Utrecht University

HoloPiano

Musical keyboard detection with the HoloLens mixed reality headset

Master's thesis

ICA-4020162

Author:
Thomas Aalbers

Supervisor:
Dr. ing. S.C.J. Bakkes

External supervisors:
I. van Voorden, MSc.
F. Németh
Ing. M. van Os

Second examiner:
Dr. Z. Yumak

November 21, 2019

Abstract

Mixed reality headsets and applications are gaining popularity for entertainment and educational purposes. They can be used to acquire new skills or assist users in operating complex devices. Detection and localization of objects in the user's vicinity are necessary to create convincing and effective tools. So far mostly real-world image markers are used to position holograms on real objects. They require printing and manual placing which might break the mixed reality experience. Accurately positioning objects with imaging sensors without the use of markers proves to be challenging and is currently state-of-the-art research.

In this study we explore the effectiveness of the RGB, reflectivity and depth imaging sensors of the HoloLens mixed reality headset in natural, artificial and dim lighting setups on a new manually designed dataset. We are also looking at the difference in performance for the dim lighting setup and if natural or artificial light influence infrared-based imaging sensors. Our imaging subject is a musical keyboard in an indoor setting.

We have concluded that the RGB, reflectivity and depth imaging sensors all pose candidates for computer vision tasks on the HoloLens mixed reality headset. However no specialized detectors for the reflectivity and depth fisheye lenses have been found in the literature. Using depth-aware and 6D pose estimation detectors requires prior experience and proves to be time-consuming to implement. Various cloud-based computer vision portals exist like CustomVision.ai that enable training an object detector but with limited metrics and export functionality.

Based on the answers to the research questions we conclude that HoloLens imaging sensors are able to solve computer vision tasks in natural, artificial and dim lighting setups. A reflectivity imaging sensor will outperform an RGB sensor for object detection tasks in natural, artificial and dim lighting setups and more research is required for depth imaging sensor performance. Careful planning on imaging sensors, lighting setups and detectors should be done when implementing new mixed reality applications to ensure optimal performance.

Acknowledgements

Many people have supported this work in one way or another and I would like to show my gratitude for their help.

First of all, I would like to thank Betabit for offering me this opportunity. To be able to research the latest technologies in a company that feels like family has been of tremendous value to me. The countless events, freedom, superb working environment and eager-to-help colleagues all make this a very valuable experience.

I would also like to thank my supervisor Sander Bakkes for always being ready to help with the multitude of challenges and questions I come across during this work. My impromptu visits and brain-dumps have been answered with quality guidance even under pressing deadlines.

I want to thank my external supervisor Marco van Os for the numerous sessions to decide on this exciting project for my master thesis. Together with my external technical supervisor Ferenc Németh; thank you for the guidance from Betabit.

I want to thank Zerrin Yumak for being my second examiner. I also want to thank Ronald Poppe for taking the time and guiding me in the field of Computer Vision.

A special thanks goes out to Anne Frijns, who helped me understand the common beginner errors and intricacies of piano teaching with her valuable experience. Another special thanks goes out to Berry van Someren, for helping me with Computer Vision and thesis writing.

I would also like to thank my Betabit colleagues for making this an enjoyable ride, but especially Bob van Hooff and Magiel van Gaalen for tackling all kinds of HoloLens challenges, and Quinten Heijn for reviewing my documents.

Lastly I want to express my thanks to my beloved ones and friends that support me through this work, you make it a pleasurable experience. There are just too many to list. Thank you!

Contents

1 Introduction **5**
- 1.1 Context . . . 5
- 1.2 Research challenges . . . 5
- 1.3 Problem statement and research questions . . . 6
- 1.4 Contributions . . . 6
- 1.5 Limitations . . . 7
  - 1.5.1 Dataset sample size . . . 7
  - 1.5.2 Generalizability . . . 7
  - 1.5.3 No specialized detectors . . . 7
  - 1.5.4 Object detection . . . 7
- 1.6 Outline . . . 7

2 Literature Study **8**
- 2.1 Data sources and search strategy . . . 8
- 2.2 Musical keyboard . . . 8
- 2.3 Piano teaching . . . 9
- 2.4 HoloLens . . . 10
  - 2.4.1 Spatial mapping . . . 11
  - 2.4.2 HoloLens Spatial Locatability . . . 12
  - 2.4.3 Sensors . . . 12
  - 2.4.4 HoloLens 2 . . . 12
  - 2.4.5 Specifications . . . 12
- 2.5 Computer vision . . . 13
  - 2.5.1 PianoAR: Traditional methods for musical keyboard detection . . . 13
  - 2.5.2 Designing quality features . . . 14
  - 2.5.3 Convolutional neural networks . . . 14
  - 2.5.4 Image recognition . . . 16
  - 2.5.5 YOLOv3: Object detection . . . 17
  - 2.5.6 MaskRCNN: Image segmentation . . . 17
  - 2.5.7 6D Pose estimation . . . 18
  - 2.5.8 2D or not 2D (using the depth camera) . . . 18
  - 2.5.9 State-of-the-art RGB detectors . . . 19
  - 2.5.10 State-of-the-art RGB-D detectors . . . 20
  - 2.5.11 Cloud-based detectors . . . 21
  - 2.5.12 Vuforia image target . . . 21
  - 2.5.13 Vuforia model target . . . 21
  - 2.5.14 Music Everywhere tracking and user performance . . . 22

3 Methodology **24**
- 3.1 Requirements . . . 24
- 3.2 Ethical considerations . . . 25
- 3.3 Sampling strategy . . . 25
- 3.4 Validity . . . 25
  - 3.4.1 Construct validity . . . 25
  - 3.4.2 Face validity . . . 25
  - 3.4.3 Content validity . . . 25
  - 3.4.4 Criterion validity . . . 25

3.5 Reliability . . . 26
3.5.1 Test-retest reliability . . . 26
3.5.2 Inter-rater reliability . . . 26
3.5.3 Parallel forms reliability . . . 26
3.5.4 Internal consistency reliability . . . 26
3.6 Development . . . 26
3.7 Time plan . . . 26

4 Dataset **28**
4.1 Data . . . 28
4.2 Independent variables . . . 28
4.3 Pre-processing . . . 29
4.4 Choreography . . . 30
4.5 Quality . . . 30
4.6 Ground truth . . . 31

5 Experiment **33**
5.1 CustomVision.ai . . . 33
5.2 Training . . . 34
5.3 Metrics . . . 34
5.4 Evaluation . . . 35

6 Results and discussion **37**
6.1 Research Question 1 . . . 37
6.1.1 Imaging sensors . . . 37
6.1.2 Detectors . . . 38
6.1.3 Support . . . 38
6.1.4 Summary . . . 39
6.2 Research Question 2 . . . 39
6.2.1 Summary . . . 40
6.3 Experiment . . . 40
6.3.1 Research Question 3 . . . 40
6.3.2 Research Question 4 . . . 41
6.3.3 Overexposure . . . 42
6.3.4 Depth . . . 42
6.3.5 Precision or recall . . . 42
6.3.6 Internal consistency . . . 43

7 Conclusion **45**
7.1 Answers to the research questions . . . 45
7.2 Answer to the problem statement . . . 46
7.3 Future research . . . 47

A Time plan **48**

B HoloLens 1 and 2 specifications **49**

Chapter 1

Introduction

Detecting objects for augmented reality applications is a challenging task that involves powerful device hardware and sophisticated algorithms. In this work we will research the performance of various imaging sensors of the Microsoft HoloLens headset for detecting a musical keyboard in natural, artificial and dim lighting setups. The approaches will be compared to determine the usefulness of the various sensors for computer vision.

1.1 Context

Mixed reality headsets and applications are gaining popularity for entertainment and educational purposes. They can be used to acquire new skills or assist users in operating complex devices. Detection and localization of objects in the user's vicinity are necessary to create convincing and effective tools.

During this project we will use the HoloLens (figure 2.2) mixed reality headset. It is a wireless head mounted device that scans the 3-dimensional environment to create realistic augmented holograms on its see-through display glasses. It features real-time voice recognition, gesture recognition, audio playback and spatial positioning to create a complete mixed reality experience.

The future goal is to create a HoloLens application that visually assists the user in playing along on a keyboard instrument to a live song. It will visually highlight the key holograms on when and how to play the notes and which keys are currently pressed. Strategies will be implemented to maximize learning potential by choosing appropriate practice modes based on user performance and song characteristics.

This project "HoloPiano" is the first iteration of the future goal. In this work we will research the detection of the musical keyboard by the HoloLens imaging sensors. We also want to develop an understanding the types of imaging sensors and when to use them in what lighting setups.

1.2 Research challenges

This section shows the research challenges that are being addressed in this work.

Traditional object detection applications use an RGB camera, 2D markers or objects with very distinctive features. However, the HoloLens is an advanced mixed reality device that sports many sensors to understand its surroundings. With these sensors it can detect complex objects in 3D space. Multiple sensors exist and finding out how they compare is valuable information when designing new mixed reality HoloLens applications.

In this work we will use object detection on a musical keyboard. The available musical instrument is a Yamaha PSR-520 MIDI digital keyboard with 61 keys and an 159mm octave width. For this project we will focus on this specific single instrument setup.

A useful feature is that most musical keyboards tend to align nicely with the world space by the two orientation dimensions pitch and roll. This means that the plane created by the top surface of the keyboard keys aligns with the ground plane and only the yaw orientation determines its final orientation. Solving the entire pose can be done by detecting the two image space locations for the left and right top foremost points of the musical keyboard and knowing the musical keyboard

width or key-count in advance. Even though these heuristics are available, we will not resort to them so that the research translates better to other objects that cannot offer this luxury.

Most augmented reality applications use a color camera to solve a computer vision task. Color cameras are very common but might not be ideal when dealing with maintenance or repair tasks that are situated in low-light rooms.

The HoloLens features a range of imaging sensors that assist in detecting objects in the vicinity. Part of this collection are color, grayscale, infrared reflectivity and infrared depth sensors featuring different orientations including far and near modes with different refresh rates. It is unclear how the different sensors and modes influence the performance of keyboard detection in different lighting setups. The depth and reflectivity sensors come with their own challenges and limitations. They use an infrared light and filter to sample the environment. In this work we will find out if they are worth the trade-off.

The HoloLens is not always connected to the internet. When there is no Wi-Fi connection, the application should still be able to do image detection tasks offline and offer a responsive solution. This means that a suitable computer vision technique can perform the detection on-device, without cloud computing and in a matter of seconds.

A sufficiently large and diverse dataset of various HoloLens imaging sensors should be designed, created and annotated to objectively measure their effectiveness for computer vision. Effective deep learning approaches typically require a large dataset to prevent overfitting[1].

An understanding of the best performing computer vision techniques should be developed. The state-of-the-art is considered the best performing detector on a specific challenge and dataset in time. We are looking for recent time-critical state-of-the-art detectors that operate on a variety of imaging sensors that the HoloLens provides.

1.3 Problem statement and research questions

From the restrictions and challenges in Section 1.1 and 1.2 our problem statement is formulated as follows.

> **PS/Problem statement:** *To what extent are HoloLens imaging sensors able to solve computer vision tasks in natural, artificial and dim lighting setups?*

To tackle the problem statement, we first address which imaging sensors there are, what computer vision technique can be used for comparing, and how the imaging sensors compare in natural, artificial and dim lighting setups. The research questions are as follows.

> **RQ1/Research question 1:** *Which HoloLens imaging sensors are supported by state-of-the-art detectors?*
>
> **RQ2/Research question 2:** *Which state-of-the-art detector is suitable for comparing imaging sensors?*
>
> **RQ3/Research question 3:** *To what extent do infrared imaging sensors improve detector performance in a dim lighting setup compared to an RGB imaging sensor?*
>
> **RQ4/Research question 4:** *To what extent do natural and artificial light influence infrared imaging sensor detector performance?*

1.4 Contributions

This work has two main contributions:

- **A musical keyboard object detection dataset is created.** For comparing the HoloLens imaging sensors, a dataset is recorded consisting of 50 randomly selected images per sensor per lighting setup resulting in a total of 450 samples. The samples include color, depth and reflectivity images recorded in natural, artificial and dim lighting setups.
- **Performance of types of imaging sensors is compared in natural, artificial and dim lighting setups.** The performance of HoloLens color, depth and reflectivity imaging sensors is compared in natural, artificial and dim lighting setups to gain insight in usefulness of sensor types for challenging computer vision tasks.

1.5 Limitations

The identified limitations of the study are described in the following subsections.

1.5.1 Dataset sample size

The created dataset contains 50 samples per setup, a lower number of samples is at risk of overfitting on the training data[1]. A suggestion is to perform experiments on datasets with at least 1000 samples per setup, which was the original plan.

1.5.2 Generalizability

The main experiment in the study is a quantitative laboratory experiment. Laboratory experiments naturally have high internal validity and low external validity. To reach higher external validity, field research is necessary and multiple datasets and detectors should be compared. Currently, similar studies do not exist in the literature and solving this weakness in the research design does not fit in the project's constraints.

1.5.3 No specialized detectors

An RGB detector is used in the experiment for the depth and reflectivity imaging sensors. Specialized depth-aware detectors exist but they could not be implemented within the project constraints. This means that the performance metrics for the depth and reflectivity imaging sensors could have been higher if specialized detectors were used.

1.5.4 Object detection

The experiment uses object detection instead of full 6D pose estimation. 6D Pose estimation could not be implemented within the project constraints. Object detection calculates detector performance by comparing predicted and ground truth bounding boxes of classes in images. See Subsection 2.5.5 for a description of object detection.

Object detection uses axis-aligned bounding boxes to indicate objects in an image. The bounding boxes also contain pixels that do not belong to the object. The detector will therefore include these pixels as part of its understanding of the class. If different types of input is used for the same detection problem, a bounding box might be different for a depth and an RGB image. This might influence the detector performance. Using 6D pose estimation will offer the same answer for all imaging sensors, this is likely a more robust comparison.

The difference in image resolution and camera projection of the imaging sensors might influence the performance metrics. This is the case because the fish eye lense warps the image compared to the perspective projection. The ground truth has been hand-annotated with approximately the same pixel precision even though the scale is different. Luckily the IoU metric is scale-invariant so the effects might be small.

1.6 Outline

In Chapter 2 a literature study is covering relevant piano specifications and teaching considerations, the mixed-reality HoloLens headset, historical image processing techniques and state-of-the-art convolutional neural network approaches for solving computer vision problems. Chapter 3 describes the methodology of this research; what state-of-the-art computer vision technique is appropriate to use with the HoloLens sensors and how to adequately compare imaging sensors for musical keyboards. Chapter 4 describes how the ground-truth of the dataset is created. Chapter 5 describes the experiments of this research; how are the imaging sensors compared and what evaluation metrics determine the performance best. Chapter 6 reports the outcome of the experiments. Chapter 7 formulates the conclusion in this work.

Chapter 2

Literature Study

2.1 Data sources and search strategy

This section explains the strategy for gathering the required sources and related work.

In essence this work is about the computer vision research domain and the HoloLens augmented reality device. At the start of the project the possibilities of implementing a piano teaching application were explored and for this reason a piano teaching interview has been included. The planned data gathering strategy is as follows:

- Consult professors and alumni that research computer vision at the University of Utrecht for paper and detector recommendations.
- Search Google scholar for the keywords: "HoloLens Piano Keyboard Mixed Augmented Reality" and "Piano Image Detection".
- Search the website; PapersWithCode for the task 6D pose estimation[2] (New in February 2019). To find state-of-the-art papers to solve computer vision problems from RGB(-D) images.
- Interview a piano teacher for best practices and common beginner mistakes to successfully implement piano holograms.

There is a preference for papers containing a reference to the source code of the implementation. The code requirement can be set as a filter for the website PapersWithCode. If the paper does not have a code implementation, but seems the best approach to solve the problem; the researchers will be asked for a possible code implementation. The latest and currently state-of-the-art papers will be researched for possible solutions.

The inclusion criteria for the papers are:

- Paper describes an image detector that can infer an image property from an image.

There is no inclusion criterion for peer-reviewed journals or the minimum number of citations because we are looking at mostly recent technology. This means we cannot afford dismissing papers that might offer a fitting solution. A benefit of this specific field of research is the widely accepted use of open source code repositories. This makes it easier for other researchers to implement said technology and validate the claimed performance. See Table 2.1 for the literature list.

2.2 Musical keyboard

This section concerns the commonalities and differences for various musical keyboards that are of interest for this project.

Musical instruments come in all shapes and sizes. Two common types are the string piano and digital keyboard. For this project we will focus on digital MIDI musical keyboards that feature the typical repeated twelve keys (seven white keys, five black keys) per octave for playing the Western musical scale. Typical keyboards have a 159mm of octave span and typical pianos have a 164 mm

Name	Citation	Open source	Consult	PapersWithCode	Google Scholar	Peer reviewed	Year	# Citations
PianoAR	Huang, Zhou, Yu, et al. [3]				✓	✓	2011	14
YOLOv3	Redmon and Farhadi [4]	✓	✓				2018	734
MaskRCNN	He, Gkioxari, Dollár, et al. [5]	✓	✓			✓	2017	661
DeepIM	Li, Wang, Ji, et al. [6]	✓		✓		✓	2018	37
Keypoints	Zhao, Peng, Wang, et al. [7]	✓		✓			2018	3
SSD	Tekin, Sinha, and Fua [8]	✓		✓		✓	2017	89
DBLP	Konishi, Hattori, and Hashimoto [9]			✓			2018	1
DenseFusion	Wang, Xu, Zhu, et al. [10]	✓		✓		✓	2019	21

Table 2.1: The literature list as a result of the search strategy. The table shows if the paper contains a reference to an open source implementation, how the paper was found, if it is published to a peer-reviewed journal, what year it was first published and the number of citations it has at the time of writing (2019-10-23). Consult, PapersWithCode and Google Scholar refer to the way the paper has been found.

Figure 2.1: An overview of a typical musical keyboard. An octave repeats horizontally every seven white keys

of octave span, most musical keyboards have either 61, 76, or 88 keys in total[11]. Figure 2.1 shows a part of a common musical keyboard.

Different layouts and colorings for keyboards exist; such as small synthesizers and inverted black and white keys for some harpsichords, they do however fall outside the scope of this project. Some musical keyboards have either a matte or glossy finish. A glossy finish will act as a mirror for light sources and is a known challenge in computer vision. The detection of the musical keyboard will focus on the spacing and coloring of the common modern digital MIDI keyboard with 61 keys and 159mm of octave span.

2.3 Piano teaching

Learning to play the piano is a challenging but rewarding endeavor. The path to success is to combine quality teaching with practice and training. The beginner phase is a specifically important moment to focus on the basics, because they will be the foundation that we build on. Piano teachers plan for you to develop a wide range of musical skills; rhythm, musicality, theory, history, instruments, score reading, improvisation, posture, and of course playing pieces.

This section is written based on conversations with an experienced piano teacher.

- **Sheet music.** Common errors for beginner piano players are score sheet reading mistakes. Mixed reality tries to solve this problem directly. Sheet music stems from the time where paper and pencil were very practical tools to do music notation. It is very effective for writing down songs and can be read and played real-time on a variety of instruments once the readers has trained this skill. However, this skill requires a lot of practice.

  Technology enables us to display desired key-presses superimposed on a digital or mixed representation and is able to respond to where we are in the song. This sense of time locality and using the powerful human visual system by superimposing the desired key-presses on

the instrument itself enables even beginners to play a variety of songs. Popular examples of this technique are Guitar Smith and Synthesia. Guitar Smith is a guitar playing game that shows floating strings and frets with approaching colored blocks that indicate what strings to strum and what frets to press. Synthesia is a keyboard playing application that shows a virtual keyboard with blocks approaching the corresponding keys to press. Again the collision indicates a key-press. An example of piano teaching via the HoloLens with holograms is shown in figure 2.24.

- **Rhythmic feeling** is trained by letting the player count the rhythm for the current song. This will internalize the process and create a good foundation for when there is no metronome or other rhythmic assistance available. The application can use repeated ascending auditory and visual cues to train the user in rhythmic feeling.
- **Posture** refers to correct joint angles while playing. A straight back, wrists above and in front of the keys and using only the fingers for depressing the keys. Feedback on posture can be provided by a piano teacher or by a more advanced digital setup that takes the user' pose into account.
- **Chords** appear in a lot of songs in many ways and are the basis of the piece' tonality. Learning about chords improve the understanding how songs are built up and create useful handles for improvisation. Most sheet music displays the chords used in the song. Digital applications can show the corresponding chord notation for broken and full chords to increase the players' awareness.
- **High difficulty.** Every new song is a new challenge and chances for success are better if you start out slow with a single hand. Often the right hand is practised first, then the left hand, and ultimately both. The first step is to get all the notes for one hand with the correct fingers in the correct order. This prepares you how to move your hand while playing the piece. After this the piece can be tried with half speed and you will get feedback on the timing of the key-presses. At full speed you will notice what quick parts you still have to practice on. Repeat this for the left hand and finally for both hands. Parts with many mistakes can be detected and focused on for repeated practice. A digital setup can do this approach very efficiently and is able to detect the parts you are struggling on for repeated practice.
- **Demonstration and evaluation** are valuable tools for getting the feeling for a music piece. Before it is practised a demonstration is done to prime the user for the end-goal. The evaluation will be done after practising a single run, and digitally can even be done visually on a per note basis while playing. Show the user the overall performance; what parts are going well and what needs more attention (timing/velocity). Suggest slower practice for difficult parts.
- **Independence** is how well the user performs without a teacher, a piano teaching application or even sheet music. Dependent on the goals of the player the teaching aids can be removed progressively to help the transition to independent playing.
- Some **final thoughts** are; when you are working towards a performance, it is good to know that muscle memory can abandon you when you are in a different situation physically or mentally. One way to solve this is to always practise in a performance setting, but this is not always realistic. Another way is to learn to read sheet music or use a digital and visual notation system and rely on that ability while performing.

  The joy of playing musical instruments comes from intrinsic motivation and autonomy. This autonomy enables you to improvise and play musical pieces with your own twist as well as react dynamically to the crowd while playing. So do not focus too much on perfect accuracy and throw in some creativity and fun!

2.4 HoloLens

This section describes the mixed reality headset that is being used for this project.

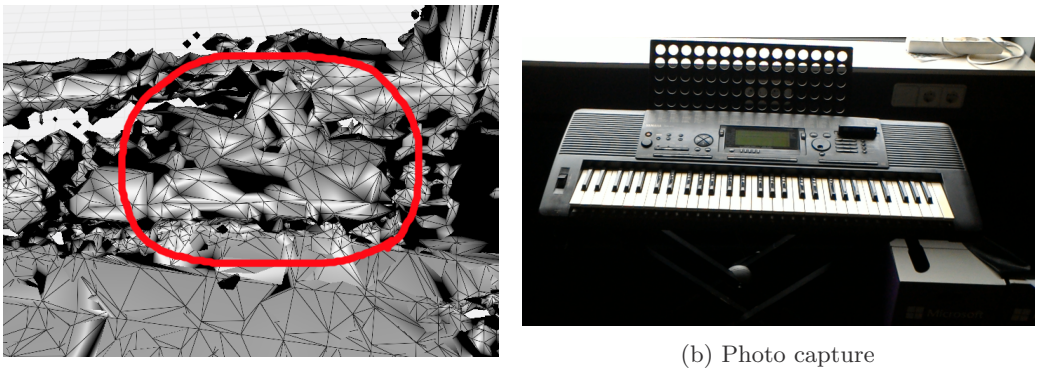
The HoloLens is a wireless head mounted device that is able to scan the 3-dimensional environment to create realistic augmented holograms on its see-through display glasses. It features real-time voice recognition, gesture recognition, audio playback and spatial positioning to create a complete mixed reality experience. See figure 2.2.

BetaBit owns the HoloLens 1 Developer Edition. It is on the market since March 30, 2016 and retailed for 3,000 US dollars. At the time of writing the HoloLens 2 is available for pre-order for 3,500 US dollars which will hit the market in 2019 Q3. Because the Hololens 2 was not available when the research started, this project will focus on the capabilities of the HoloLens 1.

Figure 2.2: The Microsoft HoloLens 1 Development Edition. (image courtesy Microsoft)

2.4.1 Spatial mapping

Building a spatial map is a mandatory task for creating an immersive mixed reality experience. The HoloLens uses two pairs of 30-FPS grayscale stereo cameras for building a 3D spatial map. It automatically generates a coarse 3D-model of the environment so applications are aware of the surroundings. This spatial mapping is built by walking and looking around, and gives polygon color feedback on parts of your surroundings that need more attention for generating a quality mapping if needed.



(a) Spatial mapping

(b) Photo capture

Figure 2.3: The spatial map of a room that contains a keyboard instrument. The instrument is encircled with a red color. Notice how the keys' fine features are not detected and the whole instrument is a coarse vertex blob. Also note that a spatial map is best viewed with stereoscopic vision.

The highest quality setting of the HoloLens 1 spatial mapping is too coarse to detect the fine features of the musical keys, see figure 2.3. A resembling shape of the musical instrument can be distinguished, but without prior knowledge it is hard to tell. We can perhaps use the 3D vertex data to detect if the musical instrument is a grand concert or digital piano, but we are specifically interested in an accurate pose of the musical keyboard. The HoloLens 1 spatial mapping is not a good feature candidate for accurate pose tracking.

2.4.2 HoloLens Spatial Locatability

A useful feature that the HoloLens automatically provides is tracking of the HoloLens pose (position and orientation in world coordinates). This will enable you to use world coordinates to place holograms, and it alleviates the need to track the object which many simpler AR devices are required to do so they do not lose the position in time with movement. This feature can be used for efficient automatic annotation of a dataset.

2.4.3 Sensors

The HoloLens features many imaging sensors. See figure 2.4 for the imaging sensors.

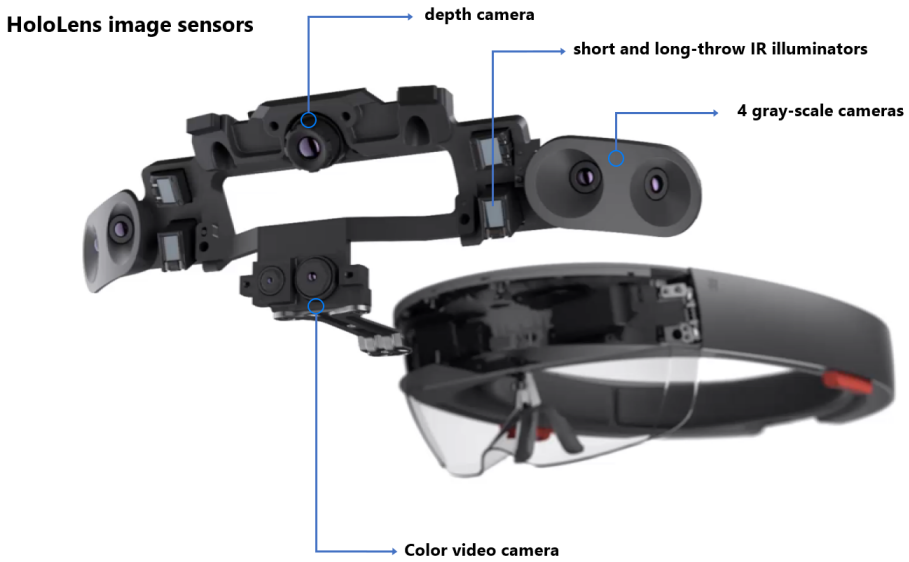


Figure 2.4: HoloLens imaging sensors. The HoloLens features an RGB/color video camera, depth camera along with short and long-throw IR illuminators and four gray-scale cameras. Image courtesy of Microsoft Marc Pollefeys [12].

The color sensor (RGB) is the standard camera at the front of the device. The other sensors are the depth sensor that supports near and far modes that can output depth and reflectivity, and the 2x2 stereo grayscale cameras pointing to the sides used specifically for inside-out tracking. Outputs of the streams can be found in figure 2.5.

2.4.4 HoloLens 2

The HoloLens 2 is announced to be released in 2019 Q3 and comes with new features e.g.; more than twice the visual field of view and higher resolution sensors with higher framerates that enable full hand-tracking. This might come in handy with detecting hand poses while playing a musical keyboard. Using the HoloLens 2 will improve the performance and accuracy of the final application because of the faster hardware and more accurate sensors. It may require retraining the musical keyboard models with the updated sensors for improved model performance. A notable addition is the Artificial Intelligence Deep Neural Network coprocessor. This means that Microsoft is investing in offline on-device Artificial Intelligence performance. Very promising for the future of HoloLens Computer Vision. With the real-time hand pose detection, feedback can be given on the used hand poses while playing. According to the piano teacher interview, this is an important skill to learn from day one. The increase in field of view will improve immersion and prevent missing notes on songs where notes are spread-out over the keyboard instrument.

2.4.5 Specifications

The hardware specifications of the HoloLens 1 and 2 are found in Appendix B.

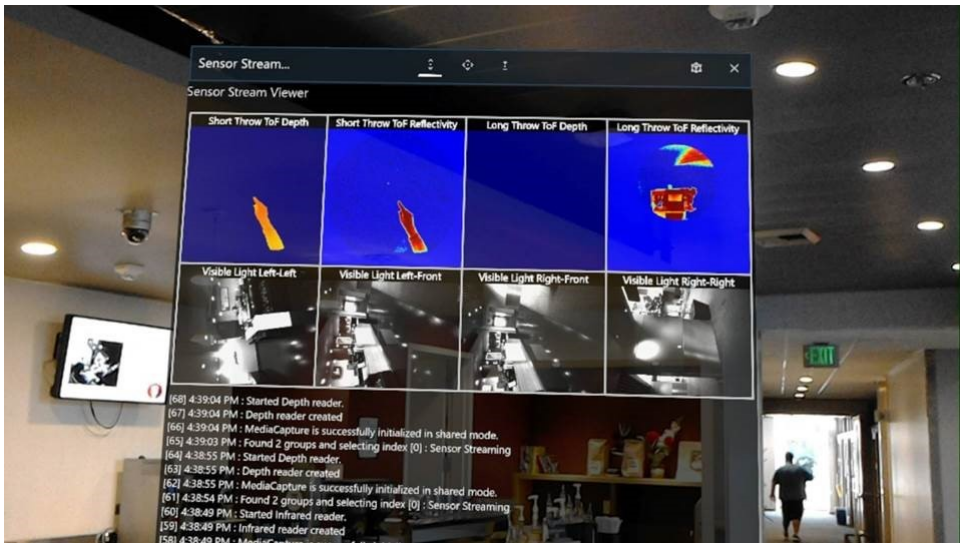


Figure 2.5: HoloLens image streams are shown augmented on the RGB/color video stream. Notice how the depth camera returns a depth image and a reflectivity image for far and near modes. From left-to-right and top-to-bottom the image streams are; near depth, near reflectivity, far depth, far reflectivity, left left grayscale, left front grayscale, right front grayscale and right right grayscale respectively. Image courtesy of Microsoft' Marc Pollefeys [12].

2.5 Computer vision

The following section explains the history and state-of-the-art related work in the Computer Vision research area.

The musical keyboard detection task is a Computer Vision problem. It requires RGB(-D) images as input and will output an image property as a result. Computer Vision solutions have changed through the years, the following subsections dive in the history and types of algorithms found in the different relevant areas.

Computer Vision has seen major improvements since its inception. These developments are supported by discovering new approaches and algorithms as well as improving hardware efficiency. The first approaches started with relatively simple tasks like digit recognition, very few image features and simple classifiers. The amount of features were reduced by calculating image properties with kernel convolutions (see Subsection 2.5.3 for convolutions) and other image operations to make sure the algorithms focus on the relevant input information. A typical image offers much data to process; $s_{datatype} * c_{count} * w * h$. Where $s_{datatype}$ is the size of the datatype per pixel, c_{count} is the amount of channels per pixel, w is the width of the image, and h is the height of the image. The size scales linearly with the image area; so for a $640 * 480$ RGB-byte image this is almost one MegaByte of uncompressed feature data. The pixel's position in the image can serve as valuable information as well, since pixels often relate to neighboring pixels.

2.5.1 PianoAR: Traditional methods for musical keyboard detection

Huang, Zhou, Yu, et al. [3] show an implementation for detecting a musical keyboard pose with an RGB-camera by relying on traditional image processing techniques. The implementation is not shared, but relies on thresholding, morphological operations, as well as an iterative end-point fit algorithm[13]. This algorithm is used to generate four corners of the threshold area in the image to create a plane. The plane should resemble the top surface of the white musical keys in 3D space and can be used for a Perspective-n-point algorithm[14] for extracting the actual musical keyboard pose. See figure 2.6 for an overview of the image processing steps.

Traditional image processing techniques require you to design an image processing pipeline which is specific for every object class. Managing theses pipelines with changing requirements is difficult and they are often susceptible to variations in input data. Figure 2.7 shows such a problem were changes in camera incident angle lower the accuracy to unusable values for this use case.

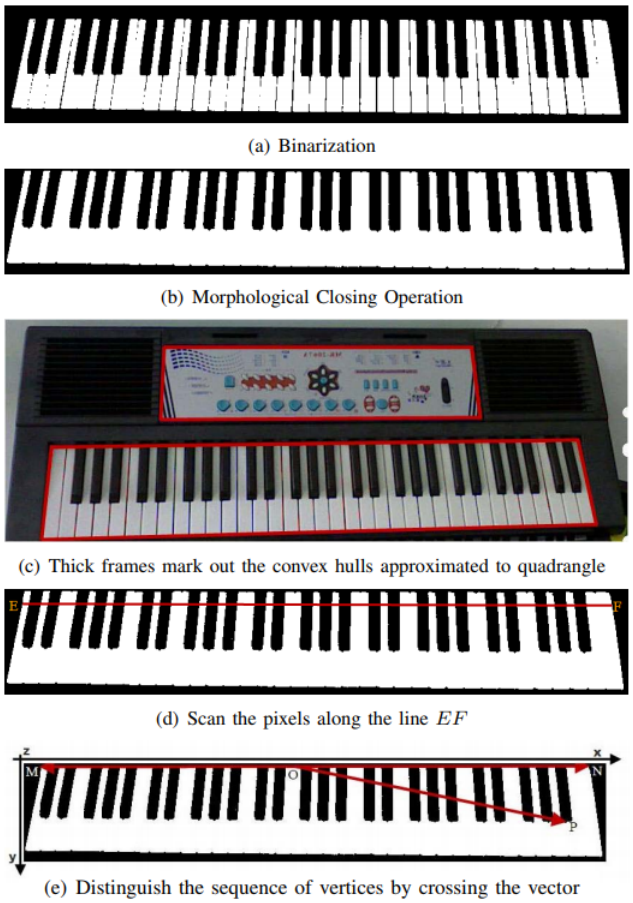


Figure 2.6: An overview of the process for detecting a musical keyboard with traditional image processing techniques. Taken from Huang, Zhou, Yu, et al. [3].

$\alpha(\circ)$	30	50	70	90
error(pixel)	21.72	8.89	4.97	1.97

Figure 2.7: The average pixel distance error for various camera incident angles. In this paper an α of 90° is a top-down view. The lower the incident angle, the harder it is to get accurate results. Taken from Huang, Zhou, Yu, et al. [3].

2.5.2 Designing quality features

While there are many classifiers that can handle big amounts of data for training, the challenge is to design enough quality features that separate the classes in a way that generalizes well for new data. A good example is the work by Viola and Jones [15]. They managed to design and implement a face detector that was comparable to state-of-the-art accuracy, but fifteen times faster in timing performance at the time. Many features were tested, sorted and selected on their ability to discern faces from non-faces. They ended up using 6061 features that were placed in a degenerative decision tree for extra timing performance. This shows the importance of selecting quality features for your model performance. Because computer vision is such a challenging field it required this many quality features to perform well.

2.5.3 Convolutional neural networks

Designing features for classifiers is one of the bigger challenges in computer vision. More processing power enables us to use artificial neural networks such as the convolutional neural network.

Biological neurons are the inspiration for the building blocks of neural networks. The biological neuron receives input signals through its dendrites resulting in an action potential. When a certain threshold is reached, the neuron fires and a signal is propagated through its axon and synapses that connect to other neurons' dendrites. A network of biological neurons is able to solve complex tasks. See Figure 2.8 for a visual representation of a biological neuron.

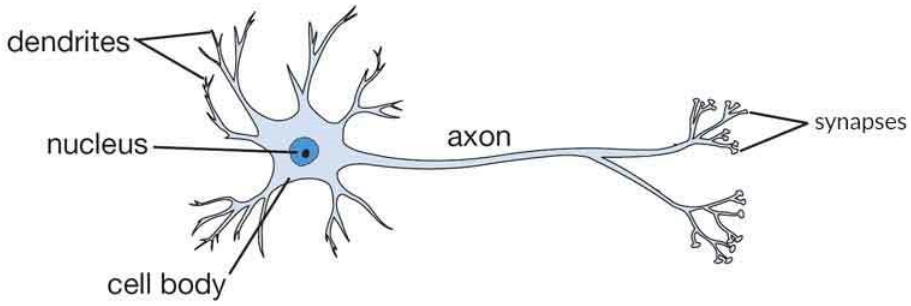


Figure 2.8: A graphical representation of a biological neuron is shown. The neuron fires when its action potential reaches a threshold. Taken from Andrej Karpathy [16].

The artificial neuron consists of an input vector $\vec{x}$ depicting the input values for this neuron. The bias b and weights $\vec{\omega}$ are stored in this neuron and can be adjusted with training. The neuron's inputs are weighted and then summed together with the bias. The activation function produces the output y based on the resulting sum. This process tries to simulate a biological neuron, but with floating point precision instead of binary signalling. See figure 2.9 for a visual representation of an artificial neuron.

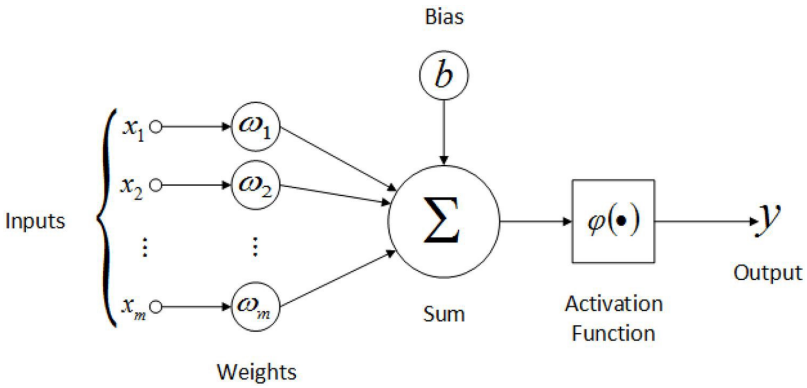


Figure 2.9: Artificial neurons are the building blocks of neural networks. It weights the inputs and sums this together with a bias. The output is transformed with an activation function and the final output is produced. See figure 2.10 for common activation functions. Taken from Jayesh Bapu Ahire [17].

The nonlinear activation function in the artificial neuron is required to approximate complex higher-order functions in networks. This simply cannot be done with only linear outputs. See figure 2.10 for a graphical representation of common activation functions.

The basis of Convolutional Neural Networks is also found in biology, where the visual cortex has a highly-layered structure of neurons. The Convolutional Neural Network is a special kind of layered Neural Network that connects neurons in a way that mimic traditional discrete 2D convolution operations. The convolutions are used to transform images and calculate image features like blur and edge-detect. To calculate such a feature, the kernel is setup to produce for example high outputs for big differences in horizontal intensity. The kernel slides over the image and calculates the sum for all weighted input values for every possible position in the image to obtain a new image. See figure 2.11 for a discrete 2D convolution operation.

The convolutional neurons have their inputs connected to some of the previous layer outputs just like a weighted 2D convolution kernel. This input group is called its receptive field. Neurons are able to capture complex features when the neuron's receptive field spans multiple layers. This is a key driver behind the translation invariance of convolutional neural networks. See figure 2.12 for visualized features for popular detection classes.

The challenge now is to optimally design the neural network for your specific needs. Determine the input and output sizes and design the hidden layers in such a way that the features can separate your classes in an effective way. It is difficult to get a good feeling for how these design choices influence the performance of the model. A visual representation of a deep convolutional neural network architecture is shown in figure 2.13

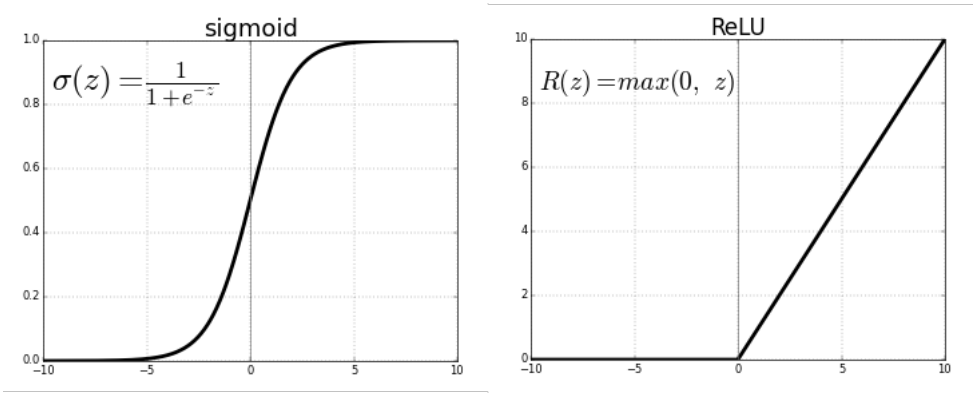


Figure 2.10: A plot of the logistic sigmoid and rectifier linear unit (ReLU) are shown. These common examples of activation functions are used to transform the neuron output so complex higher-order functions can be approximated with neural networks. Taken from Sagar Sharma [18].

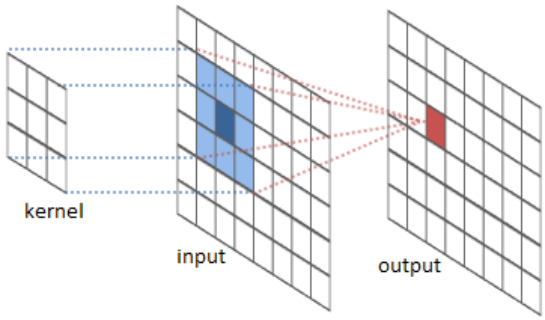


Figure 2.11: A discrete 2D convolution calculates the sum of the weighted inputs for every pixel. The input is weighted by a static 2D kernel. For a convolutional neuron the input is the receptive field, the kernel contains the weights associated with the input values for this neuron and it produces an output transformed with an activation function. See figure 2.9 for a description of an artificial neuron. Taken from Jaswanth Sreeram, Stephan Herhut, Lindsey Kuper [19].

All the neurons contain a bias and have a weight for every such connection. These features can be optimized per neuron with supervised learning through back-propagation. Back-propagation is the process of training a neural network by setting the desired output values and improving the features with gradient descend based on the obtained error on a per layer basis. This is possible, because of the differentiable properties of the network. Back-propagation does not necessarily find the global minimum error for every feature, but it has been shown that decent local minima perform roughly the same for Convolutional Neural Networks. Hyper-parameters like dropout, epochs, batch size, learning rate, momentum, weight initialization and activation functions all can be tuned to get the best performance and prevent effects like over-fitting and oscillations. A network is trained on many labeled images that is called a dataset. Creating a high quality dataset is one of the challenges for supervised learning approaches.

The outcome of a Convolutional Neural Network is a complex function approximator that works especially well on image-data. The output can be designed to be any number of neuron outputs. These values are your desired regression answers or class probabilities after a softmax layer.

2.5.4 Image recognition

The most basic image comprehension task is to recognize a class in an image. "Does this image contain a cat or a dog?" or "Which digit is written here?" The input is an image and the output is a number of probabilities for the possible classes. Popular datasets for this task are MNIST for small image grayscale digit recognition, CIFAR-10 for small image recognition with ten classes, CIFAR-100 for small image recognition with ten subclasses for every ten classes and ImageNet for all image sizes for 1000 classes. Performance is often expressed in the metric top-n error percentage. This is the percentage of predictions where the correct class is not in the top-n prediction list. This metric challenge naturally gets harder for a smaller n. Common metrics are top-1 and top-5 prediction

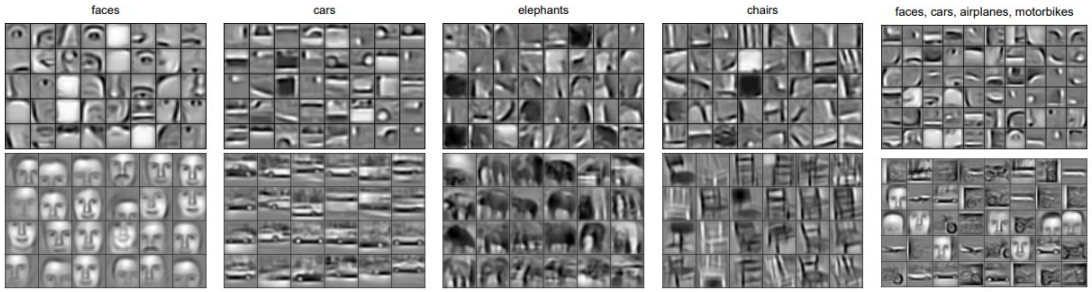


Figure 2.12: Features are visualized at different layers for a convolutional neural network trained on popular classes. The first layers can detect simple shapes like edges (not shown), complex features can detect simple shapes like eyes, noses, or a mouth, and even more complex features can detect full faces. Taken from Lee, Grosse, Ranganath, et al. [20].

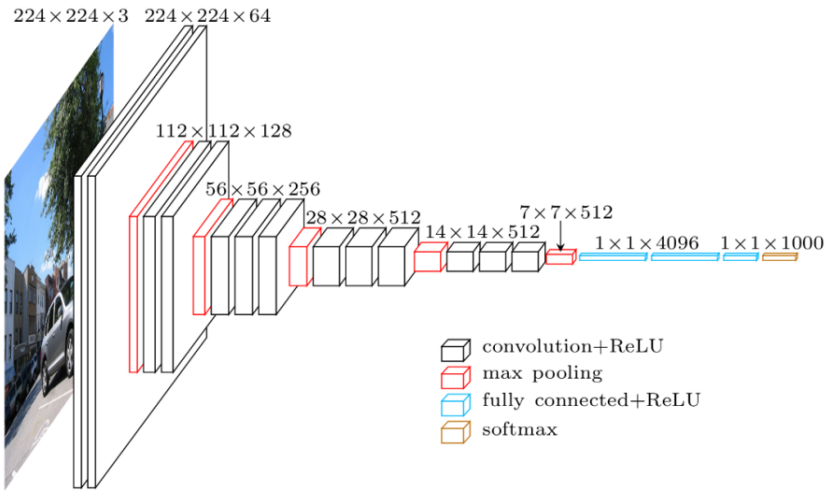


Figure 2.13: A deep convolutional neural network architecture is visualized. At the left an image of a specified size is used as input; $224 * 224 * 3$ datapoints in total. The blocks visualize the layers of the network with the dimensions shown at the top. In the convolution stages or feature learning stages the width and height is reduced with max pooling and the depth increased, so the features become more complex and translation invariant. Finally fully connected layers are used for classification of the 1000 classes this network can output. A softmax function can output probabilities for multiple classes. Taken from Simonyan and Zisserman [21].

error percentage.

2.5.5 YOLOv3: Object detection

With object detection we want to know how many and what classes were recognized and determine their location in the image. Detect zero or more bounding boxes in image space (x_1, y_1, y_2, y_2) that contain a class from a predefined set. Popular datasets for this problem are PASCAL VOC and ImageNet. Performance is often expressed in the metric mean Average Precision (mAP). It is the Area Under the Curve (AUC) calculated by the surface area of the precision-recall plot. Many flavors of calculating the surface area exist, but the common 101-point method samples all the precision values for all the recall values from 0.00 to 1.00 with a stepping size of 0.01. The metric is influenced by setting an Intersection over Union (IoU) or overlap threshold value. This threshold is commonly set at 0.5 or 0.75. To get a nice measure for multiple overlap values, they average the mAP for overlap thresholds from 0.5 to 0.95 with a stepping of 0.05. See figure 2.14 for examples of object detection.

2.5.6 MaskRCNN: Image segmentation

Image segmentation gives every pixel a class from the predefined set. This results in pixel precise annotations of objects in the image. A popular segmentation dataset is the MS COCO dataset. The metric used for this is Panoptic Quality, which is calculated from Recognition Quality and

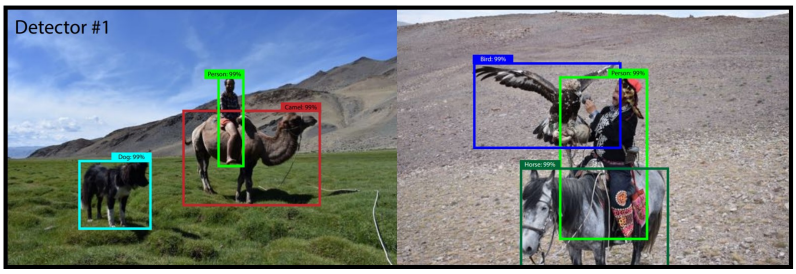


Figure 2.14: Examples of object detection with the YOLOv3 detector. Taken from Redmon and Farhadi [4].

Segmentation Quality per class. Details about COCO and the specific evaluation metrics are found at COCO Consortium [22]. See figure 2.15 for examples of image segmentation.

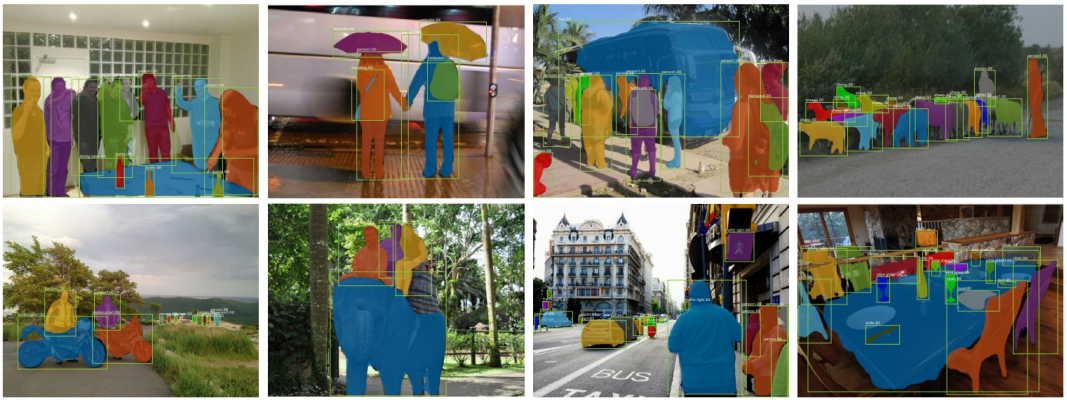


Figure 2.15: Examples of object segmentation with the Mask R-CNN detector. Taken from He, Gkioxari, Dollár, et al. [5].

2.5.7 6D Pose estimation

6D Pose estimation is detecting the position and orientation of a static 3D model in image space. This is a common problem in robotics where a robot arm needs to pick up 3D objects where the pose is not known in advance. The robots often have a camera or even a depth camera available for solving the problem of pose estimation.

This problem is solved by first detecting the region of an object class in the image, then keypoints or 3D bounding box image space points are used to train the neural network for a position and pose. The 3D bounding box corners and center are often used as regression output of the Neural Network to determine the pose. Sometimes iterative methods are implemented where they render the 3D model to compare and correct for small position and orientation errors. Iterative neural network methods are often slower, because it needs to rerun the correction network per iteration. Some state-of-the-art variants will be discussed in the following subsections.

Popular datasets for this problem are LineMOD RGB(-D) with the ADD metric, occlusion RGB with the mAP metric, and YCB-Video RGB(-D) with the ADD-S metric. The ADD and ADD-s metrics are the average distance for all the transformed model points compared between the ground truth and estimation. The ADD-S metric corrects for models that express a mirror symmetry or even an α-rotational symmetry for cylindrical objects. It chooses the lowest error distance of the symmetry possibilities. The metric is introduced by Hinterstoißer, Lepetit, Ilic, et al. [23]. See figure 2.16 for examples of pose estimations.

2.5.8 2D or not 2D (using the depth camera)

Some 6D Pose Estimation algorithms use a depth channel or a depth camera in the image input. The extra channel adds valuable information for solving the pose problem. It is difficult to incorporate the depth channel in computer vision architectures but it delivers improved accuracy as demonstrated by Wang, Xu, Zhu, et al. [10] in figure 2.17. The Microsoft HoloLens headset

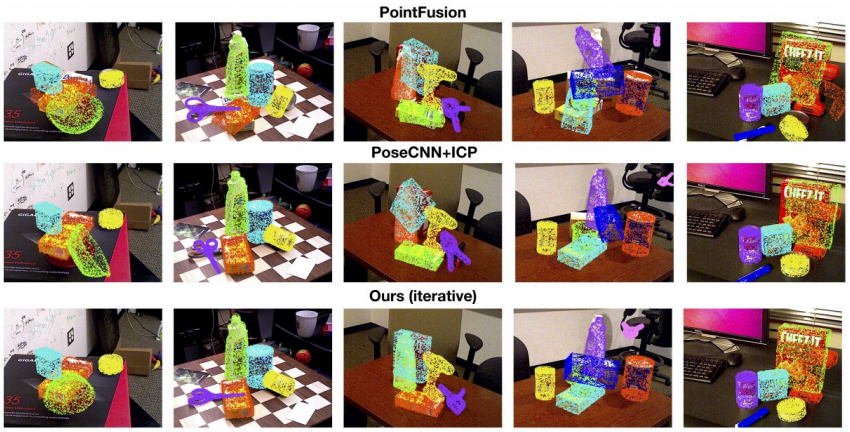


Figure 2.16: Examples of 6D pose estimations are shown here. Taken from Wang, Xu, Zhu, et al. [10].

offers the use of a depth camera and it is currently unclear how this effects accuracy and timing performance. This will be further explored in the research of this work.

	RGB		RGB-D				
	BB8 [24] w ref.	PoseCNN +DeepIM [17, 40]	Implicit [30]+ICP	SSD-6D [14]+ICP	PointFusion [41]	Ours (per-pixel)	Ours (iterative)
ape	40.4	77.0	20.6	65	70.4	79.5	92.3
bench vi.	91.8	97.5	64.3	80	80.7	84.2	93.2
camera	55.7	93.5	63.2	78	60.8	76.5	94.4
can	64.1	96.5	76.1	86	61.1	86.6	93.1
cat	62.6	82.1	72.0	70	79.1	88.8	96.5
driller	74.4	95.0	41.6	73	47.3	77.7	87.0
duck	44.3	77.7	32.4	66	63.0	76.3	92.3
eggbox	57.8	97.1	98.6	100	99.9	99.9	99.8
glue	41.2	99.4	96.4	100	99.3	99.4	100.0
hole p.	67.2	52.8	49.9	49	71.8	79.0	92.1
iron	84.7	98.3	63.1	78	83.2	92.1	97.0
lamp	76.5	97.5	91.7	73	62.3	92.3	95.3
phone	54.0	87.7	71.0	79	78.8	88.0	92.8
MEAN	62.7	88.6	64.7	79	73.7	86.2	94.3

Figure 2.17: ADD metric comparison on the LineMOD dataset for different RGB and RGB-D approaches. Notice how the DenseFusion "Ours (iterative) RGB-D approach" achieves higher accuracy than the state-of-the-art RGB approach. Taken from Wang, Xu, Zhu, et al. [10].

2.5.9 State-of-the-art RGB detectors

DeepIM

Li, Wang, Ji, et al. [6] show an implementation that currently is the state-of-the-art for the LineMOD dataset out of five entries. This means that the detector is able to achieve the best performance on a well-known dataset used for benchmarking. It is an iterative refinement technique that corrects the pose estimate by comparing the outcome with a rendered version of the 3D model and feeding that to a Deep Neural Network. In the paper they mention it runs at twelve fps on a 1080 Ti GPU with two iterations during testing. The LineMOD 2D projection accuracy of 97.5% reached is remarkable, but comes at the price of timing performance. See figure 2.18 for the architecture.

KeyPoints

Zhao, Peng, Wang, et al. [7] show an implementation that uses YOLOv3 for object detection and use Designated Surface Keypoints detection for generating the 3D bounding box. The keypoints

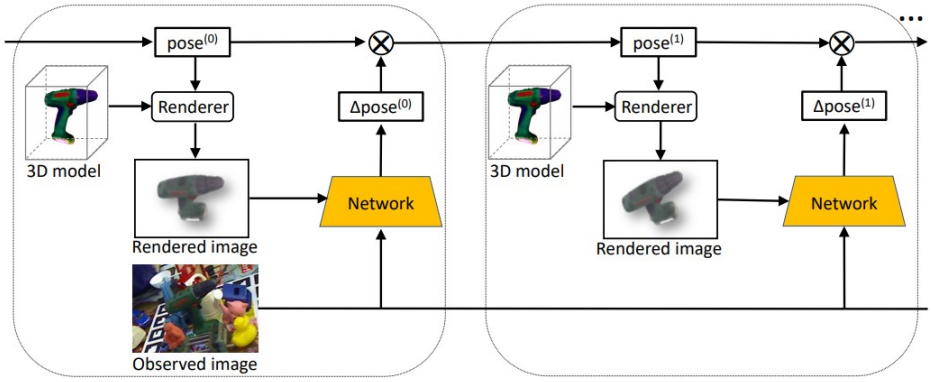


Figure 2.18: Taken from Li, Wang, Ji, et al. [6]. We propose DeepIM, a deep iterative matching network for 6D object pose estimation. The network is trained to predict a relative SE(3) transformation that can be applied to an initial pose estimation for iterative pose refinement. SE(3) stands for the Special Euclidean group in 3D, this describes both rotation and translation.

rely on the object having a texture with defined features for high performance. They achieve an accuracy of 94.5% on the LineMOD dataset which is just below state-of-the-art, but do so with at least half the performance of the real-time Single Shot Detector (25 fps versus 45 fps) without refinement. See figure 2.19 for the architecture.

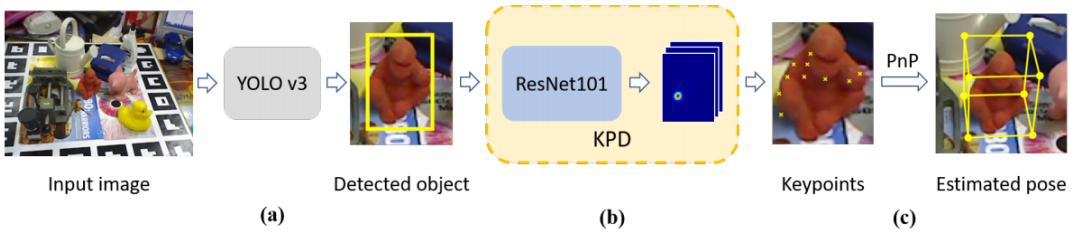


Figure 2.19: Taken from Zhao, Peng, Wang, et al. [7]. We first (a) detect bounding box then (b) localize the designated keypoints using keypoint detector (KPD). Finally (c) we use a PnP algorithm to recover the 6D pose.

SSD

Tekin, Sinha, and Fua [8] show a real-time implementation of a Seamless Single Shot 6D Object Pose Detector. The single shot architecture enables multiple object estimates to be made in a single picture for a single iteration. The implementation is based on PyTorch and achieves 50 fps on a Titan X (Pascal) without refinement and currently is the state-of-the-art out of two submissions for the OCCLUSION dataset with an mAP of 0.48 and used to be state-of-the-art for the LineMOD dataset with an accuracy of 90.37% according to PapersWithCode. See figure 2.20 for the architecture.

2.5.10 State-of-the-art RGB-D detectors

DBLP

Konishi, Hattori, and Hashimoto [9] show very promising results regarding 6D pose estimation accuracy and timing performance. However, they do not reference a code implementation. The authors were contacted and replied the code will not be available because of the connection to a commercial product. For this reason it will be omitted from the study and experiment.

DenseFusion

Wang, Xu, Zhu, et al. [10] propose a new architecture to improve the accuracy of RGB-D methods, specifically being resistant to occlusion in the input image. The implementation is based on PyTorch and they manage to achieve state-of-the-art performance with a 93.1% Mean AUC on the YCB-Video dataset out of three submissions. See figure 2.21 for the architecture.

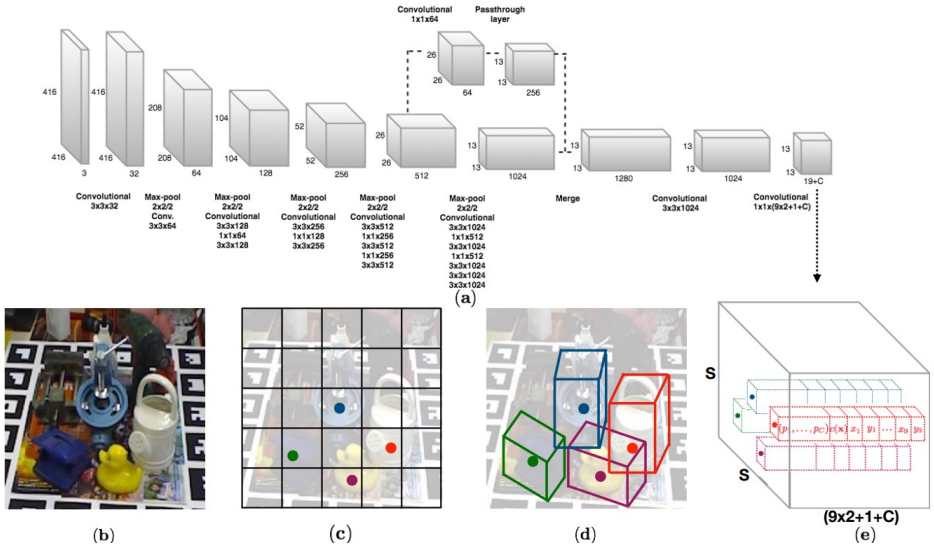


Figure 2.20: Taken from Tekin, Sinha, and Fua [8]. Overview: (a) The proposed CNN architecture. (b) An example input image with four objects. (c) The S x S grid showing cells responsible for detecting the four objects. (d) Each cell predicts 2D locations of the corners of the projected 3D bounding boxes in the image. (e) The 3D output tensor from our network, which represents for each cell a vector consisting of the 2D corner locations, the class probabilities and a confidence value associated with the prediction.

2.5.11 Cloud-based detectors

AzureML is a Microsoft cloud-based machine learning studio that can be used for Computer Vision purposes. After experimentation and problem solving it is clear that it is not possible to train an AzureML Neural Network model and export it from the cloud to use it on a device locally. This forces the application to always be connected to the internet for any Computer Vision detection uses.

However, there is a different portal available called Custom Vision [24]. This is also a Microsoft cloud-based machine learning studio, but specifically for Computer Vision tasks. It does support export of compact / lightweight models for on-device detection and it can be setup to detect an image class or bounding boxes. There is no implementation for image segmentation nor 6D pose estimation. The underlying algorithms of the various Custom Vision settings have changed since its inception and it is unclear what algorithms are used exactly. It is a quick way to create a powerful model, were it not that you have to manually annotate images in the browser. However, earlier model iterations can be used to get labeling suggestions.

2.5.12 Vuforia image target

Earlier augmented reality solutions use a technique called image targets (or tags/markers). The image target provides an accessible way to detect the world space from a single image in view. It comes at the cost of permanently requiring an image target within camera view. Vuforia is the go-to augmented reality library option for many developers because of its ease of use and support in popular game engines. Natural features can be extracted from images from a video stream or a potential image target that indicate how well a certain image can be tracked. Figure 2.22 shows the natural features in a quality marker image. The quality of an image target is determined by local contrast, detail richness and lack of repetition.

2.5.13 Vuforia model target

Vuforia also offers recognition of complex objects via its object targets, there is support for detecting self-scanned objects like toys and model cars and for larger objects there is a tool that can detect 3D models with so-called model targets. A musical keyboard falls in the category of bigger object and thus model target. A 3D model of a compatible piano has been used to setup a test and see if it manages to detect and track the piano accurately. Figure 2.23 shows that it is able to detect the musical keyboard from the camera, but the position and orientation are unusable for a decent

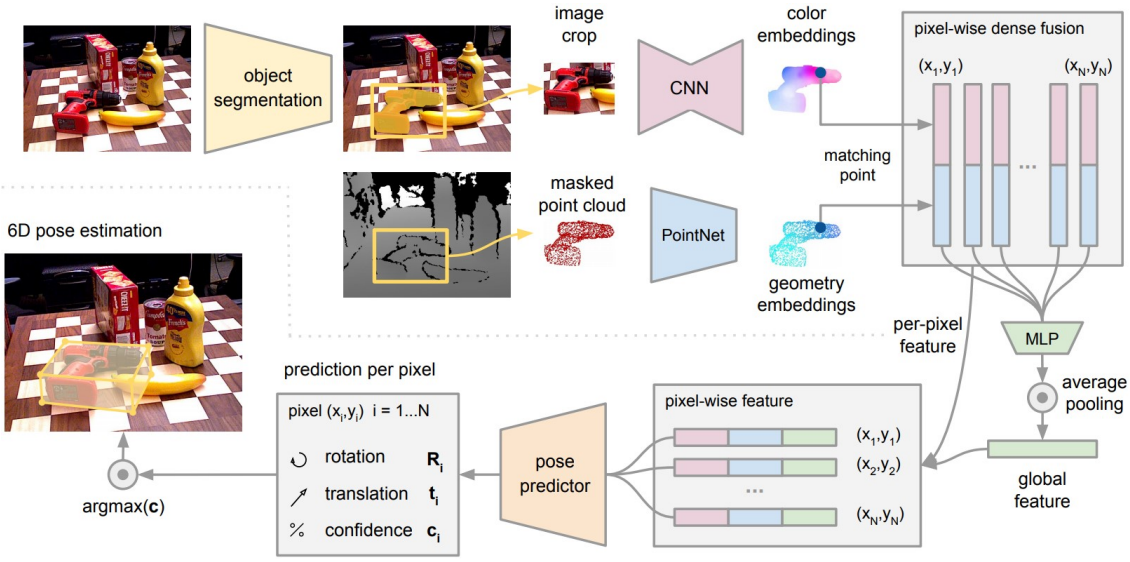


Figure 2.21: Overview of the DenseFusion 6D pose estimation model. The model generates object segmentation masks and bounding boxes from RGB images. The RGB colors and point cloud from the depth map are encoded into embeddings and fused at each corresponding pixel. The pose predictor produces a pose estimate for each pixel and the predictions are voted to generate the final 6D pose prediction of the object. (The iterative procedure of the approach is not depicted here for simplicity). Taken from Wang, Xu, Zhu, et al. [10]

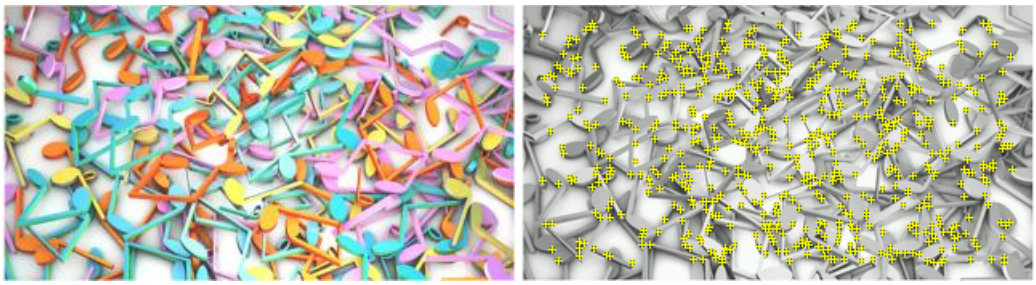


Figure 2.22: The natural features on an image marker. The amount of useful features in the image gets a high Vuforia augmentable score of five stars out of five, because of high local contract, detail richness and lack of repetition.

augmented musical keyboard experience. After detection the keyboard is tracked, but this results in even worse accuracy over time.

2.5.14 Music Everywhere tracking and user performance

Glickman, Lee, Hsiao, et al. [25] show a HoloLens keyboard instrument learning application that detects the musical keyboard via a printed Image Target. The drawback of this method is that you need to print the used Image Target and place it accurately on your musical keyboard before the virtual musical keyboard shows up in mixed reality. Luckily the HoloLens supports spatial positioning relating to world space and the Image Target can be removed after setting the correct virtual musical keyboard position.

For the user keyboard playing performance they use a Bluetooth MIDI module that can be connected to the musical instrument for real-time feedback of the pressed keyboard keys. See figure 2.24 for a mixed reality capture.

Figure 2.23: The output of Vuforia's Model Target pose detection from a guideview and subsequent tracking of the pose shows decent timing performance but poor accuracy for a musical keyboard use case. Notice how the tracked digital white keyboard overlay does not accurately match the musical keyboard in the camera feed.

Figure 2.24: A mixed reality capture from the Music Everywhere application. Figure taken from Glickman, Lee, Hsiao, et al. [25].

Chapter 3

Methodology

This chapter describes the methodology used in this work. The goal is to get a better understanding of the types of imaging sensors and how they perform at complex computer vision tasks. We will do this by selecting imaging sensors to compare, selecting a detector to compare them with and creating a dataset for these sensors in natural, artificial and dim lighting setups. With this dataset we will compare the sensors, gather precision and recall metrics, evaluate the results and answer the research questions found in Section 1.3.

3.1 Requirements

> **PS/Problem statement:** *To what extent are HoloLens imaging sensors able to solve computer vision tasks in natural, artificial and dim lighting setups?*

In this work we aim to tackle the problem statement by answering the research questions found in Section 1.3. This is necessary to get a better understanding on how advanced mixed reality headsets should be used when approaching computer vision problems.

> **RQ1/Research question 1:** *Which HoloLens imaging sensors are supported by state-of-the-art detectors?*

RQ1 requires an understanding of the HoloLens imaging sensors. This is found in technical documentation and is described in Chapter 2.4.3. It also requires an understanding of the state-of-the-art detectors, which is found in Chapter 2.5. Initial experimentation of recording with imaging sensors will show the common pitfalls of using such a sensor. Then a connection can be made between imaging sensors and detectors by finding the appropriate detector for each imaging sensor.

> **RQ2/Research question 2:** *Which state-of-the-art detector is suitable for comparing imaging sensors?*

RQ2 requires the results from RQ1, the background of the detectors in Chapter 2.5 and initial experimentation of implementing and using detectors to make an informed decision about what detector to use for comparison.

> **RQ3/Research question 3:** *To what extent do infrared imaging sensors improve detector performance in a dim lighting setup compared to an RGB imaging sensor?*

> **RQ4/Research question 4:** *To what extent do natural and artificial light influence infrared imaging sensor detector performance?*

RQ3 and RQ4 require a musical keyboard dataset built specifically with the selected HoloLens imaging sensors in natural, artificial and dim lighting setups. The used sensors and detector are found in the RQ1 and RQ2 answers. To gather the data, a recorder application is used and image pre-processing is done to make the data fit for the selected detector. See Chapter 4 on how the dataset is designed and created. The selected detector will be implemented to run the fixed quantitative laboratory experiment. See Chapter 5 for the experiment.

3.2 Ethical considerations

Ethical considerations include the internship being done at BetaBit, a Microsoft Gold Partner. This influences the scope of researched detectors and detector selection. However, it should not influence the generated performance metrics of the experiment. Every mention of a Microsoft service can be replaced by a service from a similar company that offers such a service. There is no reason to believe these alternative services yield different results.

3.3 Sampling strategy

Convenience sampling is used for creating the dataset, which means a convenient location and a convenient musical instrument are used for the experiment. Many configurations can be researched, but the premise is that this setup generalizes to most complex computer vision tasks. Random sampling is used for the images in the dataset. For every category 50 random samples are chosen from all of the setup samples to train the detector with.

3.4 Validity

Validity is how accurate the methodology is for answering research questions. The four ways of describing study validity are discussed in the following subsections.

3.4.1 Construct validity

We measure the dependent variable dataset performance that has the metric mean Average Precision (mAP). This is a widely used and accepted metric for predicting detector performance. Detector mAP is used as a measure how well a detector can predict bounding boxes and their correct classes in images. If the detector is the same but the imaging sensor is an independent variable, this metric correlates with dataset information density and the detector's ability to extract a correct class and position for the bounding box.

3.4.2 Face validity

The experiment measures detector performance under natural, artificial and dim light with an RGB, depth and reflectivity imaging sensor. The most prominent lighting setups have been identified by gut feeling. More lighting setups can be added, but these three make a good balance between effort and value. The musical keyboard is a convenience choice and the key width varies for some piano instruments. This means the trained detector models will not generalize to most pianos but this is not necessarily an issue for dataset comparison. More musical instruments can be added to make for a more diverse dataset but the added value is considered not worth the effort for this project's constraints. The same can be said for location, time of day, date and natural lighting. In the PS the musical keyboard is not mentioned, as the premise here is that a musical instrument will generalize to various augmentable objects.

3.4.3 Content validity

No content validity study has been done. This was deemed outside the scope of the study.

3.4.4 Criterion validity

No comparable studies have been found in the literature study. There are many studies that compare RGB and RGB(-D) detectors, but none that directly compare RGB, depth and reflectivity datasets for quality. This results in low criterion validity. From the related detector comparison papers the performance metrics are studied and the widely used and accepted mean Average Precision (mAP) is used in this study.

3.5 Reliability

Reliability is how precise the methodology is at generating results. The four types of reliability are discussed in the following subsections.

3.5.1 Test-retest reliability

Test-retest reliability is the reliability of a test at different times. One of the dataset setups included natural light and an RGB camera. Natural light of course varies throughout the day and year by overcast, sun position and intensity. There was no lumen meter available for measuring the natural light intensity during dataset creation, but it was a sunny day the 27th of September of 2019 in Utrecht. The natural light came from a big window behind the musical instrument. See Chapter 4 for dataset examples. Recording new samples for the natural light setup and RGB camera dataset might vary if recorded at a different time and place. Luckily RGB cameras have automatic exposure to compensate for different brightness levels. Different samples might influence detector performance, but as long as the musical keyboard is recognizable in pixel values the detector should be able to use it as training data.

3.5.2 Inter-rater reliability

Inter-rater reliability is the reliability of the same test conducted by different people. The dataset recording and pre-processing steps should be carefully executed to ensure the dataset quality. The same positions and timings should be assumed while recording to ensure a diverse and equal set of samples for every setup. Failure to do so might result in a bias for one of the setups. See Section 4.4 for a description of recording the dataset.

3.5.3 Parallel forms reliability

Parallel forms reliability is reliability of different versions of a test which are designed to be equivalent. For the experiment we could have chosen a different detector to do the lighting setup and imaging sensor comparison. There is no reason to believe different classifiers or object detectors will output very different results compared to this setup with the chosen classifier. A better detector that has been trained longer will achieve a higher precision with regards to detection performance. This means that the detector will be more effective at making comparisons with fewer samples. At the moment every lighting setup and imaging sensor has 50 samples selected which is the lower limit for avoiding overfitting problems with state-of-the-art neural network detectors [1]. A detector that is specifically designed for reflectivity or depth images will be able to achieve better performance, but is not considered a parallel form. Different locations, objects, times and dates can be used as parallel forms and should not drastically influence the results either.

3.5.4 Internal consistency reliability

Internal consistency reliability is how well individual items reflect the final measure. k-Fold cross validation is used to calculate the generalizable performance of the model. While calculating this, it also calculates a standard deviation of the performance. This gives an indication of the variability of the performance of different folds of the dataset. Effectively reporting the quality of the dataset size and consistency. This measure will be used to reflect on the internal consistency.

3.6 Development

The development of this project will be done in Visual Studio 2019 Enterprise. The project files will be source controlled in the Betabit DevOps in Git. See the citation of the author Thomas Aalbers [26] for the URL. Contact Betabit or the author of this work for possible access.

3.7 Time plan

The planning of the project spans 33 active weeks and is divided in three parts. The first part is the literature study, the second part is the implementation phase, the third part after the summer

holiday is the analysis, evaluation and also the final part. The project started the 1st of February 2019 and concludes on the 21st of November 2019. A visual time plan of the activities is found in Appendix A.

Chapter 4

Dataset

For the experiment a dataset will be created that is based on the Yamaha PSR-520 MIDI digital keyboard in natural, artificial and dim light. The dataset contains RGB, reflectivity and depth images saved from video recordings by the HoloLens imaging sensors. Common musical keyboard viewing angles and distances will be used as well as more uncommon ones to improve the robustness of the trained models.

4.1 Data

The dataset can be found at the dissemination website; http://HoloPiano.TomHash.NL/ [27]. The size of the stripped dataset is approximately 190 MiB.

4.2 Independent variables

In the dataset and experiment we have two independent variables; imaging sensor and lighting setup. See Table 4.1 for the selected imaging sensors and Table 4.2 for the selected lighting setups.

Code	Selected HoloLens imaging sensors
RGB	Photo video; RGB perspective camera. (1280*720@30Hz, RGB8)
Ref	Long-throw reflectivity; IR illuminated fisheye camera. (448*450@1Hz, G8 → RGB8)
Dep	Long-throw depth; depth-sensing fisheye camera. (448*450@1Hz, G16 → G8 → RGB8)

Table 4.1: An overview of the different HoloLens imaging sensors that we are going to compare by performance.

Code	Selected lighting setups
N	Natural light
A	Artificial light
D	Dim light

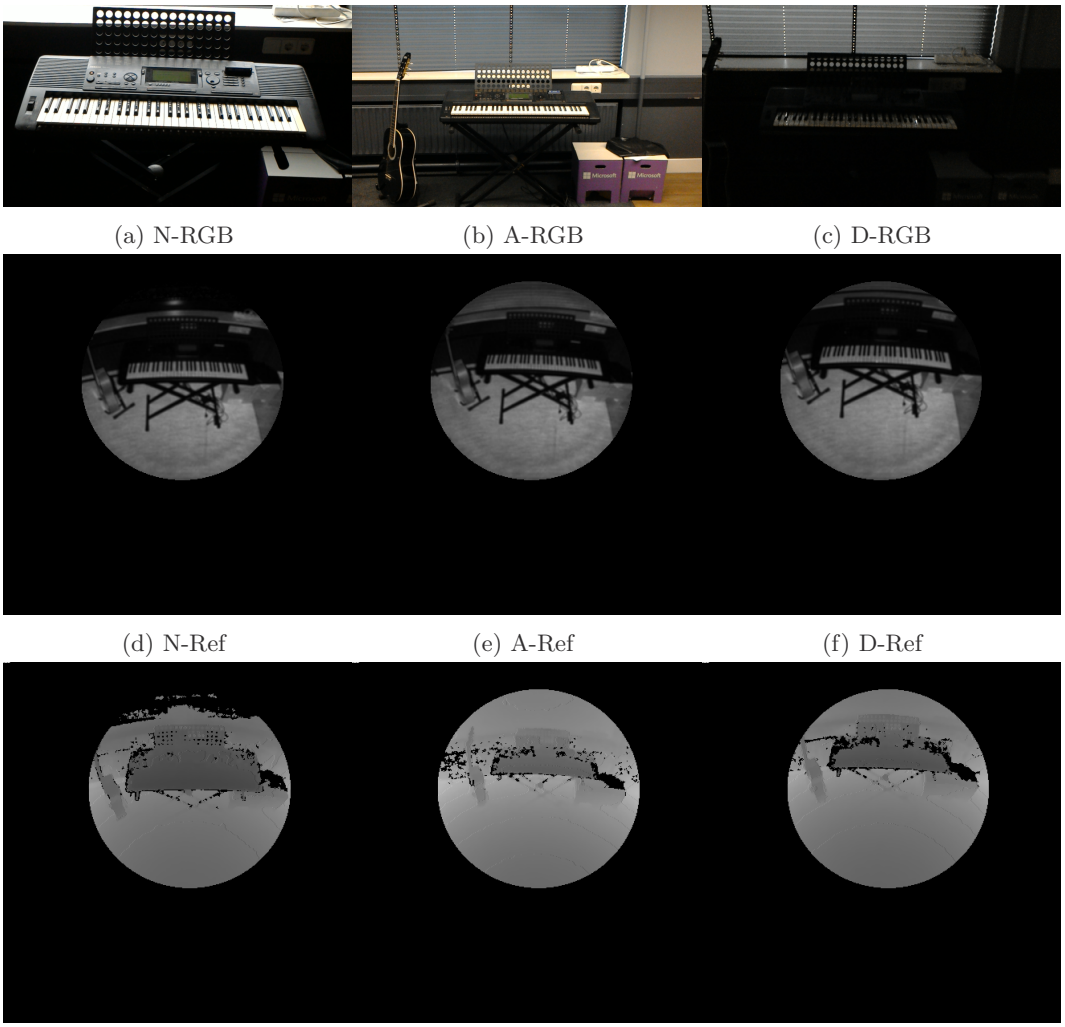
Table 4.2: An overview of the different lighting setups for the experiment.

Both variables have three discrete values that make for a total of nine test cases. The test cases are shown in Table 4.3.

Imaging sensor \Lighting setup	Natural	Artificial	Dim
RGB	N-RGB	A-RGB	D-RGB
Reflectivity	N-Ref	A-Ref	D-Ref
Depth	N-Dep	A-Dep	D-Dep

Table 4.3: An overview of the different test conditions for the experiments. The independent variables imaging sensor and lighting setup both have three discrete options resulting in nine total setups.

To get a better understanding how the independent variables influence the dataset, examples for every setup are shown in Figure 4.1.



(a) N-RGB (b) A-RGB (c) D-RGB

(d) N-Ref (e) A-Ref (f) D-Ref

(g) N-Dep (h) A-Dep (i) D-Dep

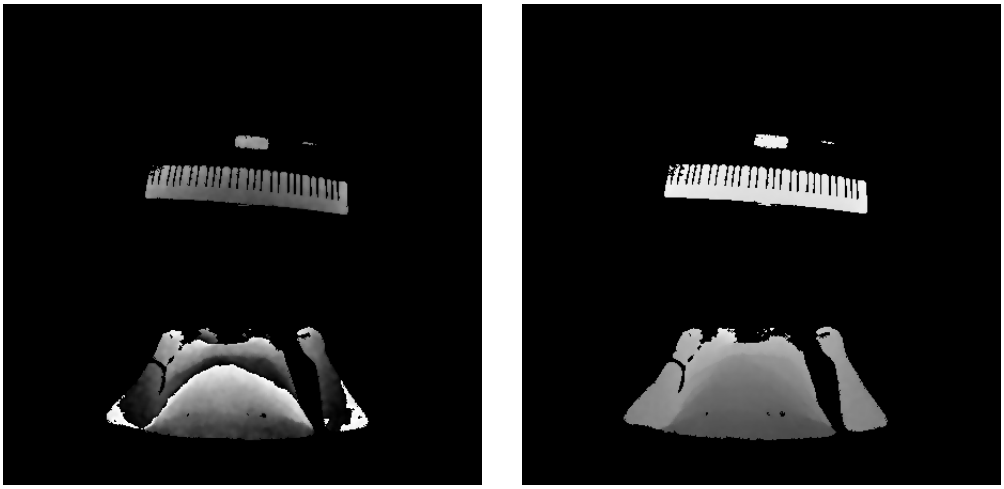
Figure 4.1: Examples of the dataset for every setup. Notice how reflectivity and depth streams seem unaffected by the lighting setup.

4.3 Pre-processing

The samples are recorded with the HoloLensForCV Recorder application [28]. The recorder application has been modified to exports the relevant imaging sensors; RGB, long-throw reflectivity and long-throw depth. The generated files can be moved from the HoloLens to a workstation with the HoloLens portal in de browser over Wi-Fi. The RGB images are Portable Pixel Map (PPM) files. The depth and reflectivity images are Portable Grayscale Map (PGM) files. These uncommon file formats are useful for exporting raw data, but to be useful for other applications we convert them to the more common Portable Network Graphics (PNG) file format. The PGM and PPM files can be opened with the program ImageJ. An important step is to import the depth maps as Raw data, 16-bit Unsigned and Little-endian byte order selected. Skip the first 17 bytes because they contain the header: P5\n448 450\n65535\n. After the images in the folder are imported, go to Image → Adjust → Brightness and Contrast, set the range from 0 to 4080. Now you can change the Image → Type → to 8-bit. After this, save the stack as PNG image sequence. The reflectivity images can automatically be opened, and saved as a PNG sequence.

The 8-bit PNG samples depth or reflectivity can directly be used with CustomVision.ai, because it populates all three RGB channels with one grayscale channel. This yields better performance than leaving the channels empty and is a fair comparison because the same neural network is used. The 16-bit depth samples are pre-processed with the ImageJ image editing tool. The 16-bit depth samples do not occupy the full 16 bits of data and the samples show a range of 0-4080. The range will be transformed to fit one channel of 8 bits, resulting in a loss of precision by a factor of 16.

16-Bit images with incorrect endianness can be identified by a banding effect. See figure 4.2 for an example.



(a) Incorrect endianness (b) Correct endianness

Figure 4.2: Depth images with incorrect (a) and correct (b) endianness.

4.4 Choreography

The same recording choreography is used for all lighting setups to prevent a bias. The goal was to record the musical instrument from a wide range of naturally occurring angles and distances. The recording starts from four meters back and one meter to the left relative to the musical instrument if viewed from the front. While recording, a slow pace is used to walk straight forward, when approximately two meters from the musical keyboard, the direction is changed to the right until one meter to the right relative to the musical instrument. From here a backwards direction is taken until four meters away. From here we walk to the center of the musical instrument, and then walk towards the instrument until two meters away. The recording is stopped here. During the recording the gaze is fixated on the middle of the musical instrument. The walking speed is adjusted so that the choreography takes at least one minute. The lowest imaging sensor framerate is 1Hz, this means that all imaging sensors provide a minimum of 50 samples with margin.

4.5 Quality

Microsoft offers a helpful page for creating quality datasets for CustomVision.ai[1]. To create a quality dataset, samples must be created that vary by:

- **Camera angle**

  A naturally occurring variety in camera angles has been provided by recording the musical instrument with a choreography spanning from left to right and front to back. Normal head tilt is used and top-down or back-facing views have been omitted.

- **Lighting**

  Lighting is an independent variable for this experiment, the dataset is divided between three lighting setups; natural, artificial and dim.

- **Background**

  One background has been used for convenience. Since the keyboard's direct surroundings are its own casing, the influence of a different background outside of the keyboard is expected to be of little influence. This is the case as long as the musical keyboard is somewhat axis-aligned in the image.

- **Visual style**

  Only a single visual style is used for the dataset, no different visual styles are recorded.

- **Individual/grouped subject(s)**

  Only a single musical instrument is recorded at a time.

- **Size**

  Only a single size of musical instrument is used for the dataset, no different sizes are recorded.

- **Type**

  Only a single type of musical instrument is used for the dataset, no different types are recorded.

The resulting dataset models will not be used for real-world applications and only for the experiment, this means that not all the properties need to be met. However, it gives a good indication on what points the dataset quality could be improved and the comparison of datasets could be improved.

The musical keyboard suffers from incorrect sampling of both matte and glossy black regions. See figure 4.3 for an example. This behavior might influence detection performance.

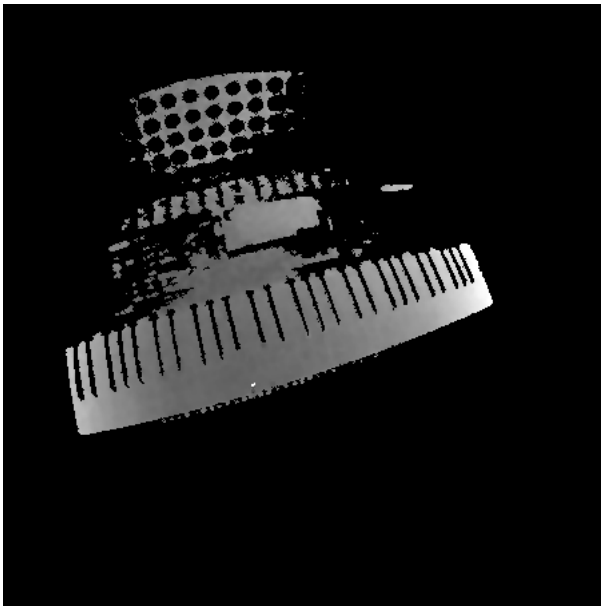


Figure 4.3: Depth image of a musical keyboard instrument. The glossy white keys respond very well to depth sampling, while the glossy black keys and the matte black housing only respond from a short distance.

When there is high contrast in the image, the RGB imaging sensor struggles with exposure. This results in lack of detail for bright and dark parts in the image, which might hurt detector performance. An imaging sensor with high dynamic range should solve this problem as the premise is that accurate detail richness is mandatory for high detector performance. See Figure 4.4 for an example of overexposure.

4.6 Ground truth

The labeling of the dataset is done in the web portal CustomVision.ai. A high-precision pointer device has been used to draw the bounding boxes to improve accuracy. The goal was to draw almost pixel-perfect bounding boxes, every bounding box that was off by more than three pixels on any edge has been redrawn. This process was checked by visual judgement. CustomVision.ai only supports exporting the trained model and not the manually labeled images, this means that the dataset will not include the manually labeled bounding boxes.

Figure 4.4: The RGB sensor struggles with exposure when facing high contrast by natural light.

A shortcoming of this dataset is that it does not use a segmentation of pixels that specifically include the subject but rather object detection which uses a bounding box. This includes surrounding pixels which will be used to train the model. Generalization might be hurt because a trained model will be sensitive to this specific surrounding. Segmentation would solve this issue, but was deemed impractical in the project constraints.

Chapter 5

Experiment

This chapter describes the experiment that is conducted in this study.

5.1 CustomVision.ai

In the experiment we use the cloud service CustomVision.ai for object detection. CustomVision.ai does not disclose which underlying neural networks are implemented for the domains, but looking at the topology they output a 13 by 13 tensor that hints at the use of a similar structure as Tekin, Sinha, and Fua [8]. The image is divided into 169 cells and for each cell a prediction is done. Bounding boxes are generated based on the predictions of these cells. See Figure 5.1 for the used CustomVision.ai compact domain topology.

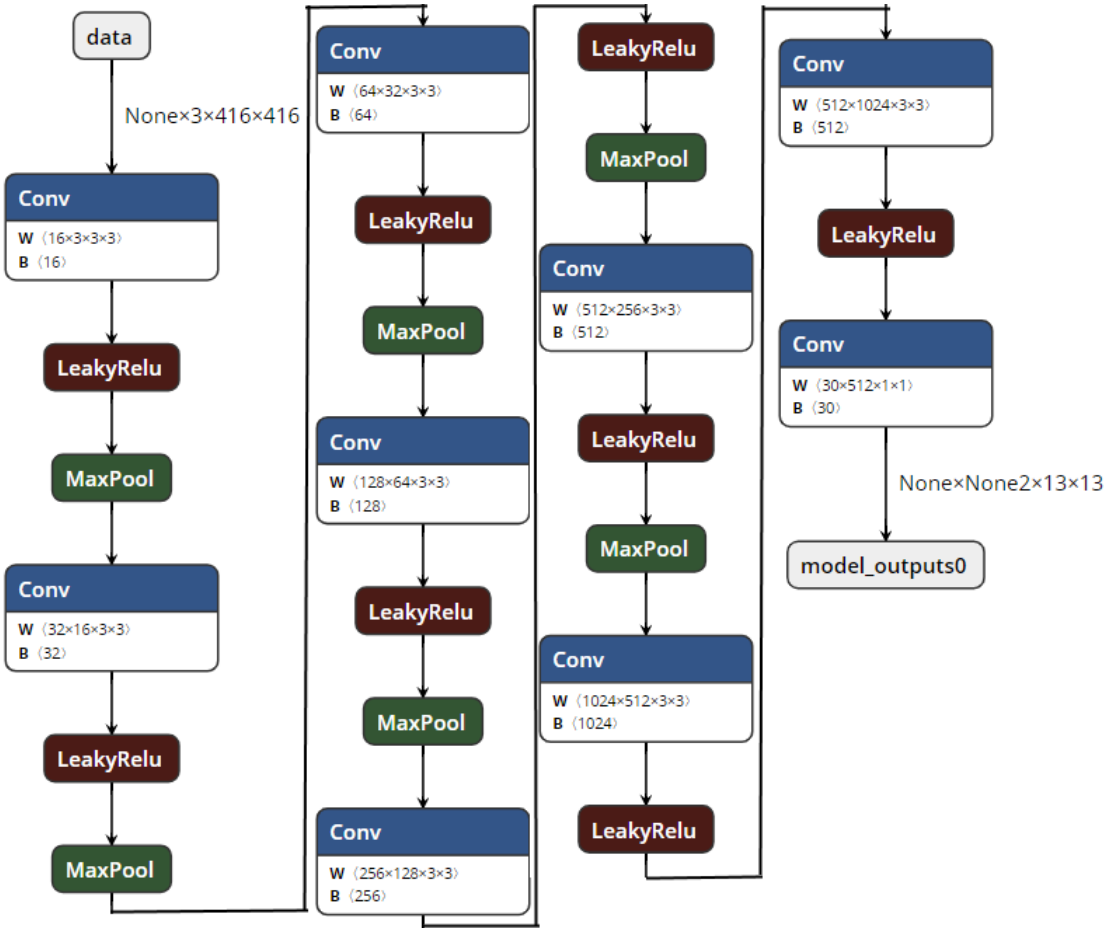


Figure 5.1: The convolutional neural network topology of the CustomVision.ai compact domain visualized in four columns for clarity. The topology connects from the bottom to the top of the next column.

Because we are planning on using this inference engine on device, the General (compact) domain is selected. This enables us to export the neural network model and delivers a faster and smaller model at a the cost of detection performance. We will export to the ONNX format, which can be used by the HoloLens via Windows Machine Learning (Windows ML). The General and Logo domains do not offer this export capability.

5.2 Training

At the start of training the network is given an initial set of weights that is known to be good at detecting all kinds of classes. The pre-weights shorten the total training time because local minimization of the error function will likely take less iterations. See Subsection 2.5.3 on how neural networks are trained with images. CustomVision.ai likely uses hyperparameter optimization for dropout, epochs, batch size, learning rate and momentum. Every setup has been trained with one hour training budget on CustomVision.ai.

5.3 Metrics

The independent variables in Section 4.2 influence the dependent variable; detector performance. Different metrics are gathered to describe detector performance.

The object detection task can be split in two: "Is the object in the image?", and "Where is the object in the image?". Both these questions will output a regression answer. A **confidence** prediction whether the object is in the image and a bounding box prediction where it was found. From this bounding box a metric can be calculated; **Intersection over Union (IoU)** or overlap. This measures how closely the location of the bounding box matches that of the ground truth. Because of the vast amount of samples the regression problem is transformed into a binary setup by choosing a threshold for confidence and overlap.

We use the trained detector as a binary classifier and generate counts for **true positive (TP)**, **false positive (FP)** and **false negative (FN)** predictions. See Figure 5.2 for the confusion matrix of binary classifiers.

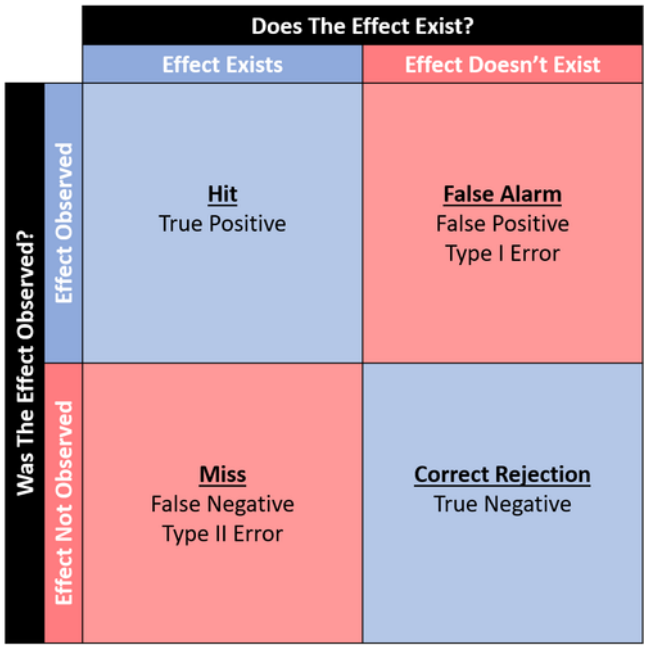


Figure 5.2: The confusion matrix for individual detector predictions of a binary classifier. Figure taken from [29].

Predictions are done by the detector that can either be positive or negative. If the prediction actually exists in the ground truth the prediction is considered a true positive. If the prediction does not correspond with the ground truth it is considered a false positive. If there is a label that the detector did not find it is considered a false negative. True negative (TN) counts are not used

because this and other computer vision datasets commonly do not have negative samples. From the TP, FP and FN metrics we can calculate the **precision** and **recall**.

$$precision = \frac{TP}{TP + FP}, recall = \frac{TP}{TP + FN}$$

A **precision recall curve (PR-curve)** can be visualized from these metrics for the possible confidence thresholds and for a specific overlap threshold. Precision and recall numbers are computed for all the confidence thresholds, these points are plotted and connected to form a precision recall curve. The precision is plotted on the vertical axis and recall on the horizontal axis. This curve shows how a classifier performs when balancing precision with recall and how precision and recall are typically inversely correlated. It is commonly used in addition to the receiver operator curve and is in comparison unaffected by "unbalanced" datasets, such as the dataset used here.

Average precision (AP) is a single value that can be calculated from the precision recall curve. The most common way is by calculating the area under the curve (AUC), this will give you one number that balances the precision and recall performance of your detector at a certain overlap threshold. This balance is made by calculating the surface area of the plot. Some detectors can detect multiple classes, for this the **mean Average Precision (mAP)** is calculated to still give a single performance result. The COCO dataset compares detectors by calculating this number for all the overlap thresholds from 0.5 to 0.95 with increments of 0.05 and averaging the outcome. If this metric is given for a certain overlap it is mentioned as mAP@0.5, with an overlap of 0.5 in this case.

F-scores can be used to determine a single performance value by balancing precision and recall, but these scores are generally not used in state-of-the-art object detector papers. The mAP is more accurate because it considers the precision recall relation globally.

In this experiment we will use the intersection over union, confidence, true positive, false positive, false negative, precision, recall and mAP metrics to be able to answer the research questions based on our needs.

5.4 Evaluation

We evaluate the performance of the nine different trained detectors and their setups found in Table 4.3. With the results of the experiment the research questions can be addressed.

The dataset will have multiple splits of the training set, validation set, and test set. Calculating the performance metric will be done with a k-fold cross-validation. The amount of test and validation samples are: $N_{test} = \frac{N_{total}}{k}$ and $N_{validation} = \frac{N_{total}}{k}$ for every fold of the cross-validation. Where k is the amount of folds. This works if $k \geq 3$. The specific k used in CustomVision.ai is not given, but typically $k = 5$ is used. Cross-validation is used to prevent overfitting of the model. This in term means that the model is in the best state to generalize to unseen data. See Figure 5.3 how overfitting is detected.

The difference in performance numbers of the experiment setups will be checked for significance with a two means two-tailed t-test with a known sample standard deviation using an alpha value of 5%.

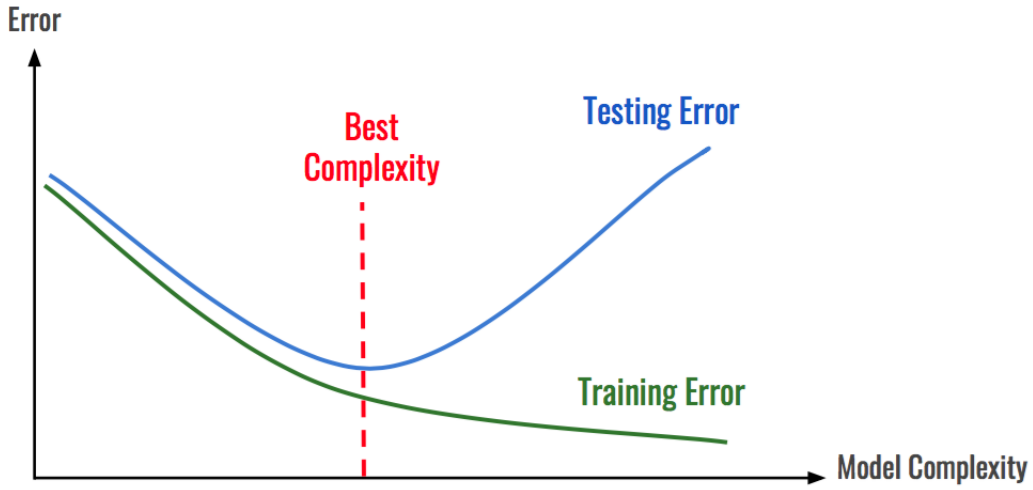


Figure 5.3: Many training interations can result in the model overfitting to the training set. A difference between the test error and training error can be observed when the model starts to be overly sensitive to the specific samples in the training set. Tricks can be used to prevent this from happening such as early-termination or regularization.

Chapter 6

Results and discussion

In this chapter we evaluate the results of the study by answering the research questions in order.

6.1 Research Question 1

> **RQ1/Research question 1:** *Which HoloLens imaging sensors are supported by state-of-the-art detectors?*

The answer to the first research question is derived from Chapter 2 and Chapter 6. We looked at the technical details of the HoloLens imaging sensors and provided an overview and selection of useful sensors. Computer vision detectors were studied and compatibility with imaging sensors is discussed.

6.1.1 Imaging sensors

The HoloLens imaging sensors have been mentioned in chapter 2.4.3. The technical details of the imaging sensors have been gathered and are provided in table 6.1.

Code	Name	Resolution	Framerate	Data type	Research sensor	Front-facing	Infrared	Fisheye lense	Selected
RGB	Photo video	1280*720	30Hz	RGB8		✓			✓
LtDepth	Long-throw depth	450*448	1Hz	D16	✓	✓	✓	✓	✓
StDepth	Short-throw depth	450*448	30Hz	D16	✓	✓	✓	✓	
LtRef	Long-throw reflectivity	450*448	1Hz	G8	✓	✓	✓	✓	✓
stRef	Short-throw refectivity	450*448	30Hz	G8	✓	✓	✓	✓	
VLLL	Visible light left left	640*480	30Hz	G8	✓				
VLLF	Visible light left front	640*480	30Hz	G8	✓				
VLRF	Visible light right front	640*480	30Hz	G8	✓				
VLRR	Visible light right right	640*480	30Hz	G8	✓				

Table 6.1: The collection of imaging sensors of the HoloLens 1 and their properties. Based on the properties a selection is made to be used in the experiment.

With the experiment we want to evaluate the effectiveness of the imaging sensors. We will select the imaging sensors that make sense for detecting a musical keyboard. It is important that the imaging sensor is front-facing, because the user will likely be using the device in front of him/her. The RGB sensor is selected because of its widespread use in augmented reality applications. It has the biggest image size, a high framerate and data richness in three color channels. It features a perspective projection with a 45° horizontal field of view when using image stabilization. In

practice, this requires the user to take at least two meters distance from the musical keyboard to have it fully enclosed in the picture frame.

There are short-throw and long-throw imaging sensors for both reflectivity and depth. The long-throw variants will be chosen, because the short-throw sensors cannot record data well at distances further than two meters. This means we select the long-throw reflectivity and long-throw depth imaging sensor. The visible light sensors are not front-facing and will therefore not be selected. It is worth noting that the long-throw imaging sensors have a framerate of 1Hz, this means that only one frame per second will be recorded. For generating at least 50 samples with margin we need at least one minute of choreography and footage.

We select the RGB, long-throw depth and reflectivity imaging sensors for dataset comparison.

6.1.2 Detectors

Many detectors exist, and a selection has been studied in chapter 2. An overview of the detectors is given in table 6.2.

Image detector	Output	Depth-aware	Open source	Online portal	Selected
Piano AR [3]	Image space key positions				
DeepIM [6]	6D pose estimation		✓		
YOLOv3 [4]	Object detection		✓		
MaskRCNN[5]	Image segmentation		✓		
DenseFusion [10]	6D pose estimation	✓	✓		
Keypoints [7]	6D pose estimation		✓		
SSD [8]	6D pose estimation		✓		
DBLP [9]	6D pose estimation	✓			
CustomVision.ai [24]	Object detection			✓	✓

Table 6.2: An overview of the studied image detectors and their properties.

The selected imaging sensors provide RGB, depth and reflectivity output. To accommodate the depth imaging sensor, attempts have been made to implement a depth-aware detector. However, correct implementation did not fit the project constraints and is therefore omitted. A specialized infrared reflectivity detector could not be found in the studied literature. As a solution we use the reflectivity and depth samples with a regular RGB detector. This data type mismatch is not necessarily a predictor for bad performance, it does however suggest that a specialized detector might achieve better performance. While keeping this in mind, it is still possible to make a useful imaging sensor comparison. See Section 4.3 for the steps taken to accommodate for differences in data types.

6.1.3 Support

We find that the studied detectors are either RGB detectors or depth-aware RGB detectors. The only perfect imaging sensor and detector match is the selected RGB Photo video imaging sensor with a selected non depth-aware detector. This combination will be the basis of the comparison and is a typical setup for computer vision.

Fisheye lense

The selected fisheye depth and reflectivity camera do not have a specialized fisheye detector. Most detectors require samples with a perspective projection, but some detectors can be adapted to support a projection texture. A solution would be to pre-process the fisheye samples to perspective projection beforehand but this can lead to performance degradation. Ideal would be to use a detector that inherently understands the fisheye projection. For object detection this projection mismatch can still be used but might influence performance. The size of this effect is unclear and has not been found in literature.

Data type

The selected reflectivity sensor outputs a G8 data type. This means eight bits of grayscale data per pixel. The selected detector has an RGB8 input, this means that the G8 sample will have to be pre-processed to an RGB8 sample. Populating the other channels with the same grayscale information will improve detector performance because of the increased feature count in the neural network. This is normally automatically done by the detector. The perfect match would be a detector that takes grayscale images as input and is optimized with the same amount of features as the three channel RGB detector.

Depth-aware

The same applies to the depth sensor which has a G16 pixel data type. The data will be pre-processed to a G8 pixel data type to support further processing. See Section 4.3 for the pre-processing steps. A better match would be to use an G16 pixel data type detector. Putting the first eight bits in the red channel and the second eight bits in the green channel seems to be a bad idea because of the way channels are implemented in convolutional neural networks. The composition of data for this value will not be preserved. The perfect match would be a depth-aware detector, most of these detectors still require an RGB8 image sample along with the depth sample with a similar projection, which also is not the case. The HoloLens can provide a reflectivity sample along with the depth sample. So the perfect match would be a fisheye G16 depth-aware and G8 reflectivity detector. This does not exist in the studied literature.

6.1.4 Summary

The RGB, reflectivity and depth imaging sensors provide a good range of possible sensors to use for mixed reality. The natural, artificial and dim lighting setups show how well the sensors perform in challenging illumination. Specialized RGB and RGB-Depth detectors are studied to implement them for comparison. It is found that infrared sensors do not have matching specialized detectors that can work with fisheye lenses or reflectivity data.

6.2 Research Question 2

> **RQ2/Research question 2:** *Which state-of-the-art detector is suitable for comparing imaging sensors?*

The answer to the second research question is derived from Chapter 2 and Chapter 6. We looked at the technical details of the various state-of-the-art computer vision detectors and provided an overview and selection of detectors. A suitable detector is chosen to perform dataset performance comparisons. See Table 6.2 for the studied detectors.

There is a big difference in investment between detectors. The investments are monetary, intellectual, hardware- and time-based. Even though some of the detectors are open source and can directly be downloaded to your computer, there is often a knowledge gap to overcome while working with them. Experience in the field is required to follow the tutorials and fill in the details that are not documented. Specific hardware, software and operating system are required to run the state-of-the-art detectors.

Attempts have been made to create an automatically annotated 6 degree-of-freedom pose estimation musical keyboard dataset, but have not succeeded due to technical reasons and constraints. As a backup, an object detection dataset is manually designed and created and computer vision techniques are filtered on this dataset requirement. See Chapter 4 for the dataset design.

Some detectors run in the cloud via a website, this poses an advantage in usability compared to the rest. These portals can also benefit companies that do not have sufficient in-house knowledge to work with the niche state-of-the-art detectors. The studied CustomVision.ai computer vision studio provides an easy-to-use online portal for doing manual labeling, training and testing. The cloud services charge money for training time and total number of cloud predictions. Calculations showed that cloud solutions were cheaper than running a local hardware setup for this specific project in time. CustomVision.ai supports both image recognition and object detection. We are interested in object detection as it more closely relates to the pose estimation problem.

All this combined motivated us to choose CustomVision.ai object detection on the compact domain as our comparison detector.

6.2.1 Summary

Many state-of-the-art detectors are technically suited for comparing imaging sensors. There is a preference for open-source 6D pose estimation detectors but these have proven to be time-consuming to implement correctly. The usability of CustomVision.ai's online portal motivated us to run RGB object detection comparisons on this platform, however it lacks more advanced computer vision types like segmentation and 6D pose estimation.

6.3 Experiment

RQ3 and RQ4 can be answered by the outcome of the experiment.

6.3.1 Research Question 3

> **RQ3/Research question 3:** *To what extent do infrared imaging sensors improve detector performance in a dim lighting setup compared to an RGB imaging sensor?*

The answer to the third research question is derived from Chapter 6. The infrared imaging sensors in the experiment are the reflectivity and depth sensors. We compared the performance of reflectivity and depth imaging sensors to the RGB sensor in a dim lighting setup.

The performance of the imaging sensors is described in the metric mean Average Precision (mAP), this is a single number that shows generalizable performance at a range of common overlap intervals. The performance of the imaging sensors is shown in Figure 6.1. We see that reflectivity and depth outperform the RGB sensor by 89.6% and 14.9% respectively. The software was not able to calculate standard deviations, so p-values cannot be truthfully calculated either. Based on a standard deviation estimate of $\sigma = 5\%$[1] for all means, the p-values are 0% and 19.64% respectively. This is calculated with a two means two-tailed t-test with a known sample standard deviation. Assuming a sample size of 10 with k-fold cross validation for all means to calculate the precision and recall. The mAP metric is then calculated from the precision recall plot. This means significance for reflectivity and insignificance for depth performance comparisons at a significance level of 5%. For a 5% standard deviation the assumption is that at least 68.27% of the population data lies in the following range $15.2 - 25.2\%$. If this assumption is true, the above holds true.

The difference in overall performance of imaging sensors is explained by higher precision and recall for higher Intersection over Union (IoU) thresholds. If bounding box performance is not important, then lower quality datasets might suffice. See Figure 6.2 for the performance of imaging sensors in a dim lighting setup at different Intersection over Union (IoU) thresholds.

The significantly higher performance of the reflectivity imaging sensor means that the infrared image stability outweighs the lower resolution, fisheye projection and single grayscale channel. In essence the reflectivity sensor uses its own infrared lighting to illuminate the scene and filters out other bands of light. This means it has to deal with a less diverse dataset compared to the RGB sensor. Why the depth imaging sensor is not able to achieve similar performance is not directly clear. In Figure 4.1 we see examples of images in the datasets. The natural lighting setup has the blinds opened to allow natural light to illuminate the scene. The glass or blinds background in the dataset might explain the difference in performance for the depth imaging sensor but no intuitive reason can be formulated. More research is required to verify this behavior.

Summary

The reflectivity imaging sensor shows significantly better performance compared to the baseline RGB sensor at +89.6%. The depth sensor is not able to reach a significant difference. The reflectivity sensor benefits from the infrared illumination that provides image stability and data richness. Why the depth sensor is not able to perform as well as the reflectivity sensor is not directly clear.

[1] A strict standard deviation of 5% is used in combination with at least 50 samples per dataset to prevent overfitting[1]. We believe that the used standard deviation is at least higher than the actual value. This motivates us to believe that the stated significance is indeed achieved.

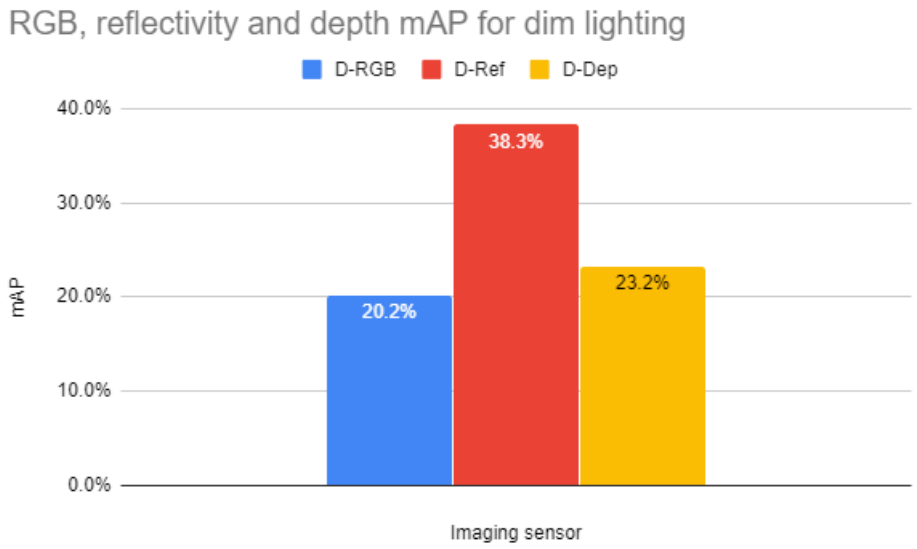


Figure 6.1: The mean Average Precision (mAP) for RGB, reflectivity and depth imaging sensors in a dim lighting setup.

6.3.2 Research Question 4

RQ4/Research question 4: *To what extent do natural and artificial light influence infrared imaging sensor detector performance?*

The answer to the fourth research question is derived from Chapter 6. The infrared imaging sensors in the experiment are the reflectivity and depth sensors. We compared the performance of natural and artificial lighting setups to the dim lighting setup for the reflectivity and depth imaging sensor.

Figure 6.3 shows the performance of all imaging sensors in all lighting setups of the experiment. The RGB and reflectivity sensors show a trend of similar performance in different lighting setups with natural being the best, then artificial and lastly dim. The depth sensor shows performance in natural light similar to the reflectivity sensor, but fails to perform as well in artificial and dim light. Natural light gives the best performance for all imaging sensors, this is counterintuitive because the assumption is that natural light contains infrared light that can interfere with the reflectivity and depth sensors. The HoloLens uses near infrared light which can pass through glass, so the natural light is able to interfere with the sensor.

We see that for the reflectivity sensor natural and artificial light surprisingly outperform the dim lighting setup by 28.5% and 14.9% respectively. For the depth sensor we see that natural light outperforms the dim lighting setup by 97.8% and artificial light underperforms the dim lighting setup with -20.3%. The software was not able to calculate standard deviations, so p-values cannot be truthfully calculated either. Based on a standard deviation estimate of $\sigma = 5\%$[2] for all means, the p-values are 0.01%, 2.01%, 0% and 4.99% respectively. This is calculated with a two means two-tailed t-test with a known sample standard deviation. Assuming a sample size of 10 with k-fold cross validation for all means to calculate the precision and recall. The mAP metric is then calculated from the precision recall plot. This means significance for all natural and artificial light performance comparisons for the reflectivity and depth imaging sensor at a significance level of 5%. For a 5% standard deviation the assumption is that at least 68.27% of the population data lies in the following ranges $33.3 - 43.3\%$ and $18.2 - 28.2\%$ for reflectivity and depth imaging sensor respectively. If this assumption is true, the above holds true.

[2] A strict standard deviation of 5% is used in combination with at least 50 samples per dataset to prevent overfitting[1]. We believe that the used standard deviation is at least higher than the actual value. This motivates us to believe that the stated significance is indeed achieved.

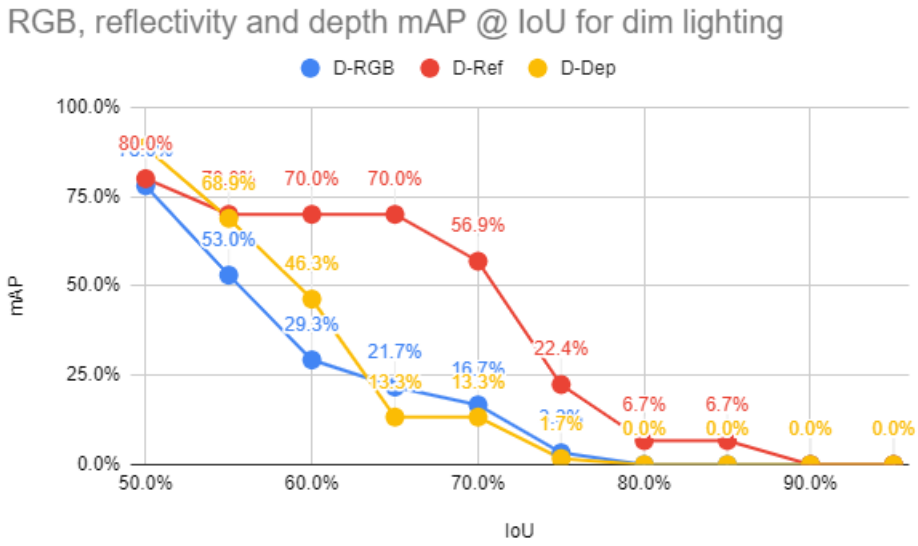


Figure 6.2: The mean Average Precision (mAP) for RGB, reflectivity and depth imaging sensors in a dim lighting setup at various Intersection over Union (IoU) thresholds.

Summary

Natural and artificial light seem to significantly influence infrared imaging sensor performance at +28.5% and +14.9% for the reflectivity sensor and +97.8% and −20.3% for the depth sensor respectively. The performance is best for natural lighting setups for both reflectivity and depth sensors. It is not clear why this is the case. Artificial light marginally improved performance for the reflectivity sensor but marginally decreased performance for the depth sensor compared to a dim lighting setup.

6.3.3 Overexposure

Some of the RGB natural (N-RGB) dataset samples are overexposed. See Figure 4.4 for an example. This might influence detector performance and more research is required to verify this might pose a problem.

6.3.4 Depth

The depth imaging sensor does not perform as well as the reflectivity sensor in artificial and dim light. This is not according to expectations. This can be due to loss of data while pre-processing from sixteen bits to eight bits of data per pixel. But even then the reflectivity sensor is able to perform better with the same bit depth. Maybe a smaller range than $[0, 4080]$ must have been chosen. This range includes all possible far sensor values and might be the cause of losing the very important smaller changes in depth.

6.3.5 Precision or recall

The mAP metric calculates detector performance that balances precision and recall. This is useful to compare overall performance of datasets or detectors but for real applications a connection to the implications of such a detection should be made. In this project the dataset can be used for detecting a musical keyboard with the HoloLens mixed reality headset. Initial experiments with a CustomVision.ai compact domain neural network running on the HoloLens resulted in a detection every three seconds. This is not considered real time because not all the frames are processed in time. The RGB sensor provides 30 frames per second and the reflectivity and depth sensor provide one frame per second. This can be remedied by off-device processing that increases latency, implementing a faster detector or skipping the frames that arrive while processing. The latter is chosen for the demo application to provide an on-device and experience. Multiple predictions can be made in succession and as long as the object is stationary these build up for a more precise

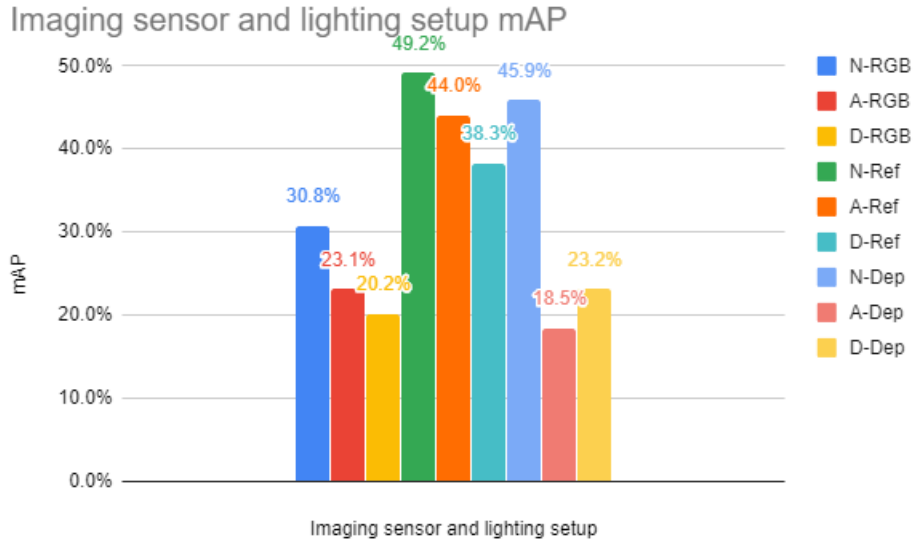


Figure 6.3: The mean Average Precision (mAP) for all imaging sensors and all lighting setups in the experiment.

prediction. Random sample consensus (RANSAC) can be used to filter inaccurate detections[30]. This means that the cost of a false negative is higher than a false positive. Thus missing a detection is worse than detecting an incorrect keyboard. We favor recall over precision.

If the user is very impatient, the user can get feedback where the application found the musical keyboard. If the user disagrees, the application can gather more samples. The cost of longer processing is rather low, the biggest risk is losing a user. Tricks can be implemented to divert the attention of the user from the detection time and this proves to be a successful strategy in literature.

The best performing imaging sensor in the experiment is the reflectivity sensor. It comes with three precision recall curves for the lighting setups natural, artificial and dim. This can be generated for every intersection over union threshold. A common threshold of 50% is chosen and results in the following Figure 6.4. The mAP@overlap metric for intersection over union thresholds can be calculated by the surface of the curve. Averaging this balanced precision recall metric for the range $[0.5, 0.05, 0.95]$ gives the final mAP measure. The natural lighting precision recall curve shows dents where precision drops. A balance between precision and recall can be made here. To favor precision, a recall can be chosen of 60% that offers 100% precision performance. To favor recall, a recall can be chosen of 90% that offers 81.8%. In practice this recall value should be translated to its original probability or confidence threshold. For the N-Ref detector the maximum confidence threshold that outputs this precision and recall for 50% overlap threshold is 26%. This threshold will improve recall with $+30.0\%$ at the cost of -18.2% precision compared to 100% precision. We also determined that the suggested application favors recall over precision because of a lower cost for lack of precision.

See Figure 6.5 for the balance of precision and recall for various probability thresholds. It is not very intuitive to pick an optimal probability threshold from this graph, but using a precision recall curve this is instantly clear.

6.3.6 Internal consistency

The incorrectly reported standard deviations of the precision and recall metrics for all tests equal 0. They are not displayed in the visual portal of CustomVision.ai, but can only be found when looking at the raw JSON responses. This is a strong indication that the standard deviations are not correctly calculated. Applying cross-validation and calculating standard deviations outside of the CustomVision.ai portal is found to be infeasible. Therefore we are forced to omit them from the study. We cannot make judgements about internal consistency of the dataset based on these missing metrics.

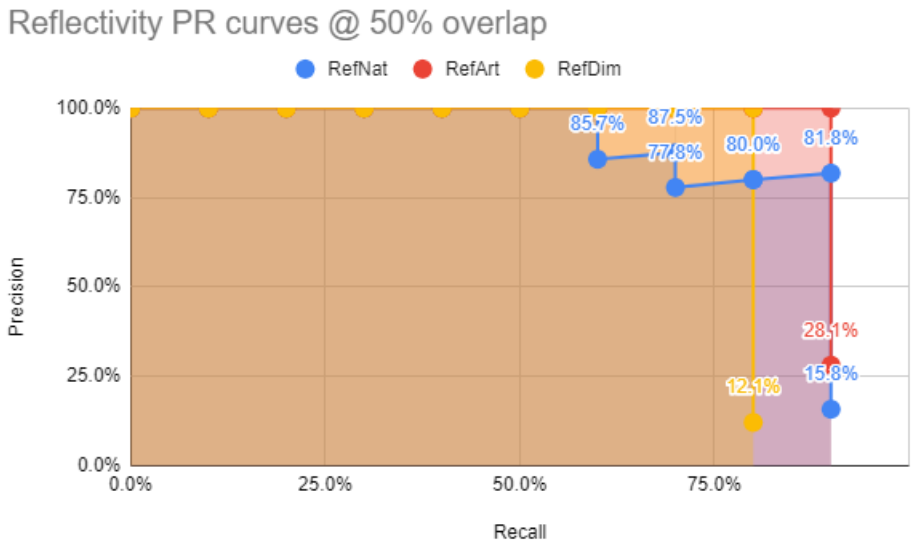


Figure 6.4: Reflectivity precision recall curve at 50% intersection over union threshold. The precision is calculated and shown with a recall stepping size of 10%.

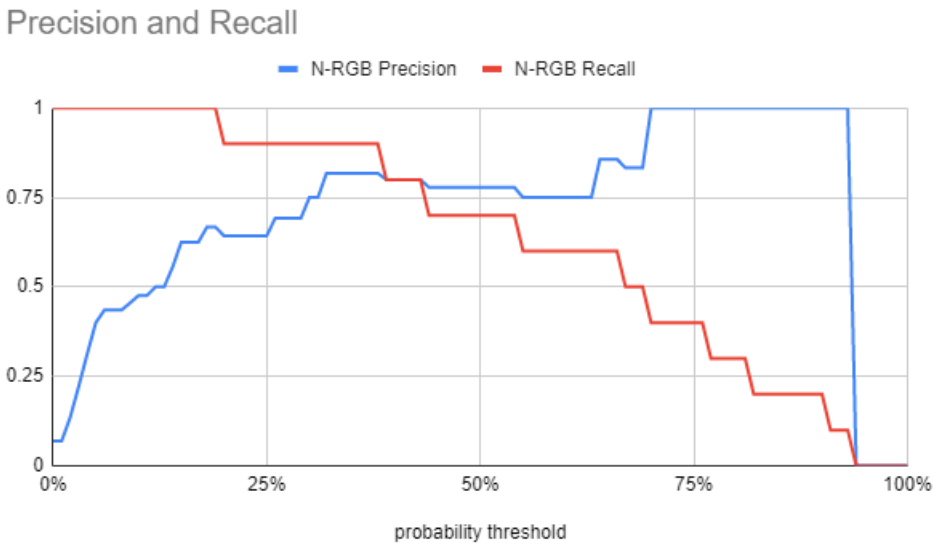


Figure 6.5: Precision and recall for various probability thresholds for the natural lighting setup and RGB imaging sensor at 50% overlap threshold.

Chapter 7

Conclusion

The following chapter concludes this work. Section 7.1 provides answers to the research questions, Section 7.2 connects the research question answers to the problem statement and Section 7.3 discusses future research directions.

We discussed and compared the useful imaging sensors that the HoloLens provides and detectors to do mixed reality computer vision in lighting setups to get an understanding of how to approach new mixed reality challenges. The literature provides many studies on comparing detectors, but comparing datasets built from different sensors and lighting setups is new. The focus of the detection performance is on recall instead of precision.

A new dataset is introduced, designed and created that explores the two independent variables imaging sensor and lighting setup. This results in nine setups for the dataset which can be used for performance comparison. The dependent variable detector performance is a direct indication of dataset quality and thus the ability of the imaging sensor to provide data richness in a lighting setup.

7.1 Answers to the research questions

In Chapter 1 we formulated four research questions. We will discuss every research question and provide an answer based on the previous chapters.

> **RQ1/Research question 1:** *Which HoloLens imaging sensors are supported by state-of-the-art detectors?*

The answer to the first research question is derived from Chapter 2 and Chapter 6. We looked at the technical details of the HoloLens imaging sensors and provided an overview and selection of useful sensors. Computer vision detectors were studied and compatibility with imaging sensors is discussed.

The RGB, reflectivity and depth imaging sensors provide a good range of possible sensors to use for mixed reality. The natural, artificial and dim lighting setups show how well the sensors perform in challenging illumination. Specialized RGB and RGB-Depth detectors are studied to implement them for comparison. It is found that infrared sensors do not have matching specialized detectors that can work with fisheye lenses or reflectivity data.

> **RQ2/Research question 2:** *Which state-of-the-art detector is suitable for comparing imaging sensors?*

The answer to the second research question is derived from Chapter 2 and Chapter 6. We looked at the technical details of the various state-of-the-art computer vision detectors and provided an overview and selection of detectors. A suitable detector is chosen to perform dataset performance comparisons.

Many state-of-the-art detectors are technically suited for comparing imaging sensors. There is a preference for open-source 6D pose estimation detectors but these have proven to be time-consuming to implement correctly. The usability of CustomVision.ai's online portal motivated us to run RGB object detection comparisons on this platform, however it lacks more advanced computer vision types like segmentation and 6D pose estimation.

> **RQ3/Research question 3:** *To what extent do infrared imaging sensors improve detector performance in a dim lighting setup compared to an RGB imaging sensor?*

The answer to the third research question is derived from Chapter 6. The infrared imaging sensors in the experiment are the reflectivity and depth sensors. We compared the performance of reflectivity and depth imaging sensors to the RGB sensor in a dim lighting setup.

The reflectivity imaging sensor shows significantly better performance compared to the baseline RGB sensor at +89.6%. The depth sensor is not able to reach a significant difference. The reflectivity sensor benefits from the infrared illumination that provides image stability and data richness. Why the depth sensor is not able to perform as well as the reflectivity sensor is not directly clear.

> **RQ4/Research question 4:** *To what extent do natural and artificial light influence infrared imaging sensor detector performance?*

The answer to the fourth research question is derived from Chapter 6. We compared the performance of natural and artificial lighting setups to the dim lighting setup for the reflectivity and depth imaging sensor.

Natural and artificial light seem to significantly influence infrared imaging sensor performance at +28.5% and +14.9% for the reflectivity sensor and +97.8% and −20.3% for the depth sensor respectively. The performance is best for natural lighting setups for both reflectivity and depth sensors. It is not clear why this is the case. Artificial light marginally improved performance for the reflectivity sensor but marginally decreased performance for the depth sensor compared to a dim lighting setup.

7.2 Answer to the problem statement

In this section we answer the problem statement given in Chapter 1. We use the answers from the research questions found in the previous section.

> **PS/Problem statement:** *To what extent are HoloLens imaging sensors able to solve computer vision tasks in natural, artificial and dim lighting setups?*

We have concluded that the RGB, reflectivity and depth imaging sensors all pose candidates for computer vision tasks on the HoloLens mixed reality headset. However no specialized detectors for the reflectivity and depth fisheye lenses have been found in the literature. Using depth-aware and 6D pose estimation detectors requires prior experience and proves to be time-consuming to implement. Various cloud-based computer vision portals exist like CustomVision.ai that enable training an object detector but with limited metrics and export functionality.

The RGB imaging sensor provides balanced detection performance across all lighting setups for scenarios where infrared imaging sensors are not available. This is useful because RGB sensors are common and can directly be used for augmented reality applications.

From the research question answers we conclude that the reflectivity imaging sensor provides superior object detection performance across all lighting setups for musical keyboards. The infrared illumination is effective in a dim lighting setup, while natural and artificial light do not impair its performance. The infrared illumination provides a stable image of the environment that improves detector performance compared to baseline RGB performance.

The depth sensor shows detector performance on par with the reflectivity sensor for a natural lighting setup, but performs similar to the baseline RGB sensor for artificial and dim lighting setups. No logical connection can be made to what might cause this. It is possible that a well-tuned depth-aware detector outperforms the reflectivity sensor.

For most mixed reality applications recall is preferred over precision, because users can reject a hologram placement or techniques like random sample consensus (RANSAC)[30] can be used to filter false positives from a population of detections. The overlap threshold is set to the required bounding box placement precision requirements and the confidence threshold is set to correspond with a peak in the precision recall curve to favor recall over detection precision.

Based on the answers to the research questions we conclude that HoloLens imaging sensors are able to solve computer vision tasks in natural, artificial and dim lighting setups. A reflectivity imaging sensor will outperform an RGB sensor for object detection tasks in natural, artificial and

dim lighting setups and more research is required for depth imaging sensor performance. Careful planning on imaging sensors, lighting setups and detectors should be done when implementing new mixed reality applications to ensure optimal performance.

7.3 Future research

The main limitations of the study include not using a depth-aware detector for the depth imaging sensor. The results of the depth imaging sensor are not useful for making informed decisions on whether or not to use a depth imaging sensor. More research is required to substantiate claims for or against using the depth imaging sensor.

Another limitation is the use of a fisheye lense for the reflectivity and depth imaging sensor in combination with a perspective projection detector. The reflectivity sensor shows superior performance compared to the RGB sensor, but this might even be higher with a specialized fisheye lense detector or pre-processing to perspective projection.

A limitation is using object detection with bounding boxes instead of full 6D pose estimation for the performance comparisons. This fact might skew performance metrics when sensor image projections are different. It is not directly clear how and how much this influences performance, but not using image space metrics alleviates this problem. Pixels inside the bounding box but oustide the object are also used for learning features in a detector, segmentation approaches solve this issue. The effect of this limitation is not clear either.

The final limitation is only providing 50 samples for every dataset setup, this is recommended as a minimum sample count to prevent overfitting[1], but most detector performance comparisons are done with 1000+ samples. An automatic annotation system should be used when creating such a dataset to prevent extensive manual labeling.

Appendix A

Time plan

Figure A.1 below shows the time plan in detail.

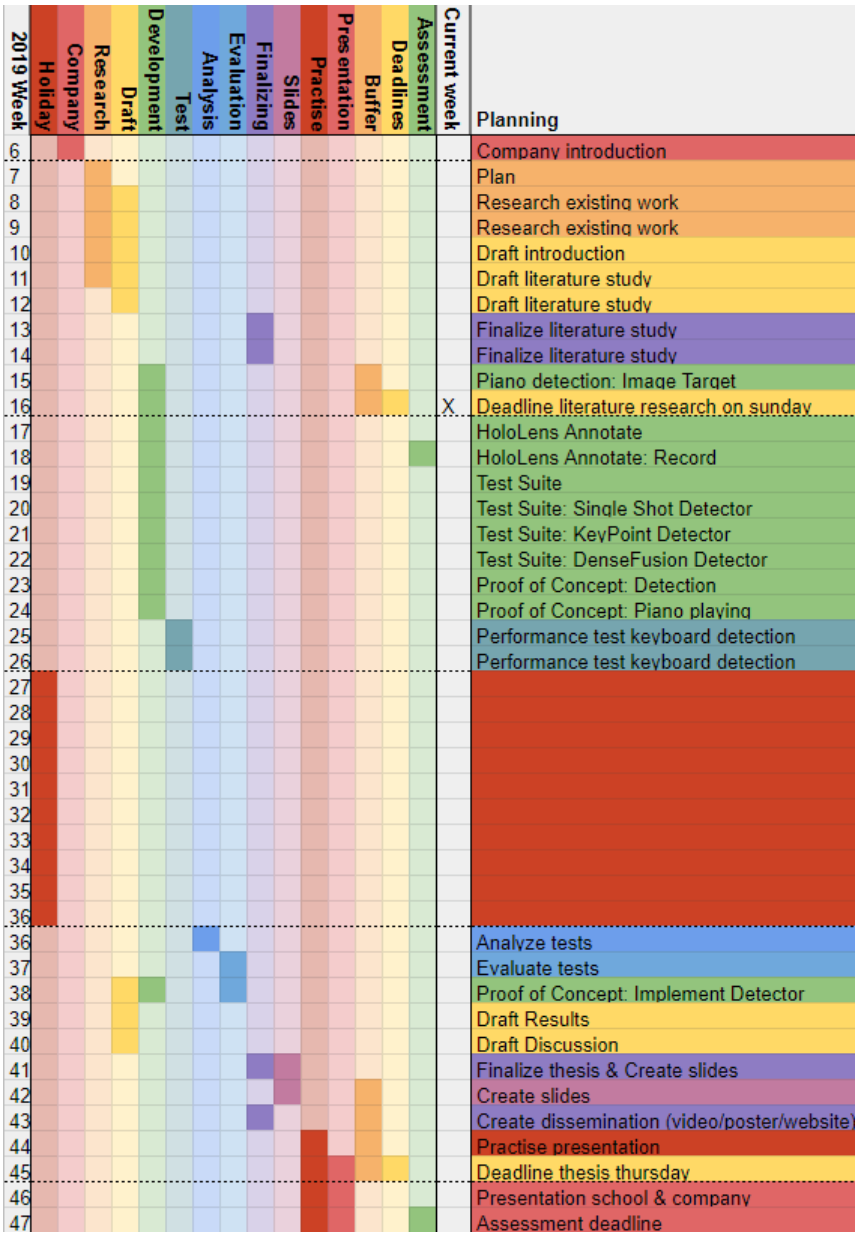


Figure A.1: The time plan of the project with activity types color-coded.

Appendix B

HoloLens 1 and 2 specifications

The hardware specifications of the HoloLens 1 and 2 are found in Table B.1.

Specifications	HoloLens 1	HoloLens 2
Market availability	*March 30, 2016*	*Estimated: Q3 2019*
Price	*3,000 US dollars*	*3,500 US dollars*
Resolution	*2.3 MP*	*2K*
Pixels per degree	*47*	*47*
Diagonal angle of view	*34 degrees*	*52 degrees*
Aspect ratio	*16:9*	*3:2*
Lenses	*Waveguide*	*Waveguide*
Weight	*579g*	*To be announced*
Eye-based rendering	*No*	*Yes*
IMU Sensors	*Accelero, Gyro, Magneto*	*Accelero, Gyro, Magneto*
Camera still	*2.4MP*	*8MP*
Camera video	*1280*720*	*1080p30*
Depth camera	*120x120 degrees of view*	*Azure Kinect sensor*
Microphone array	*4*	*5*
Speakers	*HRTF spatial sound*	*HRTF spatial sound*
Hand Tracking	*Simple gestures*	*Full tracking, manipulation*
Eye Tracking	*No*	*Real-time tracking*
6DoF Tracking	*Yes, 4 Sensors*	*Yes*
Spatial Mapping	*Yes*	*Real-time environment mesh*
Mixed Reality Capture	*Up to 720p*	*Video 1080p, photo 4K*
Processor	*Intel 32-bit (1GHz), TPM 2.0*	*Qualcomm Snapdragon 850*
GPU	*Intel*	*Adreno 630*
HPU	*HPU 1.0*	*HPU 2.0 + DNN AI coprocessor*
Memory	*2(+1 HPU) GB RAM*	*To be announced*
Storage	*64 GB (flash memory)*	*To be announced*
Bluetooth	*4.1 Low Energy*	*5.0*
Wireless Wi-Fi	*802.11ac*	*802.11ac 2x2*

Table B.1: An overview of the HoloLens 1 and 2 specifications. This work focuses on the hardware capabilities of the HoloLens 1. Specifications gathered from [31] and [32] on 2019-04-01.

Bibliography

[1] Various contributors, *How to improve your classifier*, https://docs.microsoft.com/en-us/azure/cognitive-services/custom-vision-service/getting-started-improving-your-classifier#data-quantity, Accessed: 2019-10-16.

[2] PapersWithCode, *Papers with code — 6d pose estimation*, https://paperswithcode.com/task/6d-pose-estimation, Accessed: 2019-04-05.

[3] F. Huang, Y. Zhou, Y. Yu, Z. Wang, and S. Du, "Piano ar: A markerless augmented reality based piano teaching system," vol. 2, Sep. 2011, pp. 47–52. DOI: 10.1109/IHMSC.2011.82.

[4] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. arXiv: 1804.02767. [Online]. Available: http://arxiv.org/abs/1804.02767.

[5] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017. arXiv: 1703.06870. [Online]. Available: http://arxiv.org/abs/1703.06870.

[6] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation," *CoRR*, vol. abs/1804.00175, 2018. arXiv: 1804.00175. [Online]. Available: http://arxiv.org/abs/1804.00175.

[7] Z. Zhao, G. Peng, H. Wang, H. Fang, C. Li, and C. Lu, "Estimating 6d pose from localizing designated surface keypoints," *CoRR*, vol. abs/1812.01387, 2018. arXiv: 1812.01387. [Online]. Available: http://arxiv.org/abs/1812.01387.

[8] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," *CoRR*, vol. abs/1711.08848, 2017. arXiv: 1711.08848. [Online]. Available: http://arxiv.org/abs/1711.08848.

[9] Y. Konishi, K. Hattori, and M. Hashimoto, "Real-time 6d object pose estimation on CPU," *CoRR*, vol. abs/1811.08588, 2018. arXiv: 1811.08588. [Online]. Available: http://arxiv.org/abs/1811.08588.

[10] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," *CoRR*, vol. abs/1901.04780, 2019. arXiv: 1901.04780. [Online]. Available: http://arxiv.org/abs/1901.04780.

[11] Wikipedia contributors, *Musical keyboard — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Musical_keyboard&oldid=887951801, Accessed: 2019-04-05.

[12] Marc Pollefeys, *Microsoft hololens facilitates computer vision research by providing access to raw image sensor streams with research mode*, https://www.microsoft.com/en-us/research/blog/microsoft-hololens-facilitates-computer-vision-research-by-providing-access-to-raw-image-sensor-streams-with-research-mode/, Accessed: 2019-04-08.

[13] Wikipedia contributors, *Ramer–douglas–peucker algorithm — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Ramer%E2%80%93Douglas%E2%80%93Peucker_algorithm&oldid=887702935, Accessed: 2019-04-02.

[14] Wikipedia contributors, *Perspective-n-point — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Perspective-n-Point&oldid=886390095, Accessed: 2019-04-02.

[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," vol. 1, Feb. 2001, pp. I–511, ISBN: 0-7695-1272-0. DOI: 10.1109/CVPR.2001.990517.

[16] Andrej Karpathy, *Cs231n convolutional neural networks for visual recognition —- neural networks*, http://cs231n.github.io/neural-networks-1/, Accessed: 2019-04-16.

[17] Jayesh Bapu Ahire, *The artificial neural networks handbook: Part 4*, https://medium.com/@jayeshbahire/the-artificial-neural-networks-handbook-part-4-d2087d1f583e, Accessed: 2019-04-16.

[18] Sagar Sharma, *Activation functions in neural networks*, https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6, Accessed: 2019-04-16.

[19] Jaswanth Sreeram, Stephan Herhut, Lindsey Kuper, *Bringing parallelism to the web with river trail*, http://intellabs.github.io/RiverTrail/tutorial/, Accessed: 2019-04-16.

[20] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," Jan. 2009, p. 77. DOI: 10.1145/1553374.1553453.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, Sep. 2014.

[22] COCO Consortium, *Panoptic evaluation*, http://cocodataset.org/#panoptic-eval, Accessed: 2019-04-16.

[23] S. Hinterstoißer, V. Lepetit, S. Ilic, S. Holzer, K. Konolige, G. R. Bradski, and N. Navab, "Technical demonstration on model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *ECCV Workshops*, 2012.

[24] Microsoft, *Customvision*, https://www.customvision.ai/, Accessed: 2019-10-18.

[25] S. Glickman, B. Lee, F. Y. Hsiao, and S. Das, "Music everywhere - augmented reality piano improvisation learning system," in *NIME*, 2017.

[26] Thomas Aalbers, *Betabit - git - holopiano*, https://betabit.visualstudio.com/Betabit.HoloPiano/_git/Betabit.HoloPiano, Accessed: 2019-04-18.

[27] Thomas Aalbers, *Holopiano thesis*, http://HoloPiano.TomHash.NL, Accessed: 2019-10-25.

[28] Microsoft, *Hololensforcv*, https://github.com/microsoft/HoloLensForCV, Accessed: 2019-04-03.

[29] The curious learner, *Confused by the confusion matrix: What's the difference between hit rate, true positive rate, sensitivity, recall and statistical power?* https://learncuriously.wordpress.com/2018/10/21/confused-by-the-confusion-matrix/, Accessed: 2019-10-25.

[30] Wikipedia contributors, *Random sample consensus — Wikipedia, the free encyclopedia*, https://en.wikipedia.org/w/index.php?title=Random_sample_consensus&oldid=917635122, [Online; accessed 4-November-2019], 2019.

[31] Microsoft, *Microsoft hololens 1 spec sheet*, https://www.bechtle.com/shop/medias/5a1430f79ce96955066d0b10.pdf?context=bWFzdGVyfHJvb3R8NjA0ODN8YXBwbGljYXRpb24vcGRmfGhkMi9oYTIvOTQ3MDY4ODY5MDIwNi5wZGZ8NWFkOWE2N2JhNTdjZjEwNDdhZWNkNGQwYzJlM2IwZThiMmZjYzYyMjIzZTllNzVlMGY3OWYyZTQ5ZDU5NTkyMw, Accessed: 2019-04-01.

[32] B. Lang, *Microsoft hololens 2 spec update*, https://www.roadtovr.com/hololens-2-specs-resolution-field-of-view-battery-life/, Accessed: 2019-04-01.