THE MINISTRY OF SCIENCE AND HIGHER EDUCATION OF
THE RUSSIAN FEDERATION

ITMO University
(ITMO)

Faculty of Control Systems and Robotics

Simulation of Robotic Systems
task 2

Ali Ahmad
ISU 475789

1. General Lagrangian Equation

To write the equations of motion, we begin by deriving the Lagrangian for the system L as the difference between the kinetic and potential energy of the system:

$$L(x,\dot{x})=K(x,\dot{x})-P(x)$$

The Kinetic Energy for our system:

$$K(x,\dot{x})=\frac{1}{2}m(\dot{x})^2$$

The potential energy of our system:

$$P(x)=\frac{1}{2}k(x-x_0)^2$$

Where k is the stiffness coefficient of the spring.

Thus the Lagrangian for the system L:

$$L(x,\dot{x})=\frac{1}{2}\cdot m\cdot(\dot{x})^2-\frac{1}{2}\cdot k\cdot(x-x_0)^2$$

Using Lagrange's equation:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)-\frac{\partial L}{\partial x}=Q$$

where Q is the external force acting on a system, which is a damping force

in our case $Q=-b\dot{x}$

The partial derivative of L in respect to x :

$$\frac{\partial L}{\partial x}=-k\cdot(x-x_0)$$

The partial derivative of L in respect to $\dot{x}$:

$$\frac{\partial L}{\partial \dot{x}}=m\cdot\dot{x}$$

then:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)=m\left(\frac{d^2x}{d^2t}\right)$$

Finally, the equations of motion for our system:

$$m\cdot\left(\frac{d^2x}{d^2t}\right)+k\cdot(x-x_0)=-b\cdot\left(\frac{dx}{dt}\right) \text{ Or } \left(\frac{d^2x}{d^2t}\right)=\frac{-k}{m}\cdot(x-x_0)-\frac{b}{m}\cdot\left(\frac{dx}{dt}\right)$$

2.Analytival Solution:

Let us consider the second order differential equation

$$m\cdot\left(\frac{d^2x}{d^2t}\right)+k\cdot(x-x_0)=-b\cdot\left(\frac{dx}{dt}\right)$$

which can be written as:

$$m\cdot x\left(\frac{d^2x}{d^2t}\right)+b\cdot\left(\frac{dx}{dt}\right)+k\cdot(x-x_0)=0$$

The characteristic equation:

$$mr^2+br+k=0$$

he roots ($r_{1,2}$) are found using the quadratic formula

$$r_{1,2}=\frac{-b\pm\sqrt{(b^2-4m.k)}}{2.m}$$

The discriminant calculated by the equation:

$$\Delta=b^2-4m.k$$

for the given values:

$$m=0.4\,,b=0.05\,,k=19.6\,,x_0=0.72$$

The general solution to the differential equation,

$$0.4\left(\frac{d^2x}{d^2t}\right)+0.05\left(\frac{dx}{dt}\right)+19.6x=0\,,$$

is the sum of the homogeneous solutions and the particular solution.

The homogeneous solutions have the form $e^{\lambda t}$ and solve the homogeneous differential equation:

$$0.4\left(\frac{d^2x}{d^2t}\right)+0.05\left(\frac{dx}{dt}\right)+19.6x=0\,,$$

Substituting $e^{\lambda t}$ into the homogeneous differential equation results in the following characteristic polynomial,

$$0.4\lambda^2+0.05\lambda+19.6=0.$$

The roots of the characteristic polynomial are a complex conjugate pair.

$$\lambda1=-0.0625+i6.9997$$
$$\lambda2=-0.0625-i6.9997$$

The homogeneous solution is:

$$x(t)=C_1 . e^{(-0.0625t)}\cos(6.9997t)+C_2 . e^{(-0.0625t)}\sin(6.9997t)$$

where C_1 andC_2 are constants.

For the initial conditions: $x(t_0)=0.72$, $\frac{dx}{dt}(t_0)=0$, and $t_0=0$

,

$$C_1=0.72$$

$$C_2=0.0064288$$

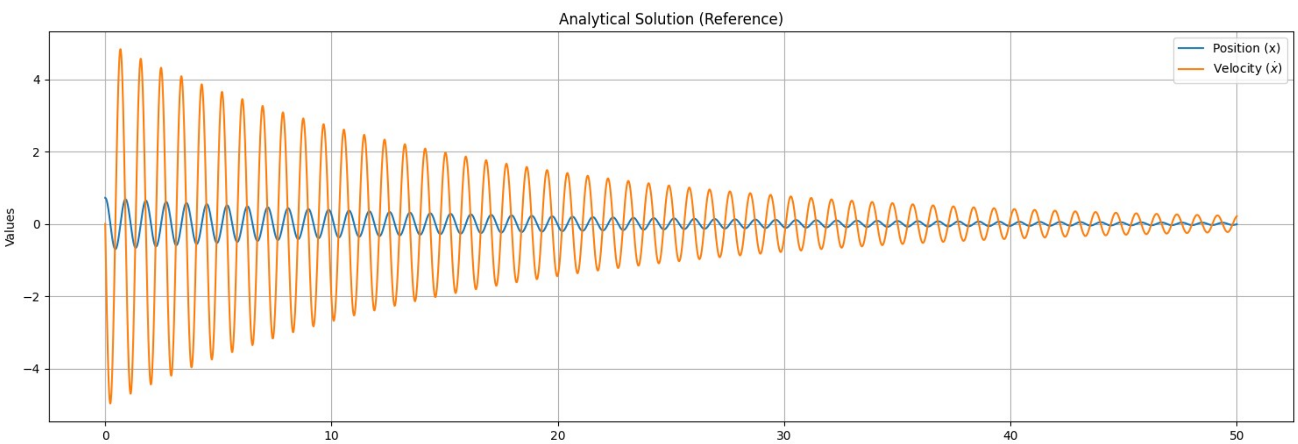
The particular solution of this differential equation is:

$$x_p=0$$

The total solution is:

$$x(t)=0.72 . e^{(-0.0625t)}\cos(6.9997t)+0.0064288 . e^{(-0.0625t)}\sin(6.9997t)$$

the result was shown as:



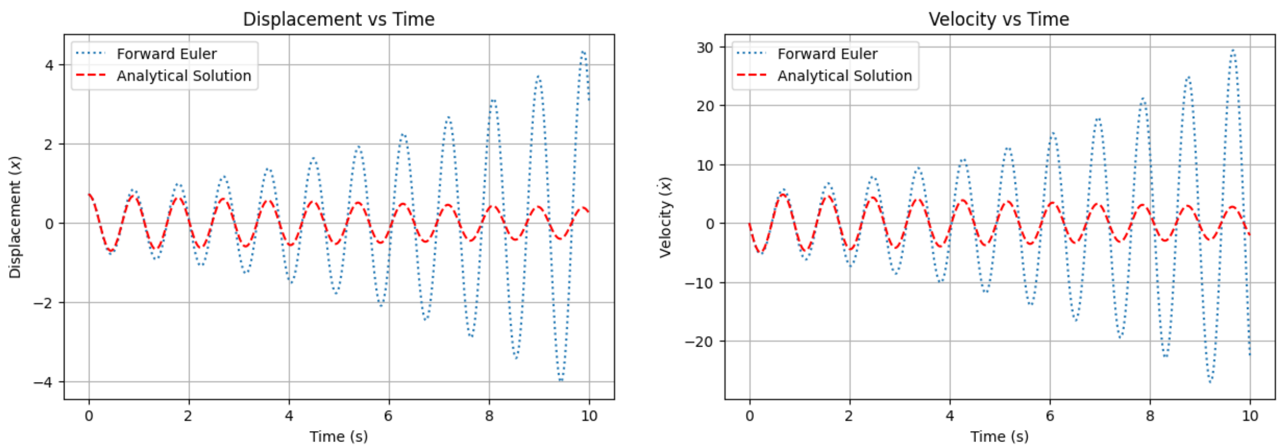
3.Simulation and discussion:

3.1 Explicit Euler

It approximates the solution at each step by using the derivative at the current point.

$$x_{k+1} = x_k + hf(x_k)$$

Here, h is the step size and the function $f(x_k)$represents the slope a x_k



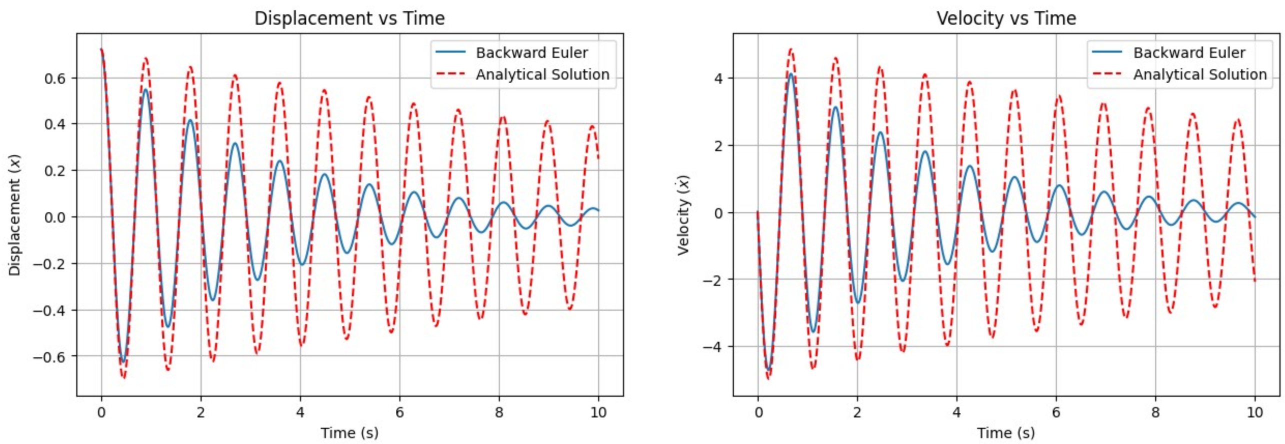
_As shown in Figure the system is unstable, with both displacement and velocity diverging over time. even if the system is stable and its state is decaying to zero (the equilibrium point). This is obvious in the case of the time step h = 0.01. Reducing the time step can help in the short term but the system will still blow up eventually and it is not suitable for high-accuracy applications because its global error is O(h) where h is the time step.

3.2 Implicit Euler

It requires solving an equation at each step and is defined by:

$$x_{k+1} = x_k + hf(x_{k+1})$$

Since $x_n + 1$ appears on both sides of the equation, an iterative root-finding method, such as Newton-Raphson, is often needed to solve for $x_n + 1$.



we notice different behavior. The system decays at a rate higher than the theoretical rate. This is due to the implicit nature of the Backward Euler method. Where we calculate the next step based on the derivative at the future point, rather than the current point. The artificial damping introduced by the Backward Euler method can be reduced by reducing the step size it is not suitable for high-accuracy applications because its global error is O(h) where h is the time step.

3.3 Runge-Kutta method of order 4

The Runge-Kutta method of order 4 (RK4) is widely used for its accuracy and stability. For a step size h, the RK4 update formula is:

$$y_{n+1}=y_n+\frac{h}{6}(k_1+2k_2+2k_3+k_4)$$

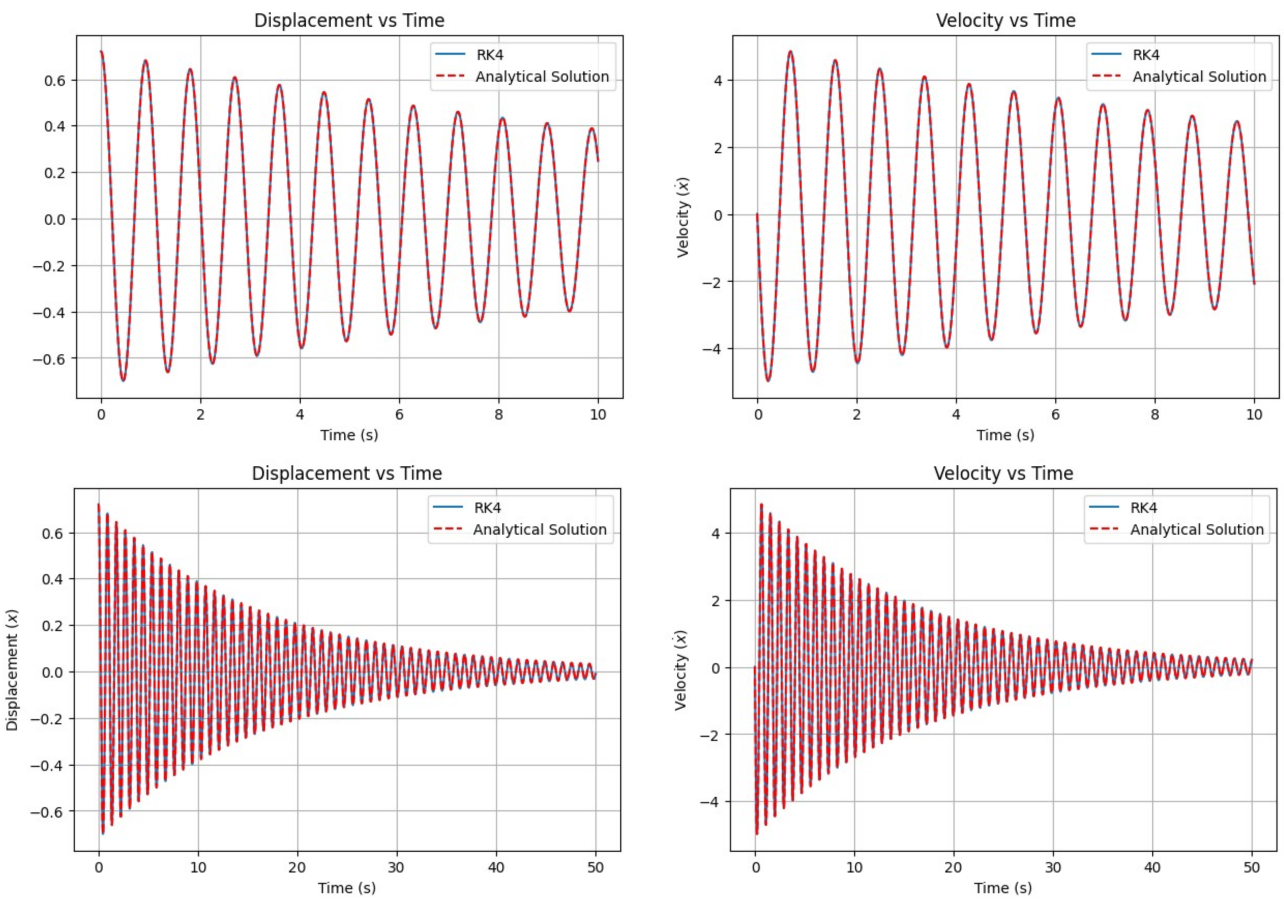
where the intermediate values are defined as:

$$k_1=f(t_n,y_n)$$

$$k_2=f\left(t_n+\frac{h}{2},y_n+\frac{h}{2}k_1\right)$$

$$k_2=f\left(t_n+\frac{h}{2},y_n+\frac{h}{2}k_2\right)$$

$$k_2=f(t_n+h,y_n+hk_3)$$



RK4 is the most efficient and stable method with global error $O(h^4)$, making it highly accurate for relatively larger step sizes.RK4 is More complex than Euler methods, as it involves multiple stages, so it is computationally expensive.

4 Comparison of Integration Methods

Accuracy

- **Explicit Euler** method is the least accurate of the three, a first-order method and the error grows significantly with larger time steps.
- **Implicit Euler** method is also first-order but its offers better overall stability and slightly improved accuracy compared to its explicit counterpart.
- **RK4** method is the most accurate. As a fourth-order method, it requires four derivative evaluations per step, allowing it to closely follow the exact solution with minimal deviation, making it the preferred method when high precision is required.

Stability

- **Explicit Euler** suffers from poor stability, especially when solving stiff problems or when using large time steps, leading to divergence and unusable results.
- **Implicit Euler** method, generally more stable than explicit methods , allowing it to handle stiff problems and much larger time steps without diverging. This is a huge advantage despite its higher step cost.
- **RK4** provides very good stability for a wide range of problems.

Cost

- **Explicit Euler** has the lowest computational cost per step, as it only requires a single straight forward evaluation of the derivative function.
- **Implicit Euler** has a higher per-step cost because it requires solving a potentially non linear equation at every step,
- **RK4** method , it requires four derivative evaluations per step, making it more expensive than Explicit Euler,