DETAILS ABOUT THE FUNCTIONS USED IN THE CODE :

1. **function [] = ForwDyn()**
   USE : To calculate the forward dynamics

2. **function [com_final,cm,P_final,L_final,P,L] = ForwKin(theta,T_mat,dq)**
   USE : Calculate the forward kinematics of the robot

3. **function [] = ForwKin_Dyn(n)**
   USE : Updates the kinematics for the calculation of dynamics

4. **function [T] = Hom_Trans(alp,a,d,theta)**
   USE : D-H Homogeneous Tranformation Matrix

5. **function [f,t,tt] = InvDyn(n)**
   USE : To calculate the inverse dynamics

6. **function [ret] = InvDyn_cal(n)**
   USE : To calculate the inverse dynamics

7. function [flag,q,Dq,pos,cdl,HG] =InvKin(x,y,z,part_Id,from,adj,theta)
   USE : To calculate the inverse kinematics

8. function [flag,q,gg,pts] =
InvKin_com(x,y,z,tg,Tr_mat,enf,from,theta,com)

   USE : To calculate the inverse kinematics on COM

9. function [flag,q,Dq,pos] =
InvKin_mult(x1,y1,z1,part_Id1,x2,y2,z2,part_Id2,from,from2,adj,theta,dq)

   USE : To check and obtain the inverse kinematics solution to reach one link from some other link

10. function [flag,q] = InvKin_mult2(x,y,z,parts,from,adj)
   USE : To calculate the inverse kinematics on any link

11. function [I_s] = Is(m,c,I)

USE : Generate spatial inertia matrix

12 . function [J] = Jacob(u,p)
USE : Jacobian matrix generated

13. function [J_com] = Jacob_CoM ()
USE : Jacobian for calculating IK of COM trajectory

14. function [x1,y1,z1] = LegTraj(l_pos,orie,sLen, sHt, sDisp, itr)
USE : To generate the walking trajectory of the humanoid

15. function[h,AG,hG,vG] = M_mats(Tr_mat,com,dq)
USE : AG_CMM matrix obtained

16. function [x,y,z] = Traj_1()

USE : Trajectory generation

17. function [x,y,z] = Traj_3()

USE : Trajectory generation

18. function [x,y,z] = Traj_4()
USE : Trajectory generation

19. function [v,w] = calVW(pid,cid,dq,fr)
USE : Calculate link velocity and angular velocity w.r.t global frame

20. function [flag] = collision_check()
USE : Self collision between the parts

21. function [Idx] = find_route(part_Id,from)
USE : Finds the route from "from"(input) to the part id

21. function [] = follow_traj(x,y,z,q,loop)
USE : simulation function

22. **function [ZMP,q] = gen_gait()**
USE : Walking gait generation

23. **function[R] = gen_rand_num(arr,n)**
USE : generating a sequence of random numbers lying in the range specified by "arr "

24. **function [v_hat] = hat(v)**
USE : hat operation (sxw = hat(s)w) [hat(s) is a 3x3 matrix]

25 **function ji = invsvd(j)**
USE : Calculate inverse of matrix using SVD

26. **function JI = invsvd_lds(J,lamda)**
USE : Inverse of a matrix using SVD-LDS

27. **function [mi] = limits_2(str)**
USE : Angle limits for each base

28. **function [distance varargout] = line_to_line(p1, p2, p3, p4)**
USE : Computes the minimum distance between two line segments

29. **function [x,z,x_h,z_h] = new_traj(n)**
USE : Generating leg and hip trajectories

30. **function d = point_to_line(pt, v1, v2)**
USE : To calculate the perpendicular distance between the point and the line defined by the two vectors v1,v2

31. **function[] = s_d(cm,com)**
USE : To get the stick figure

32. **function [] = sample_draw(p)**
USE : generate the stick figures

33. **function [v,w] = standing_vel(dq)**
USE : To calculate the velocities during standinga and walking

34. function [v,w] = walking_vel(dq)

USE : To calculate the velocities during walking

LIST OF VARIOUS PARAMETERS USED IN THE CODE :

1. **robot.parts(n).axis_loc** - denotes the axis location of robot part n
2. **robot.parts(n).joint_loc** - denotes the joint location of the robot part n
3. **robot.parts(n).Z_w** - denotes the rotation axis in the global frame n
4. **robot.parts(n).R_mat** - denotes the rotation matrix of robot part n
5. **robot.parts(n).b** - denotes the position of the child w.r.t parent in the local frame
6. **robot.parts(n).a** -axis of rotation in local frame of child w.r.t parent
7. **robot.parts(n).ar** - axis of rotation in local frame of parent w.r.t child
8. **robot.parts(n).br** - denotes the position in the local frame of the child w.r.t parent frame
9. **robot.parts(n).mass** - denotes the mass of the robot part n
10. **robot.parts(n).I** - denotes the inertia of the robot part in the global frame
11. **robot.parts(n).com_g** - denotes the COM of the robot part n in the global frame
12. **robot.parts(n).com_l** - denotes the COM of the robot part n in the local frame
13. **robot.parts(n).v** - denotes the linear velocity of the robot part n

14. robot.parts(n).w - denotes the angular velocity of the robot part n

15. robot.parts(n).P - denotes the linear momentum of the robot part n

16. robot.parts(n).L - denotes the angular momentum of the robot part n

17. robot.parts(n).dq - denotes the change of the angle of the robot link

18. robot.parts(n).ddq -denotes the angular acceleration of the part n of the robot

19. P_Id - denotes the parent Id

20 C_Id -denotes the child Id

21 Xs - denotes the transformation matrix for linear and angular momentum

22 Is - Spatial matrix

23 w0 - Spatial angular velocity

24. v0 - Linear angular velocity

25.P0 -Spatial Linear momentum

26 L0 - Spatial angular momentum

27. u -Joint torque

30 Id - Part_id

31* Xpg

32* dw

32* dv0

33* sw

34* sv

35* dw0

36* f1

37 * t1

38 * tt1

39* f0

40* t0

41* tt0