CS164: Computing with Physical Objects Handout # 4

Algorithms for Shape and Motion

Stanford University Wednesday, 29 April 2009

Homework #2: Shape Registration and Matching [80 points]

Due Date: Wednesday, 13 May 2009

For those working in teams (of up to three students), note that while separate write ups are required for the paper-and-pencil (theory) problems, a single write up per team is sufficient for the programming problem.

- **The Theory Problems**

Problem 1. [10 points]

Let $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ be two sets of n points each in $\mathbb{R}^3$. Prove that the optimal translation for aligning a_1 with b_1, a_2 with b_2, …, and a_n with b_n (in the MSE sense) is the translation that moves the centroid of A to that of B.

Problem 2. [20 points]

This problem examines the ICP algorithm for points in the one dimensional real line $\mathbb{R}$. In this simple one-dimensional setting, the only relevant transformation is a translation. As above, let A and B denote sets of n points each. We use ICP to move around the set A and align it with B.

Questions:

(a) Give an example configuration of two point sets A and B in which ICP would not move A at all, yet A and B are quite far from being optimally aligned. This shows that when ICP terminates it need not have converged to the global minimum of the MSE error function.

(b) Assuming that ICP does converge to a (local) minimum, argue that the number of steps (translations) taken along the way is at most $O(n^2)$.

It is not part of the problem, but it is also interesting to know that the ICP translations in this setting are always in the same direction — in other words, the algorithm never backtracks.

Problem 3. [10 points]

We saw in class that if S is a surface, x a point is space and p the nearest point to x on the surface S (the footpoint of x), then the squared distance function of x to S has an especially simple quadratic approximant in the neighborhood of x, when expressed in the frame centered at the footpoint p and aligned with the principal curvature directions and the normal at p. That formula is

$$D^2(x, S) \approx \frac{d}{d - \rho_1} x_1^2 + \frac{d}{d - \rho_2} x_2^2 + x_3^2,$$

where d denotes the distance from x to p (and therefore S) and ρ_1, ρ_2 denote the principal curvatures of S at p respectively.

Let y denote a point very near x. Use this formula to show that if $d \rightarrow \infty$, then the distance from y to S is well approximated by the distance of y to p, while if $d \rightarrow 0$, then the distance from y to S is well approximated by the distance from y to the plane tangent to S at the footpoint p. Thus the above formula encompasses both the point-to-point and the point-to-plane ICP variants mentioned in the lecture.

Problem 4. [10 points]

In order to match two shapes A and B, we typically (1) find feature points independently on each of A and B, (2) discover potential correspondences between features of A and those of B by comparing local descriptors of the surface shape around the feature points, and finally (3) try to bring the two shapes into registration by optimally aligning corresponding sets of features.

In general each feature of one shape may have several matching features in the other shape. This creates a combinatorial optimization problem: we need to select that set of corresponding feature pairs (with one feature from A and one from B in each pair) which leads to the best alignment of A and B. Several methods have been proposed to address this problem.

(a) We discussed in class two such methods, RANSAC and Geometric Hashing. Papers describing these methods are accessible from the class web page. Briefly describe and contrast the use of these two methods in the solution to the above problem, pointing out the advantages and drawbacks of each.

(b) How can Euclidean distances between features be used to speed up these combinatorial searches?

- **The Programming Problem**

Problem 5. [30 points]

For this assignment, you are required to implement the ICP algorithm, and use it to find the optimal alignment between two sample shapes. Your grade will depend on the quality of the final alignment as well as on your understanding of the algorithm.

In the “Programming” section of the course web-page, you will find two archive files, which contain libraries for Windows and Linux-based systems. Both of these archives have a directory called “data” containing a Model and a Scan file. These files contain the coordinates of the points of the Stanford Bunny model, and of a partial scan of the same model. Your job is to find the optimal translation and rotation, **t** and **R**, that will best align the Scan to the Model shape. In other words, the Model shape is fixed, while the Scan is allowed to move.

First, you need to estimate using ICP the translation vector $\mathbf{t}_{\text{opt}}$ and the rotation matrix $\mathbf{R}_{\text{opt}}$ which minimize the distance:

$$\mathbf{t}_{\text{opt}}, \mathbf{R}_{\text{opt}} = \arg\min_{\mathbf{t},\mathbf{R}} \sum_{i=1}^{|S|} \|\mathbf{R}x_i + \mathbf{t} - y_i\|_2^2,$$

where S is the set of points on the Scan shape, and y_i is the point on the Model shape closest to $\mathbf{R}x_i + \mathbf{t}$.

In addition to computing $\mathbf{t}_{\text{opt}}$ and $\mathbf{R}_{\text{opt}}$, please provide the number of iterations after which your program converged, and a plot of the residual error at each iteration. In other words, plot the value of

$$e_j = \sum_{i=1}^{|S|} \|\mathbf{R}_j x_i + \mathbf{t}_j - y_i\|_2^2,$$

as a function of j, where j is the iteration number of the ICP algorithm. The first value in this plot should be the error you obtain when using $\mathbf{R} = \text{Id}$ and $\mathbf{t} = 0$, and the last value should correspond to the value of the function that you obtain when you use $\mathbf{t}_{\text{opt}}$ and $\mathbf{R}_{\text{opt}}$. Note that $e_i \geq e_{i+1}$ for all i.

You also need to write a short report (1-2 pages) about the implementation of the algorithm and the behavior that you observed. One report per team is sufficient. The purpose of this report is to show your understanding of the ICP algorithm and its convergence. For example, did the error decay linearly, or quadratically? Did the convergence speed increase in the vicinity of the solution? In preparing the write-up, you can also experiment on shapes other than the ones we provided, or try different initial poses of the given shapes. How far do the shapes need to be for ICP to converge to a wrong local minimum of the residual function?

Please provide also the source-code of your program, as described on the course web-site. However, your grade will be based on the write-up and the results you give. The code will only be used to test the correctness of your implementation.