**Приложение А**

Ниже представлены результаты интегрирования траектории. Координаты цели приняты: х = 12100 м, z = 2742 м.

Таблица А.1 – Результаты интегрирования

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| t,s | x,m | y,m | z,m | Vx,m/s | Vy,m/s | Vz,m/s | V,m/s | ϑ,grad | ψ,grad | γ,grad | ωx,grad/s | ωy,grad/s | ωz,grad/s |
| 0.00 | 0.00 | 7750.00 | 0.00 | 730.51 | -952.02 | 0.00 | 1200.00 | -52.500 | 0.000 | 0.000 | 0.00000 | 0.00000 | 0.00000 |
| 1.00 | 749.98 | 6835.78 | 46.05 | 764.46 | -876.61 | 93.08 | 1166.84 | -46.639 | -10.336 | -0.096 | -0.08926 | -0.05154 | 0.07189 |
| 2.00 | 1522.47 | 5997.19 | 184.19 | 776.76 | -801.06 | 182.15 | 1130.59 | -42.571 | -16.795 | -0.524 | 0.03244 | -0.06436 | 0.06942 |
| 3.00 | 2309.02 | 5233.24 | 373.89 | 795.79 | -727.16 | 192.72 | 1095.07 | -38.933 | -14.771 | -2.480 | -0.03756 | -0.39594 | 0.05152 |
| 4.00 | 3110.77 | 4542.00 | 567.80 | 806.07 | -655.89 | 194.60 | 1057.26 | -35.648 | -13.869 | 0.955 | -0.11134 | -0.42221 | 0.05886 |
| 5.00 | 3917.80 | 3920.13 | 763.06 | 806.92 | -588.61 | 193.94 | 1017.44 | -32.625 | -12.368 | 0.654 | 0.00400 | -0.27606 | 0.05253 |
| 6.00 | 4721.31 | 3363.16 | 957.51 | 799.25 | -526.23 | 192.98 | 976.19 | -29.864 | -12.352 | 0.476 | 0.02918 | 0.06508 | 0.04578 |
| 7.00 | 5513.87 | 2865.88 | 1149.13 | 784.83 | -469.27 | 190.20 | 934.00 | -27.389 | -13.924 | -0.849 | 0.03997 | 0.09274 | 0.04202 |
| 8.00 | 6289.33 | 2422.73 | 1336.66 | 765.22 | -417.95 | 184.90 | 891.31 | -25.180 | -13.640 | -0.501 | 0.03909 | -0.03849 | 0.03607 |
| 9.00 | 7042.82 | 2028.11 | 1518.89 | 741.17 | -372.20 | 179.31 | 848.54 | -23.228 | -13.627 | -0.003 | -0.09442 | 0.00728 | 0.03195 |
| 10.00 | 7770.63 | 1676.54 | 1694.92 | 714.01 | -331.81 | 172.67 | 806.05 | -21.521 | -13.575 | 0.001 | 0.00238 | -0.00381 | 0.02775 |
| 11.00 | 8470.07 | 1362.81 | 1864.08 | 684.56 | -296.45 | 165.56 | 764.14 | -20.046 | -13.600 | 0.000 | 0.00018 | -0.00099 | 0.02380 |
| 12.00 | 9139.24 | 1082.11 | 2025.91 | 653.56 | -265.69 | 158.06 | 722.99 | -18.791 | -13.595 | 0.000 | 0.00008 | -0.00063 | 0.02009 |
| 13.00 | 9776.83 | 830.05 | 2180.12 | 621.49 | -239.08 | 150.31 | 682.65 | -17.741 | -13.595 | 0.000 | 0.00009 | -0.00060 | 0.01662 |
| 14.00 | 10382.00 | 602.74 | 2326.47 | 588.62 | -216.10 | 142.36 | 642.99 | -16.883 | -13.596 | 0.000 | 0.00007 | -0.00047 | 0.01340 |
| 15.00 | 10953.80 | 396.81 | 2464.77 | 554.85 | -196.22 | 134.20 | 603.63 | -16.203 | -13.596 | 0.000 | 0.00005 | -0.00035 | 0.01040 |
| 16.00 | 11491.20 | 209.47 | 2594.74 | 519.58 | -178.80 | 125.67 | 563.67 | -15.688 | -13.597 | 0.000 | 0.00004 | -0.00026 | 0.00764 |
| 17.00 | 11991.80 | 38.68 | 2715.84 | 480.86 | -162.92 | 116.31 | 520.86 | -15.322 | -13.598 | 0.000 | 0.00003 | -0.00018 | 0.00523 |
| 17.24 | 12106.10 | 0.00 | 2743.47 | 470.46 | -159.15 | 113.79 | 509.52 | -15.253 | -13.598 | 0.000 | 0.00003 | -0.00016 | 0.00477 |

Продолжение таблицы А.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| t,s | α,grad | β,grad | δT, grad | δN, grad | δE, grad | r,m |
| 0.00 | 0.00000 | 0.00000 | 15.00 | -15.00 | 0.00 | 14628 |
| 1.00 | 2.10743 | -2.24282 | 15.00 | -15.00 | -15.00 | 13521 |
| 2.00 | 2.57786 | -2.56168 | 15.00 | -15.00 | -15.00 | 12426 |
| 3.00 | 2.64020 | -0.98061 | 15.00 | -2.80 | -15.00 | 11352 |
| 4.00 | 2.69893 | -0.18732 | 15.00 | -3.40 | 15.00 | 10304 |
| 5.00 | 2.71607 | 0.96562 | 15.00 | -3.99 | 15.00 | 9286 |
| 6.00 | 2.75268 | 1.05273 | 15.00 | -0.23 | 15.00 | 8303 |
| 7.00 | 2.76797 | -0.30186 | 15.00 | 0.98 | -15.00 | 7357 |
| 8.00 | 2.78368 | -0.07346 | 15.00 | -0.38 | -15.00 | 6450 |
| 9.00 | 2.78949 | -0.02368 | 15.00 | 0.10 | -15.00 | 5584 |
| 10.00 | 2.78799 | 0.01822 | 15.00 | -0.03 | 1.87 | 4759 |
| 11.00 | 2.78044 | -0.00348 | 15.00 | -0.01 | 0.04 | 3975 |
| 12.00 | 2.76965 | 0.00115 | 15.00 | -0.01 | 0.00 | 3233 |
| 13.00 | 2.75965 | 0.00097 | 15.00 | -0.01 | 0.00 | 2530 |
| 14.00 | 2.75609 | 0.00053 | 15.00 | -0.01 | 0.00 | 1868 |
| 15.00 | 2.76715 | 0.00017 | 15.00 | -0.01 | 0.00 | 1244 |
| 16.00 | 2.80673 | -0.00024 | 15.00 | -0.01 | 0.00 | 660 |
| 17.00 | 2.90634 | -0.00064 | 15.00 | -0.01 | 0.00 | 118 |
| 17.24 | 2.94788 | -0.00073 | 15.00 | -0.01 | 0.00 | 50 |

На рисунках А.1-А.14 представлены графические зависимости полученной траектории.

Рисунок А.1 – Траектория в проекции на вертикальную плоскость ОХУ

Рисунок А.2 – Траектория в проекции на горизонтальную плоскость ОХZ

Рисунок А.3 – Зависимость радиус-вектора до цели от времени полета

Рисунок А.4 – Зависимость угла атаки от времени полета

Рисунок А.5 – Зависимость угла скольжения от времени полета

Рисунок А.6 – Зависимость угла тангажа от времени полета

Рисунок А.7 – Зависимость угла рысканья от времени полета

Рисунок А.8 – Зависимость угла крена от времени полета

Рисунок А.9 – Зависимость угловой скорости по оси Х от времени полета

Рисунок А.10 – Зависимость угловой скорости по оси Y от времени полета

Рисунок А.11 – Зависимость угловой скорости по оси Z от времени полета

Рисунок А.12 – Зависимость угла отклонения рулей по каналу тангажа от времени полета

Рисунок А.13 – Зависимость угла отклонения рулей по каналу рысканья от времени полета

Рисунок А.14 – Зависимость угла отклонения рулей по каналу крена от времени полета

**Приложение Б**

В таблице Б.1 представлены результаты поиска зоны, в которой цель может быть поражена. Максимальный промах принят за 50 метров. На рисунке Б.1 представлен вид зоны.

Таблица Б.1 – Координаты полученной зоны и промах

|  |  |  |
| --- | --- | --- |
| xц, м | zц, м | r, м |
| 2492.7 | 0 | 49.915 |
| 2492.7 | 1888 | 49.915 |
| 2992.7 | 2276 | 49.68 |
| 3492.7 | 2541 | 50.229 |
| 3992.7 | 2765 | 49.154 |
| 4492.7 | 2974 | 49.91 |
| 4992.7 | 3215 | 49.375 |
| 5492.7 | 3505 | 49.589 |
| 5992.7 | 3836 | 49.948 |
| 6492.7 | 4224 | 49.105 |
| 6992.7 | 4681 | 49.366 |
| 7492.7 | 5220 | 49.667 |
| 7992.7 | 5861 | 49.58 |
| 8492.7 | 6642 | 49.029 |
| 8992.7 | 6510 | 50.067 |
| 9492.7 | 6103 | 49.297 |
| 9992.7 | 5596 | 49.384 |
| 10493 | 4985 | 49.074 |
| 10993 | 4250 | 49.381 |
| 11493 | 3270 | 49.873 |
| 11993 | 1830 | 49.754 |
| 12493 | 21 | 50.208 |
| 12493 | 0 | 50.208 |

Рисунок Б.1 – Зона поражения

**Приложение В**

Для выполнения поставленной задачи было разработано программное обеспечение на языке программирования С++ в среде Microsoft Visual Studio 2017. Интегрирование полета осуществляется методом Рунге Кутты 4‑го порядка. Листинг модуля для атмосферы опущен.

#include "stdafx.h"

#include <iostream>

#include <cmath>

#include <iomanip>

#include <fstream>

#include <vector>

#define \_USE\_MATH\_DEFINES

#include "math.h"

#include "GOST4401-81.h"

using namespace std;

ofstream txtfile2("2.txt");

double a[] = { 0.0,0.5,0.5,1.0 };

#define dtdt K[0][j]

#define dxgdt K[1][j]

#define dygdt K[2][j]

#define dzgdt K[3][j]

#define dVxgdt K[4][j]

#define dVygdt K[5][j]

#define dVzgdt K[6][j]

#define domegaxdt K[7][j]

#define domegaydt K[8][j]

#define domegazdt K[9][j]

#define droRGdt K[10][j]

#define dlamdaRGdt K[11][j]

#define dmuRGdt K[12][j]

#define dnuRGdt K[13][j]

#define t par[0]

#define xg par[1]

#define yg par[2]

#define zg par[3]

#define Vxg par[4]

#define Vyg par[5]

#define Vzg par[6]

#define omegax par[7]

#define omegay par[8]

#define omegaz par[9]

#define roRG par[10]

#define lamdaRG par[11]

#define muRG par[12]

#define nuRG par[13]

#define t\_RK parbuf[0]

#define xg\_RK parbuf[1]

#define yg\_RK parbuf[2]

#define zg\_RK parbuf[3]

#define Vxg\_RK parbuf[4]

#define Vyg\_RK parbuf[5]

#define Vzg\_RK parbuf[6]

#define omegax\_RK parbuf[7]

#define omegay\_RK parbuf[8]

#define omegaz\_RK parbuf[9]

#define roRG\_RK parbuf[10]

#define lamdaRG\_RK parbuf[11]

#define muRG\_RK parbuf[12]

#define nuRG\_RK parbuf[13]

// Объявление функций

double Cx(double M);

double Cy\_alpha(double mach);

double Cy\_delta(double mach, double alpha);

double Cz\_betta(double M, double betta);

double Cz\_delta(double M, double betta);

double mx\_omegax();

double mx\_delta(double q);

double mz\_omegaz(double M);

double mz\_alpha(double M);

double mz\_delta(double M, double alpha);

double my\_omegay(double M);

double my\_betta(double M);

double my\_delta(double M, double betta);

void B\_Matr(double fi, double hi, double(\*B)[3][3]);

void Control\_prog(double tang, double risk, double kren, double dfidt, double dhidt, double \*delta\_tang, double \*delta\_risk, double \*delta\_kren);

void MATR(double ro, double lamda, double mu, double nu, double(\*A)[3][3], double(\*AT)[3][3]);

const double g = 9.81;

double K[99][99], par[99], parbuf[99], par\_drob[99],

A[3][3] = { 0 }, AT[3][3] = { 0 }, B[3][3] = { 0 };

double tang, risk, kren, m, V0, q, M, r, V, X, Y, Z, Mx, My, Mz, Mstab, alpha, betta, alpha\_full, vx, vy, vz, vr, vfi, vhi,

delta\_tang, delta\_risk, delta\_kren, fi, hi, dfidt, dhidt, x\_t, y\_t, z\_t, Vx\_t = 0, Vy\_t = 0, Vz\_t = 0,

Ix = 170,

Iy = 640,

Iz = 640,

dm = 0.95,

Sm = M\_PI\*dm\*dm / 4.0,

L = 7,

dt = 0.01;

int main()

{

Atm\_GOST();

//Начало цикла для построения эллипса

//double dx\_t = 1000;

txtfile2 << "x\_t;z\_t;r" << endl;

double dz\_t; bool flag1 = 0, flag2 = 0;

//2800

for (x\_t = 2500; x\_t <= 12500; x\_t += 100) {

//if (flag2 == 1) break;

dz\_t = 1000;

for (z\_t = 1000; z\_t <= 50000; z\_t += dz\_t) {

y\_t = 0;

x\_t = 2500;

z\_t = 1985;

//Начальные условия

double N = 2,

tang = (-60 + 0.5\*N)\*M\_PI / 180.0;

risk = 0;

kren = 0;

m = 1500 - 5 \* N;

V0 = 1200;

dt = 0.01;

t = 0;

xg = 0;

yg = 10000 - 150 \* N;

zg = 0;

Vxg = V0\*cos(tang);

Vyg = V0\*sin(tang);

Vzg = 0;

omegax = 0;

omegay = 0;

omegaz = 0;

roRG = cos(risk\*0.5)\*cos(tang\*0.5)\*cos(kren\*0.5) - sin(risk\*0.5)\*sin(kren\*0.5)\*sin(tang\*0.5);

lamdaRG = sin(risk\*0.5)\*sin(tang\*0.5)\*cos(kren\*0.5) + cos(risk\*0.5)\*cos(tang\*0.5)\*sin(kren\*0.5);

muRG = sin(risk\*0.5)\*cos(tang\*0.5)\*cos(kren\*0.5) + cos(risk\*0.5)\*sin(tang\*0.5)\*sin(kren\*0.5);

nuRG = cos(risk\*0.5)\*sin(tang\*0.5)\*cos(kren\*0.5) - sin(risk\*0.5)\*cos(tang\*0.5)\*sin(kren\*0.5);

ofstream txtfile("1.txt");

txtfile << "t,s" << ";" << "x,m" << ";" << "y,m" << ";" << "z,m" << ";" << "Vx,m/s" << ";" << "Vy,m/s" << ";" << "Vz,m/s" << ";" << "V,m/s" << ";"

<< "tang,grad" << ";" << "risk,grad" << ";" << "kren,grad" << ";" << "omega\_x,grad/s" << ";" << "omega\_y,grad/s" << ";" << "omega\_z,grad/s" << ";"

<< "alpha,grad" << ";" << "betta,grad" << ";" << "deltaT, drad" << ";" << "deltaN, grag" << ";" << "deltaE, grad" << ";" << endl;

// Интегрирование

int step = 0;

for (;;) {

//cout << t << " " << r << endl;

// Расчет промежуточных параметров на шаге

MATR(roRG, lamdaRG, muRG, nuRG, &A, &AT);

tang = asin(2 \* (roRG\*nuRG + lamdaRG\*muRG));

risk = atan2(2 \* (roRG\*muRG - lamdaRG\*nuRG), roRG\*roRG + lamdaRG\*lamdaRG - nuRG\*nuRG - muRG\*muRG);

kren = atan2(2 \* (roRG\*lamdaRG - muRG\*nuRG), roRG\*roRG - lamdaRG\*lamdaRG - nuRG\*nuRG + muRG\*muRG);

r = pow((x\_t - xg)\*(x\_t - xg) + (y\_t - yg)\*(y\_t - yg) + (z\_t - zg)\*(z\_t - zg), 0.5),

V = pow(Vxg\*Vxg + Vyg\*Vyg + Vzg\*Vzg, 0.5),

q = fro(yg)\*V\*V\*0.5,

vx = Vxg\*AT[0][0] + Vyg\*AT[0][1] + Vzg\*AT[0][2],

vy = Vxg\*AT[1][0] + Vyg\*AT[1][1] + Vzg\*AT[1][2],

vz = Vxg\*AT[2][0] + Vyg\*AT[2][1] + Vzg\*AT[2][2],

alpha = -atan2(vy, vx);

betta = asin(vz / V);

alpha\_full = pow(alpha\*alpha + betta\*betta, 0.5);

fi = asin((y\_t - yg) / r);

hi = -atan2(z\_t - zg, x\_t - xg);

B\_Matr(fi, hi, &B);

vr = B[0][0] \* (Vx\_t - Vxg) + B[0][1] \* (Vy\_t - Vyg) + B[0][2] \* (Vz\_t - Vzg);

vfi = B[1][0] \* (Vx\_t - Vxg) + B[1][1] \* (Vy\_t - Vyg) + B[1][2] \* (Vz\_t - Vzg);

vhi = B[2][0] \* (Vx\_t - Vxg) + B[2][1] \* (Vy\_t - Vyg) + B[2][2] \* (Vz\_t - Vzg);

dfidt = vfi / r;

dhidt = vhi / (r\*cos(fi));

M = V / fa(yg);

if (yg > 500) Control\_prog(tang, risk, kren, dfidt, dhidt, &delta\_tang, &delta\_risk, &delta\_kren);

X = -Cx(M)\*q\*Sm;

Y = (Cy\_alpha(M)\*alpha + Cy\_delta(M, alpha)\*delta\_tang)\*q\*Sm;

Z = (Cz\_betta(M, betta)\*betta + Cz\_delta(M, betta)\*delta\_risk)\*q\*Sm;

Mstab = 0.5 \* dm \* 1000 \* delta\_kren;

Mx = (mx\_omegax()\*omegax\*L / V)\*q\*Sm\*L + Mstab;

My = (my\_omegay(M)\*omegay\*L / V + my\_betta(M)\*betta + my\_delta(M, betta)\*delta\_risk)\*q\*Sm\*L;

Mz = (mz\_omegaz(M)\*omegaz\*L / V + mz\_alpha(M)\*alpha + mz\_delta(M, alpha)\*delta\_tang)\*q\*Sm\*L;

// Дробление шага

if (yg < 0) {

dt \*= 0.5;

for (int i = 0; i <= 13; i++) par[i] = par\_drob[i];

}

else {

for (int i = 0; i <= 13; i++) par\_drob[i] = par[i];

//вывод в файл

int st = 1 / dt;

if (step % st == 0)

txtfile

<< t << ";" << xg << ";" << yg << ";" << zg << ";" << Vxg << ";" << Vyg << ";" << Vzg << ";" << V << ";"

<< tang \* 180 / M\_PI << ";" << risk \* 180 / M\_PI << ";" << kren \* 180 / M\_PI << ";"

<< omegax << ";" << omegay << ";" << omegaz << ";"

<< alpha \* 180 / M\_PI << ";" << betta \* 180 / M\_PI << ";"

<< delta\_tang \* 180 / M\_PI << ";" << delta\_risk \* 180 / M\_PI << ";" << delta\_kren \* 180 / M\_PI << ";" << r << ";" << endl;

}

if (abs(yg) < 0.0001) break; //Условие выхода

step++;

//Цикл расчета коэффициентов

for (int j = 1; j <= 4; j++) {

for (int i = 0; i <= 13; i++) parbuf[i] = par[i] + K[i][j - 1] \* a[j - 1]; // Считаем новые аргументы

// Нормировка параметров РГ

double norm\_RG = pow(roRG\_RK\*roRG\_RK + lamdaRG\_RK\*lamdaRG\_RK + nuRG\_RK\*nuRG\_RK + muRG\_RK\*muRG\_RK, 0.5);

roRG\_RK /= norm\_RG;

lamdaRG\_RK /= norm\_RG;

nuRG\_RK /= norm\_RG;

muRG\_RK /= norm\_RG;

MATR(roRG\_RK, lamdaRG\_RK, muRG\_RK, nuRG\_RK, &A, &AT);

// Расчет промежуточных параметров на шаге

MATR(roRG\_RK, lamdaRG\_RK, muRG\_RK, nuRG\_RK, &A, &AT);

tang = asin(2 \* (roRG\_RK\*nuRG\_RK + lamdaRG\_RK\*muRG\_RK));

risk = atan2(2 \* (roRG\_RK\*muRG\_RK - lamdaRG\_RK\*nuRG\_RK), roRG\_RK\*roRG\_RK + lamdaRG\_RK\*lamdaRG\_RK - nuRG\_RK\*nuRG\_RK - muRG\_RK\*muRG\_RK);

kren = atan2(2 \* (roRG\_RK\*lamdaRG\_RK - muRG\_RK\*nuRG\_RK), (roRG\_RK\*roRG\_RK - lamdaRG\_RK\*lamdaRG\_RK - nuRG\_RK\*nuRG\_RK + muRG\_RK\*muRG\_RK));

r = pow((xg\_RK - x\_t)\*(xg\_RK - x\_t) + (yg\_RK - y\_t)\*(yg\_RK - y\_t) + (zg\_RK - z\_t)\*(zg\_RK - z\_t), 0.5),

V = pow(Vxg\_RK\*Vxg\_RK + Vyg\_RK\*Vyg\_RK + Vzg\_RK\*Vzg\_RK, 0.5),

q = fro(yg\_RK)\*V\*V\*0.5,

vx = Vxg\_RK\*AT[0][0] + Vyg\_RK\*AT[0][1] + Vzg\_RK\*AT[0][2],

vy = Vxg\_RK\*AT[1][0] + Vyg\_RK\*AT[1][1] + Vzg\_RK\*AT[1][2],

vz = Vxg\_RK\*AT[2][0] + Vyg\_RK\*AT[2][1] + Vzg\_RK\*AT[2][2],

alpha = -atan2(vy, vx);

betta = asin(vz / V),

alpha\_full = pow(alpha\*alpha + betta\*betta, 0.5),

fi = asin((y\_t - yg\_RK) / r),

hi = -atan2(z\_t - zg\_RK, x\_t - xg\_RK),

B\_Matr(fi, hi, &B),

vr = B[0][0] \* (Vx\_t - Vxg\_RK) + B[0][1] \* (Vy\_t - Vyg\_RK) + B[0][2] \* (Vz\_t - Vzg\_RK);

vfi = B[1][0] \* (Vx\_t - Vxg\_RK) + B[1][1] \* (Vy\_t - Vyg\_RK) + B[1][2] \* (Vz\_t - Vzg\_RK);

vhi = B[2][0] \* (Vx\_t - Vxg\_RK) + B[2][1] \* (Vy\_t - Vyg\_RK) + B[2][2] \* (Vz\_t - Vzg\_RK);

dfidt = vfi / r,

dhidt = vhi / (r\*cos(fi)),

M = V / fa(yg\_RK);

if (yg>500) Control\_prog(tang, risk, kren, dfidt, dhidt, &delta\_tang, &delta\_risk, &delta\_kren);

X = -Cx(M)\*q\*Sm,

Y = (Cy\_alpha(M)\*alpha + Cy\_delta(M, alpha)\*delta\_tang)\*q\*Sm,

Z = (Cz\_betta(M, betta)\*betta + Cz\_delta(M, betta)\*delta\_risk)\*q\*Sm,

Mstab = 0.5\*dm \* 1000 \* delta\_kren,

Mx = (mx\_omegax()\*omegax\_RK\*L / V + mx\_delta(q\*Sm\*L)\*delta\_kren)\*q\*Sm\*L + Mstab,

My = (my\_omegay(M)\*omegay\_RK\*L / V + my\_betta(M)\*betta + my\_delta(M, betta)\*delta\_risk)\*q\*Sm\*L,

Mz = (mz\_omegaz(M)\*omegaz\_RK\*L / V + mz\_alpha(M)\*alpha + mz\_delta(M, alpha)\*delta\_tang)\*q\*Sm\*L;

//Производные (коэффициенты K[i][j])

dtdt = 1 \* dt,

dxgdt = Vxg\_RK\*dt,

dygdt = Vyg\_RK\*dt,

dzgdt = Vzg\_RK\*dt,

dVxgdt = 1 / m\*(X\*A[0][0] + Y\*A[0][1] + Z\*A[0][2])\*dt,

dVygdt = 1 / m\*(X\*A[1][0] + Y\*A[1][1] + Z\*A[1][2] - m\*g)\*dt,

dVzgdt = 1 / m\*(X\*A[2][0] + Y\*A[2][1] + Z\*A[2][2])\*dt,

domegaxdt = (Mx / Ix - omegay\_RK\*omegaz\_RK\*(Iz - Iy) / Ix)\*dt,

domegaydt = (My / Iy - omegax\_RK\*omegaz\_RK\*(Ix - Iz) / Iy)\*dt,

domegazdt = (Mz / Iz - omegay\_RK\*omegax\_RK\*(Iy - Ix) / Iz)\*dt,

droRGdt = -(omegax\_RK\*lamdaRG\_RK + omegay\_RK\*muRG\_RK + omegaz\_RK\*nuRG\_RK)\*0.5\*dt,

dlamdaRGdt = (omegax\_RK\*roRG\_RK - omegay\_RK\*nuRG\_RK + omegaz\_RK\*muRG\_RK)\*0.5\*dt,

dmuRGdt = (omegax\_RK\*nuRG\_RK + omegay\_RK\*roRG\_RK + omegaz\_RK\*lamdaRG\_RK)\*0.5\*dt,

dnuRGdt = (-omegax\_RK\*muRG\_RK + omegay\_RK\*lamdaRG\_RK + omegaz\_RK\*roRG\_RK)\*0.5\*dt;

}

// Приращения

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

double b[5] = { 0.0, 1.0 / 6.0, 2.0 / 6.0, 2.0 / 6.0, 1.0 / 6.0 };

for (int j = 1; j <= 4; j++) par[i] += K[i][j] \* b[j];

}

// Нормировка параметров РГ

double norm\_RG = pow(roRG\*roRG + lamdaRG\*lamdaRG + nuRG\*nuRG + muRG\*muRG, 0.5);

roRG /= norm\_RG;

lamdaRG /= norm\_RG;

nuRG /= norm\_RG;

muRG /= norm\_RG;

} // Конец интегрирования траектории

// Вывод последнего шага

txtfile

<< t << ";" << xg << ";" << yg << ";" << zg << ";" << Vxg << ";" << Vyg << ";" << Vzg << ";" << V << ";"

<< tang \* 180 / M\_PI << ";" << risk \* 180 / M\_PI << ";" << kren \* 180 / M\_PI << ";"

<< omegax << ";" << omegay << ";" << omegaz << ";"

<< alpha \* 180 / M\_PI << ";" << betta \* 180 / M\_PI << ";"

<< delta\_tang \* 180 / M\_PI << ";" << delta\_risk \* 180 / M\_PI << ";" << delta\_kren \* 180 / M\_PI << ";" << r << ";" << endl;

//if ((z\_t == 0)&(r < 50)) flag1 = 1; //вошли в начало зоны

//if ((z\_t == 0)&(flag1 == 1)&(abs(r - 50) < 1E-3)) { flag2 = 1; flag1 = 0; } //вышли из зоны

//if ((flag1 == 0) | (dz\_t < 1E-10)) break;

//Дробление шага

//if ((r > 50)&(flag1 == 1)) {

if (r > 50) {

z\_t -= dz\_t;

dz\_t /= 10;

}

//Выход из цикла поиска зоны

//if ((abs(r - 50) < 1E-3)&(flag1 == 1)) {

if (abs(r - 50) < 0.1){

txtfile2 << x\_t << ";" << z\_t << ";" << r << endl;

break;

}

cout << "r = " << r << endl

<< "x\_t = " << x\_t << endl

<< "z\_t = " << z\_t << endl

<< "dz\_t = " << dz\_t << endl;

cout << endl;

}

}

cout << "END!" << endl;

txtfile2 << "r = ;" << r << endl

<< "x\_t = ;" << x\_t << endl

<< "z\_t = ;" << z\_t << endl;

system("pause");

return 0;

}

//Аэродинамика

double Cx(double M) {

return pow(73.211 / exp(M) - 47.483 / M + 16.878, -1);

}

double Cy\_alpha(double M) {

double A = 1.86\*(11.554 / exp(M) - 2.5191E-3\*M\*M - 5.024 / M + 52.836E-3\*M + 4.112);

if (A < 0) return 1.86\*1.039;

else return pow(A, 0.5);

}

double Cy\_delta(double M, double alpha) {

alpha = abs(alpha);

alpha \*= 180 / M\_PI;

double p1 = pow(243.84E-3 / exp(-alpha) + 74.309E-3, -1),

p2 = log10(1.9773\*alpha\*alpha - 25.587\*alpha + 83.354),

p3 = 18.985\*alpha\*alpha - 375.76\*alpha + 1471,

p4 = -51.164E-3\*alpha\*alpha + 805.52E-3\*alpha + 1.8929;

return 2\*(-p1\*1E-6\*M\*M + p2\*1E-2\*exp(M) - p3\*1E-6\*M - p4\*1E-3);

}

double Cz\_betta(double M, double betta) {

return -Cy\_alpha(M);

}

double Cz\_delta(double M, double betta) {

return -Cy\_delta(M, betta);

}

double mx\_omegax() {

return -0.005\*0.6786;

}

double mx\_delta(double q) {

return -1000 / (q);

}

double mz\_omegaz(double M) {

return 1.89\*(146.79E-6\*M\*M - 158.98E-3 / M - 7.6396E-6\*M - 68.195E-3);

}

double mz\_alpha(double M) {

return -766.79E-3 / exp(M) + 438.74E-3 / M + 5.8822E-3\*M - 158.34E-3 ;

}

double mz\_delta(double M, double alpha) {

alpha = abs(alpha);

alpha \*= 180 / M\_PI;

double k1 = exp(-19.488E-3\*alpha\*alpha - 378.62E-3\*alpha + 6.7518),

k2 = exp(-21.234E-3\*alpha\*alpha - 635.84E-6\*exp(alpha) - 98.296E-3\*alpha + 2.5938);

return 1.89\*pow(k1\*1E-9\*M\*M + k2\*1E-6, 0.5);

}

double my\_omegay(double M) {

return mz\_omegaz(M);

}

double my\_betta(double M) {

return mz\_alpha(M);

}

double my\_delta(double M, double betta) {

return mz\_delta(M, betta);

}

//Матрицы

void B\_Matr(double fi, double hi, double(\*B)[3][3]) {

double b[3][3] = { 0 };

b[0][0] = cos(fi)\*cos(hi);

b[0][1] = sin(fi);

b[0][2] = -cos(fi)\*sin(hi);

b[1][0] = -sin(fi)\*cos(hi);

b[1][1] = cos(fi);

b[1][2] = sin(fi)\*sin(hi);

b[2][0] = sin(hi);

b[2][1] = 0;

b[2][2] = cos(hi);

for (int i = 0; i <= 2; i++) for (int j = 0; j <= 2; j++)

(\*B)[i][j] = b[i][j];

}

void MATR(double ro, double lamda, double mu, double nu, double(\*A)[3][3], double(\*AT)[3][3]) {

double a[3][3] = { 0 };

// Матрица из нормальной земной в связную

a[0][0] = ro\*ro + lamda\*lamda - mu\*mu - nu\*nu;

a[0][1] = 2 \* (-ro\*nu + lamda\*mu);

a[0][2] = 2 \* (ro\*mu + lamda\*nu);

a[1][0] = 2 \* (ro\*nu + lamda\*mu);

a[1][1] = ro\*ro - lamda\*lamda + mu\*mu - nu\*nu;

a[1][2] = 2 \* (-ro\*lamda + nu\*mu);

a[2][0] = 2 \* (-ro\*mu + lamda\*nu);

a[2][1] = 2 \* (ro\*lamda + nu\*mu);

a[2][2] = ro\*ro - lamda\*lamda - mu\*mu + nu\*nu;

for (int i = 0; i <= 2; i++) for (int j = 0; j <= 2; j++)

(\*A)[i][j] = a[i][j];

//Из связной в нормальную

for (int i = 0; i <= 2; i++) for (int j = 0; j <= 2; j++)

(\*AT)[i][j] = a[j][i];

}

//Стабилизация и наведение

void Control\_prog(double tang, double risk, double kren, double dfidt, double dhidt, double \*delta\_tang, double \*delta\_risk, double \*delta\_kren) {

//Для наведения

const double KSI\_SST = 0.35;

const double KSI\_SSN = 0.35;

const double KSI\_SSE = 0.35;

const double K\_SST = 0.95;

const double K\_SSN = 0.95;

const double T\_SSE = 0.01;

double a11 = -mz\_omegaz(M) \* q \* Sm \* L \* L / (Iz \* V);

double a12 = -mz\_alpha(M) \* q \* Sm \* L / Iz;

double a13 = -mz\_delta(M, alpha) \* q \* Sm \* L / Iz;

double a42 = Cy\_alpha(M) \* q \* Sm / (m \* V);

double a43 = Cy\_delta(M, alpha) \* q \* Sm / (m \* V);

double vke = Cy\_delta(M,alpha);

double b11 = -my\_omegay(M) \* q \* Sm \* L \* L / (Iy \* V);

double b12 = -my\_betta(M) \* q \* Sm \* L / Iy;

double b13 = -my\_delta(M, betta) \* q \* Sm \* L / Iy;

double b42 = 0;

double b43 = -Cy\_delta(M, betta) \* q \* Sm / (m \* V);

double c11 = mx\_omegax() \* q \* Sm \* L \* L / (Ix \* V);

double c13 = mx\_delta(q\*Sm\*L) \* q\*Sm\*L / Ix;

double K\_T = (a12 \* a43 - a13 \* a42) / (a12 + a11 \* a42);

double T\_1T = -(a13) / (a13 \* a42 - a12 \* a43);

double T\_T = 1 / sqrt(a12 + a11 \* a42);

double KSI\_T = (a11 + a42) / (2 \* sqrt(a12 + a11 \* a42));

double K\_N = (b12 \* b43 - b13 \* b42) / (b12 + b11 \* b42);

double T\_1N = -(b13) / (b13 \* b42 - b12 \* b43);

double T\_N = 1 / sqrt(b12 + b11 \* b42);

double KSI\_N = (b11 + b42) / (2 \* sqrt(b12 + b11 \* b42));

double K\_E = c13 / c11;

double T\_E = 1 / c11;

double K\_fi = 9, K\_hi = 10;

double Kk\_t = 6.0, Kk\_n = 10;

double sf = KSI\_SST \* KSI\_SST\*KSI\_SST\*KSI\_SST\*T\_T\*T\_T;

double gf = 2 \* KSI\_T\*KSI\_SST\*KSI\_SST\*T\_T\*T\_1T;

double hf = T\_1T \* T\_1T\*KSI\_SST\*KSI\_SST;

double df = sqrt(sf - gf + hf);

double K\_T2 = -2 \* T\_T\*(KSI\_T\*T\_1T - KSI\_SST \* KSI\_SST\*T\_T - df) / (K\_T\*T\_1T\*T\_1T);

double K\_T1 = K\_SST \* (1 + K\_T2 \* K\_T) / (K\_T\*T\_1T\*T\_1T);

double K\_N2 = -2 \* T\_N \* (KSI\_N \* T\_1N - KSI\_SSN \* KSI\_SSN\*T\_N - sqrt(KSI\_SSN \* KSI\_SSN \* KSI\_SSN \* KSI\_SSN \* T\_N \* T\_N - 2 \* KSI\_N\*KSI\_SSN\*KSI\_SSN\*T\_N\*T\_1N + T\_1N \* T\_1N\*KSI\_SSN\*KSI\_SSN)) / (K\_N\*T\_1N\*T\_1N);

double K\_N1 = K\_SSN \* (1 + K\_N2 \* K\_N) / (K\_N\*T\_1N\*T\_1N);

double K\_E1 = (2 \* KSI\_SSE\*T\_E - T\_SSE) / (K\_E\*T\_SSE);

double K\_E2 = T\_E / (K\_E\*T\_SSE\*T\_SSE);

double K\_T22 = K\_T1 \* K\_fi \* Kk\_t;

double K\_N22 = K\_N1 \* K\_hi \* Kk\_n;

double deltaT = -K\_T2 \* (omegaz \* cos(kren) + omegay \* sin(kren)) + K\_T22 \* dfidt;

double deltaN = -K\_N2 \* (omegay \* cos(kren) - omegaz \* sin(kren)) / cos(tang) + K\_N22 \* dhidt;

double deltaE = -K\_E2 \* kren - K\_E1 \* (omegax - tan(tang)\*(omegay \* cos(kren) - omegaz \* sin(kren)));

const double angleDelta = 15\*M\_PI/180;

\*delta\_tang = (abs(deltaT) > angleDelta) ? angleDelta \*(deltaT / abs(deltaT)) : deltaT;

\*delta\_risk = (abs(deltaN) > angleDelta) ? angleDelta \*(deltaN / abs(deltaN)) : deltaN;

\*delta\_kren = (abs(deltaE) > angleDelta) ? angleDelta \*(deltaE / abs(deltaE)) : deltaE;

}

**Приложение Г**

Ниже представлены графические зависимости параметров траектории для «зоны ослепления». «Зона ослепления» начинается с высоты 500 метров и фиксирует отклонения рулей. Координаты цели х = 12100 м, z = 2742 м.

Рисунок Г.1 – Траектория в проекции на вертикальную плоскость ОХУ

Рисунок Г.2 – Траектория в проекции на горизонтальную плоскость ОХZ

Рисунок Г.3 – Зависимость радиус-вектора до цели от времени полета

Рисунок Г.4 – Зависимость угла атаки от времени полета

Рисунок Г.5 – Зависимость угла скольжения от времени полета

Рисунок Г.6 – Зависимость угла тангажа от времени полета

Рисунок Г.7 – Зависимость угла рысканья от времени полета

Рисунок Г.8 – Зависимость угла крена от времени полета

Рисунок Г.9 – Зависимость угловой скорости по оси Х от времени полета

Рисунок Г.10 – Зависимость угловой скорости по оси Y от времени полета

Рисунок Г.11 – Зависимость угловой скорости по оси Z от времени полета

Рисунок Г.12 – Зависимость угла отклонения рулей по каналу тангажа от времени полета

Рисунок Г.13 – Зависимость угла отклонения рулей по каналу рысканья от времени полета

Рисунок Г.14 – Зависимость угла отклонения рулей по каналу крена от времени полета