НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп’ютерної та програмної інженерії

Кафедра комп'ютерних інформаційних технологій

**Лабораторна робота № 2.3**

**з навчальної дисципліни**

**"Математичні моделі динамічних систем"**

Тема: Дослідження характеристик стійкості та керованості руху літака в бічній площині на ПЕОМ

Виконавець: студент групи УС-213 Дзиговський В.І.

Київ 2020

Дослідження характеристик стійкості та керованості руху літака в бічній площині на ПЕОМ

**Мета лабораторної роботи**

1. Вивчити динамічні та статичні характеристики стійкості та керованості руху літака в бічній площині.

2. Здійснити моделювання руху літака в бічній площині на ПЕОМ і дослідити характеристики його стійкості та керованості.

## **Стислі теоретичні відомості**

Просторовий рух літака описується складною системою нелінійних диференціальних рівнянь 12 порядку, як зв’язаний рух тіла, що має шість ступенів свободи, в повздовжній та бічній площинах.

Якщо відомо, що параметри руху літака в повздовжній площині не змінюються або змінюються мало і практично не впливають на параметри руху літака в бічній площині, то можна розглядати ізольований рух літака в бічній площині, який описується системою диференціальних рівнянь шостого порядку, визначених відносно .



Якщо відомо, що параметри польоту літака змінюються відносно своїх початкових значень в незначному діапазоні, то з’являється можливість здійснити лінеаризацію як самих нелінійних рівнянь, так нелінійних функцій в їх правих частинах.

Аналіз збуреного руху в бічній площині літака показує, що, за певних умов, з деяким наближенням його можна поділити на швидкозгасаючий (швидкий) і повільнозгасаючий (повільний) рух.

Швидкий рух, що складається з кутових рухів крену та рискання, розвивається в перші кілька секунд після початку дії збурень. Літак при цьому порівняно швидко обертається навколо осей *OX* і *OY*. Повільний рух стає помітним лише через якийсь час і супроводжується лінійним переміщенням центра мас літака відносно осі *OZ*.

Отже, швидкий і повільний бічні рухи ніби рознесені в часі, що дозволяє з деяким наближенням розглядати їх окремо. Поділ бічного руху на швидкий і повільний, як і у випадку поділу повздовжнього руху на короткоперіодичний і довгоперіодичний, істотно спрощує його дослідження.

На відміну від збуреного руху в повздовжній площині, характер руху в бічній площині є більш складним, тому що одночасно відбувається взаємний вплив обертальних рухів літака навколо осей *OX* і *OY* та вплив на них поступального руху центра мас літака відносно осі *OZ*.

Дослідження стійкості та керованості руху літака в бічній площині ґрунтується на поділі характеристик керованості на статичні і динамічні та взаємозв'язку їх з характеристиками стійкості.

Характеристики стійкості та керованості руху літака в бічній площині, як і характеристики стійкості та керованості в повздовжньої площині, залежать від висоти, швидкості та інших параметрів польоту.

Характеристичне рівняння руху літака в бічній площині має два дійсні і два комплексно спряжених корені. Перший дійсний корінь може бути як додатнім, так і від’ємним за знаком і є невеликим за модулем.

Якщо він додатній, то має місце так звана спіральна нестійкість літака, яка характеризується повільним і безупинним збільшенням крену літака, що супроводжується його спіральним рухом у просторі зі зменшенням висоти.

Якщо він від’ємний, то має місце спіральна стійкість літака. У цьому випадку кут крену, який після закінчення дії збурення не дорівнює нулю, повільно зменшується в процесі спірального руху літака, поки не досягне нульового значення.

Умова спіральної стійкості руху літака в бічній площині має вигляд:

>



Другому від’ємному за знаком дійсному кореню, істотно більшому за своїм модулем, ніж перший, відповідає швидкозгасаючий рух крену.

Два комплексно спряжені корені мають, як правило, невелику за модулем від’ємну дійсну частину і характеризують повільнозгасаючий рух рискання, тому що демпфірування коливань літака відносно осі *OY* визначається, в основному, вертикальним оперенням, яке має істотно меншу площу, ніж площа крила і, відповідно, менші демпфірувальні властивості.

За аналогією з короткоперіодичним рухом літака в повздовжній площині при дослідженні руху в бічній площині використовується поняття запасу стійкості по куту ковзання :



,



знак якого у літака зі стійким рухом у бічній площині завжди менше нуля.

До статичних характеристик бічної керованості літака, поруч з іншими, відноситься ефективність елеронів , яка визначається аналітично або за експериментальними даними при «дачі» елеронів зі співвідношення:



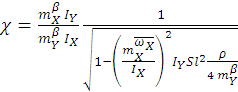
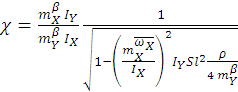
До динамічних характеристик бічної керованості літака відносяться період коливань рухів крену і рискання, час згасання збуреного руху *t*пп та показник , який характеризує відношення амплітуд кутових швидкостей крену і рискання, узятих в однаковий момент часу.



Причому, менше значення означає, що відхилення руля напрямку і поява кутової швидкості рискання не буде супроводжуватися значною кутовою швидкістю крену, тобто різким кренінням. Іншими словами, такий літак не є «строгим» щодо неточності відхилення руля напрямку і не відповідає різким рухом крену на неточність дій пілота рулем напрямку.

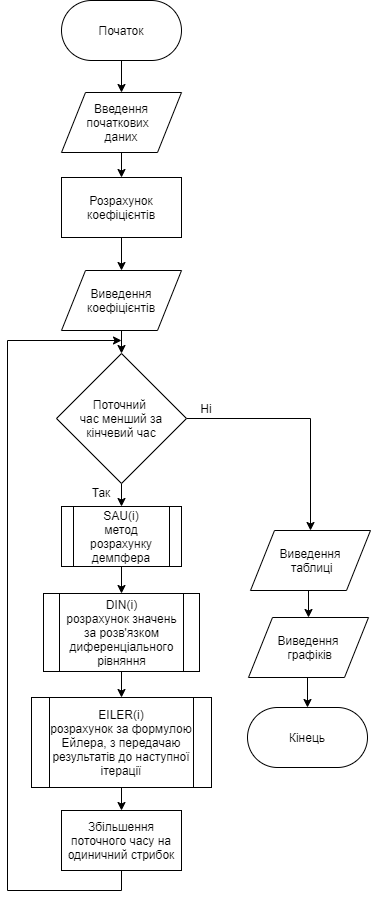


Аналітично показник визначається з такого співвідношення:



Характеристики стійкості та керованості літака, як це випливає з викладеного вище, можна визначити різними методами: шляхом льотних досліджень реального об’єкта, аналітичним шляхом, а також шляхом математичного моделювання руху літака на ПЕОМ.

**Алгоритм та програмна реалізація на Java**



import java.awt.\*;

import java.awt.event.\*;

import java.util.ArrayList;

import javax.swing.\*;

public class Lab extends JFrame{

public double S = 201.45, l = 37.55, G = 80000, Ix = 250000, Iy = 900000, V0 = 97.2, H0 = 500, p = 0.119, a = 338.36, g = 9.81,

Cy0 = -0.255, Cya = 5.78, mywy = -0.141, myb = -0.1518, myd = -0.071, Czb = -0.8136, mxd = -0.02, mxwx = -0.56, mxb = -0.1146,

mxdv = -0.07, mydv = 0, mywx = -0.026, mxwy = -0.151, Czd = -0.16; //Початкові дані

public double a1, a2, a3, a4, a5, a6, a7, b1, b2, b3, b4, b5, b6, b7, C6, abal, Cybal, X; //Коефіцієнти

public TextField Ta1, Ta2, Ta3, Ta4, Ta5, Ta6, Ta7, Tb1, Tb2, Tb3, Tb4, Tb5, Tb6, Tb7, TC6, Tabal, TCybal, TX;

public double T = 0, DT = 0.01, TD = 0, TF = 30, DD = 0.5; //Часові характеристики

public Button TT, TDN, TDE, Trus\_, Tkren\_, Tkov\_, Trus, Tkren, Tkov, TZ;

public double kwx = 1.5, Twx = 1.6, des = 0, kwy = 2.5, Twy = 2.5, dnp = 10; //Данні для демпера

public int switcher = 1; //Режим демперу

TextField Table[][]; //Таблиця

double[] min = {0,0,0,0,0,0}; //Мінімальні значення характеристик

double[] max = {0,0,0,0,0,0}; //Максимальне значення характеристик

Scrollbar SB; //Лінія прокрутки таблиці

Graphics graph; //Графіка вікна

public class Data{ //Клас, що збереігає дані на одиницю часу

public double[] Y = new double[8];

public double[] X = new double[8];

public double DN, DE;

public Data(double Y1, double Y2, double Y3, double Y4, double Y5, double Y6, double Y7, double Y8) { //Запис даних

Y[0] = Y1;

Y[1] = Y2;

Y[2] = Y3;

Y[3] = Y4;

Y[4] = Y5;

Y[5] = Y6;

Y[6] = Y7;

Y[7] = Y8;

}

}

public ArrayList<Data> Database = new ArrayList(); //Динамічний масив класу Data

public Lab(String s){ //Конструкот - основний алгоритм програми

super(s);

setLayout(null);

setSize(1545, 830);

setVisible(true);

this.setLocation(0, 0);

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

this.setResizable(false);

graph = getGraphics();

//Створення полів виводу та таблиці

Ta1 = CreatCTF("", 60, 520);

Ta2 = CreatCTF("", 240, 520);

Ta3 = CreatCTF("", 420, 520);

Ta4 = CreatCTF("", 600, 520);

Ta5 = CreatCTF("", 60, 570);

Ta6 = CreatCTF("", 240, 570);

Ta7 = CreatCTF("", 420, 570);

Tb1 = CreatCTF("", 600, 570);

Tb2 = CreatCTF("", 60, 620);

Tb3 = CreatCTF("", 240, 620);

Tb4 = CreatCTF("", 420, 620);

Tb5 = CreatCTF("", 600, 620);

Tb6 = CreatCTF("", 60, 670);

Tb7 = CreatCTF("", 240, 670);

TC6 = CreatCTF("", 420, 670);

TX = CreatCTF("",600, 670);

TCybal = CreatCTF("", 200, 720);

Tabal = CreatCTF("", 460, 720);

TT = CreateBT("T", 5, 50, 40, 40);

TDN = CreateBT("δн", 45, 50, 60, 40);

TDE = CreateBT("δe", 105, 50, 60, 40);

Trus\_ = CreateBT("Шв. ψ", 165, 50, 80, 40);

Tkren\_ = CreateBT("Шв. γ", 245, 50, 80, 40);

Tkov\_ = CreateBT("Шв. β", 325, 50, 80, 40);

Trus = CreateBT("ψ", 405, 50, 80, 40);

Tkren = CreateBT("γ", 485, 50, 80, 40);

Tkov = CreateBT("β", 565, 50, 80, 40);

Tkov = CreateBT("Z", 645, 50, 60, 40);

Table = new TextField[13][10];

for(int i = 0; i < 13; i++) {

for(int j = 0; j < 10; j++) {

Table[i][j] = new TextField("");

Table[i][j].setFont(new Font("TimesRoman", Font.PLAIN, 16));

Table[i][j].setEditable(false);

add(Table[i][j]);

}

Table[i][0].setBounds(5, 90+i\*25, 40, 25);

Table[i][1].setBounds(45, 90+i\*25, 60, 25);

Table[i][2].setBounds(105, 90+i\*25, 60, 25);

Table[i][3].setBounds(165, 90+i\*25, 80, 25);

Table[i][4].setBounds(245, 90+i\*25, 80, 25);

Table[i][5].setBounds(325, 90+i\*25, 80, 25);

Table[i][6].setBounds(405, 90+i\*25, 80, 25);

Table[i][7].setBounds(485, 90+i\*25, 80, 25);

Table[i][8].setBounds(565, 90+i\*25, 80, 25);

Table[i][9].setBounds(645, 90+i\*25, 60, 25);

}

SB = new Scrollbar(Scrollbar.VERTICAL);

SB.setBounds(705, 50, 20, 365);

SB.setBackground(Color.GRAY);

SB.setMinimum(1);

add(SB);

//Розрахунок коефіціентів

double m = G/g;

Cybal = 2\*G/(S\*p\*Math.pow(V0, 2));

TCybal.setText(Double.toString(Cybal));

abal = 57.3\*((Cybal - Cy0)/Cya);

Tabal.setText(Double.toString(abal));

a1 = (-mywy/Iy)\*S\*Math.pow(l, 2)\*((p\*V0)/4);

Ta1.setText(Double.toString(a1));

a2 = (-myb/Iy)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Ta2.setText(Double.toString(a2));

a3 = (-myd/Iy)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Ta3.setText(Double.toString(a3));

a4 = (-Czb/m)\*S\*((p\*V0)/2);

Ta4.setText(Double.toString(a4));

a5 = (-mxd/Ix)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Ta5.setText(Double.toString(a5));

a6 = (-mxwy/Ix)\*S\*Math.pow(l, 2)\*(p\*V0)/4;

Ta6.setText(Double.toString(a6));

a7 = (-Czd/m)\*S\*((p\*V0)/2);

Ta7.setText(Double.toString(a7));

b1 = (-mxwx/Ix)\*S\*Math.pow(l, 2)\*((p\*V0)/4);

Tb1.setText(Double.toString(b1));

b2 = (-mxb/Ix)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Tb2.setText(Double.toString(b2));

b3 = (-mxdv/Ix)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Tb3.setText(Double.toString(b3));

b4 = (g/V0)\*Math.cos(abal/57.3);

Tb4.setText(Double.toString(b4));

b5 = (-mydv/Iy)\*S\*l\*((p\*Math.pow(V0, 2))/2);

Tb5.setText(Double.toString(b5));

b6 = (-mywx/Iy)\*S\*Math.pow(l, 2)\*(p\*V0)/4;

Tb6.setText(Double.toString(b6));

b7 = Math.sin(abal/57.3);

Tb7.setText(Double.toString(b7));

C6 = V0/57.3;

TC6.setText(Double.toString(C6));

X = (mxb\*Iy)/(myb\*Ix\*Math.sqrt(1-Math.pow(mxwx/Ix, 2)\*Iy\*S\*Math.pow(l, 2)\*(p/(4\*myb))));

TX.setText(Double.toString(X));

//Розрахунок масиву Database

Database.add(new Data(0, 0, 0, 0, 0, 0, 0, 0));

while(T<=TF+TD) {

int i = (int)(Math.round(T/DT));

SAU(i);

DIN(i);

EILER(i);

T += DT;

CheckMaxMin(i);

}

//Виведення таблиці

for(int i = 0; i < 13; i++) {

Table[i][0].setText(Double.toString((i+1)\*DD));

Table[i][1].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).DN));

Table[i][2].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).DE));

Table[i][3].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).X[0]));

Table[i][4].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).X[2]));

Table[i][5].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).X[4]));

Table[i][6].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).Y[0]));

Table[i][7].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).Y[2]));

Table[i][8].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).Y[4]));

Table[i][9].setText(Double.toString(Database.get((i+1)\*(int)(DD/DT)).Y[5]));

}

//Створення події для смуги прокрутки

SB.setMaximum((int)(TF/DD)-2);

SB.setValue(1);

SB.addAdjustmentListener(new TableScroll());

//Створення графіків

CreateGraphicX(max[0], min[0], 780, 60, 350, 220, 0, 1);

CreateGraphicX(max[1], min[1], 780, 310, 350, 220, 2, 1);

CreateGraphicX(max[2], min[2], 780, 560, 350, 220, 4, 1);

CreateGraphicX(max[3], min[3], 1180, 60, 350, 220, 0, 0);

CreateGraphicX(max[4], min[4], 1180, 310, 350, 220, 2, 0);

CreateGraphicX(max[5], min[5], 1180, 560, 350, 220, 4, 0);

}

public void SAU(int i) { //Розрахунок демпферу

double ded = 0, dnd = 0;

switch (switcher) {

case 1:

ded = 0;

dnd = 0;

break;

case 2:

ded = kwx\*Database.get(i).Y[3];

dnd = kwy\*Database.get(i).Y[1];

break;

case 3:

ded = kwx\*Database.get(i).Y[3] - Database.get(i).Y[6]/Twx;

dnd = kwy\*Database.get(i).Y[1] - Database.get(i).Y[7]/Twy;

break;

}

if(T >= 1+DT)

dnp = 0;

Database.get(i).X[6] = ded;

Database.get(i).X[7] = dnd;

Database.get(i).DE = ded + des;

Database.get(i).DN = dnd + dnp;

}

public void DIN(int i) { //Розрахунок диф рівнянь

Database.get(i).X[0] = Database.get(i).Y[1];

Database.get(i).X[1] = -a1\*Database.get(i).Y[1] - b6\*Database.get(i).Y[3] - a2\*Database.get(i).Y[4] - a3\*Database.get(i).DN - b5\*Database.get(i).DE;

Database.get(i).X[2] = Database.get(i).Y[3];

Database.get(i).X[3] = -a6\*Database.get(i).Y[1] - b1\*Database.get(i).Y[3] - b2\*Database.get(i).Y[4] - a5\*Database.get(i).DN - b3\*Database.get(i).DE;

Database.get(i).X[4] = Database.get(i).Y[1] + b7\*Database.get(i).Y[3] + b4\*Database.get(i).Y[2] - a4\*Database.get(i).Y[4] - a7\*Database.get(i).DN;

Database.get(i).X[5] = -C6\*(Database.get(i).Y[0] - Database.get(i).Y[4]);

}

public void EILER(int i) { //Формула Ейлера

double[] YN = new double[8];

for(int j = 0; j < 8; j++)

YN[j] = Database.get(i).Y[j] + Database.get(i).X[j]\*DT;

Database.add(new Data(YN[0], YN[1], YN[2], YN[3], YN[4], YN[5], YN[6], YN[7]));

}

public void CheckMaxMin(int j) { //Перевірка максимального та минимального значення необхідних значень

for(int i = 0; i < 3; i++) {

if(Database.get(j).X[i\*2] > max[i])

max[i] = Database.get(j).X[i\*2];

if(Database.get(j).X[i\*2] < min[i])

min[i] = Database.get(j).X[i\*2];

if(Database.get(j).Y[i\*2] > max[i+3])

max[i+3] = Database.get(j).Y[i\*2];

if(Database.get(j).Y[i\*2] < min[i+3])

min[i+3] = Database.get(j).Y[i\*2];

}

}

public void paint (Graphics gr) { //Графічна частина інтерфейсу

gr.setColor(Color.lightGray);

gr.fillRect(0, 0, 1545, 830);

gr.setColor(Color.BLACK);

gr.drawLine(0, 500, 735, 500);

gr.drawLine(735, 0, 735, 830);

gr.setFont(new Font("TimesRoman", Font.PLAIN, 25));

gr.drawString("Таблиця даних", 280, 60);

gr.drawString("Графіки", 1100, 50);

gr.drawString("Розрахунок коефіцієнтів", 220, 530);

gr.setFont(new Font("TimesRoman", Font.PLAIN, 18));

gr.drawString("a1 =", 20, 570);

gr.drawString("a2 =", 200, 570);

gr.drawString("a3 =", 380, 570);

gr.drawString("a4 =", 560, 570);

gr.drawString("a5 =", 20, 620);

gr.drawString("a6 =", 200, 620);

gr.drawString("a7 =", 380, 620);

gr.drawString("b1 =", 560, 620);

gr.drawString("b2 =", 20, 670);

gr.drawString("b3 =", 200, 670);

gr.drawString("b4 =", 380, 670);

gr.drawString("b5 =", 560, 670);

gr.drawString("b6 =", 20, 720);

gr.drawString("b7 =", 200, 720);

gr.drawString("C6 =", 380, 720);

gr.drawString("χ =", 560, 720);

gr.drawString("α =", 150, 770);

gr.drawString("С =", 400, 770);

gr.setFont(new Font("TimesRoman", Font.PLAIN, 12));

gr.drawString("бал", 161, 775);

gr.drawString("yбал", 412, 775);

}

public TextField CreatCTF(String value, int x, int y) { //Метод, що створює вже готове текстове поле

TextField TF = new TextField(value);

TF.setBounds(x, y, 100, 25);

TF.setEditable(false);

TF.setFont(new Font("TimesRoman", Font.PLAIN, 18));

add(TF);

return TF;

}

public Button CreateBT(String value, int x, int y, int w, int h) { //Метод, що створює перший ряд таблиці

Button BT = new Button(value);

BT.setBounds(x, y, w, h);

BT.setFont(new Font("TimesRoman", Font.PLAIN, 18));

add(BT);

return BT;

}

public void CreateGraphicX(double max, double min, int x, int y, int w, int h, int mas, int XY) { //Метод, що створює графік

int bub;

int bx, lx = x;

int by, ly = (int)map(Database.get(0).Y[mas], min, max, y+h, y);

if (XY == 1) {

ly = (int)map(Database.get(0).X[mas], min, max, y+h, y);

}

graph.setColor(Color.white);

graph.fillRect(x, y, w, h);

graph.setFont(new Font("TimesRoman", Font.PLAIN, 20));

double start, end, delta, mod = Math.abs(max - min);//Масштабування

if (mod > 10) {

delta = 5;

start = Math.ceil(min\*2/10)\*10/2;

end = Math.floor(max);

}

else if (mod > 7) {

delta = 2;

start = Math.ceil(min/2)\*2;

end = Math.floor(max/2)\*2;

}

else if (mod > 3) {

delta = 1;

start = Math.ceil(min);

end = Math.floor(max);

}

else if (mod > 1) {

delta = 0.5;

start = Math.ceil(min\*2)/2;

end = Math.floor(max\*2)/2;

}

else if (mod > 0.3) {

delta = 0.1;

start = Math.ceil(min\*10)/10;

end = Math.floor(max\*10)/10;

}

else if (mod > 0.15) {

delta = 0.05;

start = Math.ceil(min\*20)/20;

end = Math.floor(max\*20)/20;

}

else {

delta = 0.02;

start = Math.ceil(min\*500)/500;

end = Math.floor(max\*500)/500;

}

for (double i = start; i <= end; i+=delta) { //Лінії паралельні осі OX

bub = (int)map(i, min, max, y+h, y);

if((int)Math.floor(i\*100) != 0) {

graph.setColor(Color.black);

if(i % 1 == 0)

graph.drawString(Integer.toString((int)Math.round(i\*100)/100), x-30, bub+10);

else if (((double)Math.round(i\*100)/100)\*10 % 1 == 0)

graph.drawString(Double.toString((double)Math.round(i\*100)/100), x-35, bub+10);

else

graph.drawString(Double.toString((double)Math.round(i\*100)/100), x-45, bub+10);

graph.setColor(Color.lightGray);

graph.drawLine(x, bub, x+w, bub);

}

}

for (int i = 5; i <= TF; i += 5) { //Лінії паралельні осі OY

bub = (int)map(i, 0, TF, x, x+w);

graph.setColor(Color.black);

graph.drawString(Integer.toString(i), bub-15, (int)map(0, min, max, y+h, y)+25);

graph.setColor(Color.lightGray);

graph.drawLine(bub, y, bub, y+h);

}

graph.setColor(Color.black);

graph.fillRect(x, y, 2, h);

bub = (int)map(0, min, max, y+h, y);

graph.fillRect(x, bub, w, 2);

graph.setFont(new Font("TimesRoman", Font.PLAIN, 30));

graph.drawString("^", x-6, y+20);

graph.setFont(new Font("TimesRoman", Font.PLAIN, 30));

graph.drawString(">", x+w-15, bub+11);

graph.setFont(new Font("TimesRoman", Font.PLAIN, 20));

graph.drawString("T", x+w/2, y+h+20);

switch(mas) {

case 0:

graph.drawString("ψ", x-20, bub+10);

break;

case 2:

graph.drawString("γ", x-20, bub+10);

break;

case 4:

graph.drawString("β", x-20, bub+10);

break;

}

if (XY == 1) {

graph.setFont(new Font("TimesRoman", Font.PLAIN, 40));

graph.drawString(".", x-19, bub-5);

}

graph.setColor(Color.red);

for(double i = 2\*DT; i < TF-0.1; i+=DT) {//Виведення графіку за допомогою ліній з попередньої точки до наступної

bx = (int)map(i, 0, TF, x, x+w);

if (XY == 1)

by = (int)map((Database.get((int)(i/DT)).X[mas]), min, max, y+h, y);

else

by = (int)map((Database.get((int)(i/DT)).Y[mas]), min, max, y+h, y);

graph.drawLine(lx, ly, bx, by);

graph.drawLine(lx, ly, bx+1, by);

graph.drawLine(lx, ly, bx, by+1);

graph.drawLine(lx+1, ly, bx, by);

graph.drawLine(lx, ly+1, bx, by);

graph.drawLine(lx, ly, bx-1, by);

graph.drawLine(lx, ly, bx, by-1);

graph.drawLine(lx-1, ly, bx, by);

graph.drawLine(lx, ly-1, bx, by);

lx = bx;

ly = by;

}

}

public double map(double input, double INmin, double INmax, double OUTmin, double OUTmax) { //Метод, що конвертує значення на пиксельну систему координат, знаючи максимальні та мінімальні значення

return (input-INmin)\*(OUTmax-OUTmin)/(INmax-INmin)+OUTmin;

}

public class TableScroll implements AdjustmentListener { //Слухач події для смуги прокрутки

public void adjustmentValueChanged(AdjustmentEvent e) {

int bub = SB.getValue();

for(int i = bub; i < bub+13; i++) {

Table[i-bub][0].setText(Double.toString((i)\*DD));

Table[i-bub][1].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).DN));

Table[i-bub][2].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).DE));

Table[i-bub][3].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).X[0]));

Table[i-bub][4].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).X[2]));

Table[i-bub][5].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).X[4]));

Table[i-bub][6].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).Y[0]));

Table[i-bub][7].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).Y[2]));

Table[i-bub][8].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).Y[4]));

Table[i-bub][9].setText(Double.toString(Database.get((i)\*(int)(DD/DT)).Y[5]));

}

}

}

public static void main(String[] args) {

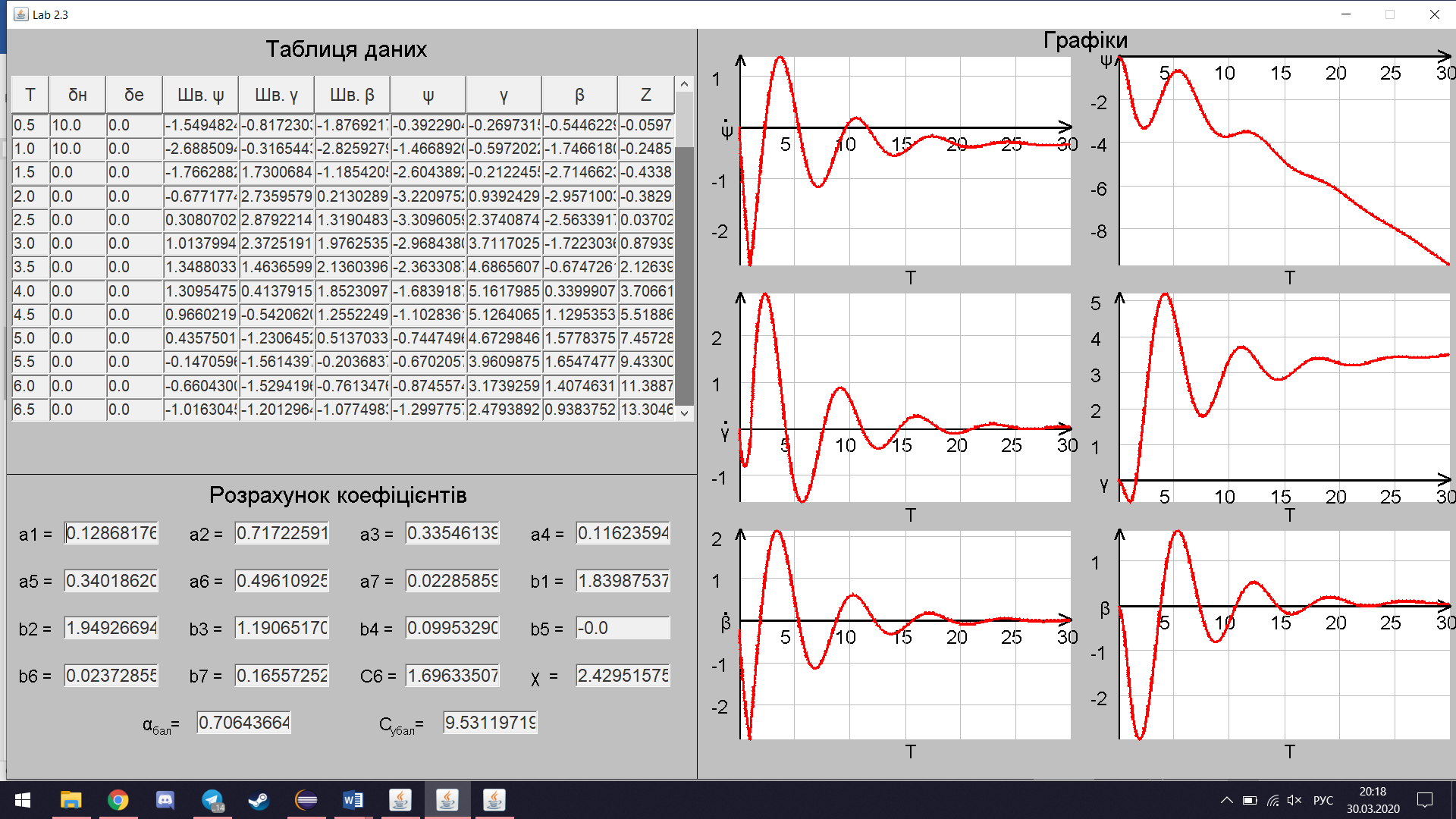
new Lab("Lab 2.3");

}

}

**Результати**

**Загальний вигляд програми**



**Чисельні результати досліджень**



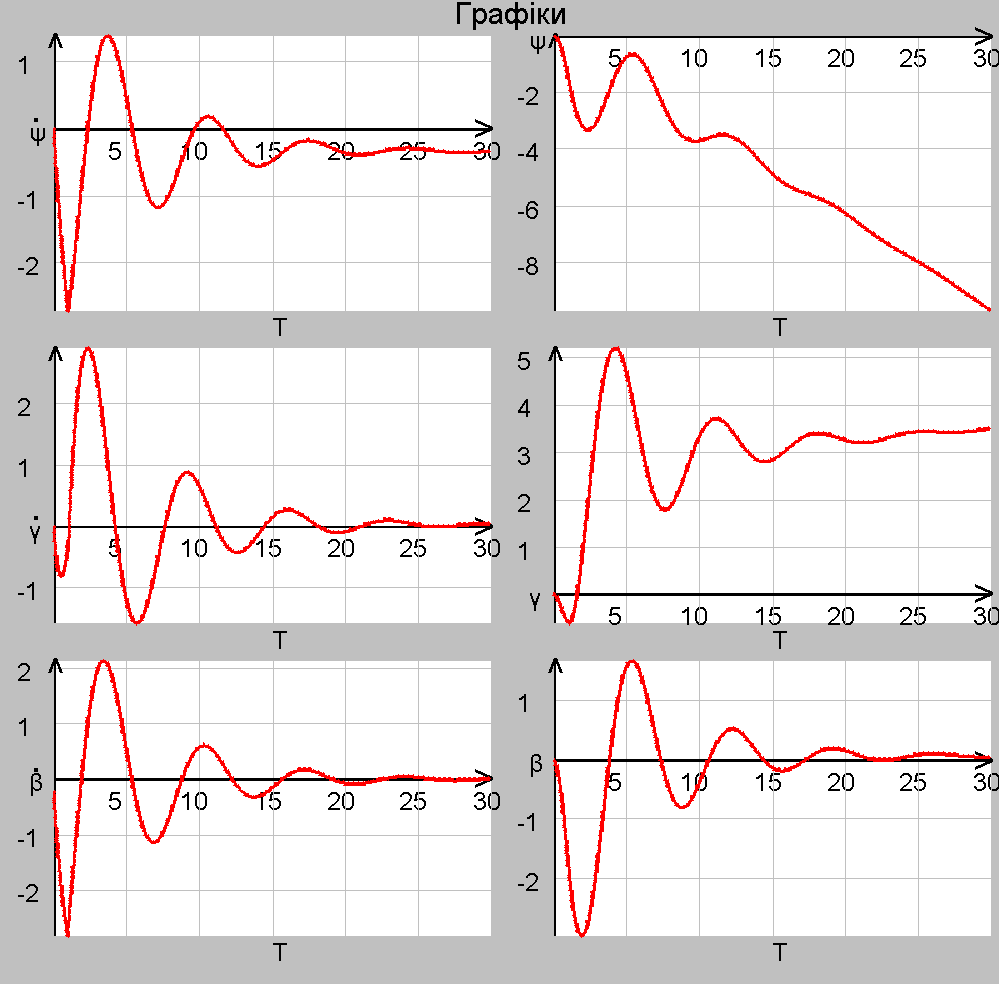
**Результати графічних досліджень**

Перший стовпець – кутові швидкості, другий – кути;

Перший ряд – рискання, другий – крен, третій – ковзання;

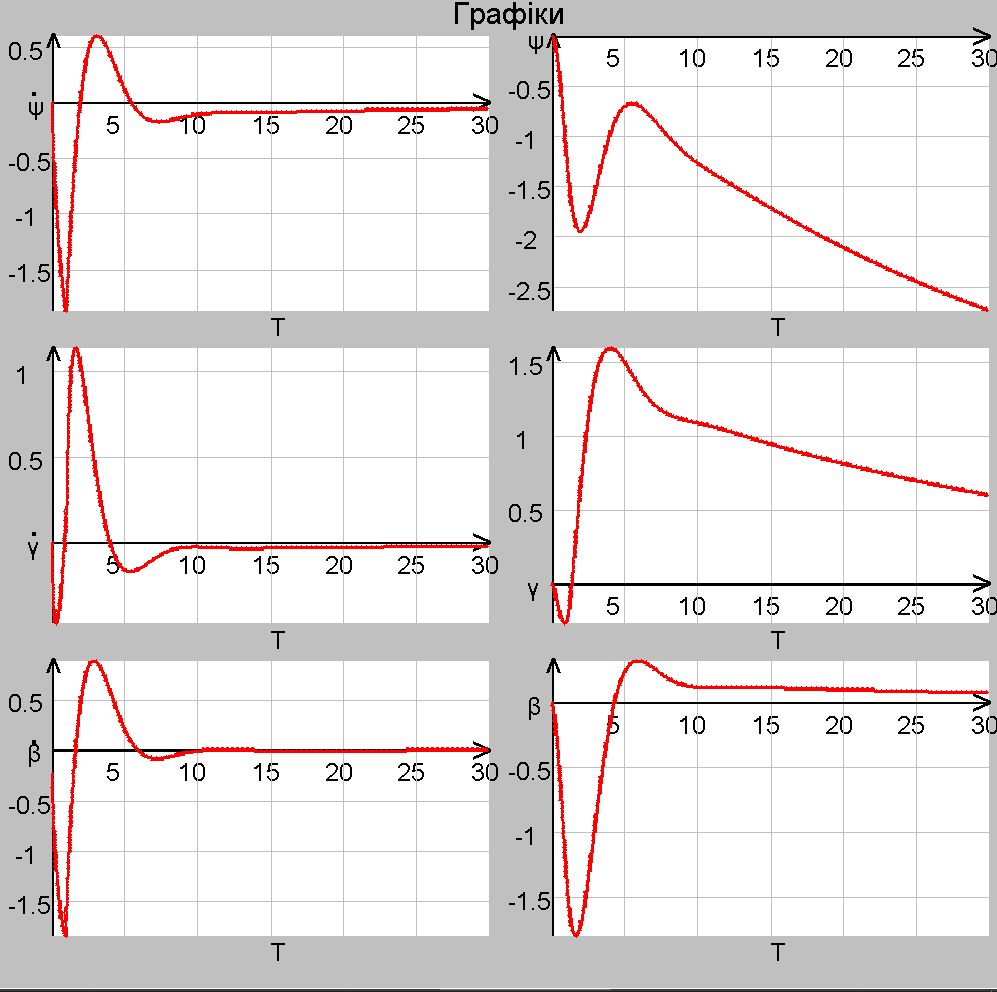
Демпфер вимкнено, = 100 , = 00





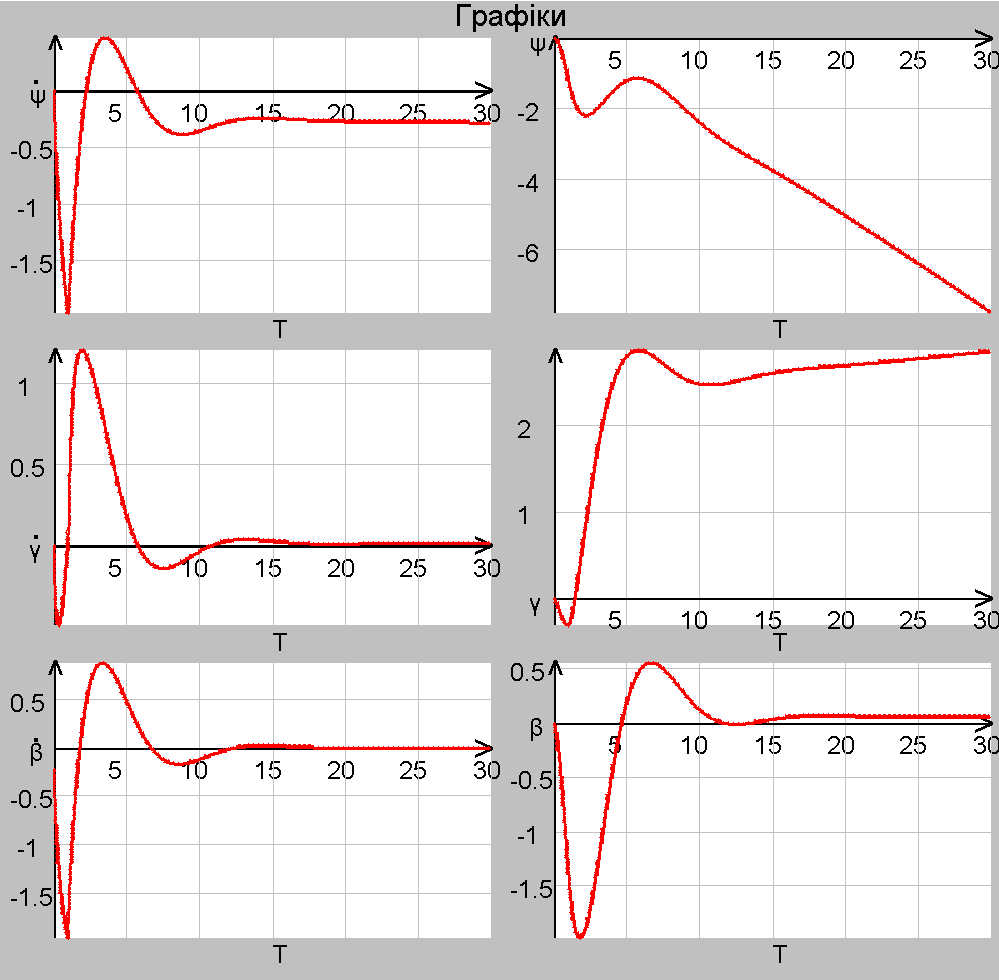
Демпфер другого закону управління, = 100 , = 00





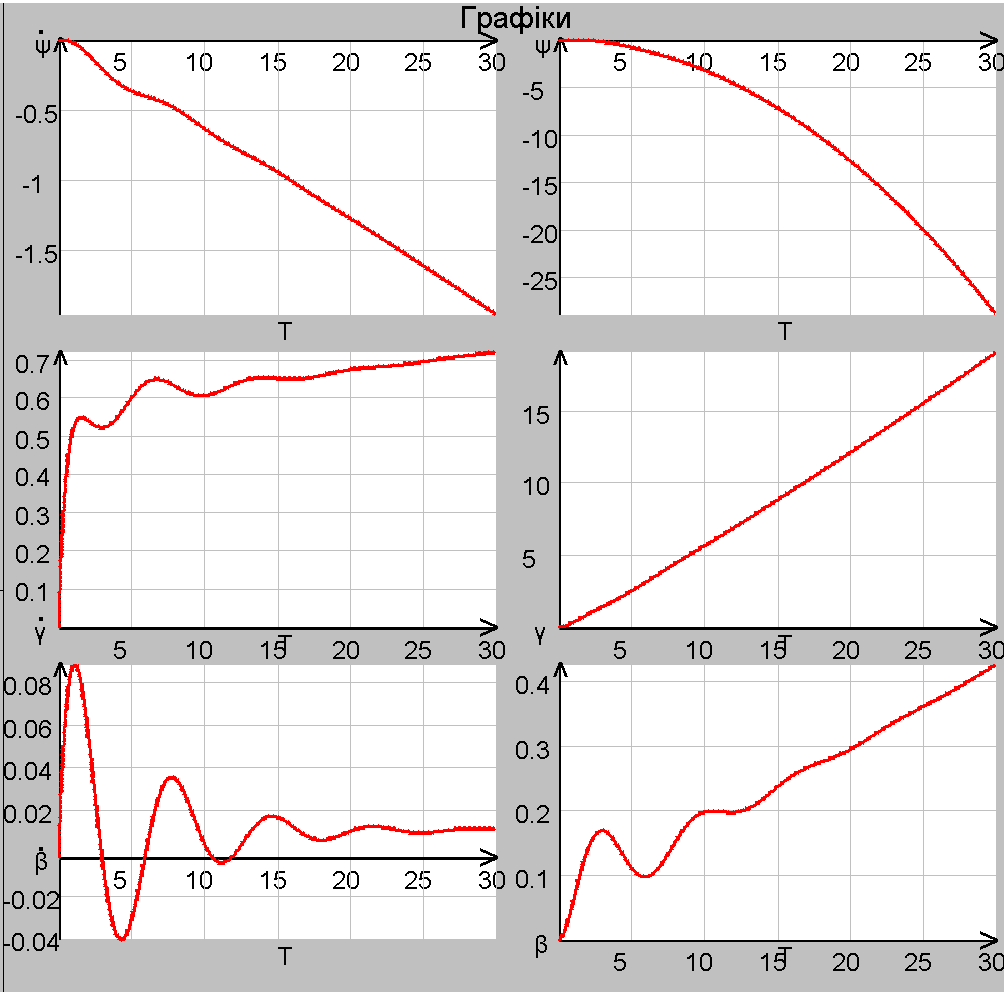
Демпфер третього закону управління, = 100 , = 00





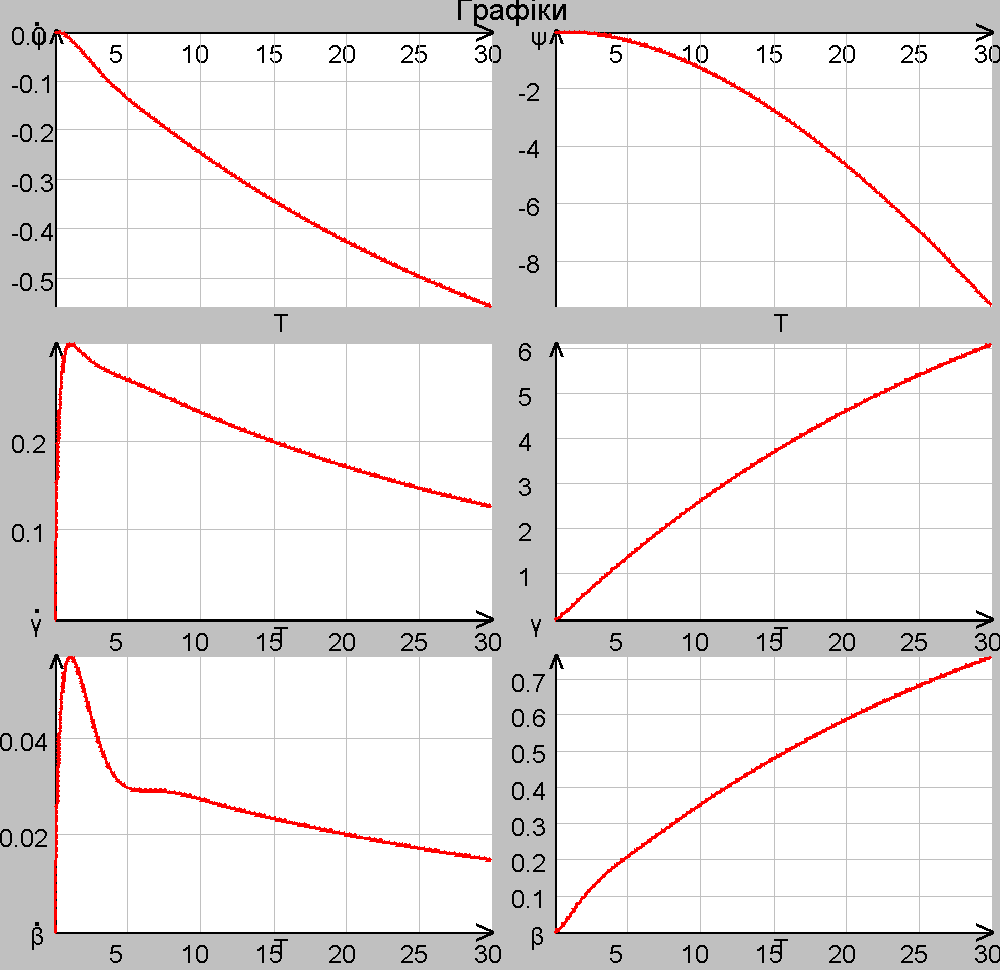
Демпфер вимкнено, = 00 , = -10





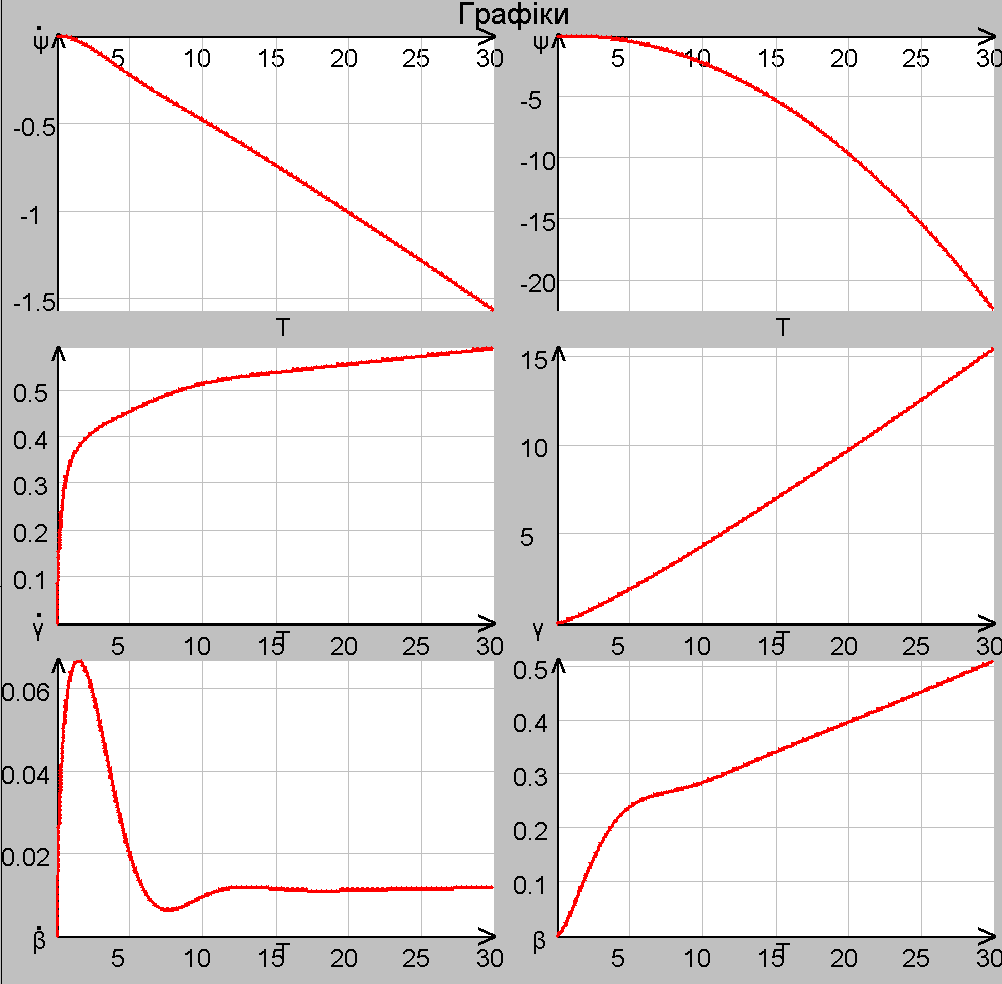
Демпфер другого закону управління, = 00 , = -10





Демпфер третього закону управління, = 00 , = -10





**Статичні та динамічні характеристики бічної керованості літака**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Закон управління  демпферів | *t*пп, c | *Тβ* , с | |  | | | , град/с | |
| Модель | Реальн. об’єкт | Модель | Реальн. об’єкт | Модель | Аналіт. розрах. | Реальн. об’єкт | Модель |
| 1 | 27 | 7-8 | 6.75 | 1 | 2.22 | 2.42 | 0.7163 | 0.72 |
| 2 | 11 | 7-8 | 7.5 | 1 | 1.71 | 2.42 | 0.7163 | 0.29 |
| 3 | 17 | 7-8 | 8.5 | 1 | 2.01 | 2.42 | 0.7163 | 0.59 |

**Висновки про стійкість руху моделі літака в бічній площині**

За результатами графічних досліджень можна стверджувати, що система є стійка, оскільки графіки швидкості кута ковзання при наданні імпульсу прямують до певного числа: до 0 у випадку з відхиленням руля напрямку на 1 секунду, та до 0.1 у випадку з «дачею» елеронів. Усе це супроводжується зміною кута ковзання: при відхилення руля напрямку на 1 секунду кінцеве значення кута ковзання змінюється тільки на протязі секунди і прямує приблизно до 0.1, при «дачі» елеронів кут ковзання змінюється постійно відповідно до зміни швидкості кута ковзання.

**Висновки про ступінь динамічної подібності розробленої цифрової математичної моделі і реального об'єкта**

За результатами таблиці статичних та динамічних характеристик бічної керованості літака можна стверджувати, що система є достатньо подібною, хоча і має певні відхилення. Наприклад показник при аналітичних розрахунках майже в 2.5 раз більше ніж реальний, як приблизно і показники моделі.



**Висновок про вплив на якість перехідних процесів при моделюванні руху літака в бічній площині виду закону управління СШУ**

За результатами графічних та табличних досліджень можна стверджувати, що при вимкненому демпфері значення графіків найбільш точні до реальних та аналітичних розрахунків характеристик бічної керованості, але мають найбільший час гасіння перехідних процесів та найбільшу кількість коливань. Демпфер другого закону має найменший час гасіння перехідних процесів, але найменш точний до реальних та аналітичних розрахунків характеристик бічної керованості. Демпфер третього закону має як гарні показники часу гасіння перехідного процесу, так і точність до реальних та аналітичних розрахунків характеристик бічної керованості.

**Висновки до лабораторної роботи:** було вивчено динамічні та статичні характеристики стійкості та керованості руху літака в бічній площині; здійснено моделювання руху літака в бічній площині на ПЕОМ і досліджено характеристики його стійкості та керованості.