Coursework MPHY0030 2020-2021 Part 2 Report

Student Number: 20091404

1. Question 1

The polynomial part is not needed because polynomial part is always used to guarantee the non-singularity of the matrix to get a single solution of the equation. But by using the Gaussian spline, which is positive definite, the non-singularity is already confirmed so there is no need to use a polynomial part.

2. Question 2

Because no polynomial part is needed, the linear algebra for Gaussian spline becomes:

$$(\boldsymbol{K} + \lambda \boldsymbol{W}^{-1}) \cdot \boldsymbol{\alpha} = \boldsymbol{q}_k$$

If there is no expected accuracy of the landmarks, then we can set W to an identity matrix, so

$$(\boldsymbol{K} + \lambda \boldsymbol{I}^{-1}) \cdot \boldsymbol{\alpha} = \widehat{\boldsymbol{q}_k}$$

$\widehat{\boldsymbol{q}_k}$ is the prediction of target points. To get the solution of $\boldsymbol{\alpha}$, we need to make sure the difference between $\widehat{\boldsymbol{q}_k}$ and $\boldsymbol{q}_k$ is smallest. We would use least square to solve it, that is

$$\min\left((\boldsymbol{q}_k - (\boldsymbol{K} + \lambda \boldsymbol{I}^{-1}) \cdot \boldsymbol{\alpha})^T \cdot (\boldsymbol{q}_k - (\boldsymbol{K} + \lambda \boldsymbol{I}^{-1}) \cdot \boldsymbol{\alpha})\right)$$

To get the minimum, the derivative of $\boldsymbol{\alpha}$ should be computed and set the result to 0. So, if we set $\boldsymbol{A} = \boldsymbol{K} + \lambda \boldsymbol{I}^{-1}$, then

$$\boldsymbol{\alpha} = (\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{q}_k$$

However, under our circumstances, the problem is that $\boldsymbol{A}$ could be a very huge matrix so that the total computation will be very slow. So, we will use the singular value decomposition (SVD) method to solve this least square problem.

For any matrices, such as $\boldsymbol{A} \in R^{m \times n}$, it could be decomposed to

$$svd(\boldsymbol{A}) = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$$

in which $\boldsymbol{U} \in R^{m \times n}$, $\boldsymbol{S} \in R^{m \times n}$, $\boldsymbol{V} \in R^{n \times n}$, $\boldsymbol{U}$, $\boldsymbol{V}$ are both singular vectors of $\boldsymbol{A}$, $\boldsymbol{S}$

is $\begin{bmatrix}\boldsymbol{\Sigma}\\0\end{bmatrix}$, $\boldsymbol{\Sigma}$ is a diagonal matrix, which values are singular values σ of $\boldsymbol{A}$, so

$$\|\boldsymbol{A}\cdot\boldsymbol{\alpha}-\boldsymbol{q}_k\| = \left\|\boldsymbol{U}\begin{bmatrix}\boldsymbol{\Sigma}\\0\end{bmatrix}\boldsymbol{V}^T\cdot\boldsymbol{\alpha}-\boldsymbol{q}_k\right\| = \left\|\begin{bmatrix}\boldsymbol{\Sigma}\\0\end{bmatrix}\boldsymbol{V}^T\cdot\boldsymbol{\alpha}-\boldsymbol{U}^T\cdot\boldsymbol{q}_k\right\|$$

$\boldsymbol{U}$ could be written as $[\boldsymbol{U_n}, \boldsymbol{U_{m-n}}]$, then

$$\|\boldsymbol{A}\cdot\boldsymbol{\alpha}-\boldsymbol{q}_k\| = \left\|\begin{bmatrix}\boldsymbol{\Sigma}\\0\end{bmatrix}\boldsymbol{V}^T\cdot\boldsymbol{\alpha}-[\boldsymbol{U_n},\boldsymbol{U_{m-n}}]^T\cdot\boldsymbol{q}_k\right\| = \left\|\begin{matrix}\boldsymbol{\Sigma V}^T\cdot\boldsymbol{\alpha}-\boldsymbol{U_n^T}\cdot\boldsymbol{q}_k\\-\boldsymbol{U_{m-n}^T}\cdot\boldsymbol{q}_k\end{matrix}\right\|$$

$$= \|\boldsymbol{\Sigma V}^T\cdot\boldsymbol{\alpha}-\boldsymbol{U_n^T}\cdot\boldsymbol{q}_k\| + \|\boldsymbol{U_{m-n}^T}\cdot\boldsymbol{q}_k\| \geq \|\boldsymbol{U_{m-n}^T}\cdot\boldsymbol{q}_k\|$$

The equal sign exists only when $\|\boldsymbol{\Sigma V}^T\cdot\boldsymbol{\alpha}-\boldsymbol{U_n^T}\cdot\boldsymbol{q}_k\| = 0$, which means

$$\boldsymbol{\Sigma V}^T\cdot\boldsymbol{\alpha} = \boldsymbol{U_n^T}\cdot\boldsymbol{q}_k$$

So,

$$\boldsymbol{\alpha} = (\boldsymbol{\Sigma V}^T)^{-1}\boldsymbol{U_n^T}\cdot\boldsymbol{q}_k$$

This is exactly the solution of this least square problem.

3. Question 3

There are several linear algebra algorithms to solve this spline fitting problem, such as SVD, QR decomposition, LU decomposition, etc. When using them to solve a least square problem, the best one is SVD because LU always needs the $\boldsymbol{M}$ above is inversible, QR is faster than SVD but less stable.

4. Question 4

Control points are points inside the query points voxel space with specified distribution. We cannot choose any points at evaluate stage, because they must correspond to the points at fit stage, which means they should be the same as the source points.

5. Question 5

We do not need the weighting parameter lambda at evaluate stage, because lambda is used to offset the approximated localization errors but there are no localization errors at evaluate stage. The points we use at fit stage are randomly transformed so there are always some errors between the points' actual position and points' theoretical position.

But at evaluate stage, the points we use are exactly the query points, which do not have localization errors.

6. Question 6

When the data set is very large, it is faster to compute K directly by matrices computing rather than using loop. The following is my vectorization strategy.

First, we suppose the query points' size is $m \times 3$ and control points' size is $n \times 3$. The distance will be a $m \times n$ matrix. The i row of query points is p_i, the j row of control points is q_j. The distance squared between p_i and q_j is

$$d_{ij}^2 = (p_{i1} - q_{j1})^2 + (p_{i2} - q_{j2})^2 + (p_{i3} - q_{j3})^2$$

$$= (p_{i1}^2 + p_{i2}^2 + p_{i3}^2) + (q_{j1}^2 + q_{j2}^2 + q_{j3}^2) - 2 \cdot (p_{i1}q_{j1} + p_{i2}q_{j2} + p_{i3}q_{j3})$$

$$= \|p_i\|^2 + \|q_j\|^2 - 2 \cdot p_i \cdot q_j{}^T$$

Then it is possible to extend the formula above to the i row of distance matrix. That is

$$d_i^2 = (\|p_i\|^2 \, \|p_i\|^2 \ldots \|p_i\|^2) + \left(\|q_1\|^2 \|q_2\|^2 \ldots \|q_j\|^2\right) - 2 \cdot p_i \cdot q^T$$

Naturally, we can extend the formula above to the whole distance matrix, which is

$$d^2 = \begin{pmatrix} \|p_1\|^2 & \cdots & \|p_1\|^2 \\ \vdots & \ddots & \vdots \\ \|p_m\|^2 & \cdots & \|p_m\|^2 \end{pmatrix} + \begin{pmatrix} \|q_1\|^2 & \cdots & \|q_n\|^2 \\ \vdots & \ddots & \vdots \\ \|q_1\|^2 & \cdots & \|q_n\|^2 \end{pmatrix} - 2 \cdot p \cdot q^T$$

Thus, we complete the vectorization method of computing the squared distance matrix.

7. Question 7

The Gaussian kernel parameter sigma indicates a shape parameter that can be used to scale the input of the radial kernel. The Gaussian kernels under different sigma are showed in Figure 1, in which the Gaussian kernels is "thinner" when sigma decreases. Thus, by using small sigma, the kernel values are more closed to 0 when r is large, which will lead to more difference compared with the original image.

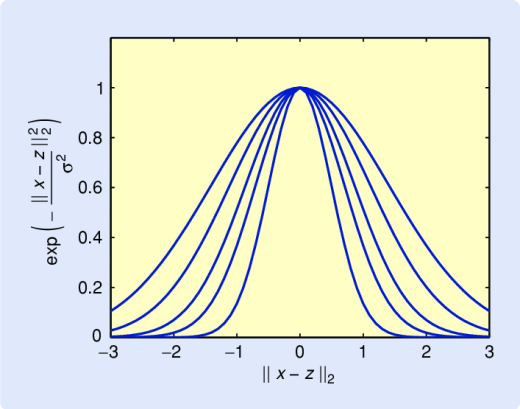

Figure 1 Gaussian kernel values under different sigma

8. Question 8

The control points represent a part of the original points' lattice space so the voxel area of control points is a part of the original medical figure, which is biophysical. To achieve a biophysically plausible transformation, the random distribution of affine transformation should be of Gaussian shape but each value in the affine matrix will have a constraint, so that there are only rotation, translation, scaling rather than reversal, shear, reflection, etc.

9. Question 9

Because of the biophysically plausible choice of control points, and biophysically plausible deformation of the control points, the interpolated voxel coordinates will also represent biophysically plausible deformation.

10. Question 10

We can compute a warped 3D image by the following steps:

1) Get a set of control points by an initialized object from class FreeFormDeformation.
2) Compute the transformed control points by using the random_transform_generator.
3) Fit the pre-transformed control points and transformed control points by using the fit function in the class RBFSpline and get spline coefficient $\boldsymbol{\alpha}$, during which the kernel values between two set of control points are computed by kernel_gaussian.

4) Initialize an object in class Image3D to represent an image which will be warped. And transform the object to lattice structure points, the query points.
5) Evaluate the query points by applying $\boldsymbol{\alpha}$ and the kernel values between query points and control points, then get transformed query points.
6) Convert the transformed query points back to an image, the warped 3D image.

11. Question 11

The 10 different randomly transformed images with 5 different z depths are in Figure2. There is surely no large deformation but just little rotation, translation and similarity,

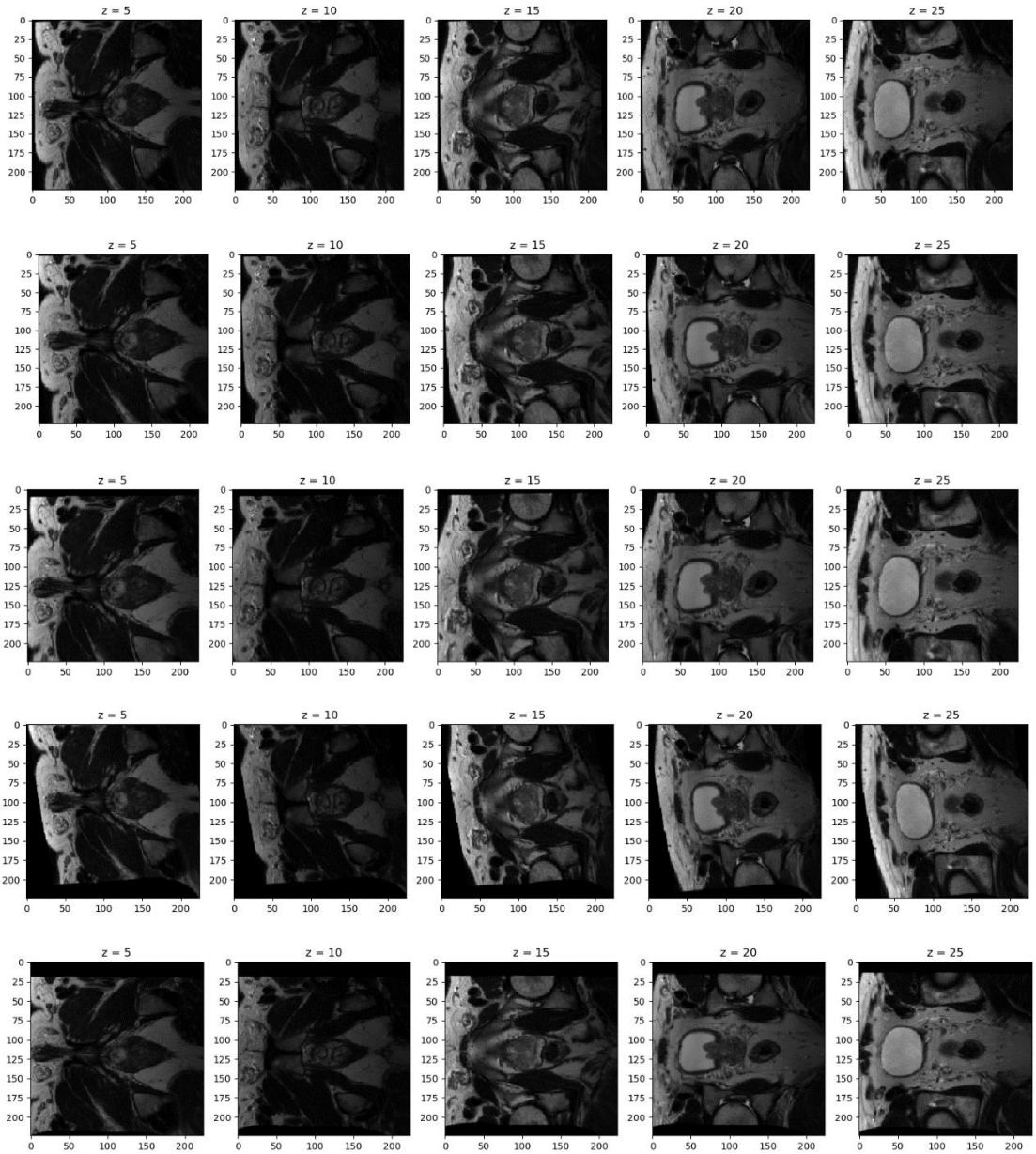

Figure 2 10 different randomly transformed images and 5 different z depth for each

which seems biophysically plausible.

12. Question 12

Figure 3 shows the results after making changes separately to the strength parameter, number of control points and the Gaussian kernel parameter. At the beginning, the z value is 5, the strength parameter is 0.5, the number of control points is 4 in each direction and Gaussian kernel parameter is 100. When we change one of these three values, the other two values maintain unchangeable.

From the top three images in Figure 3 it is clear to see that when increasing strength parameter, the images become increasingly warped and deviated from the original position. In the medium three images in Figure 3, when there are only 27 control points, the image seems less biophysically plausible and blurred at the boundary. The more control points, the clearer the whole image. The bottom three images are about sigma. As the sigma increases, the images become more and more close to the original image, which means less warped.

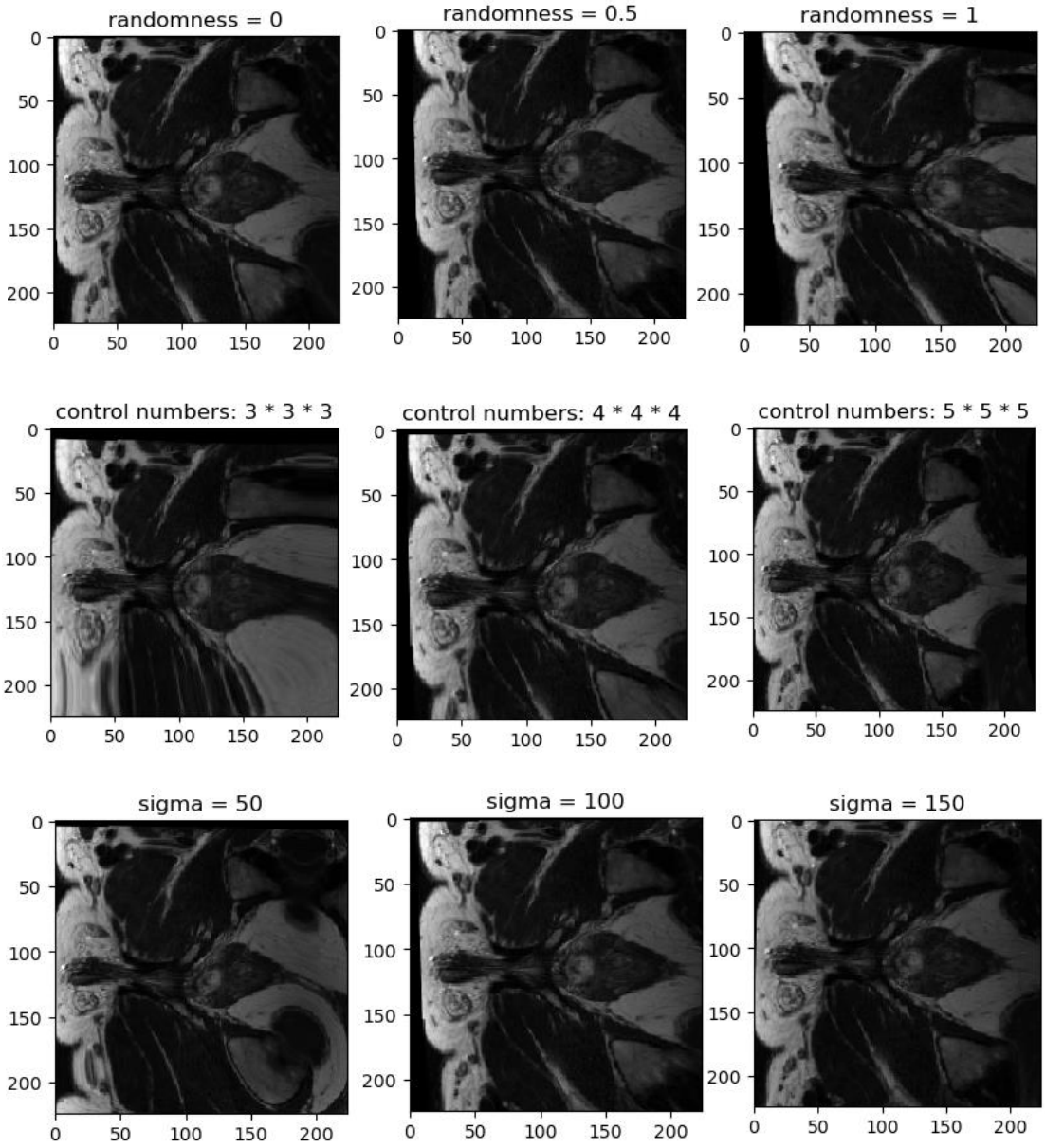

Figure 3 Top three: change randomness strength parameter
Medium three: change control number in each dimension
Bottom: change sigma values in the Gaussian kernel