



UNIVERSITÀ DEGLI STUDI DI PADOVA

Master's Degree in Computer Engineering

Department of Information Engineering

Algorithms for 3D hand pose extraction: a novel implementation

Professor:

Menegatti Emanuele

Supervisor:

Steiner Florian

Student:

Trapanotto Martino

M. 2044729

A.A. 2023/2024

Abstract

In this work we construct a marker-less 3D hand pose extraction solution relying solely on a single RGB video source. We leverage available solutions for state-of-the-art computer vision models, coupled with a custom depth estimation algorithm to reconstruct full 3D space data. This system is packaged in a Python library capable of real time pose estimation, and is evaluated over two different datasets, one reflecting our intended task and a common academic dataset. The overall results shows limits in the accuracy of the model, measured on average above 10 cm, exceeding the desired precision of a few millimeters, but we acknowledge the relevancy of the results in the larger context of the field, and discuss potential avenues for future work.

Contents

List of Figures	5
1 Introduction	7
1.1 Marker less and marker based techniques	8
1.2 2D, 3D and 2.5D poses	10
1.3 Multiview, Depth cameras and RGB cameras	11
1.4 Hand pose specific problems	17
1.5 Conclusions	18
2 State of the Art	21
2.1 InterNet	23
2.2 PeCLR	26
2.3 Mediapipe	28
2.4 A2J-Transformer	31
2.5 HTT	33
2.6 HDR	35
2.7 Conclusions	37
3 Custom solution	39
3.1 Depth estimation via apparent size	41
3.1.1 Calibration phase	44
3.1.2 3D pose reconstruction	45
3.2 Overall resulting library	46
3.2.1 Data export and visualization	47
3.3 Evaluations and Results	49
3.4 Conclusions	58
Bibliography	61

List of Figures

1.1	Examples of markers and marker based pose estimation solutions [Wang et al., 2023] [Merlau et al., 2023] [Wade et al., 2022]	8
1.2	Examples of markerless based pose estimation results [Christian Zimmermann and Brox, 2019] [Bartol et al., 2020]	9
1.3	Examples of depth images [Tompson et al.,] [Garcia-Hernando et al., 2018]	12
1.4	Examples of multi view setups [Moon et al., 2020] [Christian Zimmermann and Brox, 2019]	15
2.1	Architecture of the InterNet model [Moon et al., 2020]	23
2.2	Examples of InterNet results [Moon et al., 2020]	24
2.3	Example of contrastive learning: the distance represent similarity scores [Schroff et al., 2015]	26
2.4	Overview of contrastive learning in PeCLR	27
2.5	Overview of the architecture Mediapipe Hands, constructed using the Mediapipe Framework [Zhang et al., 2020]	29
2.6	Overview of the internal architecture of the A2J-Transformer model [Jiang et al., 2023]	32
2.7	Overview of the internal architecture of the HTT model [Wen et al., 2023]	34
2.8	Overview of the internal architecture of the HDR model [Meng et al., 2022]	35
3.1	Map of the keypoints generated by Mediapipe [med, 2024]	41
3.2	Visualization of the perspective geometry effect	42
3.3	Visualization of the 2.5D keypoints reference system	43
3.4	Folder structure of the Python package	46
3.5	Example of library displaying both 2D and 3D keypoints	48
3.6	Visualization of the evaluation using the RGBD camera, comparing the estimated depth (blue) and measured depth (red). Also visible are the 2D keypoints with measured distance and the depth map.	50
3.7	Examples of the pointing task, with small colored marks for 2D objective keypoints and measured keypoints	51

3.8	3D visualization of the results from the GoPro camera	52
3.9	Distribution of the errors values, both absolute and along the individual axes . . .	53
3.10	Examples of images from the HanCo dataset	54
3.11	Distribution of the errors values, both absolute and along the individual axes . . .	55
3.12	Distribution of the errors values along each axis, separated per hand joint	55
3.13	Examples of images used for calibration of the depth estimation model, from the HanCo dataset	56

Chapter 1

Introduction

In this work we will construct a marker less solution to extract 3D hand pose data from a single RGB view. We will start by exploring the general context of pose estimation, especially for human pose problems, and the variety of solutions and conditions these tasks are tackled in. In this, we will be discussing the exact meaning of this task and how it relates to the general task of pose detection, what challenges it defines and how these will impact the needs of our solution, exploring the differences between marker based and marker-less solutions, depth and RGB data, single and multi view systems. We will also explore the unique challenges introduced by human hands from the wider context of pose extraction.

Next, we will explore the existing set of models and solutions available on the topic, focusing on candidates that provide trained models and the needed infrastructure and documentation to exploit these technologies. These models will also be discussed in their design and objectives, before attempting to redeploy them for our needs. Candidates that meet the requirements of speed and accuracy have been selected for further analysis.

These selected model will then be associated with any possibly needed algorithm to complete the extraction process, allowing full reconstruction of 3D hand pose data based solely on the RGB information. These solutions will then be structured into a Python library and evaluated, against two datasets, one reflecting the nature of the desired task, the other a conventional academic dataset widely used in hand pose estimation research.

Hand pose estimation is a highly active field of research inside the general topic of computer vision, where the objective is to estimate and reconstruct the position of a users' hands inside a video or image, captured with one or more sensors, alongside potential other sources of data. The general task of articulated pose estimation, pertaining also to full human bodies, animals

and other entities, has been an active field of research in computer vision for decades, due to its wide range of applications and uses but also its high complexity, wide range of conditions and potential environments and formulations. [Ahmad et al., 2019] [Moeslund and Granum, 2001]

1.1 Marker less and marker based techniques

An important distinction when discussing methods to extract hand poses, is between marker-based solutions and marker-less ones.

Marker based solutions employ an array of markers and other identification methods to simplify the task at hand, focusing on tracking these markers in the scene instead of the hands or bodies themselves.

These techniques are widely employed in various fields as they remove many potential sources of noise and error, providing high precision and fast motion data[Brown et al., 2018] [Topley and Richards, 2020], although they do pose distinctive problems[Seethapathi et al., 2019].

Markers can be of various types based on the specific task, requirements and available hardware, such as purely color based features, specialized patterns suited for automated tracking or specialized materials that reflect non-visible light. Cameras used range widely, from single view RGB systems to complex synchronized multi camera Infrared arrays. [Colyer et al., 2018a]

This wide range of possible setups allows for control over the performance of the system and to adapt to various budgets. Moreover, marker-based solutions show high precision, have well established commercial solutions available for multiple use cases and have a long history of usage in multiple fields, such as motion capture for media production, robotics and machine vision tasks, special effects work and medical imaging. [Topley and Richards, 2020] [Basafa et al., 2017] [Lee and soo Lee, 2018]

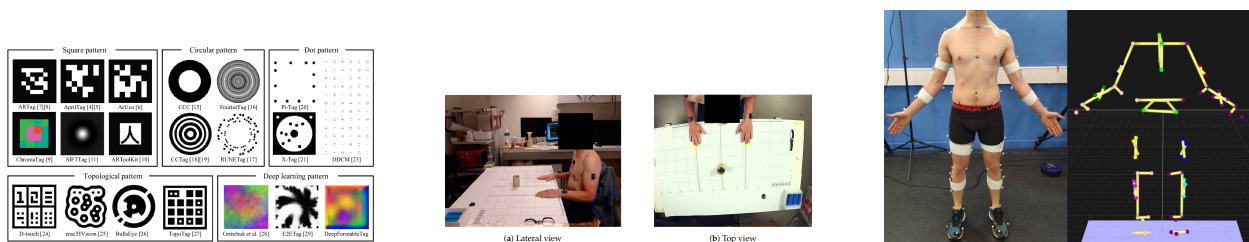


Figure 1.1: Examples of markers and marker based pose estimation solutions [Wang et al., 2023] [Merlau et al., 2023] [Wade et al., 2022]

Marker-based solutions exhibit some major issues.

- **Additional hardware requirements** raising costs and reducing available data
- **Marker application** requiring specialized training and adding complexity
- **Interference with data collection** as markers presence might disturb natural movements [Mündermann et al., 2006]
- **Inherent error** as the data tracks not the true pose, but the markers themselves
- **Stricter constraints on capture environment** due to the complex setup required

Marker-less techniques instead produce less accurate results, but at lower overall costs and without needs for complex setups or user training. They also allow for more natural usage, removing constraints in environment conditions and tracker usage, opening up possibilities of collecting and utilizing data produced in the wild.

In these solutions, the pose estimation works directly on the data produced by the camera systems, locating the keypoints in the image without the aide of markers, relying instead on visual features. [Colyer et al., 2018b] These solutions have existed for decades, but have grown anew thank to advancements in deep learning and computer vision[Chen et al., 2020c], especially convolutional neural networks (CNNs), which proved to be an incredibly powerful tool for image processing, feature detection and keypoint location[Colyer et al., 2018b].

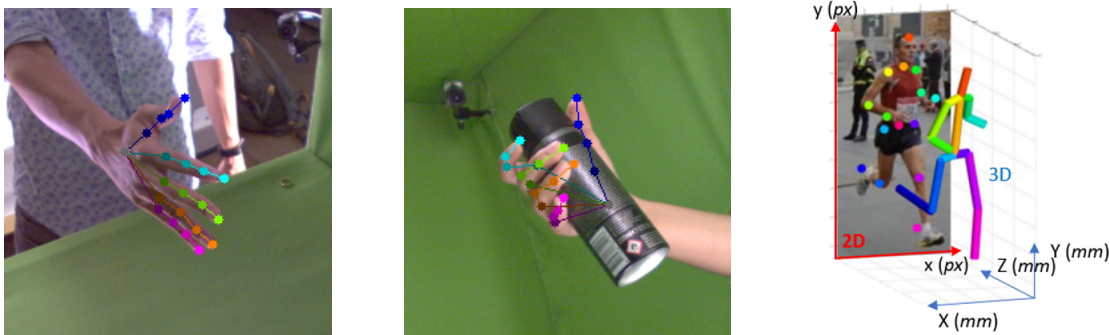


Figure 1.2: Examples of markerless based pose estimation results [Christian Zimmermann and Brox, 2019] [Bartol et al., 2020]

Marker-less solutions are still in development[Avogaro et al., 2023], as more resources are made available to train the models and new architectures are developed and refined. They highlight their advantages over marker based solutions

- **Lower hardware requirements** as less specialized cameras and setups become valid data sources [Muramatsu et al., 2022],

- **Unconstrained movement** as the capture process does not interfere with the subject [Avogaro et al., 2023]
- **Unconstrained environment**, allowing for novel situations and in-the-wild data to be collected [Muramatsu et al., 2022],
- **Larger availability of data** as requirements on subject and environments are lifted [Chatzis et al., 2020]

In conclusion, while marker based solution have existed for decades and have well established themselves in multiple sectors as of today, marker-less approaches are quickly improving, riding on the wave of deep learning developments, and will reshape significantly how many tasks are achieved, allowing for much wider use cases and more natural situations to harness these technologies, while lowering operations costs. Especially in the biomedical and user interface environments, these solutions have unmatched advantages due to the lower complexities and lack of restrictions and costs, allowing for novel applications and products to be developed alongside the evolving technology.

1.2 2D, 3D and 2.5D poses

Regarding pose estimation, another important distinction, now focusing on the output of the algorithms, is between 2D, 3D and 2.5D poses.

2D poses can be described as the localization in the image coordinates of the hand joints, in the input image. Its is a relatively well-defined task for various types of input, and as such sufficiently accurate and fast solutions to compute these exist already in literature. [Hsiao and Chiu, 2021] The problem still inherits issues linked to fundamental texture and feature ambiguity in hands, self and mutual occlusion, low quality inputs and more, but it is generally a better formulated problem with existing solutions, and often leveraged as a starting point in many 3D pose models [Chen et al., 2020b]. It is also of limited practical usage, as only a subset of problems can be solved using 2D poses, such as basic gestures recognition, image hand localization and very limited interaction tasks. More complex tasks such as hand-object interactions, more complex gestures, more advanced gestures, pointing and localization problems all require more information than is provided by a single view 2D pose. [Ahmad et al., 2019]

3D pose is a term that refers to localization of the hand joints in metric 3D space, usually in camera space or a static known reference space. These two formulation of the problem can be resolved to the same task if the camera can be assumed static with reference to the environment.

If that's the case, the two reference systems are rigidly connected by a single rotation-translation operation, which can be easily computed ahead of time, stored and applied when needed, to move between the two systems. If instead the camera is free to move and rotate, the problem grows quickly in complexity, requiring additional solutions, such as reference points or patterns be present or utilizing location algorithms, to calculate the movement of the camera. This task is in itself regarded as an open problem in an unconstrained environment and is usually not considered in most pose estimation research. Instead, the camera is usually considered to be static, and the two problems are resolved to utilizing the camera reference system and moving to a different coordinate system afterwards.

This type of output is the most useful, allowing for full location tracking, recognition of any complex gesture or behavior. It is also a deeply ambiguous problem, especially if the camera system characteristics and hand geometry are completely unknown and subject to change. [Ahmad et al., 2019] Thus, many papers and project use the term 3D pose, but do not refer to full 3D pose data.

2.5D pose is the much more common alternative output. A very influential reference is in [Iqbal et al., 2018], where the term is used to describe an intermediate representation, where hand joints positions are estimated, combining 2D coordinates and a depth estimate relative to a root joint. This approach allows to alleviate some ambiguities and issues linked to scale, camera parameters and geometry, while returning more useful data. As such, multiple papers going forward have shared this approach, producing 2.5D representation of the hand joints, relative to some root node, either a barycenter of the hand or a specific joint. Some solution also have expanded into full relative 3D coordinates, estimating metric distances between joints instead of merely returning 2D coordinates and depth estimates. This kind of output does push the boundaries of 2D poses in terms of usability, more gestures and actions can be recognized, while not raising complexity too much. [Kong and ju Kang, 2021] This format is widely used by hand pose estimation models, and can be elevated to full 3D hand pose data, given a depth estimation of the root joint, and if needed camera parameters.

In our research, we focused on full 3D hand pose data, requiring us to develop some methods further to bridge the gap between their 2.5D and our desired 3D keypoints.

1.3 Multiview, Depth cameras and RGB cameras

Beyond differences in output format, a key factor in hand pose extraction models is the input data. At this time, various models have been trained on a wide range of possible data types,

ranging from single view RGB cameras, to stereo view depth sensing setups to panopticon RGB environments, where cameras surround the subjects. These differences massively impact the architecture, cost, development opportunities, type of data, available datasets, running costs and capabilities of the resulting solution. Understanding the differences is key to an informed choice on what type of solution to choose. [Kaid and Baïna, 2023]

RGB and Depth

RGB cameras are by far the most widely used form of vision sensors. Since their introduction in the 1970s, they have become a cornerstone of both consumer electronics and research equipment. Their development was a cornerstone of the evolution of computer vision, as they allowed to capture reality by mimicking the color perception of the human eye, transforming it into a well-defined digital pattern. Today, thanks to the lowering costs of electronic manufacturing and the burgeoning capabilities of computer vision techniques, both traditional and deep learning based, these cameras are now ubiquitous.

Depth cameras instead are a much more recent development. Restricted before the 2000s to research spaces and military applications, their expansion into commercial and consumer products in the last two decades has brought a new generation of products and solutions using the unique kind of information that these cameras can provide. Depth cameras are a general class of sensors that can produce a depth map of a given view. Unlike a traditional color image, the pixels in this depth map indicate not the color or brightness at a specific image coordinate, but the distance from the sensor to the physical object in front of the camera at that location. This allows the camera to extract much more complete geometric information regarding the geometry of the scene, removing ambiguities.



Figure 1.3: Examples of depth images [Tompson et al.,] [Garcia-Hernando et al., 2018]

The specific functioning principles in depth cameras vary between different models and technologies, from structured light approaches, where a known pattern is projected onto the environment

and the resulting image is used to reconstruct the geometry, to time-of-flight sensors, where the round trip time of a light signal is used. These different underlying technologies also define what exact types of ambiguities and issues might arise in the usage of the sensors, such as texture ambiguities in stereo solutions or sensitivity to specific materials. Due to their more recent development, these cameras are not as mature as regular RGB cameras, resulting in both reduced performance metrics, such as resolution and framerate, and increased costs and complexity. The overall software and technical support for these sensors is also still limited, compared to the wide support for RGB data.

In more details, RGB sensors usually show some advantages over their depth counterparts, largely due to the maturity of the technology:

- **Lower costs**
- **Higher resolution**
- **Higher framerates**
- **Longer range** as most depth sensing technologies have limited range
- **Larger software support** in both control of the hardware, availability of processing libraries, storage and computation support

In contrast, depth sensors can only sport their unique sensing ability as their advantage, but this should not be taken lightly. The exact capabilities depend on the specific hardware in use, but depth sensors produce unique information about the true structure of a scene which can be invaluable to pose estimation tasks, which can be complex to capture using purely RGB sensors. These instead need to rely on depth clues, such as parallax, previously known information about object size or multiple points of view to extract equivalent information. In contrast, their lack of visual information means they might miss on other clues, such as texture or lighting information, which is invisible to most depth sensing technologies. [Li et al., 2019]

Combination cameras, usually referred to as RGB+D or RGBD, are available and, when used correctly, the two data streams can be combined to have an accurate depth map with matching color data in real time. This allows the advantages and properties of each sensor to fill the gaps of the other. These sensors do come at a much higher price, maintain the complex setup needed for depth cameras, and are generally much less used in both data capture and consumer products, while also displaying quite high prices. [Li et al., 2019]

We can see the importance of both kinds of data collection techniques, and their limitations, reflected in the existing datasets, both of general pose data and hand pose data. Depth data is

present, but the datasets are often much smaller in scale and variety, while also being mostly used in research. Larger and more commercially used datasets instead focus on RGB sensors, as they are already widely deployed and the resulting technologies would have much wider applications. Nonetheless, depth based hand pose extraction models are actively researched and developed, thanks in part to the falling prices of these technologies, and show incredible results and accuracy, demonstrating the potential of the unique 3D information they can harness. [Topham et al., 2022] [Doosti, 2019] [Li et al., 2019]

Single View and Multi View

Another relevant feature in possible input setups is the distinction between single and multi view systems.

Single view systems are the simplest setups, with a single camera capturing a unique point of view. This is by far the most common video setup in both potential research environments and practical consumer applications, Requiring only a single smartphone camera or webcam source, this can be deployed with no additional requirements on most platforms, and would enable usage on previously captured images and videos.

Single view systems have a series of advantages, mostly relating to their cost:

- **Very low setup cost** with no specialized hardware needs
- **Reduced setup complexity** removing calibration needs
- **Well developed computational frameworks** for both storage and processing
- **No major limitations on usage**, allowing for novel applications including processing past data
- **Large available training data**
- **Portability** due to their reduced complexity, allowing for much wider range of data and environments

[Avola et al., 2022] [Jiang et al., 2023]

Multiview systems, in contrast, describe the case where multiple cameras provide different points of view at the same time. This kind of solution requires a more complex setup, more advanced knowledge of both the camera system and of computer vision techniques, and is not commonly available side of specialized environments, bust also allows for much more powerful solutions and techniques. [Iqbal et al., 2018]

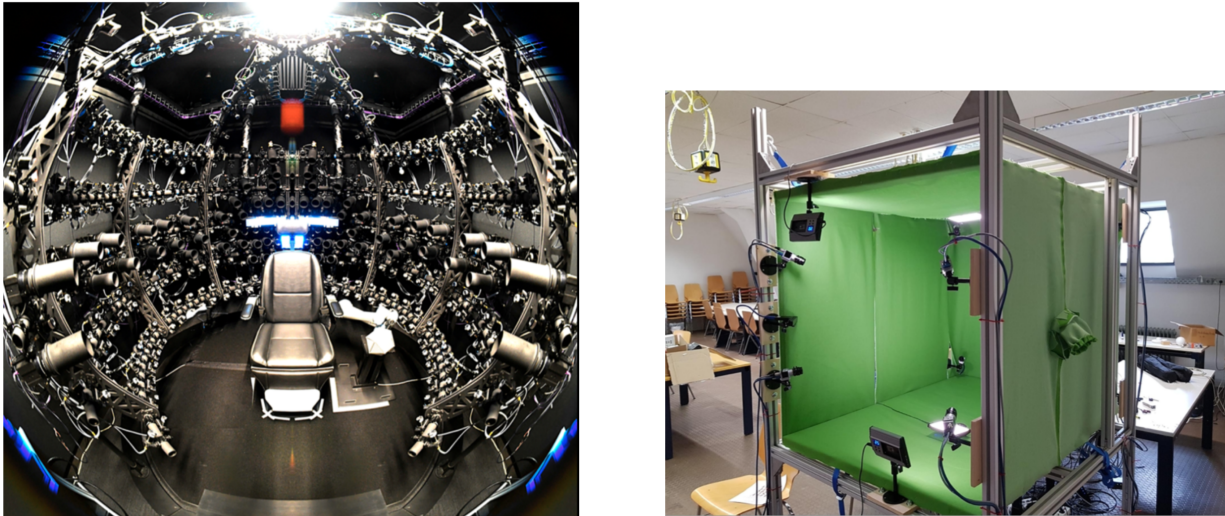


Figure 1.4: Examples of multi view setups [Moon et al., 2020] [Christian Zimmermann and Brox, 2019]

The clearest example of a multiview system is the stereoscopic vision common to most animals. Specifically, when the eyes share a large section of their field of view, such as in humans, each eye produces a different image based on the difference in position. The brain can then combine the two images, alongside other clues such as parallax motion, temporal information, motion clues and the known relative positions of the eyes, to reconstruct a 3D view of the environment, utilizing only RGB perception. [Meyer and Wilson, 1991]

This setup can be reproduced using two cameras kept at a fixed distance, looking in the same area, in what is usually called stereo vision. Like in human perception, the images can be combined to identify disparities caused by the differing positions, which in turn can be used to construct a depth map of the shared field of view. Thus, these setups can also produce accurate depth estimations of the objects observed, while relying only on visual clues. [Richard Hartley, 2003]

Multiview solutions can also instead capture the scene at multiple different angles and positions, collecting a much more complete view of the actions, thus reducing issues with occlusion and lack of knowledge of the geometry of the scene. These designs are usually referred as *panopticons*, referring to their ability to observe a scene from all points of view. They can capture a wealth of data, allowing to bypass many issues, such as occlusions and shape ambiguities. These setups are the most complex and expensive to build, requiring that the cameras are maintained at fixed positions relative to each other, accurate calibration and synchronization between all cameras, reducing flexibility and portability immensely. [Gomez-Donoso et al., 2017] Such solutions are used mostly in high-end motion capture settings or in dataset generation environments, where

all views are combined to create the most accurate data possible. [Moon et al., 2020]

These advantages exist in contrast with the advantages afforded by multiview systems, both stereo and *panopticon* designs:

- **Availability of depth clues** from reconstruction
- **Reduction of occlusion problems**, especially in systems beyond stereo vision
- **Enhanced accuracy** due to the much wider availability of data during computation
- **Enhanced reliability** due to the reduced complexity and ambiguities of the system

As discussed, these solutions require more complex setups than single camera approaches, as the two cameras need to be in known and fixed relative positions. These parameters, usually called *extrinsic parameters*, can be computed in an initial calibration phase. The system also requires accurate synchronization of the captured frames, ensuring the images are captured at the same instant. These requirements raise the complexity of setup and go beyond what many non-technical users be capable of, or what many normal camera systems might be capable of collecting accurately. [Richard Hartley, 2003]

To alleviate some issues, commercial products that encapsulate the entire task into a single camera system, using well calibrated sensors and a ready-made software stack are available, but are not common apart from research and development spaces, and are much more expensive. These systems are nonetheless quite accurate, and many depth camera systems rely on stereoscopic vision. [Doosti, 2019]

As such, while researchers are actively exploring both facets of the topic, the focus of research seems to have been moving towards single view models, for their flexibility and widely applicable usage, where the task cannot be managed by combining multiple 2D estimates with the known geometry of the system [Richard Hartley, 2003]. Multiview models remain extremely relevant in both industrial and medical applications, due to their higher accuracy and reliability, and where setup cost and complexity are less relevant. The evolving nature of stereo models and their ability to produce both RGB and depth data in a compact package might be an extremely relevant avenue in the future, as these sensors become more common and their software libraries grow to maturity, allowing for large scale usage.

1.4 Hand pose specific problems

Many of these issues and problems discussed here are not unique to the specific task of hand pose estimation, instead deriving from the wide nature of the pose estimation field in general. These highlight the general complexity of the field and the wide range of research that encompasses the topic, but each specific task has its own specialized problems and issues, added on top of these. Hand pose estimation raises new issue linked to the unique geometries of the hands, their everyday usage and the tasks that researchers and teams focus on.

Occlusion is one of the most relevant and discussed problems for hand tracking and pose estimation. [Moon et al., 2020] It references the fact that hands can often occlude a part of themselves to a camera (self-occlusion), or, if more than one hand is present, these can cover up each other (mutual occlusion). This problem raises major issues with many traditional computer vision algorithms, as relevant image features and tracking points might be lost or become partially invisible. Some solutions, such as the previously discussed panopticon setups, can partially alleviate the problem, as multiple cameras might be able to keep a global view of all relevant hand parts, but this is not a certainty.

Object interaction is a closely related topic and source of problems, as hands usually are engaged in activities where they interact with a specific object. This is generally a source of problems, such as more occlusion of the hand, but in some sections of research it is instead the focus of the work, attempting to understand the motions of the hand relative to the object, the contact areas, motion and gestures, assigning labels to actions and more. [Hampali et al., 2021]

Texture ambiguities reference the fact that the hands have little unique features to allow for feature based techniques to track movement. No well-defined dark areas or color sections are naturally present in hand, making the task much harder, especially for solution that do not rely on deep learning techniques, who show themselves to be more robust in tracking global features.

Strong variance, as hand shapes, sizes, color, features, details and even overall shape can vary greatly. Simply consider features like skin color, decorations or hand sizes and consider how these might impact what a model extracts based solely on visual information. This highlights also the need for wider cultural collaboration and participation in the field, to ensure models and solutions do not have ethnic or cultural biases in their results, due to incompleteness of training data in covering the potential variety.

Complex articulation referring to the very advanced articulation abilities of hands, which limit solutions based on silhouettes or shape, as it can't be relied upon. It is also connected with the aforementioned occlusion issue, as the articulation is often partially cause of the occlusion.

It is key to understand that this complexity is not a problem to be circumvented, but it is core to the relevancy of the produced data. The complexity of these poses is linked to their key role in human activity and interaction, from sign language translation to gesture recognition, to VR applications, these actions rely on the highly complex and often subtle movements the human hand is capable of. [Moon et al., 2020]

This is not an exhaustive list of potential issues or more uncommon, such as

- **Small apparent size**, caused when the camera is away from the hand in question and of lower resolution
- **Relation to full body pose data**, as the tasks are often studied separately, but many applications rely on combined information
- **Motion blur**, caused by fast movements and lower end camera systems

Some of these issues can be solved or aided in different conditions and using specialized technologies. As discussed previously, multiview models, especially panopticon designs, can alleviate occlusions. Depth cameras are indifferent to texture and color, reducing issues due to the variance in those fields. Markers are used in higher precision tasks as they alleviate issues on lack of features and ambiguities in rotation. But these solutions come at technical and logistic costs, and in unconstrained environment and applications, are often not valid solutions.

Thus, we highlight the additional complexity of the task, moving beyond the discussed issues and choices necessary in designing a pose estimation project, and exploring the issues that are specific to hand poses. This also highlights why, even as in recent years the field has shown such immense growth with the rise of deep learning models that have been shown capable of tackling many complex computer vision tasks, the topic remains one of compromises, complexity and active research.

1.5 Conclusions

We can observe a wide range of research avenues, combining the various paths and possibilities discussed before, and requiring us to consider multiple factors:

- **Use case** if for commercial, industrial or medical usage
- **Requirements on flexibility and accuracy**
- **Acceptable hardware and setup costs**

- **End user comfort**, especially for marker based solutions
- **Flexibility and portability**, often at odds with enhanced accuracy

These choices inform us on the models and technologies we will consider and explore. They also inform what issues and problems we'll have to tackle. For our applications, it was chosen to approach what is perhaps the most limiting case: a single view RGB camera, while also trying to achieve approximately real time pose estimation at the best accuracy possible. The actions and positions to be tracked did allow to avoid some issues, such as mutual and object occlusion, as we'll focus on tracking mostly a single hand with little object interaction, in without strong variance in hand features, lighting or fast movement. Our goal was not just to study the available models, but to utilize them directly to produce the data we needed and to create a small, functional product we could use further, encapsulating all the needed libraries, models and algorithms inside a well-defined project.

Thus, we focus now on exploring the models available in the current state of the art for Single view RGB hand pose estimation algorithms for 3D and 2.5D hand pose data, understanding their architectures, designs and abilities, testing their ability to be used in our task

Chapter 2

State of the Art

While many more models have been developed on the topic, only a small subsection has made their model available alongside the source code required to reconstruct the model. Most researches only give overviews of their training process, focusing perhaps on specific choices made during it, often without releasing the full source code of the model, or a working pretrained model. While not an absolute impedance reproduction and research, this does dramatically raise the barrier to iterate over the design and architectures, as large sections of work must be repeated by future researchers to reach the same results, before any new work can be done to improve on it. This was the major reason for many papers to be excluded from our selection, as the task of fully understanding the training process conducted by the original researchers and reproducing it entirely was deemed beyond the scope of the work. Instead, priority was given to models that released source code, trained models and that detailed the required libraries and dependencies for the technology.

Nonetheless, redeployment remains an issue, as most models are developed not as streamlines commercial applications, but as research projects. This leads to major differences in software design and architecture, as development is focused much more on fast iteration, optimization, evaluation and design flexibility, to allow researches and developers maximum freedom and result in the best performances. The resulting software is thus structured in a much more idiosyncratic way, reflective of the needs of the project at that moment, and less focused on longer term objectives such as code maintainability or ease of reproduction. Documentation is often limited, as are details regarding specific libraries used and their properties, how data flows through the model and how results are produced and made available.

Because of this, each model was first analyzed for its architecture and reported results, then

shifting our focus on redeploying the model. To aid in the process, allowing to isolate each models contexts and dependencies, we made large scale usage of containers though Docker.

Docker is a software solution that leverages unique functionalities of the Linux kernel to fully isolate applications from the main OS at low performance overhead. This allows programs to be executed in entirely separate operating systems, with differing libraries, versions, dependencies and files, while sharing the host machine’s computational hardware, such as CPU cores, RAM and hardware accelerators, such as GPUs. [Doc, 2024]

Containerization is a virtualization technique widely used in both cloud and non-cloud environments to separate processes with little performance loss. This relies on sharing of resources at the kernel level, where hardware access is managed and shared between host and container, while maintaining separation at OS level, thus ensuring autonomy and safety. [Scheepers, 2014]

This solution allows to quickly iterate on different models and architectures while sidestepping the issues brought by differing dependencies or versions required, without the sacrifices to performance usually mandated by traditional Virtual Machines (VMs).

The models considered in this chapter all rely on deep neural networks for their functioning, and leverage a wide range of architectures and datasets. Reflective of the fast evolution and expansion of the field of Computer Vision, they have been developed by a wide range of universities and teams from around the globe, with differing objectives and tasks in mind. They all share the required input modality, so they all produce 2.5D or 3D hand pose data from a single view RGB input.

The models that were studied are:

1. **InterNet**
2. **Pose Equivariant Contrastive Learning**
3. **Mediapipe**
4. **Anchor to Joint - Transformer**
5. **Hierarchical Temporal Transformer**
6. **Hand De-occlusion and Removal**

2.1 InterNet

Released at the European Convention on Computer Vision 2020, InterNet and InterHand 2.6M are respectively a pose estimation network and a hand pose dataset developed by Korean researchers in collaboration with Meta Facebook Reality Labs. The size and complexity of the work have made it a deeply impactful step in the field, as the private-public partnership has produced a 2.6 million frame dataset, which has since become integral in training and evaluation of other hand pose models.

The network uses a straightforward approach based on transfer learning. A network previously trained on image data, in this case a ResNet, is used to extract high level image features, produced by the top convolutional layers, which do not require retraining. The final fully connected stages are instead removed and substituted with new custom ones. These new layers are trained over the dataset, developing associations between the image features that ResNet extracts and the desired results. [Moon et al., 2020]

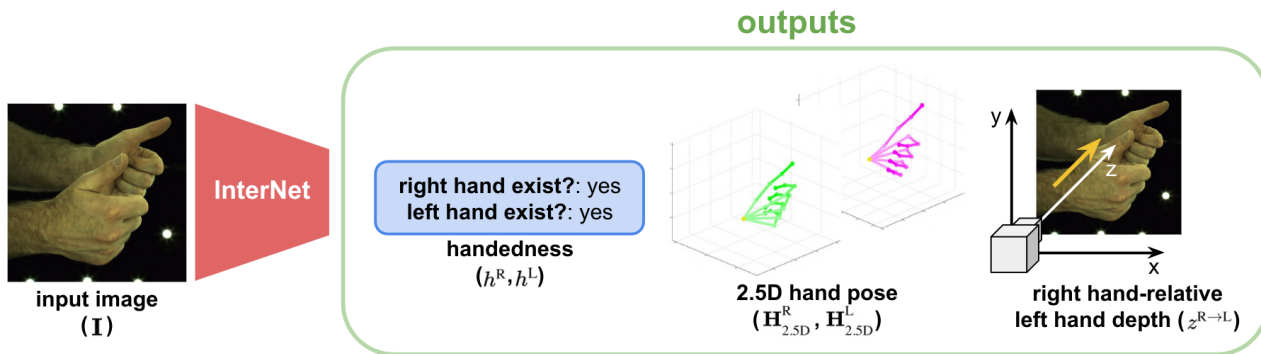


Figure 2.1: Architecture of the InterNet model [Moon et al., 2020]

This approach is a common technique employed in computer vision tasks, as it allows to leverage the feature extraction capabilities of a network trained on a large and varied dataset, like ResNet, and modify only the latter sections, where the model develops more task specific connections. This allows to lower computational and data requirements for training, while taking advantage of existing trained models to obtain high quality results, especially when the difference between the original task and the new target task is small, and we can expect the network to require only small changes to adapt to the new environment and objectives. [Szeliski,]

The aims of the InterNet project are multiple [Moon et al., 2020]:

- **Studying the impacts of interacting hand data in training**, especially with the

objective of overcoming common limitations of existing models, usually either limited to a single hand pose output, or which showed significant problems with interacting hand data. The hope is that by including complex hand interacting data in the training phase, such limitations could be overcome.

- **Studying how to better estimate multiple hand poses from a single RGB image**, by employing a more flexible architecture which can output pose data for more than one hand and can return confidence values of the presence of such hand and by estimating not direct 3D joint coordinates but using an intermediate 3D heat map technique.
- **Providing a baseline and a new dataset to expand the research space**, making high quality data available to researchers.

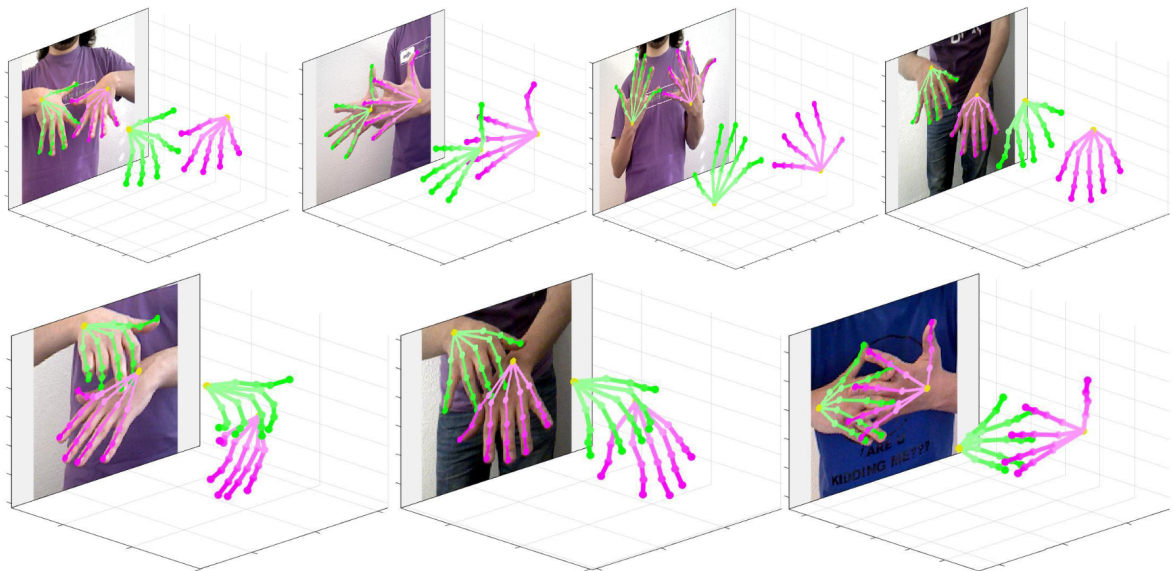


Figure 2.2: Examples of InterNet results [Moon et al., 2020]

For this, the network was the first candidate for redeployment and evaluation. This was achieved using the MMPose framework, which solved some dependency management issues and data management, allowing for quicker testing and evaluation, utilizing not only training data but our custom data streams such as webcam feeds and videos. MMPose is a PyTorch based toolkit used for various forms of pose estimation, including hands, 2D body, 3D body, face and animal pose estimation tasks. It provides a simplified experience to developers and researchers, such as a common API to access various models, evaluate results, access and process datasets, train and optimize algorithms. At the time of usage and writing, its support for 3D hand pose algorithms

was limited to InterNet. [Contributors, 2020]

The results showed some qualitative issues with the network's performance. While the accuracy of the model was quite good, there were quite a few failure modes, such as misdetection of hand location, missing hand detections, joint pose errors, pose flickering and other issues. Some of these problems were likely caused by the lack of bounding boxes around the hand sections in the frames, which are instead included in many training and evaluation datasets in multiple cases. This was highlighted as a potentially problematic addition to the pipeline, as it would require additional preprocessing of the input data with a location and detection model, raising computational costs and complexity.

This was a major concern, as what seemed to be the biggest limitation for the project was in the computational performance. Using the provided hardware, equipped with an Nvidia GTX 1080 GPU, the algorithm was not capable of real time estimation, averaging instead around 4-8 frames a second. This would pose limits the potential usage of such a model to either truly high-end machines or off-line usage, not real time scenarios.

Correcting these issues was not considered feasible or in scope of the project, so the model was not considered usable for the project.

2.2 PeCLR

PeCLR, or *Pose Equivariant Contrastive Learning*, is the name of the model developed by the Department of Computer Science, ETH Zurich to expand contrastive learning and self-supervised learning techniques to 3D geometric problems, specifically hand pose estimation. [Spurr et al., 2022]

Contrastive learning is a growing field in self-supervised approaches where unlabeled data is used to enhance detection by training the model to maximize similarity for similar data points and minimize it for different points. This is done by taking an anchor input, deriving a positive and negative input from this one and training the model to classify the positive input similarly to the anchor, and the negative input differently. An anchor input is simply an input training sample used as a reference point, to which the positive or negative are compared to. An example is in 2.3.

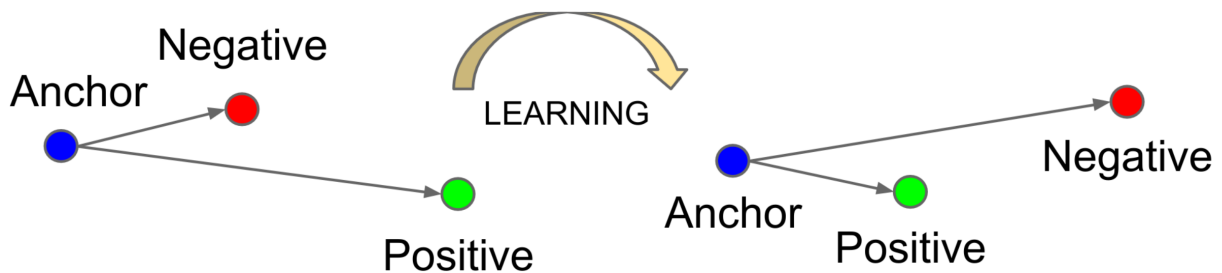


Figure 2.3: Example of contrastive learning: the distance represent similarity scores [Schroff et al., 2015]

In this case, given an unlabeled hand image used as an anchor input, a positive input could be the same image with data augmentation techniques, such as affine transformations or color drops, while a negative input might be an image from a completely different source. The network is then trained to associate the augmented image similarly to the original hand, and differently from the different hand. This should train the model to better comprehend the information, becoming for example more resistant to noise or other sources of disturbance, or more capable of recognizing similar hand poses correctly, while also developing a clearer distinction against different poses.

This technique has shown great results in other computer vision fields, and has major advantages in topics such as this due to the low volume of high quality data, the complexity of collecting high precision 3D hand pose data in unconstrained environments, the high cost for complex setups

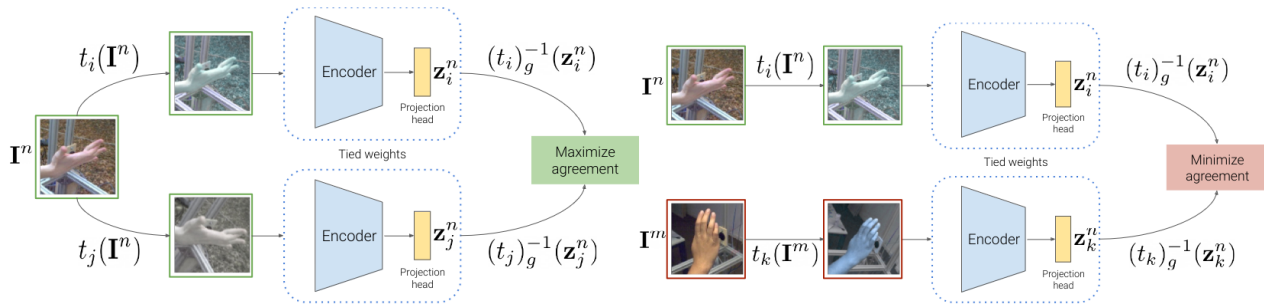


Figure 2.4: Overview of the method used during training of PeCLR. $t \in T$ identifies augmentations, while z^n identifies the model’s output. These augmentations are inverted on the outputs to realign them and ensure equivariance under geometric transformations [Spurr et al., 2022]

such as sensors or multiview systems and human or machine annotations for such data. With this approach instead, lower quality labeled data such as simple 2D labels can be used to enhance pose detection at a lower cost. [Chen et al., 2020a]

Thus, the core development of this work rests upon the generalization of this technique to a 3D problem, while maintaining equivariance for the results under simple affine transformations. This step is done as a pretraining of a ResNet model, which is then trained end-to-end on a traditionally labeled dataset. The results show improvement compared to normal ResNet based learning, approaching more specialized architectures.

During redeployment, the model exhibited decent performance and faster results than other solutions, the maturity of the technique and the resulting model was not sufficient for our purposes. Missing frames and errors, instability and jitter of the model and more technical issues linked to the early stage of the technology and surrounding technical frameworks made it unsuitable for our purposes.

While this showed great potential as a technique key to leveraging smaller datasets to extract the most information, and lowering barriers to entry for entities without the budget to construct massive custom datasets, the early stage of the technology, the lack of refinement of the model architecture and the limited amount of data likely constrained the current state of the algorithm. It was thus discarded for further usage in our case.

2.3 Mediapipe

Mediapipe is a project developed by Google, spanning a research and development framework, a set of trained models and solutions for on-device machine learning tasks. Leveraging Google’s TensorFlow Lite library, the framework and resulting models are explicitly designed and constructed to achieve real time performance on mobile devices GPUs and CPUs, and as such has a much simpler architecture and lower inference times. To reconstruct hand pose information, Mediapipe provides the Hand Landmarker model, which produces 2.5D and 2D hand keypoints from RGB images.

The model is structured as a two stage system:

BlazePalm comprises the first step, a compact Feature Pyramid Network that performs palm detection and localization. This is used to identify and locate the hand in the image, simplifying the task from a full hand detection and providing a clearer training target in terms of aspect ratio and keypoints. This component is utilized only when hand tracking is lost or at first execution, reducing computational load, especially on mobile devices

Hand Landmark Model is then utilized to identify the hand pose, cropped utilizing the previously generated bounding box. The model produces 2D and 2.5D landmarks based on the image. This component also has a role in maintaining tracking over the detected hand locations, such that the palm detection does not need to be run continuously.

While this, alongside other components of Google’s Mediapipe models are regarded as open source, as the models and the general design and functioning of the library are available to users, these models are not reproducible as training data and details of the individual module’s architecture are not shared. The paper focuses on the relatively small size of the dataset and the split between synthetic and real world data, specifically 16k real and around 100k synthetic images. This highlights how well selected data sources and have large impacts on training quality and how a combination of synthetic and real data can result in high quality models. [Zhang et al., 2020]

The model is also accompanied by documentation regarding its ability to correctly identify hands from various skin tones and genders, alongside discussions of fairness scores, possible intended uses and out-of-scope cases, and ethical information regarding training data. [med, 2021]

Redeployment in this case was relatively straightforward, as Mediapipe is the only candidate that is not part of a research effort, but a true product designed for developers and companies, and as such has extensive documentation, a large user base, examples and a well-defined API for

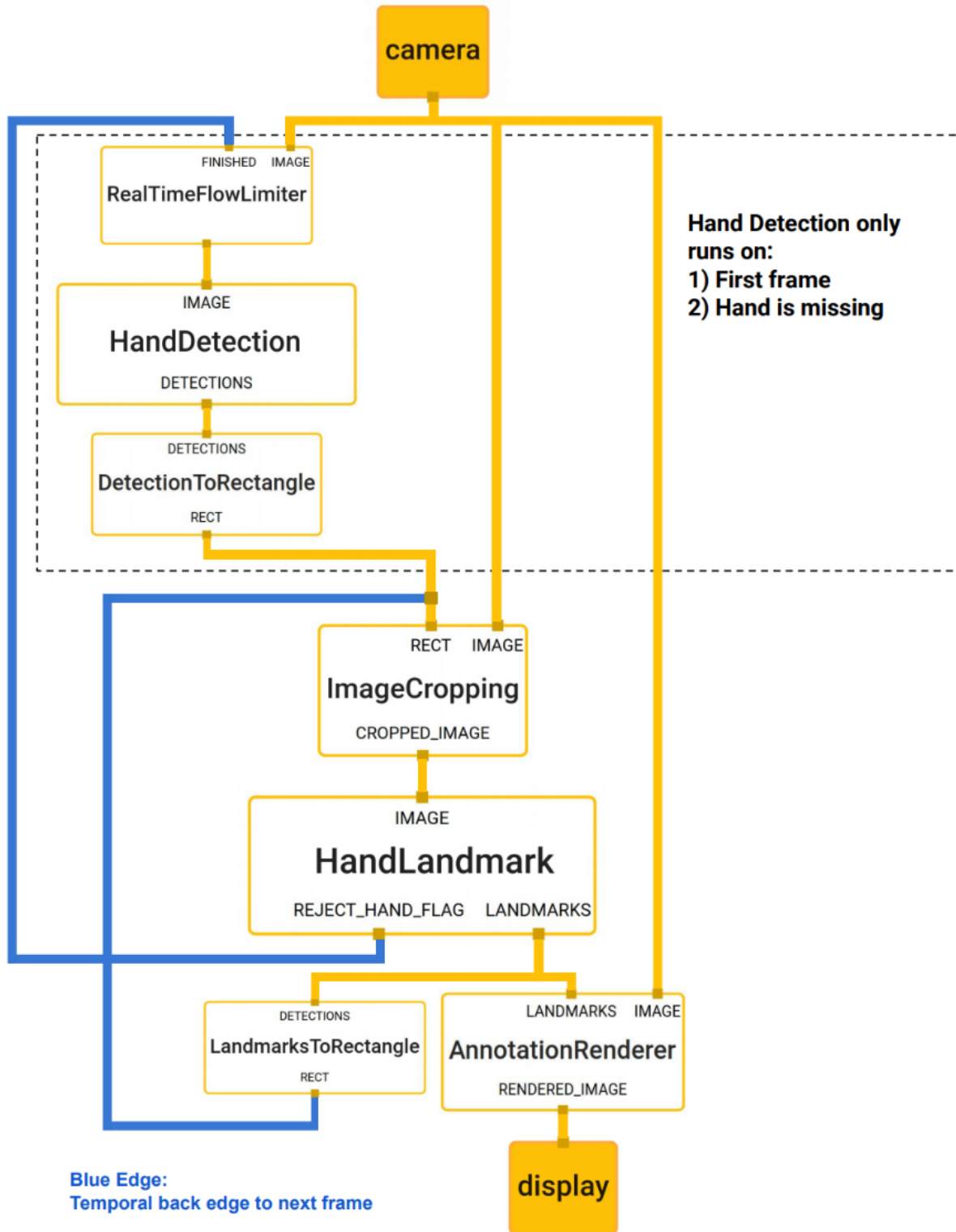


Figure 2.5: Overview of the architecture Mediapipe Hands, constructed using the Mediapipe Framework [Zhang et al., 2020]

usage. The models are also uniquely licensed under the Apache model, making them compatible with commercial use cases. Our evaluation showed good accuracy for pose estimation, especially on 2D keypoints, very fast estimation, achieving real time detection and tracking on a desktop CPU, and little to no issues in deployment. As such, the model was deemed a valid candidate for the project.

2.4 A2J-Transformer

A2J-Transformer or *Anchor to Joint Transformer*, is an RGB-based derivation of the Depth-based State-of-the-art A2J model. It was developed as a joint project of Singaporean, Chinese and AliBaba researchers.

The original A2J paper introduced a novel approach in depth based hand pose estimation: a main ResNet based trunk processes the depth data to extract high level features. These features are then passed to three different 2D CNN based branches, each with a separate objective:

- **Identify the relevant region** of the image
- **Detect local features** inside this section
- **Reconstruct the 3D position** in relation to their joints

Combined, this model achieved state-of-the-art depth estimation results on depth image data, showed how the concept could be generalized to a wider set of pose estimation tasks and resulted in much faster results compared to other 3D CNN based models. [Xiong et al., 2019]

Because of this, the model became the basis for the A2J-Transformer, which had to overcome the differences between depth and RGB data and the inherent ambiguities and uncertainties that arise. RGB data is much richer in details and 2D information, but lacks global depth information about each component, requiring additional work to solve this knowledge gap.

The researchers thus evolve the architecture by first harnessing the deeper visual information, developing a complex transformer based architecture:

- **A novel feature extractor architecture**, using a ResNet model combined with a pyramid feature extractor module.
- **Usage of 3D anchors compared to the 2D anchors**, as the RGB image lacks any, depth information which must instead be reconstructed by the network.
- **Removal of the 3 CNN branches**, substituted by a single transformer, where the encoder section processes the image features, with tokens combined with the 3D anchors during decoding.
- **Offset estimation branches**, processing the decoded token in on two parallel networks, each trained for in-plane or depth estimation tasks.

The results from the final offset estimation branches are combined with the base 3D anchors to produce the joint positions. See 2.6. [Jiang et al., 2023]

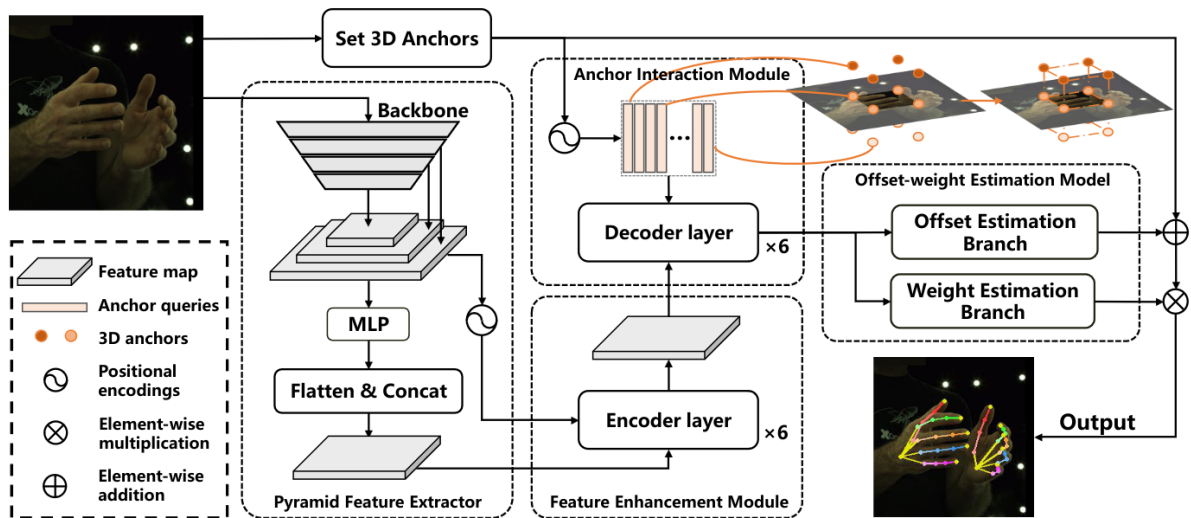


Figure 2.6: Overview of the internal architecture of the A2J-Transformer model [Jiang et al., 2023]

The reported results of the model are notable, with high precision on a range of datasets, including InterHand2.6M, and reports comparable results to other state-of-the-art designs, both in joint estimation precision and inference time.

In our attempt at redeployment, multiple technical issues were encountered, likely as a result of the high complexity of the architecture, and the limited support and material available on the construction and execution such a novel design. As a result, the model could not be redeployed, neither on custom input data nor on the original training data, resulting in the model being removed from the pool of candidates.

2.5 HTT

Developed by a combination of Hong Kong, Texas and Microsoft researchers, HTT or *Hierarchical Temporal Transformer* is a novel design that aims to harness both the image processing capabilities of ResNet models and temporal information and self attention systems of Transformer architectures.

The usage of temporal information could help alleviate many issues that arise in hand pose estimation, such as occlusion, visual ambiguities and suboptimal lighting conditions. It could also help estimate more complex movements or extract higher accuracy data, especially on dynamic pose information such as joint speed and acceleration. Usage of transformer architectures over other temporal networks such as Long Short Term Memory, or LSTMs, has shown improvements, like higher precision and overall performance, in other sequence tasks, as the multi-head self-attention mechanisms can better infer relationships between tokens inside the temporal window. It does come however at higher computational costs often and at higher training requirements, due to the raised complexity of the models. [Prince, 2023]

This design is constructed by first extracting high level image features from each frame, combining these using a sliding window approach, extracting hand pose while leveraging the wider temporal window. More specifically, a pretrained ResNet 18 model is used to extract features from each individual frame, and these output keypoints are reshaped into input tokens for the temporal transformer, called Pose Block, which estimates pose information for each frame.

It should be noted that in this paper, this approach is also extended to a gesture and action recognition task, using a second transformer layer that combines larger temporal window pose data into gesture tokens, but this section was beyond our interest and was ignored. [Wen et al., 2023]

Of note is also the difference in training of this model, which was designed around egocentric videos and trained based on these datasets, in contrast with the other models discussed here. Egocentric data is recorded not from an external point of view, but from the point of view of the hands' owner. This could impact the performance of the model outside egocentric tasks, as the unique point of view reduces somewhat appearance variations and potential poses, and limits the task, connecting the hand pose and camera position to a tighter relationship.

The reported results show that state-of-the-art accuracy can be reached on egocentric datasets, compared to similar models, although this comparison might not hold with models trained on other pose estimation tasks or other data sources. [Wen et al., 2023]

In our redeployment, after some technical difficulties, the model was correctly redeployed, but

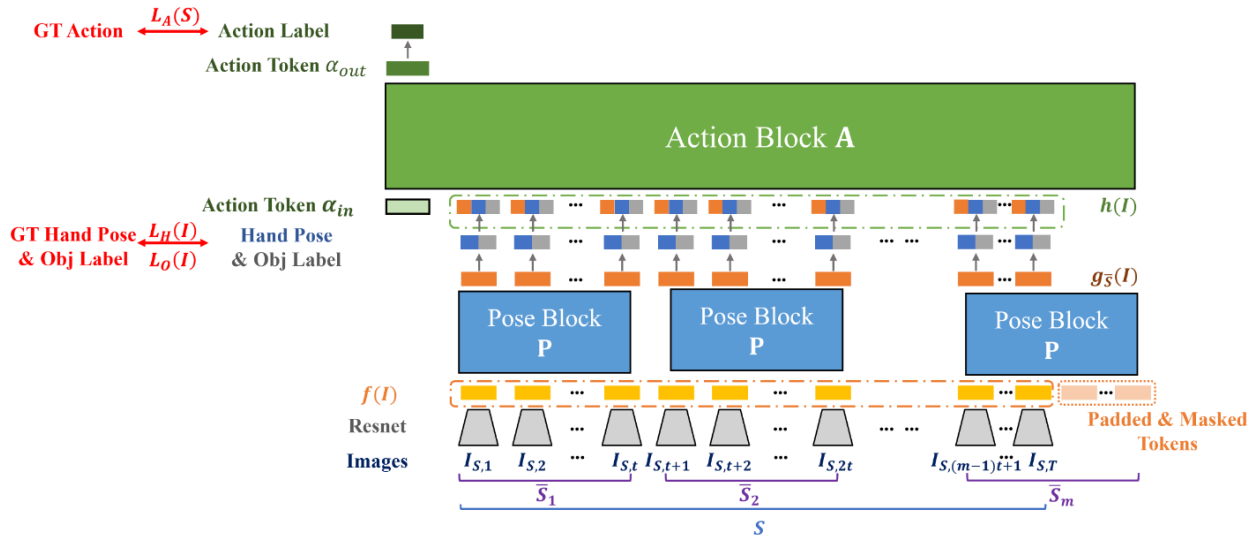


Figure 2.7: Overview of the internal architecture of the HTT model [Wen et al., 2023]

could not be applied to a live video stream. Instead, it was qualitatively evaluated on a recorded clip. The model showed many limitations, as it failed to locate and correctly track non-egocentric hand poses, showing jittery and noisy results. We expect the issues to be linked at least in part due to the drift between the training environment and our own, and perhaps also due to the early stages of development of the approach. Nonetheless, retraining on non-egocentric datasets and investigating these issues further was considered out of scope, and as such the model was deemed unsuitable.

2.6 HDR

Developed by Chinese, Australian and Hong Kong researchers, HDR or *Hand De-occlusion and Removal* is a novel approach to estimation of interacting hand poses from RGB images. The key innovation is in a different approach to interacting hand pose estimation, where, instead of treating the task as a different one from traditional single hand pose estimation, the researchers want to decompose the problem into two individual hand pose estimations for each hand. This would allow leveraging existing single hand estimation techniques.

This approach requires solving issues due to self and mutual occlusions of the hand, and in general noise due to overlapping hands, alongside accurate detection of the hands. To achieve this, the work proposes a three stage algorithm for hand pose estimation:

- **Hand Amodal Segmentation Module (HASM)**, tasked with hand detection and segmentation in the RGB image, for both hands
- **Hand De-occlusion and Removal Module (HDRM)**, tasked with de-occlusion, or reconstruction of missing sections of the hand, alongside removal of irrelevant elements in the image
- **Single Hand Pose Estimator (SHPE)**, tasked with extracting hand pose data from the cleaned hand images

As such, this modular design means that the SHPE can be a pre-existing single hand pose estimation method and reducing the need for further training, allowing researchers to focus only on the other two modules. [Meng et al., 2022]

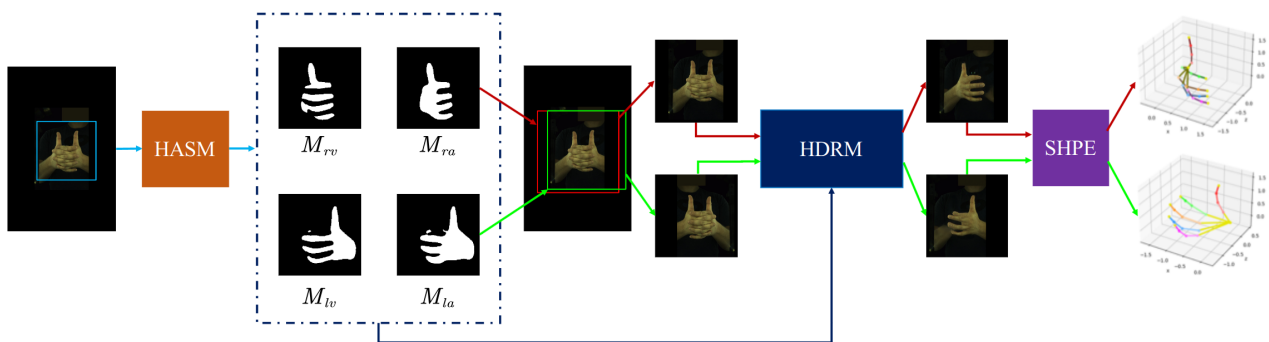


Figure 2.8: Overview of the internal architecture of the HDR model [Meng et al., 2022]

But from this a new issue arises, specifically the absence of existing amodal training datasets for the HASM module, which was solved directly by the researchers, by constructing a custom

synthetic dataset as needed, called Amodal InterHand dataset, or *AIH*, as a derivation of the InterHand dataset [Moon et al., 2020]. This novel training data is then used in retraining an existing instance segmentation model, specifically SegFormer, after conforming its output format to the needs of the downstream module. [Meng et al., 2022]

The architecture of the HDRM is wholly novel, combining transformer blocks inside an existing autoencoder model derived from previous image reconstruction research. This architecture allows for complex attention-based reconstruction of missing pixels during decoding and encoding of the image, allowing the model to combine the provided masks and input images and producing the reconstructed image, with the occluding hand removed.

The evaluation on the InterHand2.6M shows results that surpass the baseline model InterNet, highlighting the promising potential of this architecture on the task. [Meng et al., 2022]

In redeployment, multiple technical hurdles were found, both in reproducing the complex modular design and collecting all the needed libraries and models. Other issues were also encountered in adapting the model to new data sources. The results showed some limitations, both in the accuracy of the hand reconstruction, the estimated pose accuracy, and also the hand detection stage, both in terms of speed and accuracy. Another source of difficulties was the need for the model to be provided a bounding box ground truth for the input hands, even during evaluation, as the hand detection module was not capable of global detection and segmentation. These issues led to determining this model unsuitable for the project

2.7 Conclusions

From these results, we can identify some important points regarding the state of research in hand pose extraction.

Thanks to continued developments in machine learning techniques, computing capabilities, software frameworks, data collection solutions, shared technical know how and growing commercial possibilities, the entire field is in a state of expansion and sustained growth. Papers describing new solutions and approaches are published at a growing rate, showcasing a wide range of solutions. New datasets are released often, pushing the boundaries of research and expanding the possible topics to cover, such as hand-object interactions, interacting hand problems, gesture recognition and more.

This almost feverish development environment has a downside, as more and more models and techniques and solutions are proposed, reproduction studies become rare, comparisons and reviews where redeployment is a concern are less common and less research focuses on incremental results and possible combinations of differing solutions. As such, different technologies with widely different maturity states and technical contexts are compared. This is enhanced when differing underlying datasets are also used, or novel datasets are introduced to fulfill some new requirements. These differences alter the context of each research work, making for example comparing older models becomes difficult, as they were trained on completely different datasets, with different metrics and objectives.

Nonetheless, the activity leads to newer and better solutions, employing new techniques from all facets of Machine Learning research and showcasing the potential of many of these new methods, such as transformers or autoencoder or temporal models, in the context of pose extraction and 3D lifting for RGB data.

This activity is fed and in turn feeds the tight connection between private and commercial entities such as Meta and Google, and public organizations such as universities. These partnerships clearly lead to increased output. Much larger budgets and resources can be allocated by private entities quickly, allowing researchers to push the envelope on the topic, while exploring new and innovative solutions.

These partnerships do have some more complex effects, such as a diminished transparency as companies might be incentivized to not share their best models and solutions, their datasets or too many details about the underlying architectures, such as the case for Google's Mediapipe. Similar results are even more evident in spaces like LLM development and GenAI models, where some companies only allow for API access to models, while other public trained systems, but do

not share the training and architectural details.

Another result which might have connection to the commercial incentives of these partnerships, is the licensing surrounding many datasets and even some models and architectures, which are limited to research usage or non-commercial usage only. This was a complex topic to untangle in the project, as it shaped what future usages of the resulting solution might be available and which might be blocked.

An issue that was noticed was that, while there was a strong focus on metrics relating to accuracy and joint precision, only a few papers discuss the time performance of the model and compared it to others. Reporting inference times has not become the norm in the field, both as some models are not designed or built to perform at real time or similar rates, but also as models are often trained and tested on highly expensive and powerful GPU setups, which are hard to compare between generations and architectures, vary between teams and in time, and have differing results depending on specifics of the input data. In the redeployment experiments, most models struggled to achieve usable inference times on common commercial GPUs, and those that reported the hardware used in training and evaluation referenced high-end commercial models, which pose more issues to redeployment on less powerful hardware.

For the purposes of the work, constructing a novel architecture or a combination of the existing ones was deemed to be beyond scope. The required time and work, combined with the complexity of selecting one or more candidates to develop and train was deemed unreasonable. This is also compounded by the limitations in training licensing, especially in commercial usage later on, and the high complexity of redeployment itself and data management regarding the datasets. Models and datasets are often deeply intertwined, raising the complexity of adding a new data source or swapping in a new component. Collecting a custom dataset is also infeasible due to high costs and lack of relevant experience and hardware.

All these considerations, compounded with the results from the redeployment experiences, led to conclude that a pretrained and deployable model is the only possible choice going forward, showcasing good accuracy, low inference times and as liberal as possible licensing. This is quite a hard list of requirements to fulfill, but as noted before, MediaPipe does reach good results in all our metrics, and as such was chosen as the base model onto which we could construct our solution.

Chapter 3

Custom solution

As discussed in the last chapter, we selected a single model as a valid candidate for our task among the available research. The model in question, Google’s Mediapipe Hand Landmarker, or *Mediapipe model* for short, does not produce the required full 3D hand poses. It instead produces 2D and 2.5D keypoints, much like the other models surveyed. This limitation is again often not fully addressed in research, where tracking hand joint relative positions, usually in the 2.5D fashion, is the major concern, with less focus on the global localization task of 3D hand poses.

As discussed before, 2D and 2.5D keypoints do contain a lot of information about the hand pose, but they are not equivalent to 3D hand pose data, and are not sufficient to reconstruct this without additional information. The available 2D data, alongside the intrinsic camera parameters, which encode the properties of the camera’s field of view and lens system, allow us to reconstruct location information on the image plane, but leave out any depth information. The 2.5D pose data encodes the relative location of the hand joints but does not include any global location data. Thus, we are missing the depth or global location information necessary to bridge the gap between 2.5D and 3D data. [Richard Hartley, 2003]

We identify our task under the problem of **depth estimation**. In the literature that discusses the topic, different approaches are proposed:

- **RootNet** is a deep learning based solution that estimates global position of the root node of the 2.5D skeleton for full body pose estimation tasks, allowing for complete reconstruction of the pose. [Moon et al., 2019] This has been used in research on human pose estimation and has been reported as extensible to hand pose data, but no usable model for the task was available and the task of reconstruction and training was deemed outside the scope of

the project.

- **Multiple views** are a common solution to the issue in data collection apparatuses as we discussed before, due to their ability to reconstruct depth and global pose data, but are not available under the project's constraints.
- **Depth cameras**, especially combined systems that provide both RGB and depth data, are also a great solution to the problem, but much like multiple views are not available.

We considered the possibility of constructing a custom algorithm that leverages the data that Mediapipe produces for us and our task conditions, specifically the high quality 2D joints produced by the model, well known camera parameters and limited range of users. The solution designed is based on the apparent size of the hand with an added correction for the rotation angle with respect to the image plane. This approach requires minimal initial calibration phase for each user, considered acceptable setup. We also identified some supporting research for the method in VR applications, giving us some confidence in the efficacy of the method. [Reimer et al., 2023]

3.1 Depth estimation via apparent size

Depth estimation is the task of estimating distance from one point of view to an object in the scene, relying only on visual cues, instead of direct measurements or sensing apparatus. This can be achieved by leveraging perspective cues, motion cues, parallax information, or relying on multiple views. In this instance, we can only rely on a single view and the available information comes from the pose estimation data produced by Mediapipe and the RGB data itself from the video feed.

The chosen approach relies upon the apparent size of the hand, specifically the lengths of the segments connecting selected key points in the palm area. This information can be recovered using the 2D keypoints provided by Mediapipe. These keypoints were chosen as they appeared the most stable and clearest to track, with lower likelihoods of occlusions or ambiguities. They were also chosen as the larger distance between them would lead to less noise and jitter at lower resolutions or unfavorable angles. In the figure 3.1, these are the keypoints 0, 5 and 17.

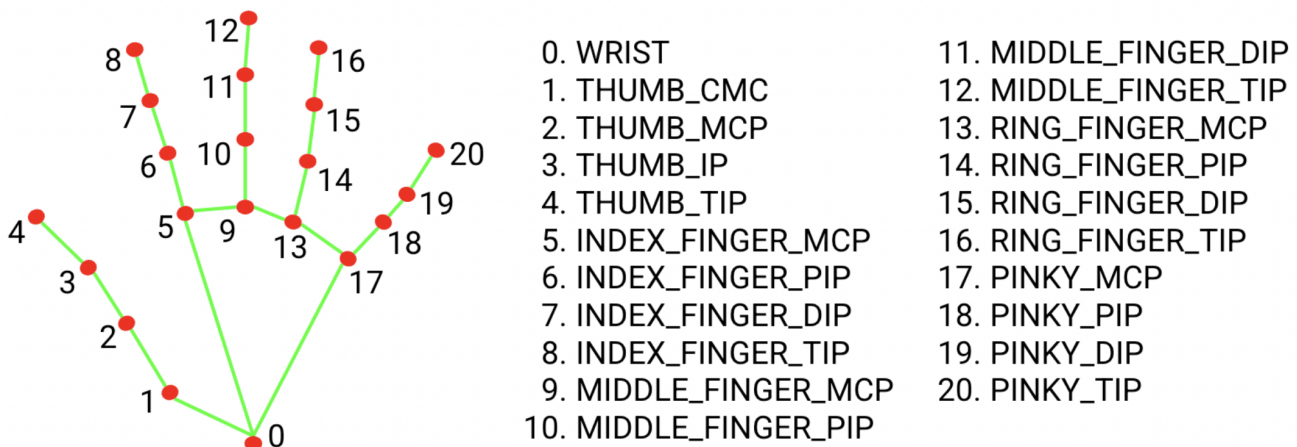


Figure 3.1: Map of the keypoints generated by Mediapipe [med, 2024]

Due to the rules of perspective geometry, we know that the apparent size of an object is inversely proportional to its distance from the viewpoint. Thus, we can extract the relative depth of the hand in two frames by taking the ratio of the two hand sizes. Moreover, if we capture a reference image at a known, fixed size, we can use it as a comparison to the other frames, and extract not just relative information but absolute depth, utilizing only 2D keypoints and a reference image. [Szeliski,]

The intrinsic camera matrix describes a camera system’s internal parameters and characteristics, such as sensor’s resolution, focal length. It is instrumental in many computer vision tasks, as it describes mathematically the protective transformation that maps 3D pints in the camera’s

field of view to 2D locations onto the image. Alongside the distortion parameters, which describe and help correct imperfections caused by the lens system, it is a fundamental characteristic of any camera system and does not change unless the camera is physically altered. These parameters are captured in a process called camera calibration, using a known object as reference. These parameters are fundamental to the working of the algorithm and were captured ahead of time for all our camera systems.

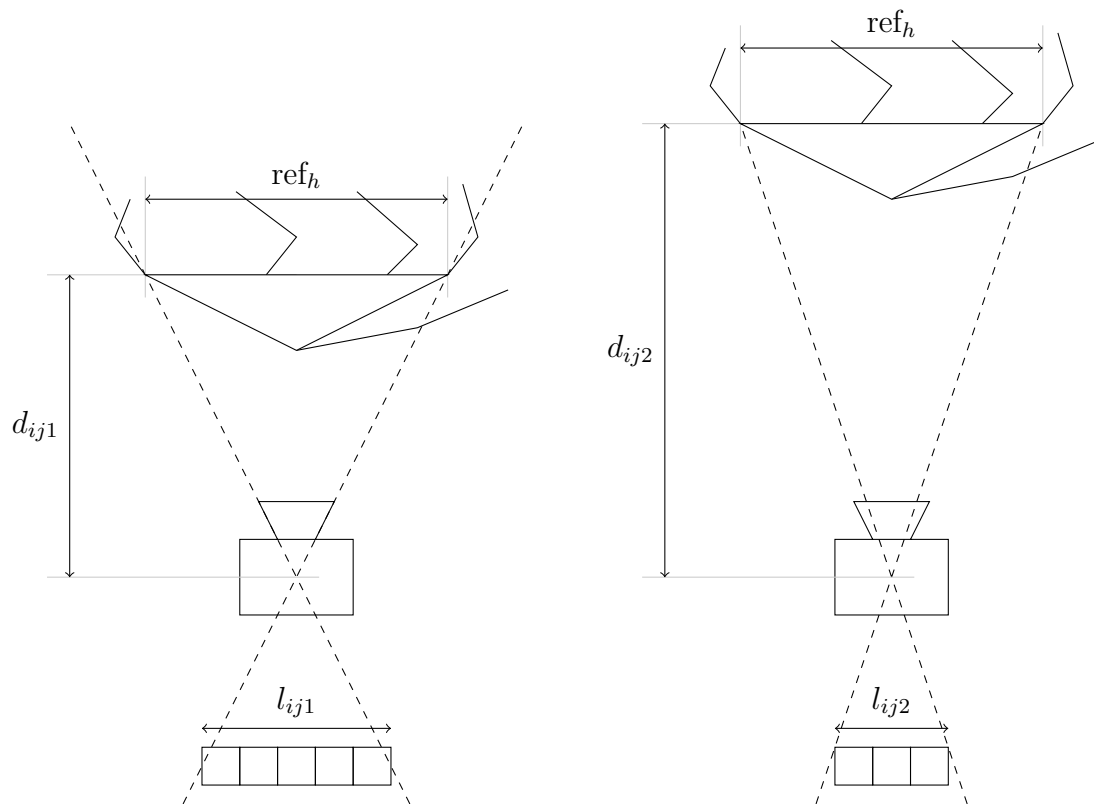


Figure 3.2: Visualization of the perspective geometry effect

An important observation is, apart from spheres and similarly rotationally symmetric objects, the apparent size of an object depends not only on the distance from the viewpoint, but also on the rotation with respect to it. Leveraging 2D keypoints, this issue becomes quite difficult to tackle, as it's not clear how to separate the changes in apparent size due to this effect, from those caused by movement along the Z axis. See 3.3 for a visualization.

This can be done instead by utilizing the 2.5D keypoints that Mediapipe gives us: These track the position of all the hand keypoints in 3D space, relative to the hand's geometric center. The positions given are also rotationally aligned with the camera view reference system. In other words, the X and Y coordinates if the keypoints describe the relative displacement of the

keypoints parallel to the image plane, and the Z component identifies its approximate relative depth.

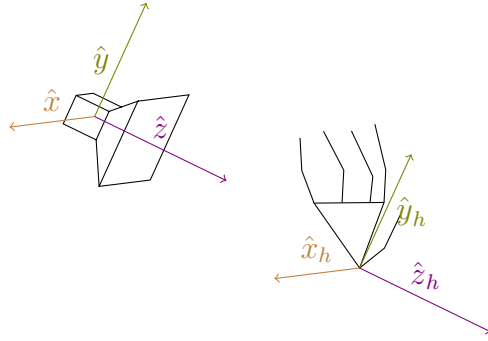


Figure 3.3: Visualization of the 2.5D keypoints reference system

The produced keypoints thus can capture information about the rotation of the hand. These keypoints still show major issues and instabilities, especially their relative distances can vary wildly when processing less favorable viewing angles or ambiguous poses, but the rotation information encoded in the pose is still sufficiently accurate to estimate the effect and correct for it.

To minimize error, in our method we focus on collecting only the rotation information between the keypoints, and we ignore data about their relative distance, which is less likely to be informative. In the formula, see 3.1, for each 2.5D keypoint pair p_i, p_j , we first compute the vector between the points $s_{i,j}$. This is then normalized, removing distance information. The normalized result $\hat{s}_{i,j}$ is used in a dot product with the normalized Z vector \hat{z} and passed through the $\arccos()$ function. The resulting angle α tell us the rotation of the vector relative to the image plane, specifically the angle between the vector \hat{z} and $\hat{s}_{i,j}$, which can be used to correct our depth estimation.

$$\begin{cases} s_{i,j} = p_i - p_j \\ \hat{s}_{i,j} = \frac{s_{i,j}}{\|s_{i,j}\|} \\ \alpha = \arccos(\hat{z} \cdot \hat{s}_{i,j}) \end{cases} \quad (3.1)$$

A final additional factor was introduced, as the rotation correction seems to introduce a small underestimation bias, especially at steeper angles. This was noticed early during a qualitative evaluation, and a small correction was added to correct it, based on the hand size and rotation. Specifically, the metric hand size ref_m is multiplied by the corrective factor $\cos(\alpha)$, based on the angle produced in the previous step, and then halved. This value is added to the final estimate.

The final equation of the depth estimation for a pair of hand keypoints is described in 3.2, where p^{2D} pixel coordinates of the keypoints, $p^{2.5D}$ the 2.5D corresponding coordinates, ref_{px} the reference hand size in pixels, ref_m the reference hand in meters, ref_d the reference distance. The final estimate is d .

$$\left\{ \begin{array}{l} l_{i,j} = \|p_i^{2D} - p_j^{2D}\| \\ s_{i,j} = p_i^{2.5D} - p_j^{2.5D} \\ \hat{s}_{i,j} = \frac{s_{i,j}}{\|s_{i,j}\|} \\ \alpha = \arccos(\hat{z} \cdot \hat{s}_{i,j}) \\ c = \frac{\text{ref}_m}{2} \cos(\alpha) \\ d = \frac{\text{ref}_{px}}{\text{ref}_d} \frac{\sin \alpha}{l_{i,j}} + c \end{array} \right. \quad (3.2)$$

The depth estimates of the three joint pairs are averaged each frame, as to give the most accurate estimate possible, for the hand depth, These estimates are also averaged between frames when processing videos, reducing further the effects of noise and small changes in depth estimate, and reducing high frequency noise components from the depth estimation.

3.1.1 Calibration phase

As previously mentioned, our method relies on an initial calibration step, where the geometry of the target hand is recorded, so that it can later be used in the estimation step to provide accurate metric distances. This step needs to extract the real world size of the hand, to be used as a reference point for subsequent comparisons, thus providing a ground truth value.

This is achieved by leveraging Mediapipe again. An image at a know distance and pose is captured, and the keypoints are extracted and used to estimate the hand's geometry. To ensure maximum accuracy, the background contains a calibration patter that is used to extract the rototranslation between the camera and the plane on which the hand rests. This rototranslation allows us to perform a homography from the image to the plane that the hand rests upon, including a small offset to take the hand's thickness into consideration. This removes any error due to small errors in camera placement and rotation, makes the process easier for the subject and allows the distance to be tested and corrected by relying on the pattern's known size.

The resulting aligned image is passed to Mediapipe, which extracts the 2D keypoints. The 2.5D keypoints are ignored, as we assume the hand to be laid flat. Leveraging the known distance

and intrinsic camera parameters, we can extract the 3D locations of the target keypoints, and store them. Alongside, we also store relevant metadata about the calibration process, such as the reference distance.

Storing the 3D locations instead of simply the hand size in pixels, allows for a wider range of uses, as now the calibration data is not wholly dependent on the intrinsic parameters of the camera. Instead, during pose extraction, a different camera can be used to collect the data and by providing the new camera’s intrinsics, we can re-project the 3D points in its model and produce the needed reference dimensions. Thus, we can calibrate the hand with a first camera A in a specific setup, and use another camera B to then capture the hand pose, while sharing the calibration data between systems.

3.1.2 3D pose reconstruction

The final step is to combine the 2D and 2.5D keypoints, the depth estimate and the intrinsic camera parameters to reconstruct the desired 3D hand pose.

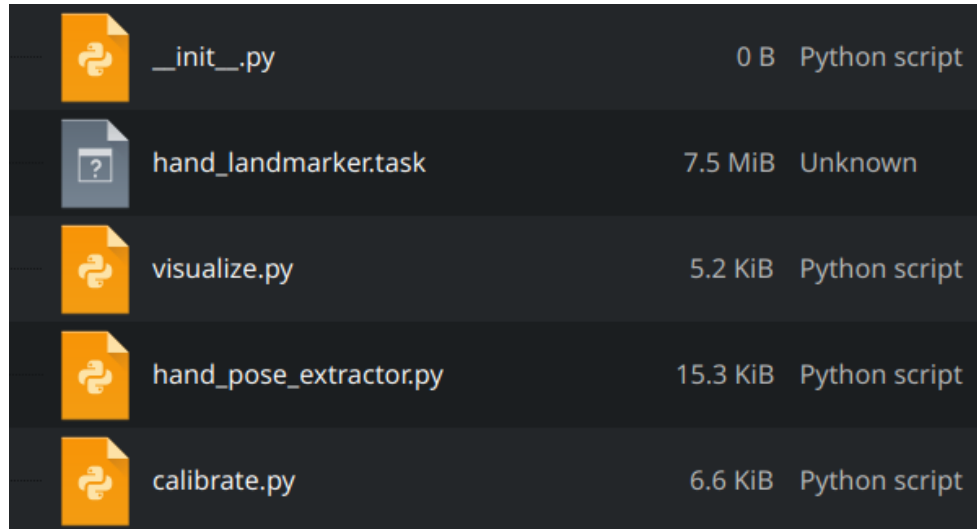
To achieve this, first we combine the 2D landmarks and the intrinsic parameters to reconstruct the 3D ray where they lay. This is done by moving the pixel coordinates to homogeneous coordinates, and then taking their product with the inverted camera matrix. This gives us the 3D vector on which those points lie. The next step is to compute the depth for each joint, thus we combine the hand depth estimate with the 2.5D joint depth. Finally, we take the product between the 3D coordinates and the depth, resulting in the reconstructed 3D pose.

$$\left\{ \begin{array}{l} P_i = \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \\ R_{3D,i} = I^{-1}P_i \\ d_i = r_d + R_i.z \\ R_{3D,itotal} = R_{3D,i}d_i \end{array} \right. \quad (3.3)$$

In 3.3, P_i defines the 2D keypoints of the i th joint homogeneous coordinates, while $R_i.z$ identifies the 2.5D keypoint’s depth value. R_d identifies the estimated depth of the hand’s root, located at its center. I indicates the camera’s intrinsic matrix. $R_{3D,itotal}$ indicates the final reconstructed 3D position of the i th joint.

3.2 Overall resulting library

The resulting algorithm is structured into a Python module, designed to be used both as a library and as a CLI program to process individual images.








	<code>__init__.py</code>	0 B	Python script
	<code>hand_landmarker.task</code>	7.5 MiB	Unknown
	<code>visualize.py</code>	5.2 KiB	Python script
	<code>hand_pose_extractor.py</code>	15.3 KiB	Python script
	<code>calibrate.py</code>	6.6 KiB	Python script

Figure 3.4: Folder structure of the Python package

The `__init__.py` file is a standard requirement for Python modules. It informs Python that the files here contained are not just a collection of scripts but are to be treated as package, so that they can be imported into another Python program to be invoked later on, or accessed via a command-line interface using the `python -m <package name>` command.

`calibrate.py` is responsible for managing calibration of the hand and producing the reference hand file. As discussed in the above section, the algorithm requires an existing calibration to gather information about the hand geometry to use as a reference in the depth estimation step. This process is camera independent, requires a single image of the hand from a calibrated camera at a known distance, and returns a NumPy file that contains the relevant information about the target hand, alongside relevant metadata about the calibration process, for debug purposes.

`hand_pose_extractor.py` is the core of the project. It performs the hand pose extraction via Mediapipe, the depth estimation and combines the data into 3 hand pose data. Needed geometric work is done here, leveraging heavily the matrix library NumPy to speed up operations. The results are made available via an HTTP server, managed by `visualize.py`, and can also be recorded to a file directly.

`hand_landmarker.task` is the Mediapipe TensorFlow Lite packaged model, provided by Google under the Apache 2.0 license and used locally to estimate the hand keypoints. This package

contains the core of the Mediapipe model, all the parameters and architecture, to be loaded by TensorFlow on initialization to produce the data needed.

`visualize.py` manages visualization and simple evaluation tasks, as well as managing exposing the output data as a JSON structure hosted on a local HTTP server. The server is run in a separate thread, making the process independent of the detection and estimation tasks. It is not used by itself, instead being invoked mostly to produce simple visualization of the resulting detection to quickly evaluate the results and to manage the flow of data to other processes.

As such, the resulting library make it simple to use an image from a first calibrated camera to extract the needed geometry parameters of a hand and to subsequently track the 3D position of the hand joints in another set of videos from other calibrated cameras, either via a command line interface or by integrating the library inside the data flow directly.

3.2.1 Data export and visualization

To help during development of the algorithm, to ensure the quality of the results and to help visualize our work, a few techniques were employed. We needed to study both 2D and 3D keypoints, and we wanted to do it in real time.

Visualizing 2D keypoints could be extracted directly from Mediapipe’s output, superimposed onto the captured frames using available tooling developed for Mediapipe and shown using OpenCV. This allowed to visually confirm the accuracy of the keypoints and correct some initial bugs and errors in detection. It also allowed us to evaluate qualitatively the tendency for false positives and false negatives in hand detection, and to understand more completely what triggered these kinds of errors.

Visualizing 3D data required a more complex solution. A first consideration was given for `matplotlib`, the Python plotting library. This would have made the project simple to manage and wholly built in Python. But managing live plotting of 3D data in unconstrained spaces with potentially multiple entities and real time updates is not a simple task for the library, with real time performance proving difficult to achieve and the documentation lacking details on the specific subject.

Instead, an external 3D program was chosen to act only as a visualizer, moving away from a Python ecosystem. The selected program is the open source 3D engine LÖVR (typically stylized as LOVR), written in C and scriptable in Lua. It was chosen due to

- **Familiarity**, as some members had already used in other projects

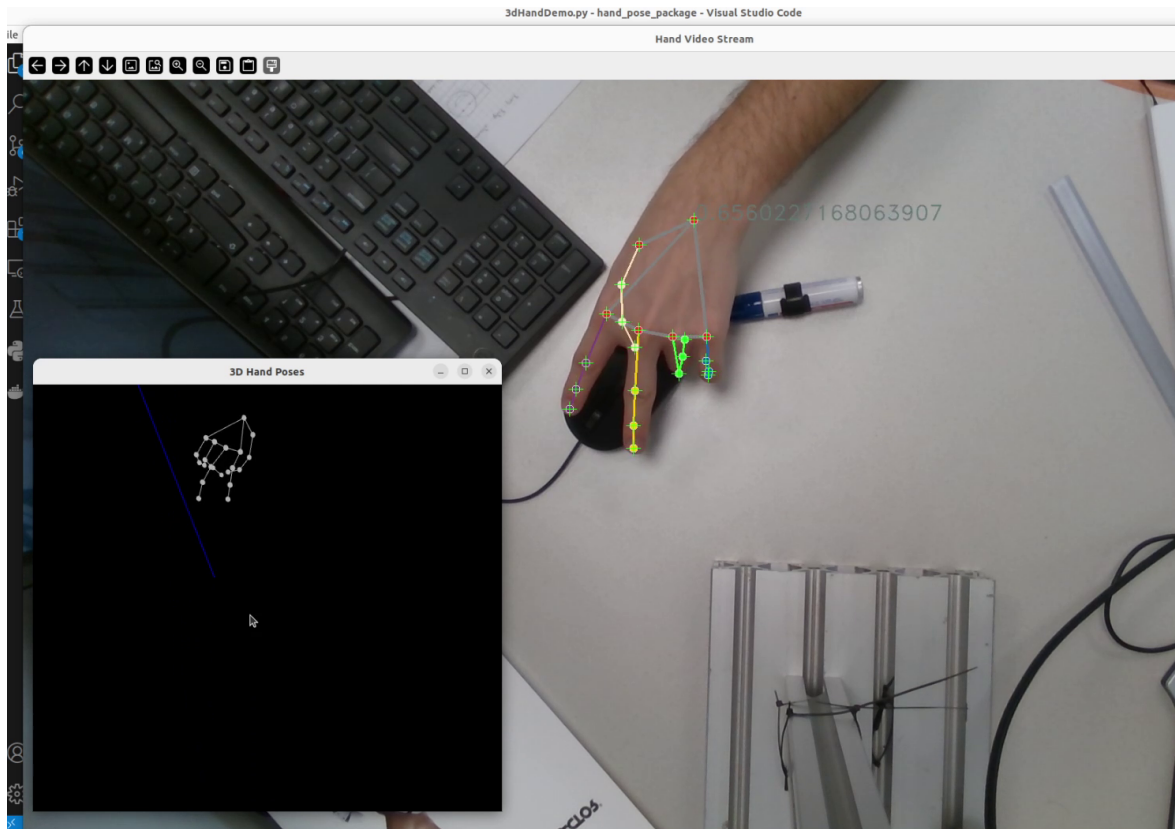


Figure 3.5: Example of library displaying both 2D and 3D keypoints

- **Low overhead**, as it has minimal resource consumption
- **High performance**, born as a VR engine, it is perfectly capable of displaying real time 3D data
- **Ease of use**, requiring only some Lua code to consume and process the stream through its integrated HTTP client

The data was packaged by the library into a JSON structure, and served optionally via an integrated Python HTTP server. The HTTP client would connect to the server and unpack the JSON data, which was then drawn with a simple skeletal representation. More elements such as known cameras and reference systems could be added to the visualization by controlling the behavior of the engine.

JSON was chosen to store and share the data, as it is a widely supported open standard file format, allowing to pass it between processes, languages and standard with a simple, machine and human-readable format.

Using this approach, the data could be visualized with very low latency, along with useful additional elements, such as reference systems or planes, and display various objects and information visually to avoid relying primarily on text output for debug during development.

3.3 Evaluations and Results

Once packaged and tested, the algorithm was evaluated in multiple stages, covering both the intended use case for our application and a more robust test of its general pose extraction capabilities.

Three stages of evaluation were done:

- **Qualitative phase**, used especially during development and as a first stage of evaluation. This was achieved by using an RGB+D camera, utilizing the measured depth as a ground truth to test our estimation against.
- **Pointing task**, designed to resemble the potential use case that was considered for the algorithm, where the objective was to track the tip of the index finger in 3D space, to measure ability to pinpoint locations.
- **HanCo**, a common hand pose dataset [Christian Zimmermann and Brox,], was used as a final evaluation step, to measure the accuracy of the algorithm against a common research dataset.

The first phase was mostly limited to development and a qualitative evaluation. While measurements were possible and these were actively used to test the algorithm's precision, this phase also highlighted some limitations and issues in using RGB+D cameras. While the estimates were very accurate and resistant to noise or lighting changes, they also showed issues linked to the limited software libraries and support available, as the camera showed inconsistent measurements of the depth, misalignments between the RGB and depth sensors, dropped frames and generally produced somewhat unreliable data streams. It also required unique libraries and modules to be installed and showed some issues in usage on different machines, pushing us to use Docker here also to fully insulate the runtime accessing the camera's data from the host OS. It was still very valuable in development by maintaining a tighter loop of development, evaluation and correction.

In the next subsections, we'll now discuss more in details the last two phases of evaluation, discussing the methodology, the data used, and the results produced, including quantitative insight on the actual performance on the system and demonstrated both the potential capabilities and limitations of this type of solution.

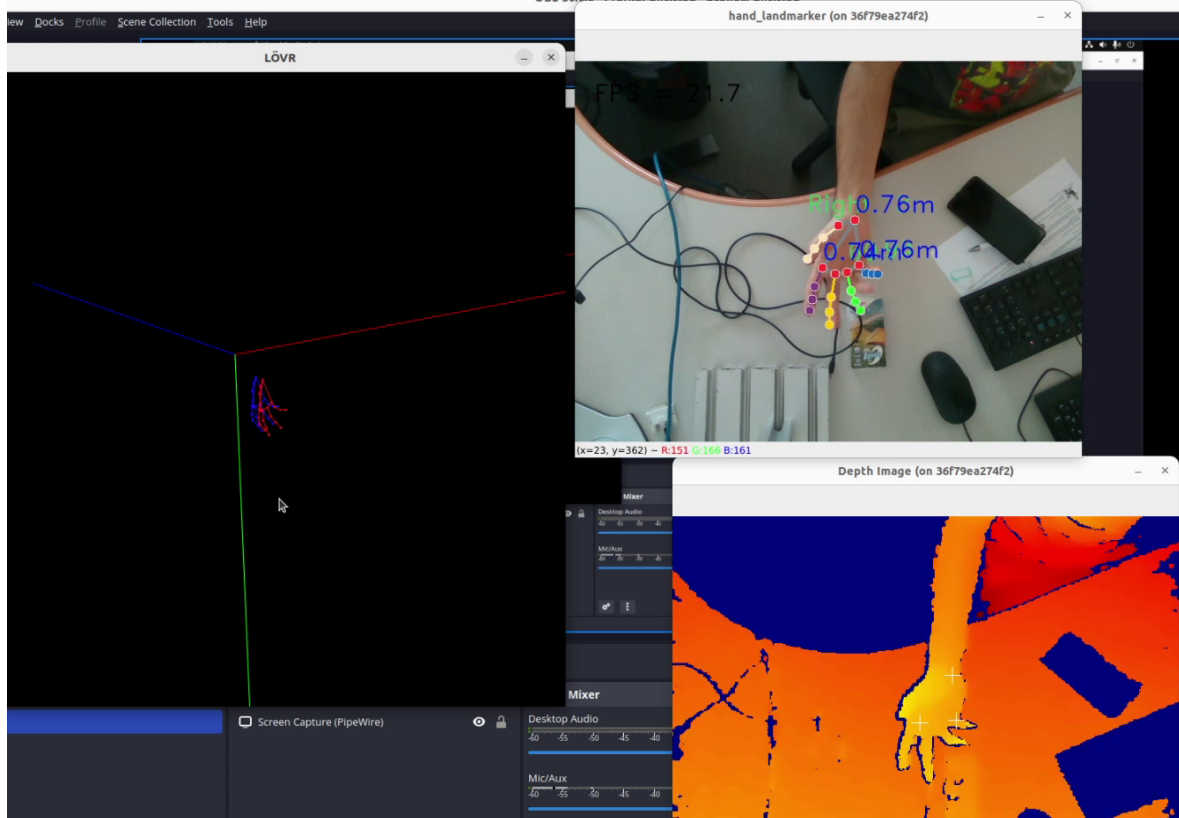


Figure 3.6: Visualization of the evaluation using the RGBD camera, comparing the estimated depth (blue) and measured depth (red). Also visible are the 2D keypoints with measured distance and the depth map.

Pointing Task

The first task we wanted to evaluate was a pointing task, specifically the accuracy of the model in tracking the 3D position of the index fingertip, to be used to point positions in 3D space in other applications. This was evaluated in a simplified environment using a calibration checkerboard as a ground truth for the locations. Multiple videos were captured, showing the hand moving over the checkerboard and stopping to point at the corners of the checkerboard pattern. The videos were captured with multiple cameras, some with a GoPro Hero 11 Black with a wide lens attachment, which required added work to correct the distortions of, and some with a IntelliSense R300 RGB+D camera, which had much less pronounced lens distortions.

The usage of the checkerboard pattern allowed us to both track the specific points in 3D space accurately, and easily tag at which frames the finger matched the position of the corners, which we could reconstruct using the known intrinsic and extrinsic parameters of the camera. The process of identifying the frames was done manually, inspecting the frames and identifying at

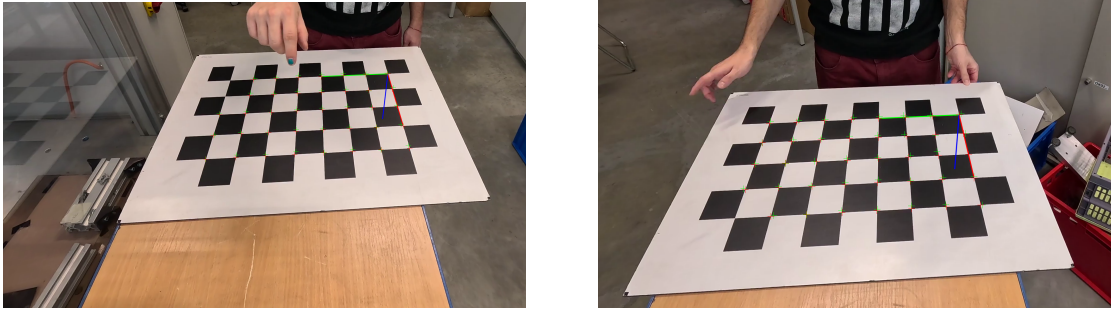


Figure 3.7: Examples of the pointing task, with small colored marks for 2D objective keypoints and measured keypoints

what instant the finger touched what position on the chessboard. To help in the task, a set of Python scripts were written to explore the video frame by frame, identify when the finger rested on the desired location and store the position and frame number pairing to disk. Calibration of the hand and of intrinsic parameters of all cameras was done ahead of time, selecting most favorable images and sequences to ensure best results.

The dataset consisted of 6 videos of the student moving their hand across the board and touching the chessboard at various corners, for a total of around 44 frames for each video, tagged with locations on 3D space. This totals around 250 3D locations to evaluate the task. A clean frame without the hand obstructing the checkerboard allowed us to accurately measure the extrinsic parameters of the scene for each video.

The results showed that while the X and Y coordinates were highly accurate in their estimations, the main source of error was in the Z coordinate, meaning that the error was linked to depth estimation. In 3.9 we can notice how while both the X and Y distributions are closely centered on 0, noting their high accuracy and stability, the Z values show both a significant bias of approximately 10 cm, with a wider spread.

These measurements showed issues of non-linearity, where the estimates vary as the hand moved across the screen away from the image center, as can be more clearly seen in 3.8. These errors were more pronounced in the videos captured using the wide angle lenses, hinting that the undistortion methods might interfere with the pose estimation or with the perspective calculations used to estimate depth.

The results show that the depth estimation solution did not achieve a very high accuracy on the task, and was subject to sensitivity in imperfections due to lens distortion and ambiguities in perspective. The results do demonstrate a limited range of error, ± 5 cm around the average of 10 cm, with an average absolute distance of around a meter from the camera for the chessboard.

Camera Space Points Comparison

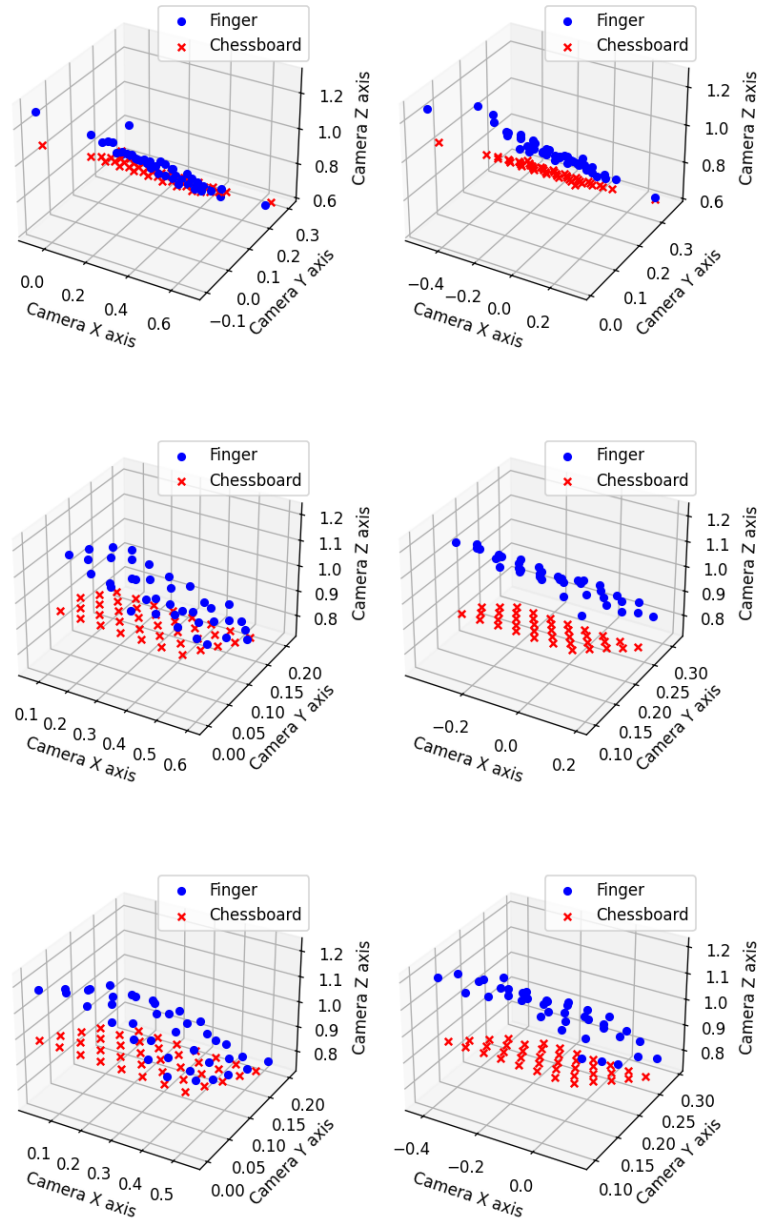


Figure 3.8: 3D visualization of the results from the GoPro camera

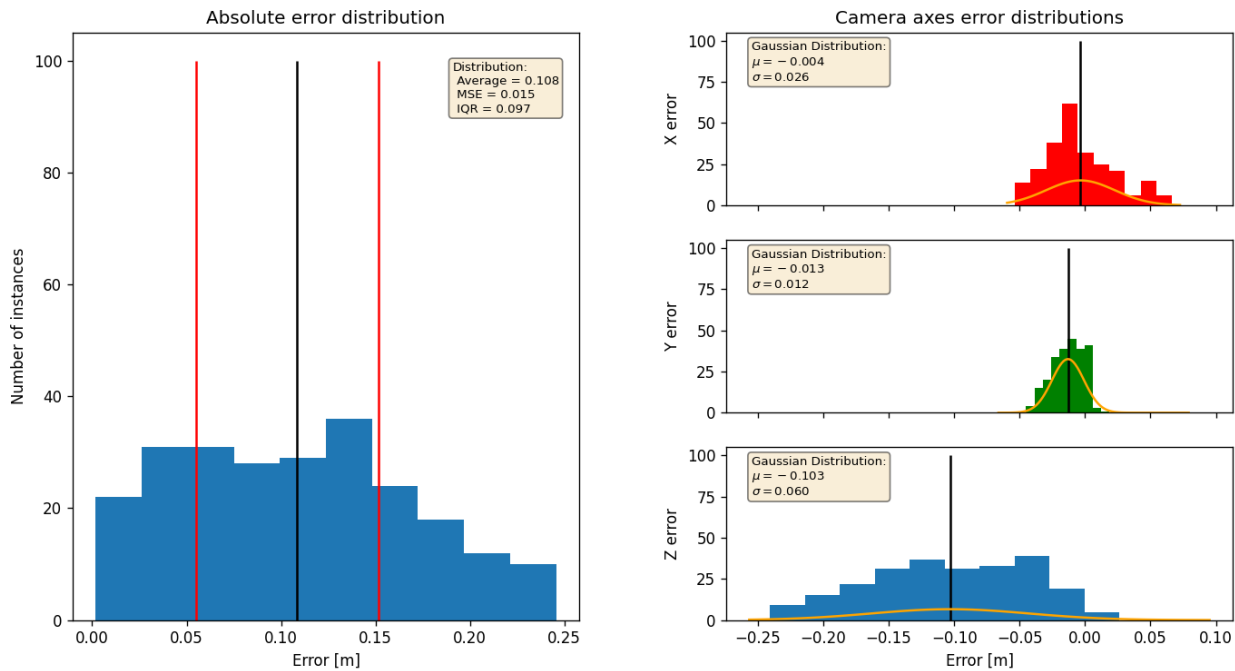


Figure 3.9: Distribution of the errors values, both absolute and along the individual axes

This results in an overall quite strong bias, and relevant variance.

This led to the conclusion that the system was not accurate enough for the desired task, where these locations had to be accurate with an error under a cm in all axes, but highlights the potential for more advanced solutions to reach higher precision and fulfill this need, while showing potential for applications with lower accuracy requirements.

HanCo Dataset

The final evaluation was on the HanCo [Christian Zimmermann and Brox,] dataset, developed by Freiburg University in Germany, as an extension of their previous FreiHand dataset [Christian Zimmermann and Brox, 2019]. Developed as part of the groups' long term research effort into the topic of hand pose estimation models.

Using such a dataset allows us to:

- **Define common ground** with other solutions and methods, by using a standard dataset
- **Evaluate beyond our use case**, by leveraging the wider set of hand poses, camera

positions and lighting conditions present in the dataset

- **Ensure high quality results**, taking advantage of the much wider scale and richness of the dataset, surpassing what could be produced internally

The dataset is composed of 1,518 short videos captured from multiple angles, resulting in a total of 860,304 frames with full 3D hand pose data available. Each video show an isolated hand, performing a specific pose or motion, on either a green background used for post-processing or on a neutral environment. These clips are recorded from 8 different angles, ensuring a wide range of points of view. The hand pose data is available in camera space 3D keypoints location for each frame, thus allowing an evaluation during motion and on a much larger dataset, alongside data detailing what participant's hand is shown, intrinsics and extrinsics of each camera.

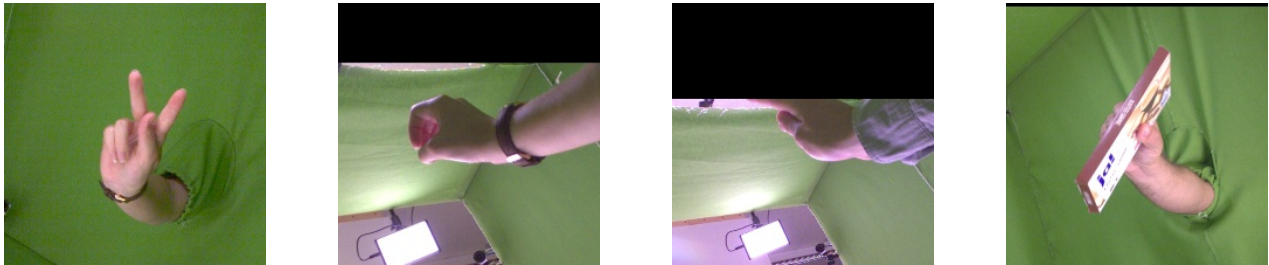


Figure 3.10: Examples of images from the HanCo dataset

The calibration step created additional complexity, as there were no direct hand measurement available as part of the dataset. Instead, for each hand used in the dataset, which were clearly labeled and thus allowed for the calibration to be reused, a frame where the hand was in the most favorable conditions was selected, and the ground truth data of the pose was used to calibrate the algorithm. This selection was done manually with help from a Python script that analyzed the ground truth of all images available and selected, for each user, the images whose joints were laid flat on a single plane. The selected images were shown to the user, which selected the best candidates. This likely reduced the accuracy of the method, introducing a certain level of bias due to the lack of accurate calibration.

In figure 3.11, we can observe a breakdown of the error of the 3D pose estimates over all 860,00 frames, both in absolute terms and for each axes. 3.12 also shows the breakdown of the error over individual joints. These results highlight similar issues as those encountered in the previous evaluation, where the X and Y coordinates display high precision and accuracy, while the Z coordinate is the source of most of the error. We can observe a clear indication of bias, with the average value and the peak of the error distribution indicating an average error in the order of 5 to 10 cm. Notable also is the wide spread of values, with errors exceeding ± 10 cm from

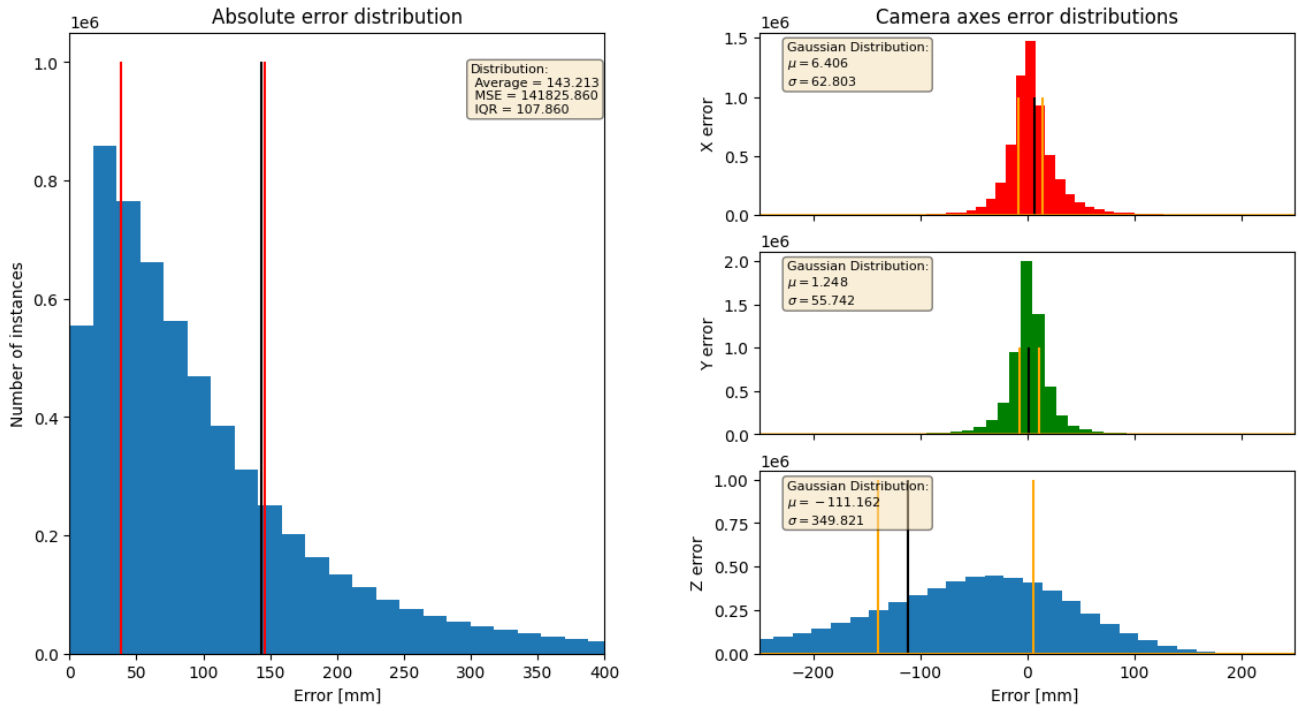


Figure 3.11: Distribution of the errors values, both absolute and along the individual axes

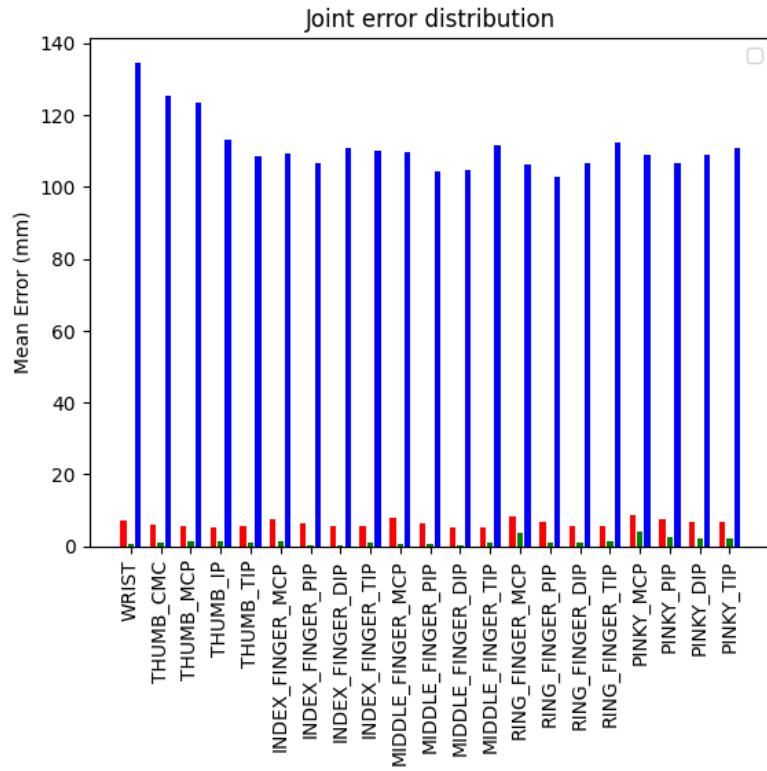


Figure 3.12: Distribution of the errors values along each axis, separated per hand joint

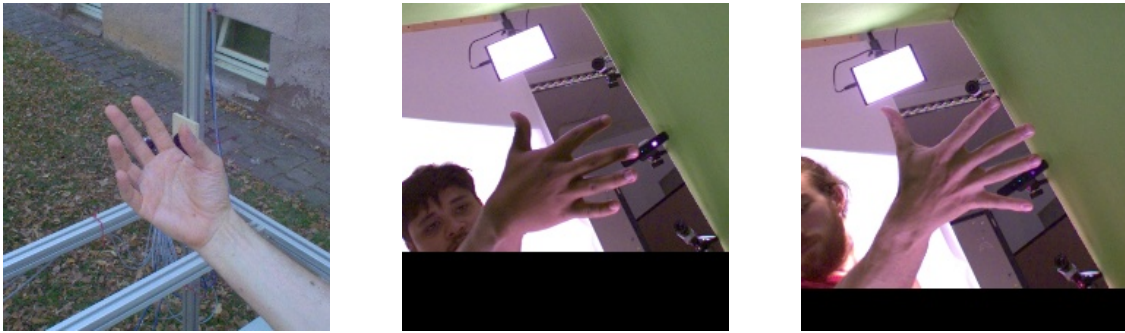


Figure 3.13: Examples of images used for calibration of the depth estimation model, from the HanCo dataset

the average. The distribution among joints also highlights how strong the difference in scale is between the axes, where the gap in mean error between the X and Y axes, respectively red and green, and the Z axis in blue is a visible reminder of this difference.

While some error, especially the bias, might be related with the difficulties in the calibration step, the overall results are consistent with our other evaluation. The algorithm shows high accuracy and precision on the X and Y components, thanks to the high quality 2D keypoints produced by Mediapipe that these are mostly based on. The depth estimation, and directly from that, the Z axis, is quite less accurate, and requires a good calibration to ensure better results, while still producing imperfections and errors, in the range of centimeters.

An important fact of the evaluation was that the detection rate of Mediapipe, so of the images with available ground truth how often it was able to recognize a hand and produce keypoints, was only around 41%. It clearly indicates the limitations of the available models in detecting and interpreting hands in less favorable conditions, even for highly developed models, once again highlighting the fundamental complexity and ambiguity of the task. It also has some influence on our interpretation of the results, as these are now intertwined with a survivorship bias, due to the failure of Mediapipe, but the overall interpretation of the accuracy in the detected frames remains the same. It is also an indicator of the intricacies and complexities of the task and dataset itself, displaying many highly unfavorable angles and ambiguous poses, in suboptimal lighting conditions.

Overall, one again the algorithm showed promising results but also highlighted some shortcomings of the method and of the underlying technologies. While some limitations could be caused by the variable quality and complexity of the evaluation material, such as very unfavorable capture angles, difficult lighting conditions, or relatively low resolution images, the results do show that the algorithm is not capable of high precision pose estimation. It also demonstrates

the limitations of the available models and the need for continued research and development of more advanced solutions, surpassing these issues.

Improvements and future work

Overall, the algorithm we developed to expand the available hand poses to 3D has shown some limitations and issues in precision. Obtaining a high quality calibration is key to better accuracy, which requires attention and is not always a feasible approach. The overall precision also is quite limited, displaying a range of error of multiple centimeters. This does limit the usability of the solution to lower stake tasks, but is not too far removed from what other models report as overall accuracy in their 2.5D and 3D estimation result, ranging usually in a few centimeters [Zimmermann et al., 2021] [Moon et al., 2020]. This might suggest that further developments and refinements might lead to better results without a total change in paradigm or major technical developments. Nonetheless, potential usage in multimedia media and UI applications is likely possible with little additional work.

Further technical developments might also be a major factor in improving performance. Continued research in both deep learning and computer vision models, and sustained commercial interest in the technology suggests that new and improved models, with higher accuracy and stability could be available in the future, allowing more complex reconstruction solutions or directly providing 3D pose data.

Future evolution of the Mediapipe library and models themselves could enable expansion to their techniques to reconstruct 3D poses from the available 2D and 2.5D data, especially if the stability of the 2.5D keypoints can be improved. Techniques based on Perspective-n-Point problem could also be developed to reconstruct the global position information. It was considered in the early stages of our work, but proved hard to adapt to the available data, and was not implemented. These problems pertain the general problem of accurately connecting global camera position, the location of 3D points their 2D projection in the image plane, and have been used in simpler pose estimation problems.

Another possible approach that was initially considered was to follow the steps of model based projects, where a constrained 3D model is deformed and manipulated to fit the known data of the hand pose, trying to lower metrics like reprojection error. These were discarded later due to the large amount of work to develop them, adapt them to the available data and necessity to calibrate the models.

A final observation pertains to the wider context of hand pose estimation and its commercial

entities. Commercial solutions such as those used in VR/AR headsets developed by both Meta, Apple and Microsoft are already showing fast, accurate and precise hand estimation, although these remain limited to lower stakes environment such as entertainment, UI and multimedia. Commercial usage is expanding, but with limitations to media production, 3D art and showcases, with limitations and slower expansions in high stakes fields such as healthcare, safety and other industrial environments. Many of these aforementioned solutions do not rely on a single RGB camera, combining either multiple RGB camera views or relying on specialized depth sensing hardware. These extra sources of information greatly help close the gap between 2D and 3D data, and allow for much more accurate results. This also results in higher costs and complexity, and highlights the fundamental difficulty of solving the problem with a single RGB video source. RGB data sources will always maintain a certain level of ambiguity regarding geometric information, especially for complex objects such as hands. Solutions based on these data sources alone will have to contend and inherit these problems.

3.4 Conclusions

In this work, we wanted to construct a working solution to the problem of reconstructing the 3D pose of human hands from a single RGB video source. As discussed multiple times here, this is a fundamentally ambiguous task, where researchers and companies are still actively working to achieve better results and construct paths to this and other intermediate solutions. A single RGB view point is inherently limited in its ability to describe the geometry of what it sees.

Commercial products and solutions under more relaxed requirements exist, relying on additional data sources to tackle this gap, such as multiple cameras, depth sensing apparatuses, visual markers and more. These solutions have been successful in their applications, especially in entertainment and consumer sectors, and have driven major investments and developments in the field, although focusing less on this specific version.

We've explored the available models and solutions produced by researchers and companies, understanding their inner workings and attempting to utilize their models for our purposes. In this endeavor, we discovered some limitations in the available research. A major segment of papers and published works do not share trained models or code to reconstruct their research, limiting heavily limiting the possible avenues for further researchers to improve upon their designs and work, or for other entities to leverage their results for commercial or other applications. As such, due to our focus on deploying a working model we focused on papers that did share source code and trained models. Even with these caveats we encountered issues in usage, as documentation

for libraries, environments, parameters and other requirements were often lacking or entirely missing. Models were sometimes given with little explanation over their usage or needs, or incomplete information. Major action was needed to bring only a few models to a usable state and decide if they reached our requirements of accuracy and speed.

Ultimately, we only selected one, the Mediapipe Hand Landmarker pose estimation model developed by Google, as a candidate for our problem. Much like other models, this did not produce 3D pose data, but only a subset of 2D and 2.5D pose data, as discussed in previous chapter. This gap required additional work to be filled, and the task of extending these keypoints to global position information became our focus.

The algorithm developed relied upon a simple geometric approach and initial calibration step to estimate the missing depth information regarding the user's hand. It was constructed to leverage the higher quality 2D keypoints, while utilizing the 2.5D rotation and pose data mostly to correct its initial estimate. The resulting method was packaged into a Python library, alongside the needed functions to load and prepare the Mediapipe model, to calibrate the system to a different user, to export and load saved data. A custom 3D visualization tool was created to aid in evaluation and usage, based on the 3D engine LOVR.

This system was then evaluated, both qualitatively to removed major bugs and correct behavior, and the quantitatively, on a custom dataset reflective of the intended usage of the solution, and on a commonly used research dataset for hand pose estimation, the HanCo dataset. These evaluations produced precise assessments of the precision and accuracy of the method: the produced data was more accurate in the image location portion, but the depth estimation produced relatively low precision data. Depth estimates errors were measured at around 10 cm on average, with a noticeable spread of around ± 10 cm more, depending on calibration and pose. These results could only partially be attributed to the more complex poses that the dataset featured, and were accompanied by a larger issue of missing detection on the part of Mediapipe's model, which detected the hands in only 41 % of the cases. This reflects the limitations of the available models and the complexity of the task.

The results put the solution outside the requirements for the intended task, of tracking the user hands to identify specific points in 3D space, with an error of only a few millimeters. These results are still not too far removed from current state-of-the-art solutions by more than one order of magnitude, while providing global 3D pose data, an uncommon form of output in the field. The reported accuracy is still suitable for some applications in entertainment and multimedia sectors, where speed and ease of use can overcome needs of high precision. We also expect future developments in the field to bring forth new models with more accurate results for researchers to

work on. Further techniques based model reconstruction or on perspective and point methods could also produce better results.

The problem remains unsolved and with inherent ambiguities lurking in it, both in the lack of information for the RGB data and in the high complexity human hands. Their mobility, complexity, tendency to be occluded by objects, clothes and each other increases only the difficulty of the task, and opens up new problems.

Bibliography

- [med, 2021] (2021). Mediapipe hands model card.
- [med, 2024] (2024). Mediapipe hands documentation.
- [Doc, 2024] (2024). What is a container? - docker resources.
- [Ahmad et al., 2019] Ahmad, A., Migniot, C., and Dipanda, A. (2019). Hand pose estimation and tracking in real and virtual interaction: a review. *Image and Vision Computing*, 89:35–49.
- [Avogaro et al., 2023] Avogaro, A., Cunico, F., Rosenhahn, B., and Setti, F. (2023). Markerless human pose estimation for biomedical applications: a survey. *Frontiers Comput. Sci.*, 5.
- [Avola et al., 2022] Avola, D., Cinque, L., Fagioli, A., Foresti, G. L., Fragomeni, A., and Panzone, D. (2022). 3d hand pose and shape estimation from rgb images for keypoint-based hand gesture recognition. *Pattern Recognition*, 129:108762.
- [Bartol et al., 2020] Bartol, K., Bojanić, D., Petković, T., D’Apuzzo, N., and Pribanić, T. (2020). A review of 3d human pose estimation from 2d images. *Proceedings of 3DBODY.TECH 2020 - 11th International Conference and Exhibition on 3D Body Scanning and Processing Technologies, Online/Virtual, 17-18 November 2020*.
- [Basafa et al., 2017] Basafa, E., Foroughi, P., Hoßbach, M., Bhanushali, J., and Stolka, P. J. (2017). Visual tracking for multi-modality computer-assisted image guidance. In *Medical Imaging*.
- [Brown et al., 2018] Brown, A. J. V., Uneri, A., Silva, T. S. D., Manbachi, A., and Siewerdsen, J. H. (2018). Design and validation of an open-source library of dynamic reference frames for research and education in optical tracking. *Journal of Medical Imaging*, 5(2):021215.
- [Chatzis et al., 2020] Chatzis, T., Stergioulas, A., Konstantinidis, D., Dimitropoulos, K., and Daras, P. (2020). A comprehensive study on deep learning-based 3d hand pose estimation methods. *Applied Sciences*.

- [Chen et al., 2020a] Chen, X., Fan, H., Girshick, R., and He, K. (2020a). Improved baselines with momentum contrastive learning.
- [Chen et al., 2020b] Chen, Y., Ma, H., Kong, D., Yan, X., Wu, J., Fan, W., and Xie, X. (2020b). Nonparametric structure regularization machine for 2d hand pose estimation. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 370–379.
- [Chen et al., 2020c] Chen, Y., Tian, Y., and He, M. (2020c). Monocular human pose estimation: A survey of deep learning-based methods. *Comput. Vis. Image Underst.*, 192:102897.
- [Christian Zimmermann and Brox, 2019] Christian Zimmermann, Duygu Ceylan, J. Y. B. R. M. A. and Brox, T. (2019). Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*.
- [Christian Zimmermann and Brox,] Christian Zimmermann, M. A. and Brox, T. Contrastive representation learning for hand shape estimation.
- [Colyer et al., 2018a] Colyer, S. L., Evans, M., Cosker, D. P., and Salo, A. I. T. (2018a). A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system. *Sports Medicine - Open*, 4(1):24.
- [Colyer et al., 2018b] Colyer, S. L., Evans, M., Cosker, D. P., and Salo, A. I. T. (2018b). A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system. *Sports Medicine - Open*, 4.
- [Contributors, 2020] Contributors, M. (2020). Openmmlab pose estimation toolbox and benchmark. <https://github.com/open-mmlab/mmpose>.
- [Doosti, 2019] Doosti, B. (2019). Hand pose estimation: A survey.
- [Garcia-Hernando et al., 2018] Garcia-Hernando, G., Yuan, S., Baek, S., and Kim, T.-K. (2018). First-person hand action benchmark with rgb-d videos and 3d hand pose annotations.
- [Gomez-Donoso et al., 2017] Gomez-Donoso, F., Orts-Escolano, S., and Cazorla, M. (2017). Large-scale Multiview 3D Hand Pose Dataset. arXiv:1707.03742 [cs].
- [Hampali et al., 2021] Hampali, S., Sarkar, S. D., and Lepetit, V. (2021). HO-3D_v3: Improving the Accuracy of Hand-Object Annotations of the HO-3D Dataset. arXiv:2107.00887 [cs].
- [Hsiao and Chiu, 2021] Hsiao, S.-C. and Chiu, C.-T. (2021). Efficient 2d keypoint-based hand pose estimation. *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1648–1652.

- [Iqbal et al., 2018] Iqbal, U., Molchanov, P., Breuel, T., Gall, J., and Kautz, J. (2018). Hand Pose Estimation via Latent 2.5D Heatmap Regression. arXiv:1804.09534 [cs].
- [Jiang et al., 2023] Jiang, C., Xiao, Y., Wu, C., Zhang, M., Zheng, J., Cao, Z., and Zhou, J. T. (2023). A2J-Transformer: Anchor-to-Joint Transformer Network for 3D Interacting Hand Pose Estimation from a Single RGB Image. arXiv:2304.03635 [cs].
- [Kaid and Baina, 2023] Kaid, A. E. and Baina, K. (2023). A systematic review of recent deep learning approaches for 3d human pose estimation. *Journal of Imaging*, 9.
- [Kong and ju Kang, 2021] Kong, D. and ju Kang, S. (2021). Downsizing heatmap resolution for real-time 3d human pose estimation. *2021 36th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, pages 1–4.
- [Lee and soo Lee, 2018] Lee, J. Y. and soo Lee, C. (2018). Path planning for scara robot based on marker detection using feature extraction and, labelling. *International Journal of Computer Integrated Manufacturing*, 31(8):769–776.
- [Li et al., 2019] Li, R., Liu, Z., and Tan, J. (2019). A survey on 3d hand pose estimation: Cameras, methods, and datasets. *Pattern Recognition*, 93:251–272.
- [Meng et al., 2022] Meng, H., Jin, S., Liu, W., Qian, C., Lin, M., Ouyang, W., and Luo, P. (2022). 3D Interacting Hand Pose Estimation by Hand De-occlusion and Removal. arXiv:2207.11061 [cs].
- [Merlau et al., 2023] Merlau, B., Cormier, C., Alaux, A., Morin, M., Montané, E., Amarantini, D., and Gasq, D. (2023). Assessing spatiotemporal and quality alterations in paretic upper limb movements after stroke in routine care: Proposal and validation of a protocol using imus versus mocap. *Sensors (Basel, Switzerland)*, 23.
- [Meyer and Wilson, 1991] Meyer, J.-A. and Wilson, S. W. (1991). Biological and computational stereo vision.
- [Moeslund and Granum, 2001] Moeslund, T. B. and Granum, E. (2001). A survey of computer vision-based human motion capture. *Comput. Vis. Image Underst.*, 81(3):231–268.
- [Moon et al., 2019] Moon, G., Chang, J. Y., and Lee, K. M. (2019). Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image.
- [Moon et al., 2020] Moon, G., Yu, S.-I., Wen, H., Shiratori, T., and Lee, K. M. (2020). Inter-Hand2.6M: A Dataset and Baseline for 3D Interacting Hand Pose Estimation from a Single RGB Image. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M., editors, *Computer*

- Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 548–564, Cham. Springer International Publishing.
- [Mündermann et al., 2006] Mündermann, L., Corazza, S., and Andriacchi, T. P. (2006). The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications. *Journal of NeuroEngineering and Rehabilitation*, 3(1):6.
- [Muramatsu et al., 2022] Muramatsu, N., da Silva, Z., Joska, D., Nicolls, F., and Patel, A. (2022). Improving 3d markerless pose estimation of animals in the wild using low-cost cameras. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3770–3776.
- [Prince, 2023] Prince, S. J. (2023). *Understanding Deep Learning*. The MIT Press.
- [Reimer et al., 2023] Reimer, D., Podkosova, I., Scherzer, D., and Kaufmann, H. (2023). Evaluation and improvement of HMD-based and RGB-based hand tracking solutions in VR. *Frontiers in Virtual Reality*, 4.
- [Richard Hartley, 2003] Richard Hartley, A. Z. (2003). *Multiple View Geometry in Computer Vision*.
- [Scheepers, 2014] Scheepers, M. J. (2014). Virtualization and containerization of application infrastructure : A comparison.
- [Schroff et al., 2015] Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.
- [Seethapathi et al., 2019] Seethapathi, N., Wang, S., Saluja, R., Blohm, G., and Kording, K. P. (2019). Movement science needs different pose tracking algorithms.
- [Spurr et al., 2022] Spurr, A., Dahiya, A., Wang, X., Zhang, X., and Hilliges, O. (2022). PeCLR: Self-Supervised 3D Hand Pose Estimation from monocular RGB via Equivariant Contrastive Learning. arXiv:2106.05953 [cs].
- [Szeliski,] Szeliski, R. *Computer Vision: Algorithms and Applications*.
- [Tompson et al.,] Tompson, J., Stein, M., Lecun, Y., and Perlin, K.
- [Topham et al., 2022] Topham, L. K., Khan, W., Al-Jumeily, D., and Hussain, A. J. (2022). Human body pose estimation for gait identification: A comprehensive survey of datasets and models. *ACM Computing Surveys*, 55:1 – 42.

- [Topley and Richards, 2020] Topley, M. and Richards, J. G. (2020). A comparison of currently available optoelectronic motion capture systems. *Journal of Biomechanics*, 106:109820.
- [Wade et al., 2022] Wade, L., Needham, L., McGuigan, P. M., and Bilzon, J. L. J. (2022). Applications and limitations of current markerless motion capture methods for clinical gait biomechanics. *PeerJ*, 10.
- [Wang et al., 2023] Wang, S., Zhu, M., Hu, Y., Li, D., Yuan, F., and Yu, J. (2023). Cylindertag: An accurate and flexible marker for cylinder-shape objects pose estimation based on projective invariants. *IEEE transactions on visualization and computer graphics*, PP.
- [Wen et al., 2023] Wen, Y., Pan, H., Yang, L., Pan, J., Komura, T., and Wang, W. (2023). Hierarchical Temporal Transformer for 3D Hand Pose Estimation and Action Recognition from Egocentric RGB Videos. arXiv:2209.09484 [cs].
- [Xiong et al., 2019] Xiong, F., Zhang, B., Xiao, Y., Cao, Z., Yu, T., Zhou, J. T., and Yuan, J. (2019). A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image.
- [Zhang et al., 2020] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., and Grundmann, M. (2020). MediaPipe Hands: On-device Real-time Hand Tracking. arXiv:2006.10214 [cs].
- [Zimmermann et al., 2021] Zimmermann, C., Argus, M., and Brox, T. (2021). Contrastive Representation Learning for Hand Shape Estimation. arXiv:2106.04324 [cs].

Ringraziamenti

A Trapper, Cinzia, Valentina, Rita, Cristina, Federico e il resto della mia famiglia,

Ad Andrea, Ferruccio, Germana, Michela, Simonetta, Annalisa e il resto della mia seconda famiglia,

A Massimo, Thomas, Gabriele, Giovanni, Elettra, Pietro, Carlo, Aron, Francesco, Alex, Jasmine, Anna, Riccardo e tutti i miei amici,

To Florian, Sesko, Lukas, Paul, Harald and all the others at Profactor,

Alla mamma