

KinePose: A temporally optimized inverse kinematics technique for 6DOF human pose estimation with biomechanical constraints

Kevin Gildea¹, Clara Mercadal-Baudart¹, Richard Blythman^{1,2}, Aljosa Smolic², and Ciaran Simms¹

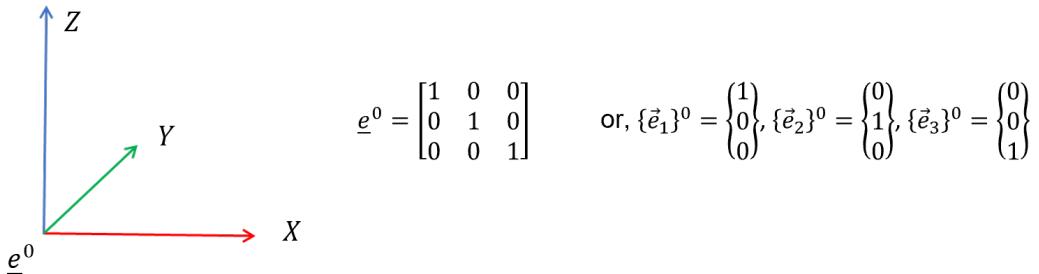
¹*School of Engineering, Trinity College Dublin*

²*V-SENSE, School of Computer Science & Statistics, Trinity College Dublin*

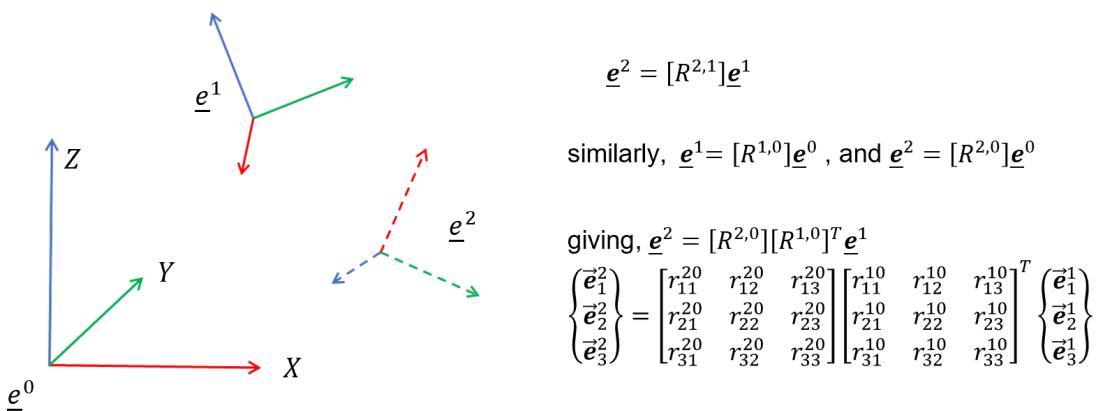
Supplementary Material

A The 3D rotation group and vector mapping

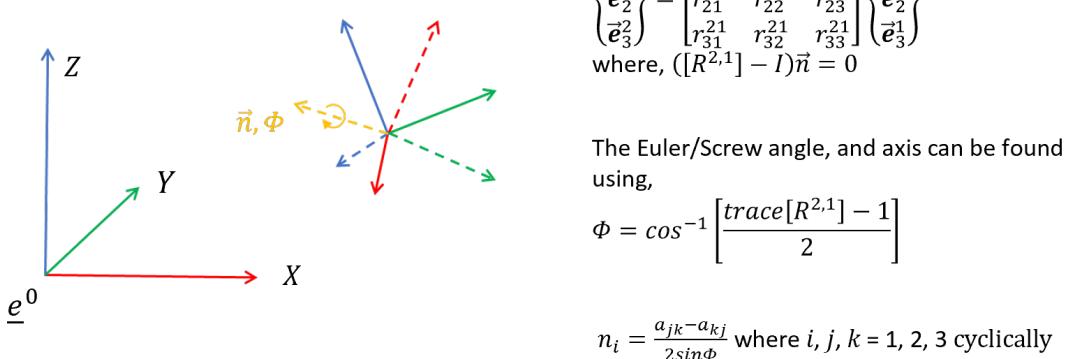
Basis vectors are used to define the directions of the cardinal axes of an object's local coordinate system. For example, object '0' is aligned with the global coordinate system, so the basis vector system can be expressed as a 3×3 identity matrix (I), with each of the basis vectors $\{\vec{e}_1\}^0$, $\{\vec{e}_2\}^0$, and $\{\vec{e}_3\}^0$ extending 1 unit from the origin along each of the cardinal axes:



Furthermore, one coordinate system defined using basis vector system \underline{e}^2 can be represented with respect to another coordinate system \underline{e}^1 by multiplying the latter by the 3×3 rotation matrix $[R^{2,1}]$. If we know the orientations of both of these coordinate systems with respect to a 3^{rd} coordinate system \underline{e}^0 , i.e. the global coordinate system, then we can find $[R^{2,1}]$:

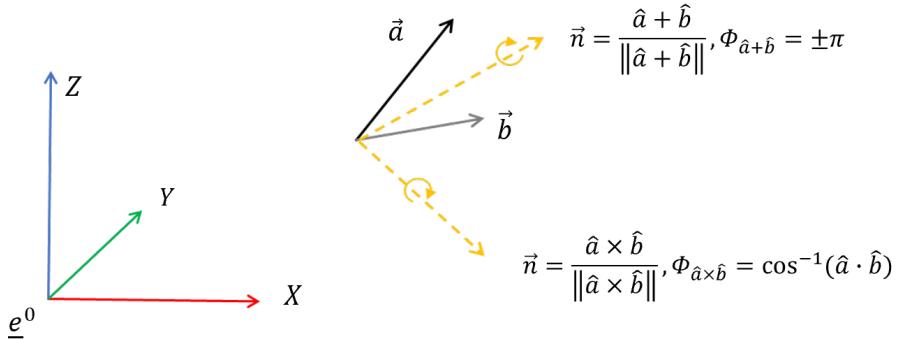


Euler's rotation theorem states that, in three-dimensional space, displacement from an initial orientation \underline{e}^1 to final orientation \underline{e}^2 is achieved by rotation through an angle about an axis with same coordinate matrix in both bases. This axis is called the Euler axis, or the Screw axis, \vec{n} , and an associated angle Φ :



Now, consider the case where instead of having two complete coordinate systems represented by basis vectors, we only have two vectors \vec{a} and \vec{b} , which we want to use to represent a rotation between the two coordinate systems. There will not be a unique solution to this problem, so we need to define the Screw axis about which we rotate. Two obvious candidate axes that can be chosen for this are:

- the cross product of the vectors, which will be mutually perpendicular to both \vec{a} and \vec{b} , or
- the sum of the normalized vectors \hat{a} and \hat{b} , which symmetrically bisects \vec{a} and \vec{b}



We can then relate the Euler axis-angle to the direction cosines of a rotation matrix using:

$$[R^{\hat{a} \rightarrow \hat{b}}] = \begin{bmatrix} n_1^2 + (n_2^2 + n_3^2)\cos(\Phi) & n_1 n_2 (1 - \cos(\Phi)) - n_3 \sin(\Phi) & n_1 n_3 (1 - \cos(\Phi)) + n_2 \sin(\Phi) \\ n_1 n_2 (1 - \cos(\Phi)) + n_3 \sin(\Phi) & n_2^2 + (n_1^2 + n_3^2)\cos(\Phi) & n_2 n_3 (1 - \cos(\Phi)) - n_1 \sin(\Phi) \\ n_1 n_3 (1 - \cos(\Phi)) - n_2 \sin(\Phi) & n_2 n_3 (1 - \cos(\Phi)) + n_1 \sin(\Phi) & n_3^2 + (n_1^2 + n_2^2)\cos(\Phi) \end{bmatrix}$$

We can use the two points obtained from these approaches, which we know are on the Euler axis solution space, along with the origin to define the plane that contains all possible rotation axes:

$$p_1 = \{0 \ 0 \ 0\}, p_2 = \frac{\hat{a} + \hat{b}}{\|\hat{a} + \hat{b}\|}, p_3 = \frac{\hat{a} \times \hat{b}}{\|\hat{a} \times \hat{b}\|}.$$

We can then define the Euler axis solution space as all unit vectors on the plane, for $\alpha : [-\pi, \pi]$:

$$\vec{n}_\alpha = \begin{cases} p_{1,1} + \cos(\alpha)(p_{2,1} - p_{1,1}) + \sin(\alpha)(p_{3,1} - p_{1,1}) \\ p_{1,2} + \cos(\alpha)(p_{2,2} - p_{1,2}) + \sin(\alpha)(p_{3,2} - p_{1,2}) \\ p_{1,3} + \cos(\alpha)(p_{2,3} - p_{1,3}) + \sin(\alpha)(p_{3,3} - p_{1,3}) \end{cases}.$$

Projecting vectors \hat{a} and \hat{b} to lay on the base of a cone with the Euler axis as the cone's axis, we can determine the associated Euler/screw angles (Φ), i.e., the azimuth angle:

$$\hat{a}_\alpha = \frac{\hat{a} \times \vec{n}_\alpha}{\|\hat{a} \times \vec{n}_\alpha\|}, \hat{b}_\alpha = \frac{\hat{b} \times \vec{n}_\alpha}{\|\hat{b} \times \vec{n}_\alpha\|},$$

$$\Phi_\alpha = \cos^{-1}(\hat{a}_\alpha \cdot \hat{b}_\alpha).$$

Now, we can formulate all possible rotation matrices representing all possible Euler axis-angle combinations, i.e. the Euler axis-angle solution space (EAS) (see Figure 1):

$$\left[R^{\hat{a} \rightarrow \hat{b}} \right]^{EAS} = \begin{bmatrix} n_{\alpha,1}^2 + (n_{\alpha,2}^2 + n_{\alpha,3}^2) \cos(\Phi_\alpha) & n_{\alpha,1}n_{\alpha,2}(1 - \cos(\Phi_\alpha)) - n_{\alpha,3}\sin(\Phi_\alpha) & n_{\alpha,1}n_{\alpha,3}(1 - \cos(\Phi_\alpha)) + n_{\alpha,2}\sin(\Phi_\alpha) \\ n_{\alpha,1}n_{\alpha,2}(1 - \cos(\Phi_\alpha)) + n_{\alpha,3}\sin(\Phi_\alpha) & n_{\alpha,2}^2 + (n_{\alpha,1}^2 + n_{\alpha,3}^2)\cos(\Phi_\alpha) & n_{\alpha,2}n_{\alpha,3}(1 - \cos(\Phi_\alpha)) - n_{\alpha,1}\sin(\Phi_\alpha) \\ n_{\alpha,1}n_{\alpha,3}(1 - \cos(\Phi_\alpha)) - n_{\alpha,2}\sin(\Phi_\alpha) & n_{\alpha,2}n_{\alpha,3}(1 - \cos(\Phi_\alpha)) + n_{\alpha,1}\sin(\Phi_\alpha) & n_{\alpha,3}^2 + (n_{\alpha,1}^2 + n_{\alpha,2}^2)\cos(\Phi_\alpha) \end{bmatrix}$$

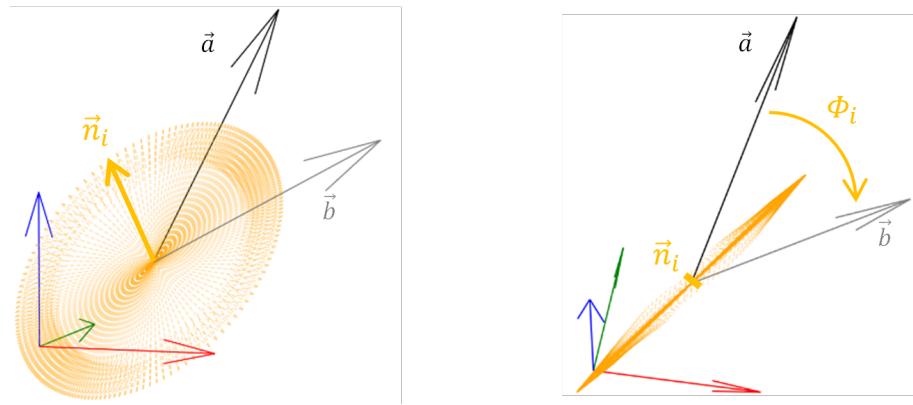


Figure 1: The Euler axis-angle solution space (EAS) for mapping/retargeting incongruent vectors.

Therefore, with a lack of complete basis vector representation for the two coordinate systems, there are infinite axis-angle combinations that can be applied to achieve a desired vector mapping/retargeting. However, candidate Euler/Screw axes are constrained to be on the plane that symmetrically bisects the vectors (Figure 1).

B Gimbal lock

For Cardan rotation order $X(\theta) \rightarrow Y(\phi) \rightarrow Z(\psi)$, we have a rotation matrix $[R^{2,1}]$:

$$[R^{2,1}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix} \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

When $\phi = \frac{\pi}{2} + n\pi$:

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then, by trigonometry:

$$= \begin{bmatrix} 0 & 0 & 1 \\ \cos(\theta + \psi) & \cos(\theta + \psi) & 0 \\ -\cos(\theta + \psi) & \sin(\theta + \psi) & 0 \end{bmatrix}$$

Relating our coefficients to direction cosines, i.e., $r_{i,j}^{2,1} = \vec{e}_i^2 \cdot \vec{e}_j^1$, we see that the direction of the x axis of the coordinate system post-rotation will always align with the z axis pre-rotation, i.e., since $\angle(\vec{e}_i^2, \vec{e}_j^1) = \cos^{-1}(\vec{e}_i^2 \cdot \vec{e}_j^1)$, then $\angle(\vec{e}_1^2, \vec{e}_3^1) = 0$, and $\angle(\vec{e}_1^2, \vec{e}_1^1), \angle(\vec{e}_1^2, \vec{e}_2^1), \angle(\vec{e}_2^2, \vec{e}_3^1), \angle(\vec{e}_3^2, \vec{e}_3^1) = \frac{\pi}{2}$.

C Boundary cases for twist angle ambiguity

Here we investigate how the IK algorithms handle twist angle ambiguity in boundary cases with extended/straight limbs (co-linear sequential links). This is demonstrated for two 15-frame motion sequences involving a phase of bent limbs, and fully extended limbs:

Case 1: Bent limbs (frames 1-11) → Extended limbs (frames 13-15)

Case 2: Extended limbs (frames 1-3) → Bent limbs (frames 4-15)

Case 1 $MPJAS_1$ errors for limb joints in the motion sequence are shown in Figure 2. Note that 1_a results in the highest errors when limbs are extended, and improvements are achieved by 1_b , 2_3 , and 2_5 .

Case 2 results are shown in Figure 3. 1_a , 1_b , and 2_3 result in high errors when limbs are extended, and improvements are only achieved by 2_5 . 1_b fails in this case because there are no preceding frames with bent limbs to use as initialization. The accuracy of 2_3 is generally lower than 2_5 in this case because its initial batch does not include frames with bent limbs, with lower peak errors throughout. However, there are cases where 2_5 under-performs 2_3 .

See [this video](#) for visualisation.

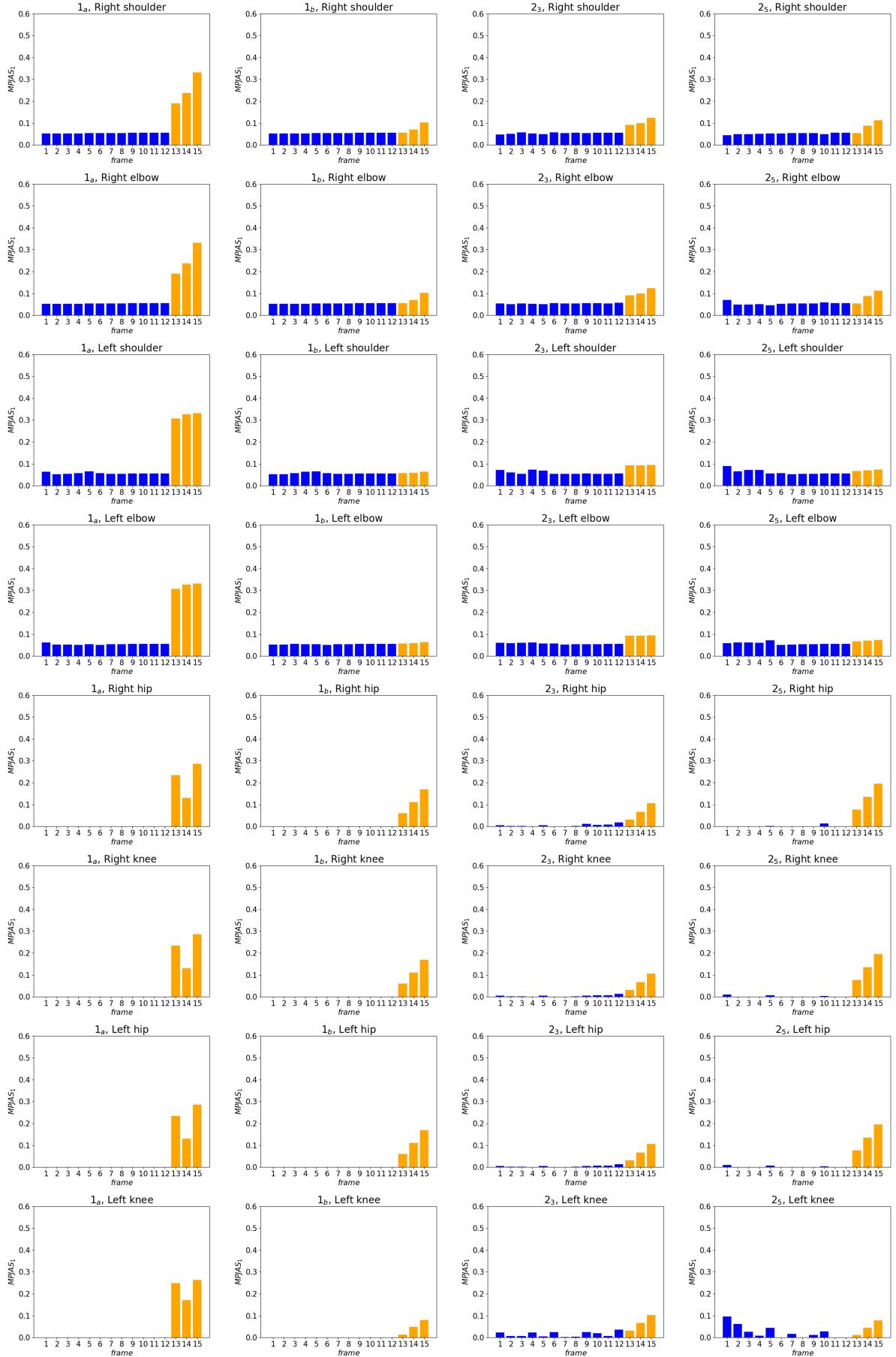


Figure 2: MPJAS₁ errors for limb joints in a 15-frame motion sequence involving an initial phase of bent limbs (blue), followed by a phase extended limbs (orange) (mean values over 10 IK runs).

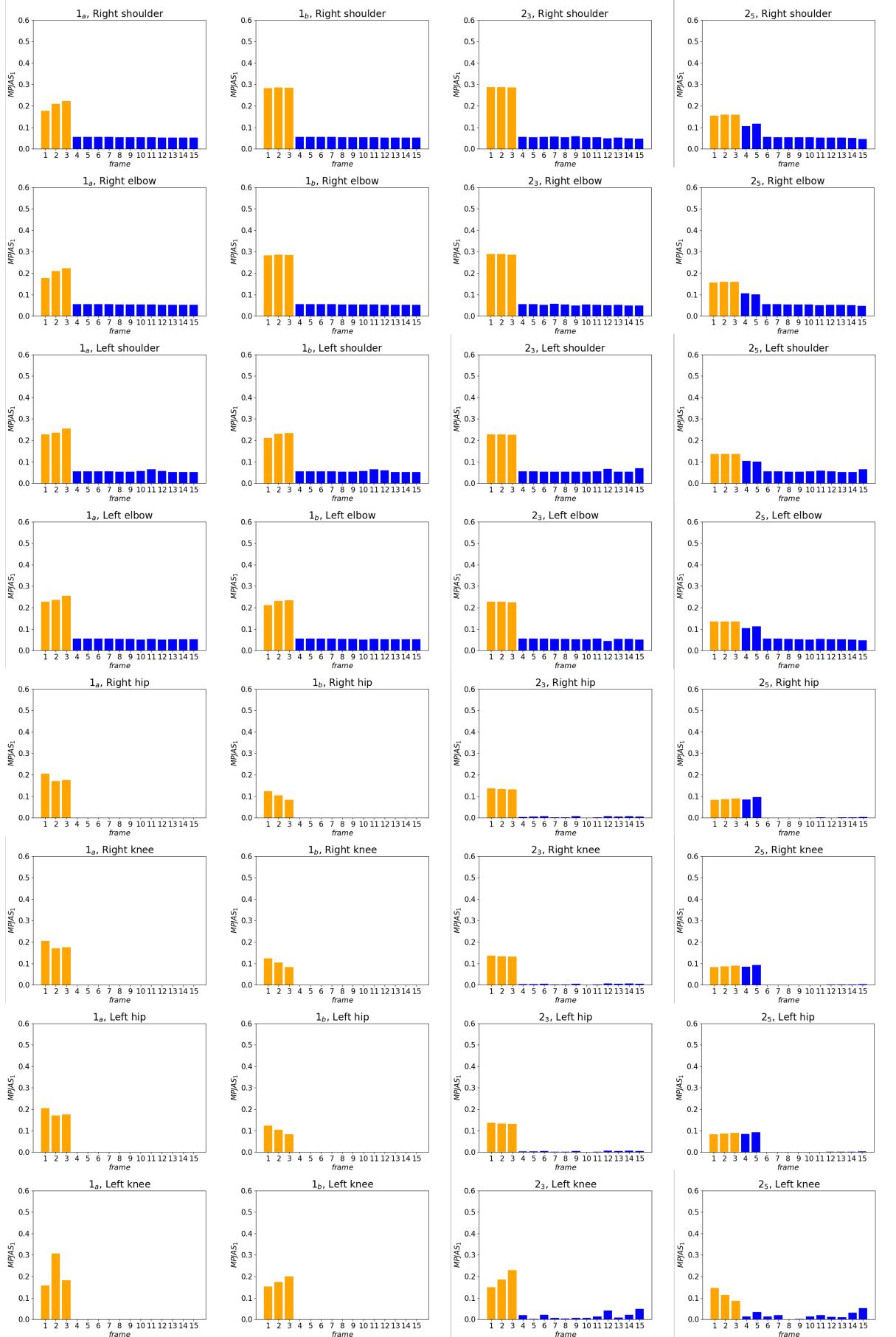


Figure 3: $MPJAS_1$ errors for limb joints in a 15-frame motion sequence involving an initial phase of extended limbs (orange), followed by a phase bent limbs (blue) (mean values over 10 IK runs).

D Accuracy for motion sequences with bent limbs only

Figure 4 shows a comparison of average average angular errors for parameterized joints in the chain for motion sequences involving motion with bent limbs throughout. In comparison to the results for motion sequences involving phases of both bent and extended limbs (Figure 6 in the main paper) the results are similar across all algorithms, demonstrating that twist angle ambiguity is not as problematic for this kind of motion.

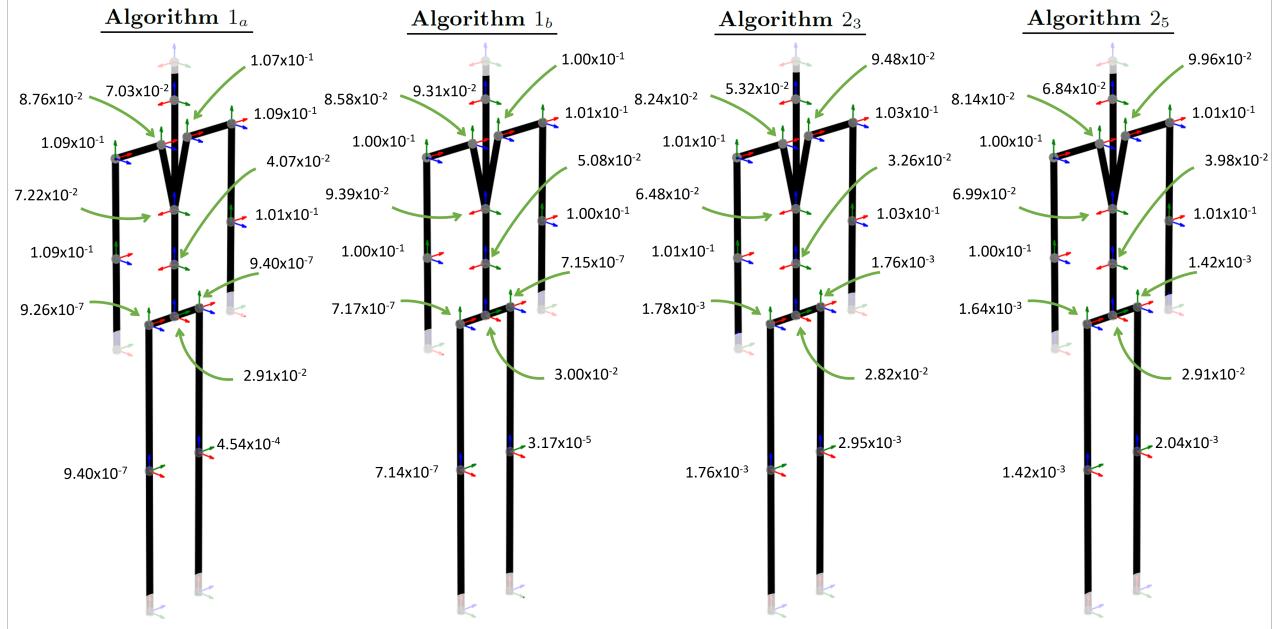


Figure 4: MPJAS₁ by joint and algorithm, for motion sequences with bent limbs only.

E Convergence to local minimum

Convergence of local optimization methods depends on the initial guess. Since IK algorithms feed optimized parameters from previous frames, having an initial guess closer to the target pose may be beneficial. Therefore, in practical applications, the subject may initially adopt a pose similar to the rest pose. Alternatively, the initial guess (i.e., Θ_0 in Algorithm 1) for model parameters could be defined closer to the target pose. For this, we include a tool for manual setting of the initial guess (Figure 5).

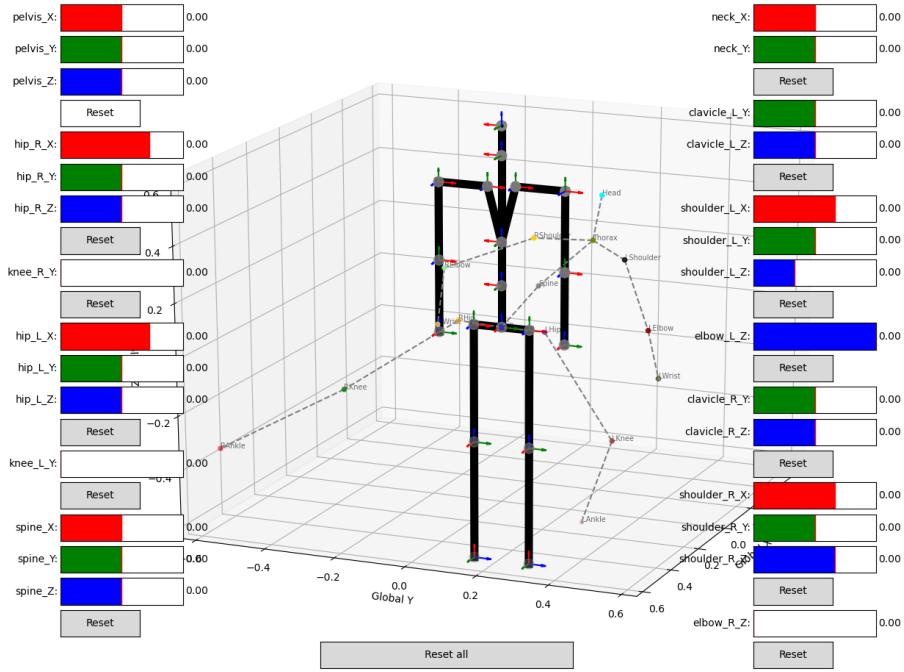


Figure 5: Optional tool for manual setting of the initial guess for model parameters in frame 1.