulla Madison Square Gardenissa, New Yorkissa. Vene oli kooltaan noin yhden metrin, ja sen perässä oli moottori. Kauko-ohjaimella ohjattiin veneessä olevaa moottoria päälle ja pois päältä, mutta sillä ei pystytty tekemään käännöksiä. (VanDomelen 2012.)

1900-luvun alusta lähtien UAV:n kehitys jatkui nopeasti toisen maailmansodan johdosta, kun Englannin kuninkaallinen laivasto tarvitsi ilmapuolustukselleen harjoituskohteita vuonna 1930. Tarkoituksena oli luoda kauko-ohjattava oikeaa lentokonetta vastaava UAV. Vuosien 1930 ja 1940 välisenä aikana ratkaisuksi syntyi De Havilland Queen Bee lennokki, joita valmistettiin nykyisten tietojen mukaan yli 400 kappaletta tähtäysharjoituksiin. Ulkonäöltään Queen Bee muistutti keskikokoista

kaksitasokonetta. Myös Yhdysvallat oli kiinnostunut hankkeesta ja kehitti vastaavanlaisen kauko-ohjattavan lentokoneen omiin tykistöharjoituksiin 1930-luvun loppupuolella. (Cole 2014.)

Toisen maailmansodan jälkeen Yhdysvallat alkoi kehittämään ja jalostamaan lennokkeja sotateollisuuteen. Tuolloin ajatuksena oli luoda lennokkeja, jotka pystyisivät taistelemaan ilmassa ja maassa liikkuvia kohteita vastaan. Uusien lennokkien kehityksestä pääsi vastuuseen yhdysvaltalainen yhtiö Teledyne-Ryan, joka voitti sopimuksen itselleen kehittämällään Firebee UAV:llä. Luvulta 1960 lähtien Firebeetä käytettiin tähtäysharjoituksiin, mistä se kehittyi monen vaiheen jälkeen tärkeäksi osaksi tiedustelua ja tiedonkeruuta. Firebeen kehittyneempi versio nimettiin myöhemmin Lightening Bugiksi. Lightening Bugin tärkeys korostui Vietnamin ja Etelä-Aasian sodassa, kun se otettiin käyttöön 1960-luvun puolivälissä. Lightening Bugia käytettiin 1970-luvun alkupuolelle asti. Vietnamin sodan lopussa vuonna 1973 Yhdysvallat luovutti 33 kappaletta lennokkeja Israeliin tiedustelu- ja tiedonkeruutehtäviin Yom Kippur -sotaan. (Cole 2014.)

Israel kiinnostui lennokkien uuden sukupolven kehityksestä sotien jälkeen, kun Yhdysvallat menetti mielenkiinnon tiedustelu- ja tiedonkeruulennokkien kehittämiseen. Lennokkien sijaan Yhdysvallat keskittyi sateliittien ja korkeresoluutioisen kuvan kehitykseen. Sillä aikaa Israel otti vahvan johtoaseman lennokkien kehityksessä ja loi lukuisia eri tiedustelu- ja tiedonkeruulennokkeja. Myöhemmin Israel myi Yhdysvaltoihin Pentagonille kehittämänsä Pioneer-mallin, jonka erikoisuutena oli sen laukaistavuus. Pioneerin pystyi laukaisemaan lentotukialukselta tai sotilastukikohdan kiitoradalta. Kooltaan Pioneer oli pienentynyt edeltäjiinsä nähden ja sitä oli helpompi käsitellä. Pian sen jälkeen kun Yhdysvallat oli saanut omat kappaleensa Pioneereista, niitä alettiin käyttämään ensimmäisessä Persianlahden sodassa. Persianlahden sodassa Pioneereilla suoritettiin yli 300 lentotehtävää. (Cole 2014.)

Suuria muutoksia lennokkien kehityksessä tapahtui Abraham Karemin myötä, jota pidetään ensimmäisen aseistetun lennokin luojana. Karem työskenteli Israelin armeijassa lentokoneinsinöörinä, mutta vuonna 1974 hän erosi ja perusti oman yrityksen. Yritys ei kuitenkaan menestynyt Israelissa, ja Karem muutti Yhdysvaltoihin, missä hän kehitti autotallissaan uudenlaisen lennokin prototyypin. Vuonna 1985 Karem esitteli lennokin rahoittajalleen DARPA:lle, joka solmi sopimuksen lennokkien kehityksestä. Prototyypistä kehitettiin parempi versio ja se nimettiin Amber UAV:ksi, mutta jonkin ajan kuluttua rahoitus kehitykseen lakkasi budjettileikkausten takia. Karem kehitti Amber UAV:stä uudenlaisen ja yksinkertaisemman version, joka nimettiin Gnat 750:ksi. Vuonna 1990 Karemin yritys yhdistyi General Atomicsiin. (Cole 2014.)

Vuonna 1993 Pentagon halusi ottaa käyttöön valvontaan tarkoitettuja lennokkeja entiseen Jugoslaviaan ja tehtävään valittiin Gnat 750. Vuotta myöhemmin CIA käytti Gnat 750 lennokkia tiedustelussaan Balkanilla. General Atomicsin omistuksessa oleva Gnat 750 osoittautui hyväksi ja uusi sotilaskäyttöinen Predator lennokki pohjautuu Gnat 750-malliin. Predator-malli otettiin käyttöön vuonna 1994 ja sen erikoisuutena oli viestintäkyky sateliittiin. Vuonna 1999 Predator-malliin lisättiin laseri, jolla pystyttiin maalaamaan iskukohde, jolloin toinen aseistettu lentokone pystyi iskemään tarkasti kohteeseen. (Cole 2014.)

Predator lennokki oli ensimmäisiä lennokkeja, joka oli raskaasti aseistettu. Aseistamattomat lennot aloitettiin ensimmäisen kerran jo vuonna 2000 Afganistanissa, mutta Predatorin ensimmäiset ilmaiskut suoritettiin Afganistanissa seuraavan vuoden marraskuussa Yhdysvaltojen toimesta. Taustalla oli syyskuussa tapahtunut World Trade Centeriin kohdistunut terrori-isku. (Cole 2014.)

2.2 Käyttösovelluksia

Alun perin lentäviä miehittämättömiä aluksia käytettiin pelkästään sotilaallisissa operaatioissa ja niiden avulla pystyttiin välttämään

suuremmat sotilastappiot. Nykypäivänä lennokkien kehitys on mahdollistanut yhä laajemman käytön myös arkisissa asioissa. Yleisin kuluttaja- ja julkiseen käyttöön suuntautunut lennokkityyppi on quadrokopteri sen helppokäyttöisyyden vuoksi. Helppokäyttöisyys perustuu siihen, että quadrokopterit ovat vakaita lennettäviä ja ne saadaan leijumaan paikallaan. Quadrokopterien suurin ongelma on niiden heikko kyky kantaa esineitä. Quadrokopereiden kantokyky on rajallinen ja suoraan kokoon verrattavissa. Tämän vuoksi on kehitetty ryhmässä toimimista, jolloin useampi quadrokopteri yhdessä samanaikaisesti nostaa esinettä. Quadrokopterien koko ja paino on haluttu pitää sellaisena että niitä on helppo siirtää. (Kumar 2012.)

Quadrokopterit ovat osoittautuneet hyväksi työkaluiksi monilla toimialoilla. Esimerkiksi ympäristöteknologiassa niillä voidaan kartoittaa maastoa ja kuvata metsiä. Niihin voidaan lisätä tiedonkeräystä varten erilaisia antureita, joilla voidaan mitata esimerkiksi ilmanlaatua, lämpötilaa ja kosteutta.

Myös rakennusalalta löytyy käyttökohteita. Kun kartoitetaan esimerkiksi rakennusten kuntoa, kopterilla pääsee hankaliin paikkoihin kuvaamaan. Koptereita voidaan myös lähettää rakennukseen kartoittamaan tiloja ja laitteen lähettämän kuvan avulla voidaan luoda kolmiulotteinen malli rakennuksesta (Kumar 2012). Kopteri voi lähettää kuvaa reaaliajassa esimerkiksi käyttäjän tietokoneelle.

Elokuvateollisuus hyödyntää koptereita ilmakuvauksissa. Nykyajan kamerat ovat niin kevyitä ja korkealaatuista kuvaa ottavia, että ne pystytään asentamaan quadrokoptereihin. Kuluttajakäyttöön löytyy halvempia kopteri-malleja, joilla pystytään harrastamaan ilmakuvaamista. Kopteri voi seurata käyttäjäänsä ja ottaa videota esimerkiksi urheillessa.

Myös poliisit ja muu virkavalta ovat alkaneet hyödyntämää koptereita työssään. Koptereilla pystytään nopeasti nousemaan ilmaan tutkimaan ympäristöä ja jäljittämään vaikka varastettua autoa. Koptereita on

käytössä myös rakennusten ennalta tutkimiseen vaarallisten henkilöiden varalta ennen rynnäkköä tiloihin. (Kumar 2012.)

3 KOPTERIN KÄYTTÖÖNOTTO

Tässä luvussa kerrotaan kopterin ominaisuuksista, laitteistosta ja käytöstä. Laite on helppo ottaa käyttöön. Ensimmäiseksi ladataan puhelimeen tai tablettiin AR.FreeFlight-sovellus käytettävän käyttöjärjestelmän perusteella. Kopterin roottorit on hyvä tarkistaa mahdollisten vaurioiden varalta ennen käyttöönottoa. Roottoreiden välitysten tulisi myös toimia hyvin.

Kun akku on kytketty kopteriin, kopteri kokeilee kaikkia roottoreitaan pienellä nykäyksellä ja sytyttää LED-valon niiden kohdalle palamaan. Valon väri kertoo kopterin tilan. Vihreä tarkoittaa, että kaikki on kunnossa. Punainen tai oranssi voi tarkoittaa jotakin seuraavista: laitteen alustus ei onnistunut täydellisesti, jossakin laitteen roottoreista on vikaa tai kopterin hätätila on mennyt päälle, eli laitteen gyroskooppi on havainnut raja-arvon ylittävän kallistuskulman ja hätäsammuttanut moottorit. FreeFlight-sovelluksen asennuksen jälkeen täytyy ottaa yhteys kopterin luomaan langattomaan yhteyteen ja tarkistaa, että kopterin Firmware eli laiteohjelmisto on ajan tasalla.

3.1 AR.Drone ja paketin sisältö

AR.Drone maksaa Suomessa noin 300 euroa ja hintaan sisältyy laitteen lisäksi suojakuoret sisällä ja ulkona lentämiseen, akku sekä laturi. Sisäkäyttöön tarkoitettu suojakuori suojaa kopterin roottorit tuomalla pienen koteloinnin niiden ympärille, jotta ne eivät osuisi esineisiin tai huonekaluihin. Ulkona lentämiseen tarkoitettu suojakuori ei suojaa roottoreita millään tavalla, mutta se turvaa itse laitteen ytimen kovemmilta iskuilta ja heikolta sateelta. Ulkokäyttöön tarkoitettu suojakuori on huomattavasti kevyempi kuin sisällä lentämiseen tarkoitettu, eikä tuulikaan tartu siihen niin helposti. Kuvassa 1 käytettävissä olevat suojakuoret.

KUVA 1. Ulkona (ylempi) ja sisällä lentämiseen käytettävät suojakuoret

3.2 Ominaisuudet ja laitteisto

Kopterin runko on tehty hiilikuidusta, jotta se olisi mahdollisimman kevyt mutta kuitenkin kestävä. Tasapainon ja vakauden kopteri saa 3-akseliselta gyroskoopilta, 3-akseliselta kiihtyvyysanturilta ja 3-akseliselta magnetometriltä. (Parrot 2016.) Nämä osat vaikuttavat lennettävyyteen olennaisesti.

Kopterin teknologia ja äly perustuu 1 GHz:n ARM Cortex A8 32-bittiseen prosessoriin, johon on lisäksi integroitu 800 MHz:n DSP TMS320DMC64x-prosessori videokuvan käsittelyä varten. Kopterilla on lisäksi 1 GB DDR RAM 200 MHz:n muistia, ja koko laitteen käyttöjärjestelmänä on Linux 2.6.32-versio. (Parrot 2016.)

Lisäksi kopterista löytyy korkealaatuinen 720p resoluutiolla ja 30 kuvaa sekunnissa (30 fps) välittävä videokamera, jolla pystytään näkemään videokuvaa 92 asteen näköalueella. Videokuva välittyy käyttäjälle FreeFlight-sovellukseen, ja tekniikalla on pystytty poistamaan kuvan välittymisestä isoimmat viiveet. (Parrot 2016.) Kameralla pystyy ottamaan lisäksi valokuvia sekä videota lennon aikana, ja kaikki otettu kuvamateriaali tallentuu hallintalaitteeseen.

Kopterin lentokorkeus mitataan ultraäänellä ja ultraäänisensoreilla. Ultraäänen nopeus on vakio, joten etäisyys saadaan laskettua äänisignaalin lähetyksestä ja sen takaisin heijastumiseen kuluneesta ajasta. Nopeuden kopteri mittaa sen pohjassa olevalla vertikaali QVGA-kameralla, joka ottaa 60 kuvaa sekunnissa (60 fps). (Parrot 2016.) Kuvassa 2 näkyy kopterin pohjassa oleva kamera, merkkivalo, ultraäänilähetin ja -sensori.

KUVA 2. Vasemmalta oikealle pääkamera, ultraääni, piirin merkkivalo ja pohjakamera

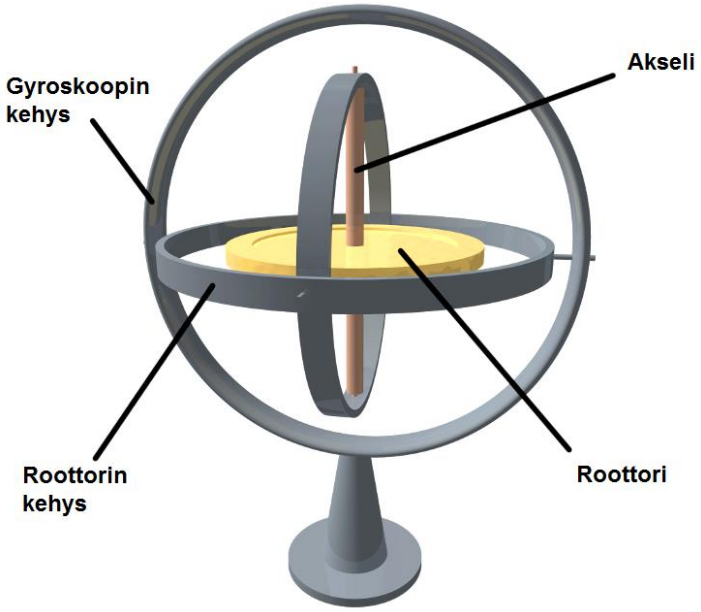
Liikkeet ja nopeuden kopteri saa sen neljältä moottorilta. Moottorit on sijoitettu rungon jokaiselle kulmalle, ja jokaisen moottorin teho on 14,5 wattia. Moottorien lavat pyörivät 28 500 kierrosta minuutissa. Jokaista moottoria ohjaa erillinen mikrokontrolleripiiri, jolla moottorien teho saadaan tasaiseksi. Piiri on vedenkestävä. Moottorivauriot vältetään kopterin hätätilalla, joka katkaisee moottoreilta tehon tarpeen vaatiessa. Voiman välitysrattaat on tehty nailonmuovista, propellien akselit ovat karkaistua terästä ja propellien laakerit ovat itsevoitelevia pronssilaakereita. (Parrot 2016.)

3.2.1 Gyroskooppi

Gyroskooppia käytetään lähes kaikissa kulkuvälineissä, joissa tarvitaan vakautta, tarkkuutta ja suuntaa, esimerkiksi lentokoneet ja laivat. Laitteen rakenne on yksinkertainen. Keskellä on vapaasti pyörivä levy, jota kutsutaan roottoriksi. Roottori on kiinnitetty vapaasti pyörivään akseliin keskelle isompaa vakaata kehystä. Roottorin pyöriessä sillä on kulmanopeus, joka ilmoitetaan kierroksina tai asteina sekunnissa. Gyroskoopin tarkoituksena on mitata tai hallita pyörimisliikettä. Mittaustuloksia saadaan x-, y- ja z-akselien suunnassa, kun mitattavalla kappaleella tapahtuu kiertoliikettä jonkin akselin ympäri. (Sparkfun 2016b.)

Gyroskoopissa on voitu ottaa käyttöön sähkömoottori, jolla roottoria pidetään liikkeessä. Gyroskooppi voi nykyään olla teknologian kehityksen myötä myös pieni elektroninen komponentti. Gyroskoopit voivat olla mekaanisia tai elektronisia. Elektronisissa laitteissa gyroskoopit ovat aina elektronisia, mutta toimintaperiaatte on edelleen sama. (Sparkfun 2016b.)

Tasapainonhallinta kopterille saadaan gyroskoopin mittaustuloksista, jotka gyroskooppi saa kulmanopeuksista (Sparkfun 2016b). Näillä mittaustuloksilla lähetetään korjaussignaaleja kopterin moottoreille. Esimerkiksi jos kopteri lähtee kallistumaan ilman käskyä sen oikealle kyljelle, niin oikean puoleiset moottorit saavat korjauskäskyn lisätä kierroksia tasapainon palauttamiseksi. Kuvassa 3 havainnollistetaan gyroskoopin rakennetta.



KUVA 3. Perinteisen gyroskoopin rakenne (Wikipedia 2006)

3.2.2 Kiihtyvyysanturi

Kiihtyvyysanturi on komponentti, joka mittaa kiihtyvyyttä. Kiihtyvyys saadaan kappaleen nopeuden muutoksista ja ilmoitetaan muodossa m/s^2. Toinen muoto ilmoittaa kiihtyvyys on g-voima. Yksi g-voima on yhtä suuri kuin maan vetovoiman aiheuttama kiihtyvyys eli 9,8 m/s^2. (Sparkfun 2016a.)

Kiihtyvyysanturilla pystytään havaitsemaan staattista ja dynaamista kiihtyvyyttä, joista staattiseen kuuluu maan vetovoima ja dynaamisiin kuuluvat tärinä sekä liikkeet. Kopterissa kiihtyvyysanturia käytetään havaitsemaan tärinää ja laitteen suuntaa, joten kiihtyvyysanturi on malliltaan dynaaminen. (Sparkfun 2016a.)

Toiminnaltaan kiihtyvyysanturi on elektromekaaninen, ja se voi havaita liikkettä kolmella eri akselilla, x-, y- ja z-akselin suunnassa, riippuen siitä, ovatko kaikki akselit käytössä. Nykyään vakiintunut tapa tehdä kiihtyvyysanturi on sijoittaa komponentin sisälle kapasitiivisia levyjä kiinteästi ja pienin jousin kiinni, jotka liikkuvat toisiinsa nähden kiihdyttäessä ja tästä johtuen niiden välinen kapasitanssi muuttuu.

Kapasitanssin muutoksesta voidaan mitata ja laskea kiihtyvyys. (Sparkfun 2016a.)

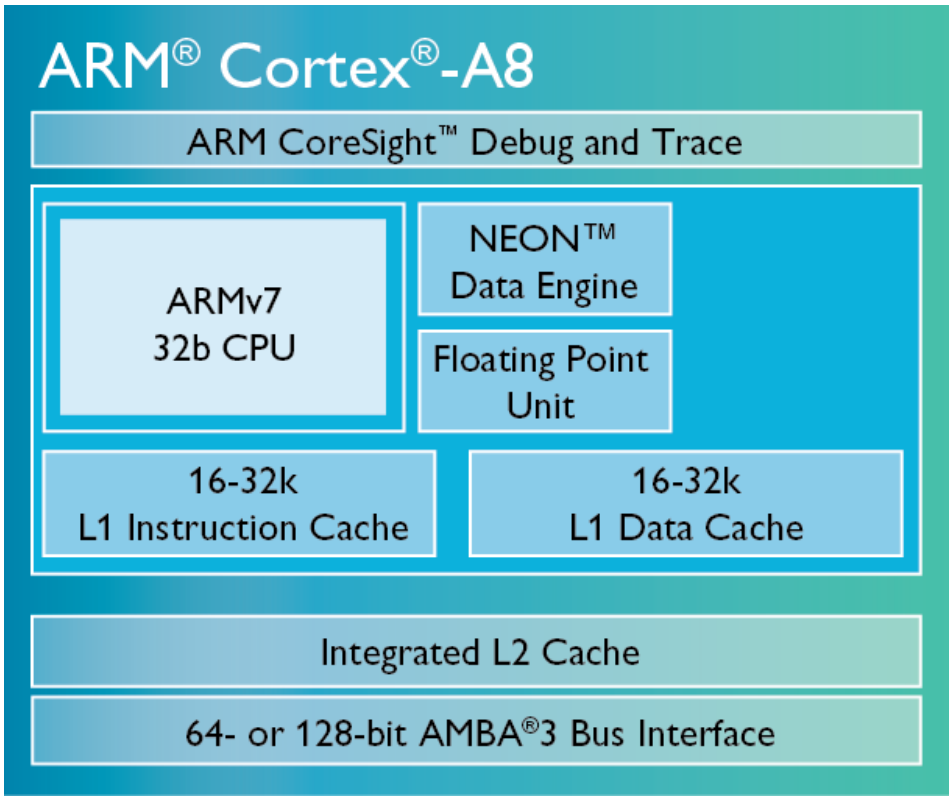
Toinen tapa on käyttää pietsosähköistä materiaalia. Tällä tavalla tehdyssä kiihtyvyysanturissa sen pieni kiderakenne joutuu rasituksen alle ja alkaa tuottamaan jännitettä, josta voidaan mitata ja laskea kiihtyvyys. Kiihtyvyyden vastuksena toimii jousi, joka estää kidettä murskautumasta ja palauttaa anturin stabiiliin tilaan, kun kiihdyttäminen loppuu. (Sparkfun 2016a.)

3.2.3 Magnetometri

Magnetometrillä pystytään mittamaan magneettikenttiä, joista saadaan havaitun magneettivuon tiheys. Kopterissa käytetään 3-akselista magnetometriä, joka tarkoittaa, että siinä on kolme eri vektoria. Magnetometri pystyy mittaamaan magneettivuon tiheyden tiettyyn suuntaan kolmiulotteisesti, ja tämän avulla pystytään määrittämään haluttuja suuntia. (Jain 2016.)

3.2.4 Cortex-A8 Prosessori

Kopterissa käytettävä prosessori on mallia ARM Cortex-A8, ja se julkaistiin ensimmäisen kerran jo vuonna 2005. Se oli tuolloin ensimmäinen ARMv7-arkkitehtuuria käytättävä prosessori. Prosessori sisältää yhden ytimen, jolla on korkea suorituskyky. Yleisimpiä käyttökohteita Cortex-A8:n prosessorille ovat useat kuluttajaelektroniikan käsittävät laitteet, kuten älypuhelimet, tabletit ja printterit. Cortex-A8-prosessoria käytetään usein myös sovelluksissa, joissa tarvitaan korkeaa suorituskykyä ja energiatehokkuutta, kuten web-yhteyden integrointi. ARM Cortex-A8 on rakenteeltaan yksinkertainen, ja sitä selvitetään kuvassa 4. (ARM 2015.)



KUVA 4. ARM Cortex-A8 lohkokaavio (ARM 2016)

3.3 FreeFlight

Kopterin tekijät ja kehittäjät ovat luoneet laitteelle sovelluksen lentämistä varten (kuva 5). AR.FreeFlight 2.4.10-sovellus on ladattavissa iOS- ja Android-käyttöjärjestelmille Applen Appstoresta sekä Googlen Play-kaupasta. Sovellus on täysin ilmainen eikä sisällä sovelluksen sisäisiä ostoksia.

KUVA 5. FreeFlight-sovelluksen näkymä

Piloting-painiketta painamalla pääsee kopterin pilottitilaan, jossa kopteria pääsee lentämään puhelimella tai tabletilla (kuva 6). AR.Drone academy-painikkeella kirjaudutaan sisään AR.Drone omistajien sivuille, johon voi lähettää ottamiaan videoita, kuvia ja lentoreittejä. Samalla AR.Drone academy kerää tietoja lentoajasta, sijainnista, käyttäjistä, lentoon lähdöistä ja nopeudesta. Photos Videos-painikkeella pääsee tarkastelemaan ottamiaan kuvia ja videoita. Kopterin laiteohjelmisto vastaa kopterin perustoimintojen toiminnasta, ja ohjelmiston pystyy päivittämään uusimpaan versioon napsauttamalla AR.Drone update -painiketta. FreeFlight-sovellus ilmoittaa, jos ohjelmisto on jo ajan tasalla. Users videos-painikkeella pääsee nopeasti katsomaan muiden lennättäjien ottamia videoita ja kuvia.

KUVA 6. Pilotti-tilan näkymä

Pilotti-tilassa ruudun alareunassa näkyy takeoff-painike, jota painamalla kopteri lähtee lentoon, ja painalluksen jälkeen tilalle vaihtuu landing-painike, jota painamalla kopteri laskeutuu. Rescue-painiketta painamalla saa lisätoimintoja (kuva 7). Esimerkiksi kopterin irroittamiseen löytyy random shake -painike, jos se on jäänyt kiinni. Over balance-painikkeella haetaan kopterille täydellistä tasapainoa, jos se ei muuten lennä vakaasti. Alareunassa näkyy lisäksi lentokorkeus metreinä ja nopeus kilometreinä tunnissa.

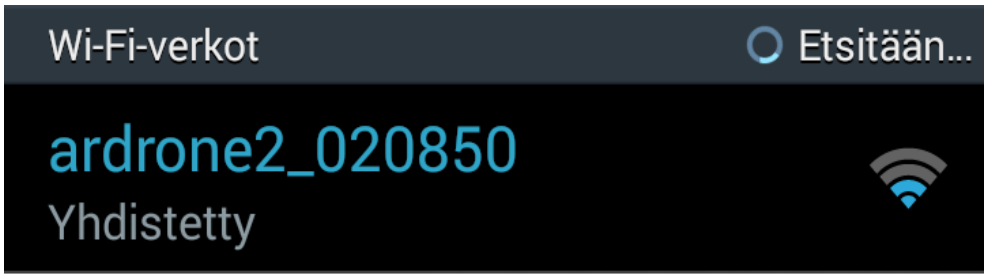
Yläreunassa vasemmalta oikealle näkyy erilaisia kuvioita, joista ensimmäisenä näkyy kopterin akun varaus prosentteina. Toisena on painike, josta pääsee kopterin asetuksiin. Kolmas näyttää senhetkisen langattoman yhteyden tilan pieninä palkkeina ja palkkien määrästä riippuen sen, mikä on yhteyden vahvuus. Neljäntenä on emergency-painike, jolla käyttäjä itse voi aktivoida emergency-tilan, joka johtaa laitteen hätäsammutukseen. Viidentenä on vaihto-painike, jolla pystyy vaihtamaan kopterissa olevien kahden kameran välillä ruudulle välittyvää kuvaa. Kuudentena on videokuvan tallennus-painike, jota painamalla kopteri lähettää videokuvaa ohjauslaitteeseen ja video tallentuu laitteen muistiin. Viimeisenä on valonkuvien ottamiseen tarkoitettu painike, jota painamalla kuvat tallentuvat ohjauslaitteen muistiin.

KUVA 7. Rescue-tila

3.4 Yhteyden luominen

Kopteria kontrolloidaan langattomalla WiFi-yhteydellä. Kopterissa on oma langattoman verkon piiri tätä varten. Laite luo ympärilleen langattoman kentän, johon käyttäjä yhdistää kontrollilaitteensa, kuten kuvassa 8. Tämän jälkeen avataan FreeFlight-sovellus hallintalaitteesta. Langattoman yheyden kantama-alue on noin 50 metriä. Sovelluksen avauduttua asetuksista voi vaihtaa kopterin näkyvää SSID-nimeä. Asetusten muokkaus onnistuu painamalla ominaisuudet-painiketta ruudun yläreunassa pilotti-tilassa.

SSID (Service Set Identifier) on langattoman lähiverkon verkkotunnus. Nimen vaihtaminen helpottaa kopterin tunnistamista muiden langattomien verkkojen joukosta. Kopteri estää useamman kuin yhden laitteen liittymisen sen lähiverkkoon, joten ensimmäiseksi liittynyt laite pysyy sen verkossa. Salasanan lisääminen laitteelle on kuitenkin suositeltavaa. Yhteys käyttää IEEE 802.11 -eli WPA2-tietoturvastandaria, joka on langattomien verkkojen salaukseen käytettävä standardi.



KUVA 8. WiFi-verkkoon yhdistämisen näkymä

3.4.1 GPS Flight Recorder

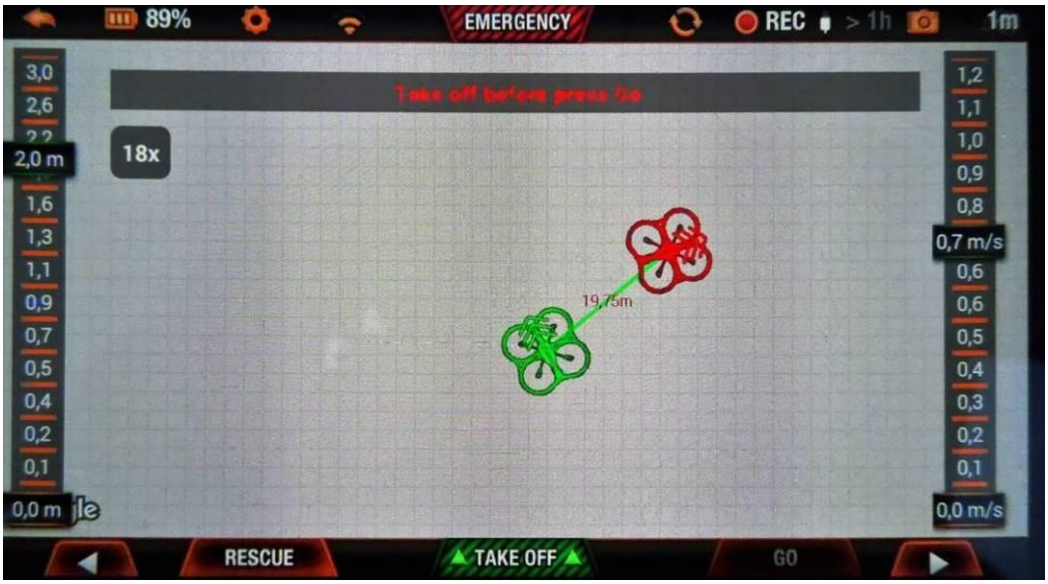
Quadrokopteriin voi ostaa lisälaitteita, kuten GPS:llä toimivan lennon tallennuksen. GPS-lisälaitteen luvataan tuovan lisää vakautta kopterin lentämiseen. Lisälaite myös tallentaa kopterin koko lennon 4 gigatavun tallennustilaan, mikä vastaa noin 350:tä lentoa. (Parrot 2016.) Tallennettua videokuvaa voi tarkastella jälkeenpäin tietokoneella. Kerätyt paikkatiedot voidaan siirtää AR.Drone academyn kartalle, josta lentoreitti nähdään kolmiulotteisesti kohta kohdalta.

GPS:llä voi tehdä ennalta suunniteltuja lentoreittejä syöttämällä paikkatietoja kopterille ennen sen liikkeelle lähtöä. Samalla kun käyttäjä tekee kopterille automaattista lentoreittiä, sille voi myös määrittää lentokorkeuden ja nopeuden sekä lähtöpisteeseen palaamisen. Tietojen syöttäminen tapahtuu FreeFlight-sovelluksessa kun GPS on kytketty kopterin USB 2.0 -väylään kiinni. Pilotti-tilassa lentoreitin suunnitteluun pääsee napauttamalla ruudun ylälaidassa sijaitsevaa viimeistä painiketta. GPS:n tarkkuus vaihtelee yhdestä metristä kymmeneen metriin riippuen signaalin voimakkuudesta, joten absoluuttista tarkkuutta GPS:llä ei pystytä saavuttamaan. Kuvassa 9 esitellään ostettavissa olevaa GPS Flight Recorderia. Jos laite toimii oikein, sen päällä oleva LED-valo alkaa välkkyä.

KUVA 9. Flight Recorder asetetaan akun päälle ja kiinnitetään USB-väylään

Lentoreitin tekeminen tapahtuu raahaamalla kopterin kuvaa ruudulla haluttuun kohteeseen. Nuolet kopterin kuvassa osoittavat kopterin etupään suunnan. Kuvien väliin muodostuu lentoreitin viiva, jonka pituus näkyy metreinä ruudulla. Jos GPS-yhteys on voimakas ja kartta ajantasalla, ruudulla näkyy satelliittikuvaa maastosta. Tämä helpottaa reittisuunnittelua.

Vasemmassa laidassa olevasta liukupalkista pystyy säätämään kopterille lentonopeuden minimi- ja maksiminopeuden raja-arvoja käyttämällä. Oikeassa laidassa on lentokorkeuden määrittämiseen käytettävä liukupalkki, jossa pystytään kahdella raja-arvolla määrittämään kopterille minimi- ja maksimilentokorkeus. Kun kopterille on tehty valmis lentosuunnitelma, tulee ennen liikkeelle lähtöä painaa takeoff-painiketta ja sen jälkeen go-painiketta. Liikkeelle lähdön jälkeen Go-painikkeen vierelle ilmestyy home-painike, jota painamalla kopteri palaa takaisin lähtöpisteeseen. Lentoreitin tekemistä esitellään kuvassa 10.



KUVA 10. Lentoreitin suunnitteluikkuna

4 OHJELMOINNIN TYÖKALUT

Quadrokopterin ohjelmointiin käytettiin opinnäytetyössä neljää eri ohjelmaa, Notepad++:aa, Node.js:ää, CMD:tä ja Ffmpegiä, sekä JavaScript-ohjelmointikieltä ja Node.js:ään asennettavia moduuleita. Kaikki nämä ohjelmat ovat ilmaisia, joko netistä ladattavissa tai valmiiksi tietokoneelle Windowsin mukana asennettu. JavaScriptin kirjoittamiseen löytyy paljon erilaisia harjoituksia ja ohjeita Internetistä. Tietokoneella kopterille ajettavat JavaScriptit tarvitsevat langattoman yhteyden kopterin ja tietokoneen välille.

4.1 CMD

CMD on Windowsin komentokehote, joten sitä ei tarvitse erikseen ladata tai asentaa. Komentokehotteella pystytään suorittamaan ohjelmia, joilla ei ole graafista käyttöliittymää (Wikipedia 2014). CMD:tä tarvitaan opinnäytetyössä skriptien ajamiseen, johon se soveltuu hyvin. Komentokehotteella pystytään debuggaamaan eli testaamaan kirjoitettua JavaScriptiä. Jos skriptissä ilmenee virhe, komentokehote ilmoittaa virheen sijainnin rivinumerona koodissa. Komentokehotetta käytetään opinnäytetyössä myös moduulien asentamiseen.

4.2 Notepad++

Notepad on yksinkertaisesti selitettynä tekstinkäsittelyyn tarkoitettu työkalu ja Notepad++ on perinteistä Notepadia kehittyneempi versio. Notepad++ on ilmainen ohjelma ja luotu lähdekoodin muokkaamista varten. Lähdekoodin muokkaus perustuu Scintillaan, joka on osa Notepad++:aa ja on täysin ilmainen lähdekoodin muokkauskomponentti. Scintillan avulla lähdekoodin muokkaamisesta tehdään tehokkaampaa ja nopeampaa (Scintilla 2016). Ohjelma tukee hyvin useita eri koodauskieliä, ja kopterin ohjelmointiin käytetään JavaScriptejä. Notepad++:n saa ladattua kehittäjän nettisivuilta ilmaiseksi. Latauksen jälkeen avataan asennustiedosto ja valitaan haluttu asennuspaikka. (Notepad++ 2016.)

4.3 JavaScript

JavaScript on Brendan Eichin kehittämä ja Netscape Communications Corporationin vuonna 1995 julkaisema dynaaminen komentosarjakieli. Myöhemmin Microsoft kehitti JavaScriptistä oman version, joka julkaistiin seuraavana vuonna nimellä Jscript. Se oli toiminnoiltaan sopiva Microsoftin omaan selaimeen Internet Exploreriin. JavaScript mielletään usein samaksi ohjelmointikieleksi kuin Java, mutta ne eroavat toisistaan paljon. Java on vanha ohjelmointikieli. JavaScriptiä käytetään luomaan nettisivuille interaktiivisia toimintoja, joista on loppupäässä apua käyttäjälle. Ajan myötä JavaScript on kehittynyt entisestään, ja nykyään sitä käytetään esimerkiksi Node.js:ssä, peliteollisuudessa sekä työpöytä- ja mobiilisovellusten tekemiseen. JavaScript muistuttaa C-ohjelmointikieltä, ja sen kehityksessä on otettu mallia Self- ja Scheme-ohjelmointikielistä. (W3Schools 2012.)

4.4 Node.js ja moduulit

Node.js on palvelinpuolella toimiva alusta, joka käyttää Google Chromen JavaScript-moottoria. Node.js on julkaistu vuonna 2009, ja sen on kehittänyt Ryan Dahl. Node.js on avoimen lähdekoodin ajoympäristö joka on alustasta riippumaton. Se mahdollistaa palvelinpuolen ja verkkosovellusten helpon kehittämisen. JavaScriptillä kirjoitetut sovellukset voidaan suorittaa Node.js:llä Node.js Runtime-osiossa esimerkiksi OS X-, Microsoft Windows- ja Linux-käyttöympäristöissä. Node.js:n kirjasto on yhdistelmä monista eri JavaScript-moduuleista. (Tutorialspoint 2016.)

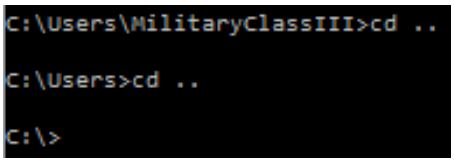
Node.js:n ominaisuuksia ovat asynkronisuus ja tapahtumaohjattavuus. Kaikki ohjelmointirajapinnat Node.js:n kirjastossa ovat asynkronisia eli estottomia. Tällöin Node.js:llä luotu palvelin ei koskaan jää odottamaan rajapinnan palauttettavaa dataa. Palvelin siirtyy seuraavaan rajapintaan kutsumalla sitä. Node.js:n ilmoitusmekanismi ilmoittaa palvelimelle, kun aikasemmalta rajapinnalta haluttu vastaus saadaan. Google Chromen

JavaScript-moottorille pohjautuva Node.js-kirjasto on nopea koodien suorittamiseen. (Tutorialspoint 2016.)

Sovellukset käyttävät usein monia eri säikeitä, jotka toimivat prosessin sisällä ja pystyvät viestimään keskenään yhteisellä muistialueella. Säikeillä pystytään toteuttamaan toimintoja samanaikaisesti muiden säikeiden rinnalla saman prosessin sisällä, mutta Node.js käyttää vain yhtä säiettä. Node.js:n yksi säie on tilanteisiin mukautuva ja suorituskyvyltään tehokas. (Tutorialspoint 2016.)

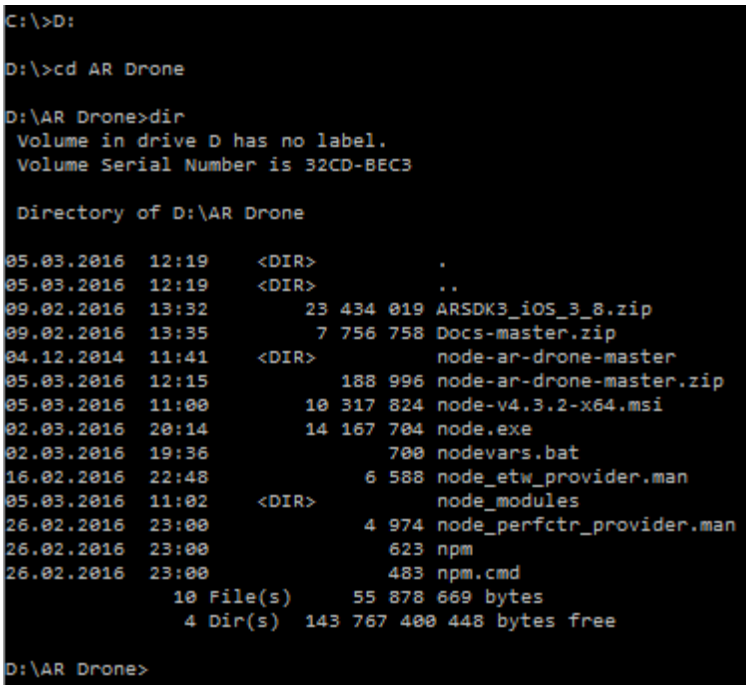
Opinnäytetyössä käytetään Node.js:n v4.3.2 LTS -versiota, jonka saa ladattua Node.js:n kehittäjien nettisivuilta ilmaiseksi. Asennuspakkauksen latauduttua ohjelmalle valitaan haluttu sijainti ja se asennetaan. Asennus tapahtuu Setup Wizardilla. Ennen kuin kopteria pääsee ohjelmoimaan tietokoneella, täytyy asentaa vielä ar-drone-moduuli, minkä saa ladattua internetistä. Latauksen jälkeen se puretaan Node.js:n asennuskansioon. (NodeCopter Core 2012.)

Moduulin asennukseen käytetään Windowsin komentokehotetta. Asennuksessa tarvitaan cd-komentoa (change directory), jolla pystytään liikkumaan kansioissa. Kansioissa pääsee liikkumaan takaisin päin lisäämällä cd-komennon perään kaksi pistettä, kuten kuvassa 11.



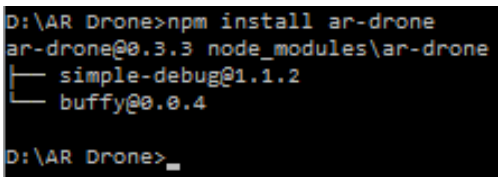
KUVA 11. Esimerkki komennosta, jolla liikuttiin askel kerrallaan C-aseman juureen

Jotta ar-drone-moduuli voidaan asentaa Node.js:ään, täytyy komentokehotteella mennä siihen asennuskansioon, missä Node.js sijaitsee. Kohdeasemaa saadaan vaihdettua D:-komennolla. Haluttuun kansioon päästään liikkumaan cd-komennolla, josta esimerkki kuvassa 12. Varmistaakseen kansion sisällön voi käyttää dir-komentoa (directory).



KUVA 12. Siirtyminen Node.js-asennuskansioon

Kansion sisältö tulisi näyttää melko samalta kuin kuvassa 11, jonka jälkeen pääsee asentamaan ar-drone-moduulin. Moduuli asennetaan komennolla npm install ar-drone, joka on asennuskomento node package managerille (NodeCopter Core 2012). Jos asennus onnistuu, tulisi komentokehotteen näyttä samalta kuin kuvassa 13.



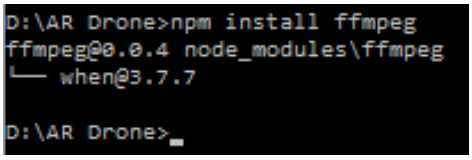
KUVA 13. ar-drone-moduulin asennus

4.5 FFmpeg

FFmpeg on työkalu, jota käytetään multimedian käsittelyyn, jos tarvitsee purkaa, koodata, muuntaa, multipleksata, lähettää, suodattaa tai toistaa ihmisen tai koneen tekemää multimedialähdettä. Se tukee useita video- ja kuvaformaatteja vanhoista uusimpiin ja kuvalähteestä riippumatta FFmpeg:n pitäisi toimia. FFmpeg:n tukemia käyttöjärjestelmiä ovat esimerkiksi Linux, Mac OS X, Microsoft Windows, BSD ja Solaris.

Opinnäytetyössä FFmpeg:ä käytetään Windows 10 -ympäristössä. Asennustiedostot saa ladattua Ffmpeg-kehittäjien nettisivuilta, ja ohjelma on täysin ilmainen. (FFmpeg 2016.)

FFmpeg:n asennus tapahtuu samalla tavalla kuin Node.js:lle asennetun ar-drone-moduulin. Ladatut asennustiedostot viedään Node.js:n asennuskansioon ja puretaan sinne. Tämän jälkeen avataan komentokehote ja siirrytään Node.js:n kansioon. Ffmpeg-moduulin asentaminen Node.js:lle tapahtuu komennolla npm install ffmpeg. Jos Ffmpeg-moduulin asentaminen onnistui, tulisi sen näyttää samalta kuin kuvassa 14.



KUVA 14. Ffmpeg-moduulin asentaminen Node.js:ään

Opinnäytetyössä FFmpeg:llä tehdään kopterille pngstream, joka tarkoittaa, että kopterilla otetaan monta kuvaa sekunnissa ja FFmpeg käsittelee otetut kuvat kuvajonoiksi. Kuvajonon lopputulos tulee muistuttamaan ulkoasultaan videota, joka lähetetään Internet-selaimeen näkymään.

5 KOPTERIN KÄSKYKANTA

Opinnäytetyössä kopteriin sovellettava ohjelmoinnin käskykanta koostuu JavaScripteistä. JavaScripteillä pystytään ohjaamaan kopterin toimintaa ja laittamaan kopteri tekemään jotakin tiettyä rutiinia. JavaScripteillä voidaan luoda kopterille automaattisia toimintoja. JavaScriptit ajetaan Node.js:llä tietokoneelta kopterille. Kopterin käskykannasta löytyy käskyjä laitteen alustamiseen, liikumiseen, moottorien ohjaukseen, objektin tunnistamiseen sekä navigointi- ja verkkodatalle.

5.1 Moduulit

Kopterin ohjelmointiin on saatavissa erilaisia moduuleita, jotka asennetaan Node.js:lle Node Package Managerilla. Käytettävät moduulit täytyy määrittää koodia kirjoitettaessa koodin alkuun. Moduulit sisältävät kirjaston asetuksille ja ominaisuuksille, joita tarvitaan kopterin ohjaamiseen. Esimerkki moduulin käyttöönottamisesta: ”var arDrone = require(’ar-drone’);”. Esimerkissä var-komennolla tehdään uusi muuttuja, jonka nimeksi määritetään ”arDrone”. Sen jälkeen määritetään sijainti, mistä asennettu moduuli ladataan käyttöön. Moduulin tulisi olla asennettuna node-modules-kansiossa. (Mehner 2014.)

5.2 Konfiguraatiot

Konfiguroinnilla kopterille saadaan käyttöön moodulien sisältämiä ominaisuuksia. Konfigurointi tapahtuu aina komennolla ”client.config”, jonka jälkeen määritetään käytettävä ominaisuus ja asetetaan ominaisuuden parametrit. Parametrit asetuksille löytyy ar-drone-moduulin constants JavaScript-tiedostosta ja AR.Drone SDK:sta. Konfigurointiasetuksia on paljon ja merkinnällä R, W tai R/W on määritetty, onko asetus muokattavissa vai ainoastaan luettavissa. Yleiskonfigurointi-komennoilla kopteri saadaan tulostamaan tietoa tai ottamaan käyttöön asetuksia. Esimerkiksi komennolla ”client.config(’general:navdata_options’, navdata_options);” määritetään

halutut sensorit ja laitteet käyttöön. Taulukosta 1 löytyy komentoja yleiskonfigurointiin.

TAULUKKO 1. Yleiskonfigurointi komentoja (Piskorski ym., 73 - 75)

Yleiskonfiguroinnit Komento	client.config(general:); Toiminto	Read/Write
general:num_version_config	Osajärjestelmän versio	R
general:num_version_mb	Emolevyn versio	R
general:num_version_soft	Sovelluksen versio	R
general:drone_serial	Kopterin sarjanumero	R
general:soft_build_date	Ohjelmiston käännös pvm	R
general:motor1*_soft	Moottoripiirilevyjen versio	R
general:motor1**_hard	Moottorien laitteisto versio	R
general:motor1***_supplier	Toimittjan versio moottoripiiristä	R
general:ardrone_name	Kopterin nimi, ei WiFi SSID	R/W
general:flying_time	Kokonais lentoaika sekuntteina	R
general:navdata_demo	Karsittu määrä navdataa	R/W
general:navdata_options	Valitaan navdataa käyttöön****	R/W
general:com_watchdog	Ajastin, asettaa kopterin leijumaan	R/W
general:video_enable	Automaattisesti jo päällä	R/W
general:vision_enable	Automaattisesti jo päällä	R/W
general:vbat_min	minAkunvaraus, pakkosammutus	R
*Vaihtoehtoisesti motor1,2,3 tai 4 **Vaihtoehtoisesti motor1,2,3 tai 4 ***Vaihtoehtoisesti motor1,2,3 tai 4 ****Vaihtoehtoisia sensorien käyttövalintoja, valitaan yksitellen		

Kontrollikonfigurointikomennoilla kopteri saadaan tulostamaan tietoa sensoreilta, joita ovat kiihdytin, gyroskooppi ja magnetometri. Kontrollikonfiguroinnilla voidaan myös määrittää kopterille raja-arvoja liikkumiseen pysty- ja vaakasuunnassa. Lisäksi voidaan määrittää, onko kopteri lentämässä sisällä vai ulkona. Esimerkiksi komennolla ”client.config(’control:altitude_max’, 1500);” kopteri lentää maksimissaan 1,5 metrin korkeudella. Taulukossa 2 on kontrollikonfiguroinnin komentoja.

TAULUKKO 2. Kontrollikonfigurointikomentoja (Piskorski ym., 76 - 81)

Kontrollointikonfiguraatiot Komento	client.config(control:); Toiminto	Read/Write
control:accs_offset	Kiihdyttimen siirtymät	R
control:accs_gains	Kiihdyttimen tulot	R
control:gyros_offset	Gyroskoopin siirtymät	R
control:gyros_gains	Gyroskoopin tulot	R
control:gyros110_offset	Tulostaa lukeman	R
control:gyros110_gains	Tulostaa lukeman	R
control:magneto_offset	Tulostaa lukeman	R
control:magneto_radius	Tulostaa lukeman	R
control:gyro_offset_thr_x*	Tulostaa lukeman	R
control:pwm_ref_gyros	Tulostaa lukeman	R
control:osctun_value	Tulostaa lukeman	R
control:osctun_test	Tulostaa lukeman	R
control:control_level	Komentoihin reagointi taso 0/1	R/W
control:euler_angle_max	Kallistuskulmat**	R/W
control:altitude_max	Max lentokorkeus, 0 - 100000 cm	R/W
control:altitude_min	50cm, ei kannata muuttaa	R/W
control:control_vz_max	Max lentonopeus, 200-2000mm/s	R/W
control:control_yaw	Max kallistus***	R/W
control:outdoor	Ulkonalento, true/false	R/W
control:flight_without_shell	Ilman sisäsuojaa lentäminen****	R/W
control:flying_mode	*****	R/W
control:hovering_range	******	R/W

*Vaihtoehtoisesti x, y tai z
**Annetaan radiaaneina 0 - 0.52 välillä, max 30°
***Kallistuksen muutosnopeus suositeltavaa 40 - 350°/s välille, radiaaneina n. 0,7-6.11rad/s
****Kertoo kopterille ettei sisällä lentämiseen tarkoitettu suojakuori käytössä, vaikutta tasapainon hakemiseen, mutta komento ei ole toiminnoiltaan sama kuin outdoor
*****Mahdollistaa leijumisen käyttöön oton ennalta määrätyn kuvion yläpuolella, vaatii myös tunnistuksen käytön, joka voidaan ottaa käyttöön komennolla detect:detect_type ja asettamalla tarvittavat parametrit
******Asettaa leijumis korkeuden kuvion yläpuolelle, etäisyys annetaan millimetreinä

Kopterin kameroita voidaan käyttää kuvioiden havaitsemiseen. Havaitsemiskonfiguraatioilla pystytään valitsemaan kuvion tunnistamiseen

käytettävä kamera ja valita kuviot, jotka tunnistetaan ja jotka aiheuttavat toimenpiteen. Esimerkiksi komennolla ”client.config(’detect:detect_type’, 2);” valitaan tunnistettavaksi kuvioksi tunnisteväritarra. Taulukossa 3 on havaitsemiseen käytettäviä komentoja.

TAULUKKO 3. Havaitsemisen konfiguraatio komentoja (Piskorski ym., 89 - 91)

Havaitsemiskonfiguraatiot Komento	client.config(detect:); Toiminto	Read/Write
detect:enemy_colors	Tunnistaa väritarran*	R/W
detect:enemy_without_shell	**	R/W
detect:detect_type	***	R/W
detect:detections_select_h	****	R/W
detect:detections_select_v_hsync	*****	R/W
detect:detections_select_v	******	R/W
*Vaihtoehtoina vihreä, keltainen ja sininen. Valinta parametreina 0,1 ja 2 **Tunnistaa toisen kopterin ulkon lentämiseen tarkoitetun suojakuoren. Valinta paremetreina 0 tai 1 ***Etsittävä kuvio, joka laukaisee toimenpiteen. Vaihtoehtoina väritarra, leijuntakuvio, useampi tunnistettava tai ei mitään ****Horisontaali eli pääkameralla tunnistettavat kuviot, jotka aiheuttavat toimenpiteen. Valinta parametreina 0,1,2,n… *****Pohjakameralla tunnistettavan kuvion valinta. Valinta parametreina 0,1,2,n… ******Pohjakameralla tunnistettavat kuviot, jotka aiheuttavat toimenpiteen. Valinta parametreina 0,1,2,n…		

5.3 Liikekomentoja

Kopterin käskykannassa on liikeelle tehtyjä komentoja. Komennoilla saadaan kopteri liikkumaan x-, y- ja z-akselin suunnassa halutulla tavalla. Myös kopterin kierrolle y-akselin ympäri on komentoja. Esimerkiksi komennolla ”client.takeoff();” kopteri nousee ilmaan ja jää leijumaan paikkalleen. Taulukossa 4 on kaikki perusliikkumiseen käytettävät komennot. Parametreina toimivat nopeus välillä 0 – 1 ja suoritettava aika mikrosekunteina.

TAULUKKO 4. Liikekomentoja (Mehner 2014)

Perusliikekäskyjä	Parametri	Toiminto
client.takeoff()	ei parametria, 'callback'	Nousee ilmaan
client.land()	ei parametria, 'callback'	Laskeutuu
client.stop()	ei parametria, 'callback'	Pysäyttää käskysarjan, jää leijumaan
client.up()	Täytyy asettaa nopeus (0 - 1)	Lisää korkeutta
client.down()	Täytyy asettaa nopeus (0 - 1)	Laskee korkeutta
client.clockwise()	Täytyy asettaa nopeus (0 - 1)	Kääntyy myötäpäivään
client.counterClockwise()	Täytyy asettaa nopeus (0 - 1)	Kääntyy vastapäivään
client.front()	Täytyy asettaa nopeus (0 - 1)	Liikkuu eteenpäin
client.back()	Täytyy asettaa nopeus (0 - 1)	Liikkuu taaksepäin
client.left()	Täytyy asettaa nopeus (0 - 1)	Liikkuu horisontaalisesti vasemmalle
client.right()	Täytyy asettaa nopeus (0 - 1)	Liikkuu horisontaalisesti oikealle

Kopterin käskykantaan kuluu myös valmiiksi luotuja animoidun liikkeen komentoja, joilla kopteri saadaan tekemään voltteja tai äkkikäännöksiä. Valmiiksi tehdyillä liikkeillä on tarkoitus tuoda esille kopterin ketteryyttä. Kyseisiä valmiiksi animoituja liikkeitä kopterille löytyy käskykannasta lähinnä siksi, että kopterille tehdyissä peleissä liikkeitä on mahdollista käyttää väistelyyn. Esimerkiksi komennolla ”client.animate(’flipAhead’, 2000);” kopteri tekee voltin etuperin. Toiminnon suorittamiselle annetaan aika mikrosekunteina. Taulukossa 5 ovat kaikki kopterin animaatioliikkeet.

TAULUKKO 5. Animoituja liikkeitä (Mehner 2014)

Liikekäskyjä	client.animate();
Komento:	
'phiM30Deg'	'phiDance'
'phi30Deg'	'thetaDance'
'thetaM30Deg'	'vzDance'
'theta30Deg'	'wave'
'theta20degYaw200deg'	'phiThetaMixed'
'theta20degYawM200deg'	'doublePhiThetaMixed'
'turnaround'	'flipAhead'
'turnaroundGodown'	'flipBehind'
'yawShake'	'flipLeft'
'yawDance'	'flipRight'

5.4 LED:ien käyttäminen

Kopterin jokaisen moottorin piirilevylle on sijoitettu LED-valo, jota voidaan sytyttää ja välkyttää kolmella eri värillä: vihreä, oranssi ja punainen. LED:ien käyttäminen on hyödyllistä silloin, jos tarvitsee saada helposti tietoa kopterin tilasta. Esimerkiksi hätätilassa valot palavat punaisena. LED:t voidaan laittaa vilkkumaan vihreänä silloin, kun saavutetaan haluttu lentokorkeus, alustaminen onnistuu tai etsitty objekti tunnistetaan. Nämä ovat hyviä LED-valojen käyttötapoja indikoimaan tapahtumaa. LED:ejä pystyy kontrolloimaan komennolla ”client.animateLeds(’blinkGreen’, 10, 5);”, jolloin kopterin valot välkkyvät vihreinä 10 kertaa sekunnissa viiden sekunnin ajan. Taulukossa 6 on kaikki ar-drone-moduulin sisältämät LED-valojen käyttökomennot.

TAULUKKO 6. LED-valojen käyttökomentoja (Mehner 2014)

LED-valojen käyttökomennot	client.animateLeds();
Komento:	
'blinkGreenRed'	'rightMissile'
'blinkGreen'	'leftMissile'
'blinkRed'	'doubleMissile'
'blinkOrange'	'frontLeftGreenOthersRed'
'snakeGreenRed'	'frontRightGreenOthersRed'
'fire'	'rearLeftGreenOthersRed'
'standard'	'rearRightGreenOthersRed'
'red'	'leftGreenRightRed'
'green'	'leftRedRightGreen'
'redSnake'	'blinkStandard'
'blank'	

6 KÄYTÄNNÖN ESIMERKKEJÄ

Opinnäytetyön aihepiiriin kuuluu kopterin ohjelmointi ja tässä luvussa kopterille tehdään muutama malliesimerkki siitä, minkälaisia JavaScriptejä kopterille voi kirjoittaa. Ensimmäiseksi kopterille tehdään yksinkertainen JavaScript, jolla se nousee ilmaan ja tekee muutamia liikkeitä, minkä jälkeen se laskeutuu. Toiseksi tehdään kuviontunnistus, jonka avulla kopteri osaa seurata tunnisteväritarraa. Kolmanneksi kopterille tehdään videolähetys selaimeen. Kaikki kirjoitetut JavaScriptit on hyvä sijoittaa Node.js-kansion juureen. Koodien suorittamiseen tarvitaan langaton yhteys tietokoneen ja kopterin välillä. Esimerkki käytettävästä koodista löytyy tämän opinnäytetyön liitesivuilta.

6.1 Liikkeelle lähteminen ja laskeutuminen

Aluksi kopterille tarvitsee määrittää ar-drone-moduuli. Samaa moduulia käytetään myös kaikissa muissa esimerkeissä. Moduulin määrittäminen tapahtuu komennolla ”var arDrone = require(’ar-drone’);”, jolloin heti koodin aluksi ladataan moduulin sisältämä kirjasto. Tämän jälkeen avataan yhteys kopterin ohjaukseen komennolla ”var client = arDrone.createClient();”. Alun määritysten jälkeen voidaan käyttää kopterille tehtyjä komentoja sen ohjaamiseen. (Mehner 2014.) Kuvassa 15 on helppo ja yksinkertainen JavaScript-koodi, joka saa kopterin liikkumaan automaattisesti ennalta määrätyillä käskyillä.

```
/* Simple drone controlling code
    Lähde: https://github.com/felixge/node-ar-drone
    15.3.2016
    Juuso Savolainen
*/

var arDrone     = require('ar-drone');              // Used module
var client      = arDrone.createClient();           // Open up control window

client.takeoff();                                   // Takes off

client
    .after(5000, function() {this.clockwise(0.5)})  // Wait for 5000ms before action
    .animate('flipBehind',1000)                     // Put action inside function
    .after(3000, function() {
        this.stop();                                // Stop sets drone to stable state
        this.land();                                // Lands the drone
        });
```

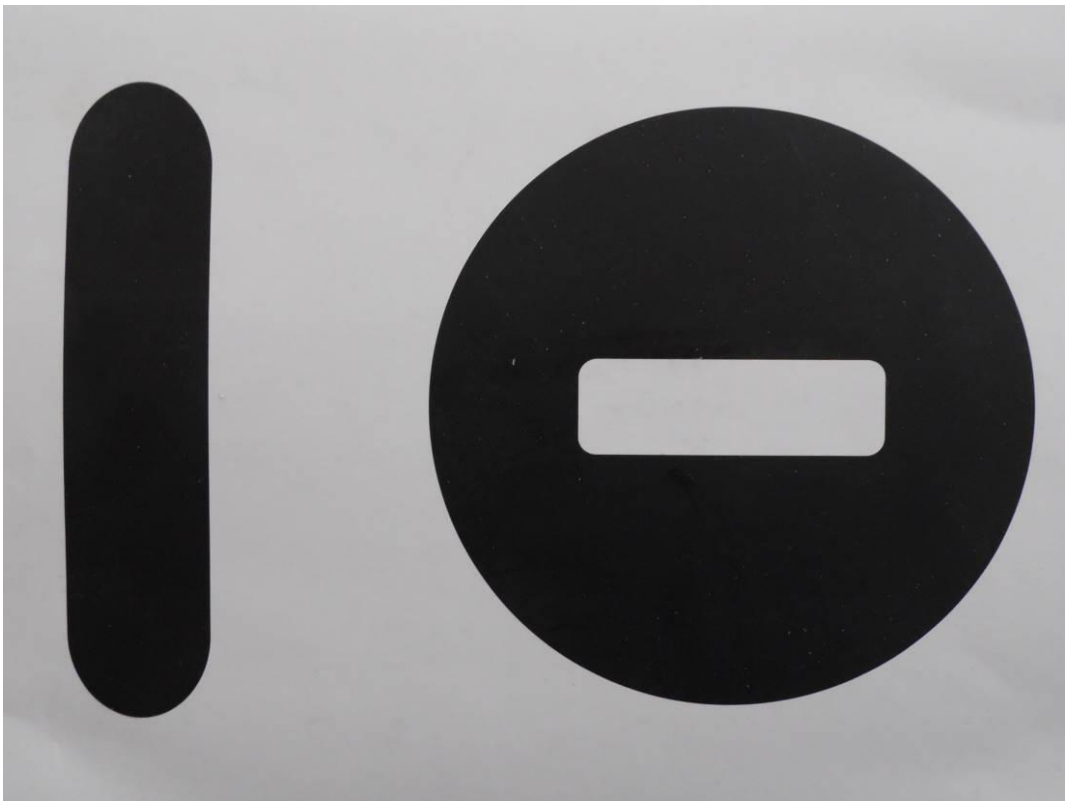
KUVA 15. Yksinkertainen liikekoodi

6.2 Kuviontunnistus

Kopterille on saatavissa sille tarkoitettuja tunnisteväritarroja, jotka toimivat kohteena kameralla tunnistamista varten. Tarrat ovat muodoltaan samanlaisia, mutta värit vaihtelevat ja eriväriset tarrat pystytään erottamaan toisistaan. Tässä esimerkissä käytettävä tarra on oranssi sininen, kuten kuvassa 16.

KUVA 16. Kuviontunnistamista varten käytettävä tarra

Kopterille on olemassa kuvio, jonka kopteri tunnistaa pohjassa olevalla kameralla ja jää leijumaan sen kohdalle, jos konfiguroinneissa on otettu käyttöön kuvan 17 tunnistaminen ja toiminnot.



KUVA 17. Paikallaan leijumiseen käytettävä kuvio

Kopterille tehdään kuviontunnistus kuvassa 16 esitetyllä tarralla. Kuviontunnistus tehdään asettamalla oikeat konfiguraatiot ja asettamalla lentoparametreja kopterille, jolloin kopteri osaa lentää kuvion perässä, jos se liikkuu. Kuviontunnistukseen ja kuvion perässä liikkumiseen käytettävä JavaScript löytyy opinnäytetyön liitteestä 1.

6.3 Videokuvan lähetys selaimeen

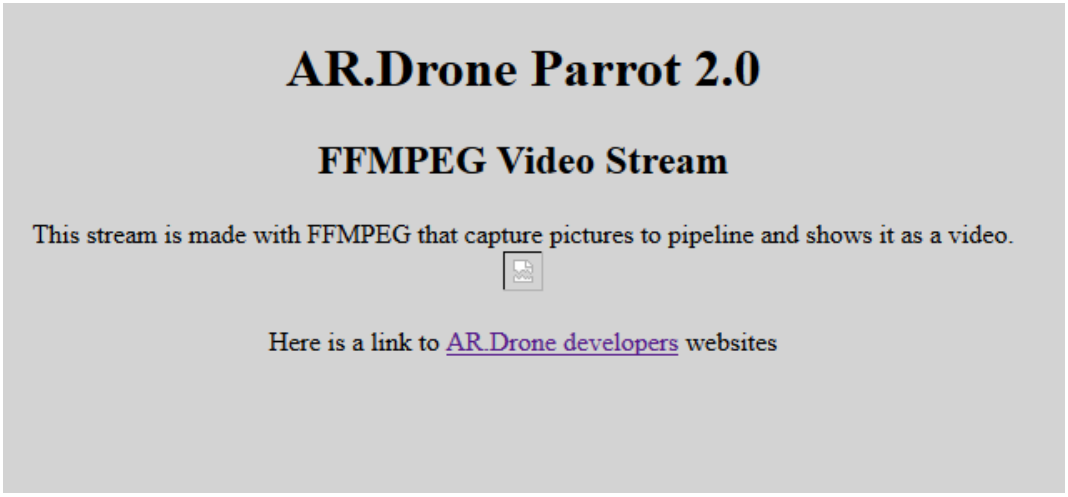
Kopterilla voidaan toteuttaa videolähetys nettiselaimeen FFmpeg:n avulla. Videolähetyksen vastaanottoa varten kopterille täytyy määrittää koodissa käytettävä http-moduuli, jota käytetään selaimeen lähetettävässä lähetyksessä. Lisäksi täytyy luoda ja avata tarvittava lähetys, joka tapahtuu ar-drone-png-stream-moduulin avauksella. Lähetys-moduulin yhteydessä määritetään lähetykselle portti, johon video lähetetään. Lisäksi tarvitaan http-moduuli, joka avustaa videolähetyksessä. Nettiselaimella voidaan kuunnella tätä määritettyä porttia ja saada videokuva näkyviin.

Videokuvan vastaanottamiselle tehdään oma nettisivu käyttämällä HTML5-kieltä (HyperText Markup Language), joka on yleinen nettisivujen koodaamisessa käytetty kieli. Video saadaan tämän avulla näkymään käyttäjälle siistitysti. Lähetyksen vastaanottoon käytettävä koodi on helppo tehdä. Opinnäytetyössä käytetty koodi on esitetty kuvassa 18.

```
<!DOCTYPE html>
<meta charset="UTF-8">
<html>
<head>
    <title>Oppari</title>
</head>
<body style="background-color: lightgrey;">

        <h1 style="text-align:center;">AR.Drone Parrot 2.0</h1>

        <h2 style="text-align:center;">FFMPEG Video Stream</h2>
            <p style="text-align:center;">
            This stream is made with FFMPEG that
            capture pictures to pipeline and
            shows it as a video.
            <br>
            <img src="http://localhost:8000"/>
            <br>
            </p>

        <footer style="text-align:center;bottom;">
        <p>Here is a link to <a href=http://ardrone2.parrot.com/>AR.Drone developers</a> websites</p>
        </footer>
</body>
</html>
```

KUVA 18. Videolähetyksen vastaanottoon käytettävä koodi

Tärkein osa lähetyksen vastaanottoa on kuvien lataaminen selaimeen. Välittyvä video on kuvien muodossa, joten koodissa voidaan käyttää komentoa ”” ja sitten haetaan näkyviin kopterin lähettämät kuvat portista 8000. Kuvassa 18 olevan HTML:n voi kirjoittaa esimerkiksi Notepadilla, kunhan tallennettavaksi tiedostomuodoksi valitaan HTML. Tallennettu koodi voidaan suorittaa käyttämällä selainta koodin avaamiseen. Kuvassa 19 on kuvan 18 kirjoitettu HTML visuaalisessa muodossa.



KUVA 19. Videolähetyksen vastaanotto selaimessa

7 YHTEENVETO

Opinnäytetyössä tutkittiin Parrot AR.Drone 2.0-quadrokopteria. Tavoitteena oli luoda käyttöohje tutkimalla kopteria sen laitteistosta ohjelmointiin, jotta kuka tahansa voisi ymmärtää, mikä drone on ja kuinka sitä käytetään tai ohjelmoidaan. Työ eteni tutkimalla miehittämättömien aluksien historiaa, laitteistoa, ohjelmointiin tarvittavia työkaluja sekä ohjelmointia. Kopterin käyttöohjeiden lisäksi kopterille tehtiin ohjelmoinnista esimerkkejä, jota edelsi saatavilla oleviin moduuleihin, ohjelmointisovelluksiin ja ohjauskäskyihin tutustuminen. Ohjelmointiesimerkkien ja ohjelmointisovelluksiin annetujen ohjeiden avulla kenen tahansa pitäisi pystyä tekemään Parrot AR.Drone 2.0:lle samat toimintaominaisuudet.

Miehittämättömien aluksien kehitys on alkanut jo 1900-luvun alkupuolella, ja taustalla on ollut vahvasti sotateollisuus. Lennokkien päätoiminen tehtävä on ollut tiedustelu- ja tiedonkeruutehtävät. Teknologian kehityksen myötä lennokkeja alettiin aseistamaan 2000-luvun taitteessa. Lennokit olivat pitkään vain armeijan saatavilla, mutta nykyään pieniä quadrokoptereita pystyy ostamaan kuka tahansa. Kuluttajille suunnatut quadrokopterit on kehitetty lähinnä ilmakuvaamista ja vapaa-ajan lennättämistä varten.

Opinnäytetyössä käytetyn kopterin laitteiston tutkimisella saatiin selvitettyä kopterin toimintaperiaate ja se, mitä kaikkea sillä on mahdollista tehdä. Lisäksi selvisi, että kopterille on saatavissa lisälaite, joka kiinnitetään sen USB-väylään. Lisälaite toimii GPS:llä, ja sillä pystytään tekemään kopterille ennalta suunniteltuja lentoreittejä. Lentoreittien suunnittelusta on hyötyä esimerkiksi metsäalueiden kuvaamisessa. Kopterin tutkimisen tuloksena luotiin kopterille pikakäyttöohje, joka on tämän työn liitteenä.

Kopterin ohjelmointia varten tutkittiin saatavilla olevia dronekehittäjien tekemiä moduuleita ja kirjastoja. Ohjelmointi edellytti myös ohjelmointisovelluksien tutkimista. Opinnäytetyössä käytetyt ohjelmointisovellukset ovat kaikki ilmaisia ja kaikkien saatavilla.

Käytettyihin sovelluksiin onnistuttiin tekemään selkeät asennusohjeet ja kertomaan itse ohjelmista ja käytöstä.

Tärkein käytetyistä sovelluksista opinnäytetyön kannalta on Node.js. Sillä pystytään ajamaan kopterille kirjoitetut JavaScriptit. Lisäksi Node.js:lle täytyi asentaa moduuleita, joilla lisättiin toimintoja ja ominaisuuksia. Moduulien asentamiseen annettiin asennusohjeet ja ohjelmointiin tehtiin käytännön esimerkkejä.

Kopterin tutkimisen tavoitteena oli luoda tekoälyä kopterille tai saada se tekemään automatisoituja toimintoja. Varsinaisen tekoälyn luominen ei onnistunut, mutta automaattisten toimintojen tekeminen onnistui. Opinnäytetyön ohjelmointivaiheessa ja JavaScriptien suorittamisessa tuli vastaan ongelma Windows 10 -käyttöjärjestelmän kanssa. Toisinaan Windows 10 ei suostunut suorittamaan kirjoitettua JavaScriptiä Node.js:n kautta toistaiseksi tuntemattoman JavaScript-virheen takia. Ohjelmointi saatiin kuitenkin onnistumaan ongelmitta Windows 7 -käyttöjärjestelmällä, jolloin myös JavaScriptien debuggaus sujui vaivatta. JavaScriptien ajaminen Node.js:llä vaatii langattoman yhteyden tietokoneelta kopterille.

Kopterin ohjelmoinnin jatkotutkimuksena voitaisiin selvittää lisää objektintunnistusta käyttäen esimerkiksi OpenCV:tä. Lisäksi ohjelmointiympäristönä voisi toimia Linux-käyttöjärjestelmä. OpenCV:llä olisi mahdollista tehdä omia tunnistettavia kuvioita kopterille ja kopterin ohjelmointi tapahtuisi JavaScript- tai C++-ohjelmointikielellä. Myös Windows 10 -käyttöjärjestelmän kanssa vastaan tulleet JavaScriptien suorittamisongelmat voisi olla yksi tutkimuskohde tätä opinnäytetyötä kehittäen.

LÄHTEET

ARM. 2015. Cortex-A8 Processor [viitattu 3.3.2016]. Saatavissa: http://www.arm.com/products/processors/cortex-a/cortex-a8.php

Cole, C. 2014. Rise of the Reapers: A brief history of drones. Drone Wars UK [viitattu 8.3.2016]. Saatavissa: http://dronewars.net/2014/10/06/rise-of-the-reapers-a-brief-history-of-drones/

FFmpeg. 2016. About FFmpeg [viitattu 5.3.2016]. Saatavissa: https://www.ffmpeg.org/about.html

Jain, P. 2016. Magnetometers. EngineersGarage [viitattu 25.2.2016]. Saatavissa: http://www.engineersgarage.com/articles/magnetometer

Kumar, V. 2012. Robots that fly and cooperate. TED 2/2012 Video [viitattu 10.3.2016]. Saatavissa: https://www.ted.com/talks/vijay_kumar_robots_that_fly_and_cooperate#t-102258

Mehner, R. 2014. Node-ar-drone. GitHub [viitattu 17.3.2016]. Saatavissa: https://github.com/felixge/node-ar-drone

Nesta. 2016. Drones: a history of flying robots [viitattu 8.3.2016]. Saatavissa: http://www.nesta.org.uk/drones-history-flying-robots

NodeCopter Core. 2012. Hacked Guide [viitattu 25.2.2016]. Saatavissa: http://www.nodecopter.com/hack

Notepad++. 2016. About Notepad [viitattu 25.2.2016]. Saatavissa: https://notepad-plus-plus.org/

Parrot. 2015. Parrot AR.Drone 2.0 [viitattu 25.2.2016]. Saatavissa: http://ardrone2.parrot.com

Piskorski, S., Brulez, N., Eline, P. & D´Haeyer, F. 2012. AR.Drone Developer Guide. Scribd [viitattu 15.3.2016]. Saatavissa: http://www.scribd.com/doc/175660012/ARDrone-Developer-Guide#scribd

Sparkfun. 2016b. Gyroscope [viitattu 25.2.2016]. Saatavissa: https://learn.sparkfun.com/tutorials/gyroscope

Sparkfun. 2016a. Accelerometer Basics [viitattu 25.2.2016]. Saatavissa: https://learn.sparkfun.com/tutorials/accelerometer-basics

Scintilla. 2016. Scintilla [viitattu 25.2.2016]. Saatavissa: http://www.scintilla.org/

Tutorialspoint. 2016. Node.js – Introduction [viitattu 3.3.2016]. Saatavissa: http://www.tutorialspoint.com/nodejs/nodejs_introduction.htm

VanDomelen, J. 2012. Tesla’s Take on Unmanned Vehicles. MentorGraphics [viitattu 8.3.2016]. Saatavissa: https://blogs.mentor.com/jvandomelen/blog/2012/06/26/tesla%E2%80%99s-take-on-unmanned-vehicles/

Wikipedia. 2006. Gyroscope [viitattu 25.2.2016]. Saatavissa: https://commons.wikimedia.org/wiki/File:3D_Gyroscope.png

Wikipedia. 2014. CMD [viitattu 25.2.2016]. Saatavissa: https://fi.wikipedia.org/wiki/CMD.EXE

W3Schools. 2012. A Short History if JavaScript [viitattu 3.3.2016]. Saatavissa: https://www.w3.org/community/webed/wiki/A_Short_History_of_JavaScript

LIITTEET

LIITE 1 Objektin seuraamiseen käytettävä koodi

LIITE 2 Kopterin pikakäyttöohje

LIITE 1

```
// LAHDEN AMMATTIKORKEAKOULU
// MIKI SILTAKOSKI JA JUUSO SAVOLAINEN
// AR DRONE - OBJECT FOLLOWING

// Used node modules:
var  arDrone = require('ar-drone')
  ,  arDroneConstants = require('ar-drone/lib/constants')
  ,  keypress = require('keypress')
  ,  tty = require('tty')
  ,  http = require('http')
;

// Create client connection:
var client  = arDrone.createClient();
// Create videostream from drone and send it to port 8000:
require('ar-drone-png-stream')(client, { port: 8000 });

// Sensor data selection:
function navdata_option_mask(c){
    return 1 << c;
}
var navdata_options = (
    navdata_option_mask(arDroneConstants.options.DEMO)
  | navdata_option_mask(arDroneConstants.options.VISION_DETECT=true)
  | navdata_option_mask(arDroneConstants.options.MAGNETO)
  | navdata_option_mask(arDroneConstants.options.WIFI)
);

// Drone configuration settings to control drone and set standards
client.config('control:altitude_max',1500);
client.config('control:outdoor',false);
client.config('control:flight_without_shell',false);
client.config('general:navdata_demo', true);
client.config('general:navdata_options', navdata_options);
client.config('video:video_channel', 0);
client.config('detect:detect_type', 2);
client.config('detect:detections_select_h', 1);
client.config('detect:enemy_colors', 3);

// Keyboard listening
keypress(process.stdin);

// Picture processing variables:
var findPic = 0; // This variable is used to find picture
var picture_x=0; // X-axis picture
var picture_y=0; // Y-axis picture
var picture_dist=0; // Distance of picture

// Variables that are used to flying. Depending on placement of picture. Without changes just idle:
var rotate = 2; // Used to rotate drone, set to 2
var altitude = 2; // Used to trim altitude, set to 2
var distance = 2; // Used to trim distance, set to 2

// Keyboard variables:
var flyMod = 5;
var oldValue = 0;
var currentValue = 0;
var temp = 0;
```

```
59
60  // Main function, loop
61  client.on('navdata', function (d) {
62
63      if (d.demo){
64
65          // näppäinsyötteiden kuuntelu funktio:
66          process.stdin.on('keypress', function (ch, key){
67              currentValue = key.name;;
68              if (currentValue != oldValue){
69                  if (currentValue == 'return'){
70                  flyMod = 1;
71                  temp = 1;
72                  }
73                  if (currentValue == 'escape'){
74                  flyMod = 2;
75                  temp = 1;
76                  }
77                  if (currentValue == 'space'){
78                  flyMod = 0;
79                  temp = 1;
80                  }
81              }
82          oldValue = currentValue;
83          });
84
85          // Keyboard commands, current action depends on flyMod value:
86          if (flyMod == 1 && temp == 1){ // enter
87              client.takeoff();
88              temp = 0;
89          }
90          if (flyMod == 2 && temp == 1){ // escape
91              client.stop();
92              temp = 0;
93          }
94          if (flyMod == 0 && temp == 1){ // spacebar
95              client.land();
96              temp = 0;
97          }
98
99          if (typeof process.stdin.setRawMode == 'function'){
100             process.stdin.setRawMode(true);
101         }else{
102             tty.setRawMode(true);
103         }
104         process.stdin.resume();
105
106
```

```
        // Picture plament on camera:
        if (d.visionDetect.nbDetected){
            picture_x=d.visionDetect.xc[0];
            picture_y=d.visionDetect.yc[0];
            picture_dist=d.visionDetect.dist[0];
            findPic = 1;
        }else{
            findPic = 0;
        }

        // Setting drone flying variables by picture placement:
        if (findPic == 1){

            if (picture_x < 300 && (rotate == 0  || rotate == 2)){rotate = 1;}
                else if (picture_x > 700 && (rotate == 1 || rotate == 2)){rotate = 0;}
                else if ( picture_x > 300 && picture_x < 700){rotate = 2;}

            if (picture_y < 250 && (altitude == 0  || altitude == 2)){altitude = 1;}
                else if (picture_y > 550 && (altitude == 1 || altitude == 2)){altitude = 0;}
                else if ( picture_y > 250 && picture_y < 550) {altitude = 2;}

            if (picture_dist < 100 && (distance == 0  || distance == 2)){distance = 1;}
                else if (picture_dist > 150 && (distance == 1 || distance == 2)){distance = 0;}
                else if ( picture_dist > 100 && picture_dist < 150){distance = 2;}
        }else{
            // Just idle
            altitude = 2;
            rotate = 2;
            distance = 2;
        }

        // Changes drones flying state with flying variables:
        if (rotate==0){client.clockwise(0.3);}
            else if(rotate==1){client.clockwise(-0.3);}
            else if(rotate==2){client.clockwise(0);}

        if (altitude==0){client.up(-0.3);}
            else if(altitude==1){client.up(0.3);}
            else if(altitude==2){client.up(0);}

        if (distance==0){client.front(0.05);}
            else if(distance==1){client.front(-0.05);}
            else if(distance==2){client.front(0);}

        if (altitude == 2 && rotate == 2 && distance == 2){
            client.stop();
        }

    }
});
```

LIITE 2

1. Varmista että käytössäsi on älypuhelin tai tabletti ja lataa laitteeseesi Applen tai Googlen sovelluskaupasta AR.FreeFlight-sovellus.
2. Lataa kopterin akku täyteen. Laturin merkkivalo ilmoittaa akun varauksen. Punainen valo ilmoittaa vajaasta varauksesta ja vihreä valo täydestä varauksesta.
3. Irroita kopterin suojakuori ja kiinnitä akku sille varattuun paikkaan keskelle kopteria.
4. Liitä akun liitin kopterin liittimeen, jonka jälkeen kopterin rootteiden alle pitäisi syttyä LED-valot. HUOM! Odota hetki ennen kuin laitat suojakuoren takaisin paikalleen. Kopteri nimittäin kokeilee kaikkia roottoreita pienellä nykäyksellä, joten älä estä liikettä.
5. Kopterin suorittaman testauksen jälkeen LED-valot palavat vihreinä, jos kopteri on tasaisella alustalla ja pystyy mittaamaan etäisyytensä maanpintaan/alustaan.
6. Aseta kopterin suojakuori paikoilleen.
7. Ota WiFi/WLAN käyttöön älypuhelimessa tai tabletissa ja etsi yhteyksistä ardrone niminen laite ja yhdistä siihen.
8. Avaa FreeFlight–sovellus.
9. Sovelluksen avauduttua avaa Piloting-painikketta painamalla pilotti-tila.
10. Painamalla takeoff-painiketta ruudun alareunassa kopteri lähtee lentoon.
11. Kopteria ohjataan pitämällä sormet ruudulla näkyvissä ”kontrollitateissa”. Vasen ottaa käyttöön puhelimen kallistamisella ohjauksen ja oikealla olevalla säädetään kopterin lentokorkeutta, sekä pystytään kääntymään vaakasuunnassa.
12. Lisäksi jos halutaan asettaa nopeus- ja korkeusrajoituksia kopterille niin täytyy painaa ruudun ylälaidassa näkyvää rattaan kuvaa, josta pääsee kopterin asetuksiin.