```
function [x_traj, uDDP] = calcDDP(xo, uo, p_target, Q_f, Q, R, Horizon, dt,
 h_0, hx1_0, hx2_0, hx3_0, gamma)

% Number of Iterations
num_iter = 20;

u_k = ones(4,Horizon);
du_k = zeros(4,Horizon);
u_k(:, :) = u_k(:,:) *uo(1);

% Initial trajectory:
x_traj = zeros(13, Horizon);
x_traj(1:13,1) = xo;



% Learning Rate:c
%gamma = 1;


for k = 1:num_iter

%-------------------------------------------------> Linearization of the
 dynamics
%-------------------------------------------------> Quadratic Approximations of
 the cost function
for  j = 1:(Horizon-1)

     [l0,l_x,l_xx,l_u,l_uu,l_ux] = fnCost(x_traj(:,j), u_k(:,j), R, Q, dt,
 p_target);
      q0(j) = dt * l0;
      q_k(:,j) = dt * l_x;
      Q_k(:,:,j) = dt * l_xx;
      r_k(:,j) = dt * l_u;
      R_k(:,:,j) = dt * l_uu;
      P_k(:,:,j) = dt * l_ux;

    [dfx,dfu] =
 fnState_And_Control_Transition_Matrices(x_traj(:,j),u_k(:,j),du_k(:,j),dt,
 h_0, hx1_0, hx2_0, hx3_0, gamma);

    A(:,:,j) = eye(13,13) + dfx * dt;
    B(:,:,j) = dfu * dt;
end


%-------------------------------------------------> Find the controls
Vxx(:,:,Horizon)= Q_f;
Vx(:,Horizon) = Q_f * (x_traj(:,Horizon) - p_target);
V(Horizon) = 0.5 * (x_traj(:,Horizon) - p_target)' * Q_f * (x_traj(:,Horizon)
 - p_target);
```

```
%------------------------------------------------> Backpropagation of the
 Value Function
for j = (Horizon-1):-1:1

   H = R_k(:,:,j) + B(:,:,j)' * Vxx(:,:,j+1) * B(:,:,j);
   G = P_k(:,:,j) + B(:,:,j)' * Vxx(:,:,j+1) * A(:,:,j);
   g = r_k(:,j) +  B(:,:,j)' * Vx(:,j+1);


   inv_H = H\eye(4);
   %feedback
   L_k(:,:,j)= - inv_H * G;
   %feedforward
   l_k (:,j) = - inv_H *g;

   % TODO: add the corresponding new ones

   % Old ones
   Vxx(:,:,j) = Q_k(:,:,j)+ A(:,:,j)' * Vxx(:,:,j+1) * A(:,:,j) + L_k(:,:,j)'
 * H * L_k(:,:,j) + L_k(:,:,j)' * G + G' * L_k(:,:,j);
   Vx(:,j)= q_k(:,j) +  A(:,:,j)' *  Vx(:,j+1) + L_k(:,:,j)' * g + G' *
 l_k(:,j) + L_k(:,:,j)'*H * l_k(:,j);
   V(:,j) = q0(j) + V(j+1)   +   0.5 *  l_k (:,j)' * H * l_k (:,j) + l_k
 (:,j)' * g;
end

%------------------------------------------------------> Forward Propagaion:

%--------------------------------> Find the controls/ forward
dx = zeros(13,1);
for i=1:(Horizon-1)
   du = l_k(:,i) + L_k(:,:,i) * dx;
   dx = A(:,:,i) * dx + B(:,:,i) * du;
   u_new(:,i) = u_k(:,i) + gamma * du;
end

u_k = u_new;


%--------------------------------> Simulation of the Nonlinear System
[x_traj] = fnSimulate(xo,u_new,Horizon,dt, h_0, gamma);
[Cost(:,k)] =  fnCostComputation(x_traj,u_k,p_target,dt,Q_f,R,Q);
x1(k,:) = x_traj(1,:);


fprintf('DDP Iteration %d,  Current Cost = %e \n',k,Cost(1,k));
end %% end iterating over the algorithm

uDDP = u_new;
end

Not enough input arguments.
```

```
Error in calcDDP (line 6)
u_k = ones(4,Horizon);
```

Published with MATLAB® R2021b