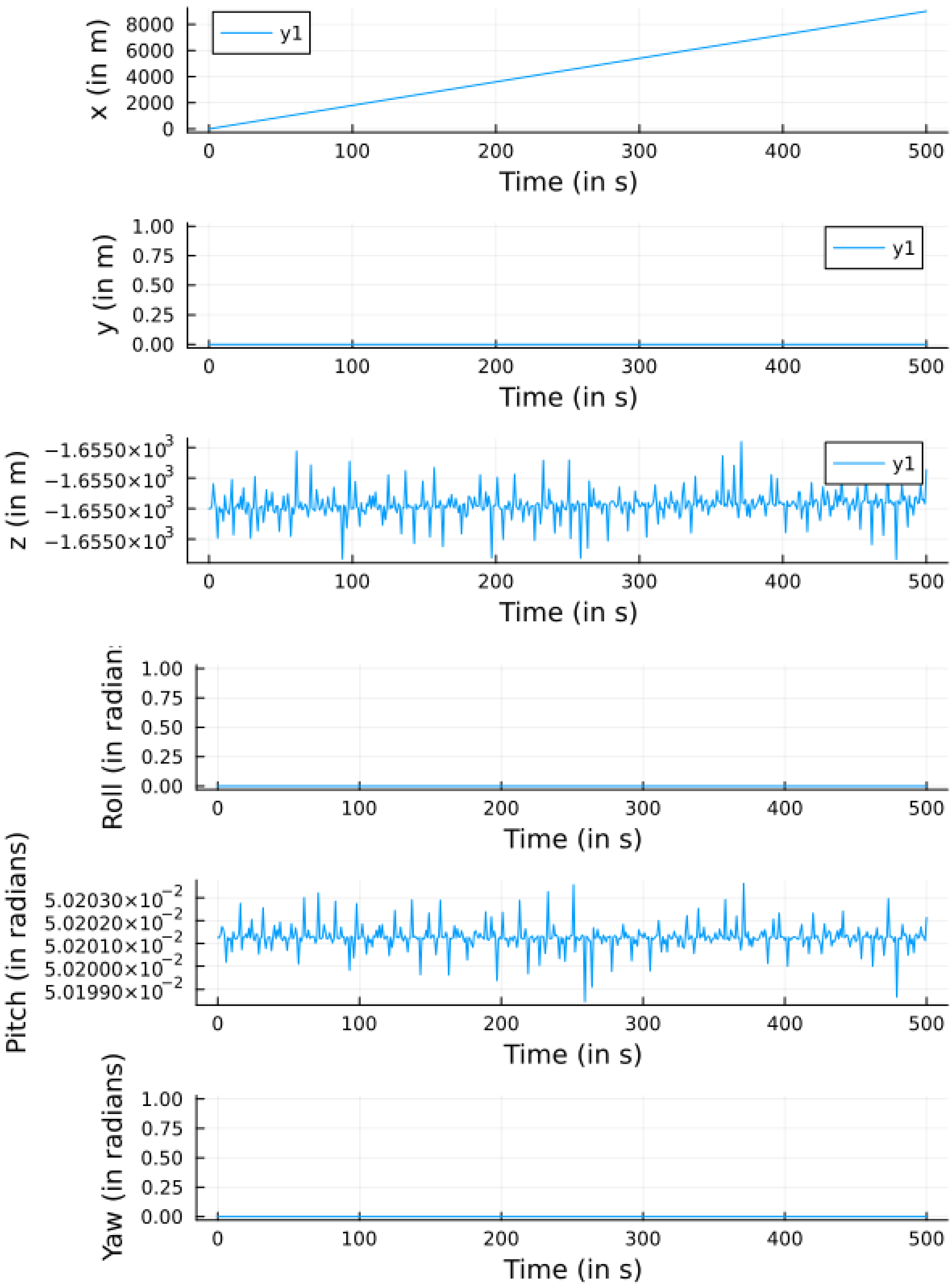
```julia
include("definitions.jl")
include("utils.jl")
include("matlab_utils.jl")
include("aircraft_eom.jl")

using LinearAlgebra
using Optim

#=
***************************************************************************************************
Functions for Part 1
=#

function GetStateAndControl(trim_definition::TrimDefinitionSL,trim_variables::TrimVariablesSL)

    #Parameters that don't matter
    x = y = ψ = 0.0
    #Parameters that are zero
    ϕ = v = p = q = r = 0.0
    #=
    Since there is no wind, γ = γ_a.
    Thus, pitch θ = flight path angle γ + angle of attack α
    Also, this is constant altitude flight. So, flight path angle, γ=0.
    =#
    θ = trim_definition.γ + trim_variables.α
    z = -trim_definition.h
    #For straight, wings-level, there is no side slip β
    β = 0.0
    wind_angles = WindAngles(trim_definition.Va,β,trim_variables.α)
    Va_vector = WindAnglesToAirRelativeVelocityVector(wind_angles)
    u = Va_vector[1]
    w = Va_vector[3]
    state = AircraftState(x,y,z,ϕ,θ,ψ,u,v,w,p,q,r)

    de = trim_variables.δe
    da = dr = 0.0
    dt = trim_variables.δt
    control = AircraftControl(de,da,dr,dt)

    return state,control
end

function GetCost(trim_definition::TrimDefinitionSL,trim_variables::TrimVariablesSL,aircraft_parameters::AircraftParameters)

    state,control = GetStateAndControl(trim_definition,trim_variables)
    wind_inertial = [0.0,0.0,0.0]
    rho = stdatmo(-state.z)
    force, moment = AircraftForcesAndMoments(state, control, wind_inertial, rho, aircraft_parameters)
    cost = norm(force,2)^2 + norm(moment,2)^2
    return cost
end

function OptimizerCostFunction(params::Vector{Float64},trim_definition::TrimDefinitionSL,aircraft_parameters::AircraftParameters)
    trim_variables = TrimVariablesSL(params...)
    cost = GetCost(trim_definition,trim_variables,aircraft_parameters)
    return cost
end

function GetTrimConditions(trim_definition::TrimDefinitionSL,aircraft_parameters::AircraftParameters)
    lower = [-pi/4,-pi/4,0.0]
    upper = [pi/4,pi/4,1.0]
    initial_tv = [0.5, 0.5, 0.5]
    results = optimize(x->OptimizerCostFunction(x,trim_definition,aircraft_parameters), lower, upper, initial_tv)
    trim_variables_list = results.minimizer
    trim_variables = TrimVariablesSL(trim_variables_list...)
    state, control = GetStateAndControl(trim_definition, trim_variables)
    return state, control, results
end

#=
***************************************************************************************************
Functions for Part 2
=#

function GetStateAndControl(trim_definition::TrimDefinitionCT,trim_variables::TrimVariablesCT)

    #Parameters that don't matter
    x = y = ψ = 0.0

    #Parameters that matter
    ϕ = trim_variables.ϕ
    z = -trim_definition.h
    #=
    Since there is no wind, γ = γ_a.
```

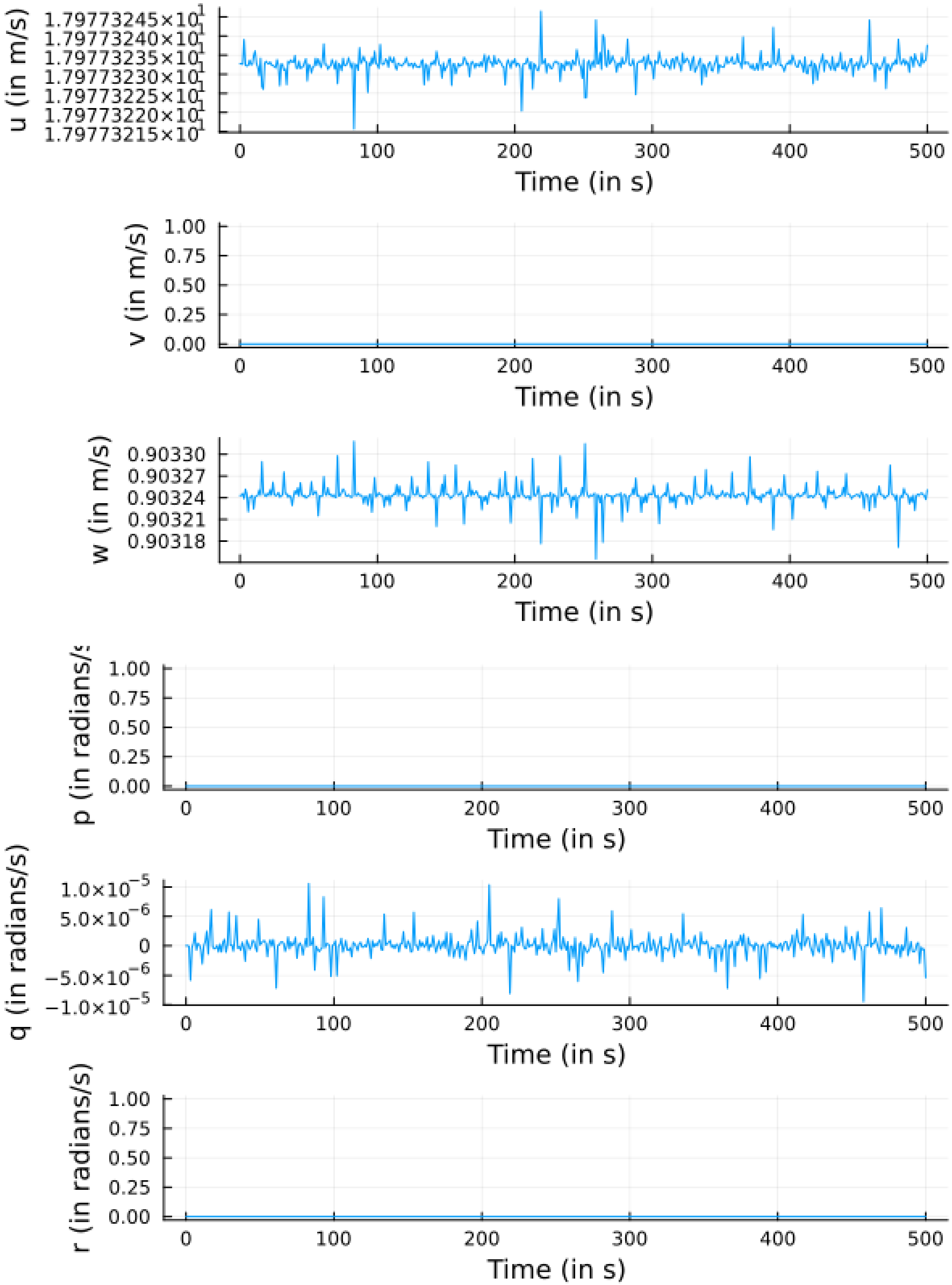
```
    Thus, pitch θ = flight path angle γ + angle of attack α
    =#
    θ = trim_definition.γ + trim_variables.α
    α = trim_variables.α
    β = trim_variables.β
    wind_angles = WindAngles(trim_definition.Va,β,α)
    Va_vector = WindAnglesToAirRelativeVelocityVector(wind_angles)
    u = Va_vector[1]
    v = Va_vector[2]
    w = Va_vector[3]
    #=
    Slides have defined the p,q,r terms using the rate of change of coarse angle χ. It is called chi
    χ_dot = (velocity_perpendicular_to_the_cirle)/R
    velocity_perpendicular_to_the_cirle = Va*cos(γ), where γ is the flight path angle.
    =#

    # R = (trim_definition.Va^2)/(9.81*tan(ϕ))
    R = trim_definition.R
    χ_dot = ( trim_definition.Va*cos(trim_definition.γ) )/ R
    p = -sin(θ)*χ_dot
    q = sin(ϕ)*cos(θ)*χ_dot
    r = cos(ϕ)*cos(θ)*χ_dot
    state = AircraftState(x,y,z,ϕ,θ,ψ,u,v,w,p,q,r)

    de = trim_variables.δe
    da = trim_variables.δa
    dr = trim_variables.δr
    dt = trim_variables.δt
    control = AircraftControl(de,da,dr,dt)

    return state,control
end

function GetCost(trim_definition::TrimDefinitionCT,trim_variables::TrimVariablesCT,aircraft_parameters::AircraftParameters)
    state,control = GetStateAndControl(trim_definition,trim_variables)
    wind_inertial = [0.0,0.0,0.0]
    rho = stdatmo(-state.z)
    tangent_speed = trim_definition.Va*cos(trim_definition.γ)
    centripetal_acceleration = (tangent_speed*tangent_speed)/trim_definition.R
    a_desired_inertial_frame = [0.0, centripetal_acceleration, 0.0]
    euler_angles = EulerAngles(state.roll, state.pitch, state.yaw)
    a_desired_body_frame = TransformFromInertialToBody(a_desired_inertial_frame,euler_angles)
    desired_force = aircraft_parameters.m*a_desired_body_frame
    aero_force, aero_moment = AeroForcesAndMomentsBodyStateWindCoeffs(state, control, wind_inertial, rho, aircraft_parameters)
    total_force, total_moment = AircraftForcesAndMoments(state, control, wind_inertial, rho, aircraft_parameters)
    force = total_force - desired_force
    cost = norm(force,2)^2 + norm(total_moment,2)^2 + aero_force[2]^2
    return cost
end

function OptimizerCostFunction(params::Vector{Float64},trim_definition::TrimDefinitionCT,aircraft_parameters::AircraftParameters)
    trim_variables = TrimVariablesCT(params...)
    cost = GetCost(trim_definition,trim_variables,aircraft_parameters)
    return cost
end

function GetTrimConditions(trim_definition::TrimDefinitionCT,aircraft_parameters::AircraftParameters)
    lower = [-pi/4,-pi/4,0.0,-pi/4,-pi/4,-pi/4,-pi/4]
    upper = [pi/4,pi/4,1.0,pi/4,pi/4,pi/4,pi/4]
    initial_tv = [0.5,0.5,0.5,0.5,0.5,0.5,0.5]
    results = optimize(x->OptimizerCostFunction(x,trim_definition,aircraft_parameters), lower, upper, initial_tv)
    trim_variables_list = results.minimizer
    trim_variables = TrimVariablesCT(trim_variables_list...)
    state, control = GetStateAndControl(trim_definition, trim_variables)
    return state, control, results
end

filename = "ttwistor.mat"
aircraft_parameters = AircraftParameters(filename)
case_num = 5
save_plots=true

#Part 1
if(case_num == 1)
    trim_definition = TrimDefinitionSL(18.0,0.0,1655)
    state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
    initial_state = collect(values(state))
    control_input = collect(values(control))
    wind_inertial = [0.0,0.0,0.0]
end

#Part 2
if(case_num == 2)
    trim_definition = TrimDefinitionSL(18.0,0.0,1655)
```

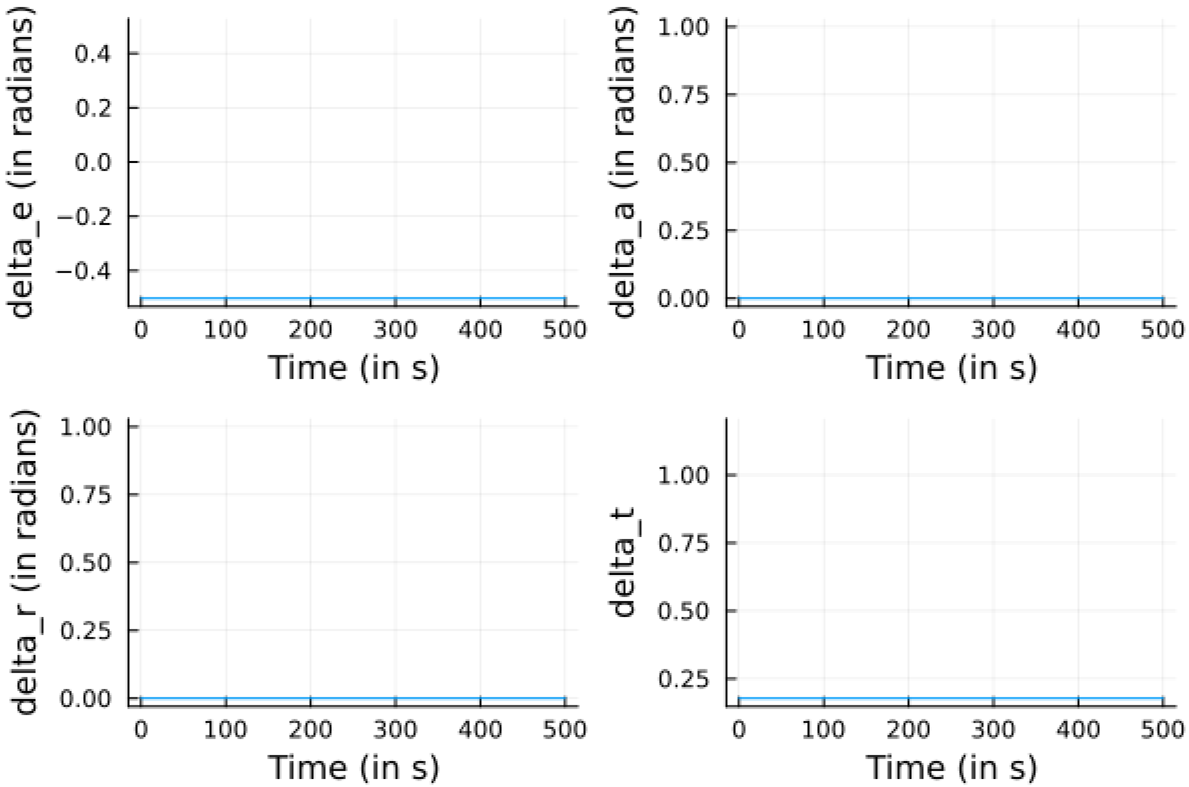
```
    state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
    euler_angles = EulerAngles(state.roll, state.pitch, state.yaw)
    initial_state = collect(values(state))
    wind_inertial = [10.0,10.0,0.0]
    wind_body = TransformFromInertialToBody(wind_inertial,euler_angles)
    initial_state[7] += wind_body[1]
    initial_state[8] += wind_body[2]
    initial_state[9] += wind_body[3]
    control_input = collect(values(control))
end

#Part 3
if(case_num == 3)
    trim_definition = TrimDefinitionSL(18.0,pi/18,1655)
    state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
    initial_state = collect(values(state))
    control_input = collect(values(control))
    wind_inertial = [0.0,0.0,0.0]
end

#Part 4
if(case_num == 4)
    trim_definition = TrimDefinitionCT(20.0,0.0,200,500.0)
    state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
    initial_state = collect(values(state))
    control_input = collect(values(control))
    wind_inertial = [0.0,0.0,0.0]
end

#HW 4 CT conditions
if(case_num == 5)
    trim_definition = TrimDefinitionCT(18.0,0.0,1655,500.0)
    state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
    initial_state = collect(values(state))
    control_input = collect(values(control))
    wind_inertial = [0.0,0.0,0.0]
end

time_interval = (0,500)
time_values = [i for i in 0:time_interval[2]]
trajectory_states = simulate(initial_state, time_interval, control_input, wind_inertial, aircraft_parameters)
control_array = [AircraftControl(control_input...) for i in 1:length(trajectory_states)]
PlotSimulation(time_values, trajectory_states, control_array , 'd', save_plots)
```

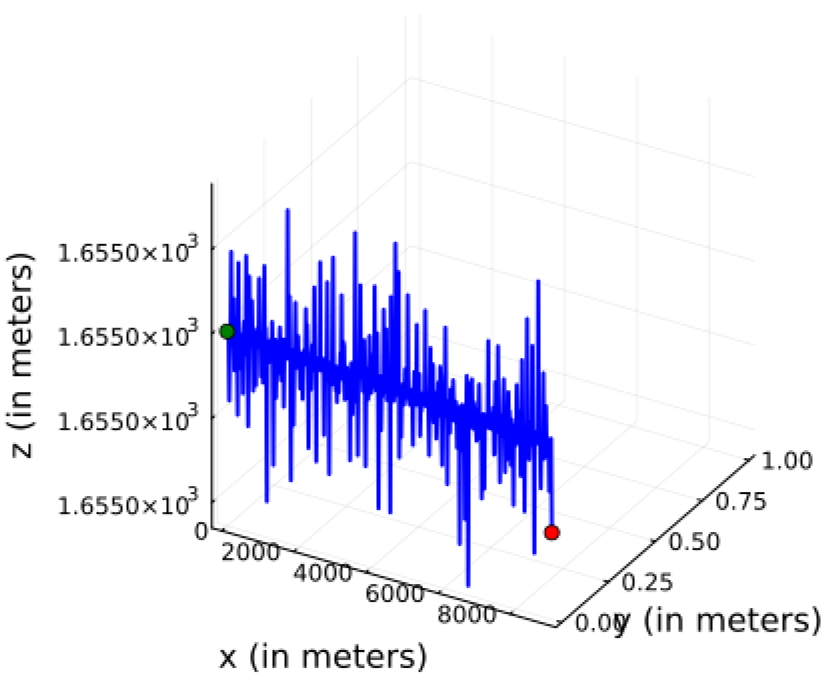
Problem 3.1



x (in m)
8000
6000
4000
2000
0
y1
0
100
200
300
400
500
Time (in s)
y (in m)
1.00
0.75
0.50
0.25
0.00
y1
Time (in s)
z (in m)
-1.6550×10^{3}
-1.6550×10^{3}
-1.6550×10^{3}
-1.6550×10^{3}
y1
Time (in s)
Roll (in radian
1.00
0.75
0.50
0.25
0.00
Time (in s)
Pitch (in radians)
5.02030×10^{-2}
5.02020×10^{-2}
5.02010×10^{-2}
5.02000×10^{-2}
5.01990×10^{-2}
Time (in s)
Yaw (in radians)
1.00
0.75
0.50
0.25
0.00
Time (in s)


u (in m/s)
1.79773245×10^1
1.79773240×10^1
1.79773235×10^1
1.79773230×10^1
1.79773225×10^1
1.79773220×10^1
1.79773215×10^1
Time (in s)
v (in m/s)
Time (in s)
w (in m/s)
0.90330
0.90327
0.90324
0.90321
0.90318
Time (in s)
p (in radians/s)
Time (in s)
q (in radians/s)
1.0×10^-5
5.0×10^-6
0
−5.0×10^-6
−1.0×10^-5
Time (in s)
r (in radians/s)
Time (in s)


delta_e (in radians)
0.4
0.2
0.0
−0.2
−0.4
delta_a (in radians)
1.00
0.75
0.50
0.25
0.00
delta_r (in radians)
delta_t
0
100
200
300
400
500
Time (in s)


z (in meters)
1.6550×10^3
0
2000
4000
6000
8000
x (in meters)
0.00
0.25
0.50
0.75
1.00
y (in meters)

Starting (Trim) State:

0.0
0.0
-1655.0
0.0
0.050201252250689686
0.0
17.977323271478497
0.0
0.9032430419093346
0.0
0.0
0.0

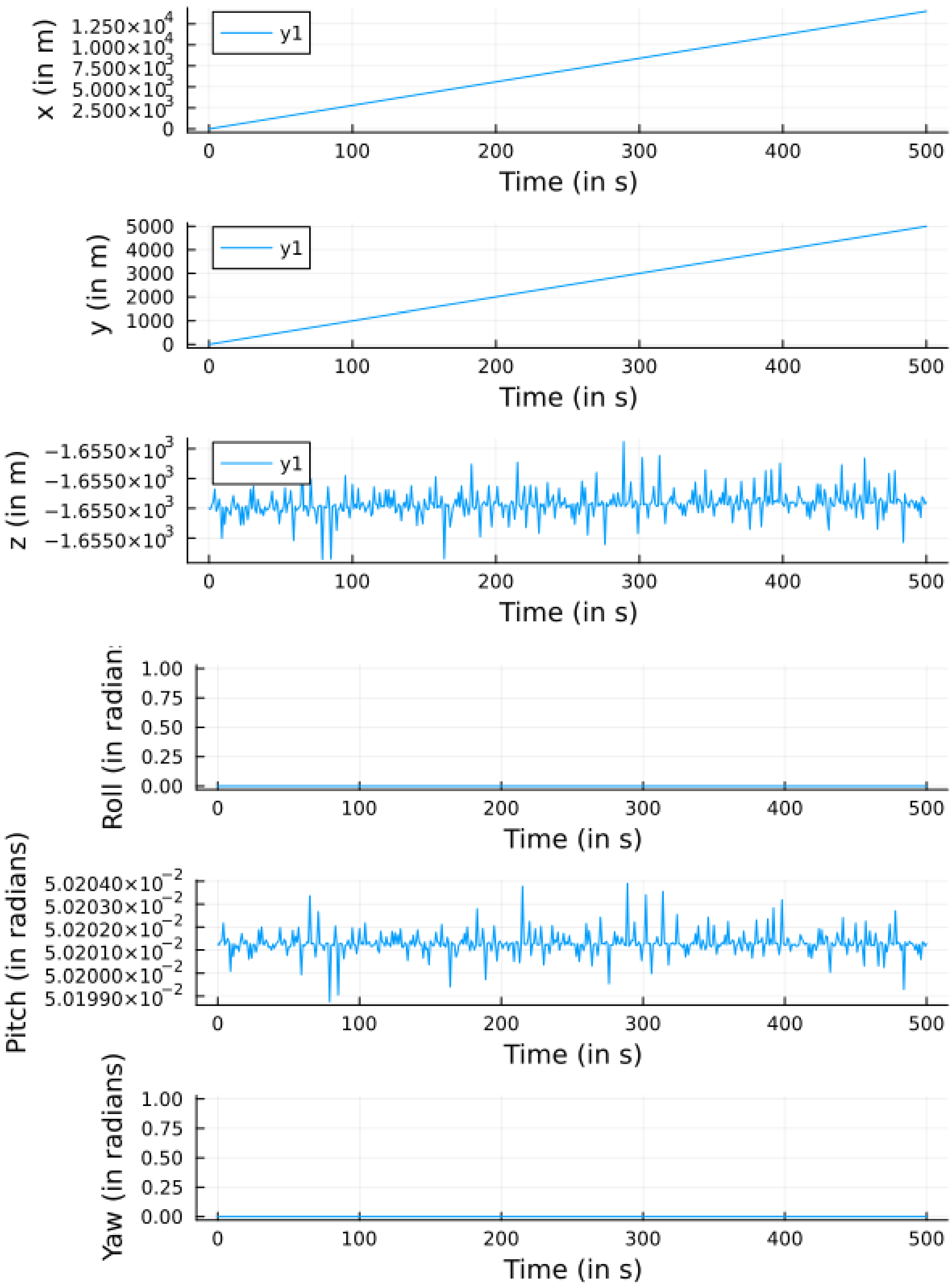
Trim Controls:

-0.5021558030179929
0.0
0.0
0.17764553656958398

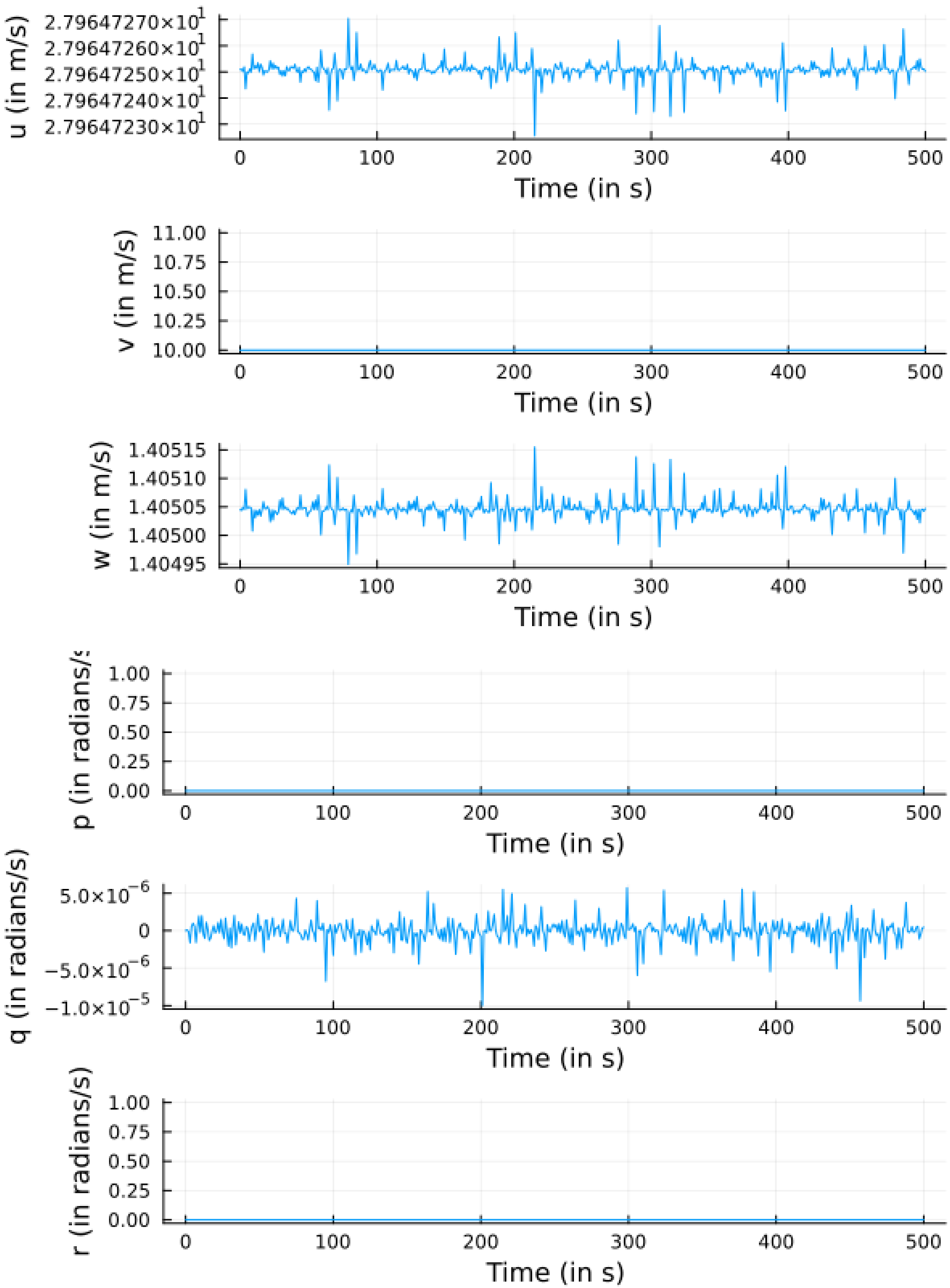
Trim Variables:

0.050201252250689686
-0.5021558030179929
0.17764553656958398

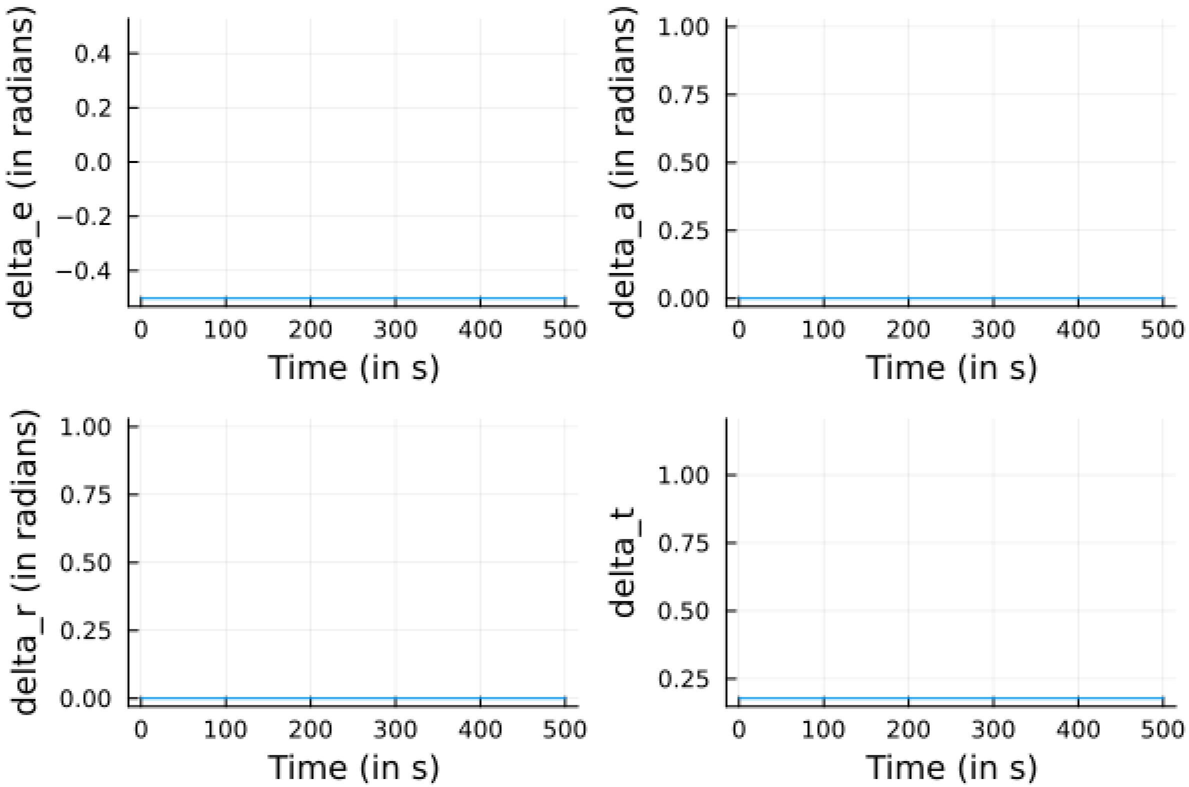
Problem 3.2

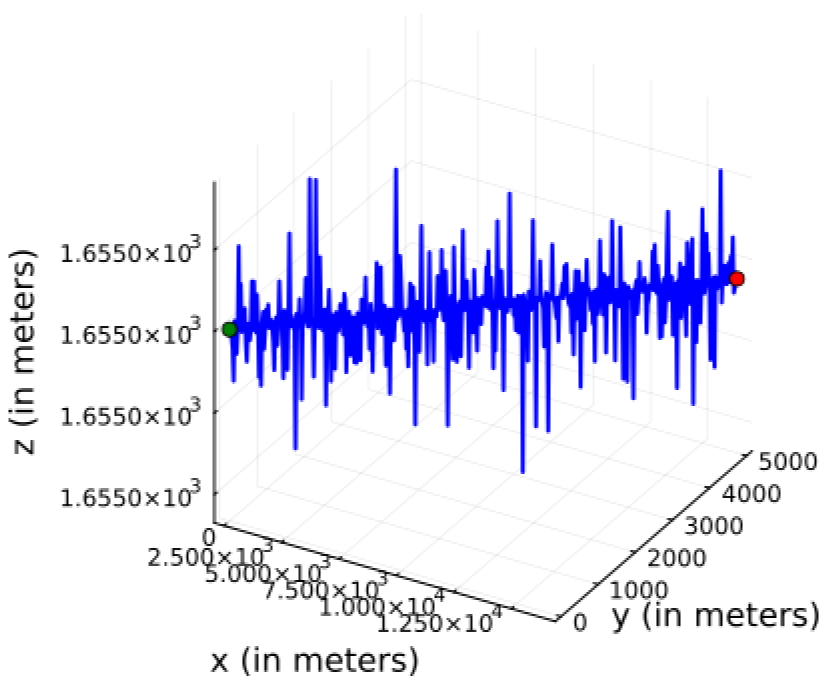


x (in m)
y (in m)
z (in m)
Roll (in radian
Pitch (in radians)
Yaw (in radians)
Time (in s)
y1
1.250×10^4
1.000×10^4
7.500×10^3
5.000×10^3
2.500×10^3
5000
4000
3000
2000
1000
−1.6550×10^3
1.00
0.75
0.50
0.25
0.00
5.02040×10^−2
5.02030×10^−2
5.02020×10^−2
5.02010×10^−2
5.02000×10^−2
5.01990×10^−2
0
100
200
300
400
500



u (in m/s)
2.79647270×10^1
2.79647260×10^1
2.79647250×10^1
2.79647240×10^1
2.79647230×10^1
v (in m/s)
11.00
10.75
10.50
10.25
10.00
w (in m/s)
1.40515
1.40510
1.40505
1.40500
1.40495
p (in radians/s)
1.00
0.75
0.50
0.25
0.00
q (in radians/s)
5.0×10^-6
0
−5.0×10^-6
−1.0×10^-5
r (in radians/s)
1.00
0.75
0.50
0.25
0.00
0
100
200
300
400
500
Time (in s)


delta_e (in radians)
delta_a (in radians)
delta_r (in radians)
delta_t
Time (in s)


z (in meters)
x (in meters)
y (in meters)

Starting (Trim) State:

0.0
0.0
-1655.0
0.0
0.050201252250689686
0.0
27.964725088966553
10.0
1.405044731858965
0.0
0.0
0.0

Trim Controls:

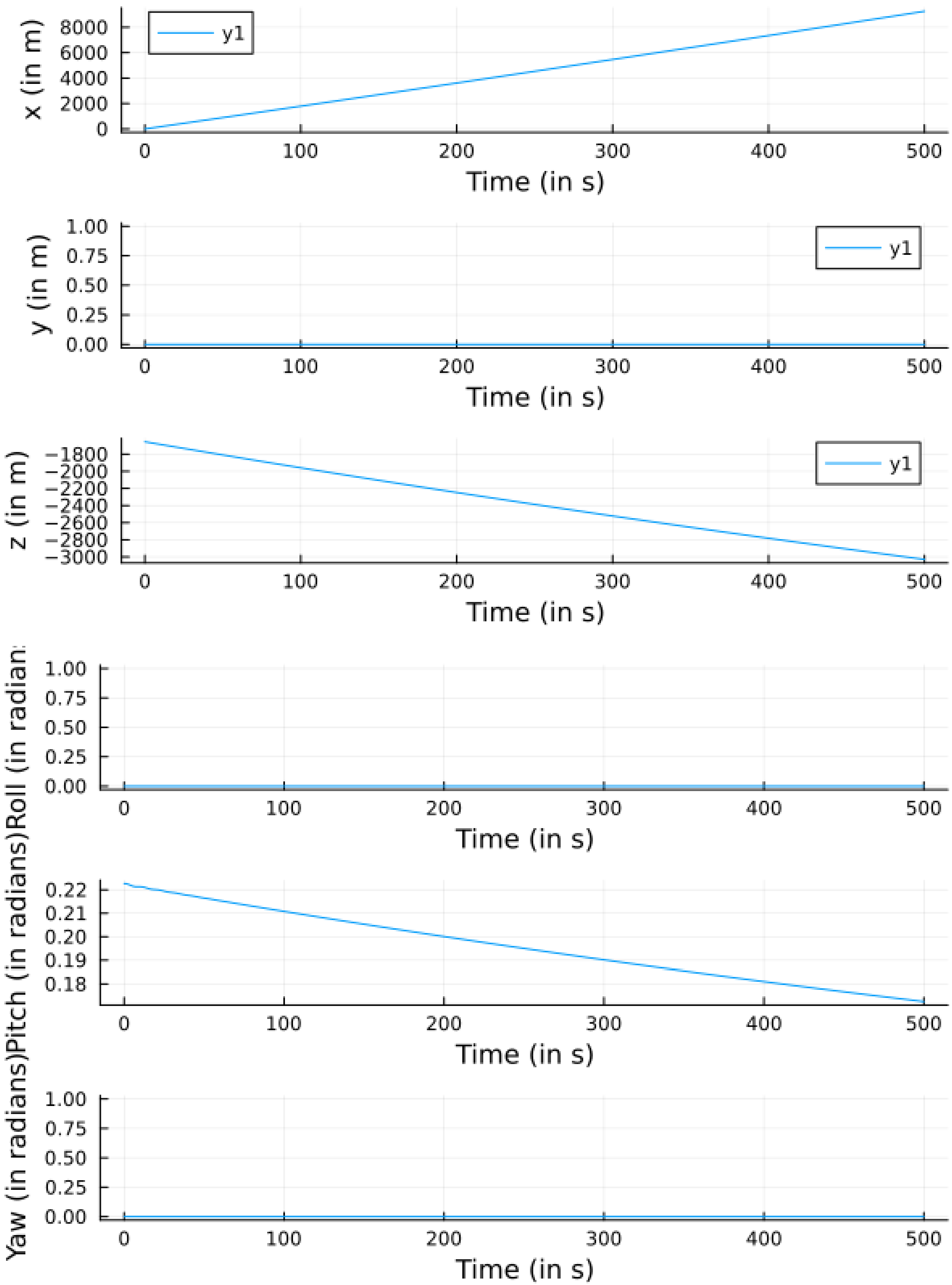
-0.5021558030179929
0.0
0.0
0.17764553656958398

Trim Variables:

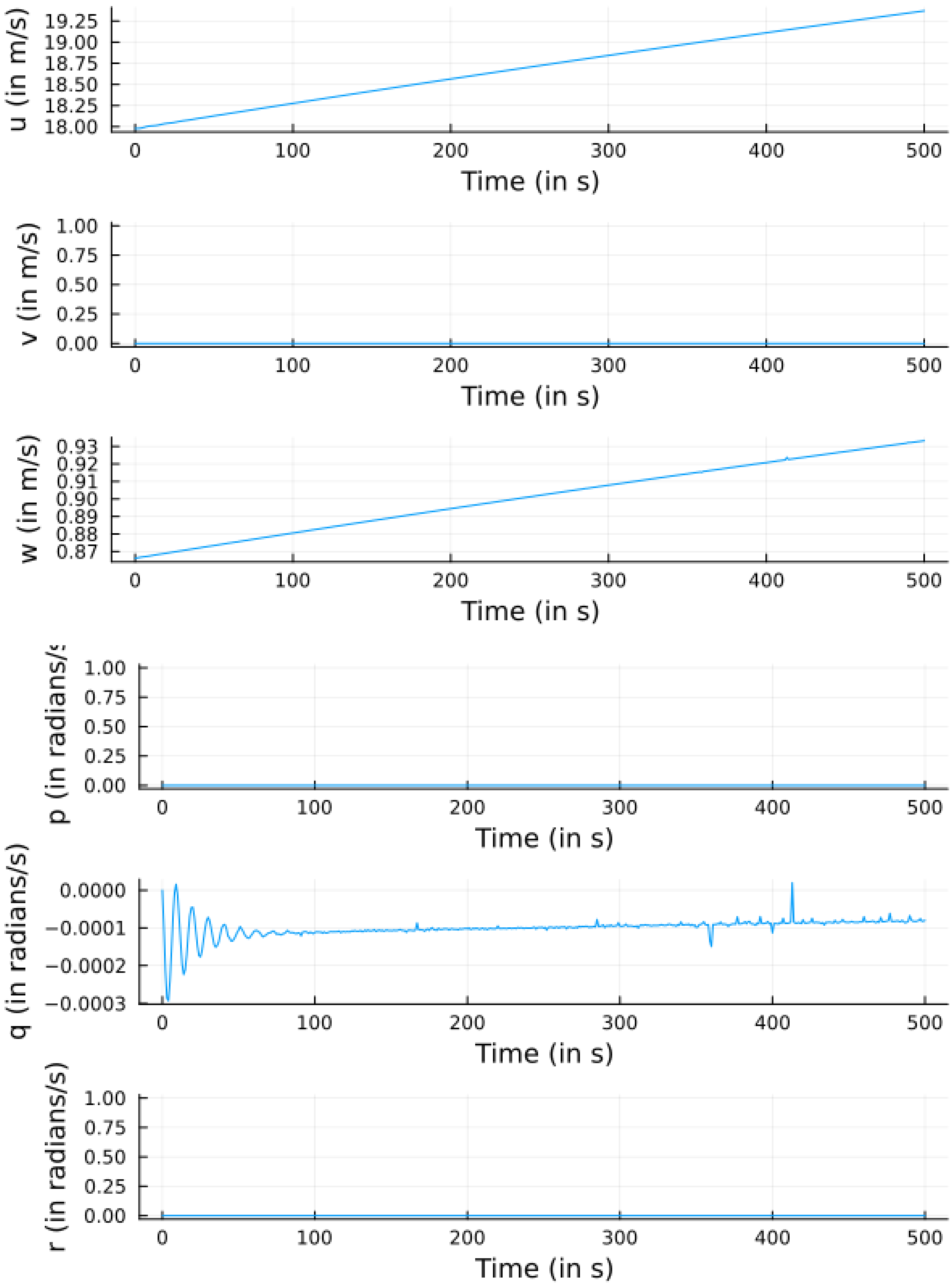
0.050201252250689686
-0.5021558030179929
0.17764553656958398

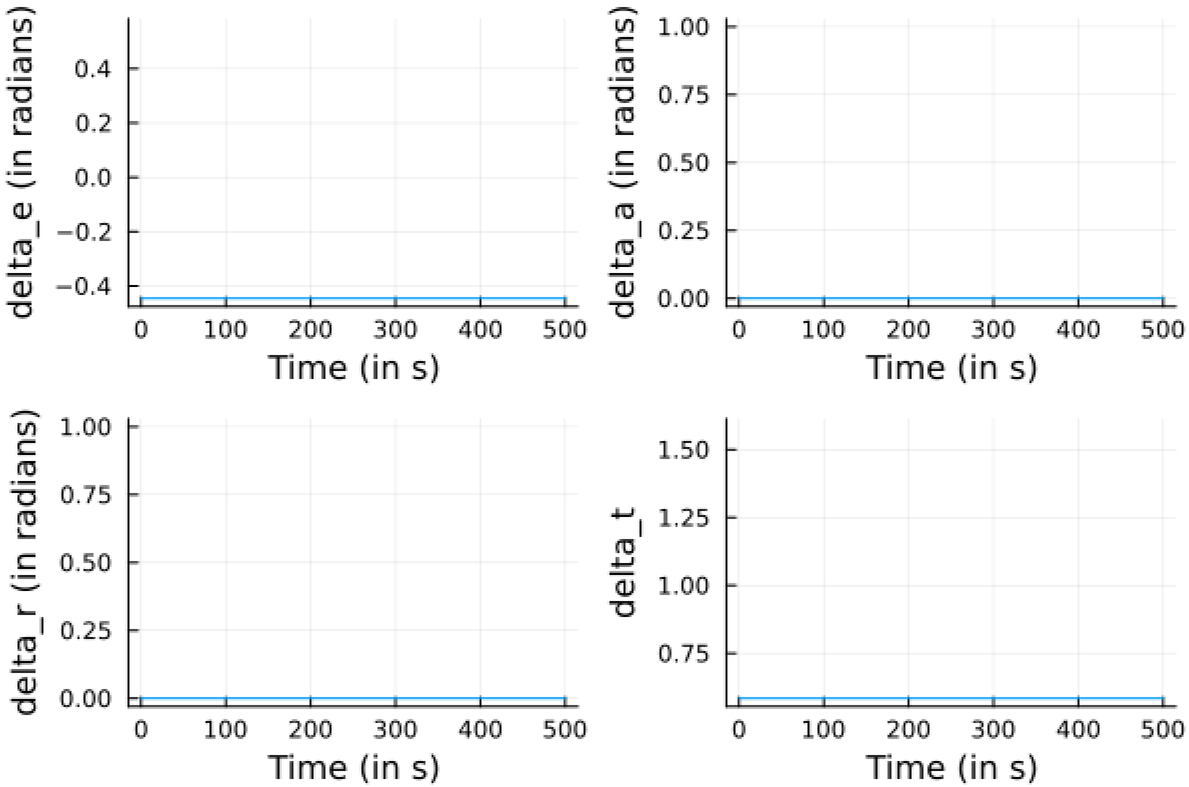
The velocity component in the start state is augmented with the inertial wind velocity expressed in the body frame to get the inertial aircraft velocity in the body frame. This is done because the trim condition is computed using the air-relative velocity vector, not the inertial aircraft velocity.

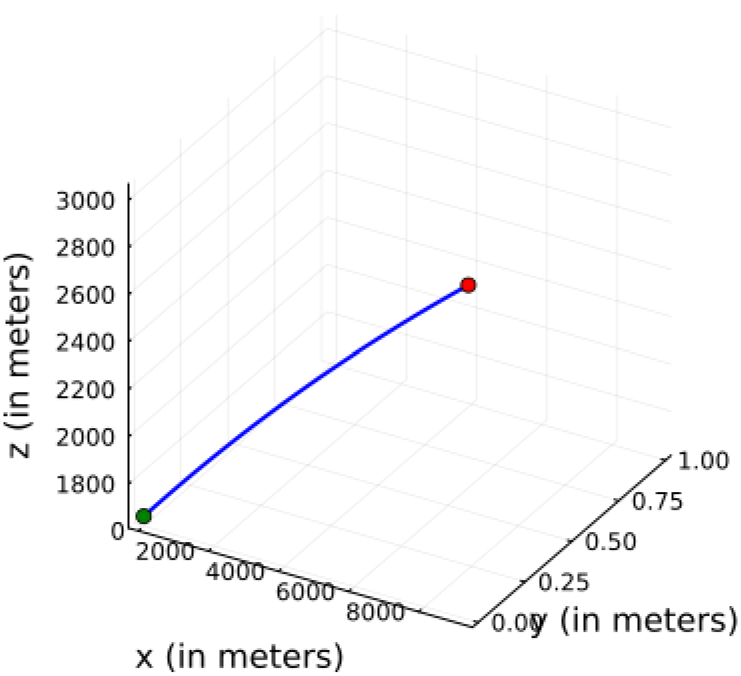
Problem 3.3



x (in m)
y1
Time (in s)
y (in m)
y1
Time (in s)
z (in m)
y1
Time (in s)
Roll (in radian
Time (in s)
Pitch (in radians)
Time (in s)
Yaw (in radians)
Time (in s)


u (in m/s)
19.25
19.00
18.75
18.50
18.25
18.00
0
100
200
300
400
500
Time (in s)
v (in m/s)
1.00
0.75
0.50
0.25
0.00
Time (in s)
w (in m/s)
0.93
0.92
0.91
0.90
0.89
0.88
0.87
Time (in s)
p (in radians/s
Time (in s)
q (in radians/s)
0.0000
−0.0001
−0.0002
−0.0003
Time (in s)
r (in radians/s)
Time (in s)


delta_e (in radians)
0.4
0.2
0.0
−0.2
−0.4
0
100
200
300
400
500
Time (in s)
delta_a (in radians)
1.00
0.75
0.50
0.25
0.00
0
100
200
300
400
500
Time (in s)
delta_r (in radians)
1.00
0.75
0.50
0.25
0.00
0
100
200
300
400
500
Time (in s)
delta_t
1.50
1.25
1.00
0.75
0
100
200
300
400
500
Time (in s)


z (in meters)
3000
2800
2600
2400
2200
2000
1800
0
2000
4000
6000
8000
x (in meters)
1.00
0.75
0.50
0.25
0.00
y (in meters)

Starting (Trim) State:

0.0
0.0
-1655.0
0.0
0.2226670348482986
0.0
17.979151993077068
0.0
0.8660794477603566
0.0
0.0
0.0

Trim Controls:

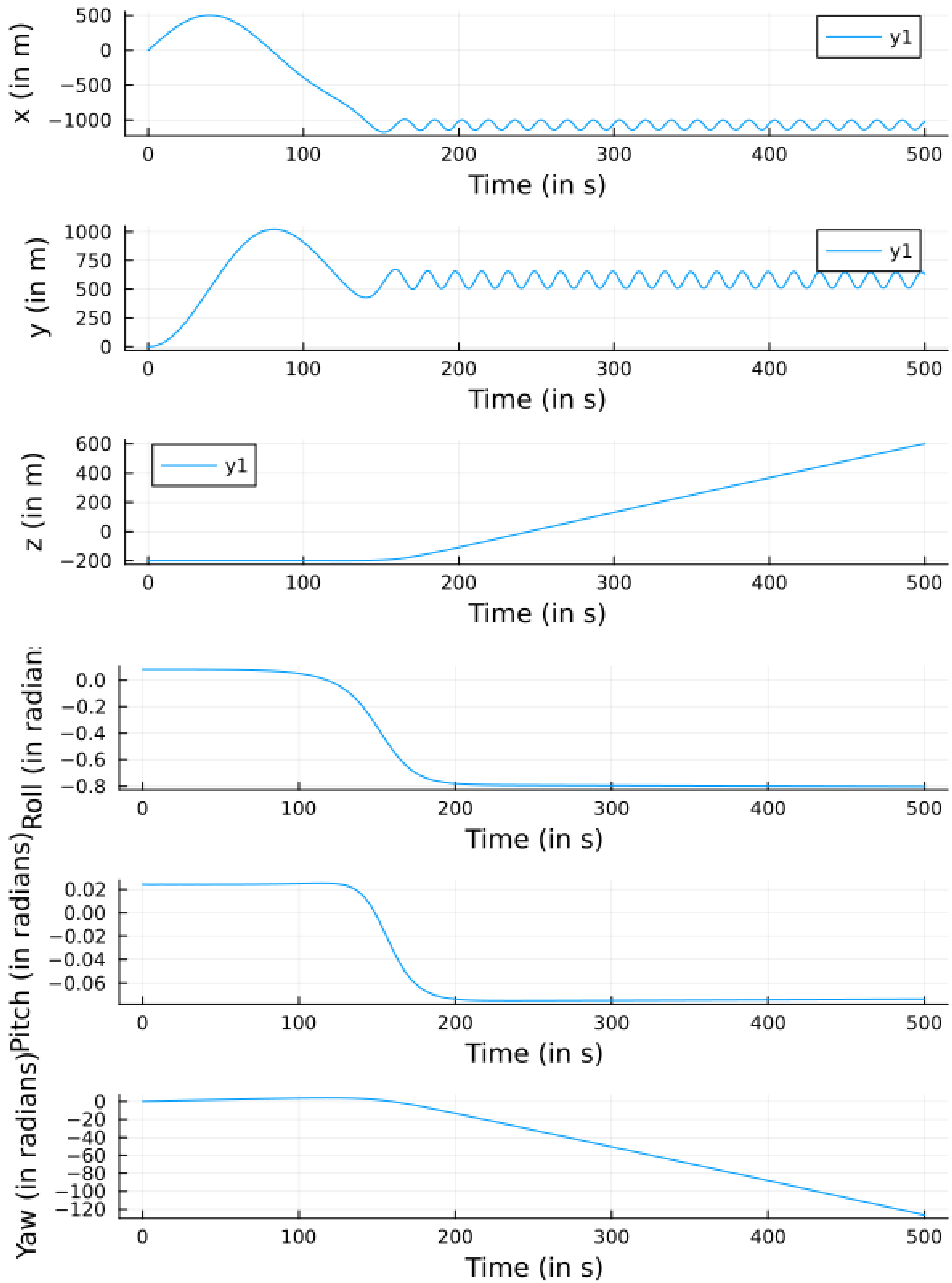
-0.44586027498230835
0.0
0.0
0.5838151089543826

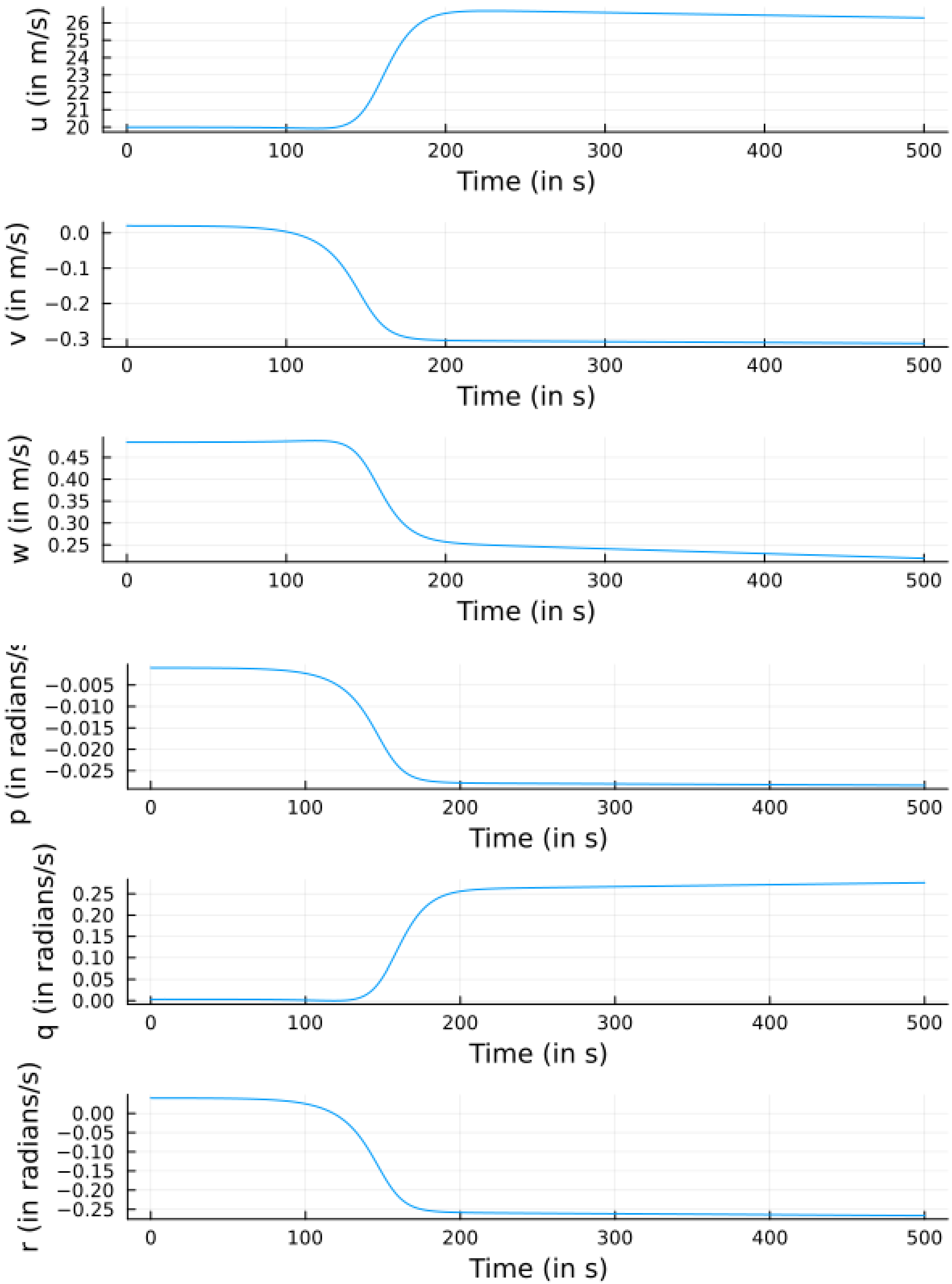
Trim Variables:

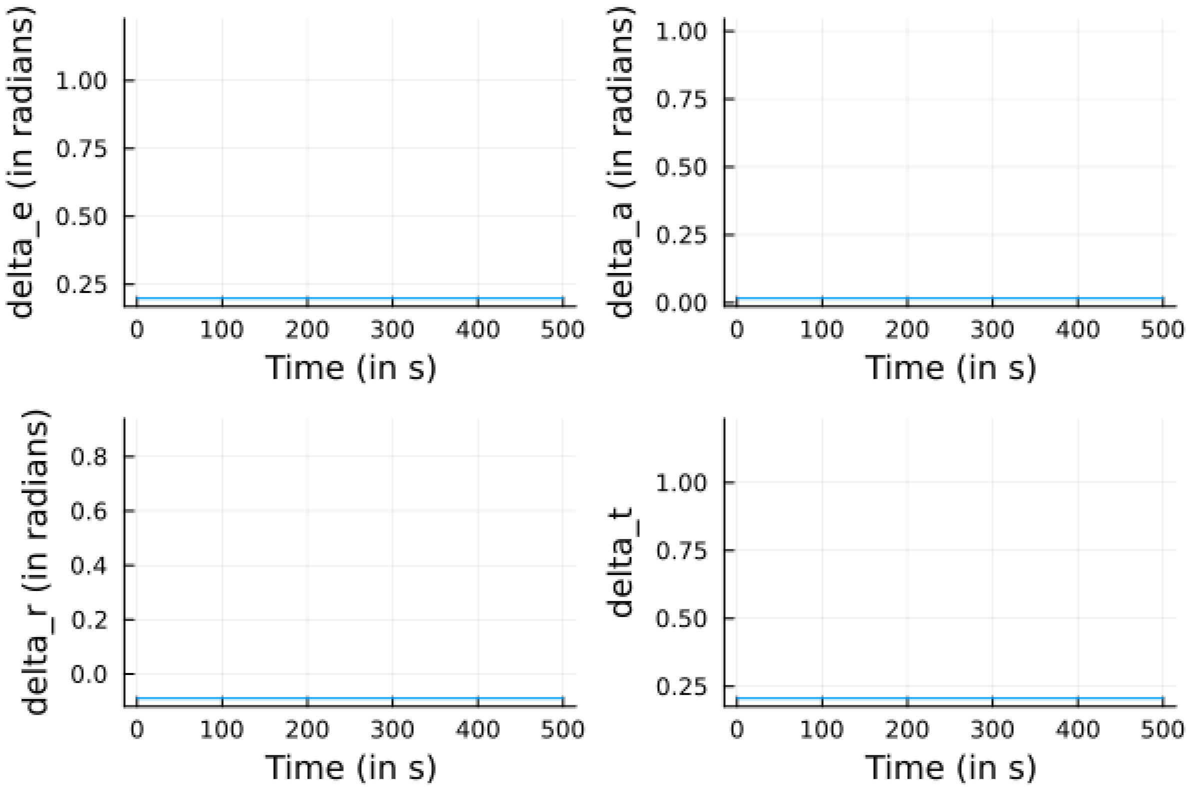
0.048134109648865664
-0.44586027498230835
0.5838151089543826

The aircraft initially flies upward, but as it gains height, the density changes and the aircraft is no longer in trim for the given control inputs. Consequently, the pitch starts to decrease, and the aircraft will eventually settle into a new straight, wing-level, horizontal flight condition.

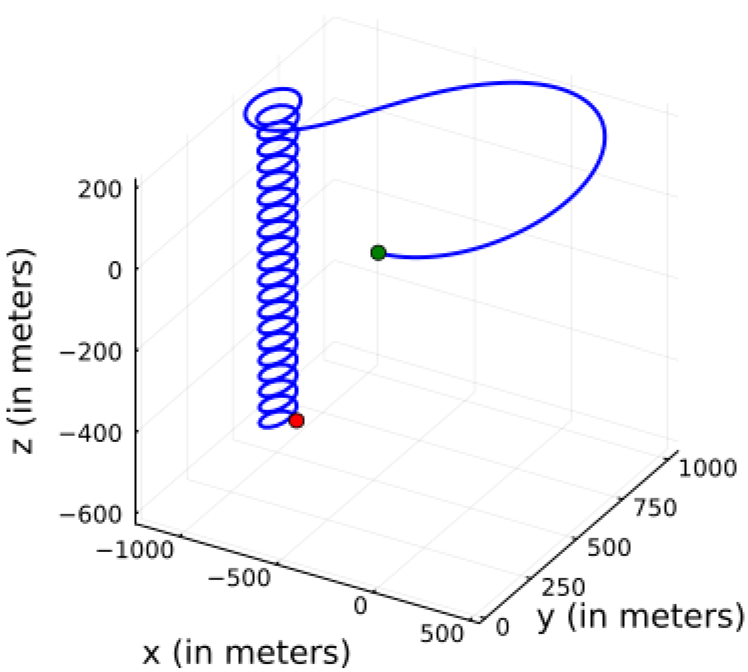
Problem 3.4


x (in m)
y1
Time (in s)
y (in m)
y1
Time (in s)
z (in m)
y1
Time (in s)
Roll (in radian
Time (in s)
Pitch (in radians)
Time (in s)
Yaw (in radians)
Time (in s)


u (in m/s)
20
21
22
23
24
25
26
0
100
200
300
400
500
Time (in s)
v (in m/s)
0.0
−0.1
−0.2
−0.3
0
100
200
300
400
500
Time (in s)
w (in m/s)
0.45
0.40
0.35
0.30
0.25
0
100
200
300
400
500
Time (in s)
p (in radians/
−0.005
−0.010
−0.015
−0.020
−0.025
0
100
200
300
400
500
Time (in s)
q (in radians/s)
0.25
0.20
0.15
0.10
0.05
0.00
0
100
200
300
400
500
Time (in s)
r (in radians/s)
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
0
100
200
300
400
500
Time (in s)



delta_e (in radians)
1.00
0.75
0.50
0.25
0
100
200
300
400
500
Time (in s)
delta_a (in radians)
1.00
0.75
0.50
0.25
0.00
0
100
200
300
400
500
Time (in s)
delta_r (in radians)
0.8
0.6
0.4
0.2
0.0
0
100
200
300
400
500
Time (in s)
delta_t
1.00
0.75
0.50
0.25
0
100
200
300
400
500
Time (in s)



z (in meters)
200
0
−200
−400
−600
−1000
−500
0
500
x (in meters)
0
250
500
750
1000
y (in meters)

Starting (Trim) State:

0.0
0.0
-200.0
0.08139317666496125
0.02423488776132034
0.0
19.994116848132784
0.020143935528348887
0.48465006455598303
-0.0009693006207633428
0.003251178482991965
0.03985586901296117

Trim Controls:

0.19813076930125995
0.016071727204618393
-0.08999161995556897
0.2057215557228874

Trim Variables:

0.02423488776132034
0.19813076930125995
0.2057215557228874
0.08139317666496125
0.001007196946708536
0.016071727204618393
-0.08999161995556897

The aircraft initially enters a banked turn to achieve the desired radius, but since the lateral mode is unstable, it transitions into a spiral mode.