Haraket denklemleri:

# Rotasyonel hareket denklemleri :



metin, el yazısı, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, yazı tipi, el yazısı, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Hava aracının parametreleri

    m = hava\_araci\_parametreleri['mass']

    Jxx =  hava\_araci\_parametreleri['Jxx'] # x ekseni etrafındaki atalet momenti

    Jyy =  hava\_araci\_parametreleri['Jyy'] # y ekseni etrafındaki atalet momenti

    Jzz =  hava\_araci\_parametreleri['Jzz'] # z ekseni etrafındaki atalet momenti

    Jxz =  hava\_araci\_parametreleri['Jxz'] # x ve z ekseni arasındaki çapraz atalet momenti

    # Γ teriminin hesaplanması

    Gamma = Jxx \* Jzz - Jxz\*\*2

    # Γ katsayılarının hesaplanması

    Gamma1 = Jxz \* (Jxx - Jyy + Jzz) / Gamma

    Gamma2 = (Jzz \* (Jzz - Jyy) + Jxz\*\*2) / Gamma

    Gamma3 = Jzz / Gamma

    Gamma4 = Jxz / Gamma

    Gamma5 = (Jzz - Jxx) / Jyy

    Gamma6 = Jxz / Jyy

    Gamma7 = ((Jxx - Jyy) \* Jxx + Jxz\*\*2) / Gamma

    Gamma8 = Jxx / Gamma

# Rotasyonel hareket denklemleri

    dot\_p = (Gamma1 \* p \* q - Gamma2 \* q \* r) + (Gamma3 \* L + Gamma4 \* N)

    dot\_q = (Gamma5 \* p \* r - Gamma6 \* (p\*\*2 - r\*\*2)) + (M / Jyy)

    dot\_r = (Gamma7 \* p \* q - Gamma1 \* q \* r) + (Gamma4 \* L + Gamma8 \* N)

# Translasyonel hareket denklemleri

metin, yazı tipi, diyagram, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

yazı tipi, diyagram, beyaz, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu metin, yazı tipi, çizgi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

**β=arcsin(v/Va​) α = arctan(w/u)**

* *Flift*​: Kaldırma kuvveti, hava aracının yukarı doğru kaldırılmasını sağlayan aerodinamik kuvvet.
* *Fdrag*​: Sürükleme kuvveti, hava aracının ilerlemesine karşı direnç gösteren aerodinamik kuvvet.
* *m*: Düşey düzlemdeki moment, hava aracının burun yukarı veya burun aşağı hareketini etkileyen moment.
* *fx*​ ve *fz*​: Hava aracının gövde eksenlerindeki kuvvet bileşenleri, burada kaldırma ve sürükleme kuvvetleri, hava aracının eğim açısı (*α*) ile dönüştürülerek gövde eksenlerine göre bileşenlere ayrılır.
* *fy*​: Yan kuvvet, hava aracının yan doğru hareketini etkileyen aerodinamik kuvvet.
* *l*: Roll momenti, hava aracının yan dönüşünü etkileyen moment.
* *n*: Yaw momenti, hava aracının düzlemdeki yatay dönüşünü etkileyen moment.

# Translasyonel hareket denklemleri

    dot\_u = r\* v - q\* w - Fx/m

    dot\_v = p\*w - r\*u + Fy/m

    dot\_w = q\*u - p\*v + Fz/m

# Euler açıları

metin, yazı tipi, ekran görüntüsü, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

    # Euler açılarının türevleri

    dot\_phi = p + q\*np.sin(phi)\*np.tan(theta) + r\*np.cos(phi)\*np.tan(theta)

    dot\_theta = q\*np.cos(phi) - r\*np.sin(phi)

    dot\_psi = q\*np.sin(phi)/np.cos(theta) + r\*np.cos(phi)/np.cos(theta)

 # Konumun türevleri

el yazısı, metin, yazı tipi, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Konumun türevleri

    s\_phi, c\_phi = np.sin(phi), np.cos(phi)

    s\_theta, c\_theta = np.sin(theta), np.cos(theta)

    s\_psi, c\_psi = np.sin(psi), np.cos(psi)

    dot\_x = (c\_theta \* c\_psi) \* u + (s\_phi \* s\_theta \* c\_psi - c\_phi \* s\_psi) \* v + (c\_phi \* s\_theta \* c\_psi + s\_phi \* s\_psi) \* w

    dot\_y = (c\_theta \* s\_psi) \* u + (s\_phi \* s\_theta \* s\_psi + c\_phi \* c\_psi) \* v + (c\_phi \* s\_theta \* s\_psi - s\_phi \* c\_psi) \* w

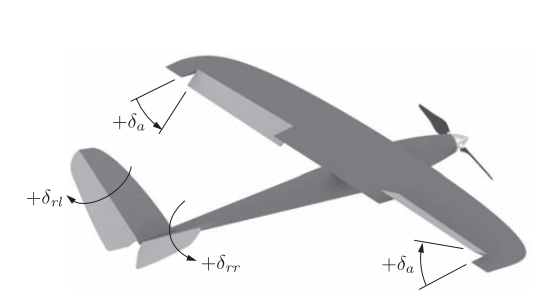
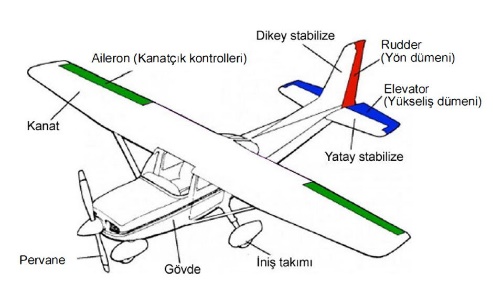
    dot\_z = (-s\_theta) \* u + (s\_phi \* c\_theta) \* v + (c\_phi \* c\_theta) \* w

metin, yazı tipi, el yazısı, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu Aerodinamik kuvvetler Aerodinamik momentler

metin, yazı tipi, el yazısı, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu



yazı tipi, beyaz, diyagram, tipografi içeren bir resim

Açıklama otomatik olarak oluşturulduyazı tipi, diyagram, beyaz, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Elevator ve Aileron Defleksiyonu:** İlk denklemin sol tarafında bir kanat üzerinde pozitif yönde elevator defleksiyonlarını (++*δel*​​ ve ++*δer*​​) gösteren bir grafik bulunuyor. Sağ tarafta ise, bu defleksiyonların nasıl genel elevator defleksiyonu (*δe*​) ve aileron defleksiyonu (*δa*​) değerlerine dönüştürüldüğünü gösteren bir matris dönüşümü yer alıyor. Bu dönüşüm, aşağıdaki şekilde matematiksel olarak ifade edilir:

* *δer*​​ ve *δel*​​ sağ ve sol elevator defleksiyonlarını ifade eder.
* *δrr*​​ ve *δrl*​​ sırasıyla sağ ve sol rudder defleksiyonlarını temsil eder.
* *δe*​ genel elevator defleksiyonunu ifade eder.
* *δr*​, sağ ve sol rudder'ların zıt yönde hareket ettiği durumlarda uçağın yaw hareketini etkileyen bir faktördür.
* *δa*​ ise aileron defleksiyonunu (veya asimetrik elevator defleksiyonunu ki bu da bir tür roll kontrolü sağlar) ifade eder.

yazı tipi, el yazısı, metin, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu

yazı tipi, el yazısı, tipografi, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu

yazı tipi, el yazısı, hat sanatı, kaligrafi, metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Clq ve Cdq 0 dır.

Lateral Aerodynamics

metin, yazı tipi, el yazısı, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

yazı tipi, metin, çizgi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Haraket denklemleri için  Fx,Fy,Fz

    Fx = -g \* np.sin(theta)  + Pq \* S \* (Cx\_a + Cxq\_a \* c \* q/ 2 \* V + Cxδe\_a \* δe) + motor\_itme # Kaldırma kuvveti

    Fy = m \* g \*  np.cos(theta) \* np.sin(phi) + Y + 0   # Sürükleme kuvveti

    Fz = m \* g \*  np.cos(theta) \* np.cos(phi) + Pq \* S \* (  Cz\_a + Czq\_a \* c \* q / 2 \* Czδe\_a \*  δe) + 0 # Yan kuvvet

Cx gibi parametreler sonuç kısmında açıklanmıştır

# Aerodinamik momentler

    l = Pq \* S \* b \* Cl + motor\_tork # Dönme momenti

    m = Pq \* S \* c \* Cm  + 0 # Düşey moment

    n = Pq \* S \* b \* Cn  + 0 # Yanal moment

Pervane torku ve İtişi (Motor Modeli)

İtme:

metin, yazı tipi, el yazısı, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Torku:

yazı tipi, tipografi, hat sanatı, kaligrafi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

δt : Yüzdesel motor gücü ifade eder. Orneğin 0.5 değeri motor %50 hızda çalışıyor anlamına gelir.

Sprop, Cprop ,Ktp ve Komega sabiti kitabın son kısmında verilmiştir.

class MotorModeli:

    def \_\_init\_\_(self,rho,Sprop,Cprop, k\_motor,Ktp,K\_,e):

        self.rho = rho       # Hava yoğunluğu

        self.Sprop = Sprop   # Pervane çapı

        self.Cprop = Cprop   # Pervane katsayısı

        self.Ktp = Ktp       # Tork sabiti

        self.K\_ = K\_         #

        self.k\_motor = k\_motor  # Motor sabiti

        self.e = e          # Motor verimliliği

    def itme\_tork\_hesapla(self, V,δt):

        # İtme (FX\_p) ve tork (Tp) hesaplamaları

        FX\_p = 0.5 \* self.rho \* self.Sprop \* self.Cprop \* ( (self.k\_motor \* δt )\*\*2 - V\*\*2 )

        Tp = -self.Ktp \* (self.K\_ \* δt)\*\*2

        FX\_p \*= self.e

        Tp \*= self.e

        return FX\_p, Tp

Atmosfer Modülü:

Atmosferin irtifaya bağlı olarak değişen sıcaklık, basınç ve yoğunluk

ISA'ya göre deniz seviyesindeki standart koşullar şunlardır:

* Sıcaklık: 15 °C (288.15 K)
* Basınç: 1013.25 hPa (hektopascal)
* Yoğunluk: 1.225 kg/m³

**Sıcaklık (T) Hesaplama Formülü:** T=T0​−L⋅hBurada:

* *T*0​: Deniz seviyesindeki standart sıcaklık (Kelvin cinsinden)
* *L*: Lapse rate (°C/metre veya K/metre cinsinden)
* *h*: İrtifa (metre cinsinden)

**Basınç (P) Hesaplama Formülü:** *​* P=P0​(1− ​L⋅h/ T0​)^g/R⋅Lg​ Burada:

* *P*0​: Deniz seviyesindeki standart basınç
* *g*: Yer çekimi ivmesi (9.80665*m*/*s*2)
* *R*: İdeal gaz sabiti (287.053 J/(*kg*⋅*K*))

**Yoğunluk (*ρ*) Hesaplama Formülü:** ρ=ρ0​(P/Po​)(T/T0​​) Burada:

* 0*ρ*0​: Deniz seviyesindeki standart yoğunluk

T = T0 - L \* hASL

    # Basınç hesabı

    P = P0 \* (1 - L \* hASL / T0) \*\* (g / (R \* L))

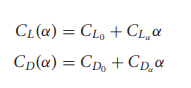
    # Yoğunluk hesabı

    rho = rho0 \* (P / P0) \* (T0 / T)

Sonuç ve özet olarak:

metin, el yazısı, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu Translasyonel hareket denklemlerideki itme sürükleme ve yanal değerlerimizi aşağıdaki formüle göre güncellemeliyiz.

Denklemin ilk kısmı, yerçekimi bileşenini temsil eder ve aracın eğim açısına (θ) bağlıdır. İkinci kısım, aerodinamik kuvvetlerin toplamını temsil eder ve aerodinamik katsayıları, aracın hızını, kanat alanını ve diğer aerodinamik değişkenleri içerir. Üçüncü kısım, pervanenin ürettiği itme kuvvetini temsil eder.

# Rotasyonel hareket denklemleri için:

Denkleme göre pervane torkunu ekleyip düzenlemeliyiz.

el yazısı, metin, yazı tipi, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Trim işlemleri:

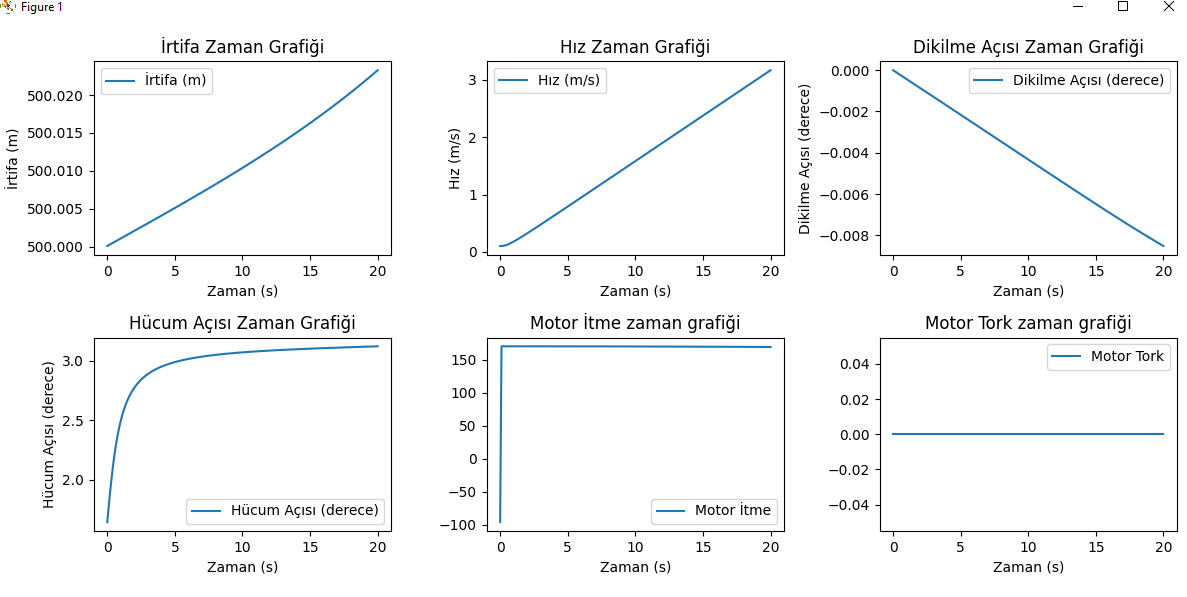
Düz sabit ve 500 metre yukardan uçuş koşullarını kabul ediyoruz

* **Pitch Trim:** Uçağın burun yukarı veya aşağı hareketini (pitch) dengelemek için elevator trim kullanılır. İdeal durumda, pitch momenti sıfırlanır, yani uçağın burunu ne yukarı ne de aşağı doğru bir moment hissetmez.
* **Roll Trim:** Uçağın bir kanadını diğerine göre yukarı veya aşağı hareketini (roll) dengelemek için aileron trim kullanılır. Roll momenti sıfırlanır ve uçak doğal bir şekilde bir yana eğilmez.
* **Yaw Trim:** Uçağın burun sol veya sağ hareketini (yaw) dengelemek için rudder trim kullanılır. Yaw momenti sıfırlanır ve uçak rüzgar veya aerodinamik asimetri nedeniyle bir yana kaymaz.

Vair 0 varsayılmıştır.

Simulasyon

1-Levye sabit durumda 20 saniyelik sabit irtifa sabit hız uçuşuna ait sonuçlar. Raporda İrtifa, hız, hareketli yüzey ve motor gücü girdileri, dikilme ve hücum açısının zamana bağlı grafikleri bulunmalıdır.



# Simülasyon parametreleri

simulasyon\_suresi = 20  # saniye

dt = 0.1  # Zaman adımı (saniye)

adim\_sayisi = int(simulasyon\_suresi / dt)

# Başlangıç durumu ve hava aracı parametreleri

initial\_state = [0.1, 0.1, 0.1, 0, 0, 0, 0, 0, 0, 0, 0, 500]  # 0 hız ve 500 yükseklik kabul edip başlatalım

hava\_araci\_parametreleri = {'mass': 13.5, 'Jxx': 0.8244, 'Jyy': 1.135, 'Jzz': 1.759,'Jxz':0.1204}

# Simülasyon döngüsü

for i in range(adim\_sayisi):

    if( hiz != [] ):

        Va = hiz[i-1]

    else:

        Va = 50 # başangıç hız 50

    motor = MotorModeli(rho,Sprop,Cprop,k\_motor,Ktp,K\_,e)

    motor\_itme, motor\_tork, = motor.itme\_tork\_hesapla(Va,δt)

    alfa , beta = alpha\_betha\_hesapla(initial\_state[2], initial\_state[0],Va)

    Fx, Fz, Fy, L, M, N= aerodinamik\_kuvvetler\_momentler(Va, rho, S, B, C, initial\_state[4],initial\_state[5],initial\_state[6],initial\_state[7],initial\_state[8],alfa,beta)

    kuvvetler\_momentler = [Fx, Fz, Fy, L, M, N]

    state\_derivative = hareket\_denklemleri(initial\_state, t, kuvvetler\_momentler, hava\_araci\_parametreleri)

    # birim zamanda değişiklikleri kaydet

    initial\_state = [initial\_state[j] + state\_derivative[j]\*0.001 for j in range(len(state\_derivative))]

                                                    # dt = 0.001

    # Sonuçları kaydet

    irtifa.append(initial\_state[11])

    hiz.append(np.sqrt(initial\_state[0]\*\*2 + initial\_state[1]\*\*2 + initial\_state[2]\*\*2))

    dikilme\_acisi.append(initial\_state[7])  # theta

    hucum\_acisi.append(np.arctan2(initial\_state[2], initial\_state[0]))  # atan(w/u)

    print(hiz)

    motor\_itme\_list.append(motor\_itme)

    motor\_tork\_list.append(motor\_tork)

Grafikler için :

# Grafikleri çiz

plt.figure(figsize=(12, 8))

plt.subplot(3, 3, 1)

plt.plot(zaman, irtifa, label='İrtifa (m)')

plt.xlabel('Zaman (s)')

plt.ylabel('İrtifa (m)')

plt.title('İrtifa Zaman Grafiği')

plt.legend()

plt.subplot(3,3, 2)

plt.plot(zaman, hiz, label='Hız (m/s)')

plt.xlabel('Zaman (s)')

plt.ylabel('Hız (m/s)')

plt.title('Hız Zaman Grafiği')

plt.legend()

plt.subplot(3, 3, 3)

plt.plot(zaman, dikilme\_acisi, label='Dikilme Açısı (derece)')

plt.xlabel('Zaman (s)')

plt.ylabel('Dikilme Açısı (derece)')

plt.title('Dikilme Açısı Zaman Grafiği')

plt.legend()

plt.subplot(3, 3, 4)

plt.plot(zaman, hucum\_acisi, label='Hücum Açısı (derece)')

plt.xlabel('Zaman (s)')

plt.ylabel('Hücum Açısı (derece)')

plt.title('Hücum Açısı Zaman Grafiği')

plt.legend()

plt.subplot(3, 3, 5)

plt.plot(zaman, motor\_itme\_list , label='Motor İtme')

plt.xlabel('Zaman (s)')

plt.ylabel('')

plt.title('Motor İtme zaman grafiği')

plt.legend()

plt.subplot(3, 3, 6)

plt.plot(zaman,  motor\_tork\_list, label='Motor Tork')

plt.xlabel('Zaman (s)')

plt.ylabel('')

plt.title('Motor Tork zaman grafiği')

plt.legend()

plt.tight\_layout()

plt.show()

2-Levye sabit durumda sabit irtifa sabit hız uçuşu yapılırken simülasyonun 10. saniyesinde elevatör açısı 0.2 saniye için 2° yapılmalı ve sonrasında trim noktasına geri çekilmelidir. 20 saniyelik simülasyon sonuçları istenmektedir. Raporda irtifa, hız, hareketli yüzey ve motor gücü girdileri, dikilme ve hücum açısının zamana bağlı grafikleri bulunmalıdır.

def elevatör\_etkisi(t):

    δe,δa = 0,0

    if 10.0 <= t <= 10.2:

        # iki tarafında aynı yönde olduğunu varsaydık

        δer = elevator\_acisi

yazı tipi, beyaz, diyagram, daire içeren bir resim

Açıklama otomatik olarak oluşturuldu        δel = elevator\_acisi

        δe = δer + δel

        δa = -δer + δel

        return δe,δa

    else:

        return δe,δa

elavatör açısı saniyler içinde değişimi bu metod da ele alınmıştır

ve açıya ait trimleme işlemi de burda yapılmıştır

Simulasyon döngüsü içinde bu şekilde kullanışmıştır

# Simülasyon döngüsü

for i in range(adim\_sayisi):

    t = dt \* i

    δe,δa = elevatör\_etkisi(t)

aerodinamik\_kuvvetler\_momentler(“parametreler..”,δe,δa)

metin, diyagram, sayı, numara, çizgi içeren bir resim

Açıklama otomatik olarak oluşturulduSimulasyon Grafiği :

3-Levye sabit durumda sabit irtifa sabit hız uçuşu yapılırken simülasyonun 10. Saniyesinde kanatçık açısı 0.2 saniye için 2° yapılmalı ve sonrasında trim noktasına geri çekilmelidir. 30 saniyelik simülasyon sonuçları istenmektedir. Raporda irtifa, hız, hareketli yüzey ve motor gücü girdileri, yatış, yönelme, dikilme ve hücum açısının zamana bağlı grafikleri bulunmalıdır.

def kanatcik\_etkisi(t):

    δe,δa = 0,0

    if 10.0 <= t <= 10.2:

        # iki tarafında aynı yönde olmadığını varsaydık

yazı tipi, beyaz, diyagram, daire içeren bir resim

Açıklama otomatik olarak oluşturuldu        δer = kanatcik\_acisi

        δel = -kanatcik\_acisi

        δe = δer + δel

        δa = -δer + δel

        return δe,δa

    else:

        return δe,δa

Kanatçıklar biri + yöne diğeri – yöne alındı.

Simulasyon döngüsü içinde:

# Simülasyon döngüsü

for i in range(adim\_sayisi):

    δe,δa = kanatcik\_etkisi(t)

aerodinamik\_kuvvetler\_momentler(“parametreler..”,δe,δa)

Grafik için koda eklenenle :

plt.subplot(3, 3, 7)

plt.plot(zaman,  yatis\_acisi, label='Yatış Açısı')

plt.xlabel('Zaman (s)')

plt.ylabel('')

plt.title('Yatış Açısı zaman grafiği')

plt.legend()

plt.subplot(3, 3, 8)

plt.plot(zaman,  yönelme\_acisi, label='Yönelme Açısı')

plt.xlabel('Zaman (s)')

plt.ylabel('')

plt.title('Yönelme Açısı zaman grafiği')

plt.legend()

Simulasyon Grafiği:

metin, diyagram, çizgi, plan içeren bir resim

Açıklama otomatik olarak oluşturuldu