# Skeleton Detection Using MediaPipe as a Tool for Musculoskeletal Disorders Analysis

Josef Böhm[1] ✉, Taotao Chen[2], Karel Štícha[2], Jan Kohout[2], and Jan Mareš[1,2]

[1]University of Pardubice, Faculty of Electrical Engineering and Informatics,
Studentská 95, 532 10 Pardubice 2
Czech Republic
[2]University of Chemistry and Technology in Prague,
Department of Mathematics, Informatics and Cybernetics
Technická 1905/5, 166 28 Praha 6
Czech Republic
josef.bohm@upce.cz

**Abstract.** Skeleton detection, also known as human pose estimation (HPE), is becoming more and more popular as it can be applied in a range of applications such as game entertainment, human-machine interaction, VR-based projects, medical rehabilitation, etc. Thanks to the booming development of deep learning, HPE solutions can be implemented using deep learning methods which require standard 2D RGB images or video sequences as input. That is, technology nowadays is making HPE solutions more and more lightweight and fast which is possible to run on mobile devices for the daily use of skeleton detection. This article covers a brief survey of current deep learning-based human pose estimation approaches in the first place. Then, a lightweight deep learning model – MediaPipe – will be illustrated from all the perspectives of its structure, working flow, strengths & weaknesses and the more concerned compatibility in platforms and programming languages. As a result, a multi-platform application for collecting movement data from patients suffering from musculoskeletal diseases relying on MediaPipe is introduced. Finally, there is a summary of achievements and obstacles of application development, which is significant as it can be a signpost for teams who are doing or about to do an application based on the MediaPipe library.

**Keywords:** Skeleton detection, MediaPipe, Desktop application, Mobile application, Musculoskeletal disorders, Deep learning, Windows, C#, Image processing

# 1 Introduction

Nowadays, skeleton detection or human pose estimation plays a more and more important role in many aspects of people's life such as healthcare, security, video game and animation industry, etc. Skeleton detection is actually a computer vision technique that aims to capture or track the human skeleton within an image or video. The basic idea to achieve skeleton detection is to identify some key points or joints of a human body, such as the head, shoulders, elbows, wrists, hips, knees, and ankles, and connect them to form a skeletal structure. The primary purpose of skeleton detection is to understand the pose, movement, and spatial relationships of human bodies in visual data. And this information can then be used in practical applications.

The requirements of such human body detection have motivated a series of input devices such as Microsoft's Xbox 360 Kinect, PS4 eye, Intel RealSense Camera, etc. All these devices use the so-called depth sensor together with a standard RGB camera to sense the 3D features of the body regardless of the illumination. However, the working mechanism of the Kinect like cameras limits the use on mobile devices such as smartphones, tablets, etc. which apparently represents the trend of the future. Due to high manufacturing and production costs, Microsoft has decided to discontinue its Kinect camera.

In recent years, there has been a focus on developing lightweight and efficient models for real-time and mobile applications. These models aim to strike a balance between accuracy and computational efficiency to enable real-time skeleton detection on devices with limited resources, such as smartphones or edge devices. A typical example is MediaPipe, which is a framework integrated with pretrained models and tools to enable the applications of skeleton detection in a wide range of aspects, healthcare, gaming, sports analysis, and more.

In this article, a MediaPipe-based solution for human skeleton detection applied to help patients with medical rehabilitation will be introduced.

## 1.1 Medical background

Skeletal tracking and human pose estimation have emerged as valuable tools in the field of medical research and clinical applications. Accurate assessment of human movement and posture is essential for diagnosing and monitoring various conditions, particularly those related to gait disorders or musculoskeletal impairments. By analyzing skeletal data, clinicians and researchers can gain insights into patients' motor function, identify abnormalities, track progress, and develop targeted rehabilitation strategies.

The ability to detect and track the human skeleton using computer vision techniques has opened up new possibilities for non-invasive and objective analysis of movement patterns. Traditional methods for assessing gait and posture often rely on subjective observations or expensive and time-consuming motion capture systems. With skeletal tracking, healthcare professionals can obtain valuable quantitative data in a more efficient and cost-effective manner.

In the medical field, skeletal tracking has found applications in various areas. A prominent example is the diagnosis and management of gait disorders. Gait abnormalities can be indicative of underlying neurological, orthopaedic, or musculoskeletal conditions. By analysing movement patterns captured through skeletal tracking, clinicians can assess parameters such as step length, cadence, symmetry, and joint angles, aiding in the identification and evaluation of gait impairments.

Furthermore, skeletal tracking has been proven beneficial in rehabilitation settings. By closely monitoring a patient's movements and body posture, therapists can objectively evaluate the effectiveness of interventions and track the progress of rehabilitation programmes. This data-driven approach allows for personalised treatment plans and better outcomes for patients undergoing physical therapy or recovering from injuries.

In addition to clinical applications, skeletal tracking has the potential for preventive healthcare and telemedicine. It enables remote monitoring and assessment of patients' motor function, providing healthcare professionals with valuable information for early detection of movement abnormalities or changes in posture. This technology holds promise for home-based rehabilitation programmes, allowing patients to receive ongoing support and guidance from healthcare providers without the need for frequent in-person visits.

As the field of skeletal tracking continues to advance, there is a growing focus on developing lightweight and efficient models that can be deployed on portable devices. This shift to mobile applications opens opportunities for point-of-care diagnostics, ambulatory monitoring, and integration with wearable devices. These advancements hold the potential to revolutionise the way we evaluate and manage musculoskeletal conditions, enabling more accessible and personalised healthcare solutions.

In general, the integration of skeletal tracking into medical research and clinical practise has the potential to enhance diagnostics, improve treatment outcomes, and transform the way we understand and address gait disorders and musculoskeletal impairments. With continued advancements in technology and increasing adoption in healthcare settings, skeletal tracking is poised to play an integral role in improving patient care and promoting better overall musculoskeletal health.

## 2 Materials and Methods: Principles and Technologies of Human Skeleton Detection

For decades, people have been developing skeleton detection techniques. Motivated by advancements in computer vision, deep learning, and camera technologies, people have gained huge progress in skeleton detection with one after another improved method. An overview of how human pose estimation problem is being developed can be summarized with Table 1 below [7].

Table 1: Classification of HPE approaches.

| An review of classified human pose estimation approaches | |
|---|---|
| Type of input | RGB image; Depth (Time of Flight) image; Infra-red (IR) image) |
| Number of cameras | Single-view; Multi-view |
| Human body models | Kinematic; Planar; Volumetric |
| Type of images | Static; Frames from video sequences |
| Number of people | Single-person pose estimation; Multi-person pose estimation |
| Dimension of HPE | 2D pose estimation; 3D pose estimation |

Usually, human pose estimation can be divided into 2 groups: 2D human pose estimation and 3D human pose estimation.

**2D human pose estimation** works with localization of key points of a human body in 2D space, namely obtaining the X and Y coordinates of human body within an image or a video frame [19].

**3D human pose estimation**, similarly, is associating with predicting the spatial coordinates of a human body in an image or frame of video.

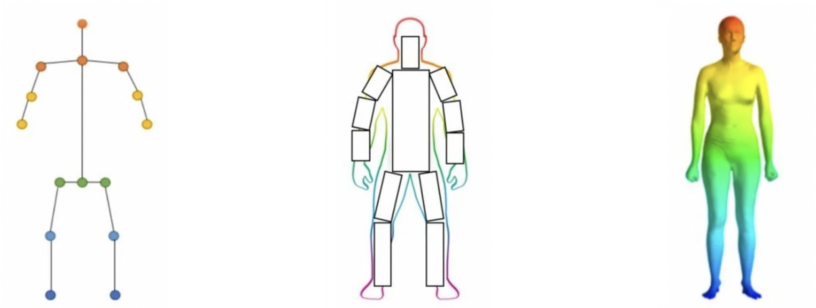
The purpose of human pose estimation is to reconstruct a human body models with some key points detected from the input data of human body. There are 3 types of human body models used as an aid in human pose estimation: kinematic model, planar model, and volumetric model as shown in Fig. 1.

***Kinematic model***, also known as a skeleton-based model, is used for 2D and 3D pose estimation. It contains less information than the other two models, as it only represents the structural information supported by human body joints. As its shortcoming, the kinematic model is not efficient for representing texture or shape information [18].

***Planar model***, or contour-based model, as the name tells, is used for 2D pose estimation. Unlike the kinematic model, this model is used to represent the human body's appearance and shape. As Fig. 1b shows, within this model, the

body parts are illustrated by several rectangles approximating the human body contours [22].

***Volumetric model***, also called volume-based model, is used to represent 3D human body with geometric shapes or meshes. The common applications of this type of model are VR game, animation production, human-machine interactive system etc.



(a) Kinematic model (b) Planar model (c) Volumetric model

Fig. 1: Three types of human body models [15].

To go through the principles and technologies of human pose estimation clearly, this part will illustrate the classic approaches and deep learning based approaches human pose estimation respectively.

### 2.1 Traditional approaches for human pose estimation

Traditional approaches for human pose estimation rely on basic computer vision algorithms such as Histogram of Oriented Gradients (HOG), Scale Invariant Feature Transform (SIFT), canny edge detector, template matching, Pictorial Structure Model (PSM) [8] etc. HOG [20] is a feature descriptor like the Canny Edge Detector. Both are used in computer vision field to detect the edges of an object in an image. HOG features are extremely popular features for human pose estimation [9], usually HOG templates containing information of various states of human body parts are learnt in advance. Li et al. (2016) [16] propose a template matching-based method to classify detected human body poses into the existed labeled action to achieve the human action recognition. However, these approaches are involved with several shortcomings, feature extraction executions are lack of accuracy and have high complexity, as those feature extraction methods are strongly influenced by the environment such as illumination, background, etc. To overcome these shortcomings, deep learning-based approaches are proposed.

### 2.2 Deep learning approaches for human pose estimation

With the rapid development of deep learning solutions for various applications, deep learning-based approaches for human pose estimation are emerging, as can be seen in Fig. 2.

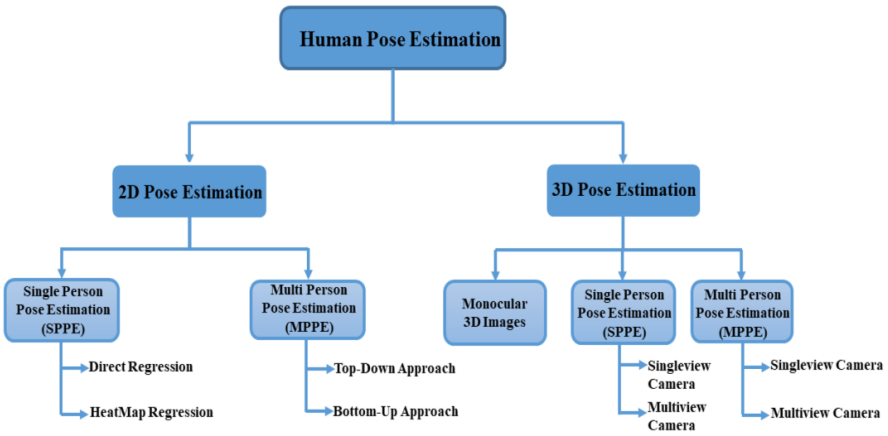
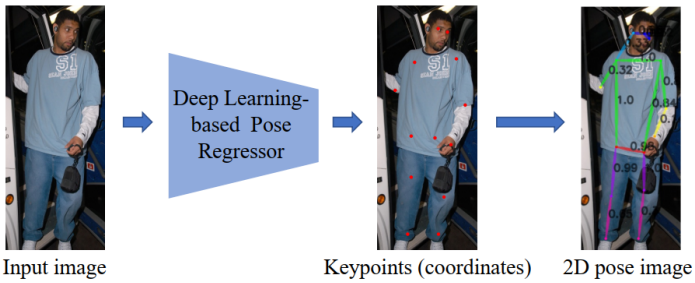


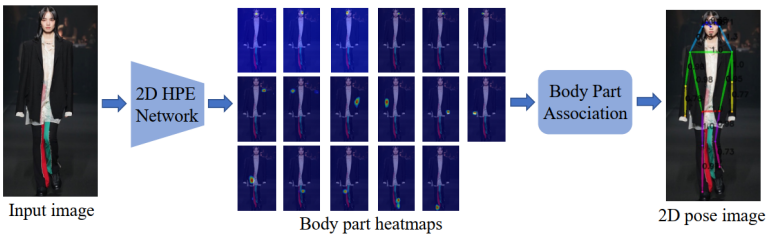

Fig. 2: An overview of deep learning-based approaches for HPE [7].

**2.2.1 2D human pose estimation.** Situations in 2D human pose estimation are single-person pose estimation (SPPE), multi-person pose estimation (MPPE). In SPPE, only one person can be detected. If there are more than one person in an image, the image is cropped until only one person in the sub-image appears before the next step. Mainly, there are two methods which are based on deep learning for 2D SPPE: regression methods and heatmap-based methods. Regression methods use an end-to-end framework to learn the mapping from the input image to body joints. Heatmap-based methods predict approximate locations of body joints, which are supervised by heatmaps representation. The working flow of these methods can be described in Fig. 3.

For 2D MPPE, there are two approaches with different working principles, the top-down approach and the bottom-up approach (see Fig. 4). Top-down method runs firstly a body detector to find individual body in an image and then estimates the body joints for each body within the detected bounding boxes. In contrast to the top-down method, the bottom-up method estimates the feature points before it can form one or more bodies with these points [18].

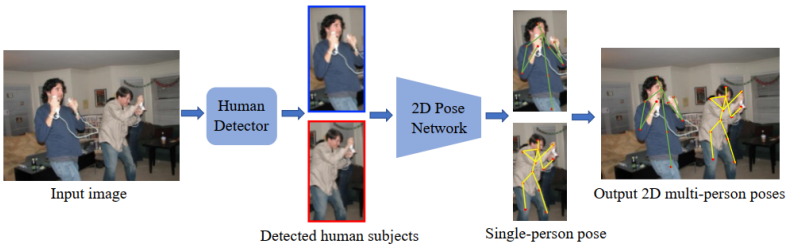



(a) Regression Methods




(b) Heatmap-based Methods

Fig. 3: 2D SPPE frameworks. (a) Regression methods build a mapping based on deep neural network between the original image and the kinematic body model. (b) Heatmap-based methods predict body joints using heatmap supervision [23].

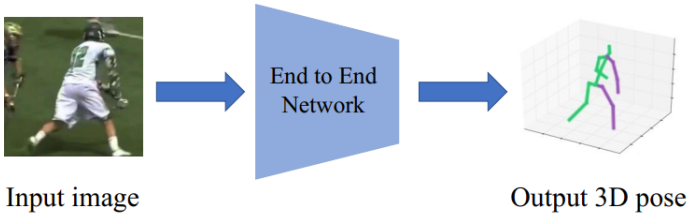



(a) Top-Down Approaches

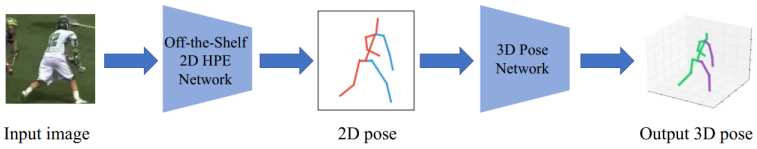

(b) Bottom-Up Approaches

Fig. 4: Two different HPE methods in Multi-person situation [23].

**2.2.2 3D human pose estimation.** 3D HPE can be divided into three solution groups: single-view single-person 3D HPE, single-view multi-person 3D HPE, and multi-view 3D HPE. According to the result to be obtained, the solutions for single-view single-person 3D HPE can be classified into skeleton-only and human mesh recovery categories. The main difference between these is that the human mesh recovery method employs a human model to get a 3D points estimation. For the skeleton result, there is a direct estimation method and a 2D to 3D lifting method. Figure 5 shows theoretically the two approaches for 3D HPE in situation of single-view and single-person. Similar to multi-person 2D HPE,




(a) Direct estimation approaches for skeleton detection




(b) 2D to 3D lifting approaches for skeleton detection

Fig. 5: Single-person 3D HPE frameworks for skeleton result. (a) Direct estimation approaches directly estimate the 3D human pose from 2D images. (b) 2D to 3D lifting approaches leverage the predicted 2D human pose (intermediate representation) for 3D pose estimation. [23]

for single-view multi-person 3D HPE, top-down approaches and bottom-up approaches are hired again. To overcome the shortcomings caused by single view strategy like: there is no detection robust when persons in images have occlusions problems, multi-view method is added to the above-mentioned approaches.

**2.2.3 Major deep learning models for human pose estimation.** Based on the approaches introduced above, many deep learning models are developed for human pose estimation in recent years. Some popular pose estimation methods based on deep learning here as examples are: OpenPose, BlazePose, DeepPose, DeepCut, PoseNet and Dense Pose.

### 2.3 MediaPipe: A Comprehensive Machine Learning Framework

MediaPipe is an open-source, cross-platform pipeline framework developed by Google. It was created to for original purpose of analyzing YouTube videos and audio in real time. Currently, this framework is still in its alpha stage and it covers multi platforms like MacOS, Windows, Android, iOS and other embedded devices like Raspberry Pi and Jetson Nano.

MediaPipe offers a range of pre-built components and tools that facilitate the development of real-time applications. These components include modules for video and audio processing, 2D and 3D perception, gesture recognition, and more. Developers can leverage these components to create custom pipelines tailored to their specific application needs.

#### 2.3.1 Architecture of MediaPipe.

MediaPipe is divided into three primary parts [17]:

(a) A framework for inference from sensory input
(b) A set of tools for performance evaluation
(c) A library of reusable inference and processing components

For a better explanation of working mode, an example of an object detection application (Fig. 6) in MediaPipe is given here, also for an understandable illustration of some basic concepts. MediaPipe enables developers to prototype a pipeline incrementally. A pipeline is a directed *graph* of components, including model inference, media processing algorithms, data transformations, etc. This component is called *Calculator* in MediaPipe. As shown in Fig. 6, each transparent rectangle stands for a **Graph**, where all processing tasks take place. The components like DetectionTracking, ObjectDetection, DetectionMerging, and DetectionAnnotation in example are **Calculators**. **Streams** connect each node in the graph to another node, which acts as carriers of **Packets**, which is the basic data unit in MediaPipe. There are so called **Side packets** which usually contain constant data for some computing situations, whereas normally streams carry data flow changing over time.

### 2.4 Skeleton Detection Principles in MediaPipe

In 2020, the MediaPipe developer, Google, presented a pose detection model called BlazePose, which is also known as the MediaPipe pose. According to the team's paper [5], BlazePose has been elaborately developed as a lightweight convolutional neural network architecture for HPE that can run on mobile devices for real-time detection purposes.

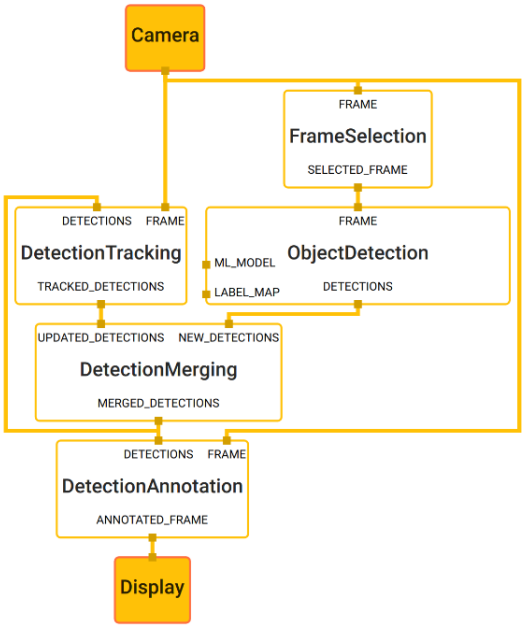



Fig. 6: Object detection using MediaPipe. The transparent boxes represent computation nodes (calculators) in a MediaPipe graph, solid boxes represent external input/output to the graph, and the lines entering the top and exiting the bottom of the nodes represent the input and output streams, respectively. The ports on the left of some nodes denote the input side packets [17].

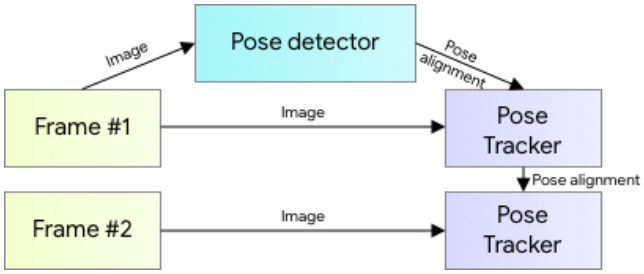



Fig. 7: Inference pipeline [5].

The model consists of a lightweight body pose detector and a pose tracker network. Each still image or video frame is taken by pose detector first for human localisation in the frame, then pose tracker network will follow to predict keypoint coordinates. For the purpose of saving computing resources of devices, the body pose detector will be called only when there is no human presence in the current frame which is reported by the face detector (see Fig. 7). From the above information, we know that BlazePose is based on a top-down HPE approach, as it detects the human body first and landmarks afterwards.

The BlazePose uses a topology (see Fig. 8) that contains 33 points to reconstruct the result of skeleton detection from each frame. BlazePose works with a neural network architecture which has been pre-trained on a large dataset consisting of 60K images with a single or few people in the scene in common poses and 25K images with a single person in the scene performing fitness exercises[5].

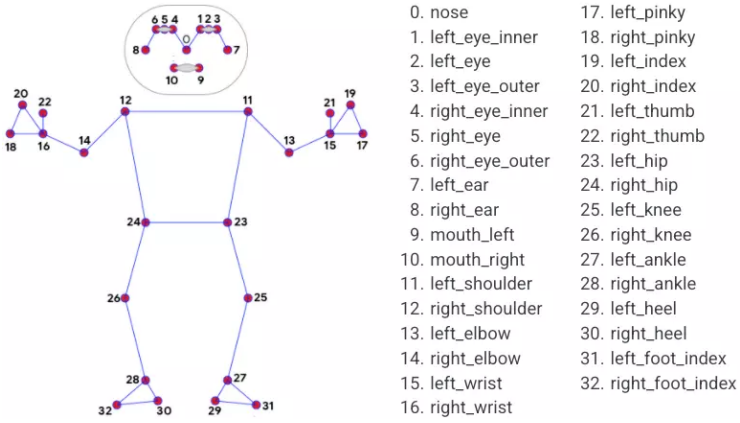



Fig. 8: Pose Landmark Model.
Image source: IT Zone [1].

### 2.5 MediaPipe Analysis: Strengths and Weaknesses

MediaPipe is a robust machine learning platform that enables the rapid development of prototype perceptual systems with inference models and other easily reusable components. A major strength of MediaPipe is its ability to provide developers with tools to build perceptual systems as a graph of modular components. These components can include inference models, media processing algorithms, and data transformations. The graph accepts input data, such as audio and video streams, and generates perceptual descriptions, such as object localisation and location streams.

MediaPipe is particularly useful for machine learning professionals, including researchers, students, and software developers [4]. They can work on implementing production-ready machine learning applications, publishing code related to research projects, and creating technology prototypes. The framework facilitates the deployment of perceptual technology in demonstrations and applications on a variety of hardware platforms, including mobile and edge devices such as Google Coral [11].

MediaPipe is characterised by its ability to address the challenges associated with adapting a perceptual application to include additional processing steps or inference models. This is often challenging due to the tight coupling of the steps [10]. By abstracting and linking individual perceptual models into sustainable systems, MediaPipe enables easy reuse of components in different systems and across different applications. In addition, MediaPipe is designed to work effectively across platforms, allowing developers to build applications on workstations and then deploy them on mobile devices, for example.

Although MediaPipe has its strengths, there are also some limitations. One is that while WebAssembly performance is generally faster than pure JavaScript, it is typically slower than native C++, which can affect the user experience [10]. MediaPipe tries to address this issue by using the GPU for image operations when possible and prefers to run as lightweight versions of all ML models as possible, although this can sometimes lead to a reduction in quality [14].

Another limitation of MediaPipe, especially its web version, is that it currently only supports calculators in sample charts and does not allow users to create their own charts from scratch, nor to add or change assets [10]. Additionally, the graph executor must be single-threaded, and TensorFlow Lite inference on GPUs is not supported. Despite these limitations, MediaPipe developers plan to continue to evolve the platform to give developers more control and potentially eliminate many of these limitations in the future [4].

### 2.6 MediaPipe Compatibility: Platforms and Languages

Primary implementation of the MediaPipe library is implemented in C++. Based on this developer decision, the library can be used among the available operating systems, which traditionally include Linux, macOS, and Windows. However, it should be mentioned that it also depends on the language because as the following text describes the various uses in specific languages, certain limitations will gradually emerge.

MediaPipe also supports Python, as it is a language widely used among machine learning experts and faces a high popularity in the developer community in general [12]. There is an API that again, as in the C++ variant, offers the possibility to use pre-made ML [2]. However, we should not neglect C++ itself, in which this

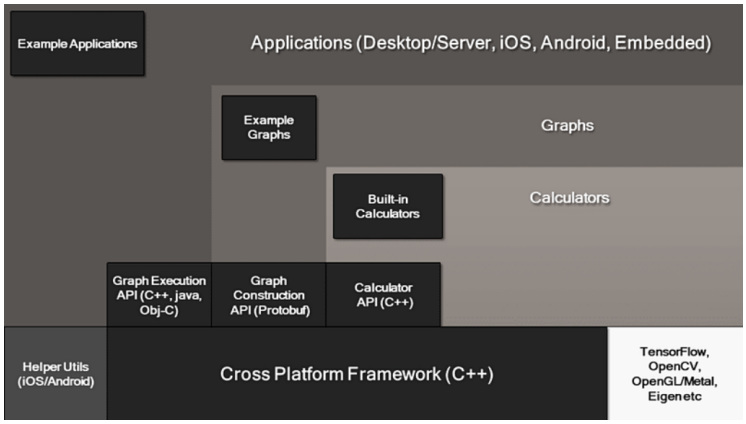



Fig. 9: MediaPipe framework architecture. Image source: Learn OpenCV [3].

library can also be used simply because it is based on this language [2]. Another well-known language supported by this library is Java and Kotlin. I deliberately mention Kotlin along with Java because these are languages that are compatible with each other and can use the libraries with each other. This is particularly useful if the implementation is orientated toward native mobile applications for android devices [11]. In the world of web services, JavaScript is also officially supported and is the main domain of all existing websites [13]. The use of this language is made possible by using the Emscripten tool, which compiles C++ into WebAssembly [13].

The C# programming language is the last language mentioned here, and this is because there is no official support from developers. Anyway, this language can still be used thanks to the C++ Wrapper for C#, which is available on GitHub as a third-party MediaPipe.NET library based on the MediaPipeUnityPlugin library, which is again a third-party library [21]. However, the actual compatibility on different operating systems ceases to be as varied as the variants mentioned above. On macOS systems it can still be used on Intel-based hardware, however for M1 processors based on a modified ARM architecture this variant is

Table 2: Compatibility of MediaPipe with different languages and operating systems.

| | Linux | macOS (Intel) | macOS (Apple Silicon) | Windows | Android | iOS |
|---|---|---|---|---|---|---|
| C++ | Yes | Yes | Yes | Yes | Yes | Yes |
| Python | Yes | Yes | Yes | Yes | No | No |
| Java/Kotlin | No | No | No | No | Yes | Yes |
| JavaScript | No | No | No | No | No | No |
| C# (non-official) | Yes | No | No | Yes | No | No |

treated as experimental and native support is non-existent. According to official information, Linux should not be a barrier for this library, nor should the latest Windows 11. It should also be noted that MediaPipe on Windows, as with macOS, is still an experimental feature with no native developer support [14, 11].

The summary of compatibility of MediaPipe is shown in Table 2.

## 3 Results

The aim of this application is to assess whether the movement of a patient contains distinct anomalies that are difficult to detect by simple observation, and it is expected that each musculoskeletal disease will have specific attributes that will allow predicting whether the patient is suffering from a particular disease or is yet to show symptoms and is at risk of a full-blown outbreak. To some extent, this feature can be seen as a predictive tool to help detect the disease before it becomes fully manifest, allowing the physician to react in time before permanent damage is done or at least to slow the disease to a point where the patient can continue to function for as long as possible.

In order to understand how this application can be used, it is necessary to briefly explain what its main functions are. First of all, this app can detect a person's skeleton from a bitmap and store this information in memory for future analysis. This image information is obtained through a camera attached to a device. In the next step, this image data is analysed and used for a teacherless AI model. The expected output is an output summary regarding the collected data that has been evaluated by the AI based on previous experiences.

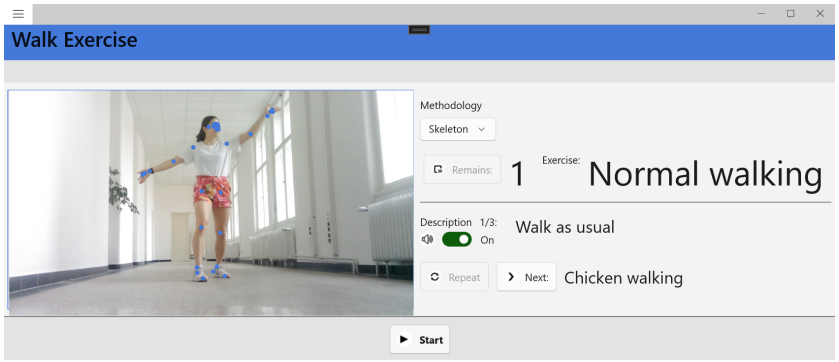



Fig. 10: Demonstration of detecting a human skeleton from an image from application
Image source: Own

Simply put, a complete model of human movement needs to be collected. Each frame will contain information about the current position of the skeleton, and this will be recorded over time. The patient being analysed will receive instructions to follow in sequence, such as walking straight from the camera to a certain distance and back. These instructions will be executed sequentially and stored by the app for later use. Once the patient has completed all the necessary steps, the analysis of these movements follows. Since this is a type of neural network learning without a teacher, it is expected to first look for various anomalies based on the previously collected information.

In a real-world setting, this application will mainly be in the hands of experts, which in this context means doctors, but there is also the option of the data evaluation taking place at the patient's home. The app will include detailed instructions on how to perform this analysis, and then all that is needed is to evaluate the collected information on a remote server, which will return the result.

## 4 Conclusion

Although this article was mainly orientated toward the theoretical level, it is also worth mentioning what the progress of this project has been so far. Currently, this application has taken a somewhat non-standard direction by choosing the Windows platform along with the C# programming language. Although a significant part of the development team is aware that this choice is not optimal in terms of official developer support, this choice was mainly dependent on the technology stack of the individual team members who are simply proficient in this language, and learning other technologies would have fundamentally slowed down the development cycle of the entire project to an unacceptable point.

In any case, the current implementation is in the testing phase, where the application handles some of the important aspects already mentioned, such as facial and skeletal analysis. It can efficiently collect image information and has implemented operations to work with this data and modify it for the core AI that can work with this information. Another important observation is that the application contains a relatively simple graphical design that is particularly suitable for workers no longer completely technically orientated, thus offering an easy to use and intuitive approach, although that it is still considering further improvements, this is for the time being at a sufficient level from a subjective point of view.

A summary of this phase of the project could be seen as a functioning ecosystem that offers medical professionals the opportunity to examine the patient on their personal PC with a connected webcam and obtain valuable data from which to evaluate the patient's condition.

### 4.1 Future Work

As MediaPipe.NET does not yet support the Android and iOS platform for pose detection (see Table 3), team will focus on implementing the necessary parts to get MediaPipe.NET to work on these platforms.

Table 3: Compatibility Matrix of MediaPipe.NET [6].

| | Linux (x86_64) | macOS (x86_64) | macOS (ARM64) | Windows (x86_64) | Android | iOS |
|---|---|---|---|---|---|---|
| Linux (AMD64) | ✓ | | | | ? | |
| Intel Mac | | ✓ | | | ? | ? |
| M1 Mac | | | ✓ | | ? | ? |
| Windows 10 (AMD64) | ✓ | | | ✓ | ? | |

### 4.2 Acknowledgement

This work was supported from the grant of Specific university research – grant No A1 FCHI 2023003 (UCT Prague) and grant No SGS 2023 016 (University of Pardubice).

# References


1. A bit about the pose classification (2023), https://itzone.com.vn/en/article/a-bit-about-the-pose-classification/
2. How to use mediapipe in c++? https://stackoverflow.com/questions/72014807/how-to-use-mediapipe-in-c (2023), [Online; accessed 07-June-2023]
3. Introduction to mediapipe (2023), https://learnopencv.com/introduction-to-mediapipe/
4. ARXIV LABS: Ar5iv: Pose tracking with 3d and rgb-d cameras. https://ar5iv.labs.arxiv.org/html/1906.08172 (2023), [Online; accessed 07-June-2023]
5. Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., Grundmann, M.: Blazepose: On-device real-time body pose tracking. CoRR abs/2006.10204 (2020), https://arxiv.org/abs/2006.10204
6. cosyneco: cosyneco/mediapipe.net: Pure .net bindings for google's mediapipe. (2023), https://github.com/cosyneco/MediaPipe.NET, přístup dne 24. června 2023
7. Dubey, S., Dixit, M.: A comprehensive survey on human pose estimation approaches. Multimedia Systems 29 (08 2022)
8. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. International journal of computer vision 61, 55–79 (2005)
9. Gong, W., Zhang, X., Gonzàlez, J., Sobral, A., Bouwmans, T., Tu, C., Zahzah, E.h.: Human pose estimation from monocular images: A comprehensive survey. Sensors 16(12) (2016), https://www.mdpi.com/1424-8220/16/12/1966
10. GOOGLE DEVELOPERS: Mediapipe on the web. https://developers.googleblog.com/2020/01/mediapipe-on-web.html (2023), [Online; accessed 07-June-2023]
11. GOOGLE DEVELOPERS: Mediapipe: Setup android. https://developers.google.com/mediapipe/solutions/setup_android (2023), [Online; accessed 07-June-2023]
12. GOOGLE DEVELOPERS: Mediapipe: Setup python. https://developers.google.com/mediapipe/solutions/setup_python (2023), [Online; accessed 07-June-2023]
13. GOOGLE DEVELOPERS: Mediapipe: Setup web. https://developers.google.com/mediapipe/solutions/setup_web (2023), [Online; accessed 07-June-2023]
14. GOOGLE DEVELOPERS: Pose landmarker. https://developers.google.com/mediapipe/solutions/vision/pose_landmarker (2023), [Online; accessed 07-June-2023]
15. Kanjee, R.: Top 9 pose estimation models of 2022 (2022), https://medium.com/augmented-startups/top-9-pose-estimation-models-of-2022-70d00b11db43

16. Li, C., Hua, T.: Human action recognition based on template matching. Procedia Engineering 15, 2824–2830 (2011), https://www.sciencedirect.com/science/article/pii/S1877705811020339, cEIS 2011
17. Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.L., Yong, M.G., Lee, J., Chang, W.T., Hua, W., Georg, M., Grundmann, M.: Mediapipe: A framework for building perception pipelines (2019)
18. Odemakinde, E.: Human pose estimation with deep learning – ultimate overview in 2023. https://viso.ai/deep-learning/pose-estimation-ultimate-overview/ (2023), [Online; accessed 18-June-2023]
19. Rastogi, K.: Know all about 2d and 3d pose estimation. https://www.analyticsvidhya.com/blog/2022/04/comprehensive-guide-for-pose-estimation/ (2022), [Online; accessed 18-June-2023]
20. Tyagi, M.: Hog (histogram of oriented gradients): An overview. https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f (2021), [Online; accessed 18-June-2023]
21. VIGNETTEAPP: Mediapipe.net. https://github.com/vignetteapp/MediaPipe.NET (2023), [Online; accessed 07-June-2023]
22. Zatolokina, L.: Human pose estimation technology capabilities and use cases in 2023. https://mobidev.biz/blog/human-pose-estimation-technology-guide (2022), [Online; accessed 18-June-2023]
23. Zheng, C., Wu, W., Yang, T., Zhu, S., Chen, C., Liu, R., Shen, J., Kehtarnavaz, N., Shah, M.: Deep learning-based human pose estimation: A survey. CoRR abs/2012.13392 (2020), https://arxiv.org/abs/2012.13392