file controller_4.m

```
function new_force =
controller(system_state_parameter_a,system_state_parameter_b,
           system_state_parameter_c,system_state_parameter_d)

% file controller_4.m

% The 5^4-FLC accepts 4 input variables from the simulator
% (angle, angular velocity, cart position and cart velocity)
% and fuzzifies them with 4 sets of 5 triangular membership functions.
% It then uses a 54 T-norm (minimum) rule-base with 17 consequent functions
% to determine the overall output.

%    Larry Bush December 4, 2001
%    email: bushl2@rpi.edu
%    Student ID: 660 220 742

% Project  :  Fuzzy Logic Controler for the Inverted Pendulum Problem
% Soft Computing
% Instructors:  Kai Goebel / Bill Cheetham

%
% ********************************************************
% ***********    5^4 FUZZY LOGIC CONTROLLER     ***********
% ********************************************************


% ********************************************************
% ***********    Input Membership Functions     ***********
% ********************************************************

% Determine membership to each input state parameter.

for i=1:5,
    membership_a(i)=mf4b(system_state_parameter_a,'angle',i);
    membership_b(i)=mf4b(system_state_parameter_b,'angular_velocity',i);
    membership_c(i)=mf4b(system_state_parameter_c,'cart_position',i);
    membership_d(i)=mf4b(system_state_parameter_d,'cart_velocity',i);
end


% ********************************************************
% ***********    Calculate Firing Strength      ***********
% ********************************************************

% min of each combintaion
% w is the combined firing strength for the rulebase.


    w = zeros(5,5,5,5);
    for i = 1:5,
     for j = 1:5,
```

```
      for k = 1:5,
       for l = 1:5,
            w(i,j,k,l) =
min(min(membership_a(i),membership_b(j)),min(membership_c(k),membership_d(l)));
       end
      end
     end
    end


% ********************************************************
% **************        consequence     ******************
% ********************************************************

%   consequence mf parameters
    consequence_mf = zeros(5,5,5,5);

% ******************************
% sub consequence mf parameters
% ******************************
    factor = 2.7;%

    high =  2;
    low  = -2;
    n=17;
% ******************************

% This section automates part of the consequent mf construction.
% It takes the above input parameters and generates a set of parameters
% to be used by st45 function.

third = (high-low)/(3*n-(n-1));

cp = zeros(n,2);
for i=1:n;
    bottom =  low+(i-1)*2*third;
    top    =  bottom+3*third;
    cp(i,:) =  [bottom top].*factor;
end


% This comment table shows two dimensions of the rulebase.
% The other two dimensions are implied by the side drawing.

%               angle(nl--pl)
%               -------------
%         1     2     3     4     5
%       -------------------------------
%cmf = [cp(1),cp(2),cp(3),cp(4),cp(5);%  1          %     /
%       cp(2),cp(3),cp(4),cp(5),cp(6);%  2  angular %    /  t  .
%       cp(3),cp(4),cp(5),cp(6),cp(7);%  3  velocity%   /  r  s    - -   cart
%       cp(4),cp(5),cp(6),cp(7),cp(8);%  4          %  /  a  o   - - velocity
%       cp(5),cp(6),cp(7),cp(8),cp(9)];% 5          % /  c  p
```

```
      % This loop sets the rule base parameters.
      cmf = zeros(5,5,5,5,2);

      for i = 1:5,
        for j = 1:5,
          for k = 1:5,
            for l = 1:5,
                cmf(i, j, k, l, :) = cp((i-1)+j+(-k+5)+(-l+5),:);
            end
          end
        end
      end

      % consequence_mf
      % This loop generates the consequent distributions over z.
      % It generates a 625 element table of distributions.
      consequence_mf = zeros(5,5,5,5,40);
      z = linspace(-6, 6, 40); % this must exceed the consequence mf space width

      for i = 1:5,
        for j = 1:5,
          for k = 1:5,
            for l = 1:5,
            consequence_mf(i, j, k, l, :) = st45_mf(z, cmf( i, j, k, l, : ) );
            end
          end
        end
      end

% qualified_consequence_mf calculation

% This loop cacluates the qualified consequent mf
% and puts it into a set of rows of vectors.

% Basically, it lops the top off of the consequence mf
% at the firing strength w.

% The consequence_mf for each w is a trapazoid membership function
% i.e. qualified_consequence_mf(:, :) = min(w, consequence_mf);

    base_5 = 0;
    for i = 1:5,
      for j = 1:5,
        for k = 1:5,
          for l = 1:5,
              base_5 = base_5 + 1;
%             qualified_consequence_membership function
              qualified_consequence_mf( base_5, :)
               = min(w(i,j,k,l),consequence_mf(i, j, k, l, :) );
          end
        end
      end
    end
```

```
% overall output mf = max(qualified_consequence_mf)
  overall_output_mf = max(qualified_consequence_mf);

% defuzzify overall output mf to get the output force
  new_force = coa(z,overall_output_mf);

% ******************************************************
```