AERSP 424: Advanced Computer Programming

Homework 1 Spring 2023

Submission Instructions:

- Submit a .zip files containing your .cpp and/or .h files.
- Your code needs to be compilable.
- If you upload an updated submission, please remove the previous submission as only the final submission will be graded.
- Using comments to explain your code is mandatory.
- Submission deadline is at 11:59 PM on Friday 2/3/2023 (The late policy mentioned in the syllabus will be applied).
- Explicitly declare variables with a proper name (easy to understand) and datatype (reflect the real-world scenario if possible).

Question 1 (40 points): *Functions, Arithmetic, and Iterations*

Given simplified nonlinear dynamics of a multirotor (each one is scalar)

$$\dot{x} = u \cdot \cos(\theta)\cos(\psi) + v \cdot (\sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi)) + w \cdot (\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi))$$

$$\dot{y} = u \cdot \cos(\theta)\sin(\psi) + v \cdot (\sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi)) + w \cdot (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))$$

$$\dot{z} = -u \cdot \sin(\theta) + v \cdot \sin(\phi)cos(\theta) + w \cdot \cos(\phi)\cos(\theta)$$

$$\dot{u} = rv - qw - g \cdot \sin(\theta) + \frac{F_x}{m}$$

$$\dot{v} = pw - ru + g \cdot \sin(\phi)\cos(\theta) + \frac{F_y}{m}$$

$$\dot{w} = qu - pv + g \cdot \cos(\phi)\cos(\theta) + \frac{F_z}{m}$$

$$\dot{\phi} = p + \left(q \cdot \sin(\phi) + r \cdot cos(\phi)\right) \cdot \tan(\theta)$$

$$\dot{\theta} = q \cdot \cos(\phi) - r \cdot \sin(\phi)$$

$$\dot{\psi} = (q \cdot \sin(\phi) + r \cdot \cos(\phi))/\cos(\theta)$$

$$\dot{p} = \frac{\left(I_y - I_z\right) \cdot qr + M_x}{I_x}$$

$$\dot{q} = \frac{(I_z - I_x) \cdot rp + M_y}{I_y}$$

$$\dot{r} = \frac{\left(I_x - I_y\right) \cdot pq + M_z}{I_z}$$

where x, y, z are positions in North-East-Down (NED) coordinate frame, u, v, w are velocities in body frame, ϕ, θ, ψ are vehicle orientation, p, q, r are body angular rates, F_x, F_y, F_z are forces, M_x, M_y, M_z are moments, m is mass, I_x, I_y, I_z are moments of inertia, and g is gravity.

- Write a function named *dynamics* to compute state derivatives based on the given equations above.
- In this function, pass state variables $(x, y, z, u, v, w, \phi, \theta, \psi, p, q, r)$, control variables $(F_x, F_y, F_z, M_x, M_y, M_z)$, and system parameters (m, I_x, I_y, I_z, g) by value, and pass state derivates by reference.
- Write another function named *integration* to implement the Euler integration method, using, i.e.,

  $$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \dot{\boldsymbol{x}}_k \cdot \Delta t$$

  where $\boldsymbol{x} = [x, y, z, u, v, w, \phi, \theta, \psi, p, q, r]^T$.
- In this function, pass state variables by reference, and pass state derivatives and the timestep by value.
- Perform a numerical integration from $t = 0$ to $t = 3$ by calling these two functions using a timestep, $\Delta t = 0.001$.
- Print out the final values of the states.

Note:

- Assume an initial condition of $x = 0, y = 0, z = -1, u = 0, v = 0, w = 0, \phi, = 0\ \theta = 0, \psi = 0, p = 0, q = 0, r = 0$.
- Apply the same control inputs $F_x = 0, F_y = 0, F_z = -mg, M_x = 1, M_y = 0, M_z = 0$ at every iteration.
- Use these values for system parameters $m =\ 1.5, I_x = 0.05, I_y = 0.05, I_z = 0.07, g = 9.81$.
- All units are in SI.
- The orientation (ϕ, θ, ψ) must be within $[-\pi, \pi)$ in every iteration.
- A negative value of any z-component implies upward direction.

Question 2 (30 points): *Pointers, and Bit Manipulation*

- Pick your favorite ***float*** number.
- Print its binary representation as a string or an array of characters.
- Replace the first eight bits of its mantissa with the inverse of its exponent bits.
- Print its new binary representation and decimal value.

Question 3 (30 points): *Variadic Template, and Function Overloading*

Write a C++ program that can print an ***arbitrary number*** (strictly positive) of arguments of ***any fundamental types*** (boolean, character, integer, floating-point, and string), ***including pointers*** of those fundamental types, solely by calling the same function.