Problem: Implement ICP

Your assignment is to implement a simplified ICP algorithm and test it on synthetic data.

I have set up a Scenario Class “HW3Scenario.m” with some useful methods (see below) and a MATLAB script “HW3Script.m”. The script creates an instance of the Scenario and then repeatedly calls a test method. The test method, in turn, calls a function HW3ICP:

```
function [ F, Cp, Cd ] = HW3ICP(H,Samples)
%  H       = Homework 3 scenario (used so you can call match)
%  Samples = Samples (Made by H.Sample( ... )
%  Finit   = Initial guess of transformation
%
global UseICP

if exist('UseICP','var') && UseICP
   % … stuff here
end

F   = Frame.eye();
[Cp,Ct,Cd] = H.InitialMatch(Samples);

%
```

```
% Students should write their own ICP code here to compute F, Cp, Cd
%
end
```

You should modify the HW3ICP function and run the HW3Script. You should feel free to modify the latter. Note that if you set the global variable UseICP to true, the script will run the ICP code in the CIS I library rather than your code. This code is not an industrial strength version, and it has only moderate accuracy, but it will give some indication of what your code should do.

In addition to the MATLAB classes and functions in the Cartesian folder, you have the following function available to you:

```
F=FindBestRigidTransformation(A,B)
% Given vct3Array A and vct3Array B
% Returns a frame F that minimizes (F*A-B)^2
```

The HW3Scenario class contains methods for simulating the surface sampling process, for performing the matching step of ICP and a Test driver program:

```
HW3 = HW3Scenario();
HW3 = HW3Scenario('HW3ScenarioMeshTree'); % Faster
```

```
% Creates a MATLAB Class with some methods to use for the assignment

[Diff,RE,ErrMax] = H.Test(Fnom,Ns,noise,Po)
%          % Samples data and calls both student program and my program
%

Samples = H.Sample(H,Fnom,Ns,noise,Po)
%          % Returns random samples from mesh surface with noise added
%          % Fnom = nominal pose of surface mesh
%          % Ns = Number of good samples
%          % Po = Probability of a sample being an outlier


[cp,ct,cd] = H.Match(TransformedSamples, ... vct3Array of samples
%                    prev_cp,           ... vct3 array of corresponding closest points
%                    prev_ct,           ... integer array of triangle indices for prev_cp
%                    prev_ct)           ... real array of distances for prev_cp
%   Performs recursive search through the tree and returns
%    cp       = vct3Array of closest points on mesh to transformed samples
%    ct       = integer array of mesh triangle indices corresponding to cp
%    cd       = distances corresponding to closest points

[cp,ct,cd] = H.InitialMatch(Samples)     % initial samples for H.Match
%     Samples = set of samples

[MedianErr, MeanErr, RMSErr, MaxErr] = DistanceStats(Cd)
```

```
%   Computes some useful statistics on Cd
```