PDEng THESIS REPORT

Feasibility and prototype of replacing commercial off-the-shelf pattern recognition solution

Raha Sadeghi
09/2021
Department of Mathematics & Computer Science

PDEng SOFTWARE TECHNOLOGY

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Feasibility and prototype of replacing commercial off-the-shelf pattern recognition solution

Raha Sadeghi

September 2021

Eindhoven University of Technology
Stan Ackermans Institute – Software Technology

PDEng Report: 2021/062

Confidentiality Status:
Public – Open Access

Partners	ASML Netherlands B.V.	Eindhoven University of Technology
Steering Group ***(By alphabetical order)***	Odysseas Papapetrou Zhifeng Sheng	
Date	September 2021	

Composition of the Thesis Evaluation Committee:

Chair:	Tim Willemse
Members:	Louise Gouteux
	Marc Geilen
	Odysseas Papapetrou
	Tim Willemse
	Zhifeng Sheng

The design that is described in this report has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct.

Contact Address
Eindhoven University of Technology
Department of Mathematics and Computer Science
MF 5.072, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
+31 402743908

Partnership
This project was supported by Eindhoven University of Technology and ASML.

Published by
Eindhoven University of Technology
Stan Ackermans Instituite

PDEng-report
2021/062

Preferred reference
Feasibility and prototype of replacing commercial off-the-shelf pattern recognition solution.
Eindhoven University of Technology, PDEng Technical Report 2021/062,
September 2021

Abstract
In the ASML holistic lithography, YieldStar is a metrology tool that provides closed-loop feedback to scanners by measuring on-product errors such as overlay and focus. YieldStar utilizes image pattern recognition techniques to measure the position shifts of wafer alignment marks. It uses this position information to build up a high-order wafer model to guide measure target positioning. YieldStar uses a Commercial-Off-The-Shelf (COTS) software library, which has expensive license costs and requires a hardware dongle, complicating machine build-up. In this project, a new pattern recognition library (RECOG) was designed and developed as a replacement for the COTS tool in the context of YieldStar wafer alignment. RECOG was developed by taking advantage of free open-source libraries, which could reduce the cost of build in YieldSatr. Additionally, there is no need for any hardware dongle. A new GUI application based on the RECOG library was developed to prototype the solution. Finally, RECOG was integrated into the YieldStar application to facilitate wafer alignment without using any hardware dongles. The results showed that RECOG could recognize patterns accurately with almost less than a pixel difference than the commercial library. The performance and flexibility of RECOG were also tested. According to the results, RECOG is flexible enough to recognize various patterns even in noisy and low contrast images in more than 90% of cases. By achieving a roughly 50-millisecond recognition time per pattern, it can align a wafer only 4 percent slower than the COTS tool, which is acceptable for the sake of feasibility study.

Keywords
Lithography process, wafer alignment, pattern recognition, YieldStar

Disclaimer Endorsement
Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the Eindhoven University of Technology or ASML. The views and opinions of authors expressed herein do

not necessarily state or reflect those of the Eindhoven University of Technology or ASML, and shall not be used for advertising or product endorsement purposes.

Disclaimer Liability

While every effort will be made to ensure that the information contained within this report is accurate and up to date, Eindhoven University of Technology makes no warranty, representation, or undertaking whether expressed or implied, nor does it assume any legal liability, whether direct or indirect, or responsibility for the accuracy, completeness, or usefulness of any information.

Trademarks

Product and company names mentioned herein may be trademarks and/or service marks of their respective owners. We use these names without any particular endorsement or with the intent to infringe the copyright of the respective owners.

Copyright

Foreword

Within ASML, besides making Lithography systems like Scanner, we also make metrology systems that allow measurement of on-product overlay and focus using diffraction-based technology and on specific OV/focus targets. To be able to position/measure the targets accurately, YieldStar, which is the ASML metrology system, uses image pattern recognition technique to measure position shifts of wafer alignment marks and use this position information to build up a high order wafer model to guide measure target positioning. However, YieldStar is using a COTS software library, which has expensive license costs and requires a hardware dongle which complicates the machine build-up.

Hence, I postpone an OOTI project to explore the possibility to replace the COTS solution with either an open-source/free software solution or an in-house solution (with the open-source being the preference.) The new solution should act as a "drop-in" replacement to the COTS solution in the sense of pattern recognition precision, flexibility, software interfacing, and performance.

I am very happy to have had Raha as the OOTI trainee to work on this project. She worked hard and cleverly to find a very good replacement to Cognex, implemented the toolbox in C#, and delivered the results to our YieldStar code archive. With experiments on the actual machines, she demonstrated the accuracy, flexibility, and performance of the new algorithm/toolbox. I am very impressed by the result she delivered and the way she could work independently in this unique period where most of us are working remotely. I am also very happy that Raha will join ASML soon. I wish her all the best in ASML and we will work together again soon.

Zhifeng Sheng

September 2021

Preface

This document is the final report of the “Feasibility of replacing YieldStar wafer alignment pattern recognition solution” project, executed by Raha Sadeghi, as the graduation project of the Professional Doctorate in Engineering (PDEng) program in Software Technology. PDEng is a two-year doctorate-level program provided by the Eindhoven University of Technology under the banner of 4TU.School for Technological Design, Stan Ackermans Institute.

The project was carried out in collaboration with ASML to investigate the feasibility of replacing the YieldStar pattern recognition library using open-source libraries. The project goal was achieved by designing and developing the new pattern recognition library and a new GUI tool for prototyping the solution.

This report begins with an explanation of the problem that the project targets. It is followed by a review of the literature and the results of prototyping various techniques to find the potential solution. The description of the proposed solution is followed by the architecture and design of the solution. Finally, the system verification and validation are explained, and the results are discussed. The readers who would like to understand the problem, domain, and requirement can check Chapters 3, 4, and 5 respectively. The ones who are interested in the details of the proposed solution, and the rationale behind the proposed solution, can review Chapters 6 and 7. Chapter 8 dives into the details of the logical view of the architecture. Therefore, readers interested in the system's architecture should review Chapters 6, 7, and 8. Some details on the implementation phase are given in Chapter 9. Those who want to examine the verification and validation results can read Chapter 10. We suggest reading Chapter 11 to the ones who would like to know more about the management of the project, the risks identified in the project, their mitigation strategies. Readers interested in the summary of the achievements in the project and suggestions for future work or the trainee’s self-reflection on the project could read Chapter 12.

Raha Sadeghi

September 2021

Acknowledgements

I would like to share my appreciation to those who played an important role in the successful completion of this project.

I am especially indebted to Zhifeng Sheng, my ASML supervisor, who helped and guided me continuously through the project. This project would not have been a success without your guidance, patience, and tremendous insight. I have learned a lot from you.

I am very grateful to my supervisor at the Eindhoven University of Technology, Odysseas Papapetrou, for his guidance during the project. His comments about this thesis and the process during the meeting were always insightful. I would like to express my sincere gratitude to Yanja Dajsuren (PDEng Software Technology Program Director) for her guidance, encouragement, and management approach during the last two years. I would also like to thank all my colleagues from the PDEng Software Technology program for being there for me during the entire duration of two years of the program.

I would not forget to remember Louise Gouteux, Mustafa Kabak, Julius Chatterjee, Erik Vermij, Kadir Luzumlar, Gilda Khosravi, Vinay Bansal, and Sjoerd van der Laag for your support throughout the project. Without your help, it would take much more time for me to get into this new domain. You were always there for me when I had a question or an issue.

Finally, I would like to offer my deepest gratitude to my family for their significant role in my personality and success. I cannot thank my mother enough for bearing the most challenging days of her life alone, sacrificing her wishes and needs, to support me in fulfilling my dreams and attaining my goals. Likewise, my father, who might not be any more physically in this world, but shaped me to be a strong woman who can stand on her own feet and always persuaded me to follow my dreams by his constant belief in me. My special thanks and gratitude goes to my brother, who devoted his time to compensate for the absence of my father and me in our family in the last two years. I would like to say a very big thank you to my mother and brother for their endless emotional support and the courage they gave me.

Raha Sadeghi
September 2021

Executive Summary

ASML is a leading lithography tool provider in the world. In the context of semiconductors, lithography is the process of printing highly complex circuit patterns on silicon wafers. YieldStar (YS) is an integrated or standalone metrology system that allows measurement of on-product overlay and focus errors, using diffraction-based technology on specific Overlay/focus targets. YS systems are usually integrated into the production line. Therefore, they can provide closed-loop feedback to the lithography system for real-time corrections to the manufacturing process. YS utilizes pattern recognition techniques to measure position shifts of wafer alignment marks. It uses this position information to build up a high-order wafer model to guide measure target positioning. YS is currently using a Commercial-Off-The-Shelf (COTS) software library. There are some concerns regarding using this library, which is as follows:

1. It has an expensive license cost, which indirectly impacts the machine cost of the build.
2. It requires a hardware dongle, which complicates the machine build-up.
3. Since it is a third-party solution, it is being used as a black box, and customizing the solution to address the edge cases is not easy.

Pattern recognition has different applications in YS. This project was conducted to explore the feasibility of finding a replacement for the COTS pattern recognition tool in the context of wafer alignment. The replacement should have comparable sub-pixel accuracy, performance, and flexibility with the COTS tool. The goal was not to improve the COTS tool result but mainly to create an in-house solution and make the recognition approach transparent and extensible. In order to achieve this objective, this project was divided into the following phases:

- Phase one: Creating a new pattern recognition library using available open-source computer vision libraries. This new library should have an interface compatible with the current COTS interface.
- Phase two: Creating a new alignment recognition tool to prototype the new solution and investigate its accuracy, performance, and flexibility.
- Phase three: Integrating the solution into the YS application and measuring wafers using the new library without utilizing COTS software and any hardware dongles.

This report presents the process and approaches taken to accomplish the phases mentioned above. RECOG, the new pattern recognition solution, exploits traditional computer vision and image processing techniques, mainly Template matching and Normalized Cross-Correlation. The library was validated against different patterns and images with different qualities and contrasts. Additionally, the library was used for aligning the wafer during the wafer measurement, and the results were compared with the results achieved by the COTS tool. Based on these results, we can conclude that RECOG can accurately recognize wafer alignment marks with sub-pixel accuracy and almost less than one-pixel difference than the COTS results. Although RECOG aligned wafers roughly 4% slower than the COTS solution, the alignment time was below the feasibility threshold set by the company (2 seconds.) Additionally, the test results showed that RECOG was almost as flexible and reliable as its commercial counterpart, even in noisy, low-contrast images. The main contribution of this project is the designed framework that can be extended to support new functionalities or achieve better performance and accuracy.

Table of Contents

List of Figures

List of Tables

1.Introduction

This document contains all theoretical and technical aspects of replacing the commercial off-the-shelf pattern recognition solution in the context of YieldStar wafer alignment. This chapter states a brief description of ASML, the initiator of this project. Then Lithography, YieldStar, the application of pattern recognition in YieldStar, and the project goal are briefly discussed. The outline section defines the roadmap of this report.

1.1 ASML

ASML is an innovation leader in the chip industry. Microchips are also known as integrated circuits, semiconductor chips, computer chips, or simply chips. These tiny pieces of silicon that are the basis of the digital world make our smartphones, cars, medical equipment, and so many other now-common devices possible. The tinier they get, the more advanced technology is needed to realize them in the everyday world. They may be small, but their impact is tremendous.

Lithography is a technology used by chip manufacturers, consisting of transferring a shrunk geometrical pattern of circuit lines from a photomask to a wafer by photomask illumination. ASML's lithography solutions have been making giant leaps on this tiny scale since 1984. ASML provides its customers with everything they need, including hardware, software, and services, to give them the power to mass produce patterns on silicon and chips.

ASML enables groundbreaking technology to solve some of society's toughest challenges. Together with their partners, they provide leading pattern solutions that drive the advancement of microchips. People at ASML design and manufacture lithography machines essential in chip or Integrated Circuits (ICs) manufacturing. ASML's customers are companies such as Intel or Samsung, who use its machines to create microchips that are eventually used in many electronic devices, including smartphones, laptops, and much more.

1.2 ASML holistic lithography

The process of producing ICs consists of many steps. Lithography is the heart of chip manufacturing. In the context of semiconductors, a lithography process can be defined as the process of exposing light on the wafer surface to create tiny structures on it. These small structures, in the end, form the ICs.

ASML delivered proven holistic lithography process control solutions to the industry 10 years ago to maximize patterning process performance and control. ASML holistic lithography involves computational lithography, metrology, and scanner control as an integrated solution to support the patterning roadmap. An example of a holistic control loop can be seen in Figure 1.

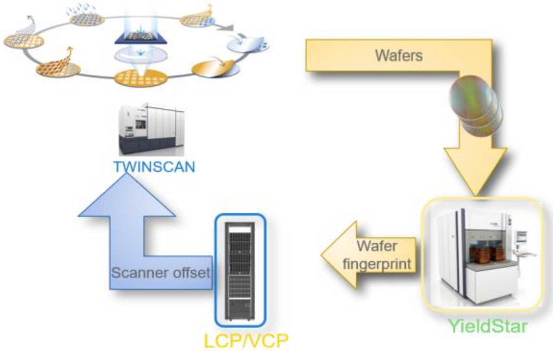


Figure 1 - ASML holistic lithography

ASML TWINSCAN machines enable the lithography procedure by exposing lights on the wafer and creating structures. For achieving massive chip production, these machines should be reliable, accurate, and have high throughput. Throughput defines the number of wafers that can be processed per time unit. Moreover, the quality of the produced wafers is paramount. In order to measure the quality of the produced wafer, there are some quality metrics in the context of lithography, which are as follows:

1. Overlay (OV), defining how well one layer is placed on top of the other layer.
2. Critical dimension (CD), defining how consistently the size of features can be reproduced.
3. Focus (F), defining how well the lines and features are described in the light-sensitive resist on the wafer.

These quality metrics define how well the lithography process in the TWINSCAN machine is performed. The quality metrics are measured with the help of the metrology tools, which will be discussed in the next section. The measurement results are passed to the Litho Computing Platform (LCP). ASML has developed several solutions for calculating required corrections by the scanners that are deployed on LCP. As Figure 1 shows the corrections should be passed to the TWINSCAN machine to scan wafers with higher performance and accuracy in the next round. This cycle of measuring wafers and calculating corrections for the next lithography step is called the holistic control and monitoring loop.

1.3 Context

There are two ways to examine the quality of the printed features on a chip: diffraction-based optical measurement and e-beam inspection. Diffraction examines how light reflects from the wafer, while e-beam observes how electrons scatter when they come into contact with the wafer [29].

ASML began developing its own diffraction-based metrology approach. The Diffraction Based Overlay (DBO) measurement principle uses a target that consists of two sets of gratings (consider gratings as some periodic parallel lines [32]) placed on top of each other. Depending on how two layers or targets are aligned on top of each other, the gratings generate a diffraction pattern. The diffraction pattern is captured by the measurement sensor. This approach makes measurements faster without compromising accuracy. YieldStar (YS) is the ASML diffraction-based metrology machine that enables measuring quality metrics such as OV, CD, and Focus [29].

YS systems are usually integrated into the production line to measure the quality of patterns on the wafer quickly and accurately, looping the data back to the lithography system for real-time corrections to the manufacturing process. Measuring quality metrics is only possible if YS knows the exact position of the structures on the wafer surface. Therefore, there are some steps before initiaing the actual measurement. One of the most critical stages is the wafer alignment.

Wafer alignment is based on is based on pattern recognition to find the marks'sposition on the wafer and measure the marks' deviation from their expected positions. Based on this data, a wafer model can be created, which can be used to update the wafer raw stage coordinate. The raw stage coordinates are the actual positions that define how the stage (the place where the wafer sits on) should move towards a target.

Computer vision aims to teach computers to process and interpret images or videos in the same way humans do. Pattern recognition, object detection, and object classification are some of the most important computer vision concepts. Pattern recognition, which is the functionality required in this work, is the automated recognition of patterns and regularities in data. It has applications in statistical data analysis, signal processing, image analysis, information retrieval, bioinformatics, data compression, and computer graphics.

Computer vision is not a new concept; it was introduced more than twenty years ago. While there are commercial and industrial-level computer vision tools in the market, some open-source libraries have also been introduced during these years. Some of these open-source libraries became comprehensive and mature enough to be used as a replacement for their commercial competitors. Users can take advantage of the already implemented algorithms of these libraries to create a customized solution addressing the problem. By using open-source solutions, error troubleshooting and tool extension are more feasible. Additionally, exploiting open-source libraries instead of the COTS software could help with cutting the company's cost. All in all, there is a significant added value in taking advantage of open source libraries instead of using commercial tools.

YS is currently taking advantage of the Cognex library (a commercial solution) as an image pattern recognition solution. Therefore, ASML initiated this project to investigate the feasibility of taking advantage of the currently available open-source libraries to replace Cognex. Raha Sadeghi conducted the project as her final design project for the Software Technology Professional Doctorate in Engineering (ST PDEng) program.

Cognex provides various functionalities and has applications in different parts of YS. This project studies the feasibility of replacing Cognex with an open-source library to address the pattern recognition issue only in the context of wafer alignment. The required Cognex functionalities that the new solution should support are discussed in Section 4.4.

1.4 Outline

This report consists of twelve chapters. In the next chapter, the stakeholder analysis is given. Chapter 3 describes the problem analysis. The fourth Chapter gives details about the project environment, relevant system components, and the project scope. Next, Chapter 5 lists the system requirements and use cases. Based on the acquired knowledge and the system requirements, we studied the literature and prototyped different techniques to define the project solution direction, which is discussed in Chapter 6. A pattern recognition solution is proposed in Chapter 7. More details about the architecture and design of the system are discussed in Chapter 8. Implementation details are discussed in the 9th chapter. Chapter 10 describes the approaches taken to verify and validate the system and reviews the results. Chapter 11 reviews the project timeline and risks that were identified in the project with their mitigation strategies. Chapter 12, the last chapter, reviews the bigger picture of this project's contribution from the company's perspective. The suggestions for future work are presented in the same chapter. Finally, the project's retrospective from the author's point of view is given.

2.Stakeholder Analysis

The individuals who are interested in or have an influence on a project are the stakeholders. Understanding their concerns and requirements and addressing them in the architecture and design phase is crucial to the project's success. This chapter discusses the interest, concerns, and involvement of each stakeholder involved in the project. The main stakeholders of this project are ASML Netherlands B.V. and the Eindhoven University of Technology. In the following sections, the names, roles, and concerns of main stakeholders are described.

2.1 ASML Netherlands

ASML Netherlands B.V. is the industrial party of this project. The outcome of this project could bring added value to the company. ASML stakeholders are responsible for providing the details and related domain knowledge of the wafer alignment. Table 1 lists the main ASML stakeholders. Their involvement level was quite different depending on their expertise and interest. Dr. Zhifeng Sheng, the project supervisor, closely monitored the project progress via weekly meetings. The rest of the stakeholders were involved in the project through necessary meetings and sprint review sessions.

Table 1. ASML stakeholders

Id	Name	Role	Concerns
1	Zhifeng Sheng	ASML Supervisor	A. Ensuring the system design satisfies stakeholders' concerns B. Helping the PDEng trainee to overcome a steep learning curve inside ASML C. Project success
2	Jurgen von Oerthel	ASML Group Leader	A. Project success B. Adherence to the company rules
3	Harald Vos	ASML PWD Department Manager	A. Project success B. Adherence to the company rules
4	Vinay Bansal Erik Vermij Mustafa Kabak Jean-Paul Mikkers Joost Verkooijen	Architects	A. Ensuring that the solution architecture adheres to the current YS software architecture B. Ensuring that the solution meets the required functionalities C. Ensuring that the new solution's performance and accuracy is comparable with the existing one D. Supporting both the new and the available pattern recognition solution
5	Louise Gouteux Julius Chatterjee Amandine Renault	Functional Designer	A. High performance (fast) pattern recognition solution B. Accurate pattern recognition solution (sub-pixel accuracy) C. Alignment profile creation D. Reliable pattern recognition solution (same result for the same input) E. Configurable pattern recognition solution

			F. No usage of any hardware dongle G. Ease of use H. Flexibility
6	ASML Customers such as Intel and Samsung	ASML Customers	A. Ease of use B. Fast lithography which implies fast wafer alignment C. Accurate results D. Reusability [both for recipes and patterns]

2.2 Eindhoven University of Technology

This section describes the main stakeholders of the Eindhoven University of Technology. They are mainly responsible for ensuring that the quality of the project deliverables meets the PDEng standards. The knowledge and experience of these stakeholders regarding the project business logic might be limited. However, they provide help and support from the academic point of view, if needed. Table 2 lists the main stakeholders and their concerns. Dr. Papapetrou was primarily involved in the project via PSG meetings. He got regularly updated either by emails or private meetings. Dr. Dajsuren occasionally participated in the PSG meetings and urgent situations meetings.

Table 2. Eindhoven University of Technology stakeholders

Id	Name	Role	Concerns
1	Yanja Dajsuren	TU/e PDEng ST Program Director	A. Ensuring that the quality of the project is following the PDEng program standards B. Trainee's graduation C. Project Success
2	Odysseas Papapetrou	TU/e Supervisor	D. Monitoring the trainee’s progress E. Helping the trainee to overcome potential difficulties by giving directions in case of need F. Evaluating the project achievements
3	Raha Sadeghi	TU/e PDEng Trainee	G. On-time graduation H. Project success I. Developing project management and soft skills J. Developing technical skills, including designing skills and computer vision K. Stakeholder expectation management

It is important to note that not all stakeholders have the same influence on and interest in the project. It is crucially important to figure out how each stakeholder could influence the project and its direction. During the project, the trainee tried to grab the attention of the stakeholders with less interest but high impact on the project by providing interesting results to increase their involvement to be able to request and get more specific information.

3.Problem Analysis

This chapter explains the project context at a very high level, helping identify the problem, followed by identifying the project goal.

3.1 Context

YieldStar is an integrated or standalone metrology system that performs quality measurements on specific wafer targets produced by the TWINSCAN machines. These quality measurements are executed in specific predefined regions, which are at known positions from a predefined basis. Once the wafer is placed in YS, the location and orientation of the wafer are unknown. As the field of view of the camera is small, for quality measurement, the wafer position has to be known up to $\pm 0.3\mu m$.

Figure 2 shows the life cycle of a wafer in YS. Once TWINSCAN exposes light on a wafer, it passes the wafer to the YS machine. Within the Wafer Exchange phase, a robot takes a wafer from the FOUP (a box of wafers exposed by the scanner) and prepares it for pre-alignment. The first pre-align step physically aligns the wafer up to $\pm 5\mu m$ based on the position of the wafer notch. In this phase, the wafer is rotated, and an edge sensor measures the edge of the wafer as it is rotating. From the edge measurements and notch position, a first wafer grid can be determined. This step is followed by a more accurate wafer alignment phase that identifies the wafer position with the required precision. After this step, the stage coordinate system is get updated, making the wafer measurement possible. During the measurement phase, the stage goes to the overlay targets based on the updated coordinates, and images can be taken from the targets to measure the quality metrics. As the focus of this project is on the wafer alignment phase, let’s dive a bit more into the wafer alignment phase.

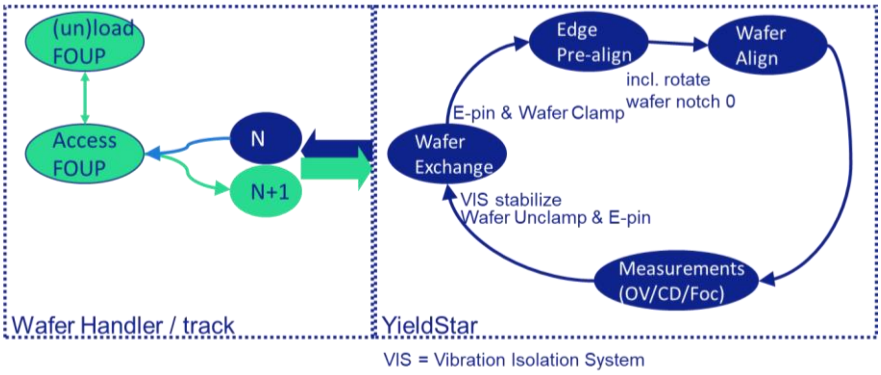


Figure 2. Wafer life cycle

Generally, wafer alignment includes two steps, namely Coarse alignment and Fine alignment. Depending on the alignment mode (fast or normal,) either both of them or only the Fine alignment are needed. While both Fine and Coarse alignments apply pattern recognition techniques to determine the position and orientation of the wafer, they target different fields on the wafer. The wafer surface consists of a plethora of structures. The main idea behind using the pattern recognition technique is to define some of these structures as marks and then try to recognize and precisely localize these marks on the wafer surface.

During the wafer alignment phase and based on the pattern recognition result, the difference between the position of the recognized mark on the wafer P_R (Real Position) and the expected mark position on the wafer P_E is measured. YS can build up a high-order wafer model to guide measure target positioning by extracting the position of enough marks and finding their shift based on their corresponding expected positions. Figure 3 illustrates the mentioned idea. After wafer alignment, YS can determine the exact position of structures on the wafer. Hence it can start the measurement phase to measure the mentioned quality metrics such as OV and provides feedback to TWINSCAN. The detail of how the measurement is being done in YS is out of this project's scope.

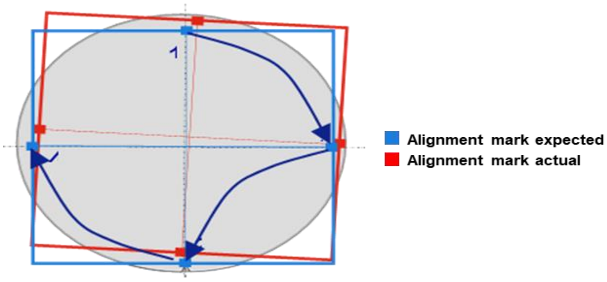


Figure 3. Wafer alignment concept

3.2 Problem Statement

ASML uses a commercial off-the-shelf library, called Cognex Vision Pro, for pattern recognition. Although Cognex works quite well, the main problem with using Cognex is that it is licensed, and it is not working without using a hardware dongle. Dongle cost is around 3,5k euro, which makes it expensive. Additionally, using the hardware dongle makes the configuration more complex. Using a hardware dongle also makes the development more challenging, as each developer or tester also needs to have a dongle. Different versions require different hardware dongles. Therefore, finding a replacement for Cognex, which is not licensed and does not require a hardware dongle, could be highly beneficial.

ASML Clients use a wide variety of shapes as patterns (markers) captured under various conditions. Cognex seems to deal well in recognizing these patterns in most cases, but sometimes it generates unexpected results, leading to false positives and false negatives. Incorrect results could lead to a non-optimal determination of the wafer coordinate system since wrong results are used for finding P_R. Note that inaccurate results mean that the wafer cannot be aligned correctly. Therefore, the overlay target cannot be located within the camera spot to be measured, which in turn could cause wafer measurement failure. Another practical case is that in a few cases, Cognex cannot recognize the patterns (mostly with very low contrast images or in case of patterns with shadows in the borders,) which is not desired without determining the reason. It is almost impossible to design and develop a solution that never produces a false positive or can always recognize all sorts of patterns. However, because Cognex is a third-party library and the algorithms being used within this library are patented and used as a black box, it is not easy to determine the source root of this behavior or customize the solution.

This project explores the possibility of finding a pattern recognition solution using open-source computer vision libraries as a replacement for Cognex in the context of the wafer alignment within YS. In this project, the main goal is to propose a new in-house and transparent solution that could make further improvements possible. Although the new solution should be comparable with Cognex in terms of accuracy and performance, some extent of deviation is acceptable, as generating the framework and investigating the feasibility of the replacement is the primary goal.

A prototype is required to test the accuracy and performance of the proposed approach. The new library should have an interface compatible with the current pattern recognition interface, such that it could be integrated within the YS application. Finally, the new library should be integrated into the YS solution such that YS can measure wafers without the Cognex dongle to evaluate the performance and accuracy of the new solution.

4.Domain Analysis

Once the wafer is located on the stage, it is prone to some extent of rotation, scale, or transformation. Therefore, YS expects a few micrometer differences between the expected pose of the targets and their actual positions. The process of finding the exact position of structures on the wafer and updating the stage coordinate system is called wafer alignment, which is the prerequisite of the wafer measurement. Pattern recognition is the heart of YS wafer alignment. This chapter discusses the domain model in more detail, and it is concluded by defining the project scope and constraints.

4.1 Pattern recognition application

Each wafer consists of multiple layers. Each wafer follows a complex manufacturing process with several steps per layer. Different photo-sensitive materials can be used to make it ready for exposure/lithography. During the exposure step, the circuit structure/pattern is printed on the wafer by exposing wafer to the light. Depending on the photoresist material being used, the appearance of the printed structure could look slightly different.

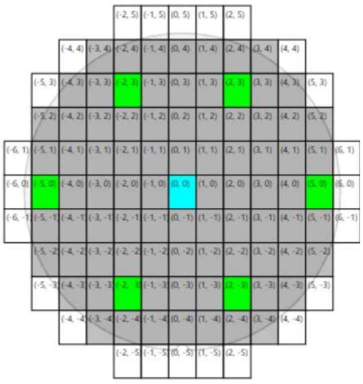


Figure 4. Wafer map of fields

Each silicon wafer has a round shape with a diameter of around 300 millimeters. One can consider a Wafer Coordinate System (WCS) for describing the positions on the wafer. The wafer surface can be divided into many fields (squares in Figure 4.) Based on the WCS, the center of the wafer is (0,0). X values increase to the right of the wafer surface, and Y values increase to the top of the wafer surface. Therefore, according to the WCS, a field located at the center of the wafer is at the position (0,0). The fields located at the center (the blue field) are called center/coarse fields. They are targets of the Coarse wafer alignment. The fields at the corners, such as the green ones, are called edge/fine fields, and they are targets of the Fine wafer alignment, which aligns the wafer more accurately.

4.1.1. Marks

Figure 5a represents a closer look at the content of a wafer field. Generally, the same set of structures is printed in all fields. The field structures can be simple shapes like regular rectangles or more complicated structures. There should be at least one unique structure or pattern among these structures, which we may call a mark or pattern. Figure 5b shows a unique structure (mark/pattern) that exists in the field image. These marks are the basis of pattern recognition in wafer alignment. Pattern recognition aims to define a mark to be recognized in the image and find its precise location.

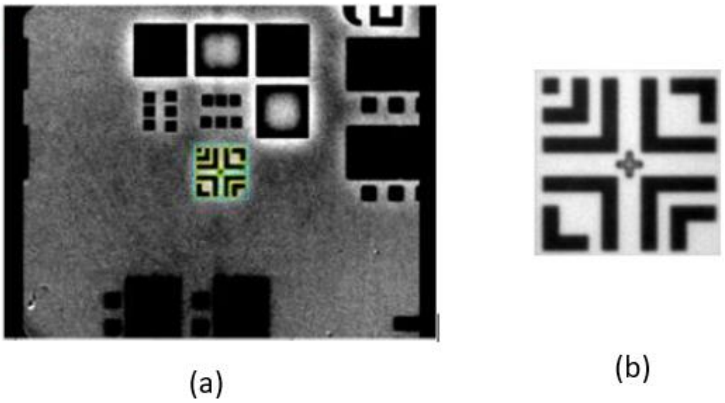


Figure 5. (a) Wafer field image with recognized pattern, (b) Pattern

4.1.2. Pattern recognition challenges

As mentioned in the previous section, different photo-sensitive materials used in the lithography procedure could cause the same pattern to look slightly distinctive in different layers, especially regarding their color intensity. Although they might have the same shape, the color of the mark shapes might be different. Hence, the proposed pattern recognition should not be dependent on the image's light intensity. Additionally, although the users try to find unique structures in the field as a pattern, still different structures in each field might share quite similar features, which could make the recognition challenging.

Images taken from the wafer fields are further processed for corrections and enhancements. However, images could have entirely different qualities or contrasts depending on the field location and other factors, such as various lenses, dust, or hardware imperfections. Additionally, images are subject to distortions such as rotations or translations due to the camera alignment. Noise can also exist, which makes recognition more challenging. Therefore, the solution should be flexible to recognize patterns despite all these difficulties. It is worth mentioning that the goal is not to recognize the pattern but to localize it. It means that the precise location of the origin of the mark should be reported.

Achieving high performance and accuracy is essential and, at the same time, challenging. Since pattern recognition is the heart of the YS alignment, any latency in this phase could lead to latency and overhead in the metrology procedure. Similarly, if the recognition cannot find the accurate position, it leads to inaccurate measurement and thus inaccurate feedback to TWINSCAN.

4.2 Cognex Vision Pro and APT

In the Problem Analysis chapter, Cognex was introduced. Cognex is a commercial tool that supports a variety of computer vision functionalities at an industrial level. It has a pattern recognition library called Cognex Vision Pro. Different versions of Cognex are being supported in the YS application.

Alignment Pattern Tool (APT) is an ASML-built tool that acts as a wrapper around some Cognex components. It also provides a graphical tool to facilitates pattern recognition. The tool supports various functionalities, from pattern creation to pattern recognition. It is a sandbox tool that allows users to play around with mark recognition, training the pattern, and fine-tuning the recognition parameters. As soon as one comes up with a set of settings (the best training and run-time parameters) that works accurately, it is possible to dump the tool’s settings and data and create an alignment profile. The alignment profile can further be used for creating an alignment recipe. The recipe definition is discussed in the next section in more detail.

Once the pattern is defined, training and localization can be done using patented algorithms such as PatMax and PatQuick. Cognex supports two different pattern types, an image-based pattern, and a model-based pattern.

In an image-based pattern, the user can grab a region of interest of the input image. Therefore, the existence of the wafer image is necessary for image-based pattern creation. On the other hand, for creating model-based patterns, users should know the exact geometric shapes of the model and lines' length in micrometer scale, but there is no need for any input image.

Figure 6 depicts a model-based and an image-based pattern representing the same structure. As it shows, the edges in the model-based one are sharper and more accurate. Using image-based patterns, edges should be detected automatically, which could be error-prone depending on the image quality. However, creating an image-based pattern is easier and does not require expert knowledge. Notably, the results gained by utilizing model-based patterns are more accurate, mainly because the train origin (usually it's center) can be defined more accurately, and the exact shape of the pattern is defined precisely. Hence, model-based patterns are used more widely.

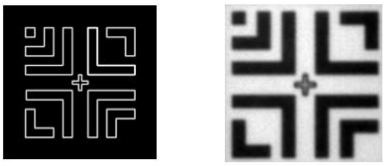


Figure 6. Model-based (left-side image) vs image-based pattern (right-side image)

Cognex can find patterns in an image of size 1000 by 1000 pixels in less than 30 milliseconds with subpixel accuracy. It finds a set of coarse and fine features during the training phase. During the execution phase, it finds the area in the image that has the highest similarity with the pattern, based on extracted features in the training phase.

4.3 YieldStar Wafer Alignment Module

After clarifying the application of pattern recognition, this section describes the general YS wafer alignment procedure, depicted in Figure 7.

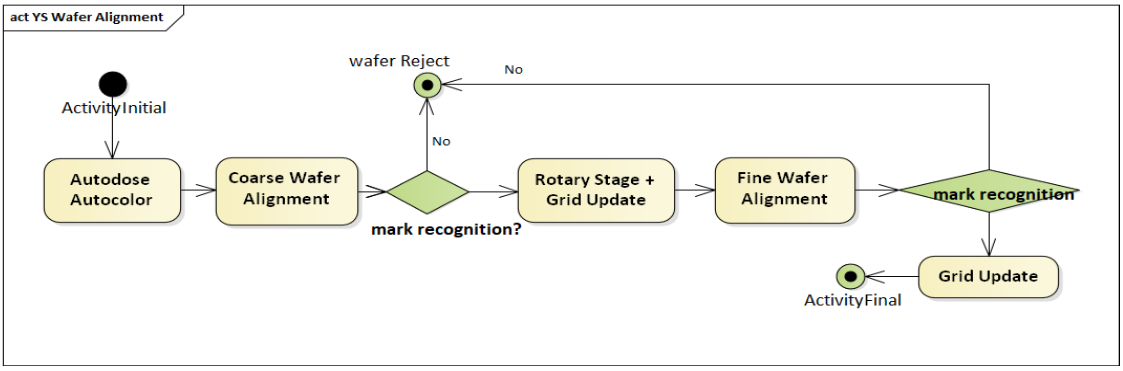


Figure 7. YieldStar wafer alignment procedure

The first step is to define the proper dose and color. The dose is the intensity level of light that shines on the wafer for the wafer alignment purpose. This step aims to find the best dose that could facilitate pattern recognition. Once the proper dose is identified, the machine can initiate the Coarse wafer alignment. Based on the result of this stage, the stage is aligned. The only reason to do Coarse alignment is to be accurate enough to find the fine marks. The Fine alignment starts only with successful pattern recognition in the Coarse phase, unless in the fast alignment mode which the Coarse alignment is not needed. Please refer to Appendix. A to find a summary of the wafer alignment domain model in one figure.

4.3.1. *Pattern recognition wrapper*

The application of pattern recognition in YS is not limited to the wafer alignment, but also it has application in the target alignment and other YS modules. A pattern recognition wrapper was introduced to increase the YS code modularity and decouple the pattern recognition solution from other modules. While the wrapper's implementation could vary depending on the pattern recognition library, all other modules and clients consuming the wrapper interfaces are decoupled from the library.

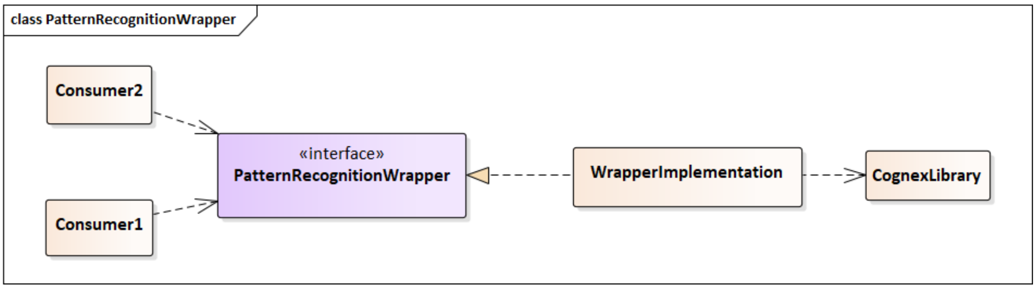


Figure 8. Pattern recognition wrapper

Figure 8 demonstrates the mentioned idea. So far, three different versions of the Cognex Vision Pro have been released, which are supported in YS. There is a separate wrapper implementation for each version. As discussed earlier, Cognex is licensed, and it can be used only if the hardware dongle is present. Since not everyone can have a hardware dongle (as it is costly), a simulation module was introduced that in the same way implements the wrapper interfaces. During the runtime, the proper wrapper implementation is used based on the installed version of the library.

4.3.2. *Alignment Recipe*

As mentioned earlier, all wafer fields contain the same set of structures. Also, despite of different photosensitive materials that can be used in different layers, patterns look almost identical in all layers with only slight differences. Additionally, customers produce a plethora of similar wafers. Therefore, the YS machines must create a recipe that can be repetitively used in massive production and across different wafer layers. Recipes define the instructions required for measuring a wafer.

There are various recipe types. Recipes contain different information, such as the selected wafer fields, the mark or target position, and the alignment profiles. Alignment profiles (.align files) contain the pattern description, train, run parameters, and other information such as the alignment mode. They can be created using APT.

Recipes can be image-based, model-based, or library-based. Image-based recipes are based on image-based patterns, and they can be created from scratch from the YS wafer alignment menu. Model-based recipes contain model-based patterns. Library-based recipes are recipes that contain more than one pattern. These patterns can be either model-based or image-based, all representing the same structure.

4.4 Project Scope

As mentioned earlier, this project targets only pattern recognition in the context of YS wafer alignment. APT and its usage in the YS wafer alignment were discussed in the previous sections. Since a tool, similar to APT, is a prerequisite for the wafer alignment and can be used to test the proposed pattern recognition solution, one of the initial goals of this project was to design and implement such a tool.

While APT supports both image-based and model-based pattern recognition, the target of this project was only model-based pattern recognition. The reasons behind this decision were as follows:

1. Supporting both types of pattern recognition within the limited time of the project was not feasible.
2. Model-based pattern recognition is being used more widely, and its result is more reliable than image-based recognition.
3. It is believed that image-based pattern recognition can be addressed based on the model-based techniques, provided that one can precisely detect pattern edges and contours. Section 7.1.1. gives more details on this topic.

Although Cognex and more precisely APT supports some additional functionalities such as model-based pattern creation, in this project, we mainly focused on addressing the most critical questions and features that requires more attention and investigation. Therefore, as we knew model-based pattern creation is doable and only requires some more effort in the implementation phase, we decided to consider adding this feature to the new tool as a feature work. Instead we added a new and more critical feature to the tool. Using this feature users can convert Cognex-based patterns to a new format, readable by the new library and tool.

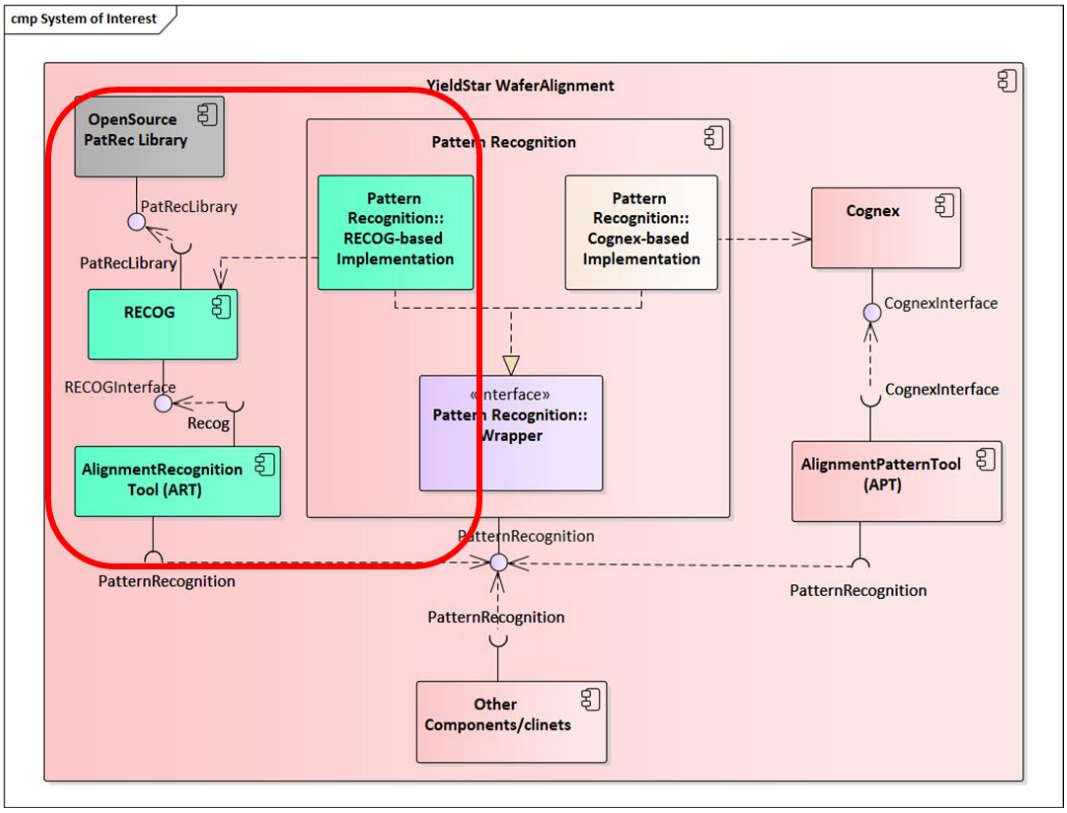


Figure 9. System of interest

The new library should be integrated into the YS application to assess the feasibility of replacing Cognex and measuring the new library's performance. This means that the pattern recognition wrapper should be implemented using the new library. Figure 9 shows a closer look at the pattern recognition module of the YS wafer alignment. While the right side of the figure shows the system's architecture before this project, the left side depicts the system of interest of this project and how the system should get updated in a high level. The main deliverables of this project can be summarized as follows:

- A new library based on the available open-source computer vision libraries, hereafter RECOG

- A new alignment pattern recognition tool based on the new library for model-based pattern recognition called Alignment Recognition Tool (ART)
- A new YS patch supporting both RECOG and Cognex libraries for the wafer alignment pattern recognition

4.5 Project Constraints

Identifying the project constraints plays a vital role in the system architecture and finding the proper solution direction. Essential project constraints are as follows:

- Utilization of any hardware dongles is not desired.
- The system shall only utilize open source libraries approved by the ASML Free and Open Source Software (FOSS) portal to be used in commercial product integration.
- The PDEng project should be finished by October 2021.

5.Requirements and Use Cases

This chapter reviews the system requirements and use cases. Requirements are the specifications that the deliverables (the new library, new tool, and the integrated YS) need to fulfill, and they were derived from the stakeholder's concerns. Use cases describe the usage of the system from the point of view of different actors. The requirements and use cases were discussed and agreed on with the stakeholders before designing the system.

5.1 Requirements Elicitation

The following sections describe the requirements of the project based on the MoSCoW method [31]. We categorized the requirements as business, functional, non-functional, architectural, and implementation requirements.

5.1.1. Business Requirements

This section lists the high-level statements of the goals, objectives, and requirements that should be met. The business requirements of this project are listed in Table 3.

Table 3. Business requirements

Id	Requirement Description	Priority
B01	Alignment Recognition Tool (ART), an APT replacement, shall facilitate pattern recognition without using Cognex or any hardware dongle.	Must
B02	YieldStar shall align wafers based on the model-based patterns without using Cognex in the wafer alignment stage.	Must

5.1.2. Functional Requirements

Functional requirements are detailed statements of capabilities, behavior, and information that the solution should address. Table 4 lists the functional requirements of this project.

Table 4. Functional requirements

Id	Requirement Description	Related concern	Priority
F01	The ART user interface shall allow the user to load the input image in either PNG or IM format. IM is an ASML internal image format.	5.G, 6.A	Must
F02	The ART user interface shall allow the user to load and import model-based patterns.	5.G, 6.A, 6.D	Must
F03	The ART user interface shall allow the user to set train parameters, including the train region and origin.	5.G, 6.A, 6.D	Must
F04	The ART user interface shall allow the user to train the model.	4.B, 5.H	Must
F05	The ART user interface shall present image metadata. Image metadata contains additional information about the image, including its size, location, resolution, and so forth.	5.G, 6.A	Should

F06	The ART user interface shall allow the user to set runtime parameters, including accept threshold, scale, and angle.	4.B, 5.E, 5.G, 6.A	Must
F07	The ART user interface shall allow the user to run a model-based pattern recognition and get the recognition result.	4.B, 5.H	Must
F08	The ART user interface shall show the recognition result in a table with information including X, Y offset (micrometer), score, angle, and the scaling factor.	5.B	Must
F09	The ART user interface shall enable the user to dump and load the pattern recognition solution (tool) either as an XML file or as an alignment profile.	5.G, 6.A, 6.D, 5.H, 4.B	Must
F10	ART shall facilitate pattern recognition using a specified rotation or scale.	4.B	Must
F11	ART shall run pattern recognition in batch for a set of images against a specific model-based pattern.	5.G, 6.A	Should
F12	ART shall save the result of batch execution in a CSV file.	5.G, 6.A	Should
F13	YieldStar shall allow users to align a wafer using either RECOG, without dongle, or using Cognex. *Description:* Integrating RECOG is required for the testing purpose and as a proof of concept, not as a production-level delivery.	4.B, 6.A	Should
F14	YieldStar shall record diagnostic data based on the result of the RECOG pattern recognition. *Description:* Diagnostic records are needed for the testing and result investigation.	4.B, 4.G	Should

5.1.3. Non-Functional Requirements

Table 5 lists the requirements that specify criteria used to judge the system's operation rather than specific functionality.

Table 5. Non-functional requirements

Id	**Requirement Description**	**Related Concern**	**Priority**
NF1	The system shall recognize each mark in less than 50 milliseconds.	4.C, 5.A, 6.B	Should
NF2	The system shall recognize each mark in less than 250 milliseconds. *Description:* The recognition time of the prototype of the system shall be less than 250 milliseconds. This recognition time is promising enough to achieve the desired performance in the production-ready application.	4.C, 5.A, 6.B	Must
NF3	The system accuracy (pattern recognition accuracy) shall be at most 0.35 micrometer.	5.B, 6.C	Should

NF4	The system accuracy (Pattern recognition accuracy) shall be at most 0.8 μm. *Description:* The pattern recognition module's accuracy shall be less than 0.8 μm. This accuracy is promising enough to achieve the desired accuracy in the production-ready application.	5.B, 6.C	Must
NF5	RECOG recognition success shall have at least 90 percent of the Cognex recognition success given the same image and pattern. *Description*: The system shall recognize patterns regardless of the image brightness or quality, as good as Cognex's capability to recognize the same patterns. The result is promising by achieving 90 percent success and it could be improved later on.	5.G, 6.A	Must
NF6	The system shall only utilize open source libraries approved by the ASML Free and Open Source Software (FOSS) portal to be used in commercial product integration.	1, 2, 3, 4	Must
NF7	The pattern recognition algorithm shall be deterministic. *Description*: The system shall report the same result given the same inputs.	5.D	Must

5.1.4. Architectural Requirement

Table 6 lists the architecturally significant requirements that should be taken into account while defining the system architecture.

Table 6. Architectural requirements

Id	**Requirement Description**	**Related Concern**	**Priority**
A1	For integrating RECOG into YieldStar, RECOG shall comply with the current architecture and interfaces of the YieldStar Pattern Recognition wrapper.	4. A	Must
A2	The system shall have a module for converting existing model-based patterns/xml/.align files to their corresponding file acceptable by the ART.	4.D	Should

5.1.5. Implementation Requirement

Table 7 describes the requirements that define implementation constraints.

Table 7. Implementation requirements

Id	**Requirement Description**	**Related Concern**	**Priority**
I01	RECOG shall have a C# interface.	4.A	Must
I02	RECOG shall be compatible with .Net framework 4.8 and C# version 7.3.	4.A	Must

5.2 Use cases

In this section, the most important use cases of the system are discussed. Use cases represent actors' interaction with the system. Defining use cases helps to figure out system requirements more precisely. Additionally, by satisfying related requirements, one can be ensured that the system corresponds to the user expectations. Figure 10 depicts the use case diagram of the system.

This system has two primary categories of actors. ASML customers and functional designers seem to have similar interactions with the system from this project’s perspective. Another main actor of the wafer alignment is the YS machine.

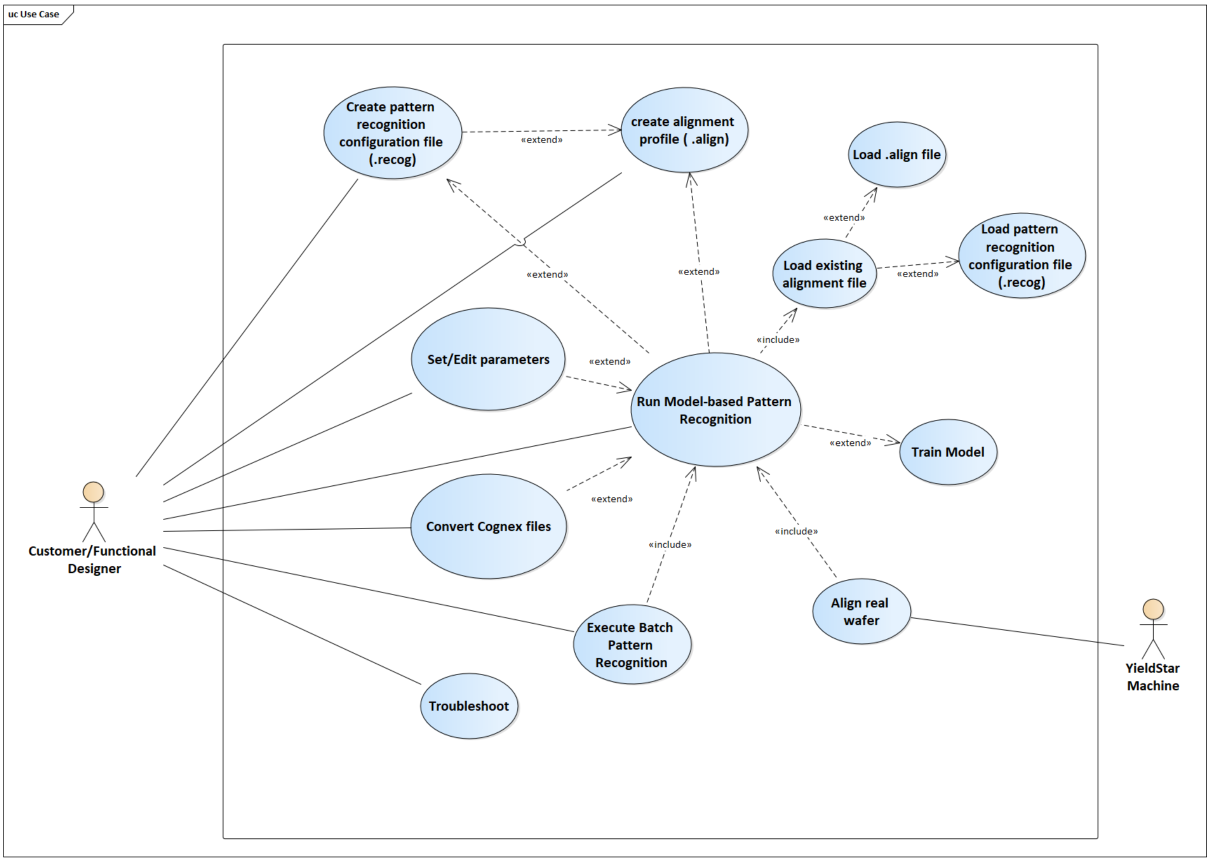

uc Use Case
Create pattern recognition configuration file (.recog)
create alignment profile (.align)
Load .align file
Load existing alignment file
Load pattern recognition configuration file (.recog)
Set/Edit parameters
Run Model-based Pattern Recognition
Train Model
Convert Cognex files
Execute Batch Pattern Recognition
Align real wafer
Troubleshoot
Customer/Functional Designer
YieldStar Machine
«extend»
«include»

Figure 10. Use case diagram

YS should be to able to align a wafer and measure the quality metrics such as OV with or without using Cognex libraries. Therefore, YS should be able to decide which library should be used and also retrieve the data from alignment profiles, regardless of the pattern recognition library that is used. Finally, YS should recognize patterns or marks to align the wafer.

Customers and functional designers need to set or edit training or run-time parameters, depending on the pattern template, to get better results. As mentioned earlier, the reusability of the alignment profiles, recipes, and patterns is one of the major concerns. Therefore, the system (new tool) shall enable users to capture pattern recognition solutions and create alignment profiles for later use.

Additionally, the system should enable functional designers or customers to convert the Cognex files into the RECOG ones, such that they can take advantage of the currently available recipes. The system shall enable users to load the available RECOG-based files and execute the pattern recognition using an alignment file, irrespective of its format.

Before executing the actual wafer alignment in the YS machine, users should run pattern recognition and find the suitable parameters facilitating the successful wafer alignment. Notably, they need to run batch pattern recognition for many input images to ensure that the parameters are set correctly. The system shall help with troubleshooting by creating recognition reports in the format of CSV, similar to Cognex, to help users finding the potential problem and fine-tune the parameters. Once the recipe is created, YS can run pattern recognition during the Coarse or Fine wafer alignment procedure.

5.3 Design Criteria

This section describes the design criteria for the proposed system. These criteria are revisited in 12.1 to verify how well the proposed solution satisfies the intended design criteria.

5.3.1. Performance

Wafer alignment plays a vital role in metrology and consequently in lithography. Any latency in the pattern recognition could dramatically impact the wafer alignment and consequently the machine throughput.

Accuracy is a big concern for a metrology tool. YS should provide feedback to the TWINSCAN machine to calibrate or correct the scanner for the next batch of wafers. Any errors, even a few nanometers, could have significant impacts on the lithography procedure. Therefore, achieving high accuracy and performance should be one of the leading design criteria.

5.3.2. Realizability

Realization and implementation of the designed approach must be achievable. The solution should be integrated into YS and tested in a production environment. In this way, based on the measured KPI, the feasibility of replacing Cognex with the new library can be investigated.

5.3.3. Extensibility

Pattern recognition has different applications in YS, and its usage is not confined to the wafer alignment. Besides, due to the project's time limit, we confined the project scope, and not all different scenarios are covered in this project. Therefore, the solution should be flexible enough to add a new feature or algorithm with minimum effort. It also implies that the maintainability of the solution is highly important.

6. Related Work and Feasibility Study

The feasibility of replacing Cognex is highly intertwined with the successful recognition and localization of patterns in images, which is referred to as object detection in the literature [1, 2, 3]. Object detection is not a new topic, and it has gone through two historical periods: “traditional object detection period” and “deep learning based detection period (after 2014)”[1]. However, note that it does not mean that the traditional Computer Vision (CV) techniques have become obsolete [3].

Traditional CV techniques commonly extract rich features from images and match extracted features with the input image [1]. On the other hand, state-of-the-art CV algorithms employ modern techniques (machine learning or deep learning algorithms) to detect objects [2]. The first question in this project was whether we should utilize the modern approaches or the traditional ones to address the problem. The first section of this chapter explains why we did not apply modern approaches. The next question was which kind of traditional techniques could be used in this project. To find an answer to this critical question, we reviewed the literature. We prototyped the potential solutions using Python, regardless of the final opensource library that we might select. We utilized the available wafer images used for troubleshooting to investigate the applicability of different techniques. The results of this study set the pillars of the proposed solution. The second section of this chapter explains the achieved results.

6.1 Modern Computer Vision Techniques

Many studies have been conducted during recent years, and promising deep learning algorithms have been introduced targeting object detection, like RNN, Fast-RNN, and YOLO [2, 3]. They all follow the same procedure. There should be a large dataset. A significant proportion of the data should be used for the training phase to capture the unique pattern's features and learn how to detect the pattern in unseen images. The remaining data can be used for algorithm validation and parameter fine-tuning. Once the pattern is trained, it can facilitate object (pattern) detection in images or videos.

Techniques based on deep learning have provided new opportunities for object detection in recent years. However, there are still applications that traditional approaches could generate good results [2]. DL could sometimes be an overkill if traditional CV techniques can solve a problem effectively, in fewer lines of code, and less complexity than DL [3]. Therefore, depending on the problem and its complexity, one should decide if using modern CV techniques is really needed or traditional techniques could address it.

Usually, large datasets are used for the training and testing of AI-based algorithms [2,3], mainly because the more training data available during the training phase, the more accurate result can be achieved. However, looking at this project’s problem specifications, there is not enough training data, if there is any. Customers could define their own set of alignment patterns specific to each product. Due to confidentiality concerns, they might not be willing to share even one of the pattern images. Not having a proper dataset implies that the essential requirement of using modern approaches cannot be met. Even if we have access to a pattern image, there is not enough data that describes the pattern with its precise location, especially in newly introduced patterns. Once we have enough annotated data for a pattern, the pattern can be trained once at ASML and be used by the customer who use the pattern. Though, note that for each product, a new alignment pattern might be required. For training the new pattern, a new set of annotated data should be available or generated, which requires an extra effort. The frequency of introducing a new product/pattern could vary depending on the company and customer preferences.

One possible solution could be to use a detector, which is pre-trained on large-scale datasets, and later fine-tune it for specific detection tasks. There are some limitations when adopting the pre-trained networks in object detection, such as the domain mismatch or differences between the category of datasets, their loss function, scale, and other differences. Because of differences, it is not always possible to transfer the pertained knowledge to the detection task as desired [2]. Note that the category of objects we need to recognize is unique.

The accuracy and precision of the pattern recognition in the wafer alignment are crucial. The algorithm should recognize the pattern with the exact described size and orientation. Modern techniques aim to find a pattern regardless of its size and orientation. Although recognition regardless of the size or orientation might sound an interesting feature, it is opposed to this project requirement. Therefore, an additional step for the refinement of results is required to exclude undesired results, leading to additional time overhead.

YOLO[1] is one of the fastest algorithms proposed so far, suitable for real-time applications, while Faster RCNN is slightly more accurate. There is always a tradeoff between gaining accuracy and performance [30]. Primarily these algorithms are not designed to be highly accurate in localization but mainly in detection or classification, and some degrees of errors in the bounding box coordinates are acceptable. However, accuracy is the main concern in this context, and even a few micrometers matters. According to the mentioned justifications, limited project time, and the need for a learning and investigation phase, as well as the potential need for data augmentation, we concluded that modern CV algorithms might not be the best option for solving the current problem in the first place. If we could address the problem using the traditional approaches, then as future work, one can apply modern techniques in addition to the traditional approaches to gain better results.

6.2 Traditional object detection techniques

Traditional object detection techniques take advantage of common CV techniques. Image registration or image alignment, which is used in computer vision, transforms different images of one scene into the same coordinate system. These images can be taken at different times (multi-temporal registration), by different sensors (multi-modal registration), and from different viewpoints [15]. Image alignment has a wide variety of applications. It is widespread in the field of medical imagery, satellite image analysis, and optical flow. Since we expect to find the template pattern in the input image and shift its location into the input image coordinate system, the problem can also be considered as an image (mark) alignment problem.

Image alignment algorithms can be categorized into two categories of intensity-based or feature-based approaches [10]. Intensity-based methods compare image intensity patterns via correlation metrics, while feature-based methods find correspondence between image features such as points, lines, and contours. It is also possible to utilize a combination of these two approaches.

Different techniques were discovered and prototyped in this project to investigate if they could fit in the current problem. The following sections summarize the most relevant approaches with their advantages and disadvantages in practice.

[1] You Only Look Once

6.2.1. *Feature-based matching*

Feature-based approaches follow three main steps: key point detection and feature description, feature matching, and image warping. Briefly, points of interest (key points) in both images should be selected, and they should be described in a similar way to be comparable with each other. Each key point in the reference image should be associated with its equivalent in the sensed image during the feature matching phase. A geometrical transformation can be calculated by mapping enough key points in images Homography matrix) to map the target image to the reference images. The matrix establishes a point-by-point correspondence between the reference and target images. Figure 11 illustrates the idea.

Different algorithms exist targeting different steps (key point extraction, description, and matching.) SIFT[2][5] and SURF[3] [6] are two of the best and most promising algorithms for feature-based matching. However, both are patented. ORB[4] [7] is an efficient alternative to SIFT and SURF, and it is not patented.

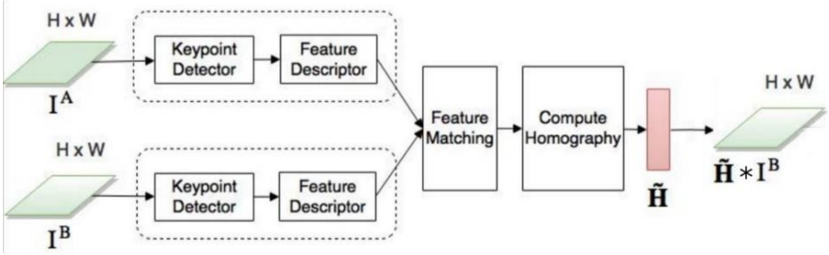


Figure 11. Feature-based matching technique

This technique is efficient if the image has a large resolution since it is based on feature extraction, and there is no need to convolute the image. Also, by utilizing the homography matrix, the sub-pixel position of each point in the template (pattern) image can be detected in the input image. However, this technique is not good if different objects share the same features or if images have fewer features [8]. Moreover, the accuracy of the homography matrix is highly dependent on how good key points are matched. Since wafer patterns are fairly similar (not too many distinctive features), after prototyping, we did not get good results, and in many cases, we observed keypoint mismatching. Therefore, this method failed.

6.2.2. *Template Matching*

Template Matching is a method for searching and finding the location of a template image in a larger image. It is an intensity-based image alignment method and one of the common approaches for object detection. It slides the template image over the input image (as in 2D convolution) and compares the template and patch of the input image under the template image [9]. There are different comparison methods, including Normalized Cross-Correlation (NCC) or square difference. These methods aim to find the similarity/dissimilarity score between two images based on some statistical techniques. The details of each approach are out of the scope of this document.

The main advantage of this technique is that there is no need for any annotated data compared to modern techniques. It is a simple method, requires fewer parameters (compared to the feature-based approaches), and gives a similarity score that can be used as a pattern recognition score. Additionally, this

[2] Scale-Invariant Feature Transform

[3] Speed Up and Robust Feature

[4] Oriented FAST and Rotated BRIEF

approach functions very well when templates have no strong features with an image, contrary to the feature-based approaches [8].

On the other hand, it is slightly slower with larger images. Upon object detection, its upper left position can be detected using template matching, but on the pixel scale. It implies that for finding the object's sub-pixel position, some further steps are still required. As mentioned earlier, this technique is intensity-based. Therefore, it only works well if two images have almost the same intensity. As mentioned in the domain analysis chapter, patterns of the same structure could have different intensities in different layers. Therefore, we could not get good results from this technique without any preprocessing and customization.

6.2.3. Contour-based matching

Contours are a set of points creating a closed shape. Contour detection is one of the image processing techniques which finds the contours of each structure in the image. The approach is to find the contours of both the input image and the template and then try to match the contours of the template to the input image contours based on their area size similarity. This approach aims to extract a set of customized and relevant features (contours) that could ease the detection. The good point about this technique is that it is not dependent on the image intensity, and it does not require setting or fine-tuning many parameters. However, this approach is not practical if the pattern contains a set of detached shapes or if there is more than one contour in the input image with the same size as the template pattern contour area.

While none of these techniques could completely address the project concern, the template matching technique seemed more promising if we could customize it. As it is mentioned, a common technique is to use a combination of the image processing techniques to come up with the best solution. The following sections review some other techniques that could ease the recognition.

6.3 Techniques for getting sub-pixel accuracy

Referring to the system requirement, we needed to find the pattern origin with sub-pixel accuracy. Most of the available techniques, such as template matching, report the pattern location with pixel accuracy. Therefore, additional steps were required to get the sub-pixel position.

The first option was utilizing feature matching algorithms such as ORB. The idea was to grab the recognized area with some additional margin and then apply ORB to find the homography matrix. Each point in the template image (pattern) could be easily mapped to the input image using the homography matrix, including pattern origin. However, due to the nature of the pattern structures of this project and their similarities, not enough good features could be extracted. Inaccurate features led to inaccurate results.

Phase Correlation is an approach to estimate the relative translative offset between two similar images. It is commonly used in image alignment and relies on a frequency-domain representation of the data, usually calculated by fast Fourier transforms [23]. This approach seemed promising because of its high performance. Besides, it generates a confidence score that could contribute in false positives detection. However, in practice, its result is prone to high error rates in noisy images [12]. With noisy images, it is hard to find the pattern edges. Inaccurate edges lead to incorrectly calculated shift results.

The sub-pixel interpolation technique is a way to get the sub-pixel value by an interpolation algorithm. Pattern recognition is done on the interpolated input and template images. The result is then substituted into the correlation expression to calculate the sub-pixel position. This technique is a key way to improve the displacement measurement accuracy in the correlation method; however, it is more time-consuming [4].

We applied the phase correlation method to many different images with different qualities and compared the results of this approach with the interpolation results. Results showed that the interpolation technique is more reliable in practice by having a lower error rate. However, it is not as accurate as phase correlation (in the case of correct detection.) Therefore, we proposed using interpolation to find sub-pixel accuracy, accepting a little more time overhead and the risk of not getting the exact sub-pixel position.

6.4 Image contrast adjustment techniques

The quality and contrast of images could be different for various reasons, such as hardware imperfections. One of the merits of Cognex that makes it stand out is its flexibility in recognition of patterns irrespective of the image quality. In low-contrast images with high noises, contours and edges cannot be adequately detected. Inaccurate feature extraction leads to pattern recognition failure. Therefore, one of the most significant prerequisites for attaining the project goal was adjusting image contrast and improving its quality. We prototyped different approaches to find the best technique.

The Histogram Equalization (HE) process is an image processing method to adjust the contrast of an image by modifying the image's histogram. HE improves the contrast of the image at the expense of boosting the contrast image noises. On the contrary, the Adaptive Histogram Equalization (AHE) technique divides an input image into an M x N grid. It then applies equalization to each cell in the grid, resulting in a higher quality output image without boosting noise as much as HE. The downside is that AHE is more computationally expensive than HE.

An alternative technique was histogram matching. In this technique, a reference image is required to get its histogram and then apply the input image's histogram to the target image. The problem with this technique was the constant need for a reference image. Moreover, a reference image does not work for all target images.

Finally, a promising method of Balanced Contrast Enhancement Technique (BCET) was proposed in [13] to enhance the MRI image contrast. As MRI images also could have relatively low contrast, and the application was quite similar (finding edges), this method was tested. Experiments showed that although this method was slightly more time-consuming than the AHE, the results were increasingly better, made it a potential solution.

6.5 Summary

After reviewing system requirements and the alternatives approaches’ features, we found some limitations for using modern approaches in this project, mainly not having access to enough data and limited project time. We investigated the most promising traditional solutions by prototyping them and applying each technique to the available data to explore the feasibility of utilizing each of them and finding the best solution. The results ensured that a proper combination of these techniques could address this project's concern and make the goal feasible. Please refer to Appendix D to find the summary table of the given information in this chapter as well as visualization of the most important image processing techniques used in the proposed solution. The next chapter describes the proposed solution in more detail.

7.Pattern Recognition

In the previous chapter, we concluded to use traditional computer vision techniques as the primary solution direction. We also reviewed some of the most common techniques that could address the current problem. This chapter describes the proposed pattern recognition solution. The proposed solution addresses the model-based pattern recognition in the context of the wafer alignment. It enables pattern recognition, finding the precise location of its origin, and assigning a score to the recognized pattern.

7.1 Proposed Algorithm

The algorithm contains two main phases, training and recognition. During the training phase, some features of the model-based pattern (template) are extracted. These features are the basis for finding the pattern in the input image in the recognition phase. Once a pattern is trained, the training result can be used to recognize the same pattern in the input (wafer) images.

7.1.1. Training Phase

A model-based pattern contains descriptions of simple geometric shapes like lines and their origin and region on the micrometer scale. We can draw pattern shapes based on the descriptions, visualize the pattern and convert them into images. Converting a model description to an image is the prerequisite of using the proposed algorithm. That is why the solution can be easily extended to address image-based pattern recognition as well. Note that converting a model-based pattern to an image implies that the detection process should be done in the image coordinate system, i.e., on a pixel scale. Therefore, the pattern definition should be converted into the image coordinate system, using the image resolution. The image resolution defines the number of pixels per micrometer.

During the training phase, the goal is to visualize the pattern and extract some pattern features that could help with successful recognition. The number of pattern contours and the minimum and maximum contour area smooth the recognition procedure.

Since we proposed visualizing the pattern and creating an image using line descriptions, we are assured that images are not noisy and have proper contrast. It means that no preprocessing on the template pattern image is required. The image should be created in a grayscale format to ease the process of contour detection. Grayscale images only contain the image intensity information per pixel. The pattern lines should be drawn using white lines on a black background image to achieve the highest contrast.

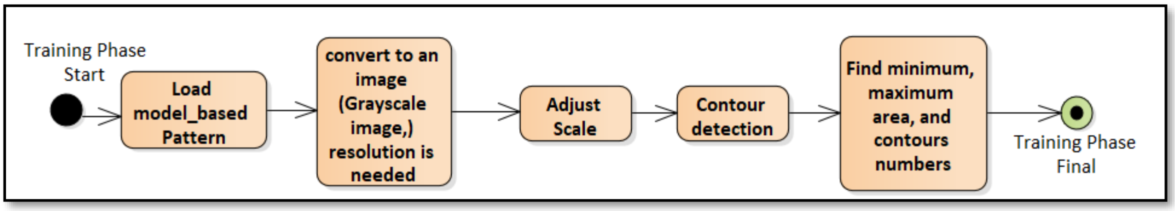

Training Phase
Start
Load model_based Pattern
convert to an image (Grayscale image,) resolution is needed
Adjust Scale
Contour detection
Find minimum, maximum area, and contours numbers
Training Phase
Final

Figure 12. Training phase

After finding the contours, the minimum and maximum contour area should be extracted as the training features. The goal is to confine the search area during the recognition phase using the training data and consequently reduce false-positive results. Figure 12 describes the training phase. Because the pattern scale could impact the contour area, we should adjust the scale during the training phase.

7.1.2. *Recognition Phase*

The main idea in the recognition phase of the proposed solution is to apply template matching using the NCC (Normalized Cross-Correlation) technique. It means that the template (the pattern image) should be slid pixel by pixel across the input image to find the recognized area in the input image. we considered four main steps for this purpose:

- Input image preprocessing to enhance the image quality and contrast
- Coarse recognition, to find the recognized area on the pixel scale
- Score calculation, to exclude false-negative results
- Fine recognition, to find the precise pattern origin on the sub-pixel scale in the input image

Preprocessing

Figure 13 depicts the recognition phase steps. The quality and contrast of the wafer images could vary depending on the machine camera or other factors. The image contrast could have a high impact on the detection result. Moreover, the same structures might have different intensity levels in different layers. Therefore, for a successful detection, a preprocessing step is required. For this purpose, we first converted the image to a grayscale one. Wafer images are not colorful and saving the image color information does not add any value. An image enhancement technique could be applied to the image to facilitate edge detection. The technique used in this algorithm was suggested in [13] to address the edge detection on low contrast medical images to detect brain tumors.

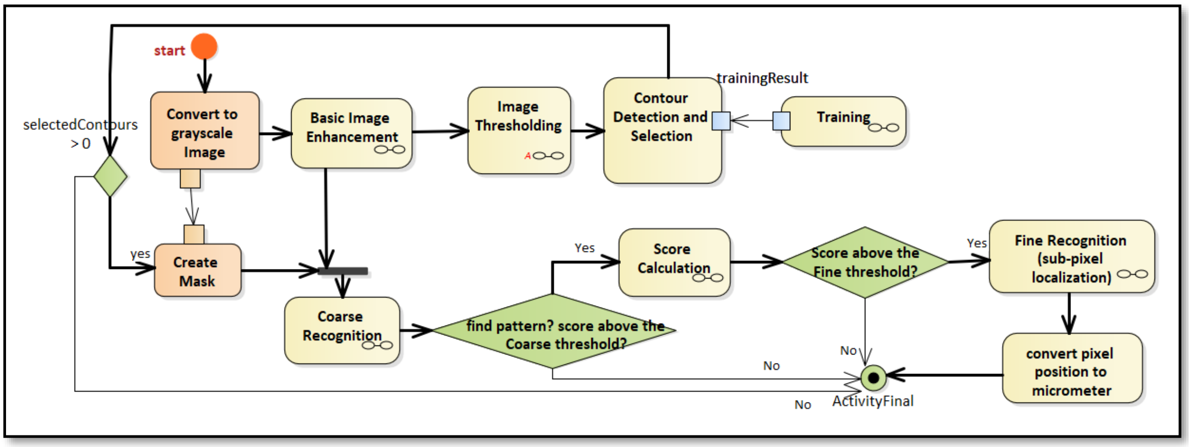

start
selectedContours
>0
Convert to grayscale Image
Basic Image Enhancement
Image Thresholding
Contour Detection and Selection
trainingResult
Training
yes
Create Mask
Coarse Recognition
find pattern? score above the Coarse threshold?
Yes
Score Calculation
Score above the Fine threshold?
Yes
Fine Recognition (sub-pixel localization)
convert pixel position to micrometer
No
No
No
ActivityFinal

Figure 13. Pattern recognition solution

Instead of searching for the pattern in the entire image, we try to first find the interesting area in the image using the training data. In this way, we can confine the search region and also exclude possible noises in the image. This technique could immensely help in reducing the false-positive results.

For this aim, we decided to find all the image contours and select those with an area size similar to the template (using the captured minimum and maximum area in the training phase), accepting some extent of deviation. Based on the selected contours, we can create an image mask.

Thresholding is an image processing technique that turns an image into a binary image, which helps in image segmentation. Thresholding is usually used as a prerequisite of contour detection. There are various ways to apply thresholding [28], such as simple or adaptive thresholding. We proposed to apply adaptive thresholding. As its name implies, instead of using a global threshold value, it determines the pixel threshold based on its surrounding pixel values. As a result, we can get different threshold values for different regions of the same image, which gives better results for images with varying illumination [22].

Gaussian blurring [24] is one the image processing techniques that help with smoothing the image and remove noises. We applied this technique before the thresholding with the same reason. Then, using the training result (minimum and maximum area of the template), only those contours in the input image should be selected that their area is within the range of the template contour size. Finally, before initiating the Coarse recognition phase, an image mask should be created by drawing selected contours on an image with a black background and applying it to the grayscale image.

Coarse recognition phase

In the suggested algorithm, we proposed using template matching in the Coarse recognition phase. The preprocessed image and the template pattern image are two inputs of this algorithm.

As template matching is an intensity-based approach and wafer images could have different intensity levels (due to the difference in the layer, photoresist material, field coordinate,) we cannot compare these two images without any modification on them. We know that model-based patterns are descriptions of a set of lines. The patterns can be recognized more accurately if valuable and comparable features can be extracted from both the input image and the pattern. With this goal, the edges of the input image should be detected. Therefore, the resulted image also contains a set of lines with almost the same intensity level. Gaussian blurring before the edge detection could remove some noises and improve the result of edge detection. Dilation is one of the morphological image processing techniques that adds pixels to the boundaries of objects in an image [25, 26]. We noticed that one iteration of dilation could connect detached corners of edges and increase the edges' intensity. If there are some shadows surrounding the patterns, blurring could make the situation worse for detecting edges, especially if pattern is brighter than the surrounding area (background). Based on experiments, we realized that concerning these images, if we apply adaptive histogram equalization instead of Gaussian blurring, better results can be achieved. Therefore, we proposed these two variants of the same algorithms, which users can decide which one could generate better results, depending on the images and the pattern, considering that the latter one could take a bit (about 5 milliseconds) more time. Therefore, using the second one is only suggested if good results cannot be achieved by the first approach. Figure 14 demonstrates the coarse recognition phase. Based on the experiments we conducted in this project, the first algorithm (applying blurring) works best in most of the cases, except regarding one specific layer. All the test results in the Verification and Validation chapter are based on the first approach, except the result of testing pattern D.

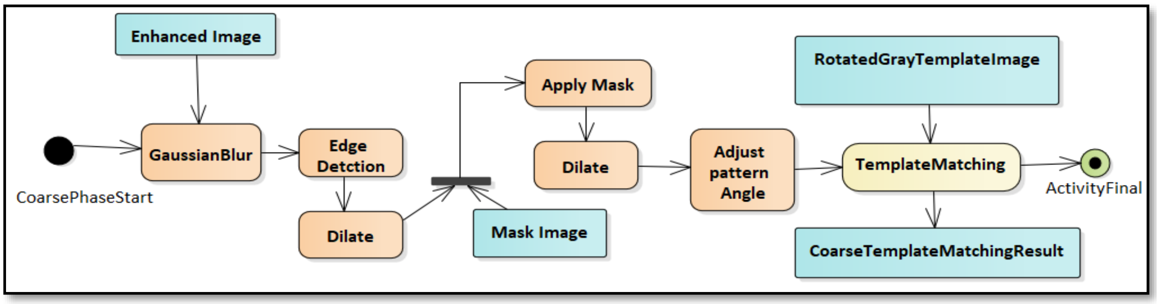


Figure 14. Coarse recognition phase

It is essential to note that not all edges are important. Hence, we applied the created mask (during the preprocessing step) on the edges to select edges of the interested area. This approach significantly reduces the risk of getting false-positive results. Depending on the image quality, there is a risk that edges cannot be found precisely, especially after applying the mask. Therefore, we proposed making the detected edges stronger by applying the dilation technique [25, 26]. Note that we applied the mask on the detected edges, not the input image. This is mainly because, applying mask on the input image, depending on the background color of the pattern, could cause an additional unwanted and incorrected edges. Incorrect edges could lead to inaccurate offset results.

If we have the template pattern and the masked edges of the input image, then we can apply the template matching technique [27] and NCC as the matching method to recognize the pattern and get the similarity score. Users might need to recognize a pattern with a specific degree of rotation. If the orientation of the template and the sliding window are not the same, the similarity score might be relatively low. Therefore, it is required to adjust the angle before initiating the recognition procedure.

There were two options. We could either rotate the input image or the template image. However, as the image size is larger than the pattern, adjusting the input image angle was more time-consuming. Therefore, in this algorithm, we suggested rotating the template. For more details on how the algorithm addresses the angle adjustment, please refer to Section 7.2. Briefly, the main idea is to introduce a bounding box for the template.

After adjusting the angle, NCC can be applied to the output image and the template image to find the position of the pattern (if it exists) on the pixel scale. Users can define a threshold score for the coarse phase. If the score is higher than the defined threshold, the final score should be calculated.

Score calculation

Although NCC returns a similarity score, this score could be relatively low and cannot be used as the sole criterion to decide on the success of the recognition. Different factors could impact getting low scores, such as:

- Difference between the intensity level of the image edges and the template
- Incomplete detection of input image edges due to the image low contrast or noises

Therefore, we proposed calculating the final score based on the intensity and similarity in the number of contours. Input image for this phase is the Region Of Interest (ROI) of the reference image in the Coarse phase. After selecting the recognized region based on the Coarse phase result as an input image for this phase, we need to amend it by applying a few more iterations of dilation. It helps in connecting detached edges; otherwise, the intensity score might remain incorrectly low.

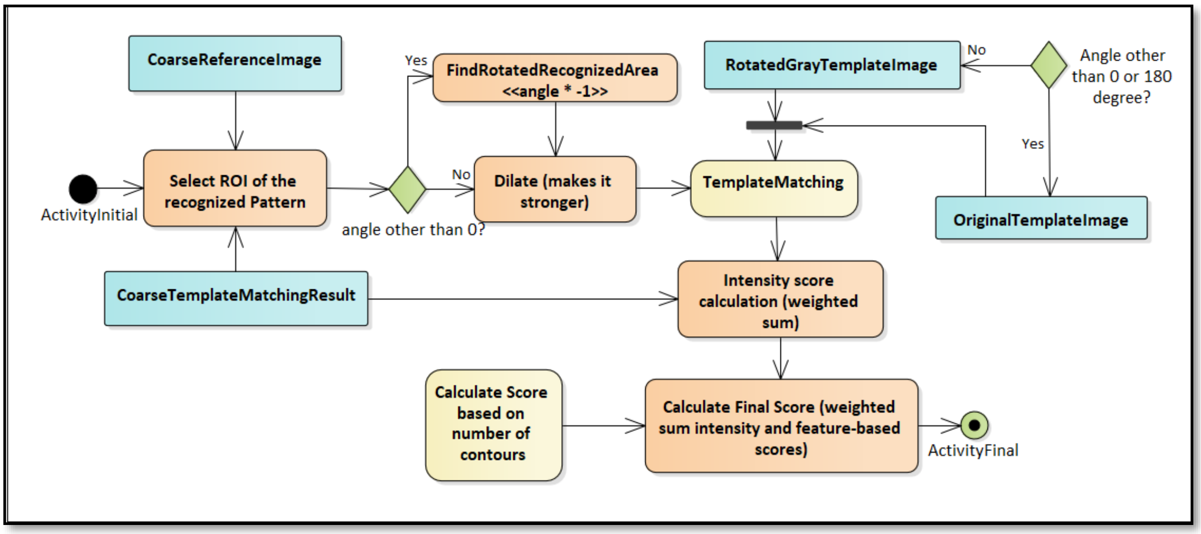

CoarseReferenceImage
Yes
FindRotatedRecognizedArea <<angle * -1>>
RotatedGrayTemplateImage
No
Angle other than 0 or 180 degree?
Yes
ActivityInitial
Select ROI of the recognized Pattern
No
Dilate (makes it stronger)
TemplateMatching
OriginalTemplateImage
angle other than 0?
CoarseTemplateMatchingResult
Intensity score calculation (weighted sum)
Calculate Score based on number of contours
Calculate Final Score (weighted sum intensity and feature-based scores)
ActivityFinal

Figure 15. Score calculation

Additionally, to detect false positives, we count the number of contours in the recognized area. Ideally, we should find the same number of contours in case of proper recognition. However, the same number of contours is not a guarantee for correct detection, as two different shapes could also have the same number of contours. Moreover, other factors like noises or image contrast could impact the detected contours. Therefore, we considered a weighted average of the mentioned score values to calculate the total score. Note that the score should be within the range of zero to one. Figure 15 depicts the score calculation steps. If the final score is above the defined accept threshold, the Fine recognition step is followed to find the sub-pixel pattern origin position. Score calculation in case of rotation could be slightly challenging, which is covered in Section 7.2.

Fine recognition phase

For finding the pattern sub-pixel position, as discussed earlier, an interpolation technique is suggested. Instead of interpolating the entire image, which could dramatically impact the performance, we suggested the interpolation of the region of interest. The region of interest is the recognized area with some additional margin to consider any potential shift that could be detected in the Fine recognition phase. NCC-based template matching on the interpolated pattern and the interpolated region of interest of the input image should be applied. Then the result should be downscaled to get the sub-pixel position of the pattern. Figure 16 illustrates the Fine recognition phase procedure.

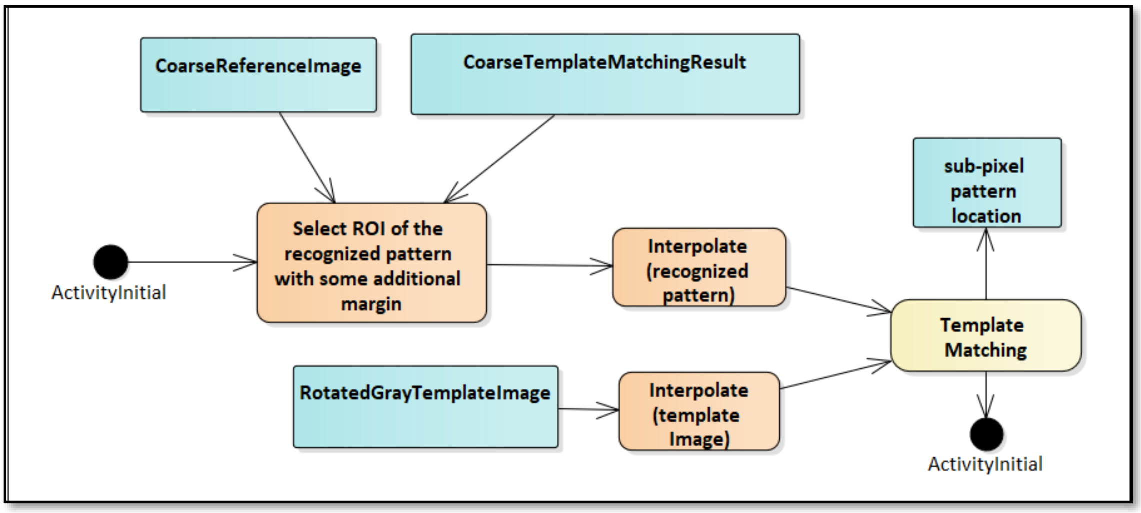

CoarseReferenceImage
CoarseTemplateMatchingResult
sub-pixel pattern location
ActivityInitial
Select ROI of the recognized pattern with some additional margin
Interpolate (recognized pattern)
Template Matching
RotatedGrayTemplateImage
Interpolate (template Image)
ActivityInitial

Figure 16. Fine recognition phase

7.2 Rotation

According to the system requirements, the proposed solution should recognize a pattern with a specific degree of rotation. Additionally, it was highly desired if ART could facilitate users to find the best angle under which the pattern could be recognized with the highest score.

In order to rotate an image, the rotation angle and its origin should be specified. Rotating a square-shaped pattern from its center is easy and can be addressed using the available open-course library functions without any specific considerations. However, this operation could be challenging if the image has a rectangle shape and the rotation origin is not the image center. This situation is tricky as it could lead to a case that the rotated image might not fit in the pattern image entirely after the rotation. Note that, after rotating a rectangular shape, its height and width might change.

To address this issue, we decided to introduce a pattern bounding box. The idea is to copy the pattern into a larger squared-shape image by aligning the center of the training region on the center of the bigger squared-shape image. In this case, we can rotate the pattern from its center and make sure the pattern remains inside the region after the rotation. Equation 1 defines the minimum width and height of the bounding box that ensures that the pattern resides inside after rotation. Allocating a larger area to the bounding box could increase the recognition time. Figure 17 shows a pattern located in a bounding box.

$$SquareSize = \sqrt{width^2 + height^2} \qquad \textit{Equation 1}$$

We search for the bounding box image (instead of the original pattern) in the input image whenever rotation is needed. This approach could cause an issue in the score calculation. Since a larger area is searched in the input image, the recognized area might contain additional data that might not be in the original pattern. Therefore, if we find the recognized area and apply another template matching, the score might be relatively low, leading to a false negative recognition. To deal with this issue, we could take two approaches:

I. Create a mask based on the original template image and apply it to the recognized region.
II. Rotate the recognized region in a reverse direction as the requested angle and apply NCC on the rotated recognized region and the original template pattern.

The first approach is more costly, as it significantly increases the recognition time. However, we only need to rotate the pattern bounding box in the second approach, which is usually smaller than the input image. Thus, we selected the second one. The rotation angle is checked each time, and a pattern bounding box is created only if the angle is other than zero or 180 degrees to boost the performance.

7.2.1. Finding the pattern offset with rotation

The final goal is to find the distance of the train origin from the image origin. During the wafer alignment, the train origin is usually the center of the pattern train region, and the image origin is the center of the input image. Therefore, the measured offset should be (0, 0) in a best-case scenario, indicating that the pattern is located at the expected position.

After Template matching, we can get the offset of the top left corner of the pattern from the top left corner of the input image. To get the accurate offset, first, we need to find the distance of the train origin from the train region's top left corner. Then, we can calculate the offset from the top left corner of the image. If the image origin is different from its top left corner, then the measured offset should be updated based on the distance of the image origin from the image’s top-left corner. Finally, we can calculate the offset based on the distance of the image origin from the image's top-left corner.

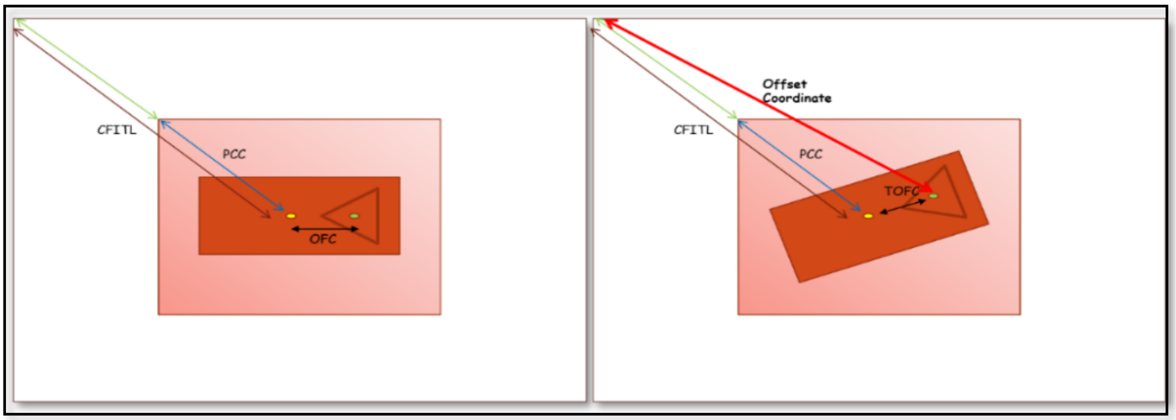


Figure 17. Measuring pattern offset

In the case of rotation and using a pattern bounding box, measuring offset is more complicated. In this case, the measured coordinate is the offset of the pattern bounding box from the image's top left corner (ITL.) Therefore, we suggested the following additional steps to find the accurate offset. Figure 17 demonstrates these steps.

1. Find OFC[5], the distance of the train origin from the train region center (center of the pattern bounding box.) Note that the train region center and center of the pattern bounding box are aligned.
2. Find PCC[6], the coordinate of the template pattern (pattern bounding box) from its top-left corner (micrometer.)
3. Convert the measured offset during the recognition to the micrometer scale.
4. Calculate CFITL[7], the coordinate of the pattern center from ITL.
5. Transform the train origin based on the angle
    a. Calculate TOFC[8], the new coordinate of OFC after the rotation, transformed origin.
    b. Find the offset of the transformed origin from the ITL (based on CFITL and the transformed origin.)
6. Finally, find the offset from the image origin instead of the image's top-left corner.

[5] Origin From Center

[6] Pattern Center Coordinate

[7] Center From the Image Top Left corner

[8] Transformed OFC

8. Architecture and Design

One of the most well-known and common approaches for describing a software-intensive system's architecture is the 4+1 view model defined by Philippe Kruchten [16]. It captures the software architecture into multiple concurrent views, including logical, process, development, physical views, and scenarios or use cases. We discussed use cases in Section 5.2. We covered the main topics related to the process view, namely the activity diagrams in the previous chapter. Please refer to Appendix A to find the sequence diagrams and class interactions. This chapter mainly focuses on the logical view.

8.1 Logical View

The logical view captures the functional requirements of the application as decomposition of structural elements or abstractions. It means it captures the application behavior into classes and packages. This section goes through the logical view of the deliverables described in Section 4.4 one by one.

8.1.1. RECOG

RECOG is the new pattern recognition library in the context of the wafer alignment. A big picture of the RECOG class diagram can be found in Appendix A. The following sections review it from three different perspectives: recognition, training, and serialization.

Figure 18 shows a close-up of the recognition perspective. The class called RecogTool is the main class responsible for pattern recognition. RecogTool (tool) owns a pattern (RecogPattern) and a run-time parameter object (RecogRunParams). It has some other properties, such as an image, a recognition algorithm, and a recognition result.

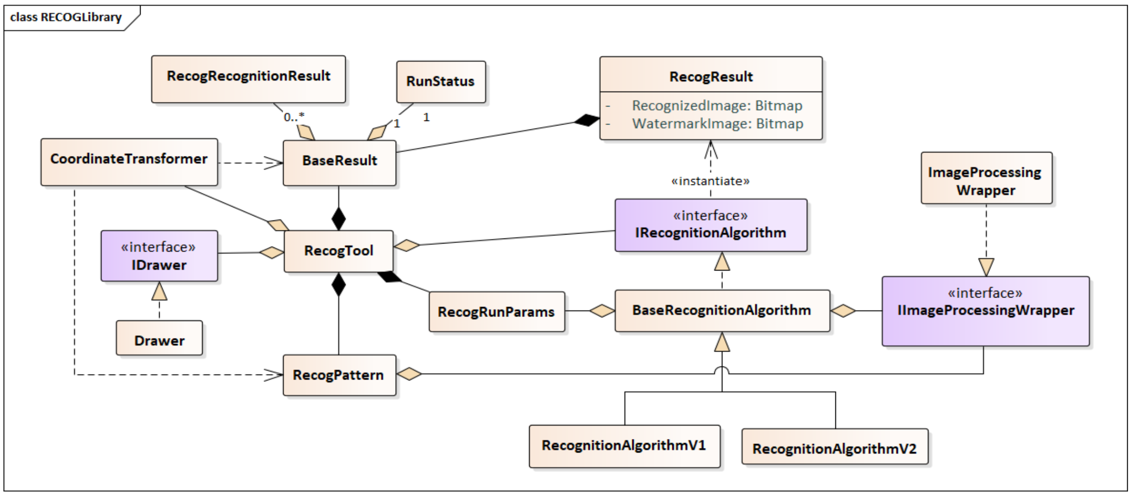


Figure 18. Recognition class diagram

IRecognitionAlgorithm is an interface designed to make the system maintainable and extensible. Recognition algorithms are subject to change and new recognition algorithms could be introduced over time. In this project, two similar recognition algorithms were proposed. We applied the strategy pattern to enable the selection of the proper algorithm at runtime. In this way, the tool is extensible to support new algorithms in the future. Additionally, the tool is not tightly coupled to the recognition algorithm implementation. Moreover, adhering to the façade algorithm, the complexity of the recognition algorithm is hidden from the tool.

We introduced an image processing interface to separate the concerns and increase the code's modularity, readability, and reusability. In this way, the recognition algorithm and the tool are loosely coupled with the implementations of the image processing functionalities. Currently, we implemented this interface mainly based on the OpenCVSharp library. However, in the future, it can be implemented differently. To comply with the YS pattern recognition wrapper, RecogTool holds an instance of BaseResult. However, an instance of IRecognitionAlgorithm can instantiate a more extensive result object, composing the BaseResult. In this way, the tool does not need to hold extra information. Finally, as coordinate transformation addresses a different concern (transforming the result from the pixel-based coordinate to the micrometer and vice versa,) a separate class takes care of this responsibility. The same logic holds for the Drawer class, which is mainly responsible for creating the result images.

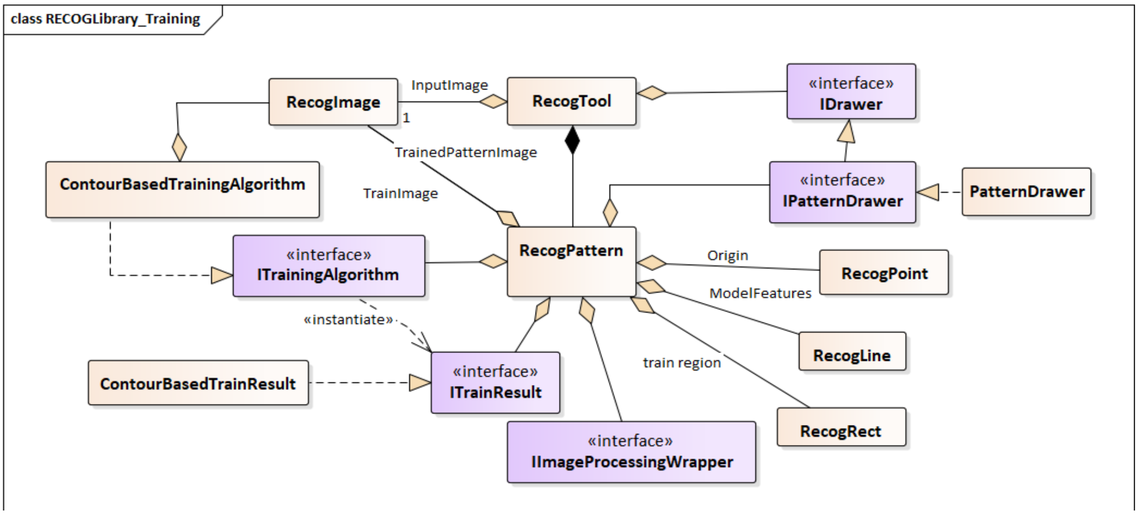


Figure 19. Training class diagram

Figure 19 focuses on classes involved in the training phase. From the training perspective, the pattern or RecogPattern class is the primary class, holding all the training data and pattern description. With a similar rationale in introducing the IRecognitionAlgorithm interface, we encapsulated the training algorithm using ITrainingAlgorithm. As a result, the pattern and the training algorithm are not tightly coupled anymore. Additionally, the pattern can easily support new training algorithms that support the extensibility of the design. By applying the strategy pattern, a proper implementation or algorithm can be selected. Depending on the training algorithm, different training results might need to be collected. Therefore, we introduce an interface with the basic properties for the training result, ITrainResult. Consequently, the design is open to support AI-based algorithms in the future with little effort.

As described, a model-based pattern description should be converted to an image. IPatternDrawer, a specialization of the IDrawer interface, was introduced to take care of this responsibility. Adding this functionality to the Drawer class is contrary to the SOLID principles. Interfaces should be specific and contain relevant functionalities. As we see, ImageProcessingWrapper is reused in the training phase, which shows the modularity of the design.

Figure 20 demonstrates classes involved in the serialization of the tool and pattern. The serialization objective is to dump the tool and pattern state and make them reusable. Separation of the serialization logic from the tool or pattern improves the extensibility and maintainability of the design and code. By defining a new interface, we hid the complexity of the serialization from the tool or pattern, made the code loose coupled. New serialization formats can be supported in the future without any changes to the tool.

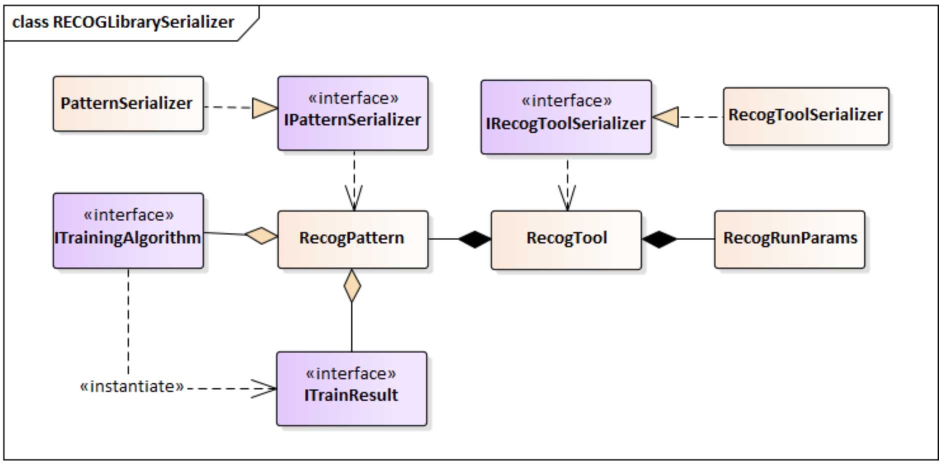


Figure 20. Serialization class diagram

8.1.2. Alignment Recognition

As described in the domain analysis Chapter, there has already been a shift towards a modularized software application in YS, and a pattern recognition wrapper was introduced to decouple the pattern recognition library from the rest of the YS business logic. To integrate RECOG into the YS solution, we needed to implement the wrapper's interfaces using the RECOG library. Figure 21 depicts the class diagram of the newly introduced class library called "AlignmentLibrary," which is the RECOG-based implementation of the YS common pattern recognition interfaces.

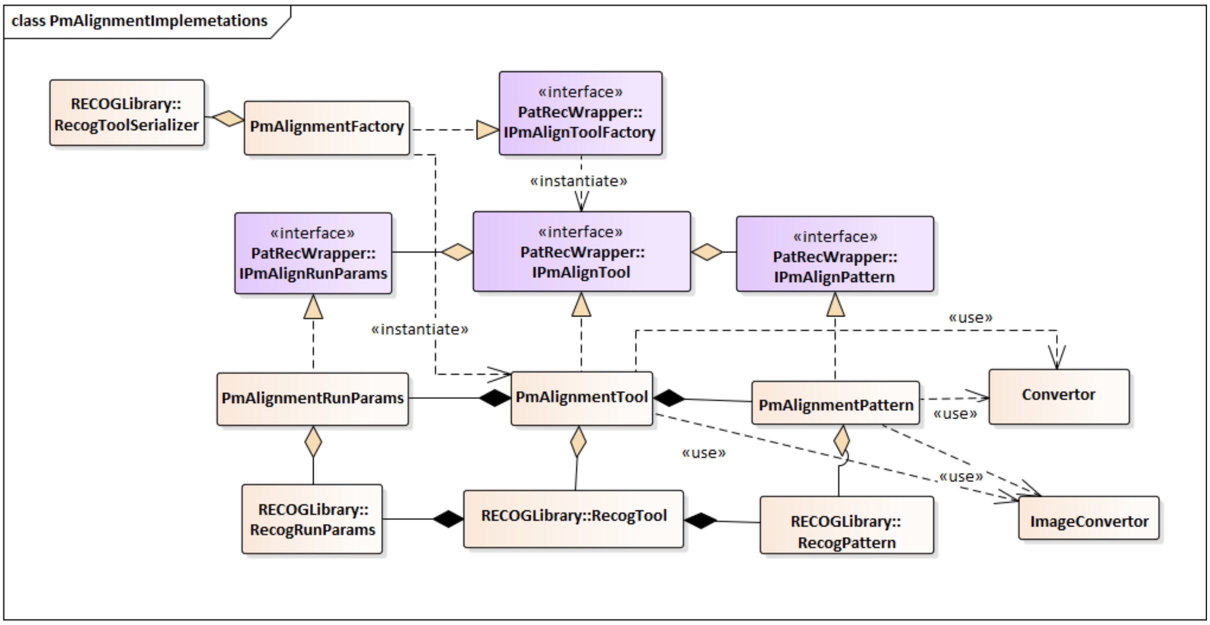


Figure 21. Alignment recognition

Following the adapter design pattern, each class implements a relevant YS pattern recognition wrapper interface and has an instance of the correspondence class in the RECOG library. Thus, it implements the target interface by delegating to a RECOG object at run-time. It is important to note that the pattern recognition interfaces were designed based on the Cognex definitions. As a result, some of the Cognex library's properties do not have a corresponding property in the RECOG library.

8.1.3. ART

APT is a Cognex-based pattern recognition tool that facilitates creating alignment profiles, running pattern recognition, troubleshooting, and other functionalities. ART[9] is a new GUI application developed as a replacement for APT, based on the RECOG library. It enables users to run pattern recognition, find out the best training and run-time parameters, and create alignment profiles. Additionally, it enables converting Cognex-based alignment profiles or Cognex objects into their corresponding RECOG-based files or objects. Using this feature, users can readily use currently available alignment profiles to run pattern recognition and customize them based on the RECOG specific parameters.

ART Design

ART is designed based on MVVM architectural design pattern and WPF. The rationale behind choosing MVVM and WPF is discussed in Appendix A. ART has three main views (tabs), including batch recognition, pattern trainer, and convertor. The 'pattern trainer' tab contains different user controls. However, almost all the user controls belong to the 'pattern trainer' tab, which should reflect a singleton instance of the tool. This implies that all the user controls sharing the same property should be notified upon any changes in the tool state. In order to address this requirement while avoiding tight coupling between view models, the mediator or observer design pattern could be applied.

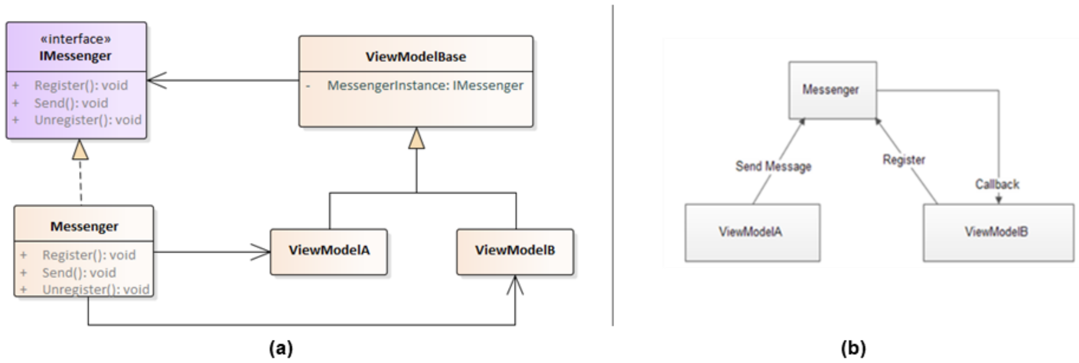


Figure 22. (a) ViewModelBase, (b) Messenger class diagram

While the Observer pattern defines a one-to-many dependency between objects, the mediator defines an object that encapsulates how a set of objects interact with each other. A mediator promotes loose coupling by keeping objects from referring to each other explicitly. Therefore, classes communicate through a mediator instead of direct communication. There are some merits in offloading the communication to a class that is only responsible for the synchronization. First, the single responsibility principle can be met. Besides, it promotes loose coupling by keeping objects from referring to each other explicitly. Figure 22b shows the idea of using a messenger or mediator to handle the communication between different ViewModels. Figure 22a depicts the idea of using a mediator. IMessenger plays the role of the mediator. All view-models should inherit from the ViewModelBase class, which has an instance of a messenger.

Figure 23 depicts the ART high-level architecture. DataAccess is a singleton class that generates and shares a single tool instance among view models using PmAlignmentFactory. ModelViewLocator contains the list of all ViewModels, realizing the proper ViewModel implementations using a proper IOC container. All ViewModels inherit from the ViewModelBase, which enables the communication

[9] Alignment Recognition Tool

between different ViewModels. There is a one-to-one mapping between each View and ViewModel. Finally, as mentioned, all ViewModels have access to a singleton object of the tool using DataAccess.

Considering different user controls for each function helps with the separation of concerns and increases the readability and maintainability of the code, accepting adding to the code complexity. The ART complete class diagram can be found in Appendix A.

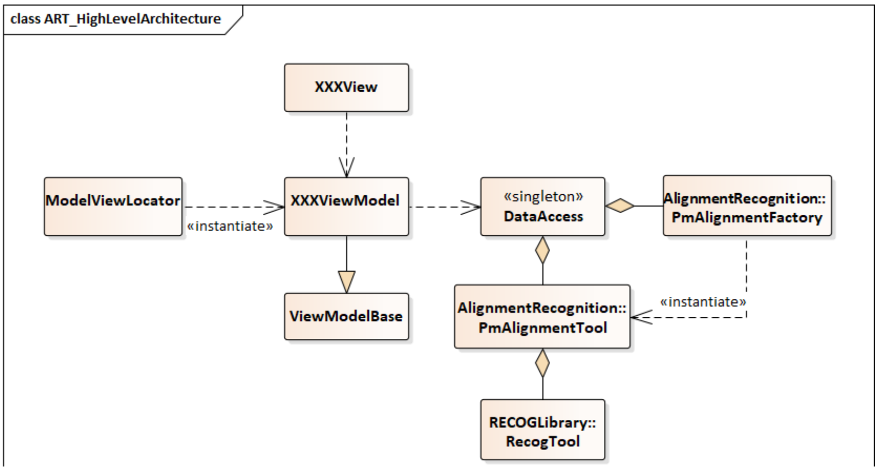


Figure 23. ART high-level architecture

8.1.4. Integrated YieldStar

As mentioned earlier, the first and the main prerequisite towards integrating RECOG into the YS application was the realization of the common pattern recognition interfaces using the RECOG library. However, it was not the only requirement. Here is the list of most important questions and concerns that needed to be answered for successful integration:

- What are the classes or interfaces that have contribution to the wafer alignment?
- How to update these files without impacting other modules and YS functionalities?
- How to support both libraries (Cognex and RECOG) at the same time?
- How to read RECOG objects while performing wafer alignment?

Figure 24 demonstrates an abstract overview of the most important classes or interfaces that play a role in the wafer alignment process. Green boxes are the newly introduced classes that are explained in the following section. As it shows, wafer alignment can be done either in the fast mode or the normal mode. During the alignment phase, the goal is to measure marks (regardless of the alignment mode), which is the responsibility of MarkMeasurer class.

Each mark has the pattern definition (description of a serialized tool object with the pattern description and all the required run parameters) as well as the pattern's expected position. There is a set of recognizers on top of each other that MarkMeasurer uses to recognize and measure required data, such as PositionOffsetRecognizer. All recognizers are built on top of the ITrainedPatterRecognizer. For every single mark, a proper ITrainedPatternRecognizer should be created. TrainedPatternRecognizer has the ownership of a tool (IPmAlignTool.) This tool could be either a Cognex or a RECOG object.

PatternDataRepository takes care of a list of the already created recognizers for each pattern data (mark.) If it is a new pattern data, then a new ITrainedPatternRecognizer should be created. Creating a new recognizer requires loading the tool object; thus, proper implementation of IPmAlignFactory and ITrainedRecognizerFactory should be in place. The implementation of the TrainedPatternRecognizer was based on the Cognex. Therefore, there was a need for a new realization of this interface based on the RECOG library. Then the factory could decide from which implementation to instantiate. RecogTrainedPatternRecognizer is a new class introduced, which is shown in green in Figure 24. It holds an instance of PmAlignmentTool (RECOG-based tool) similar to the TrainedPatternRecognizer that holds an instance of PmAlignTool (Cognex-based tool.)

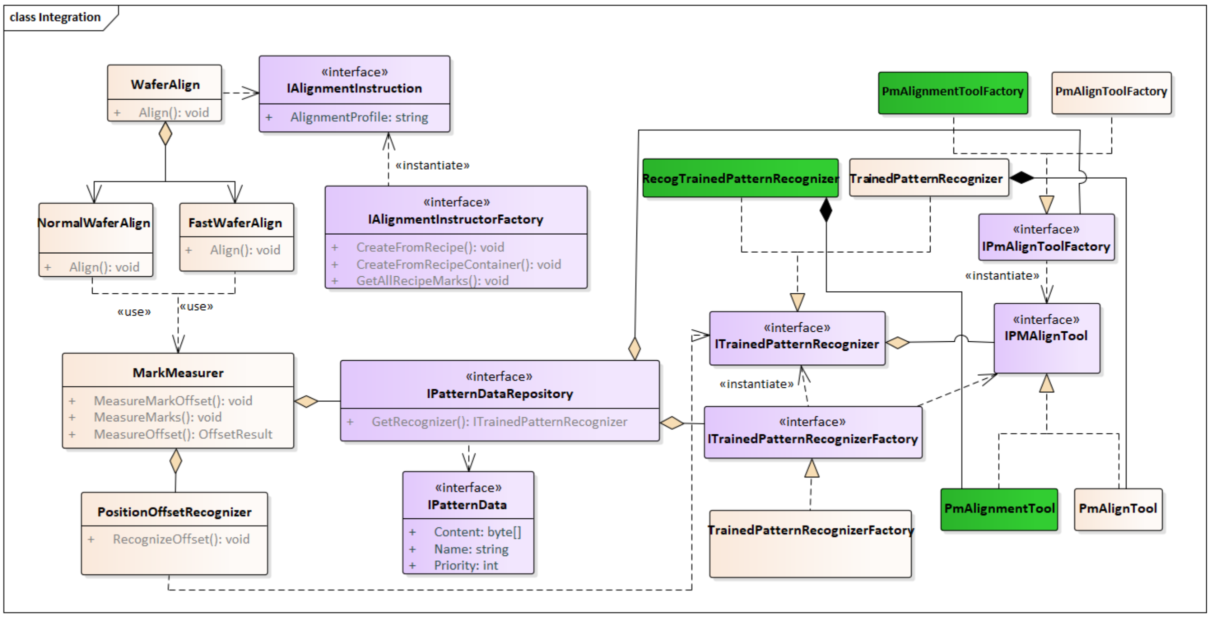


Figure 24. Wafer alignment class diagram

IPatternDataRepository required some changes to generates a correct instance of the ITrainedPattern-Recognizer. However, it is important to note that IPatternDataRepository has applications in other YS modules, such as Sensing. Therefore, the changes in this class should be safe, not to impact the other modules. Thus, we introduced a new parameter to be set during the wafer alignment.

PatternData is created based on the recipe information. However, the available recipes only contain the Cognex-based tool definition, while we needed a RECOG-based one. The ideal solution was updating the recipe thoroughly to contain the new alignment profile. Since this change required considerable time and procedure, it was not in the project scope. However, some other solutions were good enough for the sake of feasibility study and prototyping, such as:

- Converting the alignment profile on the fly: The downside of this solution was that it was not easy to customize the tool, and we needed to stick to the default configuration parameters. Additionally, the Cognex hardware dongle should be available during the conversion. As a result, by this approach, we could not evaluate if wafer alignment could be done solely using RECOG or not.
- Utilizing a customized alignment profile: We needed to create a customized alignment profile and then force YS to read the pattern data from the newly created file instead of extracting the alignment profile from the recipe. However, then there was a need for an additional file besides the recipes.

As utilizing a customized alignment profile was a better option, we selected the second option. By introducing a new configuration file, YS can support both libraries simultaneously. Using this file, YS can decide which library should be used. PatternData can be extracted from AlignmentInstructions, which can be generated using IAlignmentInstructorFactory. Therefore, changes were made in this interface implementation to accurately create the PatternData either based on the recipe or the added alignment profile.

8.2 Other Views

A physical view represents the deployment layout or infrastructure of an application. Ideally, the proposed pattern recognition library should be deployed on the YS machine. Figure 25 depicts the main components involved in the realization of the wafer alignment using RECOG. They should all be deployed on the YS machine. For the sake of simplicity, only the relevant libraries are demonstrated in this deployment diagram. It is important to mention that at this stage, these libraries are not deployed at the YieldStar, and this figure depicts the final goal. The development view illustrates a system from a programmer's perspective. Figure 9 and Figure 25 together could reflect this view well enough.

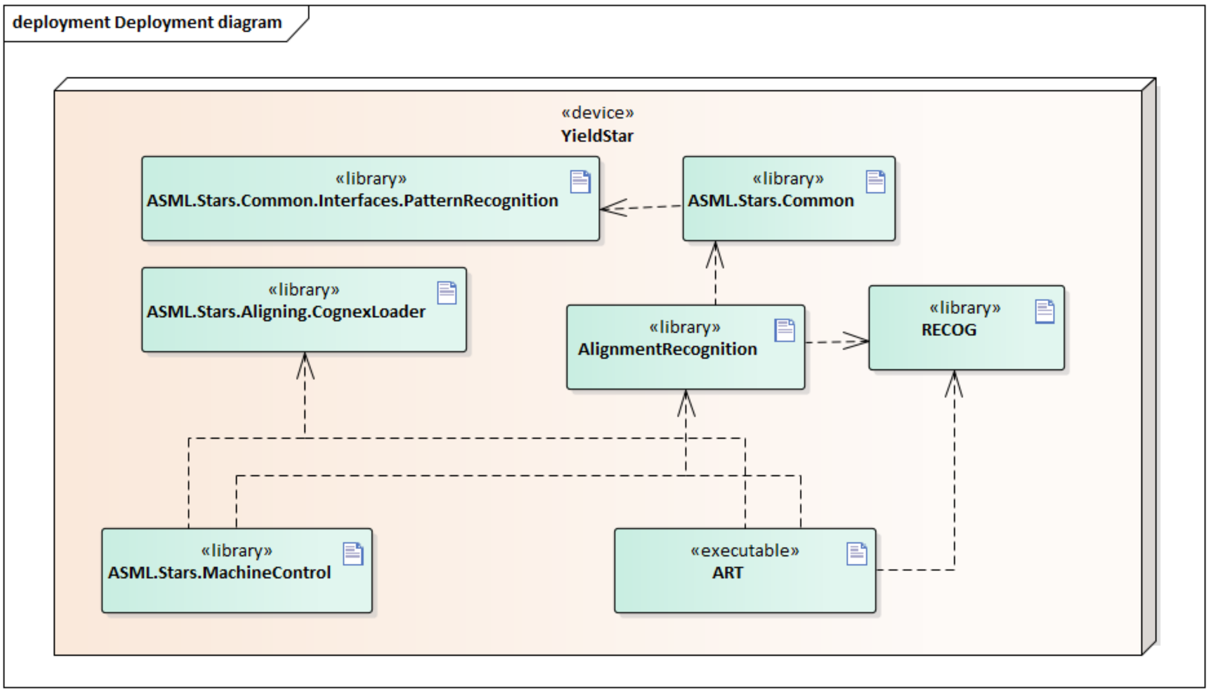

deployment Deployment diagram
«device»
YieldStar
«library»
ASML.Stars.Common.Interfaces.PatternRecognition
«library»
ASML.Stars.Common
«library»
ASML.Stars.Aligning.CognexLoader
«library»
AlignmentRecognition
«library»
RECOG
«library»
ASML.Stars.MachineControl
«executable»
ART

Figure 25. Deployment diagram

9.Implementation

We discussed the system architecture and design from different viewpoints in the previous chapter. There are some details regarding implementation, which are discussed in this chapter. All implementations are based on C# version 7.3 and .Net framework 4.8 to comply with the YS technologies.

9.1 RECOG Library

For implementing RECOG, we took advantage of the OpenCVSharp library, which is a cross-platform .NET wrapper of the OpenCV library used for image processing and computer vision algorithms. For more information about the rationale behind choosing OpenCVSharp, please refer to Appendix A. In order to improve the performance of the library, some considerations were taken into account.

The first performance bottleneck that we observed was related to the image enhancement technique we used. As mentioned in Section 6.4, BCET[10] was proposed to enhance the image contrast. However, it is a compute-intensive algorithm because some operations need to be done on each pixel or float array of the image. Therefore, processing larger images could become more time-consuming. To speed up the algorithm, we used parallel loops to apply the same set of operations on all inputs simultaneously.

Second, in Fine recognition phase, we interpolated only the recognized region with some margin instead of entire image and executed the template matching based on the recognized region. We mentioned that Gaussian blurring should be used to smooth the image and remove potential noises. The kernel size used in this process could play a significant role, such that the bigger the kernel size, the faster the algorithm since the convolution could be done faster. Therefore, we fine-tuned the algorithm with the best kernel size, based on the experiment results, although this value is configurable. Similarly, the interpolation scale has a dramatic effect on the performance. We made the interpolation scale configurable while we set its default value to the optimized value.

The next technique was related to creating image masks. After analyzing each step's elapsed time, we figured out drawing contours on the images to create a mask was the most time-consuming step. To improve the speed, instead of drawing contours, we drew the contours' rectangle. This technique's downside is a slight risk of including available noises next to the pattern edges. However, the rectangle area is usually quite similar to the contour area, especially with regard to the alignment targets. On the other hand, using the contour's rectangle, the edges are sharper, facilitating better pattern recognition results. Always, there should be a trade-off between performance and accuracy. In this way, we gained a significant improvement by this approach, without a considerable impact on the accuracy.

Finally, we decided to update the input image pixels in place instead of creating a new image and setting it during the integration phase. During the integration, a float array of the image is passed to the tool. We used the available YS's performance library to update the image pixel's data. Creating a new image and allocating memory is only required for the first usage of the recognition tool or if the image size has changed. In this way, we saved a considerable amount of time and improved the performance substantially.

During the implementation and testing phases, we faced some edge cases that RECOG was not able to address them initially; for instance, if the train region was larger than the input image, or when the

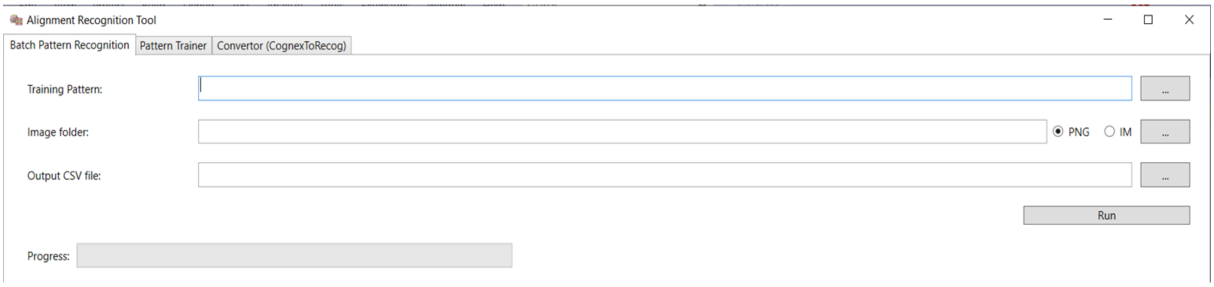
[10] Balanced Contrast Enhancement Technique

pattern presents in the image partially. For more information on how we addressed these edge cases, please refer to Appendix C.

9.2 ART

ART was developed using WPF and based on the MVVM framework. We used the MVVMLight toolkit to implement the MVVM framework. The rationale behind choosing this framework can be found in Appendix A. We tried to design ART's user interface similar to APT. In this way, users can feel the same experience while using the new tool, making the tool more user-friendly. Three ART tabs are as follows:

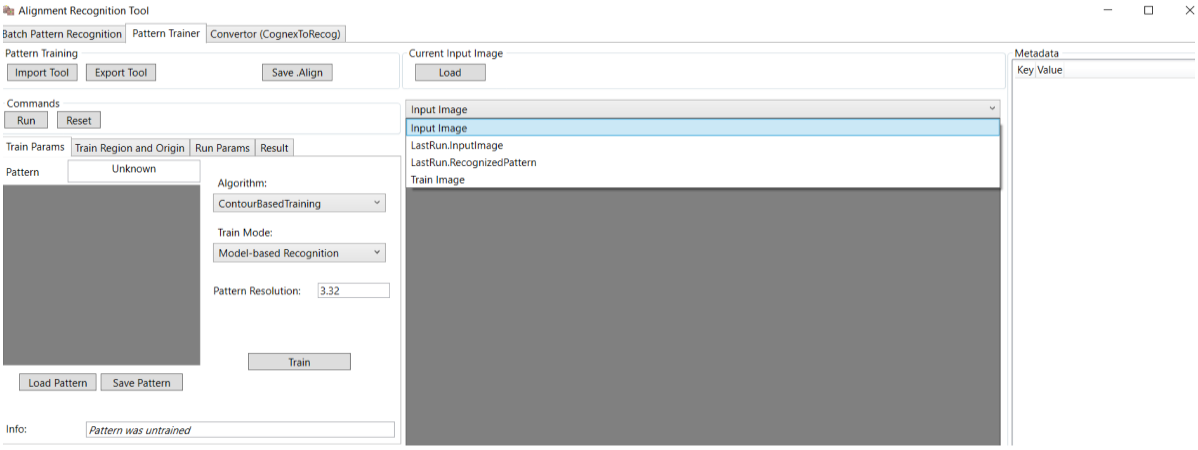
1. ***Batch Pattern recognition***: Figure 26 shows ART's batch recognition tab. Users should set the alignment file saved with the "recog" extension and define a folder containing images either in PNG or IM format. Finally, they should specify the path that the final output should be saved. The batch recognition result is saved in the CSV format, containing the recognition result per image in the directory.



● PNG ○ IM

Figure 26. ART Batch recognition tab

2. ***Pattern Trainer:*** Pattern Trainer, shown in Figure 27, is the main ART tab. Users can create, load, and export pattern recognition recipes in different formats. They can load images, train patterns, set train or run parameters, and finally, run a pattern recognition and preview the results.



Pattern: Unknown

Algorithm: ContourBasedTraining

Train Mode: Model-based Recognition

Pattern Resolution: 3.32

Info: *Pattern was untrained*

Figure 27. ART Pattern Trainer tab

3. ***Convertor:*** ART has a new and different tab compared to APT, which is the Convertor tab. Users can specify either a Cognex-based XML file containing a pattern recognition recipe or a Cognex-based alignment profile and convert them to the corresponding file formats. Figure 28 depicts a screenshot of this tab.

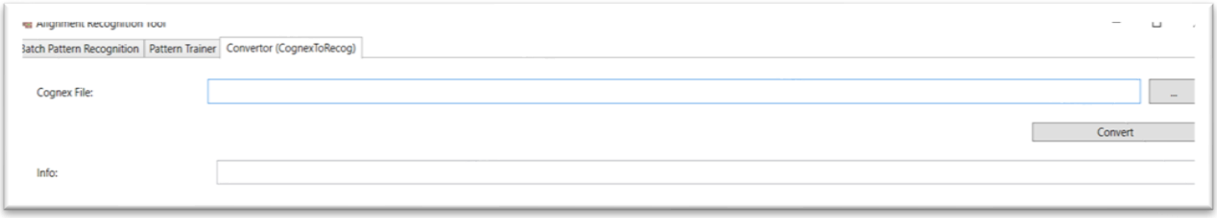

Alignment Recognition Tool
Batch Pattern Recognition
Pattern Trainer
Convertor (CognexToRecog)
Cognex File:
...
Convert
Info:

Figure 28. ART Converter tab

9.2.1. *Configurable recognition recipe*

It is not easy to get the best pattern recognition result using the same recipe for different pattern shapes and images with different qualities. The more the algorithm is configurable, the more the chance of fine-tuning the parameters and getting better results. However, introducing numerous run-time parameters could turn the recognition procedure into a hassle for the user.

To find a balance, ART suggests the best parameter values to the users as default values. Usually, without any changes to these values, the best results are achievable. There are some parameters for more advanced users that they can play with and create a customized recipe. They can hover over each parameter to learn about their usage.

Two different recognition algorithms are introduced to avoid introducing many different parameters. These two algorithms are quite similar, and there are some minor differences mostly related to preprocessing the images and calculating the score. Depending on the image quality or the pattern shape, one could outperform the other one. As this is the feasibility study and not all possible patterns could be verified in this study, both algorithms are available in ART. In this way, the best decision could be made based on various experiments by the functional engineers.

9.2.2. *Training*

This project aims to use the currently available patterns/recipes and creating a new model-based pattern from scratch was not in the project scope. Each pattern can be used in various alignment profiles, and alignment profiles can be used regularly. In other words, the reusability of these files is essential. Therefore, we enabled users to extract the pattern data from the converted tool or alignment profile and save it in an XML format. In this way, they can easily modify the content or even recreate a new model-based pattern.

For extracting and converting patterns created by Cognex, we have to find the description of lines. Notably, these lines' starting point and endpoint are not necessarily from the top left corner of the image, and they might have even negative values.

For pattern visualization, first, the line coordinates should be transformed and recalculated from the top left corner of the image to convert them to positive values. An additional margin from the top left corner should be added. This margin depends on the train region specification. Finally, the pattern resolution is needed to convert the values from the micrometer scale to the pixel scale.

In ART, users can create a tool by loading a pattern XML file. As it is shown in Figure 27, ART has four different image previews, which are as follows:

- *Input Image: It* represents the original input image.
- *LastRun.InputImage*: It illustrates the input image with a green rectangle surrounding the recognized region. If the recognition is unsuccessful, it shows a blank image.

- *LastRun.RecognizedPattern*: It demonstrates how the recognized region is the same as the pattern by aligning them on top of each other.
- *Train Image: It* represents the pattern with the train region rectangle and the train origin.

Upon any changes in the train region or train origin, the train image reflects the changes. It helps the user to set the train region and origin properly. Also, in case of any changes in the training parameters, the pattern should be retrained.

It is mentioned that knowing the pattern resolution is a must-have for pattern visualization. Upon loading an image, pattern resolution gets updated automatically based on the input image resolution, which makes the tool easier to use.

9.2.3. Tool Serialization

Each alignment profile contains a byte array of the tool. Each tool can be serialized in SOAP[11] or binary format to conform with the YS architecture. With APT, users can save the tool in the SOAP format with an XML extension. To help users quickly distinguish between the tools being serialized using the Cognex library or the RECOG library, we decided to save the tool with a new extension (recog.)

Besides, in order to increase the reusability of the serialized tool object, only the tool state will be captured and serialized. It means that properties such as the recognition or training algorithm implementation are not saved. It decouples the serialized object from the implementation. The serialized form of the tool is a property of the alignment profile.

[11] Service Oriented Architecture Protocol

10. Verification and Validation

An essential step in each project is the verification and validation of the system. Verification evaluates whether the system fulfills the specified requirements, whereas validation evaluates whether the system targets the main concerns and expectations of the stakeholders [18]. This chapter clarifies how the system is tested against the requirements, various approaches we took for testing, the expected results, and finally, the test results.

10.1 Test plan

After each step, including prototyping, implementing the library and ART, and finally, after the integration, we conducted various tests to make sure the system fulfills the requirements, detect potential bugs, fix them and gain higher confidence in the product. The main goal in testing was to verify if RECOG can recognize patterns accurately and efficiently compared to Cognex. While we tried to achieve promising performance, we gave a higher priority to accuracy than to performance. During the testing phase, we mainly used the Cognex result as the basis of how RECOG performs. We believe if RECOG can generate results as good as Cognex, then the requirements are met. The reasons behind this decision are mainly as follows:

- There was no ground truth data available. As part of the unit testing, we created some synthetic images to evaluate the library's accuracy; however, those images could not represent all actual image's quality variation or might not cover specific edge cases. Therefore, testing only against synthetic images could not reflect the library's reliability and accuracy completely.
- The project goal was to find a Cognex replacement. Therefore, it was essential to compare the results with the Cognex results.
- Finally, after integration and wafer measurement, we can calculate KPIs (defined in ASML) such as TMU[12] or matching, which could help in identifying how well RECOG is performing the pattern recognition. These metrics are measured based on the values that are dependent on the pattern recognition results. Therefore, if there is an error or inefficiency in the pattern recognition results, it should be reflected at this stage. There are specifications for these KPIs in YS that, based on them, the RECOG performance and accuracy can be appropriately evaluated.

Generally, in order to deliver new projects or any changes to the YS master branch, some steps should be taken to ensure that new changes in the code meet the expected code quality and will not break the system. First, the code should be reviewed by the main stakeholders. Second, achieving at least 80 percent test coverage is mandatory. Therefore, the stakeholders reviewed the code to make sure it complies with the YS code standard.

Depending on the project phase, test type, and goal, we performed various tests on different machines, such as the local system, DevBench (a virtual machine similar to the YS machine,) and finally on the real machine.

10.2 Verification

During the verification phase, the main goal was to check if the deliverables were error-free (as much as possible) and if all the must-have functional requirements were met. To gain high test coverage and make sure that the system was well tested, we implemented a considerable number of unit tests and

[12] Total Measurement Uncertainty

integration tests. Nunit [19], one of the .Net unit-testing frameworks, was used for the test implementation. The tests were automated using Bambo CI/CD [13] build server that is being used in YS.

Achieving high test coverage by unit testing in terms of ART (the GUI tool) was not possible. Hence, in addition to unit testing, the tool was tested manually. Note that ART manual tests implicitly test the libraries as well because the main part of ART business logic is implemented in the RECOG library.

Initially, we defined a set of test requirements based on the system requirements. Then, we created a set of test cases that could cover the test requirements. Please refer to Appendix B to find more details about the test cases, the traceability matrix (Table 16), and manual tests.

Using the available model-based patterns and the wafer images captured during a real wafer alignment on the machine, we ran a batch pattern recognition using APT to record the Cognex pattern recognition results. Then, alignment profiles and the tool XML definitions were converted to an acceptable input for RECOG, using ART. Afterward, we ran the same test with the same inputs using RECOG and ART and compared the results. To find out how reliable RECOG is in finding the patterns, we repeated the same procedure for different patterns.

Albeit mostly Cognex results were used as the ground truth, we also designed some synthetic test cases by creating triangle and trapezius pattern shapes to validate the accuracy of the pattern recognition result. As a next step, we evaluated RECOG accuracy with different rotation angles. We asserted that the measured offsets by RECOG were less than a pixel, even with rotation.

The strategy was to execute all the test cases and verify that the result matches the expectations. We achieved 87, 73, and 56 percent test coverage with respect to RECOG, AlignmentLibrary[14], and ART, respectively. We could not achieve higher test code coverage with respect to AlignmentLibrary because some of the pattern recognition wrapper (Cognex) concepts are not meaningful within the new library. Therefore, testing those features was not feasible.

10.3 **Validation**

Previously described tests ensured us that RECOG and ART passed all test cases, and with high test coverage, we gained high confidence that the possibility of getting an error is low. Also, the results demonstrated that RECOG could recognize patterns in machine images with reasonable accuracy. However, one crucial question remains. Can RECOG facilitate a real wafer measurement on the machine with comparable accuracy and performance with Cognex? By answering this question, not only can we validate the system, but we can also verify the integration of RECOG into YS.

10.3.1. Test on a local system

Since testing on the machine was expensive, we did some initial tests on local machines before running tests on the machine. These tests can be categorized into two main groups~~:~~

- Testing algorithm flexibility and accuracy
- Testing the integration by running tasks or replays

Testing algorithm flexibility and accuracy

Before the integration phase, we ran various tests to ensure that the proposed algorithm can recognize patterns accurately. Based on the NF5 requirement, the proposed solution should be flexible to

[13] Continuous Integration/ Continuous Delivery

[14] The library that contains the pattern recognition wrapper's implementation based on the RECOG library

recognize patterns irrespective of the image quality. We mentioned in Chapter 4 that depending on the machine camera or the photoresist materials used for each layer, patterns might look slightly different, and their contrast might vary. To ensure that RECOG is flexible, we tested it against different patterns (different shapes) and images of different layers of the same pattern.

Figure 29 and Figure 30 show a comparison between offset values (X offset and Y offset, respectively) measured by Cognex and RECOG library for a pattern, called D. In each figure, The X-axis shows different images in these charts, while the Y-axis shows X (Figure 29) or Y (Figure 30) offset of the pattern origin from the image origin. The unit size in the Y-axis of the figures is about 0.3 μm, which is almost equal to one pixel, based on the image resolution. As it can be seen, almost always, the difference is less than one pixel (one chart unit.). Mostly RECOG can perform as well as Cognex. There are some cases that the RECOG library can find the pattern, while Cognex cannot find it with the same setting. These are mainly those images that the pattern is covered with a shadow. Using the second algorithm described in Section 7.1.2 made the proposed solution more promising than Cognex. For some of the images (mainly images after 120) the reported offsets are quite different from the others (with both libraries.) It could be due to the field location. If we try to recognize a target at the edge fields, the target is more off centered than a target at the coarse (center) field. It can also be due to the wafer load, which operates a bit worse (the wafer is put a bit further away on the stage than usual.)

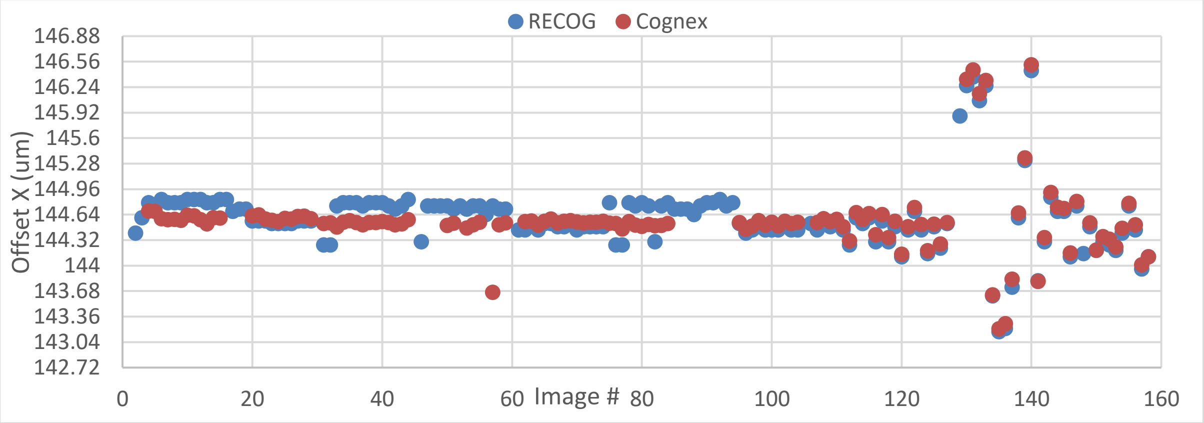


Figure 29. Recognition offset comparison, X axis, Pattern D

One of the main concerns regarding pattern recognition is that the number of false positives should be as few as possible. In Figure 30, Cognex reported a false positive result (for the 139th image) as its value is quite different from the rest, while RECOG offset seems to be more accurate.

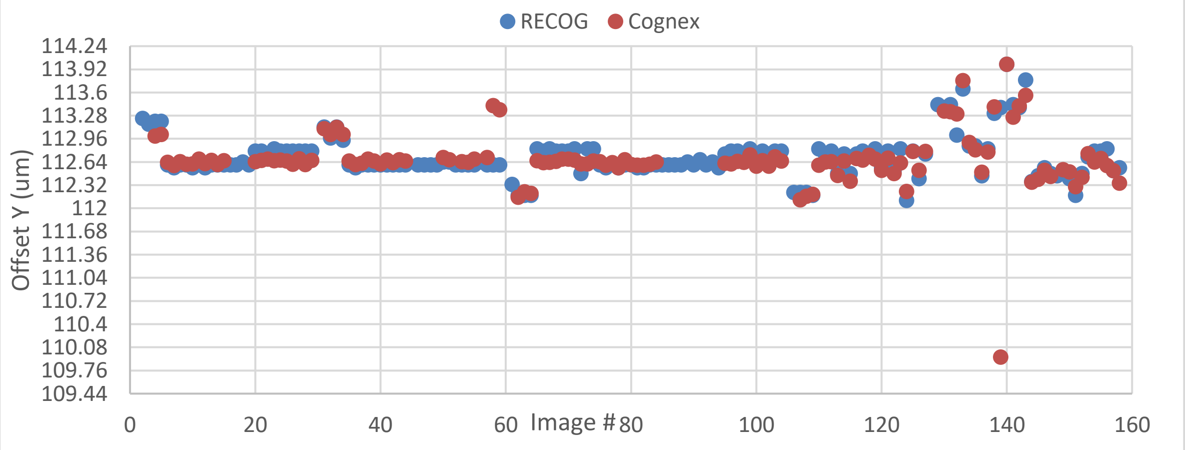


Figure 30. Recognition offset comparison, Y-axis, Pattern D

Figure 31 and Figure 32 show a comparison between offset values measured using Cognex and RECOG library for a different pattern, called V, on the X and Y axes, respectively. This pattern is quite different from pattern D; because it consists of only one shape, while pattern D consists of four detached shapes. Besides, the contrast of these images was relatively lower than the images of the former pattern. These images were very noisy. Noises were mainly attached to the shape edges, which made the recognition very challenging. Finally, the image resolution was different concerning this pattern. As it can be seen, RECOG can recognize patterns even in noisy low-contrast images with a great extent of accuracy. There were three images that Cognex couldn't recognize the pattern while RECOG was able to do so. On the other hands, for two other images (18th and 27th images), Cognex outperformed. Therefore, we can say that RECOG and Cognex had almost the same level of flexibility in recognition concerning this pattern. Please check Appendix B to find the comparison between Cognex and RECOG results for a pattern with the same structure as pattern D but on a different layer.

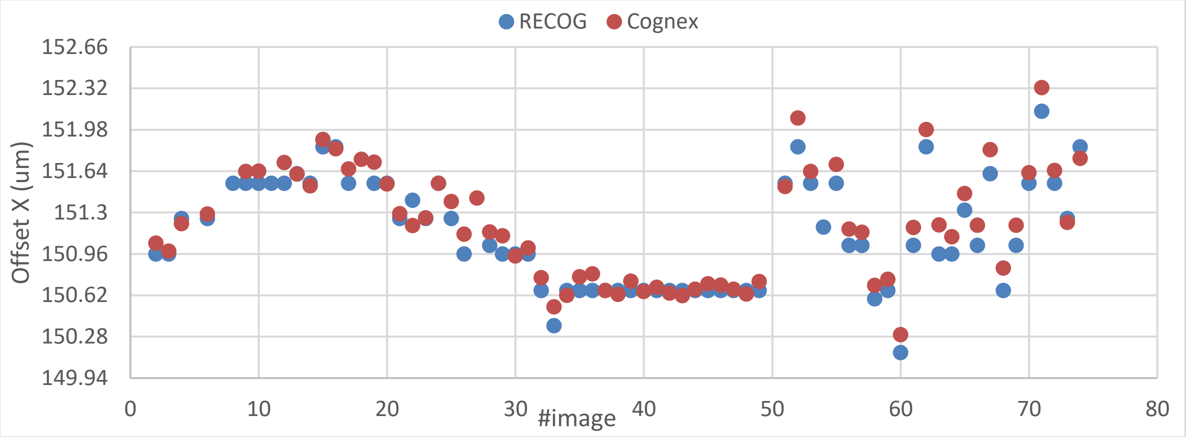


Figure 31. Recognition offset comparison, X-axis, Pattern V

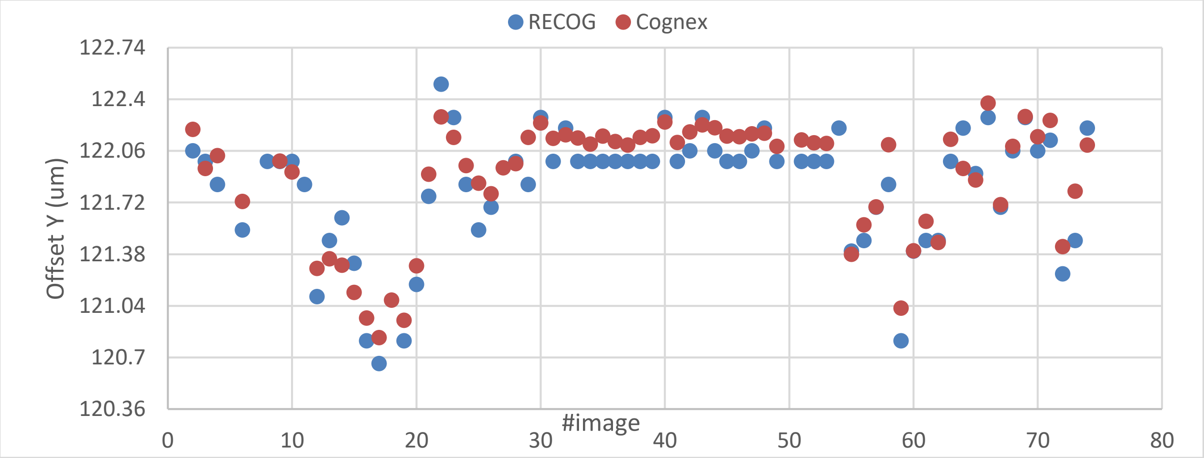


Figure 32. Recognition offset comparison, Y-axis, Pattern V

Testing the integration by running tasks or replays

The wafer alignment could be part of a procedure or a task (standalone executable.) There are some tasks in YS to measure a specific parameter, such as performance. The code execution path for running each task could be different depending on their goal. By running a wafer performance alignment task, we might not be able to verify the integration thoroughly. However, it still could cover a considerable amount of the desired execution path.

Replays are created based on the process jobs executed on the machine, containing the machine images, recipes, configuration files, and the expected results. It is possible to run replays in the simulation mode. Generally, the pattern recognition offset is reported as (0,0) in the simulation mode, which means the pattern is located at the image center. However, we forced YS to use either Cognex or RECOG while running replays to be able to compare their results. After we made sure that the performance and accuracy of the results are acceptable and we could run wafer alignment without the Cognex dongle (using the RECOG library,) we initiated testing on the machine. As we ran all these tests on the machine, we skip the results of the tests on the local machine and instead focus on the machine test results.

10.3.2. Test on machine

The main difference between testing on the actual machine and the previous tests is that the images are captured on the fly on the machine and all other enhancements on the image. We can categorize the machine tests into three groups:

- Running a task
- Running a process job (PJ)
- Running a series of sequential process jobs (PJs)

Since the definition and procedure are similar for all three, we are not going into the details of each category. The goal behind running the third test was multifold. First, we could make sure that the execution would not break while running a sequence of PJs. Second, to measure Key Performance Indicators (KPIs) with respect to the RECOG library, we needed to have a series of the result. There are some predefined KPIs for YS, and there are some tasks that simplify measuring them. For each test category, we first ran the test using RECOG and then repeated it using Cognex. Then, we drew a comparison between Cognex and RECOG results. Before initiating tests on the machine, a few preparation steps were required. More information is given in Appendix B.

Analysis of the result of running a task

For running a task, first, we set a proper wafer on the stage and then executed the performance task. Without reloading the wafer, we repeated the same task using Cognex. As we used the same wafer without reloading it, then it made sense to compare position offset measured using these two libraries. Figure 33 and Figure 34 represent the comparison between the offset values measured by Cognex and RECOG per each mark in X and Y axes, respectively.

The unit size is 0.33 μm, which is roughly equal to one pixel. As it shows, almost always, the difference between the measured values by Cognex and RECOG is less than one pixel. Please find the result of the same comparison with 0-degree rotation in Appendix B. These results show that RECOG accuracy meets the expected requirement.

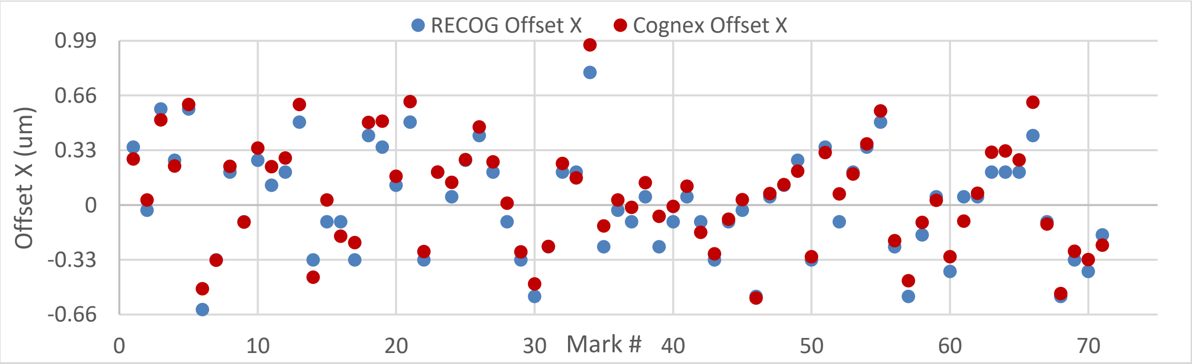


Figure 33. Cognex and RECOG offset comparison, X-axis with 180 degree

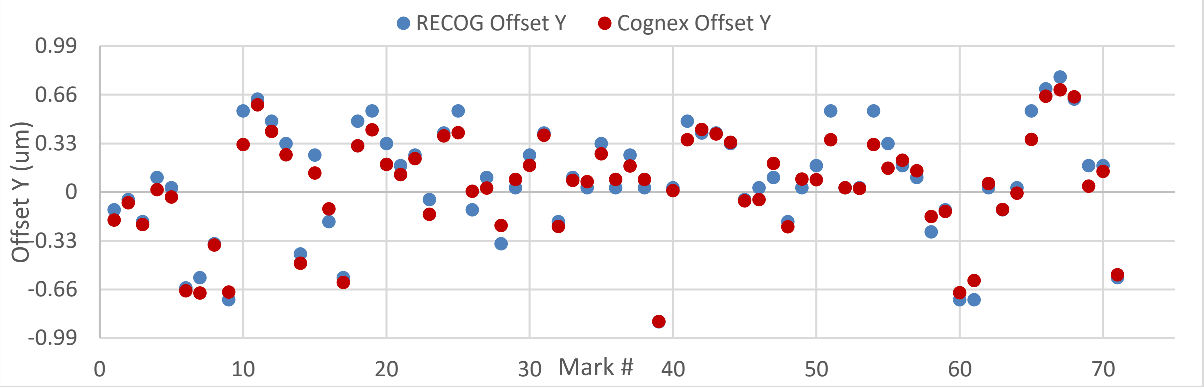


Figure 34. Cognex and RECOG offset comparison, Y-axis with 180 degree

Analysis of the result of running process jobs

Looking at only one run result might not reflect the performance and accuracy of the libraries thoroughly. By running a sequence of PJs, we can get more information, including the mean alignment time and the wafer alignment throughput, indicating the number of wafers that can be measured per hour.

Each time a process job gets started, the wafer is reloaded, which means that it might be loaded differently each time (like with a different orientation.) Consequently, the wafer model might change, and there is no guarantee that the measured positions remain the same for different runs. It means that the target position offsets (TPOs) reported after running a PJ are not comparable, even if we run a PJ twice using the same library. However, based on the YS specifications, TPO should always be less than 2.5 μm. TPO shows the offset of overlay targets from the spot center, and its value depends on the wafer alignment result.

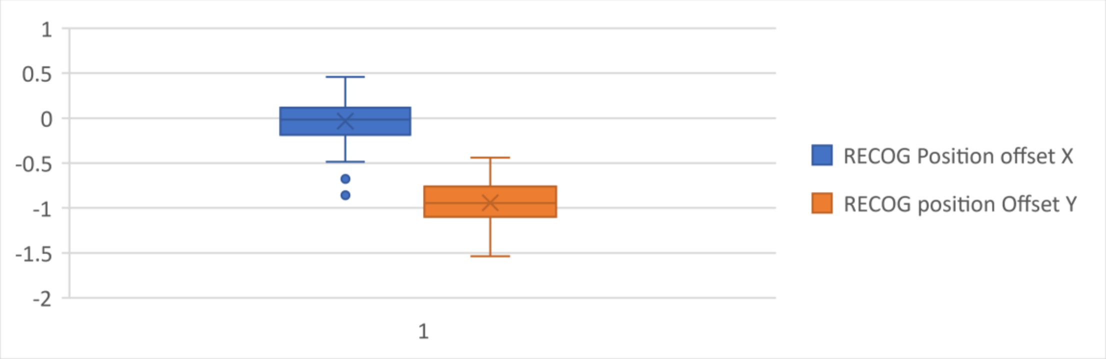


Figure 35. RECOG TPO value range in X and Y axes

Figure 35 illustrates the range of TPO values measured using RECOG on the X and Y axes. The values are less than 2.5 μm. Looking at the same values measured using Cognex, almost the same difference in both axes can be observed. It implies that the difference between the X and Y values might be related to a systematic error.

YS is a metrology machine measuring some quality metrics and giving feedback to TWINSCAN. OV offset values are one of the interesting metrics that define the scanner error offset. It shows how well a layer is printed on top of the other one. As discussed, YS should know the precise location of the target for measuring OV. Accurate calculation of this position is highly dependent on the accuracy and precision of the wafer alignment, i.e., pattern recognition result. We expect to get almost the same OV values for all targets with the same wafer, irrespective of the wafer alignment configuration or the pattern recognition library. While Figure 36 compares the measured OV values in the X-axis per target calculated using both libraries, Figure 37 compares the Y overlay values.

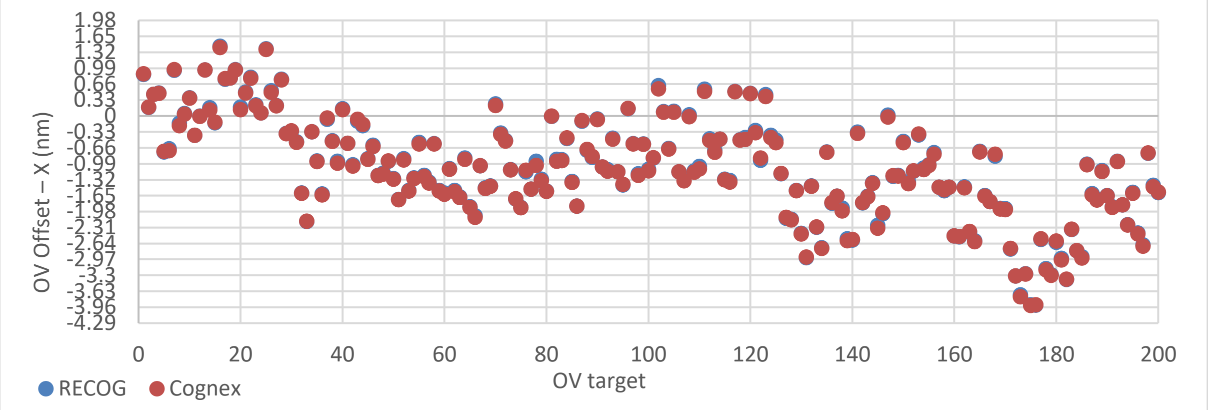


Figure 36. OV offset comparison, X-axis

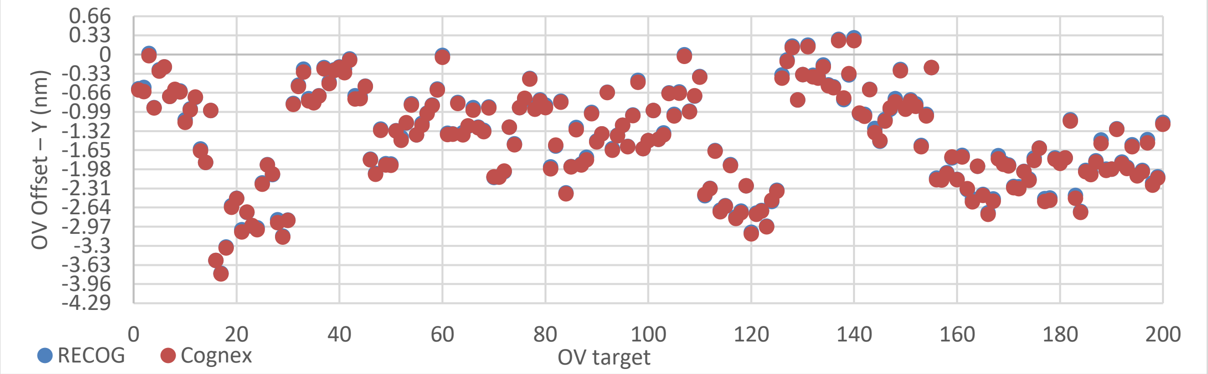


Figure 37. OV offset comparison, Y-axis

As it shows, the difference between their results is negligible. Figure 38 depicts the significant difference value in the X and Y axes. The difference is mainly between 0.01 and 0.03 nanometers and based on the specifications, the difference should be less than 0.07 nanometers.

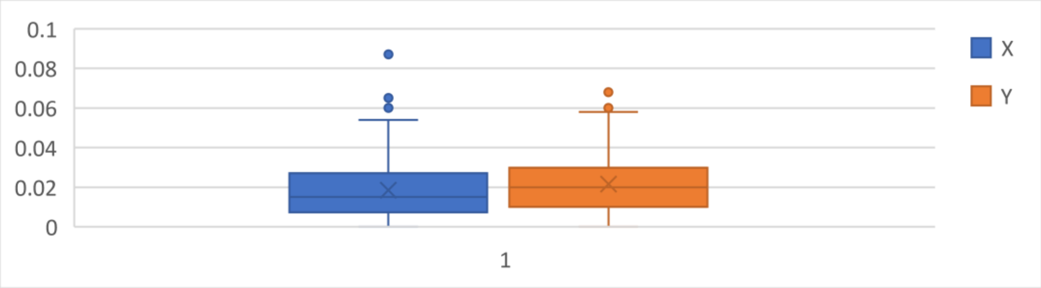


Figure 38. Differences between OV X and Y offsets calculated by RECOG and Cognex (nm)

Total Measurement Uncertainty

Total Measurement Uncertainty (TMU) is one of the significant YS KPIs. It shows the stability and precision of the YS results. Any instability in the pattern recognition result would be reflected on TMU as well. The idea is to find the level of uncertainty of the results between different runs on the same machine and with the same set of configurations.

Table 8 and Table 9 show the TMU results measured against the RECOG and Cognex libraries correspondingly. According to the YS test specifications, TMU in the X and Y direction should be less than 0.22 nm. The results confirm that not only the RECOG TMU values satisfy the specifications, but also, they are relatively close to the Cognex results.

Table 8. RECOG TMU result

ATP Calculations				
Calculation	**Machine ID**	**Recipe**	**Value**	**Result [nm]**
OV_TMU Stability	M1	Recipe A	TMU X (C10)	0.104
OV_TMU Stability	M1	Recipe A	TMU Y (C10)	0.113
OV_ TMU Stability	M1	Recipe A	Stability X (C10)	0.093
OV_ TMU Stability	M1	Recipe A	Stability Y (C10)	0.099

Table 9.Cognex TMU result

ATP Calculations				
Calculation	**Machine ID**	**Recipe**	**Value**	**Result [nm]**
OV_TMU Stability	M1	Recipe A	TMU X (C10)	0.099
OV_TMU Stability	M1	Recipe A	TMU Y (C10)	0.112
OV_ TMU Stability	M1	Recipe A	Stability X (C10)	0.089
OV_ TMU Stability	M1	Recipe A	Stability Y (C10)	0.095

Matching

Matching reflects the accuracy of the results, and it is one of the critical YS KPIs. Usually, matching is measure by running a series of PJs on different machines to determine how the results are matched. In this experiment, we repeated PJs on the same machine but with different configurations, i.e., different pattern recognition libraries for the wafer alignment. According to the YS specifications, the difference in the mean values (both in x and y axes) should be less than 0.07 nm, and the three-sigma value, which is another statistical metric, should be less than 0.17 nm. Table 10 shows the matching result. In this test, machine contribution is not considered; therefore, the values should be less than the specifications, which is the case.

Table 10. Matching result

ATP Calculations				
Calculation	**Machine ID**	**Recipe**	**Value**	**Result [nm]**
OV_Matching	M1	Recipe A	Mean X	-0.001
OV_Matching	M1	Recipe A	Mean Y	-0.001
OV_Matching	M1	Recipe A	3 Sigma X	0.035
OV_Matching	M1	Recipe A	3 Sigma Y	0.031

Throughput

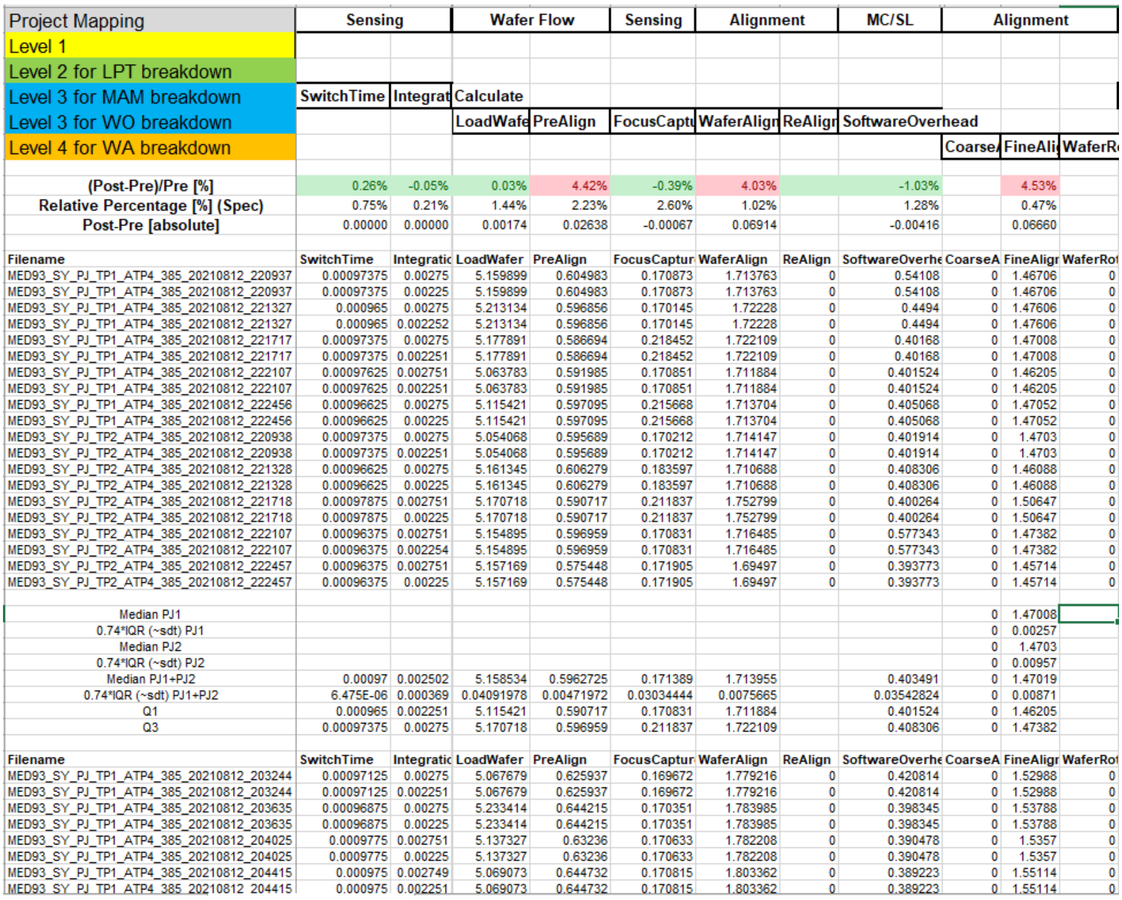
Throughput is one of the essential KPIs that shows the number of wafers that can be measured per hour. We created a process job based on the throughput recipe and repeated this job five times by both libraries. The first experiment was done using RECOG, and the second one was based on the Cognex library.

Table 11. Throughput result

KPI	RECOG	Cognex	Delta
uDBO 1WL Wafer Throughput (Level 1)	84.86	85.77	1.06%
uDBO 1WL Process Job Overhead (Level 2)	3.447	3.268	6.01%
uDBO 1WL Wafer Overhead (Level 2)	8.292	8.07	2.67%

Table 11 summarizes the result. While RECOG can measure 84.86 recipes per hour, Cognex can measure 85.77 wafers.

Table 12. Time overhead comparison between RECOG and Cognex



Project Mapping	Sensing		Wafer Flow		Sensing	Alignment		MC/SL	Alignment		
Level 1											
Level 2 for LPT breakdown											
Level 3 for MAM breakdown	SwitchTime	Integrat	Calculate								
Level 3 for WO breakdown			LoadWafe	PreAlign	FocusCaptu	WaferAlign	ReAlign	SoftwareOverhead			
Level 4 for WA breakdown									Coarse/	FineAli	WaferR
(Post-Pre)/Pre [%]	0.26%	-0.05%	0.03%	4.42%	-0.39%	4.03%		-1.03%		4.53%	
Relative Percentage [%] (Spec)	0.75%	0.21%	1.44%	2.23%	2.60%	1.02%		1.28%		0.47%	
Post-Pre [absolute]	0.00000	0.00000	0.00174	0.02638	-0.00067	0.06914		-0.00416		0.06660	
Filename	SwitchTime	Integratic	LoadWafer	PreAlign	FocusCaptur	WaferAlign	ReAlign	SoftwareOverhe	CoarseA	FineAligr	WaferRot
MED93_SY_PJ_TP1_ATP4_385_20210812_220937	0.00097375	0.00275	5.159899	0.604983	0.170873	1.713763	0	0.54108	0	1.46706	0
MED93_SY_PJ_TP1_ATP4_385_20210812_220937	0.00097375	0.00225	5.159899	0.604983	0.170873	1.713763	0	0.54108	0	1.46706	0
MED93_SY_PJ_TP1_ATP4_385_20210812_221327	0.000965	0.00275	5.213134	0.596856	0.170145	1.72228	0	0.4494	0	1.47606	0
MED93_SY_PJ_TP1_ATP4_385_20210812_221327	0.000965	0.002252	5.213134	0.596856	0.170145	1.72228	0	0.4494	0	1.47606	0
MED93_SY_PJ_TP1_ATP4_385_20210812_221717	0.00097375	0.00275	5.177891	0.586694	0.218452	1.722109	0	0.40168	0	1.47008	0
MED93_SY_PJ_TP1_ATP4_385_20210812_221717	0.00097375	0.002251	5.177891	0.586694	0.218452	1.722109	0	0.40168	0	1.47008	0
MED93_SY_PJ_TP1_ATP4_385_20210812_222107	0.00097625	0.002751	5.063783	0.591985	0.170851	1.711884	0	0.401524	0	1.46205	0
MED93_SY_PJ_TP1_ATP4_385_20210812_222107	0.00097625	0.002251	5.063783	0.591985	0.170851	1.711884	0	0.401524	0	1.46205	0
MED93_SY_PJ_TP1_ATP4_385_20210812_222456	0.00096625	0.00275	5.115421	0.597095	0.215668	1.713704	0	0.405068	0	1.47052	0
MED93_SY_PJ_TP1_ATP4_385_20210812_222456	0.00096625	0.00225	5.115421	0.597095	0.215668	1.713704	0	0.405068	0	1.47052	0
MED93_SY_PJ_TP2_ATP4_385_20210812_220938	0.00097375	0.00275	5.054068	0.595689	0.170212	1.714147	0	0.401914	0	1.4703	0
MED93_SY_PJ_TP2_ATP4_385_20210812_220938	0.00097375	0.002251	5.054068	0.595689	0.170212	1.714147	0	0.401914	0	1.4703	0
MED93_SY_PJ_TP2_ATP4_385_20210812_221328	0.00096625	0.00275	5.161345	0.606279	0.183597	1.710688	0	0.408306	0	1.46088	0
MED93_SY_PJ_TP2_ATP4_385_20210812_221328	0.00096625	0.00225	5.161345	0.606279	0.183597	1.710688	0	0.408306	0	1.46088	0
MED93_SY_PJ_TP2_ATP4_385_20210812_221718	0.00097875	0.002751	5.170718	0.590717	0.211837	1.752799	0	0.400264	0	1.50647	0
MED93_SY_PJ_TP2_ATP4_385_20210812_221718	0.00097875	0.00225	5.170718	0.590717	0.211837	1.752799	0	0.400264	0	1.50647	0
MED93_SY_PJ_TP2_ATP4_385_20210812_222107	0.00096375	0.002751	5.154895	0.596959	0.170831	1.716485	0	0.577343	0	1.47382	0
MED93_SY_PJ_TP2_ATP4_385_20210812_222107	0.00096375	0.002254	5.154895	0.596959	0.170831	1.716485	0	0.577343	0	1.47382	0
MED93_SY_PJ_TP2_ATP4_385_20210812_222457	0.00096375	0.002751	5.157169	0.575448	0.171905	1.69497	0	0.393773	0	1.45714	0
MED93_SY_PJ_TP2_ATP4_385_20210812_222457	0.00096375	0.00225	5.157169	0.575448	0.171905	1.69497	0	0.393773	0	1.45714	0
Median PJ1									0	1.47008	
0.74*IQR (~sdt) PJ1									0	0.00257	
Median PJ2									0	1.4703	
0.74*IQR (~sdt) PJ2									0	0.00957	
Median PJ1+PJ2	0.00097	0.002502	5.158534	0.5962725	0.171389	1.713955		0.403491	0	1.47019	
0.74*IQR (~sdt) PJ1+PJ2	6.475E-06	0.000369	0.04091978	0.00471972	0.03034444	0.0075665		0.03542824	0	0.00871	
Q1	0.000965	0.002251	5.115421	0.590717	0.170831	1.711884		0.401524	0	1.46205	
Q3	0.00097375	0.00275	5.170718	0.596959	0.211837	1.722109		0.408306	0	1.47382	
Filename	SwitchTime	Integratic	LoadWafer	PreAlign	FocusCaptur	WaferAlign	ReAlign	SoftwareOverhe	CoarseA	FineAligr	WaferRot
MED93_SY_PJ_TP1_ATP4_385_20210812_203244	0.00097125	0.00275	5.067679	0.625937	0.169672	1.779216	0	0.420814	0	1.52988	0
MED93_SY_PJ_TP1_ATP4_385_20210812_203244	0.00097125	0.002251	5.067679	0.625937	0.169672	1.779216	0	0.420814	0	1.52988	0
MED93_SY_PJ_TP1_ATP4_385_20210812_203635	0.00096875	0.00275	5.233414	0.644215	0.170351	1.783985	0	0.398345	0	1.53788	0
MED93_SY_PJ_TP1_ATP4_385_20210812_203635	0.00096875	0.00225	5.233414	0.644215	0.170351	1.783985	0	0.398345	0	1.53788	0
MED93_SY_PJ_TP1_ATP4_385_20210812_204025	0.0009775	0.002751	5.137327	0.63236	0.170633	1.782208	0	0.390478	0	1.5357	0
MED93_SY_PJ_TP1_ATP4_385_20210812_204025	0.0009775	0.00225	5.137327	0.63236	0.170633	1.782208	0	0.390478	0	1.5357	0
MED93_SY_PJ_TP1_ATP4_385_20210812_204415	0.000975	0.002749	5.069073	0.644732	0.170815	1.803362	0	0.389223	0	1.55114	0
MED93_SY_PJ_TP1_ATP4_385_20210812_204415	0.000975	0.002251	5.069073	0.644732	0.170815	1.803362	0	0.389223	0	1.55114	0

Table 12 reveals more details on the elapsed time per step during wafer measurement. We can see that latency is related to the alignment phase and the Fine alignment (as we did a fast alignment.) It shows that RECOG is roughly 4 percent (0.06 seconds) slower than Cognex. While the Cognex Fine alignment average time was about 1.47 seconds, RECOG required about 1.54 seconds to do the same job. The maximum accepted threshold for fast alignment is two seconds, while RECOG accomplishes this task in about 1.785 seconds. Note that the average pattern recognition time using Cognex is about 30 milliseconds, while it is about 50 milliseconds using RECOG. Also, we can see that there is a latency of roughly 4 percent in the PreAlign phase. However, the duration of PreAlign depends on the rotation of the incoming wafer, and it is not related to the pattern recognition solution. Therefore, this difference should be ignored.

11. Project Management

Project management is one of the key factors in project success. It aims to define the project roadmap and strategies to tackle difficulties and risks within the project, facilitating the achievement of the project goal within the given constraints. In this project, project management was done using earned-value analysis. During the first weeks of the project, the main milestones were determined, and later on, during each iteration, more detailed tasks were planned. The following sections explain the way of working, project planning, and risk management in this procedure.

11.1 Work-Breakdown Structure

This section explains the Work-Breakdown structure (WBS) of the project. The project had five major phases: exploration, research and literature study, design and implementation, verification and validation, and finally, project closure. Figure 39 depicts all these phases with more details. A deadline was defined for each phase to make sure the project could be completed on time.

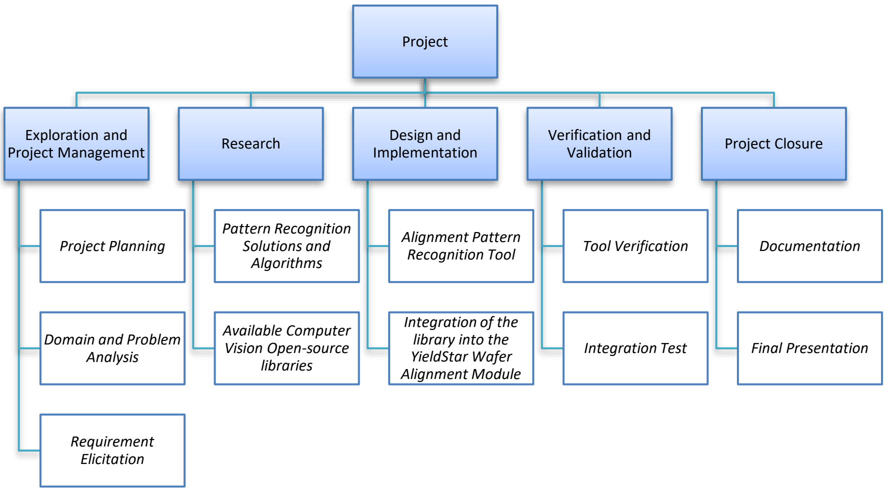

Project
Exploration and Project Management
Research
Design and Implementation
Verification and Validation
Project Closure
Project Planning
Domain and Problem Analysis
Requirement Elicitation
Pattern Recognition Solutions and Algorithms
Available Computer Vision Open-source libraries
Alignment Pattern Recognition Tool
Integration of the library into the YieldStar Wafer Alignment Module
Tool Verification
Integration Test
Documentation
Final Presentation

Figure 39. Project Work Breakdown Structure

11.2 Way of Working

During the initial meetings, the trainee defined the means of communication and the communication frequency with each stakeholder depending on their interest and contribution to the project. Communications with almost all of the stakeholders were planned mainly by Microsoft Teams or via email. The communication frequencies were as follows:

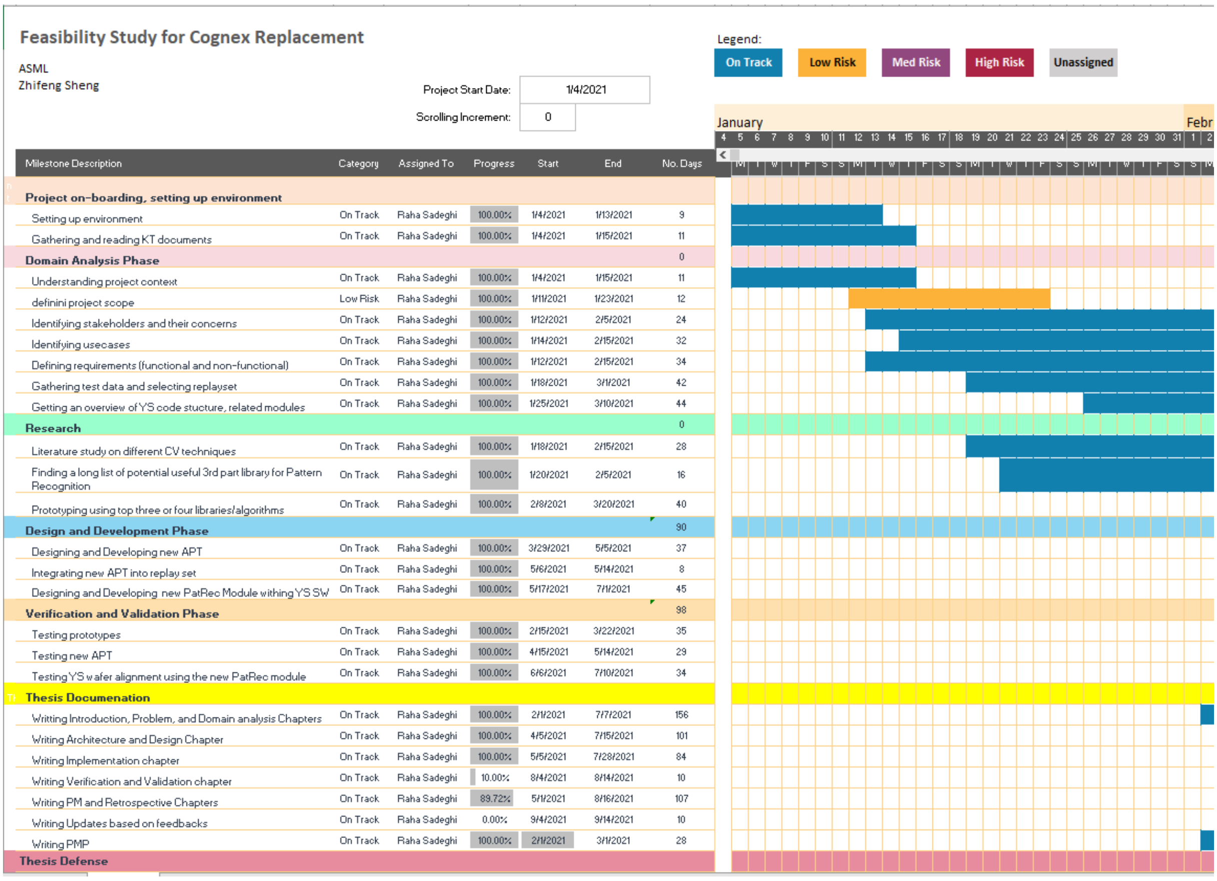
1. The ASML supervisor was updated twice a week during the first three months of the project to make sure the project was going in the right direction. The meeting schedule changed to mostly once a week during the second half of the project. In case of any obstacles, needs, or achievements, an ad-hoc meeting was planned. Also, he was present in the PSG meetings held once a month.
2. Ad-hoc meetings were planned with the main stakeholders upon a need to get the required information. Additionally, mainly at the end of each sprint or upon gaining an achievement, a demo presentation was held to keep them updated about the project progress and get early feedback. It helped in defining the following sprint stories.
3. The TU/e supervisor participated in the monthly PSG meetings to get informed about the project progress. He was informed about changes or obstacles during the project. At the start of the project, bi-weekly meetings were planned to be held if there was a need to help.

11.2.1. *Project Planning and Scheduling*

One of the first and essential steps in each project is to define the project scope. The project's scope was determined by considering limited resources (only one person and a ten-month project) to make sure the goals were achievable. Additionally, main milestones were discussed that helped with making the project clear. The project plan and its main stakeholders were discussed with the ASML supervisor.

Agile methodology was used during this project. Work was completed in sprints that had a length of two weeks. There was a total of nineteen sprints. Each sprint had a starting date, an ending date, and goals corresponding to deliverables. Due to PDEng related activities that simultaneously occurred with this project, each sprint had different working days per person. This period corresponded to the number of hours the trainee could work on the project divided by the number of working hours per day the trainee worked.

Apart from the WBS, a Gantt chart was created to track the project progress. All milestones were reflected in the Gantt chart with more details, and a deadline was assigned to each step. It was mainly used for tracking the project. Figure 40 depicts the summary of tasks in this project created using Microsoft Excel. The chart was also shared with the ASML supervisor.



Feasibility Study for Cognex Replacement

ASML
Zhifeng Sheng

Project Start Date: 1/4/2021
Scrolling Increment: 0

Milestone Description	Category	Assigned To	Progress	Start	End	No. Days
Project on-boarding, setting up environment						
Setting up environment	On Track	Raha Sadeghi	100.00%	1/4/2021	1/13/2021	9
Gathering and reading KT documents	On Track	Raha Sadeghi	100.00%	1/4/2021	1/15/2021	11
Domain Analysis Phase						0
Understanding project context	On Track	Raha Sadeghi	100.00%	1/4/2021	1/15/2021	11
definini project scope	Low Risk	Raha Sadeghi	100.00%	1/11/2021	1/23/2021	12
Identifying stakeholders and their concerns	On Track	Raha Sadeghi	100.00%	1/12/2021	2/5/2021	24
Identifying usecases	On Track	Raha Sadeghi	100.00%	1/14/2021	2/15/2021	32
Defining requirements (functional and non-functional)	On Track	Raha Sadeghi	100.00%	1/12/2021	2/15/2021	34
Gathering test data and selecting replayset	On Track	Raha Sadeghi	100.00%	1/18/2021	3/1/2021	42
Getting an overview of YS code stucture, related modules	On Track	Raha Sadeghi	100.00%	1/25/2021	3/10/2021	44
Research						0
Literature study on different CV techniques	On Track	Raha Sadeghi	100.00%	1/18/2021	2/15/2021	28
Finding a long list of potential useful 3rd part library for Pattern Recognition	On Track	Raha Sadeghi	100.00%	1/20/2021	2/5/2021	16
Prototyping using top three or four libraries/algorithms	On Track	Raha Sadeghi	100.00%	2/8/2021	3/20/2021	40
Design and Development Phase						90
Designing and Developing new APT	On Track	Raha Sadeghi	100.00%	3/29/2021	5/5/2021	37
Integrating new APT into replay set	On Track	Raha Sadeghi	100.00%	5/6/2021	5/14/2021	8
Designing and Developing new PatRec Module withing YS SW	On Track	Raha Sadeghi	100.00%	5/17/2021	7/1/2021	45
Verification and Validation Phase						98
Testing prototypes	On Track	Raha Sadeghi	100.00%	2/15/2021	3/22/2021	35
Testing new APT	On Track	Raha Sadeghi	100.00%	4/15/2021	5/14/2021	29
Testing YS wafer alignment using the new PatRec module	On Track	Raha Sadeghi	100.00%	6/6/2021	7/10/2021	34
Thesis Documenation						
Writting Introduction, Problem, and Domain analysis Chapters	On Track	Raha Sadeghi	100.00%	2/1/2021	7/7/2021	156
Writing Architecture and Design Chapter	On Track	Raha Sadeghi	100.00%	4/5/2021	7/15/2021	101
Writing Implementation chapter	On Track	Raha Sadeghi	100.00%	5/5/2021	7/28/2021	84
Writing Verification and Validation chapter	On Track	Raha Sadeghi	10.00%	8/4/2021	8/14/2021	10
Writing PM and Retrospective Chapters	On Track	Raha Sadeghi	89.72%	5/1/2021	8/16/2021	107
Writing Updates based on feedbacks	On Track	Raha Sadeghi	0.00%	9/4/2021	9/14/2021	10
Writing PMP	On Track	Raha Sadeghi	100.00%	2/1/2021	3/1/2021	28
Thesis Defense						

Figure 40. Project Gantt chart

The project plan was also reflected on the scrum board. All sprints were created during the first month of the project, and all milestones in the form of more detailed user stories were added to the board and its relevant sprint. It facilitated checking the project status both for the trainee and the ASML supervisor. Additionally, it helped in the planning of each sprint, as the goals were defined ahead.

User stories were maintained in the ASML Jira application in a separate scrum board created by the trainee for this specific project. At the start of each sprint, the user stories were defined in more detail and were reviewed by the ASML supervisor to make sure the project was on the right track.

11.2.2. Risk Management

It was essential to recognize the main project risks from the start of the project and try to find some mitigation strategies. Table 13 describes the risks identified in this project, their occurrence possibility, potential impact and result, and the mitigation strategies applied to manage each of them.

Table 13. Risks and their mitigation strategies

Id	***Description***	***Probability***	***Impact***	***Result***	***Mitigation Strategy***
1	Unavailability of the main stakeholders	Low	Medium	Not delivering the desired artifacts or delay for a specific deliverable	- Scheduling meetings early on - Giving regular reports on the progress and impediments.
2	Underestimating the project workload	Likely	Critical	Change in deliverables, delay in project completion, project failure	- Conducting a comprehensive research - Prototyping at early stages - Short loop feedback - Defining the minimum viable product
3	Selecting improper open source library/algorithm that cannot satisfy the requirements	Likely	Critical	Change in deliverables, delay in the project completion.	- Conducting comprehensive research and feasibility study - Prototyping at early stages before design and implementation - Keeping stakeholders in an early feedback loop with a regular demo on achievements or challenges
4	Lack of domain knowledge	High	Medium	Not finding the best solution, not being able to accomplish the project on time	- Pre-study and taking Online courses - Trying to build a network of experts to ask for help
5	Main Stakeholders' resignation or illness (such as Corona)	Likely	Medium	Depending on the role, could lead to the project delay for a different period	- Try to find some alternatives for asking questions and help - Building a broad network at the company - Taking necessary measures stated by the government

12. Conclusions

This chapter summarizes the project achievements and gives some suggestions for future work. Finally, a self-reflection on the project is given by the trainee.

12.1 Results and Remarks

In this project, we aimed to investigate the feasibility of replacing Cognex with a new pattern recognition solution in the context of wafer alignment. The goal was to take advantage of available open-source libraries. Cognex is a mature commercial tool offering several functionalities. Therefore, replacement should be done step by step. In Section 5.3, we defined three main design criteria, which are revisited while looking at the achievements.

Massive chip production and ease of use are two main concerns of ASML customers. ASML's goal is to reduce the cost while keeping its customer satisfied and even grab the attention of more customers. Replacing Cognex with an in-house solution could decrease the cost. Therefore, in this project, as the first try towards Cognex replacement, we focused on studying the feasibility of designing and developing a library with comparable accuracy, performance, and flexibility with Cognex, while supporting the most critical Cognex functionalities. We always kept performance (recognition time) and accuracy as the main priorities for making decisions. In the previous chapters, we explained how we tried to make the best decisions concerning performance. Additionally, we set the goal to design an extensible framework that can be improved with minimal effort in the future. Realization of the solution was the third design criterion, which was met with ART and library integration.

By integrating RECOG into YS and testing the solution on the machine, we demonstrated the feasibility of the solution. Test results were highly promising and proved that RECOG followed the YS architectural requirements. Extensibility of the solution was achieved by designing RECOG and ART based on best design practices and applying design patterns like the strategy pattern or MVVM architectural pattern. Moreover, we increased the maintainability and readability of the code by applying SOLID design principles. To demonstrate the extensibility of the solution, we already proposed two algorithms, and we demonstrated that the recognition or training algorithm could be selected during alignment recipe creation.

RECOG met the YS specifications (such as TMU, matching, and performance) based on the test results. The results gained by RECOG were highly comparable with Cognex results. The difference between the offset values measured by RECOG and Cognex was mainly less than one pixel. Additionally, test results showed that the proposed solution was considerably flexible in pattern recognition against low contrast or noisy images, making it a robust solution.

Although results showed that RECOG needed 4% more time to align wafers than Cognex, the average fast alignment using RECOG is about 1.7 ms, which is below the specifications (2 ms.) Therefore, even without any improvements, the library performance is acceptable. We believe the performance and accuracy can be improved in the future.

ART supports the main functionalities of APT. Its user interface is quite similar to APT. Thus, users can get used to it readily. It has an additional feature for converting available alignment profiles to the new version, smoothing the migration to the new library. It also provides a new view to the users to verify pattern recognition accuracy by aligning the template pattern and the recognized pattern on top of each other.

To sum up, this project proved that replacing Cognex with an in-house solution is feasible and achievable. The result of this project shows that investment in this project can bring added value to ASML. Still, there are some steps towards achieving the main goal, which is replacing Cognex. The following section gives a few suggestions for future work.

12.2 Recommendations and future work

Coming up with a solution comparable with a mature commercial tool only in ten months and one person is a valuable result, and it is highly promising. As a next step, it is recommended that experts at ASML try to utilize ART internally to test RECOG against more data from the customers and find its potential shortcomings. Based on this information, new ideas can be proposed on how and in which way the library could be improved. As mentioned earlier, the developed library and tool can be used as a basis for the replacement of Cognex shortly. Although we demonstrated the feasibility of replacing Cognex, several topics are still not fully addressed in this project or could be improved.

In this project, we tried to address model-based pattern recognition. Since we are converting model-based pattern descriptions to images, we expect that the solution could be readily extended to address the image-based patterns, provided that the edges of the image-based patterns can be detected accurately. Therefore, it might be necessary to consider a precondition for users to take high-contrast, noise-free images for image-based pattern creation. Moreover, currently, ART does not support new model-based pattern creation. Additional APT functionalities which are not implemented in ART can be added to the tool step by step once it is required. We did not consider the possibility of the presence of multiple instances of the same pattern in the input image. Therefore, if someone found it a necessity, then the algorithm should be extended.

Although we integrated the solution into YS to show that RECOG adheres to the YS architecture and test its performance, we did not modify the recipes due to the time limitation. As future work, recipes can be updated to support RECOG objects as well. Additionally, instead of using a configuration file, new equipment constant can be introduced to set the desired pattern recognition library.

As mentioned in the Domain Analysis Chapter, the application of pattern recognition is not confined to the wafer alignment. The possibility of utilizing RECOG in other contexts can be investigated as a next step, and the algorithm could be extended if needed to be used in other domains as well. In this way, YS can utilize a unified pattern recognition solution.

In Chapter 6, we discussed the feasibility of using modern CV techniques. We argued that although they are highly powerful, they might not be the best or the first option for addressing the current problem. The main difference between traditional and modern approaches is that a technique is implemented in the former, while a system is trained in the latter. In the first scenario, there is always a necessity to extract the best features, fine-tune parameters, and in the case of new data, there might be a need to parameter fine-tunning. However, if a detector can be trained properly, it should be able to deal with unforeseen cases. Therefore, it is recommended to study the possibility of applying AI-based techniques based on the proposed solution and framework for future work. We believe that it could help in detecting false positives and detecting patterns in noisy or low contrast images.

12.3 Self-reflection

The project conducted during the past nine months was a precious and fulfilling experience for me. It was a great opportunity to put together the expertise I gained during my previous work experience and the knowledge I acquired in the first year of the PDEng program. The project domain and its related technologies were almost new to me, making it challenging while interesting. The project was a combination of the software design problem and pattern recognition in practice. Without any previous background knowledge about image processing or pattern recognition, the first risk or concern was if I could find the proper solution within a limited time. Therefore, I initiated an exploratory phase, studied the literature, prototyped different techniques, and planned regular demo meetings with stakeholders. The meetings were quite helpful, as I could get early feedback and ensured I was on track. Meanwhile, I could learn more about the domain based on the discussions and result analysis. Although learning and exploring a new topic was interesting enough, investigation and endeavor to find a replacement for a mature commercial tool gave me more motivation.

There is always a risk when starting a project in a new broad domain like lithography. You should always find a balance between diving into details and gaining a high level and abstract knowledge. This project helped me immensely to develop my project management, time management, and risk management skills. Despite other projects, it was almost a solo project that I was mainly responsible for every aspect of it. I had to build a network within different teams. Each project phase had its own requirements and challenges, and I had to communicate with different stakeholders.

Throughout the execution of the project, I developed my technical skills with respect to design and analysis, C# development, and testing. I refactored the design and code several times to increase its quality and make it more extensible. Testing was one of the key phases of this project. Upon a small change, all tests had to be repeated. Finding a balance between achieving high accuracy and performance was challenging. Additionally, identifying expected results and deciding on the testing approach was highly important, as there was no ground truth data. Discussion with stakeholders and learning about their concerns and the available specifications within YS smoothed this phase.

In conclusion, the project was an excellent opportunity to broaden my horizon, challenge myself, and develop my skills. The project was completed on time and met the expectations of the stakeholders, including me.

Abbreviations and Glossary

AHE	Adaptive Histogram Equalization
APT	Alignment Pattern Tool
ART	Alignment Recognition Tool
BCET	Balanced Contrast Enhancement Technique
CD	Critical Dimension
CV	Computer Vision
DBO	Diffraction-Based Overlay
DL	Deep Learning
HE	Histogram Equalization
IC	Integrated Circuit
KPI	Key Performance Indicator
LCP	Litho Computing Platform
MVC	Model View Controller
MVP	Model View Presenter
MVVM	Model View ViewModel
NCC	Normalized Cross-Correlation
OV	Overlay
PDEng	Professional Doctorate in Engineering
PJ	Process Job
PSG	Project Steering Group
ST	Software Technology
TMU	Total Measurement Uncertainty
TPO	Target Position Offset
TU/e	Eindhoven University of Technology
WCS	Wafer Coordinate System
WPF	Windows Presentation Foundation
YS	YieldStar

Bibliography

References

[1] Shantaiya, Sanjivani, K. Verma and K. Mehta. “A Survey on Approaches of Object Detection.” *International Journal of Computer Applications* 65 (2013): 14-20.

[2] Zou, Zhengxia & Shi, Zhenwei & Guo, Yuhong & Ye, Jieping. (2019). Object Detection in 20 Years: A Survey.

[3] O' Mahony, Niall & Campbell, Sean & Carvalho, Anderson & Harapanahalli, Suman & Velasco-Hernandez, Gustavo & Krpalkova, Lenka & Riordan, Daniel & Walsh, Joseph. (2019). Deep Learning vs. Traditional Computer Vision.

[4] Lv, Yong & Feng, Qibo & Qi, Liangyu. (2008). A study of sub-pixel interpolation algorithm in digital speckle correlation method. 10.1117/12.807128.

[5] Lowe, David. (2001). Object Recognition from Local Scale-Invariant Features. Proceedings of the IEEE International Conference on Computer Vision. 2.

[6] Bay, Herbert & Tuytelaars, Tinne & Van Gool, Luc. (2006). SURF: Speeded up robust features. Computer Vision-ECCV 2006. 3951. 404-417. 10.1007/11744023_32.

[7] Rublee, Ethan & Rabaud, Vincent & Konolige, Kurt & Bradski, Gary. (2011). ORB: an efficient alternative to SIFT or SURF. Proceedings of the IEEE International Conference on Computer Vision. 2564-2571. 10.1109/ICCV.2011.6126544.

[8] Template-based versus Feature-based Template Matching. (2019). https://medium.datadriveninvestor.com/template-based-versus-feature-based-template-matching-e6e77b2a3b3a.

[9] Kai Briechle, Uwe D. Hanebeck, "Template matching using fast normalized cross correlation," Proc. SPIE 4387, Optical Pattern Recognition XII, (20 March 2001);

[10] Zitová, Barbara & Flusser, Jan. (2003). Image Registration Methods: A Survey. Image and Vision Computing. 21. 977-1000. 10.1016/S0262-8856(03)00137-9.

[11] The Model-View-Presenter (MVP) Pattern. (2010). https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff649571(v=pandp.10)?redirectedfrom=MSDN,

[12] https://en.wikipedia.org/wiki/Phase_correlation

[13] Hamad, Yousif & Simonov, Konstantin & Naeem, Mohammad. (2018). Brain's Tumor Edge Detection on Low Contrast Medical Images. 45-50. 10.1109/AiCIS.2018.00021.

[14] Pattern Recognition. (2021). https://en.wikipedia.org/wiki/Pattern_recognition

[15] Image Registration. (2021). https://en.wikipedia.org/wiki/Image_registration

[16] Kruchten, Philippe (1995, November). Architectural Blueprints — The “4+1” View Model of Software Architecture. IEEE Software 12 (6), pp. 42-50.

[17] MVVM Light Messenger. (2018). http://dotnetpattern.com/mvvm-light-messenger

[18] ISO 9001:2015, "Quality Management Systems - Fundamentals and vocabulary," International Organization for Standardization, Geneva, Switzerland. (2015). www.iso.org

[19] Nunit. (2019). https://nunit.org/.

[20] Introduction to Model/View/ViewModel pattern for building WPF apps. (2005). Introduction to Model/View/ViewModel pattern for building WPF apps | Microsoft Docs.

[21] Patterns - WPF Apps With The Model-View-ViewModel Design Pattern. (2009). Patterns - WPF Apps With The Model-View-ViewModel Design Pattern | Microsoft Docs, MSDN Maganize.

[22] Image Thresholding. (2021). https://docs.opencv.org/4.5.2/d7/d4d/tutorial_py_thresholding.html.

[23] Sarvaiya, Jignesh & Patnaik, Suprava & Bombaywala, Salman. (2009). Image Registration by Template Matching Using Normalized Cross-Correlation. Advances in Computing, Control, and Telecommunication Technologies, International Conference on. 819-822. 10.1109/ACT.2009.207.

[24] Gaussian blur. (2021). https://en.wikipedia.org/wiki/Gaussian_blur#:~:text=In%20image%20processing%2C%20a%20Gaussian,image%20noise%20and%20reduce%20detail.

[25] Lin, Rey-Sern. (2008). Edge Detection by Morphological Operations and Fuzzy Reasoning. Proceedings - 1st International Congress on Image and Signal Processing, CISP 2008. 2. 729 - 733. 10.1109/CISP.2008.346.

[26] Dilation (morphology). (2020.) https://en.wikipedia.org/wiki/Dilation_(morphology)

[27] Template Matching. (2020.) https://docs.opencv.org/4.5.1/d4/dc6/tutorial_py_template_matching.html

[28] Image Thresholding. (2020.) https://docs.opencv.org/4.5.2/d7/d4d/tutorial_py_thresholding.html

[29] Measuring Accurcy. (2019.) https://www.asml.com/en/technology/lithography-principles/measuring-accuracy

[30] Srivastava, Shrey & Divekar, Amit & Anilkumar, Chandu & Naik, Ishika & Kulkarni, Ved & Pattabiraman, V.. (2021). Comparative analysis of deep learning image detection algorithms. Journal of Big Data. 8. 10.1186/s40537-021-00434-w.

[31] MoSCoW method. (2021.) https://en.wikipedia.org/wiki/MoSCoW_method

[32] Diffraction grating. (2021.) https://en.wikipedia.org/wiki/Diffraction_grating

Appendix A. Architecture and Design

This appendix mainly discusses some additional information regarding the design and architecture of the solution. First, the domain is given to summarize the result of domain analysis. We covered the logical view of the architecture in Chapter 8. In this section, we show the big picture of the RECOG and ART class diagrams. As sequence diagrams could give too much details about the implementation phase, we included them in this appendix. Finally, in the last section of this appendix, we listed some additional design decisions to represent the rationale behind these decisions.

Domain model

Figure 41 depicts the domain model of the YS wafer alignment. It summarizes concepts covered in Chapter 4 and represents how they are related.

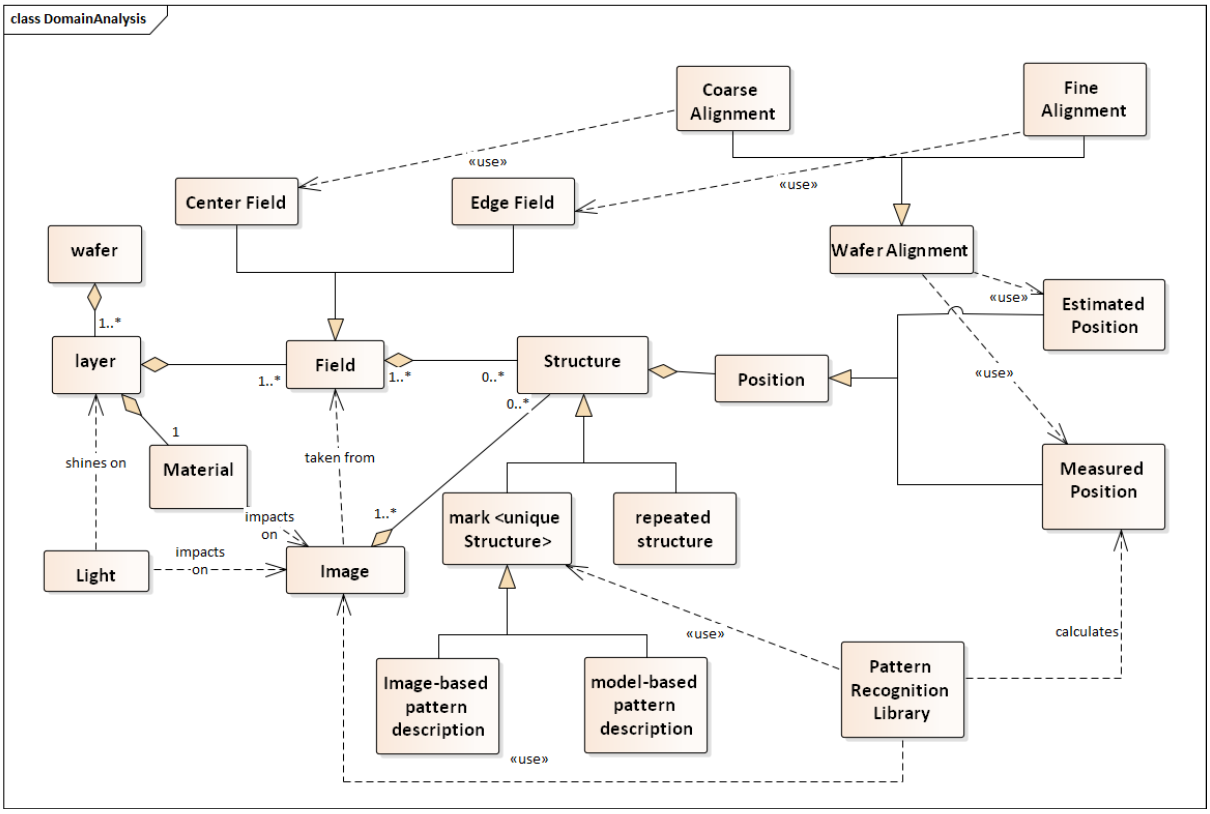


Figure 41. Wafer alignment domain model

Logical View

Figure 42 Depicts the RECOG class design.

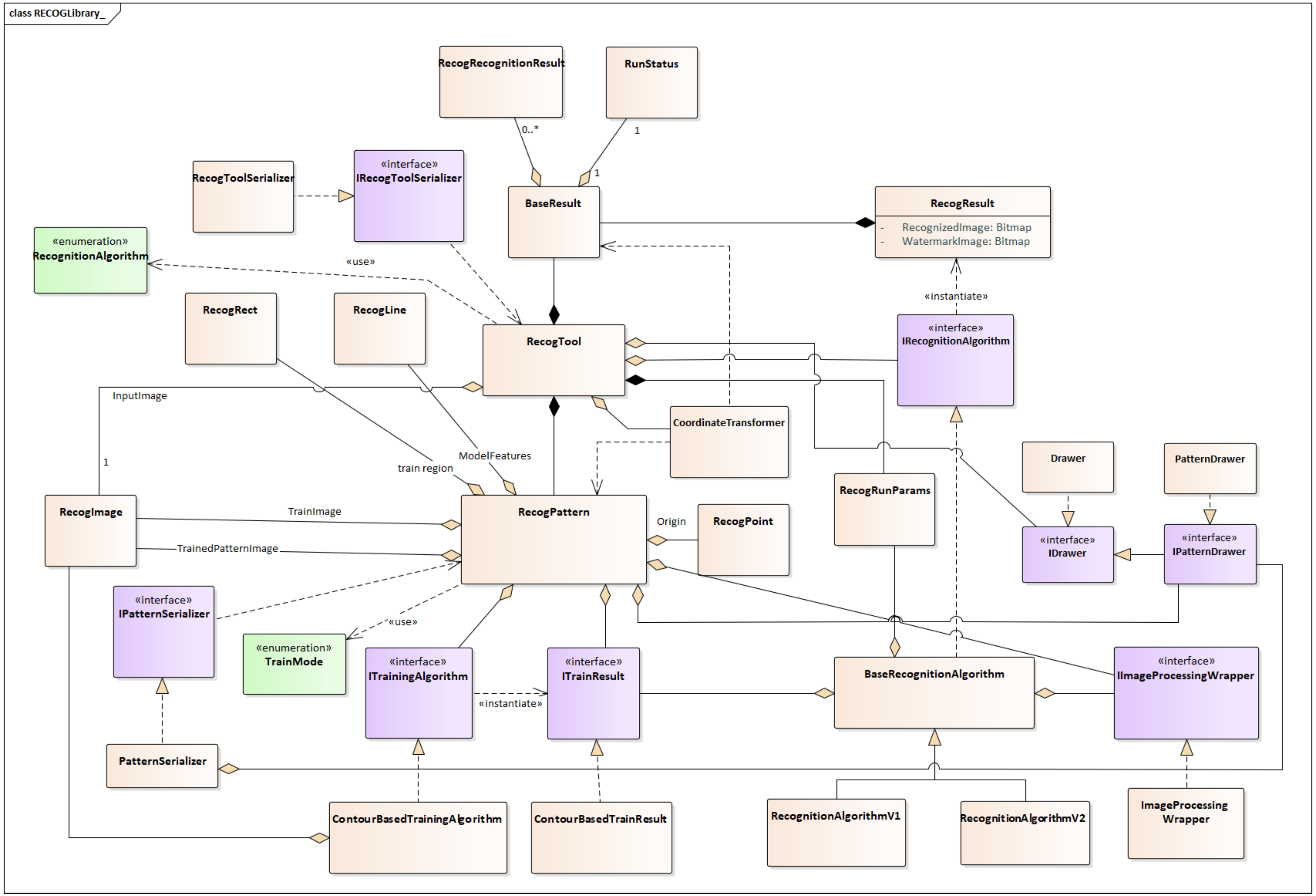


Figure 42. RECOG class diagram

Figure 43 depicts a more detailed view of the ART design. For the sake of simplicity, Views are not presented in the class diagram. ViewModels hold the state of the data in the model, and the business logic is mainly implemented in the model, the RECOG library, in this case. Batch recognition and recipe conversion logic are implemented in the separated classes, RecipeConvertor and BatchRecognizer, respectively.

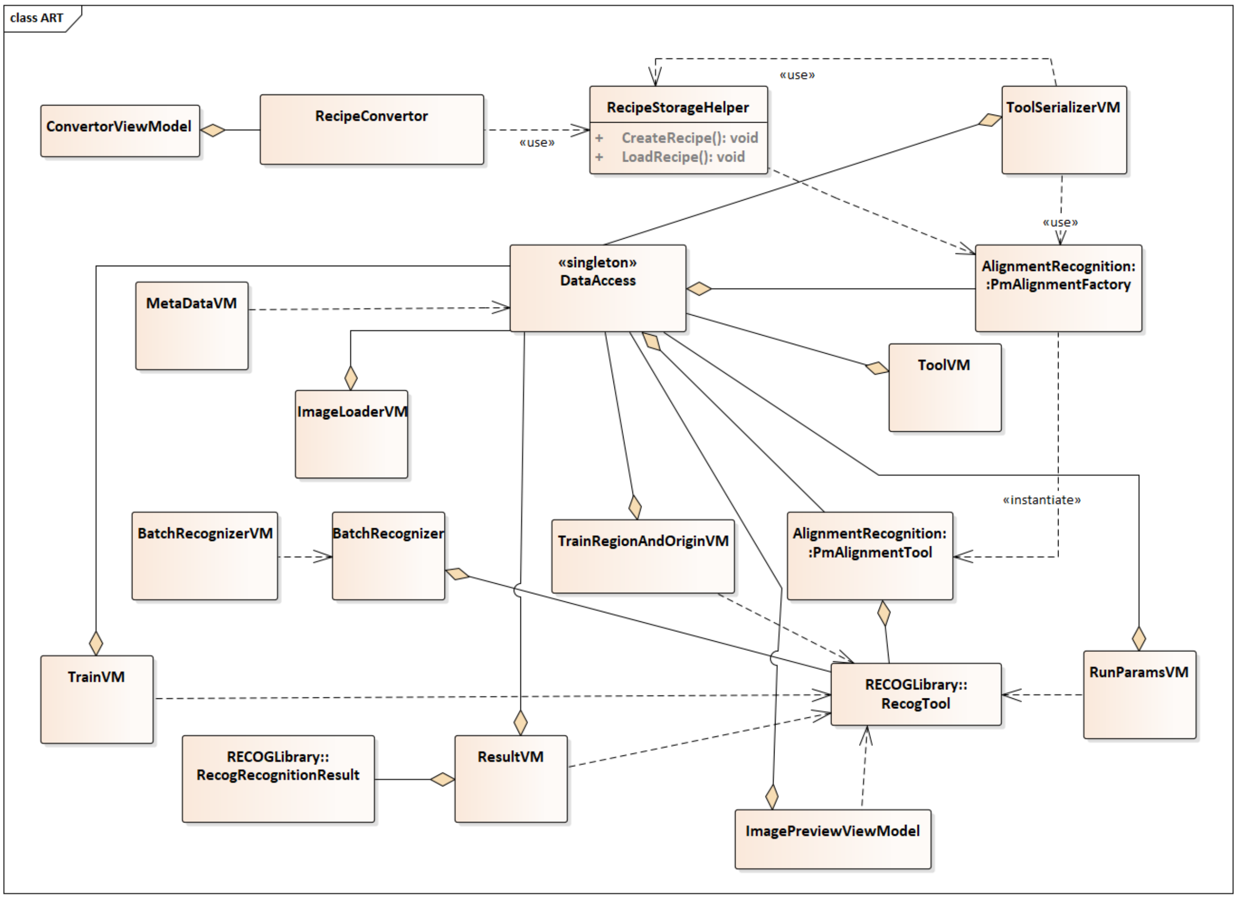


Figure 43. ART class diagram

Process View

As discussed in Chapter 6, we discussed that the process view deals with the dynamic aspects of the system, explains the system processes and how they communicate and focuses on the system run-time behavior [16]. Chapter 7 covers the main part of the process view. We also reviewed the logical view in the Architecture and Design chapter; this section mainly demonstrates the object interactions in run time using sequence diagrams. Figure 44 shows the training sequence diagram, focusing solely on the main run-time interactions during this phase.

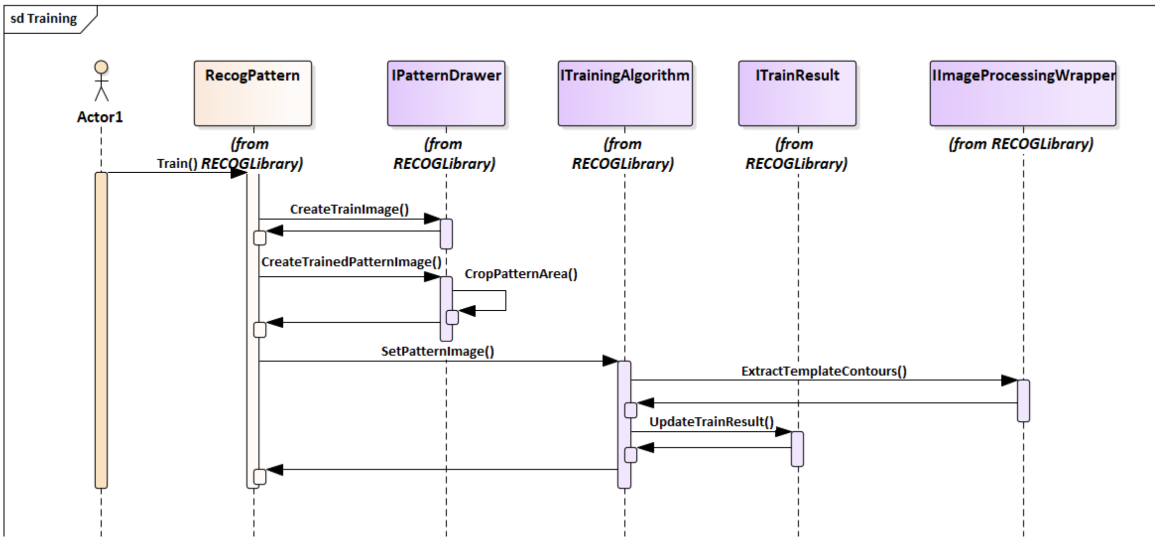


Figure 44. Training procedure

Figure 45 illustrates the sequence diagram of the pattern recognition procedure. In this diagram, details of the training sequence or iterative runs are excluded. The main goal is to show the interaction of the most significant classes and interfaces in the recognition procedure. With the same rationale with the training phase, concrete classes are not shown in the below sequence diagram to generalized it.

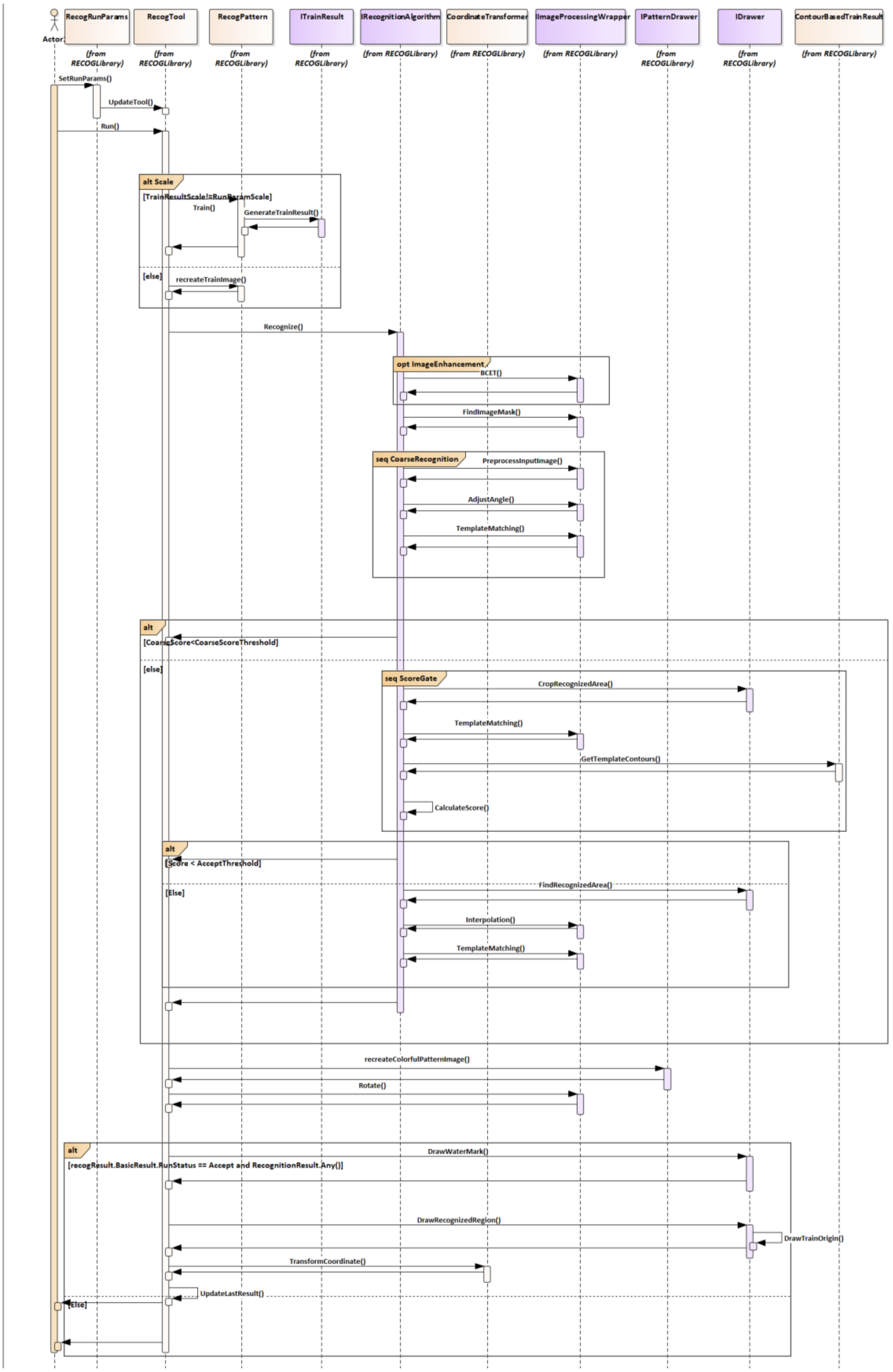


Figure 45. Pattern recognition procedure

Recipe Conversion

Addressing the need for the reusability of the available Cognex-based recipes or alignment profiles, ART has a feature that allows users to convert the available recipes to the new format. Figure 46 illustrates the sequence diagram of the recipe conversion. It is important to note that the Cognex dongle is required to deserialize the Cognex object for conversion.

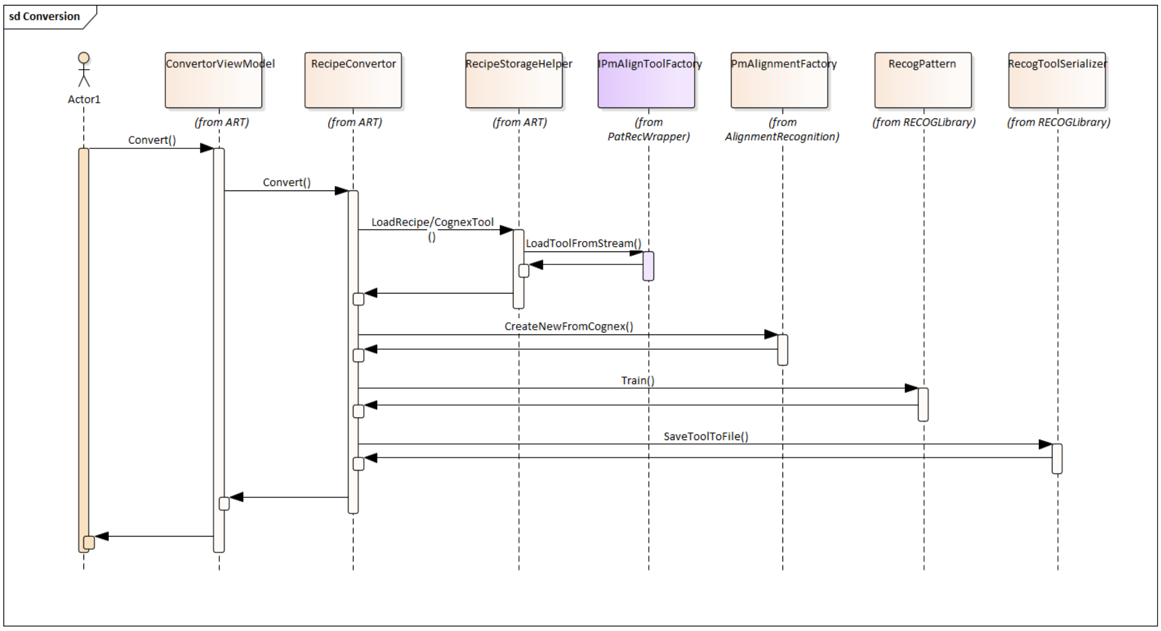


Figure 46. Conversion sequence diagram

In Section 7.2 we explained how RECOG addresses the recognition with various rotation angles. Regarding implementation, it is worth mentioning that with Cognex, rotation is clockwise, while it is counter-clockwise concerning OpenCVSharp, and therefore, RECOG. Additionally, Cognex saves the rotation in radian, while RECOG interprets rotation based on degree. While converting a Cognex recipe, first, a conversion from radian to degree is needed, and then we multiply the angle by minus one to gain the same result. *Equation 2* formulates how we convert a Cognex Angle to a RECOG angle.

$$RECOG_{Angle} = \left(\frac{180 * Cognex_{angle}}{\pi}\right) * -1 \qquad \textit{Equation 2}$$

Design decision and Technology choices

To decide on important design choices, we always first identified possible alternatives. The downsides and upsides of each approach, depending on the context and the primary concerns, were studied. Based on the results, a decision was made and shared with stakeholders to get feedback as early as possible. Here you can find a sample of such decisions and the rationale behind choosing them.

Computer Vision Open-Source Library Opportunities

During the initial project phase, different pattern recognition and image alignment techniques have been reviewed. We didn't confine ourselves to any specific library at the exploration phase but studied different techniques and algorithms. The main objective was to investigate which approach could lead to the desired result.

After deciding on the techniques that should be used, a comparison between available open-source libraries was drawn. The main criteria for this comparison were their license, supported programming language, performance, and the functionalities they support. Table 14 describes the result of the comparison between the short-listed libraries.

Table 14. Computer vision open-source libraries comparison

Library	Wrappers	License	Language	Pros	Cons
OpenCV		Apache 2	C++	Best performance	Programming Language
	Python	Apache 2	Python	Easy prototyping	Programming Language/ performance
	EMGU	GPL	C#	Programming Language	License
	OpenCVSharp	Apache 2	C#	Programming Language	Not enough documentation and samples as the C++ or Python version
	BoofCV	Apache 2	Java		Language
VLFeat	-	BSD License	C or MATLAB	Rich set of algorithms for feature extraction	Not active anymore, Last update 8/1/2018
Scikit-Image and SciPy	-	BSD 3-clause	Python	Easy prototyping	Language, and do not support all OpenCV functionalities
Accord.Net /AForge.NET	-	LGPL v3 license	C#	Rich set of image filters	License

OpenCV turned to be the most comprehensive computer vision library on the market. It also has almost the best performance in comparison to the other libraries. There are various wrappers for this library in the market. The original OpenCV is based on C/C++, and it has the best performance; however, based on the system implementation's requirement, the library should provide a C# interface.

Consequently, we drew a comparison between different alternatives to investigate the performance of the available C# wrapper. For this aim, we implemented a set of similar functionalities using both OpenCVSharp and OpenCV C++. We measured the elapsed time of invoking the C++ library and calling the C# function and compared the results. The result showed that OpenCVSharp has almost the same performance as its C++ counterpart. However, memory management in C++ is more challenging, and also maintaining a C# code is more straightforward considering that the YS code is in C#. Therefore, OpenCVSharp was selected.

WPF or Windows Form

For the design and development of ART, we needed to decide if we would like to use WPF[15] or Windows Forms Applications (Winforms). They are both mainly used to develop and design Windows applications. However, WPF can be used for developing web applications as well. APT is a Windows Form application. Windows Form applications are simpler and require less effort for implementation. WPF applications are more scalable, providing better performance and rendering speed. One of the

[15] Windows Presentation Platform

main merits of using WPF is that it can effectively separate UI from the business logic, enhancing code testability and maintainability with its great support in data binding.

Since this project was a feasibility study, creating a fancy user interface was not a priority. However, WPF enables developers to readily alter the user interface without changing any single line of the business logic. In this way, in the future, they can easily alter the user interface when the product is ready to deliver to the customers. Additionally, the majority of YS windows applications are implemented using WPF. Hence, selecting WPF complies with the YS architecture. All in all, we decided to implement ART using WPF, accepting its complexity and possibly more implementation time.

Architectural Framework

Over the last few years, a few best practices evolved to organize applications into logical components, like MVC[16], MVP[17], and MVVP[18]. MVP and MVVM are derivations of the MVC pattern and two of the most widely adopted patterns used for building user interfaces.

Since we opted to use WPF, we decided to use the MVVM architectural pattern. This pattern facilitates the separation of the graphical user interface development from the business logic development. The combination of WPF and MVVM is one of the best practices [21]. MVVM applications consist of three layers:

- The Model layer: It contains business rules, data access, and classes.
- The View layer: It defines the user interface.
- The ViewModel layer: It plays as a mediator between the view and the model layer, containing the presentation logic.

MVVM was designed to remove all GUI code ("code-behind") from the view layer by using data binding functions in WPF. The MVVM pattern attempts to gain both advantages of separation of functional development provided by MVC while leveraging the advantages of data bindings and the framework by binding data as close to the pure application model as possible [20]. Figure 47 illustrates the MVVM architectural pattern.

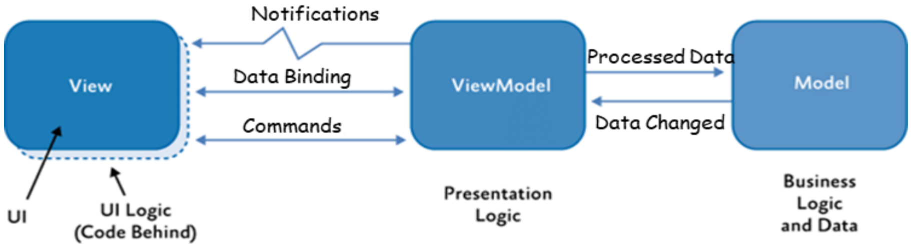


Figure 47. MVVM architecture

MVVM uses data binding, and therefore it is an event-driven architecture. In MVP [11], the presenter has knowledge about the view; however, in MVVM, the ViewModel layer has no reference to the View. It allows binding multiple views to a single ViewModel. In other words, it makes the application modules loosely coupled. Hence, we decided to apply MVVM architectural pattern, to follow the best practices in designing the system, and increase the code extensibility and maintainability.

[16] Model-View-Controller

[17] Model-View-Presenter

[18] Model-View-ViewModel

MVVM Framework

Although it is possible to implement MVVM architecture without using any framework, there are many powerful frameworks that address the common concerns, reducing the implementation time, including Prism, Caliburn.Micro, and MVVMLight. Prism and Caliburn.Micro are two more powerful frameworks, supporting lots of features. As a result, it is expected that they might have steeper learning curve in comparison to MVVMLight. While Caliburn and MVVMLight are licensed under MIT, Prism has a GPL license, which means it was not acceptable by the ASML Free and Open Source Software (FOSS) portal. We decided to use MVVMLight. The main and the most important reason for picking an MVVM framework was to address the communication between ViewModels, which was nicely supported by the MVVMLight framework, without adding unnecessary features that could not add any values to this project. Besides, the MVVMLight framework was already introduced in YS in some other YS projects, so there was no need to introduce a new library to YS.

Appendix B. Verification and Validation

This appendix gives more information about the testing phase. Chapter 5 gives a short summary of the requirements. In this Appendix, we listed all functional requirements in more detail in Table 15. Test requirements were elicited based on the requirements, and a traceability matrix was created. You can find more details about the manual tests in the following sections.

Table 15. Functional requirements in detail

Id	Requirement Description	Related concern	Priority
F01	The ART user interface shall allow the user to load the input image in IM format.	4B, 5.G, 6.A	Must
F02	The ART user interface shall allow the user to load the input image in PNG format.	4B, 5.G, 6.A	Must
F03	The ART user interface shall allow the user to import the model-based pattern.	5.G, 6.A, 6.D	Must
F04	The ART user interface shall allow the user to save the model-based pattern.	5.G, 6.A, 6.D	Must
F05	The ART user interface shall allow the user to set the resolution of the model-based pattern.	4.C, 5.B, 6.A, 6.D	Must
F06	The ART user interface shall allow the user to train the model.	4.B, 5.H	Must
F07	The ART user interface shall represent image metadata.	5.G, 6.A	Should
F08	The ART user interface shall allow the user to define the train region in the rectangle shape.	5.E, 5.G, 6.A	Must
F09	The ART user interface shall allow the user to define the train origin on the micrometer scale.	5.E, 5.B, 5.G, 6.A	Must
F10	The ART user interface shall allow the user to switch between four different views, including input image, recognized pattern, last run result image, and train image.	5.G, 6.A	Must
F11	The ART user interface shall allow the user to zoom into the last run image.	5.G, 6.A	Should
F12	The ART user interface shall allow the user to set the train region either based on the region top left corner or it's center.	4.B, 5.H	Should
F13	The ART user interface shall allow the user to set runtime parameters including accept threshold, scale, and angle.	4.B, 5.E, 5.G, 6.A	Must
F14	The ART user interface shall allow the user to run a model-based pattern recognition and get the recognition result.	4.B, 5.H	Must

F15	The ART user interface shall show the recognition result in a table with information including X, Y offset (micrometer), score, angle, and the scaling factor.	5.B	Must
F16	The ART user interface shall facilitate automatic pattern recognition execution upon changes in the parameters.	5.G, 6.A	Could
F17	The ART user interface shall enable the user to save the pattern recognition settings in the XML format.	5.G, 6.A, 6.D, 5.H, 4.B	Must
F18	The ART user interface shall enable users to save the alignment profile, with the ".align" extension.	5.G, 6.A, 6.D, 5.H, 4.B	Must
F19	The ART user interface shall enable the user to load an alignment profile.	5.G, 6.A, 6.D, 5.H, 4.B	Must
F20	The ART user interface shall enable user to load an existing XML-based pattern recognition setting and model.	5.G, 6.A, 6.D, 5.H, 4.B	Must
F21	ART shall facilitate pattern recognition using an specified rotation or scale.	4.B	Should
F22	ART shall facilitate measuring the best rotation angle in a given range, which could lead to the best pattern recognition result.	4.B	Could
F23	ART shall facilitate finding the best pattern scaling factor in a given range.	4.B	Could
F24	ART shall retrain the pattern automatically based on the input image resolution.	4.B, 5.E	Should
F25	The ART user interface shall enable the user to name the pattern.	4.B, 5.G, 6.A	Must
F26	The ART user interface shall enable the user to define the search region.	4.B, 5.B, 6.A	Could
F27	RECOG shall measure a similarity score to help detecting false positives.	4.B, 5.B	Must
F28	ART shall be able to run pattern recognition in batch for a set of images against a specific model-based pattern.	5.G, 6.A	Should
F29	ART shall save the result of batch execution in a csv file.	5.G, 6.A	Must
F30	ART shall enable users to convert a model-based align file created by Cognex to a default RECOG-based align file.	4.D, 6.A, 6.D	Should
F31	YieldStar shall let users decide whether the Cognex library or the RECOG library should be used during the wafer alignment.	4.B	Should
F32	YieldStar shall allow users to measure a wafer using RECOG.	4.B, 6.A	Should

F33	YieldStar shall record diagnostic data based on the result of the RECOG pattern recognition.	4.B, 4.G	Should

Traceability

The traceability matrix shown in Table 16 describes new test requirements defined based on the system requirement. It shows how they can be mapped to the system requirements and test cases.

Table 16. Traceability matrix

ID	Test Requirement	Ref. Requirement	Test Case	Test Approach
TR01	Converting a Cognex-based tool to a RECOG-based tool			
TR01.1	***Description***: ART shall be able to convert an align file created using Cognex to a new RECOG-based align file. ***Rationale***: Creating recipes (alignment profiles) from scratch is time-consuming. We want to make sure they can be reused. ***Pass Criteria***: The file should be converted without any exception if the Cognex dongle is present. ART should be able to load the converted align file.	F30, F31	TC-001	Manual test and unit tests
TR01.2	***Description***: ART shall be able to convert an XML file created using Cognex to a new file with the ".recog" extension that is acceptable by ART. ***Rationale***: The same as TR0.1.1., but with respect to the XML/recog file. ***Pass Criteria***: The same as TR0.1.1., but with respect to the XML/recog file.	F30	TC-004	Manual test and unit tests
TR02	***Description:*** ART shall run batch recognition, given the set of images and a RECOG-based file containing the tool description. ***Rationale***: In order to troubleshoot the system, running a batch pattern recognition is required. ***Pass Criteria***: A CSV file containing the recognition results of the images should be created.	F28, F29	TC-005	Manual test and unit tests
TR03	Save/Load a model-based pattern			
TR03.1	***Description:*** ART shall save a model-based pattern as an XML file. ***Rationale***: Each pattern can be used for different wafers. New recipes can be created based on the same pattern. ***Pass Criteria***: An XML file containing the pattern description should be created without any exception.	F04	TC-001	Manual test and unit tests
TR03.2	***Description***: ART shall load a model-based pattern saved as an XML file using ART. ***Rationale***: For creating a new recognition tool, the pattern should be loaded. Patterns can be reused.	F03	TC-002	Manual test and unit tests

	Pass criteria: Pattern should be loaded and get visualized.			
TR04	Import/Export tool			
TR04.1	***Description****:* ART shall export the tool with ".recog" extension, containing the tool pattern info, image info, train, and run parameters. ***Rationale****:* Each ".recog" file can be used to load the tool with the same setting again. ***Pass Criteria***: A file with "recog" extension should be created.	F17, F25	TC-002	Manual test and unit tests
TR04.2	***Description***: ART shall save the tool alignment profile. ***Rationale****: An a*lignment profile is required for creating a wafer alignment recipe. ***Pass Criteria****:* A file with ".align" extension should be created.	F18, F25	TC-003	Manual test and unit tests
TR04.3	***Description****:* ART shall import the tool alignment profile (.align) file. ***Rationale***: alignment profile is required for creating a wafer alignment recipe. ***Pass Criteria***: The tool's pattern with already defined parameters should be loaded.	F19	TC-001, TC-003	Manual test and unit tests
TR04.4	***Description****:* ART shall import the exported tool with ".recog" extension. ***Rationale***: Loading an already created tool can be used in batch recognition, rerunning, and troubleshooting pattern recognition. ***Pass Criteria***: The tool's pattern with already defined parameters should be loaded.	F20	TC-003	Manual test and unit tests
TR05	***Description***: ART shall run the pattern recognition given the runtime parameters and represent the result. ***Rationale***: ART shall use RECOG to recognize patterns or report the absence of patterns in the given image. ***Pass Criteria*****:** RECOG shall run the recognition procedure based on the given parameters without any exception and show the result. The result might contain the last run images and also the offset and score.	F10, F11, F13, F14, F15, F21, F22, F23, F24, F27	TC-001, TC-004	Manual test and unit tests
TR06	***Description****:* ART shall train patterns given the train parameters. ***Rationale:*** Training is the prerequisite for the recognition phase. Therefore, it should be done without any error. ***Pass criteria:*** Pattern should be visualized, and the status of the training info should change.	F05, F06, F08, F09, F12	TC-003	Manual test and unit tests

TR07	***Description:*** ART shall load images either in IM or PNG format. ***Rationale:*** Running a pattern recognition requires an input image, which could be either in PNG or IM format. ***Pass Criteria:*** Image metadata shall be represented. Input image shall be represented in the image box. Finally, recognition should be done based on the loaded image.	F01, F02, F07	TC-002, TC-004	Manual test and unit tests
TR08	***Description***: RECOG shall find the offset X and offset Y with less than 0.8-micrometer difference with the reported offset X and offset Y by Cognex. ***Rationale:*** RECOG should recognize patterns accurately. ***Pass criteria:*** The difference between the Cognex and RECOG results should be less than 0.8 um.	NF4	TC-002	Manual test and unit tests
TR09	***Description:*** Given the same image and pattern, RECOG recognition success shall be at least 90 percent of the Cognex recognition success. ***Rationale***: The system shall recognize patterns regardless of the image brightness or quality. ***Pass criteria:*** The number of successful recognition by RECOG should be more than 90% of Cognex's successful recognition.	NF5	TC-004, TC-005	Manual test
TR10	***Description***: The pattern recognition algorithm shall be deterministic. ***Rationale***: The system shall report the same result given the same inputs, proving the algorithm's stability and reliability. ***Pass Criteria:*** Getting the same result for running the same recipe.	NF6	TC-002	Manual test
TR11	***Description***: RECOG shall recognize each mark in less than 250 milliseconds. ***Rationale:*** This recognition time is promising enough to achieve the desired performance in the production-ready application. ***Pass Criteria***: recognition time should be less than 250 ms.	NF2	TC-001, TC-002, TC-005	Manual test and unit test
TR12	***Description***: YS shall measure a wafer without a Cognex dongle. ***Rationale***: RECOG shall be integrated into YS and should facilitate measuring wafers accurately and efficiently. ***Pass Criteria***: YS should align wafers without a Cognex dongle. TMU and matching values should be	F30, F31, F32, F33, NF2, NF4, NF5	TC-006	Manual test

	based on YS specifications. These values are explained in detail in the validation section.			

The short description of the test cases are as follows:

- *TC-001 Conversion test*: Ensure that .align files created using the Cognex library can be converted properly to be loaded in the ART. Make sure that the image can be set, and we can run a recognition based on the default parameters. Finally, the goal is to check if the pattern can be extracted from the tool and saved as an XML file.

- *TC-002 Accuracy test*: Ensure that pattern recognition can be done without a Cognex dongle. Additionally, the goal is to make sure the difference between the offset reported by the APT (tool based on the Cognex library) and ART is less than 0.8 μm. At the same time, we can check if the algorithm is deterministic by comparing the result with the previous test result. As another test, after loading the image, we can test if the image metadata can be represented by ART. Finally, exporting the tool with the “recog” extension can be verified.

- TC-003 *Parameter test*: Ensure that file created with the “recog” extension can be loaded. Make sure that the user can change the train region, even based on the “center” of the region. Verify if the user can change the pattern origin. Also, in this test, the goal is to check run parameters could be set correctly, and the algorithm and the tool can recognize the pattern based on the set parameters.

- *TC-004 Flexibility test:* Ensure that Cognex objects (.XML files) can be converted to the new definition. RECOG flexibility and reliability will be tested against a different pattern. By loading IM images, we can make sure the tool can represent IM images. Finally, by setting a low-contrast and noisy input image, test the library flexibility.

- *TC-005 Batch recognition test*: Ensure that ART can run batch pattern recognition. Also, checking the recognition time of a batch of images makes sure that the recognition time is less than the threshold. Finally, the robustness and flexibility of the RECOG are tested.

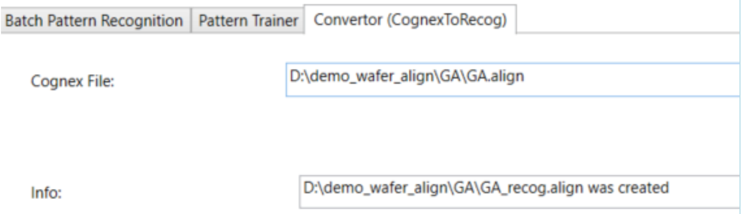
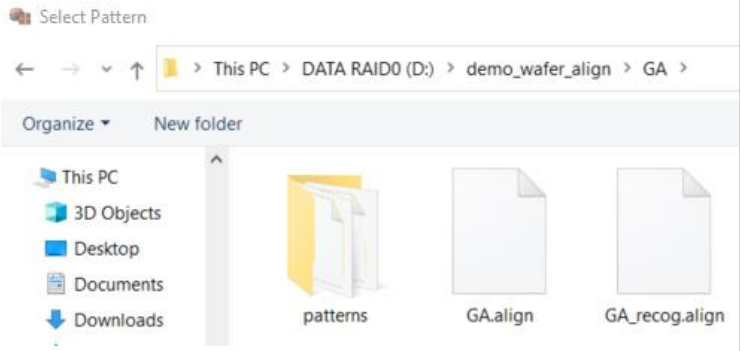
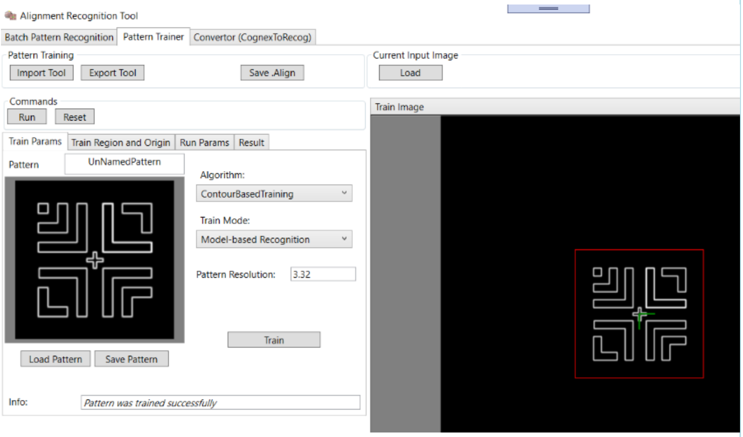
- *TC*-006 *Integration test:* Ensure that RECOG is integrated into YS. Wafer measurement and alignment can be done using RECOG. Diagnostic files can be created based on the RECOG library results. Performance and accuracy of wafer measurement using RECOG are accepted based on the YS specifications.

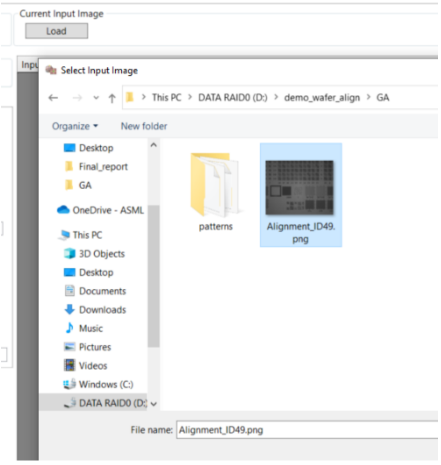
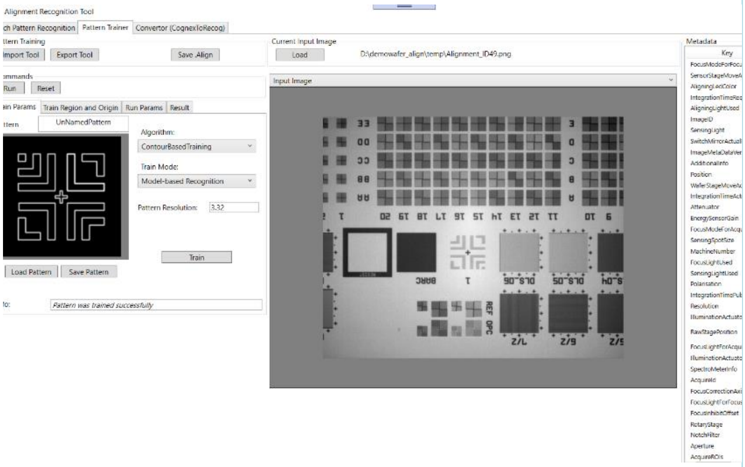
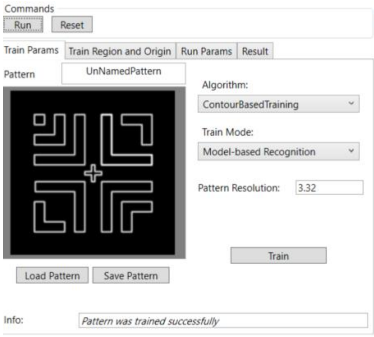
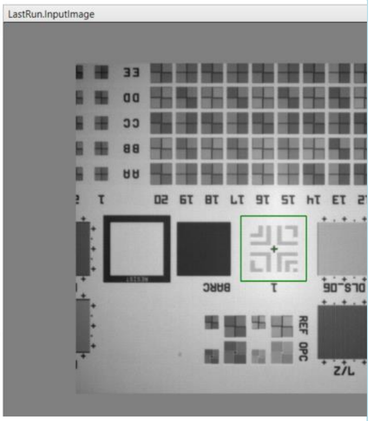
ART Manual Tests

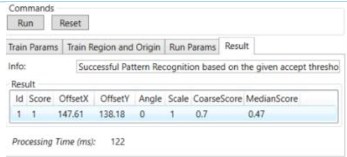
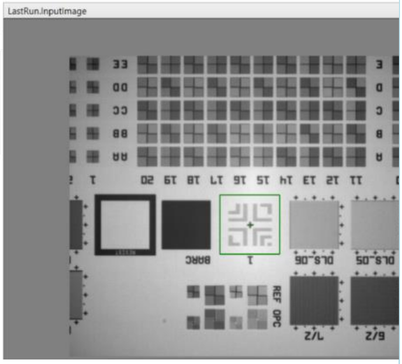
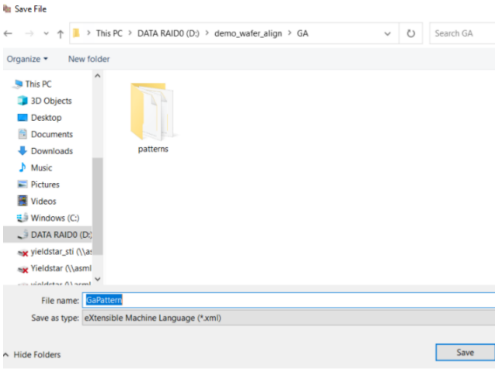
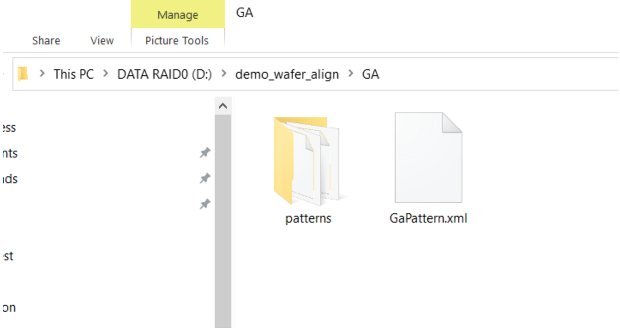
1.1.1 Tool Conversion

Table 17. TC-001 Tool Conversion

Objective	Ensure that .align files created using the Cognex library can be converted properly so that it can be loaded in ART. Ensure that the image can be set and the tool can be ran based on the default parameters. Finally, the goal is to check if the pattern can be extracted from the tool and saved as an XML file.
Pre-conditions / Set up	A Cognex dongle should be present. A Cognex-based align file should exist.
Configuration	Local system
Duration	Two minutes on average

Step ID	Execute	Expected result
1.	Load the Cognex tool (.align file,) and press convert. A new .align file at the same path should be created.	After selecting a file, the path should be represented in the Cognex file box. After pressing ‘convert’, the Info box should show a message describing if conversion was successful or an error has occurred.  Batch Pattern Recognition Pattern Trainer Convertor (CognexToRecog) Cognex File: D:\demo_wafer_align\GA\GA.align Info: D:\demo_wafer_align\GA\GA_recog.align was created
2.	Check file system while importing tool to check if the file exists.	GA_recog.align should exist in the same path as described in the Info box in the previous step.  Select Pattern This PC › DATA RAID0 (D:) › demo_wafer_align › GA › Organize ▾ New folder This PC 3D Objects Desktop Documents Downloads patterns GA.align GA_recog.align
3.	Load the recenly converted .align file by pressing the “Import Tool” button.	After loading the converted tool, the pattern should be represented in the “train params” tab. Also, one can select the “Train Image” from the combobox on the right side. Pattern with its region should be presented.  Alignment Recognition Tool Batch Pattern Recognition Pattern Trainer Convertor (CognexToRecog) Pattern Training Import Tool Export Tool Save .Align Current Input Image Load Commands Run Reset Train Params Train Region and Origin Run Params Result Train Image Train Load Pattern Save Pattern Pattern: UnNamedPattern Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: *Pattern was trained successfully*
4.	Load the image named Alignment_ID49.png.	The image should be shown in the input image tab. Also, image metadata should be represented in the metaData tab.

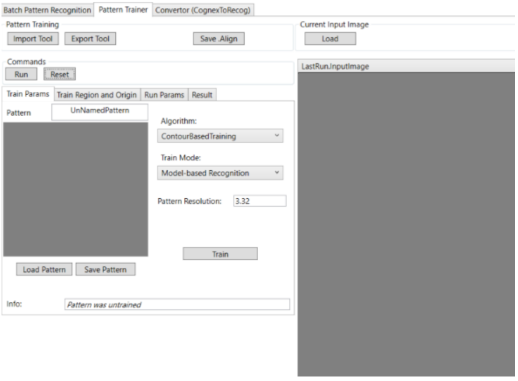
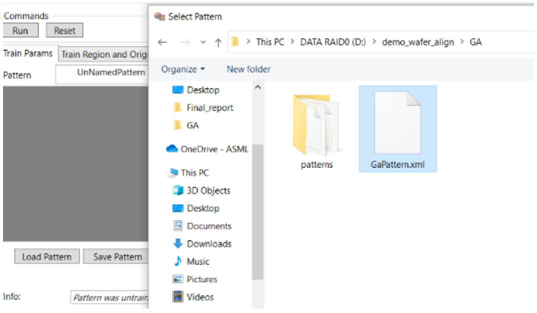
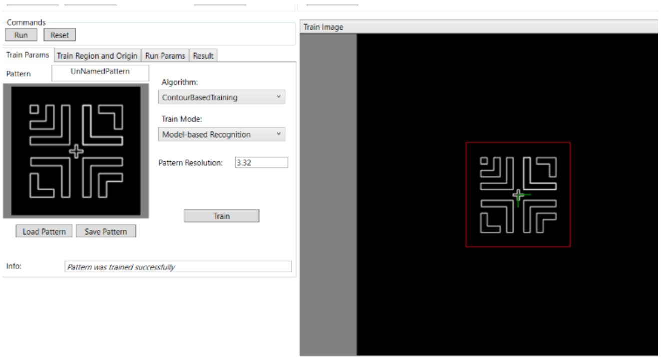
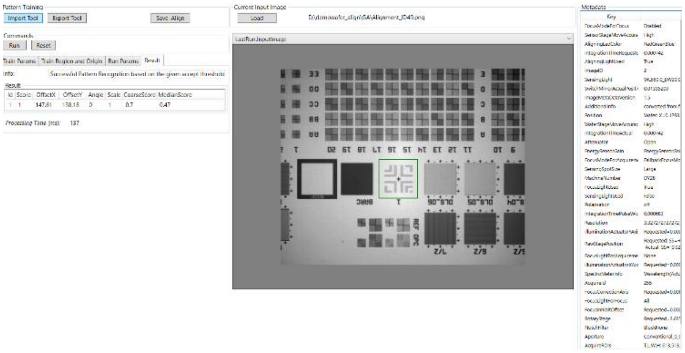
		 Current Input Image Select Input Image This PC > DATA RAID0 (D:) > demo_wafer_align > GA File name: Alignment_ID49.png  Alignment Recognition Tool Current Input Image D:\demowafer_align\temp\Alignment_ID49.png Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 to: *Pattern was trained successfully*
5.	Press "Run".	Upon running the recognition, the recognition result should be shown in the LastRun.InputImage.  Pattern: UnNamedPattern Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: *Pattern was trained successfully*  LastRun.InputImage
6.	Check the result tab.	In the result tab, there should be a record if the pattern is recognized. Also, the infobox should show the information about the success/failure of the recognition.

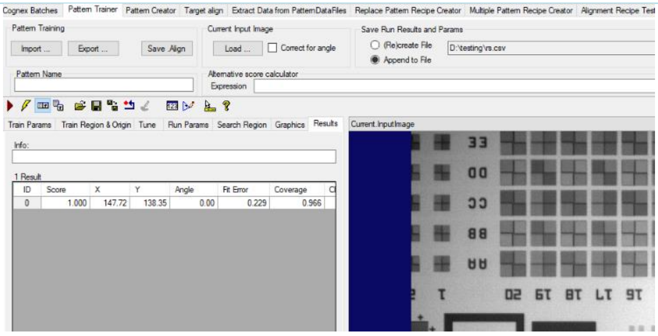
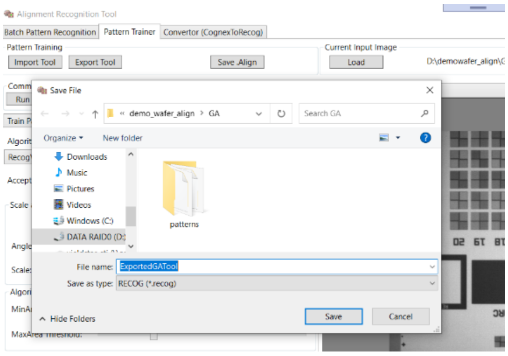
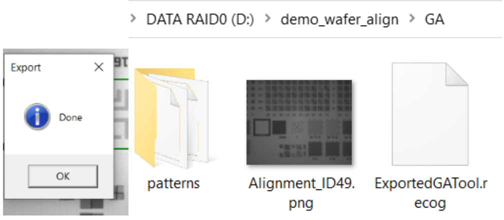
		 Commands Run Reset Train Params / Train Region and Origin / Run Params / Result Info: Successful Pattern Recognition based on the given accept thresho Result Id: 1, Score: 1, OffsetX: 147.61, OffsetY: 138.18, Angle: 0, Scale: 1, CoarseScore: 0.7, MedianScore: 0.47 Processing Time (ms): 122  LastRun.InputImage
7.	Press “Save Pattern”.	GaPattern.XML file should exist in the selected path after saving.  Save File This PC › DATA RAID0 (D:) › demo_wafer_align › GA Search GA Organize ▾ New folder This PC 3D Objects Desktop Documents Downloads Music Pictures Videos Windows (C:) DATA RAID0 (D:) yieldstar_sti (\\as Yieldstar (\\asml patterns File name: GaPattern Save as type: eXtensible Machine Language (*.xml) Hide Folders Save  Manage GA Share View Picture Tools This PC › DATA RAID0 (D:) › demo_wafer_align › GA patterns GaPattern.xml
TAR OK/NOK	*Ok*	

1.1.2 Test Recognition accuracy

Table 18. TC-002 Test recognition accuracy

Objective	Ensure that pattern recognition can be done without Cognex dongle. Additionally, the goal is to make sure the difference between the offset reported by the APT (tool based on the Cognex library) and ART is less than 0.8 micrometer. At the same time, one can check if the algorithm is deterministic or not by comparing the result with the previous test result. As another test, after loading the image, we can test if the image metadata can be represented by ART. Finally, exporting the tool as .recog file is tested.	
Pre-conditions / Set up	Make sure the Cognex dongle is not present. The pattern XML file and input image should exist. Before this test (when Cognex dongle is present,) load the GA.align file into APT and set the same input image (Alignment_ID49.png,) and run the recognition.	
Configuration	Local system	
Duration	One or two minutes in order to do all preparations and run the tests.	
Step ID	**Execute**	**Expected result**
1.	Press “Reset” to reset the tool.	Input image and the pattern should be removed.

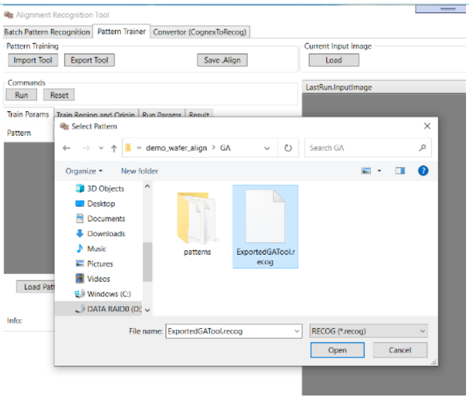
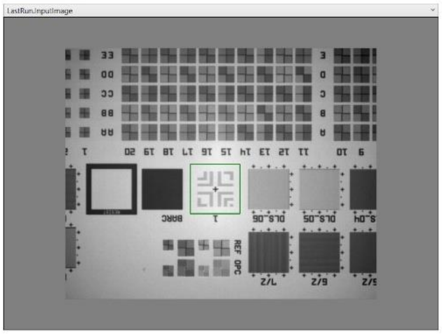
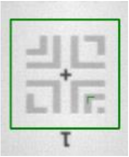
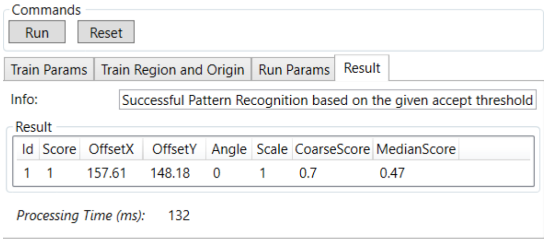
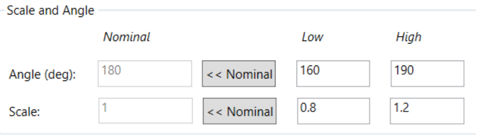
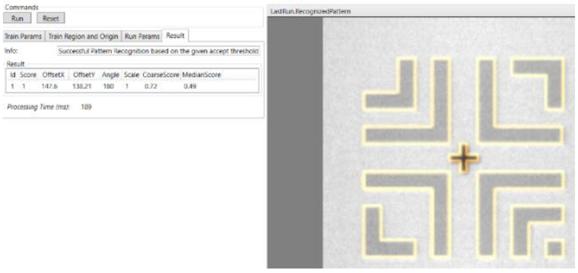
		 Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: Pattern was untrained
2.	Press the “Load Pattern” button and select the pattern created in the previous test.	The pattern should be loaded and trained automatically. Also, the Train Image view should represent the pattern with its train region and the train origin.  This PC > DATA RAID0 (D:) > demo_wafer_align > GA  Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: Pattern was trained successfully
3	Load the input image, the same input image as the previous test. Press the “Run” button one more time. Check the result tab again.	MetaData column should represent the image metadata. The result tab should show the recognition result details. The image preview section should show the LastRun.InputImage.  Info: Successful Pattern Recognition based on the given accept threshold Result Id Score OffsetX OffsetY Angle Scale CoarseScore MedianScore 1 1 147.61 138.18 0 1 0.7 0.47 Processing Time (ms): 137
4	Compare the result of APT and ART. Also, compare the result reported by ART in the previous test and the new result.	It is expected that the difference between X offset and Y offset reported by ART with the ones reported by APT be less than 0.8 micrometer. Also, the offset reported by ART in the previous test and this test should be similar.

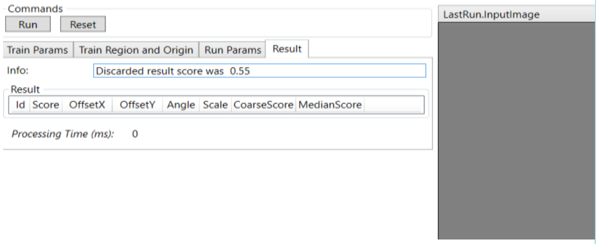
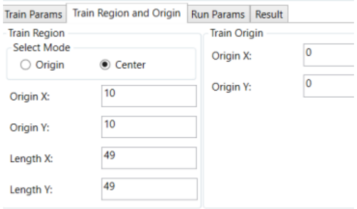
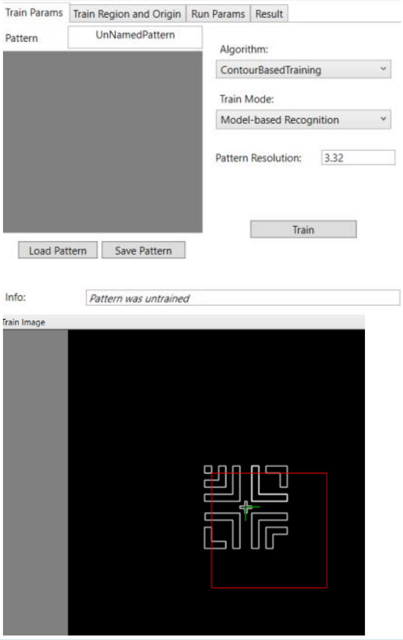
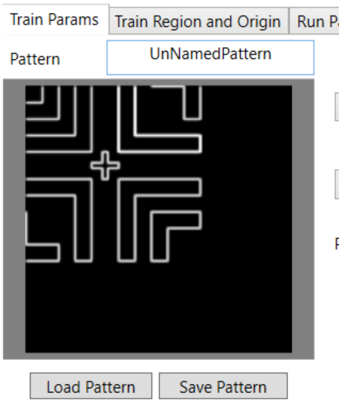
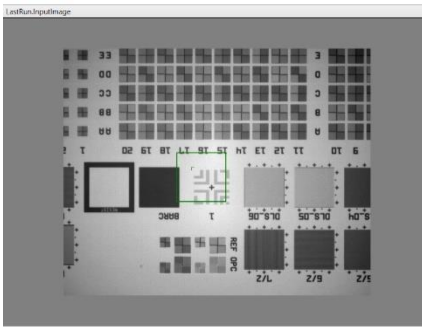
		 1 Result ID: 0, Score: 1.000, X: 147.72, Y: 138.35, Angle: 0.00, Fit Error: 0.229, Coverage: 0.966 Save Run Results and Params: ○ (Re)create File ● Append to File, D:\testing\rs.csv
5	Press “Export” button to save the tool as “.recog“ file. Fill in the “File Name” section as “ExportedGATool” without any extension.	A file with the name “ExportedGATool.recog” should exist in the path you saved the tool.  Save File demo_wafer_align › GA File name: ExportedGATool Save as type: RECOG (*.recog)  DATA RAID0 (D:) › demo_wafer_align › GA Export Done patterns Alignment_ID49.png ExportedGATool.recog
TAR OK/NOK	*Ok*	

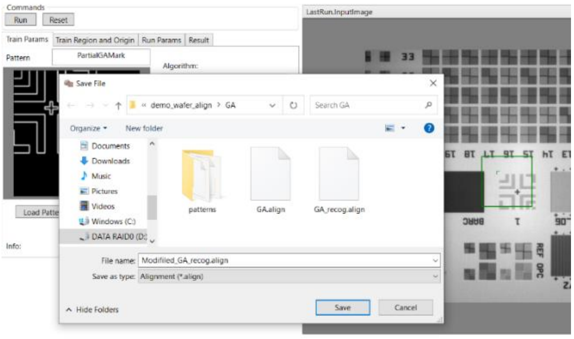
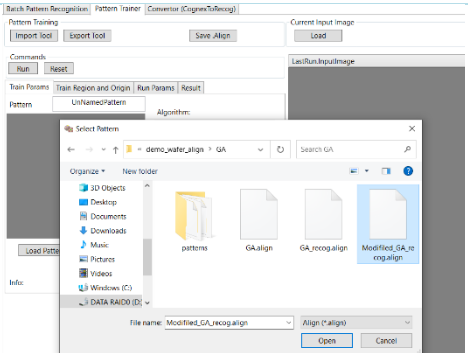
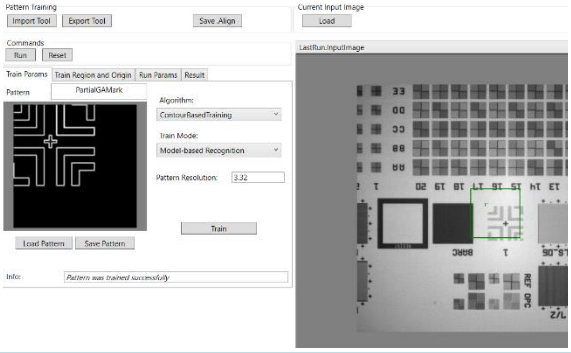
1.1.3 TEST Parameters

Table 19. TC-003 Test recognition with different parameters

Objective	Ensure that the saved “.recog” object can be loaded. Make sure that the user can change the train region, even based on the “center” of the region. They can change the pattern origin. Also, the goal is to check whether run parameters could be set correctly, and the algorithm and the tool can recognize the pattern based on the set parameters.	
Pre-conditions / Set up	Cognex should not be present. ExportedGATool.recog can be created as explained in the previous test. In summary, you need to load GA pattern and Alignmen_ID49.png as an input image, export the tool as “ExportedGATool.recog”.	
Configuration	Local system	
Duration	Two minutes on average to do preparations and run tests.	
Step ID	**Execute**	**Expected result**
1.	Reset the tool by pressing the “Reset” button. Press on the “Import Tool” button and load the “ExportedGA-Tool.recog” file created in the previous test.	The file should be located at the saved path.

		 Alignment Recognition Tool Batch Pattern Recognition Pattern Trainer Convertor (CognexToRecog) Pattern Training Import Tool Export Tool Save .Align Current Input Image Load Commands Run Reset LastRun.InputImage Train Params Train Region and Origin Run Params Result Pattern Select Pattern demo_wafer_align › GA Search GA Organize New folder 3D Objects Desktop Documents Downloads Music Pictures Videos Windows (C:) DATA RAID0 (D: patterns ExportedGATool.recog Load Patt Info: Open Cancel File name: ExportedGATool.recog RECOG (*.recog) Upon loading the “.recog” tool, the recognition should be done automatically. 
2.	On the “Train Region and Origin” tab, set the origin X and Origin Y to 10, and rerun the tool.	The new offset X and offset Y should also be shifted by 10 compared to the previous result.   Commands Run Reset Train Params Train Region and Origin Run Params Result Info: Successful Pattern Recognition based on the given accept threshold Result Id Score OffsetX OffsetY Angle Scale CoarseScore MedianScore 1 1 157.61 148.18 0 1 0.7 0.47 *Processing Time (ms):* 132
3	Set the origin again to (0,0). On the “Run Params” tab, set the scale and angle as below:  Scale and Angle *Nominal* *Low* *High* Angle (deg): 180 << Nominal 160 190 Scale: 1 << Nominal 0.8 1.2 And re-run the tool.	The angle in the result tab should be 180. The scale should be equal to 1. Also, looking at the LastRun.RecognizedPattern, the pattern should be rotated 180 degrees.  Commands Run Reset Train Params Train Region and Origin Run Params Result LastRun.RecognizedPattern Info: Successful Pattern Recognition based on the given accept threshold Result Id Score OffsetX OffsetY Angle Scale CoarseScore MedianScore 1 1 147.6 138.21 180 1 0.72 0.49 *Processing Time (ms):* 189
4	Set the accept threshold to 0.65, angle to 180, scale to 0.5, and rerun.	The result should be relatively low in comparison to the previous run because the scale is 0.5.

		 Commands: Run, Reset Train Params, Train Region and Origin, Run Params, Result Info: Discarded result score was 0.55 Result: Id, Score, OffsetX, OffsetY, Angle, Scale, CoarseScore, MedianScore Processing Time (ms): 0 LastRun.InputImage
5	Set the scale again to 1. On the “Train region and origin” tab, select “Center” as the select mode, and set the origin X and origin Y to 10.  Train Params, Train Region and Origin, Run Params, Result Train Region Select Mode: ○ Origin ● Center Origin X: 10 Origin Y: 10 Length X: 49 Length Y: 49 Train Origin Origin X: 0 Origin Y: 0	The pattern should be set to untrained.  Train Params, Train Region and Origin, Run Params, Result Pattern: UnNamedPattern Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Train Load Pattern, Save Pattern Info: *Pattern was untrained* Train Image
6	Press the “Train button.” Then run the tool.	The pattern should be partial.  Train Params, Train Region and Origin, Run P Pattern: UnNamedPattern Load Pattern, Save Pattern Recognition result should also reflect the partial pattern.  LastRun.InputImage

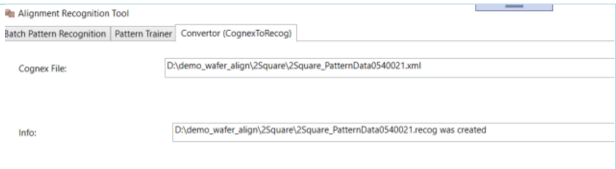
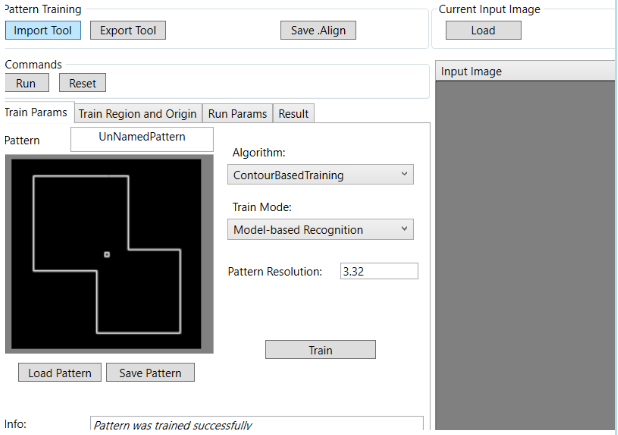
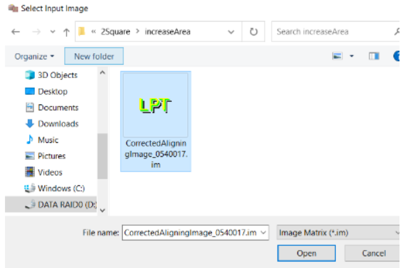
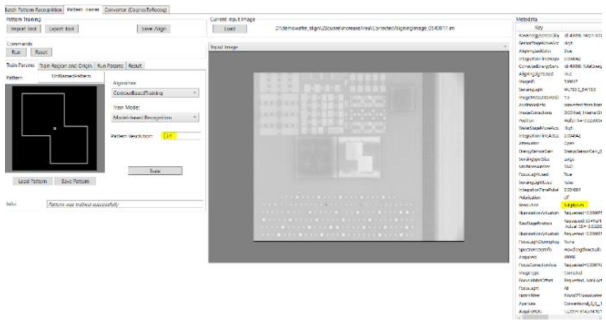
7	Give the pattern a name, such as 'PartialGAMark" and Press "Save Align" to save the alignment profile.	Look at the PatternGAMark in the pattern textbox.  Pattern: PartialGAMark Save File: demo_wafer_align > GA File name: Modifiled_GA_recog.align Save as type: Alignment (*.align)
8	Reset the tool and import the saved tool in the last step to make sure that it was saved in good order.	 Pattern: UnNamedPattern Select Pattern: demo_wafer_align > GA File name: Modifiled_GA_recog.align Align (*.align) After import, the pattern with the same name and result should be loaded.  Pattern: PartialGAMark Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: *Pattern was trained successfully*
TAR OK/NOK	*OK*	

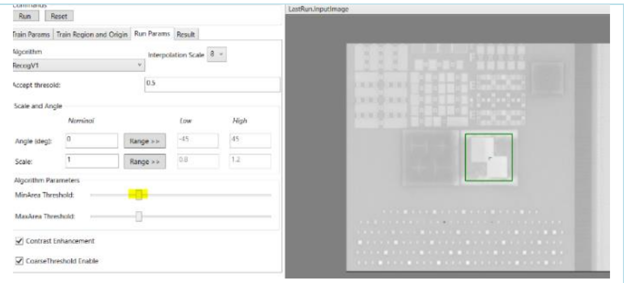
1.1.4 Test Tool Robustness

Table 20. TC-004 Test RECOG flexibility

Objective	Ensure that Cognex objects (XML files) can be converted to RECOG objects. RECOG flexibility and reliability in recognition will be tested against another pattern. By loading an IM image, make sure the tool can represent IM images. Finally, by choosing a low contrast and noisy image ensure that the library is flexible.
Pre-conditions / Set up	Test data, including the Cognex XML file, IM image, should be available.
Configuration	Local system
Duration	One or two minutes.

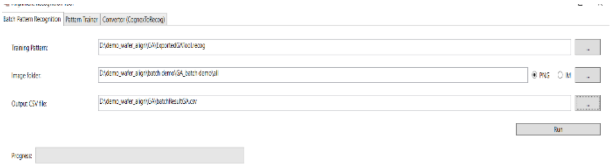
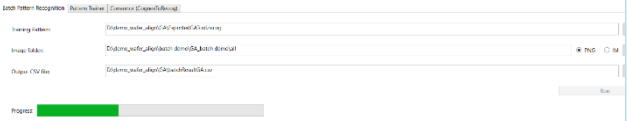
Step ID	**Execute**	**Expected result**
1.	In the "Converter" tab, press "…" to load an XML file containing the Cognex object and press "Convert."	After choosing the XML file, the path to the file should be represented in the Cognex file text box. After pressing convert, the conversion result should be represented in the "info" text box.

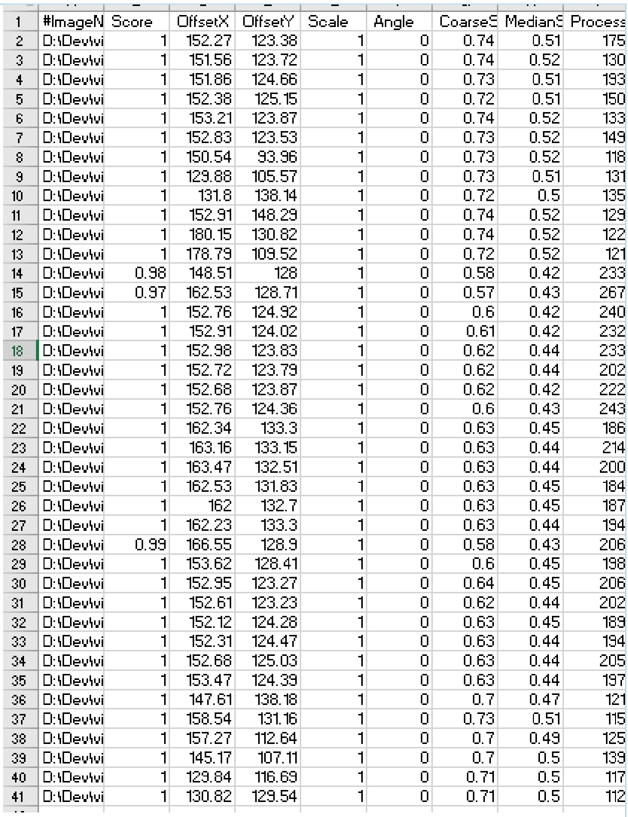
		 Cognex File: D:\demo_wafer_align\2Square\2Square_PatternData0540021.xml Info: D:\demo_wafer_align\2Square\2Square_PatternData0540021.recog was created
2.	Then in the “Pattern Trainer” tab, press “Import Tool” and load the converted tool (.recog file) at the last step.	After importing the tool, the pattern should be shown in the train params tab.  Pattern: UnNamedPattern Algorithm: ContourBasedTraining Train Mode: Model-based Recognition Pattern Resolution: 3.32 Info: *Pattern was trained successfully*
3.	Press on the “Load” button to load the IM file. After loading the IM files, the image should be represented in the input image view. It should be possible to review the image metadata in the Metadata panel. The resolution of the image metadata and pattern resolution should be the same.	After selecting and importing an IM file, the image should be shown in the Input image section. The metadata tab should reflect the image metadata. Also, the pattern resolution box in the “Train Params” tab should get updated based on the IM image resolution, shown in the metadata tab.  File name: CorrectedAligningImage_0540017.im Image Matrix (*.im) 
4.	Because there are lots of noises around the pattern shape in the input image, we increase the MinAreaThreshold (to 8) in the “Run Params” tab. Press “Run”.	The pattern should be recognized after increasing the MinAreaThreshold. Note that depending on the image contrast, noises, or even the type of pattern, the run parameters might need to be set differently. The goal is to test if the tool is configurable or not.

		 Commands: Run, Reset Tabs: Train Params, Train Region and Origin, Run Params, Result Algorithm: RecogV1 Interpolation Scale: 8 Accept thresold: 0.5 Scale and Angle — Nominal / Low / High Angle (deg): 0, Range >>, -45, 45 Scale: 1, Range >>, 0.8, 1.2 Algorithm Parameters: MinArea Threshold, MaxArea Threshold ☑ Contrast Enhancement ☑ CoarseThreshold Enable
TAR OK/NOK	*OK*	

1.1.5 *Batch recognition test*

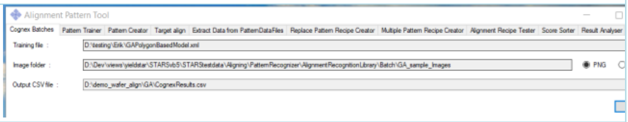
Table 21. TC-005 Test batch recognition

Objective	Ensure that ART can run batch pattern recognition. Also, by checking the recognition time of a batch of images, make sure that the average recognition time is less than the threshold (250 milliseconds.) Finally, the robustness and flexibility of RECOG is tested.	
Pre-conditions / Set up	Test data is available. Cognex dongle should be present in order to run the batch recognition using APT and compare the results.	
Configuration	Local system	
Duration	A few minutes depending on the number of images used in the recognition. Usually, it could take roughly one minute for 40 images.	
Step ID	**Execute**	**Expected result**
1.	Go to the "Batch Pattern Recognition" tab, in the "Training Pattern" row, press "…" button, and load "ExportedGATool." Select the folder containing the images. Select a path to save the recognition result and give a name to the file. Finally, press Run.	Upon selecting a file, the path to the file and its name should be shown in the corresponding box. In case you give only a name to the output CSV file, a ".csv" extension should be added automatically.  Tabs: Batch Pattern Recognition, Pattern Trainer, Convertor (CognexToRecog) Training Pattern: D:\demo_wafer_align\GA\ExportedGATool.recog Image folder: D:\demo_wafer_align\batch demo\GA_batch demo\all ● PNG ○ IM Output CSV file: D:\demo_wafer_align\GA\batchResultGA.csv Run Progress:
2.	Press "Run".	Upon pressing run, the progress bar should show the progress status of the batch recognition.  Tabs: Batch Pattern Recognition, Pattern Trainer, Convertor (CognexToRecog) Training Pattern: D:\demo_wafer_align\GA\ExportedGATool.recog Image folder: D:\demo_wafer_align\batch demo\GA_batch demo\all ● PNG ○ IM Output CSV file: D:\demo_wafer_align\GA\batchResultGA.csv Run Progress:
3.	Open the CSV file.	There should be a record for each image. The processing time for each record should be less than 250 milliseconds.



1	#ImageN	Score	OffsetX	OffsetY	Scale	Angle	CoarseS	MedianS	Process
2	D:\Dev\vi	1	152.27	123.38	1	0	0.74	0.51	175
3	D:\Dev\vi	1	151.56	123.72	1	0	0.74	0.52	130
4	D:\Dev\vi	1	151.86	124.66	1	0	0.73	0.51	193
5	D:\Dev\vi	1	152.38	125.15	1	0	0.72	0.51	150
6	D:\Dev\vi	1	153.21	123.87	1	0	0.74	0.52	133
7	D:\Dev\vi	1	152.83	123.53	1	0	0.73	0.52	149
8	D:\Dev\vi	1	150.54	93.96	1	0	0.73	0.52	118
9	D:\Dev\vi	1	129.88	105.57	1	0	0.73	0.51	131
10	D:\Dev\vi	1	131.8	138.14	1	0	0.72	0.5	135
11	D:\Dev\vi	1	152.91	148.29	1	0	0.74	0.52	129
12	D:\Dev\vi	1	180.15	130.82	1	0	0.74	0.52	122
13	D:\Dev\vi	1	178.79	109.52	1	0	0.72	0.52	121
14	D:\Dev\vi	0.98	148.51	128	1	0	0.58	0.42	233
15	D:\Dev\vi	0.97	162.53	128.71	1	0	0.57	0.43	267
16	D:\Dev\vi	1	152.76	124.92	1	0	0.6	0.42	240
17	D:\Dev\vi	1	152.91	124.02	1	0	0.61	0.42	232
18	D:\Dev\vi	1	152.98	123.83	1	0	0.62	0.44	233
19	D:\Dev\vi	1	152.72	123.79	1	0	0.62	0.44	202
20	D:\Dev\vi	1	152.68	123.87	1	0	0.62	0.42	222
21	D:\Dev\vi	1	152.76	124.36	1	0	0.6	0.43	243
22	D:\Dev\vi	1	162.34	133.3	1	0	0.63	0.45	186
23	D:\Dev\vi	1	163.16	133.15	1	0	0.63	0.44	214
24	D:\Dev\vi	1	163.47	132.51	1	0	0.63	0.44	200
25	D:\Dev\vi	1	162.53	131.83	1	0	0.63	0.45	184
26	D:\Dev\vi	1	162	132.7	1	0	0.63	0.45	187
27	D:\Dev\vi	1	162.23	133.3	1	0	0.63	0.44	194
28	D:\Dev\vi	0.99	166.55	128.9	1	0	0.58	0.43	206
29	D:\Dev\vi	1	153.62	128.41	1	0	0.6	0.45	198
30	D:\Dev\vi	1	152.95	123.27	1	0	0.64	0.45	206
31	D:\Dev\vi	1	152.61	123.23	1	0	0.62	0.44	202
32	D:\Dev\vi	1	152.12	124.28	1	0	0.63	0.45	189
33	D:\Dev\vi	1	152.31	124.47	1	0	0.63	0.44	194
34	D:\Dev\vi	1	152.68	125.03	1	0	0.63	0.44	205
35	D:\Dev\vi	1	153.47	124.39	1	0	0.63	0.44	197
36	D:\Dev\vi	1	147.61	138.18	1	0	0.7	0.47	121
37	D:\Dev\vi	1	158.54	131.16	1	0	0.73	0.51	115
38	D:\Dev\vi	1	157.27	112.64	1	0	0.7	0.49	125
39	D:\Dev\vi	1	145.17	107.11	1	0	0.7	0.5	139
40	D:\Dev\vi	1	129.84	116.69	1	0	0.71	0.5	117
41	D:\Dev\vi	1	130.82	129.54	1	0	0.71	0.5	112

Run the batch recognition for the same images based on the same pattern (in the XML format).



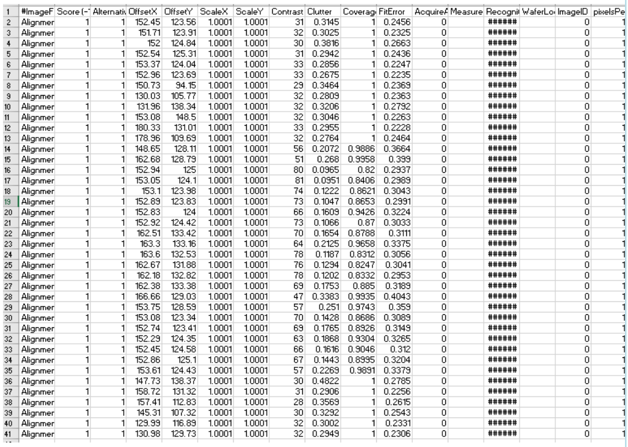
Training file : D:\testing\Erik\GAPolygonBasedModel.xml

Image folder : D:\Dev\views\yieldstar\STARSvb5\STARStestdata\Aligning\PatternRecognizer\AlignmentRecognitionLibrary\Batch\GA_sample_Images ● PNG ○

Output CSV file : D:\demo_wafer_align\GA\CognexResults.csv

Open the Cognex result file.

It is expected that RECOG could at least recognize 90% of patterns that are recognizable using the Cognex library. Comparing their results, both were able to recognize the pattern in all images.

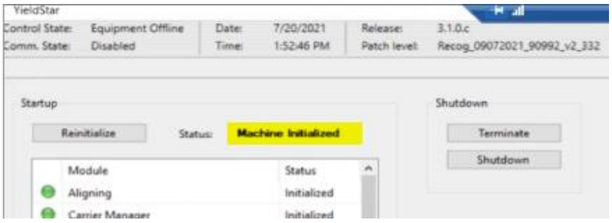
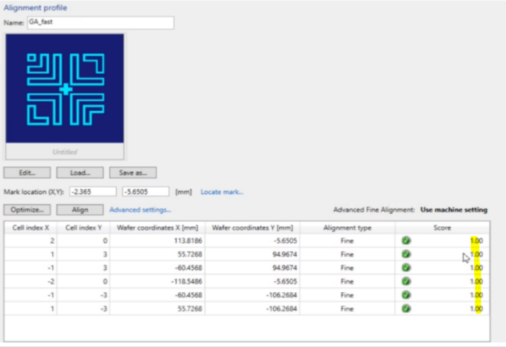
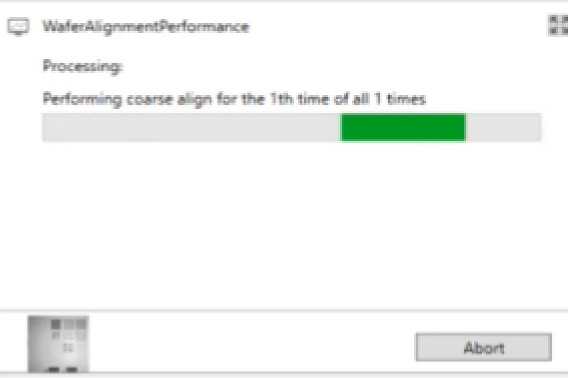
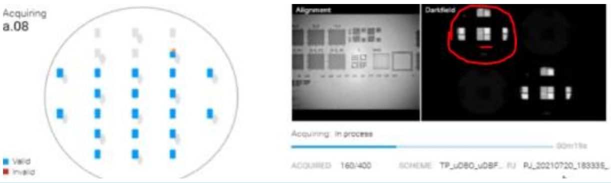


1	#ImageF	Score (-1	Alternati	OffsetX	OffsetY	ScaleX	ScaleY	Contrast	Clutter	Coverag	FitError	AcquireA	Measure	Recognit	WaferLo	ImageID	pixelsPe
2	Alignmer	1	1	152.45	123.56	1.0001	1.0001	31	0.3145	1	0.2456	0		######		0	1
3	Alignmer	1	1	151.71	123.91	1.0001	1.0001	32	0.3025	1	0.2325	0		######		0	1
4	Alignmer	1	1	152	124.84	1.0001	1.0001	30	0.3816	1	0.2663	0		######		0	1
5	Alignmer	1	1	152.54	125.31	1.0001	1.0001	31	0.2942	1	0.2436	0		######		0	1
6	Alignmer	1	1	153.37	124.04	1.0001	1.0001	33	0.2856	1	0.2247	0		######		0	1
7	Alignmer	1	1	152.96	123.69	1.0001	1.0001	33	0.2675	1	0.2235	0		######		0	1
8	Alignmer	1	1	150.73	94.15	1.0001	1.0001	29	0.3464	1	0.2369	0		######		0	1
9	Alignmer	1	1	130.03	105.77	1.0001	1.0001	32	0.2809	1	0.2363	0		######		0	1
10	Alignmer	1	1	131.96	138.34	1.0001	1.0001	32	0.3206	1	0.2792	0		######		0	1
11	Alignmer	1	1	153.08	148.5	1.0001	1.0001	32	0.3046	1	0.2263	0		######		0	1
12	Alignmer	1	1	180.33	131.01	1.0001	1.0001	33	0.2955	1	0.2228	0		######		0	1
13	Alignmer	1	1	178.96	109.69	1.0001	1.0001	32	0.2764	1	0.2464	0		######		0	1
14	Alignmer	1	1	148.65	128.11	1.0001	1.0001	56	0.2072	0.9886	0.3664	0		######		0	1
15	Alignmer	1	1	162.68	128.79	1.0001	1.0001	51	0.268	0.9958	0.399	0		######		0	1
16	Alignmer	1	1	152.94	125	1.0001	1.0001	80	0.0965	0.82	0.2937	0		######		0	1
17	Alignmer	1	1	153.05	124.1	1.0001	1.0001	81	0.0951	0.8406	0.2989	0		######		0	1
18	Alignmer	1	1	153.1	123.98	1.0001	1.0001	74	0.1222	0.8621	0.3043	0		######		0	1
19	Alignmer	1	1	152.89	123.83	1.0001	1.0001	73	0.1047	0.8653	0.2991	0		######		0	1
20	Alignmer	1	1	152.83	124	1.0001	1.0001	66	0.1609	0.9426	0.3224	0		######		0	1
21	Alignmer	1	1	152.92	124.42	1.0001	1.0001	73	0.1066	0.87	0.3033	0		######		0	1
22	Alignmer	1	1	162.51	133.42	1.0001	1.0001	70	0.1654	0.8788	0.3111	0		######		0	1
23	Alignmer	1	1	163.3	133.16	1.0001	1.0001	64	0.2125	0.9658	0.3375	0		######		0	1
24	Alignmer	1	1	163.6	132.53	1.0001	1.0001	78	0.1187	0.8312	0.3056	0		######		0	1
25	Alignmer	1	1	162.67	131.88	1.0001	1.0001	76	0.1294	0.8247	0.3041	0		######		0	1
26	Alignmer	1	1	162.18	132.82	1.0001	1.0001	78	0.1202	0.8332	0.2953	0		######		0	1
27	Alignmer	1	1	162.38	133.38	1.0001	1.0001	69	0.1753	0.885	0.3189	0		######		0	1
28	Alignmer	1	1	166.66	129.03	1.0001	1.0001	47	0.3383	0.9935	0.4043	0		######		0	1
29	Alignmer	1	1	153.75	128.59	1.0001	1.0001	57	0.251	0.9743	0.359	0		######		0	1
30	Alignmer	1	1	153.08	123.34	1.0001	1.0001	70	0.1428	0.8686	0.3089	0		######		0	1
31	Alignmer	1	1	152.74	123.41	1.0001	1.0001	69	0.1765	0.8926	0.3149	0		######		0	1
32	Alignmer	1	1	152.29	124.35	1.0001	1.0001	63	0.1868	0.9304	0.3265	0		######		0	1
33	Alignmer	1	1	152.45	124.58	1.0001	1.0001	66	0.1616	0.9046	0.312	0		######		0	1
34	Alignmer	1	1	152.86	125.1	1.0001	1.0001	67	0.1443	0.8995	0.3204	0		######		0	1
35	Alignmer	1	1	153.61	124.43	1.0001	1.0001	57	0.2269	0.9891	0.3379	0		######		0	1
36	Alignmer	1	1	147.73	138.37	1.0001	1.0001	30	0.4822	1	0.2785	0		######		0	1
37	Alignmer	1	1	158.72	131.32	1.0001	1.0001	31	0.2906	1	0.2256	0		######		0	1
38	Alignmer	1	1	157.41	112.83	1.0001	1.0001	28	0.3569	1	0.2615	0		######		0	1
39	Alignmer	1	1	145.31	107.32	1.0001	1.0001	30	0.3292	1	0.2543	0		######		0	1
40	Alignmer	1	1	129.99	116.89	1.0001	1.0001	32	0.3002	1	0.2331	0		######		0	1
41	Alignmer	1	1	130.98	129.73	1.0001	1.0001	32	0.2949	1	0.2306	0		######		0	1

TAR OK/NOK	OK

1.1.6 Integration test

Objective	Ensure that RECOG is integrated into YS. Wafer measurement and alignment can be done using RECOG. Diagnostic files can be created based on the RECOG library results. Performance and accuracy of wafer measurement using RECOG are accepted based on the YS specifications.
Pre-conditions / Set up	A new patch should be created. The configuration file should be in the recipe root folder. The converted align file (compatible with RECOG) should be available in the recipe folder with an expected name. The patch should be installed, and the machine should get initialized.

<table>
<tr><td colspan="2"><b>Configuration</b></td><td colspan="2">YieldStar machine, the user can get connected to the machine remotely.</td></tr>
<tr><td colspan="2"><b>Duration</b></td><td colspan="2">Depending on the test, it could take up to 4 hours to complete all the tests, including the installation and initialization phase.</td></tr>
<tr><td><b>Step ID</b></td><td colspan="2"><b>Execute</b></td><td><b>Expected result</b></td></tr>
<tr><td>1.</td><td colspan="2">Install the patch and initialize the machine.</td><td>The machine should get initialized without any errors.

YieldStar
Control State: Equipment Offline Date: 7/20/2021 Release: 3.1.0.c
Comm. State: Disabled Time: 1:52:46 PM Patch level: Recog_09072021_90992_v2_332
Startup
Reinitialize Status: Machine Initialized
Module Status
Aligning Initialized
Carrier Manager Initialized
Shutdown
Terminate
Shutdown</td></tr>
<tr><td>2.</td><td colspan="2">Create a wafer FOUP and set the proper wafer on the stage, then press “Align.”</td><td>You should get a score for all the fields and with a green tick.

Alignment profile
Name: GA_fast
Untitled
Edit... Load... Save as...
Mark location (X,Y): -2.365 -5.6505 [mm] Locate mark...
Optimize... Align Advanced settings...
Advanced Fine Alignment: Use machine setting<table>
<tr><th>Cell index X</th><th>Cell index Y</th><th>Wafer coordinates X [mm]</th><th>Wafer coordinates Y [mm]</th><th>Alignment type</th><th colspan="2">Score</th></tr>
<tr><td>2</td><td>0</td><td>113.8186</td><td>-5.6505</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
<tr><td>1</td><td>3</td><td>55.7268</td><td>94.9674</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
<tr><td>-1</td><td>3</td><td>-60.4568</td><td>94.9674</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
<tr><td>-2</td><td>0</td><td>-118.5486</td><td>-5.6505</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
<tr><td>-1</td><td>-3</td><td>-60.4568</td><td>-106.2684</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
<tr><td>1</td><td>-3</td><td>55.7268</td><td>-106.2684</td><td>Fine</td><td>✓</td><td>1.00</td></tr>
</table></td></tr>
<tr><td>3.</td><td colspan="2">Choose the correct wafer and set it on the stage for running the “AlignmentPerformanceTask.”</td><td>The task should get started and finished without any errors or exceptions.

WaferAlignmentPerformance
Processing:
Performing coarse align for the 1th time of all 1 times
Abort</td></tr>
<tr><td>4</td><td colspan="2">Recreate the Foup and create a process job after setting the wafer on the stage and initiate a process job.</td><td>It should get started and finished without any errors or exceptions. The pattern should be located almost in the center of the Darkfield camera spot, proving that wafer alignment is accurate enough.

Acquiring
a.08
Valid
Invalid
Alignment
Darkfield
Acquiring: In process
ACQUIRED 160/400 SCHEME TP_uDBO_uDBF... PJ PJ_20210720_183335_</td></tr>
<tr><td>5</td><td colspan="2">Repeat all these tests, using the same patch, but without the configuration file, to make sure that using the same patch, we can use Cognex. Afterward, check the performance time of running a process job with Cognex and RECOG. Additionally, check the difference between offset values computed using</td><td>1. The time for running a process job should be less than 2 seconds.
2. The difference between offset values measure by Cognex and RECOG should be less than 0.5 micrometers.</td></tr>
</table>

	RECOG and Cognex after running an alignmentPerformanceTask.	
TAR OK/NOK	*OK*	

RECOG additional accuracy test results

The accuracy test results of RECOG were demonstrated in Chapter 10, Section 10.3.1. This section demonstrates the result of comparison between the Cognex and RECOG results against a pattern called P (Figure 48 and Figure 49,) which has the same geometric shape as pattern D, but it is being used in a different layer.

Apart from the difference in the contrast of this layer images, since different photoresist material was used in this layer, the intensity level of pattern images were quite different from the D layer's images, which made the pattern recognition challenging. After looking at the images of this layer, we realized that for some of them the pattern (inside the pattern shape) was lighter than its surrounding, while for some of them it was vice versa. Sometimes only the pattern border was darker and inside the pattern shape was the same color as its surrounding. Finally, some images were too noisy. The algorithm should deal with all different situations.

As it can be seen, RECOG was successful in pattern recognition in all different situation. Therefore, we can say that RECOG is flexible and can conduct pattern recognition successfully, regardless of the image brightness, contrast, or even in the noisy images. Sometimes RECOG performed better than Cognex. For instance, by looking at Figure 49, we see that Cognex results are sometimes inaccurate (the ones who are quite different and far from the rest.) We manually tested these specific cases and compared the results and made sure that RECOG results were more accurate. Even without manuall checking, it can be seen that the Cognex values for these images are by far different from the majority, which could show a false detection.

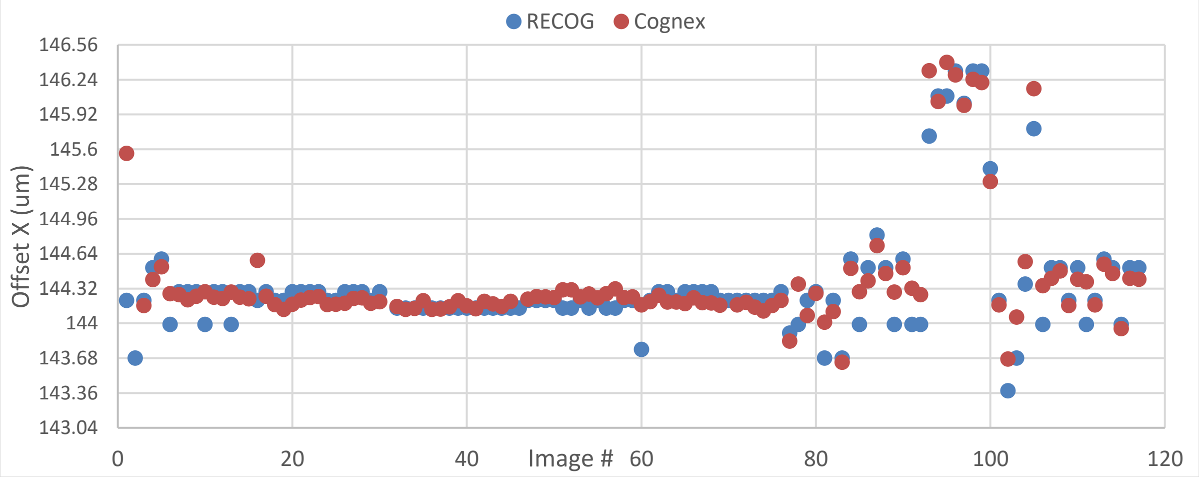


Figure 48. Accuracy comparison between Cognex and RECOG result for the pattern P, X-axis

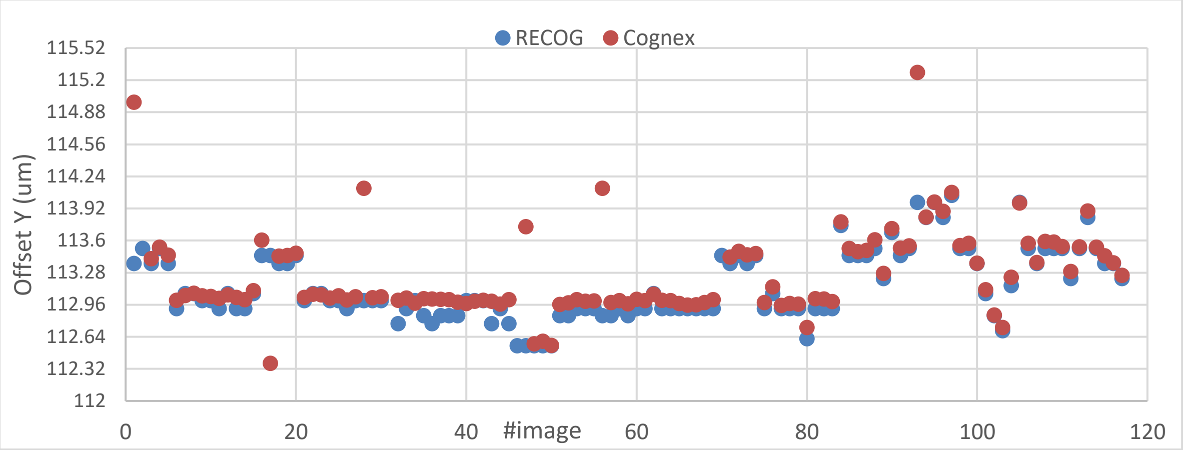
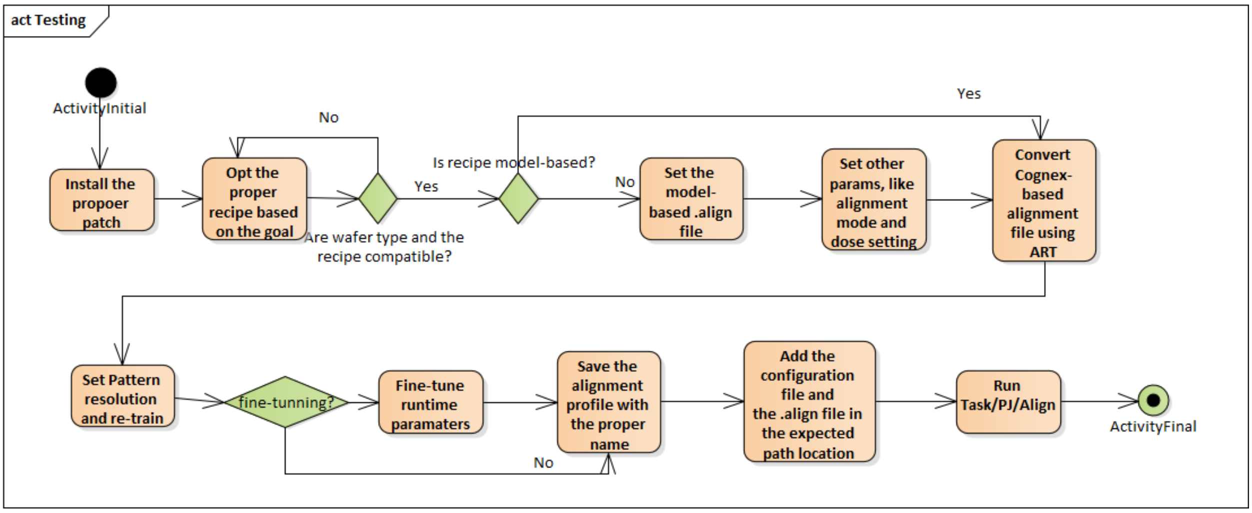


Figure 49. Accuracy comparison between Cognex and RECOG result for the pattern P, Y-axis

Machine test preparation

Figure 50 summarizes the test preparation steps. First, we needed to make sure the wafer type and the recipe were compatible. Depending on the test goal, there are different recipes available. Because we only addressed model-based recipes in this project, we needed to make sure that we were using a model-based recipe.

Image-based recipes can be converted to model-based ones if the model-based alignment profile of the same pattern is available. Replacing the alignment profile might change other recipe parameters such as alignment model (normal or fast) and dose settings. Hence, these parameters were set as well.



act Testing
ActivityInitial
Install the propoer patch
Opt the proper recipe based on the goal
No
Yes
Are wafer type and the recipe compatible?
Is recipe model-based?
No
Set the model-based .align file
Set other params, like alignment mode and dose setting
Yes
Convert Cognex-based alignment file using ART
Set Pattern resolution and re-train
fine-tunning?
Fine-tune runtime paramaters
Save the alignment profile with the proper name
No
Add the configuration file and the .align file in the expected path location
Run Task/PJ/Align
ActivityFinal

Figure 50. Machine test preparation steps

As mentioned earlier, in this project, instead of changing the recipe structure, we decided to read the pattern data from a separate alignment profile file located at a specific location. This alignment profile should be created using ART to contain the RECOG library objects. Hence, we first extracted the alignment profile from the model-based recipe and converted it to the new format using ART. If needed, users can fine-tune the parameters to get the best recognition result and so the best alignment result. Note that it is mandatory to set the proper pattern resolution and train the pattern based on the new resolution. Otherwise, the result might be inaccurate. After all, we saved the new RECOG-based alignment profile with a specific name. As the last step, we added the new configuration file at a specific location so that YS can run the alignment using RECOG.

RECOG accuracy analysis based on the Task result

After running the AlignmentPerformanceTest, we compared the result of measured offset using both libraries with 0-degree or 180-degree rotation. While we discussed the result of this comparison with the 180-degree rotation, this section demonstrates the same comparison result with no rotation (0-degree rotation.) Figure 51 and Figure 52 depicts the result in the X axis and Y axis, respectively. Similar to the 180-degree rotation, the difference is almost always less than one pixel, which is desired.

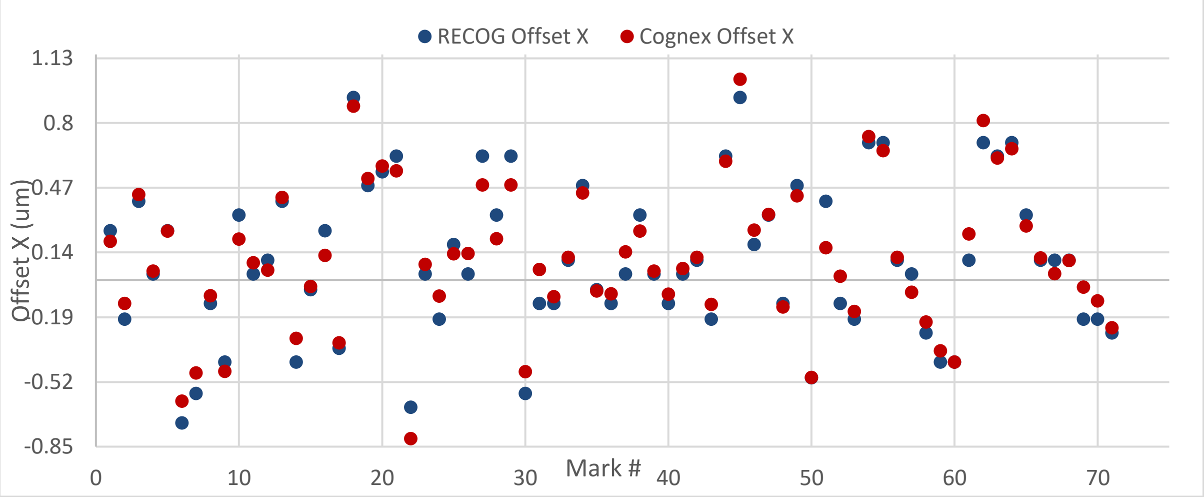


Figure 51. Offset comparison with no rotation for task execution, X axis

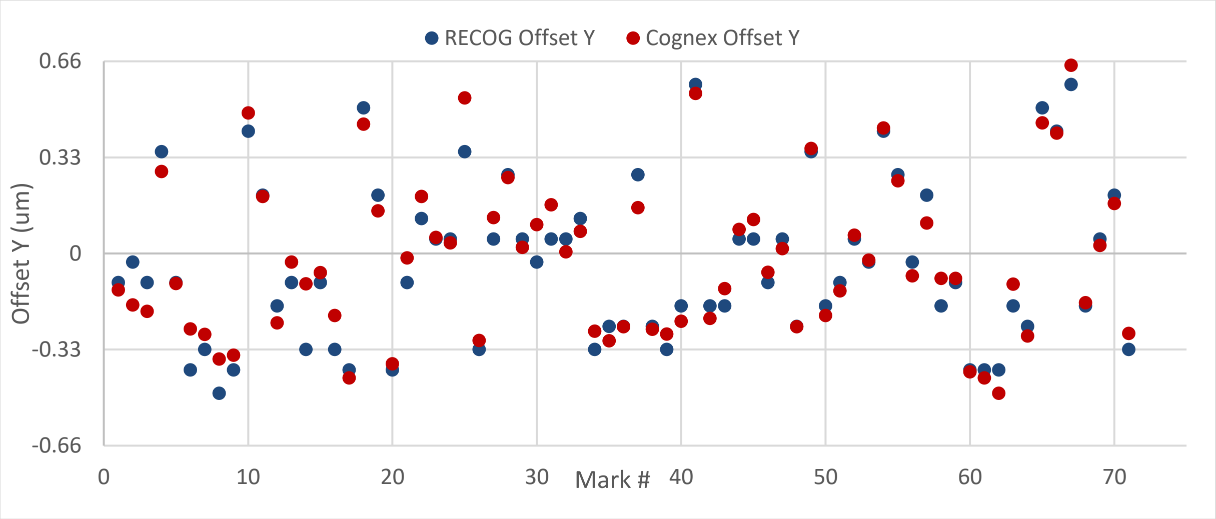


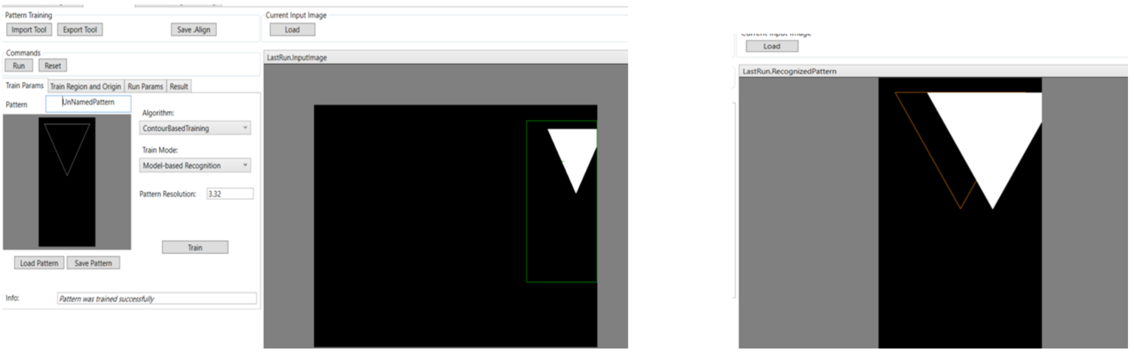
Figure 52. Offset comparison with no rotation for task execution, Y axis

Appendix C. Algorithm edge cases

This section reviews two edge cases and explains how the proposed solution deals with them.

Partial Pattern

In most cases, during the wafer alignment, the pattern is expected to be in the center of the input image. However, there might still be a few cases where the pattern resides partially in the input image. While the left side of the Figure 53a shows a pattern, on the right side of the same image, an image that partially contains this pattern can be seen. The problem with this situation is that although the algorithm might find the pattern, the reported offset might not be accurate, as shown in Figure 53b. The issue is mainly because the pattern slides through the input image during the template matching, pixel by pixel. Although it will find the most similar area to the template, it can still not find the offset accurately because the pattern and the image cannot be matched completely; therefore, a slight deviation from the correct offset is predictable.



Pattern: UnNamedPattern

Algorithm: ContourBasedTraining

Train Mode: Model-based Recognition

Pattern Resolution: 3.32

Info: Pattern was trained successfully

(a) (b)

Figure 53. Partial pattern

Usually, it is expected that the train region be smaller than the input image size. However, because the user defines the train region, it was safer to consider the possibility of the situation that the train region was bigger than the input image. The algorithm could face some issues in this case because the window size is bigger than the input image. To address this edge case, we introduced an image bounding box. It means that an additional marginal space equal to the template width and height is added to the image width and height. All these newly introduced pixels have a value equal to zero (black.) The downside of this approach is that it adds an overhead to the recognition performance, as a more extensive area should be scanned. Therefore, we added this feature as an optional feature to the algorithm and tool so that the user can decide if a partial pattern should be recognized or not.

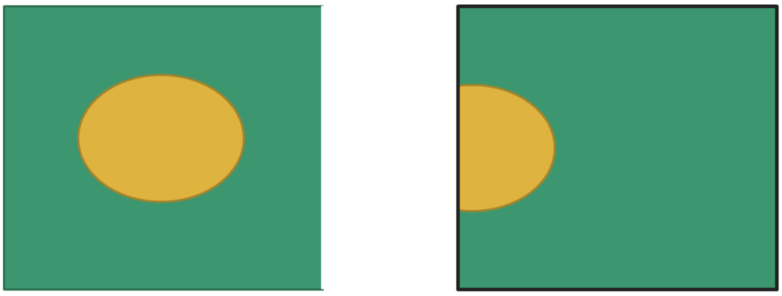
When using this feature, the recognition offset result is measured from the image bounding box. This means that an additional margin that is added to the input image impact the offset coordinate and should be subtracted from the recognition offset. *Equation 3* formulates how the final offset should be calculated.

$$\begin{aligned} Offset_x &= offset_x - TemplatePattern_{width} \\ Offset_y &= offset_y - TemplatePattern_{height} \end{aligned} \qquad \textit{Equation 3}$$

Imaginary Template Pattern Border

As discussed earlier, one of the steps during the training phase is to find the minimum or maximum area of the template pattern. Patterns could have different shapes. They could either consist of various detached shapes or have a single closed shape. If the user defines a partial section of a pattern area as the training region, the pattern might turn into an open shape. In this situation, the minimum and maximum area of the pattern are equal to the entire pattern area since no contours could be detected.

Consequently, during the recognition phase, the algorithm might not find any contours concerning the training data (the expected minimum or maximum contour area.) This situation could lead to a false negative recognition. To avoid this situation, we decided to add an imaginary border to the template pattern. While Figure 54a depicts a typical case that the pattern resides in the center, Figure 54b illustrates the edge case mentioned in this case. As it shows, adding the imaginary border helps with finding at least one contour, and therefore the minimum area is less than the actual pattern contour size.



(a) Entire pattern

(b) Partial pattern with imaginary border

Figure 54. Imaginary template pattern border

Appendix D. Image Processing

This appendix reviews the basic image processing techniques used in the proposed solution, by giving some sample images to help readers who might not be familiar with this topics. In the second section, a summary of the techniques discusses in Chapter 6 are given in a table format. Figure 55 shows a sample image used as a reference for demonstrating different techniques.

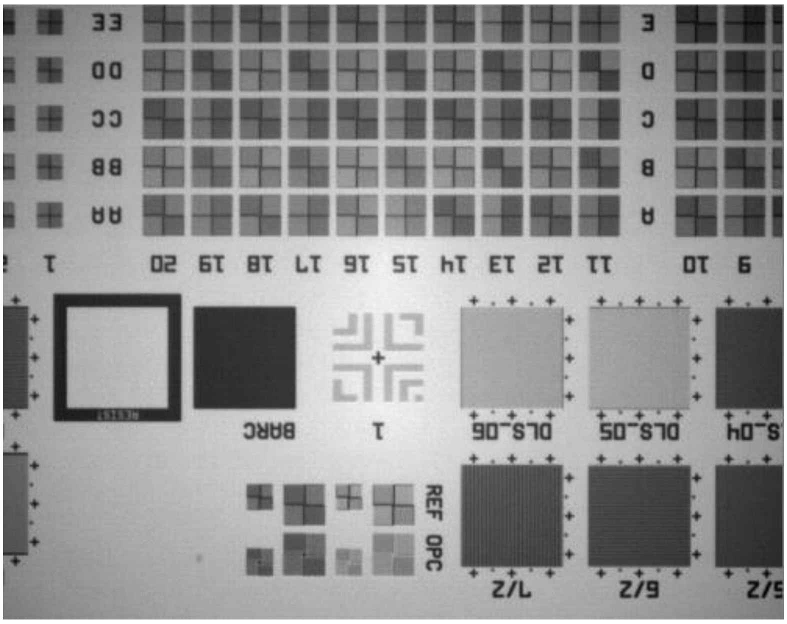


Figure 55. Sample Image with pattern G

Gaussian blurring
Figure 56 shows the effect of Gaussian blurring on the reference image. As it can be seen, this image is a bit blurry compared to the initial version. By increasing the kernel size, the image gets more blurry.

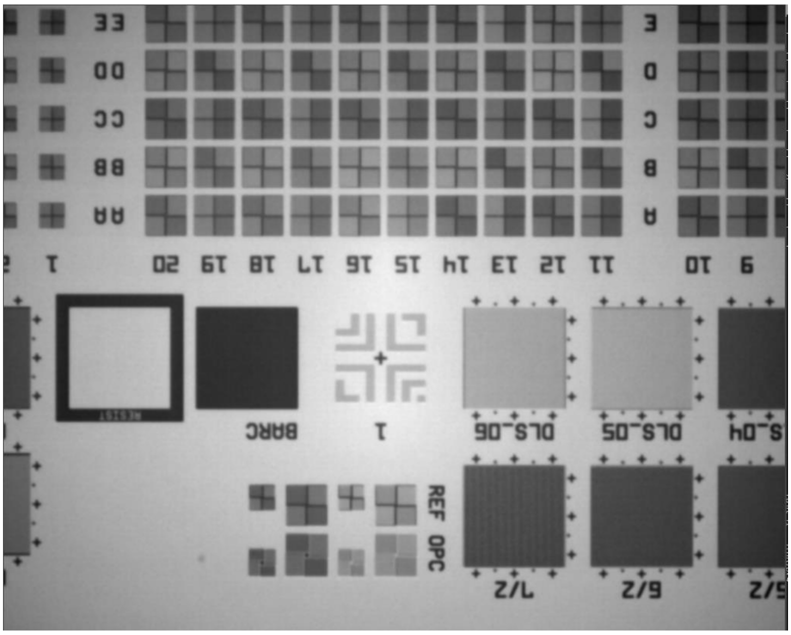


Figure 56. Gaussian blurring

Thresholding

As mentioned in Chapter 6, thresholding is the simplest method of segmenting images by converting the grayscale image into a binary one. A sample of this technique is shown in Figure 57.

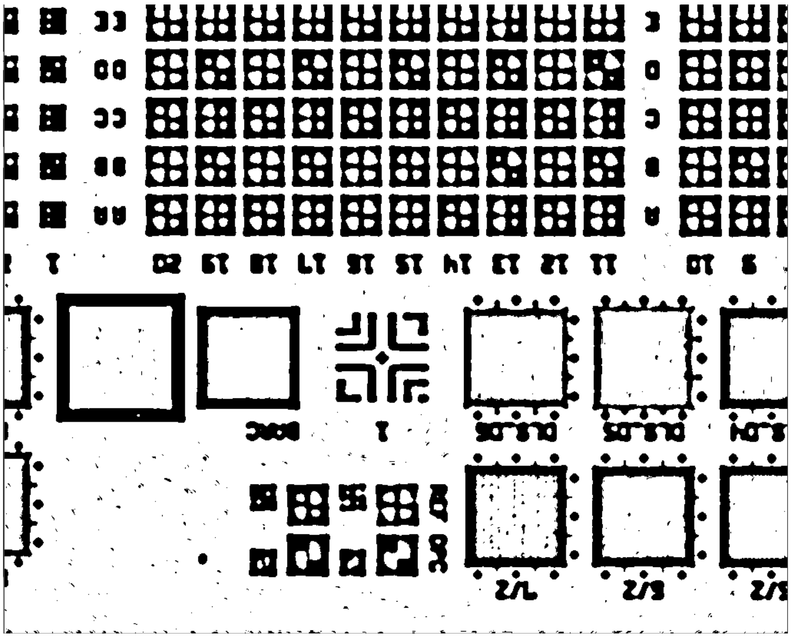


Figure 57. Reference image after applying thresholding

Edge Detection

Edge detection is a technique of image processing used to identify points in a digital image with discontinuities, simply to say, sharp changes in the image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image. Figure 58 shows the edges detected in the reference image.

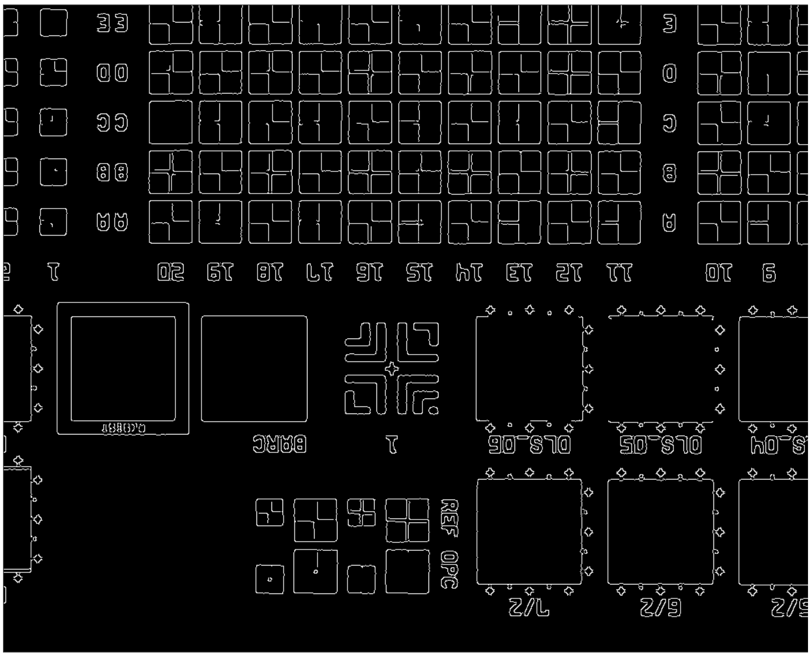


Figure 58. Result of edge detection

Dilation

Dilation expands the image pixels i.e. it is used for expanding an element A by using structuring element B. Dilation adds pixels to object boundaries. The value of the output pixel is the maximum value of all the pixels in the neighborhood. A sample of image dilation is depicted in Figure 59.

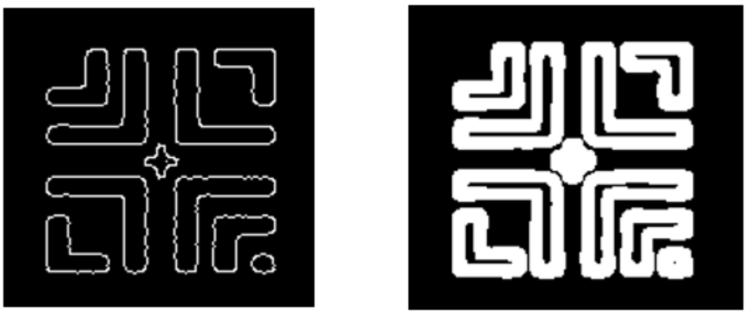


Figure 59. Initial image (left side) and dilated image (right side)

Interpolation

Image interpolation occurs when you resize or distort your image from one pixel grid to another. Zooming refers to increase the quantity of pixels, so that when you zoom an image, you will see more detail. Interpolation works by using known data to estimate values at unknown points. Image interpolation works in two directions, and tries to achieve a best approximation of a pixel's intensity based on the values at surrounding pixels. Figure 60 shows pattern G with its actual size. Figure 61 shows the interpolated image of the pattern G. while there are various methods for interpolation, we used cubic interpolation technique. Based on the experiments, we got the best results with this method.

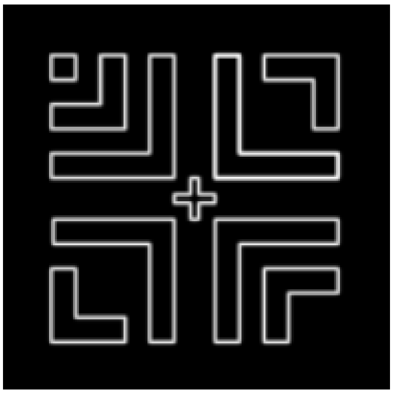


Figure 60. Pattern G

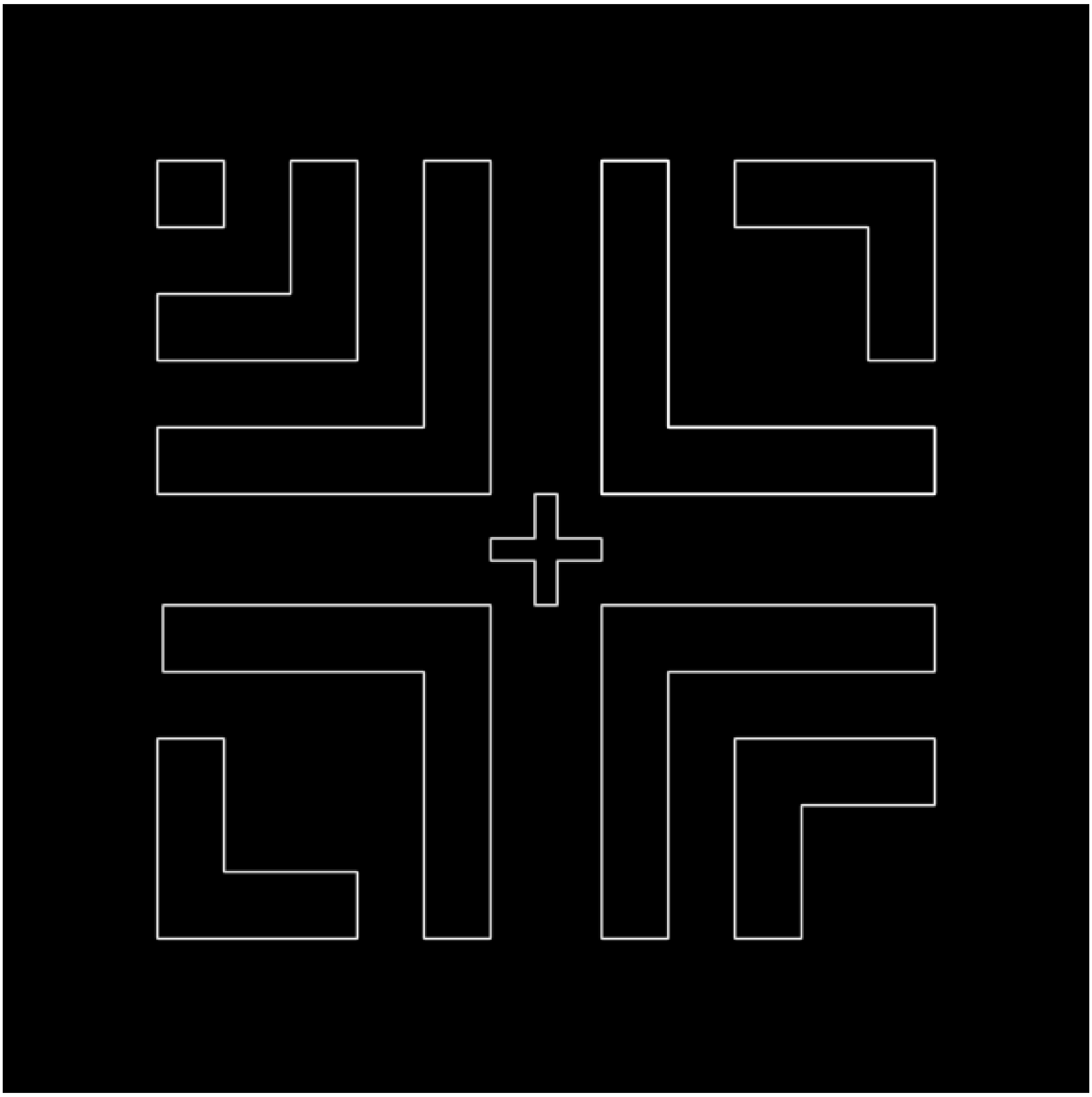


Figure 61. Interpolated pattern G

Image processing techniques summary

Table 22. Comparison of traditional and modern CV approaches

	Pros	**Cons**
Traditional Approaches [1, 2, 3]	• Customizable • No need for the training data • With highly fine-tuned parameters could localize the pattern accurately • Easy to learn, with easy concepts and techniques	• Highly dependent to the parametes such as Gaussian kernel size • Difficulty in detection in case of noisy and low-quality images

Modern Approaches (using ML/DL) [1, 2, 3]	• Can help in finding false positives • Could be a more generic approach (find patterns in noisy images better)	• Requires large annotated dataset • Mainly for classification • Performance in the cost of accuracy

Table 23. Object detection methods

	Pros	**Cons**	**Application**
Feature-based Matching [8]	• Efficient with large resolution images • Compute sub-pixel position	• Prone to false results in case of incorrect key point matching • Does not compute the score	Not a good choice if different objects share the same features or images have fewer features.
Template Matching [8]	• Compute similarity score • Simple and less parametric • Support different comparison methods	• Highly dependent on image intensity • Does not compute the sub-pixel position	A perfect fit when templates have no strong features with an image
Contour-based Matching	• Less parametric • Not dependent on image intensity	• No scoring • No support for sub-pixel accuracy	Not a good choice in the case of detached shapes

Table 24. Techniques for getting the sub-pixel position

	Pros	**Cons**
Feature Matching techniques	• Time-efficient	• Highly dependent on accurate feature mapping, otherwise could lead to inaccurate mapping (result)
Phase Correlation [12]	• Higher performance compared to the interpolation technique • computes confidence score	• High error rate, due to incorrect feature matching [1]
Interpolation [4]	• More reliable results (based on the experiments result)	• Not time efficient

About the Author

Raha Sadeghi received her Bachelor’s degree in Software Engineering from Alzahra University in 2010. As she found herself drawn by computer science, she started her Master's degree in the same field with a significant focus on Distributed Systems at Iran University of Science and Technology. Raha accomplished this degree in 2013 by proposing a trust model for service selection in a service-oriented environment as her final thesis. Upon graduation, she started working as a software developer at a national company in Iran. After five years of work experience, developing her technical knowledge, she started a new journey in 2019 by joining the PDEng program in Software Technology at the Technical University of Eindhoven to broaden her horizon. During the last two years, she did multiple projects at well-known companies, including CERN, TNO, ESA, and ASML, and experienced different test manager and software architect roles. She is mainly interested in system/software architect and design, as well as data Science.

PO Box 513
5600 Eindhoven
The Netherlands
tue.nl

PDEng SOFTWARE TECHNOLOGY

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY