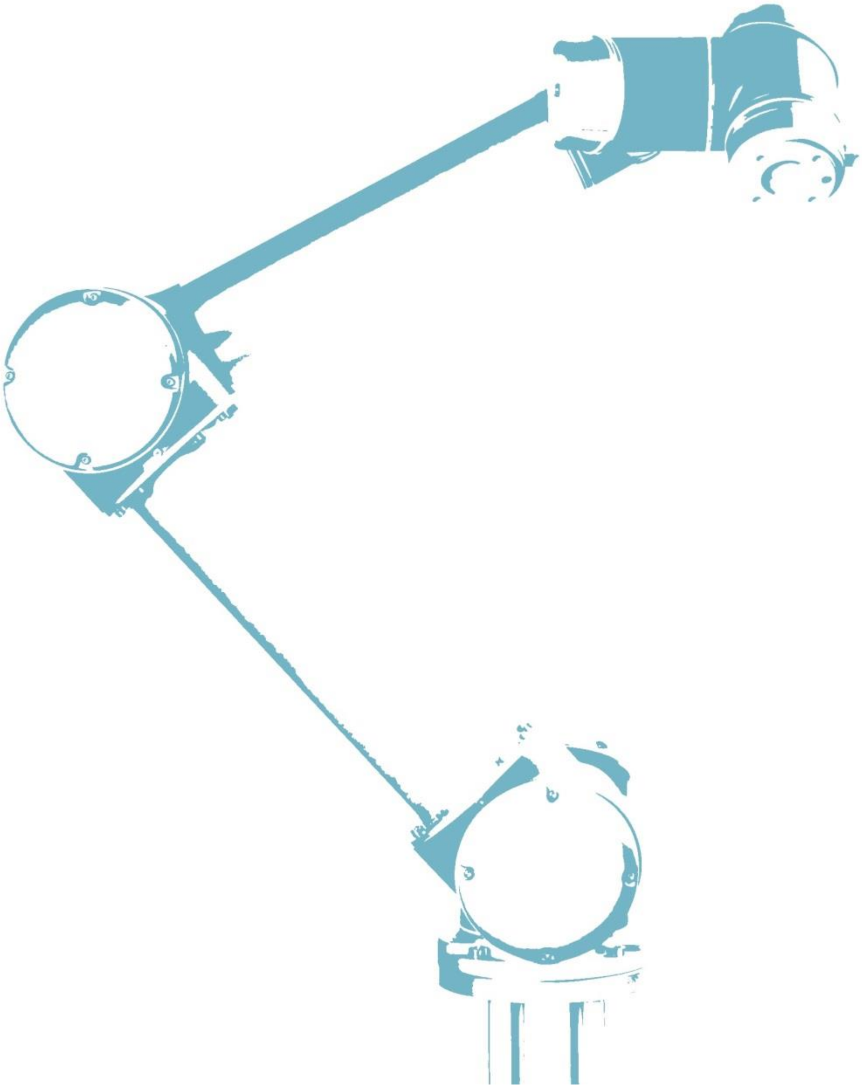
Project Vi'RA

Vision assisted Robotic Assembly



HØGSKOLEN I BUSKERUD OG VESTFOLD · PROFESJONSHØGSKOLEN ·

HBV

MEKTRON

ROBOTIKK

Kristian D. Kvam

Eirik F. Skjold

Ivar R. Leikarnes

Deividas Svaikauskas

Anders Godstad

Thor Eirik Johnsrud

Abstract

This is the final report from project Vi'RA (Vision assisted Robotic Assembly) at HBV Kongsberg spring 2015.

The introduction gives a summary of the project and links to relevant parts of the report.

The main table of contents shows the location of the documents. Each document has its own detailed table of contents and list of figures. A full list of figures can be found at the end of the document.

Contents

Abstract **3**

Contents **3**

Introduction **5**

A. Project documentation **6**

- **A.1** Project plan 6
- **A.2** Use case 18
- **A.3** Requirements 23
- **A.4** Requirement specification 44
- **A.5** Architectural overview 48
- **A.6** Test document 52
- **A.7** Test cases 63

B. Project management tools **121**

- **B.1** Cloud storage Services 121
- **B.2** Communication Technologies 124
- **B.3** Scheduling Software 127

C. Analysis **130**

- **C.1** Risk analysis and management 130
- **C.2** Workstation Risk analysis 140
- **C.3** Process analysis 148
- **C.4** Cost-Benefit Analysis 152
- **C.5** Assembly process alternatives 155

D. Technology documents **161**

D.1 Grippers 161

D.2 Screwdriver 175

D.3 Camera 180

D.4 URscript 183

D.5 Universal Robot 189

D.6 3D Modelling software 197

E. Concepts 200

E.1 Concept document 200

F. Product development documents 214

F.1 Regulator Jig 214

F.2 URcontrol 222

F.3 Cable connector mounting 235

F.4 End effector bracket 241

F.5 Gripper fingers 251

F.6 Starlock 266

F.7 Vacuum gripper tool 271

F.8 Vision development 275

G. Our work 286

G.1 Self-reflection 286

G.2 Activity breakdown 288

G.3 Potential improvements beyond this project 294

G.4 General Vision Conclusion 297

H. User manual 302

H.1 User Manual: UR Remote Control 302

I. Table of figures 312

Introduction

Our task (p. 10) was to automate parts of the assembly line for Engmark Meteor (p. 7), using the Universal Robot (p. 189) and a self-developed low-cost vision system (p. 275). This report will be used as a basis for Mektron AS (p. 7) to make a general automation method for later projects.

We chose unified process (p. 14) as our project model. This model gave us the flexibility needed to complete a task with the uncertainty that ours had. We used a Gantt diagram (p. 11) to plan our time usage. The tools we used during the project are found in project management tools (p. 121).

Early in the process, the whole group travelled on-site to Engmark Meteors factory. There we analysed the current process flow (p. 148) of the assembly line to identify potential points for automation. Based on our findings, we chose to automate the final assembly process of the griddle

Thus, we began defining the use cases (p. 18) for the system. This required a substantial amount of work to define, since the task given from our employer was unspecified and open for our input. We settled that our most important task to complete would be to fasten screws with vision and then expand the system to place components and fasten them. From our use cases, we derived our requirements (p. 23) and an architecture (p. 48) to have a better understanding of how our system would work.

To verify that we had achieved our requirements, we set up a test plan (p. 52) to shape our test cases (p. 63). An overview of the link between the requirements and the test cases can be found in the requirement specification (p. 44).

Due to the uncertainty related to our project, we chose to use resources on a thorough risk analysis (p. 130) to identify and reduce the risks. Risks when using the workstation (p. 140) are also discussed in this document. Another important part of project analysis is the economical part for the costumer in the form of a costs-benefit analysis (p. 152).

In the end of April, we changed to a two-arm solution, the documentation for this is found in the assembly process alternatives (p. 155) document.

Where we have used existing technology, we have explored the different options available and our conclusion as to which one is the most suited for our project can be found in the Technology documents (p. 161). Those components that we have developed or modified internally are described in the product development documents (p. 214). Not all ideas qualified to the final solution; those are described in concepts (p. 200). Our design includes options for expanding the system (p. 294)

A critical part of our solution is the gripper (p. 161). Our design gives the system its versatility, as conventional use of grippers is to have one tool for each task.

Finally, our reflections (p. 286) of the project give a summary and points to potential improvements of the project process.

A. Project documentation

A.1 Project plan

Abstract

This document explains the project plan for our project. It describes the main tasks that need to be done, constrains and assumptions, how we set up our project planning and documentation, and finally an overview of the group members.

Contents

Abstract **6**
Contents **6**
Figures **7**
A.1.1 Introduction **7**
A.1.2 Objectives **8**
A.1.3 Constraints and Assumptions **8**
A.1.3.1 Business constraints 8
A.1.3.2 Technical constraints 9
A.1.3.3 General assumptions 9
A.1.4 Goals **10**
A.1.5 Activities **11**
A.1.6 Schedule **11**
A.1.6.1 Gantt 11
A.1.6.2 Unified process 14
A.1.6.3 Budget 15
A.1.7 Project Members **16**
A.1.8 References **17**

Figures

Figure 1: Backlog Example 11
Figure 2: Gantt diagram 1 of 2 12
Figure 3: Gantt diagram 2 of 2 13
Figure 4: Unified Process Diagram 15
Figure 5: Expense Budget 15

A.1.1 Introduction

Our assignment is to automate parts of the assembly line at Engmark Meteor. We are supposed to accomplish this by implementing a robot system from Universal Robots and a low cost vision system. The owner of this project, Mektron AS, imports systems from Universal Robots and deliver solutions based on their products. Mektron AS wants to expand the field of application of the Universal Robot system and to explore the possibilities with implementation of a low cost vision system. As of today there are no low cost vision systems that meet all the needs of Mektron AS. We are therefore to explore the possibilities of using a Kinect or equivalent cameras with depth perception in a production process.

Engmark Meteor is a small family business located in Oslo. The company has existed since 1928 and specializes in making griddles (steketakker). The griddles they make are made almost exclusively from sheet metal and come in 2 sizes; 46 and 60 centimetres. The company separate themselves from the rest of the production industry in Norway in that they produce all the components themselves, meaning their only import is sheet metal and electrical components. Every component is assembled by hand, whether it is welding or screwing. This requires specialized equipment and skilled workers.

Meteor could have moved parts of or the whole production abroad to save time and money. A few years back, a former wholesaler of Engmark Meteor sent their product to China and ordered copies of the griddle. This eventually led to a lawsuit. To protect their product, maintain full control of the production and to keep the traditional family business going, they decided to keep the production here in Norway.

Unfortunately, it is hard for companies located in Norway to compete with companies abroad in the economy of today. The salaries of the employees increase several percent each year, meanwhile the value of the product does not. This is due to only minor changes of the product over the years, which give them no good reason to demand more money from the wholesaler. Because of this economical issue, the owner of Engmark Meteor, Håvid, sees automation as a possible solution to this problem.

They also face unique challenges because their sales are highly concentrated to the four months before Christmas. Due to limited storage, Meteor has to increase the production capacity during this period instead of having a relatively constant production rate throughout the year. It is imperative to hire additional personnel to meet their demand. The new workers require time and resources to be trained and are less effective than the permanent employees. This results in reduced overall profit per griddle.

A.1.2 Objectives

There are three main objectives when it comes to this project. The first objective is to develop a robot vision system for Mektron AS and implement it in a production process. If the system works, we will have created a good foundation for later improvement and customization. Since Engmark Meteor AS is a very important factor of the project, their need of optimization and automation in their production process, is the second objective of ours. When this objective is accomplished, we will have set up an example for Mektron AS, of how they should approach future projects for optimization and automation. The third objective is to execute a project on a bachelor level. This includes planning, organizing, developing, testing and working in a medium sized group. It is important that we show our expertise from different disciplines and use them to make right technology choices. These kinds of projects are highly depended on good communication with the customer and constant updating of the process. If we fulfil these three objectives we will have accomplished our project.

A.1.3 Constraints and Assumptions

A very important part of project management is to establish and define our constraints and assumptions. We decided to divide our constraints into two parts; business constraints and technical constraints.

A.1.3.1 Business constraints

We have set start and end dates for this project, which limits what we can accomplish. The start date is set to be the fifth of January and the end date is first of June. At the same time, there are a couple of set milestones along the project. These are as follow: first presentation, second presentation, final document hand in and finally the third presentation. These milestones require that certain documents and processes are finished, and therefore set certain time limitations for us. These milestones are to make sure we acquire a certain flow in our project, pushing us to work with requirements and solutions towards a finished project.

The first presentation is set to be on the 10th of January. At least two workdays prior to the presentation we have to send in all of our documentation. This is to give our supervisors and examiners time to prepare for our presentation. In this first phase of the project, we focus mostly on project planning, requirements and analysis. We are expected to deliver requirement specification, test specification, test plan, task description and a project plan at the end of the first phase. The first presentation will mainly focus around these documents and what we expecting to work on in phase two.

The second presentation is set to be on the 26th of March. Just like before the first presentation, we will deliver all of our work done to that point. In phase two we will focus on working with solutions for the requirements. A lot of research is needed to make right technology choices. A lot of time will go towards developing our vision system, designing the automation system and further planning of the project. We need to present what technology choices we have made up to this part, what we have tested and what changes we have made. We will also present the plan for further project work.

The final document hand in is on the 19th of May. At this point we will deliver our project. All of the documentation has to be handed in to be graded. Any plans of further work has to be planned, documented and delivered too. The only work that can be done after this point is preparation for the third presentation.

The third presentation will be held on the 1st of June. This presentation will be split in two parts. The first one will be held as a marketing part where we will present the project in a very simple matter. In this presentation we want to “sell our project”. We will have to explain the reasoning for our project in a way that would interest the people that have their production in Norway. Our solution has to be presented in an easily understandable matter. The second part will include the project execution from a technical view. Here we will present the hardware and software in a thorough matter.

We have some budget constraints that involve private funding. Most of the major components will be funded by Mektron AS, but driving, coffee and similar expenses must be funded by the group itself and so we have set the private budget to a total of 12 000 kr.

A.1.3.2 Technical constraints

The vision system is one of the main objectives in this project and there are several constrains regarding it. First of all, the vision system is defined to be a low cost solution, which limits our possibilities. The automation system will be fully dependent on the vision system for assistance. Limitations in the vision system might cause constraints of the automation systems functionality. As the vision system is supposed to be low cost, a cheap industrial solution might be considered if the consumer product does not suffice.

Mektron AS will provide us with robot systems from Universal Robots. These robots have a defined set of technical specifications that we must consider when analysing the possibilities of optimization and automation of the assembly process. The payload, area of operation and repeatability are the most important specifications of the robot. Since we are to implement a robot system that will collaborate with an assembler, we have to conduct a risk assessment for the collaboration of humans and robots. This may lead to further constraints when it comes to possible and profitable solutions.

A.1.3.3 General assumptions

When a group is working with a project it is normal to base some of the choices and solutions on assumptions. The flow of a project is often dependent on several aspects and we have to make assumptions of the time used or delivering dates. At the start of our project we gathered as a group and made a list of assumptions and expectations:

- We assume that each member of the group will spend around 550 hours of work during this project. This includes lectures, meetings, visits and general work.
- We assume that the UR10 arrives around the time of the first presentation. There will be a lot of project planning up to that point, but right after the presentation we need to start working with the robot.
- We assume that the vision hardware components will arrive some time before or at least at the same time as the robot.
- We will most likely order other hardware and so we expect that the delivery time of these components will not be too long.
- We are a group of six people and we assume that it will stay like this until the end of the project. You never know what might come up, but we wish to stay a full group for the sake of the project. Losing a member might have a negative impact on the process of the project.

- As a group we are dependent on each other's performance. We assume that each member can perform on a sufficient level.
- We have a limited time window to work with the project, and so we assume that the project planning will be executed successfully. It is crucial that each set milestone is finished on time.

A.1.4 Goals

Our overall goal with this project is to optimize the production at Engmark Meteor. To help us achieve this, we are to implement a robot system from Universal Robots and a robot vision system. By combining these two, we will make a system, which can assist in the assembly process. The vision system is primarily to assist the robot in localizing screw holes, but it may also help with localizing objects such as electrical components for the robot to assemble. We think this might lead to both lower costs and higher production capacity for Engmark Meteor.

We are to develop solutions to make the assembly process more suitable for the robot. This may or may not include changing the production line, develop specialized tools for the robot, and alter the product itself. The system needs to be user friendly, so that workers with minimal training may operate the system.

At Engmark Meteor the vision system is required as there will still be a human interaction with the product. Human interaction means that each individual product will not be identical. The system need to detect key points in a three-dimensional space regardless of deviation. This will be used primarily to assist in fastening screws, but can also be taken further to assist in other aspects of the assembly. Mektron has prior experience with Microsoft Kinect, and therefore it is natural that we explore the possibilities of taking use of this for our robot vision system. However, they are not against using a different camera solution if we find that it works better for this project.

For Mektron, their primary concern is their current client, Engmark Meteor, but they would also like to use parts of the system in other projects. This means that both the vision and the robotic solutions should be as modular as possible.

Mektron wants:

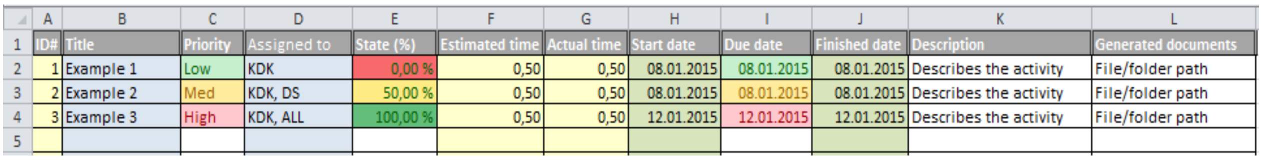
- Low cost robot vision system based on cheaper technology than current industrial solutions. It should be applicable to future projects.
- Develop a method to analyse and optimize production lines.
  - A general methodology for Mektron to use in future projects.
  - A specific methodology to apply to Engmark Meteor's production line.

Engmark Meteor wants:

- The product should retain its trademark design.
- The production should be more effective.
  - Reduce the average time it takes to produce a griddle.
  - Reduce the cost of producing a griddle.
  - Increase production capacity.

A.1.5 Activities

The backlog document is meant to hold all of our activities during this project. We made an Excel document which we divided into columns with ID number, title, priority, assigned to, state, estimated time, actual time, start date, due date, finished date, description and generated document for each of the activities. The *priority* column is divided into three different rankings: green for low priority, yellow for medium and red for high. In the *assigned to* column we first include the owner of the document and then the participants of the activity. The *state* defines how far we have come with the activity. At 0% the activity hasn't started yet, at 50% it's in working progress and at 100% it's finished. We set an estimate of time and an actual time for each of the activities. When it comes to due dates we have divided them into three priorities: green, yellow and red. Green dates are dates that have no/very little effect on other parts of the project if they are slightly delayed. Yellow dates are activities that might delay other parts of the project, but can also be altered if necessary. Red due dates are hard deadlines that cannot be altered. The final column, *Generated documents*, refers to the document name and its location in the project folder.



	A	B	C	D	E	F	G	H	I	J	K	L
1	ID#	Title	Priority	Assigned to	State (%)	Estimated time	Actual time	Start date	Due date	Finished date	Description	Generated documents
2	1	Example 1	Low	KDK	0,00 %	0,50	0,50	08.01.2015	08.01.2015	08.01.2015	Describes the activity	File/folder path
3	2	Example 2	Med	KDK, DS	50,00 %	0,50	0,50	08.01.2015	08.01.2015	08.01.2015	Describes the activity	File/folder path
4	3	Example 3	High	KDK, ALL	100,00 %	0,50	0,50	12.01.2015	12.01.2015	12.01.2015	Describes the activity	File/folder path
5												

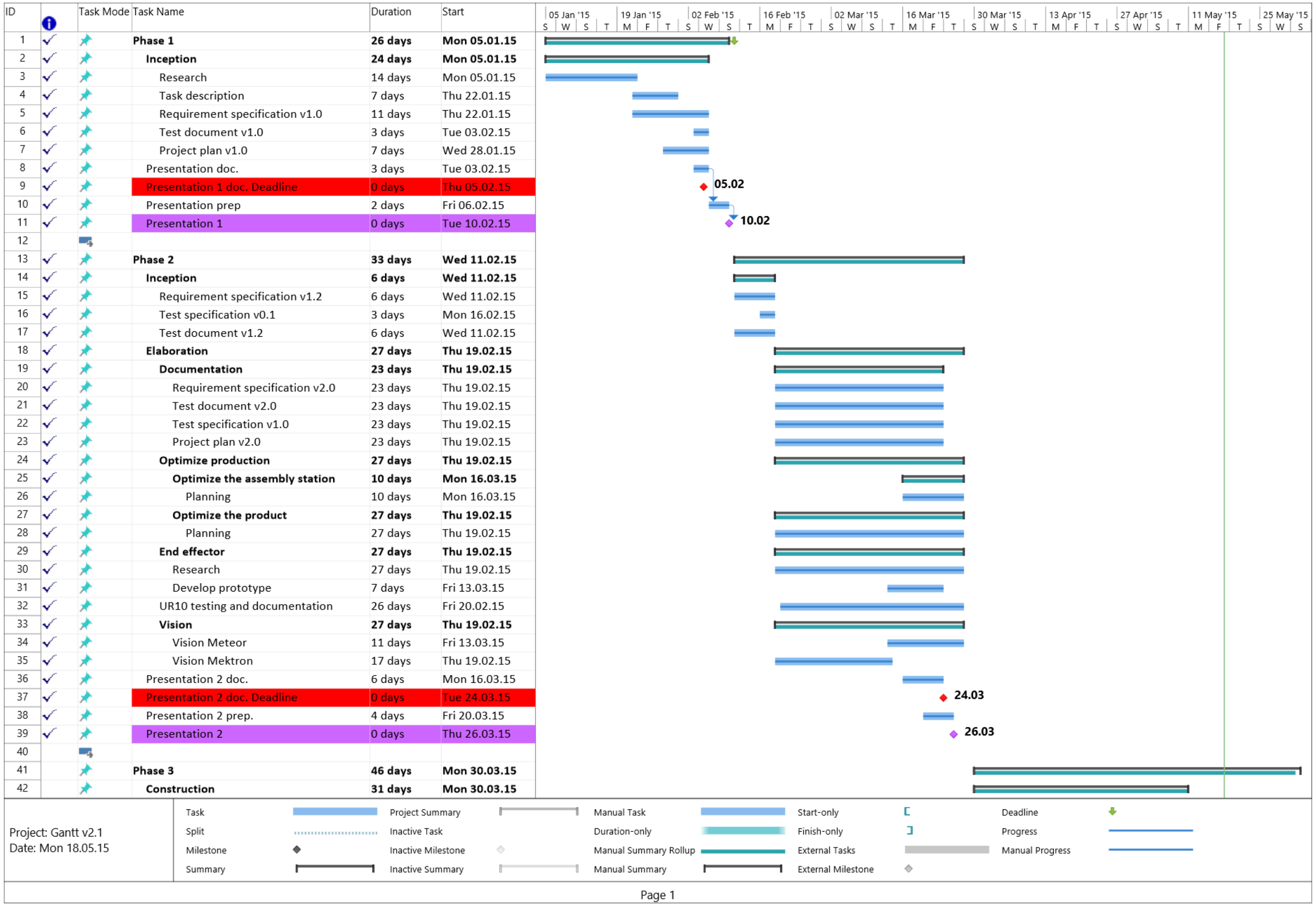
Figure 1: Backlog Example

Over we see an example of the said activities document (Figure 1). We made three different example activities to show the different rankings and naming of the activities.

A.1.6 Schedule

A.1.6.1 Gantt

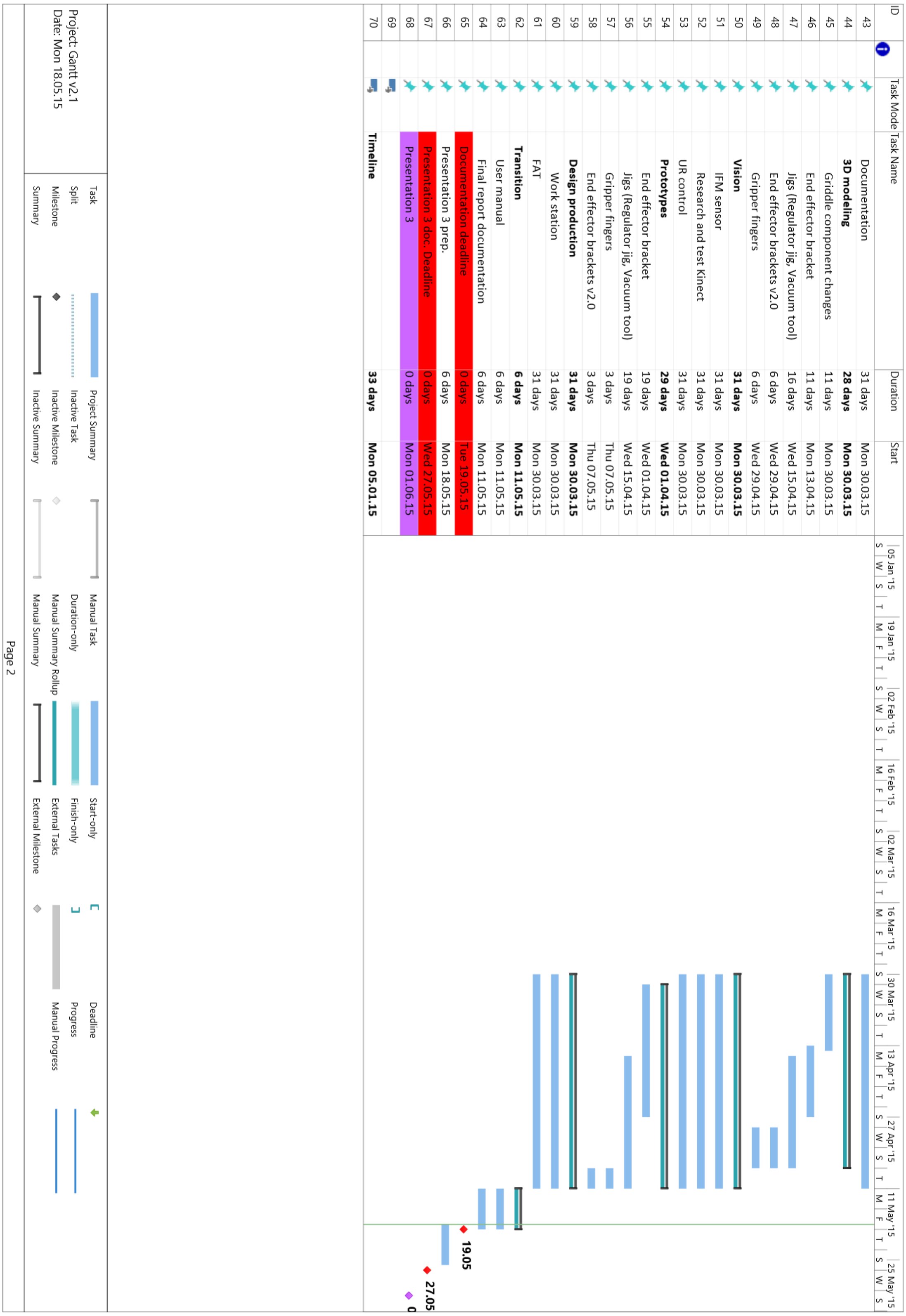
For this project we chose to use Gantt as a project planning tool. We divided the project into three phases. Each of the phases will include a presentation, document preparation for the presentation, milestones, important dates, tasks and percentage of completion. We assigned tasks and due dates to the different phases to create an overview of the whole project. We put a lot of thought into how we wanted to approach this project and decided that since we knew so little in the beginning, we will instead focus mostly on the work that needs to be done two weeks ahead of us. We set some deadlines a bit earlier than the actual deadline, to have time to complete the task if it is not completed. Our time planning was intentionally made with margins to give us flexibility in case of underestimation of a task. We will also merge in a development process framework in to the diagram, to give even better project execution. To make the Gantt diagram we used Microsoft Project.



ID	ℹ	Task Mode	Task Name	Duration	Start
1	✓		**Phase 1**	**26 days**	**Mon 05.01.15**
2	✓		**Inception**	**24 days**	**Mon 05.01.15**
3	✓		Research	14 days	Mon 05.01.15
4	✓		Task description	7 days	Thu 22.01.15
5	✓		Requirement specification v1.0	11 days	Thu 22.01.15
6	✓		Test document v1.0	3 days	Tue 03.02.15
7	✓		Project plan v1.0	7 days	Wed 28.01.15
8	✓		Presentation doc.	3 days	Tue 03.02.15
9	✓		Presentation 1 doc. Deadline	0 days	Thu 05.02.15
10	✓		Presentation prep	2 days	Fri 06.02.15
11	✓		Presentation 1	0 days	Tue 10.02.15
12					
13	✓		**Phase 2**	**33 days**	**Wed 11.02.15**
14	✓		**Inception**	**6 days**	**Wed 11.02.15**
15	✓		Requirement specification v1.2	6 days	Wed 11.02.15
16	✓		Test specification v0.1	3 days	Mon 16.02.15
17	✓		Test document v1.2	6 days	Wed 11.02.15
18	✓		**Elaboration**	**27 days**	**Thu 19.02.15**
19	✓		**Documentation**	**23 days**	**Thu 19.02.15**
20	✓		Requirement specification v2.0	23 days	Thu 19.02.15
21	✓		Test document v2.0	23 days	Thu 19.02.15
22	✓		Test specification v1.0	23 days	Thu 19.02.15
23	✓		Project plan v2.0	23 days	Thu 19.02.15
24	✓		**Optimize production**	**27 days**	**Thu 19.02.15**
25	✓		**Optimize the assembly station**	**10 days**	**Mon 16.03.15**
26	✓		Planning	10 days	Mon 16.03.15
27	✓		**Optimize the product**	**27 days**	**Thu 19.02.15**
28	✓		Planning	27 days	Thu 19.02.15
29	✓		**End effector**	**27 days**	**Thu 19.02.15**
30	✓		Research	27 days	Thu 19.02.15
31	✓		Develop prototype	7 days	Fri 13.03.15
32	✓		UR10 testing and documentation	26 days	Fri 20.02.15
33	✓		**Vision**	**27 days**	**Thu 19.02.15**
34	✓		Vision Meteor	11 days	Fri 13.03.15
35	✓		Vision Mektron	17 days	Thu 19.02.15
36	✓		Presentation 2 doc.	6 days	Mon 16.03.15
37	✓		Presentation 2 doc. Deadline	0 days	Tue 24.03.15
38	✓		Presentation 2 prep.	4 days	Fri 20.03.15
39	✓		Presentation 2	0 days	Thu 26.03.15
40					
41			**Phase 3**	**46 days**	**Mon 30.03.15**
42			**Construction**	**31 days**	**Mon 30.03.15**

Project: Gantt v2.1
Date: Mon 18.05.15

Figure 2: Gantt diagram 1 of 2



ID	Task Mode	Task Name	Duration	Start
43		Documentation	31 days	Mon 30.03.15
44		**3D modeling**	**28 days**	**Mon 30.03.15**
45		Griddle component changes	11 days	Mon 30.03.15
46		End effector bracket	11 days	Mon 13.04.15
47		Jigs (Regulator jig, Vacuum tool)	16 days	Wed 15.04.15
48		End effector brackets v2.0	6 days	Wed 29.04.15
49		Gripper fingers	6 days	Wed 29.04.15
50		**Vision**	**31 days**	**Mon 30.03.15**
51		IFM sensor	31 days	Mon 30.03.15
52		Research and test Kinect	31 days	Mon 30.03.15
53		UR control	31 days	Mon 30.03.15
54		**Prototypes**	**29 days**	**Wed 01.04.15**
55		End effector bracket	19 days	Wed 01.04.15
56		Jigs (Regulator jig, Vacuum tool)	19 days	Wed 15.04.15
57		Gripper fingers	3 days	Thu 07.05.15
58		End effector brackets v2.0	3 days	Thu 07.05.15
59		**Design production**	**31 days**	**Mon 30.03.15**
60		Work station	31 days	Mon 30.03.15
61		FAT	31 days	Mon 30.03.15
62		**Transition**	**6 days**	**Mon 11.05.15**
63		User manual	6 days	Mon 11.05.15
64		Final report documentation	6 days	Mon 11.05.15
65		Documentation deadline	0 days	Tue 19.05.15
66		Presentation 3 prep.	6 days	Mon 18.05.15
67		Presentation 3 doc. Deadline	0 days	Wed 27.05.15
68		Presentation 3	0 days	Mon 01.06.15
69				
70		**Timeline**	**33 days**	**Mon 05.01.15**

Project: Gantt v2.1
Date: Mon 18.05.15

Page 2

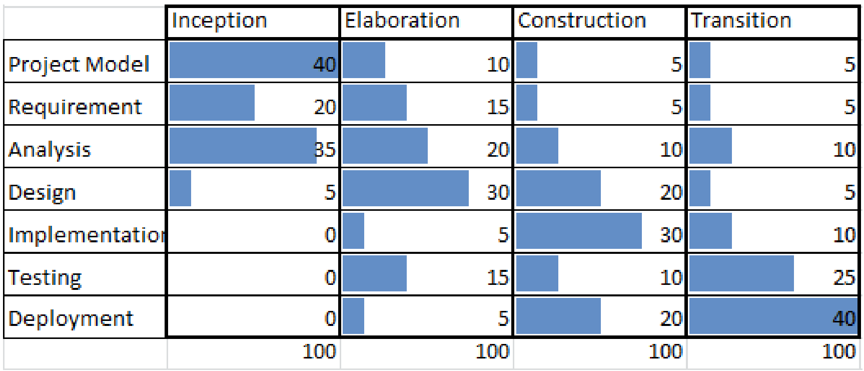
Figure 3: Gantt diagram 2 of 2

A.1.6.2 Unified process

We chose to use unified process as our development process framework. To make it fit our project, we decided to only use some of the framework and to modify it a bit. To understand what we have modified, first we must explain how unified process works. It is use-case-driven, which means that needs, architecture and iterations are driven by use cases (functional requirements). This process framework is architecture-centric, which means that the structural and dynamic dimensions are important factors for the development of our system. A well-organized architecture describes the elements of the system, their relationship, how these elements collaborate and interact to satisfy the purpose of the system and how the system behaves while in use. It is important to note that unified process is incremental and iterative. The iterations are divided into three different approaches: linear, sequential and iterative. During one of these iterations one must consider phase of the iteration, specific objectives, and satisfying the objectives within the context of the specific project. While working with this framework a good understanding of collaborations, contexts and interactions is needed. The collaboration part captures who does what activities on which part of the system. The context part focuses on the structural part of the collaboration, capturing how and when they should be done, what they produce and what they consume. Interaction part focuses on the dynamic aspect of the collaboration, capturing when and why such activities should be done and work products produced and consumed. It is important to mention that unified process is risk-confronting. The iterations explore high risks and gain feedback from previous iterations to confirm progress and discover other risks. These risks derive from use cases and architecture.

Unified process is divided into four phases: Inception, Elaboration, Construction and Transition. In the Inception phase the focus is to get to know what to build. It is expected to deliver a vision document, high-level project requirements, project scope, boundary conditions, and preliminary project schedule and cost estimate. In the Elaboration phase the focus is to get to know how to build it. It is expected to deliver detailed list of the requirements, an executable and stable architecture, verification of architectural qualities, updated risk profile and development plan, and plan for construction. In the Construction phase the focus in to build the system. It is expected to deliver a complete list of requirements, designed model, a documentation of the building process, test documentation, deliverance of the fully functional software and produced release description. In the Transition phase the focus is to deploy the system to end users. It is expected to deliver “bug-fixes”, update on the manuals, user training and a post-mortem project analysis.

As mentioned earlier we have altered unified process to fit our project. We changed business modelling to project modelling. Analysis and design got divided into two different developing disciplines. We have four phases in this process: Inception, Elaboration, Construction and Transition. In our case, the length of these phases might differ from the original framework. In our project the Inception and Elaboration phases might last longer than in other cases. Construction and Transition phases might merge a little. Some equipment might need a long period of time to develop, so the development might start already in Elaboration. Changes to the griddle might be designed and implemented in Elaboration phase too, so that we could adapt our system to these new changes. We decided to run Inception as a big iteration and then mostly run weekly iterations from the Elaboration phase. Our project has many uncertainties and risks. To approach them in a fast and safe way we decided that one week iterations would be perfect. This would give us an upper hand when it comes to identifying and managing the risks. With the modified unified process and Gantt diagram we are sure that our project will be planned well enough to be executed on a bachelor level.



	Inception	Elaboration	Construction	Transition
Project Model	40	10	5	5
Requirement	20	15	5	5
Analysis	35	20	10	10
Design	5	30	20	5
Implementatio	0	5	30	10
Testing	0	15	10	25
Deployment	0	5	20	40
	100	100	100	100

Figure 4: Unified Process Diagram

A.1.6.3 Budget

A projects budget usually consists of both pre-defined expenses and expenses that come up during the project. We didn't get any specific pre-defined expenses. All that we knew is that the solution must be low cost and so when we made the budget we always kept to the best and cheapest hardware.

Expenses	Description	Budget	Actual Cost
Office Supplies	Pen, paper, Post-It, etc...	kr 1 000	kr 1 177
Consumables	Coffe, etc...	kr 500	kr 768
Software	Licenses, etc...	kr 2 000	-
Vision hardware	Kinect 4 Windows, etc...	kr 2 000	kr 1 295
Gripper Development	Supplies for developing different grippers for UR	kr 1 000	kr 130
Transportation	Customer visits etc...*	kr 5 000	kr 3 730
Testing equipment	Materials, tools, machines, etc...	kr 1 000	kr 788
Miscellaneous	Website, etc...	-	kr 197
Total expenses		**kr 12 500**	**8085**

Figure 5: Expense Budget

In figure 3 we see our expense budget for the whole project. These are expenses we expect to take from our private funds. Some of the costs will be reimbursed at the end of the project.

A.1.7 Project Members

Name: Eirik F. Skjold

Age: 24 years

Responsibility: Web design and vision programming

E-mail: phenalor@gmail.com

Phone number: 41295192

Discipline: Software engineering

Name: Thor E. Johnsrud

Age: 21 years

Responsibility: Robot Control

E-mail: thor.e93@gmail.com

Phone number: 46679070

Discipline: Electrical engineering

Name: Ivar R. Leikarnes

Age: 22 years

Responsibility: Secretary, risk, testing

E-mail: ivar_leikarnes@hotmail.com

Phone number: 40610524

Discipline: Electrical engineering

Name: Kristian D. Kvam

Age: 30 years

Responsibility: Project manager

E-mail: kdkvam@hotmail.com

Phone number: 95903931

Discipline: Electrical engineering

Name: Anders Godstad

Age: 21 years

Responsibility: Document responsible

E-mail: anda_godstad@hotmail.com

Phone number: 99484428

Discipline: Electrical engineering

Name: Deividas Svaikauskas

Age: 22 years

Responsibility: Modelling, simulation, design

E-mail: d.svai93@gmail.com

Phone number: 46243385

Discipline: Mechanical engineering

A.1.8 References

1. Methods and tools. *Understanding the Unified Process (UP).* [13.02.2015] http://www.methodsandtools.com/archive/archive.php?id=32
2. Wikipedia. *Unified Process*. [29.01.2015] http://en.wikipedia.org/wiki/Unified_Process
3. Torbjørn Strøm & Olaf Hallan Graven, (2015). *Prosjekthåndbok ver 2015.* Kongsberg: Institutt for Ingeniørfag, HBV avdeling Kongsberg.

A.2 Use case

Abstract

This document contains the use scenario and use case diagram for the project.

Contents

Abstract **18**

Contents **18**

Figures **18**

A.2.1 Introduction **19**

A.2.2 User scenario **19**

A.2.3 Use Case, Operating **19**

A.2.4 Flowchart, Use case - Assembly process: Fastening **20**

A.2.5 Flowchart, Use case - Assembly process: Place and mount regulator **20**

A.2.6 Flowchart: Operator and assembly system cooperation **21**

A.2.7 Use Case, Exception Handling **22**

Figures

Figure 6: Use Case Diagram, Operator and assembly system interaction 19
Figure 7: Flowchart, Use Case - Assembly process: Fastening 20
Figure 8: Flowchart, Use case - Assembly process: Place and mount regulator 20
Figure 9: Flowchart, Operator and assembly system cooperation 21
Figure 10: Flowchart, Detailed Operator assembly 21
Figure 11: Flowchart, Detailed assembly system actions 22
Figure 12: Use Case, Exception Handling 22

A.2.1 Introduction

When working on a project like this, the user is a very important factor. It is therefore crucial to make user scenarios and use cases, to make sure you meet the user's requirements. To make sure we meet every requirement of the user, we make user scenarios. They give context of how a system might be interpreted, experienced and used. Since we follow unified process, which is use case driven, we have to make good and relevant use cases. Use cases are functional requirements, which are meant to drive iterations. Every iteration revolves around finding a solution for either one or several use cases.

A.2.2 User scenario

Håvid Engmark owns Engmark Meteor where he and his workers make griddles. At 7 AM Håvid arrives at work. He prepares the different stations for his workers, and starts the laser cutter. Håvid is worried, because with the current workstations, he can only make 65 griddles per day. It's extremely expensive to keep production in Norway, since foreign companies salaries are a lot lower than the Norwegian salaries. He wants to increase the griddle production, and to do this he feels like he needs to automate part of his production. Luckily, he has contact with a man who previously has sold him two robot arms, Egil Utheim. Egil comes up with a proposal to make a bachelor assignment to automate Håvid's production. A couple of students from HBV Kongsberg are on their final year of their bachelor's degree, and they are up for the task. They intend to automate his production using a Universal robot system and a low cost robot vision.

By automating part of his production, Håvid's profit will increase, and he will probably be able to keep his production in Norway. Additionally with the new automated assembly system installed, Håvid and the other assemblers do not have to do all the boring and tiresome bits anymore. Instead, they can rather focus on other parts of the production.

A.2.3 Use Case, Operating

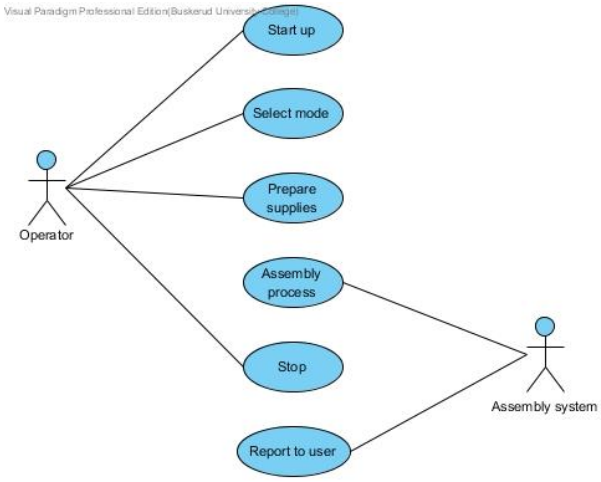


Figure 6: Use Case Diagram, Operator and assembly system interaction

This use case diagram shows the interaction between the operator and the assembly system.

A.2.4 Flowchart, Use case - Assembly process: Fastening

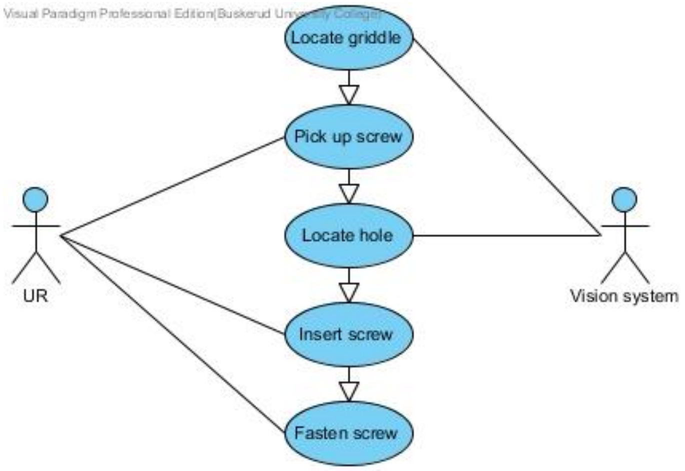


Figure 7: Flowchart, Use Case - Assembly process: Fastening

This flow chart shows the interaction between the Vision system and the Universal robot during an assembly process to fasten a screw.

A.2.5 Flowchart, Use case - Assembly process: Place and mount regulator

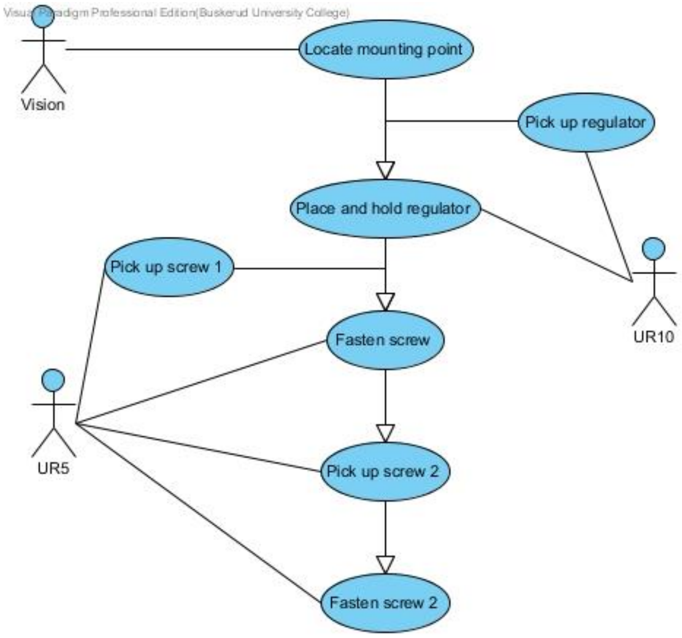


Figure 8: Flowchart, Use case - Assembly process: Place and mount regulator

This flow chart shows the interaction between the parts of the automation system during an assembly process to place and mount a regulator.

A.2.6 Flowchart: Operator and assembly system cooperation

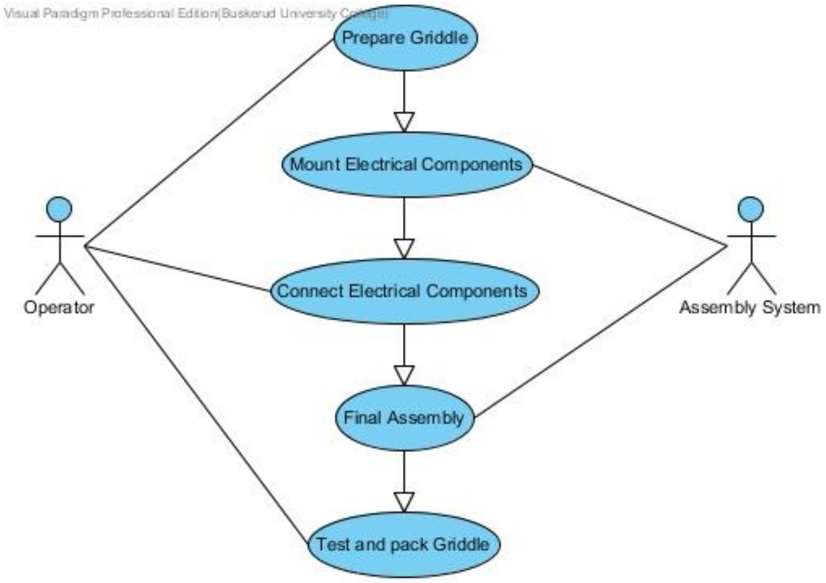

Visual Paradigm Professional Edition(Buskerud University College)
Prepare Griddle
Mount Electrical Components
Connect Electrical Components
Final Assembly
Test and pack Griddle
Operator
Assembly System

Figure 9: Flowchart, Operator and assembly system cooperation

This flowchart shows the cooperation between the assembly system and the operator. Note that the system will be designed in such a matter that there will be a small buffer between steps of the process so that multiple products will be in the system at one time. The operator will prepare one item, initiate the assembly systems first action and then prepare one more item while the assembly system works.

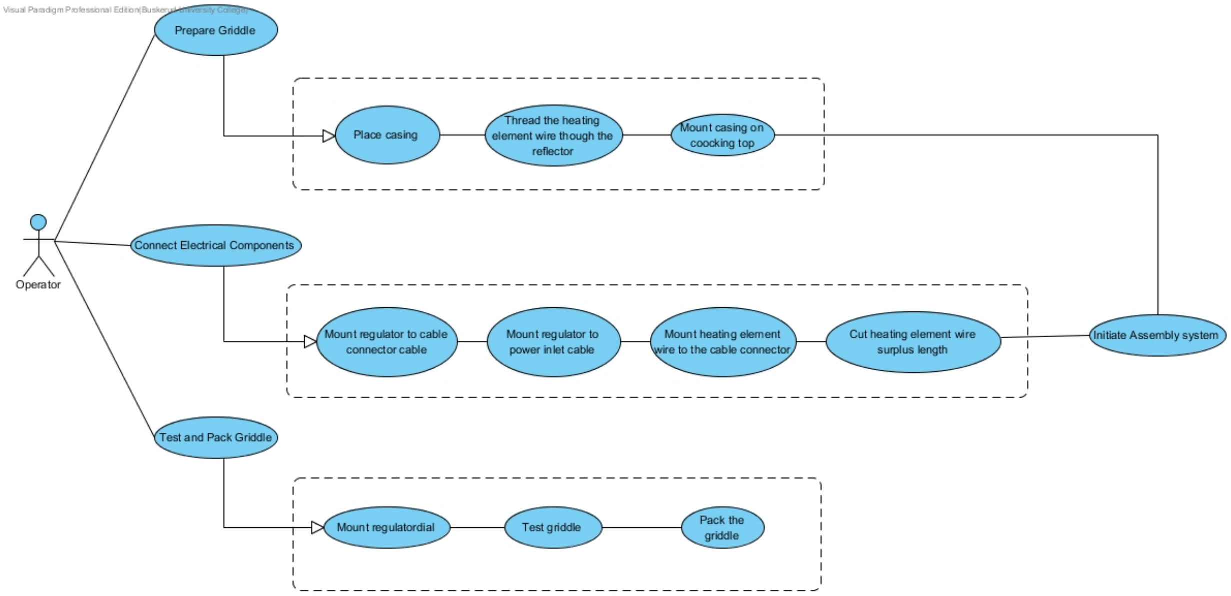

Visual Paradigm Professional Edition(Buskerud University College)
Prepare Griddle
Place casing
Thread the heating element wire though the reflector
Mount casing on coocking top
Operator
Connect Electrical Components
Mount regulator to cable connector cable
Mount regulator to power inlet cable
Mount heating element wire to the cable connector
Cut heating element wire surplus length
Initiate Assembly system
Test and Pack Griddle
Mount regulatordial
Test griddle
Pack the griddle

Figure 10: Flowchart, Detailed Operator assembly

This flowchart shows a detailed view of the operators’ assembly actions.

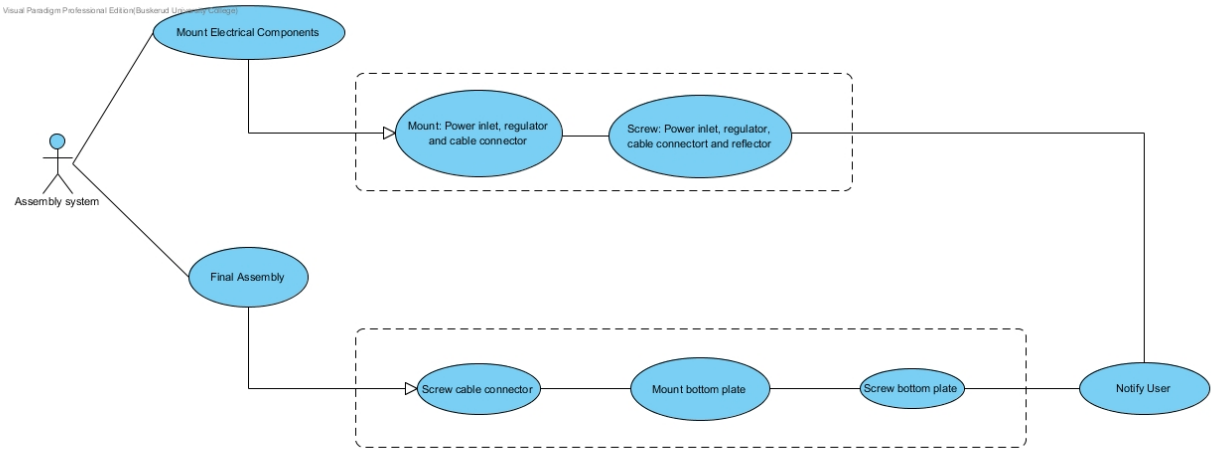

Visual Paradigm Professional Edition(Buskerud University College)
Mount Electrical Components
Mount: Power inlet, regulator and cable connector
Screw: Power inlet, regulator, cable connectort and reflector
Assembly system
Final Assembly
Screw cable connector
Mount bottom plate
Screw bottom plate
Notify User

Figure 11: Flowchart, Detailed assembly system actions

This flowchart shows a detailed view of the assembly systems assembly actions.

A.2.7 Use Case, Exception Handling

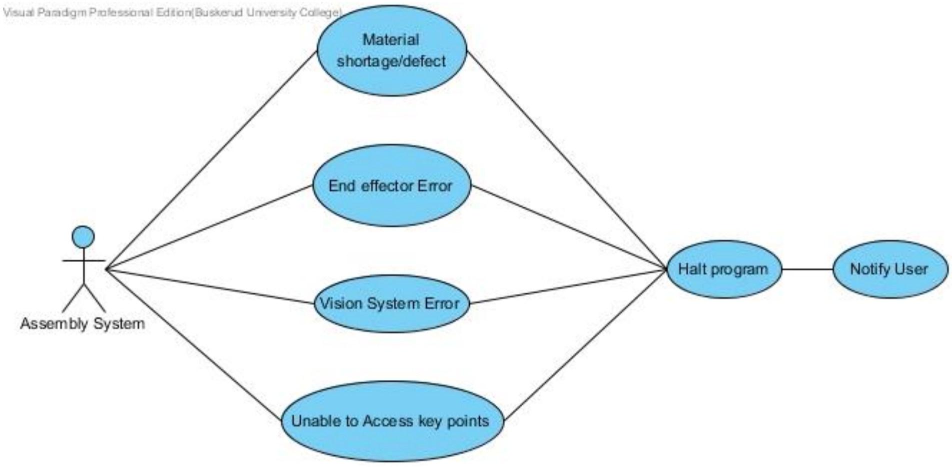


Figure 12: Use Case, Exception Handling

A.3 Requirements

Abstract

This is a summary of our needs and requirements. The requirements are sorted in relevant subsystems.

Contents

Abstract **23**

Contents **23**

A.3.1 Needs **24**

A.3.2 Requirements **27**

A.3.2.1 Automation system 27

A.3.2.1.1 Screw process 27

A.3.2.1.2 Cable connector 28

A.3.2.1.3 Power inlet 29

A.3.2.1.4 Regulator 30

A.3.2.1.5 Bottom plate 32

A.3.2.1.6 Quality 33

A.3.2.2 Vision system 35

A.3.2.2.1 Recognize screw holes 35

A.3.2.2.2 Advanced vision capabilities 38

A.3.2.2.3 Vision system requirements 39

A.3.2.3 Ease of use 41

A.3.2.4 General production 43

A.3.1 Needs

System Need ID	N01
Project Needs	The Production needs to be optimized
Rationale	Customer cannot increase price per product, and need to optimize production in terms of production time, production cost, man-hours
Parent ID	N/A
Source	Håvid Engmark, Engmark Meteor, customer
Priority	A
Acceptance Criteria	Satisfied customer
Dependencies	Somewhat successful project outcome

System Need ID	N01.01
Project Needs	Part of the production needs to be automated
Rationale	Customer needs to reduce man-hours
Parent ID	N01
Source	Håvid Engmark, customer
Priority	A
Acceptance Criteria	Automated TBD% part of the production process
Dependencies	Somewhat successful project outcome

System Need ID	N01.01.01
Project Needs	The automation-system should perform as well as an experienced assembler
Rationale	To achieve success, the system needs to be as good as a human, in terms of quality and to some degree speed
Parent ID	N01.01
Source	Håvid Engmark, customer
Priority	C
Acceptance Criteria	
Dependencies	

System Need ID	N01.01.02
Project Needs	The automation-system should use existing standards if possible and reasonable
Rationale	To ensure that we don't create new silly standards, eg. Power-supply
Parent ID	N01.01
Source	Håvid Engmark, customer Egil Utheim, Project owner
Priority	C
Acceptance Criteria	
Dependencies	

System Need ID	N01.01.03
Project Needs	The automation-system should be as user-friendly as possible
Rationale	It has to be less demanding to use the automation-system, than to do the task manually
Parent ID	N01.01
Source	Håvid Engmark, customer
Priority	B
Acceptance Criteria	An untrained person should be able to use the system after 30 min of training
Dependencies	

System Need ID	N01.02
Project Needs	The average time to build one unit from scratch should be shorter
Rationale	As it is today, Engmark Meteor does not have the capacity to produce as many units as the market demands
Parent ID	N01
Source	Håvid Engmark
Priority	C
Acceptance Criteria	Average time for one unit is decreased by TBD%
Dependencies	

System Need ID	N02
Project Needs	The griddles(steketakkene) should keep its traditional feeling and design
Rationale	Engmark Meteors market share is somewhat based upon old traditions, and their product is well-known for its classic design. Altering it too much might result in loss of market share
Parent ID	N/A
Source	Håvid Engmark
Priority	A
Acceptance Criteria	Satisfied customer
Dependencies	

System Need ID	N03
Project Needs	Most of the production should be kept at site
Rationale	Engmark Meteor is a family-business, and the owner feels that he will lose control on the product if too much of the production is outsourced
Parent ID	N/A
Source	Håvid Engmark
Priority	A
Acceptance Criteria	
Dependencies	

System Need ID	N04
Project Needs	Project owner needs documented methods for identifying possibilities to automating product assembly
Rationale	Project Owner imports robot arms to be used in production, and needs a method of identifying human actions that can be replaced by a robot arm
Parent ID	N/A
Source	Egil Utheim, Project owner
Priority	B
Acceptance Criteria	Satisfied Project Owner
Dependencies	Somewhat successful project outcome

System Need ID	N05
Project Needs	Project owner needs a low-cost vision-system that can interface with Universal Robots
Rationale	Project Owner wants to implement vision in the system he imports, but current solutions/technologies are way too expensive
Parent ID	N/A
Source	Egil Utheim, Project owner
Priority	B
Acceptance Criteria	A 3D vision system that can interface with UR, and recognize simple 3D-objects.Costs less than TBD
Dependencies	Sufficient accuracy in current "low-cost" technology

A.3.2 Requirements

A.3.2.1 Automation system

Requirement	The automation-system shall replace one or more human actions in the assembly process
Requirement ID	R08
Parent ID	N01.01
Rationale	Required by project owner and customer
Source	Håvid Engmark, Egil Utheim
Priority	A
Acceptance Criteria	At least one assembly-action is performed by the system
Test Case ID	

A.3.2.1.1 Screw process

Requirement	The automation-system shall be able to replace human screw process
Requirement ID	R08.01
Parent ID	R08
Rationale	Picking up and placing a screw is functionality that can be applied in many different assembly situations
Source	Egil Utheim
Priority	A
Acceptance Criteria	Achieved R08.01 children requirements and R08.09
Test Case ID	AS013

Requirement	The system shall be able to pick up a screw from a specific position and orientation
Requirement ID	R08.01.01
Parent ID	R08.01
Rationale	Being able to pick up a desired object is the first step in an assembly process
Source	Group
Priority	A
Acceptance Criteria	The system is able to pick up a screw from a specific position and orientation 20 times without failure.
Test Case ID	AS001

Requirement	The system shall be able to place the screw in a given hole with a TBD mm precision
Requirement ID	R08.01.02
Parent ID	R08.01
Rationale	Precise placement of components is vital to minimize error and achieve high quality
Source	Group
Priority	A
Acceptance Criteria	The system is able to place the screw in the given hole with a TBD mm precision
Test Case ID	AS002

Requirement	The system shall be able to fasten the screw
Requirement ID	R08.01.03
Parent ID	R08.01
Rationale	Fastening a component is a common assembly task
Source	Group
Priority	A
Acceptance Criteria	The system is able to fasten the screw
Test Case ID	AS003

A.3.2.1.2 Cable connector

Requirement	The system shall be able to place and fasten the E-CC (sukkerbit) in a specific position and orientation with TBD mm precision
Requirement ID	R08.02
Parent ID	R08
Rationale	Picking up and placing a E-CC is functionality that can be applied in many different assembly situations
Source	Egil Utheim
Priority	B
Acceptance Criteria	Achieved R08.02 children requirements and R08.09
Test Case ID	AS014

Requirement	The system shall pick up the E-CC
Requirement ID	R08.02.01
Parent ID	R08.02
Rationale	Being able to pick up a desired object is the first step in an assembly process
Source	Group
Priority	B
Acceptance Criteria	The system is able pick up the E-CC
Test Case ID	AS004

Requirement	The system shall place the E-CC with a TBD mm precision
Requirement ID	R08.02.02
Parent ID	R08.02
Rationale	Precise placement of components is vital to minimize error and achieve high quality
Source	Group
Priority	B
Acceptance Criteria	The system is able to place the E-CC with a TBD mm precision
Test Case ID	AS005

Requirement	The system shall fasten the E-CC with given screws
Requirement ID	R08.02.03
Parent ID	R08.02
Rationale	Fastening a component is a common assembly task
Source	Group
Priority	B
Acceptance Criteria	The system is able to fasten the E-CC with given screws
Test Case ID	AS006

Requirement	The system shall hold the E-CC in place while fastening
Requirement ID	R08.02.04
Parent ID	R08.02
Rationale	Holding a component while fastening it expands the automation systems capabilities towards full automation
Source	Group
Priority	B
Acceptance Criteria	The system is able to hold the E-CC in place while fastening it
Test Case ID	AS023

A.3.2.1.3 Power inlet

Requirement	The system shall be able to place and fasten the E-PI (strømledning) in a specific position and orientation with TBD mm precision
Requirement ID	R08.03
Parent ID	R08
Rationale	Picking up and placing a E-PI is functionality that can be applied in many different assembly situations
Source	Egil Utheim
Priority	B
Acceptance Criteria	Achieved R08.03 children requirements and R08.09
Test Case ID	AS015

Requirement	The system shall pick up the E-PI
Requirement ID	R08.03.01
Parent ID	R08.03
Rationale	Being able to pick up a desired object is the first step in an assembly process
Source	Group
Priority	B
Acceptance Criteria	The system is able to pick up the E-PI
Test Case ID	AS007

Requirement	The system shall place the E-PI with a TBD mm precision
Requirement ID	R08.03.02
Parent ID	R08.03
Rationale	Precise placement of components is vital to minimize error and achieve high quality
Source	Group
Priority	B
Acceptance Criteria	The system is able to place the E-PI with a TBD mm precision
Test Case ID	AS008

Requirement	The system shall fasten the E-PI with given screws
Requirement ID	R08.03.03
Parent ID	R08.03
Rationale	Fastening a component is a common assembly task
Source	Group
Priority	B
Acceptance Criteria	The system is able to fasten the E-PI with given screws
Test Case ID	AS009

A.3.2.1.4 **Regulator**

Requirement	The system shall be able to place and fasten the E-REG (regulator) in a specific position and orientation with TBD mm precision
Requirement ID	R08.04
Parent ID	R08
Rationale	Picking up and placing a E-REG is functionality that can be applied in many different assembly situations
Source	Egil Utheim
Priority	B
Acceptance Criteria	Achieved R08.04 children requirements and R08.09
Test Case ID	AS016

Requirement	The system shall pick up the E-REG
Requirement ID	R08.04.01
Parent ID	R08.04
Rationale	Being able to pick up a desired object is the first step in an assembly process
Source	Group
Priority	B
Acceptance Criteria	The system is able to pick up the E-REG
Test Case ID	AS010

Requirement	The system shall place the E-REG with a TBD mm precision
Requirement ID	R08.04.02
Parent ID	R08.04
Rationale	Precise placement of components is vital to minimize error and achieve high quality
Source	Group
Priority	B
Acceptance Criteria	The system is able to place the E-REG with a TBD mm precision
Test Case ID	AS011

Requirement	The system shall fasten the E-REG with given screws
Requirement ID	R08.04.03
Parent ID	R08.04
Rationale	Fastening a component is a common assembly task
Source	Group
Priority	B
Acceptance Criteria	The system is able to fasten the E-REG with given screws
Test Case ID	AS012

Requirement	The system shall hold the E-REG in place while fastening
Requirement ID	R08.04.04
Parent ID	R08.04
Rationale	Holding a component while fastening it expands the automation systems capabilities towards full automation
Source	Group
Priority	B
Acceptance Criteria	The system is able to hold the E-REG in place while fastening it
Test Case ID	AS024

A.3.2.1.5 Bottom plate

Requirement	The system shall be able to place and fasten the M-BP (bottom plate) in a specific position and orientation with TBD mm precision
Requirement ID	R08.05
Parent ID	R08
Rationale	Picking up and placing a M-BP is functionality that can be applied in many different assembly situations
Source	Egil Utheim
Priority	B
Acceptance Criteria	Achieved R08.05 children requirements and R08.09
Test Case ID	AS029

Requirement	The system shall pick up the M-BP
Requirement ID	R08.05.01
Parent ID	R08.05
Rationale	Being able to pick up a desired object is the first step in an assembly process
Source	Group
Priority	B
Acceptance Criteria	The system is able to pick up the M-BP
Test Case ID	AS025

Requirement	The system shall place the M-BP with a TBD mm precision
Requirement ID	R08.05.02
Parent ID	R08.05
Rationale	Precise placement of components is vital to minimize error and achieve high quality
Source	Group
Priority	B
Acceptance Criteria	The system is able to place the M-BP with a TBD mm precision
Test Case ID	AS026

Requirement	The system shall fasten the M-BP with given screws
Requirement ID	R08.05.03
Parent ID	R08.05
Rationale	Fastening a component is a common assembly task
Source	Group
Priority	B
Acceptance Criteria	The system is able to fasten the M-BP with given screws
Test Case ID	AS027

Requirement	The system shall hold the M-BP in place while fastening
Requirement ID	R08.05.04
Parent ID	R08.05
Rationale	Holding a component while fastening it expands the automation systems capabilities towards full automation
Source	Group
Priority	B
Acceptance Criteria	The system is able to hold the M-BP in place while fastening it
Test Case ID	AS028

A.3.2.1.6 Quality

Requirement	The system shall perform within a given quality
Requirement ID	R08.09
Parent ID	N02
Rationale	Quality in the assembly process is crucial for the end product
Source	Håvid Engmark, Group, Egil Utheim
Priority	A
Acceptance Criteria	Achieved R08.09 children requirements of priority A
Test Case ID	See children requirements of priority A

Requirement	The automation-system shall be able to perform the assembly-action 10(TBD) times without an error
Requirement ID	R08.09.01
Parent ID	R08.09
Rationale	The system should be able to run without constant operator supervision
Source	Group
Priority	A
Acceptance Criteria	Perform assembly-action TBD times with TBD mm precision
Test Case ID	AS022

Requirement	The automation-system shall be able to to continue from where it left off after an interruption
Requirement ID	R08.09.03
Parent ID	R08.09
Rationale	Ease of use, stability
Source	N01.01.01, N01.01.03
Priority	C
Acceptance Criteria	The system continues from where it left off after the interruption with no operator interaction
Test Case ID	AS017

Requirement	The automation-system shall complete an action with the same precision as an experienced assembler regardless of time spent
Requirement ID	R08.09.04A
Parent ID	R08.09, N01.01.01
Rationale	To fully replace a human action, the system needs to produce items of at least the same quality
Source	Group
Priority	A
Acceptance Criteria	The assembler is unable to identify that the automation system has assembled the product. Less than 1mm (TBD) shift in x,y,z Mounts objects with equivalent force
Test Case ID	AS018

Requirement	The automation-system shall complete an action as fast as an experienced assembler
Requirement ID	R08.09.04B
Parent ID	R08.09, N01.01.01
Rationale	To fully replace a humans action, the system needs to complete actions as fast as the human would
Source	Group
Priority	B
Acceptance Criteria	The assembler is unable to identify that the automation system has assembled the product. The system uses the same time amount to finish the task as an experienced assembler (According to time-analysis (see doc.))
Test Case ID	AS019

Requirement	The automation-system shall complete an action 20% (TBD) faster than a experienced assembler
Requirement ID	R08.09.04C
Parent ID	R08.09, N01.01.01
Rationale	More efficient system
Source	Group
Priority	C
Acceptance Criteria	The assembler is unable to identify that the automation system has assembled the product. The system uses the same time amout to finish the task 20% TBD faster than an experienced assembler does. (According to time-analysis (see doc.))
Test Case ID	AS020

A.3.2.2 Vision system

Requirement	The automation-system shall implement lowcost vision-control
Requirement ID	R02
Parent ID	N05
Rationale	Required by project owner
Source	Egil Utheim
Priority	A
Acceptance Criteria	Achieved at least req. R02.02, R02.04
Test Case ID	0

A.3.2.2.1 Recognize screw holes

Requirement	The vision-system shall recognize predefined objects with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A
Parent ID	R02
Rationale	"Low entry" level for recognizing objects, necessary to locate specific objects
Source	Egil Utheim
Priority	A
Acceptance Criteria	Achieved R02.01A children A-requirements
Test Case ID	0

Requirement	The vision-system shall recognize SH-BP with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A.01
Parent ID	R02.01A
Rationale	Necessary to achieve requirement R02.01A
Source	Group
Priority	A
Acceptance Criteria	Vision system recognizes SH-BP 9 out of 10 times
Test Case ID	VS001

Requirement	The vision-system shall recognize SH-REF with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A.02
Parent ID	R02.01A
Rationale	Necessary to achieve requirement R02.01A
Source	Group
Priority	A
Acceptance Criteria	Vision system recognizes SH-REF 9 out of 10 times
Test Case ID	VS001

Requirement	The vision-system shall recognize SH-CC with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A.03
Parent ID	R02.01A
Rationale	Necessary to achieve requirement R02.01A
Source	Group
Priority	A
Acceptance Criteria	Vision system recognizes SH-CC 9 out of 10 times
Test Case ID	VS001

Requirement	The vision-system shall recognize SH-REG with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A.04
Parent ID	R02.01A
Rationale	Necessary to achieve requirement R02.01A
Source	Group
Priority	A
Acceptance Criteria	Vision system recognizes SH-REG 9 out of 10 times
Test Case ID	VS001

Requirement	The vision-system shall recognize SH-PI with a given orientation within 0.4 meter of the sensor
Requirement ID	R02.01A.05
Parent ID	R02.01A
Rationale	Necessary to achieve requirement R02.01A
Source	Group
Priority	A
Acceptance Criteria	Vision system recognizes SH-PI 9 out of 10 times
Test Case ID	VS001

Requirement	The vision-system shall recognize predefined objects in any orientation within 0.4 meter of the sensor
Requirement ID	R02.01B
Parent ID	R02
Rationale	To create a dynamic system that can perform different tasks without a strict setup designed for automation
Source	Egil Utheim
Priority	B
Acceptance Criteria	Achieved R02.01B children B-requirements
Test Case ID	0

Requirement	The vision-system shall recognize SH-BP in any orientation within 0.4 meter of the sensor
Requirement ID	R02.01B.01
Parent ID	R02.01B
Rationale	Necessary to achieve requirement R02.01B
Source	Group
Priority	B
Acceptance Criteria	Vision system recognizes SH-BP from different angles, 9 out of 10 times
Test Case ID	VS004

Requirement	The vision-system shall recognize SH-REF in any orientation within 0.4 meter of the sensor
Requirement ID	R02.01B.02
Parent ID	R02.01B
Rationale	Necessary to achieve requirement R02.01B
Source	Group
Priority	B
Acceptance Criteria	Vision system recognizes SH-REF from different angles, 9 out of 10 times
Test Case ID	VS004

Requirement	The vision-system shall recognize SH-CC in any orientation within 0.4 meter of the sensor
Requirement ID	R02.01B.03
Parent ID	R02.01B
Rationale	Necessary to achieve requirement R02.01B
Source	Group
Priority	B
Acceptance Criteria	Vision system recognizes SH-CC from different angles, 9 out of 10 times
Test Case ID	VS004

Requirement	The vision-system shall recognize SH-REG in any orientation within 0.4 meter of the sensor
Requirement ID	R02.01B.04
Parent ID	R02.01B
Rationale	Necessary to achieve requirement R02.01B
Source	Group
Priority	B
Acceptance Criteria	Vision system recognizes SH-REG from different angles, 9 out of 10 times
Test Case ID	VS004

A.3.2.2.2 Advanced vision capabilities

Requirement	The vision-system shall recognize the orientation of an object
Requirement ID	R02.01.02
Parent ID	R02.01
Rationale	The system has to know how an object is oriented to be able to pick it up
Source	Egil Utheim
Priority	B
Acceptance Criteria	Recognize the object, with rotation in all three planes, with a margin of +/- 10 degrees
Test Case ID	0

Requirement	The vision-system shall recognize multiple objects from 3D CAD models
Requirement ID	R02.01.03
Parent ID	R02.01
Rationale	Expands the usability of the system
Source	Egil Utheim
Priority	C
Acceptance Criteria	Recognize 3 different objects from a dataset of 3 objects, in the same frame, 9 out of 10 times.
Test Case ID	0

Requirement	The vision-system shall recognize multiple objects with the vision-system
Requirement ID	R02.01.04
Parent ID	R02.01
Rationale	Greatly expands the usability of the system
Source	Egil Utheim
Priority	C
Acceptance Criteria	Recognize 3 different objects from a dataset of 3 objects, in the same frame, 9 out of 10 times.
Test Case ID	0

Requirement	The vision-system shall include neural networking to "learn" the shapes better over time
Requirement ID	R02.01.05
Parent ID	R02.01
Rationale	Improves object recognition over time, improves accuracy of system
Source	Egil Utheim
Priority	D
Acceptance Criteria	Recognition success rate increases from generation to generation (until over-learning is achieved).
Test Case ID	VS008

Requirement	The vision-system shall recognize overlapping objects of the same type
Requirement ID	R02.01.06
Parent ID	R02.01
Rationale	Required to implement "Bin-picking"
Source	Egil Utheim
Priority	C
Acceptance Criteria	The vision system identifies at least 1 object and lists according to accessibility, among several of the same objects
Test Case ID	0

Requirement	The vision-system shall recognize overlapping objects of different types
Requirement ID	R02.01.07
Parent ID	R02.01
Rationale	Enables unsorted bin-picking
Source	Egil Utheim
Priority	E
Acceptance Criteria	The vision system identifies at least 1 object, specifies type, and lists according to accessibility, among several different objects (Objects defined in test specification)
Test Case ID	VS010

A.3.2.2.3 Vision system requirements

Requirement	The programming-modules shall be made as abstract as reasonable
Requirement ID	R02.01.08
Parent ID	R02.01
Rationale	Abstract software modules can easily be applied in other applications
Source	Group
Priority	A
Acceptance Criteria	The vision system works by itself.
Test Case ID	Final delivery yes/no

Requirement	The vision-system shall be able to identify 3mm screw holes in a three-dimensional space (x,y,z) with 0.1mm precision from a distance of 0.4 meters.
Requirement ID	R02.02
Parent ID	R02
Rationale	Required by project owner, recognizing and locating screw-holes enables placement of screws on assembly-parts
Source	Egil Utheim
Priority	A
Acceptance Criteria	The vision system identifies and locates a 3mm hole in a 1mm metal-sheet from a distance of 0.4 meters, with a precision of 1.5mm in all directions
Test Case ID	VS011

Requirement	The vision-system shall give commands to the robot no more than 3 seconds after it needs it
Requirement ID	R02.03
Parent ID	R02
Rationale	Necessary in an effective system, doesn't look good if system stalls for some time between actions
Source	Group
Priority	A
Acceptance Criteria	The time between a request to the vision system is received and a response is sent should not be more than 3 seconds.
Test Case ID	VS013

Requirement	The vision-system shall interface with the existing control-system for Universal Robots
Requirement ID	R02.04
Parent ID	R02
Rationale	System will be designed for use with UR, and has to use existing interfaces
Source	Group
Priority	A
Acceptance Criteria	The vision system completes the task as expected
Test Case ID	VS014

A.3.2.3 Ease of use

Requirement	The automation-system shall be easy to use
Requirement ID	R03
Parent ID	N01.01.03
Rationale	If system is difficult to use, it won't be used
Source	Håvid Engmark, Egil Utheim
Priority	B
Acceptance Criteria	Our client verifies and validates that the system is easy to use.
Test Case ID	EoU001

Requirement	The automation-system shall notify the user if an error occurs
Requirement ID	R03.02
Parent ID	R03
Rationale	System down-time costs a lot, in terms of money and time
Source	Håvid Engmark
Priority	C
Acceptance Criteria	The system gives a notification of some kind if an error occurs
Test Case ID	EoU006

Requirement	The automation-system shall notify the user when there is a shortage of components
Requirement ID	R03.03
Parent ID	R03
Rationale	System down-time costs a lot, in terms of money and time
Source	Håvid Engmark
Priority	C
Acceptance Criteria	The system gives a notification of some kind if there is a shortage of components
Test Case ID	EoU002

Requirement	The automation-system shall be possible to operate by an assembler after 120 minutes of training
Requirement ID	R03.04
Parent ID	R03
Rationale	If system is difficult to use, it won't be used
Source	Group
Priority	B
Acceptance Criteria	An unexperienced person shall be able to use the system after 120 minutes of training
Test Case ID	EoU003

Requirement	The automation-system shall be aware of its surroundings
Requirement ID	R03.05
Parent ID	R03
Rationale	System will be safer to work with/around
Source	Egil Utheim
Priority	E
Acceptance Criteria	The system actively avoids hitting humans
Test Case ID	EoU004

Requirement	The automation-system shall use existing standards if possible and reasonable
Requirement ID	R03.06
Parent ID	N01.01.02
Rationale	System will be easier to install
Source	Group
Priority	C
Acceptance Criteria	Achieved R03.06 children A-requirements
Test Case ID	0

Requirement	The automation-system to be connect to a 230V AC outlet.
Requirement ID	R03.06.01
Parent ID	R03.06
Rationale	Use normal standards where possible
Source	Group
Priority	A
Acceptance Criteria	The systems connects to a 230V AC outlet.
Test Case ID	EoU005

Requirement	Starting the system after initial setup shall take less than 5 commands
Requirement ID	R03.07
Parent ID	R03
Rationale	Fewer commands makes the system easier and quicker to start.
Source	Group
Priority	B
Acceptance Criteria	The system starts after 5 commands
Test Case ID	EoU007

A.3.2.4 General production

Requirement	0,7 man-hours per year shall be removed from the production
Requirement ID	R05
Parent ID	N01
Rationale	Norwegian workers have high wages, reducing man-hours might reduce cost
Source	Håvid Engmark
Priority	A
Acceptance Criteria	TBD man-hours is removed from the production-process
Test Case ID	Not testable in the scope of the project

Requirement	Major components shall be made the same way as before
Requirement ID	R06
Parent ID	N02
Rationale	To meet the need of keeping the "steketakke" traditional
Source	Håvid Engmark
Priority	A
Acceptance Criteria	Product feels as it did before
Test Case ID	Final delivery yes/no

Requirement	The production of components shall be kept at current site
Requirement ID	R07
Parent ID	N03
Rationale	To meet Håvids wish of keeping the production local
Source	Håvid Engmark
Priority	A
Acceptance Criteria	No part of the production line has been outsourced
Test Case ID	Final delivery yes/no

Requirement	Mounting of isolation ring shall be simplified/altered
Requirement ID	R09
Parent ID	N01
Rationale	The current process of mounting the isolation ring is too complex and so to save on material and assembly costs a new way of mounting is needed
Source	Håvid Engmark
Priority	A
Acceptance Criteria	The new mounting process saves time and keeps the same properties.
Test Case ID	GP001

A.4 Requirement specification

Abstract

The document gives an overview of the relations between our requirements and test cases.

Contents

Abstract **44**

Contents **44**

1. Introduction **45**

2. Document explanation **45**

2.1. Requirements and test cases 46

A.4.1 Introduction

This document was created to give an overview over the requirements we have gathered for this project. These requirements will define our project when it comes to goals, testing, and end results. Quite some work must be done before a requirement specification document can be written. In our case we were only given a few basic needs from our project owner (Egil Utheim) and the customer (Håvid Engmark). To define the rest we had to make several visits to Engmark Meteor and have several meetings with external supervisor (Egil Utheim). By using these needs we were able to gather the first batch of requirements for our project.

A.4.2 Document explanation

We have gathered quite the amount of requirements and to separate them we have divided them into A-, B-, C-, D-, E-, and F-lists. The requirements in the A-list are the most important ones. They are the minimum requirements for our project. The B-list has the requirements that are very important, but not a must for the project to be finished. The C- and D-lists will have the requirements that would improve our results, but they will be only considered if there are any resources for them. The E- and F-list requirements are the ones that are least important for us and most likely won't be accomplished because of the shortage of resources.

The requirement specification EXCEL-sheet is meant to show the needs and requirements. The requirements have been given a unique ID to be easy to track. They have been briefly explained in the Rationale and given Parent ID and Source to describe where they originate from (Need/other requirement). We have also given the requirements Acceptance criteria to specify what needs to be achieved to verify the requirement, and Dependency which describes the factors that have direct effect on the realization of the requirement.

In the table below we have gathered the requirements and the representing tests. A requirement can have several tests and so it will be easy to track which tests are already finished and were successful and which failed. For the tests the Test ID will be colored green if it was successful and red if it failed. If the tests are not performed, they will stay in normal color.

The requirement that have achieved their acceptance criteria will be colored green. The ones that have yet to be achieved but are under the process will be colored yellow. The ones that have been dropped will stay in normal color.

A.4.2.1 Requirements and test cases

	Req. ID	Test ID	Comment
A-req.	R08	-	Approved by fulfilling acceptance criteria
	R08.01	AS013	
	R08.01.01	AS001	
	R08.01.02	AS002 AS021	
	R08.01.03	AS003	
	R08.09	-	Approved by fulfilling acceptance criteria
	R08.09.01	AS022	
	R08.09.04A	AS018	
	R02	-	Approved by fulfilling acceptance criteria
	R02.01A	VS001 VS003	
	R02.01A.01	VS001	
	R02.01A.02	VS001	
	R02.01A.03	VS001	
	R02.01A.04	VS001	
	R02.01A.05	VS001	
	R02.01.08	-	Approved by fulfilling acceptance criteria
	R02.02	VS011	
	R02.03	VS013	
	R02.04	VS014	
	R03.06.01	EoU005	
	R05	-	Approved by fulfilling acceptance criteria
	R06	-	Approved by fulfilling acceptance criteria
	R07	-	Approved by fulfilling acceptance criteria
	R09	GP001	
B-req.	R08.02	AS014	
	R08.02.01	AS004	
	R08.02.02	AS005	
	R08.02.03	AS006	
	R08.02.04	AS023	
	R08.03	AS015	
	R08.03.01	AS007	
	R08.03.02	AS008	
	R08.03.03	AS009	
	R08.04	AS016	
	R08.04.01	AS010	
	R08.04.02	AS011	
	R08.04.03	AS012	
	R08.04.04	AS024	
	R08.05	AS029	
	R08.05.01	AS025	
	R08.05.02	AS026	
	R08.05.03	AS027	
	R08.05.04	AS028	
	R08.09.04B	AS019	
	R02.01B	VS002	
	R02.01B.01	VS004	
	R02.01B.02	VS004	
	R02.01B.03	VS004	
	R02.01B.04	VS004	
	R02.01.02	VS005	
	R03	EoU001	
	R03.04	EoU003	
	R03.07	EoU007	
C-req.	R08.09.03	AS017 AS022	
	R08.09.04C	AS020	
	R02.01.03	VS006	
	R02.01.04	VS007	

	R02.01.06 R03.02 R03.03 R03.06	VS009 EoU006 EoU002 -	Approved by fulfilling acceptance criteria
D-req.	R02.01.05	VS008	
E-req.	R02.01.07 R03.05	VS010 EoU004	
F-req.	None	None	None

A.5 Architectural overview

Abstract

This document contains the architectural structure of project Vi'RA. It describes how our system will operate, showcased by a dynamic and a structural dimension of the architecture.

Contents

Abstract **48**

Contents **48**

Figures **48**

A.5.1 Introduction **49**

A.5.2 Dynamic dimension **49**

A.5.3 Structural dimension **51**

Figures

Figure 13: F01 - Dynamic Dimension 49
Figure 14: F01.01 - Dynamic Dimension, Get assembly state 49
Figure 15: F01.02 - Dynamic Dimension, Retrieve ID 50
Figure 16: F01.03 - Dynamic Dimension, Retrieve Part 50
Figure 17: F01.03.02 – Dynamic Dimension, Retrieve Part, Locate Part 50
Figure 18: F01.05 - Dynamic Dimension, Prepare Tool 51
Figure 19: F01.07 - Dynamic Dimension, Handle Exception 51
Figure 20: Structural Dimension 51

A.5.1 Introduction

Creating a good architecture is crucial when using Unified process as a project model. The main architecture is introduced in the elaboration phase of the Unified Process model. Its main purpose is to shape the whole system and serve as a foundation for the remaining development of the project. To cover all aspects of the system, we have described our architecture with two separate views; dynamic and structural dimension. The dynamic dimension describes the functionality of the system and how the different functions interact. Structural dimension describe what elements make up the system and the relationship of these elements.

A.5.2 Dynamic dimension

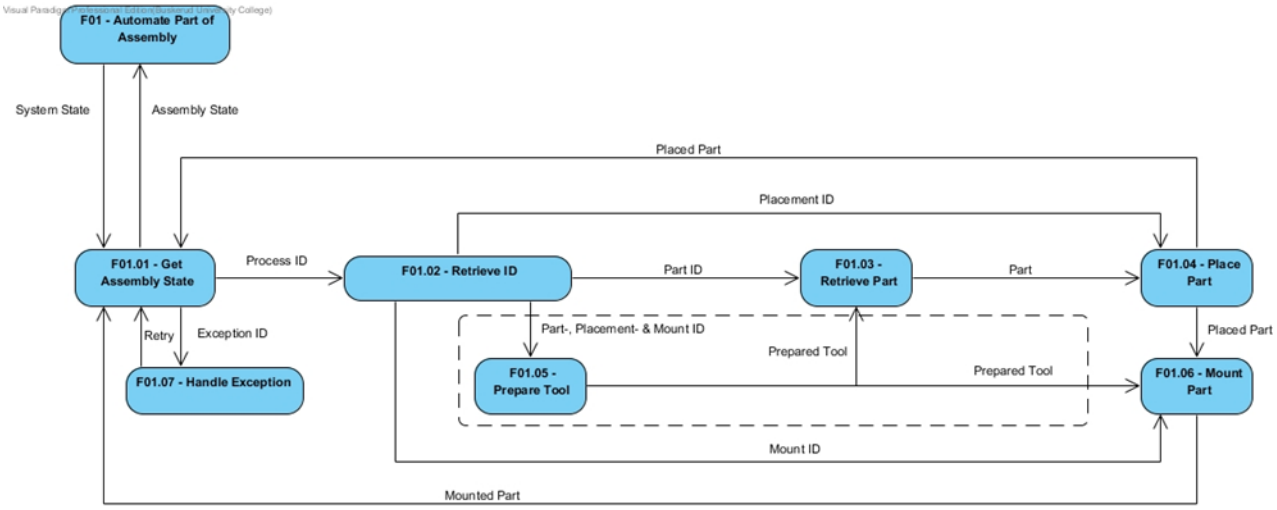


Figure 13: F01 - Dynamic Dimension

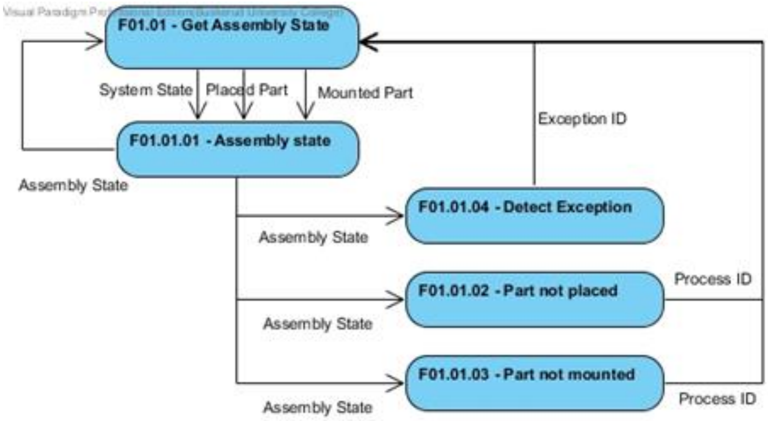

Visual Paradigm Professional Edition(Buskerud University College)
F01.01 - Get Assembly State
System State
Placed Part
Mounted Part
Exception ID
F01.01.01 - Assembly state
Assembly State
F01.01.04 - Detect Exception
Assembly State
Process ID
F01.01.02 - Part not placed
Assembly State
F01.01.03 - Part not mounted
Assembly State
Process ID

Figure 14: F01.01 - Dynamic Dimension, Get assembly state

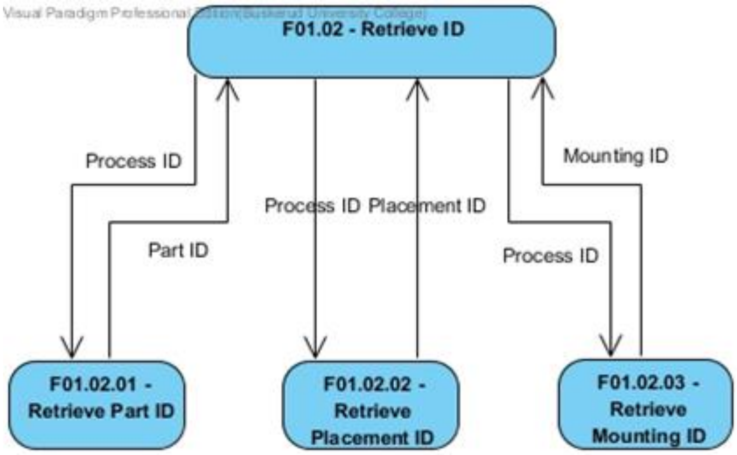

Visual Paradigm Professional Edition(Buskerud University College)
F01.02 - Retrieve ID
Process ID
Part ID
Process ID Placement ID
Mounting ID
Process ID
F01.02.01 - Retrieve Part ID
F01.02.02 - Retrieve Placement ID
F01.02.03 - Retrieve Mounting ID

Figure 15: F01.02 - Dynamic Dimension, Retrieve ID

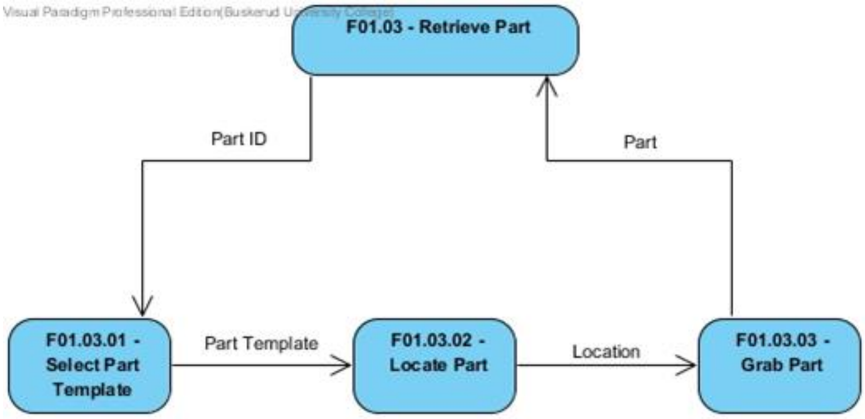


Figure 16: F01.03 - Dynamic Dimension, Retrieve Part

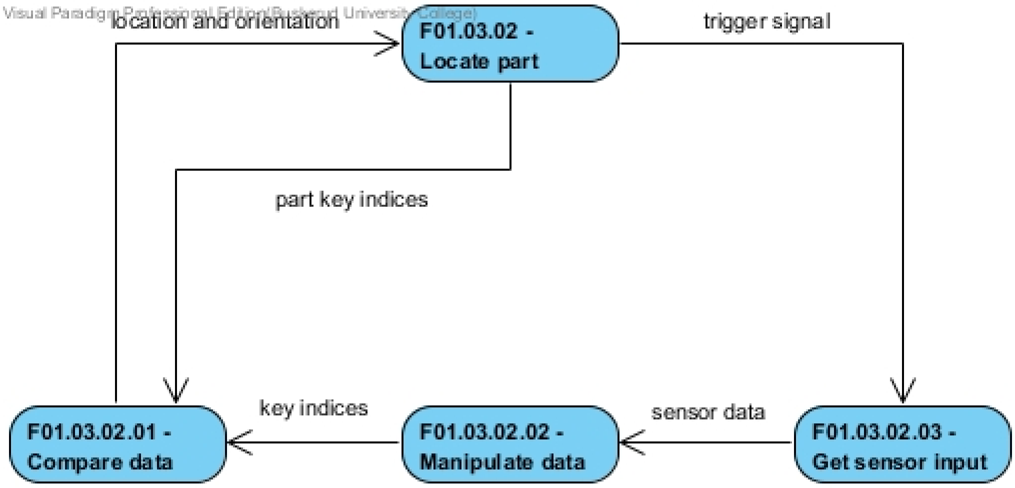

Visual Paradigm Professional Edition(Buskerud University College)
location and orientation
F01.03.02 - Locate part
trigger signal
part key indices
F01.03.02.01 - Compare data
key indices
F01.03.02.02 - Manipulate data
sensor data
F01.03.02.03 - Get sensor input

Figure 17: F01.03.02 – Dynamic Dimension, Retrieve Part, Locate Part

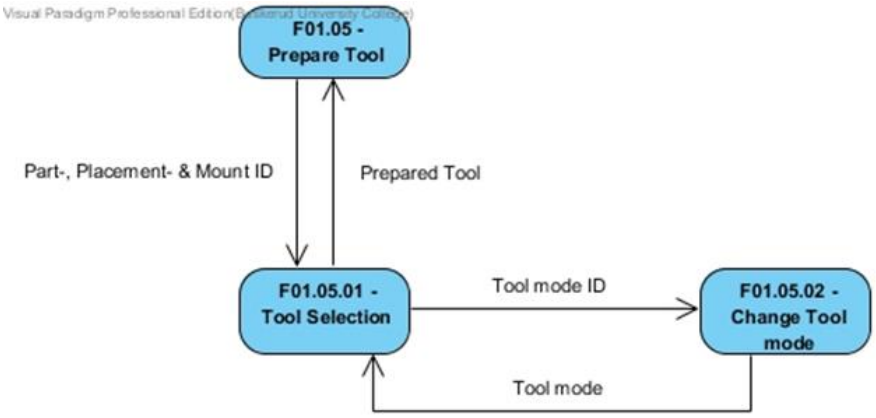

Visual Paradigm Professional Edition(Buskerud University College)
F01.05 - Prepare Tool
Part-, Placement- & Mount ID
Prepared Tool
F01.05.01 - Tool Selection
Tool mode ID
F01.05.02 - Change Tool mode
Tool mode

Figure 18: F01.05 - Dynamic Dimension, Prepare Tool

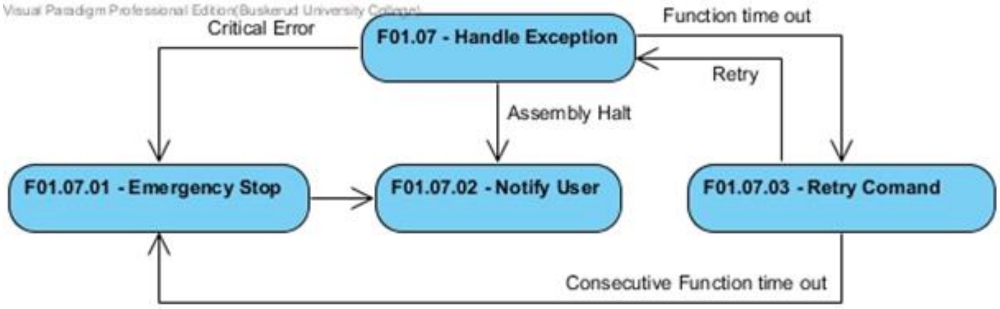

Visual Paradigm Professional Edition(Buskerud University College)
Critical Error
F01.07 - Handle Exception
Function time out
Retry
Assembly Halt
F01.07.01 - Emergency Stop
F01.07.02 - Notify User
F01.07.03 - Retry Comand
Consecutive Function time out

Figure 19: F01.07 - Dynamic Dimension, Handle Exception

A.5.3 Structural dimension

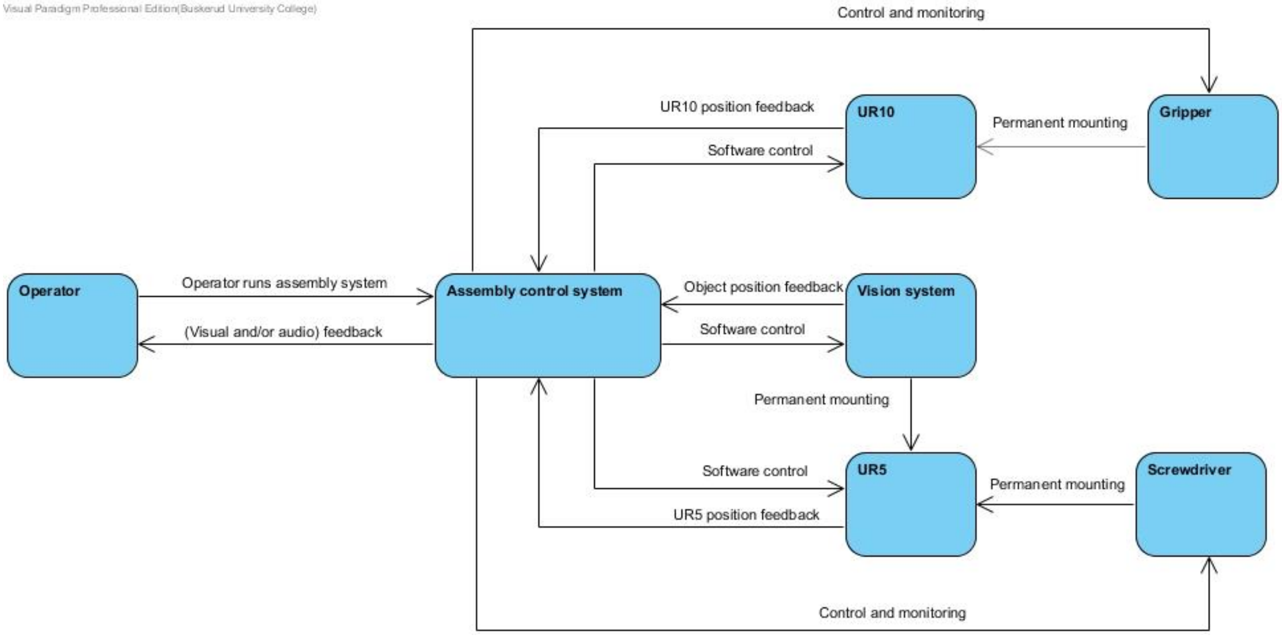


Figure 20: Structural Dimension

A.6 Test document

Abstract

This document contains the test plan for the project. It addresses the purpose of a test plan, what test strategy will be utilized, how to specify a test and how to write a test report. It will give a brief overview of the five different test strategies found in our project handbook and a more detailed walkthrough for the test strategy we have chosen. This document also includes the test case template we will utilize.

Contents

Abstract **52**

Contents **52**

Tables **53**

A.6.1 Introduction **54**

A.6.2 Validation and Verification **54**

A.6.3 Static and Dynamic Testing **54**

A.6.4 Test Strategy **55**

A.6.4.1 Test Strategy overview 55

A.6.4.1.1 Top-Down testing 55

A.6.4.1.2 Bottom-Up testing 55

A.6.4.1.3 Incremental testing 55

A.6.4.1.4 Stress testing 55

A.6.4.1.5 Back-to-Back testing 55

A.6.4.2 Our Test Strategy 55

A.6.4.2.1 Software testing 55

A.6.4.2.1.1 Unit testing (First level) 55

A.6.4.2.1.2 Integration testing (Second level) 56

A.6.4.2.1.3 System testing (Third level) 56

A.6.4.2.1.4 Acceptance testing (Fourth level) 56

A.6.4.2.1.5 Regression testing 56

A.6.4.2.2 Hardware testing 56

A.6.4.2.2.1 Vision system 56

A.6.4.2.2.2 UR-10 robotic arm 56

A.6.5 Test Techniques **57**

A.6.5.1 Black-Box testing 57
A.6.5.2 White-Box testing 57
A.6.5.3 Prototyping 57
A.6.6 Test Case 57
A.6.7 Test Process 59
A.6.7.1 Product development testing 59
A.6.7.1.1 Vision system 59
A.6.7.1.2 Assembly system 60
A.6.7.1.3 Assembly system subsections 60
A.6.7.1.3.1 Mounting bracket 60
A.6.7.1.3.2 Fingers for pneumatic gripper 60
A.6.7.1.3.3 Alternative to pneumatic gripper 60
A.6.7.1.3.4 Make the griddle more suitable for automated assembly 60
A.6.7.1.4 Integration testing 61
A.6.7.1.5 Performance testing 61
A.6.7.2 User testing 61
A.6.8 References 62

Tables

Table 1: Test Case Template 59

A.6.1 Introduction

Testing is an important part of the development process. By running tests, we identify flaws and errors in the system that we can correct to achieve better quality of our system. We can provoke errors by using the system in a wrong way to get a wider overview of the potential of improvements to develop a more robust system.

A.6.2 Validation and Verification

We can divide tests into validation and verification. These are often confused with one another, but they represent two separate activities.

Validation is the assurance that the product or system meets the stakeholders' needs. "Are we building the right product?" A validation technique is to make prototypes. [1].

Verification is the internal evaluation process of the system. It is to check if our system or product complies with a requirement, specification, or imposed condition. "Are we building the product in the correct way?" [1].

A.6.3 Static and Dynamic Testing

To validate and verify a system we have to perform both static and dynamic tests.

Static tests are performed when the system is stationary. Static test techniques involve checking if the system complies with the requirement specification and other documentation. For the software part, static testing will be to read the code to look for immediate errors. Since we only have one software member in our group, he will have to read and correct his own code. He will have to check if the code corresponds with the determined code standard, readability and syntaxes. Other static tests for the group will include checking that cables and tools are connected correctly. That the dimensions of our parts correspond to our calculated models or that no parts are missing from the system. We can run static tests at any stage of the project because it does not require a complete system or sub-system.

Dynamic tests however are performed by simulation, testing, and use of the product or system. In software testing, we will be compiling and running the entire system. This will take place towards the end of the development of the software. The Universal Robot has a simulation program we will use to see how the robotic arm will behave before we start running the physical tests. If we have to develop our own end effector for the project, we are able to run simulations in SolidWorks.

Dynamic testing is performed on the actual product with two types of test data:

- Statistic testing. This is to document the reliability of the product. We are using the system as intended. The data that goes in to the system has to be the same as when the system will be operated. [1].
- Troubleshooting. This type of testing is to provoke errors in the system. These tests are designed to make the system fail. If the system fails, the test is a success. If the system does not fail, the test fails. [1].

A.6.4 Test Strategy

A.6.4.1 Test Strategy overview

A test strategy is an overall method to perform the testing. In our project handbook, from the school, there are listed five different test strategies.

A.6.4.1.1 Top-Down testing

This is a test method where we test the system as a whole. We use dummy modules to fill in for the missing parts in the project. As we proceed along the project timeline, we will substitute the dummy modules with our own developed parts. [1].

A.6.4.1.2 Bottom-Up testing

This test strategy starts with testing of the smallest parts of the system first and then merges them together in larger subsystems. When all the subsystems are finished, they merge into a complete system. [1].

A.6.4.1.3 Incremental testing

The incremental test strategy is a modified version of both the top-down and the bottom-up strategy. In this method, we run tests the most critical units first. We can use both top-down and bottom-up interchangeably. [1].

A.6.4.1.4 Stress testing

Run the system way beyond the needs, to observe how it handles the load. This may reveal problems that would not occur through regular use. [1].

A.6.4.1.5 Back-to-Back testing

Back-to-back testing is performed by running similar modules of a system or systems in parallel with the same inputs and comparing outputs. [1].

A.6.4.2 Our Test Strategy

The overall test strategy that suits our purpose the best is the incremental strategy. This is due to the different disciplines in our project where we have software development testing, testing of the robotic arm in a fixed environment with or without different tools, and testing of usability. For software, we will utilize a bottom-up strategy and for the robotic arm testing, we will utilize a top-down strategy.

A.6.4.2.1 Software testing

The basic strategy we will use to test the software is a bottom-up model. This means we will get a hierarchy of testing, each level more complex than the one above it.

A.6.4.2.1.1 *Unit testing (First level)*

Here we test the smallest units of the program. This involves testing the functions, parameters and classes. These tests will typically be written by the person writing the code to test correct behaviour, when receiving unexpected parameters. Our software will utilize many libraries and testing of their functions will become crucial for success. [2], [3].

A.6.4.2.1.2 *Integration testing (Second level)*

Integration testing is performed when all sub units relevant for a larger functionality is in place. Then they are merged together to form a bigger unit. Although a system put together of multiple functional components should yield a valid output that is not always the case. Testing of how these components work together therefore becomes essential. [2], [4].

At this point, we will also combine the software with the hardware to run tests.

A.6.4.2.1.3 *System testing (Third level)*

This is a test of a system to check whether it complies with the requirements or not. The system tested may or may not be the complete system, but it should to be able fulfil a requirement. Typical tests in this level include non-functional tests like performance tests, stress tests and usability tests. [2], [5].

A.6.4.2.1.4 *Acceptance testing (Fourth level)*

This is testing whether or not the system fulfils the requirements made directly by primary stakeholders. These are usually not to check if the system functions or give the correct output, but rather to see if the output you get correspond with what the stakeholder needs. [2], [6].

A.6.4.2.1.5 *Regression testing*

When we add new code or alter existing code (fix errors or simply rewrite), we have to do regression testing. Regression testing involves testing all use cases made for the system. This is to exclude the possibilities of errors after e.g. addition of new code: The code might work by itself, but may induce errors other places. [2], [7].

A.6.4.2.2 **Hardware testing**

The hardware testing is divided into two areas; the vision system testing and the UR-10 robotic arm.

A.6.4.2.2.1 *Vision system*

The strategy for the vision tests will combine the back-to-back test strategy and stress testing. Since there are several camera solutions that we can choose from, we have to find the one that satisfies our needs the most. To do this we have to run similar tests on each system and compare the outputs. If the camera barely meet our needs for a perfect condition, it may not be suitable at the production site. It is therefore important to have a good margin from the limitation to the application.

A.6.4.2.2.2 *UR-10 robotic arm*

For the testing of the robotic arm, as mention earlier, we will follow a top-down test strategy. We are already familiar with the technical limitations of the product from the supplier so we will only have to test how it will interact with the vision system and the different tools.

In the first step of the testing, we will use the simulator for the UR-10. After the simulations, we will move over to the robotic arm itself. By using dummy modules in the form of fake data from imaginary tools and made up coordinates from what is supposed to be the vision system, we can observe how the robotic arm behaves in real life. Then, we will gradually replace the dummy modules with the actual tools and vision system until we have the complete system. This includes testing both the interaction between the vision system and the robotic arm, and the griddle and the robotic arm with tools.

A.6.5 Test Techniques

There are mainly two types of testing: functionality testing (black-box) and structural testing (white-box).

A.6.5.1 Black-Box testing

In a black-box test, we are only considering the input and the output of the test. For these tests, the tester does not need any knowledge about what happens inside of the system. That means that members of the group can perform tests of this kind outside their respective fields because they only have to know what the system is supposed to do and not how the system does it. Black-box tests are based on the requirement specification and checks whether the given input produces the wanted output or not. [1].

A.6.5.2 White-Box testing

The structural testing requires knowledge about the internals of the system that is to be tested. In these tests, we peek inside the system and look at the structural design of the system. A white-box test can give vital information about how many tests that will be necessary to get a complete run through of the system. [1].

A.6.5.3 Prototyping

A prototype is an early model of a system or product. The prototype is a temporary solution made for testing and display. It is possible to make more than one prototype and progressively upgrade the prototypes through the project until we have a finished product. The prototypes are also used in the validation process where the stakeholders validate whether their needs are met or not. It is easier for the stakeholders to evaluate a working product or system according to their needs rather than just evaluating from a written report. [1], [8].

A.6.6 Test Case

To perform tests, we have to define the test case. The test setup is different for most tests, but there key elements that recur for every test. Therefore, we can use the same test case layout to store information about each test. Most of the test cases will get a rough outline when we formulate requirements from use cases, but some tests will be determined later when we have done some more research and completed other tests. When we know more exactly how we will perform a test, we write a more detailed specification. To get a clear understanding of the tests and make them easy to track, we have developed a test case template that includes the necessary information to meet our requirements for documentation. The test case includes:

- A unique test case ID.
    - XX denotes what field the test pertain.
        - AS – Assembly System
        - VS – Vision System
        - GP – General Production
        - EoU – Ease of Use

    - xxx represents the number of the test.
    - Example, Assembly System test 13: AS013.
- Activity ID for traceability in the backlog.
- Test number. If we have to carry out the same test more than once, we have control for how many times we have conducted the test.
- ID of the requirement or requirements from that the test originates.
- Priority of the test.
- Test type. We can run different types of tests for the same requirement. It is therefore important to specify which one. Static test or dynamic test? Black box or white box? For dynamic tests, we also have to specify statistic or troubleshooting.
- Status of the test. Did the test succeed or did it fail? If a test is designed to find defects, it is a success when we identify some and not a failure.
- Information about creation date, due date, and date of completion.
- Test specification. A detailed description of
    - How we are supposed to perform the test.
    - What assumptions we have for the test.
    - Predefined conditions for the test.
    - Desired outputs.
- Acceptance criteria. What objectives that must be met to classify the test as a success.
- A comment; a short report about the test. If something went wrong, what went wrong and why? Detailed documentation is essential.
- The name of the tester or testers.

Test Case ID	XXxxx	"Name of the test"	
Activity ID	xxx	Test number:	xxx
Requirement ID	xxx		
Priority	High, medium, low		
Test type:	(Black Box/White Box)(Dynamic(Statistic/troubleshooting)/Static)		
Status:	Success/fail		
Created date:	yyyy.mm.dd		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	How are you supposed to perform this test? Be specific. What assumptions are made for the test? What conditions are predefined for performing this test?		

Acceptance criteria:	What are the criteria for a successful test?
Comment:	What was the outcome of the test?
Tested by:	“name”

Table 1: Test Case Template.

All the test cases are gathered in a separate document [9]. We have chosen to define test cases in the test case document only for the tests derived directly from the requirement specification. Other tests that are related to the development of a product or research of technology are described in the document regarding this product or technology.

A.6.7 Test Process

Our test process will start with the product development testing which aims towards an FAT (Factory Acceptance Test). This includes running tests for both the vision system and the assembly system that satisfy the acceptance criteria at our office. It is not enough to test the systems separately; we have to run integration tests to see if they work as one complete system.

When the FAT is complete and successful, we will move over to the SAT (Site Acceptance Test) at Engmark Meteor. We have to make sure that the system works as intended at the customer’s location as well. Although a system works perfectly at our site, it may not be the case for where it is supposed to be. The SAT will also include user testing. The user testing process is very small compared to the test process for the product development.

Some minor subsections to the assembly system’s test process include developing a mounting bracket for different end effectors, testing the different end effectors, develop fingers for our pneumatic gripper and make changes to the griddle to make it more suitable for automated assembly.

A.6.7.1 Product development testing

Testing of the assembly system will run in parallel to the vision system development. All the testing up to where the vision system integrates with the robotic arm are independent processes.

A.6.7.1.1 Vision system

We begin our development process by doing pre-tests of the different types of camera solutions we have available. A vital point in our vision system for the assembly process is to locate the holes for the screws. These are relatively small so we have to test the camera sensors to see if the resolution is high enough to localize them at all. A factor that also weighs in is the usability of the camera.

After we have determined the camera solution, we can move on with the vision system development. The development of vision systems has generally not come so far. For us, this will result in a lot of trial and error.

The vision system has to be able to give not only position, but also angle of components. For the screw holes, this is not an important feature because they will be either straight down or directly in from the side. For picking and placing components, it is important that they are oriented the correct way.

A.6.7.1.2 Assembly system

In the first step of the assembly system testing, we will explore the possibilities and limitations of the UR-10 simulator. Then, we will run simulations to see how the robotic arm behaves and moves when given specific inputs.

The next step is moving over to the physical robot arm. We will construct our own platform to mount the UR-10. A setup to keep the griddle at the same place every time and keep it in place will be built on the platform also. This gives the advantage of moving the robotic arm if necessary and the operation area for the UR-10 stays at the same place. It will also eliminate the need for calibration for every time we move something. We will then find a smart solution to send and receive coordinates between the computer and the UR-10 so the integration with the vision system will happen smoothly.

Until we have the actual tools the robot is going to use, we will use temporary mock-ups to act as the real ones.

A.6.7.1.3 Assembly system subsections

A.6.7.1.3.1 *Mounting bracket*

On the robot arm, we are going to attach different tools. Initially, we are testing to see how far we can get in the assembly process with a pneumatic gripper and a motorized screwdriver. Since these products are not made especially for the universal robots, we will have to develop our own mounting bracket. The mounting bracket has to be constructed in a way that will give full operation for all the end effectors connected to it. It will require prototyping to find the best solution.

A.6.7.1.3.2 *Fingers for pneumatic gripper*

We will also have to customize fingers for the gripper that will fit our purpose. This will involve some prototyping. Make simple versions in the beginning to see which solutions may work and then work our way from here.

A.6.7.1.3.3 *Alternative to pneumatic gripper*

If the pneumatic gripper is inadequate for picking and placing components, we have vacuum cups as an alternative. These come in many different sizes, but they will not necessary be able to replace the gripper entirely. A combination of a gripper and vacuum cups may be a solution. The outcome of the gripper testing will decide if we will have to consider the vacuum cups or not.

A.6.7.1.3.4 *Make the griddle more suitable for automated assembly*

To make the griddle more suitable for automated assembly, our client have given clearance to alter the inside of the griddle. To see if it is possible to modify anything without affecting the properties, we are going to test different solutions and measure heat distribution and see if a short-circuiting occur.

We are determined to get rid of the starlock to cut down on the number of parts. To do this, we will design a cutout for the reflector that will act the same way as a starlock. The development of this solution will be based on prototyping were we will design and redesign the cutout until it works as intended.

A.6.7.1.4 Integration testing

When all the systems and components have passed all individual test, we have to test if they are compatible. First, the system perform single sequences of the assembly process to verify that the system does what it is supposed to do. The next to do is to combine all the sequences into one complete process.

A.6.7.1.5 Performance testing

After the system manages to complete the task correctly, quality, speed and repeatability has to be tested according to the requirement specification. The quality test is successful if the griddle is as good as if a trained assembler had put it together. The speed of the robot should be as fast possible, but if this affect the quality of the assembly, reduction of the speed is necessary to achieve an acceptable level of quality. This applies for repeatability as well. The speed must be reduced if it causes the assembly system to make mistakes. If the system has to be very slow to function properly, we have to identify the weak points so they can be upgraded.

A.6.7.2 User testing

When we are close to the finished system, we have to test the product together with the ones who are going to be using it. This part of the test process is referred to as alpha testing. Since our version of the vision assisted robotic assembly system is meant for Engmark only, it is not necessary to perform beta testing. Beta testing happens after alpha testing and is only necessary when the system is going to the open market. Beta testing involves releasing the system to a group of customers or users to test the system in daily use. Since we only have one client, the development process will be between the client and us.

The extent of the user testing depends on how far we manage to develop our system. Ideally, our system will be able to control an entire station by itself, but we have to keep in mind that we may not get that far. If we can only complete the “fasten the screw”-action, we have to run tests to see how this and the user interact. The user will give us feedback and we will modify the product desired by the user.

If we do reach the ideal goal, the test will consist of to see if the user manages to choose the right program and press start.

A.6.8 References

1. Torbjørn Strøm & Olaf Hallan Graven, (2015). *Prosjekthåndbok ver 2015.* Kongsberg: Institutt for Ingeniørfag, HBV avdeling Kongsberg.

2. Wikipedia. *Software testing*. 2015. [06.02.2015] http://en.wikipedia.org/wiki/Software_testing

3. Wikipedia. *Unit testing*. 2015. [06.02.2015] http://en.wikipedia.org/wiki/Unit_testing

4. Wikipedia. *Integration testing*. [06.02.2015] http://en.wikipedia.org/wiki/Integration_testing

5. Wikipedia. *System testing*. [06.02.2015] http://en.wikipedia.org/wiki/System_testing

6. Wikipedia. *Acceptance testing*. [06.02.2015] http://en.wikipedia.org/wiki/Acceptance_testing

7. Wikipedia. *Regression testing*. [06.02.2015] http://en.wikipedia.org/wiki/Regression_testing

8. Wikipedia. *Prototype*. [06.02.2015] http://en.wikipedia.org/wiki/Prototype

9. *Test Document Test Cases*

A.7 Test cases

Abstract

This document contains all the test cases to test the requirements for the project. That include completed tests, both successful and failed, pending tests and even the unspecified tests connected to the requirements.

Contents

Abstract 63

Contents 63

A.7.1 Introduction 66

A.7.2 Test Case template 66

A.7.3 Assembly Test Cases 67

A.7.3.1 AS001 Pick up the screw (completed, success)........ 67

A.7.3.2 AS002 Place the screw 68

A.7.3.2.1 Test number 1: Placing the screw vertically (completed, success)........ 68

A.7.3.2.2 Test number 2: Placing the screw horizontally 69

A.7.3.3 AS003 Fasten the screw (completed, success) 70

A.7.3.4 AS004 Pick up the E-CC (completed, success)........ 71

A.7.3.5 AS005 Place the E-CC (completed, success)........ 72

A.7.3.6 AS006 Fasten the E-CC 73

A.7.3.7 AS023 Hold the E-CC while fastening it........ 74

A.7.3.8 AS007 Pick up the E-PI 75

A.7.3.9 AS008 Place the E-PI 76

A.7.3.10 AS009 Fasten the E-PI 77

A.7.3.11 AS010 Pick up the E-REG (completed, success) 78

A.7.3.12 AS011 Place the E-REG (completed, success) 79

A.7.3.13 AS012 Fasten the E-REG........ 80

A.7.3.14 AS024 Hold the E-REG while fastening it 81

A.7.3.15 AS025 Pick up the M-BP........ 82

A.7.3.16 AS026 Place the M-BP........ 83

A.7.3.17 AS027 Fasten the M-BP........ 84

A.7.3.18 AS028 Hold the M-BP while fastening it 85

A.7.3.19 AS013 Replace human screw process........ 86

A.7.3.19.1 Test number 1: Fastening the bottom plate (completed, success) 86
A.7.3.20 AS014 Mount E-CC without human interaction.. 87
A.7.3.21 AS015 Mount E-PI without human interaction.. 88
A.7.3.22 AS016 Mount E-REG without human interaction 89
A.7.3.23 AS029 Mount M-BP without human interaction 90
A.7.3.24 AS017 Resume assembly process after an interruption(test not defined)......................... 91
A.7.3.25 AS018 (test not defined) 92
A.7.3.26 AS019 (test not defined) 93
A.7.3.27 AS020 (test not defined) 94
A.7.3.28 AS021 Screw placement precision requirement (completed, success)............................. 95
A.7.3.29 AS022 Perform assembly-action TBD times without error 96
A.7.4 Vision Test Cases 97
A.7.4.1 VS001 Recognize screw hole with given orientation 97
A.7.4.2 VS002 Optimal camera angle 98
A.7.4.3 VS003 Optimal camera range 99
A.7.4.4 VS004 Recognize object in any orientation 100
A.7.4.5 VS005 Recognize object orientation 101
A.7.4.6 VS006 Recognize object from 3D CAD 102
A.7.4.7 VS007 Recognize object through vision scanning.. 103
A.7.4.8 VS008 Neural networking 104
A.7.4.9 VS009 Recognize overlapping objects of the same type 105
A.7.4.10 VS010 Recognize overlapping objects of different types 106
A.7.4.11 VS011 Recognize holes... 107
A.7.4.12 VS013 Vision output response test 108
A.7.4.13 VS014 Interface with existing control-system 109
A.7.5 Ease of Use Test Cases 110
A.7.5.1 EoU001 Verification and validation of ease of use 110
A.7.5.2 EoU002 Component shortage user notification 111
A.7.5.3 EoU003 Training for operation of the system (test not defined) 112
A.7.5.4 EoU004 Awareness of system surroundings (test not defined) 113
A.7.5.5 EoU005 230V AC outlet.. 114
A.7.5.6 EoU System error user notification (test not defined)... 115
A.7.5.7 EoU007 Number of commands to start the system 116
A.7.6 General Production Test Cases 117

A.7.6.1 GP001 SL/M-REF properties.. 117

A.7.6.1.1 Test number 1: First batch, Starlock_4.75_4.35_120d (completed, fail) 117

A.7.6.1.2 Test number 2: First batch, Starlock_4.95_3.8_40d (completed, fail) 118

A.7.6.1.3 Test number 3: First batch, Starlock_5.0_3.27_15d (completed, fail) 119

A.7.6.1.4 Test number 4: First batch, Starlock_5.5_3.95_35d (completed, fail) 120

A.7.1 Introduction

The test cases are made from the requirement specification. Some tests were made when the requirements were defined and some tests were only sketched to get an idea of how we were going to perform the tests and given more details or changed completely later on. The rest of the requirements needed more research before designing the tests.

Tests that are completed has the status field filled with either fail or success. The background for fail and success are marked with red and green respectively. The colour stands out and gives a quick visual feedback on the status of the test. The title of the test case is added “complete,” and either “fail” or “success” in parentheses. By doing this we can easily see in the table of contents what tests are completed.

A.7.2 Test Case template

Test Case ID	XXxxx	“Name of the test”	
Activity ID	xxx	Test number:	xxx
Requirement ID	xxx		
Priority	High, medium, low		
Test type:	(Black Box/White Box)(Dynamic(Statistic/troubleshooting)/Static)		
Status:	Success, fail		
Created date:	yyyy.mm.dd		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	How are you supposed to perform this test? Be specific. What assumptions are made for the test? What conditions are predefined for performing this test?		
Acceptance criteria:	What is the criteria for a successful test?		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3 Assembly Test Cases

A.7.3.1 AS001 Pick up the screw (completed, success)

Test Case ID	**AS001**	Pick up the screw		
Activity ID	105		Test number:	1
Requirement ID	R08.01.01			
Priority	High			
Test type:	Black Box			
Status:	Success			
Created date:	2015.03.19			
Due date:	2015.04.16			
Completed date:	2015.04.13			
Test specification:	In the real assembly process, a screw feeder will present screws for the robot. This will give the advantage of having a consistent pickup point. To test if the robot is able to pick up a screw with the attached screwdriver, we will place a screw in a fixed position in a way that simulates the screw feeder. We will then calibrate the pickup point and then run a “get a screw”-program 20 times. The outcome of this test will tell us if there is a need of a sensor that verifies that the screw is on the screw bit.			
Acceptance criteria:	The system is able to pick up a screw from a specified position and orientation 20 times without failure.			
Comment:	After calibrating the pickup point in x, y and z, the system managed to pick up the screws with no problem. The screw stayed on every time. We used 3,9x13mm AW self-drilling screws, AW bit and a magnetic bit holder.			
Tested by:	Thor Eirik Johnsrud, Ivar Roskifte Leikarnes			

A.7.3.2 AS002 Place the screw

This test is split in two because there are holes located both vertically and horizontally. To see if there are measures that must be done in relation between placing the screw horizontally and vertically.

A.7.3.2.1 Test number 1: Placing the screw vertically (completed, success)

Test Case ID	**AS002**	Place the screw	
Activity ID	117	Test number:	1
Requirement ID	R08.01.02		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.03.22		
Due date:			
Completed date:	2015.04.14		
Test specification:	We pretend that the system has picked up a screw already and is now going to place it in a specific hole. We and not the vision system give the coordinates for the screw hole to the robot. The specific hole we use for this test is a hole in a plank. The plank is lying flat on our test bench with the hole facing up. The robot comes in from the top and places the screw downwards into the hole.		
Acceptance criteria:	The system is able to place the screw in the hole with a 1,5 mm precision		
Comment:	Since the robot is accurate to 0,1mm, the precision is depended on the coordinates we give to the robot. We used the handheld pneumatic push-to-start screwdriver[1] borrowed from Engmark Meteor to perform this test. This has some looseness to the bit, but this did not affect the test. We used 3,9x13mm AW self-drilling screws, AW bit and a magnetic bit holder.		
Tested by:	Thor Eirik Johnsrud, Eirik Fosshaug Skjold		

[1] More information about the screws and screwdriver in Technology Document Screwdriver

A.7.3.2.2 Test number 2: Placing the screw horizontally

Test Case ID	**AS002**	Place the screw	
Activity ID	xxx	Test number:	2
Requirement ID	R08.01.02		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.04.24		
Due date:			
Completed date:			
Test specification:	We pretend that the system has picked up a screw already and is now going to place it in a specific hole. We and not the vision system give the coordinates for the screw hole to the robot. The specific hole we use for this test is a hole in a plank. The plank is vertical on our test bench with the hole facing to the side. The robot comes in from the side and inserts the screw straight into the hole.		
Acceptance criteria:	The system is able to place the screw in the hole with a 1,5 mm precision		
Comment:	What was the outcome of the test?		
Tested by:	Thor Eirik Johnsrud, Eirik Fosshaug Skjold		

A.7.3.3 AS003 Fasten the screw (completed, success)

Test Case ID	**AS003**	Fasten the screw	
Activity ID	118	Test number:	1
Requirement ID	R08.01.03		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.03.20		
Due date:			
Completed date:	2015.04.22		
Test specification:	The robot will have the screwdriver attached and a screw attached to the screw bit. We use self-drilling screws. We fastened the bottom plate (M-BP) to the casing (M-CA) in this test. The tip of the screw will be centered above the screw hole in the bottom plate, but there is no hole in the casing placed underneath. This is how these components will be fastened in a real automated assembly process. The screw will enter the screw hole and the robot will run a “fasten the screw”-program until the screw is completely fastened and then lift the screwdriver away.		
Acceptance criteria:	The system is able to fasten the screw completely in the hole without error.		
Comment:	We used the handheld pneumatic push-to-start screwdriver[2] borrowed from Engmark Meteor to perform this test. We used 3,9x13mm AW self-drilling screws, AW bit and a magnetic bit holder. There was no problem fastening the screw, but the self-drilling screws take some time getting through the material of the casing. 5-6 seconds average. The screws we are using need about 1700rpm, but the screwdriver can only do 500. The test was successful, but the time will be significantly lowered with a faster screwdriver.		
Tested by:	Thor Eirik Johnsrud, Ivar Roskifte Leikarnes, Kristian Dønheim Kvam		

[2] More information about the screws and screwdriver in Technology Document Screwdriver

A.7.3.4 AS004 Pick up the E-CC (completed, success)

Test Case ID	**AS004**	Pick up the E-CC		
Activity ID	134		Test number:	1
Requirement ID	R08.02.01			
Priority	High			
Test type:	Black Box			
Status:	Success			
Created date:	2015.02.18			
Due date:	2015.05.14			
Completed date:	2015.05.14			
Test specification:	E-CC – The cable connector(sukkerbit) The robot is supposed to pick up an E-CC from a specific position. We and not the vision system give the coordinates and orientation for the E-CC. The way the system has to grip the E-CC is depended on our gripper solution.			
Acceptance criteria:	The system is able to pick up the E-CC.			
Comment:	We used 3D printed gripper fingers and a Festo DHPS-A 35mm pneumatic parallel gripper to pick up an E-CC. The test was successful.			
Tested by:	Thor Eirik Johnsrud, Deividas Svaikauskas			

A.7.3.5 AS005 Place the E-CC (completed, success)

Test Case ID	**AS005**	Place the E-CC	
Activity ID	135	Test number:	1
Requirement ID	R08.02.02		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.02.18		
Due date:	2015.05.14		
Completed date:	2015.05.14		
Test specification:	E-CC – The cable connector(sukkerbit) M-PB – Power bracket The system is supposed to place the E-CC at a given position in a specific orientation. We and not the vision system give coordinates and orientation for the E-CC. The robot is already holding the E-CC and is going to place it how it is supposed to be mounted on the M-PB. The screw hole in the power bracket and the screw hole in the cable connector should align, but it is not necessary since we are using self-drilling screws. As long as the flat surface of the power bracket is beneath the E-CC, the system will be able to fasten it.		
Acceptance criteria:	The system is able to place the E-CC with a TBD mm precision		
Comment:	We used 3D printed gripper fingers and a Festo DHPS-A 35mm pneumatic parallel gripper to place an E-CC. The test was successful.		
Tested by:	Thor Eirik Johnsrud, Deividas Svaikauskas		

A.7.3.6 AS006 Fasten the E-CC

Test Case ID	**AS006**	Fasten the E-CC	
Activity ID	xxx	Test number:	0
Requirement ID	R08.02.03		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.18		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-CC – The cable connector(sukkerbit) The E-CC is already in place and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a “fasten the screw”-program until the E-CC firmly fastened.		
Acceptance criteria:	The system is able to fasten the E-CC with given screws		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.7 AS023 Hold the E-CC while fastening it

Test Case ID	**AS023**	Hold the E-CC while fastening it	
Activity ID	xxx	Test number:	0
Requirement ID	R08.02.04		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.05.01		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-CC – The cable connector(sukkerbit) The E-CC is held in place by the system, and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a “fasten the screw”-program until the E-CC firmly fastened.		
Acceptance criteria:	The system is able to hold the E-CC while fastening it with given screws		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.8 AS007 Pick up the E-PI

Test Case ID	**AS007**	Pick up the E-PI	
Activity ID	xxx	Test number:	0
Requirement ID	R08.03.01		
Priority	Low		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.18		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-PI – The power inlet The robot is supposed to pick up an E-PI. We and not the vision system give the coordinates and orientation for the E-PI. The way the system has to grip the E-PI is depended on our gripper solution.		
Acceptance criteria:	The system is able to pick up the E-PI		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.9 AS008 Place the E-PI

Test Case ID	**AS008**	Place the E-PI	
Activity ID	xxx	Test number:	0
Requirement ID	R08.03.02		
Priority	Low		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.18		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-PI – The power inlet M-CA – The casing The E-PI is to be inserted in the hole it is supposed to be inserted in. We and not the vision system give coordinates and orientation for the E-PI. The robot is already holding the E-PI and is going to place it how it is supposed to be mounted in the casing. It is important that the screw hole in the E-PI and the screw hole in the casing align.		
Acceptance criteria:	The system is able to place the E-PI with a TBD mm precision		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.10 AS009 Fasten the E-PI

Test Case ID	**AS009**	Fasten the E-PI	
Activity ID	xxx	Test number:	0
Requirement ID	R08.03.03		
Priority	Low		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-PI – The power inlet The E-PI is already in place and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a “fasten the screw”-program until the E-PI firmly fastened.		
Acceptance criteria:	The system is able to fasten the E-PI with given screws		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.11 AS010 Pick up the E-REG (completed, success)

Test Case ID	**AS010**	Pick up the E-REG	
Activity ID	136	Test number:	1
Requirement ID	R08.04.01		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.02.19		
Due date:	2015.05.14		
Completed date:	2015.05.14		
Test specification:	E-REG – The regulator The robot is supposed to pick up an E-REG. We and not the vision system give the coordinates and orientation for the E-REG. The way the system has to grip the E-REG is depended on our gripper solution.		
Acceptance criteria:	The system is able to pick up the E-REG		
Comment:	We used 3D printed gripper fingers and a Festo DHPS-A 35mm pneumatic parallel gripper to pick up an E-REG. The test was successful.		
Tested by:	Thor Eirik Johnsrud, Deividas Svaikauskas		

A.7.3.12 AS011 Place the E-REG (completed, success)

Test Case ID	**AS011**	Place the E-REG	
Activity ID	137	Test number:	0
Requirement ID	R08.04.02		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.02.19		
Due date:	2015.05.14		
Completed date:	2015.05.14		
Test specification:	E-REG – The regulator M-CA – The casing The robot is supposed to place the E-REG. We and not the vision system give the coordinates and orientation for the E-REG. The robot is already holding the E-REG and is going to place it how it is supposed to be mounted in the casing. It is important that the screw hole in the M-CA and the screw hole in the E-REG align.		
Acceptance criteria:	The system is able to place the E-REG with a TBD mm precision		
Comment:	We used 3D printed gripper fingers and a Festo DHPS-A 35mm pneumatic parallel gripper to place an E-REG. The test was successful.		
Tested by:	Thor Eirik Johnsrud, Deividas Svaikauskas		

A.7.3.13 AS012 Fasten the E-REG

Test Case ID	**AS012**	Fasten the E-REG	
Activity ID	xxx	Test number:	0
Requirement ID	R08.04.03		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	E-REG – The regulator The E-REG is already in place and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a “fasten the screw”-program until the E-REG firmly fastened.		
Acceptance criteria:	The system is able to fasten the E-REG with given screws		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.14 AS024 Hold the E-REG while fastening it

Test Case ID	**AS024**	Hold the E-REG while fastening it		
Activity ID	xxx		Test number:	0
Requirement ID	R08.04.04			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.05.01			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	E-REG – The regulator The E-REG is held in place by the system, and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a “fasten the screw”-program until the E-REG firmly fastened.			
Acceptance criteria:	The system is able to hold the E-REG while fastening it with given screws			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.15 AS025 Pick up the M-BP

Test Case ID	**AS025**	Pick up the M-BP	
Activity ID	xxx	Test number:	0
Requirement ID	R08.05.01		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.05.01		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	M-BP – The Bottom plate The robot is supposed to pick up an M-BP. We and not the vision system give the coordinates and orientation for the M-BP. The way the system has to grip the M-BP is depended on our gripper solution.		
Acceptance criteria:	The system is able to pick up the M-BP		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.16 AS026 Place the M-BP

Test Case ID	**AS026**	Place the M-BP	
Activity ID	xxx	Test number:	0
Requirement ID	R08.05.02		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.05.01		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	M-BP – The Bottom plate M-CA – The casing The robot is supposed to place the M-BP. We and not the vision system give the coordinates and orientation for the M-BP. The robot is already holding the M-BP and is going to place it how it is supposed to be mounted on the casing. It is not necessary for the screw holes in the M-CA and the screw holes in the M-BP to align since we are using self-drilling screws, but it is important that the edge of the casing underneath the screw holes in the M-BP so there is something to drill into.		
Acceptance criteria:	The system is able to place the E-REG with a TBD mm precision		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.17 AS027 Fasten the M-BP

Test Case ID	**AS027**	Fasten the M-BP		
Activity ID	xxx		Test number:	0
Requirement ID	R08.04.03			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.05.01			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	M-BP – The Bottom plate The M-BP is already in place and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a "fasten the screw"-program until the M-BP firmly fastened.			
Acceptance criteria:	The system is able to fasten the M-BP with given screws			
Comment:	What was the outcome of the test?			
Tested by:	"name"			

A.7.3.18 AS028 Hold the M-BP while fastening it

Test Case ID	**AS028**	Hold the M-BP while fastening it		
Activity ID	xxx		Test number:	0
Requirement ID	R08.04.04			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.05.01			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	M-BP – The Bottom plate The M-BP is held in place by the system, and the robot has the screwdriver ready with a screw at the screw bit. We and not the vision system give coordinates for the screw hole. The system will then move the screwdriver in position and run a "fasten the screw"-program until the M-BP firmly fastened.			
Acceptance criteria:	The system is able to hold the M-BP while fastening it with given screws			
Comment:	What was the outcome of the test?			
Tested by:	"name"			

A.7.3.19 AS013 Replace human screw process

A.7.3.19.1 Test number 1: Fastening the bottom plate (completed, success)

Test Case ID	**AS013**	Replace human screw process		
Activity ID	119		Test number:	1
Requirement ID	R08.01			
Priority	High			
Test type:	Black Box			
Status:	Success			
Created date:	2015.02.19			
Due date:				
Completed date:	2015.04.22			
Test specification:	*The intention of the test is to perform the tests AS001-AS003 tests in series without stop nor operator interaction:* For this test, we use the task of fastening the bottom plate (M-BP) to the casing (M-CA). The screw is placed in a known position and the coordinates for the fastening-hole for the bottom plate is programmed into the system. There is no pre-drilled hole in the casing underneath the hole in the bottom plate. We use self-drilling screws to place in the fastening-hole and to drill through the casing. The robot run the “get a screw”-program, places the screw according to the given coordinates and the run the “fasten the screw”-program.			
Acceptance criteria:	The automation system has successfully fastened the target screw into the target hole without error.			
Comment:	We used the handheld pneumatic push-to-start screwdriver[3] borrowed from Engmark Meteor to perform this test. We used 3,9x13mm AW self-drilling screws, AW bit and a magnetic bit holder. Picking and placing the screw went perfect. Just like in test AS001 and AS002(test#1). There was no problem fastening the screw either, but the self-drilling screws take some time getting through the material of the casing. 5-6 seconds average. The screws we are using need about 1700rpm, but the screwdriver can only do 500. The test was successful, but the time will be significantly lowered with a faster screwdriver. NB! The casing is not always a perfect circle like the bottom plate so we have to come up with a smart solution to this problem.			
Tested by:	Thor Eirik Johnsrud, Ivar Roskifte Leikarnes, Kristian Dønheim Kvam			

[3] More information about the screws and screwdriver in Technology Document Screwdriver

A.7.3.20 AS014 Mount E-CC without human interaction

Test Case ID	**AS014**	Mount E-CC without human interaction		
Activity ID	xxx		Test number:	0
Requirement ID	R08.02			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.02.19			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	*The intention of the test is to perform the tests AS004-AS006 and AS023 tests in series without stop nor operator interaction:* The automation system picks up the E-CC from a known position, places it, gets the correct screw and then fasten it while holding it in place.			
Acceptance criteria:	The automation system has successfully picked up, placed and fastened the E-CC without error			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.21 AS015 Mount E-PI without human interaction

Test Case ID	**AS015**	Mount E-PI without human interaction		
Activity ID	xxx		Test number:	0
Requirement ID	R08.03			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.02.19			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	*The intention of the test is to perform the tests AS007-AS009 tests in series without stop nor operator interaction:* The automation system picks up the E-PI from a known position, places it, then gets the correct screws and fasten it.			
Acceptance criteria:	The automation system has successfully picked up, placed and fastened the E-PI without error			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.22 AS016 Mount E-REG without human interaction

Test Case ID	**AS016**	Mount E-REG without human interaction		
Activity ID	xxx		Test number:	0
Requirement ID	R08.04			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.02.19			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	*The intention of the test is to perform the tests AS010-AS012 and AS024 tests in series without stop nor operator interaction:* The automation system picks up the E-REG from a known position, places it, then gets the correct screws and fasten it while holding it.			
Acceptance criteria:	The automation system has successfully picked up, placed and fastened the E-REG without error			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.23 AS029 Mount M-BP without human interaction

Test Case ID	**AS029**	Mount M-BP without human interaction		
Activity ID	xxx		Test number:	0
Requirement ID	R08.05			
Priority	High			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.05.01			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	*The intention of the test is to perform the tests AS025-AS028 tests in series without stop nor operator interaction:* The automation system picks up the M-BP from a known position, places it, then gets the correct screws and fasten it while holding it.			
Acceptance criteria:	The automation system has successfully picked up, placed and fastened the M-BP without error			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.24 AS017 Resume assembly process after an interruption(test not defined)

Test Case ID	**AS017**	Resume assembly process after an interruption		
Activity ID	xxx		Test number:	0
Requirement ID	R08.09.03			
Priority	Low			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.02.19			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	The automation system will be interrupted during <assembly action/process>.			
Acceptance criteria:	The system continues from where it left off after the interruption with no operator interaction.			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.3.25 AS018 (test not defined)

Test Case ID	**AS018**	Precision test	
Activity ID	xxx	Test number:	0
Requirement ID	R08.09.04A		
Priority	High		
Test type:	Black box/Static		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The automation system will complete its assembly task and a trained assembler will inspect the product.		
Acceptance criteria:	The assembler is unable to identify that the automation system has assembled the product. Less than 1mm (TBD) shift in x,y,z Mounts objects with equivalent force		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.26 AS019 (test not defined)

Test Case ID	**AS019**	Time test	
Activity ID	xxx	Test number:	0
Requirement ID	R08.09.04B		
Priority	High		
Test type:	Black box/Static		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The automation system will complete its assembly task and a trained assembler will inspect the product.		
Acceptance criteria:	The system uses the same time amount to finish the task as an experienced assembler (According to time-analysis (see doc.)) The assembler is unable to identify that the automation system has assembled the product.		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.27 AS020 (test not defined)

Test Case ID	**AS020**	Time test 2	
Activity ID	xxx	Test number:	0
Requirement ID	R08.09.04C		
Priority	High		
Test type:	Black box/Static		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The automation system will complete its assembly task and a trained assembler will inspect the product.		
Acceptance criteria:	The system uses the same time amount to finish the task 20% TBD faster than an experienced assembler does. (According to time-analysis (see doc.)) The assembler is unable to identify that the automation system has assembled the product.		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.3.28 **AS021 Screw placement precision requirement (completed, success)**

Test Case ID	**AS021**	Screw placement precision requirement		
Activity ID	120		Test number:	1
Requirement ID	R08.01.02			
Priority	High			
Test type:	Troubleshooting			
Status:	Success			
Created date:	2015.03.20			
Due date:				
Completed date:	2015.04.24			
Test specification:	This test will determine the required precision for the screw placement for the screw to fasten in the desired hole. This is an extension of AS003 and a pre-test to AS002. The self-drilling screws drill their own holes so the only need of precision is relative to the outer hole the screw is entering. These screws are pointy at the tip. The M4 machine screws have threads that need to fit. They are also flat at the tip and have a slightly bigger diameter than the self-drilling screws. We will try to fasten the screws as in a real assembly situation, but we will gradually move the impact point for the screw away from the center of the hole for each run with increments of 0,5mm. The test will continue until the distance from the center is so big that the screw fails to fasten correctly.			
Acceptance criteria:	The test is complete when we have determined the maximum deviation from the center of the screw hole where the screw is fastened.			
Comment:	For the screws to slide into their holes there can be no more than 1,5mm deviation from the center of the hole.			
Tested by:	Thor Eirik Johnsrud, Ivar Roskifte Leikarnes			

A.7.3.29 AS022 Perform assembly-action TBD times without error

Test Case ID	**AS022**	Perform assembly-action TBD times without error	
Activity ID	xxx	Test number:	0
Requirement ID	R08.09.03		
Priority	High		
Test type:	Black Box		
Status:	Success, fail		
Created date:	2015.02.21		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The automation system will perform <assembly action> TBD times in a row without human interaction		
Acceptance criteria:	The system performs the assembly action TBD times without error		
Comment:	What was the outcome of the test?		
Tested by:	"name"		

A.7.4 Vision Test Cases

A.7.4.1 VS001 Recognize screw hole with given orientation

Test Case ID	**VS001**	Recognize screw hole with given orientation	
Activity ID	170	Test number:	1
Requirement ID	R02.01A		
Priority	High		
Test type:	Black Box, Dynamic		
Status:	Success		
Created date:	2015.02.18		
Due date:	2015.05.05		
Completed date:	2015.05.05		
Test specification:	Screw holes to be recognized will be put in a low noise area. The angle and distance from the camera lens will be 40 degrees compared to tool and 0.4 meters.		
Acceptance criteria:	Vision system recognizes 9 out of 10 times, on all specified distances		
Comment:	The system was successfully able to recognize the screw holes		
Tested by:	EFS		

A.7.4.2 VS002 Optimal camera angle

Test Case ID	**VS002**	Optimal camera angle	
Activity ID	172	Test number:	1
Requirement ID	R02.01B		
Priority	High		
Test type:	Black Box		
Status:	Success		
Created date:	2015.02.20		
Due date:	2015.05.05		
Completed date:	2015.05.05		
Test specification:	The camera will be put in an angle of 0 to 90 degrees, with 15-degree steps.		
Acceptance criteria:	The range of favourable angles, giving the best resolution		
Comment:	The closer the angle is to 90 degrees, the less deviation we will get when translating coordinates. The internal lighting of the camera and the reflective surface of the product means we need an angle where the light source is not directly reflected. This proved to be at 50 degrees from plane.		
Tested by:	EFS, TEJ, DS		

A.7.4.3 VS003 Optimal camera range

Test Case ID	**VS003**	Optimal camera range		
Activity ID	173		Test number:	1
Requirement ID	R02.01A			
Priority	High			
Test type:	Black Box			
Status:	Success			
Created date:	2015.02.20			
Due date:	2015.05.05			
Completed date:	2015.05.05			
Test specification:	The camera will be put in a range of 0 m to 5 meters, with 0.1 m steps.			
Acceptance criteria:	The range of favourable distance, giving the best resolution			
Comment:	The closer the camera, the more detailed picture we get and less deviation during coordinate translation. The camera cannot be too close to the tool. The optimal range proved to be 180mm.			
Tested by:	EFS, TEJ			

A.7.4.4 VS004 Recognize object in any orientation

Test Case ID	**VS004**	Recognize object in any orientation	
Activity ID	169	Test number:	1
Requirement ID	R02.01B.XX		
Priority	High		
Test type:	Black Box, Dynamic		
Status:	Success		
Created date:	2015.02.18		
Due date:	2015.05.05		
Completed date:	2015.05.05		
Test specification:	The object to be recognized will be put in a low noise area. The distance from the camera lens will be 25cm. The object will be placed with a random rotation for each test.		
Acceptance criteria:	Vision system recognizes the object from different angles, 9 out of 10 times.		
Comment:	As long as the model is properly defined in the IFM configuration software, the camera will always find it.		
Tested by:	EFS		

A.7.4.5 VS005 Recognize object orientation

Test Case ID	**VS005**	Recognize object orientation	
Activity ID	Xxx	Test number:	xxx
Requirement ID	R02.01.02		
Priority	Low		
Test type:	Black Box, Dynamic		
Status:	(Success/fail)		
Created date:	2015.02.18		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The object to be recognized will be put in a low noise area. The distance from the camera lens will be 25 cm. The object will be placed with a random rotation for each test.		
Acceptance criteria:	Recognize the object, with rotation in all three planes, with a margin of +/- 10 degrees		
Comment:	What was the outcome of the test?		
Tested by:			

A.7.4.6 VS006 Recognize object from 3D CAD

Test Case ID	**VS006**	Recognize object from 3D CAD	
Activity ID	Xxx	Test number:	xxx
Requirement ID	R02.01.03		
Priority	Low		
Test type:	Black Box, Dynamic		
Status:	Success, fail		
Created date:	2015.02.18		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The object to be recognized will be put in a low noise area. The angle and distance from the camera lens will be 25cm and 40 degrees. The object will be placed with a random rotation for each test.		
Acceptance criteria:	Recognize object from a dataset, 9 out of 10 times		
Comment:	What was the outcome of the test?		
Tested by:	"name"		

A.7.4.7 VS007 Recognize object through vision scanning

Test Case ID	**VS007**	Recognize object through vision scanning	
Activity ID	Xxx	Test number:	xxx
Requirement ID	R02.01.04		
Priority	Low		
Test type:	Black box, Dynamic		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The object to be recognized will be put in a low noise area. The angle and distance from the camera lens will be 25cm and 45 degrees. The object will be placed with a random rotation for each test.		
Acceptance criteria:	Recognize an object from a dataset, 9 out of 10 times		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.4.8 VS008 Neural networking

Test Case ID	**VS008**	Neural networking		
Activity ID	Xxx		Test number:	xxx
Requirement ID	R02.01.05			
Priority	Low			
Test type:	Black Box			
Status:	Success, fail			
Created date:	2015.02.19			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	Recognize <an object> with precision x, and after running a learning script TBD times, the precision is increased by TBD percent.			
Acceptance criteria:	Provable improvement of precision when recognizing objects TBD			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.4.9 VS009 Recognize overlapping objects of the same type

Test Case ID	**VS009**	Recognize overlapping objects of the same type	
Activity ID	Xxx	Test number:	xxx
Requirement ID	R02.01.06		
Priority	Low		
Test type:	Black Box, Dynamic		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	Put TBD number of one <specific object> in a box of dimensions TBD. The angle and distance from the camera lens will be [TBD]. Scramble the box between tests.		
Acceptance criteria:	The vision system identifies at least 1 <object> and lists according to accessibility, among several of the same <objects>		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.4.10 **VS010 Recognize overlapping objects of different types**

Test Case ID	**VS010**	Recognize overlapping objects of different types	
Activity ID	Xxx	Test number:	xxx
Requirement ID	R02.01.07		
Priority	Low		
Test type:	Black Box, Dynamic		
Status:	Success, fail		
Created date:	2015.02.19		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	Put TBD number of TBD <different objects> in a box of dimensions TBD. The angle and distance from the camera lens will be [TBD]. Scramble the box between tests.		
Acceptance criteria:	The vision system identifies at least 1 <object>, specifies type, and lists according to accessibility, among several <different objects>		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.4.11 VS011 Recognize holes

Test Case ID	**VS011**	Recognize holes	
Activity ID	191	Test number:	1
Requirement ID	R02.02		
Priority	High		
Test type:	Black Box, Dynamic		
Status:	Success		
Created date:	2015.02.19		
Due date:	2015.05.12		
Completed date:	2015.05.12		
Test specification:	The object to be recognized will be put in a low noise area. The angle and distance from the camera lens will be 50 degrees and 180mm. The object will be placed with a random rotation for each test.		
Acceptance criteria:	The vision system identifies and locates a 3mm hole in a 1mm metal-sheet from a distance of 180mm, with a precision of 2mm in all directions		
Comment:	The software successfully recognizes the hole, and the image plane coordinates are translated to real-world coordinates with acceptable accuracy		
Tested by:	EFS		

A.7.4.12 VS013 Vision output response test

Test Case ID	**VS013**	Vision output response test		
Activity ID	171		Test number:	1
Requirement ID	R02.03			
Priority	High			
Test type:	Black Box, Dynamic			
Status:	Success			
Created date:	2015.02.19			
Due date:	2015.05.05			
Completed date:	2015.05.05			
Test specification:	Request output from vision system, and measure time until information is received.			
Acceptance criteria:	Vision system output frequency is more than 3Hz			
Comment:	Even with multiple models in a high noise stage the processing time never went over 3 seconds.			
Tested by:	EFS			

A.7.4.13 VS014 Interface with existing control-system

Test Case ID	**VS014**	Interface with existing control-system		
Activity ID	190		Test number:	1
Requirement ID	R02.04			
Priority	High			
Test type:	Black Box, Dynamic			
Status:	Success			
Created date:	2015.02.19			
Due date:	2015.05.12			
Completed date:	2015.05.12			
Test specification:	The vision system will be connected to the UR system, and given a specific task.			
Acceptance criteria:	The vision system completes the task as expected			
Comment:	The interfacing between both IFMsensor and controller, and the controller and the UR script works properly.			
Tested by:	EFS, TEJ			

A.7.5 Ease of Use Test Cases

A.7.5.1 EoU001 Verification and validation of ease of use

Test Case ID	**EoU001**	Verification and validation of ease of use	
Activity ID	xxx	Test number:	0
Requirement ID	R03		
Priority	High		
Test type:	Black Box		
Status:	Success/fail		
Created date:	2015.03.22		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	Our client, Håvid Engmark, will evaluate the ease of use of the system.		
Acceptance criteria:	Our client approves the ease of use.		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.5.2 EoU002 Component shortage user notification

Test Case ID	**EoU002**	Component shortage user notification	
Activity ID	xxx	Test number:	0
Requirement ID	R03.03		
Priority	Low		
Test type:	Black Box		
Status:	Success/fail		
Created date:	2015.02.20		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The system will be given a task to pick up and place components. We will prepare 1 component less than the system is supposed to pick up. The components will be placed so that the robot knows exactly where they are.		
Acceptance criteria:	The system gives a notification when there is a shortage of components		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.5.3 EoU003 Training for operation of the system (test not defined)

Test Case ID	**EoU003**	Training for operation of the system	
Activity ID	xxx	Test number:	0
Requirement ID	R03.04		
Priority	High		
Test type:	Black Box		
Status:	Success/fail		
Created date:	2015.02.20		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	A person that knows the system will be given a task to teach a new person. The person can choose to use system manual to explain if needed. The trainee has not worked with the system before		
Acceptance criteria:	An unexperienced person shall be able to use the system after TBD minutes of training		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.5.4 EoU004 Awareness of system surroundings (test not defined)

Test Case ID	**EoU004**	Awareness of system surroundings	
Activity ID	xxx	Test number:	0
Requirement ID	R03.05		
Priority	Low		
Test type:	Black Box		
Status:	Success/fail		
Created date:	2015.02.20		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The system will be given a task to perform with a human nearby/assisted by an assembler. The system is then supposed to finish the task without harming the human/assembler.		
Acceptance criteria:	The system actively avoids hitting/hurting humans		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.5.5 EoU005 230V AC outlet

Test Case ID	**EoU005**	230V AC outlet	
Activity ID	xxx	Test number:	0
Requirement ID	R03.06.01		
Priority	High		
Test type:	Black Box		
Status:	Success/fail		
Created date:	2015.02.20		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The system will be connected to a 230V AC outlet.		
Acceptance criteria:	The system connects to a 230V AC outlet.		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.5.6 EoU System error user notification (test not defined)

Test Case ID	**EoU006**	System error user notification		
Activity ID	xxx		Test number:	0
Requirement ID	R03.02			
Priority	Low			
Test type:	Black Box			
Status:	Success/fail			
Created date:	2015.02.20			
Due date:	yyyy.mm.dd			
Completed date:	yyyy.mm.dd			
Test specification:	The system will be given an assembly task and we will set up the working area so that the system is unable to complete the task.			
Acceptance criteria:	The system gives a notification when it encounters a general error			
Comment:	What was the outcome of the test?			
Tested by:	“name”			

A.7.5.7 EoU007 Number of commands to start the system

Test Case ID	EoU007	Number of commands to start the system	
Activity ID	xxx	Test number:	xxx
Requirement ID	R03.07		
Priority	Medium		
Test type:	Black box		
Status:	Success, fail		
Created date:	2015.03.22		
Due date:	yyyy.mm.dd		
Completed date:	yyyy.mm.dd		
Test specification:	The initial setup is complete. We are then going to start the system. The system has to be able to start by performing the fewest commands possible.		
Acceptance criteria:	The system starts after TBD commands.		
Comment:	What was the outcome of the test?		
Tested by:	“name”		

A.7.6 General Production Test Cases

A.7.6.1 GP001 SL/M-REF properties

A.7.6.1.1 Test number 1: First batch, Starlock_4.75_4.35_120d (completed, fail)

Test Case ID	GP001	SL/M-REF properties		
Activity ID	81		Test number:	1
Requirement ID	R09			
Priority	Medium			
Test type:	Black box, statistic			
Status:	Fail			
Created date:	2015.03.14			
Due date:	2015.03.16			
Completed date:	2015.03.16			
Test specification:	First batch, Starlock_4.75_4.35_120d SL/M-REF – The cutout in the M-REF that will replace the current starlock. M-REF – Reflector plate I-IR – Insulation ring Assumptions: - The cutouts sent from Meteor have the same properties as if they were in an actual M-REF. - The size of both the I-IR and the cutouts are consistent. - Labelling and cutouts match. We push the I-IR in the different cutouts by hand and analyse the following: - The average time it takes to pick up an I-IR and mount it. - How well the I-IR is fastened. - The damage inflicted to the surrounding metal.				
Acceptance criteria:	Reduction of the time spent to mount the I-IR, the I-IR is firmly attached and the M-REF and I-IR has no distinct damages.			
Comment:	The test failed because the M-REF was damaged when the E-IR was pushed in place and the E-IR was not properly attached. Timing the test was not necessary.			
Tested by:	Deividas Svaikauskas			

A.7.6.1.2 Test number 2: First batch, Starlock_4.95_3.8_40d (completed, fail)

Test Case ID	GP001	SL/M-REF properties	
Activity ID	81	Test number:	2
Requirement ID	R09		
Priority	Medium		
Test type:	Black box, statistic		
Status:	Fail		
Created date:	2015.03.14		
Due date:	2015.03.16		
Completed date:	2015.03.16		
Test specification:	First batch: Starlock_4.95_3.8_40d SL/M-REF – The cutout in the M-REF that will replace the current starlock. M-REF – Reflector plate I-IR – Insulation ring Assumptions: - The cutouts sent from Meteor have the same properties as if they were in an actual M-REF. - The size of both the I-IR and the cutouts are consistent. - Labelling and cutouts match. We push the I-IR in the different cutouts by hand and analyse the following: - The average time it takes to pick up an I-IR and mount it. - How well the I-IR is fastened. - The damage inflicted to the surrounding metal.		
Acceptance criteria:	Reduction of the time spent to mount the I-IR, the I-IR is firmly attached and the M-REF and I-IR has no distinct damages.		
Comment:	The test failed because the M-REF was damaged when the E-IR was pushed in place and the E-IR was not properly attached. Timing the test was not necessary.		
Tested by:	Deividas Svaikauskas		

A.7.6.1.3 Test number 3: First batch, Starlock_5.0_3.27_15d (completed, fail)

Test Case ID	GP001	SL/M-REF properties	
Activity ID	81	Test number:	3
Requirement ID	R09		
Priority	Medium		
Test type:	Black box, statistic		
Status:	Fail		
Created date:	2015.03.14		
Due date:	2015.03.16		
Completed date:	2015.03.16		
Test specification:	First batch: Starlock_5.0_3.27_15d SL/M-REF – The cutout in the M-REF that will replace the current starlock. M-REF – Reflector plate I-IR – Insulation ring Assumptions: - The cutouts sent from Meteor have the same properties as if they were in an actual M-REF. - The size of both the I-IR and the cutouts are consistent. - Labelling and cutouts match. We push the I-IR in the different cutouts by hand and analyse the following: - The average time it takes to pick up an I-IR and mount it. - How well the I-IR is fastened. - The damage inflicted to the surrounding metal.		
Acceptance criteria:	Reduction of the time spent to mount the I-IR, the I-IR is firmly attached and the M-REF and I-IR has no distinct damages.		
Comment:	The test failed because the E-IR was not properly attached. The M-REF was not damaged. Timing the test was not necessary.		
Tested by:	Deividas Svaikauskas		

A.7.6.1.4 Test number 4: First batch, Starlock_5.5_3.95_35d (completed, fail)

Test Case ID	GP001	SL/M-REF properties	
Activity ID	81	Test number:	4
Requirement ID	R09		
Priority	Medium		
Test type:	Black box, statistic		
Status:	Fail		
Created date:	2015.03.14		
Due date:	2015.03.16		
Completed date:	2015.03.16		
Test specification:	First batch: Starlock_5.5_3.95_35d SL/M-REF – The cutout in the M-REF that will replace the current starlock. M-REF – Reflector plate I-IR – Insulation ring Assumptions: - The cutouts sent from Meteor have the same properties as if they were in an actual M-REF. - The size of both the I-IR and the cutouts are consistent. - Labelling and cutouts match. We push the I-IR in the different cutouts by hand and analyse the following: - The average time it takes to pick up an I-IR and mount it. - How well the I-IR is fastened. - The damage inflicted to the surrounding metal.		
Acceptance criteria:	Reduction of the time spent to mount the I-IR, the I-IR is firmly attached and the M-REF and I-IR has no distinct damages.		
Comment:	The test failed because the E-IR was not properly attached. The M-REF was not damaged. Timing the test was not necessary.		
Tested by:	Deividas Svaikauskas		

B. Project management tools

B.1 Cloud storage Services

Abstract

This document contains information about considered cloud storage services.

Contents

Abstract **121**

Contents **121**

Figures **121**

B.1.1 Introduction **122**

B.1.2 Relevant technologies **122**

B.1.2.1 Dropbox 122

B.1.2.1.1 Pros 122

B.1.2.1.2 Cons 122

B.1.2.2 Microsoft OneDrive 122

B.1.2.2.1 Pros 122

B.1.2.2.2 Cons 122

B.1.2.3 Google Drive 122

B.1.2.3.1 Pros 122

B.1.2.3.2 Cons 123

B.1.3 Conclusion **123**

B.1.4 References **123**

Figures

Figure 21: Dropbox logo 122

Figure 22: OneDrive logo 122

Figure 23: Google Drive logo 122

B.1.1 Introduction

The project will produce a lot of documents and files in general, and since everybody on the project group needs access to several files, we need a system for sharing the files. The main applications we have considered are Dropbox, Microsoft OneDrive and Google Drive.

B.1.2 Relevant technologies

B.1.2.1 Dropbox

Dropbox is an application you can download to your computer. It creates a storage folder that is stored locally on your computer that synchronizes with online servers every time you are connected to the internet. Selected folders within this storage folder can easily be shared with other Dropbox-users.

Figure 21: Dropbox logo

B.1.2.1.1 Pros

- Easy to use
- First 2 GB free
- Most of us already have a Dropbox-account

B.1.2.1.2 Cons

- Tools for collaborative editing of files costs approximately 8 EUR per user per month (for students)
- Extra storage space costs approximately 10 EUR per user per month (for 1 TB)

B.1.2.2 Microsoft OneDrive

OneDrive has many of the same features as Dropbox; it can be downloaded and synced in the same way. It does not seem to be as easy to use as Dropbox, but you get 15GB for free and tools for collaborative editing. It appears to be a bit buggy.

Figure 22: OneDrive logo

B.1.2.2.1 Pros

- 15 GB free space
- More tools than Dropbox free
- Very compatible with MS Office

B.1.2.2.2 Cons

- Buggy
- None of us have used it for sharing before

B.1.2.3 Google Drive

Google Drive is mostly the same functionality as Dropbox and OneDrive, but it uses online google- tools for collaborative editing.

Figure 23: Google Drive logo

B.1.2.3.1 Pros

- 15 GB free space

- Collaborative editing (in google tools)
- Doesn’t need MS office

B.1.2.3.2 Cons

- Converting to standard file types such as .docx sometimes messes up the document

B.1.3 Conclusion

To get started fast, we went with Dropbox, since this was the most familiar application. We see now that 2 GB storage space might be insufficient, and that tools for collaborative editing would be advantageous. Solutions might be to upgrade to Dropbox for business, or change to OneDrive or Google Drive.

B.1.4 References

1. Dropbox, home page [18.02.2015]
   www.Dropbox.com
2. Onedrive, home page [18.02.2015]
   www.onedrive.com
3. Google Drive, home page [18.02.2015]
   www.drive.google.com

B.2 Communication Technologies

Abstract

This document contains information about our choice of communication technology besides calling on the phone and send e-mails as an alternative to personal meetings.

Contents

Abstract **124**

Contents **124**

Figures **124**

B.2.1 Introduction **125**

B.2.2 Relevant technology **125**

B.2.2.1 appear.in 125

B.2.2.1.1 Pros 126

B.2.2.1.2 Cons 126

B.2.3 Conclusion **126**

B.2.4 References **126**

Figures

Figure 24: appear.in logo 125

B.2.1 Introduction

In this project, we have to communicate with several different people. Some of them are close by so we can have frequent meetings with them in person. Others are located further away so personal meetings becomes a little more complicated to conduct. In an e-mail, we can send questions, answers, give updates, send pictures etc., but it is not very time efficient. A phone call solves some of the problems when we just have to talk. A conversation over the phone is easy when there are two or maybe three people talking. When there are more people, it becomes difficult to keep track of who is talking to whom. It is also not possible to show something visual through a phone call.

A solution that combines oral and visual communication is video chatting. There are several video chat tools, but the one that fell in our taste is appear.in.

B.2.2 Relevant technology

B.2.2.1 appear.in

appear.in is a browser-based free video chat for up to eight people at the same time. It started as a summer project to explore the use of WebRTC(Web Real-Time Communication). One of the great features is that there is no login or registration required. There is no need for installing files or plugins either. This makes it easy to use and to set up a video conference is very quick.

Figure 24: appear.in logo

There are three steps to follow set up a video conference with appear.in:

1. Go to appear.in and create a room. There are generated about 2000 different names for rooms, but it is possible to create own unique names. When the name is decided, press start.
2. Copy the link to the room and send it to the ones that will be part of the conversation. The link is in the form: **appear.in/roomname**
3. Wait for the other parties to appear in appear.in and start the conference.

When the room is chosen, you can lock the room so anyone who wants to join has to “knock” to enter. This is to avoid other people disturbing private conversations. The site is not limited to video chat only; it also has a text chat. It is possible to turn of the camera and/or mute your microphone.

In Google Chrome, screen sharing is available. You can choose to show your whole screen or just a selected window from e.g. a web browser or Microsoft word. This is a great feature for when you have to show something from your computer screen to others and not just yourself and your environment.

To access more features, you will have create a profile. With a profile, you can claim the URL and keep the room locked even when you are not present. The extension for Google Chrome lets you enable notifications for when somebody try to access your room. This extension needs access to the tabs and browsing data to avoid sending notifications when you are using appear.in. Another feature is the possibility to add a custom background to personalize your room.

appear.in is also available for smart phones and tablets. The appear.in app is currently just for iPhone, but an app for android is in progress. It is possible to use appear.in on android phones if they have Google Chrome, Opera or Mozilla Firefox web browser. This is because these web browsers currently are the only ones that support WebRTC.

B.2.2.1.1 Pros

- Easy to use
- Free
- No need for login
- Video chat
- Text chat
- Up to eight devices can be connected at the same time
- Screen sharing

B.2.2.1.2 Cons

- No text chat in app

B.2.3 Conclusion

We chose appear.in as an alternative communication technology because it is easy to use, quick to set up video conference and it has all the different features that we need to exchange information. There were no need to consider other alternatives because appear.in met all our needs.

B.2.4 References

1. Facebook. *appear.in.* [13.03.2015]
   https://www.facebook.com/appear.in.video/info?tab=page_info
2. WebRTC. *WebRTC.* [13.03.2015]
   http://www.webrtc.org/
3. Chrome Web Store. *appear.in.* [13.03.2015]
   https://chrome.google.com/webstore/detail/appearin/pokjppmpccggefgcenpngoleemajgnmo?hl=en
4. Telenor. *appear.in.* [13.03.2015]
   http://www.telenor.com/media/articles/2014/telenor-pioneers-video-conferencing-in-the-browser/
5. appear.in. *appear.in.* [13.03.2015]
   https://appear.in/

B.3 Scheduling Software

Abstract

This document contains information about the considered schedule software programs for this project.

Contents

Abstract 127

Contents 127

Figures 127

B.3.1 Introduction 128

B.3.2 Relevant technologies 128

B.3.2.1 Microsoft Project 128

B.3.2.1.1 Pros 128

B.3.2.1.2 Cons 128

B.3.2.2 Microsoft Excel 128

B.3.2.2.1 Pros 128

B.3.2.2.2 Cons 128

B.3.2.3 SmartDraw 129

B.3.2.3.1 Pros 129

B.3.2.3.2 Cons 129

B.3.3 Conclusion 129

B.3.4 References 129

Figures

Figure 25: Microsoft Project logo 128

Figure 26: Microsoft Excel logo 128

Figure 27: SmartDraw logo 129

B.3.1 Introduction

To manage the project we decided to set up a Gantt-diagram. This will provide us with needed time-management and overall task organization. It is important to choose a program that will be easy to use during the project and at the same time give a clear overview for non-project members. For this task we considered Microsoft Project, Microsoft Excel, and SmartDraw.

B.3.2 Relevant technologies

B.3.2.1 Microsoft Project

Microsoft Project is software designed by Microsoft to give project managers and team members support when working with a project. It is mainly designed for this purpose and therefore there are many possibilities when working with this software. Microsoft Project has a lot to offer, but it takes time to learn it completely. It is designed so that even a new user should be able to easily set up a simple, yet fully functional management of a project. This software is widely used in the management world.

Figure 25: Microsoft Project logo

B.3.2.1.1 Pros

- Free (for students at HBV)
- Many tutorials online
- Easy to use

B.3.2.1.2 Cons

- Big learning curve
- You need Microsoft Project to open the diagram
- When trying to print or exporting to pdf file the diagram gets split up into parts

B.3.2.2 Microsoft Excel

The wonders of Microsoft Excel are many, including the possibility of setting up project management diagrams. Many have worked with Excel at least once in their lifetime and therefor it is a popular option. There are many tutorials online explaining how one could set up a Gantt-diagram. The drawback is that it is a bit more difficult than it might look.

Figure 26: Microsoft Excel logo

B.3.2.2.1 Pros

- Free (for students at HBV)
- Many tutorials online

B.3.2.2.2 Cons

- Not so easy to use
- None of us have used it for scheduling before

B.3.2.3 SmartDraw

SmartDraw is used to create project charts, flowcharts, organization charts and such. It is software designed to manage whole projects. It provides the users with many different possibilities when it comes to managing both small and big projects. The biggest drawback is the cost. This was the reason to eliminate this option almost at once.

Figure 27: SmartDraw logo

B.3.2.3.1 Pros

- There is a free version of the program

B.3.2.3.2 Cons

- The free version is very limited
- Not so easy to use
- Pricey (197 dollars)

B.3.3 Conclusion

We decided to go with Microsoft Project. The reason for this is that it gave us the most benefits from the three candidates we considered. The simplicity of the software made it easy for us to set up an early version of the Gantt-diagram. Since our project does not require complex scheduling tools, Microsoft Project is a good choice for us. We also have prior experience with the program from the System engineering course.

B.3.4 References

1. Office. *Project*. [03.03.2015]
   https://products.office.com/en-us/project/project-and-portfolio-management-software
2. Office. *Excel*. [03.03.2015]
   https://products.office.com/en-us/excel
3. SmartDraw. *Absolutely the Easiest Way to Create Charts, Diagrams & Other Visuals*. [03.03.2015]
   http://www.smartdraw.com/
4. EPMA. *MICROSOFT PROJECT VS MICROSOFT EXCEL*. [03.03.2015]
   http://epmainc.com/blog/microsoft-project-vs-microsoft-excel-0
5. EPM, Microsoft Project and You. *Microsoft Project vs Excel*. [03.03.2015]
   http://pmpspecialists.com/Blog/2009/10/microsoft-project-vs-excel/

C. Analysis

C.1 Risk analysis and management

C.1.1 Abstract

This document addresses the risks regarding the project. Every risk we have found relevant is described and assigned a grade for impact and probability. Each risk has a mitigation strategy and a contingency plan. Eliminated, handled or occurred risks are removed from the list of risk and listed in compressed form in the chapter for eliminated, handled and occurred risks at the end of the document.

Contents

C.1.1 Abstract........130

Contents........130

Tables........132

C.1.2 Introduction........132

C.1.2.1 Risk analysis........132

C.1.2.2 Risk matrix........132

C.1.2.2.1 Green........132

C.1.2.2.2 Light green........132

C.1.2.2.3 Yellow........133

C.1.2.2.4 Orange........133

C.1.2.2.5 Red........133

C.1.2.3 Develop mitigation strategy........133

C.1.2.4 Develop contingency plan........133

C.1.2.5 Effective risk........133

C.1.2.6 Risk template........133

C.1.3 Areas of risk........134

C.1.3.1 Completion of the project........134

C.1.3.1.1 Risk management........134

C.1.3.1.2 Limitation of the project........134

C.1.3.2 Vision Development........134

C.1.3.2.1 Not due in time 134
C.1.3.2.2 Not being able to recognize the components to be placed 135
C.1.3.2.3 Not being able to find the “place” position for the components 135
C.1.3.3 Loss of data.......... 135
C.1.3.4 Interfacing the vision system with UR-system 136
C.1.3.5 Screwdriver 136
C.1.3.5.1 Not being able to fasten screws for the bottom plate M-BP 136
C.1.3.5.2 Not being able to fasten the screws for the regulator E-REG 136
C.1.4 Eliminated/handled/occurred risks 137
C.1.4.1 Vision development 137
C.1.4.1.1 Risk: Not receiving necessary equipment in time. 137
C.1.4.1.2 Risk: Not being able to locate holes 137
C.1.4.1.3 Risk: Necessary equipment does not work properly 137
C.1.4.1.4 Risk: Necessary equipment gets broken 137
C.1.4.2 UR-10.......... 137
C.1.4.2.1 Risk: Not receiving it in time. 137
C.1.4.2.2 Risk: Not managing to control the UR-10 with UR-Script 137
C.1.4.3 Gripper 137
C.1.4.3.1 Risk: Lack of gripper testing due to late delivery. 137
C.1.4.3.2 Risk: Ordering a gripper that turns out to not fit our purpose 138
C.1.4.4 Waiver of group members 138
C.1.4.4.1 Risk: Sickness.......... 138
C.1.4.4.2 Risk: Group member has to withdraw from the project.......... 138
C.1.4.5 Screwdriver 138
C.1.4.5.1 Risk: Not being able to pick up screws.......... 138
C.1.4.5.2 Risk: Not being able to hold screw when moving around 138
C.1.4.5.3 Risk: Not ordering and receiving new screwdriver in time to test properly before delivering final report. 138
C.1.4.6 Economy.......... 138
C.1.4.6.1 Risk: Insufficient resources.......... 138
C.1.4.7 Griddle.......... 139
C.1.4.7.1 Risk: New design due to new regulations 139

Tables

Table 1: Risk Matrix 132
Table 2: Risk template 133

C.1.2 Introduction

There are risks involved in every project. Big and small, positive and negative. Risks cover every aspect from late delivery to waiver of group members. To be a step ahead of possible risks that may jeopardize the project, we have identified and analysed relevant risks.

C.1.2.1 Risk analysis

We are considering two factors when we are analysing risks; impact and probability. How much it affects the project and how likely it is to happen. The impact ranges from 1-5 where 1 is negligible, 2 is marginal, 3 is moderate, 4 is major and 5 is catastrophic. The probability is also divided in five stages where 1 is very unlikely, 2 is unlikely, 3 is possible, 4 is likely and 5 is certain. To calculate the total risk, we multiply these numbers.

Example:

Marginal impact (2) * likely to happen (4) = total risk (8)

C.1.2.2 Risk matrix

After we have calculated the total risk, we compare the result to the risk matrix to see what rating the risk get. The rating determine how big threat the risk pose to the project and how much resources we must spend for that risk. The risk matrix is divided into five levels and each level is designated a colour; green, light green, yellow, orange and red.

Impact	**Probability**				
	Very unlikely(1)	**Unlikely(2)**	**Possible(3)**	**Likely(4)**	**Certain(5)**
Negligible(1)	**1**	**2**	**3**	**4**	**5**
Marginal(2)	**2**	**4**	**6**	**8**	**10**
Moderate(3)	**3**	**6**	**9**	**12**	**15**
Major(4)	**4**	**8**	**12**	**16**	**20**
Catastrophic(5)	**5**	**10**	**15**	**20**	**25**

Table 2: Risk Matrix

C.1.2.2.1 Green

The green area represent risks that constitute a very small threat to our project. Those risks which end up here do not need much consideration and are almost ignored. They could have been removed from the list, but we keep them in the document to control that we have indeed analysed the risk.

C.1.2.2.2 Light green

These risks need more consideration than the green ones, but they do not constitute a significant threat. They are either very unlikely to happen or the impact is low. We do however follow up the risk to avoid materialization.

C.1.2.2.3 Yellow

There is a fairly big step from green to yellow. These risks pose a significant threat to the project. Closely monitoring the risks is necessary throughout the project timeline.

C.1.2.2.4 Orange

Orange risks pose a major threat for the project. These risks need constant update and supervision to be ahead of the problem.

C.1.2.2.5 Red

These risks are critical for the project and need to be sorted out immediately.

C.1.2.3 Develop mitigation strategy

Mitigation strategies are designed to reduce the probability that a risk will materialize. The objective is to find an easy way to circumvent the problem. If the mitigation option is too costly in time or money, we will have to consider another alternative.

By developing, updating and using this document, we mitigate the risks associated with this project.

C.1.2.4 Develop contingency plan

A part of the risk mitigation is to develop a contingency plan. Contingency plans are designed to reduce the impact if a risk does materialize. When a problem occurs, we need an alternative path to follow so the project does not stagnate.

An example to a contingency plan, is to use old parts/tools if we do not receive the new ones we have ordered in time for a deadline.

C.1.2.5 Effective risk

When we have developed mitigation strategies and contingency plans for the risks, we analyse the effectiveness to see how much the probability and impact is reduced. We then use this new data to calculate the effective risk and determine how much of a threat this risk actually pose.

C.1.2.6 Risk template

To get a simple overview of the risks, we divide them into different areas and put them in separate tables.

Assessment	**Impact:**	**Probability:**	**Total risk:**	**Effective risk:**
Risk:	"What is the risk?"			
Mitigation strategy:	"What can we do to prevent the risk from happening?"			
Contingency plan:	"What is the strategy when the risk does materialize?"			

Table 3: Risk template

C.1.3 Areas of risk

C.1.3.1 Completion of the project

C.1.3.1.1 Risk management

Assessment	Impact: 5	Probability: 1	Total risk: 5	Effective risk: 5
Risk:	If one or more of the risks materializes and the contingency plan is not set to work, we may not be able to complete our project.			
Mitigation strategy:	Monitor risks and update the risk document continuously throughout the project timeline.			
Contingency plan:	If we do not complete our project, we may have to come back next year to do another project.			

C.1.3.1.2 Limitation of the project

Assessment	Impact: 4	Probability: 3	Total risk: 12	Effective risk: 4
Risk:	The total scope of what we are developing is very wide. There are few limitations set by either Mektron or Engmark Meteor. This forces us to limit the project ourselves. If the project becomes too big, we may not be able to finish it in time. This may result in a bad evaluation.			
Mitigation strategy:	To be on the safe side, we focus on the minimum requirements to get the system up and running to have something to present and deliver to Mektron and Engmark Meteor.			
Contingency plan:	The documentation for the project will also have to be in focus to show the examiners what we have been working on. It is possible that we have a solution to a problem, but this is worthless if it is just in our heads. The documentation may save us if we fail to complete the construction phase.			

C.1.3.2 Vision Development

C.1.3.2.1 Not due in time

Assessment	Impact: 4	Probability: 3	Total risk: 12	Effective risk: 6
Risk:	The vision system is not finished on time. This will not affect us that much as much it will affect Engmark Meteor and Mektron. We will not be able to run final interfacing test with the vision system and the robot, and Mektron will have an incomplete system to deliver to Engmark Meteor. Note: The vision system is able to perform the tasks we need it to (locate holes and orientation of the griddle). The system needs to be calibrated in relation to the mounting bracket and screwdriver. Status: not complete Date: 18.05.2015			
Mitigation strategy:	Put a lot of time and resources into making it work as soon as possible. Write detailed documentation so we can show what we have accomplished when the time is up.			
Contingency plan:	Transfer responsibility to another company by borrowing or buying necessary equipment (hardware, software).			

C.1.3.2.2 Not being able to recognize the components to be placed

Assessment	Impact: 1	Probability: 4	Total risk: 4	Effective risk: 2
Risk:	We cannot get the vision system to identify the different components to be placed. E-REG, E-CC, M-BP Note. The vision system is able to recognize different objects. To get the vision system to recognize the specific components, the dimensions of these components need to be added to the software. Status: not implemented Date: 18.05.2015			
Mitigation strategy:	Push the group members responsible for this and get it done.			
Contingency plan:	To identify the components with vision is not absolutely necessary, but it is a handy feature. We can put the components at pre-defined locations so that the system knows where to pick them up from. This requires more preparations from the workers side when using the assembly system. We can facilitate the system so an image recognition program can be implemented at a later point.			

C.1.3.2.3 Not being able to find the “place” position for the components

Assessment	Impact: 3	Probability: 3	Total risk: 9	Effective risk: 9
Risk:	The vision system cannot find and give the correct coordinates for the position and orientation to where the components are to be placed.			
Mitigation strategy:	Make it easier for the vision system to recognize key areas and search in the right place for the right position.			
Contingency plan:	The assembler will have to place the parts themselves. Depending on how far we get with the screw fastening development, the assembler may or may not also fasten the components themselves as before.			

C.1.3.3 Loss of data

Assessment	Impact: 5	Probability: 1	Total risk: 5	Effective risk: 2
Risk:	Lose part of or all our documentation due to technical reasons.			
Mitigation strategy:	Backup Dropbox periodically. Last backup 14.05.2015.			
Contingency plan:	Depending on the amount lost and the time left, we must consider if completion of the project is feasible. We must rely on the evaluation of the presentations and the quality of the finished product.			

C.1.3.4 Interfacing the vision system with UR-system

Assessment	Impact: 4	Probability: 2	Total risk: 8	Effective risk: 4
Risk:	Managing to interface vision system with the UR-system is necessary to get the system to work. None of the group members have prior knowledge of the UR-Script so there is a chance that we may not get the interfacing done in time. Controlling the UR-system is under control, but the interaction with the vision system is as of now not completed. 19.03.2015. If do not get the systems to interact, the project will not be finished.			
Mitigation strategy:	Consult with Egil and others for help with interfacing right away when an issue occurs.			
Contingency plan:	If the task is too difficult to complete, we have to outsource the task to a third party. A safety net for the group's part is to rely on the documentation to get a good evaluation of the project. The documentation will also help others in resuming the project from where we left off.			

C.1.3.5 Screwdriver

C.1.3.5.1 Not being able to fasten screws for the bottom plate M-BP

Assessment	Impact: 3	Probability: 2	Total risk: 6	Effective risk: 2
Risk:	The bottom plate is going to be fastened by self-drilling screws. These require sufficient speed, torque and force to penetrate the casing. When the screw has penetrated the casing, the screw has to stop when fully tightened without breaking the casing, bottom plate or itself.			
Mitigation strategy:	Do different test to adjust torque, speed and force. If none of the different combination of the three variables above works, we have to consider using other types of screws.			
Contingency plan:	The assemblers will have to continue doing this process themselves without the assistance from the automation system.			

C.1.3.5.2 Not being able to fasten the screws for the regulator E-REG

Assessment	Impact: 3	Probability: 2	Total risk: 6	Effective risk: 2
Risk:	The screws for the regulator is flat at the bottom and they are to be fastened horizontally. These screws need to be placed precise to hit the threads. The force applied has to be fairly low since the regulator is being held by another robot. The torque has to be limited so that the threads don't bread when the screw is fully tightened. The torque has to be sufficient to fasten the screws.			
Mitigation strategy:	Do different tests to adjust torque and force.			
Contingency plan:	The assemblers will have to continue doing this process themselves without the assistance from the automation system.			

C.1.4 Eliminated/handled/occurred risks

C.1.4.1 Vision development

C.1.4.1.1 Risk: Not receiving necessary equipment in time.

We have received the Kinect and Intel RealSense.

Date: 13.02.2015

We have borrowed two more cameras that are more suited for industrial use. These have been tested and they may replace the Kinect and the Intel RealSense.

Date: 12.03.2015

C.1.4.1.2 Risk: Not being able to locate holes

We have managed to locate holes with the Kinect.

11.03.2015

We have managed to locate holes with the IFM sensor.

13.03.2015

C.1.4.1.3 Risk: Necessary equipment does not work properly

The cameras we have used have worked as they should without any trouble.

C.1.4.1.4 Risk: Necessary equipment gets broken

All the equipment needed for vision development has stayed intact for the duration of the project:

Date: 18.05.2015

C.1.4.2 UR-10

C.1.4.2.1 Risk: Not receiving it in time.

We have received the UR-10.

Date: 20.02.2015.

C.1.4.2.2 Risk: Not managing to control the UR-10 with UR-Script

We have control on the critical aspects of UR-Script that we will make use of ourselves in this project.

Date: 16.03.2015

C.1.4.3 Gripper

C.1.4.3.1 Risk: Lack of gripper testing due to late delivery.

Henning from Festo has been so kind to lend us a gripper that is almost identical to the one that we had in mind to order. This has made it possible to run minor tests with the gripper relatively early in the project.

We received the gripper for testing 09.03.2015.

We received the actual gripper to be used 08.05.2015.

C.1.4.3.2 Risk: Ordering a gripper that turns out to not fit our purpose

The group has conducted thorough research to make sure that we ordered the correct gripper.

C.1.4.4 Waiver of group members

C.1.4.4.1 Risk: Sickness

Some of our group members have been sick, but only for a short period of time. Sickness has not affected the project.

Date: 12.05.2015.

C.1.4.4.2 Risk: Group member has to withdraw from the project

None of the group members had to withdraw from the project. Everyone stayed to see it through to the end.

The uncertainty about the reevaluation of a failed exam for one of the group members was not completed within the project timeline, which let him stay in the group for the duration of the project.

Date: 18.05.2015

C.1.4.5 Screwdriver

C.1.4.5.1 Risk: Not being able to pick up screws

We managed to pick up screws with the screwdriver attached to the robotic arm.

13.04.2015

C.1.4.5.2 Risk: Not being able to hold screw when moving around

The bit and magnetic bit holder makes the screw stay in place when the robotic arm is moving the screwdriver around.

13.04.2015

C.1.4.5.3 Risk: Not ordering and receiving new screwdriver in time to test properly before delivering final report.

We contacted the screwdriver supplier from Sweden some time before Easter. He got back to us May 7th and reported that his delivery time on screwdrivers is 4 weeks. It is less than 4 weeks to the end of our project so we will not be able to run tests with the final screwdriver in time to May 19th.

Håvid Engmark from Engmark Meteor sent us a temporarily screwdriver to use for testing 12.05.2015.
Received 13.05.2015.

C.1.4.6 Economy

C.1.4.6.1 Risk: Insufficient resources

Everything we have needed to complete the project have been taken care of.

Last update: 12.05.2015

C.1.4.7 Griddle

C.1.4.7.1 Risk: New design due to new regulations

There have been no new regulations that have forced Engmark Meteor to change the design of the griddle due to new regulations during the project timeline.

C.2 Workstation Risk analysis

Abstract

This document addresses the risks of using the automated assembly system. Both the current design and the initial design. It starts with the risk assessment for the first workstation design and operation idea and the safety measures for this idea. These safety measures led to a new design of the workstation and a new risk assessment was necessary. From this, there were made more safety measures that led to the current design of the workstation.

Contents

Abstract 140

Contents 140

Tables 140

C.2.1 Introduction 141

C.2.1.1 Hazard analysis 141

C.2.1.2 Safety measures 141

C.2.1.3 Further assessment 141

C.2.2 First risk assessment 142

C.2.2.1 Screwdriver drilling through hand. 142

C.2.2.2 Robot hitting assembler during assembly. 143

C.2.2.3 Robot hitting personnel refilling components and moving griddles around. 143

C.2.2.4 Screwdriver accidently stabbing assembler in the eye during assembly. 143

C.2.2.5 Safety measures for first workstation idea summary 144

C.2.2.6 Implementation of the safety measures 144

C.2.3 Second risk assessment 145

C.2.3.1 Robots hitting personnel refilling components and moving griddles around. 146

C.2.3.2 Severed fingers (from turntable) 146

C.2.3.3 Squeezing hands or fingers in the contraption for the casing when placing a griddle. 147

C.2.3.4 Safety measures for second risk assessment summary 147

C.2.3.5 Implementation of the safety measures 147

C.2.4 References 147

Tables

Table 4: Modified hazard factor rating table from Mektron AS 141
Table 5: First risk assessment hazards overview 142
Table 6: Second risk assessment hazards overview 145

C.2.1 Introduction

When operating the workstation, there are possible hazards that the personnel may be subjected to if they are unaware, using the station improperly or if they system itself has errors that accidently hurt the assembler at work. To reduce the possibility of anyone getting hurt while operating the workstation, we have identified possible hazards and made safety measures.

C.2.1.1 Hazard analysis

We use a modified version of a table given by Mektron AS to rate different factors of hazards. The table works as a reference for our analysis and a grade is given to each factor from this. When analyzing the hazards we are considering three factors; severity, probability and avoidance. The severity factor depends on the most likely consequence of a hazard, the probability is how likely it is to happen and avoidance is how likely it is to avoid the hazards from occurring.

Consequences	**Severity**	**Probability**		**Avoidance**	
Death or permanent harm(loss of arm, eye etc.)	4	Certain	5	Impossible	5
Permanent harm(loss of fingers etc.)	3	Likely	4		
Reversible harm, professional medical attention needed	2	Possible	3	Possible	3
Reversible harm, professional medical attention not needed	1	Unlikely	2		
		Very unlikely	1	Likely	1

Table 4: Modified hazard factor rating table from Mektron AS

The analysis is solely based on assumptions of how and what may happen. We have pictured different possible scenarios and evaluated the different factors against the rating table.

C.2.1.2 Safety measures

When the hazards are identified and analysed, we prepare options for safety measures. A safety measure is either hardware modification, software modification or safety instructions. We then discuss what possible ways there are to reduce or eliminate, and then evaluate if it is practical or not to implement the solution. If it takes lot of resources or if the effectiveness of the workstation is affected for a small increase in safety, we have to consider other solutions with better cost-benefit ratio.

It is not necessary to implement every safety measure option to reduce or eliminate a hazard. If one option has the same effect as another, we select the one that provides the best outcome.

C.2.1.3 Further assessment

To get a CE approval for the workstation, the system has to be built in accordance to harmonized standards. The most relevant are the machine directive, low voltage directive and EMC directive.

Directive 2004/108/EC relating to electromagnetic compatibility

Directive 2006/42/EC on machinery

Directive 2006/95/EC Low Voltage Directive (LVD)

This document is governing the risk assessment.

C.2.2 First risk assessment

We used our initial workstation design and operation idea as a base for the hazard analysis. This version has the griddle stationary at a table and the robot standing next to it on the table. A robot vision system assists the robot to locate where it is to fasten the screws. The assembler places the parts and the robot moves in and fasten the parts with a screwdriver. The assembler and robot work as a team close together. From this, we identified the hazards listed in the table below.

No.	**Hazard**	**Severity**	**Probability**	**Avoidance**	**Safety measure**	**Safe yes/no**
1	Screwdriver drilling through hand.	3	3	1	Workstation design. Human detection by the vision system. Safety instructions.	no
2	Robot hitting assembler during assembly.	2	3	3	Workstation design. Human detection by the vision system. Lower speed to reduce harm.	no
3	Robot hitting personnel refilling components and moving griddles around.	2	2	1	Safety instructions. Restrict area around the robot. Limit the area of operation for the robot, software.	yes
4	Screwdriver accidently stabbing assembler in the eye during assembly.	4	2	1	Workstation design. Human detection by the vision system. Safety instructions.	no

Table 5: First risk assessment hazards overview

C.2.2.1 Screwdriver drilling through hand.

Assessment	**Severity: 3**	**Probability: 3**	**Avoidance: 1**
Risk:	The cooperation of placing and fastening parts could possibly lead to horrible accidents if the assembler is inattentive just for one moment. After the assembler has placed a part, the robot places the screw with the screwdriver and goes on until the screw is fastened. If the assembler accidently has put a hand where the screw is supposed to be fastened, the hand will be attached to the griddle. This will cause permanent harm and require the need of professional medical attention.		
Safety measure:	#1 Use skeleton detection with the Microsoft Kinect V2 to detect human parts in the field of operation and then to avoid hurting the assembler. #2 Design the workstation in a way that forces the assemblers to have a distance from the robot while operating it. This way the assemblers will be out of reach for the robot. #3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action.		

C.2.2.2 Robot hitting assembler during assembly.

Assessment	**Severity: 2**	**Probability: 3**	**Avoidance: 3**
Risk:	Because of the tight interaction between the assembler and robot, there is a change that they bump into each other. The robot has integrated safety functions that make it stop when hitting something or someone, but injuries can still occur. The robot is not moving very fast so an injury may in the worst-case lead to the need of professional medical attention.		
Safety measure:	#1 Use skeleton detection with the Microsoft Kinect V2 to detect human parts in the field of operation and then to avoid hurting the assembler. #2 Design the workstation in a way that forces the assemblers to have a distance from the robot while operating it. This way the assemblers will be out of reach for the robot. #4 Have the robot operating in a safe mode where the speed is low but effective. If the robot should hit the assembler, the harm will be limited.		

C.2.2.3 Robot hitting personnel refilling components and moving griddles around.

Assessment	**Severity: 2**	**Probability: 2**	**Avoidance: 1**
Risk:	The robot may hit a person if the person is too close to it when the robot is in action. The robot has integrated safety functions that make it stop when hitting something or someone, but injuries can still occur. The robot is not moving very fast so an injury may in the worst-case lead to the need of professional medical attention (severity grade 2). NOTE! If a person who is not operating the robot have his or her face too close to the robot when the robot is in action, this person may be seriously injured and the loss of teeth or an eye may happen (severity grade 4).		
Safety measure:	#3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action. #5 Restrict the area around the workstation so no one enter this except for the assembler that operates the system. #6 Limit the area of operation for the robot, software.		

C.2.2.4 Screwdriver accidently stabbing assembler in the eye during assembly.

Assessment	**Severity: 4**	**Probability: 2**	**Avoidance: 1**
Risk:	Because of the tight interaction between the assembler and robot, there is a change that they bump into each other. If the assembler is having his or her face too close to the end effector and the robot does something unexpected, there is a worst-case scenario possibility of the assembler being stabbed in the eye.		
Safety measure:	#1 Use skeleton detection with the Microsoft Kinect V2 to detect human parts in the field of operation and then to avoid hurting the assembler. #2 Design the workstation in a way that forces the assemblers to have a distance from the robot while operating it. This way the assemblers will be out of reach for the robot. #3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action.		

C.2.2.5 Safety measures for first workstation idea summary

#1 Use skeleton detection with the Microsoft Kinect V2 to detect human parts in the field of operation and then to avoid hurting the assembler.

#2 Design the workstation in a way that forces the assemblers to have a distance from the robot while operating it. This way the assemblers will be out of reach for the robot.

#3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action.

#4 Have the robot operating in a safe mode where the speed is low but effective. If the robot should hit the assembler, the harm will be limited.

#5 Restrict the area around the workstation so no one enter this except for the assembler that operates the system.

#6 Limit the area of operation for the robot, software.

C.2.2.6 Implementation of the safety measures

We went away from using the Microsoft Kinect V2 in our system. This ruled out safety measure #1.

For the new design of our workstation, we implement safety measure #2 by introducing a turntable that keep the operation area for the robot and assembler separated. There is also an extra robotic arm. This robot has a short range of operation and is out of reach of the assembler. This robot will be holding the screwdriver resulting in elimination of hazard no.1, unless someone deliberately stick their hand in the way of the screwdriver. By implementing safety measure #6 and limiting the area of operation for the robots, we eliminated hazards no.2 and no.4 as well.

C.2.3 Second risk assessment

The new workstation design and operation is more robot dominant and it is safer for the assembler in relation to the robots. A turntable that is more than two lengths of a griddle separates the robot and assembler. This way, the assembler will be out of reach for the robot with the screwdriver. The robot and assembler will be working on separate griddles in parallel so the two will not get in the way of each other.

The robot with the gripper will get a griddle from a pallet and place it in a contraption on the turntable. The robots will then pick, place and fasten the cable connector and regulator. The griddle moves on to the assembler, who will attach the power inlet and connect different cables. The griddle then goes back to the robots, which will fasten the bottom plate and move the griddle over to a pallet for finished griddles. The assembler will do the turning of the table by hand.

The new design has made it safe for the assembler in relation to the robot, but we have introduced two new mechanism with moving parts that need a safety evaluation.

No.	**Hazard**	**Severity**	**Probability**	**Avoidance**	**Safety measure**	**Safe yes/no**
1	Screwdriver drilling through hand.	3	1	1	#2	yes
2	Robot hitting assembler during assembly.	2	1	1	#2, #6	yes
3	Robots hitting personnel refilling components and moving griddles around.	2	2	1	Safety instructions. Restrict area around the robot.	yes
4	Screwdriver accidently stabbing assembler in the eye during assembly.	4	1	1	#2, #6	yes
5	Severed fingers (from turntable)	3	2	1	Workstation design.	
6	Squeezing hands or fingers in the contraption for the casing (M-CA) when placing a griddle.	1	2	1	Safety instructions. The robot pick and place the griddle in the mechanism.	yes

Table 6: Second risk assessment hazards overview

C.2.3.1 Robots hitting personnel refilling components and moving griddles around.

Assessment	Severity: 2	Probability: 2	Avoidance: 1
Risk:	From the first design, the area of operation of the robots is expanded to be outside the limits of the table. This will not affect the assembler, but personnel staying too close to the robot will have a greater chance of being hit. The operation of the robots performed outside the limits of the table is moving griddles from a pallet to the table and from the table to another pallet. The small griddles weigh about 6,5kg. These will have sufficient momentum to knock out teeth or knock someone unconscious. The robot has safety functions that make it stop when hitting something, but this initiates after it has hit something. Then the damage is already done. Being hit by a griddle will not cause permanent damage. Teeth are fixable. This hazard is therefore considered as a severity of grade 2.		
Safety measure:	#3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action. #5 Restrict the area around the workstation so no one enter this except for the assembler that operates the system.		

C.2.3.2 Severed fingers (from turntable)

Assessment	Severity: 3	Probability: 2	Avoidance: 1
Risk:	The turntable poses a threat to the assembler's fingers if not operated correctly. The turntable will have a great momentum when turning because of the weight of the griddles on it. If an assembler becomes eager when operating the turntable and is unaware of where the hands are placed, fingers may be stuck between a moving part and a stationary part resulting in broken or severed fingers.		
Safety measure:	#3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action. #7 Design the turntable to have no edges or holes that hands or fingers can fit in that will act as scissors when the table is turning. #8 Add covers that protects the assembler from getting limbs in between moving parts.		

C.2.3.3 Squeezing hands or fingers in the contraption for the casing when placing a griddle.

Assessment	Severity: 1	Probability: 2	Avoidance: 1
Risk:	The plan is for the robot to pick and place griddles in a contraption that uses the weight of the griddle to make the casing more circular for easier mounting of the bottom plate. This contraption has four arms that act with forces that may hurt limbs that are pinched in between the arms and the casing. If there is a malfunction in the system and the assembler has to move the griddle by hand in or out of the contraption, he or she may be unfortunate to get their hands or fingers in between the casing and the arms of the contraption. This hazard is considered to have low consequence severity. It is easy to avoid the hazard from occurring and the probability of having to manually move the griddle in and out of the contraption is fairly low when the system is up and running. From this analysis, we have concluded that there is no need for safety measures for this particular hazard.		
Safety measure:	None needed.		

C.2.3.4 Safety measures for second risk assessment summary

#3 Instruct the personnel to follow a safe way to operate and act around the workstation when system is in action.

#5 Restrict the area around the workstation so no one enter this except for the assembler that operates the system.

#7 Design the turntable to have no edges or holes that hands or fingers can fit in that act as scissors when the table is turning.

#8 Add covers that protects the assembler from getting limbs in between moving parts.

C.2.3.5 Implementation of the safety measures

Safety measure #3 was implemented by making safety instructions [1], [2] reducing the probability of being hit by the robot, hazard no.3. Safety measure #5 is not necessary to implement if the personnel follows the safety instructions.

Safety measure #7 was implemented by making the turntable circular [3]. Since the table is chosen to be circular, there will be no moving edges that may overlap when the table is turning. This will eliminate the chance of cutting off fingers.

C.2.4 References

1. *Safety Instructions English*
2. *Safety Instructions Norwegian*
3. *Concept document robot and workstation*

C.3 Process analysis

Abstract

The first part of our task was to analyse Engmark meteors production line to identify potential areas of improvement. Our automation solution infers some changes to that production line and this document details them.

Contents

Abstract 148

Contents 148

Figures 148

C.3.1 Original assembly process 149

C.3.2 Modified assembly process 151

Figures

Figure 28: Original assembly process 149
Figure 29: Modified assembly process 151

C.3.1 Original assembly process

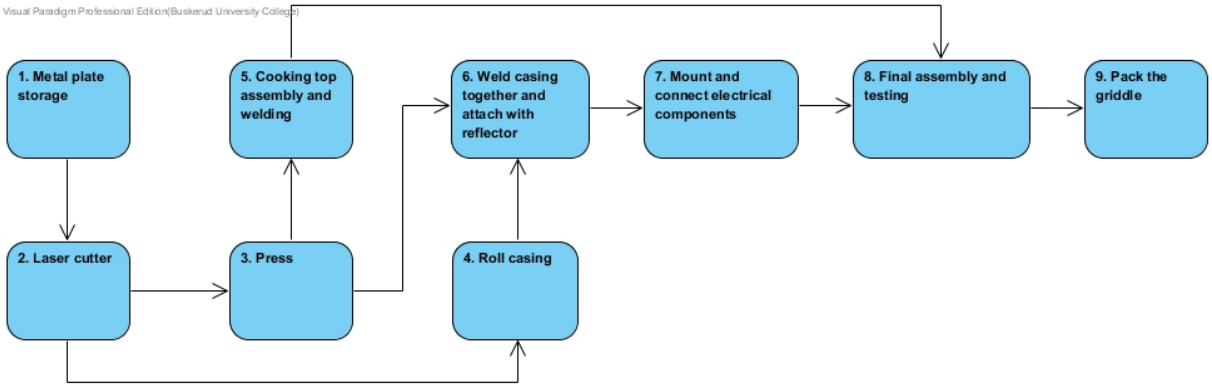

Visual Paradigm Professional Edition(Buskerud University College)
1. Metal plate storage
5. Cooking top assembly and welding
6. Weld casing together and attach with reflector
7. Mount and connect electrical components
8. Final assembly and testing
9. Pack the griddle
2. Laser cutter
3. Press
4. Roll casing

Figure 28: Original assembly process

1. Steel plate storage/feeding
   This space is for storing the metal sheets used for the griddle.

2. Laser cutter
   All metal components are automatically cut and palletized here, though the sheets are inserted to the laser cutter manually.

3. Press
   The cooking top, heat sink, reflector and bottom plate are pressed into shape here. The process is automated, using the pallets from the previous station.

4. Roll Casing
   The casing is rolled into shape. Feeding the cut parts and palletizing the rolled casings is done manually.

5. Assemble and weld cooking top
   a. Thread heating element wire through the heat sink
   b. Place the heating element on the heat sink
   c. Place cooking top on heat sink
   d. Move to automated spot welding machine
   e. (While the machine works, the operator prepares a new top)
   f. Place and weld supporting bracket(s) on the top

6. Weld casing together and attach with reflector
   a. Weld casing together to make a solid ring
   b. Weld power bracket on casing
   c. Switch spot welding electrodes (this operation takes about two minutes so a batch of casings are welded before doing so)

   d. Weld reflector to casing

7. Mount and connect electrical components
   a. Mount insulation rings to reflector
   b. Power inlet is mounted and fastened
   c. Regulator is mounted and fastened
   d. Cable connector is mounted and fastened
   e. The electrical wires between the power inlet, the regulator and the cable connector are fastened
   f. The regulator knob is mounted

8. Final assembly and testing
   a. Thread heating element wires though the reflector
   b. Fasten the cooking top to the reflector
   c. Tighten and fasten the heating element wires to the cable connector
   d. Mount and fasten the bottom plate
   e. Testing

9. Pack the griddle
   a. Assemble storage box
   b. Wash with methylated spirit
   c. Apply oil to cooking top
   d. Wrap the cooking top in plastic
   e. Put the griddle in the box
   f. Put a manual in the box
   g. Staple the box shut
   h. Palletize the box

C.3.2 Modified assembly process

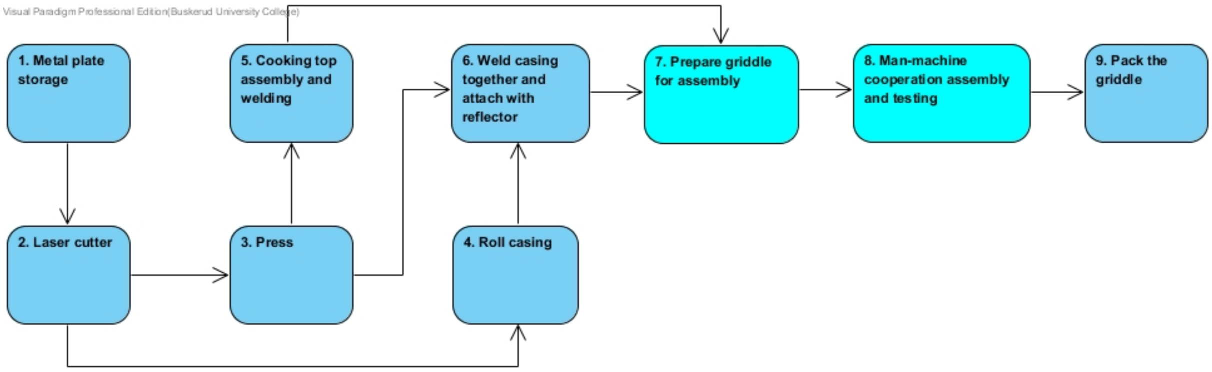

Visual Paradigm Professional Edition(Buskerud University College)
1. Metal plate storage
5. Cooking top assembly and welding
6. Weld casing together and attach with reflector
7. Prepare griddle for assembly
8. Man-machine cooperation assembly and testing
9. Pack the griddle
2. Laser cutter
3. Press
4. Roll casing

Figure 29: Modified assembly process

The implementation of the automation system will infer changes to changes to station number no. 7 and a complete redesign of station no. 8.

Note that the output from station no. 5 now goes to station no. 7 instead of no. 8.

7. Prepare griddle for assembly

a. Mount insulation rings to reflector
b. Thread heating element wires though the reflector
c. Fasten the cooking top to the reflector

8. Man-machine cooperation on assembly and testing

(AS: Assembly system, OP: Operator)

a. AS: Move the griddle from buffer to assembly position
b. AS: Place and fasten the cable connector
c. AS: Place and fasten the regulator
d. AS: Move griddle from assembly position to operator position
e. OP: Mount and fasten the power inlet
f. OP: The electrical wires between the power inlet, the regulator and the cable connector are fastened
g. OP: Tighten and fasten the heating element wires to the cable connector
h. AS: Move from operator position to assembly position
i. AS: Place and mount bottom plate
j. AS: Palletize griddle

Note that the assembly system and the operator work in parallel with each their griddle and switch between them.

C.4 Cost-Benefit Analysis

Abstract

This document gives an overview of the factors to take account for to make a cost benefit analysis of Engmark meteors automation project and the estimated Return on the investment (ROI).

Contents

Abstract **152**

Contents **152**

C.4.1 Costs **153**

C.4.2 Benefits **153**

C.4.3 Details **154**

C.4.4 References **154**

C.4.1 Costs

- Initial
    - UR10 Purchase

- During the project
    - UR10 not in use
    - Time spent on the project by costumer (Håvid)
    - End effectors (screwdriver, gripper)

- Project implementation
    - UR5
    - Time spent training operators
    - Production line adjustments
    - Workstation
        - Screw feeder
        - Construction/alterations
        - Air-, vacuum- and electrical supply

- Long term
    - Maintenance of the system

All costs are monetary, which is a good thing; we do not want this project to have a negative impact on quality, nor on welfare.

C.4.2 Benefits

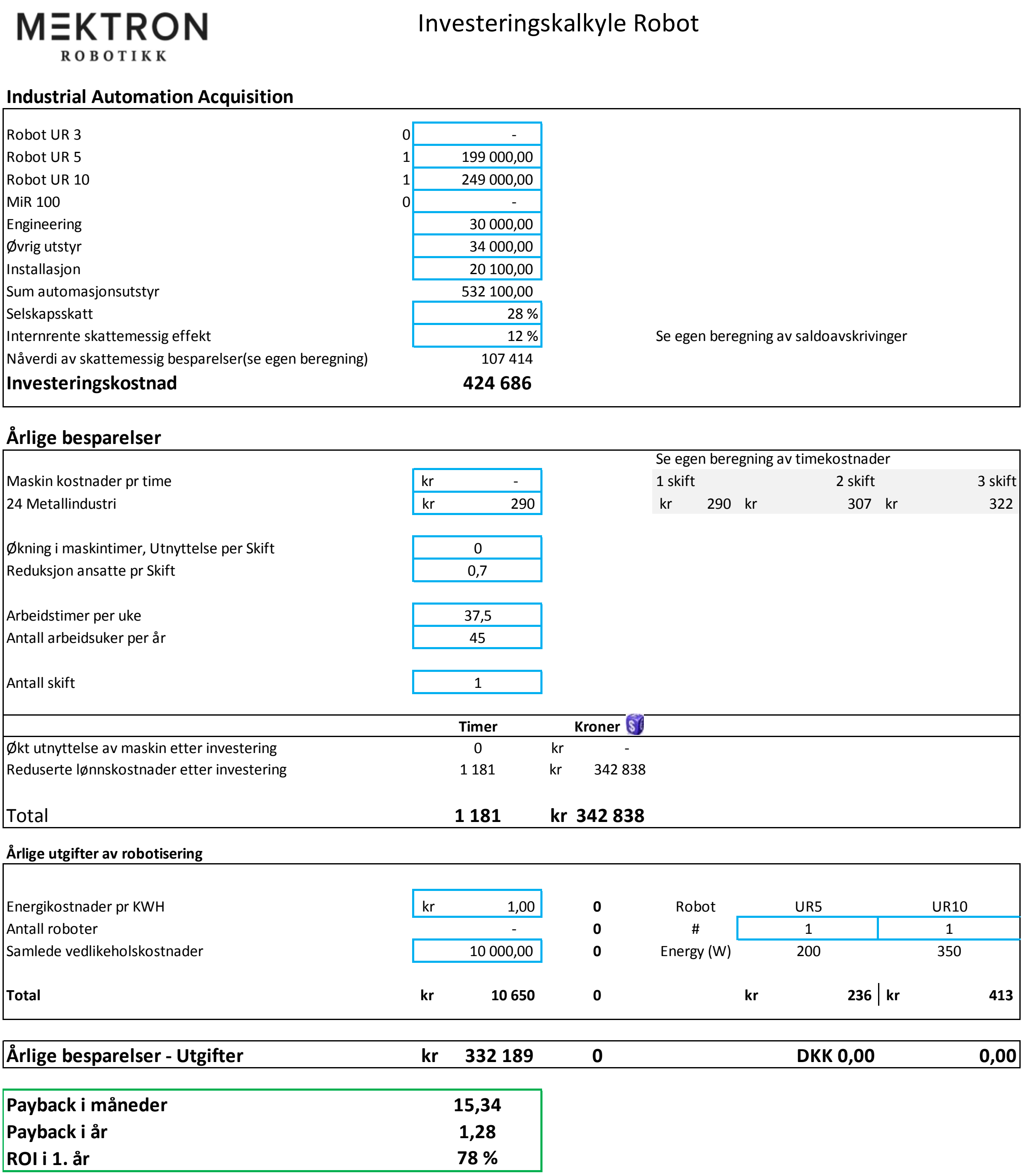
- Less wages paid, system takes over worker tasks – M
- Less training time for new employees for manual tasks - M

- Higher worker motivation – M/W
    - System assists operator to achieve production goals - M
    - Take over dull tasks – M/W
    - Take over potentially damaging (heavy/repetitive) tasks – M/W

- Production line streamlining – M/Q
    - Increased stability/foreseeability of production output (X griddles/day) - M
    - Increased stability/foreseeability of production quality (little to no variation) – Q

M – Monetary
W – Welfare
Q – Quality

C.4.3 Details



MEKTRON ROBOTIKK

Investeringskalkyle Robot

Industrial Automation Acquisition

Robot UR 3	0	-	
Robot UR 5	1	199 000,00	
Robot UR 10	1	249 000,00	
MiR 100	0	-	
Engineering		30 000,00	
Øvrig utstyr		34 000,00	
Installasjon		20 100,00	
Sum automasjonsutstyr		532 100,00	
Selskapsskatt		28 %	
Internrente skattemessig effekt		12 %	Se egen beregning av saldoavskrivinger
Nåverdi av skattemessig besparelser(se egen beregning)		107 414	
Investeringskostnad		**424 686**	

Årlige besparelser

Se egen beregning av timekostnader

Maskin kostnader pr time	kr	-		1 skift		2 skift		3 skift
24 Metallindustri	kr	290	kr	290	kr	307	kr	322
Økning i maskintimer, Utnyttelse per Skift		0						
Reduksjon ansatte pr Skift		0,7						
Arbeidstimer per uke		37,5						
Antall arbeidsuker per år		45						
Antall skift		1						

	Timer		Kroner
Økt utnyttelse av maskin etter investering	0	kr	-
Reduserte lønnskostnader etter investering	1 181	kr	342 838
Total	**1 181**	**kr**	**342 838**

Årlige utgifter av robotisering

Energikostnader pr KWH	kr	1,00	**0**	Robot		UR5		UR10
Antall roboter		-	**0**	#		1		1
Samlede vedlikeholskostnader		10 000,00	**0**	Energy (W)		200		350
Total	**kr**	**10 650**	**0**		**kr**	**236**	**kr**	**413**

Årlige besparelser - Utgifter	**kr**	**332 189**	**0**	**DKK 0,00**	**0,00**

Payback i måneder	**15,34**
Payback i år	**1,28**
ROI i 1. år	**78 %**

C.4.4 References

1. Mindtools.com, Cost-Benefit Analysis [30.03.2015] http://www.mindtools.com/pages/article/newTED_08.htm
2. Wikipedia, Cost-Benefit Analysis [30.03.2015] http://en.wikipedia.org/wiki/Cost%E2%80%93benefit_analysis

C.5 Assembly process alternatives

Abstract

This document describes the different alternatives in solving the Vi'RA automation project showing that the two-arm solution will give the greatest benefit by a wide margin, even when accounting for it having the highest cost.

Contents

Abstract 155

Contents 155

C.5.1 Alternatives 156

C.5.2 Alternative 1 157

C.5.3 Alternative 2 157

C.5.4 Alternative 3 158

C.5.5 Alternative 4 159

C.5.1 Alternatives

Alternative 1, Attachment of components	**Alternative 2**, Attachment of components and picking of the bottom plate	**Alternative 3**, Attachment of components and picking of multiple components	**Alternative 4**, Attachment of components and picking of multiple components(with the use of two robots)
Requires: • UR10 • Screwdriver • Camera • Manual Regulator Jig	Requires: • UR10 • Screwdriver • Camera • Gripper • Vacuum tool • Manual Regulator Jig	Requires: • UR10 • Screwdriver • Camera • Gripper • Vacuum tool • Regulator Jig for the gripper	Requires: • 2xUR • Screwdriver • Camera • Gripper • Vacuum tool
Solves: • Attachment of the reflector • Attachment of the cable connector • Attachment of the regulator • Attachment of the bottom plate	Solves: • Attachment of the reflector • Attachment of the cable connector • Attachment of the regulator • Placement and attachment of the bottom plate	Solves: • Attachment of the reflector • Placement and attachment of the cable connector • Placement and attachment of the regulator • Placement and attachment of the bottom plate	Solves: • Attachment of the reflector • Placement and attachment of the cable connector • Placement and attachment of the regulator • Placement and attachment of the bottom plate • Movement of the fully assembled griddle
Pros: • Relatively easy to realize	Pros: • Performs more automation tasks	Pros: • Performs a lot more automation tasks	Pros: • Efficient • Performs a lot more automation tasks • Not dependent on a regulator jig
Cons: • Not a lot of the assembly tasks are automated	Cons: • Not very efficient • Not a lot of the assembly tasks are automated	Cons: • Not very efficient • Uncertainty regarding the regulator jig	Cons: • Costs more • Little time for implementation, therefore the result will primarily be theoretical

C.5.2 Alternative 1

Work distribution:

- An operator places a griddle without attaching the reflector.
- The operator places the cable connector
- The operator puts the regulator jig on the regulator and places it on the casing.
- The operator turns the turntable
- The robot attaches the two components and the reflector
- The operator turns the turntable
- The operator removes the regulator jig
- The operator places and attaches the power inlet
- The operator connects the wires
- The operator places the bottom plate
- The operator turns the turntable
- The robot attaches the bottom plate
- The operator turns the turntable
- The operator removes the griddle from the turntable

Thoughts regarding the alternative: This alternative is almost completed, however it only solves a small part of the assembly line at Håvid and it is not very effective.

C.5.3 Alternative 2

Work distribution version 1:

- An operator places a griddle without attaching the reflector.
- The operator places the cable connector
- The operator puts the regulator jig on the regulator and places it on the casing.
- The operator turns the turntable
- The robot attaches the two components and the reflector
- The operator turns the turntable
- The operator removes the regulator jig
- The operator places and attaches the power inlet
- The operator connects the wires
- The operator turns the turntable
- The robot picks up the vacuum tool and then picks up and places the bottom plate
- The robot puts away the vacuum tool
- The robot attaches the bottom plate
  Optional:
- The robot picks up the vacuum tool, removes the griddle from the turntable and puts away the vacuum tool

Word distribution version 2:
The robot places and removes the griddle to and from the turntable, but then the reflector has to be attached when it is sent to the workstation.

Thoughts regarding the alternative: This is only a small change from alt. 1, but by using a vacuum tool, we can automate more of the assembly line. However, this alternative still requires a lot of time, because the griddle has to be sent back and forth between the operator and the automation system.

C.5.4 Alternative 3

Work distribution:

- An operator places a griddle without attaching the reflector
- The operator turns the turntable
- The robot picks up and places the cable connector
- The robot puts the regulator jig on the regulator and places it on the casing
- The robot attaches the components and the reflector
- The robot removes and places the regulator jig
- The operator turns the turntable
- The operator places and attaches the power inlet
- The operator connects the wires
- The operator turns the turntable
- The robot picks up the vacuum tool and then uses this to pick up and place the bottom plate
- The robot puts away the vacuum tool
- The robot attaches the bottom plate
- The robot picks up the vacuum tool
- The robot removes the griddle from the turntable
- The robot puts away the vacuum tool

Thoughts regarding the alternative: This alternative is similar to alternative 2, but it requires a regulator jig which the robot can pick up and place. We think this can lead to errors/inaccuracy. Additionally it requires a lot of time to pick up and put down the jigs for the robot.

C.5.5 Alternative 4

Work distribution 1:

- Robot 1 has a screwdriver and a camera, Robot 2 has a gripper
- An operator places a griddle without attaching the reflector
- The operator turns the turntable
- Robot 1 attaches the reflector
- Robot 2 picks up and places the cable connector, and holds it in place
- Robot 1 attaches the cable connector, while robot 2 still holds it in place
- Robot 2 picks up and places the regulator, and holds it in place
- Robot 1 attaches the regulator, while robot 2 still holds it in place
- The operator turns the turntable
- The operator places and attaches the power inlet
- The operator connects the wires
- The operator turns the turntable
- Robot 2 picks up the vacuum tool and then uses this to pick up and place the bottom plate
- Robot 1 attaches the bottom plate
- Robot 2 removes the griddle from the turntable
- Robot 2 puts away the vacuum tool

Work distribution version 2: The reflector is attached in advance.

- Robot 1 has a screwdriver and a camera, Robot 2 has a gripper
- Robot 2 picks up the vacuum tool and then uses it to pick up a griddle
- Robot 2 places the griddle on the turntable
- Robot 2 puts away the vacuum tool
- Robot 2 picks up and places the cable connector, and holds it in place
- Robot 1 attaches the cable connector, while robot 2 still holds it in place
- Robot 2 picks up and places the regulator, and holds it in place
- Robot 1 attaches the regulator, while robot 2 still holds it in place
- The operator turns the turntable
- The operator places and attaches the power inlet
- The operator connects the wires
- The operator turns the turntables
- Robot 2 picks up the vacuum tool and then uses this to pick up and places the bottom plate
- Robot 1 attaches the bottom plate
- Robot 2 removes the griddle from the turntable
- Robot 2 puts away the vacuum tool

Thoughts regarding the alternative: Although this alternative is more expensive, we believe that it is a much better solution than all variants with one arm. This system is more consistent, since it is not dependent on the use of a regulator jig (which is not 100% reliable). Additionally, the system will perform tasks quicker, because it does not have to retrieve and place jigs, instead it can perform tasks in parallel. If we direct all our research towards this alternative, we will be able to perform most of the test necessary to complete the assignment. The tests that absent is the cooperation between the two robots, but we can test each of the robots tasks individually.

To get two robots to work synchronously will not be a major concern, once the individual tasks are solved.

D. Technology documents

D.1 Grippers

Abstract

This document contains information about why we need a gripper and the different types of grippers that we considered for this project. The conclusion is based on needs and practical ways to grasp the different items.

Contents

Abstract **161**

Contents **161**

Figures **162**

D.1.1 Introduction **164**

D.1.1.1 Why do we need a gripper? 164

D.1.1.2 Types of grippers 164

D.1.1.3 Graspable components 164

D.1.1.3.1 Cable connector E-CC 164

D.1.1.3.2 Regulator E-REG 165

D.1.1.3.2.1 Alternative idea (one-arm solution) 166

D.1.1.3.3 Power inlet E-PI 166

D.1.1.3.4 Cable shoe 166

D.1.1.3.5 Bottom plate M-BP 167

D.1.1.3.5.1 The griddle as a whole 167

D.1.1.3.6 Reflector M-REF 167

D.1.1.4 Needs 168

D.1.1.4.1 Cost 168

D.1.1.4.2 Size 168

D.1.1.4.3 Weight 168

D.1.1.4.4 Force 168

D.1.1.4.5 Load 168

D.1.1.4.6 Robustness 168
D.1.1.5 Summary 168
D.1.2 Relevant technology 169
D.1.2.1 VERSABALL 169
D.1.2.1.1 Pros 169
D.1.2.1.2 Cons 169
D.1.2.2 Parallel electric gripper 169
D.1.2.2.1 Pros 170
D.1.2.2.2 Cons 170
D.1.2.3 Parallel pneumatic gripper 170
D.1.2.3.1 Pros 170
D.1.2.3.2 Cons 171
D.1.2.4 Angled and radial pneumatic gripper 171
D.1.2.4.1 Pros 171
D.1.2.4.2 Cons 171
D.1.2.5 Vacuum gripper 171
D.1.2.5.1 Pros 172
D.1.2.5.2 Cons 172
D.1.3 Conclusion 173
D.1.3.1 Pneumatic parallel gripper 173
D.1.3.2 Vacuum gripper 173
D.1.4 References 174

Figures

Figure 30: Cable connector E-CC, drawing 164
Figure 31: Cable connector E-CC 165
Figure 32: Regulator E-REG, drawing 165
Figure 33: Regulator in place 165
Figure 34: Magnet jig for E-REG, drawing 166
Figure 35: Power inlet E-PI 166
Figure 36: Cable shoe 166
Figure 37: Bottom plate M-BP 167
Figure 38: Reflector M-REF 167
Figure 39: VERSABALL 169
Figure 40: 2-finger adaptive gripper from Robotiq 169
Figure 41: Parallel pneumatic gripper from Festo 170

Figure 42: Angled pneumatic gripper from Festo 171
Figure 43: Radial pneumatic gripper from Festo............ 171
Figure 44: Suction cups 171

D.1.1 Introduction

D.1.1.1 Why do we need a gripper?

As of today, the assemblers at Engmark Meteor both place and fasten all parts by hand. Our goal is to optimize and automate parts of the production line by implementing a robotic arm. The main task for the robotic arm is to fasten different components with screws. Instead of having the assemblers picking and placing parts to be fastened, a gripper on the robot can do this task for them. By implementing a gripper, the robot can pick, place and fasten parts by itself and the assemblers can perform other task in the meantime. Without a gripper, the assemblers will have to wait after placing a part for the robot to fasten it and then move over to the next task. Waiting is not optimal so to implement a gripper will make the process with the robot more efficient.

D.1.1.2 Types of grippers

There are four different types of grippers to choose from [5]:

- Impactive. Jaws or claws that physically grasp by direct impact upon the object.
- Ingressive. Pins or needles, which physically penetrate the surface of the object.
- Astrictive. Suction force applied to the objects surface. Magnets or vacuum.
- Contigutive. Require direct contact for adhesion to take place. Glue or freezing.

The two types relevant for this project is the impacitve type and the astrictive type. Ingressive grippers are mostly used in textile industry and contigutive grippers are too high tech for our use.

D.1.1.3 Graspable components

The four first objects listed are relatively lightweight, up to a couple hundred grams for the heaviest, and the last two are more heavyweight where the last object weighs about 5 kg. To lift the whole griddle, a gripper needs to be able to hold at least 7 kg. To manage to control every item listed within the project timeline is most unlikely, but we have considered all the parts that are possible for the automation system to handle. If we can find a gripper solution that fits all the components, it is possible to further develop the system without having to order another gripper.

Engmark Meteor wants us to focus attention to the cable connector E-CC and the regulator E-REG. If there are no cheap and simple gripper solutions on the market to handle all the different parts, the gripper to be used should be able to handle at least these two parts. Both these parts need to be held in place while being fastened.

D.1.1.3.1 Cable connector E-CC

This part is flat on every side except for the top with the screws. This gives a lot of clean surface to grasp the part and support to gripper fingers when orienting the part for placing. The top is magnetic so it is possible to use some kind of magnet to pick it up.

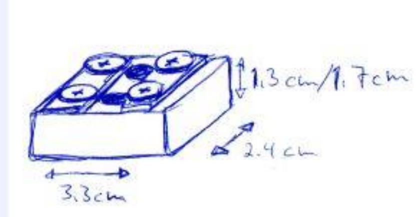


Figure 30: Cable connector E-CC, drawing

Dimensions:

- Height:
  - 1,3 cm (top ceramic material)
  - 1,7 cm (top of screw)
- Bottom:
  - 2,4 cm x 3,3 cm
- Distance between the center of the holes in the middle:
  - 1,2 cm
- Metal plate under the screws:
  - 0,8 cm x 2,1 cm

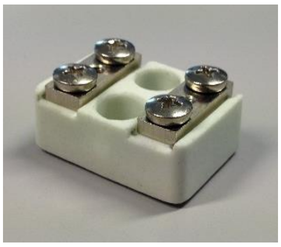


Figure 31: Cable connector E-CC

This part is to be placed flat on the power bracket [7]. It does not need to be grasped in any special way to be fit it in the right place. If we are to grip the cable connector on the outside, the gripper has to be able to open at least 2,4 cm. 3,3 cm to be sure that the gripper can grasp it from any orientation.

We considered choosing the grasping point to be inside the holes in the middle. Then the gripper only has to have an opening of about 1,0-1,2 cm. We realized quickly that this would be inconvenient because the gripper fingers would have to be tiny, so we rejected the idea.

D.1.1.3.2 Regulator E-REG

Dimensions:

- Height:
  - 4,8 cm
- Width:
  - 4,5 cm
- Front to back:
  - 2,2 cm

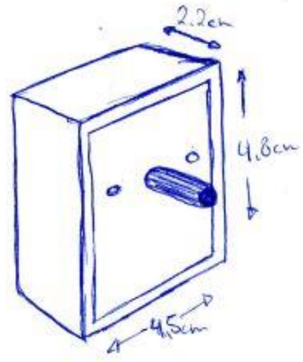


Figure 32: Regulator E-REG, drawing

This part require a minimum of 2,2cm opening. To be sure to be able to grasp it in any orientation, the gripper need an opening of over 4,8cm. This part is flat on the top, bottom and sides. At the front, it is flat except for the adjustment pin. At the back, there are many cable-connecting points that points out (see **Error! Reference source not found.**4). As for the cable connector, the regulator has a lot of lat surface that will give support for gripper fingers.

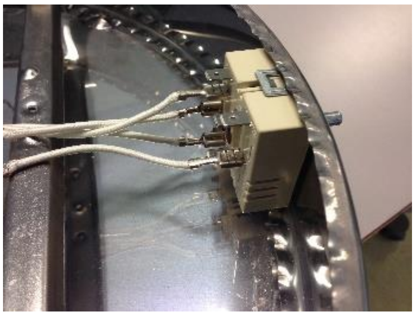


Figure 33: Regulator in place

This part is to be placed at the back of the casing below the edge where the bottom plate is to be fastened. This edge is sharp and uneven. The pin sticking out at the front is to be inserted through a hole in the casing. The diameter of the hole and the pin is 8mm and 6mm respectively. This requires a precision down to a 1mm radius. The part also needs to avoid the power bracket when being inserted. The cables at the back of the regulator will hit the power bracket so the grip has to be so tight that the regulator do not twist in the grip. To place this part, the gripper needs to be agile, precise and have a tight grip.

D.1.1.3.2.1 *Alternative idea (one-arm solution)*

An alternative to grasp the whole part with a gripper directly, we developed a jig[6] for this part where the grasping point can be adjusted to fit whatever size or shape we find is most convenient for our gripper solution. This jig would also hold the regulator in place when fastening. A drawing of the jig is shown to the right. This invention was no longer needed when another robotic arm was introduced.

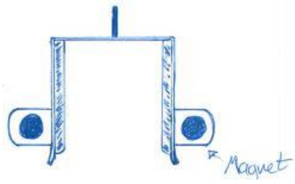


Figure 34: Magnet jig for E-REG, drawing

D.1.1.3.3 Power inlet E-PI

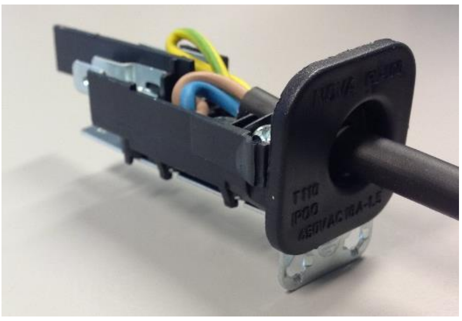


Figure 35: Power inlet E-PI

Dimensions:

- Cable:
  - 0,7 cm in diameter[4]
- Lower metal part in front with screwholes:
  - Width:
    - 2,2 cm
  - Height:
    - 0,8 cm
- Top part front, square black plastic part with holes for the power cable
  - 3,0 cm x 3,0 cm
- Total length from square plastic part to inner point
  - 8,0 cm

There are many ways to grip this part. To be sure that the gripper can grasp it at any point, it has to open at least 3,0 cm. If the gripper grasp at the cable, it is sufficient with a 0,7 cm opening. The cable is a more practical area to grasp this part because even though it is round, it has a lot more convenient surface area than any other place that is possible to grasp when placing the part. This part is to be inserted from the outside of the casing by pressing it in.

D.1.1.3.4 Cable shoe

The area to be grasped is where the "bubble" shows. The connecting point is oriented the same vertical way as the drawing shows. How the part will be grasped in this area depends on what gripper solution we chose. The cable fits both ways, as long it is vertical. To grasp this, the gripper has to be able to open at more than 2 mm.

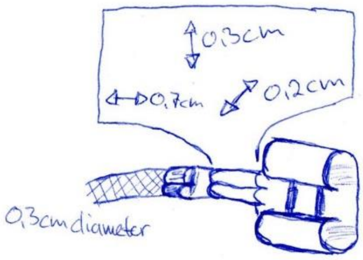


Figure 36: Cable shoe

Dimensions of the grasping area:

- Width:
  - 0,2 cm
- Height:
  - 0,3 cm

[4] Initially we had an E-PI with a cable of 0,9 cm in diameter. This turned out to be an order error.

- Length:
    - 0,7 cm
- Cable:
    - 0,3 cm in diameter

The reason why we have decided that this is the best area to grasp this part is that if the gripper grasp the cable, there will be no support to fit the cable shoe on the connecting point. It will also make it difficult to orient the shoe the right way. If the gripper grasps at the tip, it will get good support and a firm mass to hold. The backside is that it squeezes the tip together so it will not fit on the connector point. By grasping the point shown, a gripper will the same support, but it will not affect the properties of the cable show. In addition, the edge of the arches that holds the cable shoe in place will help the gripper finger not to slip when connecting.

D.1.1.3.5 Bottom plate M-BP

This is a flat circular plate with a diameter just under 41 cm. The surface area in the middle is completely flat and has no holes. Closer to the edge, there are three “legs” and some holes for venting. These parts are stacked in piles and there is no space between them. This means that it is almost impossible to fit gripper fingers between them to get only one. To grasp a bottom plate, we need something that can grasp a flat surface.

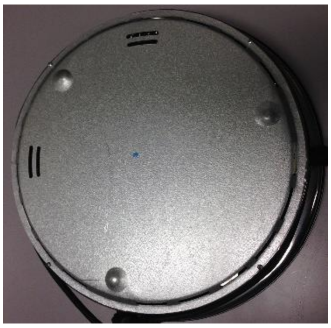


Figure 37: Bottom plate M-BP

D.1.1.3.5.1 *The griddle as a whole*

The gripper solution for lifting the bottom plate will also apply to lifting the whole griddle, as this is the best gripping point.

D.1.1.3.6 Reflector M-REF

Before the griddle makes it to the assembly station for the automation system, some parts are already mounted together and stacked in a pile. This include the cooking top, the heating element, the heat sink, the reflector and the casing. To move these halfway griddles from the pile to the station, either the assembler or the robot can move it. If the assembler moves it, there is no need for a gripper for this purpose. To save the assembler’s body and have the ability to work while the assembler is not present, a gripper solution that can move the griddle would be handy.

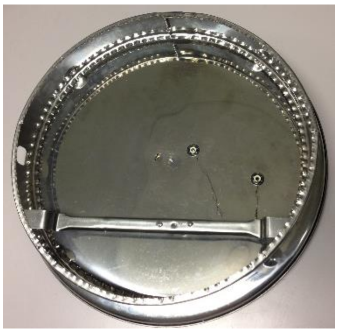


Figure 38: Reflector M-REF

The most suitable gripping point is in the center of the reflector. To grasp the reflector, the gripper needs the same properties as for gripping the bottom plate. The same solution as for the bottom plate would fit for this purpose to.

D.1.1.4 Needs

In addition to the minimal functional requirements for the gripper, other factors weigh in.

D.1.1.4.1 Cost

The gripper has to be low cost. We are developing a low cost system so the gripper should not be too pricey.

D.1.1.4.2 Size

The total end effector has to be agile so it can reach into every tiny spot we need it to. This require the gripper to be as small as possible. The gripper still has to be big enough to manage to grasp all the components.

D.1.1.4.3 Weight

The maximum load on the robotic arm is 10kg. The total weight of the small griddle is 6,5 kg. This gives us 3,5kg for the gripper, mounting bracket and any other extra equipment.

D.1.1.4.4 Force

The gripper need sufficient gripping force to be able to pick up and hold components without dropping them.

D.1.1.4.5 Load

The gripper should be able to handle the load of the small griddle, which is 6,5kg.

D.1.1.4.6 Robustness

The gripper has to be robust and durable. Accident and malfunctions can occur so the gripper should be able to bump into something without breaking. The gripper also has to be rated to grip thousands of times before wearing down. A cheap gripper will instantly save costs, but if the gripper has to be replaced regularly, the cost will increase over time and eventually be higher than the cost of a more expensive gripper.

D.1.1.5 Summary

We need a gripper that:

- can take use of flat surfaces on small components.
- can open about at least 1,2 cm (minimum difference between E-CC and E-REG).
- is able to hold the E-CC and E-REG without getting in the way of the screwdriver.
- is able to get into tight places.
- is low cost.
- is relatively small.
- is not too heavy.
- has sufficient gripping force.
- can tolerate the load of the griddle.

D.1.2 Relevant technology

D.1.2.1 VERSABALL

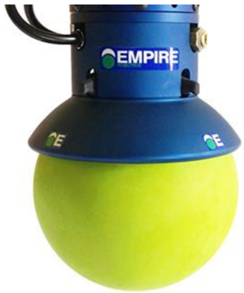


Figure 39: VERSABALL

The first gripper solution we looked at was the VERSABALL from Empire Robotics[7] that Egil Utheim from Mektron tipped us about. This particular gripper is used in many applications with robotic arms form Universal Robots and is made to fit easily with their system.

“The core principle behind VERSABALL is the jamming phase transition of granular materials. The green ball is filled with a sand like material. When air is pumped into the ball, the ball softens. The ball is the pushed against the target object. Pulling air out of the ball jams the sand like material together, causing the ball to harden. This transition from soft to hard creates the gripping forces.”[8]. This technology allows us to grip objects with any type of shape. The grip is also quite strong so it can hold a great amount of weight.

Even though this gripper has the capability of gripping all of the components, it is too bulky and impractical for our use. The sharp edges on the griddle will wear down and cut through the rubber material on the gripper. The VERSABALL is also quite heavy with the smallest version weighing almost 1,6 kg and the big one weighing just above 3,4 kg. Another factor weighing down is the price which is about 40 000 NOK.

D.1.2.1.1 Pros

- Easy to fit on the robot
- Can grip every component
- Strong grip

D.1.2.1.2 Cons

- Expensive
- Not agile
- Bulky
- Heavy

D.1.2.2 Parallel electric gripper

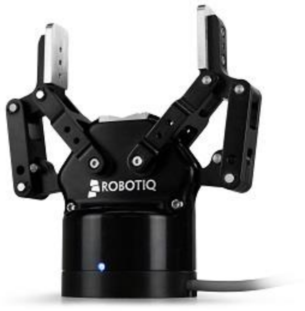


Figure 40: 2-finger adaptive gripper from Robotiq

The second type of grippers we looked at was parallel electrical grippers. The one model that caught our attention was the 2-finger 85 adaptive gripper solution developed by Robotiq [9]. Just like the VERSABALL, this fits easily with the Universal Robots system.

The gripper has 85mm stroke, which is sufficient travel to be able to grip every component. The grip force range from 60 to 200N, which is sufficient to hold every component. The parallel grip repeatability is 0,05mm and the weight is only 850g. The gripper also has position control, grip force control, speed control and grip detection.

This gripper seems perfect for perfect for this project, but it has two drawbacks; maximum payload and price. The maximum recommended payload is rated to 5 kg. As mentioned earlier the weight of the small griddle is 6,5 kg. This rules out lifting the whole griddle. The price of this gripper is about the same as the VERSABALL, about 40 000 NOK.

D.1.2.2.1 Pros

- Position control
- Grip force control
- Speed control
- Grip detection
- Easy to set up
- Lightweight
- Agile
- Adaptive fingers
- Internal and external gripping

D.1.2.2.2 Cons

- Expensive
- Cannot handle the weight of the griddle

D.1.2.3 Parallel pneumatic gripper

The pneumatic version of the parallel gripper is a cheaper and simpler alternative to the electric. Pneumatic means that it is operating on air pressure instead of e.g. electricity. This is a robust type of gripper with low maintenance. It does not have the same fancy features as the electric gripper, but it opens and closes, which is what we need it to do. However, there are accessories like position sensors and advanced air controllers that can read and control the position of the gripper finger. To utilize a pneumatic gripper will not lead to additional expenses because Engmark Meteor already has a compressed air system installed at the factory.

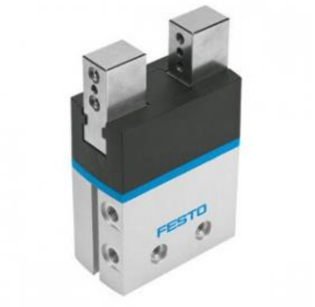


Figure 41: Parallel pneumatic gripper from Festo

The stroke to size/weight ratio is not as good as for the electric type. The parallel gripper with the longest travel that can handle the weight of the griddle is the DHPS-35-A [10] (see figure 12). This weighs 1,3 kg and has a travel of 25mm. The total closing gripping force is 910N and it can lift 450N per gripper jaw. The price for this particular gripper is about 5 000 NOK for the gripper alone and then a couple thousand for extra accessories.

D.1.2.3.1 Pros

- Same type of grip for every component
- Internal and external gripping
- Adjustable gripping force (by regulating air pressure)
- Several accessories available to make the gripper more advanced
- Low maintenance
- Reasonably priced

D.1.2.3.2 Cons

- Poor stroke to weight/size ratio
- No built in features

D.1.2.4 Angled and radial pneumatic gripper

The angled and the radial gripper work on the same principle of having a swivelling movement. The difference between the two is that the angled gripper comes with 20 degrees travel per gripper jaw and the radial has 90 degrees of travel per gripper jaw. The size of the components these grippers can handle depends on the gripper finger design. The advantage of using grippers with swivelling movement is that they can grasp objects with great variation in size.

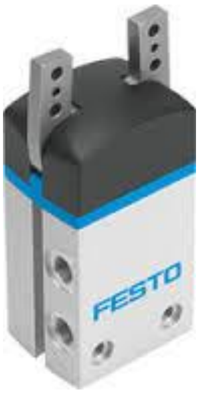


Figure 42: Angled pneumatic gripper from Festo

The biggest models of both the angled gripper, DHWS-40-A [11], and the radial gripper, DHRS-40-A[12], from Festo weighs about 800 grams and can lift 200N per gripper jaw. The total closing gripping torque for the radial gripper is 660Ncm and 1362Ncm for the angled gripper. A problem with these grippers is that it becomes difficult to pick objects from tight places and to grasp objects that are close together. The grip will change for objects of different size because of the change in the angle of the gripper jaws. These are in the same price range as the parallel pneumatic gripper and have the same options for accessories.

D.1.2.4.1 Pros

- Wide opening
- Internal and external gripping
- Adjustable gripping force (by regulating air pressure)
- Jaws can be designed to grip a wide range of components
- Several accessories to make the gripper more advanced
- Low maintenance
- Reasonably priced
- Lightweight

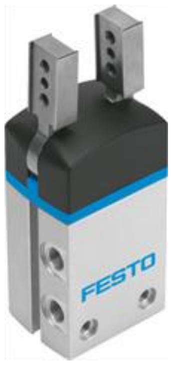


Figure 43: Radial pneumatic gripper from Festo

D.1.2.4.2 Cons

- Can be difficult to pick up objects in various situations
- Different grip for components of different size
- Inconsistent grip
- Difficult to grasp objects from tight places
- No built in features

D.1.2.5 Vacuum gripper

Figure 44: Suction cups

Vacuum grippers are designed to grip flat surface areas. As long as the surface has no holes and can keep the vacuum pressure, the vacuum gripper can grip any type of material; metal, cardboard, wood etc. The vacuum gripper use either a vacuum pump [13] or a vacuum ejector to create suction force. The vacuum pump is a pump that creates vacuum directly, but the vacuum ejector is a small T-shaped nozzle that creates vacuum from airflow. This is called the venturi effect [14]. In our case, a vacuum ejector will be the easiest and cheapest to utilize since we have compressed air available.

A vacuum gripper does not work on vacuum alone; it needs a type of suction cup as well. These come in many different shapes and sizes with different special properties. At Engmark Meteor, they use a type that is specialized for oily sheet metal [15]. They use the vacuum gripper to move metal cutouts from the laser cutter to a pallet and to move metal cutouts in and out of the metal press. The metal press needs to be lubricated regularly so the metal do not stuck in the press. This may leave a thin oily layer on the metal that leads to decreased grip. This is why they use suction cups with this special property to handle the metal components. To handle the smaller components, it will suffice with simple suction cups.

D.1.2.5.1 Pros

- Simple installation and operation
- Wide range of standard products
- Easy to customize
- Versatile
- Strong when it comes to lifting capability and gentle when it comes to object handling

D.1.2.5.2 Cons

- Not good at handling objects with holes
- Need a flat surface
- Chemical contamination and mechanical strain at contact surface

D.1.3 Conclusion

We concluded that we needed a pneumatic parallel gripper for the small components and vacuum gripper to grasp the bottom plate and reflector.

D.1.3.1 Pneumatic parallel gripper

Our first design of the workstation based on using one robot arm to pick, place and fasten the different components. After analysing this design, we noticed that it would be ineffective to have one robot handling both a gripper and a screwdriver. This led to the decision of utilizing two robotic arms; one with a screwdriver and the other with a gripper.

We first decided to focus on the two primary components, the E-CC and the E-REG. Both of these components need to be held in place while fastened. To take advantage of the square shapes and flat sides, we decided that a parallel gripper is be the best choice. This gives the best stability and the most consistent and precise grip to both grasp and place these components.

To decide what parallel gripper to choose, we considered the price, weight, stroke and the possibility of extending its application. The electric parallel gripper from Robotiq is lightweight and sufficient stroke, but it became too expensive and the maximum payload is not sufficient to lift the griddle as a whole. We looked at many other alternatives of electric parallel gripper, but they were either too weak to lift the griddle or too heavy with the griddle combined for the robot to lift. We then browsed through numerous pneumatic parallel grippers from Schunk, PHD, Parker and Festo. Mektron have contact with both Schunk and Festo so we focused our search with these two. After many hours of careful consideration, we concluded that the DHPS-35-A [10], described in chapter 2.3, is the best choice. This is reasonably priced, robust, can handle the weight of the griddle, not too heavy, not to big, strong consistent grip, low maintenance and have the possibility of being upgraded with different accessories. The total stroke of the gripper is 25mm and the size difference between the cable connector and the regulator is 12mm. This gives a little wiggle room when picking components.

D.1.3.2 Vacuum gripper

To grasp the bottom plate and the reflector, we decided to use vacuum grippers because they are already using it at Engmark Meteor with great success. We develop a vacuum gripper tool [16] that will work as an extension of the parallel gripper. The parallel gripper will be attached to the robotic arm and will connect with the vacuum tool when to grab the reflector and the bottom plate.

D.1.4 References

1. Robotics Bible. *Vacuum grippers*. [22.03.2015]
   http://www.roboticsbible.com/vacuum-grippers.html
2. Robotics News. *Vacuum Gripper for Handling Photovolt. Components* [22.03.2015]
   http://www.robotics.org/content-detail.cfm/Industrial-Robotics-News/Vacuum-Gripper-for-Handling-Photovolt-Components/content_id/1441
3. Robotic*.* *Top 5 Advantages of Servo-Electric Grippers*. [23.03.2015]
   http://blog.robotiq.com/bid/37840/Top-5-Advantages-of-Servo-Electric-Grippers
4. phd. *Grippers – Pneumatic & Electric, Angular & Parallel*. [22.03.2015]
   https://www.phdinc.com/products/category.aspx?product=grippers
5. Wikipedia. *Robot end effector.* [31.03.2015]
   http://en.wikipedia.org/wiki/Robot_end_effector
6. *Product Development Document Jig for regulator E-REG*
7. Empire Robotics. [10.05.2015]
   http://empirerobotics.com/#
8. Empire Robotics. *How It Works.* [10.05.2015]
   http://empirerobotics.com/#howitworks
9. Robotiq. *Adaptive Robot Gripper 2-finger 85.* [11.05.2015]
   http://robotiq.com/products/industrial-robot-gripper/
10. Festo. *Parallel grippers DHPS.* [12.05.2015]
    http://www.festo.com/cat/en-gb_gb/data/doc_ENGB/PDF/EN/DHPS_EN.PDF
11. Festo. *Angle grippers DHWS.* [12.05.2015]
    http://www.festo.com/cat/no_no/data/doc_engb/PDF/EN/DHWS_EN.PDF
12. Festo. *Radial grippers DHRS.* [12.05.2015]
    http://www.festo.com/cat/en-gb_gb/data/doc_ENGB/PDF/EN/DHRS_EN.PDF
13. Wikipedia. *Vacuum pump.* [12.05.2015]
    http://en.wikipedia.org/wiki/Vacuum_pump
14. Wikipedia. *Venturi effect.* [12.05.2015]
    http://en.wikipedia.org/wiki/Venturi_effect
15. piab. *Oily sheet metal.* [12.05.2015]
    https://www.piab.com/en-US/products/Suction-cups/suction-cupsgrippers1/oily-sheet-metal-suction-cups/
16. *Product Development Document Vacuum Gripper Tool*

D.2 Screwdriver

Abstract

This document addresses the requirements for the screwdriver and what type of screwdriver we have concluded is the best suited for our project. The conclusion is based on the selection of screwdrivers from a catalogue provided by Engmark Meteor.

Contents

Abstract 175

Contents 175

D.2.1 Introduction 176

D.2.1.1 Screw types 176

D.2.1.2 Speed requirement 176

D.2.1.3 Torque requirement 177

D.2.1.4 Accessories.... 177

D.2.2 Relevant technology 178

D.2.2.1 Pneumatic screwdriver 178

D.2.2.1.1 Pros 178

D.2.2.1.2 Cons.... 178

D.2.2.2 Electric screwdriver.... 178

D.2.2.2.1 Pros 179

D.2.2.2.2 Cons.... 179

D.2.3 Conclusion....179

D.2.4 References 179

D.2.1 Introduction

The main task we are to automate at Engmark Meteor is the fastening of some of the components on the griddle. These components are fastened with screws of different kind. As of today, the assemblers pick the screws by hand from a box and place it at the screw bit. Then they use a handheld push-to-start pneumatic screwdriver to fasten the screw (more info 2.1). When the screw hits the point where it is fully tightened, the screw stops rotating instantly, but the screwdriver still has some rotational energy. This energy is transferred to the assembler's wrist as a shock. The shock itself is not powerful, but there are 15 screws to be tightened in every griddle. Engmark Meteor produces about 10,000 griddles a year, which adds up to 150,000 shocks delivered to the assembler's wrist.

The main purpose of automating parts of the screwing is reducing manhours fastening screws. The reduction of stress impact to the assembler is a positive consequence. Implementing automatic fastening of screws saves both time and the assemblers' health.

D.2.1.1 Screw types

The three types of screws they use at Engmark Meteor today to fasten components is a standard M4 6mm and two different self-tapping screws. All three types have crossheads. The M4 screw fit with the threads in the regulator (E-REG) so this has to have M4 threads, but the screw head type can be a different one if needed. The other two self-tapping screw types are used to fasten the cable connector (E-CC), the power inlet (E-PI), the reflector (M-REF) to the bracket connected to the heat sink (M-HS) and cooking top (M-CT), and to fasten the bottom plate (M-BP) to the casing (M-CA). These holes do not have special threads the screw need to fit in like the M4, but the holes in the parts to be connected have to line up to use these screws.

There are four more screws in the cable connector. These are to connect cables from the regulator and the heating element. The process of connecting the cables to the cable connector is too difficult to solve with an automation system, so the assembler will still have to perform this task.

D.2.1.2 Speed requirement

To make the job easier for the automation system of having the holes to line up to fasten the screws, we change the self-tapping screws to self-drilling screws. This way, the system only has to locate the outer hole and then drilling a screw through the underlying material. Self-drilling screws require high rpm to drill through the metal. The screws we have used for testing has an operating speed of 1700-2500 rpm at 10N contact pressure [3].

The screwdriver we have used for testing is the same type of handheld pneumatic gripper they use at Engmark Meteor today. At full speed, we have reached 500 rpm. With the maximum force possible, 120N, applied by the UR-10 robotic arm and the screwdriver at full speed, it took about 4-5 seconds to drill through the metal and fully tighten the self-drilling screw. At Engmark Meteor, we tried to fasten the same type of self-drilling screw in the same type of material with an electric drill with an estimated speed of 4000 rpm. With almost no forced applied, this took under one second.

We want to apply as little force as possible to the griddle as this may cause the griddle to deform. A screwdriver with operating speed of 2000±500rpm will meet the requirement for the self-drilling screws and the time to fasten will be relatively short with almost no force applied.

There is no requirement to speed for the M4. As long as the screw enter the threads correctly, the speed is irrelevant.

D.2.1.3 Torque requirement

The minimum torque requirement for the screwdriver is based on the standards for fastening machine screws. For the M4 it is anywhere from 0,75-5Nm depending on the metal [4], [5]. Assuming that 0,75Nm is the minimum value for an M4 to stay fastened through the lifetime of a griddle, we set this as the minimum requirement to torque.

For the screws or the fastening material to stay intact when tightening, the torque has to be limited. This goes for both types of screws. There have not been conducted any tests to see at what value the different screws and materials buckle, but the screws have been fastened at about 1,5Nm[5] without leaving visible damage.

It is not necessary to have an adjustable torque feature if the torque of the screwdriver is set to be between 0,75-1,5Nm. A screwdriver with a torque near the higher value will give a better result than a torque near the lower value.

D.2.1.4 Accessories

By studying the different bits and drives used for automated assembly, we concluded that a type of hex drive would be most suitable [6]. The hex drive has a low chance of cam-out and can handle more torque. We ran some tests with hex drive screws and learned that they slide easily on to the bit, stay stable onto the bit and there were no cam-outs. Since the screwdriver is going to fasten both the self-drilling screws and the M4 screws, we had to find both types of screws with the same drive. We found both M4 and self-drilling screw with a hex drive, but not the same type. After browsing through thousands of screws from different suppliers, we decided to stick with the crosshead drive.

We noticed when testing different screws, that the crosshead drive does not stay very well at the bit when being picked up. The inner corner of the drive pulls towards the tip of the bit when reaching for the screw and it had a tendency to slide down and hang from this corner when lifted. To increase the reliability of picking and holding the screw, we concluded that a suction pick-up would be a useful accessory.

[5] Approximate value read from the torque setting of the pneumatic screwdriver borrowed from Engmark Meteor.

D.2.2 Relevant technology

D.2.2.1 Pneumatic screwdriver

Pneumatic screwdrivers are powered by air instead of electricity and are usually operated by hand. These are robust and easy to operate. These come with either a push-to-start function or a lever. The push-to-start screwdriver works by applying force against the tip of the tool to get the motor running. The air pressure determines the speed of the pneumatic screwdriver. There is no speed control on the screwdriver, it only controls on or off and fasten or loosen.

For some types of pneumatic screwdrivers, it is possible to set the screwdriver to operate at a specific torque so screws are fully tightened, but without damaging the screws, components or material. When the torque gets exceeds the set value, it shuts off automatically.

Pneumatic screwdrivers comes in variety of models with different specifications. Torque ranges from 0,05-30Nm and speed ranges from 250-2200rpm.

From the testing of the pneumatic screwdriver borrowed from Engmark Meteor, we learned that the screwdriver needs 70N applied to start and it operates and 500rpm. Other screwdrivers in the same series as this one are available with torque ranges from 0,05-27Nm and with rpm ranges from 280-2200 [4]. These cost 500-650 dollars.

D.2.2.1.1 Pros

- Adjustable torque
- Reasonably priced

D.2.2.1.2 Cons

- Needs modification to be controlled automatically
- Needs customized mounting bracket
- The push-to-start version needs 70N[6] applied to start
- No speed control

D.2.2.2 Electric screwdriver

Electric screwdrivers are the most widely used type of screwdriver. They come in a great variety of both design and specifications. Like the pneumatic screwdriver, they come with a push-to-start function or a lever. There are also models that can be operated remotely that are made specifically for automatic assembly machines.

Electric screwdrivers can be delivered with intelligent power supplies that registers if a screw is tightened or not. It is also possible to pre-program different torques for different screws in a sequence.

Even though the electric screwdrivers have more features than the pneumatic when it comes to functionality, they are less robust and durable. The electric screwdrivers with the same level of controllability as the pneumatic cost about the same. Some are cheaper, but they show signs of low quality.

[6] Value obtained from test of pneumatic screwdriver borrowed from Engmark Meteor.

D.2.2.2.1 Pros

- High rpm
- Speed control
- More accessories available
- High level of controllability

D.2.2.2.2 Cons

- More expensive than the pneumatic
- Not as robust and durable as the pneumatic

D.2.3 Conclusion

Our choice of screwdriver is based on the assortment from the HIOS-catalog provided by Håvid Engmark. We looked through hundreds of different screwdrivers from other suppliers before we learned that the screwdriver needed a vacuum system to hold the screw in place. The only screwdrivers we knew had a suction unit option was the ones in the HIOS catalogue so we checked the different screwdrivers to get the best match to the requirements.

We concluded that the αF-6500 from HIOS [1] is the best choice of screwdriver. Out of the two with the highest speed of 1700rpm, this has the highest torque range, which is 0.5-1.1Nm. It can also be fitted with a vacuum pick-up system.

Screwdriver: 6 535,- SEK
Power supply HIOS CLT-AY-61: 15 835,- SEK
Mouthpiece for suction pick-up: 275,- SEK

D.2.4 References

1. HIOS. *Brushed Screwdriver for Automatic Assembly Machines,* page 35. [24.04.2015] http://hios.com/EN/catalog2014/pageview/pageview.html?PHPSESSID=fbcb5116a31ede9bf9b84bf60153e856#page_num=35
2. Würth. *The AW recess, page 14, 0206 39 13.*[23.04.2015] http://www.wurth.com.au/Flipbook/Screws/files/assets/common/downloads/publication.pdf
3. Express Assembly. *Air screwdrivers.* [15.05.2015] http://www.expressassembly.com/catalog/new-air-screwdrivers.php#cp
4. Integrated Publishing. *Standard Torque Values – Metric System*. [12.05.2015] http://electronicfilters.tpub.com/TM-10-4330-237-13P/css/TM-10-4330-237-13P_105.htm
5. Maryland Metrics. *Standard tightening torque for metric screw threads*. [12.05.2015] http://www.marylandmetrics.com/tech/Standard_tightening_torque.pdf
6. Assemblymag. *Fasteners for Automation.* [17.02.2015] http://www.assemblymag.com/articles/84648-fasteners-for-automation

D.3 Camera

Abstract

This document sums our research of different camera options for our vision system. It concludes that the vision system will be split into two parts, one practical solution for Engmark Meteor and one abstract solution for Mektron.

Contents

Abstract 180

Contents 180

D.3.1 Introduction 181

D.3.2 Relevant technology 181

D.3.2.1 Microsoft Kinect Version 2 181

D.3.2.1.1 Pros 181

D.3.2.1.2 Cons 181

D.3.2.2 Intel RealSense 181

D.3.2.2.1 Pros 181

D.3.2.2.2 Cons 181

D.3.2.3 Google Project Tango 181

D.3.2.3.1 Pros 181

D.3.2.3.2 Cons 182

D.3.2.4 IFM sensors 182

D.3.2.4.1 Pros 182

D.3.2.4.2 Cons 182

D.3.3 Conclusion 182

D.3.4 References 182

D.3.1 Introduction

We are required to make a vision system for our final product. The main criterion for this system is to remain low-cost. Recent development in the game and camera industries has brought many new and interesting technologies. Microsoft recently released the second version of the Kinect. Intel is about to release their RealSense and Google just presented their new Project Tango.

D.3.2 Relevant technology

D.3.2.1 Microsoft Kinect Version 2

The Kinect is a low cost depth camera. This means the camera gives out two feeds, one regular RGB for colour images (1080p) but also a XYZ feed. The resolution of this feed is significantly lower (512x424) but still very competitive to other products. When price is considered, the Kinect makes a very solid case.

The Kinect works by measuring time-of-flight of a grid of nodes.

D.3.2.1.1 Pros

- High resolution
- Precise
- Solid development team behind both hardware and software

D.3.2.1.2 Cons

- Edge detection tends to work better with structured light
- Minimum range of 0.5m limits mounting options

D.3.2.2 Intel RealSense

This product is based upon stereo camera technology. This works by taking two camera feeds, pin pointing the differences between them and thus calculating angles resulting in depth perception. This technology is not yet available to the commercial market but we managed to get hold of a developer’s edition.

D.3.2.2.1 Pros

- No range limit
- Smaller and easier to mount

D.3.2.2.2 Cons

- Precision
- Noise is harder to detect (calculations are based upon each other, rather than individual measurements)
- Young software

D.3.2.3 Google Project Tango

Same as with Inter RealSense, this is based upon stereo vision. Google has made this into a compact module able to run on a specifically designed tablet.

D.3.2.3.1 Pros

- Small and easy to mount

- No need for external computing power

D.3.2.3.2 Cons

- Precision
- Noise is harder to detect (calculations are based upon each other, rather than individual measurements)
- Technology not yet available

D.3.2.4 IFM sensors

The IFM sensors stand out from the others as they are designed to perform certain operations on the camera itself. The three versions we have looked closely upon are: object recognition, pixel counter and volume measurement. They connect through a TCP/IP interface, removing the need for a dedicated computer for algorithms.

D.3.2.4.1 Pros

- Removes need for additional computing power
- Cheap (due to mentioned above)
- Fast (< 1s calculation time)
- Easy to maintain and use
- Robust (Intended for industrial use)

D.3.2.4.2 Cons

- Limited usability

D.3.3 Conclusion

We have a wish from our primary stakeholder to use the Kinect. However, in the current project (for Engmark Meteor) the IFM sensors seems to be a smarter option. They are a small firm that want a simple and maintenance free system. The IFM sensors provide this. Further testing will be needed to conclude if the sensors have the resolution we need.

We will also continue work with the Kinect and RealSense. The Kinect offers very high and precise resolution and Mektron is very interested in a general solution for different tasks (i.e. object recognition). Work will also continue with the RealSense as an aid to the Kinect (due to its smaller size it is easier to mount on a UR robot).

D.3.4 References

1. Microsoft Kinect technical specs. [17.02.2015] http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx
2. Intel RealSense technical specs. [17.02.2015] http://support.creative.com/kb/ShowArticle.aspx?sid=120808
3. Google Project Tango official site. [10.02.2015] https://www.google.com/atap/projecttango/#devices
4. IFM official site. Device O2Dxxx and O2Vxxx. [23.03.2015] http://www.ifm.com

D.4 URscript

Abstract

This document describes the different methods used for controlling the robots from Universal Robots including URScript, PolyScope GUI and remote control using TCP/IP socket.

Contents

Abstract **183**

Contents **183**

Figures **183**

D.4.1 Introduction **184**

D.4.2 Relevant technologies **184**

D.4.2.1 GUI programming with PolyScope 184

D.4.2.1.1 Pros 185

D.4.2.1.2 Cons 185

D.4.2.2 Local script programming 185

D.4.2.2.1 Pros 186

D.4.2.2.2 Cons 186

D.4.2.3 Control by Ethernet TCP/IP Premade Interface 186

D.4.2.3.1 Pros 187

D.4.2.3.2 Cons 187

D.4.2.4 Control by Ethernet TCP/IP Self-Developed Interface 187

D.4.2.4.1 Pros 187

D.4.2.4.2 Cons 188

D.4.3 Conclusion **188**

D.4.4 References **188**

Figures

Figure 45: PolyScope programming GUI 184

Figure 46: Move tab 185

Figure 47: Loading script into PolyScope program 186

Figure 48: Simple socket example 187

D.4.1 Introduction

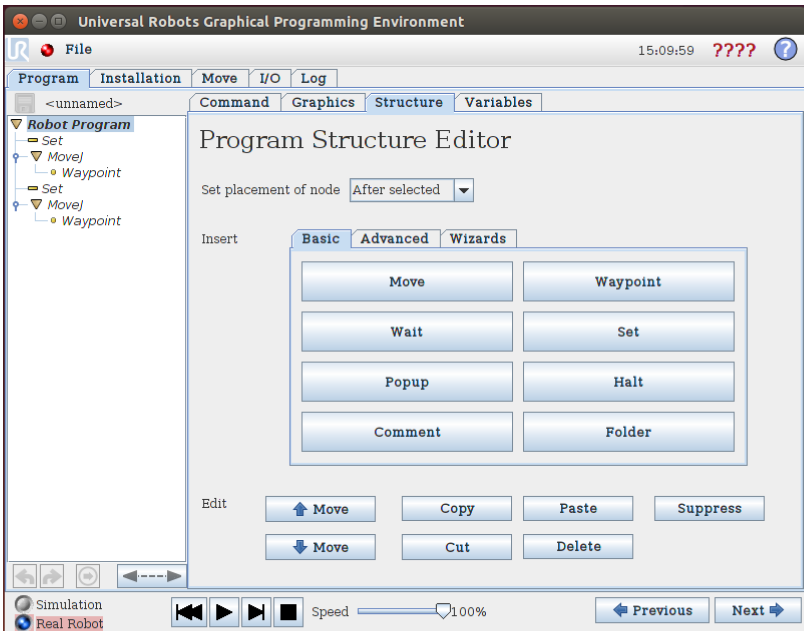
The robots from Universal Robots, as with every computer system, need to be programmed to perform a task. To make it possible to program the system for regular customers, the system is pre-programmed to interpret basic commands from a script. The designated scripting language is called URScript, which is based on a version of Python. This language contains functions that make the robot move in certain ways, in addition to read physical data. A list of special commands and functions is provided in a manual on Universal Robots support website [1].

There are mainly two ways of controlling the robot with URScript commands. You can create a script that can be run locally on the robots controller, or send commands with a TCP/IP socket connection. The controller has an integrated GUI application called PolyScope, which allows “click and place” programming of the system to allow users without much experience in programming to control the robot. It can also load scripts from USB-drives, which then can be edited and run with the GUI application.

D.4.2 Relevant technologies

D.4.2.1 GUI programming with PolyScope

The included GUI programming-tool is called PolyScope. It is an easy to use interface that requires little or no knowledge to get started. It is basically a set of buttons that represent certain pieces of code. As seen in Figure 45, the structure tab contains all functions needed to make a simple program. A function is selected, and then configured in the command tab. The program is converted to a URScript, which may be run. Scripts loaded from USB drive might also be integrated into the program made in PolyScope. One great advantage of programming the system this way, is the accessibility of real-world physical coordinates. The robot can be put in freedrive mode and moved to a desired position, which then, by the click of a button, can be read out into a position in the program. GUI programming can however be a bit limited when it comes to making advanced programs.

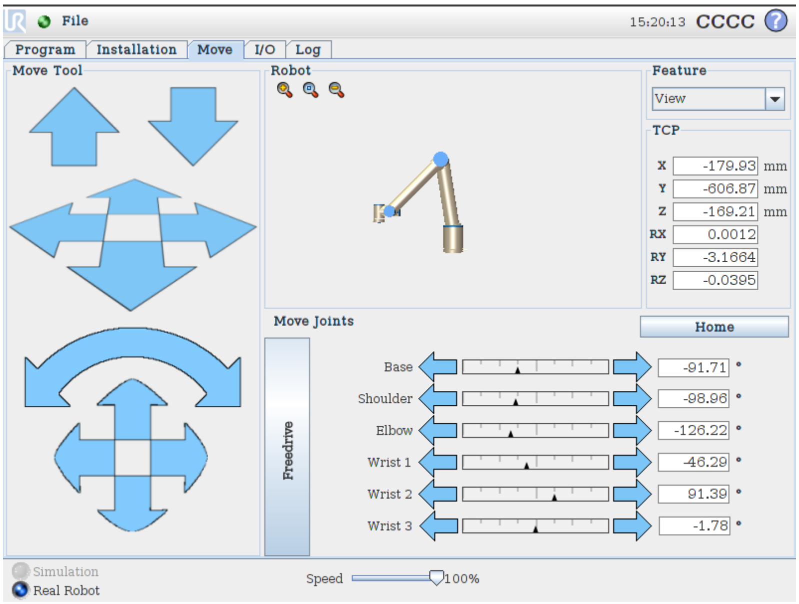

Universal Robots Graphical Programming Environment

Program Structure Editor

Set placement of node: After selected

```
Robot Program
  Set
  MoveJ
    Waypoint
  Set
  MoveJ
    Waypoint
```

Figure 45: PolyScope programming GUI



TCP		
X	-179.93	mm
Y	-606.87	mm
Z	-169.21	mm
RX	0.0012	
RY	-3.1664	
RZ	-0.0395	

Move Joints		
Base	-91.71	°
Shoulder	-98.96	°
Elbow	-126.22	°
Wrist 1	-46.29	°
Wrist 2	91.39	°
Wrist 3	-1.78	°

Figure 46: Move tab

D.4.2.1.1 Pros

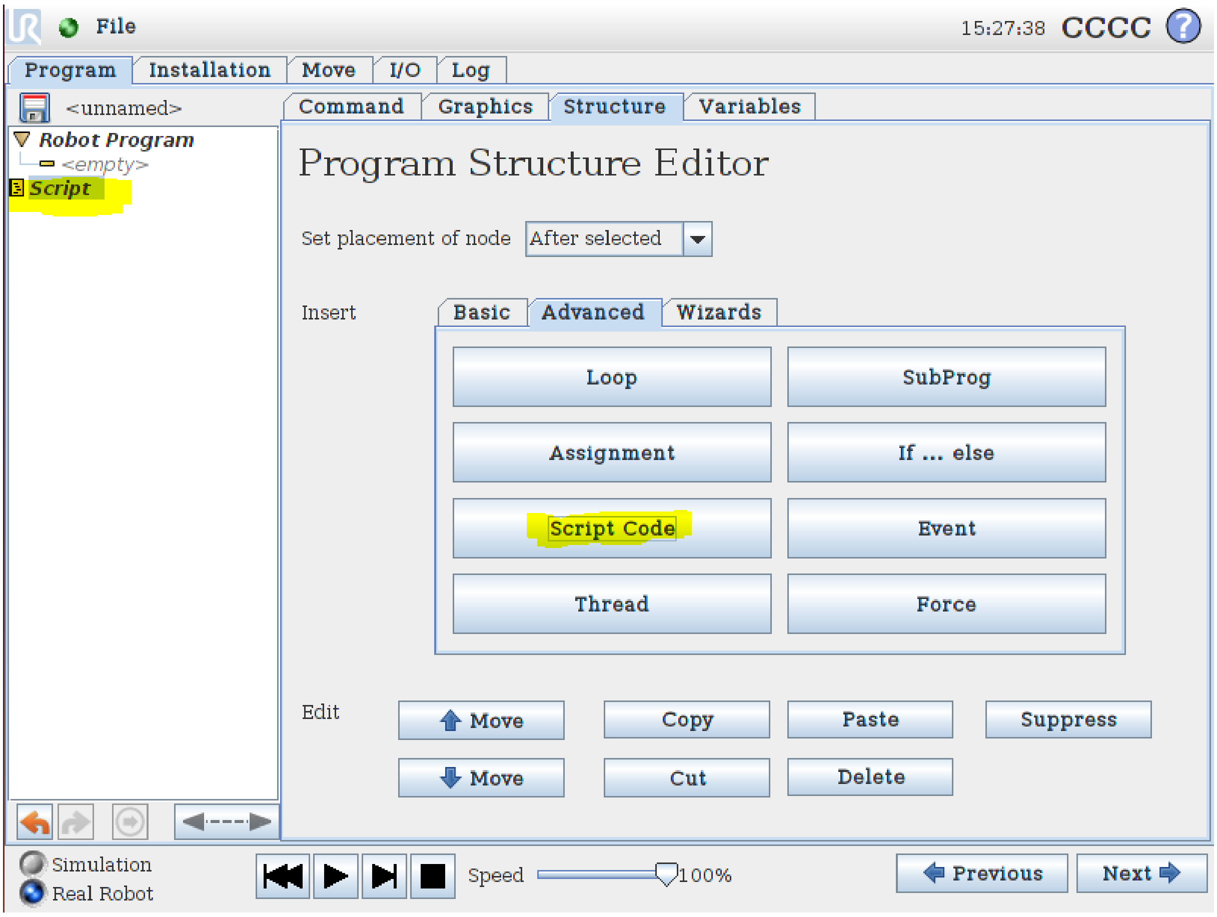
- Easy to use
- Fast
- No need for external equipment
- Easy access to positions

D.4.2.1.2 Cons

- Limited
- Robot has to be paused to edit program

D.4.2.2 Local script programming

URScripts can be written on any computer. All that is needed is a text editor. A script is written according to the URScript manual [1], and then loaded onto the Universal Robots controller. The script can be run as is, or be implemented as part of a program made in the GUI programmer. In Figure 47, it is shown how a script might be implemented into a PolyScope program.



Set placement of node: After selected

Figure 47: Loading script into PolyScope program

D.4.2.2.1 Pros

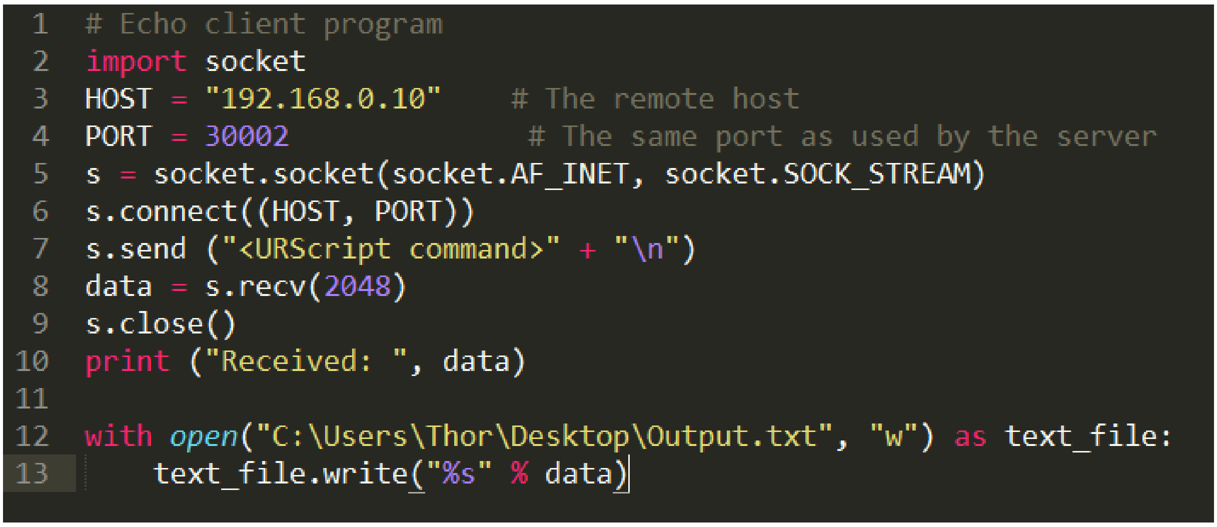
- Can be edited without stopping robot
- Grants access to full functionality of URScript

D.4.2.2.2 Cons

- Difficult to use without prior programming experience
- Robot must be paused when loading script
- Not full Python support

D.4.2.3 Control by Ethernet TCP/IP Premade Interface

The Universal Robot system can be controlled by Ethernet. The system is set up to receive URScript commands on port 30002. By creating a socket that connects with the robot on port 30002, it is easy to send commands as strings. The robot receives these strings and interprets them as if they were run as a regular script locally on the machine. The system returns a package containing positions, sensor inputs and more. An example of how to send commands to the Universal Robots system can be seen in Figure 48. The program is written in Python.



```python
# Echo client program
import socket
HOST = "192.168.0.10"    # The remote host
PORT = 30002              # The same port as used by the server
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))
s.send ("<URScript command>" + "\n")
data = s.recv(2048)
s.close()
print ("Received: ", data)

with open("C:\Users\Thor\Desktop\Output.txt", "w") as text_file:
    text_file.write("%s" % data)
```

Figure 48: Simple socket example

D.4.2.3.1 Pros

- Remote control
- Can be programmed in almost any programming language (except for the actual string sent)
- Can be edited with close to no downtime
- Simplifies interfacing with vision system

D.4.2.3.2 Cons

- Program has to be timed manually, as the robot executes commands when they are received
- Program must be run on external equipment
- Requires some knowledge about programming and Ethernet communication
- No handshake

D.4.2.4 Control by Ethernet TCP/IP Self-Developed Interface

As with the premade interface in port 30002, this communication will be on socket (usually port 30000). URScript has built-in functions that read and interpret strings from socket, which can be used to control program-flow locally on the robot. This means that we can choose how the interface will work, with handshakes and what data that will be returned. This will also allow us to access internal functions on the UR, e.g. force measurement. As with the premade interface on port 30002, controlling from a remote computer simplifies interfacing with just about any vision system.

D.4.2.4.1 Pros

- Remote control
- Can be programmed in almost any programming language
- Can be edited with close to no downtime
- Simplifies interfacing with vision system
- We can choose dataflow and handshakes etc.

D.4.2.4.2 Cons

- Requires making all functions on UR side manually (along with interface)
- Program must be run on external equipment
- Requires some knowledge about programming and Ethernet communication

D.4.3 Conclusion

Each method has their advantages in certain applications, and we might end up using all of them to accomplish our goal of using the robot with a vision system.

D.4.4 References

1. Universal Robots, URScript manual. [23.03.2015] http://www.ur-update.dk/URsupport/Manuals/ScriptManual/Release3.1/scriptmanual_en.pdf
   - A user account is however needed to gain access
2. Universal Robots, Homepage. [23.03.2015] http://www.universal-robots.com/en/
3. Zacobria, distributor of UR in Singapore [23.03.2015] http://www.zacobria.com/

D.5 Universal Robot

Abstract

This document describes the three different robotic systems made by Universal Robots, respectively UR3, UR5 and UR10.

Contents

Abstract 189

Contents 189

Figures 189

D.5.1 Introduction 190

D.5.2 Relevant technology 190

D.5.2.1 UR3 190

D.5.2.1.1 UR3 Specifications 191

D.5.2.2 UR5 192

D.5.2.2.1 UR5 Specifications 193

D.5.2.3 UR10 194

D.5.2.3.1 UR10 Specifications 195

D.5.3 Conclusion 196

D.5.4 References 196

Figures

Figure 49: UR3 190

Figure 50: UR5 192

Figure 51: UR10 194

Figure 52: UR family 196

D.5.1 Introduction

Universal Robots has three different systems that might be applicable to our project. They are all relatively cheap, easy to use and efficient. The systems have advanced safety functions that allow them to work without area restrictions. This makes them very useful in situations where cooperation with operator is beneficial. The specifications for the three robots are very much alike. In the next chapter, there is a specification list for each of them.

D.5.2 Relevant technology

D.5.2.1 UR3

UR3 is the newest invention. Released mid-March 2015, this robot is the neatest with a lifting capacity of 3 kg. It is what you could call a desktop robot. The thing that differs the most from UR5 and UR10, is the infinite spin on the last joint. While all other joints can turn only up to two full turns, the last joint on the UR3 can turn as many times as necessary. It makes it very useful for mounting screws. The speed of the joint is however very low compared to an electric or pneumatic screwdriver.

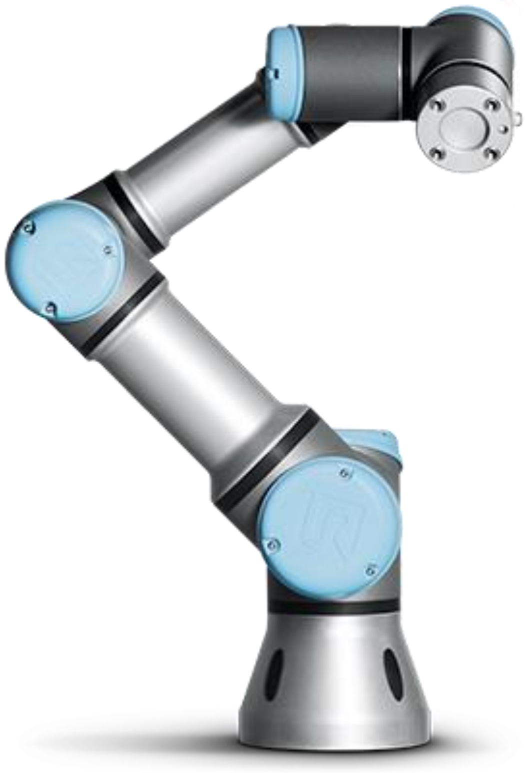


Figure 49: UR3

D.5.2.1.1 UR3 Specifications

Robot weight: 11 kg

Payload: 3 kg

Working radius: 500 mm

Joint ranges: +/- 360 "Infinite" end joint

Speed: All Wrist joints 360 deg/sec. Other joints: Max 180 deg/sec. Tool: Typical. 1 m/sec

Repeatability: +/- 0.1 mm

Footprint: Ø 118 mm

Freedom: 6 rotating joints

Size of controllerbox (WxHxD) 475mm x 423mm x 268mm

I/O ports in controller box: 16 digital in, 16 digital out, 2 analogue in, 2 analogue out

I/O ports on tool head: 2 digital in, 2 digital out, 2 analogue in

Power Supply: 24V 2A in controllerbox and 12V/24V 600 mA in tool

Communication: TCP/IP 100Mbit: IEEE 802.3u, 100Base-Tx - Ethernet Sockets, Modbus TCP

Programming: Polyscope graphical user interface on 12" touch screen with mounting

Noise: Comparatively noiseless

IP classifications: IP54

Power consumption: Approx 100 Watt using a typical program.

Collaboration operation: 15 Advanced safety functions

Materials: Aluminium, ABS plastic, PP plastic

Temperature: The robot can work in a temperature range of 0-50 deg C

Power Supply: 100-240 VAC, 50-60 Hz

6.0 metre cable between robot and control box
4.5 metre cable between touchscrfeen and control box

D.5.2.2 UR5

The UR5 was the first robot produced by Universal Robot, released in Denmark in 2009. Since then, the robot has gotten several updates. It has a lifting capacity of up to 5 kg.

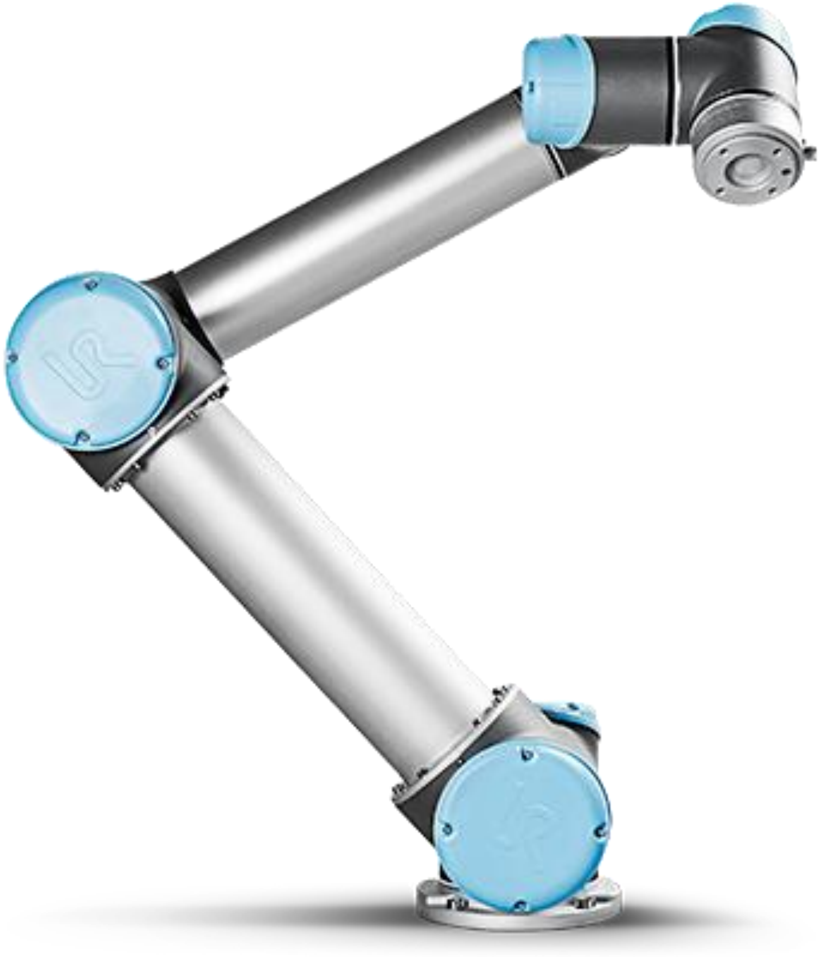


Figure 50: UR5

D.5.2.2.1 UR5 Specifications

Robot weight: 18.4 kg

Payload: 5 kg

Working radius: 850 mm

Joint ranges: +/- 360

Speed: Joint: Max 180 deg/sec. Tool: Typical. 1 m/sec

Repeatabiliy: +/- 0.1 mm

Footprint: Ø 149 mm

Freedom: 6 rotating joints

Size of controllerbox (WxHxD) 475mm x 423mm x 268mm

I/O ports in controller box: 16 digital in, 16 digital out, 2 analogue in, 2 analogue out

I/O ports on tool head: 2 digital in, 2 digital out, 2 analogue in

Power Supply: 24V 2A in controllerbox and 12V/24V 600 mA in tool

Communication: TCP/IP 100Mbit: IEEE 802.3u, 100Base-Tx - Ethernet Sockets, Modbus TCP

Programming: Polyscope graphical user interface on 12" touch screen with mounting

Noise: Comparatively noiseless

IP classifications: IP54

Power consumption: 200 Watt using a typical program.

Collaboration operation: 15 Advanced safety functions

Materials: Aluminium, ABS plastic, PP plastic

Temperature: The robot can work in a temperature range of 0-50 deg C

Power Supply: 100-240 VAC, 50-60 Hz

6.0 metre cable between robot and control box
4.5 metre cable between touchscreen and control box

D.5.2.3 UR10

With a launch in 2012, UR10 is the buff brother of the trio. It is just bigger and more powerful. It has a greater range and can lift 10 kg.

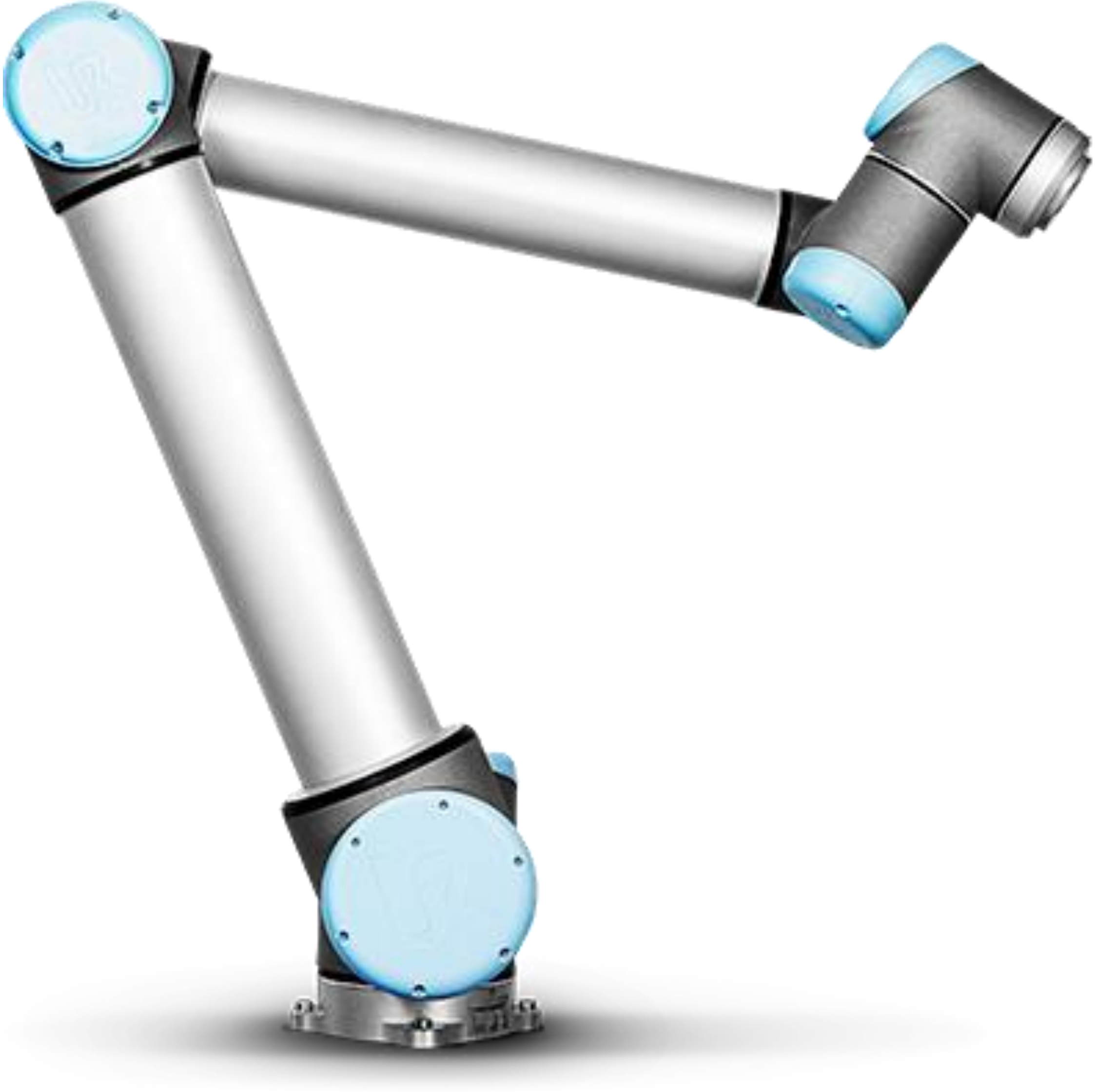


Figure 51: UR10

D.5.2.3.1 UR10 Specifications

Robot weight: 28.9 kg

Payload: 10 kg

Working radius: 1300 mm

Joint ranges: +/- 360

Speed: Base and Shoulder: 120 deg/Sec. Elbow, Wrist 1, Wrist 2, Wrist 3: 180 deg/sec. Tool: Typical. 1 m/sec

Repeatability: +/- 0.1 mm

Footprint: Ø 190 mm

Freedom: 6 rotating joints

Size of controllerbox (WxHxD) 475mm x 423mm x 268mm

I/O ports in controller box: 16 digital in, 16 digital out, 2 analogue in, 2 analogue out

I/O ports on tool head: 2 digital in, 2 digital out, 2 analogue in

Power Supply: 24V 2A in controllerbox and 12V/24V 600 mA in tool

Communication: TCP/IP 100Mbit: IEEE 802.3u, 100 Base-Tx - Ethernet Sockets, Modbus TCP

Programming: Polyscope Graphical user interface on 12" touch screen with mounting

Noise: Comparatively noiseless

IP classifications: IP54

Power consumption: 350 Watts using a typical program.

Collaboration operation: 15 Advanced safety functions

Materials: Aluminium, ABS plastic, PP plastic

Temperature: The robot can work in a temperature range of 0-50 deg C

Power Supply: 100-240 VAC, 50-60 Hz

6.0 metre cable between robot and control box
4.5 metre cable between touchscrfeen and control box

D.5.3 Conclusion

Universal Robots covers a large area with their 3 robots, from a compact table top robot to a robot with large range and lifting capacity. Håvid Engmark (our customer) has already purchased a UR10 which we have access to through our project. We might however consider the possibilities of using a UR3 or UR5 in addition to the UR10 to be able to solve more assembly tasks.

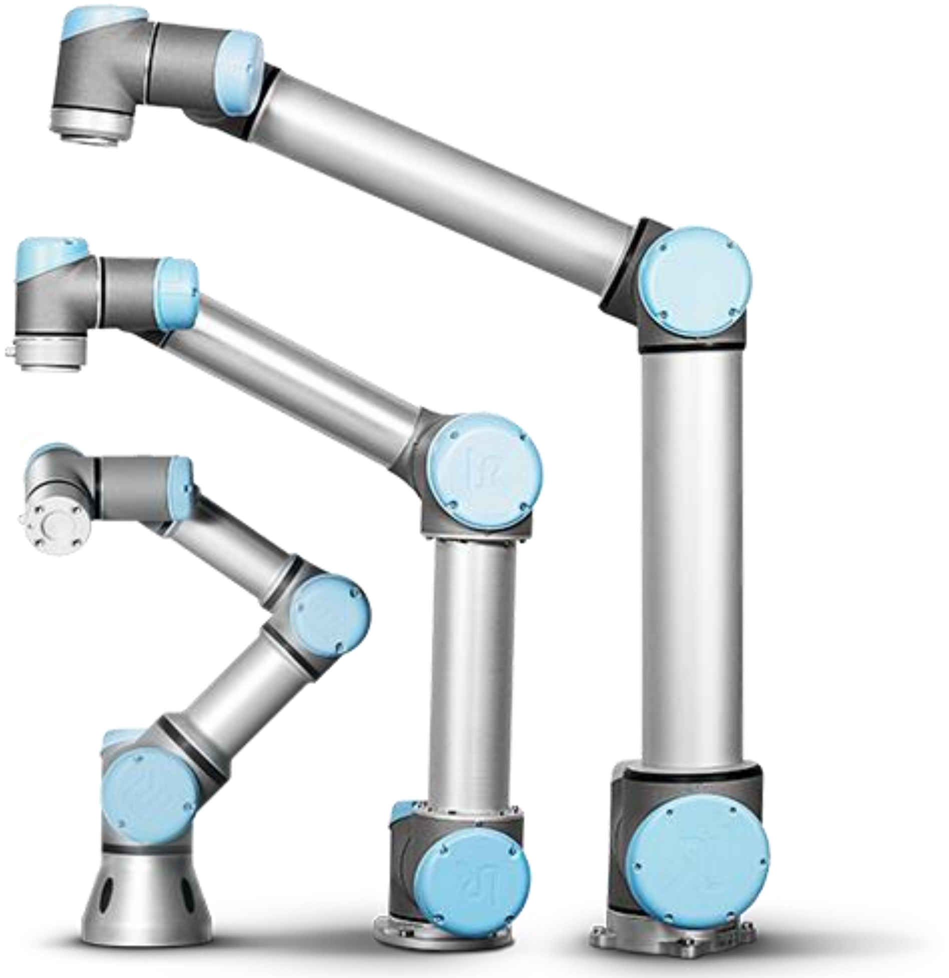


Figure 52: UR family

D.5.4 References

1. Zachobria, Universal Robot distributor, specifications for each robot [24.03.2015] http://www.zacobria.com/universal-robots-zacobria-industrial-robot-ur5-ur10-specifications.html
2. Universal Robots, Products page [24.03.2015] http://www.universal-robots.com/en/products/

D.6 3D Modelling software

Abstract

This document contains selection process for the 3D modelling software that we will use during this project.

Contents

Abstract 197

Contents 197

D.6.1 Introduction 198

D.6.2 Relevant technology 198

D.6.2.1 SolidWorks 198

D.6.2.1.1 Pros 198

D.6.2.1.2 Cons... 198

D.6.3 Conclusion...198

D.6.4 References 199

D.6.1 Introduction

We knew that our project will demand several 3D models and so we needed software to design them. There are many free programs to choose from on the internet. The problem is that most of them are very difficult to use, or they do not offer as much as the expensive programs like SolidWorks and CATIA. We are looking for something with simple basics, yet complex enough to offer features that we need during this project.

D.6.2 Relevant technology

D.6.2.1 SolidWorks

SolidWorks is one of the leading 3D modelling software in the world. It is known for its simplicity, making it easy for a person with limited experience to work with it. The program offers a package that provides all of the needed features to work with 2-3D sketching, complex surface parts, sheet metal, machined parts, and many others. It also provides flow simulation to test you component when it comes to stress analysis, thermal analysis, complex flow and such. These past years SolidWorks has been expanding greatly, which is both taking it to the next level to compete with programs like CATIA, but also making it too much for a beginner. They still manage to keep its basics simple enough, so their market includes small, medium, and big companies.

D.6.2.1.1 Pros

- Free (for students at HBV)
- Easy to use
- Already familiar with the program
- Good customer support, has its own tutorial site
- Many different features makes it perfect when working with products that consist of parts made by different methods

D.6.2.1.2 Cons

- The drawings made by students may not be used by companies to make profit
- An average workstation might not be enough when working with an assembly that has 100+ parts.

D.6.3 Conclusion

SolidWorks is very familiar to our group and so it was an easy choice. Since we get a free student version of it from school, it would be unwise to not use such good software when it does not bring any drawbacks to the project. Its basics are easy to learn, and if the mechanical engineer in our group would need any assistance, it would not take a lot of effort for another member of the group to make some simple sketches.

D.6.4 References

1. DASSAULT SYSTEMES. *SOLIDWORKS*. [04.03.2015] http://www.solidworks.com/default.htm
2. G2CROWD. *SOLIDWORKS Reviews*. [24.03.2015] https://www.g2crowd.com/products/solidworks/reviews
3. DASSAULT SYSTEMES. *DISCOVER CATIA. [04.03.2015]* http://www.3ds.com/products-services/catia/

E. Concepts

E.1 Concept document

Abstract

This document contains information about our concepts for automation and optimization of the Meteor AS production

Contents

Abstract **200**

Contents **200**

E.1.1 Figures **201**

E.1.2 Introduction **202**

E.1.3 Concepts **202**

E.1.3.1 End effector 202

E.1.3.2 Workstation 205

E.1.3.2.1 Early workstation concepts 205

E.1.3.2.2 Revised workstation concepts 206

E.1.3.3 Feeding rails 210

1.1.1. Feeding rail for cable connector 210

E.1.3.3.1 Feeding rail for regulator 210

E.1.3.4 Gripping 211

E.1.3.5 Griddle alterations 213

E.1.4 Conclusion **213**

Figures

Figure 53: End effector 203
Figure 54: Concept for fastening screwdriver 203
Figure 55: Concept for fastening camera 204
Figure 56: Early workstation concept 205
Figure 57: Test station 206
Figure 58: Workstation alternative 1 207
Figure 59: Workstation alternative 2 208
Figure 60: Workstation alternative 3 208
Figure 61: Workstation alternative 4 209
Figure 62: Feeding rail for cable connector 210
Figure 63: Feeding rail for regulator 210
Figure 64: Regulator gripping tool 211
Figure 65: Cable connector gripping tool 211
Figure 66: Bottom plate gripping tool 212
Figure 67: Power bracket concept 213

E.1.1 Introduction

The elaboration phase in our project consisted of researching and gathering different concepts. It is important to explore different possibilities when it comes to finding the right solutions, and thus discussion between project members will occur. In this project we need an end effector which will satisfy the need of gripping and fastening components. This end effector will also hold the camera, which gives it additional requirements. The robot is supposed to work in a station that is designed for robot-human cooperation. This demands a station that is designed to be optimal and both human and robot friendly. We need to consider what tasks robot can do, what components need to be provided in a special way for robot to work, what the human assembler can do, in what order the robot-human will work and so on. Some of the components demand a special way to be gripped or placed, and so a special gripping tool is needed besides the gripper. In next chapter we will discuss some of our concept solutions for some of our needs.

E.1.2 Concepts

E.1.2.1 End effector

Since we decided to use a screwdriver and a gripper, we need an end effector to hold these two tools. It is important that the end effector doesn’t limit the movement of the robot to the point where the robot will not be able to perform the intended tasks. There are two main solutions for having this kind of multi-tool end effector: an end effector holding all tools at once or an end effector with a tool changing mechanism. Both of the solutions can be viable if implemented and designed in right way. One of the cameras will most likely be fastened to the robot. Both of the end effector solutions provide with different camera fastening possibilities and limitations. Another thing to consider when deciding which concept to choose is the tubes and cords. The gripper needs air tubes to provide the air pressure, the screwdriver has a power cord and the camera has several wires to the computer and a power cord.

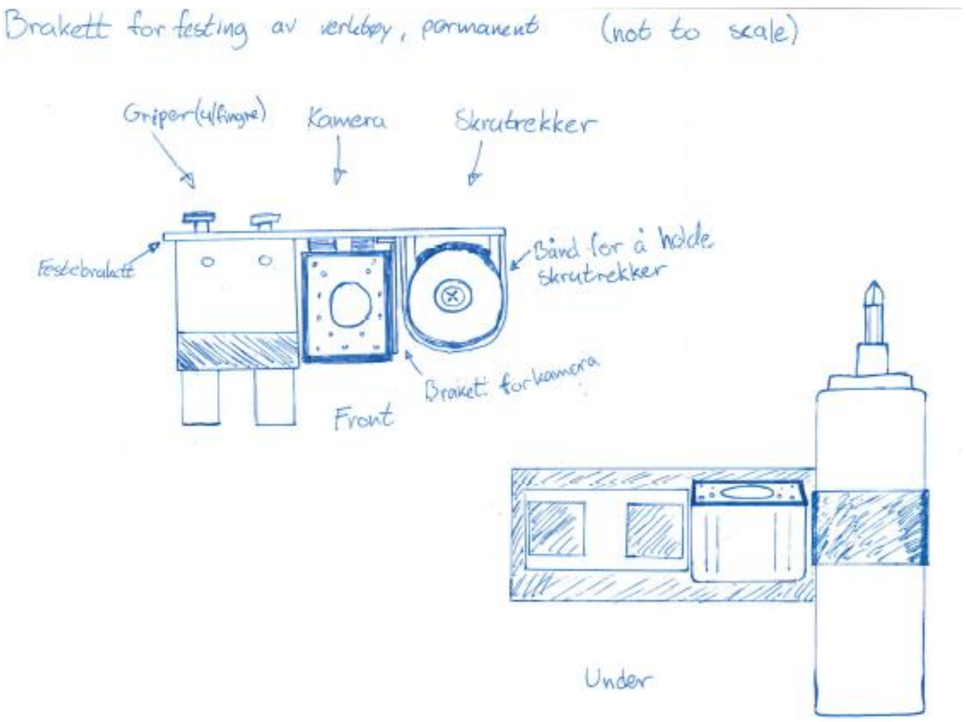

Brakett for festing av verktøy, permanent (not to scale)
Griper (u/fingre)
Kamera
Skrutrekker
Festebrakett
Bånd for å holde skrutrekker
Braket for kamera
Front
Under

Figure 53: End effector

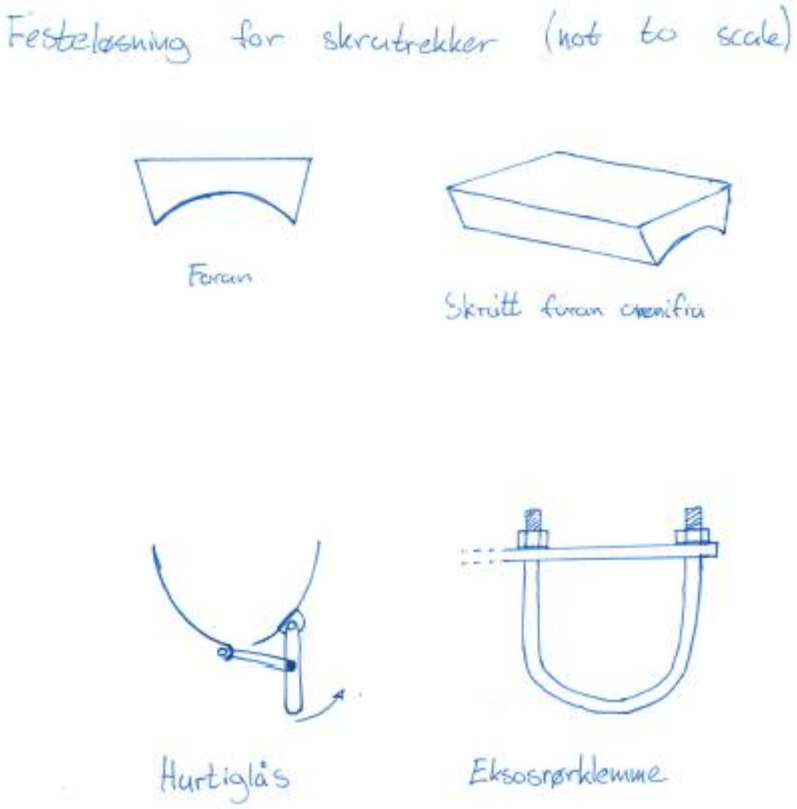


Figure 54: Concept for fastening screwdriver

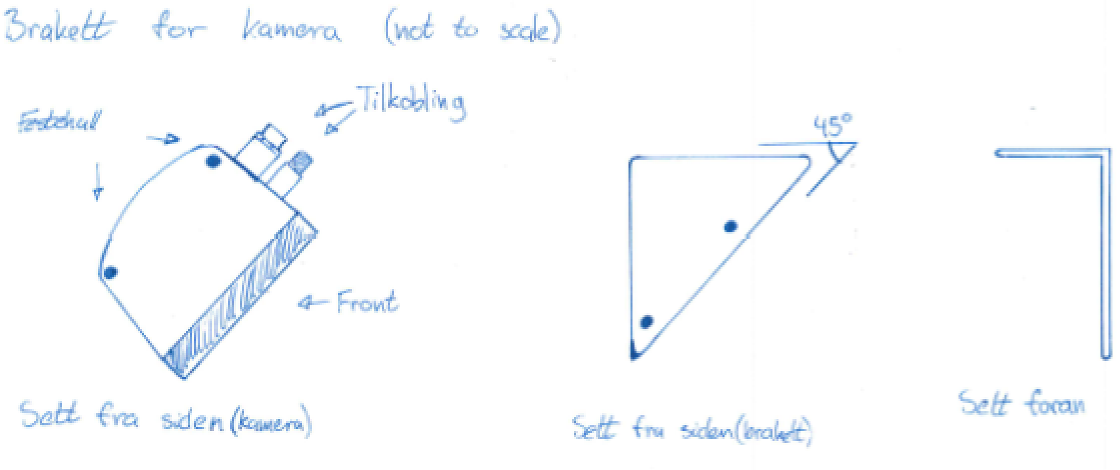


Figure 55: Concept for fastening camera

E.1.2.2 Workstation

The workstation is an important concept for an automation system. It is crucial that the workstation is accurately described, to get a good overview of the cooperation between the assembly system and the operator. We need a detailed breakdown of the entire assembly process to make sure that the operator doesn’t have any downtime. In order to establish this breakdown, we have to understand the robots and the different tools capabilities. Both the robots and gripper has limitations which need to be taken in consideration when designing the workstation. Additionally it’s important that the workstation has a constant flow of components and parts to maintain the workflow. This can be accomplished by the use of feeding rails, which will be described in the next chapter.

E.1.2.2.1 Early workstation concepts

In the beginning of the project we came up with a few different concepts for the workstation. The main concept we came up with is the one illustrated in Figure 56. This is based on using one UR10 to pick up, place and attach different components. This is done on a turntable which is controlled manually by an operator. First an operator places a prepared griddle (point 1) onto the turntable (point 2), and turns the griddle to the automation system. Here the robot picks up, places and attaches the cable connector and the regulator. The griddle is then turned to the operator, which attaches the power inlet and connects the electrical components. Then the griddle is turned back to the automation system again, which attaches the bottom plate of the griddle. Finally the operator removes the griddle (point 3), tests it and packs it. Figure 57 illustrates the test station we intend to use for testing purposes at school.

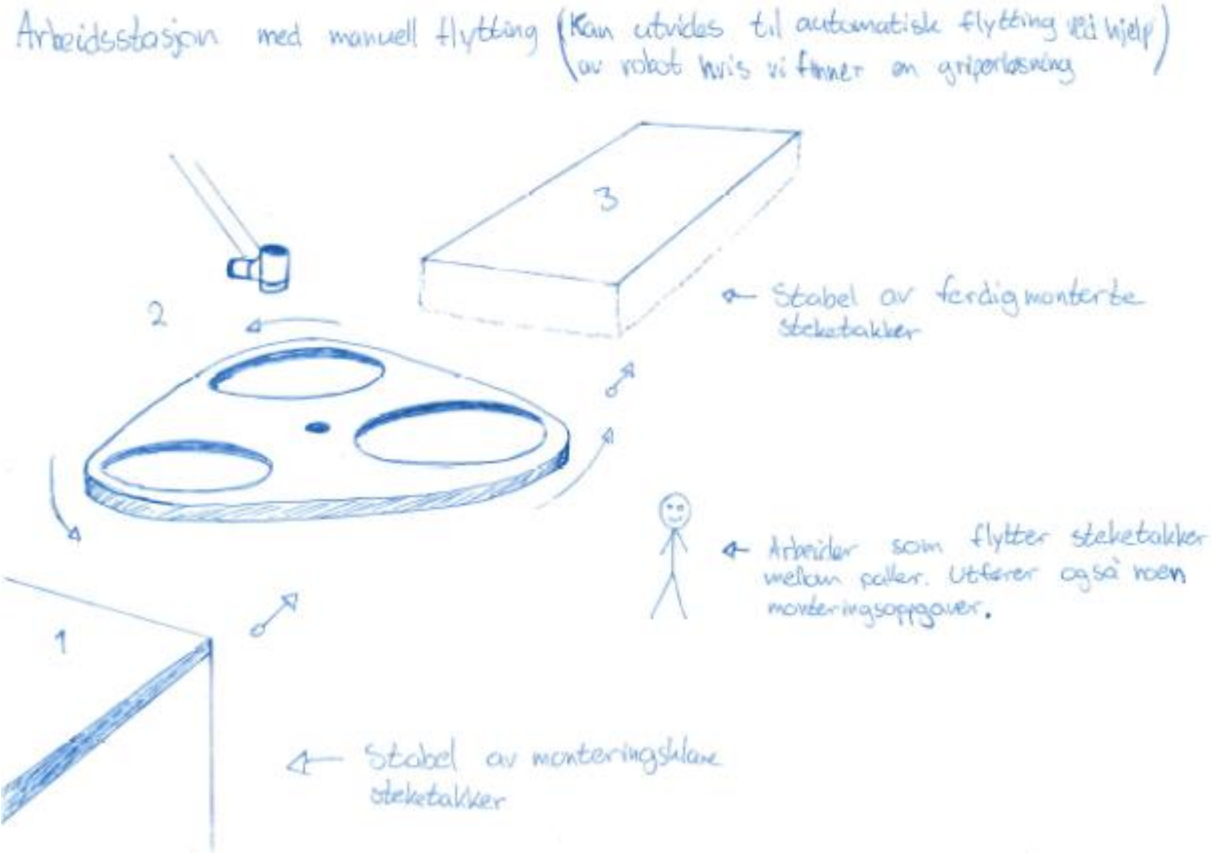

Arbeidsstasjon med manuell flytting (Kan utvides til automatisk flytting ved hjelp av robot hvis vi finner en griperløsning)
3
2
Stabel av ferdigmonterte steketakker
Arbeider som flytter steketakker mellom paller. Utfører også noen monteringsoppgaver.
1
Stabel av monteringsklare steketakker

Figure 56: Early workstation concept

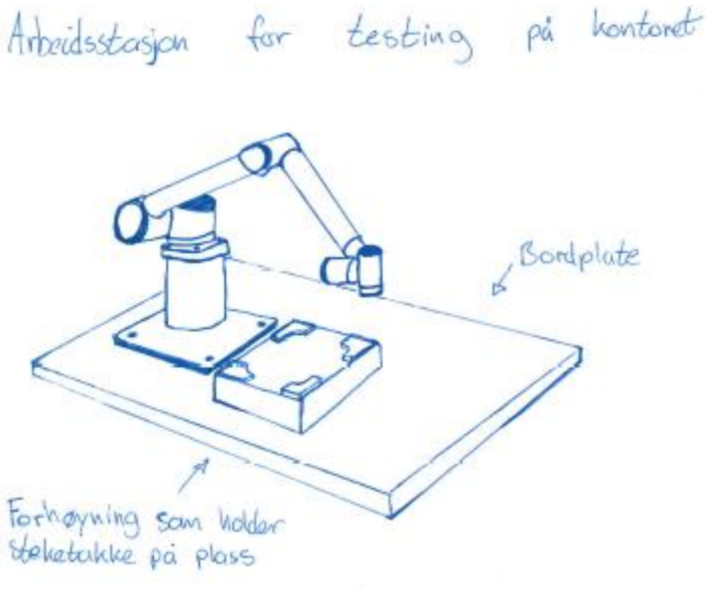

Arbeidsstasjon for testing på kontoret
Bordplate
Forhøyning som holder steketakke på plass

Figure 57: Test station

E.1.2.2.2 Revised workstation concepts

Later on in the project, an opportunity to use two robot arms for the assembly process arose. This became a possibility because the UR3 got released from Universal Robots. We discussed this solution with our customer, and he agreed that using two arms could become a reasonable solution. To make sure that this could be a suitable solution for us, we made a document called *Assembly process alternatives.* This document contains pros and cons for four different solutions for our project task. Three of the alternatives revolved around using one robot and one alternative which revolved around the use of two robots.

We drew one workstation sketch for each alternative, to get an overview of the different stations. All the alternatives are based on the early concept from figure 4. The automation system shall locate screw holes and attach components in every alternative, but they differentiate in how much of the assembly line that gets automated. To get a detailed breakdown of what each alternative consists of, see the document *Assembly process alternatives*.

Each sketch contains a square worktable with a turntable in the middle. The turntable can hold two griddles at once, and they are held in place by a casing contraption which is described in the document *Product Development Document Contraption for the Casing M-CA*. On the worktable we have the robot(s) and screw feeders on one side of the table and the operator on the other. The components are located on different sides of the table, depending on whether it is the operator or the automation system that has to place them on the griddle. The power inlet ("E-PI") has to be placed by the operator, since the automation system cannot attach it independently. Additionally there are three pallets used to store prepared griddles ("Ny ST"), store the bottom plate ("BP") and store completed griddles ("Ferdig ST"). These are placed on different sides of the table, depending on whether it is the operator or the automation system that has to move components to and from the workstation.

The first alternative (Figure 58) we created consisted of only having the automation system attach components to the griddle. In this scenario the end effector on the robot consists of a screwdriver and a camera. The operator has a big role in the assembly line, since he has to place and remove the griddle, place the cable connector, the regulator (with jig) and the bottom plate, attach the power inlet and connect the electrical components. This alternative is relatively simple and easy to realize, but it does not automate the assembly tasks we wish to fulfil. It also depends on the use of a jig for the regulator, which is not very efficient, since more time has to be spent to remove the jig.

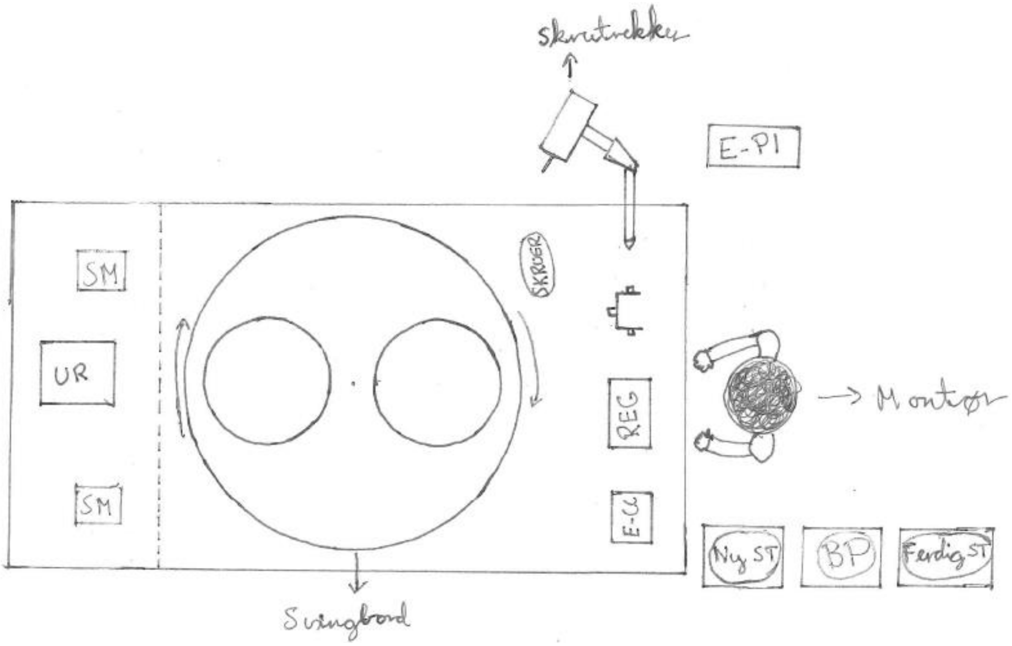


Figure 58: Workstation alternative 1

The second alternative (Figure 59) builds further upon the first alternative, in that it automates a larger part of the assembly line. In this scenario the end effector on the robot consists of a screwdriver, a camera and a gripper. The operator still has to place the cable connector and the regulator (with jig), but the automation system can now pick up and place the bottom plate independently. That is why we need a vacuum tool, which the automation system can use to pick up and place the griddle and the bottom plate. This alternative is very similar to alternative 1, but with the addition of a vacuum tool, more of the assembly line is automated. However, this alternative is also not very efficient, since it depends on the use of a regulator jig.

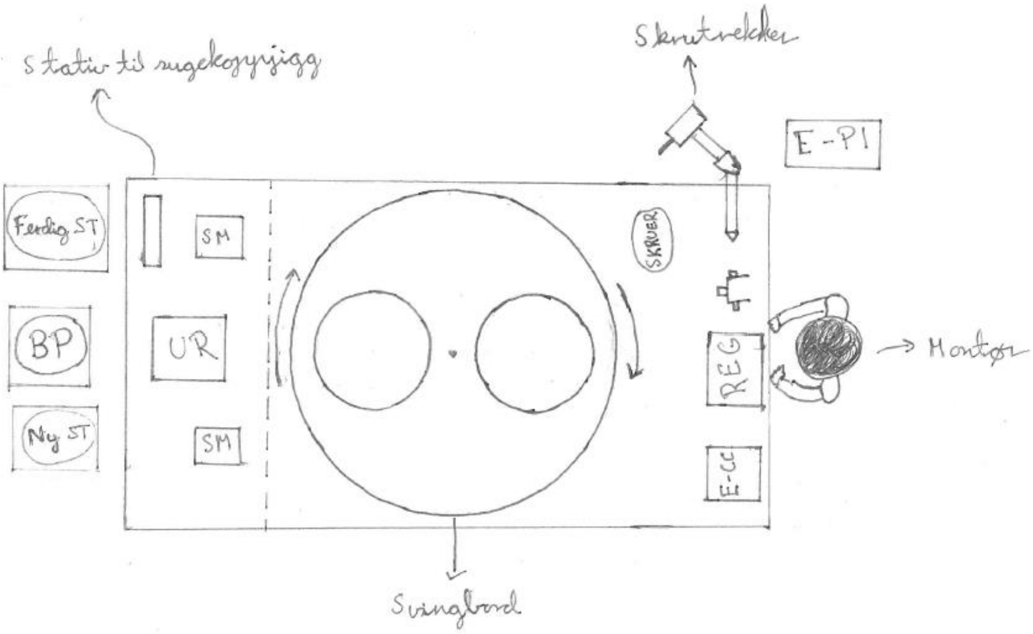


Figure 59: Workstation alternative 2

The third alternative (Figure 60) is the optimal alternative for an automation system with only one robot. In this scenario the end effector on the robot consists of a screwdriver, a camera and a gripper. The operator's tasks are placing and attaching the power inlet and connecting the electrical components. The automation system can now pick up and place components and their corresponding jigs. The cable connector and the regulator are dependent on feeding rails to maintain the automation systems workflow. Most of the assembly line is automated in this alternative, but the use of a regulator jig still leads to inefficiency and inaccuracy.

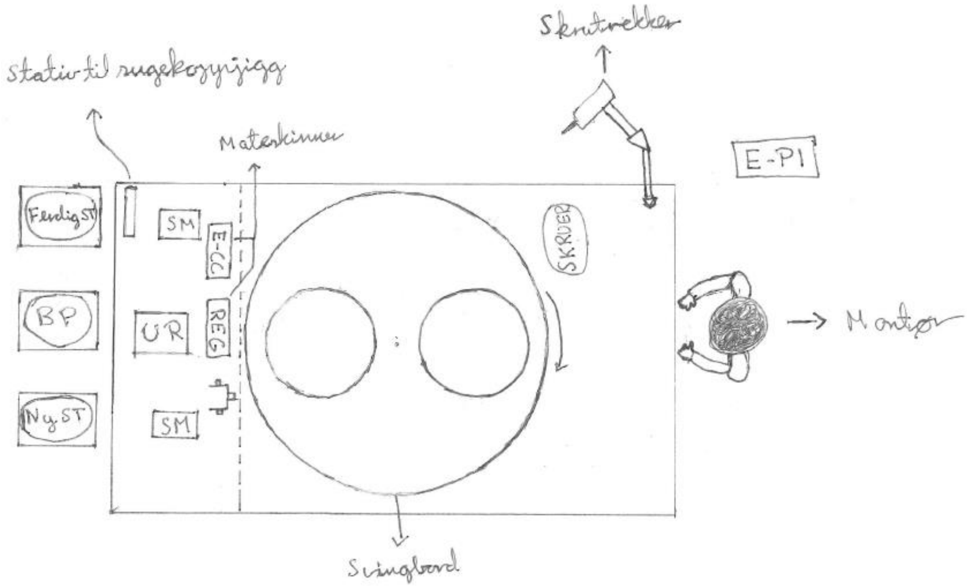


Figure 60: Workstation alternative 3

The fourth and final workstation alternative (Figure 61) we came up with is the alternative with two robots. In this scenario the end effector on one of the robots consists of a screwdriver and a camera, while the other robot consists of a gripper. The automation system performs the same tasks as in alternative 3, however it is not depended on jigs. Therefore the system can assemble a griddle faster, because it does not have to remove the jig every time a regulator is attached. Instead the robot, with the gripper attached, places and holds the different components, while the other robot fastens the screws. Additionally we will not have to worry about the precision of the jig placement, which is an important factor in the previous alternatives. This is why we find this to be the best suited automation solution, and we plan on basing our workstation on this alternative.

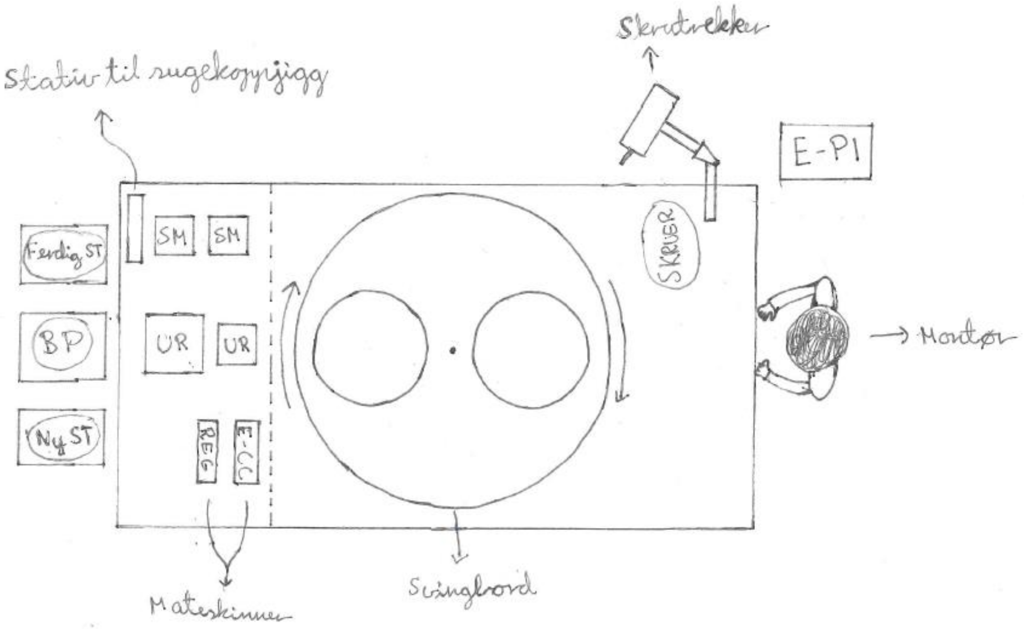


Figure 61: Workstation alternative 4

E.1.2.3 Feeding rails

It’s important to design feeding rails to make sure the automation system does not run out of available components. These are rails that the operator prepares before the automation system starts assembling griddles. Its main purpose is to make sure a component can be picked up at the exact same place every time.

1.1.1. Feeding rail for cable connector

In figure 10 we see the main concept of the feeding rail for the cable connector. It’s made of an angled metal rail that is bent to hold multiple cable connectors in a line. We want to be able to stack the cable connector in such a way that when the automation system picks a cable connector from the bottom of the rail, the next cable connector slides down to the end of the rail.

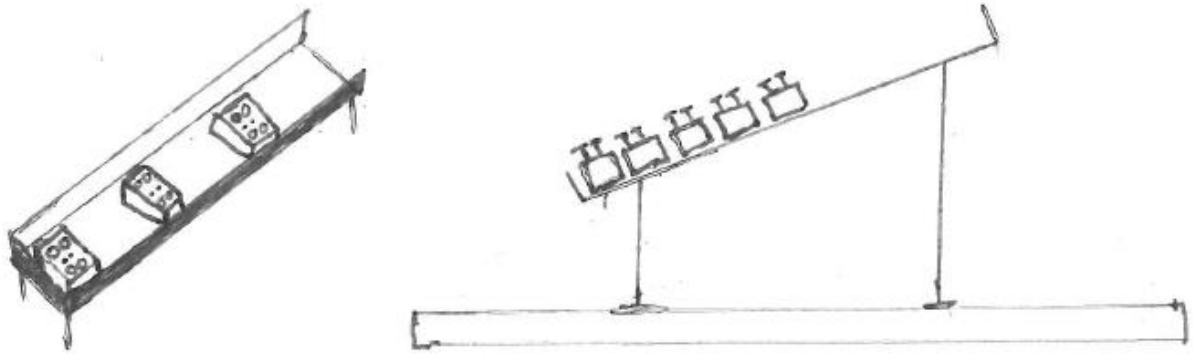


Figure 62: Feeding rail for cable connector

E.1.2.3.1 Feeding rail for regulator

Figure 63 describes the main concept of the feeding rail for the regulator. It is made of a bent, angled metal rail that can hold multiple regulators in a line. The regulators will be stacked with the wires pointing up in the air. The rail has a line cut out in the middle so the regulator pin can point out on the bottom of the rail. Just like the cable connector rails, we want to pick up the regulator from the bottom rail, and have the next regulator slide down to the end of the rail.

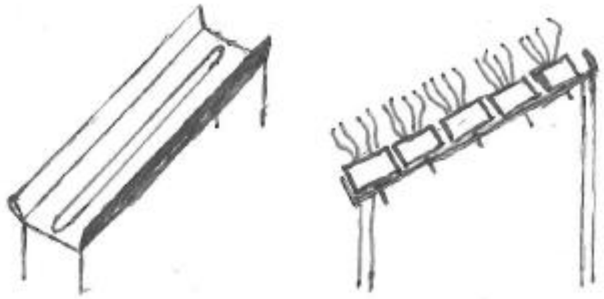


Figure 63: Feeding rail for regulator

E.1.2.4 Gripping

Grippers usually have jaws designed for gripping a specific object. In this project however, it is supposed to grip several objects with different dimensions. This raises the difficulty level of the jaw design. The solutions in our case will be a specific extended gripping tool for some of the components. This will solve the requirement for gripping different dimension objects in a way where we can still have a simple jaw design.

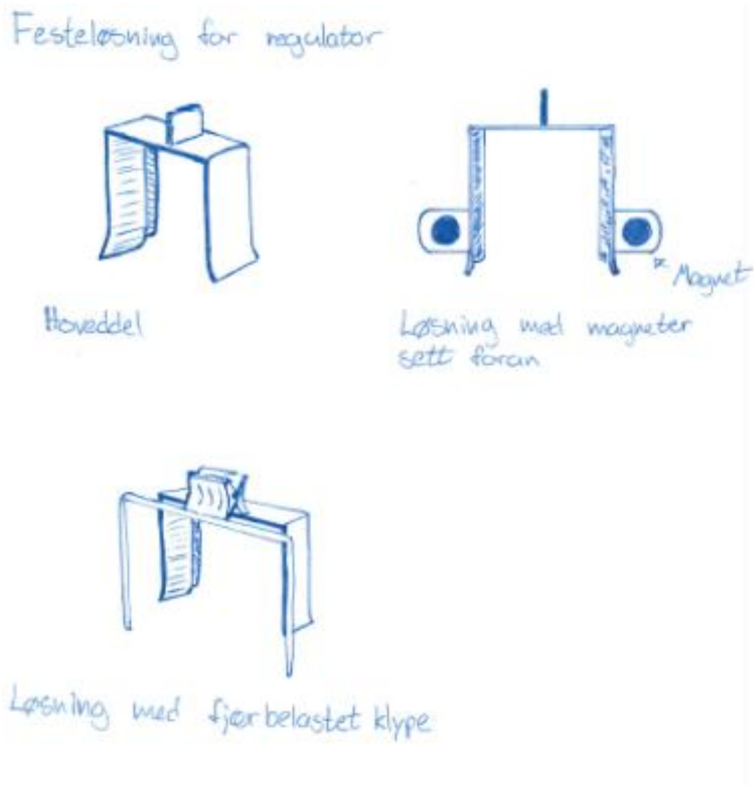

Festeløsning for regulator
Hoveddel
Magnet
Løsning med magneter sett foran
Løsning med fjærbelastet klype

Figure 64: Regulator gripping tool

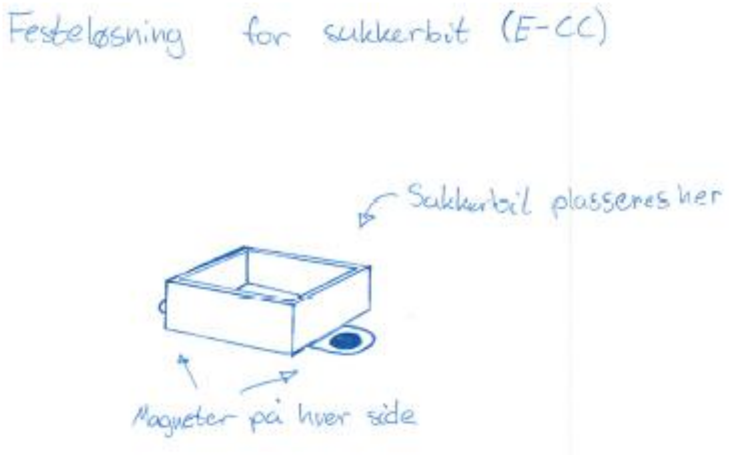


Figure 65: Cable connector gripping tool

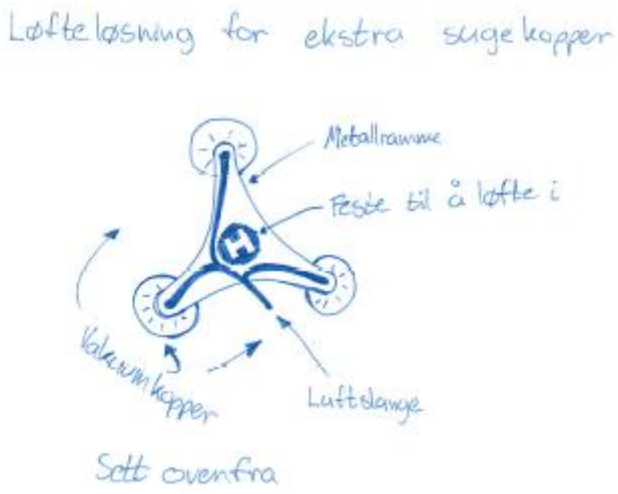


Figure 66: Bottom plate gripping tool

E.1.2.5 Griddle alterations

For either automation or optimization reasons some of the parts inside the griddle need to be altered. It might consist of re-designing the component, finding a new component to replace the old one or finding a new way to fasten the component.

A significant piece of the griddle is the power bracket, which is where the cable connector is attached. Its main purpose is to distribute heat coming from the heating element. We want to be able to make the automation system attach the cable connector to the power bracket. To achieve this, there are a couple of factors that needs to be taken into account. We have to modify the power bracket, so that it keeps the cable connector in place, while the automation system fastens it. It's also important that the power bracket has enough strength, so that it does not deform when the automation system fastens the cable connector. Figure 67 shows our concept for the improved power bracket. The sides of the bracket are bent upwards to make sure the cable connector stays in place.

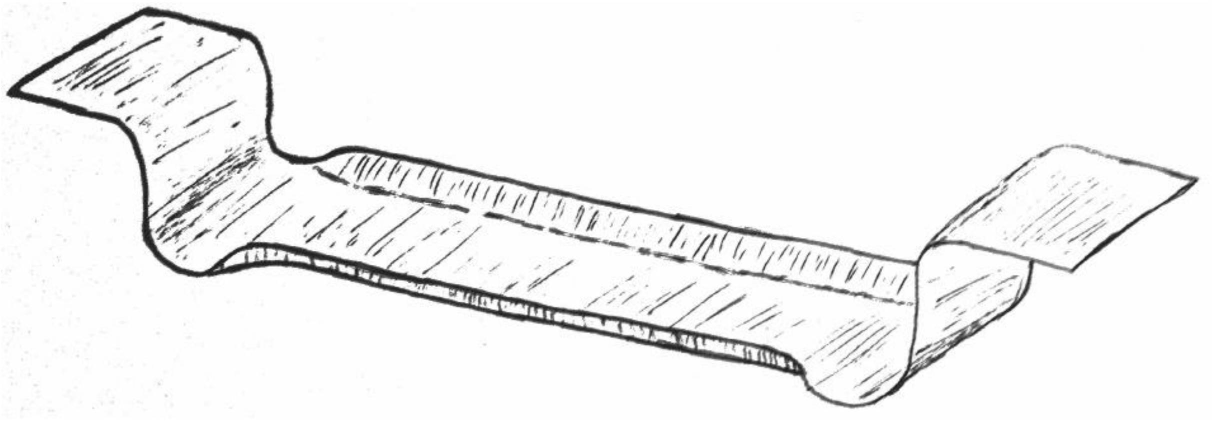


Figure 67: Power bracket concept

E.1.3 Conclusion

Only the most viable solutions will advance further into the next phase, where we will present them to Meteor and Mektron. With their cooperation we will find the most suited solution for each of the cases, make some minor changes if needed and proceed with the implementation. There is uncertainty when it comes to how far along the project we will come. We expect to finish the end effectors and gripping solutions. The workstation however might not reach the point where we actually set it up at Meteor.

F. Product development documents

F.1 Regulator Jig

Abstract

This document contains background information and a detailed description of the development of the jig for the regulator E-REG. It addresses the use of materials and choice of design.

Contents

Abstract 214

Contents 214

Figures 215

F.1.1 Introduction 216

F.1.1.1 The component 216

F.1.1.2 Challenges 216

F.1.1.2.1 Power bracket 216

F.1.1.2.2 Gripping of the regulator 216

F.1.1.2.3 Keeping component in place 216

F.1.1.2.4 Durability 217

F.1.2 Development 217

F.1.2.1 Early phase 217

F.1.2.1.1 Design 217

F.1.2.1.2 Fastening to the casing 218

F.1.2.1.3 Maintenance 218

F.1.2.2 Prototype 1 218

F.1.2.2.1 Sketching and modelling 218

F.1.2.2.2 Producing the prototype 219

F.1.2.3 Prototype 2 219

F.1.2.3.1 Changes 219

F.1.2.3.2 Material consideration 220

F.1.3 Conclusion 220

F.1.3.1 Further development 220

F.1.3.2 Thoughts 220

F.1.4 References 221

Figures

Figure 68: The regulator 216
Figure 69: Regulator positioned to the casing 216
Figure 70: Regulator jig with a clamp 216
Figure 71: Regulator jig with magnets........ 216
Figure 72: Regulator jig design 217
Figure 73: 2D jig design, drawing 218
Figure 74: 3D jig design 219
Figure 75: Prototype 1 219
Figure 76: Prototype 2 219

F.1.1 Introduction

F.1.1.1 The component

One of the components that our system is supposed to fasten is a regulator. The dimensions of this regulator are as follow: height 4.8cm, width 4.5cm and front to back 2.2cm. It has a 6 mm diameter pin sticking out in the front, which is inserted through an 8 mm hole in the casing. If placed correctly, the top of the regulator and the uneven top surface of the casing will have a maximum clearance of 2 mm. The fastening of the regulator is an easy task when performed by an assembler, but several challenges emerge when the process is supposed to be performed by a single robot arm.

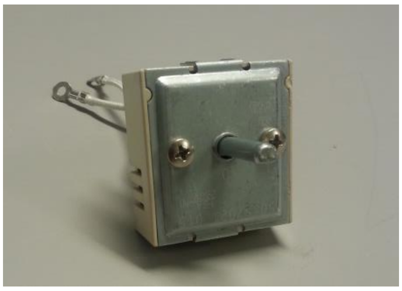


Figure 68: The regulator

F.1.1.2 Challenges

The mounting of the regulator with one robot arm generates several difficulties. Our system consists of one robot arm that has to operate a gripper, a camera and a screwdriver. Since the gripper has to let go of the component after placing it so that the screwdriver can fasten it, there is nothing to hold the component in place and it falls out of its position when trying to fasten it. To create a good solution, we had to analyse the assembly process thoroughly.

F.1.1.2.1 Power bracket

The power bracket is located close to where the regulator is supposed to be placed. The cables connected to the regulator touch the power bracket when lowering the component between the power bracket and the casing causing the regulator to tilt forward. To avoid the tilt, we have to either bend the cables to the side or create something so strong that it overrides the effect of the cables.

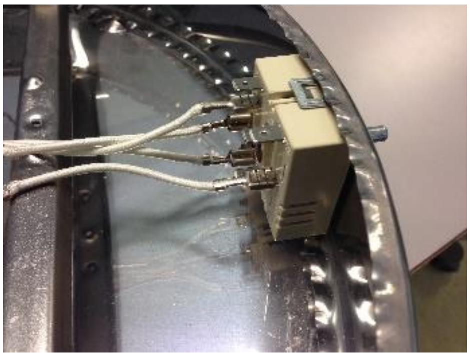


Figure 69: Regulator positioned to the casing

F.1.1.2.2 Gripping of the regulator

The gripping of the regulator has to be consistent. The pin of the regulator has to be located at the same coordinates and orientation relative to the system every time. The system “knows” where the pin is, but if the pin is not where it is supposed to be, the system will fail to insert the component. This requires high precision when it comes to feeding of the regulator to the system.

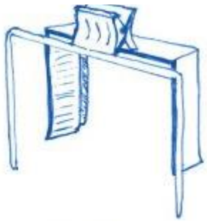


Figure 70: Regulator jig with a

F.1.1.2.3 Keeping component in place

Another difficulty is holding the regulator in place until the system has fastened it with screws. The regulator does not stay in position without extra help, which leads to the need of a jig. This jig needs to be designed in a way that would satisfy the requirements above and at the same time be easy to attach and detach from the regulator.

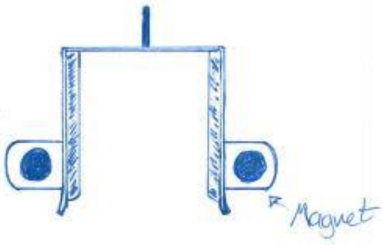


Figure 71: Regulator jig with magnets

The gripper needs a way of gripping the jig that will give good support, control and a consistent grip. It is important that this gripping point does not come in the way in the process of placing the component in the fastening position.

Provided we have a good grip on the jig and the jig is holding the regulator, the system is then supposed to place the jig with the regulator into the fastening position. When this is performed without any interference, the jig is supposed to hold the regulator in place until the system has fastened it. Magnets or a clip device (see figures 3 and 4) can do this. Both solutions have their advantages and disadvantages which will be discussed further in this document.

F.1.1.2.4 Durability

When it comes to durability and production, material choice for the jig is important. The functionality of the jig should not be affected during the assembly process for a specific period of time. Let us set the minimum lifespan of the jig to be 2 years. The number of griddles produced per year is at average around 10 000. This means the jig with a lifespan of 2 years should endure 20 000 attachments and detachments from the regulator. Since the jig will not be in mass-production, the easiest and cheapest way to make it is by cutting it out and forming it at the factory itself. Meteor has a lot of experience working with sheets of metal and this sort of material could be considered in this situation. Making a jig out of a material that is used in the factory gives the benefit of an easy maintenance. This means that in case the piece loses some of its functionality, it can be easily repaired at the spot.

F.1.2 Development

F.1.2.1 Early phase

F.1.2.1.1 Design

The stability of the regulator inside the jig is important when it comes to both transporting the regulator and placing it in the fastening position. This is why, when designing the jig, we had to determine how it would grasp the regulator and what points are crucial for the stability.

The best way to grasp the regulator is by having a jig that embraces the shape of the regulator. Therefore, we decided to use the top, sides and back of the regulator as support for the fundamental structure of the jig.

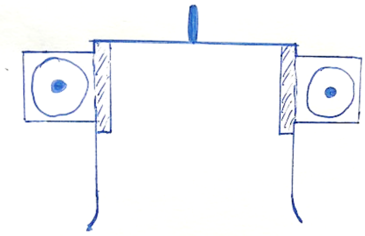


Figure 72: Regulator jig design

The gripping point for the gripper will be located at the top of the jig to give an easy access for the gripper. The design of this gripping point should comply with the gripper jaws, so that gripping consistency and control over the jig is provided.

In case the front of regulator bump into the power bracket or something else while being transported, it is important to ensure that the orientation of the regulator inside the jig is unaffected. Two bent sides, shown as shaded squares in figure 5, on the back of the jig should secure it from rotating backwards in case of a small collision.

The stability of the regulator is irrelevant if the jig is not capable of holding the component. The task of holding the component is designated to the sides of the jig. The tension caused by deformation in the material from inserting the regulator in an opening narrower than the size of the component itself, gives the jig the capability of holding the regulator. The size of deformation must be considered since too big deformation could break the material or create a tension that would lead to difficulties when attaching or detaching the jig.

F.1.2.1.2 Fastening to the casing

To hold the jig with the regulator in the position for fastening, solutions with both magnets and clamps were presented. The solution with magnets was chosen because it was smaller, simpler and more suitable for automation than the clamp solution. The magnets need to be positioned so that they have good contact with the casing. Two "wings" were designed on the sides of the jig where the magnets will be attached. These "wings" will face towards the casing so that they are parallel with the casings wall. The strength of the magnets is important to consider. Magnets chosen for this jig should not be too strong because it would make it difficult for the gripper to detach the jig from the casing, which could break the jig or damage the casing. Too strong magnets could also lead to imprecise placing of the regulator, since the magnetic force could pull the jig towards the casing before it has been placed in the right position.

On the other hand, these magnets should not be too weak either. They are supposed to hold the jig in place until regulator is fastened. During the fastening process, the jig will be subjected to a horizontal force and a rotational force from the screwdriver. The horizontal force is applied the entire time from the insertion of the screw until the screw is fully tightened. The rotary force is low while the screw is being fastened and gets high when the screw hits the point where it is tightened. Weak magnets will fail to hold the jig in place successfully. Thorough testing and consideration must be conducted to decide the magnet strength.

F.1.2.1.3 Maintenance

Another point to consider is the maintenance of the magnets. The processing of the metal material leaves swarf in the production area. These metal particles can collect onto the magnets and decrease their effect. There are several solutions to this problem. One of them is to use electromagnets that can be turned on and shut off manually. This would solve the problem of swarf on the magnets, but the cables sticking out of the magnets would possibly lead to several difficulties and make the process more complex than necessary. Since our goal is to make it as simple as possible, this solution was rejected by the group. Another way to remove the swarf is by using a cloth of some sort and cleans the magnets when needed. This is neither time consuming nor difficult, therefore it was voted the best solution.

F.1.2.2 Prototype 1

F.1.2.2.1 Sketching and modelling

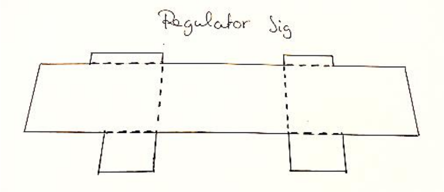


Figure 73: 2D jig design, drawing

The design of the first prototype started with a sketch. The dimensions and bending lines were calculated and drawn. The shape of the flat jig was as shown in figure 6. After the sketch was done a 3D drawing was made in SolidWorks. SolidWorks has a sheet metal function that takes into account different forming factors like bending radius, different bending positions and the K factor of the bending machine. We must take into consideration different forming outcomes and adjust the design so that the product that we get matches our needs. After the 3D model was made, we used the schools workshop to create the first prototype.

F.1.2.2.2 Producing the prototype

When producing the first prototype, we used 1mm thick sheet metal. We started by exporting the outline of the 2D drawing on to the metal sheet and then went on to cut it out. The piece we ended up with matched the drawing, which meant we were successful so far. The next step was to make the bends on the selected lines. The bending machine in the school's workshop is operated by hand, which meant we had to verify correct piece placement and bending angle by eye.

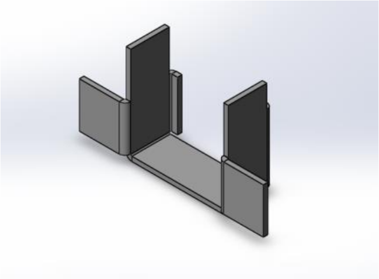


Figure 74: 3D jig design

Because of the limited functionality of the bending machine, we were not able to complete all the bends with the machine. To be able to complete all the bends in the different directions with a bending machine, it has to be possible to remove parts from the upper blade. To solve this, we started some of the bends for the model and then bent them back. This was done to create a bending line so that it could easily be bent at the right place later on by hand after completing the other bends with the machine. Using this technique, we managed to make all of the bends.

The final step was to create the holding mechanism for the regulator. The jig already had a tight fit on the regulator. Bending the sides halfway down inwards and then further down bending outwards, created an opening both wider and narrower than the size of the regulator. The big opening made it easy to slide regulator in and the narrow opening created tension in the material when inserting the regulator so that it holds the regulator in place.

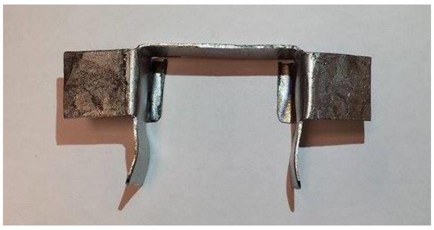


Figure 75: Prototype 1

We ended up with a prototype that gave us a lot of information about our design. The whole process was conducted entirely by members of the group, which lead to up close experience with the different problems that came up during the process. Some of the dimensions needed to be altered to make the regulator fit into the jig even better. There is a lot of uncertainty around the holding mechanism. It needs to be a viable solution with a low wear factor. This means the jig should be able to attach and detach from the regulator and the casing many times without affecting its functionality. This can be ensured with the right design of the holding mechanism and material choice.

F.1.2.3 Prototype 2

F.1.2.3.1 Changes

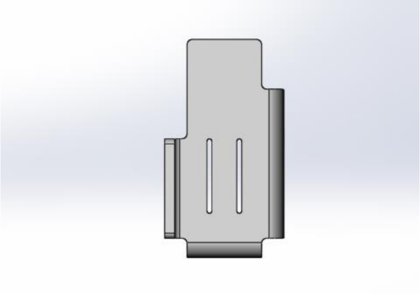


Figure 76: Prototype 2

For the second prototype, we upgraded the holding mechanism on the sides of the jig. It was re-designed to give better stability and to provide a more reliable grip. The new design now consists of two holding points on each side instead of one. The upper one will be made by bending the piece of material between the two parallel holes and the lower one will be as before. The previous prototype turned out to be a bit big, front-to-back dimension, so this prototype was designed to be a bit shorter. Fillets to the corners were changed to make the bending process easier. This also makes the piece more adjustment-friendly when it comes to using equipment with limited functionality.

F.1.2.3.2 Material consideration

An important factor to consider is the material used for this jig. Our design depends on the modulus and elastic limit of the material. We are considering using materials used at Meteor to keep the cost of the jig to a minimum. They have a variation of sheet metals made out of iron alloys. Therefore, metal and especially steel properties are important to consider. For metals, it is common to have high modulus and low elastic limit. Since our holding mechanism depends on the materials ability to stay within the elastic deformation range, the stress applied to the jig should be under the materials yield strength limit. Elasticity works differently for different materials. When it comes to metals, the atomic lattice change size and shape when forces are applied. For rubbers and other polymers, elasticity works by stretching polymer chains by the forces that are applied. When the forces cease, the materials recover their original states. Even when the deformation is elastic, the component never truly recovers completely.

The reason we also included the elasticity for polymers is the consideration to make the jig out of plastic. The jig could be molded out of a polymer material, for example polyethylene. To produce the jig out of a polymer material would be more costly and would be more suited for mass-production situation. The reason for this is the elastic properties of the material that could work perfectly for our jig.

F.1.3 Conclusion

The regulator jig concept stopped at the second prototype. The project had a turn and a second robot arm was introduced. The second arm removed the need for a jig, since the gripper and screwdriver will be attached to separate robot arms. Now the robot with the gripper can hold the component while the other with the screwdriver other fastens it. Therefore, the jig development was concluded.

F.1.3.1 Further development

The jig is still in an early prototype version and several adjustments and tests are needed to make it a viable solution. Material choice for the jig, stress tests and abrasion tests of the holding mechanism and designing the gripping point are the next steps for further development.

F.1.3.2 Thoughts

The group is satisfied with the results we have achieved to this point. The developing of this jig has given us a lot of information about the difficulties that one can face when trying to automate a two-hand job with a single robot arm. Some jigs are easy to create and operate, but a jig designed for a task like this is more difficult to develop because of the need for high precision and repeatability. Other solutions might be more beneficial, like a jig that is placed by the assembler. This would make the design of the jig less complex and at the same time remove the difficulties of a gripper operating the jig. To use a jig for a task like this is highly ineffective anyway, but it is necessary when there is no other option.

F.1.4 References

1. Wikipedia, Deformation (engineering) [15.05.2015]
   http://en.wikipedia.org/wiki/Deformation_%28engineering%29
2. Wikipedia, Polyethylene [15.05.2015]
   http://en.wikipedia.org/wiki/Polyethylene
3. Wikipedia, Elasticity (physics) [15.05.2015]
   http://en.wikipedia.org/wiki/Elasticity_(physics)
4. Wikipedia, List of Alloys [15.05.2015]
   http://en.wikipedia.org/wiki/List_of_alloys#Alloys_of_iron

F.2 URcontrol

Abstract

This document describes the process concerning development of a program for interfacing with the robot from Universal Robots, which will provide all the functionality needed to control the robot with coordinates given by the vision system. Interface and math regarding camera/vision system is described in their own document. Functionality and dataflow for each function in the UR control code can be found in more detail in the user manual [3].

Contents

Abstract **222**

Contents **222**

Figures **223**

F.2.1 Introduction **224**

F.2.1.1 Automation Cases 224

F.2.1.2 Basic Operations 226

F.2.1.2.1 Fasten screws in holes and through sheet-metal 226

F.2.1.2.2 Pick up and place a component/work piece 226

F.2.1.3 Necessary Functionality 226

F.2.1.3.1 Move Tool 226

F.2.1.3.2 Rotate Tool 227

F.2.1.3.3 Move Tool with Given Force 227

F.2.1.3.4 Change Tool Center Point 227

F.2.1.3.5 Set Outputs 227

F.2.1.3.6 Read Inputs 227

F.2.1.3.7 Get Current Waypoint 227

F.2.2 Development **228**

F.2.2.1 Preparation and Research on Universal Robots 228

F.2.2.1.1 Programming with GUI (local) 228

F.2.2.1.2 Control by Socket 228

F.2.2.1.3 Combining Remote and Local 229

F.2.2.2 Remote Control Script with Socket Interface 230

F.2.2.2.1 Interface 230

F.2.2.2.2 Script with Necessary Functions 230

F.2.2.2.2.1 Basic Moves 230
F.2.2.2.2.2 Moves with Force Control 231
F.2.2.2.2.3 Set and Reconfigure 231
F.2.2.2.2.4 Get Data 232
F.2.2.2.2.5 Make circular moves 232
F.2.2.2.2.6 Expanding Functionality 232
F.2.3 Other 233
F.2.3.1 Rotation of Tool 233
F.2.3.2 Circular Move 234
F.2.4 Conclusion 234
F.2.5 References 234

Figures

Figure 77: Fasten Reflector-plate 224
Figure 78: Place and Fasten Cable Connector 225
Figure 79: Place and Fasten Regulator 225
Figure 80: Bottom Plate 226
Figure 81: Screenshot of PolyScope 228
Figure 82: Print with UR 229
Figure 83: Example of how to open a Socket on UR 229
Figure 84: Extracts 30 Floats from String 230
Figure 85: Extract a pose and move there 230
Figure 86: Movel_force 231
Figure 87: Set_output() 232
Figure 88: Vertical and horizontal tool orientation 233

F.2.1 Introduction

This project consists of two main systems; a robot and a vision system. The robot is delivered by Universal Robots, and is delivered with a complete control-system that can be configured both remotely by TCP and with its local programming interface. Vision hardware is delivered by IFM, which delivers compact cameras where all image-processing is done in the camera itself. The camera can be configured to give digital outputs based on matches in the frame, or give the output as pixel-coordinates by TCP.

We will use the vision-system to locate key positions on a griddle made by Engmark Meteor, and then translate and feed those positions into the robot-system and use them to automate parts of the assembly. Different cases for automation are described in chapter F.2.1.1. To do this, we need an interface between the camera and robot. As mentioned in *Technology Document: Programming of Universal Robots,* the robot controller mainly supports 3 different ways of programming out of the box: local GUI programming, local script programming and script-commands sent by TCP on port 30002. During the development of our control-code for the UR we have used all 3 methods, as described in later chapters. We discovered that it was necessary to run a program on an external unit to interface between the two systems, since URScript has very limited resources when it comes to socket communication.

F.2.1.1 Automation Cases

We plan to automate several aspects of the assembly of the griddle. We wish to automate the following operations:

- Fasten reflector plate
    - Reflector plate is fastened in a bracket by two screws close to the center of the griddle.

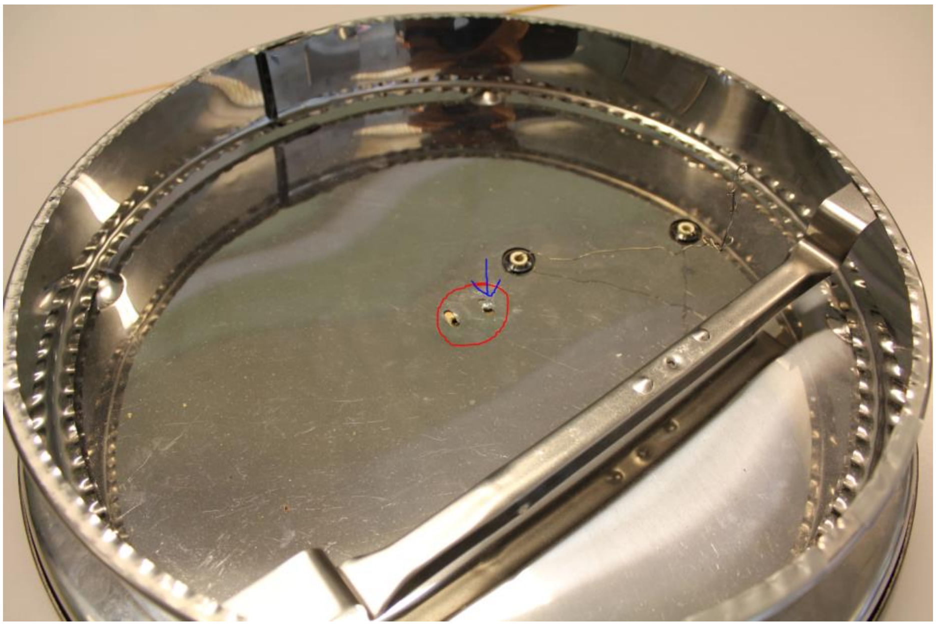


Figure 77: Fasten Reflector-plate

- Place and fasten cable connector
  - Cable connector is placed on a bracket, and fastened with one screw.

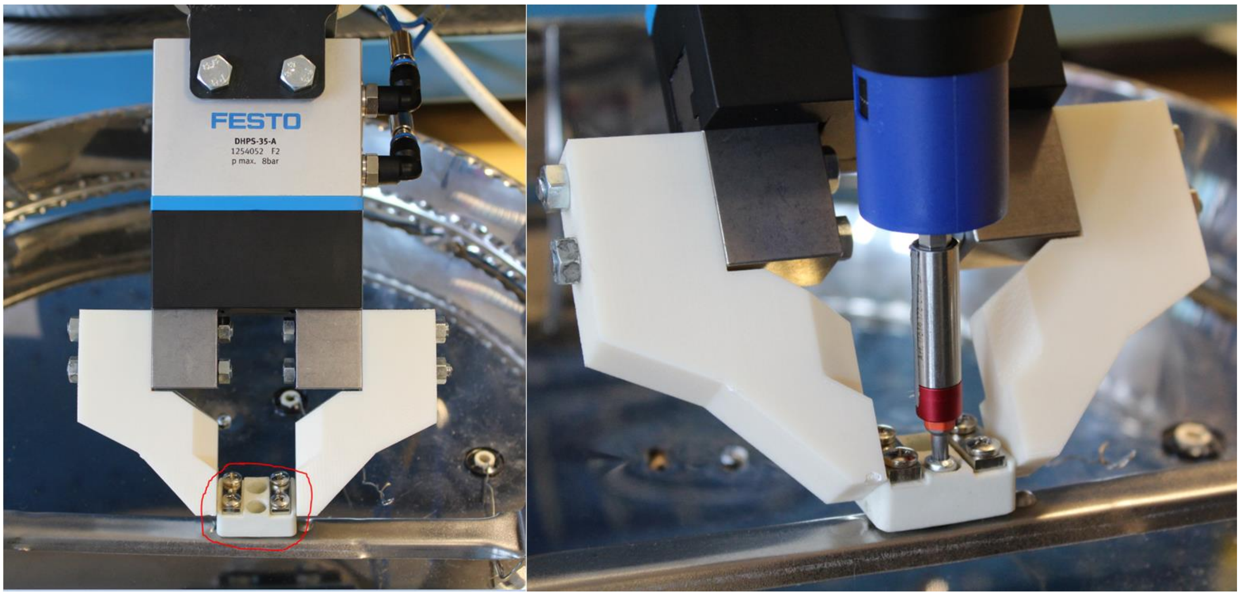


Figure 78: Place and Fasten Cable Connector

- Place and fasten regulator
  - Regulator is fastened with two machine screws with tool oriented parallel with xy-plane (horizontal).

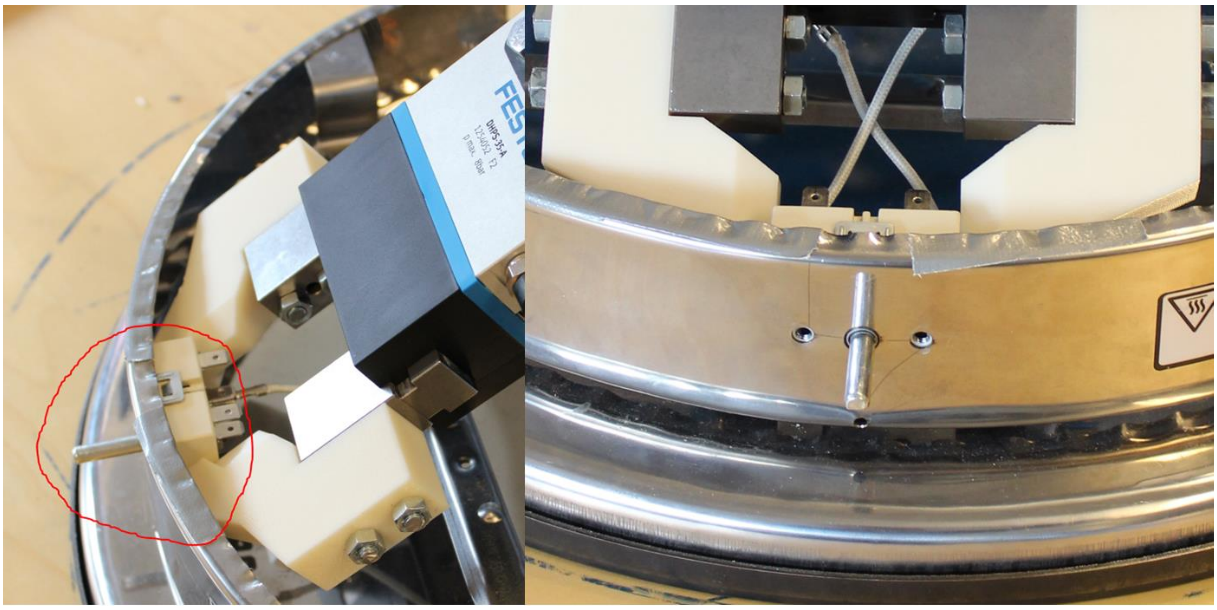


Figure 79: Place and Fasten Regulator

- Place and fasten bottom plate

- The bottom plate is fastened by 4 screws with tool oriented parallel with z-axis (vertical).

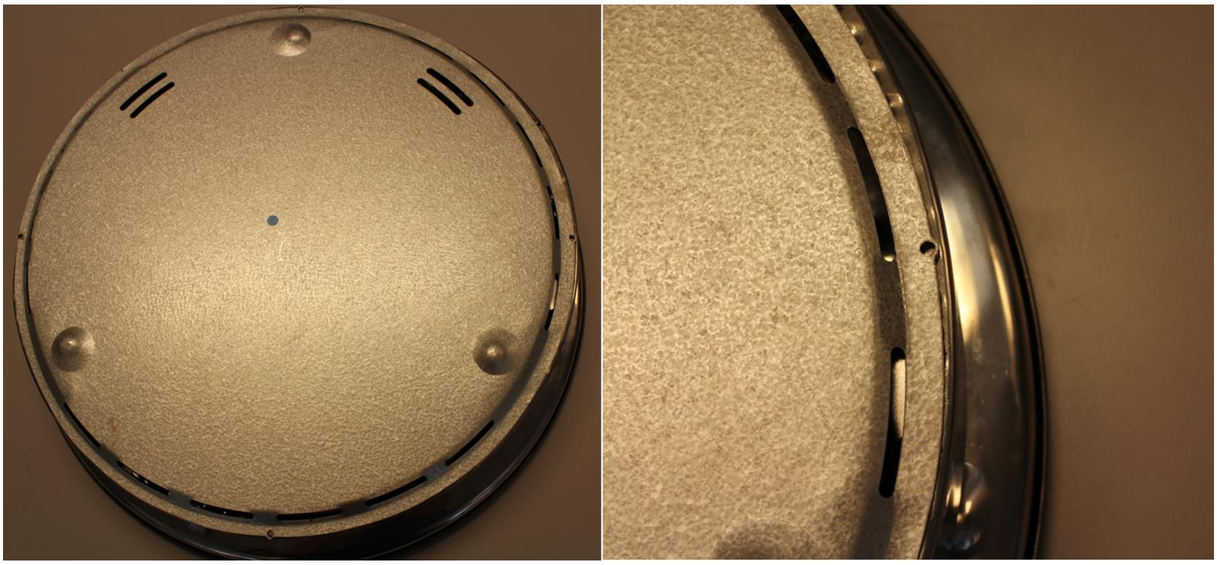


Figure 80: Bottom Plate

It looks like it all boils down to two basic operations: pick and place components, and fasten components with screws. Since each of the griddles won’t be locked in the same static position every time, we also have to be able to “scan” the subject for key positions and adjust our tools accordingly.

F.2.1.2 Basic Operations

F.2.1.2.1 Fasten screws in holes and through sheet-metal

The case that our project owner is especially interested in is making the robot system able to locate holes or other key positions, and fasten screws in that position. In our situation, it means that we have to be able to pick up a screw from a static position, locate key position (e.g. hole), and place the screw in that position. It also means that we have to supply the robot arm with a screwdriver that can be controlled digitally, and apply certain force while the screw is fastened. The griddle we shall assemble, requires fastening of screws with screwdriver in a vertical position pushing down along global z-axis, and also horizontally parallel with xy-plane pushing towards the center point of the griddle.

F.2.1.2.2 Pick up and place a component/work piece

For our system to be functional and useful, it needs to be able to pick up and place key components with a given orientation relative to the griddle.

F.2.1.3 Necessary Functionality

F.2.1.3.1 Move Tool

We need to be able to move the tool from one position to another in order to scan for key positions and perform assembly actions. Universal Robots provides a few different ways to perform a move, see movej(), movel() and movec() in the URScript manual [2].

F.2.1.3.2 Rotate Tool

Most of the time, the tool will be required to work oriented perpendicular at the xy-plane. The regulator, however, requires to be fastened from an angle parallel with the xy-plane (see Figure 79). We must therefore be able to rotate the tool.

F.2.1.3.3 Move Tool with Given Force

Some operations that the automation system shall perform require a certain amount of applied force. In the case of fastening a screw, the robot will have to apply a force on the screw to ensure that the screwdriver follows the screw as it rotates downwards. Moving towards a waypoint until a force is met, might also be useful in applications where components are placed on top of each other. The system will then be able to "seek down" until it hits something and then pick it up instead of having to calculate the exact position of the next piece. The bottom-plate will be placed in stacks where this should prove useful.

F.2.1.3.4 Change Tool Center Point

The automation cases require the use of different tools (mainly screwdriver and gripper). In order to accurately use different tools, we need to be able to reconfigure TCP (Tool Center Point) in the middle of a running sequence.

F.2.1.3.5 Set Outputs

The controller from Universal Robots includes several outputs, both digital and analogue. These will be used to control external equipment like pneumatic valves and screwdriver.

F.2.1.3.6 Read Inputs

The controller from Universal includes several inputs, both digital and analogue. Some operations might require the use of external sensors.

F.2.1.3.7 Get Current Waypoint

In order to convert coordinates from the camera to real-life coordinates relative to the robot base, we need to know the current position of the camera.

F.2.2 Development

F.2.2.1 Preparation and Research on Universal Robots

Once we knew that we were going to use a robot system from Universal Robots, we began research on how to program it. Universal Robots supplies a user manual [1] that describes most of the robots functionality, and a script manual [2] that describes the dedicated programming language, URScript. We installed a simulator running PolyScope, the graphical user interface that comes with the robot. This allowed us to practice on programming (and in general controlling) the robot system even before it arrived. We received the robot the 20th of February, and after only two hours we had our first program up and running.

F.2.2.1.1 Programming with GUI (local)

Universal Robots offers a graphical interface for programming and controlling the robot. This interface is called PolyScope. It offers an easy to use and intuitive method of how to program the robot. PolyScope runs on a pendant (touch-screen) connected directly to the control-machine, and can also be run as a simulator on any linux-machine. For more information on how to use the GUI, see the user manual from Universal Robots [1].

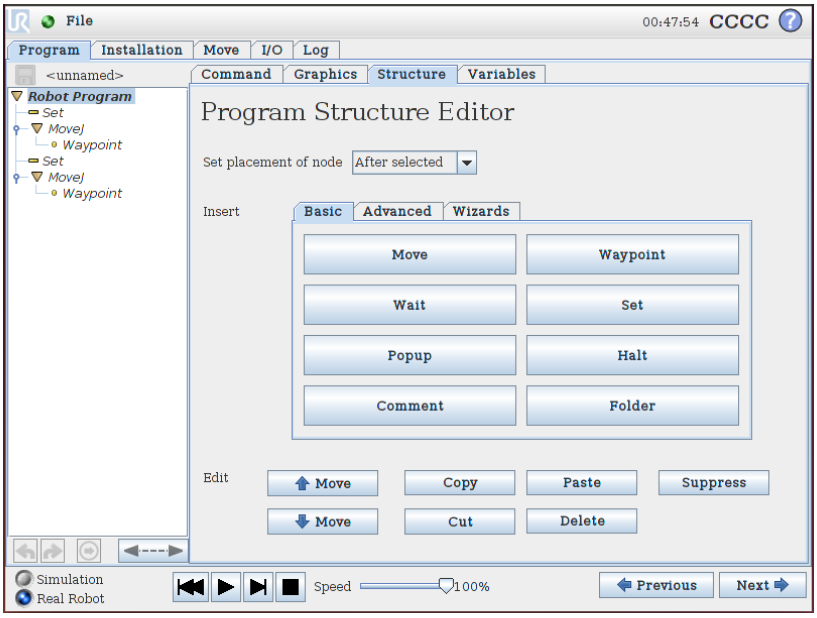


Figure 81: Screenshot of PolyScope

F.2.2.1.2 Control by Socket

To be able to use data from a vision system to control the robot, it needs to be controlled from a remote computer. With PolyScope there are mainly two ways of doing that, and both uses socket. We could use the premade interface on port 30002 that has the robot controller acting as a host which reads and executes strings sent from client, or we could define our own interface on port 30000 and have the controller acting as a client.

To begin with, we wanted to use the existing interface on port 30002. To test the possibility of using this way to control the robot, we decided to make a program that let the robot write text with a pen. We quickly discovered that this interface was somewhat limited when it comes to feedback. There is also no way of validating if the commands sent has been finished, or if they even have reached the controller. So to make a program that supports all the functionality we need, we will make a program that runs locally in PolyScope, and receives commands with a self-defined Ethernet interface.

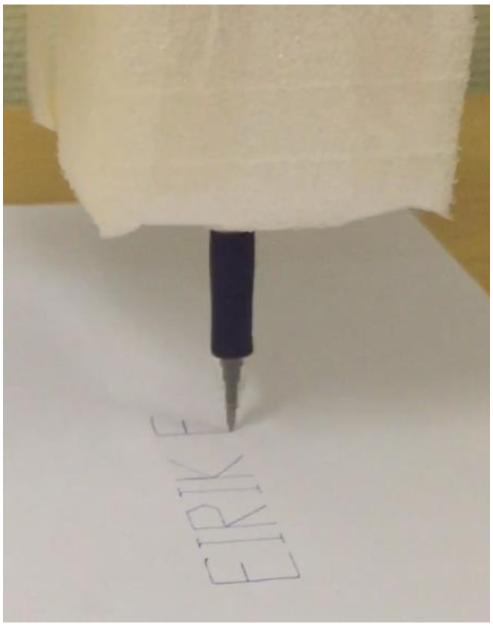
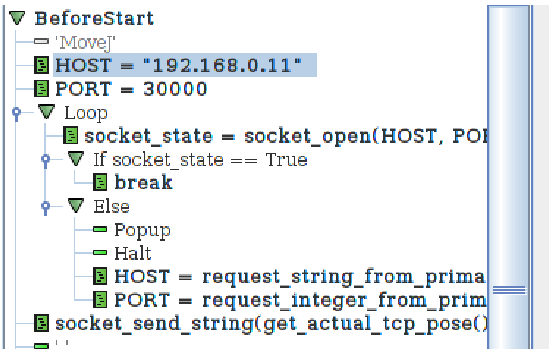


Figure 82: Print with UR

F.2.2.1.3 Combining Remote and Local

For our project we need to utilize the force-control available on the system from Universal Robots. We also need to be sure that the commands we send to the controller are received and processed. The interface on port 30002 does not offer sufficient feedback to meet these needs. Therefore, we had to make a program that offers exactly the functionality we need, with an interface for configuration via socket/Ethernet. A script will be run on the external computer that controls the program flow on the UR, and supplies it with necessary data from the vision system.

URScript has built in commands for communication by socket. For details, see *URScript manual chapter 5, module interfaces* [2].



```
▽ BeforeStart
  'MoveJ'
  HOST = "192.168.0.11"
  PORT = 30000
  ▽ Loop
    socket_state = socket_open(HOST, POR
    ▽ If socket_state == True
      break
    ▽ Else
      Popup
      Halt
      HOST = request_string_from_prima
      PORT = request_integer_from_prim
  socket_send_string(get_actual_tcp_pose()
  ''
```

Figure 83: Example of how to open a Socket on UR

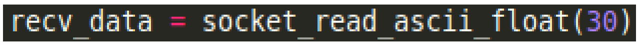
F.2.2.2 Remote Control Script with Socket Interface

F.2.2.2.1 Interface

Before making a program with the functionality that we need, we had to define the interface which would handle all dataflow between the external computer and the robot controller. Most examples we found on the Internet (including Universal Robots own support forum), used timing (simple wait commands) to control when data would be sent and read. Most examples also seemed to suggest sequential transference of strings. We wanted a more rigid solution with a handshake that confirmed that the right data was received. After some testing, we came up with an interface with handshakes where all data needed to run a given function. Here is how it works (for more detailed information, see *User Manual: UR Remote Control*):

- External computer hosts a socket connection on port 30002
- Robot controller works as a client and connects with host as in Figure 83, and immediately starts listening on socket
- External computer uses classes that generate a string of 30 floats based on some input, and sends them to controller
- Robot controller reads this message and replies if the data is valid
- Robot controller then executes a sub-program based on the content of the string, and replies as it goes back to listening.

We chose to use a string of floats because of limited resources when it comes to decomposing strings in URScript. URScript has a command that reads a string of floats from the socket, and stores them as a list. Lists are easier to process with URScript commands.



```
recv_data = socket_read_ascii_float(30)
```

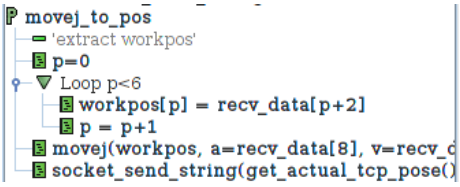
Figure 84: Extracts 30 Floats from String

F.2.2.2.2 Script with Necessary Functions

F.2.2.2.2.1 *Basic Moves*

We need to be able to move the tools around. This function is very basic. They read a waypoint from the socket in form of a *pose* (xyz-coordinates to represent position, and rxryrx to represent a tool orientation) and run them in a movel or movej function (see URScript-manual). Movej() calculates the “fastest” path to the new pose (not necessarily a straight line), while movel() moves the tool in a straight line to the new pose. These functions seem to work great.

- Movej()
- Movel()



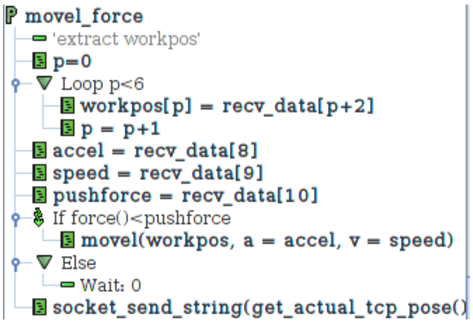
```
movej_to_pos
  'extract workpos'
  p=0
  Loop p<6
    workpos[p] = recv_data[p+2]
    p = p+1
  movej(workpos, a=recv_data[8], v=recv_d
  socket_send_string(get_actual_tcp_pose()
```

Figure 85: Extract a pose and move there

F.2.2.2.2.2 *Moves with Force Control*

Some of the operations we want the robot to perform require force control. Universal Robots have an integrated force calculation based on the input current to the motor in each joint that can be used two ways. The robot can be configured to apply a given force in a direction or towards a plane using the force_mode() command (see URScript manual), but this function causes some strange movements when there is no opposing force. The other method is to use the calculated force as a parameter in an if-statement. PolyScope offers continuously check of if-statements, which makes moving towards a waypoint until a given force is met very simple.

- Movel_force()
  - Movel_force(), as in Figure 86, moves tool towards a new pose until force() is more than a given force.
- Seek_down_force()
  - This function works in the same way as movel_force(), but instead of moving towards a new pose, it steps down 0.5mm at a time until given force is exceeded. Tests showed that this worked great when fastening self-drilling screws in sheet-metal.

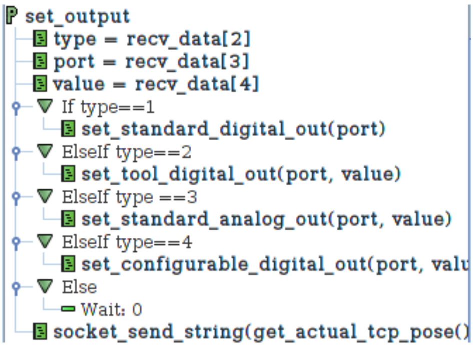


```
movel_force
  'extract workpos'
  p=0
  Loop p<6
    workpos[p] = recv_data[p+2]
    p = p+1
  accel = recv_data[8]
  speed = recv_data[9]
  pushforce = recv_data[10]
  If force()<pushforce
    movel(workpos, a = accel, v = speed)
  Else
    Wait: 0
  socket_send_string(get_actual_tcp_pose()
```

Figure 86: Movel_force

F.2.2.2.2.3 *Set and Reconfigure*

- Set_toolcenterpoint()
  - To begin with, we planned to use one robot with two tools. That meant that we had to be able to reconfigure the TCP (Tool Center Point) while running the robot. The set_tcp() function in URScript reads a pose and sets an offset from mounting point on the robot arm, in addition to configure a tool orientation. Even though we have moved over to using two arms with one tool on each, this function has become useful (see chapter F.2.3.1).
- Set_output()
  - The robot has several outputs, and we need to be able to configure these remotely to control the end-effectors on the robots. There are 4 different kinds of outputs on the controller, and each is accessed in its own way. The function reads 3 floats, where the first represent a port-type, the second a port number, and the third a value (0/1 for digital, 0 to 1 for analog).



```
set_output
   type = recv_data[2]
   port = recv_data[3]
   value = recv_data[4]
   If type==1
      set_standard_digital_out(port)
   ElseIf type==2
      set_tool_digital_out(port, value)
   ElseIf type ==3
      set_standard_analog_out(port, value)
   ElseIf type==4
      set_configurable_digital_out(port, valu
   Else
      Wait: 0
   socket_send_string(get_actual_tcp_pose()
```

Figure 87: Set_output()

F.2.2.2.2.4 *Get Data*

- Get_input()
  - Required to get sensordata read to external computer. Looks very much like the function to set outputs, but returns the value of a given port instead.
- Get_current_pos()
  - Necessary to translate pixel-coordinates to robot base coordinates.

F.2.2.2.2.5 *Make circular moves*

The griddle is circular, and some of the key positions that we want to find, is on the edge of this circle. So to scan for positions and move around the griddle efficiently, we need to calculate points on this circle, and make circular moves around it.

- Get_wp_circle()
  - We have made a function running on the robot controller that calculates waypoints on a circle with a centerpoint and a radius. These waypoints can be used to do a simple move to the circle, or a circular move along the circle.
- Move_circular()
  - Reads two waypoints and calculates a circular path based on the current position and the two waypoints. We will use this to scan for key positions along the edge of the griddle, and to position the tool. There seems to be some odd behavior at the end of a circular move (as described in F.2.3.2), so a circular move should be followed by a movel() to ensure that positioning is correct. To ensure that tool follows the planned path and that the tool rotates in the right direction, a circular move should not rotate more than 90 degrees around an axis at a time.

F.2.2.2.2.6 *Expanding Functionality*

User manual for our UR-control program describes how to expand the functionality of the program.

F.2.3 Other

F.2.3.1 Rotation of Tool

Rotations on the UR-system is represented by 3 “coordinates”; rx, ry and rz. While working on the griddle, the screwdriver will have to work from at least two different orientations; locked pointing down parallel with a vertical axis to fasten screws from above, and parallel with a horizontal line while rotating around the vertical axis. We had to be able to anticipate how the tool would be oriented with a given rxryrz to calculate new positions. As easy as this sounds, it proved to be a major challenge.

As a start, we tried to figure out what the rxryrz representation was. One might believe a quick search on the internet would reveal the method used to calculate tool orientation by Universal Robots, but this was not the case. We found a couple of examples describing how the values changed, but only about one axis at the time. We did a few tests rotating the toolhead on our robot, and discovered that rotating about one axis at a time made a lot of sense, but with two or more axis at a time it was very little intuitive. So we contacted our distributor, but he didn’t know either. We got forwarded to Universal Robots and a partner of our distributor, RocketFarm. RocketFarm responded quickly and told us that Universal Robots used something called Axis-Angle, and that they had went through a whole lot of trouble to convert them to something more intuitive. We spent a lot of time trying to get the hang of Axis-Angle, but made little progress. We started looking for alternative options, and discovered, since rotations around one axis was simple, that we could set a new tool-orientation for each working pose. When working from above the griddle (left sketch in Figure 88), an orientation parallel with z-axis (vertical) would set to be (rx,ry,rz)=(0,0,0). All operations that the robot shall do from above can be done from this orientation. When working from the side of the griddle (right sketch in Figure 88), we rotate the tool 90 degrees about either x- or y-axis (horizontal axis) and sets that position as the new tool orientation (0,0,0). To move around the griddle with the tool always pointing at center, we only need to rotate around z-axis (change rz). This is not the best solution as it gets kind of messy and is somewhat limited, but it is by far the easiest solution (and the only solution we had time for).

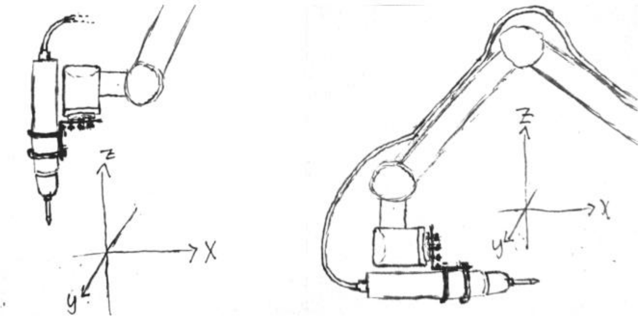


Figure 88: Vertical and horizontal tool orientation

F.2.3.2 Circular Move

Movec() is a function in URScript. Initially we wanted to use this command to “scan” the griddle in one move, making the robot move in a circle around the griddle with the tool pointing to the edge the whole time. Movec() reads two waypoints, a “via”waypoint, and a target waypoint. Based on the current position of the tool and the two waypoints, it calculates the shortest circular arch from current position to the target. As mentioned before, a waypoint is composed of three Cartesian coordinates (x,y,z), and a vector representing tool orientation (rx,ry,rz). Research told us that we had to use at least two movec() commands to make a complete circle, but we also learned that the orientation in the via-waypoint was ignored. It means that the tool orientation can’t be adjusted continuously in a circular move using movec() with more than 1 non-zero value of (rx,ry,rz) without having to go through axis-angle calculations. As mentioned in chapter F.2.3.1, we had solve this by first reconfiguring the tool orientation and then use the get_wp_circle() [2] to calculate three waypoints, move to the first, and then run through the other two with a movec().

Another problem that we have been unable to resolve is that a movec() skips its smooth deceleration-phase and comes to a sudden stop often a bit after the last waypoint. This means that performing a movec() at full speed might result in an emergency stop and increased stress. Some measures must be made until this bug is fixed. The maximum speed of the movec() must be low, and a movec() should be followed by a movel() to ensure that position is correct.

F.2.4 Conclusion

PolyScope and URScript works great for simple tasks, but for more advanced programming, it is somewhat limited (e.g. string handling). The documentation around URScript seems to be unfinished as it does not cover all the functions that actually exist in the language (e.g. request_string_from_primary_client()), and lacks some details about other functions. This has caused some great headaches for our programmers when getting errors that make little or no sense.

We have made a program that supplies us with all the functionality we need to perform all the operations necessary to assemble the griddle, and also made a foundation that easily can be built upon in later projects. We spent a lot of time trying to fix some kinks, some we solved, and some we had to work around. All in all, we are satisfied with the resulting software, and are looking forward to see it working in the final system.

F.2.5 References

1. Universal Robots UR Manual, 18.05.2015: http://www.ur-update.dk/URsupport/Manuals/UserManual/Release3.1/UR10_User_Manual_en_Global.pdf
2. Universal Robots URScript manual, 18.05.2015: http://www.ur-update.dk/URsupport/Manuals/ScriptManual/Release3.1/scriptmanual_en.pdf
3. User Manual: UR Remote Control, 18.05.2015

F.3 Cable connector mounting

Abstract

We set out to see if we could simplify the design of Engmark Meteors griddle. The goal was to simplify the cable connector mounting by mounting it directly to the reflector.

The tests conclude that the original design is superior.

Contents

Abstract 235

Contents 235

Figures 235

Charts 235

F.3.1 Introduction 236

F.3.2 Development tests 236

F.3.2.1 Reference test 236

F.3.2.2 Full conductor test 237

F.3.2.3 Reduced conductor test 238

F.3.2.4 Insulator test 239

F.3.3 Conclusion 240

Figures

Figure 89: Original cable connector design reference test 236
Figure 90: Full conductor temperature test 237
Figure 91: Reduced conductor temperature test 238
Figure 92: Insulating conductor temperature test 239

Charts

Chart 1: Original cable connector design reference test 236
Chart 2: Full conductor temperature test 237
Chart 3: Reduced conductor temperature test 238
Chart 4: Insulating conductor temperature test 239
Chart 5: Electrical connector temperature test summary 240
Chart 6: Fastening point temperature test summary 240

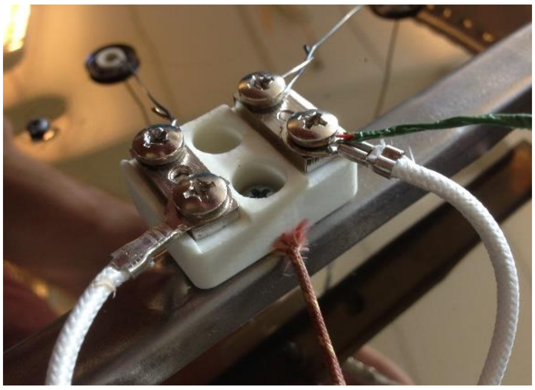
F.3.1 Introduction

The goal of our project is to automate the assembly of Engmark Meteors griddle. One of the challenges we face is that the current way of mounting the cable connector forces us to make substantial alterations to the griddle, and most likely to our automation system. If the cable connector were simply placed on the power bracket, it would fall off,

F.3.2 Development tests

F.3.2.1 Reference test

First we needed a reference to measure the results of the others test against. Therefore, we measured the temperature development of the original griddle design.



On the picture, we see the points measured on the cable connector: Where the electrical cable is connected (green) and where the cable connector is fastened (orange).

Figure 89: Original cable connector design reference test

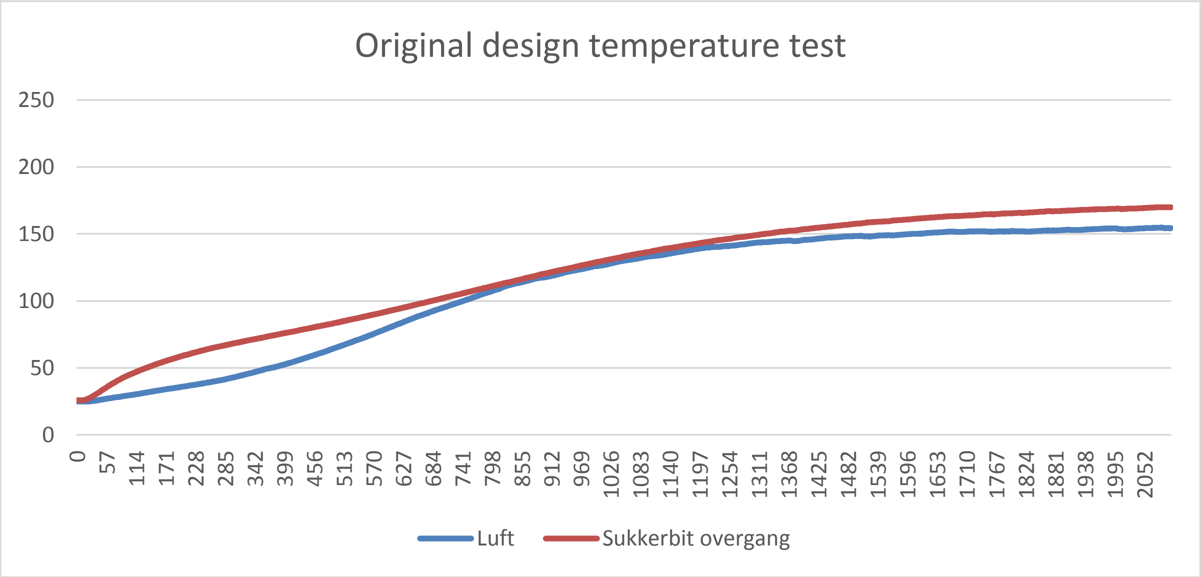


Chart 1: Original cable connector design reference test

Here we see that the temperature rises steadily and eventually flattens out towards acceptable temperatures (> 180 degrees Celsius).

F.3.2.2 Full conductor test

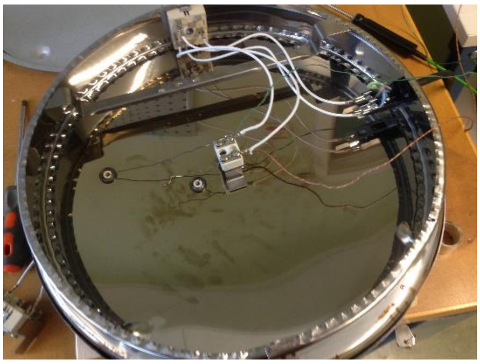


Figure 90: Full conductor temperature test

In this test, we install the cable connector directly to the reflector by having a screw connecting the cable connector directly. It is supported by a bracket in direct contact with both the reflector and the cable connector.

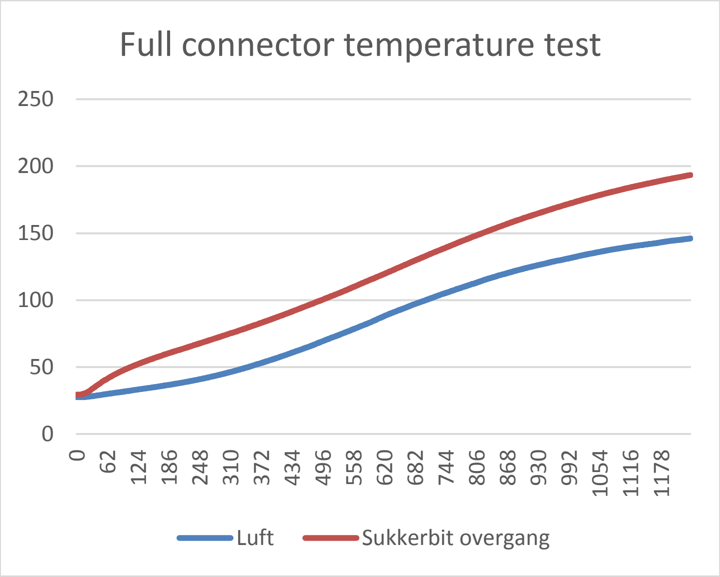


Chart 2: Full conductor temperature test

Here we see that both the temperature in the fastening point and in the electrical connection point rises very quickly, both exceeding 200 degrees rapidly.

Note that the test was aborted after 1200 seconds (20 minutes) as to not damage the electrical components of the griddle. The horizontal size of the chart has been adapted to represent this.

F.3.2.3 Reduced conductor test

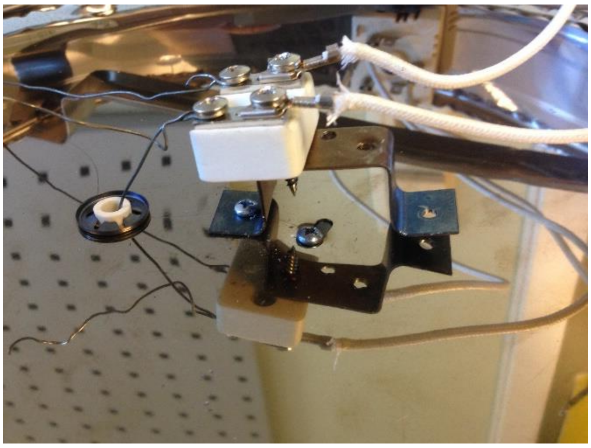


Figure 91: Reduced conductor temperature test

In this test, we altered how the cable connector is mounted on the bracket. A shorter screw was used on the connector itself and the bracket is connected with only one side.

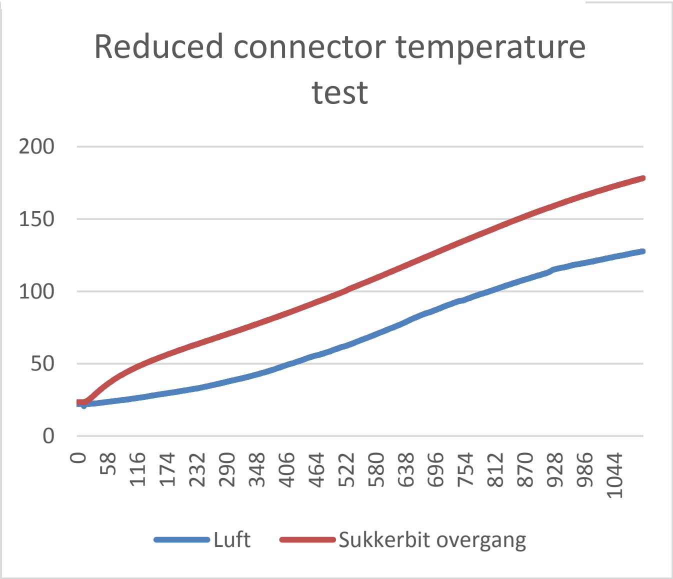


Chart 3: Reduced conductor temperature test

Here we see that temperature development on the connector point is similar to that of the full connector test. The temperature of the fastening point is lower, but still higher than the reference test.

Note that the test was aborted after 1100 seconds (18 min, 20 sec) as to not damage the electrical components of the griddle. The horizontal size of the chart has been adapted to represent this.

F.3.2.4 Insulator test

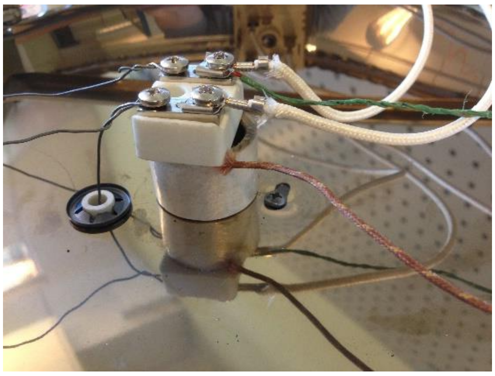


Figure 92: Insulating conductor temperature test

In this test, we replaced the metal bracket with a non-conductive material. The cable connector is fastened directly to the reflector with a metal screw and is supported by the insulator.

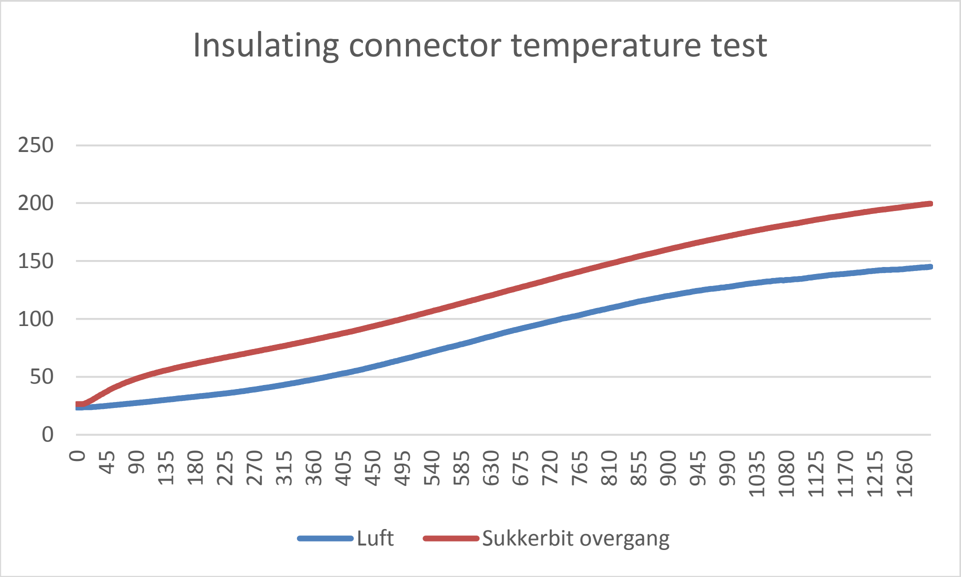


Chart 4: Insulating conductor temperature test

Here we see a temperature test similar to the reduced conductor test.

Note that the test was aborted after 1300 seconds (21 min, 40 sec) as to not damage the electrical components of the griddle. The horizontal size of the chart has been adapted to represent this.

F.3.3 Conclusion

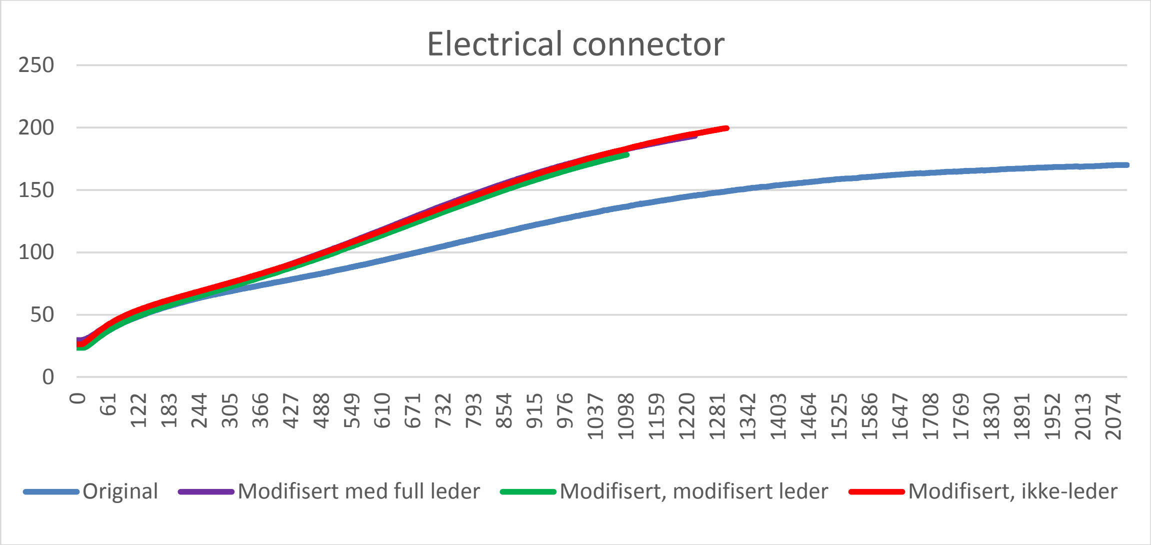


Chart 5: Electrical connector temperature test summary

This chart shows the temperature development on the cable connector where the electrical connector is fastened during the different tests.

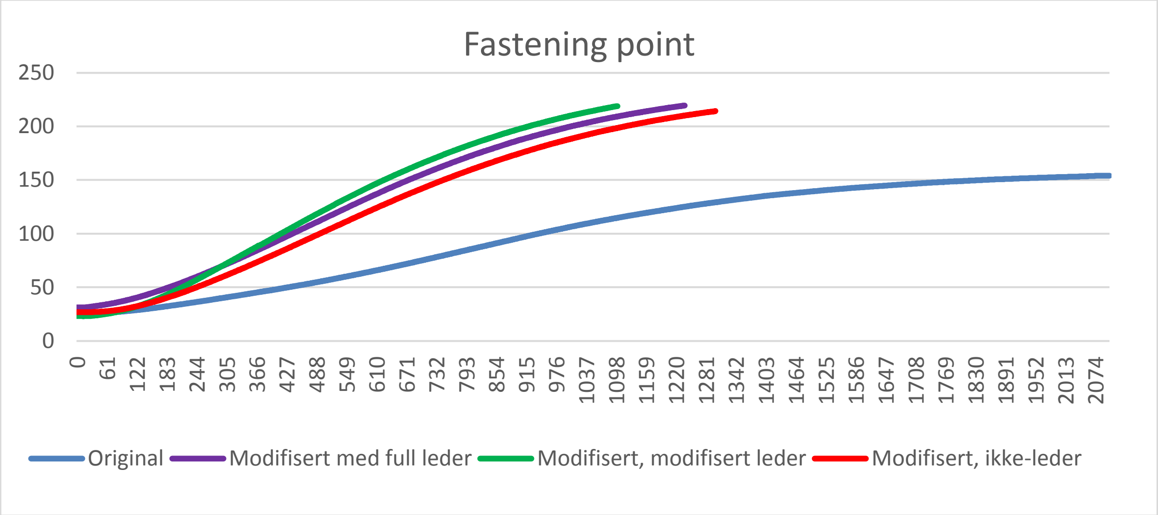


Chart 6: Fastening point temperature test summary

This chart shows the temperature development where the cable connector is fastened during the different tests.

The tests show clearly that the original design is superior to our alterations. This confirms our suspicion that the power bracket is acting as a heat sink for the cable connector and we will have to implement a solution using this knowledge.

F.4 End effector bracket

Abstract

This document contains a detailed description of which designs were considered, which was chosen to be prototyped and end result of the end effector brackets which will hold the tools needed to automate the production at Meteor.

Contents

Abstract 241

Contents 241

Figures 242

F.4.1 Introduction 243

F.4.1.1 Tools 243

F.4.1.1.1 Camera 243

F.4.1.1.2 Gripper 243

F.4.1.1.3 Screwdriver 243

F.4.1.2 Concepts 243

F.4.1.2.1 Tool change concept 244

F.4.1.2.2 All in one concept 244

F.4.1.2.3 Double robot concept 244

F.4.1.3 Chosen solutions 244

F.4.1.3.1 One robot 244

F.4.1.3.2 Two robots 245

F.4.2 Development 245

F.4.2.1 Tool set up 245

F.4.2.1.1 All in one concept 245

F.4.2.1.1.1 Gripper 245

F.4.2.1.1.2 Screwdriver and camera 245

F.4.2.2 Prototypes 245

F.4.2.2.1 One robot bracket 245

F.4.2.2.1.1 The design 245

F.4.2.2.1.2 First prototype 246

F.4.2.2.1.3 Second prototype 247

F.4.2.2.2 Two robot brackets 248

F.4.2.2.2.1 The design 248

F.4.2.2.2.2 Prototype 249

F.4.3 Conclusion 249

F.4.3.1 One robot bracket 249

F.4.3.2 Two robot brackets 250

F.4.3.2.1 Gripper bracket 250

F.4.3.2.2 Camera and screwdriver bracket 250

F.4.3.3 Material choice 250

F.4.4 References 250

Figures

Figure 93: IFM sensor 243

Figure 94: Festo gripper 243

Figure 95: Sumake screwdriver 243

Figure 96: Main bracket 246

Figure 97: Camera bracket 246

Figure 98: Assembly 247

Figure 99: Prototype 2 248

Figure 100: 3D model of gripper bracket 248

Figure 101: 3D model of camera and screwdriver bracket 249

Figure 102: Camera and screwdriver bracket prototype 249

F.4.1 Introduction

To solve the automation of the griddle, our system will use several tools. We need a tool to move components to a specific location, a tool to fasten these components, and a tool to recognize griddle orientation and key positions.

F.4.1.1 Tools

F.4.1.1.1 Camera

The camera our system shall use is an IFM sensor. It can be fastened by using the holes on the sides.

Dimensions:

- Height:
  - 5,4 cm
- Bottom:
  - 6 cm x 4,2 cm

Figure 93: IFM sensor

F.4.1.1.2 Gripper

The gripper that our system shall use is a DHPS-A 35 mm parallel pneumatic gripper from Festo. It can be fastened to a bracket in two ways, by either using the fastening holes on the sides or on the bottom.

Dimensions:

- Height:
  - 142 mm
- Bottom:
  - 88 mm x 45 mm
- M8 holes for fastening the gripper
- M6 holes for fastening the fingers

Figure 94: Festo gripper

F.4.1.1.3 Screwdriver

The screwdriver that we shall use for testing is a Sumake full-auto shut off air composite screwdriver. It can be fastened by being strapped at the handle point.

Dimensions:

- Width:
  - 4,7 cm at the widest
- Length:
  - 24 cm

Figure 95: Sumake screwdriver

F.4.1.2 Concepts

During the assembly process, the system will be picking up, placing and fastening components. It will focus on one component at a time, until it has done the operations required to fasten that specific component. Two concepts were considered for our system at the start. *Assembly process alternatives*

A second robot arm was introduced after the Easter, which led to change in the systems set up. The access to another robot led to a need for new concepts for a two robot arm system. *Assembly process alternatives*

F.4.1.2.1 Tool change concept

The first concept was a tool changing end effector that would change between screwdriver and the gripper by connecting and disconnecting the brackets holding these two tools. In this concept, the robot will be holding one tool at the time, thus the end effector will work as a simple extension of the robot arm. However the end result would be a drop in efficiency, as a result of constant tool changing.

Additionally the tool changing end effector is quite complex in comparison with the second concept. A tool changing end effector would need a separate camera fastening, so that the camera is always on the robot arm. Another solution might involve camera being fastened to a bracket that will hold one of the other tools, but this means that a well-planned use of the camera is needed. The assembly must be thought through so that the use of camera is well coordinated and hole/shape locations are all detected before the tool is changed. This is a complex approach which might not be efficient.

F.4.1.2.2 All in one concept

The second concept was an end effector consisting of all three tools fastened to a bracket at once. The second concept will eliminate the time spent changing tools and only require a change in the orientation of the end effector. The end effector will however be bigger, and this may lead to some limitations to robot movement. The end result for this concept would be a lot faster and simpler than the first concept, at the expense of robot mobility.

F.4.1.2.3 Double robot concept

After the second robot was introduced, a third concept was created by the group. This concept consisted of two different brackets. The first one is designed only for the gripper. This means that the focus for this bracket was to withstand the forces that will be applied while the gripper is lifting components. The second bracket is for operation of a camera and a screwdriver. This bracket is designed to optimize the cooperation between these two tools. The end result of this concept would be an optimized assembly process, where the only difficulty is the synchronization of robot cooperation.

F.4.1.3 Chosen solutions

F.4.1.3.1 One robot

After a thorough consideration, the group decided that the second concept should be chosen for further development for one robot solution. This decision was made because the second end effector had numerous advantages over the first one. The main advantage over the tool changing solution was the simplicity of all in one bracket. A bracket holding all of the tools was much easier to design, produce and test than a tool-changing system. The bracket can be easily made out of a material which is available through Meteor. This would make the cost of the bracket very small, while still delivering the product that fulfils our requirements. One of the other advantages is the camera placing. This concept has a camera always located on the working bracket. The availability of the camera makes the assembly process more effective.

F.4.1.3.2 Two robots

There was only one concept created for a two robot arm assembly system. This concept was efficient and simple and therefore it was chosen for further development for two robot solution.

F.4.2 Development

F.4.2.1 Tool set up

F.4.2.1.1 All in one concept

When developing the end effector bracket we considered the tool placement. A gripper, a screwdriver and a camera should be fastened to the UR10 in a way that would restrict the movement as little as possible. It should be as optimal as possible when it comes to the use of the tools under the assembly and the weight distribution.

F.4.2.1.1.1 *Gripper*

Since we are planning to lift components ranging from cable connector to the whole griddle, we decided that the gripper should be located as close to the middle of the arm as possible. This is to ensure that most forces working on the bracket are located as close to the mounting point to the arm as possible. This should secure stability of the bracket and minimize shear stress. The stability is desired when it comes to picture processing.

F.4.2.1.1.2 *Screwdriver and camera*

As we do not wish for this bracket to be too long in one direction (which would restrict movement), a decision was made to place the tools on both sides of the gripper. The screwdriver was placed to the right of the gripper and the camera to the left. To minimize the interference of each of the tools, they were placed in different angles. This required changing orientation of the end effector bracket when operating the different tools.

For testing we used a screwdriver that we borrowed from Meteor, while another one will be provided as soon the testing is done. The first design was therefore made to fit the screwdriver we have borrowed. Adjustments in the final design will be made to fit the new screwdriver. To make sure that the camera could be pointing in several directions and not just at a locked angle, a separate bracket was made for it. This camera bracket was fastened to the main bracket, with options to change angles in both x- and y-axis. This provided us with versatility which came in handy when aligning the camera field of view with the screwdriver and testing for optimal angle.

F.4.2.2 Prototypes

F.4.2.2.1 One robot bracket

F.4.2.2.1.1 *The design*

The first prototype bracket was designed for the 35mm gripper, pneumatic screwdriver from Meteor and an IFM sensor camera. The first thing we did was to measure the distance between the three tools to set up a drawing/sketch of the bracket. It is important that the tools can operate without any risks of interference with each other. A design choice was made to give the bracket two “wings” on each of the sides. These so called “wings” will be bent 90 degrees so that the gripper can be fastened to the bracket on the sides with bolts.

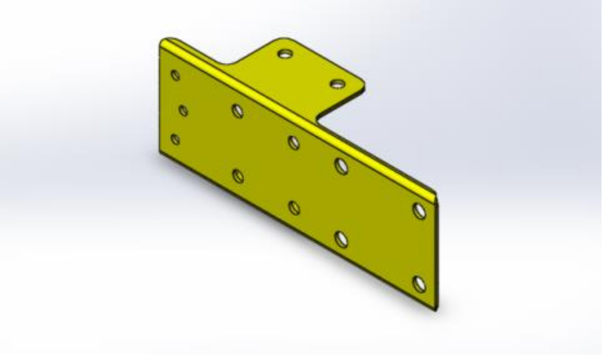


Figure 96: Main bracket

Because of the weight distribution and forces that could appear on the bracket, the fastening point between the bracket and UR10 should be located as close to the middle of the bracket as possible. The same goes for the gripper since it will carry the components. This should ensure the center of gravity to be in the middle, so that the bracket is not exposed to unnecessary heavy forces on the edges, which could lead to shear stress in the bracket. To make the bracket even sturdier we decided to make a 90 degree bend alongside each of the brackets longest sides.

We designed a separate camera bracket which will be fastened to the main bracket. This will make the bending process easier and more change-friendly in case we decide to alter the angles for the camera. It is possible that the camera will be adjusted to point to either the gripper or the screwdriver if we discover a need for this.

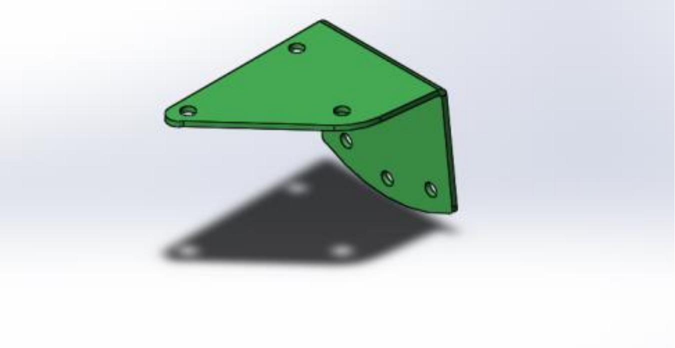


Figure 97: Camera bracket

F.4.2.2.1.2 *First prototype*

After we gathered all of the needed data, sketches were modelled in SolidWorks. Sheet metal function were used to make the brackets. Sheet metal provides a variation of useful functions which lets you work with the bending radius, bending positions and adjustment of the machine to match the bends. These functions come in handy so that you can take in consideration different outcomes and receive a component that matches the drawing exactly.

The camera bracket has three fastening points to the main bracket. This will secure the stability of the camera. The bracket also has two fastening positions for the camera. One that points the camera parallel with the screwdriver and another that points it 45 degrees relative to the screwdriver and the gripper.

To make sure that our designs fit perfectly, we imported CAD models of the IFM camera and the gripper from the homepages. Appropriate 3D model of the screwdriver was made and all three tools were put into an assembly with the brackets. This gave us reassurance that our design for the brackets had a perfect fit. Since all three tools have their own cables for either electric power or air pressure, it was wise to plan ahead for how these cables will be handled. With our design all cables should be easily guided upwards towards the bracket and be fastened alongside the robot arm. When the team was happy with the design, drawings were converted into dwg format and sent to be cut out at Meteor.

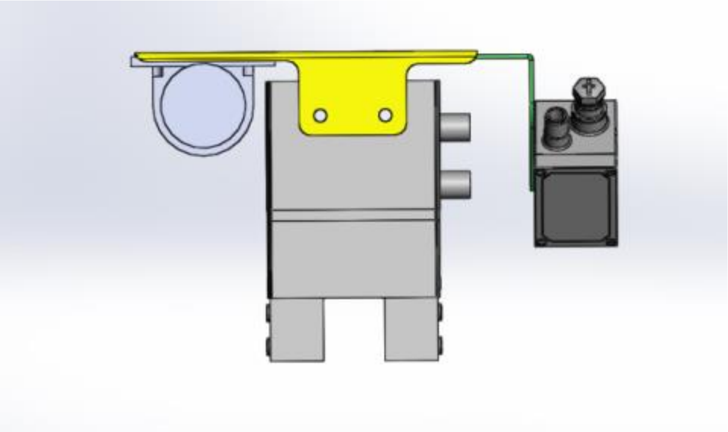


Figure 98: Assembly

After the brackets were cut out we needed to bend them to form their shape. The brackets were formed using the bending machine we have in our schools workshop. The brackets we ended up with were then bolted to the robot, and tested to see if the tools fit as planned. The first prototype of the brackets fit perfectly, but adjustments were made during this period of time, and so a second prototype was needed.

F.4.2.2.1.3 *Second prototype*

The group decided to change the set-up of the tools to optimize the functionality of the robot. The change was made to both brackets, but only the camera bracket was directly affected by this change. A special hole that lets the camera bracket to be rotated 45 degrees was designed on the main bracket. This gives the camera ability to be turned towards the screwdriver, so that it can see the tip of the screw. This set up will let the camera see a hole while the screwdriver is placed directly over it. This will remove the time needed to move the screwdriver after the camera has located orientation of a hole.

Matching hole was designed on the camera bracket to give the camera 45 degrees of possible positioning instead of 0 and 45. The new brackets were made out of the same material and in the same way as the previous prototype. They were used for further testing of the camera and robot. When the right new gripper and screwdriver arrives the plan is to design new brackets to fit perfectly for the new tools.

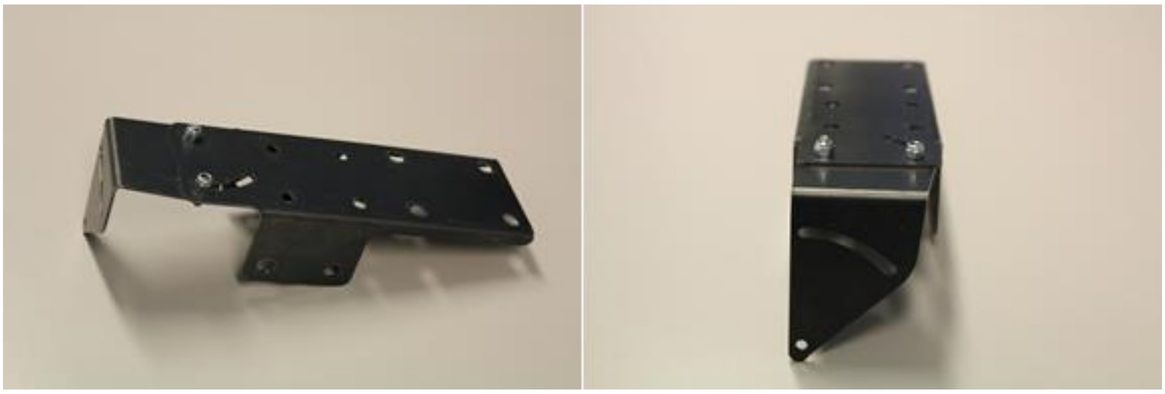


Figure 99: Prototype 2

F.4.2.2.2 Two robot brackets

F.4.2.2.2.1 *The design*

The design for the gripper bracket is a lot like the earlier brackets design. In the middle it has a fastening point to the robot arm. The fastening of the gripper is designed just as in the previous bracket. Just as in the one robot arm design, to make the bracket even sturdier we decided to make a 90 degree bend alongside each of the brackets longest sides.

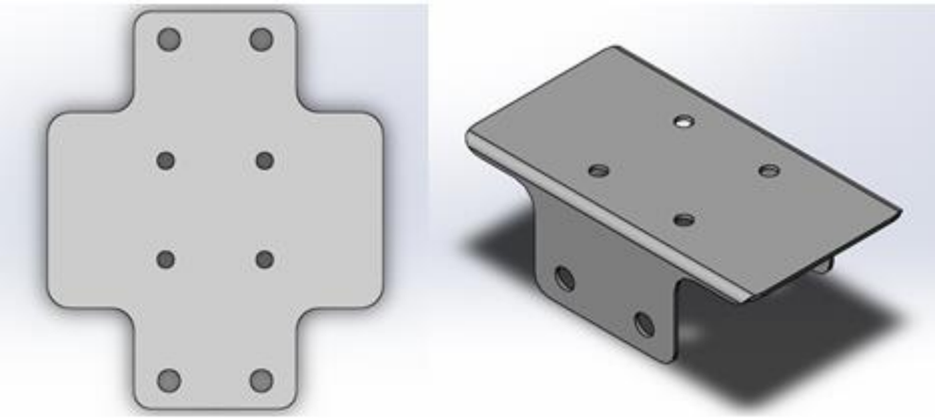


Figure 100: 3D model of gripper bracket

The camera and screwdriver bracket design was changed from the earlier brackets design. The bracket will have two fastening points to the robot arm. This will provide the user with variation in the orientation of the tools. The new bracket has to have fastening points for the camera designed so that the camera has a specific angle. At the same time it needs a variation in position so adjustments can be made in relation to the new screwdriver. The camera lens will be positioned facing the screwdriver. The screwdriver choice is still unclear, thus the design of the bracket has to provide a variation of screwdriver fastening position.

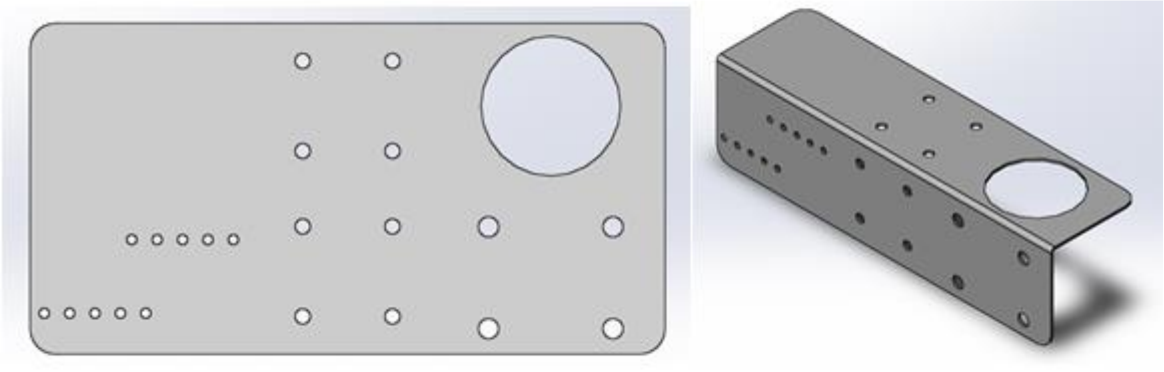


Figure 101: 3D model of camera and screwdriver bracket

F.4.2.2.2.2 *Prototype*

When the sketches were done and 3D models were made, we went to the schools workshop to make the prototypes. This time we made the whole prototypes ourselves. This way we did not need to wait for the prototypes to come back before we could test them. We cut out a piece of sheet metal that had the right dimensions and went on to drill the fastening holes. When the holes were finished we made a bend in the middle of the bracket that was horizontal with the length of the bracket.

We tested the bracket and it worked perfectly. This bracket gave the group enough camera position variation so that the screwdriver position could be adjusted to the wanted height. It also provided the camera with protection so it is not damaged in case of a collision.

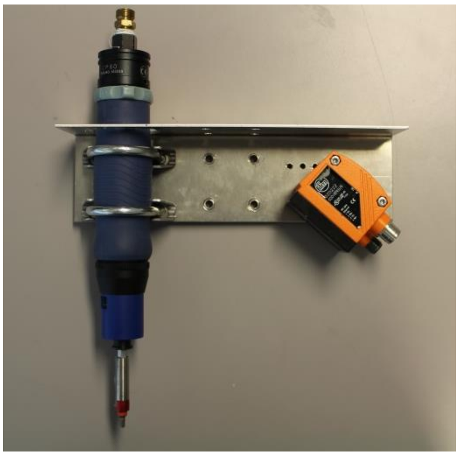


Figure 102: Camera and screwdriver bracket prototype

F.4.3 Conclusion

F.4.3.1 One robot bracket

The development of one arm end effector bracket stopped after the second prototype. The reason for this was the introduction of a second arm for the project. The brackets that we have developed up to this point are still in prototype phase, and designed for the tools used under the development phase. Adjustments are needed if the brackets will be used with new tools. When it comes to one arm solution we are satisfied with the information we have collected during the process of developing these brackets. During the tests that we have performed, these brackets have worked as required. We have created a solution that is low-cost and easy to manufacture.

F.4.3.2 Two robot brackets

F.4.3.2.1 Gripper bracket

The gripper bracket is designed for the DHPS-A 35mm parallel pneumatic gripper. It is designed to provide with stable and safe operation of the gripper.

F.4.3.2.2 Camera and screwdriver bracket

The bracket is designed to be used for the IFM sensor and Sumake screwdriver. The bracket provides with needed variation in camera position, screwdriver position and overall tool orientation. It is also designed to protect the camera in case of a collision.

F.4.3.3 Material choice

For the material we will use a simple 2.5mm thick steel plate they use at Meteor. Meteor has used this kind of steel plate to make their current vacuum tools. They have tested the vacuum tool with the largest griddle, which weighs about 2 kilograms more than the one we will be lifting. Thus the conclusion was made by the group that the material is strong enough to be used for all of the brackets.

F.4.4 References

1. Figure 93 [15.05.2015] http://www.conrad.com/medias/global/ce/5000_5999/5000/5070/5072/507203_BB_00_FB.EPS_1000.jpg
2. Figure 95 [12.05.2015] http://im01.taiwantrade.org/resources/member/982/productcatalog/cp30_1.jpg

F.5 Gripper fingers

Abstract

This document describes the development of a pair of gripper fingers for a DHPS-A 35 mm pneumatic parallel gripper from Festo, that will be used to automate the assembly of a griddle at Engmark Meteor AS.

Contents

Abstract 251

Contents 251

Figures 252

F.5.1 Introduction 254

F.5.1.1 Grippers and jaws 254

F.5.1.2 Objects to grip 254

F.5.1.2.1 Cable connector E-CC 254

F.5.1.2.2 Regulator E-REG 254

F.5.1.2.3 Vacuum tool 255

F.5.1.3 Jaw types 255

F.5.1.3.1 Encompassing and retention 255

F.5.1.3.2 Friction 255

F.5.1.4 Needs 255

F.5.1.4.1 Cost 255

F.5.1.4.2 Weight 256

F.5.1.4.3 Robustness 256

F.5.1.4.4 Object retention 256

F.5.2 Development 256

F.5.2.1 Design 256

F.5.2.1.1 Gripping cable connector E-CC 256

F.5.2.1.2 Gripping regulator E-REG 257

F.5.2.1.3 Gripping vacuum tool 257

F.5.2.1.4 Sketch 257

F.5.2.1.5 Forces applied to the gripper fingers 258

F.5.2.1.5.1 Lifting forces 258

F.5.2.1.5.2 Pressure forces 258

F.5.2.1.6 Material choice 259
F.5.2.2 Prototype 1 260
F.5.2.2.1 3D modelling prototype 1 260
F.5.2.2.2 Simulation 261
F.5.2.2.2.1 Static test 261
F.5.2.2.3 3D printed gripper fingers 263
F.5.2.2.4 Testing by using the prototype 263
F.5.3 Conclusion 265
F.5.3.1 Material 265
F.5.3.2 Friction pads 265
F.5.3.3 General thoughts 265
F.5.4 References 265

Figures

Figure 103: Cable connector E-CC 254
Figure 104: Regulator E-REG 254
Figure 105: Vacuum tool gripping point 255
Figure 106: Gripping actions 255
Figure 107: Gripping of cable connector 256
Figure 108: Gripping of regulator 257
Figure 109: Gripper force graph 258
Figure 110: Minimum gripping force formula 259
Figure 111: Gripper finger 3D design 260
Figure 112: Assembly and interference check 260
Figure 113: Fixtures table 261
Figure 114: Fixtures 261
Figure 115: Weight forces, gripper force forces, mesh 262
Figure 116: vonMises stress simulation and most affected areas 262
Figure 117: Gripper finger prototype 263
Figure 118: Cable connector testing 264
Figure 119: Cable connector fastening 264
Figure 120: Placing regulator 264

F.5.1

Introduction

F.5.1.1 Grippers and jaws

Grippers are widely used in automated assembly since repetitive movement of picking up a component and placing it in a specific place, can be easily replaced by a robot. The jaws that the grippers use are usually designed for a specific component, which ensures a precise and secure grip. Sometimes a robot is set up to grip several components by using the same gripper. The components in this case have similar shape and size. If the variation in size or shape is big, an expensive gripper might be needed, which might not be an alternative for smaller companies. Another solution for solving the variation in shape and size is a special set of jigs and gripper fingers designed to work together. This is relevant for our project, since we are to grip several components and a jig with variation in size and shape.

F.5.1.2 Objects to grip

The gripper that we chose will have to grip two components and a jig. The two components have a variation in size and shape. When preparing to design gripper fingers, we took a closer look at the objects that the gripper will grip.

F.5.1.2.1 Cable connector E-CC

This part is flat on every side except for the top with the screws. This gives a variety of gripping possibilities.

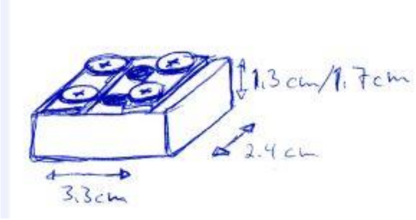


Figure 103: Cable connector E-CC

Dimensions:

- Height:
  - 1,3 cm (top ceramic material)
  - 1,7 cm (top of screw)
- Bottom:
  - 2,4 cm x 3,3 cm
- Weight:
  - 31 grams

F.5.1.2.2 Regulator E-REG

This part requires a minimum of 2,2cm opening. To be sure to be able to grip it in any orientation, the gripper needs an opening of over 4,8cm. This part is flat on the top, bottom and sides. At the front, it is flat except for the adjustment pin. At the back, there are many cable-connecting points. As with the cable connector, the regulator has a lot of flat surface that will give a variety of gripping possibilities.

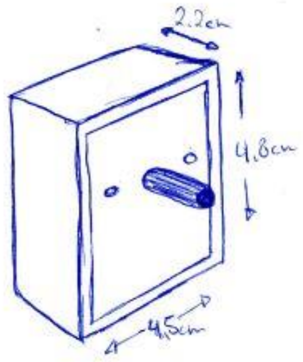


Figure 104: Regulator E-REG

Dimensions:

- Height:
  - 4,8 cm
- Width:
  - 4,5 cm
- Front to back:

    - 2,2 cm
- Weight:
    - 97 grams

F.5.1.2.3 Vacuum tool

One of the uses of the vacuum tool is to lift a finished griddle from the workstation. This means that the vacuum tool will be used to move heavy objects. The design of the gripper fingers and vacuum tool gripping has to provide a simple, precise and secure interlocking mechanism for moving the tool with payload.

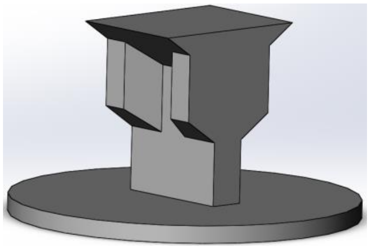


Figure 105: Vacuum tool gripping point

Dimensions:

- Weight:
    - 1.5 kg

F.5.1.3 Jaw types

As mentioned the gripper jaws are designed to securely hold and move objects and components at task. The contact that the jaws make with the object is a major factor when it comes to required gripping force. There are two types of gripping actions:

F.5.1.3.1 Encompassing and retention

Encompassing and retention gripping jaws will have a design that will let them grip and cradle an object. The added stability to the object which also leads to less required gripping force makes it a preferred method. An increase in the gripper travel stoke might be needed to encompass an object in relation to friction gripping.

F.5.1.3.2 Friction

Friction gripping jaws rely completely on the force of the gripper and finger friction to hold the object. They are easiest to fabricate and usually require more gripping force to hold the objects. This is especially noted with the objects that do not have flat surfaces. Parts can easily be dislodged by forces applied from any directions. Friction gripping may require up to four times more force than the Encompassing or retention gripping.

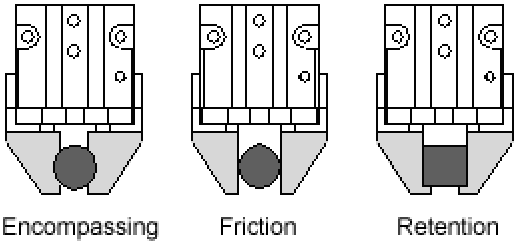


Figure 106: Gripping actions

F.5.1.4 Needs

The gripper fingers that we will design have to be strong, light and cheap. To achieve these properties, needs and requirements have to be specified.

F.5.1.4.1 Cost

The cost of the gripper jaws depend on material and machining costs. The goal is to make a pair of gripper fingers that will be cheap and at the same time strong, so that they fulfil our needs. Suggested solutions will be weighted when it comes to price and only the best solution will be chosen.

F.5.1.4.2 Weight

The gripper has a specific finger weight limit. The limit is set to be 450 grams per finger. The weight limit leads to consideration of different materials that will both fulfil the strength requirements and be light at the same time. The design of the fingers must be optimized so that there is no excess of material. This will decrease the weight and also could lead to lower costs.

F.5.1.4.3 Robustness

The gripper fingers need to be robust to be able to withstand the force applied from the gripper when the gripper is gripping and lifting an object. The material and design play an important role when it comes to the toughness of the fingers. Simulations will be performed in SolidWorks to find the weak points in the design. These simulations will also be used to verify the material solution.

F.5.1.4.4 Object retention

The gripper fingers have to have a design that ensures secure gripping of the objects. Secure gripping can be described as an ability to both hold the object without dropping it, and giving the gripper control over the object at task. To secure the gripping, gripper fingers design might be adapted to handle the heavier objects with higher factor of safety.

F.5.2 Development

F.5.2.1 Design

The design of our gripper fingers relied a lot on securing the gripping of the components and securing the gripping of the vacuum tool. The available stroke length of our gripper is 2.5cm. Since the objects have a variation in size, the group had to consider the most optimal orientation of which to grip from for each of them to make best use of the available stroke length.

F.5.2.1.1 Gripping cable connector E-CC

The cable connector is the smallest component that our system is supposed to grip. To decide how the gripper will be gripping cable connector, the group had to consider the total component gripping stroke. To make the difference in total component gripping stroke smallest possible, it was most optimal to grip the cable connector as in Figure 107.

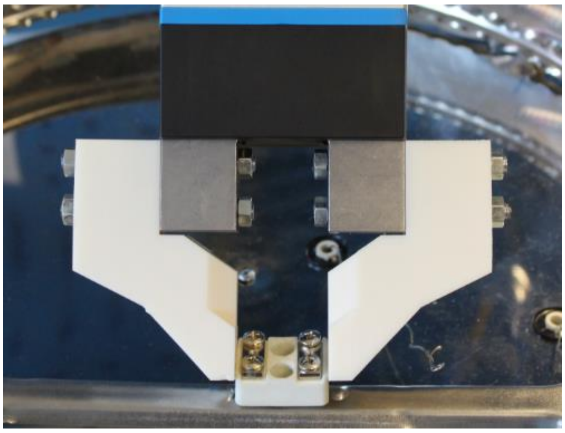


Figure 107: Gripping of cable connector

Because the cable connector is the smallest component, the minimum gripper finger opening will be 2 mm smaller than the cable connector gripping length. The reason for making the distance 2 mm shorter is to give a separate position for when the gripper fingers are fully closed. This is needed so that the system can detect when the gripper has failed to grip a component, by either shortage of components or another kind of error.

F.5.2.1.2 Gripping regulator E-REG

Regulator is the widest component that our system will be gripping. Like with the cable connector, when deciding how to grip the regulator, we considered the total component gripping length. To make it as short as possible, the group decided to grip the regulator as in Figure 108. This leads to the gripping length for the regulator to be 4.5cm.

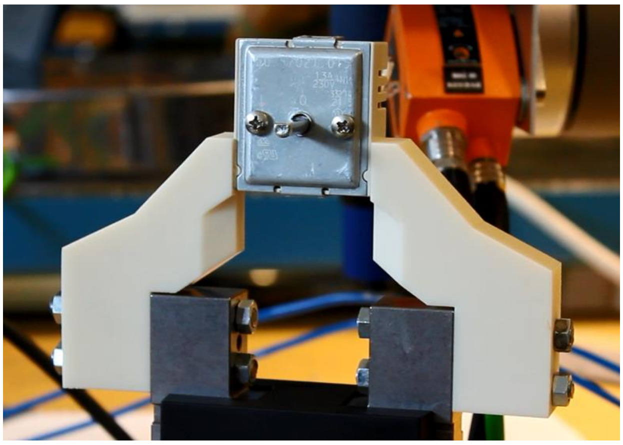


Figure 108: Gripping of regulator

The maximum gripper finger opening shall be at least 6 mm wider than the length of the regulator gripping stroke. The reason for making the distance 6 mm wider is to compensate for the possible uncertainty when gripping the component. Regulators will be fed by a feeding rail which could present a small variation in regulators position. To compensate for this we need the extra stroke length.

F.5.2.1.3 Gripping vacuum tool

Since the gripping tool will be lifting the whole griddle, the gripper fingers shall be strong enough to withstand the weight of the vacuum tool and the griddle combined. The design of fingers and the design of vacuum tool gripping point are very co-dependent. This ensures a solution that secures a precise vacuum tool gripping and a secure and well controlled operating.

F.5.2.1.4 Sketch

The group had a long discussion about how the gripper fingers should be designed. All of the group members agreed that the design should be a mix of retention and friction gripping. The components that the gripper will grip are light, and so friction gripping should be more than enough to move them safely. When it comes to the vacuum tool, the total weight of it with the griddle can be close to 8 kilograms, which is quite a lot. To grip and hold the vacuum tool, the gripper fingers needed retention like design.

Another important design feature was precise gripping of the vacuum tool gripping point. We decided to design the gripper fingers so that they can slide into the gripping position when gripping the gripping point, in case vacuum tool orientation has slightly changed. At the same time, this design will provide stable handling of the tool by interlocking gripper fingers and vacuum tool gripping point.

F.5.2.1.5 Forces applied to the gripper fingers

F.5.2.1.5.1 *Lifting forces*

The gripper is supposed to lift objects ranging in weight. When designing the gripper fingers it was important for us to make an estimate of the heaviest objects weight. This was needed so that simulations could be performed on the fingers. This would ensure that the material and design that we have chosen is right for our gripper fingers.

The lightest object our gripper fingers will be lifting is the cable connector I-IR and the heaviest object will be the vacuum tool that has the griddle attached to it. This means that the minimum vertical force will be the force needed to lift the vacuum tool with the griddle attached to it.

The weight will be applied as force working on the relevant faces of the gripper fingers. We estimated that the weight of the vacuum tool with an attached griddle will be roughly 8 kilograms. This estimation was made from adding the weight of the griddle and the approximate calculated weight of the vacuum tool. Because of the uncertainty we will perform the tests with 10.2 kg (100N).

F.5.2.1.5.2 *Pressure forces*

The gripper force is dependent on operating pressure and lever arm (gripper finger length). At Meteor they operate their vacuum tools with 6 bar pressure. To see if the gripping force we get from 6 bar is enough, we need to calculate the minimum gripping force needed to hold the minimum vertical force. Gripping forces will be used in the simulation of the gripper fingers, to decide the end material and design choice. Figure 5 shows the gripping force per gripper jaw as a function of operating pressure (2, 4, 6, 8 bar) and lever arm.

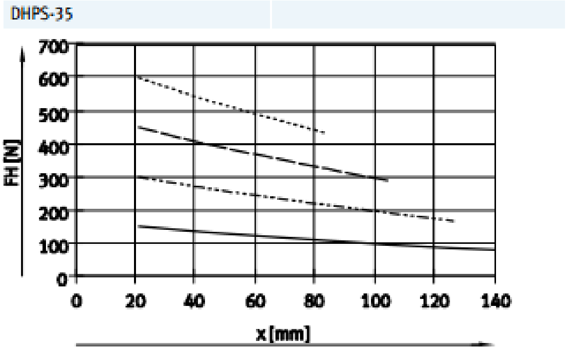


Figure 109: Gripper force graph

Using the formula in Figure 110, we found out that the minimal gripper force needed to hold the vacuum tool with an attached griddle (100N) is 140N. This number is calculated without friction and contact surface taken in consideration. From Figure 109 we can see that gripping force supplied by 6 bar pressure is above 140N, and therefore the simulations will be done with gripping force supplied from 6 bar pressure.

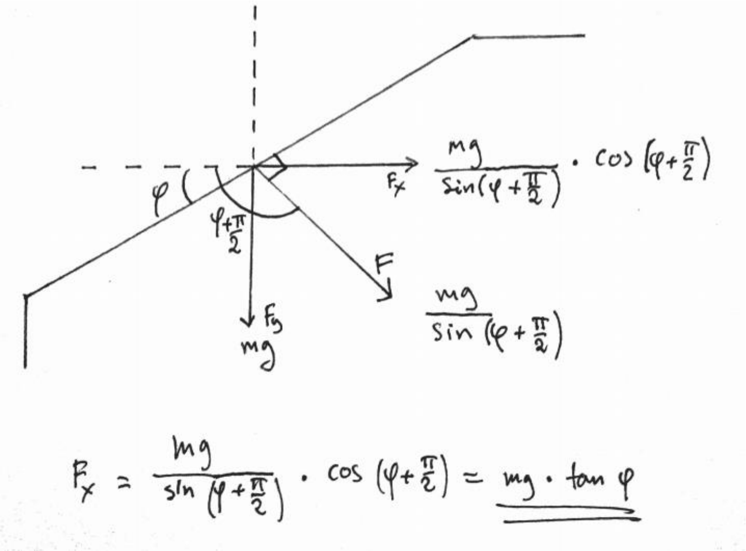

$\frac{mg}{\sin(\varphi+\frac{\pi}{2})} \cdot \cos(\varphi+\frac{\pi}{2})$
F_x
φ
$\varphi+\frac{\pi}{2}$
F
F_g
mg
$\frac{mg}{\sin(\varphi+\frac{\pi}{2})}$
$F_x = \frac{mg}{\sin(\varphi+\frac{\pi}{2})} \cdot \cos(\varphi+\frac{\pi}{2}) = \underline{\underline{mg \cdot \tan\varphi}}$

Figure 110: Minimum gripping force formula

Just to be safe, the group performed a test where we used 3D printed fingers and 3D printed vacuum tool gripping point, to simulate the gripper lifting 10 kg. The gripper was supplied with 4 bar pressure and the gripper had no problems holding the weight.

F.5.2.1.6 Material choice

Aluminium is widely used as material for gripper fingers because of the low weight and good corrosion resistance. This is why we considered aluminium as material for gripper fingers. In general pure aluminium is soft and ductile therefore for commercial use, aluminium alloys are used. Aluminium alloys are classified into two groups, casting alloys and wrought alloys. Casting alloys have a low melting point, but also a low tensile strength. Our gripper fingers need to be strong and have a light weight. They will be machined out, therefore an alloy that had good machining properties was desired. Because of the required high strength to weight ratio, wrought aluminium alloys were further considered.

There are 8-series of this alloy, where the first digit indicated the major alloying elements. There are mainly four series that could be viable:

- The 2 series is alloyed with copper that results in high strength and excellent machining characteristics. These alloys have poor corrosion resistance. Our gripper fingers will be used in a closed environment so they will not be exposed to extreme weather. Because of the poor corrosion resistance, products from this alloy are typically coated in some way.
- The 5(alloyed with magnesium) series are non-heat-treatable alloys. They are strain hardenable and have a moderate strength that relates to the amount of magnesium in the alloy. They have high corrosion resistance, and therefore widely used in salt water applications. These alloys are also very suited for welding, and that is shown by the fact that one group of these alloys is used as welding wire.
- The 6(alloyed with magnesium and silicon) series are easy to machine, weld and can be participation hardened. The tensile strength of this series is lower than the 2 and 7 series, but the workability of the alloy makes it widely used. 6061 alloy is one of the most commonly used general-purpose aluminium alloy.

- 7(alloyed with zinc) series are the high strength alloys. They have high tensile strength but somewhat lower corrosion resistance than the 5 and 6 series. This is why they tend to be coated just as the 2 series.

Which wrought aluminium alloy will be used for the gripper finger material, is also very dependent on the availability. We could choose a solution, that theoretically would be perfect by delivering the strength and lightness we need. But when it comes to actually making a product out of that material, the cost and availability might make that solution worse than the other considered alloys. Our choice was limited to the alloys that can be bought and delivered in a short period of time. We researched for companies that specialize in aluminium alloys and machining companies that have aluminium alloy suppliers.

F.5.2.2 Prototype 1

F.5.2.2.1 3D modelling prototype 1

The gripper fingers were 3D modelled in Solidworks. The design was made so that the gripper fingers could be easily fastened to the gripper. The design was calculated so that the gripper has a starting distance between the fingers of 3.1 cm. With the stroke distance of 2.5 cm, the end distance between the fingers will be 5.6cm. The dimensions of the fingers are:

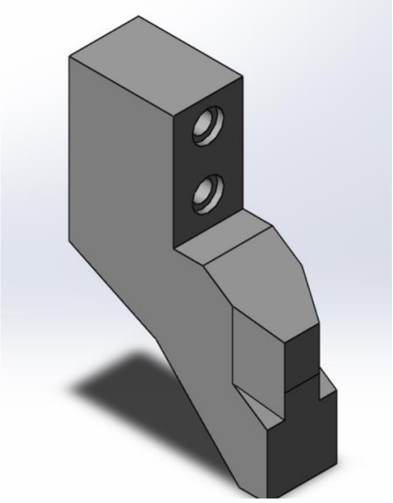
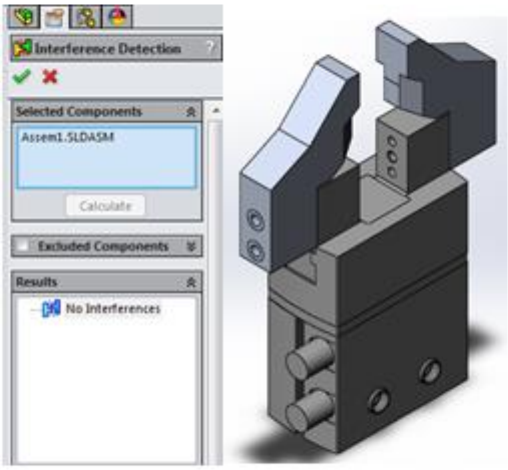


Figure 111: Gripper finger 3D design

- Length:
  - 8.3 cm
- Width:
  - 5.7 cm
- Thickness:
  - 2 cm

To make sure that the fingers fit perfectly for the gripper, we used CAD model of the gripper to make an assembly with the gripper fingers. This reassured us that the fit was perfect. We ran an interference check to make sure no parts of the components were interfering with each other. The fingers were also put into an assembly with the vacuum tool gripping point to check if they match.



Interference Detection

Selected Components

Assem1.SLDASM

Calculate

Excluded Components

Results

No Interferences

Figure 112: Assembly and interference check

F.5.2.2.2 Simulation

F.5.2.2.2.1 *Static test*

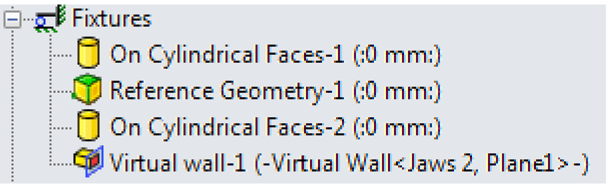

Fixtures
On Cylindrical Faces-1 (:0 mm:)
Reference Geometry-1 (:0 mm:)
On Cylindrical Faces-2 (:0 mm:)
Virtual wall-1 (-Virtual Wall<Jaws 2, Plane1>-)

Figure 113: Fixtures table

We did a static test to check if they can withstand the weight and gripper forces. The first thing we did was to set fixtures on the gripper finger to simulate it being fastened to the gripper.

We used advanced fixtures to make the cylindrical and reference geometry fixtures. They will represent the bolts that will fasten the fingers to the gripper. The cylindrical fixtures are locked in radial translation, simulating that the fingers do not move because of the bolts inserted through these holes. The reference geometry fixture was made to simulate the head of a bolt. The fingers will be kept in place towards the gripper by the head of the bolt, thereby making sure the fingers are fixed in horizontal direction. Reference geometry translation is set to be normal to plane that goes parallel to the gripper finger.

The virtual wall fixture was made to simulate the gripper extending jaw that the gripper finger will be fastened to. The finger will be fastened to this gripper extending jaw, and therefore act as a virtual wall.

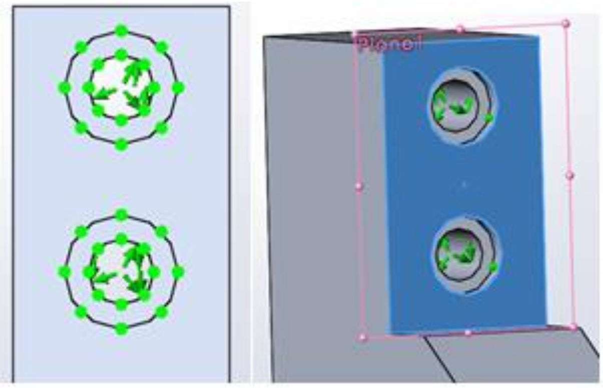


Figure 114: Fixtures

External loads that we applied on the gripper finger in the simulation had to be calculated first. The vertical forces that the fingers will be exposed for, are highest when gripping the vacuum tool. However the biggest stress will be applied when gripping the smallest components, because of the lever arm. These forces had to be spread through all of the surfaces that were affected when a simulation is performed.

The biggest weight that the grippers will be lifting is the vacuum tool with the griddle attached. The weight will be applied as force onto the fingers. This force had to be spread through all of the surfaces that were affected. We also applied gravity load onto the gripper fingers.

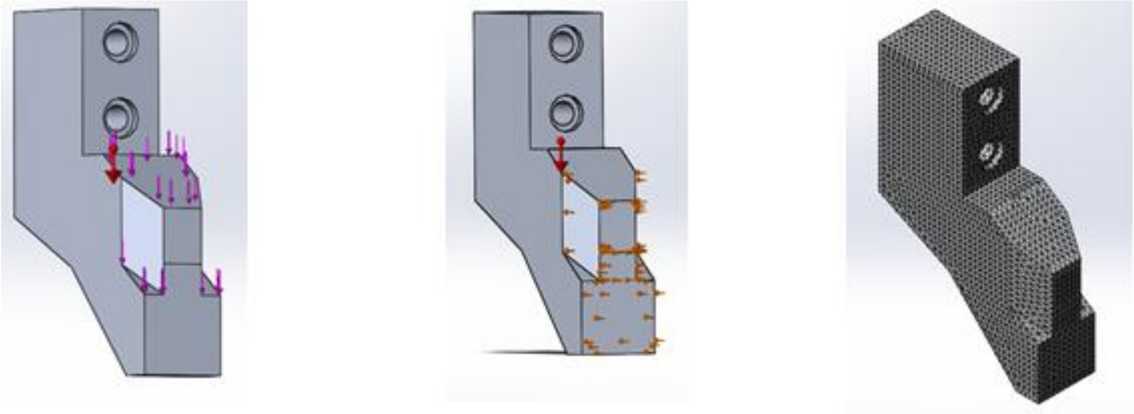


Figure 115: Weight forces, gripper force forces, mesh

The results showed us vonMises stress graph where the materials yield strength and the stress applied to the gripper fingers were shown. Here we saw exactly where the highest stress appeared on the fingers. This information can be used to optimize the design by changing the design in weak areas to strengthen them and remove the unnecessary material in the very strong areas. As seen in Figure 116, the pieces experience most stress around the lower fastening hole and in the narrowing in the back of the gripper finger.

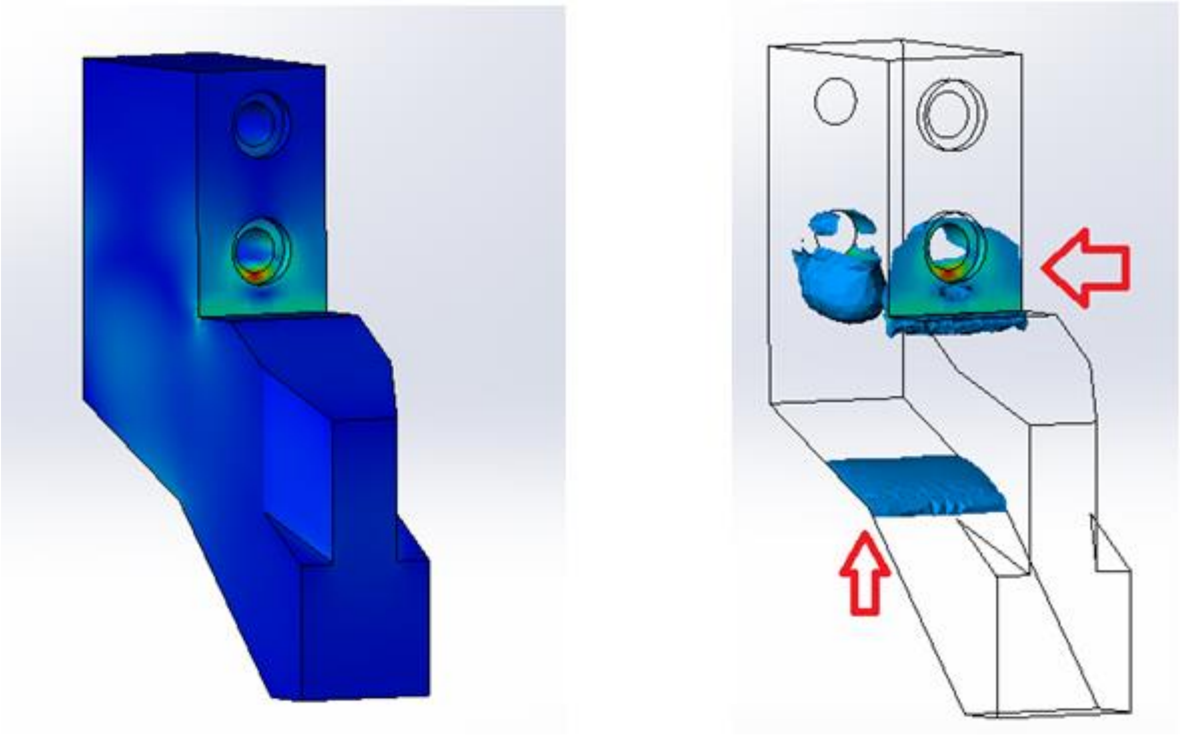


Figure 116: vonMises stress simulation and most affected areas

The displacement graph is used to show the deformation size. As long the results of the vonMises stress graph are under the yield strength, the displacement size in the displacement graph is elastic and therefore not permanent.

The group decided that the factor of safety should be set at 2 for the gripper fingers. This would ensure that the fingers are strong enough to operate up to two times higher forces than the ones we calculated. Factor of safety is important to secure the fingers in case the material has any impurities or other defects that would decrease the yield strength and other properties from the theoretical ones.

The materials used for the simulation was 6063-T4 wrought aluminium alloy. We used this material as an example to find the weak spots of the gripper fingers design. Since the material choice is still under discussion, a new simulation will be performed later on. The main design of the fingers will not change, but the thickness of the fingers will later on be altered to optimize the size and remove excess material.

F.5.2.2.3 3D printed gripper fingers

We used the schools 3D printer to print out the gripper fingers in plastic. This way we could use them as prototypes to see if they fulfil our needs. Usually when 3D printing, the parts can have some excess of material that might need some finishing for the piece to match the drawing perfectly. In this case the holes needed to be widened a little. The M6 screws did not fit in the original 3D printed holes. Also the flat part that is supposed to grip the components was slightly uneven and we used sandpaper to make them as even as possible. All in all the fingers fit perfectly on to the gripper. After they were fastened, they were ready to undergo tests to see if the design fulfils the requirements of object gripping.

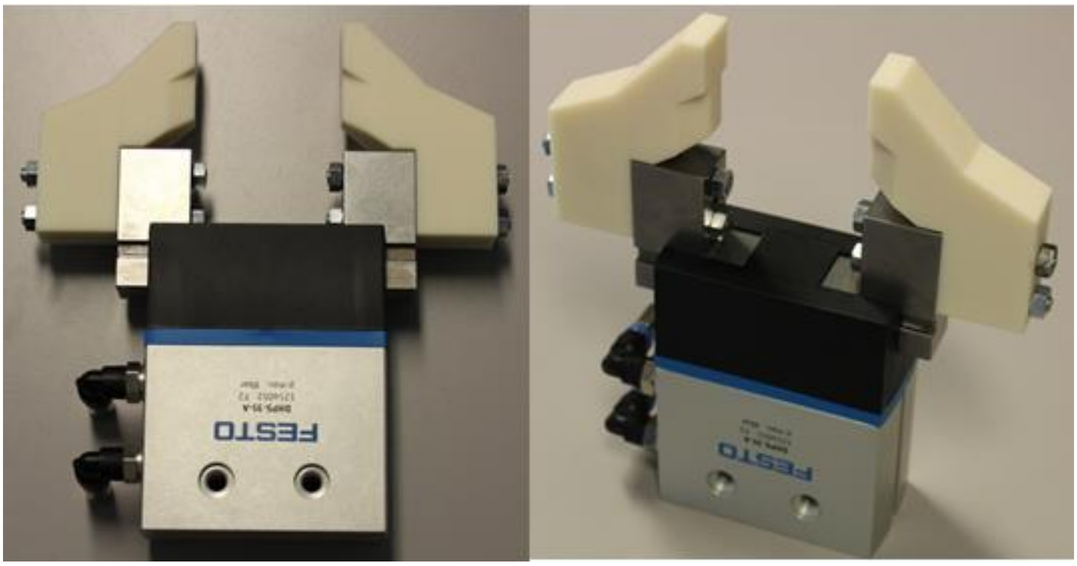


Figure 117: Gripper finger prototype

F.5.2.2.4 Testing by using the prototype

The prototype fingers were used to test if the gripper can grip the components in the planned way. The fingers were first tested to see if they can support the gripping force from the gripper. They were tested with gripping pressure of up to 4 bars. The gripper fingers showed a slight elastic deformation, but since no plastic deformation was seen, the fingers were approved to operate with this pressure by the group members. The gripping force needed to hold a cable connector I-IR and a regulator M-REG is a lot smaller, but since the gripper fingers do not have any extra friction pads on them, this gripping force was chosen just so that the gripper would not drop the components under testing.

Then they were used to grip and move components from the start to the end position, which was their fastening position. The design of the fingers proved to be good and the tests were all successful. One thing that was noted was the thickness of the gripper fingers. If the fingers are any thicker it could be a problem of gripping and placing components.

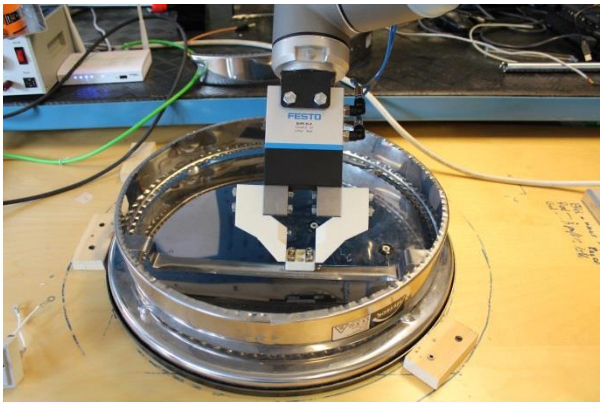


Figure 118: Cable connector testing

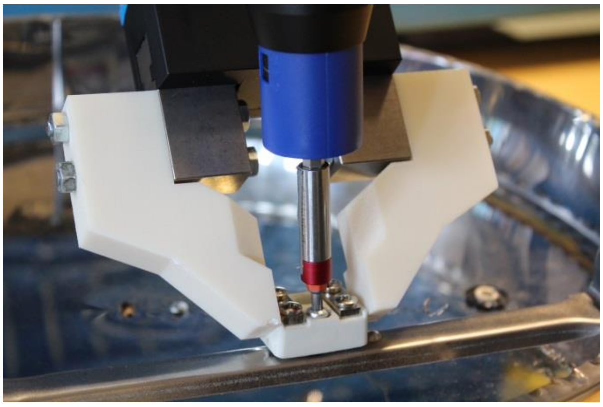


Figure 119: Cable connector fastening

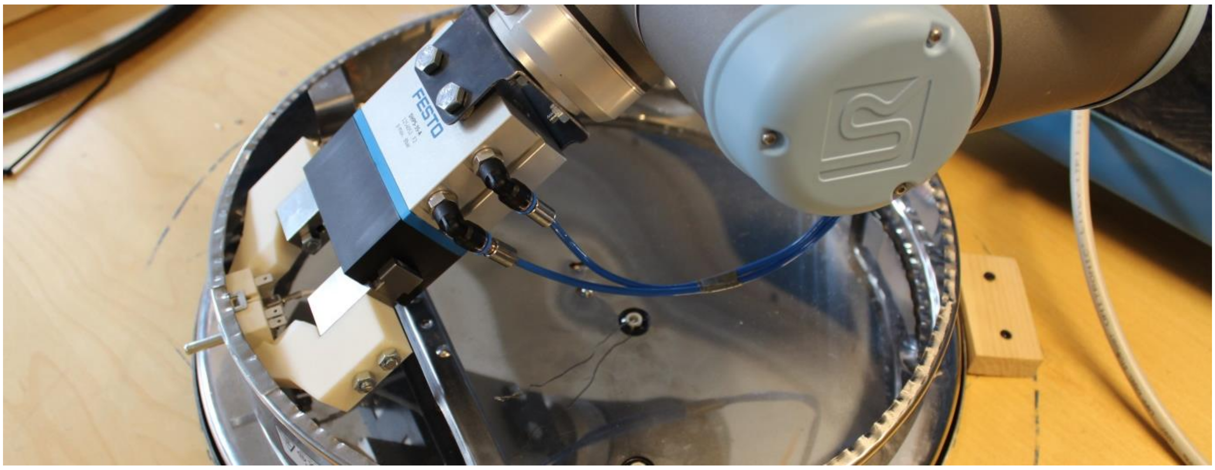


Figure 120: Placing regulator

F.5.3 Conclusion

F.5.3.1 Material

As mentioned earlier, the material choice depends on availability and price. From the materials that fulfil the needs, only the one with the best cost to strength ratio will be chosen. Only the thickness of the material will be affected by the material choice. The gripper fingers will be simulated with the chosen material, to check for the optimal thickness before they are produced.

F.5.3.2 Friction pads

Friction pads will be used to give gripper fingers more friction. This will increase the object moving safety factor.

F.5.3.3 General thoughts

The main objective was to design a pair of gripper fingers that could be used to grasp several objects. Our design fulfils this need and we are satisfied with our design. With the provided precision and safe gripping of the objects, the gripper fingers will be sufficient for automated assembly at Engmark Meteor. The simulations that were done helped to locate the weak spots of the design. This information will be used in further development and testing of the gripper fingers.

F.5.4 References

1. Figure 106[10.05.2015]
   http://www.omega.ca/prodinfo/images/gripper_jaw.gif
2. Anodising Aluminium – Choose The Right Aluminium Alloy for your Anodising [16.05.2015]
   http://eloksal.no/kjekt_vite_om_/content_3/filelist_51b748f2-c6fd-485f-87a5-b81d8000f3b1/1402049253144/legeringsegenskaper.pdf
3. ALUMINIUM DATA [15.05.2015]
   http://www.bloomet.com/pdf/pdf-aluminum.pdf
4. WIKIPEDIA. Aluminium alloy [18.05.2015]
   http://en.wikipedia.org/wiki/Aluminium_alloy#Wrought_alloys
5. Figure 109[18.05.2015]
   http://www.festo.com/cat/no_no/data/doc_engb/PDF/EN/DHPS_EN.PDF

F.6 Starlock

Abstract

This document contains a description of the developing phase of the SL/M-REF.

Contents

Abstract **266**

Contents **266**

Figures **266**

F.6.1 Introduction **267**

F.6.1.1 Components 267

F.6.1.1.1 Ceramic insolation ring I-IR 267

F.6.1.1.2 Reflector plate M-REF 267

F.6.1.1.3 Fastening ring SL 267

F.6.2 Development **267**

F.6.2.1 Design 267

F.6.2.2 First batch 268

F.6.2.3 Second batch 269

F.6.3 Conclusion **270**

Figures

Figure 121: SL/M-REF_4.75_4.35_120d 268
Figure 122: SL/M-REF_4.95_3.8_40d 268
Figure 123: SL/M-REF_5.0_3.27_15d 268
Figure 124: SL/M-REF_5.5_3.95_35d 269
Figure 125: SL/M-REF pieces from first batch 269
Figure 126: SL/M-REF_5.5_3_20d 269
Figure 127: SL/M-REF_5.5_5_35d 270
Figure 128: SL/M-REF_5.5_6_special 270
Figure 129: SL/M-REF_6_5_40d 270
Figure 130: SL/M-REF_6_special 2 270

F.6.1 Introduction

While visiting Meteor for the first time, we observed the assembly process of the griddle. In one of the stations, you were supposed to fasten a ceramic insolation ring (I-IR) to a reflector plate (M-REF). To do this, they used a specific fastening ring (SL) that would hold one side of the I-IR, and lock it in place against the M-REF. The group had an idea to change the shape of the hole in the M-REF, making the I-IR sit in place without the need of extra equipment. This would make both the process faster and eliminate the cost of the SL. The solution was easy: shape the holes in a way that would hold the I-IR. There were two important factors that would decide the shape of the new holes. These new holes will be referred to as SL/M-REF from this point on.

F.6.1.1 Components

F.6.1.1.1 Ceramic insolation ring I-IR

This component is meant to insulate the heating element from the reflector plate M-REF so that the griddle does not short circuit.

Dimensions:

- Height:
  - 8 mm
- Bottom:
  - 9 mm in diameter
  - 16 mm in diameter

F.6.1.1.2 Reflector plate M-REF

The reflector plate M-REF is meant to reflect heat from the heating element towards cooking top M-CT.

F.6.1.1.3 Fastening ring SL

Fastening ring SL is meant to hold the ceramic insolation ring I-IR fastened to the reflector plate M-REF.

F.6.2 Development

When developing the SL/M-REF, we had to look closer into the SL and the M-REF. The materials of these components are different, which gives them different characteristics. The SL is made out of a lot thinner piece of metal than the M-REF. It has a shape that is designed to lock the I-IR in place, without receiving visual plastic deformation. The locking pins usually bend towards insertion direction and hold the I-IR with the force produced from elastic deformation of the locking pins. As mentioned, M-REF is made out of a thicker piece of metal, which makes it act differently when put in the same situation as SL.

F.6.2.1 Design

When designing the SL/M-REF, we had to keep in mind the deformation factor that could occur during the process of inserting the I-IR through the SL/M-REF. Deforming the M-REF could weaken the fastening mechanism, which could lead to the SL/M-REF failing to hold the I-IR in place. The SL/M-REF is also supposed to hold the I-IR without damaging it. To prevent major deformation during fastening process and to insulate the M-REF from the E-HE (heating element), it is crucial to have a design that meets these requirements. Locking mechanisms that we wanted to try ranged from triangle to trapezoid. A big variation in design is needed during the start of this project. This way we will discover the most optimal shape of the locking pins. After that, small adjustments will be made to the working design, to make it a perfect fit for the process.

F.6.2.2 First batch

The first batch of SL/M-REF consisted of holes with small triangles, which are either pointy or filleted, as locking pins. This design choice was made to check if the locking pins, made out of M-REF material, could withstand the force during the process of inserting the I-IR. Four designs were sent to Meteor to be cut out. The test pieces arrived about half a week later and were ready for testing.

The tests were executed by putting the I-IR under the test piece, so that the center of the I-IR is aligned with the center of the SL/M-REF. The tests failed and all of the designs were rejected. The reason for this was that the locking pins were plastically deformed, which lead to bad holding capability. Also the M-REF received plastic deformation from the pins getting deformed. The tests showed that design of longer pins was needed, plus that the diameter of the holes could be bigger, which could decrease the chance of M-REF deformation.

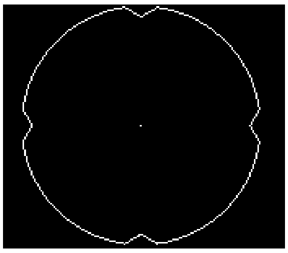


Figure 121: SL/M-REF_4.75_4.35_120d

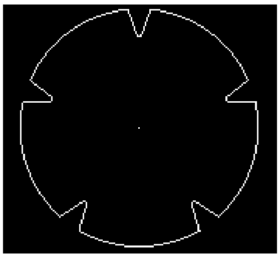


Figure 122: SL/M-REF_4.95_3.8_40d

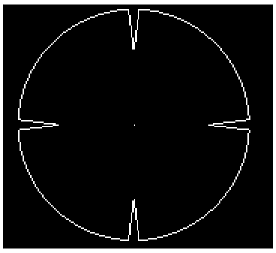


Figure 123: SL/M-REF_5.0_3.27_15d

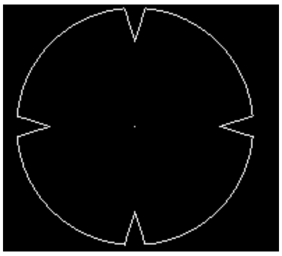


Figure 124: SL/M-REF_5.5_3.95_35d

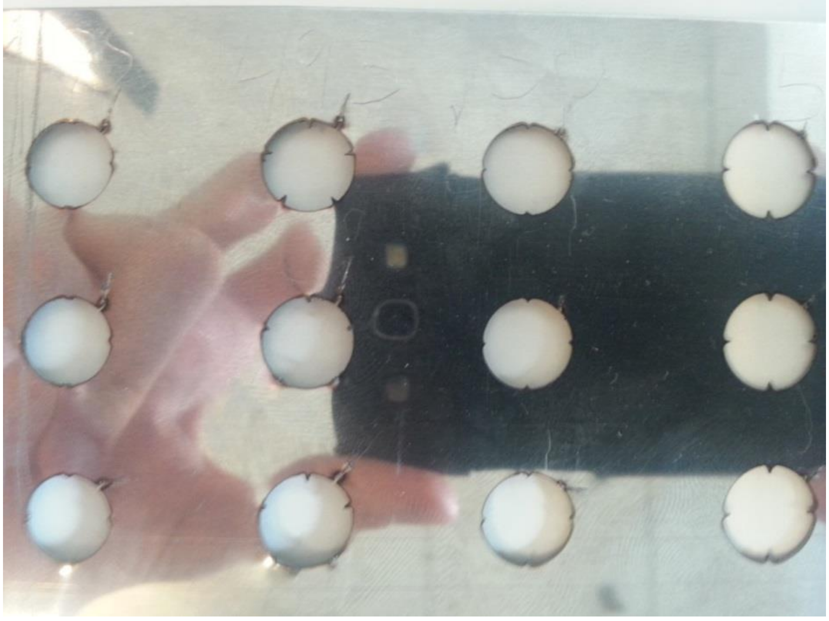


Figure 125: SL/M-REF pieces from first batch

F.6.2.3 Second batch

The second batch of SL/M-REF consisted of longer, pointy triangles and trapezoids. The decision to go over to longer fastening mechanisms was, as mentioned before, to ensure a better fastening and reduce plastic deformation of the M-REF. This time five different designs were sent to be cut out. We have talked with the customer and neither of the designs was a viable solution which could go into further development.

The customer himself has tested several of his own designs but neither of them were viable. The process of inserting the I-IR caused plastic deformation to the locking pins.

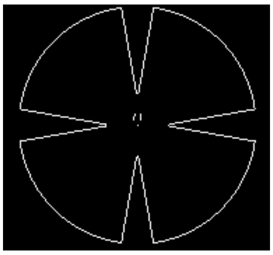


Figure 126: SL/M-REF_5.5_3_20d

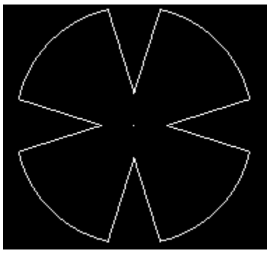


Figure 127: SL/M-REF_5.5_5_35d

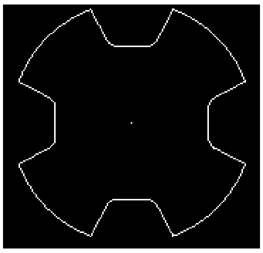


Figure 128: SL/M-REF_5.5_6_special

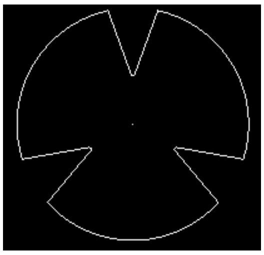


Figure 129: SL/M-REF_6_5_40d

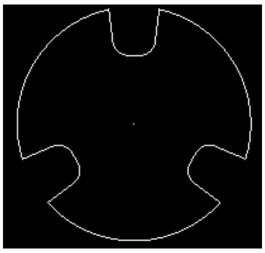


Figure 130: SL/M-REF_6_special 2

F.6.3 Conclusion

The developing process of SL/M-REF is time consuming. The process consists of a lot of testing and coming to the right solution by trying out different concepts. The fact that we have to send the drawings to Oslo, for then to wait for the test pieces to be delivered makes the development a very slow process. It would take less time if we had easier access to the laser cutter. The development was concludes because a lot of on sight testing was needed.

F.7 Vacuum gripper tool

Abstract

This document addresses our solution to expand the use of the robot to handle more tasks. This include managing to grasp the reflector M-REF and the bottom plate M-BP to move griddles in and out of the workstation.

Contents

Abstract 271

Contents 271

Table..271

Figures 271

F.7.1 Introduction..272

F.7.2 Development 272

F.7.2.1 Analysis 272

F.7.2.2 Suction cup setup 273

F.7.2.3 Gripper connecting point 273

F.7.2.4 Choice of suction cups.. 273

F.7.3 Conclusion 274

F.7.4 References..274

Table

Table 7: Suction cup specifications [1], [2]... 274

Figures

Figure 131: Vacuum gripper tool, 4 cups 272
Figure 132: Vacuum gripper tool, 3 cups 272
Figure 133: Suction cup setup, reflector M-REF.. 273
Figure 134: Suction cup setup, center-to-center 11cm, 10mm/square.. 273
Figure 135: Gripper connecting point 273
Figure 136: Gripper finger 273

F.7.1 Introduction

The parallel gripper we chose for gripping the smaller components is unsuited to grasp the reflector M-REF and the bottom plate M-BP directly. To handle these components, we chose to utilize vacuum grippers because they use it at Engmark Meteor today with great success.

The vacuum gripper tool they use consists of a simple 2,5mm steel plate and three or four suction cups mounted to it. The steel plate is a cut-out from the laser cutter with eight holes in it; four to mount suction cups and four screw holes to attach it to the robot. This is something Håvid Engmark from Engmark Meteor has created himself.

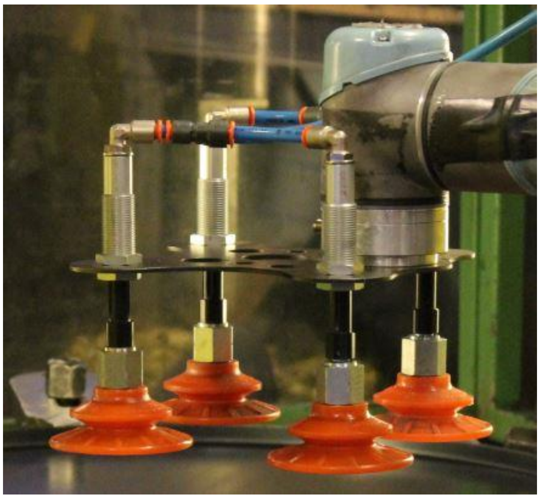


Figure 131: Vacuum gripper tool, 4 cups

When the first UR-10 arrived at Engmark Meteor, Håvid wanted to test its capabilities. He therefore attached the vacuum gripper tool to the robot, picked up one of the large griddles and waved it around at full speed. The griddle did not fall off and the steel plate did not buckle. This showed that a simple 2,5mm steel plate has sufficient strength to handle the forces generated by waving a 9,2kg griddle around.

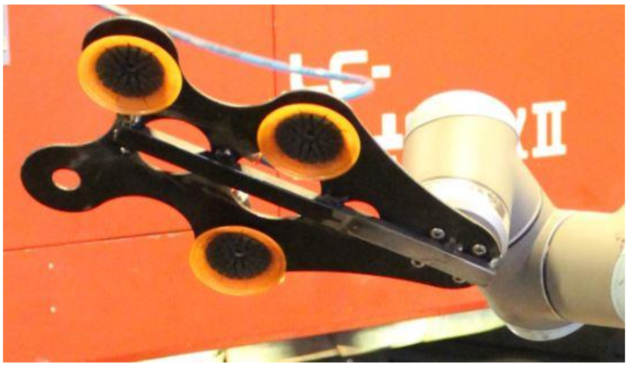


Figure 132: Vacuum gripper tool, 3 cups

Our purpose for the gripper is not to handle the 9,2kg griddle nor to wave it around. The task for the vacuum gripper tool is to pick and place loads up to 6,5kg, which is the total weight of the small griddle. We could have copied the design completely to use it as a gripper, but we wanted to create a more agile gripper design that in addition did not have to be permanently attached to the robot.

F.7.2 Development

F.7.2.1 Analysis

The analysis that led to the choice of separated grippers was how the mobility of the robot would be affected by having both the parallel gripper and a vacuum gripper tool attached at the same time. We quickly realized that not only would the mobility of the robot be heavily affected, but also the agility of the grippers. Another problem was that the total end effector would become so big that it would get in the way of the other robot arm. This led to the conclusion of not having both type of grippers permanently attached.

We chose the parallel gripper to be the permanently attached gripper because it requires the most mobility when handling the small components. Instead of having to change between the grippers by attaching and detaching both of them and to have a separate mechanism to connect the different grippers to the robot, the parallel gripper will both act as gripper and as a tool connector. This is the most effective solution in relation to not having both tools permanently attached.

F.7.2.2 Suction cup setup

We chose to use a suction cup setup like the one in Figure 132. The reason we chose not to have a fourth cup, like in Figure 131, is that the heating wires coming out through the insulation rings will get in the way and be squished. The reason we chose not to have two cups, is that the grip would be unstable if not managing to grip at the centre of gravity.

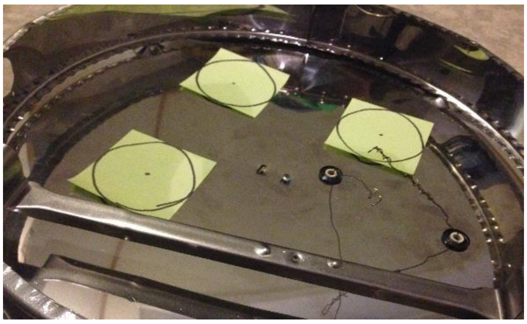


Figure 133: Suction cup setup, reflector M-REF

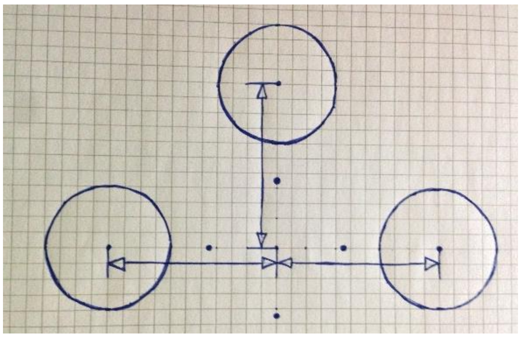


Figure 134: Suction cup setup, center-to-center 11cm, 10mm/square

By placing the suction cups on a radius of 11cm, we can utilize suction cups with a radius up to 40mm and still have some room for error when gripping the reflector.

F.7.2.3 Gripper connecting point

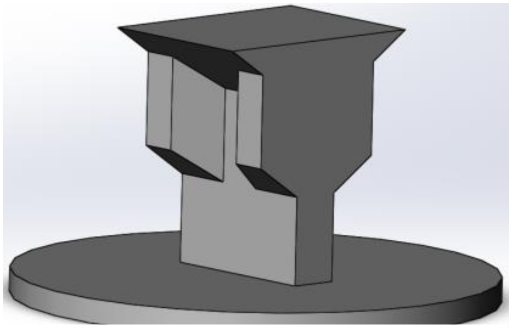


Figure 135: Gripper connecting point

To connect the vacuum gripper tool to the parallel gripper, we had to create a handle for the parallel gripper to hold on to. For the two to have a consistent fit and to give stability, we design the gripper fingers [3] and the handle to interlock. Figure 135 and Figure 136 shows how the parts may look like. The handle has angled walls that will act as a slot and the upper parts of the gripper fingers are formed to act as wedges and slide right in. This allows some room for error when gripping the vacuum gripper tool.

To get the least amount of torque applied to the handle and gripper finger, the position of the handle has to be at the centre of gravity. The centre of gravity will in this case not be at the centre between the three suction cups, but at the centre as if there were four cups. The connection point will be attached to the metal plate like the four dots in Figure 134 shows.

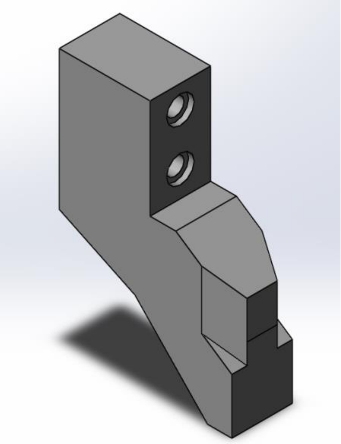


Figure 136: Gripper finger

F.7.2.4 Choice of suction cups

The suction cup type in shown in Figure 131 is from the BFF-series from piab [1]. There are 3 different sizes under 40mm radius; 15mm, 23mm and 35mm. Figure 132 shows the FCF-series from piab [2]. This model has two types of cups that are 40mm or smaller; 27mm and 40mm.

Model	Lifting force at .6 bar (N)	Lifting force at .9bar (N)	Radius (mm)
BFF30P	23 (24’)	30 (27’)	15
BFF40P	45 (43’)	60 (56’)	23
BFF60P	82 (77’)	106 (122’)	35
FCF25P	19 (19’)	29 (28’)	27
FCF35P	34 (42’)	50 (58’)	40

Table 7: Suction cup specifications [1], [2]

Values valid for steel sheet with surface finish R_a and 2-3g/m^2 of press oil. ‘) Dry metal sheet

F.7.3 Conclusion

This vacuum gripper tool solution is still a concept at an early stage in the development process. The next step will be to make a prototype and test for strength and centre of gravity. There will also have to be made a decision of what type of suction cups, vacuum ejectors and control unit to utilize for the tool. In addition to decide the final design and choice of components for the vacuum gripper tool, there has to be made a stand for the tool to be placed when not in use.

F.7.4 References

1. piab. *FCF – Flat Concave Friction.* [17.05.2015] https://www.piab.com/Products/suction-cups/application/oily-sheet-metal/fcf--flat-concave-friction-35125-mm/fcf35p-fcf50p-fcf75p-fcf100p-fcf125p/
2. piab. *BFF – Bellows Flat Friction.* [17.05.2015] https://www.piab.com/Products/suction-cups/application/oily-sheet-metal/bff--bellows-flat-friction-40110-mm/bff40p-bff60p-bff80p-bff110p/
3. *Product Development Document Gripper Fingers*

F.8 Vision development

Abstract

This document will describe the direct path to why we chose the vision system we did and the parts we had to develop around that system.

Contents

Abstract **275**

Contents **275**

Figures **275**

F.8.1 Introduction **276**

F.8.2 Development Processes **276**

F.8.2.1 Initial decisions 276

F.8.2.2 Choice of hardware 276

F.8.2.3 Point Clouds 276

F.8.2.4 Work station and development platform 277

F.8.2.5 Initial testing 278

F.8.2.6 Development environment 278

F.8.2.7 Data filtration and plane meshing 279

F.8.2.8 Two-dimensional object recognition 280

F.8.2.9 The IFM sensor 281

F.8.2.10 Artificial Neural Networking 282

F.8.2.11 3D object recognition 282

F.8.2.12 2D to 3D coordinate mapping 283

F.8.3 Conclusion **284**

Figures

Figure 137: Point cloud example 277
Figure 138: Kinect test picture 279
Figure 139: Normal test picture 280
Figure 140: 2-D Hole recognition with Kinect 281
Figure 141: Hole recognition with IFM 282
Figure 142: 3D object recognition 283

F.8.1 Introduction

Our vision system is described in two parts. The first part is a general application for Mektron to apply in some of their many projects. This means the overall approach is creating a system that is as versatile as possible. The second part is creating a specific solution for Engmark Meteor to assist in the improvement of their production.

F.8.2 Development Processes

F.8.2.1 Initial decisions

Starting out we had no experience with vision systems. Our development thus started with an extensive research period.

To make a decision about what sort of vision system we should be using in our system we first had to find the options and what specifications that would affect us the most. Using vision to control a robot is a new field where development is ongoing at a rapid rate. Due to this, we decided on taking a broad approach with “fast-fail”. “Fast-fail” is when you take a system, test it as early as possible in a very minimalistic form to see whether it complies with requirements. If the requirements are met, further tests are done until you are left with a system that fulfils the requirements you have.

As vision systems are such a broad field we decided to approach the problem the same way. Not knowing what to look for when finding a system is a complicated task, especially since many of the systems are currently under development and not “marketed” in a way that is easy to understand without prior knowledge.

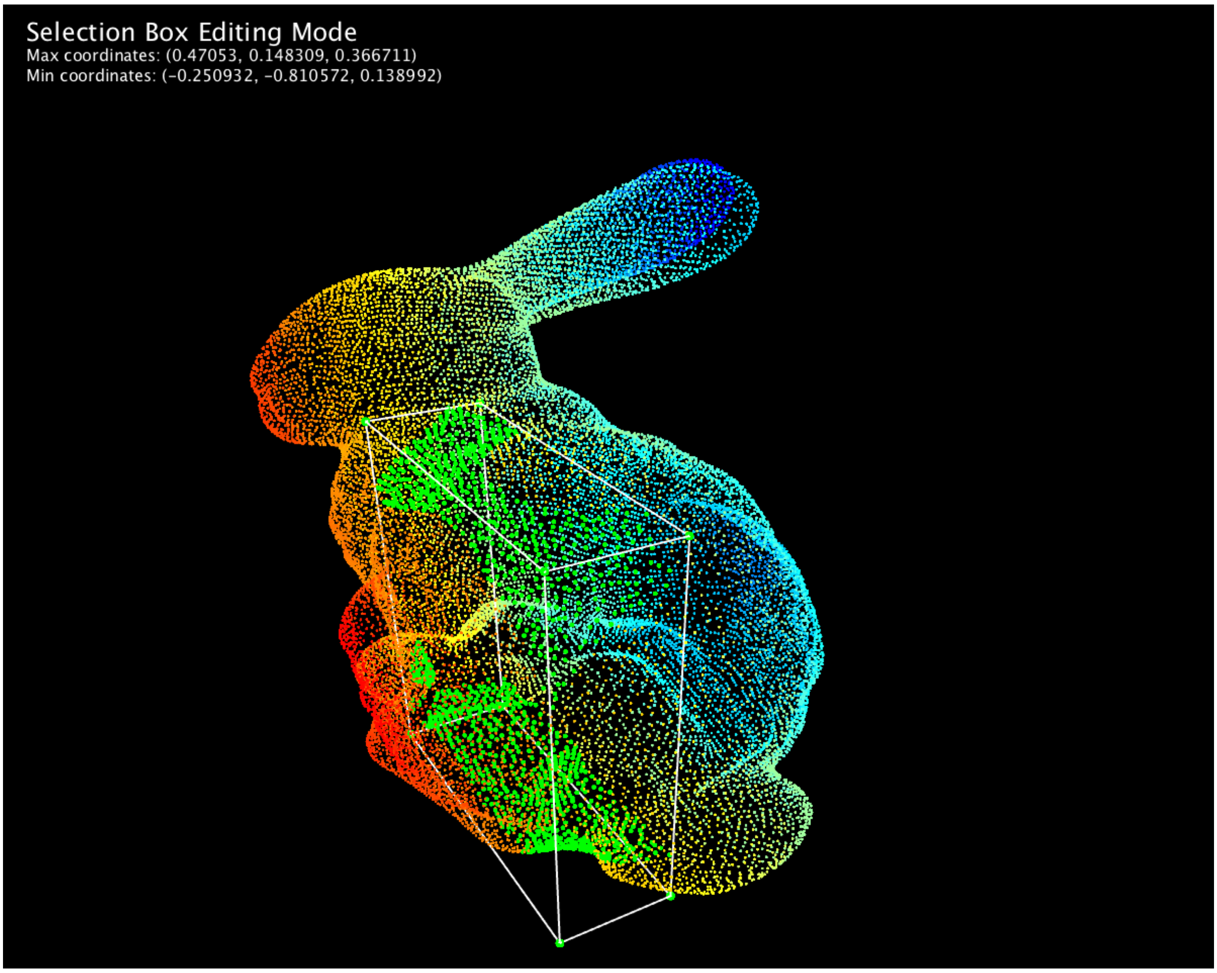
F.8.2.2 Choice of hardware

Since this part of our project would be software based we would have to use existing hardware. Vision is a new up-and-coming market in both robotics and other industries (like games). Xbox released its first version of the Kinect sensor in 2010, opening 3D vision to hobbyists around the world. Since then we have seen a rapid development. Many other companies have also joined the trend, releasing their own cameras and developers kits. With this commercialisation comes a natural drop in prices, lowering the bar to start developing.

These cameras are gradually becoming better and deliver precision most thought unthinkable only a couple years ago. This opens up a whole new range of usages for the systems. With this precision it is now possible to aid robots in their tasks, specifically, in production.

F.8.2.3 Point Clouds

The output of a 3D camera is a set of points, generally referred to as a “Point Cloud”. A point cloud can be seen as an image where each pixel has a depth value.



Selection Box Editing Mode
Max coordinates: (0.47053, 0.148309, 0.366711)
Min coordinates: (-0.250932, -0.810572, 0.138992)

Figure 137: Point cloud example

Early in the project it became clear that PCL is the library of choice for many independent programmers. It has a good community and good documentation that proved vital for rapid learning.

F.8.2.4 Work station and development platform

Initially we had to assess what would be required of the work station. Analysing Point Clouds requires a lot of processing power. The data sets are often huge and unlike a regular image, they often need reorganizing to be useful. We also faced the dilemma with underestimating hardware needs vs. wasting too much money on processing power. For us the former seemed like the worse of the two (having to stop development and wait for new hardware could prove fatal). Hardware were then picked based upon rough estimates and recommendations found online. The result was a powerful workstation that should be able to handle most we could throw at it.

In parallel with this we also looked upon different hardware solutions. The Kinect v2 was an obvious choice as Mektron proposed it. The Kinect uses time-of-flight, meaning it shoots out a grid of IR-laser and measures the time it takes for the beams to reflect off a surface and return to picked up by a sensor. We also looked up Intel RealSense and other different stereo cameras. Stereo cameras work by having two cameras with fixed distance between them. The images they take are compared and key points are recognized in both images. Using these key points and their difference in the two pictures, you can calculate the perspective.

With the setup complete we had to choose a development platform. The choice was between some flavour of Linux and Windows. Where Linux offers customizability, Windows offer a stable platform with well tested tools. Seeing as both the Kinect and Intel RealSense had official SDK's for Windows, the choice became rather obvious. While the open-source community offers a lot of help, we simply did not have time to learn a new environment.

With Windows also comes the Visual Studio IDE, offering many useful tools that help in debugging.

F.8.2.5 Initial testing

Initially we sought out to test as much as possible. Following the "fail-fast" model our goal was to get as much as possible up and running, learning in the process, before making a decision on what system to focus on. First up was installing the official SDKs. Given that the two system represent the two most common ways to obtain 3D information, we also would get good insight in those areas.

The installation went seamlessly, giving hope but also limiting options. We initially thought of using the depth to recognize holes – a sudden increase in depth would indicate a hole. But in both cases the resolution simply wasn't there. Both types of sensor need a minimum distance to operate, limiting how close we could place the sensor. We also had problems with the products reflective surface. When the Kinect measures time-of-flight, it does so by analysing the light reflected. With a reflective surface we got measurements from both the product itself and whatever surface could be seen in its reflection.

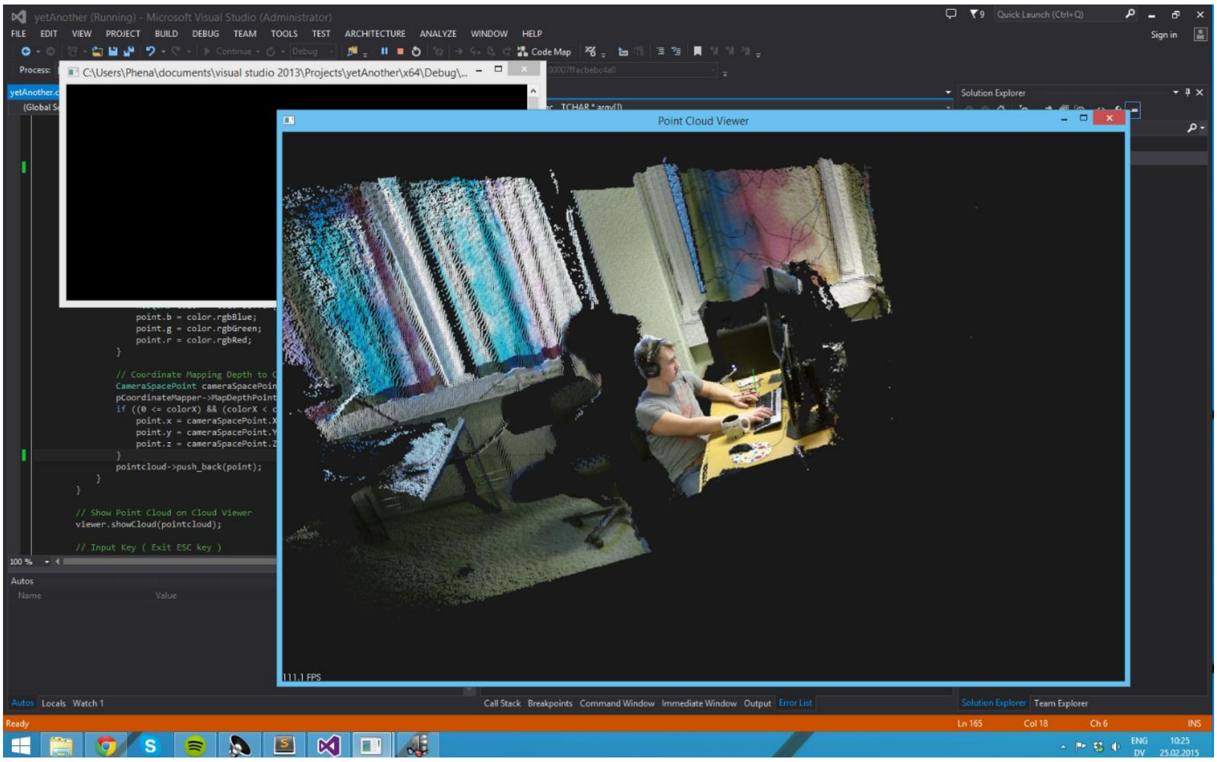
After some testing we also found the stereo vision system to be far less accurate.

Since we were looking into many possibilities with the vision system; not only related to the product but also recognizing the components that goes in it, we decided to keep a main focus on the Kinect.

F.8.2.6 Development environment

Next we would need to find some Libraries to help us. From earlier research we knew that PCL would help us a great deal. Properly building it however, proved to be far more difficult than we initially thought. PCL depends upon several third party libraries, like boost, FLANN, Eigen, QT and VTK. While there exists all-in-one installers of precompiled libraries, these were outdated. Having next to no experience in C++ and working with linkers, a lot of learning went into the process. After a lot of trial and error, research while waiting for compilations we managed to get a working system up and going.

Next in line was interfacing with the Kinect. While the Kinect SDK comes with functions to do this, they were not compatible with PCL. Luckily the Kinect community are very active and helpful. Unluckily the Kinect v2 had not been on the market long enough for an easy interface to be made. Combine this with minimal experience with both Visual Studio and C++ in general, a lot of time went into testing and searching for solutions. Eventually we found a grabber (essentially and interface between the Kinect SDK and PCL) on a Japanese blog, which proved to work. After some hassling with linkers and library includes, we finally got results.


yetAnother (Running) - Microsoft Visual Studio (Administrator)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Continue
Debug
Code Map
Process:
C:\Users\Phena\documents\visual studio 2013\Projects\yetAnother\x64\Debug\...
Point Cloud Viewer
111.1 FPS
Solution Explorer
100 %
Autos
Name
Value
Autos Locals Watch 1
Call Stack Breakpoints Command Window Immediate Window Output Error List
Solution Explorer Team Explorer
Ready
Ln 165
Col 18
Ch 6
INS
ENG
10:25
25.02.2015

```
                point.b = color.rgbBlue;
                point.g = color.rgbGreen;
                point.r = color.rgbRed;
            }

            // Coordinate Mapping Depth to C
            CameraSpacePoint cameraSpacePoin
            pCoordinateMapper->MapDepthPoint
            if ((0 <= colorX) && (colorX < c
                point.x = cameraSpacePoint.X
                point.y = cameraSpacePoint.Y
                point.z = cameraSpacePoint.Z
            }
            pointcloud->push_back(point);
        }
    }

    // Show Point Cloud on Cloud Viewer
    viewer.showCloud(pointcloud);

    // Input Key ( Exit ESC key )
```

Figure 138: Kinect test picture

F.8.2.7 Data filtration and plane meshing

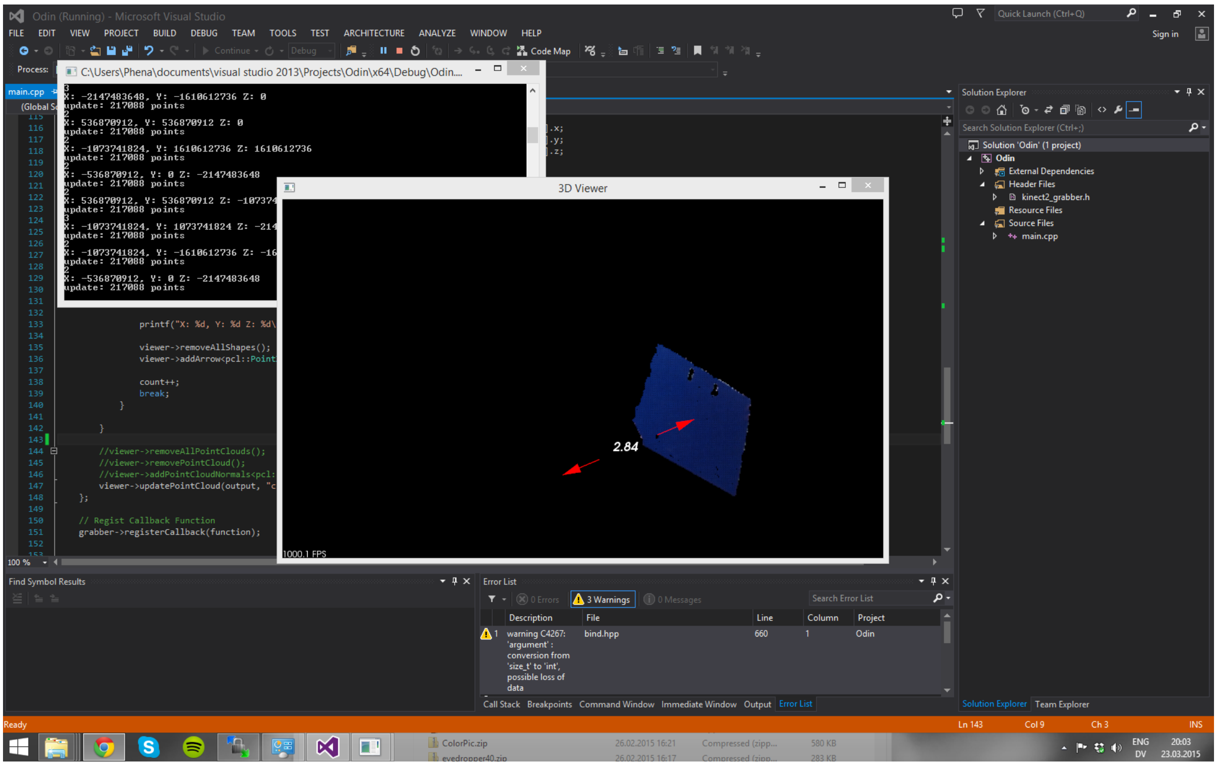
The first thing we wanted to find was a method to extract a plane and its normal. This is a feature which can be used in many applications. Most interesting (after some enthusiastic discussions with the project owner) is bin picking. Bin picking is when you have a set of objects in an unordered box, and manage to recognize one that can be picked up by a tool (e.g. a robotic arm). Most tools for grabbing objects require a flat surface for the gripper to hold on to. Recognizing surfaces (and other geometries) is also a key component for object recognition in general.

To effectively check whether two points are part of the same plane, you calculate a “normal” in each point. The normal in a point is decided by looking upon its neighbours within a certain distance and calculate a plane estimation. The normal of this “plane” is then the normal of that point. This is repeated for each point. The normal are then compared. If the normal of two neighbours are within a certain range, they can be said to belong to the same plane/surface. This way you can put points into groups where a group is a surface.

While this sounds easy in theory, we quickly ran into problems. The Kinect has a resolution of 640 * 480 (not evenly distributed) giving a total of 217088 points to analyse. When calculating normal you first need to organize these into a tree structure, and then search though each one and its neighbours. This is time-consuming. It became apparent that filtering the data first, extracting the minimal amount of points needed for a given accuracy was paramount.

For testing we decided to filter by colour. By removing the points that did not correspond with a certain colour (in our case, blue) we were left with a significantly reduced data set. With the data now reduced to < 10.000 points, we were able to mesh the surface in adequate time and display only that group of points.

To find the normal and centroid of that group/plane we iterated the set of points and calculated the average of the position data of both the point itself, and its normal. This was the result:



C:\Users\Phena\documents\visual studio 2013\Projects\Odin\x64\Debug\Odin....

```
3
X: -2147483648, Y: -1610612736 Z: 0
update: 217088 points
2
X: 536870912, Y: 536870912 Z: 0
update: 217088 points
2
X: -1073741824, Y: 1610612736 Z: 1610612736
update: 217088 points
2
X: -536870912, Y: 0 Z: -2147483648
update: 217088 points
2
X: 536870912, Y: 536870912 Z: -107374
update: 217088 points
3
X: -1073741824, Y: 1073741824 Z: -214
update: 217088 points
2
X: -1073741824, Y: -1610612736 Z: -16
update: 217088 points
2
X: -536870912, Y: 0 Z: -2147483648
update: 217088 points
```

```
                printf("X: %d, Y: %d Z: %d\

                viewer->removeAllShapes();
                viewer->addArrow<pcl::Point

                count++;
                break;
            }

        }

        //viewer->removeAllPointClouds();
        //viewer->removePointCloud();
        //viewer->addPointCloudNormals<pcl:
        viewer->updatePointCloud(output, "c
    };

    // Regist Callback Function
    grabber->registerCallback(function);
```

0 Errors | 3 Warnings | 0 Messages

	Description	File	Line	Column	Project
1	warning C4267: 'argument' : conversion from 'size_t' to 'int', possible loss of data	bind.hpp	660	1	Odin

ColorPic.zip	26.02.2015 16:21	Compressed (zipp...	580 KB
eyedropper40.zip	26.02.2015 16:17	Compressed (zipp...	283 KB

ENG DV 20:03 23.03.2015

Figure 139: Normal test picture

F.8.2.8 Two-dimensional object recognition

In parallel to this we also looked into object recognition in two dimensions. While a 3D camera has many applications, 2-dimensional imaging have been along a lot longer, as have libraries dedicated to processing them. This combined with the problems with the reflective surface of the product, practically making the Kinect unfit for the task, we had to look into other solutions for aiding the robot in the products assembly. OpenCV is a powerful and very easy to use library that offers a ton of features. By first making a grayscale of an image, filtering by threshold based upon its neighbours (to further enhance edges) and using a Hough-transform we were able to locate circles/holes in the product easily.

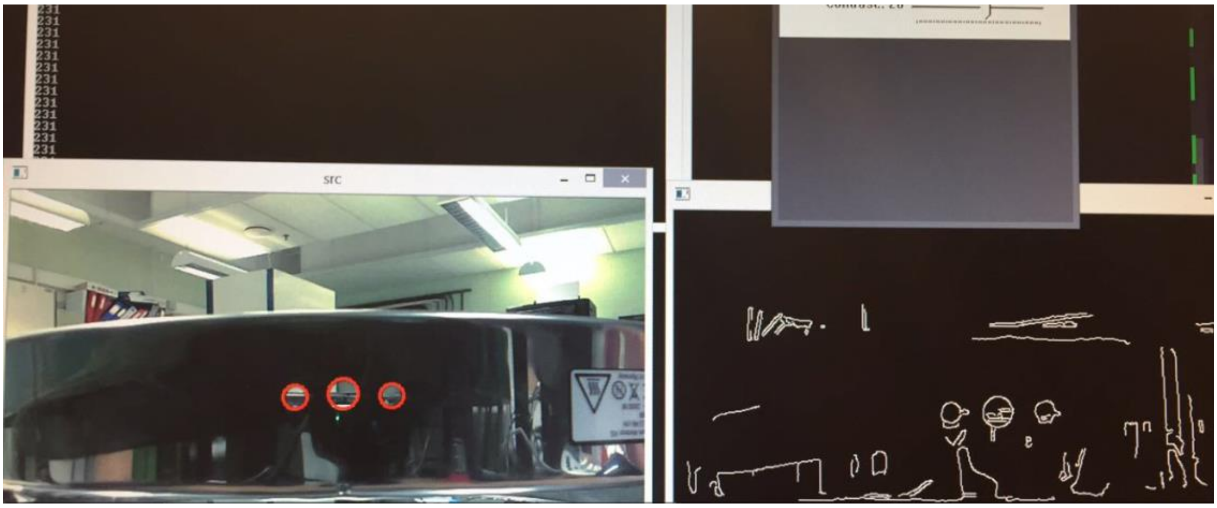


Figure 140: 2-D Hole recognition with Kinect

F.8.2.9 The IFM sensor

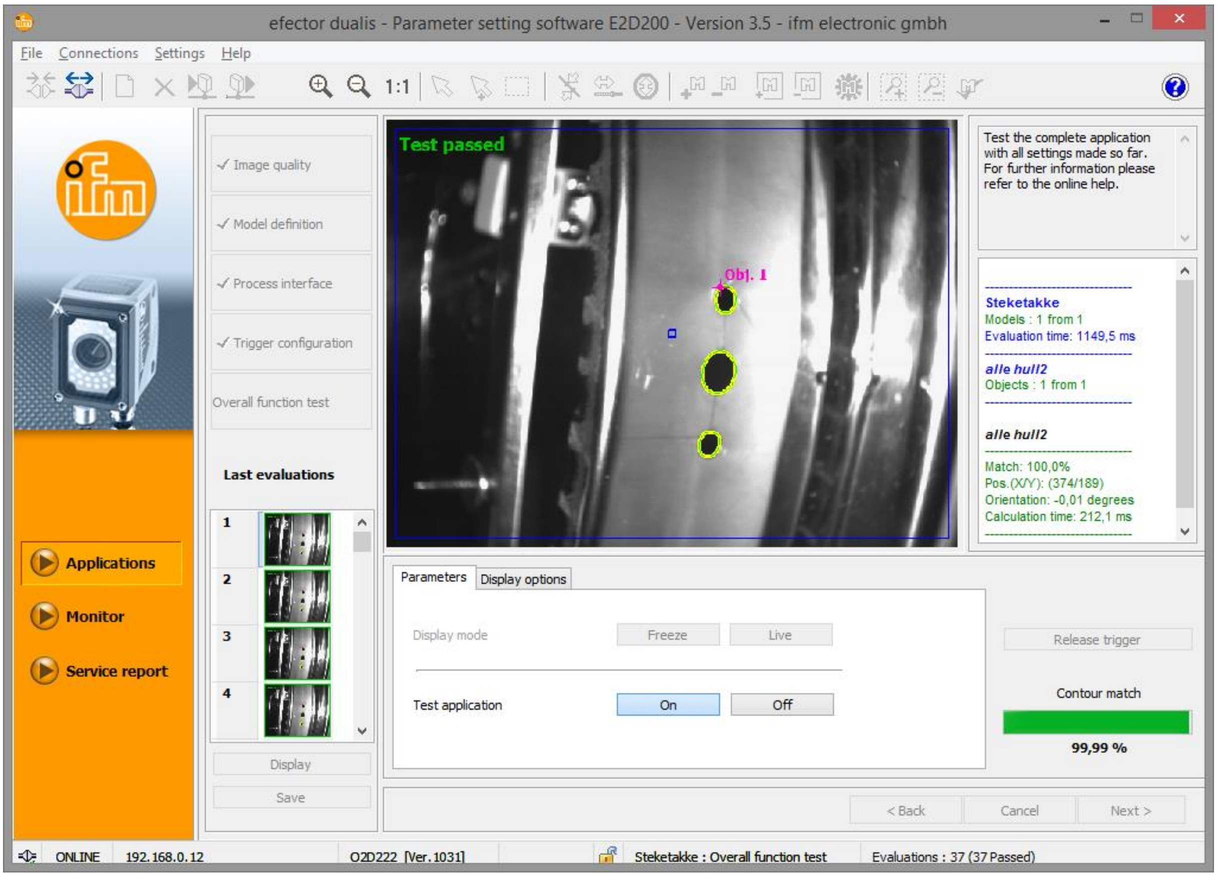
Meanwhile we did this, we also came into contact with Sivilingeniæør J.F. Knudtzen A.S. They are a distributer for a company named IFM, which deliver sensors that do both imaging and processing. After a meeting with information and demonstration of their products we were convinced that this was worth looking further into. With these sensors we could not only eliminate the need for an external computing station, they were also robust and made for industrial use. They were also well within the budget (the price would become lower than a camera plus an image processing station). We were allowed to borrow a couple of these sensors for further testing.

The O2D222 is an object recognition sensor. While its internals are not openly available, it becomes clear from the configuration software that it works by edge detection (similar to what we achieved with OpenCV). While its configuration options are limited, we thought it to be within what we needed to successfully recognize what we needed on the product (mainly screw holes, but we also looked into finding the orientation and position of the product itself).

We also got to borrow the O2V102 which is a pixel counter. It works by creating a threshold and counting the pixels in the resulting blobs. We could also analyse the blobs, checking for roundness and size). We were able to find holes with this sensor, but the O2D222 proved to be much more accurate.

At this point we decided to split the vision system into two parts. The practical part that would be delivered to Engmark Meteor, and a theoretical part that Mektron could learn from and utilize in future projects.

Interfacing with the IFM sensor proved to be harder than we initially thought. The sensor have a TCP/IP interface, but it became apparent that IFM's clients usually utilize the digital interface. As we needed the coordinates of the objects recognized, this was out of the question. First we had to set up a network with static IP's for the sensor and configure them properly and then try to establish a connection. After a lot of emails and telephones back and forth with J.F. Knudzen A.S. we got grip of the protocol.



Test the complete application with all settings made so far. For further information please refer to the online help.

Steketakke
Models : 1 from 1
Evaluation time: 1149,5 ms

alle hull2
Objects : 1 from 1

alle hull2
Match: 100,0%
Pos.(X/Y): (374/189)
Orientation: -0,01 degrees
Calculation time: 212,1 ms

Display mode: Freeze | Live

Test application: [✓ On] | Off

Contour match
99,99 %

ONLINE 192.168.0.12 | O2D222 [Ver.1031] | Steketakke : Overall function test | Evaluations : 37 (37 Passed)

Figure 141: Hole recognition with IFM

F.8.2.10 Artificial Neural Networking

We also had a look at artificial neural networking (ANN). When recognizing objects you find that no two objects will be exactly identical. Even if they were, camera uncertainties would still make their projections slightly different. Where most functions are deterministic, ANNs are good at making generalizations. An ANN simulates a brain by having neurons and synapses. The synapses connect the neurons and are weighted (basically an edge map). Each input neuron pass their value to their neighbours. The value is altered by the weight of the connecting synapse. When the information reach the output nodes, the output is compared with wanted output and weights are altered using a recursive derivative. This process is called learning. Once an ANN is sufficiently learnt, it will be able to recognize the objects it is trained to. To benefits of such a system is that it removes the need for a specific algorithm for each object. Once the ANN is there, you can provide it training sets of virtually any object (Within certain limits, foremost the size of the ANN and processing power).

F.8.2.11 3D object recognition

During this process we also continued testing with PCL. Object recognition is a very useful and easily applicable application. The common approach to recognizing objects using 3D is first to use an algorithm to find key points. What defines a key point depends upon the application it will be used for. Typically it means focusing edges and removing points that reside on the same plane. This is best illustrated with an image of e.g. a table. The image contain a lot of data. It tells us where the table is and where the table top is. If we only look at the edges of the same image. We can still see the table top, defining the table's geometry. While we still have a lot of the same data regarding geometry, yet the total data set is significantly reduced.

When you apply the key point algorithm to both the model and the stage (image in which the object resides), the same points should be found. These points can then be linked to each other by e.g. distance. The stage is then searched for points with similar neighbours. An example of this is illustrated here:

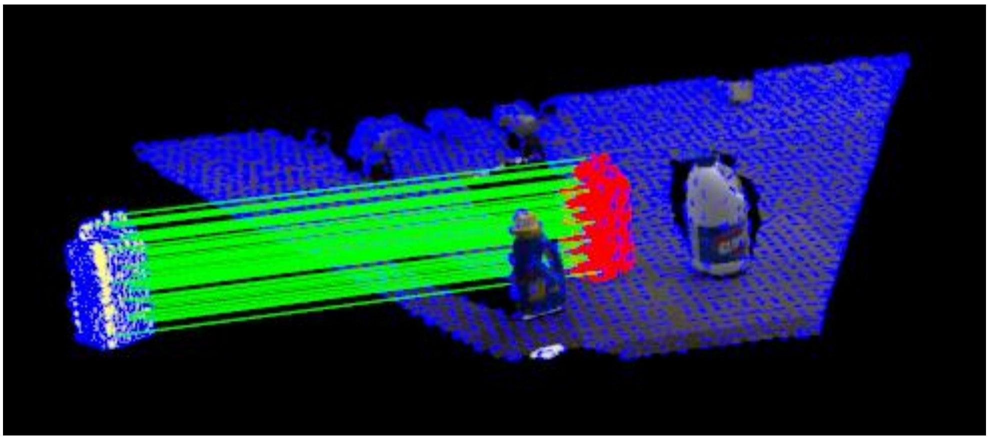


Figure 142: 3D object recognition

Here the key points are highlighted with blue, the model is placed at the left side of the stage, the object found in the stage is highlighted in red and the corresponding points are linked with green lines.

F.8.2.12 2D to 3D coordinate mapping

Since we decided to use a two dimensional camera for our practical application, we had to translate the coordinates from the cameras image plane to real-world coordinates. Since we have full control of where the camera is located relative to the robotic arm (the camera is part of the tool on the robot), we can calculate this. We created a python class that both abstracted the communication with the sensor and translated its output to real-world coordinates. While mapping coordinates to a flat plane was simple enough, mapping to a cylindrical object proved much harder. In a plane you have only have an exponential function mapping the y coordinate, and a linear transform based upon the y coordinate for the x coordinate. When mapping to a cylinder you have a new dimension, z, and all three variables depend upon each other.

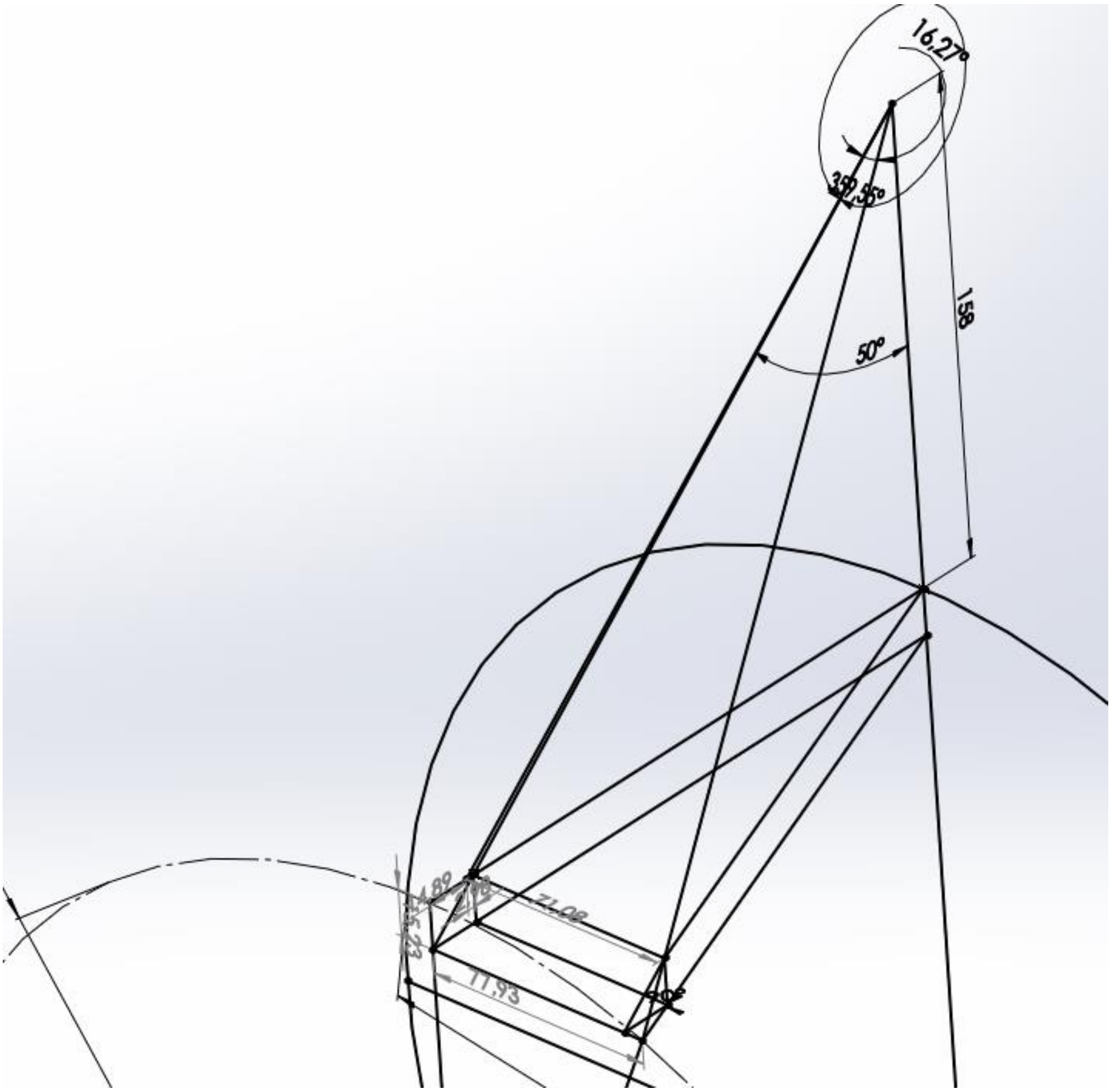


Figure 143: Projection upon a cylinder

After a lot of headache, we carefully wrote out all the equations and a system began to emerge. After a lot of organizing and substitution we were left with a single equation with a single unknown. We had a solution. Euphoria.

F.8.3 Conclusion

The vision system was the biggest uncertainty in our project. While we aimed for a full system which would do everything from recognizing holes to bin-picking we knew we might as well end up with nothing at all. Since none of us had any prior knowledge in the area, it was hard to define exactly how much time to dedicate to researching different concepts and possible solutions. Taking the broad approach that we did, we learnt a lot about vision systems and what to look for. Without this knowledge we would not have been able to properly deduce that the IFM sensor would be sufficient for our practical application at Engmark Meteor. Our project owner, Mektron, has also been very interested in our progress, absorbing the knowledge and conclusions we obtained. Based upon this knowledge Mektron is now able to make a decision whether further development using 3D vision is worthwhile.

We have been able to see the need for such a 3D system ourselves during a visit to one of Mektrons potential clients. We were shown a very manual and repetitive task which could easily be done by a robot – given that a vision system is in place. It is moments like this we feel engineering is really useful.

While we spent more time than estimated learning to compile and link libraries, we are happy with the overall progress. The learning curve has been very steep, for better and for worse, and although we didn't get to delve as deep into everything as we would like, we now know the possibilities.

G. Our work

G.1 Self-reflection

Abstract

This is project group Vi'RA final evaluation of our project.

Contents

Abstract 286

Contents 286

G.1.1 Working as a group...287

G.1.1.1 The good 287

G.1.1.2 The bad 287

G.1.2 The product...287

G.1.2.1 The good 287

G.1.2.2 The bad 287

G.1.1 Working as a group

G.1.1.1 The good

This project has taught us a lot. Despite a lot of initial confusion we managed to stay optimistic throughout the entire project. We have learnt from each other and our respective fields of study. We have studied a lot on our own in subjects we never would have touched elsewise. We have made seen parts of industrial Norway and its problems – and thought of solutions. Most of all we have had fun throughout the entire project, finally utilizing what we have learnt through the years. Everything from advanced system engineering to basic trigonometry.

G.1.1.2 The bad

Things take time! It is frustrating how much time seemingly simple tasks can consume.

We did not invest as much time as we should have early in the process to define our task. This led to some delays and us having to revisit the elaboration.

We should have explored the option of using a second UR arm earlier in the project. This was something we discussed in the group but not with our stakeholders.

G.1.2 The product

G.1.2.1 The good

We are happy with the solution that we have sketched for Engmark Meteor. We did not get as far we had hoped but the solutions we have delivered give a good basis for further work .

Our gripper solution solves a rather complicated assembly task with an elegant and comparatively cheap result.

G.1.2.2 The bad

We were a bit optimistic/enthusiastic when discussing the assignment early on. We had a lot of ideas and wanted make a complete product. We should have limited our delivery and focused on doing delivering well defined and thoroughly tested system.

G.2 Activity breakdown

Abstract

This document gives a detailed overview of the time spent by each group member on each task

Contents

Abstract 288

Contents 288

G.2.1 Detailed time spent 289

G.2.2 Graphical representation of time spent 293

G.2.1 Detailed time spent

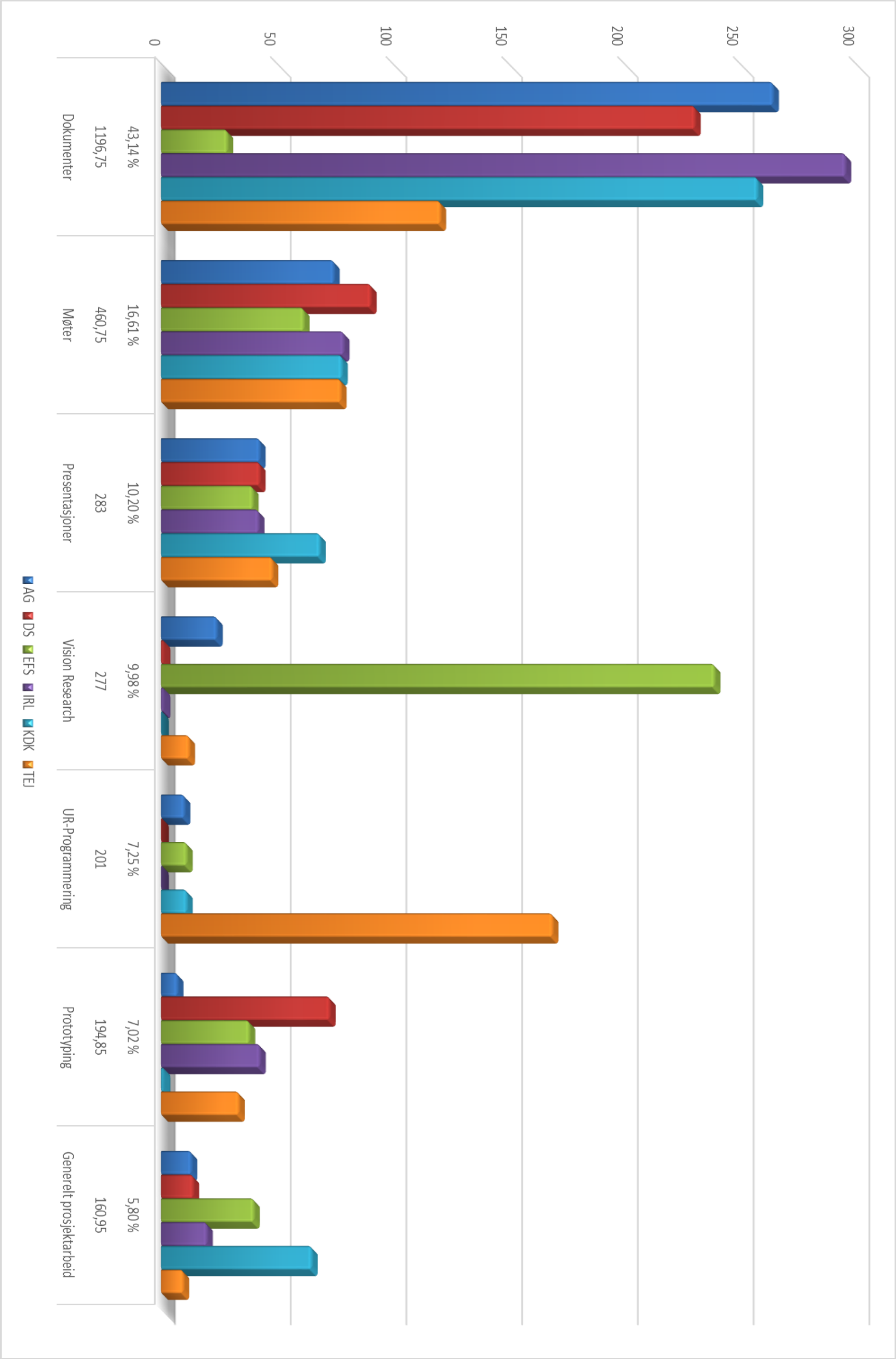
	TOTAL		Anders	Dave	Eirik	Ivar	Kristian	Thor Eirik
			0	0	0	0	0	0
	2774,3		435,25	451	457,5	479,55	481	470
Dokumenter	1196,75		264,25	230,5	28,5	295,5	257,5	120,5
Dokumentstandard	49,5		23	7		11	4,5	4
Dokument, Arkitektur	56		20,5	8			20	7,5
Dokument, Backlog	29,5		22,5	0,5		0,5	2	4
Dokument, Budsjett	11,5		7,5					4
Dokument, Cost-Benefit	28,5						28,5	
Dokument, Designdokument	2						2	
Dokument, Gantt	58,5		28,5	30				
Dokument, Griddle part list	1						1	
Dokument, Kravspesifikasjon	101,5		18,5	19	7	4	23	30
Dokument, Kravspesifikasjon beskrivelse	0,5			0,5				
Dokument, Løsningsalternativer for Engma	25,25		12	2,75		3,5	3	4
Dokument, Mektron automasjonsmetodikk	10		1				9	
Dokument, Prosjektbeskrivelse (A4)	6		6					
Dokument, Produsjonanalyse Engmark	62		11	4	7,5	6,5	28	5
Dokument, Risiko	43,5					42	1,5	
Dokument, Risiko Arbeidsstasjon	18,75					18,75		
Dokument, Test case	63		8	9,5		29,5	13	3
Dokument, Testplan	48,5		0,5		2	40		6
Dokument, Unified process	46		15,5	21		2,5	6	1
Dokument, Use Case	80,75		21,25	21,5		4	27	7
Dokument, UR I/O porter	14						14	
Dokument, Visjonsdokument/oppgavetekst	28		8			13	6,5	0,5

Konsept, Arbeidsstasjon	18		10,5	2		5,5		
Konsept, Strømskinne	2			2				
Konsept, End effector feste	1			1				
Konsept, Få-steketakka-rund-jig-som-treng	28,75					28,75		
Konsept, Mateskinne	7,75		5			2,75		
Utviklingsdokument, Arbeidsstasjon	17		15	2				
Utviklingsdokument, End effector feste	23,5			21		2,5		
Utviklingsdokument, Endring av design stek	35						35	
Utviklingsdokument, Griperfingre	11			11				
Utviklingsdokument, Mateskinne	2		2					
Utviklingsdokument, Regulator Jig	15		1	13		1		
Utviklingsdokument, Starlock	5			5				
Utviklingsdokument, Strømskinne	9		8	0,5		0,5		
Teknologidokument, Filesharing	2							2
Teknologidokument, Griper	129,75		17	40,25		49	5,5	18
Teknologidokument, Kommunikasjonstekn	5					5		
Teknologidokument, Project managing tool	5			3		2		
Teknologidokument, Skrutrekker	27,25			1		22,25		4
Teknologidokument, SolidWorks	4			4				
Teknologidokument, Urscript	17							17
Teknologidokument, Vision	14,5		1		11			2,5
Sluttrapport, Føre inn	8						8	
Sluttrapport, Planlegging/Strukturering	25		1	1	1	1	20	1

Møter	460,75		74,5	90,5	61,5	78,75	78	77,5
Befaring hos Engmark meteor	142		22	31	17	23,5	27	21,5
Gruppediskusjon, Endelig løsning	64,5		11	13	6	10,5	11	13
Gruppediskusjon, Grupperegler	12		2	2	2	2	2	2
Iterasjonsmøte	98		16,5	16,5	16,5	16,5	15,5	16,5
Møte med Håvid	6			2		2,5	1,5	
Møte med Egil	85,25		14,5	16,5	13,5	14,25	11,5	15
Møte med Henning	4			1		1	1	1
Møte med Øyvind	49		8,5	8,5	6,5	8,5	8,5	8,5
Vision Research	277		24	1	239	1	0	12
Research Artifitial Neural Networks	17				17			
Research Rotasjonsmatriser	41,5		21		8,5			12
Research Vision	13,5		3	0,5	9	1		
Research Vision, Brennvidde	25				25			
Research Vision, Filtre	19				19			
Research Vision, IFM	42				42			
Research Vision, Kalkulasjoner i planet	18				18			
Research Vision, Kinect	22,5			0,5	22			
Research Vision, Kinect korellasjon	19				19			
Research Vision, Kurvet translasjon	14				14			
Research Vision, Massesenter	6				6			
Research Vision, Normaler	14				14			
Research Vision, Objektgjenkjenning (hull)	11				11			
Research Vision, OpenCV	10				10			
Research Vision, RealSense	4,5				4,5			
Prototyping	194,85		7	72,75	38	42,6	1	33,5
Prototype Arbeidsstasjon	21		2	3,5		6	1	8,5
Prototype End effektor feste	23,5		2,5	9		8		4
Prototype Griper	7,5					7,5		
Prototype Griperfingre	28,25		2	11,25		3,5		11,5
Prototype Regulator Jig	16,6			11		5,6		
Prototype Sarg-bli-rund	3					3		
Prototype Skrutrekker	13,5		0,5			8		5
Prototype Testbenk vision	43,5				38	1		4,5
3D modellering, End effector feste	7			7				
3D modellering, Kamera feste	6			6				
3D modellering, Starlock	8,5			8,5				
3D modellering, Strømskinne	7,5			7,5				
3D modellering, Steketakke	1,5			1,5				
3D modellering, Griperfingre	7,5			7,5				
UR-Programmering	201	0	10	0	11	0	11	169
URscript	107		10				9	88
Urscript, Sirkulær bevegelse	25,5							25,5
Urscript, Position feedbak	3							3
Urscript, Remote control	49							49
UR simulator	5,5						2	3,5
UR og Vision interfacing	11				11			

Generelt prosjektarbeid	160,95		13	13,75	40	19,7	65	9,5
Admin, Generelle HBV-dokumenter	4						4	
Admin, Mailing	13,5		0,5	3,5	2	1,5	6	
Admin, Oppfølgingsdokument	11,5						11,5	
Admin, Prosjektkalender	3,5						3,5	
Admin, Referat	9,95			1,75		5,7	2,5	
Admin, Timelister	16,5		1,5	1			12,5	1,5
Admin, Oppsummere timer	13						13	
Forelesning Olaf	44		7,5	6,5	4	7,5	12	6,5
Nettside, utvikling	28				28			
Nettside, Blog	17		3,5	1	6	5		1,5
Presentasjoner	283		42,5	42,5	39,5	42	68,5	48
Presentasjon 1, Dokumentarbeid	24		4	4	4	4	4	4
Presentasjon 1, Forberedelser	105		14,5	14,5	17	17,5	28	13,5
Presentasjon 1, Se andre presentasjoner	7,5		1	2		1,5	2	1
Presentasjon 2, Dokumentarbeid	35,5		5	5	4	3	6	12,5
Presentasjon 2, Forberedelser	96,5		15,5	15,5	14,5	14	21,5	15,5
Presentasjon 2, Se andre presentasjoner	10,5		2,5	1,5		2	3	1,5
Presentasjon 3, Planlegging	4						4	

G.2.2 Graphical representation of time spent



G.3 Potential improvements beyond this project

Abstract

This document will discuss possible future extensions of our delivered product.

Contents

Abstract 294

Contents 294

1. Introduction 295

2. Vision 295

2.1. Engmark Meteor 295

2.2. Mektron 295

3. System expansions 296

4. Further automation of the assembly line 296

G.3.1 Introduction

Our system provide solutions to many of the problems brought to us by our project owner and their client.

The vision system delivered to Engmark Meteor can be made easier to configure. As it is a python script will have to be added to configure the parameters of the camera translation. Mektron have received a lot of information about the current status of vision systems, what is possible and rough insight in the development of such a system.

G.3.2 Vision

G.3.2.1 Engmark Meteor

While intuitive to a programmer, the python script that handles the interface between still requires manual configuration in the script. An alternative to this is having a configuration file that the script reads. This file could be written by an intuitive program, e.g. a webpage with a simple form to add different configuration modes. These modes could then be select from the UR script by sending simple socket messages. This would make any user interaction with the interfacing computer obsolete. In effect the user would have an isolated system where all interaction can happen over Ethernet.

G.3.2.2 Mektron

3D vision is undoubtedly the future of robotics. With the coming of cheaper and better hardware, software will soon follow up. Already we can see hobbyists making systems that challenge expensive industrial solutions. Our estimate is that within three-four years, we will see a lot of new vision systems on the commercial market.

If we were to suggest directions to do further research within 3D vision, it would first and foremost be within filtering/key point algorithms. What took us most by surprise was the amount of calculations required compared to 2D image processing. A good key point extraction algorithm will drastically reduce the number of points to compare in an object recognition setting. We imagine such an algorithm where basic geometric shapes (spheres, boxes, cylinders e.g.) is recognized. A model would then consist of one or more geometric shapes. The model would then be defined by these shapes, their information (length, height, radius…) and their relation (position) compared to each other. You would then reduce the amount of data to compare by a drastic amount.

We also imagine using ANNs to help the process. If you have a CAD model or a scan of the object you want to recognize, a program could easily make a training set out of it. Given that the model is now described as a combination of geometric shapes, the amount of input nodes are greatly reduced. This means an easier network which will train faster on smaller data sets. This can then be used to create a very modular solution, where the only configuration needed to implement e.g. bin-picking is to scan a model, run an algorithm to find its geometric components, set a point and a vector for the robot to “attack” (place the gripper, suction cup e.g.) and let the ANN handle generalizations, making sure the success rate is high enough. We imagine this can be done to the point where implementation will only require minimum amounts of programming, if any at all.

G.3.3 System expansions

The part that puts restrictions on the possibility to complete the automation of the assembly is for it to be able to **mount the power inlet** and **connect all the electrical wiring**. Solving these challenges would enable the system to handle the entire assembly of the griddles.

Our analyses show that it would require an advanced vision system to be able to identify the loose wires and grip them precisely. The gripper itself would most likely have to be specialized for solving just this task. Another option would be to redesign the current design for automation but this is not something we have explored adequately.

In-assembly **buffer**: The possibility to do the same assembly action on several griddles in series. The assembly system will have several griddles waiting to be processed and it will repeat the same assembly action on each of them while stacking them up for the next operator or assembly action. This gives the operator more flexibility in operating the system, as it would be possible to prepare a stack of griddles for assembly and then perform another task while the system works.

G.3.4 Further automation of the assembly line

Packing is the final part of the assembly line and the most likely candidate for further automation. This task is repetitive and has little variation making it suited for automation. It is also sensible to expand the system out from the existing parts to tie the system together into a cohesive assembly line.

Any further automation will most likely have to include new stations and the possibility to move batches between different stations will be necessary for an efficient system. **Conveyers** would enable the system to move the (pallets with) griddles between stations.

G.4 General Vision Conclusion

G.4.1 Abstract

This document will describe the design of our vision system that will be delivered to Engmark Meteor.

G.4.2 Contents

G.5.1 Abstract 297

G.5.2 Contents 297

G.5.3 Introduction 298

G.5.4 The IFM sensor 298

G.5.4.1 Hardware 298

G.5.4.2 Configuration 298

G.5.4.3 Interfacing 298

G.5.5 Abstraction layer 298

G.5.6 Translation 298

G.5.6.1 Translation functions 299

G.5.6.2 Precision 301

G.4.3 Introduction

The IFM sensor O2D222 is an all-in-one solution. It packs both imaging and processing into a robust package designed for industrial use. Its output is customizable and one of its options is to send the objects location and orientation over a socket. These coordinates are in the image plane and need to be translated to real world coordinates before they are useful. To do this and to wrap the functions of the camera most commonly used we have created an abstraction layer in python. This layer will handle all communication with the sensor as well as translate the output from it, presenting it to the user as a set of coordinates with the cameras focal point at the center.

G.4.4 The IFM sensor

G.4.4.1 Hardware

The sensor chosen for our application is the O2D222 which is an "object recognition sensor. Although the software is not available it is clear that it edge detection to recognize objects. The sensor has a basic TCP/IP interface with several commands.

The sensor also have internal lighting, making it less dependent on external light conditions.

G.4.4.2 Configuration

Configuration is done through a software that can run on any Windows machine. The software connects to the sensor and you can thereafter "program" the sensor. This is includes configuring the cameras exposure rate, the edge detection threshold and match percent for a successful match. It also has a "model builder" which makes creating new models easy and intuitive.

G.4.4.3 Interfacing

The sensor has two cable connections. One cable provides power and also have 5 programmable pins. These pins can be set by configuring the sensor. Typical applications for this are digital outputs for whether an object is recognized or not. The connection is a M12, 8 pole plug.

The other cable connection is for an Ethernet cable. This cable is required to configure the sensor and to utilize the TCP/IP interface. The connection is a M12, 4 pole socket.

G.4.5 Abstraction layer

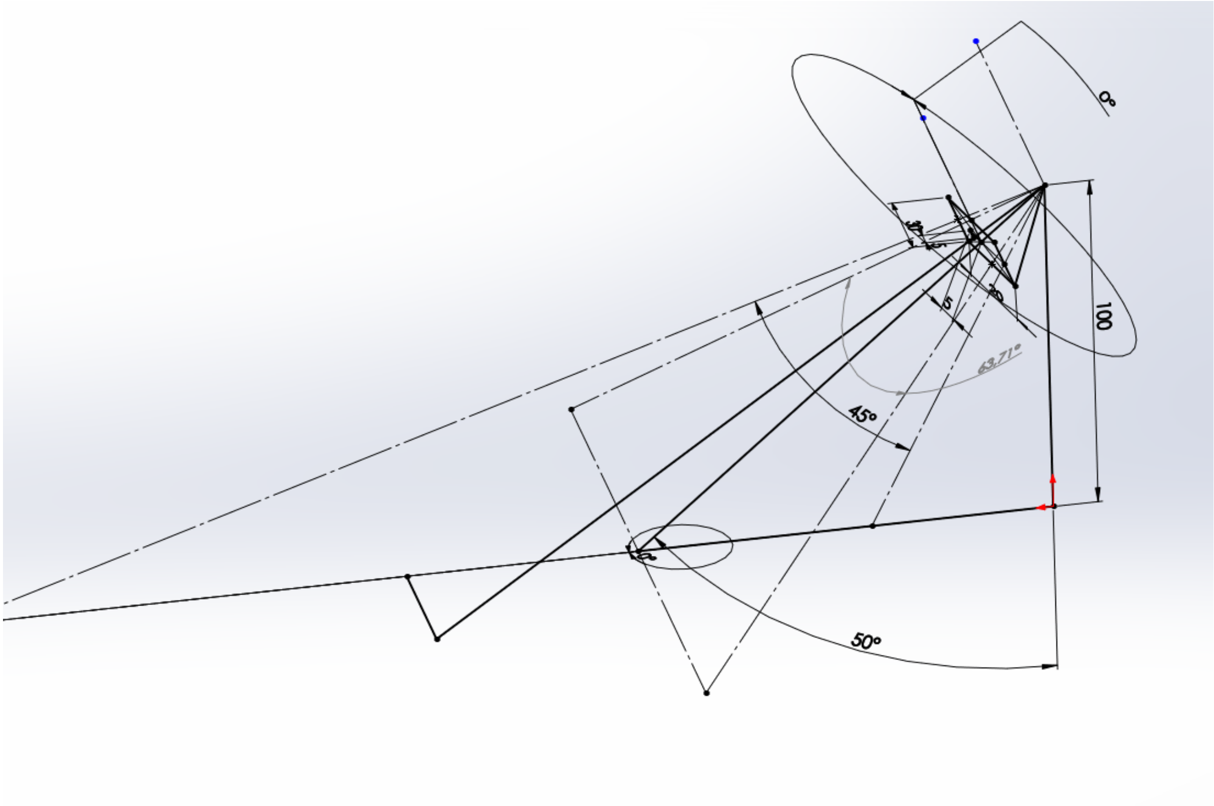
To make communication with the sensor easier and more intuitive we decided to wrap it in an abstraction layer. The language of choice became python due to its simplicity. While python is not the fastest language, it delivers simplicity within our time requirements. Python is also very readable, making it easier to extend for 3rd party developers.

G.4.6 Translation

The output from the sensor is coordinates in the camera image plane. These coordinates alone tells us little about the objects position in the real world. Given that we know the geometry of the product in question, we can make functions that does this for us. Due to the cameras internal lighting and the products reflective surfaces, we were unable to put the camera perpendicular to the surface. This meant that the image plane and the surface (in this case a plane) were not a linear mapping but a trigonometric function.

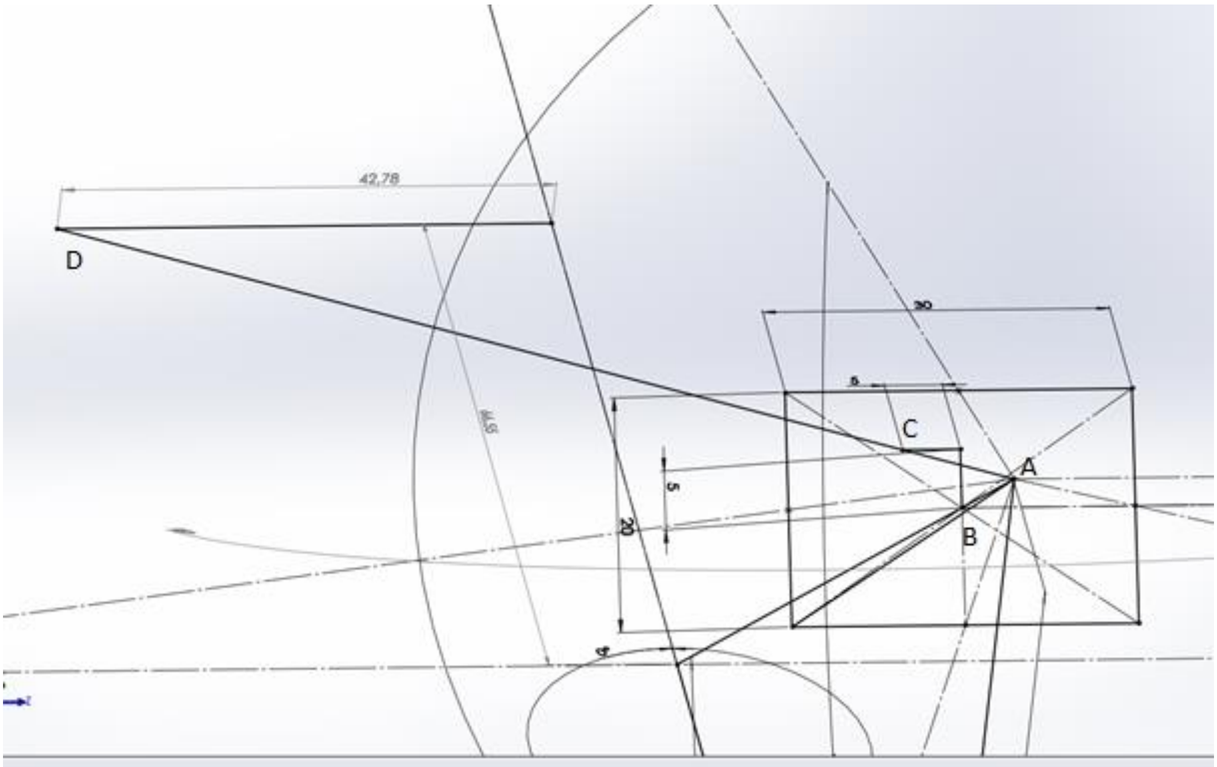
G.4.6.1 Translation functions

The problem is illustrated here:



The dotted lines represent the cameras field of view. The plane they project upon the real world surface. The height (perpendicular to the surface plane) represents the distance from the surface to the camera lens.

The image plane is further illustrated here:



Where the rectangle represents the image. A represent the focal point. B represent the center of the image. C is an arbitrary point in the image plane and D its real world projection. The length AB (center of camera to focal point) is the focal length (which is perpendicular to the image plane). The height and width of the image plane represent the image resolution (given that the imaging sensor has square "pixels").

With this information we can derive the following:

$$\varphi_x = \tan^{-1}\left(\frac{2\tan\left(\frac{\alpha_x}{2}\right)x}{c_x}\right)$$

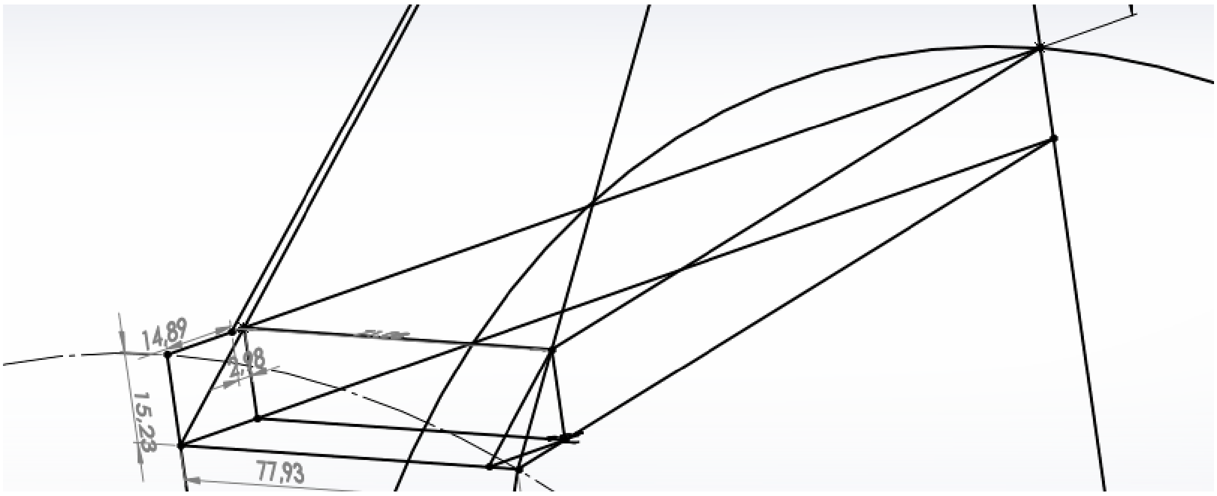
$$\varphi_y = \tan^{-1}\left(\frac{2\tan\left(\frac{\alpha_y}{2}\right)y}{c_y}\right)$$

Which gives us the angles from the focal vector to the arbitrary point in the image plane. This again gives us the following:

$$x_r = \frac{h\tan(\varphi_x)}{\cos(\theta + \varphi_y)}$$

$$y_r = h * \left(\tan(\theta + \varphi_y) - \tan(\theta)\right)$$

When projected upon a cylinder, we get a new dimension Z. All three variables also become dependent upon each other, making a set of three unknown.



Further calculations gives:

$$\rho = \left(\frac{\tan(\varphi_x)}{\cos(\theta + \varphi_y)} \right)^2$$

$$z_r = \frac{r - h\rho - \sqrt{r^2 - 2h\rho r - h^2\rho}}{\rho + 1}$$

$$y_r = \tan(\theta + \varphi_y) * (h + z) - h \tan(\theta)$$

$$x_r = \frac{(h + z) \tan(\varphi_x)}{\cos(\theta + \varphi_y)}$$

We can also derive the angle which is needed to rotate the cylinder for the point of interest's tangent to be perpendicular to the cameras y-axis:

$$\phi_r = \sin^{-1}\left(\frac{x}{r}\right)$$

G.4.6.2 Precision

While these formulas should give accurate real-world coordinates, uncertainties like the "pixel" width/height of the imaging sensor, lens warp and low image resolution will make results deviate slightly. This deviation is larger the further from the center of the camera the object is recognized. To adjust for this you pass the camera coordinates through two matrices commonly named "Camera matrix" and "Lens matrix". These matrices can be obtained by taking a series of pictures of a predefined object (a checkerboard is often used) or by prompting the manufacturer for them.

H. User manual

H.1 User Manual: UR Remote Control

Contents

Contents 302

Figures 303

H.1.1 Introduction 304

H.1.2 Universal Robot Script 304

H.1.2.1 Functions on UR Control Code 304

H.1.2.1.1 moveTo_J()... 304

H.1.2.1.2 moveTo_L() 304

H.1.2.1.3 forceMove() 305

H.1.2.1.4 seekDownForce() 305

H.1.2.1.5 setTCP() 305

H.1.2.1.6 setOutput()... 305

H.1.2.1.7 getInput() 306

H.1.2.1.8 getPose() 306

H.1.2.1.9 getWPCircle() 306

H.1.2.1.10 moveCircular() 306

H.1.2.2 Expanding Functionality.. 307

H.1.3 Class for Remote Control 307

H.1.3.1 Functions... 307

H.1.3.1.1 setMoveType(moveType) 307

H.1.3.1.2 makeWP(pos).. 307

H.1.3.1.3 sendCommand(function, parameters) 308

H.1.3.1.4 moveTo(pos, accel, speed) 308

H.1.3.1.5 moveDelta(deltaPos, accel, speed).. 308

H.1.3.1.6 forceMove(pos, accel, speed, force)... 308

H.1.3.1.7 seekDownForce(accel, speed, force) 308

H.1.3.1.8 setTCP(pos) 309

H.1.3.1.9 setOutput(portType, port, value) 309

H.1.3.1.10 getInput() 309

H.1.3.1.11 getPose() 310

H.1.3.1.12 getWPCircle(centerPoint, radius, angle).. 310

H.1.3.1.13 moveCircular(posVia, posTarget, accel, speed) 310

H.1.4 References 311

Figures

Figure 143: Datastring for getPose() 307

H.1.1 Introduction

The control code for remote control of the robot systems is composed of a program that shall run locally on the robot controller, and a class that simplifies sending the data necessary to control the program on the controller from an external computer. All current functions on the robot are described in detail in chapter H.1.2.1, with necessary data, a description of functionality, and returned data. Chapter H.1.2.2 describes how to add new functions to the program on the robot, and chapter **Error! Reference source not found.** describes how to make corresponding additions to the rogram that runs on the external computer to send necessary data. The class is written in python.

H.1.2 Universal Robot Script

The program on the robot controller is composed of a main loop that listens for strings continuously. When a string is available on the socket, the program will read it as a list of 30 floats. This is because of the limited string handling possibilities in URScript, and because most of the data needed to perform operations on the UR are floats. Items in a list can be easily accessed with syntax, very much like in regular python-programming. URScript code “value = list[i]” returns the item at position “i” from the list. The command that reads the string from a socket returns a list with 31 items, where the first represents the length of the list. The main loop runs the next item through a set of if-statements, and runs a sub-program accordingly.

H.1.2.1 Functions on UR Control Code

Most of the functions containing movements need one or more poses to execute. A pose is represented as a point and a rotation (6 floats) with a p in front to indicate that it is a pose: “p[x, y, z, rx, ry, rz]”. For remote control purposes, we only send the pose as “x, y, z, rx, ry, rz”, and let the program on the UR side wrap it into a pose. Coordinates are measured in meters, while rotation is an axis-angle representation.

H.1.2.1.1 moveTo_J()

Data string	"(0, pose, a, v, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Robot moves from current pose to new pose following the most efficient path seen in joint-space (see movej() command in URScript manual[1]) with maximum acceleration “a”(m/s^2) and velocity “v”(m/s)
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	setMoveType(“joint”), followed by a moveDelta() or moveTo() [Class for Remote Control]

H.1.2.1.2 moveTo_L()

Data string	"(1, pose, a, v, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Robot moves from current pose to new pose following a linear path in tool-space (see movel() command in URScript manual[1]) with maximum acceleration “a”(m/s^2) and velocity “v”(m/s)
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	setMoveType(“linear”), followed by a moveDelta() or moveTo() [Class for Remote Control]

H.1.2.1.3 forceMove()

Data string	"(2, pose, a, v, force, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Moves linear from current pose to new *pose* until *force* is met, with acceleration *a* and maximum velocity *v*
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	forceMove()

H.1.2.1.4 seekDownForce()

Data string	"(3, 0, 0, 0, 0, 0, 0, a, v, force, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Moves tool 0.5mm down until *force* is met. Might be removed with future updates
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	seekDownForce()

H.1.2.1.5 setTCP()

Data string	"(4, pose, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Reconfigures Tool Center Point to an offset in position x,y,z and orientation rx,ry,rz (*pose*). Move-commands will now use this new TCP to calculate joint angles to reach new waypoint/pose.
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	setTCP()

H.1.2.1.6 setOutput()

Data string	"(5, port_type, port, value, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Used to set any output on the robot controller. *port_type* == 1: runs “set_standard_digital_out(*port*, *value*)”* • *port* range 0 to 7 • *value* is 0 (False) or 1 (True) *port_type* == 2: runs “set_tool_digital_out(*port*, *value*)”* • *port* range 0, 1 • *value* is 0 (False) or 1 (True) *port_type* == 3: runs “set_standard_analog_out(*port*, *value*)”* • *port* range 0, 1 • *value* is between 0 and 1 (0 is low, 1 is high) *port_type* == 4: runs “set_configurable_digital_out(*port*, *value)”** • *port* range 0 to 7 • *value* is 0 (False) or 1 (True) *See URScript-manual for details
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”

Corresponding class function	setOutput()

H.1.2.1.7 getInput()

Data string	"(6, port_type, port, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Used to read values on input ports. *port_type* == 1: runs "get_standard_digital_in(*port*)"* • *port* range 0 to 7 *port_type* == 2: runs "get_tool_digital_in(*port*)"* • *port* range 0, 1 *port_type* == 3: runs "get_standard_analog_in(*port*)"* • *port* range 0, 1 *port_type* == 4: runs "get_configurable_digital_in(*port)"** • *port* range 0 to 7 *See URScript-manual for details
Returns	Returns value as string
Corresponding class function	getInput()

H.1.2.1.8 getPose()

Data string	"(7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Reads current pose and returns as a string.
Returns	Returns current pose as a string "p[x,y,z,rx,ry,rz]"
Corresponding class function	getPose()

H.1.2.1.9 getWPCircle()

Data string	"(8, pose_center, radius, angle, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Used to calculate a new pose on a circle with *radius* from *pose_center* rotated *angle* degrees around z-axis (think unit-circle, positive x-axis is 0 degrees). Might be useful for circular moves
Returns	Returns a pose.
Corresponding class function	getWPCircle()

H.1.2.1.10 moveCircular()

Data string	"(9, pose_via, pose_target, a, v, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
Description	Performs a circular move from current pose via *pose_via* to *pose_target*, with *a* = acceleration and *v* = velocity. Tool-orientation in pose_via (rx,ry,rz), is ignored.

	*Velocity should be kept low, since the movec() command in URScript skips the deceleration-phase. *A circular move should be followed by a moveTo(*pose_target*) to ensure that current position is correct, as the movec() command in URScript is a bit buggy.
Returns	Returns current pose as a string “p[x,y,z,rx,ry,rz]”
Corresponding class function	moveCircular()

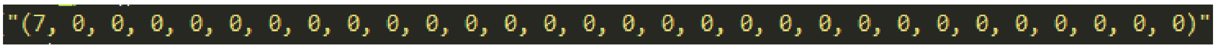
H.1.2.2 Expanding Functionality

It is easy to expand functionality on the UR-side of the system. Simply add new sub-programs through the “advanced” tab under structure, add a new “elseif” in the main loop, and assign a number to call it. Use UR manual [1] and/or the URScript manual [1] as aid. A corresponding function should be made in the program that runs on the external computer.

H.1.3 Class for Remote Control

We have made a class that makes it more efficient to program the system from the external computer. The main functionality of the class functions is to generate the data string that is sent to the controller. Each function takes a set of parameters, and puts it together in the order required by the subprogram on the robot controller. This is all written in Python code.

Each subprogram on robot side is called by its own custom string. The interface we have made uses a list of 30 floats to store all data needed to run one subprogram. The content of this list is then sent as a string formatted like in Figure 144, where the first float represents what subprogram to run. The rest of the string is used to carry the customized data necessary to run your subprogram.



```
"(7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)"
```

Figure 144: Data string for getPose()

H.1.3.1 Functions

H.1.3.1.1 setMoveType(moveType)

Parameters	*moveType* = “linear” or “joint”
Description	Reads string from *moveType* and selects movement type (“linear” for movel()*, “joint” for movej()*) *See URScript manual
Returns	None

H.1.3.1.2 makeWP(pos)

Parameters	*pos*: list of 6 floats [x,y,z,rx,ry,rz]
Description	Makes sure *pos* is correctly formatted, adds zeroes if length of list is shorter than 6

Returns	List of 6 floats (pos) [x,y,z,rx,ry,rz]

H.1.3.1.3 sendCommand(function, parameters)

Parameters	*function*: selects which function to run on UR-side *parameters*: is a list of parameters required for that function
Description	Generates a string with 30 floats based on input parameters, and transmits to socket.
Returns	None

H.1.3.1.4 moveTo(pos, accel, speed)

Parameters	*pos*: a list of 6 floats [x,y,z,rx,ry,rz] that represents the position to move to *accel*: one float representing maximum acceleration in m/s^2 (0 to 1.2) *speed*: one float representing maximum velocity in m/s (0 to 1.2)
Description	Reads *pos* and sends a move command based on current value of *moveType*
Returns	None

H.1.3.1.5 moveDelta(deltaPos, accel, speed)

Parameters	*deltaPos*: a list of 6 floats [x,y,z,rx,ry,rz] position to move to *accel*: one float representing maximum acceleration in m/s^2 (0 to 1.2) *speed*: one float representing maximum velocity in m/s (0 to 1.2)
Description	Reads *deltaPos*, calculates new pose based on current pose and *deltaPos,* and sends a move command based on current value of *moveType*
Returns	None

H.1.3.1.6 forceMove(pos, accel, speed, force)

Parameters	*pos*: a list of 6 floats [x,y,z,rx,ry,rz] that represents the position to move to *accel*: one float representing maximum acceleration in m/s^2 (0 to 1.2) *speed*: one float representing maximum velocity in m/s (0 to 1.2) *force*: one float representing the force to apply in newton
Description	Moves tool linear towards *pos* until *force* is met
Returns	None

H.1.3.1.7 seekDownForce(accel, speed, force)

Parameters	*accel*: one float representing maximum acceleration in m/s^2 (0 to 1.2) *speed*: one float representing maximum velocity in m/s (0 to 1.2) *force*: one float representing the force to apply in newton
Description	Makes robot move 0.5mm downwards along z-axis until *force* is met. May be removed in future updates
Returns	None

H.1.3.1.8 **setTCP(pos)**

Parameters	*pos*: a list of 6 floats [x,y,z,rx,ry,rz] that represents the offset and the orientation of the Tool Center Point
Description	Used to configure Tool Center Point. Changing orientation might be helpful to simplify rotations.
Returns	None

H.1.3.1.9 **setOutput(portType, port, value)**

Parameters	*portType*: one float to select port type [1 -> 4] *port*: one float to select port [0 -> 7] *value*: one float to set output signal (0 or 1 for digital, 0 to 1 for analog)
Description	Used to set outputs on controller, standard voltage levels is 0v or 24v for digital ports and 0v to 24v for analog ports. Different port types: *port_type* == 1: runs "set_standard_digital_out(*port*, *value*)"* • *port* range 0 to 7 • *value* is 0 (False) or 1 (True) *port_type* == 2: runs "set_tool_digital_out(*port*, *value*)"* • *port* range 0, 1 • *value* is 0 (False) or 1 (True) *port_type* == 3: runs "set_standard_analog_out(*port*, *value*)"* • *port* range 0, 1 • *value* is between 0 and 1 (0 is low, 1 is high) *port_type* == 4: runs "set_configurable_digital_out(*port*, *value)"** • *port* range 0 to 7 • *value* is 0 (False) or 1 (True) *See URScript-manual for details
Returns	None

H.1.3.1.10 **getInput()**

Parameters	*portType*: one float to select port type [1 -> 4] *port*: one float to select port [0 -> 7]
Description	Used to read values on input ports. *port_type* == 1: runs "get_standard_digital_in(*port*)"* • *port* range 0 to 7 *port_type* == 2: runs "get_tool_digital_in(*port*)"* • *port* range 0, 1 *port_type* == 3: runs "get_standard_analog_in(*port*)"* • *port* range 0, 1 *port_type* == 4: runs "get_configurable_digital_in(*port)"** • *port* range 0 to 7 *See URScript-manual for details
Returns	Returns value on *port* as a float

H.1.3.1.11 getPose()

Parameters	None
Description	Used to read current pose from robot
Returns	Pose as a list of floats; [x,y,z,rx,ry,rz]

H.1.3.1.12 getWPCircle(centerPoint, radius, angle)

Parameters	*centerPoint*: a list of 6 floats [x,y,z,rx,ry,rz] that represents center of circle *radius*: one float describing circle radius in meters *angle*: one float describing position on the circle rotated counterclockwise from x-axis (think unit-circle)
Description	Used to calculate positions (including tool-rotations) on a circle. Combine with moveCircular() to make circular moves
Returns	Returns calculated pose as a list of floats; [x,y,z,rx,ry,rz]

H.1.3.1.13 moveCircular(posVia, posTarget, accel, speed)

Parameters	*posVia*: : a list of 6 floats [x,y,z,rx,ry,rz] representing a via waypoint, rx,ry,rz is ignored *posTarget*: : a list of 6 floats [x,y,z,rx,ry,rz] representing target waypoint *accel*: one float representing maximum acceleration in m/s^2 (0 to 1.2) *speed*: one float representing maximum velocity in m/s (0 to 1.2*)
Description	Used to make a circular move from current position, via *posVia*, ending up in *posTarget*. *A bug in the movec() command in URScript causes the robot to skip the deceleration-phase, use a low speed to avoid damaging equipment
Returns	None

H.1.4 References

1. Universal Robots, UR User Manual, 18.05.2015: http://www.ur-update.dk/URsupport/Manuals/UserManual/Release3.1/UR10_User_Manual_en_Global.pdf
2. Universal Robots, URScript Manual 3.1, 18.05.2015: http://www.ur-update.dk/URsupport/Manuals/ScriptManual/Release3.1/scriptmanual_en.pdf

I. Table of figures

Figure 1: Backlog Example 11
Figure 2: Gantt diagram 1 of 2 12
Figure 3: Gantt diagram 2 of 2 13
Figure 4: Unified Process Diagram 15
Figure 5: Expense Budget 15
Figure 6: Use Case Diagram, Operator and assembly system interaction 19
Figure 7: Flowchart, Use Case - Assembly process: Fastening 20
Figure 8: Flowchart, Use case - Assembly process: Place and mount regulator 20
Figure 9: Flowchart, Operator and assembly system cooperation 21
Figure 10: Flowchart, Detailed Operator assembly 21
Figure 11: Flowchart, Detailed assembly system actions 22
Figure 12: Use Case, Exception Handling 22
Figure 13: F01 - Dynamic Dimension 49
Figure 14: F01.01 - Dynamic Dimension, Get assembly state 49
Figure 15: F01.02 - Dynamic Dimension, Retrieve ID 50
Figure 16: F01.03 - Dynamic Dimension, Retrieve Part 50
Figure 17: F01.03.02 – Dynamic Dimension, Retrieve Part, Locate Part 50
Figure 18: F01.05 - Dynamic Dimension, Prepare Tool 51
Figure 19: F01.07 - Dynamic Dimension, Handle Exception 51
Figure 20: Structural Dimension 51
Figure 21: Dropbox logo 122
Figure 22: OneDrive logo 122
Figure 23: Google Drive logo 122
Figure 24: appear.in logo 125
Figure 25: Microsoft Project logo 128
Figure 26: Microsoft Excel logo 128
Figure 27: SmartDraw logo 129
Figure 28: Original assembly process 149
Figure 29: Modified assembly process 151
Figure 30: Cable connector E-CC, drawing 164
Figure 31: Cable connector E-CC 165
Figure 32: Regulator E-REG, drawing 165
Figure 33: Regulator in place 165
Figure 34: Magnet jig for E-REG, drawing 166
Figure 35: Power inlet E-PI 166
Figure 36: Cable shoe 166
Figure 37: Bottom plate M-BP 167
Figure 38: Reflector M-REF 167
Figure 39: VERSABALL 169
Figure 40: 2-finger adaptive gripper from Robotiq 169
Figure 41: Parallel pneumatic gripper from Festo 170
Figure 42: Angled pneumatic gripper from Festo 171

Figure 43: Radial pneumatic gripper from Festo 171
Figure 44: Suction cups 171
Figure 45: PolyScope programming GUI 184
Figure 46: Move tab 185
Figure 47: Loading script into PolyScope program 186
Figure 48: Simple socket example 187
Figure 49: UR3 190
Figure 50: UR5 192
Figure 51: UR10 194
Figure 52: UR family 196
Figure 53: End effector 203
Figure 54: Concept for fastening screwdriver 203
Figure 55: Concept for fastening camera 204
Figure 56: Early workstation concept 205
Figure 57: Test station 206
Figure 58: Workstation alternative 1 207
Figure 59: Workstation alternative 2 208
Figure 60: Workstation alternative 3 208
Figure 61: Workstation alternative 4 209
Figure 62: Feeding rail for cable connector 210
Figure 63: Feeding rail for regulator 210
Figure 64: Regulator gripping tool 211
Figure 65: Cable connector gripping tool 211
Figure 66: Bottom plate gripping tool 212
Figure 67: Power bracket concept 213
Figure 68: The regulator 216
Figure 69: Regulator positioned to the casing 216
Figure 70: Regulator jig with a clamp 216
Figure 71: Regulator jig with magnets 216
Figure 72: Regulator jig design 217
Figure 73: 2D jig design, drawing 218
Figure 74: 3D jig design 219
Figure 75: Prototype 1 219
Figure 76: Prototype 2 219
Figure 77: Fasten Reflector-plate 224
Figure 78: Place and Fasten Cable Connector 225
Figure 79: Place and Fasten Regulator 225
Figure 80: Bottom Plate 226
Figure 81: Screenshot of PolyScope 228
Figure 82: Print with UR 229
Figure 83: Example of how to open a Socket on UR 229
Figure 84: Extracts 30 Floats from String 230
Figure 85: Extract a pose and move there 230
Figure 86: Movel_force 231
Figure 87: Set_output() 232

Figure 88: Vertical and horizontal tool orientation 233
Figure 89: Original cable connector design reference test 236
Figure 90: Full conductor temperature test 237
Figure 91: Reduced conductor temperature test 238
Figure 92: Insulating conductor temperature test 239
Figure 93: IFM sensor 243
Figure 94: Festo gripper 243
Figure 95: Sumake screwdriver 243
Figure 96: Main bracket 246
Figure 97: Camera bracket 246
Figure 98: Assembly 247
Figure 99: Prototype 2 248
Figure 100: 3D model of gripper bracket 248
Figure 101: 3D model of camera and screwdriver bracket 249
Figure 102: Camera and screwdriver bracket prototype 249
Figure 103: Cable connector E-CC 254
Figure 104: Regulator E-REG 254
Figure 105: Vacuum tool gripping point 255
Figure 106: Gripping actions 255
Figure 107: Gripping of cable connector 256
Figure 108: Gripping of regulator 257
Figure 109: Gripper force graph 258
Figure 110: Minimum gripping force formula 259
Figure 111: Gripper finger 3D design 260
Figure 112: Assembly and interference check 260
Figure 113: Fixtures table 261
Figure 114: Fixtures 261
Figure 115: Weight forces, gripper force forces, mesh 262
Figure 116: vonMises stress simulation and most affected areas 262
Figure 117: Gripper finger prototype 263
Figure 118: Cable connector testing 264
Figure 119: Cable connector fastening 264
Figure 120: Placing regulator 264
Figure 121: SL/M-REF_4.75_4.35_120d 268
Figure 122: SL/M-REF_4.95_3.8_40d 268
Figure 123: SL/M-REF_5.0_3.27_15d 268
Figure 124: SL/M-REF_5.5_3.95_35d 269
Figure 125: SL/M-REF pieces from first batch 269
Figure 126: SL/M-REF_5.5_3_20d 269
Figure 127: SL/M-REF_5.5_5_35d 270
Figure 128: SL/M-REF_5.5_6_special 270
Figure 129: SL/M-REF_6_5_40d 270
Figure 130: SL/M-REF_6_special 2 270
Figure 131: Vacuum gripper tool, 4 cups 272
Figure 132: Vacuum gripper tool, 3 cups 272

Figure 133: Suction cup setup, reflector M-REF 273
Figure 134: Suction cup setup, center-to-center 11cm, 10mm/square 273
Figure 135: Gripper connecting point 273
Figure 136: Gripper finger 273
Figure 137: Point cloud example 277
Figure 138: Kinect test picture 279
Figure 139: Normal test picture 280
Figure 140: 2-D Hole recognition with Kinect 281
Figure 141: Hole recognition with IFM 282
Figure 142: 3D object recognition 283
Figure 143: Projection upon a cylinder 284
Figure 144: Data string for getPose() 307