

KINOVA

TOGETHER IN ROBOTICS



MEET GEN3 LITE

USER GUIDE

Contents

List of Tables.....	viii
List of Figures.....	xii
Notices.....	xvi
Copyright notice.....	xvi
Disclaimer.....	xvi
Warranty.....	xvii
Important notices.....	xvii
Introduction.....	1
Normal use definition.....	1
Risk assessment.....	2
Safety directives and warnings.....	2
Robot components.....	6
Base.....	7
Base connector panel.....	7
Base LED indications.....	8
Base mounting accessories.....	9
Quick connect system.....	9
Mounting plate.....	11
Table mounting clamp.....	13
Actuators.....	13
Robot structure and skin.....	14
Wrist and Gripper.....	15
Robot communications and network interfaces.....	16
Getting started.....	18
What's in the box?.....	18
Manipulating the robot joints when the robot is powered off.....	21
Robot mounting options.....	21
Securing the mounting plate on a table with the table mounting clamp.....	23
Base underside bolting pattern.....	24

Mounting plate bolting pattern.....	25
Attaching the mounting plate to a surface with screws.....	27
Installing the robot on the quick connect base.....	29
Installing the quick connect system.....	30
Attaching the robot base to the mounting plate.....	33
Mounting the robot directly onto a surface.....	35
Powering robot from a battery using a custom wiring harness.....	36
Robot power adapter.....	38
E-stop (optional).....	38
Powering on the robot.....	39
Power-up, booting, and initialization sequence.....	40
Operating the robot.....	40
Supported control devices.....	41
Connecting an Xbox gamepad to the robot, wired.....	42
General control device mapping for the Xbox gamepad.....	42
Twist linear map.....	44
Twist angular map.....	45
Joint map.....	47
Home and retract positions.....	48
Connections between computer and robot.....	49
Establishing a communication link between the robot and a computer using USB (RNDIS).....	50
Connecting to a computer using a USB type-A to Ethernet adapter and Ethernet cable (Optional).....	53
Kinova® Kortex™ Web App.....	56
Accessing Kinova® Kortex™ Web App.....	58
Changing the robot wired connection IP address and connecting the robot to a LAN.....	60
Connecting a computer to the robot with Wi-Fi.....	62
Dimensions, specifications, and capabilities.....	64
Schematic and dimensions.....	64
Technical Specifications.....	65
Sensors.....	68
Readings of the base that are available.....	69
Readings of the actuators that are available.....	70
Readings of the wrist that are available.....	70
Readings of the gripper pose and twist that are available.....	71

Readings of the motor for the gripper fingers that are available.....	72
Effective workspace.....	72
Payload vs. workspace.....	74
Robot control.....	75
High-level and low-level robot control.....	75
Control features.....	76
Singularity avoidance.....	76
Protection zones.....	77
Joint limits.....	79
Cartesian limits.....	80
High-level and low-level robot control methods reference.....	81
High-level control modes description.....	84
Trajectory control modes.....	84
Joystick control modes.....	86
Low-level control detailed description.....	87
Configurations and safeties.....	88
Configurable parameters.....	88
Base configuration.....	88
Actuators configuration.....	90
Device configuration.....	92
Safety items.....	92
Base safeties.....	93
Actuators safeties.....	94
Gripper safeties.....	97
Kinova® Kortex™ Web App.....	100
Device availability of Kinova® Kortex™ Web App.....	100
Platform and browser support.....	103
User login.....	104
Web App layout and navigation.....	105
Pages menu.....	106
Robot control panel.....	108
Pose virtual joystick control.....	108
Translation and rotation joystick controls.....	109
Finger controls.....	110
Additional settings.....	110

Speed control.....	110
Reference frames.....	110
Z and θ_z toggles.....	110
Angular virtual joystick control.....	111
Additional settings.....	112
Virtual joystick keyboard shortcuts.....	112
Actions panel.....	114
Snapshot tool.....	114
Configurations page group.....	115
Robot configurations.....	115
Speed Limits.....	116
Controllers.....	117
Wireless & Networks.....	119
Safeties.....	120
Operations page group.....	121
Actions.....	122
Sequence editor.....	126
Importing and exporting actions or sequences.....	127
Playing back actions and sequences.....	128
Protection Zones.....	129
Systems page group.....	132
System Information.....	133
Monitoring.....	134
Upgrade.....	136
Upgrading the robot firmware and software.....	138
Users.....	139
Creating a new user profile.....	141
API for developers.....	142
Devices and services.....	142
Available services.....	143
Services, methods, and messages.....	145
Kinova® Kortex™ API and Google Protocol Buffer.....	145
Service client-server model.....	145
Blocking and non-blocking calls.....	146
Device routing and transport.....	147
Robot servoing modes.....	147
High-level servoing.....	148

Low-level servoing.....	149
Low-level bypass servoing.....	150
Notifications.....	150
Error management.....	150
Error codes.....	151
Kinova® Kortex™ GitHub repository.....	156
Kinova® Kortex™ ROS and ROS2 packages and GitHub repository overview.....	156
Kinova® Kortex™ MATLAB® API and GitHub repository overview.....	157
Guidance for advanced users.....	158
Standard robot frames.....	158
Homogeneous transforms.....	159
Homogeneous transform matrices.....	160
Denavit-Hartenberg (DH) parameters.....	161
Singular configurations.....	163
Inertial parameters definition.....	165
Inertial parameters of the robot.....	166
Maintenance.....	170
Troubleshooting.....	172
Troubleshooting resources.....	172
Tips for troubleshooting.....	172
How to respond to safety warnings and errors.....	173
Harmonized standards.....	176
Contacting Kinova support.....	178
Glossary.....	179

List of Tables

Table 1: Safety.....	2
Table 2: General.....	4
Table 3: Maintenance.....	5
Table 4: LED interpretation.....	9
Table 5: Actuator and gripper IP addresses.....	17
Table 6: KR32222HAR cable pinout.....	37
Table 7: Power-up sequence LEDs indications.....	40
Table 8: General control map elements (common controls applying to all maps).....	43
Table 9: Twist angular general controls plus.....	46
Table 10: Joint: general controls plus.....	47
Table 11: Robot geometric parameters.....	65
Table 12: Safety/Security.....	65
Table 13: Environmental.....	66
Table 14: Controller (base).....	66
Table 15: Robot.....	66
Table 16: Actuators.....	67
Table 17: Wrist and gripper.....	67
Table 18: Software/control.....	68
Table 19: Base readings available through API.....	69
Table 20: Actuators readings available through the API.....	70
Table 21: Wrist readings available through the API.....	70
Table 22: Gripper readings available through the API.....	71

Table 23: Finger readings available through the API.....	72
Table 24: Joint position limits.....	79
Table 25: Joint speed limits.....	79
Table 26: Joint acceleration hard limits.....	79
Table 27: Joint torques soft limits.....	80
Table 28: Control modes and relevant joint limits.....	80
Table 29: Behavior at joint limits.....	80
Table 30: Gen3 lite robot Cartesian limitations.....	81
Table 31: Send or play a trajectory (Cartesian or joint).....	81
Table 32: Send Cartesian command to tool.....	82
Table 33: Send command to joints.....	82
Table 34: Send gripper commands.....	83
Table 35: Low-level cyclic commands.....	83
Table 36: High-level control modes.....	84
Table 37: Cartesian reference frames options.....	86
Table 38: Base configuration.....	88
Table 39: Actuator configuration (Kinova.Api.ActuatorConfig).....	90
Table 40: Device configuration.....	92
Table 41: Base Safety items.....	93
Table 42: General actuator safety items (apply the same to all actuator sizes).....	94
Table 43: Small actuator specific safeties.....	96
Table 44: Medium actuator specific safeties.....	96
Table 45: Large actuator specific safeties.....	97
Table 46: Gripper Safety items.....	97

Table 47: Organization of the Page menu.....	107
Table 48: Indicators and control indicators.....	107
Table 49: Cartesian pose translation joystick shortcuts.....	112
Table 50: Cartesian pose orientation joystick shortcuts.....	113
Table 51: Cartesian gripper controls.....	113
Table 52: Angular joystick actuators control shortcuts.....	113
Table 53: Angular joystick gripper controls.....	114
Table 54: Available actions.....	122
Table 55: Definition of dimension according to geometry.....	131
Table 56: Details of the Monitoring page.....	135
Table 57: Error codes description.....	151
Table 58: General (used when there are no more specific error codes available).....	151
Table 59: Protocol Server (reported by the protocol server).....	152
Table 60: Protocol Client (reported by the protocol client).....	153
Table 61: Session/users.....	153
Table 62: Configuration.....	154
Table 63: Control/sequence.....	154
Table 64: Configuration backup.....	154
Table 65: Operating mode.....	154
Table 66: Servoing.....	155
Table 67: Mapping.....	155
Table 68: Network.....	155
Table 69: Resources available from the repository.....	156
Table 70: Transformation matrices.....	160

Table 71: Classical DH parameters.....	162
Table 72: Selected singularity configurations description.....	164
Table 73: Base.....	166
Table 74: Link 1.....	167
Table 75: Link 2.....	167
Table 76: Link 3.....	168
Table 77: Link 4.....	168
Table 78: Link 5.....	169
Table 79: Gripper.....	169
Table 80: Preventive maintenance.....	170
Table 81: Base safeties handling.....	173
Table 82: Actuators safeties handling.....	174
Table 83: Applicable Standards.....	176
Table 84: Existing control modes.....	180
Table 85: Types of snapshots.....	183
Table 86: Examples of topics.....	184

List of Figures

Figure 1: Robot main components.....	6
Figure 2: Controller connector panel.....	8
Figure 3: Quick connect: table screw mounting (left) and table clamp (right) mounting.....	10
Figure 4: Quick connect (closeup).....	11
Figure 5: Mounting plate mounting holes patterns (dimensions in mm).....	12
Figure 6: Table mounting clamp and mounting plate.....	13
Figure 7: Wrist and gripper.....	15
Figure 8: Grip options.....	16
Figure 9: Shipping case contents (top layer).....	19
Figure 10: Shipping case contents (bottom layer).....	20
Figure 11: Robot mounting options.....	22
Figure 12: Base and mounting holes.....	25
Figure 13: Mounting plate bolting pattern.....	26
Figure 14: Quick connect base.....	30
Figure 15: Quick connect mounting holes.....	31
Figure 16: Mounting knob location.....	31
Figure 17: Top and bottom view of the mounting plate.....	33
Figure 18: KR32222HAR power cable.....	37
Figure 19: Power button and indicators.....	39
Figure 20: General control map elements with Xbox gamepad.....	43
Figure 21: Twist linear controls with Xbox gamepad.....	45

Figure 22: Twist angular controls with Xbox gamepad.....	47
Figure 23: Joint controls with Xbox gamepad.....	48
Figure 24: Reach home and retract controls.....	49
Figure 25: Robot dimensions (measurements in mm).....	64
Figure 26: Robot nominal workspace (measurements in mm).....	73
Figure 27: Desktop.....	101
Figure 28: Tablet.....	102
Figure 29: Smartphone.....	103
Figure 30: User login.....	104
Figure 31: Initializing.....	105
Figure 32: Web App layout.....	106
Figure 33: Pose joystick panel.....	109
Figure 34: Angular virtual joystick panel.....	111
Figure 35: Configurations page group.....	115
Figure 36: Robot configurations page.....	116
Figure 37: Speed Limits page.....	117
Figure 38: Controllers page.....	118
Figure 39: Xbox Mapping tab.....	119
Figure 40: Wireless & Networks page.....	120
Figure 41: Safeties page.....	120
Figure 42: Operations page group.....	121
Figure 43: Actions page.....	122
Figure 44: '+' menu.....	123
Figure 45: Action filter.....	124

Figure 46: Information card.....	124
Figure 47: 'More' pop-up menu.....	125
Figure 48: Parameter modifications.....	125
Figure 49: Sequence editor.....	126
Figure 50: Actions playback bar.....	128
Figure 51: Protection Zones page.....	129
Figure 52: Protection Zone card.....	130
Figure 53: More menu.....	130
Figure 54: Configure protection zone: Zone Shape pane.....	131
Figure 55: Configure protection zone: Limits pane.....	132
Figure 56: Systems page group.....	132
Figure 57: System Information page.....	133
Figure 58: Monitoring page.....	135
Figure 59: Upgrade page.....	137
Figure 60: Users page.....	139
Figure 61: Card editing interface.....	140
Figure 62: Create new user.....	140
Figure 63: Services on multiple devices.....	143
Figure 64: Device routing.....	147
Figure 65: High-level servoing.....	148
Figure 66: Low-level servoing.....	149
Figure 67: Low-level bypass servoing.....	150
Figure 68: Standard frames.....	158
Figure 69: Frame definitions and dimensions (all joints at 0 position, dimensions in mm).....	160

Figure 70: DH parameters frames definitions.....	163
Figure 71: Inertial parameters conventions.....	166

Notices

Information about the current document and legal information regarding the product are outlined in the various notices.

Copyright notice



Refer to the *Term and Conditions* on the Kinova website: Kinova website <https://www.kinovarobotics.com/legal/terms-and-conditions>.

Disclaimer

Kinova[®] and the Kinova logo are registered trademarks of Kinova inc., herein referred to as Kinova.

All other brand and product names are trademarks or registered trademarks of their respective owners.

The mention of a product name does not necessarily imply an endorsement by Kinova. This manual is furnished under a lease agreement and may only be copied or used in accordance with the terms of such lease agreement. Except as permitted by the lease agreement, no part of this publication may be reproduced, stored in any retrieval system, or transmitted, modified in any form or by any means, electronic, mechanical, recording, or otherwise, without prior written consent of Kinova.

The content of this user guide is for informational use only and is subject to change without notice. It should not be construed as a commitment by Kinova. Kinova assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

Changes are made periodically to the information herein and will be incorporated into new editions of this publication. Kinova may make improvements, changes to the products, and changes to the software programs described in this publication at any time.

Any questions or comments concerning this document, the information it contains, or the product it describes may be addressed through support@kinova.ca.

Warranty



Refer to the *Term and Conditions* on the Kinova website: Kinova website <https://www.kinovarobotics.com/legal/terms-and-conditions>.



Refer to sales conditions for the defined life of service of the device.

Important notices



Warning: Read all instructions before using this product and any third-party options.



Warning: Read all warnings on the product and in this guide.

Introduction

Welcome to the Kinova® Gen3 lite robot.

Thank you for choosing our robot as a tool for your path-breaking research needs.

This document is meant to provide you with all the information you need to get up and running with your new robot and get the most out of it.

We are here to help you in your journey. If you need any help or have any questions about how to get to where you're going with the robot, please feel free to contact our support team at support@kinova.ca.

Normal use definition

Use the robot only for the expected uses.

The definition of normal use includes lifting, pushing, pulling, or manipulating a maximum load.

mid-range, continuous 600 g

full-range, continuous 500 g

The robot is designed to hold, move, and manipulate objects in the user environment. However, for some loads in certain positions near maximum load and reach, holding an object for an extended period of time may result in heating. To protect the robot hardware from excessive heat, safety thresholds shut down the robot if the temperature rises above a certain threshold. Before this is reached, an API notification is rendered as a user alert on the Kinova® Kortex™ Web App.

The robot includes a number of temperature-related safeties.

Base CPU core and ambient temperatures

Actuators CPU core and motor temperatures

Gripper Micro-controller core and gripper motor temperatures

During normal operation, the robot joints are subject to heating. If there are any temperature warnings, put down any object as soon as is practical and place the robot into a stable rest position to allow it to cool down.

The expected service life of the robot is five years when the robot is used continuously for one hour per day. The warranty period is two years.

Risk assessment

Before proceeding, it is imperative that a risk assessment be performed. Risk assessments are required by law in many countries.

Because Gen3 lite robot is a machine, the safety of the robot depends on how well it is integrated with its environment and with other machines.

There are two recommended international standards for conducting a risk assessment.

- [International Organization for Standardization \(ISO\) 12100](#)
- ISO 10218-2

The risk assessment takes into consideration all activities carried out in the context of the robot application.

Examples:

- Teaching the robot, during set-up
- Development of the robot installation
- Robot troubleshooting
- Robot maintenance
- Everyday robot operation

The risk assessment must be completed before integration of the robot in an application and must be kept up to date with any changes in the parameter settings, work environment, or tasks of the robot. The risk assessment must address configuration settings and the need for any additional emergency stop buttons.

Safety directives and warnings

Directives, warnings and safety considerations for the Kinova® Gen3 lite robot.



Important: Before operating the robot for the first time, make sure that you have read, completely understood, and complied with all of the directives, warnings, and cautionary notes listed in the tables. Failure to do so may result in serious injury or death to the user, damage to the robot, or a reduction in its useful life.

Table 1: Safety



Warning: There is no mechanical brake on the robot. If the power supply is cut or an unrecoverable error occurs, be aware that the robot will fall. However, mechanisms are in place within the actuators that will slow the descent in the absence of external power.



Warning: For your personal safety, and that of others, it is strongly recommended that the following be carried out:

Risk assessment, before integration of the robot into a given application.



Warning: For your personal safety, and that of others, **never:**

Use the robot near a flame or source of heat.

Exceed the maximum specified payload.

Attempt to stop the robot or prevent its movement by holding it.

Attempt to backdrive/force the robot joints to turn by hand while the robot is powered on. This may cause damage to the robot. The joints may be turned slowly by hand only when the robot is powered off.

Install the robot base within 20 cm of your body (base contains a Wi-Fi transmitter).

Use the robot to submerge objects in water.

Operate or store the robot outside the recommended temperature ranges.

Manipulate containers with hot or extremely cold liquids using the robot.

Operate the robot in an environment which includes atomized flammable dust/particles or explosive/flammable gases, and so on.

Manipulate sharp objects with the robot.

Drop or hit the robot, or force parts of the robot against itself. If this happens, before using the robot again, move each robot through the entirety of its joint range and then run the demo sequence, verifying no errors or warnings are present. In case of any doubts, contact Kinova support.

Move the robot recklessly in such a way that there is a realistic risk of injuring bystanders or breaking objects in the operating environment of the robot.

	Power up and boot, reboot, or upgrade firmware of the robot unless the robot is in a stable position.
	Warning: For your personal safety, and that of others, always ensure that: The robot does not encounter any obstacles (persons or objects). Although inherently safe in its default configuration, disabling the robot safeties requires that the user be responsible for ensuring a secure working space. Children are not left unsupervised with the robot. The end effector never collides with a hard surface. The grasping of objects by gripper fingers is stable, to prevent the risk of dropped or thrown objects (if using a gripper). Eye protection is worn when manipulating fragile objects with the robot. The robot has its base securely fixed to the work surface when in operation. Before using the robot, it is confirmed that there are no warnings. The robot is protected adequately before being used near any messy process; for example, welding or painting
	Warning: Do not operate the robot when the relative humidity exceeds the maximum specified limit. In such a case, put down any object in the gripper, bring the robot to a resting position and wait until the humidity decreases to an allowable value.
	Warning: The robot is not certified for use in applications in sterile environments; for example food production, pharmaceuticals, medical, surgical.

Table 2: General

	Warning: Do not connect the USB ports on the base to one another.
	Warning: It is recommended that surge protection be used to protect the robot against external surges on the main AC line which might be caused by lightning or other abnormal conditions.
	Warning: The base must be mounted as specified in the installation section, with particular attention to the bolt pattern, strength requirements and any table or tripod-specific mounting.



Warning: For transport, the robot and accessories should be placed in the original packaging with the shipping box secured in place to avoid excessive jostling during transport.



Warning: Do not expose the robot to heavy rain.

Table 3: Maintenance



Warning: Perform regular cleaning of the robot, as well as visual inspection for damage and wear, particularly the gripper and fingers.



Warning: If the robot is exposed to rain, contact Kinova support to schedule maintenance by an authorized Kinova technician.



Warning: Immediately following exposure to saline air conditions, contact Kinova support to schedule maintenance by authorized Kinova technician.



Warning: Verify robot functionality if the robot is suspected to be potentially damaged following improper manipulation of the robot or if the robot is submitted to an excessive shock or drop.



Warning: Perform a quick routine check when starting the robot to catch defects that could occur with time.



Warning: If the robot loses a part (for example a shell due to impact) or a part breaks, shut down the robot safely and leave it off. Contact Kinova technical support.

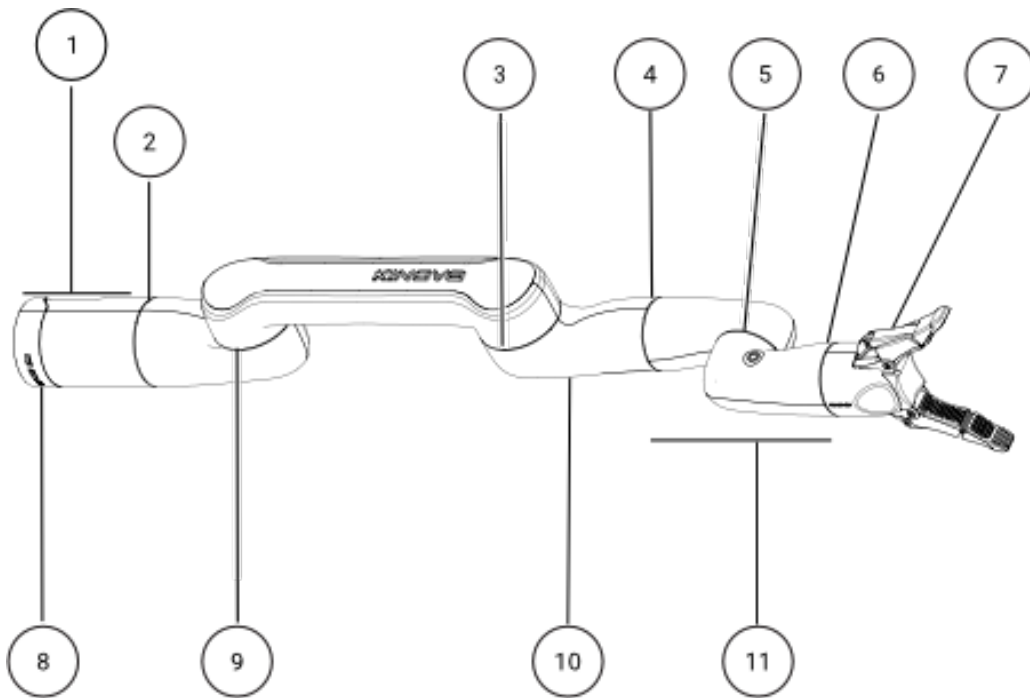
Related topics

[Risk assessment](#) on page 2

Robot components

Gen3 lite robot consists of several components.

The robot has a base with a controller, actuators, a robot structure and skin, and a wrist with a gripper.



1 Base

2 Actuator 1

3 Actuator 3 (Elbow joint)

4 Actuator 4

5 Actuator 5

6 Actuator 6

7 Gripper

8 Controller

9 Actuator 2 (Shoulder joint)

10 Forearm

11 Wrist

Figure 1: Robot main components

Base

The robot base is used in the physical mounting of the robot, power, communication, and control. The base is the brains of the robot and contains a number of important internal components.

The robot base includes a built-in controller for the robot. The base also includes a connector panel at the rear for connecting to power and external devices. The base has four mounting holes on its bottom.

The internal components of the controller include:

- CPU
- Wi-Fi adapter
- Ethernet switch and Ethernet interface
- USB interface
- Piezoelectric buzzer for audio warnings and alarms
- Accelerometers that can act as a gyroscope

A Linux web server runs on the controller and manages connectivity between the controller and the arm devices, and between the controller and an external computer. The base also includes a [Dynamic Host Configuration Protocol \(DHCP\)](#) DHCP server.

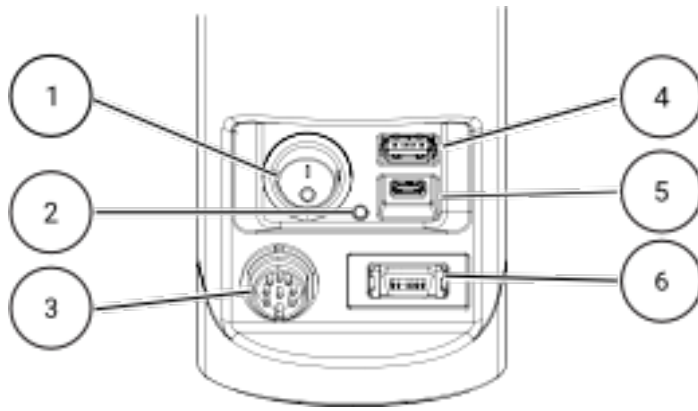
Base connector panel

The base connector panel is located on the back of the controller part of the robot base. It provides a connection point for the power supply and various cables. It also contains the power on/off switch and an LED indicator.

The base connector panel is located at the back of the base.



Important: Cables connected to the base controller must be less than 3 m long. If it is longer, perform a risk analysis. Cables longer than 3 m can potentially have an effect on radio frequency emissions and the immunity of the product.

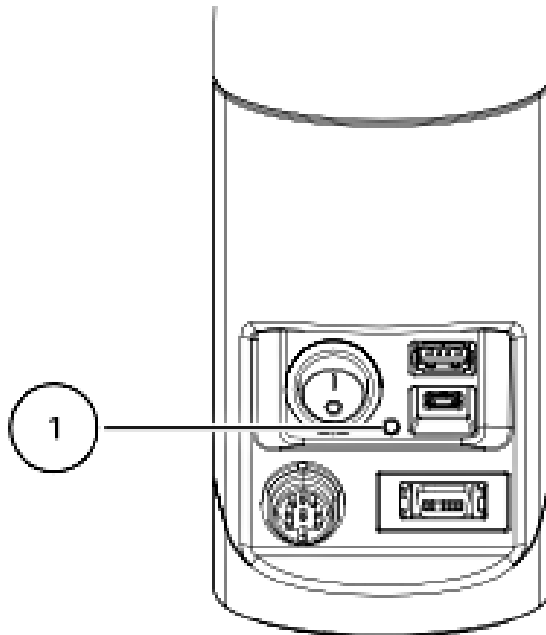


- | | |
|--|---|
| 1 On/Off power switch | 4 USB 2.0, type A |
| 2 Blue/Green/Red/Amber LED indicator | 5 Micro-USB B |
| 3 Power and RNET connector
Lumberg 0317 08 | 6 14-pin mini D ribbon joystick expansion
connector (internal use only) |

Figure 2: Controller connector panel

Base LED indications

The base connector panel includes a multi-color LED indicator.



- 1** Blue/Green/Red/Amber LED indicator

Table 4: LED interpretation

Color	Meaning
Blue	The robot is booting.
Green	The robot is ready to be used and there are no alarms and no faults.
Amber	The robot is updating.
Red	The robot is in a fault condition.

Base mounting accessories

The robot includes a number of accessories used to help with mounting the robot in different ways on a horizontal surface.

For the robot, there are different options for mounting the robot on a horizontal surface for use.

The robot package includes mounting accessories to make it easier to mount the robot. Some of these come standard with the robot, while others are optional.

- Quick connect system
 - Quick connect base
 - Mounting knob (pre-installed on the bottom of the robot base)
- Mounting plate
- Table mounting clamp (optional to use)

The robot base also includes four mounting holes on the bottom of the base that allow the robot to be mounted directly to a surface.

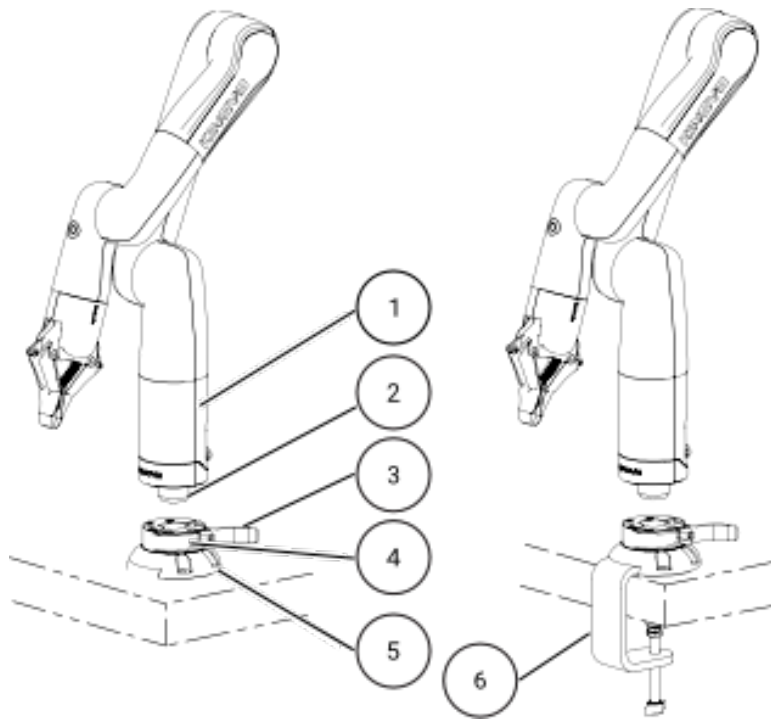
Quick connect system

The quick connect base is a two-part structure. The quick connect base can be mounted in one place, and the robot can be attached and detached to it quickly.

The base of the robot is equipped with a quick connect system that allows for simple connect and disconnect of the robot base from the mounting point. Connecting and disconnecting from the mounting point means the robot to be quickly detached from the mounting site. The quick connect system can be useful.

- Transport
- Removal of the arm for servicing, or
- Convenient re-siting of the arm between multiple installation sites

The quick connect system is supported out of the box and is the simplest and quickest way to get set up with the robot.



1 Robot base

2 Mounting knob

3 Quick connect clamp

4 Quick connect base

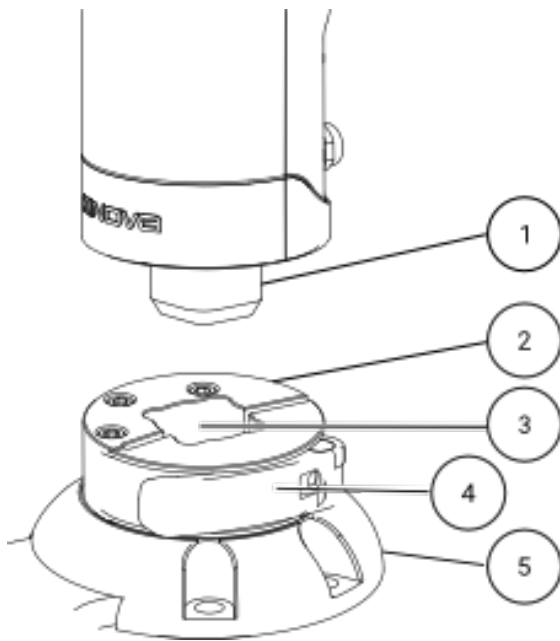
5 Mounting plate

6 Table mounting clamp

Figure 3: Quick connect: table screw mounting (left) and table clamp (right) mounting

A mounting plate attaches firmly to the mounting surface, either semi-permanently with screws or temporarily using a table mounting clamp. A quick connect base is attached to the mounting plate and can be detached easily from the docking point for the robot.

A removable mounting knob is affixed to the bottom of the robot and slides into a corresponding hole on the quick connect base; a quick connect clamp locks the mounting knob to secure the robot in place. The robot ships with the mounting knob pre-installed by default.



1 Mounting knob

2 Quick connect base

3 Hole for mounting clamp

4 Quick connect clamp

5 Mounting plate

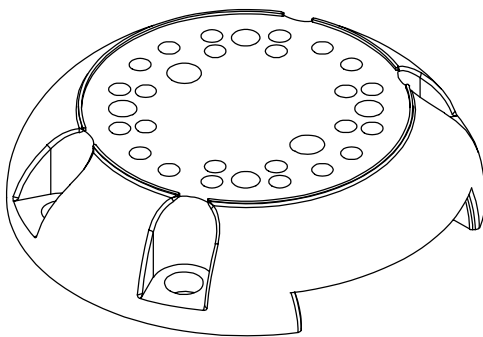
Figure 4: Quick connect (closeup)

To remove the robot from the quick connect base, flip open the quick connect clamp, detach any connected cables, and slide the robot out of the quick connect base.

Mounting plate

The mounting plate is useful for mounting the robot in different configurations.

The mounting plate provides a foundation for mounting the robot onto a horizontal surface. The mounting plate attaches to the surface, and the robot can be mounted on top of it.



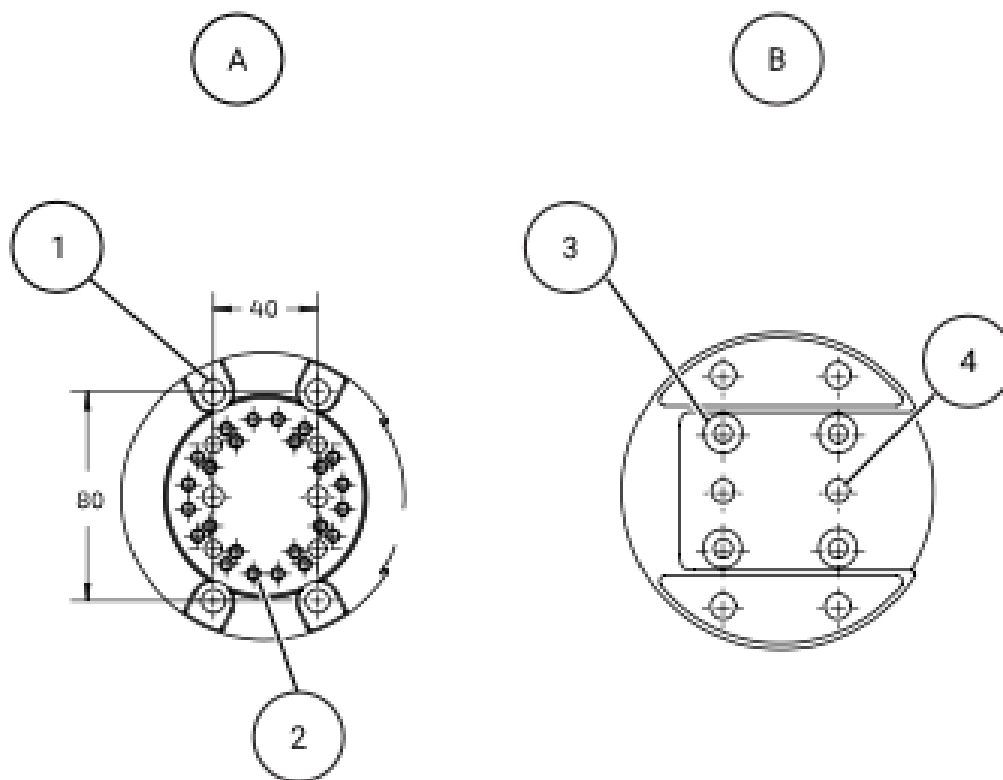
The mounting plate can be secured to the surface using either screws or the table mounting clamp.

It is faster to set up and remove the robot when the table mounting clamp is used; it is a good option for initial setup of the robot, for setting up demos, and, in general, for any temporary installation.

Securing the mounting plate to a surface with screws is a more durable solution, but requires extra set-up time to drill holes into the surface according to the mounting plate holes pattern.

The mounting plate has a number of holes drilled through it to support various mounting options.

- 4 mounting holes for mounting on table or aluminum extrusion
- 24 mounting holes to attach quick connect base; use 3 holes in 8 different possible orientations)
- 4 mounting holes to mount robot base without quick connect
- 2 M8 threaded mounting holes to attach table mounting clamp



A Top view

1 4 x mounting holes for table mount and aluminum extrusion

2 24 x mounting holes for quick connect: 8 possible positions

B Bottom view

3 4 x mounting holes for base without quick connect

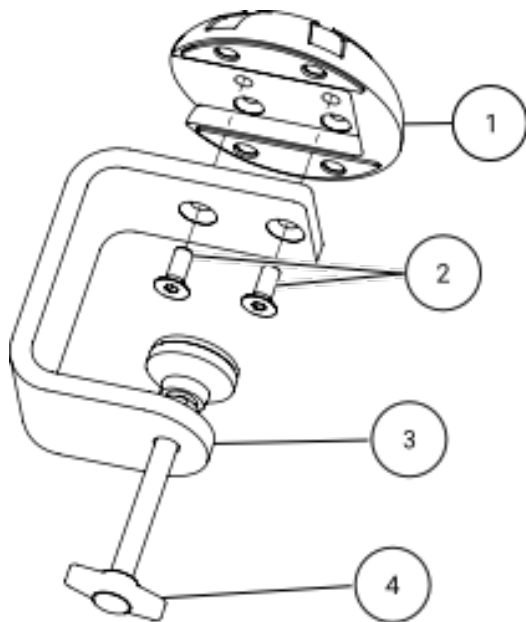
4 24 x M8 threaded mounting holes for clamp

Figure 5: Mounting plate mounting holes patterns (dimensions in mm)

Table mounting clamp

The table mounting clamp provides a quick way to secure the mounting plate on a table top.

The clamp attaches to the bottom of the mounting plate with screws. The clamp can then be secured onto the table with a tightening knob on the clamp.



1 Mounting plate

2 Screws

3 Table mounting clamp

4 Tightening knob

Figure 6: Table mounting clamp and mounting plate



Note: The table mounting clamp is always available to use. It is not mandatory to use. There are ways to mount the robot securely without the mounting clamp.

Actuators

The robot contains actuators which power the rotational movement at the robot joints. The robot contains three sizes of actuators with distinct performance capabilities. The actuators contain sensors.

The rotational motion at each of the joints of the robot is powered by rotary actuators. There is one actuator for each joint.

There are three types of actuator.

- Small
- Medium

- Large

Each actuator is equipped with the same components.

- Current sensing
- Temperature sensing (motor)
- Temperature sensing (CPU)
- Relative position and velocity sensing
- Absolute position sensing

Wrist joints, which are joints 4 to 6, use small actuators. Medium actuators are used for joints 1 and 3. A single large actuators is used for the shoulder joint, which is joint 2.

Actuators use either high-efficiency planetary gearboxes in the small and medium actuators, or low-backlash strain wave gearing in the large actuator.

The actuators are connected to each other with a series of internal cables. These cables are for power and for Ethernet data transmission.

Robot structure and skin

The robot is held together by a structural skeleton that is a mounting point for actuators and providing passages for internal wiring. The robot is covered by outer skin panels.

The robot structure is made up of two main elements.

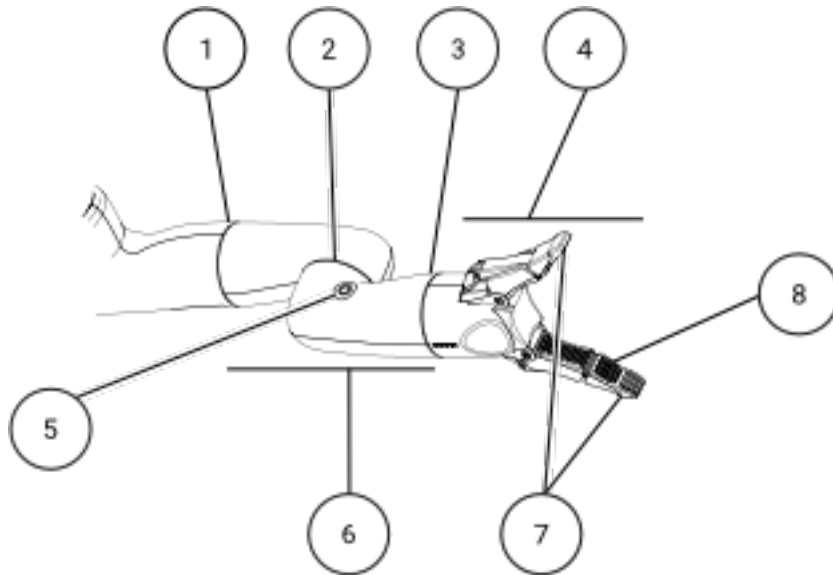
1. The robot skeleton gives structural integrity for the robot, mounting interface for actuators and control boards, and passage for internal wiring.
2. The robot skin protects the inner components of the robot and gives an appealing look to the robot.



Note: In normal operations, there is no need or reason to remove skin panels.

Wrist and Gripper

The wrist is made up of the last three actuators of the robot and includes buttons. A two finger gripper is attached allowing a range of different grip options.



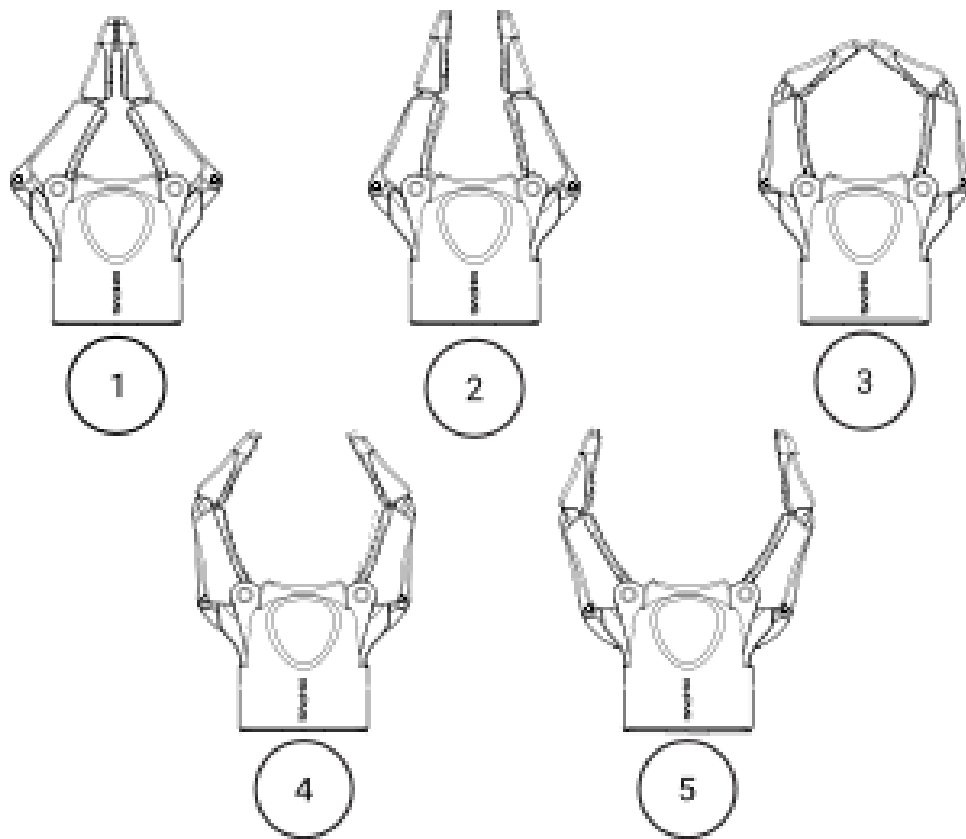
- | | |
|--------------|--------------------------------|
| 1 Actuator 4 | 5 Capacitive sense buttons x 2 |
| 2 Actuator 5 | 6 Wrist |
| 3 Actuator 6 | 7 Fingers |
| 4 Gripper | 8 Finger grip |

Figure 7: Wrist and gripper

The robot has a three-axis wrist with the range of motion enabled by actuators four through six.

The robot incorporates a two finger gripper. The finger body is composed of two different materials. The finger structure is made of a hard, but flexible plastic for structural integrity and strength, and a soft, rubber-like plastic for the grip.

The gripper fingers are actuated by a linear actuator inside the robot wrist and can open and close to accommodate a variety of grip types and object sizes.



1 Pinch grip

3 62 mm grip

5 100 mm grip

2 Small object grip

4 80 mm grip

Figure 8: Grip options

Robot communications and network interfaces

The robot is made up of a number of devices. These devices contain Ethernet switches and are connected together in a network.

The devices in the robot, from the base of the arm through the chain of actuators, to the interface module at the end of the arm, are daisy chained together using internal cables that carry power and communications.

The base, actuators, and gripper each contain Ethernet switches.

The devices and the cables in the robot communicate with each other through Ethernet communications; the data is carried between the base, the actuators, and gripper over the internal cables. Users can connect to the robot base from a developer computer to control and configure the robot through one of two wired connections.

- USB type-A port on the robot base.

An optional USB type-A to Ethernet adapter is available.

- [Remote Driver Network Interface Specification \(RNDIS\)](#) through a Micro-USB B port on the base.

A Micro-USB B to USB type-A cable is available.

Use the optional USB A to Ethernet adapter to connect to the robot base over a [Local Area Network \(LAN\)](#).

A Wi-Fi adapter in the base is used to connect wirelessly to the robot base over a local Wi-Fi network.

Each device connected to one of the Ethernet switches has an IP address to allow internal routing of communications.

The actuators and the gripper have default IP addresses.

Table 5: Actuator and gripper IP addresses

Device	IP address
Actuator 1	10.10.0.10
Actuator 2	10.10.0.11
Actuator 3	10.10.0.12
Actuator 4	10.10.0.13
Actuator 5	10.10.0.14
Actuator 6	10.10.0.15
Gripper	10.10.0.18

Getting started

The minimum information needed to start working with Gen3 lite robot includes: unboxing, physically mounting the robot securely, provisioning electrical power, controlling the robot with an Xbox gamepad, connecting a computer to the robot, and connecting to the Kinova® Kortex™ Web App.

A quick start guide, which covers much of the same material in a simplified visual format, is provided in a printed form in the box. A downloadable digital version is also made available on the Kinova website in the [Technical Resources](https://www.kinovarobotics.com/product/gen3-lite-robots#Product_resources) (https://www.kinovarobotics.com/product/gen3-lite-robots#Product_resources) section.

What's in the box?

Kinova® Gen3 lite robot shipping case contains the parts needed to put the robot together.



The interior of the shipping box has two layers.

On opening the box, you see the top layer, which contains the printed quick start guide. There are handles on the left and right side of the top layer allowing you to lift out the top layer.

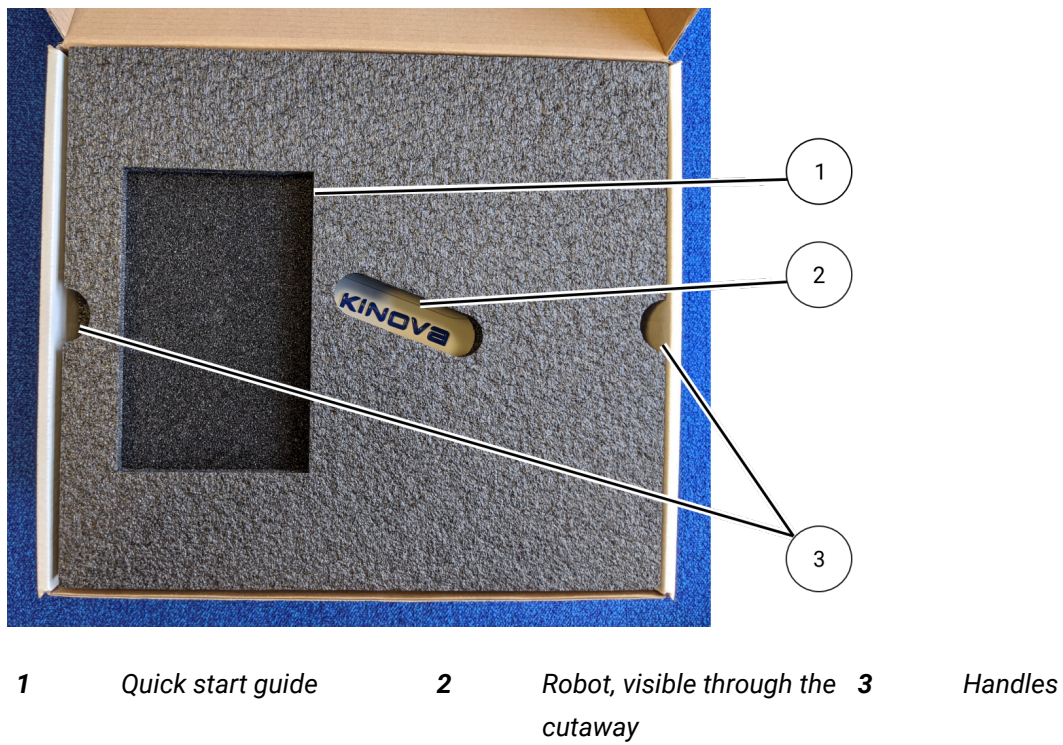


Figure 9: Shipping case contents (top layer)

Lifting out the top layer reveals the bottom layer containing the robot and its accessories.



1 Table mounting clamp

3 Robot

5 Quick connect base and mounting plate

2 Power supply

4 Cables

Figure 10: Shipping case contents (bottom layer)

The bottom layer of the box has the minimum items needed to use Gen3 lite robot.

- Robot
- Power adapter and cable
- Mounting plate and quick connect base
- Table mount clamp (optional to use)
- Micro-USB B to USB type-A cable
- Tools and fasteners

Optional items, when included, are shipped with the robot, but packaged separately.

- Xbox gamepad and cable
- Power cable E-stop
- USB-A to Ethernet adapter to help connect the robot to a computer Ethernet port or local area network



Note: The shipping case is also useful for transportation and storage of the robot. Make sure to save it and the packing layers in the box for future use.

Manipulating the robot joints when the robot is powered off

The robot joints can be manipulated by hand when the robot is powered off, which is useful when setting up the robot for the first time.

When the robot is *switched on*, the actuators hold their *position* and prevent the *joints* from moving in response to external forces and torques. When the power is on, the arm does not move except when commanded. The arm joints are stiff and the joints cannot be rotated manually by a person.

When the robot is switched off, like it is when you first receive the robot, the joints can be moved by hand slowly.



Note: The shoulder joint, which is joint 2, is stiffer than the others, but can be moved.

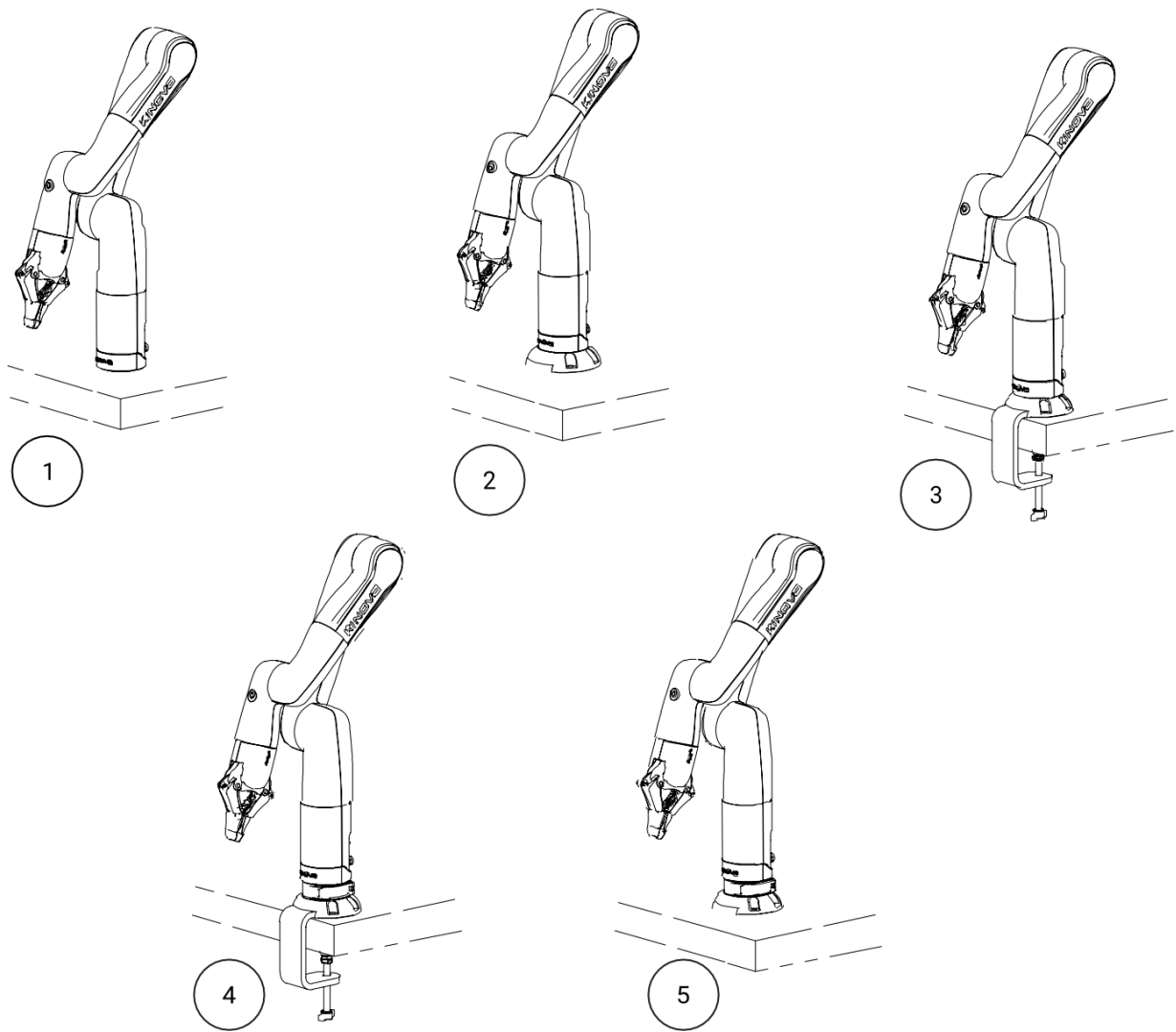
Moveability of the joints when the robot is not switched on is useful when taking the robot out of the box and setting it up. Joints of the robot can be arranged into a stable, balanced position before mounting and switching on the robot.

Robot mounting options

The first step to getting started with the arm after unboxing is to physically mount the arm in a stable manner so that the robot can be connected and used.

It is possible to mount the robot in different ways, depending on the needs of your particular application.

- Mount the robot using the quick connect system and an installation that uses a table clamp and a mounting plate affixed to surface with screws
- Mount the robot onto mounting plate and secure the mounting plate with a table clamp and screws, affixed to the surface.
- Mount the robot base directly onto the surface using screws.



1 Base directly to surface

3 Mounting plate with table mounting clamp

5 Quick connect system with mounting plate directly to surface

2 Mounting plate directly to surface

4 Quick connect system and table clamp

Figure 11: Robot mounting options

Securing the mounting plate on a table with the table mounting clamp

Secure the mounting plate on a table using the table mounting clamp.

Before you begin

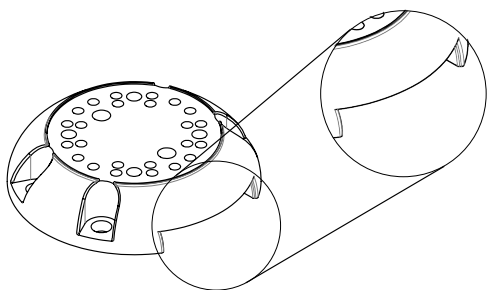
Determine in advance whether the robot is to be mounted in a fixed way onto the mounting plate or it is to be mounted with the quick connect setup. Fixed mountings require mounting the robot onto the mounting plate first; it is the best way to access the bottom of the mounting plate when affixing the robot base to the mounting base.

About this task

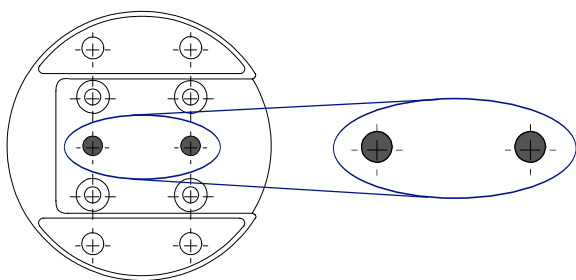
Use the table clamp for a quick, temporary setup of the robot on a table top. The table clamp must be affixed with screws to the mounting plate.

Procedure

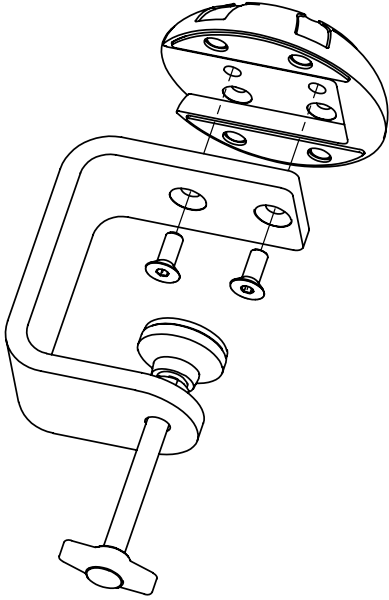
1. Place the table mounting clamp into the slot on the bottom of the mounting plate.



2. Align the two holes on the table mounting clamp with the two corresponding holes on the bottom of the mounting plate.



3. Attach the table clamp to the mounting plate with screws.



4. Turn the tightening knob on the table clamp to open up the clamp and then slide the clamp, with the mounting plate attached, over the edge of the table.
5. Turn the tightening knob by hand until the mounting plate is firmly clamped onto the table top.



Note: Do not over torque.

Related topics

[Table mounting clamp](#) on page 13

[Attaching the robot base to the mounting plate](#) on page 33

[Installing the quick connect system](#) on page 30

Base underside bolting pattern

Before attempting to affix the robot base directly to a surface, become familiar with the bolting pattern on the bottom of the base.

The underside of the base has four M6 screw holes for mounting purposes, with the centers of the holes at the corners of a 40 mm x 40 mm square. These holes can be used for attaching the robot base to either the mounting plate or directly to a surface. In the second case, mounting directly onto a surface, holes must be drilled through the surface so that screws can go through from the other side and into the base mounting holes from underneath.

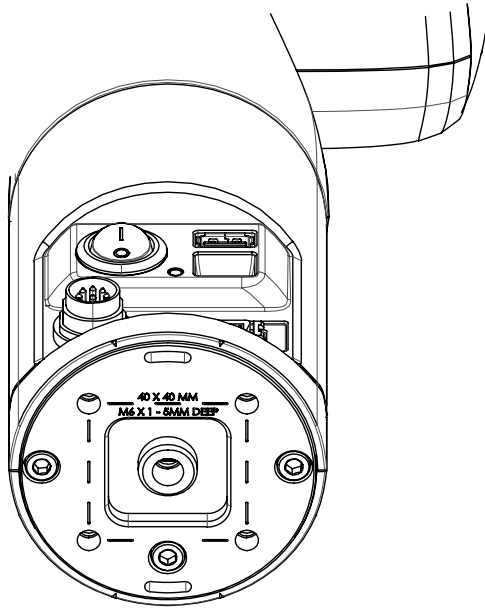
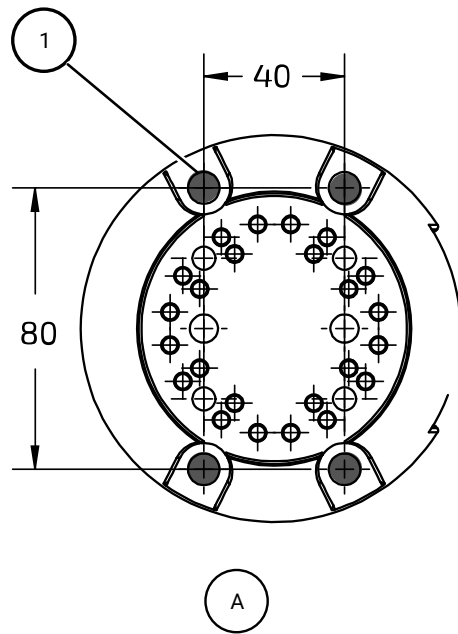


Figure 12: Base and mounting holes

Mounting plate bolting pattern

The bolting pattern of the mounting plate for affixing it to a surface or aluminum extrusion with screws is useful when mounting the robot to a surface using the mounting plate.

The mounting plate has four screw holes available for mounting the plate to a flat horizontal surface or to an aluminum extrusion.



A Top view of mounting plate

1 4 x mounting holes for table mount and aluminum extrusion

Figure 13: Mounting plate bolting pattern

Attaching the mounting plate to a surface with screws

One of the ways to mount the robot is directly to a surface that may be neither flat nor horizontal.

Before you begin

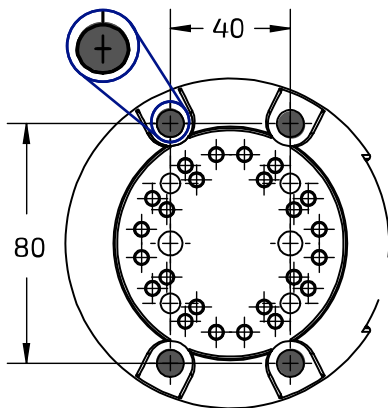
Gather four (4) M8 socket head cap screws (SHCS) for attaching the mounting plate. Determine in advance whether the robot is to be mounted in a fixed way onto the mounting plate or it is to be mounted with the quick connect setup. Fixed mountings require mounting the robot onto the mounting plate first; it is the best way to access the bottom of the mounting plate when affixing the robot base to the mounting base.

About this task

The mounting plate can be mounted to a surface using either a table mounting clamp or directly with screws. The clamping mount offers a faster, simpler, and more flexible solution that is good for temporary setups like for demos or initial experimentation. Attaching the mounting plate with screws offers a more stable, permanent solution that works better for a fixed installation at a site. [Installing the robot on the quick connect base.](#) on page 29

Procedure

1. Drill holes in the mounting surface based on the bolting pattern of the mounting plate.



2. Place the mounting plate on the surface, aligning the mounting holes with the holes in the surface.
3. Pass the screws through the mounting holes from the top of the mounting plate and affix the mounting plate to the surface using the four M8 screws.

What to do next

Connect the robot to the power supply and E-stop.

Related topics

[Attaching the robot base to the mounting plate](#) on page 33

[Installing the quick connect system](#) on page 30

[Powering on the robot](#) on page 39

[Mounting plate bolting pattern](#) on page 25

Installing the robot on the quick connect base.

The quick connect base makes installation fast.

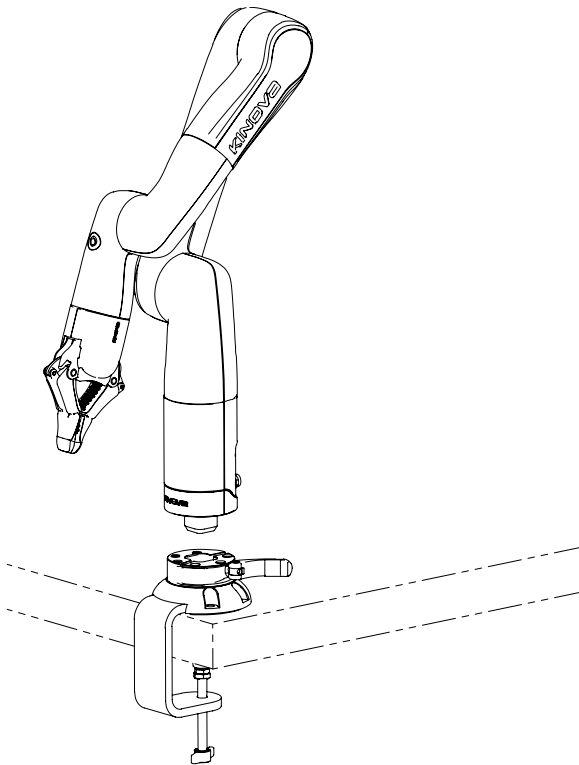
Before you begin

Install the quick connect system.

About this task

Procedure

1. Make sure that the quick connect clamp on the quick connect base is opened.
2. Lower the base of the robot so that the mounting knob goes into the opening on the quick connect base.



3. Close the quick connect clamp to secure the robot in place after the robot is fully lowered onto the quick connect base.

Related topics

[Robot mounting options](#) on page 21

[Installing the quick connect system](#) on page 30

Installing the quick connect system

The quick connect system includes the quick connect base and the base mounting knob.

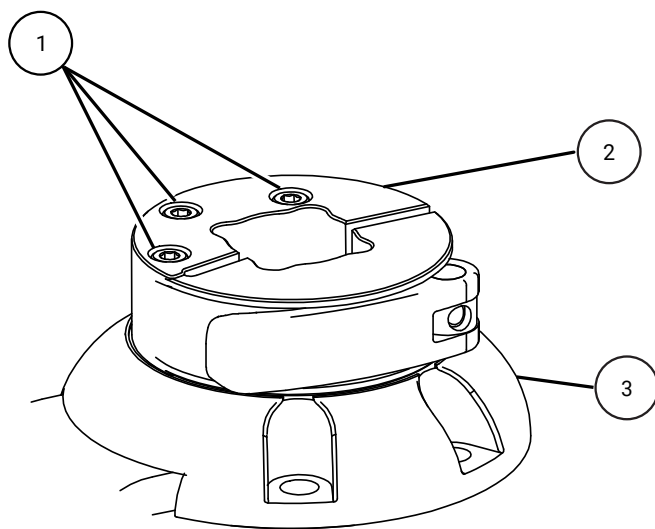
Before you begin

Use screws to install the quick connect base on the mounting plate and a screw to install the mounting knob on the bottom of the robot. The mounting plate must be mounted to a surface, either with screws or using the table mounting clamp.

About this task

The quick connect system offers a mechanism to easily move a robot between multiple installation sites. The quick connect system includes a mounting base where the robot can be quickly docked and undocked without having to go to the trouble of a full installation and deinstallation.

The mounting plate has 24 same-sized mounting holes that allow the quick connect base to be attached in eight different orientations. The quick connect base has three corresponding-sized mounting holes.

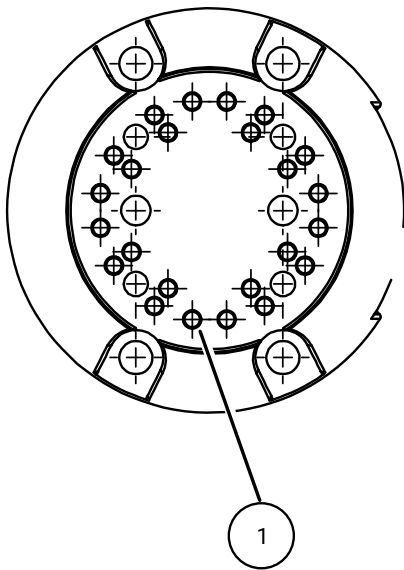


1 Mounting holes

2 Quick connect base

3 Mounting plate

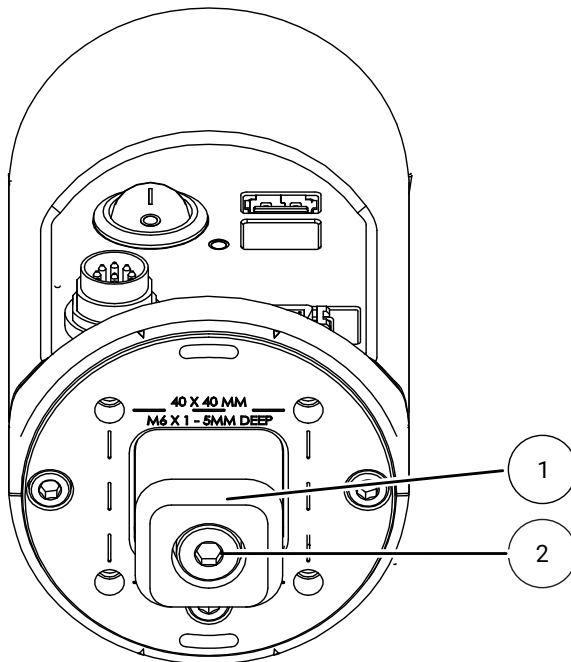
Figure 14: Quick connect base



- 1 24 x mounting holes for quick connect
8 possible positions

Figure 15: Quick connect mounting holes

The robot is shipped with the mounting knob already attached to the bottom of the robot base.



- 1 Mounting knob

- 2 Screw

Figure 16: Mounting knob location

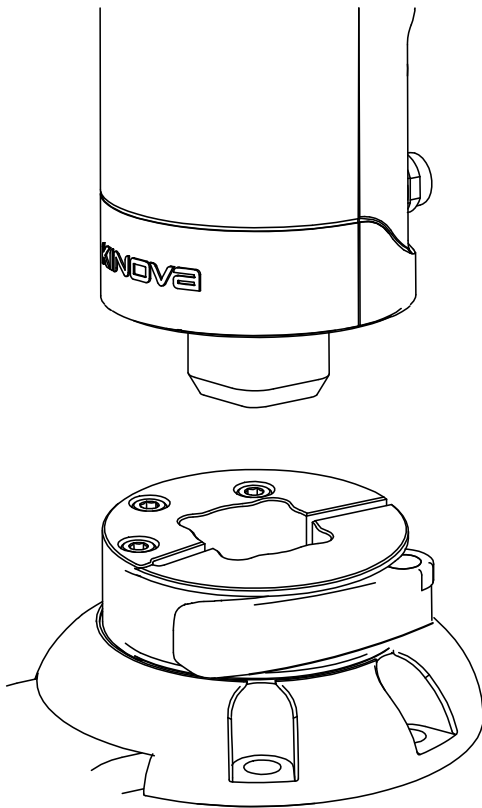
Procedure

1. Align the quick connect base on the mounting plate so that holes line up.
2. Attach the quick connect base to the mounting plate using screws.
3. Re-attach the mounting knob to the bottom of the robot base with a single screw in the center of the knob if it is not attached.



Note: The mounting knob is removed when the robot is mounted directly on a surface or directly onto the mounting plate.

Results



Attaching the robot base to the mounting plate

One of the ways to mount the robot is to attach the robot base to the mounting plate.

Before you begin

Remove the mounting knob if it is still attached. Detach the mounting plate if it is still attached.

Access to the bottom of the mounting plate is required.

About this task

The mounting plate has four M6 mounting holes on the top in a square pattern matching the bolting pattern on the bottom of the robot base. Use the M6 mounting holes to mount the robot base onto the mounting plate.

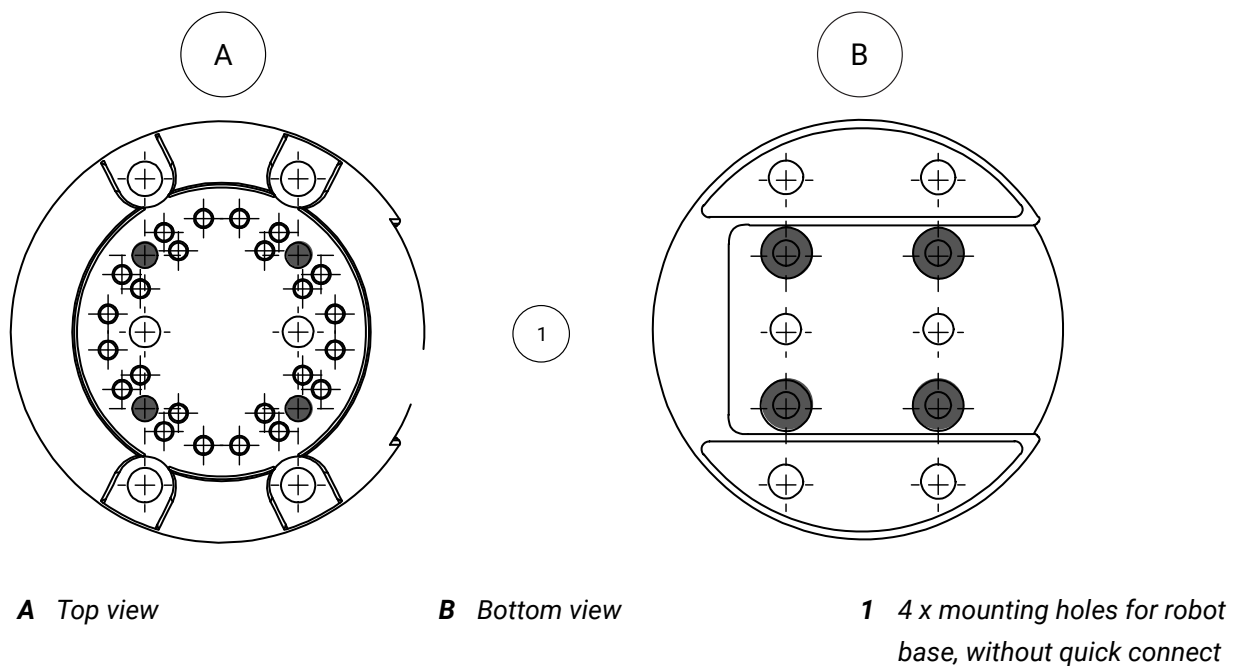
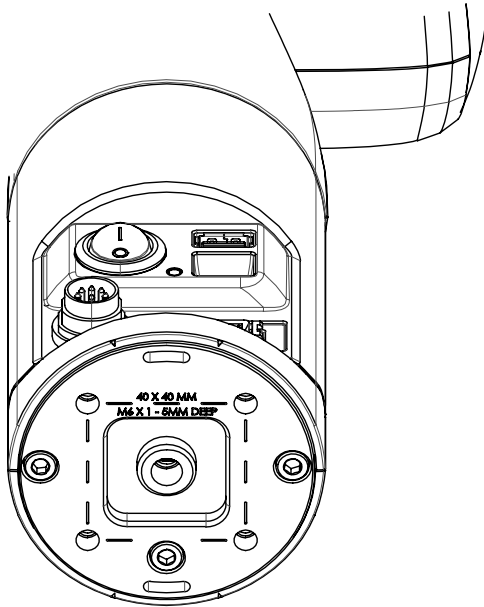


Figure 17: Top and bottom view of the mounting plate

Procedure

1. Make the top surface of the mounting plate face the bottom of the robot base.

2. Align the square hole pattern on the mounting plate with the holes on the bottom of the robot base.



3. Pass M6 screws through the mounting plate from below into the holes in the bottom of the robot base.
4. Tighten the screws.

Mounting the robot directly onto a surface

The robot can be mounted on a flat, horizontal surface without using the quick connect system.

Before you begin

Gather four (4) M6 screws to affix the robot to the surface. Remove the mounting knob if it is attached.

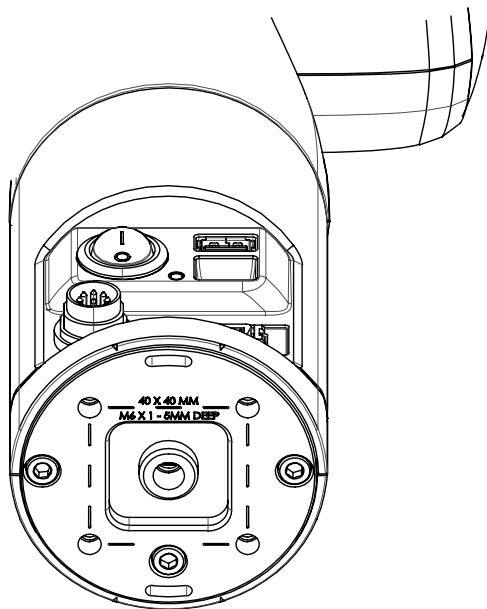
About this task

Affix the robot base to the surface using screws and sunk holes in the surface.

Procedure

1. Use the base bolting pattern as a guide to drill holes through the surface.

Since the base is to be mounted directly to the surface, and the holes on the robot base are on the underside, the holes in the surface must be drilled all the way through the mounting surface.

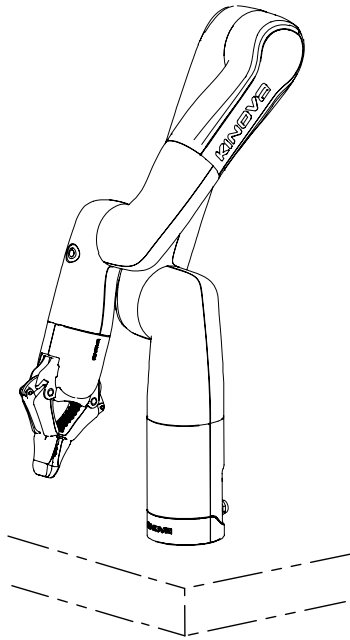


2. Use appropriate length M6 screws to mount the robot base controller or the mounting plate to the surface.



Tip: The screws must go through the mounting surface from the other side.

Results



What to do next

Connect the robot to the power supply.

Related topics

[Base underside bolting pattern](#) on page 24

[Powering on the robot](#) on page 39

Powering robot from a battery using a custom wiring harness

Provide power to the robot from a battery using a custom wiring harness.

The most straightforward way to supply power to the robot is using the supplied power adapter and cable to feed the robot from a nearby electrical outlet. This is the **recommended solution** for robot power supply **for the vast majority of users**.

In some applications, however, for example, for integration of the robot with a mobile platform, a wall outlet is not be feasible as a source of power, and users may want to supply the robot with power from a battery (24 V battery required).



Warning: The following information is intended for advanced users only.



Warning: Certification of the robot is under the assumption of power supply from a wall outlet using the supplied power adapter. Use of a battery as power source may result

in departure from these standards. If you have a wall outlet available, use it as a power source.



Warning: Powering the robot using a battery power supply is done at the user's risk, and Kinova is not responsible for any damage that might occur. We recommend that users contact Kinova technical support for any questions related to powering the robot with a battery source.

For those who want to explore this option and understand the risks involved, Kinova provides (sold separately) a power cable (p/n KR32222HAR) to connect the robot to a battery. The details are as follows:

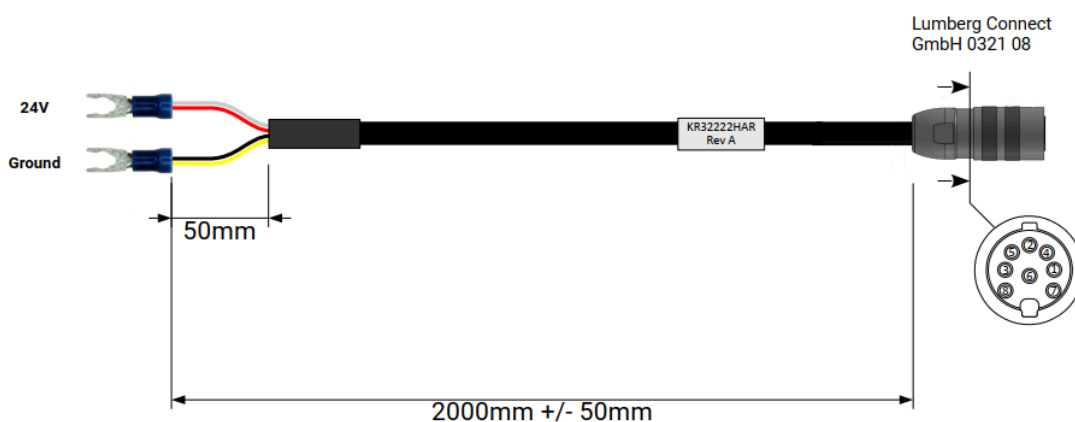


Figure 18: KR32222HAR power cable

Table 6: KR32222HAR cable pinout

#	Signal	Wire color
1	24V_INPUT	Red
2	24V_RETURN	Black
3	N/C	No connection
4	24V_RETURN	Yellow
5	N/C	No connection
6	N/C	No connection
7	24V_INPUT	White
8	N/C	No connection

Robot power adapter

The robot comes with a power adapter and cable to supply the robot with power from a wall outlet.

The robot is powered by a 24V power supply (Delta 150AAS24B).

The power adapter allows power to be supplied to the robot using a wall outlet as a source. The cable from the power adapter connects to the power connector on the base controller using a Lumberg 0321 08 connector.



Warning: When the power is cut, the robot descends. There are mechanisms within all the actuators to slow the fall of the arm for safety purposes. However, it is recommended that if possible, users cradle the robot as it falls.

E-stop (optional)

An optional emergency stop (E-stop) allows users to shut down the robot quickly in case of an emergency.

Integrated E-stop with power adapter cable

The E-stop option includes a push-button E-stop integrated in series with the power adapter.

To engage the E-stop, press down on the red button on top of the E-stop. This interrupts the power supply to the robot, causing it to shut it down.

To disengage the E-stop, rotate the button clockwise until it pops up.

Behavior of the robot after cut to power supply

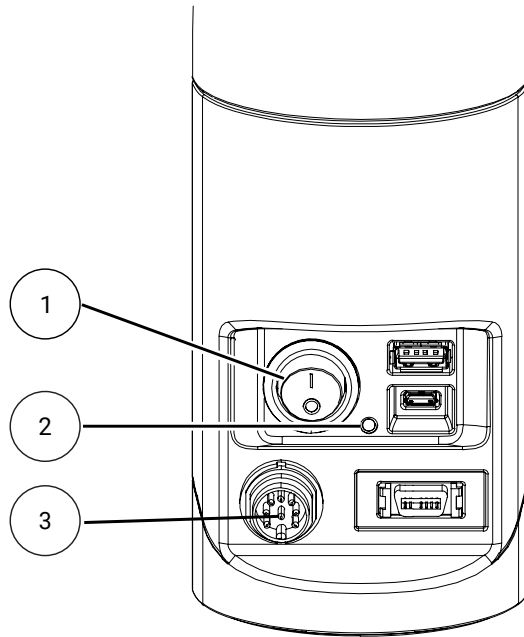
After the power supply to the robot or actuators is cut, the robot falls, gracefully, under its own weight.

Powering on the robot

After the robot is mounted and secured, connect the robot to an electrical power source.

About this task

Connect the robot to the power supply and use the power button on the base of the robot to switch the robot on and off.



1 Power on/off switch

2 Blue/Green/Red/Amber LED indicator

3 Power

Figure 19: Power button and indicators

Procedure

To switch on the robot



Note: When the robot has recently been powered down, wait a minimum of five seconds before attempting to power the robot back on.

1. Connect the captive cable from the power supply to the circular Lumberg connector on the rear connector panel in the base of the robot, rotating the outer cylindrical locking shell of the connector until it is just tight enough to secure the connector.
2. Plug the power supply into a wall outlet.

3. Move the power switch button to the ON position.



Note: When the robot is properly powered on, the status LED illuminates green.

The power up sequence is initiated.

To switch off the robot

4. Move the power switch button to the OFF position.

The robot initiates a shutdown procedure.

Power-up, booting, and initialization sequence

When the power button of the robot is pressed, the robot goes through a boot up and initialization sequence. The LED indications on the base give visual feedback during the power-up sequence.

When the power switch is pressed to initiate a power-up, the robot goes through a regular boot up and initialization sequence.

The base LEDs provide visual feedback of the progress through the sequence.

Table 7: Power-up sequence LEDs indications

Sequence step	LEDs indications
Robot booting	Blue LED
Robot ready for use	Green LED

From start to finish, the process should take no more than 30 seconds, except during a firmware update.

Operating the robot

There are different ways to control the robot.

To operate the robot, use a device or use software.

- Physical gamepad, the Xbox controller in particular
- Virtual joysticks over a network connection, such as Kinova® Kortex™ Web App with virtual joysticks
- Programmatically (Kinova® Kortex™ [Application Programming Interface \(API\)](#))

Supported control devices

The robot currently supports the Xbox gamepad. In particular, the Xbox 360, Xbox One, Xbox Series S, and Xbox Series X.

Connecting an Xbox gamepad to the robot, wired

Connect an Xbox gamepad to the robot with a cable to have a wired connection.

Before you begin

Materials needed

- Xbox gamepad
- Micro-USB B to USB type-A cable (included)

Procedure

1. Connect the Micro-USB B connector plug of the cable into the Micro-USB B port on the Xbox gamepad.
2. Connect the USB type-A end of the cable into the USB type-A connector on the base controller of the robot.

General control device mapping for the Xbox gamepad

Customize the mapping between the Xbox gamepad and the actions on the robot using the [API](#) or Web App.

Some controls apply the same across all maps.

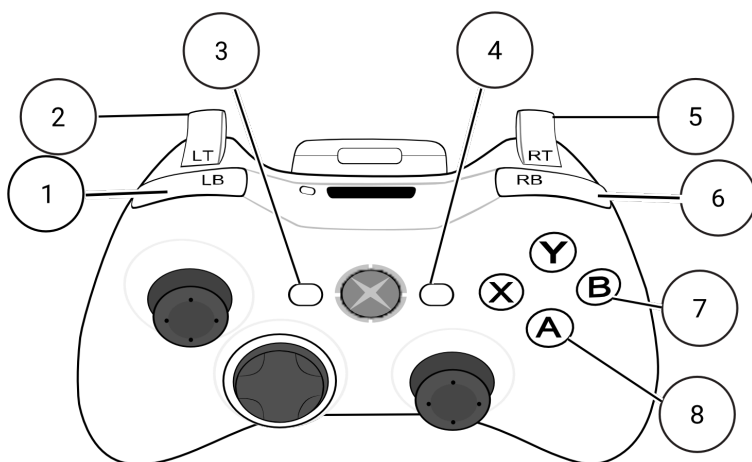
- Changing the active control [map](#) to the next or previous map in the list
- Opening and closing the gripper
- Clearing faults or E-Stop
 - A fault state or the triggering of an E-Stop signal makes itself known through a red [light-emitting diode \(LED\)](#) on the [base](#) controller of the robot.
 - The robot cannot be moved this if state is present.
 - Pressing the left bumper clears the fault (removes the E-Stop) and returns the LED to green, returning normal control.
- Applying emergency stop signal
 - Stops the robot
 - As with a fault, the LED changes to red and control of the robot is prevented until this state is cleared.
- Reaching home or retract position

The available control maps are in a sequential list, starting with **Twist linear** and ending with **Joint**. Pressing the **View** or **Menu** buttons causes the active control map to switch to the previous or next control map on the list. The list can be thought of as circular; selecting previous when on

the first map cycles around to the last map, and conversely, selecting the next map when on the last map cycles around to the first.

Table 8: General control map elements (common controls applying to all maps)

Action	Control		
Reach defined <i>pose</i>	Retract pose	A (hold down)	button
	Home pose	B (hold down)	
Navigate controller maps	previous	View button	
	next	Menu button	
Gripper command	close	Left	trigger
	open	Right	
Clear fault		Left	bumper
Stop robot		Right	



- | | |
|------------------------|------------------------------|
| 1 Clear fault | 5 Gripper open |
| 2 Gripper close | 6 Stop |
| 3 Previous control map | 7 Hold to reach Home pose |
| 4 Next control map | 8 Hold to reach Retract pose |

Figure 20: General control map elements with Xbox gamepad

The robot has three default control maps for the Xbox gamepad.

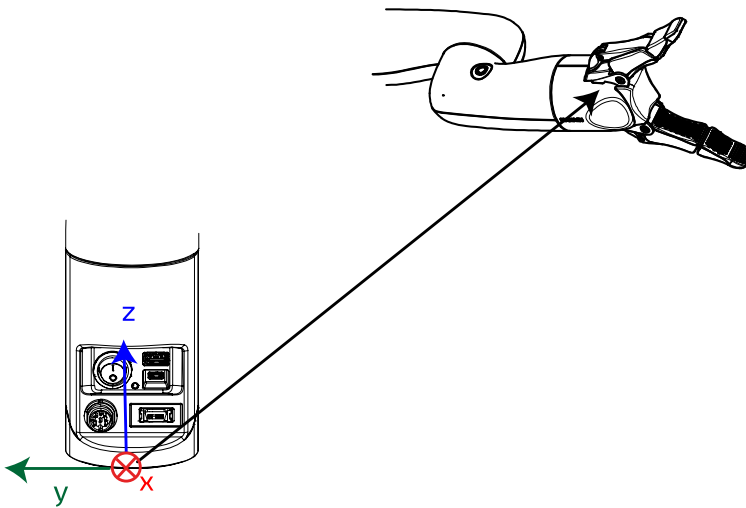
1. Twist linear controls the robot *end effector* translations by velocity.
2. Twist angular controls the robot end effector rotations by velocity.

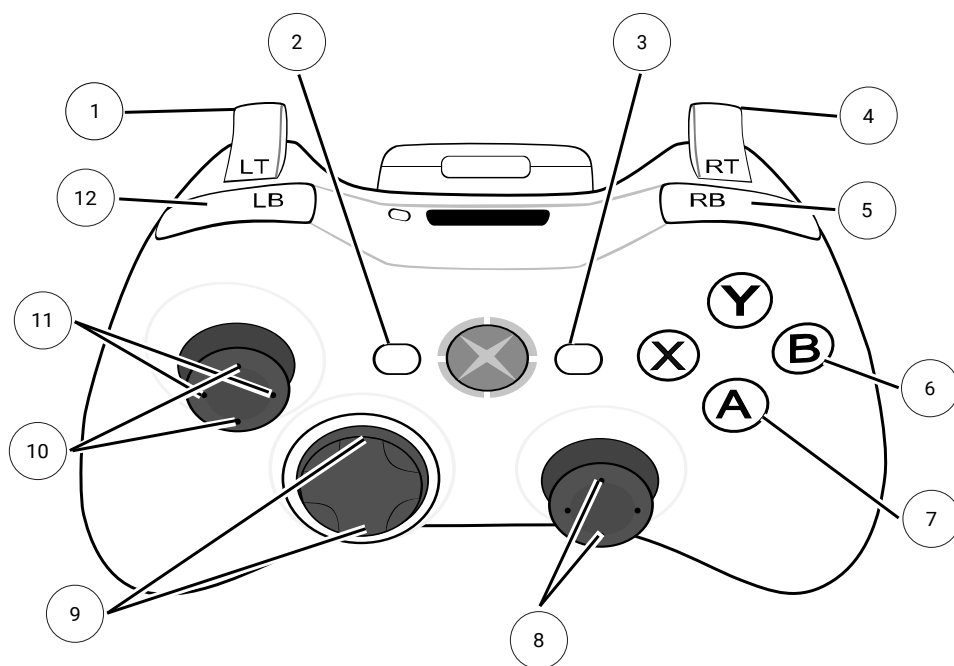
3. Joint controls the robot *joint* by joint by velocity

Twist linear map

Twist linear controls the robot end effector translations by velocity.

Twist linear is the default gamepad *map* when the robot is turned on and the controller is connected. In this mode the *tool* is translated in space with respect to the configured Cartesian translation *frame*; by default the *base frame*. The tool *orientation* does not change in this map. The user controls the linear velocity of the tool, including the linear speed.





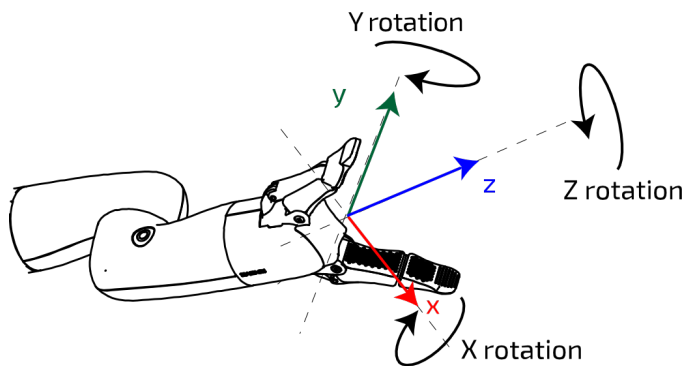
- | | | |
|------------------------|------------------------------|---------------------------------|
| 1 Previous control map | 5 Hold to reach home pose | 9 X translation |
| 2 Next control map | 6 Hold to reach retract pose | 10 Y translation |
| 3 Gripper open | 7 Z translation | 11 Clear fault
Remove E-stop |
| 4 Apply E-stop | 8 Speed +/- | 12 Gripper close |

Figure 21: Twist linear controls with Xbox gamepad

Twist angular map

The twist angular map is used to change the orientation of the gripper/tool without translating the gripper/tool. It controls the robot end effector rotations by velocity.

Twist angular can be thought of as a companion to the Twist linear control map. In Twist linear, the **tool** is translated with respect to the configured Cartesian translation frame while leaving the **orientation** unchanged. In Twist angular, the control is pure rotation of the tool within the configured Cartesian orientation frame (by default the tool reference **frame**), around the three axes of that frame. The user controls the angular velocity of the tool in relation to those three axes.



Twist linear and Twist angular together specify a twist (consisting of three linear velocity terms and three angular velocity terms) to be applied to the end effector (Cartesian control).

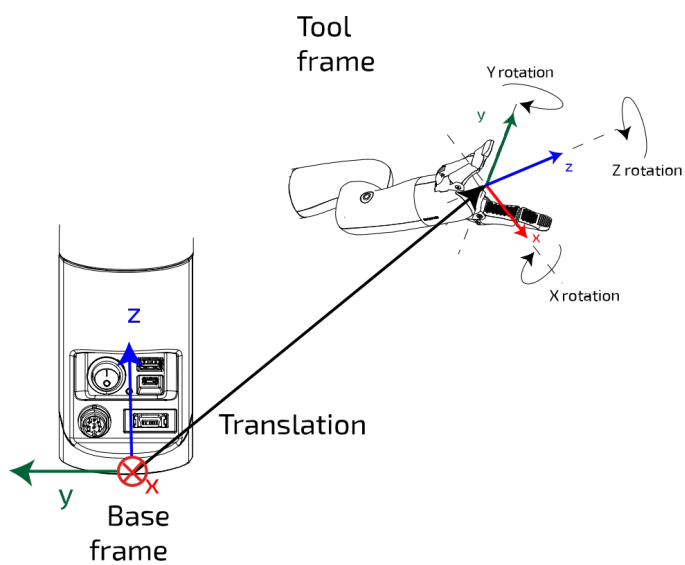


Table 9: Twist angular general controls plus

Action	Control		
Cartesian Y rotation	-	left	L stick
	+	right	
Cartesian X rotation	-	down	
	+	up	
Cartesian Z rotation	-	left	R stick
	+	right	
Speed	decrease	down	D-pad
	increase	up	

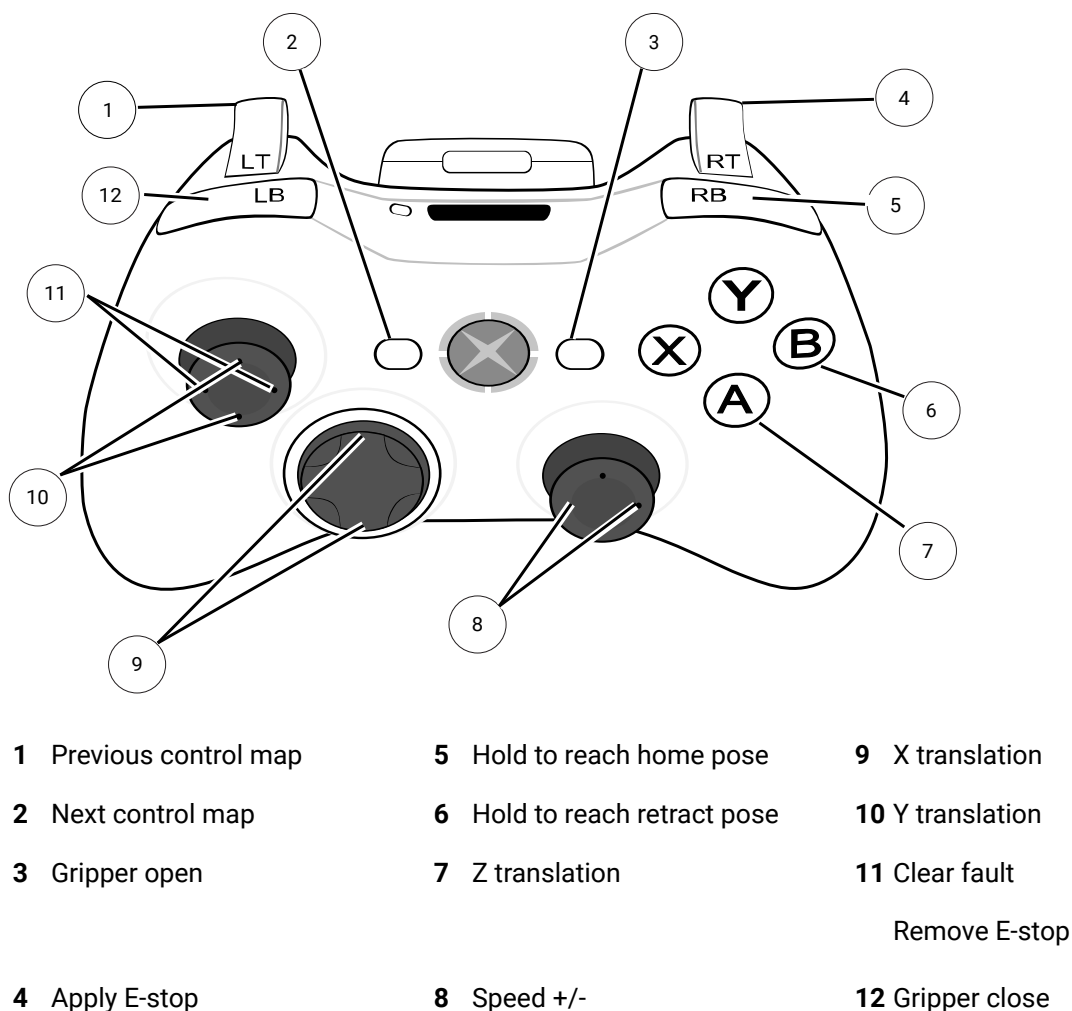


Figure 22: Twist angular controls with Xbox gamepad

Joint map

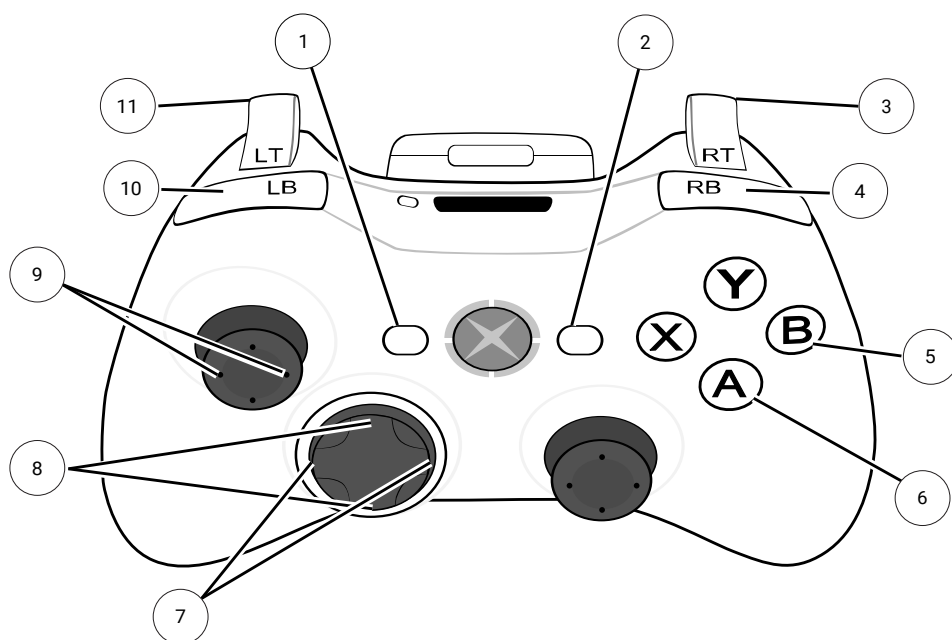
The joint map allows actuators to be chosen and moved one by one. Joint maps control the robot joints by velocity.

Joint control offers direct control of the rotational movement of the [joint](#) actuators. In this mode you can toggle through the joints (actuators) one by one, starting with the first and going through in increasing order. When it reaches the last actuator, it cycles back to the first. The joint angular speed (ω) can be controlled.

Table 10: Joint: general controls plus

Action		Control	
Joint speed	ω^-	left	L stick
	$>\omega^+$	right	

Action	Control		
Speed	increase	up	D-pad
	decrease	down	
Navigate joints	Previous	left	
	Next	right	



- | | | |
|------------------------|---------------------------|------------------|
| 1 Previous control map | 5 Reach home pose | 9 Joint speeds |
| 2 Next control map | 6 Reach retract pose | 10 Clear fault |
| | | Remove E-stop |
| 3 Gripper open | 7 Previous and next joint | 11 Gripper close |
| 4 Apply E-stop | 8 Speed +/- | |

Figure 23: Joint controls with Xbox gamepad

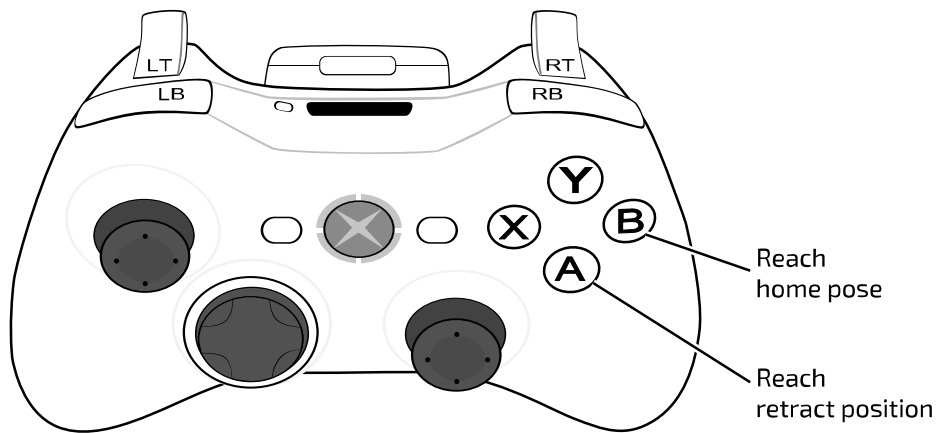
Home and retract positions

Preconfigured Xbox gamepad maps set aside two buttons to move the robot to the home and retract positions of the robot.

The robot includes two preconfigured *poses* that can be reached using the Xbox gamepad.

The Home position sets the robot into a convenient ready *position*. The home position is reached by holding down the **B** button on the Xbox gamepad.

The retract position folds the robot up into a compact pose. The retract position can be used when the robot is not being used. The retract position is reached by holding down the **A** button on the Xbox gamepad.



1 Reach home pose

2 Reach retract position

Figure 24: Reach home and retract controls

Connections between computer and robot

To control the robot through the software, a computer must be connected to it.

Establishing a communication link between the robot and a computer using USB (RNDIS)

Describes steps to connect the robot to a computer with a USB.

Before you begin

Gather a Micro-USB B to USB type-A cable, which is provided with the robot. Switch on the the robot.

About this task

The default method to connect the robot to a computer is to connect the micro USB of the robot base to a USB-A port on the computer. The DHCP server on the robot base is enabled by default and assigns an IP address to the computer RNDIS adapter to allow Ethernet over USB communication between the robot and the computer. No configuration is necessary to enable the communication. This is the most straightforward way to connect the robot to a computer, and provides simple point-to-point connectivity between robot and computer.



Note: Connection to a computer over a [LAN](#) is not possible.

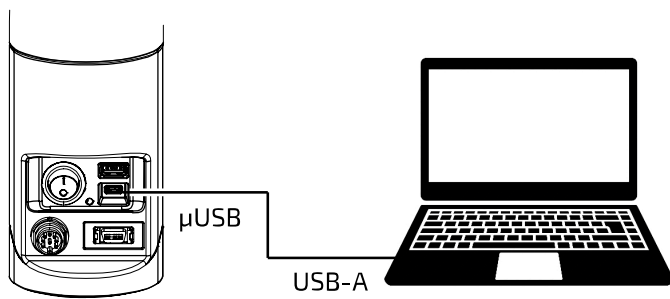


Note: It is assumed that the computer operating system is Windows 10. Details for other operating systems are different, but similar. For more details, see the documentation for the OS.

To enable connecting to a computer over a local area network, use an alternate method using a USB type-A to Ethernet adapter and an Ethernet (RJ-45) cable.

Procedure

1. Connect one end of the Micro-USB B to USB type-A cable to the Micro-USB B connector in the robot base.
2. Connect the other end of the cable to a USB type-A port on the computer.

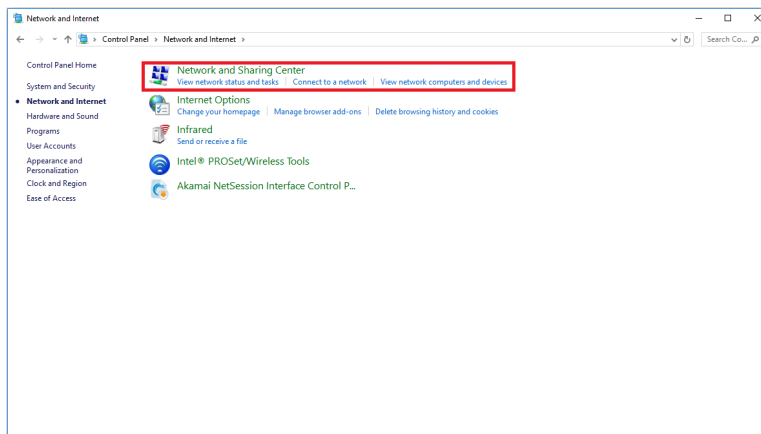
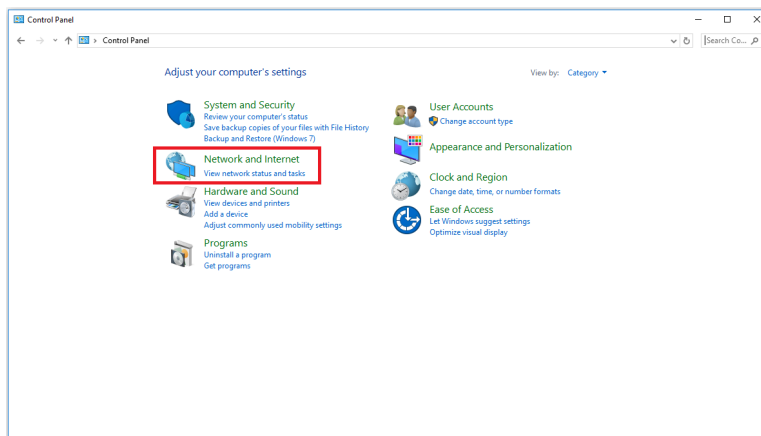


3. Confirm whether the connection is made by connecting to the robot with the Web App interface.

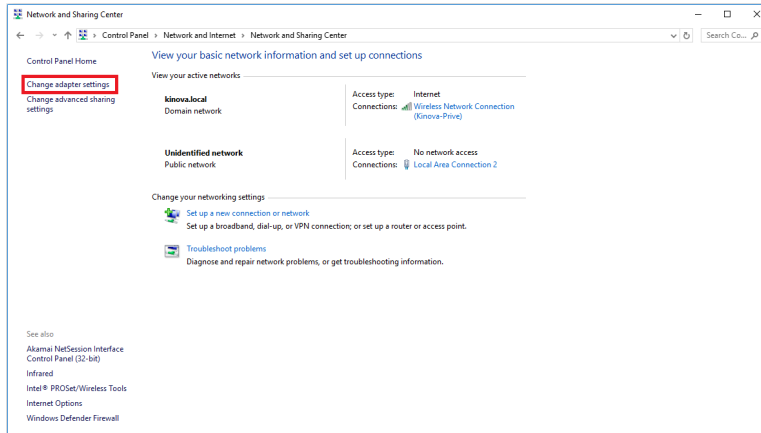
Option	Description
Connection with Web App works	Nothing more to do
Connection with Web App does not work	Manually configure the computer RNDIS adapter settings

Manually configure the computer RNDIS adapter settings

4. Open **Control Panel > Network and Internet > Network and Sharing Center** on the computer in Windows.



5. Select **Change adapter settings**.



6. Select the RNDIS adapter and choose **Properties**.

7. Select **Internet Protocol Version 4 (TCP/IPv4)** and choose **Properties**.

8. Select **Use the following IP address** and enter the IPv4 address: 192.168.1.11 and Subnet mask: 255.255.255.0

9. Press **OK**.

Results

The computer is physically connected to the robot and configured to communicate with the robot.

Related topics

[Connecting to a computer using a USB type-A to Ethernet adapter and Ethernet cable \(Optional\) on page 53](#)

[Accessing Kinova Kortex Web App on page 58](#)

Connecting to a computer using a USB type-A to Ethernet adapter and Ethernet cable (Optional)

Connect the robot to a computer using a USB type-A to Ethernet adapter and Ethernet cable when communication over a local area network is needed.

Before you begin

Material needed

- USB type-A to Ethernet adapter (optional item for robot packages)
- Ethernet (RJ-45) cable

About this task

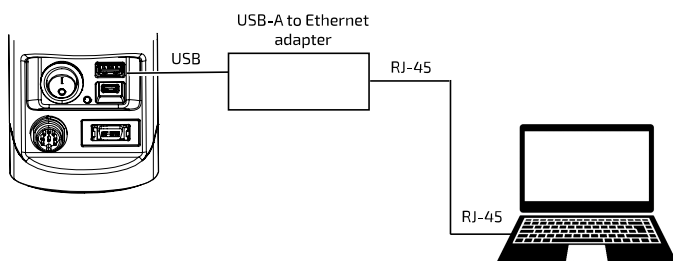
The default way to establish a communication link between the robot and a computer for communication and control is with a USB cable connection, using either a Micro-USB B to USB type-A cable using RNDIS or Ethernet over USB. The default way to establish communication does not work over [LAN](#).



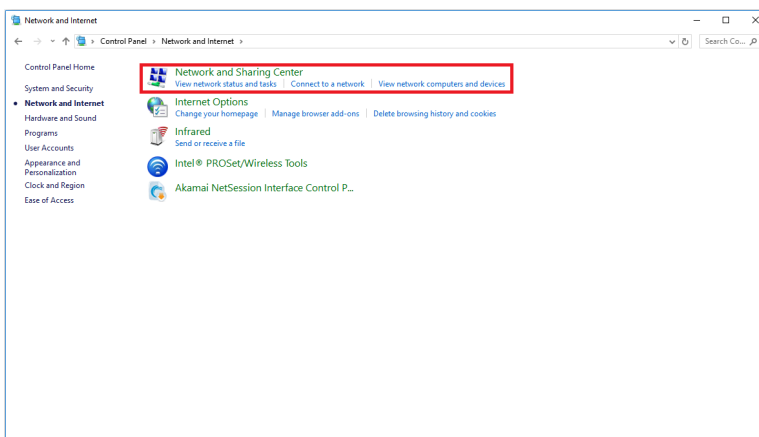
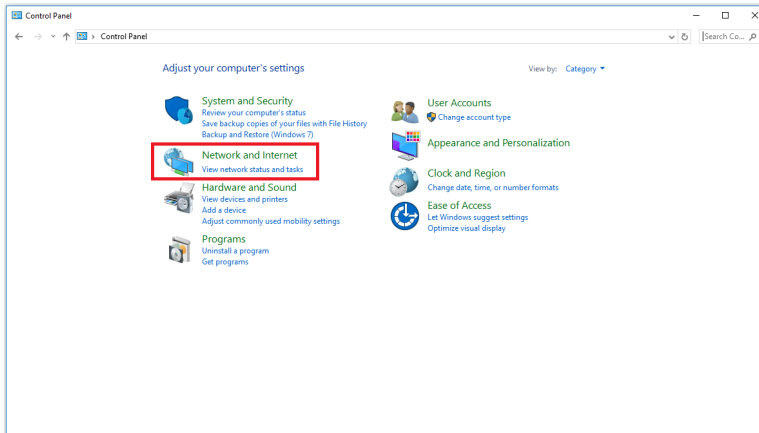
Note: The description and illustrations here are for Windows 10. The steps may follow a different order or interface for other operating systems.

Procedure

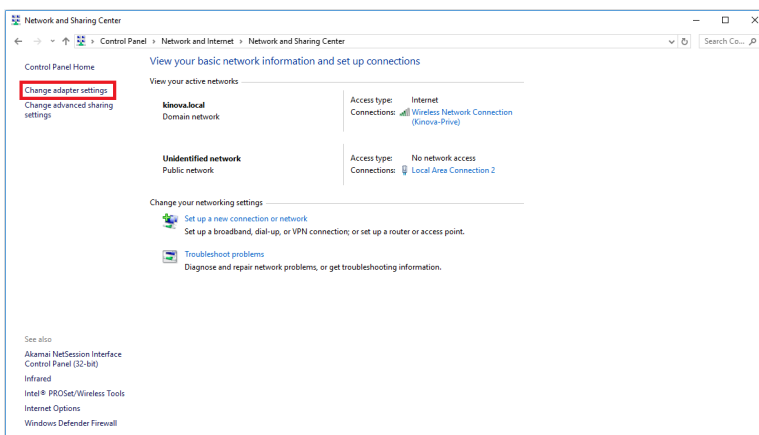
1. Connect the USB type-A to Ethernet adapter to the USB type-A port in the robot base.
2. Connect an RJ-45 Ethernet cable from the wired network adapter of the computer to the adapter Ethernet port.



3. Open **Control Panel > Network and Internet > Network and Sharing Center** on the computer.

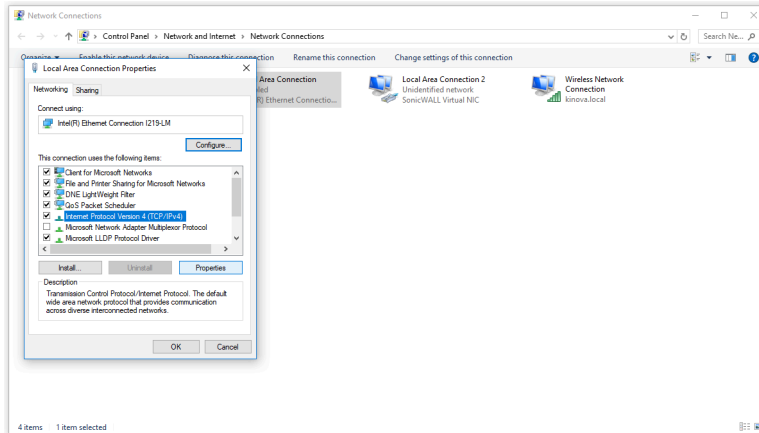


4. Select **Change adapter settings**.

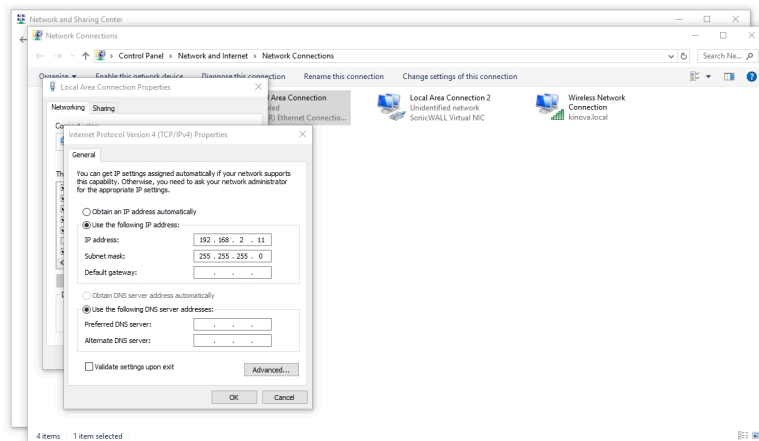


5. Select wired Ethernet adapter (that is, **Local Area Connection**) and choose **Properties**.

6. Select **Internet Protocol Version 4 (TCP/IPv4)** and choose **Properties**.



7. Select **Use the following IP address** and enter the IPv4 address: 192.168.2.11 and Subnet mask: 255.255.255.0



8. Press **OK**.

Results

The computer connected physically to the robot and configured to be ready to communicate with the robot.

What to do next

Connect the computer to the robot using Web App.

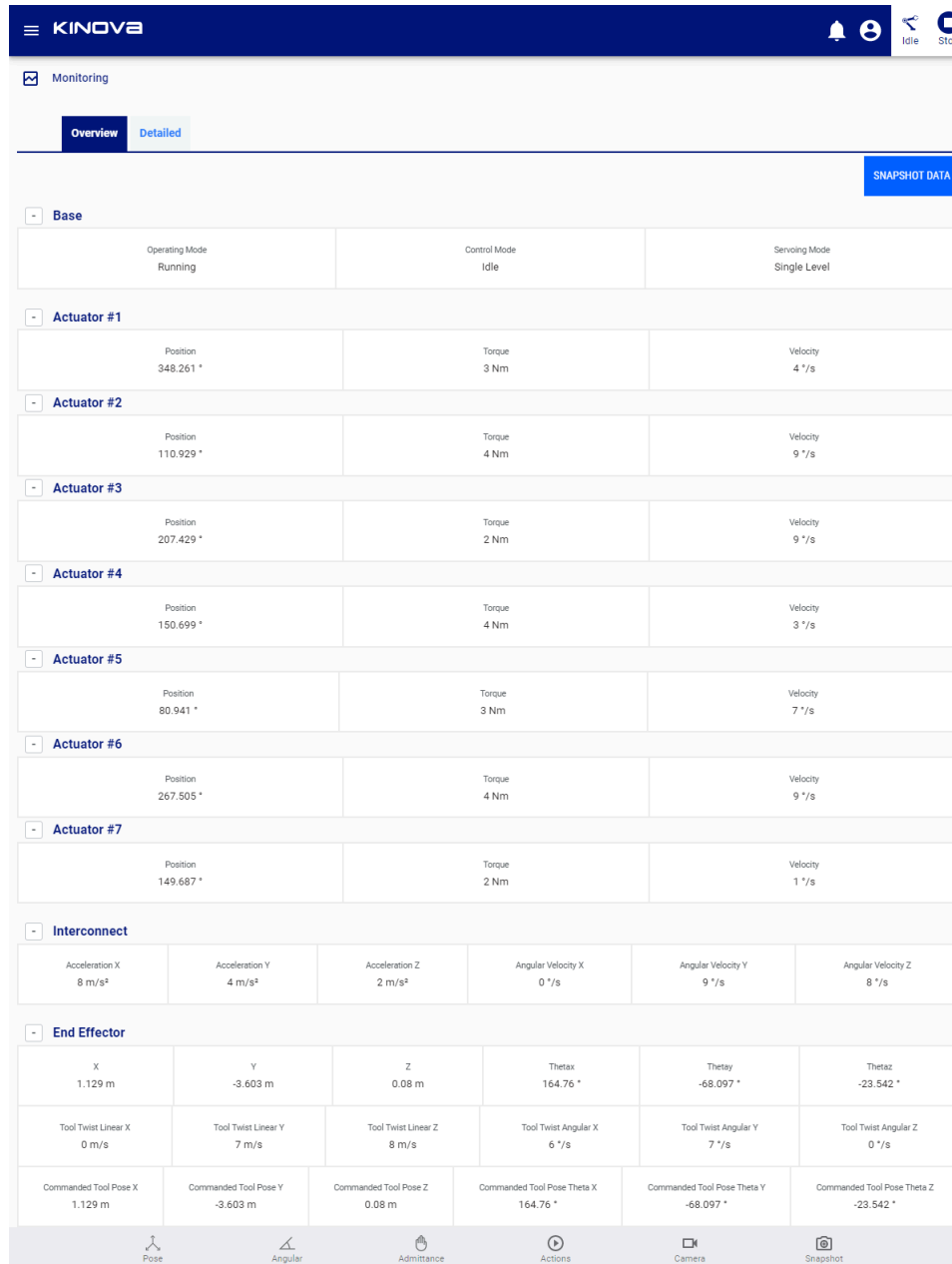
Related topics

[Accessing Kinova Kortex Web App](#) on page 58

Kinova® Kortex™ Web App

Use Kinova® Kortex™ Web App to interact with the robot and perform basic tasks without using programming commands.

Web App provides an HTML web browser based GUI to interact with the arm and perform basic tasks without using programming commands.



Web App allows users to control and configure the robot with the [Graphical User Interface \(GUI\)](#).

- Real-time control of the robot in different modes using different virtual joysticks
- Configuring

- robot performance parameters and safety thresholds
- protection zones
- network settings
- backup management
- [user profiles](#)
- Reading
 - system information
 - notifications
- Defining robot [poses](#) and [trajectories](#)
- Managing control [mappings](#) for physical controllers
- Monitoring robot parameters
- Upgrading the robot firmware

The Kinova® Kortex™ Web App can be run from a desktop or laptop [personal computer \(PC\)](#) connected by wired connection to the robot, or from any device on the same local network. Devices include desktops, laptops, smartphones, and tablets. The local network type includes local Wi-Fi networks.

Related topics

[Kinova Kortex Web App](#) on page 100

[Guidance for advanced users](#) on page 158

Accessing Kinova® Kortex™ Web App

Launch Web App from the computer and use it to control the robot.

Before you begin

Use a computer that is connected to the robot. Valid robot connections are directly wired, wired over a [LAN](#), or wirelessly. Obtain the [Internet Protocol \(IP\)](#) address of the robot on the network over which you are connected.

About this task

Procedure

1. Enter the appropriate IP address for the arm [base](#) from the computer web browser to access Web App.



Note: By default, the IP address to use here is:

- 192.168.1.10 if connecting point to point with [Universal Serial Bus \(USB\)](#) cable over [RNDIS](#)
- 192.168.2.10 if connecting over Ethernet cable using USB-type A to Ethernet adapter



Note: If the robot is configured with a different IP address so that the computer and robot are both connected to the same local area network, whether wired or over Wi-Fi, use the newly configured IP address.

Web App launches with a login window.

2. Enter the credentials.



Note: By default, the username-password pair is admin/admin.

3. Click **CONNECT**.

The application initializes. If all is successful, the application launches on the **Monitoring** screen that displays live parameters for the robot.

Monitoring

Overview

Download Data

Base

Operating Mode

Control Mode

Servoed State

Running

Idle

Single Joint

Actuators

Control

Position

Torque

Velocity

Actuator #1

Actuator #2

Actuator #3

Actuator #4

Actuator #5

Actuator #6

247.873°

168.823°

80.88°

82.227°

878.873°

84.881°

2.8Nm

2.8Nm

4.8Nm

2.8Nm

0.8Nm

2.8Nm

0°/s

0°/s

0°/s

2°/s

1°/s

7°/s

7°/s

2°/s

Interconnect

Acceleration X

Acceleration Y

Acceleration Z

Angular Velocity X

Angular Velocity Y

Angular Velocity Z

0 m/s²

0 m/s²

0 m/s²

4°/s

1°/s

0°/s

End Effector

X

Y

Z

Theta

Theta

Theta

Tool Pose Linear X

Tool Pose Linear Y

Tool Pose Linear Z

Tool Pose Angular X

Tool Pose Angular Y

Tool Pose Angular Z

Estimated Tool Pose X

Estimated Tool Pose Y

11.772 M

-12.223 M

0.143 M

151.881°

488.488°

180.084°

0.616

7.9616

0.7619

6°/s

8°/s

1°/s

11.772 M

-12.223 M

Estimated Tool Pose Z

Estimated Tool Pose Theta X

Estimated Tool Pose Theta Y

Estimated Tool Pose Theta Z

0.143 M

151.881°

488.488°

180.084°

Related topics

[Changing the robot wired connection IP address and connecting the robot to a LAN](#) on page 60

Changing the robot wired connection IP address and connecting the robot to a LAN

Change the wired connection IP address of the robot and connect the robot to a local area network.

Before you begin

An Ethernet cable, wired connection to the computer, as well as the [LAN](#) port of the computer, must already be established. Information about the available [IP](#) addresses on the [LAN](#) is need to connect the robot.

About this task

Configure the robot to connect a computer to the robot remotely over a local area network.



Note: For security reasons, we do not recommend connecting the robot to a WAN. The network should be a simple local area network with low traffic.



Note: Connecting to the robot over a network is only available under the [USB-A](#) to Ethernet adapter and Ethernet cable connection option. The [RNDIS](#)/Ethernet over USB option is only for point-to-point connection between the robot and a single computer.

Procedure

1. Launch Web App and navigate to the **Networks** page.
2. Click **USB type-A Dongle**.
3. Modify the IPv4 address, IPv4 subnet mask, and IPv4 gateway to match an available IP address with the IP address range of the network.



Note: Once the robot network parameters are modified, the client computer loses connection with the robotic arm.

4. Physically disconnect the robot from the computer and connect it with an Ethernet cable to the LAN at a network switch.
5. Restore IP settings compatible with the LAN on the computer local network adapter and connect the computer physically to the LAN.
6. Use the computer to ping the robot at its newly configured robot IP address to confirm that communication is established.

What to do next

From the computer web browser, enter the new robot IP address to access Web App.

Related topics

[Connecting to a computer using a USB type-A to Ethernet adapter and Ethernet cable \(Optional\)](#)
on page 53

Connecting a computer to the robot with Wi-Fi

It is possible to connect the computer to the robot over Wi-Fi without any wires.

Before you begin

A wired connection between the computer and robot must already be established. The Wi-Fi router must be set to broadcast its [Service Set Identifier \(SSID\)](#) so that the robot can find the network. The router must also be configured to accept new devices. Get the Wi-Fi password.

About this task

The robot features an integrated Wi-Fi adapter to connect to a local Wi-Fi connection. When the connection is established, other devices on the same Wi-Fi network can also connect to the robot wirelessly with the [IP](#) address assigned for the robot on the wireless network.

Procedure

1. Launch Web App on a computer connected to the robot with wired Ethernet and connect to the robot.
2. Select the **Configurations** page group and select **Wireless & Networks** in the main navigation panel of Web App to go to the **Wireless & Networks** page.
3. Select the **Wi-Fi** tab.
The **Wi-Fi** page lists all of the detected Wi-Fi networks.
4. Choose one of the networks and click the corresponding **Connect** text button.



Note: It is not recommended to connect to Wi-Fi networks that are potentially insecure. Minimum security settings of WPA2 are recommended.

A pop-up window launches to sign in to the network, with information about the signal strength and security settings.

5. Enter the password for the network and click the **CONNECT** button.



Note: A **Connect Automatically** toggle is available here. When it is toggled on, the robot [base](#) connects automatically to the wireless network using the previously entered password whenever the robot is powered on.

6. Take note of the wireless network IPv4 address that the robot obtains after clicking the **CONNECT** button.

7. On any wireless device (laptop, tablet, smartphone) connected to the same Wi-Fi network, open a web browser and type the IP address that the robot obtained at [6](#) on page 62.



Note: The address corresponds to the address of the robot on the Wi-Fi network.

The Web App login screen launches.

8. Enter the appropriate user name and password, and click **CONNECT**.

Results

Web App is connected to the robot through the Wi-Fi network adapter of the robot. Configure, monitor, and control the robot wirelessly with either Web App or with [API](#) control.



Restriction: A Wi-Fi connection is not recommended for 1 kHz (low-level) control of the robot due to potential latency issues; a wired connection must be used for this purpose.

Dimensions, specifications, and capabilities

Before using the Gen3 lite robot, become familiar with its size, shape, and weight.

Schematic and dimensions

A schematic diagram of the robot and the key physical dimensions of the robot help plan where to install the robot.

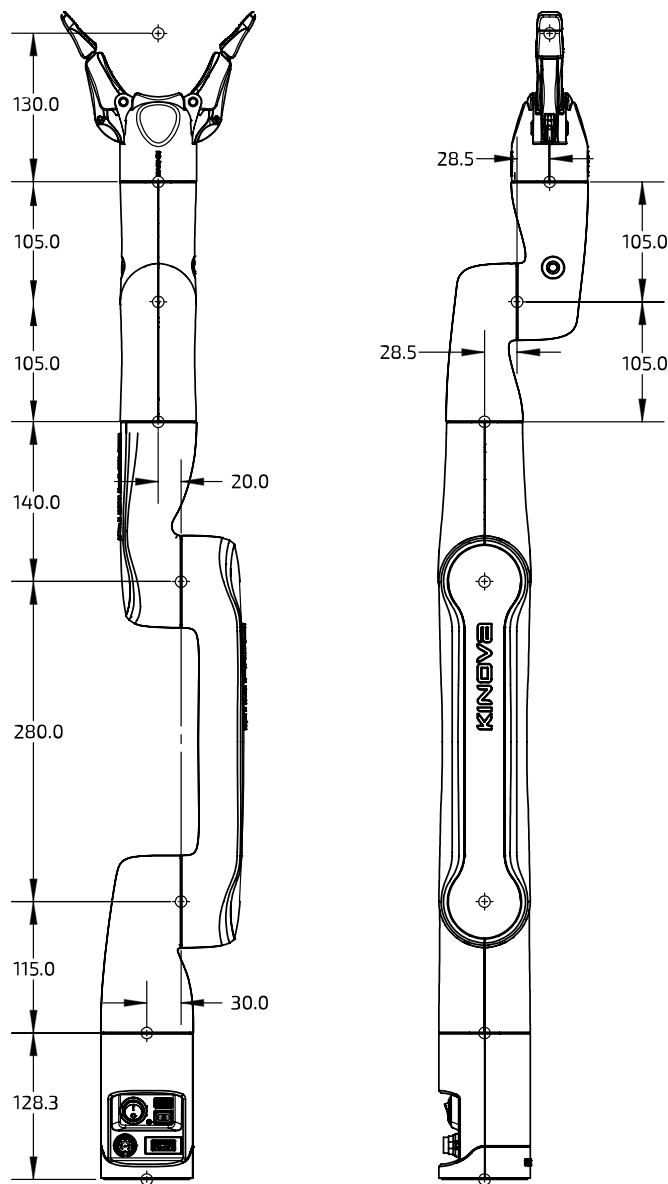


Figure 25: Robot dimensions (measurements in mm)

The maximum reach of the robot, as defined by the distance from the shoulder (Actuator 2 frame) to the middle of the gripper, is 76.0 cm.

Table 11: Robot geometric parameters

Description	Length (mm)
Base to actuator 1	128.3
Base to shoulder	243.3
Upper arm length	280.0
Forearm length (elbow to wrist)	140.0
First wrist length	105.0
Second wrist length	105.0
Last actuator to gripper center	130.0
Joint 1-2 offset	30.0
Joint 2-3 offset	0.0
Joint 3-4 offset	20.0
Joint 4-5 offset	28.5
Joint 5-6 offset	28.5

Technical Specifications

Technical specifications for the Kinova® Gen3 lite robot are categorized for ease of reference. Some of these also appear within the main body of the text.

Table 12: Safety/Security

Feature	Detail
Power monitoring	current and voltage monitoring
Position monitoring	user defined protection zones, joint limits
Thermal monitoring	warning/shutdown near maximum core and motor temperatures

Table 13: Environmental

Parameter	Value(s)
Temperature	-25°C to 40°C (operating)
	0°C to 40°C (startup)
Robot ingress protection	IP22
Relative humidity (non-condensing)	15 to 90%
Pressure	70 to 106.0 kPa
Sound pressure level (nominal)	< 60 dBa
Power supply input voltage	90: 264 VAC

Table 14: Controller (base)

Feature	Detail
LED indicator	blue/green/amber/red LED
USB type-A receptacle	For charging small devices such as phones, connecting gamepad wired, connecting USB-A to Ethernet adapter
Micro-USB B receptacle	Ethernet over USB (RNDIS)
Wi-Fi (IEEE 802.11a/b/g/n)	Kinova® Kortex™ Web App
14-pin joystick/expansion connector	joystick*, discrete I/O*, emergency stop, expansion
circular connector [Lumberg 0317 08]	power
piezo-electric buzzer	configurable for audio warnings and alarms*
sensors	gyroscope and accelerometer
Web server	to support Web App
DHCP server	simplified (no configuration) connectivity with computer

Table 15: Robot

Parameter	Value(s)
Weight	5.4 kg
Payload	600 g (mid-range, continuous)

Parameter	Value(s)
	500 g (full-range, continuous)
Maximum reach (fully extended)	760 mm (to gripper center reference)
Maximum Cartesian translation speed	low-level: 30 cm/s
	high-level: 25 cm/s
Degrees of freedom	6
Actuators	small: actuators 4, 5, 6
	large: actuator 2
	medium: actuators 1, 3
Power supply voltage	21 to 30 VDC (24 VDC nominal)
Robot power usage	20 W (average)
	100 W (peak)
Materials	Aluminum
	Polycarbonate, reinforced nylon
Internal communications and control	Ethernet (100 Mbps)

Table 16: Actuators

Feature	Value(s)
Sensors	Position, current (motor), voltage, temperature (motor and CPU)

Table 17: Wrist and gripper

Feature	Function
Gripper	2 finger gripper actuated by one linear actuator
Sensors	accelerometer and gyroscope, voltage (motor and microcontroller), temperature (motor and microcontroller), motor current, finger position and velocity

Table 18: Software/control

Feature	Detail
Low-level control	position, velocity, current
High-level control	Cartesian twist (linear and angular velocity), joint speed, Cartesian and angular trajectory
High-level control features (mode dependent)	protection zones: rectangular, spherical, cylindrical singularity handling
Servoing modes	High level, low-level
Supported ROS distribution	Kinetic Kame, Melodic Morenia
Boot time	30 seconds
Internal communication frequency	1 kHz

Sensors

The robot contains a number of different sensors. There are sensors in the base, actuators, and wrist and gripper. Sensors data can be accessed using the cyclic communications with the base.

The robot components contain the different types of sensors.

Base sensors	Voltage
	Current
	Temperature
	6-axis accelerometer/gyroscope
Actuator sensors	Motor current
	Motor temperature
	CPU temperature
	Angular Hall sensors
	Magnetic angular sensor
	24 V monitor
Wrist and gripper sensors	Capacitive sensors (qty. 2)
	Microcontroller temperature
	Microcontroller board voltage

6-axis accelerometer/gyroscope

Gripper motor current

Gripper motor temperature

Gripper motor voltage

Gripper finger position and velocity from gripper motor

Users can read data from some sensors by using the [APIs](#) or through the **Monitoring** page of Web App.

The API method `RefreshFeedback()` in the `BaseCyclic` API returns a data structure with readings from sensors.

- Base
- Actuators
- Wrist and gripper
 - Wrist
 - Gripper pose and twist
 - Gripper finger motors



For detailed information on how to unpack this data in an application, see the `BaseCyclic` API documentation.

Readings of the base that are available

Base sensor readings available through cyclic communications.

Table 19: Base readings available through API

Field name	Description
<code>arm_voltage</code>	arm voltage in V
<code>temperature_cpu</code>	CPU temperature in °C
<code>temperature_ambient</code>	ambient temperature in °C
<code>imu_acceleration_x</code>	IMU measured acceleration (X-axis) of base in m/s^2
<code>imu_acceleration_y</code>	IMU measured acceleration (Y-axis) of base in m/s^2

Field name	Description
imu_acceleration_z	IMU measured acceleration (Z-axis) of base in m/s^2
imu_angular_velocity_x	IMU measured angular velocity (X-axis) ω_x of base in $^\circ/\text{s}$
imu_angular_velocity_y	IMU measured angular velocity (Y-axis) ω_y of base in $^\circ/\text{s}$
imu_angular_velocity_z	IMU measured angular velocity (Z-axis) ω_z of base in $^\circ/\text{s}$

Readings of the actuators that are available

Actuators sensors readings available through cyclic communications.

Table 20: Actuators readings available through the API

Field name	Description
position	angular position of the actuator in $^\circ$
velocity	angular velocity of the actuator in $^\circ/\text{s}$
torque	Approximate equivalent torque derived from motor current in $\text{N}\cdot\text{m}$
current_motor	motor current in A
voltage	main board voltage in V
temperature_motor	actuator motor temperature in $^\circ\text{C}$ (one of the three phases)
temperature_core	microcontroller temperature in $^\circ\text{C}$

Readings of the wrist that are available

Wrist sensors readings available through cyclic communications.

Table 21: Wrist readings available through the API

Field name	Description
imu_acceleration_x	IMU Measured acceleration (X-axis) of the gripper. (m/s^2)

Field name	Description
imu_acceleration_y	IMU Measured acceleration (Y-axis) of the gripper. (m/s ²)
imu_acceleration_z	IMU Measured acceleration (Z-axis) of the gripper. (m/s ²)
imu_angular_velocity_x	IMU Measured angular velocity (X-axis) of the gripper. (°/s)
imu_angular_velocity_y	IMU Measured angular velocity (Y-axis) of the gripper. (°/s)
imu_angular_velocity_z	IMU Measured angular velocity (Z-axis) of the gripper. (°/s)
voltage	Voltage of the main board in V
temperature_core	Microcontroller temperature. (°C)

Readings of the gripper pose and twist that are available

Gripper pose and twist sensors readings available through cyclic communications.

Table 22: Gripper readings available through the API

Field name	Description
tool_pose_x	Measured Cartesian position (X-axis) of the gripper in m
tool_pose_y	Measured Cartesian position (Y-axis) of the gripper in m
tool_pose_z	Measured Cartesian position (Z-axis) of the gripper in m
tool_pose_theta_x	Measured Cartesian orientation (X-axis) of the gripper in °
tool_pose_theta_y	Measured Cartesian orientation (Y-axis) of the gripper in °
tool_pose_theta_z	Measured Cartesian orientation (Z-axis) of the gripper in °
tool_twist_linear_x	Measured Cartesian linear velocity (X-axis) of the gripper in m/s

Field name	Description
<code>tool_twist_linear_y</code>	Measured Cartesian linear velocity (Y-axis) of the gripper in m/s
<code>tool_twist_linear_z</code>	Measured Cartesian linear velocity (Z-axis) of the gripper in m/s
<code>tool_twist_angular_x</code>	Measured Cartesian angular velocity (X-axis) of the gripper in °/s
<code>tool_twist_angular_y</code>	Measured Cartesian angular velocity (Y-axis) of the gripper in °/s
<code>tool_twist_angular_z</code>	Measured Cartesian angular velocity (Z-axis) of the gripper in °/s

Readings of the motor for the gripper fingers that are available

Gripper finger motor sensors readings available through cyclic communications.

Table 23: Finger readings available through the API

Field name	Description
<code>position</code>	Position of the gripper fingers in percentage (0-100%)
<code>velocity</code>	Velocity of the gripper fingers in percentage (0-100%)
<code>current_motor</code>	Current consumed by the gripper motor (mA)
<code>voltage</code>	Motor Voltage (V)
<code>temperature_motor</code>	Motor temperature. (°/s)

Effective workspace

The effective workspace is the region in space reachable by the gripper of the robot.

The effective workspace refers to the region in three-dimensional space which is reachable by the robot end effector. It is impacted by several factors, including the number and length of the links, the joint ranges, and the shape of the links

There are two definitions of effective workspace, the first being larger than the second.

Payload vs. workspace

The payload of the robot is the maximum mass that the robot can hold up at the end effector.

Payload is generally not one constant figure, but depends on a few factors.

- Radial distance from the base: the payload is highest when it is closest to the base, and decreases as the end effector gets farther out from the base axis.
- Temporary vs. continuous: the robot has a maximum payload that can be handled temporarily for a short period of time. However, continued use of the arm with that payload for an indefinite period causes the arm to heat up, as the heat generated by the strain on the actuator exceeds the rate at which heat can be dissipated. However, a smaller mass can be handled for an indefinite period. The smaller mass is referred to as the continuous payload limit.

Examples of acceptable payloads:

- 500 mL bottle of water
- Cup
- Tool, such as a screwdriver, fork, spoon, and so on
- Phone

If the maximum payload is exceeded, one or more symptoms may appear.

- Arm power cut and drop
- Arm low voltage
- Actuator following error
- Actuator fault (motor driver fault)
- Hall sequence error

Robot control

Control of the Gen3 lite robot includes high-level and low-level controls, control features, control methods, and control modes. Control features include collision avoidance, protection zones, and limits of movement.

High-level and low-level robot control

The robot can be controlled in either high-level or low-level robot control. There are benefits to both forms of control. High-level control is easier with more protections, while low-level control offers faster commands and finer-grained control with less protection.

High-level is the default on boot-up and offers the safest and most straight-forward control.

In both high-level and low-level, commands are sent through the robot [base](#).

In high-level control, commands are sent to the base through a single command using the `Kinova.Api.Base` [API](#). These commands are processed by Kinova robot control libraries.

The robot control library applies high-level control features like:

- [Singularity avoidance](#)
- [Protection zones](#)
- Cartesian and joint limits
- Configurable speed and acceleration limits

The control library also breaks down the command into smaller commands that the base sends incrementally to the individual robot actuators through its 1 kHz communication loop with the actuators.

In low-level control, the user sends a series of small commands to each actuator and the gripper as part of a user-defined loop, at a maximum rate of 1 kHz, using the `Kinova.Api.BaseCyclic` [API](#). The base receives these commands and routes them to the appropriate actuators and the gripper through its own 1 KHz communication loop with robot devices.

High-level control is simpler to use and offers added protections. However, it is slower because the overhead of processing by the Kinova control library. High-level control runs at 40 Hz.

Low-level control offers faster commands and finer-grained control of the robot. The commands of the user can be applied directly to the actuators without being changed by the high-level control features, which is useful, for example, when implementing a controller external to the

robot. The user should be aware that the protection offered by the high-level control libraries is not be applied as a result.

Related topics

[API for developers](#) on page 142

[Robot servoing modes](#) on page 147

Control features

The robot has control features that improve the safety and usability of the robot, and protect it from damage.

Each control feature is discussed in greater detail.

- [Singularity avoidance](#)
- [Protection zones](#)
- Angular limits
- Cartesian limits
- Configurable kinematic limits

Singularity avoidance

The singularity avoidance feature of the robot control library handles or avoids singularities in Cartesian [control modes](#). The robot behavior, including tool speed may be altered near a singularity.

A singularity refers to any robot configuration (set of [joint angles](#)) that causes the Jacobian matrix relating actuator rotation speed to [end effector](#) velocities to be ill-conditioned, thus rendering the solution mathematically unstable. The ill-condition comes from the determinant approaching zero or the matrix losing rank.

At a singularity, the mobility of the robot is reduced, meaning the arbitrary motion of the manipulator in a Cartesian direction is lost (losing a degree of freedom). This occurs when two or more robot axes become colinear, leading to unpredictable and potentially extreme [joint](#) velocities when trying to attain a certain Cartesian [pose](#). For example, when two axes become colinear in space, rotation of one can be canceled by counter-rotation of the other, leaving the actual joint location indeterminate. Near a singularity a small linear end effector motion requires disproportionately large angular velocities of the actuators.



Note: The robot controller firmware features capabilities to handle and avoid singularities in any Cartesian mode. Because a singularity cannot occur unless inverse kinematics are calculated, singularities do not occur in any of the joint modes.



Note: The robot behavior may change somewhat at or near a singularity. For example, the [tool](#) speed may be reduced or the motion may deviate from the commanded motion.



Note: The robot has [singularity avoidance](#) algorithms. However, the robot cannot use a [Cartesian mode](#) to move if the motion were to start in a singular pose. It must use a Joint [angular mode](#) to exit the singularity.

Related topics

[Singular configurations](#) on page 163

Protection zones

Users can define geometric protection zones where the movement of checkpoints on the robot is either prevented. These can be defined using either the Kinova® Kortex™ Web App or developer [API](#). [Overview](#)

The user defines [protection zones](#) programmatically or by using Web App, based on a few basic geometric shapes. The user can specify a speed limitation in the envelope of defined thickness surrounding each protection zone.

One or more protection zones can be configured to define geometric volumes about the robot [base](#), where the motion of the robot [end effector](#) is either limited or precluded.

By defining suitable protection zones, the robot can be set to prevent collisions with known fixed obstacles in the immediate environment of the robot while in operation.



Note: Protection zones are active only in Cartesian [control modes](#). These protections are not available in angular control modes.



Note: If multiple protection zones are used, we recommend that the same envelope speed limitations be used for each.



Note: There is no hard limit on the number of protection zones that can be simultaneously active, but for best results it is recommended to activate no more than four protection zones at one time.



Note: Envelope-only protection zones, where all the dimensions of the protection zone are set to zero but the envelope thickness is set to a non-zero value are NOT supported.

Robot tool behavior

The [tool](#) of the robot never enters protection zones. If the robot is commanded to stop in or pass through a protection zone and if it is configured, any motion of the tool towards the inside of the

protection zone stops at the outer boundary of the protection zone. The tool slides on the outer surface of the zone, but not enter inside the zone.

The tool can move in the surrounding envelope, but at a reduced speed.

Checkpoints and behavior of checkpoints



Warning: The reference frame is centered on actuators. The checkpoint is on the reference frame. Configure protection zones with additional space to take into account the physical dimensions of the actuators.

Additional checkpoints are used for protection zones and are defined for the robot at the centers of actuators 3-6 and near the center of actuator 2.

The motion stops at the outer surface of the protection zone for checkpoints. Checkpoints move in the envelope surrounding a protection zone, but at a reduced speed.

Protection zone shapes

Protection zones can be defined using one of three basic shape types.

Protection zone shape	Description
Rectangular prism	<i>Position</i> of center, length, width, and height dimensions, and angular <i>orientation</i> of the rectangular prism are configurable
Cylindrical	Position of center, radius, height, and angular orientation of the cylinder are configurable
Spherical	Position of center and radius of sphere are configurable

A planar or disc-shaped protection zone can be defined by setting the thickness of the zone to zero in either a rectangular prism or cylindrical protection zone.

Editing protection zones

Protection zones can be defined, edited, and deleted using either Web App or the developer [API](#).

Joint limits

The robot has joint limits used in robot high-level control, which impacts the position, speed, acceleration, and torque applied by the joints.

When controlling the robot in high-level, the robot control library applies a number of different [joint](#) limits for safety purposes.

- Joint position
- Joint speed
- Joint acceleration
- Joint torques

The limits applied in a particular situation depend on the current high-level control mode.

Table 24: Joint position limits

Actuator	Joint limits	
	Lower limit	Upper limit
1	- 154.1°	+ 154.1°
2	- 150.1°	+ 150.1°
3	- 150.1°	+ 150.1°
4	- 148.98°	+ 148.98°
5	- 144.97°	+ 145.0°
6	- 148.98°	+ 148.98°

Table 25: Joint speed limits

Actuator	Limit (magnitude)
joints 1-5	57.3 °/s (1.0 rad/s)
joint 6	90.0 °/s (1.57 rad/s)

Table 26: Joint acceleration hard limits

Actuator	Limit (magnitude)
joint 1	57.3 °/s ² (1.0 rad/s ²)
joint 2	28.65 °/s ² (0.5 rad/s ²)
joint 3	22.92 °/s ² (0.5 rad/s ²)

Actuator	Limit (magnitude)
joint 4	$57.3 \text{ }^\circ/\text{s}^2$ (1.0 rad/s^2)
joints 5 - 6	$572.95 \text{ }^\circ/\text{s}^2$ (10.0 rad/s^2)

Table 27: Joint torques soft limits

Actuator	Limit (magnitude)
joints 1 - 6	9.0 N·m

Table 28: Control modes and relevant joint limits

Control mode	Joint limits applied
Cartesian trajectory	joint position, joint speed
angular trajectory	joint position, joint speed, joint acceleration, joint torque (for pre-computed trajectories)
Cartesian joystick	joint position, joint speed
angular joystick	joint position, joint speed, joint acceleration

Table 29: Behavior at joint limits

Limit	Behavior when limit is reached
joint acceleration	joint acceleration reaches the maximum acceleration limit
joint speed	acceleration go to zero and joint speed reaches the maximum at speed limit
joint position	joint acceleration and joint speed goes to zero and motion of the joint stops at the position limit

Cartesian limits

The robot has limits on tool motion applying in high-level Cartesian control. These limits apply on top of underlying joint limits.

Cartesian limits on the motion of the *tool* in *Cartesian modes* (these limits do not apply in *angular modes*) are applied as follows:

- In Cartesian joystick, the magnitude (*vector* norm) of the linear and rotational speeds of the tool are capped. *Twist* commands whose linear and/or rotation speed exceed these limits are

be re-scaled proportionally so that the **magnitudes** of tool linear and angular speeds do not exceed the limits, but the commanded **directions** of tool movement are respected.

- For all Cartesian modes, the individual joint speeds are capped at their limits. The speed of all [joints](#) are scaled proportionally.
 - No individual joint exceeds the speed limit
 - The desired direction of motion is respected
- For all Cartesian modes, joint positions are capped at their limits.

Table 30: Gen3 lite robot Cartesian limitations

Limit		Value
twist limits	linear	0.25 m/s
	angular	45.8 °/s (0.80 rad/s)

Related topics

[Joint limits](#) on page 79

High-level and low-level robot control methods reference

Control robot movement by using key methods in the [API](#). For the full [API](#) documentation and code examples, please see the Kinova® Kortex™ GitHub repository or watch our videos on <https://www.youtube.com/watch?v=zQewb08M4sA&list=PLz1XwEYRuku5rZjJWBr6SDi93jgWZ4FHL>

High-level (Kinova.Api.Base)

Table 31: Send or play a trajectory (Cartesian or joint)

Method	Description
PlayCartesianTrajectory (ConstrainedPose)	Moves to the specified pose (with specified Cartesian constraint on trajectory)
	 Note: This method will be deprecated in a future software version.
PlayCartesianTrajectoryPosition (ConstrainedPosition)	Moves to the specified position (with specified Cartesian constraint on trajectory)
	 Note: This method will be deprecated in a future software version.

Method	Description
PlayCartesianTrajectoryOrientation (ConstrainedOrientation)	Moves to the specified <i>orientation</i> (with specified Cartesian constraint on trajectory)  Note: This method will be deprecated in a future software version.
PlayJointTrajectory (ConstrainedJointAngles)	Moves to the specified <i>joint angles</i> (with specified joint constraint on trajectory)  Note: This method will be deprecated in a future software version.
PlaySelectedJointTrajectory (ConstrainedJointAngle)	Moves specified joint to the specified joint angle (with specified joint constraint on trajectory).  Note: This method will be deprecated in a future software version.
PlayPreComputedJointTrajectory (PreComputedJointTrajectory)	Play pre-computed angular trajectory
ExecuteWaypointTrajectory(WaypointList)	Executes a trajectory defined by a series of waypoints in <i>joint space</i> or in <i>Cartesian space</i>

Table 32: Send Cartesian command to tool

Method	Description
SendTwistCommand (TwistCommand)	Sends a twist command to tool (velocity and angular velocity)
SendTwistJoystickCommand (TwistCommand)	Sends a twist joystick command to tool. The twist values sent to this call are expected to be a ratio of maximum value (between -1.0 and +1.0)

Table 33: Send command to joints

Method	Description
SendJointSpeedsCommand (JointSpeeds)	Sends a joint speeds command, that is the desired speed of one or many joints
SendSelectedJointSpeedCommand (JointSpeed)	Sends a speed command for a specific joint

Method	Description
SendJointSpeedsJoystickCommand (JointSpeeds)	Sends the desired joystick speeds for one or multiple joints. Values sent to this call are expected to be a ratio of maximum value (between -1.0/+1.0)
SendSelectedJointSpeedJoystickCommand (JointSpeed)	Sends a joystick speed for a specific joint. Value sent to this call is expected to be a ratio of maximum value (between -1.0 and +1.0)

Table 34: Send gripper commands

Method	Description
SendGripperCommand (GripperCommand)	Sends a command to move the gripper. Commands the fingers of the gripper in either position or velocity.

Low-level (Kinova.Api.BaseCyclic)**Table 35: Low-level cyclic commands**

Method	Description
Refresh (Command)	Sends a new incremental refresh command to the actuators and gripper for the current 1 ms interval. Command actuators position, angular velocity, and motor current, as well as gripper finger motors. Receives feedback on current status of base , actuators, tool, and gripper.
RefreshCommand (Command)	Sends a new incremental refresh command to the actuators and gripper for the current 1 ms interval. Command actuators position, angular velocity, and motor current, as well as gripper finger motors. No feedback provided.
RefreshFeedback ()	Receives feedback on current status of base, actuators, tool, and gripper.

High-level control modes description

The robot is controllable through a number of high-level [control modes](#).

Table 36: High-level control modes

Control mode	Mode type
Trajectory modes	Angular trajectory
	Cartesian trajectory
Joystick control modes	Cartesian joystick
	Angular joystick

Trajectory control modes

Trajectories define a path of the robot through space, either Cartesian or joint, over a time period. There are different ways available to define trajectories.

Using trajectory [control modes](#), the user can command the robot to a desired endpoint.

There are a few different ways a [trajectory](#) can be defined.

Reach endpoint

Users can provide an endpoint to reach, and let the control software of the robot compute how to follow a [path](#) to reach the endpoint. The robot controller computes an interpolated trajectory, between current [pose](#) and target pose, to reach the final [position](#) while satisfying configured limits, and commands the robot to follow this trajectory. These trajectories, once defined, can be played back. This is a good setting when the robot must go to a desired end state, but there is little concern about the exact path it follows getting there and exactly how long it takes.

Pre-computed trajectory

Users can supply a pre-computed trajectory to the robot. A pre-computed trajectory is generally auto-generated by some sort of path planning software or algorithm rather than built manually. A pre-computed trajectory is defined as a time series of settings for [joints](#) angular positions, velocities, and accelerations at each timestamp. The robot control software verifies that the trajectory is valid and reasonable, satisfying configured limits. Users can indicate a desired continuity mode for the trajectory against which the trajectory can be checked; in other words, the trajectory can be checked against its position, or position and velocity, or position, velocity and acceleration.

The robot control library performs the validations on the pre-computed trajectory.

- Trajectory is non-empty
- For each trajectory point, position, velocity, and acceleration values must be provided for each joint in the robot
- Trajectory contains no NaN values
- Timestamp of the first trajectory point must be 0.000 seconds and the difference in time stamps between successive points must be 0.001 seconds
- Joint positions, speeds, accelerations, and torques must be in the range of robot joint limits
- Continuity; the trajectory is continuous in terms of one or more continuity checks :
 - Position: joint position variation between successive timesteps is less than the maximum variation, based on joint speed limit
 - Speed: speed values are consistent, in the range of tolerances, with derivative of position
 - Acceleration: acceleration values are consistent, in the range of tolerances, with derivative of speed
- Trajectory execution, when starting the trajectory: the joint positions and speeds for the first point in the trajectory must match the initial robot joint positions and speeds



Note: For safety reasons, trajectories failing any of the validation checks are rejected, and an error [notification](#) is displayed. To get more detailed information about why a particular trajectory fails, use the method `GetTrajectoryErrorReport()` in `Kinova.Api.Base`.

Trajectory modes

In Cartesian Trajectory mode, the positions are defined in terms of the desired [Cartesian space](#) pose of the [tool frame](#). This mode enables [singularity avoidance](#). Cartesian trajectory mode can be activated by sending a pose endpoint or a list of Cartesian [waypoints](#). Cartesian Trajectory mode is not compatible with pre-computed trajectories.

In Angular Trajectory mode the positions are defined in terms of the desired joint angles for the actuators. Angular Trajectory mode can be activated by sending an endpoint joint state, a list of angular waypoints, or a pre-computed trajectory.

Users have the option to apply constraints to trajectories. There are three options available for constraints.

- Duration: the time period (in seconds) in which the trajectory is to be carried out
- Speed: the maximum speed, measured in m/s for Cartesian and °/s for angular, of the motion while carrying out the trajectory
- No constraint: the robot moves to the endpoint without the time or speed being specified



Note: If a requested trajectory constraint causes angular limits to be exceeded in the course of the trajectory, such as when the duration is too short and requires a speed or acceleration that is not feasible, the trajectory is rejected.

In angular trajectories, all three options, duration, speed, and no constraint, are available.

In Cartesian trajectories, the speed constraint or no constraint options are available.

Trajectory control modes are set by sending a trajectory using the appropriate [API](#) methods, or the Web App **Actions** page.

Joystick control modes

Joystick control modes allow the robot to be controlled by sending motion (speed) commands by either Cartesian or joint.

Joystick Control modes provide the user the ability to create a desired motion of the robot by sending commands to the robot. This is done with joystick control inputs (physical gamepad or virtual joystick) or directly using [API](#) commands.

In **Cartesian Joystick** mode the motion of the robot [end effector](#), both linear and angular, is controlled. Cartesian Joystick is entered by:

- Sending [API twist](#) commands (specify linear and angular velocity of the end effector)
- Activating Xbox gamepad Twist linear and Twist [angular modes](#)
- Sending Web App Pose virtual joystick control inputs in velocity control

This mode provides for [singularity avoidance](#) and obstacle avoidance (protection zones).

For Cartesian control, the reference [frames](#) for translation and [orientation](#) control are configurable:

Table 37: Cartesian reference frames options

Cartesian frame mode	Reference frame for translation	Reference frame for orientation
mixed mode	base	tool
tool mode	tool	tool
base mode	base	base



Note: Orientations are defined using an **x-y-z extrinsic convention**. That is, rotation about the fixed x-axis, followed by rotation about the fixed y-axis, followed by rotation about the fixed z-axis.

In **Angular Joystick** mode the [joints](#) of the robot are moved in angular space using joint angular velocity commands or joystick control inputs provided to the actuators. The joints can be moved individually or together.

Enter Angular joystick mode in multiple ways.

- Send [API](#) joint speed commands
- Activate Xbox gamepad Joint [mapping](#)
- Send commands through Web App Angular virtual joystick

Low-level control detailed description

Low-level control is achieved by using the `UDPTransportClient` command over port 10001.

The command establishes a communication channel between [API](#) cyclic clients and the robot at a rate of 1 kHz. The cyclic clients provide methods to obtain feedback from the sensors and to send commands directly to devices; using the cyclic client methods bypasses the regular overhead from the controller.

Configurations and safeties

To make sure people, the environment, and the robot are safe, configure the Gen3 lite robot and define the safety limitations from Web App.

Configurable parameters

There are a number of parameters of the robot that are configurable to customize the operation of the robot. The parameters can be configured using the API and Web App.

The parameters can be configured using the appropriate Kinova® Kortex™ APIs.



For more details on how to perform configuration using the APIs, see the [API](#) documentation.

Some of these parameters can also be configured using the Web App [GUI](#).

To configure the parameters with Web App

1. Launch Web App.
2. Navigate to the **Robot Configurations** page.
3. Access configurable parameters on one of the available tabs.

Related topics

[Accessing Kinova Kortex Web App](#) on page 58

Base configuration

Base configurations cover a range of core configurable items. These can be configured through the Kinova® Kortex™ API or the Kinova® Kortex™ Web App.

Table 38: Base configuration

Configurable item	Description	Where to configure it
User Profiles	Create, read, update and delete user profiles	Kinova.Api.Base API Web App > Users

Configurable item	Description	Where to configure it
Protection Zone	Create, read, update and delete protection zones (for obstacle avoidance). Configurable parameters are: • enabled/disabled • zone shape type (rectangular prism, cylinder, sphere) • zone origin and orientation • zone dimensions • envelope thickness • zone limitation types (velocity) and values • envelope limitation types (velocity) and values	Kinova.Api.Base API Web App > Operations > Protection Zones
Control Mapping	Create, read control mapping	Kinova.Api.Base API Web App > Configuration > Controllers
Action	Create, read, update, delete action	Kinova.Api.Base API Web App > Operations > Actions
Sequence	Create, read, update, delete a sequence of actions	Kinova.Api.Base API Web App > Operations > Actions
IPv4	Set IPv4 configured (for specified network adapter): • IP address • subnet mask • default gateway	Kinova.Api.Base API Web App > Configurations > Wireless & Networks
Communication Interface	Enable communication interface: • network type (Wi-Fi or Ethernet) • enabled/disabled	Kinova.Api.Base API Web App > Configurations > Wireless & Networks
Wi-Fi	Set: • SSID • security key • automatic connection allowed • country code	Kinova.Api.Base API Web App > Configurations > Wireless & Networks

Configurable item	Description	Where to configure it
Restore factory settings	Delete all configurations and reverts settings to factory defaults (except network settings)	Kinova.Api.Base API Web App > Configurations > Robot > Arm
Restore factory product configuration	Restore product configuration to factory product configuration	Kinova.Api.Base API

Actuators configuration

Actuators configurations set items related to the functioning of the actuators. Most of these can be configured only through the API. Zeroing of the actuators torque offsets can be done through Web App.

Table 39: Actuator configuration (Kinova.Api.ActuatorConfig)

Configurable item	Description	Where to configure it
Axis offsets	Set actuator axis offset position	Kinova.Api.ActuatorConfig API
Torque offset	Set actuator torque zero configuration	Kinova.Api.ActuatorConfig API Web App > Configurations > Robot > Actuators
Control mode	Set actuator control mode. Options: • position • velocity • current • torque	Kinova.Api.ActuatorConfig API

Configurable item	Description	Where to configure it
Control loop parameters	Configure an individual control loop parameter: • joint or motor position • joint or motor velocity • joint torque • motor current Configure: • error saturation value • output saturation value • kAz • kBz • error dead band value	Kinova.Api.ActuatorConfig API
Vector drive parameters	Set vector drive parameters: • kpq • kiq • kpd • kid	Kinova.Api.ActuatorConfig API
Encoder derivative parameters	Set encoder derivative parameters: • maximum window width • minimum encoder tick count	Kinova.Api.ActuatorConfig API
Command mode	Set command mode. Options: • cyclic - cyclic data only • asynchronous - configuration messages only • cyclic jitter compensation using position or position and velocity inputs  Note: These options are available in the API but should NOT be used	Kinova.Api.ActuatorConfig API
Servoing	Enable servoing	Kinova.Api.ActuatorConfig API

Device configuration

Device configurations impact settings for robot devices. These can be configured through the API.

Table 40: Device configuration

Configurable item	Description	Where to configure it
Run mode	Set device run mode (Run, Calibration, Configuration, Debug, Tuning)	<code>Kinova.Api.DeviceConfig</code> API
IPv4 settings	(For devices other than base) Set device IPv4 address, subnet mask, default gateway	<code>Kinova.Api.DeviceConfig</code> API
Safeties	Enable/disable, set warning and/or error thresholds for specific safeties	<code>Kinova.Api.DeviceConfig</code> API
Capacitive sensor	Set mode and thresholds	<code>Kinova.Api.DeviceConfig</code> API

Safety items

There are a number of safety items related to different components of the robot. These are viewable in Web App **Safeties** page.

Safety items, and their associated warning and error thresholds are active on the robot and are viewable within the **Safeties** page of Web App. There are several categories of [safeties](#):

- Base (arm) safeties
- Actuators safeties
- Gripper safeties

The tables that follow give more information about the safeties, including:

- **Description:** significance of the safety item
- **Hard limit (lower):** the minimum allowable value for the item
- **Hard limit (upper) :** the maximum allowable value for the item
- **Default warning/error threshold:** default configurations for the safety thresholds.

Base safeties

The following base-related safety items are active on the robot and are viewable in Web App.

Table 41: Base Safety items

Safety Item	Description	Threshold	warning
			error
Maximum Core Temperature	CPU temperature is above upper limit. CPU automatically downclocks at 80 °C and shuts down at 115 °C.	75.0 °C	
		85.0 °C	
Minimum Voltage	The base voltage reading is below lower limit	18.0 V	
		16.0 V	
	 Note: The minimum voltage must be lower than the maximum voltage		
Maximum Voltage	The base voltage reading is above upper limit.	30.0 V	
		31.0 V	
	 Note: The maximum voltage must be higher than the minimum voltage		
Inrush current limiter fault	Inrush current limiter fault triggered	n/a	
NVRAM Corrupted	Non-volatile memory corrupt	n/a	
Network Initialization Error	Arm is detected but arm internal connectivity not available	n/a	
Joint Detection Error	The number of detected joints does not match the configured arm joint count	n/a	
Network Error	Loss of actuator cyclic data response	n/a	
Joint Fault	At least one joint is in a fault state	n/a	
Emergency Stop Activated	Emergency stop activated	n/a	
Emergency Line Activated	Emergency line activated	n/a	
Discrete Input Stuck Active	Joystick Control indicator (GPI) stuck	n/a	

Safety Item	Description	Threshold	warning
			error
Incompatible firmware version	Firmware incompatible with hardware board revision	n/a	
Firmware Update Failure	Indicates a failure in the firmware update process	n/a	

Actuators safeties

The following actuator-related safety items are active on the robot and are viewable in Web App.

Table 42: General actuator safety items (apply the same to all actuator sizes)

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Following Error	The error between the command and the reported position is above upper limit. Note: Only active when in servoing state	0°		3.0°	
		10°		5.0°	
Maximum Velocity	The computed velocity of the actuator is greater than threshold °/sec.	0 °/s		180 °/s	
		250 °/s		200 °/s	
Magnetic Position	Position step of more than threshold °/ms has been read on the magnetic sensor.	0°		1.25°	
		2°		1.50°	
Hall Position	Position step of more than threshold °/ms	0°		0.1758	
		1°		0.3516	
Hall Sequence	Invalid Hall sequence detected.	n/a		n/a	

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Hall magnetic mismatch	Compares the current absolute position (as seen from Hall sensors) versus the current absolute position (as seen from the magnetic sensor). Makes sure the difference between those values are inside the defined tolerances.	0.0°		4.0°	
		10.0°		4.5°	
Maximum motor current	Checks the motor current. Makes sure it is below the defined upper limit.	0 A		11 A	
		12 A		12 A	
Minimum Voltage	The voltage reading is below lower limit.	14.0 V		18.0 V	
		24.0 V		16.0 V	
	 Note: The minimum voltage thresholds must be lower than the maximum voltage thresholds.				
Maximum Voltage	The voltage reading is above upper limit.	24.0 V		28.0 V	
		36.0 V		32.0 V	
	 Note: The maximum voltage thresholds must be higher than the minimum voltage thresholds.				
Maximum Core Temperature	Core temp above upper limit	0 °C		80.0 °C	
		100.0 °C		90.0 °C	
Non-Volatile Memory Corrupted	Non-volatile memory corrupt	n/a		n/a	

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Motor Driver Fault	Driver chip reported a major fault	n/a		n/a	
	 Note: electronic protection cannot be deactivated				
Watchdog Triggered	Watchdog was triggered	n/a		n/a	

Table 43: Small actuator specific safeties

Safety item	Description	Hard limit	lower	Threshold	warning
			upper		error
Joint Limit Low	Checks if the actuator is within the defined lower position limit.	-154°		-150°	
		0°		-152°	
Joint Limit High	Checks if the actuator is within the defined upper position limit.	0°		150°	
		154°		152°	
Maximum motor temperature	Checks the motor temperature. Makes sure it is below the defined upper limit.	0°		60°	
		80°		70°	

Table 44: Medium actuator specific safeties

Safety item	Description	Hard limit	lower	Threshold	warning
			upper		error
Joint Limit Low	Checks if the actuator is within the defined lower position limit.	-159°		-155°	
		0°		-157°	
Joint Limit High	Checks if the actuator is within the defined upper position limit.	0°		155°	
		159°		157°	

Safety item	Description	Hard limit	lower	Threshold	warning
			upper		error
Maximum motor temperature	Checks the motor temperature. Makes sure it is below the defined upper limit.	0°		60°	
		90°		75°	

Table 45: Large actuator specific safeties

Safety item	Description	Hard limit	lower	Threshold	warning
			upper		error
Joint Limit Low	Checks if the actuator is within the defined lower position limit.	-159°		-155°	
		0°		-157°	
Joint Limit High	Checks if the actuator is within the defined upper position limit.	0°		155°	
		159°		157°	
Maximum motor temperature	Checks the motor temperature. Makes sure it is below the defined upper limit.	0°		60°	
		90°		70°	

Gripper safeties

The following gripper-related safety items are active on the robot and are viewable in Web App.

Table 46: Gripper Safety items

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Following error	Compares the position command given by the base versus the current command as seen by the actuator. Makes sure the difference between those values are inside the defined tolerances.	0°		8°	
		15°		12°	

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Maximum velocity	Checks if the velocity as seen by the actuator is within the defined limits.	0°/s		150°/s	
		200°/s		170°/s	
Joint limit low	Checks if the actuator is within the defined lower position limit.	-5°		-1°	
		0°		-2°	
Joint limit high	Checks if the actuator is within the defined upper position limit.	0°		105°	
		115°		110°	
Hall position	Compares the current absolute position (as seen from Hall sensors) versus the last absolute position (as seen from the Hall sensors). Makes sure the difference between those values are inside the defined tolerances.	0		0.1758	
		1		0.3516	
Hall sequence	Checks if the Hall sequence is valid. This is done evaluating a parameter that does this in the process phase.	n/a		n/a	
Maximum motor current	Checks the motor current.	0 A		1.2 A	
	Makes sure it is below the defined upper limit.	1.8 A		1.4 A	
Minimum Voltage	Checks the entry voltage.	14.0 V		18.0 V	
	Makes sure it is above the defined lower limit.	24.0 V		16.0 V	
 Note: The minimum voltage must be lower than the maximum voltage					

Safety Item	Description	Hard limit	lower	Threshold	warning
			upper		error
Maximum Voltage	Checks the entry voltage. Makes sure it is below the defined upper limit.	24.0 V		28.0 V	
		36.0 V		32.0 V	
	 Note: The maximum voltage must be higher than the minimum voltage				
Maximum motor temperature	Checks the motor temperature. Makes sure it is below the defined upper limit.	0 °C		60 °C	
		90 °C		70 °C	
Maximum core temperature	Checks the MCU core temperature. Makes sure it is below the defined upper limit.	0 °C		80 °C	
		100 °C		90 °C	
Non-volatile memory (NVRAM) corrupted	Checks the non-volatile memory used for storage. Makes sure the memory isn't corrupted.	n/a		n/a	
Motor driver fault	Checks if the motor gate driver is in fault. This is done by polling the pin assigned for this purpose on the driver board.	n/a		n/a	
1000 ms		200 ms			
Watchdog triggered	Checks if the watchdog got triggered.	n/a		n/a	

Kinova[®] Kortex[™] Web App

Web App is used for controlling, configuring and monitoring the robot.

Device availability of Kinova[®] Kortex[™] Web App

The Kinova[®] Kortex[™] Web App is available on desktop computers, tablets, and smartphones.

Web App is a responsive web application. It is designed to adapt itself to various aspect ratios and resolutions enabling it to run on multiple platforms that support the Google Chrome browser.

This includes:

- Desktop or laptop computer
- Tablet computer
- Smartphone

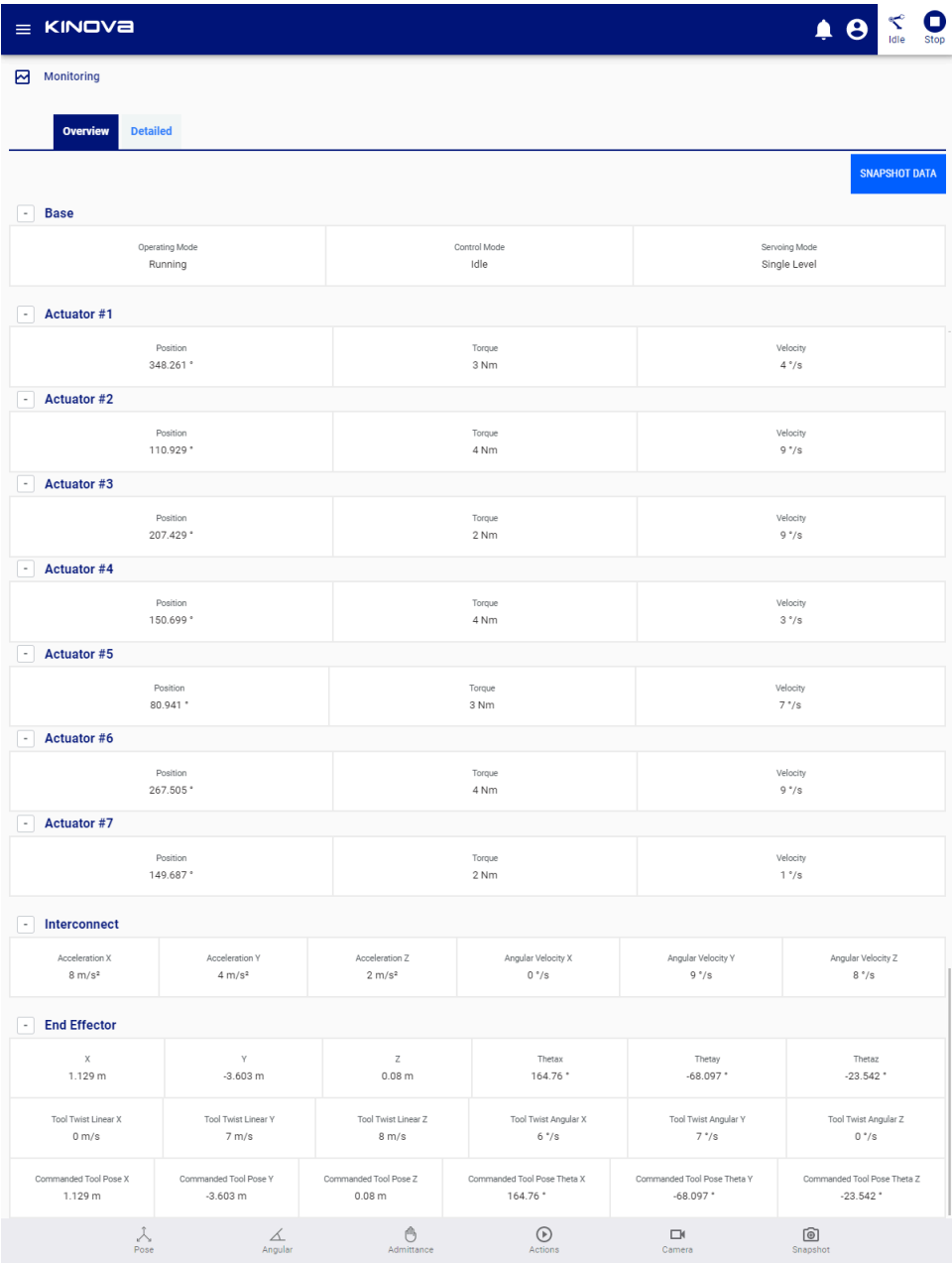


Figure 27: Desktop

</

Figure 28: Tablet

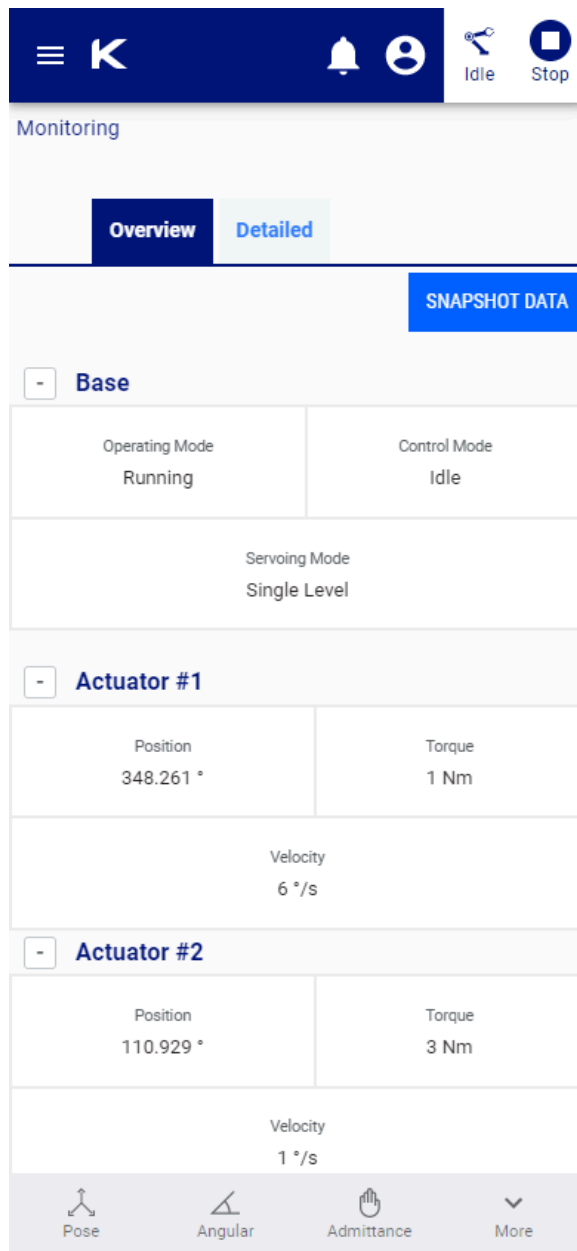


Figure 29: Smartphone

Platform and browser support

The Kinova® Kortex™ Web App supports specific platforms and browsers.

Operating system support

Kinova® Kortex™ Web App is supported on various operating systems (OS).

- Microsoft Windows 7/10/11 (32-bit or 64-bit)
- Ubuntu LTS 16.04 (64-bit) and later

- Android 8.1 and higher

Kinova® Kortex™ development computer platform

The hardware and OS requirements for a development computer are multi-platform.

- Microsoft: Windows 7/10/11 (32-bit or 64-bit)
- Linux: Ubuntu 16.04 (64-bit) and later

Browser support

- Chrome

Other platforms and browsers are not tested; some features may work differently in those cases.

User login

Given the robot base [IP](#) address and a computer connected to the robot, users can log in to Web App with username and password through a web browser.

After establishing a network connection between the device and the robot, open a web browser and enter the IP address for the robot [base](#) external interface.

Web App launches, ending in a login popup.

Enter the username and password and press **CONNECT**.

There are default username and password for the robot.

- username: admin
- password: admin

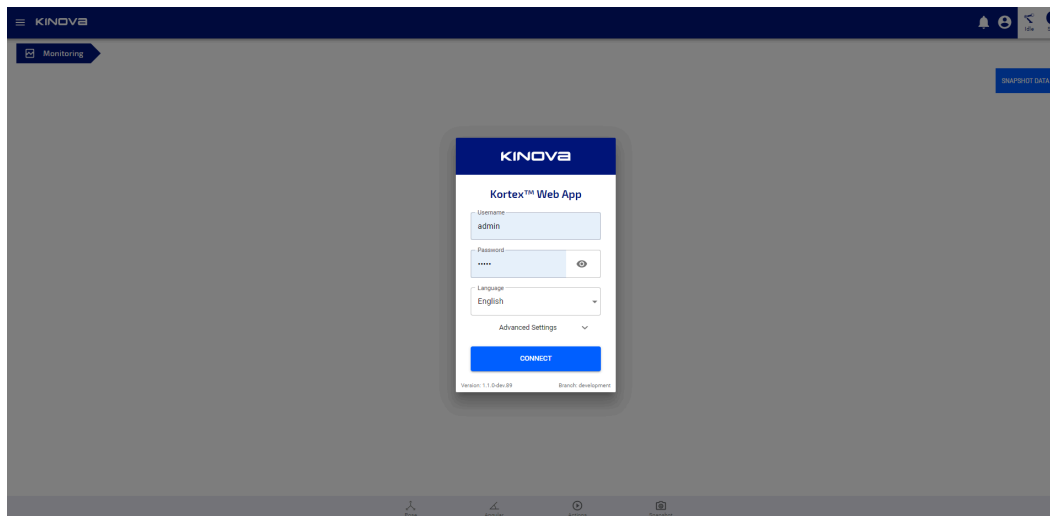


Figure 30: User login

Click **CONNECT**, to launch and initialize Web App. While the application initializes, Web App gives visual feedback to the user on the status of initialization of the application.

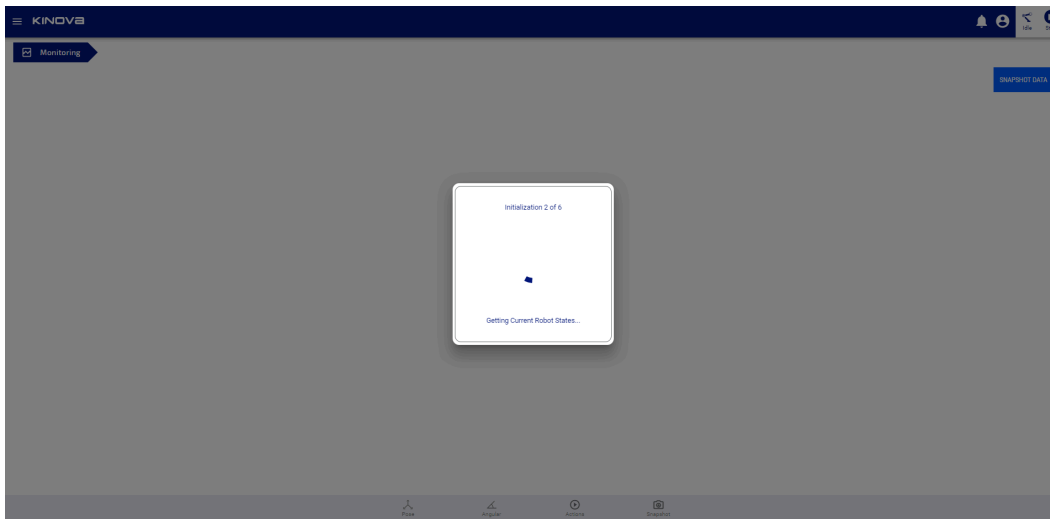


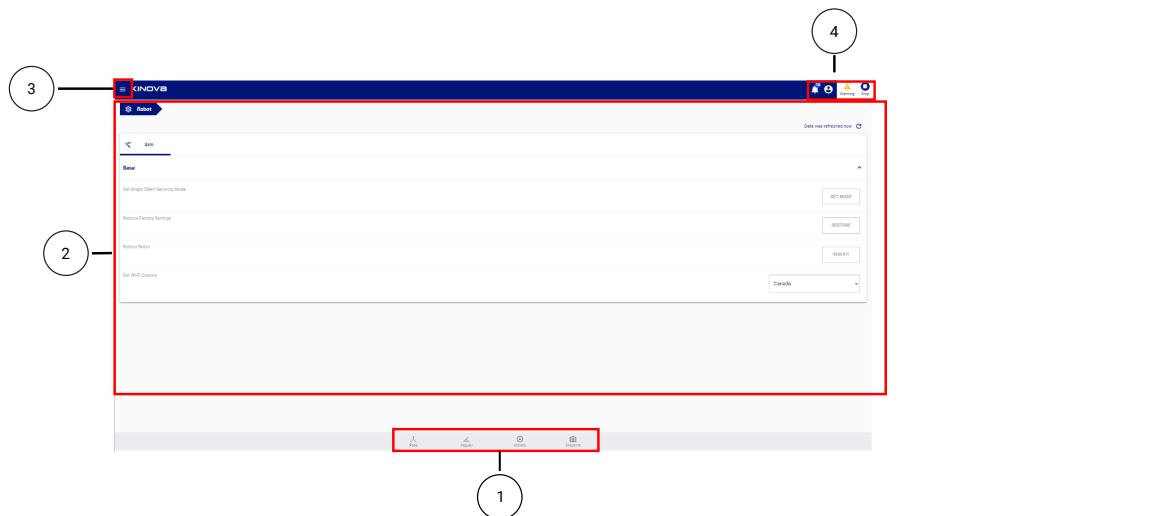
Figure 31: Initializing...

Web App layout and navigation

An overview of the basic layout and navigation of Web App helps to maximize the use of the application.

The Kinova® Kortex™ Web App screen is divided into several main sections:

- Main navigation panel
- Main information panel
- Notification bar
- Shortcuts panel
- Robot control panel
- Mode indicator, user icon, and stop



- | | | | |
|---|------------------------|---|--|
| 1 | Robot control panel | 3 | Pages menu |
| 2 | Main information panel | 4 | Notifications indicator, user icon, mode indicator, stop |

Figure 32: Web App layout

Pages menu

In the middle of the screen is the main information panel containing the contents of each page of the application. The page can be changed from a pages menu on the left of the screen. The menu is hidden by default, but can be launched by clicking the menu icon in the upper left.

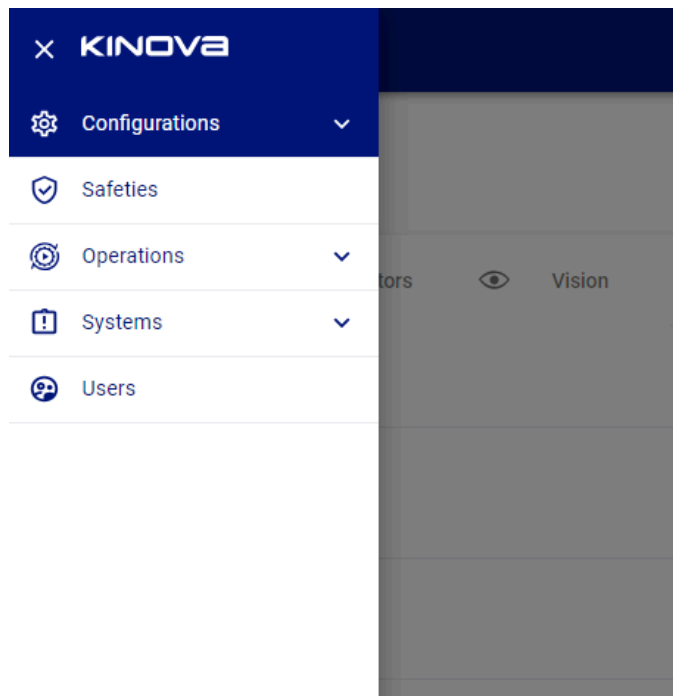


Table 47: Organization of the Page menu

Menu item	Submenu items
 Configurations	 Robot  Speed Limits  Controllers  Wireless & Networks
 Safeties	
 Operations	 Actions  Protection Zones
 Systems	 System Information  Monitoring  Upgrade
 Users	

Click on any of the indicators to launch a pop-up that shows more information.

Table 48: Indicators and control indicators

Indicator icon	Indicator	Mode or robot state	Description
	Notifications indicator		Number of most the important <i>notifications</i>
	User profile icon		Shows the user session icon
	Control mode		Status of the control mode situation of the robot
		Warning	Robot is in warning state
		Error	Robot is in a fault state

Indicator icon	Indicator	Mode or robot state	Description
		Idle	Robot is not currently being controlled by any user session; waiting
		Playing	A sequence is being played on the robot
		Running	The robot is being actively controlled by a user
	Stop		Button control which, when tapped, stops the arm from moving

Robot control panel

The control panel is on the bottom of the screen, and consists of a group of four buttons.

Use some of the buttons to launch pop-up windows for virtual joystick controls.

-  Pose Virtual Joystick
-  Angular mode control

Control the movement of the robot with the virtual joysticks instead of a physical controller.

In the same area, there are two other controls.

-  **Play action** Launches a window to play a selected action
-  **Snapshot** Captures a Cartesian, angular, or gripper [pose](#)

Click on one of the buttons in the robot control panel to launch a smaller window from the bottom of the screen that reveals the selected control panel.

Click the same button again to clear the smaller window at the bottom.

Pose virtual joystick control

The Pose virtual joystick provides an interface to control Cartesian movement of the robot from Web App.

The Pose virtual joystick panel allows users to control the position and [orientation](#) of the [end effector](#) through Web App, using a mouse on laptop or desktop computer, or touch control (on a tablet or smartphone).



Figure 33: Pose joystick panel

The Pose virtual joystick panel is launched by clicking the first button () on the robot control panel.

Translation and rotation joystick controls

Control the linear and angular motion of the tool with the Pose virtual joystick controls.

There are two sets of joysticks.

Linear Use to apply a translational motion to the end effector

Angular Use to apply a rotational motion to the end effector at the current position

Each set of joysticks features a 2-axis joystick for controlling the x and y axes, and a 1-axis joystick to control in the z-axis. For the 2-axis linear joystick, the user can configure the joystick [axis](#) that is assigned to control the y direction movement.

As the controls are moved, a display is provided for the current position (x, y, z) and orientation (θ_x , θ_y , θ_z) of the end effector.



Note: The orientation representation uses an x-y-z extrinsic convention.

Finger controls

Open and close the fingers using a single 1-axis joystick control.

Push the control upwards to open the fingers and downwards to close. The fingers position can be controlled between 0%, which is fully closed, and 100%, which is fully open.

Additional settings

To keep the user interface less cluttered, some Cartesian virtual joystick settings and information are hidden from immediate view, but can still be accessed.

Click the caret icon on the upper right-hand side to reveal additional controls and information.

- current linear and angular velocity display
- invert z and θ_z toggles
- reference frame selection
- gripper speed control
- speed limit

Speed control

The actuators speed and finger speed can be adjusted between 0 and 100% of the hard limits for the robot.

Reference frames

Use reference frames to help understand how the robot moves.

The position of the end effector can be specified in one of three reference [frame](#) conventions.

Reference frame convention	Description
Mixed	Linear in base reference frame, angular in tool reference frame
Base	Linear and angular in base reference frame
Tool	Linear and angular in tool reference frame

Z and θ_z toggles

Toggle the meaning of the 1-axis z-direction controls.

The default for the 1-axis z-direction controls is that up increases the z-position or z-angle, while down decreases it. The direction controls can be reversed using the Invert Z and Invert θ_z toggles.

Angular virtual joystick control

The Angular virtual joystick provides an interface to control the joint by joint movement of the robot from Web App.

The angular virtual joystick panel allows users to control the robot *joint angles* and gripper fingers through Web App using a mouse (or touch control on a tablet or smartphone).

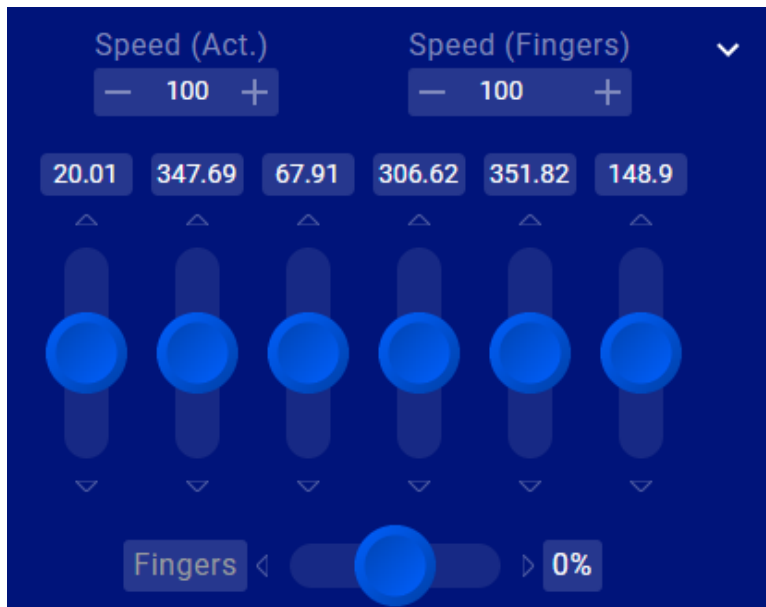


Figure 34: Angular virtual joystick panel

The joint angles are controlled through angular velocity (controlling the joint speed for each actuator)

The angular joystick panel is launched by clicking the second button  on the robot control panel.

Control the angle of each actuator, as well as the opening and closing of the fingers when a gripper is installed, with the virtual joystick controls. As the virtual joystick controls are manipulated, the robot arm *joints* respond accordingly.



Note: For joints with joint rotation limits, the robot enforces software joint angle limits to prevent these joints from reaching the physical limits. When the user controls these joints, the software prevents the arm joints from responding when the limits are reached.

The value of each angle is displayed in degrees. The value displayed is restricted to one full rotation from 0 to 360°.

Additional settings

To keep the user interface less cluttered, some angular virtual joystick settings and information are hidden from immediate view, but can still be accessed.

Click the caret icon on the right-hand side to reveal additional controls. Use the settings to control the maximum actuators and finger speed, between 0 and 100% of maximums.

To control the angle of each actuator, use the virtual joystick controls to apply a velocity in the given direction. Pushing the joystick up causes the angle to increase, while pushing it down causes it to decrease. The further up or down the joystick is pushed, the higher the angular speed for the joint, to the maximum set limit. The angle continues to change as long as the joystick is being pushed.

Control the *end effector* finger *position* with another joystick if an end effector is installed. The values for the finger state range between 0%, fully closed, and 100%, fully open. Push the joystick to the right to increase the percentage (and open the fingers). Push the joystick to the left to decrease the percentage and close the fingers.

Virtual joystick keyboard shortcuts

Keyboard shortcuts are available for Web App virtual joysticks when accessing Web App from a desktop device with keyboard. This serves as an alternate to mouse and touch for controlling the robot with the virtual joysticks.

The virtual joysticks for Web App are controllable with mouse or touch inputs. Some people, particularly those with a background in *PC* gaming, may find it more natural to control using keyboard shortcuts. When accessing Web App with a desktop device that has a keyboard, such as a desktop or laptop PC, there are handy keyboard shortcuts available for the joystick controls.

Table 49: Cartesian pose translation joystick shortcuts

Control		Shortcut
X translation	+	W
	-	S
Y translation	+	A
	-	D
Z translation	+	Q
	-	E

Table 50: Cartesian pose orientation joystick shortcuts

Control		Shortcut key
X rotation	+	I
	-	K
Y rotation	+	L
	-	J
Z rotation	+	U
	-	O

Table 51: Cartesian gripper controls

Control		Shortcut key
Gripper fingers	open gripper fingers	C
	close gripper fingers	V

Increase the angles for joints 1 to 7 by using the keys 1 to 7 on the top row of the keyboard.

Decrease the angles by using the letter keys QWERTYU on the second row of the keyboard.

Table 52: Angular joystick actuators control shortcuts

Control		Shortcut key
Actuator 1 angle	+	1
	-	Q
Actuator 2 angle	+	2
	-	W
Actuator 3 angle	+	3
	-	E
Actuator 4 angle	+	4
	-	R
Actuator 5 angle	+	5
	-	T
Actuator 6 angle	+	6

Control		Shortcut key
	-	Y
Actuator 7 angle	+	7
	-	U

Table 53: Angular joystick gripper controls

Control		Shortcut
Gripper fingers	open gripper fingers	C
	close gripper fingers	V

Actions panel

The **Actions** panel of Web App launches a control panel for the currently selected Action or Sequence.

The  **Actions** panel provides controls to play, stop, or loop a selected Action.

The panel pops up from the bottom of the screen when an action is selected in the **Actions** page. Clicking the control panel Actions icon toggles the visibility of the panel.

Snapshot tool

The snapshot tool in Web App captures and saves one of three types of instantaneous snapshots of the robot or gripper positioning.

 The [snapshot](#) tool lets users capture a snapshot of a current [position](#). This is a useful feature to help with building preset sequences (for demos or to capture/program a fixed set of movements).

Press the snapshot button to reveal a set of three snapshot options.

-  Cartesian pose
-  Joints position
-  Gripper position

Press one of the respective snapshot buttons to capture the chosen type of snapshot of the current robot position. This is saved, and shows up as one of the saved actions that is viewable in the **Actions** page.

Configurations page group

The **Configurations** page group of Web App includes several pages useful for robot configuration.

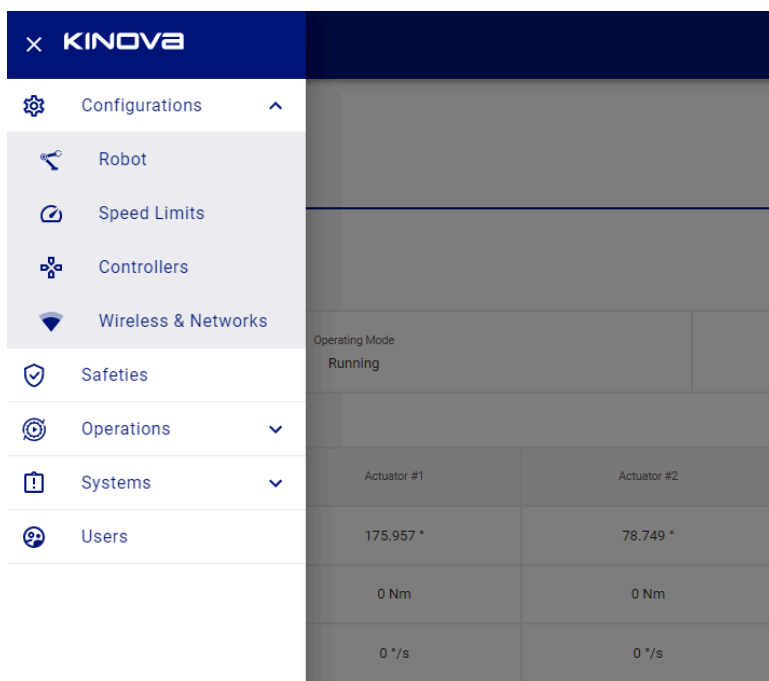


Figure 35: Configurations page group

⚙️ The **Configurations** page group contains pages allowing users to perform preliminary set up and configuration for robot hardware.

- 🤖 **Robot**
- ⚙️ **Speed Limits**
- ⚙️ **Controllers**
- 📶 **Wireless & Networks**

Robot configurations

Configure a broad range of robot parameters from the **Robot configurations** page of Web App.

🤖 From the **Robot configurations** page, adjust the configurable parameters of the robot hardware to customize the behavior of the robot.

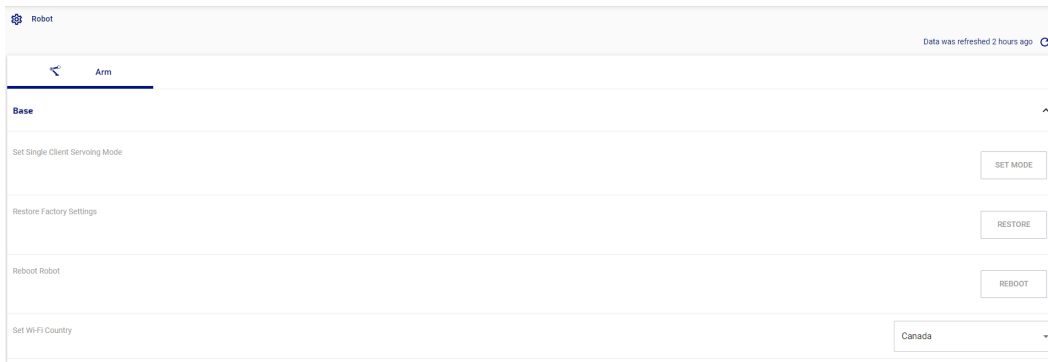


Figure 36: Robot configurations page

The configurable items are broken into sections by devices.

- Arm

Some configurable parameters of the robot can be configured on this page. Some other configurable items are handled on their own pages.

- **Protection Zones**
- **Controllers** to switch active control [mapping](#)
- **Actions** to define actions
- **Users** to edit [user profiles](#)
- **Wireless & Networks** to update network settings

Related topics

[Configurable parameters](#) on page 88

Speed Limits

Use the **Speed Limits** page of the Web App to set various soft limits for the robot, both globally, and by control mode.

 From the **Speed Limits** page, configure various soft speed limits for the robot, in line with options available through the `Kinova.Api.ControlConfig` [API](#).

Speed Limits

Basic Advanced

This section allows you to set a speed limit globally to all control modes.

Linear m/s or

Angular °/s or

Joint Speeds °/s °/s °/s °/s °/s °/s °/s or

Joint Accelerations °/s² °/s² °/s² °/s² °/s² °/s² °/s² or

Figure 37: Speed Limits page

The page contains two tabs.

- **Basic**
- **Advanced**

The Basic tab allows for setting speed limits globally for all *control modes*:

- Linear speed and angular speed for the tool (Cartesian)
- Joint speeds and accelerations (for each *joint*)

Set joint speed and acceleration limits for specific control modes from the **Advanced** page. The particular limits configurable under advanced depends on the mode chosen.

For any setting, set a specific value or set the value to the maximum.

Controllers

View and select control mappings for the robot from the **Controllers** page of Web App.

 View and toggle between the defined control mappings for any physical controllers associated with the robot from the **Controllers** page. Also, use the tools on the page to create control mappings.

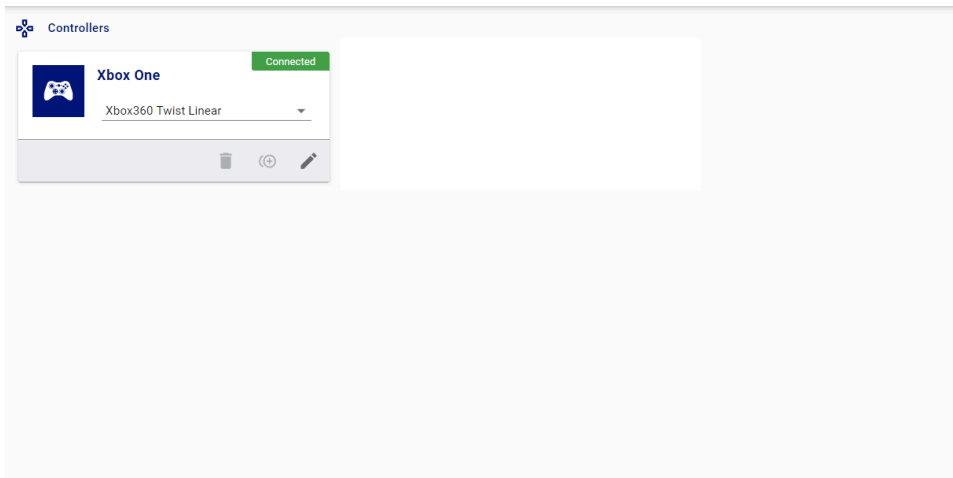


Figure 38: Controllers page

A mapping is a correspondence between the different controls on the controller and the resulting action produced in the robot.

The main information panel of the **Controllers** page has cards with control mapping information for each controller. The cards are in two sizes, large and small.

The **XBox Mapping** is for a generic Xbox gamepad. The robot has three preset control mappings defined for the Xbox gamepad. These mappings correspond to the mappings that can be toggled using the physical buttons on the gamepad.

- Xbox 360 Twist Linear
- Xbox 360 Twist Angular
- Xbox 360 Joint

Hitting the edit button on a card launches a window with tabs for the different control mapping modes available for the controller.

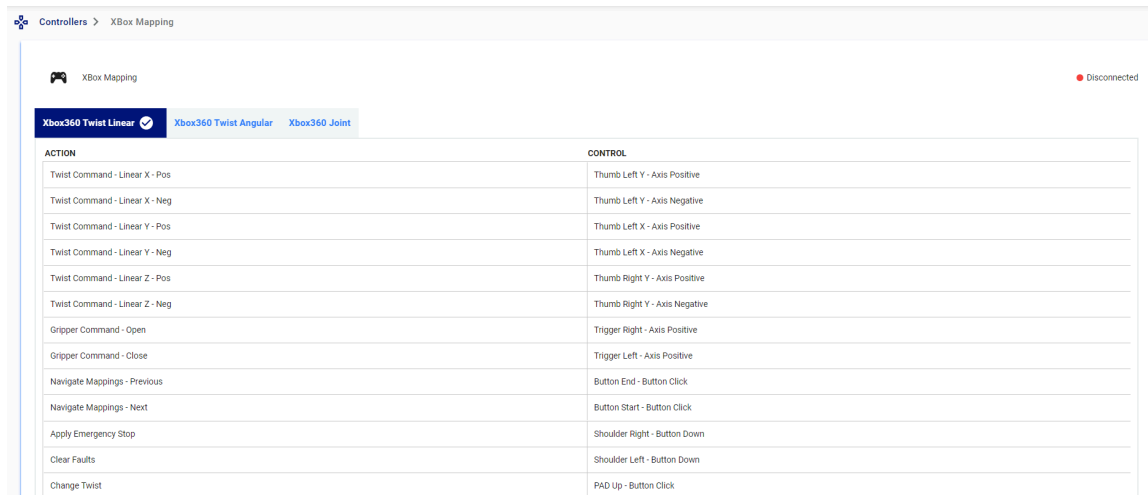
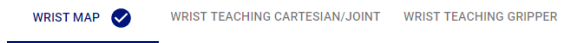


Figure 39: Xbox Mapping tab

At any given time, one map is set as active for the controller. The active controller map is indicated with a check mark icon. Another mapping can be set as active by clicking on the tab for the mapping.



Wireless & Networks

Configure network settings from the **Wireless & Networks** page of Web App.

📶 The **Networks** page is used to set network parameters for:

- Wi-Fi
- [RNDIS](#)/Micro [USB](#)
- USB-A Dongle

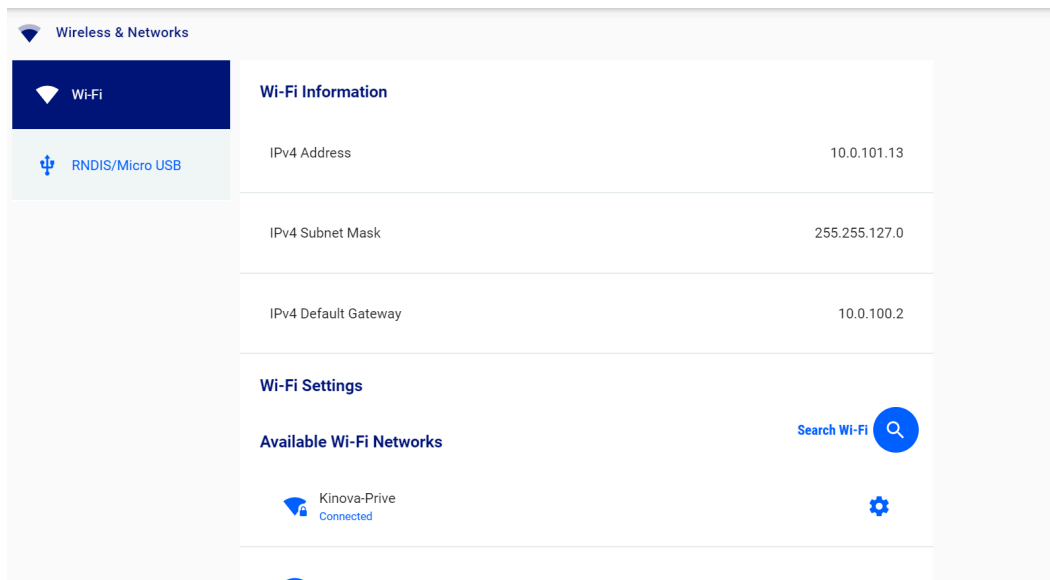


Figure 40: Wireless & Networks page

The page has tabs for each currently available connection method.

Enable and disable the DHCP server from the **RNDIS/Micro USB** settings page.

From the **Wi-Fi** page, enable and disable the Wi-Fi transceiver. Enable Wi-Fi networking with the robot, and find and connect to available Wi-Fi networks.

Safeties

View safety settings on the robot from the **Safeties** page of Web App.



The **Safeties** page allows users to view safety thresholds.

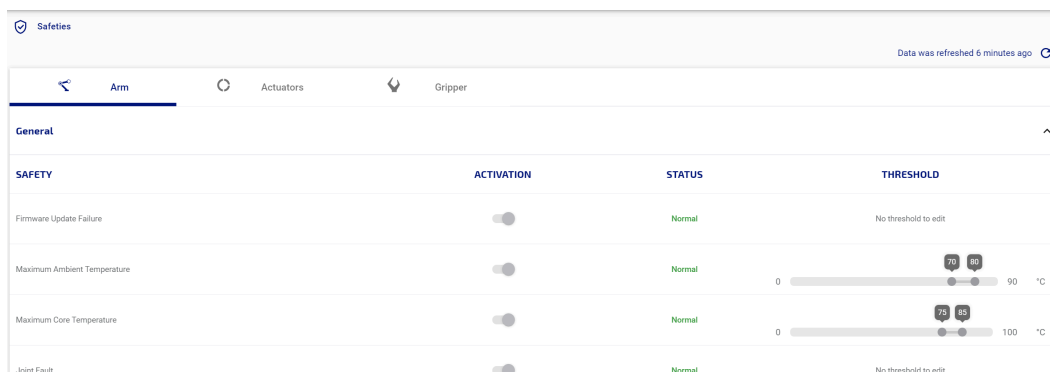


Figure 41: Safeties page

There are two types of safety thresholds.

Error

An error is a departure from normal parameters that is more serious than warnings and represents a situation which could damage the robot or endanger the user. The thresholds for errors are set at a more extreme level than warning thresholds.



Note: An error triggers an emergency stop for the robot.

Warning

A warning serves to signal that the robot is moving away from normal operational status toward an error state. A warning does not stop the robot.



Note: Some safety items do not have warning thresholds, only error thresholds.

Related topics

[Safety items](#) on page 92

Operations page group

All pages related to operating the robot are on the **Operations** page of Web App groups together pages useful to operating the robot.

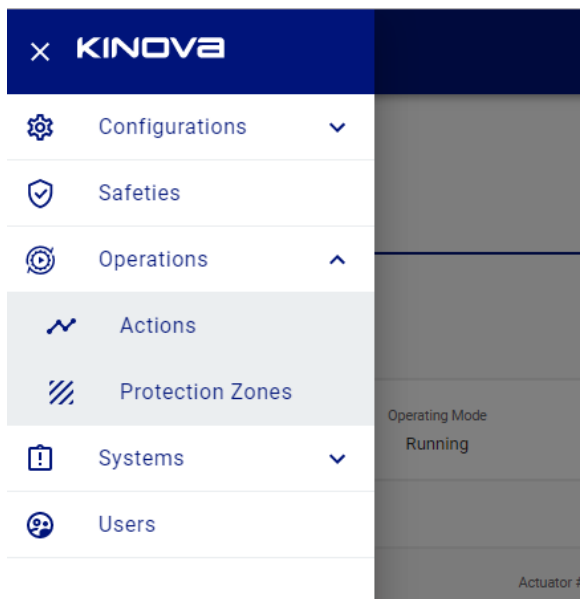


Figure 42: Operations page group

☺ The **Operations** page group contains pages allowing users to fine tune the configuration and performance of the robot after the initial setup.

-
-  **Actions**
-  **Protection Zones**
-  **Camera**

Actions

Define and edit robot actions from the **Actions** page of Web App. Also, play back actions and assemble them together into sequences of actions from the **Actions** page.



The **Actions** page allows user to define, view, and edit robot actions, as well as build sequences and play back actions and sequences.

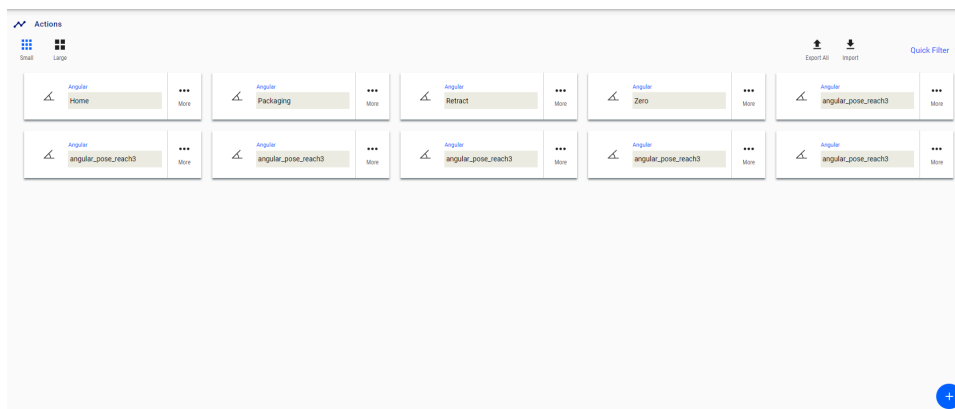


Figure 43: Actions page

Several actions are available in Web App.

Table 54: Available actions

Action	Description
Pose	Go to a Cartesian pose
Angular	Go to a combination of joint angles
End effector	Change the gripper state
Sequence	Take a series of actions one after the other

A **Cartesian pose** represents a single Cartesian endpoint for the robot. A pose consists of x, y, and z coordinates representing the position of the end effector, and the three angles θ_x , θ_y , and θ_z representing the orientation of the end effector.

Angular represents the set of joint angles for each of the arm joints.

End effector represents the gripper state, from 0% (fully open) to 100% (fully closed).

A **Sequence** is defined as a sequence of keyframe poses on a timeline, which are interpolated by the software in the robot base to produce intermediate frames to smooth the motion. Sequences are a sequential combination of Cartesian poses, angular settings, and end effector poses. Sequences may also include timed delays between movements.

The main information panel of the page shows cards with all the defined actions and sequences. New actions or sequences can be added with the + icon in the lower right of the main panel. Select the type of new item to create from the contextual menu that is launched.

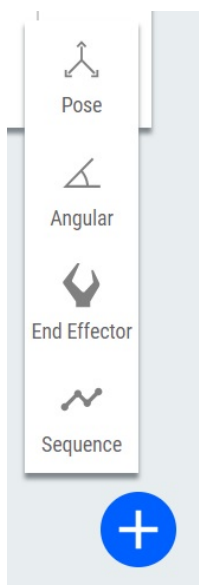


Figure 44: '+' menu

If the screen gets crowded after a while, filter to show only the chosen types of actions.

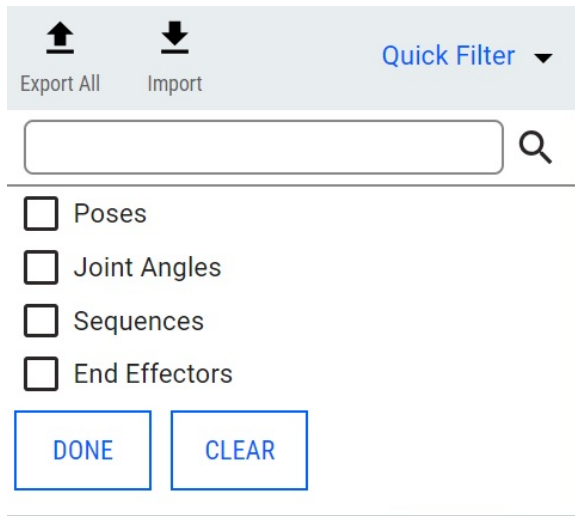


Figure 45: Action filter

Information about defined actions is viewable in information cards which come in two sizes, small and large, which can be toggled with buttons at the top of the main panel. The large cards carry full information about the action.

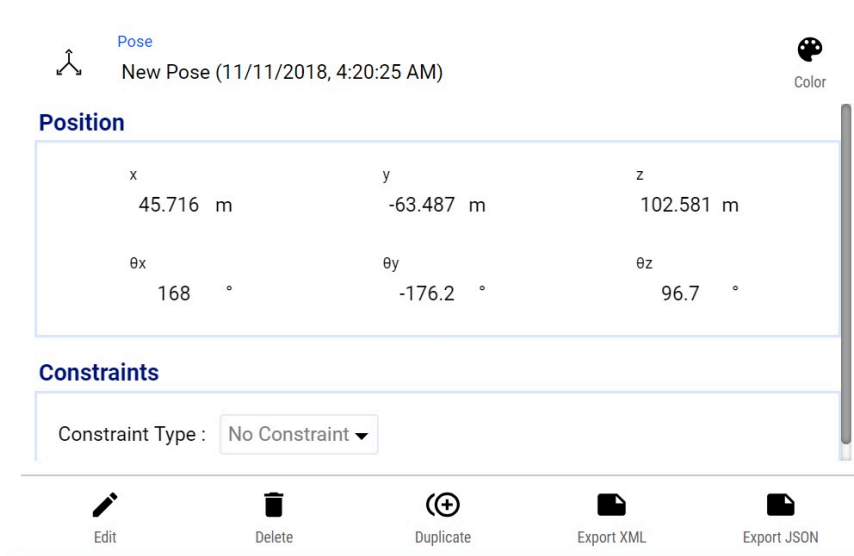


Figure 46: Information card

Interact with the large cards to work more effectively.

- Edit the action
- Delete the action
- Duplicate the action
- Assign a color code to the card
- Export a representation of the action to XML or JSON

- Apply relevant constraints (speed, duration) on the action, depending on the type of action

Small cards on the other hand show a more compact view, with the functions other than **Play** hidden. Click **More** to access a contextual menu of the remaining functions.

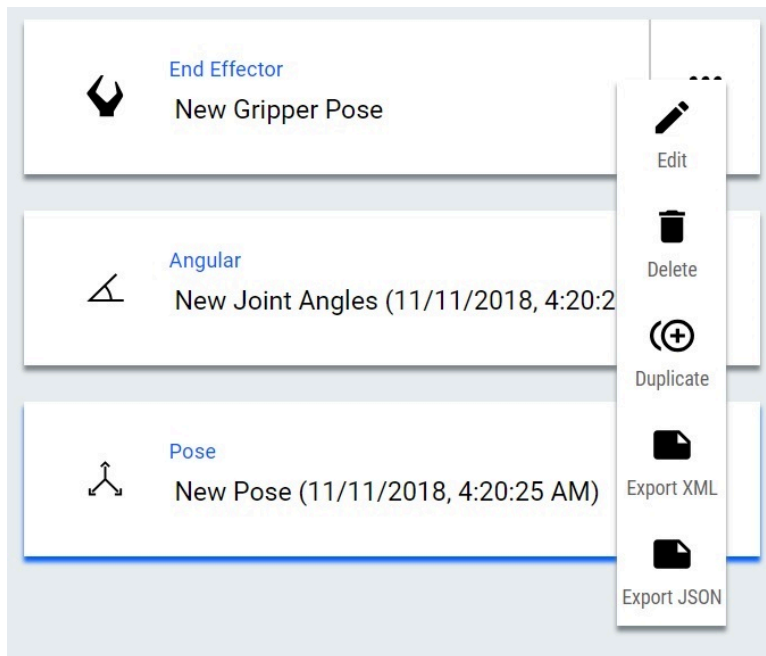


Figure 47: 'More' pop-up menu

Choose **Edit** to modify the parameters of the action.



Figure 48: Parameter modifications

Use the snapshot button () at the bottom of the screen to capture the current robot Cartesian pose, angular positions, or gripper state. Any pose captured by the snapshot tool is displayed on the **Actions** page.

Related topics

[Snapshot tool](#) on page 114

Sequence editor

Build and edit sequences of actions from the **Actions** page in Web App.

Choose **More > Edit** on a sequence on the **Actions** page to launch a sequence editor panel.

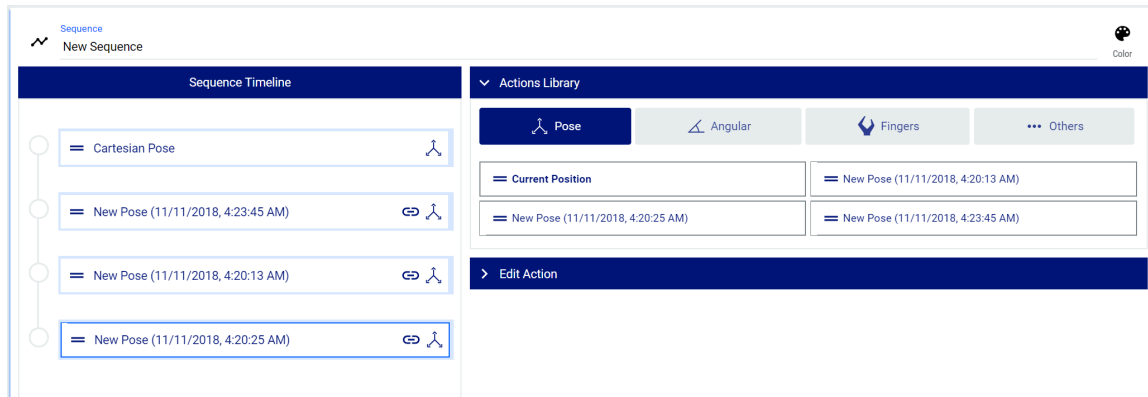


Figure 49: Sequence editor

The sequence editor is made up of three sections.

- **Sequence timeline**
- **Actions library**
- **Edit action**

The **Sequence timeline** displays the steps in the sequence. The link icon () on a step indicates that the action is a reference to an existing standalone action.



Note: If that standalone action is modified outside this sequence it has an impact on the sequence.

The **Actions library** displays the already saved actions, organized into tabs grouped by type. Click one of the available actions to add it to the sequence. There is also an option to add the **Current Position** to the sequence. Depending on the **Actions** page that is activated, the current position as a Cartesian pose, as joints position, or as gripper fingers position is added.

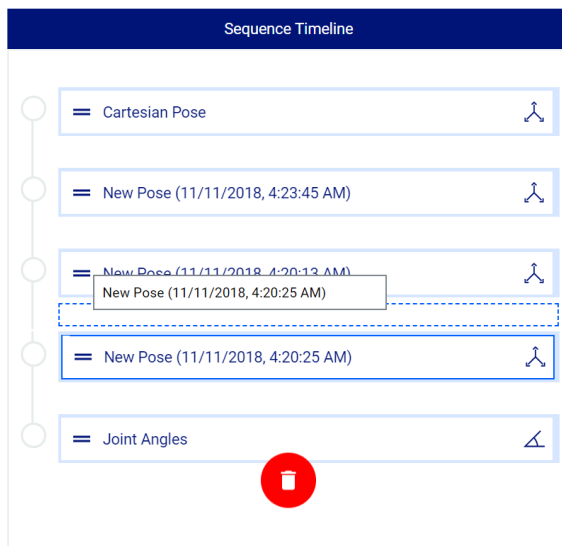
Directly edit the parameters of the action in the **Edit action** pane.

The link icon indicates that the action referred to in the Sequence is a reference to an existing standalone action. If that standalone action is modified outside this sequence it has an impact on the sequence. By opposition, if the action, such as the first one in the screen capture, does not have that link icon, it means that it does not exist outside the Sequence.

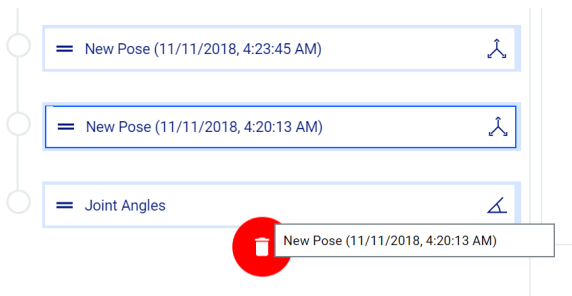
Deleting actions or re-ordering the steps of a sequence

To change the position of an action in the sequence, or remove it from the sequence entirely, simply hover over the left side of the box for the action until the cursor changes from a hand icon to a two-dimensional arrow icon.

Click and drag to move the selected action. To move it to another position, drag the action and release it in the desired position. The dotted outline indicates where the action is in the sequence.



Notice that a red garbage can icon also appears when an action is selected to move. To delete the action from the sequence, drag the action onto the garbage can until the garbage can visibly expands.



Then release to delete.

Importing and exporting actions or sequences

Import and export actions or sequences from the **Actions** page.

All defined actions or sequences can be exported to XML or JSON, to share with others. Similarly, a JSON or XML action file can be imported from the computer. The  **Export All** and  **Import** functions are available at the top of the main panel.

Playing back actions and sequences

Play back actions and sequences from the **Actions** page. Use the controls to manage the playback.

When an action or sequence is selected by clicking on its card, the item is loaded in a playback bar at the bottom of the page.



Figure 50: Actions playback bar

When the play button is pressed, the robot moves directly to execute the described action or sequence.

The **Hold to Play** toggle controls the playback; by default, it is activated. When the toggled on, the playback continues as long as the play button is held down. When not toggled on, a single press of the play button suffices for the playback to execute completely.

The **Play** button plays the sequence.

The **Pause** button stops the playback while keeping the playhead at the same [position](#). The **Play** button changes to a Pause button while the sequence is playing. When the **Play** button is pressed again, the motion continues exactly where it left off.

The **Stop** button stops the movement and returns the playhead to the beginning.

In the case of a Cartesian [pose](#), angular position, or gripper position, the robot, or gripper, interpolates linearly between the present position and the target position and move smoothly and directly to the target position.

For a sequence, the robot first goes directly to the first item in the sequence, and then traces out a [path](#) that goes through the positions on the sequence. A progress bar above the playback bar shows the progress of the playback through the steps.

For sequences, an additional **Loop toggle** control can be toggled on or off. When toggled on, a sequence plays through all the steps and then goes directly to the pose of the first step. This is useful for demonstrations.

Protection Zones

Use the **Protection Zones** page of Web App to define zones around the robot where movement is prevented or limited.



The **Protection Zones** page allows user to define three-dimensional geometric volumes about the robot where the robot either cannot go or where the maximum speed is reduced. A protection zone is intended to limit the possibility of the robot running into either the user or objects near the robot. Protection zones only work when controlling the robot in Cartesian mode; when controlling the robot in Angular mode, they are ignored.

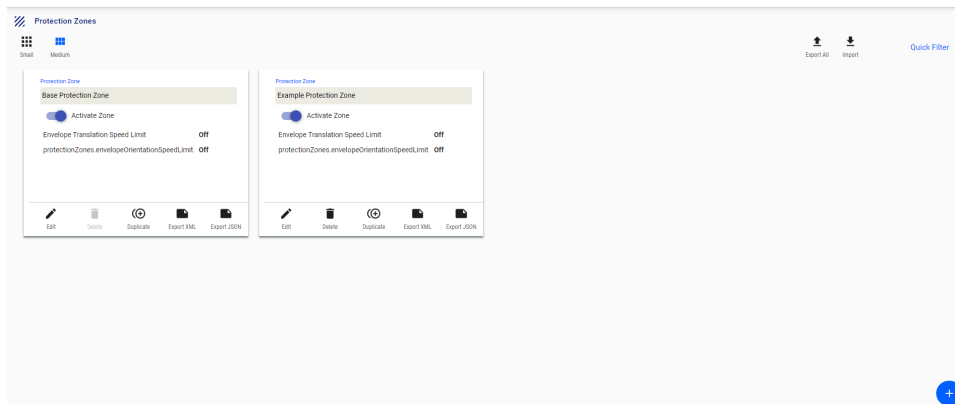


Figure 51: Protection Zones page

Define multiple protections zones from the **Protection Zones** page. The defined protection zones are displayed as cards in the main information panel. The cards can be displayed in two different sizes.

- Small
- Medium

Toggle the card sizes using the buttons at the top of the main information window. The medium cards show fuller detail about the settings for the protection zone, namely whether the **Envelope Translation Speed Limit** and **Envelope Orientation Speed Limit** are turned on. It also includes controls to activate, **Edit**, **Delete**, and **Duplicate** protection zones.

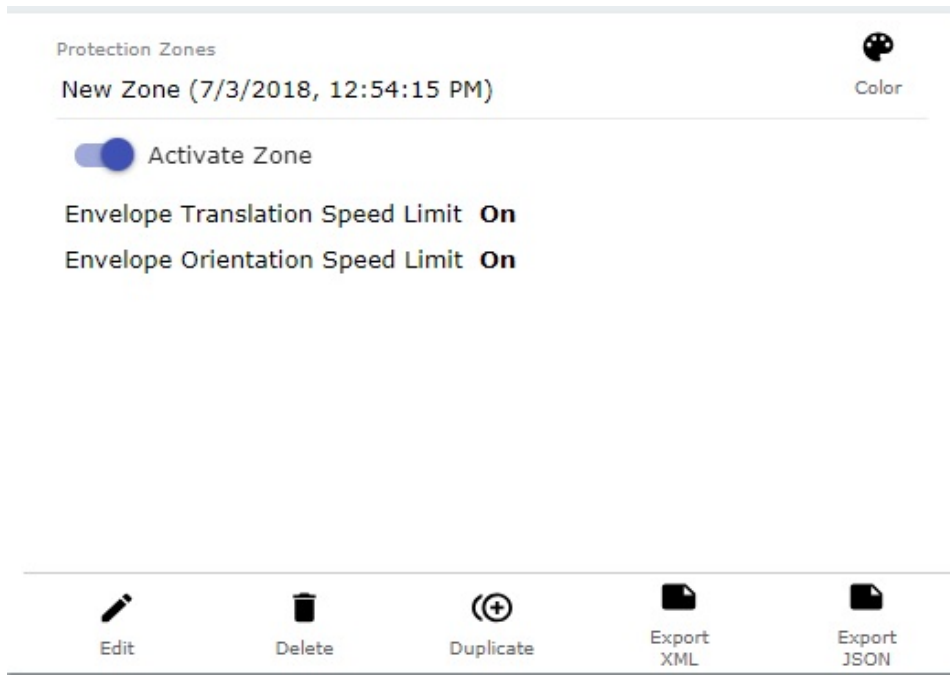


Figure 52: Protection Zone card

Small cards show a more compact view. Only the name of the protection zone and a toggle to activate the zone are displayed. Click **More** to launch a contextual menu to take additional action, such as edit, delete, or duplicate, the selected protection zone.

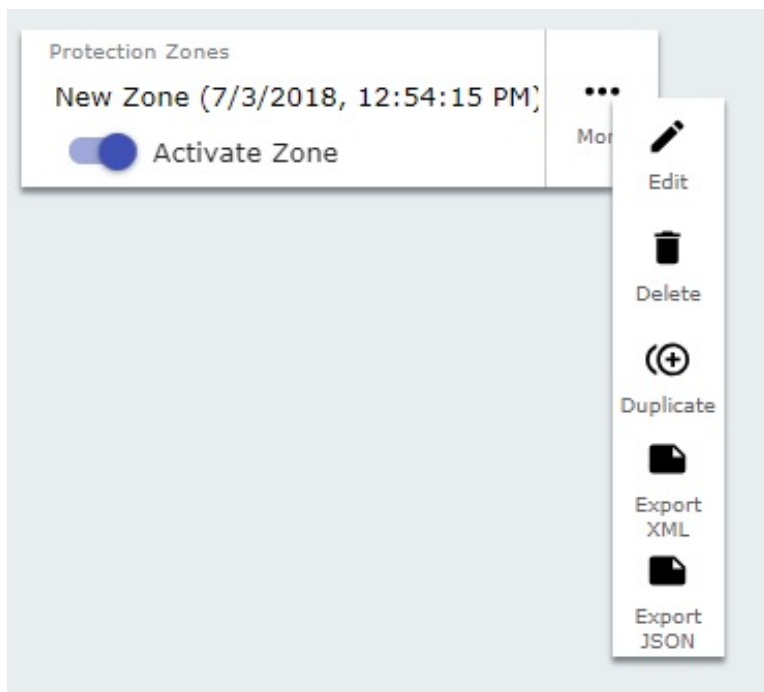


Figure 53: More menu

Multiple protection zones can be active at the same time.

A new protection can be added by clicking the **+** sign icon in the lower right of the main information panel. This creates an empty protection zone and adds a new card to the page. Click **Edit** to configure the protection in a new interface window.

There are two available panes: **Zone Shape** and **Limits**.

Zone Shape pane

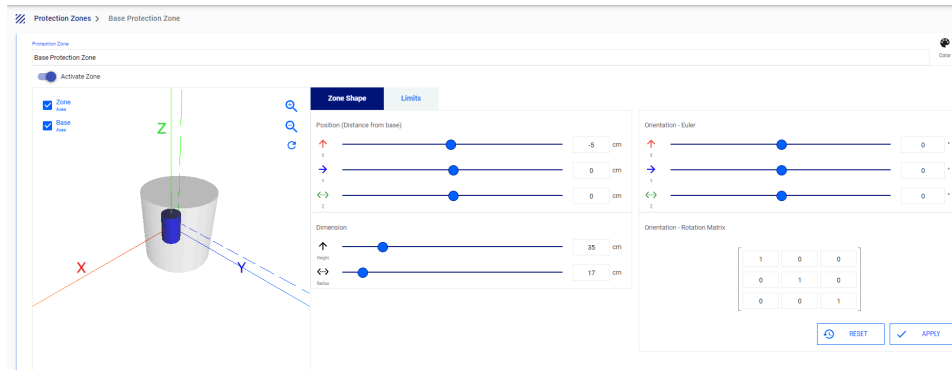


Figure 54: Configure protection zone: Zone Shape pane

Click **Zone Shape** to access the **Zone Shape** pane and define the protection zone geometry. The geometry is the shape of the protection zone. There are three available shapes.

- Rectangular prism
- Sphere
- Cylinder

The dimensions of each of the shapes is in cm and is configurable. The shape determines the configurable dimension.

Table 55: Definition of dimension according to geometry

Geometric shape	Dimension definition
Rectangular prism	Height, width, and depth
Sphere	Radius
Cylinder	Height and radius

An additional thickness (in cm) can be defined around the outside of the protection zone where motion is allowed, but where velocity is limited.

By default, the protection zone is centered on the robot base, with its z-axis aligned with the base z-axis. The default protection zone center is also configurable. The center of the protection zone can be displaced in the x, y, and z directions. It can also be rotated around each of the base axes.

Limits pane

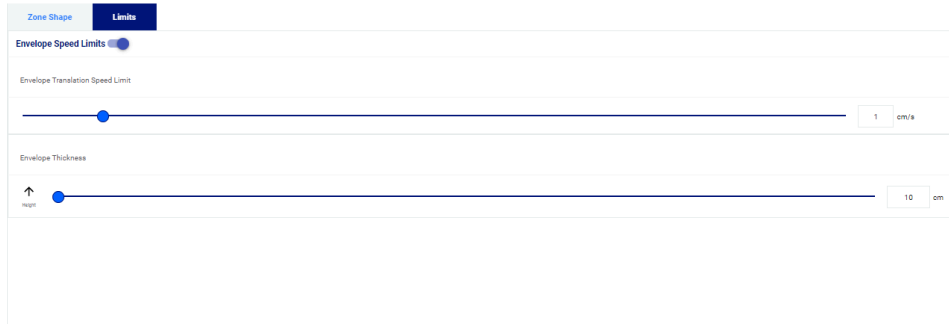


Figure 55: Configure protection zone: Limits pane

Click **Limits** to access the **Limits** pane and set limits on velocity in an envelope surrounding the shape.

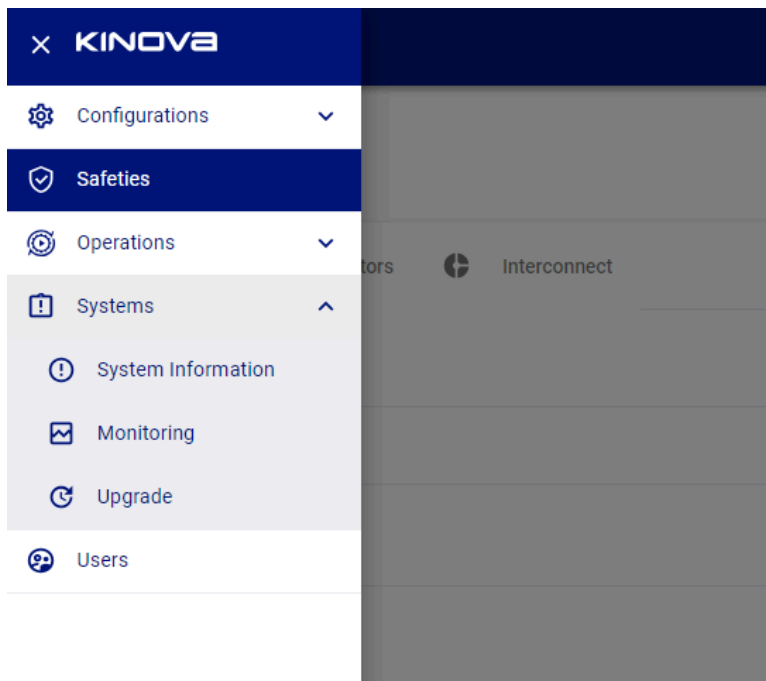
Set the speed limitation by using the **Envelope Translation Speed** slider to a value between 0 cm/s and 10 cm/s.

Use the **Envelope Thickness** slider to configure the height of the envelope between 0 cm and 1000 cm.

Systems page group

The Systems page group of Web App groups together pages to update and view status and configuration information for the system.

Figure 56: Systems page group



The **Systems** page group contains pages for robot devices information, monitoring, and software and firmware upgrades.

- **System Information**
- **Monitoring**
- **Upgrade**

System Information

View high-level hardware and firmware details for the robot from the **System Information** page of Web App.

The **System Information** page gives a quick high level view of hardware and firmware configuration details.

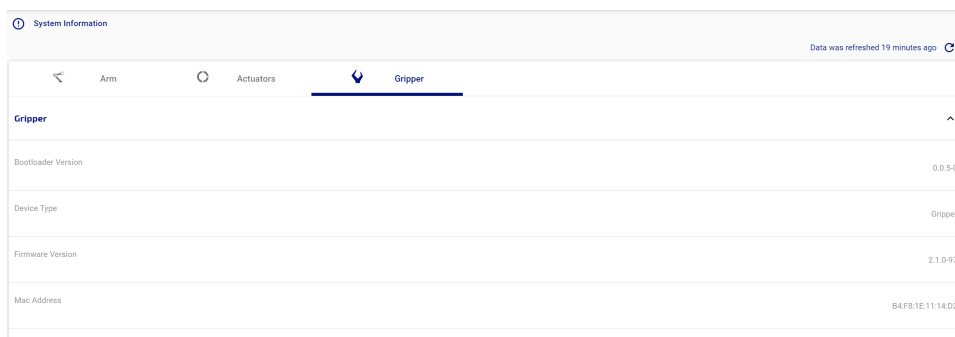


Figure 57: System Information page

The information on the page is grouped into different pages.

- **Arm** (product and base)
- **Actuators** (for each of the individual actuators in robot)
- **Gripper**

The type of information on each page is specific to the part of the system they represent.

- Bootloader version
- Device type
- Firmware version
- MAC address
- Part number, part number revision, and serial number

Product gives product-level information.

- KIN (Kinova information number)
- Model ID
- Model year
- Assembly plant
- Degrees of freedom
- Base type
- End effector type
- Arm laterality
- Wrist type

Monitoring

View real-time monitoring information about the status and performance of the robot from the **Monitoring** page of Web App.



The **Monitoring** page allows for real-time monitoring of status and performance information for the robot. The **Monitoring** page is the first page that opens when opening a new session using Web App.

Monitoring

Overview

Detailed

Download Data

Base

Operating Mode

Running

Control Mode

Idle

Servoing Mode

Single Level

Actuators

Enabled

Actuator #1

Actuator #2

Actuator #3

Actuator #4

Actuator #5

Actuator #6

Position

348.987°

355.168°

155.812°

60.987°

355.302°

279.885°

Torque

-0.044 Nm

-0.048 Nm

-0.027 Nm

0.037 Nm

0.017 Nm

0.033 Nm

Velocity

-0.003°/s

0°/s

-0.007°/s

0.002°/s

-0.013°/s

0.006°/s

End Effector

X

Y

Z

Tool X

Tool Y

Tool Z

Tool Tool-Limit X

Tool Tool-Limit Y

Tool Tool-Limit Z

Tool Tool-Angle X

Tool Tool-Angle Y

Tool Tool-Angle Z

Position

Velocity

0.129 M

0.034 M

0.046 M

0.087°

145.754°

13.933°

Branch

Branch

Branch

0.087°/s

0.005°/s

0.005°/s

-0.006°/s

0.001°/s

0.002°/s

Motor Current

23.817 A

Motor Temperature

47.089°C

Motor Case Temperature

78.707°C

Figure 58: Monitoring page

The monitoring information is divided into sections.

- Base
- Actuators
- End effector

There are two views available.

- Overview
- Detailed

Table 56: Details of the Monitoring page

Overview page information	Detailed page information
Base	
Operating mode	Operating mode
Example: Maintenance, update, shutting down, run, in fault	Example: Maintenance, update, shutting down, run, in fault
Control mode	Control mode
Example: Angular joystick, Cartesian joystick, torque control	Example: Angular joystick, Cartesian joystick, torque control
<i>servoing</i> mode	Servoing mode
Example: single level (high level), low level	Example: single level (high level), low level
	Arm voltage (V)
	CPU temperature (°C)
	Acceleration x, y, z of the base in m/s ²

Overview page information	Detailed page information
	Angular velocity x, y, z of base in °/s
	Commanded tool pose: x, y, z, θ_x , θ_y , θ_z
Actuators for each joint	
Measured position in °	Measured position in °
Measured torque in N·m	Measured velocity in °/s
Measured velocity in °/s	Measured torque in N·m
	Motor current in A
	Voltage in V
	Motor temperature in °C
	Core temperature in °C
End Effector	
Position and orientation: x, y, z, θ_x , θ_y , θ_z	Position and orientation: x, y, z, θ_x , θ_y , θ_z
Velocity: x, y, z, θ_x , θ_y , θ_z	Tool Twist: x, y, z, ω_x , ω_y , ω_z
Tool Twist: x, y, z, θ_x , θ_y , θ_z	
Snapshot button	
Click the Snapshot button to export a snapshot of the currenting monitoring data for the robot.	
The snapshot of the monitoring data is saved locally on the computer in JSON format. Share the monitoring data with Kinova support for troubleshooting purposes.	

Upgrade

Update the software of the robot with a downloaded software package from the **Upgrade** page of Web App.



The **Upgrade** page provides a simple interface to perform upgrades to the robot.

Robot upgrade files are bundled as a package (.swu file).

The robot [upgrade package](#) includes three main parts.

- Robot devices firmware updates:
 - Base controller
 - Actuators
- Web App upgrade package

- Kinova® Kortex™ **API** upgrade package

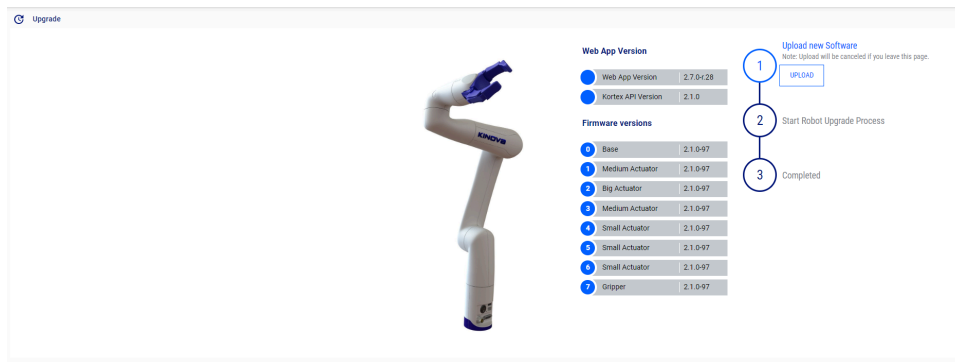


Figure 59: Upgrade page

The **Upgrade** page provides an interface to upload a new upgrade package and initiate the upgrade.

The page also provides information on the current Web App and Kinova® Kortex™ API versions, as well as the current firmware versions of the robot devices.

Upgrading the robot firmware and software

Upgrade the robot firmware and software using Web App.

Before you begin

Make sure the prerequisites are met before proceeding.

- A new robot update package needs to have been previously downloaded to the development computer.
- The development computer needs to be connected to the robot, either with a wired Ethernet connection or with Wi-Fi.
- The user needs to have a Web App session open on the robot.

About this task

Web App is used to upgrade the robot firmware and software using a new [upgrade package](#) on the development computer. The upgrade package covers all devices in the arm, and all devices are upgraded as part of the upgrade process.



Important: If Wi-Fi is enabled before starting the upgrade process, Wi-Fi stays enabled after the upgrade is completed. If Wi-Fi is disabled before the upgrade process, Wi-Fi stays disabled after the upgrade is completed.

Procedure

1. Browse to the Web App **Upgrade** page.
2. Click the **Upload** button on **Upload New Software**.
3. Browse the development [PC](#) disk to select the new firmware package.

The new package uploads to the robot. If the upload is unsuccessful, an error message is displayed. If the upload is successful, the upgrade process continues.

4. Start the upgrade after the firmware package uploads successfully.



Note: Web App refreshes twice during the upgrade, Login after each refresh.



Note: A message states that the upgrade is in progress when there is an attempt to log in during the upgrade process.

Results

The [LED](#) on the [base](#) of the robot is green when the process is over. Web App also indicates when the process is finished.

Users

Edit user profile information from the **Users** page of Web App.



The **Users** page is used to define, set, and edit user profiles for the robot.

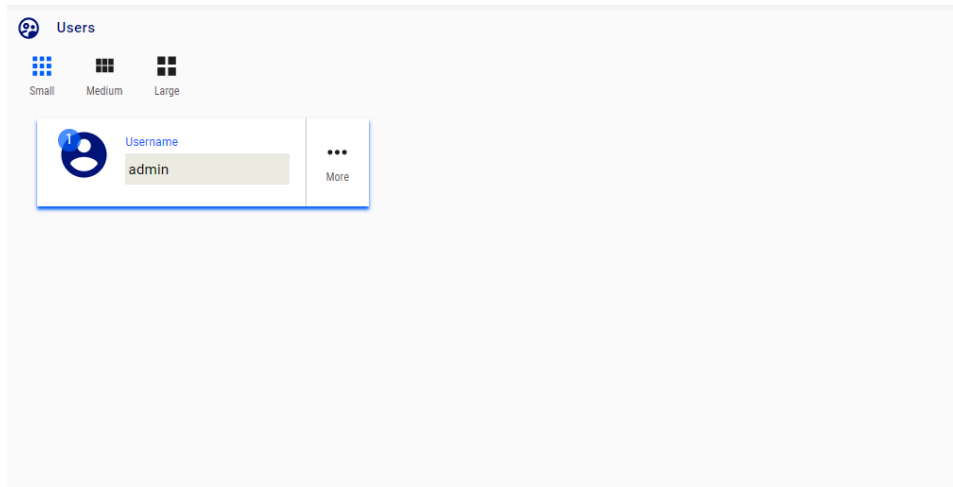


Figure 60: Users page

Defined profiles are displayed as information cards on the main information panel of the page. The cards are in three different sizes.

- Large
- Medium
- Small

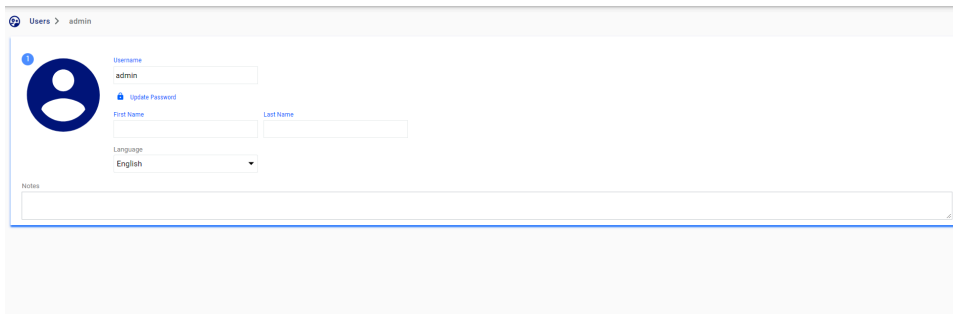
Use the buttons on the upper right of the main information panel to change card sizes.

Large cards show a full set of information. The large card displays the user name and language.

Medium cards are slightly smaller.

Small cards show a more compact view. Click the **More** button, to launch a contextual menu with the options to **View**, **Edit**, **Delete**, or **Duplicate** the profile.

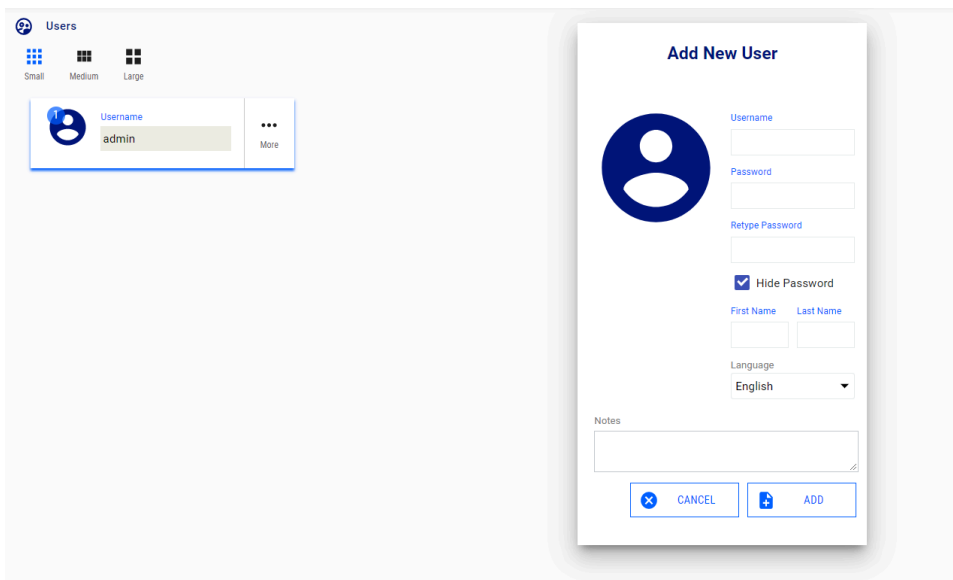
Click **Edit** launches an editing interface where it is possible to configure the profile.



The image shows a web interface for editing a user card. At the top, there's a breadcrumb trail: "Users > admin". Below this, on the left, is a circular profile icon with a blue background and a white person silhouette. To the right of the icon are several input fields: "Username" (containing "admin"), "Update Password" (with a lock icon), "First Name", "Last Name", and "Language" (a dropdown menu currently set to "English"). At the bottom of the card is a "Notes" section with a text area.

Figure 61: Card editing interface

Click on the  button in the lower right corner of the page to create a new user. A menu to add information for the new user launches.



The image shows the "Add New User" modal dialog. On the left, a sidebar shows the "Users" section with three view options: "Small", "Medium", and "Large". Below these is a list of users, with "admin" highlighted. To the right of the list is a "More" button. The main part of the screen is a modal dialog titled "Add New User". It features a large circular profile icon on the left. To the right of the icon are input fields for "Username", "Password", and "Retype Password". Below these is a checkbox labeled "Hide Password" which is checked. Further down are fields for "First Name" and "Last Name", followed by a "Language" dropdown menu set to "English". At the bottom of the modal is a "Notes" text area. At the very bottom of the modal are two buttons: "CANCEL" and "ADD".

Figure 62: Create new user

Creating a new user profile

Create a user profile in Web App.

About this task

Create user profiles from the **Users** page.

Procedure

1. Click the  button to add a new, empty [user profile](#).
A window to enter information for the profile launches.

Add New User



Username

Password

Retype Password

☒ Hide Password

First Name **Last Name**

Language

Notes

 CANCEL  ADD

2. Enter the information for the user profile, including name, user name, and password.
3. Click **ADD** to create the new user profile.

Results

The new user profile is created. The new user can log in to Web App with the credentials.

API for developers

Develop custom software applications for the robot by using the API.

The robot is enabled by Kinova® Kortex™, the Kinova software framework and application development platform. With this growing and evolving framework, configure and control the robot programmatically, adapting to your specific needs and supporting you in integrating new Kinova products into robotics applications. The Kinova® Kortex™ [API](#) supports multiple robot products from Kinova as a cross-hardware development framework.



Note: Some of the specific features of the API are hardware dependent and may not be available on the robot.

APIs are currently provided for the following languages:

- C++
- Python
- MATLAB®, simplified API supporting a subset of Kinova® Kortex™ functionality

Kinova also offers [Robot Operating System \(ROS\)](#) packages covering most of the same functionalities.

The pages that follow describe the general philosophy and approach of the APIs.

The following GitHub repositories contain additional developer guidance and resources, including detailed API documentation, setup instructions, and source code examples:

- Kinova® Kortex™ API: https://github.com/Kinovarobotics/Kinova-kortex2_Gen3_G3L/
- Kinova® Kortex™ ROS: Kinova® Kortex™ [kinovarobotics/ros_kortex](#)
- Kinova® Kortex™ ROS2: Kinova® Kortex™ [kinovarobotics/ros2_kortex](#)
- Kinova® Kortex™ MATLAB® API: [kinovarobotics/matlab_kortex](#)

Tutorial videos are available on the [Kinova® YouTube channel](#) (<https://www.youtube.com/watch?v=zQewb08M4sA>).

Devices and services

The [API](#) consists of services, which define interfaces implemented and available on the various robot devices.

The robot consists of several devices:

- [base](#) controller
- actuators (each actuator is a distinct device)
- gripper

A service consists of methods and communication exchange data structures. The devices in the robot each implement a particular set of services, some of which are available across multiple devices. The methods available as part of a service on a device are accessed through [remote procedure call \(RPC\)](#).

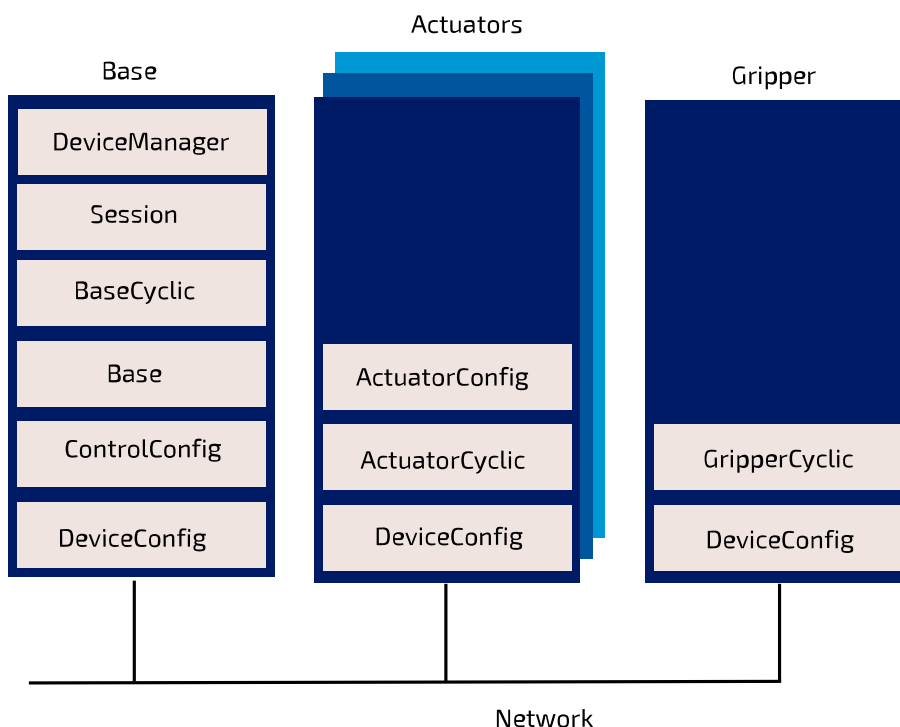


Figure 63: Services on multiple devices

Available services

Kinova makes available a number of services for developers, each of which includes functions and data types supported for C++ and Python.

- `Session`: provides functions for opening and closing sessions with the robot. This service is used at the beginning and end of every session with the robot to authenticate the user.



In practice, end users do not use the Session service directly, but use a SessionManager object. See the GitHub documentation for more details.

- **Base:** broadly useful service. Provides functions for configuring a range of base-related functionalities as well as high-level control for the robot.
- **DeviceManager:** provides a list of device information used for internal communication routing purposes.
- **Cyclic data communications** (sending commands to devices and/or receiving status feedback on a periodic or as-requested basis). Cyclic data communications are used with low-level or low-level bypass [servoing](#), and are intended to be called by [API](#) clients as part of a user-defined control loop through a `UDPTTransportClient` on port 10001.
 - For low-level servoing cyclic communication
 - **BaseCyclic:** sending commands to actuators and gripper motors and obtain feedback from base, actuators, wrist interface, and gripper motors.
 - For low-level bypass servoing cyclic communication
 - **ActuatorCyclic:** sending commands to an actuator and obtaining feedback from the actuator
 - **InterconnectCyclic:** sending commands to interface module and obtaining feedback from the interface module
- **Configuration related**
 - **ControlConfig:** get/set control library configuration
 - **ActuatorConfig:** get/set actuator configuration
 - **DeviceConfig:** get/set general device configuration



For full details on available services, see the Kinova® Kortex™ GitHub repository.

Services, methods, and messages

The [API](#) services offer a set of [RPC](#) and pub/sub methods. The methods exchange data which are structured as Google Protocol Buffer message objects.

Kinova® Kortex™ API and Google Protocol Buffer

The Kinova® Kortex™ API uses the Google Protocol Buffer.

The Kinova® Kortex™ [API](#) is based on the Google Protocol Buffer 3 mechanism for serializing structured data. Using Protocol Buffer, the API is made available in C++ and Python languages.

Developers accustomed to Protocol Buffer can see `.proto` files on the Kinova® Kortex™ GitHub repository. These files are published as a means to document the services and methods offered through the API.

The API data structures are based on Google Protocol Buffer messages. Extensive documentation has been made available by Google explaining the different mechanisms offered to:

- Set a field in a message
- Read a OneOf element in a message
- Go through a nested object
- Set a nested object
- Get/set a collection



For more details on how the above works, check out the following documentation on the Google Protocol Buffer website:

- C++ tutorial: <https://developers.google.com/protocol-buffers/docs/reference/cpp-generated>
- Python tutorial: <https://developers.google.com/protocol-buffers/docs/reference/python-generated>

Service client-server model

Services operate on a client-server model. The server component of the service runs on the device itself. The client component runs on the client computer.

Services offer a set of device functionalities which are transparently exposed to the end-user through [RPC](#) and pub/sub methods.

The [API](#) is built on a transparent client/server communication protocol which allows an end-user (client side) to call methods on robot devices.

Blocking and non-blocking calls

Remote procedure calls on the robot are generally available in both blocking and non-blocking forms. Non-blocking calls are available using Future/Promise and by registering callback functions.

The [API](#) defines interfaces of methods to be executed on devices in the robot.

The methods can be one of two types, depending on what the client application does while waiting for the response:

- [blocking](#)
- [non-blocking](#)

With a blocking call, the flow of the client application pauses and waits for the remote procedure call to return a response before proceeding. With [non-blocking](#) call, the procedure call is sent, and the flow of the application carries on while waiting for the response. When the response arrives, the caller handles the response.

For the Python API, only blocking calls are enabled.

In the context of C++, remote procedure calls in the API can in general be set as either *blocking* or *non-blocking*.

There are two types of non-blocking calls available in the C++ API:

- Future/Promise
- Registered callback functions



For more information on how this works, see the API documentation on the Kinova® Kortex™ GitHub repository.

Device routing and transport

Device routing and low-level transport helps to reduce errors.

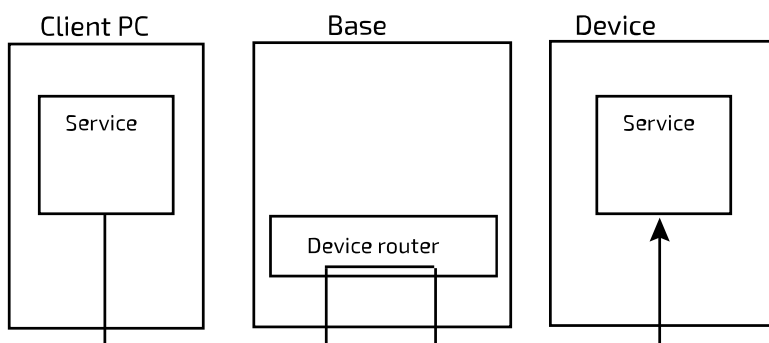


Figure 64: Device routing

Use the [API](#) to communicate with the robot devices. Using a device identifier the RPCs and pub/sub methods of the API are routed by the robot [base](#) and directly bridged to the intended device.

The routing from the client computer, through the base to a device is managed by a `RouterClient` object in the API. A Session on the robot depends on a `RouterClient` object.

`RouterClient` objects depend, for low-level details of the transport of the routed methods, on `Transport` objects. A `Transport` object uses a specific underlying network transport protocol. The API defines two types of `Transport` objects, one using [Transmission Control Protocol \(TCP\)](#) as the protocol, and another using [User Datagram Protocol \(UDP\)](#).

`TCP Transport` objects are used for high-level robot control and configuration.

`UDP Transport` object are used for low-level (real-time) cyclic communication with the robot.



Note: Since Sessions require a `RouterClient`, and a `RouterClient` is tied to a `Transport` object using a specific protocol, separate Sessions must be created if high-level control/configuration and low-level control are used in the same application.



For more details, see the API documentation on GitHub.

Robot servoing modes

There are multiple [servoing](#) modes on the robot.

A servoing mode is a modality through which commands are transmitted to robot devices during operation. Depending on the servoing mode chosen, the details involved in controlling through the [API](#) are different.

There are two servoing modes:

- High-level
- Low-level
- Low-level bypass

High-level servoing

High-level [servoing](#) is the default servoing mode for the robot on bootup.

In high-level servoing, users connect to the [base](#) through the [API](#) (whether directly, or through the Kinova® Kortex™ Web App built on top of the API), sending command inputs. The base routes commands to the actuators, and manages a 1 kHz control loop.

High-level servoing is the recommended servoing mode for non-advanced users.

High-level servoing allows a client to control the robot by sending it a target (angular or Cartesian) [position](#) or velocity through an API method which is sent once (that is, no high frequency client-controlled communication between the client [PCPC](#) and the robot). High level API calls are redirected to the robot control library to calculate inverse kinematics (breaking down the command into commands for actuators) and apply limits ([protection zones](#), singularity management, self-collision avoidance).

The base then manages the execution of the command through the 1 kHz communications with the actuators.

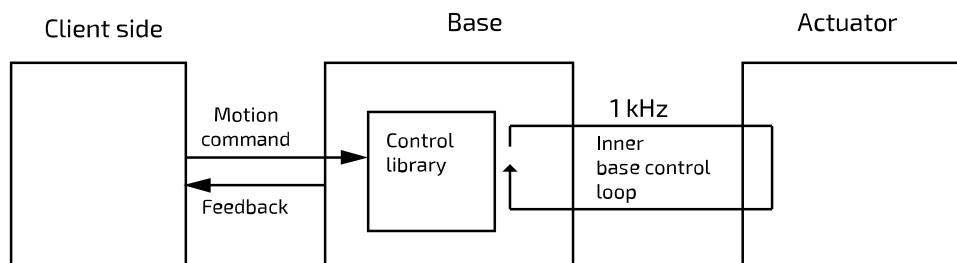


Figure 65: High-level servoing

High-level servoing is the default servoing mode for the robot on bootup. In high-level servoing, users connect to the base through the API (whether directly, or through Web App built on top of the API), sending command inputs. The base routes commands to the actuators, and manages a 1 kHz control loop.

High-level servoing is the recommended servoing mode for non-advanced users. High-level servoing allows a client to control the robot by sending it a target (angular or Cartesian) position or velocity through an API method which is sent once (that is, no high frequency client-controlled communication between the client PC and the robot).

High level API calls are redirected to the robot control library to calculate inverse kinematics (breaking down the command into commands for actuators) and apply limits (protection zones, singularity management, self-collision avoidance).

The base then manages the execution of the command through the 1 kHz communications with the actuators.

Low-level servoing offer lighter and faster API methods, but at the cost of having to manage these details yourself.

Sessions and control permissions

As soon as someone takes control of the robot by sending a control command (whether from API calls, Web App session, or Xbox gamepad input) to the robot, the control mode changes from IDLE to SERVOING. In this mode, control commands from other sessions sent with Web App or API methods are blocked while the control mode is in SERVOING and this session has control. However, after a predefined "grace period" of 7.5 seconds elapses with no new control commands from the user, the robot control mode returns to IDLE and someone else can take control by sending control inputs with Web App or API calls.

Override by physical controls

Physical controls of the robot with a connected Xbox gamepad or the buttons on the robot wrist override user session control of the robot through Web App or API calls. These physical controls always take precedence immediately, without having to wait for the grace period to elapse.

Low-level servoing

In low-level [servoing](#), the API client connects to the [base](#) and sends commands through the base for routing.

The base ensures device routing and internal communications with the actuators, but the high-level functionalities for the base control loop (robot kinematics, [trajectory](#) management, and so on.) are no longer available.

Low-level servoing allows clients to control each actuator individually by sending command increments (bypassing the kinematic control library).

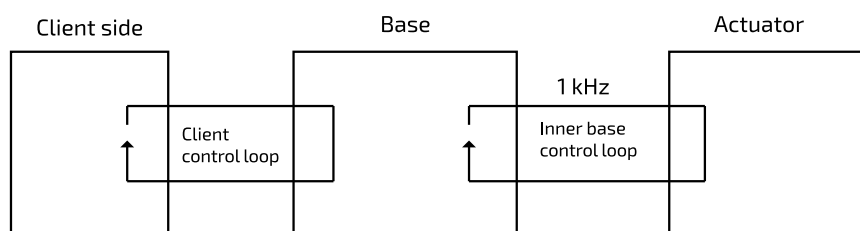


Figure 66: Low-level servoing

Low-level bypass servoing

In low-level bypass servoing mode, the client PC uses [UDP](#) communication directly with the different actuators; there is no device routing involved.

The client [path](#) is able to communicate directly with the actuators using their [IP](#) addresses. Since the control here bypasses the [base](#) completely, the client is fully responsible for managing the control loop.

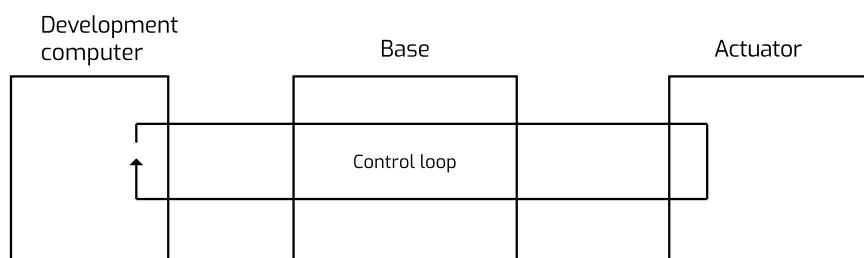


Figure 67: Low-level bypass servoing

Low-level and low-level bypass [servoing](#) offer lighter and faster API methods, but at the cost of having to manage these details yourself.

Notifications

The robot [base](#) can provide [notifications](#) on different topics as requested by a client application that has a session open with the robot.

The robot base uses a Publish/Subscribe design pattern. That is, rather than needing to poll periodically for updates, the client application subscribes to a list of [Topics](#). Whenever a change happens related to that topic, whether caused by the same client session, or another, a publisher sends a notification to all subscribers. Notifications are surfaced to clients through the API, and are also displayed in the **Notifications** page of Web App.

Client applications can also unsubscribe from a topic.



Methods for subscribing and unsubscribing from notification topics are described in the API documentation on the Kinova® Kortex™ GitHub repository.

Error management

When an [API](#) method is called, sometimes an error results.

There are three main categories of errors:

- Protocol server errors

- Protocol client errors
- Device errors

The first two categories of errors include all errors relating to the internal communication protocol. (ex: invalid, unsupported or unknown calls, out of session call, and so on.)

The other category is for errors coming from the target device.

For each high level category, there are also more detailed and specific errors.



For more information about the error codes that can be produced, see the Kinova® Kortex™ GitHub documentation.

Error codes

Some [API](#) calls result in errors. When an error occurs, an exception is thrown which includes information about the type of error.

The error information includes at minimum an Error Code and a Sub-error code.

The error code identifies the general type of error, while the sub-error code provides more detailed information.

The following reference tables provide guidance in interpreting error codes.

Table 57: Error codes description

Name	Description
SUCCESS	No errors
ERROR_PROTOCOL_SERVER	Protocol buffer server error
ERROR_PROTOCOL_CLIENT	Protocol buffer client error
ERROR_DEVICE	Device error

Sub-error codes description

Table 58: General (used when there are no more specific error codes available)

Name	Description	Why is error reported?
SUB_SUCCESS	No errors	Sub-error code is set to sub-success when error code is set to success

Name	Description	Why is error reported?
FAILED	Failure	Used to report an error for which there is no more specific sub-error code
UNIMPLEMENTED	Unimplemented method	Reports back that the remote procedure call is not implemented on the robot, though it is available through the service
INVALID_PARAM	Invalid parameter	The remote procedure call is supported, but parameter is considered invalid by the robot (that is, outside admissible range)

Table 59: Protocol Server (reported by the protocol server)

Name	Description	Where is error reported?
UNSUPPORTED_SERVICE	Service not recognized by protocol server	Rx side
UNSUPPORTED_METHOD	Function not recognized by protocol server	Rx side
TOO_LARGE_ENCODED_FRAME_BUFFER	Encoded frame larger than what transport permits	Tx side
TOO_LARGE_ENCODED_PAYLOAD_BUFFER	Encoded payload is larger than what transport permits	
FRAME_ENCODING_ERR	Unable to encode frame	Tx side
FRAME_DECODING_ERR	Unable to decode frame	Rx side
INCOMPATIBLE_HEADER_VERSION	Frame header version differs from what is expected by protocol server	Rx side
UNSUPPORTED_FRAME_TYPE	Frame type not recognized by protocol server	Rx side
UNREGISTERED_NOTIFICATION_RECEIVED	Protocol server receiving unregistered notification	
PAYLOAD_DECODING_ERR	Unable to decode payload	
INVALID_SESSION	Session not recognized by protocol server	Rx side
INVALID_DEVICE	Device ID when bridging is not found or already in use.	

Table 60: Protocol Client (reported by the protocol client)

Name	Description	Where is error reported?
TOO_LARGE_ENCODED_FRAME_BUFFER	Encoded frame larger than what transport permits	Tx side
TOO_LARGE_ENCODED_PAYLOAD_BUFFER	Encoded payload is larger than what transport permits	
FRAME_ENCODING_ERR	Unable to encode frame	Tx side
FRAME_DECODING_ERR	Unable to decode frame	Rx side
INCOMPATIBLE_HEADER_VERSION	Frame header version differs from what is expected by protocol client	Rx side
UNSUPPORTED_FRAME_TYPE	Received frame type is not recognized by the protocol client (or unexpected, unknown, or unsupported)	Rx side
UNREGISTERED_NOTIFICATION_RECEIVED	Protocol client received notification for which it does not have a registered notification to call back	
PAYLOAD_DECODING_ERR	Unable to decode payload	
UNREGISTERED_FRAME_RECEIVED	Client received a response that it was not expecting (that is, for which it did not send a remote procedure call)	

Robot - the following are sub-error codes that can be reported by the robot

Table 61: Session/users

Name	Description	When is error reported?
INVALID_PASSWORD	Password does not match specified user	<code>createSession()</code> , <code>changePassword()</code> function calls
USER_NOT_FOUND	Unrecognized user	<code>createSession()</code> function call

Table 62: Configuration

Name	Description	When is error reported?
ENTITY_NOT_FOUND	Operation failed because could not retrieve entity	<code>readX()</code> , <code>updateX()</code> , and <code>deleteX()</code> function calls

Table 63: Control/sequence

Name	Description	When is error reported?
ROBOT_MOVEMENT_IN_PROGRESS	When robot refuses the new control command because robot movement is in progress	<code>executeAction()</code> , <code>playX()</code> , <code>sendTwistX()</code> , <code>sendWrenchX()</code> function calls
ROBOT_NOT_MOVING	When attempting to stop a robot movement when it is not yet moving	<code>stopAction()</code> , <code>stop()</code> , <code>pause()</code> , <code>pauseAction()</code> function calls

Table 64: Configuration backup

Name	Description	When is error reported?
NO_MORE_STORAGE_SPACE	Unable to execute action because no storage left	<code>createConfigurationBackup()</code> function call

Table 65: Operating mode

Name	Description
ROBOT_NOT_READY	When sending a command while the robot initialization is not complete
ROBOT_IN_FAULT	When sending a command while the robot is in a fault
ROBOT_IN_MAINTENANCE	When sending a command while the robot is in maintenance
ROBOT_IN_UPDATE_MODE	When sending a command while the robot is in update
ROBOT_IN_EMERGENCY_STOP	When the robot was manually put into an emergency stop

Table 66: Servoing

Name	Description
SINGLE_LEVEL_SERVOING	A second client is attempting to control the robot while it is in single-level servoing mode
LOW_LEVEL_SERVOING	Attempting to send high-level control to the robot while it is in low-level servoing mode

Table 67: Mapping

Name	Description
MAPPING_GROUP_NON_ROBOT	Trying to add non-root MapGroup to Mapping
MAPPING_INVALID_GROUP	Trying to add invalid or non-existent MapGroup to Mapping
MAPPING_INVALID_MAP	Trying to add invalid or non-existent Map to Mapping
MAP_GROUP_INVALID_MAP	Trying to add invalid or non-existent Map to MapGroup
MAP_GROUP_INVALID_PARENT	Trying to add MapGroup under invalid parent
MAP_GROUP_INVALID_CHILD	Trying to add invalid or non-existent child to MapGroup
MAP_GROUP_INVALID_MOVE	Trying to change MapGroup's parent: move not supported
MAP_IN_USE	Deleting a Map used in Mapping or MapGroup

Table 68: Network

Name	Description
WIFI_CONNECT_ERROR	Unable to connect to Wi-Fi network
UNSUPPORTED_NETWORK_TYPE	Unsupported network type

Kinova® Kortex™ GitHub repository

The repository offers access to a number of resources for developers.

Table 69: Resources available from the repository

Setup instructions and release notes
Detailed API documentation
C++ and Python Code examples



For more detailed information about developing applications using the API, visit the Kinova® Kortex™ GitHub repository at https://github.com/Kinovarobotics/Kinova-kortex2_Gen3_G3L3.

Kinova® Kortex™ ROS and ROS2 packages and GitHub repository overview

The Kinova® Kortex™ ROS and ROS2 packages `ros_kortex` and `ros2_kortex` are the official repository containing ROS packages to interact with Kinova® Kortex™ and related products. It consists of a number of ROS packages built on top of the client Kinova® Kortex™ [API](#).

The [ROS](#) packages are designed to work with ROS Melodic Morenia, ROS Noetic Ninjemys and ROS2 Humble.

ROS Melodic is compatible with Ubuntu 18.04 (Bionic)

Methods provided by the underlying API are offered as ROS services and [topics](#), depending on the method.

- [RPC](#) methods are exposed through ROS services
- pub/sub methods are exposed through ROS topics

The ROS Messages correspond to the message type definitions of the underlying API.

The ROS interface can be accessed using either Python (rospy) or C++ (roscpp).

Support is included for Gazebo and MoveIt.

Detailed documentation of the packages is available on the Kinova GitHub repository at github.com/kinovarobotics/ros_kortex, or github.com/Kinovarobotics/ros2_kortex for ROS and ROS2 packages respectively.

The repository includes various packages related to ROS development:

- Setup instruction and release notes
- `kortex_api` (package containing header files and libraries needed to use the C++ Kinova® Kortex™ API)
- `kortex_control` (package contains configuration files for the `ros_control` controllers used to control the simulated robot)
- `kortex_description` (package contains URDF and STL files of the robot)
- `kortex_driver` (ROS node package to allow direct communication with the robot [base](#))
- `kortex_examples` (examples needed to understand the basics of `ros_kortex`)
- `kortex_gazebo` (package contains files to simulate the robot)
- `kortex_moveit_config` (contains all the auto-generated MoveIt! configuration ROS packages)



Important: Low-level control is not available through ROS because ROS is not capable of handling the required 1 kHz communication rate.

Kinova® Kortex™ MATLAB® API and GitHub repository overview

The Kinova® Kortex™ MATLAB® [API](#) provides a wrapper giving access, using MATLAB® and Simulink®, to a subset of the Kinova® Kortex™ API functionality.

This package has been tested on Ubuntu 16.04 (64-bit) and Windows 10.

More detailed information is available on the `matlab_kortex` GitHub repository at github.com/kinovarobotics/matlab_kortex.

Guidance for advanced users

Advanced users of the robot may want to take advantage of additional information, such as the robot frames, homogeneous transforms, and other information.

Standard robot frames

The robot has two standard frames: base frame, also known as base reference frame, and tool frame, also known as end effector reference frame.

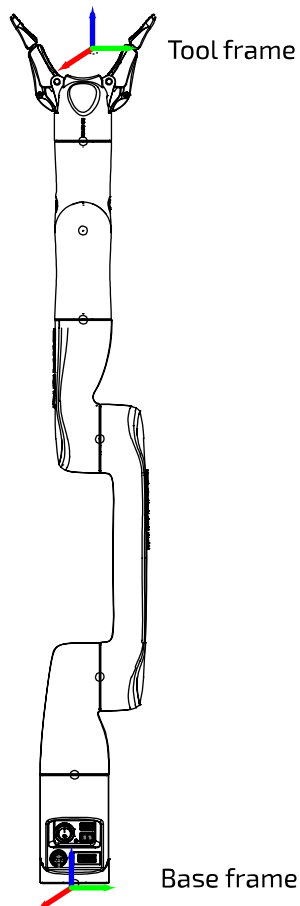


Figure 68: Standard frames

Different control modes make use of different frames.

Homogeneous transforms

Homogeneous transforms for the robot are provided. Homogeneous transforms define transformations between successive reference frames in the robot, allowing coordinates in one frame to be expressed in terms of another.

The forward kinematics of the robot are determined by homogeneous transform matrices. These matrices represent the transformations from one *frame* (*base*, *joint*, or *interface*) to the next along the kinematic chain. These transformations allow for the coordinates in one frame to be expressed in terms of another.

The overall transformation from the *base frame* to the *tool frame* is given by:

$${}^B T_{TOOL}^* = {}^B T_1^* T_2^* T_3^* T_4^* T_5^* T_6^* T_{TOOL}^*$$

Here:

$${}^{i-1} T_i^* = {}^{i-1} T_i^* R_z(q_i)$$

Where:

${}^{i-1} T_i^*$ is the matrix for the general transformation matrix from frame [i-1] to frame [i].

${}^{i-1} T_i$ is the transform from the previous frame [i-1] to the current frame [i] when q_i , the angle for joint i, is 0.



Note: q_i is the angle in radians.

$R_z(q_i)$ is the transformation matrix for a rotation of q_i around joint i; the z axis for the joint frame is always defined to be along the joint axis of rotation.

$$R_z(q_i) = \begin{pmatrix} c q_i & -s q_i & 0 & 0 \\ s q_i & c q_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$c q_i = \cos(q_i)$$

and

$$s q_i = \sin(q_i)$$

Homogeneous transform matrices

Refer to the homogeneous transform matrices of the robot.

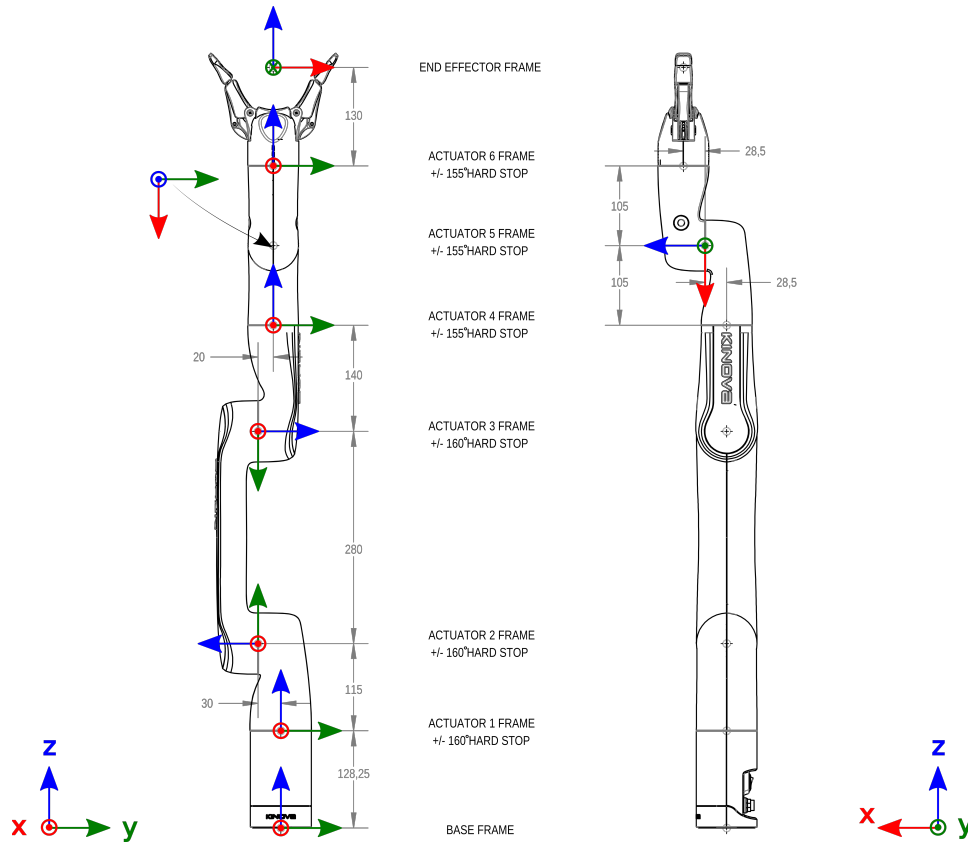


Figure 69: Frame definitions and dimensions (all joints at 0 position, dimensions in mm)

Table 70: Transformation matrices

Transformation	${}^{i-1}\mathbf{T}_i$	${}^{i-1}\mathbf{T}_i^*$
Base to frame 1	${}^B T_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.1283 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^B T_1^* = \begin{pmatrix} c q_1 & -s q_1 & 0 & 0 \\ s q_1 & c q_1 & 0 & 0 \\ 0 & 0 & 1 & 0.1283 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
Frame 1 to frame 2	${}^1 T_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -0.0300 \\ 0 & 1 & 0 & 0.1150 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^1 T_2^* = \begin{pmatrix} c q_2 & -s q_2 & 0 & 0 \\ 0 & 0 & -1 & -0.0300 \\ s q_2 & c q_2 & 0 & 0.1150 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

Transformation	${}^{i-1}\mathbf{T}_i$	${}^{i-1}\mathbf{T}_i^*$
Frame 2 to frame 3	${}^2T_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0.2800 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^2T_3^* = \begin{pmatrix} cq_3 & -sq_3 & 0 & 0 \\ -sq_3 & -cq_3 & 0 & 0.2800 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
Frame 3 to frame 4	${}^3T_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -0.1400 \\ 0 & 1 & 0 & 0.0200 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^3T_4^* = \begin{pmatrix} cq_4 & -sq_4 & 0 & 0 \\ 0 & 0 & -1 & -0.1400 \\ sq_4 & cq_4 & 0 & 0.0200 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
Frame 4 to frame 5	${}^4T_5 = \begin{pmatrix} 0 & 0 & 1 & 0.0285 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0.1050 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^4T_5^* = \begin{pmatrix} 0 & 0 & 1 & 0.0285 \\ sq_5 & -cq_5 & 0 & 0 \\ -cq_5 & sq_5 & 0 & -0.1050 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
Frame 5 to frame 6	${}^5T_6 = \begin{pmatrix} 0 & 0 & -1 & -0.1050 \\ 0 & 0 & -1 & 0.10593 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	${}^5T_6^* = \begin{pmatrix} 0 & 0 & -1 & -0.1050 \\ sq_6 & cq_6 & 0 & 0 \\ cq_6 & -sq_6 & 0 & 0.0285 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
Frame 6 to gripper	${}^6T_{EE} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.1300 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	



Note: units are in meters for homogeneous transform translations in the right-hand column of each matrix.

Denavit-Hartenberg (DH) parameters

Denavit-Hartenberg (DH) parameters offer another convenient way to specify the reference frame transformations for the robot kinematic chain.

The Classical DH parameters for the transformation between frames are defined as:

$${}^i T_{i+1} = \begin{pmatrix} \cos(\theta_i) & -\cos(\alpha_i)\sin(\theta_i) & \sin(\alpha_i)\sin(\theta_i) & a_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\alpha_i)\cos(\theta_i) & -\sin(\alpha_i)\cos(\theta_i) & a_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The following table gives the Classical DH parameters for the robot.

Table 71: Classical DH parameters

i	α_i	a_i (mm)	d_i (mm)	θ_i
1	$\pi/2$	0.0	(128.3 + 115.0)	q_1
2	π	280.0	30.0	$q_2 + \pi/2$
3	$\pi/2$	0.0	20.0	$q_3 + \pi/2$
4	$\pi/2$	0.0	(140.0 + 105.0)	$q_4 + \pi/2$
5	$\pi/2$	0.0	(28.5 + 28.5)	$q_5 + \pi$
6	0	0.0	(105.0 + 130.0)	$q_6 + \pi/2$

Here q_i refers to the joint angles.

The DH parameters use the following frames definitions.

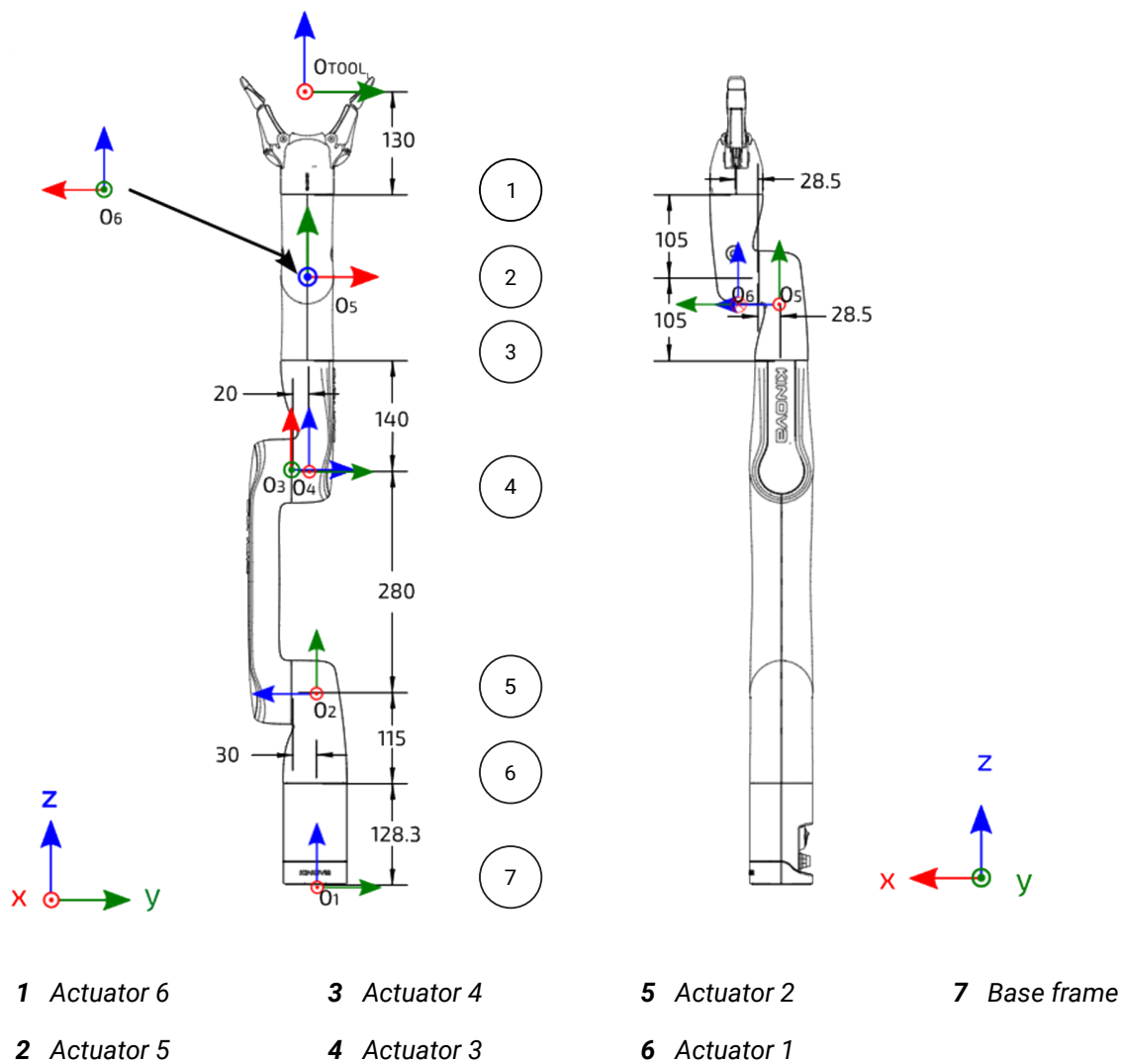


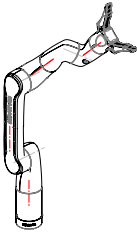
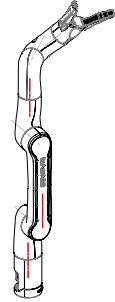
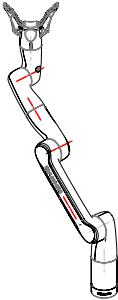
Figure 70: DH parameters frames definitions

Singular configurations

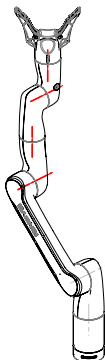
Singularities generally occur when a particular angular configuration of the robot causes axes to be aligned, causing the robot to lose degrees of freedom and experience limitations in movement in some directions while operating the robot in Cartesian mode.

There are many ways that this could potentially happen, and an exhaustive listing would be difficult. The following table highlights some important classes of singularities for the robot, explaining how they occur and how the robot behavior is altered near the singularity while in Cartesian mode.

Table 72: Selected singularity configurations description

Singularity class	Definition	Description
Joint angles shown: [45, 0, 45, 45, 0, 45]	$q_2 = 0$ and $q_5 = 0$	The axis of joint 1 is aligned with the joint 2 - joint 3 link. The axes of joints 4 and 6 are also aligned.
		
Joint angles shown: [45, 0, 0, 45, 45, 45]	$q_2 = 0$ and $q_3 = 0$	There is a 3-way alignment between the axis of joint 1, the joint 2 - joint 3 link, and the axis of joint 4.
		
Joint angles shown: [45, 45, 0, 90, 45, 45]	$q_3 = 0$ and $q_4 = \pi/2$	There is an alignment between the joint 2 - joint 3 link and the axis of joint 4. The axes of joints 3 and 5 are also aligned.
		

Singularity class	Definition	Description
Joint angles shown: [45, 45, 45, 90, 0, 45]	$q_4 = \pi/2$ and $q_5 = 0$	The axes of joints 3 and 5 are aligned. The axes of joints 4 and 6 are aligned.



Inertial parameters definition

Inertial parameters are provided for the robot.

The key inertial parameters of each independently moving rigid link segment of the robot.

- Mass in kg
- Centers of masses in meters
- Moments of inertia in $\text{kg} \cdot \text{m}^2$

The six distinct moments of inertia (I_{xx} , I_{xy} , I_{xz} , I_{yy} , I_{yz} , I_{zz}) are presented in tabular form.

Conventions used

1. The *center of mass* of a link is always expressed in terms of the reference *frame* of the precedent *joint*.
2. The mass of a link segment includes the shell and portions of the actuators at each end of the link (as applicable) that are enclosed within the link and move rigidly with the link.
3. The moments of inertia of the link segments are taken in a frame defined at the center of mass of the link segment and aligned with the precedent joint frame.



Note: These conventions align with those used in URDF files in the `<inertial>` section of `<link>` definitions.

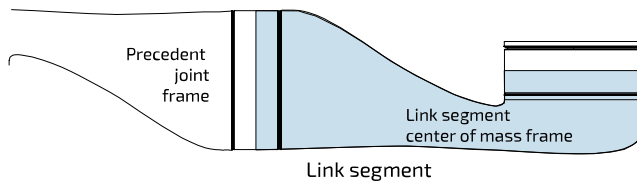


Figure 71: Inertial parameters conventions

Inertial parameters of the robot

Use the inertial parameters of the link segments of the robot to make proper computations before using the robot.

Table 73: Base

Physical quantity	Value	
mass (kg)	1.1461	
center of mass coordinates (m)	[0.002443, 0.000156, 0.086167]	
moments of inertia (kg · m ²)	lxx	0.0033585
	lxy	0.00000039
	lxz	0.00010989
	lyy	0.00331100
	lyz	0.00000191
	lzz	0.00077158

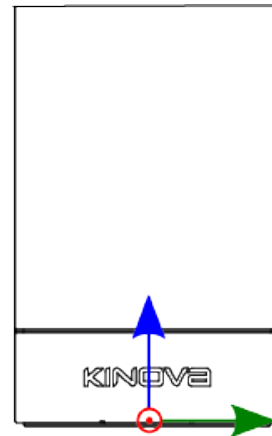
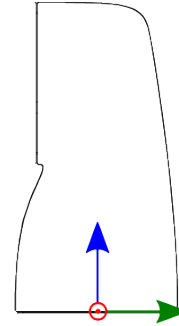


Table 74: Link 1**Physical quantity Value**

mass (kg)	0.9597	
center of mass coordinates (m)	[0.000025, 0.022135, 0.099377]	
moments of inertia (kg · m ²)	lxx	0.0016595
	lxy	0.00000002
	lxz	0.000000036
	lyy	0.0014036
	lyz	0.00034927
	lzz	0.00089493

**Table 75: Link 2****Physical quantity Value**

mass (kg)	1.1776	
center of mass coordinates (m)	[0.029983, 0.21155, 0.045303]	
moments of inertia (kg · m ²)	lxx	0.0114930
	lxy	0.00000010
	lxz	0.000000016
	lyy	0.0010285
	lyz	0.0014077
	lzz	0.0113350



Table 76: Link 3

Physical quantity	Value	
mass (kg)	0.59768	
center of mass coordinates (m)	[0.030156, -0.095022, 0.007356]	
moments of inertia ($\text{kg} \cdot \text{m}^2$)	lxx	0.0016326
	lxy	0.0000071
	lxz	0.0000015
	lyy	0.0002980
	lyz	0.0000960
	lzz	0.0016909

**Table 77: Link 4**

Physical quantity	Value	
mass (kg)	0.52693	
center of mass coordinates (m)	[0.005752, 0.010004, 0.087192]	
moments of inertia ($\text{kg} \cdot \text{m}^2$)	lxx	0.00069098
	lxy	0.00000024
	lxz	0.00016483
	lyy	0.00078519
	lyz	0.00000074
	lzz	0.00034115

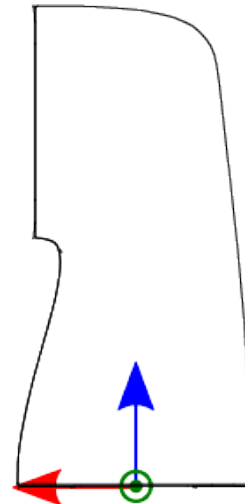
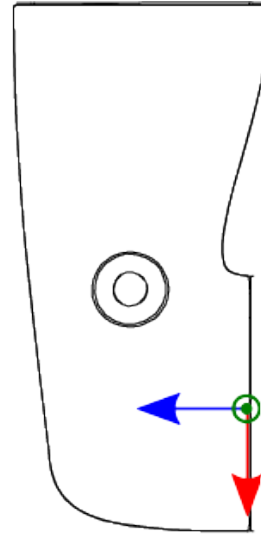
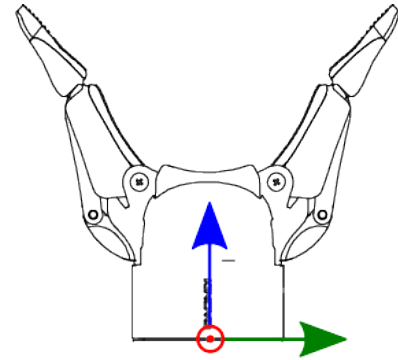


Table 78: Link 5

Physical quantity	Value	
mass (kg)	0.58097	
center of mass coordinates (m)	[0.080565, 0.009804, 0.018728]	
moments of inertia ($\text{kg} \cdot \text{m}^2$)	lxx	0.00021268
	lxy	0.00000521
	lxz	0.00000291
	lyy	0.0010637
	lyz	0.00000011
	lzz	0.0010847

**Table 79: Gripper**

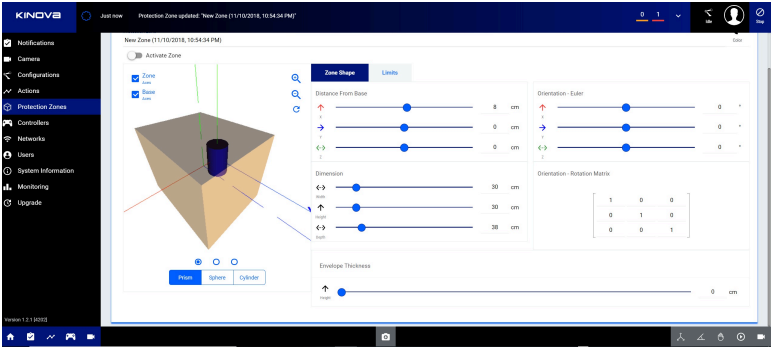
Physical quantity	Value	
mass (kg)	0.2018	
center of mass coordinates (m)	[0.00993, 0.00995, 0.06136]	
moments of inertia ($\text{kg} \cdot \text{m}^2$)	lxx	0.0003428
	lxy	0.00000019
	lxz	0.0000001
	lyy	0.00028915
	lyz	0.00000027
	lzz	0.00013076



Maintenance

All maintenance is preventive and aids in the longevity of the robot. Currently, none of the components of the robot are field replaceable. Contact Kinova for assistance in the case of any component breakdown or malfunction.

Table 80: Preventive maintenance

Action	Description
Setting protection zones	Volumetric protection zones must be established around the robot to protect it from potential damage caused by collisions with known obstacles. Protection zones can be set using Kinova® Kortex™ Web App. 
Updating firmware	Kinova releases updates periodically to robot and robot device firmware to fix known bugs and expand the capabilities of the robot. For best results, it is recommended to regularly update firmware using Web App. 
Updating development packages	Kinova releases updates periodically for the Kinova® Kortex™ ROS packages on the Kinovarobotics/kortex and Kinovarobotics/ros_kortex GitHub repositories. These updates fix known bugs and expand the capabilities of the robot.

Action	Description
Cleaning	Clean the product with a damp cloth and all-purpose household cleaner.
Gripper fingers visual inspection	Check for signs of wear and replace in case of excessive wear or breakage every six months. Never submerge fingers or gripper when connected to the robot. If necessary, remove the fingers with a Phillips T2 screwdriver, and clean with warm water and mild soap.

Troubleshooting

Whenever there are issues with the robot, check the troubleshooting resources available to you and follow general tips.



DANGER: If the robot loses a part, for example a shell due to impact, or if a part breaks, shut down the robot safely and leave it off. Contact Kinova technical support.

Troubleshooting resources

There are several resources that can be used to help diagnose issues when they occur

Resource	Description
Kinova® Kortex™ Web App notifications	
Web App monitoring	The monitoring page provides useful status information on the robot components, including the base , all actuators, and the interface. Notably, currents, voltages, CPU core temperatures and motor temperatures from the sensors are updated in real-time on the monitoring page.
Web App safeties page	When a safety item's warning or error threshold is exceeded, the safety item is highlighted in the Robot Configurations Safety page.
Base controller LED indicators	LEDs on the robot base controller connector panel provide visual feedback on the robot status.
API errors	
GitHub	Information on known issues and workarounds
Release notes	Information on known issues and workarounds

Tips for troubleshooting

For any given issue, there are ways to figure out what is happening.

When the robot enters a fault state, the robot become unresponsive until the fault is cleared. The Xbox gamepad can be used to clear faults, press the left bumper once and proceed.

Consult Web App:

- Check the **Monitoring** page for high-level status information on various components.
- Check the **Notifications** page for any recent [notifications](#).
- Check the **Safeties** page to see if the robot has passed a warning or error threshold. If any safety is triggered, the safety item looks up the information on the safety for guidance on handling.

Remember that the behavior of the robot changes when the robot nears singularities or enters the envelope of [protection zones](#). If robot behavior deviates from what is expected, verify whether one of these two cases applies.

For API-related errors, check the reference tables for guidance on the source of the error and how to deal with it.

Kinova recommends updating robot firmware and Kinova® Kortex™ API packages regularly to keep up with the latest bug fixes and ensure optimal performance.

As part of periodic software updates, Kinova publishes release notes on the Kinova website. The release notes describe known issues, limitations, and workarounds, as well as information about new features and previous bugs fixed in the release.

If all else fails, try rebooting the robot.

If there still are issues, contact Kinova support at support@kinova.ca.

How to respond to safety warnings and errors

The robot has a number of warning and error thresholds set for safety purposes. These are viewable, and in some case configurable, in Kinova® Kortex™ Web App. Respond to safety warnings and error states experienced when operating the robot.

Table 81: Base safeties handling

Safety	Most Probable Cause
Incompatible Firmware version	• Firmware issue
Firmware Update Failure	• Firmware issue • Communication issue
Maximum Ambient Temperature	• CPU heat sink issue • Unknown thermal issue

Safety	Most Probable Cause
Maximum Core Temperature	• CPU heat sink issue • Unknown thermal issue
Joint Fault	• Joint error/warning state
Joint Detection Error	• Communication issue: internal Ethernet or power cable • Base EEPROM issue
Network Initialization Error	• Base CPU board issue • Joint issue
Maximum Current	• Shorted phases on a joint • Payload exceeded
Maximum Voltage	• Power supply issue • Electronic component failure
Minimum Voltage	• Power supply issue • Electronic component failure
Emergency Stop Activated	• Xbox gamepad emergency stop button clicked • Web App emergency stop button clicked
Emergency Line Asserted	• Joint not programmed • Joint in a boot loop • Electrical component failure
Inrush Current Limiter Fault	• Payload exceeded • Electrical component failure

Table 82: Actuators safeties handling

Safety	Most Probable Cause
Following error	• Communication issue • Firmware issue
Maximum velocity	• Communication issue • Firmware issue

Safety	Most Probable Cause
Magnetic position	• Magnet improperly glued
Hall position	• Hall sensor major malfunction
Hall sequence	• Hall sensor major malfunction(s)
Maximum motor current	• Shorted phases • Bad motor • Payload exceeded
Non-volatile memory corrupted	• Incomplete calibration(s) • No system information entered • No torque calibration
Motor driver fault	• Shorted phases • Hall sensor issue
Watchdog triggered	• Firmware issue

Harmonized standards

The Gen3 lite robot is built with specific standards in mind.

Table 83: Applicable Standards

Item	Detail
EMI/EMC	CISPR 11/24 (equivalent to EN 55011/55024)
	FCC class B (CFR 47 part 15)
Flame retardance	UL94-V0 (shell, PCB)
	UL94-V1 (cables)
Laser	21 CFR 1040.10 (except for deviations in Laser Notice No. 50 2007-06-24)
Universal Serial Bus	USB 2.0
Ethernet	IEEE 802.3az-2010
Wi-Fi	IEEE 802.11a/b/g/n/ac
ISO/IEC	ISO 7176 <i>Wheelchairs</i>
	ISO 10218-1 <i>Industrial robots</i>
	ISO 10993-1 : 2009 <i>Biological evaluation of medical devices</i>
	ISO 12100 <i>Safety of machinery</i>
	ISO 13849 <i>Safety-related parts of control systems</i>
	ISO 14539 <i>Vocabulary only</i>
	ISO 13482:2014 <i>Robots and robotic devices</i>
	ISO 13732-1:2006 Ergonomics of the thermal environment ...
	ISO/TS 15066:2016 <i>Collaborative robots</i>
	IEC 60204-1 <i>Safety of machinery – Electrical</i>
	IEC 60601-1 <i>Medical electrical equipment</i>
	IEC 60825-1 <i>Class 1 laser hardware limitation</i>
	IEC 60950-1 <i>IT equipment - Safety</i>
	IEC 61000-6-1/6-3 <i>EMI/EMC</i>

Item	Detail
CE	IEC 61140/A1:2004 <i>Protection against electric shock – Common aspects for installation and equipment</i>
	IEC 62304 <i>Medical Device Software Development</i>
	2002/96/EC <i>WEEE</i>
	2004/108-EC <i>EU Electromagnetic compatibility (EMI/EMC)</i>
	2006/42/EC <i>Machinery directive</i>
	2011/65/EU <i>ROHS</i>
	2014/30/EU <i>EMC</i>
	2014/35/EU <i>LVD</i>
	2014/53/EU <i>Radio Equipment directive</i>
USA	MDR 2017-745 <i>Gap Assessment and CE Transition Strategy</i>
	OSHA 1910.211 <i>Definitions (To be confirmed)</i>
	OSHA 1910.212 <i>General requirements for all machines (To be confirmed)</i>
AU	OSHA 1910.214 <i>Cooperation machinery (To be confirmed)</i>
	AS 2939–1987 <i>Industrial robot systems - Safe design and usage</i>
CA	CAN/CSA-Z434-14 <i>Safety</i>

Contacting Kinova support

Here is where to turn for related support and advice.

For support and advice on hardware related issues, contact us at support@kinova.ca.

For development guidance and software-related questions, check out our GitHub repositories:

- Kinova® Kortex™ GitHub: github.com/Kinovarobotics/Kinova-kortex2_Gen3_G3L/
- Kinova® Kortex™ ROS GitHub: github.com/kinovarobotics/ros_kortex
- Kinova® Kortex™ ROS2 GitHub: github.com/kinovarobotics/ros2_kortex
- Kinova® Kortex™ MATLAB API GitHub: https://github.com/Kinovarobotics/matlab_kortex

Glossary

angular mode

independent joint control; each axis of the manipulator is controlled separately

API

a description of the functions and services of a system that are available to use to operate a system; a subset of an [Interface Control Document \(ICD\)](#)

axis

a fixed line with direction;

used for the measurement of coordinates or angles, depending to whether the robot movement is linear or rotational

base

the bottom-most part of the arm that is placed on a surface for installation; support section of the arm

base frame

the reference frame located at the center of the bottom surface of the arm's base; the origin frame in [Cartesian space](#)

base support

the stable platform to which the base is attached

blocking

the inability for a robotic program to proceed to its next step until the current, blocking, action is finished

Cartesian mode

mode of operation used to control the translational and orientation velocities of a [tool](#) in [Cartesian space](#)

Cartesian space

the Euclidean space described by X, Y and Z axes of the Cartesian [coordinate system](#)

center of mass

unique point of a rigid body where an applied force generates linear acceleration and no angular acceleration

CISPR

Comité International Spécial des Perturbations Radioélectriques

control modes

one of several modalities of controlling the motion of the robot while it is in run mode.

Different modes provide different means to describe or guide the desired motion.

Table 84: Existing control modes

Angular joystick	Cartesian waypoint trajectories
Angular trajectory	Cartesian admittance
Angular waypoint trajectories	Joint admittance
Cartesian joystick	Null space admittance
Cartesian trajectory	Force control

coordinate system

a system used to represent a position in three-dimensional space, consisting of three coordinate axes and an origin

DHCP

protocol used in networks

DoF

the number of variable directions needed to define the configuration of a robot, particularly for all the directions in which the motion can occur

EMC

the capability of electrical equipment to function properly in the electromagnetic environment

EMI

unwanted electromagnetic interference generated by an external source

end effector

device specifically designed for attachment to the flange of the wrist

endpoint

nominal commanded [position](#) that the robot tries to reach with the [Tool Center Point \(TCP\)](#) at the end of a motion path

Euler angle

the rotation in three dimensions of a rigid body in terms of a of three rotations with respect to a [coordinate system](#)

event

an action that happens, and is recognized by the software or firmware

frame

a system used to represent a position in three-dimensional space, consisting of three coordinate axes and an origin

gravity vector

gravitational acceleration with direction and strength; measured in m/s^2 in each Cartesian direction (X, Y, Z)

GUI

a way of interacting with a program of a computer that involves windows, dialogs, icons, and menus

ICD

a package of information that contains all the input and output information of a system without sharing the actual design of the system;

IEEE

Institute of Electrical and Electronics Engineers

IP

Internet Protocol

ISO

International Organization for Standardization

IT

Information Technology

joint

section of the manipulator system that allows one rotational degree of freedom

joint angle

the position of every joint of a robot as series of angles

joint space

the set of all possible joint [positions](#)

LAN**LED**

Light-emitting diode

map

a set of associations between controller device inputs and actions to be triggered by those inputs when the map is active

mapping

a full definition of the possible correspondences between controller device inputs and actions that are triggered by those inputs when the mapping is active; it can consist of multiple [maps](#), for example, to enable multiple different modes on the same controller device

non-blocking

the ability for a robotic program to proceed to its next step even when the current action is not finished

notifications

a log of an [event](#) related to a particular [topic](#) that happens while a user is using the robot

null space

mathematical space of joint speeds where the robot can change its configuration (generate joint speed and motion) without changing the end-effector pose (Null Twist at the end effector)

orientation

the pure rotations that should be applied to the body to move it from a reference placement to its current placement

path

continuous locus of points, or positions in three dimensional space, traversed by the [TCP](#) and described in a specified [coordinate system](#)

path – angular

the set of at least two angular [pose](#), through which the actuator values angles should pass during motion

path – Cartesian

the set of at least two Cartesian [poses](#), through which the [tool](#) of the robot should pass during motion

path planning

computation of a path to reach a goal pose subject to applicable constraints and criteria

payload

the load the robot carries, excluding the tool attached to the flange

payload – max

computation of a path to reach a goal pose subject to applicable constraints and criteria

PC

personal computer

pinch point

an area in which a body part can get in between a stationary part and a moving part of the machinery

pose

the position and orientation of a rigid body in [Cartesian space](#)

position

the definition of the location of an object location in 3D space, usually defined by a 3D [coordinate system](#) using X, Y, and Z coordinates

power off

the act of interacting with the robot such that electricity is removed from the robot

power on

the act of interacting with the robot such that electricity becomes available to the robot

protection zone

an area where the robot cannot enter

RNDIS

Remote Driver Network Interface Specification

ROS

Robot Operating System

RPC

Remote Procedure Call

RTSP

Real Time Streaming Protocol

safeties

hardware limitations that are monitored to increase robot safety

Safety zone**servoing mode**

a modality through which commands are transmitted to robot devices during operation

High-level servoing

The user controls the robot by sending a single command to the base. The base manages the low-level details of executing the command, breaking it down, applying any relevant high-level protections, and routing commands to the desired devices through a 1 kHz communication loop with the devices.

Single level

a single user sends commands to a base

Low-level servoing

The user controls the robot by sending a series of actuator commands to the base through a user-controlled loop. The base routes these commands to the desired device through its own 1 kHz communication loop with the actuators.

singularity avoidance

strategy to avoid configurations where the robot loses its ability to move the end effector in a given direction no matter how it moves its joints

snapshot

a capture of the instantaneous position of the robot

Table 85: Types of snapshots

Cartesian position	Joint angles
Gripper position	Combined robot position and gripper position

SSID

Service Set Identifier

switch off

the act of interacting with the robot such that electricity is removed from the robot

switch on

the act of interacting with the robot such that electricity becomes available to the robot

TCP

Transmission Control Protocol

TCP

the position and orientation of the tool; tool center point

TCP/IP

set of communication protocols used in computer networks and in the Internet

tool

the device attached to the end of a robot, designed to directly interact with the environment

tool configuration

configurations made to enable use of a [tool](#) at the [end effector position](#) of the robot; consists of the mass and [center of mass](#) of the tool, and the [tool transform](#)

tool frame

a [coordinate system](#) attached to the [end effector tool](#)

tool transform

translation and orientation between the interface module reference [frame](#) and the reference frame of the [tool](#) attached to the end of the robot

topic

a set of related robot [events](#) to which the user can subscribe and receive notifications as part of a Publisher-Subscriber (pub-sub) arrangement

Table 86: Examples of topics

User	Configuration change or backup
Controller device input	Factory restore
Safety	Protection zone
Action or sequence	Control, operation, or servoing mode
Connection and disconnection of arm, controller, or tool	



For a full list of events, refer to https://github.com/Kinovarobotics/Kinova-kortex2_Gen3_G3L/blob/master/linked_md/NotificationTopics.md

trajectory

a time-parametrized path in the robot workspace that can be defined by the user

trajectory mode

mode of operation in which the user specifies an endpoint, in [joint space](#) or [Cartesian space](#), that the robot should reach

twist

generalized velocity [vector](#), which is a combination of translational velocity and rotational velocity

UDP

User Datagram Protocol

upgrade package

a file with the extension `.swu` that contains firmware updates for all modules on the robot (base, actuator)

USB

industry computer bus standard that makes it easier to transfer data between devices

user profile

a collection of basic information about the person using the robot, along with username and password credentials for access

Control a robot according to the login credentials, which contains the permissions for reading, updating, and deleting different configuration items.

[Notifications](#) for [events](#) happening during a user's session are associated with the user who is logged in.

View notifications from the **Notifications** page in Web App if Web App is launched and connected to the robot before the notifications are sent.

vector

a mathematical representation of physical quantities that have both magnitude and direction, expressed in a [coordinate system](#)

waypoint

a desired intermediate point to be reached during a robot movement, either a Cartesian pose or a set of [joint angles](#)

Create a list of waypoints to make a trajectory that changes direction without reducing velocities to zero by smoothing changes in direction.

WEEE

Waste of Electrical and Electronic Equipment

TOGETHER IN ROBOTICS



PROUDLY MADE IN
📍 QUEBEC, CANADA

KINOVA

kinovarobotics.com | info@kinovarobotics.com | +1 (855) 654-6682

© Kinova inc 2025. All rights reserved.

¹Intel, the Intel logo, and Intel Core are trademarks of Intel Corporation or its subsidiaries.

EN-UG-017-r4.0-202512