General advice

- Be physically present in the lab during business hours. Unless you have a course.
- Every graduate student gets a desk in the lab. You will need to request a key to one of the secretaries (email keeps changing). It requires a 20$ cash deposit.
- A full lab is a happy lab. It shows to the world we are productive. And it shows you care about the well-being of the lab.
- When you are in the lab, we can have impromptu meetings. These 5 minute meetings might save you a week of debugging or going in the wrong direction.
- Students have lunch together.
- The lab usually goes for drinks on Fridays around 16h.
- All code should be saved in our github.
- We use SLACK extensively. It's good to interact with it, be logged on as much as possible. Plus INTERACT a lot with reactions! A thumb up 👍, a smile or laugh 🤣 or whatever emoji you like. It keeps it lively!
- You will be asked to lend a hand for experiments. In return, you will get help later in your studies. And it's good for team building.
- You will need a Compute Canada account. The link to set it up is the following: https://www.computecanada.ca/research-portal/account-management/apply-for-an-account/ . I need to be your sponsor, and for this you need to know my (Philippe) CCRI: **`wjf-342-01`** .
- We have a quick start guide to get ou up-to-speed on using Compute Canada ressources : https://github.com/norlab-ulaval/Norlab_wiki/wiki/Compute-Canada-Quickstart
- Consider seriously registering to the `robotics-worldwide` mailing list. There are often lots of interesting opportunities (datasets, code, jobs, internships, conferences, etc)
- If you feel you are lacking some computer science skills, check this MIT site for a quick overview of things like shell scripting, data wrangling, code versioning, editors etc.
- To get some tips about searching for information, there is a free training course called *Contruire ma réussite* here.
- To quote the movie Idiocracy: "*You like money? I like money too!*". Everyone loves money. Our University gives financial incentives when you complete certain milestones in time. The information is available below. The **onus is on you chasing the money**, not on us reminding you to do it.
    - https://www.ift.ulaval.ca/fileadmin/ift/documents/PDF/Doctorat/Bourses-re%CC%81ussite-FESP-2018-2019-FSG.pdf
    - https://www.fsg.ulaval.ca/fileadmin/fsg/documents/PDF/Procedure-bourse-FESP-Capsule.pdf

Scholarships

- It's important to apply to all possible scholarships (NSERC, FR-QNT). First, it gives you money. Second, it might free up some budgets for other activities (conferences, field trips, equipment). But importantly, it helps build your academic resumé. Future scholarships depend on previous ones. Even when you apply to a professor position, we look to see if you have won any scholarships during your undergrads. Why? Because it's the closest thing to writing a grant. And it shows your willingness to write, to sell yourself, and demonstrates your excellency.
- We have a number of examples of applications from previous students. You can find them here: https://drive.google.com/drive/folders/1w5UBDKLxsHYPUfcCfwgYQVSNDs5AlAJz?usp=sharing.
- Have your supervisor help you fine-tune and proofread your demand. Ask a colleague to proof-read your demand.

- **For master : 4 paragraphs context/problematic, one objective, methodology, impact (2-3 sentences). At least 6 references, ideally 10. NO BLANK SPACE. No need to write the title, nor to use page numbers. References DO NOT count on the 1 page limit.**
- **For PhD : at least 3 objectives, to target 3 papers essentially.**

Reading Papers

- There will be a lot more reading than implementation at the beginning of your research, which we are not used to as bachelor students. See https://web.stanford.edu/class/ee384m/Handouts/HowtoReadPaper.pdf for some tips on reading papers (thanks William!)
- Our conferences:
    - **Robotics**: RSS, ICRA, IROS, FSR (now defunct), CRV
    - **Computer vision**: CVPR, ICCV, ECCV, WACV, BMVC, CRV
- Our journals:
    - **Robotics**: IEEE Transactions on robotics (TRO), Journal of Field Robotics (JFR), Autonomous Robotics (AuRo), Robotics and Autonomous Systems (RAS), RA-L (letters)
    - **Computer Vision**: T-PAMI
- You are only as good and knowledgeable as the number of papers you read. The more the better. Get in the habit of reading 1-3 papers per week as a background activity. During paper writing you might need to read a lot more, but quickly. Or when you are investigating an area.
- Keep track of all your readings via Mendeley (https://www.mendeley.com/).
- Create tables to synthesize the papers you have read that are going to be part of your *Previous Work* section of a paper/thesis. An example of a table is like this:

Table 2.1 – Comparison of deep learning approaches to grasping

Paper	Date	Modality	Feature Extraction	Proposal Approach	Single Output	Predicts Object Class	Predicts Grasp ability	Angle Discretization (bins)	Spatial Discretization (grid)	Anchor Boxes	Trained on
Joseph Redmon and Angelova (2015), Direct Reg.	2015	RG-D	AlexNet	-	✓	-	-	-	-	-	CGD
Joseph Redmon and Angelova (2015), Reg. + Class.	2015	RG-D	AlexNet	-	-	✓	-	-	-	-	CGD
Joseph Redmon and Angelova (2015), MultiGrasp	2015	RG-D	AlexNet	-	-	-	✓	-	7×7	-	CGD
Kumra et al. (2017)	2016	RGB-D	ResNet-50	-	✓	-	-	-	-	-	CGD
Pinto et al. (2016)	2016	RGB	AlexNet	random sampling	-	-	18	-	-	-	New data (700k grasps)
Johns et al. (2016)	2016	Depth	AlexNet	-	-	-	✓	6	33×44	-	New data (1000 ShapeNet objects)

  The selection of columns will be a challenge, but it's part of the thinking process. Once you have a table like this, it's easier to cluster prior work. Each cluster of work then becomes its own paragraph or sub-section in the Previous Work section. I encourage you to use Google Sheets for those tables, so I can easily peruse them.
- Set up the proper *google scholar* alerts on the topic you are investigating. The alerts might serve as a way to ensure your weekly minimum reading.
- If you want to sift through all the daily new arxiv submissions, check http://www.arxiv-sanity.com/

- Take the time to read papers that are sometimes a little bit outside of your area of research. It's good to have some broadness, and you never know when a paper might be useful to you. It might be 10 years down the line that you find the use for an idea you read in a paper!
- Get involved in reading groups.

Experimentations

- **You might want to consider *Failing Fast***. Don't stick to an idea if it will not produce results. Of course, you also need to be persistent. Find a happy trade-off between the two. Also graduate studies is like a safe-space where you are allowed to fail, without hard consequences (most of the time).
- **Always save your intermediary results in an easily-readable format (CSV or others)**. For instance, if you do tree diameter estimation, save the estimation for all trees in a single file. Then write a script that will extract the statistics from that file. That's instead of having a giant script that takes the raw input data, performs data processing, sends it to the neural net/geometric approach, computes statistics, and produces a graph. The reason is that later on, for a journal paper extension for instance, someone else might need the raw results to plot better results.
- **Always expect to redraw the same figure in 4 different formats**. One for the paper, one for the presentation, one for your thesis, and one for the poster. This means that you should have a script that can be easily modified to redraw the same figure. This also means that the data needed to make the plot can be accessed easily at any time. See previous item.
- **Always have intermediary visualizations showing the data/results at various stages of the pipeline.** Think of it as a way to explain to some neophyte how your algorithm works, at each stage. Think of plotting them almost all the time, when you do your experiments. This way:
    - You will catch bugs early.
    - You will be able to debug easily, by seeing where the bugs appear.
    - You often need a figure in a paper/presentation that shows the processing pipeline. You will be able to visually demonstrate the pipeline, for free.
- **If you do deep learning on images, always check your training pipeline for bottlenecks on disk, CPU and GPU usage.** You can do this with `top` (or `htop`) and `nvidia-smi -lms 500` . Your GPU must be close to 100% usage during training. If not, you have a problem. First, make sure you have the SIMD version of Pillow installed. These can increase by 400% the throughput of CPU image processing (which could solve the CPU bottleneck). Also remember that Linux does file caching in RAM. So if your dataset size on disk is smaller than like 30% of your RAM size, the linux file system will silently cache it in RAM entirely, bypassing the disk.
- **In deep learning, nights have been invented to train or search for hyperparameters.** Think about having a generic script to run such training, for instance with a list of hyperparameters. In the morning, you can start your day by sifting through results. An unused GPU is a useless GPU.
- **Six months of working on tweaking algorithms might save you three boring weeks of collecting and labelling data.** Think about it ;)

Writing

- A complete lifecycle of an idea is: Workshop paper → conference paper → journal paper.
- Not all ideas will do the complete cycle. If the idea + experimentations are mature, we go straight to a conference paper. FSR is nice as it translates to an invitation to a journal, if the paper is good.
- Journals are worth more, in the view of the community. They also typically gather 3-5 times more citations than conferences.
- You will have to write a lot, to improve your scientific communication skills. The more you write, the better you will get at it.
- You can't write more than 2 hours a day in a productive manner. So, plan to write over many days.
- At the end of your 2 hours, you might feel that there are a few things that you would have liked to keep writing about. Pen these things down quickly. Next time you sit down to write, you already have something to get you started. Getting started is always the hardest part.
- If it helps, just write things down as bullet lists. It will help structure the document/select what to talk about or not. Then writing becomes expanding these lists into sentences or paragraphs
- Not everything you write will make it in the paper/thesis. But every bit of experience will stay with you.
- It's easier for your supervisor to guide you when he has a written-down document that he can focus on, all at once.
- In my experience, it takes in general 5 iterations of corrections between a student and a supervisor before a text is in a good state.
    - The first iteration will have a LOT of corrections. Don't feel bad. Don't feel intimidated.
    - The following ones will have exponentially-decreasing number of corrections.
- Read the following sites:
    - https://en.wikipedia.org/wiki/Weasel_word
    - How to write a good abstract: https://users.ece.cmu.edu/~koopman/essays/abstract.html
    - To help you figure out what goes into an introduction vs. what goes into prior work: https://cs.stanford.edu/people/widom/paper-writing.html
- A paper should answer these questions (notice the overlap with the one in the website above):
    - Why is this problem important?
    - Why wasn't it solved before?
    - What's the key idea in the solution?
    - How do you show that it really works?
    - How can others use your results?
- C Compiler rule applies: concepts should be explained/described before they are referenced!
- Clean code rules apply too! Concepts should be declared NEAR where they are used.
- We use the package for abreviations \usepackage{**acronym**}
- Use autoref. See https://en.wikibooks.org/wiki/LaTeX/Labels_and_Cross-referencing
- For more packages, look at our official preamble `.tex` file here:
    - `https://github.com/norlab-ulaval/latexGoodPractices/blob/master/preamble.tex`
- Nice figures tell the whole story. When you think about a paper, you should envision **THE figure** that will capture the idea and results. Then your research work becomes producing this figure.
- For some other tips, see these two presentations from influential people : Cyrill Stachniss and Michael Milford.

- Do not forget punctuation in equations:

these two stages can be summarized as estimating the rigid transformation $\hat{\boldsymbol{T}}$ by minimizing

$$\hat{\boldsymbol{T}} = \arg\min_{\boldsymbol{T}} \sum_{i=1} \sum_{j=1} \rho\Big(e(\boldsymbol{T}, \boldsymbol{p_i}, \boldsymbol{q_j})\Big), \quad (1)$$

where $\rho(\cdot)$ is the cost function. The error function $e(\cdot)$ is the

functions associated with M-estimators are analyzed using their influence function $\psi(\cdot)$ and weight function $w(\cdot)$, such that

$$\psi(e) = \frac{\partial \rho(e)}{\partial e} \quad \text{and} \quad w(e) = \frac{\psi(e)}{e}$$

The influence function $\psi(\cdot)$ is used to evaluate whether an

- For your thesis' list of figures/tables, make sure you create both **long** and **short captions** in latex: `\caption[Short version for LoF]{Long version to appear next to the figure}`
- Journal papers get roughly 4 times the number of citations than their conference versions. They are worth it.
- Workshop papers can be good for exposure (you can have more people in a workshop track than a paper track) and to bounce ideas. And to force you to write something, that will eventually become a conference paper.
- Figures are very important. Well-polished figures will help your paper get accepted. For some guidelines regarding figures, see this cheatsheet (Thanks Johann Laconte): https://drive.google.com/file/d/1mteAT-clCJ7PDZbpQKn5FWdzQj7A98MW/view?usp=sharing

Oral presentations

- A visually-pleasing presentation wins. It's sad, but the container matters as much as the content. This is why we have marketing departments in businesses. A product won't sell by itself. Your research IS a product.
- Use our lab logo (`https://github.com/norlab-ulaval/visualIdentity`). The university logo.
- Do not use *beamer* or *google slides*. Why?
    - *Beamer*: it makes it difficult to have intuitive placement of text and images, so you will not be inclined to do it. And it's nearly impossible to do animation.
    - *Google slides*: the font size and arrangement by default is horrendous. When you give your talk, the network WILL be down, and you won't be able to access your presentation online. Ask people for our default powerpoint Norlab template.
- Ensure that the TAKE HOME message on each slide is clear. Why are you showing me this graph? This slide?
- Have very limited text. Use telegraphic style.
- Make sure you adapt your content to the audience.
- C Compiler rule applies: concepts should be explained/described before they are referenced!
- Clean code rules apply too! Concepts should be declared NEAR where they are used.
- Everybody likes to learn a little something. So if you give a talk to a research group, it's nice to have 5-10 minutes of explaining an algorithm or a technology.
- Equations are nearly impossible to digest in a talk. Make sure you leave enough time to explain every term/variable in it. With that rule in place, you will realize you won't have enough time to explain more than a handful (one?) equation. That is why we have papers: to store and explain equations.
- Practice many times. 10 times. Over a few days. Take notes on what are the key sentences you like.
- Do a practice talk in front of our group. People will get exposed to your project, and you will get invaluable feedback.

Administration

- **Purchases**. The purchasing process at Laval U. is **HEAVILY regulated**, and is CONSTANTLY CHANGING. Before making purchases, always consult with your supervisor. Purchases of equipment MUST be done with a purchase order (*bon de commande*) or a virtual credit card number ISSUED by the university. Failure to respect these rules might mean you do not get reimbursed, or that our lab keeps accumulating negative karma. The sub-200$ rule has been abolished.