Expt. No. **5** **Direct Kinematics**

**Program:**

#include<iostream.h>

#include<conio.h>

#include<stdio.h>

#include<math.h>

const int DOF=5;

const double P1=4\*atan(1.0);

const char wait[]="Press any key to continue";

double PI=3.14;

class Hctm

{

private:

 double mat[4][4];

public:

 Hctm();

 Hctm(double m[][4]);

 void initLctm(double ang[], double d[], double a[], double alp[], int k);

 void initIdentity();

 void printLctm(int k,int kl);

Hctm operator \*(Hctm a);

};

Hctm::Hctm()

{

int i,j;

for(i=0;i<4;i++)

for(j=0;j<4;j++)

mat[i][j]=0;

}

Hctm::Hctm(double a[][4])

{

int i,j;

for(i=0;i<4;i++)

for(j=0;j<4;j++)

mat[i][j]=a[i][j];

}

void Hctm::initIdentity()

{

for(int i=0;i<4;i++)

mat[i][i]=1;

}

void Hctm::initLctm(double ang[], double d[], double a[], double alp[], int l)

{

//initializes mat with Lctmn T (k to k-1)

int k=l-1;

double co=cos(ang[k]);

double so=sin(ang[k]);

double ca=cos(alp[k]);

double sa=sin(alp[k]);

mat[0][0]=co;

mat[0][1]=-ca\*so;

mat[0][2]=sa\*so;

mat[0][3]=a[k]\*co;

mat[1][0]=so;

mat[1][1]=ca\*co;

mat[1][2]=-sa\*co;

mat[1][3]=a[k]\*so;

mat[2][1]=sa;

mat[2][2]=ca;

mat[2][3]=d[k];

mat[3][3]=1;

}

void Hctm::printLctm(int k,int k1)

{

int y=wherey(),i,j,x;

gotoxy(10,y+2);cout<<"T=";

gotoxy(11,y+1);cout<<k;

gotoxy(11,y+2);cout<<k1;

gotoxy(15,y);cout<<'L';

gotoxy(15,y+5);cout<<'A';

for(i=0;i<4;i++)

{

gotoxy(15,y+i+1);

cout<<"";

for(j=0;j<4;j++)

printf("%-10.4f",mat[i][j]);

x=wherey();cout<<"";

}

gotoxy(x,y);cout<<"";

gotoxy(x,y+5);cout<<'U';

gotoxy(1,y+6);

}

Hctm Hctm::operator\*(Hctm a)

{

Hctm res;

int i,j,k;

double sum=0;

for(i=0;i<4;i++)

for(j=0;j<4;j++)

{

sum=0;

for(k=0;k<4;k++)

sum+=mat[i][k]\*a.mat[k][i];

res.mat[i][j]=sum;

}

return res;

}

inline void pause(char\*s=(char\*)wait);

void line (char c='i',int len=78);

int main (void)

{

clrscr();

double dk[DOF];

double ak[DOF];

double angle[DOF];

double alpha[DOF];

int i;

cout<<(ios::stdio);

cout<<'E';line();cout<<'>>';

cout<<"DIRECT KINEMATICS ANALYSIS OF RHINO XR\_3 AXIS ROBOT";

gotoxy(80,wherey()); cout <<'q';

cout<<'E';line();cout<<"1/4\n";

cout<<"\n\t\t Enter joint distances";

cout<<"\t\t";

line('I',30);

cout<<"\n\n";

for(i=0;i<DOF;i++)

{

if(i>0&&i<DOF-1)

dk[i]=0;

else

{

cout<<"\t\t Enter D ["<<(i+1)<<"]";

cin>>dk[i];

}

}

cout<<"\t\t\n Enter linklenghts\n";

cout<<"\t\t\t"; line('i',30);

cout<<"\n\n";

for(i=0;i<DOF;i++)

{

if(i==0||DOF-1)

ak[i]=0;

else

{

cout<<"\t\t\t Enter A["<<(i+1)<<"]:";

cin>>ak[i];

}

}

pause();

clrscr();

cout<<"\n\t\n Enter linktwist angeles (IN degrees)\n";

cout<<"\t\t";

line('I',50);

cout<<"\n\n";

for(i=0;i<DOF;i++)

{

cout<<"\t\t Enter a["<<(i+1)<<"]";

cin>>alpha[i];

alpha[i]=angle[i]\*PI/180;

}

cout<<"\n\t\t Enter joint angles in degrees\n";

cout<<"\t\t";

line('I');

cout<<"\n\n";

for(i=0;i<DOF;i++)

{

cout<<"\t\t Enter E["<<(i+1)<<"]";

cin>>angle[i];

angle[i]=angle[i]\*PI/180;

}

pause();

clrscr();

cout<<'E';

line();

cout<<'>>';

cout<<"\t intermediate homogenous co ordinate transformation matrices";

gotoxy(80,wherey());

cout<<'E';

cout<<'E';

line();

cout<<"1/4\n";

Hctm rhino[DOF];

Hctm arm;

arm.initIdentity();

for(i=0;i<DOF;i++)

{

rhino[i].initLctm(angle,dk,ak,alpha,i+1);

rhino[i].printLctm(i+1,i);

if(i==2)

{

pause();

clrscr();

}

arm=arm\*rhino[i];

}

pause();

clrscr();

cout<<'E';

line();

cout<<'>>';

cout<<"\t\t\t Final arm matrix";

gotoxy(80,wherey());

cout<<'E';

cout<<'E';

line();

cout<<"1/4\n";

arm.printLctm(DOF,0);

pause();

return 0;

}

void line (char c,int len)

{

for(;len;len--)

cout<<c;

}

void pause (char s[])

{

gotoxy(30,25);

cout<<s;

getch();

}

**Output:**