Honours Logbook

Semester 1 2024 - Semester 1 2025

N.B.: This logbook contains notes and updates on my progress and findings throughout my Honours year.

Maintainer:
        Jeffrey Liang   ($u7013004$),   ANU School of Computing

Contributors:
        Jeffrey Liang   ($u7013004$),   ANU School of Computing

Contents

1 Overview

Welcome to my progress logbook!

Hello. Topic. Challenges. Research interest.

2 Progress Logbook

2.1 Progress

Sun 12/05/2024:

Reading MeshSDF paper.

Sat 18/05/2024:

Reading MeshSDF, the Learning Implicit Fields papers, and Structure from Motion. Need to look up what the following mean:

- differentiable rendering, surface reconstruction

Sun 19/05/2024:

Reading more from Structure-from-Motion Revisited. Didn't really understand much of SfM last night.

Thu 27/06/2024:

Attempting 3D version of the ellipse problem.
Question: Why do we invert the u_is? Answer: you bring the 1 over and multiply by some expression on both sides.

Fri 5/07/2024:

Having been making headway on the non-axis-aligned ellipsoid. I've been able to fit an ellipse to one but I haven't been able to do the bi-level optimisation. I'm going to try a couple approaches:

- reparametrise everything to radians and a, b, c rather than squared inverses of semi-axes

- try original objective function rather than focusing on optimising rotation

- last resort is to figure out what the SqrtBackward error is

Sun 7/07/2024:

Was able to switch implementation to radians and a, b, c rather than squared inverses $1/a^2, ..., 1/c^2$. It's still temperamental though and will hopefully be finetuned when meeting with Chamin who hopefully has a few tricks. Going to read some papers today about DRWR and the DDN and take notes to try and fully understand derivations etc.

Sat 20/07/2024:

Reading additional papers and waiting for Steve's meeting. Reading "Small Steps and Level Sets" by Chamin and will document.

Mon 29/07/2024:

Trying to do experiments to ensure I know the cause of my errors and that it's not some implementation error. Will do:

- small volume initialisations

- close to ground truth initialisations

- tightening the constraint slowly

- PCA?

Ideal semi-axes lengths is: 0.28209 Trials:

1. initialisation: 0.2, 0.4, 0.3, 30°, 25°, -120°. final: 0.271, 0.296,0.28, 29.98°, 24.99°, 60.0°

2. initialisation: 0.1, 0.2, 0.3, 30°, 25°, -120°. final: 0.479, 0.29,0.0913, 83.83°, 128.2°, 47.02° (no improvement).
   the issue here is that the fitted ellipse is too far away from a sphere for it to approach a sphere?

3. initialisation: close to ideal. result: optimised fine and as expected.

Perhaps farthest point sampling would prevent the ellipse from over concentrating at the poles.

Issue! For Trial 2, a close reinitialisation of 0.1, 0.3, 0.5 and angles 30°, 25°, -120° worked well. In fact, an exact reinitialisation worked?! But then when reverting back so that the fitted ellipse had those params, the gradients calculated were zero.

Steps to reproduce error:

1. Initialise with 0.1, 0.2, 0.3 and 30°, 25°, -120°. Run the first cell and then second cell, and we should see no deformation of the fitted ellipsoid to a sphere - the parameters remain as 0.479, 0.29,0.0913, 83.83°, 128.2°, 47.02°. The gradients are very close to 0.

2. Initialise with the parameters 0.479, 0.29,0.0913 and 83.83°, 128.2°, 47.02°. We see that the gradients are non-zero and we optimise to a sphere.

a, b, c = 0.4786546028, 0.2901328092,0.09132074619
yaw, pitch, roll = 83.82856741, 128.3492915, 47.0228292

Thu 15/08/2024:

Steve told me that I needed to update the DDN package and it worked!
I spent the past week a bit confused on how to do projection but Steve gave me some clear instructions.
I've switched the volume and surface area constraints, but the optimisation is being worse.

Sun 18/08/2024:

Going to work on optimising the volume to be a certain target now and read a paper. Then, I'll write some things in the template. Efficiency!

Wed 4/09/2024:

For the past few days, I've been trying to mathematically formulate the projection. Just tested it in code today and it seems that I am minimising the area of the cross-section with the x-y plane instead. So, just need to take Schur complement instead of projection with matrix P.

One strange observation is that the gradients seem to increase in magnitude as time goes on; not decreasing.

Sun 15/09/2024:

Finally writing about the point-sampling approach. Have been busy with side project website.

Fri 20/09/2024:

Trying to add a regulariser between iterations for lower problem.

Sun 22/09/2024:

Going to fully do the regulariser and report results. Will also provide warm-starting with previous solution.

Mon 23/09/2024:

Tried switching to float64 and it's not working. Going to try add more views. For

Sun 29/09/2024:

Worked on Chamfer distance and it worked! Forgot to take inverse of square root matrix but figured it all out after reading my projection notes.

Going to write my ellipsoid stuff in this logbook until I have to copy it over into my thesis.

> **2024-09-29**, : Figure out Blender stuff, proposal, and whether to use other nearest-neighbour method.

Sun 6/10/2024:

Wrote up some stuff for ellipsoid. Still unsure how to tackle boundary?

Mon 21/10/2024:

Basically spent 2 weeks without doing work. Time to do more! Going to attempt the alpha shapes approach. Hopefully converting back and forth from tensor to numpy won't do weird things.

Thu 23/01/2025:

Not sure how the mesh will be able to handle concavities if only the boundary points are detected in the upper-level loss, these will only be able to handle convex parts. I guess that the lower-level loss will handle that by moving in those concave sections in more.

Wed 26/02/2025:

Have explored many many options for the physics simulation. I will list them out. The aim is to get good data by Friday. I will aim by tonight to have tried various options and have thought about which one is best. The options are:

1. NVIDIA Omniverse - pros: minimal effort to set up inflatable, can set up scene easily, cons: unsure how mesh would be extracted, I will need access to Pranav's computer

2. Feather - pros: physics looks robust, easily usable interface, should be easily to extract mesh and it's in C++, can simulate interactions, cons: will need to manually set up environment, port to Blender and render there, requires Linux

3. Q-Minh: pros: good interface, physics looks good, should be easy to extract mesh, in C++, cons: need to port to Blender

4. ~~liujustin53: unknown, will have to try~~ BAD!

5. PBS-soft-body (ETH): can interact with rigid body if the rigid body is in the scene and not moving

6. PBDTaichi: Python, pretty basic, can't handle collisions with other objects

FML. Feather can only work on OpenGL 4.3 so I'll try on Windows at home.

Can also ask Jiahao about the apt stuff for libgl and other RandR (RandR library and headers were not found).

Can use the PBAToolkit in Python, so that should be doable on Mac. Same with some of the other ones.

The Feather one, I have almost no hope for. I can try to run it on the GPU but the OpenGL is so far behind.

Thu 27/02/2025:

This really sucks right now bruh. I am bumping into dead ends and I wonder if the only thing I can use is Pranav's computer atm. Going to try vcpkg again for Feather and Q-Minh (he replied and said he hasn't ported it over to Python yet but can do it next week). In the meantime, I can try liujustin53, PBS-soft-body (ETH). Verdict: liujustin53 can't do collisions.

Fri 28/02/2025:

Ok, need to ask if I can somehow use a computer with a GPU. If not, rent one? NVIDIA Omniverse has

everything, I just need to be able to use it.

My options otherwise are PBDTaichi, PBS-soft-body, and Feather. All can do volume preservation.

Feather is super slow but can do bending and length constraints, and can handle collisions with other objects. Investigate solver iteration.

PBDTaichi has bending and length constraints, but cannot handle collisions with other objects.

PBS-soft-body has length and friction constraints, can handle collisions with static objects in the scene, cannot handle collisions with other objects.

For now, I need some really basic data to get the pipeline working.

Discussion with Steve: he'll ask IT, I will generate a simple example in Blender first. Also code up the volume.

Mon 3/3/2025:

Yesterday, I worked on the three Blender shapes. I have three equations which produce surfaces of volume 1.4 pi. The next step is to visualise these in Blender.

I also tested the correctness of my signed volume function and it works correctly. As the mesh resolution increases, the volume approximates the true volume better and better.

Going to write about my background.

Tue 4/3/2025:

Made the mesh fairly high res, most of the volume is lost lengthways, so focus on upping resolution for that. Was able to render them.

Wed 5/03/2025:

Get the camera matrix. Get the Canny edge detector. After that, tidy up point cloud section and talk about how to perform surface reconstruction.

Sat 8/3/2025:

Important to have no nothing days. Today, I will write more about SfM in my thesis. The goal of the long weekend will be to have results of the mesh changing.

Wed 12/03/2025:

Turns out the SciPy was hanging because it takes 4 minutes per iteration when using a subdivision level of 4. Takes 10 seconds if subdivision level of 3, and basically instant with subdivision level of 2. I think including the manually providing the gradients should speed up the computation.

Today, I want to write the code for the node and make sure that it works for subdiv levels 2 and 3. After that, I want to write the Chamfer distance loss function and hook it up. Then I'd like to visualise the results. Should be good!

3   Ellipsoid Formulation

An ellipsoid is a 3-dimensional surface obtained by deforming a sphere under an affine transformation. As ellipsoid surfaces can be represented as the zero set of a quadratic polynomial in three variables, it is considered a quadric surface.

There are multiple mathematical representations of ellipsoids. Ellipsoids have three pairwise perpendicular axes of symmetry known as principal axes which intersect at the centre of the ellipsoid. The most general ellipsoid is a triaxial ellipsoid, where each of these three principal axes are of different lengths. The line segments along these axes from the origin to the surface boundary are named the principal semi-axes as they are half the length of the principal axes. An ellipsoid centred at the origin and with its axes aligned to the standard basis of $\mathbb{R}^3$ is commonly represented in Cartesian coordinates as the points $(x, y, z) \in \mathbb{R}^3$ which satisfy

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1, \tag{3.1}$$

where a, b, and c are the lengths of the semi-axes. If we let $\mathbf{x} = [x, y, z]^\top \in \mathbb{R}^3$, then we can represent this expression becomes

$$\mathbf{x}^\top \mathbf{A} \mathbf{x} = 1, \tag{3.2}$$

where

$$\mathbf{A} = \begin{bmatrix} 1/a^2 & 0 & 0 \\ 0 & 1/b^2 & 0 \\ 0 & 0 & 1/c^2 \end{bmatrix}.$$

However, ellipsoids need not be axis-aligned with respect to the standard basis. Rather than having $\mathbf{A}$ be a diagonal matrix with the inverse squared semi-axes lengths, we can generalise ellipsoids to be non-axis-aligned by letting $\mathbf{A}$ be any symmetric positive-definite matrix – that is, having $\mathbf{A} \in \mathbb{S}^3_{++}$ such that $\mathbf{A} = \mathbf{A}^\top \succ 0$. This represents an ellipsoid $\mathcal{E}$ as the convex set

$$\mathcal{E} = \left\{ \mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}^\top \mathbf{A} \mathbf{x} = 1, \mathbf{A} \in \mathbb{S}^3_{++} \right\}. \tag{3.3}$$

We can further perform an eigendecomposition on $\mathbf{A}$ to express it in terms of an orthonormal rotation matrix $\mathbf{R} \in SO(3)$ and a diagonal matrix $\mathbf{\Lambda}$ containing the eigenvalues of $\mathbf{A}$, which are the inverse squares of the semi-axes lengths:

$$\mathbf{A} = \mathbf{Q}^\top \mathbf{\Lambda} \mathbf{Q}, \tag{3.4}$$

where $\mathbf{\Lambda} = \begin{bmatrix} 1/a^2 & 0 & 0 \\ 0 & 1/b^2 & 0 \\ 0 & 0 & 1/c^2 \end{bmatrix}$ and thus

$$\mathcal{E} = \left\{ \mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x}^\top \mathbf{Q}^\top \mathbf{\Lambda} \mathbf{Q} \mathbf{x} = 1, \mathbf{Q} \in SO(3) \right\}. \tag{3.5}$$

4   Deep Declarative Nodes

Deep neural networks are themselves optimisation problems, where a loss function is minimised over the network weights. Backpropagation enables this optimisation to occur efficiently. However, incorporating constraints into a neural network complicates this procedure as constraints need to be respected in the optimisation process. Furthermore, constrained optimisation is often iterative and not naturally compatible with the differentiable nature of the current neural networks.

Deep declarative networks (DDNs) provide a new framework to achieve backpropagation through certain constrained optimisation problems. The conditions for these problems are that we need the optimal solution to the optimisation problem and twice-differentiable objective and constriant functions, and thus the method used to find the optimal solution can be non-differentiable, such as RANSAC.

5   Project Book Components

We can use sections organize the text into different parts such that specific topics are addressed in their own section.

> **TODOs**
>
> You will often have pending tasks that you need to track. This project logbook allows you to include both high and low priority todos that will be summarised in a list at the end of the file. Use the commands \hightodo{<date_added>}{<author_key>} and \lowtodo, including a date is recommended for tracking purposes.
>
> Note: define your userId in the preamble

2016-05-23, : Try the code in the newest version of numpy compiled with optimized BLAS.

2016-05-27, **u7013004**: Perform convergence analysis with the latest version of the code.

5.1   Algorithms

> Sometimes the most straightforward way to explain a procedure is just to give it in a algorithmic format, it takes a little time but it will force you to go through the steps and you will most likely be able to reuse it on you paper. For the full documentation see here https://texdoc.org/serve/algorithmicx/0.

Algorithm 1 Euclid's algorithm

1: **procedure** EUCLID(a, b)   ▷ The g.c.d. of a and b
2:      $r \leftarrow a \bmod b$
3:      **while** $r \neq 0$ **do**   ▷ We have the answer if r is 0
4:          $a \leftarrow b$
5:          $b \leftarrow r$
6:          $r \leftarrow a \bmod b$
7:      **return** b ▷ The gcd is b

5.2   Code

> If the algorithm is too vague and you feel like you need the source code you can also insert it. You can put LaTeX code inside by using <@ @> delimiters, highlight small pieces of the code with <| |> delimiters, and highlight full lines (see the source code). For now only Python colored syntax highlighting is available, but most languages are supported and the color scheme can be added. More information available here https://www.overleaf.com/learn/latex/Code_listing.
>
> Note that these pieces of code in the LaTeX document must start with no indent.

```python
def DTW_distance(s1, s2):
    """
    Function to compute the Dynamic Time Warping in Python between two
    signals
    """
    DTW={}

    for i in range(len(s1)):
        DTW[(i, -1)] = float('inf') # By default ∞
    for i in range(len(s2)):
```

```
10          DTW[(-1, i)] = float('inf') # By default ∞
11      DTW[(-1, -1)] = 0
12
13      for i in range(len(s1)):
14          for j in range(len(s2)):
15              dist= (s1[i]-s2[j])**2
16              DTW[(i,j)] = dist + min(DTW[(i-1, j)],DTW[(i, j-1)], \
17                  DTW[(i-1, j-1)])
18
19      return sqrt(DTW[len(s1)-1, len(s2)-1])
```

5.3   Diagrams

For simple diagrams I highly recommend learning TiKZ, you will be drawing the diagrams in pure LaTeXwhich has a steep learning curve but once you get used to it, it can be quite easy to display and do `for` loops to draw multiples line at once.
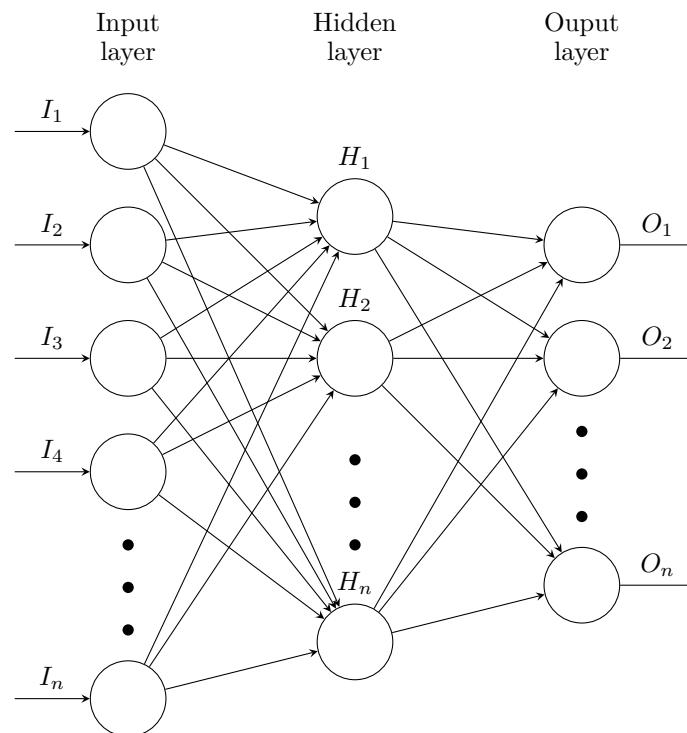
However, sometimes you will need more complicated diagrams (or maybe you do not like TiKZ, in that case I recommend a vector drawing tool such as Inkscape which allows LaTeX embedding)

5.4   Figures

In general the best way to visualize your results will be some figures, I recommend Python's matplotlib for generating them or any other tool you are familiar with.

5.5　Tables

> LaTeX booktab environments are really good to showcase and track your results, however they can get fairly messy. My suggestion is to generate them via Python automatically and store the results in either a plain text file or a spreadsheet (there are packages to read spreadsheets with Python)

$\sigma \setminus \tau$	0	1	2	3	4	5	6	7	8
0.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
0.2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
0.4	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
0.6	98.6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
0.8	84.7	99.5	100.0	100.0	100.0	100.0	100.0	100.0	100.0
1.0	28.1	98.3	99.9	100.0	100.0	100.0	100.0	100.0	100.0
1.2	1.3	88.7	99.4	99.9	99.8	99.9	100.0	99.9	100.0
1.4	0.0	57.1	96.2	99.3	99.0	99.3	99.4	99.8	99.7
1.6	0.0	18.6	81.2	93.0	93.7	94.8	95.6	92.3	93.3
1.8	0.0	2.4	42.8	67.0	70.1	72.1	69.0	69.1	68.6
2.0	0.0	0.1	9.0	23.1	24.5	26.9	28.2	27.3	27.3
t(ms)	27.92	40.23	77.30	157.27	252.05	342.18	381.46	399.85	413.72

Table 5.1: Performance of the algorithm for 128-bit key and with multiple readings per key

6   Topic C to focus on

A   References

> Do not forget to cite the papers that you are using in your research, this way your work will be infinitely easier to write down and to review when the time comes.

B   Resources

> It is a good idea to record sources that explain concepts or provide tools so the research is both better documented and if someone has to continue it there is enough supporting documentation.

- Quick read in DTW and Keogh Lower Bounding.
  http://alexminnaar.com/time-series-classification-and-clustering-with-python.html
  http://nbviewer.jupyter.org/github/alexminnaar/time-series-classification-and-clustering/
  blob/master/Time%20Series%20Classification%20and%20Clustering.ipynb

- Parallelizing DTW – Good article on making a parallel version of DTW. Uses Keogh lower bound not as a linear approximation but as a pruning device.
  https://www.andrew.cmu.edu/user/mmohta/15418Project/finalreport.html

- Deep Learning

  - Intro to LSTM
    https://colah.github.io/posts/2015-08-Understanding-LSTMs

  - Intro to CNN
    https://colah.github.io/posts/2014-07-Conv-Nets-Modular/

  - Why are LSTMs are so useful, impressive result in character pattern and syntax learning
    https://karpathy.github.io/2015/05/21/rnn-effectiveness/

C TODOS

> Here you will have all your TODOs grouped with anchor links to the parts of the document where they are. Really handy if you do not know where to continue with your project.

Todo list