User Manual

Table of contents

1. Introduction

The goal of KineXYZ is to offer a 3D development platform that will enable developers to build new revolutionary interactive mobile apps and games for Augmented- and Virtual Reality (A/VR), interactive sports and any other field a developer can think of.
To read more about the benefits of KineXYZ for diverse user groups such as for example Biomechanical research engineers, game developers and technical students, refer to the document "KineXYZ functional product specification".
This document describes all the functionalities of KineXYZ and how to use this development platform. Now you can learn how to animate your world.

Tip

Next to this manual, there are also application notes that give a deeper insight into the possibilities of the tool. Please follow this link to the KineXYZ Application Notes.

AppStore link: https://apps.apple.com/app/kinexyz/id1516231657.

2 Kinematic modeller

When the app is started, the screen after the welcome screen shows the kinematic modeller.

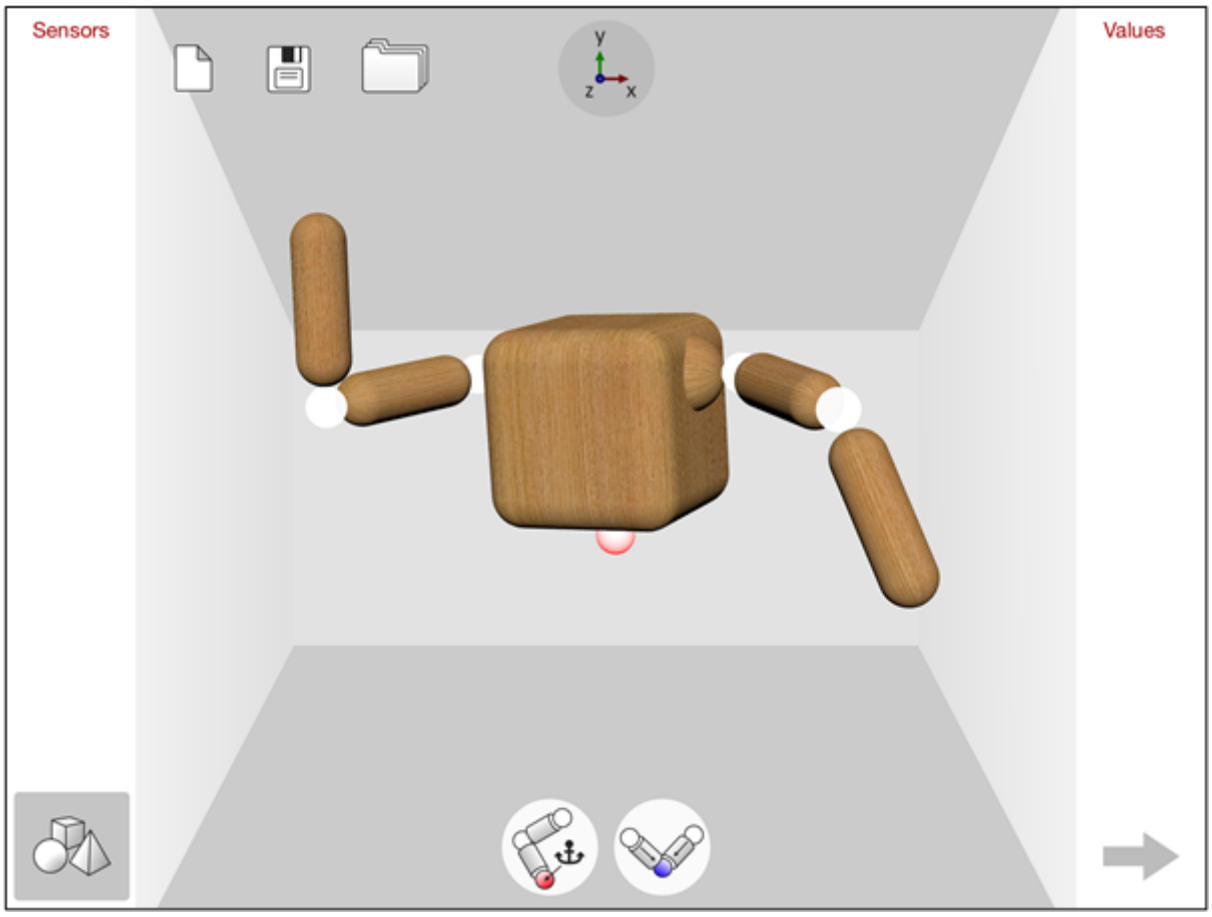


Figure 1: Kinematic modeller

The kinematic modeller is designed for three purposes:

- Build kinematic chains out of virtual objects.
- Associate and align virtual objects to physical sensors.
- Display output values of the dataflow diagram.

The last point is made possible because for each object (or segment) in a kinematic chain, values can be selected and used for further processing in the dataflow diagram view as will be discussed in more detail in "Dataflow diagram".

***Note**: the coordinate system is* right-handed *because when the x-axis is rotated to the y-axis, the result is the positive z-axis (see "Coordinate system").*

2.1 Viewport

The kinematic modeller shows a scene, a 3D virtual environment in which everything is represented. This scene is perceived by means of a *viewport*. This viewport can be controlled. This way the scene can be observed from different perspectives. In total there are three viewport controls; zoom, move and rotate. How to use these controls, is explained next.

Zoom

Zooming is done by a two fingers pinching movement. Bring together to zoom out, move apart to zoom in.

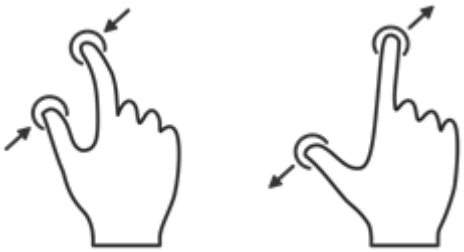


*Figure 2: **Pinching movements to zoom out (left) and zoom in (right).***

Rotate

Rotating the camera is done by a one finger swiping movement. This way the camera can be moved up, down, left or right. *Note that a diagonal rotation is not possible.*

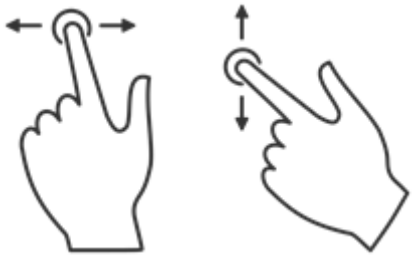


*Figure 3: **Swiping movements to rotate left/right (left) and rotate up/down (right).***

2.2 Coordinate system

A very important aspect of any 3D software (modeling or animation) is the definition and direction of the coordinate system, more specifically the direction of the axis and the handedness: there are left hand and right hand coordinate systems as illustrated in the figure 4.

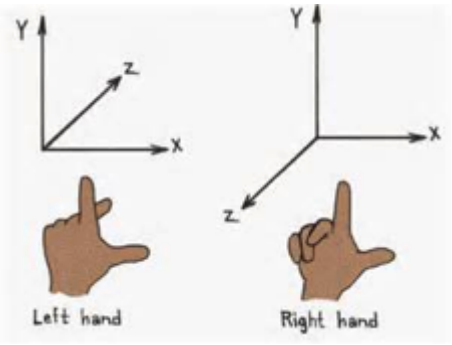


Figure 4: Handedness of a coordinate system..

Apart from the handedness, the direction of the y-axis is also important. In most cases y is up, but there are also systems that have z-up orientation.

The handedness of the coordinate system also defines the direction of a positive rotation. This is illustrated in the figures below:

	
In the right-handed coordinate system the direction in which your hand closes to make a fist is the direction of a positive rotation around any axis represented by the extended right-hand thumb.	In the left-handed coordinate system the direction in which your hand closes to make a fist is the direction of a positive rotation around any axis represented by the extended left-hand thumb.

Unfortunately, not everybody uses the same handedness for the coordinate system, although since OpenGL uses a right-handed coordinate system, there seems to be a preference for right-handed. On the other hand, Unity3D uses a left-handed coordinate system.

The following lists the coordinate system of some of the major 3D animation tools/formats.

Tool	Handedness	Direction
Unity 3D	Left	y-up
Cinema 4D	Left	y-up
Direct3D	Left	y-up
Unreal	Left	z-up
3D Studio Max	Left	z-up
LightWave 3D	Left	y-up
COLLADA	Left	z-up
OpenGL	Right	y-up
Unigine	Right	z-up
SceneKit	Right	y-up
Blender	Right	z-up
Motionbuilder	Right	y-up

Cheetah3D	Right	y-up

2.3 Sensors

To manipulate the virtual objects in the scene, Bluetooth enabled Inertial Measurement Units (IMUs) can be connected to the iPad and assigned to specific objects.
KineXYZ uses the Xsens DOT. At this point it is assumed that the user is familiar with the operation of the Xsens DOT and the basics of Bluetooth operation.

Detect sensors

To enable the iPad to detect sensors, the “Sensors” column on the left of the screen must be dragged towards the middle of the screen. This will start the detection, or *scanning* process.

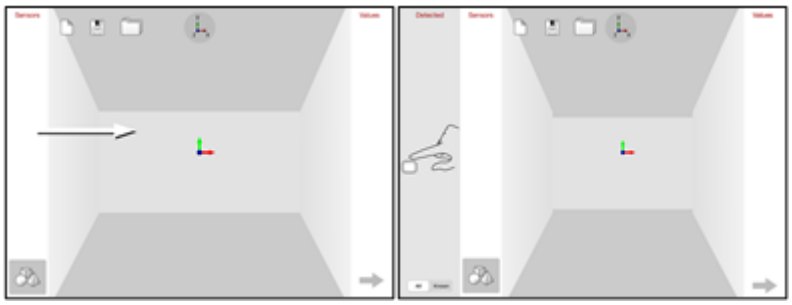


Figure 5: Dragging the “Sensors” column to the left, enables the detection and connection of sensors.

Once scanning is enabled and the sensors are switched-on, the sensors will connect and appear first in the “Detected” column. At this point the sensors will be configured to start measuring and when the first measurement is received for the sensor, it will move to the “Sensors” column.
***Note**: if sensors don’t move to the “Sensors” column or are switched-on, but don’t properly connect, power-cycle them and try again. In some cases a reset of Bluetooth on the iPad might be necessary.*
By rotating the sensor around its vertical axis, a sensor can now be identified in the interface. As soon as all required sensors are properly connected, scanning can (and preferably must) be stopped by sliding the “Sensors” column back to the left (see figure 6).

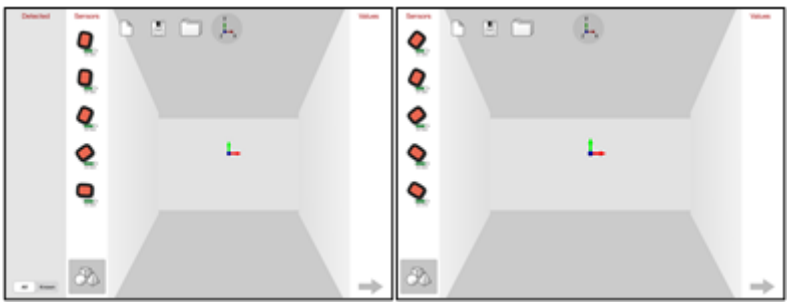


Figure 6: When all sensors are connected, slide the column back to the left to stop scanning.

When using more sensors, one or more sensors can show a small delay. This has to do with the internal BLE scheduling on iOS. During scanning, the app will automatically perform a reschedule of the lagging sensors. This will cause the affected sensors to disconnect and

reconnect. This process will continue until all sensors are properly connected. One can easily determine when the process finished by rotating the sensors simultaneously.

Check sensor heading

To be able to use the sensors in a combined kinematic model, all sensors must have the same reference as to the direction where the *heading* is 0. To check this, place the physical sensors in a row on a table, all facing in the same direction and check whether the sensors on the screen all have the same heading. If there are differences in the heading of the sensors on the screen, while the physical sensors are still perfectly aligned, something is wrong.
To solve this, the Xsens DOT offers the possibility to reset the horizontal orientation, i.e. the *heading*. This functionality must be used to realign sensors because when sensors are switched-on and are connected to the app, their heading can be different.
The process to do this is illustrated in figure 7.

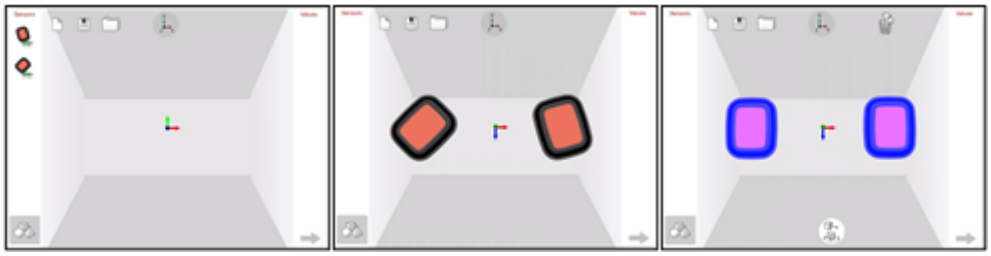


Figure 7: Selecting sensor objects and pressing the reset orientation button.

The steps illustrated in figure 7 are as follows.

1. Place the connected sensors on a flat surface (with no metal) and make sure that the sensors are aligned, i.e. pointing in the same direction. One can use the charging case for example to ensure the alignment.
2. Drag the sensors into the 3D view.
3. To be able to monitor the result, rotate the viewport by swiping down.

4. Select all sensors in the sece. This will make the heading reset button (see figure 8) appear.
5. Press the heading reset button.

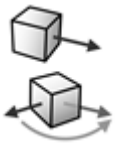


Figure 8: Heading reset button.

As a result, the sensors will now all point to the same heading.

Disconnecting sensors

In some cases you might want to disconnect sensors, e.g. because you don't need all of them or other sensors nearby were advertising and should not have connected.
To disconnect a sensor, long-press the small sensor in the "Sensors" column. A pop-up will appear asking whether you want to disconnect the sensor.
When "Yes" is clicked, the sensor will disconnect immediately.
When this is done during scanning, the sensor will obviously immediately reconnect. To prevent this, the slider at the bottom of the "Detected" column must be shifted to "Known". When this is done, a disconnected sensor will not reconnect.
It is also possible to create something like a whitelist. When you have connected the sensors for the first time, you can slide the slider from "All" to "Known". When you then come into a room with other sensors, only the previously connected sensors will connect.

2.4 Objects

In KineXYZ, objects are basic 3D geometries with no substance (they are partially transparent) and have no sensor associated with it. Like sensors, objects can be dragged into the scene. How to do this, as well as changing the properties of objects will be discussed in this paragraph.

Add objects to scene

To add an object to the scene, press the objects-button (figure 9).

Figure 9: Objects button.

Next, all available objects are shown. These can be dragged and dropped to the middle, adding a partially transparent geometry to the scene (see figure 10).

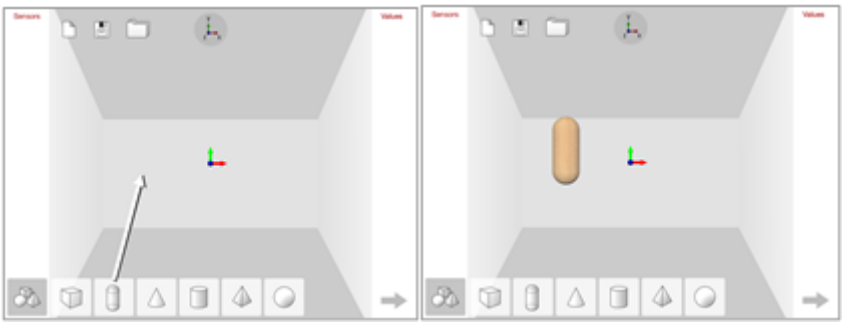


Figure 10: Objects-button pressed (left) and dragged into scene.

Properties

Objects can be customized by changing their properties. To do this, first select an object. Then press the properties-button (figure 11).

Figure 11: Properties button.

Next, the properties-window appears showing all variables that can be altered; the object's name, its position (x,y,z) and the object's dimensions (see figure 12).

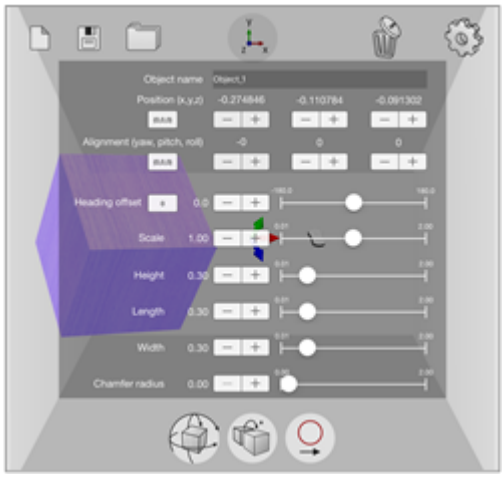


figure 12: Properties-window for a cube-object.

***Note**: giving an object a name can be especially handy when using the object's values in a later stage of the process. Hereby, values can easily be identified as belonging to an object.*

2.5 Models

In KineXYZ, an object can be turned into a controlled model by associating a sensor to an object. Subsequently, the model will adopt the orientation of the physical sensor.

To do this, choose the sensor and drag it onto the object. When the sensor is dragged over the object, the object will be selected. Now drop the sensor and the object will become opaque, indicating that the orientation of the object is now controlled by a sensor. The steps are illustrated in figure 13.

Figure 13: Dragging a sensor over an object will turn the object into a model.

An alternative method to do this is to drag a sensor into the scene and select both the sensor and the object. Now the insert-sensor-button (figure 14) appears.

Figure 14: Insert button.

When the insert button is pressed, the selected sensor is assigned to the selected object as

seen in figure 15.

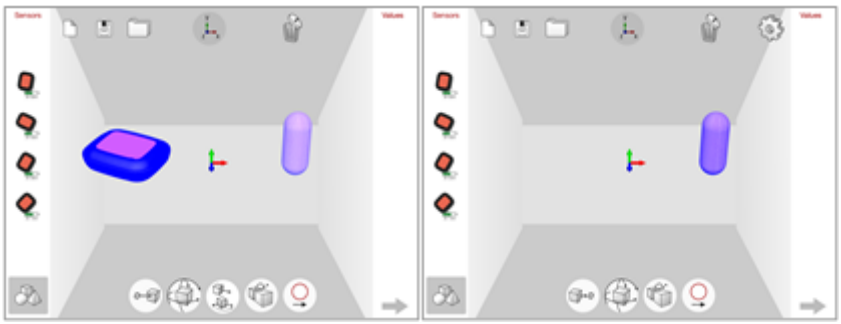


Figure 15: Pressing the insert button (leftmost button) assigns the selected sensor to the selected segment.

To detach a sensor from an object again, select the object and press the remove button.

Figure 16: Remove button.

2.6 Orientation

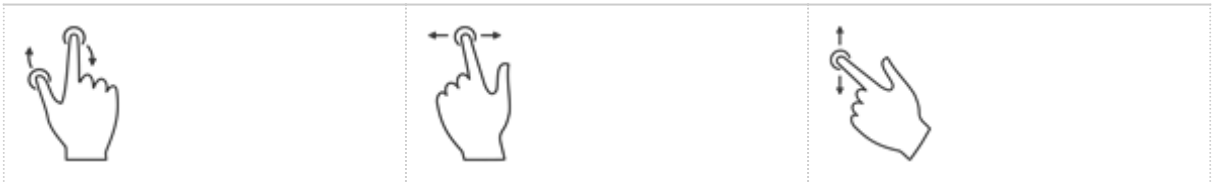
The orientation of an object is static and is initially determined by the default orientation assigned to the object when it was added to the scene. Editing the orientation of an object means changing its orientation relative to the scene's virtual environment. Controlled models on the other hand have a dynamic orientation adopted from the physical sensor. Editing the orientation of a controlled model means changing the orientation of the virtual object relative to the orientation of the sensor. This can also be called 'static calibration'.

Free orientation

The first way of changing the orientation of an object or model is by free orientation. To use this function, make sure an object or model is selected as well as the orientation-button (figure 17).

Figure 17: Orientation button.

Now, the orientation of the selected object or model can be edited by rotating it around one of its axes. Rotation around an axis is done by the finger controls depicted below.



Rotation around an axis **perpendicular** to the view plane is done by using a two-finger rotation.	Rotation around the axis **vertical** to the view plane is done by panning sideways with one finger.	Rotation around the axis **horizontal** to the view plane is done by panning up and down with one finger.

These gestures can be used as illustrated in figure 18.

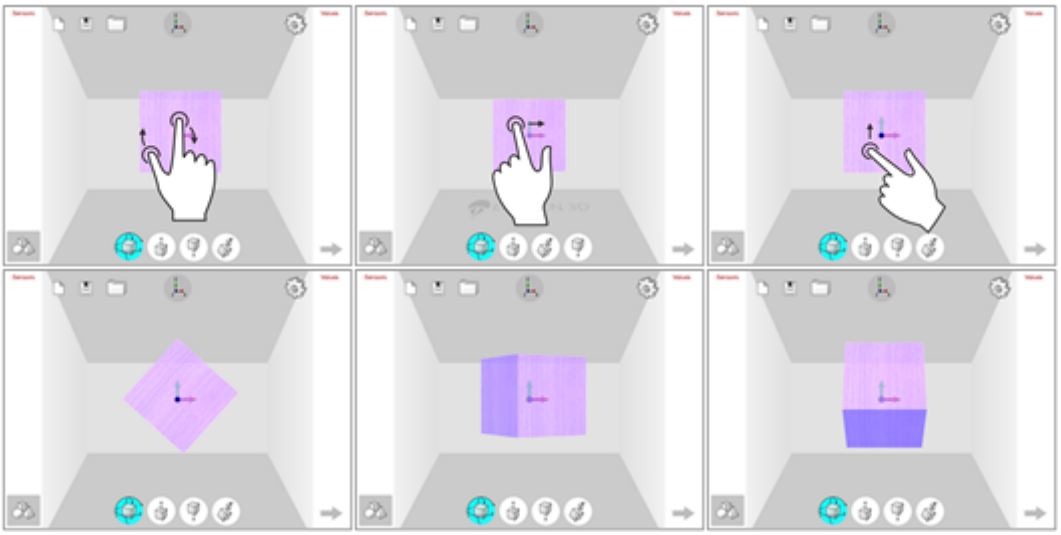


Figure 18: Changing the orientation using gestures.

Vertical alignment

Next to changing the orientation of an object or model using the finger controls, the vertical-alignment-buttons can be used (figure 19). These buttons will appear as soon as the orientation-button and a single object are selected. It automatically aligns the local y-axis of the selected object with the direction of the virtual gravitational force of the scene. The first button will align the object upright, the second will align the object upside down.

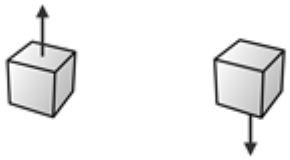


Figure 19: Vertical alignment buttons.

It is advised to first place the object with a sensor attached to it, in an upright position and then press the align-upright button. This sets the virtual model upright as well. This process is illustrated in figure 20.

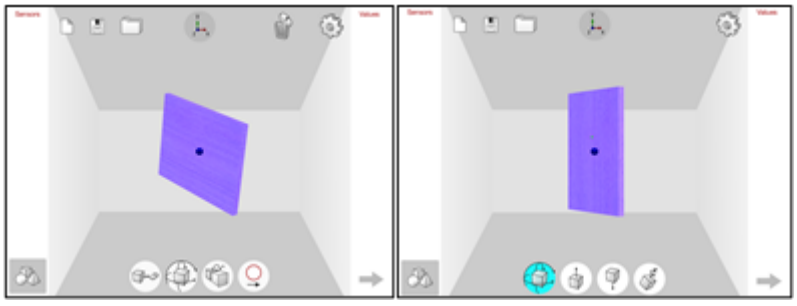


Figure 20: Vertical alignment of an object placed upright.

Horizontal alignment

The orientation of an object or model can be changed using the horizontal-alignment-button (figure 21). This button will appear as soon as the orientation-button and a single object are selected.

Figure 21: Horizontal alignment.

It is to be used after the object with a sensor attached to it is tilted slightly forward. Knowing the object is tilted forward, this function can automatically make sure the horizontal axis of the virtual object and the horizontal axis of the sensor are aligned correctly.
The result is that the front of the physical object corresponds with its virtual counterpart.

Heading alignment

Lastly, the orientation of an object or model can be changed using the heading-alignment button (figure 22). This button will appear after two models or an object and a model are selected.

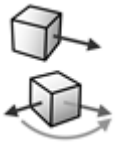


Figure 22: Heading-alignment button

The order of selection is of vital importance. The object or model that is selected second will get the heading of the first selection.
Note that heading alignment is fundamentally different from the vertical or horizontal alignment. Instead of adjusting the orientation of an object while keeping the orientation of the sensor the same, heading alignment means rotating the sensor around the y-axis of the virtual scene.
Note that the heading alignment can also be specified manually by adjusting the “Heading offset” in the properties window of the model.

2.7 Joints

In KineXYZ, a joint is a connection between two objects (and/or models). It is represented in the scene as a visible object and can be selected and manipulated. The rest of this paragraph will explain how to add joints between objects.

Adding joints

First make sure, at least two objects are present in the scene. Now the joints-button (figure 23) appears. Press this button and the connection points of all objects in the scene will become visible.

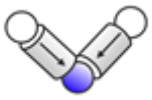


Figure 23: Joints button.

Now a joint can be formed by selecting two connection points. The joint is visualized as a blue coloured connection point. The process is illustrated in figure 24.

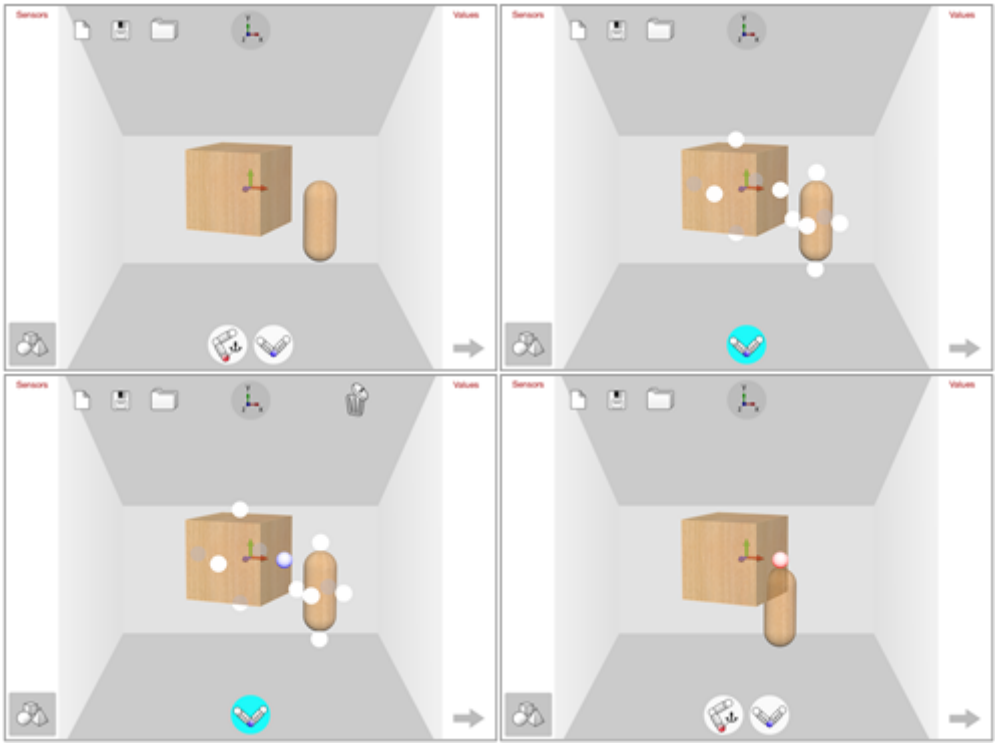


Figure 24: Creating joints by connecting objects.

***Note**: the first joint in a kinematic chain automatically turns into an anchor-point. This is discussed in more detail in the next section.*

2.8 Anchor points

A kinematic chain is built by connecting objects. Such a structure of objects always has a fixed point which is the starting point in determining the pose of the kinematic chain. This fixed point is called an anchor-point. By default, an anchor point is created as soon as two objects are joined together. However, in KineXYZ you can also adjust the anchor point of a kinematic structure. How this is done is explained next.

Adding anchor points

Now the Anchor-point-button (figure 25) appears as soon as an object is added to the scene.

Figure 25: Anchor button.

Press the anchor button and the connection points of all objects in the scene will become visible. Now an anchor point can be defined by selecting one of the connection points on the

kinematic chain. The anchor point is visualized as a red coloured connection point as illustrated in figure 26.

Figure 26: Changing the anchor point of a kinematic chain.

2.9 Values

Each object or model has a sensing point. By default, this is the origin in the local coordinate system of the object or model. This sensing point of an object has the following predefined kinematic values.

- q = orientation (quaternion)
- p = position (x,y,z coordinate in the scene)

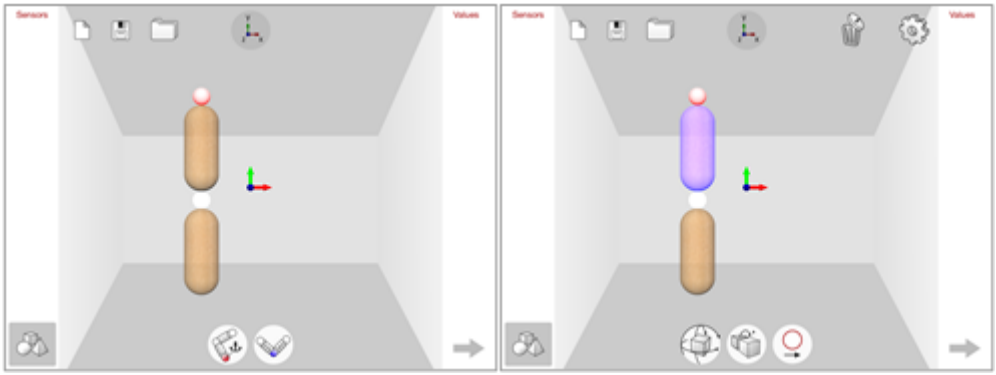
These values update with each movement of the corresponding object model. Once selected these values become available in the dataflow diagram view. How to select (and deselect) these kinematic values for further processing is explained in the remainder of this paragraph.

Selecting values

First select an object or model. Next, press the values-button (figure 27).

Figure 27: Values button.

Now the orientation and position values appear (see figure 28) and can be used in the dataflow diagram.



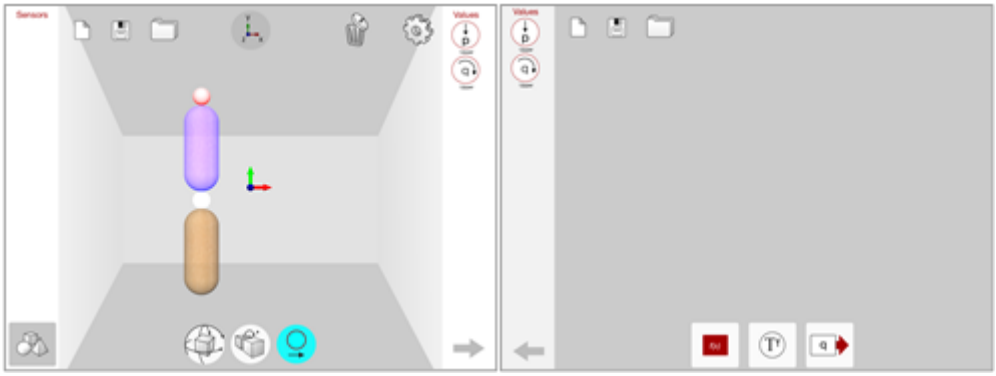


Figure 28: Pressing the values button allows the values to be used in the dataflow diagram.

Besides the values of objects, also the position of a joint can be used. To do this, the joint must be selected and the values button pressed as illustrated in figure 29.

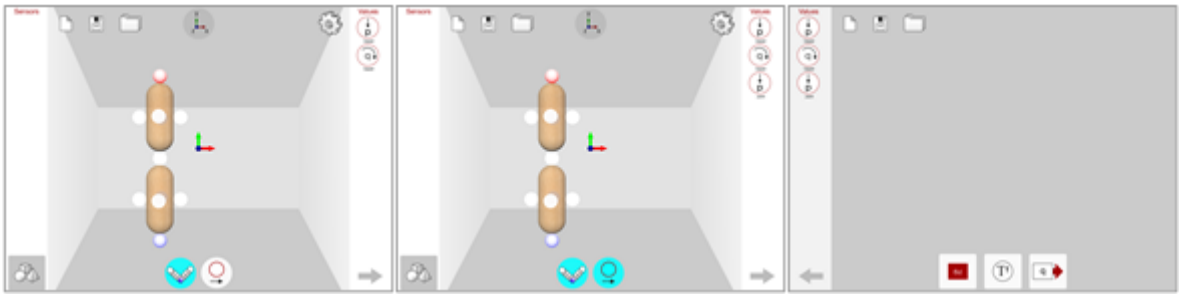


Figure 29: Selecting position from a joint.

Values can be de-selected by selecting the object or model and pressing the values-button (again). Now the values will disappear from the values column.

2.10 Combine

Basic objects can be added to the scene in the kinematic modeller. However, once in the scene these objects can be combined into new objects. Combined objects, move, are selected, are removed, can be copied and have properties like a single object. The sensing point of a combined object is set to be the same as the sensing point of the object which was selected first before combining. There are two main reasons to combine objects.

First, by combining objects, the virtual object in KineXYZ can get a closer resemblance to actual objects. This makes it easier to do sensor to object alignment.

Second, by combining objects the connection points are combined as well. This way, connection points can be placed anywhere on the object.

Combining Objects

First, select the two objects you like to combine. Next press the combine-button (figure 30).

Figure 30: Combine button.

Now the objects are combined as illustrated in figure 31.

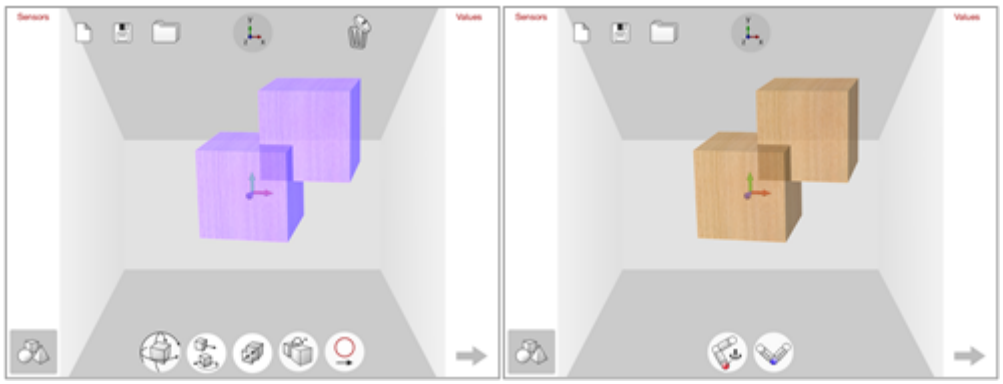


Figure 31: Pressing the Combine button when two objects selected combines them into one.

Removing connection points

When combining objects, the resulting object is likely to have many connection points. Often there will be so many connection points, it becomes hard to select the right one. To overcome this difficulty, redundant connection points can be removed.

To remove a connection point, press the joint-button, select an unnecessary connection and press the remove-button. The steps are given in figure 32.

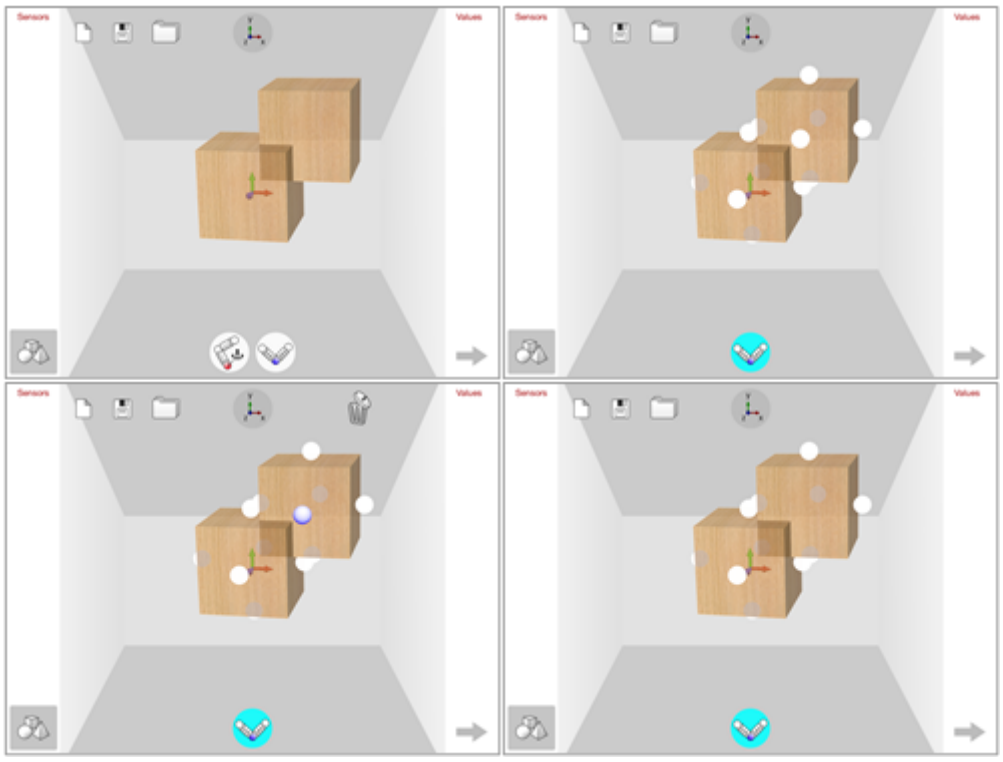


Figure 32: Steps to remove a redundant connection point.

2.11 Copy

In KineXYZ, single objects as well as selections of multiple objects can be copied. When building structures which contain many of the same objects, coping can be very useful. Also, when an object's properties were customized, the copy function makes it easy to add multiple of exactly the same objects to the scene. Note that sensors cannot be copied, since they represent a physical sensor. How to copy a single or multiple objects is explained next.

Copying Objects

First, select one or more objects. Next, press the copy-button (figure 33). Now a copy of the selected object(s) will be added to the scene at the origin.

Figure 33: Copy button.

2.12 Remove

Objects, joints and sensors can be removed from the scene using the same button. When removed, an object will disappear from the scene. When deleting a joint, all objects connected by that joint will disconnect. Removing a sensor will move it back to the Sensors-column. Note that there is no undo function, a remove action is irreversible. How to remove it is explained next.

Removing objects, joints or sensors

First, select one or more objects, joints or sensors. Next, press the remove-button (figure 34). Now the selected object(s), joint(s) or sensors(s) will be removed from the scene.

Figure 34: Delete button.

2.13 Clear scene

The scene in the kinematic modeller can be cleared using the clear-scene-button (figure 35).

Figure 35: Clear-scene button.

Next, you are asked to confirm whether the scene should really be cleared by a yes-or-no pop-up question (see figure 36).

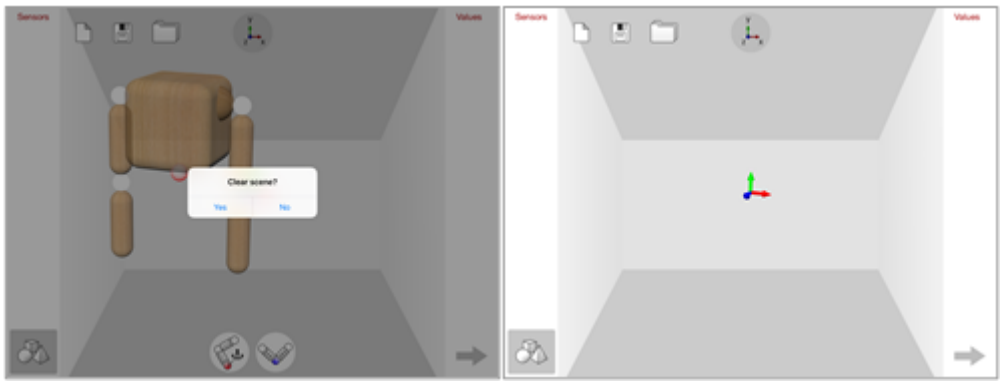


Figure 36: Clearing the scene.

2.14 Save and import

A scene that was built in the kinematic modeller can be saved. After saving the scene will appear in the scene list. From here, saved scenes can be reopened at a later moment or during a new session. Saving is required before a scene can be shared.

Saving a scene

In order to save a scene, press the save-button (figure 37). Next the save scene window appears. Here the scene can be given a name and a description. If the scene is to be shared, it can be handy to leave an email address in the description of the scene.

Figure 37: Save button.

Import a scene

In order to import a scene, press the folder-button (figure 38). The scene-list window will appear showing all the scenes that were saved on the iPad.

Figure 38: Folder button.

To import a scene, first select a scene. Next press the import-button (figure 39) in the bottom right corner of the scene-list window.

Figure 39: Import button.

2.15 Share

Scenes that were saved can be shared via email. This allows for easy collaboration over long distances.

Sharing a scene

In order to share a scene, press the folder-button (figure 38). The scene-list window will appear showing all the scenes that were saved on the iPad. To share a scene, first select a scene and then press the email-button (figure 40) next to the import-button in the bottom right corner of the scene-list window.

Figure 40: Email button.

Receiving a scene

If a scene is shared, it is sent to its receiver via an email with an attachment. The receiver opens this email on his/her iPad. Next, the attachment must be opened with KineXYZ. To do this, press and hold the attachment icon. A window will appear, from which the KineXYZ icon must be chosen to open and save the attachment in the KineXYZ app.

3. Dataflow diagram

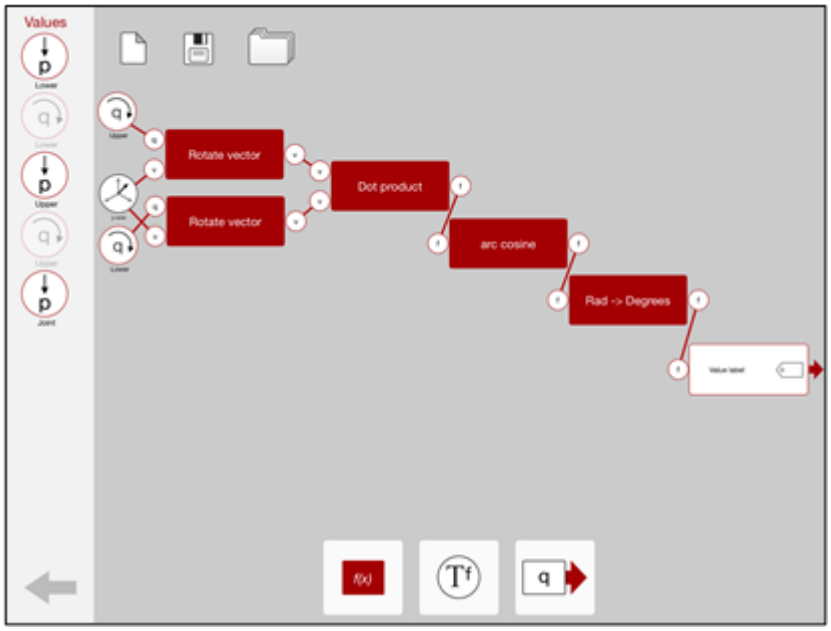


Figure 41: Dataflow diagram

The dataflow diagram is designed to process the values which were selected in the kinematic modeller. The goal is to set up streams which can be executed in the player. This way, among many others, the orientation of an object or model can be streamed to other devices, or the angle between two objects can be shown in real time.
Setting up streams is done by building a dataflow diagram out of nodes and relations. In total there are four different nodes; value-nodes, function-nodes, constant-nodes and stream-nodes. How to use these nodes in building a dataflow diagram and setting up streams will be explained in this chapter.

3.1 Scrolling

The dataflow diagram view shows a 2D canvas. One can scroll around on this canvas and zoom in and out.

Moving over the canvas

To move over the canvas, use the one finger panning movement.

Figure 42: Panning movement to move over the canvas.

Zoom

To zoom your canvas, use a two finger pinching movement. Zoom out when brought together, zoom in when moved apart.

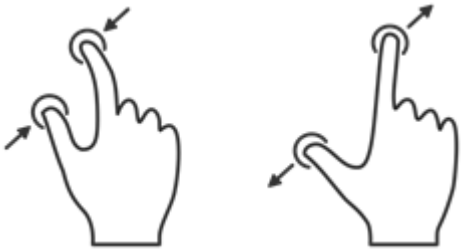


Figure 43: Pinching gestures to zoom in and out.

3.2 Value-node

The values that were selected in the Kinematic modeller view are now shown as value-nodes in the Values-column on the left side of the interface. These value-nodes represent the position or orientation of objects or models in the kinematic modeller (figure 44).

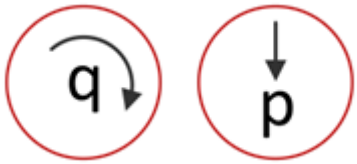


Figure 44: Value nodes (orientation) and (position).

When dragged into the dataflow diagram, these values can be connected to other nodes, allowing these values to be processed as illustrated in figure 45.

Figure 45: Adding values in the diagram.

3.3 Functions

Functions require one or more input values and return an output value. The inputs are shown as small circles on the left side of the function, outputs are shown on the right side.

Figure 46: The rotate vector function.

In the middle of both input and output circles a character is shown, indicating the type of value it can be connected to. In total there are 5 different types of values.

- Quaternion (q)
- Vector (v)
- Float (f)
- Boolean (b)
- String (s)
- Integer (i)

Adding functions

Adding a function can be done by dragging the function-button (figure 47) onto the canvas.

Figure 47: Function button.

The location can be chosen where the function must be placed (see figure 48). This can be changed later as well by dragging the function around.

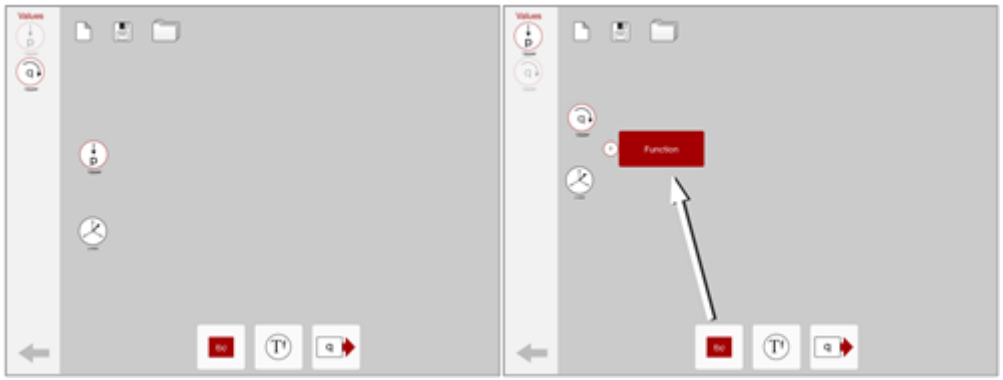


Figure 48: Dragging function node into the diagram.

Now the function window will appear, showing all functions to choose from. Press on one of the functions to select it and add it to the dataflow diagram as illustrated in figure 49.

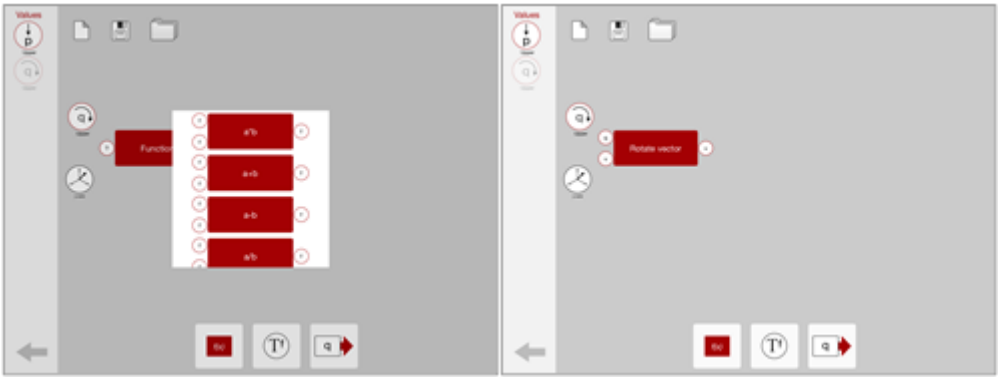


Figure 49: Selecting a function.

Function properties

After a function is added to the dataflow canvas, it can be selected by pressing the properties-button, its properties can be viewed and or edited (see figure 50).

Figure 50: Function properties.

3.4 Constants

Constants contain static values which can be integrated into the dataflow diagram much like values. These static values can be customized by the user.

Adding constants

Adding a constant-node can be done by dragging the constants-button (see figure 51) onto the canvas.

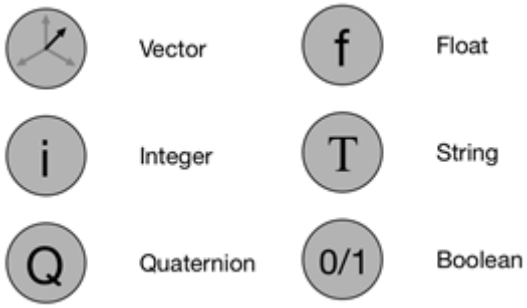
Figure 51: Constants button.

Now the constants-window will appear, showing all constants to choose from (see figure 52).

Figure 52: Adding a constant node.

All constants

In total there are 5 different kinds of constants to choose from:



Customizing constants

To customize a constant, select the constant and press the properties button. Next, a window will appear showing all values that can be customized for the selected constant. This process is illustrated in figure 53.

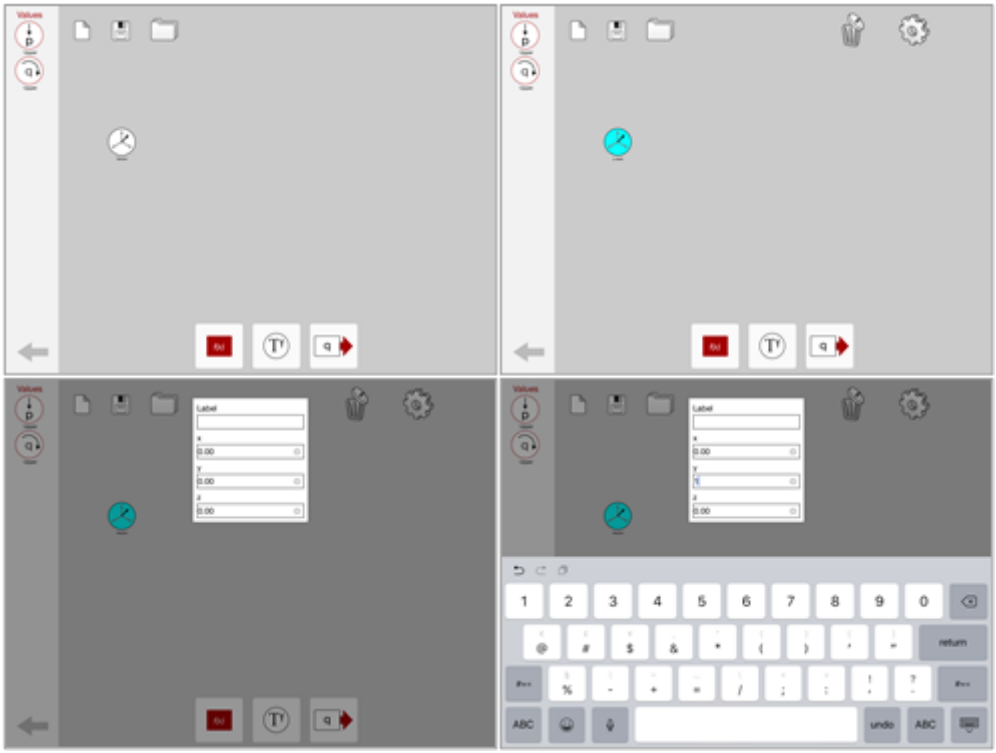


Figure 53: Changing the properties of a constant.

Tip

When filling in “pi” for a float value, the float will get the value of *pi*

3.5 Output

Output-nodes can transfer values from the dataflow diagram to other places both inside and outside the KineXYZ app. Values derived from models, objects and/or complete kinematic chains can be displayed using a label output or sent to other apps/devices using the output-node.

Adding outputs

Adding an output-node can be done by dragging the output-button (figure 54) onto the canvas. Now the outputs-window will appear, showing all outputs to choose from.

Figure 54: Output button.

All outputs

In total there are the following output-nodes to choose from:

	**Set anchor joint**, takes the name of the joint as input and sets it as the anchor-point for the associated kinematic chain.
	**Set orientation**, takes the name of an object as input and sets its orientation according to given input. Note that it is not possible to set the orientation of a model that is controlled by a sensor.
	**Network stream**, allows streaming the value(s) assigned to this streamer to other devices by IP address and a Port number.
	**Set position**, takes the name of an object as input and sets its position according to given input.
	**Value label**, shows a label with a live update of the value assigned to this streamer in the dataflow diagram.
	**Translate**, takes the name of the object as input and translates the position with the given vector.

The details of using the “Network stream” are specified in more detail in “Streaming”.

3.6 Connecting

Connections between nodes define how the data ‘flows’ between nodes. These connections are represented by red lines. How to add these connections is discussed next.

Adding connections

To add a connection, select an output and then an input or vise versa. Note that when an input or output is selected, only inputs and outputs of the same value type remain available for selection. This process is illustrated in figure 55.

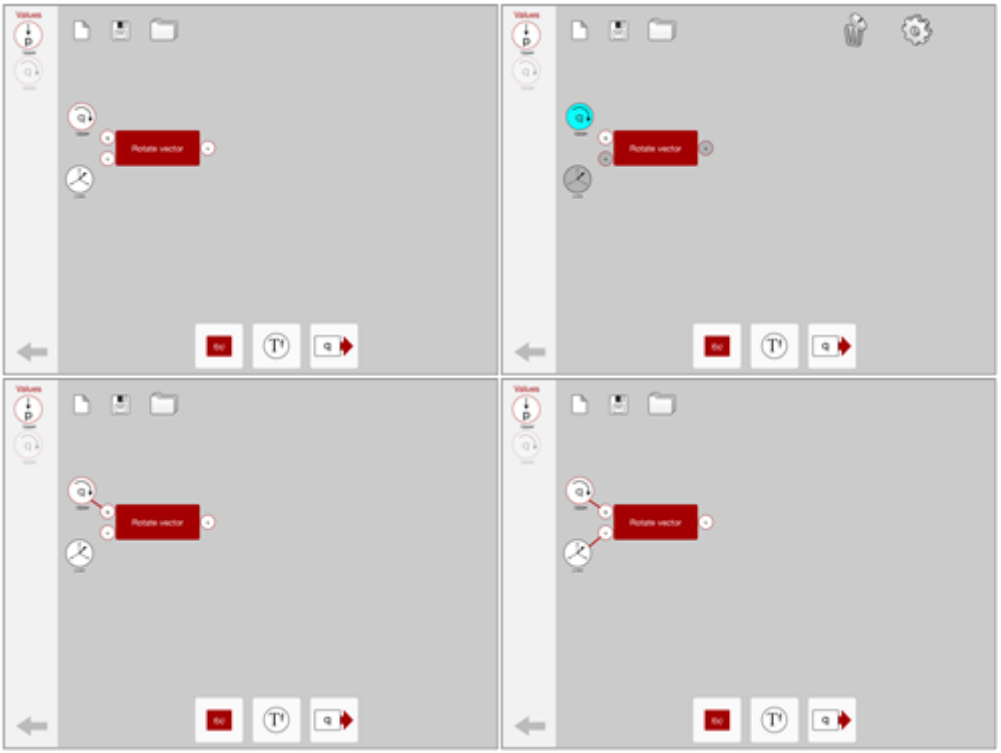


Figure 55: Adding connections.

3.7 Containers

A container can contain functions and constants. One container can even be a fusion of one or more other containers. In the dataflow diagram, containers look very much like functions, only containers have a gray border.

Containers are useful for two main reasons. First, containers make a dataflow diagram easier to understand. Second, containers make a dataflow diagram well-organized, since a single container-node can represent many other nodes.

Making a container is done by selecting and dragging one node over another node until both are highlighted. Releasing the selected node fuses the highlighted nodes into a single container. However, dragging just any node over another node will not always result in both nodes getting highlighted. This can happen because two nodes can only be fused if all the following criteria are met:

- Only constants, functions and containers can be fused.
- Only connected nodes can be fused.
- Two nodes are only highlighted if the node that is dragged over the other node is an input for that other node.

- The output of the node that is dragged over the other node may not be connected to multiple nodes. (note: in KineXYZ, nodes including container-nodes, can only have a single output).

3.8 Remove

Both nodes and connections can be removed from the dataflow diagram. Multiple nodes and/or connections can be removed simultaneously. Note that there is no undo function, a remove action is irreversible. How to remove is explained next.

To remove a node, select the node. Next, press the remove-button. The node will be removed from the dataflow diagram. When removing a node, all connections to and from that node are removed as well.

To remove a connection, select the connection line. Next, press the remove-button. The connection will be removed from the dataflow diagram.

3.9 Save and import

A kinematic chain that was built in the kinematic modeller can be saved. This is done using the save button.

3.10 Share

A kinematic chain that was saved can be shared via email.

4. Streaming

4.1 Overview

To be able to use the data generated in the dataflow diagram in external applications, KineXYZ offers the possibility of *streaming* data as an XML or JSON string inside a UDP packet to an external device or *companion* app on the iPad itself.

To do this, the following steps are to be taken:

1. Set the object and property name of the value.
2. Extract the timestamp.
3. Set the object and property name of the timestamp.
4. Add a Network Streamer node to the dataflow diagram.
5. Configure the Network Streamer.
6. Connect the nodes.

These steps are described in detail in the following sections, but first some background information is given on the format of the UDP packet.

4.2 Background

The values generated in a dataflow diagram can be streamed on the network. Consider the example of a segment called “Upper arm”. The position and orientation of this object can be

streamed to the network using the network stream as given in figure 56.

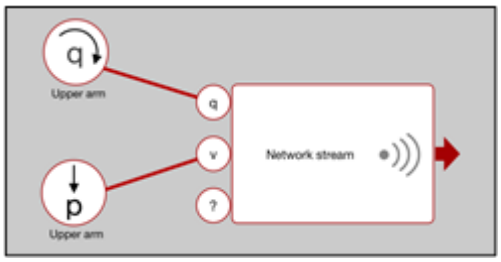


Figure 56: Streaming the kinematic values of a segment.

The data can be sent as an XML or JSON string inside a UDP packet. The format of the XML string for this example is as follows:

```
<UpdateObject><Object Name="Upper arm"><Position><x>-0.1112</x><y>-0.0934</y>
<z>0.0200</z></Position><Orientation><x>-0.1511</x><y>0.9944</y><z>-0.0623</z><w>-0.0101</w></Orientation></Object></UpdateObject>
```

The bold red are the scalars of the actual kinematic values. By default these values already are associated with an object and have a property name ("Position" and "Orientation"). However, this is only the case for these default values. For all other values the *object name* and *property name* are not defined by default and must be specified explicitly.

How this is described in the following section.

4.3 Setting the object and property name

The problem with streaming a calculated value like a joint angle to an external application is that the generated value itself is not associated with a specific single object and no property name is specified either.

This means that if the output would be connected to a network stream, nothing would be transmitted. It is therefore that an object name and property name must be specified for the calculated value.

To give some background first, the following describes how the object and property name are used to create the XML string.

Note first that all values in the diagram have the following properties:

Field	Type	Description
timestamp	float	The system time at which the value was received or generated, expressed in us.
objectName	string	The name of the object associated with the value.
propertyName	string	The name of the property that is represented by the value.

The XML string starts with the following:

```
<UpdateObject><Object Name="OBJECT_NAME">
```

Next, for each property associated with the object, the XML representation of the property will be added. This starts with:

```
<PROPERTY_NAME>
```

The XML representation of the property ends with:

```
</PROPERTY_NAME>
```

When the XML representation of all the properties are added, the XML string ends with:

```
</Object></UpdateObject>
```

Now that we know how KineXYZ generates an XML string, assume that we want to stream for example a joint angle (for a full description on how to calculate a joint angle see the document "KineXYX quick start - Streaming a joint angle") and that the XML-string must be as follows:

```
<UpdateObject><Object Name="Joint">
<angle>angle</angle>
<timestamp>timestamp</timestamp>
</Object></UpdateObject>
```

To explicitly set the object name, the "Set object name" function is used. Likewise, to set the property name, the "Set property name" function is used. These functions take a string as input and a value for which the name needs to be set.

The diagram to set the object name to "Joint" and the property name to "angle" is given in figure 57.

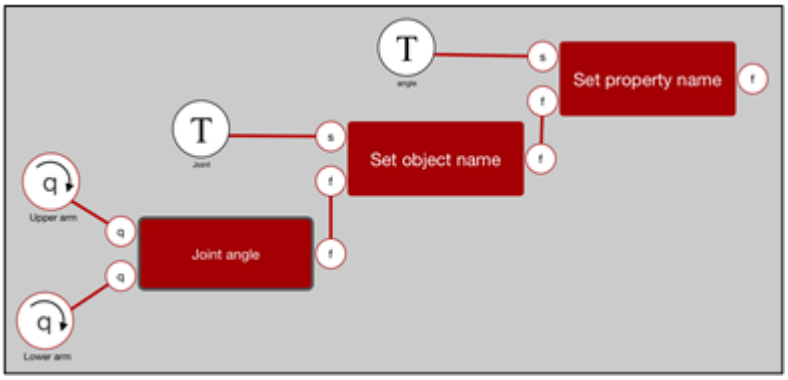


Figure 57: Setting the object name to "Joint" and the property name to "angle".

The output of this diagram now generates a float value with the object name and property name filled in.

4.4 Extract the timestamp

The next step is to extract the timestamp as a value. This can be done by using the "Timestamp" function and extending the diagram as given in figure 58.

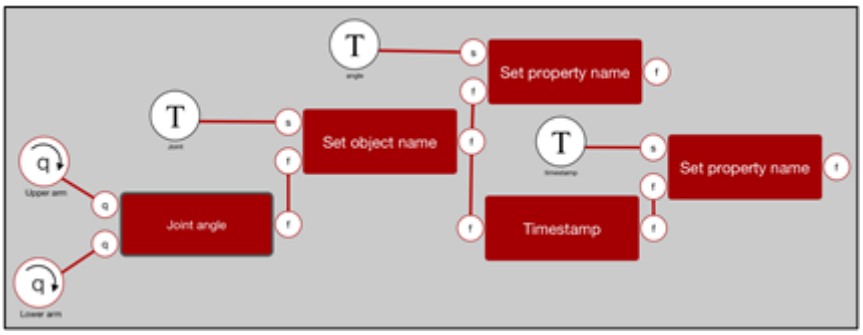


Figure 58: Extracting the timestamp and adding it as a property.

Both outputs can now be connected to a network stream as illustrated in figure 59.

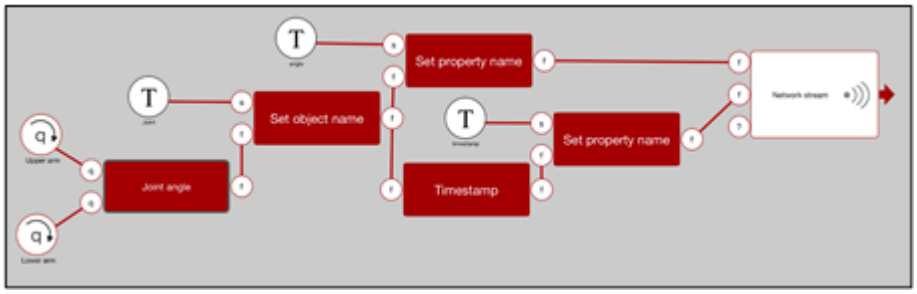


Figure 59: Streaming the two properties.

Since both properties now have the same object name, they are packed as a single packet.

4.5 Configure the network stream

Note that the default format of the network stream is specified as “JSON”. If the XML format is required, this must be changed in the properties of the network stream (see figure 60).

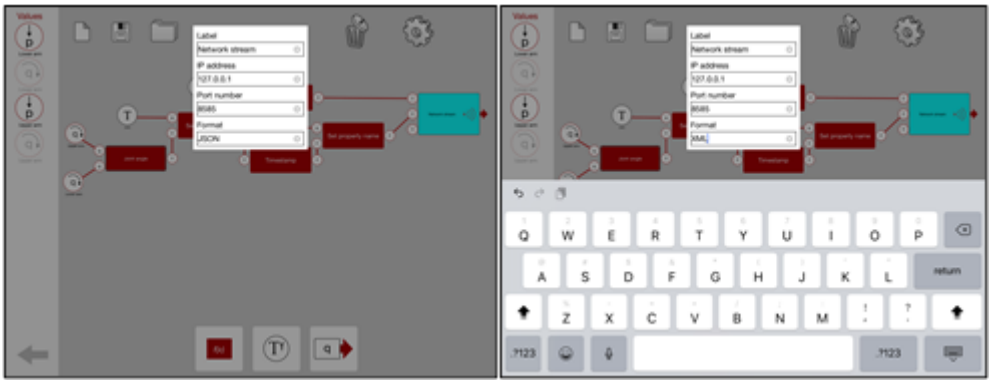


Figure 60: Properties of a Network Stream node.

Note: the displayed IP address 127.0.0.1 belongs to the host name “localhost”, so on the device itself. This host name can also be typed explicitly if so required.

***Note**: because KineXYZ works on updates, a Network streamer must have received updates for all values for a specific object. Otherwise no UDP packet will be generated.*