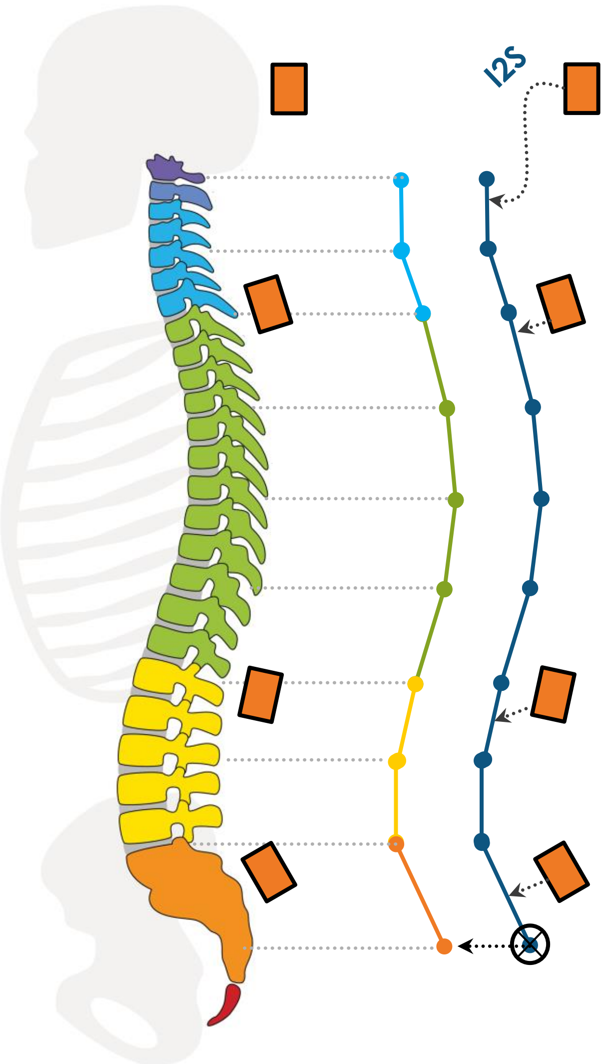
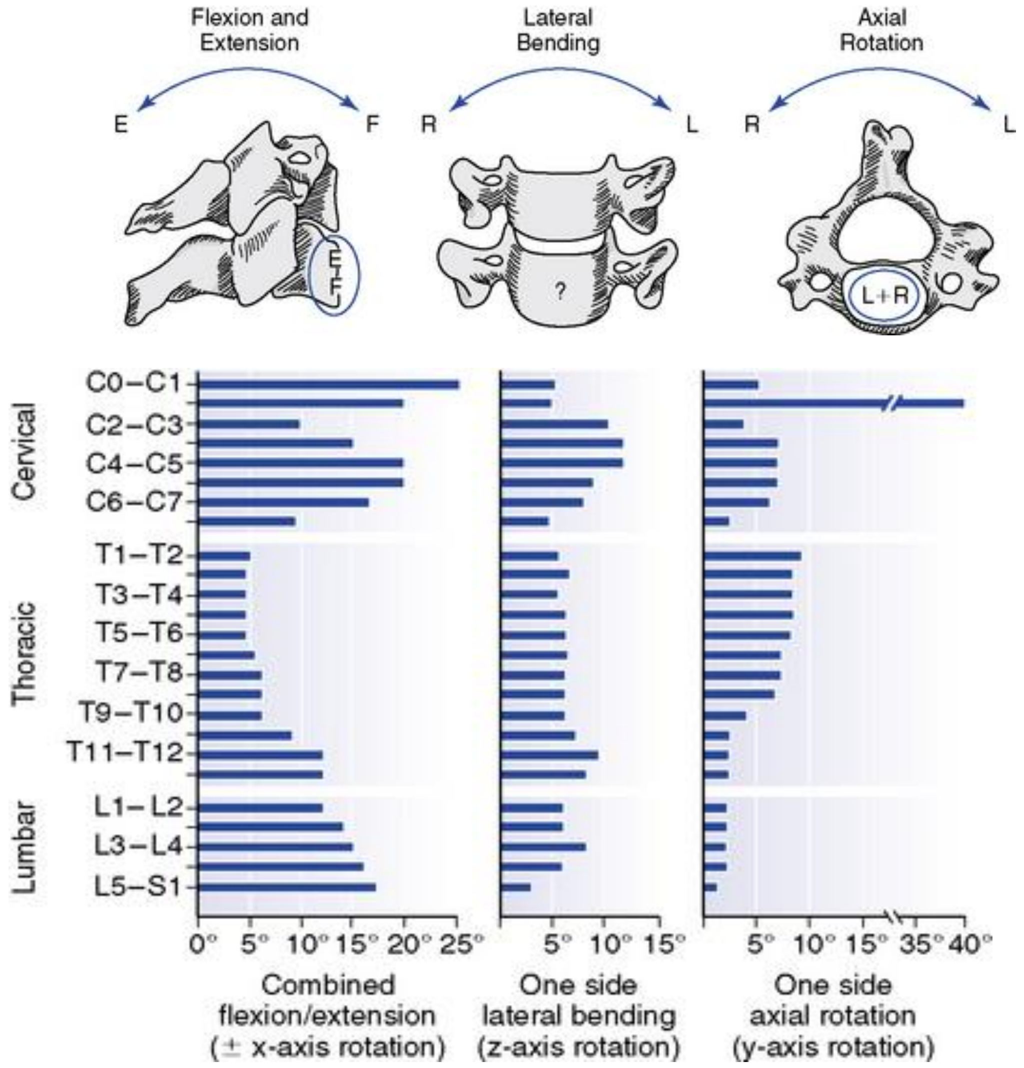
Improved Spine Modeling and Tracking



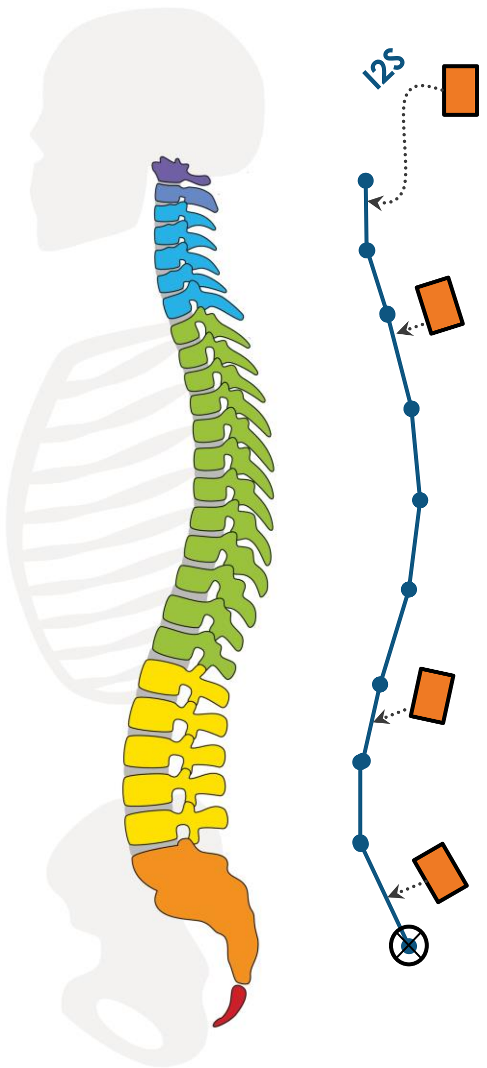
3 sensor / n segments model:

- sensors at **turning points** / spinal transition zones
- **chain** of n vertebral segments with **lenght function** $l(n)$, $l\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$
- **(1) interpolate** rotation and position between sensors
- **(2) decompose & distribute** total **spine rotation** among joints according to a RoM distribution $RoM(n)$, $RoM\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$



"*Clinical biomechanics of the spine*", White, A. A., & Panjabi, M. M. 1990

Spine Modeling and Tracking – cont‘d (1)



(1) inter-sensor interpolation

intuitive: parametric 3d C^2 splines on **position data**

$$s(\tau(n)),\ \ s\colon \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}^3,\ \ \tau(n) = \frac{\sum_0^n l(n)}{\sum l(n)} \in [0,1]$$

- positions from inertial sensors ☹
+ C^2 between support positions ☺
- yields position-induced segment rotations ☹

interpolation based on **orientation data**:

$$slerp(q_0, q_1, \tau),\ \ slerp\colon \mathbb{R}_{\geq 0} \longrightarrow \mathrm{SO}(3),\ \ \tau \in [0,1]$$

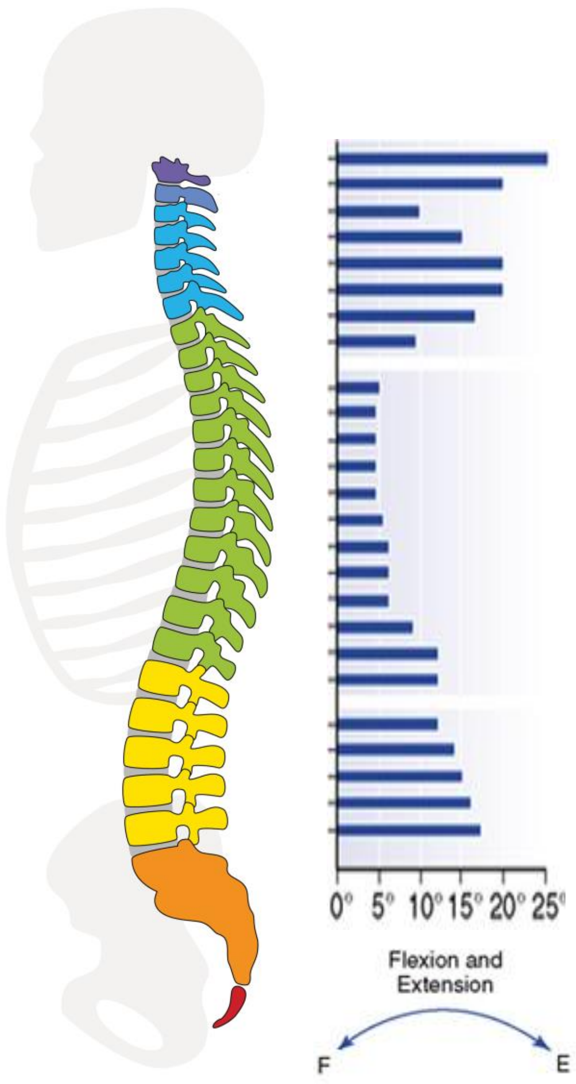
+ orientations from inertial sensors ☺
- def. locally between two rotations, no “spline behavior“ ☹
~ yields length-induced positions (?)

SLERP-splines: use $slerp$ to construct **bezier splines** on the **surface** of the **4d unit sphere** where quaternions live
+ C^2 between support orientations ☺

combine both approaches:

position splines + orientation splines

(!) solution is non-unique ➔ optimize within tracking framework!
(joint connections constraints)



(2) rotation distribution over vertebral segments

idea: “weight the interpolated rotations found with **(1)** according to the RoM distribution of a *normal, healthy* spine“

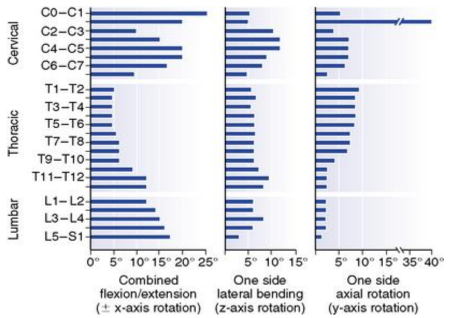
➔ natural curvature (?)

technically:

first: decompose 3d rotation into its 1d components

(➔ isolate twist / axial rotation, otw. inducing swing, using e.g. swing-twist quaternion decomposition)

➔ address individual RoM table per axis



second: remap the interpolation parameter

➔ “weighted“ interpolation along the curve length

➔ requires the RoM distribution to be a (discrete) **pdf**

➔ we define $\tau^*(n) = \sum_{k=0}^{n} RoM(k)$ via the **cdf**

and use τ^* as the new spline parameter

finally:

optimize interpolated vertebral positions and orientations as (virtual) tracking states

➔ joint connection constraints + intervertebral (physical?) RoM constraints + ???

Spine Modeling and Tracking – Visualization