Name: Guy Joseph
ROBOTICS HW1

Excellent BM1 Check page 3 of Answer Key

Question 1-
A Robot can be distinguished from other form of automated machine by the fact that a computer or a microprocessor controls it. This means that the robot can be reprogrammed to accomplished many tasks. Robots also have external sensors, which allow them to adapt to their environment.

Question 2-
Define terms

- **Join Variables** (revolute & pragmatic) are the relative displacement between adjacent links.
- **Workspace** of a manipulator is the total volume the end-effector can reach and manipulate.
- **Accuracy** is how close a manipulator can come to a given point within a workspace.
- **Repeatability** is how close a manipulator can return to a previously though point.
- **Resolution** is the smallest increment of motion that a controller can sense.
- **Wrists** refer to the joints in the kinematics chain between the arm and hand. Robot with spherical wrists has wrists whose joints intersect at a common point.
- The **end-effector** of a manipulator is basically its hand (the mechanical device the manipulator uses to contact and manipulate objects).
- **Forward kinematics** determines the position and orientation of the end-effector in term of joint variables (return x,y,z coordinates based on angles)
- **Inverse kinematics** determines the joint angles using position and orientation (giving x,y,z calculate the appropriate angles)
- **Trajectory Planning** is the ability to smoothly control the end-effector' paths from point A to point B

Question 3-
Robots can be classify by they geometry, kinematics structure, power source, application area and method of control.

Question 6-
Application for non-servo Robot is mainly in the area of materials transfer (i.e., filter defective product, assembly line, displacing or moving object from point A to point B). The reason is because they rely on predetermine mechanical stop.
Point-to-Point robot can be used to determine the smoothness of a surface and application alike.
Continuous paths Robots have a wide range of application and are only limited by they mechanical structure and number of DOF. They applications vary from assembly line to painting.

Question 7-
Application apply to continuous path Robot and not to Point-to-Point robot
Painting, Satellite Recovery, Automatic Navigation, Medical Surgery and under-sea exploration

Question 8-
Application where computer vision is useful
Scene recognition, Analysis of 3-D object, Medical surgery, Cleaning nuclear reactor, Space mission

Question 9-
Tactile sensing or force feedback control is useful in tasks that involve extensive contact with the environment tasks such as: Handling sensitive materials like washing a window, painting, grinding, deburring etc…

Question 11-
The effect of fully automated all the factories in the US would have an economical impact as well as a social impact. Such automation would result in a decrease of labor cost, increase in precision and productivity. Thus, factories would be able to respond to the high market demand for certain products and

provides them at a cheaper price. This would in turn decrease the cost of living. However, factory workers would be unhappy because they would loose their job.

Question12-
Banning the used of all industrial Robot in the US would generate and economic crisis. Because, the US would be incapable to compete with the rest of the world. Certain products require a high level of precision (i.e., microprocessor fabrication) which can't be reach by human.
Moreover, factories would have to hire and trained new people. This could consequently raise the cost of living.

Question 13-
Possible application of redundant manipulator would be cleaning nuclear reactor. I would be desirable to have two manipulator in case one filed the other one can rescue the defective one or take over. Same principal applies for space missions.

Question 14-
a)

b) Resolution of the linear link
For n=10 and L=1m
D= 1 * sqrt(2*(1-cos(90)))
D=sqrt(2)=1.4142
R=d/2^n
R= 1.4142 / 2^10=.001381

c) Resolution of the rotational link

D= 1 * 90
R= 90/2^10 = .0878

Question 15-
For theta=180degrees and l=50*10^-2

D = 50*10^-2 * 180 / 2^8 = .17578

Question 17-
Most of the time industry buys pre-fabric Robot and and try to make them fit to accomplish a task. Computational error can occur because of the fact that the actual geometry of the Robot is unknown or not well known. Another reason may be, the manipulator is not rigid enough and introduce errors through bending under pressure from heavy load and so on. Accuracy can be affected by a lot of factor, where as repeatability is only affected by the controller resolution.

Question 18-
Direct sensing can be used to increase the accuracy by tracking the trajectory of the robot and comparing it with the expected path. Then use feed back to control any error. Or in crease the number of decoder used.

Question 20-
For a1=a2=1 T1=pi/6 T2=pi/2
X=0.366025 m

Y=1.366025 m

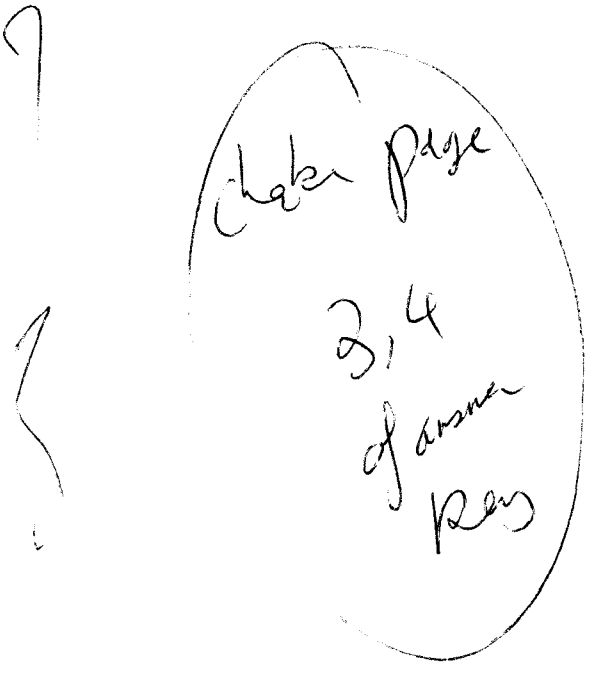
Question 21-
For a1=a2=1 X=Y=0.5

T1=-24.295193 degrees
T2=138.590385 degrees

Question 22-
For a1=a2=1 DT1=1 DT2=2

a) Dx= -0.054 m/s
Dy= -0.011 m/s

b) Dx=-0.064 m/s
Dy=0.012 m/s



Question 25-
It is possible to have an infinite number of solutions because the inverse kinematics is based on the forward kinematics equations which is non-linear.

```
//Guy Joseph
//P1

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define TRUE  1
#define FALSE 0
#define m_pi  3.14159265359

float DtoR(float theta);
float RtoD(float theta);
void  Kinematics();
void  Ken(float T1, float a1, float T2, float a2, float *x, float *y);
void  InvKinematics();
void  InvKen(float x, float y, float a1, float a2, float* T1, float *T2);
void  VelKinematics();
void  VelKen(float a1, float a2, float T1, float T2, float DT1, float DT2, float *x, float *y);
void  InvVelKinematics();
void  InvVelKen(float a1, float a2, float dx, float dy, float T1, float T2, float *DT1, float *DT2);
void  AccKinematics();
void  Accel(float a1, float a2, float T1, float T2,  float DT1,  float DT2,  float DDT1,  float DDT2
, float *x, float *y);
void  InvAccKinematics();
void  InvAccel(float a1, float a2, float T1, float T2, float DT1, float DT2, float DDx, float DDy, f
loat *DDT1, float *DDT2);

void main()
{
    int choice=0;
    int loop=TRUE;
    while(loop)
    {
        printf("\nThis program was compile under Microsoft Virsual C++\n");
        printf("1- Forward Kinematics\n");
        printf("2- Inverse Kinematics\n");
        printf("3- Velocity\n");
        printf("4- Inverse Velocity\n");
        printf("5- Acceleration\n");
        printf("6- Inverse Acceleration\n");
        printf("7- Exit the program\n");
        printf("Your Selection: ");
        scanf(" %d",&choice);
        switch(choice)
        {
        case 1:
            Kinematics();
            break;
        case 2:
            InvKinematics();
            break;
        case 3:
            VelKinematics();
            break;
        case 4:
            InvVelKinematics();
            break;
        case 5:
            AccKinematics();
            break;
        case 6:
            InvAccKinematics();
            break;
        case 7:
            loop=FALSE;
            break;
        }
    }
}


void Kinematics()
{
    float T1,T2;
    float a1,a2,x,y;

    a1=a2=x=y=0.0;
    T1=T2=0.0;
```

```
    printf("Enter Theta1 (degrees): ");
    scanf(" %f",&T1);
    printf("Enter Theta2 (degrees): ");
    scanf(" %f",&T2);
    printf("Enter Length of 1st Link (a1): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2): ");
    scanf(" %f",&a2);

    T1=DtoR(T1);
    T2=DtoR(T2);
    Ken(T1,a1,T2,a2,&x,&y);
    printf("X= %f\n",x);
    printf("Y= %f\n",y);
}

void InvKinematics()
{
    float x,y,T1,T2;
    float a1,a2;

    a1=a2=x=y=0.0;
    T1=T2=0.0;

    printf("Enter x coordinate: ");
    scanf(" %f",&x);
    printf("Enter y coordinate: ");
    scanf(" %f",&y);
    printf("Enter Length of 1st Link (a1) (in meter): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2) (in meter): ");
    scanf(" %f",&a2);

    InvKen(x,y,a1,a2,&T1,&T2);
    printf("Theta1= %f\n",RtoD(T1));
    printf("Theta2= %f\n",RtoD(T2));

}

void VelKinematics()
{
    float x,y,T1,T2,DT1,DT2;
    float a1,a2;

    a1=a2=x=y=0.0;
    T1=T2=0.0;

    printf("Enter Theta1: ");
    scanf(" %f",&T1);
    printf("Enter Theta2: ");
    scanf(" %f",&T2);
    printf("Enter Theta1 per second: ");
    scanf(" %f",&DT1);
    printf("Enter Theta2 per second: ");
    scanf(" %f",&DT2);
    printf("Enter Length of 1st Link (a1): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2): ");
    scanf(" %f",&a2);

    VelKen(a1,a2,DtoR(T1),DtoR(T2),DtoR(DT1),DtoR(DT2),&x,&y);
    printf("Dx = %f\n",x);
    printf("Dy = %f\n",y);
}

void InvVelKinematics()
{
    float dx,dy,T1,T2,DT1,DT2;
    float a1,a2;

    a1=a2=dx=dy=0.0;
    T1=T2=0.0;

    printf("Enter Theta1: ");
    scanf(" %f",&T1);
    printf("Enter Theta2: ");
    scanf(" %f",&T2);
    printf("Enter dx: ");
    scanf(" %f",&dx);
    printf("Enter dy: ");
```

```
    scanf(" %f",&dy);
    printf("Enter Length of 1st Link (a1): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2): ");
    scanf(" %f",&a2);
    InvVelKen(a1,a2,dx,dy,DtoR(T1),DtoR(T2),&DT1,&DT2);
    printf("DTheta1 = %f\n",RtoD(DT1));
    printf("DTheta2 = %f\n",RtoD(DT2));
}

void AccKinematics()
{
    float x,y,T1,T2,DT1,DT2,DDT1,DDT2;
    float a1,a2;

    a1=a2=x=y=0.0;
    T1=T2=0.0;

    printf("Enter Theta1: ");
    scanf(" %f",&T1);
    printf("Enter Theta2: ");
    scanf(" %f",&T2);
    printf("Enter Theta1 per second: ");
    scanf(" %f",&DT1);
    printf("Enter Theta2 per second: ");
    scanf(" %f",&DT2);
    printf("Enter Theta1 per second square (acceleration): ");
    scanf(" %f",&DDT1);
    printf("Enter Theta2 per second square (acceleration): ");
    scanf(" %f",&DDT2);
    printf("Enter Length of 1st Link (a1) (in meter): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2) (in meter): ");
    scanf(" %f",&a2);

    Accel(a1,a2,DtoR(T1),DtoR(T2),DtoR(DT1),DtoR(DT2),DtoR(DDT1),DtoR(DDT2),&x,&y);
    printf("Acceleration ddx = %f\n",x);
    printf("Acceleration ddy = %f\n",y);

}

void InvAccKinematics()
{
    float x,y,T1,T2,DT1,DT2,DDT1,DDT2,DDx,DDy;
    float a1,a2;

    a1=a2=x=y=0.0;
    T1=T2=0.0;

    printf("Enter Theta1: ");
    scanf(" %f",&T1);
    printf("Enter Theta2: ");
    scanf(" %f",&T2);
    printf("Enter Theta1 per second: ");
    scanf(" %f",&DT1);
    printf("Enter Theta2 per second: ");
    scanf(" %f",&DT2);
    printf("Enter acceleration ddx: ");
    scanf(" %f",&DDx);
    printf("Enter acceleration ddy: ");
    scanf(" %f",&DDT2);
    printf("Enter Length of 1st Link (a1): ");
    scanf(" %f",&a1);
    printf("Enter Length of 2nd Link (a2): ");
    scanf(" %f",&a2);

    InvAccel(a1,a2,DtoR(T1),DtoR(T2),DtoR(DT1),DtoR(DT2),DDx,DDy,&DDT1,&DDT2);
    printf("Inverse Acceleration T1 = %f\n",RtoD(DDT1));
    printf("Inverse Acceleration T2 = %f\n",RtoD(DDT2));
}


void Ken(float T1, float a1, float T2, float a2, float *x, float *y)
{
    *x=(float)(a1*cos((double)T1)+a2*cos((double)(T1+T2)));
    *y=(float)(a1*sin((double)T1)+a2*sin((double)(T1+T2)));
}

void InvKen(float x, float y, float a1, float a2, float* T1, float *T2)
{
```

```
    double D,tmp,tmp2;
    D=(x*x + y*y -(a1*a1) -(a2*a2))/(2*a1*a2);
    tmp=sqrt(1-D*D);
    tmp=atan2(tmp,D);
    *T2=(float)tmp;
    tmp=a2*sin(tmp)/(a1+a2*cos(tmp));
    tmp2=atan(tmp);
    tmp=atan2(y,x);
    *T1=(float)(tmp-tmp2);
}

void  VelKen(float a1, float a2, float T1, float T2, float DT1, float DT2, float *x, float *y)
{
    *x=(float)(-a1*DT1*sin((double)T1) - a2*(DT1+DT2)*sin((double)(T1+T2)));
    *y=(float)(a1*DT1*cos((double)T1) + a2*(DT1+DT2)*cos((double)(T1+T2)));
}

void InvVelKen(float a1, float a2, float dx, float dy, float T1, float T2, float *DT1, float *DT2)
{
    double InvD,D;
    double sint1pt2,cost1pt2;
    sint1pt2=sin(T1+T2);
    cost1pt2=cos(T1+T2);
    D= -a1*a2*sin(T1)*cost1pt2+a1*a2*cos(T1)*sint1pt2;
    InvD=(float)(1.0/D);
    *DT1=(float)(InvD*a2*(cost1pt2*dx+sint1pt2*dy));
    *DT2=(float)(InvD*((-a1*cos(T1)-a2*cost1pt2)*dx+(-a1*sin(T1)-a2*sint1pt2)*dy));

}

void  Accel(float a1, float a2, float T1, float T2,  float DT1,  float DT2,  float DDT1,  float DDT2
, float *x, float *y)
{
    double tmp1,tmp2;
    tmp1=-a1*DT1*DDT1*cos(T1);
    tmp2=-a2*cos(T1+T2);
    *x=(float)(tmp1-tmp2*DT1*DDT1-tmp2*DT1*DDT2-tmp2*DT2*DDT1-tmp2*DT2*DDT2);

    tmp1=-a1*DT1*DDT1*sin(T1);
    tmp2=-a2*sin(T1+T2);
    *y=(float)(tmp1-tmp2*DT1*DDT1-tmp2*DT1*DDT2-tmp2*DT2*DDT1-tmp2*DT2*DDT2);
}

void  InvAccel(float a1, float a2, float T1, float T2, float DT1, float DT2, float DDx, float DDy, f
loat *DDT1, float *DDT2)
{
    double InvD,D,tmp1,tmp2,tmp3,tmp4;
    double sint1pt2,cost1pt2;
    sint1pt2=sin(T1+T2);
    cost1pt2=cos(T1+T2);

    tmp2=a1*cos(T1)*DT1*a2*sint1pt2;
    tmp1=tmp2*DT1+tmp2*DT2;
    tmp3=-a1*sin(T1)*DT1*a2*cost1pt2;
    tmp2=tmp3*DT1+tmp3*DT2;
    D = tmp1+tmp2;
    InvD=1.0/D;

    tmp1=-a2*sint1pt2*DT1-a2*sint1pt2*DT2;
    tmp2=a2*cost1pt2*DT1+a2*cost1pt2*DT2;
    tmp3=-a1*cos(T1)*DT1-a2*cost1pt2;
    tmp4=tmp3*DT1+tmp3*DT2;
    tmp3=a1*sin(T1)*DT1;

    *DDT1=(float)(tmp1*DDx*InvD+tmp2*DDy*InvD);
    *DDT2=(float)(tmp3*DDx*InvD+tmp4*DDy*InvD);
}

float DtoR(float theta)
{
    return (float)((theta*m_pi)/180.0);
}

float RtoD(float theta)
{
    return (float)((180.0*theta)/m_pi);
}
```