University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Level 4 Project Report Template

John H. Williamson

September 14, 2018

Abstract

Every abstract follows a similar pattern. Motivate; set aims; describe work; explain results.

“XYZ is bad. This project investigated ABC to determine if it was better. ABC used XXX and YYY to implement ZZZ. This is particularly interesting as XXX and YYY have never been used together. It was found that ABC was 20% better than XYZ, though it caused rabies in half of subjects.”

Acknowledgements

We give thanks to the Gods of LaTeX, who in their eternal graciousness, have granted that this document may compile without errors or overfull boxes.

Education Use Consent

Contents

1 Introduction

TODO: Remove the guidance notes from your dissertation before submitting!

Why should the reader care about what are you doing and what are you actually doing?

1.1 Guidance

Motivate first, then state the general problem clearly.

1.2 Writing guidance

1.2.1 Who is the reader?

This is the key question for any writing. Your reader:

- is a trained computer scientist: *don't explain basics*.
- has limited time: *keep on topic*.
- has no idea why anyone would want to do this: *motivate clearly*
- might not know *anything* about your project in particular: *explain your project*.
- but might know precise details and check them: *be precise and strive for accuracy.*
- doesn't know or care about you: *personal discussions are irrelevant*.

Remember, you will be marked by your supervisor and one or more members of staff. You might also have your project read by a prize-awarding committee or possibly a future employer. Bear that in mind.

1.2.2 References and style guides

There are many style guides on good English writing. You don't need to read these, but they will improve how you write.

- *How to write a great research paper* Peyton Jones (2017) (**recommended**, even though you aren't writing a research paper)
- *How to Write with Style* Vonnegut (1980). Short and easy to read. Available online.
- *Style: The Basics of Clarity and Grace* Williams and Bizup (2009) A very popular modern English style guide.

- *Politics and the English Language* Orwell (1968) A famous essay on effective, clear writing in English.
- *The Elements of Style* Strunk and Whyte (2007) Outdated, and American, but a classic.
- *The Sense of Style* Pinker (2015) Excellent, though quite in-depth.

Citation styles

- If you are referring to a reference as a noun, then cite it as: "Orwell (1968) discusses the role of language in political thought."
- If you are referring implicitly to references, use: "There are many good books on writing (Orwell 1968; Williams and Bizup 2009; Pinker 2015)."

There is a complete guide on good citation practice by Peter Coxhead available here: `http://www.cs.bham.ac.uk/~pxc/refs/index.html`. If you are unsure about how to cite online sources, please see UNSW (2019). [1]

1.2.3 Plagiarism warning

WARNING

If you include material from other sources without full and correct attribution, you are commiting plagiarism. The penalties for plagiarism are severe. Quote any included text and cite it correctly. Cite all images, figures, etc. clearly in the caption of the figure.

1.2.4 Quoting text

If you are quoting a long passage, use a `quote` environment:

> If you scribble your thoughts any which way, your readers will surely feel that you care nothing about them. They will mark you down as an egomaniac or a chowderhead -or, worse, they will stop reading you. The most damning revelation you can make about yourself is that you do not know what is interesting and what is not.

(Vonnegut 1980)

If you are quoting inline, like Simon Peyton-Jones' following remark, use quotation marks "Conveying the intuition is primary, not secondary" (Peyton Jones 2017).

[1] Specifying an online resource like `https://developer.android.com/studio` in a footnote sometimes makes more sense than including it as a formal reference.

2 Background

What did other people do, and how is it relevant to what you want to do?

2.1 Guidance

- Don't give a laundry list of references.
- Tie everything you say to your problem.
- Present an argument.
- Think critically; weigh up the contribution of the background and put it in context.
- **Don't write a tutorial**; provide background and cite references for further information.

3 Analysis/Requirements

What is the problem that you want to solve, and how did you arrive at it?

3.1 Guidance

Make it clear how you derived the constrained form of your problem via a clear and logical process.

The analysis chapter explains the process by which you arrive at a concrete design. In software engineering projects, this will include a statement of the requirement capture process and the derived requirements.

In research projects, it will involve developing a design drawing on the work established in the background, and stating how the space of possible projects was sensibly narrowed down to what you have done.

4 Design

How is this problem to be approached, without reference to specific implementation details?

4.1 Guidance

Design should cover the abstract design in such a way that someone else might be able to do what you did, but with a different language or library or tool. This might include overall system architecture diagrams, user interface designs (wireframes/personas/etc.), protocol specifications, algorithms, data set design choices, among others. Specific languages, technical choices, libraries and such like should not usually appear in the design. These are implementation details.

5 Implementation

What did you do to implement this idea, and what technical achievements did you make?

5.1 Guidance

You can't talk about everything. Cover the high level first, then cover important, relevant or impressive details.

5.2 General guidance for technical writing

These points apply to the whole dissertation, not just this chapter.

5.2.1 Figures

Always refer to figures included, like Figure 5.1, in the body of the text. Include full, explanatory captions and make sure the figures look good on the page. You may include multiple figures in one float, as in Figure 5.2, using `subcaption`, which is enabled in the template.

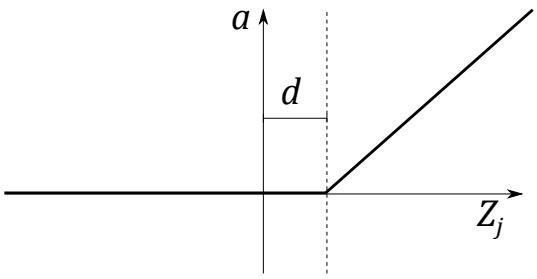
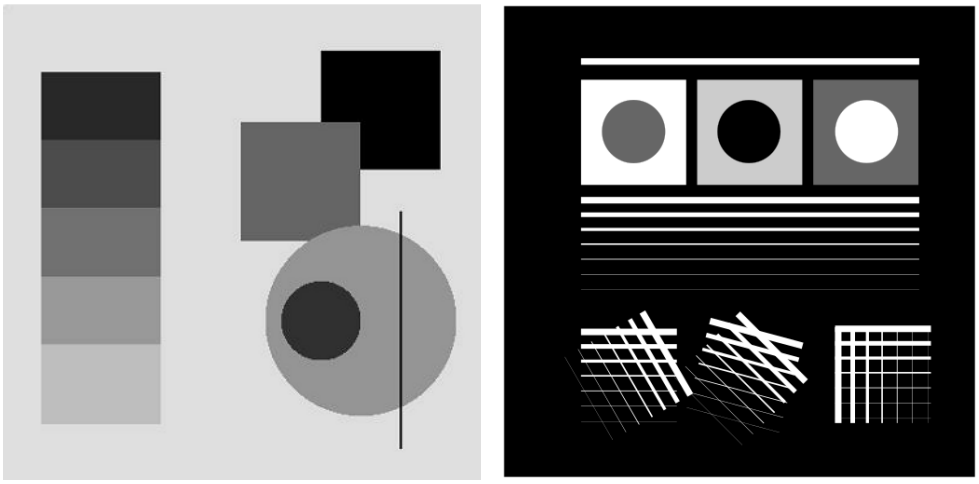


Figure 5.1: *In figure captions, explain what the reader is looking at: "A schematic of the rectifying linear unit, where a is the output amplitude, d is a configurable dead-zone, and Z_j is the input signal", as well as why the reader is looking at this: "It is notable that there is no activation* at all *below 0, which explains our initial results."* ***Use vector image formats (.pdf) where possible****. Size figures appropriately, and do not make them over-large or too small to read.*



(a) *Synthetic image, black on white.* ***(b)*** *Synthetic image, white on black.*

Figure 5.2: *Synthetic test images for edge detection algorithms. (a) shows various gray levels that require an adaptive algorithm. (b) shows more challenging edge detection tests that have crossing lines. Fusing these into full segments typically requires algorithms like the Hough transform. This is an example of using subfigures, with* `subref`*s in the caption.*

5.2.2 Equations

Equations should be typeset correctly and precisely. Make sure you get parenthesis sizing correct, and punctuate equations correctly (the comma is important and goes *inside* the equation block). Explain any symbols used clearly if not defined earlier.

For example, we might define:

$$\hat{f}(\xi) = \frac{1}{2}\left[\int_{-\infty}^{\infty} f(x)e^{2\pi i x\xi}\right], \tag{5.1}$$

where $\hat{f}(\xi)$ is the Fourier transform of the time domain signal $f(x)$.

5.2.3 Algorithms

Algorithms can be set using `algorithm2e`, as in Algorithm 1.

Data: $f_X(x)$, a probability density function returing the density at x.
σ a standard deviation specifying the spread of the proposal distribution.
x_0, an initial starting condition.
Result: $s = [x_1, x_2, \ldots, x_n]$, n samples approximately drawn from a distribution with PDF $f_X(x)$.
begin
    $s \longleftarrow []$
    $p \longleftarrow f_X(x)$
    $i \longleftarrow 0$
    while $i < n$ **do**
        $x' \longleftarrow \mathcal{N}(x, \sigma^2)$
        $p' \longleftarrow f_X(x')$
        $a \longleftarrow \frac{p'}{p}$
        $r \longleftarrow U(0, 1)$
        if $r < a$ **then**
            $x \longleftarrow x'$
            $p \longleftarrow f_X(x)$
            $i \longleftarrow i + 1$
            append x to s
        end
    end
end

Algorithm 1: The Metropolis-Hastings MCMC algorithm for drawing samples from arbitrary probability distributions, specialised for normal proposal distributions $q(x'|x) = \mathcal{N}(x, \sigma^2)$. The symmetry of the normal distribution means the acceptance rule takes the simplified form.

5.2.4 Tables

If you need to include tables, like Table 5.1, use a tool like https://www.tablesgenerator.com/ to generate the table as it is extremely tedious otherwise.

5.2.5 Code

Avoid putting large blocks of code in the report (more than a page in one block, for example). Use syntax highlighting if possible, as in Listing 5.1.

Table 5.1: *The standard table of operators in Python, along with their functional equivalents from the* `operator` *package. Note that table captions go above the table, not below. Do not add additional rules/lines to tables.*

Operation	**Syntax**	**Function**
Addition	`a + b`	`add(a, b)`
Concatenation	`seq1 + seq2`	`concat(seq1, seq2)`
Containment Test	`obj in seq`	`contains(seq, obj)`
Division	`a / b`	`div(a, b)`
Division	`a / b`	`truediv(a, b)`
Division	`a // b`	`floordiv(a, b)`
Bitwise And	`a & b`	`and_(a, b)`
Bitwise Exclusive Or	`a ^b`	`xor(a, b)`
Bitwise Inversion	`~a`	`invert(a)`
Bitwise Or	`a | b`	`or_(a, b)`
Exponentiation	`a ** b`	`pow(a, b)`
Identity	`a is b`	`is_(a, b)`
Identity	`a is not b`	`is_not(a, b)`
Indexed Assignment	`obj[k] = v`	`setitem(obj, k, v)`
Indexed Deletion	`del obj[k]`	`delitem(obj, k)`
Indexing	`obj[k]`	`getitem(obj, k)`
Left Shift	`a <<b`	`lshift(a, b)`
Modulo	`a % b`	`mod(a, b)`
Multiplication	`a * b`	`mul(a, b)`
Negation (Arithmetic)	`- a`	`neg(a)`
Negation (Logical)	`not a`	`not_(a)`
Positive	`+ a`	`pos(a)`
Right Shift	`a >>b`	`rshift(a, b)`
Sequence Repetition	`seq * i`	`repeat(seq, i)`
Slice Assignment	`seq[i:j] = values`	`setitem(seq, slice(i, j), values)`
Slice Deletion	`del seq[i:j]`	`delitem(seq, slice(i, j))`
Slicing	`seq[i:j]`	`getitem(seq, slice(i, j))`
String Formatting	`s % obj`	`mod(s, obj)`
Subtraction	`a - b`	`sub(a, b)`
Truth Test	`obj`	`truth(obj)`
Ordering	`a <b`	`lt(a, b)`
Ordering	`a <= b`	`le(a, b)`

```
def create_callahan_table(rule="b3s23"):
   """Generate the lookup table for the cells."""
   s_table = np.zeros((16, 16, 16, 16), dtype=np.uint8)
   birth, survive = parse_rule(rule)

   # generate all 16 bit strings
   for iv in range(65536):
      bv = [(iv >> z) & 1 for z in range(16)]
      a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p = bv

      # compute next state of the inner 2x2
      nw = apply_rule(f, a, b, c, e, g, i, j, k)
      ne = apply_rule(g, b, c, d, f, h, j, k, l)
      sw = apply_rule(j, e, f, g, i, k, m, n, o)
      se = apply_rule(k, f, g, h, j, l, n, o, p)

      # compute the index of this 4x4
      nw_code = a | (b << 1) | (e << 2) | (f << 3)
      ne_code = c | (d << 1) | (g << 2) | (h << 3)
      sw_code = i | (j << 1) | (m << 2) | (n << 3)
      se_code = k | (l << 1) | (o << 2) | (p << 3)

      # compute the state for the 2x2
      next_code = nw | (ne << 1) | (sw << 2) | (se << 3)

      # get the 4x4 index, and write into the table
      s_table[nw_code, ne_code, sw_code, se_code] = next_code

   return s_table
```

Listing 5.1: *The algorithm for packing the* 3×3 *outer-totalistic binary CA successor rule into a* $16 \times 16 \times 16 \times 16$ *4 bit lookup table, running an equivalent, notionally 16-state* 2×2 *CA.*

6 Evaluation

How good is your solution? How well did you solve the general problem, and what evidence do you have to support that?

6.1 Guidance

- Ask specific questions that address the general problem.
- Answer them with precise evidence (graphs, numbers, statistical analysis, qualitative analysis).
- Be fair and be scientific.
- The key thing is to show that you know how to evaluate your work, not that your work is the most amazing product ever.

6.2 Evidence

Make sure you present your evidence well. Use appropriate visualisations, reporting techniques and statistical analysis, as appropriate. The point is not to dump all the data you have but to present an argument well supported by evidence gathered.

If you use numerical evidence, specify reasonable numbers of significant digits; don't state "18.41141% of users were successful" if you only had 20 users. If you average *anything*, present both a measure of central tendency (e.g. mean, median) *and* a measure of spread (e.g. standard deviation, min/max, interquartile range).

If you use statistical procedures, make sure you understand the process you are using, and that you check the required assumptions hold in your case.

If you visualise, follow the basic rules, as illustrated in Figure 6.1:

- Label everything correctly (axis, title, units).
- Caption thoroughly.
- Reference in text.
- **Include appropriate display of uncertainty (e.g. error bars, Box plot)**
- Minimize clutter.

See the file `guide_to_visualising.pdf` for further information and guidance.

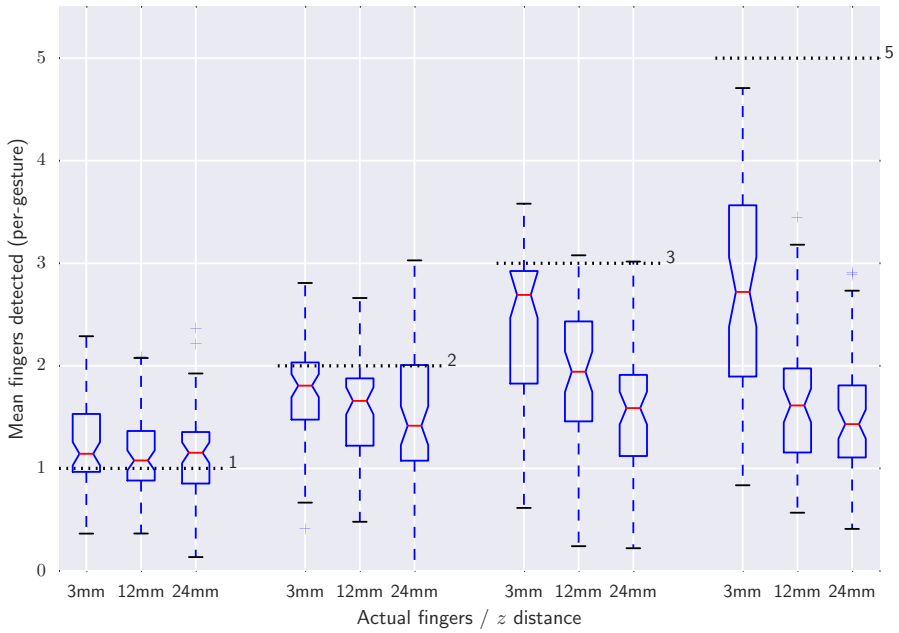


Figure 6.1: *Average number of fingers detected by the touch sensor at different heights above the surface, averaged over all gestures. Dashed lines indicate the true number of fingers present. The Box plots include bootstrapped uncertainty notches for the median. It is clear that the device is biased toward undercounting fingers, particularly at higher z distances.*

7 Conclusion

Summarise the whole project for a lazy reader who didn't read the rest (e.g. a prize-awarding committee). This chapter should be short in most dissertations; maybe one to three pages.

7.1 Guidance

- Summarise briefly and fairly.
- You should be addressing the general problem you introduced in the Introduction.
- Include summary of concrete results ("the new compiler ran 2x faster")
- Indicate what future work could be done, but remember: **you won't get credit for things you haven't done**.

7.2 Summary

Summarise what you did; answer the general questions you asked in the introduction. What did you achieve? Briefly describe what was built and summarise the evaluation results.

7.3 Reflection

Discuss what went well and what didn't and how you would do things differently if you did this project again.

7.4 Future work

Discuss what you would do if you could take this further – where would the interesting directions to go next be? (e.g. you got another year to work on it, or you started a company to work on this, or you pursued a PhD on this topic)

A Appendices

Use separate appendix chapters for groups of ancillary material that support your dissertation. Typical inclusions in the appendices are:

- Copies of ethics approvals (you must include these if you needed to get them)
- Copies of questionnaires etc. used to gather data from subjects. Don't include voluminous data logs; instead submit these electronically alongside your source code.
- Extensive tables or figures that are too bulky to fit in the main body of the report, particularly ones that are repetitive and summarised in the body.
- Outline of the source code (e.g. directory structure), or other architecture documentation like class diagrams.
- User manuals, and any guides to starting/running the software. Your equivalent of `readme.md` should be included.

Don't include your source code in the appendices. It will be submitted separately.

Bibliography

Orwell, G. (1968), Politics and the English language, *in* 'The collected essays, journalism and letters of George Orwell', Harcourt, Brace, Javanovich, pp. 127–140.

Peyton Jones, S. (2017), How to write a great research paper, *in* '2017 Imperial College Computing Student Workshop, ICCSW 2017, September 26-27, 2017, London, UK', pp. 1:1–1:1.

Pinker, S. (2015), *The sense of style: The thinking person's guide to writing in the 21st century*, Penguin Books.

Strunk, W. and Whyte, E. (2007), *The Elements of style*, Penguin.

UNSW (2019), 'How do i cite online sources?', `https://student.unsw.edu.au/how-do-i-cite-electronic-sources`. Last accessed: 2019-02-27.

Vonnegut, K. (1980), *How to write with style*, International Paper Company.

Williams, J. M. and Bizup, J. (2009), *Style: the basics of clarity and grace*, Pearson Longman.