CS612 Homework Assignment 5

Due May 9, 2023, on Gradescope

1. **Term project:** Based on the one-hot encoding from last HW assignment, write code to read a file with a multiple sequence alignment (such as the ones I linked on the course webpage), and create a matrix with the one-hot encoded sequences. Attach your code.

2. **Term project:** Write a (concise, but not too short) progress report about your term project. Describe what you have done so far, what difficulties (if any) you ran into. Detail your choice of programming language, implementation etc. Even if you haven't started yet (which you should have!) write about your plans for starting.

3. **Geometric hashing:** For geometric hashing, the kind of transformation we allow determines the number of variables needed to establish a reference frame. For each of the following transformation types, determine the number of independent variables needed to establish a reference frame.

   (a) 2-dimensions, translation.

   (b) 2-dimensions, translation and rotation.

   (c) 3-dimensions, translation.

   (d) 3-dimensions, translation and rotation.

   (e) 2-dimensions, scaling.

4. **Geometric hashing:** To keep things simple, we will only consider a two-dimensional space. Using a regular grid, the space is divided up into a series of squares of width w. Each square becomes a "bin". and the bins are indexed as though they are a two-dimensional array. When data points are read, they are assigned to bins. An example, with w = 8 is shown here in Figure 1 and explained below. The square containing four points and bounded on the left by 8, on the right by 16, on the bottom by 0 and on the top by 8 is bin (1, 0). The bin containing three points bound on the right by -8, on the left by -16, on the bottom by 8 and on the top by 16 is bin (-2,1) etc. In general, a bin index (i,j) given coordinates (x,y) is $i = \lfloor x/w \rfloor$ and $j = \lfloor y/w \rfloor$.

   (a) How are the other non-empty bins indexed and how many points are in each one?

   (b) Given the following hash function: $h(i, j) = |i * 378551 + j * 63689|$, what are the hash values and bin indices for the following pairs: (8,10), (-6,3), (27,-18)?

5. **Reference Frame in 2D:** Given the coordinates for the following two points: $p_1 = (16, 23)$ and $p_2 = (30, 6)$.

   (a) Calculate the translation (a 2D vector) and rotation (one angle) associated with this reference frame.

   (b) Calculate the coordinates of the following points:
   - $p_2 = (30, 6)$
   - $p_3 = (10, 10)$
   - $p_4 = (32, 15)$

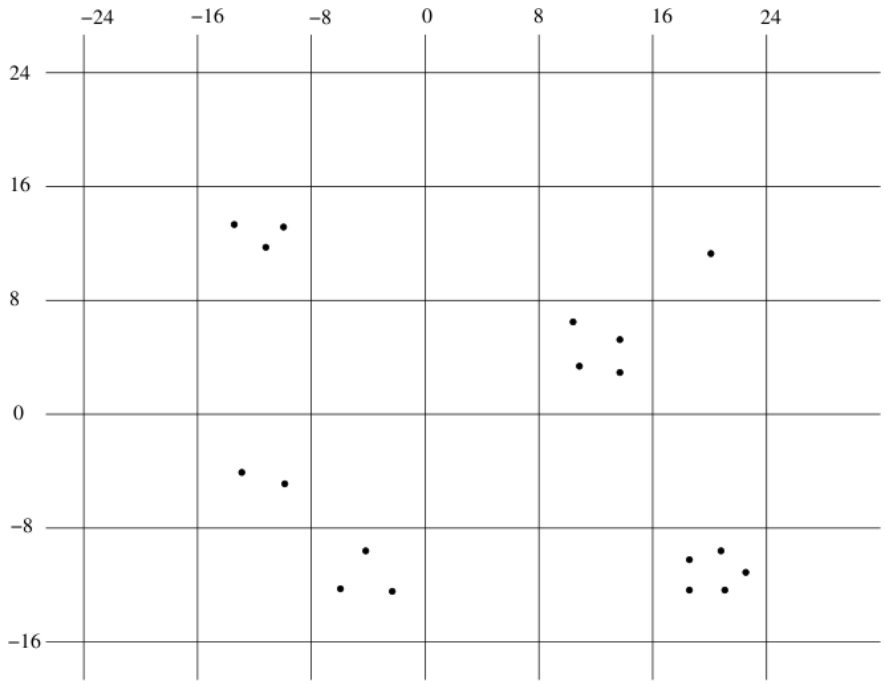

Figure 1: Hash table example

Remember that p_2 should be lying on the x axis! This is a way to check whether your calculations are right.

6. **Docking:** ClusPro is one of the state-of-the-art docking programs. Go to the server at https://cluspro.bu.edu/. As a receptor select trypsin (PDB code 2PTN). Leave the "chains" window empty – it only has one chain. As a ligand select trypsin inhibitor (PDB code 1BA7). Select chain B. No need to upload anything, the server searches in the PDB. Hit "dock". I recommend to open an account so that you'll get a notice when the job is ready. Notice that it may take a while (even a couple of hours) because the jobs run on a server and the processing time depends on the queue size.

   (a) When the job is finished go to the results page. You will see an illustration of the top 10 models. Please include a screenshot of the page in your submission file.

   (b) Comparison to the "real" complex. The above example is what is called *unbound docking* – docking two proteins that are similar, but not the same, as the "real" complex, reflecting the fact that upon binding the two structures may change a bit (making the problem more difficult). Download the solutions and unzip the file. The results are named model.000.*.pdb. Download a docked trypsin-inhibitor complex – 1AVW. To estimate the quality of the docking (the RMSD of the complexes from the native structure) we have to resort to a quick and dirty solution. The goal is to bring the native complex and the solution complexes to the same number of atoms.
   - Upload all the solutions (model.000*.pdb) onto vmd .
   - Save their backbone only as a .crd file. Do it by clicking $File \rightarrow Save\ coordinates$ and select crd as file format and "backbone" as the selection of atoms to save. Call your file complexes.crd . The crd file contains only the atomic coordinates, no amino acid or atom information.
   - Save the backbone of the native file, 1avw.pdb, the same way, only as a pdb file. Call it 1avw_bb.pdb
   - Manually remove from 1avw_bb.pdb residues 627 and 677 of chain B, as well as the last atom of chain A (that's an OXT – terminal oxygen). These are the amino acids that appear in the

native file but not in the docked ligand 1BA7, and for RMSD calculations VMD requires all frames to have the same number of atoms.

- Upload 1avw_bb.pdb onto vmd. Using $File \rightarrow Load\ data\ into\ molecule$, upload the complexes.crd file you saved above. You should have 11 frames: The native file and the 10 solutions.
- Compare the complexes: Select $Extensions \rightarrow Analysis \rightarrow RMSD\ trajectory\ tool$. Click "align" to optimally align all the structures against the first frame which is native reference. Check the "save" button and click RMSD on top. The file trajrmsd.dat will contain the RMSD between the first frame and the other 10. Copy it to your submission.

(c) Now that you've done that, take a look at the native complex vs. the 10 complexes. Which one has the smallest RMSD to the native complex? Which one has the largest?