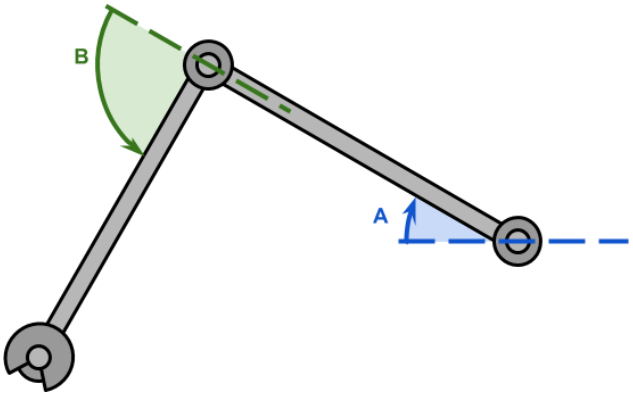
# Kinematic model specification




| | |
|---|---|
| Version: | 1.0 |
| Date: | 27 Jan 2022 |
| Author(s): | Fred Dijkstra |
| Document id: | mrx9mktm |

# Document history

| Version | Date | Author(s) | Description |
|---|---|---|---|
| 1.0 | 27 Jan 2022 | Fred Dijkstra | • First full version. |

# Table of contents

# Introduction

To be able to visualize the motion capture of a user performing an exercise or map the orientations of the relevant segments from an animated 3D model, a simplified kinematic model must be created in the scene.

Such a kinematic model comprises all the segments that could be tracked in a practical situation and the associated joints.

In this document the hierarchy of the model is described as it appears in the scene. Based on this hierarchy, it is described how measured orientations of the segments can be used to animate the kinematic model.

In situations where not all segments are measured, only a subset of the kinematic model is required. As will be described, this is only possible when the subset is a single *kinematic chain*.

It is also described how using the hierarchical structure of the model makes it possible to use positions of the joints as well when the dimensions of the segments are considered.

# Hierarchy

The kinematic model is constructed by a number of *body parts* which are *joints* and *segments*. These are added as *nodes* to the scene. In figure 1 the naming of the body parts is given.

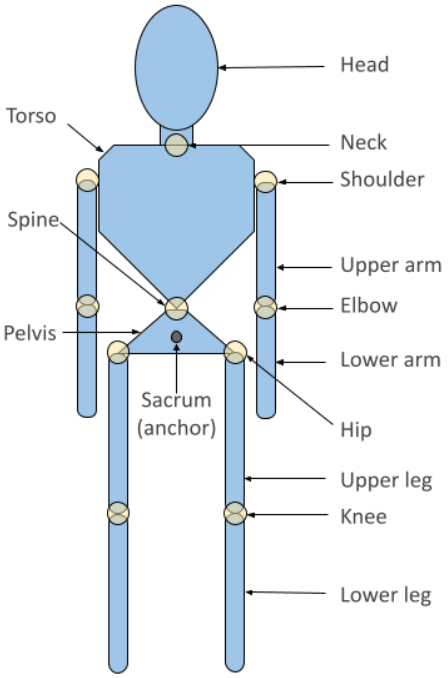



***Figure 1: Kinematic model comprising joints and segments.***

*The kinematic model is a simplified representation of the human body and all the joints are modeled as simple ball-and-sockets. For example, the shoulders are extremely simplified as there is neither a shoulder bone (scapula) nor a collar bone (clavicle). For assessing the movement, this type of model suffices in almost all practical situations. Note that measuring the shoulder accurately is almost impossible anyway and as such useless to make this part of the assessment.*

The indicated kinematic model is constructed in a node hierarchy as given in figure 2.

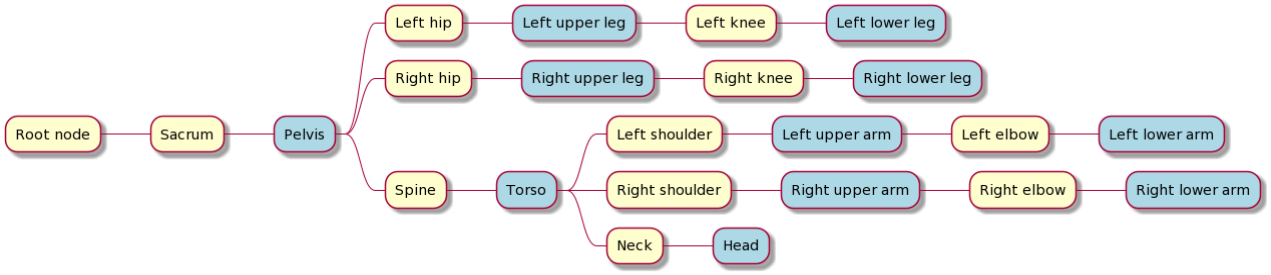



***Figure 2: Node hierarchy (light blue are segments).***

Note that the sacrum is not really a joint and is a node without a mesh. For the complete model, the center-point of the pelvis coincides with this point, in other words, the relative position of the pelvis is (0,0,0). The pose of the sacrum is explicitly added by the mocap recording process and as such part of the kinematic data.

# Orientations

Nodes in a scene will rotate around their *pivot* point[2] as it is called in the SceneKit SDK. The pivot point is expressed in the coordinate system of the node and by default it is located at $(0,0,0)$. However, this is not correct as illustrated in figure 3 where an example is given for the upper and lower arm.

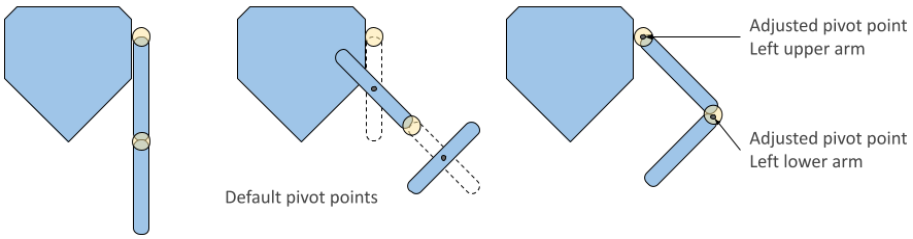



***Figure 3: Joints are implicitly implemented as pivot points.***

This means that we cannot simply place the measured orientations on the corresponding segments.

Because of the hierarchy as was shown in figure 2, each segment is connected to a joint. This means that the joint serves as the root-node of the segment, i.e. the segment is the child-node of the joint.

These subsegments are illustrated in figure 4.

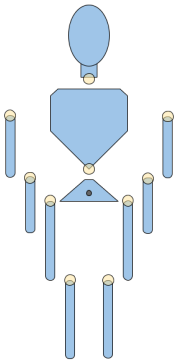


***Figure 4: Subsegments where each segment is a child-node of the joint.***

Note that - except for the sacrum - each joint will also be the child of a segment higher-up in the hierarchy.

This means that after the kinematic model is constructed in the scene as described in the previous section, the orientation of a segment is to be changed *implicitly* by changing the orientation of the joint to which it is attached.

Note that this means that not only the attached segment will rotate but also all the contained joints and segments. This means that all orientations must be expressed in *world coordinates*, i.e. in the coordinate system of the root-node of the scene.

# Subsets

For practical reasons, not all recordings might necessarily contain all the segments. As an example, consider a situation as illustrated in figure 5 in which only the right arm is captured. In this example, the ‘anchor’ is placed at the right shoulder joint and only the upper- and lower arm are displayed.

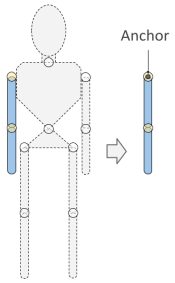



***Figure 5: Only a subset of the body was captured.***

Now consider a situation in which segments are captured that are not connected by intermediate segments. Two examples are illustrated in figure 6. In these situations, all the other segments could be deleted from the scene, but the result would be more than one kinematic chain. Because there can only be one anchor, the recording process must ensure that the chain is connected as illustrated in figure 6, where the anchor coincides with the joint highest in the hierarchy.

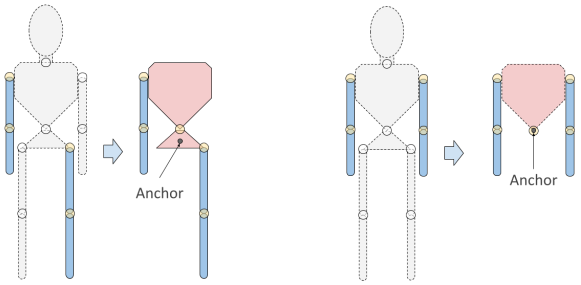



***Figure 6: Connecting the kinematic chain.***

For a recording read from a file it can be assumed that there is one kinematic chain. However when creating a recording it must be explicitly prevented that unconnected chains can be stored when segments are selected.

# Positions

With the node hierarchy properly set, the *position* of a joint will change when the orientation of a parent segment changes. Likewise, when the position of the parent segment changes, the joint is translated with the same vector.

It is therefore that if the position of the child segment is supplied as part of the kinematic data, it will not be used. Only the position of the *anchor* will be used.

In most practical recordings the position of the anchor will not change as this does not give a lot of useful information for assessing the exercise.

While orientations are ‘dimensionless’, positions depend on the scaling. Therefore differences between the scaling of the recording and the visualization need to be considered. For exercise recordings, it can be assumed that the scaling is the same because the kinematic model used during recording will be the same as used for visualisation.

# References

[1] *Echtfit prototyping tool specification*
Fred Dijkstra
Versie: 1v1, November 26, 2020
Document id: y4lx88wv

[2] The pivot point for the node's position, rotation, and scale.
https://developer.apple.com/documentation/scenekit/scnnode/1408044-pivot