DS-3001

Design & Fabrication Project Report

DFP\_013: MOTOR SPEED CONTROL BASED ON GEAR

VIBRATION WITH SMART LUBRICATION SYSTEM

Rajiv Kumar Tiwari(21bsm043)

Gaurav Singh(21bcs086)  
Omkar Gadam(21bcs083)

Bikash Sabar(21bcs060)

Ashish Kol(21bcs041)

Saurabh Kumar(21bme048)  
Adi Pramod Patil(21bds003)

​

​

* **Abstract:**

This report outlines the development and integration of a sophisticated system that combines gear vibration analysis to regulate motor speed and a smart lubrication system integrated with IoT technology for remote monitoring and control. The system aims to enhance operational efficiency, minimize wear and tear, and enable real-time maintenance in industrial settings.

* **Introduction :**

In various industrial applications, precise control of motor speed and efficient maintenance of gear systems are critical for optimal performance. Traditional approaches often lack real-time monitoring capabilities and require manual intervention. To address these challenges, our project focuses on merging gear vibration analysis for speed control and an IoT-driven smart lubrication system to enable remote management.

* **Objectives :**

Develop a control mechanism utilizing gear vibration analysis to regulate motor speed.Design and implement a smart lubrication system that responds to real-time operational needs.Integrate IoT capabilities for remote monitoring and control of the entire system.Improve operational efficiency, reduce downtime, and extend equipment lifespan.

* **Methodology :**
* Gear Vibration Analysis for Speed Control:

Employed accelerometers to collect vibration data from the gear meshing.Utilized MATLAB for signal processing, feature extraction, and algorithm development to determine optimal motor speed adjustments based on vibration patterns.

Integrated Arduino for data acquisition and transmission between sensors and the control system.

* Data Collection and Processing:

Gather data using sensors accelerometers to measure vibrations related to gear meshing.

Process this data in MATLAB to extract relevant features or parameters indicative of the gear condition or vibration intensity.

* Control Algorithm Development:

Develop a PID Controller in MATLAB based on the extracted features from the sensor data. This Controller determines the required motor speed adjustment based on the gear meshing vibrations.

* Interface between MATLAB and Arduino Nano/Relay :

The MATLAB controller generates a control signal of voltage based on the calculated motor speed adjustment.Connect the output from the MATLAB controller to the input of the Arduino Nano. Ensure compatibility between the control signal generated by MATLAB and the input interface of the Arduino Nano.

* Transmission of Control Signals:

Send the control signals from MATLAB to the Arduino Nano. These signals instruct the Arduino Nano to adjust the RPM through voltage supplied to the motor, thereby controlling its speed based on the detected gear meshing vibrations.

* Automatic Speed Adjustment:

The Arduino Nano receives the control signals and automatically adjusts the motor's RPM and voltage according to the received instructions. This allows the motor to run at the desired speed specified by the MATLAB controller.

* Testing and Validation:

Tachometer is used to test the complete system to verify that the Arduino Nano responds accurately to the control signals from MATLAB, resulting in the desired motor speed adjustments corresponding to the detected vibrations.

* Smart Lubrication System:

Designed a smart lubrication system with controllable dispensers.Utilized IoT-enabled microcontrollers to regulate lubricant flow based on real-time gear conditions.Implemented feedback mechanisms to ensure precise lubrication dosage as per system demands.

* IoT Integration:

Employed IoT devices and cloud-based platforms for remote monitoring and control.

Developed a user interface accessible via a web or mobile application for real-time system status monitoring and adjustment of motor speed and lubrication remotely.

* Physical Model Building :
* Gear Design and Modeling:

Unity software is used to design and model the gears with precision, ensuring accurate tooth profiles, pitch diameters, and tooth clearances.

Unity tools allow engineers to visualize gear interactions, verify clearances, and assess potential issues before manufacturing.

* Assembly and Animation:

Unity facilitates the assembly of gears, showcasing their meshing interactions. Animated simulations can depict how gears interact during rotation, aiding in visualizing meshing behavior and identifying potential interferences.

* Clearance Analysis:

Unity tools enable clearance analysis, ensuring proper alignment and clearances between gear teeth and components

* Gear Meshing Vibration Analysis:

Specific simulation software, such as specialized vibration analysis tools software with vibration modules is employed to analyze gear meshing vibrations.

* Data Collection:

Sensor accelerometer are strategically placed on or near the gears to measure vibrations during operation.

Vibration data such as amplitude, frequency, and patterns are collected from these sensors.

* Matlab/Simulink:

MATLAB offers various signal processing and analysis toolboxes that can be utilized for vibration signal analysis.

Simulink, a MATLAB-based simulation platform, can be used to model and analyze the dynamic behavior of gear systems.

* Proteus/Arduino IDE:

Proteus and Arduino IDE synergize for circuit analysis and microcontroller programming.

Proteus enables detailed circuit simulation and analysis.

Arduino IDE facilitates programming for microcontrollers like Arduino.

Seamless integration streamlines electronic design and prototyping workflows.

* Preparing the Model:
* Gear System Model:

Create a detailed 3D model of the gear system in Unity Mechanical. Ensure accurate representation of gears, shafts, bearings, and other components.

* Material Properties:

Assign appropriate material properties to the gear components within unity , considering their elasticity, density, and other mechanical characteristics.

* Simulating Gear Damage:
* Define Damage Parameters:

Introduce simulated damage scenarios to the gear model (e.g., missing teeth, crack formations, wear patterns) to represent various types of gear damage.

* Mesh Generation:

Generate a suitable mesh for the gear model. Ensure proper refinement in critical areas to accurately capture the effects of damage and contact between gear teeth

* Vibration Analysis:
* Perform Analysis:

Run the vibration analysis in unity software to obtain vibration response data associated with the gear damage scenarios introduced earlier.

* Post-Processing:

Review and analyze the results obtained from the vibration analysis. Evaluate frequency spectra, displacement patterns, stress distributions, and other relevant data to identify signatures of gear damage-induced vibrations.

* Comparison and Interpretation:

Compare the vibration data obtained from different damage scenarios to establish correlations between specific types of gear damage and the resulting vibration patterns.

* Stress Variation:

Vibration can induce cyclic loading and unloading, causing stress variations in gear teeth and other components. These stress variations might exceed the material's fatigue limit, leading to potential fatigue failure.

* Strain Distribution:

Vibration-induced dynamic loads can cause strains in gears and related structures, affecting their structural integrity. Strain variations may lead to localized deformations or changes in the material's properties.

* Dynamic Loads:

Gear meshing and other mechanical interactions subject components to dynamic loads during vibration, resulting in dynamic stress and strain distribution across the gear teeth and associated structures.

* Frequency Analysis:

Vibration analysis often involves frequency domain analysis, examining the vibration signatures' frequency spectra to identify resonances, characteristic frequencies, and potential modes of failure.

* APP for Remote monitoring And Control.

Remote Monitoring: Allows monitoring of devices and sensors from anywhere via smartphone app.

Remote Control: Enables remote control of connected devices, such as turning them on/off or adjusting settings.

Customizable Interface: Offers customizable user interface with various widgets for different applications.

Actuator: On/Off actuator for lubrication when needed remotely through App.

* Application.

Automatic speed control in driverless cars adjusts motor speed based on real-time feedback, ensuring optimal performance, safety, and fault tolerance.

Increased Efficiency: Adjusts motor speed based on gear condition, optimizing performance and reducing energy consumption.

Extended Equipment Lifespan: Prevents further damage to gears by controlling motor speed, thereby prolonging equipment lifespan.