Software Engineering Assignment – Spring 2025

José Miguel de Almeida Pedro

1 Introduction

A large subfield of visual perception is visual localization and mapping, where we are interested in estimating a map/representation of the environment that is consistent across multiple views and, simultaneously, being able to determine the positions and orientations of the cameras that captured those images. Historically, a large amount of work focused on determining points of interest that could be reliably detected and identified across multiple images, using these points and their correspondences across images to build sparse maps of the environment. While this line of work remains state of the art in large-scale localization, the maps it produces are of little use for tasks other than localization due to their sparsity and lack of information. Meanwhile, recent developments in machine learning and deep learning have given rise to radiance field representations. These maps fully describe the visual appearance of the scene and allow rendering photorealistic images, making them potentially more useful for many tasks. However, they are comparatively much worse for localization, requiring bootstrapping from the previous sparse maps.

My research focuses on better understanding localization in photorealistic representations, also called direct localization, and improving the performance of such methods. Since direct localization is typically formulated as an optimization problem and solved with first-order optimizers, one of the failure modes is for optimization to converge in a local minimum. I am currently investigating how to incorporate generative modeling into the optimization process, in order to more consistently reach the global minimum.

2 Lecture Principles

The two lecture principles that most resonated with me were Property-based Testing and Data Validation.

2.1 Property-based Testing

Property-based testing amounts to validating that the software that has been implemented fulfills some known axioms of the problem. In the lectures, the example given was that reversing a list twice should yield the original list. As I understand it, the main point here is that stating and verifying this axiom does not require knowing how to implement the list reversal program, while still providing very strong constraints on what programs pass the test – only functions on lists which are their own inverse are valid.

In my research, this idea is very important for properly handling camera *poses*. The pose consists of a position and orientation, and while any set of three numbers is a valid position, orientations impose constraints onto their representation. Therefore, it is important to ensure that all components of the localization pipeline that manipulate poses, whether hand-crafted or learned from data, preserve the structure of pose representations. This kind of property-based assertions are ubiquitous throughout my code.

2.2 Data Validation

Data validation has played an important role in my current project. Visual localization encompasses several problems and the one I am working one is about determining the relative pose between two images. For this problem to even make sense, there has to be some *visual overlap* between the images, i.e., there has to be a portion of both images that sees the same thing. Therefore, a large part of the beginning of the project was curating and generating data with desired overlap properties and validating it.

3 Guest Lecture Principles

My PhD project is an industry collaboration between Ericsson and KTH, so the organizational aspects discussed in the guest lectures were quite relevant. The two principles that I think are most related to my research are Problem- vs Solution-space and System Vision.

Since Ericsson is a telecommunication company and my research is in computer vision, there is somewhat of a mismatch between the technical expertise of most people at the company and mine. The scope of my project exists at the intersection between these two areas of expertise and the company stakeholders mainly care about how my research enables the company's main business interests. Therefore, in order to have productive discussions, it is very important to steer them towards problem-space discussions and ensure we have a system vision of the problem. Otherwise, we risk getting stuck discussing minute technical problems that are not in both our areas of expertise, which has happened.

4 Data Scientists vs. Software Engineers

> Do you agree on the essential differences between data scientists and software engineers put forward in these chapters? Why or why not?

The book characterizes data scientists as preferring to focus on model building, spending a lot of time on data curation and not focusing so much on practicalities like training costs or inference latency. Software engineers are characterized as being more solution oriented, focused on keeping to time and cost budgets, with a larger emphasis on the testing, deployment and maintenance of the solution than its development.

While I think that there is a real distinction between a software engineer and a data scientist, I think the portrayal of data scientists in the book is too narrow. Either very few people are data scientists as described, or the definition is insufficient. Today, developing a machine learning model requires some basic notions of software engineering, in order to deal with the ever growing scale of models, the necessity to distribute them (even during training), monitoring their training, etc.

The book has a figure where they depict software engineers as possessing/requiring a set of skills and data scientists another, with little overlap. I think, as the field of data science progresses, this overlap becomes larger. However, I don't think it will ever be total, and in that sense I agree with the distinction between the roles.

> Do you think these roles will evolve and specialise further or that "both sides" will need to learn many of the skills of "the other side" and that the roles somehow will merge? Explain your reasoning.

I think software engineering, both as a discipline and as a set of skills, is much more mature than data science. Therefore, I think there is a clear need that is being met by software engineering, and software engineers don't *have* to learn so many of the skills of data scientists as data scientists have to learn the skills of software engineers.

I can also see that software engineering will become even less about writing computer programs, as highly complex tasks become increasingly dominated by data-driven solutions. In that scenario, the role of software engineers will shift even more towards the infrastructure supporting these data-drive solutions, as well the project management aspects, and less on the business logic.

Data scientists will have to pick up many infrastructure skills from software engineering, as training models becomes increasingly complex and laborious.

5 Paper analysis

5.1 Engineering Challenges for AI-Supported Computer Vision in Small Uncrewed Aerial Systems [1]

Core Ideas: This paper examines the challenges of integrating data-driven computer vision solutions in small UAVs. They propose to split the development into three tracks: computer vision, hardware and software; initially, each component can be developed individually, and then integrated with each other component separately, and finally integrating all components together. Testing follows a similar approach, with each component being tested individually, then pairwise, and finally complete systems-level testing.

This incremental approach to development and testing enables the transition from the development of the pure AI component of the computer vision part, to the software engineering methodology to integrate it into a functioning product.

Relation to my Research: My research centers around developing the computer vision component that would eventually be integrated into such a system. Therefore, their integration and testing methodology provides useful points to consider when designing experiments, as well as driving development. For example, the integration and testing between the computer vision and hardware components aims to validate that the developed system is actually robust to the hardware that is going to be used. This includes making sure that the computer vision component is robust to the actual noise characteristics of the camera, to the motion blur during normal motion conditions, etc. These are all aspects that are often underrepresented in typical computer vision training datasets (although they are much more common in robotics datasets), so keeping these things in mind is very important to ensuring the outcome of my research is generally useful, not just a gimmick that improves some number in some benchmark.

Integration in a Project and Adaptation: The ideas put forward in the paper are, in essence, its integration in a project. In order to be better aligned with my WASP project, we could apply this development methodology to an augmented reality (AR) project. My research would provide accurate localization of the headset, and would require testing and validation of the hardware components, ensuring its robust to real world conditions, and the software, ensuring the estimates from the computer vision component lead to correct overlays, etc. I could adapt my research to better integrate with this methodology by ensuring that, at least in part, training data contains data from the target hardware, as well as evaluating not just on computer vision metrics, but also on targets that are required by the application.

5.2 Automotive Perception Software Development: An Empirical Investigation into Data, Annotation, and Ecosystem Challenges [2]

Core Ideas: This paper examines the lack of adequate requirements for data annotation for machine learning in the automotive industry. Through interviews with industrial practitioners from the automotive industry in Sweden, they look to identify the causes for poor quality annotations, that in turn leads to poor performance of deployed machine learning models. They identify several main causes, namely: ambiguous requirements for annotations (e.g., precision vs consistency), low diversity in the data collection process, and poor collaboration between data collectors and annotators. They propose to introduce iterative dataset curation, continually refining the dataset to improve labeling. They also make the claim that consistency in labeling and diversity in data are more important for model performance than raw precision.

Relation to my Research: Pose estimation is a field with scarcely labeled data, since accurately measuring the pose of a camera often requires expensive motion-capture systems. This limits the amount of data, but also its variation, since motion-capture systems only work indoors. In practice, a lot of the data is *pseudo*-labeled, i.e., a robust technique for camera pose estimation is used to get labels. This often involves using methods which are computationally expensive, and would not be suitable for real-time pose estimation, as well as integrating sensor data from multiple sources. The prevailing focus in the literature is in very accurate labeling, which seems to not follow the conclusions of this paper, so there could be room to revisit pseudo-labeling of camera pose estimation datasets.

Integration in a Project and Adaptation: The main idea of iterative dataset curation can be immediately applied to my project. I am developing a machine learning-based method for camera pose estimation and I am fully in control of the data generation for training. Therefore, I can (and, in some sense, already do) apply this iterative approach to generating training data, by evaluating my models on varied held-out data and using the performance as a guiding signal for curating the training dataset. The challenge here will be to not allow this iterative data curation process to poison the test set, which will require choosing the test set upfront and then only evaluating it once development is finalized, and basing the iteration on a validation set instead.

6 Research Ethics & Synthesis Reflection

To choose the papers, I looked through the long-paper proceedings from CAIN for 2023, 2024 and 2025. I ended up choosing both papers from CAIN 2023 because they were the ones that I felt were most directly relevant for my research, since both of them focused on perception systems. My search process consisted of screening the titles and, if the title looked relevant, reading the abstracts. I felt like the titles and the abstracts were not misleading and I did not encounter a situation where I read the paper that was not roughly what I expected from the title and abstract.

Originality: To ensure originality, I refrained from looking at other colleagues' assignments on the repository or from using LLMs. I read the papers that I discussed, the first two chapters from the "Machine Learning in Production" book, the slides available on Canvas and the notes I had taken during the course. The material in this assignment results from my interpretation of those materials and the writing is my own. Therefore, to the extent that one can be original, this assignment should be original content.

References

[1] Muhammed Tawfiq Chowdhury and Jane Cleland-Huang. Engineering challenges for ai-supported computer vision in small uncrewed aerial systems. 2023.

[2] Hans-Martin Heyn, Khan Mohammad Habibullah, Eric Knauss, Jennifer Horkoff, Markus Borg, Alessia Knauss, and Polly Jing Li. Automotive perception software development: An empirical investigation into data, annotation, and ecosystem challenges. *arXiv preprint arXiv:2303.05947*, 2023.