```
function EP = Euler2EP( seq_string, angle_vector )
%Euler2EP Turn an Euler Angle set into EPs
%
fcnPrintQueue(mfilename('fullpath'))
EP = zeros(4,1);

% Get the DCM first
DCM = Euler2DCM(seq_string, angle_vector);

%Sheppard's method:
%Compute Bi^2 terms
B0_sq = (trace(DCM)+1)/4;
B1_sq = (1 + 2*DCM(1,1) - trace(DCM))/4;
B2_sq = (1 + 2*DCM(2,2) - trace(DCM))/4;
B3_sq = (1 + 2*DCM(3,3) - trace(DCM))/4;

% Find the largest value, assume its sqrt is negative, then use eqn 3.96 in
% S&J to solve for the rest.
largest = max([B0_sq; B1_sq; B2_sq; B3_sq]);
if B0_sq == largest
    B0 = sqrt(B0_sq);
    B1 = (DCM(2,3)-DCM(3,2))/4/B0;
    B2 = (DCM(3,1)-DCM(1,3))/4/B0;
    B3 = (DCM(1,2)-DCM(2,1))/4/B0;
elseif B1_sq == largest
    B1 = sqrt(B1_sq);
    B0 = (DCM(2,3)-DCM(3,2))/4/B1;
    B2 = (DCM(1,2)+DCM(2,1))/4/B1;
    B3 = (DCM(3,1)+DCM(1,3))/4/B1;
elseif B2_sq == largest
    B2 = sqrt(B2_sq);
    B0 = (DCM(3,1)-DCM(1,3))/4/B2;
    B1 = (DCM(1,2)+DCM(2,1))/4/B2;
    B3 = (DCM(2,3)+DCM(3,2))/4/B2;
elseif B3_sq == largest
    B3 = sqrt(B3_sq);
    B0 = (DCM(1,2)-DCM(2,1))/4/B3;
    B1 = (DCM(3,1)+DCM(1,3))/4/B3;
    B2 = (DCM(2,3)+DCM(3,2))/4/B3;
end

EP = [B0; B1; B2; B3];

end
```

Published with MATLAB® R2013b