NAO

Eine Einführung

1. JULI 2017

ANNA PÖSSNIKER, DAVID BAUMGÄRTEL

Inhaltsverzeichnis

Erste Schritte

Wichtige Vorabinformationen

Der NAO-Roboter sollte alle drei Monate aufgeladen werden, damit die Batterie keinen Schaden davonträgt. Zudem darf er weder beklebt noch angezogen werden, da ansonsten Sensoren beschädigt werden können oder der Roboter überhitzt.
Nach dem Laden des Roboters sollte zudem kurz gewartet werden, da der Kopf des Roboters heiß werden kann.

Login im Aldebaran-Store

Um sich im Aldebaran-Store anzumelden, kann man folgenden Link nutzen: *https://sso.aldebaran-robotics.com/pf/adapter2adapter.ping?TargetResource=https://store.aldebaran.com/*.

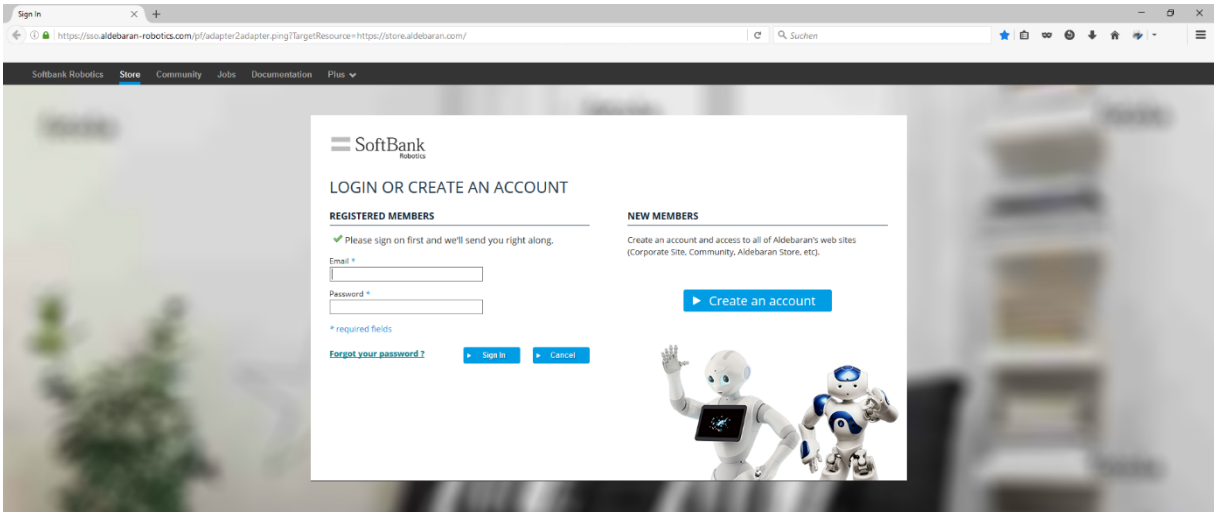


Abbildung 1: Die Aldebaran Login-Seite

Unboxing

NAO sollte vorsichtig aus seiner Verpackung geholt werden und in eine aufrechte Sitzposition auf einem harten, flachen Untergrund gebracht werden. Dieser Untergrund sollte kein Teppichboden sein!
Nun sollte der „Hello Guide“ und der „Safety Guide“ gelesen werden.

Ein/Ausschalten des NAO

Um den NAO ein- und auszuschalten muss lediglich der Button auf seiner Brust gedrückt werden. Beim Anschalten reicht es diesen 1 Sekunde lang zu drücken, beim Ausschalten sind ca. 5 Sekunden nötig. Ein Sicherheitsausschalten ist durch das Drücken des Knopfes für 9 Sekunden implementiert.
Zweimaliges Drücken des Brustbuttons löst die Anspannung der Motoren, sodass NAO wieder bewegt werden kann ohne dabei Schaden anzurichten.

Inbetriebnahme

Nachdem der NAO per Kabel ans Ethernet angeschlossen und dann gestartet wurde, gibt er auf Knopfdruck seine IP-Adresse aus. Diese ist beispielsweise: *192.168.178.116* . Um auf NAO zuzugreifen, wird diese Adresse direkt im Browser in der Adressleiste eingegeben. Dort wird ein Passwort verlangt. Standardmäßig lautet dies:

Username: *nao*
Passwort: *nao*

Des Weiteren wurde NAO mit dem oben genannten Aldebaran Account gelinkt. Zudem kann man NAO verschiedene WLAN-Zugangsdaten beibringen. Er meldet sich automatisch in ihm bekannten Netzwerken an, wenn er nicht per Kabel am Ethernet angeschlossen wurde und kann dann auch kabellos mit Inhalten gefüttert werden. Dieses Vorgehen wird auch empfohlen, da der NAO bei angeschlossenem Ethernet-Kabel weniger Bewegungsfreiraum hat und die Gefahr besteht, dass er aufgrund des Kabels hinfällt. Es sollte jedoch darauf geachtet werden, dass NAO seine IP-Adresse wechselt, wenn er das Netzwerk oder die Zugangsart auf ein Netzwerk wechselt. Falls er seine IP-Adresse nicht noch einmal ausgibt, sollte zunächst die nächste IP-Adresse getestet werden, also beispielsweise: *192.168.178.117* . Im Normalfall gibt NAO seine IP-Adresse aber durch Drücken auf seinen Brustbutton aus.

Die NAO-Webpage

Über die IP-Adresse des NAO gelangt man auf die Webpage des Roboters. Hier stehen nützliche Informationen über den Akkustand und die aktuelle Firmwareversion. Zudem können hier der Name des Roboters und die Login-Daten verändert werden.

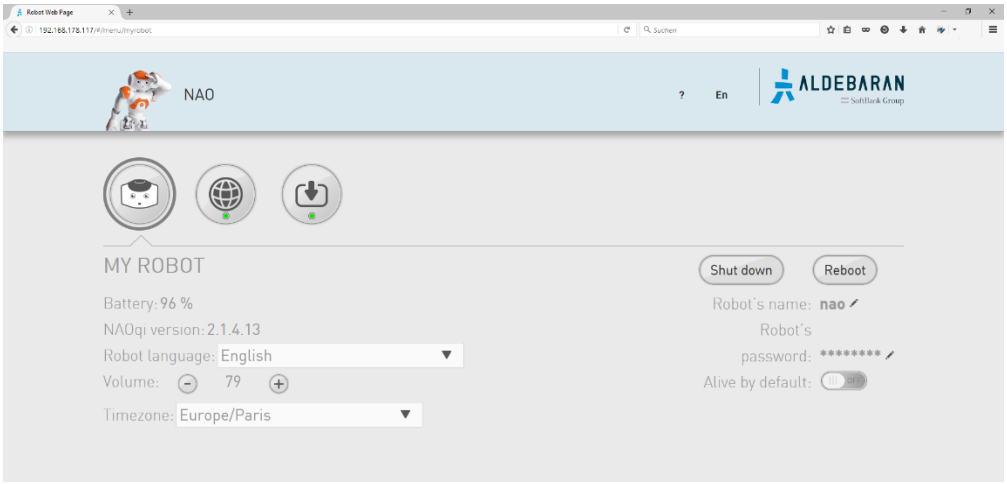


Abbildung 2: Die Webpage des NAO

Kommunikation mit NAO

NAO versteht auch ohne eine installierte Applikation gesprochene Worte und folgt Gesichtern mit seinen Augen, wenn das autonome Leben des Roboters aktiviert ist. Falls seine Augen blau leuchten hört NAO zu. Sind sie grün verarbeitet NAO das Gesprochene. Wechseln sich weiß und blau ab, so hat der Dialog für NAO noch nicht begonnen.

Installieren von Applikationen

Im Aldebaran Store findet man zunächst einen Reiter mit der Aufschrift „NAO“ am oberen linken Bildrand. Durch einen Klick auf diesen gelangt man zur Applikationenseite des NAOs.

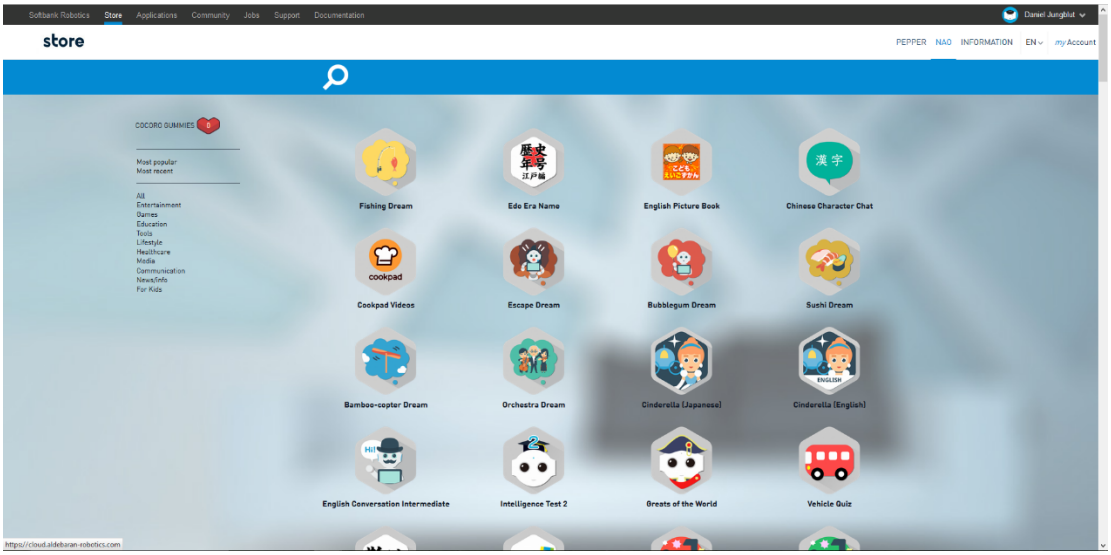


Abbildung 3: Der Aldebaran-Store. Oben links ist ein Reiter mit der Aufschrift "NAO" zu finden.

Auf der Applikationenseite findet man wiederum den Reiter „Manage“ oberhalb des Store-Bildes mit dem Roboter. Durch einen Klick auf diesen Reiter gelangt man zu seinen eigenen Robotern.

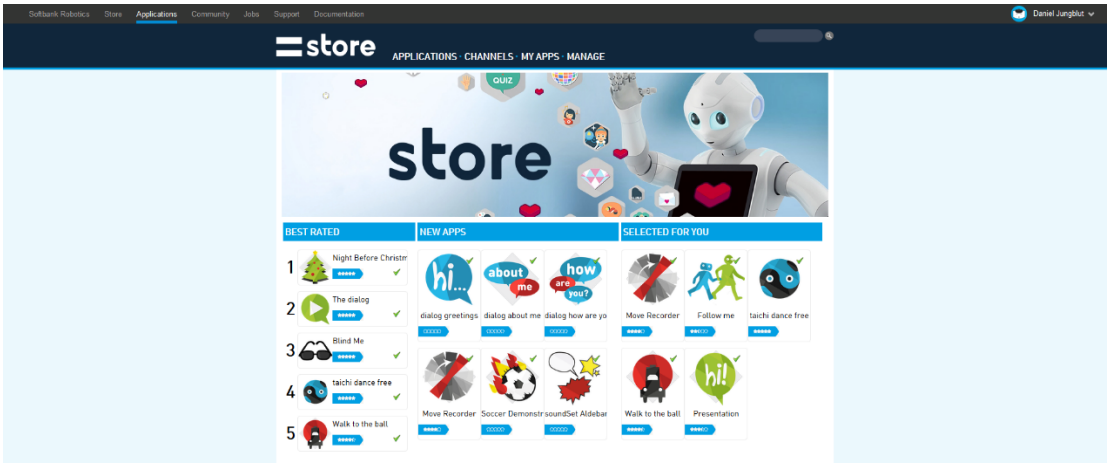


Abbildung 4: Die Applikationenseite. Oberhalb des Storebildes befindet sich der Reiter "Manage"

Durch die Auswahl des gewünschten Roboters im nächsten Fenster kommt man nun zu dem gewünschten Fenster, in dem die automatischen Updates mit Hilfe einer Checkbox oben links aktiviert werden können.

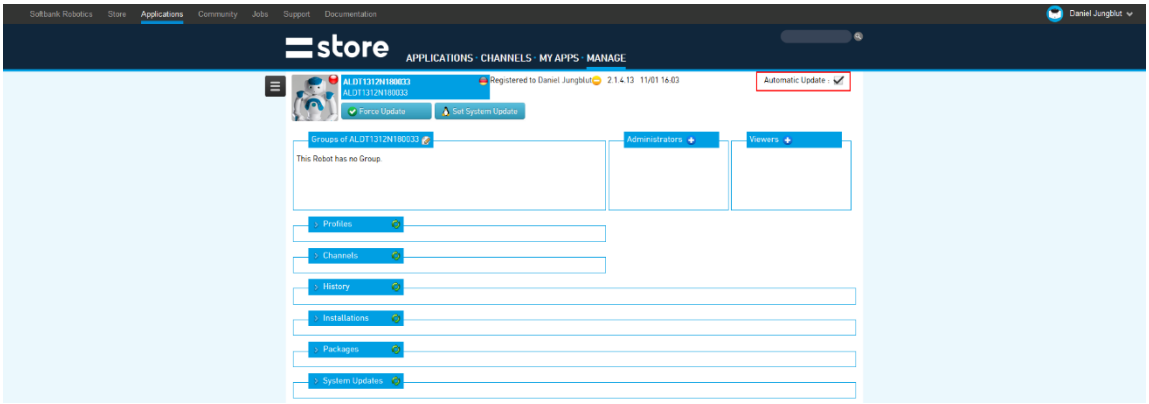


Abbildung 5: Das "Manage"-Fenster des Roboters. Die Checkbox für automatische Updates befindet sich oben links.

Danach können im Store Applikationen ausgewählt und auf NAO übertragen werden. Auf der Webpage von NAO sieht man welche Applikationen installiert sind.
Zu Beginn wurde NAO der „Basic Channel“ beigebracht, bei dem er einfache Bewegungen durchführen kann und auf vorgefertigte Fragen antworten kann.

FTP File Server

NAO kann zudem als eigener Datenserver genutzt werden. Hierfür muss man lediglich über den Link *ftp://nao.local* im Browser auf ihn zugreifen oder nutzt dafür einen FTP Client wie FileZilla. Die Zugangsdaten sind die oben erwähnte Kombination aus Username und Passwort.
Beispielsweise könnte man also wichtige Daten, die man für eine Präsentation des NAOs benötigt direkt auf NAO speichern.

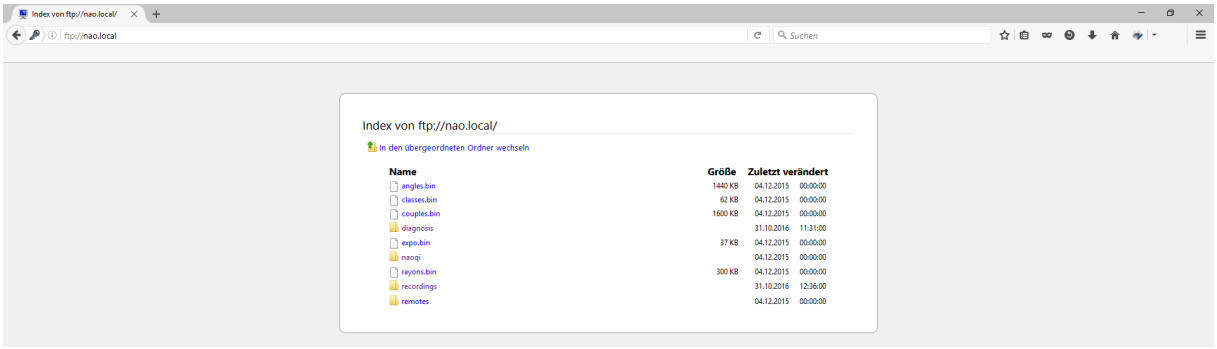
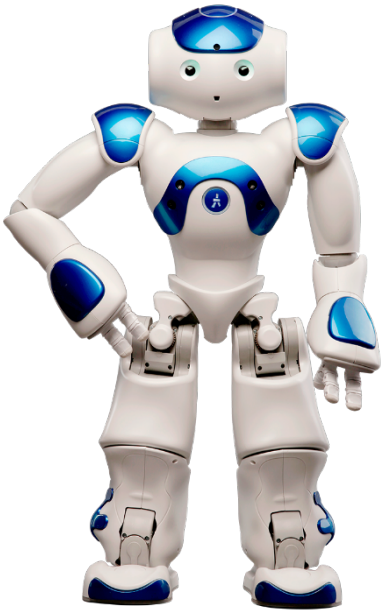


Abbildung 6: Der FTP File Server des NAO

Technische Daten des NAO

Rechenleistung

NAO verfügt über einen Intel Atom Prozessor mit 1,6 Gigahertz und einen Arbeitsspeicher von 1 Gigabyte. Das Betriebssystem des NAO ist das sogenannte NAOqi. Dieses ist lediglich ein Framework, welches auf einem Linux Betriebssystem basiert. Daher ist auch eine Verbindung via SSH mit NAO möglich, empfiehlt sich aber nur für Linux Profis. Wichtig ist zudem der interne Speicher von 8 Gigabyte. Auf diesem sind die Programme von NAO gespeichert.



Interaktionsmöglichkeiten

Durch die Vielzahl an LEDs, welche im NAO verbaut sind, können bereits einfache Interaktionen durchgeführt werden. So können sich diese bei unterschiedlichem Programmablauf unterschiedlich färben, damit die Reaktion des NAO für sein Gegenüber erkennbar ist. Über die zwei integrierten Lautsprecher kann NAO zudem kommunizieren. Zudem bietet sein menschliches Auftreten viele Möglichkeiten auf eine annähernd menschliche Kommunikation, beispielsweise durch Handzeichen oder ähnliches.
Durch das integrierte WLAN Modul kann NAO zudem E-Mails versenden oder erhalten. Mit etwas Aufwand könnte ihm auch das Twittern beigebracht werden.

Motoren

Insgesamt hat NAO 25 Freiheitsgrade in seinen Bewegungen und kann mittels Choregraphe sehr präzise, flüssige Bewegungen durchführen. Sogar komplexe Bewegungsabläufe sind möglich. Beispielsweise könnte NAO das Tanzen beigebracht werden.

Sensoren

Damit NAO seine Umwelt wahrnehmen kann, sind mehrere Sensoren in ihm verbaut. Zwei HD-Kameras mit einer Auflösung von 1280 x 960 Pixeln sind für die integrierte Bild- und Gesichtserkennung zuständig. Eine dieser Kameras befindet sich kurz oberhalb und die andere Kamera kurz unterhalb von NAOs Augenpartie. Über die vier Mikrofone kann NAO bestimmen aus welcher Richtung er Töne wahrgenommen hat und diese aufnehmen. Das Frontmikrofon befindet sich oberhalb der Frontkamera auf dem Kopf des NAOs. Des Weiteren befindet sich ein Mikrofon auf der Rückseite des Kopfes des NAOs und jeweils ein weiteres Mikrofon ist an den Seiten des Kopfes am Rand der Lautsprecher verbaut.
Dass NAO nicht gegen Objekte läuft, können ihm zudem vier Sonare in seiner Brust helfen. Außerdem verfügt NAO über zwei Bumper an seinen Füßen und insgesamt acht Force Sensing Resistoren. Diese geben NAO zurück, ob Kräfte auf ihn wirken und sind ebenfalls in seinen Füßen verbaut. Ein Gyrosensor hilft NAO zudem das Gleichgewicht zu halten.

Batterieleistung

Die 21,6 V 2Ah Lithium-Ionen-Batterie befindet sich am Rücken des NAOs und kann diesen bis zu 90 Minuten lang betreiben. Ein vollständiges Aufladen der Batterie kann mehrere Stunden dauern.

Choregraphe

Die einfachste Methode NAO zu programmieren bietet die Software „Choregraphe“ von Aldebaran Robotics. Diese grafische Programmierumgebung stellt verschiedene vorgefertigte Programmbausteine zur Verfügung und erinnert an die Programmierumgebung von LEGO Mindstorms. Sie sollte auch für Schülerinnen und Schüler verständlich sein und eignet sich daher für den Einstieg in die NAO-Programmierung. Um keinen Schaden anzurichten sollte zudem zu Beginn eine virtuelle Roboterversion genutzt werden.

Download

Unter dem Link *https://community.ald.softbankrobotics.com/en/resources/software/language/en-gb* findet man die Software „Choregraphe“ mit dem NAO programmiert wird. Hierbei muss besonders auf die Firmwareversion, auch NAOqi Version genannt, geachtet werden. Zudem muss nach der Installation der Lizenzschlüssel eingegeben werden um die Software zu aktivieren.

Aufbau von Choregraphe

Beim Start von Choregraphe öffnet sich folgendes Fenster:

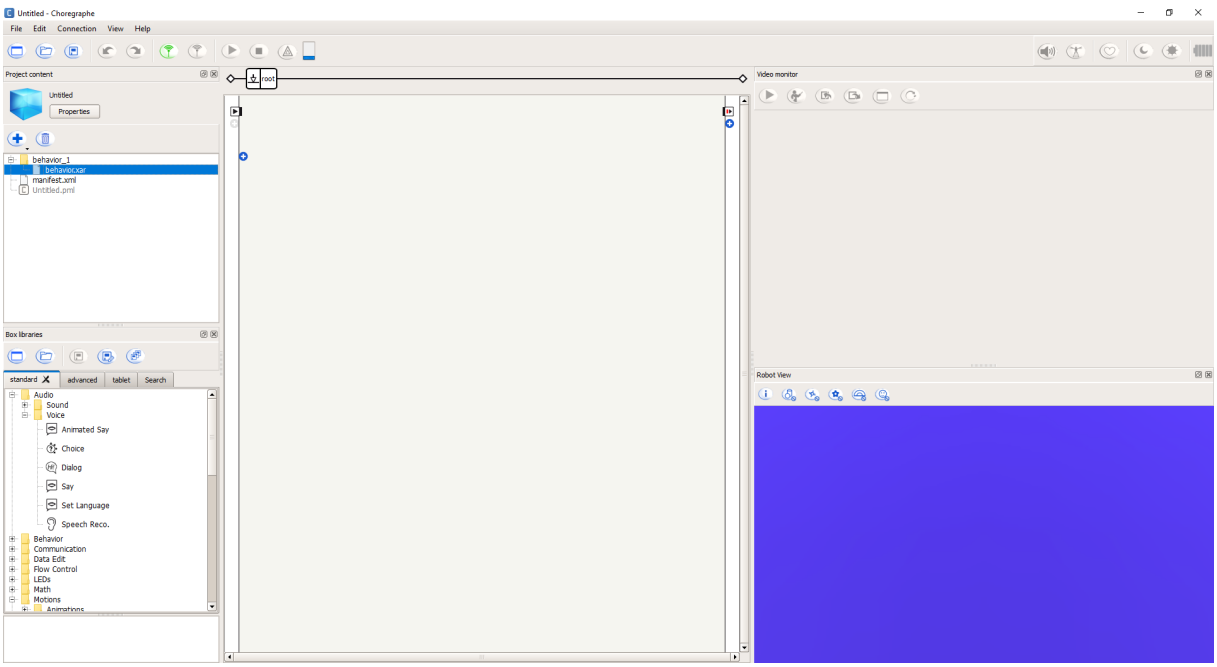


Abbildung 7: Das Choregraphe-Fenster

Man kann die Ansicht in insgesamt sechs Blöcke unterteilen. Oben ist die Menüleiste zu finden – die linke Seite dieser Leiste wird zum Abspeichern eines Projekts und zum Verbinden mit dem NAO gebraucht (s. Abb. 8). Auf der rechten Seite oben sind einige Eigenschaften des verbundenen Roboters, wie zum Beispiel der Akkustand und die aktuelle Lautstärke der Lautsprecher, direkt dargestellt (s. Abb. 9). Zudem kann dort der Animationsmodus aktiviert werden, um NAO Bewegungen beizubringen, und der „Autonomous Life“-Modus kann hier ebenfalls ein- und ausgeschaltet werden, welcher dafür zuständig ist, dass sich NAO von selbst bewegt und mit seiner Umgebung interagiert. Dies kann allerdings bei der Ausführung eigener Programme durchaus störend sein.

Abbildung 8: Die linke Seite der Menüleiste

Abbildung 9: Die rechte Seite der Menüleiste

Auf der linken Seite des Choregraphe-Fensters befinden sich das Projektfenster (s. Abb. 10), sowie das Funktionalitätenmenü (s. Abb. 11). Im Projektfenster sieht man die aktuell auf dem Roboter gespeicherten Projektdateien. Das ist zum einen das programmierte Verhalten, zum anderen können hier aber auch benötigte Audiofiles oder Texte liegen.

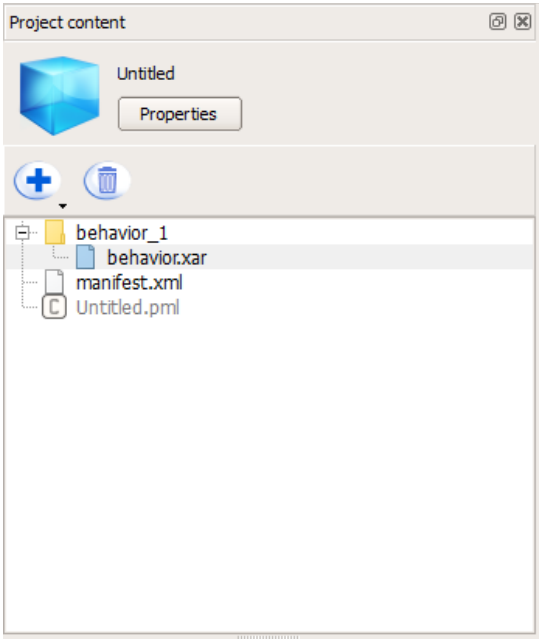


Abbildung 10: Das Projektfenster

Das Funktionalitätenmenü stellt den wichtigsten Teil von Choregraphe dar. Hier liegen vorgefertigte Bausteine bereit, um den NAO programmieren zu können. Zu jedem dieser Bausteine existiert ein Block mit Namen, Bild und Beschreibung. Die einzelnen Bausteine sind bereits in Ordnern sortiert, die die verschiedenen Funktionsbereiche des NAO aufzeigen. In dem Ordner „Audio“ findet man beispielsweise alle Programmteile, die eine Audioausgabe zur Folge haben. Ein wichtiger Baustein ist hier der „Animated Say“-Baustein, der einen Text in eine Sprachausgabe umwandelt und dabei eine natürliche Bewegung des NAOs ermöglicht.

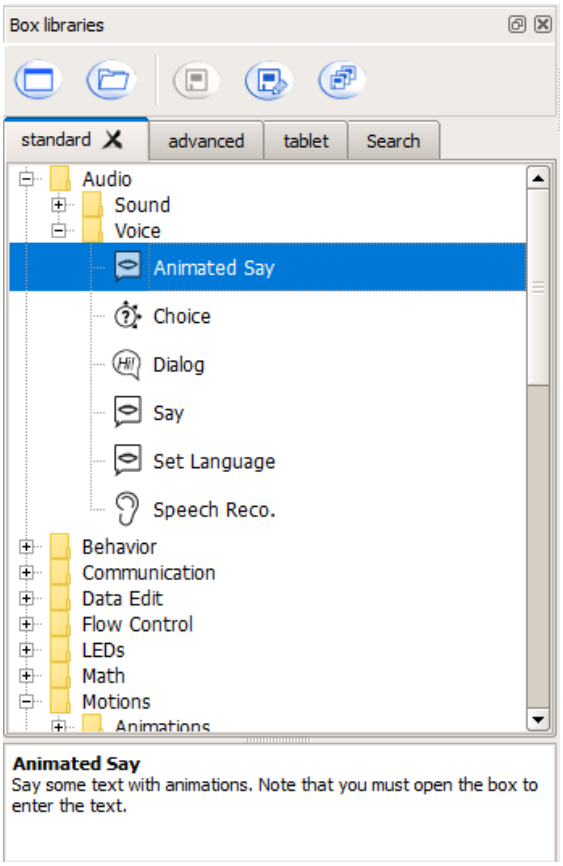


Abbildung 11: Funktionalitätenmenü

In der Mitte des Choregraphe-Fensters ist eine freie Fläche auf der mit Hilfe der Bausteine aus dem Funktionalitätenmenü ein Flussdiagramm aus Programmteilen zu einem Verhalten des Roboters zusammengestellt werden kann (s. Abb. 12). Hier wird der Roboter also grafisch programmiert.

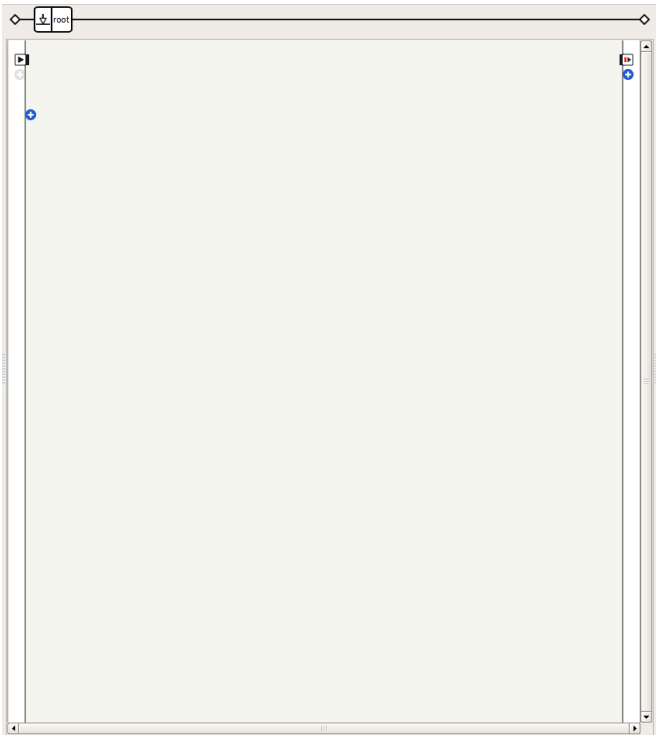


Abbildung 12: Programmierfläche

Über die Plus-Symbole können In- und Outputs zu der Programmierfläche hinzugefügt werden. Durch Klicken auf einen hier liegenden Baustein kann man an diesen Veränderungen vornehmen oder sogar seinen Quellcode betrachten. Die Programmfläche wird in der Literatur auch häufig „Diagramming Space“ genannt.
Zu guter Letzt findet man auf der rechten Seite des Fensters den Videomonitor (s. Abb. 13), die Pose Library (s. Abb. 14), den Robot View (s. Abb. 15 und 16) und die Robot Applications (s. Abb. 17). Der Videomonitor zeigt das Bild, welches der NAO-Roboter aktuell über seine integrierten Kameras aufzeichnet. Da aktuell noch kein Roboter mit Choregraphe verbunden wurde, ist dieses Fenster aktuell grau.

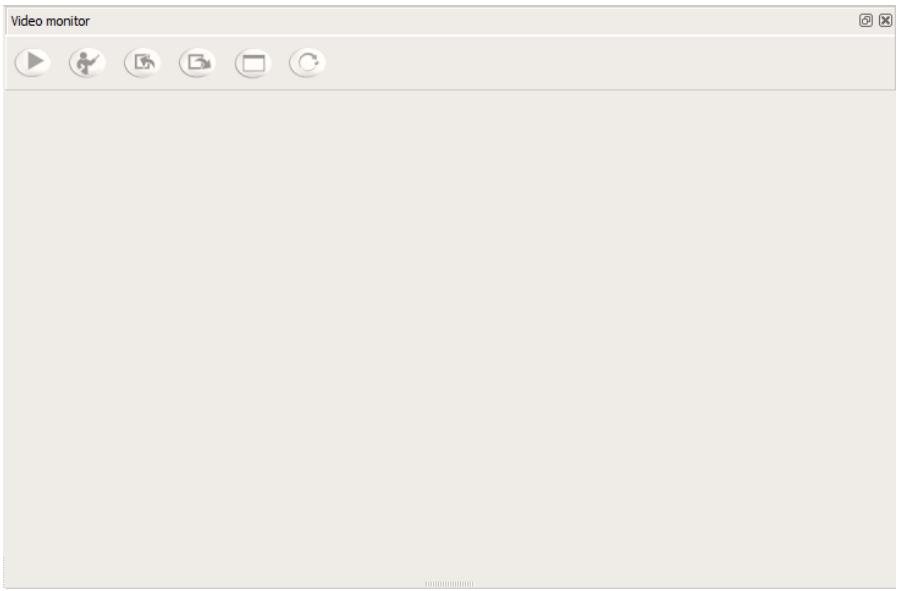


Abbildung 13: Der Videomonitor

Alternativ lässt sich hier die Pose Library einblenden, welche die wichtigsten Posen des NAOs zur Verfügung stellt. Zudem kann man hier eigene Posen des NAOs über einen Klick auf den blauen Plusbutton ablegen, um diese stets schnell griffbereit zu haben.

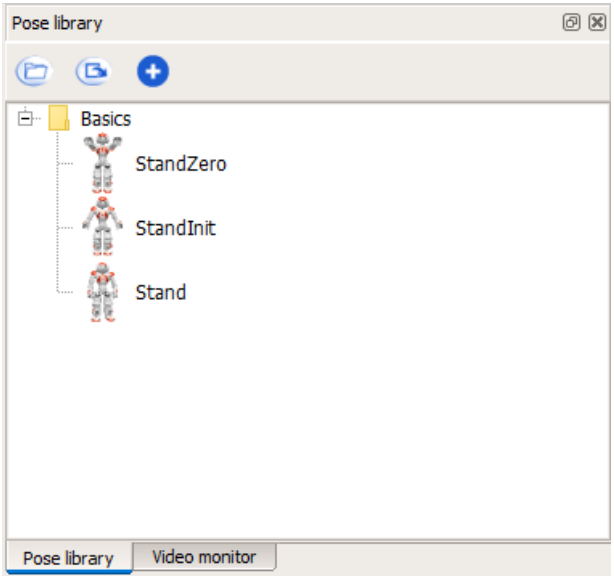


Abbildung 14: Die Pose Library

Im Robot View sieht man den aktuell verbundenen Roboter in einer virtuellen Umgebung. Diese Ansicht ist vor allem praktisch für die Arbeit mit einem virtuellen Roboter oder um zu überprüfen, ob

NAO auch das tut, was er eigentlich tun sollte. Darüber hinaus können Bewegungen einzelner Motoren direkt am Robot View durchgeführt werden.

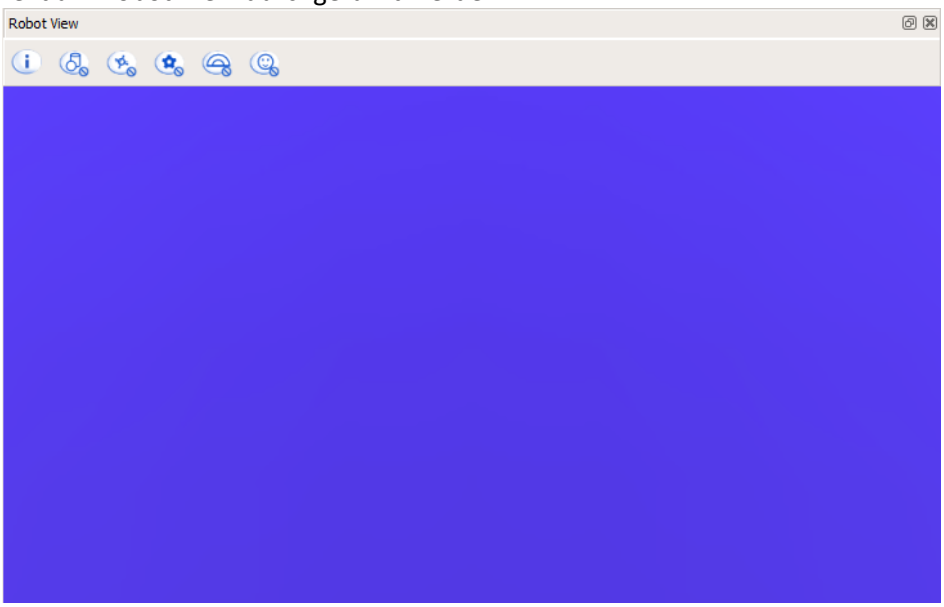


Abbildung 15: Der Robot View

Zu Beginn ist auch dieses Fenster noch leer, da Choregraphe noch mit keinem Roboter verbunden ist. Sobald ein Roboter mit Choregraphe verbunden wurde, sollte das Robot View-Fenster folgendermaßen aussehen:

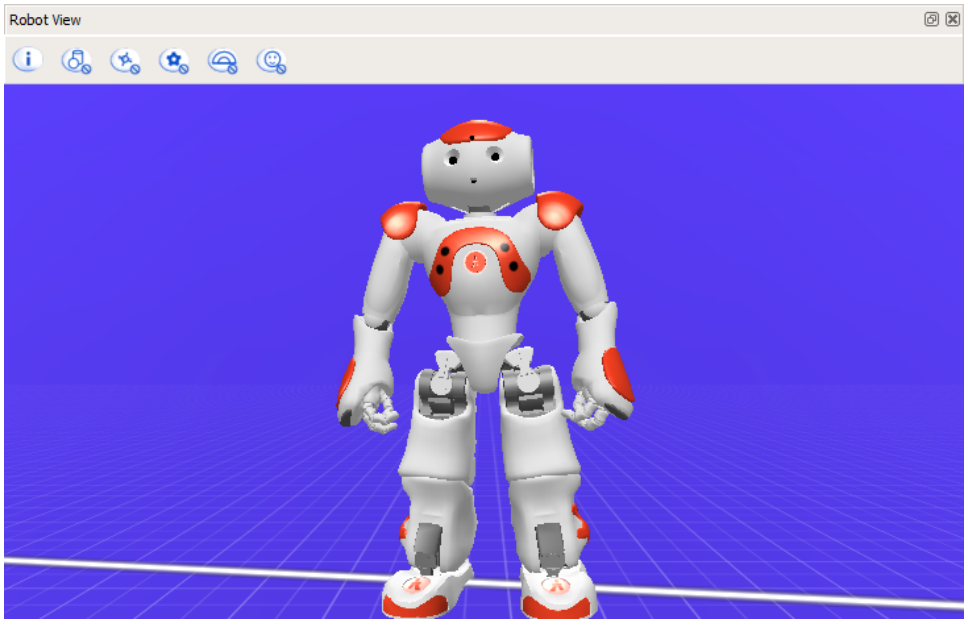


Abbildung 16: Der Robot View mit einem Roboter

Durch einen Klick auf die Schaltfläche „Robot Applications“ gelangt man zu den auf dem Roboter gespeicherten Applikationen. Zum einen findet sich hier stets die zuletzt auf NAO übertragene Applikation, zum anderen aber auch alle Applikationen, die vorinstalliert sind oder manuell auf den NAO übertragen wurden.
Ein selbstprogrammiertes Verhalten kann über den Button oben links langfristig auf dem NAO gespeichert werden. Über die Mülleimerschaltfläche können bereits installierte Applikationen entfernt werden. Mit Hilfe der dritten Schaltfläche des Robot Applications Menüs, der karierten Fahne, kann eine Applikation als Standardprogramm festgelegt werden, welches dann beim

Hochfahren des NAOs automatisch abgespielt wird. Die letzte Schaltfläche ist dafür dar alle laufenden Verhalten des NAOs zu beenden. Diese Schaltfläche ist besonders wichtig, da auf dem Roboter mehrere Verhalten gleichzeitig aktiv sein können und so sichergestellt werden kann, dass eine Applikation alleine ausgeführt wird, was zu weniger Komplikationen im Programmablauf führt.

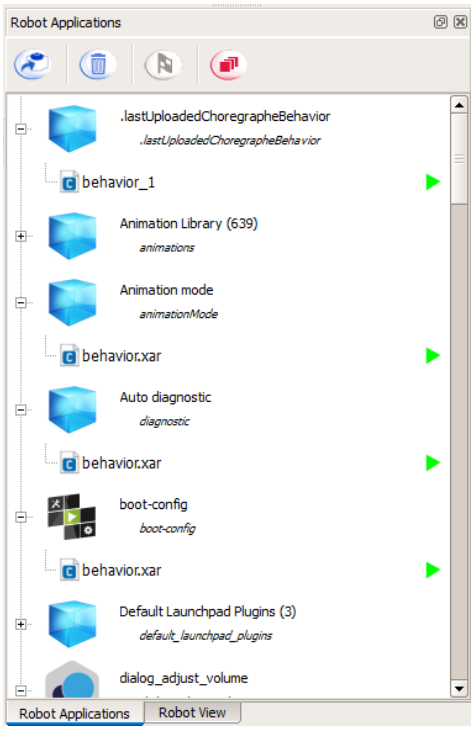


Abbildung 17: Robot Applications

Zudem sollte es eine Applikation mit dem Namen „Stay Relaxed“ auf dem Roboter geben, welche genutzt werden sollte, damit sich NAO beim Halten einer Pose nicht überhitzt. Wird NAO also in den nächsten Minuten nicht genutzt, sollte er über diese Applikation in eine entspannte Sitzposition gebracht werden.

NAO mit Choregraphe verbinden

NAO verbinden

Damit man sinnvoll mit NAO arbeiten kann, muss dieser mit Choregraphe verbunden werden. Man klickt hierfür auf den grünen Button in der Menüleiste, der in Abbildung 18 dargestellt ist:

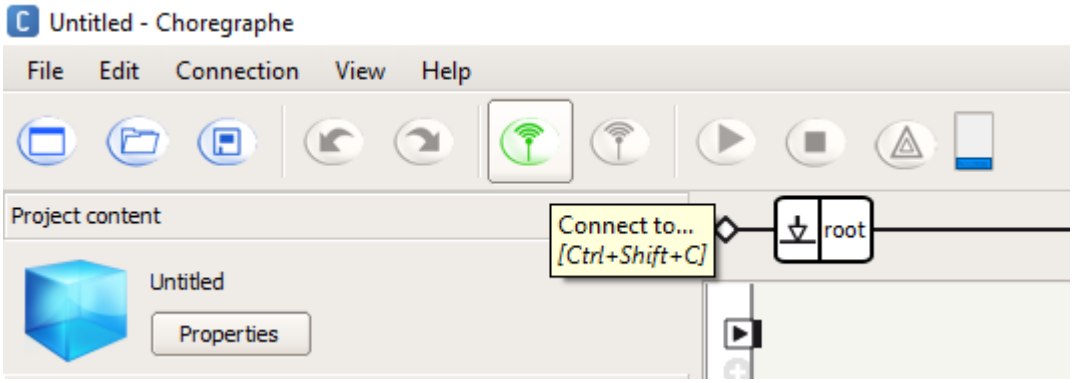


Abbildung 18: Der "Connect to" Button

Daraufhin öffnet sich ein Fenster, in welchem man seinen Roboter auswählen kann. Ist der eigene Roboter noch nicht in der Liste der Roboter zu finden, kann alternativ die IP-Adresse des Roboters angegeben werden.

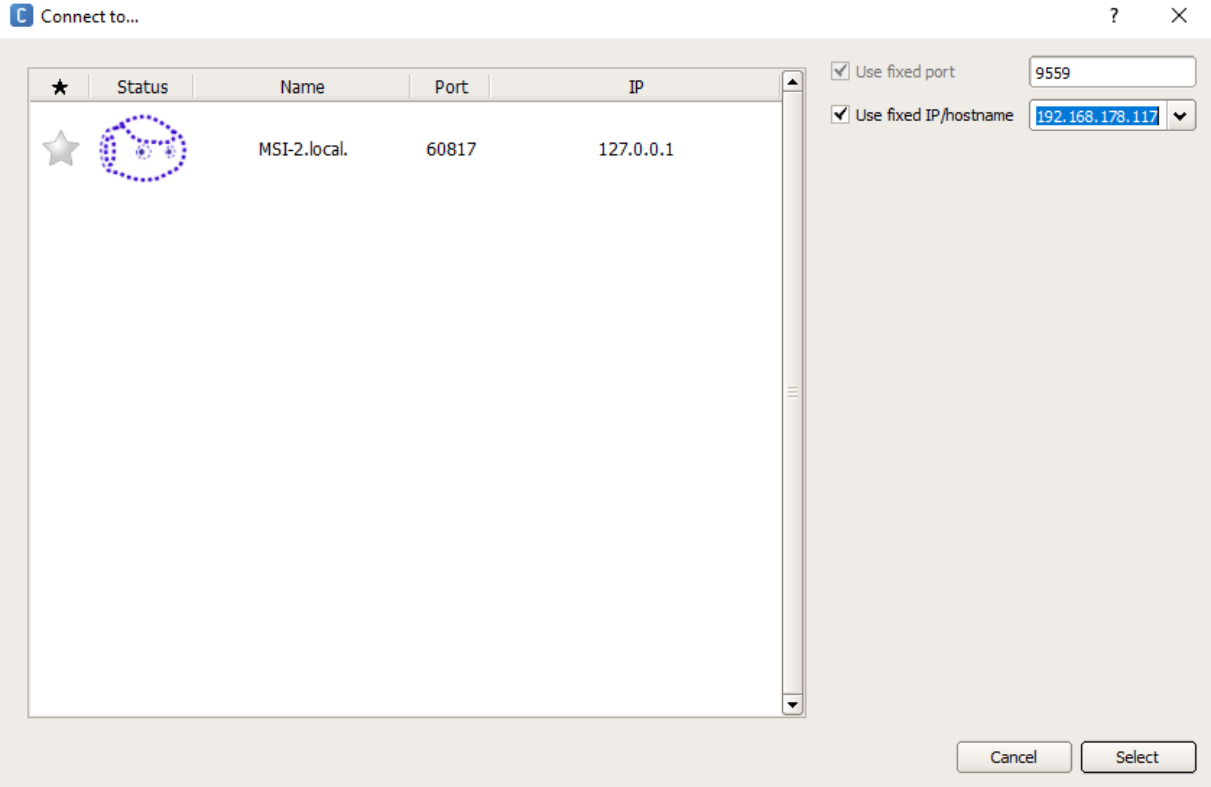


Abbildung 19: Das "Connect to" Fenster

Mit einem Klick auf „Select“ bestätigt man seine Auswahl. Choregraphe versucht nun sich mit dem ausgesuchten NAO zu verbinden. Den aktuellen Verbindungsstatus kann man dem Fensternamen von Choregraphe entnehmen.

Virtuellen Roboter verbinden

Anstatt des NAOs kann man auch einen virtuellen Roboter zum Testen seiner Programme nutzen. Dies ist besonders im Hinblick auf den Schuleinsatz sinnvoll und ist eine gute Möglichkeit den Roboter selbst zu schonen. Um die Verbindung herzustellen wählt man im Menü den Reiter „Connection“ aus und klickt danach auf „Connect to virtual robot“. Der Verbindungsstatus ändert sich dann sofort und im Robot View erscheint der virtuelle Roboter an dem die Programmierung getestet werden kann.

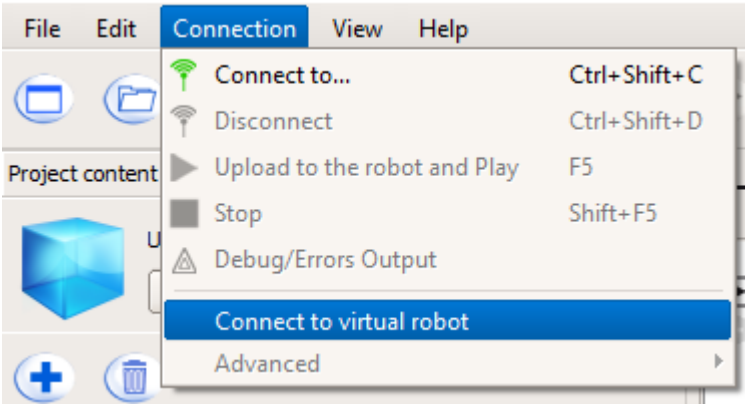


Abbildung 20: Mit virtuellem Roboter verbinden

Monitor

Monitor ist eine durch Choregraphe bereitgestellte Applikation, welche die Sensordaten des NAO ausgibt. Somit kann zum einen die Funktionalität des Roboters überprüft werden oder die eigene Programmierung einfacher nachvollzogen werden.
Monitor ist als eigenes Programm auf einem Computer zu finden, auf dem bereits Choregraphe installiert wurde. Nach dem Starten von Monitor wechselt man auf folgendes Fenster.

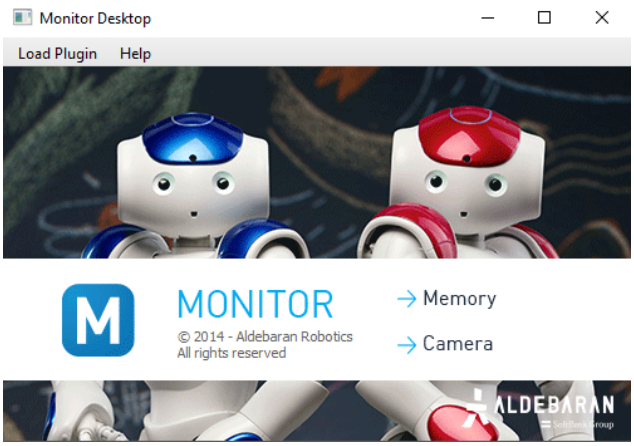


Abbildung 21: Monitor

Oben links kann man unter dem Reiter „Load Plugin“ verschiedene Plugins des Roboters laden, alternativ kann man auf den Speicher und die Kamera des NAOs direkt über die beiden Buttons „Memory“ und „Camera“ zugreifen.

Camera View

Wenn man sich mit dem Plugin „Camera View“ verbindet oder den „Camera“ Button auf der Startseite von Monitor auswählt, gelangt man zunächst zur Auswahl des eigenen Roboters bevor man die Kamerasicht des NAOs angezeigt bekommt. Hier können verschiedenste Einstellungen vorgenommen werden und auch Videos aufgezeichnet und Bilder gemacht werden.

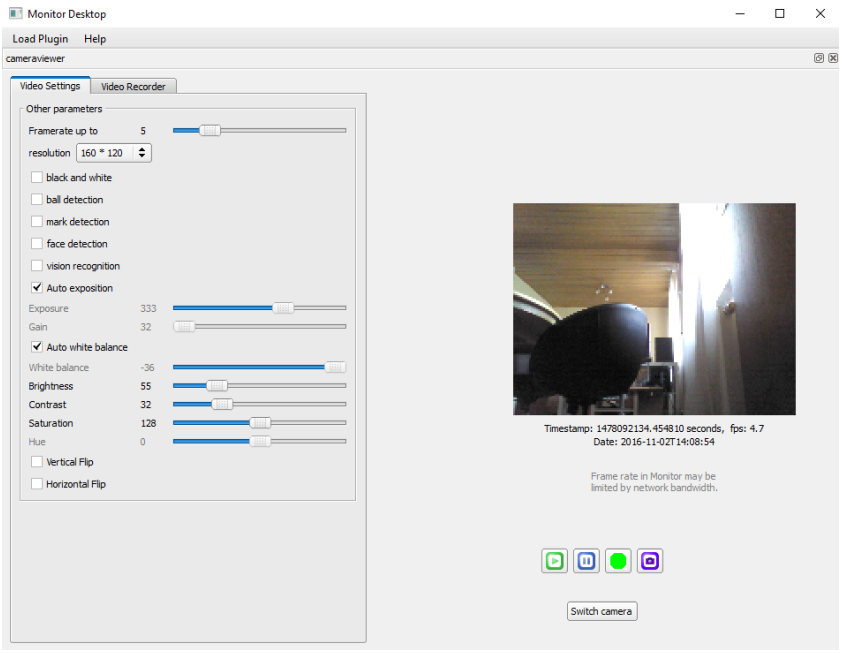


Abbildung 22: Die Kamera des NAOs

Besonders wichtig sind hierbei die vier Buttons unter dem Kamerabild. Hiermit kann die Übertragung des Bildes gestartet und pausiert werden. Zudem können hier direkt Video- und Fotoaufnahmen gestartet werden. Da des Weiteren die Funktionen „face detection“ und „vision recognition“ ausgewählt werden können, kann hier getestet werden, wie gut Objekte und Gesichter von NAO erkannt werden oder ob diese erneut erlernt werden sollten.

Memory Viewer

Mit Hilfe des Memory Viewers können sämtliche Bauteile, Sensoren und Informationen ausgelesen und geplottet werden. Hierfür wählt man zunächst im Startfenster von Monitor den Button „Memory“ aus und verbindet sich dann mit seinem Roboter. Im folgenden Fenster erstellt man sich zunächst eine eigene Konfigurationsdatei.

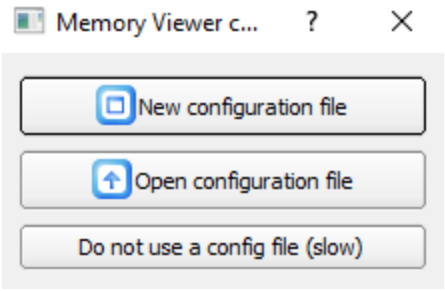


Abbildung 23: Konfigurationsfenster des Memory Views

Im darauffolgend erscheinenden Fenster wählt man dann die zu überwachenden Bauteile oder Verhaltensmuster aus und speichert diese gemeinsame Auswahl als eigene Konfigurationsdatei ab.

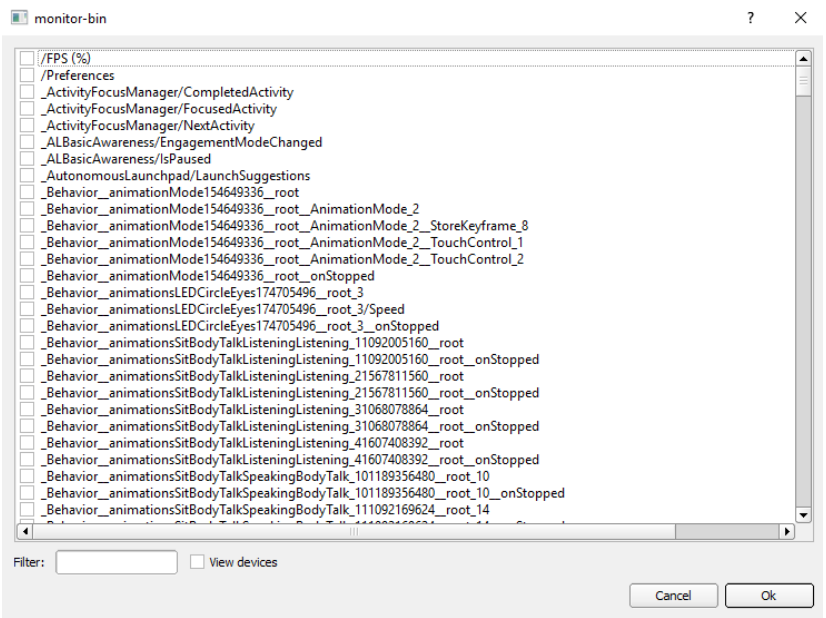


Abbildung 24: Auswahlfenster des Memory Views

Daraufhin öffnet sich der Memory Viewer und man kann sich in Echtzeit die aufgezeichneten Daten des NAO ausgeben und plotten lassen. Dadurch kann der Roboter sehr einfach ausgelesen werden. Bei falschen Reaktionen kann somit sehr leicht getestet werden, ob der Fehler an der eigenen Programmierung oder an den empfangenen Daten des NAOs liegt.

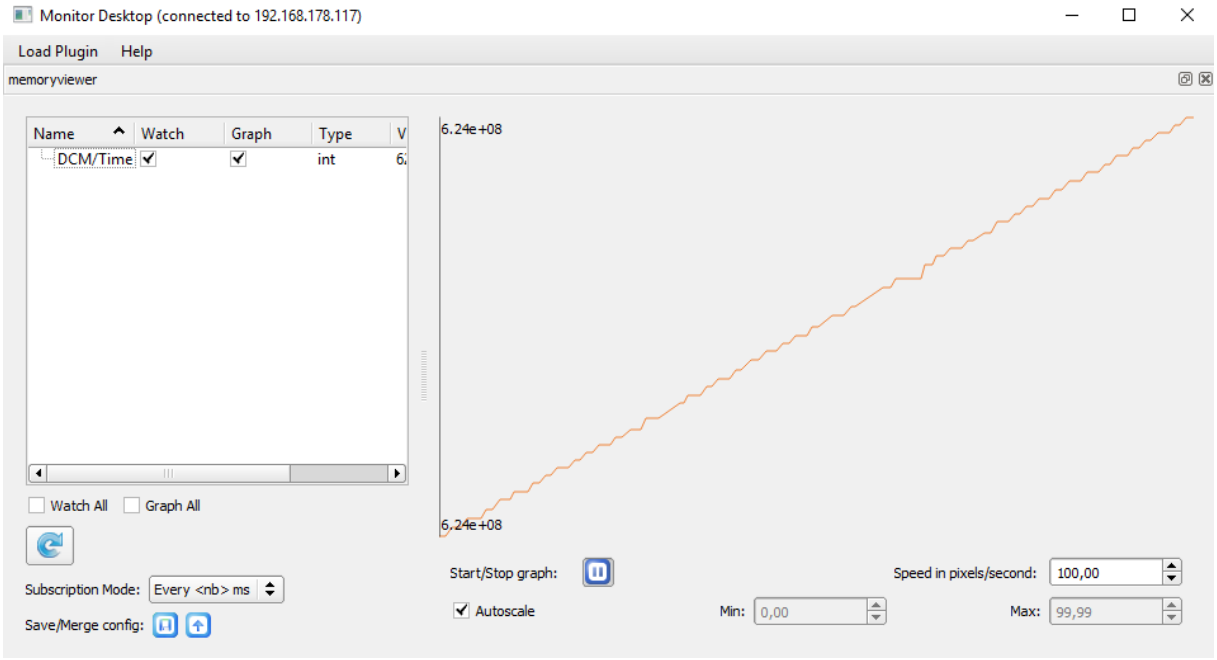


Abbildung 25: Memory View - hier wird die interne Zeit des NAOs geplottet

Das erste NAO-Programm

Zunächst soll ein einfaches Programm für den NAO erstellt werden, bei dem er lernt „Hallo Welt“ zu sagen. Zunächst benötigt man hierfür den „Set Language“-Baustein aus dem Ordner „Audio“ und dem Unterordner „Voice“. Der Baustein wird dann per Drag-and-Drop auf die Programmierfläche gezogen.

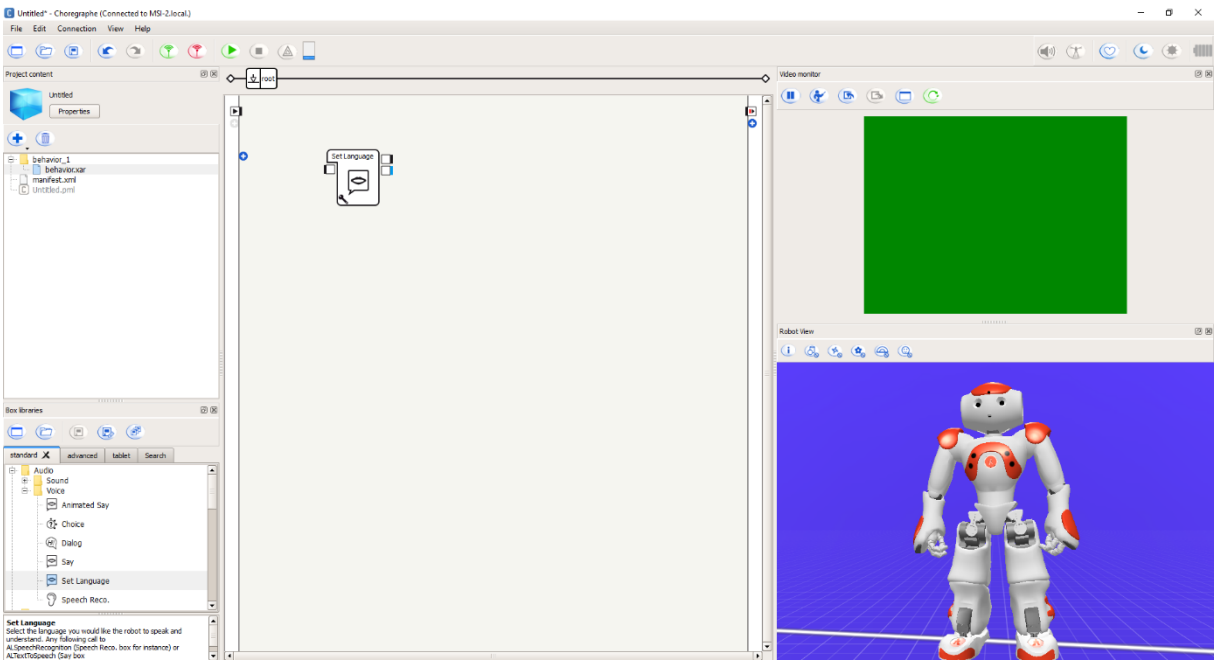


Abbildung 26: „Set Language“ liegt auf der Programmierfläche

Durch einen Klick auf den Schraubenschlüssel an der linken unteren Ecke des Bausteins öffnet sich das Eigenschaftenfenster des Bausteins, in dem die Sprache festgelegt werden kann zu der NAO wechselt.

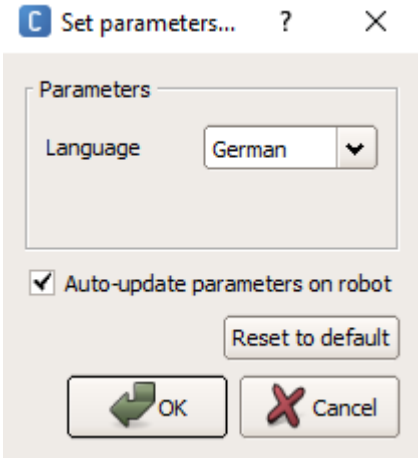


Abbildung 27: Eigenschaften von "Set Language"

Als nächstes wird der Baustein „Say“ benötigt. Dieser liegt wie „Set Language“ im Ordner „Audio“ und im Unterordner „Voice“. Auch dieser wird auf die Programmierfläche gezogen und im Anschluss wird auch hier auf den Schraubenschlüssel in der linken unteren Ecke des Bausteins geklickt. Hier kann die Stimmfarbe durch Anpassen der Stimmhöhe und der Geschwindigkeit verändert werden.

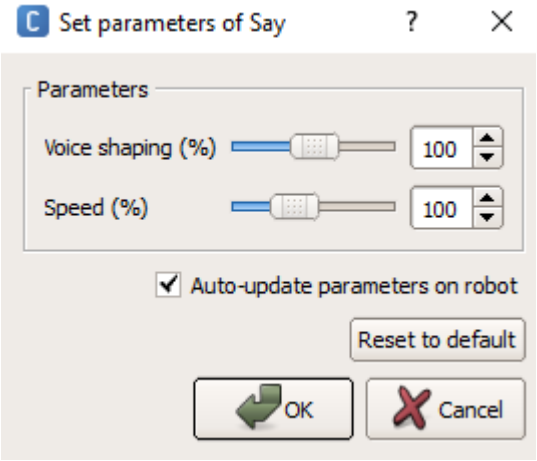


Abbildung 28: Eigenschaften des "Say"-Bausteins

Zunächst sollte man sich noch mit dem allgemeinen Aufbau eines Bausteins vertraut machen. Jeder Baustein hat auf der linken Seite Eingaben und auf der rechten Seite Ausgaben.

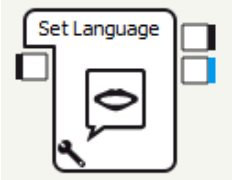


Abbildung 29: Der "Set Language"-Baustein

Den Datentyp der Ein- und Ausgabeparameter erfährt man, indem man mit der Maus über die jeweilige Ein- und Ausgabe fährt. Daraufhin wird eine kurze Beschreibung innerhalb eines gelben Kastens angezeigt. Der „Set Language“-Baustein hat beispielsweise eine Eingabe des Datentyps „Bang“ und eine Ausgabe des Datentyps „Bang“, falls der Baustein erfolgreich durchlaufen wurde, und eine Ausgabe des Datentyps „String“, falls das Umstellen der Sprache einen Fehler verursacht hat.

Der Datentyp „Bang“ ist innerhalb von Choregraphe als Signal definiert. Sobald ein Baustein fertig abgearbeitet wurde sendet er mit Hilfe einer „Bang“-Variable ein Signal an den nächsten Baustein, damit dieser mit der Ausführung beginnen kann. Weitere Datentypen in Choregraphe sind „Number“, „String“ und „Dynamic“. „Number“ hat stets gelbe Markierungen und speichert Zahlen, „String“ weist blaue Markierungen auf und beinhaltet Texte und „Dynamic“ ist in grau eingefärbt und kann alle bereits vorgestellten Datentypen, sowie mehrere Datentypen gleichzeitig speichern.
Durch einen Doppelklick auf einen Baustein gelangt man bei vielen Bausteinen zur inneren Struktur des Bausteins. Der Baustein „Say“ besteht beispielsweise aus einem Textbaustein und der eigentlichen Sprachausgabe.

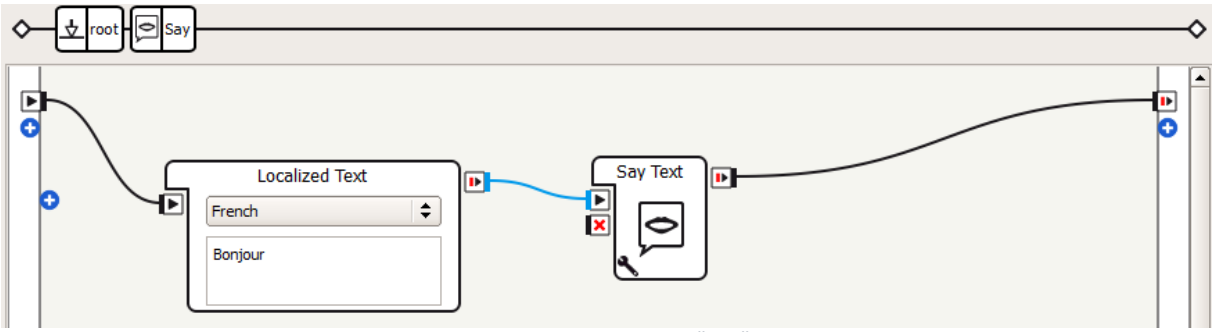


Abbildung 30: Innere Struktur des "Say"-Bausteins

In dem „Localized Text“-Feld kann die Sprache erneut auf Deutsch umgestellt werden und danach der Text zu „Hallo Welt“ abgeändert werden. Durch einen Klick auf „root“ oberhalb der Programmierfläche gelangt man zurück zu der vorherigen Ansicht, bei der man die Bausteine „Set Language“ und „Say“ sehen kann. Damit die Bausteine überhaupt aktiviert werden, müssen nun noch Verbindungen zwischen den Bausteinen gelegt werden. Hierfür klickt man auf das Startsymbol am linken Rand der Programmierfläche und zieht bei gedrückter linker Maustaste ein Verbindungskabel zum passenden Eingang des gewünschten Bausteins, hier also dem „Set Language“-Baustein. Danach verbindet man diesen Baustein mit dem „Say“-Baustein und den „Say“-Baustein mit dem Endsymbol am rechten oberen Rand der Programmierfläche.

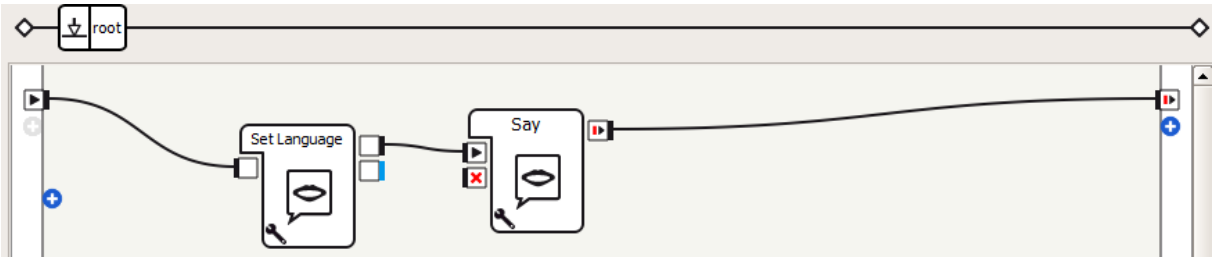


Abbildung 31: Der fertige Aufbau unseres ersten Programms

Nun muss das Programm nur noch über den grünen „Play“-Button im Menü gestartet werden und der Roboter sollte „Hallo Welt“ sagen. Das Programm kann alternativ auch durch das Drücken der „F5“-Taste gestartet werden. Über den „Stop“-Button oder die Tastenkombination „Shift + F5“ können laufende Programme unterbrochen werden.

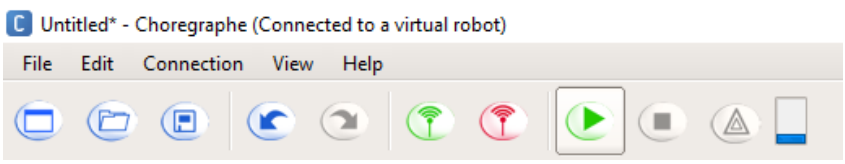


Abbildung 32: Der "Play"- und der "Stop"-Button

Die einzelnen Choregraphe-Bausteine

Im Funktionalitätenmenü von Choregraphe werden viele vorgefertigte Bausteine zur Programmierung des NAO angeboten. Im Folgenden soll auf die wichtigen Bausteine und deren Einsatzmöglichkeiten eingegangen werden. Die Reihenfolge der Bausteine orientiert sich an dem in Choregraphe implementierten Ordnersystem.

Audio

Play Sound

Baustein	**Kurzbeschreibung**
	Spielt eine Audiodatei des Formats .wav, .ogg oder .mp3 ab. Empfohlenes Dateiformat: .wav *Eingabe 1:* Starten der Wiedergabe (Datentyp: Bang) *Eingabe 2:* Stoppen der Wiedergabe (Datentyp: Bang) *Ausgabe:* Wiedergabe beendet (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Dateiname, Beginnposition, Lautstärke, Balance L/R, Wiederholen der Wiedergabe
Einsetzungsmöglichkeiten	
- Musikwiedergabe während einer Bewegung des NAOs - Wiedergabe aufgenommener Sprachdateien	

Record Sound

Baustein	**Kurzbeschreibung**
	Nimmt eine Audiodatei auf. Entweder wird die Aufnahme mit dem vorderen Kopfmikrofon durchgeführt und als .ogg Datei abgespeichert oder alternativ werden alle vier Mikrofone (eines in jede Richtung) genutzt und die Datei im .wav Format gespeichert. Falls die Datei nicht temporär gespeichert wird, findet man diese nach der Aufnahme im Ordner "*~/recordings/microphones/*". *Eingabe 1:* Starten der Aufnahme (Datentyp: Bang) *Eingabe 2:* Stoppen der Aufnahme (Datentyp: Bang) *Ausgabe:* Absoluter Speicherort der aufgenommenen Datei nach Beendigung der Aufnahme (Datentyp: String) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Dateiname, genutzte Mikrofone, temporäre Speicherung, Timeout der Aufnahme
Einsetzungsmöglichkeiten	
- Aufnahme der Umgebungsgeräusche - Aufnahme eines Dialogs	

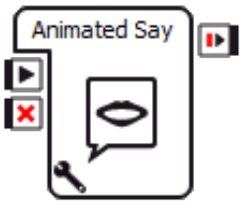
Set Speaker Volume

Baustein	Kurzbeschreibung
	Stellt die Lautstärke der Lautsprecher ein. *Eingabe:* Starten der Einstellung (Datentyp: Bang) *Ausgabe:* Einstellung ausgeführt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Lautstärke der Lautsprecher
Einsetzungsmöglichkeiten	
- Anpassung der Lautstärke vor der Wiedergabe einer Datei - Anpassung der Lautstärke vor einem Dialog	

Sound Location

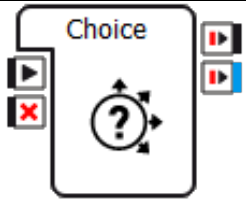
Baustein	Kurzbeschreibung
	Lokalisiert Geräusche über die vier Kopfmikrofone des NAOs. *Eingabe 1:* Starten der Lokalisierung (Datentyp: Bang) *Eingabe 2:* Stoppen der Lokalisierung (Datentyp: Bang) *Ausgabe 1:* Lokalisierung ausgeführt (Datentyp: Bang) *Ausgabe 2:* Ort der Geräuschquelle (Datentyp: Array bestehend aus zwei Variablen des Datentyps Number – der horizontale und vertikale Winkel der Geräschquelle zur Kopfposition) *Ausgabe 3:* Kopfposition (Datentyp: Array bestehend aus sechs Variablen des Datentyps Number – die ersten drei geben die x-, y- und z-Position des Kopfes an, die letzten drei geben die Orientierung des Kopfes zurück) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Threshold (Schwellwert) der Lokalisierungsgenauigkeit, Lautstärkensensitivität um Nebengeräusche zu ignorieren
Einsetzungsmöglichkeiten	
- Auf Geräuschquelle zulaufen - Positionierung des Roboters über ein Geräusch	
- Reaktion des Roboters auf laute Geräusche	

Animated Say

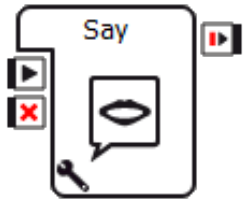
Baustein	Kurzbeschreibung
	Textausgabe mit gleichzeitiger Bewegungsanimation. *Eingabe 1:* Starten der Ausgabe (Datentyp: Bang) *Eingabe 2:* Stoppen der Ausgabe (Datentyp: Bang) *Ausgabe:* Textausgabe beendet (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Stimmfarbe durch Anpassen der Stimmhöhe und der Geschwindigkeit, Modus der Körpersprache *Einstellungsmöglichkeiten durch Doppelklick auf den Baustein:* Animation und Text – Eine Liste der möglichen Animationen ist unter

	http://doc.aldebaran.com/2-1/naoqi/audio/alanimatedspeech_advanced.html zu finden.
Einsetzungsmöglichkeiten	
- „Hallo“ sagen und dabei mit dem Arm winken - Menschlicheres Auftreten des NAOs durch passende Bewegungen beim Sprechen	

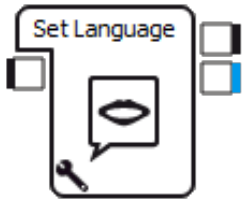
Choice

Baustein	**Kurzbeschreibung**
	Spracherkennung für zuvor eingegebene Antwortmöglichkeiten auf eine eigene Frage. *Eingabe 1:* Starten der Erkennung (Datentyp: Bang) *Eingabe 2:* Stoppen der Erkennung (Datentyp: Bang) *Ausgabe 1:* Spracherkennung beendet (Datentyp: Bang) Ausgabe 2: erkannte Antwortmöglichkeit (Datentyp: String) *Einstellungsmöglichkeiten durch Doppelklick auf den Baustein:* Sprache der Frage, eigene Frage, Sprache der Antworten, Antwortmöglichkeiten auf die eigene Frage
Einsetzungsmöglichkeiten	
- Verschiedene Reaktionen auf verschiedene Antworten - Überprüfen der Korrektheit einer Antwort	

Say

Baustein	**Kurzbeschreibung**
	Sprachausgabe eines Textes. *Eingabe 1:* Starten der Ausgabe (Datentyp: Bang) *Eingabe 2:* Stoppen der Ausgabe (Datentyp: Bang) *Ausgabe 1:* Sprachausgabe beendet (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Stimmfarbe durch Anpassen der Stimmhöhe und der Geschwindigkeit *Einstellungsmöglichkeiten durch Doppelklick auf den Baustein:* Sprache des Textes, Text
Einsetzungsmöglichkeiten	
- Dialog mit dem Roboter - Präsentation eines Themas durch NAO	

Set Language

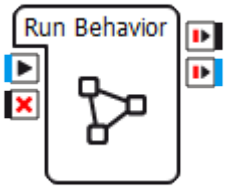
Baustein	Kurzbeschreibung
	Änderung der Sprache für die Sprachausgabe. *Eingabe 1:* Starten der Änderung (Datentyp: Bang) *Ausgabe 1:* Änderung erfolgreich beendet (Datentyp: Bang) *Ausgabe 2:* Fehlerausgabe, falls Änderung nicht erfolgreich (Datentyp: String) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Sprache
Einsetzungsmöglichkeiten	
- Globale Änderung der Sprache	

Speech Recognition

Baustein	Kurzbeschreibung
	Spracherkennung von zuvor eingegebenen Wörtern. *Eingabe 1:* Starten der Erkennung (Datentyp: Bang) *Eingabe 2:* Stoppen der Erkennung (Datentyp: Bang) *Ausgabe 1:* Spracherkennung beendet (Datentyp: Bang) *Ausgabe 2:* erkanntes Wort (Datentyp: String) *Ausgabe 3:* kein Wort erkannt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Liste der zu erkennenden Wörter, Schwellenwert der Spracherkennung, visuelles Feedback durch Augen-LEDs, Word Spotting – falls aktiviert erkennt NAO Wörter auch innerhalb von Sätzen
Einsetzungsmöglichkeiten	
- Starten eines Verhaltens auf Kommando - Reaktion auf bestimmte Wörter	

Behaviour

Run Behaviour

Baustein	Kurzbeschreibung
	Start der Ausführung eines auf dem NAO gespeicherten Verhaltens. Ausführung des Bausteins misslingt, falls Verhalten nicht auf NAO existiert. Vorsicht: Es kann zu Verzögerungen kommen, da das auszuführende Verhalten zunächst geladen werden muss. *Eingabe 1:* Name des zu startenden Verhaltens (Datentyp: String) *Eingabe 2:* Stoppen der Ausführung (Datentyp: Bang) *Ausgabe 1:* Ausführung erfolgreich beendet (Datentyp: Bang) *Ausgabe 2:* Fehlermeldung (Datentyp: String)

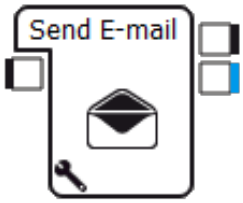
Einsetzungsmöglichkeiten
- Ausführung eines heruntergeladenen Verhaltens - Modularisierung einzelner Programme

Communication

Fetch E-mail

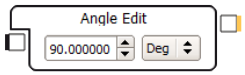
Baustein	Kurzbeschreibung
	Testen eines E-Mail-Zugangs auf neue Mails. *Eingabe 1:* Starten des Zugriffs auf den Mailserver (Datentyp: Bang) *Ausgabe 1:* E-Mail mit Autor, Text und Anhängen (Datentyp: Dynamic) *Ausgabe 2:* Fehlermeldung, falls etwas nicht funktioniert (Datentyp: String) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* E-Mailadresse, Passwort (Vorsicht! Über Roboter auslesbar), POP-Adresse, Portnummer des Zugriffs, SSL-Aktivierung/Deaktivierung, nur letzte/mehrere Mails
Einsetzungsmöglichkeiten	
- Auslesen eines Mailkontos - Vorlesen einer E-Mail	

Send E-mail

Baustein	Kurzbeschreibung
	Versenden einer E-Mail über einen E-Mail-Zugang. *Eingabe 1:* Versenden der E-Mail starten (Datentyp: Bang) *Ausgabe 1:* E-Mail erfolgreich gesendet (Datentyp: Bang) *Ausgabe 2:* Fehlermeldung, falls etwas nicht funktioniert (Datentyp: String) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Eigene E-Mailadresse, Passwort (Vorsicht! Über Roboter auslesbar), Empfänger-E-Mailadresse, Betreff der E-Mail, Text, Anhänge, SMTP-Adresse, Portnummer des Zugriffs
Einsetzungsmöglichkeiten	
- Versenden eines zuvor aufgenommenen Fotos an die Personen auf dem Foto - Versenden von Statusdaten zur Datenüberwachung auf Kommando	

Data Edit

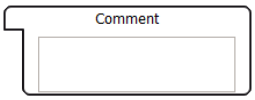
Angle Edit

Baustein	Kurzbeschreibung
	Änderung eines Winkels. *Eingabe 1:* Starten der Änderung (Datentyp: Bang) *Ausgabe 1:* Winkelzahl (Datentyp: Number) *Einstellungsmöglichkeiten:* Winkelzahl, Winkeltyp (Grad-/Bogenmaß)
Einsetzungsmöglichkeiten	
- Manuelle Einstellung eines Gelenkes	

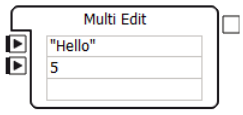
Color Edit

Baustein	Kurzbeschreibung
	Änderung/Einstellung einer Farbe. *Eingabe 1:* Starten der Änderung (Datentyp: Bang) *Ausgabe 1:* Winkelzahl (Datentyp: Array bestehend aus drei Variablen des Datentyps Number – den RGB-Werten der eingestellten Farbe) *Einstellungsmöglichkeiten:* Farbe
Einsetzungsmöglichkeiten	
- Ändern der LED-Augenfarbe des NAO	

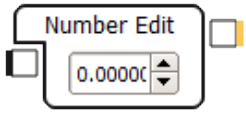
Comment

Baustein	Kurzbeschreibung
	Kommentarfunktion innerhalb der Programmierumgebung. *Eingabe 1:* Anhängen des Kommentars an beliebiger Stelle *Einstellungsmöglichkeiten:* Kommentartext
Einsetzungsmöglichkeiten	
- Auskommentieren der eigenen Programmierung, um Verständlichkeit für andere User zu erhöhen	

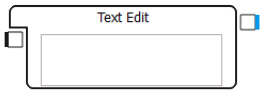
Multi Edit

Baustein	Kurzbeschreibung
	Mehrere Änderungen in einem Baustein. *Eingabe 1:* Wert 1 soll übernommen werden (Datentyp: Bang) *Eingabe 2:* Wert 2 soll übernommen werden (Datentyp: Bang) … Ausgabe 1: neuer Wert und Nummer des Eingangs (Datentyp: Dynamic) *Einstellungsmöglichkeiten:* Eingabe der verschiedenen Werte
Einsetzungsmöglichkeiten	
- Anpassung einer Zahl je nachdem welches Wort erkannt wurde - Einsparung von mehreren anderen Edit-Bausteinen	

Number Edit

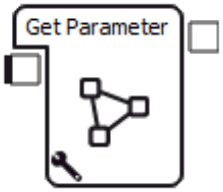
Baustein	Kurzbeschreibung
	Änderung/Erstellen einer Zahl. *Eingabe 1:* Starten der Änderung (Datentyp: Bang) *Ausgabe 1:* neue Zahl (Datentyp: Number) *Einstellungsmöglichkeiten:* Zahl mit Nachkommastellen
Einsetzungsmöglichkeiten	
- Konstantendefinition	

Text Edit

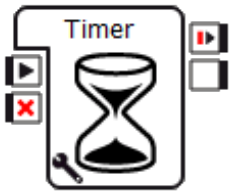
Baustein	Kurzbeschreibung
	Änderung/Erstellen eines Textes. *Eingabe 1:* Starten der Änderung (Datentyp: Bang) *Ausgabe 1:* Text (Datentyp: String) *Einstellungsmöglichkeiten:* Text
Einsetzungsmöglichkeiten	
- Definition eines Textes, den NAO im Anschluss sagen soll	

Flow Control

Get Parameter

Baustein	Kurzbeschreibung
	Gibt bestimmten Wert der vorherigen Box zurück. *Eingabe 1:* Holen des bestimmten Wertes (Datentyp: Bang) *Ausgabe 1:* ausgelesener Wert (Datentyp: Dynamic, je nach Wert) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Name des auszulesenden Parameters
Einsetzungsmöglichkeiten	
- Auslesen von Zwischenberechnungen, ohne vorherigen Baustein zu verändern	

Timer

Baustein	Kurzbeschreibung
	Sendet „Bang“-Signal periodisch weiter. *Eingabe 1:* Starten des Timers (Datentyp: Bang) *Eingabe 2:* Stoppen des Timers (Datentyp: Bang) *Ausgabe 1:* Timer beendet (Datentyp: Bang) *Ausgabe 2:* Zeit abgelaufen (Datentyp: Bang, wiederholt sich periodisch) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Zeitperiode
Einsetzungsmöglichkeiten	
- Ansage der Uhrzeit alle fünf Minuten	

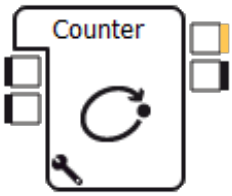
Wait

Baustein	Kurzbeschreibung
	Wartet eine bestimmte Zeit bevor eine Ausgabe erfolgt. Bei wiederholter Eingabe wird die interne Zeit zurückgesetzt. *Eingabe 1:* Starten der Wartezeit (Datentyp: Bang) *Eingabe 2:* Stoppen der Wartezeit (Datentyp: Bang) *Ausgabe 1:* Zeit abgelaufen (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Zeitintervall, Aktivität bei gestopptem Baustein
Einsetzungsmöglichkeiten	
- Warte darauf, dass 5 Sekunden nichts geschieht	

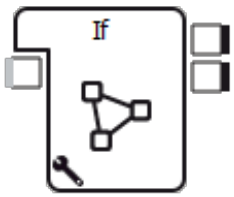
Delay

Baustein	Kurzbeschreibung
	Wartet eine bestimmte Zeit bevor eine Ausgabe erfolgt. Bei wiederholter Eingabe wird die interne Zeit aufsummiert. *Eingabe 1:* Starten/Erhöhen der Wartezeit (Datentyp: Bang) *Eingabe 2:* Stoppen der Wartezeit (Datentyp: Bang) *Ausgabe 1:* Zeit abgelaufen (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Zeitintervall, Aktivität bei gestopptem Baustein
Einsetzungsmöglichkeiten	
- Verzögern des Ablaufs je nach Anzahl der Eingaben	

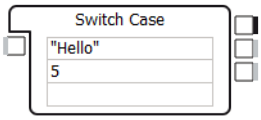
Counter

Baustein	Kurzbeschreibung
	Repräsentiert eine Zahl, die jedes Mal um eine konstante Zahl erhöht wird, falls eine Eingabe festgestellt werden kann. Ist der Maximalwert erreicht, wird der Counter wieder zurückgesetzt. *Eingabe 1:* Erhöhen des Counters (Datentyp: Bang) *Eingabe 2:* Zurücksetzen des Counters (Datentyp: Bang) *Ausgabe 1:* aktueller Zahlenwert (Datentyp: Number) *Ausgabe 2:* Counter wurde zurückgesetzt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Initialwert, Schrittweite, Maximalwert
Einsetzungsmöglichkeiten	
- Zählen eines bestimmten Verhaltens	

If

Baustein	Kurzbeschreibung
	Vergleicht einen Wert mit einem vordefinierten Wert. Wenn der Vergleich korrekt ist, dann wird Ausgabe 1 aktiviert, ansonsten Ausgabe 2. *Eingabe 1:* Vergleich der Werte starten (Datentyp: Dynamic) *Ausgabe 1:* Bedingung erfüllt (Datentyp: Bang) *Ausgabe 2:* Bedingung nicht erfüllt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Operator der Bedingung (<,<=,=,>,>=,≠), Vergleichswert
Einsetzungsmöglichkeiten	
- Verzweigung innerhalb des Roboterverhaltens - Unterschiedliche Reaktion des Roboters je nach Eingabe	

Switch Case

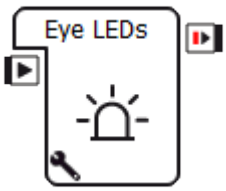
Baustein	Kurzbeschreibung
	Testet einen Wert gegen eine Liste vordefinierter Werte. Wenn der Wert mit einem der Werte in der Liste übereinstimmt wird dessen Ausgabe aktiviert, ansonsten wird der Defaultwert weitergegeben. *Eingabe 1:* Starten des Wertetests (Datentyp: Dynamic) *Ausgabe 1:* Keine Übereinstimmung gefunden (Datentyp: Bang) *Ausgabe 2:* Eingabe stimmt mit Wert an Listenplatz 1 überein (Datentyp: Dynamic, je nach gespeichertem Wert) *Ausgabe 3:* Eingabe stimmt mit Wert an Listenplatz 2 überein (Datentyp: Dynamic, je nach gespeichertem Wert) Weitere Listenelemente können über einen Rechtsklick auf den Baustein erstellt werden. Auf diese Weise können auch Werte aus der Liste entfernt werden.
Einsetzungsmöglichkeiten	
- Verzweigung innerhalb des Roboterverhaltens - Unterschiedliche Reaktion des Roboters je nach Eingabe	

Wait For Signals

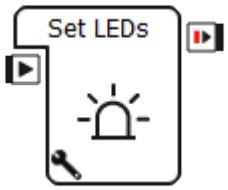
Baustein	Kurzbeschreibung
	Wartet auf alle Eingangssignale, bevor eine Ausgabe generiert wird. *Eingabe 1:* erstes Signal, auf das gewartet wird (Datentyp: Bang) *Eingabe 2:* zweites Signal, auf das gewartet wird (Datentyp: Bang) *Ausgabe 1:* alle Signale wurden erhalten (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Zusammenfassen mehrerer Programmabläufe (beispielsweise könnte NAO sprechen und sich parallel dazu bewegen – mit diesem Baustein könnte man auf die Beendigung beider Abläufe warten)	

LEDs

Eye LEDs

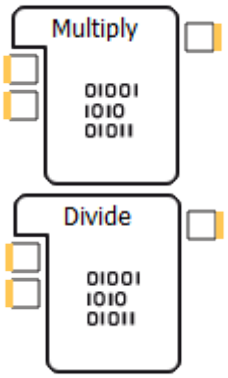
Baustein	Kurzbeschreibung
	Einstellung der Farbe der Augen-LEDs. *Eingabe 1:* Starten des Farbwechsels (Datentyp: Bang) *Ausgabe 1:* Farbwechsel abgeschlossen (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Seite des Roboters, Dauer des Farbwechsels *Einstellungsmöglichkeiten durch Doppelklick auf den Baustein:* Farbe, auf welche gewechselt werden soll
Einsetzungsmöglichkeiten	
- Farbwechsel der Augen, falls NAO auf eine Eingabe wartet	

Set LEDS

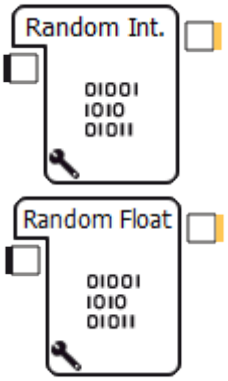
Baustein	Kurzbeschreibung
	Einstellung der Leuchtkraft der LEDs. *Eingabe 1:* Starten des Intensitätswechsels (Datentyp: Bang) *Ausgabe 1:* Intensitätswechsel abgeschlossen (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* LED-Gruppe, Intensität in Prozent, Dauer des Intensitätswechsels
Einsetzungsmöglichkeiten	
- Abdunklung einzelner LEDs - Blinkeffekt der LEDs durch abwechselnde Erniedrigung und Erhöhung der Intensität	

Math

Multiply / Devide

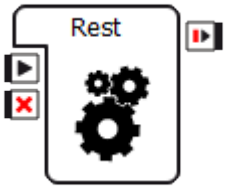
Baustein	Kurzbeschreibung
	Multiplikation / Division zweier Zahlen. *Eingabe 1:* erster Faktor / Dividend (Datentyp: Number) *Eingabe 2:* zweiter Faktor / Divisor (Datentyp: Number) *Ausgabe 1:* Produkt / Quotient (Datentyp: Number)
Einsetzungsmöglichkeiten	
- Berechnung neuer Gelenkpositionen - Berechnung des prozentualen Anteils eines Wertes bezüglich eines anderen Wertes	

Random Int. / Float

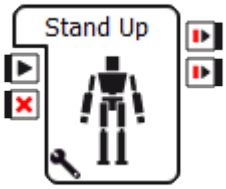
Baustein	Kurzbeschreibung
	Erzeugung einer zufälligen ganzen / ganzrationalen Zahl. *Eingabe 1:* Starten der Erzeugung (Datentyp: Bang) *Ausgabe 1:* erzeugte Zahl (Datentyp: Number) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Minimalwert, Maximalwert (beide Werte können bei der Generierung der Zufallszahl getroffen werden)
Einsetzungsmöglichkeiten	
- Zufällige Positionierung des Arms des NAOs - Simulation eines Würfelwurfes durch NAO	

Motion

Rest

Baustein	**Kurzbeschreibung**
	Bewegt NAO in eine Ruheposition und schaltet danach alle Motoren ab. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung erfolgreich abgeschlossen (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Erreichen einer schonenden Position, in der keine Motoren überhitzen	

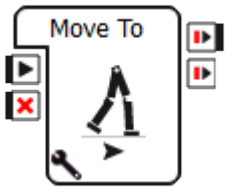
Stand Up

Baustein	**Kurzbeschreibung**
	Bewegt NAO in eine stehende Position. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung erfolgreich abgeschlossen (Datentyp: Bang) *Ausgabe 2:* Bewegung missglückt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Anzahl der Versuche die Bewegung auszuführen
Einsetzungsmöglichkeiten	
- Erreichen der Startposition für die meisten Programme	

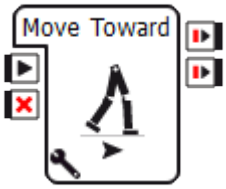
SitDown

Baustein	**Kurzbeschreibung**
	Bewegt NAO in eine sitzende Position. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung erfolgreich abgeschlossen (Datentyp: Bang) *Ausgabe 2:* Bewegung missglückt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Erreichen der Endposition der meisten Programme	

Move To

Baustein	Kurzbeschreibung
	Bewegt NAO zu einer vorher spezifizierten Position. Vorsicht: Die Fehlertoleranz dieses Bausteins ist sehr gering und kann durch einen Doppelklick auf diesen abgeändert werden. Dabei sollten die Werte positionErrorThresholdPos und positionErrorThresholdAng auf ungefähr 0.18 erhöht werden. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung erfolgreich an der gewünschten Position abgeschlossen (Datentyp: Bang) *Ausgabe 2:* Bewegung missglückt (Datentyp: Array bestehend aus drei Variablen des Datentyps Number – der Abstand in x- und y-Richtung von der gewünschten Position, sowie der Winkel zu dieser aus Sicht des Roboters) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Spezifikation der Position durch die Entfernung in x-Richtung, y-Richtung und den Winkel aus Sicht des Roboters; Aktivierung/Deaktivierung der Armbewegungen
Einsetzungsmöglichkeiten	
- Positionierung des Roboters	

MoveToward

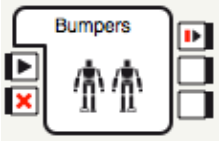
Baustein	Kurzbeschreibung
	Bewegt NAO in Richtung einer vorher spezifizierten Position. Dabei beendet dieser die Bewegung nicht, falls er die Position erreicht haben sollte. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung beendet (Datentyp: Bang) *Ausgabe 2:* Bewegung missglückt (Datentyp: Bang) *Einstellungsmöglichkeiten mit dem Schraubenschlüssel:* Spezifikation der Position durch die Entfernung in x-Richtung, y-Richtung und den Winkel aus Sicht des Roboters; Aktivierung/Deaktivierung der Armbewegungen
Einsetzungsmöglichkeiten	
- Dauerhafte Bewegung in eine vordefinierte Richtung	

Obstacte Avoidance

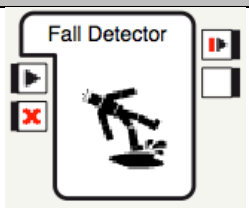
Baustein	Kurzbeschreibung
	Bewegt NAO stets geradeaus. Sobald ein Hindernis auftaucht, dreht sich NAO nach rechts und läuft weiter geradeaus. *Eingabe 1:* Starten der Bewegung (Datentyp: Bang) *Eingabe 2:* Abbruch der Bewegung (Datentyp: Bang) *Ausgabe 1:* Bewegung beendet (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Finden eines Ausgangs in einem Labyrinth	

Sensing

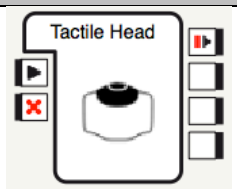
Bumpers

Baustein	Kurzbeschreibung
	Wartet auf das Drücken des rechten oder linken Fußsensors. *Eingabe 1:* Starten des Wartens auf Stimulation der Fußsensoren (Datentyp: Bang) *Eingabe 2:* Abbruch des Wartens auf Stimulation der Fußsensoren (Datentyp: Bang) *Ausgabe 1:* Warten wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Linker Fußsensor wurde gedrückt (Datentyp: Bang) *Ausgabe 3:* Rechter Fußsensor wurde gedrückt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Notausknopf	

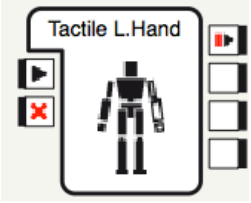
Fall Detector

Baustein	Kurzbeschreibung
	Bemerkt, wenn NAO hingefallen ist und der „fall manager process“ aktiviert wurde. *Eingabe 1:* Startet das Warten auf einen Sturz von NAO (Datentyp: Bang) *Eingabe 2:* Abbruch des Wartens auf einen Sturz von NAO (Datentyp: Bang) *Ausgabe 1:* Warten wurde abgebrochen (Datentyp: Bang) *Ausgabe 2*: Sturzereignis wurde wahrgenommen (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Bei Sturzgefahr, z.B. während eines Tanzes	

Tactile Head

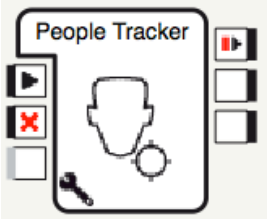
Baustein	Kurzbeschreibung
	Wartet auf das Drücken eines Kopfsensors. *Eingabe 1:* Starten des Wartens auf Stimulation der Kopfsensoren (Datentyp: Bang) *Eingabe 2:* Abbruch des Wartens auf Stimulation der Kopfsensoren (Datentyp: Bang) *Ausgabe 1:* Warten wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Vorderer Kopfsensor wurde gedrückt (Datentyp: Bang) *Ausgabe 3:* Mittlerer Kopfsensor wurde gedrückt (Datentyp: Bang) *Ausgabe 4:* Hinterer Kopfsensor wurde gedrückt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Verschiedene Wahlmöglichkeiten für die weitere Abfolge	

Tactile L. Hand

Baustein	Kurzbeschreibung
	Wartet auf das Drücken eines linken Handsensors. *Eingabe 1:* Starten des Wartens auf Stimulation eines der linken Handsensoren (Datentyp: Bang) *Eingabe 2:* Abbruch des Wartens auf Stimulation der Handsensoren (Datentyp: Bang) *Ausgabe 1:* Warten wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Linker Sensor der linken Hand wurde gedrückt (Datentyp: Bang) *Ausgabe 3:* Handrückensensor der linken Hand wurde gedrückt (Datentyp: Bang) *Ausgabe 4:* Rechter Sensor der linken Hand wurde gedrückt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Verschiedene Wahlmöglichkeiten für die weitere Abfolge	

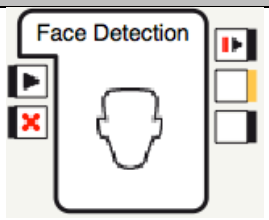
Trackers

People Tracker

Baustein	Kurzbeschreibung
	NAO verfolgt ihm genannte/bekannte Personen. *Eingabe 1:* Starten der Verfolgung (Datentyp: Bang) *Eingabe 2:* Abbruch der Verfolgung (Datentyp: Bang) *Eingabe 3:* Liste mit den Personen-IDs, die NAO verfolgen soll (Datentyp: Dynamic) *Ausgabe 1:* Verfolgung beendet (Datentyp: Bang) *Ausgabe 2:* Verfolgung missglückt (Datentyp: Bang) *Ausgabe 3:* Ziel erreicht (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Zur Verfolgung einer Person	

Vision

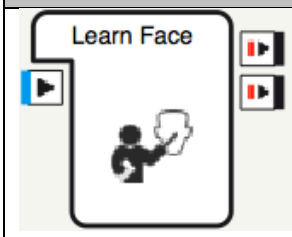
Face Detection

Baustein	Kurzbeschreibung
	Nimmt Menschengesichter wahr, auch die, die noch nicht in seiner Datenbank gespeichert sind, und gibt die Anzahl der wahrgenommenen Gesichter aus. *Eingabe 1:* Startet die Gesichtserkennung (Datentyp: Bang) *Eingabe 2:* Abbruch der Gesichtserkennung (Datentyp: Bang) *Ausgabe 1:* Gesichtserkennung wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Ausgabe der Anzahl der wahrgenommenen Gesichter (Datentyp: Number) *Ausgabe 3:* Keine Gesicht wurde wahrgenommen (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Einen Menschen wahrzunehmen	

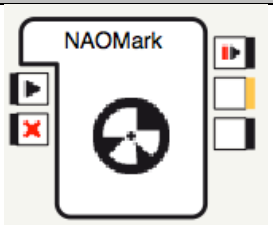
Face Recognition

Baustein	Kurzbeschreibung
	Nimmt Menschengesichter wahr und erkennt die Gesichter wieder, die in seiner Datenbank mittels der „Learn Face Box“ bereits gespeichert sind. *Eingabe 1:* Startet die Gesichtserkennung (Datentyp: Bang) *Eingabe 2:* Abbruch der Gesichtserkennung (Datentyp: Bang) *Ausgabe 1:* Gesichtserkennung wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Ausgabe des Namens der wiedererkannten Person (Datentyp: String)
Einsetzungsmöglichkeiten	
- Einen Menschen wiederzuerkennen und seinen Namen zu kennen	

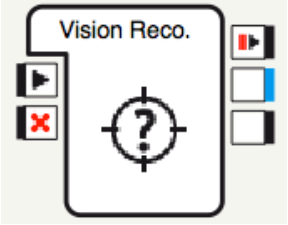
Learn Face

Baustein	Kurzbeschreibung
	Bringt NAO ein neues Gesicht bei, das in seiner Datenbank gespeichert werden soll. *Eingabe 1:* Name der Person, dessen Gesicht NAO sich merken soll (Datentyp: String) *Ausgabe 1:* Gesichtserkennung wurde erfolgreich beendet (Datentyp: Bang) *Ausgabe 2:* Gesichtserkennung ist missglückt (Datentyp: String)
Einsetzungsmöglichkeiten	
- NAO einen Namen einer Person beibringen.	

NAOMark

Baustein	Kurzbeschreibung
	Erkennt eine NAOMark und gibt deren ID aus. *Eingabe 1:* Startet die Erkennung (Datentyp: Bang) *Eingabe 2:* Abbruch der Erkennung (Datentyp: Bang) *Ausgabe 1:* Erkennung wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Ausgabe der ID der erkannten NAOMark (Datentyp: Number) *Ausgabe 3:* Keine NAOMark wurde erkannt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Durch NAOMarks können verschiedene Programmabläufe gestartet werden und im Programm gesprungen werden.	

Vision Recognition

Baustein	Kurzbeschreibung
	Erkennt verschiedene Objekte, Bilder oder Orte, die ihm zuvor beigebracht wurden. *Eingabe 1:* Startet die Erkennung (Datentyp: Bang) *Eingabe 2:* Abbruch der Erkennung (Datentyp: Bang) *Ausgabe 1:* Erkennung wurde abgebrochen (Datentyp: Bang) *Ausgabe 2:* Ausgabe der Bezeichnung des erkannten Objekts (Datentyp: String) *Ausgabe 3:* Kein Objekt wurde erkannt (Datentyp: Bang)
Einsetzungsmöglichkeiten	
- Erkennung eines Ortes oder eines bekannten Schauspielers	

Lern-Modus für Vision Recognition

Damit NAO verschiedene Objekte, Bilder oder Orte wiedererkennen kann, müssen diese ihm zuvor gezeigt und beigebracht worden sein. Dieser „Lernprozess“ wird über den Videomonitor im oberen rechten Teilfenster in Choregraphe gesteuert.
Zunächst muss NAO mit Choregraphe verbunden werden; erst dann wird im Videomonitor das Bild gezeigt, welches NAO aktuell über seine Kamera aufzeichnet.

Abbildung 33: Leiste beim Videomonitor

Nun kann NAO bspw. ein neues Bild in vier Schritten beigebracht werden:

1. Als erstes erstellen wir eine neue Datenbank: Hierfür muss der zweite Button von rechts „New Vision Recognition Database“ angeklickt werden und das neu geöffnete Fenster bestätigt werden.
2. Im zweiten Schritt muss der Learn-Button angeklickt werden. Nach dem Anklicken des Buttons hat man 4 Sekunden Zeit, um das gewünschte Bild in die NAO-Kamera zu halten.
3. Nun wird das Bild im Videomonitor angezeigt. Im dritten Schritt wählt man die Fläche des Motivs aus, das NAO wiedererkennen soll. Hierfür klickt man in das Bild und legt dadurch die

Außengrenzen des Motivs fest. Sobald alle vier Außengrenzen markiert sind, öffnet sich eine neue Box, in der man den Namen des Bildes sowie die Ansicht eingeben muss.

4. Im letzten Schritt muss die neu erstellte Datenbank auf NAO selbst gespeichert werden. Dafür muss der rechte Button angeklickt werden.

Im Anschluss kann über den „Vision Recognition“-Baustein getestet werden, ob NAO das Bild wiedererkennt. Zu einer bildlichen Veranschaulichung des Learn-Modus empfiehlt sich auch folgendes YouTube-Video: https://www.youtube.com/watch?v=BigrGEvFTJs.

Beispiel einer Choregraphe-Anwendung

Im letzten Kapitel wurden die einzelnen Programmierbausteine von Choregraphe vorgestellt. Um die Programmierung des NAOs verständlicher zu machen, wird im Folgenden ein Programm im Detail betrachtet. Der grobe Aufbau einer Choregraphe-Anwendung wurde bereits im Kapitel „Das erste NAO-Programm“ präsentiert, weshalb nun eine komplexere Programmierung besprochen wird.

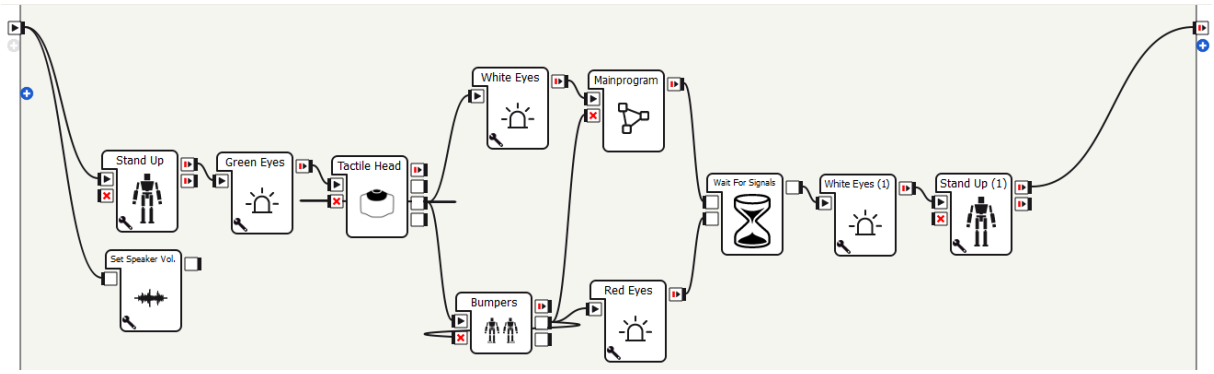


Abbildung 34: Beispielprogramm

Das gezeigte Programm soll als übergeordnete Steuerung eines eigentlichen Programmes dienen, damit dieses im Notfall abgeschaltet werden kann und zudem einen geordneten Ablauf bereitstellen.
Zu Beginn steht NAO durch den Befehl „Stand Up“ auf. Parallel dazu wird die Lautstärke seiner Lautsprecher auf „100“ gesetzt. Interessant ist hierbei, dass NAO Befehle die mit demselben Ausgang verbunden sind gleichzeitig ausführt, sobald der Ausgang aktiviert wird.
Sobald NAO aufgestanden ist, wechselt er die Farbe der LEDs in seinen Augen auf grün. Danach wartet NAO darauf, dass der mittlere Button auf seinem Kopf gedrückt wird. Der vordere und hintere Button haben hierbei keine Funktion, da lediglich am mittleren der drei Ausgänge Verbindungslinien gezogen wurden. Wird der mittlere Button gedrückt werden zwei neue Bausteine aktiviert und der Kopfsensorbaustein beendet. Dies ist besonders wichtig, da er ansonsten weiterhin aktiv wäre und somit das restliche Programm immer dann gestartet werden würde, sobald man den mittleren Kopfbutton drückt. Sensoren sollte man also stets beenden, wenn diese nicht mehr benötigt werden.
Nach Beendigung der Kopfsensoren werden zwei parallele Programmteile ausgeführt. Zum einen wechselt die Augenfarbe des NAOs zurück zu weiß und das eigentliche Programm, hier „Mainprogram“, wird ausgeführt. Hierbei handelt es sich um einen selbsterstellten Baustein, worüber im nächsten Kapitel genaueres erklärt wird.
Zum anderen werden die Bumpersensoren an NAOs Füßen aktiviert. Hier soll der linke Bumper als Notaus-Knopf fungieren. Sobald er berührt wird, wechselt die Augenfarbe des NAOs zu rot und das „Mainprogram“ wird beendet. Zusätzlich wird der eigene Sensorbaustein beendet.
Als nächster Baustein wurde ein „Wait For Signals“ eingesetzt. Dieser Baustein wartet auf alle eingehenden Signale. In diesem Fall wartet NAO also darauf, dass seine Augenfarbe auf rot gesetzt wird und das eigentliche Programm beendet ist. Das restliche Programm wird also nur dann ausgeführt, wenn das Programm unterbrochen wurde, da ansonsten die Augenfarbe nicht abgeändert wird. Falls beide Signale erhalten wurden wechselt NAO seine Augenfarbe ein letztes Mal

zurück auf weiß und steht wieder auf, um die Ausführung in derselben Position zu beenden, wie sie begonnen wurde.

Eigene Bausteine erstellen

Um übersichtlichere Programme zu erstellen oder eigene Funktionen zu realisieren muss man eigene Bausteine in Choregraphe erzeugen. Dies ist glücklicherweise sehr einfach umzusetzen. Über einen Rechtsklick auf der Programmierfläche öffnet sich folgendes Menü:

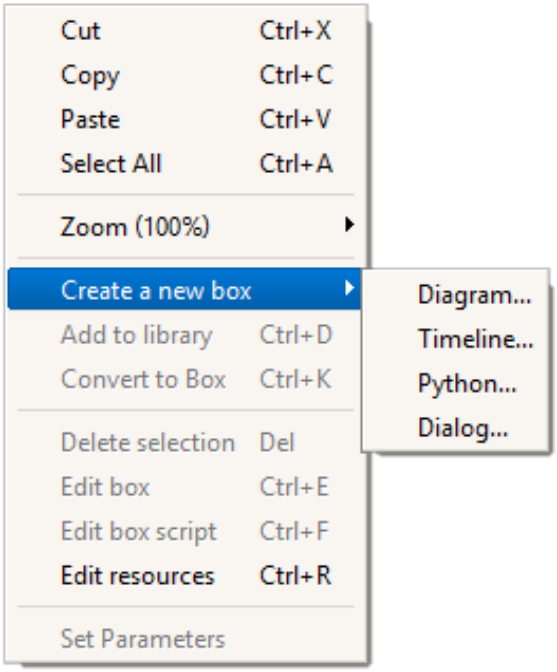


Abbildung 35: Menüpunkte um eigenen Baustein zu erstellen

Unter dem Reiter „Create a new box“ findet man nun vier Auswahlmöglichkeiten. Bei „Diagram“ handelt es sich um die Standardprogrammiervariante von Choregraphe. Man erhält also einen Baustein, in dem man wiederum mit Bausteinen programmieren kann. Diese Funktion ist also besonders dann wichtig, falls ein Programm übersichtlich gestaltet werden soll und nicht alle Funktionen direkt ersichtlich sein müssen.
Die zweite Möglichkeit ist die sogenannte „Timeline“. Hiermit können insbesondere Bewegungen sehr einfach realisiert werden und auf einer Zeitleiste angeordnet werden. Dieser Programmiermodus soll im nächsten Kapitel besprochen werden.
Zwar ist das Angebot an vorgefertigten Bausteinen in Choregraphe sehr hoch, doch es kommt durchaus vor, dass eine gewünschte Funktion noch nicht vorimplementiert wurde. Dafür gibt es den dritten Programmiermodus „Python“. Hierbei wird ein Choregraphe-Baustein erstellt, welcher in der Programmiersprache Python implementiert werden kann. Hierbei kann es sinnvoll sein die API des NAOs besser kennen zu lernen, was unter *http://doc.aldebaran.com/1-14/naoqi/index.html* möglich ist. Als kleines Beispiel, wie ein solcher Python-Baustein aussehen könnte, soll nun eine Implementierung eines Typecasts vom Datentyp „String“ zum Datentyp „Number“ dienen, welcher nicht standardmäßig in der Choregraphe- Bibliothek zu finden ist.

Typecast als Python-Baustein

Zunächst erstellen wir also eine Box des Typs „Python“ und füllen das daraufhin geöffnete Fenster aus.

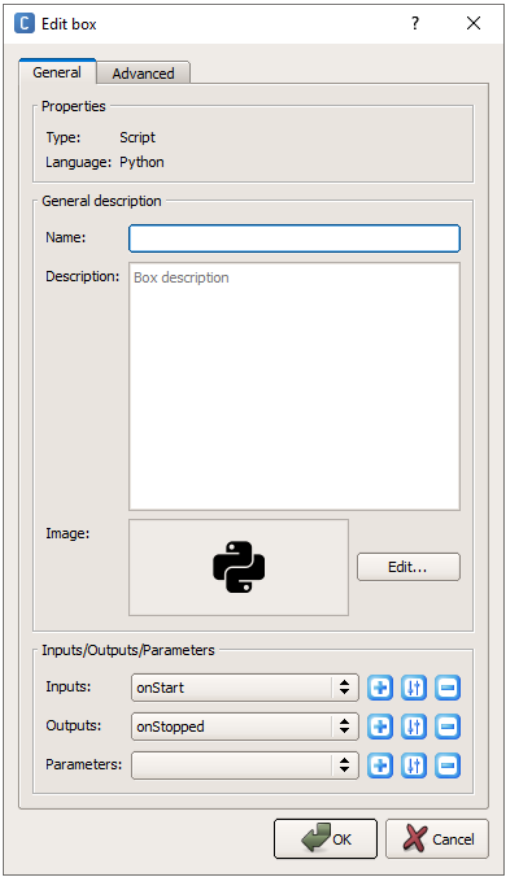


Abbildung 36: Bearbeitung der eigenen Pythonbox

Unter „Name“ kann dem Baustein als erstes ein präziser Name gegeben werden. Unter dem Reiter Image kann zudem das anzuzeigende Bild auf dem Baustein verändert werden. Hier relevant sind zudem die Datentypen der In- und Outputs. Über die „+“- und „-“-Schaltflächen können neue In- und Outputs zu dem Baustein hinzugefügt werden. Über die Schaltfläche in der Mitte können diese dann bearbeitet werden. Es öffnet sich ein neues Fenster, in welchem der Name des Inputs und dessen Datentyp verändert werden kann. Da ein Typecast von „String“ nach „Number“ realisiert werden soll, wird also der „Type“ auf „String“ abgeändert.

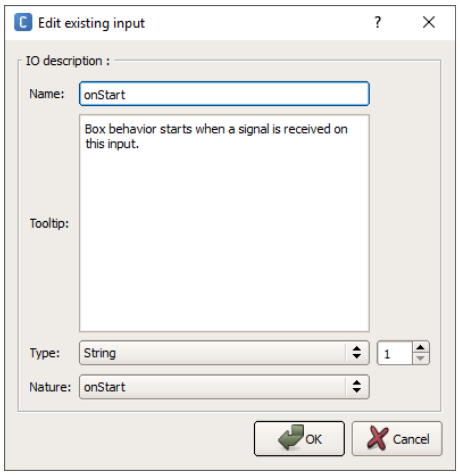


Abbildung 37: Bearbeitung des Inputs

Analog wird der Typ der Ausgabe in „Number“ abgeändert. Sobald die Bearbeitung abgeschlossen ist, wird der Baustein erstellt und auf der Programmierfläche in Choregraphe abgelegt.

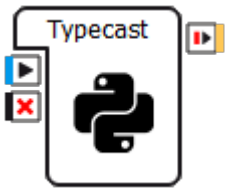


Abbildung 38: Ein eigener Typecast-Baustein

Bisher hat dieser Baustein allerdings keinerlei Funktionalität. Durch einen Doppelklick auf den Baustein gelangt man zu seinem Quelltext. Standardmäßig sind bereits die Funktionen *__init__(self), onLoad(self), onUnload(self), onInput_onStart(self)* und *onInput_onStop(self)* vorgegeben. Die Namen der „onInput“ Funktionen setzt sich aus selbigem Teil und dem Namen der Eingabevariablen zusammen. Haben diese Funktionen weitere Parameter, so handelt es sich dabei um die gewünschten Eingaben.

Typecast

```
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)

    def onLoad(self):
        #put initialization code here
        pass

    def onUnload(self):
        #put clean-up code here
        pass

    def onInput_onStart(self, p):
        #self.onStopped(p) #activate the output of the box
        pass

    def onInput_onStop(self):
        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
        self.onStopped(p) #activate the output of the box
```

Abbildung 39: Der Quelltext des Typecast-Bausteins

Um nun den eigentlichen Typecast durchzuführen, muss zunächst der Eingabestring eingelesen werden. Hier ist die Eingabe bereits durch den Parameter „p“ in der Funktion *onInput_onStart(self, p)* gegeben und kann somit direkt verwendet werden. Zunächst sollte überprüft werden, ob die Eingabe

tatsächlich eine Zahl darstellt. Dies kann mittels der Unicode-Funktion *isnumeric(…)* durchgeführt werden. Ist dieser Wert wahr, so wird der eigentliche Typecast durchgeführt, ansonsten soll nichts geschehen. Da der Typecast bereits in Python existiert, kann nun direkt der Ausgabewert überschrieben werden. Für den Typecast wird die Funktion *int(…)* genutzt. Durch self.onStopped(…) wird die Ausgabe der eigenen Box aktiviert. Kombiniert man diese Codeabschnitte erhält man folgende Implementierung.

```
def onInput_onStart(self, p):
    u = unicode(p)
    if u.isnumeric() == 1:
        self.onStopped(int(p));
```

Abbildung 40: Implementierung des Typecasts

Durch einen geschickten Aufbau kann diese Implementierung in Choregraphe getestet werden.

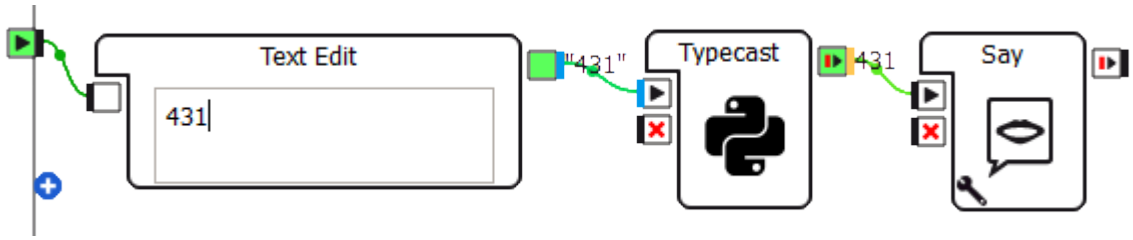


Abbildung 41: Positiver Test des eigenen Bausteins

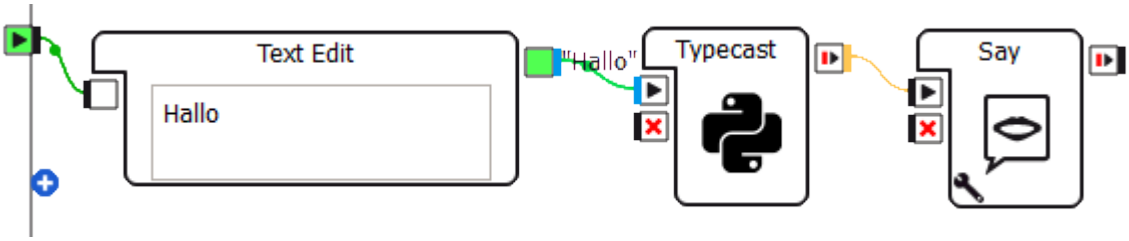


Abbildung 42: Negativer Test des eigenen Bausteins

Somit funktioniert der Typecast-Baustein wie gewünscht. Für andere Implementierungen benötigt man häufig ein tieferes Verständnis für die API des NAOs.

Eigene Bewegungen erstellen

Bevor eine eigene Bewegung mit Hilfe des Timeline-Bausteins erstellt wird, sollte stets geprüft werden, ob nicht bereits eine passende Bewegung in der Animationsbibliothek des NAO existiert. Diese sollte mit dieser Anleitung mitgeliefert werden. Wie Bibliotheken in Choregraphe eingebunden werden, soll nun erläutert werden.

Vorhandene Bibliotheken nutzen

Um eine vorhandene Bibliothek von Funktionen zu nutzen, muss diese zunächst in Choregraphe eingelesen werden. Dafür sind die Schaltflächen im Menüpunkt „Box Libraries“ am linkeren unteren Bildschirmrand zuständig.

Abbildung 43: Das Box Libraries Menü

Die erste Schaltfläche ist dazu da eine eigene Bibliothek zu erstellen. Per Drag-and-Drop können zu dieser sämtliche Bausteine aus Choregraphe hinzugefügt werden. Die hinteren drei Schaltflächen werden zum Abspeichern der eigenen Bibliotheken genutzt.
Zum Einspielen einer neuen Bibliothek, wie der Animationsbibliothek, wird also die zweite Schaltfläche benötigt. Mit einem Klick auf diese öffnet sich ein Dateiexplorer, über welchen die bereits angelegte Bibliothek gesucht werden kann. Sobald diese gefunden ist, wählt man die Bibliothek aus und kann diese über die „Open"-Schaltfläche geöffnet werden. Nach wenigen Augenblicken kann diese dann in Choregraphe genutzt werden.

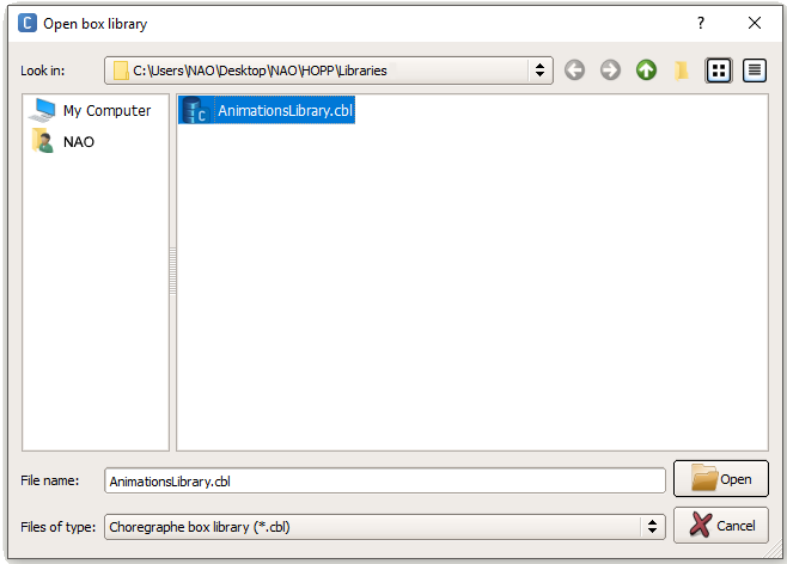


Abbildung 44: Explorer zum Finden der neuen Bibliothek

Die Animationsbibliothek, welche mit diesem Dokument mitgeliefert wird, ist in drei Unterordner aufgeteilt. Dabei spielt die aktuelle Ausgangspose des NAOs eine tragende Rolle. Die Animationen sind daher so unterteilt, dass manche von ihnen für einen stehenden NAO geeignet sind, andere nur für einen sitzenden NAO. Dies sollte stets beachtet werden, da NAO sonst hinfallen könnte. Im dritten Unterordner findet man LED-Animationen. Die Bewegungsanimationen sind zudem danach geordnet, was sie zeigen. So gibt es Bewegungen, welche Emotionen animieren, andere sind dazu da auf bestimmte Dinge zu reagieren. Zudem finden sich stets noch Animationen in der Bibliothek, welche dafür gedacht sind, dass NAO die Aufmerksamkeit auf sich ziehen kann. So sind hier beispielsweise eine Staubsaugeranimation oder eine Luftgitarrenanimation implementiert.

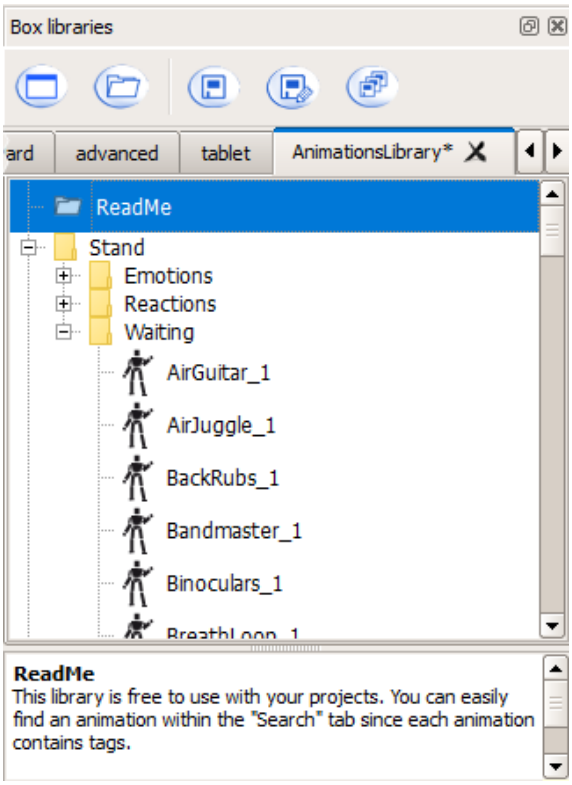


Abbildung 45: Die Animationsbibliothek

Der Timeline-Baustein

Falls keine passende Bewegung in der Animationsbibliothek existiert, so muss die Bewegung selbst erstellt werden. Dies geschieht mit dem Timeline-Baustein, welcher in dem Ordner „Templates“ in Choregraphe zu finden ist.

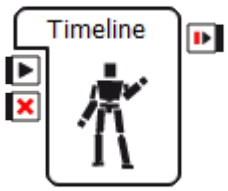


Abbildung 46: Der Timeline-Baustein

Durch einen Doppelklick auf den Baustein gelangt man zu einer neuen Programmierfläche, welche über eine Zeitachse am oberen Rand verfügt.

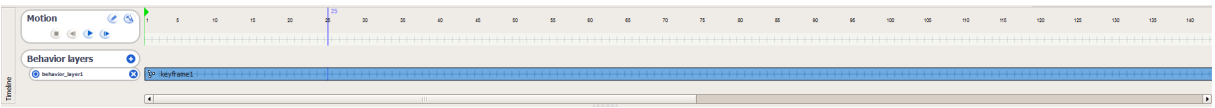


Abbildung 47: Die Zeitachse des Timeline-Bausteins

Um NAO eine Bewegung beizubringen, muss jede wichtige Körperposition in einem sogenannten Frame festgehalten werden. NAO animiert die Bewegungen zwischen diesen Frames automatisch.
Zunächst soll die Anfangsposition von NAO in einem Frame festgehalten werden. Oftmals steht NAO zu Beginn einer Bewegung. Mit Hilfe des „Stand Up“ Bausteins wird NAO in eine aufrechte Position gebracht. Im Anschluss kann mit einem Rechtsklick auf die Zeitachse ein Menü geöffnet werden, in welchem auch der Reiter „Store joints in keyframe“ enthalten ist.

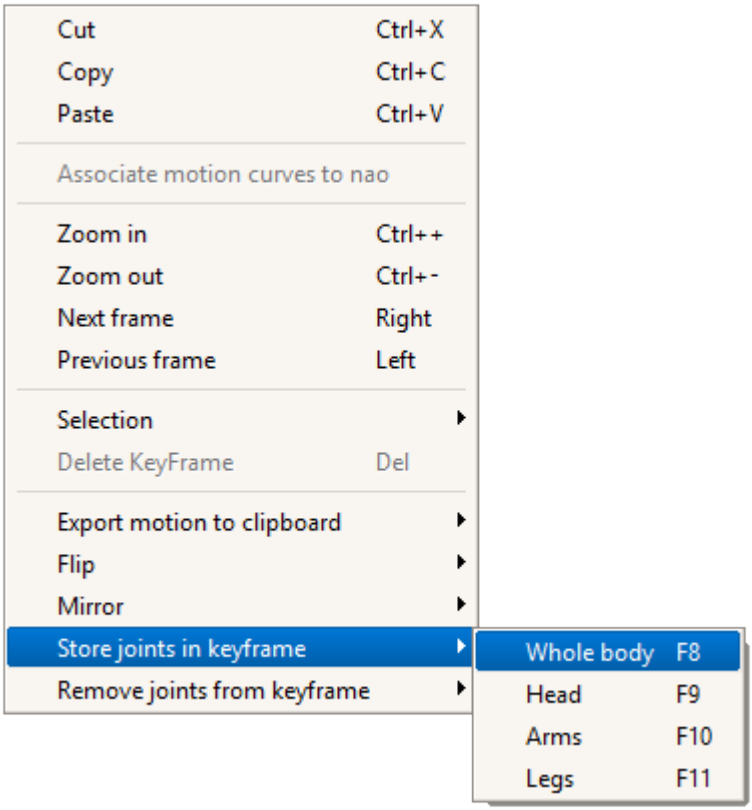


Abbildung 48: Das Menü der Zeitachse

Durch die Auswahl von „Whole body“ werden sämtliche Parameter der Motoren und Gelenke gespeichert und als Keyframe auf der Zeitachse abgelegt. Jeder weitere Keyframe muss auf dieselbe Weise auf der Zeitachse abgelegt werden.
Um einzelne Teile des Roboters zu bewegen gibt es mehrere Möglichkeiten. Zum einen kann über den Menübutton „Animation Mode“ der Animationsmodus am echten Roboter aktiviert werden.

Abbildung 49: Der Animationsmodus (grün: ausgeschaltet, rot: angeschaltet)

Sobald sich NAO im Animationsmodus befindet, muss jedes Körperteil, welches bewegt werden soll, über den Robot View am unteren rechten Bildrand ausgewählt werden, um dessen Versteifung auszuschalten. Dann kann das Körperteil bewegt werden und danach stellt man die Versteifung erneut an, um die Position des Körperteils zu halten. Hat man alle Körperteile in der gewünschten Position, speichert man die Roboterposition als neuen Keyframe auf der Zeitachse.

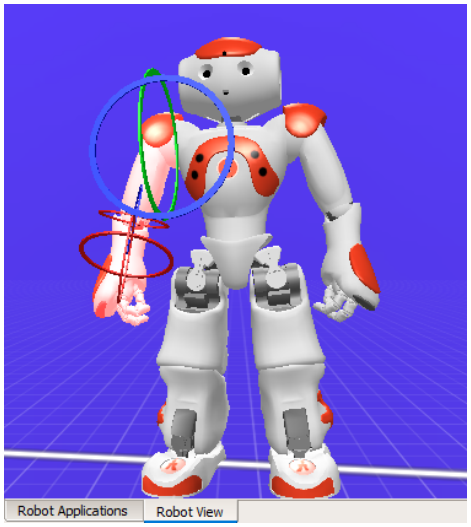


Abbildung 50: Auswahl eines Körperteils

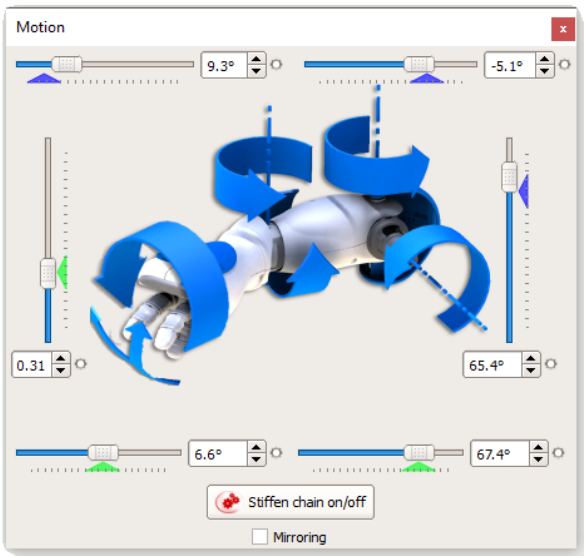


Abbildung 51: Parameter dieses Körperteils mit der Einstellungsmöglichkeit der Versteifung (rot: steif, grün: beweglich)

Eine andere Möglichkeit bietet das Robot View selbst an. Auch hier kann ohne den Roboter eine Position simuliert werden. Analog zur Animation mit dem Roboter wählt man hier das betroffene Körperteil aus und kann dann die Parameter direkt abändern und sieht direkt am Roboter oder im Robot View die Auswirkungen.
Bei allen Methoden besteht die Gefahr, dass NAO sein Gleichgewicht verliert und hinfällt. Stellen Sie also stets sicher, dass Sie bei der Ausführung neuer Bewegungen zur Stelle sind, falls NAO diese nicht korrekt ausführen kann. Auch wenn NAO beim virtuellen Test im Robot View nicht hinfällt, kann dies mit dem echten NAO geschehen.
Sobald alle wichtigen Keyframes in der Zeitachse abgespeichert sind, kümmert man sich zumeist um die passende musikalische Untermalung oder um die passenden Worte, die NAO parallel zu der Bewegung sagen soll. Dafür legt man zunächst im Zeitachsenmenü einen neuen Behaviour Layer an.

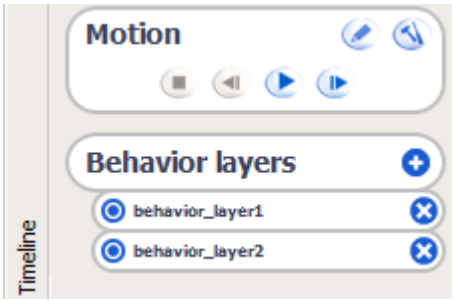


Abbildung 52: Behaviour Layer

Dort erstellt man am korrekten Zeitpunkt mit einem Rechtsklick und die Auswahl des Menüpunktes „Insert KeyFrame“ einen neuen Keyframe. Dieser kann auf der Zeitachse verschoben werden und dann mit den normalen Choregraphe Bausteinen programmiert werden, um beispielsweise einen Sound abzuspielen.

Debugging

Ein wichtiger Bestandteil jeder Programmiersprache ist das Debugging – also das Auffinden von Fehlern innerhalb eines Codes. Auch Choregraphe verfügt über einen Debugger, welcher dem Programmierer oftmals die Arbeit erleichtert. Fehlerhafter Code, beziehungsweise fehlerhafte Bausteine werden dabei während der misslungenen Ausführung rot eingefärbt.

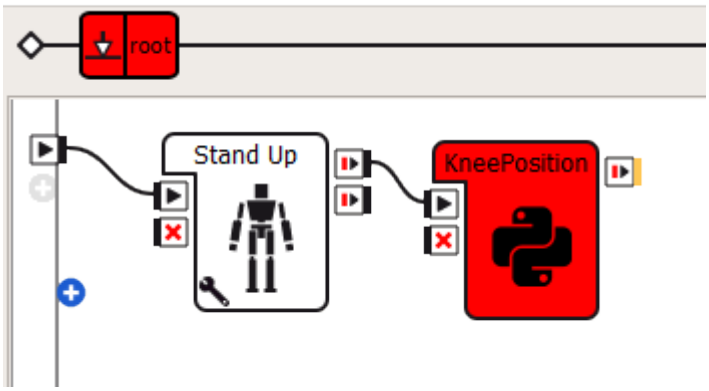


Abbildung 53: Fehlerhafter Code im Baustein KneePosition

Um den eigentlichen Fehler ausfindig zu machen, muss der Mauszeiger über dem roten Baustein positioniert und gehalten werden. Nach kurzer Zeit wird die Fehlermeldung als Kommentar eingeblendet.

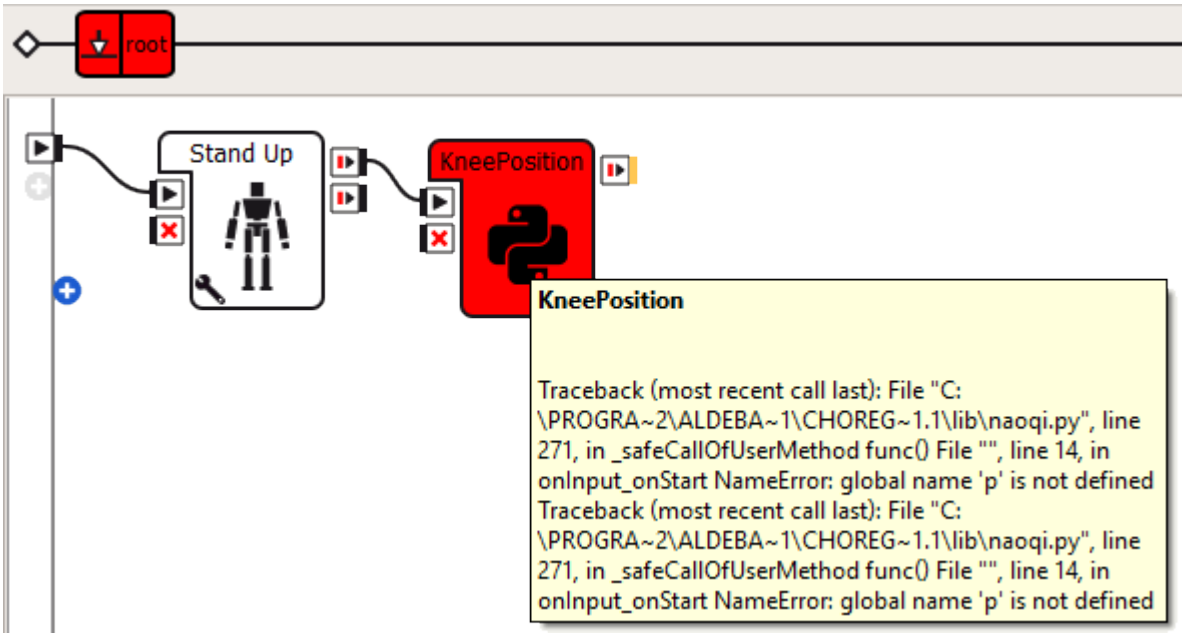


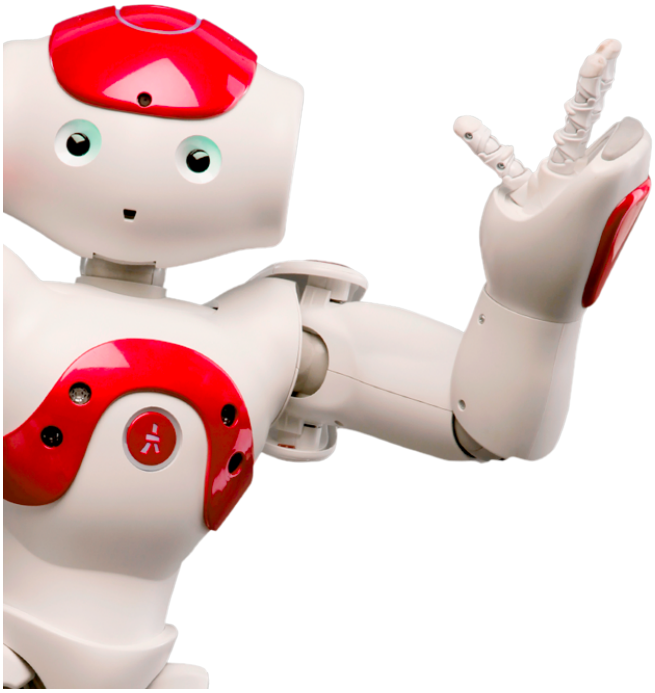
Abbildung 54: Traceback des Fehlers in KneePosition

Hier wurde beispielsweise eine Variable genutzt, welche zuvor nicht deklariert wurde. Nutzt man die Kommentarfunktion auf einem korrekt ausgeführten Baustein an, so wird unter anderem der letzte Output des Bausteins angezeigt, was sehr zur Korrektheitsprüfung eigener Bausteine beitragen kann.
Zwar sind eigene Python-Bausteine am fehleranfälligsten, jedoch kommt es leider nicht nur bei ihnen zu Fehlern. Besonders die vordefinierten Bausteine „Choice“ und „Choice Light“ sind fehleranfällig. Im Normalfall genügt es allerdings diese Bausteine neu anzulegen und damit den jeweiligen alten Baustein zu ersetzen. Funktioniert das erstellte Programm dann immer noch nicht sollte zunächst Choregraphe neu gestartet werden bevor man einen Experten hinzuziehen sollte. Oftmals findet sich auch im Forum von Aldebaran Robotics, welches unter *https://community.ald.softbankrobotics.com/en/forum/* zu erreichen ist, eine geeignete Lösung zu kleineren Problemen.

Schlusswort

Ziel dieser Einführung ist es eine kurze Einleitung in Choregraphe zu bieten und NAO somit in der Schule einsetzbar zu machen. Themen wie die Programmierung eigener Bausteine in Python wurden bewusst nur angeschnitten, da dies deutlich mehr Einarbeitungszeit benötigen würde. Prinzipiell steht einem als Programmierer mit dem NAO aber ein sehr mächtiges Werkzeug gegenüber, welches mit einfachen Mitteln komplexe Inhalte zur Verfügung stellt. So können ohne Probleme Themen wie Objekterkennung und Spracherkennung im Unterricht angesprochen werden. Darüber hinaus bietet NAO Lehrern moderne, interaktive Möglichkeiten Unterricht zu gestalten.

Falls Sie mehr über NAO und seine Einsatzmöglichkeiten erfahren wollen, ist das Buch „Using NAO: Introduction to interactive humanoid robots“ von Prof. Kisung Seo zu empfehlen. Ebenfalls sehr gut aufgearbeitet ist die Dokumentation, welche unter *http://doc.aldebaran.com/1-14/* zu finden ist.



Vielen Dank für Ihr Interesse und stets viel Erfolg mit Ihrem NAO-Roboter!

Quellen

- „Aldebaran Community Forum“, *https://community.ald.softbankrobotics.com/en/forum*, 14.12.2016
- „Aldebaran Documentation“, *http://doc.aldebaran.com/2-1/home_nao.html*, 14.12.2016
- „NAO Software 1.14.5 documentation“, *http://doc.aldebaran.com/1-14/*, 14.12.2016
- „NAO Software 1.14.5 documentation – Animation Mode“, *http://doc.aldebaran.com/1-14/software/choregraphe/animation_mode.html*, 12.12.2016
- Prof. Kisung Seo: *Using NAO: Introduction to interactive humanoid robots*. Aldebaran Robotics, 2013.
- Robo Phil: „Aldebaran NAO Choregraphe Tutorials Basic“, *https://www.youtube.com/playlist?list=PLma9tC7VHPpi8Bz4i2P5FuMlfMhjZ3nJ-*, 26.10.2016

Bildquellen

- „NAO Frontansicht“, *https://store.aldebaran.com/skin/frontend/aldebaran/default/images/nao-novice.png*, 02.11.2016
- „Choregraphe Icon“, *http://doc.aldebaran.com/1-14/_images/choregraphe_ico.png*, 02.11.2016
- „Monitor“, *http://doc.aldebaran.com/1-14/_images/monitor_start.png*, 02.11.2016
- „NAO rot“, *https://store.aldebaran.com/skin/frontend/aldebaran/default/images/prehome/nao_red.png*, 14.12.2016