Engineering Tripos Part IIA THIRD YEAR

Module 3G4: Medical Imaging & 3D Computer Graphics

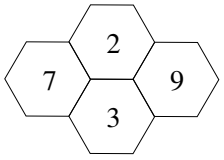
Solutions to 2021 Tripos Paper

1. **Tomographic reconstruction algorithms**

(a) Unlike filtered backprojection, iterative algorithms are easily adapted to incorporate prior knowledge into the reconstruction process. For example, the sensitivity of each projection to each attenuation element can be modelled in detail, allowing for imperfect collimation and attenuation correction when reconstructing SPECT or PET. Also, an appropriate noise model can be incorporated into ML-EM (with early stopping) or MAP iterative reconstruction, allowing for Poisson noise in PET/SPECT. The downside is their computational expense, and until recently they were only really feasible with nuclear medicine imaging, where the resolution is relatively low and the data sets are consequently not too large. However, with recent advances in low-cost, powerful computing, high resolution CT data sets can now be iteratively reconstructed in reasonable times. Compared with filtered backprojection, iterative reconstruction can reduce imaging noise by as much as 40%. The clinician can then choose whether to reduce the X-ray dose for the same image quality as filtered backprojection, or keep the same X-ray dose and obtain a better reconstruction. [30%]

(b) Because of the top right set of projections, the left hand bar has attenuation coefficient 7 and the right hand bar has attenuation coefficient 9.

Now using the top left set of projections, we can see that the top bar must have attenuation coefficient $9 - 7 = 2$. Similarly, the bottom bar must have coefficient $12 - 9 = 3$. We can check these values using the final set of projections on the row below. This gives the top bar coefficient as $11 - 9 = 2$, and the bottom bar coefficient as $10 - 7 = 3$. Hence the solution is:



[10%]

(c) The AART update formula for projection i is

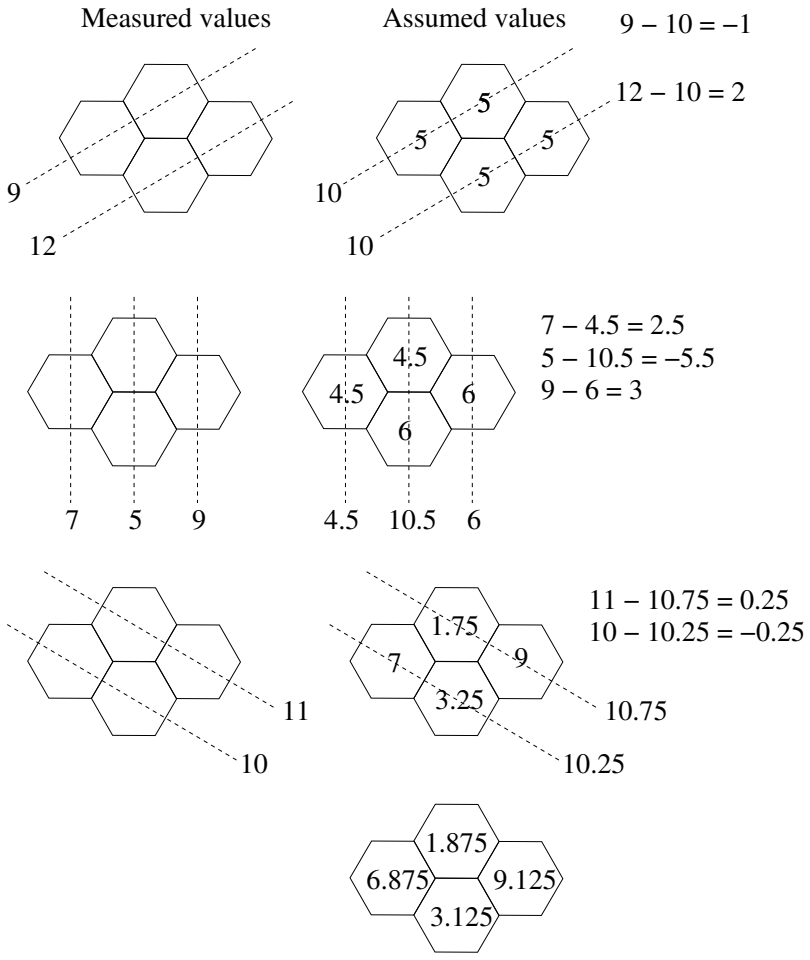
$$\mu_j^{(k+1)} = \mu_j^{(k)} + \frac{\beta^{(k)}}{N}\left(r_i - \sum_j \mu_j^{(k)}\right) \quad , \quad \text{for all } j \text{ with } c_{ij} = 1$$

where r_i is the projection data, μ_p is the linear attenuation coefficient of pixel p, c_{ij} is the sensitivity of projection i to attenuation in pixel j, k is the iteration index, β is the relaxation term and N is the number of pixels contributing to the projection.

We start by assuming a value of 5 for each rod. Along the first projection line we measure 9 but have assumed 10, giving an error of -1. This is divided by 2, because there are two

rods involved to give a correction of -0.5 for each rod. The assumed value for each of the rods involved along this projection line now becomes 4.5.

We now consider the second projection line in the same direction. In this case the measured value is 12 and the assumed value is 10. The difference is $+2$, which is again divided by two and added to the assumed values of the rods. This gives values of 6 for these rods.



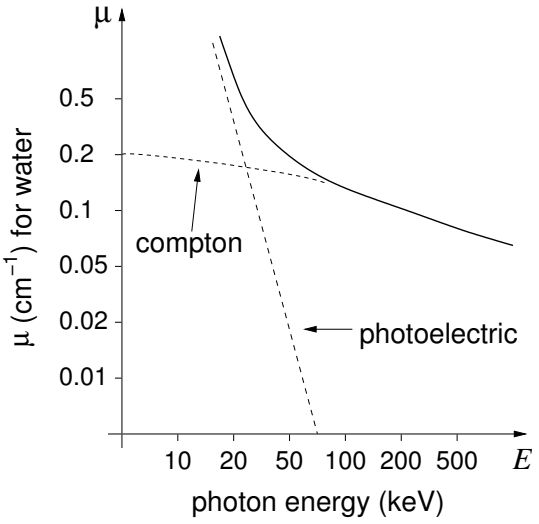
We now consider the projection in a vertical direction. The first line only passes through one rod. The measured value is 7 and the assumed value is 4.5. The error is thus 2.5. As there is only one rod involved, this value is simply added to the assumed value of the rod to give a new assumed value of 7.

We proceed to handle the second and third lines in the vertical direction in a similar way. Once this is done, we handle the third projection direction which yields values that are less than 0.2 away from the correct solution, as required.

Faster convergence may be possible by processing the projections in a different order. It is generally better to change the direction of the projections as much as possible between each set of updates. Such subtleties were not considered in the preceding solution! [40%]

(d) Tomographic reconstruction algorithms assume that the X-ray attenuation process is homogeneous. This will not be the case if the X-ray beam contains X-rays at a range of different energies (i.e. different frequencies), because each different energy component is subject to a different attenuation coefficient.

Since low energy X-rays are attenuated more than high energy X-rays, the further the beam penetrates into matter, the more its spectrum concentrates at higher frequencies and the less subsequent attenuation there is. This gives rise to "beam hardening" artefacts, which are usually apparent as streaks or cupping.



We can make the X-ray beam more mono-energetic by using an aluminium or copper beam hardening filter. Low energy X-rays are absorbed more by the filter than high energy X-rays. The resulting beam is therefore left with a higher proportion of high energy X-rays. [20%]

Assessors' remarks: This question tested the candidates' understanding of CT reconstruction, with a focus on the additive algebraic reconstruction technique (AART). Part (a) was poorly answered, with most candidates failing to engage with the question, and instead providing lengthy, generic descriptions of filtered backprojection and iterative reconstruction algorithms, in the misguided expectation that this might pick up some marks. In contrast, parts (b) and (c) were well answered, with almost all candidates demonstrating a good understanding of AART, the difficulties posed my polychromatic X-ray sources and how these difficulties might be ameliorated.

2. **Multi-segment splines**

(a) A cubic parametric curve is defined as a cubic function of a parameter t, which is 0 at the start of each segment and 1 at the end. The curve is found by multiplying a row vector parameter matrix $[t^3 \ t^2 \ t \ 1]$ by a 4×4 basis matrix and a column geometry matrix. Catmull-Rom splines and B-splines use the same geometry matrix, which is simply a vector of four control points. The curve segment is generally between the second and

third control points. Both types of spline can be used for multi-segment curves by sharing three of the control points in each geometry matrix. Hence $N + 3$ points are needed to define N curve segments.

However, the Catmull-Rom spline *interpolates* the points in its geometry matrix, whereas the B-spline does not. Each segment in a Catmull-Rom spline will start at the second control point and end at the third. Segments in a B-spline will also tend to occupy the space near the second and third control points, but do not (in general) interpolate them.

The B-spline, on the other hand, possesses better continuity, being continuous in the second parametric derivative (i.e. C^2), whereas the Catmull-Rom spline only possess C^1 continuity. The B-spline is also guaranteed to be entirely contained within the convex hull of its control points, which is not the case for the Catmull-Rom spline. [20%]

(b) (i) For the end segment, the curve s is defined as follows:

$$s = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

For the first parametric derivative, we want to use $[3t^2\ 2t\ 1\ 0]$, and at the end $t = 1$, hence:

$$s' = \begin{bmatrix} 0 & -0.5 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

So for $s' = 0$, we want $\mathbf{p}_6 = \mathbf{p}_4$. Since $s' = 0$ at this point, we need to look at the second derivative for the direction of the curve: just before $s' = 0$, the velocity must have equalled the negative of the acceleration. So the direction is $-s''$. In this case, we use $[6t\ 2\ 0\ 0]$ with $t = 1$, giving:

$$s'' = \begin{bmatrix} -1 & 4 & -5 & 2 \end{bmatrix} \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

So, given $\mathbf{p}_6 = \mathbf{p}_4$:

$$s'' = -\mathbf{p}_3 + 6\mathbf{p}_4 - 5\mathbf{p}_5 = (\mathbf{p}_4 - \mathbf{p}_3) - 5(\mathbf{p}_5 - \mathbf{p}_4) = \mathbf{a} - 5\mathbf{b}$$

The direction is actually the negative of this, $5\mathbf{b} - \mathbf{a}$ (or the normalised version of this vector, since a direction implicitly does not have a magnitude). [25%]

(ii) For the B-spline, we have:

$$s = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

At the end $t = 1$:

$$s = \frac{1}{6}\begin{bmatrix} 0 & 1 & 4 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

So, for $s = \mathbf{p}_5$, we want:

$$\mathbf{p}_5 = \frac{1}{6}\left(\mathbf{p}_4 + 4\mathbf{p}_5 + \mathbf{p}_6\right), \Rightarrow \mathbf{p}_6 = 2\mathbf{p}_5 - \mathbf{p}_4 = \mathbf{p}_5 + \mathbf{b}$$

i.e. symmetrically opposite $\mathbf{p}_5$ from $\mathbf{p}_4$. At this point, the acceleration s'' is:

$$s'' = \begin{bmatrix} 0 & 1 & -2 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

So, for $t = 1$ and $\mathbf{p}_6 = 2\mathbf{p}_5 - \mathbf{p}_4$:

$$s'' = \mathbf{p}_4 - 2\mathbf{p}_5 + \mathbf{p}_6 = 0$$

[20%]

(iii) Following the same argument as in (i), for $s' = 0$ at the end of the curve we have

$$s' = \frac{1}{6}\begin{bmatrix} 0 & -3 & 0 & 3 \end{bmatrix}\begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

So for $s' = 0$, we want $\mathbf{p}_6 = \mathbf{p}_4$, which is exactly the same as for the Catmull-Rom curve. At this point, the location is given by:

$$s = \frac{1}{6}\begin{bmatrix} 0 & 1 & 4 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{p}_5 \\ \mathbf{p}_6 \end{bmatrix}$$

But $\mathbf{p}_6 = \mathbf{p}_4$, hence:

$$s = \frac{1}{6}\left(\mathbf{p}_4 + 4\mathbf{p}_5 + \mathbf{p}_6\right) = \frac{1}{3}\left(\mathbf{p}_4 + 2\mathbf{p}_5\right) = \mathbf{p}_5 - \frac{1}{3}\mathbf{b}$$

i.e. two-thirds of the way between $\mathbf{p}_4$ and $\mathbf{p}_5$. [20%]

(iv) Scheme (i) is useful for interpolating points (since it uses a Catmull-Rom basis), but will interpolate all the way up to the last point, not just the second-to-last, without biasing the gradient at the last point. Scheme (ii) allows the use of the approximating, but smooth,

B-spline, but still forces the spline to pass through the first and last points. If defining a motion path, scheme (iii) allows the object to come to rest at the end of the curve. [15%]

Assessors' remarks: There was a wide range of answers to this question. Most candidates understood how to construct multi-segment splines, though some did not discuss how to rearrange the geometry matrices between each segment. The first part of (b)(i) was generally answered well, though many candidates did not substitute $t = 1$ until quite late, making the calculations more complex than they needed to be. The last part of (i), calculating the direction when the velocity was zero, was the most difficult part of the question and only occasionally answered correctly. (b)(ii) and (iii) were mostly answered well, and there were some good discussions in (iv).

3. **Radial Basis Functions**

(a) Nearest-neighbour interpolation simply involves selecting the closest measured value at each location. For rectilinear two-dimensional data, bi-linear interpolation involves using a weighted sum of the data at the corners of the measurement rectangle surrounding each point. These values are weighted according to the x and y distance from each corner (normalised to unit distance across the rectangle), with the weight in each case as $(1 - x)(1 - y)$ so that this equals 1 at the measurement locations.

For unstructured data, a triangulation needs to be imposed on the data first, for instance using Delaunay triangulation, and then bi-linear interpolation can be used within each triangle, from the measurement points at each corner. This is typically linear along two edges of the triangle, then again between these two interpolated points. [20%]

(b) (i) The top left 3×3 block of numbers in the left hand matrix are the individual radial functions $\phi(|\mathbf{p} - \mathbf{p}_0|)$ evaluated at the measurement locations, where $\mathbf{p} = x$ in this case, since the data is one dimensional. The argument to each of these functions is the distance (in x) between the measurement node and each of the other measurement nodes in turn. This is a symmetrical matrix because the distance from x_0 to x_1 is the same as that from x_1 to x_0.

The last two columns of this matrix contain the polynomial part of the radial basis function, which in this case is of first degree, so the first of these columns is just a constant, and the second is equal to x. The last two rows of the matrix also represent the polynomial, in the same manner.

The next column vector contains the constants in the RBF, where the λ_i values weight the basis function for the i-th node, c_0 is the constant term in the polynomial, and c_1 is the multiplier of x in this polynomial.

The right hand column vector contains the values at the measurement locations $f(x_i)$. [10%]

(ii) The first rows of this matrix equation state that the value of the RBF $s(x_i)$, evaluated at each measurement node x_i, should be the same as the data measured at that node $f(x_i)$. This ensures that the RBF does actually interpolate the data.

The last two rows are known as the *side conditions*. These state that the sum of *any* polynomial of the same degree as that in the RBF, evaluated over all of the measurement nodes, must be equal to zero. This is the same as saying that $\sum_i 1 = 0$ and $\sum_i x_i = 0$. If the polynomial exists, these are necessary to ensure that there is a unique solution to the equation, i.e. that the matrix is invertible. [10%]

(iii) The full RBF is as follows:

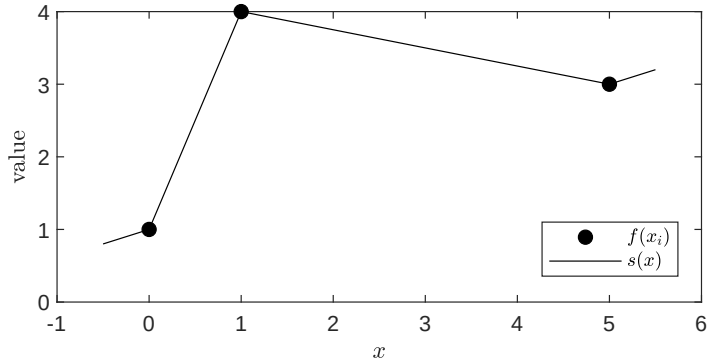
$$s(x) = c_0 + c_1 x + \sum_{i=1}^{3} \lambda_i \sqrt{(|x_i - x|^2 + \alpha^2)}$$

The numbers on the diagonal of the given matrix are $\phi(|x_i - x_i|) = \phi(0)$. But

$$\phi(0) = \sqrt{(0^2 + \alpha^2)} = \alpha$$

Hence $\alpha = 0.6$ for the given matrix equation. [25%]

(iv)



The result of setting $\alpha = 0$ is given in the graph above. In this case $\phi(r) = (r^2 + 0)^{\frac{1}{2}} = r$. Hence the basis is linear, the whole RBF will follow the same form as each of the basis functions, so the whole interpolant is also linear. (Note that it is not expected that the value of $s(x)$ outside the measurement node range will be plotted accurately.) [15%]

(v) Given the answer to (iv), the parameter α controls the smoothing of the RBF, with $\alpha = 0$ giving no smoothing at all. α with a very small value will only allow $s(x)$ to curve very close to each measurement node, whereas larger values of α will allow the curve to spread over a larger range.

To select an appropriate value for α, consider that if $|x_i - x| \ll \alpha$ the RBF is approximately a constant, whereas for $|x_i - x| \gg \alpha$ it is approximately linear. So we need α to be of the same order as the greatest distance apart of two nodes in the data to ensure it is all smoothed. Alternatively, we could set it to the greatest range over which we want the curve smoothness to be influenced by each measurement point. [20%]

Assessors' remarks: Most candidates could explain the different interpolation types in (a), though the application to unstructured data was a little more sketchy. The RBF explanations in (b)(i)–(iii) were mostly answered very well, with candidates generally losing

marks only when they neglected to specify the purpose of the equations in (ii). It was good to see that many candidates spotted in (iv) that this was a linear RBF and hence the interpolant would also be linear, and the subsequent discussions in (v) were mostly also good, though α really controls the 'smoothness' here rather than the 'tension'.

4. **Phong and Blinn shading**

(a)

$$I_s = \begin{cases} I_p k_s \cos^n \alpha = I_p k_s \left(\mathbf{R}.\mathbf{V}\right)^n & \text{for } \alpha, \theta < 90^\circ \\ 0 & \text{otherwise} \end{cases}$$

I_p is the intensity of the point light source. k_s is a specular reflection coefficient in the range 0 to 1. n is the specular reflection exponent. [10%]

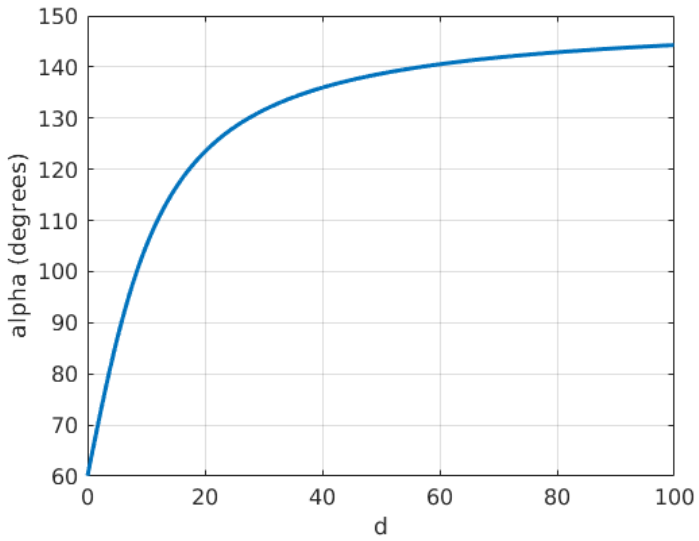
(b) The variable `n` is the specular exponent in (a). It controls the tightness of the highlight. For high values of n, I_s will be significant only for small values of α: the surface will behave like a mirror. For lower values of n, the highlight will be more spread out, giving the effect of a slightly shiny surface.

We would expect this line of GLSL code to appear in the fragment/pixel shader, since we are told that this is an implementation of Phong shading. In Phong shading, lighting calculations are performed at each pixel. In contrast, in Gouraud shading the lighting calculations would be performed in the vertex shader, with the results then interpolated to the pixels. [20%]

(c) The angle between **V** and the floor is $\tan^{-1}(h/d)$, and the angle between **R** and the floor is ϕ. It follows that

$$\alpha = \pi - \phi - \tan^{-1}(h/d)$$

The graph below is for $\phi = 30^\circ$ and $h = 10$.



[30%]

(d) At the locus of points in the rendering where $\alpha = 90°$, the specular term I_s is clipped with a discontinuity in its gradient. For moderate to large values of n this will not be noticeable, since the gradient of I_s will already be very small at this boundary. But for small values of n, we can expect to observe a sharp change in intensity along the boundary. [20%]

(e) **H** is known as the halfway vector because it lies half way between the illumination and viewing directions: $\mathbf{H} = (\mathbf{L} + \mathbf{V})/\left|\mathbf{L} + \mathbf{V}\right|$. One advantage of Blinn's method is that **H** is constant when the light source and the viewer are both at infinity. The specular computation then requires just a single dot product at each vertex, saving the expense of calculating **R** (which is not constant for curved surfaces) at each vertex. The second advantage is that the artefact identified in (d) is avoided, since the angle between **H** and **N** never exceeds $90°$ (unless the light source is behind the surface, in which case the surface is not directly illuminated anyhow). [20%]

Assessors' remarks: This question tested the candidates' understanding of the Phong model, introducing an unfamiliar weakness of the specular term when the light source is behind the viewer. In (a), while most candidates could formulate the specular term up to a point, very few remembered the $\alpha < 90°$ condition, despite this being spelled out in the subsequent line of GLSL. This rather scuppered most candidates' attempts at (d): even if they had derived the correct relationship between α and d in (c) — and, disappointingly, only around half did — the significance of α exceeding $90°$ was missed. Understanding of fragment and vertex shaders in (b) was similarly weak. More positively, many candidates gave a good account of Blinn's method in (e).

Andrew Gee & Graham Treece
May 2021