CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Utilizing Computer Vision for the Analysis of Manufacturing Processes

Evaluating Automation Possibilities for the AviX Software Suite

Bachelor's thesis in Computer science and engineering

Oussama Anadani
Herman Bergström
Gustav Fåhraeus
Oscar Helgesson
Simon Svensson
Hanna Tärnåsen

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Bachelor's thesis 2021

Utilizing Computer Vision for the Analysis of Manufacturing Processes

Evaluating Automation Possibilities for the AviX Software Suite

Oussama Anadani
Herman Bergström
Gustav Fåhraeus
Oscar Helgesson
Simon Svensson
Hanna Tärnåsen

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Utilizing Computer Vision for the Analysis of Manufacturing Processes
Evaluating Automation Possibilities for the AviX Software Suite
Oussama Anadani Herman Bergström Gustav Fåhraeus Oscar Helgesson Simon Svensson Hanna Tärnåsen

Supervisor: Pedro Petersen Moura Trancoso, Department of Computer Science and Engineering
Advisor: Oskar Ljung, Solme AB
Examiner: Sven Knutsson, Department of Computer Science and Engineering

Bachelor's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Image modified using the created applications face blur and pose estimation functionality. Source adapted from [1].

Typeset in LaTeX
Gothenburg, Sweden 2021

Utilizing Computer Vision for the Analysis of Manufacturing Processes
Evaluating Automation Possibilities for the AviX Software Suite
Oussama Anadani, Herman Bergström, Gustav Fåhraeus, Oscar Helgesson, Simon Svensson, Hanna Tärnåsen

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Throughout the manufacturing industry, video recordings are used to help standardize work and develop training material for companies. Solme AB develops a software suite named AviX which aids in the analysis of these recordings. This report aims to evaluate how computer vision technology could be utilized to increase the functionality of the AviX suite. Furthermore, the report will evaluate how the technology could be used to automate analysis currently performed manually in the program. The evaluated features are face blur, tool highlighting, ergonomic risk detection, and footstep counting. A software platform is developed in Java, primarily with the use of OpenCV, to serve as a proof-of-concept for Solme. To support the flexibility of changing the set of enabled features, the application was constructed modularly and the features were implemented independently. The thesis concludes that there is potential to extend the functionality of the AviX suite by utilizing computer vision. Automated face blurring has been achieved with a considerable success rate, increasing the privacy of people appearing in the video recordings. Moreover, the automation of ergonomic risk detection showed promising results which indicate that manually performed analysis can indeed be automated.

Index Terms—Artificial intelligence (AI), computer vision, ergonomic risk, face blur, face detection, neural networks, object detection, pose estimation, step counting, tool highlighting.

Sammanfattning

Inom tillverkningsindustrin används videoinspelningar frekvent för att standardisera arbetsuppgifter samt för att fungera som utbildningsmaterial internt inom företag. Solme AB utvecklar en mjukvaruplatform, vid namn AviX, som syftar till att underlätta analysen av dessa inspelningar. Målet av denna rapport är att evaluera hur datorseende kan användas för att utöka funktionaliteten hos AviX. Vidare kommer rapporten evaluera hur teknologin kan användas för att automatisera analys som för tillfället utförs manuellt i programmet. Funktionaliteten som utvärderas är ansiktsblurrning, verktygsmarkering, detektering av ergonomisk risk, samt stegräkning. En mjukvaruplatform har utvecklats i Java, primärt med hjälp av OpenCV, för att användas som ett koncepttest åt Solme. För att få flexibilitet och göra det lätt att välja vilken funktionalitet som ska vara aktiverad har applikationen byggts modulärt och de olika funktionerna har implementerats oberoende av varandra. Rapporten drar slutsatsen att det finns potential att utöka funktionaliteten hos AviX genom att utnyttja datorseende. Automatisk blurrning av ansikten har uppnåtts med betydande framgång, vilket ökar integriteten hos personerna som visas i videoinspelningarna. Dessutom visade detekteringen av ergonomisk risk lovande resultat vilket indikerar att manuellt utförd analys kan automatiseras.

Nyckelord—Ansiktsblurrning, ansiktsigenkänning, artificiell intelligens (AI), datorseende, ergonomisk riskbedömning, neurala nätverk, objektdetektering, stegräkning, verktygsmarkering.

Acknowledgement

We would first and foremost like to thank our supervisor Pedro Petersen Moura Trancoso. The thesis would not have been possible without your positive energy, ideas, and constant will to help push the project forward.

Furthermore, we would like to thank Oskar Ljung for his continued support throughout the project. Your constant interest and cooperative spirit ensured the project stayed on the right path. We also want to thank Uma Shankar Subramani and the rest of the AviX team for their guidance in the construction of the application.

Contents

1
Introduction

Video recordings are used in the manufacturing industry to help standardize work and develop training material for companies [2]. By analyzing these recordings, companies can also restructure their workstations to provide a more efficient workflow, as well as to reduce poor ergonomic conditions for the workers.

Solme AB [3] is a software company specialized in video analysis of industrial processes. Their product, a software suite named AviX [4], offers tools for ergonomic analysis and optimizations of manufacturing processes. AviX is based on recording the work process, timing the tasks, and classifying their utility. When the process is visualized, several modules enable evaluation and optimization. These modules provide tools for line balancing and execution optimization [5], failure mode and effects analysis (FMEA) [6], ergonomic analysis, single-minute exchange of dies (SMED) [7] analysis and product design.

1.1 Purpose

Recently, Solme has been approached by customers who have had concerns regarding the privacy of their employees. The company has as such begun exploring different features and solutions that would make the manufacturing workers more comfortable with the video analysis. One of the suggested solutions is to implement a feature that would enable users of AviX to blur the faces of people who appear in the recordings by utilizing computer vision.

When discussing the possibility to use computer vision technology to blur faces, Solme also showed interest in exploring other ways the technology could be utilized in their software suite. As a result, they want to investigate if it would be possible to automate the analysis currently performed manually by users of AviX.

The purpose of this thesis is to explore how computer vision can be used to improve upon or create new, features for the AviX software suite. The primary feature will be the blurring of faces, but together with Solme an additional set of analytical features to examine has been decided upon. These additional features are: 1) tool highlighting; 2) ergonomic risk detection; and 3) footstep counting. The functionality of these features will be explained in detail in Chapter 3. The goal is to produce a prototype that can serve as a proof-of-concept for Solme from which they can gain insight into the potentials and limitations of utilizing computer vision to offer

these new features. As different analysis features will be explored and implemented, the aim is to build a modular platform that allows the user to easily change which analysis is to be executed.

1.2 Problem Description

The problem posed in this thesis is to produce a proof-of-concept platform using openly available AI models to implement features desired by Solme. The main requirement for the platform is that it should be created with Java. Furthermore, it should utilize openly available resources in the form of code and classes, henceforth referred to as open-source libraries. The features that are to be implemented are:

- **Face Blurring** - The ability to blur faces that appear in the video. This will be the primary feature.
- **Tool Highlighting** - Highlight tools used in the video for educational purposes.
- **Ergonomic Risk Detection** - Timestamp moments in the video containing non-ergonomic motions.
- **Footstep Counting** - Count the number of steps taken in a video.

The platform should be able to take a video file as input and output a modified version of the file. Furthermore, it should output the results of the analysis performed in an appropriate document format. This could for example mean outputting a video with faces blurred and a document containing timestamps for non-ergonomic motions. Additionally, the user should be able to select which analysis is to be performed. As the focus of this project is not to build a graphical user interface (GUI), command-line arguments will be used.

The proof-of-concept is supposed to work as a standalone application. The goal is not to integrate any functionality into the AviX suite at this point, but rather build a platform in Java that could be used as a reference if Solme wishes to pursue these features.

1.3 Ethical Evaluation

Computer vision applications today can capture exceedingly detailed and personalized information. This has given rise to ethical concerns as the technology can be used for different purposes. For example, developers of computer vision have collaborated with authorities to monitor people of a specific ethnic group [8]. Furthermore, there is also the issue of people using computer vision to generate synthetic media where the person appearing in the video is manipulated to look like someone else, commonly called 'deepfakes' [9].

Even though this work is vastly different from the examples above, there is un-

derstandably a negative stigma attached to computer vision systems [10]. People do not generally like being surveilled, and it should be fully within their rights to choose not to participate in the videos. In the case of this project, the people appearing on video are consenting adults who have been informed of what the footage will be used for. Furthermore, different use cases of computer vision have different ethical implications. As such, the implications of each feature that is to be implemented will be discussed in the following sections.

The blurring of faces is a feature with a large social aspect. According to the recent EU General Data Protection Regulation (GDPR), the face is a person's most visible and primary form of identification [11]. As such, the face blurring of manufacturing workers can be seen as a measure to protect their privacy. However, it should be noted that face blurring does not ensure total de-identification. Identifying individuals through other personal traits such as their bodily physical appearance and clothing is still possible.

In addition, there are outcomes of the face blurring feature which might not be in the interest of the workers. Firstly, studies show that blurring faces not only de-identify them but also de-humanizes them [12]. This can subsequently decrease the watchers' ability to relate to the participants of the video. Secondly, it is important to recognize that Solme's interest in face blurring is not inherently to protect the privacy of manufacturing workers. Instead, the goal is to make these workers feel more comfortable with being recorded by offering them a sense of privacy.

In contrast, the tool highlighting feature is different in that it does not analyze any individual per se. It only looks for inanimate objects, not people. It could be argued that the feature is in the interest of the employees as they are presented with a clearer learning experience. However, even if it is not the intended effect of the feature, it might work as an additional form of surveillance. The feature could potentially identify an object that a person is not supposed to be using. For example, it could identify a phone being used by an employee, even though the company has a strict no-phone policy. This could in turn give the employees the feeling that they are being monitored.

Lastly, it is hard to identify problematic scenarios for the ergonomic risk detection and step counting features. The ergonomic analysis feature is meant to benefit both the employees and the employers. Working extended periods at an assembly line that has poor ergonomic conditions can lead to long-term injuries [13]. This is naturally undesirable for the employee but can also be expensive for the employer. With this in mind, it is easy to see how identifying which stations have poor ergonomic conditions is of great interest. In the case of the step counting feature, a positive aspect could be ensuring that the manufacturing process allows the workers to walk an appropriate distance each day. It may seem obvious that walking too much is taxing on the body, but so is walking too little [14].

1.4 Thesis Outline

After this introduction, the thesis will start by offering background knowledge to important areas in Chapter 2. In this chapter, an overview of what computer vision is will be presented, as well as a quick introduction to the AviX software suite. Furthermore, the section will cover work related to this project.

In Chapter 3 the features will be discussed in detail. The functionality of each feature will be defined, while also offering explanations as to how AviX is to benefit from them.

Next, in Chapter 4 the work process will be described. This will include descriptions of how the libraries were evaluated, how features were explored as well as how testing was performed.

A rundown of the software created will be presented in Chapter 5. The section will cover the program structure as well as the implementations of each feature. The program flow when processing a video will be explained to offer an overview of the prototype.

The results and discussion presented in Chapter 6 will firstly focus on how well the proof-of-concept works as a whole. It will then subsequently go over each feature and discuss its performance.

Lastly, in Chapter 7 the report will offer some conclusions and ideas for potential future work. Using the results from the previous chapter, the section will summarize the issues that Solme would encounter if the software of this kind was to be implemented in their AviX suite. Furthermore, it will offer closing thoughts as to which features could work well given enough effort, and which features may prove more difficult to implement.

2
Background

This section aims to provide the reader with the essential background knowledge needed to understand the content of the thesis.

Firstly, key concepts surrounding computer vision used within the project will be introduced and explained to aid the reading comprehension during the technology-focused sections of the thesis.

Furthermore, a brief introduction of AviX will be presented so that the reader can understand the connection between the project and the software suite.

Lastly, a brief presentation of previous solutions to similar problems will be shown in order to give the reader an overview of previous work performed in the area.

2.1 Computer Vision

Computer vision refers to the use of different methods within the realm of artificial intelligence to extract information from visible media such as digital images or video. To classify as computer vision, the AI needs to not only extract information but also do something with the information, such as recommend or take an action [15].

Computer vision as a technology has been around for decades. The earliest implementations of the technology (the late 1960s) were attempts to mimic the human visual system [16]. This resulted in the technology that is today called 'image tagging', i.e. computers being able to categorize images based on what is predicted to appear in them.

2.1.1 Artificial Intelligence (AI)

There is no generally accepted definition of what artificial intelligence, or AI, means. In this project, AI is interpreted as the entire set of tools and technology that can be used or interpreted in a way that bears a resemblance to human intelligence [17]. This project primarily refers to the detection of faces in videos and the estimation of human poses. AI can as such be seen as a term that encapsulates many other techniques and methods [18]. One such method being machine learning.

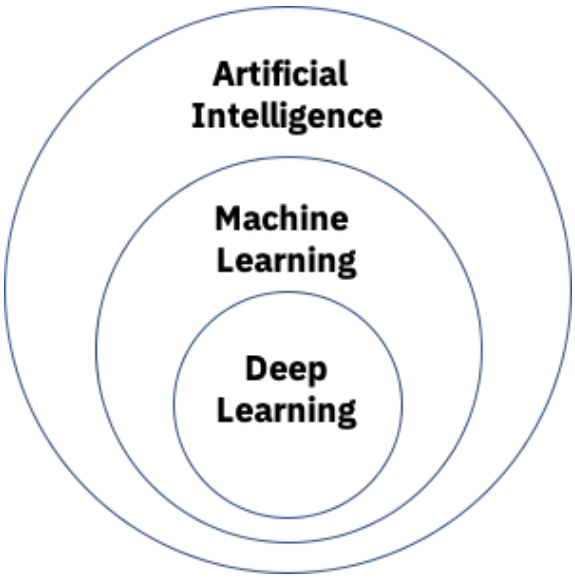


Fig. 1. The relationship between artificial intelligence, machine learning, and deep learning.
Source: Adapted from [17].

2.1.2 Machine Learning (ML)

Machine learning is generally described as mathematical methods constructed to make predictions and optimization through the use of experience [19]. The term experience is meant to symbolize data from which a method can *learn* how to do a specific task and the quality and quantity of the incoming data points.

More concretely, in this project, ML refers to the way the methods used in different libraries are constructed. Different libraries used in aspects of this project contain, or can use, different networks and algorithms to achieve a specific goal. For example, the networks used to detect the position of faces in images are trained through machine learning. Algorithms trained through ML are the key to performing the computer vision tasks this project requires [20].

Different algorithms that have gone through different training regimes differ in their performance and efficiency. When optimizing an algorithm, it is provided a large amount of data. In the context of face detection, an algorithm would be provided with a large number of images containing faces as well as labels stating if and where a face is located. The algorithm uses these images and labels to learn how to make estimated guesses. Algorithms or networks optimized under certain conditions are therefore also more accurate when predicting under these same conditions.

2.1.3 Neural Networks (NN)

A subpart of ML is neural networks. Neural networks aim to imitate the human brain through a collection of algorithms [21]. A neural network works by taking an input, sending it forward to an arbitrary number of hidden layers (algorithms with constants and coefficients that are optimized when training), before mapping it to outputs representing different predictions. In the end, a neural network output can range from being binary, such as *yes* or *no*, to a multitude of values mapping for example different colors that can be identified in an image [22]. A neural network that contains two or more hidden layers is usually referred to as a deep neural network (DNN)[21].

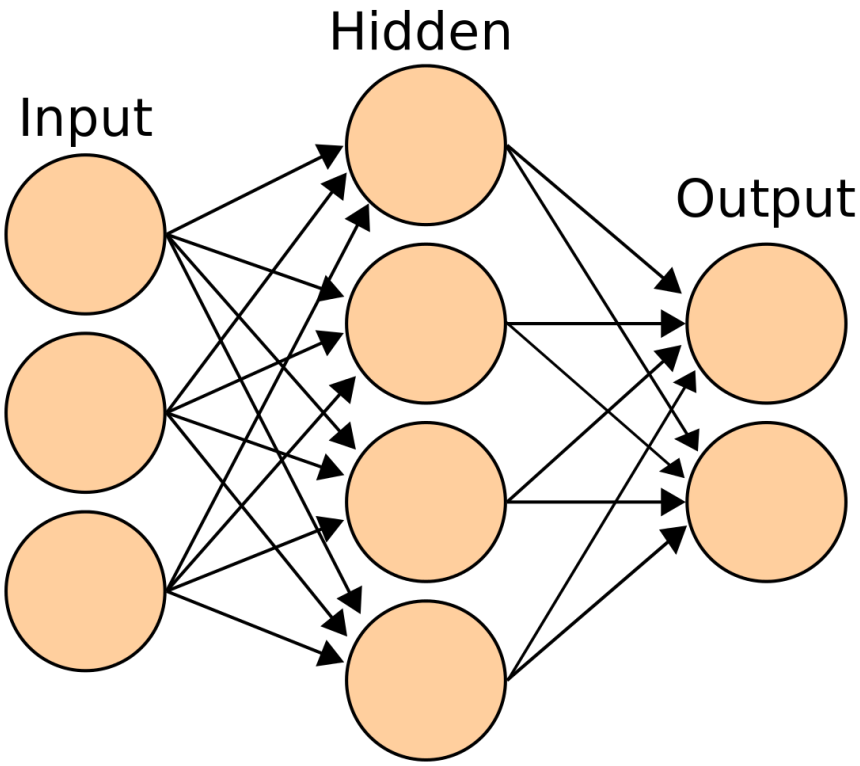


Fig. 2. A neural network featuring a three dimensional input layer, a hidden layer and two types of output.
Source: Adapted from [23]

All this combined allows for inputs to be sent through a network with multiple different paths to finally result in some sort of output that can be used to reach conclusions or make predictions.

2.2 The AviX Software Suite

AviX is a software suite, created by Solme AB, that is used for optimization and analysis of production and design processes. AviX consists of modules, with different

modules for different purposes. For example, there is 'AviX Method' and 'AviX Ergo', the first being their tool for method and time studies that come with a built-in media player. The latter, 'AviX Ergo', is a module that through video processing technology analyzes the ergonomics of the workplace [24].

2.3 Related Work

In this section, related work to the work made in this thesis will be presented. Firstly, work related to the face blurring feature will be presented. Secondly, related work to the ergonomic risk detection will be highlighted. Thereafter, related work to the tool highlighting and footstep counting features respectively will be introduced.

Face detection is used in many different applications for a multitude of reasons. There is the use case where the face detection is subsequently followed by face blurring. This is primarily performed because of privacy concerns of people appearing in images or videos, which often is a necessity when using street-level images. The most notable example of this is Google Street View [25, 26]. Furthermore, computer vision can also be utilized for full-body de-identification, as a continuance to face-blurring [27]. This is done in a way that retains the human features while allowing for increased privacy of individuals.

Previously, software has been developed that aims to assist in the process of assessing ergonomic risk. For example, Venkatesulu and Koundinya implemented a solution during their master thesis project at Chalmers where they used wearable sensors to assess ergonomic risk [28]. The sensors (accelerometers) were placed on different segments of a person, and as this person was going about his or her work the sensors measured the movement of the body. Using this data, information such as the angles between certain joints was extrapolated and used to calculate a score using the Rapid Entire Body Assessment (REBA) method. REBA is a tool that uses generally applicable ergonomic standards which results in it being used in a wide variety of working environments to quantify ergonomic risk [29].

Regarding tool highlighting, computer vision can be used for more than just detecting which tools that are being used, it can also be used to evaluate the wear of a cutting tool [30]. Using the tool in different ways and subsequently evaluating the wear can lead to finding optimal usage methods for minimal wear. That is favorable from an economical perspective.

In the case of footstep counting, historically some form of pedometer has been used to keep track of a person's footsteps. One paper that more closely relates to this project presents a solution where footsteps were being counted by analyzing footage of a person doing agility training [31]. No sensors were used and it had a 97% accuracy while processing in real-time. The limitations of their solution are that the camera has to be stationary, and the movement is constricted to a pre-configured square where a single person's movements are predictable.

3

Features for AviX

This chapter will explain the features in more detail. It will be going deeper into what the features are and why they are useful for AviX. How these features ended up being implemented in the program will be explained in Chapter 5.

3.1 Face blurring

The face blurring feature edits the frames in a video to blur faces. An example is showcased in Fig. 3. The means of the feature is the anonymizing of facial features. However, as mentioned in Section 1.3, more parts of a human can be used to identify the human such as clothes, posture, and body. Therefore, it is not complete anonymization.

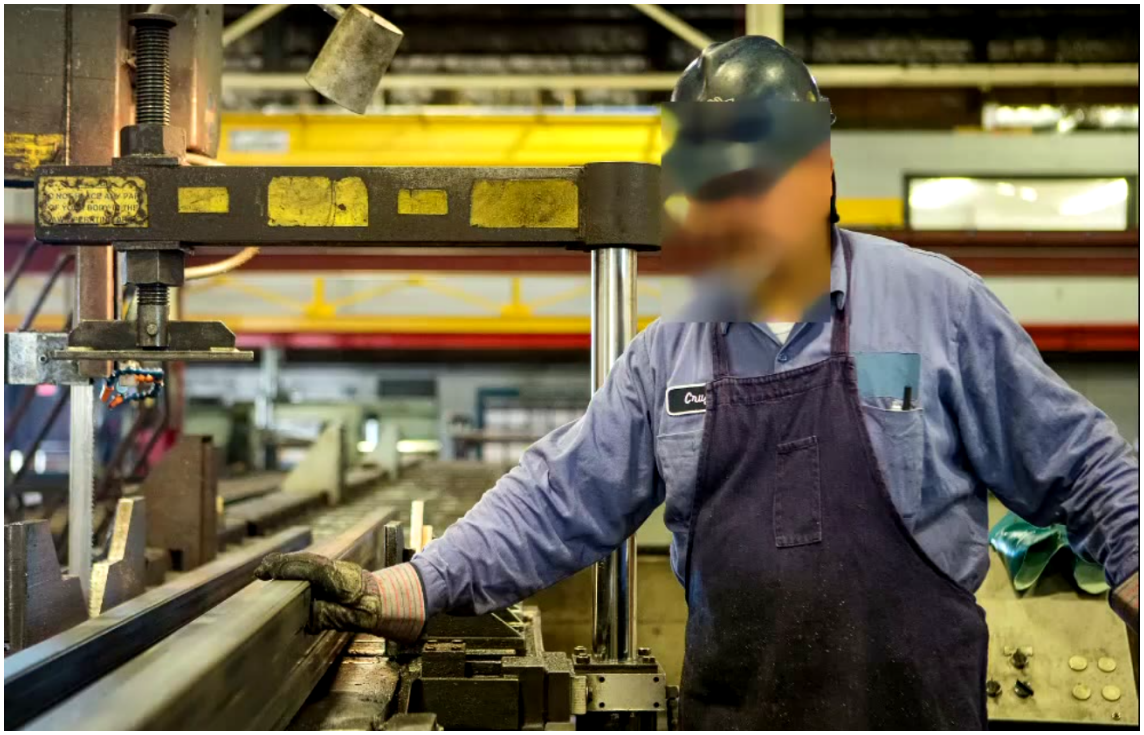


Fig. 3. Face detected with OpenCV's Deep Neural Network (DNN) module.
Source: Adapted from [32]

Face blurring can be divided into two separate steps; where the first step is best described as the use of computer vision for face detection, or, to identify human faces in digital images [33]. The second step involves blurring the identified face in the image.

Solme desires face blurring to be implemented into AviX due to customer interest. Their customers are the companies where the filming takes place, and those companies' workers are the people being filmed in the videos for AviX. Hence, the blurring is being added for them to feel more comfortable being filmed and therefore be more inclined to participate in the videos.

3.2 Tool Highlighting

The term tool highlighting is a specific case of object detection that has tools and machines as identifiable classes. The general purpose of object detection is to automatically locate and categorize objects of interest in videos. Different objects can be chosen to be categorized based on which objects are relevant to the intended use case. An example of object detection is showcased in Fig. 4.

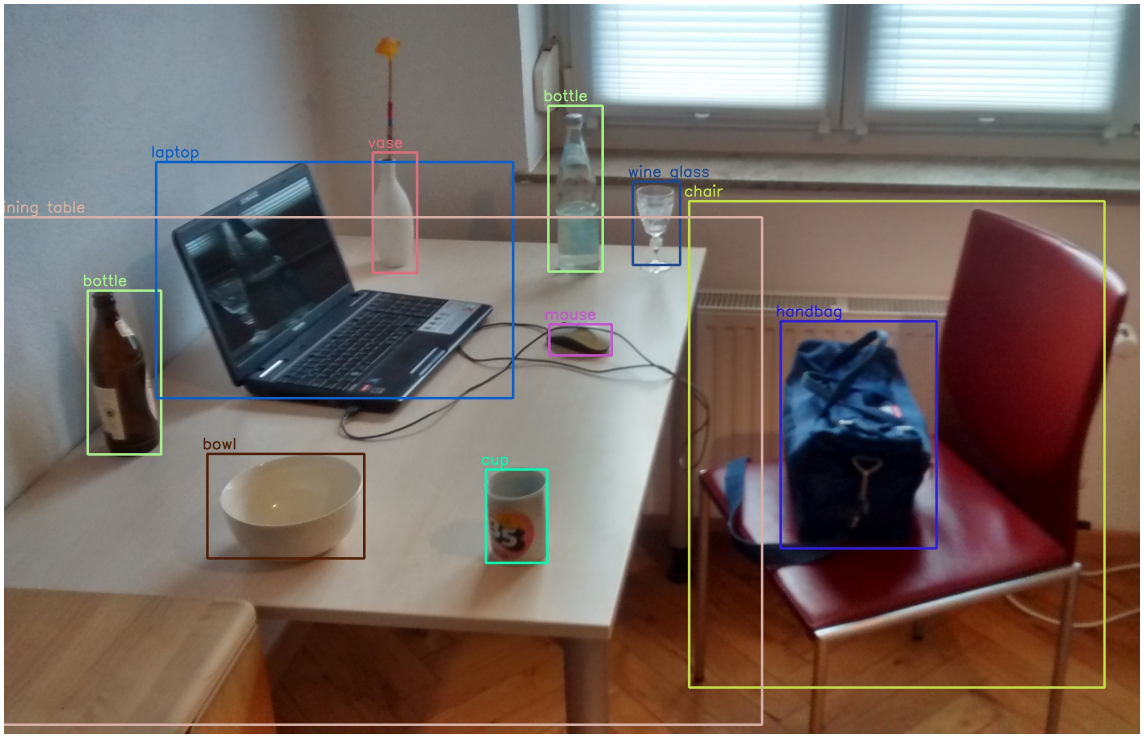


Fig. 4. Objects detected with OpenCV's Deep Neural Network (DNN) module.
Source: [34].

Tool highlighting can be implemented with DNNs trained on tools and machines. The coordinates of objects as well as their names will be output from the DNN, which can be used to automatically insert the bounding box and the name in each

image of the video.

AviX is among other things, used for educational purposes. Therefore, highlighting what tools and machines that are being used could aid the learning process for employees. To prevent unnecessary screen cluttering, it is important that this feature can be turned off when the identification of tools is not of interest.

3.3 Ergonomic Risk Detection

The feature aims to detect when subjects on film are in positions of especially high ergonomic risk. This is achieved through the use of key points to make wider assumptions about when a person appearing in the video could potentially be under higher risk for injury.

With the help of machine learning, it is possible to extract 2D coordinates of certain anatomical key points of people appearing in a video frame. This technology is usually referred to as pose estimation. Examples of such key points are the shoulder, elbow, and eyes as can be seen in Fig. 5. Thereafter, these joints can be used to determine if a person is in a high risk pose. The occasions where such a pose is detected will then be timestamped with the purpose of being analyzed further.

Fig. 5. Pose detected by OpenPose.
Source: Adapted from [35].

In the AviX Software suite, there is a module called 'AviX Ergo' which aims to assist Solme's customers with ergonomic risk assessments by performing video analysis. This process requires a person to look through all the video footage to detect high-risk positions. The goal of this feature is to aid that person by using AI to analyze the video, creating timestamps at potentially harmful positions, thus making the process of assessing ergonomic risk quicker.

3.4 Footstep Counting

Footstep counting refers to automatically counting the number of footsteps taken by a person in a video. As a metric for measuring physical activity, counting steps has many benefits: 1) it is intuitive; 2) objective; and 3) they constitute a basic unit of human ambulatory activity [36].

One way that the feature can be implemented is by using the same solution as the ergonomic risk detection feature. With a 2D skeleton that is generated frame by frame, a primitive footstep counter can work by analyzing ankle or foot movement.

The main use case for footstep counting is as a metric of efficiency. Since walking does not directly contribute to parts of the manufacturing process being assembled, it is desirable to minimize it. Walking is at times a necessary task to perform when parts need to be moved. However, the parts could be stationed closer together making the walking distance shorter. Calculating the number of steps enables the possibility to evaluate different methods of organizing the manufacturing process to find the optimal one. On the other hand, the feature could also be used for ergonomic purposes. As mentioned in Section 1.3, walking too little can be taxing on the body. By analyzing the results of this feature, companies can ensure that workers take a sufficient amount of steps when working on a station.

4
Method

This chapter describes the methodology used to obtain the project results. Firstly, the overall work process will be presented, followed by the methodology for selecting what libraries to use and which were chosen. Lastly, a summary of how the evaluation of the software was conducted is given.

4.1 Work Process

The work was performed in subgroups, each with a different feature in focus, generally following the steps below:

1. Research a feature to gain an understanding of how it works and how it could be useful.
2. Research how the feature could be implemented in a way that follows the requirements.
3. Create an implementation of the feature.
4. Test and evaluate different AI models of the feature and explore what changes and additions could be made to the software to yield better results.

To allow for this distributed work style, a common interface for the features was designed. This led to the subgroups not having to worry about each other as long as the code was implemented towards this interface. Furthermore, the common interface supported the modularity of the software, allowing features to work in different configurations and the implementation being decoupled from specific feature implementations.

4.2 Utilized Libraries

Finding suitable libraries was one of the first tasks performed. The libraries needed to satisfy the requirements in the problem description, such as being compatible with Java and aid in the implementation of desired features. During this process, other factors were taken into accounts, such as costs, accessibility, extensibility, and compatibility with other libraries.

Using the requirements set out, OpenCV (see Appendix A.1.1) was the primary library chosen and used for the proof-of-concept. OpenCV was chosen primarily as it is a well-rounded open-source computer vision library that supports Java and

offers extensibility to multiple features. The support of many features is beneficial as it eases the implementation.

The other chosen library was TensorFlow (see Appendix A.1.2). TensorFlow is an ML-oriented library that offers a wide range of functionality, ranging from AI model creation to running existing models. In this project, it was specifically used to implement the ergonomic risk detection and footstep counting features. However, the library has limited Java support and was primarily selected as it has prominent examples of pose estimation implementations.

4.3 Software Evaluation

The purpose of the software evaluation during the project has been twofold:

1. Comparing the quality of the feature implementations.
2. Comparing the quality of different computer vision libraries and services.

The evaluation has been performed in parallel with the feature implementations. Solme insisted on the exploration of multiple implementation routes before settling on the one deemed preferable. These different routes took the form of using different libraries and APIs in separate implementations of the same feature. They also took the form of using different machine learning models in an otherwise identical implementation.

As it became more clear which the most suitable library was, the evaluation became more detailed and implementation-specific within that library. This testing served to evaluate if the additions and/or changes being made to the application improved the overall quality.

It is however important to note that the type of evaluation performed varies from feature to feature. While detailed tests have been performed, some features have only been evaluated at a higher level. The main reason for this is that evaluations are being performed to gain an understanding of the performance of different features. This means that the evaluation results should offer deeper insight into which parts of the program work well and which parts do not. Evaluation that does not offer any such new information is deemed redundant.

Not all features received an in-depth evaluation. Rather, only the face blur and ergonomic risk detection features did. The method used when evaluating these two in detail will be described in the coming sections. Tool highlighting and step counting were only evaluated at a higher level. This is due to the features having more apparent performance issues, rendering in-depth evaluation unnecessary.

4.3.1 Face Blur

The evaluation of the face blur feature was performed manually. This was because there is no available data set containing faces under the same conditions as the typical video provided by Solme. Because of this, an approach in which the feature was evaluated based on how well it could detect faces in a pre-existing data set could not be chosen. Instead, a manual frame-by-frame approach was chosen to evaluate the quality of the face blur feature under appropriate conditions.

First, a suitable ten-second clip from an AviX video was chosen. Different clips have been tested for different purposes which will be discussed shortly. After the clip is selected, it is viewed frame-by-frame, for every frame noting how many present faces should be detected. The video is then processed by the face blur implementation in question. Lastly, the output video is analyzed frame-by-frame again, noting which faces were detected correctly.

When doing this, the results were saved and an accurately detected face was denoted as a true positive (TP), a non-face that was detected as a face was denoted as a false positive (FP), and a face that was not detected was categorized as a false negative (FN).

When it comes to the different clips selected for testing, sequences are chosen to be representative of both good and poor face detection conditions. Good face detection conditions are defined as follows:

- The people in the video are facing the camera.
- The people in the video are standing relatively still.
- The camera is stable and does not move around noticeably much.

Subsequently, poor conditions are defined as:

- The people in the video are facing away from the camera (while still being identifiable).
- The people in the video are moving around rapidly.
- The faces are partially covered.
- The camera is shaky and moves around.

Face detection was evaluated on multiple occasions throughout the project, as it is Solme's highest requested feature. It was first evaluated to compare different libraries. During this process, simple face detection implementations were made using each library. The implementations were then evaluated on the same clip of average quality to gain an overview of how well the different libraries performed.

At the later stages of the project, face detection evaluations were performed to compare different DNN models and see how well the proof-of-concept performed under different conditions.

4.3.2 Ergonomic Risk Detection

The evaluation of the ergonomic risk detection took two different forms: one with controlled and chosen footage and another where the implementation was evaluated on footage provided by Solme's customers. In both cases, the questions posed were:

- How accurate is the time stamping of the movements?
- Are there any false positives (FP) and/or false negatives (FN) occurring?

The controlled footage was filmed using webcams and consisted of one group member going through a "choreography" made out of all the movements that should trigger a detection. The videos provided by Solme's customers represented typical footage that AviX would analyze. The feature was opted to be tested this way to see how it performed under the best conditions that could be set up, as well as more typical footage.

It is of interest to evaluate whether the detections are triggered correctly as right or left regardless if the person is filmed from the front or from behind. To evaluate this, the video and timestamps were compared to ensure the accuracy of the detections. Factors such as lighting and positioning need to be taken into account when discussing the results generated from the tests.

During the evaluations, some situations require human evaluation to determine if the timestamp of a detection should be considered correct. For this, the acceptable time range was chosen to be one second. This means that the movement has room to be somewhat misplaced. There is also subjectivity to deal with when it comes to what is considered a stretch away from the body.

5
Implementation

This section will give an overview of the implementation of the program. An abstract description of the structure will first be given to offer a general understanding of the software. The implementation of AI models and their corresponding features will then be explained in greater detail.

5.1 Program Structure

The program analyses a given input video and outputs a modified version of that video. The output will vary depending on which features have been enabled. Additionally, the program outputs a JSON document containing eventual timestamps of interest. During this process, the application iterates over every frame of the video, processing one frame at a time. The analysis of a frame is performed by passing it through the enabled evaluators. Exactly what an evaluator is and what the different versions are will be explained in Section 5.2. For now, the evaluators can be summarized as the classes in charge of running the images through the neural networks and interpreting the results.

Although each frame is individually sent to the evaluators, there is still information to be gained by comparing frames one to another. For example, a face being detected in one frame implies a higher chance of the next frame containing a face in the close-by area. To cater for this need, the program is constructed using buffers, which save previous frames as well as previous results generated by the evaluators. These buffers are limited in size and when they fill up different optimization algorithms can be utilized to improve the results. These algorithms will be explained in detail in Section 5.4. A flowchart of the abstract program structure which has been provided can be seen in Fig. 6.

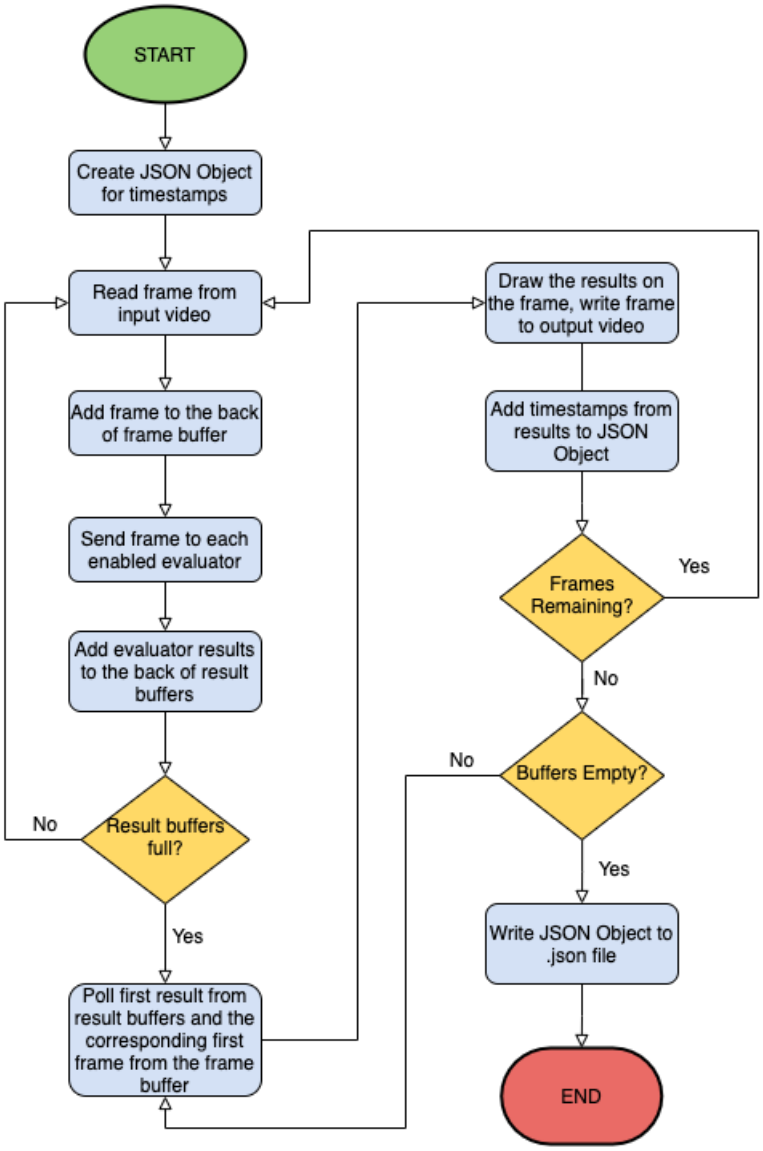


Fig. 6. A simplified version of the program flow.

5.2 Evaluators

The evaluators are in charge of analyzing a given frame by passing it through different neural networks. After the network has finished processing the frame, the evaluators interpret the output and each one generates a detection result. The data contained within a detection result varies from evaluator to evaluator. The detection result is subsequently added to that evaluator's result buffer.

There are three different evaluators, each has its own detection result and each has its own buffer. The evaluators are, Face Detection, Object Detection, and Pose Detection. The relationship between evaluators, detection results, and result buffers can be seen in Fig. 7.

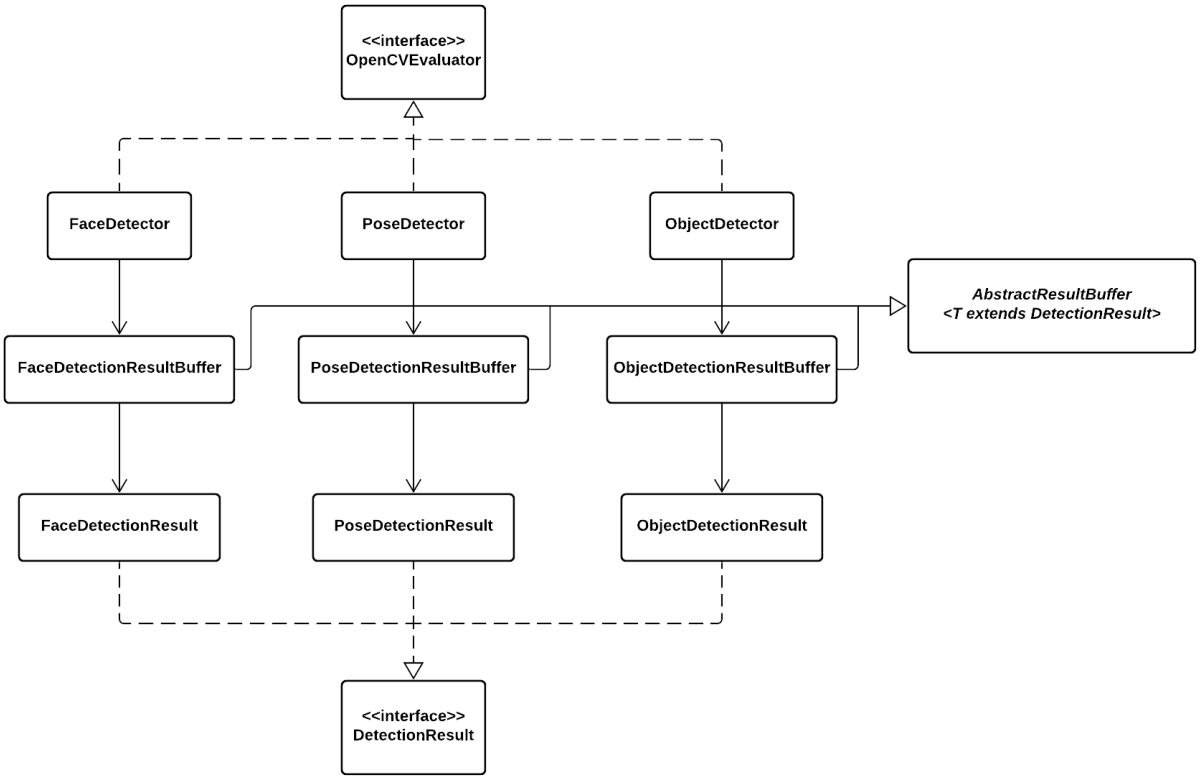


Fig. 7. A simplified UML displaying the relationship between evaluators and their interfaces.

The evaluator structure displayed in Fig. 7 enables extension by making it simple to add further computer vision analysis to the project. This would be accomplished by firstly making a new class, which implements the "OpenCVEvaluator" interface, to handle the new AI model. Secondly, one would create a class that implements the "DetectionResult" interface to represent the result of the evaluator. Lastly, the developer would create a class that extends the "AbstractResultBuffer" class to store the detection results created by the evaluator.

5.2.1 Face Detection

The face detection evaluator is used to detect faces in a given image. When the evaluator is initialized it loads the DNN model which is to be utilized. The model is interchangeable with models of similar input and output formats. When building this proof-of-concept, two different models were tested: ResNet-10 [37] and Single Shot Scale-invariant Face Detector (S^3FD) [38]. These two models were chosen to be tested because one is much faster than the other, which makes for more nuanced results.

After the images have been sent through the neural network, the output is processed. The detections generated by the model contain information about the confidence of the detection, as well as the left, right, top, and bottom coordinates. A threshold

is defined and used to filter out detections with too low confidence. The detections with sufficient confidence are then added to the detection result.

The detection result instantiated by this evaluator is of the class "FaceDetectionResult". It consists of a list of rectangles, each rectangle representing a detected face. Every rectangle has a top-left coordinate, as well as a width and a height. These instances are then added to the face detection result buffer.

5.2.2 Object Detection

The object detection evaluator also uses DNN models. The DNN that was implemented is called MobileNetSSD [39, 40], which detects 21 different common objects such as animals and vehicles.

It has an output format similar to the DNN of the face detection evaluator. That is, a rectangle around each detected object as well as the detection confidence. However, this model also outputs the name of the object that is detected. The detection result is of the class "ObjectDetectionResult". This result contains rectangles, as well as the names of the detected objects.

5.2.3 Pose Detection

The pose detection evaluator makes assumptions about the pose of a person in an image based on the spatial locations of anatomical key points. The model used for generating these key points is the neural network OpenPose [41]. OpenPose is an open-source system for multi-person 2D pose estimation.

TensorFlow is utilized to run an image through the OpenPose model. The output of the model is the coordinates for the detected key points. These key points are then used to build skeletons that are encapsulated by the ”Human” class. As the joints are also labeled, they can be used to build limbs for the skeletons. Lastly, a detection result of the class ”PoseDetectionResult” is generated containing a list of all detected humans.

5.3 Image processing

Although different AI models are utilized throughout the application, it is important to remember that a model will always be limited by the image it receives. As the project only explores openly available pre-trained models, many of the models require the image to be of a different format than what it was originally captured in. As such, images will have to be resized in order to match the input format of the model. Consequently, the results given by the AI models will differ depending on how these transformations are made.

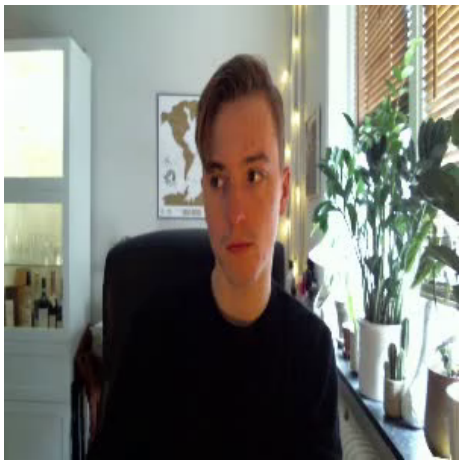
OpenCV offers a resize function that is used throughout the application. This function takes an argument that decides which interpolation method to use. If no

such argument was given, it defaults to bilinear interpolation, which is the one used in this project.

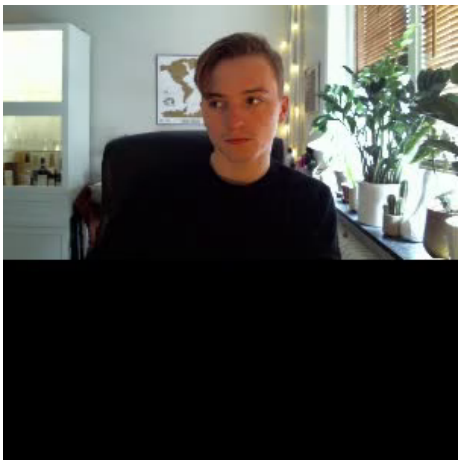
While these algorithms may do a good job of resizing the image, an issue arises when the original image has an aspect ratio different from the ratio of the model. For example, say that the input format of the AI model is 300x300, and the original image is 600x400. Utilizing the OpenCV resize function at this point to transform the 600x400 image to a 300x300 would lead to distortion. This will heavily affect the performance of the models as every person in the image is subsequently distorted. For example, the face detection models will have a harder time recognizing faces with unusual proportions as they have not been trained on distorted images.

In order to resolve this issue, a resize function was implemented that would preserve the ratio of the given image. The idea is that, while maintaining the proportions of the original image, the method resizes it to its largest possible size without exceeding any of the boundaries. In the above-given example, this would mean resizing the 600x400 image to a 300x200. The function then pushes either rows or columns of black pixels to the end of the picture in order to ensure that it matches the input dimensions of the given model. An example of what this may look like can be seen in Fig. 8.

(a) Original image



(b) Resized without aspect ratio preserved



(c) Resized with aspect ratio preserved

Fig. 8. Resizing a 1600x896 image to 300x300 with and without the aspect ratio being preserved.

This solution has two major downsides. Firstly, the image is made smaller than it otherwise would have been which makes detection more difficult. Secondly, the network will analyze the black pixels which result in wasted processing time.

A favorable solution would most likely be to find a model (or adjust the current one) to match the aspect ratio of the video recordings. However, as the purpose of this thesis has not been to find or build the most suitable models, this was considered to be outside of the scope.

5.4 Feature Implementation

The following sections will discuss how the different features are implemented. The different ways the evaluators from Section 5.2 are utilized will be covered.

5.4.1 Face Blur

OpenCV offers functionality for the blurring of images using different algorithms. As such, by using the face detection evaluator to get the position of faces in an image, the blurring of faces becomes possible. This result can be seen in Fig. 9. However, different alternatives have been explored to improve the face blurring quality. Two optimization algorithms have been implemented in an attempt at blurring faces not detected by the neural network.

Fig. 9. Image modified by the implemented face blur feature.
Source: Adapted from [42]

The first of the algorithms use body parts detected by the pose detection evaluator to estimate the location of a face. These body parts are namely the ears, eyes, nose, and neck. Whenever one of these body parts is detected at a place where the face detection found no result, a new detection is added. However, as of right now, the algorithm only uses one key point as a reference. The exact location and size of the face blur are therefore estimated.

The second attempt at improving face detection is by using an algorithm to reduce flickering. When viewing the generated videos it was noted that, even though a face has hardly moved, the blur sometimes disappeared for a few frames only to return shortly. An example of this can be seen in Fig. 10. The idea was to add detections for the frames that were missed by utilizing the buffer architecture. The algorithm identifies if the first and last frames contain any similar detections. The comparison is performed by checking if the first frame has face detections that overlap with detections of the last frame. If this is the case, an assumption that the detection should also be present in the frames in between is made. The application then loops through all the frames in between, adding the detection from the first frame to each one. The results of these improvements will be presented in Section 6.1.

Fig. 10. Four consecutive frames of a video processed by the application. As can be seen, the third frame should have a blur at approximately the same location as the other frames.

5.4.2 Tool Highlighting

The tool highlighting feature was implemented using the object detection evaluator. Since the evaluator stores information regarding the position as well as the title of objects, highlighting the objects becomes a rudimentary task. Using this information the application draws a rectangle around each detected object in the image. Each drawn rectangle also has a label containing the name of the object placed next to it. An example of this output format can be seen in Fig. 11. The cars are used due to a lack of relevant data sets which will be discussed in Section 6.2.

Fig. 11. Image modified by the implemented tool highlighting feature.
Source: Adapted from [43]

5.4.3 Ergonomic Risk Detection

The ergonomic risk detection feature is implemented using the pose detection evaluator. An ergonomic assessment is performed on each detected human in a frame. This assessment compares the coordinates of different key points to see if the human is currently in a potentially harmful pose. The definition of harmful poses, as provided by Solme, are as follows:

- An arm is kept at or above shoulder level
  - The wrist is above the shoulder
  - The elbow is above the shoulder
- An arm is stretched out from the body
  - There is a 45° angle between the elbow and shoulder, indicating a stretch away from the body
- An arm is kept at or below knee level
  - The wrist is below the knee
  - The elbow is below the knee

If one of these poses is detected, a boolean indicating the pose will be set to true. An example of a frame where such a pose is detected can be seen in Fig. 12.

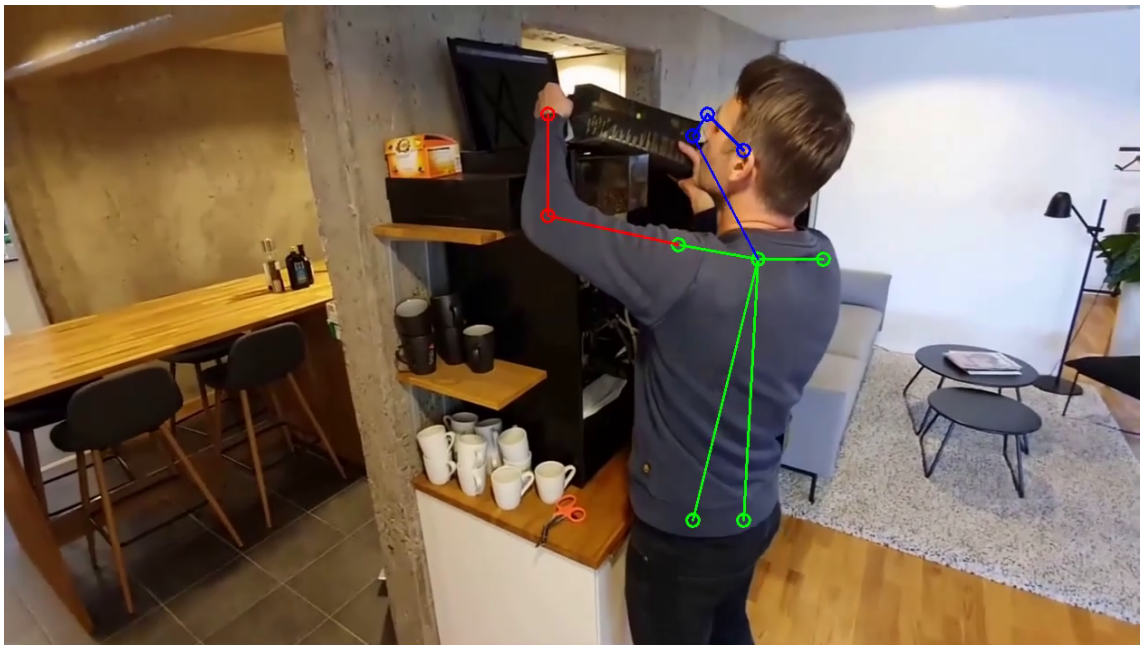


Fig. 12. Example frame where the wrist and elbow are detected above shoulder level.

The application checks the pose detection result buffer to see if it contains any of the non-ergonomic motions. If any such result is found, the program will check if it has not recently created timestamp for the same motion. The reasoning behind this is to avoid cluttering the output timestamp file.

When the application detects a new non-ergonomic motion, it creates a timestamp with the name of that motion, such as ”left elbow below knee”. The program will then check the current frame and use the frame rate of the input video to generate the current time. The timestamp will then be added to a JSON object. At the end of execution, the JSON object is written to an output JSON file. An example of such a timestamp file can be seen in Fig. 13.

```
{
    "Right arm stretch":[5.5,16.1],
    "Left wrist below knee":[11.8],
    "Left arm stretch":[3.6,14.2]
}
```

Fig. 13. Example of an output timestamp file. Each non-ergonomic motion has its own array of timestamps representing moments where the motion occurs in the video. The timestamps are given in seconds.

The JSON file is as of right now generated to demonstrate what the output format of this feature could look like. Solme could for example in the future be able to use this output to create timestamps in the AviX media player, allowing users to jump between the non-ergonomic motions.

5.4.4 Footstep Counting

A simple version of footstep counting has also been implemented using the information provided by the pose detection evaluator. It operates under the assumption that there is exactly one person in the given video and that this person appears in every frame.

To get the context of movement, two consecutive frames are compared against each other. The step counter is incremented if an ankle is rising in the second frame but not the first. To check if an ankle is rising, the application compares its y-coordinates in the different images. The application looks at one leg at a time. After counting a step taken with the right leg, it awaits a step taken with the left.

As of right now, the number of steps taken is displayed in the corner of the output video. An example of this can be seen in Fig. 14. This output format could be changed to fit Solme's needs in the future.

Fig. 14. An example of the current output format of the step counting feature.
Source: Adapted from [44]

6
Results and Discussion

The resulting prototype is a modular application that is adaptable. The modularity results from the possibility to add or remove features without having to restructure the program, since the features are not necessarily dependent on each other. However, there is the possibility to make the evaluators interact with each other by using the buffers. For example, as mentioned in Section 5.4.1, a face blur optimization utilizes body parts found by the pose estimation to make conclusions about the presence of a face. Adaptability is also a characteristic of the program since it is possible to switch between different DNN models depending on the use case.

All of the tests were completed using JDK 15 (Java Development Kit), OpenCV 4.3.0, TensorFlow 1.15.0, and OpenPose version 1.7.0. Further details on the face blur test setup will be presented in its section. All feature results will be presented in their corresponding sections as well as a discussion regarding performance, usability, and limitations.

6.1 Face Blur

The face blur test results consist of two neural network models tested on two videos. Both videos contain footage previously filmed for the AviX platform and are therefore relevant test material. Furthermore, the videos were recorded under good and poor conditions, as described in Section 4.3. Good and poor conditions in the videos should not be confused with the properties of the video files themselves. As the videos are not allowed for public use, example images will not be included.

Firstly, the specifications of the computer that ran the tests are presented. Secondly, the properties of the test videos are provided in Table I. Thirdly, the results are presented in Table II. fourthly, a discussion that interprets the results is held.

The computer system was equipped with an Intel i7-9700KF CPU @ 3.6 GHz 8 core processor, 16 GB of RAM, and was running Windows 10. The confidence threshold was set to 40% in all tests.

In the provided tables, 'Sensitivity' refers to the proportion of blurred faces to all faces according to $sensitivity = \frac{TruePositives}{TruePositives+FalseNegatives}$. FPS and FP stands for frames per second and false positives, respectively. The entries labeled *Optimized* refer to the attempted optimizations presented in Section 5.4.1.

Table I
Video properties of the good and poor conditions test videos

Property	Good	Poor
Resolution	640x480	640x480
FPS	30	30
Frames	288	300
Faces	288	373

Table II
Results from the face blur tests

(a) Good conditions

Model	Sensitivity [%]	FP	FPS
ResNet-10	100	3	16.0
ResNet-10 Optimized	100	3	11.3
S^3FD	100	0	1.3
S^3FD Optimized	100	0	1.3

(b) Poor conditions

Model	Sensitivity [%]	FP	FPS
ResNet-10	53	41	16.3
ResNet-10 Optimized	68	66	12.4
S^3FD	95	3	1.3
S^3FD Optimized	95	32	1.4

As seen in Table II, the ResNet-10 model [37] is approximately ten times faster than the S^3FD model [38]. The speed gained from using ResNet-10 needs to be weighed against its lower accuracy. The optimizations cause a significant speed decrease when using the ResNet-10 model, but have no impact on the S^3FD model. This is because the amount of instructions added by the optimization code is proportionally insignificant in comparison to the S^3FD model instructions.

The speed of the models was measured by the execution time of the program. This will change depending on the hardware that the program is being run on, so the absolute values are not of interest. However, the relative speed is interesting as a metric of comparison since this should not be as hardware dependent.

Every face was detected in the video with good conditions. In this scenario, the ResNet-10 model is favorable to use since it saves processing time. However, in the video with poor conditions, the S^3FD model had a significantly higher sensitivity, rendering it favorable.

The optimizations do not increase the number of true positives when the accuracy of the model is high, as is apparent from the good conditions test. However, the optimizations can increase the sensitivity when the accuracy of the model is low, such as the ResNet-10 model under poor conditions. False positives are more likely to be added by the optimizations when the conditions are poor.

There is the option to raise the detection threshold, which will decrease the number

of false positives as well as the number of true positives. In the context of face blurring, a false negative is considered more detrimental than a false positive. A false positive could at most lead to an obstructed view of the manufacturing process, while a false negative could lead to the person's face being identifiable. This means that it is acceptable to use a low detection threshold. Furthermore, it implies that the optimizations are desirable to use for the ResNet-10 model.

6.2 Tool Highlighting

Although an evaluator that supports object detection was made, it was not possible to find a model that supports relevant objects. Since the target videos are taken in a factory setting, the application requires an object detection model trained at detecting tools and machines used in these factory settings. However, tool highlighting is still possible to achieve by fitting the current evaluator to a model that is trained on relevant objects.

Furthermore, no data set containing relevant objects was found, which could have been used to train a model. Available data sets where objects have already been annotated include ImageNet [45], Visual Genome [46], COCO [47], Google's Open Images [48] and more. The problem with these data sets is that they only include the most common everyday objects which are uninteresting for Solme's use case.

Creating a data set requires a large amount of work since a multitude of pictures has to be annotated with bounding boxes around objects of interest. Furthermore, training a model on the finished data set would also prove time-consuming. Considering that Solme did not think this feature was of high priority, this was not pursued further.

6.3 Ergonomic Risk Detection

A video was produced under controlled conditions where the application was able to consistently detect the person going through all of the eight hazardous movements, subsequently generating accurate timestamps. A similar video was produced under dimmer lighting conditions where the poses were not as consistently detected. It can as such be concluded that lighting is a factor when it comes to the accuracy of the pose estimation.

It was noted when setting up the controlled video conditions that it is required that the whole body part is present in the video in order for OpenPose to detect it. The body parts that this occurs most frequently to are hands and feet. For some of the hazardous movements to be detected, it is required that the hands are detected, causing this to be an issue. As seen in Fig. 15, once the hands were outside the screen the joints in the arms could no longer be accurately estimated subsequently not appearing at all.

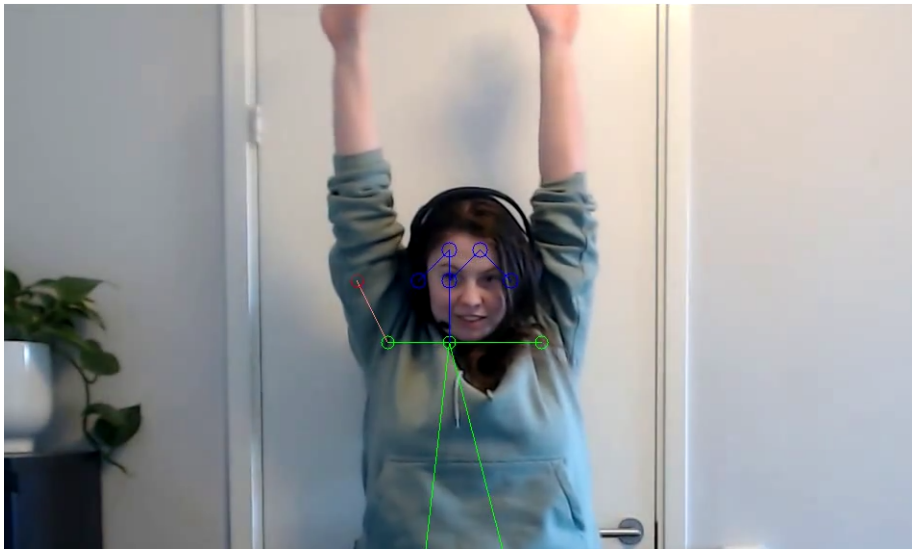


Fig. 15. Image where hands outside of the camera view renders OpenPose unable to detect arms.

Similar issues were present in the video representing realistic conditions for a typical AviX video. The lighting was less of a problem since factories are usually well lit. However, issues occurred due to the angles the workers were filmed from and the fact that they were covered by equipment, or partly outside the frame.

In conclusion, the program detected all movements of interest provided that OpenPose had detected the pose accurately. The accuracy of the skeleton is therefore the factor that has room for the biggest improvements. Those improvements can to a large extent be achieved with more careful filming.

6.4 Footstep counting

After the initial research period that took place during the first weeks of the project, it was deemed that footstep counting would take a lower priority among the features due to its complexity. No DNN was found that directly supports the implementation of footstep counting. Instead, the solution is based on analyzing the body parts detected by the pose estimation.

Solme deems that a footstep has occurred every time a foot touches down on the floor. By using the ankle coordinate generated by the pose estimation, footsteps could be counted under very strict conditions.

- Only one person can be in the frame.
- The person has to be standing on the floor at the very start.
- The person has to be close enough for the pose estimation to detect them, but far enough to see their lower body.
- The person should walk towards the camera at a constant distance from the camera, meaning that the camera has to move backwards. For the reason that

they should remain the same size in the camera's view.
- The person's feet should only move in the camera view if they are walking, meaning that the camera should not pan up and down.

Under these strict conditions, the implementation of footstep counting does work. Unfortunately, the method of solely analyzing pose estimation provides very poor results in realistic conditions. Additional footsteps will be counted when the camera is panning up and down. If a person leaves the frame and comes back it is hard to establish that they are the same person. It is also hard to keep track of multiple people who stay on screen because of how the pose estimation misses detections. In conclusion, the data provided by pose estimation is not enough to infer a proper footstep count.

7
Conclusions and Future Work

In this chapter, conclusions that can be drawn from this work are presented. Additionally, potential future work is suggested.

7.1 Conclusions

The final product has fulfilled the requirements set out as a proof-of-concept for Solme and has also highlighted key potentials and limitations. The goal was to construct a modular platform that enables changing what type of analysis is to be run with ease, which has been completed. Moreover, the final version of the primary feature, face blurring, is deemed to perform well with 100% sensitivity and 0 false positives in 288 frames under good conditions. Under poor conditions, provided the right model, the feature performs slightly worse with 95% sensitivity and 32 false positives in 300 frames.

In terms of what has been identified from the proof-of-concept, evaluation of the face blurring and ergonomic risk detection underlines the potential that these two features have. Additionally, obstacles for the step counting and tool highlighting features have been identified and presented. The produced platform provides a strong basis for further automating manual tasks currently performed by Solme.

Furthermore, the evaluation also presented limitations of using computer vision to automate tasks. An especially prominent limitation is the quality of the average video processed by AviX. Poorly lit spaces, low image quality, and shaky footage all reduce the models' ability to yield good results. Moreover, the lack of pre-trained models that have been trained in a manufacturing setting leads to drawbacks. Lastly, by only having access to a single camera angle, depth information is lost. This in turn limits the possibilities of both the step counting and ergonomic risk detection features.

7.2 Future Work

In terms of what future work that can be done, a solid foundation that Solme can choose to continue working on if they so please has been created. Additionally, further training models for the face blurring and tool highlighting features would increase the performance of the end-product. This could be accomplished by training the models on data sets containing workers wearing helmets and industry tools

respectively.

Moreover, if the software had access to depth, a general standard ergonomic evaluation could be implemented rather than creating timestamps for potentially harmful positions for later evaluation. One such general standard is REBA [29]. In such an implementation, the feature could instead give the workstation an ergonomic score and advise employers whether a workstation needs to implement some change.

Lastly, another programming language could be used to implement a similar application. There are languages such as Python which are better supported by ML libraries compared to Java. Furthermore, Python has a large amount of online tutorials and resources for building computer vision projects.

Bibliography

[1] "Factory worker girl hard helmet indoors industrial industry," *PxHere*, Jun. 10, 2018. [Online]. Available: https://pxhere.com/en/photo/1558739 (visited on 05/05/2021).

[2] C. Behle, "The multiple benefits of video surveillance in manufacturing," *AXIS Secure Insights*, Jan. 4, 2021. [Online]. Available: https://www.axis.com/blog/secure-insights/video-surveillance-manufacturing/ (visited on 02/12/2021).

[3] "Solme ab," *AviX*, n.d. [Online]. Available: https://www.avix.se/om-oss (visited on 05/09/2021).

[4] "Avixsuite," *AviX*, n.d. [Online]. Available: https://www.avix.se (visited on 05/09/2021).

[5] C. Lamarre, "What is line balancing and how to achieve it," *Tulip*, Aug. 14, 2019. [Online]. Available: https://tulip.co/blog/lean-manufacturing/what-is-line-balancing-and-how-to-achieve-it/ (visited on 05/09/2021).

[6] Juran, "Guide to failure mode and effect analysis – fmea," *Juran*, Apr. 2, 2018. [Online]. Available: https://www.juran.com/blog/guide-to-failure-mode-and-effect-analysis-fmea/ (visited on 05/09/2021).

[7] A. Tezel, "Introduction to smed: A neglected method in lean construction," *Lean Construction Blog*, Aug. 18, 2016. [Online]. Available: https://leanconstructionblog.com/Single-Minute-Exchange-of-Dies-A-Neglected-Method-in-Lean-Construction.html (visited on 05/09/2021).

[8] D. Harwell and E. Dou, "Huawei tested AI software that could recognize uighur minorities and alert police, report says," *The Washington Post*, Dec. 8, 2020. [Online]. Available: https://www.washingtonpost.com/technology/2020/12/08/huawei-tested-ai-software-that-could-recognize-uighur-minorities-alert-police-report-says/ (visited on 04/23/2021).

[9] Ó. F. Civieta and J. Ravindran, "FBI warns of the rise of 'deepfakes' in coming months and explains how to spot them easily," *Business Insider*, Mar. 29, 2021. [Online]. Available: https://www.businessinsider.com/fbi-investigation-generated-computer-ai-artificial-intelligence-abuse-misinformation-porn-2021-3?r=US&IR=T (visited on 04/23/2021).

[10] R. Schmelzer, "Should we be afraid of ai?" *Forbes*, 2019-10-31. [Online]. Available: https://www.forbes.com/sites/cognitiveworld/2019/10/31/should-we-be-afraid-of-ai/?sh=5638db7e4331 (visited on 04/27/2021).

[11] "The eu general data protection regulation (gdpr) and face images," *D-ID*, Sep. 2018. [Online]. Available: https://www.deidentification.co/wp-

content/uploads/2018/09/White-Paper-GDPR-and-D-ID.pdf (visited on 04/29/2021).

[12] K. M. Fincher and P. E. Tetlock, "Perceptual dehumanization of faces is activated by norm violations and facilitates norm enforcement.," *Journal of Experimental Psychology: General*, vol. 145, no. 2, pp. 131–146, Feb. 2016. DOI: 10.1037/xge0000132. [Online]. Available: https://doi.org/10.1037/xge0000132.

[13] "The advantages of ergonomics," *Oregon OSHA*, n.d. [Online]. Available: https://www.oshatrain.org/courses/pdf/ergoadvantages.pdf (visited on 03/28/2021).

[14] T. R. Waters and R. B. Dick, "Evidence of health risks associated with prolonged standing at work and intervention effectiveness," *Rehabilitation Nursing*, vol. 40, no. 3, pp. 148–165, May 2015. DOI: 10.1002/rnj.166. [Online]. Available: https://doi.org/10.1002/rnj.166.

[15] "Computer vision," *IBM*, 2021-05-12. [Online]. Available: https://www.ibm.com/topics/computer-vision (visited on 04/27/2021).

[16] S. A. Papert, "The summer vision project," *Artificial Intelligence group*, Jul. 1, 1966. [Online]. Available: https://dspace.mit.edu/handle/1721.1/6125 (visited on 04/22/2021).

[17] "Artificial intelligence (ai)," *IBM*, 2020-06-03. [Online]. Available: https://www.ibm.com/cloud/learn/what-is-artificial-intelligence (visited on 04/11/2021).

[18] "What is ai?" *Council of Europe*, 2021. [Online]. Available: https://www.coe.int/en/web/artificial-intelligence/what-is-ai (visited on 04/12/2021).

[19] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed., ser. Adaptive Computation and Machine Learning. Cambridge, MA: MIT Press, 2018, 504 pp., ISBN: 978-0-262-03940-6.

[20] E. Hjelmås and B. K. Low, "Face detection: A survey," *Computer Vision and Image Understanding*, vol. 83, no. 3, pp. 236–274, Sep. 2001. DOI: 10.1006/cviu.2001.0921. [Online]. Available: https://doi.org/10.1006/cviu.2001.0921.

[21] "Ai vs. machine learning vs. deep learning vs. neural networks: What's the difference?" *IBM*, May 27, 2020. [Online]. Available: https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks.

[22] S.-C. Wang, "Artificial neural network," in *Interdisciplinary Computing in Java Programming*, Springer US, 2003, pp. 81–100. DOI: 10.1007/978-1-4615-0377-4_5. [Online]. Available: https://doi.org/10.1007/978-1-4615-0377-4_5.

[23] "Artificial neural network," *Wikimedia*, Feb. 22, 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:Artificial_neural_network.svg (visited on 05/14/2021).

[24] "Avix ergo," *AviX*, n.d. [Online]. Available: https://www.avix.se/lean-production-verktyg/avix-ergo (visited on 04/27/2021).

[25] A. Devaux, N. Paparoditis, F. Precioso, and B. Cannelle, "Face blurring for privacy in street-level geoviewers combining face, body and skin detectors," Institut Géographique National, Jan. 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.392.525&rep=rep1&type=pdf.

[26] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent, "Large-scale privacy protection in google street view," in *2009 IEEE 12th International Conference on Computer Vision*, IEEE, Sep. 2009. DOI: 10.1109/iccv.2009.5459413. [Online]. Available: https://doi.org/10.1109/iccv.2009.5459413.

[27] K. Brkic, I. Sikiric, T. Hrkac, and Z. Kalafatic, "I know that person: Generative full body and face de-identification of people in images," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, Jul. 2017. DOI: 10.1109/cvprw.2017.173. [Online]. Available: https://doi.org/10.1109/cvprw.2017.173.

[28] R. Venkatesulu and S. Venkatesh Koundinya, "Human motion analysis based on human pose estimation," M.S. thesis, CHALMERS UNIVERSITY OF TECHNOLOGY, SE-412 96 Gothenburg, 2020.

[29] M. Middlesworth, "A step-by-step guide to the reba assessment tool," *ErgoPlus*, Oct. 17, 2020. [Online]. Available: https://ergo-plus.com/reba-assessment-tool-guide/ (visited on 04/10/2021).

[30] D. Kerr, J. Pengilley, and R. Garwood, "Assessment and visualisation of machine tool wear using computer vision," *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 7-8, pp. 781–791, May 2005. DOI: 10.1007/s00170-004-2420-0. [Online]. Available: https://doi.org/10.1007/s00170-004-2420-0.

[31] E. C. Latorre, M. D. Zuniga, E. Arriaza, F. Moya, and C. Nikulin, "Automatic registration of footsteps in contact regions for reactive agility training in sports," *Sensors*, vol. 20, no. 6, p. 1709, Mar. 2020. DOI: 10.3390/s20061709. [Online]. Available: https://doi.org/10.3390/s20061709.

[32] "Worker metal steel," *Pixbay*, Aug. 11, 2016. [Online]. Available: https://pixabay.com/photos/worker-metal-steel-manufacturing-4395768/ (visited on 05/05/2021).

[33] M. Singh, "How to blur faces in images using opencv in python," *TechGeekBuzz*, [Online]. Available: https://www.techgeekbuzz.com/how-to-blur-faces-in-images-using-opencv-in-python/ (visited on 03/02/2021).

[34] "Detected-with-yolo–schreibtisch-mit-objekten," Jan. 14, 2019. [Online]. Available: https://commons.wikimedia.org/wiki/File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg (visited on 05/05/2021).

[35] M. Blume, "Kansas city assembly," *Wikimedia*, Oct. 8, 2008. [Online]. Available: https://commons.wikimedia.org/wiki/File:Kansas_City_Assembly.png (visited on 05/05/2021).

[36] D. R. Bassett, L. P. Toth, S. R. LaMunion, and S. E. Crouter, "Step counting: A review of measurement considerations and health-related applications," *Sports Medicine*, vol. 47, no. 7, pp. 1303–1315, Dec. 2016. DOI: 10.1007/

s40279-016-0663-1. [Online]. Available: https://doi.org/10.1007/s40279-016-0663-1.

[37] V. Feng, "An overview of resnet and its variants," *Towards data science*, Jul. 15, 2017. [Online]. Available: https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035 (visited on 05/05/2021).

[38] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S^3fd: Single shot scale-invariant face detector," *CoRR*, vol. abs/1708.05237, Aug. 2017. arXiv: 1708.05237. [Online]. Available: http://arxiv.org/abs/1708.05237.

[39] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, Jun. 17, 2017. arXiv: 1704.04861. [Online]. Available: http://arxiv.org/abs/1704.04861.

[40] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, Dec. 8, 2015. arXiv: 1512.02325. [Online]. Available: http://arxiv.org/abs/1512.02325.

[41] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *CoRR*, vol. abs/1812.08008, 2018. arXiv: 1812.08008. [Online]. Available: http://arxiv.org/abs/1812.08008.

[42] "Industrial workers in uniform and safety equipment relaxing on a break drinking coffee and talking inside factory," *freepik*, 2020. [Online]. Available: https://www.freepik.com/free-photo/industrial-workers-uniform-safety-equipment-relaxing-break-drinking-coffee-talking-inside-factory_11030706.htm (visited on 05/11/2021).

[43] M. Blume, "Audi s4 (20100325-dsc01393)," *Wikimedia*, Mar. 25, 2010. [Online]. Available: https://commons.wikimedia.org/wiki/File:Audi_S4_(20100325-DSC01393).jpg (visited on 05/11/2021).

[44] D. Larson, "Jimmy likes his treadmill," *vimeo*, Dec. 30, 2015. [Online]. Available: https://vimeo.com/150376758 (visited on 05/11/2021).

[45] "An update to the imagenet website and dataset," *ImageNet*, 2021. [Online]. Available: https://www.image-net.org/update-mar-11-2021.php (visited on 05/07/2021).

[46] "Visual genome," *Visual Genome*, 2021. [Online]. Available: https://visualgenome.org/ (visited on 05/07/2021).

[47] "Coco dataset," *COCO*, 2021. [Online]. Available: https://cocodataset.org/#home (visited on 05/07/2021).

[48] "Open images dataset," *Google Open Source*, 2021. [Online]. Available: https://opensource.google/projects/open-images-dataset (visited on 05/07/2021).

[49] "Intel," *OpenCV*, 2021. [Online]. Available: https://opencv.org/intel/ (visited on 04/29/2021).

[50] "About ffmpeg," *FFmpeg*, 2019. [Online]. Available: https://www.ffmpeg.org/about.html (visited on 04/29/2021).

[51] "Install tensorflow java," *TensorFlow*, 2021. [Online]. Available: `https://www.tensorflow.org/install/lang_java` (visited on 02/16/2021).

A

Appendix 1

A.1 Library Description

A.1.1 OpenCV

OpenCV is a large open-source computer vision library and is as such not limited to any particular feature. It was originally developed by Intel in 1998 and became publicly available in 2000 [49]. Intel is still funding the core development team and maintains the build farm of OpenCV [49]. OpenCV supports the Java programming language. However, while this may be the case, the majority of online resources and learning material for this library is written in Python or C++.

With OpenCV, it is also possible to satisfy all the video processing needs as it is a video and image processing library. That means that it supports blurring, drawing pose estimation, reading a video file and exporting a video file. Despite that, it does not have support for audio processing so the exported video file will be missing the audio. The audio can be added to the output video file using for example 'ffmpeg' [50].

A.1.2 TensorFlow

TensorFlow is an open-source platform for machine learning. By combining TensorFlow with OpenCV or ND4J, it can serve as a reliable way to create and deploy machine learning models. TensorFlow has a community hub on their website where hundreds of trained models can be downloaded and deployed for free [51]. TensorFlow was used in this project to deploy the pose estimation deep neural network.