CITS4404 Assignment 2

Practical Project - Camouflage Worms

Background

It is sometimes said that *the eye is the window to the soul*. Whether or not that is true, is does provide a window to the central nervous system, and in turn provide a window to markers of not only diseases of the eye, but also to some systemic diseases.

Optical Coherence Tomography (OCT) is a non-invasive imaging technique that allows imaging beneath the surface of a semi-opaque object using the reflection of light from different densities within the body of the object. As it is not harmful, it has been particularly useful for imaging of the human retina. It allows us to obtain a picture of what is happening in the layers of nerves behind the eye.

Figure 1 illustrates the nerves found behind the eye, marked with nine identified layers, from the internal limiting membrane (ILM) to the retinal pigment epithelium (RPE). We are most familiar with the rods and cones in the photoreceptive layer (PL). (Did you know that the retina is actually wired "back to front"? There may be good reasons for this.)

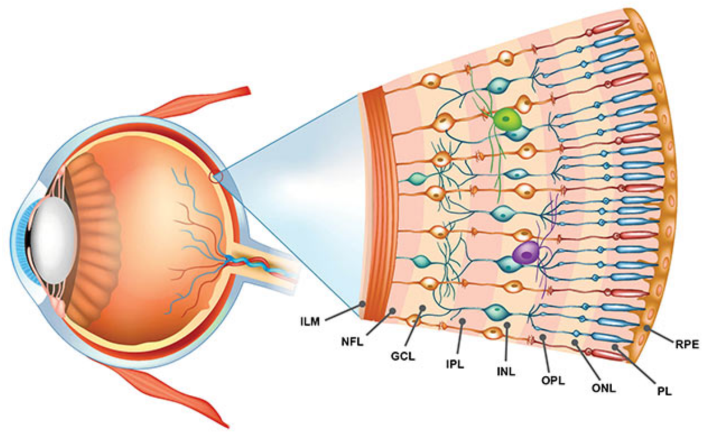

Figure 1. Structure of the eye.

Researchers and clinicians are interested in the structure, and changes over time, of the nerve layers to look for markers of diseases such as diabetes [1], Alzheimers [2] and hypertension (coronary disease) [3].

Figure 2 shows an OCT scan in the region of the fovea (the depression in the centre) with four of the layers marked [2].

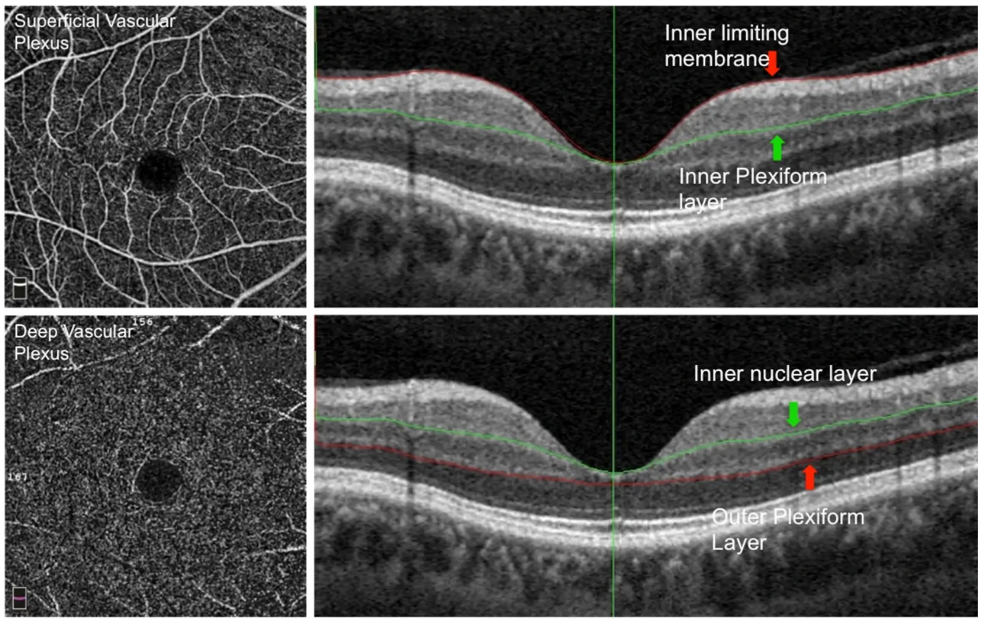

Figure 2. OCT scans from Cunha et al [2].

More severe pathology, and its affect on the layers, can be seen in Figure 3, taken from [4].

There is considerable potential for the beneficial use of AI both to identify disease, and to track its progression. To achieve this it is necessary to automatically identify regions and boundaries in the images.

A significant issue with OCT is that the images are normally quite noisy, particularly when taken in an efficient manner. This makes it harder for both clinicians and automated approaches to recognise pathology and track changes over time. For this reason it remains an open research topic to try to remove noise, and identify regions of the retina and layer boundaries.

In this project we will explore a novel adaptive approach to see if it shows promise for cleaning and sharpening images.

Aim

The aim of this project is to choose and test the use of a stochastic population-based optimisation algorithm, along with a particular representation using Bézier curves and a suitable cost function, as a way of giving greater definition to an OCT retinal slice.

“Greater definition” includes reducing the visual effects of noise, while at the same time making sharper and more distinct the boundaries between regions — that is, giving greater definition the layers.

More details on each of these three components is provided below.

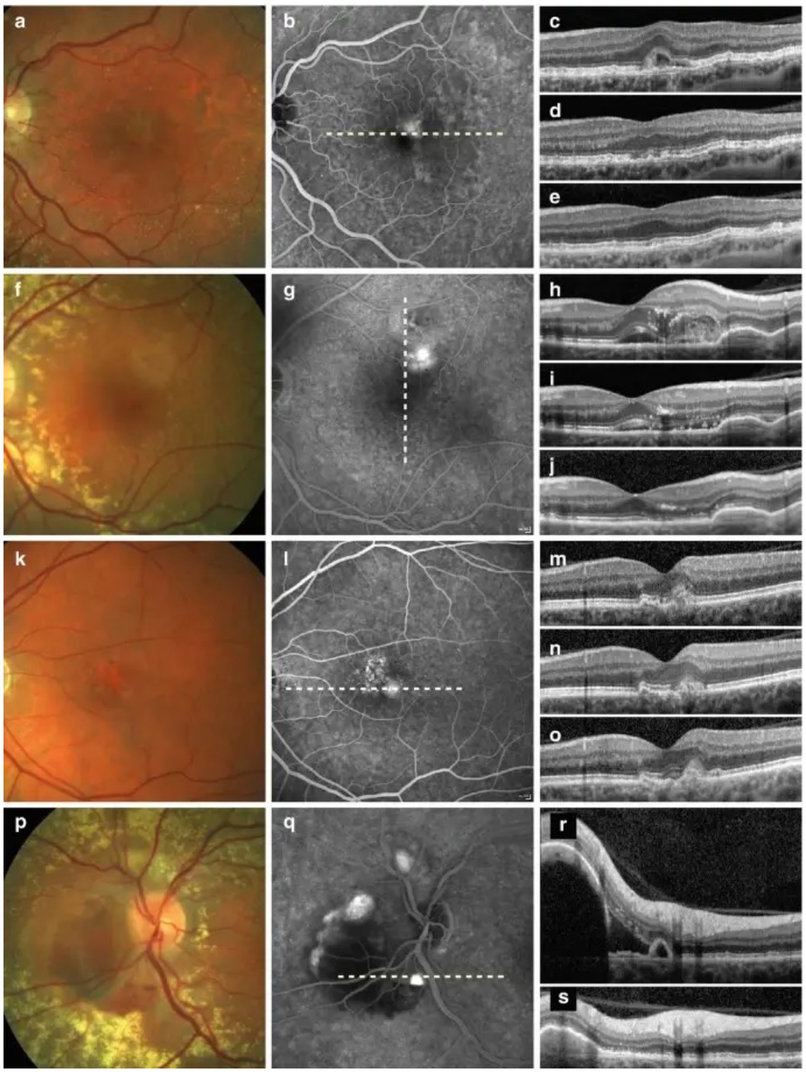

Figure 3. Diagnostic imaging of diseased retina, from Müller et al [4].

Choice of Algorithm

The algorithm chosen must be a (meta-heuristic) population-based algorithm. This includes evolutionary algorithms, genetic algorithms, swarm algorithms, ant-colony optimisation, and others.

It is suggested that project groups consider choosing, or building upon, one of the algorithms that group members researched in Assignment 1. While this is not a strict requirement, it provides an opportunity to leverage off that work, and ensures that a contemporary algorithm is considered.

It is not intended that too much of the project time be spent implementing the algorithm, so that time is available for testing and experimentation with the

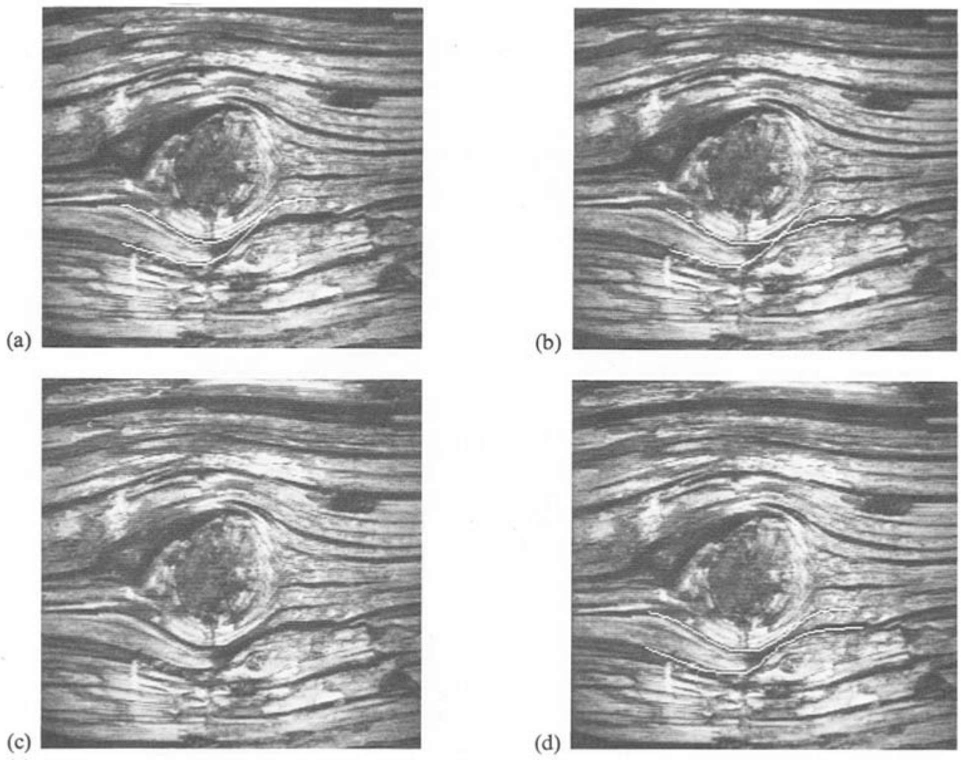

Figure 4. Kass et al's "snakes" finding contours in a cross section of wood [5].

cost function. A simplified version of the algorithm may therefore be used if necessary.

Some of the papers may also provide code that can be used and modified, with appropriate attribution. Using optimisation functions from a standard library however will not score highly.

The algorithms used in this project should be generic. That is, they could equally be applied to another problem domain. The representation and cost function, on the other hand, will be developed for this specific problem. This *separation of roles* is similar to the motivation behind modular and object-oriented programming.

Representation

Natural images rarely have sharp lines or straight edges - rather the intensity (or grayscale value in the case of a black and white image) tends to change more quickly at "edges" of features in the image than at other areas.

An image can be thought of as being like a terrain, with lighter areas being the peaks and darker areas being the valleys (in fact it is easy to render the image as a 3-d landscape). Like the map of a terrain, contour lines can be drawn along paths of constant intensity, or height.

The representation (or language) used in this project is inspired in part by the “snakes” model of Kass et al [5]. Snakes are an “active contour model” consisting of splines that can be bent to fit intensity contours in images. These contours are packed most closely along paths with rapid changes in intensity, or high intensity gradients, and tend to represent the edges of regions or objects in images. The snakes can there be used for edge detection. An example from [5] is shown in Figure 4.

The snake works by seeking to minimise an energy function (another name for a cost function) with a number of components. For example, if it is deemed that it takes energy to bend the snake, then the snake will “want” to be straighter. This helps the snake smooth out noise (rather than twisting around every noisy pixel). This tendency competes against other components of the energy function that seek to follow a contour. By weighting each energy term by a “tuning parameter”, the practitioner can change the behaviour of the snakes.

Camouflage Worms

For our project we will use as our model a set objects that we will call *camouflage worms*, or ***camo worms*** for short. We will call the set a *clew* - the collective noun for worms.

The general idea behind camo worms is that they are able to cover regions of the image (unlike the snakes) and adapt their colour to the predominant colour of the region - thus “smoothing out” the noise. At the same time, they are able to bend to align with contours, or edges.

In order to keep the number of parameters manageable the camo worms are realised as bendable oblongs. A randomly initialised clew of camo worms is shown in Figure 5. Notice that some worms are better camouflaged than others. Some real “camo worms” are also shown for comparison.

Technically the worm is defined by a quadratic Bézier curve, plus two additional parameters, a width and a colour. The Bézier curve itself requires three control points, and therefore requires six parameters (in 2-d space). Each worm is therefore defined by eight parameters.

While the simplest representation for the Bézier curve is three pairs of coordinates for the three control points, there is little intuitive relationship between the points and others qualities of the worm. We will therefore use a different set of six parameters that define the worm’s *location*, *size*, *orientation*, and *deviation from a straight line segment.*

The parameters are illustrated in the Figure 6.

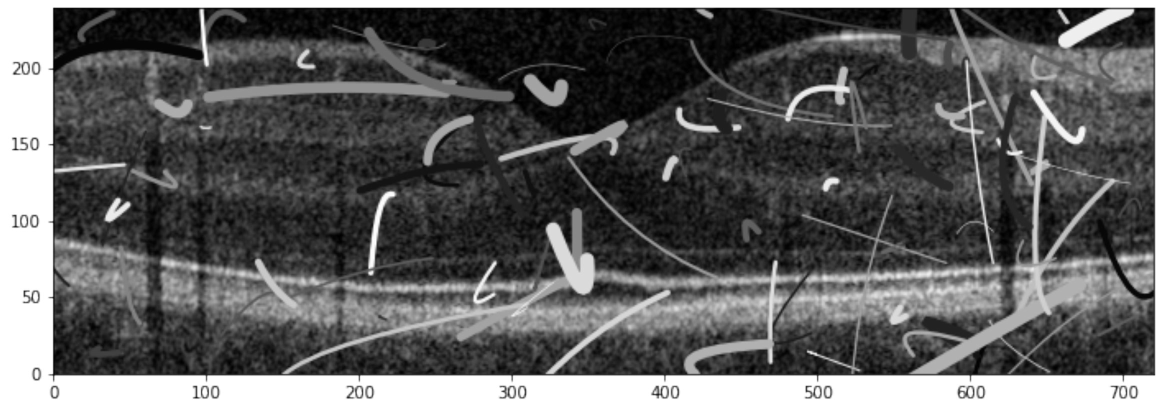

Figure 5. (a) A clew of 100 camo worms randomly initialised on an OCT image. (b) Comparison with some real life “camo worms” - the Australian *velvet worm* [6].

Our worms are defined with respect to a “centre point”, shown as the blue dot in Figure 6. The positions of the control points, shown in orange, are defined relative to the centre point. The three control points in turn determine the shape of the Bezier curve.

We will call the line segment connecting the two end control points the “centre line”. This is the shape the worm would have if it had no curvature. The curvature comes from “dragging” the mid control point away from the centre line.

The positions of the control points are defined by the following parameters:

- (x,y) - the co-ordinates of the centre point
- r - the “radius” of the end control points from the centre point (the centre line therefore has length $2r$)
- θ - the angle of the centre line from the x-axis

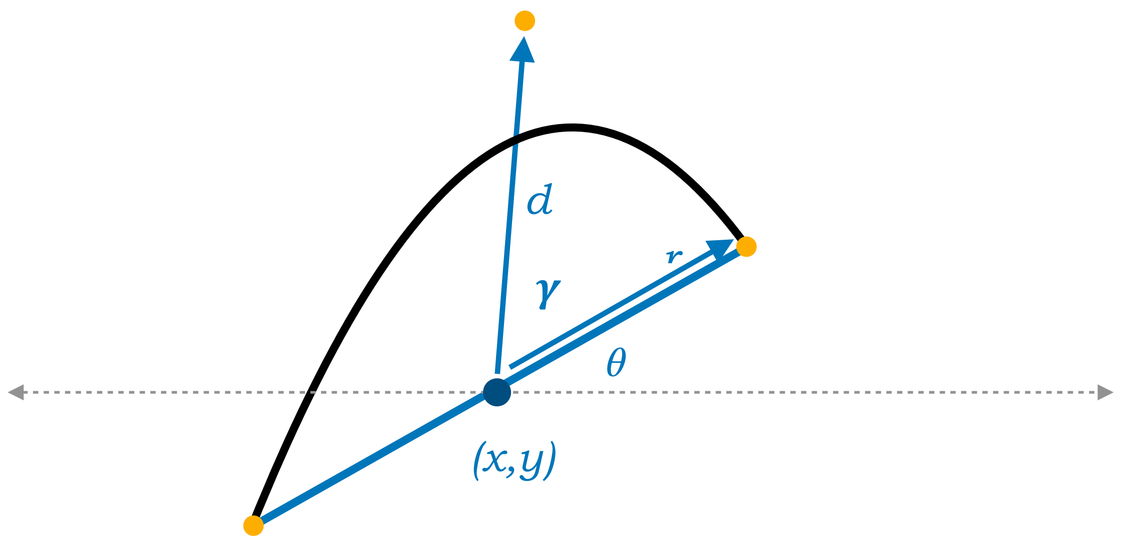

Figure 6. Paramaters defining a worm's Bézier curve.

- *d* - the radius of deviation of the mid control point from the centre
- ***γ*** - the angle of the line segment joining the centre and mid control points from the centre line

Notice that in this representation there are two parameters controlling the location, two parameters controlling the size and orientation (of the centre line), and two parameters controlling the curvature. Adding parameters for the width (thickness) and colour gives the eight parameters in total for each worm.

An actual rendering of a camo worm with similar parameter values to the diagram in Figure 6 is shown in Figure 7.

Cost Function

The adaptive "behaviour" of the worms is determined by the cost function that your algorithm will seek to minimise (or maximise). It is up to you to determine the cost function, with the provision of course that it should not use information that is instance-specific other than that obtained by sampling. That is, it should work equally well on a different test image.

More specifically there are three types of information that you may wish to make use of in the cost function:

1. *Internal knowledge* - the worm's knowledge of itself, such as its size, curvature and colour.
2. *Group knowledge* - such as the worm's position relative to other members of the group.

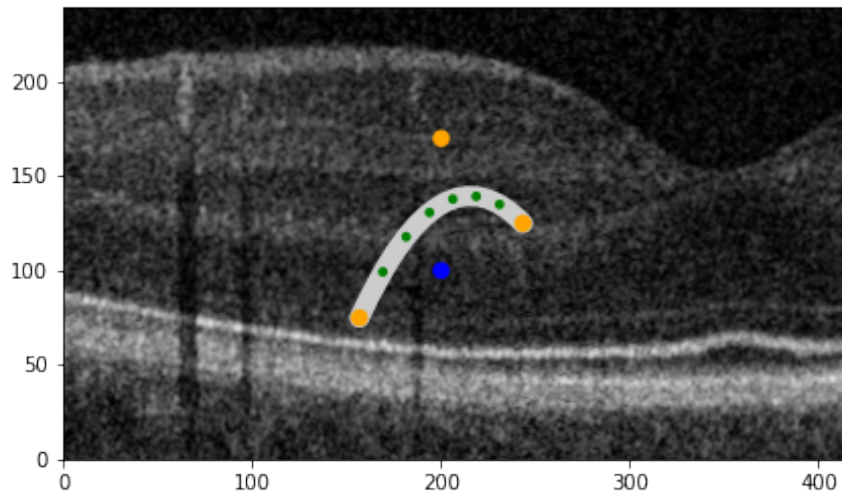

Figure 7. Plot of a rendered camo worm similar to the diagram in Figure 6, with centre point and control points shown, as well as some intermediate sample points.

3. *Environment knowledge* - the environment may be sampled in the vicinity of the worm to determine the image intensity.

Assuming a clew of n worms, the cost function can therefore be characterised as

$$C(\overrightarrow{v}) = \sum_{j=0}^{n} \left(w^I C^I(\overrightarrow{v}_j) + w^G C^G(\overrightarrow{v}_j) + w^E C^E(\overrightarrow{v}_j) \right)$$

where w^I, w^G, and w^E are weights (or tuning parameters) for the costs related to internal knowledge, group knowledge and environmental knowledge respectively. The corresponding cost functions may of course each consist of more than one component.

Here each $\overrightarrow{v}_j$ is an 8-dimensional vector describing one of the n worms (we have left the image itself implicit). $\overrightarrow{v}$ can be considered a concatenation of the $\overrightarrow{v}_j$, that is, an $8n$ dimensional vector, although we won't represent it that way in the (object-oriented) code.

It is your job to decide what goes into each component of the cost function, and how much weight to give them. In doing so it may be helpful to think about the following influences on the adaptive behaviour of the camo worms.

Drivers Created by the Cost Function

The cost function can be thought of as creating "drivers" for the behaviour of the set of worms from iteration to iteration.

- In large areas where there is little change in intensity (other than noise) you may wish to “use up” as few worms as possible to cover these areas. Therefore you may wish the worms to grow large (long and wide).
- In areas where there are long ridges, you may wish the worms to grow long to smooth over the noise, but remain narrow so as not to lose detail.
- A worm can only bend in one direction. Therefore in areas with a lot of detail you may need a large number of small worms to preserve the detail.
- Significant amounts of overlapping will “waste” worms that could be better used in sharpening up the detail.

It is important to note that a worm does not know if it is in, for example, a large area with little change in intensity. It has to evolve to “figure this out”.

With these in mind, the following are some example suggestions you might try encoding in the cost function.

Internal Drivers

1. The worm prefers to grow larger.
2. The worm prefers to be straighter.

Relationships within the Group

3. The worm prefers to be distant from other worms (it likes its own space).

Relationships with the Environment

4. A worm prefers to be closer in colour to the image that it covers (it likes to be camouflaged!).

Notice that these drivers, or penalties, may work together and may work in competition. For example, 1 and 3 may tend to make the worms spread out and fill the space. On the other hand, 4 will tend to have a constraining influence on worm size.

It is up to you to define what these drivers mean in more detail. For example, in the case of 4, how much penalty is attributed for covering a speckled area versus crossing over a “ridge” of intensity.

Evaluation

You will need to give some thought to how you evaluate the overall success or otherwise of your population-based algorithm (and cost function). Initially the changes you make will likely have a large effect on the outcome, and it may be sufficient to rely on *qualitative* evaluation (drawing the outcome and relying on human judgements).

As your approach becomes more refined and changes become smaller you may find it useful to devise (one or more) *quantitative* measures of performance.

Provided Code

In order to allow you to focus on the algorithm and cost function, the representation has been encoded for you. The code also includes some utility and visualisation functions, and can be found in CamoWorms on Cocalc.

The code consists of three main parts - some code to initially read in and crop the image, a CamoWorm class, and a Drawing class. The remaining functions provide some demonstrations of their use (generating the above figures), and are not particularly intended for further use. (It is recommended you continue with an object-oriented approach.)

While most of the code is self explanatory, it is worth pointing out that the Camo_Worm class is built using matplotlib.bezier as an internal representation. Therefore, while some utility methods are provided for convenience and demonstration purposes, you also have access to the bezier attribute, and hence to the additional methods of matplotlib.bezier.

Deliverables

The project has three deliverables: a seminar, a report, and a code repository.

Seminar

Each team will be required to give a 15 minute seminar on their project. Seminars will be scheduled in the last two weeks of semester. At this stage it is envisaged that seminars will be presented face to face, although this may change and some or all seminars may be delivered via Teams depending on the situation (primarily Covid numbers at that time).

The seminar should include the following:

- A high level explanation of the optimisation algorithm chosen, and any distinguishing features. The explanation should make use of diagrams and/or pseudocode rather than actual code. Issues in the implementation may be discussed, but a full methodology is not required.

- A discussion of the considerations that went into choosing cost functions, and a description of the cost function(s) chosen.

- Presentation of results. Graphical and/or table representations are preferred. You may also wish to show a partial run of the algorithm if it is able to demonstrate progress in the short time available (for example, using a reduced size clew for demonstration purposes). Alternatively a short pre-recorded "movie" of the algorithm running (which allows it to be sped up) may be useful.

- You may wish to interleave the above two points (cost function and results) to demonstrate how changes in the cost function affected the results.
- Conclusions from your work.

Report

Your report should be in the style of a conference paper. The conference proceedings from Assignment 1 can be considered as a guide.

The abstract should provide a high level overview of your paper - the relevant background, the choices you made, and the results.

The body of the paper should provide more detail. Note that in the background or introduction you do not need to repeat information from this project specification (you can refer to this paper) - your focus should be on the algorithm that you have chosen to explore. You should report on the algorithm implementation (again diagrams and/or pseudocode are preferred to code) and the decisions you made, the cost functions you explored, the results you obtained and your conclusions.

Code

The code, and any results files, should be provided either in a CoCalc project directory (of a specified team member), or in a repository such as GitHub. The code should allow reproduction of the results. Instructions for running the code can be included in a notebook or a readme file.

The report should provide a pointer to where the code, and instructions, can be found.

Practicalities

- This is a group project and will be assigned to teams of approximately 4 members.
- The report (and code link) is due by **11:59pm, Wednesday 25th May**. Late assignments will penalised according to the University's standard rules for late submissions described in the Unit Outline.
- The report is limited to a **maximum of 3000 words** excluding diagrams and references. Text beyond the maximum word count will not be marked. A word count must be included on the title page, along with the **names and student numbers of the team members**.
- The assignment must be submitted as a **pdf** file.
- Referencing should follow the IEEE style (as covered in the first lecture).
- Submission will be through the LMS.

- This assignment is **group work**. Submitting work is considered a declaration that the work submitted is entirely the work of the team members.
- A Teams channel has been provided for queries relating to the assignment.

References

[1] Orduna-Hospital, E., Sanchez-Cano, A., Perdices, L. et al. “Changes in retinal layers in type 1 diabetes mellitus without retinopathy measured by spectral domain and swept source OCTs,” in *Sci Rep,* 11, 10427, 2021. https://doi.org/10.1038/s41598-021-89992-w

[2] Cunha, L.P., Almeida, A.L.M., Costa-Cunha, L.V.F. et al., “The role of optical coherence tomography in Alzheimer’s disease,” in *Int J Retin Vitr,* 2, 24, 2016. https://doi.org/10.1186/s40942-016-0049-4

[3] Sun, C., Ladores, C., Hong, J. et al. “Systemic hypertension associated retinal microvascular changes can be detected with optical coherence tomography angiography,” in *Sci Rep,* 10, 9580, 2020. https://doi.org/10.1038/s41598-020-66736-w

[4] Müller, P.L., Wolf, S., Dolz-Marco, R., Tafreshi, A., Schmitz-Valckenberg, S., Holz, F.G. (2019), in *Ophthalmic Diagnostic Imaging: Retina.* Bille, J. (eds) *High Resolution Imaging in Microscopy and Ophthalmology.* Springer, Cham. https://doi.org/10.1007/978-3-030-16638-0_4

[5] Kass, M., Witkin, A., Terzopoulos, D., “Snakes: Active contour models,” in *International journal of computer vision,* 1, 4, p.321-331, 1988.

[6] Blaxter, M., Sunnucks, P., “Velvet worms,”, in Current Biology, 21, 7. https://www.cell.com/action/showPdf?pii=S0960-9822%2811%2900208-9