# Open-source software library for real-time inertial measurement unit data-based inverse kinematics using OpenSim

Jere Lavikainen[1], Paavo Vartiainen[1], Lauri Stenroth[1], Pasi A. Karjalainen[1]

[1] Department of Technical Physics, University of Eastern Finland, Kuopio, Finland

Corresponding Author:

Jere Lavikainen[1]

Yliopistonranta 1 F (Melania), 70210 Kuopio, Finland

Email address: jere.lavikainen@uef.fi

## Abstract

**Background.** Inertial measurements (IMUs) facilitate the measurement of human motion outside the motion laboratory. A commonly used open-source software for musculoskeletal simulation and analysis of human motion, OpenSim, includes a tool to enable kinematics analysis of IMU data. However, it only enables offline analysis, i.e., analysis after the data has been collected. Extending OpenSim's functionality to allow real-time kinematics analysis would allow real-time feedback for the subject during the measurement session and has uses in e.g. rehabilitation, robotics, and ergonomics.

**Methods.** We developed an open-source software library for real-time inverse kinematics (IK) analysis of IMU data using OpenSim. The software library reads data from IMUs and uses multithreading for concurrent calculation of IK. 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. We published the code under an open-source license on GitHub.

**Results.** A standard desktop computer calculated full-body inverse kinematics from treadmill walking at 1.5 m/s with data from 12 IMUs in real-time with a mean delay below 55 ms and reached a throughput of more than 90 samples per second. A laptop computer had similar delays and reached a throughput above 60 samples per second with treadmill walking. Minimal walking kinematics, motion of lower extremities and torso, were calculated from treadmill walking data in real-time with a throughput of 130 samples per second on the laptop and 180 samples per second on the desktop computer, with approximately half the delay of full-body kinematics.

**Conclusions.** The software library enabled real-time inverse kinematical analysis with different numbers of IMUs and customizable musculoskeletal models. The performance results show that

39  subject-specific full-body motion analysis is feasible in real-time, while a laptop computer and
40  IMUs allowed the use of the method outside the motion laboratory.
41
## 42 Introduction
43  Inertial measurement units (IMUs) are measurement devices that contain triaxial magnetometers,
44  gyroscopes, and accelerometers. IMUs used in biomechanics are usually packed into cases that
45  fit on a human palm. They utilize sensor fusion algorithms such as Kalman filters to estimate the
46  three-dimensional orientation of the IMUs in space (Paulich et al., 2018). This information can
47  be used as an alternative to marker-based optical motion tracking systems to perform analysis of
48  human movement. Compared with optical motion tracking systems, IMUs are cheaper, can be
49  attached to the subject without the palpation of anatomical landmarks, do not suffer from line-of-
50  sight issues, are not limited to a specific target volume and can be used in field conditions. These
51  advantages come at a small cost of accuracy compared with optical motion tracking systems
52  (e.g., joint angle errors in the lower limbs are generally between 0 and 15 degrees but vary
53  strongly per joint and motion type) (Poitras et al., 2019), and IMU-specific error sources such as
54  drifting (Saber-Sheikh et al., 2010). In addition, IMUs can be coupled with electromyography
55  (EMG) electrodes to further enhances the versatility of these sensors for analyzing human
56  movement in sports and clinical applications (e.g., Cometa Srl, Cometa Systems | Wireless EMG
57  and IMU Solutions; Delsys Incorporated, Trigno Avanti Platform - Delsys; Noraxon USA,
58  Ultium EMG | Noraxon USA).
59
60  Analysis of kinematics of motion is typically done offline after measurement and data collection
61  in a process called inverse kinematics (IK). Recent years have seen progress in some real-time
62  IK (RTIK) analysis solutions and systems, but these studies (Bonnet et al., 2013; Borbély &
63  Szolgay, 2017; Falisse et al., 2018; Miezal et al., 2017; van den Bogert et al., 2013; Yi et al.,
64  2021) have mostly focused on specific marker sets and models and their generalization to
65  arbitrary measurement setups is difficult. For example, in real-time IMU-based applications,
66  Bonnet et al. (2013) estimated the RTIK of the trunk and lower limbs using a single IMU located
67  at the lower back. In another example, sensors containing IMUs and EMG electrodes, which
68  were placed on surface musculature, were used in a recent study (Yi et al., 2021) to calculate the
69  real-time kinematics and kinetics of the lower limb. However, these studies rely on complex
70  computational methods, making it difficult for others to repeat and adapt the experiment without
71  knowledge of sensor fusion or deep learning. The IMU-based RTIK solution by Miezal et al.
72  (2017) does not allow easy switching between musculoskeletal models or quantify how real-time
73  the solution is.
74
75  It has been shown that a full biomechanical analysis of joint and muscle function can be obtained
76  in real-time with a C/C++ software library (van den Bogert et al., 2013). The software library
77  was capable of reading marker data and performing IK and inverse dynamics on a full-body
78  model at more than 120 samples per second. However, the software is commercial and relies on

79   a single predefined model rather than a subject-specific or user-defined musculoskeletal model,
80   limiting its usefulness in research. Additionally, Falisse et al. (2018) showed that comparing its
81   outputs with those of another similar software (OpenSim 3.3) resulted in statistically significant
82   differences in joint kinematics, kinetics, and muscle forces, highlighting the dependency of the
83   output on the selected model. Hence, it is invaluable that the user can select or generate a model
84   that best fits to the application or research question. The OpenSim software for analyzing the
85   kinematics and dynamics of musculoskeletal systems (Delp et al., 2007) offers a solution to the
86   aforementioned issues because it is free and open source, it has a graphical user interface, and it
87   works with customizable musculoskeletal models. While its IK algorithm originally utilized only
88   marker-based motion capture data, since version 4.1 (Seth et al., 2018) it has been possible to
89   utilize IMU orientations as input to the IK algorithm of OpenSim to solve skeletal motion (i.e.,
90   joint angles). The IMU orientation-based IK algorithm minimizes the sum of squares of the
91   difference between experimental IMU orientations and corresponding segment orientations of
92   the model. Other software with capabilities similar to OpenSim (Damsgaard et al., 2006) exist,
93   but the fact that OpenSim is open-source makes it readily available to anyone, enables a variety
94   of community-made modifications and add-ons, and enables the user to view the source code to
95   better understand and troubleshoot the workings of the software. Its customizable models allow
96   the creation of personalized bony geometry, e.g., from imaging data (Valente et al., 2017), and
97   the inclusion of muscles in the models allows retrieval of muscle length and similar data that can
98   be used in further analyses. Therefore, although other software may offer kinematics or kinetics
99   based on biomechanical models, enabling real-time analysis with OpenSim has advantages in
100  verifiable and customizable motion analysis and research. While OpenSim has been used for
101  marker-based data to calculate inverse kinematics and inverse dynamics of human motion in
102  real-time (Pizzolato et al., 2017), OpenSim-based real-time calculation of IMU-based kinematics
103  with open-source code would enable others to adapt the solution for customizable motion
104  analysis in portable settings.
105
106  IMU-based RTIK solutions that utilize the OpenSim API been recently developed by Stanev et
107  al. (2021) and Slade et al. (2022). Slade et al. developed and tested an open-source IMU-based
108  IK system for a microcontroller that can be carried on the subject. Their implementation uses a
109  simplified musculoskeletal model and relaxed IK error tolerance to enable real-time IK with the
110  limited computational power of the microcontroller. With the full computational capacity of the
111  microcontroller (i.e., 4 threads), they could calculate full-body IK at a throughput of
112  approximately 20 operations per second and at a delay of approximately 200 ms. Stanev et al.
113  (2021) have published an open-source software framework that allows kinematical and
114  dynamical analysis of motion. Their software performs the analysis in real-time and supports
115  both IMU- and marker-based data.
116
117  The aim of this study was to develop a freely available software library that reads orientation
118  data from IMUs and calculates the IK on a user-given musculoskeletal model in real-time using

119 OpenSim 4.1 API. The development of this work was done prior to the publication of works by
120 Slade et al. (2022) and Stanev et al. (2021). Since release of the aforementioned software, our
121 work is not novel but represents an alternative implementation to solve the same problem. The
122 development occurred independently of these software and may therefore provide the
123 community additional value to these previous implementations. To assess the performance of our
124 software we quantified the software's IK execution time and throughput with different numbers
125 of processor threads calculating the IK and different numbers of IMUs, and to determine if
126 lowered input data frequency resulting from live visualization meaningfully affects calculated
127 ranges of motion (ROMs).

128

129 **Materials & Methods**

130

131 **Working principles of the software**

132

133 A software library for reading real-time IMU orientation data as quaternions and processing the
134 data to calculate the IK of a musculoskeletal model was developed using C++ and published on
135 GitHub (https://github.com/jerela/OpenSimLive). The software utilizes OpenSim 4.1 API to
136 invoke methods that calibrate the musculoskeletal model and perform the IK to solve joint angles
137 using quaternion-based orientation data from live measurements.

138

139 For each time point, the IK algorithm of OpenSim uses orientation information from all IMUs to
140 find the poses of individual bodies of the musculoskeletal model that, in the least squares sense,
141 minimize the error between experimental IMU orientations and the orientations of the
142 corresponding bodies. For information about the mathematics behind the IK algorithm, see (Delp
143 et al., 2007), where it is explained for markers instead of IMUs; for IMUs, IMU orientation error
144 is minimized instead of marker coordinate error. OpenSim's IK algorithm for IMU data is briefly
145 presented in supplementary materials, although we did not modify it.

146

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

156

157 The orientation information from IMUs is combined in a time series table that contains only one
158 sample, i.e., time point. The time series table is given to OpenSim's IK solver object, which

159   solves the IK for that time point. The process is repeated for each sample. The resulting time
160   series of joint angles can be saved in a text file in .mot format, which allows the output to be
161   viewed using the OpenSim graphical user interface. The read quaternions, and also EMG time
162   series in the case of Trigno Avanti sensors, can be saved in a text file for later offline analysis.
163
164   The working principle of the software library is straightforward. Producer-consumer thread
165   synchronization is used to get orientation data from IMUs. A producer thread and a consumer
166   thread run concurrently. The producer reads orientation data as quaternions from the IMUs and
167   saves it into a buffer that is shared between the threads. The consumer reads and removes data
168   from the buffer and assigns an IK task to a thread in a thread pool. This way there can be several
169   concurrent IK calculations, improving the throughput of the program. The maximum number of
170   concurrent IK threads is defined by the user. If that number is already active when the consumer
171   thread starts another IK thread, the consumer thread will wait for one of the IK threads in the
172   thread pool to finish. A diagram illustrating the workflow is shown in Fig. 1.
173
174   **[FIGURE 1 GOES HERE]**
175   Figure 1: A diagram illustrating the working principle of the inverse kinematics (IK) workflow.
176   Orientation data of inertial measurement units (IMUs) is read as quaternions by the producer
177   thread and saved to a buffer. Time values are saved to another buffer. The consumer thread reads
178   data from both buffers and initiates new threads that calculate IK based on the data. IK threads
179   output joint angle values for the model. Within an IK thread, the IK output can be sent to a
180   visualizer window. The visualization is based on the Simbody (i.e., the physics engine used by
181   OpenSim) API and not a part of our software library and hence not described here in more detail.
182   When the program finishes, the IK output frames can be sorted in a time-ascending order and
183   saved to file.
184
185   The software library has been tested to work on 64-bit Windows 7 and Windows 10 operating
186   systems. The source code of the software library is available on GitHub at
187   https://github.com/jerela/OpenSimLive.
188
189   **Experimental data**
190
191   Performance tests of the software library and error comparison of joint angles between real-time
192   IK and offline IK were done for walking data of a single subject. The subject was of legal age,
193   had no known musculoskeletal disorders or other conditions that affect gait, and gave their
194   written consent to participate in the study. A total of 12 IMUs (Xsens MTw Awinda, Enschede,
195   Netherlands) were strapped on the subject's upper arms, forearms, chest, pelvis, thighs, shanks,
196   and feet, as shown in supplementary information. We recorded ten trials while the subject
197   walked on an instrumented treadmill (Motek Medical B.V., Amsterdam, Netherlands) at a speed
198   of 1.5 m/s (5.4 km/h) and the IMUs transmitted their orientations at a sample rate of 60 Hz. Each

199    trial contained approximately a minute of gait data. The subject was instructed to take the
200    standard anatomical position at the beginning of each trial to calibrate the IMUs on the
201    musculoskeletal model as per the standard IMU calibration procedure of OpenSim (elaborated in
202    supplementary materials). When the subject was in the desired calibration pose, the user of our
203    software library pressed a key to calibrate the IMUs. No model scaling was done because we
204    tracked only sensor orientations which are independent of model dimensions.

205

206    RTIK was calculated during the measurements. The Simbody visualization of RTIK was
207    enabled, which reduced the throughput of RTIK from the 60 Hz sampling frequency of the IMUs
208    to 45 Hz on average on the desktop computer. This RTIK output was saved to file at the end of
209    each trial. Additionally, the time series of received IMU orientations as quaternions were saved
210    to a file at the full 60 Hz sampling frequency after each trial; the time series were used in
211    calculating execution times and throughputs of the software library.

212

213    **Performance tests**

214

215    To quantify the performance of the software library, we conducted performance tests using
216    computer-generated IMU orientations and IMU orientations from recorded human walking
217    (Table 1). Two performance measures were quantified: throughput and execution times of IK.
218    The throughput describes how many IK operations are calculated per second on average when
219    the communication between the producer and the consumer threads is included in the operation.
220    The execution times describe the delays that the calculation of IK for a single time frame takes,
221    i.e., how long it takes after retrieving IMU orientations from one time point to retrieve the
222    corresponding joint angles of the musculoskeletal model. Therefore, throughputs are increased
223    by using multiple IK threads but execution times are not. To retrieve time points, the
224    std::chrono::high_resolution_clock class was used in the C++ implementation.

225

226    Table 1: Conducted performance tests, the parameters that were varied in them, and their
227    purposes.
228    **[TABLE 1 GOES HERE]**
229    IMU = inertial measurement unit; MS = musculoskeletal

230

231    The tests were conducted with two musculoskeletal models, the Gait2392 lower extremities and
232    torso model (23 degrees of freedom, DOFs; referenced from here on as the lower body model)
233    (Anderson & Pandy, 1999, 2001; Delp et al., 1990; Yamaguchi & Zajac, 1989) and the Hamner
234    full-body model (29 DOFs, referenced from here on as the full-body model) (Hamner et al.,
235    2010). In tests involving the lower body model, data from 1 and 7 IMUs were used; with the full-
236    body model, data from 1, 7, and 12 IMUs were used (Table 2). All joint angles that were
237    unlocked in the model by default were solved by the IK algorithm of OpenSim, but meaningful
238    results were obtained only for joint angles defined by the available IMU data, such as joint

239   angles between two segments that both had an attached IMU sensor. Finally, the tests were
240   conducted with two computers, a laptop (HP EliteBook 8570w: Windows 10 Education 64-bit,
241   Intel Core i7-3740QM 2.70 GHz 8-CPU processor, 8192 MB RAM) and a desktop (Fujitsu
242   Celsius W550 Power: Windows 10 Education 64-bit, Intel Core i7-6700 3.40 GHz 8-CPU
243   processor, 32768 MB RAM). The laptop computer was used to examine if the software can
244   perform sufficiently using portable devices that allow the measurements to be performed outside
245   a laboratory environment. The desktop computer was used to examine how large difference the
246   increased computing power of a typical desktop compared to a typical laptop makes on the
247   performance of the software.
248
249   Table 2: The MS model segments whose inverse kinematics were calculated during performance
250   tests, presented by the number of IMUs whose data was utilized.
251   **[TABLE 2 GOES HERE]**
252   IMU: inertial measurement unit; MS = musculoskeletal
253
254   Execution times with computer-generated IMU orientations were measured with random unit
255   quaternions. The quaternions represented random and sometimes unrealistic poses, which varied
256   greatly between time frames and were time-consuming to calculate. Therefore, the measured
257   execution times represent performance that is worse than in normal human motion, i.e., they
258   were a benchmark of unrealistically poor performance. The execution times were measured with
259   data from different numbers of IMUs (1 and 7 for the lower body model and 1, 7, and 12 for the
260   full-body model). Each execution time measurement lasted until 10 000 IK operations were
261   calculated. We report the mean, standard deviation and 95% confidence interval of the 10 000
262   execution times.
263
264   Finally, IMU orientations from real human walking were used to calculate throughputs and
265   execution times. Performance was measured with pre-recorded quaternion orientations from ten
266   one-minute walking trials. Although the throughputs and execution times were calculated after
267   the walking trials had been recorded, the performance tests were designed to simulate real-time
268   measurement by feeding the quaternion data into the test environment one data frame at a time.
269   The use of pre-recorded quaternion orientations enabled measuring throughputs above the
270   sampling frequency of the IMUs; otherwise, the throughput would be limited to the sampling
271   rate of the IMUs because IK operations could only be solved at the rate the IMU orientations are
272   received. The tests were conducted with different numbers of IMUs (1 and 7 for the lower body
273   model and 1, 7, and 12 for the full-body model). Furthermore, different numbers of IK threads
274   (1, 2, 4, 6, 8) were used during throughput tests. The performance tests on real walking data
275   allowed us to evaluate performance during a common human motion measurement.
276
277   **Error comparison of joint angles**
278

279 Our software library includes an option to visualize motion like in OpenSim GUI by invoking
280 methods from the Simbody API (namely, from the SimTK::Visualizer class), which OpenSim
281 API is built upon. The use of the Simbody visualizer in a real-time IK thread slows the thread
282 noticeably, and if the throughput drops below the sampling frequency of the IMUs, the IK
283 threads skip some orientation frames from the IMUs in a real-time measurement. This frame
284 drop may negatively affect the accuracy of gait parameters derived from the IK solution.
285 Although not a core feature of our software library, the Simbody visualization during RTIK may
286 be of interest of some. Thus, we evaluated the effect of the frame drops on a kinematic variable
287 that is often of interest and potentially affected by the frame drops, namely the ranges of motion
288 (ROMs) of the joint angles. To this end we compared ROM between visualized RTIK
289 (calculated in this case at approximately 45 Hz) and offline IK at the 60 Hz sample rate of the
290 IMUs. Offline IK was calculated from IMU orientations that were stored after each of the ten
291 walking trials; RTIK with Simbody visualization was calculated on the desktop computer during
292 each walking trial. Orientations from all 12 IMUs were included, and the full-body model was
293 used in the analysis. The measured motion exerted 26 of the model's DOFs. For determining
294 ROMs, the IK data was divided into periods based on the cyclical nature of the flexion-extension
295 angle of the right knee. For each of the resulting 715 gait cycles (total from 10 walking trials),
296 the difference between the highest and the lowest value of each joint angle was taken as its
297 ROM. The results were reported as the mean absolute error (MAE) and 95% confidence interval
298 (95% CI) between offline IK and RTIK ROM. Note that we calculated errors to evaluate the
299 effect of visualization on ranges of motion and to demonstrate how the software library works,
300 not to validate IMU-based IK. For IMU-based IK validation, see other studies such as (al Borno
301 et al., 2022; Tagliapietra et al., 2018).
302
303 **Results**
304
305 **Performance tests**
306
307 Performance test results showed that increasing model complexity and the number of IMUs for
308 orientation tracking increased execution times (Fig. 2) and decreased IK throughput (Fig. 3).
309 Increasing the number of IK calculating threads increased throughput (Fig. 3). The desktop
310 computer always had lower execution times and higher throughput than the laptop computer in
311 the same performance tests (Fig. 2 and Fig. 3).
312
313 Using computer-generated random unit quaternions, the means and standard deviations of
314 execution times (operation delays) increased with increasing number of IMUs similarly on both
315 the desktop and the laptop (Table 3, Fig. 4). With one IMU, the full-body model was 60-65%
316 slower than the lower body model and had more variation in execution times. The mean
317 execution times were approximately 25% longer on the laptop than on the desktop.
318

319    Table 3: Mean, standard deviation (STD) and 95% confidence interval (CI) of execution times of
320    a single inverse kinematics (IK) operation. The values are calculated over 10 000 IK operations
321    for two different musculoskeletal models, two different computers and 1, 7 or 12 inertial
322    measurement units (IMUs). Randomly selected unit quaternions were used as IMU orientations.
323    **[TABLE 3 GOES HERE]**
324    CI = confidence interval; IK = inverse kinematics; IMU = inertial measurement unit; STD =
325    standard deviation
326
327    Execution times calculated from human walking (Fig. 2) are shorter than the corresponding
328    execution times from computer-generated data (Table 3, Fig. 4) and remain below 55 ms even
329    with 12 IMUs and the full-body model. Performance with real walking data follows the same
330    patterns as with computer-generated data. Execution times increase (Fig. 2) when the number of
331    IMUs or model complexity (number of DOFs) is increased.
332
333    Execution times below 30 ms using the laptop and below 25 ms using the desktop were reached
334    with the lower-body model and 7 IMUs (Fig. 2). Furthermore, even with the laptop, full body
335    kinematics can be calculated with execution times below 60 ms, and provided at least four IK
336    threads are used, with throughputs above 50 Hz (Fig. 3).
337
338    **[FIGURE 2 GOES HERE]**
339    Figure 2: Execution times and standard deviations (shaded area) of a single inverse kinematics
340    operation with respect to the number of inertial measurement units (IMUs), i.e., number of
341    segments with measured orientation data. The execution times are presented as mean over ten
342    trials for two different musculoskeletal models (lower body model and full-body model) using
343    one, seven, or 12 IMUs. IMU quaternions orientations were retrieved from previously recorded
344    walking trials.
345
346    Throughput tests show that on the laptop, human walking can be solved at more than 60 IK
347    operations per second when using eight IK threads, the full-body model and 12 IMUs (Fig. 3).
348    The minimal IMU setup to record motion of all lower-body DOFs, 7 IMUs with the lower-body
349    model, reached a throughput of 130 with eight IK threads. On the desktop, the corresponding
350    throughputs were 90 and 180, respectively.
351
352    **[FIGURE 3 GOES HERE]**
353    Figure 3: Inverse kinematics (IK) throughput with respect to the number of IK threads used,
354    measured on a desktop computer (left) and a laptop computer (right). The throughputs are
355    presented as mean over ten trials. The measurements were repeated with two different
356    musculoskeletal models, and using one, seven or 12 inertial measurement units (IMUs). IMU
357    quaternion orientations were retrieved from previously recorded walking trials.
358
359    **[FIGURE 4 GOES HERE]**

360    Figure 4: Execution times and standard deviations (shaded area) of a single inverse kinematics
361    operation with respect to the number of inertial measurement units (IMUs), i.e., number of
362    segments with random orientation data. The values are calculated over 10 000 IK operations for
363    two different musculoskeletal models, two different computers and 1, 7 or 12 inertial
364    measurement units (IMUs). Randomly selected unit quaternions were used as IMU orientations.
365

366    **Error comparison of joint angles**
367    Enabling Simbody visualization during the measurement session reduced IK throughput but,
368    compared to IK at full 60 Hz sampling frequency of the IMUs, caused only minimal differences
369    in calculated ranges of motion (Fig. 5). The mean ROM error for all DOFs was 0.0675 degrees.
370    The greatest MAE in ROM was observed in ankle joints (up to 360% of the mean for all joints),
371    followed by the left hip joint (Fig. 5). Pelvis and upper extremities had the smallest ROM error.
372    All MAEs remained below 0.3 degrees.
373

374    **[FIGURE 5 GOES HERE]**
375    Figure 5: Mean absolute error (MAE) between real-time inverse kinematics and offline inverse
376    kinematics ranges of motion (ROMs) of the exerted degrees of freedom of the musculoskeletal
377    model and the 95% confidence interval of the error. Solid bars show MAEs of ROMs for each
378    exerted degree of freedom. Confidence intervals are shown as error bars centered on the top of
379    the MAE bars.
380

381    **Discussion**
382    We present an open-source software library for the real-time inverse kinematical analysis of
383    IMU data with user-defined musculoskeletal models using OpenSim 4.1. Full-body IK can be
384    calculated for random orientations in less than 100 ms; using real walking data, it can be done in
385    less than 60 ms. On a desktop computer, the software library can solve RTIK at 180 samples per
386    second while tracking the pelvis and lower extremities and at 90 samples per second while
387    tracking the full-body kinematics. On a laptop computer, the corresponding throughputs were
388    130 and 60 samples per second, respectively. Using 12 IMUs to track walking and visualizing
389    the results on a full-body running model, RTIK was solved at 45 samples per second on a
390    desktop computer. The drop from the IMU output sampling rate of 60 Hz resulted in a minimal
391    difference in calculated joint ROMs (<0.3 degrees). The software library allows the use of RTIK
392    virtually without limitations due to location or environment, which opens possibilities for a
393    variety of applications including rehabilitation, ergonomics, and human-machine interfaces for
394    controlling collaborative robots. Observing the movement of interest in a natural environment is
395    important because a laboratory setting may affect how a person moves (Friesen et al., 2020).
396

397    **Performance tests**
398

399  We investigated the execution times and throughputs of the IMU-based IK to determine if the
400  output can be considered real-time. Pizzolato et al. (2017) used an execution time of 75 ms as the
401  threshold for a real-time system. It was based on a study by Kannape and Blanke (2013) in which
402  the subjects were able to identify the displayed motion as self-generated in real-time in over 80%
403  of the cases if the delay in motion display was less than 75 ms. Even with a delay of 210 ms,
404  subjects identified the visualized motion as self-generated in real-time in 50% of the cases.
405  Borbély and Szolgay (2017) noted that the IK algorithm of OpenSim 3.3 had an execution time
406  of about 145 ms, thus calculating IK at about 7 Hz and "falling behind the generally accepted
407  practice in human movement recording of at least 50 Hz". Therefore, a real-time application
408  should achieve IK throughput of 50 operations per second with an execution time below 75 ms
409  for any single operation. With our software library, we aimed to achieve this target by using
410  multithreading and the IK algorithm of OpenSim 4.1.
411
412  Another interesting finding by Kannape and Blanke (2013) was that subjects modulated their
413  stride based on the delay between the motion and its visualization. Therefore, it is important to
414  minimize the delay when preparing a real-time measurement setup to prevent subjects from
415  altering their gait characteristics based on delayed visual feedback.
416
417  Live visualization is unnecessary in applications where IK is an intermediate output that is used
418  to estimate contact forces, instruct a robot arm in rehabilitation applications or calculate gait
419  parameters, to name a few examples. Thus, the performance tests were designed so that they
420  evaluate only the performance of IK, which is the core feature of the software library. Although
421  we adopt the visualization-based 75 ms criterion for real-time motion from Kannape and Blanke,
422  our performance tests were conducted without visualization. Our software library relies on
423  Simbody visualization and lacks an elegant visualization solution of its own, which is a
424  limitation that should be acknowledged.
425
426  The performance tests show that IK throughput is more sensitive to the number of IMUs than the
427  DOFs of the model, although model complexity also increases computational load because joint
428  angles with no experimental data to solve them are still considered in the IK algorithm.
429  Execution times clearly increased with the number of IMUs, although model complexity also
430  affected them noticeably. Therefore, minimally enough IMUs and DOFs should be chosen to
431  enable high throughput and short execution times in real-time.
432
433  The performance of the software library would benefit from improvements particularly in
434  solving IK and visualizing the results. Because a new quaternion time series table is created for a
435  single IMU data point every time an IK frame needs to be solved, the software library is
436  computationally much heavier than it would be if a single time series table could act as a
437  permanent buffer that is updated with new IMU data points. This implementation was not
438  possible using the OpenSim API during the development stage of our software library but

439     appears to be supported in the latest version of the API. Furthermore, the data points in real-time
440     IK may be solved in inconsistent order depending on how quickly each IK thread finishes.
441     Expired IK frames may be omitted if the real-time IK is used as input in further real-time
442     analyses but the missing data points may require interpolation or other consideration, e.g., if
443     filtering the IK is required. Thus, future development of the software library could implement
444     existing interpolation functionalities from the OpenSim API. Additionally, if the visualizer
445     would be implemented in its own thread, it could be used without blocking the IK solving
446     threads and it would enable greater IK throughput while visualization is enabled. This
447     improvement would also enable the solving of joint angles at a high frame rate and visualizing
448     them at a reduced frame rate for performance reasons.
449
450     Finally, it should be noted that the software library was only tested on the Windows operating
451     system, and the performance tests were conducted on its operation as a whole. Thus, no proper
452     profiling analysis was done to discover which parts of the software library have potential for
453     performance optimization.
454
455     **Execution times of the IK operation**
456
457     Real motion, such as walking, contains a combination of different orientations, most of which
458     are within a typical model's joint angle boundaries. Randomly generated unit quaternions used in
459     execution time tests often result in unrealistic poses. As a result, the IK based on randomized unit
460     quaternions is heavier to calculate than the average orientations during walking, or any typical
461     human motion. Therefore, the execution times from computer-generated data can be interpreted
462     as the worst performance when analyzing human motion without live visualization.
463     Consequently, if the execution times with computer-generated data are sufficiently small for
464     real-time analysis, then any realistic motion should be processed with smaller or equal execution
465     times. On the other hand, we can assume that most human motions can be analyzed with
466     execution times like those from real walking data because joints angles are likely to change at a
467     similar rate and exert constraints similarly.
468
469     For both models using computer-generated random unit quaternions, the standard deviations of
470     the execution times are on the same scale as the mean execution times, implying that there is
471     great variation in the execution time. The randomized nature of the used quaternion orientations
472     is a likely contributor to the high standard deviation, because randomized orientations
473     occasionally lead to strange segment orientation combinations that do not reflect valid human
474     motion and take the IK algorithm a long time to solve. This is further supported by the much
475     smaller standard deviation in real walking data (Fig. 2).
476
477     Before conclusions are drawn from the computer-generated random unit quaternion-based
478     execution times, it should be acknowledged that the results varied somewhat on repeated runs

479 with the same parameters. This implies that running the tests more than 10 000 iterations could
480 improve the precision of the results. However, because the purpose of the computer-generated
481 random unit quaternions was to estimate worst-case performance for the hardware of the tested
482 computers and the effects of differing the number of DOFs of the model and segments with IMU
483 orientations, which they show well, we chose not to repeat the tests with more iterations.
484
485 For both computer-generated (Table 3) and real walking data (Fig. 2), with one and seven IMUs,
486 the execution times are shorter and vary less for the lower body model than for the full-body
487 model. Both the mean execution times and the standard deviations are smaller on the desktop
488 than on the laptop. However, the execution times vary less with 12 than with seven IMUs on the
489 full-body model.
490
491 Because execution times calculated on real walking data remained below 55 ms (Fig. 2), the
492 software library is capable of real-time inverse kinematics analysis of the full body even on a
493 laptop. Using computer-generated random unit quaternions, the 95% confidence intervals of
494 execution times are roughly 1% of the mean execution time in all cases, meaning that the
495 execution times stay consistently below 75 ms except when 12 IMUs are used. In that case, the
496 execution times stay consistently below 100 ms, which is still less than half of the 210 ms delay
497 that marks 50% confidence in perceiving motion as real-time (Kannape & Blanke, 2013).
498 Therefore, while RTIK is clearly possible with normal walking, some complicated motions may
499 result in longer execution times but could nonetheless be analyzed practically in real-time.
500 Because the execution times represent the minimum delay from the orientation data retrieval to
501 the moment we can visualize or further analyze the IK output, the number of IMUs in a real-time
502 measurement should be chosen considering the delays that are acceptable for the application.
503
504 **Throughputs**
505
506 Fig. 3 shows that increasing the number of concurrent processor threads increased the throughput
507 until about eight threads, which was the maximum CPU core number for both computers.
508 Increasing the number of IK threads further had no meaningful effect on the throughput, which
509 was also observed in an earlier study on RTIK (Pizzolato et al., 2017). The throughput plateau
510 resulted from CPU utilization of the computer reaching 100% and is thus hardware dependent.
511
512 The increase in throughput by multithreading is especially large with a small number of threads
513 and one IMU. For example, throughput increases from less than 25 to more than 150 when the
514 number of IK threads increases from one to two on the laptop. Doubled computational capacity
515 alone cannot explain the increase in throughput. The effect is also present on the desktop.
516 Furthermore, the relationship between the throughput and the number of IK threads is clearly
517 nonlinear whereas an earlier RTIK study found it almost linear (Pizzolato et al., 2017). No

518  explanation for this phenomenon was found, but it should be addressed in the future
519  development of the software library.
520
521  For 1 IMU and 4 or more concurrent threads, the lower body model with 23 DOFs had
522  approximately 20% higher throughput than the full-body model with 29 DOFs. For 7 IMUs, the
523  lower body model throughput was 40% higher than that of the full-body model. Therefore,
524  model selection has a noticeable effect on the performance of RTIK and the model with the
525  smallest sufficient number of DOFs should be chosen to reach maximal RTIK performance.
526
527  The software library is clearly capable of calculating IK at a higher rate than the lower limit of
528  50 Hz named by Borbély and Szolgay (2017), but requires multithreading to reach it with
529  complex musculoskeletal models. For portable real-time gait measurements, a laptop should be
530  able to achieve sufficiently high IK throughput when 7 IMUs are used. We reached a throughput
531  of 130 samples per second in such a scenario, although others' results will vary depending on the
532  hardware of the laptop. Nonetheless, a throughput of 130 samples per second should be sufficient
533  for most gait analysis applications.
534
535  **Error comparison**
536
537  Because loss of frequency may lead to reduced accuracy in measuring sharp peaks in joint
538  angles, joints where motion direction changes fast are likely to have high ROM error (Fig. 5).
539  During walking, ankle flexion (ankle_angle_r and ankle_angle_l) undergoes fast changes, which
540  explains why its ROM error stands out. However, because all ROM errors remain consistently
541  below 0.3 degrees, the effect of the drop in visualized IK from 60 to 45 Hz on ROM is very
542  small.
543
544  The ROM error of left hip adduction stands out because it is visibly higher than that of the right
545  hip. The error is caused by an artifact in IMU signal that caused the left leg to be violently jerked
546  to the right after the left toe-off phase. The artifact is probably caused by the distortion of
547  magnetic fields near the ferromagnetic laboratory hardware, which the left leg was closer to.
548
549  **Comparison to other solutions for IMU-based RTIK using OpenSim**
550
551  In the introduction, we briefly presented some existing solutions for real-time IMU-based IK.
552  Two of them, OpenSenseRT (Slade et al., 2022) and the solution by Stanev et al. ( 2021), utilize
553  the OpenSim API like our software library. OpenSenseRT had comparable execution times and
554  throughputs but to optimize performance, they loosened IK solver tolerance and simplified their
555  musculoskeletal model by removing muscles and locking unused joints. Therefore, the
556  performance results are not easily comparable to ours, where many joints remained unlocked
557  despite not having experimental IMU data to solve them uniquely and muscles of the model were

558  left as they are (although muscles are not part of IK calculations, we made preliminary and
559  unreported observations that OpenSim's standard IMU-based IK throughput is slightly lower on
560  a model with muscles compared with a model without muscles). OpenSenseRT is a good
561  solution for IMU-based IK where the Raspberry Pi computing unit is carried with the subject,
562  while our solution relies on a computer separate from the subject, limiting its applicability
563  slightly but allowing higher performance due to numerous options of laptop hardware and
564  requiring less work to use different musculoskeletal models. The solution by Stanev et al. allows
565  the use of different musculoskeletal models easily. They support both marker-based and IMU-
566  based IK and use lower-level API functions to calculate IK quickly. Furthermore, their solution
567  goes beyond IK, enabling even real-time inverse dynamics and joint reaction force analysis.
568  Their software architecture relies on two threads: one to collect orientation from IMUs or marker
569  positions and then perform IK, and another to perform preprocessing and further musculoskeletal
570  analysis in real-time. Our solution has one thread for collecting orientation from IMUs and a
571  user-defined number of threads for IK. Therefore, although the solution by Stanev et al. is
572  superior to ours in terms of number of features, our solution may allow higher throughputs due to
573  the variable number of IK threads that is only limited by computer hardware.
574

575  ## Conclusions
576  An open-source software library that builds upon the widely used OpenSim software was
577  developed and published for IMU-based RTIK. It allows the joint angles of any OpenSim-
578  compatible musculoskeletal model to be analyzed in real-time. While another real-time solution
579  was concurrently and independently developed by Stanev et al. (2021), its IK calculation does
580  not utilize multithreading, which may limit its throughput, although its IK calculation relies on
581  lower-level API classes that are faster than those used by the software library developed in this
582  study when a single thread is used. The authors encourage others to contribute to the open-source
583  project. The development of the software library will closely follow the development of
584  OpenSim to utilize its built-in functionality for processing live data. The software library could
585  be utilized in real-time estimation of joint moments, muscle forces, and joint contact forces based
586  only on IMU data. Ground reaction forces and moments and kinematics are required for solving
587  the equations of motion for the musculoskeletal model using inverse dynamics. It has been
588  shown that ground reaction forces and moments can be predicted from IMU-derived kinematics
589  (Karatsidis et al., 2017; Stanev et al., 2021). Moreover, estimation of muscle forces using
590  optimization techniques uses kinematics and inverse dynamics estimates of joint moments as
591  inputs and estimates of joint contact forces can be derived based on kinematics, inverse
592  dynamics, and muscle forces. Hence, IMUs could be potentially used for the real-time estimation
593  of musculoskeletal dynamics outside the laboratory and implemented in the software library in
594  the future. Another interesting future application is the use of RTIK output together with EMG.
595  Thus, combining IK output with EMG in real-time may provide interesting possibilities for
596  estimating muscle forces and musculoskeletal loading using EMG driven musculoskeletal

597  simulations (Sartori et al., 2011), for biofeedback to optimize rehabilitation or ergonomics or for
598  biosignal-based operating systems.

599

600

601  **References**

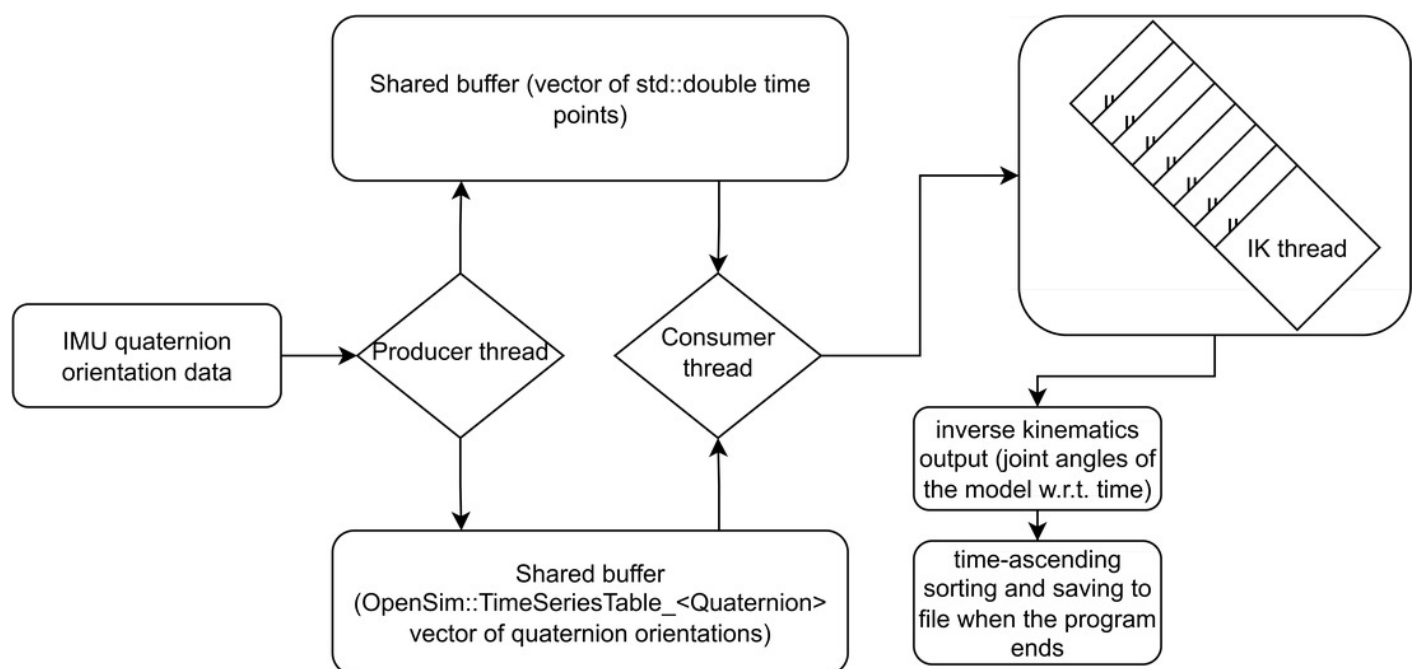
602  al Borno, M., O'Day, J., Ibarra, V., Dunne, J., Seth, A., Habib, A., Ong, C., Hicks, J.,
603      Uhlrich, S., & Delp, S. (2022). OpenSense: An open-source toolbox for inertial-
604      measurement-unit-based measurement of lower extremity kinematics over long
605      durations. *Journal of NeuroEngineering and Rehabilitation*, *19*(1).
606      https://doi.org/10.1186/s12984-022-01001-x
607  Anderson, F. C., & Pandy, M. G. (1999). A dynamic optimization solution for vertical
608      jumping in three dimensions. *Computer Methods in Biomechanics and Biomedical
609      Engineering*, *2*(3), 201–231. https://doi.org/10.1080/10255849908907988
610  Anderson, F. C., & Pandy, M. G. (2001). Dynamic optimization of human walking. *Journal
611      of Biomechanical Engineering*, *123*(5), 381–390. https://doi.org/10.1115/1.1392310
612  Bonnet, V., Mazza, C., Fraisse, P., & Cappozzo, A. (2013). Real-time estimate of body
613      kinematics during a planar squat task using a single inertial measurement unit. *IEEE
614      Transactions on Biomedical Engineering*, *60*(7), 1920–1926.
615      https://doi.org/10.1109/TBME.2013.2245131
616  Borbély, B. J., & Szolgay, P. (2017). Real-time inverse kinematics for the upper limb: A
617      model-based algorithm using segment orientations. *BioMedical Engineering Online*,
618      *16*(1). https://doi.org/10.1186/s12938-016-0291-x
619  Cometa. (n.d.). *Cometa Srl, Cometa Systems | Wireless EMG and IMU solutions*.
620      Retrieved February 10, 2021, from https://www.cometasystems.com/
621  Damsgaard, M., Rasmussen, J., Christensen, S. T., Surma, E., & de Zee, M. (2006).
622      Analysis of musculoskeletal systems in the AnyBody Modeling System. *Simulation
623      Modelling Practice and Theory*, *14*(8), 1100–1111.
624      https://doi.org/10.1016/j.simpat.2006.09.001
625  Delp, S. L., Anderson, F. C., Arnold, A. S., Loan, P., Habib, A., John, C. T., Guendelman,
626      E., & Thelen, D. G. (2007). OpenSim: Open-source software to create and analyze
627      dynamic simulations of movement. *IEEE Transactions on Biomedical Engineering*,
628      *54*(11), 1940–1950. https://doi.org/10.1109/TBME.2007.901024
629  Delp, S. L., Loan, J. P., Hoy, M. G., Zajac, F. E., Topp, E. L., & Rosen, J. M. (1990). An
630      Interactive Graphics-Based Model of the Lower Extremity to Study Orthopaedic
631      Surgical Procedures. *IEEE Transactions on Biomedical Engineering*, *37*(8), 757–767.
632      https://doi.org/10.1109/10.102791
633  Delsys. (n.d.). *Delsys Incorporated, Trigno^TM Avanti Platform - Delsys*. Retrieved February
634      10, 2021, from https://www.delsys.com/trigno/
635  Falisse, A., van Rossom, S., Gijsbers, J., Steenbrink, F., van Basten, B. J. H., Jonkers, I.,
636      van den Bogert, A. J., & de Groote, F. (2018). OpenSim versus human body model: A
637      comparison study for the lower limbs during gait. *Journal of Applied Biomechanics*,
638      *34*(6), 496–502. https://doi.org/10.1123/jab.2017-0156

639  Friesen, K. B., Zhang, Z., Monaghan, P. G., Oliver, G. D., & Roper, J. A. (2020). All eyes
640      on you: how researcher presence changes the way you walk. *Scientific Reports*,
641      *10*(1). https://doi.org/10.1038/s41598-020-73734-5
642  Hamner, S. R., Seth, A., & Delp, S. L. (2010). Muscle contributions to propulsion and
643      support during running. *Journal of Biomechanics*, *43*(14), 2709–2716.
644      https://doi.org/10.1016/j.jbiomech.2010.06.025
645  Kannape, O. A., & Blanke, O. (2013). Self in motion: Sensorimotor and cognitive
646      mechanisms in gait agency. *Journal of Neurophysiology*, *110*(8), 1837–1847.
647      https://doi.org/10.1152/jn.01042.2012
648  Karatsidis, A., Bellusci, G., Schepers, H. M., de Zee, M., Andersen, M. S., & Veltink, P. H.
649      (2017). Estimation of ground reaction forces and moments during gait using only
650      inertial motion capture. *Sensors (Switzerland)*, *17*(1).
651      https://doi.org/10.3390/s17010075
652  Miezal, M., Taetz, B., & Bleser, G. (2017). Real-time inertial lower body kinematics and
653      ground contact estimation at anatomical foot points for agile human locomotion.
654      *Proceedings - IEEE International Conference on Robotics and Automation*, 3256–
655      3263. https://doi.org/10.1109/ICRA.2017.7989371
656  Noraxon. (n.d.). *Noraxon USA, Ultium EMG | Noraxon USA*. Retrieved February 10, 2021,
657      from https://www.noraxon.com/our-products/ultium-emg/
658  Paulich, M., Schepers, M., Rudigkeit, N., & Bellusci, G. (2018). Xsens MTw Awinda:
659      Miniature Wireless Inertial Motion Tracker for Highly Accurate 3D Kinematic
660      Applications. In *Xsens Technologies* (Issue May).
661  Pizzolato, C., Reggiani, M., Modenese, L., & Lloyd, D. G. (2017). Real-time inverse
662      kinematics and inverse dynamics for lower limb applications using OpenSim.
663      *Computer Methods in Biomechanics and Biomedical Engineering*, *20*(4), 436–445.
664      https://doi.org/10.1080/10255842.2016.1240789
665  Poitras, I., Dupuis, F., Bielmann, M., Campeau-Lecours, A., Mercier, C., Bouyer, L., & Roy,
666      J.-S. (2019). Validity and Reliability of Wearable Sensors for Joint Angle Estimation: A
667      Systematic Review. *Sensors*, *19*(7), 1555. https://doi.org/10.3390/s19071555
668  Saber-Sheikh, K., Bryant, E. C., Glazzard, C., Hamel, A., & Lee, R. Y. W. (2010).
669      Feasibility of using inertial sensors to assess human movement. *Manual Therapy*,
670      *15*(1), 122–125. https://doi.org/10.1016/j.math.2009.05.009
671  Sartori, M., Reggiani, M., Lloyd, D. G., & Pagello, E. (2011). A neuromusculoskeletal model
672      of the human lower limb: Towards EMG-driven actuation of multiple joints in powered
673      orthoses. *IEEE International Conference on Rehabilitation Robotics*.
674      https://doi.org/10.1109/ICORR.2011.5975441
675  Seth, A., Hicks, J. L., Uchida, T. K., Habib, A., Dembia, C. L., Dunne, J. J., Ong, C. F.,
676      DeMers, M. S., Rajagopal, A., Millard, M., Hamner, S. R., Arnold, E. M., Yong, J. R.,
677      Lakshmikanth, S. K., Sherman, M. A., Ku, J. P., & Delp, S. L. (2018). OpenSim:
678      Simulating musculoskeletal dynamics and neuromuscular control to study human and
679      animal movement. *PLoS Computational Biology*, *14*(7).
680      https://doi.org/10.1371/journal.pcbi.1006223

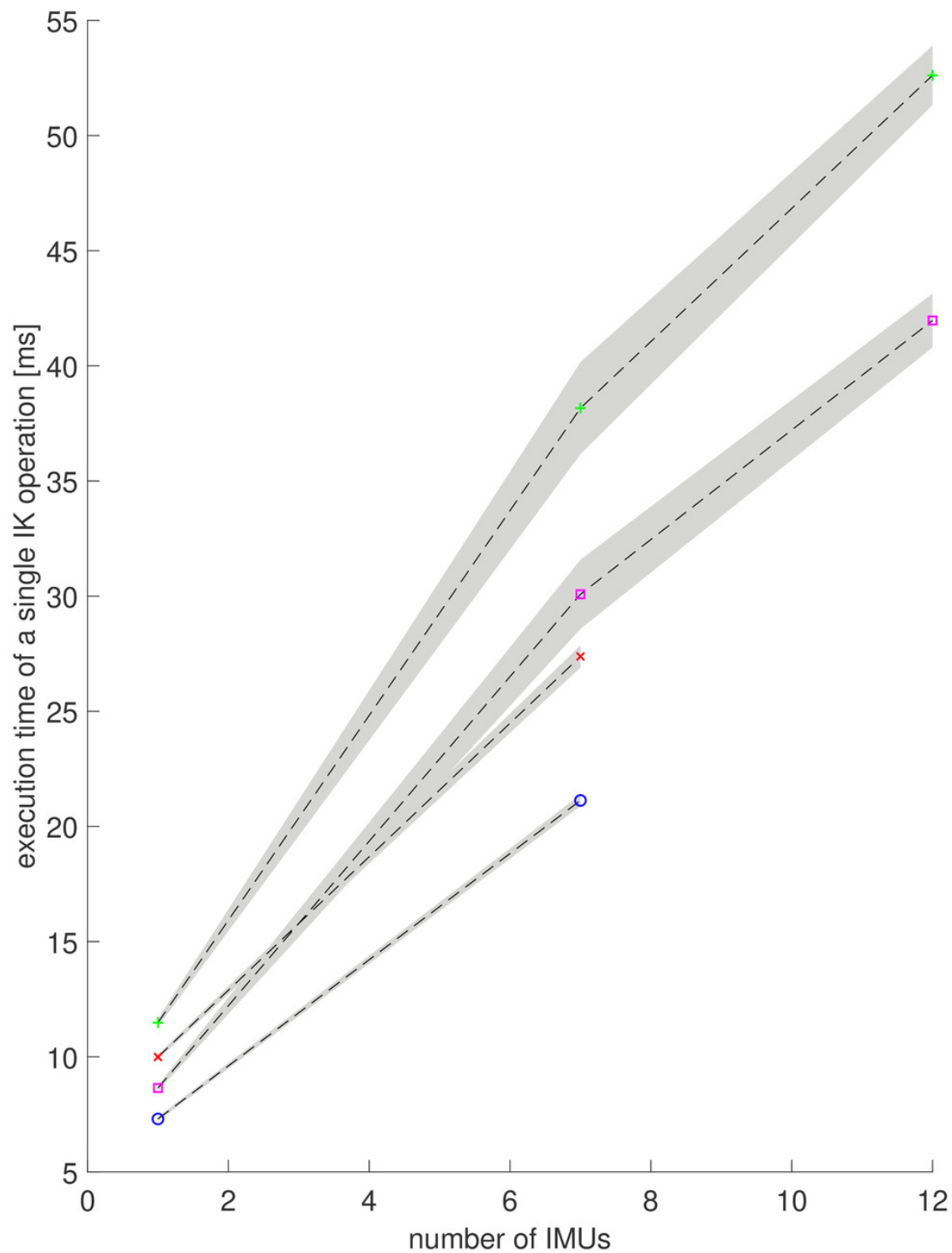
681 Slade, P., Habib, A., Hicks, J. L., & Delp, S. L. (2022). An Open-Source and Wearable
682   System for Measuring 3D Human Motion in Real-Time. *IEEE Transactions on*
683   *Biomedical Engineering*, *69*(2), 678–688. https://doi.org/10.1109/TBME.2021.3103201
684 Stanev, D., Filip, K., Bitzas, D., Zouras, S., Giarmatzis, G., Tsaopoulos, D., & Moustakas,
685   K. (2021). Real-time musculoskeletal kinematics and dynamics analysis using marker-
686   and imu-based solutions in rehabilitation. *Sensors*, *21*(5), 1–20.
687   https://doi.org/10.3390/s21051804
688 Tagliapietra, L., Modenese, L., Ceseracciu, E., Mazzà, C., & Reggiani, M. (2018).
689   Validation of a model-based inverse kinematics approach based on wearable inertial
690   sensors. *Computer Methods in Biomechanics and Biomedical Engineering*, *21*(16),
691   834–844. https://doi.org/10.1080/10255842.2018.1522532
692 Valente, G., Crimi, G., Vanella, N., Schileo, E., & Taddei, F. (2017). NMSBUILDER:
693   Freeware to create subject-specific musculoskeletal models for OpenSim. *Computer*
694   *Methods and Programs in Biomedicine*, *152*, 85–92.
695   https://doi.org/10.1016/j.cmpb.2017.09.012
696 van den Bogert, A. J., Geijtenbeek, T., Even-Zohar, O., Steenbrink, F., & Hardin, E. C.
697   (2013). A real-time system for biomechanical analysis of human movement and
698   muscle function. *Medical and Biological Engineering and Computing*, *51*(10), 1069–
699   1077. https://doi.org/10.1007/s11517-013-1076-z
700 Yamaguchi, G. T., & Zajac, F. E. (1989). A planar model of the knee joint to characterize
701   the knee extensor mechanism. *Journal of Biomechanics*, *22*(1), 1–10.
702   https://doi.org/10.1016/0021-9290(89)90179-6
703 Yi, C., Jiang, F., Bhuiyan, M. Z. A., Yang, C., Gao, X., Guo, H., Ma, J., & Su, S. (2021).
704   Smart healthcare-oriented online prediction of lower-limb kinematics and kinetics
705   based on data-driven neural signal decoding. *Future Generation Computer Systems*,
706   *114*, 96–105. https://doi.org/10.1016/j.future.2020.06.015
707
708

# Figure 1

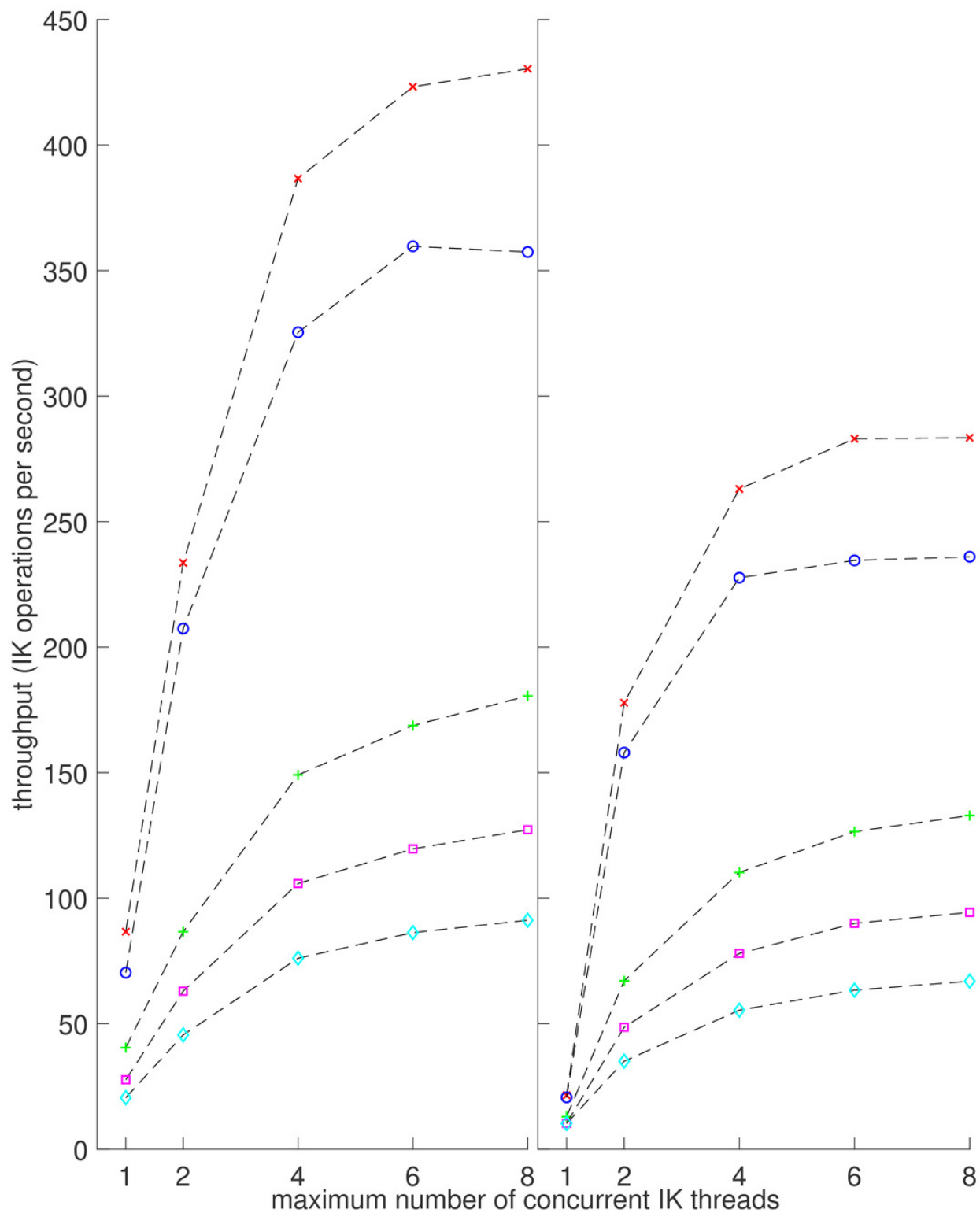
A diagram illustrating the working principle of the inverse kinematics (IK) workflow.

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. Within an IK thread, the IK output can be sent to a visualizer window. The visualization is based on the Simbody (i.e., the physics engine used by OpenSim) API and not a part of our software library and hence not described here in more detail. When the program finishes, the IK output frames can be sorted in a time-ascending order and saved to file

# Figure 2

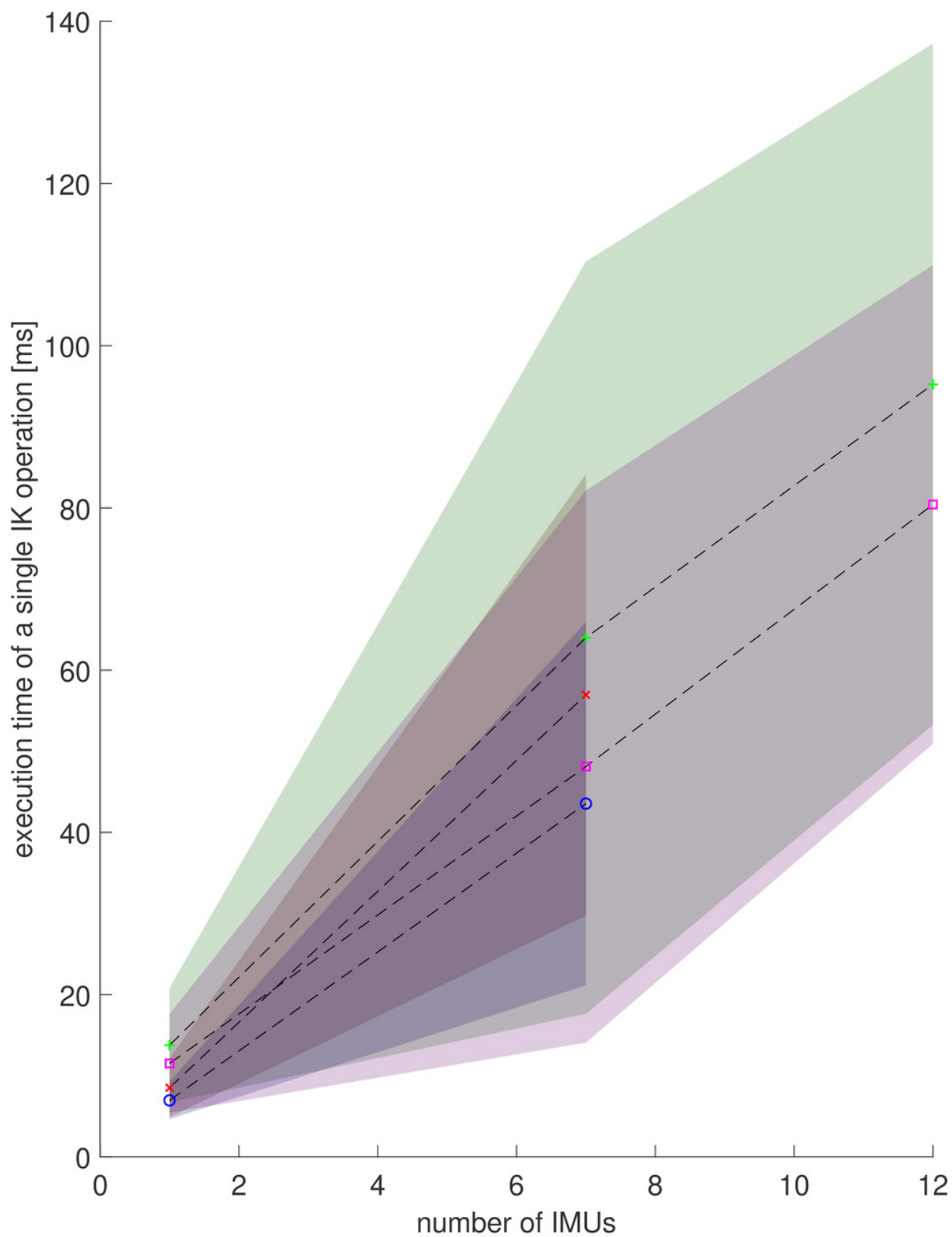
Execution times and standard deviations (shaded area) of a single inverse kinematics operation with respect to the number of inertial measurement units (IMUs).

The execution times are presented as mean over ten trials for two different musculoskeletal models (lower body and torso model and full-body model) using one, seven, or 12 IMUs. IMU quaternions orientations were retrieved from previously recorded walking trials.

# Figure 3

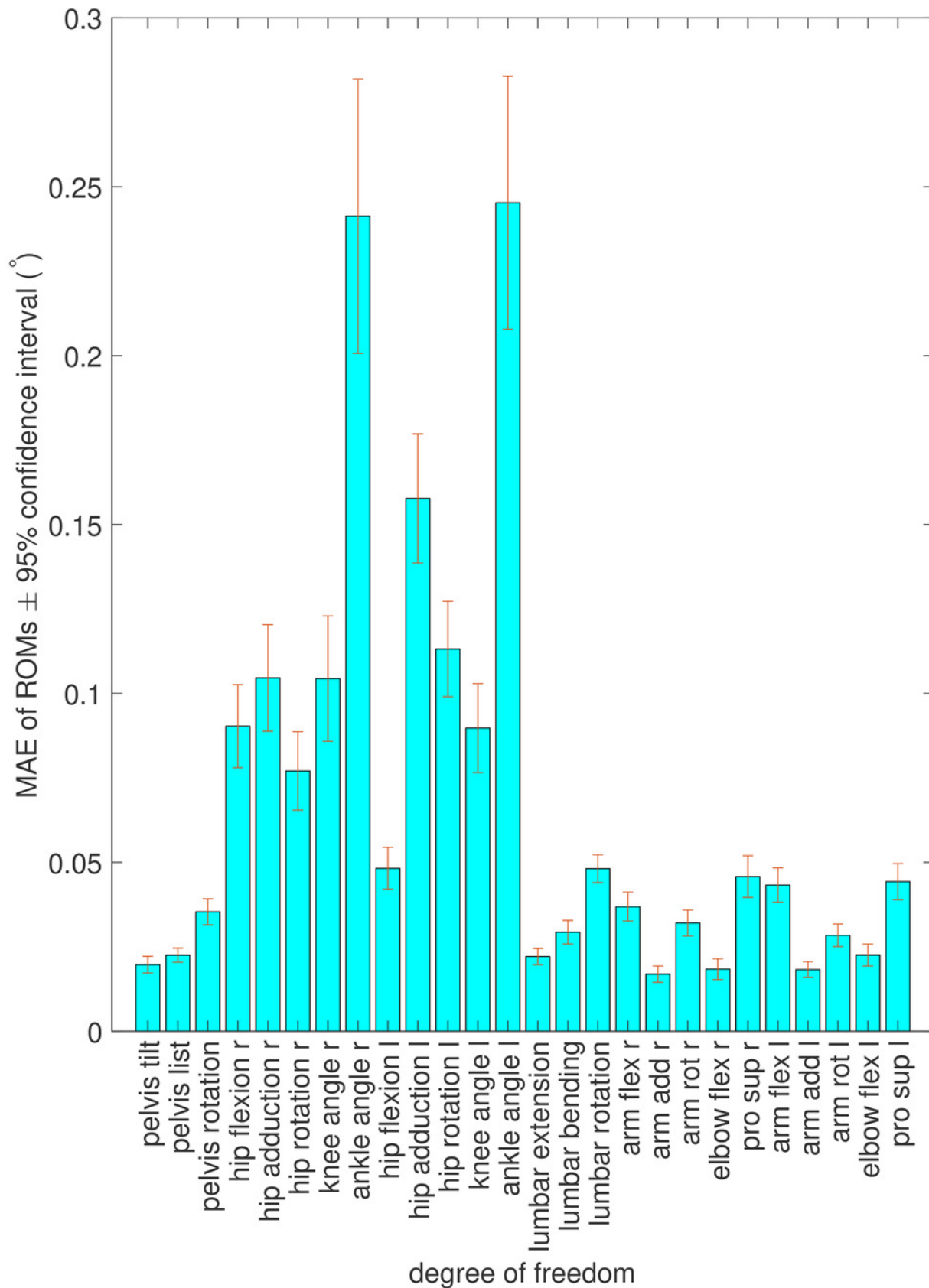
Inverse kinematics (IK) throughput with respect to the number of IK threads used, measured on a desktop computer (left) and a laptop computer (right).

The throughputs are presented as mean over ten trials. The measurements were repeated with two different musculoskeletal models, and using one, seven or 12 inertial measurement units (IMUs). IMU quaternion orientations were retrieved from previously recorded walking trials.

# Figure 4

Execution times and standard deviations (shaded area) of a single inverse kinematics operation with respect to the number of inertial measurement units (IMUs), i.e., number of segments with random orientation data.

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.

# Figure 5

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.

Conducted performance tests, the parameters that were varied in them, and their purposes.

IMU = inertial measurement unit; MS = musculoskeletal

1  Table 1: Conducted performance tests, the parameters that were varied in them, and their
2  purposes.

| Test | Varied parameters | Purpose |
|---|---|---|
| **Execution times with computer-generated random unit quaternions** | Computer, MS model, number of IMUs | Benchmark upper limit of execution times |
| **Execution times with gait data** | Computer, MS model, number of IMUs | Quantify execution times in normal use scenario |
| **Throughputs with gait data** | Computer, MS model, number of IK threads, number of IMUs | Quantify throughput in normal use scenario |

3  IMU = inertial measurement unit; MS = musculoskeletal

The MS model segments whose inverse kinematics were calculated during performance tests, presented by the number of IMUs whose data was utilized

IMU: inertial measurement unit; MS = musculoskeletal

1   Table 2: The MS model segments whose inverse kinematics were calculated during performance
2   tests, presented by the number of IMUs whose data was utilized.

| Number of IMUs | Segments described by IMU orientations |
|---|---|
| **1** | pelvis |
| **7** | pelvis, thighs, shanks, feet |
| **12** | pelvis, thighs, shanks, feet, torso, upper arms, lower arms |

3   IMU: inertial measurement unit; MS = musculoskeletal

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.

CI = confidence interval; IK = inverse kinematics; IMU = inertial measurement unit; STD = standard deviation

1  Table 3: Mean, standard deviation (STD) and 95% confidence interval (CI) of execution times of
2  a single inverse kinematics (IK) operation. The values are calculated over 10 000 IK operations
3  for two different musculoskeletal models, two different computers and 1, 7 or 12 inertial
4  measurement units (IMUs). Randomly selected unit quaternions were used as IMU orientations.

|  | Lower body, 1 IMU | | Lower body, 7 IMUs | | Full body, 1 IMU | | Full body, 7 IMUs | | Full body, 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 |

6  CI = confidence interval; IK = inverse kinematics; IMU = inertial measurement unit; STD =
7  standard deviation