Maketa multirotorske letjelice u 4-propelerskoj konfiguraciji

Kulaš, Mislav

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

MAKETA MULTIROTORSKE LETJELICE U 4-PROPELERSKOJ KONFIGURACIJI

Diplomski rad

Mislav Kulaš

Osijek, 2017.

SADRŽAJ

1. UVOD

Dron ili UAV (eng. unmanned aerial vehicle) je bespilotna letjelica tj. zrakoplov bez posade u njemu koji se može nadzirati na daljinu te letjeti samostalno na unaprijed isplaniranom planu leta ili uz pomoć autonomnog sustava. Postoje dvije izvedbe dronova s obzirom kako ostvaruju polijetanje i letenje:

-VTOL [1]: (eng. Vertial take-off and landing) letjelice koje polijeću poput helikoptera i uzgon se ostvaruje propelerom. Prikazano na slici 1.1.

-CTOL [2]: (eng. Convectional take-off and landing) letjelice koje ostvaruju uzgon pomoću krila kao što su avioni. Prikazano na slici 1.2.

U ovom radu bit će prikazana izrada drona koji je vrsta rotokoptera te za savladavanje sile teže koristi četiri propelera koji generiraju silu uzgona potrebnu za letenje. Propeleri su raspoređeni simetrično te su pogonjeni elektromotorima bez četkica tzv. BLDC (eng. Brushless DC electric motor) koji crpe energiju iz baterija. Najvažnije obilježje ovakvog tipa letjelica su upravo mogučnosti vertikalnog poljetanja i slijetanja, lebdenje u mjestu i horizontalnog leta te zbog prve tri navedene karakteristike spadaju u VTOL skupinu dronova.

Slika 1.1. VTOL dron[3]

Slika 1.2. CTOL dron[4]

Nakon što je u uvodu napravljena klasifikacija dronova u odnosu na koji način polijeću i slijeću u drugom poglavlju su navedene primjene dronova i njihova korist. U trećem poglavlju bit će opisana izrada same makete, shema spajanja dijelova i električna shema te korišteni elekrični i mehanički dijelovi i njihove karakteristike. Zatim će biti opisan upravljački sustav kojeg čini mikrokontroler (Arduino NANO mikrokontroler koji se temelji na ATmega 328 mikroupravljaču arhitekture AVR) , mjerni (MPU-6050 senzor koji ima integrirani žiroskop i akcelerometar) i izvršni članovi (brushless motori). Nadalje u poglavlju se još opisuje idejno rješenje, algoritam i programska implementacija algoritma te također i alati korišteni u izradi. U četvrtom poglavlju se vrši testiranje makete te testiranje ispunjenja zadatka diplomskog rada. U zaključku ćemo se osvrniti na sve prepreke koje su prilikom ispunjenja zadatka susretnute te ideje za daljnje unaprijeđenje letjelice ne samo softverski (unapređenje upravljačkog programa) nego također i s hardverske strane.

1.1. Zadatak diplomskog rada

U okviru ovog diplomskog rada potrebno je izraditi VTOL drona u 4-propelerskoj konfiguraciji tzv. quadcopter (eng. quadrotor helicopter) te implementirati njegove mogučnosti vertikalnog polijetanja i slijetanja te lebdenja u zraku. Maketa ćemo opremiti senzorima i aktuatorima i mikrokontolerom pomoću kojih će se ostvariti glavne karaktiristike drona koje ga izdvajaju naspram drugih letjelica te implementirati upravljački program u mikrokontroler za ostvarivanje navedenih mogučnosti.

Izrada sustava je podjeljena u 5 faza. Prva faza je pravljenje mehaničkog ustroja, tj sastavljanje makete. U drugoj fazi će se izrađivati električki ustroj sustava što znači da se u drugoj fazi izrade ugrađuju svi neophodni električni elementi koji omogućuju uspješan rad makete što čine senzori, aktuatori, mikroupravljač, baterija i ožičenje tih elemenata prema električnoj shemi. U trećoj fazi se definira idejno rješenje algoritma da bi se u četvrtoj fazi izrađivao program na osnovu algoritma te isprogramirao mikroupravljač. U petoj fazi sustav se pušta u rad te se testira njegov i ispravljaju pogreške u kodu.

2. PRIMJENA I RAZVOJ QUADCOPTERA

U ovom poglavlju opisan je razvoj i upotreba dronova. Ova cjelina je podjeljena na tri potpoglavlja. U prvom potpoglavlju govori se općenito o dronovima i njihovoj primjeni. U drugom se potpoglavlju govori o teoriji letenja i u trećem potpoglavlju o zakonskoj regulativi koja je uvedena.

2.1. Općenito o dronovima

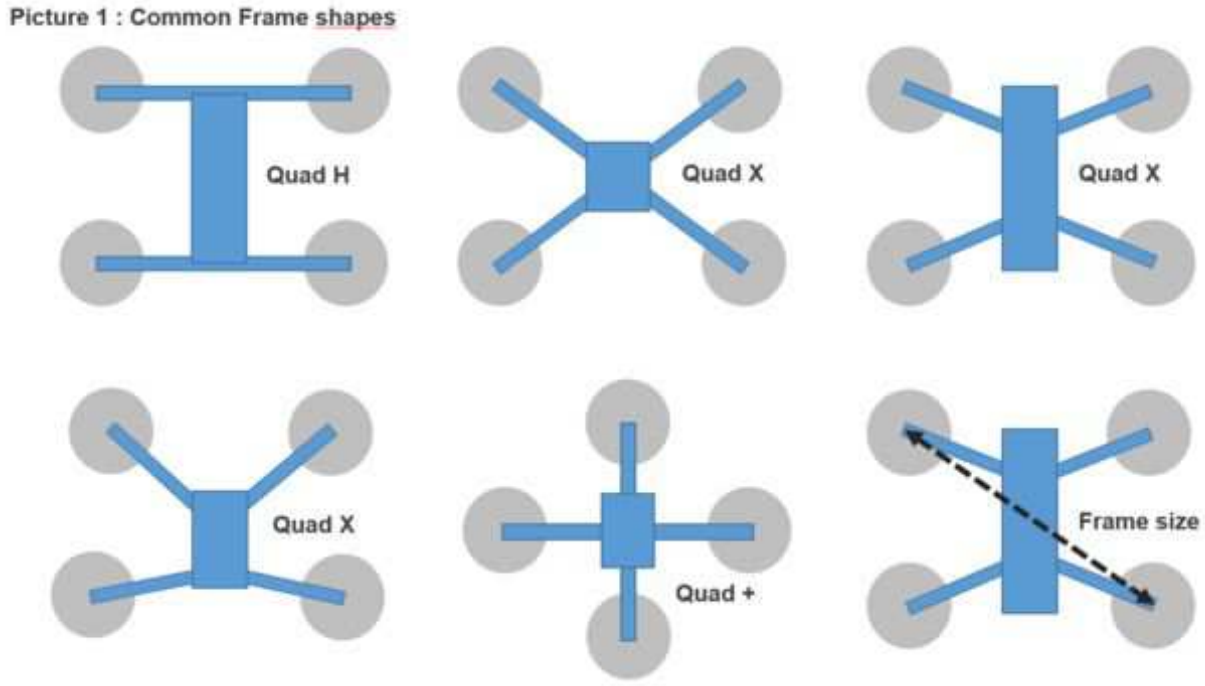
Dronovi su bespilotne letjelice nastale u težnji da se smanji mogučnost ozljede pilota prilikom leta u opasnim uvjetima te smanjenje troškova upotrebe helikoptera i ostalih letjelica jer razvojem tehnologije sve se više razvijaju i bespilotene letjelice te možemo reći tako da se i pronalaze svakodnevno nove primjenje tih letjelica. U današnje vrijeme osim za vojne svrhe sve se više upotrebljavaju i u civilne svrhe.

Neke od mogučnosti primjene u civilne svrhe:

1. Nadzor
    1.1. Promatranje širenja požara
    1.2. Poplave
    1.3. Granice
    1.4. Širenje zagađenja
    1.5. Patrole na moru
    1.6. Traženje unesrečenih
    1.7. Nadzor usjeva
2. Znanstvena i geološke svrhe
    2.1. 3D mapiranje reljefa
    2.2. Proučavanje vulkana
    2.3. Geodetske svrhe
    2.4. Pročavanje oceana i atmosfere

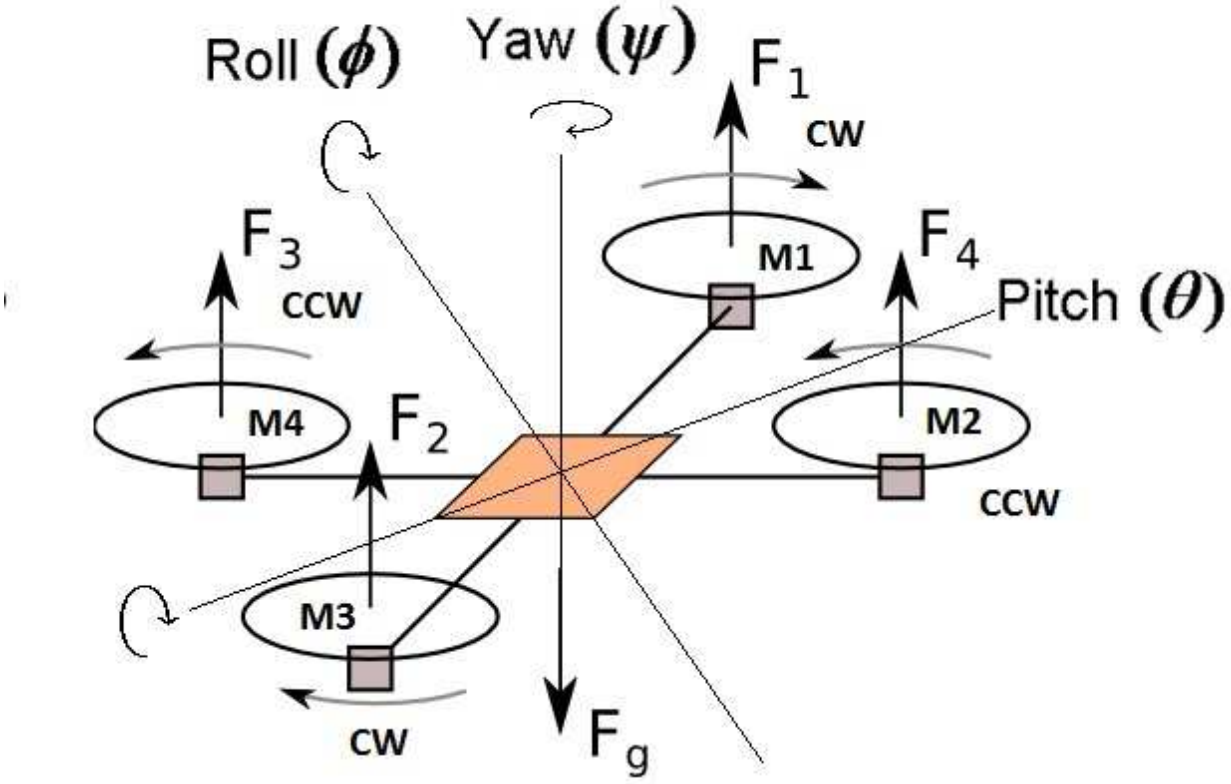
2.2. Teorija letenja 4-propelerskog drona

U ovome potpoglavlju govori se teoriji koja omogučava quadcopteru da leti. Postoji par verzija okvira za quadcoptere (slika 2.2.1), a razlika je u smještaju motora s obzirom na osi oko kojih letjelica može rotirati da bi ostvarila željeni smjer leta. Teoriju ćemo prikazati na x okviru jer u kasnijem poglavju se izrađuje letljelica na malo modificiranom x okviru.



Slika 2.2.1. Najčešći oblici okvira 4-propelerskih dronova[5]

U slučaju okvira u obliku „+“ (okvir dobio ime jer su mu motori smješteni kao matematički znak za zbrajanje) osi rotacije oko osi x i y prolaze kroz motore te za pojedinu rotaciju oko jedne od osi sudjeluju samo dva motora. Za razliku od dronova koji su napravljeni na okviru u obliku „+“ ostali oblici okvira kao što su razne modifikacije okvira u obliku „x“ te okvira u obliku „H“ za rotaciju oko pojedine osi sudjeluju svi motori. Detaljnije možemo vidjeti na idućoj stranici gdje se nalazi slika 2.2.2. te ćemo na primjeru te slike pojasniti kako se ostvaruje let prema naprijed, nazad, lebdenje u mjestu, skretanje lijevi i desno te rotiranje u mjestu u smjeru kazaljke na satu i suprotno od kazaljke na satu.



Slika 2.2.2 Osi rotacije i sile na letjelici[7]

Kao što se može vidjeti iz slike 2.2.2. imamo tri osi rotacije propinjanje (eng. pitch)(**θ**), valjdanje (eng.roll)(**ϕ**) i skretanje (eng. yaw)(**ψ**) te četiri sile uzgona F1, F2, F3 i F4 koji generiraju motori M1, M2, M3, M4, silu teže Fg i dva različita smjera rotacije motora CW (eng. clockwise) smjer kazaljke na satu i CCW (eng. counter clockwise) suprotan smjer od kazaljke na satu. Prvo ćemo objasniti zašto se neki motori okreću u smjeru kazaljke na satu, a drugi suprotno. Bez obzira na te razlike u rotaciji motora svi stvaraju silu uzgona prema gore tj. suprotno od djelovanja sile teže. Potreba za rotaciju dva motora suprotno od drugog para motora je ta što inače letjelica ne bi bila uporabljiva jer rotacijom motora nastaje sila momenta koja se opire okretanju motora te bi u slučaju rotacije sva četiri motora u istom smjeru letjelica bi nekontrolirano rotirala oko **ψ** osi. Da bi se to spriječilo motori su razmješteni tako da zbroj momenta pri istoj brzini rotacije motora teži u nulu (nikad ne može biti nula jer motori zbog konstrukcije nikad ne mogu biti posve identični te će uvijek postojati jako mala rotacija u jednu stranu). Naravno u slučaju da letjelicu želimo rotirati oko **ψ** osi onda ćemo u ovisnosti u kojem smjeru je želimo rotirati motore koji se okreću suprotno od željene rotacije povečati broj okretaja, a onima koji rotiraju u smjeru željene rotacije smanjiti broj okretaja. U koliko želimo da letjelica ide prema naprijed/nazad tada ona rotira oko **θ** osi. Za ostvarivanje rotacije oko osi **θ** tj. da bi smo s letjelicom letjeli prema naprijed potrebno je prednjim

motorima M1 i M4 smanjiti broj okretaja, a stražnjim motorima M2 i M3 povečati. Kao što se može vidjeti na slici 2.2.2 imat ćemo dva motora (jedan rotira CW, a jedan CCW) naprijed koji rotiraju manjom brzinom i odzad 2 motora (jedan rotira CW, a jedan CCW) većom brzinom te opet imamo što se tiče rotacije oko osi ψ vektori momenata motora koji kad bi se zbroji težili bi u nulu. Za kretanje unazad samo obrnemo situaciju od kretanja prema naprijed te u ovome slučaju prednjim motorima M1 i M4 povečamo broj okretaja, a zadnjim motorima M2 i M3 smanjimo. Ukoliko želimo s letjelicom skretati lijevo/desno moramo letjelicu zarotirati oko osi ϕ. Za ostvarivanje navedenog kretanja opet moramo pogledati sliku 2.2.2. te ćemo da bi skretali u lijevo povečati broj okretaja motorima M1 i M2, a smanjiti broj okretaja motora M3 i M4. Za kretanje u desno samo je potrebno obrniti situaciju. Što se tiće momenata oni opet teže u nulu jer imamo parove motora koji generiraju momente približno istoga iznosa, ali suprotnih smjerova djelovanja. Sada ćemo objasniti kako letjelica ostvaruje poljetanje, slijetanje te lebdenje. Gledajući sliku 2.2.2. možemo zaključiti da ukoliko želimo poletjeti moramo djelovati većom silom nego što je iznos sile teže, ali u suprotnom smjeru. Svaki motor generira uzgon te smo sile uzgona motora označili s F1, F2, F3 i F4 te možemo zaključiti da za poljetanje vrijedi formula (2-1).

$$F1+F2+F3+F4 > Fg \tag{2-1}$$

Za slijetanje vrijedi da zbroj uzgona mora biti manji od iznosa sila teže te za slijetanje vrijedi formula (2-2).

$$F1+F2+F3+F4<Fg \tag{2-2}$$

Za lebdenje zbroj uzgona mora biti jednak iznosu sile teže te vrijedi formula (2-3).

$$F1+F2+F3+F4=Fg \tag{2-3}$$

2.3. Zakonska regulativa

Razvojem elektronike smanjila se cijena izrade dronova i oni su se počeli sve više upotrebljavati u civilne svrhe. Sve većom dostupnošću dronova civilima tako su zabilježene i razne nezakonite uporabe dronova diljem svijeta gdje su bili čak i ugroženi ljudski životi, ali i prekršeni zakoni i ugrožena privatnost pojedinaca. Kako bi se spriječila zloupotreba bespilotnih letjelica države su počele uvoditi razne regulative i zakone o istima. Zakoni su doneseni kako bi se prvenstveno zaštitili ljudi zbog toga što bi im mogao biti ugrožen život. Budući da su postali lako dostupni te su također mogu i za malo novaca samostalno izraditi postoji mogučnost invazije zračnog prostora ili upotreba unutar zabranjenje zone leta za sve neregistrirane letljelice (aerodromi).

U Narodnim novinama broj 49/15 od 6. svibnja 2015. godine objavljen je Pravilnik o sustavima bespilotnih zrakoplova koji stupa na snagu osmog dana nakon objave. Tim Pravilnikom, kako je navedeno u članku 1, propisuju se opći, tehnički i operativni uvjeti za sigurnu uporabu bespilotnih letjelica, sustava bespilotnih zrakoplova i zrakoplovnih modela te uvjeti kojima moraju udovoljavati osobe koje sudjeluju u upravljanju tim zrakoplovima i sustavima. Odredbe Pravilnika primjenjuju se na sustave bespilotnih zrakoplova, operativne mase (ukupna masa u trenutku polijetanja) do i uključujući 150 kg koji se koriste na području Republike Hrvatske. Odredbe ovog Pravilnika ne odnose se na sustave bespilotnih zrakoplova kada se koriste za državne aktivnosti, na one koji ne mogu postići kinetičku energiju veću od 79 J te kada se koriste u zatvorenom prostoru. Nadležna institucija za izdavanje odobrenje i općenito regulacije svih operacija je Hrvatska agencija za civilno zrakoplovstvo - HACZ.

Prije izvođenja letačkih operacija, svaku letjelicu je potrebno propisno označiti:

1. Bespilotni zrakoplov koji se koristi za izvođenje letačkih operacija kao i zrakoplovni model operativne mase veće od 5 kg mora biti označen identifikacijskom negorivom pločicom.
2. Za bespilotne zrakoplove operativne mase ispod 5 kg koji se koriste u letačkim operacijama prihvatljivo je da se bespilotni zrakoplov označi identifikacijskom naljepnicom umjesto negorivom pločicom.
3. Označavanje bespilotnog zrakoplova koji se koristi za izvođenje letačkih operacija mora izvršiti operator.
4. Označavanje zrakoplovnog modela mora izvršiti vlasnik.

5. Identifikacijska negoriva pločica ili naljepnica mora sadržavati slijedeće podatke:
    a. Identifikacijsku oznaku bespilotnog modela
    b. Ime, adresu i informacije za kontakt operatora ili vlasnika

6. Identifikacijska negoriva pločica ili naljepnica mora biti odgovarajuće veličine koja omogućuje jasnu identifikaciju i mora biti postojanim načinom pričvršćena.
7. Operator sustava bespilotnog zrakoplova u letačkim operacijama ili vlasnik zrakoplovnog modela dužan je zamijeniti identifikacijsku negorivu pločicu ili naljepnicu čim dođe do promjene podataka ili se ona ošteti do neprepoznatljivosti ili izgubi.
8. Identifikacijsku oznaku za bespilotni zrakoplov koji se koristi za izvođenje letačkih operacija kategorije D dodjeljuje Hrvatska agencija za civilno zrakoplovstvo.
9. Identifikacijsku oznaku za zrakoplovni model određuje vlasnik, odnosno operator za bespilotni zrakoplov koji se koristi za izvođenje letačkih operacija kategorije A, B i C, na način da ne smije započinjati velikim latiničnim slovom „D“

Dronovi su podijeljeni s obzirom na operativnu masu u tri klase:

1. Klasa 5: do 5 kilograma
2. Klasa 25: od 5 kilograma do 25 kilograma
3. Klasa 150: od 25 kilograma do 150 kilograma

Pravilnikom [ ] su još i klasificirana područja letenja:

1. Klasa I – Područje u kojem nema izdignutih građevina ili objekata i u kojem nema ljudi, osim rukovoditelja i osoblja nužnog za letenje.
2. Klasa II – Područje u kojem postoje pomoćni gospodarski objekti ili građevine koje nisu namijenjene za boravak ljudi i u kojem nema ljudi, osim rukovoditelja i osoblja koje je nužno za letenje. Dozvoljen je samo povremeni prolazak, bez zadržavanja, ljudi kroz područje (biciklisti, šetači i sl.).
3. Klasa III – Područje u kojem postoje građevine ili objekti primarno namijenjeni za stanovanje, poslovanje ili rekreaciju (stambene zgrade, stambene kuće, škole, uredi, sportski tereni i sl.).
4. Klasa IV – Područje uskih urbanih zona (središta gradova, naselja).

Prije početka leta potrebno je i kategorizirati isti. Kategorizaciju se vrši tako što se određuje razina rizika kojom bi izvođenje leta predstavljalo za okolinu i podjeljeno je u četiri kategorije A, B, C i D. U ovoj kategorizaciji kategorija A predstavlja najmanju opasnost, a kategorija D najveću opastnost. U dolje navedenoj tablici 2.3.1.[6] koja prikazuje kategorizaciju letačkih misija u ovisnosti o klasi letjelice i kategorije područja letenja propisuju se i određeni zahtjevi koje se treba ispuniti.

Klasa sustava bespilotnog zrakoplova	Klasa područja izvođenja letenja			
	I Neizgrađeno područje	II Izgrađeno nenaseljeno područje	III Naseljeno područje	IV Gusto naseljeno područje
5 OM < 5 kg	A	A	B	C
25 5 ≤ OM < 25 kg	A	B	C	D
150 25 ≤ OM ≤ 150 kg	B	C	D	D

Tablica 2.3.1. Kategorija letačkih operacija[7]

Obavezna dokumentacija koju prilikom izvođenja letačkih operacija mora imati operater sa sobom:

1. letački priručnik ili upute za upotrebu sustava bespilotnog zrakoplova,
2. izvornik ili ovjerenu presliku odobrenja za izvođenje letačkih operacija, ako je primjenjivo
3. policu osiguranja, kada je primjenjivo,
4. za kategorije A i B letačkih operacija, dokaz o poznavanju primjenjivih zrakoplovnih propisa, psihofizičkoj sposobnosti i osposobljenosti za upravljanje tipom/modelom sustava bespilotnog zrakoplova,
5. za kategorije C i D letačkih operacija:
    a. Operativni priručnik
    b. Dokaz o osposobljenosti za upravljanje sustavom
    c. Pilotsku dozvolu ili potvrdu o položenom teorijskom ispitu iz poznavanja pravila letena koje provodi Agencija
    d. Dokaz o psihofizičkoj sposobnosti za upravljanjem sustavom bespilotnih letjelica

Let bespilotnom letjelicom bi se trebao odvijati danju. Prije leta trebalo bi provjeriti ispravnost cijelog sustava, imali i oštećenja i na odgovarajući način pričvrstiti svu opremu i teret na bespilotnoj letjelici kako ne bi došlo do ispadanja. Meteorološki i ostali uvjeti na području letenja ne smiju utjecati na sigurnost izvođenja leta. Tijekom leta treba osigurati sigurnu udaljenost bespilotne letjelice od ljudi, životinja, objekata, vozila, plovila, drugih zrakoplova, cesta, željezničkih pruga, vodenih putova ili dalekovoda koja ne smije biti manja od 30 metara, dok minimalna udaljenost od skupine ljudi jeste 150 metara. Let bespilotne letjelice uvijek mora biti unutar vidnog polja rukovatelja i na udaljenosti do 500 metara od rukovatelja. Potrebno je osigurati da se let bespilotne letjelice odvija izvan kontroliranog zračnog prostora i na udaljenosti najmanje 3 kilometra od aerodroma i prilazne ili odlazne ravnine aerodroma. Tijekom leta zabranjeno je izbacivati predmete iz ili s bespilotne letjelice. [8]

Ukratko, najvažnija pravila kojih bi se trebali pridržavati za izvođene letačkih operacija:

1. Operator je odgovoran za svaki let
2. Potrebno odobrenje za let u zabranjenim područjima
3. Provjeriti dron od oštećenja prije svakog leta
4. Držati razmak od drugih vozila i ljudi (30 m)
5. Dron uvijek držati u vidokrugu operatera
6. Paziti na pravila o privatnosti
7. Operater je odgovoran za izbjegavanje sudara
8. Dostaviti izjavu HACZ-u za svaku letačku operaciju

.

3. MAKETA MULTIROTORSKE LETJELICE U 4-PROPELERSKOJ KONFIGURACIJI

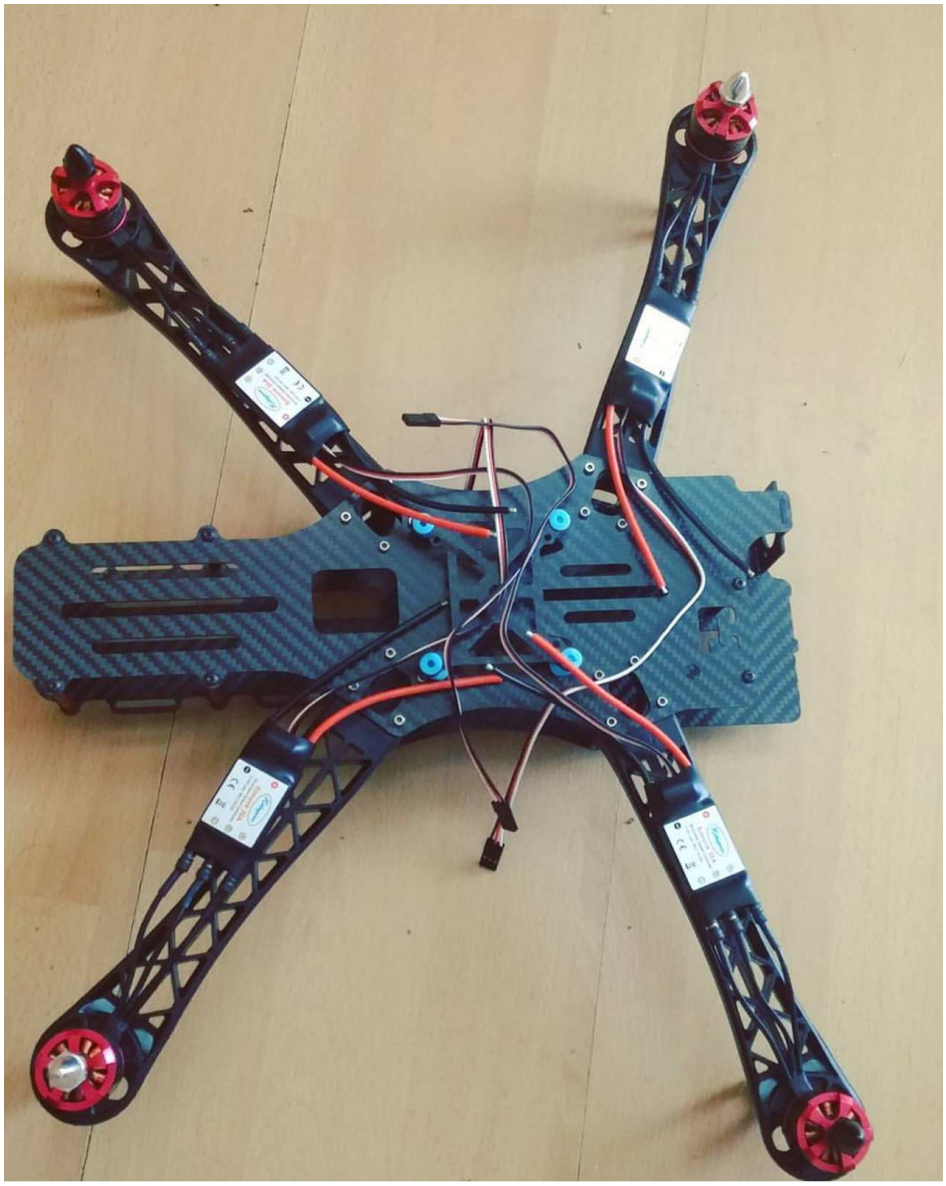
U ovom poglavlju bit će detaljnije opisan postupak izrade makete. U prvom potpoglavlju ovoga poglavlja govori se o ciljevima izrade makete, projektiranje makete da zadovolji zadane ciljeve. U drugom potpoglavlju opisana je izrada same letjelice i njezini elekronički i mehanički dijelovi. U trećem potpoglavlju se prikazuje blokovska shema i električna shema makete te je opisan algoritam upravljana, alati korišteni u izradi te implementacija upravljačkog algoritma na mikroupravljač.

3.1. Projektiranje makete

Pri projektiranju bilo kojeg projekta potrebno je prvo odrediti ciljeve koje bi projekt trebao ispunjavati. Ciljeve koje bi maketa trebala ispunjavati:

1. Dimenzija
2. Cijena
3. Materijal
4. Težina
5. Snaga motora
6. Kapacitet baterije
7. Veličina propelera
8. Upravljanje pomoću mikroupravljača

Maketa letjelice bi trebala biti odgovarajuće veličine za lakšu montažu svih potrebnih dijelova tako da je korišten okvir reptile f500 prikazan na slici 3.1.1.



Slika 3.1.1. Okvir za montiranje dijelova reptile f500

Idući cilj je cijena te je pokušano da maketa bude što jeftinija, a da se ostali ciljevi mogu ispuniti. Slijedeća dva cilja idu zajedno jer da bi smo nešto napravili čvrsto često podrazumijeva da je to i teško no što je važno za letjelice općenito važno je također i za ovu maketu, a to je da masa bude što manja moguća. Da bi se ispunila oba cilja koristi se okvir od karbonskih vlakana jer je to materijal koji se odlukuje velikom čvrstoćom što nam je potrebno da u slučaju pada letjelica bude što više čitava, ali i malom masom što nam opet nadovezuje se na iduće ciljeve. Manjom masom cijele letjelice mogu se uzeti manje snažni motori te nam to također omogučiti i veću autonomnost tj. duže letjenje bez da moramo uzimati bateriju većeg kapaciteta. S obzirom na snagu motora

motora i težinu sveukupne letjelice uzimamo i odgovarajuću veličinu propelera. Također maketa mora biti i upravljanja s mikroupravljačam. Mikroupravljač mora zadovoljavati prethodne ciljeve te je odabran zbog svojih dimenzija Arduino Nano s čipom Atmega 328P.

3.2. Izrada makete

3.2.1. Električni i mehanički dijelovi letjelice i upravljača letjelice

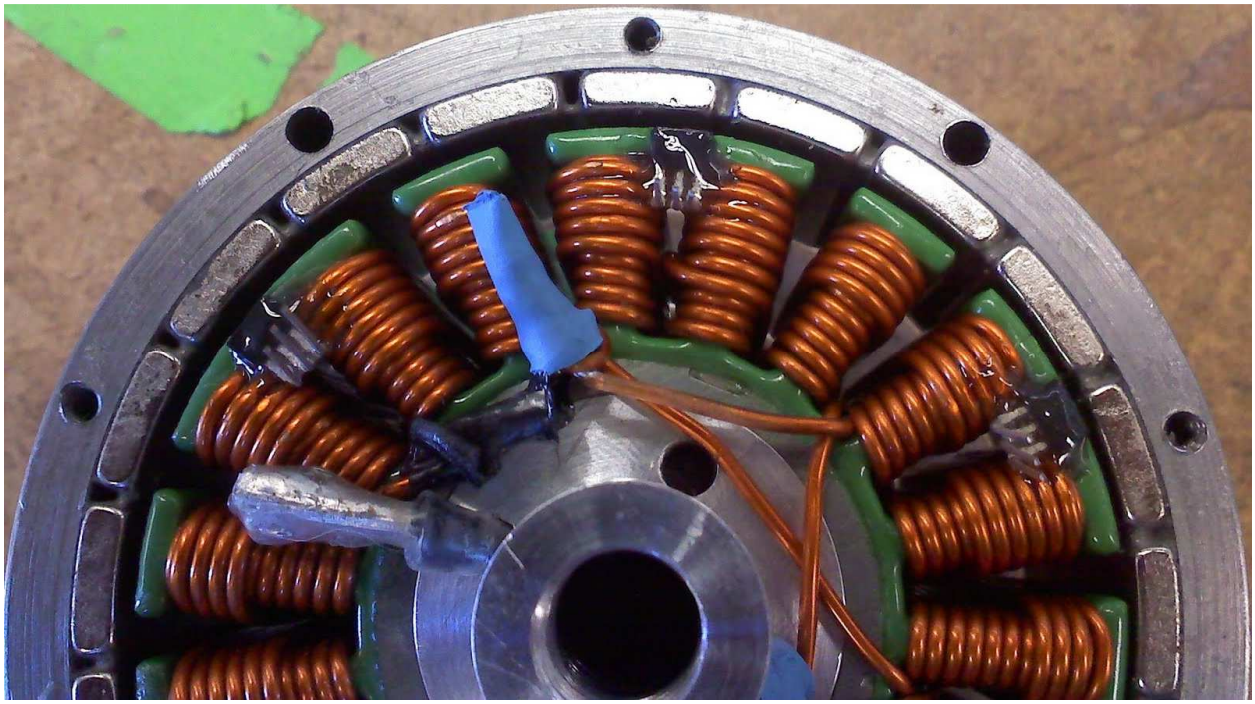
U ovome poglavlju opisana je izrada i korišteni dijelovi. Dimenzija letjelice bez montiranih propelera je 430mm x 276mm x 90mm. U tablici 3.2.1.1 navedeni su svi dijelovi korišteni te njihova težina i količina.

Dio	Količina	Težina
Karbonski okvir	1	350g
BLDC motor	4	55g
ESC	4	10g
Li-poly baterija 10000 maH	1	650g
Karbonski propeleri 1045	4	10g
Mpu-6050	1	5g
Arduino Nano	1	5g
Bluetooth HC-05	1	5g
Težina ukupno:		**1315g**

Tablica 3.2.1.1. Dijelovi za izradu letjelice

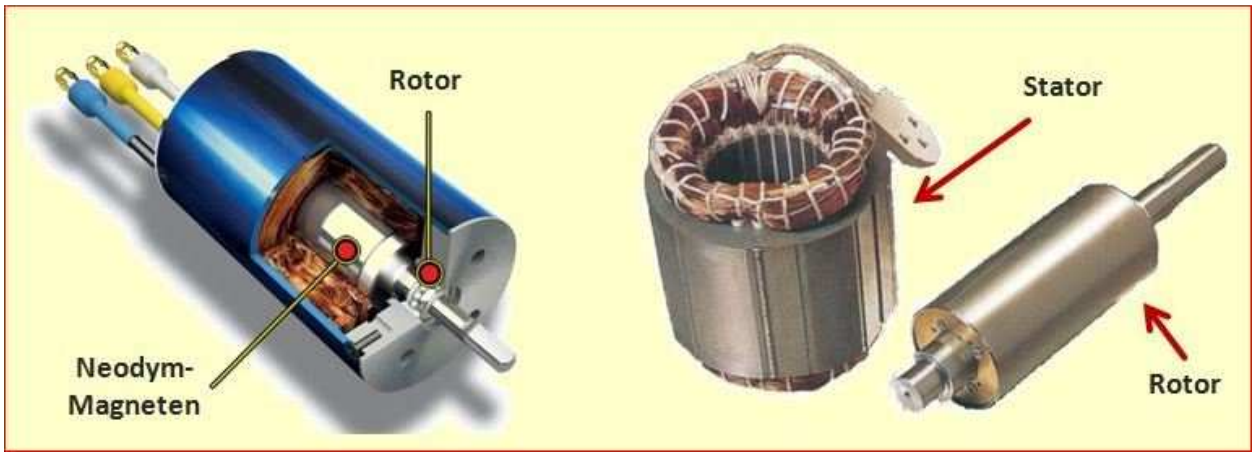
Motori korišteni na letjelici su električni motori bez četkica te su po tome i nazvani BLDC (eng. brushless DC motor) tj. sinkroni motori koji moraju imati poseban kontroler koji će istosmjernu struju pretvoriti u izmjeničnu za pojedinu fazu motora. Svaki motor ima tri faze koje naizmjenično uključuju i isključuju. Glavnu vrijednost koju moramo obratiti na njima je Kv. Tu vrijednost ne bi trebali pomijesati s oznakom za kiloVolt(kV). Ova vrijednost na motoru nam govori koliko će okretaja u minuti motor ostvariti priključkom na napon od 1V u praznom hodu te će primjerice motor s 980Kv priključen na napon od 10V imati 9800 okretaja u minuti. Još jedva važna komponenta je broj magneta korištenih u rotoru jer što veći broj magneta znači manji broj okretaja,

ali i veći moment te može pogoniti veće propelere. Možemo ih susresti u hard diskovima i ventilatorima za hlađenje računala. Odlikuju ih velika snaga s obzirom na svoju veličinu. Na slici 3.2.1.1. je prikazana unutrašnjost jednog takvog motora.



Slika 3.2.1.1. Unutrašnjost BLDC motora[9]

Na slici možemo vidjeti namote koji čine stator motora i magnete koji su na rotoru motora. Postoje dvije izvedbe ovih motora tzv. bačvasti (eng. inrunner) gdje su namotaji na vanjskoj strani te rotor s magnetima unutar namotaja kao što je prikazano na slici 3.2.1.2. Ove motore odlikuje veliki Kv što znači da imaju mali moment. U izradi maketa koriste se za izradu modela auta i brodova gdje pomoću reduktora povečaju izlaznu snagu.



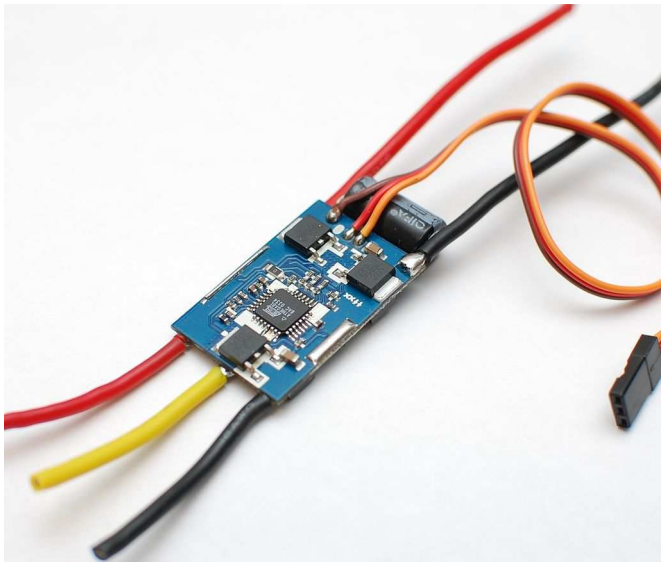
Slika 3.2.1.2 Presjek inrunner BLDC motora[10]

Drugi oblik BLDC je tzv. disk (eng. outrunner) takvi motori imaju manji Kv za razliku od inrunner te također i veći moment sile ostvaruju. Za razliku od inrunner motora kojima je stator okolo rotora u outrunner motorima je rotor oko statora te na slici 3.2.1.1. možemo vidjeti primjer takvog motora te na slici 3.2.1.3. motor koji je korišten u izradi makete.

Slika 3.2.1.3 Motor korišten u maketi [11]

Da bi se mogao koristiti BLDC motor potreban mu je kontroler koji uključuje i isključuje faze motora tzv. ESC (eng. electronic speed control) te muje svrha također i da upravljabrzinom okretaja samog motora. Na slici 3.2.1.4. možemo vidjeti unutrašnjost jednog ESC. U maketi je korišten ESC sa Simonk upravljačkim softverom koji je trenutno najbolji softver za ESC i oznake 30A što znači da može kontrolirati motor koji najviše koristi 30 ampera. Također korišteni ESC ima i BEC (eng. battery eliminator circuit) što nam omogučava da svu elektroniku koja radi na na drugom naponu ne moramo koristiti drugog napona za napajanje druge elektronike. U našem

slučaju koristimo napon od 11.1V dok npr. Mikroupravljač koristi 5V. BEC smanjuje napon baterije na 5V te smo upravo i iskoristili tu mogučnost u projektu da napajamo mikroupravljač i ostale komponente kojima je za rad potreban napon od 5V.



Slika 3.2.1.4. ESC [12]

ESC da bi regulirao broj okretaja motora dobiva od glavnog upravljača impuls u trajanju od 1000 uS do 2000 uS tj. koristi impuls kao i servo motori no za razliku od njih gdje dužina impulsa se pretvara u kut zakreta servo motora u slučaju BLDC motora odgovara brzini okretanja gdje je impuls u trajanju od 1000 milisekundi označava brzinu od 0 okretaja u minuti, a impuls od 2000 milisekundi maksimalnoj brzini okrataja koju motor može postići pri određenom naponu i opterečenju.

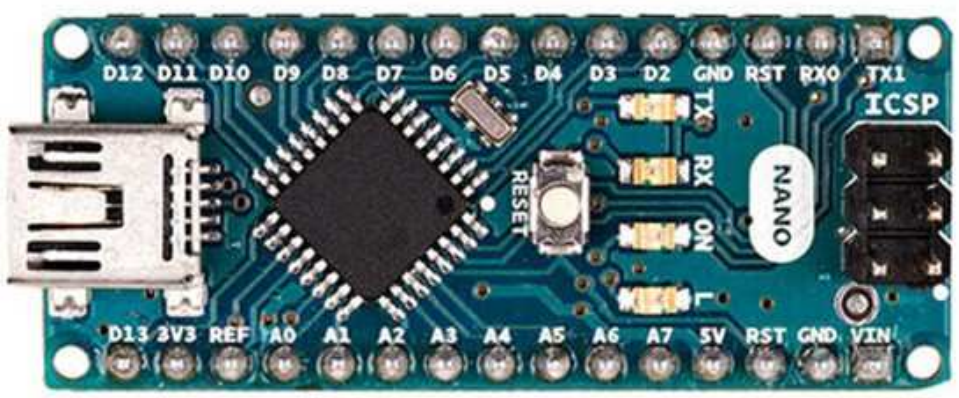
Da bi letjelica mogla ostvariti uzgon osim motora potrebni su i propeleri. Budući da imamo 2 para motora koji se okreću suprotno također nam i trebaju propeleri za te motore koji će ostvarivati uzgon prema gore. Na slici 3.2.1.5. pokazani su propeleri koji se koriste na letjelici. Prema smjeru okretanja imamo lijeve i desne propelere. Ako se propeler okreće suprotno od kazaljke na satu onda je desni, a ako se okreće u smjeru kazaljke na satu onda je lijevi . Također propelere možemo podijeliti na vučne i potisne. Vučni se nalaze ispred motora, a potisniiza motora. Za našu konfiguraciju potrebna su nam četiri vučna propelera te dva od njih moraju biti lijeva, a druga dva desna. Korišteni su APC 10x45 propeleri od karbonskih vlakana. Prvi broj označava promjer propelerima, drugi kut uvijanja. Obje veličine su u inčima. Kut uvijanja nam govori koliki put će prevaliti točka na propeleru s radijosom R tijekom jednog puno okretaja propelera. Putanja koju točka prevali je u obliku helikoide.

Slika 3.2.1.5. Prikazani su jedan lijev i jedan desni propeler[13]

Korišteni mikroupravljač je Arduino Nano prikazan na slici 3.2.1.6. te specifikacije su mu navedene u tablici 3.2.1.2.

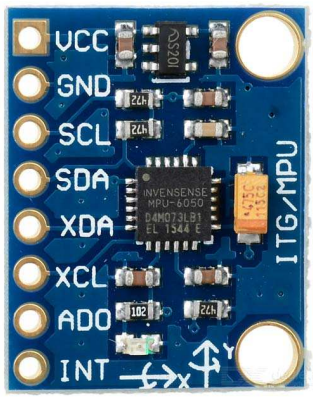
Microcontroller	Atmel ATmega328
Operating Voltage (logic level)	5v
Input Voltage (recommended)	7v-12v
Input Voltage (limits)	6v-20v
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40mA
Flash Memory	32 KB of which 2 KB used by bootloader
SRAM	2KB
EEPROM	1KB
Clock Speed	16MHz
Dimensions	0.73" x 1.70"
Length	45 mm
Width	18 mm
Weight	5g

Tablica 3.2.1.2 Specifikacija Arduino Nano[14]



Slika 3.2.1.6. Arduino Nano

Za letenje quadcoptera potreban nam je žiroskop i akcelerometar jer pomoću njih možemo znati položaj tj. kut letjelice spram pojedinih osi rotacije letjelice. U izradi makete korišten je čip MPU- 6050[16] koji objedinjuje na jednoj pločici i žiroskop i akcelerometar. Na slici 3.2.1.7. je prikazan MPU-6050. Za komunikaciju s mikrokontrolerom koristi se I2C komunikacija.

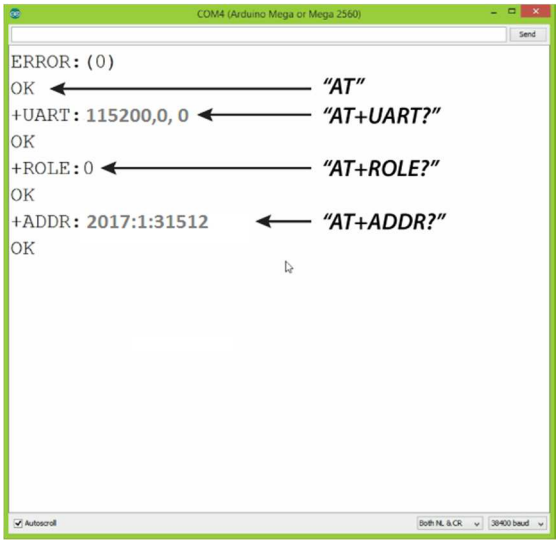


Slika 3.2.1.7. MPU-6050[17]

Za komunikaciju između daljinskog upravljača i upravljačkog sklopa na letjelici korišten je HC-05 bluetooth modul prikazan na slici 3.2.1.8. gdje na daljinskom upravljaču radi kao SLAVE, a na letjelici kao MASTER.

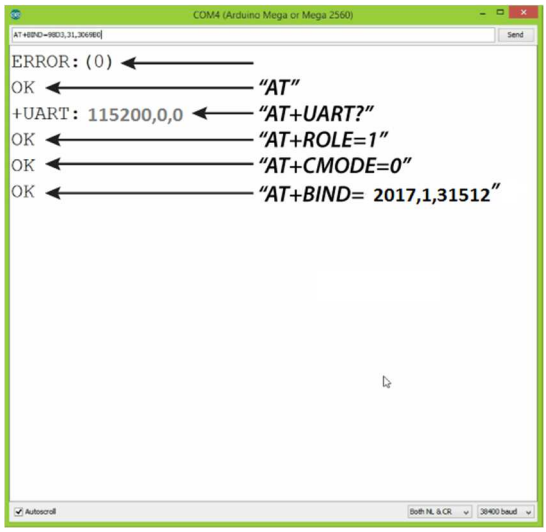
Slika 3.2.1.8. HC-05 Bluetooth module

Za konfiguriranje modula potrebno je koristiti AT naredbe te modul „natjerati“ u AT mod. Da bi modul natjerali u „AT“ mod potrebno je držati stisnut tipkalo na modulu i spojiti na napajanje modul. Na slikama 3.2.1.9. i 3.2.1.10. pokazane su naredbe koje treba putem serijske komunikacije poslati modulu u ovisnosti koji mod želimo da modul radi.



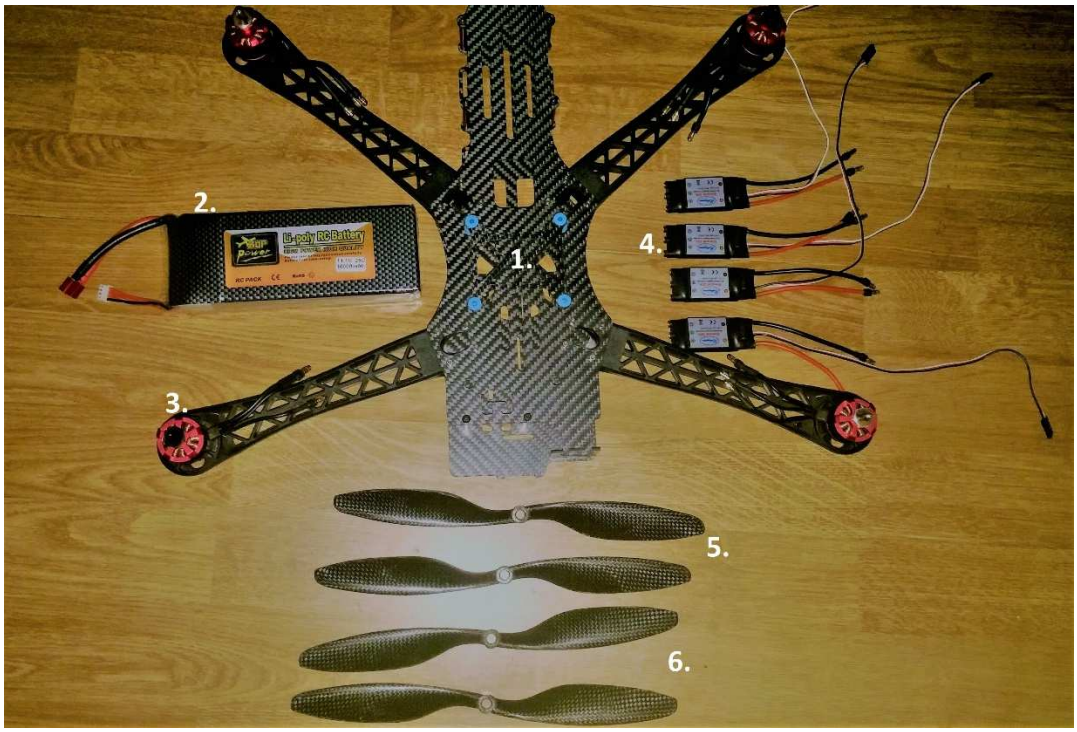
Slika 3.2.1.9. Konfiguracija HC-05 kao slave

Potrebno je upisati naredbe koje su označene „“. Role=0 je oznaka za slave mod, a naredbnom „AT+ADDR?“ smo dobili adresu modula, te nam je to potrebno da se master modul može spojiti samo na slave navedenom adresom. Svaki modul ima svoju adresu te nam to čak i sigurnosna značajka da se netko drugi ne može putem bluetooth spojiti na letjelicu.



Slika 3.2.1.10. Konfiguracija HC-05 kao master

Za napajanje cijele letjelice koristi se Li-poly baterija 3s 10 000 maH. Na slici 3.2.1.11. označena je brojkom 2. Oznaka 3s znači da ima 3 čelije te tako možemo izračunati nazivni napon paketa ove baterije, a to je 11.1V. Brojka 10 000 maH nam govori kapacitet baterije, a govori nam koliko dugo može davati struju određene jakosti. U slučaju ove baterija npr. struju od 1A mogla bi davati 10 sati, a struju od 10A jedan sat. Ostali dijelovi na slici su pod brojem 1 okvir letjelice, brojem 3 označeni su motori, brojem 4 označeni su ESC te brojem 5 i 6 označeni su lijevi i desni propeleri.



Slika 3.2.1.11. Dijelovi letjelice

3.2.2. Programski kodovi za konfiguraciju MPU-6050

4. Programski kod za konfiguraciju MPU-6050

```
void setup_mpu_6050_registers(){
 Wire.beginTransmission(0x68);         //Započinanje komunikacije I2C na adresi 0x68
 Wire.write(0x6B);                     //Slanje zahtjeva na registar 0x6B
 Wire.write(0x00);                     //Postavljanje vrijednosti u registar
 Wire.endTransmission();
 //konfiguriranje min i max vrijednosti akcelerometra (+/-8g)
 Wire.beginTransmission(0x68);
 Wire.write(0x1C);                     //zahtjev za registar 0x1C
 Wire.write(0x10);                     //postavljanje vrijejednosti
 Wire.endTransmission();
 //konfiguriranje žiroskopa (500dps full scale)
 Wire.beginTransmission(0x68);
 Wire.write(0x1B);                     //slanje zahtjeva na registar 0x1B
 Wire.write(0x08);                     //Postavljanje vrijednosti
 Wire.endTransmission();
 }
```

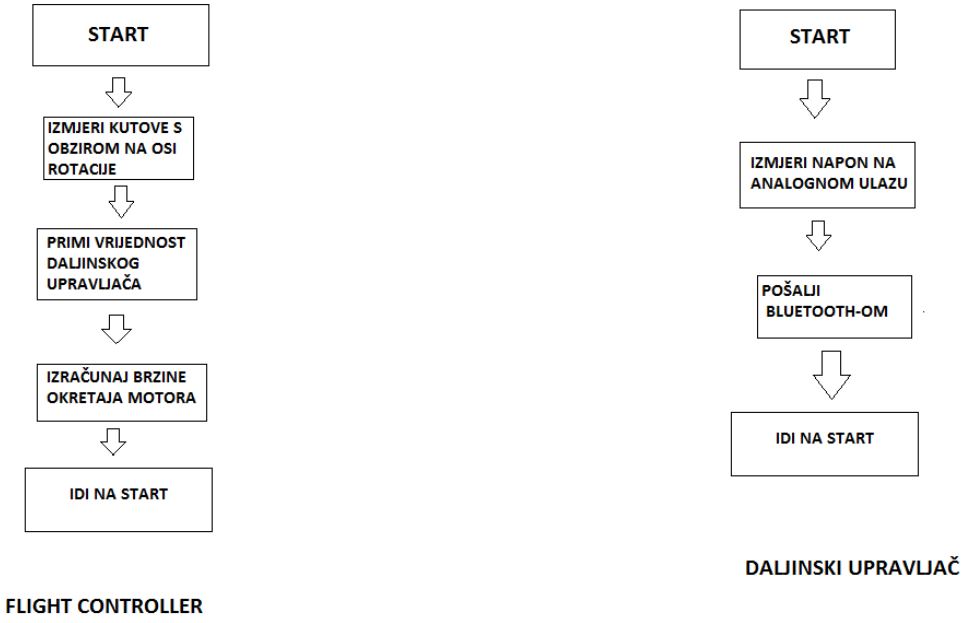
Programski kod za čitanje vrijednosti žiroskopa i akcelerometra MPU-6050

```
void read_mpu_6050_data(){                    //rutina za citanje vrijednosti senzora
 Wire.beginTransmission(0x68);
 Wire.write(0x3B);                     //Slanje zahtjeva za registar 0x3B
 Wire.endTransmission();
 Wire.requestFrom(0x68,14);                   //Zahtjev za 14 bajtova s MPU-6050
```

```
while(Wire.available() < 14);                    //cekamo dok se ne prime svi bajtovi
acc_x = Wire.read()<<8|Wire.read();
acc_y = Wire.read()<<8|Wire.read();
acc_z = Wire.read()<<8|Wire.read();
temp = Wire.read()<<8|Wire.read();
gyro_x = Wire.read()<<8|Wire.read();
gyro_y = Wire.read()<<8|Wire.read();
gyro_z = Wire.read()<<8|Wire.read();
}
```

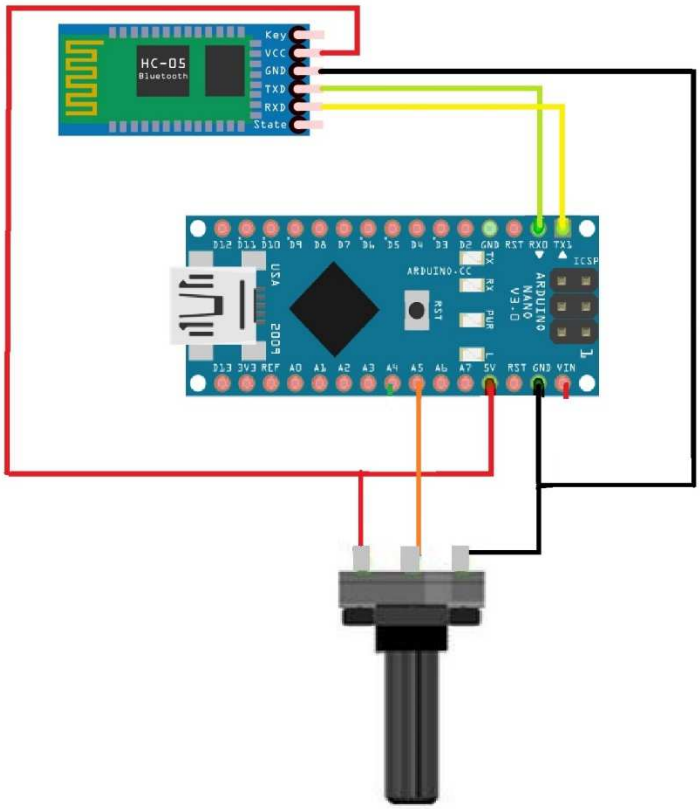
3.3. Implementacija rješenja flight controllera

U ovome potpoglavlju implementirat ćemo na osnovu blokovske sheme kod za mikroupravljač da bi se ostvarilo lebdenje letjelice. Također pokazana je i električna shema flight controllera i daljinskog upravljača te njihovi kodovi. Daljinski upravljač služi za povečavanje i smanjivanje broja okretaja motora, tj. on upravlja polijetanjem i slijetanjem letjelice. Na slici 3.3.1. možemo vidjeti blokovsku shemu flight controllera i daljinskog upravljača.

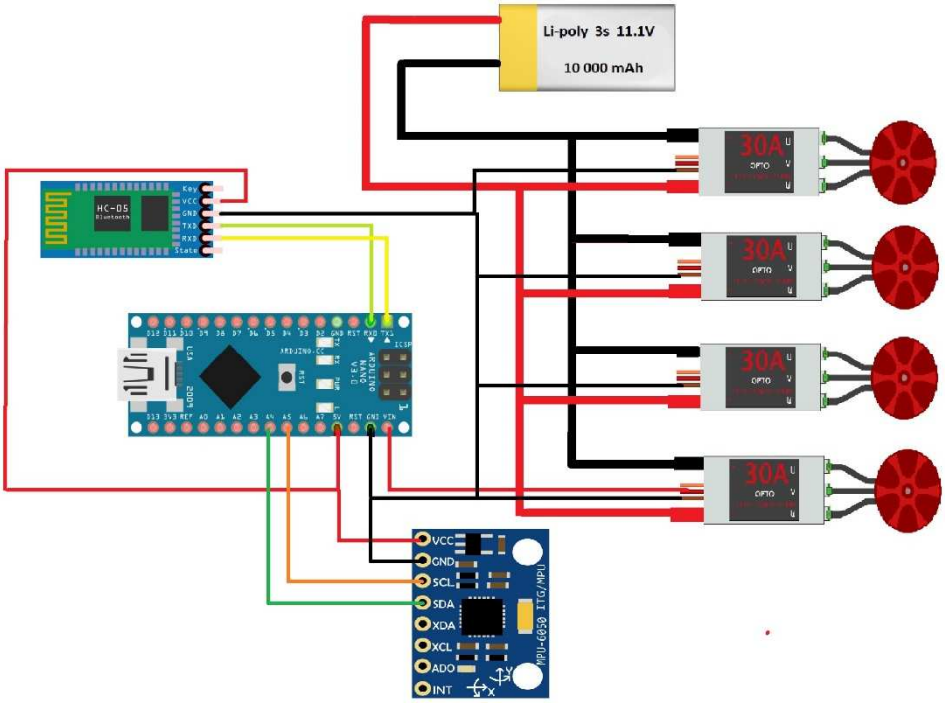


Slika 3.3.1. Blokovska shema cijelog sustava

Na slikama 3.3.2. i 3.3.3. pokazane su elekrične sheme daljinskog upravljača i flight controllera. U prilogu se nalaze programski kodovi za daljinski upravljač i flight controller.

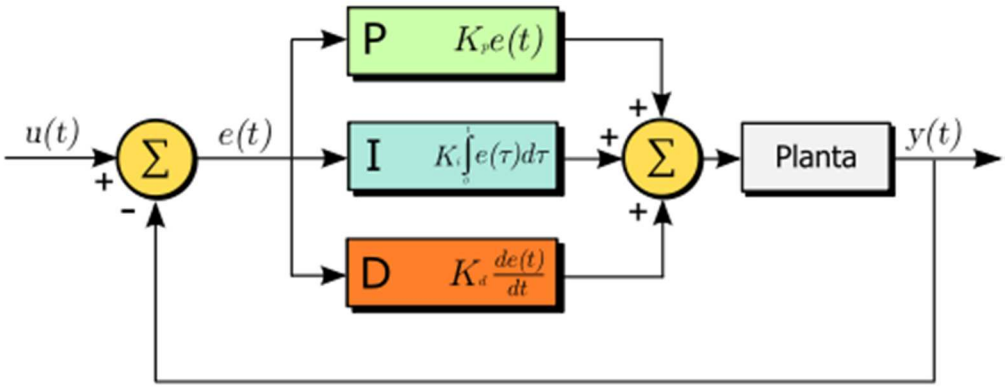


Slika 3.3.2. Električna shema daljinskog upravljača



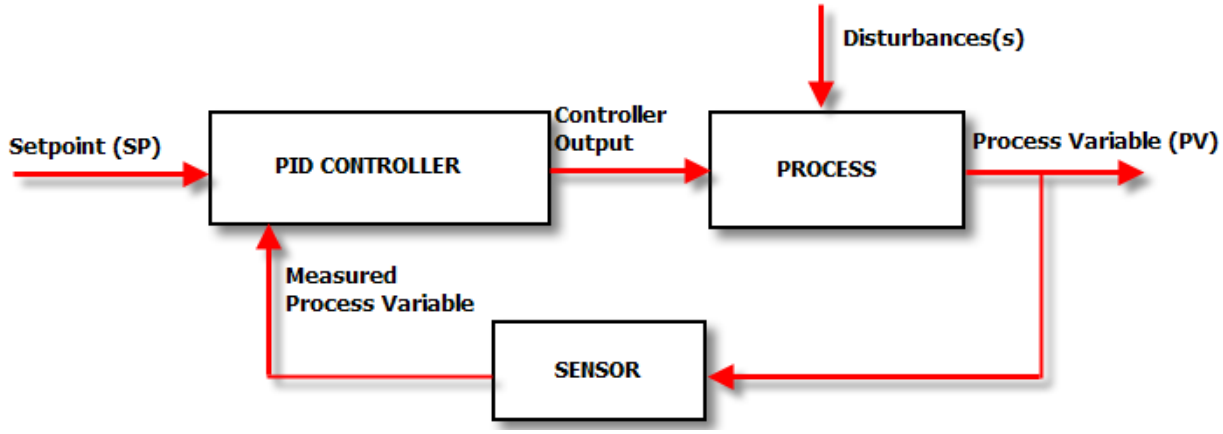
Slika 3.3.3. Električna shema flight controllera

Za algoritam koji se koristi za kontrolu stabilnosti leta je PID (eng. proportional-integral-derivative) algoritam koji je vrlo često korišten za automatizaciju postrojenja u kojem se konstanto potrebno upravljati određenim parametrom u postrojenju. Na slici 3.3.4. prikazan je PID algoritam. Na slici možemo vidjeti oznaku u(t) što nam je ulazna veličina, e(t) je razlika između ulazne veličine i izlazne veličine postrojenja y(t) dakle greška. Kp, Ki, Kd su koeficijenti PID algoritma pomoću kojih algoritam računa utjecaj greške na proporcionalni, integralni i derivativni izlaz koji se zbrajaju i šalju postrojenju.



Slika 3.3.4. PID algoritam [18]

Na osnovu slike 3.3.5. na jednostavnijem primjeru pojasnit ćemo princip rada algoritma.



Slika 3.3.5. Prikaz implementacije PID algoritma u procesu[19]

Princip rada ćemo objasniti na primjeru letjelice i to zbog jednostavnijeg prikaza kontrolu rotacije oko jedne osi npr. pitch osi. Setpoint nam je vrijednost koju želimo postići, a u našem slučaju će to biti kut od 0° stupnjeva nagiba s obzirom na pitch os. Ukoliko kut nije 0° jer je došlo do smetnje npr. vjetar je zapuhao i izbacio iz ravnoteže letjelicu PID algoritam na osnovu koeficijenata Kp,

Ki, Kd računa koliki broj okretaja mora imati pojedini motor te šalje tu vrijednost motorima. Kut se ponovo mjeri i računa se greška s obzirom na trenutno izmjereni kut i željeni kut. Vrlo važno je napomenuti da dva ista postrojenja neće imati iste vrijednosti za Kp, Ki i Kd jer neće biti posve istih karakteristika (nema savršene kopije). Korištena metoda za pronalazak PID koeficijenata je Zigler-Nichols metoda[20]. Kreirali su je John G. Zigler i Nathaniel B. Nichols. Metoda je vrlo dobra u nekim slučajevima, ali nije optimalna za sustve koji ne smiju postići kritične ostilacije jer se mogu jako oštetiti. Metoda određuje koeficijente tako što Ki i Kd izjednačimo s 0, a Kp vrijednost povečavamo dok sustav ne dosegne konstantne oscilacije, zatim izmjerimo period oscilacija i Kp pri kojem se te oscilacije događaju te prema slici 3.3.6. možemo odrediti Ki i Kd. Ta metoda nije precizna, ali možemo približno odrediti koefcijente.

Ziegler–Nichols method			
Control Type	K_p	K_i	K_d
P	$K_u/2$	-	-
PI	$K_u/2.2$	$1.2K_p/T_u$	-
classic PID	$0.60K_u$	$2K_p/T_u$	$K_pT_u/8$
Pessen Integral Rule	$0.7K_u$	$2.5K_p/T_u$	$0.15K_pT_u$
some overshoot	$0.33K_u$	$2K_p/T_u$	$K_pT_u/3$
no overshoot	$0.2K_u$	$2K_p/T_u$	$K_pT_u/3$

Slika 3.3.6. Određivanje vrijednosti PID koeficijenata Zigler-Nichols metodom[21]

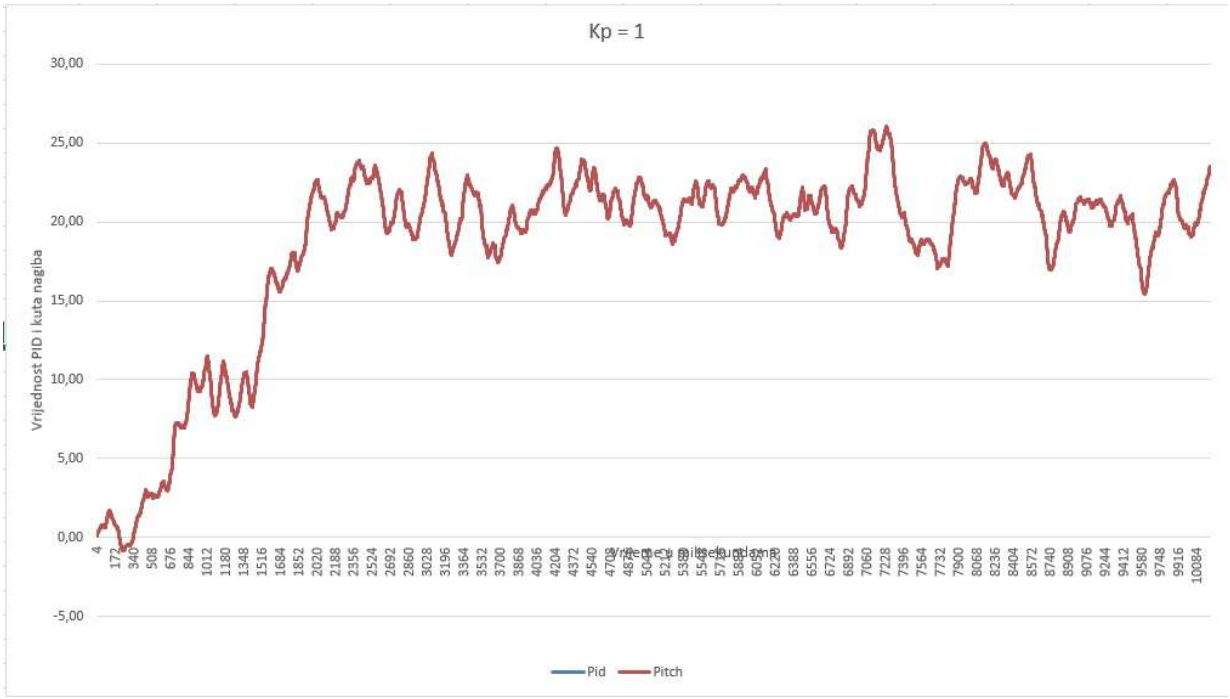
4. EVALUACIJA RJEŠENJA MODELA LETJELICE

U ovome poglavlju ćemo evaluirati rješenje stabilnosti letjelice korištenjem PID regulatora i određivanje Ziegler-Nichols-ovom metodom PID koeficijente. Mjerenja su vršena da su za vrijednosti uzimane iduće vrijednosti PID koeficijenata prikazano u tablici 4.1. Željeni kut u svim mjerenjima s obzirom na pitch os je 0°.

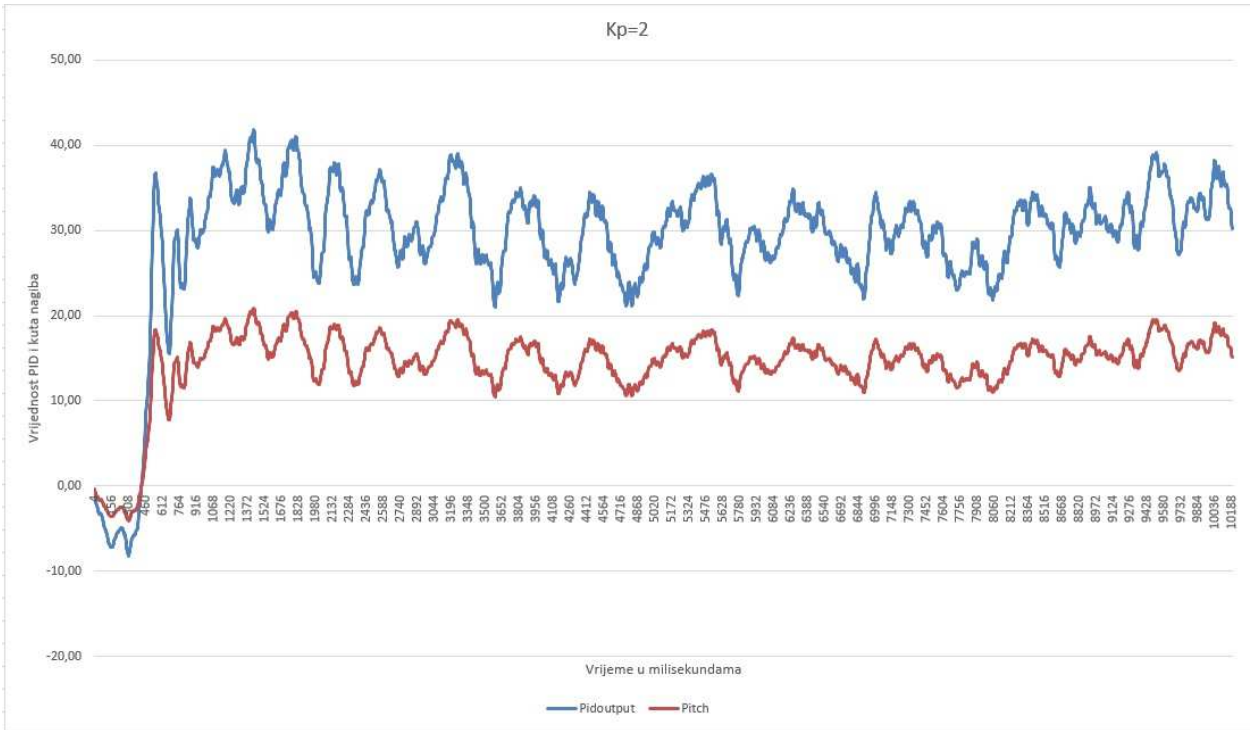
Mjerenje	Kp	Ki	Kd
1.	1	0	0
2.	2	0	0
3.	4	0	0
4.	10	0	0
5.	10	0.001	0
6.	10	0.01	0
7.	10	0.02	0
8.	10	0.05	0
9.	10	0.02	10
10.	10	0.02	20
11.	10	0.02	40
12.	10	0.02	80

Tablica 4.1. Parametri korišteni u testiranju

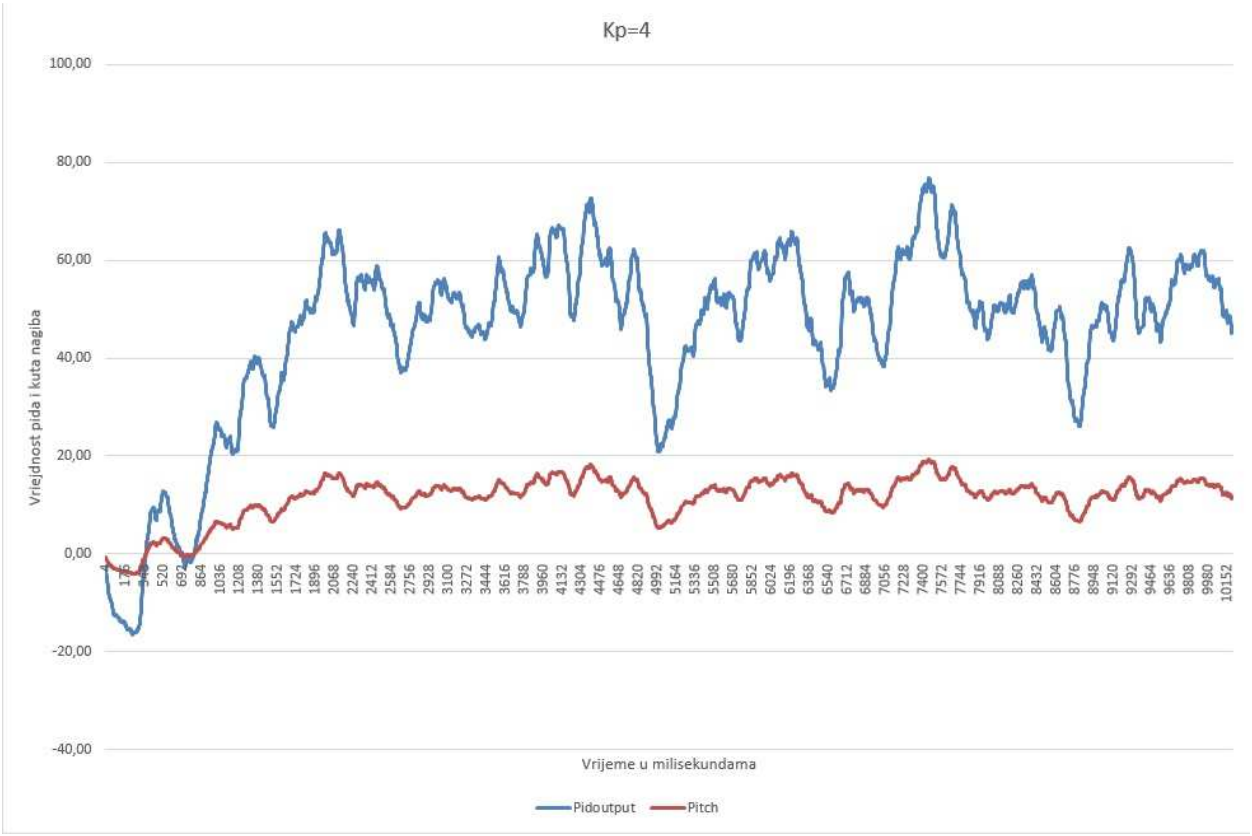
Mjerenja kako su prikazana u tablici idu i redom u grafičkom prikazu vrijednosti izlaza PID-a i nagiba letjelice s obzirom na korištene parametre prilikom testiranja. Crvenom linijom je prikazan kut nagiba dok s plavom liniju vrijednost izračunate PID vrijednosti.



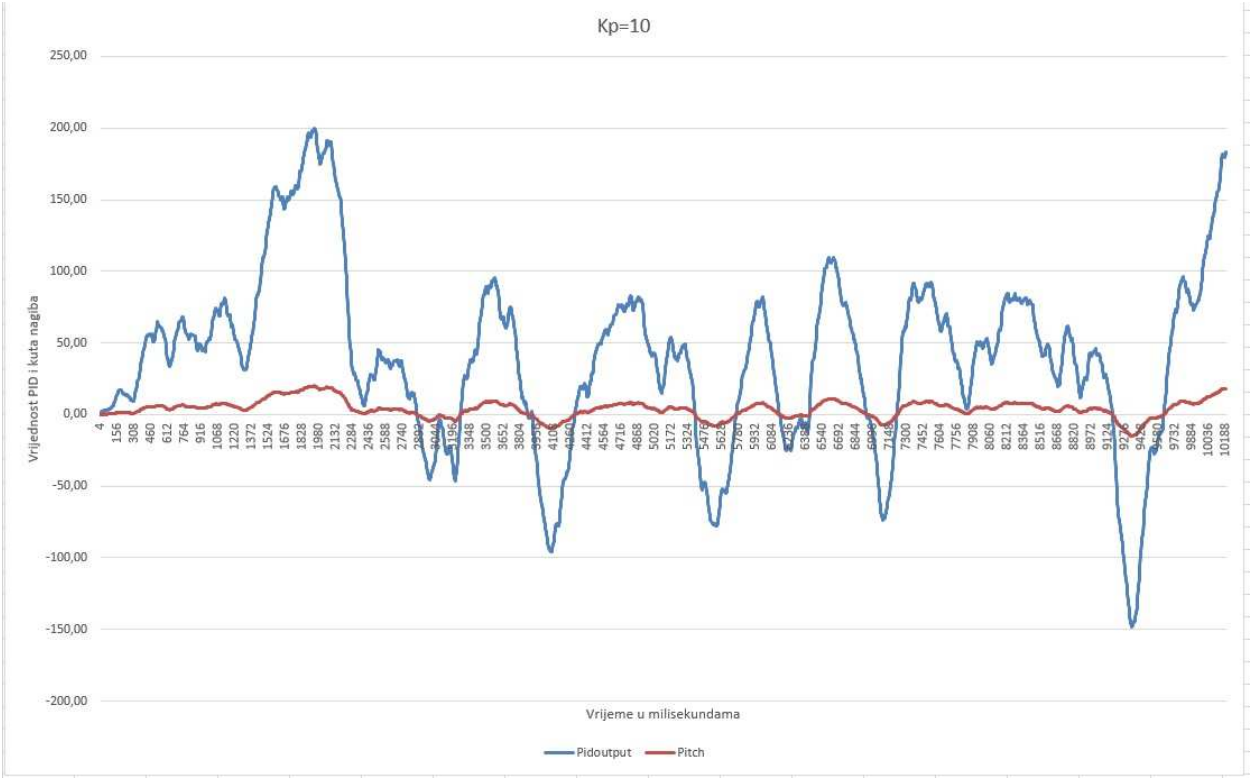
Slika 4.1. Prikaz testiranja s vrijednosti Kp=1



Slika 4.2. Prikaz testiranja s vrijednosti Kp=2

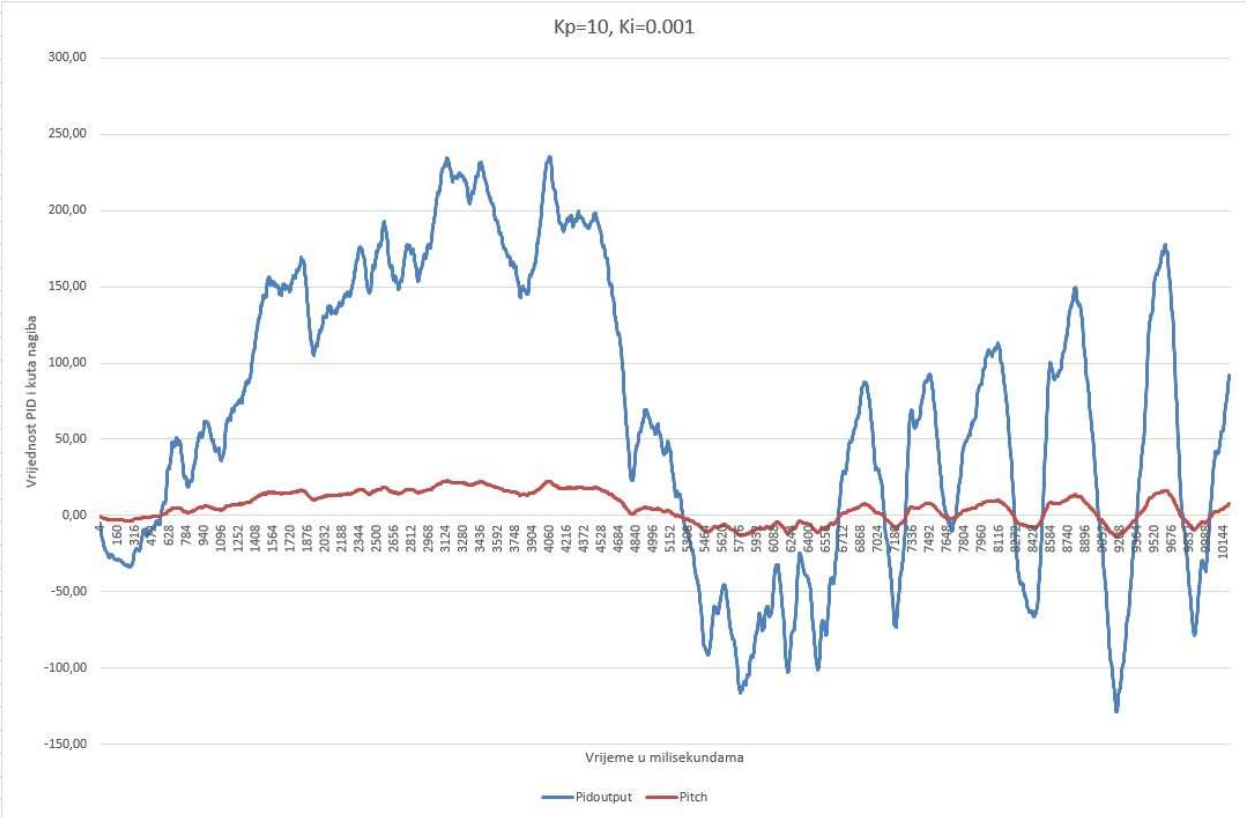


Slika 4.3. Prikaz testiranja s vrijednosti Kp=4

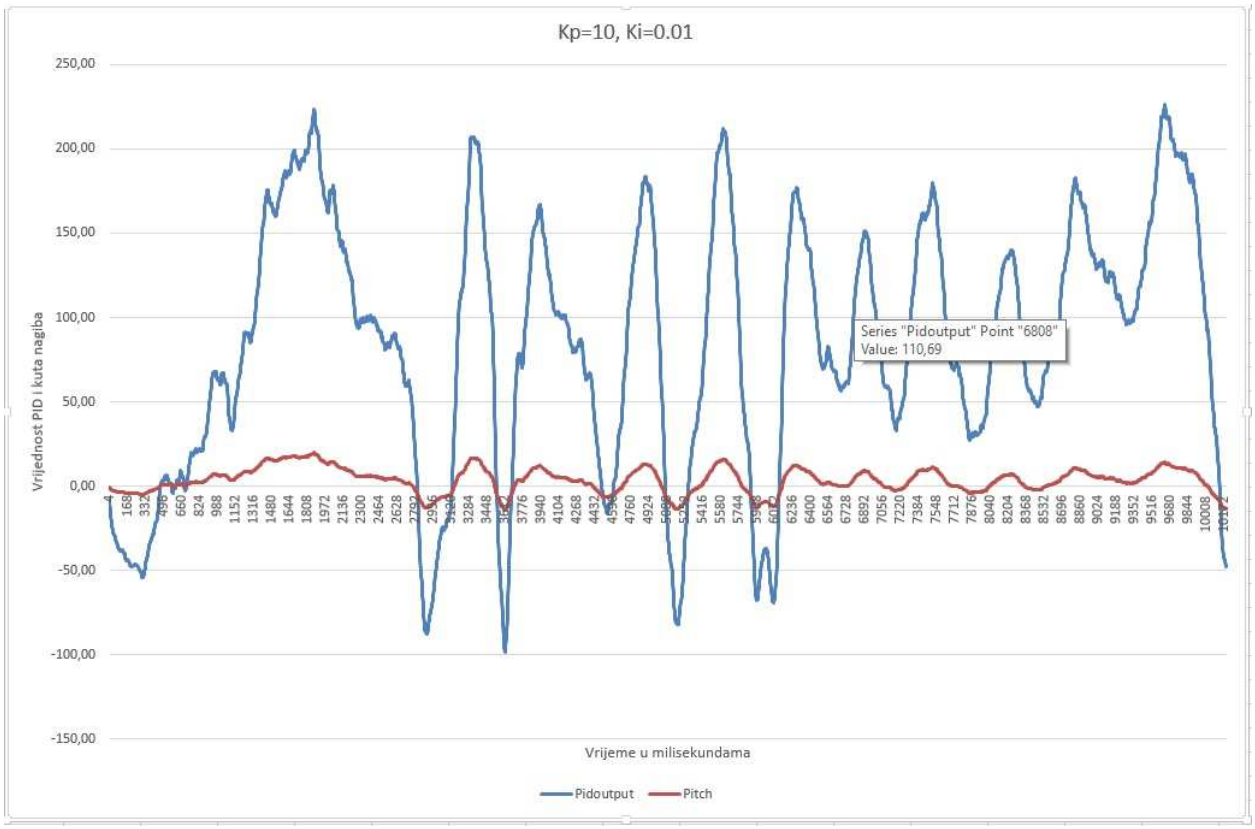


Slika 4.4. Prikaz testiranja s vrijednosti Kp=10

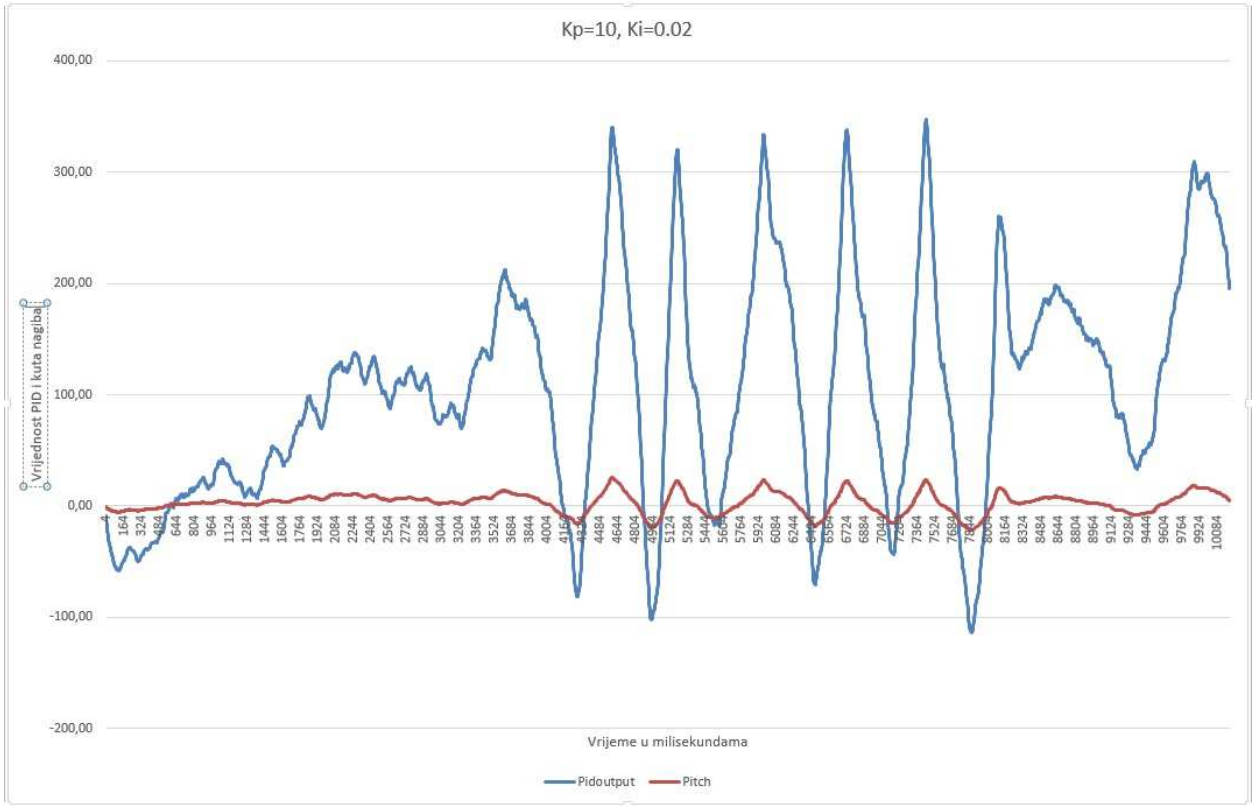
Iz slika se može zaključiti da prilikom korištenja samo Kp vrijednosti koristimo P regulator te vrijednost PID izlaza je jednaka umnošku kuta nagiba i Kp parametra. Iz grafičkog prikaza još se može zaključiti da povečanjem Kp vrijednosti letjelica sve više pokušava da se približi željenom kutu nagiba od 0°.



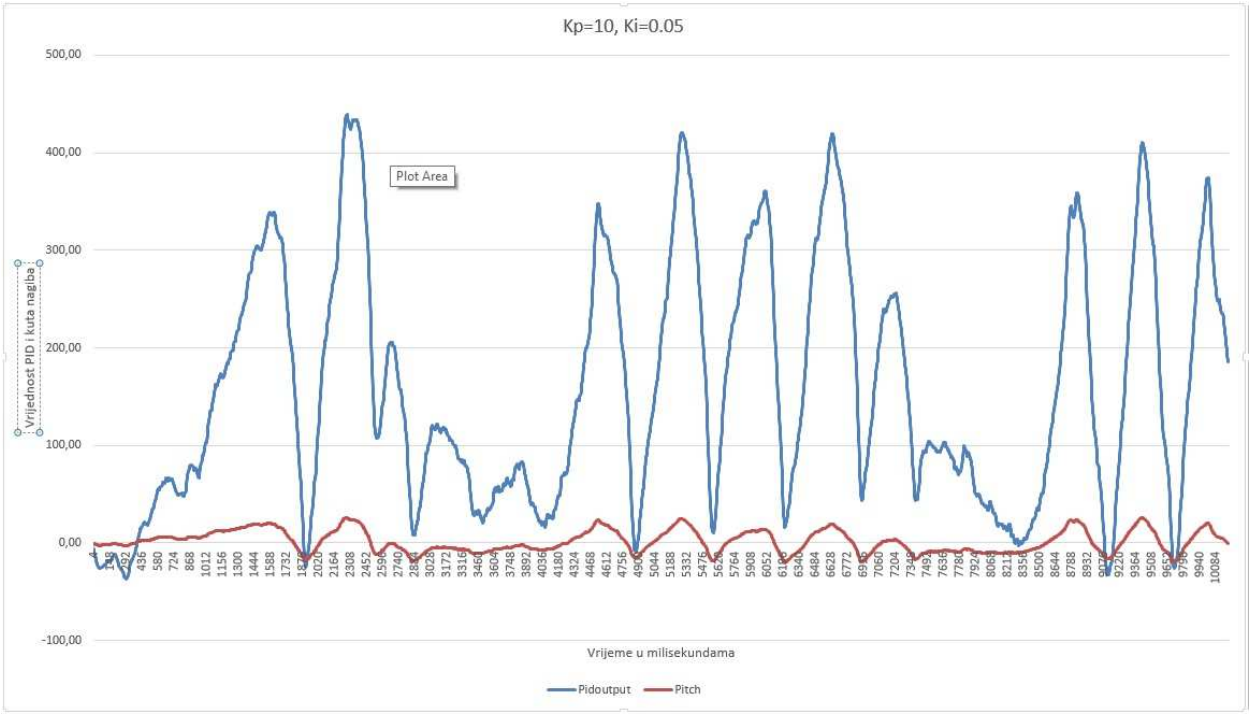
Slika 4.5. Prikaz testiranja s vrijednosti Kp=10 i Ki=0,001



Slika 4.6. Prikaz testiranja s vrijednosti Kp=10 i Ki=0,01

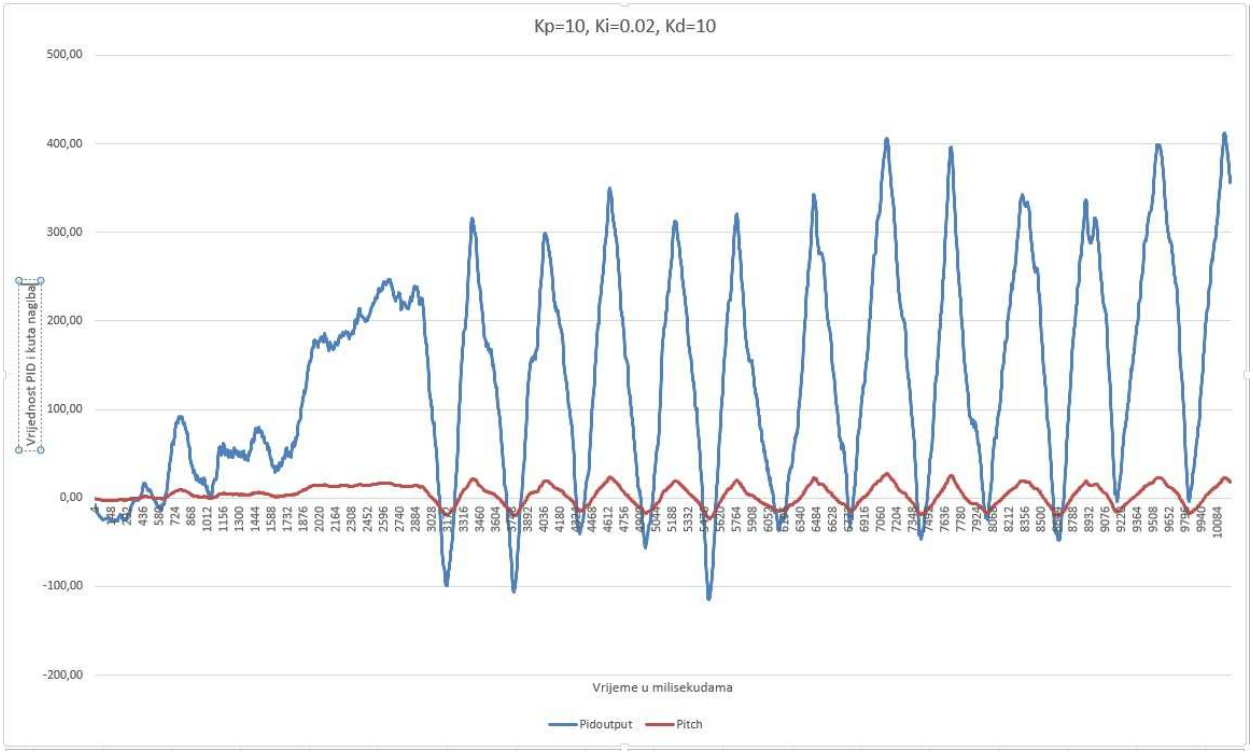


Slika 4.7. Prikaz testiranja s vrijednosti Kp=10 i Ki=0.02

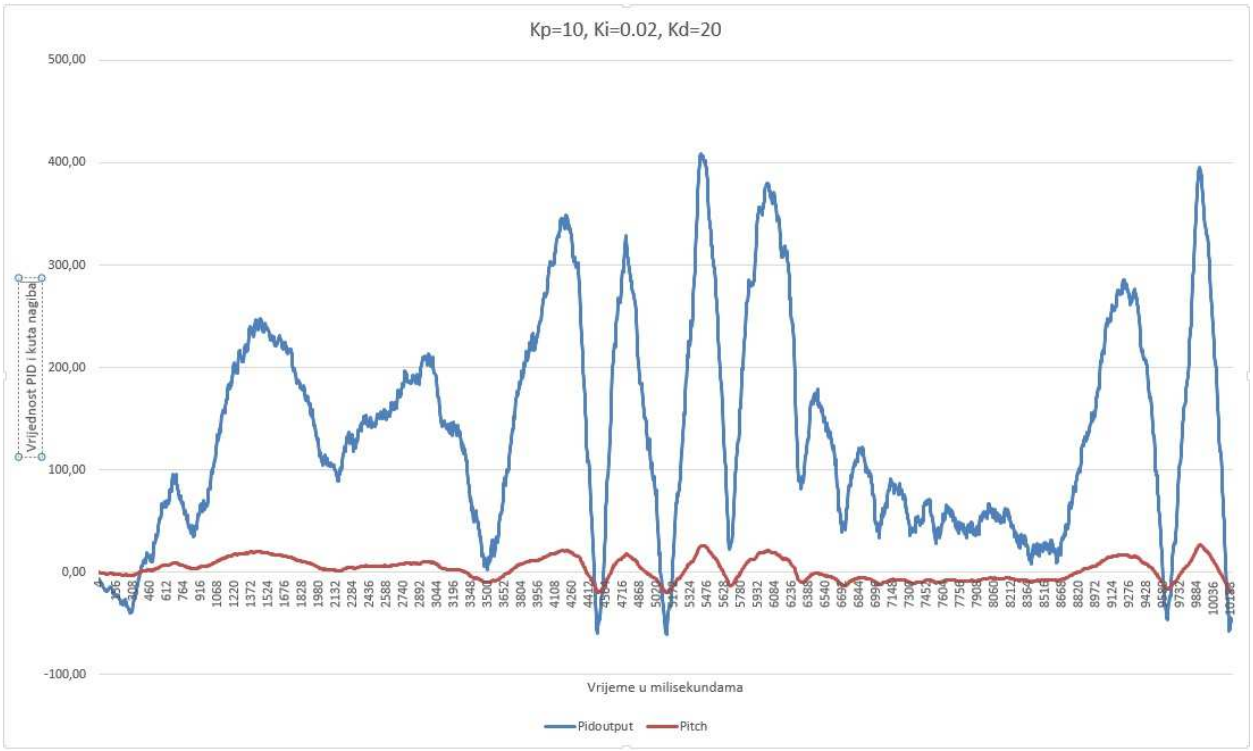


Slika 4.8. Prikaz testiranja s vrijednosti Kp=10 i Ki=0.05

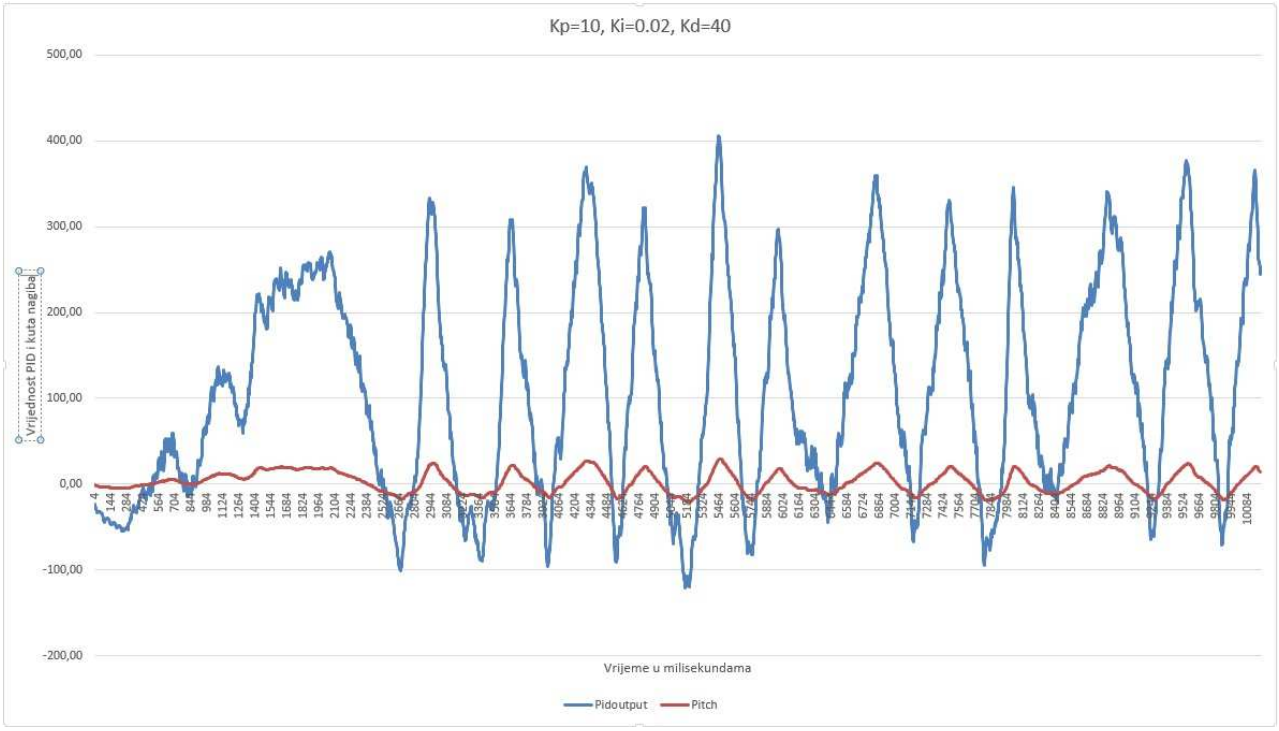
Dodavanjem koeficijenta Ki dobili smo PI regulator te to integralno djelovanje nam dodaje sumaciju pogreške u vremenu te se zbrajanjem s djelovanjem proporcionalnog djelovanja dolazimo sve bliže željenom nagibu. No još sustav ne zna kada treba usporiti snagu motora pa „pretjera“ u tom nastojanju da letjelica bude nagnuta pod željenim kutem.



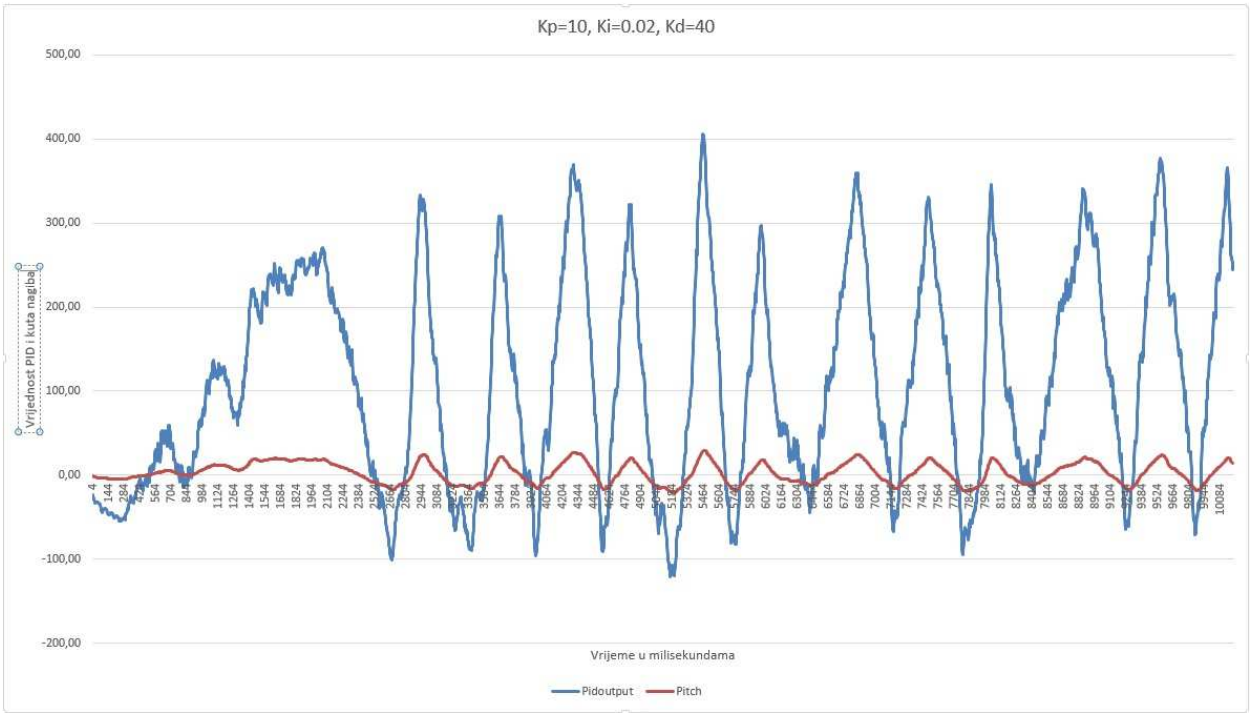
Slika 4.9. Prikaz testiranja s vrijednosti Kp=10, Ki=0.02 i Kd=10



Slika 4.10. Prikaz testiranja s vrijednosti Kp=10, Ki=0.02 i Kd=20



Slika 4.11. Prikaz testiranja s vrijednosti Kp=10, Ki=0.02 i Kd=40



Slika 4.12. Prikaz testiranja s vrijednosti Kp=10, Ki=0.02 i Kd=80

Dodavanjem Kd vrijednosti dobili smo PID regulator te povečanjem Kd vrijednosti postižemo relativno ima sve manje amplitude oscilacija i sve duže je pozicioniran oko željene vrijednosti. Metoda korištena pri projektiranju nije previše pouzdana što se može i vidjeti iz priloženih grafova te također mnogo utječe koliko se precizno može izmjeriti period oscilacija kojega treba izmjeriti s većom rezolucijom vremena.

5. ZAKLJUČAK

U ovom diplomskom radu obrađena je tema autonomnih bespilotnih letjelica te je napravljena i maketa letjelice u 4-propelerskoj konfiguraciji. Bespilotne letjelice u današnje vrijeme sve više pokazuju svoje praktične primjene od poljoporivrede za nadzor usjeva te montiranjem posebnih kamera može poljoprivrednicima pokazati na razne probleme s usjevima, također manji su troškovi nego za korištenje helikoptera pri traganju za izgubljenim osobama na planirama ili za preživljelima nakon raznih vremenskih ne prilika te se zbog svoje veličine mogu proći kroz mnoge prostore gdje helikopteri ne mogu proći.

Prilikom implementacije i rješavanja problema ovog diplomskog rada možemo reći da je djelomično rješen rad no nažalost ne i potpuno. Flight controller je potpuno gotov bezobzira što se PID paramtri nisu uspjeli pronaći budući da je njih za svaku letjelicu treba naći posebno jer su oni jedinstveni te također i daljinski upravljač vezano za potrebe u ovom radu, a to je lebdenje. U svakom slučaju tijekom obrade ove teme mnogo sam naučio o samim letjelicama, ali i o mikroupravljačima te posebice efikasnijem njihovom programiranju te korištenje prekidnih rutina i timer-a. Razlog zbog kojeg nije bilo moguće postići bolju kontrolu djelomično je razlog i u odabiru okviru gdje je zbog jednostavnosti trebao biti korišten malo manji okvir, ali i oblika „+“ te naravno i baterije koja je predviđena za okvir koja je upola lakša nego korištena. Razlog vidim u dosta teškom postizanju pronalaska centra gravitacije jer već jako malim pomakom baterije u okviru dovodi do većeg ne balansa mase. Prostora za poboljšanje svakako ima te osim korištenja lakše baterije svakako pri određivanju PID parametara pomoglo bi izrada matematičkog modela letjelice ili korištenje nekog drugog modela određivanja parametara.

6. LITERATURA

[1] VTOL definicija , https://en.wikipedia.org/wiki/VTOL , (Pristupljeno 29.6.17.)

[2] CTOL definicija, https://en.wikipedia.org/wiki/CTOL , (Pristupljeno 29.6.17.)

[3] VTOL dron slika, http://www.robotshop.com/en/550mm-rtf-quadcopter-uav.html , (Pristupljeno 29.6.17.)

[4] CTOL dron slika, http://www.militaryaerospace.com/articles/2015/06/gray-eagle-attack-drones.html ,(Pristupljeno 29.6.17.)

[5] Narodne novine 49/2015: Pravilnik o sustavima bespilotnih zrakoplova

[6] Oblici okvira bespilotnih letjelica, http://www.instructables.com/id/How-to-Design-and-3D-Print-Your-Very-Own-Quadcopte/ , (Pristupljeno 20.9.17.)

[7] Sile koje djeluju na letjelicu, https://hardsoftlucid.wordpress.com/2012/01/ , (Pristupljeno 20.9.17.)

[8] Kategorizacija letačkih misija, http://www.svijetosiguranja.eu/hr/novosti/sve-veca-rasprostranjenost-dronova-donosi-korist-ali-i-nove-rizike,19453.html , (Pristupljeno 30.6.17.)

[9] Slika unutrašnjosti outrunner BLDC motora, http://mitrocketscience.blogspot.hr/2011/08/hall-effect-sensor-placement-for.html , (Pristupljeno 20.9.17.)

[10] Slika unutrašnjosti inrunner BLDC motora, http://www.rcplanesguru.org/2015/09/electronics-and-propulsion-system-in-rc_14.html , (Pristupljeno 20.9.17.)

[11] Motor korišen u izradi makete, https://www.channeleffect.com/ImageHosting/viewImage?CompanyId=9369&FilePath=Images/74069 , (Pristupljeno 20.9.17.)

[12] ESC, https://en.wikipedia.org/wiki/Electronic_speed_control , (Pristupljeno 20.9.17.)

[13] Specifikacija mikroupravljača Arduino Nano, https://store.arduino.cc/arduino-nano , (Pristupljeno 19.9.17.)

[14]Slika propelera, https://ae01.alicdn.com/kf/HTB1d3m1HpXXXXXzaXXXq6xXFXXXA/2-Pairs-1045-10X4-5-Carbon-Fiber-Propeller-CW-CCW-Props-for-DJI-Phantom-X525-F450.jpg (Pristupano 19.9.17.)

[15] Arduino Nano , https://store.arduino.cc/arduino-nano , (Pristupljeno 19.9.17)

[16] Specifikacija MPU-6050, https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf , (Pristupljeno 19.9.17.)

[17] Slika MPU-6050 čipa, https://www.arduino.in.th/product/56/3-axis-accelerometer-gyro-module-mpu6050, (Pristupljeno 21.9.17.)

[18] Shema PID algoritma, https://upload.wikimedia.org/wikipedia/commons/thumb/2/2d/PID.svg/500px-PID.svg.png , (Pristupljeno 21.9.17.)

[19] PID algoritam u sustavu, https://www.csimn.com/CSI_pages/PIDforDummies.html , (Pristupljen 21.9.17.)

[20] Zigler-Nichols metoda određivanja PID koeficijenata, https://en.wikipedia.org/wiki/Ziegler%E2%80%93Nichols_method , (Pristupljeno 21.9.17.)

[21] Zigler-Nichols metoda određivanja koeficijenata tablica vrijednosti , http://notesnet.blogspot.hr/2013/02/zieglernichols-method.html , (Pristupljeno 21.9.17.)

SAŽETAK

U ovome radu teorijski i praktično je prikazana bespilotna letjelica u 4-propelerskoj konfiguraciji te njihove primjene u razne svrhe. Nadalje je pokazana izrada makete i programiranje flight controllera letjelice. Osim toga prikazani su potrebni dijelov i izradu te njihove karakteristike i spajanje istih u funkcionalnu cjelinu. Pokazana je također implementacija upravljačkog algoritma na mikroupravljač, metoda Ziegler-Nichols-onova za pronalazak PID paramatera te korištenje nekih funkcija samog mikroupravljača kao što su prekidne rutine i timer-i za efikasnije programiranje istih.

KLJUČNE RIJEČI

Bespilotna letjelica, mikroupravljač, Arduino, PID, žiroskop i akcelerometar.

ABSTRACT

In this master thesis are theoretically and practicaly represented unmanned aerial vehicle in 4-propeller configuration and their application. Furthemore is represented main objective of this master thesis. Buidling a model of unmanned aerial vehicle and programming flight controller for it. In addition, the required parts and design are presented with their characteristics and their integration into a functional unit. Also demonstrated is the implementation of the control algorithm on a microcontroller, the Ziegler-Nichols-method for finding a PID parameters, and the use of some microcontrollers functions such as interrupt routines and timers for more efficient programming.

KEYWORDS

Unmanned aerial vehicle, microcontroller, Arduino, PID, gyro and accelerometer.

ŽIVOTOPIS

Mislav Kulaš rođen je 12. listopada 1992. godine u Đakovu. Odrastao je u Đakovu gdje je pohađao Osnovnu školu Ivana Gorana Kovačića. Nakon završene osnovne škole s odličnim uspjehom upisuje Gimnaziju Antuna Gustava Matoša u Đakovu, smjer prirodoslovno-matematički. Poslije završetka srednje škole s vrlo dobrih uspjehom polaže državnu maturu postiže također vrlo dobar uspjeh. S obzirom na veliki interes prema tehničkim znanostima za nastavak obrazovanja upisuje se na Elektrotehnički fakultet u Osijeku, Preddiplomski studij računarstva, te isti završava u rujnu 2014. godine s temom završnog rada „Vozilo upravljano mikroupravljačem koje obilazi prepreke“. Nakon završetka preddiplomskog studija upisuje Sveučilišni diplomski studij Računarstva, smjer Procesno računarsto.

PRILOZI

Prilog 1: Programski kod daljinskog upravljača

```
const int numReadings = 10;
int readings[numReadings];      // očitanja s analog inputa
int readIndex = 0;              // pozicija trenutne vrijednosti
int total = 0;
int average = 0;                // prosjek
int lastAverage=0;
int inputPin = A0;

boolean  button=false;
boolean lastButton=false;

void setup() {
  // inicijalizacija serijske komuniakcije
  Serial.begin(115200);
  pinMode(5, INPUT);
  // initialize all the readings to 0:
  for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings[thisReading] = 0;
  }
}

void loop() {
  // oduzmi zadnju vrijednost
  total = total - readings[readIndex];
  //pročitaj s analognog ulaza
  readings[readIndex] = analogRead(inputPin);
  // dodaj vrijendost sveukupnoj
  total = total + readings[readIndex];
  readIndex = readIndex + 1;

  // ako smo na kraju array-a
```

```
 if (readIndex >= numReadings)
 {
   readIndex = 0;
 }

 // izračunaj prosjek
 average = total / numReadings;
 if(abs(lastAverage-average)>10)
 {
   lastAverage=average;
    average=map(average, 0,1023,0,220);
 // posašlji kao ASCII vrijednost
 Serial.write(average);
 }

 delay(4);        // odgoda do idućeg čitanja
}
```

Prilog 2: Programski kod flight controllera

```
#include<Wire.h>
#include<avr/io.h>
#include<avr/interrupt.h>
#define F_CPU 16000000
#define USART_BAUDRATE 115200
#define BAUD_PRESCALE (((F_CPU / (USART_BAUDRATE * 8UL)))-1)
volatile unsigned int a;


unsigned long loop_time, esc_loop_timer;
unsigned long esc_1_timer,esc_2_timer,esc_3_timer,esc_4_timer;
int esc_1, esc_2, esc_3,esc_4;

int gyro_x, gyro_y, gyro_z;
int temperature,cal_counter;
long gyro_calib_x, gyro_calib_y, gyro_calib_z;
boolean gyro_angle=false;
long acc_x, acc_y, acc_z, acc_total_vector;
float angle_roll_acc, angle_pitch_acc;

float angle_pitch, angle_roll;
int angle_pitch_buffer, angle_roll_buffer;
double angle_pitch_output, angle_roll_output;
```

```
int throttle = 1000;
int pid_max = 450;
//pid koeficijenti
float Kp_pitch = 1;//4.5+-
float Ki_pitch = 0;
float Kd_pitch = 0;

float Kp_roll = 0;
float Ki_roll = 0;
float Kd_roll = 0;

float Kp_yaw = 0;
float Ki_yaw = 0;
float Kd_yaw = 0;

float pitch_setpoint, gyro_pitch_input,  pid_i_mem_pitch, pitch_output, last_pitch_d_error;
float pid_error_temp, roll_setpoint, gyro_roll_input,  pid_i_mem_roll, roll_output,
last_roll_d_error;
float yaw_setpoint, gyro_yaw_input,  pid_i_mem_yaw, yaw_output, last_yaw_d_error;
double gyro_pitch, gyro_roll, gyro_yaw;

void setup() {

 DDRD |= B11110000;          //digital pinovi 7,6,5,4 konfigurirani kao izlaz (pinMode)

 Wire.begin();
```

```
  TWBR = 12;                //postavljen clock I2C komunikacije na 400kHz.
  setup_MPU_6050();


  for(cal_counter=0; cal_counter<1000; cal_counter++){                  //zastoj nakon konfiguracije senzora MPU6050
    PORTD |= B11110000;                                                 //Postavi digiralne pinove 4, 5, 6 i 7 HIGH.
    delayMicroseconds(1000);                                            //Pričekaj jednu sekundu
    PORTD &= B00001111;                                                 //Postavi digiralne pinove 4, 5, 6 i 7 LOW.
    delayMicroseconds(3000);
  }
  digitalWrite(13, HIGH);                                               //uključi ledicu na pinu 13
                                                                        //kalibriraj senzor mpu6050
  for(cal_counter=0; cal_counter<2500; cal_counter++){
    read_data_MPU_6050();
    gyro_calib_x += gyro_x;
    gyro_calib_y += gyro_y;
    gyro_calib_z += gyro_z;

    PORTD |= B11110000;
    delayMicroseconds(1000);
    PORTD &= B00001111;
    delayMicroseconds(3000);
  }
```

```
  gyro_calib_x /=2500;                                    //izračunaj srednju vrijednost pogreške žiroskopa
  gyro_calib_y /=2500;
  gyro_calib_z /=2500;




  digitalWrite(13, LOW);                                  //ugasi ledicu na pinu 13 jer je kalibracija gotova
  delay(2000);
  USART_INIT();
  pid_i_mem_pitch = 0;
  pid_i_mem_roll = 0;
  pid_i_mem_yaw = 0;
  last_pitch_d_error= 0;
  last_roll_d_error = 0;
  last_yaw_d_error = 0;

  loop_time=micros();

}
void loop() {
    read_data_MPU_6050();
    gyro_x -= gyro_calib_x;                               //kompenziraj vrijednost.
    gyro_y -= gyro_calib_y;
    gyro_z -= gyro_calib_z;
```

```
  //Izračunaj vrijednost kuta 0.0000611 = 1 / (250Hz x 65.5)
  angle_pitch += gyro_x * 0.0000611;                                   //Izračunaj vrijednost pitch kuta prijedenog i dodaj sveukupnoj vrijednosti
  angle_roll += gyro_y * 0.0000611;                                    //Izračunaj vrijednost roll kuta prijedenog i dodaj sveukupnoj vrijednosti
  //0.000001066 = 0.0000611 * (3.142(PI) / 180degr) Arduino trigonometrijske funkcije su u radijanima
  angle_pitch += angle_roll * sin(gyro_z * 0.000001066);               //Ako smo se zarotirali oko yaw osi transformiraj roll kut na pitch kut
  angle_roll -= angle_pitch * sin(gyro_z * 0.000001066);

  //Accelerometer angle calculations
  acc_total_vector = sqrt((acc_x*acc_x)+(acc_y*acc_y)+(acc_z*acc_z));  //Izračunaj totalni vektor akceleracije
  //57.296 = 1 / (3.142 / 180) The Arduino asin function is in radians
  angle_pitch_acc = asin((float)acc_y/acc_total_vector)* 57.296;       //Izračunaj pitch kut
  angle_roll_acc = asin((float)acc_x/acc_total_vector)* -57.296;       //Izračunaj roll kut

  angle_pitch_acc -= 0;                                                //Kalibracija akcelerometra
  angle_roll_acc -= 0;

  if(gyro_angle){
    angle_pitch = angle_pitch * 0.96 + angle_pitch_acc * 0.04;     //Ispravi kut dobiven žiroskopom s kutom dobivenim akcelerometrom
    angle_roll = angle_roll * 0.96 + angle_roll_acc * 0.04;
  }
  else{                                                                //Na početku preuzmi kut od akcelerometra
```

```
    angle_pitch = angle_pitch_acc;
    angle_roll = angle_roll_acc;
    gyro_angle = true;
  }

  //za prigušenje naglih povečanja vrijednosti kutova korišten je komplementaran filter
  angle_pitch_output = angle_pitch_output * 0.9 + angle_pitch * 0.1;
  angle_roll_output = angle_roll_output * 0.9 + angle_roll * 0.1;

    pitch_setpoint = 0;
    roll_setpoint = 0;
    yaw_setpoint= 0;

//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>      ULAZI ZA PID
<<<<<<<<<<<<<<<<<<<<<<<<<<<<
    calculate_PID();

// izračun duljine pulsa za motor

    if (throttle > 1800) throttle = 1800;
    esc_1 = throttle - pitch_output + roll_output - yaw_output;
    esc_2 = throttle + pitch_output + roll_output + yaw_output;
    esc_3 = throttle + pitch_output - roll_output - yaw_output;
    esc_4 = throttle - pitch_output - roll_output + yaw_output;

    if (esc_1 < 1100) esc_1 = 1100;                              //Ako je vrijednost impulsa manja ne
gasi motore
```

```
  if (esc_2 < 1100) esc_2 = 1100;
  if (esc_3 < 1100) esc_3 = 1100;
  if (esc_4 < 1100) esc_4 = 1100;
  if(esc_1 > 2000)esc_1 = 2000;                                   //Limitraj vrijednost impulsa.
  if(esc_2 > 2000)esc_2 = 2000;
  if(esc_3 > 2000)esc_3 = 2000;
  if(esc_4 > 2000)esc_4 = 2000;

  if(throttle < 1050) esc_1=esc_2=esc_3=esc_4=1000;

while(micros() - loop_time < 4000);
loop_time=micros();

PORTD |= B11110000;
esc_1_timer = esc_1 + loop_time;
esc_2_timer = esc_2 + loop_time;
esc_3_timer = esc_3 + loop_time;
esc_4_timer = esc_4 + loop_time;

while(PORTD >= 16){
  esc_loop_timer = micros();
  if(esc_1_timer <= esc_loop_timer){
    PORTD &= B11101111;
  }
  if(esc_2_timer <= esc_loop_timer){
    PORTD &= B11011111;
```

```
    }
    if(esc_3_timer <= esc_loop_timer){
      PORTD &= B10111111;
    }
    if(esc_4_timer <= esc_loop_timer){
      PORTD &= B01111111;
    }
  }
}


//>>>>>>>>>>>>>>>>>>>   konfiguriranje MPU6050   <<<<<<<<<<<<<<<<<<<<<<<<<<<<

void setup_MPU_6050(){

  Wire.beginTransmission(0x68);
  Wire.write(0x6B);
  Wire.write(0x00);
  Wire.endTransmission();
  //konfiguriraj akcelerometar (+/-8g)
  Wire.beginTransmission(0x68);
  Wire.write(0x1C);
  Wire.write(0x10);
  Wire.endTransmission();
  //konfiguriraj žiroskop na 500dps
  Wire.beginTransmission(0x68);
```

```
  Wire.write(0x1B);
  Wire.write(0x08);
  Wire.endTransmission();
}

//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>    ČITANJE VRIJEDNOSTI SENZORA
MPU6050    <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

void read_data_MPU_6050(){
  Wire.beginTransmission(0x68);
  Wire.write(0x3B);
  Wire.endTransmission();
  Wire.requestFrom(0x68,14);
  while(Wire.available() < 14);
  acc_x = Wire.read()<<8|Wire.read();
  acc_y = Wire.read()<<8|Wire.read();
  acc_z = Wire.read()<<8|Wire.read();
  temperature = Wire.read()<<8|Wire.read();
  gyro_x = Wire.read()<<8|Wire.read();
  gyro_y = Wire.read()<<8|Wire.read();
  gyro_z = Wire.read()<<8|Wire.read();
}

//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>    PID CODE
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

```
void calculate_PID(){


    pid_error_temp = angle_pitch_output - pitch_setpoint;
    pid_i_mem_pitch += Ki_pitch * pid_error_temp;
    if(pid_i_mem_pitch > pid_max) pid_i_mem_pitch=pid_max;
    else if(pid_i_mem_pitch < pid_max * -1)pid_i_mem_pitch = pid_max *-1;
    pitch_output= Kp_pitch * pid_error_temp + pid_i_mem_pitch + Kd_pitch * (pid_error_temp - last_pitch_d_error);
    if(pitch_output > pid_max)pitch_output = pid_max;
    else if(pitch_output< pid_max * -1)pitch_output = pid_max * -1;
    last_pitch_d_error = pid_error_temp;


    pid_error_temp = angle_roll_output - roll_setpoint;
    pid_i_mem_roll += Ki_roll * pid_error_temp;
    if(pid_i_mem_roll > pid_max) pid_i_mem_roll=pid_max;
    else if(pid_i_mem_roll < pid_max * -1)pid_i_mem_roll = pid_max *-1;
    roll_output= Kp_roll * pid_error_temp + pid_i_mem_roll + Kd_roll * (pid_error_temp - last_roll_d_error);
    if(roll_output > pid_max)roll_output = pid_max;
    else if(roll_output< pid_max * -1)roll_output = pid_max * -1;
    last_roll_d_error = pid_error_temp;


    pid_error_temp = gyro_yaw_input - yaw_setpoint;
    pid_i_mem_yaw += Ki_yaw * pid_error_temp;
```

```
   if(pid_i_mem_yaw > pid_max) pid_i_mem_yaw=pid_max;
   else if(pid_i_mem_yaw < pid_max * -1)pid_i_mem_yaw = pid_max *-1;
   yaw_output= Kp_yaw * pid_error_temp + pid_i_mem_yaw + Kd_yaw * (pid_error_temp - 
last_yaw_d_error);
   if(yaw_output > pid_max)yaw_output = pid_max;
   else if(yaw_output< pid_max * -1)yaw_output = pid_max * -1;
   last_yaw_d_error = pid_error_temp;


}


//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>     UART INICIJALIZACIJA 
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
void USART_INIT(){
  UBRR0 = BAUD_PRESCALE;
  UCSR0C =((0<<USBS0)|(1<<UCSZ01)|(1<<UCSZ00));
  UCSR0B = ((1<<RXEN0)|(1<<TXEN0) | (1<<RXCIE0));
  UCSR0A =(1<<U2X0);
  sei();
}


//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>     PREKIDNA RUTINA 
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<


ISR(USART_RX_vect){
  a=UDR0;
  throttle= map(a,0,220,1000,1550);
}
```