MECHANICAL VIBRATIONS

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY

REPORT

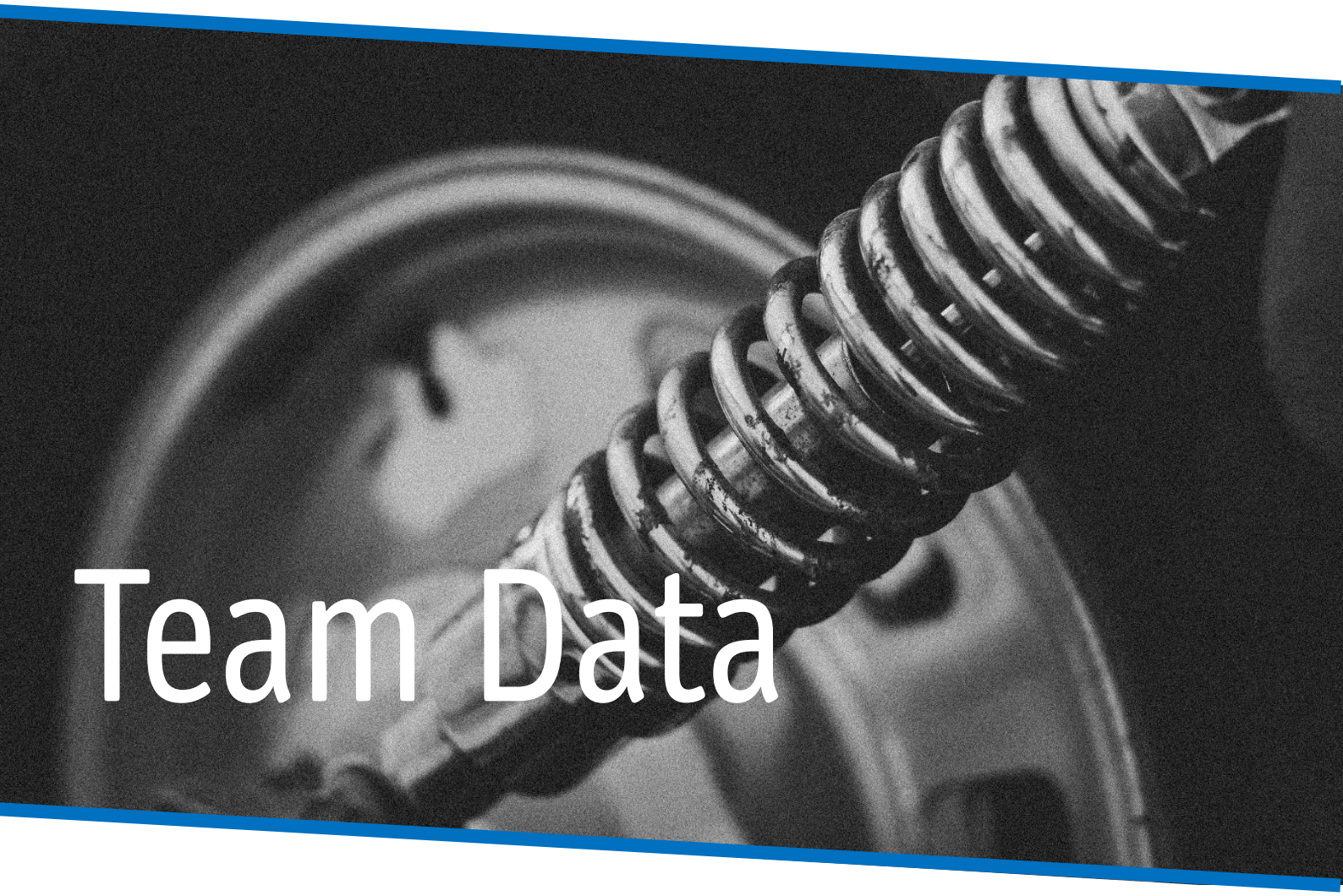
Teamwork

Under Supervision of

Dr. Basiony EL-Souhaily

Dr. Osama Mokhemer

MECHANCAL DESIGN DIVISION | MECHANICAL ENGINEERING



Team Data

Ayman Muhammed Hussien El-sayed Muhammed	21010334
Ahmed Samy Elsaiid Mohamed Ahmed Elshimy	21010085
Ahmed Saiid Abdelnaby Abdelrahman	21010090
Marwan Mohamed Mohamed Ibrahim Saad	19016618
Mohamed Alaa Hassan Mabrouk	21011192
Mohamed Ashraf Ali Mohamed	21011086
Mahmoud Mohamed Abdallah Elnahas	21011285
Ziad Ashraf Mohamed Ali	21010561
Ahmed Mohamed Refaat Bassiouny	21010158
Ahmed Mohamed Ahmed Saleh	21010144

Rotating Unbalance Project

Abstract

Mechanical vibration plays a critical role in the design, maintenance, and performance of rotating machinery. This report investigates the phenomenon of rotating unbalance, a primary source of mechanical vibration in rotating systems. The experimental project conducted focuses on analyzing the dynamics of rotating unbalance and its effects on system stability and performance. Using experimental setups, the study quantifies the relationship between rotational speed, unbalance magnitude, and vibration amplitude. The findings underscore the importance of identifying and mitigating rotating unbalance to improve the reliability of mechanical systems. Practical insights and recommendations are provided to aid in diagnosing and reducing vibration in real-world applications.

Rotating Unbalance, Harmonic, Frequency, dynamics

Introduction

Rotating machinery is ubiquitous in industries ranging from automotive to aerospace and manufacturing. Despite its essential role, such machinery is often prone to mechanical vibrations, which can lead to increased wear, fatigue, and eventual failure of components. Among the various causes of mechanical vibration, rotating unbalance is one of the most common and critical issues.

Rotating unbalance occurs when the mass distribution of a rotating component is not uniform around its axis of rotation. This imbalance generates centrifugal forces that induce vibrations in the system, particularly at high rotational speeds. Left unaddressed, these vibrations can compromise the efficiency and safety of the equipment.

The subsequent sections of this report detail the experimental setup, data collection methods, analysis, and findings. The report concludes with a discussion of the implications of rotating unbalance and recommendations for practical applications.

The phenomenon of mechanical vibrations often poses significant challenges in engineering systems, especially in rotating machinery such as pumps. These vibrations, arising from periodic excitation forces like **unbalanced masses**, can lead to performance inefficiencies, structural damage, and even system failure. To mitigate these issues, dynamic vibration

absorbers are employed. These devices are carefully designed to counteract the vibrations by introducing a secondary mass-spring system tuned to the excitation frequency.

This project focuses on the simulation and analysis of a vibrating pump mounted on a hinged-hinged beam with a square cross-section. The pump experiences periodic excitation forces due to its rotating unbalance. **A dynamic vibration absorber is attached beneath the pump to mitigate these vibrations**. By analyzing the vibration response of the system with and without the absorber, as well as using absorbers with varying masses, this study aims to understand the optimal conditions for vibration suppression. Simulink is employed to model the system and simulate its behavior under different conditions.

Design Requirements

Problem Understanding

- understanding of the vibrating pump and the effect of periodic excitation forces on the hinged-hinged beam.
- Study the working principle of dynamic absorbers and their role in mitigating vibrations.

Mathematical Modeling

- The first critical step in the project is to construct an accurate mathematical model of the system.
- The model should incorporate the primary mass (m_1), the absorber mass (m_2),.the stiffness constants (k_1 and 〖k〗_2), and the vibration amplitudes (x_1 (t) and x_2 (t) as depicted in the provided schematic

Simulink Model Development

- Using the derived mathematical equations, develop a Simulink model to simulate the system's dynamic response.
- Validate the model by comparing theoretical predictions with simulation results.

Parameter Variation

- Analyze the effect of varying the absorber mass (m_2) to 50%, 100%, and 150% of the optimal absorber mass.
- Simulate the system response for each case and determine the absorber's effectiveness in reducing vibrations.

Practical Implementation

After achieving a reliable simulation model, plan for practical solutions or experimental validation.

Ensure the practicality of absorber design for real-world applications.

Performance Evaluation

- Evaluate the vibration suppression performance by comparing the system's response before and after the installation of the dynamic absorber.
- Optimize the system for minimal vibrations under periodic excitation

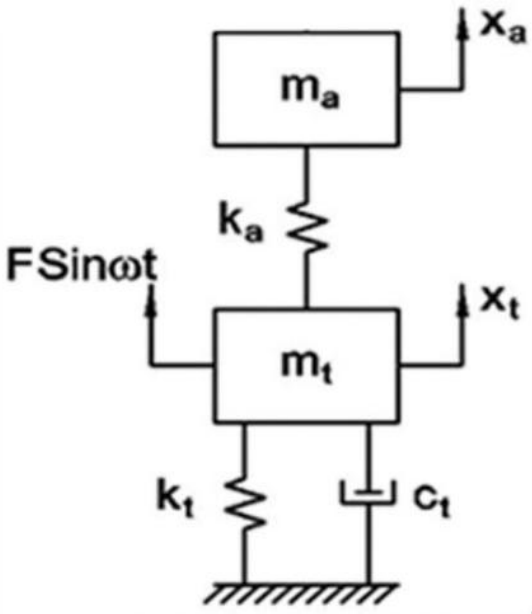
Mathematical Model

$$m_1\ddot{x_1} + c_1\dot{x_1} + k_1x_1 + k_2(x_1 - x_2) = Fsin(\omega t)$$

$$m_2\ddot{x_2} + k_2(x_2 - x_1) = 0$$

Where:

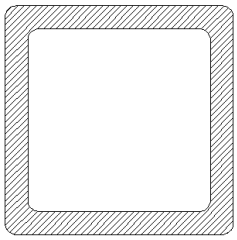
m_1 : mass of the system.
m_2 : mass of the absorber.
k_1 : stiffness of the system.
k_2 : stiffness of the absorber.
c_1 : damping of the system.
x_1 : displacement of the system.
x_2 : displacement of the absorber.
$\dot{x_1}$: velocity of the system.
$\ddot{x_1}$: acceleration of the system.
$\ddot{x_2}$: acceleration of the absorber.



Calculations

From design requirements the pump is vibrating on a beam and the cross section of the beam is square. So, we chose the material to be AISI 304 with cross section $1cm \times 1\ cm$ and a thickness of $1\ mm$

Beam properties



- $E = 200\ GPA$
- $Sy = 205\ GPA$
- $Density = 8000\ kg/m^3$

Beam stiffness calculations

$$k_{beam} = \frac{48EI}{L^3}$$

$$I = \frac{b^4 - (b-2t)^4}{12}$$

Where

- **L** is the beam length
- **b** is the square beam side length
- **E** is the modulus of elasticity
- **I** is the second moment of inertia
- **t** is the thickness of the square beam
- The beam material is Stainless steel AISI 304 with elastic modulus of 193GPA

$$I = \frac{0.01^4 - (0.01 - 2 * 0.001)^4}{12} = 4.95 * 10^{-10}$$

$$k_{beam} = \frac{48 * 200 * 10^9 * 4.95 * 10^{-10}}{0.5^3} = 37.78\ kN/m$$

Absorber stiffness calculations

Mass of the system $(m_1) = 550\ gram$

Natural frequency of the system $= \sqrt{\frac{k_{beam}}{m_1}} = \sqrt{\frac{37780}{0.55}} = 262.1\ rad/s$

- $\omega_s = 262.1\ rad/s\ =\ 2502.95\ RPM$
- $\omega = 2500\ RPM = 261.8\ rad/s$
- So, the system is approximately at resonance.
- To minimize the resonance ω_a should be equal to ω.
- To more safety $m_2 \approx \frac{m_1}{2} \approx 250\ gram$

$$k_2 = m_2 * \omega_a^2 = 0.25 * 261.8^2 = 17.13\ kN/m$$

$$k_2 = \frac{Gd}{8NC^3}$$

Where

- **C** is the spring index
- **d** is the coil wire diameter
- **N** is the number of active coils
- **G** is the modulus of rigidity

Assumptions

- $C = 5$
- $d = 2\ mm$

$$N = \frac{Gd}{8k_2C^3} = \frac{80 * 10^9 * 0.002}{8 * 17130 * 5^3} = 9.33 \approx 10$$

$$k_2 = \frac{Gd}{8NC^3} = \frac{80 * 10^9 * 0.002}{8 * 10 * 5^3} = 16\ kN/m$$

$$\omega_s = \sqrt{\frac{k_2}{m_2}} = \sqrt{\frac{16000}{0.25}} = 252.98\ rad/s = 2415.8\ RPM$$

Fabrication and Manufacturing

After the design was finalized and thoroughly reviewed multiple times to ensure all design requirements were successfully met, a bill of materials (BOM) for the entire assembly was created (as shown in table 1).

Table 1 | Bill of Material

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	wooden frame	Wood Beams (25mm*35mm)	1
2	775 Motor-775	12Volt 12000RPM	1
3	square beam of motor	AISI 304 Stainless Steel square beam extrusion (10mm*10mm 1mm thickness)	1
4	KP-000 Bore 10mm Bearings Pillow O	Bearing with a housing	2
5	motor fixation part	3mm Steel sheet (CNC Fiber-Laser)	1
6	Bearing - link linkage	3mm Steel sheet (CNC Fiber-Laser)	4
7	frame legs	Wood block	2
8	wood spacer	6mm Acrylic sheet (CNC CO2-Laser)	4
9	coupler	35 mm copper Cylinder (Lathe Turning)	1
10	Unbalanced mass	3mm Steel sheet (CNC Fiber-Laser)	1

The BOM was prepared to:

1. List the quantity of each required part.
2. Specify the various raw materials needed.
3. Outline the fabrication and manufacturing process for each part.

Raw material:

- 3mm Steel sheets (180mm*120mm)
- 6mm Acrylic Sheets (55mm*55mm)
- 10mm*10mm AISI 304 Square Extrusion 1mm thickness
- 35mm Copper Bars
- Wood Bars (25mm*35mm)

Figure 1 | Raw Material Used

Manufacturing process

CNC Fiber-Laser

All parts made from 3mm steel sheets were manufactured using a CNC Fiber-Laser machine capable of cutting 2D parts (Figure 2).

DXF files were prepared and sent to the workshop for processing (Figure 3).

Figure 2 | Fiber Laser CNC Machine

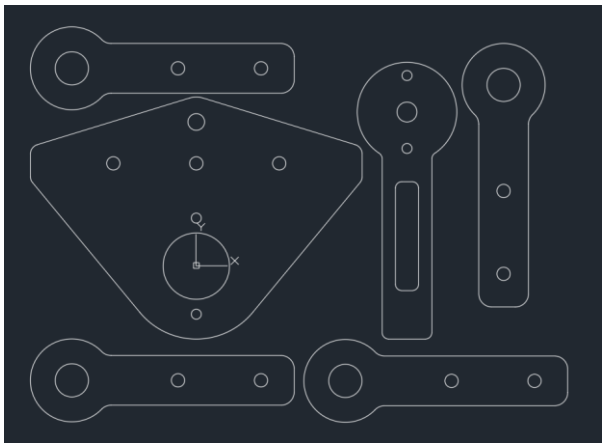


Figure 3 | DXF file

CNC CO2-laser

All parts made from 6mm Acrylic sheets were manufactured using a CNC CO2-Laser machine capable of cutting 2D parts (Figure 4).

Figure 4 | Fiber Laser CNC Machine

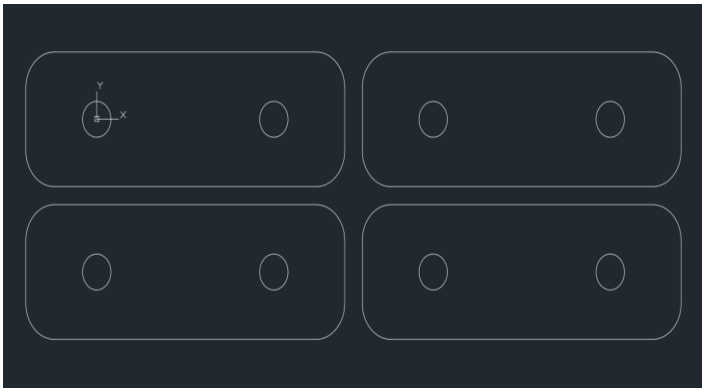


Figure 5 | DXF file

Lathe Turning

The motor coupler includes a special feature that can only be manufactured using a lathe turning machine. An engineering drawing with all the required dimensions was created (Figure 6).

Figure 6 | Turning machining

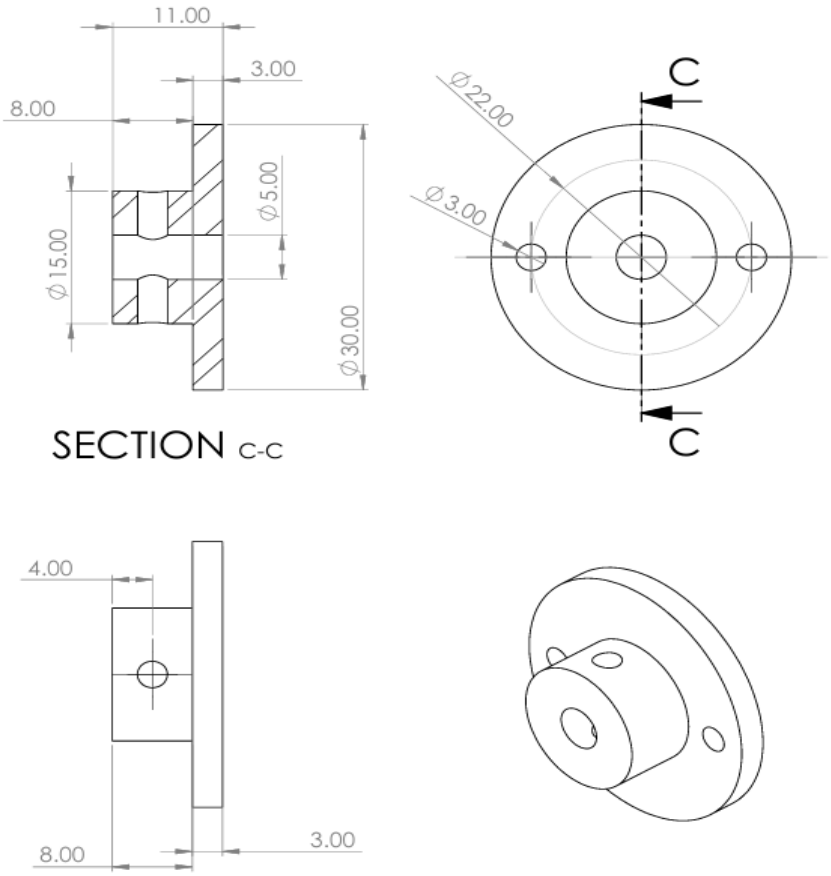


Figure 7 | Mechanical Drawing of motor coupling

Carpentry

The wooden Frame was made in a carpenter shop as well as the raw material

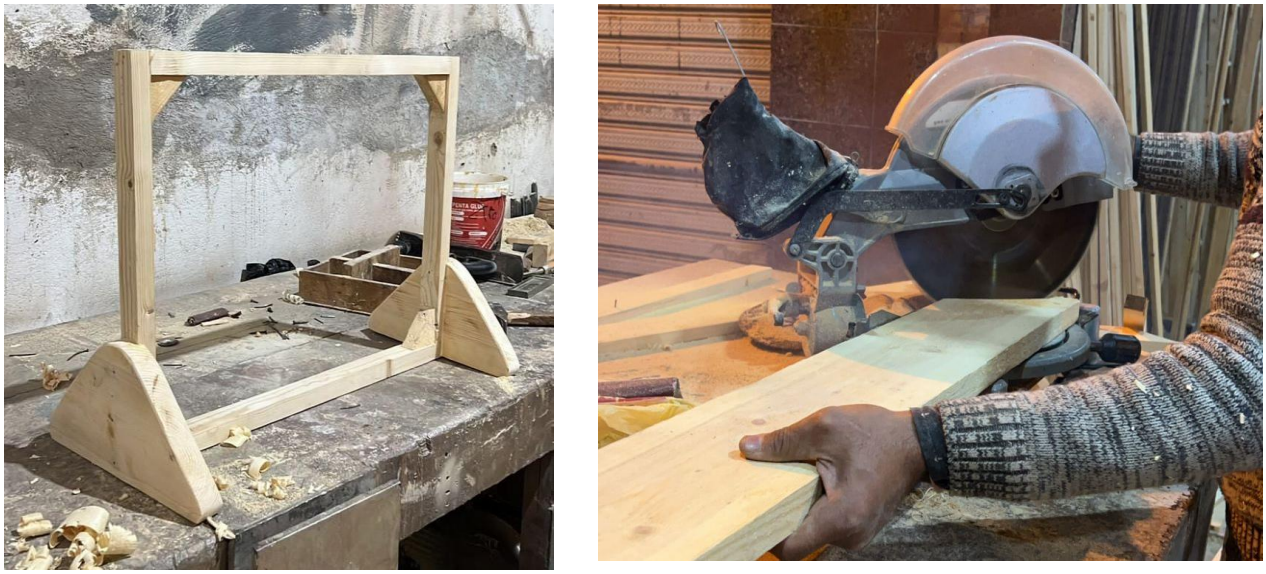
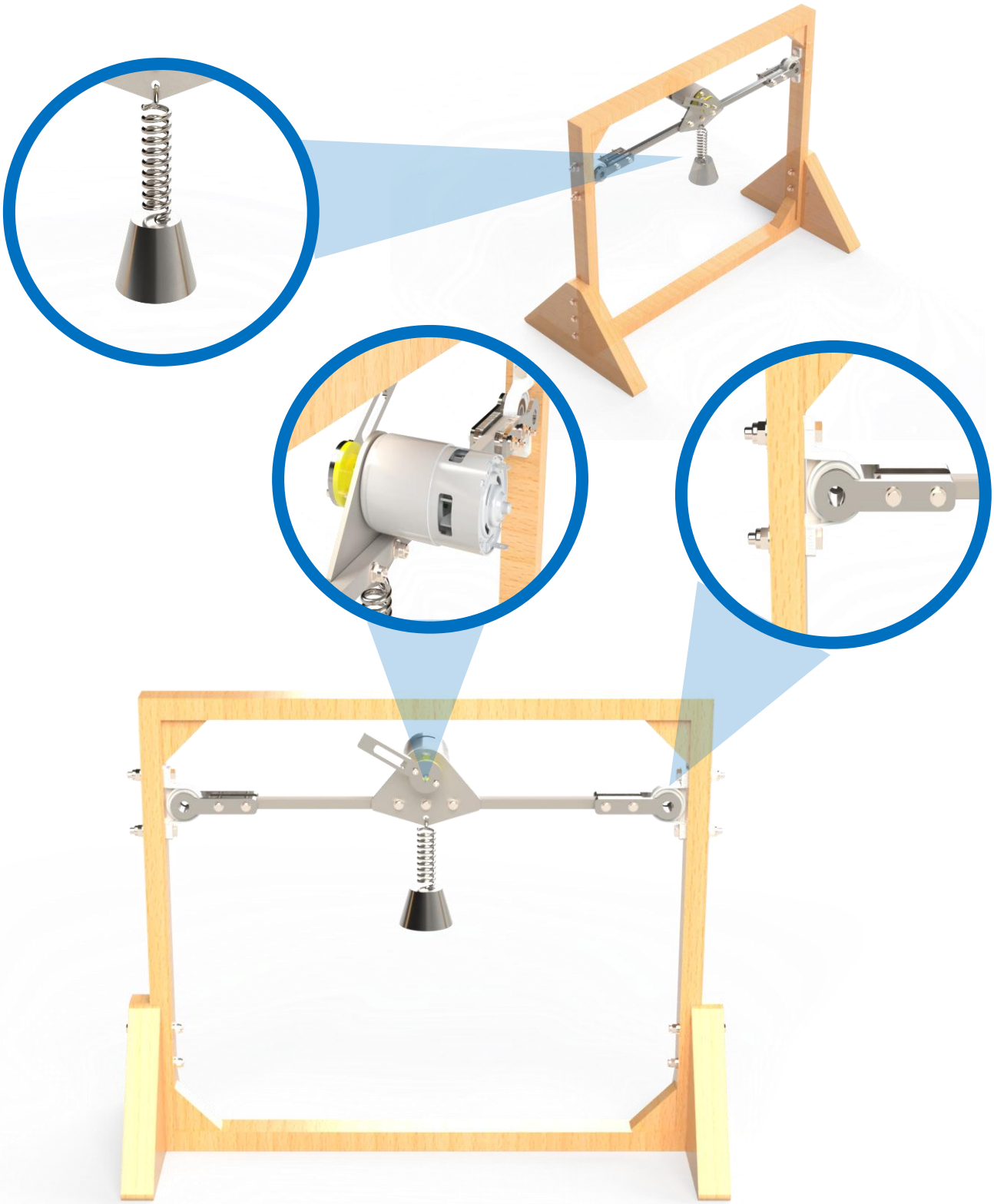


Figure 8 | Wooden Frame

Final Design



Controlling Motor's speed

Why controlling motor's speed

Prevent Overloading

- Running the motor at full speed continuously may draw excessive current, potentially overheating or damaging the motor.
- With velocity control, the motor operates within safe limits.

Energy Efficiency

- Full-speed operation consumes maximum energy, even if the application doesn't need it.
- Controlling velocity optimizes power usage and prolongs battery life.

Smooth Operation

- Direct connection to a battery causes sudden starts and stops, leading to mechanical stress on the motor and connected systems.
- PWM control provides smooth acceleration and deceleration.

Programmable Control

- With an Arduino, you can program the motor to operate in response to specific inputs or conditions (e.g., sensors, timers).
- This makes the system adaptable and intelligent, unlike a static direct connection to a battery.

Safety:

- Sudden or uncontrolled motor movements can be dangerous in systems with moving parts.
- Controlled velocity reduces the risk of accidents and allows for emergency stopping.

Protection of Components:

- A sudden surge of power when connecting directly to a battery can damage sensitive mechanical or electronic components.
- Motor drivers and PWM signals regulate power delivery, ensuring safe operation.

How to control the motor's speed:

Controlling Motor Speed Using Microcontroller:

Components Used:

- Arduino UNO
- Cytron Dual Channel 10A DC (5-30 V) MDD10A
- DC Motor
- Potentiometer
- Breadboard and Jumper Wires
- 12V Battery for Power Supply

Working Principle:

- The potentiometer is used as an analog input device to control the speed of the motor.
- The Arduino reads the voltage from the potentiometer and maps it to a PWM (Pulse Width Modulation) signal.
- This PWM signal is sent to the motor driver, which adjusts the motor's speed accordingly.

WHY USE THE MDD10A Instead of L298N?

- **Higher Current Capability:** Handles up to 10A continuous current per channel, suitable for high-power motors.
- **Wide Voltage Range:** Supports motors operating at 5V to 30V, making it versatile for various applications.
- **Efficient MOSFET-Based Design:** Generates less heat compared to the L298N's transistor-based design.
- **Simpler Connections:** Directly compatible with PWM signals from microcontrollers like Arduino.
- **Direction and Speed Control:** Provides precise motor control with separate inputs for speed (PWM) and direction.

Connection Details

3.1. Potentiometer to Arduino:

- **Purpose:** The potentiometer is used as an input to adjust the motor speed.
- **Connections:**
  - **One outer pin** of the potentiometer → **5V** on Arduino.
  - **Other outer pin** of the potentiometer → **GND** on Arduino.
  - **Middle pin (wiper)** of the potentiometer → **A0 (Analog Pin 0)** on Arduino.

3.2. Cytron MDD10A Motor Driver to Arduino:

- **Purpose:** The motor driver controls the motor's speed and direction using signals from the Arduino.
- **Connections for Motor Channel 1 (Single Motor Setup):**
  - **PWM1** on MDD10A → **Pin 10** (PWM-capable pin) on Arduino (for speed control).
  - **DIR1** on MDD10A → **Pin 9** on Arduino (for direction control).
- **GND** on MDD10A → **GND** on Arduino (common ground is essential for proper operation).

3.3. Cytron MDD10A to Motor:

- **Purpose:** The motor driver delivers power to the motor and regulates its operation.
- **Connections:**
  - **M1+** on MDD10A → **Positive terminal** of the DC motor.
  - **M1-** on MDD10A → **Negative terminal** of the DC motor.

3.4. Cytron MDD10A to Power Source (12V Battery)

- **Purpose:** Supplies power to the motor via the motor driver.
- **Connections:**
  - **VM (Voltage Motor)** on MDD10A → **Positive terminal** of the 12V battery.
  - **GND** on MDD10A → **Negative terminal** of the 12V battery.

3.5. Powering the Arduino

Use the Arduino's USB cable for power during testing (connect to a computer or USB power source).

Code Implementation

```
int in2 = 9;
int ConA = 10;               // Don't forget this is a PWM DI/DO
int speed1;



void setup() {
pinMode(9, OUTPUT);         //Direction Control
pinMode(10, OUTPUT);        //PWM
Serial.begin(9600);
delay(50);
}



void TurnMotorA(){                //We create a function which control the direction
and speed          //Switch between this HIGH and LOW to change direction
digitalWrite(in2, HIGH);
speed1 = analogRead(A1);
speed1 = map(speed1, 0, 1023, 0, 255);
Serial.println(speed1);
analogWrite(ConA,speed1);              // Then inject it to our motor
}


void loop() {
TurnMotorA();                          //one function that keeps looping you can add
another one with different direction or stop
}
```

Explanation of Code

5.1. Pin Setup:

- `int in2 = 9;`: Pin 9 controls the **direction** of the motor.
- `int ConA = 10;`: Pin 10 sends a **PWM signal** to control the motor's speed.
- `int speed1;`: Variable to store the motor's speed value (calculated from the potentiometer).

5.2. setup() Function:

- pinMode(9, OUTPUT);: Sets pin 9 as an output for direction control.
- pinMode(10, OUTPUT);: Sets pin 10 as an output for speed control (PWM).
- Serial.begin(9600);: Starts the Serial Monitor for debugging.
- delay(50);: Adds a small delay.

5.3. TurnMotorA() Function:

This function does the actual work of controlling the motor:

A. Set Motor Direction:

- digitalWrite(in2, HIGH);: Sets the motor's direction to forward.
- (To reverse direction, you can set this to LOW later.)

B. Read Potentiometer Value:

- speed1 = analogRead(A1);: Reads the position of the potentiometer from pin A1. The value ranges from **0 to 1023**.

C. Map Potentiometer Value to Motor Speed:

- speed1 = map(speed1, 0, 1023, 0, 255);: Converts the potentiometer's value (0-1023) to a PWM value (0-255). This makes the motor speed proportional to the potentiometer position.

D.Send Speed to Motor:

- analogWrite(ConA, speed1);: Sends the calculated PWM value to pin 10 to control the motor's speed.

E.Debugging:

- `Serial.println(speed1);`: Prints the motor speed to the Serial Monitor for monitoring.

5.4. loop() Function

- `TurnMotorA();`: Keeps calling the `TurnMotorA()` function in an infinite loop to continuously adjust the motor speed and direction based on the potentiometer's position.

How It Works

1. The potentiometer is read to determine the motor's speed.
2. The speed is mapped to a value the motor driver can use (0-255).
3. The motor direction is set to **forward** (by default in this code).
4. The speed is sent to the motor driver via PWM on pin 10.

Key Notes

- **Direction Control**: This code currently only supports **one direction** (`HIGH`). To change direction, modify `digitalWrite(in2, HIGH);` to `LOW` or create another function for reverse direction.
- **Speed Control**: The potentiometer adjusts the motor speed smoothly.
- **Serial Monitor**: You can monitor the speed values (0-255) in the Arduino Serial Monitor.

Conclusion

By controlling the motor's velocity with a driver and an Arduino, you gain precise control, efficiency, and versatility, while protecting the motor and other components. Directly connecting a motor to a 12V battery is only suitable for simple systems that don't require speed regulation, direction control, or programmable behavior.

Results

System parameters

- $Unbalance\ mass\ =\ 30\ gram$
- $eccentricity\ of\ unbalanced\ mass\ =\ 2\ cm$
- $velocity\ of\ motor\ =\ 2500\ RPM$
- $Mass\ of\ the\ system\ =\ 550\ gram$
- $Mass\ of\ the\ absorber\ =\ 250\ gram$

Assumptions

1. *The system has no damping*
2. *The system has damping coefficient of* 0.05

MATLAB Code

```
% System parameters
clc;clear,clf;
m = 0.03; % unbalanced mass
e = 0.02; % eccentricity of unbalanced mass
w =2500*2*pi/60 ; % angular velocity of motor (rad/sec)
F_amp = m*e*w^2 ; % Amplitude of the excitation force (N)
m1 = 0.55; % Mass of the pump (kg)
m2 = 0.125*2;  % Mass of the absorber (kg)
k1 = 38000; % Stiffness of the pump (N/m)
k2 = 17000;  % Stiffness of the absorber (N/m)
w_n = (k1/m1)^0.5;  % natural frequency of the system
w_a = (k2/m2)^0.5;  % natural frequency of the absorber
zeta=0.0;   % damping factor
c_1=zeta*2*m1*w_n;  % damping coeffecient


a=(k1+k2)/m1;b=-k2/m1;c=-k2/m2;d=k2/m2;e=a+d;f=a*d-b*c;

w_n1=((e-(e^2-4*f)^0.5)/2)^0.5;
w_n2=((e+(e^2-4*f)^0.5)/2)^0.5;

a1=(a-w_n1^2)/-b;
b1=(a-w_n2^2)/-b;

x_1=((k2-w^2*m2)*F_amp)/((k1+k2-w^2*m1)*(k2-w^2*m2)-k2^2);
x_2=(k2*F_amp)/((k1+k2-w^2*m1)*(k2-w^2*m2)-k2^2);
B2=((x_2-a1*x_1)*w)/((b1-a1)*w_n2);
B1=-(x_1*w+B2*w_n2)/w_n1;
r=w/(k1/m1)^0.5;
```

```
x_st=F_amp/k1;
x_o=(x_st)/((1-r^2)^2+(2*r*zeta)^2)^0.5;
if zeta==0
t=0:0.0001:6;

if r==0
      a.  x=x_st/2*w_n*t;
else
      a.  x=x_o*(sin(w*t)-r*sin(w_n*t));
end

x1=x_1*(sin(w*t))+B1*sin(w_n1*t)+B2*sin(w_n2*t);
x2=x_2*(sin(w*t))+a1*B1*sin(w_n1*t)+b1*B2*sin(w_n2*t);
figure(1)
plot(t,x)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Machine Response without absorber' );

figure(2)
plot(t,x1)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Machine Response with absorber' );

figure(3)
plot(t,x2)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Response of absorber' );

elseif zeta<0
t=0:0.0001:0.2;
fprintf("wrong value for zeta")

elseif zeta~=0
t=0:0.0001:0.2;
x=x_o*(sin(w*t));
x1=x_1*(sin(w*t));
x2=x_2*(sin(w*t));

figure(1)
plot(t,x)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Machine Response without absorber' );

figure(2)
plot(t,x1)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Machine Response with absorber' );

figure(3)
```

```
plot(t,x2)
grid on
xlabel('time (sec)');
ylabel('amplitude (m)');
title('Response of absorber' );
end

% Constants
ws = sqrt(k1 / m1); % Natural frequency of the main system (rad/s)

% Mass ratios to analyze
w_wa = linspace(0, 2, 100000); % Frequency ratio (w/wa)

% Set up the figure
% Parameters for each mass ratio
m2 = 0.125*2;          % Mass of absorber
k2 = 17000;                  % Stiffness of the absorber (N/m)
wa = sqrt(k2 / m2);       % Absorber natural frequency (rad/s)

% Machine response: X1/Xst
X1_Xst = abs((1 - (w_wa).^2) ./ ...
             i.  ((1 + k2/k1 - (w_wa * ws / wa).^2) .* ...
            ii.  (1 - (w_wa).^2) - k2/k1));
% Absorber response: X2/Xst
X2_Xst = abs(1 ./ ...
             i.  ((1 + k2/k1 - (w_wa * ws / wa).^2) .* ...
            ii.  (1 - (w_wa).^2) - k2/k1));

% Plot machine response

figure(4)
subplot(1, 2,1);
plot(w_wa, X1_Xst, 'r', 'LineWidth', 1.5);
xlabel('Frequency ratio (\omega / \omega_a)');
ylabel('abs(X_1 / X_{st})');
title('Machine Response' );
grid on;
      axis([0 2 0 20]);

      % Plot absorber response
      subplot(1, 2,2);
      plot(w_wa, X2_Xst, 'b', 'LineWidth', 1.5);
      xlabel('Frequency ratio (\omega / \omega_a)');
      ylabel('abs(X_2 / X_{st})');
      title('Absorber Response' );
      grid on;
      axis([0 2 0 20]);

      Reg_1=abs((w-w_n1)/w*100);
      Reg_2=abs((w-w_n2)/w*100);
      reg=[Reg_1  Reg_2];
      fprintf("\nThe natural frequencies is at least : \n +- %f %% from the
   operating speed\n",min(reg))
```

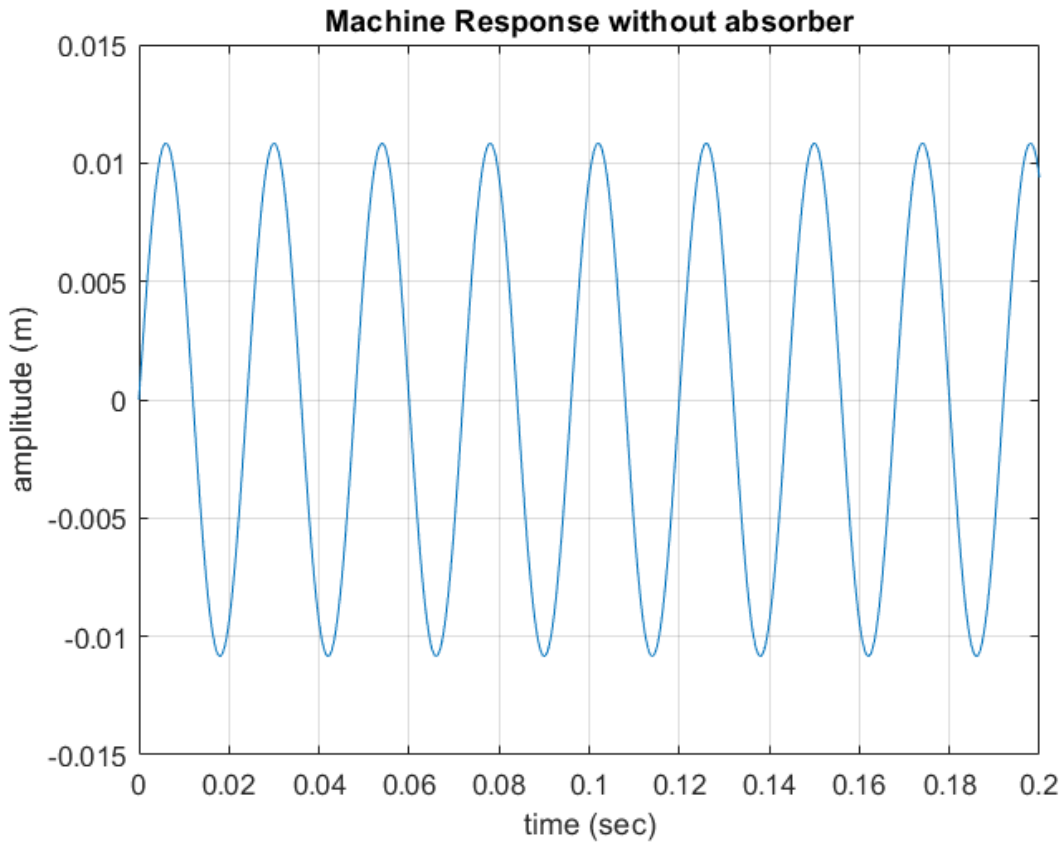
Explaining of code

- 1~14 : System parameters and basic calculations.
- 15~17 : calculations of natural frequencies of the two-degree system.
- 18~26 : calculating the amplitudes of the response before and after the absorber.
- 27~78 : if condition to plot the response in case of damped and not damped system.
- 79~108 : plotting the effect of the ratio of frequency of the motor to the natural frequency of the absorber on the amplitude of the system and the absorber.
- 109~112 : calculating of percentage of the speed range according to working speed.

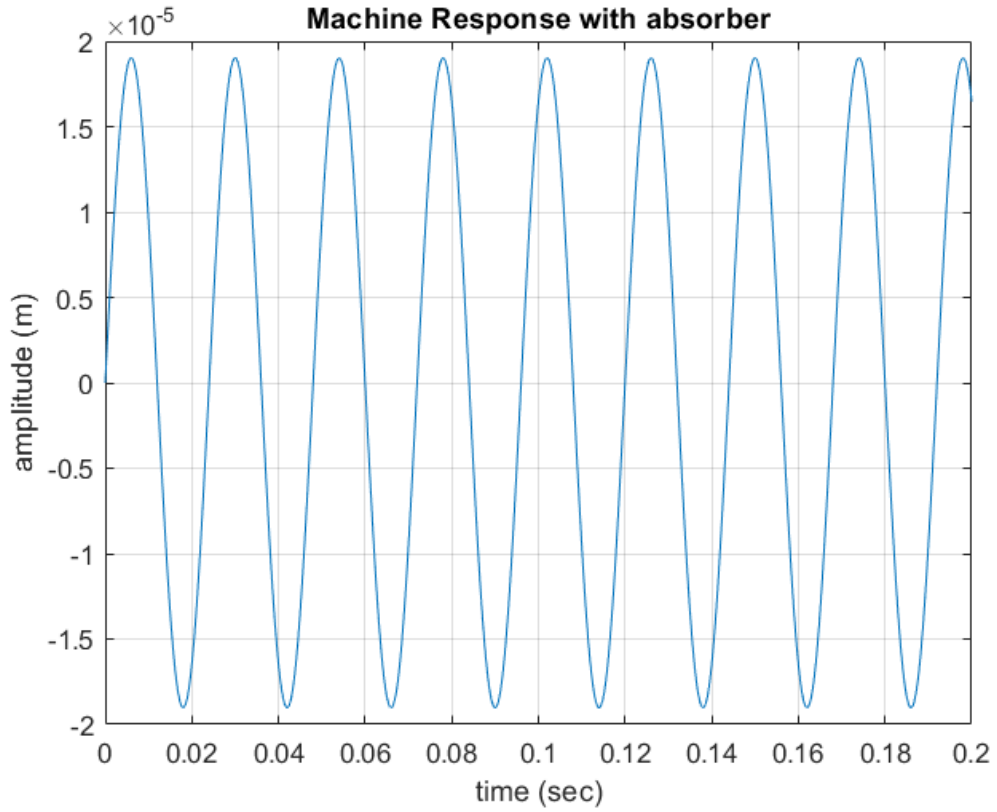
Results from MATLAB

1. Damped system

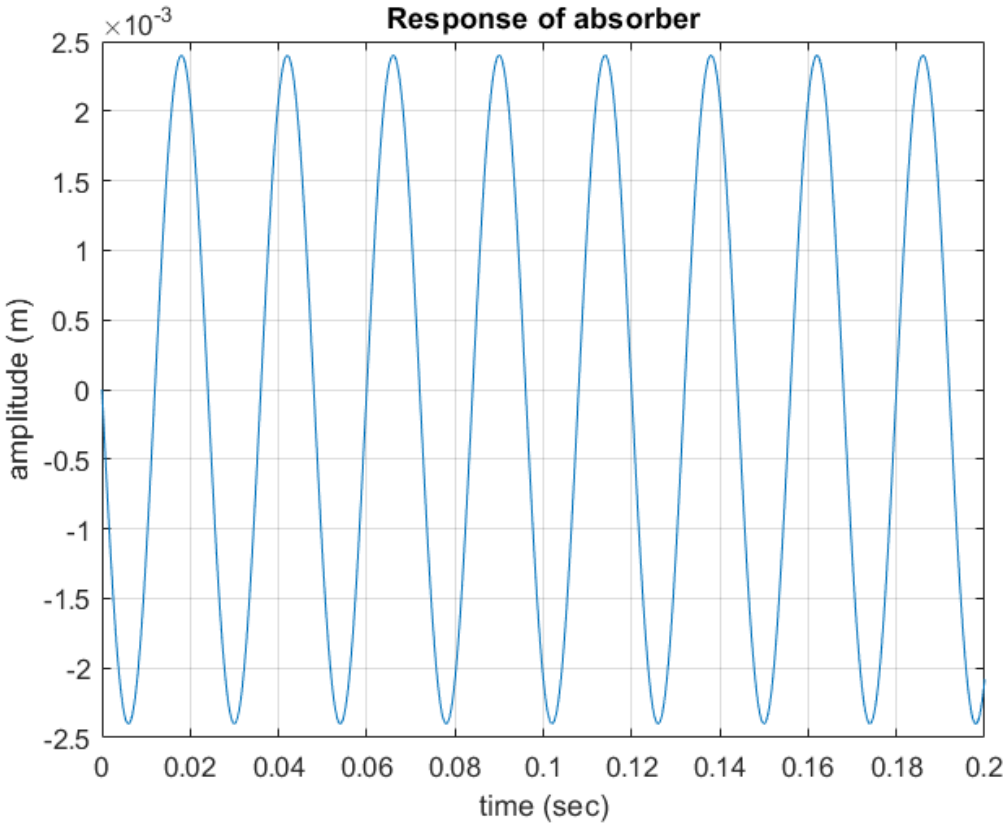
- Response of the system before adding the absorber:



- Response of the system after adding the absorber:

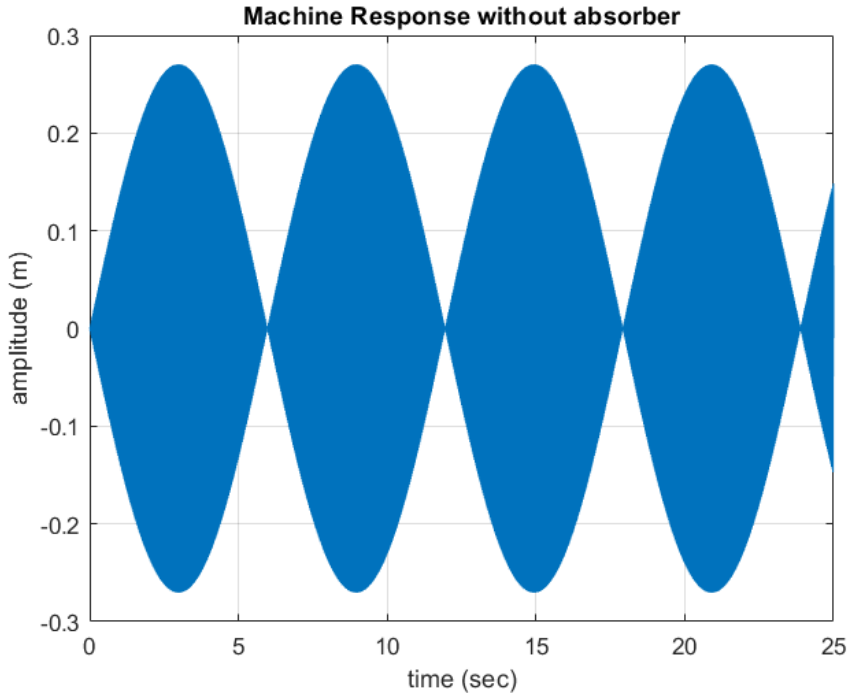


- Response of the absorber:

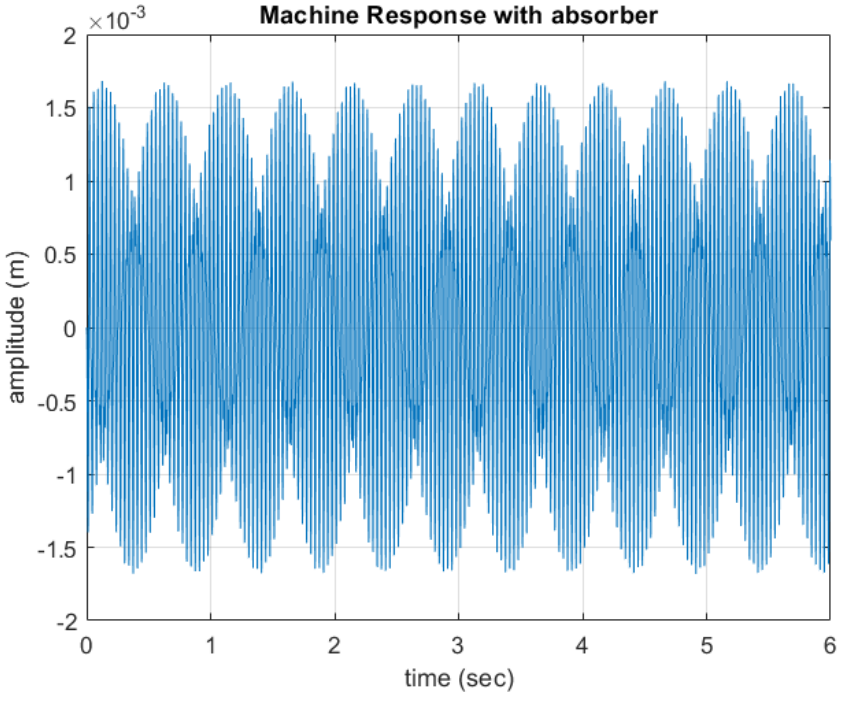


2. With no damping

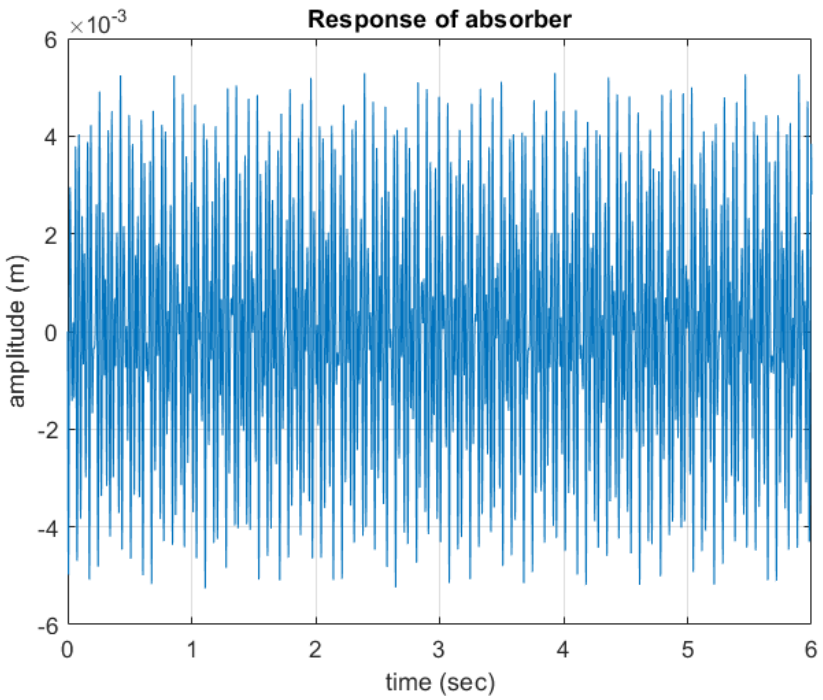
- Response of the system before adding the absorber:



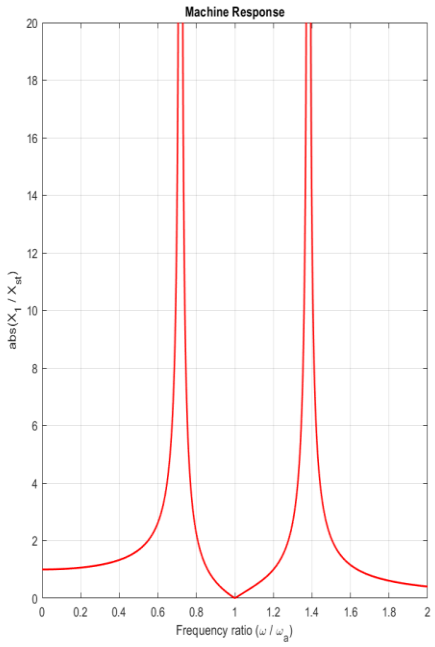
- Response of the system after adding the absorber:

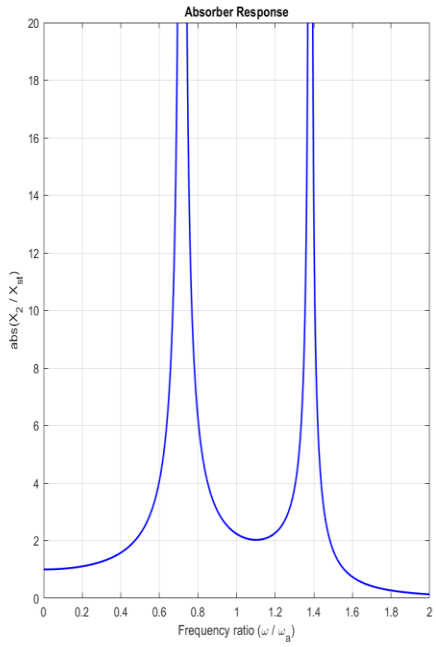


- Response of the absorber:



1. In extra we plotted the effect of the ratio of frequency of the motor to the natural frequency of the absorber on the amplitude of the system and the absorber.
($MATLAB\ code$ 79 ~108)

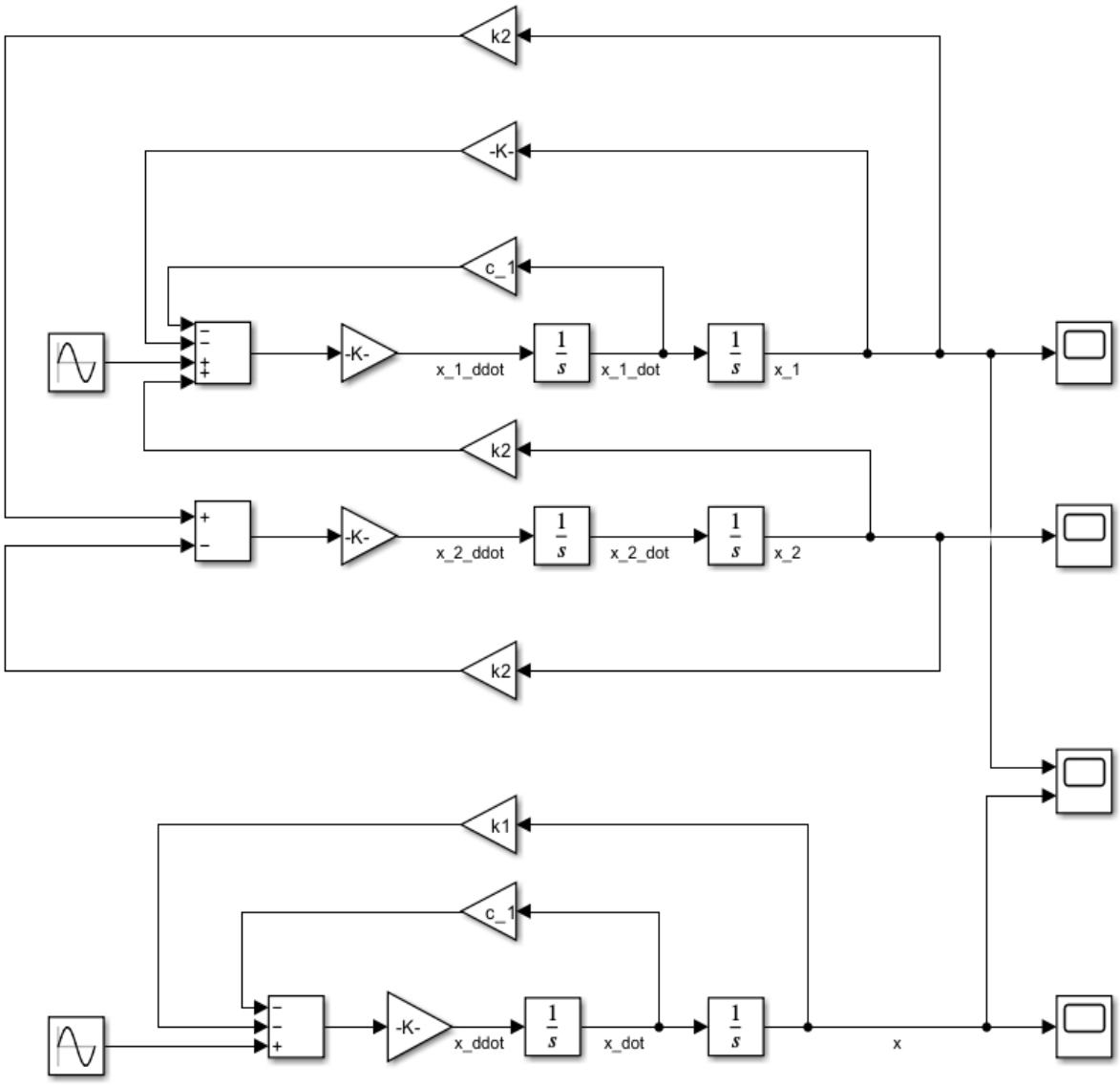




2. In extra we calculated of percentage of the speed range according to working speed. ($MATLAB\ code\ 109{\sim}112$)

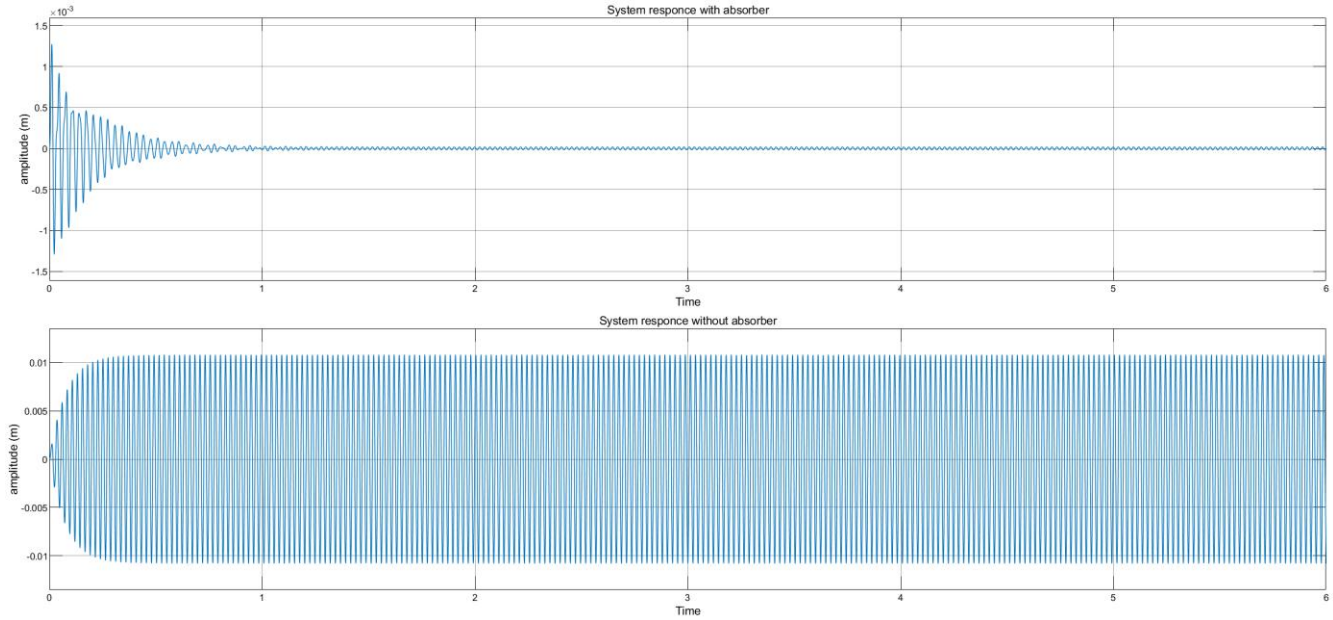
   - $The\ natural\ frequencies\ is\ at\ least:$
     $\pm 28.089163\ \%\ from\ the\ operating\ speed$

Modeling using Simulink

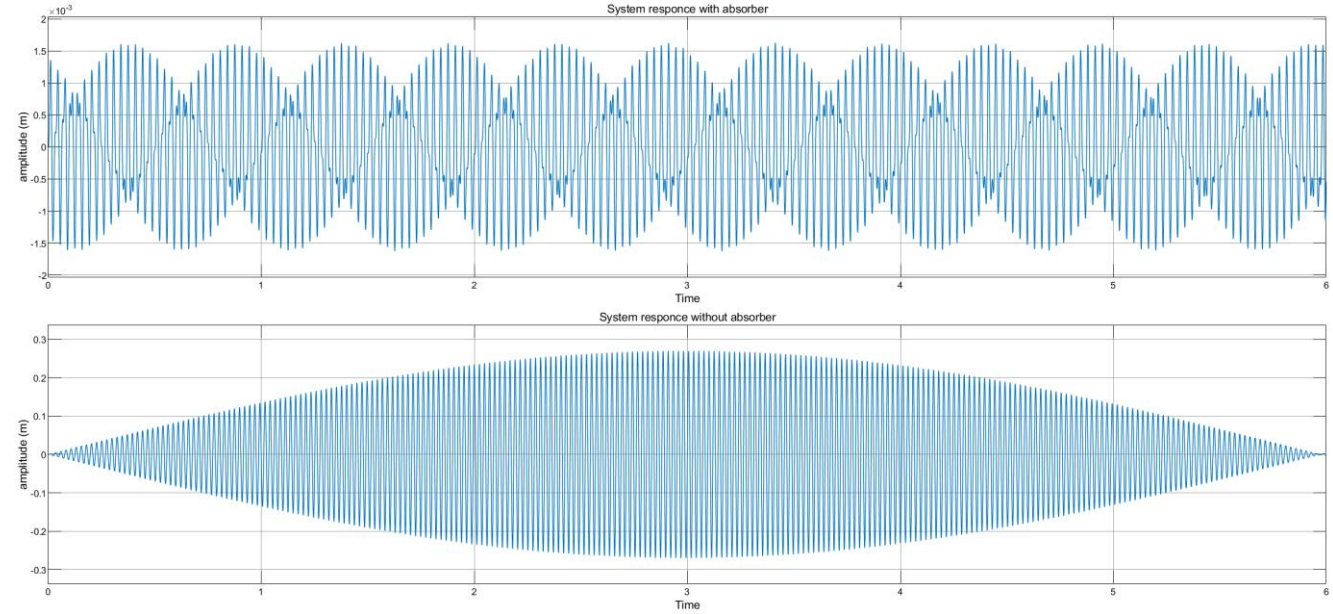


Results from Simulink

1. Damped system



2. With no damping



Conclusion

The system is experiencing rotational unbalance, resulting in excessive amplitude, which adversely affected its stability and performance. Such unbalances are common in rotating machinery and can lead to increased wear, noise, and potential failure if not addressed, the maximum amplitude was 10.83mm, we aim to reduce this maximum amplitude.

There are three methods to decrease this amplitude:
- Introducing damping into the system.
- Modifying the system's mass and spring stiffness.
- Incorporating an auxiliary mass system.

The required method for amplitude reduction is an auxiliary mass system. This approach effectively countered the rotational unbalance, reducing the amplitude to a more acceptable level and thereby improving the system's overall stability. The auxiliary mass system worked by altering the dynamics of the system, thereby mitigating the forces caused by the unbalance.

The successful reduction of amplitude through this method highlights the importance of innovative solutions like the use of auxiliary masses to enhance the performance of mechanically rotating systems. By introducing the auxiliary mass, we were able to achieve a more controlled and efficient operation, reducing the risk of mechanical failure and increasing the longevity of the system. Additionally, this method offers a relatively simple and cost-effective way to address the problem of rotating unbalance.

To adapt the auxiliary mass system for this application, we utilized a spring with a stiffness of 17000 N/mm, a coil diameter of 2mm, a mean spring diameter of 10mm, and 10 coils. Additionally, the optimal mass for the auxiliary mass system is 250g, and we will use variable masses of 50%, 100% and 150%.

- With the 50% of the optimal mass, the maximum amplitude of the auxiliary mass system was 4.88mm, while the maximum amplitude of the system itself was 2.42mm. This resulted in a 77.65% reduction in the system's maximum amplitude.

- With the 100% of the optimal mass, the maximum amplitude of the auxiliary mass system was 2.399mm, while the maximum amplitude of the system itself was 0.019mm. This resulted in a 99.82% reduction in the system's maximum amplitude.
- With the 150% of the optimal mass, the maximum amplitude of the auxiliary mass system was 1.59mm, while the maximum amplitude of the system itself was 0.81409mm. This resulted in a 92.48% reduction in the system's maximum amplitude.

As observed in the previous points, the maximum amplitude of the system initially decreases and then increases, while the maximum amplitude of the auxiliary mass system continues to decrease.

In conclusion, the use of an auxiliary mass system proves to be an effective and practical solution for reducing amplitude caused by rotating unbalanced masses, offering a promising approach to improving system performance and durability.