

Modul de Operare al unui Acumulator într-un Sistem Auto cu Propulsie Electrică

Autor: Nemeti Alin Alexandru¹

alinnemeti14@gmail.com

Coordonatori: Șef lucr.dr. ing. Cosmin RUS², Șef lucr.dr.ing. Marius RÎSTEIU²

¹*Universitatea din Petroșani, Facultatea I.M.E., specializarea: TTIA , anul 1*

²*Universitatea din Petroșani, Facultatea I.M.E., Departamentul:A.C.I.E.E.*

Rezumat

Acest proiect se concentrează pe proiectarea și implementarea unui simulator de propulsie electrică, având ca scop simularea și înțelegerea comportamentului unei mașini electrice în diverse condiții de conducere. Prin intermediul acestui simulator, se urmărește oferirea unei experiențe realiste de conducere a unei mașini electrice, permițând utilizatorilor să exploreze caracteristicile de accelerație, frânare și gestionare a bateriei. Prin acest demers, se are în vedere îmbunătățirea înțelegerii tehnologiilor de propulsie electrică și evaluarea performanței vehiculelor electrice într-un mediu virtual controlat și sigur.

1. Introducere

Odată cu creșterea preocupărilor legate de impactul asupra mediului și nevoia de a reduce dependența de combustibili fosili, vehiculele electrice au devenit din ce în ce mai populare și importante în peisajul transportului modern. Aceste vehicule reprezintă o soluție promițătoare pentru reducerea emisiilor de gaze cu efect de seră și pentru promovarea unei mobilități mai durabile și mai ecologice. În acest context, proiectul nostru se concentrează pe dezvoltarea unui simulator de propulsie electrică, cu scopul de a furniza o platformă interactivă și educațională pentru înțelegerea și explorarea caracteristicilor mașinilor electrice. Acest simulator își propune să ofere un instrument util pentru testarea și evaluarea performanței vehiculelor electrice în diferite scenarii de conducere, contribuind la îmbunătățirea cunoștințelor noastre despre tehnologiile de propulsie electrică și comportamentul acestor vehicule în diverse condiții de utilizare.

În cadrul acestei introduceri, vom explora contextul și motivația din spatele proiectului nostru, evidențiind importanța și relevanța sa în contextul actual al mobilității durabile și al tranziției către vehiculele electrice. De asemenea, vom prezenta obiectivele principale ale proiectului și modul în care acesta contribuie la îmbunătățirea cunoștințelor noastre despre tehnologiile de propulsie electrică și a performanței vehiculelor electrice.

2. Componentele principale ale aplicației

- Interfața Utilizatorului (UI) este compusa din butoanul de accelerație/frână , etichete, indicatori grafici pentru viteză/baterie;
- Motorul de Simulare are 2 componente principale: Logica de simulare a accelerației și a frânei și gestionarea bateriei;
- Componente de Vizualizare: Elementele grafice pentru reprezentarea mașinii electrice și a mediului înconjurător, afișaje pentru informații detaliate despre performanța mașinii electrice, cum ar fi distanța parcursă, consumul de energie etc;
- Gestiunea Datelor: Stocarea datelor de configurare și gestionarea datelor de simulare;

Motorul de simulare:

Logica de simulare a accelerării: Gestionează comportamentul mașinii electrice în timpul accelerării, ajustând viteza și nivelul bateriei în funcție de acțiunile utilizatorului.

Logica de simulare a frânării: Simulează procesul de frânare al mașinii electrice și impactul asupra vitezei și nivelului bateriei.

Gestionarea bateriei: Monitorizează nivelul bateriei și efectuează calcule pentru a simula descărcarea și încărcarea bateriei în timpul utilizării.

Fiecare componentă joacă un rol crucial în funcționarea și utilitatea aplicației, oferind utilizatorului o experiență interactivă și educativă în conducerea unei mașini electrice simulate.

3. Proiectarea și implementarea

Pentru acest proiect, am ales să folosesc limbajul de programare C# împreună cu mediul de dezvoltare integrat (IDE) Visual Studio. C# este un limbaj de programare modern și robust, dezvoltat de Microsoft, care oferă o sintaxă clară și ușor de înțeles, fiind ideal pentru dezvoltarea de aplicații Windows.

Visual Studio este un mediu de dezvoltare extrem de popular, recunoscut pentru funcționalitățile sale avansate de editare de cod, gestionare a proiectelor și depanare. Beneficiază de instrumente puternice de design pentru interfețe grafice, precum și de o integrare perfectă cu platforma .NET, ceea ce face lucrul cu limbajul C# foarte eficient și productiv.

C# și Visual Studio sunt două dintre cele mai răspândite și apreciate instrumente de dezvoltare pentru aplicații Windows. Eu fiind familiarizat cu aceste tehnologii, ceea ce facilitează înțelegerea și gestionarea codului sursă. Atât C#, cât și Visual Studio sunt concepute pentru a facilita dezvoltarea de aplicații Windows robuste și scalabile. Având la dispoziție unelte puternice de depanare, testare și gestionare a proiectelor, putem asigura că aplicația noastră va fi fiabilă și stabilă în timpul utilizării.

Am început prin analiza cerințelor proiectului și stabilirea obiectivelor principale. Am elaborat un plan detaliat care cuprindea etapele de implementare a funcționalităților și de creare a interfeței grafice. În cadrul mediului de dezvoltare Visual Studio, am inițiat proiectul și am definit arhitectura acestuia, inclusiv crearea claselor și a elementelor de bază ale interfeței grafice. Ulterior, am dezvoltat logica aplicației pentru a gestiona funcționalitățile esențiale ale motorului electric, cum ar fi accelerația, frânarea și monitorizarea nivelului bateriei și a temperaturii. Am proiectat și implementat interfața grafică folosind instrumentele de design din Visual Studio, asigurându-ne că utilizatorul poate interacționa eficient cu aplicația. Am urmat un proces riguros de testare pentru a verifica funcționalitățile aplicației și pentru a identifica eventualele erori sau bug-uri. Am făcut ajustări finale pentru a optimiza performanța și a îmbunătăți experiența utilizatorului.

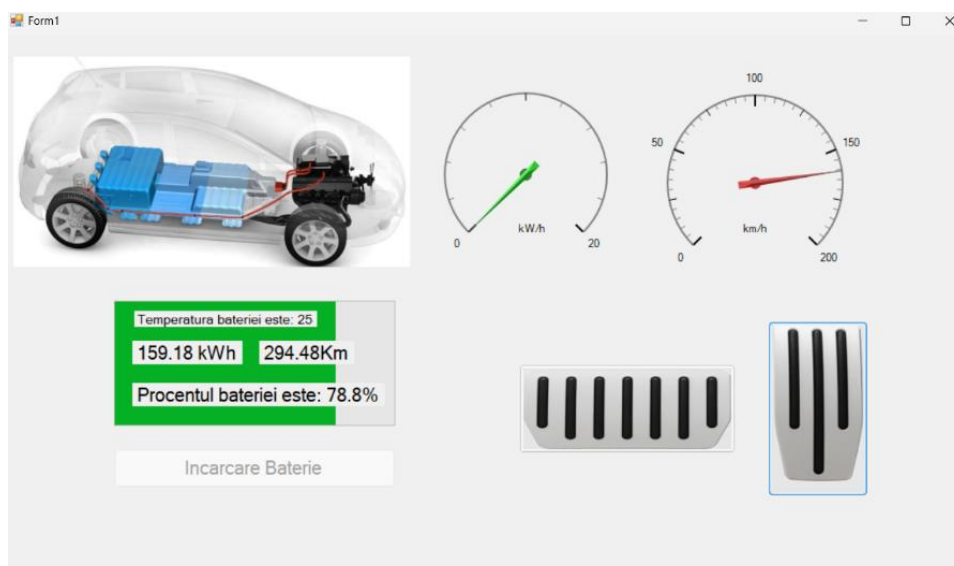


Fig. 3.1 Accelerare

Când butonul de accelerare (fig. 3.1) este apăsat, viteza motorului crește, iar consumul de energie al acestuia crește corespunzător. Astfel, nivelul bateriei începe să scadă mai rapid, reflectând consumul sporit de energie al motorului.

De exemplu, dacă motorul accelerează mai rapid, acesta va consuma o cantitate mai mare de energie din baterie pentru a menține viteza crescută. Prin urmare, în timp ce butonul de accelerare este apăsat, nivelul bateriei va scădea mai rapid în bara de progres. Când butonul de accelerare nu mai este apăsat crește puțin de tot nivelul bateriei și indicatorul de regenerare energie.

Butonul de fână scade viteza în indicatorul de viteză mai repede iar butonul de încărcare (fig. 3.2) baterie încarcă bateria încet simulând încărcarea la o stație specială de încărcare.

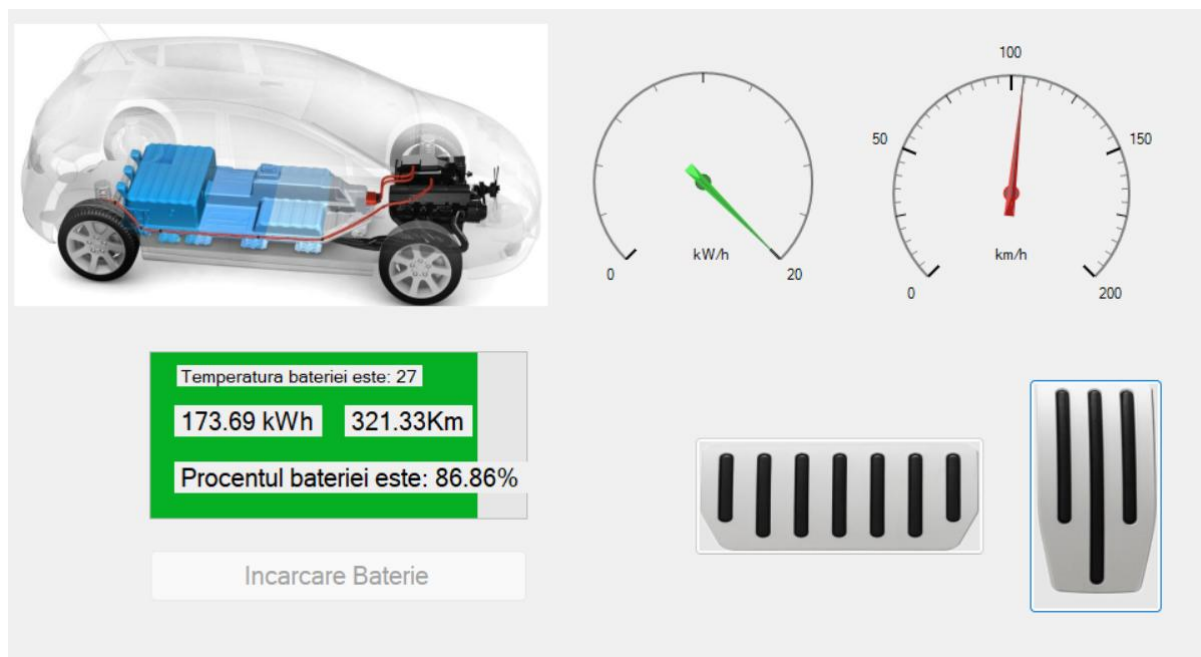


Fig. 3.2 Încărcare Baterie

Timer1: Actualizarea continuă a afișajului și a stării sistemului: Timer1 este responsabil pentru actualizarea continuă a afișajului și a stării sistemului în cadrul aplicației. Acest timer rulează periodic și actualizează informațiile afișate utilizatorului, precum viteza, nivelul de încărcare a bateriei, temperatura, etc. în funcție de starea actuală a motorului electric și a altor variabile relevante.

Timer2: Gestionarea încărcării bateriei: Timer2 este utilizat pentru gestionarea încărcării bateriei în cadrul aplicației. Acest timer rulează în mod periodic și actualizează nivelul de încărcare a bateriei în funcție de acțiunile utilizatorului și de alte factori care afectează încărcarea bateriei.

Timer3: Monitorizarea și gestionarea acțiunilor frânei: Timer3 monitorizează acțiunile utilizatorului legate de frânare și gestionează corespunzător aceste acțiuni în cadrul aplicației. Acest timer este responsabil pentru gestionarea efectelor frânării asupra vitezei și a stării sistemului.

Timer4: Gestionarea situațiilor critice ale bateriei și vitezei: Timer4 este utilizat pentru gestionarea situațiilor critice ale bateriei și vitezei în cadrul aplicației. Acest timer monitorizează constant starea bateriei și a vitezei și intervine în situațiile în care acestea ajung la valori critice, luând măsuri corespunzătoare pentru gestionarea acestor situații.

Timer5: Gestionarea încărcării bateriei după situațiile critice: Timer5 este responsabil pentru gestionarea încărcării bateriei după ce au avut loc situații critice în cadrul aplicației. Acest timer rulează în mod periodic și actualizează nivelul de încărcare a bateriei pentru a readuce sistemul la o stare sigură și funcțională după intervenția în situațiile critice.

4. Codul aplicației

Aceste segmente de cod (ilustrate în figurile 4.1, 4.2, 4.3 și 4.4) reprezintă un exemplu de implementare a unei aplicații destinate simulării unui motor electric, dezvoltată în limbajul de programare C# și adaptată pentru utilizarea platformei Windows Forms.

Viteza motorului electric este actualizată în funcție de acțiunile utilizatorului (accelerare, frânare). Nivelul de încărcare a bateriei este monitorizat și actualizat în funcție de consumul de energie al motorului electric și de alte surse de încărcare, cum ar fi regenerarea energiei la frânare. Temperatura bateriei este simulată și afișată utilizatorului, variind între anumite limite în funcție de acțiunile și starea motorului electric.

Cod:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Motor_electric
{
    public partial class Form1 : Form
    {
        Random a = new Random();
        double j; //pt nr random
        bool accel = false; //acceleratie
        double v; //viteza
        double x; //val kwh
        double y; //range
        double i = 0; //incarcare decelerare
        double z = 100; //procent baterie
        bool frana = false;
        public Form1()
        {
            InitializeComponent();

            private void button1_MouseDown(object sender, MouseEventArgs e)
            {
                accel = true;
            }

            private void button1_MouseUp(object sender, MouseEventArgs e)
            {
                accel = false;
            }

            private void timer1_Tick(object sender, EventArgs e)
            {
                double xa = Math.Round((double)x, 2);
                double ya = Math.Round((double)y, 2);
                double za = Math.Round((double)z, 2);
                progressBar1.Value = Convert.ToInt16(z);
                x = z * 2;
                y = x * 1.85;
                labelkwh.Text = xa + " kWh";
                labellvl.Text = "Procentul bateriei este: " + za.ToString() + "%";
            }
        }
    }
}
```

```

labelrange.Text = ya + "Km";
if (accel == true && v <= 200 || accel == true && v >= 200)
{
    v = v + 3;
    aGauge1.Value = Convert.ToInt16(v);
    z = z - v * 0.005;
}
else if (accel == false && v >= 0)
{
    v--;
    aGauge1.Value = Convert.ToInt16(v);
}
for (j = 0; j <= 5; j++)
{
    if(timer2.Enabled == true)
        labeltemp.Text = "Temperatura bateriei este: " + a.Next(35, 40);
    else labeltemp.Text = "Temperatura bateriei este: " + a.Next(25, 30);
}
if (accel == false && v > 0 && i<=20 && z<=100)
{
    i = i + 1;
    aGauge2.Value = Convert.ToUInt16(i);
    z = z + 0.01;
}
else if (i > 0)
{
    i = i - 1;
    aGauge2.Value = Convert.ToUInt16(i);
}
if (v > 0)
    button3.Enabled = false;
else button3.Enabled = true;
}
private void button3_Click(object sender, EventArgs e)
{
    timer2.Enabled = true;
}

private void timer2_Tick(object sender, EventArgs e)
{
    if (z <=100)
    {
        z = z + 0.01;
    }
}

private void button2_MouseDown(object sender, MouseEventArgs e)
{
    frana = true;
}

private void button2_MouseUp(object sender, MouseEventArgs e)
{
    frana = false;
}

private void timer3_Tick(object sender, EventArgs e)
{
    if (frana == true && v>0)
        v = v - 5;
    if(z >= 100)
        timer2.Enabled = false;
}

```

```

private void timer4_Tick(object sender, EventArgs e)
{
    if ((accel == false && z<=5) || (accel == true && z <= 5))
    {
        v = 0;
        aGauge1.Value --;
        aGauge2.Value = 0;
        button1.Enabled = false;
        button3.Visible = false;
        button4.Visible = true;
        timer1.Enabled = false;
    }
}

private void button4_Click(object sender, EventArgs e)
{
    timer5.Enabled = true;
}

private void timer5_Tick(object sender, EventArgs e)
{
    if (z <= 100)
    {
        z = z + 0.01;
    }
}
}
}

```

5. Concluzii

Proiectul " Simulator de propulsie electrică" reprezintă o aplicație interactivă și educativă ce oferă o simulare realistă a funcționării unui motor electric. Scopul său principal este de a furniza utilizatorilor o platformă intuitivă pentru înțelegerea conceptelor fundamentale legate de vehiculele electrice și pentru explorarea factorilor ce influențează performanța și starea unui motor electric.

Prin intermediul unei interfețe grafice prietenoase, utilizatorii pot interacționa direct cu motorul electric simulat, experimentând diferite scenarii și observând efectele modificărilor asupra acestuia. De la accelerare și frânare până la monitorizarea nivelului de încărcare a bateriei și a temperaturii, aplicația oferă o gamă largă de funcționalități pentru a ilustra diverse aspecte ale tehnologiei vehiculelor electrice.

În concluzie, " Simulator de propulsie electrică" nu numai că facilitează înțelegerea conceptelor legate de motoarele electrice, dar și încurajează explorarea și experimentarea într-un mediu sigur și interactiv. Acest proiect reprezintă un instrument valoros pentru educație și cercetare în domeniul vehiculelor electrice și contribuie la promovarea tehnologiilor durabile și prietenoase cu mediul înconjurător.

6. Bibliografie

1. The C# Programming Language By Anders Hejlsberg, Mads Torgersen, Scott Wiltamuth, Peter Golde
2. Programming C#: Building .NET Applications with C# By Jesse Liberty
3. Microsoft Visual C# 2013 Step by Step By John Sharp
4. C# 10 in a Nutshell: The Definitive Reference By Joseph Albahari
5. The Electric Car: Development and Future of Battery, Hybrid and Fuel-cell Cars By Michael Hereward Westbrook
6. Optimal behavior of electric vehicle parking lots as demand response aggregation agents by Miadreza Shafie-Khah, Ehsan Heydarian-Forushani, Gerardo J Osório, Fábio AS Gil, Jamshid Aghaei, Mostafa Barani, João PS Catalão