AE352 Project Notes

Braden Cardamone

November 2024

1 EOM's

$$
\begin{aligned}
m\ddot{x} &= T(\sin(\phi)\sin(\psi) + \cos(\phi)\sin(\theta)\cos(\psi)) \\
m\ddot{y} &= T(\sin(\phi)\cos(\psi) - \cos(\phi)\sin(\theta)\cos(\psi)) \\
m\ddot{z} &= T\cos(\phi)\cos(\psi) - g \\
I_x\ddot{\phi} &= l(T_1 - T_2 - T_3 + T_4) + (I_y - I_z)\dot{\theta}\dot{\psi} \\
I_y\ddot{\theta} &= l(T_1 + T_2 - T_3 - T_4) + (I_z - I_x)\dot{\phi}\dot{\psi} \\
I_z\ddot{\psi} &= \frac{m_{rotors}}{m}(T_1 - T_2 + T_3 - T_4) + (I_x - I_z)\dot{\phi}\dot{\theta}
\end{aligned}
$$

2 Performance Goal 1

```
x0, y0, z0 = 0.0, 0.0, 1.0
phi0, theta0, psi0 = 0.0, 0.0, 0.0
xdot0, ydot0, zdot0 = 0.0, 0.0, 0.0
phidot0, thetadot0, psidot0 = 0.0, 0.0, 0.0


def T1(t):      # Front left
    return m * g / 4


def T2(t):      # Front right
    return m * g / 4


def T3(t):      # Back right
    return m * g / 4


def T4(t):      # Back left
    return m * g / 4
```

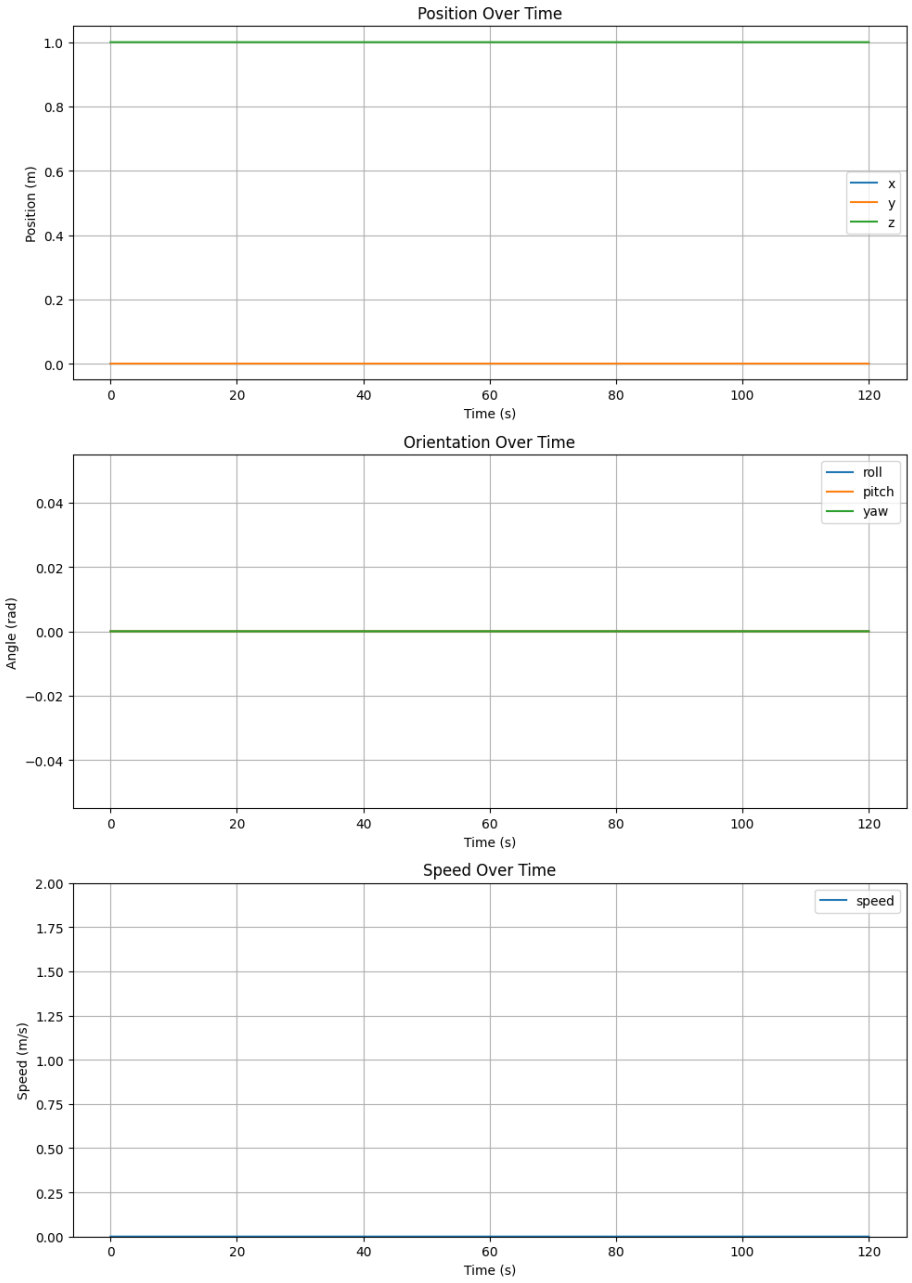

Figure 1: Performance Goal 1: 2D

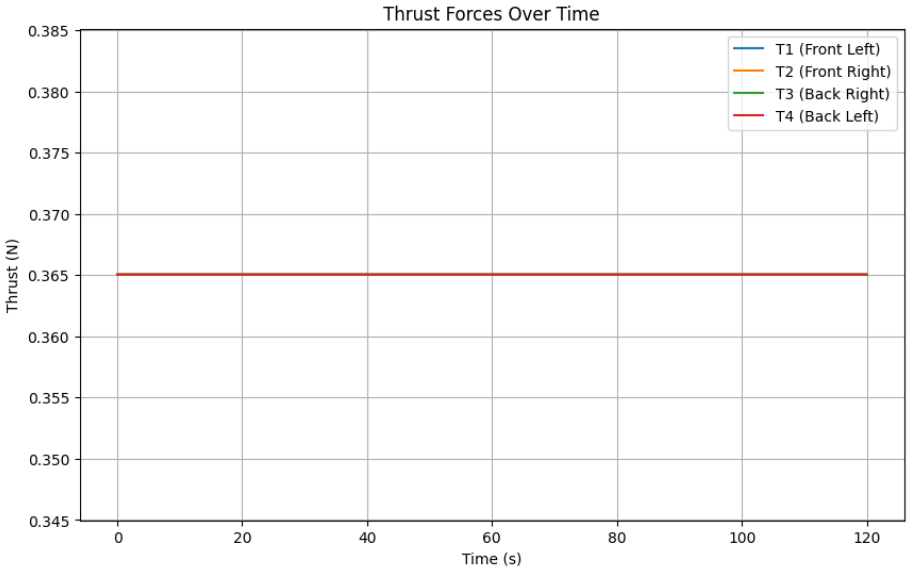

Figure 2: Performance Goal 1: Forces

3 Performance Goal 2

```
x0, y0, z0 = 0.0, 0.0, 1.0
phi0, theta0, psi0 = 0.0, 0.0, 0.0
xdot0, ydot0, zdot0 = 0.0, 0.0, 0.0
phidot0, thetadot0, psidot0 = 0.0, 0.0, 0.0


r = 2
v = 0.5
y0 = r
xdot0 = v
theta0 = np.arcsin(v**2 / (g * r))
psidot0 = 0.25


# Maximum thrust 1.53
def T1(t):      # Front left
    return m * g / (4*np.cos(theta0))

def T2(t):      # Front right
    return m * g / (4*np.cos(theta0))

def T3(t):      # Back right
    return m * g / (4*np.cos(theta0))

def T4(t):      # Back left
    return m * g / (4*np.cos(theta0))
```

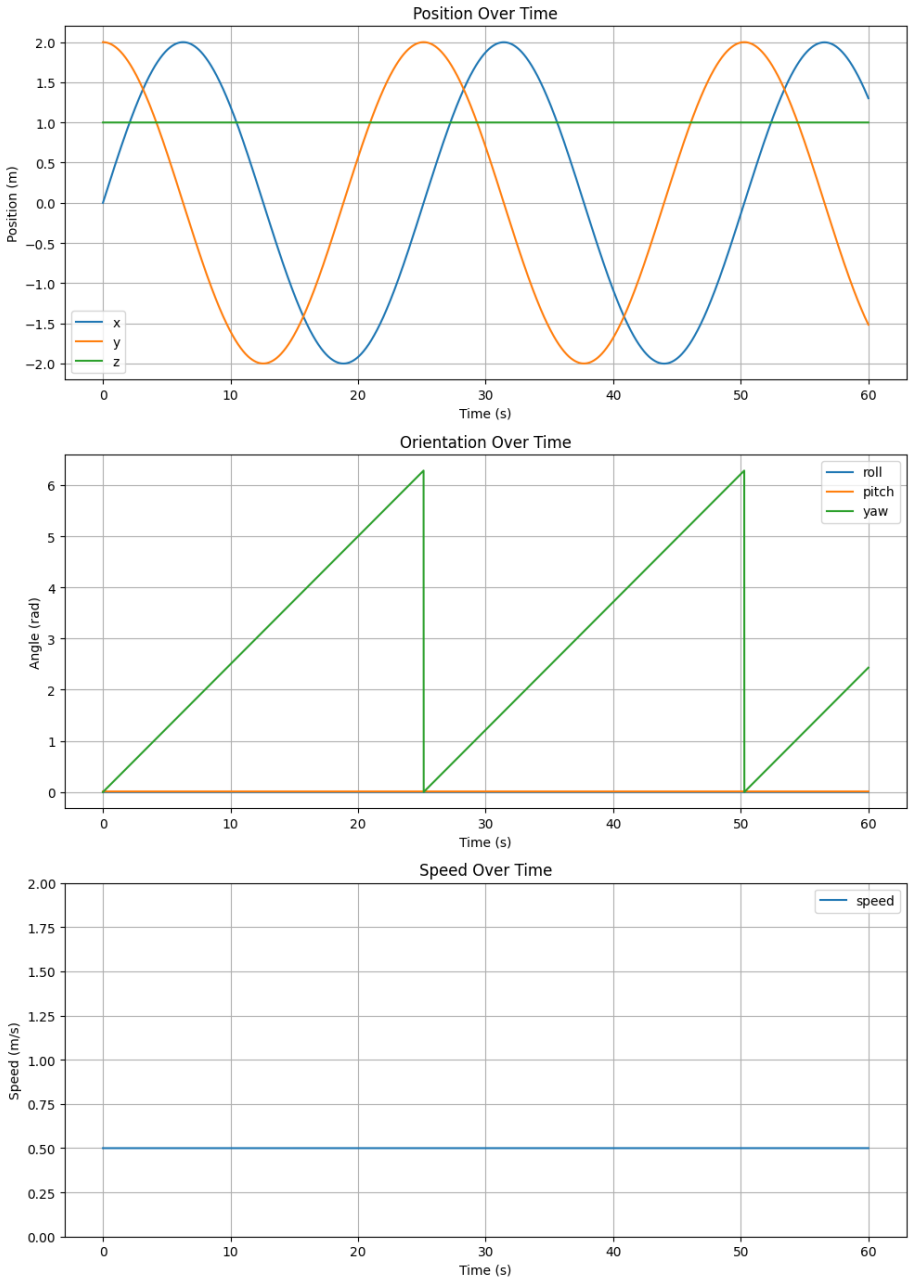

Figure 3: Performance Goal 2: 2D

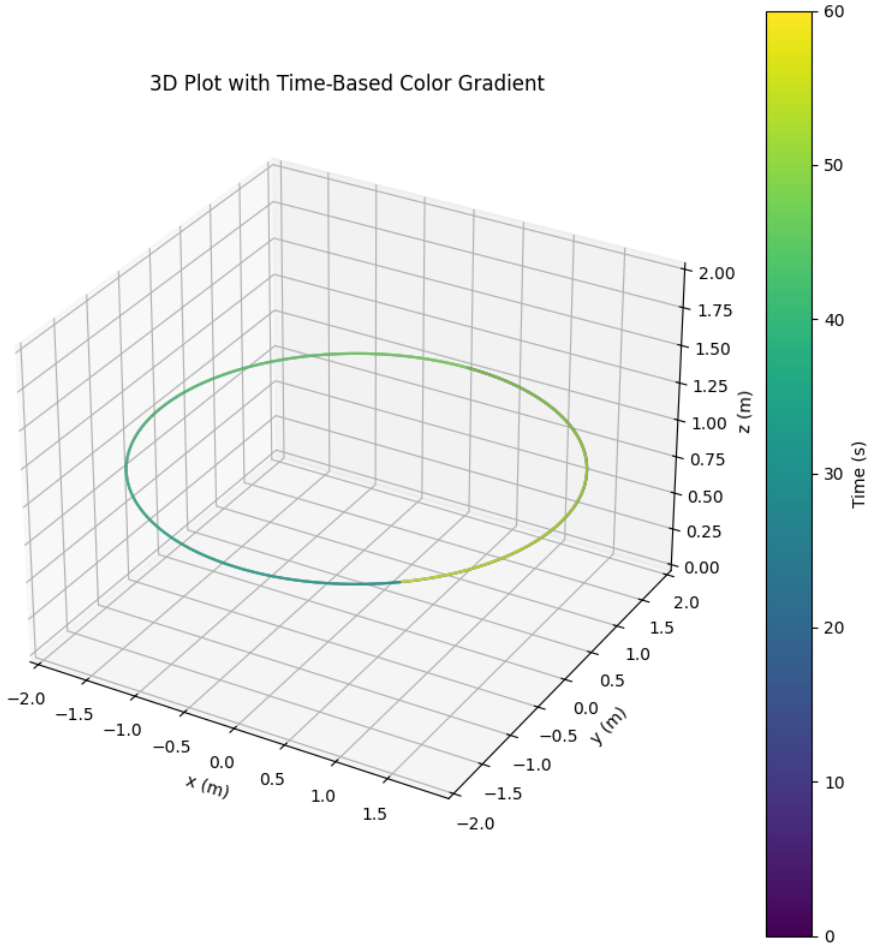

Figure 4: Performance Goal 2: 3D

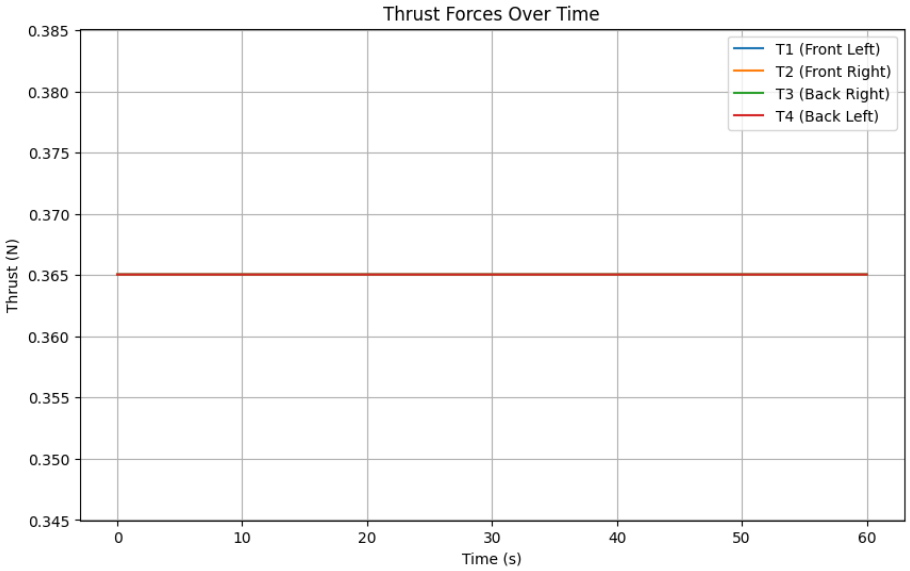

Figure 5: Performance Goal 2: Forces

4 Performance Goal 3

```
x0, y0, z0 = 0.0, 0.0, 0.0
phi0, theta0, psi0 = 0.0, 0.0, 0.0
xdot0, ydot0, zdot0 = 0.0, 0.0, 0.0
phidot0, thetadot0, psidot0 = 0.0, 0.0, 0.0


# Maximum thrust 1.53
def T1(t):      # Front left
    # Ascend
    if t < 1.4:
      T1 = 0.4
    elif t < 1.5:
      T1 = -0.12373

    # Fly forward
    elif t < 1.6:
      T1 = 0.34005
    elif t < 1.7:
      T1 = 0.4282
    elif t < 1.8:
      T1 = 0.4282
    elif t < 1.9:
      T1 = 0.34005
    elif t < 301.5:
      T1 = m * g / 4
    elif t < 301.6:
      T1 = 0.39005
    elif t < 301.7:
      T1 = 0.3782
    elif t < 301.8:
      T1 = 0.3782
    elif t < 301.9:
      T1 = 0.39005

    # Rest
    elif t < 302.9:
      T1 = m * g / 4

    # Turn left
    elif t < 303.9:
      T1 = 0.34457
    elif t < 304.9:
      T1 = 0.38553
```

```
    # Fly forward
    elif t < 305:
      T1 = 0.34005
    elif t < 305.1:
      T1 = 0.4282
    elif t < 305.2:
      T1 = 0.4282
    elif t < 305.3:
      T1 = 0.34005
    elif t < 604.9:
      T1 = m * g / 4
    elif t < 605:
      T1 = 0.39005
    elif t < 605.1:
      T1 = 0.3782
    elif t < 605.2:
      T1 = 0.3782
    elif t < 605.3:
      T1 = 0.39005

    # Descend
    else: T1 = m * g / 4.001

    return T1

def T2(t):      # Front right
    if t < 1.4:
      T2 = 0.4
    elif t < 1.5:
      T2 = -0.12373

    elif t < 1.6:
      T2 = 0.34005
    elif t < 1.7:
      T2 = 0.4282
    elif t < 1.8:
      T2 = 0.4282
    elif t < 1.9:
      T2 = 0.34005
    elif t < 301.5:
      T2 = m * g / 4
    elif t < 301.6:
      T2 = 0.39005
    elif t < 301.7:
      T2 = 0.3782
    elif t < 301.8:
```

```
      T2 = 0.3782
    elif t < 301.9:
      T2 = 0.39005

    elif t < 302.9:
      T2 = m * g / 4

    elif t < 303.9:
      T2 = 0.38553
    elif t < 304.9:
      T2 = 0.34457

    elif t < 305:
      T2 = 0.34005
    elif t < 305.1:
      T2 = 0.4282
    elif t < 305.2:
      T2 = 0.4282
    elif t < 305.3:
      T2 = 0.34005
    elif t < 604.9:
      T2 = m * g / 4
    elif t < 605:
      T2 = 0.39005
    elif t < 605.1:
      T2 = 0.3782
    elif t < 605.2:
      T2 = 0.3782
    elif t < 605.3:
      T2 = 0.39005

    else: T2 = m * g / 4.001

    return T2

def T3(t):      # Back right
    if t < 1.4:
      T3 = 0.4
    elif t < 1.5:
      T3 = -0.12373

    elif t < 1.6:
      T3 = 0.39005
    elif t < 1.7:
      T3 = 0.3782
    elif t < 1.8:
```

```
        T3 = 0.3782
    elif t < 1.9:
        T3 = 0.39005
    elif t < 301.5:
        T3 = m * g / 4
    elif t < 301.6:
        T3 = 0.34005
    elif t < 301.7:
        T3 = 0.4282
    elif t < 301.8:
        T3 = 0.4282
    elif t < 301.9:
        T3 = 0.34005

    elif t < 302.9:
        T3 = m * g / 4

    elif t < 303.9:
        T3 = 0.34457
    elif t < 304.9:
        T3 = 0.38553

    elif t < 305:
        T3 = 0.39005
    elif t < 305.1:
        T3 = 0.3782
    elif t < 305.2:
        T3 = 0.3782
    elif t < 305.3:
        T3 = 0.39005
    elif t < 604.9:
        T3 = m * g / 4
    elif t < 605:
        T3 = 0.34005
    elif t < 605.1:
        T3 = 0.4282
    elif t < 605.2:
        T3 = 0.4282
    elif t < 605.3:
        T3 = 0.34005

    else: T3 = m * g / 4.001

    return T3

def T4(t):      # Back left
```

```
if t < 1.4:
  T4 = 0.4        # Lift up
elif t < 1.5:
  T4 = -0.12373   # Stop lift

elif t < 1.6:
  T4 = 0.39005    # Tilt forward
elif t < 1.7:
  T4 = 0.3782     # Stop tilt
elif t < 1.8:
  T4 = 0.3782     # Tilt backward
elif t < 1.9:
  T4 = 0.39005    # Stop tilt
elif t < 301.5:
  T4 = m * g / 4  # Fly straight
elif t < 301.6:
  T4 = 0.34005    # Tilt backward
elif t < 301.7:
  T4 = 0.4282     # Stop tilt
elif t < 301.8:
  T4 = 0.4282     # Tilt forward
elif t < 301.9:
  T4 = 0.34005    # Stop tilt

elif t < 302.9:
  T4 = m * g / 4  # Rest

elif t < 303.9:
  T4 = 0.38553    # Yaw left
elif t < 304.9:
  T4 = 0.34457    # Stop yaw

elif t < 305:
  T4 = 0.39005    # Tilt forward
elif t < 305.1:
  T4 = 0.3782     # Stop tilt
elif t < 305.2:
  T4 = 0.3782     # Tilt backward
elif t < 305.3:
  T4 = 0.39005    # Stop tilt
elif t < 604.9:
  T4 = m * g / 4  # Fly straight
elif t < 605:
  T4 = 0.34005    # Tilt backward
elif t < 605.1:
  T4 = 0.4282     # Stop tilt
```

```
elif t < 605.2:
  T4 = 0.4282     # Tilt forward
elif t < 605.3:
  T4 = 0.34005    # Stop tilt

else: T4 = m * g / 4.001  # Descend

return T4
```

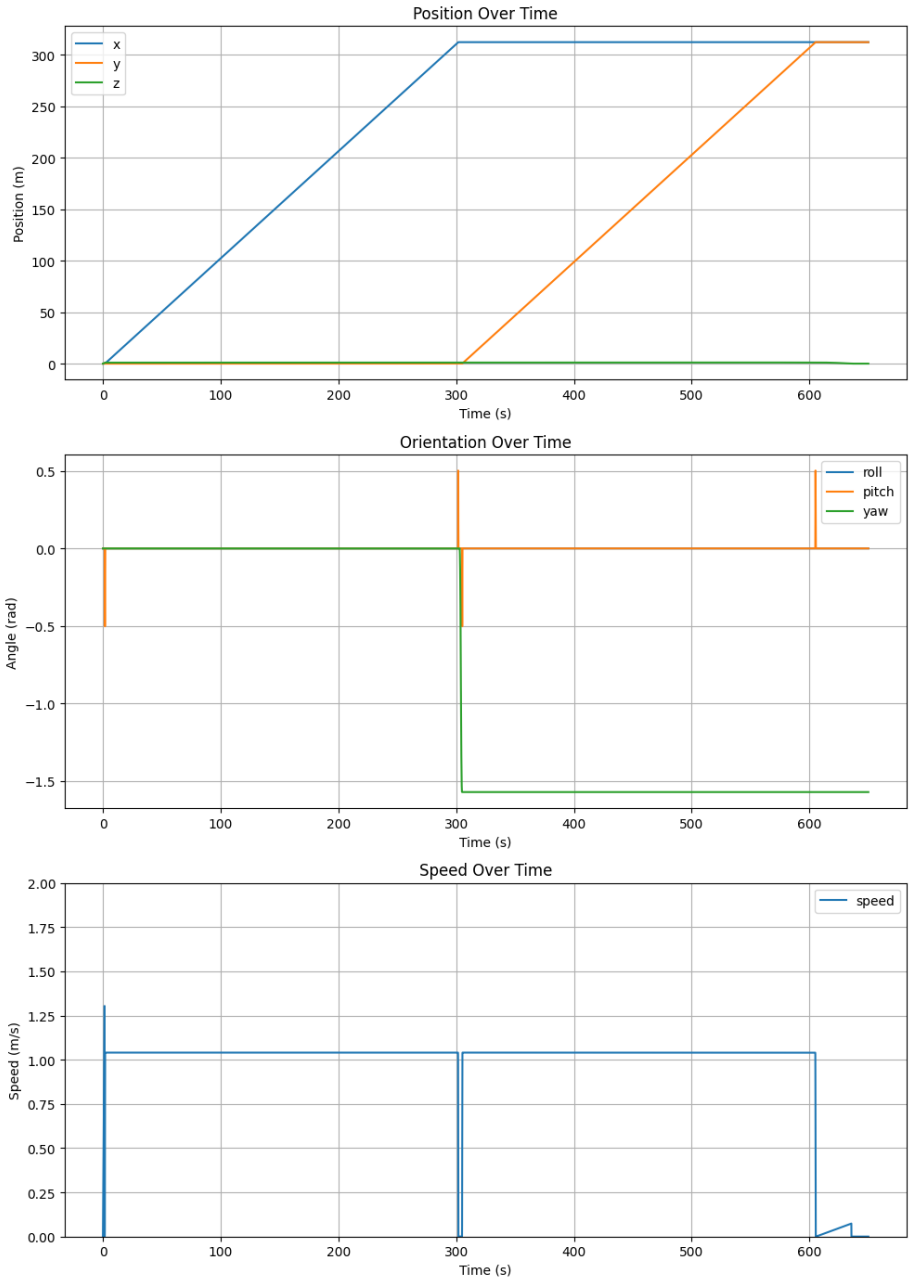

Figure 6: Performance Goal 3: 2D

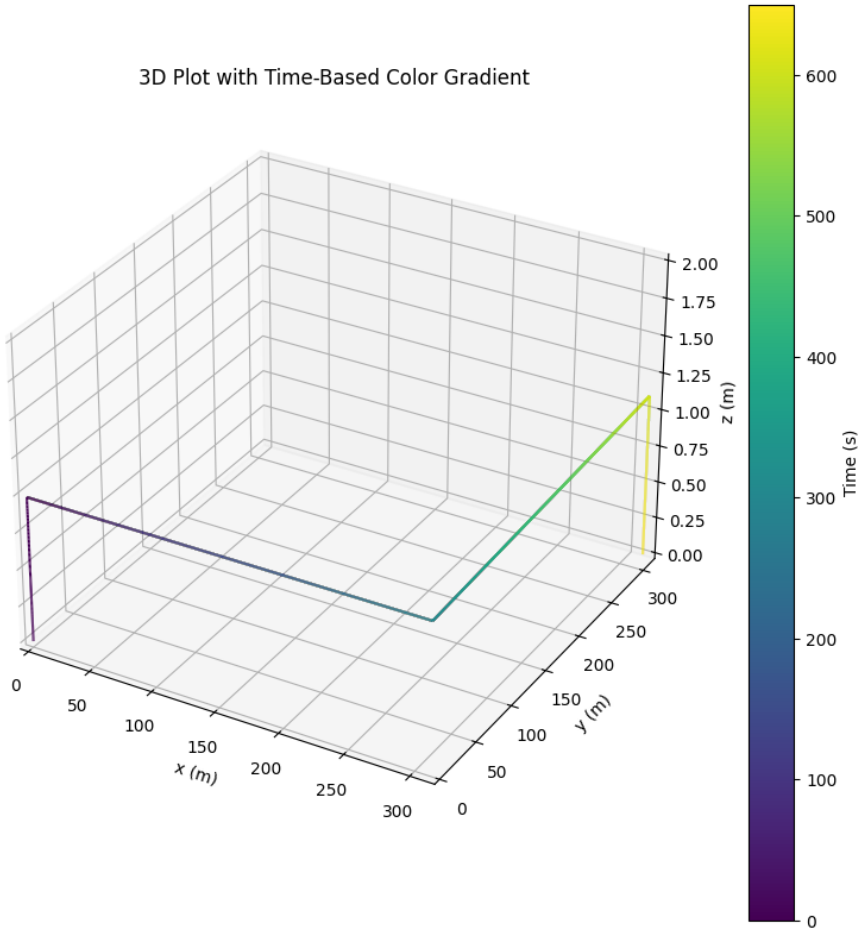

Figure 7: Performance Goal 3: 3D

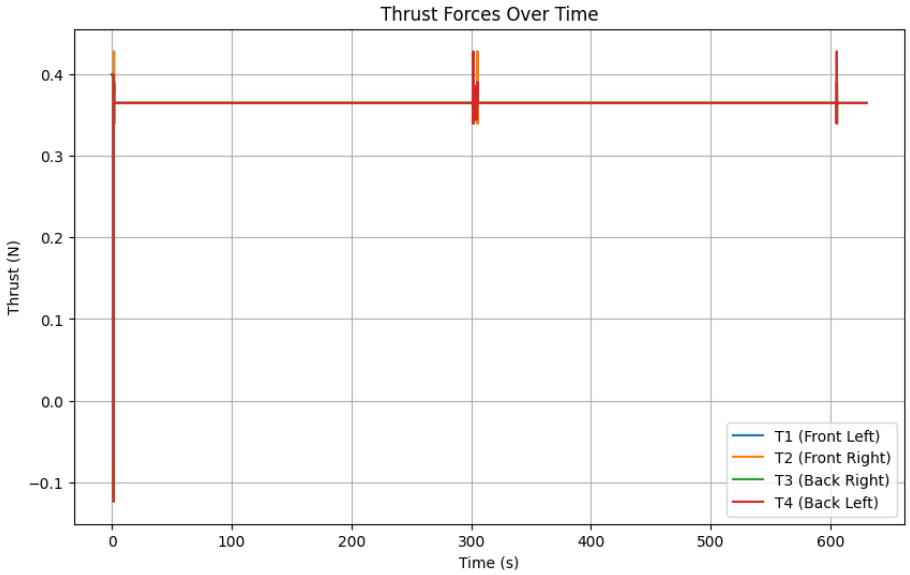

Figure 8: Performance Goal 3: Forces

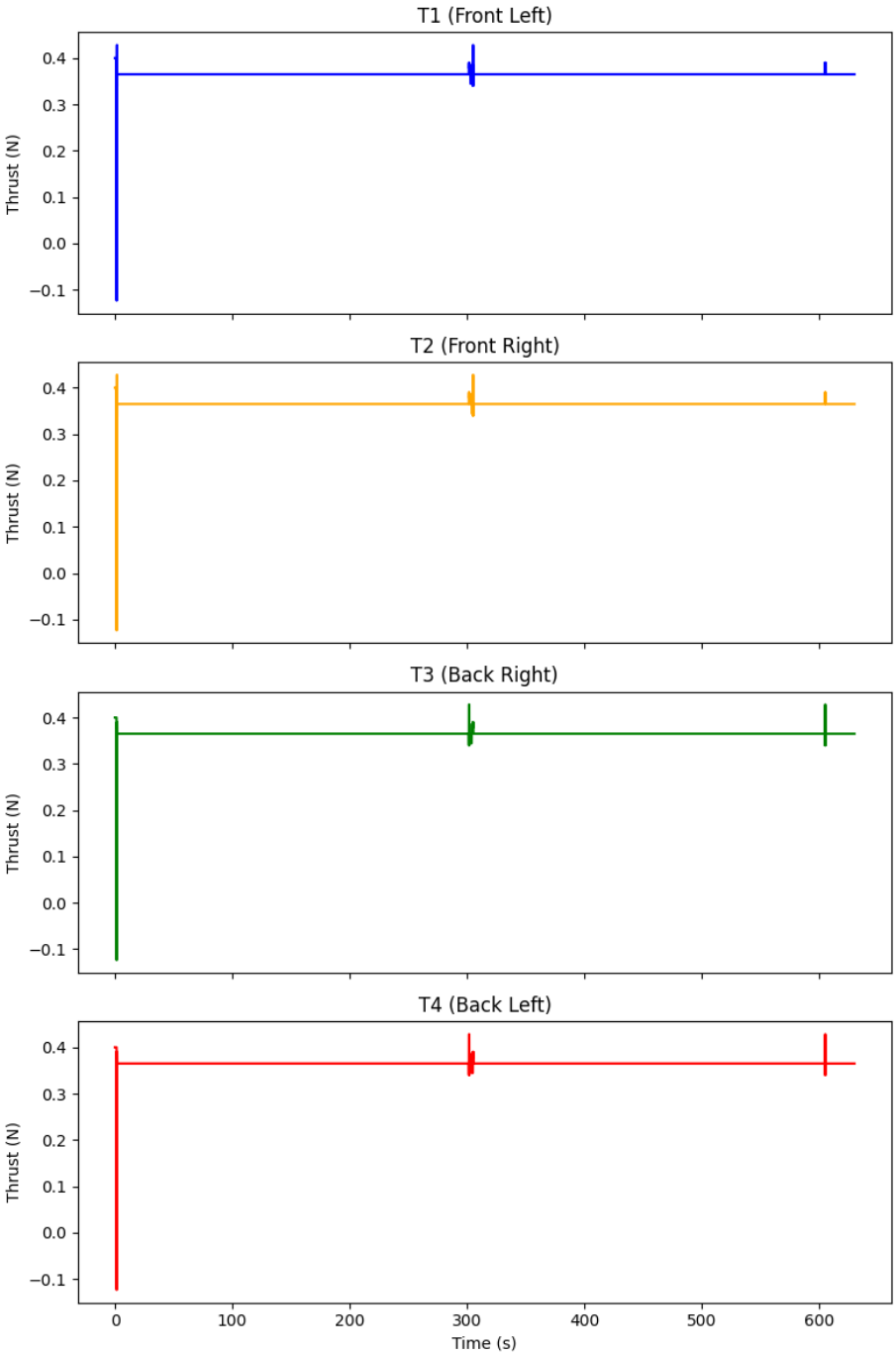

Figure 9: Performance Goal 3: Forces