Upravljanje sustavom više bespilotnih letjelica

Papić, Marija

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 53

UPRAVLJANJE SUSTAVOM VIŠE BESPILOTNIH LETJELICA

Marija Papić

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 53

UPRAVLJANJE SUSTAVOM VIŠE BESPILOTNIH LETJELICA

Marija Papić

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. listopada 2023.

DIPLOMSKI ZADATAK br. 53

Pristupnica:	**Marija Papić (0036503041)**
Studij:	Informacijska i komunikacijska tehnologija
Profil:	Automatika i robotika
Mentorica:	izv. prof. dr. sc. Tamara Petrović

Zadatak: **Upravljanje sustavom više bespilotnih letjelica**

Opis zadatka:

Radom na diplomskom zadatku potrebno je osmisliti i implementirati upravljanje sustavom bespilotnih letjelica primjenom Reynoldsovih pravila te upravljanje formacijom primjenom konsenzus protokola. Da bi se to ostvarilo, potrebno je ostvariti lokalizaciju svake letjelice primjenom Optitrack sustava, te implementirati algoritme niže razine upravljanja bespilotnim letjelicama. Potrebno je osigurati da se letjelice ne sudaraju. Rješenje je potrebno provjeriti u Gazebo simulatoru te na stvarnom sustavu primjenom bespilotnih letjelica.

Rok za predaju rada: 9. veljače 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 53

UPRAVLJANJE SUSTAVOM VIŠE BESPILOTNIH LETJELICA

Marija Papić

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. listopada 2023.

DIPLOMSKI ZADATAK br. 53

Pristupnica:	**Marija Papić (0036503041)**
Studij:	Informacijska i komunikacijska tehnologija
Profil:	Automatika i robotika
Mentorica:	doc. dr. sc. Tamara Petrović

Zadatak: **Upravljanje sustavom više bespilotnih letjelica**

Opis zadatka:

Radom na diplomskom zadatku potrebno je osmisliti i implementirati upravljanje sustavom bespilotnih letjelica primjenom Reynoldsovih pravila te upravljanje formacijom primjenom konsenzus protokola. Da bi se to ostvarilo, potrebno je ostvariti lokalizaciju svake letjelice primjenom Optitrack sustava, te implementirati algoritme niže razine upravljanja bespilotnim letjelicama. Potrebno je osigurati da se letjelice ne sudaraju. Rješenje je potrebno provjeriti u Gazebo simulatoru te na stvarnom sustavu primjenom bespilotnih letjelica.

Rok za predaju rada: 9. veljače 2024.

SADRŽAJ

1. Uvod

Bespilotne letjelice (eng. UAV - Unmanned Aerial Vehicles) sve više postaju ključni alat u mnogim područjima, kao što su nadzor, istraživanje, potraga i vojne operacije. Karakterizirane su malim dimenzijama, niskom cijenom, izuzetnom okretnošću i sposobnošću pristupa područjima koja su opasna za ljude. Razvoj elektronike je doprinio smanjenju njihove veličine, i mogućnošću opreme različitim kombinacijama senzora.

Posljednjih godina se javlja interes za razvojem sustava s više UAV letjelica koji omogućuju efikasnije i brže rješavanje složenih zadataka koristeći međusobnu koordinaciju između letjelica. Takvi sustavi, uz dobro razvijeno upravljanje mogu donijeti niz prednosti, od kojih su neke vremenska efikasnost, cijena, veća otpornost na kvarove.

U ovom radu cilj je ostvariti koordinirano kretanje sustava s više letjelica pomoću Reynoldsovih pravila [14] i konsenzus protokola [13]. U tu svrhu koristit će se simulator Gazebo [2] i stvarne letjelice Crazyflie [1]. Prvi korak u razvoju takvog sustava je matematičko modeliranje jedne UAV letjelice. Ispravno modeliranje kinematike i dinamike letjelice preduvjet je za razvoj upravljačkog dijela sustava. Cilj ovoga dijela je sinteza regulatora niže razine za upravljanje letjelicom.

Zatim, koristeći razvijeni upravljački model, provedene su simulacije UAV letjelice. Pokazano je pravilno uzlijetanje, lebdjenje, promjena visine i kuta nagiba. Rezultati simulacija potvrđuju valjanost matematičkog modela i pružaju temelj za daljnje istraživanje.

Koristeći razvojno okruženje RotorS [8] (javni repozitorij za simulaciju UAV letjelica u Gazebu, razvijen na ETH Zürich) razvijen je sustav s tri letjelice koje se kreću u skladu s Reynoldsovim pravilima. To je set jednostavnih pravila ponašanja pojedinačnih agenata u sustavu, što rezultira složenim ponašanjem cijelog skupa. Naknadno, poštujući konsenzus protokol, tj. poštujući određena pravila interakcije među agentima, postignuta je željena formacija sustava .

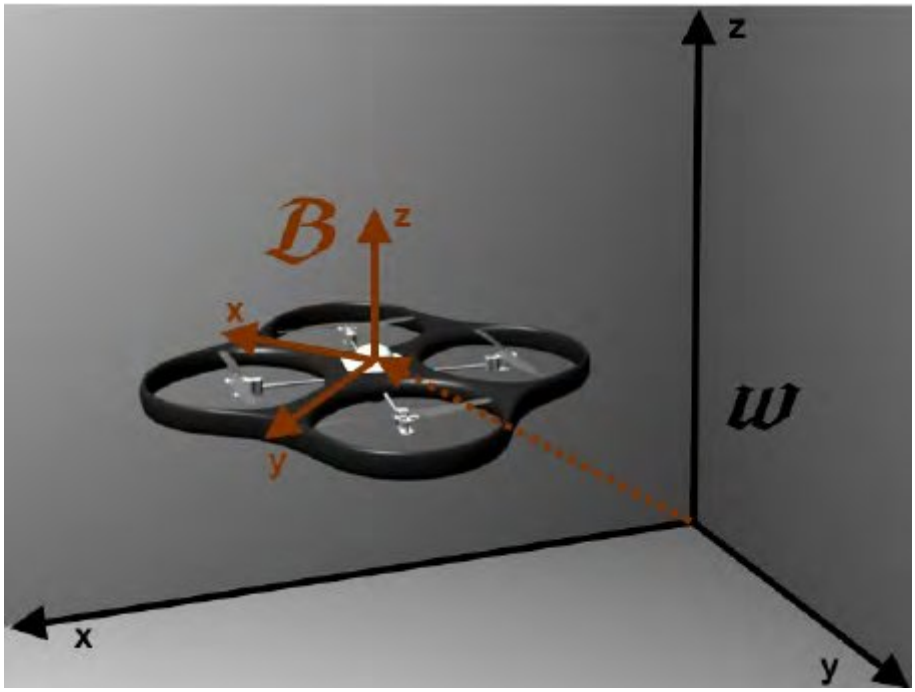
Ostatak rada je strukturiran na sljedeći način: U drugom poglavlju detaljno je obrađen matematički model letjelice, izveden je kinematički i dinamički model. U trećem poglavlju se radi upravljanje, odabran je PID regulator u kaskadnoj petlji za poziciju

i orijentaciju. U četvrtom poglavlju su obrađeni višerobotski sustavi i njihove metode upravljanja. U petom poglavlju su prikazane simulacije iz Gazebo simulatora te pripadni grafovi za prikaz promjena pri mijenjanju parametara.

2. Matematičko modeliranje letjelice

2.1. Osnovni koncepti kvadkoptera

U ovom dijelu opisat će se princip rada letjelice s četiri rotora, kvadkoptera. Osim takvih letjelica, postoje i drugi modeli s različitim brojem rotora, ali odabran je taj model zbog simetrije sustava i jednostavnijeg upravljanja. Kao prvi korak u razvoju modela kvadkoptera potrebno je definirati koordinatne sustave u kojima će se opisati model. Koriste se dva koordinatna sustava, fiksni (eng. *World frame*) i mobilni (eng. *Body frame*), prikazani na slici 2.1.

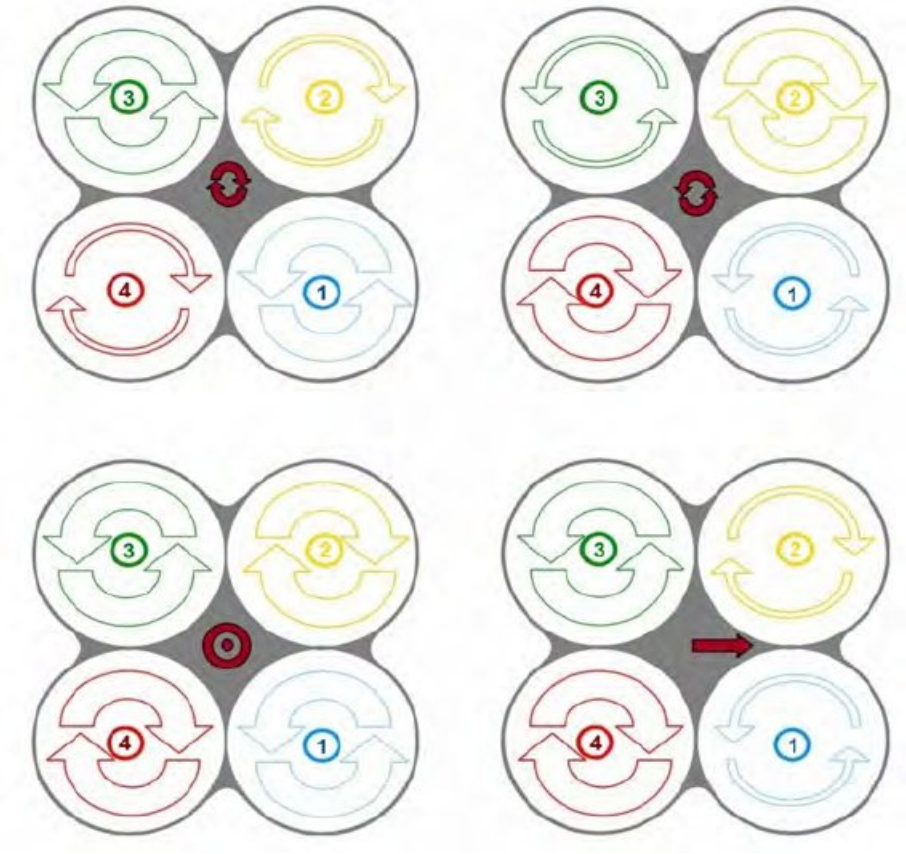


Slika 2.1: Mobilni (B) i fiksni (W) koordinatni sustavi

Kao fiksni koordinatni sustav izabran je O_{ENU} (East-North-Up). Ishodište koordinatnog sustava tijela (mobilni sustav) je u centru mase tijela.

Pozicija i orijentacija kvadkoptera se kontroliraju na temelju promjene brzina če-

tiri motora. Na kvadkopter mogu djelovati sljedeće sile i momenti: sila potiska koju stvaraju rotori, momenti poniranja i valjanja koje uzrokuju razlike u potiscima rotora, gravitacija, žiroskopski efekt i moment skretanja. Propeleri kvadkoptera su postavljeni u kvadratnu formu, tako da se oni koji su jedan do drugog okreću u suprotnim smjerovima kako bi se poništavanjem momenta skretanja održala stabilnost letjelice. Ovo je inherentno svojstvo letjelica s četiri rotora, a problem je npr. letjelicama s jednim propelerom, jer se počnu okretati u suprotnom smjeru (3. Newtonov zakon).



Slika 2.2: Upravljanje brzinom za osnovne kretnje letjelice [6]

Za opis gibanja čvrstog tijela u prostoru je potrebno šest stupnjeva slobode. Ta gibanja se u kvadkopteru implementiraju podešavanjem rotacijskih brzina rotora. To su gibanja prema naprijed i unatrag, bočna i vertikalna gibanja, te rotacijske kretnje skretanje, poniranje i valjanje. Zbog šest izlaza (stupnjeva slobode) i četiri ulaza (rotora kojima se upravlja), kvadkopter se smatra podaktuiranim sustavom.

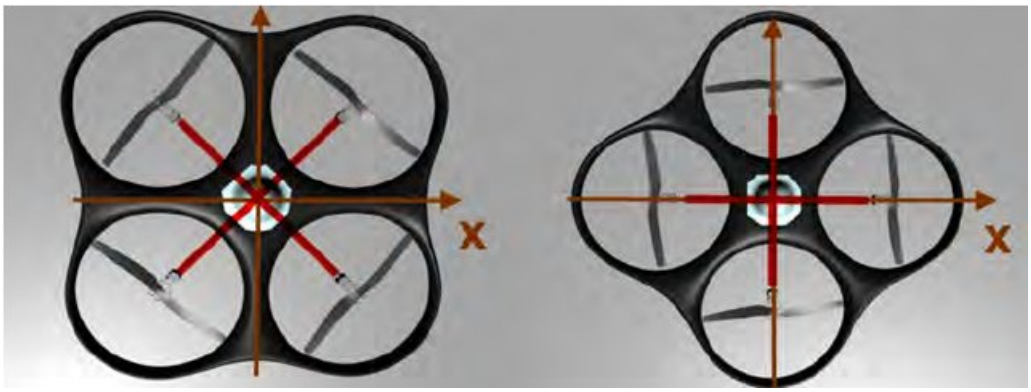
Na slici 2.2 prikazane su četiri osnovne kretnje kvadkoptera, debljina strelice označava veću brzinu:

– uzdizanje ili spuštanje podizanje po z osi (donja lijeva slika), postiže se povećanje ili smanjivanjem brzine svih motora za istu vrijednost

– skretanje: rotacija oko z osi u smjeru kazaljke na satu i obrnuto (gornja lijeva i desna slika)

– poniranje: smanjenje brzina motorima 1 i 2, povećanje motorima 3 i 4

S obzirom na položaj četiriju rotora u odnosu na koordinatni sustav tijela, postoje dvije konfiguracije kvadkoptera, a to su križna (X) i plus konfiguracija, prikazane na slici 2.3. U gornjem opisu je korištena X konfiguracija.



Slika 2.3: Usporedba X (lijevo) i Plus (desno) konfiguracije kvadkoptera

U X konfiguraciji su parovi rotora koji imaju isti smjer vrtnje u suprotnim kvadrantima koordinatnog sustava a u plus konfiguraciji su položeni na osi x i y. Recimo da se želi ostvariti gibanje u smjeru x osi. U plus konfiguraciji tada treba samo promijeniti brzinu para rotora koji leži na osi x, jednom smanjiti brzinu a drugom povećati. U križnoj konfiguraciji treba mijenjati brzinu svih rotora, zadnja dva povećati a prednja dva smanjiti. Iako se s gledišta upravljanja čini jednostavnije izabrati Plus konfiguraciju, smislenije je raditi s X konfiguracijom zbog toga što je lakše promijeniti brzinu svih rotora za manju vrijednost nego samo dva rotora za veću vrijednost koja bi možda preopteretila motore pogotovo ako letjelica nosi teret [11].

2.2. Kinematika

U ovom poglavlju se izvodi matemetički model kvadkoptera na temelju izvoda u [7]. Kinematika se bavi gibanjem tijela bez uzimanja u obzir sila i momenata koji djeluju na tijelo.

Kvadkopter je čvrsto tijelo u 3D prostoru i njegova pozicija se može opisati sa šest parametara, tri za poziciju i tri za orijentaciju. Taj vektor zapisan u fiksnom koordinatnom sustavu je:

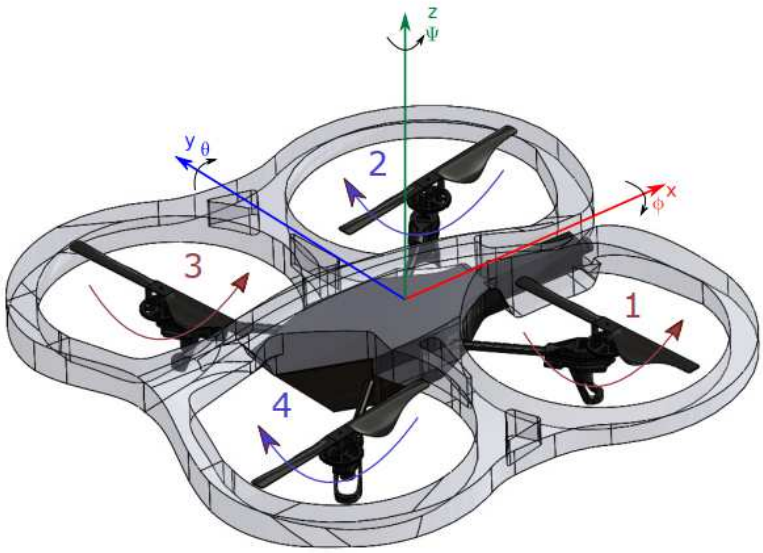
$$\boldsymbol{\eta} = [\mathbf{v} \quad \boldsymbol{\omega}]^T = [x\ y\ z\ \phi\ \theta\ \psi]^T \tag{2.1}$$

On sadrži linearne i kutne pozicije kvadkoptera u fiksnom koordinatnom sustavu.

Za orijentaciju je izabran opis Eulerovim kutevima, iako se u tom slučaju može stvoriti problem koji ograničava raspon mogućih orijentacija, tzv. *gimbal lock*. Osim toga može se koristiti i zapis u kvaternionima, ali tada se javlja drugo ograničenje, kvaternioni ne daju jedinstveni prikaz orijentacije. Opis Eulerovim kutovima predstavlja sekvencu tri elementarne rotacije oko osi koordinatog sustava da bi se dobila veza između dva koordinatna sustava.

Koristi se notacija *skretanje-poniranje-valjanje*, *(eng. yaw-pitch-roll)* što znači da se vrše tri elementarne rotacije, prvo se kvadkopter zakreće oko svoje *z* osi, zatim *y* osi i na kraju *x* osi.

Na slici 2.4 se može vidjeti kvadkopter s pridruženim mobilnim koordinatnim sustavom koji se poklapa s njegovim težištem. Označeni su i Eulerovi kutovi rotacije oko svake osi.



Slika 2.4: Koordinatni sustav pridružen tijelu kvadkoptera (body frame). [3]

Rotacijske matrice koje predstavljaju te rotacije su:

$$\mathbf{R_x}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{2.2}$$

$$\mathbf{R_y}(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2.3}$$

$$\mathbf{R_z}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$\mathbf{R_{zyx}}(\phi, \theta, \psi) = \mathbf{R_z}(\psi) \cdot \mathbf{R_y}(\theta) \cdot \mathbf{R_x}(\phi) \tag{2.5}$$

$$= \begin{bmatrix} C(\psi)C(\theta) & C(\psi)S(\phi)S(\theta) - C(\phi)S(\psi) & S(\phi)S(\psi) + C(\phi)C(\psi)S(\theta) \\ C(\theta)S(\psi) & C(\phi)C(\psi) + S(\phi)S(\psi)S(\theta) & C(\phi)S(\psi)S(\theta) - C(\psi)S(\phi) \\ -S(\theta) & C(\theta)S(\phi) & C(\phi)C(\theta) \end{bmatrix} \tag{2.6}$$

Dobivena je matrica složene rotacije (2.6) koja povezuje fiksni i mobilni koordinatni sustav (vrijedi $C(\theta) = \cos(\theta)$, $S(\theta) = \sin(\theta)$ itd.).

Dakle, definiran je vektor pozicija u fiksnsom koordinatnom sustavu (2.1), a vektor koji sadrži linearne i kutne brzine u mobilnom koordinatnom sustavu je: $[u\ v\ w\ p\ q\ r]^T$, pa se njihov odnos može zapisati na sljedeći način:

$$\mathbf{v} = \mathbf{R} \cdot \mathbf{v}^B \tag{2.7}$$

$$\boldsymbol{\omega} = \mathbf{T} \cdot \boldsymbol{\omega}^B\ , \tag{2.8}$$

gdje $\mathbf{v} = [\dot{x}\ \ \dot{y}\ \ \dot{z}]^T \in \mathbb{R}^3$, $\boldsymbol{\omega} = [\dot{\phi}\ \ \dot{\theta}\ \ \dot{\psi}]^T \in \mathbb{R}^3$, $\mathbf{v}^B = [u\ \ v\ \ w]^T \in \mathbb{R}^3$, $\boldsymbol{\omega}^B = [p\ \ q\ \ r]^T \in \mathbb{R}^3$, a $\mathbf{T}$ je matrica kutnih transformacija.

Ukupni vektor brzine u mobilnom (B) koordinatnom sustavu povezan je na sljedeći način s vektorom brzine u fiksnom (W) koordinatnom sustavu:

$$\boldsymbol{\eta} = [\mathbf{p}^W \quad \boldsymbol{\Theta}^W]^T \tag{2.9}$$

$$\boldsymbol{\nu} = [\mathbf{v}^B \quad \boldsymbol{\omega}^B]^T \tag{2.10}$$

$$\dot{\boldsymbol{\eta}} = \mathbf{J} \cdot \boldsymbol{\nu} \tag{2.11}$$

Matrica $\mathbf{J}$ je definirana na sljedeći način:

$$\mathbf{J} = \begin{bmatrix} \mathbf{R} & \mathbf{0_{3x3}} \\ \mathbf{0_{3x3}} & \mathbf{T} \end{bmatrix} \tag{2.12}$$

Matrica transformacija $\mathbf{T}$ povezuje kutne brzine u mobilnom koordinatnom sustavu (p, q, r) i promjene kutnih pozicija u fiksnom koordinatnom sustavu $(\dot{\phi}, \dot{\theta}, \dot{\psi})$. Definirana je na sljedeći način:

Kutne brzine u mobilnom sustavu $(\omega_x, \omega_y, \omega_z)$ i promjene kutnih pozicija u fiksnom sustavu $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ veže matrica kutnih transformacija:

$$\mathbf{T} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \dfrac{\sin(\phi)}{\cos(\theta)} & \dfrac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.13}$$

Iz 2.11 i 2.12 dobiven je kinematički model kvadkoptera:

$$\begin{cases} \dot{x} = w[S(\phi)S(\psi) + C(\phi)C(\psi)S(\theta)] - v[C(\phi)S(\psi) - C(\psi)S(\phi)S(\theta)] + u[C(\psi)C(\theta)] \\ \dot{y} = v[C(\phi)C(\psi) + S(\phi)S(\psi)S(\theta)] - w[C(\psi)S(\phi) - C(\phi)S(\psi)S(\theta)] + u[C(\theta)S(\psi)] \\ \dot{z} = w[C(\phi)C(\theta)] - u\,[S(\theta)] + v\,[C(\theta)S(\phi)] \\ \dot{\phi} = p + r[C(\phi)T(\theta)] + q\,[S(\phi)T(\theta)] \\ \dot{\theta} = q\,[C(\phi)] - r\,[S(\phi)] \\ \dot{\psi} = r\,\dfrac{C(\phi)}{C(\theta)} + q\,\dfrac{S(\phi)}{C(\theta)} \end{cases} \tag{2.14}$$

2.3. Dinamika

Ukupna sila koja djeluje na kvadkopter može se opisati sljedećom relacijom:

$$m(\boldsymbol{\omega}^B \times \mathbf{v^B} + \dot{\mathbf{v}}^\mathbf{B}) = \mathbf{F}^B\,, \tag{2.15}$$

gdje je m masa kvadkoptera a $\mathbf{F}^B = [F_x\ F_y\ F_z]$ je ukupna sila.

Promjena orijentacije kvadkoptera je opisana sljedećom jednadžbom:

$$\frac{d}{dt}(\mathbf{I} \cdot \omega) = -\omega \times (\mathbf{I} \times \omega + \mathbf{M_{sum}}) \tag{2.16}$$

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{2.17}$$

I je matrica inercije kvadkoptera s obzirom na centar mase i uz pretpostavku da je simetričan, što znači da je matrica dijagonalna. ω je kutna brzina kvadkoptera u mobilnom koordinatnom sustavu, a $\mathbf{M_{sum}}$ je suma vanjskih momenata koji djeluju na letjelicu.

Rastavljanjem kutne brzine po x,y i z smjerovima dobivaju se sljedeći izrazi:

$$I_{xx} \cdot \dot{\omega}_x = (I_{yy} - I_{zz})\,\omega_y\,\omega_z + (F_2 + F_3 - F_1 - F_4) \cdot l\cos(45°) \tag{2.18}$$

$$I_{yy} \cdot \dot{\omega}_y = (I_{zz} - I_{xx})\,\omega_x\,\omega_z + (F_3 + F_4 - F_1 - F_2) \cdot l\cos(45°) \tag{2.19}$$

$$I_{zz} \cdot \dot{\omega}_z = M_1 + M_2 + M_3 + M_4 \tag{2.20}$$

Kutne brzine veže matrica **T** (2.13):

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \mathbf{T} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.21}$$

Ovakvo pojednostavljenje vrijedi za male kutove kretanja.

Za linearne brzine vrijedi:

$$m\dot{v}_x = F_z^b(\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi)) \tag{2.22}$$

$$m\dot{v}_y = F_z^b(\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi)) \tag{2.23}$$

$$m\dot{v}_z = F_z^b\cos(\phi)\cos(\theta) - mg \tag{2.24}$$

F_z^b je ukupna sila potiska u z smjeru (u mobilnom koordinatnom sustavu):

$$F_z^b = F_1 + F_2 + F_3 + F_4 \tag{2.25}$$

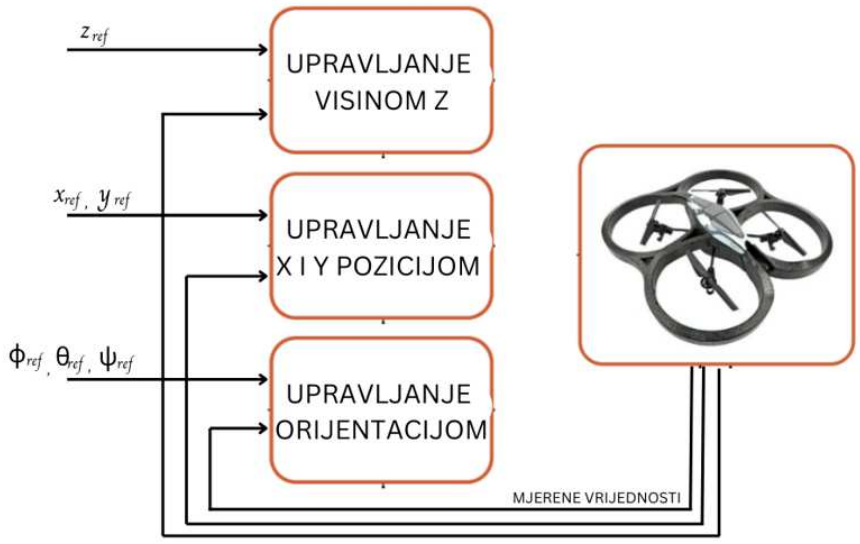
Pozicija u fiksnom koordinatnom sustavu se dobije integriranjem:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.26}$$

3. Upravljanje letjelicom

Upravljanje letjelicom zahtijeva temeljitu analizu kako bi se postigla željena stabilnost leta. UAV kao sustav je nelinearan, multivarijabilan, spregnut i podaktuiran, što znači da je kompleksan za upravljanje [6]. Jedan od mogućih pristupa za upravljanje je kaskadno upravljanje s PID regulatorom u unutarnjoj i vanjskoj petlji, gdje unutarnja petlja upravlja bržim promjenama, a vanjska petlja sporijim. Ključni koraci u ovom procesu su izrada matematičkog modela letjelice, linearizacija tog modela te odabir PID parametara na osnovu identificirane prijenosne funkcije sustava.

Osim toga, druge strategije koje se koriste uključuju linearno-kvadratne regulatore (LQR) [15], adaptivno upravljanje [4], upravljanje s podržanim učenjem [9] i druge. U sljedećem dijelu će se razraditi pristup opisan u [3], prikazan na slici 3.1.



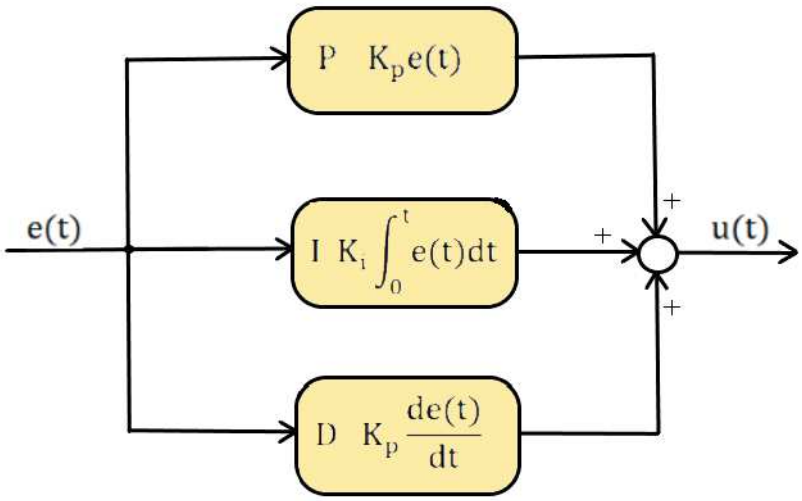
Slika 3.1: Upravljanje letjelicom sustavom PID regulatora u kaskadi

Koristi se kaskadno upravljanje s PID regulatorom, prvo za visinu i orijentaciju, a potom za upravljanje po brzini u x i y smjeru. Iz istog izvora su uzeti parametri modela kvadkoptera *Parrot AR.Drone* koji su potrebni za izračun (masa, duljina kraka,

konstante motora).

3.1. PID regulator

Za upravljanje se bira PID regulator, prikazan na slici 3.2, koji će biti korišten za sve stupnjeve slobode, implementiran u kaskadnoj petlji.



Slika 3.2: Standardni PID regulator u vremenskoj domeni

U vremenskoj domeni upravljačka vrijednost regulatora je zadana na sljedeći način [5]:

$$u(t) = u_p(t) + u_i(t) + u_d(t) = K_p \cdot e(t) + K_i \cdot \frac{1}{T_i} \int_0^t e(t)dt + K_d \cdot \frac{de(t)}{dt} \tag{3.1}$$

U jednadžbi 3.1, $e(t)$ predstavlja regulacijsko odstupanje, tj. razlika reference i izlaza sustava. Konačna upravljačka veličina $u(t)$ je zbroj proporcionalne, integralne i derivabilne komponente (u_p, u_i, u_d).

Umjesto standardne forme često se koristi paralelna, sa zamjenama: $K_r = K_p$, $T_i = \frac{K_p}{K_i}$ i $T_d = \frac{K_d}{K_p}$. Tada je jednadžba ovakvog oblika:

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(t)dt + T_d \frac{de(t)}{dt} \right) \tag{3.2}$$

Da bi se algoritam implementirao u programskom kodu, potrebno je diskretizirati

svaku komponentu:

$$u_p(k) = K_p \cdot e(k) \tag{3.3}$$

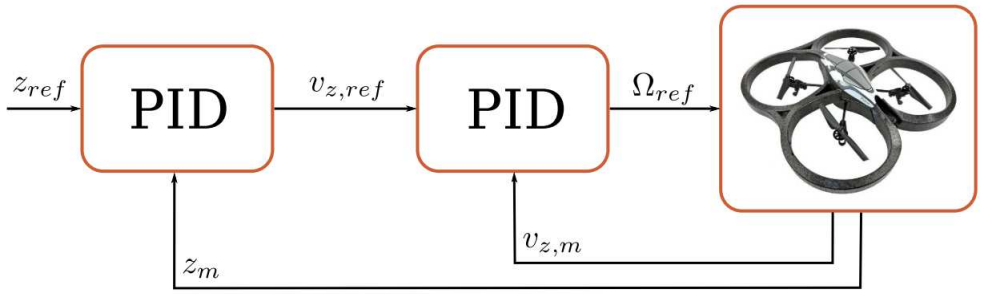
$$u_i(k) = u_i(k-1) + \frac{K_p}{T_i}\, e(k)\, T_s = u_i(k-1) + K_i \cdot e(k) \cdot T_s \tag{3.4}$$

$$u_d(k) = K_p T_d \frac{e(k) - e(k-1)}{T_s} = K_d \cdot \frac{\Delta e(k)}{T_s} \tag{3.5}$$

U ovakvom obliku PID algoritam se jednostavno implementira u Pythonu, pazeći dodatno na krajnje vrijednosti upravljačkog signala.

3.2. Upravljanje visinom

Na slici 3.3 je prikazana upravljačka petlja za visinu, koja se sastoji od procesa (dron), čiji su izlazi mjerena visina (z koordinata pozicije) i brzina (linearna brzina u z smjeru). U unutrašnjoj petlji se regulira brzina, a u vanjskoj visina u z smjeru.



Slika 3.3: Upravljanje visinom kvadkoptera uz PID regulatore u kaskadi

Cilj je naći prijenosnu funkciju $\frac{\Delta v_z(s)}{\Delta \Omega_{ref}}$, koja opisuje promjenu linearne brzine u z-smjeru s obzirom na promjenu referentne rotacijske brzine motora. Potrebno je linearizirati jednadžbu 2.24, koja prikazuje promjenu brzine kvadkoptera u z-smjeru.

Radna točka će biti stanje lebdjenja, pa se prvo može izračunati Ω_0, brzina koju svaki motor treba doseći da bi se ostvarilo stanje lebdjenja, tj. izjednačavanje sile potiska s težinom letjelice. Uz pretpostavku malih promjena kuteva ϕ i θ u stanju

lebdjenja i aproksimacije trigonometrijskih funkcija ($\cos(\phi) = 1$ i $\cos(\theta) = 1$) slijedi:

$$m\dot{v}_z = F_z^b \cos(\phi)\cos(\theta) - mg \tag{3.6}$$

$$m\dot{v}_z = F_z^b - mg \tag{3.7}$$

$$F_z^b = mg \tag{3.8}$$

$$F_z^b = \sum_{i=1}^{4} F_i = b_f \sum_{i=1}^{4} (\Omega_i)^2 = 4b_f\Omega_0^2 \tag{3.9}$$

$$\Omega_0^2 = \frac{mg}{4b_f} \tag{3.10}$$

$$\Omega_0 = 650.95 \text{ rad/s} \tag{3.11}$$

Dobivena je brzina motora potrebna za lebdjenje. Nadalje, radi se linearizacija jednadžbe 2.24 , koristeći razvoj u Taylorov red u odabranoj radnoj točki i aproksimaciju prvog reda:

$$m\dot{v}_z = F_z^b - mg \qquad / : m \tag{3.12}$$

$$\dot{v}_z = \frac{4b_f\Omega^2}{m} - g = f(\Omega) \tag{3.13}$$

$$f(\Omega) = f(\Omega_0) + \left.\frac{df}{d\Omega}\right|_{\Omega_0} \cdot \Delta\Omega \qquad (\Delta\Omega = \Omega - \Omega_0) \tag{3.14}$$

$$= \frac{8b_f\Omega_0}{m} \cdot \Delta\Omega \tag{3.15}$$

$$\Delta\dot{v}_z = 0.03\,\Delta\Omega \tag{3.16}$$

Veličina $\Delta\Omega$ se naziva perturbacijska varijabla i označava odstupanje varijable Ω od mirnog položaja.

Motor kvadkoptera se pojednostavljeno može modelirati na sljedeći način [3]:

$$T_m \cdot \dot{\Omega}_i + \Omega_i = \Omega_{i,ref} \quad i = 1, 2, 3, 4 \tag{3.17}$$

U prethodnoj jednadžbi Ω_i predstavlja rotacijsku brzinu *i*-tog motora mjerenu u rad/s, T_m je vremenska konstanta motora a $\Omega_{i,ref}$ je referentna brzina motora.

Svaki motor proizvodi silu potiska F_i proporcionalnu kvadratu brzine vrtnje, i moment M_i koji se javlja zbog sile otpora:

$$F_i = b_f \cdot \dot{\Omega}_i^2 \cdot \hat{k} \tag{3.18}$$

$$M_i = \zeta_i\, b_f\, b_m \cdot \Omega_i^2 \cdot \hat{k} \tag{3.19}$$

gdje je $\zeta_i = 1$ za $i = 2, 4$ ili $\zeta_i = -1$ za $i = 1, 3$. Jedinični vektor $\hat{k}$ je u smjeru z osi mobilnog koordinatnog sustava, a b_f i b_m su konstante potiska i momenta motora.

Iz jednadžbe 3.17 slijedi:

$$T_m \cdot \dot{\Omega}_i + \Omega_i = \Omega_{i,ref} \qquad (\Omega_i = \Delta\Omega_i, \text{ jednadžba je linearna}) \tag{3.20}$$

$$T_m \cdot \dot{\Delta\Omega}_i + \Delta\Omega_i = \Delta\Omega_{i,ref} \quad /\mathcal{L} \tag{3.21}$$

$$T_m \cdot s \cdot \Omega_i + \Delta\Omega_i = \Delta\Omega_{i,ref} \tag{3.22}$$

$$\Delta\Omega_i(s) = \frac{\Delta\Omega_{i,ref}(s)}{T_m \cdot s + 1} \tag{3.23}$$

Iz 3.16 se Laplaceovom transformacijom dobije prijenosna funkcija:

$$\Delta\dot{v}_z = 0.03\,\Delta\Omega \quad /\mathcal{L} \tag{3.24}$$

$$s \cdot \Delta v_z = 0.03\,\Delta\Omega \tag{3.25}$$

$$\frac{\Delta v_z(s)}{\Delta\Omega(s)} = \frac{0.03}{s} \tag{3.26}$$

Iz jednadžbi 3.23 i 3.26 slijedi:

$$\frac{\Delta v_z}{\Delta\Omega_{ref}} = \frac{\Delta v_z}{\Delta\Omega} \cdot \frac{\Delta\Omega}{\Delta\Omega_{ref}} \tag{3.27}$$

$$= \frac{0.03}{s} \cdot \frac{1}{T_m \cdot s + 1} \tag{3.28}$$

$$= \frac{0.03}{0.0125s^2 + s} = \frac{2.4}{s(s + 80)} \tag{3.29}$$

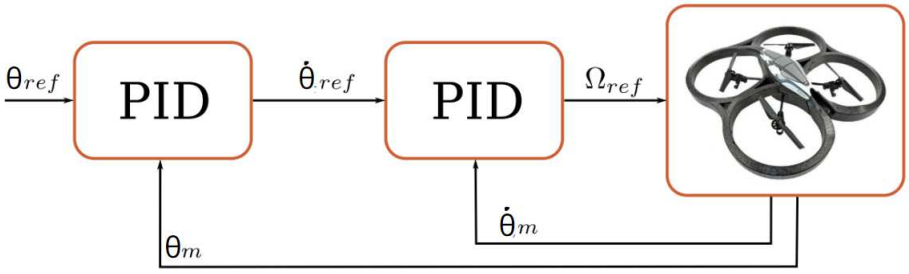
Dobivena je prijenosna funkcija na osnovu koje treba odrediti parametre regulatora unutrašnje petlje. Parametri vanjskog PID regulatora se dobiju tako što se prvo pronađe nova prijenosna funkcija, iz zatvorenog kruga dobivenog PID regulatora i prijenosne funkcije drona.

Tablica 3.1: PID parametri

	Brzina (v_z)			Visina (z)		
Metoda	K_p	K_i	K_d	K_p	K_i	K_d
PID Tuner	63.26	18.19	0	0.5075	1.887	0
Eksperimentalno	75	10	0.41472	0.5	0.125	0

3.3. Upravljanje orijentacijom

Upravljanje po visini i orijentaciji zapravo ide istodobno, jer letjelica ne može održavati željenu visinu bez upravljanja kutovima ϕ, θ i ψ, a orijentacija se testira dok je letjelica



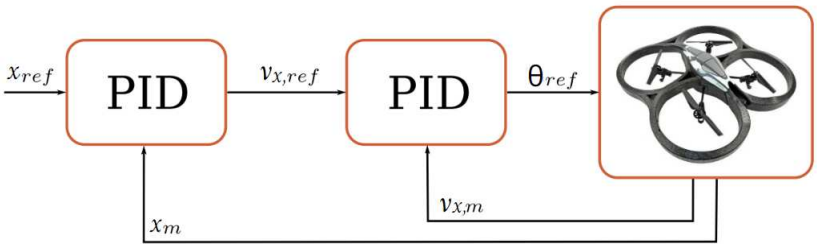
Slika 3.4: Upravljanje orijentacijom kvadkoptera uz PID regulatore u kaskadi

u zraku. Kaskadna petlja za upravljanje poniranjem prikazana je na slici 3.4. Petlja upravljanja je ista i za ϕ, i ψ,.

Traži se prijenosna funkcija $\frac{\Delta\dot{\theta}(s)}{\Delta\Omega_{\theta,ref}(s)}$ a kao izlaz se daje $\Delta\Omega_{ref}$ koja se dodaje referentnoj brzini motora koju generira kontroler visine.

3.4. Upravljanje horizontalnom pozicijom

Za upravljanje x i y pozicijom koristi se sustav prikazan na slici. Upravljačka varijabla za x smjer je promjena kuta poniranja θ, a za y će biti kut valjanja ϕ.



Slika 3.5: Upravljanje pozicijom i brzinom u x smjeru uz PID regulatore u kaskadi

4. Upravljanje sustavom više letjelica

4.1. Višerobotski sustavi

Višerobotski sustav (eng. MRS - Multi-robot System) se općenito može definirati kao dinamička mreža koju karakterizira prostorno raspoređen skup dinamičkih čvorova, koji su koordinirani ciljevima misije i mogućom dinamičkom spregom između čvorova. Inspiracija za istraživanje i razvoj ovakvih sustava često dolazi iz prirode, promatrajući mehanizme i komunikaciju koje koriste npr. roj pčela ili jato riba [12].

Radi boljeg razumijevanja njihove raznolikosti i primjene, bitno je napraviti klasifikaciju ovih sustava. Postoje različite, a u [17] je predložena klasifikacija sa sljedećih pet dimenzija. Višerobotski sustavi mogu biti koordinirani i nekoordinirani (mehanizam koordinacije može biti jednostavan do složen, ovisno o potrebi, a nekoordinirani sustav ima jednostavan dizajn koji ima manji rizik za kvarove). Mogu biti i homogeni (svi roboti imaju iste karakteristike) i heterogeni (različiti roboti, ili samo razlike u načinu upravljanja ili fizičke razlike). Zatim, kooperativni (svi roboti u sustavu međusobno djeluju i rade za zajednički cilj) i kompetititivni (više robota međusobno se natječe kako bi zadovoljili vlastiti interes). Mogu biti komunikacija ili bez komunikacije. Reaktivni i deliberativni

U stranoj literaturi se koriste i nazivi *robot swarm* (koji često znači upotreba više robota, i katkad inspirirano prirodom) i *multi-agent system* (što je širi pojam, može označavati i sustave bez fizičkih entiteta).

4.2. Reynoldsova pravila

S ciljem poboljšanja vjerodostojnosti prikaza svih vrsta pokreta iz prirode u računalnoj animaciji, 1987.g. Craig Reynolds je iznio članak u kojem izlaže set pravila koji se mogu pridati svakom entitetu u grupi [14]. Kao polazni primjer uzima se jato ptica, pa se gibanje naziva *"flocking"* (*eng. flock*), a svaki objekt u njemu je boid (od *eng. bird-*

oid, iako nije bitan tip objekta) Da bi najbolje predstavio ponašanje stvarnog životinje, Reynolds nije stvorio nikakvu središnju kontrolu, već je umjesto toga programirao svakog boida da osjeti vlastitu okolinu i odluči kamo će se kretati. To je rezultiralo fluidnim kretanjem vrlo sličnim pravim jatima ptica. Pravila flokinga prikazana su na slici 4.1 i mogu se opisati na sljedeći način:

i predstavlja promatranog agenta, $j \in 1, ..., N$ su indeksi svih agenata u vidokrugu, $\mathbf{b_i}$ predstavlja vektor pozicije agenta i a $\mathbf{v_i}$ predstavlja vektor brzine agenta i, C_c, C_s, C_a su konstante skaliranja.

1. Separacija: agenti se pokušavaju odmaknuti od obližnjih agenata kako bi izbjegli sudare. Algoritam: Stvaranje vektora brzine u suprotnim smjerovima. Oduzimanje vektora pozicije za agenta koji nije promatrani, i negativan predznak za vektor brzine u suprotnom smjeru.

$$\text{Separacija}(i) = -\frac{C_s}{N-1} \sum_{j \neq i} \frac{\mathbf{b}_j - \mathbf{b}_i}{||\mathbf{b}_j - \mathbf{b}_i||^2}$$

2. Poravnanje: agenti pokušavaju uskladiti svoj smjer brzine s drugim agentima. Ako agent nije promatrani, uzima se u obzir za računanje prosječnog vektora brzine.

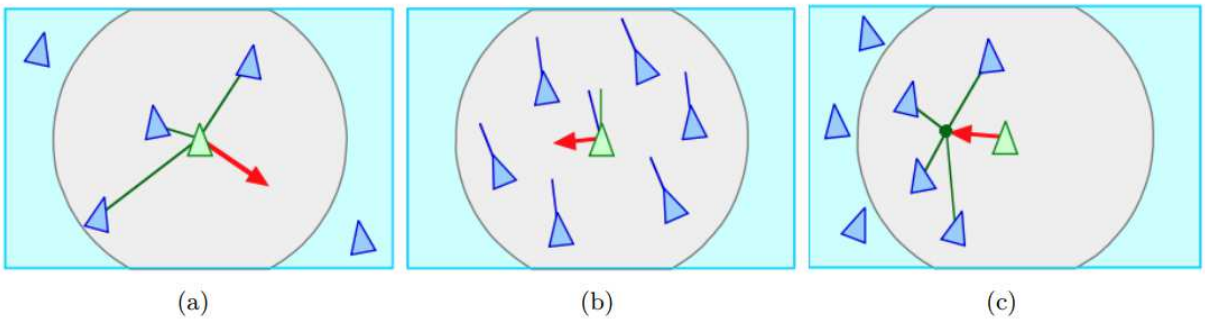
$$\text{Poravnanje}(i) = \frac{C_a}{N-1} \sum_{j \neq i} \mathbf{v}_j - \mathbf{v}_i$$

3. Kohezija: agenti se pokušavaju približiti drugim agentima kako bi formirali jato. Ako agent nije promatrani, računa se prosječna pozicija (centar mase), i oduzima od pozicije trenutnog agenta kako bi se dobio vektor brzine prema centru mase

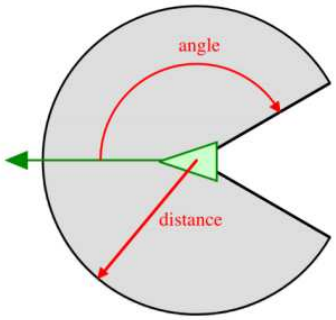
$$\text{Kohezija}(i) = \frac{C_c}{N-1} \sum_{j \neq i} \mathbf{b}_j - \mathbf{b}_i$$

Pokazuje se da su ova tri pravila dostatna za grupiranje i zajedničko gibanje agenata. Svaki boid reagira samo na okolinu u danom krugu, određeno željenim kutom i radijusom, prikazano na slici 4.2. Radijus se mjeri gledajući iz centra boida, a kut iz smjera kretanja. Ostali članovi jata izvan tog vidokruga se ignoriraju.

Osim ova tri pravila, za potrebe simulacija i vjernijeg prikaza ponašanja jata, dodatno se implementiraju još neki zahtjevi:



Slika 4.1: Reynoldsova pravila flockinga: (a) Separacija (b) Poravnanje (c) Kohezija



Slika 4.2: Vidokrug jednog boida

1. Navigacija do određene točke: sustavu se zadaje željena točka u prostoru koja predstavlja cilj kretanja

$$\text{Navigacija}(i, \mathbf{p}_{\text{cilj}}) = \frac{\mathbf{p}_{\text{cilj}} - \mathbf{b}_i}{\|\mathbf{p}_{\text{cilj}} - \mathbf{b}_i\|}\,, \tag{4.1}$$

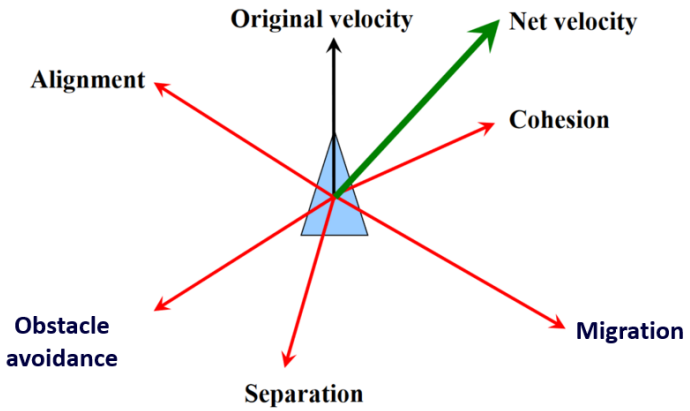
   gdje je $\mathbf{b}_i$ pozicija boida i $\mathbf{p}_{\text{cilj}}$ željena točka u prostoru

2. Izbjegavanje prepreka: u slučaju nailaska na prepreku, treba je uspjeti na određenoj udaljenosti prepoznati i zaobići

$$\text{Izbjegavanje}(i, \mathbf{p}_{\text{prepreka}}) = \begin{cases} \frac{\mathbf{b}_i - \mathbf{p}_{\text{prepreka}}}{\|\mathbf{b}_i - \mathbf{p}_{\text{prepreka}}\|}, & \text{ako je boid udaljen od prepreke} \\ \mathbf{0}, & \text{inače} \end{cases} \tag{4.2}$$

   gdje je $\mathbf{b}_i$ pozicija boida i $\mathbf{p}_{\text{prepreka}}$ pozicija prepreke.

Zbrajanjem doprinosa svih pravila dobiva se ukupni vektor brzine koji djeluje na jedinku 4.3.



Slika 4.3: Kombiniranje vektora svih pravila

Za tri osnovna pravila, rezultantni vektor koji se iterativno računa za svakog agenta će imati ovakav oblik:

$$\mathbf{v}_i = -\frac{C_s}{N-1}\sum_{j\neq i}\frac{\mathbf{b}_j - \mathbf{b}_i}{||\mathbf{b}_j - \mathbf{b}_i||^2} + \frac{C_a}{N-1}\sum_{j\neq i}\mathbf{v}_j - \mathbf{v}_i + \frac{C_c}{N-1}\sum_{j\neq i}\mathbf{b}_j - \mathbf{b}_i \quad (4.3)$$

Obično se dodaju i koeficijenti za svako pravilo kojima se može naglasiti važnost svake komponente.

4.3. Konsenzus protokol

U višerobotskim sustavima, komunikacija između robota je nužna kako bi se ostvarili zajednički ciljevi i postigla usklađenost u radu. Može biti korisna radi koordinacije i podjele zadatka, npr. u heterogenom višerobotskom sustavu u kojem različiti roboti imaju različite zadatke.

Komunikacija je potrebna i zbog sigurnosti i bolje prilagodbe promjenama, npr. u okruženjima u kojima se mogu pojaviti nepredviđene prepreke, roboti moraju moći komunicirati kako bi se prilagodili novim uvjetima.

Nadalje, roboti mogu jednostavno dijeliti informacije o svojem stanju, a pomoću te informacije se mogu rasporediti u zadanu formaciju. U višerobotskom sustavu, protokol konsenzusa (sporazuma) je pravilo interakcije koje određuje razmjenu informacija među agentima [12].

Modeliranje interakcije može biti temeljeno na grafovima, tj. definirano skupom čvorova i bridova, općenito $G_n = (V_n, E_n)$.

Slika 4.4: Neusmjereni i usmjereni graf

Na slici 4.4 je prikazan jednostavan primjer grafa za 3 agenta. Za topologiju komunikacije definiranu usmjerenim grafom vrijedi:

$$(V_n, E_n), V_p = 1, ..., p, E_p \subseteq V_p \times V_p \, , \tag{4.4}$$

tako da za brid $(i, j) \in E_p$ vrijedi da agent j prima informaciju od agenta i.

Za reprezentaciju grafa definira se matrica susjedstva A (*eng. adjacency matrix*). To je kvadratna matrica čiji elementi pokazuju da li su parovi vrhova u grafu susjedni ili ne. Elementi te matrice su $A = [a_{ij}] \in R^{\,p \times p}$, a njihova vrijednost može biti $a_{ij} > 0$ ako $(j, i) \in E_p$ ili $a_{ij} = 0$ inače.

Protokol konsenzusa je onda definiran na sljedeći način:

$$\dot{x}_i(t) = \sum_{j=1}^{n} a_{ij}[(x_j(t) - x_i(t))] \, , \tag{4.5}$$

gdje je x_i stanje robota i. Konsenzus je postignut ako $\forall x_i(t)$ i $\forall i, j \in 1, ..., n$ vrijedi $|x_j(t) - x_i(t)| \to 0$ za $t \to \infty$.

U višerobotskim sustavima često se želi postići određeni organizirani raspored ili geometrija među robotima, tj. formacija. Formacije mogu biti fiksne ili dinamičke, ovisno o potrebama zadatka. Fiksne formacije podrazumijevaju statički raspored robota tijekom cijelog zadatka, dok dinamičke formacije omogućuju robotima da se prilagode promjenama.

Formacija se može specificirati na sljedeći način:

$$\begin{aligned} D = &\{d_{ij} \in R \mid d_{ij} > 0, i, j = 1, ..., n, i \neq j\} \text{ t.d.} \\ &\exists \, \xi_1, ..., \xi_n \in R^p, \; ||\xi_i - \xi_j|| = d_{ij} \; \forall i, j \end{aligned} \tag{4.6}$$

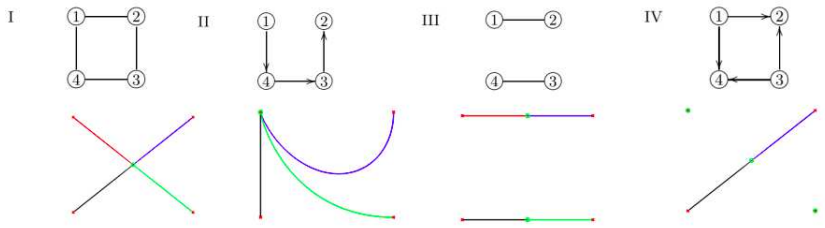
U izrazu 4.6 d_{ij} predstavlja udaljenost između agenata u formaciji a ξ_i poziciju agenta i u formaciji. Za kontrolu formacije konsenzusom može se dakle koristiti modificirana jednadžba 4.5:

$$\dot{x}_i(t) = \sum_{j=1}^{n} a_{ij}[(x_j(t) - x_i(t)) - (\xi_j(t) - \xi_i(t))] \, , \tag{4.7}$$

Konsenzus protokoli se mogu koristiti i za randevu problem, tj. za sustave u kojima više robota istovremeno dolazi na zajedničku a priori nepoznatu lokaciju određenu timskim pregovorima 4.5.

$$r_i = [x_i \; y_i]^T \in R^2 \tag{4.8}$$

$$\dot{r_i} = \sum_{j=1}^{n} a_{ij}(r_j - r_i) \tag{4.9}$$



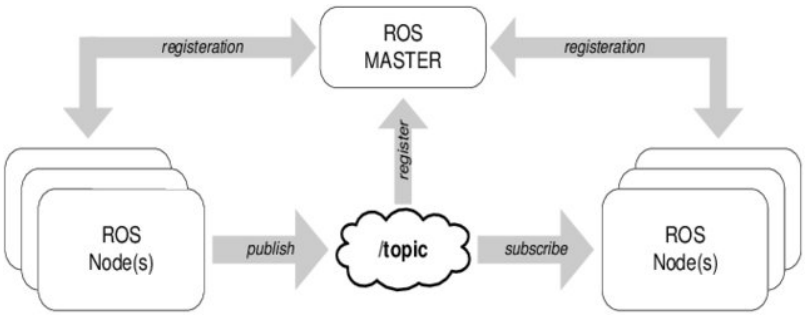
Slika 4.5: Primjer komunikacijkih topologija

5. Rezultati

5.1. Simulacijsko okruženje

Za daljnje simulacije se koristilo razvojno okruženje RotorS [8]. To je modularno okruženje za simulaciju MAV (Micro Aerial Vehicle) letjelica koji omogućuje brz početak u korištenju različitih MAV-ova i kontrolera. RotorS je dobra početna točka za upuštanje u rješavanje zadataka više razine, kao što su detekcija prepreka, planiranje putanje, korištenje više letjelica ili SLAM (Simultaneous Localization and Mapping) zadatke.

Repozitorij sadrži fizički točne modele letjelica kao što su AscTec Hummingbird, AscTec Pelican ili AscTec Firefly. Također na raspolaganju su i mnogi senzori kao što je IMU, odometrijski senzor i VI-senzor, koji se mogu postaviti na letjelicu. Za svaku letjelicu je definirano i upravljanje. RotorS uključuje i mnoge osnovne launch datoteke, od najjednostavnije kojom se samo otvori Gazebo i pozicionira letjelica u okruženju, do onih u kojima se definira više letjelica i dodani sustavi upravljanja, definiranja kretanja.

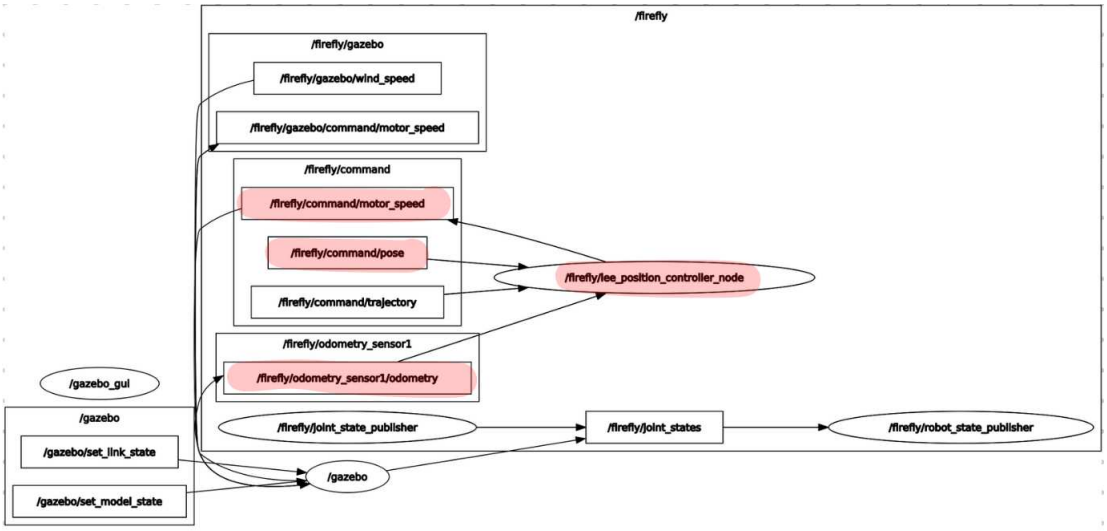


Slika 5.1: Model komunikacije u ROS okruženju

ROS (Robotic Operating System) je sustav otvorenog koda koji služi za razvoj robotskih sustava. Glavna funkcija ROS-a je omogućiti komunikaciju između korisnika, robota i opreme izvan računala, poput senzora, kamera itd. ROS sustav sastoji se od niza malih i neovisnih programa koji mogu istovremeno raditi, nazvanih ROS čvorovi

(*eng. node*), pri čemu svaki čvor komunicira s drugim čvorovima slanjem ili primanjem poruka *(eng. message)*. Neki čvorovi pružaju informacije drugim čvorovima putem ROS teme *(eng. topic)*. ROS glavni čvor odgovoran je za uspostavljanje komunikacije između čvorova te pruža usluge imenovanja i registracije čvorovima u ROS sustavu. Također prati čvorove izdavače i pretplatnike na teme 5.1.

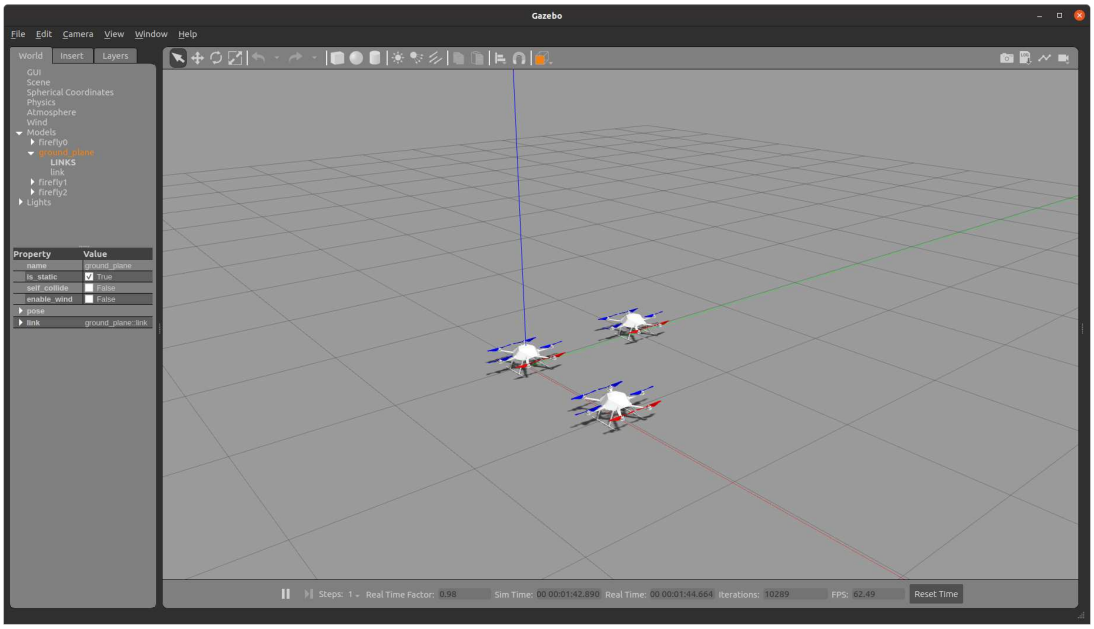
Također koristan ROS alat je *rqt_graph*, koji generira graf aktivnih čvorova, gdje elipse predstavljaju čvor, pravokutnik temu, a strelice predstavljaju tok pretplata 5.2.



Slika 5.2: Graf ROS čvorova nakon pokretanja jedne Firefly letjelice

Na slici 5.3 je prikazan Gazebo simulator s tri letjelice. Sustav se pokreće *launch* datotekom u kojoj je definiran model svijeta (*world* datoteka), modeli letjelica (*URDF* datoteke) i njihovi upravljački čvorovi. Kao model letjelice odabran je heksakopter Firefly. Željeni model se jednostavno mijenja promjenom argumenta *mav_name* u *launch* datoteci. Upravljanje po poziciji izvodi čvor *lee_position_controller_node*, a za svaki model letjelice su već pripremljeni potrebni parametri regulatora. Pokretanje letjelice se postiže slanjem poruke tipa PoseStamped na temu *firefly/command/pose* a upravljački čvor se pobrine da se potrebne vrijednosti pošalju na *firefly/command/motor_speed*

Simulacija bespilotnih letjelica u ROS-u i Gazebu uključuje jedinstvene izazove u usporedbi s prizemljenim robotima, posebno kada je u pitanju otkrivanje prepreka. U idućem dijelu su prikazani rezultati provedenih simulacija.



Slika 5.3: Gazebo simulator s tri Firefly letjelice

5.2. Reynoldsova pravila - simulacija

5.2.1. Testiranje osnovnih pravila

Simulacija pravila je prvo napravljena u simulatoru Stage. To je 2D simulator u kojem su agenti, roboti jednostavne točke bez mase (postižu zadanu brzinu trenutno). U tom okruženju je i izbjegavanje prepreka jednostavno, Stage simulator izdaje poruke za temu *map*, u kojoj je dana mreža popunjenosti. Kod letjelica je ovaj zadatak zahtjevniji.

Implementirana je funkcija koja računa brzine koristeći formulu 4.3. Definirane su C_c, C_s, C_a za koheziju, separaciju i poravnanje.

U Python implementaciji, cilj ovog algoritma je izračunati krajnji vektor brzine, kao zbroj svakog pojedinačnog pravila. Dobiveni rezultat je zapravo linearna brzina, koja se može prikazati kao dio *Twist* poruke iz *geometry_msgs* paketa. Pošto je upravljanje po poziciji, potrebno je tu poruku pretvoriti u poruku pogodnu za slanje na *firefly/command/pose*, što se može napraviti na sljedeći način:

$$x = x + v_x \cdot \Delta t \tag{5.1}$$

$$y = y + v_y \cdot \Delta t \tag{5.2}$$

$$z = z + v_z \cdot \Delta t \tag{5.3}$$

$$\tag{5.4}$$

Za bolju ilustraciju efekta blizine prepreke ili susjednog agenta, umjesto linearne

funkcije odabrana je Gauss funkcija, tj. inverz. Invertirana Gaussova funkcija se koristi za skaliranje efekta usporavanja kada se agent približi danoj točki.

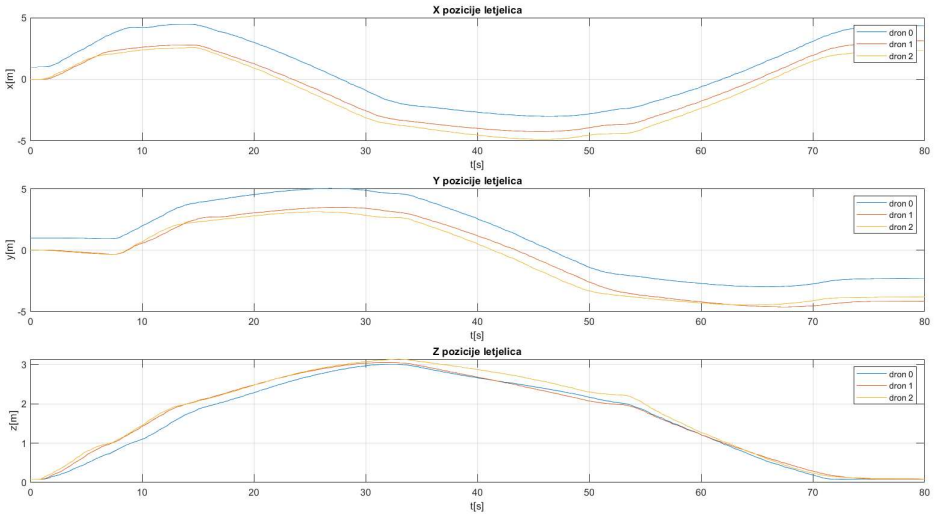
Ove funkcije ne nameću eksplicitno fiksnu udaljenost između agenta, nego im je cilj postići prirodno izgledajuće ponašanje jata balansiranjem kohezije i sila razdvajanja. Stvarna udaljenost između tijela rezultat je međusobnog djelovanja tih sila.

5.2.2. Navigacija

Kao dodatno pravilo ponašanja, uvodi se navigacija i implementira pomoću 4.1. Dodaje se set točaka u prostoru koje će predstavljati $\mathbf{p}_{\text{cilj}}$.

Svako pravilo, tj. njegov izračun brzine je skaliran s konstantom, i dodano je C_n za navigaciju. Da bi se prikazao utjecaj svake sile, mijenjane su vrijednosti konstanti.

Navigacija je prvo izvedena na način da samo jedna letjelica prima informaciju o sljedećoj ciljnoj točki. Na taj način druge dvije letjelice trebaju pratiti njeno kretanje i održati grupno ponašanje. Rezultati kretanja su prikazani na slici 5.4.



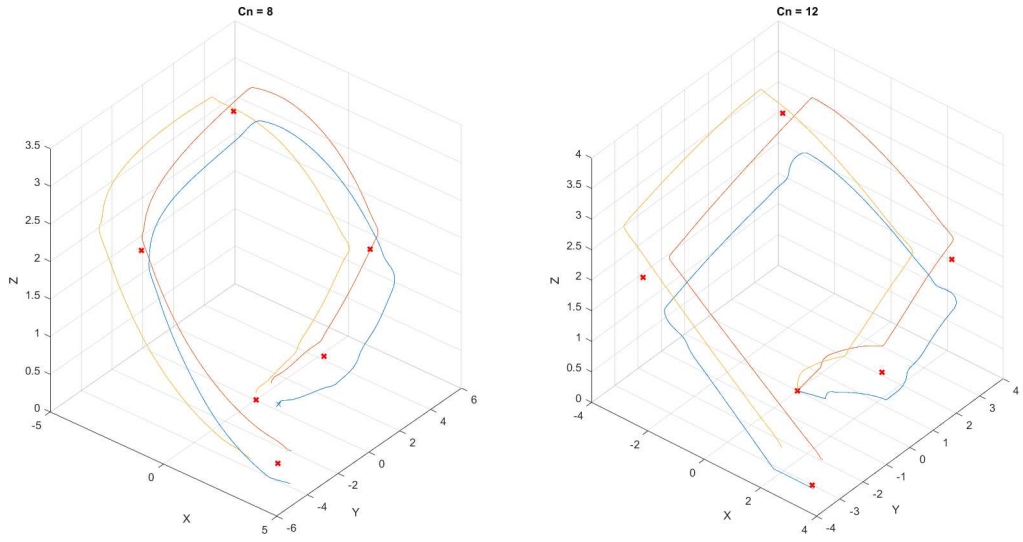
Slika 5.4: Kretanje sustava po putanji $\mathbf{p}_{\text{cilj}} = [(0,0,0),(3,3,1),(-3,-3,2),(0,0,2)]$

Potom je navigacija izvedena tako da svaka letjelica zna cilj kretanja, i tijekom kretanja trebaju održati potrebnu separaciju, koheziju i poravnanje. Zadana je putanja kretanja kao skup točaka u prostoru:

$$\mathbf{p}_{\text{cilj}} = [(0,0,0),(3,0,1),(3,3,2),(-3,3,3),(-3,-3,2),(3,-3,0)]$$

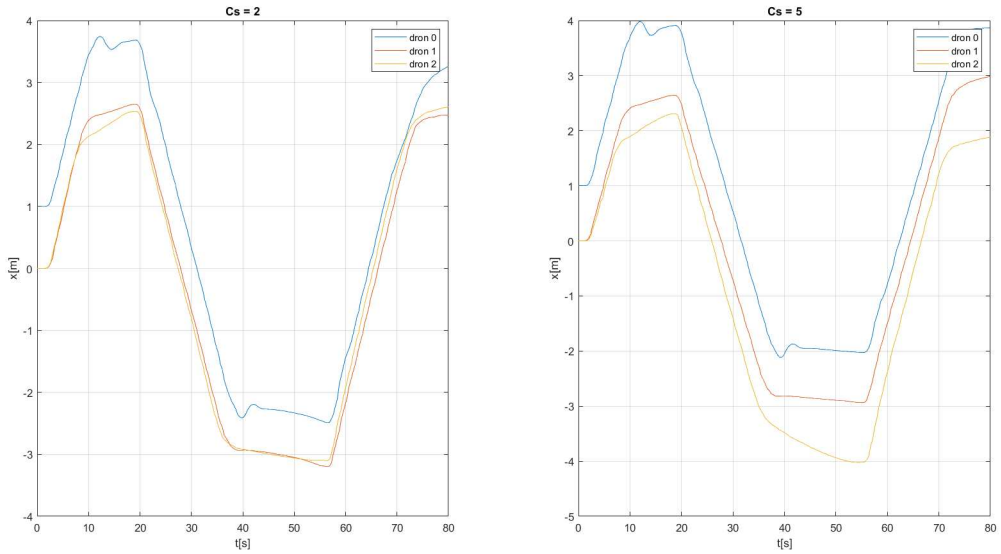
Na slici 5.5 su prikazani rezultati simulacije kretanja po zadanoj putanji uz promjenu parametra C_n (konstanta navigacije). Vidi se da kada je vrijednost tog parame-

tra veća, generira se veća vrijednost komponente navigacije u cjelokupnom vektoru, te letjelice brže i izravnije postižu cilj.



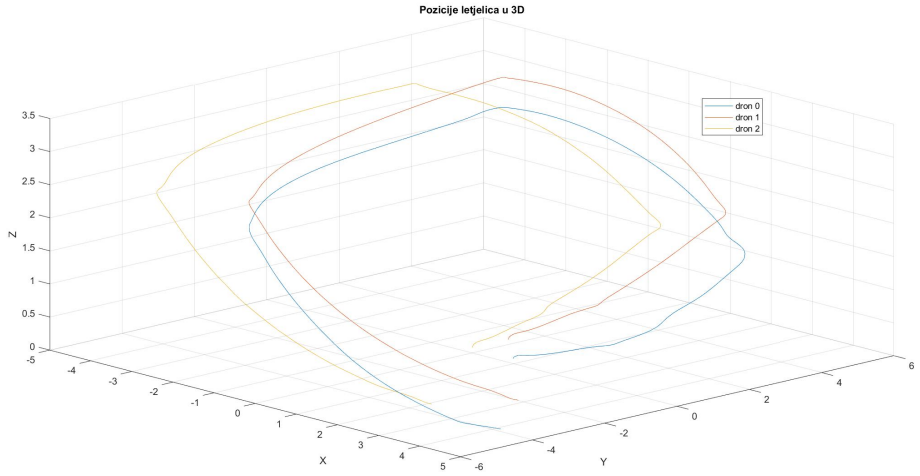
Slika 5.5: Kretanje uz parametre C_s = 3, C_c = 3, C_a = 0.3 i promjenu C_n

Na slici 5.6 prikazane su pozicije letjelica za promjenu parametra Cs (konstanta separacije).

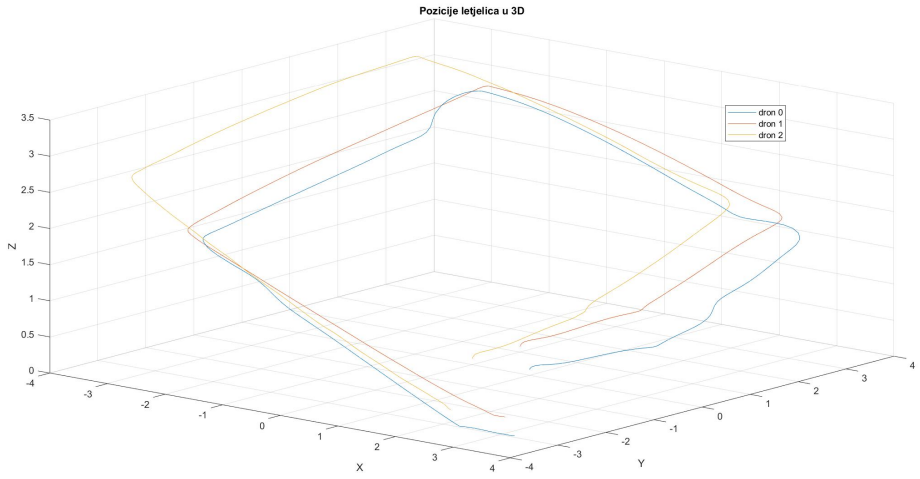


Slika 5.6: Kretanje uz parametre C_c = 3, C_a = 0.3, C_n = 10 i promjenu C_s

Na slikama 5.7 i 5.8 prikazane su pozicije letjelica za promjenu parametra Cc (konstanta kohezije).



Slika 5.7: Kretanje uz parametre C_s = 3, C_a = 0.3, C_n = 10 i promjenu C_c = 3



Slika 5.8: Kretanje uz parametre C_s = 3, C_a = 0.3, C_n = 10 i promjenu C_c = 6

Na slikama se vidi efekt promjene parametara na udaljenost letjelica tijekom kretanja.

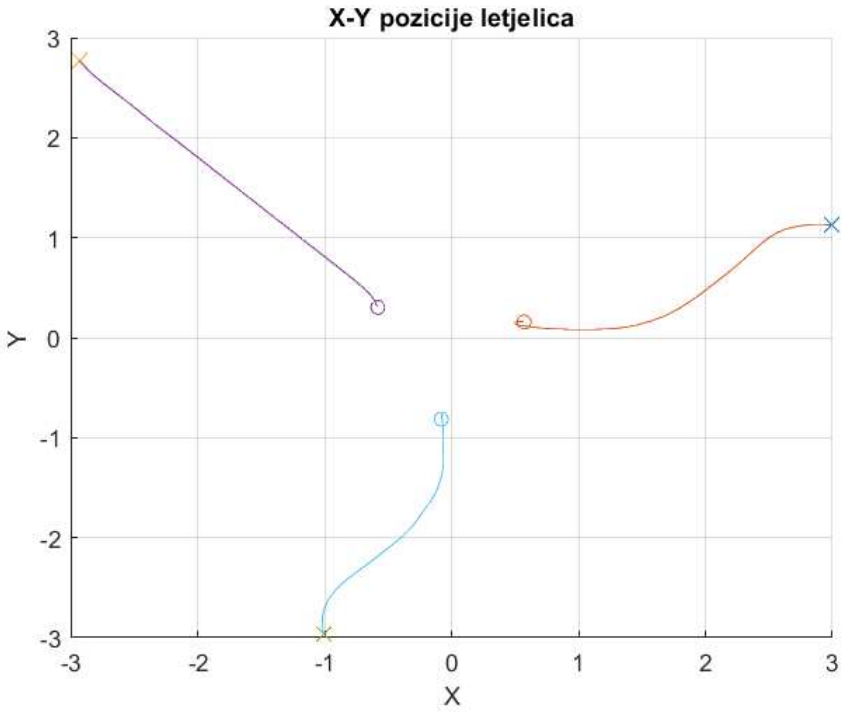
5.2.3. Izbjegavanje prepreka

Prepoznavanje i izbjegavanje prepreka se može izvesti na više načina, koristeći lidar, kameru, OctoMap, SLAM ili fuziju više senzora za veću točnost i pouzdanost. Ovdje će se za prepoznavanje prepreke korisiti Visual-Inertial (VI-) senzor koji pruža podatake s IMU-a i stereo-kamere. Senzor se može postaviti na jednu letjelicu koja se ponaša kao lider. Dobiju se podaci s teme */firefly0/vi_sensor/camera_depth/depth/points*,

tipa poruke *PointCloud2*. Računa se novi vektor na temelju 4.2 i dodaje u ukupni vektor brzine.

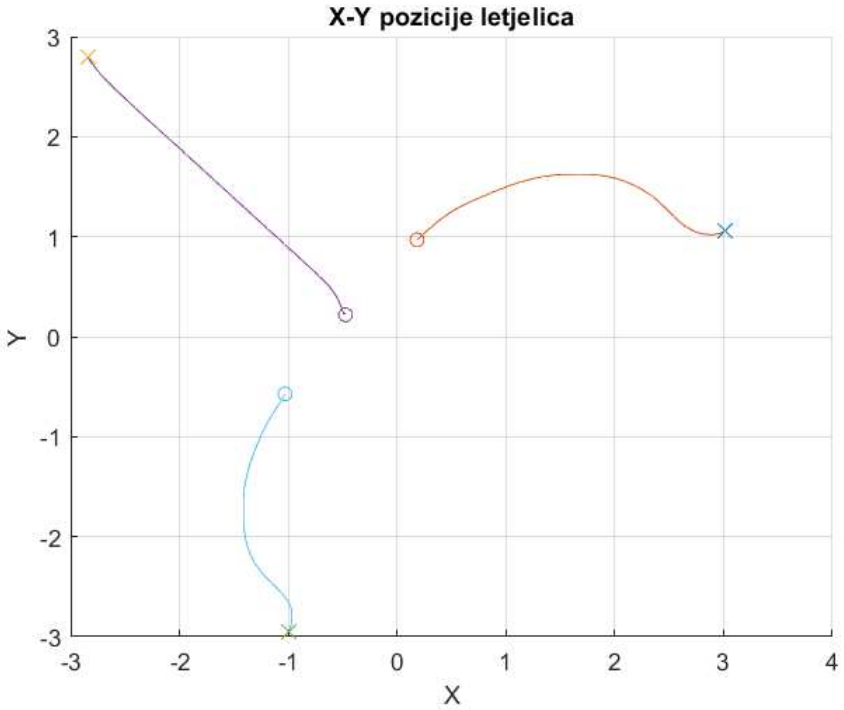
5.3. Konsenzus protokol - simulacija

Koristeći konsenzus protokol i implementaciju jednadžbe 4.5, postignuto je okupljanje letjelica u jednu točku. Dodatno, uz definiranu matricu D 4.6, postignuta je formacija u trokut ili liniju. Na slikama 5.9 i 5.10 prikazano skupljanje tri letjelice u zadanu formaciju.



Slika 5.9: Letjelice u formaciji trokut

U ovom jednostavnom primjeru su pretpostavke idealnih uvjeta, tj. da je komunikacija točna i pouzdana, što je naravno izazov za kompleksniji sustav. Također, za sustav s više agenata nije isplativo, a ponekad niti moguće imati potpuno povezan graf.



Slika 5.10: Letjelice u formaciji linija

6. Zaključak

U ovom radu istraženo je korištenje sustava s više bespilotnih letjelica. Rezultati ukazuju na potencijal i mogućnost korištenja takvih sustava za postizanje složenih zadataka u različitim domenama.

Matematičko modeliranje letjelice predstavilo je temeljni korak u razumijevanju njihove kinematike, dinamike i upravljačkih sustava. Precizno modeliranje omogućuje bolje razumijevanje ponašanja letjelica u različitim uvjetima, a rezultati simulacija potvrdili su valjanost razvijenog matematičkog modela.

Daljnji razvoj i istraživanje sustava s više UAV letjelica ima nekoliko perspektiva. Optimizacija algoritama za koordinaciju i upravljanje letjelicama može poboljšati učinkovitost i pouzdanost sustava. Također, dodavanje dodatnih letjelica u sustav može proširiti mogućnosti i primjene, ali istovremeno iziskuje složenije strategije koordinacije.

Kroz daljnja istraživanja i primjenu naprednih tehnologija, očekuje se da će sustavi s više UAV letjelica postati sve značajniji i češći u budućnosti, pridonoseći napretku različitih industrija i društvenih područja.

LITERATURA

[1] Crazyflie 2.1. https://www.bitcraze.io/products/crazyflie-2-1/. Accessed: 2024-02-19.

[2] Gazebo. https://gazebosim.org/home. Accessed: 2024-02-19.

[3] Aerial robotics: Laboratory exercises. https://www.fer.unizg.hr/predmet/zrarob, 2023.

[4] Agho, C. Dynamic model and control of quadrotor in the presence of uncertainties. Master's thesis, University of South Carolina, 2017. URL https://scholarcommons.sc.edu/etd/4069.

[5] Baotic, M., Mišković, N., Lešić, V., i Novoselnik, B. Automatsko upravljanje: Predavanja. https://www.fer.unizg.hr/predmet/autupr, 2023.

[6] Bogdan, B. i Orsag, M. Zračna robotika: Predavanja. https://www.fer.unizg.hr/predmet/zrarob, 2023.

[7] Bresciani, T. Modelling, identification and control of a quadrotor helicopter. Magistarski rad, Lund University, 2008.

[8] Furrer, F., Burri, M., Achtelik, M., i Siegwart, R. *Robot Operating System (ROS): The Complete Reference (Volume 1)*, poglavlje RotorS—A Modular Gazebo MAV Simulator Framework, stranice 595–625. Springer International Publishing, Cham, 2016. ISBN 978-3-319-26054-9. doi: 10.1007/978-3-319-26054-9_23. URL http://dx.doi.org/10.1007/978-3-319-26054-9_23.

[9] Koch, W., Mancuso, R., West, R., i Bestavros, A. Reinforcement learning for uav attitude control, 2018.

[10] LARICS. Urs aerial. https://github.com/larics/urs_aerial.git, 2023.

[11] Orsag, M. i Bogdan, S. Influence of forward and descent flight on quadrotor dynamics. *Recent Advances in Aircraft Technology*, stranice 141–156, 2012.

[12] Petrović, T., Nađ, D., i Bogdan, B. Višerobotski sustavi: Predavanja. `https://www.fer.unizg.hr/predmet/vissus`, 2023.

[13] Proskurnikov, A. i Cao, M. *Consensus in Multi-Agent Systems*. 11 2016. ISBN 9780471346081. doi: 10.1002/047134608X.W8332.

[14] Reynolds, C. W. Flocks, herds and schools: A distributed behavioral model. U *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, stranice 25–34, 1987.

[15] Sabatino, F. Quadrotor control: modeling, nonlinearcontrol design, and simulation. Magistarski rad, 2015. URL `https://api.semanticscholar.org/CorpusID:61413561`.

[16] Skorobogatov, G., Barrado, C., i Salamí, E. Multiple uav systems: A survey. *Unmanned Systems*, 8(02):149–169, 2020.

[17] Verma, J. i Ranga, V. Multi-robot coordination analysis, taxonomy, challenges and future scope. *Journal of Intelligent And Robotic Systems*, 102, 2021. doi: 10.1007/s10846-021-01378-2.

UPRAVLJANJE SUSTAVOM VIŠE BESPILOTNIH LETJELICA

Sažetak

Ovaj rad istražuje korištenje sustava s više bespilotnih letjelica (UAV) kroz matematičko modeliranje, simulacije i analizu. Kroz precizno modeliranje pojedinačne letjelice i simulacije, potvrđena je valjanost matematičkog modela i pružen uvid u ponašanje letjelica. Nadalje, simulacija sustava s tri UAV letjelice demonstrirala je njihovu sposobnost suradnje i koordinacije, uz primjenu pravila izbjegavanja sudara i prepreka. Rezultati ukazuju na potencijal korištenja sustava s više UAV letjelica za postizanje složenih zadataka u različitim područjima, te na važnost matematičkog modeliranja i simulacija u razvoju takvih sustava.

Ključne riječi: UAV; dron; višerobotski sustav; PID regulator; konsenzus; ROS; Gazebo

MULTI-UAV SYSTEM CONTROL

Abstract

This thesis explores the utilization of multi-unmanned aerial vehicle (UAV) systems through mathematical modeling, simulations, and analysis. By accurately modeling an individual UAV and conducting simulations, the validity of the mathematical model was confirmed, providing insights into the behavior of the vehicles. Furthermore, the simulation of a three-UAV system demonstrated their ability to collaborate and coordinate, employing collision avoidance and obstacle rules. The results indicate the potential of multi-UAV systems in accomplishing complex tasks across various domains, highlighting the significance of mathematical modeling and simulations in their development.

Keywords: UAV; drone; multi-robot system; PID controller; consensus; ROS; Gazebo