



Norwegian University of
Science and Technology

Methods for Augmenting Physical Prototype Activities in Early Stage Product Development

Development of a system designed to
accelerate prototyping and learning in the
early stages of product development.

Sampsa Matias Ilmari Kohtala

Master of Science in Mechanical Engineering

Submission date: June 2018

Supervisor: Martin Steinert, MTP

Norwegian University of Science and Technology
Department of Mechanical and Industrial Engineering

Sampsa Matias Ilmari Kohtala

Methods for Augmenting Physical Prototype Activities in Early Stage Product Development

Development of a system designed to accelerate prototyping and learning in the early stages of product development.

Master's Thesis in Mechanical Engineering

Trondheim, June, 2018

Supervisor: Prof. Martin Steinert

Teaching Assistants: M.Sc. Jørgen Andreas Bogen Erichsen

M.Sc. Heikki Sjöman

Norwegian University of Science and Technology

Faculty of Engineering

Department of Mechanical and Industrial Engineering

 **NTNU**
Norwegian University of
Science and Technology



This page is intentionally left blank.

Abstract

In product development and engineering design research, discoveries have revealed the benefits of using prototypes and prototyping to increase the probability of successfully making new innovative and novel products. With the increasing competition in the global markets, innovation is key for sustainability and competitiveness. In the early stage of product development, large solution spaces are explored along with ambiguous information and uncertainty. This critical stage will greatly affect the cost and quality of later development activities, such as optimization and manufacturing.

This thesis aims to augment physical prototype activities in the early stages of product development by introducing state-of-the-art technologies and alternative design approaches, to make the process simpler and faster in many scenarios. The goal is to allow the generation of more design iterations in the pre-requirement stages, ultimately increasing the probability of making better products faster.

The main methods explored are photogrammetry-based 3D scanning, converting hand drawn sketches to physical parts, documenting microcontroller-based prototypes and using object recognition frameworks. These methods have been developed and implemented into a physical system called Protobooth v2, that users can interact with to make prototypes and accelerate learning, intended for both experienced and inexperienced product developers and teams. This system builds on an ongoing research project at the TrollLabs laboratory at NTNU in Trondheim, aiming to capture prototypes to enable research on early stage product development.

Each of the main methods have been developed into functional prototypes and are presented with experiments and practical examples. They have been implemented with the physical Protobooth v2 system. Using Protobooth v2 is simple and it demonstrates a potential for augmenting physical prototype activities, by converting complex models to digital representations and documenting microcontroller output. Additionally, by utilizing computer vision and artificial neural networks, the system can automatically detect and recognize components to provide useful information about the components while building prototypes.

This page is intentionally left blank.

Sammendrag

Produktutvikling og forskning innen ingeniørvitenskap har avdekket fordelene ved å bruke prototyper og prototyping for å øke sannsynligheten for å lykkes med å lage nye og innovative produkter. Med økende konkurranse på verdensmarkedet er innovasjon en nøkkel til bærekraft og konkurranseevne. I de tidlige fasene av produktutvikling utforskes store løsningsrom sammen med tvetydig informasjon og usikkerhet. Dette kritiske stadiet vil i stor grad påvirke kostnadene og kvaliteten på senere utviklingsaktiviteter, som for eksempel optimalisering og produksjon.

Denne masteroppgaven tar sikte på å øke fysiske prototypeaktiviteter i de tidlige stadiene av produktutvikling, ved å introdusere toppmoderne teknologier og alternative designtilnæringer for å gjøre prosessen enklere og raskere i mange scenarier. Målet er å tillate generering av flere design iterasjoner i de tidlige fasene av utviklingsprosjekter, og dermed øke sannsynligheten for å produsere bedre produkter raskere.

Hovedmetodene som er utforsket er fotogrammetri-basert 3D-skanning, konvertering av håndtegnede skisser til fysiske deler, dokumentasjon av mikrokontrollerbaserte prototyper og bruk av objektgjenkjenning. Disse metodene er utviklet og implementert i et fysisk system som heter Protobooth v2, som brukere kan interagere med for å lage prototyper og akselerere læring, beregnet for både erfarne og uerfarne produktutviklere og team. Dette systemet er basert på et pågående forskningsprosjekt på TrollLabs laboratoriet ved NTNU i Trondheim, som har målet å fange prototyper for å muliggjøre forskning på de tidlige stadiene av produktutvikling.

Hver av hovedmetodene er utviklet til funksjonelle prototyper og presenteres med eksperimenter og praktiske eksempler. De er implementert med det fysiske Protobooth v2-systemet. Å bruke Protobooth v2 er enkelt og det demonstrerer et potensial for å forbedre produksjonen av fysiske prototyper, ved å konvertere komplekse modeller til digitale representasjoner og dokumentere mikrokontrollere. I tillegg kan systemet, ved hjelp av datasyn og kunstige nevralt nettverk, automatisk oppdage og gjenkjenne komponenter for å gi nyttig informasjon om komponentene mens man bygger prototyper.

This page is intentionally left blank.

Preface

This thesis for the master's degree in Mechanical Engineering was carried out between January and June 2018, at the NTNU Department of Mechanical and Industrial Engineering (MTP). The task has been performed at the research and prototyping laboratory TrollLabs, supervised by Prof. Martin Steinert, M.Sc. Jørgen Erichsen and M.Sc. Heikki Sjöman.

If you are interested in the early stages of product development and building physical prototypes, you might find this thesis interesting. Although a considerable effort has been put on utilizing computer science throughout the project, it is done from a mechanical engineer's perspective with a strong focus on practical application. This thesis contains practical examples from using the proposed methods, such as videos and 3D models, which can be examined closer by following the provided links or QR-codes to youtube.com and sketchfab.com.

Acknowledgements

I would again like to thank Martin Steinert, Jørgen Erichsen and Heikki Sjöman for having enabled this very exciting project while providing excellent support and guidance. Special thanks to Jørgen for his wisdom and this formatted text template for the thesis.

This page is intentionally left blank.

Table of contents

ABSTRACT	III
SAMMENDRAG	V
PREFACE	VII
ACKNOWLEDGEMENTS	VII
TABLE OF CONTENTS	IX
LIST OF FIGURES	XIII
1 INTRODUCTION	1
1.1 FORMAL PROBLEM DESCRIPTION.....	1
1.2 INTRODUCTION TO THE MASTER'S THESIS	1
2 BACKGROUND AND THEORY	3
2.1 PROJECT THESIS	3
2.2 PROTOTYPE THEORY	3
2.2.1 <i>Prototypes and Prototyping</i>	3
2.2.2 <i>Knowledge in Product Development using Prototypes</i>	4
2.2.3 <i>Design Methodology</i>	5
2.3 TECHNOLOGY REVIEW AND THEORY	6
2.3.1 <i>Computer Vision Library</i>	7
2.3.2 <i>3D scanning with Photogrammetry</i>	7
2.3.3 <i>Artificial Intelligence and Object Recognition</i>	10
2.3.4 <i>Image Vectorization</i>	12
2.3.5 <i>Arduino Microcontrollers</i>	13
3 DEVELOPMENT OF PROTOBOOTH V2	15
3.1 PROTOBOOTH V2 SYSTEM.....	15
3.2 SKETCH TO LASER	17
3.2.1 <i>Introduction</i>	17
3.2.2 <i>Sketch to Laser Pipeline</i>	18
3.2.3 <i>Early Concept Testing</i>	19
3.2.4 <i>Drawing Experiment</i>	20
3.2.4.1 <i>Data and Hypotheses</i>	20
3.2.4.2 <i>Experimental Setup</i>	21
3.2.4.3 <i>Experiment Results</i>	24
3.2.4.4 <i>Experiment Discussion</i>	27

3.2.5	<i>Line Resolution</i>	28
3.2.6	<i>Distortion</i>	29
3.2.7	<i>Practical Examples</i>	30
3.3	VIDEO WITH SERIAL DATA.....	32
3.3.1	<i>Introduction</i>	32
3.3.2	<i>Preliminary Concept Testing</i>	33
3.3.3	<i>Measuring Performance</i>	33
3.3.3.1	Codecs.....	34
3.3.3.2	Serial Communication Speeds using Arduino.....	36
3.3.3.3	Writing Text to Video Frames.....	37
3.3.3.4	Discussion on Performance.....	40
3.3.4	<i>Implementing Improvements</i>	40
3.3.5	<i>Results and Discussion</i>	41
3.3.6	<i>Practical Examples</i>	42
3.4	3D SCANNING.....	44
3.4.1	<i>Introduction and 3D Scanning Pipeline</i>	44
3.4.2	<i>Practical Testing and Experimenting with Photogrammetry</i>	46
3.4.2.1	Using 3D Scanning with Protoboost v2.....	46
3.4.2.2	Scene and Camera Settings.....	47
3.4.2.3	Increase Robustness.....	47
3.4.2.4	Scale.....	48
3.4.2.5	Scanning Small Objects.....	49
3.4.2.6	Limitations.....	50
3.4.3	<i>Post Processing</i>	50
3.4.3.1	Scale, Selection and Export.....	50
3.4.3.2	Smoothing.....	51
3.4.3.3	Combining Models.....	52
3.4.3.4	Convergent Modelling.....	52
3.4.4	<i>Practical Examples</i>	52
3.5	OBJECT RECOGNITION.....	54
3.5.1	<i>Introduction</i>	54
3.5.2	<i>Using YOLO with Darknet</i>	55
3.5.3	<i>Training a Neural Network with Darknet</i>	56
3.5.4	<i>Experimenting with Object Recognition</i>	58
3.5.5	<i>Detecting Materials Experiment</i>	59
3.5.5.1	Training.....	59
3.5.5.2	Results.....	59
3.5.5.3	Discussion.....	60
3.5.6	<i>Detecting Components Experiment</i>	61

3.5.6.1	Training	61
3.5.6.2	Results.....	62
3.5.6.3	Discussion	63
3.5.7	<i>Practical Examples</i>	63
3.5.8	<i>Object Recognition Implications and Solutions</i>	65
3.6	SYSTEM DETAILS	67
3.6.1	<i>Using Proto booth v2</i>	67
3.6.2	<i>Load Cells</i>	68
3.6.2.1	Robustness	68
3.6.2.2	Accuracy.....	69
3.6.2.3	Weight Measuring Applications and Limitations	71
4	PROTOBOOTH V2 PROSPECTS	73
4.1	HYPOTHESES, EXPERIMENTS AND PREDICTIONS	73
4.1.1	<i>Simplicity Hypothesis</i>	73
4.1.2	<i>Complexity and Speed Hypothesis</i>	74
4.1.3	<i>Learning Hypothesis</i>	75
4.1.4	<i>Motivation Hypothesis</i>	76
4.1.5	<i>Design Fixation Hypothesis</i>	76
4.2	POSSIBLE DRAWBACKS AND LIMITATIONS OF PROTOBOOTH V2	77
5	IMPLICATIONS AND OTHER FEATURES	79
5.1	PROTOBOOTH V2	79
5.2	OTHER POSSIBILITIES.....	80
5.2.1	<i>Motion Magnification and Color Amplification</i>	80
5.2.2	<i>Speech, Handwriting and Face Recognition</i>	80
5.2.3	<i>Sketch and Single Image to 3D Model</i>	80
5.2.4	<i>Smartphone App</i>	82
5.2.5	<i>Mechanical Strength Testing</i>	82
6	CONCLUSION	83
	BIBLIOGRAPHY	85
	APPENDED CONFERENCE PAPER	89
	APPENDIX A: PROGRAMS	
	APPENDIX B: NS-ISO 2768-1 TABLE OF TOLERANCES	
	APPENDIX C: DRAWING EXPERIMENT TASK AND RESULTS	
	APPENDIX D: PROJECT THESIS	
	APPENDIX E: RISK ASSESSMENT	

This page is intentionally left blank.

List of Figures

Figure 1: Wayfaring and probing model.....	6
Figure 2: Multi view stereo visualized, where the images are matched using known camera parameters (from SfM). Correspondence between pixels in the images are used to define a 3D shape. From Furukawa and Hernández (2015).	9
Figure 3: Methods for evaluating detections in object recognition. Illustrations from (AlexeyAB, 2018).....	11
Figure 4: Bezier curves at different interpolations and orders.....	13
Figure 5: Protobooth v2 and its main elements. Hidden underneath the table are LEDs and a NUC mini PC. The functions inside parentheses are discussed later throughout this chapter.....	16
Figure 6: Sketch with various lines, captured and converted with Protobooth v2 and laser cut onto a 3mm MDF-plate.....	20
Figure 7: Drawing experiment task: Draw the rectangle and circle shown above, in millimeters.	22
Figure 8: Tools available during the experiment.	22
Figure 9: Values used for analyzing the experiment results.	23
Figure 10: Boxplot of the participants' drawing capabilities. The 'X' represents the average and circles are outliers. Note that the angles (A) are measured in degrees and not millimeters.	24
Figure 11: Captured BMP image of a test sample and the resulting binary PNM image after using the threshold function. Note the barely visible circle on the top image.	26
Figure 12: One test sample in vector format (DXF), measured in QCAD.	26
Figure 13: Comparing an 8MP and 2MP camera for capturing and converting a sketch to vectors.	29
Figure 14: Detect and transform square example with OpenCV. The square is automatically detected and then transformed.	30

Figure 15: Light attachment example using sketch to laser with Protoboost v2. The first (top left) image shows how the light source was intended to be attached, with steps in between leading to the final solution shown at the end (bottom right).....31

Figure 16: Cable hook example. Top images: Capturing the shape of the Rexroth profile with a yellow chalk, and drawing the sketch using the shape. Bottom images: comparing the laser cut MDF part with the original sketch, followed by its practical application.....32

Figure 17: Pseudo code for the test program. “num_frames” can also be interpreted as number of iterations.34

Figure 18: Frame rates for different codecs while recording 300 frames.....35

Figure 19: Snapshots from recorded video using MJPG (left) and XVID (right) codecs, with no apparent difference in quality.36

Figure 20: Plot showing the rate of messages (lines per second) at which the Arduino Uno is capable of sending through serial communication, based on the amount of characters in each message (separated by newlines).....37

Figure 21: A frame with 34 lines with 110 characters each (110 characters/line), with the current text and frame size.....38

Figure 22: Rates for adding text to a video frame, including printing rates for an Arduino Uno.....39

Figure 23: Plot showing the change in frame rate for different amounts of characters appended to each frame, where lines represent calculated frame rates from previous tests and ‘X’ representing actual test results.....40

Figure 24: Snapshot from a recorded 30fps video, with real time serial data from an Arduino Uno.....42

Figure 25: A frame from the ultrasonic sensor video example. QR-link: <https://youtu.be/LtFkuiJcVeQ>.....42

Figure 26: Plot of the ultrasonic sensor readings versus timestamp data.43

Figure 27: Prototypes (modules) from the modular robot system. QR-link to a playlist presenting the modules, captured with the video with serial data method:

https://www.youtube.com/playlist?list=PLj9XjbRcnJ_0AMBmT0Idtu9-BngYM0qmp	43
Figure 28: Outputs from the 3D scanning pipeline using COLMAP and OpenMVS: (1) Sparse point-cloud with camera poses (COLMAP). (2) Dense point cloud. (3) Mesh reconstructed from the dense point cloud. (4) Refined mesh. (5) Refined mesh with texture. (6) Image of the real object for comparison. QR-link: https://skfb.ly/6xnGz	45
Figure 29: 3D scan setup with and without markers. The setup using markers resulted in a successful reconstruction, whereas the one without failed.	48
Figure 30: Left image is taken from the set that successfully reconstructed a 3D model, and the middle from a set that failed. The middle image is slightly more blurred (out of focus). Results from the successful scans is shown next to the real object on the right image. QR-link: https://skfb.ly/6yYCW	49
Figure 31: Typical objects that are difficult to scan with photogrammetry: LED, metal screw and photoresistor.....	50
Figure 32: Measuring and selection examples using MeshLab. The model is provided by MeshLab.	51
Figure 33: Handmade clay model to scanned and digitalized 3D model. QR-link: https://skfb.ly/6yZW8	53
Figure 34: Modifying the original 3D scanned model. From left: mesh selected for removal, scaled STL model imported in NX, smoothing and other features added to the model. In the last image the model is split in half and assembled with an Arduino Micro and a radio transceiver, for demonstrational purposes.	53
Figure 35: 3D scanned models using COLMAP and OpenMVS. QR-link: https://sketchfab.com/sampsamik/models	54
Figure 36: Object recognition test with YOLO, on Windows 10 with CUDA. Predictions and detections with bounding boxes are shown to the right and confidence values on the left.	56
Figure 37: Example using "Yolo_mark" for manually labelling training data. The content of the corresponding auto-generated text file is shown below the image, with	

object/class ID, object center (x and y value), width and height, relative to image size.	57
Figure 38: Some of the images used for training a classifier to recognize medium density fiberboard. Images are captured with the original Protobooth.	59
Figure 39: Some of the results using the classifier trained to detect MDF. The images at the top show correct detections and the bottom show wrong detections containing cardboard, a wallet and plastic box.....	60
Figure 40: Objects that are used for training a classifier: Solder sucker, Arduino Uno, HX711 breakout board, load cell and ultrasonic sensor.	61
Figure 41: Successful real-time detections of each of the five objects. QR-link: https://youtu.be/fFkLbG0M1QA	62
Figure 42: Real-time detections with inaccurate classifications, from left: motor drive module detected as an Arduino Uno, a utility knife as a solder sucker and keys as an ultrasonic sensor. QR-link: https://youtu.be/4CCUZ17RP_M	63
Figure 43: Informative windows provided when a load cell is detected in and by Protobooth v2. QR-link: https://youtu.be/T-hgF6VMqiQ	64
Figure 44: Node-red interface linking functions and executable programs.....	67
Figure 45: Protobooth v2 with arrows pointing to the load cells.....	68
Figure 46: Protobooth v2 load cell values during a period of 4.8 hours without added weight.....	69
Figure 47: Plot of commercial scale and load cell values, from lightest to heaviest object (M6, M8, M10, M12 and M16 nuts, and a small and large bolt).	70
Figure 48: The blue models are generated from the corresponding sketches. The orange shapes are the nearest shapes from the training dataset. Illustrations retrieved from Lun et al. (2017).....	81
Figure 49: Examples from Choy et al. (2016) for constructing a 3D model from a single image.....	81

1 Introduction

1.1 Formal Problem Description

Explore and develop methods for making prototypes, in the form of tangible artifacts, and methods for aiding design practitioners in early stage product development.

This includes experimenting with photogrammetry-based 3D scanning, for generating digital 3D representations of physical prototypes to discover its potential benefits in early stage product development. In addition, continue exploring other technologies and methods for making prototypes and increasing knowledge through available techniques and by producing functional prototypes.

1.2 Introduction to the Master's Thesis

The aim of this thesis is to explore and develop methods for augmenting physical prototype activities to simplify and speed up the process, ultimately allowing the creation of more design iterations in the early stage of Product Development (PD). The methods are implemented into a physical system called Protobooth v2. The physical system was built in the project thesis (Appendix D) and is further developed in this master's thesis. The system is based on a prototype capturing tool created by Sjöman, Erichsen, Welo, and Steinert (2017) aimed at enabling research, to better discover and understand causalities in early stage PD. Although both systems are similar in their physical structure and can be combined into one instrument, the focus in this thesis is to develop methods for augmenting prototyping activities and learning for developers, as opposed to capturing and documenting prototypes for research.

In the pre-requirement stages, or the fuzzy front end (Herstatt & Verworn, 2004) of engineering design, designers are often working in large solution spaces along with ambiguous information and uncertainty. The greatest potential for discovering and testing new innovative solutions lie in these early stages, as they will greatly affect the cost and quality of the later converging development activities, such as optimization and manufacturing.

Interacting with prototypes in the form of tangible artifacts is an important dimensions of tacit knowledge, where tacit knowledge is regarded as a valuable asset for knowledge acquisition according to Nonaka, Toyama, and Konno (2000). Prototypes, both external

and internal reflective prototypes as coined by Erichsen, Pedersen, Steinert, and Welo (2016), can be used as learning tools by conceptualizing and sharing ideas, often with the intention of testing functionalities or suggesting the appearance of a product concept (Lim, Stolterman, & Tenenbergh, 2008). With the benefits of interacting with prototypes through prototyping, more is often better, thus requiring a certain amount of time to be created. A solution for reducing both time and cost is the use of low fidelity prototypes (Bryan-Kinns & Hamilton, 2002).

Thus, by augmenting physical prototype activities through simplifying and speeding up the process, the goal of developing Protobooth v2 is to allow its users to generate more design iterations in the early pre-requirement stages of PD and to accelerate learning, to ultimately make better products. Both experienced and inexperienced product developers can benefit from this system. Methods are explored to discover opportunities in engineering design research by documenting performance of microcontroller-based prototypes, providing an effortless way to produce physical parts by hand drawn sketches, transforming physical prototypes into digital 3D models and using object recognition to quickly provide knowledge when developing prototypes.

The Appended Conference Paper “Augmenting Physical Prototype Activities in Early Stage Product Development” has been written as a result of the work done in this thesis. It includes a few different details, such as addressing common prototyping scenarios and compares other commercial tools to the proposed system. It serves as an overview and a supplement to this thesis.

In Chapter 2, a short background on the pre-master’s (project) thesis is provided, followed by an overview of theory covering the upcoming subjects relating to prototyping, product development and applied technologies. An overview of the physical Protobooth v2 system is provided at the beginning of Chapter 3, followed by a comprehensive description of the development of the four main methods (or modules) explored. Each of the four modules are presented with their own introduction, method, results and discussion, with some testing, experiments, and practical examples. The chapter ends with describing the complete system, including a few extra details. Chapter 4 is a discussion on the resulting system and attempts to hypothesize how it can augment physical prototype activities. Implications and other interesting possibilities are discussed in Chapter 5, followed by the conclusion in Chapter 6.

2 Background and Theory

2.1 Project Thesis

The project thesis included in Appendix D was made during the fall of 2017. It was an attempt to further explore and develop methods for capturing prototypes for enabling research on early stage PD. Its most relevant results are the physical system (Protobooth v2), which is used and developed further in this master's thesis, and the preliminary testing of photogrammetry-based 3D scanning. It also contains a general overview of prototype and prototyping theory. Although the aim is different in this thesis, the physical system has provided simple ways to test new functions to aid product developers.

2.2 Prototype Theory

2.2.1 Prototypes and Prototyping

Physical prototypes are commonly used in product development for representing ideas that potentially can become, or aid the development of, fully functional products. Products are, in this context, artifacts with a set of functions that meets the needs of end users. Ulrich and Eppinger (2012) define prototype as “an approximation of the product along one or more dimensions of interest”. This definition of a prototype embraces more diverse forms of prototypes, thereby including prototypes that are not restricted to physical objects. Additionally, they define prototyping as the process of developing such an approximation of the product. Lim et al. (2008) define prototypes as “the means by which designers organically and evolutionarily learn, discover, generate, and refine designs. They are design-thinking enablers deeply embedded and immersed in design practice and not just tools for evaluating or proving successes or failures of design outcomes”. Cognitive strategies are important for successful PD, such as insights gained through the translation of concepts through media (J. A. Edelman et al., 2009), e.g. physical prototypes.

Ullman (2009) use prototypes as tools to verify and validate an idea, and categorize them as proof-of-concept, proof-of-product, proof-of-process and proof-of-production. According to Ulrich and Eppinger (2012), prototypes have four purposes: learning, communication, integration and milestone. Prototypes can thus be used to answer questions (learning), sharing knowledge (communications), ensure compliance with other components or subsystems (integration) and to demonstrate progress (milestone). J. Edelman and Currano (2011) use resolution and abstraction to differentiate prototypes. The

resolution of a prototype is described by J. Edelman and Currano (2011) as the level of refinement or granularity that can be observed in the fit and finish of a shared representation (e.g. hand drawn sketch as low resolution and a computer aided draft as high resolution). By abstraction they mean amplification through simplification (i.e. deliberately translating something familiar into something unfamiliar), for example using wood instead of steel to prototype a car frame. Prototypes can range from low fidelity (e.g. hand drawn sketch) to high fidelity (e.g. functional and tangible 3D model) (Bryan-Kinns & Hamilton, 2002), where low fidelity prototypes are often used in the early (pre-requirement) stages of PD to save time and cost. Bryan-Kinns and Hamilton (2002) also name a few other dimensions that can affect the form and content of prototypes, such as the ability of designers to use these tools, time available to develop the prototype and cost.

Ulrich and Eppinger (2012) argue that prototypes can reduce the risk of costly iterations, by testing and uncovering unanticipated phenomena, or unknown unknowns (Sutcliffe & Sawyer, 2013). In the design, test and build process described by Leifer and Steinert (2011), creating prototypes as fast as possible is crucial for testing particular ideas, where the acceleration of learning is key. These prototypes tend therefore to be of low resolution and tangible, often resulting in utilizing simple materials such as cardboard, clay and wood etc. Discovering and testing innovative solutions in the early stage or fuzzy front end (Herstatt & Verworn, 2004) of new product development is crucial, as they will greatly affect the cost and quality of the later converging development activities, such as optimization and manufacturing. The level of radicalness of the final solution is often decided during the early stage of PD (Leifer & Steinert, 2011).

2.2.2 Knowledge in Product Development using Prototypes

One of the key elements in early stage PD is the generation and utilization of knowledge (Nonaka & Takeuchi, 1995; Ringen & Welo, 2015; Sutcliffe & Sawyer, 2013). Learning mechanisms in PD can be categorized into three loops (Leifer & Steinert, 2011). Learning loop one is based on explicit knowledge (formal, fact-based information) and aims to retain knowledge from the development projects. Learning loop two involves the informal space between PD team members and their coach (facilitator). Tacit knowledge, learning loop three, is the skill and learnings of the individuals (i.e. designers). Interacting with conceptual knowledge assets (i.e. prototypes), that can be in the form of tangible artifacts, is one of the most valuable dimensions of tacit knowledge as a means of knowledge

acquisition (Chou & He, 2004; Nonaka et al., 2000). Prototypes, both external and internal reflective prototypes (Erichsen et al., 2016), can be used as learning tools by conceptualizing and sharing ideas, often with the intention of testing functionalities or suggesting the appearance of a product concept (Lim et al., 2008).

One of the well-known factors limiting the output of conceptual design (and generation of new knowledge) is design fixation (Jansson & Smith, 1991). Design fixation can occur when exposed to examples of solutions, even if flaws in the examples are present. A study done by Viswanathan, Atilola, Esposito, and Linsey (2014) showed that building physical prototypes can help designers identify flaws and reduce the effect of design fixation. However, using physical models can cause heavily invested PD teams to fixate on their initial ideas. To reduce the effect of fixating with physical models, Viswanathan and Linsey (2013) suggest minimizing the sunk cost (money, time or effort) associated with building prototypes.

2.2.3 Design Methodology

In the pre-requirement stage, or fuzzy front end, of radical and novel product development, the pre-planned stage-gate model is rarely suitable. Although an efficient (governance) model for managing risk in PD projects, it leaves little room for exploring new and large solution spaces. These types of challenges are better solved through wayfaring and probing.

In the Hunter-Gatherer Model based on wayfaring (Steinert & Leifer, 2012), the hunters (designers) use wayfaring skills to find their way in a vast solution space, through exploration, prototyping and abduction, while the gatherers validate and optimize theories and frameworks. It is by no means a fixed model with steps to follow, but rather a personal and context dependent pathway alternative (Steinert & Leifer, 2012). It is based on small, agile and multidisciplinary teams, that together find their way in uncertainty, to discover unknown unknowns, and set the course towards the next big idea. In the last step, when the hunt is over, optimization, manufacturing and other engineering routines are followed through. The wayfaring approach is illustrated in Figure 1 below, from Gerstenberg, Sjöman, Reime, Abrahamsson, and Steinert (2015).

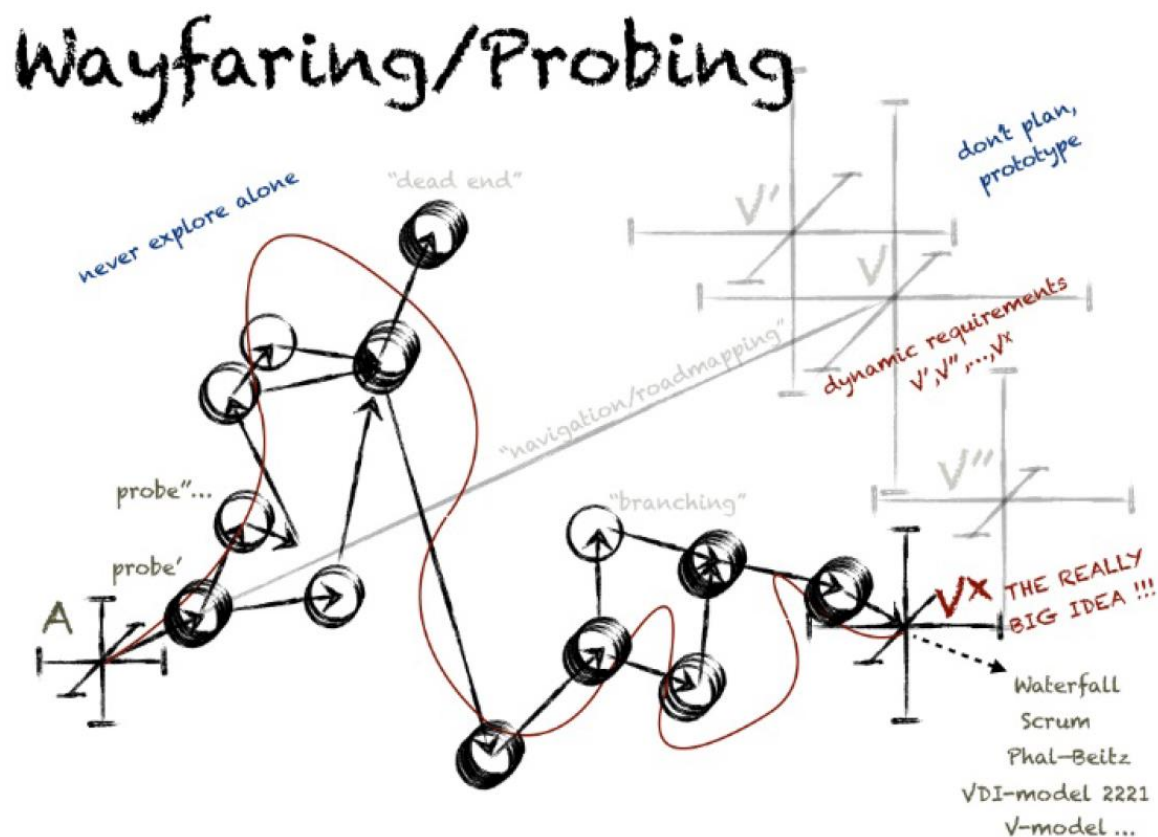


Figure 1: Wayfaring and probing model.

The wayfaring journey consists of probes: circles of designing, building and testing ideas and prototypes (Gerstenberg et al., 2015). Each probe consists of divergent and convergent thinking, generating questions and answers, producing low resolution prototypes, and subsequently altering the wayfaring path dynamically towards better ideas, based on abductive reasoning. It is a dynamic journey where requirements emerge and change over time.

One important enabler for creative thinking and accelerating physical prototyping is makerspaces, that are flexible and promote absence of fixed processes (Leifer & Steinert, 2011). Makerspaces with rapid prototyping tools and machines allows change to happen, by simply providing the necessary equipment for developers to explore and realize concepts through probing.

2.3 Technology Review and Theory

This section presents relevant theory for this thesis. Subjects presented here serves as a fundament for better understanding how different problems throughout the thesis are solved, and which methods are used. The degree of details presented here is in line with

what is required to understand the basics of these methods. Due to the increase of open source software with good documentation, many of these methods are simple to test and apply for various applications.

2.3.1 Computer Vision Library

Computer vision, a field often considered a part of artificial intelligence and computer science, is the transformation of data from still images or video into either a decision or a new representation (Bradski & Kaehler, 2008). Open Source Computer Vision Library (OpenCV) is a free computer vision and machine learning software library ("OpenCV about," 2017). It contains more than 2500, both classic and state-of-the-art, optimized algorithms. OpenCV can be used for many different applications, including image and video capturing and editing, image transformation and conversion, and has modules for utilizing object detection and classification frameworks. OpenCV has interfaces for C++, C, Python, Java and Matlab, and supports Windows, Linux, Android and Mac OS. C++ is a popular general-purpose, object-oriented and compiled programming language, with a large community of support for using OpenCV.

The OpenCV library is particularly suitable for simple web camera applications, by using e.g. C++ to make programs for controlling camera input and output. Logitech C920 and C930e webcams are commonly used with OpenCV with their simple plug and play interface, while producing high quality (full-HD, 1080p) images and video (30fps).

An important contributor for the increase in applications and robustness of computer vision, is the development of General-Purpose Computing on Graphics Processing Units (GPGPU), or GPU-accelerated computing. GPU-accelerated computing can utilize thousands of GPU cores in parallel, whereas a Central Processing Unit (CPU) runs sequentially, thus increasing the speed substantially. CUDA is a widely used parallel computing platform and programming model for GPU-accelerated computing, developed by NVIDIA ("CUDA Zone," 2018). It is however limited to work on a selection of NVIDIA GPUs.

2.3.2 3D scanning with Photogrammetry

Shape and appearance related data from real objects can be collected and analyzed with 3D scanning technologies (e.g. photogrammetry), and then processed through surface reconstruction algorithms to create a digitalized 3D model of the object. 3D scanning has

become more common in the industry, used for reverse engineering and quality control. In PD it is often used to accurately model complex geometries, such as faces, hands, bones or limbs (for prosthetics), to make custom products. However, it is generally not associated with the pre-requirement stages of PD, therefore often avoided or not considered until later development stages.

Photogrammetry is defined by Schenk (2005) as “the science of obtaining reliable information about the properties of surfaces and objects without physical contact with the objects, and of measuring and interpreting this information.”. It is studied in the field of computer vision. Photogrammetry has become an increasingly robust 3D scanning method in the past few years, due to the availability and reduced cost of high resolution digital cameras and processing power.

Structure from Motion (SfM) is an important process in photogrammetry, where images of a scene (or object) from different viewpoints are projected to form a 3D structure. Incremental SfM (Schonberger & Frahm, 2016) is a process that first finds overlapping scenes in the input images before estimating image poses and reconstructing a scene structure made of points (i.e. point cloud). Overlapping images are found through feature extraction, commonly by using the Scale Invariant Feature Transform method by Lowe (2004), then by matching features to find image pairs, and lastly verifying the overlapping image pairs by estimating transformations that maps feature points. This results in a scene graph, where images are nodes and verified pairs of images are edges. The scene graph is used further to first select an initial image pair, then register images and triangulate points in the scene. Bundle adjustment is used to refine camera and point parameters to minimize the (reprojection) error.

Multi View Stereo (MVS) consist of techniques that use stereo correspondence (Furukawa & Hernández, 2015), which is the problem of finding parts of an image corresponding to parts of another image. MVS is dependent on a set of images and their parameters (computed from SfM) such as camera poses (relative locations and orientations of 2D images), focal length and number of pixels. MVS calculates depth and normals of the scene and generates a dense point cloud. The MVS method is illustrated in Figure 2.

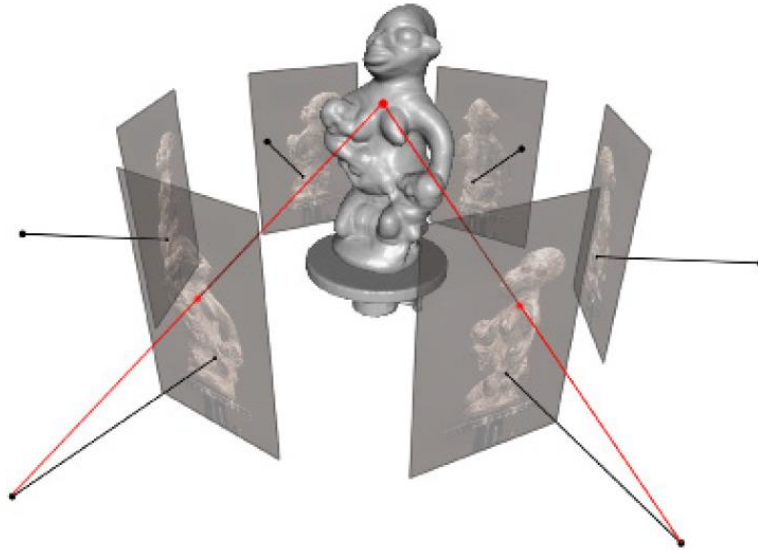


Figure 2: Multi view stereo visualized, where the images are matched using known camera parameters (from SfM). Correspondence between pixels in the images are used to define a 3D shape. From Furukawa and Hernández (2015).

With a dense point cloud, surface reconstruction algorithms such as Screened Poisson Surface Reconstruction (Kazhdan & Hoppe, 2013), and modifications using interface classifiers for preserving weakly supported surfaces (Jancosek & Pajdla, 2014), are used to generate a 3D mesh (vertices, edges and facets). A common last step of photogrammetry is to apply texture to the reconstructed model. This is typically performed in two steps: select view(s) to texture each face, then optimize the texture for consistency to avoid seams (Waechter, Moehrle, & Goesele, 2014).

Even though SfM and MVS can produce highly accurate 3D models, it can sometimes be necessary to manually edit the end results. MeshLab (Cignoni et al., 2008) is an open source mesh processing tool with a Graphical User Interface (GUI). It can input point clouds and meshes in many different formats (e.g. PLY, STL, OBJ and VRML), and export to several formats. It comes with many useful functions, including mesh cleaning filters, remeshing filters, colorization, measuring and slicing to name a few. It also has methods for surface reconstruction (for example using the Screened Poisson algorithm), smoothing and gluing different meshes together.

Furukawa and Hernández (2015) mention three main limitations of photogrammetry: lack of texture, thin structures and non-Lambertian surfaces (e.g. metals and glass). Some of these limitations can be reduced by adding something to the model to reduce reflectance.

2.3.3 Artificial Intelligence and Object Recognition

Artificial Intelligence (AI) is the study of intelligent agents that receive percepts from the environment and perform actions (Russell & Norvig, 2016). Numerous subfields use AI, in applications ranging from training computers to play chess, drive cars and to identify objects in images.

Given examples of inputs and corresponding outputs (training set), these agents can learn (supervised learning). When the possible output is a finite set of values (e.g. sunny, cloudy or rainy), the learning problem is called classification, as opposed to continuous numbers in a regression problem. While the training set is used to fit parameters of the classifier (intelligent agents), a validation set is often used for tuning the parameters (Ripley, 2007), by validating the classifier on other data than it is trained with.

Overfitting is a problem that can occur for all types of learners, caused by learning attributes from a set of examples that cannot be used on other data. For example, if an agent is trained to detect if a dice has landed on the number six, but given examples of the weight, time, color and light of dices, it might learn to detect the number six whenever it detects a certain weight at a certain time. This can produce classifiers that works very well on the training set, giving the impression of being a good classifier, but fail with other data. Using a validation set can reduce this effect.

The error rate can indicate how good or bad an agent is at predicting the correct output, but a better measure is loss (often used interchangeably). Loss is calculated using a loss function, which subtracts the amount of utility of the prediction from the utility of the correct answer, so the goal is to minimize loss. Utility is a measure of the preferred outcome. Using a system for finding and removing trash as an example: classifying a piece of trash as a banknote and a banknote as a piece of trash both have the same error, but in this case, it is better to classify trash as a banknote, to keep a piece of trash (low loss) rather than removing a valuable banknote (high loss).

Artificial neural networks have been studied and applied to encompass these functionalities, inspired by how the human brain works through networks of brain cells (neurons). These artificial networks consist of nodes (input and output nodes) connected by directed links. Links propagate actions between nodes and has a numeric weight for determining the strength and sign of the connection. These nodes are categorized into layers (e.g. input nodes as one input layer). When a network has nodes in between the input and

output layers, the network has hidden layers. With more than one hidden layer, we get a deep neural network.

Convolutional Neural Network (CNN), one of biologically inspired models, is commonly used for visual (pattern) classification problems (Matsugu, Mori, Mitari, & Kaneda, 2003). A method for classifying objects in images proposed by Redmon, Divvala, Girshick, and Farhadi (2016) consists of three main steps: first resize the image, then run a single CNN on the image to predict object locations (bounding boxes) and class (object category or name) probabilities for those locations, and lastly threshold the detections by a confidence. Their network architecture consist of 24 convolutional layers, inspired by a deep CNN called GoogLeNet from Szegedy et al. (2015), followed by 2 fully connected layers. The convolutional layers extract features from the images, while the connected layers predict output probabilities and coordinates. Thus, they can both detect objects in images and classify them, hence recognizing objects (object recognition).

Different methods are used for evaluating the accuracy of object recognition models. Precision is measured by the percentage of the detection that is correct. Recall is the percentage of the real object that is correctly detected. Intersect of Union (IoU) measures the overlap of the object area and detected area, divided by their combined area (union). These methods are illustrated in Figure 3 below.

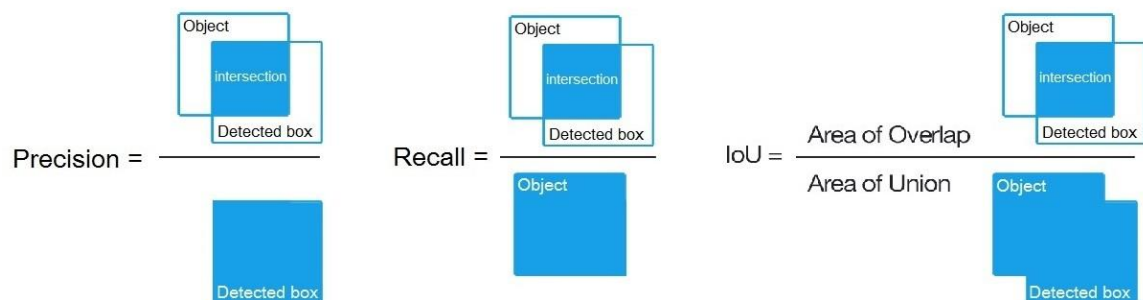


Figure 3: Methods for evaluating detections in object recognition. Illustrations from (AlexeyAB, 2018).

Another common method for evaluating object recognition models is (mean) Average Precision (AP or mAP), used in the Pascal Visual Object Classes (VOC) challenge (Everingham, Van Gool, Williams, Winn, & Zisserman, 2011). mAP is a combination several parameters, including some of the aforementioned methods.

2.3.4 Image Vectorization

Graphic images comes in two basic forms: raster (or bitmap) and vector images (Bah, 2009). Raster images are made of individual pixels with individual colors. Vector images are made of straight and curved lines, defined by their start and end points, and curve parameters such as radius etc. The main difference is clear when the images are enlarged, where a raster image would become jagged while a vector image will remain smooth. Image vectorization is the process of tracing and converting raster images to vector images. Vector images can have many formats, for example Scalable Vector Graphics (SVG) and Drawing Interchange Format (DXF). Due to their nature, vector files are suited for applications within graphical design and Computer Aided Design (CAD), and often used for making cutting patterns for automatic laser cutters and mechanical mills (e.g. printed circuit board mills).

A common method for making smooth lines after image vectorization, is using Bezier curves. Bezier curves can be described both by polynomial functions and simple interpolations. The top illustrations in Figure 4 below from Kamermans (2017) show how different interpolations at a specific interpolation ratio (steps) affect the curve. The ratio, or step, is simply the distance from the first point to the in-between (interpolated) point divided by the in-between point and the next point. More interpolations lead to more points and thus smoother curves. At the bottom of the illustration is shown different degrees of interpolation. Points also have a weight for dynamically changing the interpolations, e.g. to reduce the effect of a line that stretches considerably further away from the other. These are the basics of a large variety of Bezier algorithms.

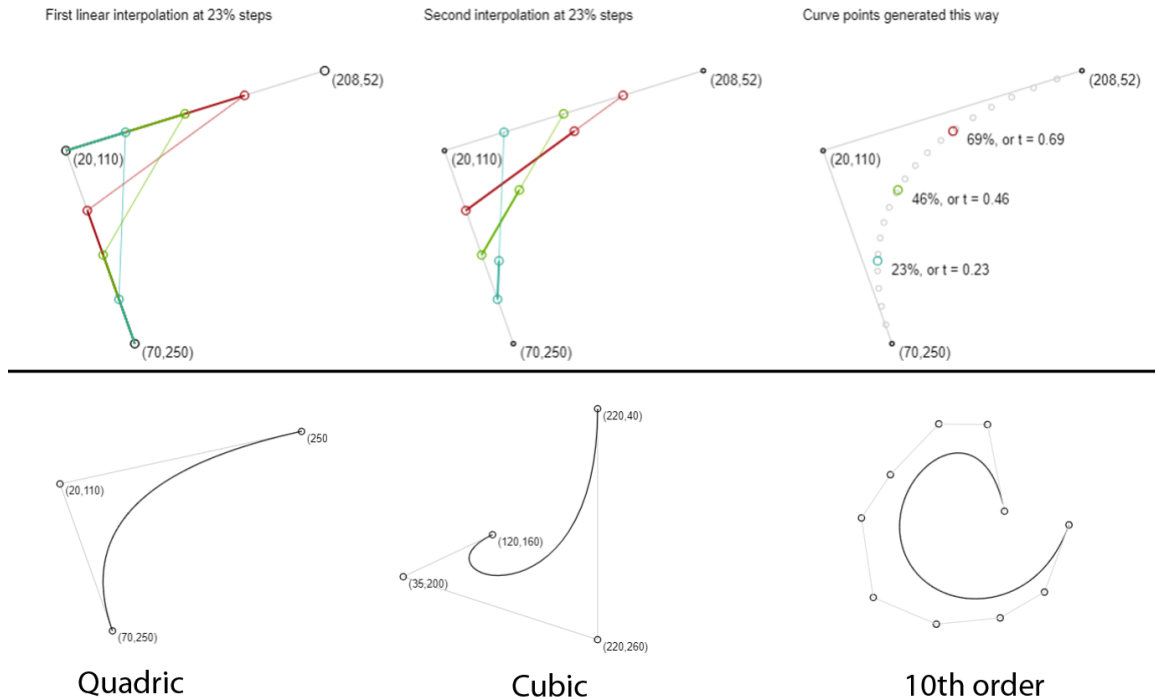


Figure 4: Bezier curves at different interpolations and orders.

2.3.5 Arduino Microcontrollers

Ready-made microcontrollers, such as the Arduino with an ATmega328 microcontroller, provides a simple and fast method for interacting with the physical environment by controlling actuators and using sensors. They are widely used in early stages of PD to quickly make functional prototypes. It is a great way to facilitate ideas and test them in the real world, and to demonstrate how the prototype (potential product) works.

Arduinos have a built in USB serial interface, for exchanging serial data (one byte at a time) between a computer. Usually, data from the microcontroller is displayed through software, such as the Arduino integrated development environment and Processing. Serial communication is normally used for visualizing how the microcontroller works, for finding bugs, logging sensor data and for sending messages between different modules in a system. Serial communication can also be used for sending messages from a computer to an Arduino.

A load cell is an example that can be read by an Arduino, by the Arduinos' digital and analog input and output (I/O) interfaces, usually through an amplifier circuit (e.g. HX711). A load cell is a piece of metal (often aluminum) with strain gauges attached. When a load is applied to the metal, it will deform along with the strain gauge, and the electrical

2 Background and Theory

resistance in the strain gauges change. The change in resistance (signal) is then amplified and measured. In the linear elastic region of the metal, load can be calculated based on the change in resistance due to linear elastic deformation. Load cells are highly accurate, often down to one tenth of a gram, and are consequently sensitive to vibrations, stress and temperature.

3 Development of Protoboost v2

In this chapter, the physical Protoboost v2 setup is described, followed by a thorough description of the four separate (main) modules developed in this thesis, including data from testing, some experiments, practical examples, and discussions on potential benefits and limitations of the modules relating to early stage PD. Both quantitative research and empirical studies are conducted, where the first two modules (sketch to laser and video with serial data) include both and the other two (3D scanning and object recognition) are mostly based on empirical analysis. This chapter ends with a general overview of the current system and usage, including a few extra details such as weight measuring applications and limitations.

3.1 Protoboost v2 System

A few changes and improvements have been made to Protoboost v2 since the pre-master's thesis (Section 2.1 and Appendix D). An Intel NUC mini PC with Linux (Debian 9) has been added to develop applications, control hardware and maintain functions, mostly by using C++. It is running on a Core i3-7100 2.4GHz CPU with Intel HD Graphics 620 GPU (not supported by CUDA). The NUC is connected with an Arduino Mega 2560 for controlling the turntable, LEDs and reading the load cells. The main elements of the physical system are illustrated in Figure 5 below.

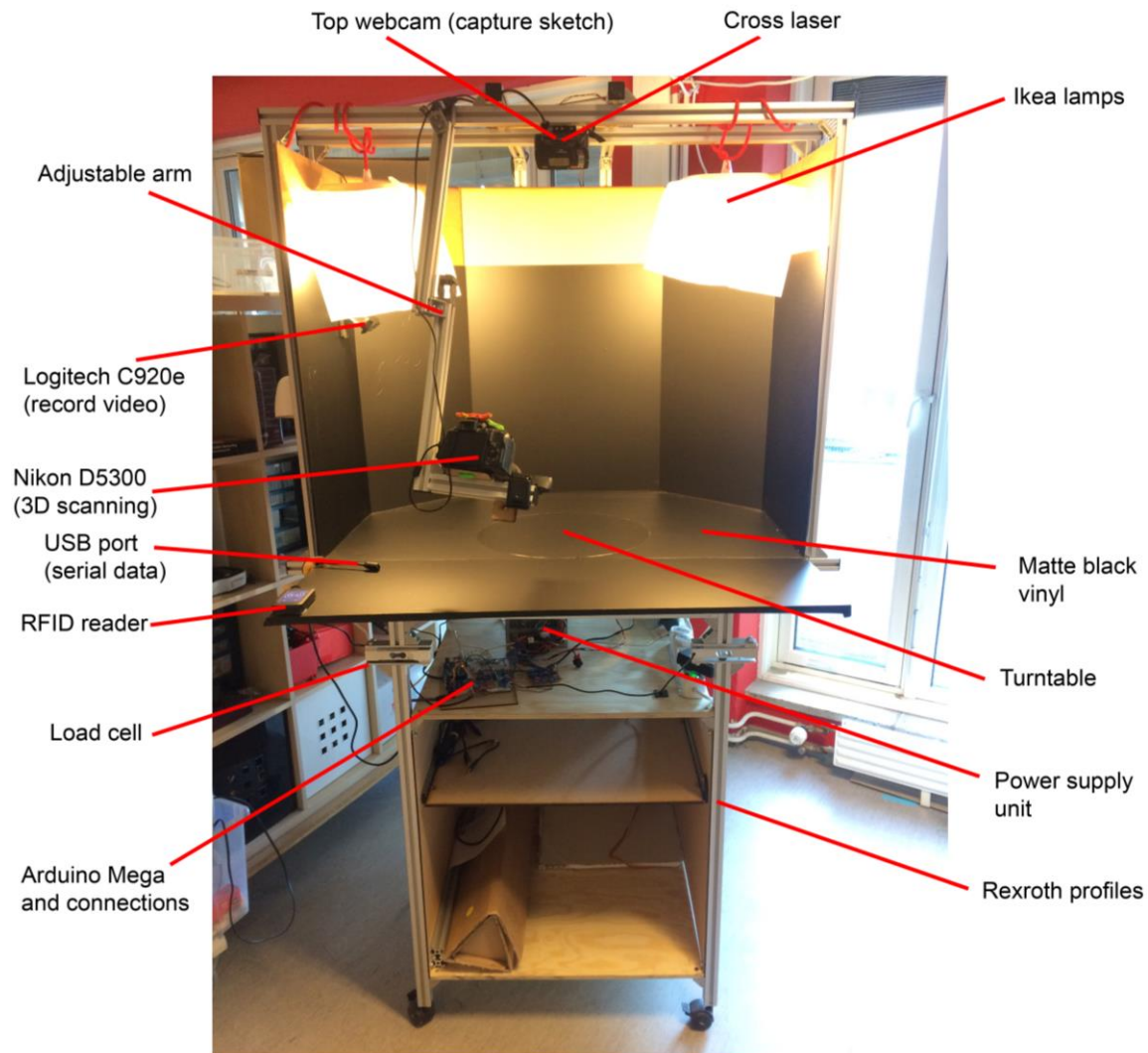


Figure 5: Protoboost v2 and its main elements. Hidden underneath the table are LEDs and a NUC mini PC. The functions inside parentheses are discussed later throughout this chapter.

LEDs have been added underneath the table, to indicate the status of Protoboost v2 when it is used (e.g. showing different colors while capturing images and after completion).

Each of the four load cells are connected to the Arduino through HX711 amplifiers, enabling each load cell to be read individually.

Logitech C920e and C930 are used for capturing images or video frames, and a Digital Single-Lens Reflex (DSLR) camera (Nikon 5300D) has been included for 3D scanning to capture higher resolution images and to enable control over other parameters such as exposure and focal length. The Nikon camera is controlled using gPhoto2, which is a command-line frontend to libgphoto2 (a C programming library designed to access and control digital cameras).

Different programs have been made (Appendix A-1) to easily control the turntable, LEDs and load cells, using C++ and serial communication. C++ programs are compiled and executed through the Linux terminal, as command line arguments. The turntable for example can be set to a specific rotation speed and be activated with the program through the terminal. Node red, a Node.js framework with its browser-based flow editor, connects programs. An RFID reader is connected to the NUC and read through Node-red for activating and interacting with Protoboost v2 and its software implementations.

The Appended Conference Paper describes the system and some of its modules (functions), including details on common PD scenarios and arguments for why and how Protoboost v2 can augment physical PD activities. In the following sections, more focus is put on the 4 main modules and their technical solutions and practical limitations, including testing, some experiments, and practical examples.

3.2 Sketch to Laser

3.2.1 Introduction

The idea of leveraging hand drawn sketches to produce physical parts (low- to high-fidelity prototype) was introduced in the pre-master's thesis. This method is not a common practice in PD activities, likely due to existing software for converting images to vector images being more directed towards other applications and fields (e.g. visual arts). It is believed that the proposed method can help product developers make physical parts faster and with less effort in many scenarios, especially during early stage PD projects. This method has consequently been further explored, implemented and tested with Protoboost v2, used at the TrollLabs laboratory at NTNU with a Gravograph LS1000 XP laser cutter.

To get an overview of possibilities in converting an image of a hand drawn sketch to a vectorized approximation of the sketch, existing solutions were explored. The following programs were found and tested:

- Potrace: Finds outlines.
- Linetracer: Lines enclosed.
- Autotrace: Finds outlines.
- WinTopo: Windows only, no automation possibility.
- Inkscape (also includes potrace): No automation possibility.
- F-engrave: No automation possibility.

- KVEC: Command line application (automation possible), works with both Windows and Linux, finds centerlines.

Each program is made for tracing raster images and transforming them to vector files that are smooth and scalable (without losing resolution). From the list of programs the only working solution that can both trace centerlines and be automated (i.e. through command line arguments) is KVEC (Kuhl, 2010).

KVEC is a freeware program that can convert raster graphics to vector graphics. It is also able to trace lines and generate vectors at the midpoint of the lines. It supports many input files such as BMP, Tiff, PNM and GIF, and output formats such as DXF and SVG, where DXF is commonly used for laser cutting. Generated vectors can be scaled and smoothed using the Bezier algorithm. The Bezier algorithm is not supported for the DXF format but will instead be simulated and approximated by polylines. The Bezier algorithm parameters can be modified, to allow more or less deviation from the original sketch in the generated lines.

A proposed system for applying this method is introduced in the next subsection, followed by early test results that ultimately generated new questions to be asked. An experiment, that seeks to understand how developers (i.e. designers) drawn, is conducted to get quantifiable data that can help discover suitable applications and technical limitations of the proposed system. Example use cases will be provided at the end.

3.2.2 Sketch to Laser Pipeline

To enable Protoboost v2 to capture sketches, the C920e web camera is used. The camera is attached at the top of Protoboost v2 (see Figure 5 in the previous section) aiming down at the center of the table, approximately 0.80 meters apart. Camera parameters were adjusted, such as digital zoom, focus and exposure, to make the sketch as clear as possible. 1920x1080 pixel BMP raster images are captured and further processed using OpenCV. In the program (Appendix A-6), written in C++, the colored image is first converted to a grayscale image, then to a binary black and white image (PNM and GIF format). A threshold function is used to make the drawn lines black (value 0) and the white paper white (value 1), with a predefined threshold value acquired from testing. KVEC is then used to trace the lines and generate vectors, which are smoothed by using a simulated Bezier algorithm, and scaled one-to-one by a scale factor calculated through testing. In addition, as desired by a mechanical engineering student at the lab, the image is also converted to a

scaled PDF file. Converting and scaling the image to PDF is done using ImageMagick (ImageMagick Studio, 2008), which is an open source software for converting and editing raster and vector image files. Finally, the DXF and PDF file are sent to a google drive folder available at the computer connected to the laser cutter in TrollLabs.

At the computer connected to the laser cutter, it is possible to drag and drop the DXF and PDF file into the laser cutting software Gravostyle. The DXF can be cut directly by the laser cutter, but normally it is necessary to ungroup separated lines and apply a cutting sequence. The PDF can be used as a background layer to manually trace lines and draw vectors using the software GUI, and then cut the parts with the modified lines. It was also discovered that the DXF file can be imported to Autodesk Fusion 360 as a sketch, which can be further processed to make CAD-models.

3.2.3 Early Concept Testing

After having fine-tuned the threshold and camera parameters, testing showed some issues that should be addressed. The main problem was the varying quality of generated vectors based on drawn lines and their color and size. E.g. using a thin pencil might result in weak lines that are only partially detected after using the threshold function. To compensate, a lower threshold could be used, but would create more noise in the image. Noise in the binary image will produce many small vectors when using KVEC. Thick and dark lines work well but have lower accuracy, since the vectors are placed in the middle of the lines, thus difficult for the designer to draw an accurate sketch for laser cutting. It was also discovered that the noise issue was different based on time of day, resulting most likely from the change in light in the room throughout a day. A simple test sample is shown in Figure 6 below, where lines are drawn using different markers, ballpoint pens and pencils. The weakest lines are the result from using a pencil. It is clear how the laser cut MDF is incomplete for the weakest drawn lines.

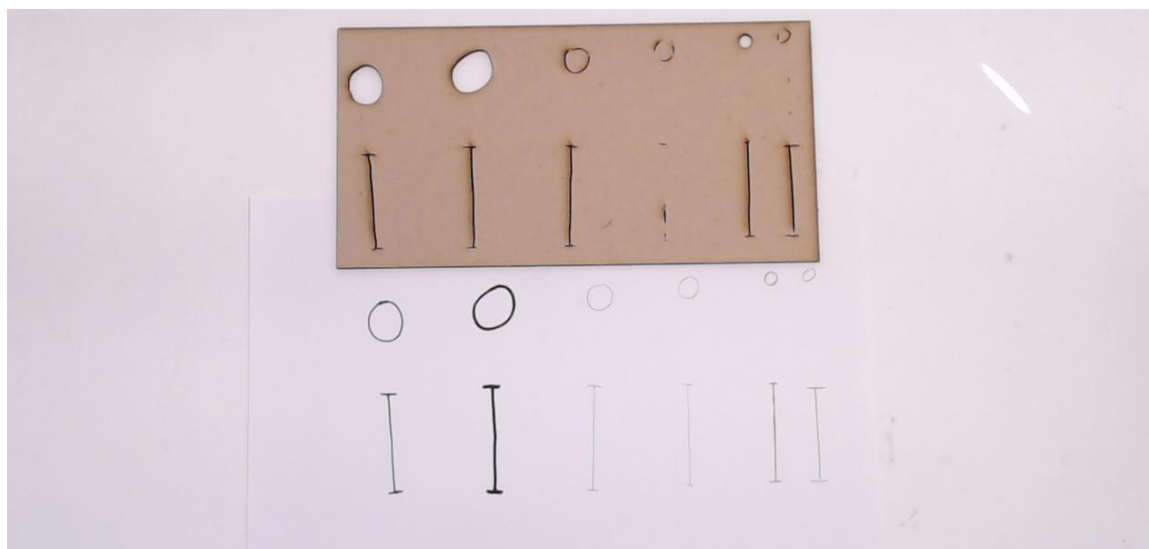


Figure 6: Sketch with various lines, captured and converted with Protoboost v2 and laser cut onto a 3mm MDF-plate.

The last but most important consideration for this method, is the drawing capabilities of users, i.e. designers and product developers. With computer software, drawings can be perfectly simulated and generated using actual numbers as input, whereas drawing by hand might be subject to personal preferences, such as drawing tools, and their analog nature. It can be valuable to discover how accurately people can draw, compared with time, and what kind of tools they prefer when drawing. Then it is possible to tailor the application for their drawing preferences and discover the accuracy and limitations of this method, compared to using the conventional methods such as CAD or a vector graphics editor (e.g. Inkscape). An experiment was therefore conducted, described in the following subsections.

3.2.4 Drawing Experiment

To find the most suitable applications for converting hand drawings to laser cuttable files, a pilot experiment was conducted to discover the drawing abilities of designers (accuracy and speed), along with discovering the technical limitations of the proposed system. The study is considered a pilot experiment due to the relatively few number of participants, and not having a fixed and controlled environment during experimentation.

3.2.4.1 Data and Hypotheses

To evaluate accuracy, deviation in millimeters can be measured between the captured hand drawings and the actual dimensions intended for the drawing. For a measure of quality, the deviations can be compared to standardized engineering tolerances for mechanical components. E.g. an accuracy of $\pm 0.3\text{mm}$ would be considered a permitted medium

deviation for basic sizes between 30 - 120mm according to the NS-ISO 2768-1 standard (see the table in Appendix B) for general tolerances (Hartvigsen, Lorentsen, Michelsen, & Seljevoll, 2006), whereas an accuracy of $\pm 5\text{mm}$ is relatively high and could be accepted for very rough prototypes. In this context, making sketches by using computer software is considered 100% accurate.

The following null hypothesis was formed:

Humans can produce hand drawn sketches for laser cutting that are equally accurate compared to sketches made using computer aided design tools.

The alternative hypothesis is stated as follows:

Humans cannot produce hand drawn sketches for laser cutting that are equally accurate compared to sketches made using computer aided design tools.

To include both accuracy and time, the following null hypothesis was formed:

Drawing speed does not influence accuracy.

With the alternative hypothesis:

Drawing speed influences accuracy.

Additionally, by observing which tools were chosen and used by designers during the experiment and assessing the quality of drawn lines by capturing them through Protoboost v2, it is possible to tailor the system for the most common usage scenarios. E.g. if most people prefer pencils, Protoboost v2 should be able to capture and convert pencil drawings to laser cuttable files.

3.2.4.2 Experimental Setup

To measure drawing accuracy, participants were asked to draw a rectangle and circle with given dimensions. The dependent variable is the deviation between the drawn lines and the given dimensions. They were provided the illustration shown in Figure 7 below, which was scaled down so it could not be used directly as a reference.

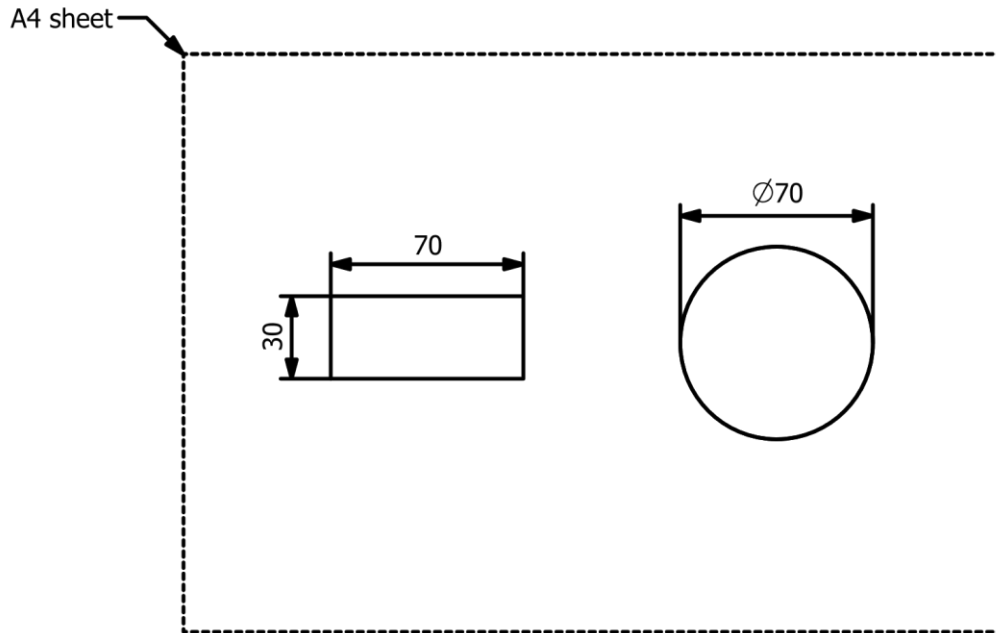


Figure 7: Drawing experiment task: Draw the rectangle and circle shown above, in millimeters.

The experiment consisted of two tasks, denoted T1 and T2 (the experiment tasks can be found in Appendix C). In T1 the participants were asked to draw as accurately as they could within 5 minutes, and in T2 as fast as they could within 30 seconds. They were provided a set of tools they could use for completing the tasks, including pencils, ball point pens, markers, rulers, a compass tool and a protractor (see Figure 8 below).



Figure 8: Tools available during the experiment.

For each task, time was measured as the independent variable. The task was not stopped if the participant used more time than was specified. The time limitations were used as a guide to distinguish between accuracy and speed, and not considered for determining success or failure of the tasks.

After running the experiments, the system was first calibrated (scaled) by capturing a printed one-to-one scaled sketch of the drawing task with Protoboost v2. Then, each drawing from the experiment was captured and converted to vectorized DXF files, as described in Section 3.2.2. The DXF files were then opened and measured using QCAD (an open source application for computer aided drafting). The measured values (W-width, H-height, A-angle and D-diameter) were subtracted by the correct dimensions and used in the analysis, as demonstrated in the figure below. As an example for the notations used, the deviation in width for test T1 is denoted as “W-T1”.

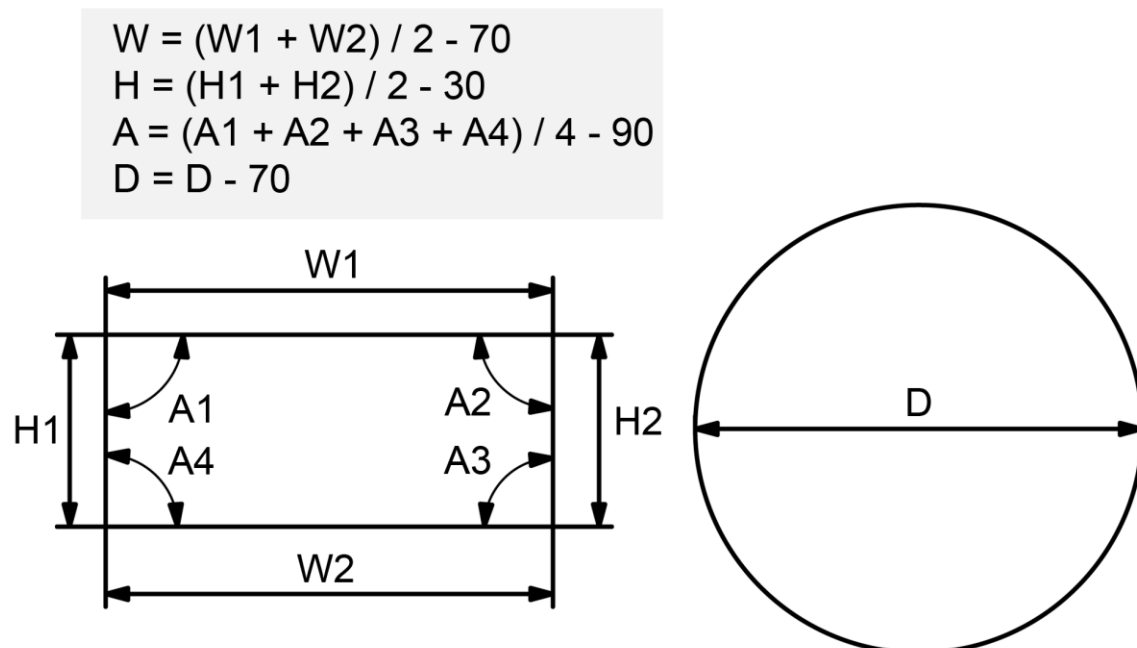


Figure 9: Values used for analyzing the experiment results.

To test if there is a significant difference in the values for T1 and T2, a comparison of standard deviations (F-test) and means (paired samples t-test including Shapiro-Wilk test for normality) is conducted using a statistical software (free trial) called Medcalc by Schoonjans (2016).

3.2.4.3 Experiment Results

The population (N = 14) consists mostly of mechanical engineering students, in addition to a few non-engineering students (4 participants). Deviations from the given dimensions were measured (where 0 would be a perfect ‘score’) for each value, as illustrated in Figure 9. A boxplot of the results is shown in Figure 10 below, followed by time data for the experiment in Table 1.

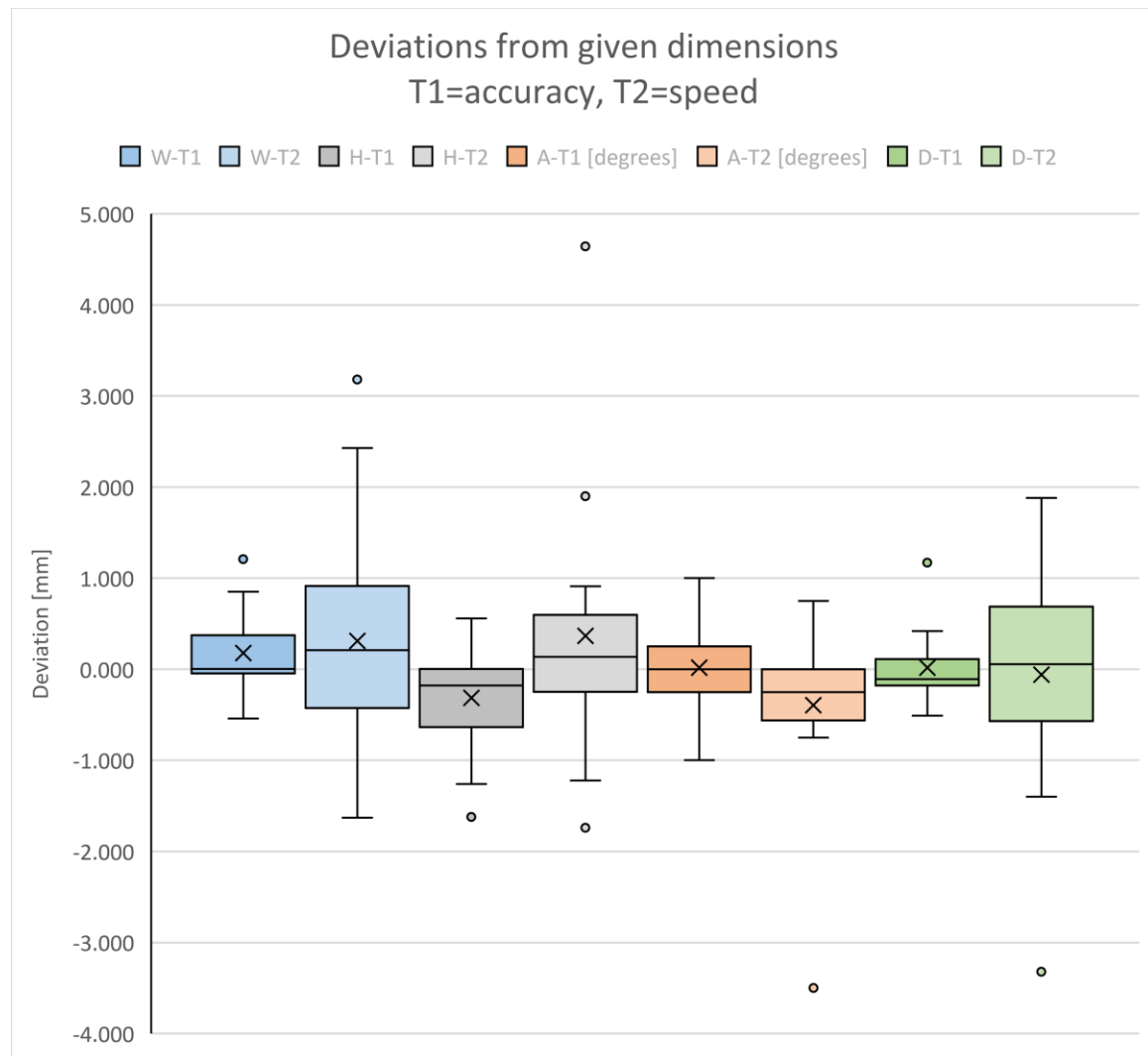


Figure 10: Boxplot of the participants' drawing capabilities. The 'X' represents the average and circles are outliers. Note that the angles (A) are measured in degrees and not millimeters.

Table 1: Time data for the experiment.

<i>Time data</i>	T1	T2
Average task completion time	1min 6.86s	36.36s
Standard Deviation (SD)	1min 1.69s	10.40s
Slowest time	4min 50sec	53s
Fastest time	54s	21s

Almost every participant chose to use a pencil, the compass tool and a ruler. After asking each participant what type of digital tool they would normally use for this type of task, half of the participants would use a CAD tool (in addition to other drawing software) and the other would use sketch or drawing tools (such as Microsoft Paint and Word, Inkscape and Adobe Illustrator).

Some of the captured and processed data is shown in the figures below. Every participant chose to use the compass for drawing the circle, which resulted in the lines being too weak to be detected by Protoboost v2 (see Figure 11). They were instead measured manually with a digital caliper.



Figure 11: Captured BMP image of a test sample and the resulting binary PNM image after using the threshold function. Note the barely visible circle on the top image.

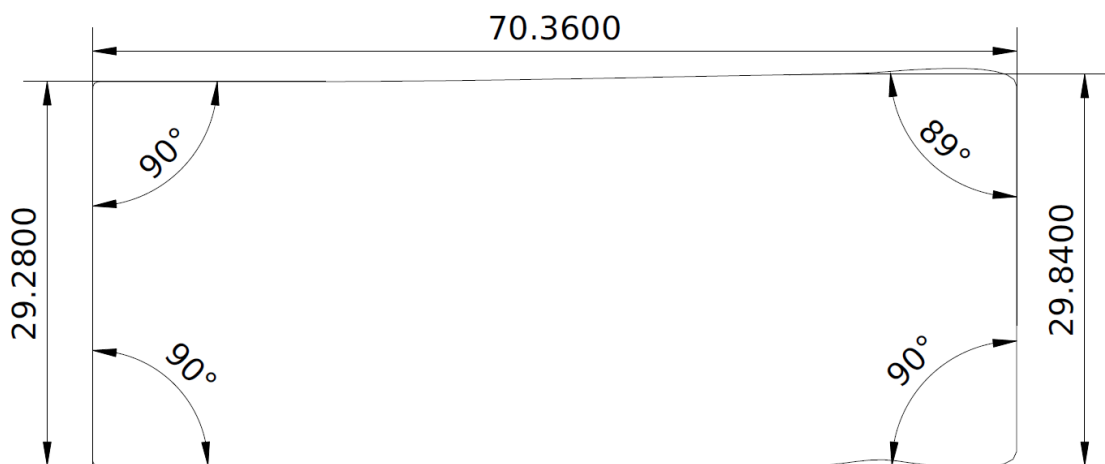


Figure 12: One test sample in vector format (DXF), measured in QCAD.

The F-test is used to compare standard deviations between the two tests. Results in Table 2 show that for every dimension, the deviations between T1 and T2 are statistically significantly different ($P < 0.05$).

Table 2: Results from the F-test.

<i>F-test (N=14)</i>	SD, T1	SD, T2	F-statistic	Significance level
Width	0.46mm	1.31mm	8.1101	P = 0.001
Height	0.59mm	1.51mm	6.5501	P = 0.002
Angle	0.52mm	0.97mm	3.4797	P = 0.032
Diameter	0.39mm	1.30mm	11.1111	P = 0.001

The paired samples t-test showed that there is not a statistically significant difference between the means ($P > 0.1$ in each case), with the height and angle dataset rejecting normality ($P < 0.5$) by the Shapiro-Wilk test. The t-test results are included in Appendix C.

3.2.4.4 Experiment Discussion

With the paired samples t-test results, we cannot reject the null hypothesis for accuracy versus time, implying that the average accuracy is the same when drawing fast and slow. However, the deviations are significantly different according to the F-test. We can argue that time pressure yields lower accuracy in most cases due to the deviation (and variation) in the results. Consequently, based on the F-test, the null hypothesis can be rejected, and the alternative hypothesis “drawing speed influences accuracy” can be accepted.

Based on the most conservative standard deviation obtained in T1 (Table 2), we can argue that the drawing accuracy by humans is roughly $\pm 0.59\text{mm}$, when given enough time (Table 1) and tools (Figure 8). For basic sizes between 30 - 120mm, according to the NS-ISO 2768-1 standard for general tolerances (see Appendix B), this accuracy is considered coarse. This accuracy is usually not suitable for machined parts (e.g. bolts and axles in manufacturing) but is acceptable in many prototyping scenarios (e.g. laser cut MDF prototypes). It is also argued that the time pressured (T2) accuracy of $\pm 1.51\text{mm}$ would be acceptable in many early stage product development projects, as will be demonstrated

further in Practical Examples. We must however reject the null hypothesis and accept the alternative hypothesis: “Humans cannot produce hand drawn sketches for laser cutting that are equally accurate compared to sketches made using computer aided design tools”, due to CAD producing zero deviation.

Since many of the engineering students use CAD, experimenting should be done to determine the tradeoff between accuracy, complexity and speed compared to hand drawn sketches. However, users that are unfamiliar with CAD tools (e.g. those who would use Microsoft Word and Paint) would still benefit from this system, assuming most people can draw by hand and use common drawing tools.

Because the experiment is considered a pilot experiment, it is also limited in its impact and accuracy. A more controlled environment and more participants should be considered for future experiments. Also, a consideration of measuring methods should be reviewed, as the accuracy of the method used in this experiment can be influenced by the one doing the measuring (e.g. curved lines shown in Figure 12 can be interpreted in different ways). The angle is also rounded to the nearest degree, thus less accurate than the other measurements.

There is clearly a preference of using pencils, which also can be erased and modified as an advantage. Some of the sketches were difficult to capture with the current system, often resulting in many vectors from noise as mentioned previously in Early Concept Testing. Solutions and other capturing methods will be discussed in the next subsection, including testing and comparing a higher resolution camera. There is also a tendency for lines to be distorted further away from the camera center. Solutions to this problem will be presented in subsection 3.2.6.

3.2.5 Line Resolution

It was hypothesized that a higher resolution camera will result in higher resolution vectors (more accuracy) and less noise (better distinguish lines from the background). This was tested by capturing a sketch with an iPhone 5s, with a resolution of 3264x2448 pixels (8MP) and comparing it with the same sketch captured with the C920e (2MP) web camera.

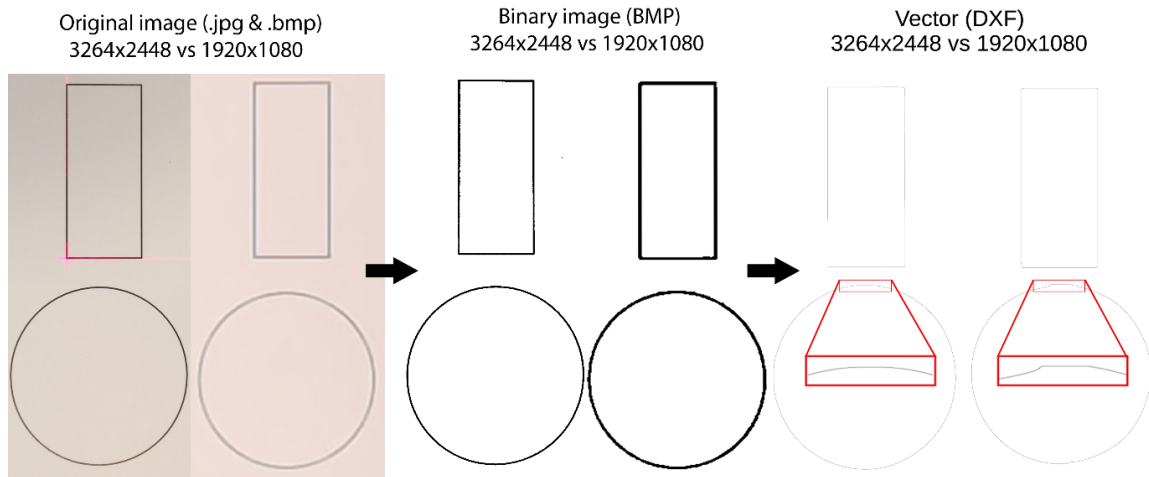


Figure 13: Comparing an 8MP and 2MP camera for capturing and converting a sketch to vectors.

Results are shown in Figure 13 above. The output vectors are more accurate and has more resolution, by using the 8MP camera, as shown on the right illustration in the figure. However, the problem of noise did not seem to be improved.

A quick test was also done by scanning one of the sketches from the experiment in a normal paper scanner. It did not provide any clearer representations of weak pencil lines, where for example the circle was still too weak to be detected. Changing parameters for the scanner was however not tested.

A solution could be to use a friction pen when drawing. They produce strong lines that are easier to detect, while having the ability to easily be removed using friction. Different lighting should also be considered, which might improve detection of weak lines.

3.2.6 Distortion

It was discovered that long straight lines can be bent if they are captured at the outer edges of the camera frame. The program is not implemented with a function to take care of this flaw. However, testing is done with OpenCV using image transformations, including affine transformation and warping, with Canny edge detection to automatically find a square in an image. Proof of the concept is shown in Figure 14.



Figure 14: Detect and transform square example with OpenCV. The square is automatically detected and then transformed.

If the sketch, captured with Protoboost v2, is drawn on a square paper, it can easily be detected and transformed as demonstrated above. If the size of the paper is known (or detected) the correct scale can also be applied.

3.2.7 Practical Examples

An example case of the proposed method is presented in the Appended Conference Paper. In Figure 15 is shown another example using the sketch to laser method. Here a bracket is made for holding a light source above a camera (for a 3D scanning setup). An A4 paper and a pencil was used to make a rough outline of the bracket, by using the actual objects (camera and Rexroth profiles) as guide. Afterwards, the sketch was refined further before capturing it with Protoboost v2. In the laser cutting software (Gravostyle), the weaker lines (circles from using a compass) were redrawn.

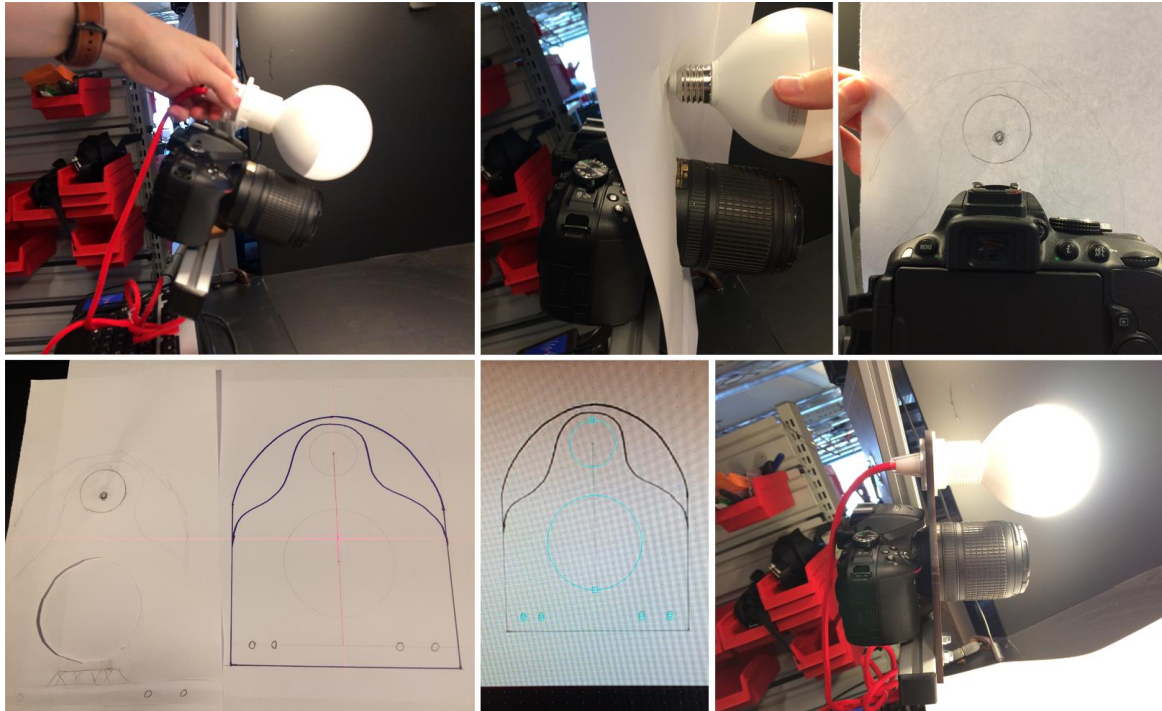


Figure 15: Light attachment example using sketch to laser with Protoboost v2. The first (top left) image shows how the light source was intended to be attached, with steps in between leading to the final solution shown at the end (bottom right).

In the next example, a cable hook is created directly from a sketch. The hook is attached to a moderately complex profile. The shape of the profile is captured by holding a paper against it and applying pressure on it with a yellow colored chalk. A sketch is then drawn around the roughly highlighted profile by free hand, with an accuracy of roughly $\pm 2\text{mm}$ relative to the profile. After capturing the sketch with Protoboost v2 and opening the generated DXF file in Gravostyle, the cutting sequence was manually applied before cutting the part out of an MDF plate. No modifications were needed for the sketch itself. This example is illustrated in Figure 16 below. The part has a lot of room and does not fit tightly, but it is fully functional for its intended use (i.e. low-fidelity prototype).

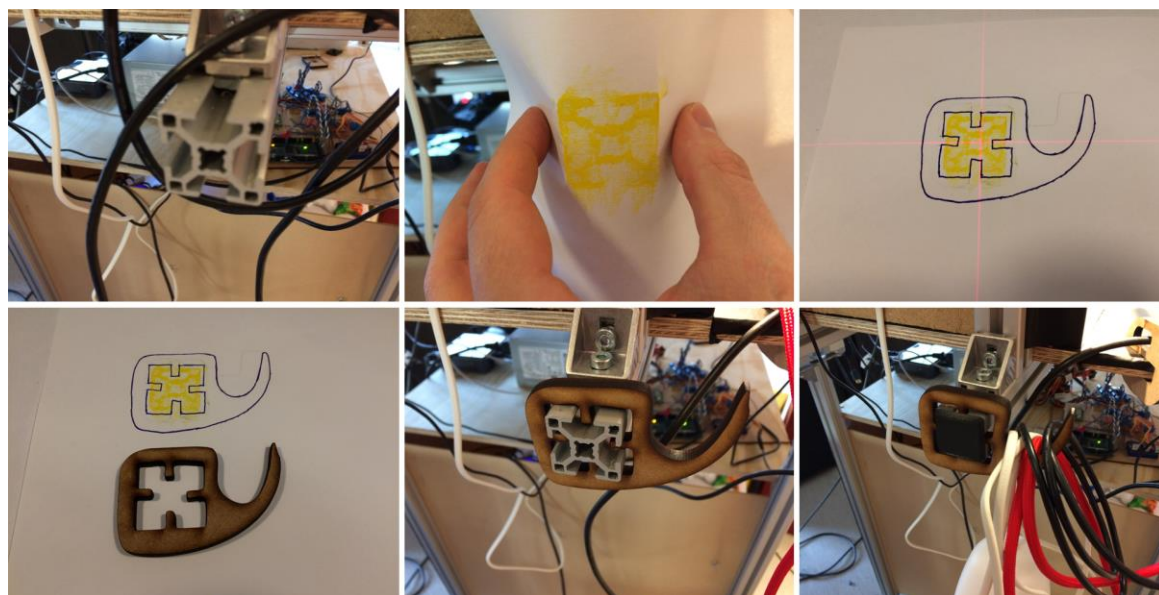


Figure 16: Cable hook example. Top images: Capturing the shape of the Rexroth profile with a yellow chalk, and drawing the sketch using the shape. Bottom images: comparing the laser cut MDF part with the original sketch, followed by its practical application.

3.3 Video with Serial Data

3.3.1 Introduction

It has become more common to use microcontrollers in product development. Ready-made microcontrollers, such as the Arduino (described more in detail at Subsection 2.3.5 on page 13), provides a simple and fast method for interacting with the physical environment by controlling actuators and using sensors.

Even though it is common to use the Arduino Integrated Development Environment (IDE) and its serial monitor for displaying how microcontrollers work programmatically, it is often not convenient for documenting or sharing this knowledge. It can also be a steep learning curve to simply log sensor data from a microcontroller to a computer, to enable plotting and other forms of data processing.

To simplify the documentation of microcontroller-based prototypes, and logging output for further processing, an attempt was made to record video of a prototype while displaying the serial output in real time on the video. The proposed method is presented in this chapter, including its limitations. Improvements to the system (C++ program) were made after testing the early concept.

3.3.2 Preliminary Concept Testing

One of the first prototypes (C++ programs) made for recording video with real time serial data from an Arduino recorded at a fixed duration of 20 seconds. The program was used with a Logitech C930e web camera, capable of recording full-HD (1080p) video at 30fps. For each frame that was captured, serial data was read and put onto the frame, if available. Testing showed that 20fps was the maximum recording speed possible with the program. Additionally, to make it more flexible it was necessary to enable a way of stopping the recording.

To keep interaction with Protoboost v2 in a simple way, the RFID reader (see Figure 5 on page 16) for starting the recording (activating Protoboost v2) was chosen to also stop the process. Node-red has a Human Interface Device (HID) node that can receive byte-arrays from HID-devices and send it to another node, which is used in the Protoboost by Sjöman et al. (2017) for identifying the user with their personal RFID access card. To utilize the RFID-reader for communicating with a running program, another program was made (Appendix A-2) for writing to a (status) text file when the RFID-reader is used. Another running program can then read the text file simultaneously and do actions based on what is written in the file.

For each frame captured, the program will read available serial data, store the data in a C++ vector (array) of strings, put the strings on the frame line by line, with specified text color, font and size, and read the text file to check if status has changed to stop recording (RFID-reader activated).

The frame rate was in some scenarios lower than the observed maximum of 20fps, most likely due to the intermediate steps during recording. It was thus desired to discover limitations of the hardware used and how frame rate is changing based on different parameters, and if it is possible to capture high fps consistently by making the program faster and more reliable without resorting to more costly equipment. This information can also be useful for other applications where the speed of image capturing can influence the overall system performance (e.g. computer vision in robotics).

3.3.3 Measuring Performance

To measure the performance of the program and the camera (specifications provided in Table 3 below) for different parameters (independent variables), time in milliseconds was

measured (dependent variable). For each iteration or loop in the program, the measured time was appended to a text file. Pseudo code is depicted in Figure 17, where “CODE” is where the main code being measured is located. Although the program must measure time, write it to a text file, count frames and check in a while-loop when to stop, these steps are considered fast enough to be ignored in the calculations (running the code in Figure 17, with “CODE” empty, results in 0 milliseconds being measured as the total runtime). The program was run 3 times for every independent variable, and the average for each data point was used to generate one dataset for further analysis. It is also important to note that during the tests, no motion is recorded in the video. Under normal circumstances motion is usually present, which can reduce frame rate.

```
num_frames = 0
start_clock = now
while (num_frames < 300)
{
    /*
    CODE
    */
    num_frames += 1
    clock_now = now
    write_to_file << clock_now - start_clock
}
```

Figure 17: Pseudo code for the test program. “num_frames” can also be interpreted as number of iterations.

Table 3: Hardware specifications.

Camera	Logitech C930e web camera
Resolution	1920x1080, HD 1080p
Video extension	.avi (Audio Video Interleave)
Frame rate capacity	30fps @ 1080p
Computer	Intel NUC, Core i3-7100U

3.3.3.1 Codecs

The aim of the first tests were to discover how different codecs (compression formats) influence the video capturing speed. By capturing 300 frames and measuring the time between each, it is possible to calculate the frame rate in frames per second (fps) by simply

dividing one (frame) by the amount of time used when capturing the frame. The way to record video using OpenCV is by first capturing a frame, then adding the frame to the video using the video writer class using a specified codec. Codecs and compression formats are denoted by their Four Character Code (FOURCC) for the remainder of this chapter.

Many different codecs were tested, and the most noteworthy results are plotted in Figure 18.

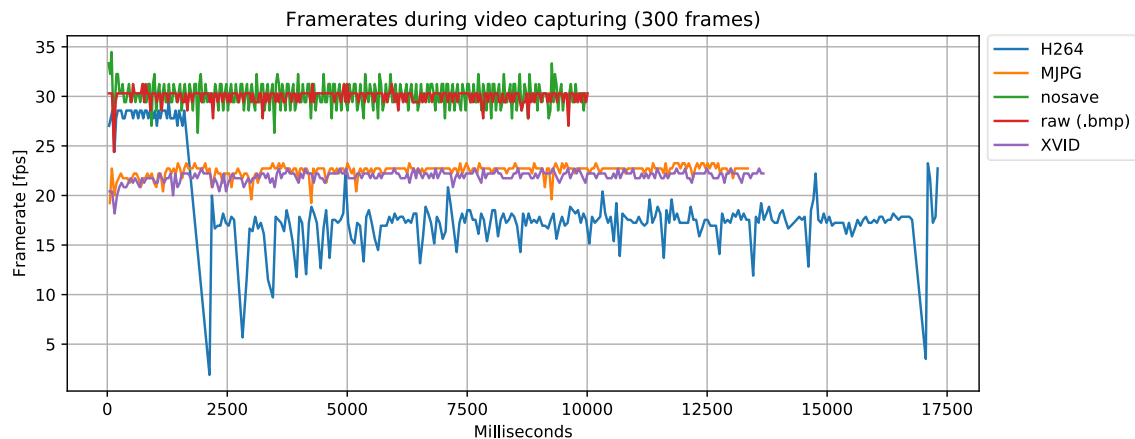


Figure 18: Frame rates for different codecs while recording 300 frames.

Capturing frames without decoding or saving them (nosave) and only capturing the raw data (raw .bmp) results in the maximum possible fps specified in the datasheet for the camera, as expected. Capturing and downloading the raw image for each frame is more stable ($SD = 0.65$) compared to simply capturing a frame without doing anything with it ($SD = 1.25$). The H.264 codec can capture high fps in the beginning, but oddly drops to around 1.4fps on the 46th frame and varies considerably for the remaining frames ($SD = 4.42$). MJPG and XVID have similar recording speeds, where MJPG has a slightly higher mean frame rate and standard deviation. With MJPGs 89.8MB file size against 10.9MB with XVID for the 300 frames video, and a subjective assessment of the quality of the video frames (see Figure 19 below), it seems that XVID is arguably the better suited codec between the two, with less space requirement and no distinctive difference in image quality.



Figure 19: Snapshots from recorded video using MJPG (left) and XVID (right) codecs, with no apparent difference in quality.

3.3.3.2 Serial Communication Speeds using Arduino

Serial data transfer speed between an Arduino Uno and the NUC was measured to get an idea of how communication speed varies with regards to the amount of data sent. Several strings containing a set of characters were sent from the Arduino and read character by character in the program on the NUC. Time was measured between each newline (character ‘\n’) received. Because both receiving serial data and iterating through each character in the C++ program was found to be on average less than 1 millisecond (tested by receiving and iterating up to 1100 characters between newlines), the actual time measured in the program (Appendix A-3) is the serial printing speed of the Arduino. Consequently, the rate of messages sent from a microcontroller and read by the program should not directly influence the video recording speed. Arduino Uno serial printing speed-rates are plotted in Figure 20.

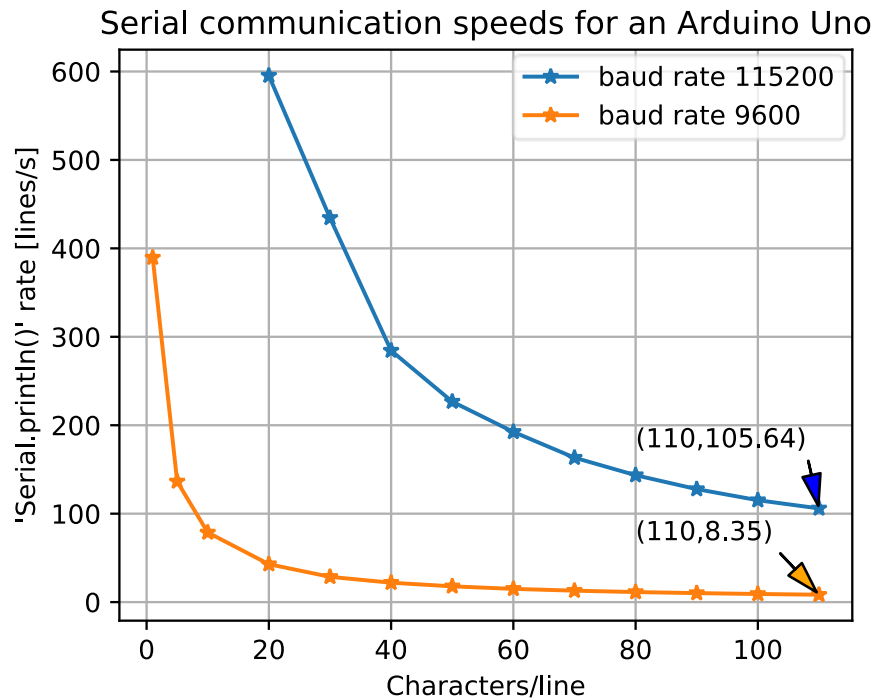


Figure 20: Plot showing the rate of messages (lines per second) at which the Arduino Uno is capable of sending through serial communication, based on the amount of characters in each message (separated by newlines).

The code running on the Arduino during testing is simply a loop repeating the line “Serial.println(message)”, where ‘message’ is a string of random characters sent with a newline at the end. The baud rates at 9600 (communication at 9600 symbols per second) and the maximum of 115200 were tested. It should be noted that for some sequences during the 20 and 30 characters per line tests were equal in time, for the 115200 baud rate, causing a printing rate of infinity. These were removed before the plot shown in Figure 20.

3.3.3.3 Writing Text to Video Frames

The last step for being able to display serial data from microcontrollers on the video in real time, is writing the data onto each video frame. This is accomplished by the ‘putText’ function provided by OpenCV, which renders a text string into an image with specified text location, font, size, color and thickness. To make the display smooth and familiar, a function was made to print the data in the same way that is used in the serial monitor for the Arduino IDE, where lines are printed from top to bottom. Every new line received from the Arduino is first appended to a C++ vector of strings. Based on frame and text size, the number of lines that can fit into each frame is calculated. The text and frame size used for the testing allowed a total of 34 lines to be added, as shown in

Figure 21 below. When the vector reaches the number of lines limit, the first entry in the vector is removed, and the new line of text is added to the back of the vector. Then, for each frame, a for-loop iterates through the vector and prints each line at the appropriate position on the frame.

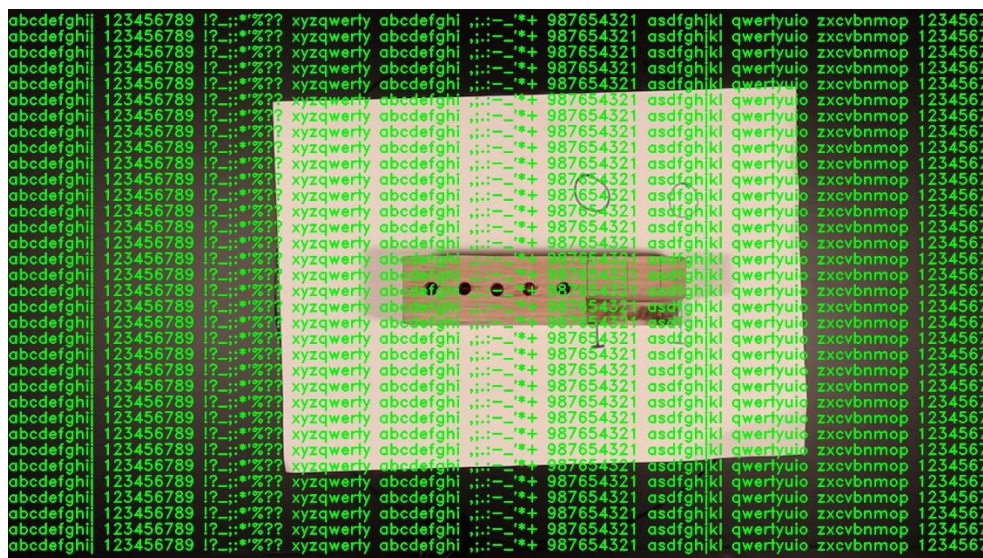


Figure 21: A frame with 34 lines with 110 characters each (110 characters/line), with the current text and frame size.

This process gets slower and slower for each new line added to the vector, until a constant is reached when the vector is full and starts removing lines before adding new. In the following graph (Figure 22), the time used by the program to add 34 lines of text to a frame is measured, and the corresponding line writing-rate (theoretically equal to frame rate capacity) is calculated and used in the y-axis. Each time interval is measured by writing a constant number of characters per line to the frame. For example, with 60 characters/line, a total of 60 times 34 lines (2040) characters are written to each frame. The lines were written to a frame with a resolution of 1920x1080 pixels. Printing rates for an Arduino Uno from Figure 20 are included in Figure 22 for comparison.

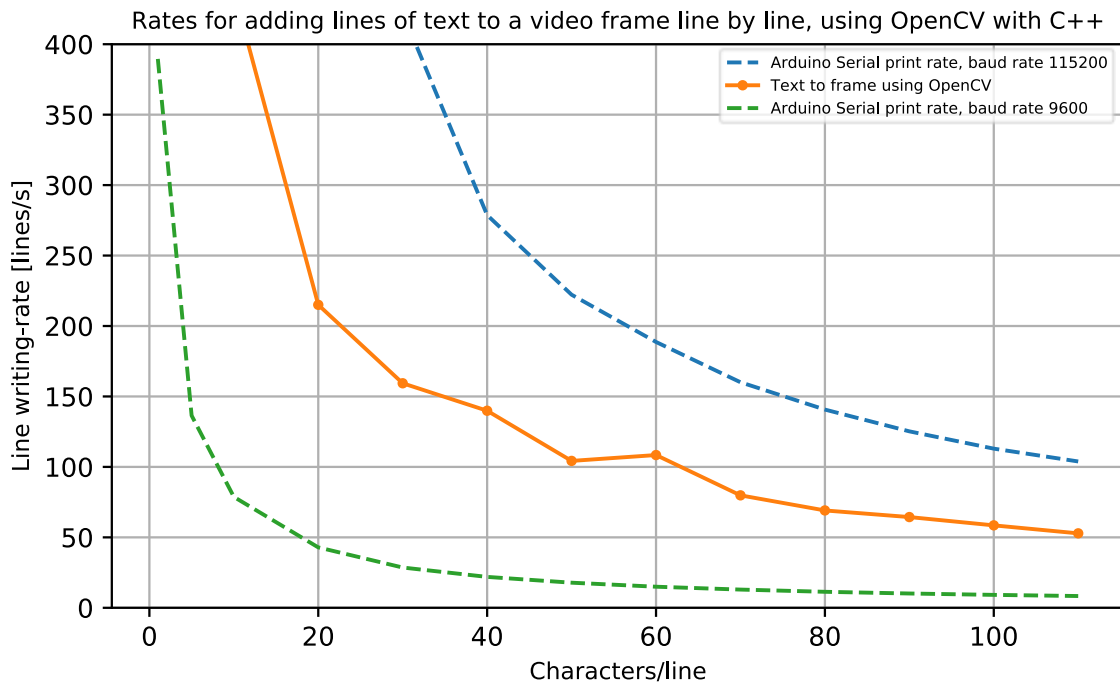


Figure 22: Rates for adding text to a video frame, including printing rates for an Arduino Uno.

From Figure 22 it is clear that, with a baud rate of 115200, the Arduino can send data faster than is possible to keep up with by writing the data to the video frame. In a worst-case scenario, this could delay the video recording and result in a very low frame rate without being able to display all the data.

To check the behavior when writing text to each frame while recording a video, the “text to frame using OpenCV” plot from Figure 22 was combined with the frame rate of both recording with raw images and the MJPG codec, then tested in practice. The results are shown in Figure 23, where the lines represent the calculated frame rates, and “X” the measured frame rates from testing.

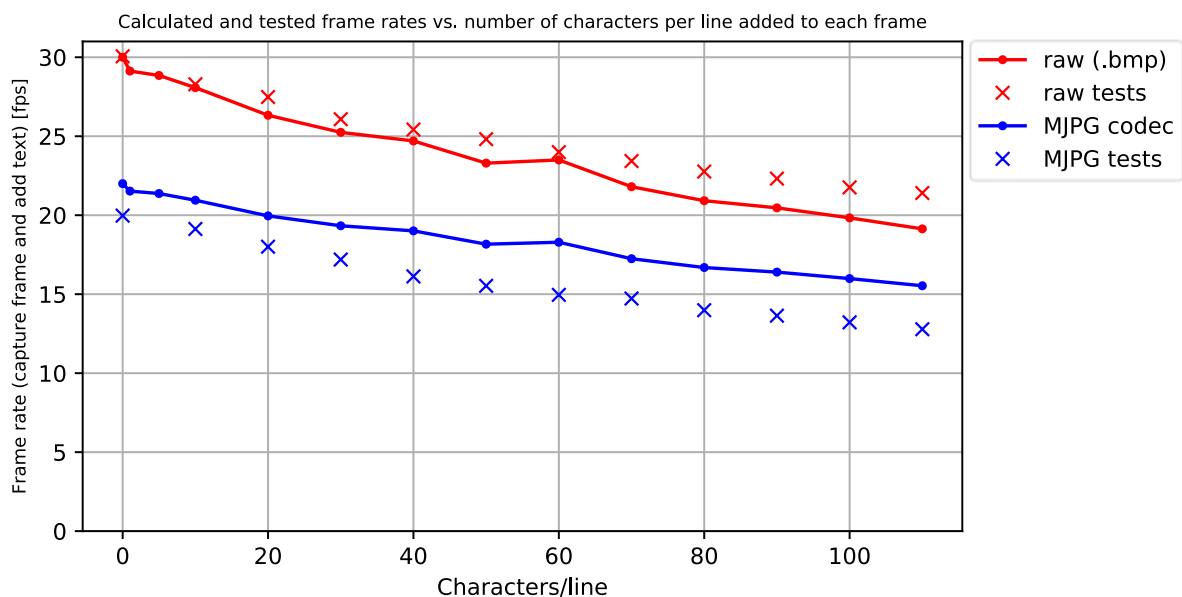


Figure 23: Plot showing the change in frame rate for different amounts of characters appended to each frame, where lines represent calculated frame rates from previous tests and 'X' representing actual test results.

The prediction and actual test results show a correlation, where frame rate is reduced when more characters per line are written to each frame.

3.3.3.4 Discussion on Performance

As the rates for reading and printing data to a frame are relatively fast, the frame rate of the camera and writing text to frame are the bottlenecks. To account for all the factors discussed in this section up until now, it is hypothesized that separating parts of the program, by capturing data before concatenating them, will increase frame rate and robustness.

3.3.4 Implementing Improvements

To test the hypothesis, a new program was made that first records video by saving only the raw images and writes frame numbers with corresponding timestamps to a text file, and simultaneously writes the received serial data with timestamps to another text file. After the recording is stopped, the images (frames) saved are loaded one by one, while reading the text files. Serial data can then be added to the appropriate frames, by comparing the timestamps in the text files, to get a video with real time serial data. Thus, if the frame rate of the camera can reach 30fps with 1080p quality, it should be possible to include serial data from microcontrollers in post processing without any loss of speed or quality. Then it is also possible to do heavier calculations before making a video containing the data, for example some forms of plotting.

3.3.5 Results and Discussion

After testing the new program (Appendix A-3) using the post processing technique, by printing up to about 110 characters per line (see Figure 24 below), the frame rate can reach the maximum of 30fps. The new program can capture 1080p 30fps video with large amounts of serial data. Frame rates were observed by simply recording a stop watch and counting the number of frames during a one second period. Observations showed that 30 frames passed for every second. Thus, we can confirm the hypothesis, that it is indeed possible to maximize frame rate, stability and data collection with varying data transfer rates by including a post processing stage. A function is also implemented for reading through the text file containing frames and timestamps, to calculate the average framerate before creating the video file (in case something causes the frame rate to decline, e.g. other processes running in the background).

A problem with capturing both frames and serial data in series, is the timestamp accuracy. For each frame captured, there will be a close to equal timestamp generated and applied for each consecutive serial data (line of text). The serial data is provided in a separate file that can be used for other data processing later, such as plotting, and should have its timestamps as accurate as possible. Therefore, it would be beneficial to capture frames and serial data separately to get more accurate timestamps. This can be accomplished using multithreading.

It is possible to utilize multithreading to simultaneously process data. Multithreading has not been implemented with the video recording program, but testing has shown that it works for running two threads (two different processes in parallel). The number of threads is limited by the two independent CPUs (cores) in the NUC. This method could also be combined with the post processing stage, to increase the overall program speed.

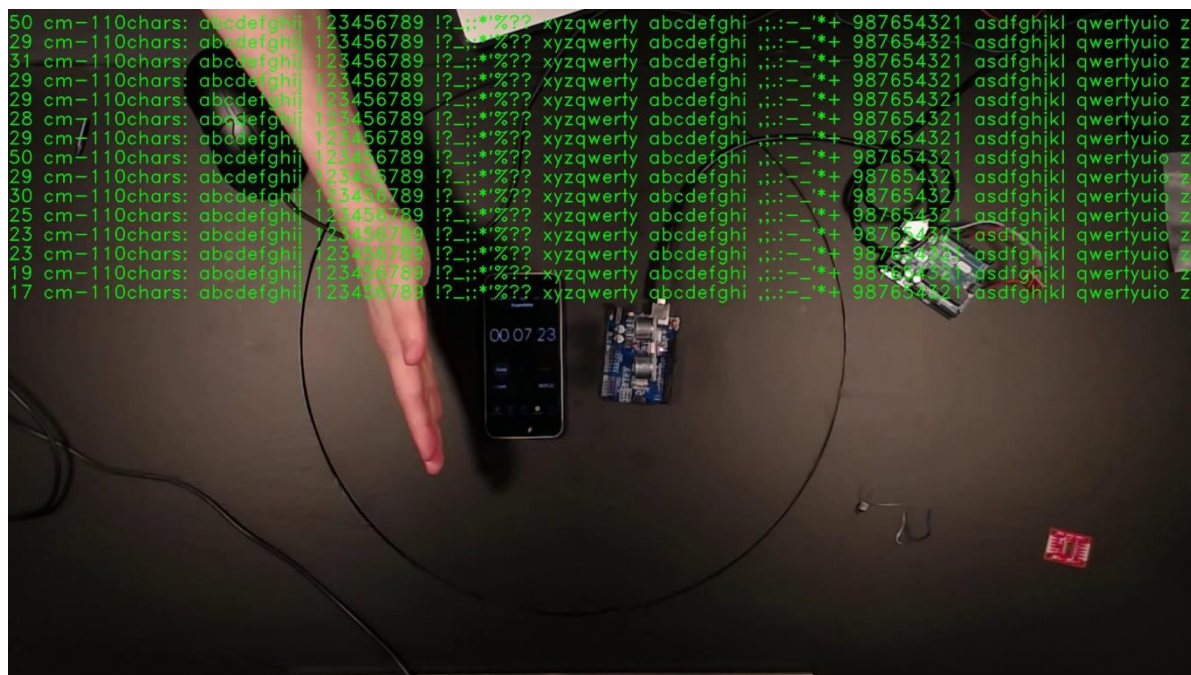


Figure 24: Snapshot from a recorded 30fps video, with real time serial data from an Arduino Uno.

3.3.6 Practical Examples

A test example was made by recording a video of an ultrasonic sensor programmed with an Arduino Uno. The video can be found on YouTube by following the link in Figure 25. The text file of the serial data, generated during the recording, was simply dragged into Excel and plotted as shown in Figure 26 below.

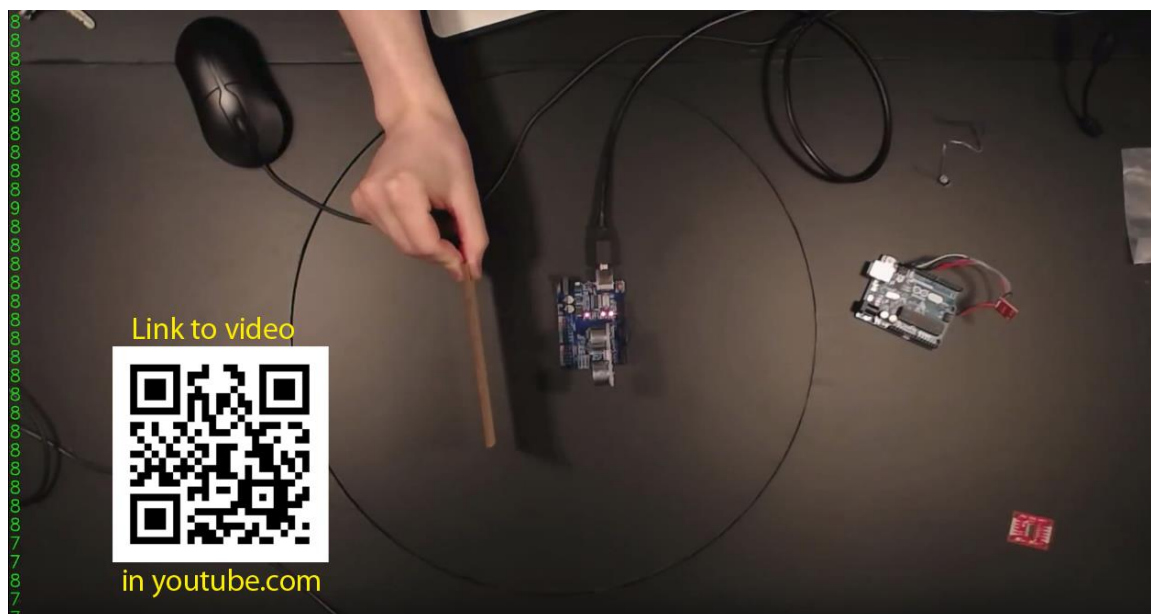


Figure 25: A frame from the ultrasonic sensor video example. QR-link: <https://youtu.be/LtFkuiJcVeQ>.

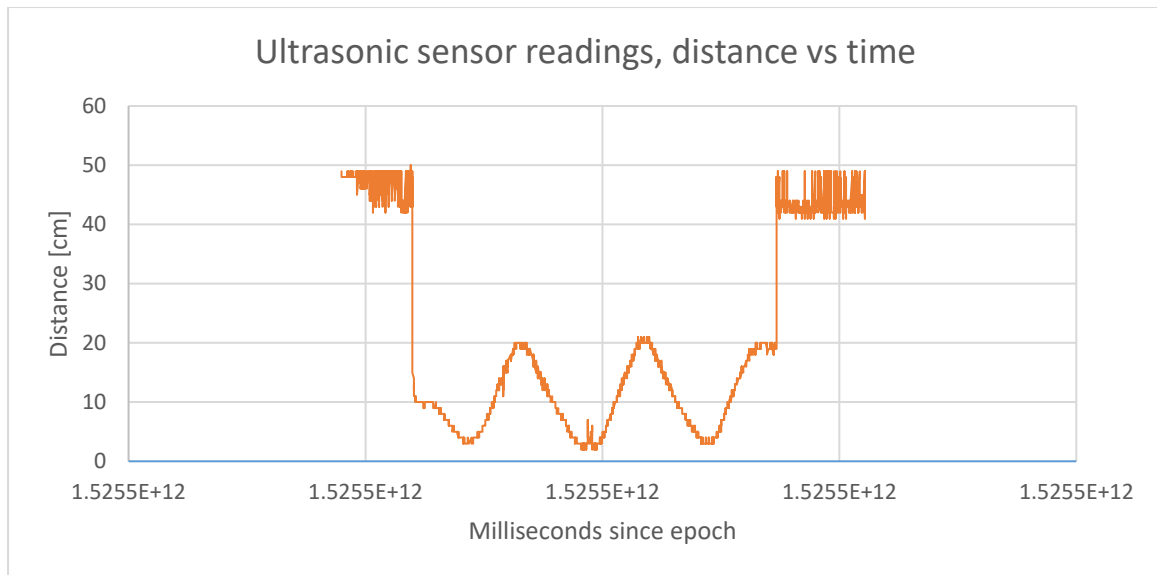


Figure 26: Plot of the ultrasonic sensor readings versus timestamp data.

A microcontroller-based modular robot system, consisting of different modules (boxes) that can communicate with each other through radio signals, has been developed in a previous project at TrollLabs. To showcase how the prototypes could be used, a handful of modules was created: a joystick, car, robot arm, power supply and sensor module (see Figure 27 below). These modules have been documented using the proposed method and uploaded to YouTube (link in the figure below). One of the videos made demonstrates how different modules react to signals received by the joystick module. By simply connecting the joystick module to Protoboost v2, it was easy to demonstrate what types of signals were sent to the other modules. This example is also included in the Appended Conference Paper.

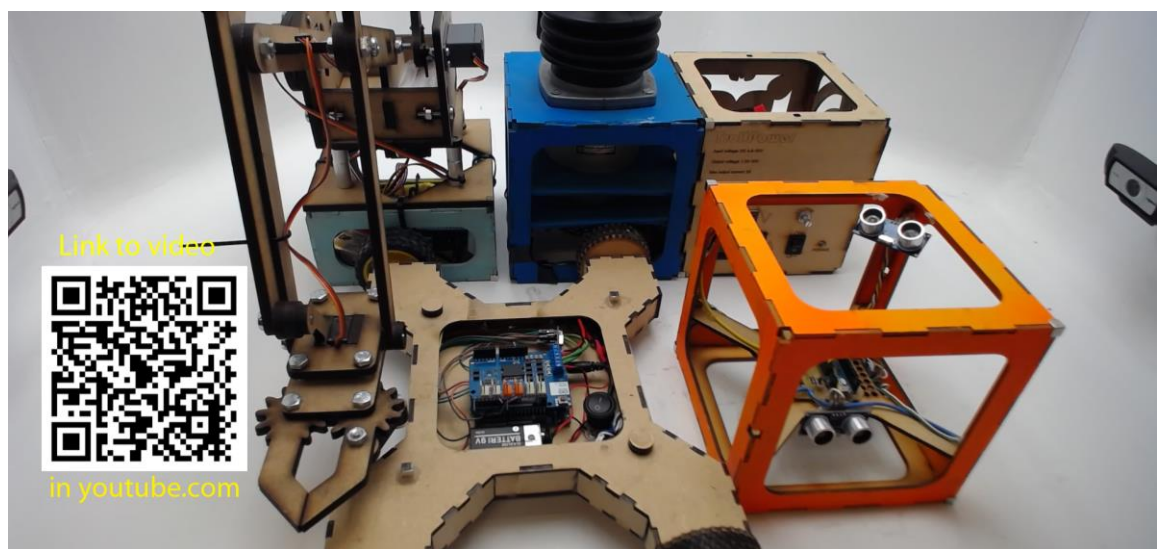


Figure 27: Prototypes (modules) from the modular robot system. QR-link to a playlist presenting the modules, captured with the video with serial data method: https://www.youtube.com/playlist?list=PLj9XjbRcnJ_0AMBmT0Idtu9-BngYMOqmp.

3.4 3D Scanning

3.4.1 Introduction and 3D Scanning Pipeline

To introduce 3D scanning possibilities with Protobooth v2, 3D reconstruction systems utilizing photogrammetry has been implemented and tested. In the pre-master's project, COLMAP and OpenMVS was found to provide satisfactory results.

A DSLR camera and turntable (see Figure 5 on page 16) is used to automatically capture an object from several different viewpoints. The images are further processed to reconstruct a digital 3D model. The initial stages for the 3D reconstruction pipeline consists of three steps: feature detection and extraction (for each image), feature matching and geometric verification, and lastly combining data from the first two steps to generate points by triangulating features, ultimately reconstructing a sparse point-cloud of the object or scene. The reconstruction is refined using bundle adjustment. These three steps are accomplished using COLMAP (Schonberger & Frahm, 2016; Schönberger, Zheng, Frahm, & Pollefeys, 2016), which is a general-purpose SfM and MVS pipeline with a command-line and graphical interfaces. For the aforementioned steps, COLMAP will also automatically extract focal length information from the embedded EXIF (Exchangeable Image File Format) information and undistort images.

OpenMVS ("OpenMVS," n.d.) is used for the final stages of reconstruction. Camera poses, their corresponding undistorted images and the sparse point-cloud, generated using COLMAP, are used to export an NVM project file (N-View Match format used in VisualSFM by C. Wu (2013)). The NVM format is supported by and used as input for the OpenMVS pipeline, which contains the SfM workspace with image paths and 3D models (i.e. point-clouds). OpenMVS consists of four modules: Dense Point-Cloud Reconstruction, Mesh Reconstruction, Mesh Refinement and Mesh Texturing. The dense point cloud module creates a more complete point-cloud, where speed is prioritized, based on the Patch-Match algorithm (Barnes, Shechtman, Finkelstein, & Goldman, 2009). The mesh reconstruction module estimates a mesh with the dense point-cloud as input. It is based on an algorithm by Jancosek and Pajdla (2014) that is robust against outliers and weakly supported surfaces. The mesh refinement module is based on the paper "High Accuracy and Visibility-Consistent Dense Multiview Stereo" by Vu, Lbatut, Pons, and Keriven (2012), which is the main feature that has provided great results from using OpenMVS compared to other methods. In the final module texture is added to the model.

OpenMVS has implemented a technique by Waechter et al. (2014) for this step, which can automatically and efficiently handle issues such as inaccurate camera parameters, image scale variation, blur and exposure variation in the images. Steps and outputs from the complete 3D scanning pipeline is illustrated in Figure 28 below. As example, the illustration uses a prototype of a modular sensor platform, that was captured from several angles using Protoboost v2.

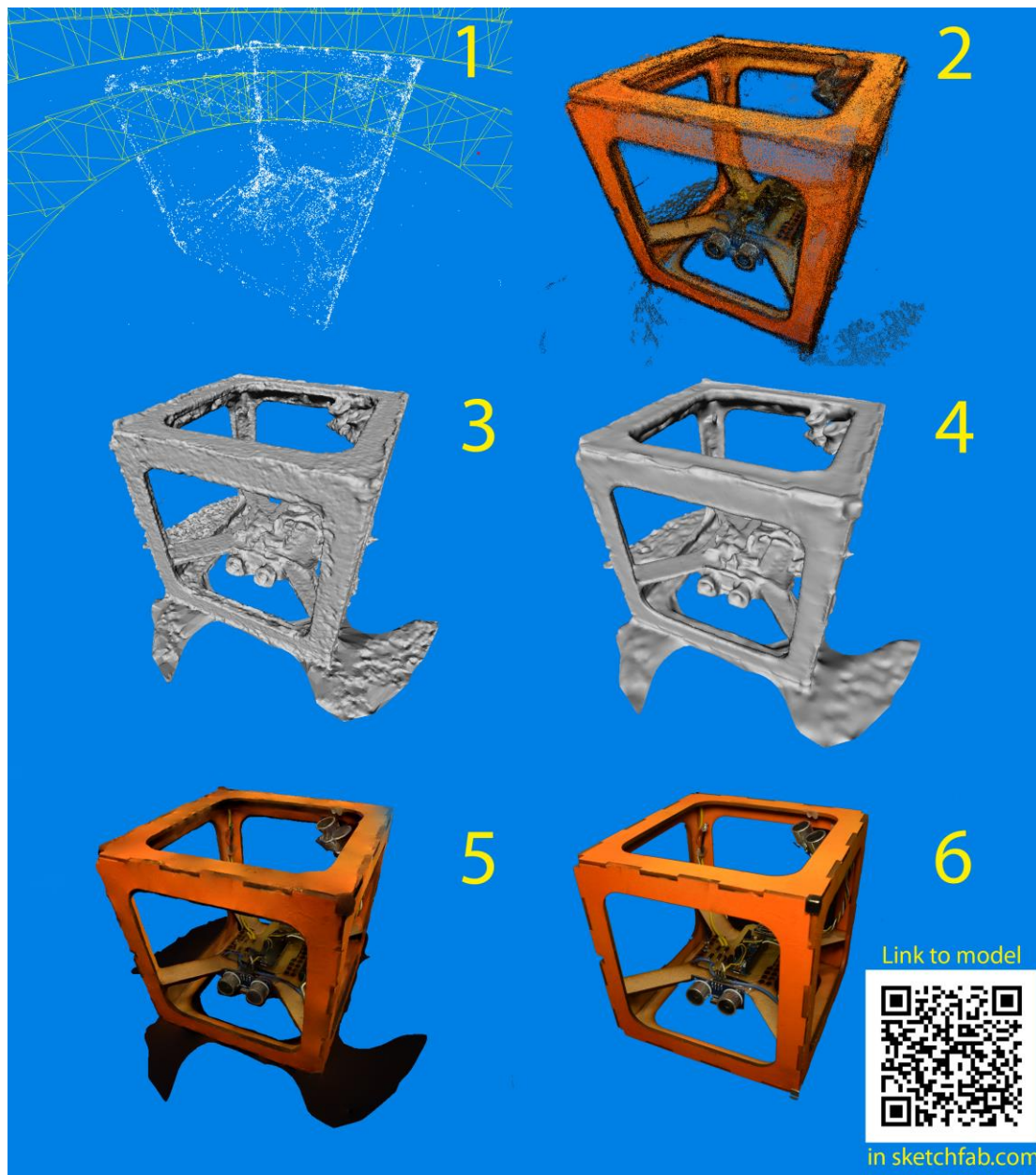


Figure 28: Outputs from the 3D scanning pipeline using COLMAP and OpenMVS: (1) Sparse point-cloud with camera poses (COLMAP). (2) Dense point cloud. (3) Mesh reconstructed from the dense point cloud. (4) Refined mesh. (5) Refined mesh with texture. (6) Image of the real object for comparison. QR-link: <https://skfb.ly/6xnGz>.

Each step mentioned in the previous paragraphs are executed through command line arguments, provided by compiled source codes from COLMAP and OpenMVS. A program

is made with C++ that automatically executes each step sequentially while recording process duration (Appendix A-4). The program only needs one input, which is the path to a folder containing images. The folder is used as the workspace, where all outputs are saved. The main outputs include point-clouds and meshes with and without texture, in Polygon File Format (PLY) and MVS formats. PLY files can be opened and modified with e.g. MeshLab, and the MVS files can be viewed using the Viewer program provided by OpenMVS.

Since the open source algorithms and programs have been made by experts and highly skilled people within the broad field of computer science, more effort has been put on using and testing the system for practical applications relating to early stage product development, in this thesis. What follows is an overview of general usage and results from using the system, including limitations and best practices, as well as some post processing techniques to get the most out of the proposed application.

3.4.2 Practical Testing and Experimenting with Photogrammetry

3.4.2.1 *Using 3D Scanning with Protoboost v2*

Using the proposed 3D scanning pipeline with Protoboost v2 is simple: the object must first be placed at the center of the table (marked by a cross laser), then the DSLR camera must be moved to a position where the whole object is within frame, adjust zoom and focus, and finally scan an RFID card to activate the capturing process (turntable and camera).

The camera captures a total of 28 images while the turntable turns at 2.1rpm (limited by the capturing and downloading speed of the camera using gPhoto2). This gives a viewing angle of 12.9 degrees, which is within the typical optimal range of 5-15 degrees according to Furukawa and Hernández (2015). After this process, lasting roughly 30 seconds, the object can be removed, and the reconstruction is processed automatically. A typical object scanned in Protoboost v2 takes about 7-15 minutes to reconstruct (using another computer than the NUC, discussed later in Subsection 3.4.2.6) but is highly dependent on the size and complexity of the model. It is usually enough to do one scan, but if important areas of the object cannot be captured at once, it is possible to move the object and capture another 28 images. As long as the next scan(s) contain overlapping features with the other image set(s), COLMAP will be able to match features in the images and construct a sparse point cloud for further processing.

3.4.2.2 Scene and Camera Settings

To only reconstruct the object being scanned, the background should be removed from the images. This process is difficult in practice, but approximations can be made using a homogeneous background. To keep the object in focus with regards to lighting and exposure, a matte black background is used in Protoboost v2. This will reduce reflections from the surrounding compared to using lighter colors such as white or green.

The shutter speed of the camera is kept relatively fast to avoid motion blur while the turntable rotates the object. The camera should capture the whole object in focus, as far as possible, to preserve details. A high f-number should thus be used, which is the ratio between the focal length and effective aperture diameter. Since a higher f-number will reduce aperture it will also reduce the amount of light that reaches the image sensor, which is compensated for by using a low ISO setting. Another benefit by using a low ISO setting is that the amount of noise in the image is kept low. Setting the f-number to 9, ISO to 100 and shutter speed to 1/40 seconds has worked well for many 3D scans with Protoboost v2, using the Nikon 5300D camera with a “*AF-S DX NIKKOR 18-140mm f/3.5-5.6G ED VR*” lens. Images are captured with a resolution of 2992x2000 pixels. Higher resolutions can increase detail and quality but can take longer time and require more memory to process. Computer memory is the most crucial element during the reconstruction process, especially during mesh refinement, and can result in failure if the capacity is reached.

3.4.2.3 Increase Robustness

Using markers can help the SfM algorithm to align photos and improve feature detection. It is not always necessary, but testing has shown that in some cases they provoke successful reconstruction of models where they otherwise would fail. An example is shown in Figure 29 in which the scan using markers succeeded in reconstructing the model and the one without markers failed. The numbers on the markers have no purpose other than to differentiate the otherwise identical sticky notes used as markers.

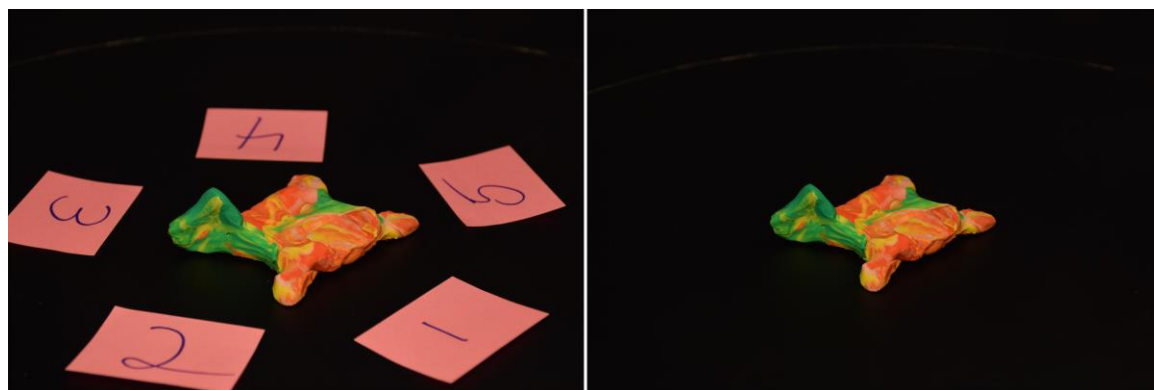


Figure 29: 3D scan setup with and without markers. The setup using markers resulted in a successful reconstruction, whereas the one without failed.

Sometimes it is good practice to raise flat objects, or lean them against some form of support, to capture the largest and most important surfaces in one scan. The model shown in Figure 29 was used as an example in the Appended Conference Paper, in which the model was supported vertically to capture the largest areas. The scan used in the paper was significantly improved compared to the setup illustrated in the last figure. Additionally, when the object is flat the camera has to focus closer to the table, which can in some cases result in more reflection disturbing the end result.

Modelling clay is particularly suited for 3D scanning with photogrammetry due to its uniform and Lambertian texture, especially when mixing different colored clay to increase the number of unique features detectable by the SfM algorithms.

3.4.2.4 Scale

In SfM with a single camera, the model can only be recovered up to scale, meaning that the scale cannot be calculated automatically, although solutions exist that can resolve this issue, e.g. by using refraction (Kume, Fujii, Yamashita, & Asama, 2016). To calculate the scale factor, a known size must be measured on the digital model. The scale factor is given by dividing the known size with the measured size.

To check if a correlation existed between the scale factor, camera parameters and distance to object, a short experiment was conducted. A clay model was scanned multiple times with the camera at different distances and focal lengths. Although the sample size was small (15 scans), no correlation was found for being able to automatically calculate scale factor.

3.4.2.5 Scanning Small Objects

An experiment, or rather a challenge, was conducted to discover if very small objects can be scanned using the proposed system. The object used for the challenge was an otolith, which is a calcium carbonate structure from the inner ear of a fish. The otolith used, shown in Figure 30, is approximately 22x8.5x1.8mm in size. A Sigma 105mm f/2.8 EX DG Macro Lens was used to capture the object.

The reconstruction failed due to the small thickness of 1.8mm. However, after removing images containing the thin sides, reconstruction succeeded in making a 3D model of one of the larger sides. The process was done to each side and MeshLab was then used to glue the two models together into one.

Another challenge with the small object was to capture a clear image, as the focal plane is very fine when using a macro lens. Figure 30 shows two very similar images of the otolith, where the middle one is slightly more out of focus, which ultimately resulted in reconstruction failure. The object must be placed at the center of the turntable very precisely to remain in focus while turning.

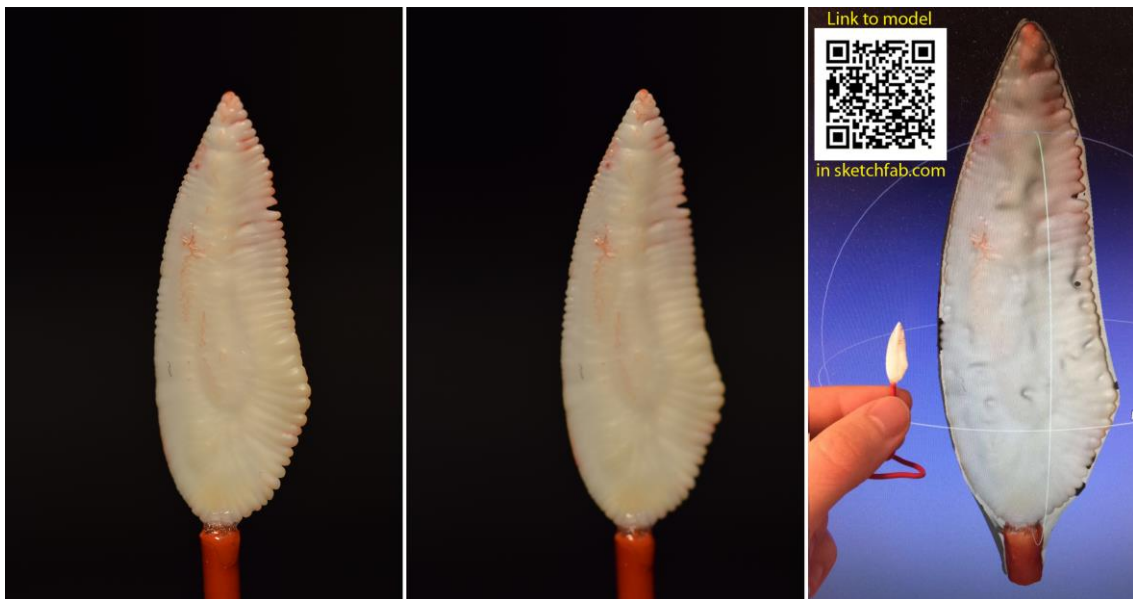


Figure 30: Left image is taken from the set that successfully reconstructed a 3D model, and the middle from a set that failed. The middle image is slightly more blurred (out of focus). Results from the successful scans is shown next to the real object on the right image. QR-link: <https://skfb.ly/6yYCW>.

Although the result is not perfect, the general shape is noticeably accurate. Compared to the limitations of range sensor-based 3D scanning equipment for objects at this scale, the results are positively promising.

3.4.2.6 Limitations

Texture can be a challenge for photogrammetry, as mentioned in Subsection 2.3.2. In Figure 31 several different objects are shown that failed during reconstruction.



Figure 31: Typical objects that are difficult to scan with photogrammetry: LED, metal screw and photoresistor.

To increase the likelihood of successful reconstruction of challenging surfaces, it is important to capture high quality images. The MVS algorithms are able to pick up weak textures from shadowing effects in some cases. However, higher quality images require better computer hardware for the algorithms to run properly. While the dark background helps to highlight the object, it can also hide dark objects. It is however possible to add sheets of white paper behind and underneath the model in those scenarios

Due to the limitations of the NUC computer, reconstruction has been done with another computer throughout this thesis. An Asus laptop with an i7-7700HQ CPU running at around 3.5GHz with a 16GB memory capacity has been used instead. 16GB memory has barely managed some of the models presented in this chapter, where the NUC with 8GB of RAM would certainly fail. It is possible to reduce image size (quality), but the NUC would still be considerably slower with its less powerful CPU.

3.4.3 Post Processing

MeshLab (introduced in Subsection 2.3.2) has been extensively used throughout the project as a first step after a successful reconstruction. Being open source, having a GUI and supporting many file formats makes it a superb software for point-cloud and mesh editing.

3.4.3.1 Scale, Selection and Export

After a successful 3D scan, the PLY models can be opened in MeshLab. Usually the first step is to use the measuring tool to measure the distance between two points on the model

that is known beforehand (or measured on the real model). The scale factor can then be calculated and used to generate a matrix transformation that will scale the model.

Nonessential or bad points and meshes can be selected for removal. Selection can be done in several ways, where the normal method is to use the click and drag to make a rectangular selection, or a paint brush for higher precision. Examples are shown below in Figure 32.

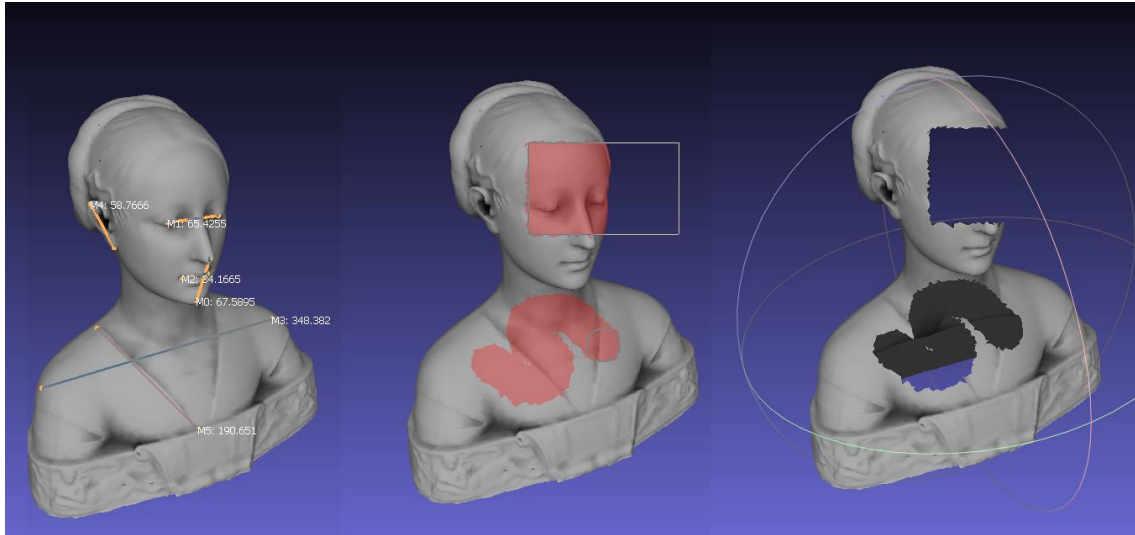


Figure 32: Measuring and selection examples using MeshLab. The model is provided by MeshLab.

Another useful method for selecting specific points or meshes, is to apply the conditional face and vertex selection tool. It is then possible to for example select specific colors on the model automatically. This is useful for simplifying the process of removing unwanted features in the final model, by for example using same colored markers and support. Usually, the texture must first be transformed to vertex color before applying color (RGB value) conditions.

The model can be exported to many different formats through MeshLab, such as STL, OBJ and VRML. STL is supported by most CAD software and often used for 3D printing.

3.4.3.2 Smoothing

MeshLab has tools for refining rough meshes. With the Quadric Edge Collapse Decimation tool, it is possible to specify the desired amount of faces for the model, and it will automatically reduce to the specified number of faces. Laplacian Smooth calculates the average position between vertices and applies smoothing at a specified number of steps. Smoothing can also be done with Siemens NX (CAD-software), with comparable results.

3.4.3.3 *Combining Models*

Adding (gluing) models together can be done with MeshLab, as was done for the otolith model discussed in Subsection 3.4.2.5. MeshLab provides an align tool with an intuitive visual interface for aligning and gluing models. This tool is especially useful for combining incomplete parts of the same object or scene together.

3.4.3.4 *Convergent Modelling*

Siemens NX 11 have introduced Convergent Modelling, which allows designers to combine solids, facets and surfaces in one model. This is very useful when adding features or modifying 3D scans, since the scanned models consists of facets and common CAD procedures use solid modelling. Examples are shown in the next subsection, where practical examples are presented.

3.4.4 Practical Examples

Due to some of the limitations of photogrammetry concerning texture, modelling clay is particularly suited for 3D scanning. Modelling clay can easily be molded to make complex shapes that are very hard and time-consuming with conventional CAD methods. Mixing different colored clay can also improve the end result, as mentioned in Subsection 3.4.2.3.

An example use case of 3D scanning with Protoboost v2 is shown in Figure 33 and Figure 34 below, where the fidelity of a prototype is increased through simple steps. In the example, a custom fitted handgrip is made. It was made for experimental purposes but could for example be used as a ski pole or camera grip. The complex shapes are formed simply by squeezing the clay with the hand it is intended for. These shapes can be very difficult to make with CAD tools. In this particular example, the clay model was created and scanned in less than 5 minutes and reconstruction finished after 7 minutes and 53 seconds. Although markers were used in the shown example, the clay model was also scanned without markers (the middle part of Figure 35 below), enabling it to be 3D printed directly without manual modifications (except changing scale).

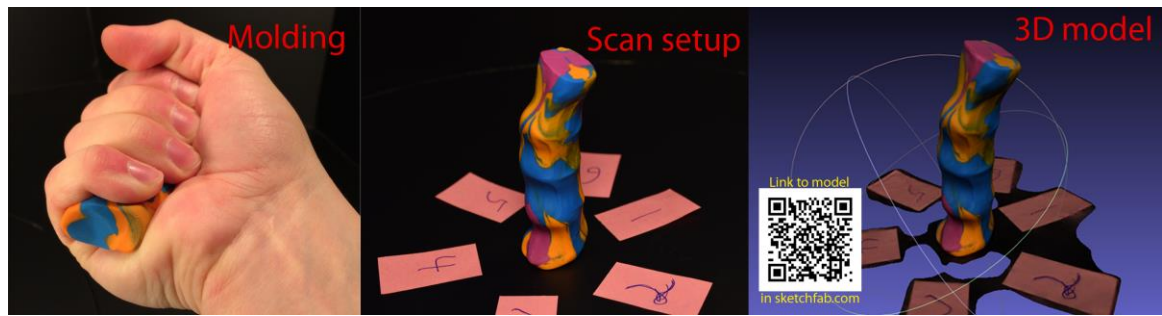


Figure 33: Handmade clay model to scanned and digitalized 3D model. QR-link: <https://skfb.ly/6yZW8>.

Post processing of the model is shown in Figure 34. It demonstrates how complex 3D scanned parts can be further modified and developed. Convergent Modelling with NX 11 is used to add thickness to the surfaces and adding threads for a potential camera attachment. It is also possible to split the model, to for example 3D print a case so that components can be put inside the model.

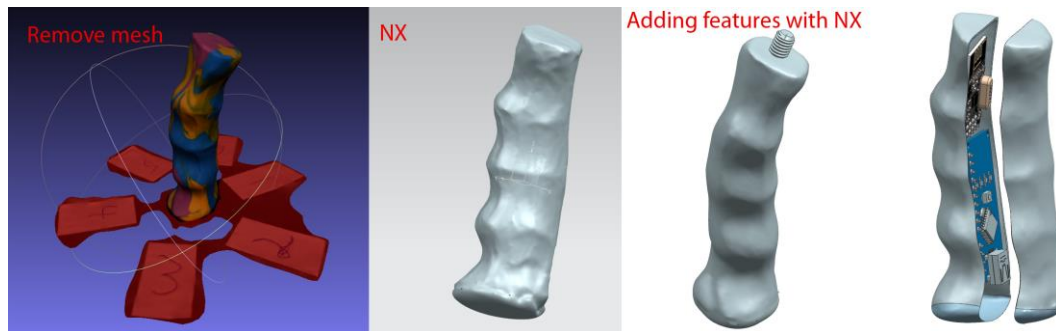


Figure 34: Modifying the original 3D scanned model. From left: mesh selected for removal, scaled STL model imported in NX, smoothing and other features added to the model. In the last image the model is split in half and assembled with an Arduino Micro and a radio transceiver, for demonstrational purposes.

A different example case was demonstrated in the Appended Conference Paper, where a clay model was scanned and 3D printed to make a detachable attachment for a Raspberry Pi to a camera. Other models scanned with the proposed method are shown in Figure 35, and can be found by following the provided link.

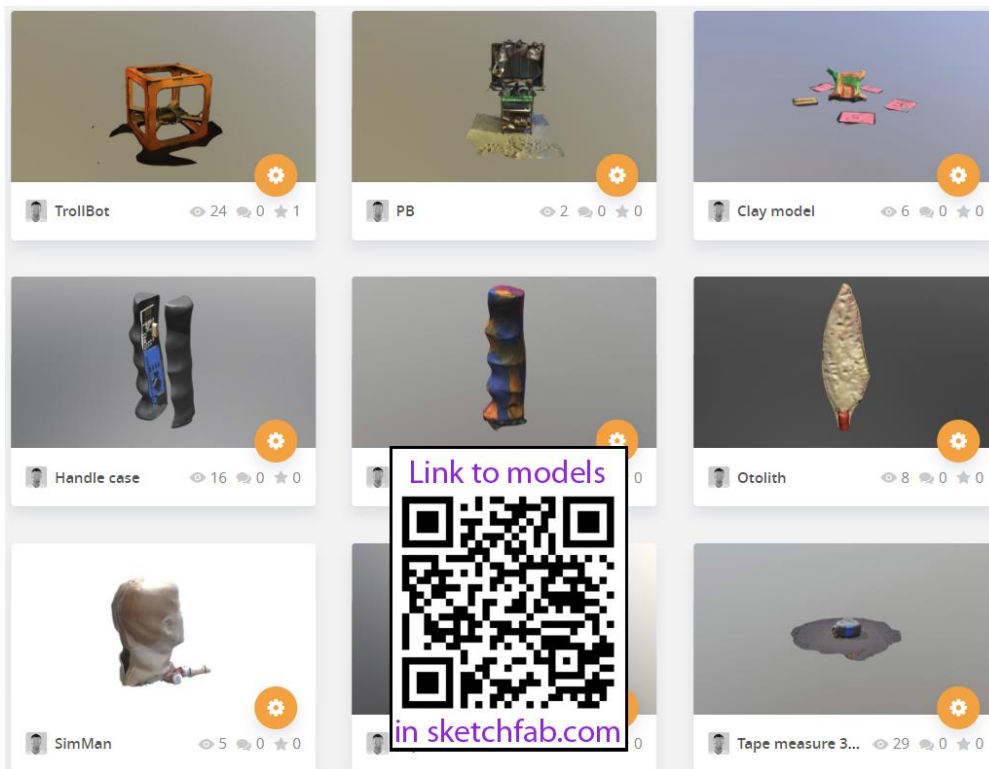


Figure 35: 3D scanned models using COLMAP and OpenMVS. QR-link: <https://sketchfab.com/sampsamik/models>.

3.5 Object Recognition

3.5.1 Introduction

Object recognition modules through OpenCV were introduced in the pre-master's thesis. Small tests were conducted for face detection using Haar feature-based cascade classifiers (Viola & Jones, 2001), and image classification using GoogLeNet (Szegedy et al., 2015), a 22 layer network trained with a deep learning framework called Caffe (Jia et al., 2014). In this chapter, another framework is introduced and used further for testing and exploring possible applications for object recognition in early stage PD.

The framework used in this thesis is called Darknet (not to be confused with the dark web that exists on darknets, which is a hidden part of the internet), which is an open source neural network framework by Redmon (2016), written in C and CUDA. Redmon and Farhadi (2016) introduced a state-of-the-art object detection system that can detect over 9000 different objects, which they call YOLO9000 (You Only Look Once). They have recently improved it with YOLOv3, as explained in their eccentric paper (Redmon & Farhadi, 2018). Their method can detect objects in real time with high accuracy and speed, with up to 78.6mAP at 40fps with YOLOv2 on Pascal VOC2007 (dataset consisting of

annotated consumer photographs), and 73.4mAP on VOC2012. These performances match other state-of-the-art detectors such as Faster R-CNN and SSD512 but is faster. Darknet uses a mechanism for training both classification and detection data, providing neural networks that can classify objects and predict their bounding box coordinates. The terms classifier and weight-file are used interchangeably in the remainder of this chapter, referring to a trained model (neural network) that can be used for object recognition.

A forked GitHub Repository made by AlexeyAB (2018), for using darknet with Windows and Linux, has been used. In this chapter, general usage of the framework along with practical tests, experiments and examples are presented, resulting in a discussion on possible applications of this method within the context of PD and prototyping activities.

3.5.2 Using YOLO with Darknet

After installing Darknet, using it is straight forward. Darknet comes with command line applications for automatically recognizing objects in images, videos and in real-time from e.g. a webcam. Pre-trained models (weight-files) can be downloaded from Redmon (2016). To use the model, a configure file (.cfg) is also needed, which includes settings such as detection resolution and learning rates.

An example is shown in Figure 36 below. “yolo.weight” and “yolo.cfg” was used for this example, which are trained with the COCO dataset (Lin et al., 2014) with 80 object categories. A computer with NVIDIA GeForce GTX 1050 GPU was used, enabling the use of CUDA for GPU-accelerated computing. The prediction completed after 0.073 seconds. Running the exact same program with the same picture on the NUC (Linux without GPU computing) resulted in a similar prediction but used 15 seconds. Consequently, using GPU-accelerated computing is highly beneficial, in this case being roughly 200 times faster.

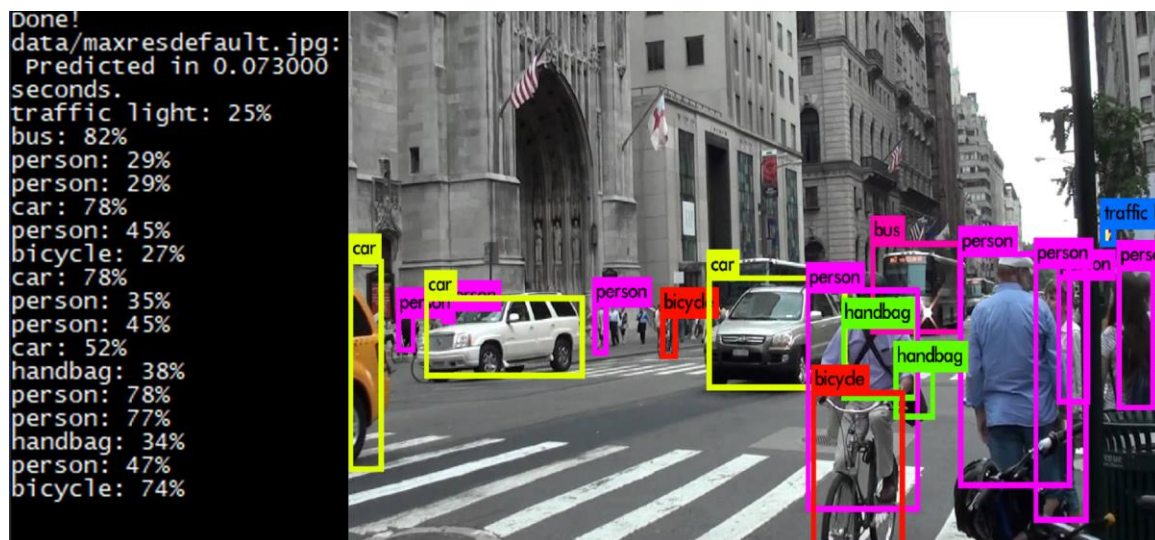


Figure 36: Object recognition test with YOLO, on Windows 10 with CUDA. Predictions and detections with bounding boxes are shown to the right and confidence values on the left.

3.5.3 Training a Neural Network with Darknet

Making custom classifiers (i.e. weight-files) for use with Darknet is done through supervised learning. The approach used for training with Darknet, including common practices, is provided by AlexeyAB (2018) and Redmon (2016).

Images with the objects to be trained must be labeled (input-output examples). Each image needs a corresponding text file, containing classes (object names identified by a number) and their relative positions in the image. An image with an object covering the left half would have 0.25 and 0.5 as x and y values (object center), with 0.5 width and 1 in height, with the origin at the top left corner.

Labelling data has been done using “Yolo_mark” by AlexeyAB (2018). The labelling tool provides a GUI for loading images and manually creating boxes around objects with corresponding classes. The process is shown in Figure 37, where an object has been marked and assigned a class ID. After an image is marked in the program, a text file is automatically generated with the values mentioned in the previous paragraph.

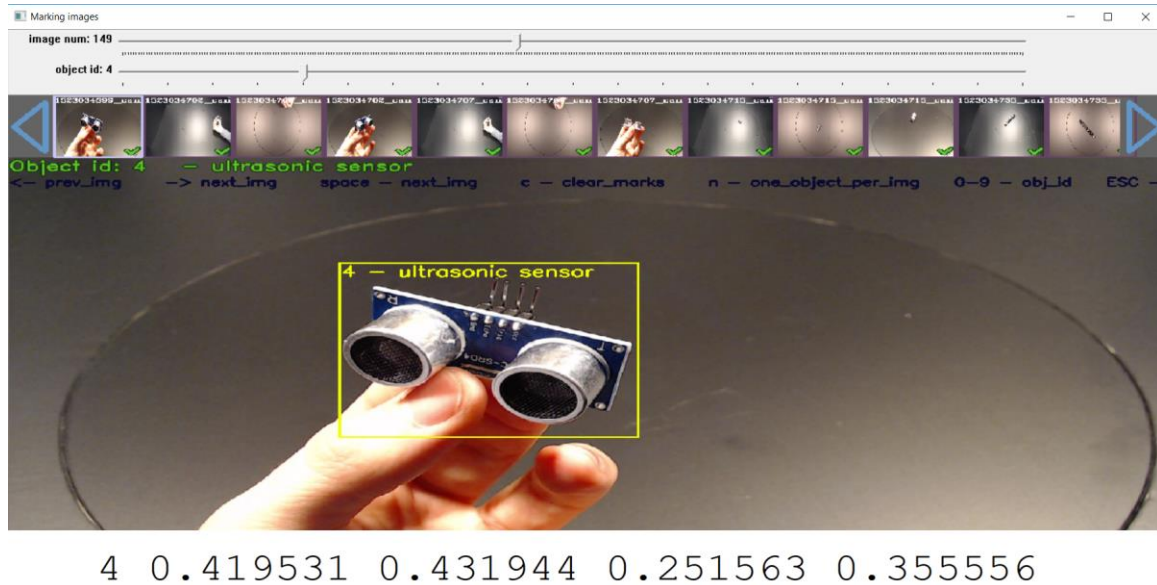


Figure 37: Example using "Yolo_mark" for manually labelling training data. The content of the corresponding auto-generated text file is shown below the image, with object/class ID, object center (x and y value), width and height, relative to image size.

After producing a set of labeled data (it is also beneficial for the set to contain non-labeled data, i.e. negative samples), they need to be divided into train and validation sets. While the training set is used to fit the parameters of the classifier, the validation set is used to tune the parameters. No definitive ratio between train and validation data is found, but an 80/20 ratio (80% train and 20% validation) is commonly used.

It is good practice to use pre-trained weights for the convolutional layers, which can be used as baselines during training and be further improved for the custom dataset. They contain layers that already can process colors, shapes and other features useful for many pattern recognition applications. They have been used in this thesis, and are provided by Redmon (2016).

Different information is displayed while training the data with Darknet. The most important to note is the average loss (explained in Subsection 2.3.3). Training can be stopped when the loss stops decreasing for several iterations. Weight-files are saved every 100 iterations. A rule of thumb is that 2000 iterations for each class (object) is enough. However, the last iterations do not always produce the most accurate classifiers, due to overfitting (see Subsection 2.3.3). When training is complete, some of the latest saved weight-files should be reviewed to find the one with the highest IoU. Given a weight-file, Darknet can automatically calculate the IoU for every validation image provided.

It is highly recommended to use GPU-accelerated computing for training, resulting in training lasting for hours instead of days. When a new weight-file is created it can be used similarly to the example shown in Subsection 3.5.2, for detecting custom objects.

3.5.4 Experimenting with Object Recognition

To test if object recognition can be applied to augment PD activities (and potentially aid research on prototyping activities), two main functions were explored. The first function relates to physical PD activities in a laboratory and workshop setting, where designers spend their time on building prototypes and testing ideas. In this specific case, TrollLabs is used as an example.

There are many tools, mechatronics parts and sensors in the lab, used by design and mechanical engineering practitioners ranging from students to PhDs. Less experienced users of the lab might not recognize all the available components. Keeping track of how all the available tools and components (e.g. sensors) work is close to impossible even for experienced users. The common approach to learn (or re-learn) how to use these components is asking other users of the lab for help, reading books or using a personal computer with internet. The process can be cumbersome and slow, especially if the name of the component is unknown. To test if improvements can be made to the process of obtaining knowledge in this context, the following component recognition hypothesis was formed:

Object recognition can be used to recognize specific components commonly used in a prototyping laboratory, to provide information about the detected components fast and reliably.

Another relevant feature to detect is materials, especially for researching output from PD activities as in Sjöman et al. (2017) by detecting materials used in prototypes. The following material detection hypothesis was formed:

Object recognition can be used to detect specific materials in images.

Experiments have been conducted by training neural networks using Darknet and CUDA, with data captured by both Protoboost v2 and the original Protoboost by Sjöman et al. (2017), and testing the trained models on new data. The experiments and results will be detailed in the following subsections.

3.5.5 Detecting Materials Experiment

To test the material detection hypothesis, a commonly used material for making prototypes at TrollLabs, Medium Density Fiberboard (MDF), was used. The main elements of the experiment consist of capturing training data, training a classifier, testing the classifier on other (unbiased) data and discussing the results.

3.5.5.1 Training

Images of prototypes containing MDF, captured by the original Protoboost, were used as training data. A total of 26 images of 4 different prototypes were used, some of which are depicted in Figure 38. They were labelled and trained as described in Subsection 3.5.3.

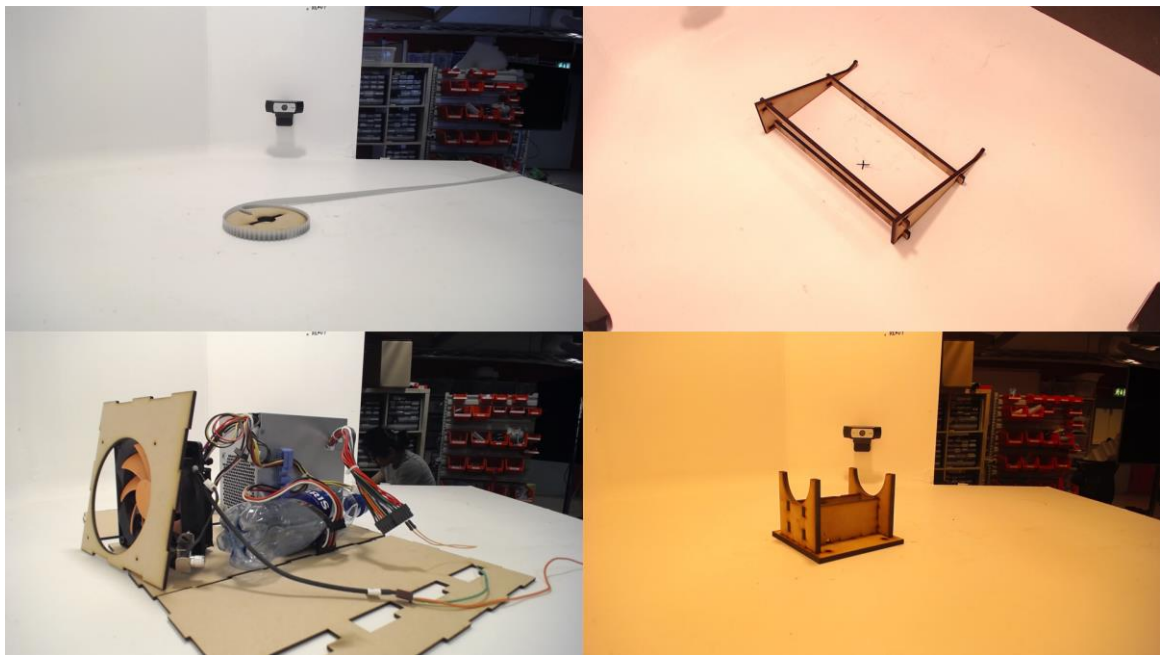


Figure 38: Some of the images used for training a classifier to recognize medium density fiberboard. Images are captured with the original Protoboost.

3.5.5.2 Results

After 2000 iterations, the average loss did not surpass 0.13. The classifier was trained for a total of 1 hour (33.3 iterations per minute). The last weight-file saved had an average IoU of 56.73%. A closer inspection of the saved weight-files showed that a classifier at 1500 iterations had the highest average IoU, at 60.46%, which was used for testing the hypothesis. Some of the classifiers (weight-files) along with their training and configuration data can be found in Appendix A-5.

3 Development of Protobooth v2 - Object Recognition

The classifier was tested on 14 images of randomly picked MDF residues from laser cutting, captured with the original Protobooth. The classifier correctly identified MDF in 13/14 images, with an average confidence of 61%.

The classifier was also tested on 30 images of prototypes made by users at TrollLabs. 18/30 images were correctly classified. 10 of the images contained MDF, in which the classifier detected 7/10 correctly. 20/30 images did not have MDF, where the classifier wrongfully detected 8/20 images to contain MDF. A summary of both tests is shown below in Table 4, and some of the visual results are provided in Figure 39.

Table 4: MDF classifier test results summary.

Image set (number of images)	Images with MDF	Correct MDF detections	Average confidence for correct detection [%]	Wrong MDF detections	Average confidence for wrong detection [%]
MDF residues (14)	14	13	61	0	-
Prototypes (30)	10	7	46.1	8	44.3

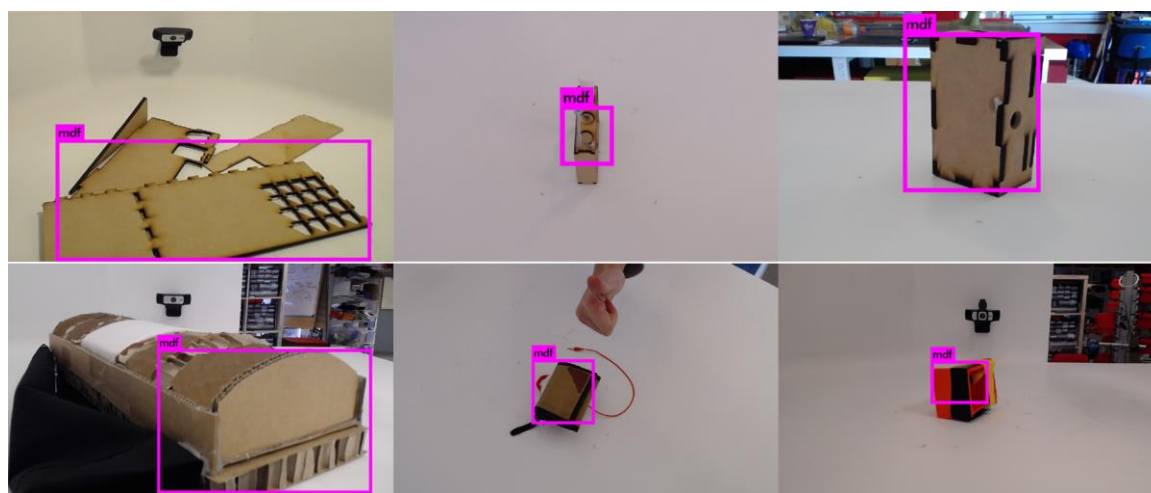


Figure 39: Some of the results using the classifier trained to detect MDF. The images at the top show correct detections and the bottom show wrong detections containing cardboard, a wallet and plastic box.

3.5.5.3 Discussion

The experiment is limited, as only one material is tested, and with a small set of training data. However, the data show a potential to detect and recognize materials, in this case only MDF. More data and experimentation are required for a more reasonable evaluation of the

material detection hypothesis: “Object recognition can be used to detect specific materials in images”.

Wrong predictions have been made on objects with similar color and shapes to MDF, e.g. cardboard and boxes, as demonstrated in Figure 39. The confidence for wrong detections are close to the same as confidences for correct predictions, so a higher detection threshold is not applicable in this scenario. Potential solutions will be discussed further in Object Recognition Implications and Solutions.

3.5.6 Detecting Components Experiment

Protoboath v2 was used to test the component recognition hypothesis (as defined in Subsection 3.5.4). Five objects were selected to represent common objects used in a prototyping laboratory: Arduino Uno, load-cell, solder sucker, HX711 (breakout board) and ultrasonic sensor. The same approach was used as in the previous experiment, testing was however done by using the classifier on real-time video on Linux (through Protoboath v2) without GPU-accelerated computing. Accuracy and reliability is assessed empirically by viewing detections in the live video.

3.5.6.1 Training

The objects trained for the experiment are shown in the below figure, and the proportion of images used are presented in Table 5. The dataset can be found in Appendix A-5. The images were captured in Protoboath v2, using 3 Logitech webcams simultaneously to capture the objects from different viewpoints.

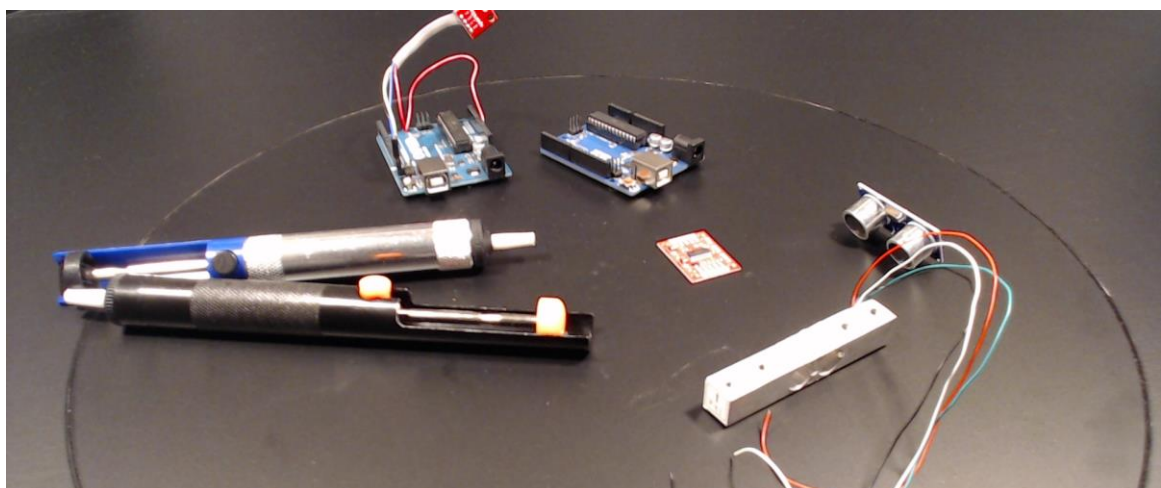


Figure 40: Objects that are used for training a classifier: Solder sucker, Arduino Uno, HX711 breakout board, load cell and ultrasonic sensor.

3 Development of Protobooth v2 - Object Recognition

Table 5: Images captured for each class (object). Some images contain multiple objects.

Images captured of each class						
Arduino Uno	Load-cell	Solder sucker	HX711	Ultrasonic sensor	Negative samples	Total images
90	88	97	125	84	66	342

3.5.6.2 Results

Average loss did not surpass 0.08 towards 14600 iterations. Training lasted for 8 hours and 23 minutes (29 iterations per minute). The last weight-file had the highest average IoU of 76.74%.

The objects were put in front of the camera to observe the detections made by the classifier (see Figure 41). Detections were made approximately every 315 milliseconds (3.17fps). Every object was detected at some point, but in some cases wrongly classified (see Figure 42). The test videos were recorded and uploaded to YouTube, which can be viewed by following the links in the figures below.

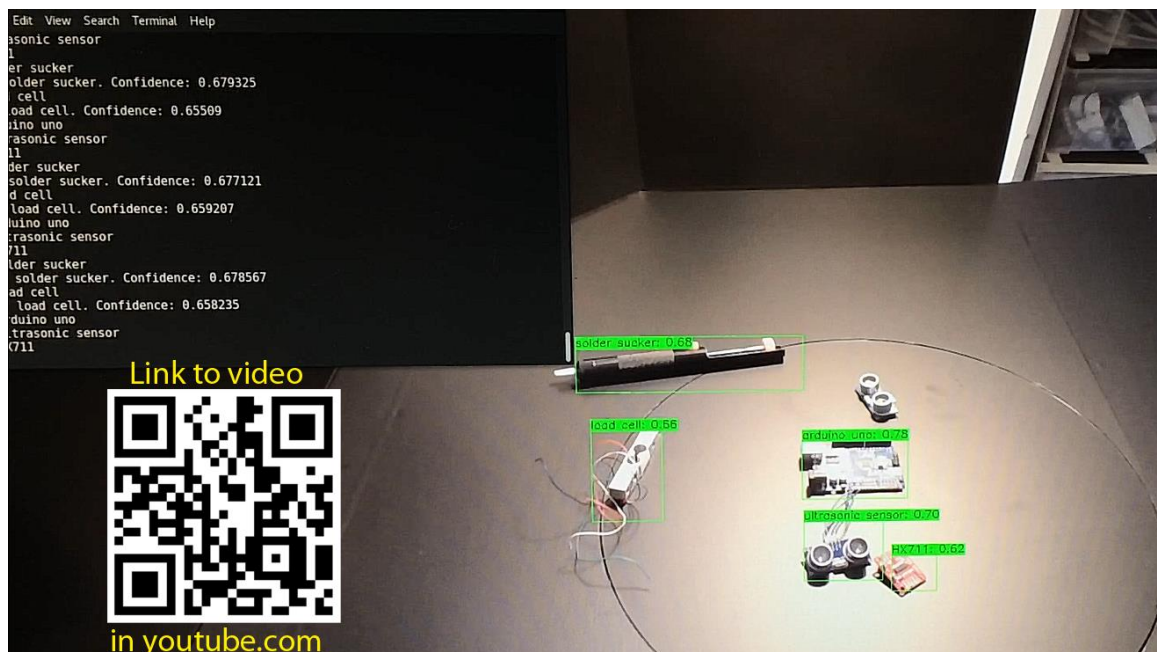


Figure 41: Successful real-time detections of each of the five objects. QR-link: <https://youtu.be/fFkLbGOM1QA>.

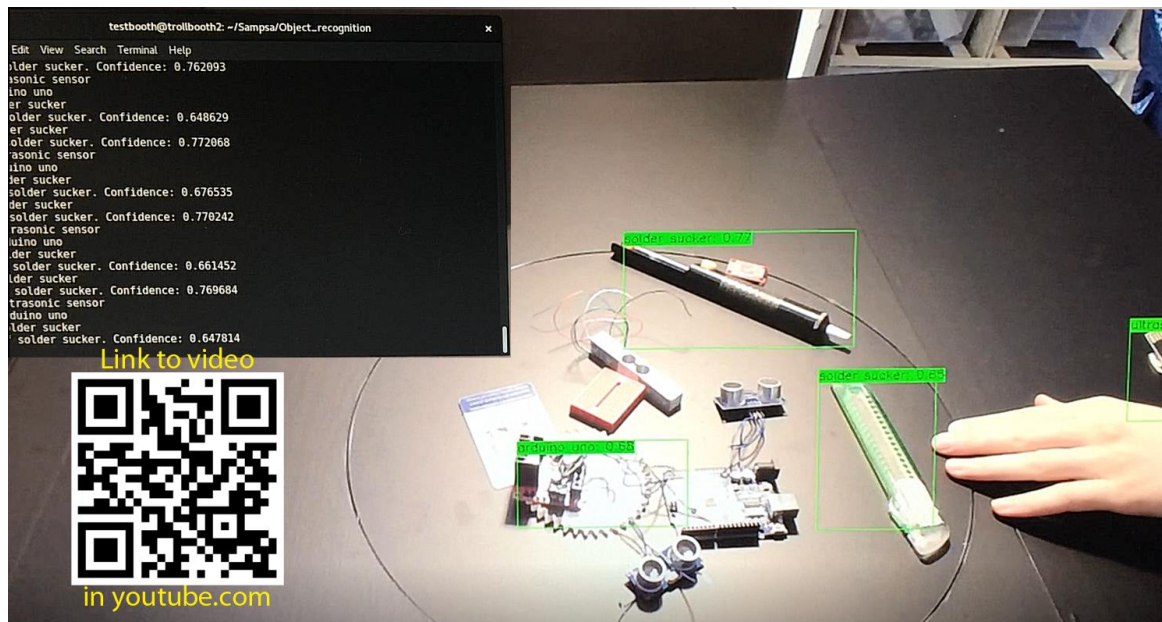


Figure 42: Real-time detections with inaccurate classifications, from left: motor drive module detected as an Arduino Uno, a utility knife as a solder sucker and keys as an ultrasonic sensor. QR-link: https://youtu.be/4CCUZ17RP_M.

3.5.6.3 Discussion

Based on an empirical assessment on using the classifier, the initial part of the component recognition hypothesis, “*Object recognition can be used to recognize specific components commonly used in a prototyping laboratory...*”, is accepted as plausible. The main reason for accepting the hypothesis is grounded in the fact that by showing each component clearly to the camera, the object will eventually be detected and classified correctly. This is not unlike how a human would approach the problem of recognizing an object, where a closer inspection will often yield a better understanding. The detections may also have been affected by the lighting conditions, which was not adapted for best possible detections.

Using the classifier for practical applications will be further demonstrated in the next subsection, supporting the latter part of the component recognition hypothesis: “... *to provide information about the detected components fast and reliably*”.

3.5.7 Practical Examples

To make use of object recognition for augmenting PD activities in a workshop or laboratory setting (e.g. TrollLabs), a program was made to detect objects and provide information about the detected objects. It is then possible for users to simply show an object to Protobooth v2, and it will automatically provide useful information about the object. The classifier for detecting 5 objects, explained in the previous subsections, was used in this example.

3 Development of Protobooth v2 - Object Recognition

The program (Appendix A-5) uses OpenCV to read from Darknet with the trained weight-file (using OpenCV's deep neural network module) and to display data in the form of informative pictures. A list of detected objects is displayed on a window in real time, where objects can be selected to get more information. If for example a load cell is shown to Protobooth v2, it will appear in the list of currently detected objects, and general information on how it works can be displayed along with wiring diagrams for connecting to an Arduino and a calibration code that can be used. The different displays for the load cell is shown in Figure 43 below, with a link to a YouTube video demonstrating this feature for an ultrasonic sensor, including showing an example of a prototype that has used the sensor before. If a solder sucker is shown to Protobooth v2, a video is presented showing how to properly use the tool.

HX711
arduino uno
solder sucker
load cell
ultra...

Wiring diagram showing HX711 module connected to Arduino Uno. Pin connections: +EXCITATION (RED) to A, -SIGNAL (WHITE) to B, -EXCITATION (BLACK) to D.

Load cell calibration code

```
#include "HX711.h"
#include <Wire.h>
#define CLKA 2
#define DOUTA 3

float calibration_factor = 83.780; //Calibrate and find correct factor
HX711 scale(DOUTA, CLKA);

void setup() {
  Serial.begin(9600);
  scale.set_scale(calibration_factor);
  scale.tare(50);
}

void loop() {
  int load = scale.get_units(10); //Read average of 10 readings
  Serial.print(load); Serial.print("\t");
  Serial.print("Cal.fact.: "); Serial.println(calibration_factor, 3);

  while (Serial.available()) {
    char t = Serial.read();
    if (t == '+') calibration_factor += 0.5;
    else if (t == '-') calibration_factor -= 0.5;
    else {
      scale.tare(50);
      break;
    }
  }
  scale.set_scale(calibration_factor);
}
```

Link to video
in youtube.com

Figure 43: Informative windows provided when a load cell is detected in and by Protobooth v2. QR-link: <https://youtu.be/T-hgF6VMqiO>.

This system is still a prototype, and the displays are viewed through a screen connected to Protobooth v2. A keyboard is used to navigate through the information.

Another classifier has also been tested, trained to detect sketches (also mentioned in the Appended Conference Paper). This classifier was used when activating Protobooth v2 (scanning an RFID card), to check if a sketch was present. If a sketch was detected, the sketch to laser method (Section 3.2) was activated, otherwise the video recording program (Section 3.3) was activated. This was experimented with to test how object recognition can be used to simplify the user experience, by automatically selecting the appropriate functions to run. This system was used during a one-month period, by users of TrollLabs, and it did not fail a classification during the time.

3.5.8 Object Recognition Implications and Solutions

Some parameters can affect the accuracy of the proposed object recognition system. It is possible to adjust detection resolution in the configuration file (.cfg) to detect objects at certain sizes. Capturing higher quality images (higher resolution) will provide more details, with the cost of processing speed. Differentiating for example MDF and cardboard might be impossible in some scenarios. It is easier to detect laser cut MDF, with the distinctive dark laser-cut edges. More caution should thus be put on deciding what kind of data is trained and for what purpose. Generally, training and validating more data would most likely prevent wrong detections and overfitting, where for example a red box could be trained as a box, and a laser-cut MDF box as an MDF box, to prevent all boxes from being classified as MDF.

Both training and testing is done in the same environment (Protobooth v2), thus accuracy is lower outside these boundaries. The benefit is that accuracy is easier to achieve when used within Protobooth v2, which is also the case for the proposed system. Using the trained models outside of Protobooth v2 would require more diverse data, e.g. different backgrounds and light conditions, to increase accuracy and robustness. Protobooth v2 is consequently suited as a tool for capturing, training and using the trained models. Furthermore, labelling training data could potentially be semi-automated by detecting the objects automatically while letting users classify the objects before training.

Darknet is easily implemented with C++ and OpenCV, as demonstrated in Subsection 3.5.7, thus making programs for customizing the input and output of object recognition feasible (e.g. manipulating detected bounding box coordinates), to tailor it for different applications. It is thus possible to integrate other inputs that can reinforce object recognition, by for example using load sensors to measure the weight of objects put inside Protoboath v2. If the load sensor can distinguish between various sizes of the same type of screw for instance, then this information could be used when detecting the screw to determine its actual size. Load cells used with Protoboath v2 and their limitations are further explained in the next section (System Details).

With the detect objects and provide information method, other data can also be added, e.g. datasheets for components. Data can also be provided in different ways, e.g. sending the datasheet to the user's email, which can be acquired through scanning a personal RFID card. It is also possible to automatically upload code to a connected Arduino, for example the load cell calibration code shown in Figure 43. Detections can also be combined, e.g. detecting an Arduino and ultrasonic sensor simultaneously, thus showing how to connect them and provide example Arduino code for reading the sensor. It should thus be possible to continuously improve the database based on usage and feedback.

Furthermore, a GUI and user interaction design should be further explored and developed to utilize the proposed system, in a simple and intuitive way.

3.6 System Details

A tested method for using the proposed system and modules is described here, not including object recognition which has not been tested together with the other modules. Tests and applications of the load cells implemented with Protoboost v2 will also be discussed.

3.6.1 Using Protoboost v2

When a user wants to interact with Protoboost v2, the only required interaction is scanning a RFID card on the RFID reader (see Section 3.1). A series of nodes in Node-red is then activated, as shown in Figure 44 below. It first checks if the Nikon camera for 3D scanning is activated. If it is, then the turntable and 3D scanning pipeline is activated. Otherwise a program is executed that captures an image from the top camera and runs a classifier on it, to check if a sketch (or white piece of paper) is placed on the table. The classifier determines if either the sketch to laser or video with serial data program is executed next.

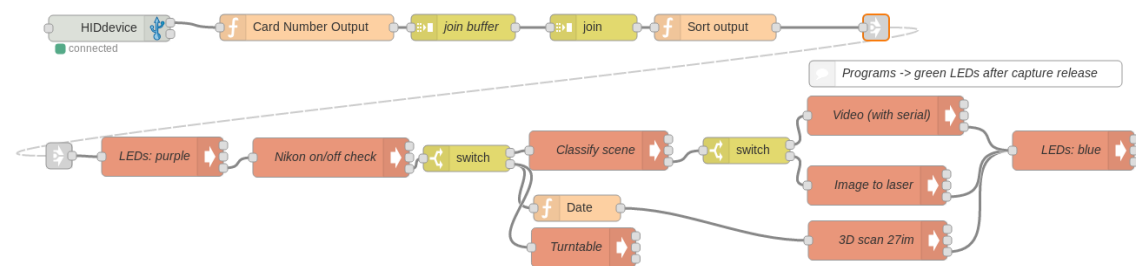


Figure 44: Node-red interface linking functions and executable programs.

For 3D scanning it is necessary to activate and position the camera manually first. LEDs are used to indicate at which stage the program is currently at, where purple means the prototype is being captured, green means capturing is complete and the user can remove the prototype, and blue meaning Protoboost v2 has processed every stage and is ready to be used again. A cross laser is used to mark the turntable center, which will automatically be turned off while capturing prototypes (or sketches).

Each program will turn the green LEDs on after having captured images or video, and the outputs (depending on which program is run: DXF vector files, PDF files, 3D point clouds and meshes, video and serial data) will be copied to a google drive folder so that the files are made available for the users. Currently, users can access the files at a computer located at TrollLabs, used with the laser cutter. The object recognition module for detecting objects

and displaying information has not been included in the current setup. It has only been tested on the side.

3.6.2 Load Cells

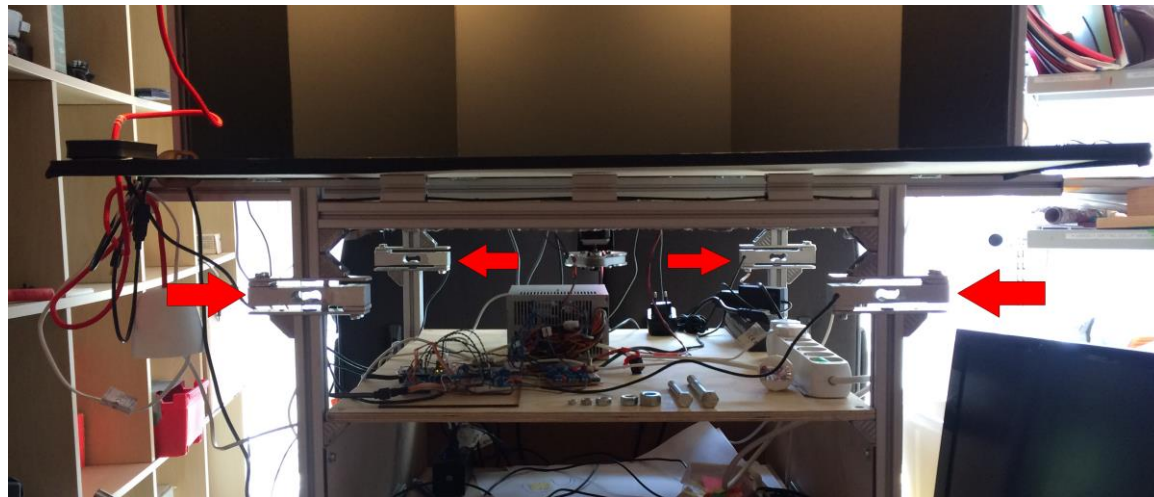


Figure 45: Protoboost v2 with arrows pointing to the load cells.

Each 4 legs of Protoboost v2 is connected to a 50kg load cell, shown in Figure 45 above, to enable weight measurements of objects put into the booth. Each load cell is connected to an amplifier (HX711) and read individually by an Arduino Mega.

A few tests were conducted in order to determine how robust (change in values over time) and accurate (small loads) the load cells are with the current setup. The tests and results will be discussed in the following subsections, followed by Weight Measuring Applications and Limitations.

3.6.2.1 Robustness

Weight was measured over a period of 4.8 hours without any added load. Results are provided in the figure and table below. 18 outliers were removed from the data, in which 16 outliers was in the range -1630 to -17109, and two above 4600, resulting most likely from random noise. Linear regression is used for simplicity, as illustrated in Figure 46. Nonlinearity can be observed in the data, but considering the long duration, linearity is accurate at certain intervals.

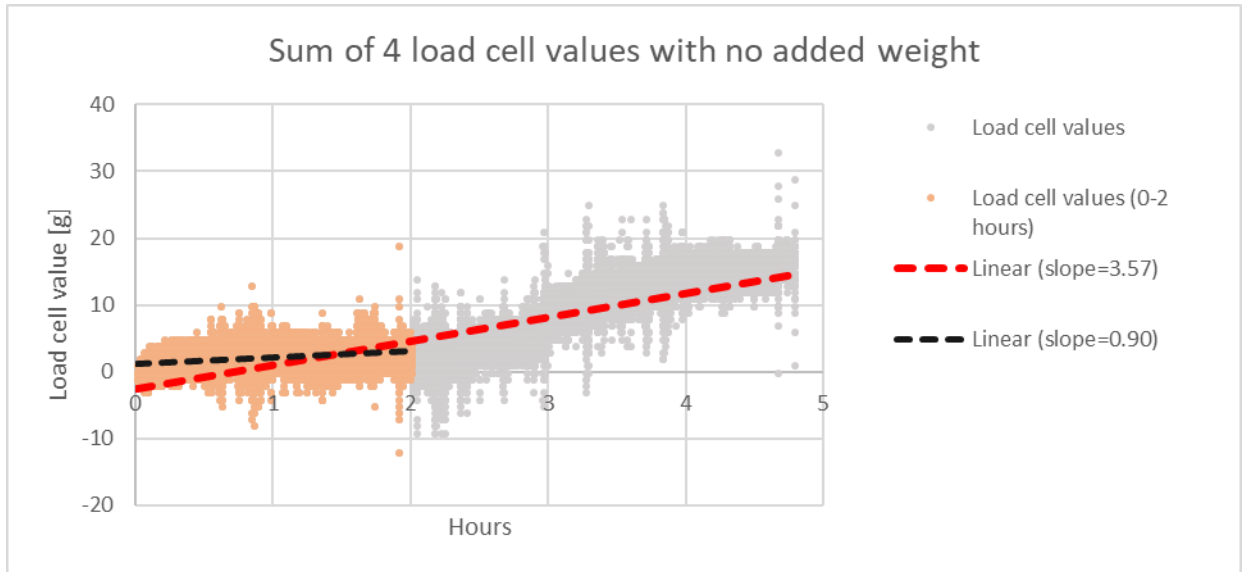


Figure 46: Protoboost v2 load cell values during a period of 4.8 hours without added weight.

Looking at the table below, deviation and the rate of change (grams/hour) are relatively low for the first 2 hours and towards the end, compared to the more drastic change in the middle 2-4 hours.

Table 6: Load cell data with no added weight.

<i>Duration</i> <i>[hours]</i>	<i>#</i> <i>values</i>	<i>Average</i> <i>[g]</i>	<i>SD</i> <i>[g]</i>	<i>Max</i> <i>[g]</i>	<i>Min</i> <i>[g]</i>	<i>Load-rate</i> <i>[grams/hour]</i>
0-4.8	188120	5.27	5.50	33	-12	3.57
0-2	78528	2.17	1.44	19	-12	0.90
2-4	78716	7.81	4.55	25	-9	7.34
4-4.8	30876	15.01	1.31	33	0	1.79

3.6.2.2 Accuracy

To test the accuracy of the load cells, 2 different sized bolts (s: small and l: large) and 5 nuts (M6, M8, M10, M12 and M16) were measured with Protoboost v2 and compared with a commercial scale with a resolution of 1 gram. Each object was measured in sequence,

3 Development of Protoboost v2 - System Details

from smallest to largest, for 30 seconds with 30 seconds of no load in between. Plot and table is shown below.

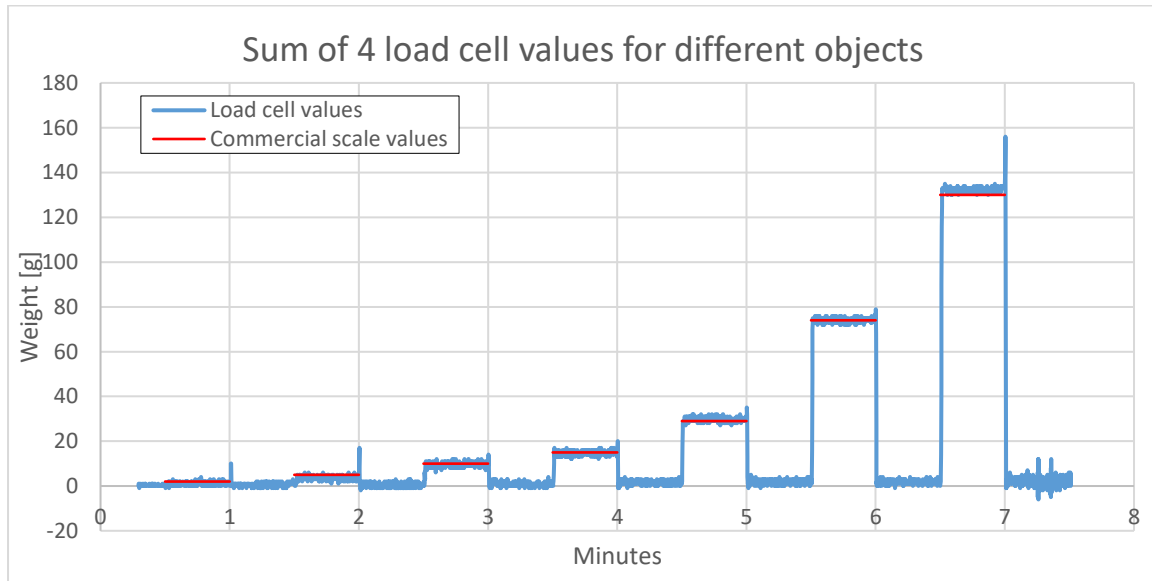


Figure 47: Plot of commercial scale and load cell values, from lightest to heaviest object (M6, M8, M10, M12 and M16 nuts, and a small and large bolt).

Table 7: Weight values from a commercial scale and load cells.

<i>Object</i>	<i>Commercial scale [g]</i>	<i>Load cells average [g]</i>	<i>SD load cells [g]</i>	<i>In between load cell average [g]</i>
<i>M6 nut</i>	1*	1.03	0.82	0.14
<i>M8 nut</i>	5	3.59	0.87	0.43
<i>M10 nut</i>	10	9.55	1.03	0.91
<i>M12 nut</i>	15	14.64	1.04	1.47
<i>M16 nut</i>	29	29.65	1.04	1.79
<i>Bolt s</i>	74	74.15	0.96	0.68
<i>Bolt l</i>	130	132.43	0.92	2.08

* value fluctuated between 0 and 2

Rounding the load cell values in Table 7 to the nearest gram, would results in 4 out of 7 measurements being identical to the commercial scale. There is also a tendency for the load cell values to remain at higher values after removing larger weights. Standard deviation is close to 1g for each measurement.

3.6.2.3 Weight Measuring Applications and Limitations

It is shown by the plots in Figure 46 that the load cells will drift towards higher values at different rates. Based on the values presented in Table 6, the smallest changes occur during the first 2 hours. Thus, resetting the load cells within every 2 hours is advisable, or always compare values before and after adding a load. To increase stability, within the 2 initial hours (SD = 1.44g) and in Table 7 (SD \approx 0.95g), the average of several measurements should be calculated, in addition to removing potential outliers. Small values from the load cell setup is comparable to commercial scales, as shown in Figure 47 and Table 7.

The change and variation in values could be affected by change in temperature. They are additionally more likely affected by the stress in the load cells due to the stiff connections to the structure of Protoboost v2. Stresses can occur in the load cells when Protoboost v2 is moved slightly, which might be the reason for an increase in values after applying higher loads, as shown in the last column of Table 7. Ideally, they should be attached by some combination of roller and ball joints at the connection between the aluminum bracket and Rexroth profile to reduce stress and momentum.

The accuracy and robustness is acceptable for simple load cases, but more testing should be done at higher loads and with filtering methods to get the most out of the load cell setup. A proper calibration technique should also be applied. To answer the proposition made previously in Object Recognition Implications and Solutions: with the possibility to distinguish load values within roughly ± 1 -2g it is for example be possible to detect nuts and their size with object recognition reinforced by measuring the weight (currently sizes of M6 and higher). An M6 and M8 nut could thus be detected with object recognition and differentiated when including weight measurements from the load cells. It might be possible to detect materials with object recognition reinforced by the weight, e.g. to distinguish cardboard from MDF.

This page is intentionally left blank.

4 Protobooth v2 Prospects

Protobooth v2 is essentially an instrument that product developers can interact with to make physical parts, share knowledge by demonstrating their creations and to learn quickly from the information it can provide by recognizing components (and maybe materials). It is a system that introduces effortless and fast methods for generating designs, documenting microcontroller prototypes and learning during PD projects, in new ways that are uncommon or difficult in traditional PD and tools. Having presented the different modules and functions, this chapter attempts to connect the system as whole, to discuss and hypothesize the impact (and improvements) it can have on early stage PD. Suggestions on how these hypotheses can be tested through experiments will be included, along with predicting the outcome of such experiments, ending with a discussion on potential negative aspects of the proposed system. These experiments are provided as suggestions for future work and have not been conducted due to time limitations for this thesis.

Arguments and discussions for using Protobooth v2 as opposed to other available methods (e.g. commercial 3D scanners) are provided in the Appended Conference Paper. Parts of the paper will also be used in this chapter, to explain how the proposed system can augment physical prototype activities in early stage product development.

4.1 Hypotheses, Experiments and Predictions

The main goal of Protobooth v2 and its four main modules is to augment physical prototyping activities by simplifying and speeding up the process, ultimately allowing the generation of more design iterations in the early pre-requirement stages of product development and design. As demonstrated in Subsection 3.6.1, Protobooth v2 is simple to use. Both experienced and inexperienced product developers can thus benefit from this system, to accelerate learning and ultimately make better products faster.

4.1.1 Simplicity Hypothesis

Many users (e.g. engineers, designers, creators and students from PD companies and universities etc.) aiming to realize products (and prototypes) often have a certain set of tools that they are familiarized with and frequently use. The same users often have a broader spectrum of tools and equipment that could be used, yet they tend to stick with methods and machinery that are familiar. A real example of such under-utilization is users 3D-printing small boxes for various microcontrollers (Arduino Uno, Raspberry Pi, etc.), and using such simple

geometry (e.g. rectangles) that a laser cutter would produce more durable and more accurate models in less time than a 3D printer. Therefore, by lowering the threshold of learning new equipment and machinery for realizing new products and prototypes will also make the users better equipped to solving their various tasks.

It is thus hypothesized that Protobooth v2 can simplify the transition from low resolution prototyping to higher fidelity prototypes, as such increase the productivity of developers, giving them a simpler way to interact with production machines such as laser cutters and 3D printers. They are then able to learn faster and make better designs and products.

This could be tested in practical PD projects at schools or universities, by providing and logging the usage of Protobooth v2, and compare end results based on the system usage. If for example one group use Protobooth v2 allot, and as a result creates more prototypes, the technical solution and novelty of the end result could be compared with other groups that did not utilize the system.

Using Protobooth v2 is predicted to allow more designs to be created, resulting in more prototypes, thus more knowledge generated and ultimately better end products.

4.1.2 Complexity and Speed Hypothesis

The development of CAD has enhanced PD in many ways, such as decreasing cost and increasing speed of production. However, in the early stages, when a large solution space is explored, CAD and its fixed architecture limits changes in design and ideation (Leifer & Steinert, 2011). It is a sophisticated tool that require training to be used properly and efficiently. When geometries and shapes become organic and complex, it can be very time consuming to make the models through software. Making complex models in the real world, with your hands and by using other objects to form and generate the desired shape, can be done faster in addition to being more easily tested in practice. One of the advantages with CAD is communication and documentation. With the proposed 3D scanning module, similar communication and documentation possibilities can be achieved, by digitalizing (3D scanning) physical prototypes. This can also be applied in virtual reality systems in the future, e.g. to collaborate with physical pre-requirement prototypes over large distances through augmented reality sessions, or letting customers experience the prototypes remotely and in large quantities. CAD is commonly used in the early stages of PD to enable the use of 3D printing. A simple way to 3D print physical models can be achieved with the proposed 3D scanning system, by for example making and scanning clay models, to reduce the dependency of CAD.

Thus, it is hypothesized that helping developers realize their ideas, by producing prototypes, will ultimately result in better solutions (and in less time). By contributing to more design alternatives, through helping design iterations emerge simpler and quicker, a single individual or a team of developers can gain more knowledge with less time, especially when making complex 2D (laser cut) and 3D (printed) parts. If this is the result of implementing and using the methods and system presented in this thesis, then the goal of augmenting physical prototype activities in early stage product development is successfully achieved.

A controlled experiment where two groups of participants either use Protobooth v2 or CAD to create a design that must fit with a complex shape, and another two groups with the same approach but creates an uncomplicated design, could be done to compare time and end result. This can be done for both the sketch to laser method and 3D scanning.

For Protobooth v2 to achieve its goal, participants using it should be able to create the complex part faster and more accurately than participants using CAD. The uncomplicated part could potentially be faster to make with Protobooth v2, although perhaps slower to reconstruct the digital model if used with 3D models, but less accurate than the CAD. There might of course be a certain level of complexity and simplicity where both methods are equal, so a careful consideration of design task should be made before such an experiment.

4.1.3 Learning Hypothesis

Being competitive in new PD requires being up to date on current and past possibilities and limitations of design. Efficiency in utilizing the available knowledge and technologies when producing prototypes is paramount for staying with or ahead the status quo. Knowledge can nowadays be accessed anywhere at any time, with endless resources on the internet, but can as a result be overwhelming. It can occasionally be difficult to find the exact problem you are looking for, and it often requires considerable time and effort to acquire the knowledge while at the same time trying to build prototypes. Using the proposed object recognition method, to simply, quickly and reliably get information about various, available prototyping components without having to shift too much focus from the physical development activities, can reduce the knowledge acquisition effort while increasing productivity.

With the proposed object recognition method for detecting components in a prototyping laboratory, it is hypothesized that automatically detecting objects and providing information about the objects is more effective and efficient than conventional methods

when developing physical prototypes, thus increasing development and learning speed while reducing the effort.

Testing the hypothesis can be done by measuring the time used by participants producing specific sensor-based prototypes, where one group has access to Protobooth v2 and the other have a computer with internet or other resources.

If all the required information to accomplish the task is provided by Protobooth v2, it should enable the task to be completed allot faster. Directly providing information about a physical component, by showing it to Protobooth v2, should be simpler than searching for keywords on the internet, and probably faster than other tools, especially when Protobooth v2 can provide various data such as example code and wiring diagrams.

4.1.4 Motivation Hypothesis

By helping less experienced product developers (i.e. early mechanical engineering students) the system might also increase motivation and confidence. “The experience of creating something that works has been shown to be a strong motivational driver.” (Slåttsveen, Steinert, & Aasland, 2016). Having more alternatives for making physical prototypes can increase learning by doing and give a confidence boost by being able to produce physical parts through simple methods.

Allowing users to interact with and use Protobooth v2 freely during PD projects should be done in order to get feedback. A questionnaire should then be created to ask how they perceived the system and if they felt more motivated and confident after using the system.

Some users might find (parts of) the system very useful for some scenarios, especially users that find CAD difficult and cumbersome.

4.1.5 Design Fixation Hypothesis

By introducing new ways of generating prototypes and documenting microcontroller-based prototypes, design fixation (explained in Subsection 2.2.2) can potentially be reduced. Making prototypes fast, and with less focus on making the design look perfect, can reduce the PD teams and individuals being too invested in their initial ideas, thus testing more ideas and learn faster.

An experiment, where a 3D printed model for a specific application is to be made, where one group use CAD and the other use clay with 3D scanning, could be done to compare the number of design iterations made and the end result (and speed).

Since making a perfect clay model is difficult, it would likely result in more designs being made quickly, compared to a refined and time-consuming CAD model. Being able to test the clay model is in itself an iteration of the prototype, enabling knowledge to be generated before 3D printing. The CAD model requires 3D printing to enable practical testing, or some form of simulation that can take a lot of time. Using clay is also cheap which can lower the effort, i.e. sunk cost as described in Subsection 2.2.2, to reduce design fixation.

Sharing a video of a prototype, while simultaneously showing how it works programmatically, can enhance communication. This will more easily allow feedback, and potentially provide faster insights for the designer from other perspectives as to how the prototype works and reacts to different stimuli.

The conceptual properties gained and learned from showing a video with serial data from a prototype versus only the prototype itself can be compared by asking several participants to analyze them. It is predicted that the participants who analyze the video with serial data will get a deeper understanding of how the prototype works compared to other methods, thus give a more accurate description of the prototype in addition to feedback on improvements etc.

4.2 Possible Drawbacks and Limitations of Protoboost v2

It is important to be aware of possible negative side effects of the proposed system, in order to reduce their impact and to improve the system in the future, or to evaluate its relevance in specific PD strategies.

Users that do not trust their drawing or molding capabilities, or have a high standard on accuracy, might spend more time on drawing or molding (3D scanning) than using CAD-software. Some users might produce worse prototypes by overestimating the system or their own drawing capabilities. The proposed system might not produce the results that is expected in every case. Unfamiliar systems can also have a high threshold for being used, and often require some form of introduction and training, regardless of how simple it might be. New machines can often be viewed as “just another tool”, and thus not be fully utilized due to the threshold of familiarizing with new equipment. The goal of making Protoboost

v2 as simple and straightforward to use as possible is aimed at reducing some of these potential drawbacks.

The early stages of PD, or wayfaring (Section 2.2.3), is highly context dependent. Protoboost v2 is subsequently not necessarily a better approach than other methods in every case. It is however an alternative that should always be available, simple to use, fast and reliable.

The proposed 3D scanning system and sketch to laser method can have difficulties in capturing very small details accurately, in addition to the limitations of texture and reflection for 3D scanning with photogrammetry. In some cases, to get the most out of the systems, the designer should be able to modify the model with CAD or mesh editing software (e.g. MeshLab). Adding details to the scanned model, such as small threads, is difficult without software. Thus, using 3D scanning can in many cases require more skills and also take longer time compared to CAD modelling.

5 Implications and Other Features

A system has been developed with methods for augmenting physical PD activities. The system is however not completely put together. This chapter highlights some of the features that is missing, and suggestions on how to make the system complete before potentially using it in PD projects. An overview of other interesting features, that possibly can be included with the system to further augment PD activities, is provided afterwards.

5.1 Protobooth v2

With all the proof-of-concepts for sketch to laser, video with serial data, 3D scanning and object recognition, some of which are tested in a working system (Subsection 3.6.1), more efforts should be done to implement the functions into a robust and user-friendly application. The system with 3D scanning, sketch to laser and video recording with serial data has been tested, but a simple way to interact with the object recognition module has not been developed. An interesting possibility is to integrate a large touch-screen to display info and allow interaction with the presented information (e.g. simple moving and zooming). The object recognition module can also be active all the time, with simple buttons to navigate through the information, while the other modules are activated whenever the RFID scanner is used, effectively interrupting and pausing the object recognition program.

The codes in Appendix A for all the different modules contains local paths and dependencies. They are consequently not suitable for direct implementation into other systems. The codes must manually be changed and compiled. This includes installing programs and modifying their paths, such as KVEC for line tracing, COLMAP and OpenMVS for 3D scanning, Darknet for object recognition, and OpenCV and gPhoto2 for controlling cameras etc.

Although examples of using each of the four main modules have been presented, there is a lack of user testing by other product developers. With the hypotheses and experiments presented in Section 4.1, more effort on testing the system in real practical cases should be done, to fully understand the impact and benefits the system can have to augment PD and to get feedback.

Some of the technical flaws in the system include the problem of detecting weak pencil lines when capturing sketches and the presence of distorted lines (Subsection 3.2.5 and

3.2.6), in addition to applying accurate timestamps when logging serial data (Subsection 3.3.5), and the sometimes wrong classifications when detecting object (Subsection 3.5.8). Most of these issues have been addressed with potential solutions, which should be considered and further tested in potential future implementations of the system.

5.2 Other Possibilities

This section presents some suggestions for additional techniques that can be interesting to implement with a potential future Protobooth v2 system.

5.2.1 Motion Magnification and Color Amplification

Subtle changes in color and motion from a video recording can be used to discover information hidden from human sight. The Eulerian Video Magnification method proposed by H.-Y. Wu et al. (2012) can for example visualize the human pulse, simply by recording a video of a person to reveal slight changes in skin color due to blood circulation. Through spatial decomposition and temporal filtering for each frame, they are able to amplify the resulting signals. This could be used to analyze prototype performance, for example to discover dynamic behavior and vibrations in an actuating prototype.

5.2.2 Speech, Handwriting and Face Recognition

Speech, handwriting and face recognition methods has become increasingly robust over the past years, especially due to large investments by large corporations (e.g. Google and Apple). Speech and face recognition can be used to interact with Protobooth v2 in simple ways, for example asking Protobooth v2 to capture an object and send information about it to the email of the recognized person. Speech and handwriting can be used for documenting prototypes, by automatically generating a text document of an oral or written tutorial or explanation of the prototype.

5.2.3 Sketch and Single Image to 3D Model

An interesting possibility in making 3D models out of sketches in the form of line drawings is proposed by Lun, Gadelha, Kalogerakis, Maji, and Wang (2017). Their method utilizes a deep network with an encoder and a decoder. The sketch is converted into shape information through the encoder, while the decoder converts the information to depth and normal maps to capture the underlying surface from several viewpoints. MVS can then be used to generate a point cloud and a mesh. Their network is trained with a dataset that

contains 3D shapes with corresponding sketches. Some of the results they have obtained is illustrated in Figure 48 below.

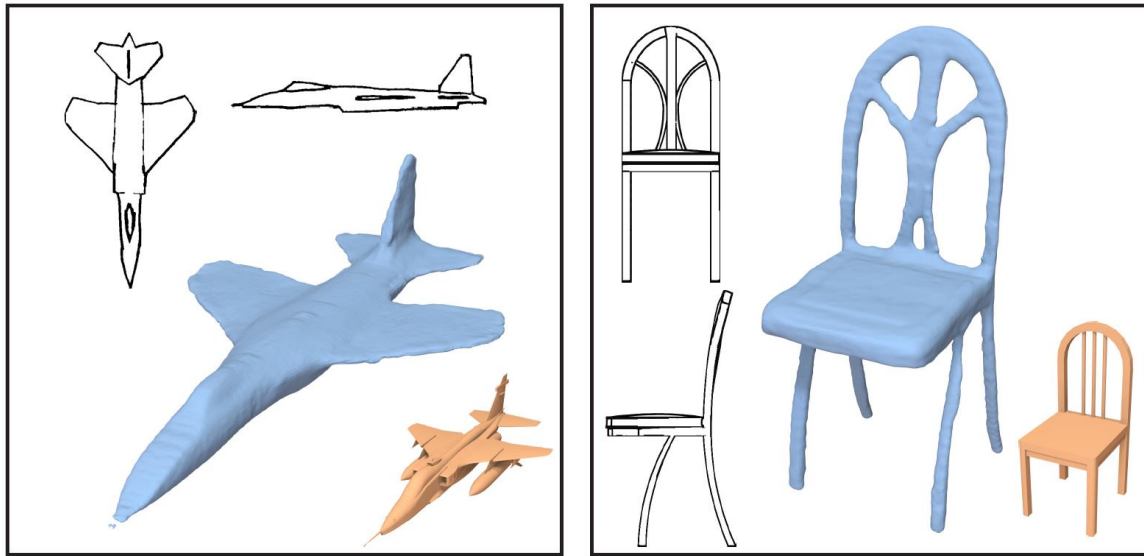


Figure 48: The blue models are generated from the corresponding sketches. The orange shapes are the nearest shapes from the training dataset. Illustrations retrieved from Lun et al. (2017).

Another fascinating approach to generate 3D models is proposed by Choy, Xu, Gwak, Chen, and Savarese (2016). Their deep convolutional neural network can map the underlying 3D shapes from single or multiple images, which is trained on data from an information-rich 3D model repository. Some of their results are shown in Figure 49.

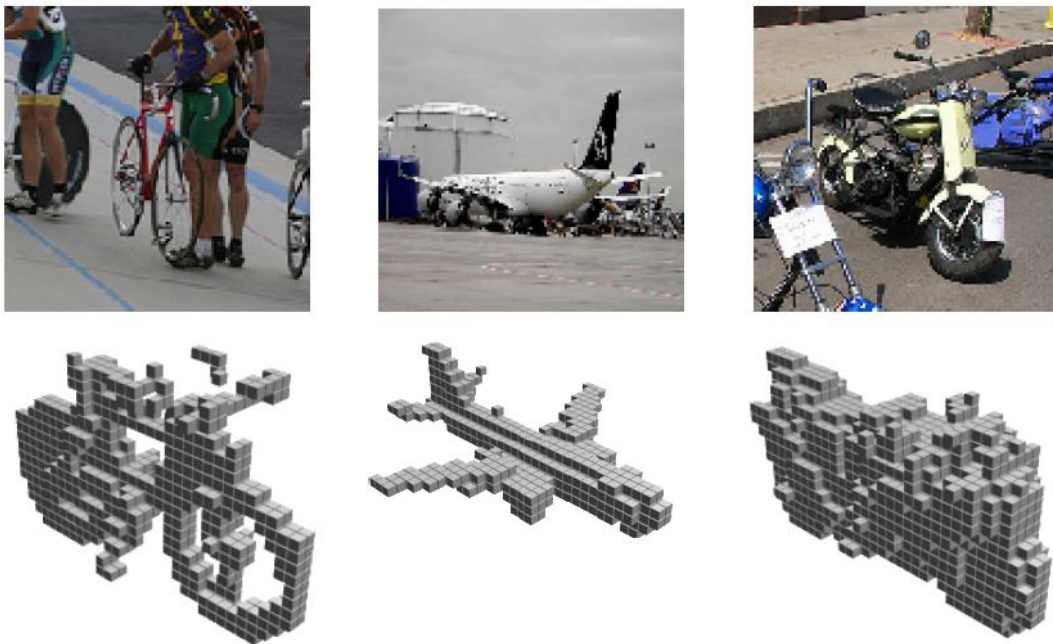


Figure 49: Examples from Choy et al. (2016) for constructing a 3D model from a single image.

5.2.4 Smartphone App

Many of the functions described in this thesis can be implemented into an application for smartphones. Sketch to laser could for example be implemented with the transformation technique described in Subsection 3.2.6, to increase accuracy from hand-captured images. Capturing serial data while recording video could be achieved with a bluetooth dongle attached to the prototype to communicate with the phone. Several apps exist for 3D scanning with photogrammetry, such as Trnio, that use cloud-based processing. However, making an app that includes all the proposed methods would require allot more time and effort, and does not have the benefit of controlled lighting and background.

5.2.5 Mechanical Strength Testing

Simple methods for mechanical strength testing, such as tensile and vibration, is feasible to implement in Protoboost v2, to further push prototype fidelity and learning in early stage PD. It can serve as a quick reference on strength properties of prototypes or materials and geometries. Loads can be measured accurately with similar load cells described in Subsection 3.6.2. It is also possible to make modules, e.g. using Arduino with a memory shield module, that can be used to log data during different scenarios and include it with the system. For example, having a vibration measuring device (e.g. Arduino with piezo elements and accelerometers) along with motorcycle ride to log the dynamic motions and vibrations, then copy the pattern to a vibration test setup to be able to do simple preliminary testing on strength and fatigue on prototypes within specific motorcycle applications. This method might be suitable for common prototyping materials, such as MDF and 3D printer plastics (e.g. PLA and ABS).

This can be an interesting system in and of itself. However, by including more applications to Protoboost or Protoboost v2, data and usage can be logged for research, to discover and understand causalities in early stage PD and provide feedback for the developer(s) (Sjöman et al., 2017), and to share test setups.

6 Conclusion

Four main methods for augmenting physical prototype activities in early stage product development have been explored, tested and implemented with the physical system called Protoboost v2.

The video with serial data module can reliably record (Arduino) microcontroller-based prototypes while showing its real time serial data, to effortlessly document its physical and internal (programmatic) functions.

A pilot experiment has shown that it is feasible for designers to produce sketches for laser cutting physical parts through Protoboost v2, with an acceptable accuracy in many prototyping scenarios even when drawing fast.

The use of photogrammetry to produce digital models, especially through the use of clay models, show potential to increase the speed of generating complex prototypes. Thus, increasing prototype fidelity can be achieved in a simple and fast way.

A classifier has been trained for detecting and recognizing common prototyping tools and components, which can be used to automatically provide useful information for designers trying to realize and build prototypes.

Techniques for potentially improving the modules have been verified, such as automatically correcting distorted images of sketches, using load cells to reinforce object recognition by distinguishing between similar objects, using multithreading to increase accuracy and speed when capturing video and serial data from microcontrollers, and different methods for increasing robustness of photogrammetry-based 3D scanning.

More testing is required, through controlled experiments or real product development projects, to fully understand the extent to which Protoboost v2 can simplify and speed up the development of physical prototypes, to ultimately make better products faster. Based on practical examples, and engineering design research supporting the use of tangible prototypes, the system has shown to potentially augment physical prototype activities in early stage product development.

This page is intentionally left blank.

Bibliography

- AlexeyAB. (2018). darknet for Windows and Linux. *GitHub repository*. Retrieved May 20, 2018, from <https://github.com/AlexeyAB/darknet> & https://github.com/AlexeyAB/Yolo_mark
- Bah, T. (2009). *Inkscape: Guide to a Vector Drawing Program (Digital Short Cut)*: Pearson education.
- Barnes, C., Shechtman, E., Finkelstein, A., & Goldman, D. B. (2009). PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics-TOG*, 28(3), 24.
- Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*: " O'Reilly Media, Inc."
- Bryan-Kinns, N., & Hamilton, F. (2002). *One for all and all for one?: case studies of using prototypes in commercial projects*. Paper presented at the Proceedings of the second Nordic conference on Human-computer interaction.
- Chou, S.-W., & He, M.-Y. (2004). Knowledge management: The distinctive roles of knowledge assets in facilitating knowledge creation. *Journal of Information Science*, 30(2), 146-164.
- Choy, C. B., Xu, D., Gwak, J., Chen, K., & Savarese, S. (2016). *3d-r2n2: A unified approach for single and multi-view 3d object reconstruction*. Paper presented at the European Conference on Computer Vision.
- Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., & Ranzuglia, G. (2008). *Meshlab: an open-source mesh processing tool*. Paper presented at the Eurographics Italian Chapter Conference.
- CUDA Zone. (2018). Retrieved June 1, 2018, from <https://developer.nvidia.com/cuda-zone>
- Edelman, J., & Currano, R. (2011). Re-representation: affordances of shared models in team-based design. In *Design thinking* (pp. 61-79): Springer.
- Edelman, J. A., Leifer, L., Banerjee, B., Sonalkar, N., Jung, M., & Lande, M. (2009). *Hidden in plain sight: affordances of shared models in team based design*. Paper presented at the DS 58-2: Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 2, Design Theory and Research Methodology, Palo Alto, CA, USA, 24.-27.08. 2009.
- Erichsen, J. A., Pedersen, A. L., Steinert, M., & Welo, T. (2016). *Using prototypes to leverage knowledge in product development: Examples from the automotive industry*. Paper presented at the Systems Conference (SysCon), 2016 Annual IEEE.
- Everingham, M., Van Gool, L., Williams, C., Winn, J., & Zisserman, A. (2011). *The pascal visual object classes challenge 2012 (voc2012) results (2012)*. Paper presented at the [URL <http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html>](http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html).
- Furukawa, Y., & Hernández, C. (2015). Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2), 1-148.

- Gerstenberg, A., Sjöman, H., Reime, T., Abrahamsson, P., & Steinert, M. (2015). *A Simultaneous, Multidisciplinary Development and Design Journey—Reflections on Prototyping*. Paper presented at the International Conference on Entertainment Computing.
- Hartvigsen, H., Lorentsen, R., Michelsen, K., & Seljevoll, S. (2006). *Verksted håndboka*. Norway: Gyldendal Norsk Forlag.
- Herstatt, C., & Verworn, B. (2004). The ‘fuzzy front end’ of innovation. In *Bringing technology and innovation into the boardroom* (pp. 347-372): Springer.
- ImageMagick Studio, L. (2008). ImageMagick. In.
- Jancosek, M., & Pajdla, T. (2014). Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces. *International scholarly research notices, 2014*.
- Jansson, D. G., & Smith, S. M. (1991). Design fixation. *Design studies, 12*(1), 3-11.
- Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., . . . Darrell, T. (2014). *Caffe: Convolutional architecture for fast feature embedding*. Paper presented at the Proceedings of the 22nd ACM international conference on Multimedia.
- Kamermans, M. P. (2017). A Primer on Bézier Curves. Retrieved May 2, 2018, from <https://pomax.github.io/bezierinfo/#whatis>
- Kazhdan, M., & Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG), 32*(3), 29.
- Kuhl, K.-H. (2010). KVEC. Retrieved from [http://www.kvec.de/\(27.05.2018\)](http://www.kvec.de/(27.05.2018)).
- Kume, H., Fujii, H., Yamashita, A., & Asama, H. (2016). *Scale reconstructable structure from motion using refraction with omnidirectional camera*. Paper presented at the Mechatronics (MECATRONICS)/17th International Conference on Research and Education in Mechatronics (REM), 2016 11th France-Japan & 9th Europe-Asia Congress on.
- Leifer, L. J., & Steinert, M. (2011). Dancing with ambiguity: Causality behavior, design thinking, and triple-loop-learning. *Information Knowledge Systems Management, 10*(1-4), 151-173.
- Lim, Y.-K., Stolterman, E., & Tenenber, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction (TOCHI), 15*(2), 7.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., . . . Zitnick, C. L. (2014). *Microsoft coco: Common objects in context*. Paper presented at the European conference on computer vision.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision, 60*(2), 91-110.
- Lun, Z., Gadelha, M., Kalogerakis, E., Maji, S., & Wang, R. (2017). 3D shape reconstruction from sketches via multi-view convolutional networks. *arXiv preprint arXiv:1707.06375*.

- Matsugu, M., Mori, K., Mitari, Y., & Kaneda, Y. (2003). Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, 16(5-6), 555-559.
- Nonaka, I., & Takeuchi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation*: Oxford university press.
- Nonaka, I., Toyama, R., & Konno, N. (2000). SECI, Ba and leadership: a unified model of dynamic knowledge creation. *Long range planning*, 33(1), 5-34.
- OpenCV about. (2017). Retrieved November 22, 2017, from <https://opencv.org/about.html>
- OpenMVS. (n.d.). Retrieved, from <http://cdcseacave.github.io/openMVS/>
- Redmon, J. (2016). Darknet: Open source neural networks in c. *Pjreddie. com.[Online]*. Available: <https://pjreddie.com/darknet> [Accessed: 22-May-2018].
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*.
- Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767*.
- Ringen, G., & Welo, T. (2015). *Knowledge based development practices in systems engineering companies: A comparative study*. Paper presented at the Systems Conference (SysCon), 2015 9th Annual IEEE International.
- Ripley, B. D. (2007). *Pattern recognition and neural networks*: Cambridge university press.
- Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*: Malaysia; Pearson Education Limited.
- Schenk, T. (2005). Introduction to photogrammetry. *The Ohio State University, Columbus*, 106.
- Schonberger, J. L., & Frahm, J.-M. (2016). *Structure-from-motion revisited*. Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
- Schönberger, J. L., Zheng, E., Frahm, J.-M., & Pollefeys, M. (2016). *Pixelwise view selection for unstructured multi-view stereo*. Paper presented at the European Conference on Computer Vision.
- Schoonjans, F. (2016). Medcalc statistics for biomedical research: software manual. *Mariakerke: Medcalc Statistical Software*, 295.
- Sjöman, H., Erichsen, J. A. B., Welo, T., & Steinert, M. (2017). Effortless Capture of Design Output. 7.
- Slåttsveen, K. B., Steinert, M., & Aasland, K. E. (2016). *Increasing student confidence and motivation in a project-based Machine Construction and Mechatronics course*.
- Steinert, M., & Leifer, L. J. (2012). 'Finding One's Way': Re-Discovering a Hunter-Gatherer Model based on Wayfaring. *International Journal of Engineering Education*, 28(2), 251.

- Sutcliffe, A., & Sawyer, P. (2013). *Requirements elicitation: Towards the unknown unknowns*. Paper presented at the Requirements Engineering Conference (RE), 2013 21st IEEE International.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). *Going deeper with convolutions*.
- Ullman, D. (2009). *The mechanical design process*: McGraw-Hill Science/Engineering/Math.
- Ulrich, K. T., & Eppinger, S. D. (2012). *Product design and development, 2000*. New York: MacGraw-Hill.
- Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features*. Paper presented at the Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on.
- Viswanathan, V., Atilola, O., Esposito, N., & Linsey, J. (2014). A study on the role of physical models in the mitigation of design fixation. *Journal of Engineering Design*, 25(1-3), 25-43.
- Viswanathan, V., & Linsey, J. S. (2013). Role of sunk cost in engineering idea generation: an experimental investigation. *Journal of Mechanical Design*, 135(12), 121002.
- Vu, H.-H., Labatut, P., Pons, J.-P., & Keriven, R. (2012). High accuracy and visibility-consistent dense multiview stereo. *IEEE transactions on pattern analysis and machine intelligence*, 34(5), 889-901.
- Waechter, M., Moehrle, N., & Goesele, M. (2014). *Let there be color! Large-scale texturing of 3D reconstructions*. Paper presented at the European Conference on Computer Vision.
- Wu, C. (2013). *Towards linear-time incremental structure from motion*. Paper presented at the 3DTV-Conference, 2013 International Conference on.
- Wu, H.-Y., Rubinstein, M., Shih, E., Gutttag, J., Durand, F., & Freeman, W. (2012). Eulerian video magnification for revealing subtle changes in the world.

Appended Conference Paper

The peer reviewed conference paper for Norddesign 2018 is included here. It is based on the work done in this thesis, including some extra details on prototyping scenarios, example cases and comparisons to other existing technical solutions, but is overall less technical. In the paper, Protobooth v2 is called only Protobooth, to include the work done by Sjöman, Erichsen, Welo and Steinert (2017) with the initial prototype capturing tool enabling research on early stage product development. The paper was submitted 03.06.2018.

This page is intentionally left blank.

Augmenting Physical Prototype Activities in Early-Stage Product Development

Sampsa M. I. Kohtala¹, Jørgen A. B. Erichsen¹, Heikki Sjöman¹, Martin Steinert¹

¹ Norwegian University of Science and Technology (NTNU),
Department of Mechanical and Industrial Engineering (MTP)

smkohtal@stud.ntnu.no

jorgen.erichsen@ntnu.no

heikki.sjoman@ntnu.no

martin.steinert@ntnu.no

Abstract

Prototyping in the early-stage of product development is widely used for exploring solutions and generating knowledge. Prototypes in the form of tangible artefacts have many advantages, including expressing and transferring tacit knowledge, creating proof of concepts, learning by doing and testing ideas. In the era of digitalization, this paper attempts to discover opportunities in engineering design research by transforming physical prototypes into digital 3D models, and methods for converting hand drawn sketches to physical parts, in addition to capturing microcontroller output.

With the increasing development and robustness of computer vision and photogrammetry algorithms over the past few years, simple methods for generating digital models of real objects have surfaced. By providing pictures of a prototype, these algorithms can generate a digital 3D representation, including color and texture.

A system for capturing information and knowledge from early-stage product development has been developed, and consists of a digital repository for collecting, storing and sharing data from design output (prototypes), and a physical instrument for capturing the input data. The physical instrument consists of several cameras used for taking pictures of prototypes, and a turntable to capture many angles for further processing to generate a 3D model. In addition, also a tool for producing laser cut pieces from sketches and a microcontroller logger is developed. It is aimed at advancing the discovery and understanding of causalities in the early stage of PD. As a positive side effect of enabling better research, the system can benefit its users (designers) by providing a basis for documentation and feedback.

Through practical experimentation and testing, we aim to discover the potential of methods such as photogrammetry in aiding practitioners reflect through design output (i.e. prototypes) in the early-stages of product development, as well as discussing the limitations of capturing design output from product development projects. Various ways of representing the prototype repository will be discussed, where making the prototypes accessible through virtual reality is

one possible concept that is discussed. The digital repository currently consists of data from various projects, including mechanical engineering student projects, a start-up developing a coreless ring motor, as well as projects from a multi-national product manufacturing company located in Norway.

Keywords: Prototypes, Capturing Prototypes, 3D models, Protobooth, Product Development, 3D scanning

1 Introduction

Prototyping in the early-stage of product development is widely used for exploring solutions and generating knowledge. Prototypes in the form of tangible artefacts have many advantages, including expressing and transferring tacit knowledge, creating proof of concepts, learning by doing and testing ideas (Erichsen, Pedersen, Steinert, & Welo, 2016). In the era of digitalization, this paper attempts to discover opportunities in engineering design research by documenting performance of microcontroller-based prototypes, providing an effortless way to laser cut hand drawn sketches and transforming physical prototypes into digital 3D models.

With the increasing development and robustness of computer vision and photogrammetry algorithms over the past few years, simple methods for generating digital models of real objects have surfaced. By providing pictures of a prototype, these algorithms can generate a digital 3D representation, including color and texture.

One of the main problems with researching tools and methods in early-stage product development research is access to representative cases and companies that want to disclose their methods and findings from such projects. That is why the authors developed a tool named Protobooth that helps designers to capture and remember their prototypes better than before it was possible while allowing researchers to tap into the companies' and individuals' documenting process.

1.1 Research background

In the pre-requirement stages, or the fuzzy front end (Herstatt & Verworn, 2004) of engineering design, designers are often working in large solution spaces along with ambiguous information and uncertainty. The greatest potential for discovering and testing new innovative solutions lie in these early stages, as they will greatly affect the cost and quality of the later converging development activities, such as optimization and manufacturing.

One of the key elements in early-stage PD is the generation and utilization of knowledge (Nonaka & Takeuchi, 1995; Ringen & Welo, 2015; Sutcliffe & Sawyer, 2013). Learning mechanisms in PD can be categorized into three loops (Leifer & Steinert, 2011). Learning loop one is based on explicit knowledge and aims to retain knowledge from the development projects. Learning loop two involves the informal space between PD team members and their coach. Tacit knowledge (Polanyi, 2009), learning loop three, is the skill and learnings of the individuals (i.e. designers). We argue that interacting with prototypes, in the form of tangible artifacts, is one of the most valuable dimensions of tacit knowledge as a means of knowledge acquisition (Nonaka, Toyama, & Konno, 2000). Prototypes, both external and internal reflective prototypes (Erichsen et al., 2016), can be used as learning tools by conceptualizing and sharing ideas, often with the intention of testing functionalities or suggesting the appearance of a product concept (Lim, Stolterman, & Tenenber, 2008). With the benefits of interacting

with prototypes (and prototyping), more is often better, thus requiring a certain amount of time to be created. A solution for reducing both time and cost is the use of low fidelity prototypes (Bryan-Kinns & Hamilton, 2002).

Thus, our aim in this paper is to augment physical prototyping activities by simplifying and speeding up the process, ultimately allowing the generation of more design iterations to emerge in the early pre-requirement stages of product design. If we succeed, both experienced and inexperienced product developers can benefit from our system, to ultimately make better products faster.

1.2 The Proto booth Project

In order to address the problem of getting a consistent method for documenting early-stage PD projects, a system for capturing project output from early-stage PD projects has been developed by the authors. This system is comprised of three main parts; a physical sensor platform for capturing data from project output (i.e. prototypes), a repository for storing aforementioned captured data and a user interface for interacting with the repository.

The system, which originates from the efforts described by Sjöman, Erichsen, Welo, and Steinert (2017), uses a multitude of sensors, including 7 cameras, load cells and RFID readers to capture data on prototypes (as well as sketches) from PD projects. Although rough details were presented (Sjöman et al., 2017), the system has seen some major upgrades in fidelity, performance and scale. Consequently, usage of the system has increased, and there are now around 50 users (of which about 20 are active each week), and the repository has roughly 400 prototype scans to date (early March 2018). One of the core principles of this system is that adding data to the repository should be effortless, thus increasing the chance of users wanting to utilize the system and leading to the capture of more project data.

The main objective of this research is to enable research of early-stage product development through capturing design output from projects (as detailed by Sjöman et al. (2017)). A goal of performing this research is to be able to feed this knowledge back into the PD process, in order to make better products. Consequently, this paper aims to present some key concepts that have been explored in a research setting with mechanical engineering graduate students, in order to aid designers in early-stage PD projects.

1.3 Aiding Designers in Early-Stage PD Projects

Currently, the system can be categorized as a tool that users (designers) can use for documenting project progress. However, in this paper, we aim to highlight how we can experiment with adding more incentives and features to the system, thus making the system more useful to designers doing early-stage product development.

As the authors' research laboratory closely relates (both in proximity and activity) to the product realization lab at the mechanical engineering department at our university, the authors experience that users aiming to realize products (and prototypes) often have a certain set of tools that they are familiarized with and frequently use. The same users have a broader spectrum of tools and equipment that could be used, yet they tend to stick with methods and machinery that are familiar. A real example of such under-utilization is users 3d-printing small boxes for various micro-controllers (Arduino Uno, Raspberry Pi, etc.), and using such simple geometry (e.g. rectangles) that a laser cutter would produce more durable and more accurate models in less time than a FDM 3D printer. Therefore, we believe that lowering the threshold of learning

new equipment and machinery for realizing new products and prototypes will also make the lab users better equipped to solving their various tasks.

Based on both observation and experience from PD activities and some insight into start-up activity, we also believe that helping developers realize their ideas by producing prototypes will ultimately result in better solutions (and in less time). By contributing to more design alternatives, through helping design iterations emerge simpler and quicker, a single individual or a team of developers can gain more knowledge with less time. If this is the result of the methods presented in this paper, we have successfully achieved our goal of augmenting physical prototype activities in early-stage product development.

2 Prototype Realization Scenarios

In our prototyping laboratory, most activities consist of building prototypes, in either individual or team-based projects. Projects usually revolve around high novelty product challenges in the fuzzy front end (pre-requirement stages) of product development. Attempting to solve these challenges are done through iterations (Steinert & Leifer, 2012) where designing, building and testing is done in several probes along the project timeline. The core concept of this model is to promote iterative learning cycles (Eris & Leifer, 2003) driven by rapid conceptual and tangible prototyping.

In this chapter, we present some common usage scenarios, and some challenges that we have identified in these scenarios. The aim of doing this is to highlight various prototype realization challenges, and later address how they could be solved.

2.1 Scenario 1: Designing and producing physical parts

A common tool for creating physical prototypes is the laser cutter, mainly used for cutting materials such as Medium Density Fiberboard (MDF) and acrylic glass. It is popular due to its speed, accuracy and reliability, but requires training in order to be utilized properly and efficiently. Experienced users will use software, often Computer Aided Design (CAD) - software, to create a 2D sketch, export it as a vector file, save it to a removable drive, transfer it to the computer connected with the laser cutter, modify the cutting sequence and material properties in the laser cutting software, and finally press print to start cutting. Even experienced developers can spend a lot of time on making a laser cuttable file with the desired dimensions and geometry. Having a computer and measuring tools near the prototype is commonly observed in the laboratory during the building and testing of prototypes. Conceptual properties can be lost in the process of making prototypes for the laser cutter, as the physical connection between the idea and its practical use can be lost in all the software. Not until having produced the parts is it possible to perceive its real properties and applicability. This can result in designs that are not properly scaled, and not based on initial concepts of size and fit, thus promoting rework or results that are 'just good enough' and stays as is.

Many new and inexperienced users are more hesitant in using the laser cutter, mostly due to the uncertainty with learning new machines. Beginners have been observed to instead use more traditional tools, such as a saw, to form materials when creating prototypes. The threshold for becoming familiar with new software can be high, especially in early stages of development combined with a lack of experience. While learning and generating new knowledge is important in new product development (NPD), it can be time-consuming and best spent on the challenge rather than learning to operate new tools and machines. Consequently, building might take up

more time and focus compared to testing and developing new ideas. The time from concept to physical prototype should therefore be reduced if possible.

2.2 Scenario 2: Documenting performance of microcontroller-based prototypes

It has become more common to use microcontrollers in product development and mechanical engineering courses (Slåttsveen, Steinert, & Aasland, 2016). Microcontrollers, such as the Arduino, provides a simple and fast method for interacting with the physical environment by controlling actuators and using sensors. Mechanical engineering students are encouraged to learn how to utilize them to build functional prototypes.

It is a less frequent practice to document such prototypes in an effective and effortless way. Usually they are documented by providing the script filled with comments explaining the code. When the actual prototype is used or demonstrated, it is difficult to show how it works programmatically. Printing serial data is used for debugging and understanding how the prototype works, but this content is often only used by the programmer who is writing and testing the code or prototype.

2.3 Scenario 3: Utilization of 3D technology

CAD is a tool mostly used later in the development process, when requirements and specifications are made (and set). In the early stages of development, it is in many scenarios considered to be less effective, especially when developing novel products with a large solution space. In this case, it is often more rewarding to build prototypes that can be tested in the real world, to get a better sense of the concept(s) being tested and how they perform and to discover unknown unknowns (Gerstenberg, Sjöman, Reime, Abrahamsson, & Steinert, 2015). However, when prototypes need higher resolution it is common to utilize rapid prototyping tools such as 3D printing. The normal approach is then to create a 3D model with CAD. Designing and drawing 3D models can be a slow process, especially if the shapes are complex and/or based on an abstract concept. As in the case for making 2D parts for laser cutting, the perception can be skewed in the process, as the CAD can perfectly simulate an ideal part but differ in the real world. The scale of a model is an example that is often underestimated, resulting in small, unstable and weak parts that are difficult to produce. It is also common for newer engineering students to find CAD difficult and cumbersome.

With powerful computers, it is possible with 3D scanning to generate accurate 3D models of real objects quickly. It could potentially be feasible to, within a few seconds, have an augmented reality representation of the real model at a remote location, or 3D-print it to get a physical copy. It would also be possible to combine the real-world prototypes with virtual reality prototypes to make assemblies at several different locations at the same time, thus keeping production separate at long distances. The authors imagine having sessions with teams where people work at different locations around the globe but are able to virtually assemble each part together to simulate and test the overall functionality. Another possibility could be that customers could test the physical prototypes or user experience prototypes before costly production takes place.

3 Technology solutions

In this paper we highlight experiments with new features for aiding designers in early-stage PD activities. Additionally, we will present the technologies behind these experiments, including

the hardware used and the software implementations. Furthermore, we will provide possible benefits that this system can bring to early-stage PD in general and present an in-depth look at user activity in our research laboratory with examples of how the system can help these users with realizing prototypes and documenting development. First, we present existing solutions and argue why our system is solving the challenges in a more efficient way.

3.1 Technical solutions

3.1.1 Existing solutions

3D scanning has become a more common tool in the industry, used for reverse engineering and quality control. In PD, it is often used to accurately model complex geometries, such as faces, hands, bones or limbs (for prosthetics), to make custom products. However, it is generally not associated with the pre-requirement stages of PD, therefore often avoided or not considered until later development stages. Some of the main drawbacks with most commercial 3D scanning equipment is cost, scanning time and complexity. HP 3D Structured Light Scanner is an example, which is a professional-level instrument using a camera and a projector. The instrument is highly accurate (up to 0.05 mm) and fast, but is an advanced system consisting of many elements that needs to be manually adjusted to work properly, in addition high cost. Matter and Form 3D Scanner is a user-friendly system and a low-cost alternative. It is however slow and more limited in the object size it can 3D-scan. There are also many handheld scanners that provide more flexibility during the scanning process, and mobile apps that utilize photogrammetry to reconstruct 3D models from pictures through cloud processing. Depth-sensor based systems does not have very high resolution or are very expensive and advanced. The method of Photogrammetry is often more suitable due to availability of good cameras with high resolution. Generally, commercial Photogrammetry programs are interactive and fixed in its capabilities. Photogrammetry can be difficult, but is very flexible, accessible and low cost, thus having the potential to be tailored for specific usage and automation.

Using hand drawn sketches to directly produce physical parts is not common practice in PD activities to the knowledge of the authors. Converting hand drawings to digital copies or g-code for engraving is used mostly in the visual arts. A drawing is then typically scanned with a normal paper scanner and manually processed with software such as Inkscape and Photoshop.

Ready-made microcontrollers (e.g. Arduino, etc.) are widely used in early-stages of PD to quickly make functional prototypes. It is a great way to facilitate ideas and test them in the real world and show how the product works. It is however less common to intuitively demonstrate and document how the physical prototype works while simultaneously demonstrating how it works internally (programmatically). Usually, data from the microcontroller is displayed through software, such as the Arduino integrated development environment or Processing.

3.1.2 What we offer through Protobooth

Our system provides an all-in-one solution to different challenges in product design and development. The different technologies discussed in the previous chapter are integrated into one instrument, in such a way that it is as simple to use as possible. Furthermore, we use open source software developed by the community, which enables more control and automation possibilities while being free of charge. In Protobooth, we control the programs and the physical environment (lighting and background color), which enables us to tailor its functions to the users' needs. In this way we can promote new innovative approaches for producing prototypes and help developers realize ideas.

Photogrammetry is usually done by experts that are familiar with its limitations. In the controlled environment of Protoboosth, we can experiment with different materials and textures, and use this information to automatically adjust parameters based on what is put inside. Users can then utilize this technology without being experts in photogrammetry or having to learn new 3D scanning equipment. Photogrammetry is also a very fast way to capture the required spatial data compared to other sensor-based methods; however model reconstruction time can vary greatly based on processing power and data quality. Another benefit with photogrammetry is that the scanned model can vary in size, from small components to large structures. Another benefit is that the reconstructed model will always include detailed texture, providing accurate, yet simple, possibilities for visual communication. This is promising for virtual reality applications in the near future.

Making hand drawn sketches to directly produce physical parts in the laser cutter is seldom observed in PD activities, due to existing software being more directed towards other applications. We believe this approach can help designers make parts faster and with less effort in many scenarios. By implementing this function in Protoboosth, users can make simple designs and quickly have results they can use.

Even though it is common to use the IDEs (and serial monitors) for displaying how microcontrollers work, it is often not convenient for documenting or sharing this knowledge. It can also be a steep learning curve to simply log sensor data from a microcontroller to a computer, to enable plotting and other forms of data processing. With Protoboosth, these steps are accomplished by simple plug and play, linking the physical prototype to the actual (real-time) code output. The users can thus focus more on the actual prototyping while getting data out of their activities and prototypes for free.

3.2 Hardware

The main elements of the physical Protoboosth, some of which are depicted in Figure 1, consist of:

- Structure made of an aluminum frame, wooden walls and a table
- Several Logitech C930e and C920 web cameras
- A turntable
- LEDs
- Intel NUC with Debian 9
- RFID reader
- Arduino Mega
- Nikon D5300 camera
- Cross laser



Figure 1. The Protoboost used for testing new features. The camera used for detecting and capturing sketches is shown on the top right image.

The Arduino Mega handles the LEDs, cross laser and a stepper motor driving the turntable. The Arduino, RFID reader and cameras are connected to and controlled through NUC.

3.3 Software

Most of the software used is open source, except laser cutting software “Gravostyle”. Logitech web cameras are used with an open source computer vision library [OpenCV], while the Nikon camera is controlled through command line arguments provided by gPhoto2. Node-red, a Node.js framework with its browser-based flow editor, connects hardware and programs in an orderly manner, shown in Figure 2. The main programs are written in C++.

A program for controlling an Arduino mega through serial communication is used for simply setting turntable parameters, activation and changing color of LEDs.

The object classification program is used to capture an image of the current object placed in Protoboost and classifying it with a trained neural network, using YOLOv2 by Redmon and Farhadi (2016). It can detect if a sketch is present or not, as such determine the next program to run.

Video recording is accomplished using OpenCV. If an Arduino microcontroller is connected while recording, the serial data from the microcontroller is displayed on the video in real time. A separate text document with the data is also provided.

The sketch to laser cutter program captures an image of the sketch placed in Protobooth. It converts the image to a scaled PDF file and a binary image. The binary image is further processed into a vectorized DXF file using KVEC.

For 3D-scanning, a program was made, utilizing gPhoto2, to capture several images with Nikon D5300 while the object rotates on the turntable. 3D reconstruction from the images is accomplished through several steps, using COLMAP (Schönberger, Zheng, Frahm, & Pollefeys, 2016) and OpenMVS. Steps include feature detection, extraction and matching, generating a point cloud, densifying the point cloud, generating a mesh, refining the mesh and finally adding texture to the model.

Google drive is used for sending files between Protobooth and laser cutter PC. This procedure is used in the experimentation stage. In the future, all data can be sent to the users' repository, accessible through a web interface.

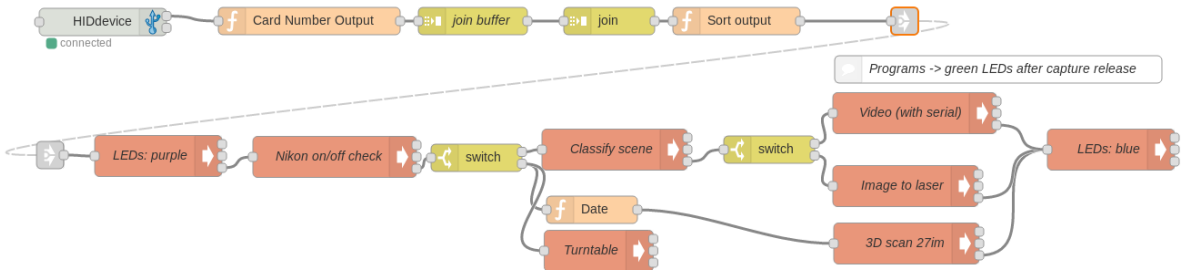


Figure 2. Node-red interface linking functions and executable programs.

One of the goals of Protobooth is to keep it as simple and nonintrusive as possible, lowering the threshold for using it while helping both experienced and inexperienced product developers realize and convey their ideas. The users can simply place their prototype in Protobooth and scan their personal ID access card on the RFID reader. This activates a series of programs, illustrated in Figure 2, based on what is put inside Protobooth. For 3D scanning it is necessary to activate and position the camera manually first. LEDs are used to indicate at which stage the program is currently at, where purple means the prototype is being captured, green means capturing is complete and the user can remove the prototype, and blue meaning Protobooth has processed every stage and is ready to be used again. The cross laser is used to mark the center and will automatically turn of while capturing prototypes.

4 Experimenting with aiding designers

4.1 Sketch to laser

Protobooth is capable of capturing hand drawn sketches and convert them to a scaled PDF and vector (DXF) file. After detecting centerlines in the sketch and applying a simulated Bezier algorithm, both provided by KVEC, a DXF file is generated containing vectors made up of approximated polylines. The files are automatically sent to a shared folder on the computer that has the laser cutter software installed. After a user has captured the sketch with Protobooth, it takes approximately 10 seconds before the files are ready and received at the laser PC. By simply dragging and dropping these files into the laser cutting software, they can be modified and prepared for cutting.

When a vector is selected in the software every connected vector is highlighted, which is essential when applying the cutting sequence to different parts of the design and not having to select each vector individually. The PDF is a scaled version of the original image captured of the sketch. When this file is opened in the software it is possible to manually draw vectors over the image to make refined and customized cutting patterns of the original sketch.

4.1.1 Practical example

An example case, where a locking mechanism was needed for a drawer, was used to test and demonstrate the system. Without knowing the dimensions or exactly how it should look, the normal approach would be to take measurements, make a few designs and build them for testing. Instead, a piece of paper was placed at the area of interest, and an outline was drawn directly where it should fit. Using a lead pencil made it simple to refine the sketch after the first rough version was drawn.

After scanning the sketch in Protobooth, the DXF file was dropped into the laser cutting software. Like in any case when using the software to cut materials, the only required change made to the vectors was changing color to specify the cutting sequence, which was done with a few mouse clicks. After setting laser the parameters and choosing the material, the part was cut smoothly.

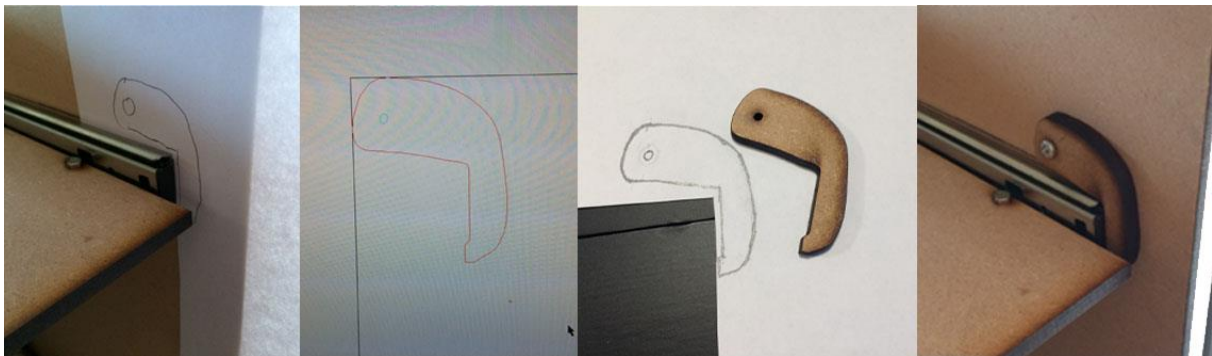


Figure 3. From left: pencil drawing, vectors generated from the drawing (colors are selected for cutting sequence), comparing the result with the original sketch, and lastly showing its final application. Note that the leftmost sketch was slightly modified before scanning, as shown on the middle-right image.

4.1.2 Limitations and possible solutions

A reoccurring problem with this approach has been a presence of many vectors not part of the sketch, in addition to not discovering thin or weak lines. The simple explanation is the quality of the picture taken with the Logitech web camera, or the lighting condition. Although decent for video recording, 2MP is low for taking pictures. In some cases, this might have caused noise in the binary image due to the difference in observed color between white paper and drawn lines being too small. Using a higher resolution camera with different lighting can resolve this issue.

4.2 Video recording with serial output

When connecting a microcontroller to the available USB port, Protobooth (NUC) will open the port and read the serial data that is received. Currently only Arduino based microcontrollers are tested and accepted by the program. When a user scans their RFID card a signal is sent to the Arduino which will reset it. Video recording is then started and synced with the serial output.

The output text is written to the video frames using OpenCV. With the current hardware setup, it is possible to record 1080p video up to 30fps.

4.2.1 Practical example

A microcontroller-based system was developed at a course by a team of three students including one of the authors. It is a modular robot system consisting of different modules (boxes) that can communicate with each other through radio signals. Each box is equipped with an Arduino and a transceiver.

To showcase how the prototypes could be used, a handful of modules was created: a joystick, car, robot arm, power supply and sensor module (see Figure 4). Even though the team behind the project consisted of only three students, it was not entirely clear to everyone how these modules worked together, as they were built and programmed by different team members.

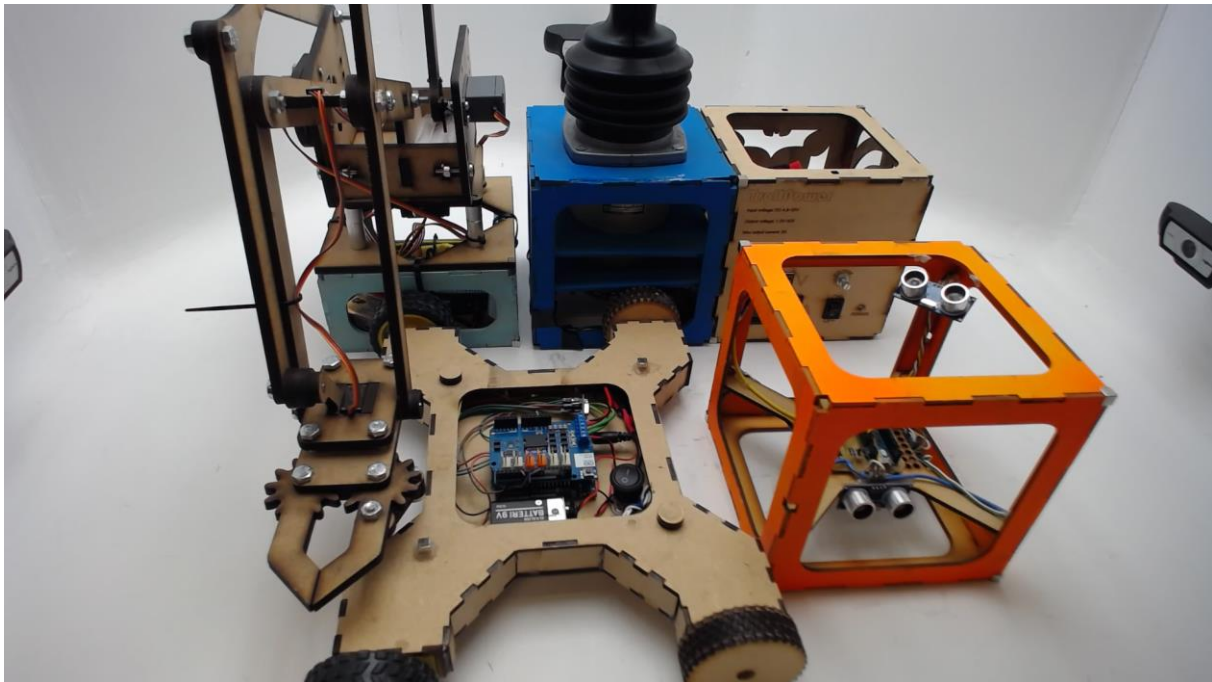


Figure 4. Prototypes from the modular robot system.

The system has since been documented using Protobooth. One of the videos made, shown in Figure 5, demonstrates how different modules react to signals received by the joystick module. By simply connecting the joystick module to Protobooth, it was easy to demonstrate what types of signals were sent to the other modules. In Figure 5 the numbers represent the angle of one of the servos on the arm, which are changing based on the joystick movement at different rates while the arm moves accordingly. It is shown on the bottom that the yellow button is pressed on the joystick, and moments later the car is controlled instead of the arm. In this way it is clearly demonstrated how the modules react and how they work together. A 36 seconds long video was needed to properly demonstrate the functionality of this setup.

Other short videos were also made (Kohtala, 2018), showing how to set up and initiate communication with the different modules. Not only can this improve communication within teams, but other users can play with these prototypes now that they can simply learn how to use them on their own.

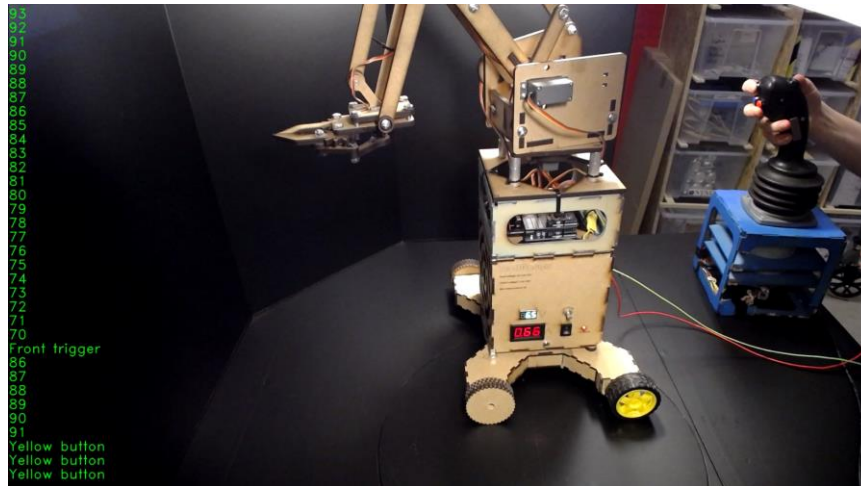


Figure 5. Snapshot from a video captured with Protoboost showing how different modules work together. The joystick module on the right is connected to Protoboost while recording, and its serial output is displayed in real time on the left side of the video frame.

4.2.2 Limitations

It requires good programming practices to maximize the potential of this method. Even if data is printed to the screen, it does not guarantee that the video with serial text makes sense to other viewers.

4.3 3D-scanning

Protoboost has a semi-automated process for 3D-scanning a prototype. A DSLR camera must be aimed manually to capture the object, before an automated process rotates the object while capturing a total of 28 images. This process takes about 28 seconds, restricted by the capturing and downloading speed of the camera using gPhoto2.

Photogrammetry is used to reconstruct a 3D model from a set of images. COLMAP and OpenMVS is used to automatically generate a complete 3D model. Through several steps, the algorithms outputs point clouds and meshes with MVS and PLY file formats. The last stage generates a refined mesh with texture. Processing time can vary based on the complexity of the object and the image quality, in addition to processing power. MeshLab can be used to further modify the outputs or to simply export the model to STL format, which is supported by most CAD and 3D printing software.

4.3.1 Practical example

An attachment for a Raspberry Pi to a Nikon camera was made with modelling clay, as an experiment for testing the system. Different colored clay was mixed to add texture and improve feature detection. The clay was quickly molded by pressing it against the camera and Raspberry, and then shaped by hand. A small piece of clay was also used to hold the model upright while scanning, to capture the largest and most critical surfaces in one scan.

Markers were used to further assist the algorithms to detect features and align photos. The markers and support were removed in MeshLab after the reconstruction was complete. A function in NX was used to automatically fill holes on the model, which came from removing the clay support in MeshLab, in addition to scaling the model after measuring two points on both the clay and digital model. A feature was also added with NX for the locking mechanism which was not present on the clay model, shown as the light brown part in Figure 6.

To save time another computer than the NUC was used. With an i7-7700HQ CPU running at around 3.5GHz with a 16GB memory capacity, reconstruction finished after 10 minutes. Different stages of the process are shown in Figure 6. The final 3D-printed prototype (see Figure 7) fits nicely to the camera and is easy to connect while remaining rigid in place.

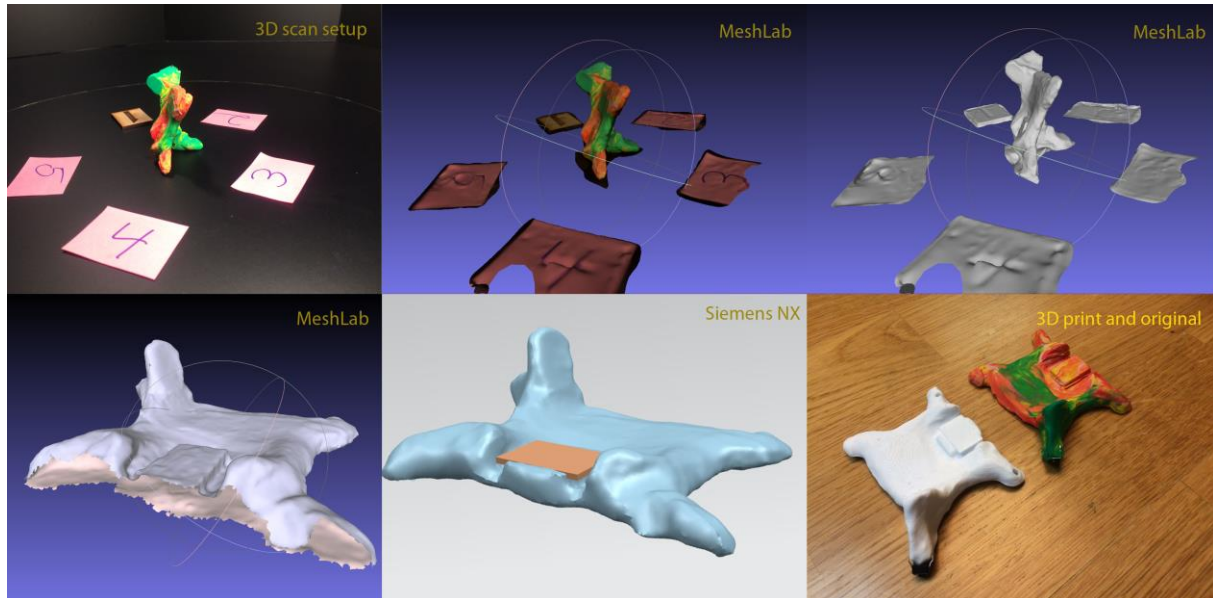


Figure 6. 3D-scan to 3D-print.

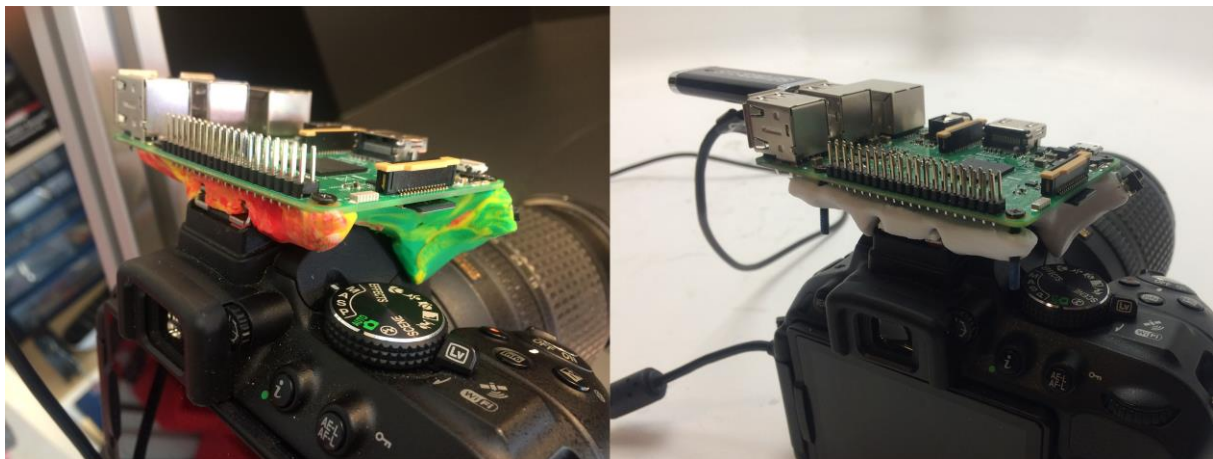


Figure 7. Modelling clay to 3D-print result.

4.3.2 Limitations and possible solutions

Low memory can result in reconstruction failure, thus a decent computer is required to execute the programs properly and in less time. Results from scanning does currently requires some manual processing to remove unwanted features before printing the model. However, solutions exist in MeshLab where specific colors on the mesh can be selected and removed. It is then possible to use markers, and support if needed, with specific colors to be more easily removed from the final model. The reconstructed model is not scaled and must be manually calculated and set in software to get a correct digital representation. It might be possible to implement methods to apply and calculate scale factor based on camera parameters such as focal length and distance to object.

5 Discussion and conclusion

We have presented observations from early-stage PD activities and common prototype realization scenarios that can be challenging for some designers. To address these challenges, new features has been added to the prototype capturing system (Protobooth) developed by the authors, including 3D scanning, techniques for converting hand drawn sketches to laser cut parts and a way to document microcontroller-based prototypes.

Through practical examples, we have proposed how these methods can aid designers develop physical prototypes in early-stage PD and discussed their limitations and possible solutions. The main methods explored to augment physical prototype activities are simplifying the utilization of 3D scanning technology, a fast and simple way to make laser cut parts and a plug-and-play method for documenting microcontroller-based prototypes. Additionally, we have tested object recognition methods to make the user experience of Protobooth even simpler, by automatically detecting what is inserted before running the next appropriate step. Results from experimenting with these features has shown the potential to aid designers. However, it requires more testing and feedback from users to determine its true potential.

We aim to further improve the system based on user feedback, by keeping Protobooth as an evolutionary yet functional tool.

Acknowledgement

This research is supported by the Research Council of Norway through its user-driven research (BIA) funding scheme, project number 236739/O30.

References

- Bryan-Kinns, N., & Hamilton, F. (2002). *One for all and all for one?: case studies of using prototypes in commercial projects*. Paper presented at the Proceedings of the second Nordic conference on Human-computer interaction.
- Erichsen, J. A., Pedersen, A. L., Steinert, M., & Welø, T. (2016). *Using prototypes to leverage knowledge in product development: Examples from the automotive industry*. Paper presented at the Systems Conference (SysCon), 2016 Annual IEEE.
- Eris, O., & Leifer, L. (2003). Facilitating product development knowledge acquisition: interaction between the expert and the team. *International Journal of Engineering Education*, 19(1), 142-152.
- Gerstenberg, A., Sjöman, H., Reime, T., Abrahamsson, P., & Steinert, M. (2015). *A Simultaneous, Multidisciplinary Development and Design Journey—Reflections on Prototyping*. Paper presented at the International Conference on Entertainment Computing.
- Herstatt, C., & Verworn, B. (2004). The ‘fuzzy front end’ of innovation. In *Bringing technology and innovation into the boardroom* (pp. 347-372): Springer.
- Kohtala, S. (2018). TrollBot video serial playlist. Retrieved May 30, 2018, from https://www.youtube.com/playlist?list=PLj9XjbRcnJ_0AMBmT0Idtu9-BngYM0qmp
- Leifer, L. J., & Steinert, M. (2011). Dancing with ambiguity: Causality behavior, design thinking, and triple-loop-learning. *Information Knowledge Systems Management*, 10(1-4), 151-173.

- Lim, Y.-K., Stolterman, E., & Tenenberg, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 15(2), 7.
- Nonaka, I., & Takeuchi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation*: Oxford university press.
- Nonaka, I., Toyama, R., & Konno, N. (2000). SECI, Ba and leadership: a unified model of dynamic knowledge creation. *Long range planning*, 33(1), 5-34.
- Polanyi, M. (2009). *The tacit dimension*: University of Chicago press.
- Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*.
- Ringen, G., & Welo, T. (2015). *Knowledge based development practices in systems engineering companies: A comparative study*. Paper presented at the Systems Conference (SysCon), 2015 9th Annual IEEE International.
- Schönberger, J. L., Zheng, E., Frahm, J.-M., & Pollefeys, M. (2016). *Pixelwise view selection for unstructured multi-view stereo*. Paper presented at the European Conference on Computer Vision.
- Sjöman, H., Erichsen, J. A. B., Welo, T., & Steinert, M. (2017). Effortless Capture of Design Output. 7.
- Slåttsveen, K. B., Steinert, M., & Aasland, K. E. (2016). *Increasing student confidence and motivation in a project-based Machine Construction and Mechatronics course*.
- Steinert, M., & Leifer, L. J. (2012). 'Finding One's Way': Re-Discovering a Hunter-Gatherer Model based on Wayfaring. *International Journal of Engineering Education*, 28(2), 251.
- Sutcliffe, A., & Sawyer, P. (2013). *Requirements elicitation: Towards the unknown unknowns*. Paper presented at the Requirements Engineering Conference (RE), 2013 21st IEEE International.

This page is intentionally left blank.

Appendix A

Programs



QR-link to google drive with the files: <https://drive.google.com/open?id=1wmZQzJ1baiEOD11fASxr-vJaIJujuXs0>

- 1- Arduino
- 2- Status
- 3- Video with Serial Data
- 4- 3D scanning
- 5- Object Recognition
- 6- Sketch to Laser

This page is intentionally left blank.

Appendix B

NS-ISO 2768-1 Table of Tolerances

From Hartvigsen, H., Lorentsen, R., Michelsen, K., & Seljevoll, S. (2006). *Verksted håndboka*. Norway: Gyldendal Norsk Forlag.

This page is intentionally left blank.

**Toleranser for ikke toleransesatte mål
(NS-ISO 2768-1)
Tillatte avvik**

Mål i mm

Basismål		Tillatte avvik Nøyaktighetsgrad			
over	t. o. m.	Fin	Middels	Grov	Meget grov
0,5 -	3	± 0,05	± 0,1		
3 -	6	± 0,05	± 0,1	± 0,2	± 0,5
6 -	30	± 0,1	± 0,2	± 0,5	± 1
30 -	120	± 0,15	± 0,3	± 0,8	± 1,5
120 -	315	± 0,2	± 0,5	± 1,2	± 2
315 -	1 000	± 0,3	± 0,8	± 2	± 3
1 000 -	2 000	± 0,5	± 1,2	± 3	± 4
2 000 -	4 000	± 0,8	± 2	± 4	± 6
4 000 -	8 000		± 3	± 5	± 8
8 000 -	12 000		± 4	± 6	± 10
12 000 -	16 000		± 5	± 7	± 12
16 000 -	20 000		± 6	± 8	± 12
nærmeste IT-grad ¹⁾ t.o.m. 500 mm		12	14	16	17

¹⁾ ISO grunntoleranser.

Angivelse på tegninger:

På tekniske tegninger anbringes henvisning til denne standarden på det stedet i tittelfeltet som er reservert for slike data. Henvisningen skjer ved angivelse av den valgte nøyaktighetsgraden og standardens nummer, f.eks. middels NS-ISO 2768-1. Eventuelt kan også tabellen for de tillatte avvikene stemples eller trykkes på tegningen.

This page is intentionally left blank.

Appendix C

Drawing Experiment Task and Results

This page is intentionally left blank.

Drawing experiment

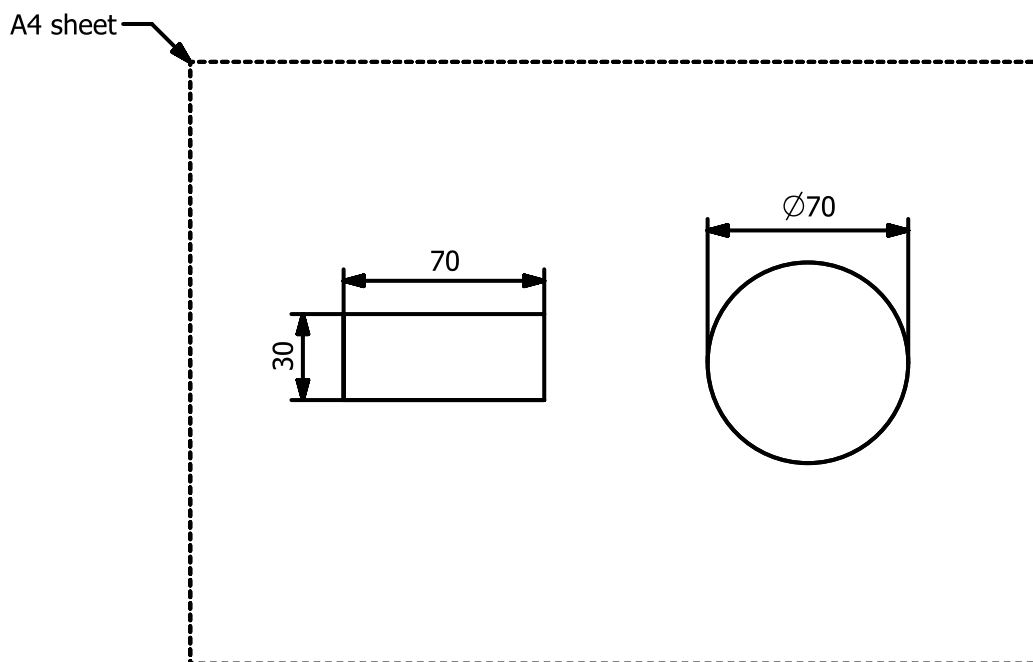
Thank you for participating in this experiment for my master's thesis. I've built a system where drawings can be captured and used for cutting materials in a laser cutter. I want to discover how accurately, and fast, people can draw to get an indication of what applications this system is suited for.

The experiment consists of 2 tasks. For each task you will be given an A4 paper and a set of tools. You can test the tools before starting the task, but the compass ("passer") must be closed before starting. The time starts when you are ready.

Follow the task instructions and draw the figures approximately as shown in the figure.

Task 1: Draw the rectangle and circle shown below, in millimeters, as *accurately* as you can within 5 minutes.

Task 2: Draw the rectangle and circle shown below, in millimeters, as *fast* as you can within 30 seconds. You also get points for accuracy.



Paired samples t-test

Sample 1	T1_Angle
Sample 2	T2_Angle

	Sample 1	Sample 2
Sample size	14	14
Arithmetic mean	90.0179	89.6071
95% CI for the mean	89.7158 to 90.3199	89.0446 to 90.1697
Variance	0.2737	0.9492
Standard deviation	0.5232	0.9743
Standard error of the mean	0.1398	0.2604

Paired samples t-test

Mean difference	-0.4107
Standard deviation of differences	1.1710
Standard error of mean difference	0.3130
95% CI	-1.0868 to 0.2654
Test statistic t	-1.312
Degrees of Freedom (DF)	13
Two-tailed probability	P = 0.2121

Differences

Shapiro-Wilk test for Normal distribution of differences	W=0.8162 reject Normality (P=0.0080)
--	---

Paired samples t-test

Sample 1	T1_Diameter
Sample 2	T2_diameter

	Sample 1	Sample 2
Sample size	14	14
Arithmetic mean	70.0171	69.9407
95% CI for the mean	69.7893 to 70.2450	69.1885 to 70.6929
Variance	0.1557	1.6973
Standard deviation	0.3946	1.3028
Standard error of the mean	0.1055	0.3482

Paired samples t-test

Mean difference	-0.07643
Standard deviation of differences	1.3817
Standard error of mean difference	0.3693
95% CI	-0.8742 to 0.7213
Test statistic t	-0.207
Degrees of Freedom (DF)	13
Two-tailed probability	P = 0.8392

Differences

Shapiro-Wilk test for Normal distribution of differences	W=0.9419 accept Normality (P=0.4428)
--	---

Paired samples t-test

Sample 1	T1_Height
Sample 2	T2_Height

	Sample 1	Sample 2
Sample size	14	14
Arithmetic mean	29.6846	30.3668
95% CI for the mean	29.3452 to 30.0240	29.4978 to 31.2358
Variance	0.3455	2.2654
Standard deviation	0.5878	1.5051
Standard error of the mean	0.1571	0.4023

Paired samples t-test

Mean difference	0.6821
Standard deviation of differences	1.5713
Standard error of mean difference	0.4199
95% CI	-0.2251 to 1.5894
Test statistic t	1.624
Degrees of Freedom (DF)	13
Two-tailed probability	P = 0.1283

Differences

Shapiro-Wilk test for Normal distribution of differences	W=0.8601 reject Normality (P=0.0305)
--	---

Paired samples t-test

Sample 1	T1_Width
Sample 2	T2_Width

	Sample 1	Sample 2
Sample size	14	14
Arithmetic mean	70.1768	70.3100
95% CI for the mean	69.9107 to 70.4429	69.5541 to 71.0659
Variance	0.2124	1.7142
Standard deviation	0.4609	1.3093
Standard error of the mean	0.1232	0.3499

Paired samples t-test

Mean difference	0.1332
Standard deviation of differences	1.2508
Standard error of mean difference	0.3343
95% CI	-0.5890 to 0.8554
Test statistic t	0.399
Degrees of Freedom (DF)	13
Two-tailed probability	P = 0.6967

Differences

Shapiro-Wilk test for Normal distribution of differences	W=0.9621 accept Normality (P=0.7572)
--	---

This page is intentionally left blank.

Appendix D

Project Thesis

Note that most of the appendices mentioned in the project thesis are not included, as they are not relevant for the master's thesis.

This page is intentionally left blank.

Sampsa Matias Ilmari Kohtala

Methods for Capturing Prototypes

Further exploring and developing an evolutionary prototype repository

Project Thesis in Mechanical Engineering

Trondheim, December, 2017

Supervisor: Ph.D. Martin Steinert

Teaching Assistants: M.Sc. Jørgen Andreas Bogen Erichsen

M.Sc. Heikki Sjöman

Norwegian University of Science and Technology

Faculty of Engineering

Department of Mechanical and Industrial Engineering

 **NTNU**
Norwegian University of
Science and Technology



This page is intentionally left blank.

Abstract

This project thesis aims to explore and develop methods for capturing information and knowledge from early-stage product development. These methods are put into a physical instrument, that can be used to facilitate research on product development by capturing prototypes. Furthermore, the instrument can support designers in their projects by providing effortless documentation possibilities.

An overview of literature on prototypes and technology is provided, to better understand why and how capturing prototypes can benefit research and support designers. 3D scanning is one of the main methods addressed and tested, used for capturing and creating digital models of prototypes.

The task has been carried out by applying techniques from mechanical engineering and computer science. It started in the fuzzy front end of product development, and ended in the implementation stage.

This page is intentionally left blank.

Preface

This project thesis is based on ongoing research at the TrollLabs laboratory, and aims to advance research in early-stage product development. It is a part of the specialization program in Product Development and Materials at NTNU. The project was carried out between August and December 2017.

I would like to thank Martin Steinert, Jørgen Erichsen and Heikki Sjöman for providing a very interesting and educational task, with excellent support and guidance. Special thanks to Jørgen for creating this template.

This page is intentionally left blank.

Table of contents

ABSTRACT	III
PREFACE.....	V
TABLE OF CONTENTS.....	VII
LIST OF FIGURES.....	IX
1 INTRODUCTION.....	1
2 OVERVIEW OF PROTOTYPES, DESIGN METHODOLOGY AND TECHNOLOGY.....	5
2.1 PROTOTYPE THEORY.....	5
2.1.1 <i>Definitions of prototypes</i>	5
2.1.2 <i>Types of prototypes</i>	5
2.1.3 <i>Purpose of prototypes</i>	6
2.1.4 <i>Value of prototypes</i>	7
2.1.5 <i>Measuring prototypes</i>	7
2.1.6 <i>The Hunter-Gatherer Model</i>	8
2.1.7 <i>The fuzzy front end</i>	8
2.2 TECHNOLOGY REVIEW	9
2.2.1 <i>3D scanning and reconstruction</i>	9
2.2.2 <i>Computer vision</i>	11
3 DEVELOPMENT	13
3.1 USERS.....	13
3.2 USAGE	13
3.3 PROTOTYPING AND TESTING	14
3.3.1 <i>3D scanning and reconstruction</i>	15
3.3.2 <i>OpenCV</i>	16
3.3.3 <i>Proto booth</i>	17
3.3.3.1 Modular structure.....	17
3.3.3.2 Turntable.....	18
3.3.3.3 3D scanning.....	18
3.3.3.4 Other functionality.....	18
4 THE NEW PROTOBOOTH	21
4.1 STRUCTURE.....	21
4.2 WEIGHT	21
4.3 FEEDBACK.....	22

4.4	OTHER FUNCTIONALITY	23
4.5	DIGITIZING PROTOTYPES	23
4.5.1	3D scanning	23
4.5.2	3D reconstruction.....	24
5	SYSTEM AND LIMITATIONS.....	29
5.1	IDENTIFYING USERS	29
5.2	ACCESSING THE REPOSITORY	30
5.3	3D SCANNING.....	30
6	DISCUSSION AND FURTHER WORK	31
6.1	USING 3D SCANNING IN EARLY PRODUCT DEVELOPMENT.....	31
6.2	OBJECT CLASSIFICATION	32
6.3	LEVERAGING HANDWRITTEN SKETCHES.....	32
7	CONCLUSION	33
	BIBLIOGRAPHY.....	35
	APPENDICES.....	37
	APPENDIX A: TECHNICAL DRAWINGS	

List of Figures

Figure 1: “A repository for capturing, elaborating and sharing the process and artifact output from design activity.” (Sjöman et al., 2017).....	1
Figure 2: Protobooth located at the TrollLabs laboratory in NTNU.	2
Figure 3: Multi-view stereo pipeline. Clockwise: input imagery, posed imagery, reconstructed 3D geometry, textured 3D geometry. (Furukawa & Hernández, 2015)	11
Figure 4: 3D reconstruction with VisualSfM and MeshLab.	15
Figure 5: An early low-resolution prototype of a 3D scanning setup. Turntable with a DC motor, adjustable power source and a web camera connected to a pc.	16
Figure 6: A few examples using OpenCV; face and eyes detection, background subtraction and, image classification. Matlab on the bottom for live object classification from live video recording.	17
Figure 7: Scale prototype, using a 10kg load cell, HX711 amplifier and Arduino UNO.....	19
Figure 8: Protobooth v2.....	21
Figure 9: Load cell setup.	22
Figure 10: Displacement and stress from linear analysis of a load cell bracket, with a 50kg load. Deformations are scaled by a factor of four in the illustrations.	22
Figure 11: Adjustable camera arm, with buttons for choosing program and adjusting focus.	24
Figure 12: 3D reconstruction pipeline.	25
Figure 13: Processing time compared to number of images used.	26
Figure 14: Resulting mesh. Starting from the left, 18, 22, 36, 51 and 75 images used.	26
Figure 15: QR-link to sketchfab.com for viewing the tape measure model [https://skfb.ly/6vpTv].	26
Figure 16: Comparison of the final textured mesh and the real object, with different background and lighting. Defects present on the second model are discussed in section 5.3.	27
Figure 17: Comparing the real model with a 3D scanned and printed version. The 3D model is scaled, and the background removed manually with CAD before printing.	27

This page is intentionally left blank.

1 Introduction

This thesis focuses on a system for capturing information and knowledge from early-stage product development (PD), developed by Sjöman, Erichsen, Welo, and Steinert (2017) as part of an ongoing research project at the Department of Mechanical and Industrial Engineering at NTNU. It is aimed at advancing the discovery and understanding of causalities in the early stage of PD. As a side effect of enabling better research, the system can benefit its users (later referred to as designers) by providing documentation and feedback throughout their PD process.

The system consists of a digital repository, depicted in Figure 1, for collecting, storing and sharing data from design output (prototypes), and a physical instrument called Protobooth (a photo booth for prototypes) for capturing the data. Tangible artifacts, such as written ideas, sketches and low-resolution explorative prototypes, are used as proxies for explicit design output in the repository (Sjöman et al., 2017). Designers can simply put their prototype inside the Protobooth, swipe their personal RFID access card, and the data (pictures of the prototype) is automatically sent to the repository. The repository is accessed through a web interface, where the designer can reflect and elaborate on their documented prototypes. The researcher has access to every entry, including date, time, user information and other relevant data. Thus, it is possible to study input/output (cause and effect) relations by capturing artifacts over time-series (Sjöman et al., 2017).

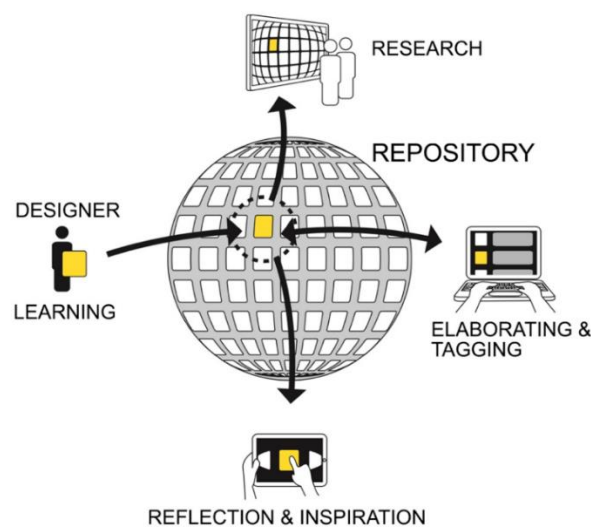


Figure 1: "A repository for capturing, elaborating and sharing the process and artifact output from design activity." (Sjöman et al., 2017)

The physical Protobooth is shown in Figure 2, located at the TrollLabs laboratory in NTNU. It consists of a wooden frame with a table-like structure surrounded by walls. Seven static web cameras are pointing towards the center of the table. An RFID card reader is placed at the corner edge, and is used to activate the prototype capturing process (taking photos), and to identify the user. Users can access the database through a web interface, where they can view and edit their data (pictures and text). Protobooth and the repository is considered an evolving prototype, meaning it is continuously being maintained, improved and tested.

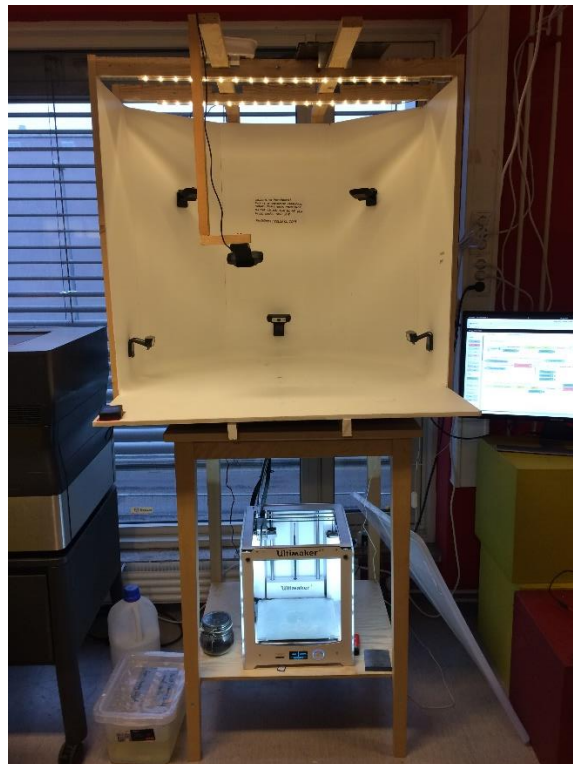


Figure 2: Protobooth located at the TrollLabs laboratory in NTNU.

The early stage of new product development can greatly influence the outcome of innovation projects (see section 2.1.6). In this stage, uncertainty is high, the solution space large and the road to success unknown. It is the “fuzzy” part of PD. In this stage, any discovery that increases the knowledge about causalities, can facilitate the evolution of new and improved methods of tackling innovation and new product development challenges. In the ever-increasing competition in the global marketplaces, not only does innovation itself create an advantage, but the time to market and the novelty of innovations can significantly determine success. The challenge is to capture and make sense of both input and output during early PD projects.

Through wayfaring (see section 2.1.6) this project thesis aims at discovering new methods of capturing prototypes, and how these methods can be applied to assist both practitioners and researchers of early-stage product development. As the repository is continuously being improved and maintained by its initial creators, the focus of this thesis is based mainly on the physical Protobooth. The main areas of focus can be summarized into the following points:

- Find ways of quantifying prototypes
- Building and testing concepts
- Create practical prototypes
- Implement findings with the existing system provided by Sjöman et al. (2017)

A literary review is conducted to establish an overview of relevant prototype theory, and to highlight the importance of documenting and researching prototypes, followed by an overview of technologies (Chapter 2). Chapter 3 describes the development process, providing practical examples and exploration of possibilities, including an overview of usage and users of the Protobooth. Chapters 4 & 5 go into details of the resulting system and its possibilities, as well as presenting the limitations of the system. A discussion on the findings and the possibilities of further work (Chapter 6) is followed by a conclusion (Chapter 7).

This page is intentionally left blank.

2 Overview of Prototypes, Design Methodology and Technology

2.1 Prototype theory

2.1.1 Definitions of prototypes

A prototype is commonly associated with physical objects representing an idea that has the potential to become a fully functional product within a specific application. Ulrich and Eppinger (2012) define prototype as “an approximation of the product along one or more dimensions of interest”. Here, the ‘product’ is the desired result of a PD process, and dimensions of interest are the characteristics and functions of the product. This definition of a prototype embraces more diverse forms of prototypes, as the characteristics of a product can be drawn and the functions can be simulated, thereby including prototypes that are not restricted to physical objects. Additionally, they define prototyping as the process of developing such an approximation of the product.

Lim, Stolterman, and Tenenberg (2008) define prototypes as “the means by which designers organically and evolutionarily learn, discover, generate, and refine designs. They are design-thinking enablers deeply embedded and immersed in design practice and not just tools for evaluating or proving successes or failures of design outcomes”. A prototype can simply be described as a tool that aids the creation and ideation of new potential products.

2.1.2 Types of prototypes

Ulrich and Eppinger classify two dimensions of prototypes. The first is the the degree to which a prototype is physical compared to analytical. Tangible artifacts, such as handmade approximations and proof-of-concept prototypes, are considered physical prototypes, while analytical prototypes are nontangible, including mathematical and visual models that can be analyzed. The second dimension is the measure of how comprehensive, as opposed to focused, a prototype is. Comprehensive prototypes will have many qualities of the finished product. Focused prototypes have less qualities, and are often used for testing elementary features. They further categorize prototypes into alpha-, beta- and preproduction-prototypes, that works as milestones in the PD cycle.

As discussed by Lim et al. (2008), there have been many attempts to identify different types of prototypes, however, the lack of a fundamental prototype definition makes it difficult to categorize them. Consequently, they developed the anatomy of prototypes, and proposed the idea of two key aspects, or dimensions; prototypes as filters and prototypes as manifestations of design ideas. The filtering dimension includes appearance, data, functionality, interactivity, and spatial structure, relating to what the designer tries to represent in a prototype. The core aspects of the manifestation dimension are materials, resolution and scope. Although filters and manifestations are not two different types of prototypes, they represent characteristics of a prototype and enables us to differentiate between the physical prototype and the idea behind it.

When the speed of learning is important, it is convenient to create low resolution prototypes (Leifer & Steinert, 2011) which can be created quickly and with simple materials such as cardboard and tape. High resolution prototypes require more resources, time, and money, and are usually created closer to the end of the PD process.

2.1.3 Purpose of prototypes

The most commonly defined purpose of a prototype is to verify and validate an idea. To verify an idea is to test it so that the feasibility of the idea requirements can be determined. To validate an idea is to test how feasible the initial requirements are. An idea can be prototyped to verify its possibilities, while validating the idea itself.

According to Ulrich and Eppinger (2012), prototypes have four purposes: learning, communication, integration and milestone. The degree of each purpose can vary from prototype to prototype. Learning from a prototype can be accomplished through testing, while simultaneously providing alternatives for communication. A new prototype can be produced as a milestone, demonstrating a new level of functionality based on prior learnings.

Lim et al. (2008) argues that the purpose of prototypes is “to find the manifestation that, in its simplest form, will filter the qualities in which the designer is interested without distorting the understanding of the whole”.

Through studying real cases from automotive industries, Elverum and Welo (2016) discovered how low-fidelity prototypes were used to enable multiple tests and ‘lessons learned’, without committing and locking into specific solutions early on. In some cases,

high-fidelity prototypes were created to secure contracts with Original Equipment Manufacturers. In addition, rapid prototyping (3D printing) were used to support the development process, by replacing components with long lead time, and for testing components to get a more realistic perspective.

The core purpose of a prototype is to generate and express knowledge.

2.1.4 Value of prototypes

A prototype that fulfils a purpose, as described in section 2.1.3, can be considered a valuable prototype. In this sense, having knowledge about the purpose of a prototype is paramount for generating value. However, creating a prototype without a clear and defined purpose, can still generate value, by uncovering unknown unknowns. The value of a prototype should therefore be based on the knowledge generated through the process of producing it and/or the end result, and not solely on the degree to which predefined purposes are accomplished. There will be a certain amount of risk generating a prototype without any clearly defined purpose, but in many cases, it is better than nothing (depending on the resources and time required to produce the prototype).

In the development process, the value of a prototype can be divided into three main categories; process, artifact and experiment (Elverum & Welo, 2016). In the process of making a prototype (prototyping) value is generated through ‘learning by doing’. When an artifact (prototype) can be used to facilitate communication and share knowledge, value is embedded in the artifact. In this case, the data from experimentation and process is not directly adding value to the prototype. Experimenting with a prototype can generate new knowledge. It is then the testing, and not the prototype itself, that is creating value.

A proposed definition on the value of a prototype: the degree to which knowledge, generated from the prototype, is applicable and reused later in the development process and/or the final product.

2.1.5 Measuring prototypes

Based on the previous subsections regarding prototypes, it is argued that measuring the value of a prototype can enhance the learnings from prototypes and prototyping in PD. Obtaining the value of different prototypes will benefit research, as the reason for differences and variation in the value of prototypes can be compared and explored.

Based on the proposed definition of value, prototypes in PD should be continuously documented with short time increments to provide data of the prototypes with ‘a common thread’ following it to the end of product release. Common threads that can be traced from the final product to the initial prototype, can be used to determine the value of the prototype by studying the documented data. It is therefore essential that prototypes are documented well, as time from start to finish of a PD process can last from weeks to years. Good research requires flawless documentation.

Creating a tool for capturing prototypes (design output) and prototyping (process/activities, input), and a system for managing and storing the data, which is nonintrusive and time-efficient for both the designer and researcher, can benefit such an attempt.

2.1.6 The Hunter-Gatherer Model

In the Hunter-Gatherer Model (Steinert & Leifer, 2012), the hunters (designers) use wayfaring skills to find their the way in a vast solution space, through exploration, prototyping and abduction, while the gatherers validate and optimize theories and frameworks. It is by no means a fixed model with steps to follow, but rather a personal and context dependent pathway alternative (Steinert & Leifer, 2012). It is based on small, agile and divers teams, that together find their way in uncertainty, to discover unknown unknowns, and set the course towards the next big idea. In the last step, when the hunt is over, optimization, manufacturing and other engineering routines are followed through.

In the early stage of new product development, low resolution prototypes can save both time and resources, and will allow the creation of more alternatives with relative ease (Leifer & Steinert, 2011). Fast iterations is key to generating radical innovations and increasing the chance of a break through. Leifer and Steinert (2011) advice project teams to stay in the early phase for more than a third of the available project time. Once a prototype has a certain level of resolution, design changes tend to become incremental.

2.1.7 The fuzzy front end

The early stage, or “fuzzy front end”, of the innovation process can greatly influence quality, costs and timing (Herstatt & Verworn, 2001). Most of the concepts are generated, customer needs found, and technologies explored during the fuzzy front end, that will ultimately decide the course of the innovation project. This stage will determine the level of radicalness of the final solution (Leifer & Steinert, 2011). As the cost of change is low,

and degrees of freedom are high, creating and testing prototypes rapidly can enhance knowledge and allow for better decisions in the future. Due to the front end naturally being the most “fuzzy” process within PD, there are challenges in discovering how this stage can be influenced to maximize the outcome. It also creates challenges for researching such effects, as the process is often nonlinear and tacit.

2.2 Technology review

2.2.1 3D scanning and reconstruction

Shape and appearance related data from real objects can be collected and analyzed with 3D scanning technologies, and then processed through surface reconstruction algorithms to create a digitalized 3D model of the object. These techniques are being applied in many different fields, ranging from manufacturing and quality control to architecture and the film industry.

Data in the form of point coordinates (x, y and z) from an object are collected to form a point cloud, which can be further processed to reconstruct surfaces and create various kinds of meshes. Color and texture can be applied from photographs to add more details.

Coordinate measuring machines (CMM) are called contact scanners. They have a probe (Pettersson, 2009), that can measure, with high accuracy, the coordinate at the point of contact with an object. All measured points can be collected in a coordinate system, forming a point cloud. CMM are made of highly accurate components, and are either automatic or manually controlled, which makes them expensive and slow.

Several non-contact scanners exist, which can be further divided into active and passive scanners. Active scanners emit radiation or light on the object and subsequently measure reflected or penetrated radiation to calculate point coordinates. Passive scanners use similar approaches, but without emitting any kind of radiation.

Laser scanners use the time of flight principle to estimate the distance between the transmitter and a point of an object (Boehler & Marbs, 2002). Triangulation is used to calculate the distance between measured points. Up to thousands of points are collected during a 3D scan, forming a point cloud. Laser scanners can be used on large structures, as they are very accurate at both short and long distances. They are somewhat slow due to many points being measured one at a time.

Structured-light scanners, such as the HP 3D Structured Light Scanner ("HP 3D Scan.," 2017), project a grid onto the object, while a camera from another angle captures the deformed patterns of the grid to calculate the position of several points. Compared to CMM and laser, structured-light scanners can obtain many points simultaneously, thus reducing scanning time without any loss of accuracy. However, the setup is more complex and time consuming.

Photogrammetry is defined by Schenk (2005) as “the science of obtaining reliable information about the properties of surfaces and objects without physical contact with the objects, and of measuring and interpreting this information.”, and is based on photographic images. It enables point clouds to be generated with multiple photographs, which can be achieved using only a single camera and software. The distance between points must be scaled by a known distance, since a single camera cannot make accurate measurements alone. The process of reconstructing a 3D model from photographs is highly dependent on algorithms and software, while the accuracy and end result can be affected by image quality such as lighting, number of pixels, number of images and exposure. It is considered a computer vision (CV) problem (Furukawa & Hernández, 2015).

According to Furukawa and Hernández (2015), expensive laser range sensor systems are being replaced by image based techniques, due to the success of multi-view stereo (MVS) technologies. MVS is an algorithm that takes images as input, and creates a 3D model as output. The generic pipeline for MVS is shown in Figure 3. The technology is crucial for digital mapping, where for example Google use it to create 3D models of cities (Furukawa & Hernández, 2015). For smaller scale applications, such as reconstructing the human face and small artifacts, film and gaming industries are active users. MVS is dependent on other algorithms, including Structure from Motion (SfM), which computes camera parameters.

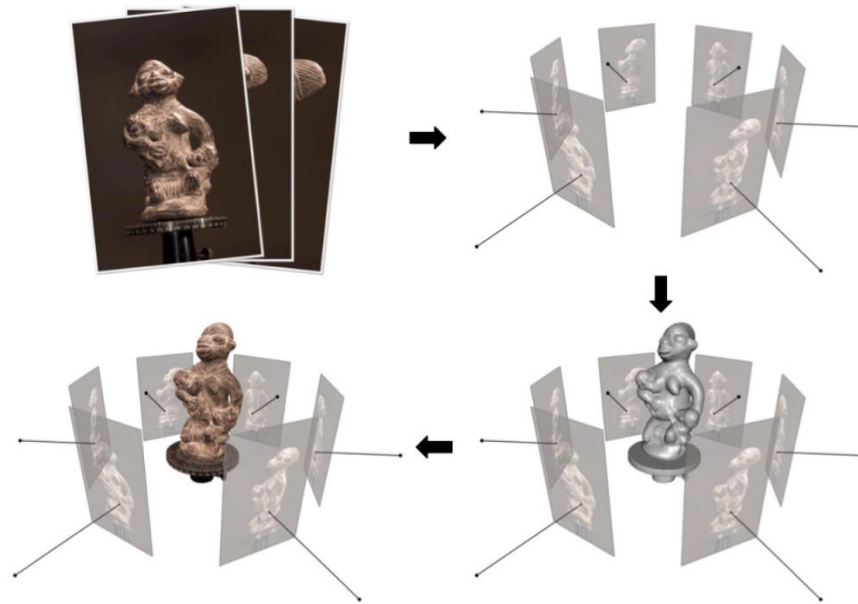


Figure 3: Multi-view stereo pipeline. Clockwise: input imagery, posed imagery, reconstructed 3D geometry, textured 3D geometry. (Furukawa & Hernández, 2015)

2.2.2 Computer vision

Open Source Computer Vision Library (OpenCV) is a free computer vision and machine learning software library ("OpenCV about," 2017). It contains more than 2500, both classic and state-of-the-art, optimized algorithms. They can be used for many different applications, including face recognition, object detection, movement tracking and classifying objects. OpenCV has interfaces for C++, C, Python, Java and Matlab, and supports Windows, Linux, Android and Mac OS.

This page is intentionally left blank.

3 Development

3.1 Users

From the researchers' perspective, the main users of Protoboosth are early phase product developers, such as R&D engineers in companies, and students of mechanical engineering. The goal is to generate value (information from early-stage product development), by selling the product (Protoboosth and digital repository). With the researcher as the user, the aim of the system is to handle and ease the understanding of uncertainty, ambiguous information, and vast solution spaces in early-stage product development, thus enabling more research potential and challenges (Sjöman et al., 2017).

Protoboosth is currently in use by Laerdal Medical, an international manufacturing and developing company for medical training products, and students and practitioners of product development related engineering in a research laboratory at NTNU.

3.2 Usage

Sjöman et al. (2017) argues that capturing output with short time increments is desired for research. Imposing rules for designers to use the repository can increase usage and data generation, but on the other hand be daunting and provoke less valuable utilization of the system. Making it nonintrusive and straightforward is thus a good approach, benefiting the user while generating valuable insights and data for the researcher.

As discussed in section 2.1.1, Lim et al. (2008) define prototypes, not only as tools for evaluating the success or failure of design outcome, but as design-thinking enablers. By capturing prototypes, the user can benefit from being able to revisit and reflect on past learnings and discovery, thus promoting new innovative ideas. Documenting work is often necessary, and an obvious benefit for the user. Although there exist many methods for individuals to capture (e.g. mobile phone camera) and manage their prototypes, the effect of capturing through an organized and structured manner can ease the process, especially for the researcher. Again, the importance of a nonintrusive system is paramount.

Lowering the threshold for using the repository is mainly achieved by Protoboosth, with its intuitive design and RFID activation, which is both simple and time efficient. However, even simpler methods for activating Protoboosth can be applied, with the use of computer vision.

From Laerdal Medical, most feedback has been on issues with connections and communication with the repository. These issues are handled by the creators of the repository, and is not the focus of this project thesis.

By having experience using Protobooth during other projects at NTNU, and observing students, some needs have been discovered. Some argue that the quality, combined with the constrained angles and positions of the cameras, prevent them from seeing the advantage of using Protobooth compared to conventional methods. Some users have even put their prototype into Protobooth due to the background and lighting conditions, only to take pictures with their mobile phones. Some users have expressed a lack of benefits from using the repository, wanting it to be more of a tool that can aid their prototyping process directly, and additionally provide more data from their prototype. As the key factor for the researcher is to acquire data, adding more ways of capturing prototypes will benefit both users.

3.3 Prototyping and testing

By utilizing wayfaring, as described in Section 2.1.6, several methods for quantifying prototypes were found and tested quickly. Both low and high resolution prototypes has been created, for testing concepts and creating useful products. One crucial factor for being able to generate prototypes in different levels of quality and resolution is the design space. Having easy access to machines, tools and materials, increases both the speed and ability to generate prototypes. However, when being aware that a prototype can be produced in high quality, it is sometimes tempting to use more time on refining the idea before using energy to create it. The biggest issue with such an approach, is that design changes tend to become incremental, as discussed in Section 2.1.6.

Information technology is undoubtedly an area that has many possibilities for improving Protobooth, both in terms of infrastructure and new ways of capturing prototypes. Because of experience and familiarity, C++ has been used as the programming language, using a Windows 10 operating system. OpenCV is the main library that is used for exploration, testing and creating prototypes within computer vision applications.

3.3.1 3D scanning and reconstruction

From reviewing 3D scanning technologies (section 2.2.1), it is evident that combining photogrammetry and computer vision algorithms will provide the most flexibility, while being the most available, inexpensive and simple method to incorporate with Protobooth.

The very first 3D scan was done with a free iPhone app called Trnio. It provided proof of the capability of photogrammetry, representing how well it works for generating 3D models of real objects. However, it is dependent on moving a camera around the object with a certain motion, and uses remote processing through a cloud.

An open source GUI application for SfM, called VisualSfM, is created by Wu (2013). It takes multiple images as input, uses an image matching algorithm to triangulate features, eventually generating a point cloud. It can generate a dense point cloud as well, by utilizing a Patch-based Multi-view Stereo (PMVS) software by Furukawa and Ponce (2010). The steps are illustrated in Figure 4, including step five in which a mesh has been generated based on the dense point cloud, using an open source 3D mesh editing and processing software called MeshLab (Cignoni et al., 2008).

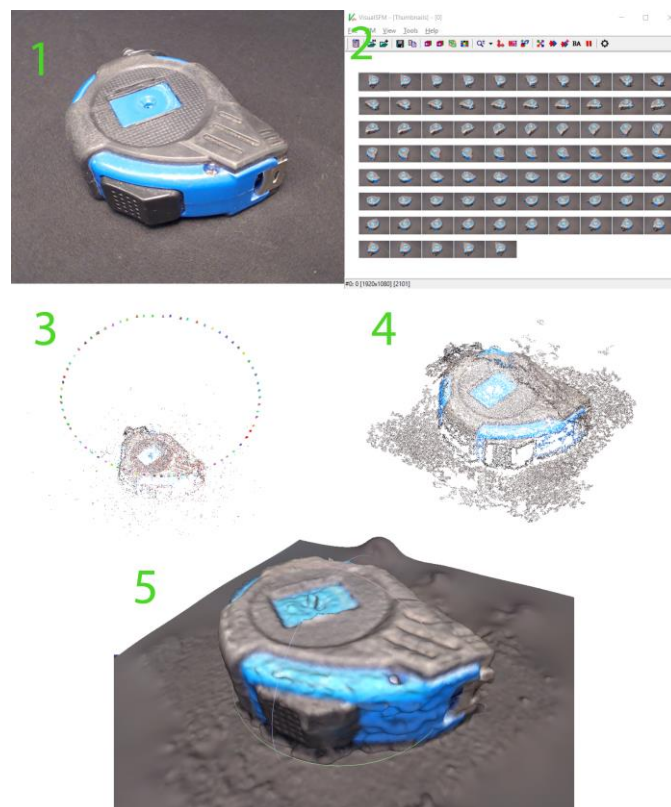


Figure 4: 3D reconstruction with VisualSfM and MeshLab.

The images used for the 3D reconstruction, shown at step one and two in Figure 4, are captured with a static 1080p web camera by Logitech. 75 images were taken of a tape measure on top of a continuously rotating turntable, shown in Figure 5. VisualSFM can automatically calculate and visualize camera positions relative to the object, shown as a ring of points in step three in Figure 4.

The final 3D model is clearly rough, and should be improved. Other algorithms are tested, and the best solution is discussed in Chapter 4.5.

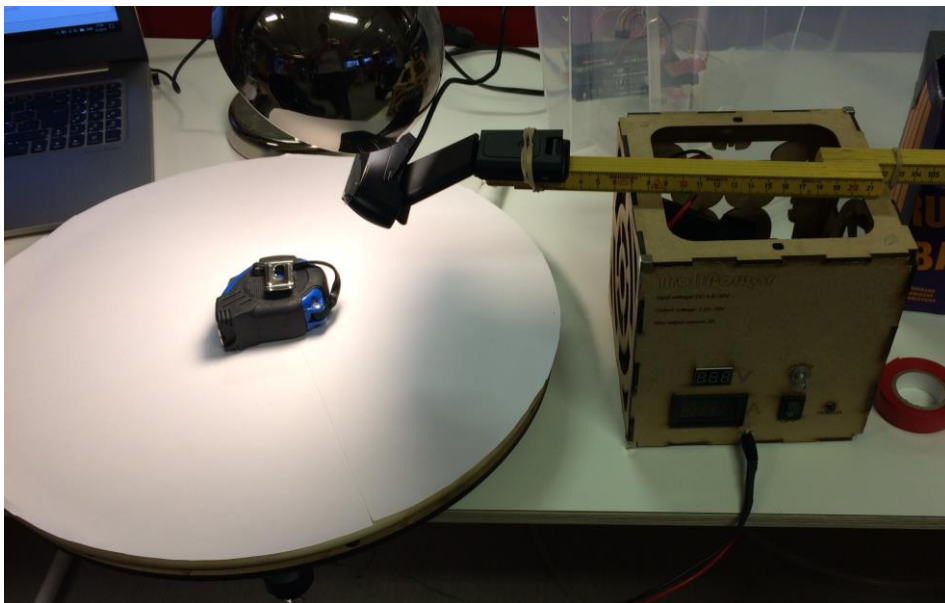


Figure 5: An early low-resolution prototype of a 3D scanning setup. Turntable with a DC motor, adjustable power source and a web camera connected to a pc.

3.3.2 OpenCV

OpenCV provides several useful tools, as mentioned in section 2.2.2. A few examples are shown below in Figure 6. On the top left, face and eyes are detected from a live video stream, using Haar cascade classifiers (Viola & Jones, 2001). Background subtraction is used on the image to the right, using the same video stream. The small region of white pixels represents a change between two or more frames. In this particular scenario it is caused by blinking an eye, as shown on the left image.

In the middle of Figure 6, a deep neural network (dnn) module, along with a trained network, is used to classify the image on the left. It is correctly identified as a redbone, with

a confidence of nearly 63.7%. The lower images are taken while running a Matlab program. It can classify objects from a live video stream, by utilizing a convolutional neural network called AlexNet. The number of objects that can be detected is dependent on the training data provided. In this case, about one thousand different objects can be identified.

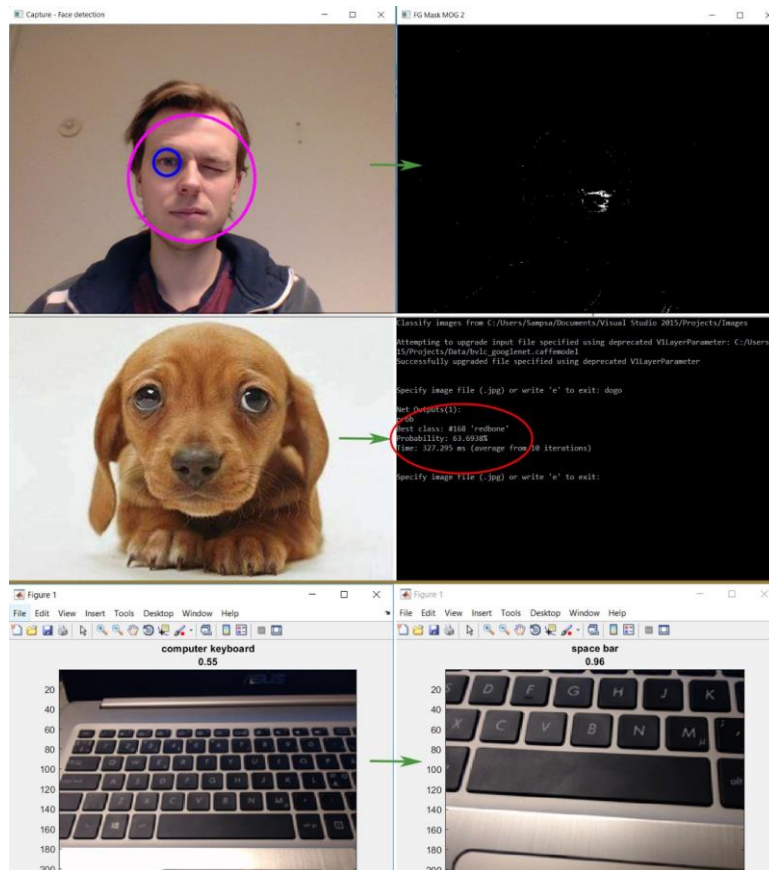


Figure 6: A few examples using OpenCV; face and eyes detection, background subtraction and, image classification. Matlab on the bottom for live object classification from live video recording.

3.3.3 Protobooth

Emphasis has been put on developing a new, high-resolution prototype of Protobooth. It is envisioned to enable effortless replication, to lower the threshold for new companies and other projects to start using the system. It should be relatively simple to assemble, while having the ability to evolve through further testing and development. Iterations along the building process were used to increase the confidence in the product.

3.3.3.1 Modular structure

To build a new Protobooth, which is high resolution, flexible and simple to rebuild, it is convenient to utilize existing modular profiles. Boch Rexroth profiles were chosen, due to

strength, availability and ease of assembly. 3D-files from their webpage ("Bosch Rexroth profile series," 2017) enabled quick and simple configurations to be generated using CAD software. The main body is kept similar to the existing Protoboosth shown in Figure 2 at the Introduction. A full CAD model of the setup was created to aid the building process, enabling design iterations, and for fitting other components. This is especially beneficial for fitting 3D printed parts.

3.3.3.2 Turntable

Early successes of 3D scanning (see section 3.3.1) using a simple turntable provided the means to incorporate a robust and precise turntable (see Appendix A). Using the bicycle steering principle as inspiration, the main body was designed and fitted to the CAD model. With access to rapid prototyping (3D printing) complexity is no longer a decisive constraint, making a computer aided approach convenient for design. Only two printed iterations were needed to find a suitable configuration, however several attempts with modelling were needed before settling on a design.

An aluminum flange and shaft were turned for transferring motion between the table and a stepper motor. Simple gears were quickly prototyped using plexiglas sheets, with the help of laser cutting.

3.3.3.3 3D scanning

To enable 3D scanning, it is necessary to place a camera at an appropriate position relative to the object. Using Rexroth profiles and rotational joint connectors, an arm can be attached to Protoboosth, with a camera at the end to be manually positioned with ease. Buttons connected to an Arduino, with serial communication to a computer, can be used for controlling the execution of the 3D scan. The computer is running a script controlling the camera, and the rate at which images are taken.

3.3.3.4 Other functionality

Having the possibility of measuring weight can by itself make the use of Protoboosth more appealing and useful. It is even more useful in combination with other measurements, such as shape and size to calculate mass, and examine possible materials.

In a load cell, force is converted into an electrical signal through strain gauges, due to the linear change in resistance from elastic deformation. This method can provide high accuracy with relatively simple functionality. The load cell in Figure 7 consist of four strain

gauges connected in a wheatstone bridge. Using more strain gauges in a certain configuration yields better accuracy, by compensating for temperature and bending sensitivity. Due to the small changes in resistance, an amplifier is needed to read the voltage difference.

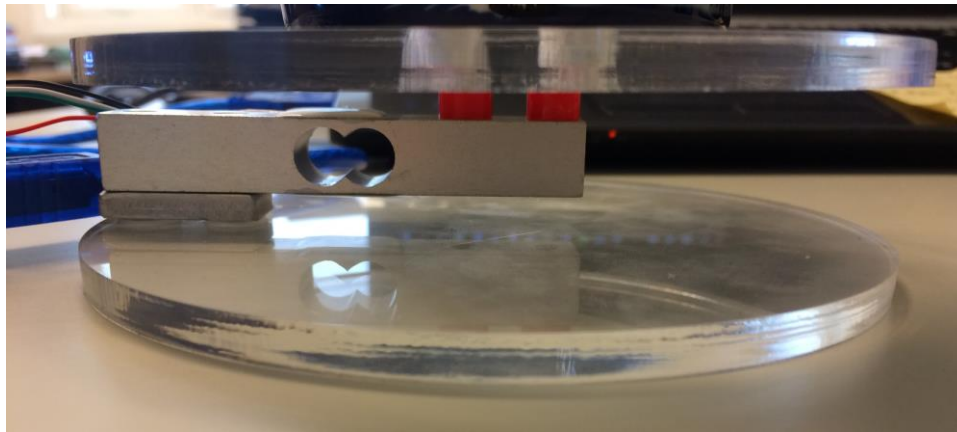


Figure 7: Scale prototype, using a 10kg load cell, HX711 amplifier and Arduino UNO.

A simple cross laser has also been tested. It is intended for assisting users in locating the point where all the cameras are facing. It also has other applications in a workshop environment, providing visual aid for evaluating straight lines, and alignment of features on objects. To keep Protoboost simple and nonintrusive for the users, different sensors can be used to activate the laser without requiring user interaction, such as range finders, ultrasonic sensors and infrared. Rather than implementing sensors for doing a simple task, the cameras used for taking pictures of prototypes can be utilized as motion detectors. OpenCV provides algorithms for using the background subtraction method, where pixels from the video frame are removed (become black) if the pixels are identical to the previous frame. The remaining pixels can be counted to determine if something has changed in the video.

A display provided by Protoboost can have many benefits, but is crucial when adjusting the camera for 3D scanning. A small projector is tested and implemented.

This page is intentionally left blank.

4 The New Protoboosth

This chapter will present the resulting system.

4.1 Structure

The frame is assembled with modular aluminum profiles, using gussets and bolted connections. Wheels provide mobility. Walls and tables are made of both plywood and medium-density fiberboard. With hinges on the table, it can be folded down by sliding the upper half back and forth, enabling Protoboosth to fit through most standard doors. Walls and shelves provide rigidity and storage space.



Figure 8: Protoboosth v2.

4.2 Weight

4mm thick aluminum brackets with 6mm spacing are used for connecting load cells to the frame, at each of the four legs. Based on linear analysis (see Figure 10), the brackets should have enough stiffness, but fail near the weight capacity of 50kg. This will protect the load

cell from damage and faulty output. 250MPa yield strength was assumed for the aluminum alloy. The load cells are connected in parallel to an amplifier, which is then read with an Arduino. Accuracy is roughly ± 1 grams.



Figure 9: Load cell setup.

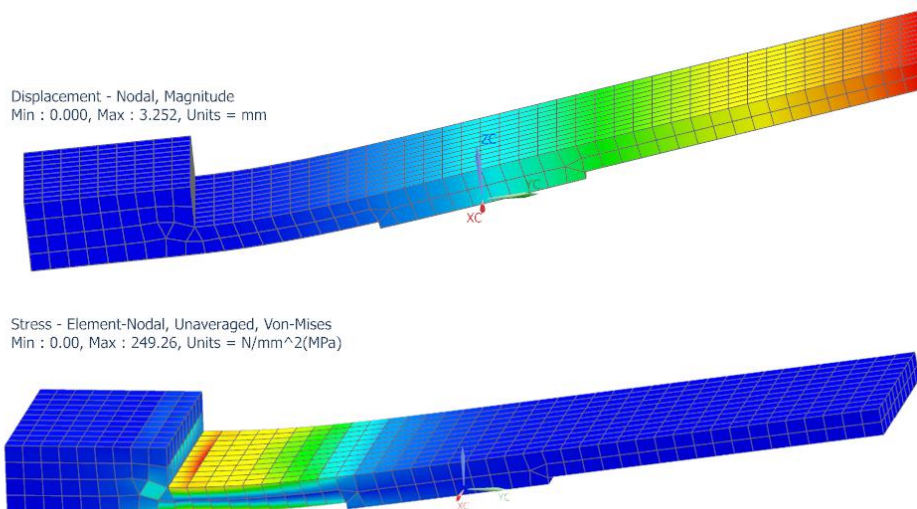


Figure 10: Displacement and stress from linear analysis of a load cell bracket, with a 50kg load. Deformations are scaled by a factor of four in the illustrations.

4.3 Feedback

A program was made with C++ that can display sensor readings from an Arduino to a window on the computer (see Appendix E). Weight and other data, along with the camera view used for capturing details and running 3D scans, can be displayed onto the wall of Protobooth through a projector connected to a computer. The pocket projector PicoPix by

Philips, used with a projection mapping software, is fixed to the frame with a 3D printed mount.

Another program (see Appendix D) was made that can detect movement from video. It can be adjusted to only detect movement within a certain area of the video frame, as well as defining the amount of activity required before being triggered to perform an action. It is intended for detecting usage of Protoboost by observing if something is put on the table. It is currently used for activating the cross laser, without requiring interaction from the user.

4.4 Other functionality

Small brackets and connectors were made with simple materials using a laser cutter and 3D printer (see Appendix G). They can be used to add more components without permanently altering the structure, while providing easy adjustment capability.

A power strip is used to provide power for the components, and a PC power supply provide DC current with different voltages and capacity. Consequently, only one power cable and one ethernet cable needs to be unplugged when the Protoboost is moved.

4.5 Digitizing prototypes

The main method explored for quantifying prototypes, and to potentially aid designers, is the use of 3D scanning to generate a digital model of real prototypes. To keep Protoboost nonintrusive and simple, photogrammetry was chosen. The data required to generate a 3D model is a set of images. The process of capturing images is simple, and is of little significance for the users with the proposed solution.

4.5.1 3D scanning

A web camera is attached to a flexible arm (see Figure 11), fixed at the top frame of Protoboost. Friction from the rotational joints can be adjusted, for both moving and holding the arm in place. Buttons are attached to the arm to provide control over capturing method, and for adjusting camera focus. The two capturing methods are 2D, for taking single detailed pictures, and 3D, for running a 3D scan.



Figure 11: Adjustable camera arm, with buttons for choosing program and adjusting focus.

The turntable consists of a 3D printed body made of PLA plastic. It is press fitted with ball bearings to make the shaft turn smoothly with both high and low weight on top. It is bolted to the frame, with adjustment possibilities. A stepper motor is attached with screws to the printed body with timing belts and gears connecting the rotor to the turntable shaft. Microstepping is used to provide smooth movement at the expense of torque. Testing has shown that torque is not decisive as long as the center of mass of the object is close to the turntable center.

4.5.2 3D reconstruction

The best results from testing photogrammetry programs are obtained using COLMAP (Schonberger & Frahm, 2016; Schönberger, Zheng, Frahm, & Pollefeys, 2016) and OpenMVS ("OpenMVS," n.d.). The steps going from a set of images to a 3D model is illustrated in Figure 12. Meshconv (Min, 2017) is only used for converting to a stereolithography (STL) file, which is convenient for CAD and 3D printing. Each program is executed through command line arguments. A C++ program was created to run each step automatically while providing control over file management (see Appendix C). This program is the basis for implementing 3D scanning with Protobooth and the repository.

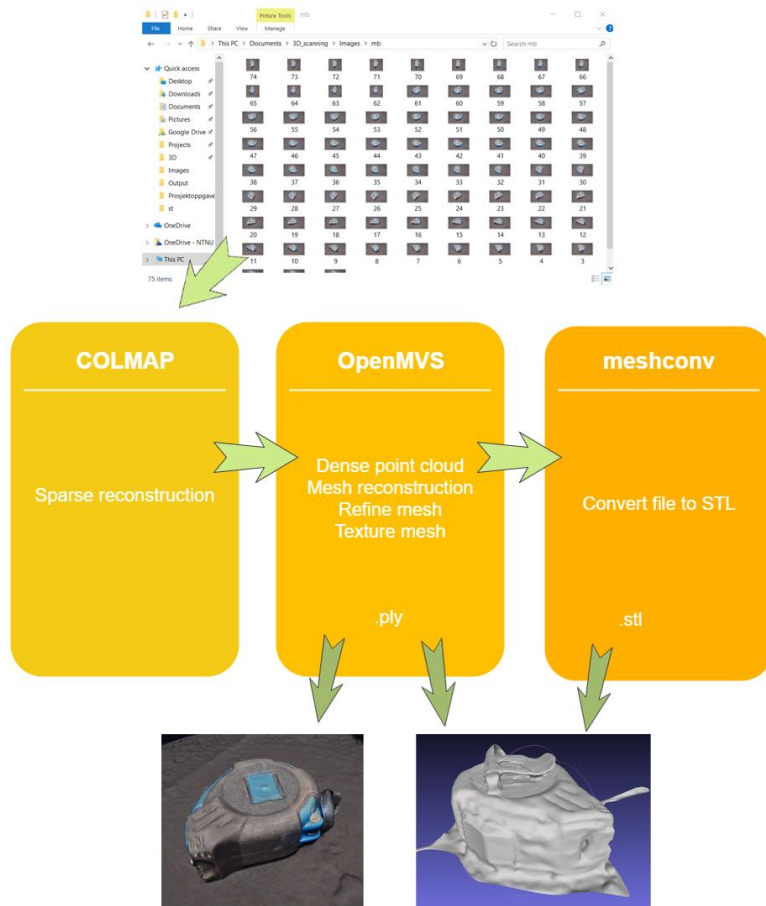


Figure 12: 3D reconstruction pipeline.

Testing is performed with an Intel® Core™ i7-7700HQ processor and 16GB RAM. Highest memory usage during reconstruction has been detected at 6.65GB. The scanning process for these tests was done with the object rotating a full round at 6.5rpm, lasting a total of 9.2 seconds. Figure 14 shows the resulting mesh for some of the tests. The graph below illustrates the difference in processing time related to the number of images used.

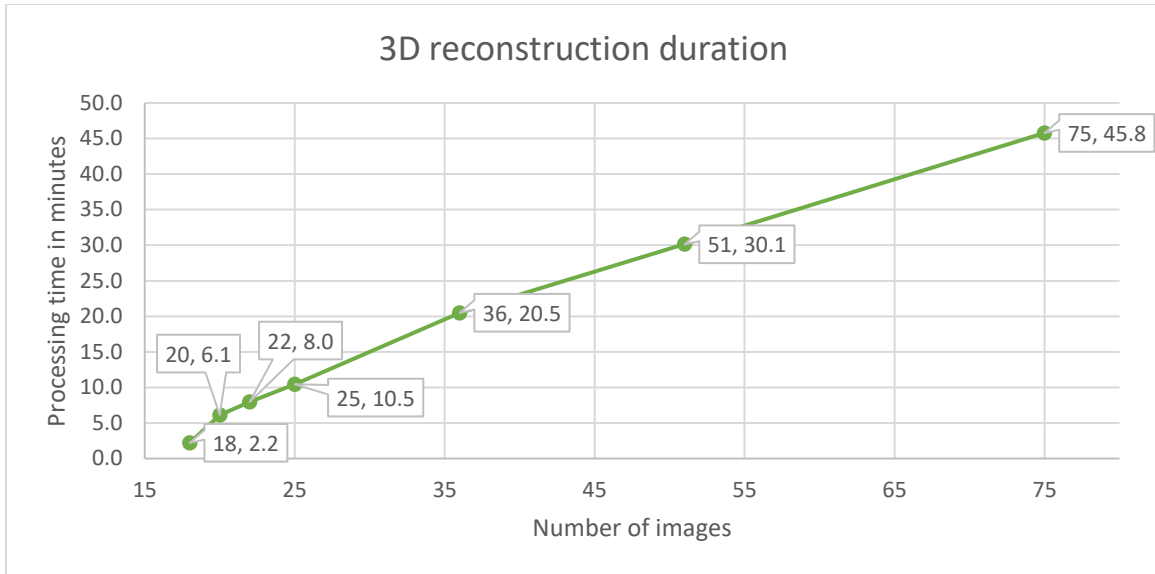


Figure 13: Processing time compared to number of images used.

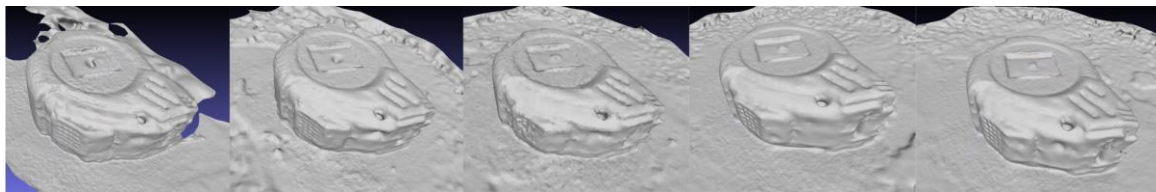


Figure 14: Resulting mesh. Starting from the left, 18, 22, 36, 51 and 75 images used.

The graph is close to linear. Processing time is thus increasing with approximately 40 seconds per image. With less than 18 images, the program produced errors and could not finish. Visually, the mesh from using 22 pictures is almost as good as the once using more, both with and without texture.

The tests are done with a black turntable, which contribute to better light distribution on the object perceived by the camera, and thus better texture on the final model. However, using a white table caused the program to ignore most of the table, reducing processing time accordingly. Comparisons of different tests are shown in Figure 16, with the real object compared with a 3D scanned and printed version in Figure 17.



Figure 15: QR-link to sketchfab.com for viewing the tape measure model [<https://skfb.ly/6vpTv>].

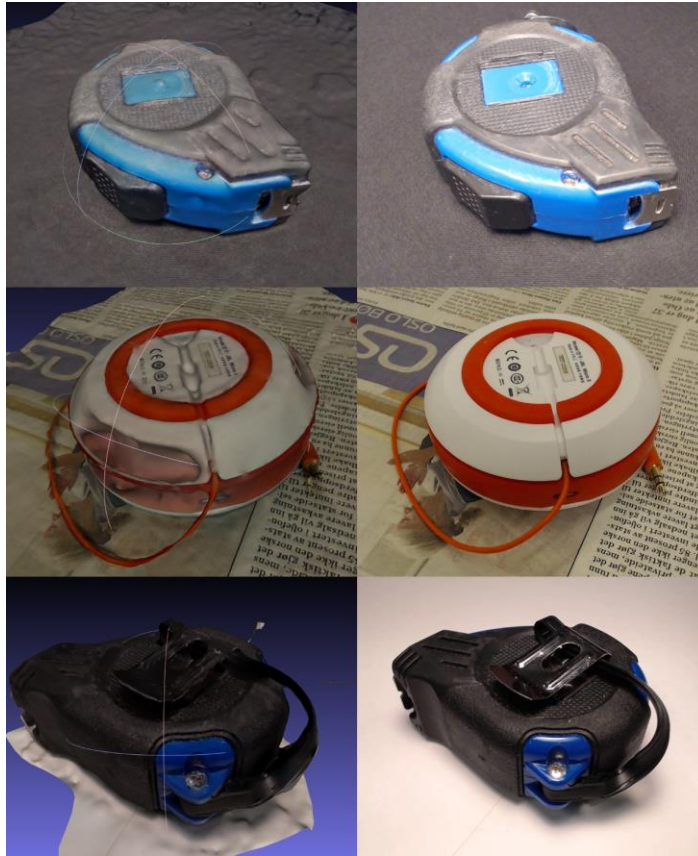


Figure 16: Comparison of the final textured mesh and the real object, with different background and lighting. Defects present on the second model are discussed in section 5.3.



Figure 17: Comparing the real model with a 3D scanned and printed version. The 3D model is scaled, and the background removed manually with CAD before printing.

This page is intentionally left blank.

5 System and Limitations

A proposed flow of operations running on Protoboosth is illustrated in Appendix F. It is not completely assembled in the real world yet. However, programs and modules are tested and ready to be implemented. With the different possibilities using Protoboosth, normal mode, single image capturing and 3D scanning, it is important to keep the process simple and nonintrusive for the users. Simple button pushes are required to choose a capturing method, and to adjust focus, otherwise the process is the same, using RFID to identify the user and activate a process. With the buttons located near the camera, and having to position the camera manually, it should be easy to understand how they work. Observing users is still necessary to discover faults in the system and to make improvements. Other solutions can easily be implemented if needed.

Protoboosth will automatically return to standby when interaction stops, letting the users only focus on what they need to do for capturing their prototype, and not have to worry about anything else. A timer is necessary to determine how long Protoboosth will stay in a certain mode without interaction before returning to standby. More comprehensive testing and observation should be conducted to determine an appropriate delay time between interactions.

It should also be tested how processing is handled when the system is utilized at maximum capacity (e.g. reconstructing a 3D model while running a new 3D scan). A queuing system for processes might be necessary.

5.1 Identifying users

Face recognition can potentially replace the need for RFID. It can also be used to determine when, and how long a user is interacting with Protoboosth, thus neglecting the delay previously mentioned. Although face recognition technology can simplify the user experience of Protoboosth, it can be subject to ethical issues. Recording and saving personal information is not something that can be handled with ease, and requires good security and trust. Monitoring people might invade their privacy. Such methods are also subject to functional creep¹.

¹ “The gradual widening of the use of a technology or system beyond the purpose for which it was originally intended, especially when this leads to potential invasion of privacy” (“function creep,” n.d.)

Face detection, demonstrated in section 3.3.2, could be an alternative for detecting interactions. However, issues with the program can result in some users not being recognized, which could potentially provoke some unnecessary complications while using a notable amount of processing capacity.

5.2 Accessing the repository

With the display provided by a pocket projector, users can potentially access the repository via Protobooth. In this case, it should only have a few possibilities for editing entries in order to keep Protobooth as simple and transparent as possible. However, it could be interesting to observe if a seemingly complex system attracts or intimidates users, and if it will encourage more or less data to be generated.

5.3 3D scanning

Furukawa and Hernández (2015) mention three main limitations of photogrammetry, or more specifically MVS algorithms: lack of texture, thin structures and non-Lambertian surfaces. An example of a problem that can occur due to lack of texture is shown in the middle object in Figure 16 (section 4.5.2). Bad lighting could also be a reason for the strange mesh present in the model. Problems with thin structures is also apparent, looking at the wire on the same model. Furukawa and Hernández (2015) provide many solutions to this problem that should be considered in the future. MVS assumes Lambertian surfaces, thus making materials such as polished metal and plexiglas a challenge for reconstruction. By controlling the lighting, which can be done in a laboratory environment, it is possible to increase the quality of these surfaces as well.

Scaling becomes an issue when dimensions of the object are of interest. It is currently done manually with a CAD or 3D printing software, by measuring the difference between two points on the real object, and then comparing it with the 3D model.

Higher accuracy is achieved with more baseline². However, for increased density and robustness, smaller baselines are desired. The typical optimal baseline is in the range 5-15 degrees, according to Furukawa and Hernández (2015). The best results discussed in section 4.5 used baselines between 4.8° (75 images) and 16.4° (22 images).

² Distance or angle between two images facing the same point, used for calculating distance with the triangulation principle.

6 Discussion and Further Work

Mostly explorative and proof of concept prototyping, and producing modules for different functionality, has been done. It remains to put everything together into a system. The remaining work can be accomplished within an effective week or two, and consists of painting the walls, fixing the lights, connecting cables, assemble programs, connect to the repository and other small adjustments.

6.1 Using 3D scanning in early product development

There are many potential benefits with using 3D scanning in the early stage of product development, for both practitioners and researchers. Although 3D scanning has been used in product development for a long time, it is usually aimed at projects where the need to model organic shapes for producing customized products is essential. Such projects often utilize expensive handheld 3D scanners, providing accuracy and speed. The aim of this thesis is on using simple and cheap methods for capturing prototypes, in which photogrammetry is suitable. The potential benefits of this method are endless.

The visual results obtained from 3D scanning during this project thesis are of high quality. It can be beneficial for remote collaboration, as the lack of good images can often distort the true perception of a prototype. Being able to move and observe prototypes with texture and color is very different compared to 2D representations. This method of showcasing output from product development might have been neglected in the past, due to 3D scanning being perceived as an expensive and time-consuming effort. Although there are limitations to the approach used in this thesis, the results prove that satisfactory results can be obtain with simple methods. Another benefit in terms of remote collaboration and perception is the ability to 3D print the prototype to make it tangible.

As discussed in section 2.1.2 and 2.1.3, (low resolution) prototypes are convenient when the speed of learning is important (Leifer & Steinert, 2011), and are used by designers to filter the qualities they are interested in (Lim et al., 2008). However, when knowledge is generated, the physical prototype itself can lose its purpose and be neglected for the remaining development process, especially if it is low resolution. In other words, knowledge generated from a prototype can easily become tacit, potentially losing some of its value for later use (see proposed definition of value in section 2.1.4). Being able to replicate lost prototypes can aid sharing and reexperiencing knowledge. This could be

accomplished by having repository of virtual reality prototypes. It will also provide researchers with more details to be explored.

In the era of digitalization, this method can potentially be used to transfer real prototypes to digital versions. The contribution of 3D scanning in product development should be explored and researched further by practical approaches.

6.2 Object classification

Object classification algorithms through OpenCV have proven useful for identifying objects, from early testing demonstrated in section 3.3.2. Further development and implementation can be tested with Protoboost, to automatically classify prototypes and categorize them by recognizing materials, electronics, components, tools used and other distinctive features.

6.3 Leveraging handwritten sketches

Another interesting but late idea emerged: the possibility to scan handwritten sketches in Protoboost and generate files that can be processed in machines (e.g. laser cutter and PCB mills). This method can be used to create simple prototypes very fast. Borders around objects can be drawn on paper to produce a part that will fit perfectly, instead of using the slow process of taking accurate measurements and using software. This method will provide knowledge, for the researcher, of the machine used in addition to the part being made.

This is a typical computer vision problem. Hough line transformation, feature matching algorithms and optical character recognition can be used for detecting lines and reading characters for defining material and machine parameters. Testing and potential implementation can be further explored.

7 Conclusion

From reviewing prototype theory, a definition on the value of prototypes has been proposed: the degree to which knowledge, generated from the prototype, is applicable and reused later in the development process and/or the final product. With the basis in this definition, it is argued that capturing and documenting prototypes can advance research on early-stage product development.

Attempts have been made to discover ways of capturing prototypes. The challenge started in the fuzzy front end of product development, and has been met through the wayfaring method. 3D scanning, weight sensors, and an improved Protobooth has been the focus after the early stage. Through iterations and proof-of-concept testing, a new Protobooth is developed, with new functions for capturing prototypes, along with the functionality of the initial Protobooth by Sjöman et al. (2017). The full system has not been completely put together, although most programs and prototypes are ready for implementation.

The possibilities of generating digital models of real prototypes, using photogrammetry and a turntable, has been explored. The approach is proven to be simple, and the results provide support for further research on the potential of using this method to aid both practitioners and researchers of early-stage product development.

This page is intentionally left blank.

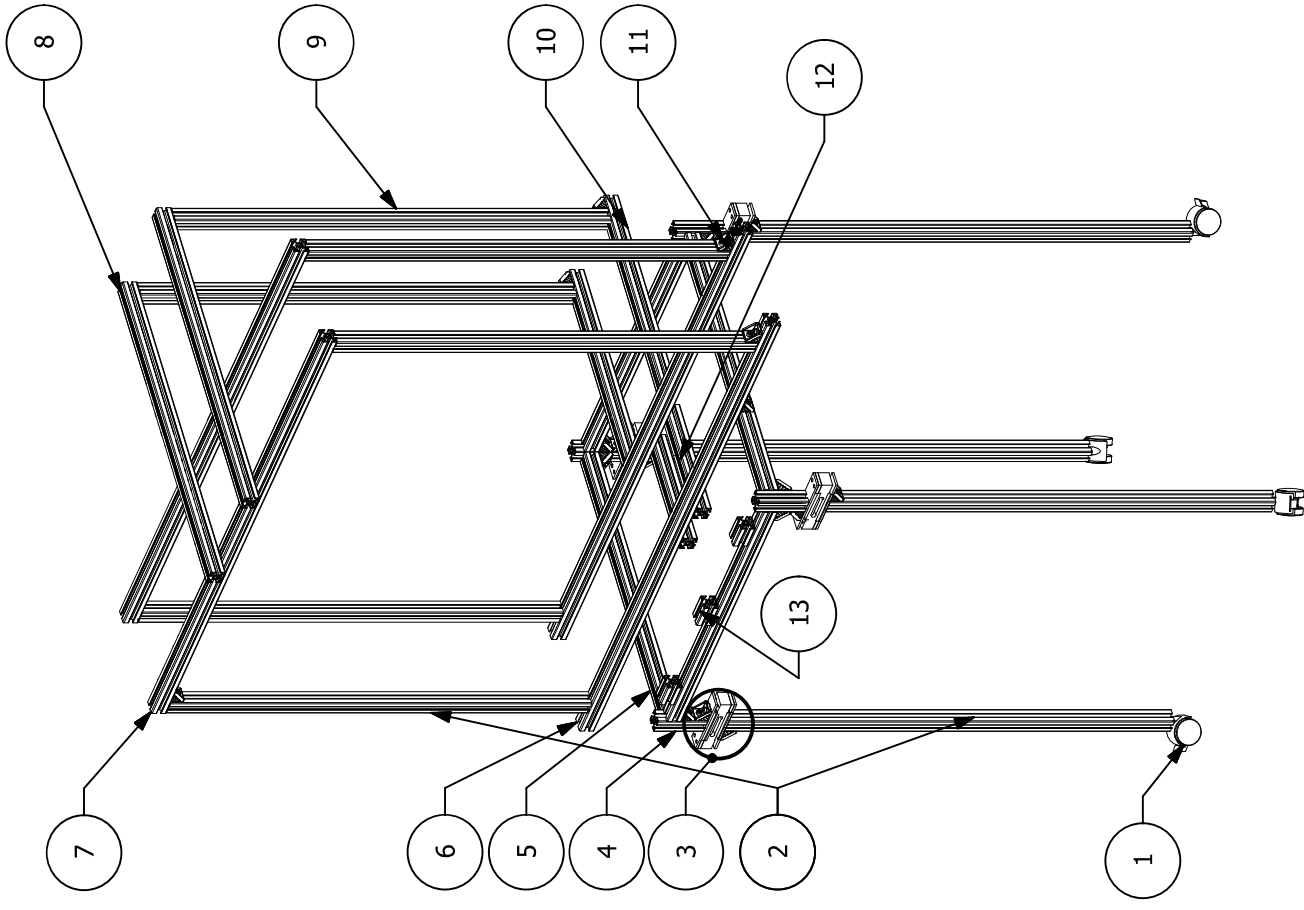
Bibliography

- Boehler, W., & Marbs, A. (2002). 3D scanning instruments. *Proceedings of the CIPA WG*, 6, 9-18.
- Bosch Rexroth profile series. (2017). Retrieved 8.12.17, from http://www13.boschrexroth-us.com/Framing_Shop/Product/Default.aspx?Group=101
- Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., & Ranzuglia, G. (2008). *Meshlab: an open-source mesh processing tool*. Paper presented at the Eurographics Italian Chapter Conference.
- Elverum, C. W., & Welo, T. (2016). Leveraging prototypes to generate value in the concept-to-production process: a qualitative study of the automotive industry. *International Journal of Production Research*, 1-13. doi:10.1080/00207543.2016.1152406
- function creep. (n.d.). *Collins English Dictionary - Complete & Unabridged 10th Edition*. Retrieved November 23, 2017, from <http://www.dictionary.com/browse/function-creep>
- Furukawa, Y., & Hernández, C. (2015). Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2), 1-148.
- Furukawa, Y., & Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8), 1362-1376.
- Herstatt, C., & Verworn, B. (2001). The 'fuzzy front end' of innovation. In *Bringing technology and innovation into the boardroom* (pp. 347-372): Springer.
- HP 3D Scan. (2017). Retrieved November 21, 2017, from <http://www8.hp.com/h20195/v2/GetDocument.aspx?docname=4AA6-9359ENW>
- Leifer, L. J., & Steinert, M. (2011). Dancing with ambiguity: Causality behavior, design thinking, and triple-loop-learning. *Information Knowledge Systems Management*, 10(1-4), 151-173.
- Lim, Y.-K., Stolterman, E., & Tenenbergs, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 15(2), 7.
- Min, P. (2017). meshconv. Retrieved, from <http://www.patrickmin.com/meshconv>
- OpenCV about. (2017). Retrieved November 22, 2017, from <https://opencv.org/about.html>
- OpenMVS. (n.d.). Retrieved, from <http://cdceacave.github.io/openMVS/>
- Pettersson, B. (2009). Coordinate measuring machine. In: Google Patents.
- Schenk, T. (2005). Introduction to photogrammetry. *The Ohio State University, Columbus*, 106.
- Schonberger, J. L., & Frahm, J.-M. (2016). *Structure-from-motion revisited*. Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.
- Schönberger, J. L., Zheng, E., Frahm, J.-M., & Pollefeys, M. (2016). *Pixelwise view selection for unstructured multi-view stereo*. Paper presented at the European Conference on Computer Vision.

- Sjöman, H., Erichsen, J. A. B., Welo, T., & Steinert, M. (2017). Effortless Capture of Design Output. 7.
- Steinert, M., & Leifer, L. J. (2012). 'Finding One's Way': Re-Discovering a Hunter-Gatherer Model based on Wayfaring. *International Journal of Engineering Education*, 28(2), 251.
- Ulrich, K. T., & Eppinger, S. D. (2012). Product design and development, 2000. *New York: MacGraw-Hill*.
- Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features*. Paper presented at the Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on.
- Wu, C. (2013). *Towards linear-time incremental structure from motion*. Paper presented at the 3DTV-Conference, 2013 International Conference on.

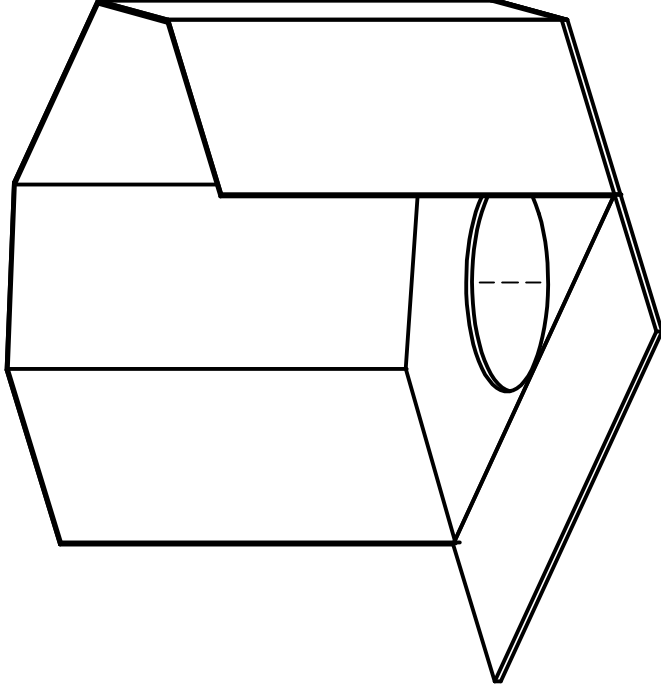
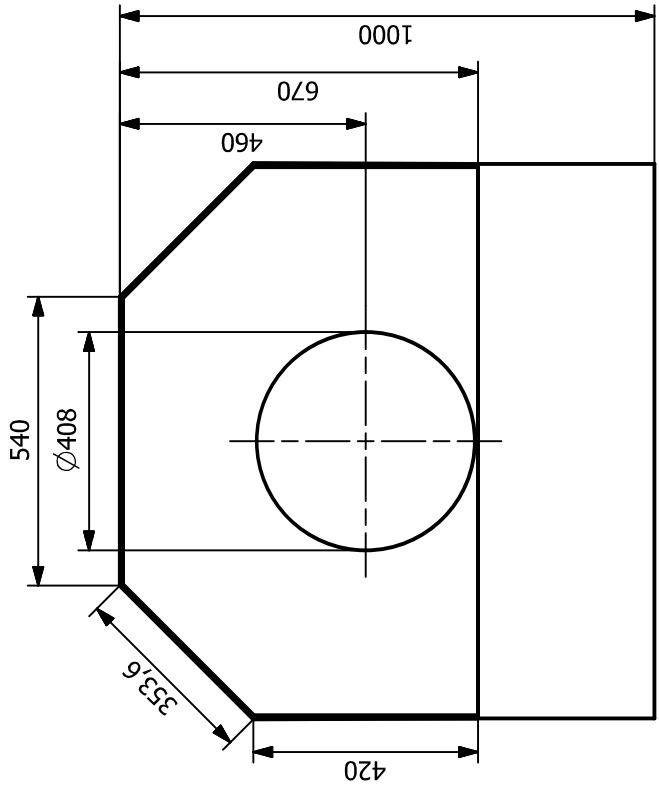
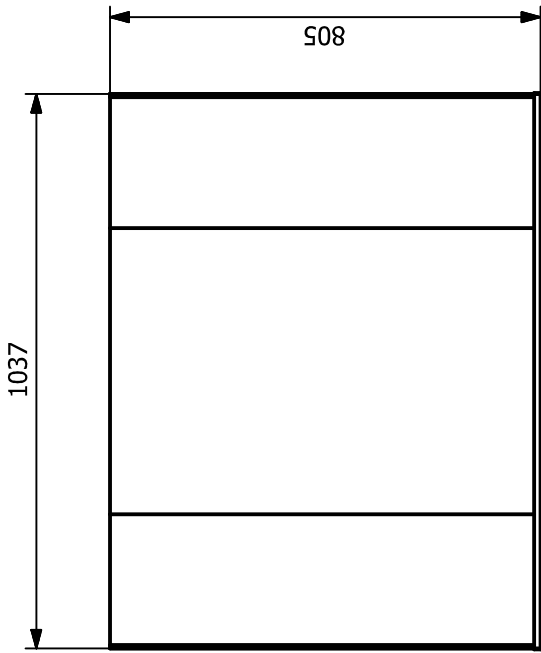
Appendix A

Technical drawings

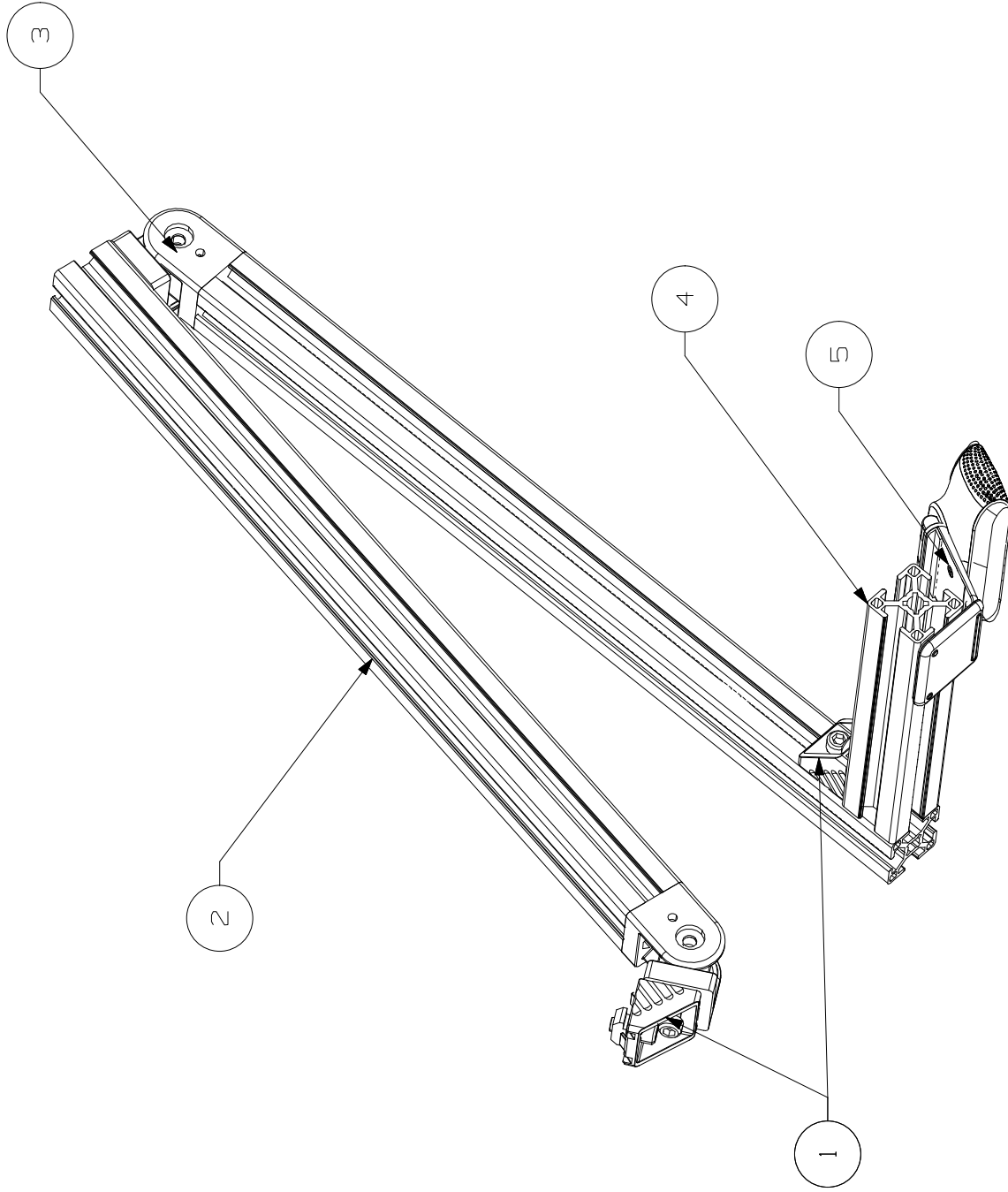


PC NO	Part name	Part ID	Quantity
1	Caster wheel	A r.30.c-M8 X o	4
2	Rexroth 30 profile	A r.30.850 al o	8
3	Load Cell assembly	D	4
4	Rexroth 30 profile	A r.30.100 al o	4
5	-	A r.30.620 al o	4
6	-	A r.30.1200 al o	2
7	-	A r.30.1100 al o	2
8	-	A r.30.700 al o	2
9	-	A r.30.880 al o	2
10	-	A r.30.450 al o	2
11	Rexroth gusset (not all in illustration)	A r.30-g st o	34
12	Rexroth 30 profile	A r.30.250 al o	2
13	-	A r.30.50 al o	3

Dato	Konstr./Tegnet	Bokk_jent	Mallestokk	NTNU - IPM
20.11.17	SMIK	Prosjektsjansetkode	1:10	
<p style="text-align: center;">Erstatning for:</p>				Erstattet av:
<p style="text-align: center;">Frame assembly</p>				Part ID: A
Henvi/sjening:				Beregning:



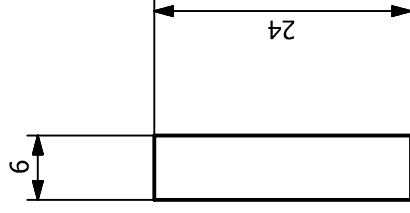
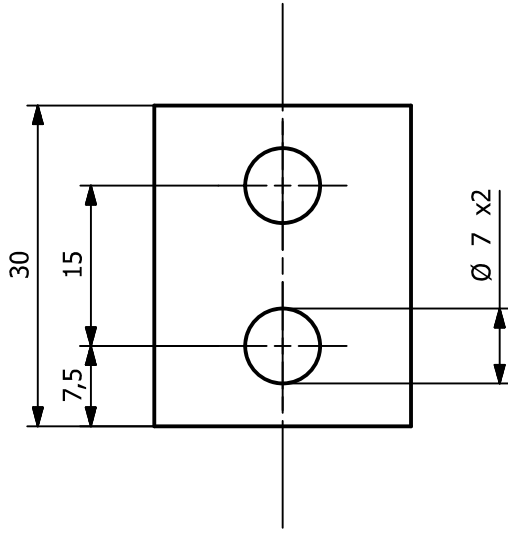
Dato	Konstr./Tegnet	Bokk_jent	Målestokk	NTNU - IPM	
				Erstattet av:	Erstattet av:
20.11.17	SMJK	Prosjektansvar	1:10	Part ID: B	
Table and walls				Beregning:	
				Henvi sning:	



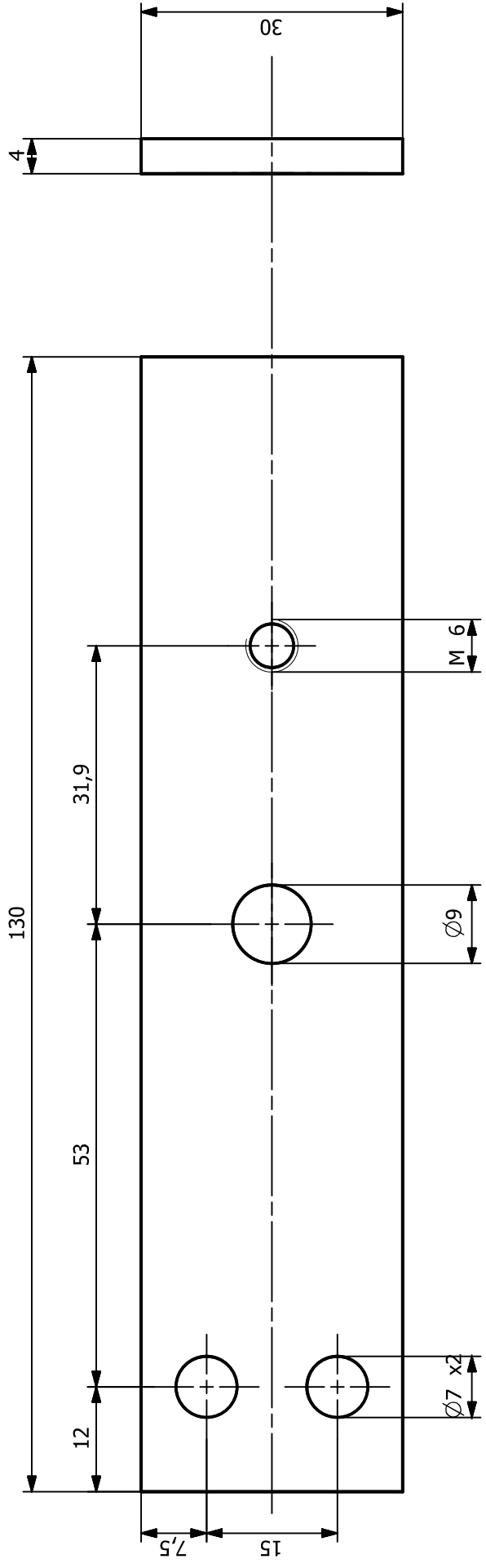
PC NO	Part name	Part ID	Quantity
1	Rexroth gusset	C r.30.g st o	2
2	Rexroth 30 profile	C r.30.400 al o	2
3	Rexroth multi angle	C r.30.ma X o	2
4	Rexroth 30 profile	C r.30.150 al o	1
5	Camera		1

Dato	Konstr./Tegnet	Mod/lestokk	NTNU - IPM
	20.11.17	SMIK	
Erstatter: av:		Erstatter: av:	
Henvi/sning:		Part ID: C	
Beregning:			

Adjustable camera arm assembly

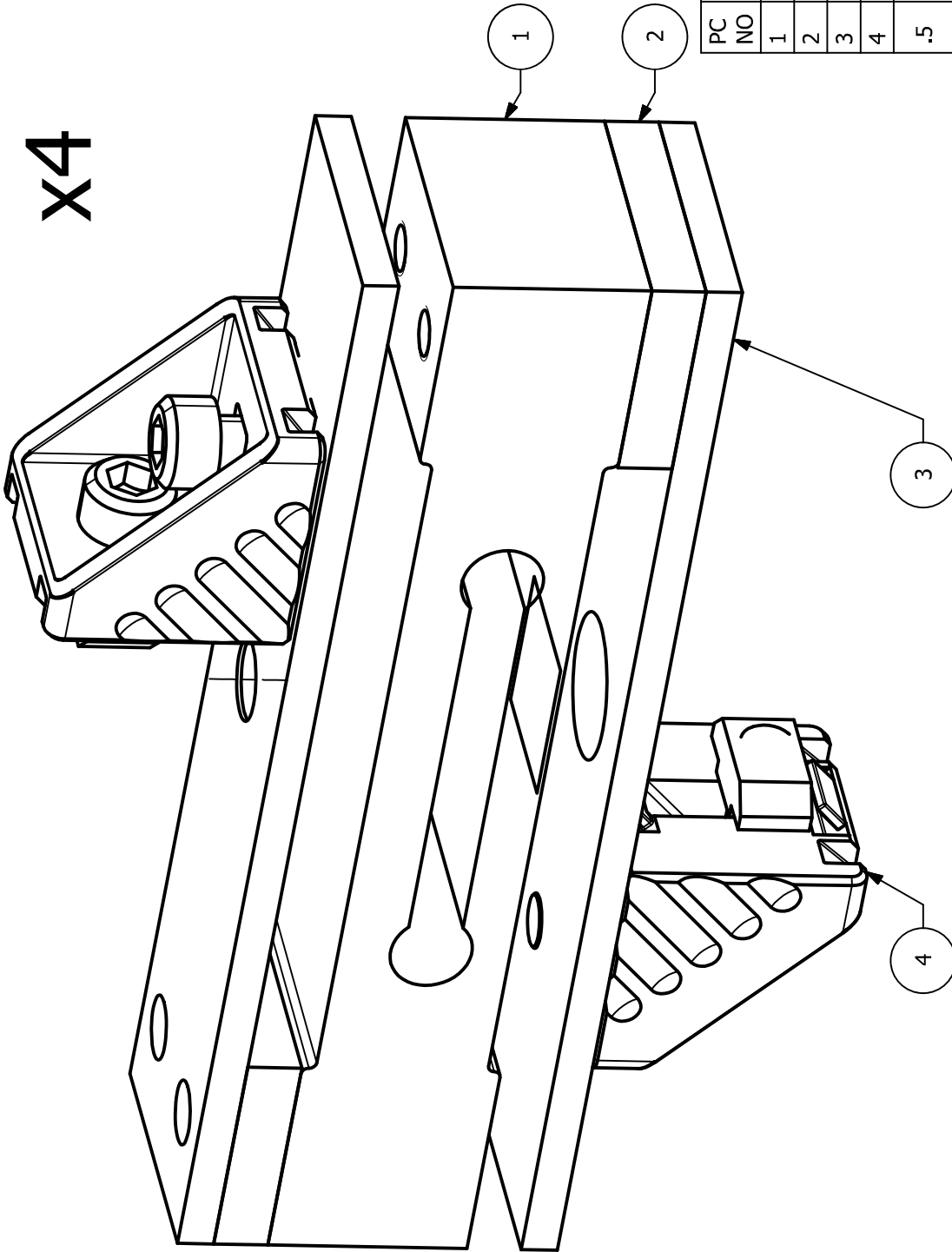


Dato	Konstr./Tegnet	Bokk_jent	Modlestokk	NTNU - IPM
	20.11.17	SMIK		
Projeksjonsmetode		Erstatning for:	Erstattet av:	Part ID: D b.30-24-6 al m
Henvi/sning:			Beregning:	
Load cell bracket short				



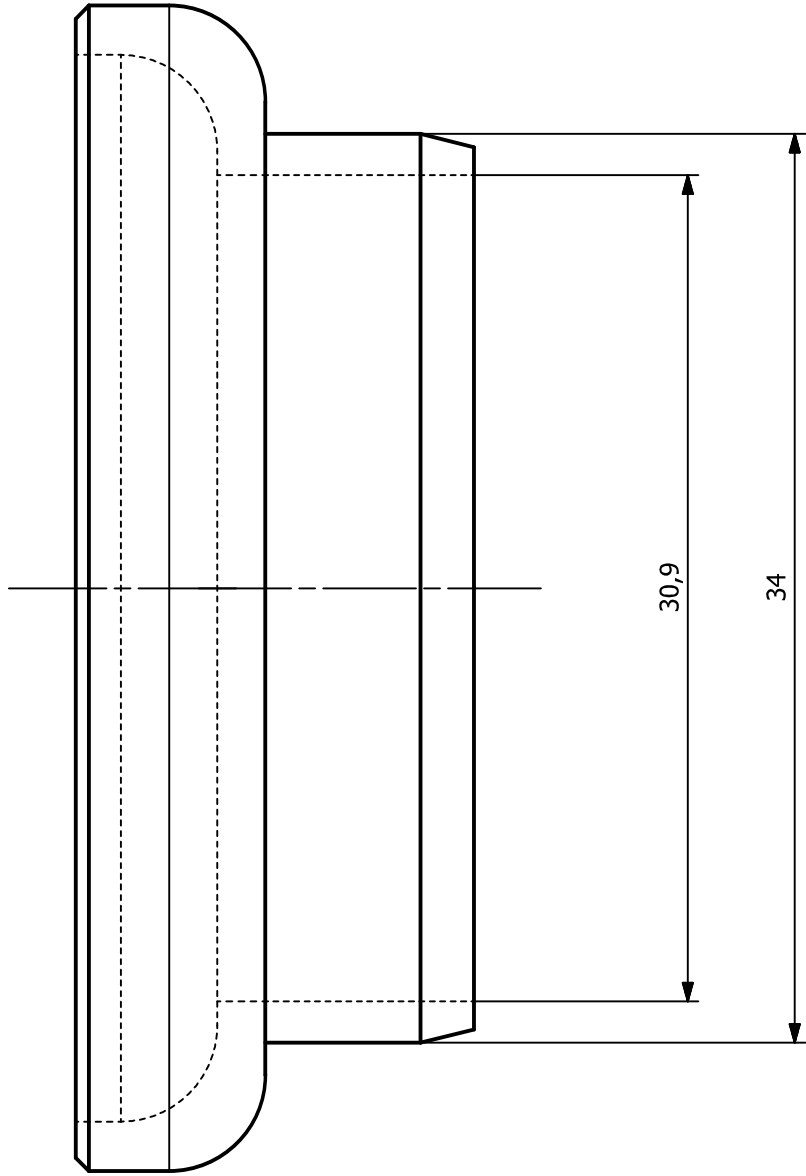
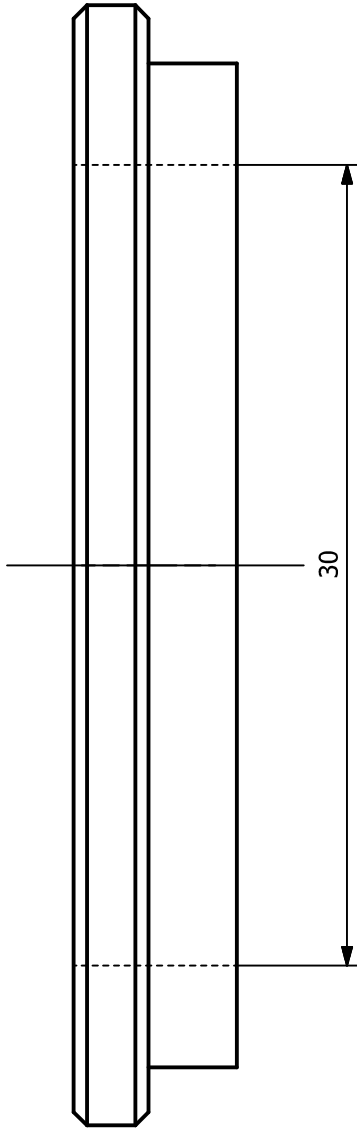
Date	Konstr./Tegnet	Godkjent	Modulstokk	NTNU - IPM
20.11.17	SMIK	Projektsjansetode	2:1	Erstatning for:
Load cell bracket				Erstattet av:
Ø9 hole countersunk				Part ID: D b.130-30-4 al m
Henvi/sning:			Beregning:	

X4

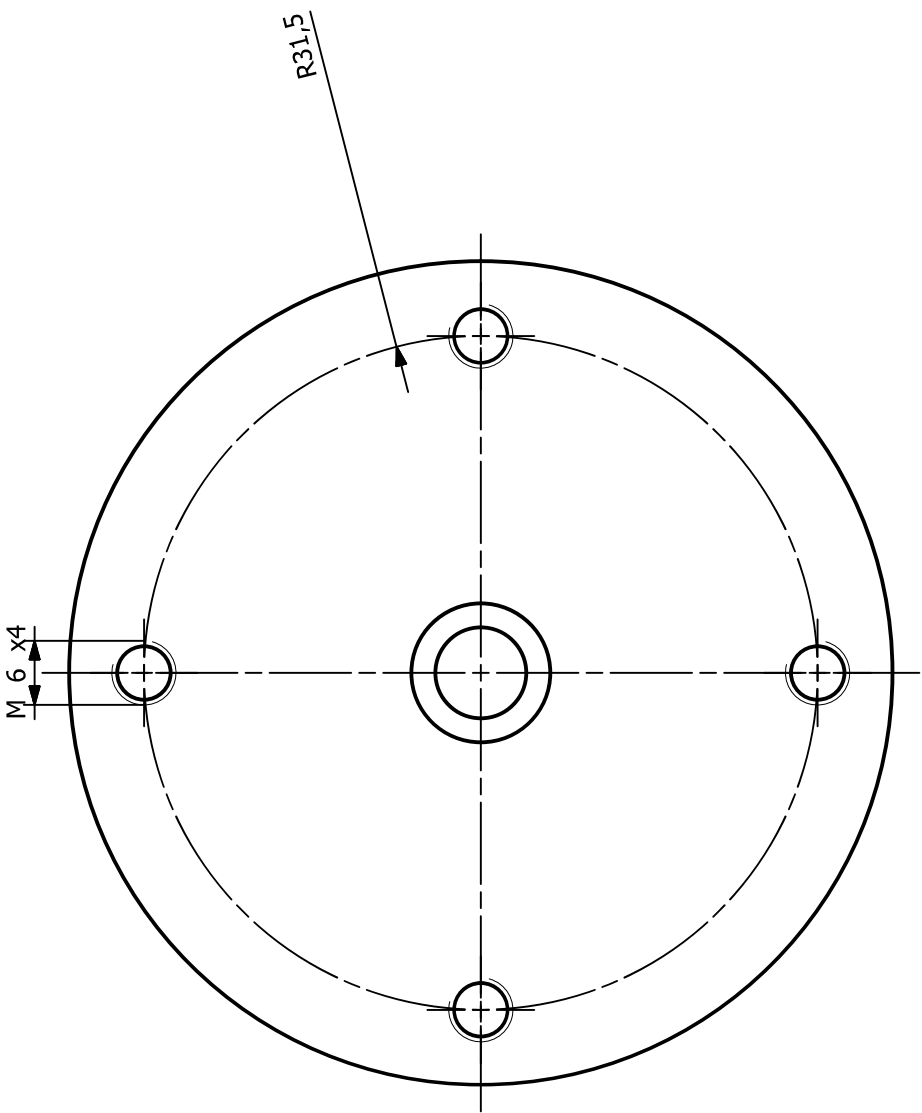
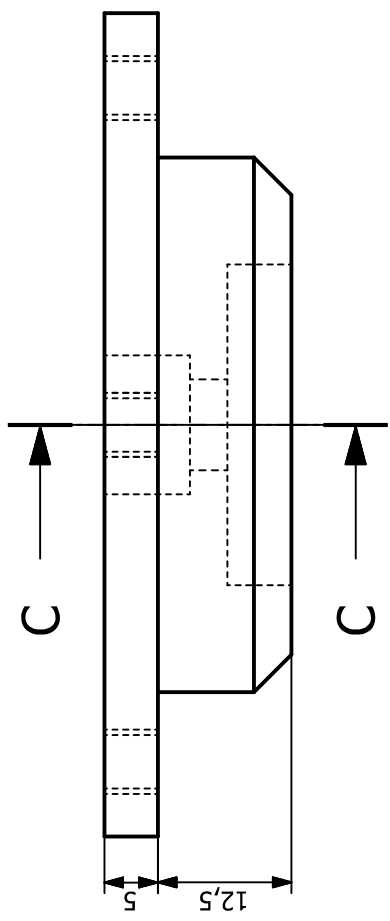
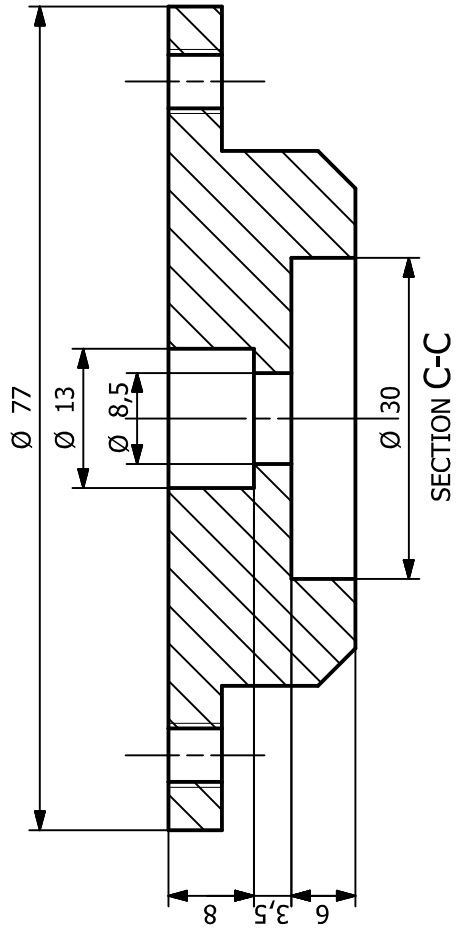


PC NO	Part name	Part ID	Quantity
1	Load cell 50kg		1
2	Short bracket	D b.30-24-6 al m	2
3	Long bracket	D b.130-30-4 al m	2
4	Rexroth gusset	D r.30.g st o	2
.5	M6 hex bolt (not in drawing)	D sc.h.M6-25-10 st o	4
.6	M8 countersunk (not in drawing)	D sc.c.M8-20-x st o	2

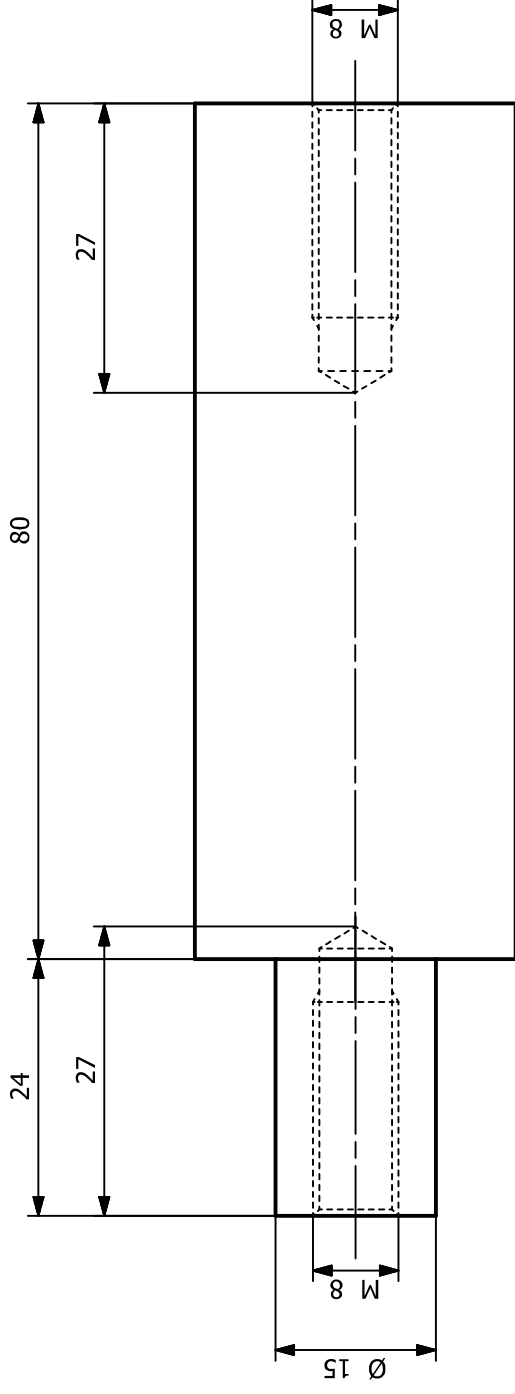
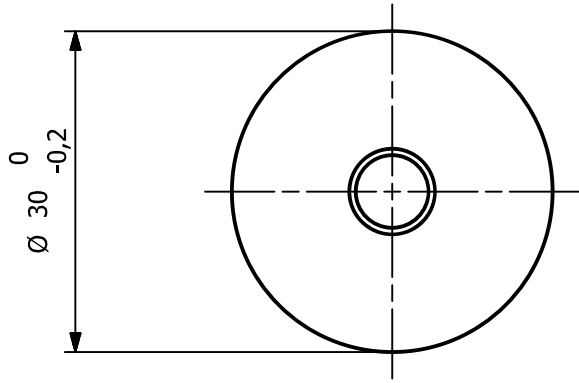
Dato		Modlestokk		NTNU - IPM	
20.11.17	Konstr./Tegnet	Book_jent	2:1	Erstattet av:	
SMJK	Projeksjonsmetode	2:1		Erstattet av:	
Load cell assembly				Part ID: D	
Henvi sning:				Beregning:	



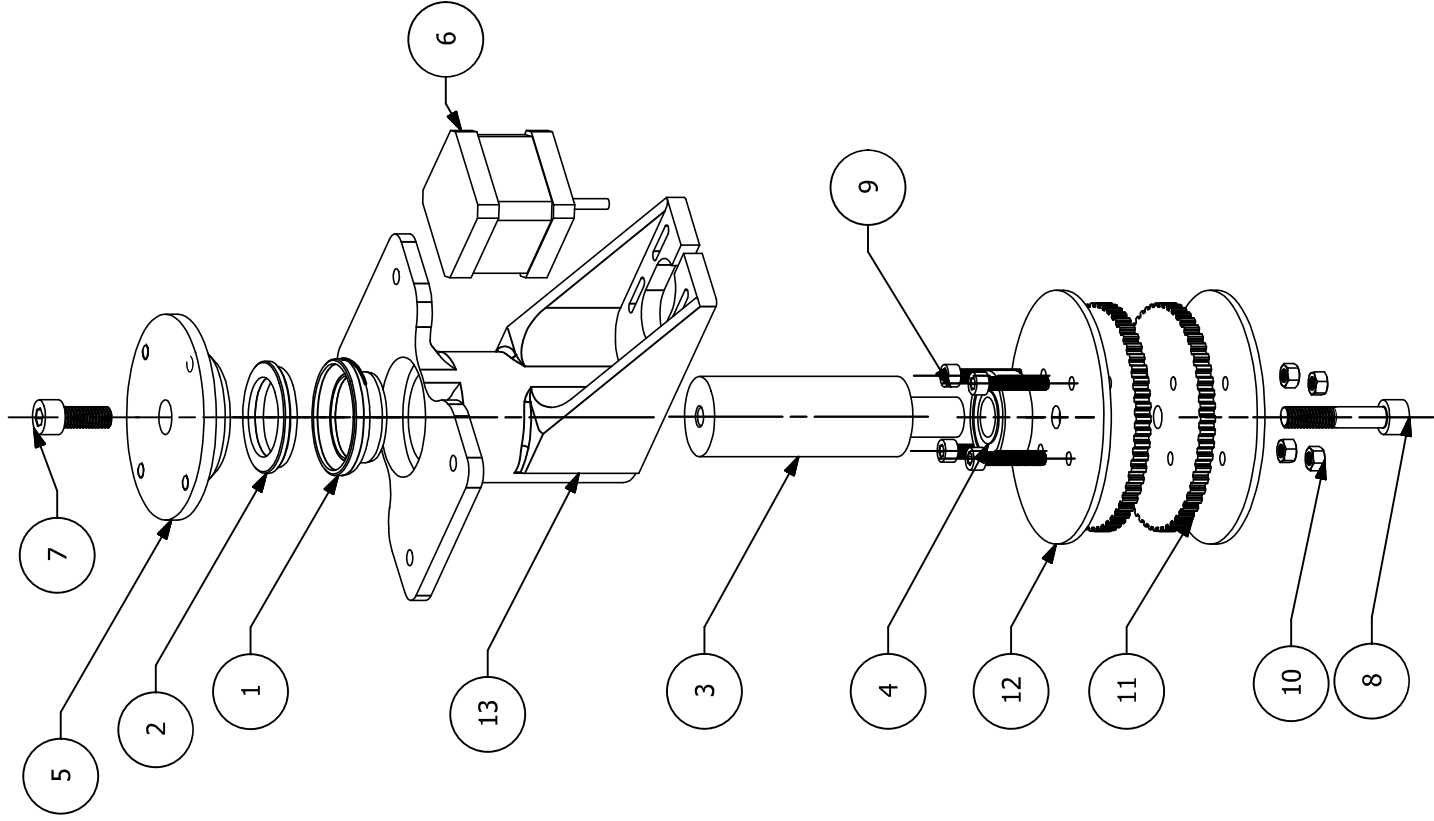
Dato	Konstr./Tegnet	Bokkjent	Modlestokk	NTNU - IPM
19.11.17	SMIK	Prosjektansettede	5:1	Erstatning for:
Steering head bearing from bicycle, critical dimensions				Erstattet av:
Henvi/sning:				Part ID: E be.sh.b.30 st o E be.sh.t.30 st o
				Beregning:



Dato	19.11.17	Konstr./Tegnet	SMIK	Bok_tent	Pro_eks_omsnrkode	Målestokk	2:1	NTNU - IPM	
								Erstattet av:	Erstattet av:
Turntable flange							Part ID:		E f.77 alt
Henvi/sning:							Beregning:		



Dato	Konstr./Tegnet	BokJent	Modlestokk	NTNU - IPM
19.11.17	SMJK	Prosjektansettede	2:1	Erstattet av:
Turntable shaft				Part ID:
Henvi/sning:				E s.30 alt
Beregning:				



PC NO	Part name	Part ID	Quantity
1	Steering head bearing bottom	E be.sh.b.30 st o	1
2	Steering head bearing top	E be.sh.t.30 st o	1
3	Shaft	E s.30 al t	1
4	Ball bearing 15x35x11	E be.ba.15-35-11 st o	1
5	Flange	E f.77 al t	1
6	Stepper motor	E m.s-42BYGHW609D4P1-3 X o	1
7	M8x20 bolt	E sc.s.M8-20-6 st o	1
8	M8x40 bolt	E sc.s.M8-40-6 st o	1
9	M5x25 bolt	E sc.s.M5-25-4 st o	4
10	M5 nut	E n.M5 st o	4
11	Timing belt gear	E p.g.84-T5/54 plexi I	2
12	Gear side edge	E p.g.95 plexi I	2
13	Main body	E mb pla p	1
.14	Timing belt gear stepper (not in drawing)	E p.g.21-T5/16 al o	1
.15	Timing belt (not in drawing)	E p.b.528-T5/330 X o	1
.16	M3x15 bolt (not in drawing)	E sc.p.M3-15-x st o	4
Dato		Moelstokk	NTNU - IPM
19.11.17		1:2	
Konstr./Tegnet		Projeksjonskode	Erstattet av:
SMJK			
Turntable assembly			Part ID: E
Henvi sning:			
Beregning:			

Google drive with 3D models and drafts:

https://drive.google.com/drive/folders/12ldNQDP9O_-97qmUcUv9wqLVFedAisCN?usp=sharing



This page is intentionally left blank.


Appendix E

Risk assessment

This page is intentionally left blank.



Sampsa Kohtala

ID	29356	Status	Date
Risk Area	Risikovurdering: Helse, miljø og sikkerhet (HMS)	Created	28.05.2018
Created by	Sampsa Matias Ilmari Kohtala	Assessment started	28.05.2018
Responsible	Sampsa Matias Ilmari Kohtala	Actions decided	
		Closed	28.05.2018

Risk Assessment:**Risk assessment for master's thesis****Valid from-to date:**

1/14/2018 - 6/11/2018

Location:

Trondheim, MTP Gløshaugen

Goal / purpose

Risk assessment for my master's thesis conducted at the TrollLabs laboratory at NTNU Trondheim.

Background

Master project at MTP Gløshaugen, in which practical work might occur.

Description and limitations

Creation and testing of prototypes in a workshop environment, using different tools and machines. Limited to 3D Printing Laboratory, Mechatronics Laboratory - Gløshaugen and TrollLABS.

Prerequisites, assumptions and simplifications

Usage of specific tools and machines are assumed to take place, based on previous experience using these workshops during other courses and projects.

Attachments

[Ingen registreringer]

References

[Ingen registreringer]

Summary, result and final evaluation

The summary presents an overview of hazards and incidents, in addition to risk result for each consequence area.

Hazard: TrollLabs and mechatronics lab

Incident: Fire

Consequence area:	Helse	Risk before actions:		Risiko after actions:	
	Ytre miljø	Risk before actions:		Risiko after actions:	
	Materielle verdier	Risk before actions:		Risiko after actions:	
	Omdømme	Risk before actions:		Risiko after actions:	

Incident: Incorrect usage of tools

Consequence area:	Helse	Risk before actions:		Risiko after actions:	
	Materielle verdier	Risk before actions:		Risiko after actions:	

Incident: Shock

Consequence area:	Helse	Risk before actions:		Risiko after actions:	
	Materielle verdier	Risk before actions:		Risiko after actions:	

Incident: Exposure to gasses

Consequence area:	Helse	Risk before actions:		Risiko after actions:	
--------------------------	-------	----------------------	--	-----------------------	--

Hazard: 3D print lab

Incident: Incorrect usage of machines

Consequence area:	Helse	Risk before actions:		Risiko after actions:	
	Materielle verdier	Risk before actions:		Risiko after actions:	

Final evaluation

This risk assessment is of general nature. No extreme experiments are conducted in the master's thesis, only simple workshop type scenarios (e.g. making prototypes with fiberboard).

Organizational units and people involved

A risk assessment may apply to one or more organizational units, and involve several people. These are listed below.

Organizational units which this risk assessment applies to

- Institutt for maskinteknikk og produksjon

Participants

[Ingen registreringer]

Readers

[Ingen registreringer]

Others involved/stakeholders

Martin Steinert

The following accept criteria have been decided for the risk area Risikovurdering: Helse, miljø og sikkerhet (HMS):

Helse



Materielle verdier



Omdømme



Ytre miljø





Overview of existing relevant measures which have been taken into account

The table below presents existing measures which have been taken into account when assessing the likelihood and consequence of relevant incidents.

Hazard	Incident	Measures taken into account
TrollLabs and mechatronics lab	Fire	Fire protection
	Incorrect usage of tools	Protective equipment
	Incorrect usage of tools	Training
	Shock	Protective equipment
	Exposure to gasses	Protective equipment
	Exposure to gasses	Ventilation
3D print lab	Incorrect usage of machines	Training

Existing relevant measures with descriptions:

Fire protection

Fire extinguisher, fire blanket, exits

Protective equipment

Goggles, ear protection,

Training

[Ingen registreringer]

Ventilation

Soldering, welding



Risk analysis with evaluation of likelihood and consequence

This part of the report presents detailed documentation of hazards, incidents and causes which have been evaluated. A summary of hazards and associated incidents is listed at the beginning.

The following hazards and incidents has been evaluated in this risk assessment:

- **TrollLabs and mechatronics lab**
 - Fire
 - Incorrect usage of tools
 - Shock
 - Exposure to gasses
- **3D print lab**
 - Incorrect usage of machines

Detailed view of hazards and incidents:

Hazard: TrollLabs and mechatronics lab

Incident: Fire

Cause: Soldering

Description:

Spilling molten metal from soldering.

Cause: High voltage/current

Description:

Testing self made circuits with high power

Likelihood of the incident (common to all consequence areas): **Unlikely (1)**

Kommentar:

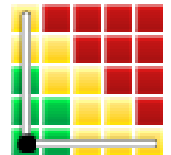
Spills of molten metal from soldering are extremely small and easy to controll.

Consequence area: Helse

Assessed consequence: **Small (1)**

Comment: If fires occur, they will most likely be small and slow.

Risk:

**Consequence area: Ytre miljø**

Assessed consequence: **Medium (2)**

Comment: [Ingen registreringer]

Risk:

**Consequence area: Materielle verdier**

Assessed consequence: **Medium (2)**

Comment: [Ingen registreringer]

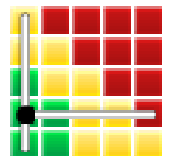
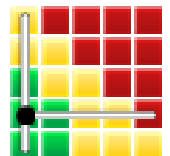
Risk:



**Consequence area: Omdømme***Assessed consequence:* **Medium (2)***Comment:* [Ingen registreringer]**Risk:****Incident: Incorrect usage of tools**

Likelihood of the incident (common to all consequence areas): **Less likely (2)***Kommentar:*

Incorrect usage of tools happens even for experienced users.

Consequence area: Helse*Assessed consequence:* **Small (1)***Comment:* [Ingen registreringer]**Risk:****Consequence area: Materielle verdier***Assessed consequence:* **Small (1)***Comment:* [Ingen registreringer]**Risk:**

Incident: Shock

Cause: Testing circuits

Likelihood of the incident (common to all consequence areas): **Unlikely (1)**

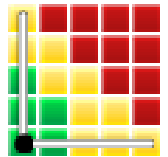
Kommentar:

Electronic components that are modified or created are usually low voltage.

Consequence area: Helse

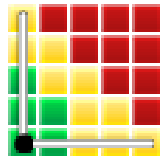
Assessed consequence: **Small (1)**

Comment: [Ingen registreringer]

Risk:**Consequence area: Materielle verdier**

Assessed consequence: **Small (1)**

Comment: [Ingen registreringer]

Risk:**Incident: Exposure to gasses**

Cause: Soldering

Cause: Laser cutting

Likelihood of the incident (common to all consequence areas): **Unlikely (1)**

Kommentar:

[Ingen registreringer]

Consequence area: Helse

Assessed consequence: **Medium (2)**

Comment: [Ingen registreringer]

Risk:



Hazard: 3D print lab

Incident: Incorrect usage of machines

Cause: Careless use

Likelihood of the incident (common to all consequence areas): **Unlikely (1)**

Kommentar:

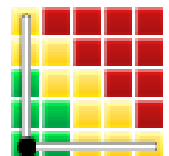
Machines are fairly simple to use, even for new users.

Consequence area: Helse

Assessed consequence: **Small (1)**

Comment: [Ingen registreringer]

Risk:



Consequence area: Materielle verdier

Assessed consequence: **Medium (2)**

Comment: [Ingen registreringer]

Risk:





Overview of risk mitigating actions which have been decided:

Below is an overview of risk mitigating actions, which are intended to contribute towards minimizing the likelihood and/or consequence of incidents:

Overview of risk mitigating actions which have been decided, with description:



Detailed view of assessed risk for each hazard/incident before and after mitigating actions