Simulacija letjelice koja se kreće u dvodimenzionalnom prostoru

Kokić, Nikola

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni prijediplomski studij Elektrotehnika i informacijska tehnologija / Računarstvo

SIMULACIJA LETJELICE KOJA SE KREĆE U DVODIMENZIJALNOM PROSTORU

Završni rad

Nikola Kokić

Osijek, 2024

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P: Obrazac za ocjenu završnog rada na sveučilišnom prijediplomskom studiju

Ocjena završnog rada na sveučilišnom prijediplomskom studiju

Ime i prezime pristupnika:	Nikola Kokić
Studij, smjer:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. pristupnika, god.	R4660, 28.07.2021.
JMBAG:	0165091303
Mentor:	prof. dr. sc. Robert Cupec
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Simulacija letjelice koja se kreće u dvodimenzionalnom prostoru
Znanstvena grana završnog rada:	**Procesno računarstvo (zn. polje računarstvo)**
Zadatak završnog rada:	Izraditi simulator letjelice s automatskim pozicioniranjem koja se kreće u dvodimenzionalnom prostoru uključujući i simulaciju vjetra. Simulator treba imati grafičko korisničko sučelje koje omogućuje prikaz trenutne vrijednosti pozicije letjelice, njezine brzine te upravljačkih signala upravljačkog algoritma kao i zadavanje vrijednosti parametara upravljačkog algoritma. Nadalje, treba omogućiti snimanje vrijednosti navedenih signala tokom simulacije te njihov zapis u datoteku. Tema rezervirana za: Nikola Kokić

Datum prijedloga ocjene završnog rada od strane mentora:	09.09.2024.
Prijedlog ocjene završnog rada od strane mentora:	Izvrstan (5)
Datum potvrde ocjene završnog rada od strane Odbora:	16.09.2024.
Ocjena završnog rada nakon obrane:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio sveučilišni prijediplomski studij:	24.09.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O IZVORNOSTI RADA

Osijek, 24.09.2024.

Ime i prezime Pristupnika:	Nikola Kokić
Studij:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R4660, 28.07.2021.
Turnitin podudaranje [%]:	3

Ovom izjavom izjavljujem da je rad pod nazivom**: Simulacija letjelice koja se kreće u dvodimenzionalnom prostoru**

izrađen pod vodstvom mentora prof. dr. sc. Robert Cupec

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

SADRŽAJ

1. UVOD

Mnogi prirodni procesi su previše složeni za čovjeka da bi njima sam upravljao. Automatsko upravljanje je danas vrlo potrebno, jer olakšava čovjeku obavljanje mnogih zadataka [1][2]. Upravljački krugovi, odnosno regulatori, su jezgra automatizacije raznih industrijskih sustava kao što su kemijski procesi, elektroenergetika i proizvodnja [1][2]. Regulatori se također koriste u raznim vozilima, kućanskim aparatima, robotici pa čak i u video igrama [1][2][3], to jest simulacijama.

Kako raste složenost sustava kojima se upravlja, tako raste i složenost regulatora. Ovisno o sustavu, podešavanje parametara regulatora i testiranje sustava može biti dugotrajno. Ukoliko sustav možemo modelirati linearnim matematičkim funkcijama, moguće je ubrzati proces podešavanja i testiranja. Međutim, u praksi se vrlo često susrećemo s nelinearnim sustavima, što otežava primjenu pojednostavljenih modela [1].

U ovom radu razmatra se problem simulacije letjelice u prostoru s dvije dimenzije i uz djelovanje vanjskih sila kao što su gravitacija i vjetar. Podešavanje regulatora za letjelicu, kao što je kvadrokopter, može vrlo lako dovesti do nepredviđenih posljedica ukoliko je osoba koja ih izrađuje neiskusna. Iznenadna destabilizacija u zraku može dovesti do kolizije, materijalne štete na letjelici, pa čak i ugroziti čovjeka. Kako bi se izbjegle nesreće te znatno ubrzao proces podešavanja i testiranja koriste se simulacije. U razvoju video igara može biti vrlo zahtjevno uvesti automatiziranog igrača da upravlja simuliranim objektom tako da se kreće glatko, robusno i bez iznenadnih impulsa tako da njegovo kretanje izgleda uvjerljivo. Svrha izrade ove simulacije jest dovesti do boljeg razumijevanja rada regulatora te ispitati automatsko upravljanje kao alat za izradu automatiziranog igrača u svijetu razvoja video igara.

U nastavku je obrađena metodologija pri čemu je prikazano teorijsko znanje o signalima, sustavima, regulatorima i ostalim elementima potrebnim za izradu matematičkog modela letjelice. Također je matematički opisana letjelica i struktura regulatora. Zatim se u poglavlju implementacije opisuju implementacija teorijskog znanja, osnove korištenja Unity alata za razvoj video igara te način kako se mogu izraditi model letjelice i regulatori u C# programskom jeziku. Nakon toga se prikazuju rezultati simulacije za različite parametre letjelice i simulacijskog okružja.

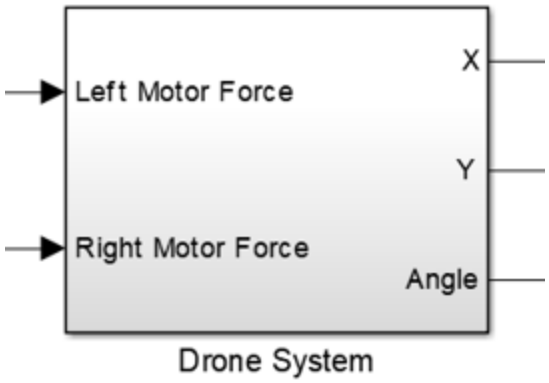
1.1. Zadatak završnog rada

Izraditi simulator letjelice s automatskim pozicioniranjem koja se kreće u dvodimenzionalnom prostoru uključujući i simulaciju vjetra. Simulator treba imati grafičko korisničko sučelje koje omogućuje prikaz trenutne vrijednosti pozicije letjelice, njezine brzine te upravljačkih signala upravljačkog algoritma kao i zadavanje vrijednosti parametara upravljačkog algoritma. Nadalje, treba omogućiti snimanje vrijednosti navedenih signala tokom simulacije te njihov zapis u datoteku.

2. METODOLOGIJA

Letjelica ima 2 motora paralelno usmjerena prema gore. Motori se zajedno translatiraju i rotiraju s letjelicom. Oba motora su udaljena od centra letjelice na njezinoj lokalnoj x osi l metara, jedan s jedne, a drugi s druge strane centra. Na lokalnoj y osi, oba motora su na poziciji nula metara od centra.

Letjelicu gledamo kao točkasto kruto tijelo na koje djeluje gravitacija s ubrzanjem od 9,81 m/s^2. Parametri letjelice su masa (m), linearni (d_{lin}) i kutni (d_{ang}) koeficijent trenja. Sustav letjelice, na slici 2.1, je višeulazni i višeizlazni, odnosno MIMO sustav (*eng. Multiple Input Multiple Output*). Ulaz letjelice su sile lijevog i desnog motora. Model letjelice ima 3 izlaza, pozicija definirana x i y koordinatom te kut nagiba između okomice letjelice i vertikalne osi globalnog koordinatnog sustava pridruženog prostoru u kojemu se letjelica kreće.



Slika 2.1 Blok sustava letjelice

Svaki izlaz predstavlja drugi integral ubrzanja u smjeru neke od osi te kutnog ubrzanja. Rotacijsko gibanje (2-1) se ostvaruje razlikom sila [2] između motora, razlika je pojačana udaljenošću l motora od centra s dodanim negativnim utjecajem kutnog trenja d_{ang} [4]. Moment inercije J ovisi o masi i obliku letjelice.

$$J\theta'' = l(F_L - F_D) - d_{ang}\theta' \tag{2-1}$$

Sila vertikalnog gibanja (2-2) predstavlja zbroj sila lijevog (F_L) i desnog (F_D) motora u vertikalnom smjeru, utjecaja translacijskog trenja d_{lin}, koje djeluje u suprotnom smjeru, te utjecaja gravitacijske sile koja djeluje prema dolje, gdje je g gravitacijsko ubrzanje. Većom silom motora se

postiže veća vertikalna sila, a veći nagib letjelice smanjuje utjecaj motornih sila u vertikalnom smjeru. [4]

$$my'' = (F_L + F_D)cos\theta - d_{lin}y' - mg \tag{2-2}$$

Horizontalna sila (2-3) je zbroj [2] komponenata ulaznih sila u horizontalnom smjeru i utjecaja translacijskog trenja, koje djeluje u suprotnom smjeru. Apsolutna horizontalna sila se povećava većim apsolutnim nagibom letjelice. [4]

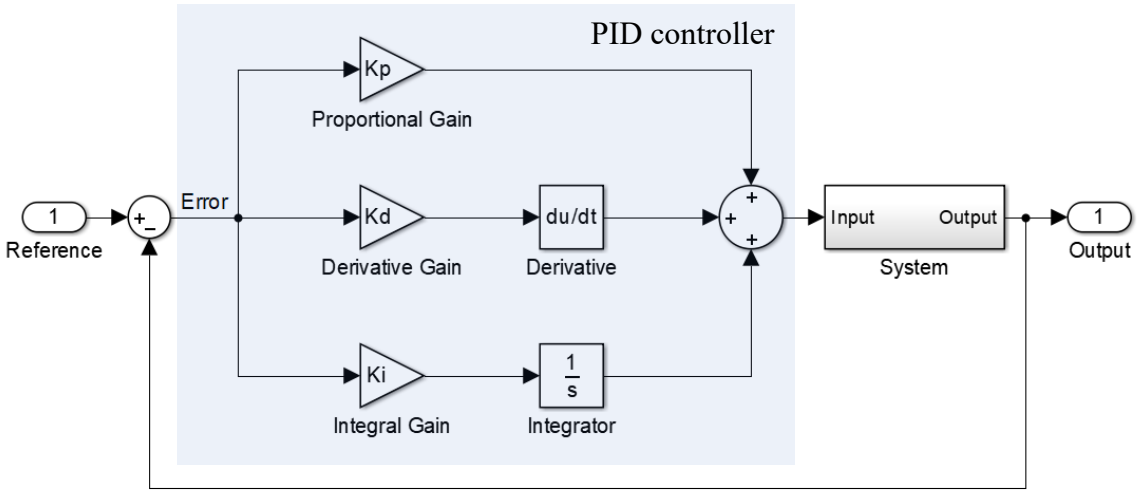
$$mx'' = (F_L + F_D)sin\theta - d_{lin}x' \tag{2-3}$$

Iz gore navedenih jednadžbi gibanja može se zaključiti da nije moguće nezavisno upravljati pozicijom i kutom, pošto se horizontalna pozicija letjelice može promijeniti jedino ako je kut θ različit od nule pod pretpostavkom da na letjelicu ne djeluje neka vanjska sila, kao npr. vjetar. Isto tako, vrijedi i obrnuto: nije moguće da kut θ bude različit od nule, a da letjelica stoji mirno na zadanoj poziciji. Zadatak je dovesti sustav u referentno stanje, da izlazna x i y pozicija sustava bude jednaka referentnoj poziciji, stoga nam je za svaki izlaz potreban regulator, koji dovodi sustav u željeno stanje, odnosno minimizira pogrešku. Pogreške x i y pozicije su opisane kao razlike referentne pozicije x_{ref} i izlazne pozicije x (2-4) te razlika referentne pozicije y_{ref} i izlazne pozicije y (2-5).

$$E_x(t) = x_{ref}(t) - x(t) \tag{2-4}$$

$$E_y(t) = y_{ref}(t) - y(t) \tag{2-5}$$

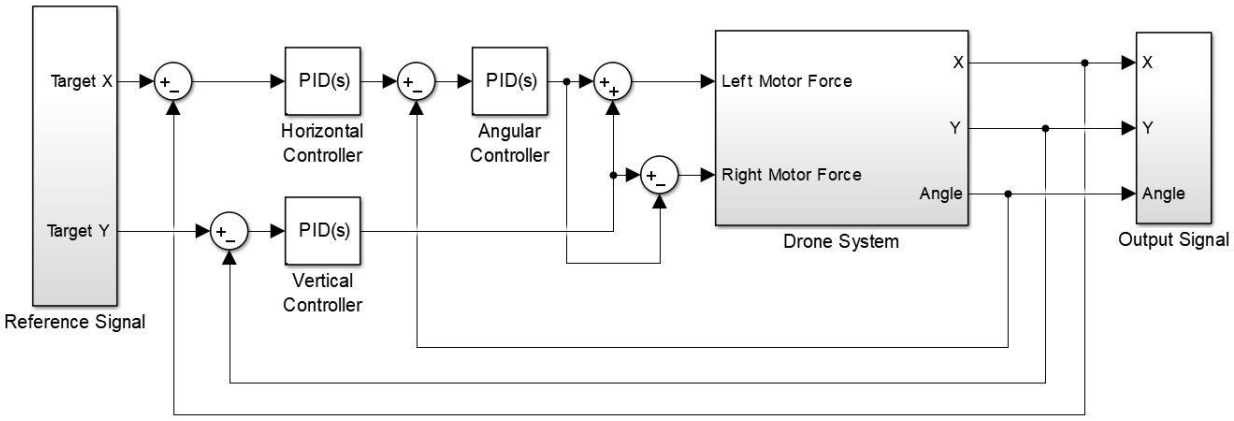
Za regulaciju je korišten PID regulator, označen plavim poljem na slici 2.2, koji se sastoji od 3 dijela. Proporcionalni dio je pojačanje razlike referentnog i izlaznog stanja, odnosno pogreške, koeficijentom K_p te je glavna sastavnica ovog regulatora. Derivativni dio je pojačanje derivacije pogreške koeficijentom K_d te je potreban za prigušenje oscilacija izlaznog signala. Integralni dio je pojačanje integrala pogreške koeficijentom K_i te je potreban za nadoknađivanje sile zbog vanjskih utjecaja. U slučaju letjelice to su gravitacija i vjetar. Svim dijelovima regulatora je ulaz ista pogreška, a svi njihovi izlazi se zbrajaju stvarajući upravljački signal za letjelicu.



Slika 2.2 Dijagram PID regulatora

Za upravljanje letjelicom je potrebno 3 regulatora, onoliko koliko ima i izlaza. Iz prijašnjih jednadžbi gibanja sustava, može se primijetiti kako razlika motornih sila utječe na kut, a njihova suma na x i y poziciju. Zbog utjecaja gravitacije i prioriteta održanja y pozicije, mora se ograničiti maksimalni i minimalni kut koji može doći na ulaz kutne regulacije.

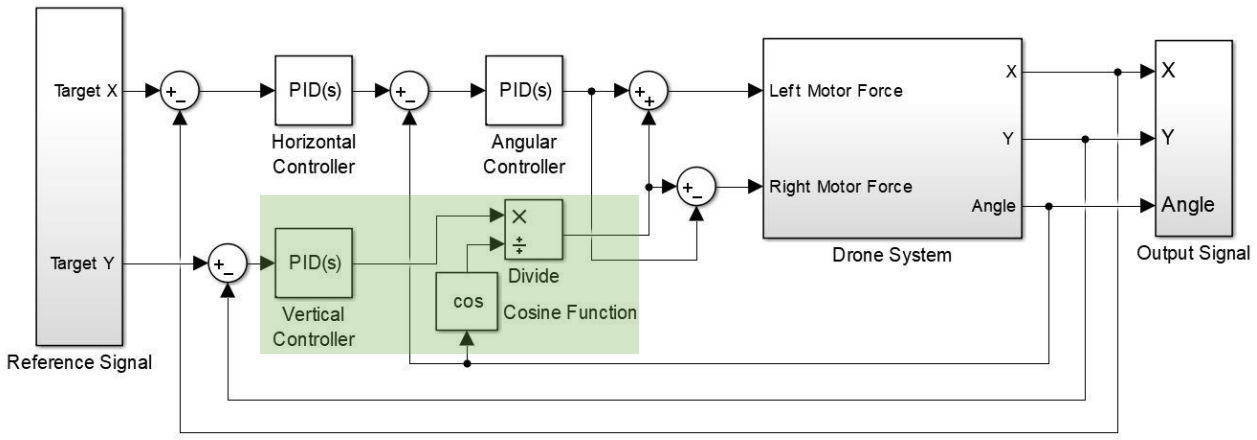
Najjednostavnija regulacija u razmatranom sustavu prikazanom na slici 2.3 je regulacija vertikalne pozicije y. Ulaz u regulator vertikalne pozicije je pogreška vertikalne pozicije y, a njegov izlaz se dodaje u jednakom iznosu upravljačkim signalima oba motora. Pošto se promjena horizontalne pozicije ostvaruje nagibom letjelice, horizontalna regulacija je ostvarena kaskadnom regulacijom s vanjskom regulacijskom petljom koja regulira horizontalnu poziciju i unutrašnjom petljom koja regulira kut. Regulator horizontalne pozicije x izračunava referentni signal kuta. Regulator kuta izračunava kutnu pogrešku i na temelju nje, upravljački signal stvara moment sile na letjelicu tako da se isti iznos dodaje lijevom motoru, a desnom oduzima.



Slika 2.3 Dijargram regulacije letjelice

Pošto je letjelica definirana kao točkasto kruto tijelo, vanjske sile uvijek utječu na središte letjelice i ne stvaraju dodatni moment na letjelicu. Pošto kut letjelice samo služi za promjenu horizontalne pozicije, dok njegova točnost sama po sebi nije važna, kutnoj regulaciji nije potreban integralni dio.

Na slici 2.4 se nalazi druga varijanta regulatora koja bi trebala osigurati istu visinu promjenom kuta letjelice. U zelenom označenom polju dijagrama slike 2.4, izlaz vertikalnog regulatora se naknadno dijeli s kosinusom kuta nagiba kako bi se nadoknadio gubitak vertikalnog potiska pod većim nagibom letjelice.



Slika 2.4 Dijargram regulacije letjelice s nadoknađivanjem vertikalnog potiska

3. IMPLEMENTACIJA

U ovom poglavlju objašnjena je praktična implementacija metodologije objašnjene u drugom poglavlju. Opisano je kako izgledaju signali, određeni sustavi i pojedini regulatori u C# programskom jeziku za Unity engine. Najprije se upoznajemo s Unity alatom i svim njegovim funkcijama koje se koriste u nastavku.

3.1. Unity

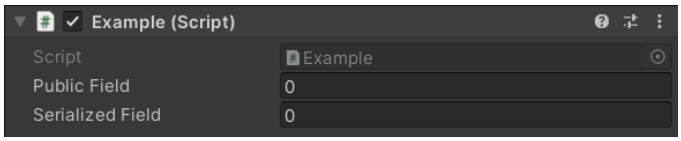
Unity engine je razvojni alat za video igre namijenjen za više platformi. Napisan je u C++ programskom jeziku, a C# se koristi za skriptiranje pomoću Unity APIa. Unity je u mogućnosti praviti trodimenzionalne i dvodimenzionalne igre te interaktivne simulacije.

Monobehaviour je osnovna Unity klasa nužna za realizaciju korisničkog kôda u sceni simulacije. Sadrži predefinirane funkcije u kojima korisnik može napisati željeno ponašanje [5]. *Start* funkcija se pokreće početkom korištenja simulacije. *Update* funkcija se pokreće za svaku sličicu koju stvara Unity. Za tu funkciju se često koristi varijabla *deltaTime* klase *Time*, koja opisuje period *Update* izvedbe prilikom simulacije. Isto tako postoji funkcija *FixedUpdate* koja je varijanta *Update* funkcije s konstantnim periodom izvedbe tijekom simulacije, a ime varijable za period izvedbe je *fixedDeltaTime* unutar klase *Time*. U kodu 3.1 je prikazan primjer korisničke klase *Example* koja nasljeđuje *MonoBehaviour* Unity klasu namijenjenu za realizaciju u Unity sučelju.

```
using UnityEngine;

public class Example : MonoBehaviour
{
    private float privateField;
    public float publicField;
    [SerializeField]
    private float serializedField;

    void Start()
    {
        // Start is called before the first frame update
    }
    void Update()
    {
        // Update is called once per frame
    }

    void FixedUpdate()
    {
        // FixedUpdate is called once per frame of fixed period
    }
}
```

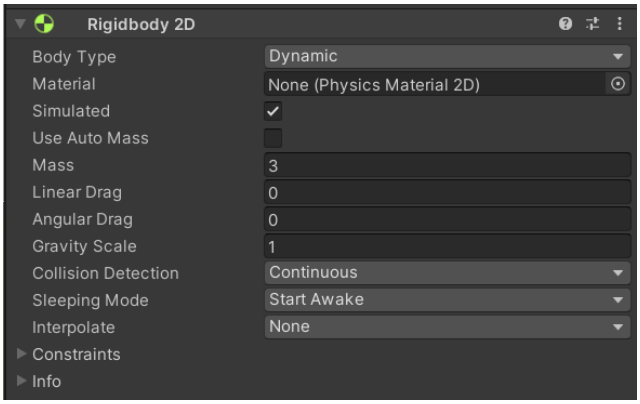
Kod 3.1 Primjer korisničke klase za Unity sučelje

Važno je napomenuti kako su javne varijable i privatne varijable sa *SerializeField* atributom vidljive u Unity sučelju kao na slici 3.1 gdje je prikazana realizacija primjera korisničke klase.



Slika 3.1 Primjer realizacije korisničke klase *Example*

Najvažnija komponenta nužna za simulaciju krutog tijela u Unity-u jest *Rigidbody* komponenta. Također postoji i *Rigidbody2D* komponenta koja je funkcionalno ista kao i *Rigidbody*, samo što je prilagođena za dvodimenzionalni prostor [5]. Na slici 3.2 se nalaze parametri *Rigidbody2D* komponente. Najvažniji parametri su *mass, linear drag, angular drag* i *gravity scale*. Sadrži varijable vektora pozicije (*position*), brzine (*velocity*), kuta (*angle*) i kutne brzine (*angularVelocity*).



Slika 3.2 *Rigidbody2D* komponenta u Unity sučelju

Vector2D je struktura podataka motora koja se sastoji od dvije vrijednosti s pomičnom decimalnom točkom. Također sadrži statičnu funkciju *SignedAngle* koja računa kut u stupnjevima s predznakom između dva 2D vektora smjera [5].

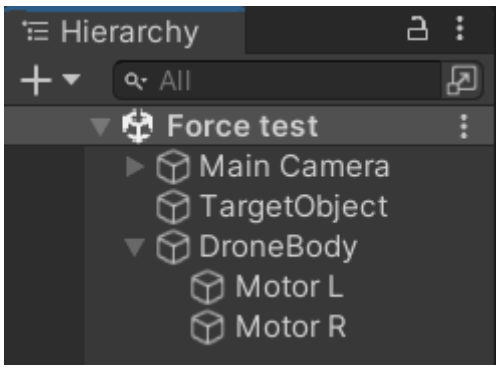
Komponenta *Transform* je klasa koja svaki *MonoBehaviour* objekt u Unity sceni sadrži u svojoj kompoziciji. Iz te komponente se može izvući stanje o poziciji varijablom *position* te stanje o rotaciji u obliku vektora *eulerAngles* [5]. Zbog dvodimenzionalnosti simulacije, koristi se samo *z* varijabla vektora *eulerAngles*.

Mathf je struktura podataka koja sadrži kolekciju matematičkih funkcija. Jedna od statičnih funkcija je *Clamp* koja za predanu vrijednost, minimum i maksimum, vraća tu istu vrijednost ograničenu na interval minimuma i maksimuma. Ako je veća od maksimuma, vraća maksimum. Ako je manja od minimuma, vraća minimum. Statična funkcija *DeltaAngle* vraća najkraću razliku predana dva kuta u intervalu od -179 do 180 stupnjeva [5].

AddForce funkcija je dio *Rigidbody2D* klase koja *Rigidbody2D* objektu dodaje silu. Parametar te funkcije je sila *Vector2D* tipa podatka. *AddForceAtPosition* je varijanta funkcije *AddForce* koja uz parametar sile još treba 2D vektor pozicije u prostoru simulacije, odnosno globalnu poziciju. Ova funkcija stvara moment sile na pozvani objekt ako parametar pozicije nije pozicija centra mase objekta [5].

3.2. Implementacija

Letjelica u Unity sceni se sastoji od 3 objekta kao što je prikazano na slici 3.3, glavno tijelo letjelice pod imenom *DroneBody*, lijevi (*Motor L*) i desni (*Motor D*) motor. Glavno tijelo sadrži *Rigidbody2D* komponentu te klasu *Drone* koja predstavlja implementaciju regulacije letjelice. Glavno tijelo je ujedno i roditelj ostala dva objekta. Lijevi i desni motor su u lokalnom prostoru odmaknuti 1 metar od centra roditeljskog objekta.



Slika 3.3 Lista objekata i hijerarhije Unity scene

Oba motora u sceni su implementirana korisničkom klasom *Motor*, koja je izvedenica *MonoBehaviour* klase te koja konstanto potiskuje *DroneBody* varijabilnom silom (*thrust*) u funkciji *FixedUpdate* kao u kodu 3.2. *Motor* iz dohvaćene reference na roditeljsko kruto tijelo (*rb*) pretvara kut letjelice u vektor smjera. Zatim se taj vektor smjera skaliran potisnom silom motora predaje *AddForceAtPosition* funkciji te se navodi pozicija objekta o kojoj ovisi moment sile.

```
    private void FixedUpdate()
    {
        Vector2 direction = new Vector2(
            x: Mathf.Cos((rb.rotation + 90) * Mathf.Deg2Rad),
            y: Mathf.Sin((rb.rotation + 90) * Mathf.Deg2Rad)
            );

        rb.AddForceAtPosition(direction * thrust, (Vector2)transform.position);
    }
```

Kod 3.2 *FixedUpdate* funckija *Motor* klase

Ponašanje PID regulatora je implementirano u korisničkoj klasi *PIDControler* koja ne nasljeđuje niti jednu klasu. Predviđeno je da *MonoBehaviour* objekt letjelice poziva njezinu funkciju *Update*, koja ovog puta nije predefinirana Unity funkcija. *Update* funkcija u ovom slučaju računa i vraća upravljački signal svaki put kada je pozvana. Kako bi funkcija izračunala izlazni signal,

potreban joj je period izvedbe simulacije (*dt*), signal pogreške (*error*), njezine brzine (*velocity*) i, pošto se radi o diskretnom sustavu s određenim ograničenjima, odnosno motori letjelice su postavljeni u jednom smjeru, mogu se postaviti parametri minimuma (*outputMin*) i maksimuma (*outpuMax*) izlaza u svrhu prevencije zasićenja integratora.

U kodu 3.3 je opisan izračun upravljačkog signala. U funkciji se najprije provjerava je li period izvedbe (*dt*) manji ili jednak nuli te ako je, onda se prekida program s porukom kako je greška u parametru perioda izvedbe. Inače, računa proporcionalni dio regulatora (*P*) pojačavajući pogrešku (*error*) varijablom koja predstavlja proporcionalno pojačanje (*proportionalGain*). Zatim se računa integralni dio funkcijom *PreventIntegralSaturation* koda 3.4, koja svakom svojom izvedbom pridodaje varijabli integrala (*I*) pogrešku (*error*) pojačanu integralnim pojačanjem (*integralGain*) te umnoškom s periodom izvedbe (*dt*). Varijabli integrala se ništa ne pridodaje ukoliko kontrolni signal prošle izvedbe (*result*) ima tendenciju rasta, a veći je od maksimuma ili ukoliko kontrolni signal (*result*) ima tendenciju smanjenja, a manji je od minimuma. Derivativni dio (*D*) se računa pojačanjem brzine pogreške (*velocity*) derivativnim pojačanjem (*derivativeGain*). Upravljački signal (*result*) je na kraju suma proporcionalnog, integralnog i derivativnog djela. Signalu je sačuvana vrijednost za buduću izvedbu te je poslan kao izlazna varijabla funkcije *Update*.

```
public float Update(float dt, float error, float velocity, float outputMin =
float.NegativeInfinity, float outputMax = float.PositiveInfinity)
    {
        if (dt <= 0)
            throw new System.ArgumentOutOfRangeException(nameof(dt));

        float P = proportionalGain * error;
        PreventIntegralSaturation(dt, error, outputMin, outputMax);
        float D = derivativeGain * velocity;
        result = P + I + D;
        return result;
    }
```

Kod 3.3 Implementacija PID regulatora u klasi *PIDControler*

```
private void PreventIntegralSaturation(float dt, float error, float outputMin, float
outputMax)
    {
        float S = 1;
        if ((result > outputMax && error > 0) || (result < outputMin && error < 0))
            S = 0;
        I += S * error * integralGain * dt;
    }
```

Kod 3.4 Implementacija prevencije integralne saturacije u klasi *PIDControler*

Nakon implementacije klase PID regulatora, opisana je implementacija pojedinih izlaza letjelice u klasi *Drone*. U kodu 3.5 je prikaz dijela koda klase Drone, koja nasljeđuje *MonoBehaviour*, u kojoj su izlistana sva potrebna polja koja se naknadno postavljaju u Unity sučelju. Polje *target* je potrebno za dobivanje referentnog signala pozicije iz njegove klase *Transform*. Lijevi i desni motori su definirani ovdje kako bi se poslije u kodu mogla svakom motoru postaviti zasebna potisna sila. Referenca na komponentu krutog tijela (*Rigidbody2D*) se odnosi na kruto tijelo letjelice. Slijede 3 reference *PIDControler* klase za svaku pojedinu regulaciju potrebnu za horizontalnu i vertikalnu poziciju te kut. Varijabla *angleThreshold* predstavlja prag vrijednosti kuta potrebne za horizontalnu i kutnu regulaciju. Uobičajena vrijednost joj je 15, ali se ona može naknadno podesiti u Unity sučelju. Varijable *thrustLeft* i *thrustRight* predstavljaju potisak lijevog i desnog motora, a izračunavaju se iz izlaza regulatora. Varijabla *targetAngle* je izlaz horizontalnog regulatora, a *thrustX* i *thrustY* su izlazi kutnog i vertikalnog regulatora.

```
    [SerializeField]
    private Transform target;

    [SerializeField]
    private AreaEffector2D forceField;

    [SerializeField]
    Motor motorLeft;

    [SerializeField]
    Motor motorRight;

    [SerializeField]
    private Rigidbody2D rb;

    [SerializeField]
    private PIDControler verticalController;

    [SerializeField]
    private PIDControler horizontalController;

    [SerializeField]
    private PIDControler angularController;

    [SerializeField]
    private float angleThreshold = 15;

    [SerializeField]
    private float thrustLeft = 0, thrustRight = 0, targetAngle = 0, thrustX = 0, thrustY
= 0, errorX = 0, errorY = 0, errorA = 0;
```

Kod 3.5 Polja varijabli i referenci u klasi *Drone*

U kodu 3.6 je implementirana kompletna regulacija letjelice unutar klase *Drone* pod funkcijom *FixedUpdate* tako da se regulacija letjelice izvršava konzistentno s periodom zadanim varijablom *fixedDeltaTime*. Prvo se poziva *Update* funkcija PID regulatora odgovornog za vertikalnu regulaciju. Predaju joj se parametri perioda izvedbe, pogreška (razlika referentne y pozicije i y pozicije letjelice), negativna brzina letjelice po y osi, te ekstremi, minimalni i maksimalni mogući potisak. Ekstremi potiska su sume jednog ekstremnog mogućeg potiska svakog motora. Regulator sprema rezultat poziva funkcije u *thrustY* varijablu. Slijedi horizontalni regulator koji pretvara horizontalnu pogrešku u referenti signal kuta *targetAngle*. Referentni signal kuta se zatim ograničava *Clamp* funkcijom *Mathf* klase između negativnog i pozitivnog praga kuta. Referentni signal kuta se prosljeđuje kutnom regulatoru koji računa kutnu pogrešku *DeltaAngle* funkcijom *Mathf* klase. Vertikalni potisak (*thrustY*) se naknadno dijeli s kosinusom kuta nagiba kako bi se poništio gubitak vertikalnog potiska pod većim nagibom. Vertikalni (*thrustY*) i horizontalni (*thrustX*) potisci se pretvaraju u potiske za lijevi i desni motor. Zatim se ti potisci šalju motorima njihovom funkcijom *SetThrust* kako bi se potisak realizirao.

```
        errorY = target.transform.position.y - transform.position.y;

        thrustY = verticalController.Update(
            dt: Time.fixedDeltaTime,
            error: errorY,
            velocity: -rb.velocity.y,
            outputMin: 0,
            outputMax: 200
            );

        errorX = target.transform.position.x - transform.position.x;
        targetAngle = horizontalController.Update(
            dt: Time.fixedDeltaTime,
            error: errorX,
            velocity: -rb.velocity.x,
            outputMin: -angleThreshold,
            outputMax: angleThreshold
            );

        targetAngle = Mathf.Clamp(targetAngle, -angleThreshold, angleThreshold);

        errorA = Mathf.DeltaAngle(-targetAngle, transform.eulerAngles.z);
        thrustX = angularController.Update(
            dt: Time.fixedDeltaTime,
            error: errorA,
            velocity: rb.angularVelocity,
            outputMin: -100,
            outputMax: 100
            );

        thrustY = thrustY / Mathf.Cos(rb.rotation * Mathf.Deg2Rad);

        //To go rightwards, left side motor must be stronger than right side
        thrustLeft = thrustY + thrustX;
        thrustRight = thrustY - thrustX;
        motorLeft.SetThrust(thrustLeft);
        motorRight.SetThrust(thrustRight);
```

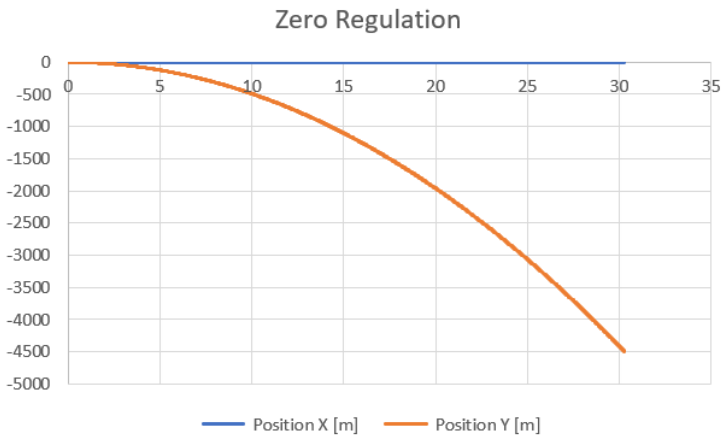
Kod 3.6 Implementacija regulacije letjelice unutar *FixedUpdate* funkcije *Drone* klase

4. REZULTATI

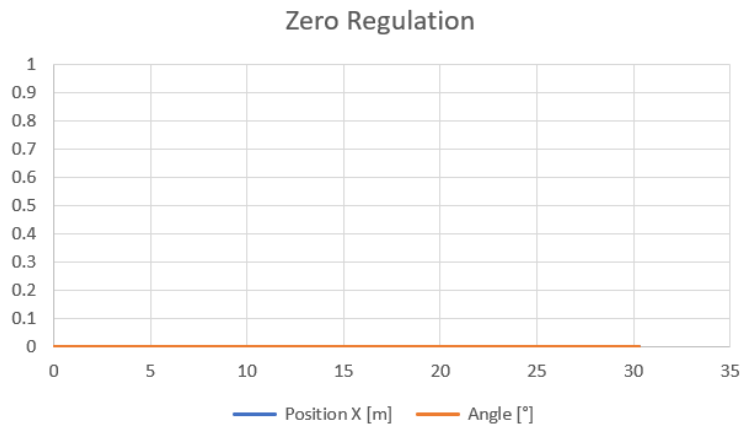
U ovom poglavlju prikazani su rezultati izvedbe simulacija nekoliko slučaja. Najprije je promatrana letjelica bez ikakve regulacije i ulaza. Zatim je promatrana simulacija letjelice s vertikalnom regulacijom, pa onda test horizontalne regulacije, jer vertikalna regulacija je uvijek nužna zbog prisustva gravitacije. Nakon toga, slijedi opis simulacija s kombiniranom regulacijom, a na kraju se razmatra slučaj s kompletnom regulacijom s prisustvom vjetra u smjeru referentnog signala te suprotnom smjeru.

U svim slučajevima simulacije, početno stanje pozicije je uvijek nul vektor, isto tako i vektor brzine letjelice te kut i kutna brzina su nula, a u komponenti krutog tijela, masa letjelice je 3 kilograma, translacijski i kutni koeficijent trenja je 0, a gravitacijska skala je 1, što odgovara Zemljinoj gravitaciji. Maksimalni mogući potisak svakog motora je neograničen, a minimalan iznosi 0.

Referentni signal pozicije je nebitan jer su parametri regulacije postavljeni na nulu u svrhu promatranja ponašanje letjelice bez ikakvih ulaza. Nakon izvedbe simulacije u trajanju od 30 sekundi, dobivamo rezultat horizontalne pozicije (plava krivulja) i vertikalne pozicije (narančasto) u metrima, kao na slici 4.1, a na slici 4.2 rezultat horizontalne pozicije (plavo) i kuta u stupnjevima (narančasto). Vertikalna pozicija se na ubrzan način smanjuje, odlazi prema negativnoj beskonačnosti te time zaključujemo da je sustav uistinu nestabilan. U simulaciji je zanemaren otpor zraka. Između 25. i 30. sekunde, letjelica prijeđe put od 1500 metara, što znači da se letjelica gibala brzinom od 300 m/s. Takav slučaj je u stvarnosti nemoguće ostvariti slobodnim padom, zbog prisutnosti otpora zraka. Kut i horizontalna pozicija ostaju netaknuti.

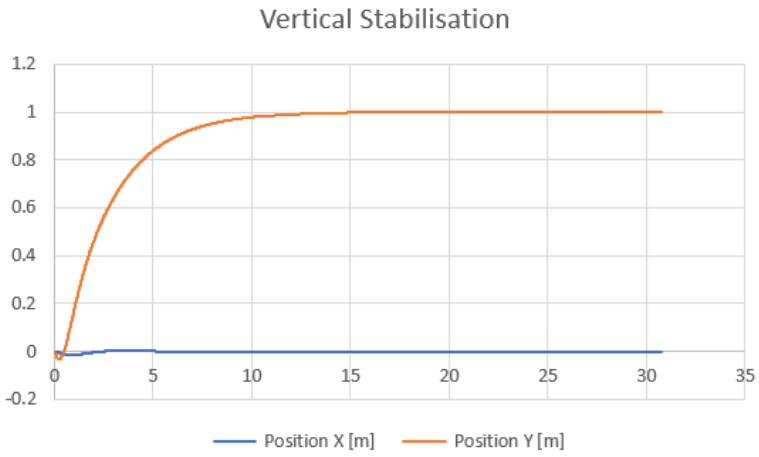


Slika 4.1 Pozicija letjelice bez regulacije



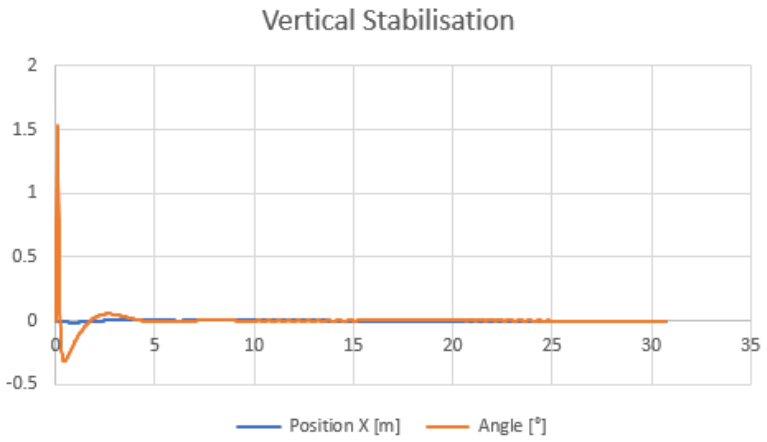
Slika 4.2 Horizontalna pozicija i kut letjelice bez regulacije

Nakon uključivanja cjelokupne regulacije i testiranja uz različite vrijednosti parametara najbolji rezultat je postignut za sljedeće parametre vertikalnog regulatora: 15 za proporcionalno, 7 za derivativno i 5 za integralno pojačanje; sljedeće parametre horizontalnog regulatora: 10 za proporcionalno, 8 za derivativno i 1 za integralno pojačanje te za sljedeće parametre kutnog regulatora: 1 za proporcionalno, 0.1 za derivativno i 0 za integralno pojačanje. Prvo su prikazani rezultati regulacije vertikalne pozicije. Pokretanjem simulacije za referentni signal se daje podatak u obliku 2D vektora $[x \quad y]^T$, gdje x predstavlja horizontalnu referentnu poziciju, a y vertikalnu. Za referentni signal vektora $[0 \quad 1]^T$ dobivamo rezultate na slikama 4.3 i 4.4. U svim izlaznim karakteristikama sustav letjelice dolazi u stabilno stanje. Na slici 4.3 je vidljivo kako letjelica uspješno savladava gravitaciju. Ona brzo prilazi referentnoj visini do pete sekunde, a nakon toga se polako približava stacionarnoj vrijednosti.



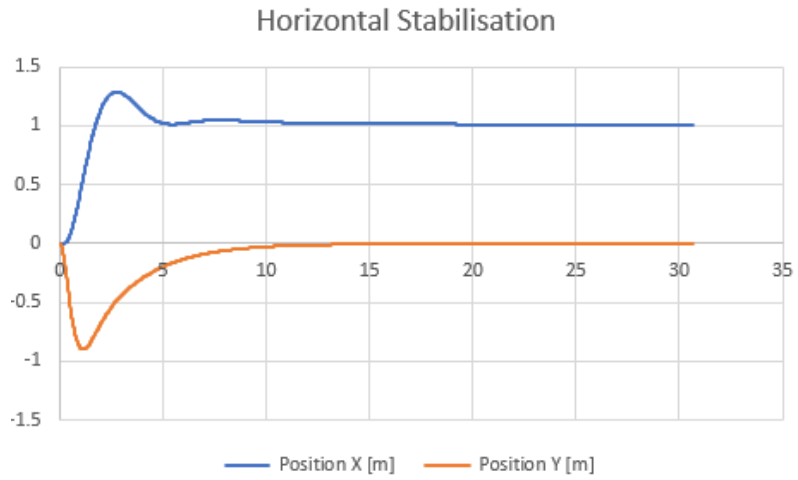
Slika 4.3 Pozicija letjelice u testu verikalne regulacije

Na narančastom grafu na slici 4.4 su vidljive oscilacije u prvoj sekundi. U petoj sekundi kut polako dolazi na referentnu vrijednost. U desetoj sekundi grafa kuta se iznenadno pojavljuje oscilacija, ali vrlo male amplitude te joj se amplituda smanjuje i nestaje u petnaestoj sekundi. Slične oscilacije se također pojavljuju od dvadesete do dvadeset i pete sekunde.



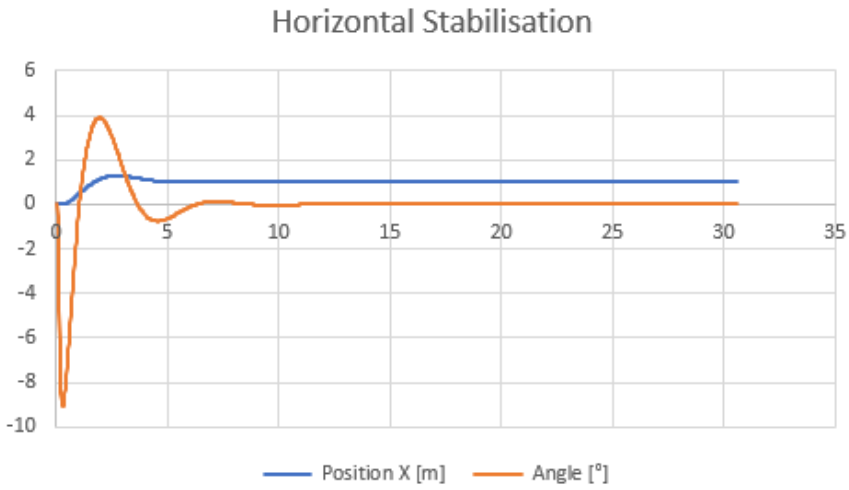
Slika 4.4 Horizontalna pozicija i kut letjelice u testu verikalne regulacije

U sljedećem slučaju, se pokreće simulacija s referentnim signalom vektora $[1 \quad 0]^T$. Iako se prvotno prati samo regulacija horizontalne pozicije, važno je pratiti sve izlazne signale letjelice. Na slici 4.5 graf horizontalne pozicije (plavo) u prvoj sekundi, letjelica gotovo miruje. Zatim naglo ubrzava u pozitivnom smjeru, premašujući referentnu poziciju. Nakon pete sekunde, s blagom oscilacijom dolazi u referentno stanje u desetoj sekundi. Na slici 4.5 narančasti graf vertikalne pozicije naglo pada u prvoj sekundi, a zatim djelovanjem integratora vertikalnog regulatora, letjelica nadvladava gravitacijsku silu te polako dostiže svoju referentnu, ujedno i početnu, poziciju u desetoj sekundi.



Slika 4.5 Pozicija letjelice u testu horizontalne regulacije

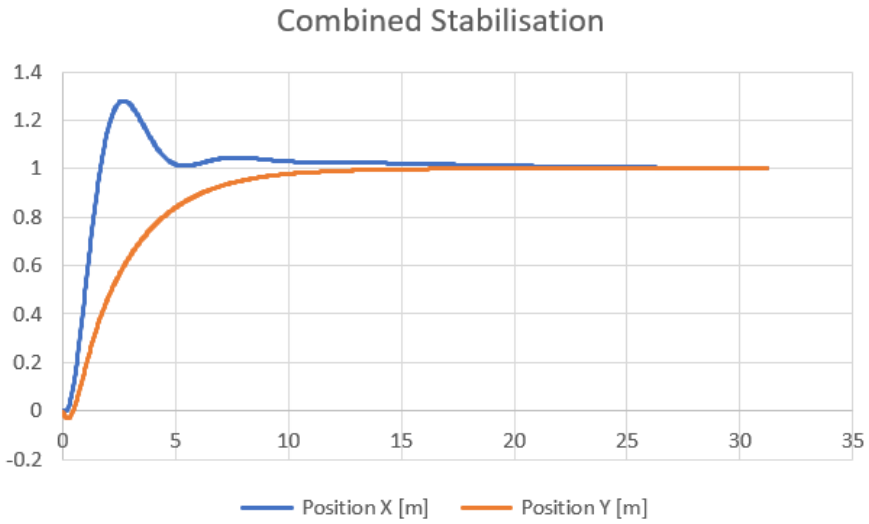
Slijede rezultati regulacije horizontalne pozicije. Na slici 4.6 se promatraju samo izlazi bitni za horizontalnu regulaciju. Narančasti graf kuta letjelice značajnije pokazuje ponašanje te se time može jasnije obrazložiti ponašanje izlazne vrijednosti horizontalne pozicije. U prvom djeliću sekunde gdje se čini da je horizontalna pozicija u zastoju, kut naglo pada te u prvoj sekundi naglo raste što objašnjava i rast horizontalne pozicije. Kut uglavnom oscilira oko vrijednosti nule, što je vrijednost koja ne daje horizontalan potisak. Značajne oscilacije traju do pete sekunde, a gotovo nestaju tek u desetoj sekundi.



Slika 4.6 Horizontalna pozicija i kut letjelice u testu horizontalne regulacije

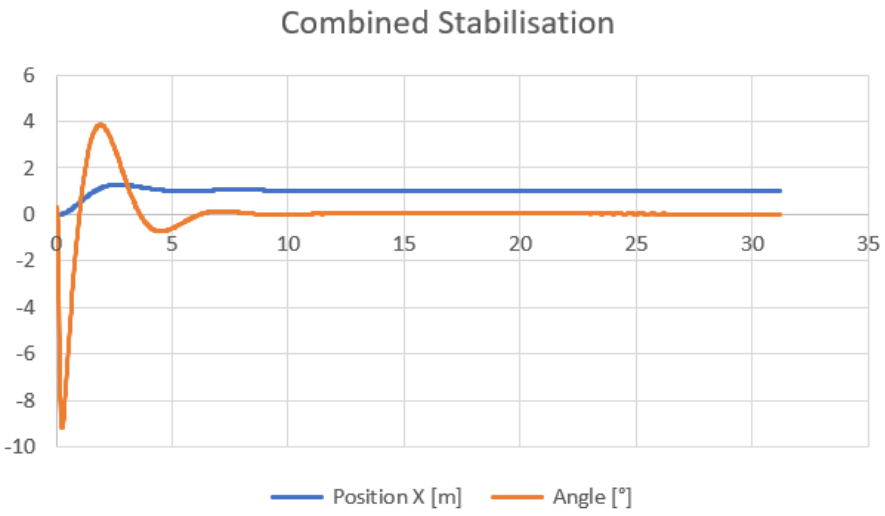
Nakon što su analizirani rezultati regulacije vertikalne i horizontalne pozicije, slijedi prikaz rezultata kombinirane regulacije, tj. istovremene regulacije vertikalne i horizontalne pozicije. Za bolju

analizu međusobnog utjecaja horizontalne i vertikalne regulacije, pokreće se simulacija s referentnim signalom vektora $[1 \quad 1]^T$. Rezultati su slični rezultatima prethodnih testova. Na slici 4.7, plavi graf horizontalne pozicije je identičan grafu iste pozicije na prethodnoj slici 4.5 testa horizontalne regulacije, a narančasti graf vertikalne pozicije identičan istom grafu na slici 4.3 testa vertikalne regulacije.



Slika 4.7 Pozicija letjelice u testu kombinirane regulacije

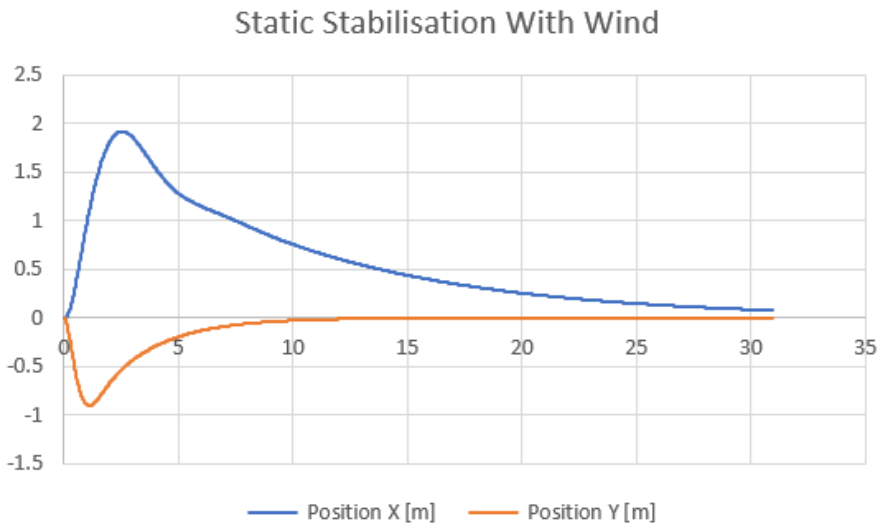
Na slici 4.8 se opet promatraju učinci kombinirane regulacije, no grafovi su identični grafovima testa horizontalne regulacije na slici 4.6. Jedina razlika između grafova jest da na slici 4.8 narančasti graf kuta ima minimalne oscilacije oko dvadeset i pete sekunde. Te oscilacije su identične istog grafa na slici 4.4 u testu vertikalne regulacije.



Slika 4.8 Horizontalna pozicija i kut letjelice u testu kombinirane regulacije

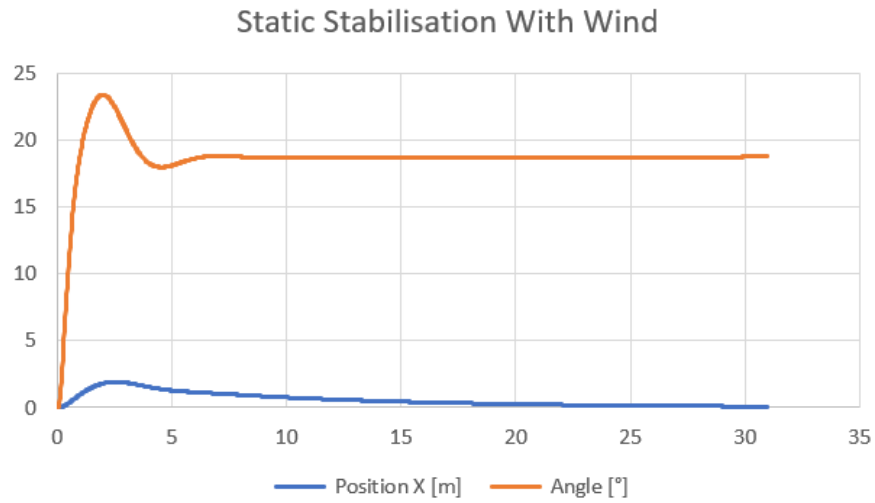
Na temelju do sada opisanih simulacija, može se zaključiti kako horizontalna i vertikalna regulacija ne utječu jedno na drugo u značajnoj mjeri. U sljedećim testovima u simulaciju se uključuje vjetar koji djeluje istom silom na sve objekte unutar simulacije. Sila vjetra je u svim testovima iznosa 10 N, a smjer vjetra ovisi o vrsti testa.

U simulaciju se uključuje vjetar s vektorom smjera $[1 \quad 0]^T$ i zadaje se letjelici referentni signal $[0 \quad 0]^T$ tako da letjelica zadrži svoju početnu poziciju kompenzirajući djelovanje vjetra. Na slici 4.9 je prikazano ponašanje vertikalne pozicije (narančasto) koje je slično onome prikazanom na slici 4.5 u testu horizontalne regulacije, jer je u vertikalnoj osi simulacije prisutna samo gravitacija. S druge strane, ponašanje horizontalne pozicije (plavo) ima zanimljivije ponašanje koje je nalik ponašanju vertikalne pozicije, ali sa suprotnim predznakom. Karakterizira ga nagli rast do 2.5 sekunde te nakon toga vraćanje prema referentom signalu. Graf se naizgled „lomi“ u petoj sekundi te se polako i usporeno vraća početnoj poziciji.



Slika 4.9 Pozicija letjelice u testu kombinirane regulacije s vjetrom

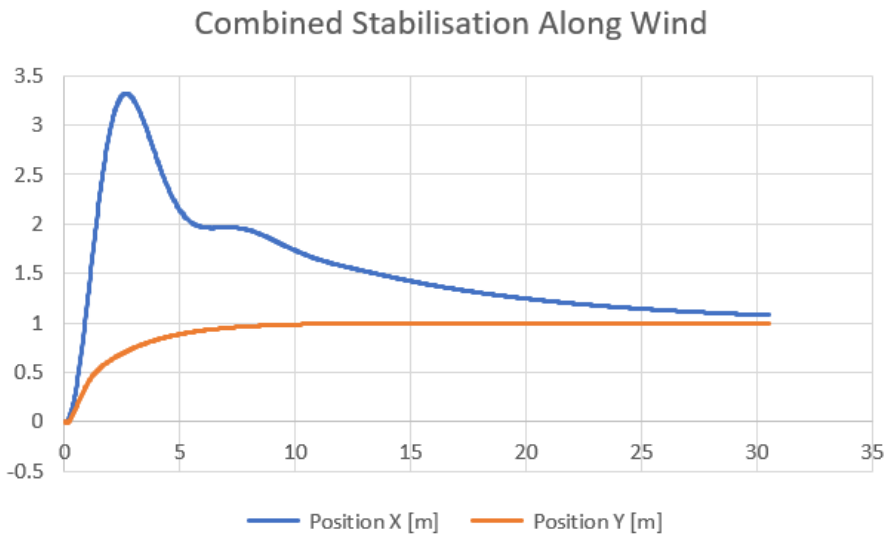
Uspoređujući grafove na slici 4.10, graf kuta (narančasto) svojim lokalnim minimumom u petoj sekundi uzrokuje lom na grafu horizontalne pozicije na slici 4.9. U prijašnjim testovima, vrijednost kuta je oscilirala oko vrijednosti 0°. U ovom testu, vrijednost kuta oscilira ipak oko 18° zbog prisutnosti vjetra, kako bi se poništila vanjska sila i dovela letjelicu u referentno stanje.



Slika 4.10 Horizontalna pozicija i kut letjelice u testu kombinirane regulacije s vjetrom

Sljedeći simulacijski test zadaje letjelici referentni signal $[1 \quad 1]^T$, a smjer vjetra je jedinični vektor $\left[\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right]^T$. Na slici 4.11 graf vertikalne pozicije prije dolazi u referentno stanje nego u svim prijašnjim testovima, jer sila vjetra poništava dio gravitacijske sile. Graf horizontalne pozicije ima

puno veća odstupanja od prijašnjih testova, zbog toga što u početnom stanju letjelica ubrzava prema referentnoj vrijednosti, pa uz pomoć vanjske sile, premašuje referentnu vrijednost. Na plavom grafu na slici 4.11 se pojavljuje lomljenje oko pete sekunde, slično lomljenju istog grafa na slici 4.9 testa kombinirane regulacije s vjetrom.



Slika 4.11 Pozicija letjelice u testu kombinirane regulacije u smjeru s vjetrom

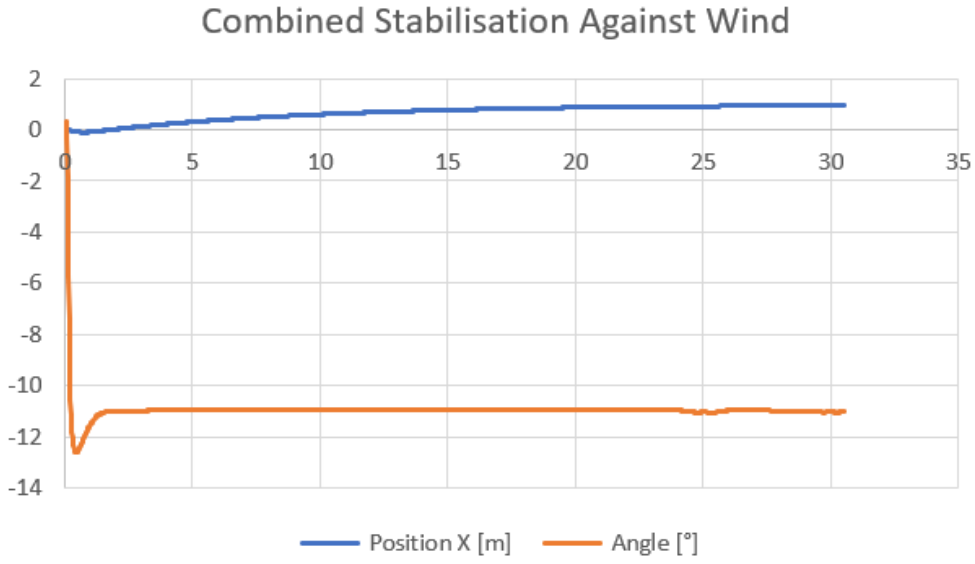
Grafovi na slici 4.12 su slični grafovima slike 4.10 testa kombinirane regulacije s vjetrom, samo što su na slici 4.12 oscilacije vrijednosti kuta puno jače, a kut djelom poprima i negativne vrijednosti. Graf horizontalne pozicije umjesto usporavanja prema nuli, usporava prema jedinici, jer je u ovom testu različita referenta pozicija u odnosu na test prikazan na slici 4.10.

Slika 4.12 Horizontalna pozicija i kut letjelice u testu kombinirane regulacije u smjeru s vjetrom

Na kraju je prikazana simulacija kombinirane regulacije protiv vjetra. Smjer vjetra je suprotan prijašnjem vektoru smjera, odnosno $\left[-\frac{1}{\sqrt{2}} \quad -\frac{1}{\sqrt{2}}\right]^T$, a referentni signal jednak prijašnjem $[1 \quad 1]^T$. Oba grafa pozicije na slici 4.13 idu u negativnom smjeru u prvoj sekundi zbog značajnog djelovanja vanjskih sila, pa nakon toga rastu prema referentom stanju. Očita razlika u grafovima jest da graf horizontalne pozicije puno sporije prilazi referentnoj poziciji od grafa vertikalne pozicije, te se na grafu horizontalne pozicije od dvadeset i pete sekunde pa do kraja simulacije pojavljuju male oscilacije.

Slika 4.13 Pozicija letjelice u testu kombinirane regulacije protiv vjetra

Na slici 4.14 je zanimljivo promatrati narančasti graf kuta letjelice. Kut se vrlo brzo smanjuje, strmo raste u negativnom smjeru, te se ustaljuje oko vrijednosti -11° i to sve u roku dvije sekunde. Graf ovog testa ima najmanje vrijeme dolaska do ustaljene vrijednosti od svih ostalih testova.



Slika 4.14 Horizontalna pozicija i kut letjelice u testu kombinirane regulacije protiv vjetra

Analizom provedenih simulacija, može se zaključiti kako regulatori letjelice nisu idealno postavljeni. U većini grafova vertikalne pozicije može se primijetiti kako se vrijednost uvijek

usporeno približava referentnoj vrijednosti, pogotovo u testu horizontalne regulacije. Kutni regulator izvršava zadovoljavajuću regulaciju.

5. ZAKLJUČAK

Simulacija letjelice može poslužiti u svrhu predviđanja ponašanja letjelice korištenjem određenih parametara regulatora, te u svrhu učenja o automatskom upravljanju i PID regulaciji fizičke letjelice ili za simulaciju letjelice u video igrama.

Rezultati simulacije su zadovoljavajući, jer uporabom regulacije na nestabilnom sustavu dobivamo stabilan sustav koji prati referentno stanje te ostaje u njemu. Uvođenjem dodatnih vanjskih sila u simulaciju otežava dolazak u referentno stanje, no uz pomoć integratora unutar PID regulatora, sustav uspijeva kompenzirati djelovanje poremećaja te opet uspješno dolazi u referentno stanje. Sustav bez regulacije je očekivano nestabilan, no ono što nije očekivano su iznenadne oscilacije tijekom kutne i horizontalne regulacije. Iako su oscilacije vrlo male amplitude, one svakako nisu poželjne.

Zbog diskretnosti i ograničenosti sustava, simulaciju bi bilo dobro unaprijediti tako da regulatori imaju mogućnost izbjegavanja iznenadnih i nepoželjnih oscilacija. Isto tako bi bilo dobro proširiti ju u treću dimenziju prostora, uvodeći još jednu dimenziju pozicije te dvije dimenzije rotacije. Također bi bilo dobro proširiti simulaciju uvođenjem prepreka i rješenja za izbjegavanje kolizije.

LITERATURA

[1] Z. Vrhovski, Automatsko upravljanje, Bjelovar, 2013.

[2] Lj. Kuljača, Z. Vukić, Automatsko Upravljanje Sistemima, Zagreb, 1985

[3] Y. Wang, J. R. Chardonnet, F. Merienne, Modeling online adaptive navigation in virtual environments based on PID control, Berlin, 2023.

[4] S. Bouabdallah, R. Siegwart, Full control of a quadrotor, Zurich, 2007.

[5] Unity Technologies, Unity documentation, 2024, dostupno na: https://docs.unity3d.com/ScriptReference [29.6.2024.]

SAŽETAK

Letjelica s dva motora simulirana je u dvodimenzionalnom prostoru. Matematički je opisano gibanje letjelice te su projektirani odgovarajući regulatori horizontalne i vertikalne pozicije te kuta letjelice. Objašnjeno je korištenje Unity engine programa i programskog jezika C# za realizaciju simulacije, implementaciju sustava letjelice i njezinu regulaciju pomoću više PID regulatora. Analizirano je ponašanje letjelice upravljane PID regulatorima uz djelovanje vjetra kao poremećaja.

Ključne riječi: automatsko upravljanje, kvadrokopter, PID, regulacija, Unity, simulacija letjelice

ABSTRACT

Simulation of aircraft moving in two-dimensional space

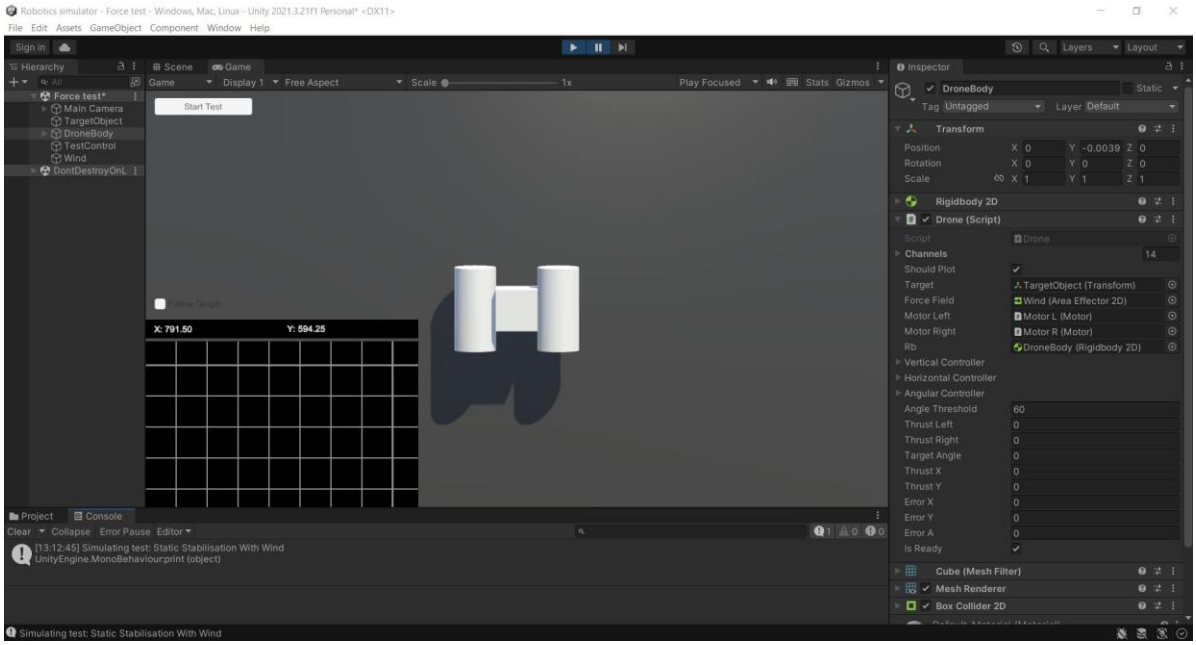
Aircraft with two thrusters is simulated in two-dimensional space. The movement of the aircraft is mathematically described and its corresponding control loops for regulation of the horizontal and vertical position, and the angle of the aircraft were developed. The use of the Unity engine and the C# programming language was explained for the realisation of the simulation, the implementation of the aircraft system and its control with multiple PID controllers. Behaviour of the aircraft controlled by PID controller was analysed in the presence of wind as a disturbance.

Key words: aircraft simulation, automatic control, control loops, drone, PID, Unity

PRILOG

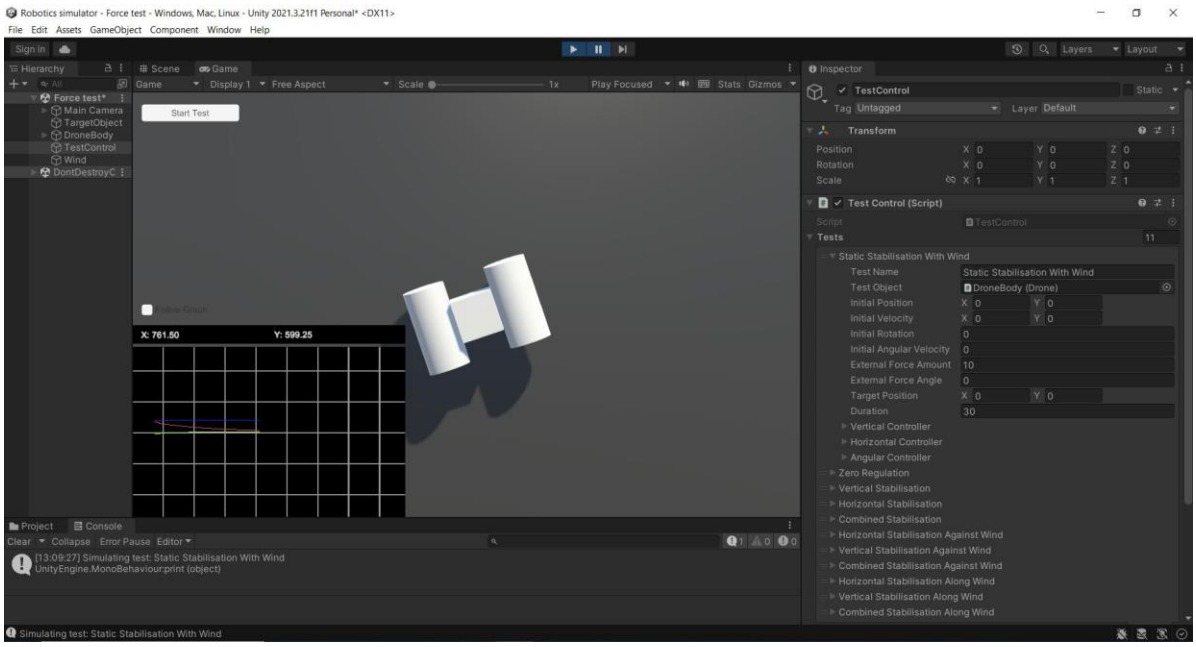
U ovom poglavlju su prikazani neobrađeni rezultati simulacija koje nisu nužne za objašnjenje rada letjelice i njezinog sustava upravljanja.

Na slici P.1 i P.2 je prikazano uobičajeno Unity sučelje koje se sastoji od: hijerarhije scenskih objekata na lijevoj strani, liste komponenata pod imenom *Inspector* na desnoj strani, konzole i prikaza datoteka unutar projekta na dnu ekrana, te prikaza scene i pogleda igračeve kamere u sredini. Na slici P.1 je odabran kompozitni objekt imenom *DroneBody,* koji predstavlja našu letjelicu, te je moguće vidjeti komponentu *Drone.* Simulacija se nalazi u prvoj iteraciji.



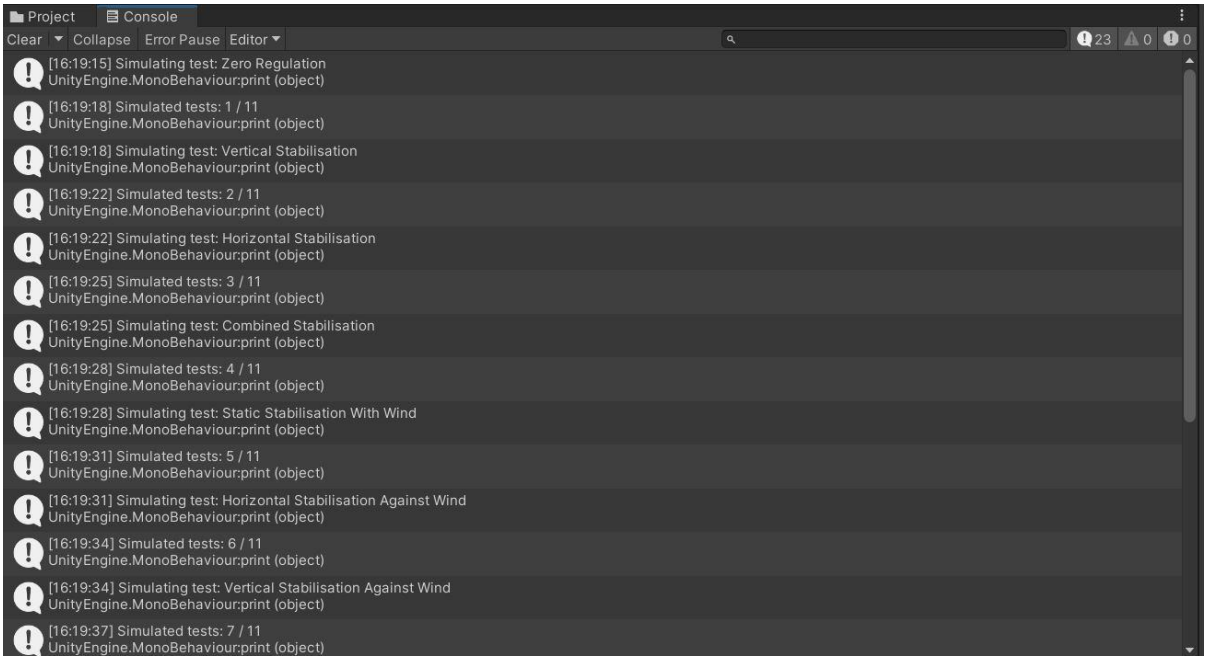
Slika P.1 Unity sučelje u prvoj iteraciji simulacije

Na slici P.2 je odabran scenski objekt imenom *TestControl* te sadrži istoimenu komponentu *TestControl,* koja je zadužena za postavljanje parametara simulacije i drona te snimanje izlaznih signala letjelice. Simulacija je zadržana nakon 4 sekunde izvedbe. U srednjem prozoru, osim što je vidljiv model letjelice, vidljiv je graf koji prikazuje izlazne karakteristike letjelice. Plava krivulja predstavlja kut, crvena horizontalnu, a zelena vertikalnu poziciju letjelice.

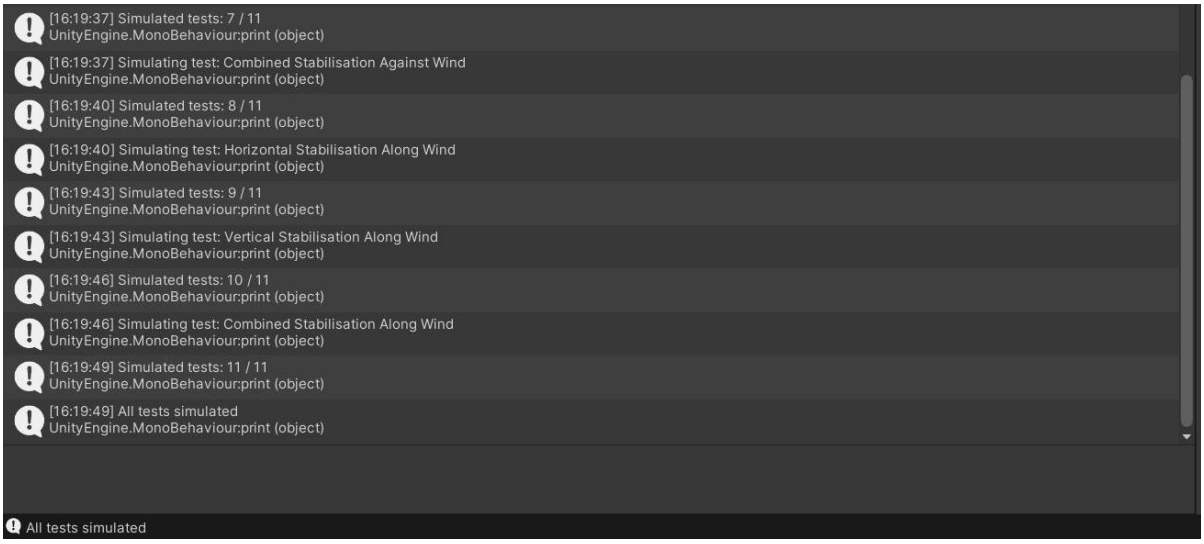


Slika P.2 Unity sučelje nakon nekoliko iteracija simulacije

Na slici P.3 i P.4 su ispisana stanja kroz koja je simulacija prolazila. U uglatim zagradama pojedinog ispisanog stanja se nalazi vrijeme iskazano u satima, zatim u minutama, pa sekundama.



Slika P.3 Prikaz izlaza simulacije u konzoli 1/2



Slika P.4 Prikaz izlaza simulacije u konzoli 2/2

Na slici P.5 se nalazi konačni izlaz simulacije u obliku CSV datoteka (*Comma Separated Values*). Ti se podaci učitavaju u Excel programu te ih se obrađuje.

Name	Date modified	Type	Size
Zero Regulation.csv	23/08/2024 13:06	Microsoft Excel Co...	145 KB
Static Stabilisation With Wind.csv	23/08/2024 13:06	Microsoft Excel Co...	252 KB
Combined Stabilisation Along Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	253 KB
Vertical Stabilisation Along Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	272 KB
Horizontal Stabilisation Along Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	257 KB
Combined Stabilisation Against Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	253 KB
Vertical Stabilisation Against Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	274 KB
Horizontal Stabilisation Against Wind.csv	14/08/2024 16:43	Microsoft Excel Co...	257 KB
Combined Stabilisation.csv	14/08/2024 16:42	Microsoft Excel Co...	263 KB
Horizontal Stabilisation.csv	14/08/2024 16:42	Microsoft Excel Co...	261 KB
Vertical Stabilisation.csv	14/08/2024 16:42	Microsoft Excel Co...	276 KB

Slika P.5 Prikaz izlaza simulacije u obliku CSV datoteka