43075-01 Probabilistic Shape Modelling

Lecturers
Dr. Marcel Lüthi ⟨marcel.luethi@unibas.ch⟩

Introduction	15. March 2022
Discussion	22. March 2022

Before starting this exercise, you should have worked through Weeks 1 - 2 of the FutureLearn course. You should also read the software tutorials corresponding to the material covered in these two weeks, on `https://scalismo.org/docs`. We recommend that you use VSCode to do the practical exercises.

1. Project

1.1. Download the data

Download the project data from the Adam workspace:
`https://adam.unibas.ch/goto_adam_fold_1368991.html`

1.2. Rigid alignment

Once you have downloaded the data, you are ready for the first real step of the project: Rigid alignment of the data.
The femur shapes and corresponding landmarks you should use for alignment can be found in the downloaded zip-archive, under the folder `meshes` and `landmarks`. There is also a reference mesh with corresponding landmarks in the folder `reference-mesh` and `reference-landmarks`. Rigidly align all the meshes to the reference mesh. Visually check that the bones are well aligned. Use the relevant code snippets from the Scalismo tutorials (`https://scalismo.org/docs/tutorials/`). Particularly relevant tutorials for this task are tutorial 3 and tutorial 6.
Save all the aligned meshes in a separate folder. You may also want to transform and save the landmarks, as they might prove useful in a later step.
In order to load and save landmarks, use the methods in the class `LandmarksIO`:

```
val landmarks: Seq[Landmark[_3D]] =
    LandmarkIO.readLandmarksJson3D(new java.io.File("landmarkFile.json")).get
LandmarkIO.writeLandmarksJson(landmarks, new java.io.File("landmarkFile.json")).get
```

1.3. Analyzing the data

Visualize one bone together with the landmarks. Note that landmark `L2` and `L5` characterize the length rather well. Landmarks `l3` and `l4` are a good proxy for the width of the bone. Compute the distances between these two pairs of landmarks for every bone. Write the result in a csv file. Use your favorite programming language and plotting library to create a scatterplot of the data, where the length and the width are the two axes, and the points show the individual examples. Also compute the mean and variance for these measurements.
The following code snippet shows how to write a CSV File from Scala.

```
// Class to store the measurements
case class Measurement(id : String, length : Double, width : Double)

def writeCSV(csvFile : java.io.File, measurements :Seq[Measurement]) : Unit = {
    import java.io.PrintWriter
    val printWriter = new PrintWriter(csvFile)
    printWriter.write("id, length, width\n")
    for (measurement <- measurements) {
        printWriter.write(s"${measurement.id},
                           ${measurement.length},
                           ${measurement.width}\n")
    }
    printWriter.close()
}
```