Open-Source Software Library For Real-Time Inertial Measurement Unit Data-Based Inverse Kinematics Using OpenSim

Jere Lavikainen (✉ jere.lavikainen@uef.fi)

University of Eastern Finland <https://orcid.org/0000-0002-7636-8229>

Paavo Vartiainen

University of Eastern Finland

Lauri Stenroth

University of Eastern Finland

Pasi Karjalainen

University of Eastern Finland

Research Article

Keywords: OpenSim, real-time, inverse kinematics, inertial measurement unit, motion analysis

Posted Date: October 19th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-775793/v2>

License: This work is licensed under a Creative Commons Attribution 4.0 International License. [Read Full License](#)

Abstract

An open-source software library for multithreaded real-time inverse kinematical (IK) analysis of inertial measurement unit (IMU) data using OpenSim was developed. Its operation delays and throughputs were measured with a varying number of IMUs and parallel computing IK threads using two different musculoskeletal models, one a lower-body and torso model and the other a full-body model.

Full-body inverse kinematics with data from 12 IMUs could be calculated in real-time with a mean delay below 100 ms and at more than 900 samples per second. Live visualization of IK is an option but results in limited IK throughput. The effect of this limitation was assessed by comparing the range of motion (ROM) of each joint from visualized real-time IK to the ROM from offline IK at IMU sampling frequency, resulting in mean ROM differences below 0.3 degrees.

The software library enables real-time inverse kinematical analysis with different numbers of IMUs and customizable musculoskeletal models, making it possible to do subject-specific full-body motion analysis outside the motion laboratory in real-time.

Introduction

Inertial measurement units (IMUs) are measurement devices that contain triaxial magnetometers, gyroscopes and accelerometers. IMUs used in biomechanics are usually packed into cases that fit on a human palm. Sensor fusion algorithms such as Kalman filters are used to estimate the three-dimensional orientation of the IMUs in space. [1]

IMUs can be used as an alternative to marker-based optical motion tracking systems to perform motion analysis. Compared with optical motion tracking systems, IMUs are cheaper, can be attached to the subject without the palpation of anatomical landmarks, do not suffer from line-of-sight issues, are not limited to a specific target volume and can be used in field conditions. This comes at a small cost of accuracy compared with optical motion tracking systems, and IMU-specific error sources such as drifting. [2]

Although IMUs have been used in the study of human motion and biomechanics for less than two decades, they have been proven to be a good solution to overcome the shortcomings of optical motion tracking systems and their popularity is increasing. Moreover, sensors in which IMU is coupled with electromyography (EMG) electrodes further enhances the versatility of these sensors for analyzing human movement in sports and clinical applications [3–5].

OpenSim [6] is an open-source software for analyzing the kinematics and dynamics of musculoskeletal systems. The inverse kinematics (IK) tool is commonly used in OpenSim. It takes motion tracking data, which is often marker-based, as its input and solves the motion of the model (i.e., joint angles) by minimizing the sum of squares of the difference between experimental and model markers positions. As of OpenSim 4.1 [7], inverse kinematics can also be calculated using IMU data as the input. Here, the differences between the model and experimental sensor orientations are minimized.

IK analyses are typically done offline so that the IK is analyzed after the measurement is finished. Recent years have seen progress in some real-time or online analysis solutions and systems, but these studies [8–12] have mostly focused on specific marker sets and models and their generalization to arbitrary measurement setups is difficult. For example, in real-time IMU-based applications, Bonnet et al. [8] estimated the real-time inverse kinematics (RTIK) of the trunk and lower limbs using a single IMU located at the lower back. In another example, sensors containing IMUs and EMG electrodes, which were placed on muscles, were used in a recent study [12] to calculate the real-time kinematics and kinetics of the lower limb. However, these studies rely on complex computational methods, making it difficult for others to repeat and adapt the experiment without knowledge of sensor fusion or deep learning.

It has been shown that a full biomechanical analysis of joint and muscle function can be obtained in real-time with a C/C++ software library [9]. The software library was capable of reading marker data and performing IK and inverse dynamics on a full-body model at more than 120 samples per second. However, the software is commercial and relies on a single predefined model rather than a subject-specific or user-defined musculoskeletal model, limiting its usefulness in research. Additionally, Falisse et al. [11] showed that comparing its outputs with those of OpenSim 3.3 resulted in statistically significant differences in joint kinematics, kinetics and muscle forces, highlighting the dependency of the output on the selected model. Hence, it is invaluable that the user can select or generate a model that best fits to the application or research question.

A software architecture developed by Pizzolato et al. for RTIK and inverse dynamics of human motion [13] utilizes OpenSim to enable analysis with different marker placement schemes and musculoskeletal models. However, it is based on a now-outdated version of the OpenSim application programming interface (API) and does not support IMU-based motion data. Consequently, reliance on optical motion tracking limits its applicability to laboratory settings.

While this study was being reported, Stanev et al. [14] published an open-source software framework that allows the kinematical and dynamical analysis of subject-specific musculoskeletal models using the OpenSim API. Their software performs the analysis in real-time and supports both IMU- and marker-based data. The differences between it and the one developed in this study are evaluated in Conclusions.

The aim of this study was to develop a freely available software library that reads orientation data from IMUs and calculates the IK on a user-given musculoskeletal model in real-time using OpenSim 4.1 API. Further, the aim was to quantify the software's IK execution time and throughput with different numbers of processor threads calculating the IK and different numbers of IMUs, and to determine if lowered input data frequency resulting from live visualization meaningfully affects calculated ranges of motion (ROMs).

Methods

Working principles of the software

A software library for reading real-time IMU orientation data as quaternions and processing the data to calculate the IK of a musculoskeletal model was developed using C++ and published on GitHub. The software utilizes OpenSim 4.1 API to invoke methods that calibrate the musculoskeletal model and perform the IK to solve joint angles using quaternion-based orientation data from live measurements.

The software library supports Xsens MTw Awinda™ (Xsens Technologies B.V., Enschede, Netherlands) and Delsys Trigno (Delsys Inc., Natick, Massachusetts, USA) IMUs. The open-source nature of the software library allows others to add support for other devices. IMU orientations are received wirelessly as quaternions using the Xsens Device API or individual quaternion elements are read from a byte stream via socket communication sent by Trigno Control Utility. Information about which IMU corresponds to which body on the musculoskeletal model is read from an XML file. Instead of reading orientation data from the actual IMUs, an option to generate randomized quaternion orientations for testing purposes without IMUs is available.

The orientation information from IMUs is combined in a time series table that contains only one sample, i.e., time point. The time series table is given to OpenSim's IK solver object, which solves the IK for that time point. The process is repeated for each sample. The resulting time series of joint angles can be saved in a text file in .mot format, which allows the output to be viewed using the OpenSim graphical user interface. The read quaternions, and also EMG time series in the case of Trigno Avanti sensors, can be saved in a text file for later offline analysis.

The working principle of the software library is straightforward. Producer-consumer thread synchronization is used to get orientation data from IMUs. A producer thread and a consumer thread run concurrently. The producer reads orientation data as quaternions from the IMUs and saves it into a buffer that is shared between the threads. The consumer reads and removes data from the buffer and creates a new IK thread in a thread pool. This way there can be several concurrent IK calculations, improving the throughput of the program. The maximum number of concurrent IK threads is defined by the user. If that number is already active when the consumer thread starts another IK thread, the consumer thread will wait for one of the IK threads in the thread pool to finish. A diagram illustrating the workflow is shown in **Fig. 1**.

The software library has been tested to work on 64-bit Windows 7 and Windows 10 operating systems. The source code of the software library is available on GitHub at <https://github.com/jerela/OpenSimLive>.

Execution times of the IK operation

The execution time of a single IK operation was determined by calculating the duration between its starting and finishing points. The mean, standard deviation (STD) and 95% confidence interval of the execution time was calculated for 10 000 IK operations. Because the orientation retrieval time of IMUs was negligibly small compared with the execution time of the computationally heavier IK and varied between Xsens and Delsys IMUs, only IK execution time was studied. The execution time represents the delay between receiving orientation data from IMUs and retrieving the corresponding IK output. For the orientation input for IK, unit quaternions were randomly generated instead of using real data. The use of random quaternions ensured that execution times were calculated for a large variety of different orientation combinations.

Execution time was calculated separately using the Gait2392 lower extremities and torso model (23 degrees of freedom, DOFs; referenced from here on as the lower body model) [15–18] with 1 and 7 IMUs and the Hamner full-body model (29 DOFs, referenced from here on as the full-body model) [19] with 1, 7 and 12 IMUs. With 1 IMU, the corresponding body segment was the pelvis. With 7 IMUs, they were the pelvis, thighs, shanks and the feet. With 12 IMUs, they were the torso, upper arms and lower arms in addition to the segments of the setup with 7 IMUs.

Throughputs

The throughput of the IK, defined as IK operations per second, was measured by calculating how many IK operations the software library could finish in one minute. In addition to the IK calculation itself, the measurement workflow included the producer thread pushing identity quaternions and their time points to the shared buffers and the consumer thread reading them. The measurements were done with both the lower body model and the full-body model using 1, 2, 4, 6, 8, 12, 16 and 24 threads for concurrent IK. For the lower body model, the measurements were done with 1 and 7 IMUs, and with 1, 7 and 12 IMUs for the full-body model, similarly to how the execution time measurements were measured. Instead of measuring real human motion, an identity quaternion was used as the orientation for each IMU, ensuring that the IK throughputs are comparable between repetitions and that the throughputs represent ideal performance.

The execution time and throughput tests were run with two computers, one a desktop (Fujitsu Celsius W550 Power: Windows 10 Education 64-bit, Intel Core i7-6700 3.40 GHz 8-CPU processor, 32768 MB RAM) and the other a laptop computer (HP EliteBook 8570w: Windows 10 Education 64-bit, Intel Core i7-3740QM 2.70 GHz 8-CPU processor, 8192 MB RAM). The desktop results represent the performance of the software library in a stationary laboratory setup while the laptop results illustrate how the software library can perform in portable measurement settings outside a laboratory. The live visualization of RTIK was disabled during both performance tests.

Error comparison of joint angles

When live visualization is disabled, the typical RTIK solving frequency is higher than the sampling frequency of the IMUs and therefore the IK solution includes data at the sampling frequency of the IMUs. However, when live visualization is enabled during RTIK, its throughput drops leading to RTIK solving frequency that is lower than the sampling frequency of the IMUs. In this case, samples are skipped which may negatively affect the accuracy of gait parameters derived from the IK solution. We evaluated the effect of this frame drop by comparing the ranges of motion (ROMs) of the joint angles solved with RTIK to ROMs of the joint angles solved with offline IK at the full sampling frequency of the IMUs.

To measure how accurately ROM can be analyzed in real time with and without visualization, subject motion during walking on an instrumented treadmill (Motek Medical B.V., Amsterdam, Netherlands) was measured with IMUs and RTIK was calculated. The live visualization of RTIK was enabled, which reduced the frequency of RTIK from the 60 Hz sampling frequency of the IMUs to 45 Hz on average when using the desktop computer. The time series of IMU orientations as quaternions were saved to a file at the full 60 Hz sampling frequency after each trial and analyzed to obtain offline IK. Ten walking trials, each containing approximately a minute of walking data at a speed of 1.5 m/s, were measured with Xsens MTw Awinda IMUs. The full-body model was used in the analysis and the measured motion exerted 26 of its DOFs.

The subject was 25 years old, had a height of 181 cm and a weight of 85 kg and gave his written consent to participate in the study. A total of 12 IMUs were strapped on the subject's upper arms, forearms, chest, pelvis, thighs, shanks and feet, as shown in supplementary information (Online Resource 1). The subject was instructed to take the standard anatomical position at the beginning of each trial to calibrate the IMUs on the musculoskeletal model as per the standard IMU calibration procedure of OpenSim.

RTIK and offline IK were used to calculate ROM for all joints of the model from 715 gait cycles. The ROMs were compared between RTIK and offline IK to determine effect of the lower IK sampling frequency due to the visualization on the ROMs. ROM was selected as the variable of interest since it indicates if the lower sampling frequency has a major effect on the IK solution. The results were reported as the mean absolute error (MAE) and 95% confidence interval between offline IK and RTIK ROM.

Results

Execution times of the IK operation

The means and standard deviations of execution times (operation delays) increased with increasing number of IMUs similarly on both the desktop and the laptop (Table 1). With one IMU, the full-body model (Hamner) was 60–65% slower than the lower body model (Gait2392) and had more variation in execution times. The mean execution times were approximately 25% longer on the laptop than on the desktop.

Table 1

Mean, standard deviation (STD) and 95% confidence interval (CI) of execution times of a single inverse kinematics (IK) operation. The values are calculated over 10 000 IK operations for two different musculoskeletal models, two different computers and 1, 7 or 12 inertial measurement units (IMUs). Randomly selected unit quaternions were used as IMU orientations.

	Gait2392, 1 IMU		Gait2392, 7 IMUs		Hamner, 1 IMU		Hamner, 7 IMUs		Hamner, 12 IMUs	
	Desktop	Laptop	Desktop	Laptop	Desktop	Laptop	Desktop	Laptop	Desktop	Laptop
Mean time (ms)	6.97	8.52	43.55	56.96	11.52	13.77	48.12	64.01	80.42	95.24
STD (ms)	2.31	3.62	22.39	27.19	6.08	7.05	34.03	46.36	29.49	42.00
95% CI (ms)	0.05	0.07	0.44	0.53	0.12	0.14	0.67	0.91	0.58	0.82

Throughputs

Throughputs higher than 1100 and 700 operations per second were reached on the desktop and the laptop, respectively (**Fig. 2**). The throughput of the desktop was 30–50% greater than that of the laptop when more than one IK thread was used. When one IK thread was used, the throughput of the desktop was 630–1560% greater than that of the laptop, with the use of fewer IMUs leading to a greater difference in throughput. The lower body model performed faster than the full-body model, and model selection had a higher impact on throughput than the number of IMUs. Maximum CPU utilization is seen as flattening of the curve after eight threads for the desktop and six threads for the laptop.

Error comparison of joint angles

The mean ROM error for all DOFs was 0.0675 degrees. The greatest MAE in ROM was observed in ankle joints (up to 360% of the mean for all joints), followed by the left hip joint (**Fig. 3**). Pelvis and upper extremities had the smallest ROM error. All MAEs remained below 0.3 degrees.

Discussion

We present an open-source software library for the real-time inverse kinematical analysis of IMU data with user-defined musculoskeletal models using OpenSim 4.1. Full-body IK can be calculated for a single sample in less than 100 ms. On a desktop computer, the software library can solve RTIK at more than 1100 samples per second while tracking the pelvis and the lower extremities and more than 900 samples per second while tracking the full-body kinematics. On a laptop computer, the corresponding throughputs were 700 and 600 samples per second, respectively. Using 12 IMUs to track walking and visualizing the results on a full-body running model, RTIK was solved at 45 samples per second. The drop from the IMU output sampling rate of 60 Hz resulted in a minimal difference in calculated joint ROMs (< 0.3 degrees). The software library allows the use of RTIK virtually without limitations due to location or environment. This opens possibilities for a variety of applications including rehabilitation, ergonomics and human-machine interfaces for controlling collaborative robots. Moreover, it has been shown that a laboratory setting may affect how a person moves [20] and thus it is beneficial that the movement of interest can be observed in the real environment of that movement or behavior.

Execution times and throughputs

We investigated the execution times and throughputs of the IMU-based IK to determine if the output can be considered real-time. Pizzolato et al. [13] used an execution time of 75 ms as the threshold for a real-time system. It was based on a study by Kannape and Blanke [21] in which the subjects were able to identify the displayed motion as self-generated in real-time in over 80% of the cases if the delay in motion display was less than 75 ms. Even with a delay of 210 ms, subjects identified the visualized motion as self-generated in real-time in 50% of the cases. Borbély and Szolgay [10] noted that the IK algorithm of OpenSim 3.3 had an execution time of about 145 ms, thus calculating IK at about 7 Hz and “falling behind the generally accepted practice in human movement recording of at least 50 Hz”. Therefore, a real-time application should achieve IK throughput of 50 operations per second with an execution time below 75 ms for any single operation. With our software library, we aimed to achieve this target by using multithreading and the IK algorithm of OpenSim 4.1.

Another interesting finding by Kannape and Blanke [21] was that subjects modulated their stride based on the delay between the motion and its visualization. Therefore, it is important to minimize the delay when preparing a real-time measurement setup to prevent subjects from altering their gait characteristics based on delayed visual feedback.

Live visualization is unnecessary in applications where IK is an intermediate output that is used to estimate contact forces, instruct a robot arm in rehabilitation applications or calculate gait parameters, to name a few examples. Thus, the performance tests were designed so that they evaluate only the performance of IK, which is the core feature of the software library.

Real motion, such as walking, contains a combination of different orientations, most of which are within a typical model's joint angle boundaries. The constant identity quaternions used in throughput tests represent the calibration pose reoccurring repeatedly, while randomly generated unit quaternions used in execution time tests often result in unrealistic poses. This makes the IK based on identity quaternions simpler and the IK based on randomized unit quaternions heavier to calculate than the average orientations during walking, or any typical human motion. Therefore, the execution times can be interpreted as the worst performance and the throughput results as the best performance when analyzing human motion without live visualization.

Execution times of the IK operation

Table 1 shows that with one and seven IMUs, the execution times are shorter and vary less for the lower body model (Gait2392) than for the full-body model (Hamner), which implies that the execution time of the lower body model is more consistent than that of the full-body model. Both the mean execution times and the standard deviations are smaller on the desktop than on the laptop. However, the execution times vary less with 12 than with seven IMUs on the full-body model.

For both models, the standard deviations of the execution times are on the same scale as the mean execution times, implying that there is great variation in the execution time. The randomized nature of the used quaternion orientations is a likely contributor to the high standard deviation, because randomized orientations occasionally lead to strange segment orientation combinations that do not reflect valid human motion and take the IK algorithm a varying amount of time to solve. During the development of the test program, it was noticed that the results varied greatly, implying that more than 10 000 IK operations are required to draw lasting conclusions. However, running the test even with 10 000 operations could take up to 20 minutes, so the number of operations was left as it was.

The 95% confidence intervals of execution times are roughly 1% of the mean execution time in all cases, meaning that the execution times stay consistently below 75 ms except when 12 IMUs are used. In that case, the execution times stay consistently below 100 ms. Although measuring full-body motion with 12 IMUs fails to meet the best criterion for delay, it is less than half of the 210 ms delay that marks 50% confidence in perceiving motion as real-time [21]. Therefore,

while the execution times with 12 IMUs are not ideal, they are still acceptable. Because the execution times represent the minimum delay from the orientation data retrieval to the moment we can visualize or further analyze the IK output, the number of IMUs in a real-time measurement should be chosen considering the delays that are acceptable for the application.

Throughputs

Figure 2 shows that increasing the number of concurrent processor threads increased the throughput until about eight threads, which was the maximum CPU core number for both computers. Increasing the number of IK threads further had no meaningful effect on the throughput, which was also observed in an earlier study on RTIK [13]. It was observed during the testing that CPU utilization reached 100% at six and eight concurrent threads with the desktop and the laptop, respectively, while memory utilization did not get close to its maximum capacity. In terms of CPU clock rate, the desktop was 26% faster than the laptop. The closest match to this performance difference in throughput is found with two IK threads (30–33% faster).

The increase in throughput by multithreading is especially large when a low number of threads are used. For example, throughput increases from less than 50 to approximately 400 when the number of IK threads increases from one to two on the laptop. Doubled computational capacity alone cannot explain the increase in throughput. The effect is less pronounced but still present on the desktop. Furthermore, the relationship between the throughput and the number of IK threads is clearly nonlinear whereas an earlier RTIK study found it almost linear [13]. No explanation for this phenomenon was found, but it should be addressed in the future development of the software library.

For any number of IMUs and 4 or more concurrent threads, the lower body model with 23 DOFs performed approximately 10% faster than the full-body model with 29 DOFs. Therefore, model selection has a noticeable effect on the performance of RTIK and the model with the smallest sufficient number of DOFs should be chosen to reach maximal RTIK performance.

The number of IMUs had a smaller effect on throughput than model selection and computer hardware. Although it was not reported in the results, the number of IMUs was observed to affect the throughput more significantly when joint angle boundaries of the model are exceeded. Furthermore, in that situation the performance effect of model selection increased and became more important than computer hardware.

The software library is clearly capable of calculating IK at a higher rate than the lower limit of 50 Hz named by Borbély and Szolgay [10], but requires multithreading to reach it with complex musculoskeletal models. The throughputs reported in this study should be sufficient to match the sampling frequency of most IMUs because they typically have a maximal sampling frequency well below 500 Hz. For instance, the maximum sampling rate of Xsens MTw Awinda IMUs is 120 Hz [1].

Error comparison

Because loss of frequency may lead to reduced accuracy in measuring sharp peaks in joint angles, joints where motion direction changes fast are likely to have high ROM error (**Fig. 3**). During walking ankle flexion (ankle_angle_r and ankle_angle_l) undergoes fast changes, which explains why its ROM error stands out. However, because all ROM errors remain consistently below 0.3 degrees, the effect of the drop in visualized IK from 60 to 45 Hz on ROM is very small.

The ROM error of left hip adduction stands out because it is visibly higher than that of the right hip. The error is caused by an artifact in IMU signal that caused the left leg to be violently jerked to the right after the left toe-off phase. The artifact is probably caused by the distortion of magnetic fields near the ferromagnetic laboratory hardware, which the left leg was closer to.

Conclusion

An open-source software library that builds upon the widely used OpenSim software was developed and published for IMU-based RTIK. This allows the joint angles of any OpenSim-compatible musculoskeletal model to be analyzed in real-time. While another real-time solution was concurrently and independently developed by Stanev et al. [14], its IK calculation does not utilize multithreading, which may limit its throughput, although its IK calculation relies on lower-level API classes that are faster than those used by the software library developed in this study when a single thread is used. The authors encourage others to contribute to the open-source project. The development of the software library will closely follow the development of OpenSim to utilize its built-in functionality for processing live data. The software library could be utilized in real-time estimation of joint moments, muscle forces and joint contact forces based only on IMU data. Ground reaction forces and moments and kinematics are required for solving the equations of motion for the musculoskeletal model using inverse dynamics. It has been shown that ground reaction forces and moments can be predicted from IMU-derived kinematics [14, 22]. Moreover, estimation of muscle forces using optimization techniques uses kinematics and inverse dynamics estimates of joint moments as inputs and estimates of joint contact forces can be derived based on kinematics, inverse dynamics and muscle forces. Hence, IMUs could be potentially used for the real-time estimation of musculoskeletal dynamics outside the laboratory and implemented in the software library in the future. Another interesting future application is the use of RTIK output together with EMG. Thus, combining IK output with EMG in real-time may provide interesting possibilities for estimating muscle forces and musculoskeletal loading using EMG driven musculoskeletal simulations [23], for biofeedback to optimize rehabilitation or ergonomics or for biosignal-based operating systems.

Declarations

Funding

This work was supported by the European Union (European Regional Development Fund) and the University of Eastern Finland under the projects Human measurement and analysis research and innovation laboratories (HUMEA, project IDs A73200 and A73241) and Digital Technology RDI Environment (Digi Center, project IDs A74338 and A74340).

Conflicts of interest

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and material

Not applicable.

Code availability

The software library described in this article is published open-source at <https://github.com/jerela/OpenSimLive>.

Author's contributions

JL conceptualized and developed the software library and wrote the original manuscript draft and planned and conducted the measurements and their analysis.

PV conceptualized the software library and planned and conducted the measurements. He also provided resources during the research, supervised the research and reviewed and edited the manuscript.

LS conceptualized the software library and planned the measurements. He also provided resources during the research and reviewed and edited the manuscript.

PAK supervised and administered the research and acquired funding for it.

Ethics approval

Not applicable.

Consent to participate

The subject signed a participation form with their informed consent to participate in the study.

Consent for publication

The subject signed a participation form with their informed consent to allow the publication of their data recorded during this study.

References

1. Paulich M, Schepers M, Rudigkeit N, Bellusci G (2018) Xsens MTw Awinda: Miniature Wireless Inertial Motion Tracker for Highly Accurate 3D Kinematic Applications
2. Saber-Sheikh K, Bryant EC, Glazzard C et al (2010) Feasibility of using inertial sensors to assess human movement. *Man Ther* 15:122–125. <https://doi.org/10.1016/j.math.2009.05.009>
3. Cometa, Srl, Cometa Systems | Wireless EMG and IMU solutions. <https://www.cometasystems.com/>. Accessed 10 Feb 2021
4. Delsys Incorporated, Trigno™ Avanti Platform - Delsys. <https://www.delsys.com/trigno/>. Accessed 10 Feb 2021
5. Noraxon USA, Ultium EMG, | Noraxon USA. <https://www.noraxon.com/our-products/ultium-emg/>. Accessed 10 Feb 2021
6. Delp SL, Anderson FC, Arnold AS et al (2007) OpenSim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Trans Biomed Eng* 54:1940–1950. <https://doi.org/10.1109/TBME.2007.901024>
7. Seth A, Hicks JL, Uchida TK et al (2018) OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS Comput Biol* 14:. <https://doi.org/10.1371/journal.pcbi.1006223>
8. Bonnet V, Mazza C, Fraisse P, Cappozzo A (2013) Real-time estimate of body kinematics during a planar squat task using a single inertial measurement unit. *IEEE Trans Biomed Eng* 60:1920–1926. <https://doi.org/10.1109/TBME.2013.2245131>
9. Van Den Bogert AJ, Geijtenbeek T, Even-Zohar O et al (2013) A real-time system for biomechanical analysis of human movement and muscle function. *Med Biol Eng Comput* 51:1069–1077. <https://doi.org/10.1007/s11517-013-1076-z>

10. Borbély BJ, Szolgay P (2017) Real-time inverse kinematics for the upper limb: A model-based algorithm using segment orientations. *Biomed Eng Online* 16:. <https://doi.org/10.1186/s12938-016-0291-x>
11. Falisse A, Van Rossom S, Gijsbers J et al (2018) OpenSim versus human body model: A comparison study for the lower limbs during gait. *J Appl Biomech* 34:496–502. <https://doi.org/10.1123/jab.2017-0156>
12. Yi C, Jiang F, Bhuiyan MZA et al (2021) Smart healthcare-oriented online prediction of lower-limb kinematics and kinetics based on data-driven neural signal decoding. *Futur Gener Comput Syst* 114:96–105. <https://doi.org/10.1016/j.future.2020.06.015>
13. Pizzolato C, Reggiani M, Modenese L, Lloyd DG (2017) Real-time inverse kinematics and inverse dynamics for lower limb applications using OpenSim. *Comput Methods Biomech Biomed Engin* 20:436–445. <https://doi.org/10.1080/10255842.2016.1240789>
14. Stanev D, Filip K, Bitzas D et al (2021) Real-time musculoskeletal kinematics and dynamics analysis using marker- and imu-based solutions in rehabilitation. *Sensors* 21:1–20. <https://doi.org/10.3390/s21051804>
15. Yamaguchi GT, Zajac FE (1989) A planar model of the knee joint to characterize the knee extensor mechanism. *J Biomech* 22:1–10. [https://doi.org/10.1016/0021-9290\(89\)90179-6](https://doi.org/10.1016/0021-9290(89)90179-6)
16. Delp SL, Loan JP, Hoy MG et al (1990) An Interactive Graphics-Based Model of the Lower Extremity to Study Orthopaedic Surgical Procedures. *IEEE Trans Biomed Eng* 37:757–767. <https://doi.org/10.1109/10.102791>
17. Anderson FC, Pandy MG (1999) A dynamic optimization solution for vertical jumping in three dimensions. *Comput Methods Biomech Biomed Engin* 2:201–231. <https://doi.org/10.1080/10255849908907988>
18. Anderson FC, Pandy MG (2001) Dynamic optimization of human walking. *J Biomech Eng* 123:381–390. <https://doi.org/10.1115/1.1392310>
19. Hamner SR, Seth A, Delp SL (2010) Muscle contributions to propulsion and support during running. *J Biomech* 43:2709–2716. <https://doi.org/10.1016/j.jbiomech.2010.06.025>
20. Friesen KB, Zhang Z, Monaghan PG et al (2020) All eyes on you: how researcher presence changes the way you walk. *Sci Rep* 10:. <https://doi.org/10.1038/s41598-020-73734-5>
21. Kannape OA, Blanke O (2013) Self in motion: Sensorimotor and cognitive mechanisms in gait agency. *J Neurophysiol* 110:1837–1847. <https://doi.org/10.1152/jn.01042.2012>
22. Karatsidis A, Bellusci G, Schepers HM et al (2017) Estimation of ground reaction forces and moments during gait using only inertial motion capture. *Sensors (Switzerland)* 17:. <https://doi.org/10.3390/s17010075>
23. Sartori M, Reggiani M, Lloyd DG, Pagello E (2011) A neuromusculoskeletal model of the human lower limb: Towards EMG-driven actuation of multiple joints in powered orthoses. *IEEE Int Conf Rehabil Robot*. <https://doi.org/10.1109/ICORR.2011.5975441>

Figures

Figure 1

A diagram illustrating the working principle of the inverse kinematics (IK) workflow. Orientation data of inertial measurement units (IMUs) is read as quaternions by the producer thread and saved to a buffer. Time values are saved to another buffer. The consumer thread reads data from both buffers and initiates new threads that calculate IK based on the data. IK threads output joint angle values for the model, which can be sent to a visualizer window. When the program finishes, the IK output frames can be sorted in a time-ascending order and saved to file.

Figure 2

Inverse kinematics (IK) throughput with respect to the number of IK threads used, measured on a desktop computer (left) and a laptop computer (right). The throughput is calculated as mean operations per second over a one-minute trial. The measurements were repeated with two different musculoskeletal models, and using one, seven or 12 inertial measurement units (IMUs). Identity quaternions were used as IMU orientations.

Figure 3

Mean absolute error (MAE) between real-time inverse kinematics and offline inverse kinematics ranges of motion (ROMs) of the exerted degrees of freedom of the musculoskeletal model and the 95% confidence interval of the error. Solid bars show MAEs of ROMs for each exerted degree of freedom. Confidence intervals are shown as error bars centered on the top of the MAE bars.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- [Onlineresource1.pdf](#)