Parti ale documentului scris pentru licenta:

* Software pentru sisteme embeded
* Tehnologii folosite
* Modele UML pentru calculatorul de bord
* Descrierea simulatorului
  + Explicarea modalitatii in care se modifica acele celor doua indicatoare
  + Explicarea preluarii mesajelor vocale folosind google si progamul de taiere/editare audio
  + Explicarea preluarii datelor pentru parcurgerea scenariilor din tabele

**Firmware**

În electronica şi informatica, firmware-ul este un termen folosit adesea pentru a desemna programele fixe, de obicei destul de mici, şi structurile de date ce controlează intern diverse dispozitive electronice. Printre exemplele tipice de dispozitive care conţin produse destinate utilizatorului final din gama firmware, ar fi calculatoare sau telecomenzi, piese de calculator şi dispozitive precum hard-disk-uri, tastaturi, ecrane TFT sau carduri de memorie, pana la instrumente stiintifice şi roboti industriali. De asemenea, produse de consum mai complexe, cum ar fi telefoanele mobile, camerele digitale, sintetizatoarele, etc, conţin firmware pentru a permite efectuarea functiilor de bază, precum şi pentru implementarea funcţiilor de nivel înalt.

Nu există graniţe stricte între firmware şi software, ambii termeni descriptivi fiind destul de vagi. Cu toate acestea, termenul de firmware a fost iniţial introdus în scopul de a contrasta cu software-ul de nivel superior care poate fi schimbat fara înlocuirea unei componente hardware, acesta fiind de obicei implicat in asigurarea functionarii operatiilor de bază cu nivel scăzut de complexitate. fără de care un dispozitiv ar fi complet nefuncţional. Firmware-ul este, de asemenea, un termen relativ, pentru ca cele mai multe dispozitive integrate conţin firmware la mai mult de un nivel. Subsisteme, cum ar fi CPU, memorii flash, regulatoare de comunicare, module LCD, şi aşa mai departe, au propriul lor (de obicei fix) cod de program şi / sau microcod, considerat ca fiind "parte a hardware-ului", de catre firmware-ul de nivel superior.

Firmware-ul simplu de obicei se află în ROM sau OTP /PROM, în timp ce firmware-ul mai complex (de multe ori la graniţa cu software-ul) foloseste de obicei memorie flash pentru a permite actualizări, cel puţin în dispozitivele moderne. Motive comune pentru actualizarea firmware-ul includ rezolvarea bug-urilor sau adăugarea de caracteristici noi pentru dispozitiv. Făcând acest lucru, de obicei implică încărcarea unui fişier imagine binara (furnizat de către producător) în dispozitiv, în conformitate cu o procedură specifică; acest lucru, este uneori dorit (de către producătorul dispozitivului) a fi efectuat de către utilizatorul final.

**Originea termenului**

Ascher Opler a inventat termenul de "firmware" într-un articol din 1967 din revista Datamation. Iniţial, aceasta a însemnat microcod – continutul unui spatiu de depozitare ce poate fi scris (o memorie mica specializata de mare viteză), care a definit si a implementat setul de instructiuni al calculatorului. Daca este necesar, se poate reincarca firmware-ul pentru a specializa sau a modifica instructiunile ce pot fi executate de unitatea centrala de procesare (CPU). Asa cum a fost folosit initial, firmware-ul contrasta cu hardware-ul (chiar cu CPU-ul ) si cu software-ul (instructiunile normale executate de CPU). Nu era compus din instructiuni masina ale unitatii centrale de procesare, ci din microcod de nivel inferior folosit la implementarea instructiunilor masina. A existat la granita dintre hardware si software, de aici si numele de “firmware”.

**Firmware in anul 2010**

Conceptul de “firmware” a evoluat in a insemna aproape orice continut programabil al unui dispozitiv hardware, nu doar codul maisna pentru un procesor, dar si configuratii si date pentru circuite integrate specializate destinate aplicatiilor (“application-specific integrated circuits - ASICs”), dispozitive logice programabile, etc.

**Calculatoare personale**

Sub anumite aspecte, diferitele componente firmware sunt la fel de importante precum sistemul de operare intr-un calculator functional. Totodata, spre deosebire de majoritatea sistemelor de operare moderne, firmware-ul are rareori un mecanism automat evoluat de a se autoactualiza pentru a rezolva orice probleme legate de functionalitate detectate dupa transportul si punerea efectiva in functiune a unitatii.

In momentul de fata, oricine poate actualiza cu usurinta BIOS-ul intr-un calculator personal modern; dispozitive precum placile video sau modemurile deseori se bazeaza pe firmware dinamic, incarcat de un driver al dispozitivului, si pot fi prin urmare actualizate in mod transparent prin mecanismul de actualizare propriu al sistemului de operare. In contast, firmware-ul din dispozitivele de depozitare este actualizate foarte rar, chiar si cand este folosita memorie flash (mai degraba decat ROM); nu existe mecanisme standardizate pentru detectarea si actualizarea versiunilor de firmware. Cu toate acestea, in practica, asemenea dispozitive au o rata de functionalitate scazuta in comparatie cu dispozitivele in care firmware-ul poate fi actualizat.

**Periferice**

Majoritatea perifericelor calculatoarelor sunt in sine mini calculatoare specializate. In timp ce dispozitivele externe au firmware incarcat intern, o data cu anul 2010, placile video moderne si placile pentru dezvoltare (expansion cards (se pot introduce in placa de baza pentru a adauga functionalitate)), au deseori parti de firmware incarcate de sistemul de operare la pornire, acest lucru oferind un grad mai ridicat de flexibilitate. Acest tip de hardware poate fi astfel nefunctionabil pana in momentul in care calculatorul gazda nu a terminat a-I “alimenta” cu firmware-ul necesar, in mod normal printr-un driver specializat (mai exact: printr-un subsistem de start-up dintr-un pachet de drivere ale dispozitivului). Driverele moderne aferente dispozitivelor interne cat si a celor externe, pot contine de asemenea o interfata grafica cu utilizatorul directa, destinata configurarii, ce adesea foloseste pati dintr-un program normal de interfatare cu utilizatorul, in plus fata de call-urile sistemului de operare, si/sau alte interfete destinare driverelor pentru dispozitive.

**Produse destinate consumatorilor**

O data cu 2010, majoritatea dispozitivelor portabile destinate ascultarii de muzica, au suport pentru actualizare de firmware. Anumite companii folosesc actualizari de firmware pentru a adauga noi formate de fisiere ce pot fi ascultate ; iriver a adaugat formatul Vorbis astfel, de exemplu. Alte functionalitati ce pot fi schimbate o data cu actualizarea firmware-ului pot include interfata cu utilizatorul, sau timpul de viata al bateriei. Majoritatea telefoanelor mobile au posibilitatea de actualizare a firmware-ului, in mare parte pentru aceleasi motive; Unele pot fi actualizate chiar pentru a mari calitatea receptiei sau a sunetului, ilustrand faptul ca firmware-ul este folosit la mai mult de un nivel in produse complexe (in microcontrollere gen CPU fata de dispozitive de analiza a semnalului – digital signal processors (DSP) in caz particular).

**Automobile**

Incepand cu anul 1996, majoritatea automobilelor au inceput sa foloseasca un calculator de bord si diversi senzori pentru detectarea problemelor mecanice. Incepand cu 2010, vehiculele moderne include de asemenea sisteme ABS controlate de calculator, precum si sisteme de control a transmisiei (Transmission Control System - TCS). Driverele folosite pot obtine in timp real, in timpul mersului informatii cum ar fi gradul de economisire a combustibilului, sau presiunea rotilor. Distribuitorii locali pot actualiza firmware-ul pentru majoritatea tipurilor de maisni.

**Definitia IEEE**

Institute of Electrical and Electronics Engineers (IEEE) Standard Glossary of Software Engineer Terminology, Std 610. 12-1990 defineste firmware-ul asa cum urmeaza:

*“Combinatia dintre un dispozitiv hardware si instructiuni de calculator si date ce rezida ca software ce poate fi doar citit (read-only) pe acel dispozitiv”.*

*Note: (1) Acest termen este uneori folosit pentru a face referire doar la un dispozitiv hardware, sau doar la instructiunile de calculator sau date, dar aceste definiri sunt depreciate. (2) Confuzia ce inconjoara acest termen a facut unele persoane sa sugereze ca el poate fi evitat in totalitate.*

**Descrierea proiectului**

**Calculul de coordonate pentru cele trei indicatoare pentru controlul manual pornit :**

Pentru a putea asigura o imagine dinamica cat mai realista a indicatoarelor pentru viteza, respectiv pentru turatie, a fost necesara impartirea pane-ului (componenta Java Swing) in mai multe zone, exprimate in program prin urmatoarele variabile **:**

* **zona1, zona2** (pentru vitezometru);
* **zona3**, **zona4** (pentru turometru);

Acest lucru a fost necesar in principal datorita faptului ca imaginea indicatorului pentru viteza ciprinde toate cele 4 cadrane ale unui cerc, acul trebuind sa poata realiza o rotatie de 360˚, precum si faptului ca axele de simetrie ale imaginii nu coincid cu axele de coodronate aferente componentei din Swing. Originea spatiului de lucru, fiind astfel in coltul din stanga sus, iar cele doua axe (abscisa si ordonata) avand sensuriledupa cum se poate observa in figura de mai jos :

(0,0)

x



Fig1: spatiul de lucru al pane-ului din Swing

y

Pentru o exemplificare mai detaliata a metodei folosite, vom lua in considerare cazul particular al indicatorului pentru viteza.

Ideea de baza in desenarea dinamica a celor 3 “ace” aferente indicatoarelor, este gasirea a 3 perechi de coordonate, corespunzatoare varfurilor acestora. In cazul vitezometrului, cele doua coordonate calculate sunt reprezentate in program prin variabilele **x** si **y**, asa cum vom vedea intr-o portiune de cod prezentata in continuare.

Asadar, modificarea in mod dinamic a acestor doua coordonate, relativ fata de actiunile utilizatorului, in modul de folosire ce permite controlul manual, sau fata de setul de date citite la rularea unui scenariu de testare, presupune modificarea uneia dintre coordonate (cresterea sau scaderea ei cu o anumita valoare – **pas1**, **pas2**, sau chiar o constanta), si calcularea celei de-a doua coordonate in functie de prima. Asa cum vom vedea, in functie de anumite circumstante aparute datorita spatiului de lucru, coordonata modificata va trebui sa fie abscisa, iar in altele ordonata, pentru obtinerea unui calcul cat mai precis.

 De asemenea, si formula folosita pentru calculul celei de-a doua coordonate va fi diferita, in functie de cele 4 zone in care este impartit pane-ul, asa cum se poate observa in figura de mai jos:

zona de sus

****

****

zona

de mijloc

dreapta

zona

de mijloc

stanga

****

zona de jos

Fig 2 : Impartirea in zone a imaginii indicatorului pentru viteza pentru calculul de coordonate

In program, cele 4 zone sunt exprimate cu ajutorul variabilelor **zona1**, si **zona2**, astfel:

**zona de jos ⇔zona1 = 1 si zona2 = 0 ;**

**zona de sus ⇔zona1 = 0 si zona2 = 1 ;**

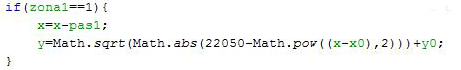
**zona de mijloc stanga ⇔zona1 = 0 si zona2 = 0 si x ≤ 304 ;**

**zona de mijloc dreapta ⇔zona1 = 0 si zona2 = 0 si x ≥ 590 ;**

Formula folosita, pentru calculul in acest caz a coordonatei **y** deriva din ecuatia cercului, ceea ce este un fapt usor de inteles, avand in vedere forma indicatorului, si modul in care acul acestuia trebuie sa se “miste” . Aceasta variaza in functie de cele 4 zone, dupa cum urmeaza :

**daca acul se afla in zona de jos (1)**

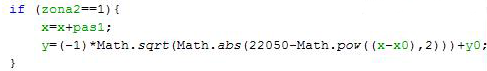
Portiunea de cod unde se realizeaza acest lucru poate fi vazuta in figura de mai jos :



Asa cum se poate observa, variabila este cea care este modificata, iar variabila  este cea calculata. Variabila va varia in functie de valoarea vitezei, determinand ca acul sa isi schimbe pozitia mai repede (la viteze mici) sau mai incet (la viteze mari).

**daca acul se afla in zona de sus (2)**

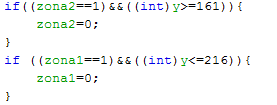
Portiunea de cod unde se realizeaza acest lucru poate fi vazuta in figura de mai jos :



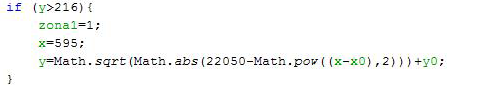
perechea de coordonate (), reprezinta originea cercului descris, sau altfel spus, celalalt capat al acului indicatorului, ce ramane fix pe parcursul functionarii programului.

Pentru cele doua zone de mijloc, coordonata va ramane constanta, iar coordonata **,**  va creste, sau va scade cu o valoare constanta (va scade pentru zona de mijloc stanga si va creste pentru zona de mijloc dreapta). Necesitatea existentei celor doua zone de mijloc, se datoreaza faptului, ca trecerea acului de la la trebuie sa fie una continua, fapt ce nu se intampla, daca pur si simplu se trece de la formula **(1)** la formula **(2).**

Se pune acum problema schimbarii intr-o maniera dinamica a acestor zone, in functie de valorile celor doua coordonate. Acest lucru este realizat in program prin metoda , unde se face trecerea din in zona de mijloc dreapta precum si trecerea din in zona de mijloc stanga, astfel:



Iesirea din cele doua zone de mijloc si intrarea in zonele de sus respectiv jos, este facuta tot in functie de valoarea coordonatei , precum se poate observa in figura de mai jos:



(aceasta portiune de cod face trecerea din zona de mijloc dreapta in zona de jos)

Aflandu-ne in cazul de utilizare ce permite controlul manual, modificarea pozitiei acului este realizata in program la detectarea evenimentului de apasare a sagetii de sus, sau a celei de jos de la tastatura, fiecare dintre cele doua cazuri avand cate o metoda aferenta, , respectiv,  **.** Aceste doua metode,coroborate cu cea dedicata setarii zonelor, asigura practic modificarea in mod dinamic si destul de realist a pozitiei acului de indicare a vitezei la momentul actual de timp.

**** In continuare este prezentata una dintre cele doua metode, cea corespunzatoare apasarii sagetii sus de la tastatura , altfel spus accelerarii manuale, pentru a avea o imagine de ansamblu asupra modalitatii prin care se calculeaza aceste coordonatele necesare.

**Calculul vitezei si turatiei**

Este realizata in cadrul programului prin intermediul metodelor si, fiind bazata pe metoda regresiei polinomiale.

**Regresia polinomiala**

O problema binecunoscuta in multe arii de cercetare este aceea care presupune existenta unui set de date privind doua sau mai multe variabile aleatoare, scopul modelarii fiind descrierea relatiei dintre ele in vederea prognozarii valorii uneia in raport cu valorile celeilalte sau celorlalte.

Aceasta problema se pune atunci cand intre variabilele aleatoare considerate exista o legatura consistenta, bazata pe natura intima a fenomenelor care stau la baza lor. Este posibil ca, din punct de vedere formal, doar pe baza datelor numerice, acestea sa para corelate, de exemplu toate sa aiba tendinta de crestere in acelasi timp, acest fapt fiind insa sustinut de natura fenomenelor in cauza.

In cazul in care legatura intre cele doua variabile statistice nu este liniara, si totusi banuim ca exista, avem o regresie neliniara, de exemplu, regresia polinomiala. Atunci, se gaseste curba respectiva de regresie (corespunzatoare, de exemplu, ecuatiei polinomiale de regresie).

Un exemplu de astfel de regresie polinomiala va furniza o ecuatie de forma:

, unde, de obicei, X este asa numita variabila predictoare iar Y este variabila prognozata, adica variabila de raspuns.

Coeficientii necunoscuti  **,** vor fi calculati folosind metoda CMMP (celor mai mici patrate), rezolvand sistemul de necunoscute:

**1 . .**

**1 . .**

**. = . . . . . . .**

**. . . . . . .**

**1 . .**

In cazul calcularii vitezei si turatiei variabila predictoare este una dintre coordonate (ori x ori y), in functie de zona careia ii corespund aceste coordonate sau chiar de valorile lor.

Vom analiza pentru inceput cazul calcularii vitezei. Metoda in cauza folosita este **,** si asa cum am mentionat anterior, formulele folosite depind de anumite zone in care am considerat a fi impartit spatiul de lucru, in cauza, componenta pane din Swing. Impartirea a fost facuta cum putem observa in Fig 3.

Aceasta impartire este datorata imposibilitatii folosirii aceleiasi coordonate ca variabila predictoare, deoarece, dupa cum se poate observa si din figura, variatia coordonatei pentru **zona de sus** este foarte mica, precum si a coordonatei pentru **zona de jos**,iar pentru obtinerea unui polinom capabil sa realizeze o predictie cat mai precisa, variatia trebuie sa fie semnificativa. Asadar, vor fi gasite 3 polinoame prin metoda de regresie polinomiala pentru a acoperi intregul arc de interes al indicatorului pentru viteza.



zona

din dreapta

zona

din stanga

zona

de sus

****

****

zona

de jos

Fig 3 : Impartirea in zone a imaginii indicatorului pentru viteza pentru calculul vitezei

Primul polinom va corespunde zonei de sus , exprimata in program printr-o limitare a valorii coordonatei **,** setul de date corespunzator acestei zone fiind prezentat in continuare, precum si graficul aferent unei regresii polinomiale de ordin 3. Variabila predictoare este in acest caz coordonata , variatia ei in aceasta portiune fiind mult mai semnificativa decat cea a coordonatei

|  |  |
| --- | --- |
| x | v |
| 358 | 55 |
| 379 | 60 |
| 403 | 65 |
| 428 | 70 |
| 454 | 75 |
| 480 | 80 |
| 505 | 85 |
| 529 | 90 |

Polinomul obtinut astfel este cel de mai jos :

Portiunea de cod din cadrul metodei pentru calculul vitezei aferenta acestui calcul este cea de mai jos:



Cel de-al doilea polinom folosit, va corespunde zonei din stanga jos a indicatorului, asadar o combinatie intre **zona din stanga** si **zona de jos** din **Fig3,** variabila predictoare fiind in acest caz coordonata **.** Setul de date corespunzator acestei zone este cel de mai jos, precum si graficul aferent unei regresii polinomiale de ordin 3.

|  |  |
| --- | --- |
| y | v |
| 349 | 0 |
| 326 | 10 |
| 304 | 15 |
| 279 | 20 |
| 254 | 25 |
| 230 | 30 |
| 208 | 34 |
| 181 | 40 |
| 157 | 45 |
| 136 | 50 |
| 117 | 55 |

Polinomul obtinut astfel este cel de mai jos :

Portiunea de cod din cadrul metodei pentru calculul vitezei aferenta acestui calcul este cea de mai jos:

****

In acest caz, variabila este cea care determina apartenenta in **zona din stanga** a indicatorului.

Cel de-al treilea polinom, va corespunde astfel zonei din dreapta jos a indicatorului, asadar, graficul sau descriind un arc ce va strabate in acelasi timp **zona din dreapta** si **zona de jos** din **Fig3.** Variabila predictoare este si in acest caz tot . Setul de date corespunzator acestei zone este cel de mai jos, precum si graficul aferent unei regresii polinomiale de ordin 3.

|  |  |
| --- | --- |
| y | v |
| 112 | 90 |
| 128 | 95 |
| 145 | 100 |
| 167 | 105 |
| 192 | 110 |
| 218 | 115 |
| 244 | 120 |
| 268 | 125 |
| 293 | 130 |
| 316 | 135 |
| 334 | 140 |

**Scenarii de testare a aplicatiei**

Scenarii principale mai complexe:

Scenariu 1:

1. Porneste motorul si asteapta 5 secunde
2. Declanseaza semnalul de atentionare pentru usi deschise(semnalul va ramane activ timp de 10 secunde apoi va fi dezactivat);
3. Accelereaza pana la viteza de 50 M/h;
4. Asteapta la viteza de 50M/h timp de 10 sec;
5. Accelereaza pana la viteza de 70 M/h ;
6. Asteapta la viteza de 70M/h timp de 5 secunde;
7. Decelereaza pana la viteza de 20M/h;
8. Declanseaza semnalul de atentionare pentru nivel scazut de combustibil(semnalul va ramane activ timp de 10 secunde apoi va fi dezactivat);
9. Opreste motorul;
10. La 0M/h asteapta 5 secunde.
11. Declanseaza semnalul de atentionare pentru lumini aprinse(semnalul va ramane activ timp de 10 secunde apoi va fi dezactivat);

Scenariu 2:

1. Porneste motorul si asteapta 10 secunde;
2. Declanseaza semnalul de atentionare pentru centura de siguranta (semnalul va ramane activ timp de 10 secunde apoi va fi dezactivat);
3. Accelereaza pana la viteza de 100M/h;
4. Asteapta la viteza de 100M/h timp de 10 secunde;
5. Franeaza pana la viteza de 50M/h;
6. Declanseaza semnalul de avertizare pentru nivel scazut al bateriei (semnalul va ramane activ timp de 10 secunde apoi va fi dezactivat);
7. Decelereaza pana la viteza de 0 M/h;
8. La 0M/h asteapta 5 secunde.
9. Opreste motorul .

Scenarii mici de test:

Scenariu mic 1:

1. Porneste motorul si asteapta 10 secunde
2. Opreste motorul

Scenariu mic 2:

1. Porneste motorul si asteapta 5 secunde
2. Declanseaza semnalul de atentionare pentru usi deschise (ramane activ 10 secunde)
3. Opreste motorul

Scenariu mic 3:

1. Porneste motorul si asteapta 10 secunde
2. Declanseaza semnaul de atentionare pentru centura de siguranta
3. Opreste motorul

Scenariu mic 4:

1. Porneste motorul si asteapta 5 secunde
2. Accelereaza pana la viteza de 60 M/h
3. Asteapta la viteza de 60M/h 5 secunde
4. Opreste motorul

Scenariu mic 5:

1. Porneste motorul si asteapta 5 secunde
2. Accelereaza pana la viteza de 50M/h
3. Declanseaza semnalul de atentionare pentru lipsa de combustibil (ramane activ 10 secunde)
4. Opreste motorul

Scenariu mic 6:

1. Porneste motoul si asteapta 5 secunde
2. Declanseaza semnalul de atentionare pentru nivel scazut al bateriei (ramane activ 10 secunde)
3. Opreste motorul

Scenariu mic 7:

1. Porneste motorul
2. Accelereaza pana la 60M/h
3. Franeaza pana la 30M/h unde asteapta 5 secunde
4. Opreste motorul

Scenariu mic 8:

1. Porneste motorul
2. Accelereaza pana la 100M/h unde astapta 10 secunde
3. Decelereaza pana la 10M/h
4. Opreste motorul

Scenariu mic 9:

1. Porneste motorul
2. Accelereaza pana la 100M/h
3. Franeaza pana la 0M/h
4. Opreste motorul
5. Declanseaza semnalul de atentionare pentru lumini aprinse (ramane activ 10 secunde)

**Diagrama de clase**

****