Linrob URCap Handbuch - DE

Inhalt

Author: Karim Zanaty

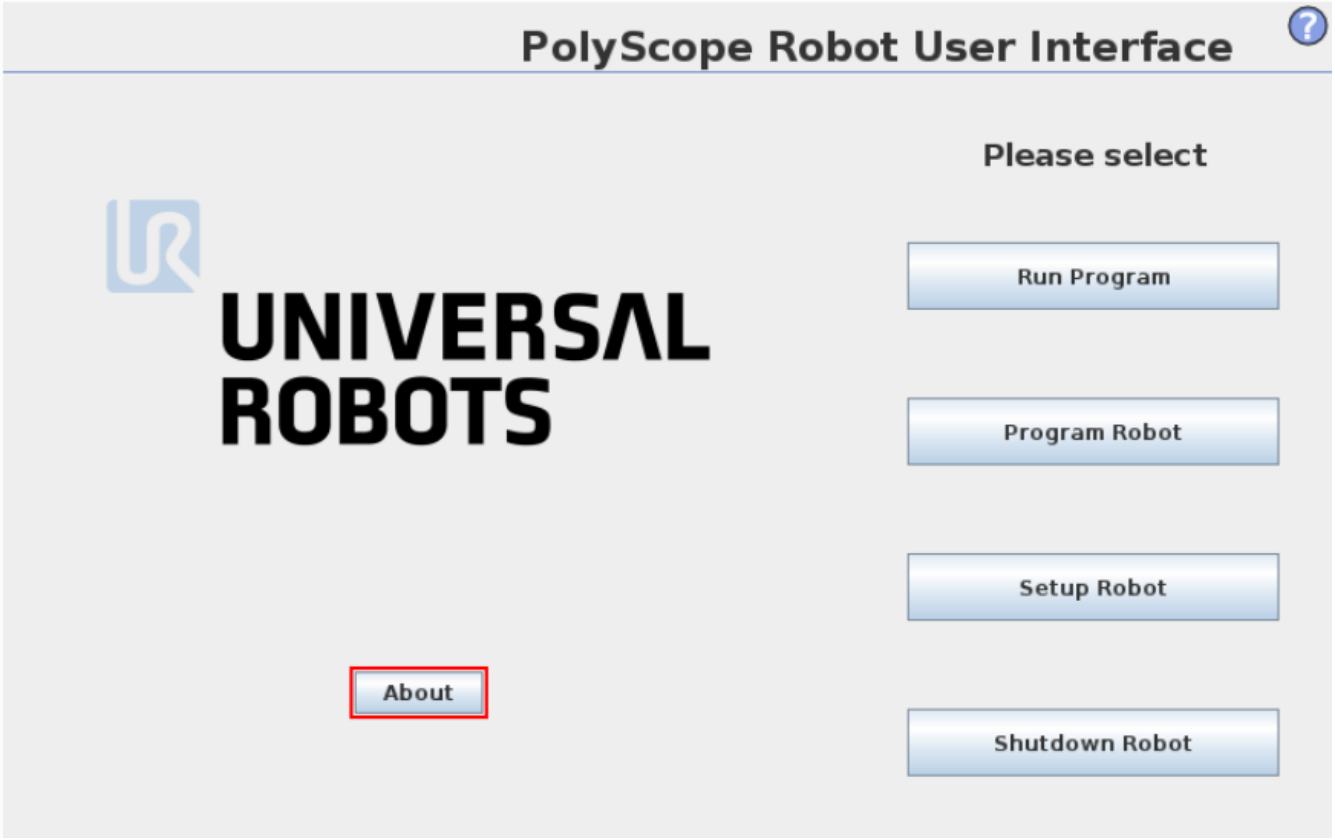
Version: 1.4.7.5

1. Allgemeine Informationen

Das Linrob URCap funktioniert mit *Polyscope 5.0.x* und höher.

Das Linrob URCap funktioniert noch nicht mit der neuesten Version, *Polyscope X*.

Um die derzeit in Ihrem Roboter installierte Polyscope-Version zu überprüfen, klicken Sie auf dem Startbildschirm auf die Schaltfläche „Info“.



2. Den linrob mit UR verbinden

2.1. Die letzte Version laden

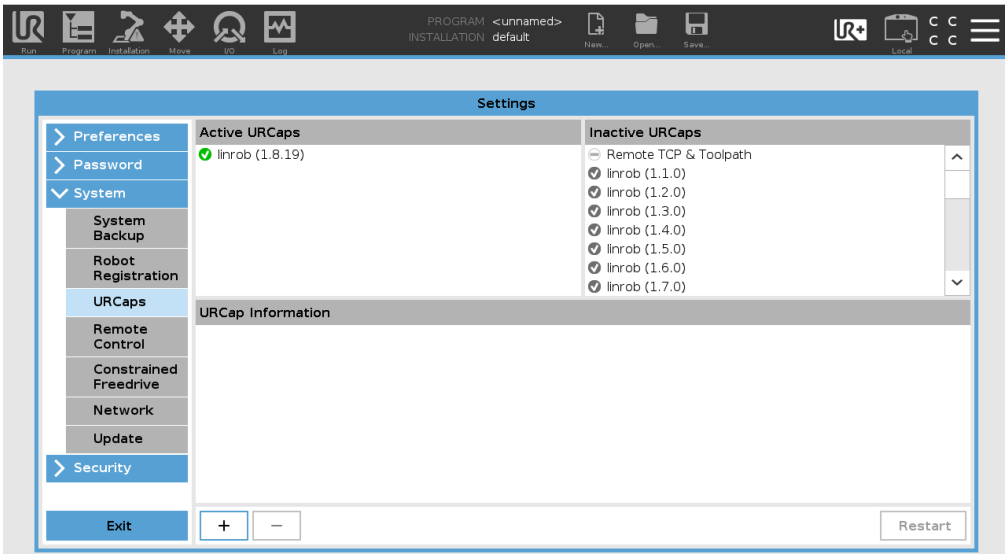
Besuchen Sie https://linrob.io/produkte/ur, um das aktuellste Handbuch zu erhalten. Laden Sie ebenfalls die entsprechende *linrob-x.x.x.urcap* und speichern diese Datei auf einem USB-Stick.

2.2. Software installieren

Nachdem Sie den Schritten unter Die letzte Version laden gefolgt sind, stecken Sie den USB Stick in das Teach Pendant des Roboters.

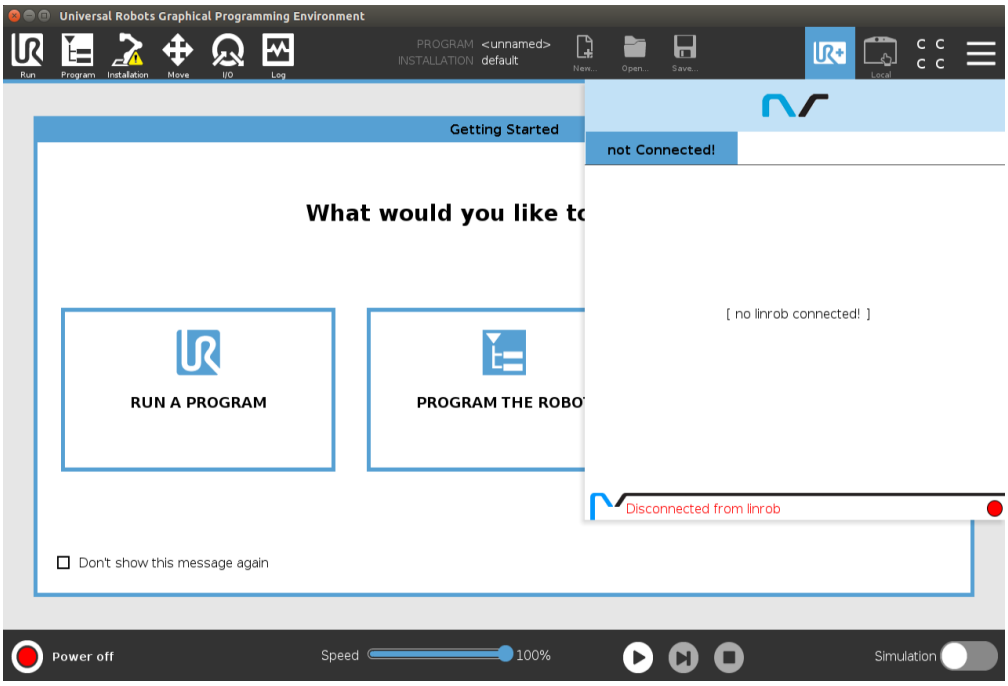
Tippen Sie auf das *Burger menu* in der oberen rechten Ecke und wählen Sie *System* aus den Einstellungen aus.

Im Untermenü des URCaps kann über den "+" Button ein neues URCap geladen werden.Öffnen Sie *linrob-x.x.x.urcap* und drücken Sie anschließend den *Neustart* Button, um es zu aktivieren.



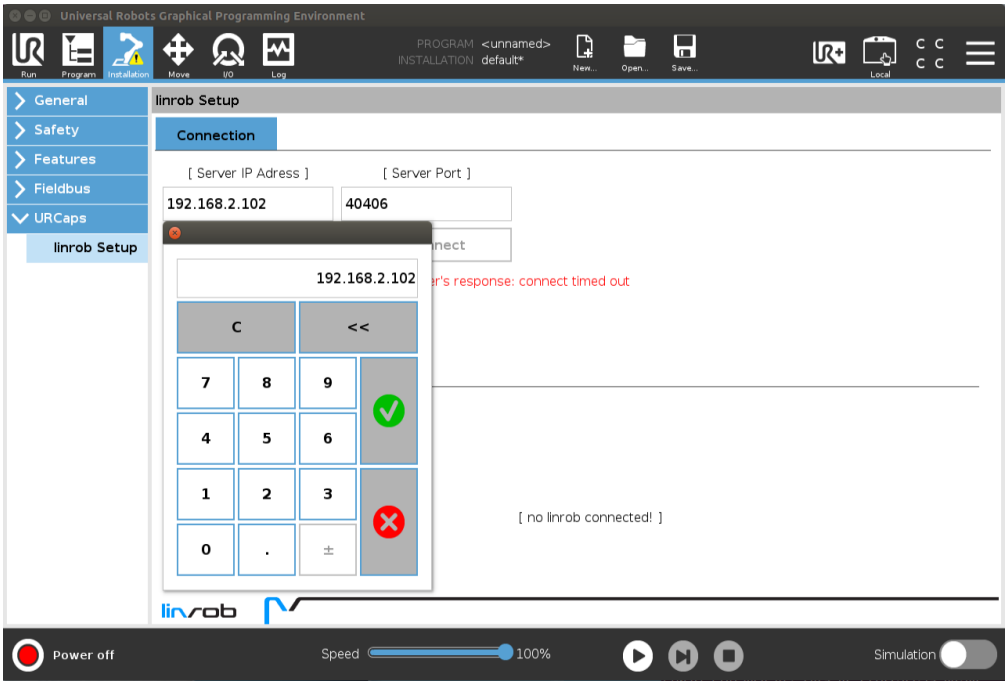
2.3. Linrob aktivieren

Wenn kein linrob verbunden ist, wird der folgende Bildschirm dies anzeigen.



Über den Reister *Installation* und das *URCaps* Menü gelagen Sie zum "*linrob Setup*".

Ein neues Fenster zeigt die Einstellungen für die Netzwerkverbindungen an.



Der Port des Linrobs ist immer : 40400

Die IP Adresse ist üblicherweise 192.168.1.1

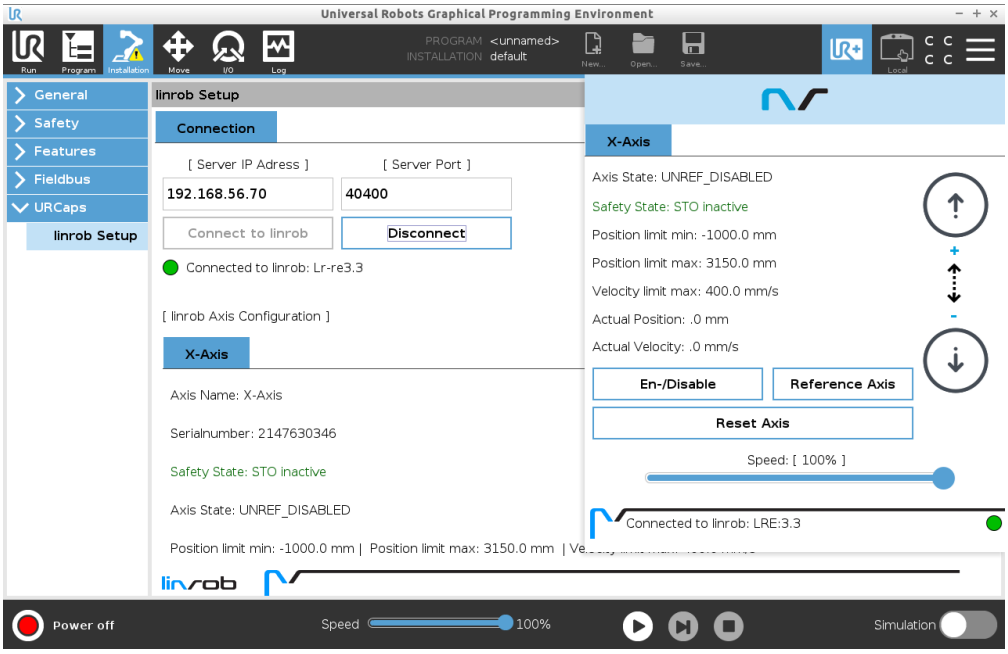
Herzlichen Glückwunsch, Sie sind nun mit dem linrob verbunden!

3. Die linCap Benutzeroberfläche

Die linrob Achsen lassen sich über das Plugin einfach und in kurzer Zeit einbinden.. Dieses Kapitel dokumentiert die Schnittstelle des Plugins.

3.1. Schnittstelle

Die Schnittstelle bietet eine benutzerfreundliche Umgebung um den linrob einzustellen.



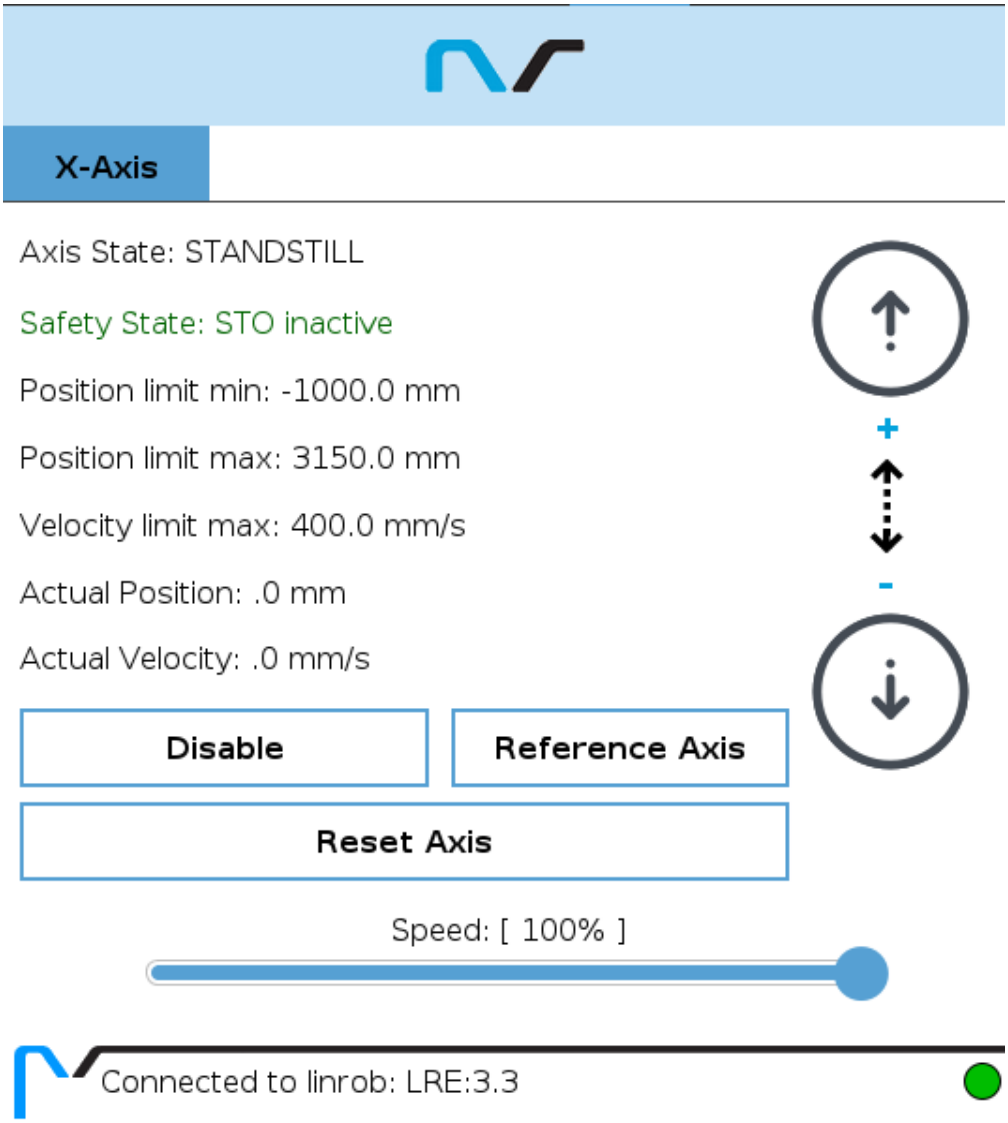
Die Oberfläche ist in mehrere Sektionen unterteilt:

Verbindung

Diese Sektion zeigt den aktuellen Verbindungsstatus. Einmal verbunden, zeigt es die IP Adresse sowie den Port an, welche für die aktive Verbindung verwendet wurden. Über die Buttons kann die Verbindung entweder aufgebaut oder wieder gelöst werden.

Linrob Achskonfiguration

Unter dem Verbindungsstatus ist die `Linrob Achskonfiguration` zu finden. Sie zeigt aktuelle Werte der Werte an wie den STO Status, Status informationen und aktuelle Limits.



Die "UR+" Toolbar in der oberen rechten Ecke ist eine schnell zu erreichende Oberfläche zur manuellen Operation mit der Achse sowie Anzeige aktueller Daten der Achse.

Der Schieberegler für die Geschwindigkeit unter dem RESET Button ist exklusiv für den Job-Betrieb und hat keinen globalen Einfluss auf die Achse.

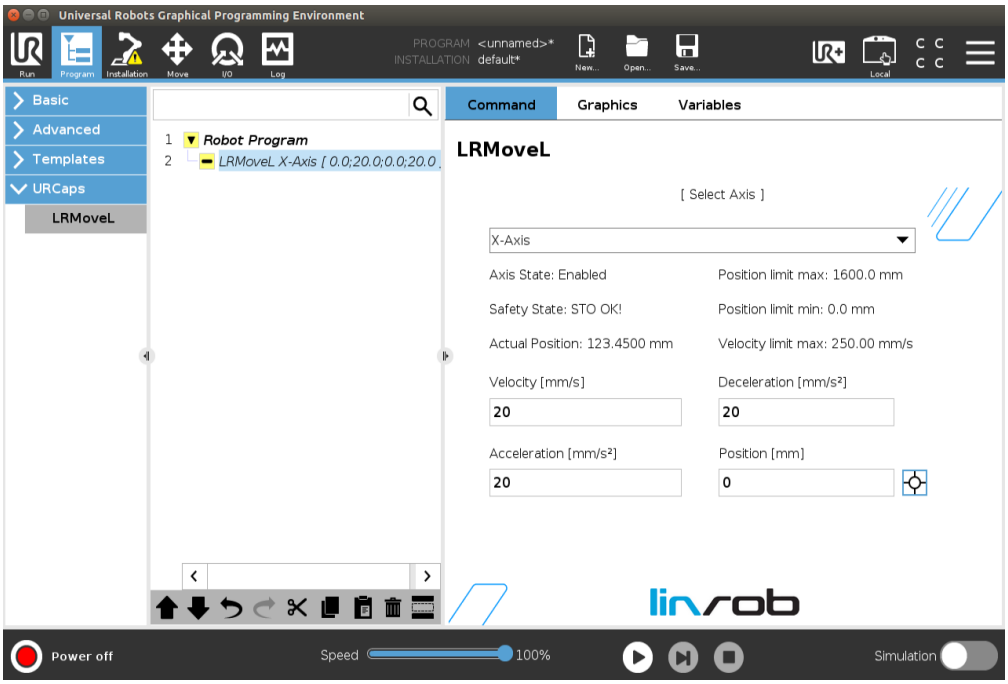
Die maximale Geschwindigkeit ist 250 mm/s.

In dieser Version können weitere Faktoren nicht vom Benutzenden geändert werden. Änderungen können gerne auf Anfrage implementiert werden.

3.2. Knoten benutzen, um den linrob zu programmieren

Auf den "Program" Tab wechseln.

- Den URCaps Button drücken.
- Das einzig verfügbare Kommando ist aktuell LRMoveL. Fügen Sie dieses Kommando ein so oft Sie es benötigen. Jede Achse und jedes Kommando benötigt ein eigenes Kommando.



Die Box neben dem Script erscheint gelb wenn keine gültige Position oder Geschwindigkeit eingetragen wurde. Anderenfalls ist diese weiß.

Nun können Sie den linrob über ein Programm in die gewünschte Position bewegen.

```
Die erlaubten Limits variieren zwischen jeder Applikation und hängen
u.A.
von internen Testergebnissen oder Absprachen mit dem Kunden ab.
Die Einheiten der Parameter sind folgendermaßen spezifiziert:
```

```
Beschleunigung: mm/s²
Entschleunigung: mm/s²
Geschwindigkeit: mm/s
Position: mm
```

3.3. Befehle

Die folgenden über Buttons ausgeführten Befehle sind fest implementiert:

3.3.1. Enable and Disable

Der Enable and Disable Button hängt vom Status der Achse ab. Nur im deaktivierten Zustand kann die Achse aktiviert. Nur im aktivierten Zustand kann die Achse deaktiviert werden.

Die Funktionen Disable und Enable werden im Hintergrund aufgerufen.

3.3.2. Referenz

Eine unreferenzierte Achse muss immer referenziert werden. Ansonsten werden keine Bewegungskommandos akzeptiert.

Die Achse wird sich in einer festen geringen negativen Geschwindigkeit bewegen bis das Signal des Induktivsensors erkannt wird. And dieser Position wird die Achse auf "0" gesetzt. Anschließend fährt die Achse auf die Position 100.00 mm.

Bitte interagieren Sie während dieses Prozesses nicht mit der Achse.

3.3.3. Reset

Der Reset ist in zwei Scenarien notwendig:

- Achse befindet sich im Status "OUTDATED"
- Achse befindet sich im Status "ERRORSTOP"

"Outdated" bedeutet, dass die Achse sich in einem Konfigurationsmodus befindet und eventuell Probleme während des Hochfahrens der Anlage aufgetreten sind. Ein einfacher Reset könnte das Problem beheben. In manchen Fällen ist ein Neustart der Anlage erforderlich.

Ein "error stop" tritt auf, wenn der Regler in seiner Funktionalität behindert wird. Dies kann auftreten im Falle von ungültigen Befehlen bis hin zum Not-Aus.

3.3.4. Jog

Die zwei Pfeile hoch und runter auf der rechten Seite des Tabs erlauben das manuelle Verfahren der Achse. Die Zielgeschwindigkeit (250 mm/s * Slider override) und weitere dynamische Parameter können nicht verändert werden.

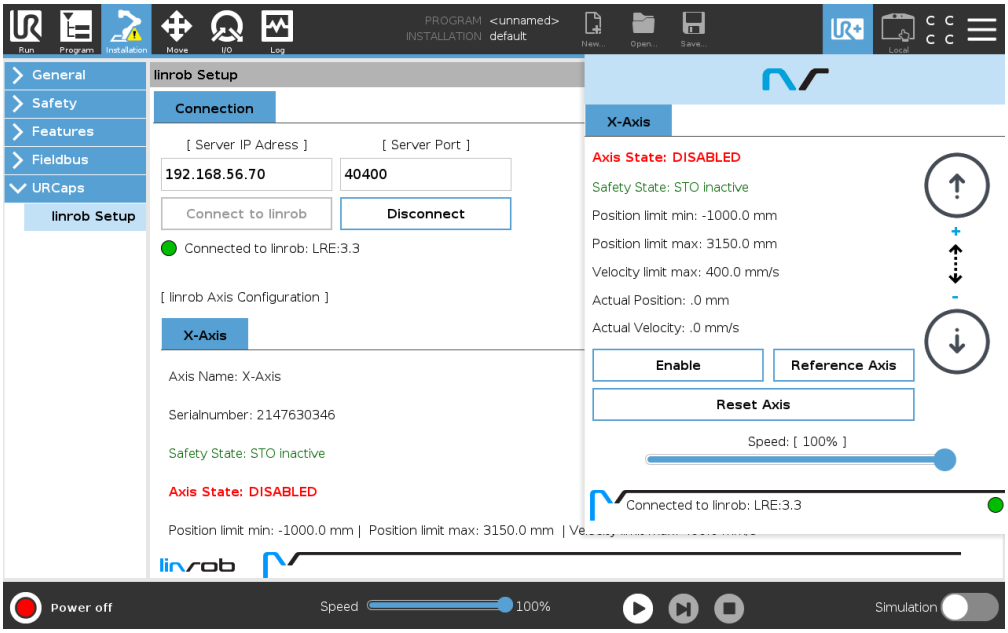
Nur Achsen im Status "STANDSTILL" können und sollten manuell verfahren werden.

Die folgenden Script-Kommandos können innerhalb der Programmierumgebung des URs verwendet werden und beeinflussen den Regler aktiv.

3.3.5. Disable

```
lr_disable_servo(axis)
```

Bei der Ausführung wird der Regler deaktiviert. Dies ist in der Benutzeroberfläche erkennbar.



Die Achse ist nicht operational und befindet sich in einem sicheren Zustand.

```
axis           (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.3.6. Enable

```
lr_enable_servo(axis)
```

Diese Funktion aktiviert den Regler und bringt den Motor in Regelung. Vorhandene Bremsen werden gelöst.

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.3.7. Move

```
lr_move_l(axis, position, velocity, acceleration, deceleration)
```

Diese Funktion verfährt die Achse absolut.

```
    axis            (int)::
        Achse zur Auswahl. Aktuell nicht in Verwendung!
    position        (float)::
        Zielposition in mm.
    velocity        (float)::
        Zielgeschwindigkeit in mm/s.
    acceleration    (float)::
        Zielbeschleunigung in mm/s².
    deceleration    (float)::
        Zielentschleunigung in mm/s².
```

3.4. Hilfsfunktionen

Gegensätzliche zu den Kommandos, welche keinen direkten Rückgabewert liefern, stellt das linCap Hilfsfunktionen zur Verfügung, welche hilfreiche Informationen über die Achse bereitstellen. Diese werden üblicherweise als Bedingungen für Schleifen etc. verwendet.

3.4.1. Achse in Bewegung

```
lr_axis_in_motion(axis)
```

Gibt zurück, ob die ausgewählte Achse sich in Bewegung befindet (string boolean).

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.2. Warte auf Position

```
lr_wait_for_position(axis)
```

Gibt zurück, ob die zuletzt kommandierte Position erreicht wurde (string boolean).

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.3. STO Status

```
lr_get_actual_sto_state(axis)
```

*Gibt STO Status des ausgewählten Reglers zurück (**Safe Torque Off**). Üblicherweise melden alle Regler denselben Wert zurück. (string boolean)*

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.4. Status

```
lr_get_actual_state(axis)
```

Gibt den aktuellen Status der ausgewählten Achse zurück. Der Status ist einer von:

OUTDATED_RESET
: Achse muss zurückgesetzt werden, sie befindet sich in einem Konfigurationsmodus.

OUTDATED
: Achse muss zurückgesetzt werden, sie befindet sich in einem Konfigurationsmodus. (Bug state)

UNREF_DISABLED
: Achse ist deaktiviert und nicht referenziert. Bitte Enable.

UNREFERENCED
: Achse ist aktiviert, aber nicht referenziert. Bitte Referenz.

DISABLED
: Achse ist deaktiviert. Es kann keine Bewegung ausgeführt werden.

STANDSTILL
: Achse ist bereit, um Befehle auszuführen.

STANDSTILL_PENDING
: Achse wird aktuell aktiviert.

ABORTING
: Aktuelle Bewegung wird angehalten. Entschleunigungsphase.

DISCRETE_MOTION
: Achse ist in Bewegung.

ERRORSTOP
: Fehler aufgetreten und beeinflusst den Regler. Achse muss zurückgesetzt werden.

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.5. Aktuelle Position lesen

```
lr_get_actual_position(axis)
```

Gibt die aktuelle Position in mm zurück.

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.6. Get velocity

```
lr_get_actual_velocity(axis)
```

Gibt die aktuelle Geschwindigkeit in mm/s zurück.

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.4.7. Reference axis

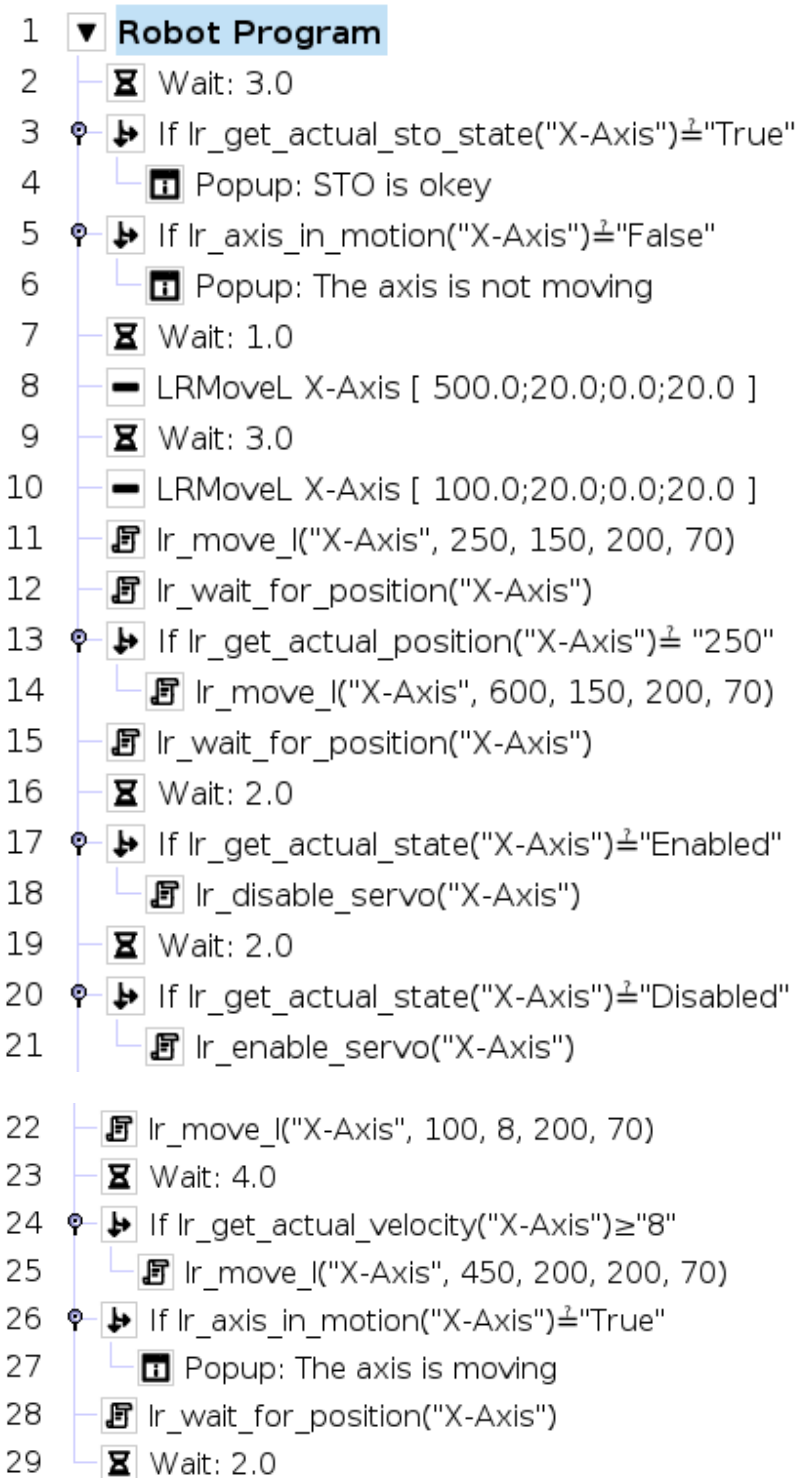
```
lr_reference_axis(axis)
```

Gibt keinen Wert zurück. Führt die Referenzierung aus.

```
axis            (int)::
    Achse zur Auswahl. Aktuell nicht in Verwendung!
```

3.5. Beispiel Programm

Das folgende Beispielprogramm soll das Programmieren der Achse verdeutlichen.



1. **Line 2:** Das Programm beginnt mit einer Warte-Funktion von 3 Sekunden (`Wait: 3.0`).
2. **Lines 3-4:** Der aktuelle STO Status der Achse wird überprüft. Bei `True` wird eine Meldung "STO is okey" angezeigt. (`If lr_get_actual_sto_state("X-Axis")=="True"` gefolgt von `Popup: STO is okey`).
3. **Line 5-6:** Das Programm überprüft, ob die Achse in Bewegung ist. Bei `False` wird eine Meldung "The axis is not moving" angezeigt. (`If lr_axis_in_motion("X-Axis")=="False"` gefolgt von `Popup: The axis is not moving`).

4. **Line 7:** Warte-Funktion von 1 Sekkunde (`Wait: 1.0`).
5. **Line 8:** Ein LRMoveL Kommando bewegt die Achse auf die Position 500 (`LRMoveL X-Axis [ 500.0;20.0;0.0;20.0 ]`).
6. **Line 9:** Warte-Funktion von 3 Sekunden (`Wait: 3.0`).
7. **Line 10:** Ein LRMoveL Kommando bewegt die Achse auf die Position 100 (`LRMoveL X-Axis [ 100.0;20.0;0.0;20.0 ]`).
8. **Line 11-12:** Die Achse soll auf die Position 250 fahren und das Programm wartet, bis diese Position erreicht ist(`lr_move_l("X-Axis", 250, 150, 200, 70)` gefolgt von `lr_wait_for_position("X-Axis")`).
9. **Line 13-14:** Die aktuelle Position wird überprüft. Wenn diese 250 ist, wird ein weiteres Kommando abgesetzt (`If lr_get_actual_position("X-Axis")=="250"` gefolgt von `lr_move_l("X-Axis", 600, 150, 200, 70)`).
10. **Line 15-18:** Wenn die Position erreicht wurde und 2 Sekunden gewartet, überprüft das Programm, ob die Achse aktiv ist (enabled). Wenn dies der Fall ist, wird die Achse deaktiviert (`If lr_get_actual_state("X-Axis")=="Enabled"` gefolgt von `lr_disable_servo("X-Axis")`).
11. **Line 19-20:** Der Status wird nochmals überprüft. Wenn die Achse inaktiv ist, wird diese wieder aktiviert (`If lr_get_actual_state("X-Axis")=="Disabled"` gefolgt von `lr_enable_servo("X-Axis")`).
12. **Line 22-23:** Das Programm bewegt die Achse zu einer neuen Position in verringerter Geschwindigkeit und wartet für 4 Sekunden (`lr_move_l("X-Axis", 100, 8, 200, 70)` gefolgt von `Wait: 4.0`).
13. **Line 24-25:** Die aktuelle Geschwindigkeit wird überprüft. Wenn diese 8.0 beträgt, wird die Achse erneut kommandiert (`If lr_get_actual_velocity("X-Axis")=="8"` gefolgt von `lr_move_l("X-Axis", 450, 200, 200, 70)`).
14. **Line 26-27:** Überprüfen, ob die Achse in Bewegung ist. Falls `True`, zeigt eine Meldung dies an: "The axis is moving" (`If lr_axis_in_motion("X-Axis")=="True"` gefolgt von `Popup: The axis is moving`).
15. **Line 28:** Es wird gewartet, bis die Achse ihre Zielposition erreicht. (`lr_wait_for_position("X-Axis")`).
16. **Line 29:** Warten von 2 Sekunden (`Wait: 2.0`).

```
Dieses Beispielprogramm zeigt die Verwendung der linCap Kommandos für
* die Bewegung der Achse
* De/Aktivieren der Achse
* Überprüfen des Achsstatus
* Überprüfen weiterer Parameter in Verbindung mit Bedingungen und Pop-Ups
```

```
Innerhalb der Installationsschnittstelle ist die Ausführung des LRMoveL-Befehls eine
```

```
eigenständige Aktion: Der Befehl überwacht selbstständig seinen
Fortschritt und
wird abgeschlossen, sobald die angegebene Position erreicht ist.
```

```
Im Programmknoten hingegen erfordert der Befehl lr_move_l eine
explizite Bestätigung
des Positionserreichens. Dies geschieht mithilfe des Befehls
lr_wait_for_position,
der als Wächter fungiert und sicherstellt, dass das Programm anhält,
bis die Bewegung
zur beabsichtigten Position bestätigt wird.
```

4. Kombinierte Bewegung von Roboter und Achse

Eine kombinierte, nicht synchrone, Bewegung von Roboter und Achse kann folgendermaßen erreicht werden:

Eine Achsbewegung kann über das Skript `lr_move_l` ausgeführt werden. Anschließend kann direkt ein `moveL` oder `moveJ` Kommando, abhängig von den Anforderungen, kommandiert werden, um den Roboter zu bewegen.

Da das Programm nicht auf den Abschluss der Bewegung der Achse wartet, werden beide Bewegungen zeitgleich ausgeführt. Dies ist keine synchrone Bewegung!

5. Statuswechsel

In diesem Kapitel behandeln wir das Antwortverhalten des linrob auf das Verwenden der Start, Stop und Pause Buttons des UR Programm-Interfaces.

Ursprünglicher Status: Stopped

Während der Installation des linCaps wird der ursprüngliche Status auf "Stopped" gesetzt. Somit wird sichergestellt, dass der Benutzende ein Programm aktiv starten muss.

Übergänge: Start, Pause, Stop

Start zu Pause: Der linrob wechselt in einen temporären pausierten Zustand. Nach dem Wechsel wird die Achse die aktuelle Bewegung anhalten.

Pause zu Start/Fortsetzen: Der linrob wechselt aus dem pausierten Status und einen laufenden Betrieb. Wenn die Achse in ihrer Bewegung unterbrochen wurde, wird der letzte Befehl erneut gesendet. Es wird nicht gewartet bis die Zielposition erreicht wurde. (See "Known issues")

Stop: Die Applikation wird gestoppt. Die Achse führt das letzte Kommando noch vollständig aus. Um das Programm neu aufzunehmen, muss diese von einem ausgewählten Punkt oder dem Beginn neu gestartet werden.

Übergänge: Einschalten, Ausschalten

Power On to Power Off: Der Zustandsübergang des Roboters deaktiviert auch die Achse, sofern sie nicht bereits deaktiviert ist.

Power Off to Power On: Der Zustandsübergang des Roboters aktiviert auch die Achse, sofern sie nicht bereits aktiviert ist.

6. Safety

Die konfigurierbaren digitalen Ausgänge der UR Roboters sollen verwendet werden, um das Signal des Not-Aus an die Achse weiterzugeben. Die notwendige Verschaltungen sind bitte den entsprechenden E-Plänen zu entnehmen.

Durch die Betätigung der Not-Aus-Taste am Programmierhandgerät wird sofort eine globale Sicherheitsreaktion ausgelöst. Diese Aktion stoppt nicht nur alle Vorgänge des Roboters, sondern stoppt auch jegliche Bewegung der Linrob-Achse.

Nach der Aktivierung des Notstopps, insbesondere beim Joggen der Achse über die UR+-Symbolleiste, ist es unbedingt erforderlich, das System ordnungsgemäß zurückzusetzen. Bei der Wiederherstellung muss die Achse erneut aktiviert werden.

Wenn der Notstopp aktiviert wird, während das Hauptprogramm des Roboters über die Programmschnittstelle ausgeführt wird, wird das gesamte Programm sofort angehalten. Zur Wiederherstellung kann die erneute Aktivierung der Achse erforderlich sein.

Sollten Betriebsparameter über die angegebenen Bereiche hinaus angepasst werden müssen, ist es ratsam, sich an unser Support-Team zu wenden, um Unterstützung zu erhalten.

Bei Verwendung von SafetyGuard-Stoppsignalen behandelt der Roboter dies nicht als Signal Not-Aus an sich. Beim erneuten Loslassen des Signals werden der Roboter und die Achse wieder aktiviert werden automatisch in ihrem eigenen Tempo zurückgesetzt und aktiviert. Dies kann je nach Vorgang bis zu 5 Sekunden dauern vom Ausgangszustand der Achse.

Wenn während dieser Zeit ein Programm ausgeführt wurde, wird es wie in Statuswechsel beschrieben angehalten und automatisch fortgesetzt.

7. Bekannte Probleme

Jede Software hat bekannte Probleme. Obwohl wir kontinuierlich an dem Produkt und den jeweiligen Funktionen arbeiten, können diese auftreten. Wir möchten jeden Kunden ermutigen, uns uns unbekannte Fehler zu melden und das linCap so iterativ zu verbessern!

In der aktuellen Version bleibt der Jog-Button hervorgehoben, wenn während des Jog-Betriebs der Not-Aus betätigt wird. Dieser Status bleibt bestehen auch wenn der Button nicht mehr gedrückt ist. Im normalen Modus setzt sich dieses wieder bei Betätigung zurück. Dies beeinträchtigt die Funktionalität nicht.

Das linCap wird, besonders in der aktuellen Zeit, regelmäßig aktualisiert. Neben dem linCap wird auf der Steuerung eine weitere Software installiert, welche die Befehle der UR in ausführbare Kommandos für den linrob übersetzt.

Wir arbeiten aktuell an einem intuitiven, einfach zu folgenden Kompatibilitätscheck. Bis zur Veröffentlichung möchten wir die Kunden ermutigen, sich mit uns in Kontakt zu setzen sobald unerwartete Inkompatibilitäten auftreten.

Auf einigen Computern neigt Polyscope dazu, einzufrieren, wenn der Linrob-Controller seine Stromversorgung verliert, während der Roboter voll mit Strom versorgt bleibt. Dies kann nur durch einen Neustart beider Komponenten (linrob und UR) behoben werden.

Wenn der Roboter während des normalen Betriebs die Verbindung zum Linrob-Controller trennt, geht die RTDE-Verbindung (RealTimeDataExchange) verloren und kann nicht wiederhergestellt werden. Dies geschieht, wenn der Roboter neu startet oder seine Stromversorgung verliert. Dies kann nur durch einen Neustart beider Komponenten (linrob und UR) behoben werden.