Project – 20201

Topics in Data structure, Algorithm, C/C++ Programming

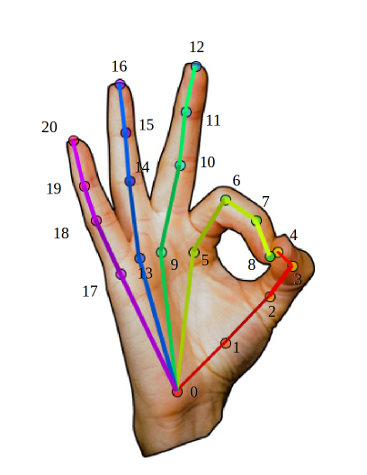
**General information**

1. **Group:** about 3-4 students
2. **Topic:** Students can choose one among following topics. They can also propose their own topics**.** A topic can be done by several groups.
3. **Proposed topics must:**

* Relate to a practical application.
* Use fundamental structures such as array, graph, tree, stack, queue.
* Read data from files.
* Pre-process or filter data.
* Process data (feature extraction, distance, sorting, search).
* Can use third-party libs for displaying/reading data but all other processing must be implemented.

**Project 1: Hand pose**

Hand pose is a set of connected joints as illustrated in the below figure. Data of each joint hand pose is usually captured by a sensor. Each joint is a point in 3D space with 3 coordinates (x, y, z). A hand pose with 21 joints will be stored as a row of 64 elements of a text file: the first element is the ID of hand pose (normally corresponding to the time stamp) and the following 63 elements are 21x3 coordinates of 21 joints. Hand pose are widely used for hand gesture recognition in many applications such as human machine interaction (Virtual reality, Augmented Reality) or Robotics (grasping), etc.



In the below row, 0000 is ID of the hand pose, and following are 63 values corresponding to coordinates (x,y,z) of 21 joints.

**0000** 77.625400 87.232600 383.293300 92.583300 83.007100 380.632400 96.910600 57.860300 455.163700 75.404300 62.390400 456.142000 58.042900 68.121300 450.971100 42.588400 72.286700 449.058200 85.970700 40.845400 415.326200 81.564400 15.438700 437.486600 75.748300 0.170600 460.225600 89.101500 24.760600 479.809700 73.605100 1.479900 478.441300 62.161900 -3.340400 462.762700 50.971200 20.429900 484.792100 33.221300 -10.053300 505.605500 20.259600 -30.249700 505.921400 30.540800 27.833500 479.278200 8.576500 -4.341800 501.885400 -2.598700 -25.411400 509.381100 16.341500 49.211200 475.110600 -4.836900 30.591700 496.132100 -15.817000 13.362300 504.292200

When the hand performs a dynamic gesture, the pose of hand varies, each time the hand pose coordinates will be stored as a row in a file.

**0000** 77.625400 87.232600 383.293300 92.583300 83.007100 380.632400 96.910600 57.860300 455.163700 75.404300 62.390400 456.142000 58.042900 68.121300 450.971100 42.588400 72.286700 449.058200 85.970700 40.845400 415.326200 81.564400 15.438700 437.486600 75.748300 0.170600 460.225600 89.101500 24.760600 479.809700 73.605100 1.479900 478.441300 62.161900 -3.340400 462.762700 50.971200 20.429900 484.792100 33.221300 -10.053300 505.605500 20.259600 -30.249700 505.921400 30.540800 27.833500 479.278200 8.576500 -4.341800 501.885400 -2.598700 -25.411400 509.381100 16.341500 49.211200 475.110600 -4.836900 30.591700 496.132100 -15.817000 13.362300 504.292200

**0001** 67.376300 91.927100 387.268800 82.469800 87.748000 385.426800 82.653800 62.119800 459.919500 61.116200 66.602100 459.715000 44.060200 72.333500 453.612200 28.728400 76.481700 450.859500 74.028500 44.865400 419.145300 68.779800 18.201200 440.111400 61.923700 1.643200 461.624800 70.213000 26.416100 478.208100 55.717700 2.472100 478.963600 44.509000 -1.792200 462.958000 32.769000 23.300200 487.481800 11.463600 -9.244800 508.350900 -1.697600 -29.306700 508.896900 12.887600 31.581000 481.122700 -12.439200 -1.529300 503.474200 -24.159500 -22.201700 511.238700 0.267100 54.398300 475.104000 -22.650400 36.616400 494.626100 -33.975200 19.614000 502.791100

**0002** 58.309900 94.280500 389.954000 73.308900 89.746800 388.179400 72.256600 63.257500 462.363000 50.839700 68.279200 462.045100 33.981400 74.506000 455.878900 18.780200 79.068300 453.057000 63.158300 46.492600 420.780300 56.863800 19.669800 440.996700 49.482300 2.120900 461.528300 54.505100 26.417300 472.538000 42.621500 1.755000 479.349500 30.973000 -2.365600 463.622600 20.268800 23.303900 491.049800 -3.507000 -11.674600 513.607500 -16.954600 -31.448700 515.644800 -1.230900 31.116400 484.463500 -30.850800 -5.382200 508.508300 -42.969800 -25.644900 516.727500 -12.838200 57.527400 478.777800 -39.065200 39.659700 500.112800 -50.518700 23.137800 509.048400

1. Define a data structure **struct** hand Joint to represent a joint of a hand pose with following information.
   * + 3 coordinates: x, y z
     + ID of joint
     + An array of joints is connected to it. For example, joint 0 has 5 connected joints (1, 5, 9, 13, 17), joint 1 has 2 connected joints (0, 2)
2. Define a data structure **struct** handPose to represent a hand pose with following information:
   * + Timestamp of the hand pose
     + Array of 21 joints, each joint is a **struct handJoint** defined in (1)
3. Write a function to read row by row a txt file as described above, return a pointer that points to an array of **struct** handPose storing handPoses charged from the file. Notice that we don’t know the number of rows in the txt file.
4. Write a function that calculates the Euclidian distance of every pair of joints of a hand pose. The function returns a pointer that points to 2D array, each element of the array (i,j) is Euclidian distance between ith joint and jth joint. It would be a matrice of size (21x21)
5. Write a function that calculates the movement of each joint between two times. The function takes two handPoses at timestamp t1 and t2 as input, returns a pointer that points to an 1D array of size 21 elements, each element i of the array is the Euclidian distance of the ith joint at timestamp t1 and ith joint at timestamp t2
6. Write a function to **sort** the array obtained from (5) and return the sorted array and the ID of the joint that moves the most.
7. If we represent a handpose as a binary tree, define a struct Node to represent each joint of hand pose
8. Write a function to insert nodes into the tree, returns the pointer that point the root node (corresponding to 0th joint)
9. Write a function to traversal the tree in pre-order, in-order and post-order
10. Write a function that determines longest path of the tree. The function returns to the array storing the ID of node belonging to the longest path.
11. Phần 6 ý tưởng là sẽ tạo 1 cái struct để tạo ra 21 giá trị con theo dõi cái mảng 21 phần tử
12. Phần từ 7-10 thì duyệt sang phải vẫn còn có vấn đề

**Project 2: Human pose**

Human pose is a set of connected joints as illustrated in the below figure. Data of each joint human pose is usually captured by a sensor. Each joint is a point in 3D space with 3 coordinates (x, y, z). A human pose with 20 joints will be stored as a row of 61 elements of a text file: the first element is the ID of hand pose (normally corresponding to the time stamp) and the following 60 elements are 20x3 coordinates of 20 joints. Human pose are widely used for human activity recognition in many applications such as life logging, healthcare, gaming.

|  |  |
| --- | --- |
| Skeleton joint points tracked by Kinect Sensor | Download Scientific Diagram |  |

1. Define a data structure **struct** humanJoint to represent a joint of a human pose with following information
   * 1. 3 coordinates: x, y z
     2. ID of joint
     3. An array of joints connected to it. For example, joint 0 has 5 connected joints (1, 5, 9, 13, 17), joint 1 has 2 connected joints (0, 2)
2. Define a data structure **struct** humanPose to represent a human pose with following information:
   * 1. Timestamp of the human pose
     2. Array of 20 joints, each joint is a **struct humanJoint** defined in (1)
3. Write a function to read row by row a txt file as described above, return a pointer that points to an array of **struct** humanPose storing humanPoses charged from the file. Notice that we don’t know the number of rows in the txt file.
4. Write a function that checks if the humanPose is a symmetric (length of corresponding bone connecting to a pair of joints is **similar (smaller than a given threshold)** in left and right side of the human. The function will return 1 if all bones satisfy the similarity condition, 0 otherwise
5. Write a function that calculates the movement of each joint between two times. The function takes two humanPoses at timestamp t1 and t2 as input, returns a pointer that points to an 1D array of size 21 elements, each element i of the array is the Euclidian distance of the ith joint at timestamp t1 and ith joint at timestamp t2
6. Write a function to **sort** the array obtained from (5) and return the sorted array and the k- ID joints that move the most.
7. If we represent a humanpose as a general tree, the root node is the node ID = 4 define a struct Node to represent each joint of human pose
8. Write a function to insert nodes into the tree, returns the pointer that point the root node (corresponding to 4th joint)
9. Write a function to traversal the tree in pre-order, in-order and post-order
10. Write a function that converts this tree to a binary tree.