Conversion of Petrol-Based Engine Bike to Electric Bike Using BLDC Motor and Arduino Coding

A Sample Thesis Document

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering

Department of Mechanical Engineering   2020

Abstract

The transition gasoline based vehicles to electric mobility is essential to reduce carbon emissions, fuel costs, and environmental pollution. This study focuses on the conversion of a petrol-based two-wheeler into an electric bike by integrating a Brushless DC (BLDC) motor and an Arduino-based control system. The design includes retrofitting the frame with an electric drivetrain, configuring an energy-efficient battery system, and developing an Arduino-controlled throttle and braking system. The outcomes demonstrate the feasibility of retrofitting as a cost-effective and environmentally friendly alternative to traditional motorcycles.

Chapter 1: Introduction

Background

Rising fuel prices, environmental concerns, and technological advancements in electric mobility have accelerated the demand for electric vehicles.

Problem Statement

Petrol-based motorcycles contribute significantly to urban air pollution and greenhouse gas emissions. Converting existing bikes to electric offers a low-cost sustainable solution.

Objectives

1. To design and implement an electric drivetrain using a BLDC motor.
2. To develop an Arduino-controlled throttle and braking system.
3. To evaluate performance in terms of efficiency, range, and cost.

Scope

Focuses on conversion of a single standard petrol-based bike with modifications limited to motor, controller, and battery integration.

Chapter 2: Literature Review

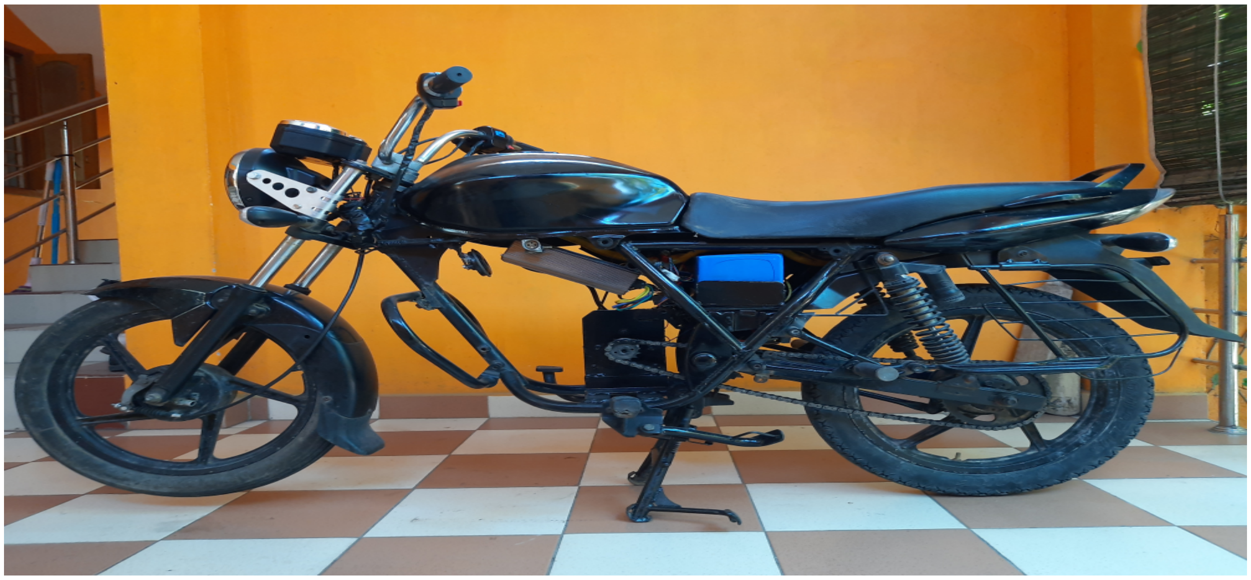
- Review of electric vehicle (EV) technologies.
- BLDC motor advantages: high efficiency, low maintenance, smooth torque characteristics.
- Role of Arduino in motor control: PWM signals, throttle input, regenerative braking logic.
- Previous studies on petrol-to-electric retrofitting.

Chapter 3: Methodology

1. Bike Selection & Disassembly: Choosing a lightweight motorcycle frame; removing internal combustion engine and fuel system.
2. Motor Selection: Using a 1–2 kW BLDC motor suitable for urban commuting (speed 50–70 km/h).
3. Battery Pack: Lithium-ion battery pack (48V/60V depending on motor rating).
4. Arduino Coding:
   - Throttle signal interpretation.
   - Generating PWM signals for BLDC controller.
   - Monitoring speed and battery voltage.
   - Implementing safety cut-offs.
5. Integration: Mounting motor, battery, and controller into the bike chassis.
6. Testing: Range test, speed test, charging time, and cost analysis.

Chapter 4: Results & Discussion

- Successful integration of BLDC motor and Arduino-based control.
- Achieved speed ~50 km/h and range ~40 km per charge (Test run) .
- Charging time: ~ 6 hours from standard AC socket.
- Cost analysis: Conversion significantly cheaper than purchasing a new EV.
- Challenges: Weight balance, heat management, limited battery capacity.



A final output of the Gasoline Bike with metallic frame chassis made into Electic Bike with all consumables inclusive of BLDC based motor , Audino enhanced Operative system , Along with Lithium Ion battery as a conceptualisation used for Tesla Vehicles significantly

Significant faults and difficulties are identified through every successive trail runs after each tests , particular misfortunes and difficulties observed are corrected again .

Chapter 5: Conclusion & Future Work

Converting petrol bikes to electric using BLDC motors and Arduino control is technically feasible and economically viable. Reduces emissions, operating cost, and maintenance.

Future improvements:
- IoT integration for remote monitoring.
- Regenerative braking optimization.
- Use of AI-based energy management systems.

References

(To be filled with IEEE/APA style references such as EV research papers, Arduino BLDC projects, motor datasheets, and EV retrofitting case studies.)

Sample Arduino Code for BLDC Motor Control:

```cpp
int throttlePin = A0;
int pwmPin = 9;

void setup() {
  pinMode(pwmPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int throttleValue = analogRead(throttlePin);
  int pwmValue = map(throttleValue, 0, 1023, 0, 255);
  analogWrite(pwmPin, pwmValue);
  Serial.println(pwmValue);
  delay(100);
}
```