基于视频的体育动作标准度评估算法

摘要： 本研究旨在开发一种基于视频的体育动作标准度评估算法，以解决传统姿态检测方法在设备依赖性和实时反馈方面的不足。首先本人对当前体育训练中的问题和需求进行了分析，发现传统评估方法在设备成本和实时性上存在限制。通过研究现有的计算机视觉和机器学习技术，选择MoveNet人体骨骼点检测模型作为基础框架，并结合姿态匹配技术，实现了对深蹲、引体向上动作的识别和评估。在算法实现过程中，我逐步克服了姿态匹配精度和动作实时性滞后等困难，通过不断优化算法和调整模型参数提高了计算效率，有效改善了上述问题。最终搭建了一套的动作姿态实时评分系统，能够在动作进行过程中及时给出评分和反馈。后续的测试结果表明，新开发的算法在准确性和实时性上均达到了较高水平，为体育训练提供了一种全新的高效评估方法。本研究的成果将为体育教练员、运动员和普通用户提供更加便捷有效的训练辅助工具。

关键词：计算机视觉；TensorFlow；神经网络；MoveNet；姿态检测；

Algorithm for Evaluating Sports Action Standardization Based on Video Streams

Abstract: This study aims to develop a video-based algorithm for assessing the standardization of sports movements, addressing the shortcomings of traditional methods in device dependency and real-time feedback. Firstly, an analysis of the current problems and demands in sports training was conducted, revealing limitations of traditional assessment methods in terms of device costs and real-time constraints. Subsequently, by researching existing computer vision and machine learning technologies, the MoveNet human body skeleton key point detection model was selected as the foundational framework. Combined with pose matching techniques, recognition and assessment of squatting and pull-up movements were achieved. Throughout the algorithm implementation process, challenges such as pose matching accuracy and real-time lag in movement were gradually overcome. This was achieved through continuous algorithm optimization and adjustment of model parameters, thereby enhancing computational efficiency and effectively addressing the aforementioned issues. Ultimately, a concise real-time scoring system was constructed, providing timely scoring and feedback during movement execution. Test results indicate that the new algorithm achieves high levels of accuracy and real-time performance, offering a novel and efficient assessment method for sports training. The outcomes of this study will provide coaches, athletes, and general users with a more convenient and effective training aid.

Keywords: Computer Vision; TensorFlow; Neural Networks; MoveNet; Pose Detection;

目录

摘要 i

Abstract i

[目录 iii](#_Toc163777428)

[1 绪论 1](#_Toc163777429)

[1.1 课题目的和意义 1](#_Toc163777430)

[1.2 国内外研究现状 2](#_Toc163777431)

[1.3 算法创新点 2](#_Toc163777432)

[1.4 论文章节安排 3](#_Toc163777433)

[2 技术与工具 4](#_Toc163777434)

[2.1 平台技术 4](#_Toc163777435)

[2.2 开发环境 6](#_Toc163777436)

[3 算法设计 7](#_Toc163777437)

[3.1 模型初始化 7](#_Toc163777438)

[3.2 姿态检测识别 8](#_Toc163777439)

[3.2.1 视频帧读取 8](#_Toc163777440)

[3.2.2 骨骼点识别 8](#_Toc163777441)

[3.2.3 骨骼点归一化 10](#_Toc163777442)

[3.3 测量标准建立 11](#_Toc163777443)

[3.3.1 深蹲 11](#_Toc163777444)

[3.3.2 引体向上 13](#_Toc163777445)

[3.4 姿态评估匹配 14](#_Toc163777446)

[3.4.1 相似度计算 15](#_Toc163777447)

[3.4.2 多重匹配过滤 17](#_Toc163777448)

[3.4.3 最佳匹配选择 17](#_Toc163777449)

[3.5 算法优化措施 19](#_Toc163777450)

[3.5.1 创建Ball树 19](#_Toc163777451)

[3.5.2 并行计算 19](#_Toc163777452)

[4 算法实现 21](#_Toc163777453)

[4.1 算法主流程 21](#_Toc163777454)

[4.2 姿态识别流程 21](#_Toc163777455)

[4.3 相似度评估流程 22](#_Toc163777456)

[4.4 姿态匹配流程 23](#_Toc163777457)

[5 算法函数实现 25](#_Toc163777458)

[5.1 视频处理与帧读取 25](#_Toc163777459)

[5.2 姿态估计与特征提取 25](#_Toc163777460)

[5.3 数据结构与搜索树构建 26](#_Toc163777461)

[5.4 相似度度量与匹配 26](#_Toc163777462)

[5.5 运动状态分析与计数 28](#_Toc163777463)

[5.6 运动质量评分 29](#_Toc163777464)

[5.7 主逻辑与流程控制 30](#_Toc163777465)

[6 算法测试 31](#_Toc163777466)

[6.1 环境配置 31](#_Toc163777467)

[6.2 文件目录 31](#_Toc163777468)

[6.3 操作说明 31](#_Toc163777469)

[6.4 运行实例 32](#_Toc163777470)

[6.4.1 本地视频模式 32](#_Toc163777471)

[6.4.2 摄像录制模式 34](#_Toc163777472)

[7 成果和结论 35](#_Toc163777473)

[8 致谢 36](#_Toc163777474)

[9 参考文献 37](#_Toc163777475)

[附录 39](#_Toc163777476)

# 绪论

开发与研究基于视频的体育动作标准度评估算法是一项具有重要实践意义和广泛应用前景的任务。在现代体育领域，准确评估运动员的动作标准度对于提高竞技水平、预防运动伤害以及优化训练方法至关重要。

## 课题目的和意义

随着体育科技的不断发展，运动员的训练和比赛姿态对于他们的表现和健康来说变得越来越关键。对运动动作姿态进行科学评估不仅有助于优化训练方法，预防运动伤害，也对提高运动效率和成绩至关重要。准确的姿态评估可以指导运动员进行更有效的训练，从而在竞技体育中取得更好的成绩。

同时，深度学习算法和姿态检测模型的快速发展与应用为体育运动姿态的评估提供了新的可能性。姿态检测模型能够在短时间内处理海量的运动数据，提供更加精准和全面的分析，这对于指导运动员的训练和比赛策略具有重大意义。例如，通过分析大量的运动数据，可以精确地识别出运动员在特定动作中的不足，从而提供针对性的训练建议。此外，利用姿态检测模型，配合机器学习和深度学习，可以自动化地分析运动姿态，为教练员和运动员提供即时反馈，极大地提高了训练的效率和效果。

除了在特定的竞技体育领域的应用，该课题研究对于普及体育运动和提高公众的身体健康也有积极作用：分析普通人的运动动作，不仅可以帮助他们进行更加安全有效的锻炼，还能够预防运动相关的伤害。这对于提高公众的整体健康水平和生活质量具有重要意义。

体育运动始终是现代生活的重要组成部分。开发基于视频的体育动作标准度评估应用算法，不仅对运动员和教练在高水平竞技体育领域至关重要，也对推广健康生活方式和提高公共健康水平具有深远影响。随着技术的不断发展，这一领域展现出广阔的研究和应用前景，预期将在未来带来更多的创新和突破。

从基础训练到专业运动领域，姿态评估技术的进步不仅改善了运动员的表现和健康状况，还在更广泛的社会层面产生积极影响。通过对运动员的动作进行细致分析和调整，不仅能够提高他们的竞技水平，还可以减少运动相关的潜在伤害风险。这些技术的普及应用也有助于普通民众更好地理解正确的运动姿势，并在日常生活中保持健康的身体素质。因此，姿态评估技术的发展不仅在专业体育领域发挥重要作用，也在促进社会健康和生活质量方面发挥着不可或缺的作用。

## 国内外研究现状

在国际层面上，体育动作姿态评估的研究已经取得了显著进展，特别是在运动生物力学、计算机视觉和人工智能等领域。多项研究表明，利用先进的传感器技术和高分辨率摄像头捕捉运动员的动作数据，结合人工智能，尤其是机器学习和深度学习技术，可以有效地对运动姿态进行分析和评估。

一些研究使用穿戴式传感器，如加速度计和陀螺仪，实时捕捉运动员的动作数据，这些传感器能够提供关于运动员运动速度、方向和加速度等详细信息。通过计算机视觉的应用，如采用高速摄像头捕捉运动员的动作，利用计算机视觉技术提取运动特征。这些技术能够生成运动员动作的三维重建，提供更加全面的运动分析。利用深度学习技术，特别是卷积神经网络（CNN）和循环神经网络（RNN），被广泛应用于运动数据的分析，例如对篮球、足球等运动中的动作进行深度学习分析，不仅有助于优化运动员的表现，还能有效减少受伤风险。

国内在体育动作姿态评估算法的研究方面也在逐步发展。随着科技的进步和人工智能技术的普及，越来越多的高校和研究机构开始关注运动姿态分析的重要性，并在此领域进行探索和实践。国内许多高校和研究机构已经开始布局运动生物力学和运动数据分析领域，投入资源进行相关研究。国内研究者们也在尝试将机器学习、深度学习等先进技术应用于运动姿态的分析，例如，一些研究通过分析羽毛球、乒乓球等运动的运动员姿态，以帮助提升运动员的技术水平和比赛表现，国内研究者正努力缩小与国际先进水平之间的差距。在国际和国内双重推动下，基于视频的体育动作标准度评估算法的研究和应用前景更加广阔，将为运动员、教练员以及整体社会的健康与发展带来重要的推动力。

## 算法创新点

本项目结合了先进的技术和实用的功能，为体育训练和评估提供了一种全新的方法，具有较高的创新性和实用性。本项目有多个方面的创新之处：

1. 基于视频的体育动作标准度评估：采用视频流作为输入，可同时接受视频与摄像头采集的画面，而不仅仅是传统的基于传感器的方法，有效提高可用性，因为许多人可能没有专业的传感器设备，但他们可能有摄像头或智能手机。
2. 采用新一代人体骨骼点检测模型和姿态匹配技术：该算法利用了先进的机器学习技术，使用MoveNet模型取代了传统的PoseNet模型，能够更加准确地检测和跟踪人体骨骼点，从而实现对深蹲和引体向上动作的准确识别和评估。
3. 实时评分和反馈：系统能够在动作进行过程中实时给出评分，而非是等到动作完成后才提供反馈。这种实时性可以有效帮助用户更快地调整姿势和动作，从而提高体育运动的训练效果。
4. 支持多科目切换：系统能够根据用户的需求切换识别的体育动作，包括深蹲和引体向上。该功能这增加了算法的灵活性，使其可以应用于不同类型的体育训练和评估场景。
5. 动作评分的定制化：系统提供了可调整的评分阈值参数，用户可以根据自己的需求和标准来定制评分规则。这种算法灵活性可以适应不同用户和不同训练阶段的需求。

## 论文章节安排

本文档详细地阐述了本次的研究项目，涵盖了从项目概述到深入解析的各个阶段。文档起始于绪论部分，包含了研究的起因、目标以及其重要性。接下来在“技术与工具”章节，列举了项目开发过程中所依赖的核心技术平台和环境。在“关键算法设计”章节，深入探讨了人体姿态检测、模型的预初始化过程、测量匹配标准、核心匹配算法及其优化策略等关键技术细节。

随后，“算法流程设计”部分展示了算法的基本架构以及关键模块的设计思路。“关键函数实现”则对算法中的重要函数进行了说明，包括但不限于它们的参数、功能、运行逻辑等。“算法测试章节”详细介绍了测试环境的设置、文件结构、操作指南以及实际运行示例，确保用户能够有效地理解和复现本项目中的所有工作。文档的最后，本人阐述了项目的成果和结论，总结了本研究所达成的目标以及展望。

在致谢部分，本人向所有对本次项目给予支持的个人与学校表达感激之情。参考文献中列出了在本项目中所有参考和引用的文献，为有兴趣深入了解本项目的读者提供了进一步的阅读材料。

# 技术与工具

本项目主体采用Python编写，以下为本项目所使用到的技术与工具：

## 平台技术

1. Anaconda3

Anaconda是一个用于科学计算的开源软件包管理算法和环境管理系统，由Anaconda, Inc.开发并维护。Anaconda集成了常用的Python数据科学工具和库，使得本项目中科学计算、数据分析、机器学习和大数据处理等任务运行调试更加便捷。

Anaconda的核心组件是Anaconda Distribution，它是一个包含了最新Python解释器、一系列常用科学计算库以及其他工具的集合。其中包括了诸如NumPy、SciPy、Pandas、Matplotlib等常用的科学计算和数据分析库，还囊括了用于机器学习的工具包如scikit-learn和TensorFlow等。Anaconda Distribution的一个重要特性是它自带了名为conda的包管理工具，可创建和管理不同的Python环境。这使得用户可以轻松地管理不同项目的依赖关系，并在不同项目中使用不同版本的软件包，而不会相互干扰。

1. TensorFlow

TensorFlow是由Google团队开发的深度学习框架之一，它是一个完全基于 Python 语言设计的开源的软件。TensorFlow 可以训练和运行深度神经网络，它能应用在许多场景下，比如，图像识别、手写数字分类、递归神经网络、单词嵌入、自然语言处理、视频检测等等。TensorFlow 可以运行在多个 CPU 或 GPU 上，同时也可以运行在移动端操作系统上（如Android、iOS 等），架构灵活，具有良好的可扩展性。

1. OpenCV

OpenCV (Open Source Computer Vision Library) 是一款由英特尔公司发起并维护的开源跨平台计算机视觉库。其提供了丰富的图像处理和计算机视觉功能，可广泛应用于图像处理、对象检测与跟踪、人脸识别、手势识别以及机器学习等各种视觉应用程序开发领域。OpenCV实现了多种通用算法，成为计算机视觉领域最重要的研究工具之一。

1. MoveNet

MoveNet 是由 Google 开发的实时多人姿势估计模型。该模型旨在从视频流中检测和跟踪多个人体的姿势，包括身体的关键点位置和姿态。MoveNet 基于深度学习技术，采用了轻量级的神经网络架构，可以在移动设备和嵌入式系统上实现实时的姿势估计。

该估计模型采用了深度可分离卷积（DepthwiseSeparable Convolution）和逐步扩张卷(Dilated Convolution)等操作，以减少参数数量和计算量，从而实现轻量级和高效的模型。MoveNet 模型的整体结构包括两个主要组件：移动网络（MobileNet）和动作分支（Pose orAction Branch）。MoveNet 整体的结构如上所示，Backbone 部分是比较经典的带三层 deconv 的 MobileNetv2，并且使用了残差连接来获取浅层特征。

以下图2.1为标准MoveNet模型结构图：

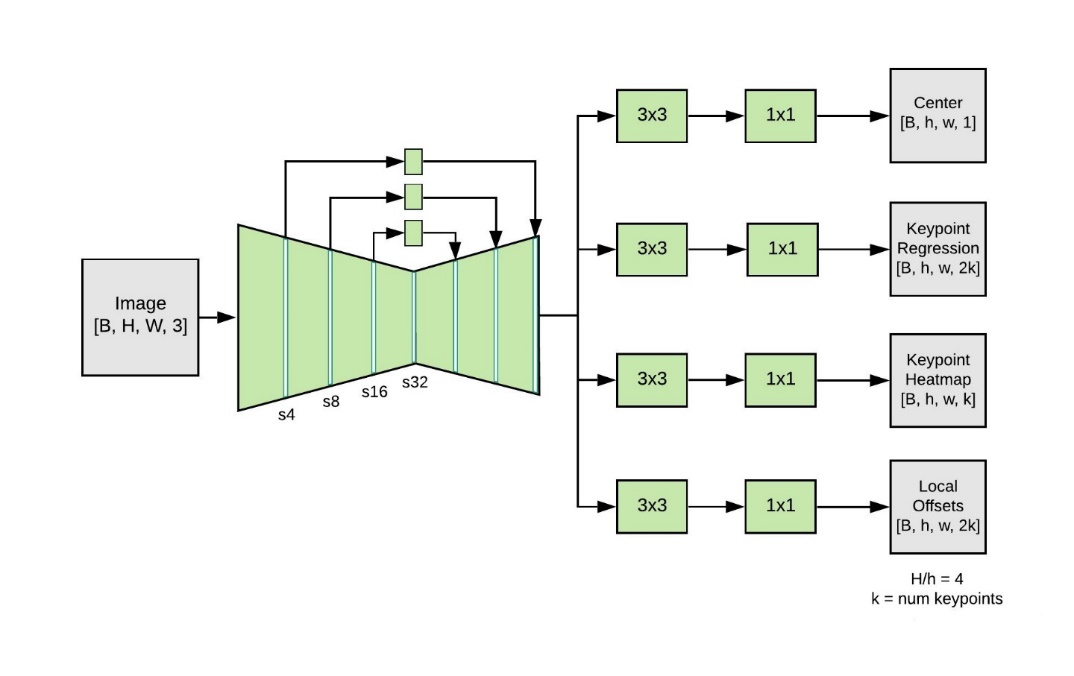


图 ‎2.1 MoveNet模型结构

其中：

* Center Heatmap[B, h, w, 1]：预测人体的几何中心，主要用于存在性检测，用Heatmap上一个锚点来代替目标检测的Box。
* Keypoint Regression[B, h, w, 2k]: 基于中心点来回归17个关节点坐标值。
* Keypoint Heatmap[B, h, w, k]：每种类型的关键点使用一张Heatmap进行检测，这意味着多人场景下这张Heatmap中会出现多个高斯核。
* Local Offsets[B, h, w, 2k]：回归Keypoint Heatmap中各高斯核中心跟真实坐标的偏移值，用于消除Heatmap方法的量化误差。

MoveNet模型主要分为两个版本，可适用于不同的识别场景，在本项目中均会采用：

1. MoveNet Thunder

MoveNet Thunder 是 Google 开发的高精度版本的 MoveNet 模型。其在保持实时性的同时，专注于提高姿势估计的准确性和稳定性。这个版本通常用于对姿势估计精度要求较高的应用场景，如医疗诊断、运动分析等。

1. MoveNet Lightning

MoveNet Lightning 则是Google 开发的轻量级版本的 MoveNet 模型。其注重模型的小型化和性能的优化，以实现在资源受限的设备上实时运行。这个版本通常用于移动端和嵌入式设备，如智能手机、智能眼镜等，以提供实时的姿势估计功能。

## 开发环境

表格2-1与2-2为本项目设计所使用的硬件环境与软件环境：

表格 ‎2‑1 硬件环境规格

|  |  |
| --- | --- |
| **类别** | **硬件规格** |
| CPU | 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz 8 Core 16 Threads |
| GPU | Nvidia GeForce RTX 3070 Laptop |
| 硬盘 | 2048 GB Samsung PM9A1 |
| 内存 | 32 GB DDR4 3200MHz |
| 摄像头1 | 640\*480分辨率内置摄像头，用于摄像录制测试 |
| 摄像头2 | 1920\*1080分辨率外置摄像头，用于采集图像数据 |

表格 ‎2‑2 软件环境规格

|  |  |
| --- | --- |
| **类别** | **软件/框架/工具** |
| 操作系统 | Windows 11 Pro 22631.3296 x64 |
| 开发环境 | Anaconda3 (Python 3.10.11) |
| 深度学习框架 | TensorFlow、Torch 2.1.2 (CUDA 11.8)、xFormers 0.0.23 |
| 编程工具 | JetBrains PyCharm Professional 2024.1、Microsoft VSCode |
| 数据分析工具 | Pandas、NumPy 等用于数据处理和分析的库 |
| 可视化工具 | Jupyter Notebook |

# 算法设计

基本检测思路：抓取当前帧的骨骼与标准库图像（base\_data）进行相似度计算，通过比对识别运动中up和down两种姿态以及关节活动幅度，给出该次运动姿态的标准度评价分数。

## 模型初始化

预设TensorFlow SavedModel文件：该文件存储了目前已训练好的 MoveNet 模型，包括模型本体与权重参数，以本项目中所用到的thunder模型为例：

* saved\_model.pb：用于存储模型的计算图结构和操作定义。该文件描述了模型的网络架构，包括各层之间的连接方式和操作的定义。在模型部署和加载过程中，TensorFlow 可根据这个文件来构建模型的计算图。
* variables 目录：用于存储了模型的权重参数。这些参数是模型在训练过程中学到的数值，在推理过程中用于计算模型的输出。权重参数的保存格式包括两个文件：.data 文件与 .index 文件。其中，.data 文件存储了模型参数的具体数值，而 .index 文件则存储了参数的索引信息，包括参数的名称和形状等。

以上三个文件和一个目录组合起来构成了一个完整的TensorFlow SavedModel文件。通过加载saved\_model.pb文件和variables目录中的权重文件来恢复模型并在姿态识别中进行推理。

设置一个MoveNet模型的Python类：用于加载和使用预训练的MoveNet模型进行单人姿势估计。

1. 定义初始化方法（\_\_init\_\_）：用于创建Movenet 类的实例时。

在初始化方法中，根据传入的模型名称，选择加载对应的MoveNet模型。如果模型名称中包含movenet\_lightning，则加载Lightning版本的模型；如果包含 movenet\_thunder，则加载 Thunder 版本的模型。

1. 定义预测方法 (predict)：用于对输入的图像进行姿势估计。

* 首先对输入的图像进行预处理，包括颜色空间转换和图像大小调整。
* 接着调用加载的模型进行推理，获取图像中人体的关键点及其置信度。
* 最后对输出进行后处理，将关键点坐标映射回原始图像坐标空间。

1. 创建main程序加载模型。

加载模型实例，并使用OpenCV库读取一张图像。并利用循环对图像进行多次姿势估计，并输出每次推理的时间。

## 姿态检测识别

使用Movenet模型姿态检测模型，对输入的视频帧进行姿态检测，获取人体关键点的位置信息。接着，通过对于人体骨骼点的二维坐标估计，从而获取骨骼坐标点以评估当前目标的运动动作姿态。

### 视频帧读取

在运动标准度检测方法中，帧读取主要负责从视频文件或摄像头中读取视频帧，并将每一帧作为输入送入Movenet模型进行姿势估计。通过以下步骤来实现：

1. 视频帧读取：使用OpenCV库中的VideoCapture模块进行视频帧的读取。首先，输入视频文件或从摄像头中捕获视频流，然后利用VideoCapture对象的read()方法逐帧读取视频。
2. 帧处理：读取到的每一帧图像会被送入Movenet模型进行姿势估计，得到人体关键点的坐标信息，用于后续相似度估计判断。
3. 循环处理：在视频流未结束的情况下，持续读取并处理每一帧图像。
4. 资源释放：循环结束后，释放视频流相关的资源。

### 骨骼点识别

1. 确定骨骼点

在人体姿态标准度识别中，肢体的协调动作可以通过人体在空间中各个关节的坐标来量化。这些坐标点（骨骼点）的集合构成了所谓的人体姿态坐标集。人体姿态估计模型的主要功能是确定某一帧图像中人物姿态的关键点在图像像素坐标系中的二维位置。

使用传统COCO人体关节点中（用于MoveNet和PoseNet），包括以下17个骨骼点：

0: 鼻子、1: 左眼、2: 右眼、3: 左耳、4: 右耳、5: 左肩、6: 右肩、7: 左肘、8: 右肘、9: 左腕、10: 右腕、11: 左胯、12: 右胯、13: 左膝、14: 右膝、15: 左踝、16: 右踝。

具体实例如图3.1所示：

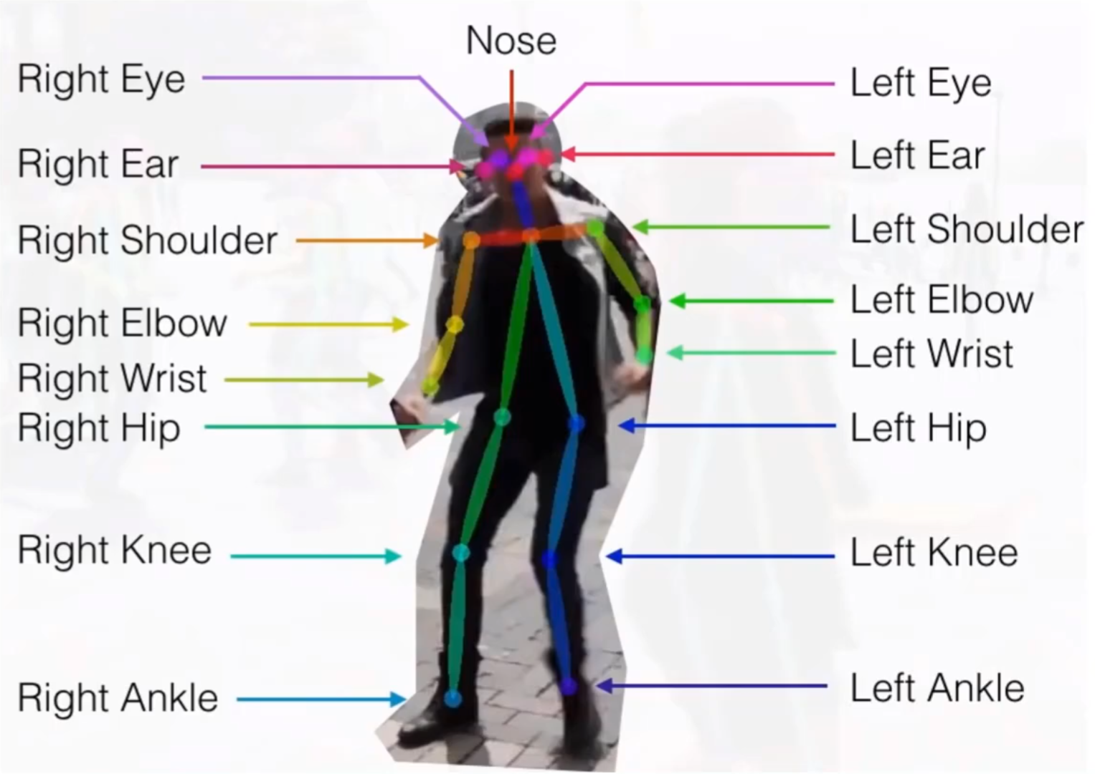


图 ‎3.1 标准17点人体姿态模型

1. 绘制骨骼点

创建姿态检测，用于在图像上绘制姿势估计结果中的骨骼点和连接线的函数。通过将上述的人体关键点和骨骼连接线绘制在图像上，可以直观地展示出人体的姿势以及关键点之间的关联。此外，通过计算关键点之间的夹角，可以量化和评估特定姿势或动作的准确性和标准度。该部分主要包含以下两个主要函数：

1. 创建plot\_skeleton函数用于绘制骨骼点和连接线，以可视化姿势估计结果。此函数接收两个参数：image（图像）和points（姿势估计结果中的关键点）。

* 将points转换为NumPy数组，并将其类型设置为整数类型。然后，它遍历每个关键点，并在图像上绘制一个小圆圈来表示关键点的位置。
* 遍历预定义的连接线索引（skeleton\_lines），并使用OpenCV的line函数绘制连接线。
* 返回处理后的图像。

1. 创建calculate\_angle函数用于计算三个给定点（关键点）之间的夹角，判定姿态类型。此函数接收三个坐标点作为参数，并计算由这三个点形成的夹角。

* 计算三个点之间的距离，并使用余弦定理计算出夹角。为了避免除零错误，需要在分母中添加了一个很小的值（1e-6）。
* 将弧度转换为角度，并返回角度值。
* 函数中的三个点是按照顺序传递的，因此假设这三个点是按照特定的顺序排列的（例如，点1、点2、点3）。

### 骨骼点归一化

对从姿态检测器得到的人体关键点坐标进行归一化处理，使得大小位置不同的人体在图像上的关键点位置具有相同的尺度和位置参考，便于模型识别的后续处理和比较。这一步骤包括将关键点坐标进行偏移和缩放，以及计算关键点的置信度和总置信度。

详细的逻辑步骤如图4.6所示：

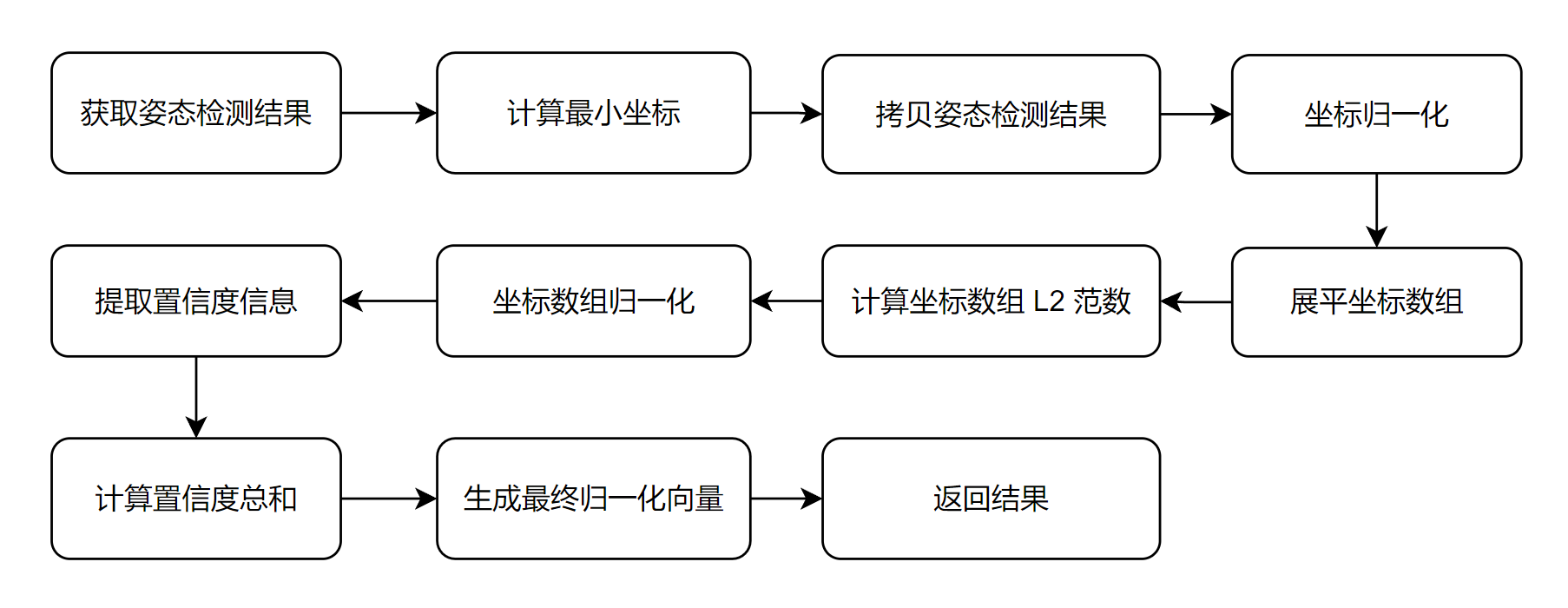


图 ‎3.2 骨骼点归一化步骤

详细操作流程步骤如下：

1. 获取关键点坐标和置信度：

从姿态检测器的结果中提取每个关键点的坐标和置信度。

1. 确定参考原点：

计算所有关键点的最小x坐标和最小y坐标，用这个最小坐标点作为参考原点，相当于将所有关键点整体移动到图像的左上角。

1. 偏移关键点坐标：

为了保护原始数据，对姿态检测结果进行深度拷贝。然后，将所有关键点的坐标减去最小坐标，实现将人体关键点移动到图像的左上角。这一步保持了关键点之间的相对位置不变。

1. 展平关键点坐标并归一化：

为便于后续处理，将关键点坐标数组展平为一维数组。然后，计算这个一维数组的L2范数（或欧几里得范数）：

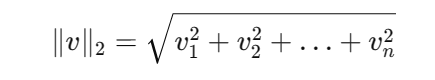


图 ‎3.3 L2范数公式

并使用此范数对坐标数组进行归一化处理，以确保所有关键点处于同一尺度：

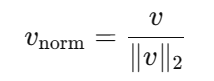


图 ‎3.4 归一化向量算式

通过这种方式，解决了图像中人体大小不一致的问题，确保了不同人体的关键点位置具有相同的尺度和位置参考。

1. 处理置信度信息：

提取出关键点的置信度信息，并计算所有关键点置信度的总和。

。

1. 形成归一化向量：

将归一化后的关键点坐标、置信度信息以及总置信度连接在一起，形成最终的归一化向量。

1. 返回结果：

将原始的姿态检测结果和归一化后的关键点向量返回，供后续处理使用。

## 测量标准建立

在这个项目中，深蹲（squat）和引体向上（pull\_up）的评分标准主要依据人体骨骼点的位置关系计算得出。对于每一种运动，项目定义了一套特定的评分机制，以量化运动的标准度或完成度。以下是对两种运动评分标准的详细解释：

深蹲与引体向上两种运动都有up（1）和down（-1）两种姿态，以及一个无法识别的姿势状态none（0）。初始状态默认是up，然后当运动者做完一次，即先后完成一次down和up，算法可给出该次评价分数。

### 深蹲

深蹲（Squat）标准姿势：在正面拍摄的视角下，运动者的足关节应与肩关节保持同宽的位置，同时髋关节应低于膝关节，形成一个标准的下蹲姿势。根据运动者臀部下沉的程度来确定，臀部下沉得越低，表示姿势越正确，得分也就越高。以下为深蹲动作的up与down两种姿态的模型参考标准图：



图 ‎3.5 深蹲动作“up姿态”参考图



图 ‎3.6 深蹲动作“down姿态”参考图

深蹲（Squat）评分标准：

深蹲的评分主要基于膝盖和髋部的相对位置关系，以及膝盖和踝关节的垂直距离比率。

评分逻辑如下：

1. 计算膝盖和髋部的中点：计算左右膝盖（关键点13和14）以及左右髋部（关键点11和12）的中点坐标。
2. 计算膝盖到髋部的垂直距离（ankle\_hip\_dist\_y）：计算踝关节（关键点15和16的中点）到髋部中点的垂直距离。
3. 计算膝盖到踝关节的垂直距离（ankle\_knee\_dist\_y）：计算踝关节到膝盖中点的垂直距离。
4. 计算距离比率并转换为分数（dist\_y\_ratio）：使用踝到膝的垂直距离除以踝到髋的垂直距离（加上一个很小的数以防除以0），得到一个比率，然后将此比率转换为分数（乘100，最高分为100分）。

### 引体向上

引体向上（Pull Up）动作标准姿势：在正面拍摄的视角下，在开始引体向上动作时，运动者的两手应略微宽于肩宽，手臂自然垂直下垂。在上拉的过程中，下巴的位置应高于手握处。引体向上动作的评估标准是根据运动者脸部点相对于小臂位置的高度来确定，脸部点越高，表示动作完成得越好，得分也就越高。以下图3.5与图3.6为引体相上动作的up与down两种姿态的模型参考标准图：



图 ‎3.7 引体向上动作“up姿态”参考图



图 ‎3.8 引体向上动作“down姿态”参考图

引体向上（Pull Up）评分标准：

引体向上的评分主要基于肘部和腕部的相对位置，以及脸部相对于这些关键点的位置。

评分逻辑如下：

1. 计算肘部的平均垂直位置：计算左右肘部（关键点7和8）的平均垂直位置。
2. 计算腕部的平均垂直位置：计算左右腕部（关键点9和10）的平均垂直位置。
3. 计算脸部中心的垂直位置：计算脸部（关键点0至4的平均位置）的中心垂直位置。
4. 根据脸部位置评分：

* 如果脸部中心位置高于肘部位置，说明运动员未能充分拉起身体，分数为0。
* 如果脸部中心位置低于腕部位置，说明运动员已完全拉起，分数为100。
* 如果脸部位置处于肘部和腕部之间，则根据脸部相对于这两个位置的比例计算分数。

## 姿态评估匹配

在本项目的运动标准度检测算法中，匹配算法是用来比较当前帧图像中的人体关键点与预先录制的标准动作关键点之间的相似程度，并找到最匹配的动作作为当前动作的评估结果。在本项目中，匹配算法的实现需要以下几个关键步骤：

1. 相似度计算：使用相似度计算函数来比较当前帧图像中的关键点向量与标准动作的关键点向量之间的相似度。在项目中，采用了余弦相似度作为相似度计算的方法。该计算过程在后续cosin\_distance\_matching函数中实现。余弦相似度的计算方法是将两个向量的点积除以它们的范数的乘积。
2. 多重匹配过滤：在计算得到相似度之后，会将匹配结果与相似度阈值进行比较。如果匹配结果的相似度低于阈值，则该匹配结果可能不够可靠，会被过滤掉。
3. 最佳匹配选择：经过相似度计算和过滤之后，会选择具有最高相似度的标准动作作为当前帧图像的最佳匹配。这个过程会在find\_best\_match函数中实现。
4. 匹配结果返回：最终的匹配结果将包括匹配到的标准动作的序号和相似度值。这个结果可以被用来评估当前动作的标准度，并根据需要进行相应的处理。

匹配算法能够快速而准确地找到当前帧图像中与标准动作最相似的动作，并将其作为当前动作的评估结果，从而实现了对运动标准度的检测与评估。

### 相似度计算

在本运动标准度检测方法中，相似度计算是一个重要步骤，其用于比较当前帧图像中检测到的人体关键点与预先录制的标准动作关键点之间的相似程度。在本项目中，该相似度计算采用了两种主要的数学方法：余弦相似度和欧式距离。这两种方法可分别从方向相似性和位置差异两个维度对姿态进行评估。图3.7为余弦欧氏距离图示：

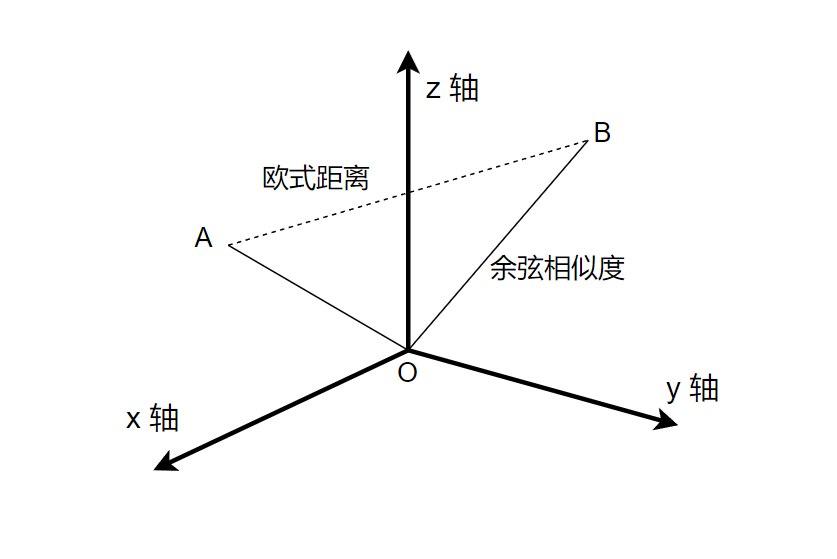


图 ‎3.9 三维空间中余弦相似度和欧式距离

以下为详细描述：

* 余弦相似度：

在本项目中，使用向量空间中两个向量夹角的余弦值作为衡量两个个体间差异大小，用于运动员运动姿态与标准姿态的相似度和差异，同时修正不同运动员之间由于体型等原因产生的度量标准不统一的问题。取值范围在[-1, 1]之间，值越接近1表示两个向量越接近，越接近-1表示两个向量越远离，而0表示两个向量之间的夹角为90度。余弦相似度可以由以下公式表示：

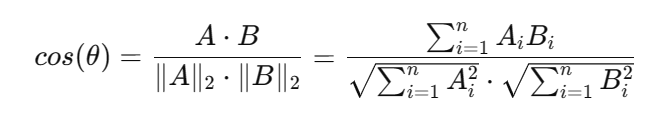


图 ‎3.10 余弦相似度计算公式

其中，*A*和*B*分别是两个向量，*Ai*​和*Bi*​是向量的第*i*个坐标。

同时，定义基于余弦相似度的距离度量，用于衡量两个向量之间的相似性。基于余弦相似度的距离度量由以下公式表示：

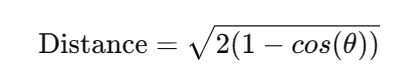


图 ‎3.11 基于余弦相似度的距离度量计算公式

* 欧式距离：

在本项目中，使用向量空间中两个向量终点间的绝对距离衡量两个个体间差异大小。 用于比较同一个运动员运动姿态的相似度或差异。欧式距离可以由以下式算式表示：



图 ‎3.12 欧氏距离计算公式

其中， *A*和*B*为同维度向量，*Ai*​ 和 *Bi*​是向量的第*i*个坐标。

* 关键点向量表示：

在本项目中，人体关键点被表示为一个向量，其中包含了关键点的坐标信息。该向量会忽略置信度和其他附加信息，只关注关键点的位置信息。

计算实现过程：

1. 关键点向量提取：从当前帧图像中提取的关键点向量中截取出关键点的坐标部分，作为输入的姿势向量。
2. 余弦相似度计算：将标准动作的关键点向量与当前帧图像中提取的关键点向量进行余弦相似度计算。
3. 欧式距离转换：通过余弦相似度计算得到的相似度值，转换为欧式距离作为最终的相似度度量值。这里采用欧式距离的平方根作为最终的相似度值，用于衡量当前姿势与标准动作之间的差异程度。

计算得到的相似度值将被用于后续的匹配，用于找到当前帧图像中与标准动作最相似的动作。这样就能够通过比较当前动作与标准动作之间的相似度，来评估当前动作的标准度。

### 多重匹配过滤

多重匹配过滤确保了从一系列可能的匹配结果中筛选出最可信的匹配。这个过程涉及将每个匹配结果的相似度与预设的阈值进行比较，从而决定哪些结果足够接近被视为有效匹配。以下是本项目中采用的两种过滤方式：

#### 相似度阈值过滤

为确保匹配结果的准确性，本项目中引入了相似度阈值（match\_dist\_thresh）。该阈值用于筛选出与标准动作相差较大的匹配结果，即当算法计算出的相似度值低于此阈值时，相应的当前帧将被认定为不匹配并排除。阈值设置具有一定的灵活性，可根据实际需求调整以优化运动标准度检测效果。

在本项目中，该阈值定为0.3，意味着仅当相似度超过0.3，结果才被视为有效，否则将予以忽略。实验表明，相似度极限值为0.15，在此水平下视频识别能正常进行，但通过摄像头捕获的识别效果明显下降。经过反复测试，确定将匹配相似度阈值设定为0.3，以达到最佳匹配效果。

#### 视频帧抖动过滤

由人体运动或摄像机移动引起的画面抖动现象，会对匹配结果造成干扰。为解决此问题，本算法中采用了连续帧匹配与累积平均策略，以降低抖动对结果的影响，增强匹配的稳定性和精确度。

以本项目中维护的一个姿态类别频次队列（其中“key”代表姿态类别，“value”代表该姿态检测到的次数）举例：

输入原始队列：[[up, 3], [down, 15], [up, 15], [down, 1], [up, 10], [down, 10], [up, 20]]。

通过将视频帧抖动过滤阈值设置为10，即“value”需大于或等于10才予以保留。应用此规则后，队列更新为：[[down, 15], [up, 15], [up, 10], [down, 10], [up, 20]]。

通过以上操作，可有效过滤掉频次较低的抖动引起的异常帧，提高算法分析结果的准确性和可靠性。

### 最佳匹配选择

经过上述的相似度计算和多重匹配过滤，接下来的步骤是从所有有效的匹配结果中选择一个最佳的匹配。在本项目中，最佳匹配选择的实现依赖于以下策略：

1. 连续帧的处理

通过引入frame\_reader函数，以时间序列的方式使算法逐帧读取视频数据。每一帧都被视为时间序列中的一个独立观测点，承载着关于动作和姿态的瞬时信息。通过将这些连续帧放入input\_queue队列，系统不仅保持了帧之间的时间顺序，而且为后续的并行处理和动作分析提供了数据基础。这种方法允许系统在保留动作时间连续性的前提下，对每一帧进行进一步分析。

1. 动作频次加权

通过increament\_squat\_count和count\_total\_reps函数，算法能有效地识别并累积特定动作在一系列连续帧中的出现频率。这种动作频次加权机制不仅区分了单一动作与连续动作序列，还对动作的连续性进行了精确量化，为动作的持续性和频率提供了直观的量化指标。

1. 姿态变化的连续性考虑

通过设置find\_best\_match、filter\_match等函数，使算法系统对每一帧中的姿态与一组预定义的姿态模板进行匹配，并根据匹配结果进行动作评分。这一过程不仅涉及到对单帧内姿态的分析，还包括了对帧与帧之间姿态变化连续性的考量。系统通过分析姿态的变化规律，能够识别出动作的起始、执行和结束阶段，进而对动作的完整性和流畅性进行评价。

一个完整的深蹲动作不仅需要识别出深蹲的最低和最高点，还要分析达到最低点前后姿态变化的连续性，以确保动作的标准性和一致性。以连续深蹲为例，算法通过累积统计，提供了对动作频次的直观量化表达。在过滤处理之后，若输入序列为[[up, 10], [up, 15], [up, 20], [down, 11], [down, 13], [down, 20], [up, 20]]，算法将执行连续动作的合并处理，将该序列优化为[[up, 45], [down, 44], [up, 20]]。此过程不仅使得动作序列更加连贯，也极大增强了算法对于目标动作模式的识别能力。

1. 加权距离匹配

为了精确评估两组关键点之间的相似度，本项目引入了加权距离匹配机制。通过 weighted\_distance\_matching 函数，算法计算了考虑每个关键点置信度的加权距离。这种方法不仅评估了关键点位置的差异，也根据每个关键点的置信度赋予不同的权重，从而提高了匹配的准确性和鲁棒性。加权距离的计算公式如下：

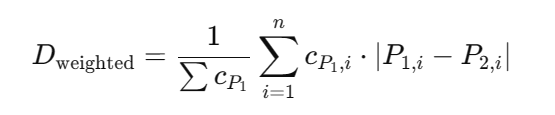


图 ‎3.13 加权距离匹配公式

其中，*cP*1​,*i*​是*P*1​中第*i*个关键点的置信度，*P*1,*i*​和*P*2,*i*​分别是两组关键点中第*i*个关键点的位置坐标，∑*cP*1​​是*P*1​中所有关键点置信度的总和。

## 算法优化措施

为了提高算法效率，本项目中采用了以下优化措施，包括但不限于数据结构优化、并行计算与多线程：

1. 数据结构优化：为了快速检索标准动作关键点数据，项目中使用了基于Ball树的数据结构。通过构建 Ball 树，可以将标准动作关键点数据进行组织和索引，以便快速地在实时帧图像中进行匹配。
2. 并行计算与多线程：在帧读取和关键点检测阶段，项目采用了多线程的方式来提高效率。通过多线程同时处理视频帧和关键点检测，可以减少处理时间，从而提高算法的整体性能。

### 创建Ball树

本项目中使用到了Ball树（Ball Tree），主要是为了高效地处理和查询高维数据点之间的相似度或距离，尤其是在进行最近邻搜索时。该数据结构可以通过高维空间效率、复杂距离计算、动态插入和删除、相似度搜索来显著提高搜索效率，减少计算时间，特别是在项目中这种需要快速比较人体姿态向量的情况。

在本项目中的应用：

在本项目中，需要用到Ball树来优化查找与当前捕捉到的姿态最相似的历史姿态的过程。例如，通过预先计算并存储一系列标准姿态的特征向量，然后构建一个Ball树来组织这些数据。当实时处理视频流并提取出当前帧的姿态特征向量时，使用Ball树快速找到最相似的标准姿态，以此来判断当前的运动状态和标准度，从而为用户提供即时的运动质量反馈。

### 并行计算

本项目中使用到了多线程来实现并行计算，主要是用于提高视频处理的效率和响应速度。在人体姿态估计和运动标准度检测这样的应用中，处理视频流数据是计算密集型的任务，特别是当涉及到实时或近实时的视频分析时。多线程可有效地利用计算资源，提高应用的性能。以下是多线程在本项目中的应用：

* 非阻塞视频帧读取：通过将视频帧的读取操作放在一个单独的线程中，可以确保即使帧的处理比读取更耗时，视频捕捉也不会因此被阻塞，可连续不断地从视频源（如摄像头或视频文件）获取帧，而不会错过任何帧。
* 并行处理视频帧：在另一个或多个线程中进行视频帧的处理（如人体姿态估计、运动状态判断等）可以与帧的读取并行进行。当一个帧正在被处理时，另一个线程可同时读取下一帧，从而提高整体处理速度。
* 改善用户界面响应：在主UI线程之外使用工作线程来处理耗时的任务可避免检测预览界面冻结或变得不响应。用户可以继续与预览界面互动，同时后台线程处理可更加复杂的计算任务。

在项目中的实现：

本项目中使用了threading.Thread来创建新线程，用于视频帧的读取。本项目中，frame\_reader函数被设计为在一个单独的线程中运行，从视频源异步读取帧，并将它们放入一个线程安全的队列（queue.Queue）中。这样，处理线程可以从队列中获取帧并进行进一步处理，而不会直接影响到帧的读取过程。

以下是在本算法中一段使用Python编写的示范代码：

* t0 = threading.Thread(target=frame\_reader, args=(video\_path, input\_queue))
* t0.start()

通过以上方式，本项目能够实现高效的视频处理流程，提高对动态视频数据的处理能力，特别是在需要实时反馈的应用场景中。使用多线程是一种常见的方法来提高计算密集型应用的性能，尤其是在本项目中使用的多核处理器架构下。

# 算法实现

本项目是一个基于视频的体育运动标准度检测算法，通过MoveNet估计模型，实时姿势检测和相似度匹配来评估运动动作的标准度，并提供相应的评分和统计信息。

## 算法主流程

以下为本项目的算法基本流程图：

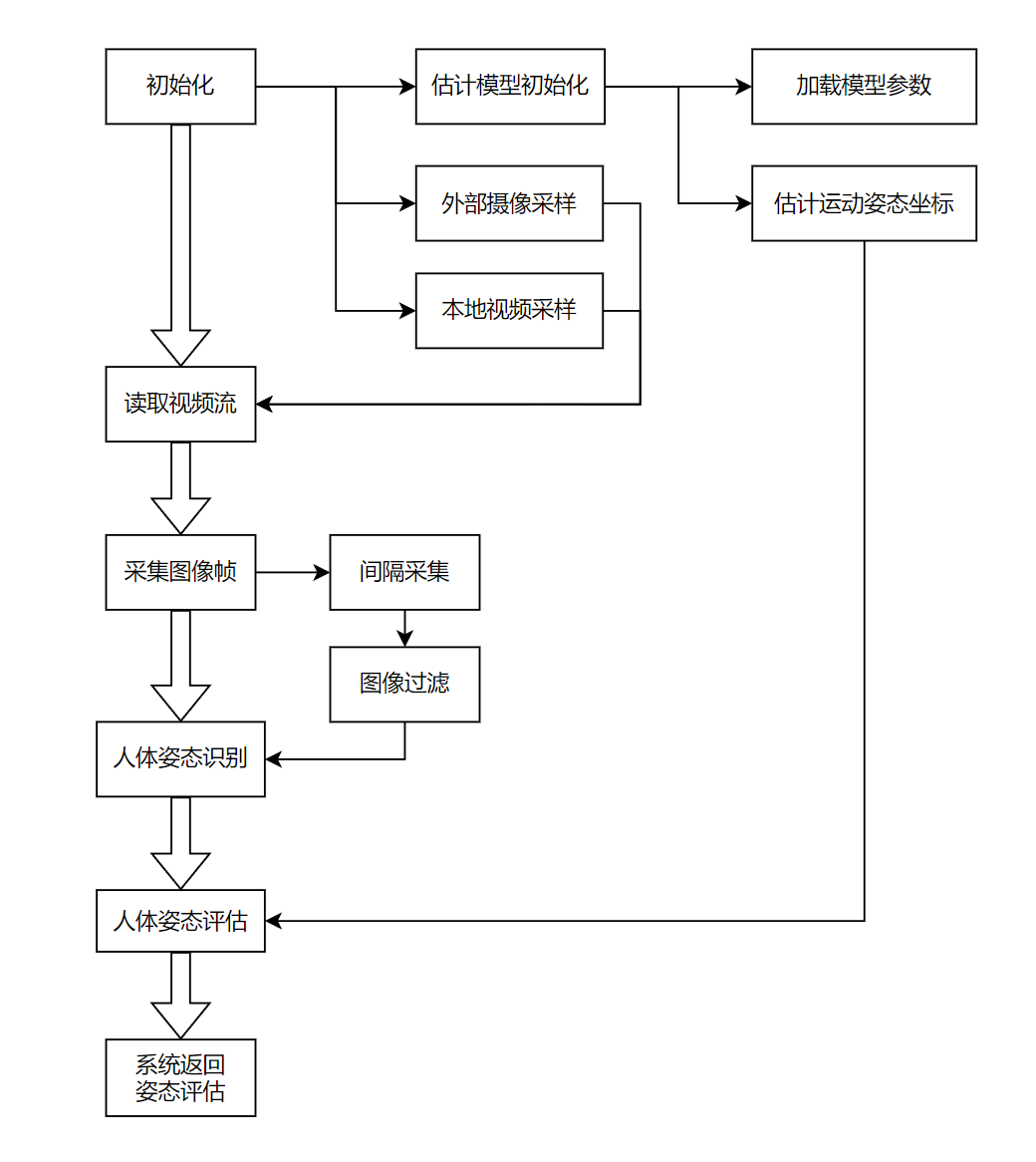


图 ‎4.1 算法主流程图

## 姿态识别流程

以下为本项目中的人体姿态识别流程图：

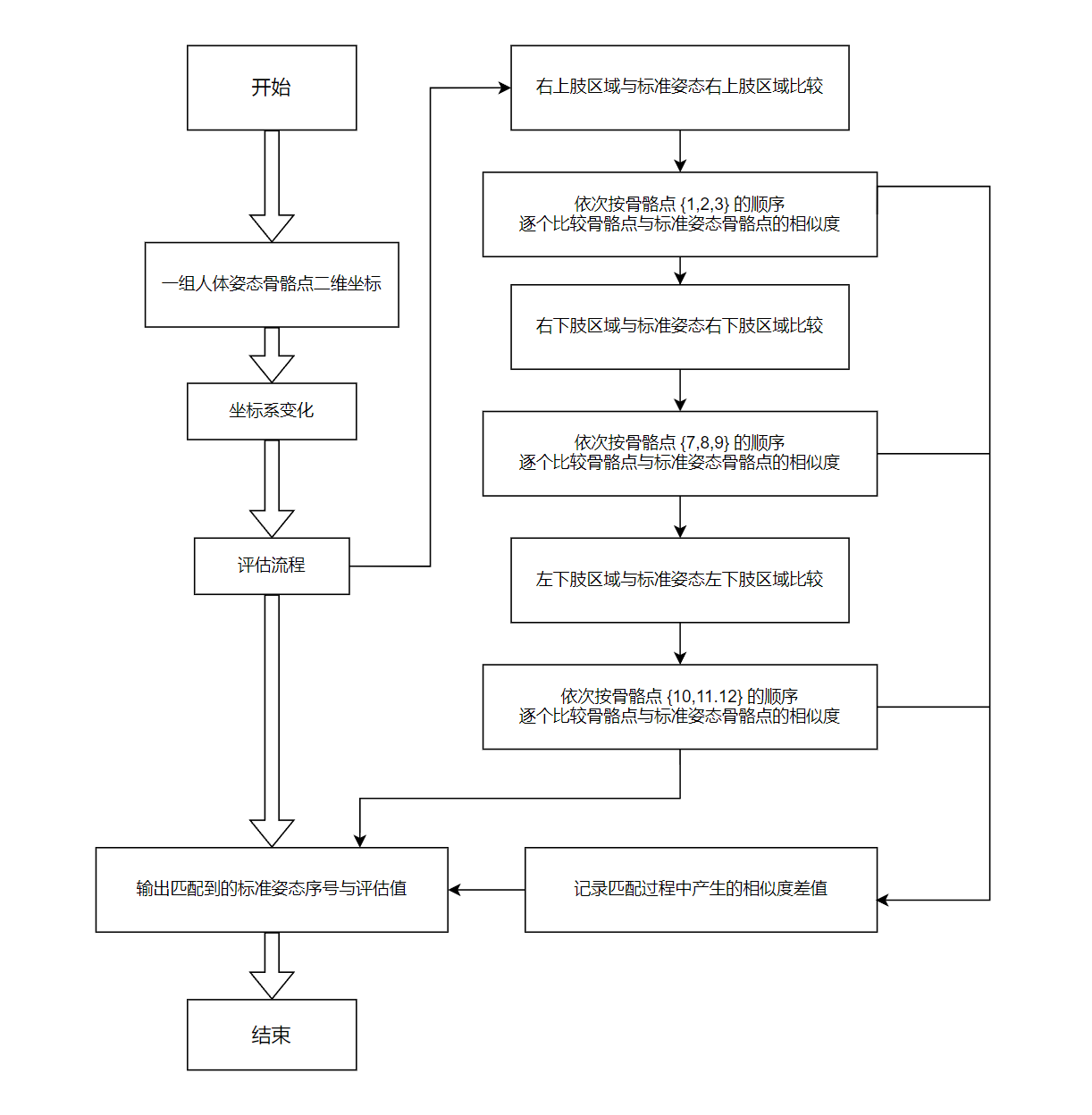


图 ‎4.2姿态识别流程图

## 相似度评估流程

以下为本项目中的人体相似度估计流程图：

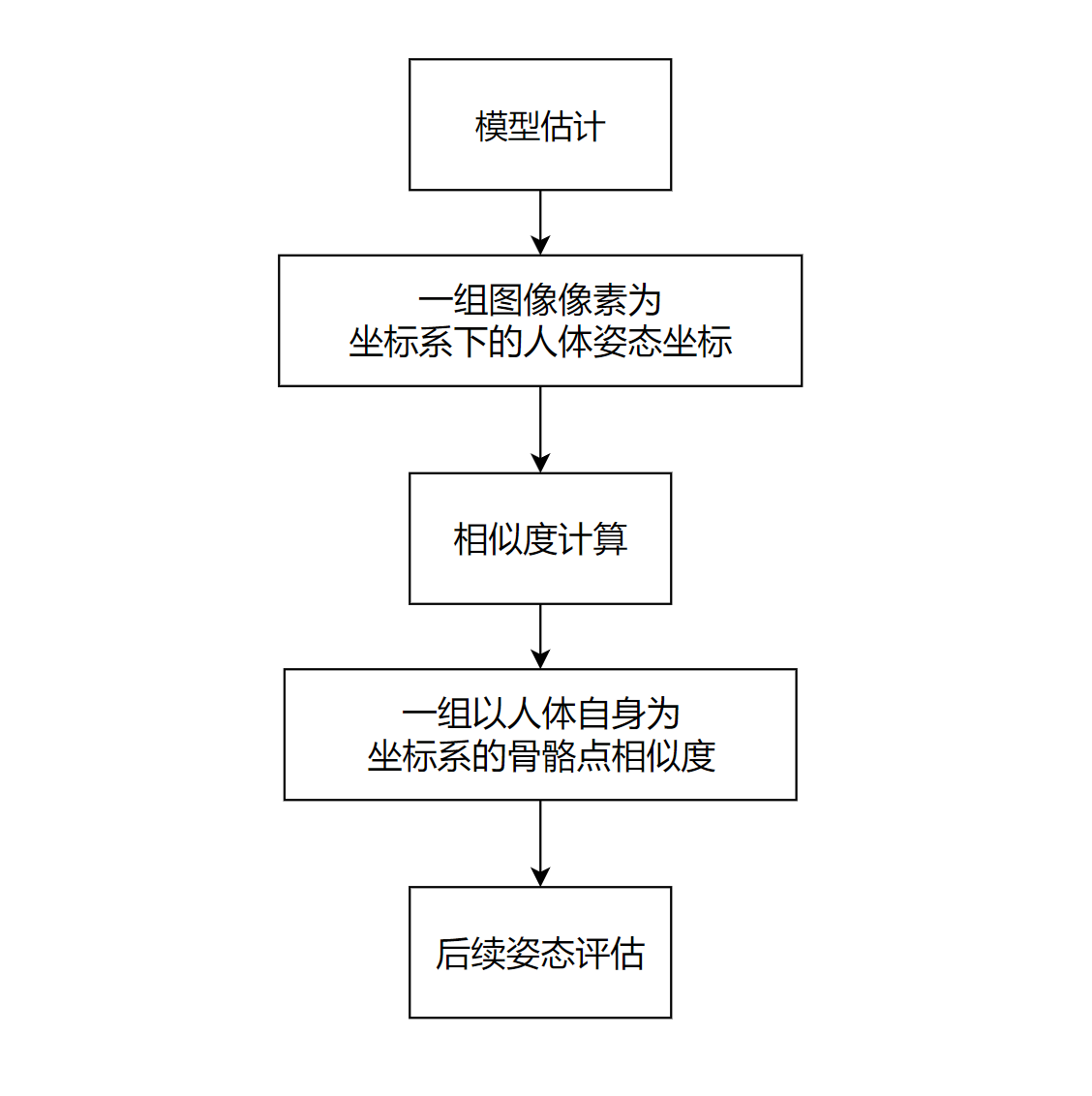


图 ‎4.3 相似度评估流程图

## 姿态匹配流程

以下为本项目中的人体姿态匹配流程图：

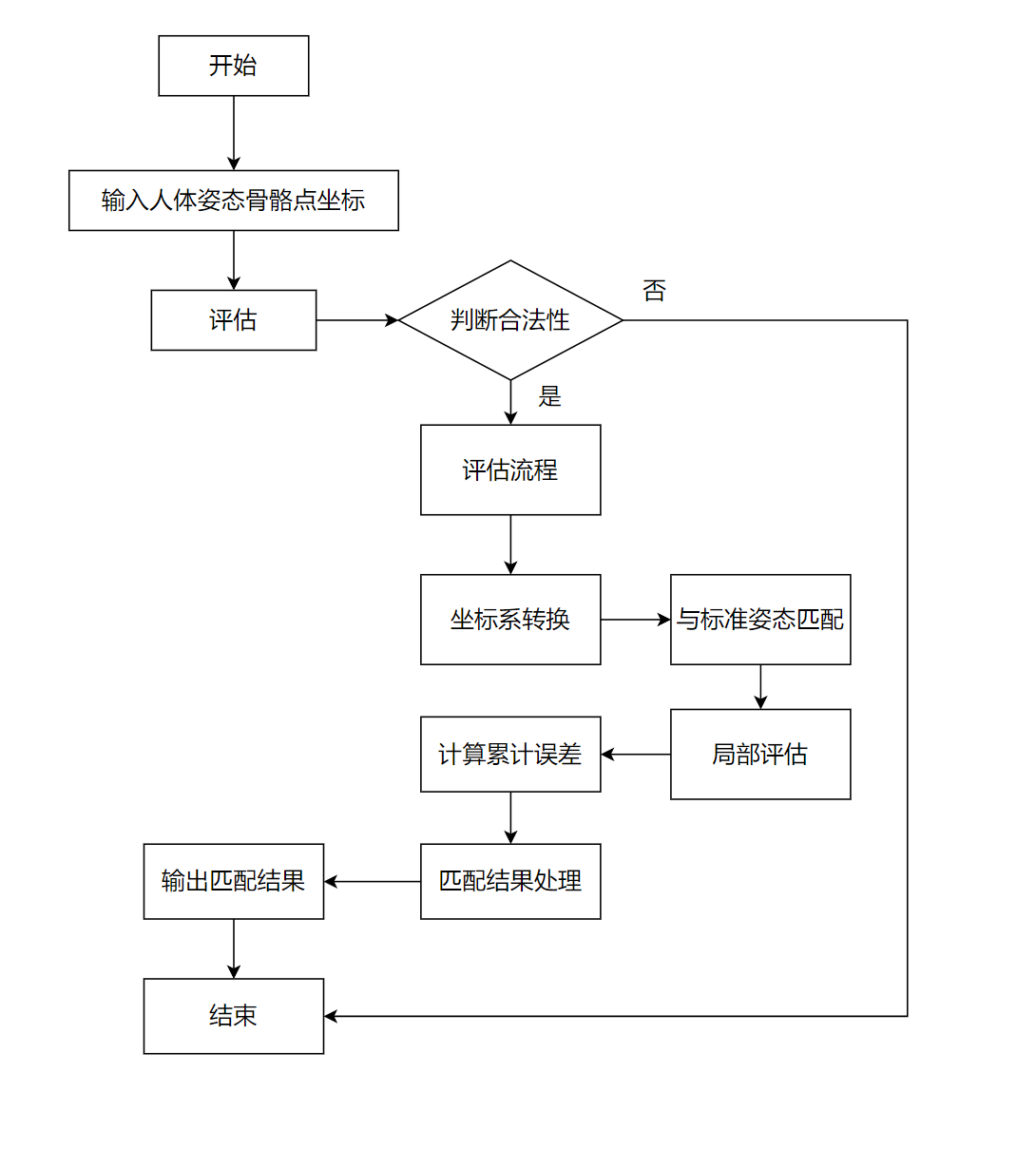


图 ‎4.4 姿态匹配流程图

# 算法函数实现

本项目为基于视频的运动动作检测算法，可以通过摄像头或者本地视频文件输入，实时识别人体姿态，并根据预定义的基准数据进行匹配，最终给出运动动作的评分。以下是在本项目中实现的关键函数（已按照函数功能进行了模块分类），包括函数功能、参数说明、运行流程、函数调用等内容的简要说明：

## 视频处理与帧读取

frame\_reader(video\_path, input\_queue):

1. 函数功能：从视频源（本地文件或摄像头）异步读取每一帧图像，并将帧编号与图像数据放入input\_queue中。
2. 参数说明：

* video\_path: 视频源的路径。本地文件的路径或摄像头索引（0表示默认摄像头）。
* input\_queue: 一个线程安全的队列，用于存储读取的视频帧和它们的编号，以供后续处理。

1. 接口说明：

* init\_video\_capture(source, frame\_rate): 初始化视频流，设置视频源和帧读取频率。
* tart\_video\_stream(): 开始视频流读取。
* read\_frame(): 读取视频帧并存入队列。

1. 运行流程：

此函数在一个独立的线程中运行，函数循环读取视频帧，直到视频结束或外部标志（frame\_reader\_flag）被设置为False。对于每一帧，它将帧编号和帧数据作为列表项放入队列中。如果读取失败（视频结束），则放入[-1, None]标识结束。

1. 函数调用：在main()函数中，通过创建新线程启动此函数，以实现视频帧的并行读取。

## 姿态估计与特征提取

1. get\_pose\_vector\_from\_one\_file(detector, image\_path):
2. 函数功能：读取单个图像文件，使用姿态检测器提取姿态向量。
3. 参数说明：

* detector: 姿态检测模型的实例，用于对输入图像进行姿态估计。
* image\_path: 要处理的图像文件的路径。

1. 运行流程：

读取指定路径的图像文件，使用detector对其进行姿态估计，然后提取并返回归一化的姿态向量。

1. 函数调用：在构建基础数据集姿态向量时调用。
2. get\_pose\_vector\_from\_pose\_detector(results, height, width):
3. 函数功能：从姿态检测结果中提取并归一化姿态向量。
4. 参数说明：

* results: 姿态检测模型的输出，通常包含了图像中所有检测到的关键点及其置信度。
* height: 输入图像的高度。
* width: 输入图像的宽度。

1. 运行流程：

该函数将姿态检测结果转换为归一化的姿态向量，便于后续的处理和比较。归一化过程通常包括调整关键点坐标的尺度，使之与图像尺寸无关。

1. 函数调用：处理视频帧以获取姿态数据时调用。

## 数据结构与搜索树构建

make\_search\_tree(detector, base\_data\_path, action\_image\_num, metric):

1. 函数功能：创建一个Ball树，用于高效的姿态匹配。
2. 参数说明：

* detector: 姿态检测模型的实例。
* base\_data\_path: 基础数据集的路径。
* action\_image\_num: 每种动作类别中的图像数量。
* metric: 用于Ball树中的度量函数，如余弦相似度或加权距离函数。

1. 接口说明：

* load\_search\_tree(tree\_path): 加载基准数据的搜索树。
* match\_and\_score(pose\_vector): 匹配姿态向量并计算评分。

1. 运行流程：

根据基础数据集中的图像，使用detector提取姿态向量，然后基于这些向量和指定的metric构建一个Ball树。这个树之后用于快速查找与给定姿态最相似的基础姿态，加载预先构建好的基准数据的搜索树结构。对于每一帧的姿态向量，利用搜索树进行快速匹配，找到最相似的基准数据。通过计算姿态向量之间的相似度，得出匹配结果。最后根据相似度结果给出运动动作的评分，并将结果输出。

1. 函数调用：在main()函数中调用，初始化时构建用于姿态匹配的搜索树。

## 相似度度量与匹配

1. cosin\_distance\_matching(pose\_vector1, pose\_vector2):
2. 函数功能：计算两个姿态向量之间的余弦距离。
3. 参数说明：

* pose\_vector1: 第一个姿态向量。
* pose\_vector2: 第二个姿态向量。

1. 运行流程：使用余弦相似度公式计算两个向量的相似度，然后根据相似度计算距离。余弦距离越小，表明两个姿态越相似。
2. 函数调用：在进行姿态匹配时调用，在make\_search\_tree函数构建搜索树时指定度量标准。
3. weighted\_distance\_matching(pose\_vector1, pose\_vector2):
4. 函数功能：根据置信度权重计算两个姿态向量之间的加权距离。
5. 参数说明：

* pose\_vector1: 第一个姿态向量。
* pose\_vector2: 第二个姿态向量。

1. 运行流程：

根据关键点的置信度对两个姿态向量之间的距离进行加权计算，得到一个考虑置信度的加权距离。这种方法在比较姿态相似度时能够给予置信度高的关键点更大的权重。以下是

1. 函数调用：此函数提供另一种姿态匹配度量，用于度量测试。
2. find\_best\_match(pose\_vector, tree):
3. 函数功能：在Ball树中找到与给定姿态向量最相似的项。
4. 参数说明：

* pose\_vector: 待匹配的姿态向量。
* tree: 构建好的Ball树，用于高效地进行最近邻搜索。

1. 运行流程：使用Ball树执行最近邻搜索，找到最匹配的姿态向量。
2. 函数调用：处理每一帧时调用，以找到与当前姿态最相似的基础姿态。
3. filter\_match(match, dist\_thresh, action\_image\_num):
4. 函数功能：根据相似度阈值过滤匹配结果。
5. 参数说明：

* match: 包含匹配结果的字典，通常有index和dist两个键。
* dist\_thresh: 相似度距离的阈值，用于判断是否接受匹配结果。
* action\_image\_num: 动作图像的数量，用于调整匹配逻辑。

1. 运行流程：

根据dist\_thresh评估match中的相似度距离，如果距离大于阈值，则认为没有找到有效匹配，相应调整匹配结果。

1. 函数调用：在找到最佳匹配后调用，用于确定匹配是否足够好。

## 运动状态分析与计数

1. increament\_squat\_count(pose\_counter, pose\_category):
2. 函数功能：根据当前姿态类别更新姿态计数器。
3. 参数说明：

* pose\_counter: 一个列表，用于跟踪连续出现的姿态类型及其次数。
* pose\_category: 当前帧的姿态类别（如"up"或"down"）。

1. 运行流程：更新一个队列，记录连续出现的姿态类型及其次数。
2. 函数调用：在确定当前帧的姿态类别后调用，用于统计运动次数。
3. count\_total\_reps(count, pose\_counter, pose\_num\_1rep\_thresh, category\_num, verbose=False)
4. 函数功能：计算总的运动次数。
5. 参数说明：

* count: 当前的运动次数，即已经完成的动作循环数。
* pose\_counter: 一个列表，每个元素是一个包含姿态类别（如"up"或"down"）和相应出现次数的列表。这反映了一个运动周期内不同姿态的出现频率和顺序。
* pose\_num\_1rep\_thresh: 一个阈值，用于确定某个姿态出现的次数是否足以被计为一个完整的动作。这有助于过滤掉短暂的或偶然的姿态变化，确保只有完成的动作才被计数。
* category\_num: 动作循环中需要识别的姿态类别数。例如，在深蹲中，可能需要识别"up"和"down"两个类别，因此此参数为2。
* verbose: 一个布尔值，用于控制函数是否打印出额外的信息，以便于调试和分析。默认值为False。

1. 运行流程：

根据姿态计数器中的数据过滤并统计完成的运动次数。

* 函数首先根据pose\_num\_1rep\_thresh过滤pose\_counter中的数据，移除那些出现次数不足以构成一个完整动作的姿态记录。
* 然后，根据过滤后的pose\_counter内容，计算实际完成的动作次数。这涉及到确定连续的"up"和"down"动作对，并确保它们满足动作定义的标准。
* 如果识别到新的完整动作，count会递增，并将update\_score\_flag设置为True以指示需要更新得分。

1. 函数调用：在一系列姿态判断完成后调用，用于确定运动者完成了多少次完整的运动。

## 运动质量评分

1. update\_best\_squat\_score(pose, best\_squat\_score):
2. 函数功能：更新深蹲的最佳得分。
3. 参数说明：

* pose: 当前帧检测到的姿态数据，通常是一个包含人体各关键点坐标和置信度的数组或列表。
* best\_squat\_score: 目前为止记录的最佳深蹲得分，得分的计算基于姿态的质量，如姿势的深度、对称性。

1. 运行流程：

根据当前姿态计算得分，并与之前的最佳得分比较，更新最佳得分。

* 根据pose中关键点的位置计算当前深蹲的得分，涉及到分析膝盖与髋部的相对高度，确保膝盖不超过脚尖，以及评估背部的直立度等。
* 将计算出的得分与best\_squat\_score进行比较，如果当前得分更高，则更新best\_squat\_score为当前得分。
* 返回更新后的最佳得分。

1. 函数调用：在每次检测到深蹲姿态后调用，用于跟踪最佳运动执行质量。
2. update\_best\_pull\_up\_score(pose, best\_pull\_up\_score):
3. 函数功能：更新引体向上的最佳得分。
4. 参数说明：

* pose: 当前帧检测到的姿态数据，类似于深蹲，包含了人体各关键点的坐标和置信度。
* best\_pull\_up\_score: 目前为止记录的最佳引体向上得分，得分基于姿态的质量。如上身抬升的高度，手臂的伸展程度。

1. 运行流程：

根据当前姿态计算得分，并与之前的最佳得分比较，更新最佳得分。

* 分析pose中的关键点位置来计算引体向上的得分，包括到测量腕部与头部的相对位置，以确定身体是否成功抬升至足够的高度。
* 比较当前得分与best\_pull\_up\_score，如果当前得分更高，就更新best\_pull\_up\_score。
* 返回更新后的最佳得分。

1. 函数调用：在每次检测到引体向上姿态后调用，用于跟踪最佳运动执行质量。

## 主逻辑与流程控制

main()

1. 函数功能：程序的主入口，控制算法整体运行流程，协调各模块之间的数据交互。负责初始化、循环处理视频帧、显示结果、处理用户输入。
2. 接口说明：

* initialize\_system(): 初始化算法参数，加载模型和基准数据。
* run\_system(): 运行算法主循环，依次处理视频帧，进行姿态检测、匹配与评分等操作。
* stop\_system(): 停止算法运行，释放资源。

1. 运行流程：

初始化参数和模型，创建搜索树，启动帧读取线程，主循环中处理视频帧，进行姿势检测，进行相似度匹配，进行计数和评分更新，显示视频帧和信息，处理按键事件，释放资源。

1. 函数调用：作为程序的起点被直接调用。

# 算法测试

## 环境配置

命令行模式：

1. 环境安装：将项目中movenet.tar.gz加压放到miniconda3(或者conda3)/envs下。
2. 启用终端：打开自带的miniconda3 prompt命令行。
3. 环境激活：输入conda activate movenet。
4. 切换路径：使用cd命令进入项目代码所在路径。
5. 选择模式：在main函数开头处，选择是否启用摄像头及运动类别。
6. 命令输入：输入python main.py，即可开始运行代码。代码执行结果保存在result文件夹中。

## 文件目录

* base\_data/: 存放基准姿势数据。
* movenet/: 包含模型主体。
* result/: 输出检测结果视频的目录。
* utils/: 放置 Movenet 姿态检测的辅助工具。
* video/: 包含检测所需的视频素材。
* main.py: 运动姿态识别的主程序。
* cameraTest.py: 用于摄像头测试的程序（仅供测试使用）。
* setup.py: Python 编译优化工具。
* 17pose.png: 包含17个人体骨骼点的参考图片。

## 操作说明

* 当use\_camera = True时：

通过按键切换运动检测类型：

* + - 按下1键选择深蹲。
    - 按2键选择引体向上。

系统将调用摄像头进行视频检测。

* 当use\_camera = False时：

需手动设置检测类别，适用于本地视频检测。

* 运行main.py以启动项目。
* 在任意时刻可按ESC键退出程序。
* 代码执行结果将以MP4格式保存至result/目录。

## 运行实例

将原始视频0.mp4放置在video/ 目录下。

### 本地视频模式

#### 引体向上模式

1. 手动更改检测项目模式：

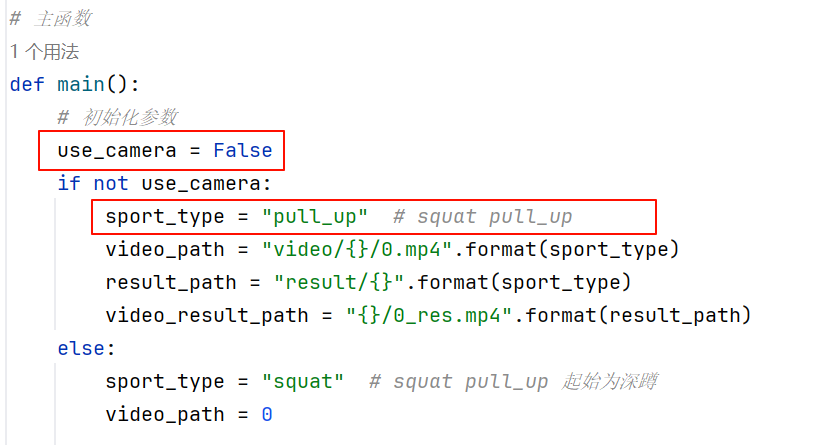


图 ‎6.1 引体向上模式

1. 界面显示：

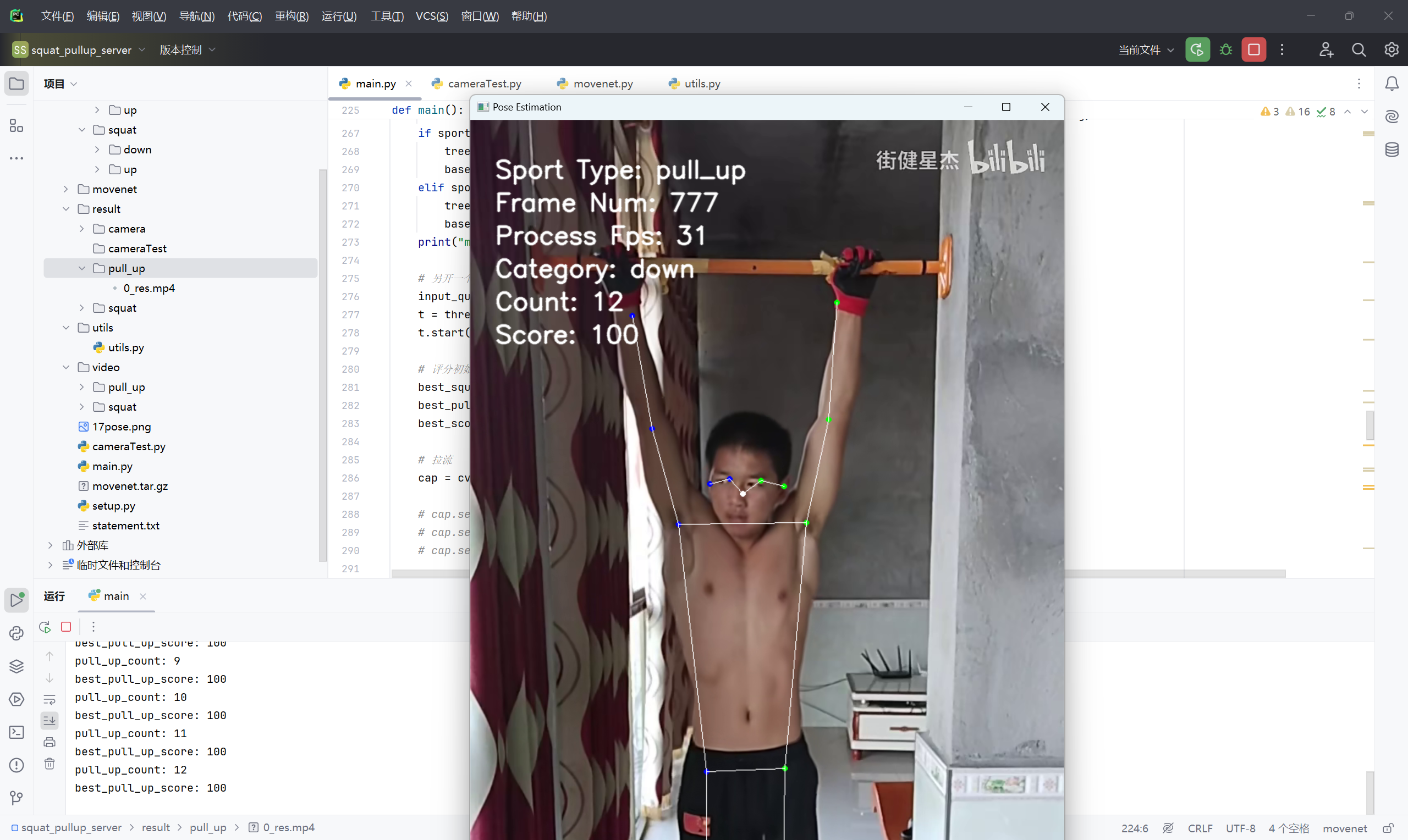


图 ‎6.2 引体向上检测界面预览

1. 结果输出：



图 ‎6.3 引体向上检测结果输出

#### 深蹲模式

界面显示：

### 摄像录制模式

# 成果和结论

本次研究旨在开发一种基于视频的体育动作标准度评估算法，以解决传统方法在设备依赖性和实时反馈方面的不足。通过对现有技术的研究和分析，结合姿态匹配技术和机器学习算法，本项目成功实现了对深蹲、引体向上等体育动作的识别和评估。在完成本研究的过程中，取得了以下的主要成果和结论：

本项目成功开发了基于Movenet人体骨骼点检测模型的体育动作识别算法。该算法能够准确地识别视频中的人体骨骼点，并通过姿态匹配技术对深蹲、引体向上等动作进行识别和评估。实验结果表明，新开发的算法在准确性和鲁棒性方面表现出色，为体育动作标准度评估提供了可靠的技术支持。

同时，本项目搭建并实现了一个可用的实时评分体系，能够在动作进行过程中即时给出评分和反馈。这个系统结合了新开发的识别算法，结合多运动姿态的不同标准，能够实现对不同体系动作执行过程的实时监测和评估，直观简洁，为用户提供了及时的指导和反馈，有效提高了训练效果。

此外，项目中还对算法本身进行了深入的性能评估和优化。通过对多种估计模型以及检测算法的调试和参数优化，有效地提高了检测过程的计算效率和实时性，基本解决了人体在姿态匹配精度和实时性方面遇到的困难，使得算法能够在实际的姿态检测应用中具有较好的性能表现与资源开销。

综上，本研究的成果为体育训练和评估提供了一种全新的、高效的评估方法。新开发的算法以及新一代姿态检测模型的应用，将为体育教练员、运动员乃至普通用户提供更加便捷、有效的训练辅助工具，提高了检测效率的同时降低了设备要求的门槛，有望在体育训练和运动健康管理领域产生重要的应用和推广价值。

本研究的成果为体育动作标准度评估领域的研究和实践提供了重要的技术支持和理论基础，具有一定的创新性和实用性，对相关领域的发展具有积极的推动作用。在未来，我将继续深入研究和优化算法，进一步提升其性能和适用性，为体育训练和运动健康管理提供更加完善的解决方案。

# 致谢

本次毕业设计项目的顺利完成，离不开我的导师姜丽老师的悉心指导。从选题的初步确定，到开题报告、初稿的修改，再到定稿的细节调整以及论文格式的问题，导师都给予了我极大的支持和指导。她不仅是一位优秀的导师，更是一位负责任、耐心细致的教育者，她协助我发现问题，提出了许多宝贵的意见和建议，帮我理清了设计思路以及操作方法，为课题提出了有效的改进方案，让我受益匪浅。我深知自己在这段时间里成长了许多，这都离不开老师的倾囊相授。在此，我衷心感谢导师的教诲，我会永远铭记在心。

我十分感谢导师将引领我进入计算机视觉领域。本论文研究的方向是基于视频的体育动作标准度评估算法实现。作为计算机科学技术的一个分支，我有幸深入探索神经网络模型和理论框架，体验当前信息科学与技术中的源源不断的创新，并领悟图像处理算法的深刻内涵。尽管我仍然是计算机视觉学习道路上的一个新人，但我会继续努力前行，坚持脚踏实地，不断学习和成长。我相信，在姜丽老师的指导下，我会在这个领域取得更多的进步和成就。

与此同时，我要感谢大学四年中所有任课老师以及辅导员对我的严格要求。他们在我学习期间给予了我学术与生活上的指点和帮助，不仅让我学到了专业知识，同时也了解了为人处世的道理，使我能够在今后的求学道路上继续奋斗。他们的言传身教，将成为我人生道路上宝贵的财富，我将倍加珍惜。

在即将毕业的最后时刻，我非常感激在大学生涯中能够遇到一群志同道合的朋友们。他们不仅在学术上给予我不同的视角与专业的指导，同时也是与我共同成长的伙伴。在交往过程中，我们建立了信任，彼此鼓励、支持和帮助，使我的大学生活充实而满足，留下了珍贵难忘的回忆。我相信，我们在未来的人生道路上将会携手共进，共同追求更加美好的未来。

最后，再次感谢所有曾经帮助过我的良师益友们，以及在毕业论文设计中被我引用或参考的论著的作者们。没有你们的支持和帮助，我无法完成这篇毕业论文。愿我们共同的努力和付出，成就更加美好的未来。

# 参考文献

1. 王芫. 基于深度神经网络的人体运动姿态估计与识别.电子科技大学. 2020
2. 蔡敏敏. 基于人体姿态估计的视频动作对比分析研究. 2022
3. Mihai Fieraru, Anna Khoreva, Leonid Pishchulin, Bernt Schiele. Learning Human Pose Estimation Features with Convolutional Networks. 2018.

https://arxiv.org/abs/1804.07909

1. Arjun Jain, Jonathan Tompson, Mykhaylo Andriluka, Graham W. Taylor, Christoph Bregler. Learning human pose estimation features with convolutional networks. 2013. https://arxiv.org/abs/1312.7302
2. Rishabh Bajpai, Deepak Joshi. MoveNet: A Deep Neural Network for Joint Profile Prediction Across Variable Walking Speeds and Slopes. 2021. IEEE.

https://ieeexplore.ieee.org/document/9406043

1. Pramod Murthy, Bertram Taetz, Arpit Lekhra. DiveNet: Dive Action Localization and Physical Pose Parameter Extraction for High Performance Training. 2023. IEEE.

https://ieeexplore.ieee.org/document/10097490

1. Bernhard Hollaus, Bernhard Reiter, Jasper C. Volmer. Catch Recognition in Automated American Football Training Using Machine Learning. Sensors-Basel, January 2023. https://www.mdpi.com/1424-8220/23/2/840
2. Wenbo Li, Zhicheng Wang, Binyi Yin, Qixiang Peng, Yuming Du, Tianzi Xiao, Gang Yu, Hongtao Lu, Yichen Wei, Jian Sun. Rethinking on Multi-Stage Networks for Human Pose Estimation. May 2019.

https://arxiv.org/abs/1901.00148

1. Wei Tang, Pei Yu, Ying Wu. Deeply Learned Compositional Models for Human Pose Estimation. 2018.

https://openaccess.thecvf.com/content\_ECCV\_2018/html/Wei\_Tang\_Deeply\_Learned\_Compositional\_ECCV\_2018\_paper.html

1. Bin Xiao, Haiping Wu, Yichen Wei. Simple Baselines for Human Pose Estimation and Tracking. 2018.

https://openaccess.thecvf.com/content\_ECCV\_2018/html/Bin\_Xiao\_Simple\_Baselines\_for\_ECCV\_2018\_paper.html

1. Feng Zhang, Xiatian Zhu, Mao Ye. Fast Human Pose Estimation. 2019.

https://openaccess.thecvf.com/content\_CVPR\_2019/html/Zhang\_Fast\_Human\_Pose\_Estimation\_CVPR\_2019\_paper.html

1. Yanrui Bin, Xuan Cao, Xinya Chen, Yanhao Ge, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, Changxin Gao & Nong Sang. Adversarial Semantic Data Augmentation for Human Pose Estimation. November 2020.

https://link.springer.com/chapter/10.1007/978-3-030-58529-7\_36

1. Ke Sun, Bin Xiao, Dong Liu, Jingdong Wang. Deep High-Resolution Representation Learning for Human Pose Estimation. 2019.

https://arxiv.org/abs/1902.09212

1. TFBlog: Next-Generation Pose Detection with MoveNet and TensorFlow.js:

https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html

1. TFHub: movenet/singlepose/lightning:

https://www.kaggle.com/models/google/movenet/frameworks/tensorFlow2/variations/singlepose-lightning/versions/4?tfhub-redirect=true

附录

附录说明文字。附录篇幅不宜过大。如果文档或程序过多，可以写明附录见所附软盘或光盘。没有请删去此部分内容。