Research Disassembly

|  |
| --- |
| **Date** **:** **18-03-2024** |
| **Version** **:** **1** |
| **Status** **:** **In progress** |
| **Author** **:** **Tim Spieringhs** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Amendments** | **Status** |
| 1 | 21-03-2024 | Tim Spieringhs | Setup document | In progress |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **To** |
|  |  |  |
|  |  |  |
|  |  |  |

# Preface

Table of contents

[Preface 2](#_Toc161647976)

[Problem definition 4](#_Toc161647977)

[Cause 4](#_Toc161647978)

[Goal 4](#_Toc161647979)

[Main question and sub-questions 4](#_Toc161647980)

[Solution and existing solutions 4](#_Toc161647981)

[What is the DOT framework? 5](#_Toc161647982)

[Research sub-question 1 6](#_Toc161647983)

[Workshop: Brainstorming 6](#_Toc161647984)

[Workshop: Requirements priorization 6](#_Toc161647985)

[Conclusion 6](#_Toc161647986)

[Research sub-question 2 7](#_Toc161647987)

[Workshop: Brainstorming 7](#_Toc161647988)

[Workshop: Requirements priorization 7](#_Toc161647989)

[Conclusion 7](#_Toc161647990)

[Research sub-question 3 8](#_Toc161647991)

[Library: Community research 8](#_Toc161647992)

[Conclusion 8](#_Toc161647993)

[Final conclusion 9](#_Toc161647994)

# Problem definition

# Cause

# Goal

# Main question and sub-questions

Main Question: What computer vision approach can be used to properly detect product parts - in the context of product disassembly?

**Sub-question 0:** What are the existing approaches in the industry, and why have these been chosen?

**Sub-question 1:** How best to set up/focus the computer vision approach to make 'learning' a product as easy as possible (little resources, effort and time saving)?

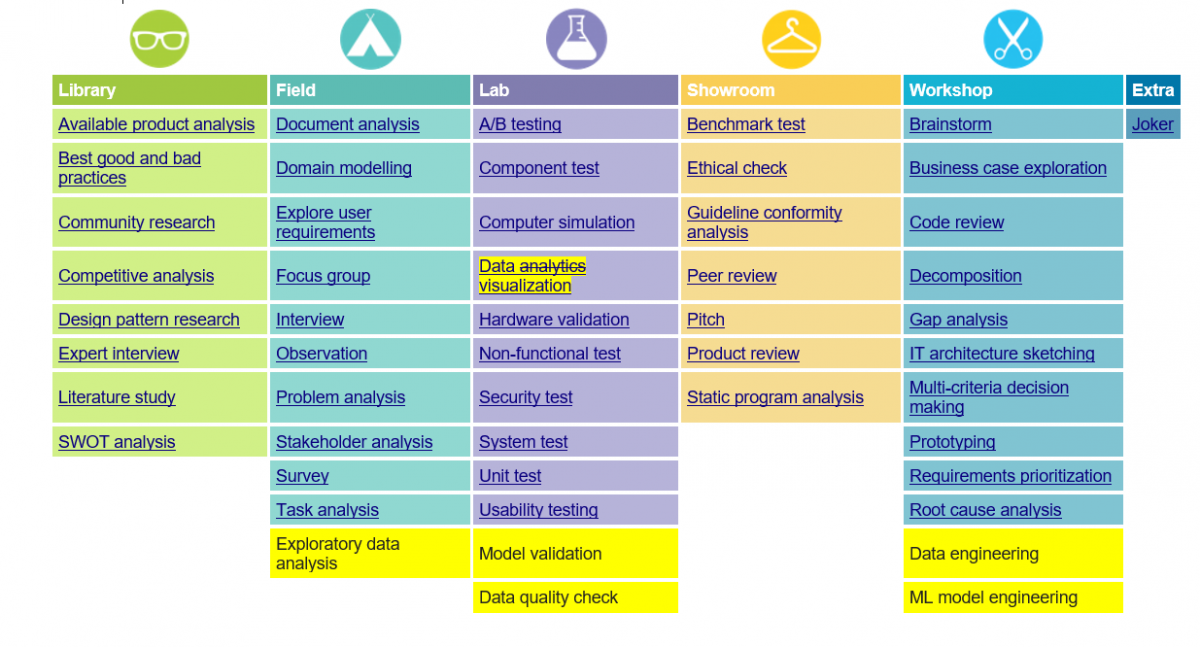
**Sub-question 2:** How can we make it easier for computer vision methods to detect products?

**Sub-question 3:** How can we integrate identification data with manipulation tasks (robotic or human) and later tasks in the disassembly line?

# Solution and existing solutions

# What is the DOT framework?

The DOT framework is a research method primarily used in the ICT world. This framework consists of 5 components: library, field, lab, workshop, and showroom.

1. The library (library) is about researching and dissecting existing solutions to the problem.
2. The field (field) is about the end users. Here, you mainly look through the eyes of the end users and map out the problems they may encounter.
3. The lab is about testing your ideas. You check if your idea is feasible, or if it works the way you want it to.
4. The workshop (workshop) is about creating sketches, prototypes, and other creations. By creating it, you can quickly see what works and what doesn't. You also analyse what can be improved.
5. The showroom is a combination of the library and workshop/lab, as you will have your idea or prototype reviewed by experts.

# Research sub-question 0

**Sub-question 0:** What are the existing approaches in the industry, and why have these been chosen?

## Introduction

## Library: Available product analysis

**TensorFlow**TensorFlow is one of the most widely used deep learning frameworks, backed by Google and has a large and active community. It offers the following tasks:

* **Data Preprocessing:** In TensorFlow you can clean, normalize, and enlarge your data to prepare it for effective model training.
* **Model Training:** The framework provides a flexible environment for building and training deep learning models. It supports various neural network architectures, allowing you to tailor your model to your specific task.
* **Production Deployment:** TensorFlow doesn't just excel in research, it's also production-ready. The framework offers tools for deploying trained models in real-world applications, enabling you to leverage your deep learning models at scale.

**PyTorch**PyTorch has emerged as a strong contender in the deep learning arena backed by Meta and Linux, known for its ease of use and flexibility. Backed by a large and active community, it offers a powerful set of tools for various deep learning tasks.

* **Dynamic Computational Graph:** Unlike TensorFlow's static approach, PyTorch utilizes a dynamic computational graph. This means you can change data easily during training making it way more flexible. Yet this also makes the training slower.
* **Rich Ecosystem of Libraries:** PyTorch is easy to implement with other popular scientific computing libraries like NumPy and Pandas, making it familiar and comfortable for many data scientists. Additionally, a vast ecosystem of third-party libraries makes PyTorch's a good option when it needs to multiple tasks.
* **Production Deployment:** While research-oriented, PyTorch offers tools for deploying trained models in production environments. Libraries like TorchScript can help convert PyTorch models into a more optimized format for production use.

**OpenCV**OpenCV isn't directly backed by a single company like Microsoft with Azure AI or Google with TensorFlow. Instead, OpenCV thrives as a vibrant open-source project. This means it's a collaborative effort driven by a global community of developers and researchers. It has the following tools:

* **Community-Driven Development:** Open source makes it so that there are way more improvements through contributions from a vast developer pool. This ongoing development ensures OpenCV stays updated with the latest advancements in computer vision algorithms and functionalities.
* **Accessibility and Transparency:** Being open-source makes OpenCV free to use and modify for any purpose. This accessibility allows anyone to leverage its functionalities in their projects, encouraging innovation and widespread adoption.
* **Platform Independence:** OpenCV boasts cross-platform support, meaning it can run on different operating systems like Windows, Linux, and macOS. This flexibility makes it a valuable tool for developers working in diverse environments.
* **Extensive Documentation and Community Support:** The open-source community surrounding OpenCV has produced a lot of documentation, tutorials, and online forums. This large support system makes it easier for new users to learn and use OpenCV effectively.

**Azure AI**Azure AI, a cloud-based deep learning services from Microsoft, empowers developers and data scientists to build, train, and deploy intelligent applications without managing complex infrastructure. It has the following tools:

* **Cloud-Based Scalability:** You can use Microsoft's cloud infrastructure for training and deploying deep learning models. You can also scale resources up or down as needed, ensuring resource efficiency.
* **Pre-trained Models & Services:** Azure AI offers a rich library of pre-trained models for multiple tasks, including object detection and image classification. This eliminates the need to train models from scratch, saving time and resources. Additionally, Azure AI provides managed services like Custom Vision and Personalizer, offering a user-friendly interface for building models without a lot of coding.
* **Integration with Azure Ecosystem:** Azure AI works really well with other Azure services like Azure Storage and Azure Machine Learning. This allows for centralized data management, model training workflows, and deployment within the Azure environment.
* **Flexibility & Customization:** While offering pre-trained models and services, Azure AI also provides flexibility for experienced developers. You can use frameworks like TensorFlow and PyTorch within Azure to build and customize your own deep learning models.

**YOLO**YOLO is a cutting-edge object detection algorithm known for its real-time performance and accuracy. Unlike traditional object detection algorithms that require multiple passes through an image, YOLO YOLO detects an object and its bounding boxes in a single pass through. This approach enables YOLO to achieve remarkable speed without having less accuracy, making it really usefull for applications requiring real-time object detection, such as video surveillance, autonomous vehicles, and robotics. YOLO models are typically trained using deep learning techniques, and implementations are available in popular deep learning frameworks like OpenCV, TensorFlow and PyTorch. This means it’s a addon and not a framework itself. I added YOLO to the research because it can possible add to every single framework but it’s not a contender.

## Library: Expert interview

**Peter Lambooij interview**  
I had a conversation with Peter Lambooij, an esteemed expert in computer vision, to discuss the objectives of our project and gather his insights on how to approach it effectively using computer vision techniques. During our discussion, Peter emphasized the importance of diversifying our approach rather than relying solely on one solution. He suggested incorporating multiple methods such as colour and shape detection along with utilizing barcodes to enhance the success rate of our project.

Moreover, Peter highlighted the significance of considering factors that may be intuitive to humans but not inherently understood by machines. For instance, he pointed out the role of shadows, which are typically perceived naturally by human eyes but may require explicit training for machines to recognize. This underlines the need to incorporate such nuanced aspects into our computer vision models to ensure their robustness and accuracy in real-world scenarios.

**AM-Flow**AM-Flow is a company that makes sorting systems for 3D printed parts that also checks if the product is up to standards. **T**hey tackle the problem of detecting products that have not been seen before through the 3D models and using that data to detect products. They are mainly focused on shapes because of the simplicity of the products they need to detect. They use 2 sets of 4 camera`s 1 set detects the bigger parts while 1 set that is more zoomed in detects the smaller parts. They also deal with shadows through flashing the product when the pictures are being taken from multiple sides.

AM-Flow is a leading company specializing in advanced sorting systems for 3D printed parts. Our cutting-edge technology not only ensures efficient sorting but also guarantees that all products meet the highest quality standards. Our innovative solutions address the challenge of detecting previously unseen products by leveraging 3D models and utilizing data-driven detection techniques.

With a primary focus on shapes, we simplify the detection process for various products. To accomplish this, we employ two sets of four cameras. One set is designed to detect larger parts, while the other set zooms in to accurately detect smaller components. Additionally, we have developed a unique approach to overcome the issue of shadows. By strategically flashing the product during the imaging process from multiple angles, we effectively minimize the impact of shadows and optimize the accuracy of our detection system.

## Library: Competitive analysis

**Azure AI**

* **Unique selling point:** Easy-to-use, cloud-based platform for building, training, and deploying deep learning models without managing infrastructure. Ideal for businesses or individuals who lack in-house deep learning expertise or infrastructure expertise.
* **Focus:** Accessibility, Pre-trained Models, Scalability
* **Ideal for:** Ideal for building and deploying deep learning models in specialized industries like healthcare or finance, leveraging Microsoft's industry-specific cloud solutions.

**TensorFlow**

* **Unique selling point:** Highly flexible and powerful framework for various deep learning tasks, backed by Google's research and development. Ideal for researchers, data scientists, and developers who require maximum control and customization of their models.
* **Focus:** Flexibility, Research & Development, Large Community
* **Ideal for:** Ideal for research in deep learning, natural language processing, and other advanced AI fields due to its flexibility and extensive research community.

**PyTorch**

* **Unique selling point:** Known for its dynamic computational graph, making it easier to experiment and debug models compared to TensorFlow's static approach. Popular among researchers who value ease of use and a Pythonic coding style.
* **Focus:** Ease of Use, Research, Rich Ecosystem of Libraries
* **Ideal for:** Ideal for research due to its dynamic computational graph, PyTorch allows researchers to experiment and debug on the fly, accelerating deep learning exploration.

**OpenCV**

* **Unique selling point:** Free, open-source computer vision library with a large community and a lot of documentation. Perfect for developers and hobbyists who need real-time computer vision functionalities without a heavy investment.
* **Focus:** Open-Source, Real-Time Computer Vision, Large Community Support
* **Ideal for:** Idealfor robotics applications or embedded systems that require real-time computer vision tasks with limited computational resources.

## Conclusion

|  |  |  |
| --- | --- | --- |
| **TensorFlow** | | |
| **Aspect** | **Reasoning** | **Grading 1-10** |
| **Accuracy** | Can achieve high accuracy with careful model design and training. | 8 |
| **Speed** | Slower than OpenCV for basic tasks, but can be optimized for performance. | 7 |
| **Community & Support** | Large and active community with extensive documentation and resources. | 9 |
| **Training Time** | Flexible training capabilities, but training complex models can be time-consuming. | 7 |
| **Flexibility** | Highly flexible for various deep learning tasks. | 9 |
| **Ease of Use** | Steeper learning curve compared to OpenCV, but good tutorials are available. | 7 |
| **Resource Efficiency** | Resource usage varies depending on model complexity, but can be optimized. | 7 |
| **Total** | | 54 |

|  |  |  |
| --- | --- | --- |
| **PyTorch** | | |
| **Aspect** | **Reasoning** | **Grading 1-10** |
| **Accuracy** | Similar to TensorFlow in achieving high accuracy with proper model design and training. | 8 |
| **Speed** | Similar speed considerations as TensorFlow; can be optimized for performance. | 7 |
| **Community & Support** | Growing and active community with good documentation and resources. | 8 |
| **Training Time** | Flexible training capabilities, training time depends on model complexity. | 7 |
| **Flexibility** | Highly flexible for various deep learning tasks. | 9 |
| **Ease of Use** | Considered slightly more user-friendly than TensorFlow for research. | 8 |
| **Resource Efficiency** | Similar resource usage considerations as TensorFlow. | 7 |
| **Total** | | 54 |

|  |  |  |
| --- | --- | --- |
| **OpenCV** | | |
| **Aspect** | **Reasoning** | **Grading 1-10** |
| Accuracy | Not high if you detect objects with only OpenCV but with YOLO it would be way more accurate | 5 |
| Speed | Fast for computer vision tasks, slower than TensorFlow/PyTorch for deep learning. | 7 |
| Community & Support | Large and active community with extensive resources and tutorials. | 10 |
| Training Time | Limited training capabilities for deep learning models. | 3 |
| Flexibility | Primarily for computer vision tasks, less flexible for general deep learning. | 6 |
| Ease of Use | Beginner-friendly with many tutorials and well-documented functions. | 9 |
| Resource Efficiency | Efficient for computer vision tasks, especially without deep learning models. | 9 |
| **Total** | | 49 |

# 

|  |  |  |
| --- | --- | --- |
| **Azure AI** | | |
| **Aspect** | **Reasoning** | **Grading 1-10** |
| **Accuracy** | Accuracy depends on the specific deep learning service used. | N/A |
| **Speed** | Speed varies depending on the service. | N/A |
| **Community & Support** | Microsoft provides documentation and support for Azure AI services. | 8 |
| **Training Time** | Training time depends on the service, dataset size, and hardware used. | N/A |
| **Flexibility** | Offers various deep learning services for different tasks, but may not be as flexible as standalone frameworks. | 8 |
| **Ease of Use** | Can be easier than building models from scratch, but requires familiarity with Azure services. | 8 |
| **Resource Efficiency** | Pay-as-you-go model for cloud resources, can be efficient if used optimally. | 8 |
| **Total** | | N/A |

# Research sub-question 2

**Sub-question 1:** How best to set up/focus the computer vision approach to make 'learning' a product as easy as possible (little resources, effort and time saving)?

## Library: Available product analysis Library: Expert interview Library: Best, good and bad practices

## 

## Conclusion

# Research sub-question 3

**Sub-question 2:** How can we make it easier for computer vision methods to detect products?

## Library: Community research Library: Expert interview Workshop: Prototyping

**Sub-question 3:** How can we integrate identification data with manipulation tasks (robotic or human) and later tasks in the disassembly line?

## Conclusion

# Final conclusion

Source list

|  |  |  |
| --- | --- | --- |
|  |  | **https://pytorch.org/** |
|  |  | https://www.tensorflow.org/ |
|  |  | https://opencv.org/ |
|  |  | https://docs.ultralytics.com/ |
|  |  | https://www.simplilearn.com/keras-vs-tensorflow-vs-pytorch-article |
|  |  | https://www.quora.com/What-is-the-difference-between-OpenCV-and-Tensorflow |
|  |  | https://opencv.org/blog/pytorch-vs-tensorflow/ |
|  |  | https://towardsdatascience.com/which-is-better-for-your-machine-learning-task-opencv-or-tensorflow-ed16403c5799 |
|  |  | https://stackshare.io/stackups/opencv-vs-pytorch#:~:text=OpenCV%20is%20an%20open%2Dsource,the%20key%20differences%20between%20them. |