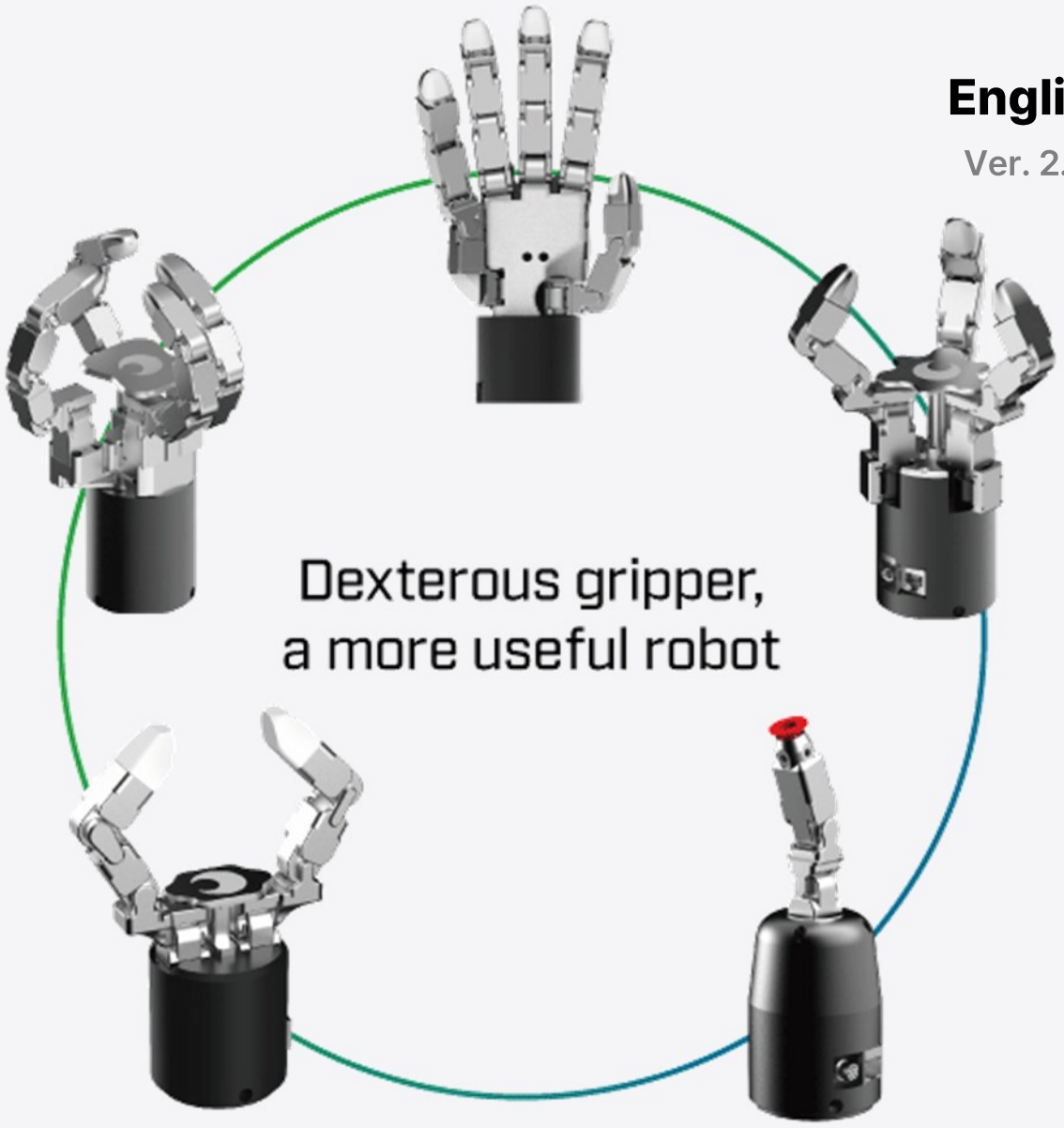
Delto Gripper

Manager and SDK

User Manual



English

Ver. 2.0.0

INFORMATION

Copyright

This user manual was created by Tesollo, and all copyrights belong to Tesollo, and any reproduction based on this manual and distribution to third parties is prohibited.

Notice of revisions to the user guide

This documentation is subject to revision for technical improvements.

Manual Version: 2.0.0 English (2026.04)

© 2026 Tesollo Inc. All rights reserved

Tesollo Corporation

Headquarters: 26F, 165, Convensia-daero, Yeonsu-gu, Incheon, Republic of Korea
Office: 26F, A-Building, 43, Iljik-ro, Gwangmyeong-si, Gyeonggi-do, Republic of Korea
Support T: +82 2 6914 6620
E: support@tesollo.com
H: http://en.tesollo.com/

© 2026 Tesollo Inc. All rights reserved

Table of Contents

Chapter 1. About the documentation **11**

Chapter 2. Precautions **12**

Chapter 3. Getting started **13**

3.1 Install DG Manager 13

3.1.1 Installation file location 13

3.1.2 Install DG Manager 14

3.2 Uninstall DG Manager 18

Chapter 4. Connection Screen **20**

4.1 Login screen 20

4.1.1 Screen layout 20

4.1.2 Operator Mode Connection 22

4.1.3 Developer Mode Connection 26

4.1.4 Connection Check 28

4.2 Main screen 30

4.2.1 Screen layout 30

Chapter 5. Monitoring screen **33**

5.1 Screen layout 33

Chapter 6. Control screen **35**

6.1 Grasp mode 35

6.1.1 Screen layout 35

6.1.2 Grasp control 36

6.1.3 Setting up grasp data 36

6.1.4 Setting the Position Control Mode 38

6.2 Position control 39

6.2.1 Screen layout 39

© 2026 Tesollo Inc. All rights reserved

6.2.2 Position control operations....40
6.2.3 Teaching mode....41
6.3 Gain settings....42
6.3.1 Screen layout....42
6.3.2 Setting the Gain Mode....43
6.3.3 Gain settings....44
6.4 Setting the current....45
6.4.1 Screen layout....45
6.4.2 Setting the current mode....45
6.4.3 Setting the current....46
Chapter 7. Recipe settings screen....47
7.1 Screen layout....47
7.2 Pose recipes....48
7.2.1 Screen layout....48
7.2.2 Setting pose recipes....49
7.2.3 Operation pose recipes....50
7.3 Gain recipes....51
7.3.1 Screen layout....51
7.3.2 Setting gain recipes....52
7.3.3 Operation gain recipes....53
7.4 Grasp recipes....54
7.4.1 Screen layout....54
7.4.2 Setting grasp recipes....55
7.4.3 Operation grasp recipes....56
7.5 Blend recipes....57
7.5.1 Screen layout....57
7.5.2 Setting blend recipes....58

© 2026 Tesollo Inc. All rights reserved

7.5.3 Operation blend recipes 60
Chapter 8. Additional features 61
8.1 Settings 61
8.1.1 Screen layout 61
8.1.2 Boot Recipe 63
8.1.3 Joint Offset 64
8.1.4 Joint Inpose 65
8.1.5 Moving Inpose 66
8.1.6 Low Pass Filter 66
8.1.7 Torque Limit Mode 67
8.1.8 Backup Settings / EEPROM Write 68
8.1.9 Restore Settings 69
8.2 Tools 70
8.2.1 Screen layout 70
8.2.2 Fingertip Sensor 71
8.2.3 GPIO 74
8.2.4 Firmware Download 75
8.2.5 Set IP 77
8.3 Spy 78
8.4 Themes 79
Chapter 9. SDK Structure 80
9.1 Define 80
9.2 Enum 81
9.2.1 enum DG_RESULT 81
9.2.2 enum DG_MODEL 81
9.2.3 enum BLEND_MOTION_STATUS 81
9.2.4 enum COMMUNICATION_MODE 82

© 2026 Tesollo Inc. All rights reserved

9.2.5 enum CONTROL_MODE....82
9.2.6 enum GAIN_MODE....82
9.2.7 enum DEVELOPER_MODE_COMMAND....82
9.2.8 enum DEVELOPER_MODE_RECEIVED_DATA_TYPE....83
9.2.9 enum DG_GRASP_MODE....84
9.2.10 enum DG_SENSOR_TYPE....86
9.3 Struct....87
9.3.1 struct ReceivedGripperData....87
9.3.2 struct RecipeBlendData....88
9.3.3 struct GripperSystemSetting....88
9.3.4 struct GripperSetting....89
9.3.5 struct RecipePoseData....90
9.3.6 struct RecipeGainData....90
9.3.7 struct RecipeGraspData....91
9.3.8 struct ReceivedFingertipSensorData....91
9.3.9 struct ReceivedGPIOData....92
Chapter 10. System functions....93
10.1 System Settings Functions....93
10.1.1 SetGripperSystem....93
10.1.2 SetGripperOption....94
10.1.3 ConnectToGripper....95
10.1.4 DisconnectToGripper....96
10.1.5 SystemStart....96
10.1.6 SystemStop....97
10.1.7 SetIp....97
10.1.8 SetJointEncoderZero....98
10.2 Callbacks....99

© 2026 Tesollo Inc. All rights reserved

10.2.1 CallbackForOnConnected....99
10.2.2 CallbackForOnDisconnected....99
10.2.3 CallbackForOnReceivedGripperData....100
10.2.4 CallbackForOnCommunicationPeriod....101
10.2.5 CallbackForOnReceivedFingertipSensorData....102
10.2.6 CallbackForOnReceivedGPIOData....103
10.3 Other system functions....104
10.3.1 SetLowPassFilterAlpha....104
10.3.2 SetBootMode....104
10.3.3 SetGPIOOutput....105
10.3.4 SetGPIOOutputAll....106
10.3.5 SetTorqueLimitMode....106
10.3.6 SetBootRecipe....107
10.3.7 EEPROMWrite....108
10.3.8 BackupRecipeData....108
10.3.9 RestoreRecipeData....109
10.4 Data Retrieval....110
10.4.1 GetReceivedGripperData....110
10.4.2 GetCommunicationPeriod....111
10.4.3 GetReceivedFingertipSensorData....112
10.4.4 GetReceivedGPIOData....113
10.4.5 GetLibraryVersion....114
Chapter 11. Motion Setting Functions....115
11.1 Grasp Setup Functions....115
11.1.1 SetGraspData....115
11.1.2 SetGraspForce....116
11.2 Setting Gain P....117

© 2026 Tesollo Inc. All rights reserved

11.2.1 SetJointGainP 117
11.2.2 SetJointGainPFinger 118
11.2.3 SetJointGainPAll 119
11.2.4 SetJointGainPBase 119
11.3 Setting Gain D 121
11.3.1 SetJointGainD 121
11.3.2 SetJointGainDFinger 122
11.3.3 SetJointGainDAll 123
11.3.4 SetJointGainDBase 123
11.4 Setting Gain I 125
11.4.1 SetJointGainI 125
11.4.2 SetJointGainIFinger 126
11.4.3 SetJointGainIAll 127
11.4.4 SetJointGainIBase 128
11.5 Setting the Gain PID 129
11.5.1 SetControlPIDMode 129
11.5.2 SetJointGainPID 130
11.5.3 SetJointGainPIDFinger 131
11.5.4 SetJointGainPIDAll 132
11.5.5 SetJointGainPIDAllEqual 133
11.5.6 SetJointGainPIDBase 134
11.6 Setting the motion time 135
11.6.1 SetMotionTimeJoint 135
11.6.2 SetMotionTimeFinger 136
11.6.3 SetMotionTimeAll 137
11.6.4 SetMotionTimeAllEqual 138
11.6.5 SetMotionTimeBase 139

© 2026 Tesollo Inc. All rights reserved

11.7 Setting the Position Control Mode 140
11.7.1 SetPositionModeJoint 140
11.7.2 SetPositionModeFinger 141
11.7.3 SetPositionModeAll 142
11.7.4 SetPositionModeBase 143
11.8 Setting Fingertip Sensor 144
11.8.1 SetFingerTipDataZero 144
11.9 Setting the Current I Gain 145
11.9.1 SetJointCurrentGainI 145
11.9.2 SetJointCurrentGainIFinger 146
11.9.3 SetJointCurrentGainIAll 147
11.9.4 SetJointCurrentGainIBase 148
11.10 Setting the Current P Gain 149
11.10.1 SetJointCurrentGainP 149
11.10.2 SetJointCurrentGainPFinger 150
11.10.3 SetJointCurrentGainPAll 151
11.10.4 SetJointCurrentGainPBase 152
Chapter 12. Recipe Functions 153
12.1 Recipe Pose Functions 153
12.1.1 UpdateRecipeCurrentPoseData 153
12.1.2 UpdateRecipeTargetPoseData 154
12.1.3 LoadRecipePoseData 155
12.2 Recipe Gain Functions 156
12.2.1 UpdateRecipeGainData 156
12.2.2 LoadRecipeGainData 157
12.3 Recipe Grasp Functions 158
12.3.1 UpdateRecipeGraspData 158

© 2026 Tesollo Inc. All rights reserved

12.3.2 LoadRecipeGraspData 159
12.4 Recipe Blend Functions 160
12.4.1 UpdateBlendJoint 160
12.4.2 ClearMoveBlendJoint 161
12.4.3 AddMoveBlendJoint 162
12.4.4 SetMoveBlendJoint 163
12.4.5 StartMoveBlendJoint 164
12.4.6 StopMoveBlendJoint 165
Chapter 13. Motion Functions 166
13.1 ManualTeachMode 166
13.2 StartGraspMotion 167
13.3 Position control 168
13.3.1 MoveJoint 168
13.3.2 MoveJointFinger 169
13.3.3 MoveJointAll 170
13.3.4 MoveServoJoint 171
13.3.5 MoveJointBase 172
13.4 Current Control 173
13.4.1 SetCurrentControlMode 173
13.4.2 SetTargetCurrentJoint 174
13.4.3 SetTargetCurrentFinger 175
13.4.4 SetTargetCurrentAll 176
13.4.5 SetTargetCurrentBase 177
Appendix 1. Glossary and Description 178
Appendix 2. Error list 182

© 2026 Tesollo Inc. All rights reserved

Chapter 1. About the documentation

- Words, abbreviations, and other terms that are used on their own, such as 'Developer Mode, J1', are listed in 'Appendix 1. Glossary and Description' in case of confusion

- DG Manager and the SDK have been tested in a Windows 10 x64, Window 11 x64, Ubuntu 24.04 x64, Ubuntu 22.04 x64, C++20 environment and are also compatible with similar OS environments. Additionally, they have been verified to run on the following PC specifications
  * CPU: AMD Ryzen 5 5600 6-Core Processor
  * RAM: 16GB
  * Graphic: NVIDIA GeForce RTX 3060

- The Operator Mode communication protocols are 'Modbus RTU' and 'Modbus TCP'. Developer Mode uses its own protocol based on TCP/IP

© 2026 Tesollo Inc. All rights reserved

Chapter 2. Precautions

- Depending on your PC performance and network environment, the program may not work smoothly.

- DG Manager can be used to operate the Delto Gripper series provided by Tesollo. However, supported features may vary depending on the firmware version and gripper model, and some features may not be compatible. Therefore, please ensure that you consider the firmware version and gripper model when using DG Manager and the SDK.

- Failure to follow the instructions in this manual and incorrect operation may result in malfunctions. This may cause damage to the gripper and in some cases may not be covered by after-sales service.

- If you need any other support, please contact Tesollo Support (support@tesollo.com)

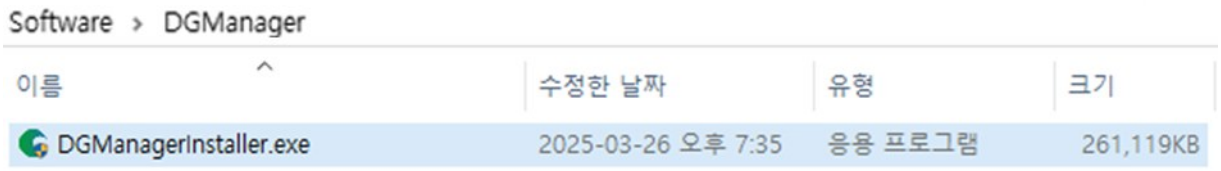
© 2026 Tesollo Inc. All rights reserved

Chapter 3. Getting started

3.1 Install DG Manager

3.1.1 Installation file location

3.1.1.1 Windows



Inside the USB drive included in the gripper packaging, navigate to the folder path shown in the illustration: “Software - DGManager.” There, you will find an installation file named “DGManagerInstaller_ver_x_x_x.exe” (where ver_x_x_x indicates the version of DG Manager). Run this installation file to install DG Manager.

3.1.1.2 Ubuntu 24.04 / 22.04

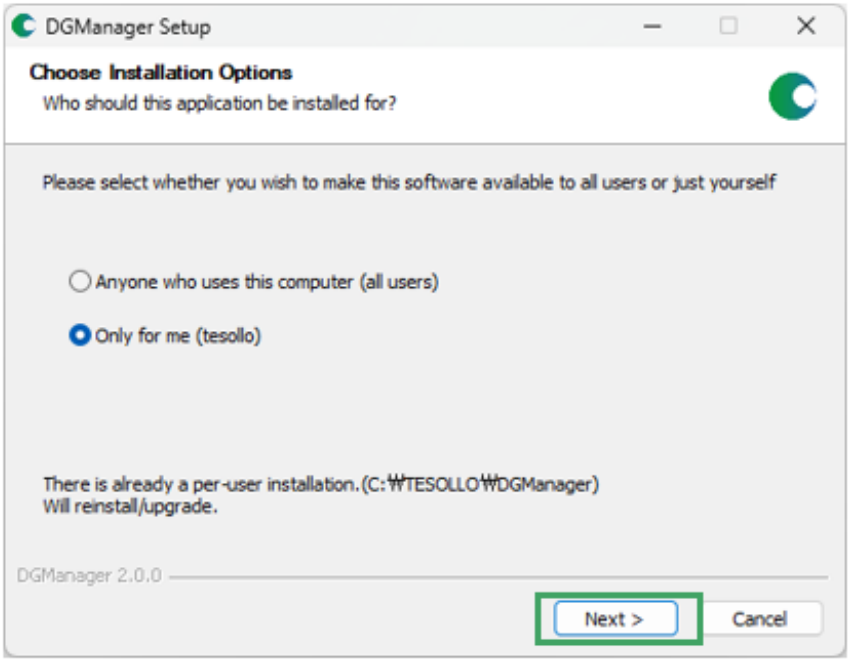


Inside the USB drive included in the gripper packaging, navigate to the folder path shown in the illustration: “Software – DGManager-Linux.” There, you will find an installation file named “DGManager_linux_ver_x_x_x.zip” (where ver_x_x_x indicates the version of DG Manager). You can run DGManager after extracting the archive.

© 2026 Tesollo Inc. All rights reserved

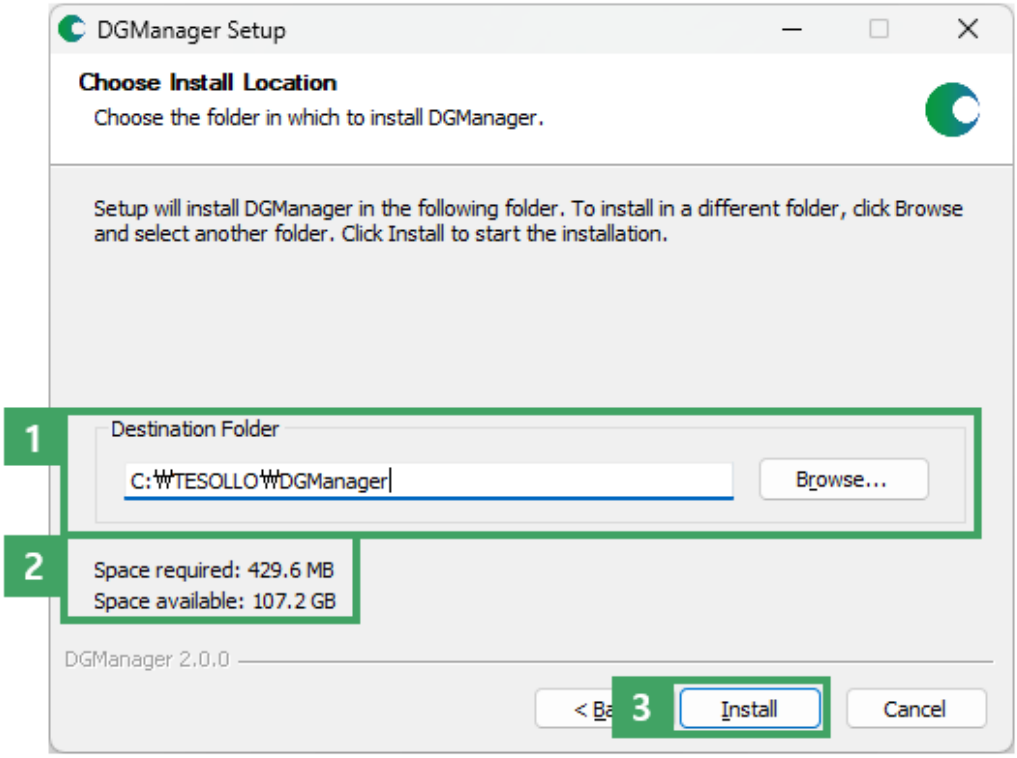
3.1.2 Install DG Manager

3.1.2.1 Windows



When you run the program, the installation file screen is displayed as shown above. Click 'Next>' to go to the next screen.

© 2026 Tesollo Inc. All rights reserved

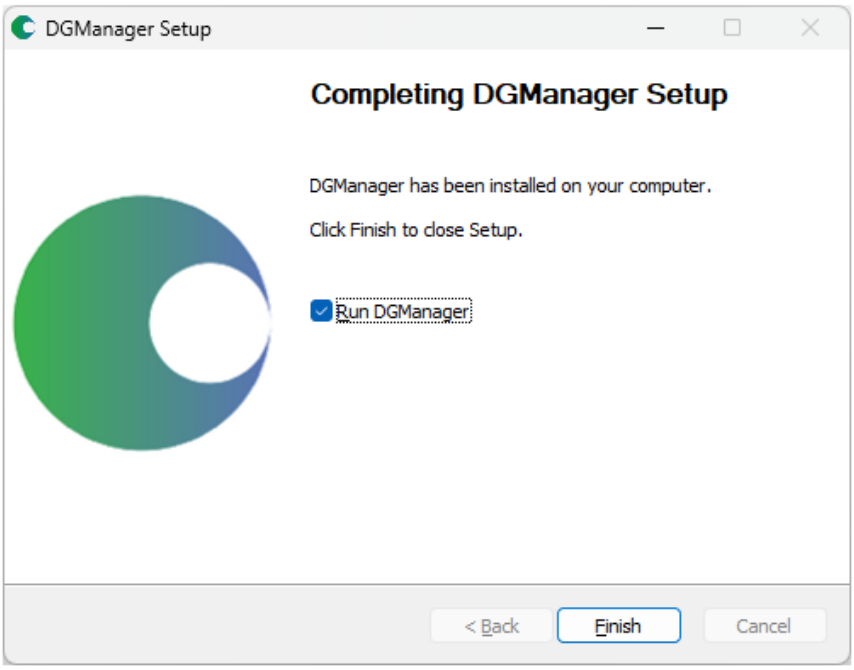


After setting the file path, start the installation. Make sure you have enough free space on your storage before starting the installation.

Item

Number	Name	Description
1	Destination Folder Browse	Set the path to the folder where you want to install the files
2	Check capacity	Displaying file size and install space capacity
3	Install	Start installing files

© 2026 Tesollo Inc. All rights reserved



When the file installation is successfully completed, a screen like the one above will appear, and a shortcut to DGManager.exe will be created on the desktop.



© 2026 Tesollo Inc. All rights reserved

3.1.2.2 Ubuntu 24.04 / 22.04

Example

```
bash
        # Extract the compressed file
        unzip DGManager_linux_ver_1_2_0.zip
        # Enter the extracted directory
        cd DGManager-linux-x64/
        # Install dependencies, set execution permissions, and run
        . install_run.sh
        # Grant execution permissions and run (without installing libraries)
        . run.sh
        # Execute the program
        ./DGManager
```

After extracting the archive, you can run DGManager.

If you need to install dependency packages and set execution permissions, please run the install_run.sh script first.
(This process requires an internet connection.)

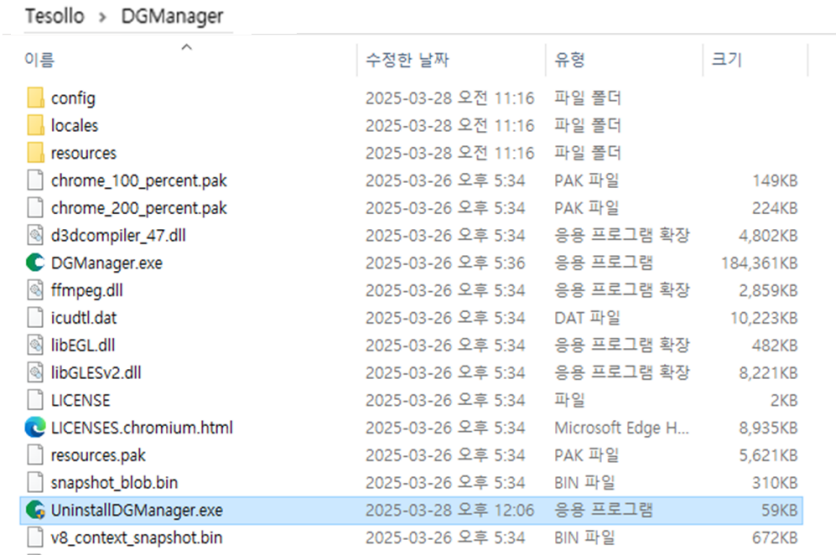
If you only need to grant execution permission and run the program without installing libraries, run the run.sh script.

To simply execute the program, run the ./DGManager file directly.

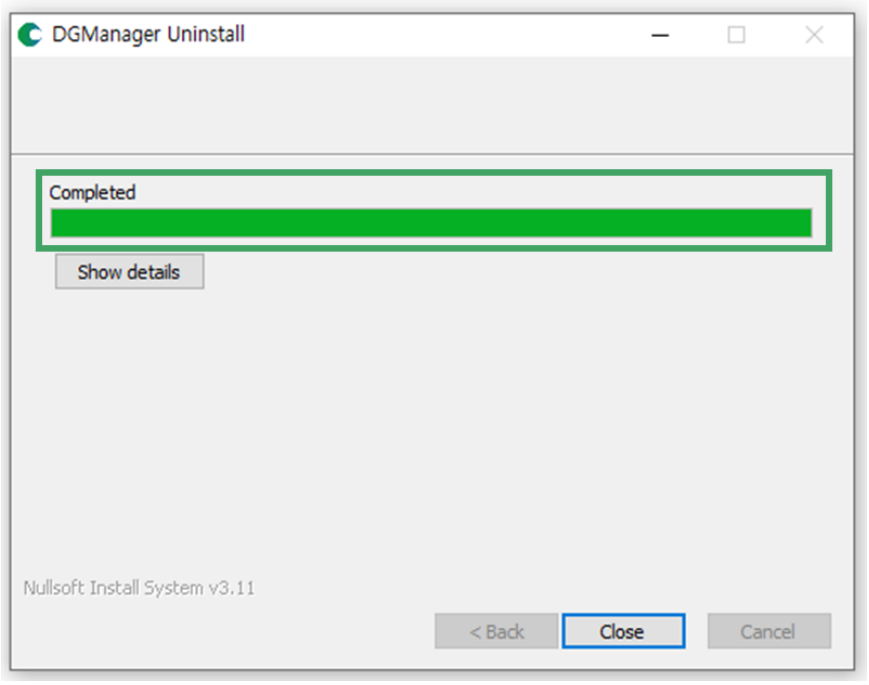
© 2026 Tesollo Inc. All rights reserved

3.2 Uninstall DG Manager

3.2.1.1 Windows



Enter the file installation path and you will find a file named 'UninstallDGManager.exe'. Run that file to start uninstalling the DG Manager program.



When the file removal is complete, you'll see a screen like the one shown above.

© 2026 Tesollo Inc. All rights reserved

3.2.1.2 Ubuntu 24.04 / 22.04

Since the files are shared as a ZIP archive, you can simply delete the extracted directory to remove all files on Linux.

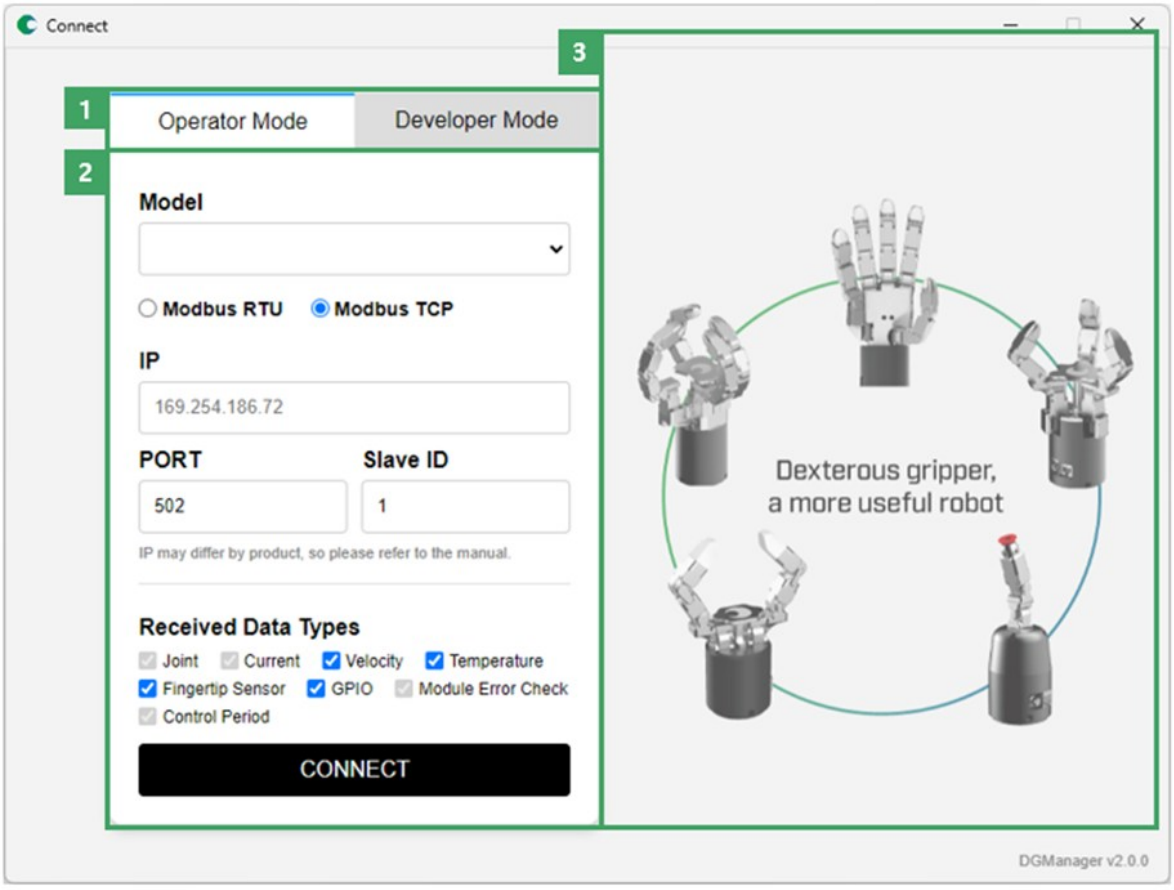
© 2026 Tesollo Inc. All rights reserved

Chapter 4. Connection Screen

4.1 Login screen

4.1.1 Screen layout

Description



Delto Gripper supports **Operator Mode** and **Developer Mode.**

Operator Mode

This mode allows control using the internal controller of the product. It supports the Modbus Protocol, enabling easy control of the gripper either by simply modifying the register map or using the provided UI.

If you are using Operator Mode and the SDK, you must set the “Torque limit mode” to “off” in order to use “stall torque.”

© 2026 Tesollo Inc. All rights reserved

Developer Mode

This mode uses a user-developed controller that receives information from each joint of the gripper and allows you to control each joint directly.

*** Caution**

Using 'Torque Limit Mode' off in Operator Mode or using stall torque in Developer Mode may cause a rapid increase in product temperature and potential product damage. Please note that repairs (A/S) may be difficult in case of failure.

Item

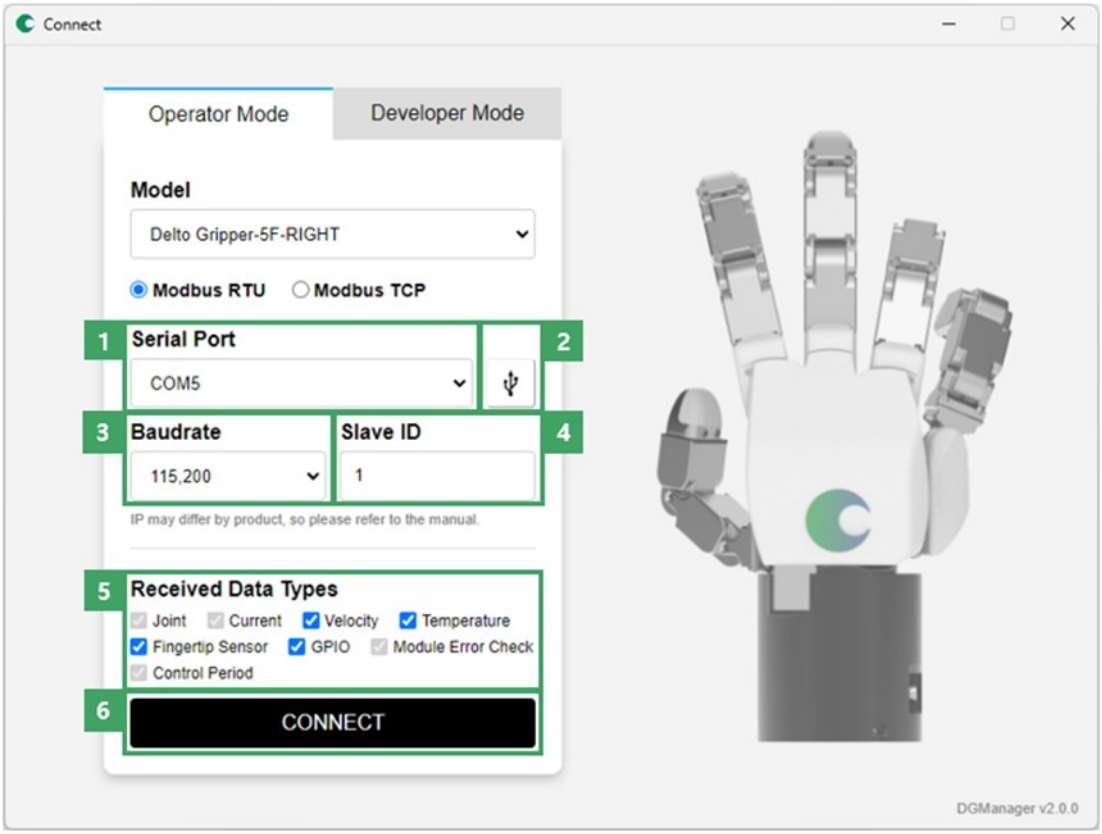
Number	Name	Description
1	Select a control mode	Set the control mode to choose to between 4.1.2 Operator Mode Connection and 4.1.3 Developer Mode Connection
2	Enter connection settings	Required settings based on your control and communication method
3	Gripper Model Image	Show Selection Gripper Model Image

© 2026 Tesollo Inc. All rights reserved

4.1.2 Operator Mode Connection

4.1.2.1 Set up a Modbus RTU Connection

Description



Modbus RTU Connection Example

If the gripper model you are connecting is DG-3F-M using Modbus RTU via the COM8 port, with a baud rate of 115200 and Slave ID 1, first select “Delto Gripper-3F-M” under “Model.” Then select “Modbus RTU,” click the Scan button, choose “COM8” under “Serial Port,” select “115,200” under “Baudrate,” and enter 1 for “Slave ID.” Finally, click the CONNECT button. If the communication is successfully established, the system will proceed to the main screen.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Serial Port	Select the serial port connected to the gripper
2	Scan	Scan the currently connected serial port
3	Baudrate	Select the gripper's set Baudrate ▢ 115200
4	Slave ID	Enter Modbus Slave ID ▢ Slave ID range: 1 - 255
5	Received Data Types	Select the data type to be received
6	CONNECT	Connect gripper communication with a set value

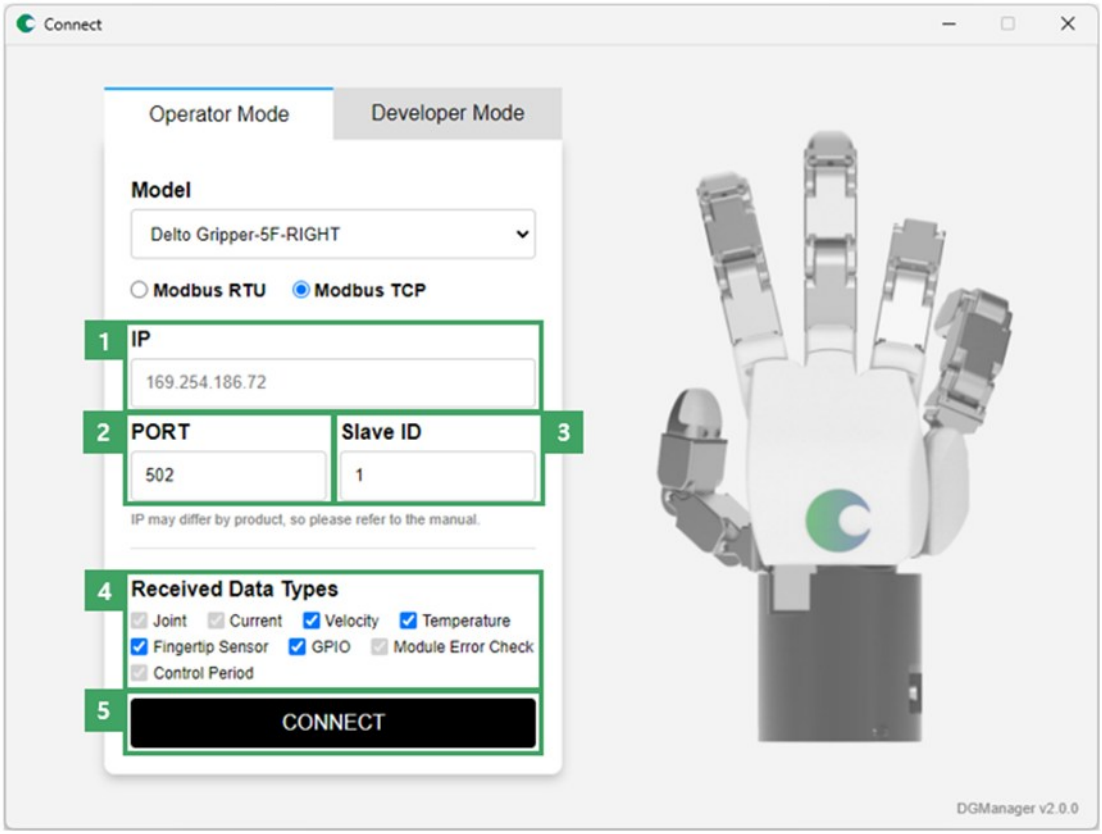
Note

10.1.1 SetGripperSystem
10.1.3 ConnectToGripper
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

4.1.2.2 Set up a Modbus TCP Connection

Description



Example Modbus TCP Connection

If the gripper model you are connecting is DG-3F-M, with the IP address set to 169.254.186.72, port 502, and Slave ID 1, first select “Delto Gripper-3F-M” under “Model.” Then select “Modbus TCP” and enter the corresponding values in the fields below. Enter “169.254.186.72” for “IP,” “502” for “PORT,” and “1” for “Slave ID.”

Finally, click the “CONNECT” button. If the communication is successfully established, the system will proceed to the main screen.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	IP	IP Input for Gripper Communication Connection ▢ Gripper default IP: '169.254.186.72' ▢ DG-5F-L default IP: '169.254.186.73'
2	PORT	Enter a port for the gripper communication connection ▢ Port range: 1 - 65535 ▢ Gripper default port: 502
3	Slave ID	Enter Modbus Slave ID ▢ Slave ID range: 1 - 255
4	Received Data Types	Select the data type to be received
5	CONNECT	Connect gripper communication with a set value

Note

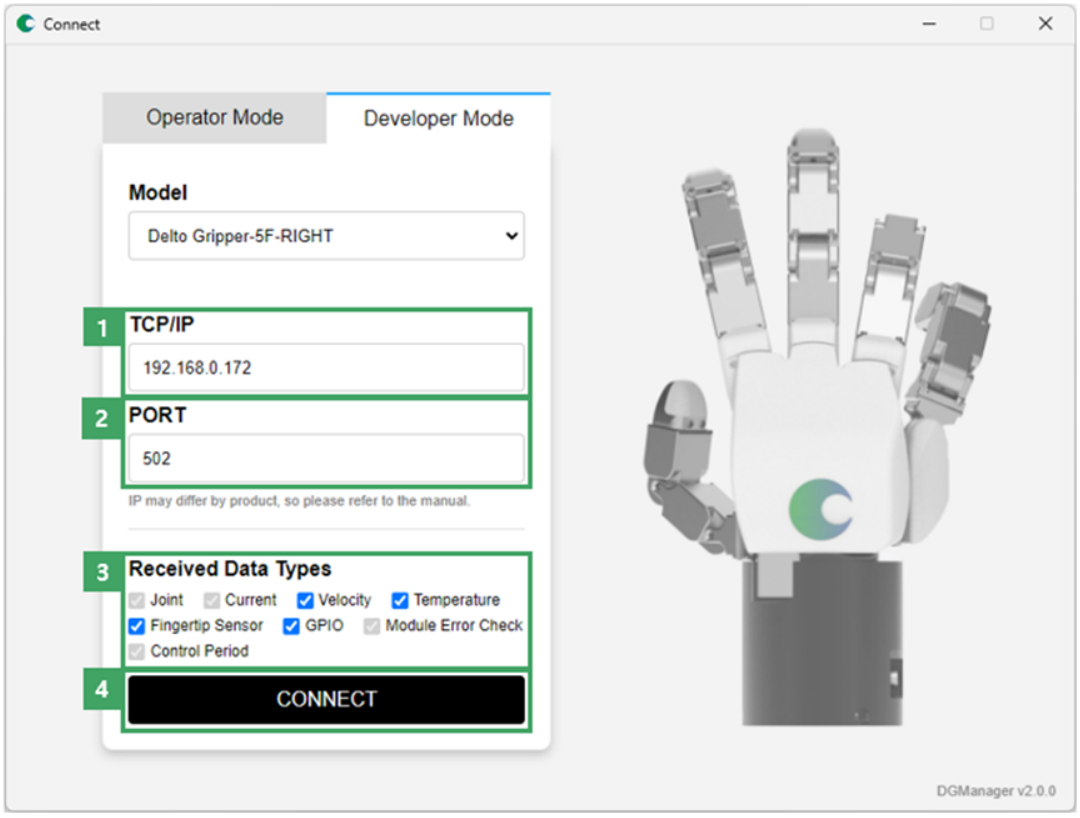
10.1.1 SetGripperSystem
10.1.3 ConnectToGripper
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

4.1.3 Developer Mode Connection

4.1.3.1 Developer Mode TCP/IP Connection

Description



Example Developer Mode TCP/IP Connection

If the model is Delto Gripper-3F-M, IP is 169.254.186.72, and port is 502, first select 'Delto Gripper-3F-M' under 'Model'. Then enter '169.254.186.72' in the TCP/IP field and 502 in the Port field. Then enter 1 in the Slave ID field, and finally click the CONNECT button to complete the connection.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	TCP/IP	IP Input for gripper communication connection ▢ Gripper default IP: '169.254.186.72' ▢ DG-5F-L default IP: '169.254.186.73'
2	PORT	Enter a port for the gripper communication connection ▢ Port range: 1~ 65535 ▢ Gripper default port: 502
3	Received Data Types	Select the data type to be received
4	CONNECT	Connect gripper communication with a set value

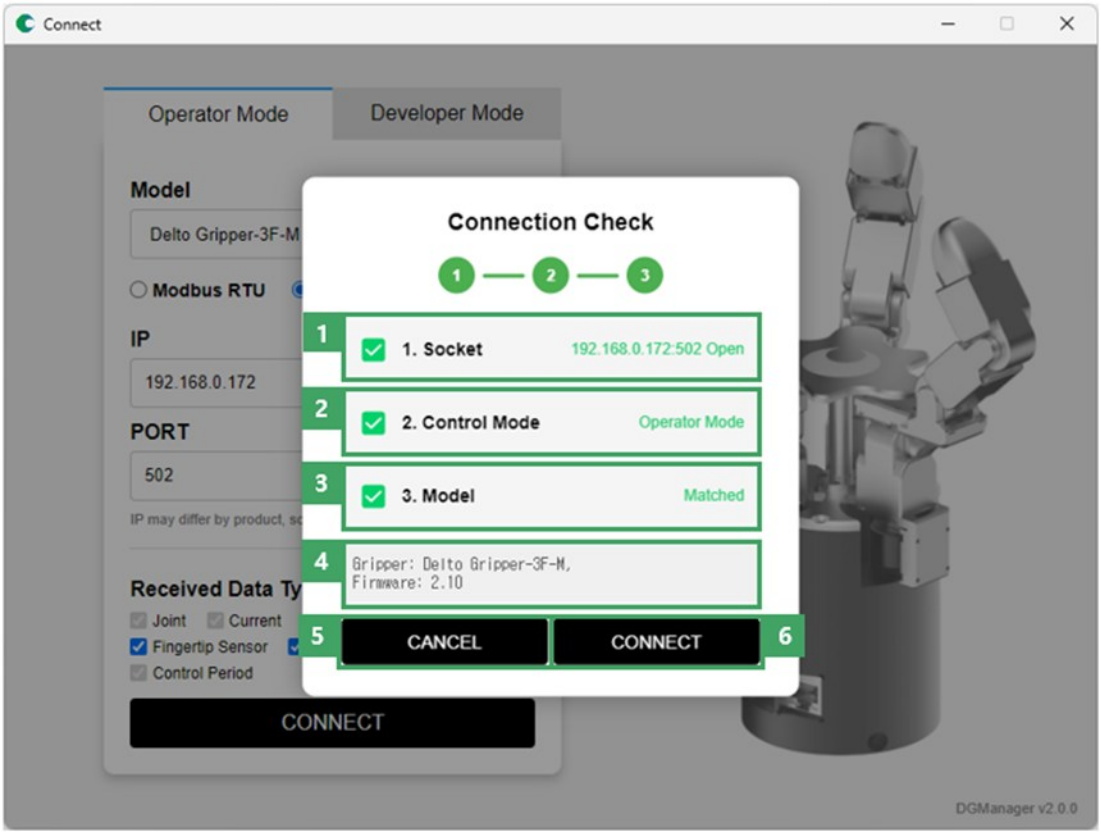
Note

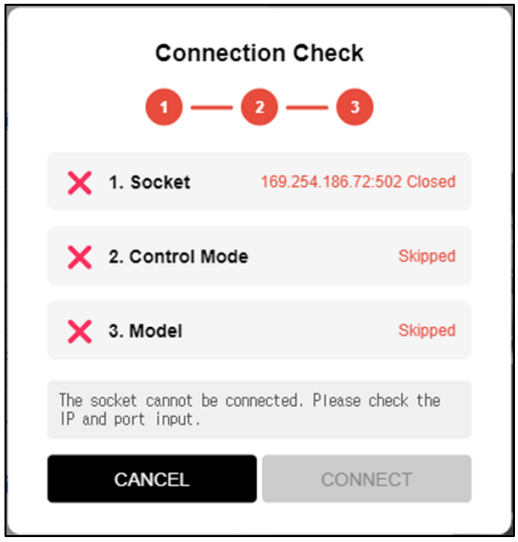
10.1.1 SetGripperSystem
10.1.3 ConnectToGripper
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

4.1.4 Connection Check

Description







Connection Check is the step to ensure proper communication with the Delto Gripper by checking the current socket status, control mode, and model information.

© 2026 Tesollo Inc. All rights reserved

Item

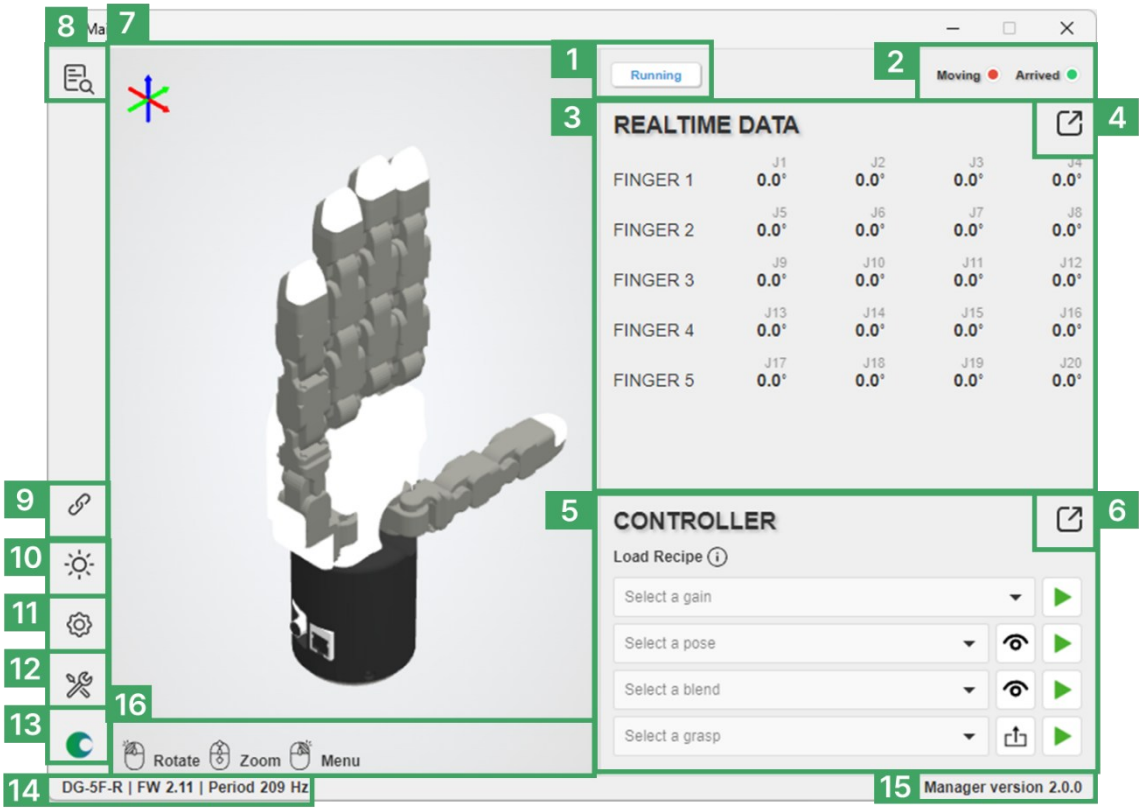
Number	Name	Description
1	Socket	Verify that the IP range between the current device and the gripper is correct
2	Control Mode	Compare the gripper control mode with the control mode selected in DG Manager
3	Model	Compare the model selected in DGManager with the robot model to be connected.
4	Log	Display logs for each item
5	CANCEL	Close the current window
6	CONNECT	Attempt to connect to the gripper

© 2026 Tesollo Inc. All rights reserved

4.2 Main screen

4.2.1 Screen layout

Description



The main screen allows you to access the full range of functions for the gripper.

You can turn gripper control on or off, and monitor real-time joint angle information through a 3D simulation. Additionally, you can preview and execute motions using recipes, and access various advanced settings such as theme switching, disconnecting, and viewing information. The gripper can be easily controlled through the “Start/Stop Gripper System” section on the main screen. When in the “Ready” state, selecting “Start” activates the system and prepares it to receive commands. When in the “Running” state, selecting “Stop” halts the gripper’s operation and ends control.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Starting/Stopping the Gripper System	▢ Ready: Gripper waiting state ▢ Running: Gripper system activation status ▢ Start: Perform gripper system activation ▢ Stop: Perform a gripper system stop
2	Show current status	▢ Moving: Color green if the current gripper is moving, red if it is stationary ▢ Arrived: Color green or red when the target value reaches within the set range
3	REAL-TIME	Shows the real-time position of each joint
4	Monitoring screen	Open Chapter 5 Monitoring screen
5	CONTROLLER	View saved recipes, and run the selected recipe See below for detailed Motional descriptions
6	Control screen	Open Chapter 6 Control screen and Chapter 7 Recipe settings screen
7	Simulation Screen	Get the current gripper's position and render in real time Control joints by dragging them with the mouse
8	Spy Screen	See 8.3 Spy
9	Disconnecting Gripper Communication	Disconnecting Gripper Communication
10	Changing themes	Change to one of the light / dark themes
11	Gripper Settings	Gripper 8.1 Settings Open the screen
12	Gripper Tools	Gripper 8.2 Tools Open the screen
13	About the program	Open the program information window
14	Gripper information	Model / Firmware Version / Communication Period
15	DGManager version	Display the current DGManager program version
16	Simulation control button	Control the simulation screen and assign colors to joints

© 2026 Tesollo Inc. All rights reserved

Note

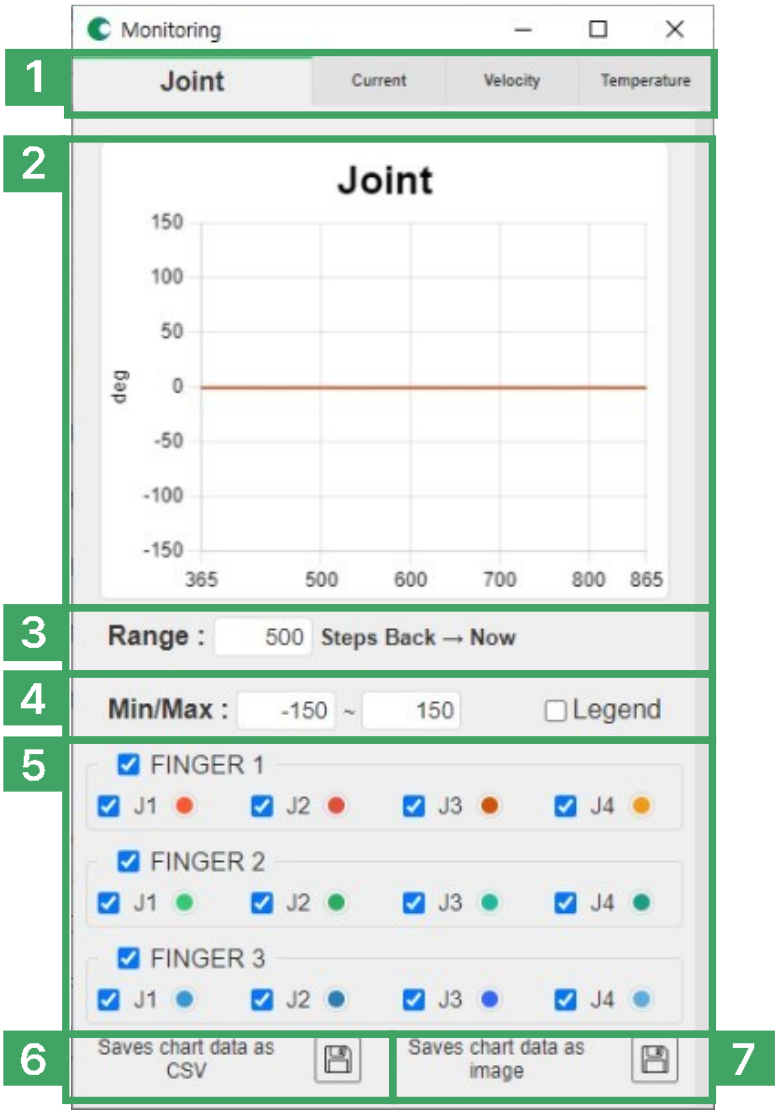
Chapter 5 Monitoring screen
Chapter 6 Control screen
Chapter 7 Recipe settings screen
Chapter 8 Additional features
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

Chapter 5. Monitoring screen

5.1 Screen layout

Description



The Monitoring screen provides a real-time display of selected graph items and the ability to save data. You can select joints, current, velocity, and temperature to view in a real-time graph, set the horizontal range (number of data to save) and vertical range of the graph, select the joints to display on the graph, and save the current graph data as a CSV file or image PNG file.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Select a graph item	Select the data to display ☐ Joint ☐ Current ☐ Velocity ☐ Temperature
2	Real-time graph screen	Graph the data you receive in real time
3	Range	The number of graph data that is a horizontal range
4	Min/Max	Set the Min/Max value for the graph's vertical range
5	Selecting and displaying joints	The joints to display on the graph
6	Saves chart data as CSV	Save the current graph data to a CSV file
7	Saves chart data as image	Save the current image of the graph as a PNG file

Note

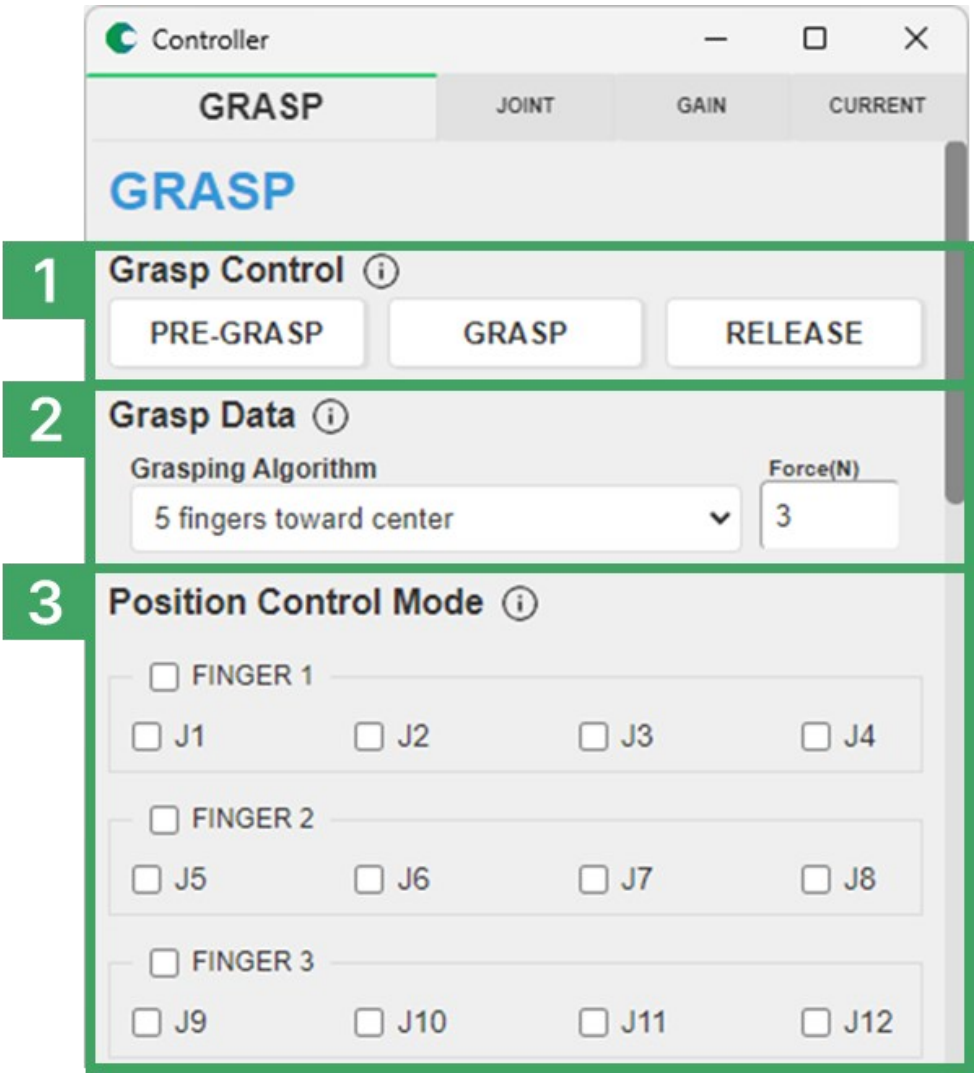
Velocity and Temperature data are not provided on DG-3F-B

© 2026 Tesollo Inc. All rights reserved

Chapter 6. Control screen

6.1 Grasp mode

6.1.1 Screen layout



Number	Name	Description
1	Grasp Control	Performing GRASP and RELEASE
2	Grasp Data	Setting grasp options and grasp strength
3	Position Control During Grasping	Set each joint to position control mode

© 2026 Tesollo Inc. All rights reserved

6.1.2 Grasp control

Description



The ability to manipulate grasps and releases.

Item

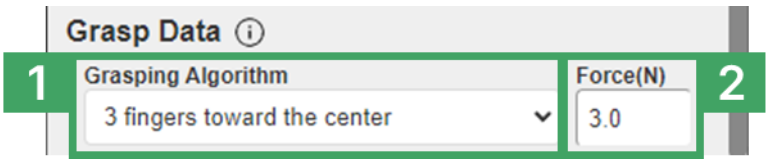
Number	Name	Description
1	PRE-GRASP	Load an example pose for testing the "GRASP" function
2	GRASP	Are performed with the values set in 6.1.3 Setting up grasp data
3	RELEASE	Release is 6.2.2 Position control operations Perform a position move with the last target joint position value

Note

* See 'Appendix 2. Error list' for errors

6.1.3 Setting up grasp data

Description

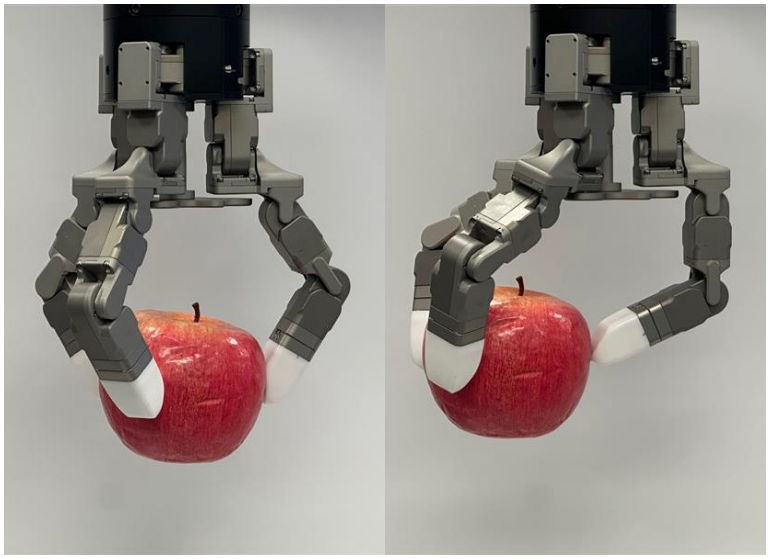


This feature allows you to configure the gripper's grasping data. This feature is not available on the DG-1F-M model. Each model provides its own Grasping Algorithm. The PRE-GRASP pose varies for each Grasping Algorithm. The joint position data for the PRE-GRASP pose of each grasping algorithm can be found in '6.2.2 Position control operations' or '8.3 Spy'.

By default, the real-time force control grasping algorithm is enabled. This algorithm allows each fingertip involved in the grasp to follow the center between the fingers.

© 2026 Tesollo Inc. All rights reserved

Because it operates in real time, even if an external force causes the joint positions to change, the gripper will attempt to maintain the grasping posture based on the new joint positions. In particular, this performance can be easily verified when the position control mode described in '6.1.4 Setting the Position Control Mode' is disabled for all joints.



[Maintain grasp posture at current position even if gripper joint position changes]

Item

Number	Name	Description
1	Grasping Algorithm	See 9.2.9 enum DG_GRASP_MODE
2	Force	Set grasp strength (Unit: N)

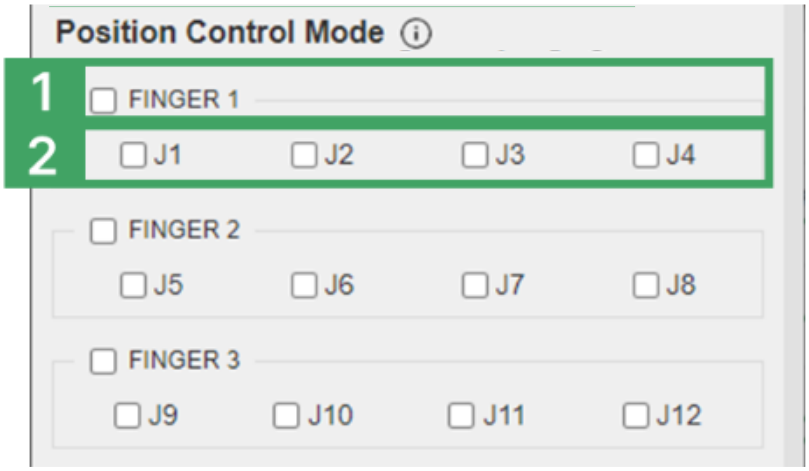
Note

11.1 Grasp Setup Functions
* Not available series: DG-1F-M
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.1.4 Setting the Position Control Mode

Description



Joints set to Position Control mode will perform position control instead of grasping when performing a grasp. This is used when you do not want to move a particular joint during a grasp.

If you check J1 and J2 and then perform a grasp on ' 6.1.2 Grasp control ', then joint 1 and joint 2 will be excluded from the grasp algorithm and will only perform a translate to the target position value.

Item

Number	Name	Description
1	FINGER 1 – FINGER n	Selecting a button selects all joints of the finger
2	J1 - Jn	Selecting a button selects the corresponding joint

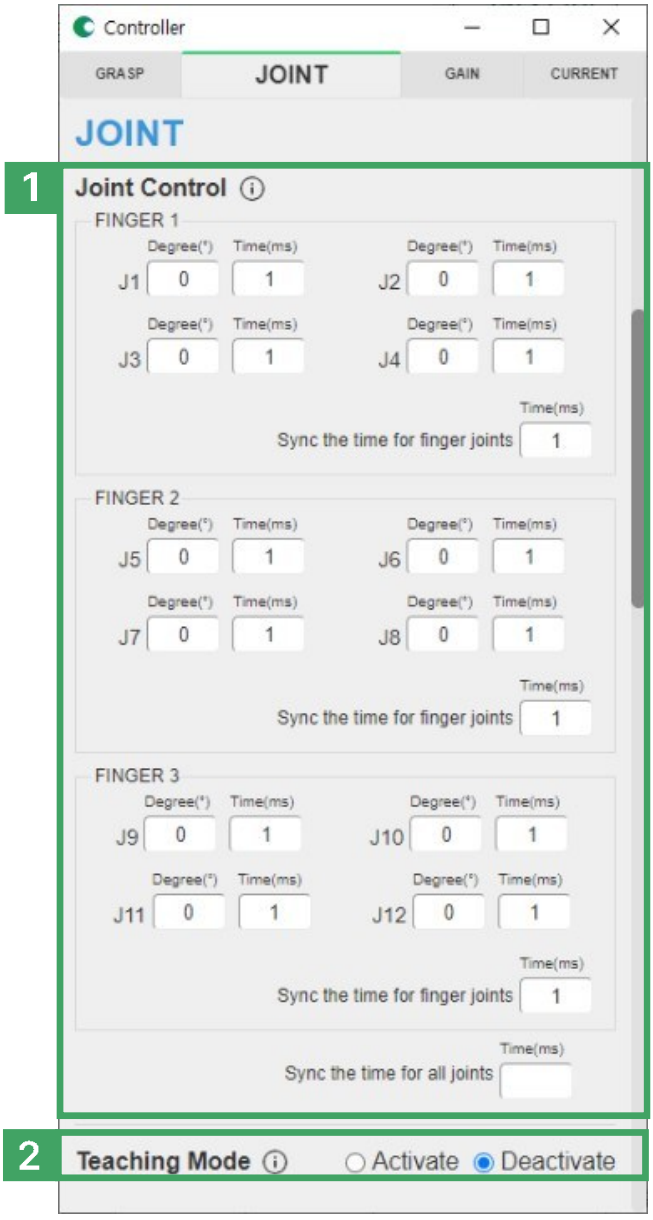
Note

11.7 Setting the Position Control Mode
* Not available series: DG-1F-M
* The base joint of the DG-4F-M is controlled by joints 17 and 18.
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.2 Position control

6.2.1 Screen layout

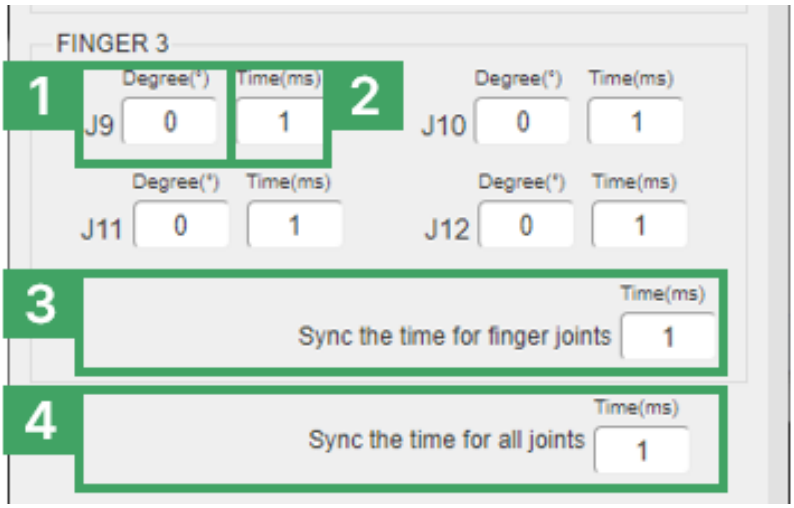


Number	Name	Description
1	Joint Control	Enter a target value to control the position of the joint
2	Teaching Mode	Set up teaching mode for direct instruction

© 2026 Tesollo Inc. All rights reserved

6.2.2 Position control operations

Description



Position control allows you to reposition the gripper joints. To perform a position move, you need to set a target value. The unit of the target value is 'degree'. You can also set the motion time to reach the target position. The unit of motion time is 'ms'.

If the Time of J1 is entered as 1000ms and the degree value is entered as 60°, the first joint corresponding to J1 will move from the current position to 60° in 1000ms. The 'Time' you set here may not be an exact number depending on the performance of the controller when used in Developer Mode and may not be an exact number depending on the current control value for the joint.

Item

Number	Name	Description
1	J1-Jn Degree	Enter a target position value (Unit: degrees)
2	J1-Jn Time	Enter a motion time to reach the target position (Unit: ms)
3	Sync the time for finger joints	Enter a motion time for all joints of the selected finger (Unit: ms)
4	Sync the time for all joints	Enter a motion time for all joints (Unit: ms)

Note

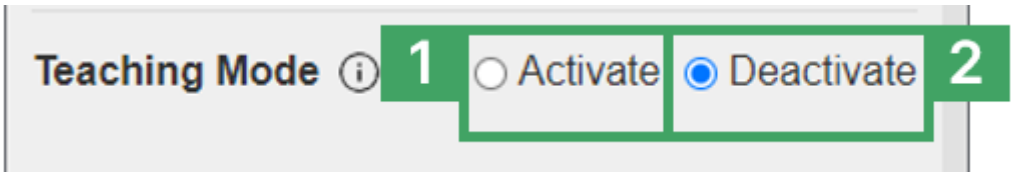
* Number of joints and fingers varies by model

© 2026 Tesollo Inc. All rights reserved

11.6 Setting the motion time
11.7 Setting the Position Control Mode
* See 'Appendix 2. Error list' for errors

6.2.3 Teaching mode

Description



Teach mode is a feature that allows you to teach the gripper directly. The operator can teach the gripper by entering target values for the gripper joints, or the operator can utilize Teach Mode to perform teaching through direct instruction. When the 'Activate' button is selected to use Teach Mode, the gripper will enter the Teach Mode state and the user can physically manipulate the gripper to move it to the desired position. This is useful for creating recipe poses that save the current position.

Item

Number	Name	Description
1	Activate	Enable teaching mode
2	Deactivate	Disable teaching mode

Note

13.1 ManualTeachMode
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.3 Gain settings

6.3.1 Screen layout



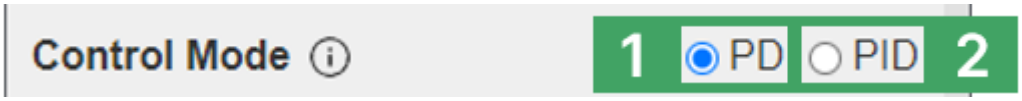
Gain setting is a tuning process used to optimize the joint movement of the gripper for specific purposes. Depending on the tuning method, you can improve the backdrivability of the gripper or enhance its positioning accuracy.

© 2026 Tesollo Inc. All rights reserved

Number	Name	Description
1	Control Mode	Set PD, PID Gain Mode
2	Set Gain	Set the PID Gain Value for Each Joint

6.3.2 Setting the Gain Mode

Description



The Gain Mode setting allows you to set the PD, PID control mode. The Delto Gripper Series operates in PD control by default. This is because it allows the object and gripper to react flexibly to physical damage in collision situations. On the other hand, for higher positioning accuracy, we recommend using PID control mode.

Item

Number	Name	Description
1	PD	Set to PD control
2	PID	Set to PID control

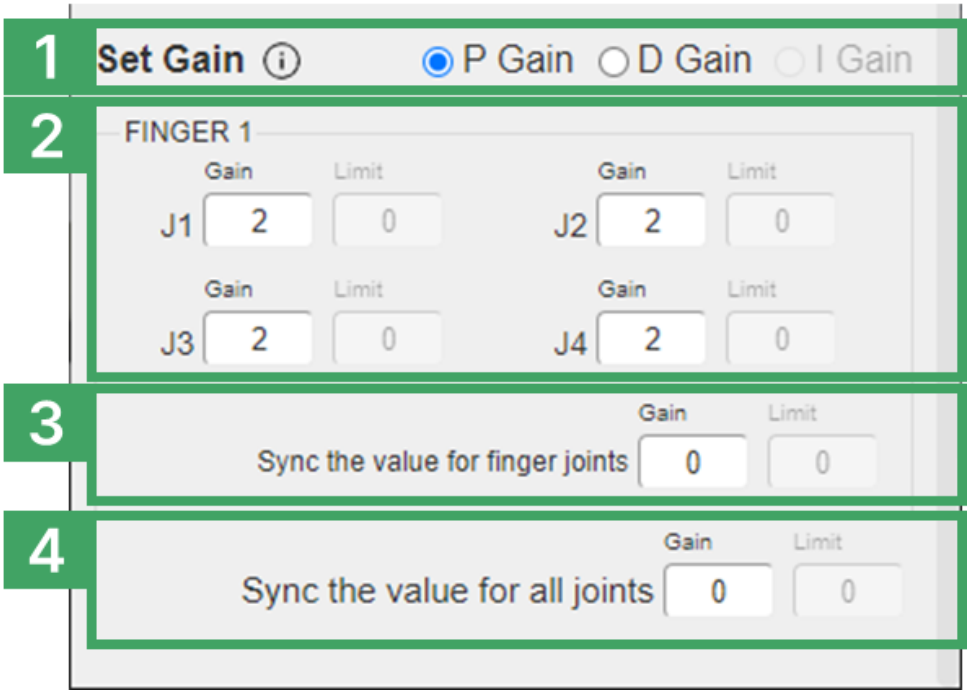
Note

11.5.1 SetControlPIDMode
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.3.3 Gain settings

Description



Set the joint, finger, and overall PID gains.

Item

Number	Name	Description
1	Set Gain	Select a gain to set
2	J1-Jn Gain	Set a gain value for that joint
3	Sync the value for finger joints	Set a gain value for all joints of the selected finger
4	Sync the value for all joints	Set a gain value for all joints

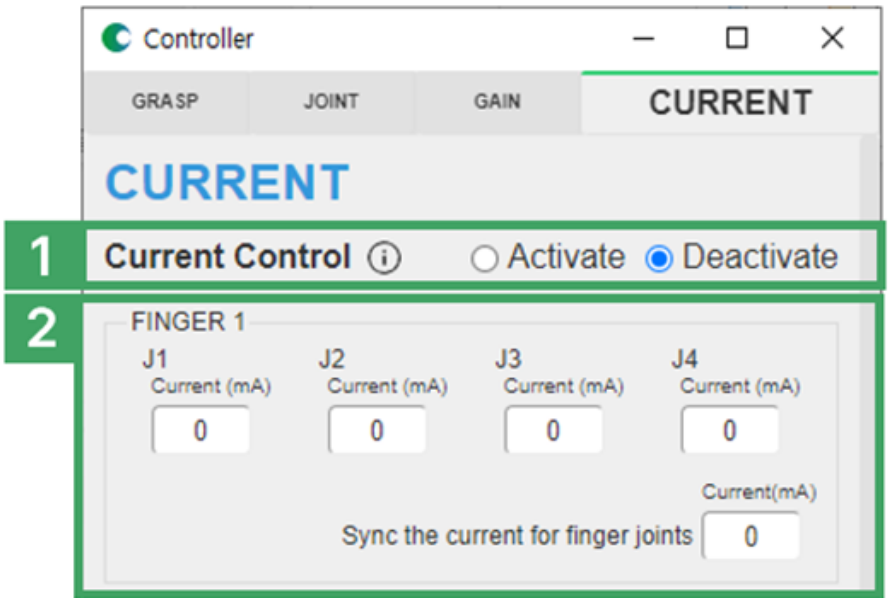
Note

* Number of joints and fingers varies by model
11.2 Setting Gain P
11.3 Setting Gain D
11.4 Setting Gain I
11.5 Setting the Gain PID
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.4 Setting the current

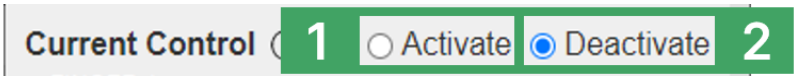
6.4.1 Screen layout



In Current Control Mode, you can control the gripper joints using current commands. However, setting PID gain values is not available in this mode.

6.4.2 Setting the current mode

Description



When current control mode is enabled, each joint will operate with current control.

Item

Number	Name	Description
1	Activate	Enable Current Control
2	Deactivate	Disable current control

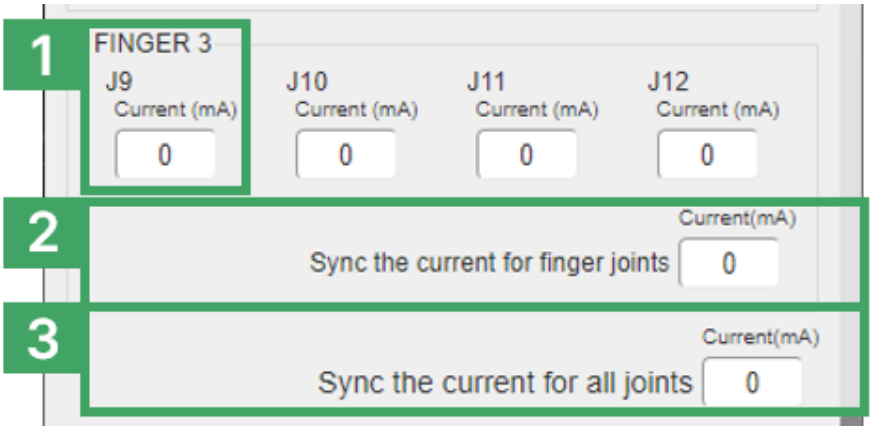
Note

13.4.1 SetCurrentControlMode
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

6.4.3 Setting the current

Description



The current setting is applied when current control mode is enabled. The current value can be set up to 150 mA for M models and up to 360 mA for S models.

Item

Number	Name	Description
1	J1-Jn Current	Set a current value for that joint
2	Sync the current for finger joints	Set a current value for all joints of the selected finger
3	Sync the current for all joints	Set a current value for all joints

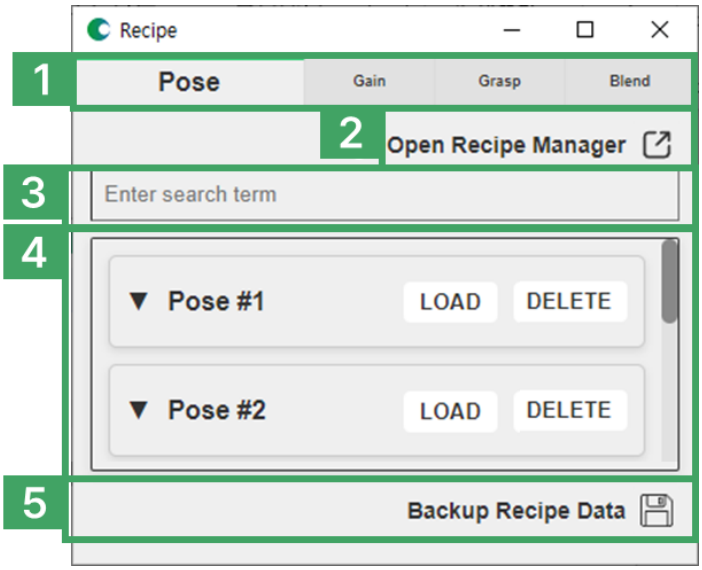
Note

13.4 Current Control
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

Chapter 7. Recipe settings screen

7.1 Screen layout



The recipe settings screen displays the currently stored recipes. You can search for a recipe to review it, press the LOAD button to execute the selected recipe, or press DELETE to remove it. You can back up the configured recipes within the program and save them either to the gripper or to a PC. In Operator Mode, this function is labeled as EEPROM WRITE. When EEPROM WRITE is performed, the current settings are permanently stored in the gripper's memory. This refers to non-volatile memory, meaning the data will remain even after the gripper is powered off and restarted.

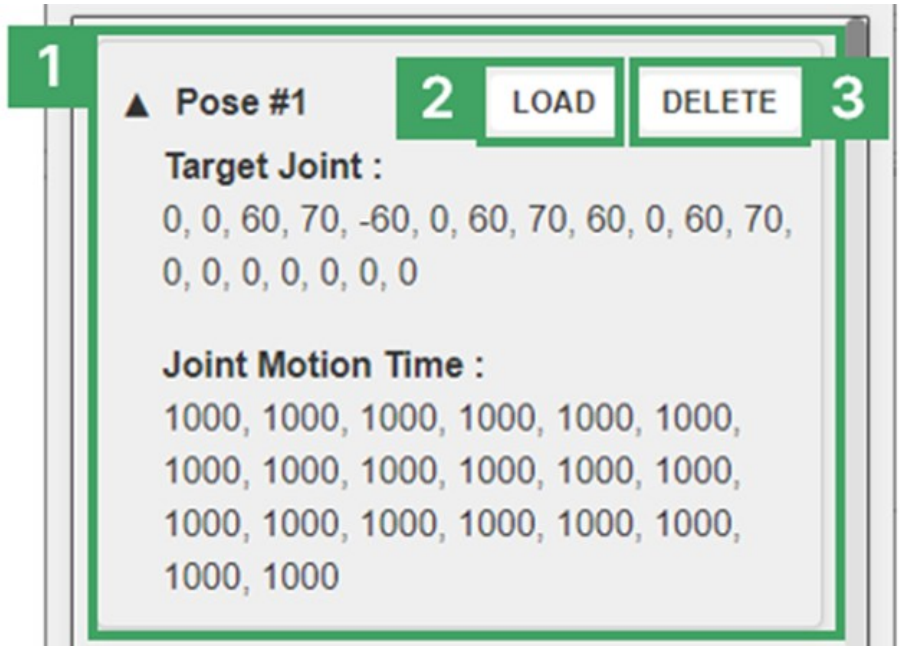
Item

Number	Name	Description
1	Recipe Menu	Select a recipe item
2	Open Recipe Manager	Open the admin to edit or add a recipe
3	Search Recipe	Features for searching for recipes
4	Show Recipe List	Show a list of saved recipe items
5	Backup Recipe Data	Back up recipe items depending on the mode used ▢ Operator Mode: EEPROM WRITE ▢ Developer Mode: BACKUP RECIPE DATA

ⓒ 2026 Tesollo Inc. All rights reserved

7.2 Pose recipes

7.2.1 Screen layout



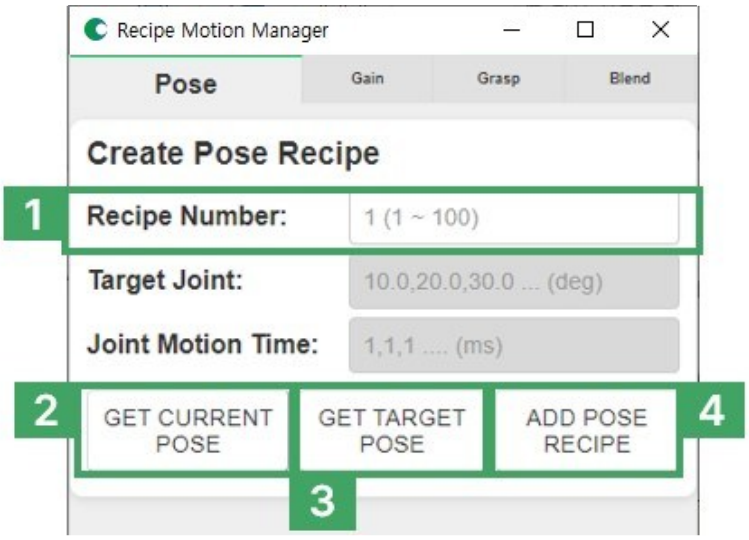
The currently saved pose recipes are displayed. Each pose recipe includes target joint values and motion time. You can load a recipe by clicking the “LOAD” button or delete it by clicking the “DELETE” button.

Item

Number	Name	Description
1	Recipe information	Show recipe information ▢ Target Joint Value ▢ Joint Motion Time
2	LOAD	Load the selected recipe into the manager screen and operate the gripper
3	DELETE	Delete the recipe

© 2026 Tesollo Inc. All rights reserved

7.2.2 Setting pose recipes



Enter the number of the recipe you want to save and load the current joint position values or target joint position values. Click the Add Pose Recipe button to save the recipe. GET TARGET POSE retrieves the joint degree values entered in '6.2 Position control'.

Item

Number	Name	Description
1	Recipe Number	Enter the recipe number to save ▢ In Operator Mode, up to 100 items can be stored.
2	GET CURRENT POSE	Get current joint position and motion time values
3	GET TARGET POSE	Get target joint position and motion time values
4	ADD POSE RECIPE	A pose recipe with the recipe number you set

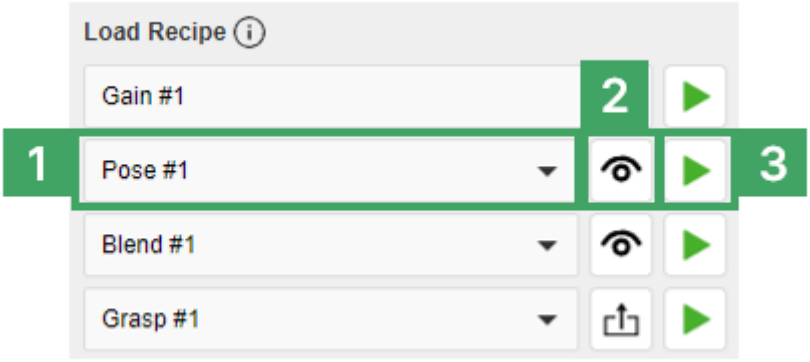
Note

12.1.1 UpdateRecipeCurrentPoseData
12.1.2 UpdateRecipeTargetPoseData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

7.2.3 Operation pose recipes

Description



On the main screen, select and the desired pose recipe.

Item

Number	Name	Description
1	Choose a Pose Recipe	Choose a Pose Recipe
2	Preview a Pose Recipe	You can preview the pose recipe movements through simulation The simulation runs only while the button is being pressed
3	Importing Pose Recipes	Importing Pose Recipes

Note

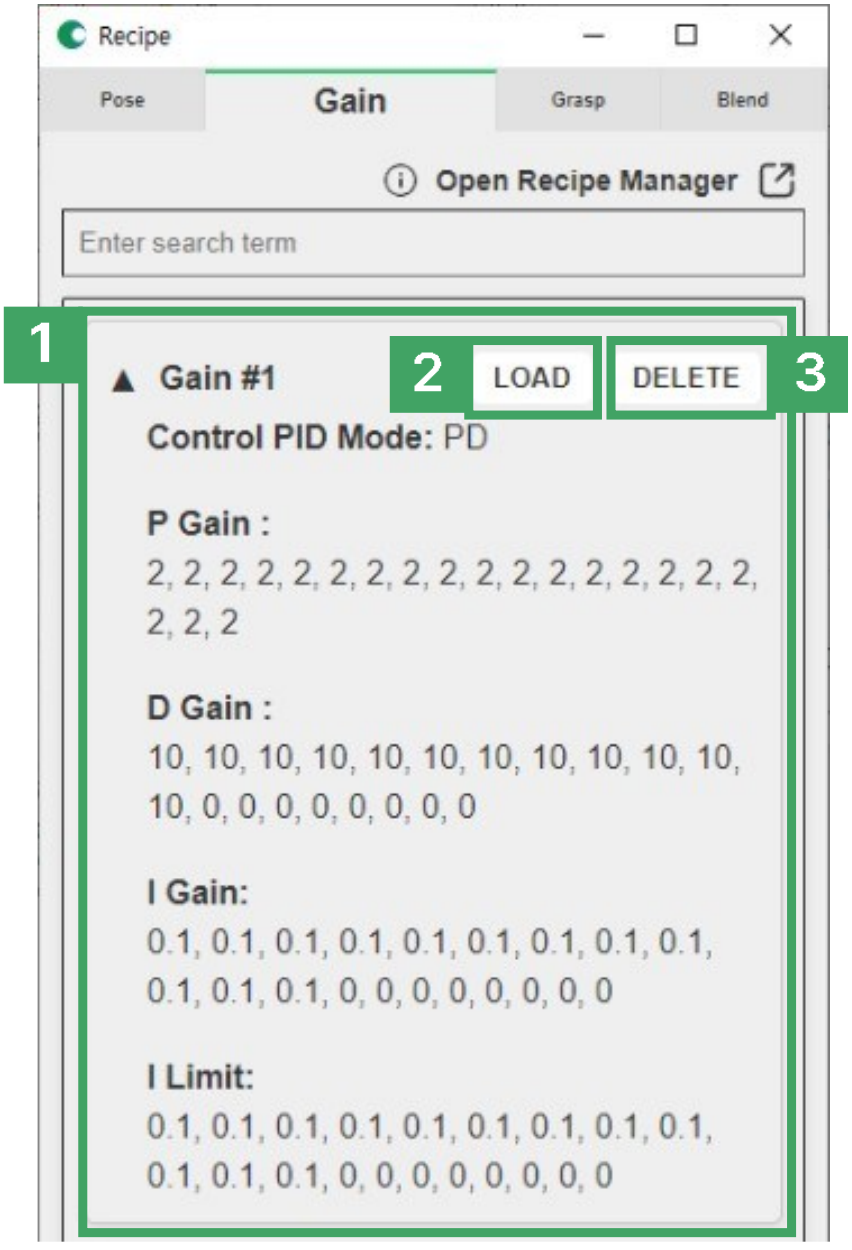
12.1.1 UpdateRecipeCurrentPoseData
12.1.2 UpdateRecipeTargetPoseData
12.1.3 LoadRecipePoseData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

7.3 Gain recipes

7.3.1 Screen layout

Description



Displays the currently saved gain recipe. Gain recipes include P gain, D gain, I gain, and maximum error integral. You can modify the recipe by pressing the LOAD button or delete the recipe by pressing DELETE button.

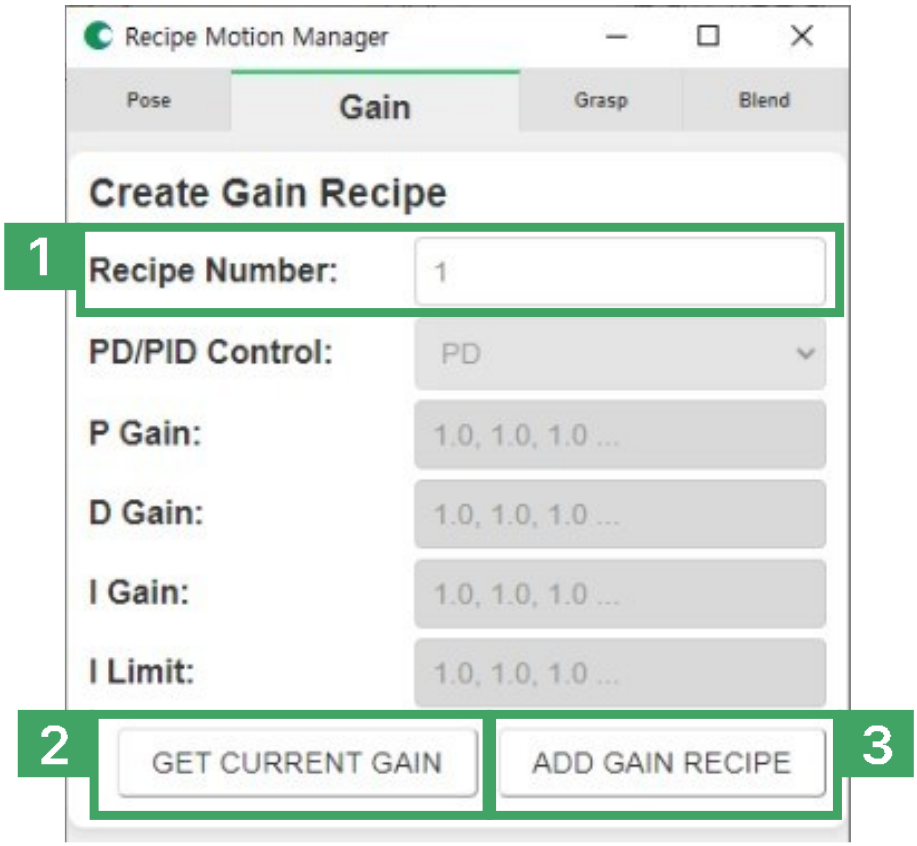
© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Recipe information	Show recipe information ☐ P Gain ☐ D Gain ☐ I Gain ☐ I Limit. Maximum error integral
2	LOAD	Load the selected recipe into the manager screen and operate the gripper
3	DELETE	Delete the recipe

7.3.2 Setting gain recipes

Description



Enter the desired recipe number and retrieve the current gain value. Click the Add Gain Recipe button to save the recipe.

© 2026 Tesollo Inc. All rights reserved

Item

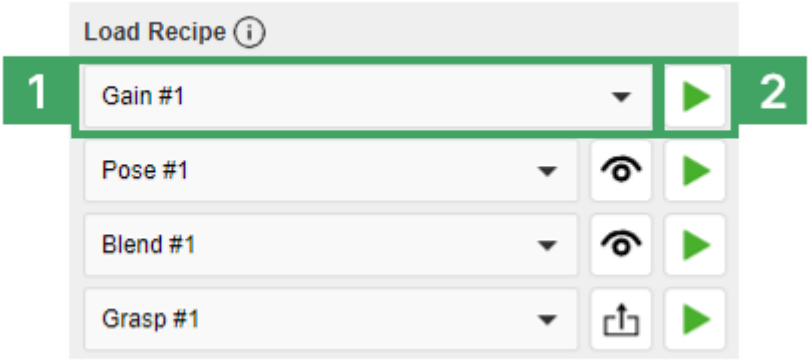
Number	Name	Description
1	Recipe Number	Enter the recipe number to save ▢ In Operator Mode, up to 20 items can be stored.
2	GET CURRENT GAIN	Retrieve current gain values
3	ADD GAIN RECIPE	Save the gain recipe with the recipe number you set

Note

12.2.1 UpdateRecipeGainData
* See 'Appendix 2. Error list' for errors

7.3.3 Operation gain recipes

Description



On the main screen, select and the desired gain recipe.

Item

Number	Name	Description
1	Select a gain recipe	Select a gain recipe
2	Importing a gain recipe	Importing a gain recipe

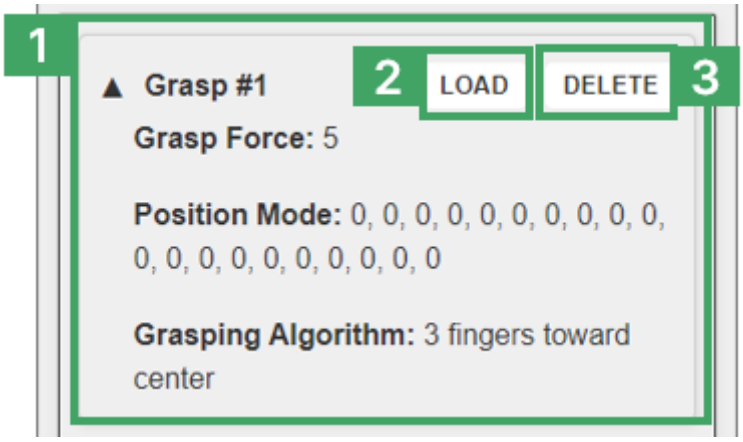
Note

12.2.2 LoadRecipeGainData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

7.4 Grasp recipes

7.4.1 Screen layout



The currently saved grasp recipes are displayed. Each grasp recipe includes settings such as grasping force, joint control, grasping mode, and grasping options.

You can modify a recipe by clicking the “LOAD” button or delete it by clicking the “DELETE” button.

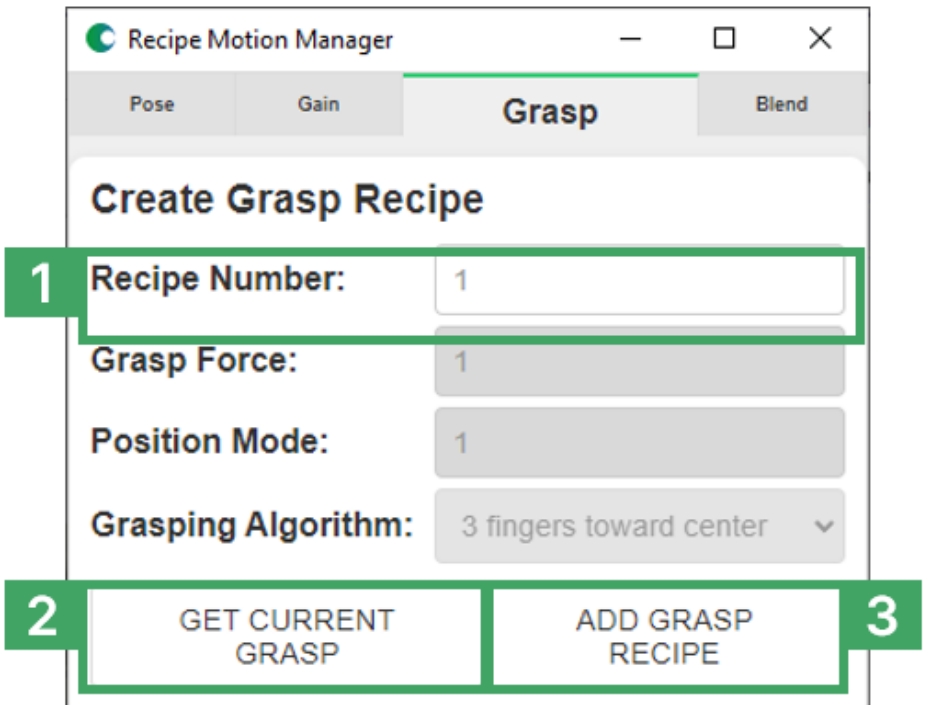
Item

Number	Name	Description
1	Recipe information	Show recipe information ▢ Grasp Force ▢ Position Mode ▢ Grasp Algorithm
2	LOAD	Load the selected recipe into the manager screen and operate the gripper
3	DELETE	Delete the recipe

© 2026 Tesollo Inc. All rights reserved

7.4.2 Setting grasp recipes

Description



Enter the recipe number you wish to save, then load the current grasping configuration.

Click the “ADD GRASP RECIPE” button to save the recipe.

Item

Number	Name	Description
1	Recipe Number	Enter the recipe number to save ▢ In Operator Mode, up to 20 items can be stored.
2	GET CURRENT GRASP	The current grasp value
3	ADD GRASP RECIPE	Save a grasp with a set recipe number

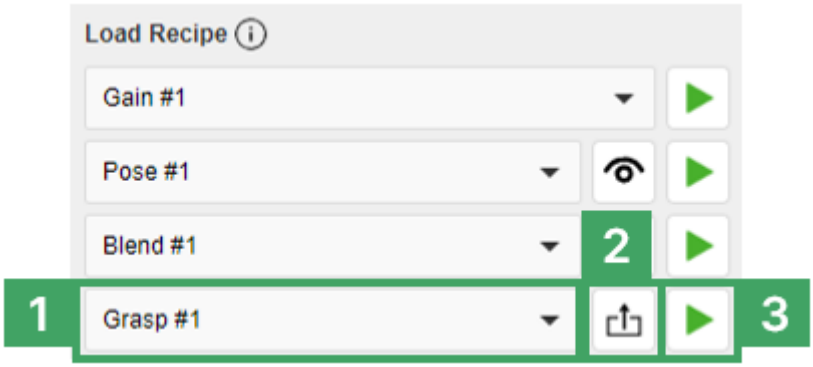
Note

12.3.1 UpdateRecipeGraspData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

7.4.3 Operation grasp recipes

Description



Select and load the desired recipe from the main screen.

Item

Number	Name	Description
1	Select a grasp recipe	Select the grasp recipe to use from the list
2	LOAD	Load the selected grasp recipe
3	GRASP	▶: Perform a grasp action ■: Perform a grasp release

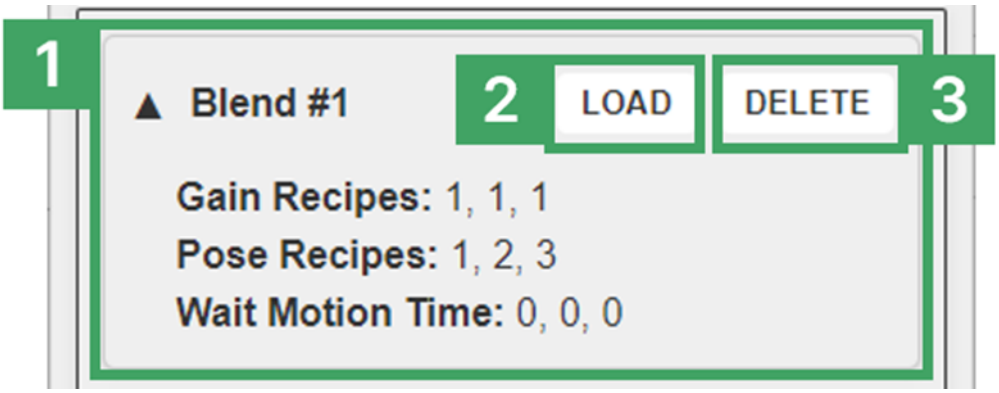
Note

12.3.2 LoadRecipeGraspData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

7.5 Blend recipes

7.5.1 Screen layout



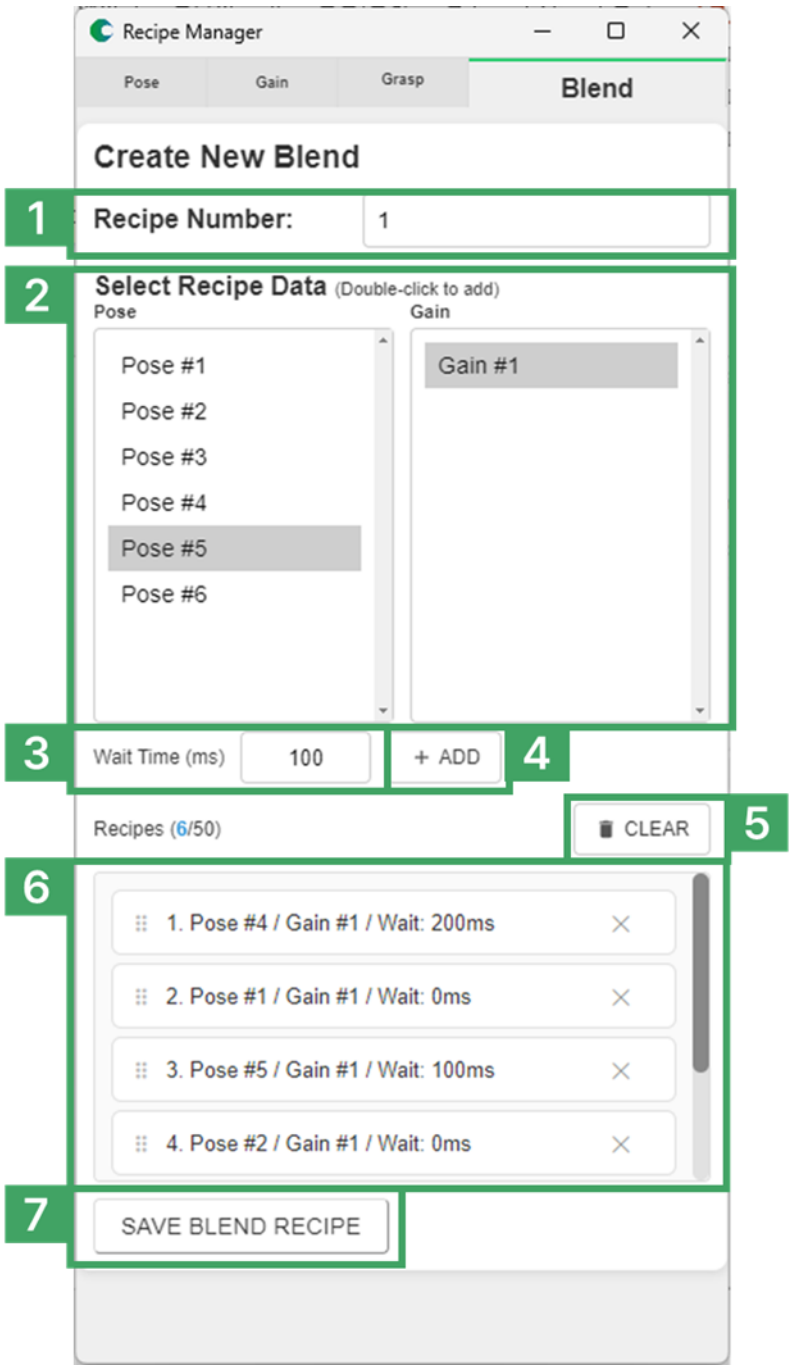
The currently saved recipes are displayed. Each blend recipe includes a gain recipe, a pose recipe, and motion time. You can load a recipe by clicking the “LOAD” button or delete it by clicking the “DELETE” button.

Item

Number	Name	Description
1	About blend recipes	Show recipe information ▢ Gain Recipes ▢ Pose Recipes ▢ Wait Motion Time(ms)
2	LOAD	Load the current blend recipe
3	DELETE	Delete the currently selected blend recipe

© 2026 Tesollo Inc. All rights reserved

7.5.2 Setting blend recipes



Specify the blend recipe number you want to save, set the pose, gain, and wait time, then click the Step 4 button “ADD SELECTED RECIPE” to add the blend motion. Repeat this process as needed to add all desired blend motions. Once completed, click the Step 7 button “SAVE BLEND RECIPE” to save the blend recipe.

© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Recipe Number	Enter the recipe number to save ▢ In Operator Mode, up to 10 items can be stored.
2	Select Recipe Data	Choose a Pose, Gain Recipe
3	Wait Time	Sets the wait time for blend motion. After executing the current recipe, the system waits for the specified time before executing the next recipe.
4	ADD	Add the selected pose, gain recipe, and wait time to the list ▢ Store up to 50 in a single blend
5	CLEAR	Resetting the recipe list
6	Blend List	A blend list with the added pose recipe, gain recipe, and motion time
7	SAVE BLEND RECIPE	Save a blend recipe with a set recipe number

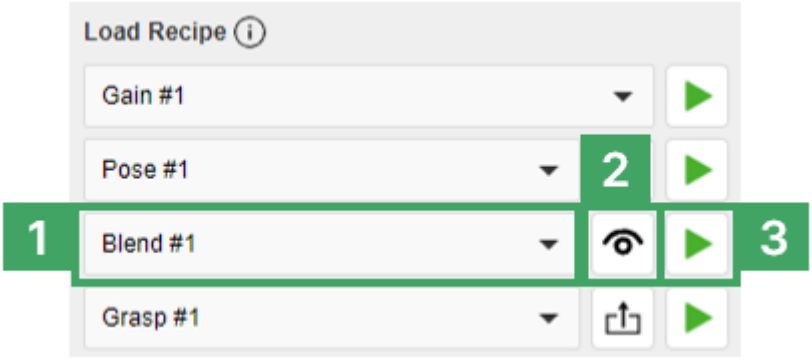
Note

12.4.1 UpdateBlendJoint
12.4.2 ClearMoveBlendJoint
12.4.3 AddMoveBlendJoint
12.4.4 SetMoveBlendJoint
12.4.5 StartMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

7.5.3 Operation blend recipes

Description



On the main screen, select and the desired blend recipe.

Item

Number	Name	Description
1	Select a blend recipe	Select a blend recipe
2	Preview blend recipes	Preview pose recipe movements through simulation. The simulation runs while the button is being pressed
3	Importing blend recipes	Importing blend recipes

Note

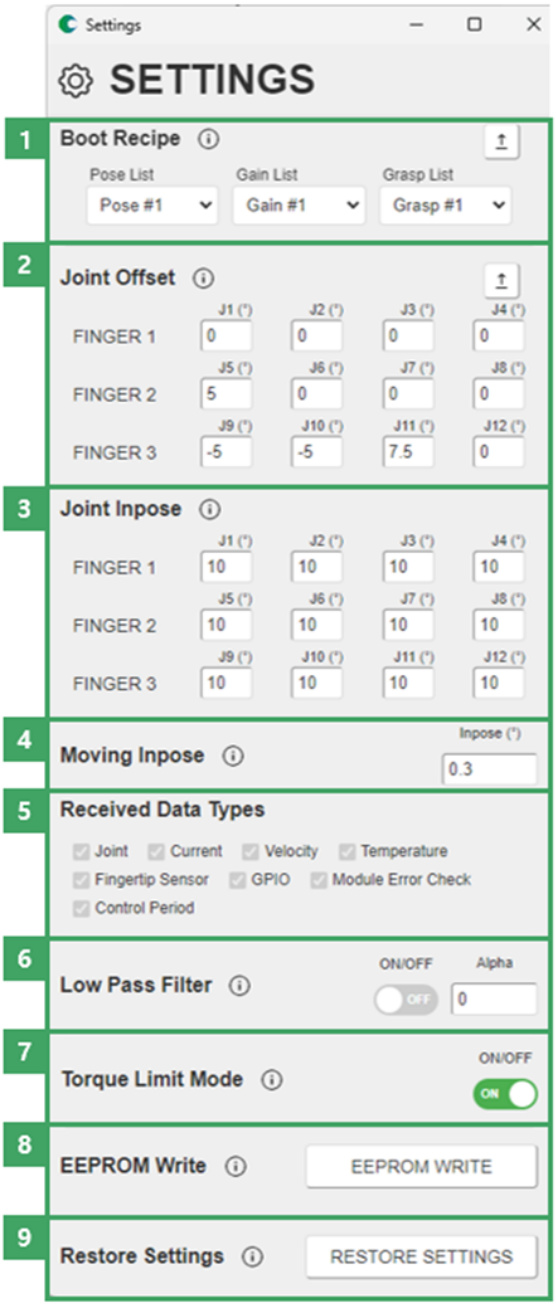
12.4.5 StartMoveBlendJoint
12.4.6 StopMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

Chapter 8. Additional features

8.1 Settings

8.1.1 Screen layout



The Settings screen is where you can change gripper settings, such as the gripper's boot recipe and offset.

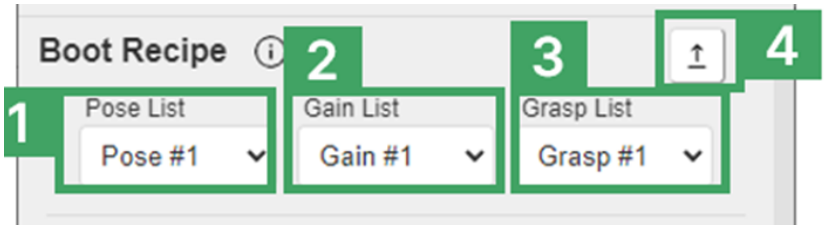
© 2026 Tesollo Inc. All rights reserved

Item

Number	Name	Description
1	Boot Recipe	Setting up recipes at the start of gripper control
2	Joint Offset	Set the offset value for each joint of the gripper
3	Joint Inpose	Setting the scope for joint arrival checks
4	Moving Inpose	Setting the Range for Checking Joint Movement
5	Received Data Types	Display data received from the gripper Can be configured before connection
6	Low Pass Filter	Setting up a low-pass filter
6	Torque Limit Mode	Setting the Torque Limit Mode M Model only ▢ Use with caution
7	Backup Settings	Perform backup depending on the mode used ▢ Operator Mode: EEPROM WRITE ▢ Developer Mode: BACKUP SETTINGS
8	Restore Settings	Restore the system

ⓒ 2026 Tesollo Inc. All rights reserved

8.1.2 Boot Recipe



The boot recipe function allows you to specify the recipe that will be applied when the gripper is powered on and control begins.

Item

Number	Name	Description
1	Pose Boot Recipes	Setting up a Pose Recipe to Set Up on Initial Bootup
2	Gain Boot Recipe	Setting up a gain recipe to set on initial boot
3	Grasp Boot Recipes	Setting up grasp recipes to set up on initial boot
4	Apply Button	Apply the configured boot recipe data

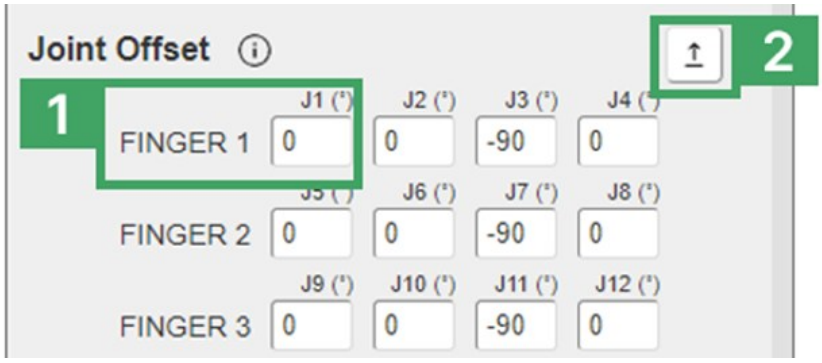
Note

* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.3 Joint Offset

Description



Joint offset settings are used to define the joint's zero position in software when the physical position of a joint is misaligned. If a specific joint deviates from the zero pose when the gripper is moved to the zero position, manually align the joint to the correct zero pose, then record the change in position as the joint offset.

Item

Number	Name	Description
1	Joint Offset J1-Jn	Set J1 - Jn Joint Offset Values (Unit: degree)
2	Apply button	Apply the configured Joint Offset, Joint Inpose, and Moving Inpose all at once

Note

10.1.2 SetGripperOption
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.4 Joint Inpose

Description



Joint Inpose is a range-setting feature used to determine whether each joint has reached its target position. For example, if Joint 1 (J1) is set to a tolerance of 3 degrees and the current joint position is 30 degrees, then a target position of 60 degrees will be considered "reached" once the joint reaches 57 degrees.

This feature is particularly useful when performing continuous position movements, allowing the next operation to begin even before the joint has fully reached its target.

Press the "Apply" button in section '8.1.3 Joint Offset' to apply the settings.

Item

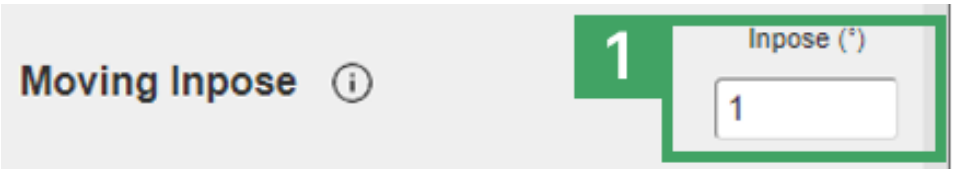
Number	Name	Description
1	J1 - Jn Inpose	Set the J1 to J20 joint arrival check range (unit: degree)

Note

10.1.2 SetGripperOption
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.5 Moving Inpose



Moving Inpose is a feature that sets a range for checking joint movement. If the Inpose value is 1 degree, the joint is considered to have moved if the amount of change in the joint is more than 1 degree.

Press the "Apply" button in section '8.1.3 Joint Offset' to apply the settings.

Item

Number	Name	Description
1	Inpose	Setting the range of motion checks for a joint (Unit: degree)

Note

10.1.2 SetGripperOption
* See 'Appendix 2. Error list' for errors

8.1.6 Low Pass Filter



Set the low-pass filter.

Item

Number	Name	Description
1	ON / OFF	Enable or disable filters
2	Alpha	Set filter gain values

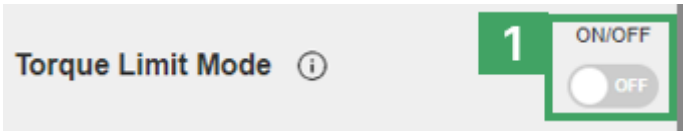
Note

10.3.1 SetLowPassFilterAlpha
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.7 Torque Limit Mode

Description



Torque Limit Mode is a feature that removes the torque limit on the gripper, so that you can use the maximum torque rather than the rated torque provided by the gripper. This feature is supported only on M models.

Item

Number	Name	Description
1	ON/OFF	Enable or disable filters

Note

*** Caution**

Using 'Torque Limit Mode' off in Operator Mode or using stall torque in Developer Mode may cause a rapid increase in product temperature and potential product damage. Please note that repairs (A/S) may be difficult in case of failure.

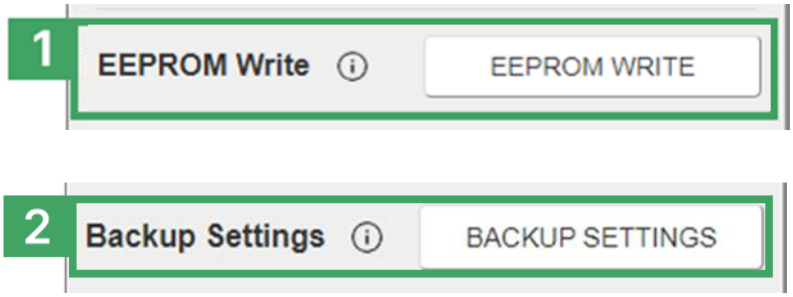
10.3.5 SetTorqueLimitMode

* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.8 Backup Settings / EEPROM Write

Description



Backup Settings / EEPROM Write is a function that saves the currently applied configuration data to the PC or the gripper. In Operator Mode, backup is performed automatically when communication is connected.

Please note that if the gripper power is turned off or communication is disconnected while saving data to EEPROM, data corruption may occur.

When DGManager is launched for the first time, a guide screen explaining how to use it is displayed. After that, it can be accessed in 4.2.1 Screen layout, item 13: About the program.

Item

Number	Name	Description
1	EEPROM WRITE	In Operator Mode, save the currently configured values to the gripper.
2	BACKUP SETTINGS	In Developer Mode, back up the currently configured settings to the PC

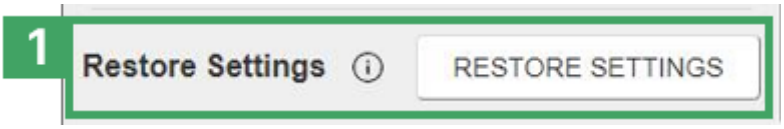
Note

10.3.8 BackupRecipeData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.1.9 Restore Settings

Description



The Restore Settings feature allows you to load a backup file from an external path and restore the data to the gripper.

The related files are saved either when connecting in Operator Mode or during Backup Settings in Developer Mode mentioned in '8.1.8 Backup Settings / EEPROM Write'.

The save path is as follows:

{installation path}\resources\config\{Model Name}\{Control Mode}

Example) DGManager-win32-x64\resources\config\DG-3F-B\Operator

When DGManager is launched for the first time, a guide screen explaining how to use it is displayed. After that, it can be accessed in 4.2.1 Screen layout, item 13: About the program.

Item

Number	Name	Description
1	RESTORE SETTINGS	Restoring data from a backup file

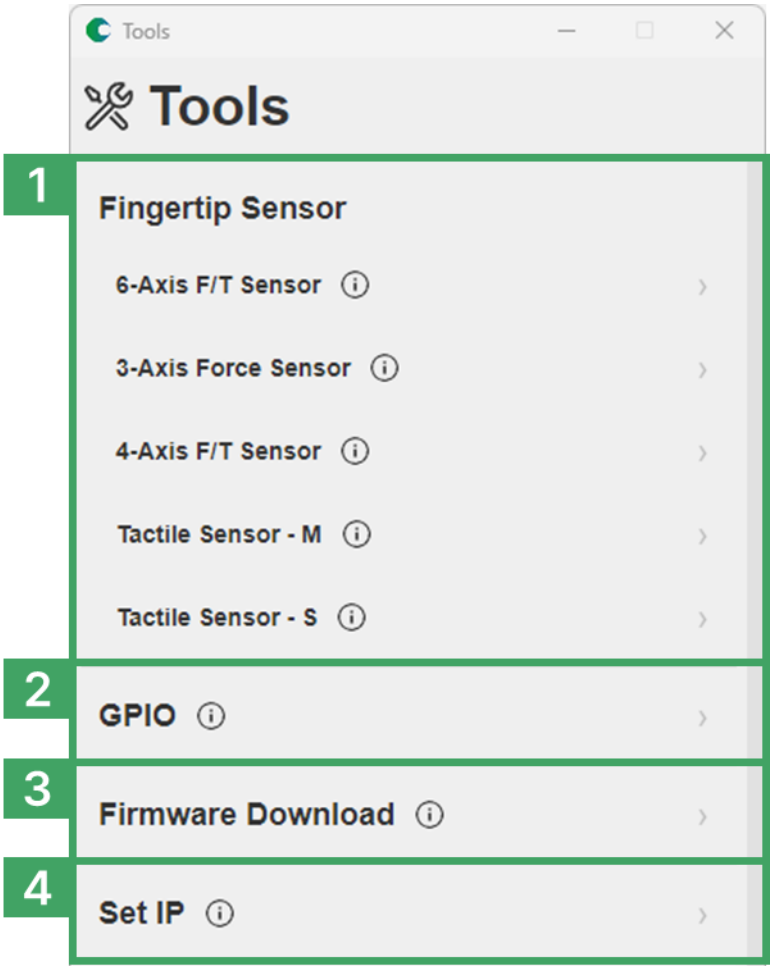
Note

10.3.9 RestoreRecipeData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.2 Tools

8.2.1 Screen layout

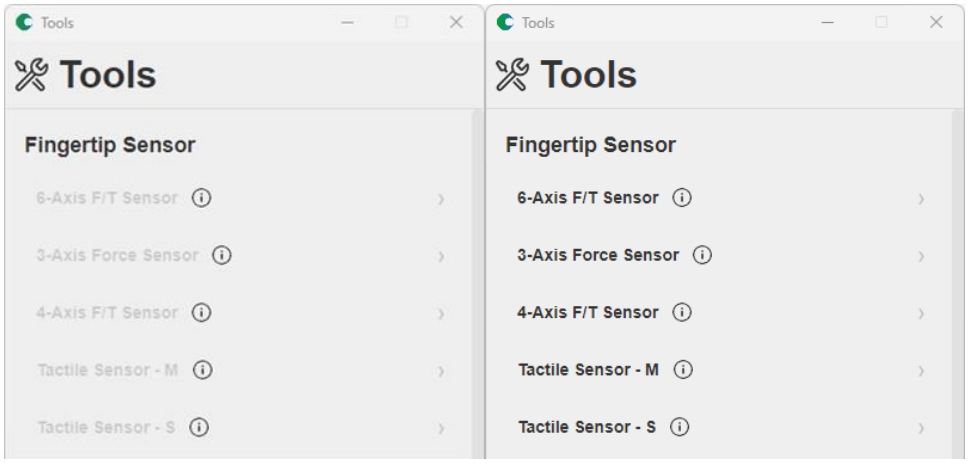


Number	Name	Description
1	Fingertip Sensor	Monitoring Fingertip Sensor Data
2	GPIO	Open the Panel for Gripper GPIO Control
3	Firmware Download	Select the gripper firmware file to download the gripper's firmware
4	Set IP	Set gripper IP and PORT

© 2026 Tesollo Inc. All rights reserved

8.2.2 Fingertip Sensor

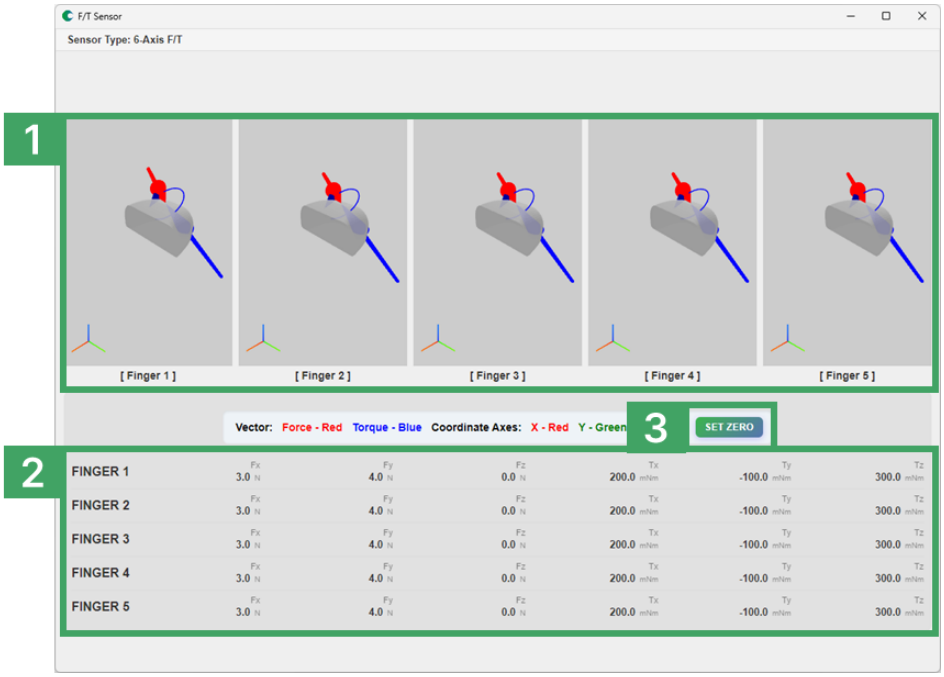
Description



The Delto Gripper can be equipped with sensor-integrated fingertips. The available fingertip versions differ depending on the M and S models. This feature is not available on firmware that does not support the fingertip option, and the corresponding fingertip sensors are activated based on the fingertip option stored in the firmware.

8.2.2.1 6-Axis F/T Sensor

Description



© 2026 Tesollo Inc. All rights reserved

6-axis force/torque sensor data can be monitored, allowing verification of force and torque values. This feature is available only on M models.

Item

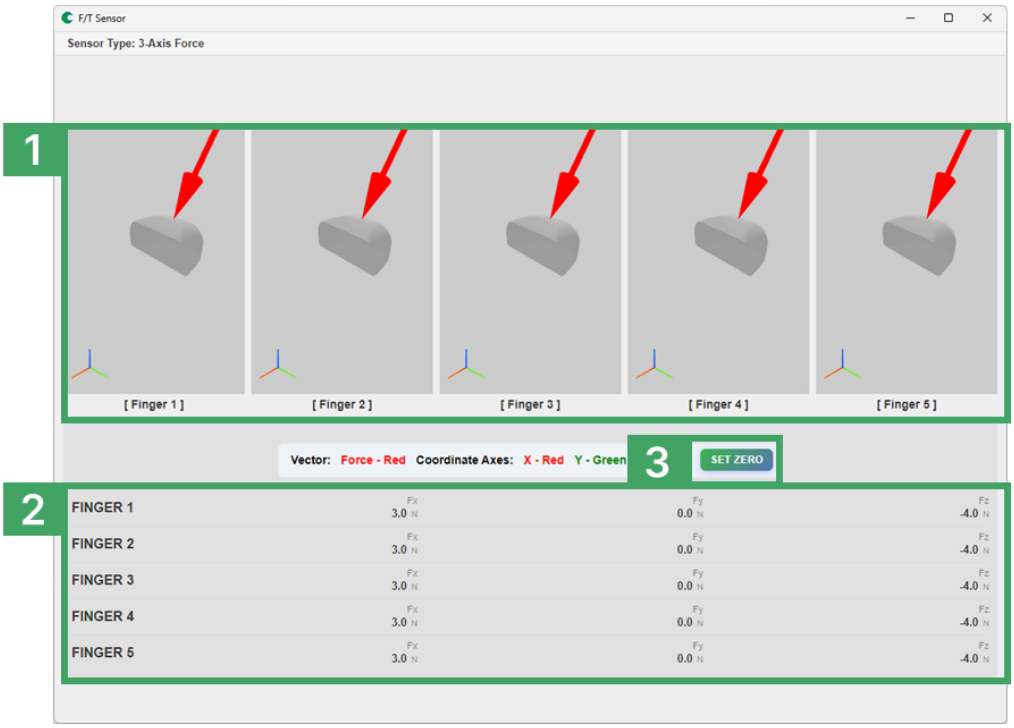
Number	Name	Description
1	3D Visualization of Data	3D Visualization of FT Sensor Data
2	FT Sensor Data	FT Sensor Data (Unit: N, mN·m)
3	FT Sensor Reset	Reset FT Sensor Data to 0

Note

10.2.5 CallbackForOnReceivedFingertipSensorData
11.8.1 SetFingerTipDataZero
* See 'Appendix 2. Error list' for errors

8.2.2.2 3-Axis Force Sensor

Description



3-axis force sensor data can be monitored, allowing verification of force values. This feature is available only on M models.

© 2026 Tesollo Inc. All rights reserved

Item

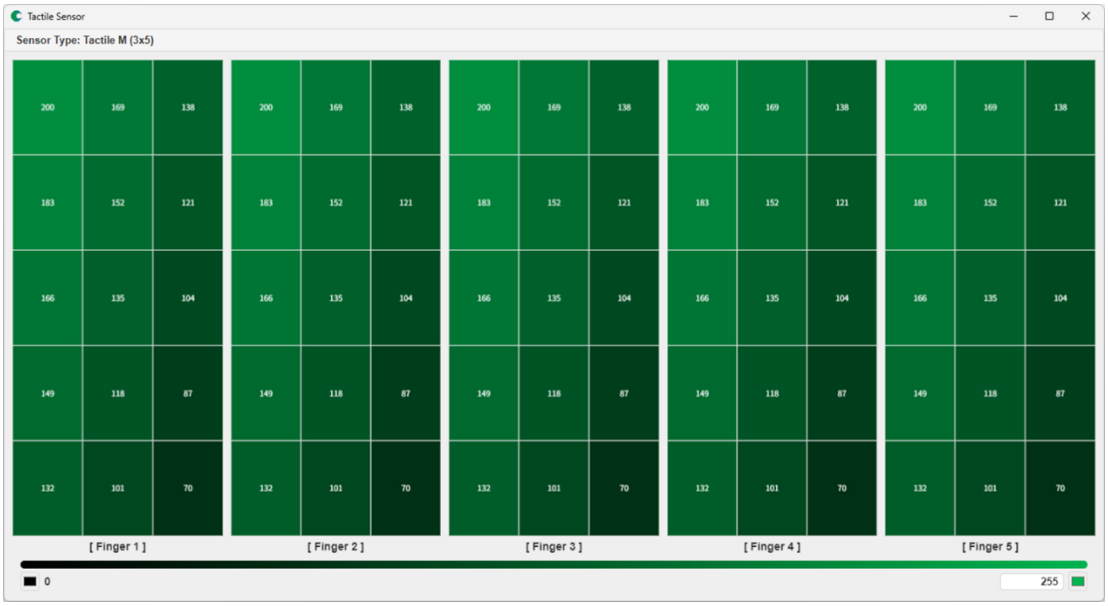
Number	Name	Description
1	3D Visualization of Data	3D Visualization of FT Sensor Data
2	FT Sensor Data	FT Sensor Data (Unit: N)
3	FT Sensor Reset	Reset FT Sensor Data to 0

Note

10.2.5 CallbackForOnReceivedFingertipSensorData
11.8.1 SetFingerTipDataZero
* See 'Appendix 2. Error list' for errors

8.2.2.3 Tactile Sensor – M/S

Description



Tactile sensor data can be monitored. The M model consists of 3×5 nodes, and the S model consists of 3×6 nodes, with intensity values available for each node.

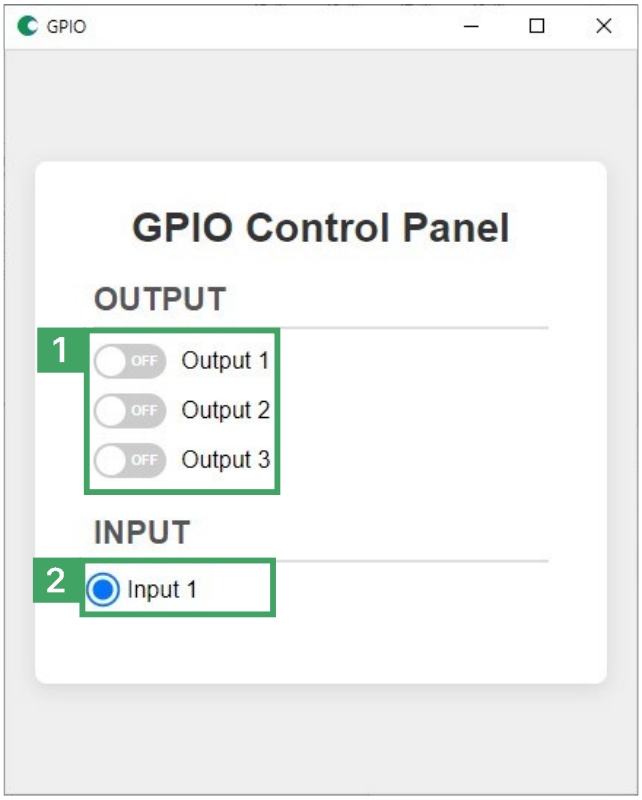
Note

10.2.5 CallbackForOnReceivedFingertipSensorData
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.2.3 GPIO

Description



GPIOs are built into the gripper controller to check output ON / OFF and input status. This works in conjunction with an 8PIN cable connected to the gripper power supply, allowing the gripper GPIO controller to send and receive IO signals to and from external devices.

Item

Number	Name	Description
1	Output	Performing Gripper Output ON / OFF
2	Input	Show Gripper Input Status

Note

10.2.6 CallbackForOnReceivedGPIOData
10.3.3 SetGPIOOutput
10.3.4 SetGPIOOutputAll
* See 'Appendix 2. Error list' for errors

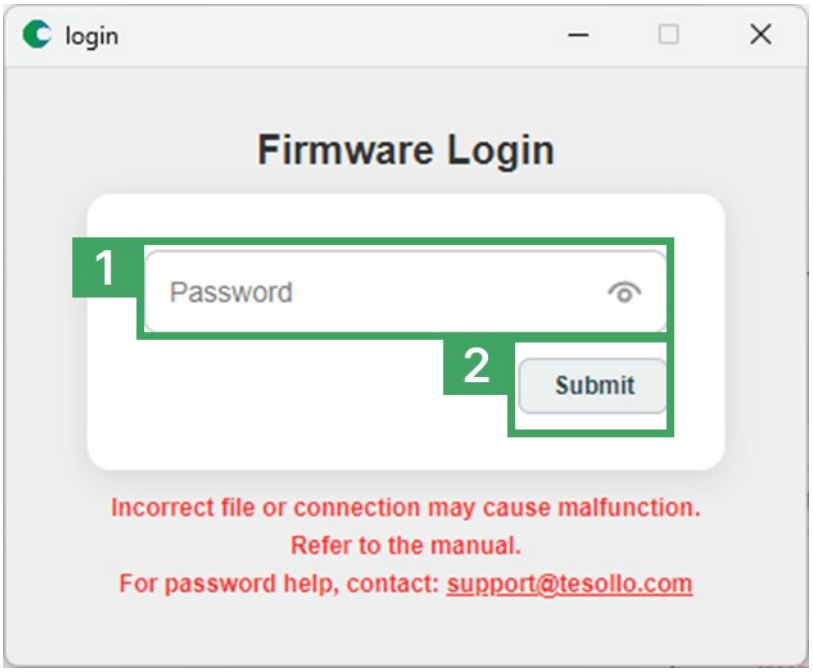
© 2026 Tesollo Inc. All rights reserved

8.2.4 Firmware Download

Firmware Download is a function used to update the main controller firmware of the gripper. A password is required to use this function. If you need the password, please contact our support team at support@tesollo.com.

8.2.4.1 Login Screen

Description



This is a login screen designed to prevent unauthorized firmware update attempts.

By entering the password, you can proceed to the firmware download screen.

Item

Number	Name	Description
1	Password Entry	For the password, please contact support@tesollo.com
2	Submit	Login

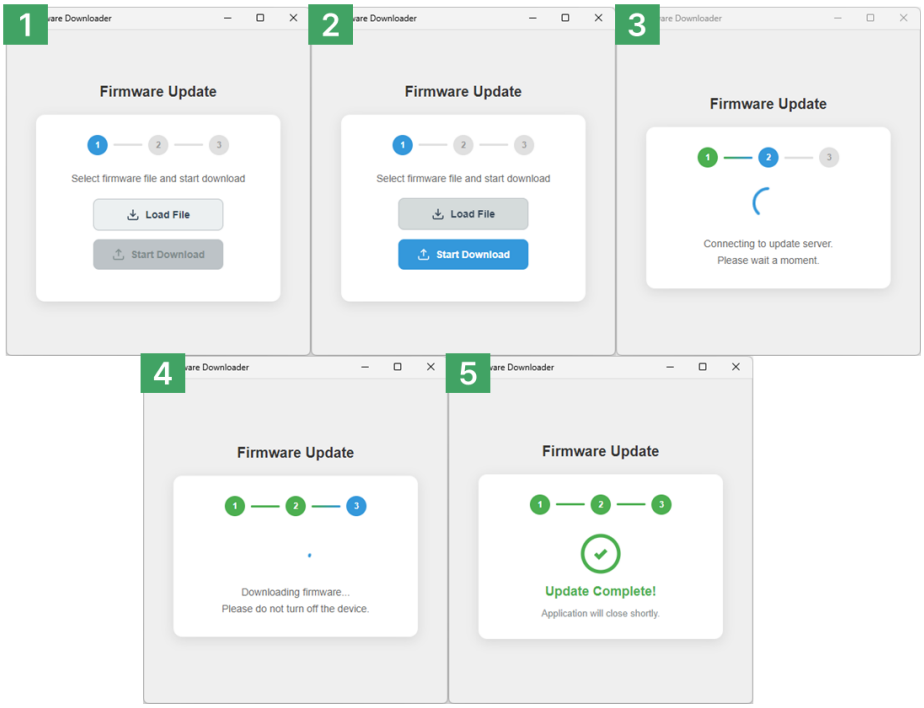
Note

* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.2.4.2 Firmware Download Screen

Description



When screen 1 appears, first select “Load File” to load the firmware file to be downloaded, then select “Start Download.” The process will proceed to screen 3, and when screen 5 is displayed, the firmware has been successfully downloaded.

Afterward, the gripper will automatically reboot, and the program must be restarted to reconnect.

Item

Number	Name	Description
1	LOAD FILE	Load firmware from storage
2	START DOWNLOAD	Perform firmware download and enter boot mode

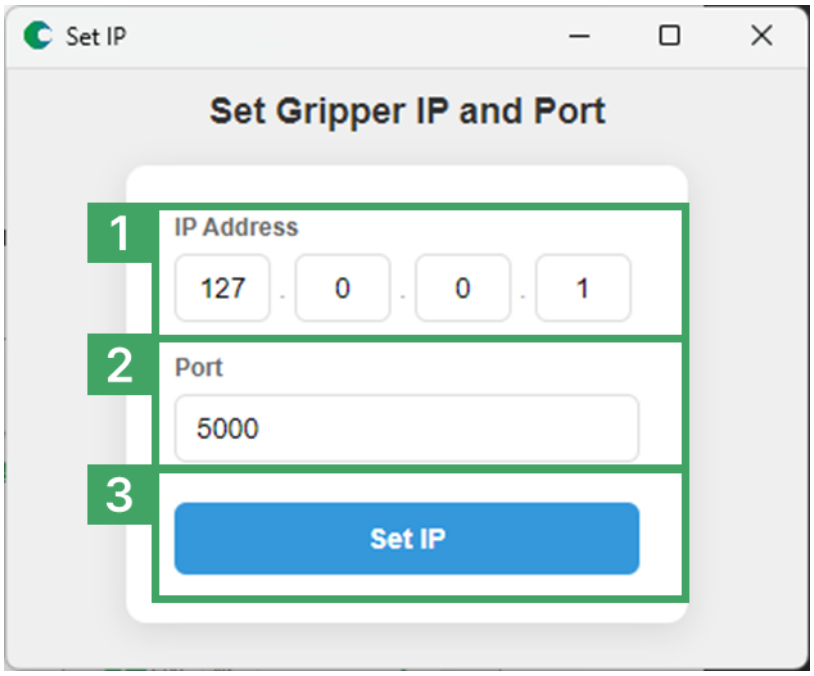
Note

10.3.2 SetBootMode
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.2.5 Set IP

Description



Set the IP and port of the gripper.

Item

Number	Name	Description
1	IP	Enter the IP address
2	Port	Enter the port number
3	Set IP	Set the entered IP address and port

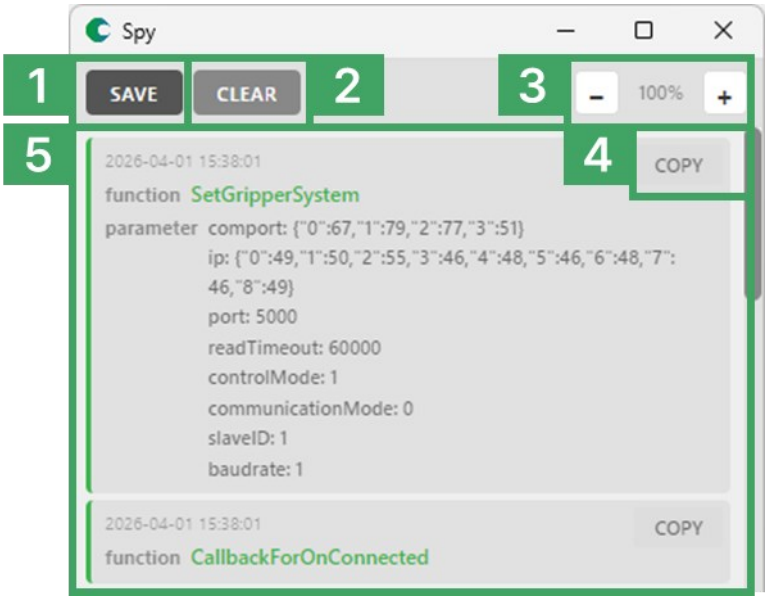
Note

10.1.7 SetI
* See 'Appendix 2. Error list' for errors

© 2026 Tesollo Inc. All rights reserved

8.3 Spy

The Spy is a screen where you can view logs of events that occur while using DG Manager. When you press a button in the Manager to operate the gripper, the corresponding command is displayed in function form. This feature allows you to easily understand how to control the gripper.



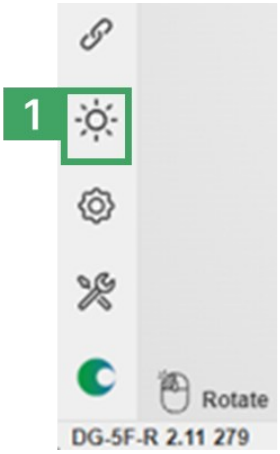
Number	Name	Description
1	SAVE	Save the spy log as a .txt file
2	CLEAR	Delete the entire contents of the current spy log
3	Font Size	Adjust the font size of the Spy log
4	COPY	Copy log
5	Display LOG	Real-time performed SDK functions and parameters

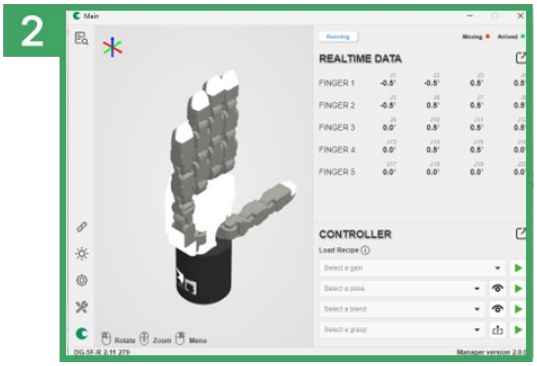
Note

Chapter 10 System functions
Chapter 11 Motion Setting Functions
Chapter 12 Recipe Functions
Chapter 13 Motion Functions

© 2026 Tesollo Inc. All rights reserved

8.4 Themes







Item

Number	Name	Description
1	Ability to change themes	Change light / dark themes
2	Light Themes	Uses dark text on a light background
3	Dark Themes	Uses light text on a dark background

© 2026 Tesollo Inc. All rights reserved

Chapter 9. SDK Structure

9.1 Define

This chapter describes the constant values defined for use in the SDK. If you enter a constant value that deviates from these constant values, it can cause an error in your program.

Name	Value	Description
MAX_JOINT_COUNT	20	Maximum number of joints
MAX_FINGER_COUNT	5	Maximum number of fingers
MAX_FINGER_JOINT_COUNT	4	Maximum number of joints per finger
MAX_BASE_JOINT_COUNT	2	Maximum number of joints per base
CARTESIAN_COORDINATE_POSE_COUNT	6	Number of Cartesian coordinate system data (X, Y, Z, RX, RY, RZ)
MAX_RECEIVED_DATA_SIZE	1024	Maximum communication data size
MAX_GRIPPER_IP_ADDRESS_SIZE	32	Gripper IP Size
MAX_GRIPPER_IP_BYTE_LENGTH	4	Number of gripper IP address zones
MAX_COMPORT_NAME_SIZE	32	Comport Name Size
MAX_BLEND_COUNT	10	Number of blend motions saved
MAX_BLEND_ADD_POSE_COUNT	50	Number of recipes that store a single blend action
MAX_RECIPE_POSE_COUNT	100	Number of recipe poses saved
MAX_RECIPE_GAIN_COUNT	20	Number of recipe gains saved
MAX_RECIPE_GRASP_COUNT	20	Number of recipe grasps saved
MAX_GRIPPER_GPIO_SIZE	4	Number of GPIOs (3 outputs, 1 input)
MAX_RECEIVED_DATA_TYPE_COUNT	7	Number of data types you can receive

© 2026 Tesollo Inc. All rights reserved

9.2 Enum

9.2.1 enum DG_RESULT

Error return codes used in the SDK

Value	Name	Description
0	DG_RESULT_NONE	Normal
Other	ERROR	* See 'Appendix 2. Error list' for errors

9.2.2 enum DG_MODEL

Enumerated constants for setting up gripper models

Value	Name	Description
0x0000	DG_MODEL_NONE	Model not set
0x1F02	DG_MODEL_DG_1F_M	M Model 1-Finger
0x2F02	DG_MODEL_DG_2F_M	M Model 2-Finger
0x3F01	DG_MODEL_DG_3F_B	B Model 3-Finger
0x3F02	DG_MODEL_DG_3F_M	M Model 3-Finger
0x4F02	DG_MODEL_DG_4F_M	M Model 4-Finger
0x5F12	DG_MODEL_DG_5F_LEFT	M Model 5-Finger Left-Hand
0x5F22	DG_MODEL_DG_5F_RIGHT	M Model 5-Finger Right-Hand
0x5F14	DG_MODEL_DG_5F_S_LEFT	S Model 20 DoF 5-Finger Left-Hand
0x5F24	DG_MODEL_DG_5F_S_RIGHT	S Model 20 DoF 5-Finger Right-Hand
0x5F34	DG_MODEL_DG_5F_S15_LEFT	S Model 15 DoF 5-Finger Left-Hand
0x5F44	DG_MODEL_DG_5F_S15_RIGHT	S Model 15 DoF 5-Finger Right-Hand

9.2.3 enum BLEND_MOTION_STATUS

Enumerated constants for checking state in blend Motion

Value	Name	Description
0	BLEND_MOTION_STATUS_STOP	Blend Motion stalled
1	BLEND_MOTION_STATUS_RUN	Blend Motion states
2	BLEND_MOTION_STATUS_COMPLETE	Blend action completion status

© 2026 Tesollo Inc. All rights reserved

9.2.4 enum COMMUNICATION_MODE

Enumerated constants to set gripper communication mode

Value	Name	Description
0	COMMUNICATION_MODE_ETHERNET	Ethernet communication connections
1	COMMUNICATION_MODE_RS485	Serial Communication Connection

9.2.5 enum CONTROL_MODE

Enumerated constants to set gripper control mode

Value	Name	Description
0	CONTROL_MODE_OPERATOR	Operator Mode
1	CONTROL_MODE_DEVELOPER	Developer Mode

9.2.6 enum GAIN_MODE

Enumerated constants to Set Gripper PID Control Mode

Value	Name	Description
0	GAIN_MODE_PD	PD Control Mode
1	GAIN_MODE_PID	PID Control Mode

9.2.7 enum DEVELOPER_MODE_COMMAND

Enumerated constants for gripper Developer Mode-specific commands

Value	Name	Description
1	DEVELOPER_MODE_COMMAND_GET_DATA	Receive set data
2	DEVELOPER_MODE_COMMAND_SET_RECEIVED_DATA	Set which data to receive
5	DEVELOPER_MODE_COMMAND_SET_DUTY	Enter Joint Duty
6	DEVELOPER_MODE_COMMAND_SET_GPIO	GPIO (Output) Settings
7	DEVELOPER_MODE_COMMAND_SET_IP_ADDRESS	Setting the Gripper IP and PORT

© 2026 Tesollo Inc. All rights reserved

8	DEVELOPER_MODE_COMMAND_GET_ID_AND_VERSION	Receive the gripper ID and firmware version
9	DEVELOPER_MODE_COMMAND_GET_JOINT_ID	Receive Gripper Module Anomaly ID
10	DEVELOPER_MODE_COMMAND_SET_BOOT_MODE	Enter boot mode
11	DEVELOPER_MODE_COMMAND_SET_FT_OFFSET_ZERO	Fingertip sensor offset setting

9.2.8 enum DEVELOPER_MODE_RECEIVED_DATA_TYPE

Enumerated constants of data types that can be received from the gripper

Value	Name	Description
1	DEVELOPER_MODE_RECEIVED_DATA_TYPE_JOINT	Joint position
2	DEVELOPER_MODE_RECEIVED_DATA_TYPE_CURRENT	Joint current
3	DEVELOPER_MODE_RECEIVED_DATA_TYPE_ TEMPERATURE	Joint temperature
4	DEVELOPER_MODE_RECEIVED_DATA_TYPE_VELOCITY	Joint velocity
5	DEVELOPER_MODE_RECEIVED_DATA_TYPE_ FINGER_FT_SENSOR	Fingertip FT sensor
6	DEVELOPER_MODE_RECEIVED_DATA_TYPE_GPIO	GPIO State
7	DEVELOPER_MODE_RECEIVED_DATA_TYPE_MODULE_ ERROR_CODE	Module error code
8	DEVELOPER_MODE_RECEIVED_DATA_TYPE_ CONTROL_PERIOD	Gripper internal control period

© 2026 Tesollo Inc. All rights reserved

9.2.9 enum DG_GRASP_MODE

Enumerated constants for setting the gripper grasp mode.

DG_GRASP_MODE_2F is only available for the DG-2F-M model.

DG_GRASP_MODE_3F is only available for DG-3F-B and DG-3F-M models.

DG_GRASP_MODE_4F is only available for the DG-4F-M model.

DG_GRASP_MODE_5F is only available for the DG-5F-R and DG-5F-L models.

Value	Name	Description
0	DG_GRASP_MODE_NONE	Not setting grasp mode
1	DG_GRASP_MODE_3F_3FINGER	Three-finger focus grasp mode
2	DG_GRASP_MODE_3F_2FINGER_1_AND_2	Focused grasp mode with 1 and 2 fingers
3	DG_GRASP_MODE_3F_2FINGER_1_AND_3	Focused Grasp mode with 1st and 3rd fingers
4	DG_GRASP_MODE_3F_2FINGER_2_AND_3	Focused grasp mode with 2 and 3 fingers
5	DG_GRASP_MODE_3F_3FINGER_PARALLEL	Three-finger parallel grasp mode
6	DG_GRASP_MODE_3F_3FINGER_ENVELOP	Three-finger envelop grasp mode
21	DG_GRASP_MODE_5F_5FINGER	Five-finger focus grasp mode
22	DG_GRASP_MODE_5F_3FINGER	Three-finger focus grasp mode
23	DG_GRASP_MODE_5F_3FINGER_PARALLEL	Three-finger parallel grasp mode
24	DG_GRASP_MODE_5F_2FINGER_1_AND_2	Focused grasp mode with finger 1 and fingers 2
25	DG_GRASP_MODE_5F_2FINGER_1_AND_3	Focused grasp mode with finger 1 and fingers 3
26	DG_GRASP_MODE_5F_2FINGER_1_AND_4	Focused grasp mode with finger 1 and fingers 4

© 2026 Tesollo Inc. All rights reserved

27	DG_GRASP_MODE_5F_2FINGER_1_AND_5	Focused grasp mode with finger 1 and fingers 5
28	DG_GRASP_MODE_5F_5FINGER_PARALLEL	Five-finger parallel grasp mode
29	DG_GRASP_MODE_5F_5FINGER_ENVELOP	Five-finger envelop grasp mode
31	DG_GRASP_MODE_4F_4FINGER	Four-finger focus grasp mode
32	DG_GRASP_MODE_4F_4FINGER_PARALLEL	Four-finger parallel grasp mode
33	DG_GRASP_MODE_4F_4FINGER_ENVELOP	Four-finger envelop grasp mode
34	DG_GRASP_MODE_4F_4FINGER_RIGHT_PARALLEL	Right-hand–shaped parallel grasp mode
35	DG_GRASP_MODE_4F_4FINGER_LEFT_PARALLEL	Left-hand–shaped parallel grasp mode
36	DG_GRASP_MODE_4F_2FINGER_1_AND_2	Focused grasp mode with finger 1 and fingers 2
37	DG_GRASP_MODE_4F_2FINGER_1_AND_4	Focused grasp mode with finger 1 and fingers 4
38	DG_GRASP_MODE_4F_2FINGER_3_AND_4	Focused grasp mode with finger 3 and fingers 4
51	DG_GRASP_MODE_2F_2FINGER	Focused grasp mode with finger 1 and fingers 2
52	DG_GRASP_MODE_2F_2FINGER_ENVELOP	Two-finger envelop grasp mode

© 2026 Tesollo Inc. All rights reserved

9.2.10 enum DG_SENSOR_TYPE

Enumerated constants for Fingertip Sensor Type Classification.

Value	Name	Description
1	DG_SENSOR_TYPE_FT_SENSOR_6_AXIS	6 Axis Force/Torque Sensor
2	DG_SENSOR_TYPE_FT_SENSOR_3_AXIS	3 Axis Force/Torque Sensor
3	DG_SENSOR_TYPE_TACTILE_M	M Series Tactile sensor
5	DG_SENSOR_TYPE_TACTILE_S	S Series Tactile sensor

© 2026 Tesollo Inc. All rights reserved

9.3 Struct

9.3.1 struct ReceivedGripperData

Struct for checking the data provided by the gripper

Data Types	Name	Size (byte)	Description
float	joint	80	Position information for gripper joints (Unit: degree)
int	current	80	Current Information for Gripper Joints (Unit: mA)
int	velocity	80	Velocity Information for Gripper Joints (Unit: rpm) (DG-3F-B models are not supported)
float	temperature	80	Temperature information for gripper joints (Unit: °C) (DG-3F-B models are not supported)
int	moving	4	Movement state information for all joints of the gripper
int	targetArrived	4	Target Position Arrival Status Information for All Gripper Joints
int	blendMoveState	4	Status information when using the Blend Joint feature
Int	currentBlendIndex	4	Current running blend index information
int	productID	4	Gripper ID
int	firmwareVersion	4	Gripper firmware version
int	moduleErrorCode	4	Module error code
int	controlPeriod	4	Gripper internal control period

Note

Data type x MAX_JOINT_COUNT = 80 byte. joint, current, velocity, temperature
9.2.3 enum BLEND_MOTION_STATUS

© 2026 Tesollo Inc. All rights reserved

9.3.2 struct RecipeBlendData

Recipe Setup Structures for Using Blend Motion

Data Types	Name	Size (byte)	Description
int	recipePoseNumber	4	Recipe Pose Number
int	recipeGainNumber	4	Recipe Pain Number
int	blendWaitTime	4	Setting the wait time after an action completes
int	number	4	Blend action unique number

9.3.3 struct GripperSystemSetting

Struct for setting up gripper communication connections

Data Types	Name	Size (byte)	Description
char	comport	32	Setting comfort for RS485 connections
char	ip	32	Setting the IP for TCP/IP connections
int	port	4	Setting the PORT for TCP/IP connections
int	readTimeout	4	Setting the communication timeout time
int	controlMode	4	Setting the Gripper Control Mode
int	communicationMode	4	Setting the Gripper Communication Mode
int	slaveID	4	Setting the Slave ID for a Modbus connection
int	baudrate	4	Setting the baudrate for RS485 connections

© 2026 Tesollo Inc. All rights reserved

9.3.4 struct GripperSetting

Struct for basic gripper operation settings

Data Types	Name	Size (byte)	Description
float	jointOffset	80	Setting the Position Offset for Gripper Joints
float	jointInpose	80	Setting the angle to check the reach of a gripper joint
float	tcpInpose	20	Setting the distance to determine the reach of a fingertip location
float	orientationInpose	20	Setting the angle for fingertip reach checks
int	receivedDataType	24	Set the type of data you want to receive
float	movingInpose	4	Setting angles to determine movement of gripper joints
int	jointCount	4	Number of joints in the gripper model currently in use
int	fingerCount	4	The number of fingers of the gripper model currently in use
DG_MODEL	model	4	Set the gripper model you are currently using
int8_t	dutyByteLength	1	Incoming data size for the current model

Note

Data type x MAX_JOINT_COUNT= 80 byte. jointOffset, jointInpose
Data type x MAX_FINGER_COUNT = 20 byte. tcpInpose, orientationInpose
Data type x MAX_RECEIVED_DATA_TYPE_COUNT = 24 byte. receivedDataType
9.2.2 enum DG_MODEL

© 2026 Tesollo Inc. All rights reserved

9.3.5 struct RecipePoseData

Struct for recipe pose data setting

Data Types	Name	Size (byte)	Description
float	targetJoint	80	Set Target Joint Position Value
int	jointMotionTime	80	Set the time it takes to reach your target location
int	number	4	Recipe Pose Unique Number
int	mode	4	Set position control mode (disabled)

Note

Data type x MAX_JOINT_COUNT= 80 byte. targetJoint, jointMotionTime

9.3.6 struct RecipeGainData

Struct for recipe gain data settings

Data Types	Name	Size (byte)	Description
float	gainP	80	Setting the P Gain
float	gainD	80	Setting the D Gain
float	gainI	80	Setting up I Gain
float	iLimit	80	Setting the Maximum Error Integral
int	controlPIDMode	4	Setting the PID Control Mode
int	number	4	Recipe gain unique number
int	mode	4	Set position control mode (disabled)

Note

Data type x MAX_JOINT_COUNT= 80 byte. gainP, gainD, gainI, iLimit
9.2.6 enum GAIN_MODE

© 2026 Tesollo Inc. All rights reserved

9.3.7 struct RecipeGraspData

Struct for recipe grasp data settings

Data Types	Name	Size (byte)	Description
float	graspForce	4	Set grasp strength
int	positionMode	80	Locking joints in position control instead of grasp mode
int	graspMode	4	Setting the grasp mode
int	graspOption	4	Setting the grasp option (To be added later)
int	smoothGrasping	4	Smooth grasp mode (To be added later)
int	number	4	Recipe grasp unique number

Note

Data type x MAX_JOINT_COUNT= 80 byte. positionMode
9.2.9 enum DG_GRASP_MODE

9.3.8 struct ReceivedFingertipSensorData

Struct for viewing gripper fingertip sensor data

Data Types	Name	Size (byte)	Description
int	sensorType	4	Attached sensor type
int	attachedFinger	20	Array of Finger Positions with attached sensors
float	forceTorque	120	Force/Torque information from the gripper fingertip sensors
uint16_t	tactile	180	Tactile information from the gripper fingertip sensors

Note

Data type x MAX_FINGER_COUNT = 20 byte. attachedFinger
Data type x MAX_FINGER_COUNT x 6(number of sensor data) = 120 byte. forceTorque
Data type x MAX_FINGER_COUNT x 18(number of sensor data) = 180 byte. tactile

© 2026 Tesollo Inc. All rights reserved

9.3.9 struct ReceivedGPIOData

Struct for checking gripper GPIO data

Data Types	Name	Size (byte)	Description
int	GPIO	16	Gripper GPIO Data (Output 1,2,3 / Input 1)

References

Data type x MAX_GRIPPER_GPIO_SIZE= 16 byte. GPIO

© 2026 Tesollo Inc. All rights reserved

Chapter 10. System functions

10.1 System Settings Functions

10.1.1 SetGripperSystem

Description

This function is used to input the configuration values required to establish communication with the gripper. It must be executed first; if this function is not performed, no other functions can be executed.

Parameter

Data Types	Variables	Description
struct	GripperSystemSetting	9.3.3 struct GripperSystemSetting

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
// For Operator Mode Modbus RTU connection
GripperSystemSetting setting;
setting.communicationMode = COMMUNICATION_MODE_RS485;
setting.controlMode = CONTROL_MODE_OPERATOR;
char port[MAX_COMPORT_NAME_SIZE] = "COM1";      // Connectable comfort input
memcpy(setting.comport, port, MAX_COMPORT_NAME_SIZE);
setting.baudrate = 115200;      // Enter the baudrate set in the gripper
setting.slaveID = 1;      // Enter the Slave ID set on the gripper
DG_RESULT result = SetGripperSystem(setting);

// For Operator Mode Modbus TCP connection
GripperSystemSetting setting;
setting.communicationMode = COMMUNICATION_MODE_ ETHERNET;
setting.controlMode = CONTROL_MODE_OPERATOR;
char ip[MAX_GRIPPER_IP_ADDRESS_SIZE] = "169.254.186.72";      // // Enter the gripper IP address
```

© 2026 Tesollo Inc. All rights reserved

```
        memcpy(setting.ip, ip, MAX_GRIPPER_IP_ADDRESS_SIZE);
        setting.port = 502;       // Enter the Port set in the gripper
        setting.slaveID = 1;      // Enter the Slave ID set on the gripper
        setting.readTimeout= 60000;      // Enter the communication timeout time of the gripper
        DG_RESULT result = SetGripperSystem(setting);

        // For Developer Mode TCP/IP connections
        GripperSystemSetting setting;
        setting.communicationMode = COMMUNICATION_MODE_ETHERNET;
        setting.controlMode = CONTROL_MODE_DEVELOPER;
        char ip[MAX_GRIPPER_IP_ADDRESS_SIZE] = "169.254.186.72";      // // Enter the gripper IP
address
        memcpy(setting.ip, ip, MAX_GRIPPER_IP_ADDRESS_SIZE);
        setting.port = 502;      // Enter the Port set in the gripper
        setting.slaveID = 1;      // Enter the Slave ID set on the gripper
        DG_RESULT result = SetGripperSystem(setting);
```

10.1.2 SetGripperOption

Description

Options to set default values for gripper motion. If this option is not selected, you will not be able to control the gripper.

Parameter

Data Types	Variables	Description
GripperSetting	setting	9.3.4 struct GripperSetting

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
        // When setting up the DG-3F-M
        float offset[MAX_JOINT_COUNT] = { 0 };
        float inpose[MAX_JOINT_COUNT] = { 0 };
        GripperSetting gs;
        offset[0] = 2; // offset joint 1 by 2 degrees
        inpose[2] = 3; // Set the 3rd joint inpose 3 degrees
```

© 2026 Tesollo Inc. All rights reserved

```
gs.model = DG_MODEL_DG_3F_M; // set gripper model
memcpy(gs.jointOffset, offset, sizeof(offset));
memcpy(gs.jointInpose, inpose, sizeof(inpose));
gs.movingInpose = 0.3f;
DG_RESULT result = SetGripperOption(gs);
```

10.1.3 ConnectToGripper

Description

Attempt to establish communication with the gripper. The callback functions included in the 'InitializedCallback' function are described in detail in '10.2 Callbacks'.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
void InitializedCallback()
{
    CallbackForOnConnected(*OnConnected);
    CallbackForOnDisconnected(*OnDisconnected);
    CallbackForOnReceivedGripperData(*RecvGripperData);
    CallbackForOnCommunicationPeriod(*OnCountPeriod);
    CallbackForOnReceivedFingertipSensorData(*RecvSensorData);
    CallbackForOnReceivedGPIOData(*RecvGPIOData);
}
InitializedCallback();
DG_RESULT result = ConnectToGripper();
```

© 2026 Tesollo Inc. All rights reserved

10.1.4 DisconnectToGripper

Description

Attempt to disconnect communication with the gripper.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        DG_RESULT result = DisconnectToGripper();
```

10.1.5 SystemStart

Description

Start the system. You can receive data from the gripper and control it only after performing this function.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        DG_RESULT result = SystemStart();
```

© 2026 Tesollo Inc. All rights reserved

10.1.6 SystemStop

Description

Stop the system. Once this function is performed, data from the gripper will no longer be received, and control will not be possible.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
DG_RESULT result = SystemStop();
```

10.1.7 SetIp

Description

Set the gripper IP and Port.

Parameter

Data Types	Variables	Description
int	ip	Enter the IP to change
int	port	Enter the port to change

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int ip[4] = { 169, 254, 186, 72 }; // Enter an IP address
int port = 10000; // Enter port
DG_RESULT result = SetIp(ip, port);
```

© 2026 Tesollo Inc. All rights reserved

10.1.8 SetJointEncoderZero

Description

Set current joint position as the zero reference.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
DG_RESULT result = SetJointEncoderZero();
```

© 2026 Tesollo Inc. All rights reserved

10.2 Callbacks

10.2.1 CallbackForOnConnected

Description

Receive events about communication connections with the gripper.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
	static void OnConnected()
	{
		std::cout << "Connected gripper server" << std::endl;
	}
	DG_RESULT result = CallbackForOnConnected(*OnConnected);
```

10.2.2 CallbackForOnDisconnected

Description

Receive an event when communication with the gripper is lost.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
	static void OnDisconnected()
	{
		std::cout << "Disconnected gripper server" << std::endl;
	}
	DG_RESULT result = CallbackForOnDisconnected(*OnDisconnected);
```

© 2026 Tesollo Inc. All rights reserved

10.2.3 CallbackForOnReceivedGripperData

Description

Receive data from the gripper.

Parameter

Data Types	Variables	Description
struct	ReceivedGripperData	9.3.1 struct ReceivedGripperData

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedGripperData gripperData;
static void RecvGripperData(const ReceivedGripperData data)
{
    gripperData = data;
}
DG_RESULT result = CallbackForOnReceivedGripperData(*RecvGripperData);
```

© 2026 Tesollo Inc. All rights reserved

10.2.4 CallbackForOnCommunicationPeriod

Description

Receive the communication period with the gripper. This value represents the number of times the gripper and PC communicate back and forth in one second. Depending on the specifications of the controller associated with the gripper, this value may be lower, and if it is, smooth control may not occur.

Parameter

Data Types	Variables	Description
int	period	Communication period

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int countPeriod = 0;
static void OnCountPeriod(const int period)
{
    countPeriod = period;
}
DG_RESULT result = CallbackForOnCommunicationPeriod(*OnCountPeriod);
```

© 2026 Tesollo Inc. All rights reserved

10.2.5 CallbackForOnReceivedFingertipSensorData

Description

Receive gripper fingertip sensor data.

Parameter

Data Types	Variables	Description
struct	ReceivedFingertipSensorData	Fingertip sensor data

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedFingertipSensorData sensorData;
static void RecvSensorData(const ReceivedFingertipSensorData recvSensorData)
{
    sensorData = recvSensorData;
}
DG_RESULT result = CallbackForOnReceivedFingertipSensorData(*RecvSensorData);
```

© 2026 Tesollo Inc. All rights reserved

10.2.6 CallbackForOnReceivedGPIOData

Description

Receive gripper GPIO data.

Parameter

Data Types	Variables	Description
struct	ReceivedGPIOData	GPIO data

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedGPIOData gpioData;
static void RecvGPIOData(const ReceivedGPIOData recvGPIOData)
{
    gpioData = recvGPIOData;
}
DG_RESULT result = CallbackForOnReceivedGPIOData(*RecvGPIOData);
```

© 2026 Tesollo Inc. All rights reserved

10.3 Other system functions

10.3.1 SetLowPassFilterAlpha

Description

Sets the low-pass filter gain. The lower the set gain value, the smoother the motion.

Parameter

Data Types	Variables	Description
int	isUsed	Enable filters
float	alpha	Set filter gain values

Returns

Value	Description
DG_RESULT_NONE	Normal
DG_RESULT_SYSTEM_SETTING_NOT_PERFORMED	System setup is incomplete
DG_RESULT_DATA_IS_NOT_BOOLEAN	Only 0 and 1 can be entered
DG_RESULT_DATA_IS_NEGATIVE	Negative values not allowed

Example

```
C/C++
        float alpha = 0.1; // Set the gain value
        int isUsed = 1; // filter On/Off
        DG_RESULT result = SetLowPassFilterAlpha(isUsed, alpha);
```

10.3.2 SetBootMode

Description

This is the boot mode entry function for firmware download. A password is required to enter boot mode. Please contact support@tesollo.com for the password.

Parameter

Data Types	Variables	Description
char*	password	Entering the boot mode will be possible once the correct password is entered

© 2026 Tesollo Inc. All rights reserved

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        DG_RESULT result = SetBootMode(password);
```

10.3.3 SetGPIOOutput

Description

Set the gripper GPIO output.

Parameter

Data Types	Variables	Description
int	gpio	Set output values
int	outputNumber	Set the output number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        int gpio = 1; // output true output
        int outputNumber= 1; // set output number
        DG_RESULT result = SetGPIOOutput(gpio, outputNumber);
```

© 2026 Tesollo Inc. All rights reserved

10.3.4 SetGPIOOutputAll

Description

Sets the gripper global GPIO output.

Parameter

Data Types	Variables	Description
int*	gpio	Set output values

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int gpio[3] = { 1, 1, 1 }; // output true output
DG_RESULT result = SetGPIOOutputAll(gpio);
```

10.3.5 SetTorqueLimitMode

Description

Set the torque limit mode for the gripper.

Parameter

Data Types	Variables	Description
int	isOn	Setting the Torque Limit Mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

© 2026 Tesollo Inc. All rights reserved

Example

```
C/C++
        int isOn = 0; // turn off torque-limited mode
        DG_RESULT result = SetTorqueLimitMode(isOn);
```

Note

*** Caution**

Using 'Torque Limit Mode' off in Operator Mode or using stall torque in Developer Mode may cause a rapid increase in product temperature and potential product damage. Please note that repairs (A/S) may be difficult in case of failure.

10.3.6 SetBootRecipe

Description

Set the recipe to start upon gripper booting.

Parameter

Data Types	Variables	Description
int	recipePoseNumber	Set the pose recipe number
int	recipeGainNumber	Set the gain recipe number
int	recipeGraspNumber	Set the grasp recipe number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        DG_RESULT result = SetBootRecipe(1,1,1); // 1Set the Recipe 1 pose, Recipe 1 gain, and Recipe 1 grasp number as the recipe to start upon booting
```

© 2026 Tesollo Inc. All rights reserved

10.3.7 EEPROMWrite

Description

Save the currently set values to EEPROM. In Operator Mode, they are saved to the gripper's Operator Modeler storage, while in Developer Mode, they are saved to the PC storage.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        DG_RESULT result = EEPROMWrite(); //Save the currently set values to EEPROM
```

10.3.8 BackupRecipeData

Description

This is a function that allows you to back up the data applied to the gripper and save it as a file on the PC

Parameter

Data Types	Variables	Description
char*	path	Backup file path

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        char path[] = "C:\\data\\recipe_backup";
        DG_RESULT result = BackupRecipeData(path);
```

© 2026 Tesollo Inc. All rights reserved

10.3.9 RestoreRecipeData

Description

This feature is only supported in Operator Mode. It reads data from a backup file at the specified path and transfers it to the connected gripper.

Parameter

Data Types	Variables	Description
char*	path	Backup file path

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
char path[] = "C:\\data\\recipe_restore";
DG_RESULT result = RestoreRecipeData(path);
```

© 2026 Tesollo Inc. All rights reserved

10.4 Data Retrieval

10.4.1 GetReceivedGripperData

Description

Retrieves data that has been received from the gripper.

Parameter

Data Types	Variables	Description
struct*	receivedGripperData	Retrives data from the received gripper

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedGripperData receivedGripperData;
DG_RESULT result = GetReceivedGripperData(&receivedGripperData);
```

Note

10.2.3 CallbackForOnReceivedGripperData

© 2026 Tesollo Inc. All rights reserved

10.4.2 GetCommunicationPeriod

Description

Retrieves the communication period with the gripper.

Parameter

Data Types	Variables	Description
int*	period	Retrives data from the communicate period

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
  int period;
  DG_RESULT result = GetCommunicationPeriod(&period);
```

Note

10.2.4 CallbackForOnCommunicationPeriod

© 2026 Tesollo Inc. All rights reserved

10.4.3 GetReceivedFingertipSensorData

Description

Retrieves force/torque sensor data the finger tips.

Parameter

Data Types	Variables	Description
struct*	receivedFingertipData	Retrives data from the received fingertip sensor

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedFingertipSensorData receivedFingertipData;
DG_RESULT result = GetReceivedFingertipSensorData(&receivedFingertipData);
```

Note

10.2.5 CallbackForOnReceivedFingertipSensorData

© 2026 Tesollo Inc. All rights reserved

10.4.4 GetReceivedGPIOData

Description

Retrieves data from GPIO

Parameter

Data Types	Variables	Description
struct*	receivedGPIOData	Retrives data from the received GPIO

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ReceivedGPIOData receivedGPIOData;
DG_RESULT result = GetReceivedGPIOData(&receivedGPIOData);
```

Note

10.2.6 CallbackForOnReceivedGPIOData

© 2026 Tesollo Inc. All rights reserved

10.4.5 GetLibraryVersion

Description

Retrieves the version of the currently used library.

Parameter

Data Types	Variables	Description
int*	version	Array to store copied version information

Example

C/C++

```
int version[3] = {0,0,0}; // Library version: Save in the order of major, minor, and patch.
GetLibraryVersion(version);
std::cout << "Received Library Version : major : " << version[0]
          << ", minor : " << version[1]
          << ", patch : " << version[2]
          << std::endl;
```

© 2026 Tesollo Inc. All rights reserved

Chapter 11. Motion Setting Functions

11.1 Grasp Setup Functions

Sets the grasp algorithm based on force control. It provides grasp modes including five-fingered, three-fingered, two-fingered, parallel, and enveloping, and allows the use of grasp options, position control modes, and more to perform grasp actions in various ways.

11.1.1 SetGraspData

Description

To perform a grasp, you set up the grasp data.

Parameter

Data Types	Variables	Description
int	graspMode	Setting the grasp mode
float	graspForce	Set grasp strength
int	graspOption	Set grasp options
int	smoothGrasping	Using soft grasps

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_ON 1
DG_GRASP_MODE graspMode = DG_GRASP_MODE_3F_3FINGER; // three-finger focus grasp mode
float graspForce = 3.0;
int graspOption = DG_GRASP_OPTION_NONE; // no grasp option
int smoothGraspping = IS_ON; // use smooth grasping
DG_RESULT result = SetGraspData(graspMode, graspForce, graspOption, smoothGrasping);
```

© 2026 Tesollo Inc. All rights reserved

Note

13.2 StartGraspMotion

11.1.2 SetGraspForce

Description

Change grasp strength. This function allows you to change the grasp strength of any grasp without setting anything else.

Parameter

Data Types	Variables	Description
float	graspForce	Set grasp strength

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
    float graspForce = 5.0;
    DG_RESULT result = SetGraspForce(graspForce);
```

Note

13.2 StartGraspMotion

© 2026 Tesollo Inc. All rights reserved

11.2 Setting Gain P

Set the P gain of the gripper. Higher P gain settings allow you to reach the target point faster, but may cause the gripper to jitter.

11.2.1 SetJointGainP

Description

Set the P gain value for a single joint.

Parameter

Data Types	Variables	Description
float	gainP	Enter P Gain
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP = 2.0;
int jointNumber = 1;
DG_RESULT result = SetJointGainP(gainP, jointNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.2.2 SetJointGainPFinger

Description

Set the P gain value for a single finger's joint.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP[MAX_FINGER_JOINT_COUNT] = { 3.0, 3.0, 3.0, 3.0, 3.0 };
int fingerNumber = 3;
DG_RESULT result = SetJointGainPFinger(gainP, fingerNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.2.3 SetJointGainPAll

Description

Set the P gain value for all joints.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP[MAX_JOINT_COUNT] = { 0 };
gainP[0] = gainP[1] = 1; // set 1st and 2nd joint gain
gainP[4] = gainP[6] = 2; // Set the 5th and 7th joint gains
DG_RESULT result = SetJointGainPAll(gainP);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

11.2.4 SetJointGainPBase

Description

Set the P gain value of the base joint. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

© 2026 Tesollo Inc. All rights reserved

Example

C/C++

```
float gainP[MAX_BASE_JOINT_COUNT] = { 0 };
gainP[0] = 1; // Set the base joint gain for joint 17, corresponding to finger 1.
gainP[1] = 2; // Set the base joint gain for joint 18, corresponding to finger 4.
DG_RESULT result = SetJointGainPBase(gainP);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.3 Setting Gain D

Set the D gain of the gripper. D gain can prevent overshoot caused by P gain and can increase the stability of the system. However, high settings can cause the gripper to jitter.

11.3.1 SetJointGainD

Description

Set the D gain value for a single joint.

Parameter

Data Types	Variables	Description
float	gainD	Enter the D gain
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainD = 2.0;
int jointNumber = 1;
DG_RESULT result = SetJointGainD(gainD, jointNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.3.2 SetJointGainDFinger

Description

Set the D gain value for one finger.

Parameter

Data Types	Variables	Description
float*	gainD	Enter the D gain
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainD[MAX_FINGER_JOINT_COUNT] = { 3.0, 3.0, 3.0, 3.0, 3.0 };
int fingerNumber = 3;
DG_RESULT result = SetJointGainDFinger(gainD, fingerNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.3.3 SetJointGainDAll

Description

Set the D gain value for all joints.

Parameter

Data Types	Variables	Description
float*	gainD	Enter the D gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainD[MAX_JOINT_COUNT] = { 0 };
gainD[0] = gainD[1] = 1; // set 1st and 2nd joint gain
gainD[4] = gainD[6] = 2; // Set 5th and 7th joint gains
DG_RESULT result = SetJointGainDAll(gainD);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

11.3.4 SetJointGainDBase

Description

Set the D gain value of the base joint. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
float*	gainD	Enter the D gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

© 2026 Tesollo Inc. All rights reserved

Example

C/C++

```
        float gainD[MAX_BASE_JOINT_COUNT] = { 0 };
        gainD[0] = 1; // Set the base joint gain for joint 17, corresponding to finger 1.
        gainD[1] = 2; // Set the base joint gain for joint 18, corresponding to finger 4.
DG_RESULT result = SetJointGainDBase(gainD);
```

Note

12.2 Recipe Gain Functions

13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.4 Setting Gain I

Set the I gain and maximum error integral of the gripper. The I gain is the gain you set to reduce error and make it sensitive to small changes to reach the target value precisely. A high value of I gain reacts quickly to error but can cause the gripper to jitter, while a low value slows down error compensation and delays recovery of the error integral value, which can cause vibration. The maximum error integral value serves to prevent the error integral value from accumulating rapidly; too low a value may not compensate for the error sufficiently, and too high a value may slow the recovery of the error integral value.

The I gain continues to accumulate the control signal until the error reaches zero. Therefore, in situations where the gripper cannot recover from the error on its own, failure due to joint overload can occur. Therefore, precise tuning to the proper value is required for stable control and care must be taken when using it. It is recommended to tune the I gain value and the maximum error integral in 0.01 increments.

11.4.1 SetJointGainI

Description

Set the I gain value and maximum error integral for a single joint.

Parameter

Data Types	Variables	Description
float	gainI	Enter the I gain
float	iLimit	Setting the Maximum Error Integral
int	jointNumber	Enter the joint number

© 2026 Tesollo Inc. All rights reserved

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
    float gainI = 0.015;
    float iLimit = 0.06;
    int jointNumber = 1;
    DG_RESULT result = SetJointGainI (gainI, iLimit, jointNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

11.4.2 SetJointGainIFinger

Description

Set the I gain value and maximum error integral for a single finger.

Parameter

Data Types	Variables	Description
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
    float gainI[MAX_FINGER_JOINT_COUNT] = { 0.015, 0.015, 0.015, 0.015 };
    float iLimit[MAX_FINGER_JOINT_COUNT] = { 0.06, 0.06, 0.06, 0.06 };
    int fingerNumber = 1;
```

© 2026 Tesollo Inc. All rights reserved

```
DG_RESULT result = SetJointGainIFinger(gainI, iLimit, fingerNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

11.4.3 SetJointGainIAll

Description

Set the I gain value and maximum error integral for all joints.

Parameter

Data Types	Variables	Description
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainI[MAX_JOINT_COUNT] = { 0 };
float iLimit[MAX_JOINT_COUNT] = { 0 };
gainI[0] = gainI[1] = 0.015; // Set the 1st and 2nd joint gains
iLimit[0] = iLimit[1] = 0.06; // Set the 1st and 2nd joint maximum error integrals
DG_RESULT result = SetJointGainIAll(gainI, iLimit);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.4.4 SetJointGainIBase

Description

Set the I gain value and maximum error integral of the base joint. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainI[MAX_BASE_JOINT_COUNT] = { 0 };
float iLimit[MAX_BASE_JOINT_COUNT] = { 0 };
gainI[0] = gainI [1] = 0.015; // Set the base joint gain for joint 17,18.
iLimit [0] = iLimit[1] = 0.06; // Set the base joint the maximum error integral for joint 17, 18.
  DG_RESULT result = SetJointGainIBase (gainI, iLimit);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5 Setting the Gain PID

11.5.1 SetControlPIDMode

Description

Set the gripper controller. This function can set the gripper controller to PD mode and PID mode.

Parameter

Data Types	Variables	Description
int	mode	Setting the Gain Mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int mode = GAIN_MODE_PD; // set to PD gain mode
DG_RESULT result = SetControlPIDMode(mode);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5.2 SetJointGainPID

Description

Set the P, I, D gains and maximum error integral for a single joint.

Parameter

Data Types	Variables	Description
float	gainP	Enter P Gain
float	gainD	Enter the D gain
float	gainI	Enter the I gain
float	iLimit	Setting the Maximum Error Integral
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP = 2.0;
float gainD = 1.0;
float gainI = 0.015;
float iLimit = 0.06;
int jointNumber = 1;
DG_RESULT result = SetJointGainPID (gainP, gainD, gainI, iLimit, jointNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5.3 SetJointGainPIDFinger

Description

Set the P, I, D gains and maximum error integrals for a single finger.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain
float*	gainD	Enter the D gain
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP[MAX_FINGER_JOINT_COUNT] = { 3.0, 3.0, 3.0, 3.0, 3.0 };
float gainD[MAX_FINGER_JOINT_COUNT] = { 1.0, 1.0, 1.0, 1.0, 1.0 };
float gainI[MAX_FINGER_JOINT_COUNT] = { 0.015, 0.015, 0.015, 0.015 };
float iLimit[MAX_FINGER_JOINT_COUNT] = { 0.06, 0.06, 0.06, 0.06 };
int fingerNumber = 1;
DG_RESULT result = SetJointGainPIDFinger(gainP, gainD, gainI, iLimit, fingerNumber);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5.4 SetJointGainPIDAll

Description

Set the P, I, D gains and maximum error integrals for all joints.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain
float*	gainD	Enter the D gain
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP[MAX_JOINT_COUNT] = { 0 };
float gainD[MAX_JOINT_COUNT] = { 0 };
float gainI[MAX_JOINT_COUNT] = { 0 };
float iLimit[MAX_JOINT_COUNT] = { 0 };
gainP[0] = gainP[1] = 3; // Set 1st and 2nd joint gains
gainD[0] = gainD[1] = 1; // Set the 1st and 2nd joint gains
gainI[0] = gainI[1] = 0.015; // Set the 1st and 2nd joint gains
iLimit[0] = iLimit[1] = 0.06; // Set the 1st and 2nd joint maximum error integrals
DG_RESULT result = SetJointGainPIDAll(gainP, gainD, gainI, iLimit);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5.5 SetJointGainPIDAllEqual

Description

Set the P, I, D gains and maximum error integral for all joints in bulk.

Parameter

Data Types	Variables	Description
float	gainP	Enter P Gain
float	gainD	Enter the D gain
float	gainI	Enter the I gain
float	iLimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP = 2.0;
float gainD = 1.0;
float gainI = 0.015;
float iLimit = 0.06;
int jointNumber = 1;
DG_RESULT result = SetJointGainPIDAllEqual(gainP, gainD, gainI, iLimit);
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.5.6 SetJointGainPIDBase

Description

Set the P, I, D gains and maximum error integrals for base joints. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
float*	gainP	Enter P Gain
float*	gainD	Enter the D gain
float*	gainI	Enter the I gain
float*	iLimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP[MAX_BASE_JOINT_COUNT] = { 0 };
float gainD[MAX_BASE_JOINT_COUNT] = { 0 };
float gainI[MAX_BASE_JOINT_COUNT] = { 0 };
float iLimit[MAX_BASE_JOINT_COUNT] = { 0 };
gainP[0] = gainP[1] = 1; // Set the base joint gain for joint 17,18.
gainD[0] = gainD[1] = 5; // Set the base joint gain for joint 17,18.
gainI[0] = gainI[1] = 0.015; // Set the base joint gain for joint 17,18.
iLimit [0] = iLimit [1] = 0.06; // Set the base joint maximum error integrals
DG_RESULT result = SetJointGainPIDBase (gainP, gainD, gainI, iLimit)
```

Note

12.2 Recipe Gain Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.6 Setting the motion time

Sets the time for the gripper to reach the target position. A shorter motion time results in faster movement, while a longer motion time results in slower movement. If the motion time is set very low, the gripper operates at full speed. This feature applies only to position control. The behavior of the motion time may vary depending on the controller specifications. The SDK uses an internal timer to manage the motion time, but its accuracy cannot be guaranteed if the controller environment is not optimal. Additionally, movement during the motion time depends on the gripper settings, so very low motion times may not be reliable.

11.6.1 SetMotionTimeJoint

Description

Set the time to move the position for a single joint.

Parameter

Data Types	Variables	Description
int	motionTime	Set the time to move the location (in ms)
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int motionTime = 1000;
int jointNumber = 1;
DG_RESULT result = SetMotionTimeJoint(motionTime, jointNumber);
```

Note

12.1 Recipe Pose Functions

© 2026 Tesollo Inc. All rights reserved

13.3 Position control

11.6.2 SetMotionTimeFinger

Description

Set the time to move the position of a single finger.

Parameter

Data Types	Variables	Description
int*	motionTime	Setting the time to travel to a location
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int motionTime[MAX_FINGER_JOINT_COUNT] = { 0 }
int fingerNumber = 1;
motionTime[0] = 1000; // Set finger #1 to 1 second of positional movement time
motionTime[2] = 1500; // Set finger #3 to 1.5 seconds of positional movement time
DG_RESULT result = SetMotionTimeFinger(motionTime, fingerNumber);
```

Note

12.1 Recipe Pose Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.6.3 SetMotionTimeAll

Description

Set the motion time for each joint individually.

Parameter

Data Types	Variables	Description
int*	motionTime	Setting the time to travel to a location

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int motionTime[MAX_JOINT_COUNT] = { 0 };
motionTime[0] = 1000; // Set joint #1 to 1 second of positional movement time
motionTime[2] = 1500; // Set joint #3 to 1.5 seconds of positional movement time
DG_RESULT result = SetMotionTimeAll(motionTime);
```

Note

12.1 Recipe Pose Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.6.4 SetMotionTimeAllEqual

Description

Set the motion time for all joints at once.

Parameter

Data Types	Variables	Description
int	motionTime	Setting the time to travel to a location

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int motionTime = 1500; // set the position movement time to 1.5 seconds
DG_RESULT result = SetMotionTimeAllEqual(motionTime);
```

Note

12.1 Recipe Pose Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.6.5 SetMotionTimeBase

Description

Sets the movement time for each base joint individually. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
int*	motionTime	Setting the time to travel to a location

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int motionTime[MAX_BASE_JOINT_COUNT] = { 0 };
motionTime[0] = 1000; // Set joint #17 to 1 second of positional movement time
motionTime[1] = 1500; // Set joint #18 to 1.5 second of positional movement time
DG_RESULT result = SetMotionTimeBase(motionTime);
```

Note

12.1 Recipe Pose Functions
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

11.7 Setting the Position Control Mode

Position control mode is applied during grasp actions. Joints set to position control mode perform position control instead of grasping during grasp actions. This is when you do not want certain joints to move during a grasp, and can be used to implement different grasping methods.

11.7.1 SetPositionModeJoint

Description

Set the position control mode for a single joint.

Parameter

Data Types	Variables	Description
int	positionMode	Enable position control mode
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_ON 1
int positionMode = IS_ON;
int jointNumber = 1;
// Set position control only for joint #1
DG_RESULT result = SetPositionModeJoint(positionMode, jointNumber);
```

Note

12.3 Recipe Grasp Functions
13.2 StartGraspMotion

© 2026 Tesollo Inc. All rights reserved

11.7.2 SetPositionModeFinger

Description

Set the position control mode for all joint of a single finger.

Parameter

Data Types	Variables	Description
int*	positionMode	Enable position control mode
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_OFF 0
#define IS_ON 1
int positionMode[MAX_FINGER_JOINT_COUNT] = { IS_ON, IS_ON, IS_OFF, IS_OFF };
int fingerNumber = 1;
// Set position control only for the 1st finger 1st and 2nd joints
// 1st finger 3rd and 4th joints position control mode release
DG_RESULT result = SetPositionModeFinger(positionMode, fingerNumber);
```

Note

12.3 Recipe Grasp Functions
13.2 StartGraspMotion

© 2026 Tesollo Inc. All rights reserved

11.7.3 SetPositionModeAll

Description

Sets the position control mode for all joints.

Parameter

Data Types	Variables	Description
int*	positionMode	Enable position control mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_OFF 0
#define IS_ON 1
int positionMode[MAX_JOINT_COUNT] = { IS_OFF };
positionMode[0] = IS_ON; // Set position control only for joint #1
positionMode[11] = IS_OFF; // joint #12 position control mode release
DG_RESULT result = SetPositionModeAll(positionMode);
```

Note

12.3 Recipe Grasp Functions
13.2 StartGraspMotion

© 2026 Tesollo Inc. All rights reserved

11.7.4 SetPositionModeBase

Description

Sets the position control mode for base joints. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
int*	positionMode	Enable position control mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_OFF 0
#define IS_ON 1
int positionMode[MAX_BASE_JOINT_COUNT] = { IS_OFF };
positionMode[0] = IS_ON; // Set the position control mode ON for joint 17, corresponding to finger 1
positionMode[1] = IS_OFF; // Set the position control mode OFF for joint 18, corresponding to finger 4.
DG_RESULT result = SetPositionModeBase(positionMode);
```

Note

12.3 Recipe Grasp Functions
13.2 StartGraspMotion

© 2026 Tesollo Inc. All rights reserved

11.8 Setting Fingertip Sensor

This setting is related to the sensor data attached to the gripper fingertips. This feature is available when using the optional fingertip sensor.

11.8.1 SetFingerTipDataZero

Description

Resets the current fingertip data to zero. Calling the function again restores the original raw data.

Return

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
DG_RESULT result = SetFingerTipDataZero();
```

Note

10.1.2 SetGripperOption

© 2026 Tesollo Inc. All rights reserved

11.9 Setting the Current I Gain

This setting is related to the current I gain used in the gripper's current control.

11.9.1 SetJointCurrentGainI

Description

Sets the current I gain for a single joint.

Parameter

Data Types	Variables	Description
float	currentGain	Enter the current I gain
float	currentILimit	Setting the Maximum Error Integral
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainI = 0.005;
float currentILimit = 1000.0;
int jointNumber = 1;
DG_RESULT result = SetJointCurrentGainI(currentGainI, currentILimit, jointNumber);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.9.2 SetJointCurrentGainIFinger

Description

Sets the current I gains for all joints of a single finger.

Parameter

Data Types	Variables	Description
float*	currentGainI	Enter the current I gain
float*	currentILimit	Setting the Maximum Error Integral
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainI[MAX_FINGER_JOINT_COUNT] = { 0.005, 0.005, 0.005, 0.005 };
float currentILimit[MAX_FINGER_JOINT_COUNT] = { 1000.0, 1000.0, 1000.0, 1000.0 };
int fingerNumber = 1;
DG_RESULT result = SetJointCurrentGainIFinger(currentGainI, currentILimit, fingerNumber);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.9.3 SetJointCurrentGainIAll

Description

Sets the current I gains for all joints.

Parameter

Data Types	Variables	Description
float*	currentGainI	Enter the current I gain
float*	currentILimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainI[MAX_JOINT_COUNT] = { 0 };
float currentILimit[MAX_JOINT_COUNT] = { 0 };
currentGainI[0] = currentGainI[1] = 0.005; // Set the current gains for the first and second joints
currentILimit[0] = currentILimit[1] = 1000.0; // Set the maximum error integral for the first and second joints
DG_RESULT result = SetJointCurrentGainIAll(currentGainI, currentILimit);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.9.4 SetJointCurrentGainIBase

Description

Set the I gain and maximum error integral of the base joint same current gain for all joints.

Parameter

Data Types	Variables	Description
float*	currentGainI	Enter the current I gain
float*	currentILimit	Setting the Maximum Error Integral

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainI[MAX_BASE_JOINT_COUNT]= { 0.005, 0.005 };
float currentILimit[MAX_BASE_JOINT_COUNT] = { 1000.0, 1000.0 };
DG_RESULT result = SetJointCurrentGainIBase(currentGainI, currentILimit);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.10 Setting the Current P Gain

This setting is related to the current P gain used in the gripper's current control.

11.10.1 SetJointCurrentGainP

Description

Sets the current P gain for a single joint.

Parameter

Data Types	Variables	Description
float	currentGain	Enter the current P gain
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainP = 0.005;
int jointNumber = 1;
DG_RESULT result = SetJointCurrentGainP(currentGainP, jointNumber);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.10.2 SetJointCurrentGainPFinger

Description

Sets the current P gains for all joints of a single finger.

Parameter

Data Types	Variables	Description
float*	currentGainP	Enter the current P gain
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainP[MAX_FINGER_JOINT_COUNT] = { 0.005, 0.005, 0.005, 0.005 };
int fingerNumber = 1;
DG_RESULT result = SetJointCurrentGainPFinger(currentGainI, fingerNumber);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.10.3 SetJointCurrentGainPAll

Description

Sets the current P gains for all joints.

Parameter

Data Types	Variables	Description
float*	currentGainP	Enter the current P gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainP[MAX_JOINT_COUNT] = { 0 };
currentGainP[0] = currentGainP[1] = 0.005; // Set the current gains for the first and second joints
DG_RESULT result = SetJointCurrentGainPAll(currentGainP);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

11.10.4 SetJointCurrentGainPBase

Description

Set the P gain and maximum error integral of the base joint same current gain for all joints.

Parameter

Data Types	Variables	Description
float*	currentGainP	Enter the current P gain

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float currentGainP[MAX_BASE_JOINT_COUNT]= { 0.005, 0.005 };
DG_RESULT result = SetJointCurrentGainPBase(currentGainP);
```

Note

13.4 Current Control

© 2026 Tesollo Inc. All rights reserved

Chapter 12. Recipe Functions

12.1 Recipe Pose Functions

12.1.1 UpdateRecipeCurrentPoseData

Description

Saves the current gripper joint position and motion time to the specified pose recipe number.

Parameter

Data Types	Variables	Description
int	recipeNumber	Recipe Pose Number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float joint[MAX_JOINT_COUNT] = { 0 }; // Enter the number of gripper joints
int motionTime = 500; // set the time to arrive at the gripper target position
int recipeNumber = 1; // set recipe number
DG_RESULT result = MoveJointAll(joint);
if (result == DG_RESULT_NONE) result = SetMotionTimeAllEqual(motionTime);
if (result == DG_RESULT_NONE) result = UpdateRecipeCurrentPoseData(recipeNumber);
```

Note

11.2.1 SetJointGainP - 11.5.1 SetControlPIDMode
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

12.1.2 UpdateRecipeTargetPoseData

Description

Saves the current gripper joint target position and motion time to the specified pose recipe number.

Parameter

Data Types	Variables	Description
int	recipeNumber	Recipe Pose Number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float joint[MAX_JOINT_COUNT] = { 0 }; // Enter the number of gripper joints
int motionTime = 500; // set the time to arrive at the gripper target position
int recipeNumber = 1; // set recipe number
DG_RESULT result = MoveJointAll(joint);
 if (result != DG_RESULT_NONE)      return result;

result = SetMotionTimeAllEqual(motionTime);
 if (result != DG_RESULT_NONE)      return result;

result = UpdateRecipeTargetPoseData(recipeNumber);
 if (result != DG_RESULT_NONE)      return result;
```

Note

11.2.1 SetJointGainP - 11.5.1 SetControlPIDMode
13.3 Position control

© 2026 Tesollo Inc. All rights reserved

12.1.3 LoadRecipePoseData

Description

This works by reading the pose recipe information for the entered number.

Parameter

Data Types	Variables	Description
int	recipeNumber	Recipe Pose Number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int recipeNumber = 1; // recipe number
DG_RESULT result = LoadRecipePoseData(recipeNumber);
```

Note

12.1.1 UpdateRecipeCurrentPoseData
12.1.2 UpdateRecipeTargetPoseData

© 2026 Tesollo Inc. All rights reserved

12.2 Recipe Gain Functions

12.2.1 UpdateRecipeGainData

Description

Saves the currently set P, I, and D gains, maximum error integral, and PID control mode to the entered number of gain recipes.

Parameter

Data Types	Variables	Description
int	recipeNumber	Recipe gain number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float gainP = 1; // set P gain
float gainD = 5; // set D gain
float gainI = 0.01; // set I gain
float iLimit = 0.02; // Set the maximum value of the error integral
int controlPIDMode = GAIN_MODE_PID; // set PID control mode
int recipeNumber = 1; // set recipe number
DG_RESULT result = SetJointGainPIDAllEqual(gainP, gainD, gainI, iLimit);
  if (result != DG_RESULT_NONE)      return result;

result = SetControlPIDMode(controlPIDMode);
  if (result != DG_RESULT_NONE)      return result;

result = UpdateRecipeGainData(recipeNumber);
  if (result != DG_RESULT_NONE)      return result;
```

Note

12.2.2 LoadRecipeGainData

© 2026 Tesollo Inc. All rights reserved

12.2.2 LoadRecipeGainData

Description

Reads and sets the gain recipe data for the entered number.

Parameter

Data Types	Variables	Description
int	recipeNumber	Recipe gain number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int recipeNumber = 1; // recipe number
DG_RESULT result = LoadRecipeGainData(recipeNumber);
```

Note

12.2.1 UpdateRecipeGainData

© 2026 Tesollo Inc. All rights reserved

12.3 Recipe Grasp Functions

12.3.1 UpdateRecipeGraspData

Description

Saves the currently set Grasp Options, Grasp Mode, Smooth Grasp Mode, Position Control Mode, and Grasp Strength settings to the entered Grasp Recipe.

Parameter

Data Types	Variables	Description
int	graspNumber	Recipe grasp number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float graspForce = 5; // Set the grasp force
int graspOption = 1; // set the grasp option
int smoothGraspping = 1; // set smooth grasping mode
int positionMode[MAX_JOINT_COUNT] = { 0 }; // Enter the number of gripper joints
int recipeNumber = 1; // set recipe number
DG_GRASP_MODE graspMode = DG_GRASP_MODE_3F_3FINGER; // set grasp mode
DG_RESULT result = SetPositionModeAll(positionMode);
 if (result != DG_RESULT_NONE)      return result;
result = SetGraspData(graspMode, graspForce, graspOption, smoothGrasping);

if (result != DG_RESULT_NONE)      return result;
result = UpdateRecipeGraspData(recipeNumber);
```

Note

11.1 Grasp Setup Functions
11.7.1 SetPositionMode
11.7.2 SetPositionModeFinger
11.7.3 SetPositionModeAll

© 2026 Tesollo Inc. All rights reserved

12.3.2 LoadRecipeGraspData

Description

Reads and sets the grasp recipe data for the entered number.

Parameter

Data Types	Variables	Description
int	graspNumber	Recipe grasp number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int recipeNumber = 1; // recipe number
DG_RESULT result = LoadRecipeGraspData(recipeNumber);
 if (result != DG_RESULT_NONE)      return result;

result = StartGraspMotion(1);
 if (result != DG_RESULT_NONE)      return result;
```

Note

12.3.1 UpdateRecipeGraspData

© 2026 Tesollo Inc. All rights reserved

12.4 Recipe Blend Functions

12.4.1 UpdateBlendJoint

Description

Adds the pose recipe, gain recipe, and wait time after completion of the action from the input blend action data to the blend action recipe.

Parameter

Data Types	Variables	Description
struct	blendData	9.3.2 struct RecipeBlendData

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
RecipeBlendData blendData;
blendData.blendWaitTime = 500; // set the wait time after the action completes
blendData.recipeGainNumber = 1; // set gain recipe number
blendData.recipePoseNumber = 1; // set pose recipe number
DG_RESULT result = UpdateBlendJoint(blendData);
```

Note

12.1.1 UpdateRecipeCurrentPoseData
12.1.2 UpdateRecipeTargetPoseData
12.2.1 UpdateRecipeGainData

© 2026 Tesollo Inc. All rights reserved

12.4.2 ClearMoveBlendJoint

Description

Deletes the list of currently added blend data.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        ClearMoveBlendJoint();
```

Note

12.4.1 UpdateBlendJoint
12.4.3 AddMoveBlendJoint
12.4.4 SetMoveBlendJoint
12.4.5 StartMoveBlendJoint
12.4.6 StopMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

12.4.3 AddMoveBlendJoint

Description

Add the updated blend data to the blend list.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ClearMoveBlendJoint();

RecipeBlendData blendData;
blendData.recipePoseNumber = 1;
blendData.recipeGainNumber = 3;
blendData.blendWaitTime = 0;

DG_RESULT result = UpdateBlendJoint(blendData);
if (result == DG_RESULT_NONE) {
        DG_RESULT result = AddMoveBlendJoint();
}
```

Note

12.4.1 UpdateBlendJoint
12.4.2 ClearMoveBlendJoint
12.4.3 AddMoveBlendJoint
12.4.4 SetMoveBlendJoint
12.4.5 StartMoveBlendJoint
12.4.6 StopMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

12.4.4 SetMoveBlendJoint

Description

Saves the list of currently added blends to the Blend Motion list.

Parameter

Data Types	Variables	Description
int	blendNumber	Enter a Blend Motion Number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
ClearMoveBlendJoint();

RecipeBlendData blendData;
blendData.recipePoseNumber = 1;
blendData.recipeGainNumber = 3;
blendData.blendWaitTime = 0;
DG_RESULT result = UpdateBlendJoint(blendData);
if (result != DG_RESULT_NONE)	return result;
result = AddMoveBlendJoint();
if (result != DG_RESULT_NONE)	return result;

blendData.recipePoseNumber = 2;
blendData.recipeGainNumber = 2;
blendData.blendWaitTime = 1000;
DG_RESULT result = UpdateBlendJoint(blendData);
if (result != DG_RESULT_NONE)	return result;
result = AddMoveBlendJoint();
if (result != DG_RESULT_NONE)	return result;

if (result == DG_RESULT_NONE)
{
	int blendNumber = 1;
	result = SetMoveBlendJoint(blendNumber);
}
```

© 2026 Tesollo Inc. All rights reserved

Note

12.4.1 UpdateBlendJoint
12.4.2 ClearMoveBlendJoint
12.4.3 AddMoveBlendJoint
12.4.5 StartMoveBlendJoint
12.4.6 StopMoveBlendJoint

12.4.5 StartMoveBlendJoint

Description

Performs a blend motion with the entered blend number.

Parameter

Data Types	Variables	Description
int	blendNumber	Enter a Blend Motion Number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
C/C++
        int blendNumber = 1;
        DG_RESULT result = StartMoveBlendJoint(blendNumber);
```

Note

12.4.1 UpdateBlendJoint
12.4.2 ClearMoveBlendJoint
12.4.3 AddMoveBlendJoint
12.4.4 SetMoveBlendJoint
12.4.5 StartMoveBlendJoint
12.4.6 StopMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

12.4.6 StopMoveBlendJoint

Description

Stop the blend motion currently being performed.

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
DG_RESULT result = StopMoveBlendJoint();
```

Note

12.4.1 UpdateBlendJoint
12.4.2 ClearMoveBlendJoint
12.4.3 AddMoveBlendJoint
12.4.4 SetMoveBlendJoint
12.4.5 StartMoveBlendJoint

© 2026 Tesollo Inc. All rights reserved

Chapter 13. Motion Functions

13.1 ManualTeachMode

Description

Set the gripper to manual teaching mode. Manual teaching mode allows a human to set the gripper pose and teach all joints directly.

Parameter

Data Types	Variables	Description
int	isOn	Setting manual operation mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
int isOn = 1; // set manual operation mode
ManualTeachMode(isOn);
```

© 2026 Tesollo Inc. All rights reserved

13.2 StartGraspMotion

Description

Perform a grasp motion.

Parameter

Data Types	Variables	Description
int	isGrasp	Setting up grasp motion performance

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
#define IS_ON 1
DG_GRASP_MODE graspMode = DG_GRASP_MODE_3F_3FINGER; // three-finger focus grasp mode
float graspForce = 3.0;
int graspOption = DG_GRASP_OPTION_NONE; // no grasp option
int smoothGraspping = IS_ON; // use smooth grasping
DG_RESULT result = SetGraspData(graspMode, graspForce, graspOption, smoothGrasping);
if (result == DG_RESULT_NONE){
        int isGrasp = IS_ON;
        result = StartGraspMotion(isGrasp);
}
```

Note

11.1.1 SetGraspData
11.7 Setting the Position Control Mode

© 2026 Tesollo Inc. All rights reserved

13.3 Position control

13.3.1 MoveJoint

Description

Performs a repositioning of one joint.

Parameter

Data Types	Variables	Description
float	targetJoint	Enter the target position value (in degrees)
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float targetJoint = 20.9; // set target position 20.9 degrees
int jointNumber = 1;
DG_RESULT result = MoveJoint(targetJoint, jointNumber);
```

© 2026 Tesollo Inc. All rights reserved

13.3.2 MoveJointFinger

Description

Performs a repositioning of a single finger joint.

Parameter

Data Types	Variables	Description
float*	targetJoint	Enter the target position value (in degrees)
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float targetJoint[MAX_FINGER_JOINT_COUNT] = { 0 };
targetJoint[0] = 0;
targetJoint[1] = 0;
targetJoint[2] = 60; // Set target position 60 degrees
targetJoint[3] = 60; // Set target position 60 degrees
int fingerNumber = 1;
DG_RESULT result = MoveJointFinger(targetJoint, fingerNumber);
```

© 2026 Tesollo Inc. All rights reserved

13.3.3 MoveJointAll

Description

Performs a repositioning of all joints.

Parameter

Data Types	Variables	Description
float*	targetJoint	Enter the target position value (in degrees)

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float targetJoint[MAX_JOINT_COUNT] = { 0 };
targetJoint[0] = targetJoint[4] = targetJoint[8] = 0;
targetJoint[1] = targetJoint[5] = targetJoint[9] = 0;
targetJoint[2] = targetJoint[6] = targetJoint[10] = 60; // Set target position 60 degrees
targetJoint[3] = targetJoint[7] = targetJoint[11] = 60; // Set target position 60 degrees
DG_RESULT result = MoveJointAll(targetJoint);
```

© 2026 Tesollo Inc. All rights reserved

13.3.4 MoveServoJoint

Description

Performs real-time position movement for all joints. This function is not affected by the motion time setting and executes movements immediately. It is available only in Developer Mode.

Parameter

Data Types	Variables	Description
float*	targetJoint	Enter the target position value (in degrees)

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float targetJoint[MAX_JOINT_COUNT] = { 0 };
targetJoint[0] = targetJoint[4] = targetJoint[8] = 0;
targetJoint[1] = targetJoint[5] = targetJoint[9] = 0;
targetJoint[2] = targetJoint[6] = targetJoint[10] = 60; // Set target position 60 degrees
targetJoint[3] = targetJoint[7] = targetJoint[11] = 60; // Set target position 60 degrees
DG_RESULT result = MoveServoJoint(targetJoint);
```

© 2026 Tesollo Inc. All rights reserved

13.3.5 MoveJointBase

Description

Moves the base joints to their target positions. This function is specific to the DG-4F-M.

Parameter

Data Types	Variables	Description
float*	targetJoint	Enter the target position value (in degrees)

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
float targetJoint[MAX_BASE_JOINT_COUNT] = { 0 };
targetJoint[0] = 70;
targetJoint[1] = -70;
DG_RESULT result = MoveJointBase(targetJoint);
```

© 2026 Tesollo Inc. All rights reserved

13.4 Current Control

This provides guidance on how to drive each joint using current control. In current control mode, setting the PID gain values is not possible

13.4.1 SetCurrentControlMode

Description

Set the gripper joints to current control mode.

Parameter

Data Types	Variables	Description
int	isOn	Setting the current control mode

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

```
int isOn = 1; // set current control mode
DG_RESULT result = SetCurrentControlMode(isOn);
```

© 2026 Tesollo Inc. All rights reserved

13.4.2 SetTargetCurrentJoint

Description

Set the target current value for a single joint.

Parameter

Data Types	Variables	Description
int	targetCurrent	Enter a target current value (mA)
int	jointNumber	Enter the joint number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int targetCurrent = 50; // target current value 50mA
int jointNumber = 1; // joint number 1
DG_RESULT result = SetTargetCurrentJoint(targetCurrent, jointNumber);
```

© 2026 Tesollo Inc. All rights reserved

13.4.3 SetTargetCurrentFinger

Description

Set the target current value for a single finger joint.

Parameter

Data Types	Variables	Description
int*	targetCurrent	Enter a target current value (mA)
int	fingerNumber	Enter the finger number

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int targetCurrent[4] = { 50,50,50,50 }; // target current value 50mA
int fingerNumber = 1; // finger 1
DG_RESULT result = SetTargetCurrentFinger(targetCurrent, fingerNumber);
```

© 2026 Tesollo Inc. All rights reserved

13.4.4 SetTargetCurrentAll

Description

Set the target current value for all joints.

Data Types	Variables	Description
int*	targetCurrent	Enter a target current value (mA)

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int targetCurrent[MAX_JOINT_COUNT];
for (int i = 0; i < MAX_JOINT_COUNT; i++)
{
        targetCurrent[i] = 50;
}
DG_RESULT result = SetTargetCurrentAll(targetCurrent);
```

© 2026 Tesollo Inc. All rights reserved

13.4.5 SetTargetCurrentBase

Description

Set the target current value for base joints.

Data Types	Variables	Description
int*	targetCurrent	Enter a target current value (mA)

Returns

Number	Value	Description
0	DG_RESULT_NONE	Normal
If it is not zero	ERROR	Appendix 2 Error list

Example

C/C++

```
int targetCurrent[MAX_BASE_JOINT_COUNT];
targetCurrent[0] = 50; // target current value 50mA
targetCurrent[1] = 50; // target current value 50mA
DG_RESULT result = SetTargetCurrentBase(targetCurrent);
```

© 2026 Tesollo Inc. All rights reserved

Appendix 1. Glossary and Description

B

Baudrate

The data transfer rate, where the unit of data transfer is baud.

Blend list

A collection of lists for continuous positional Motions with recipe poses and recipe gains.

Blend motion

A motion with a saved blend list.

Blend move

Actuates a list of blends consisting of recipe poses and recipe gains in a continuous loop in the order they are registered.

C

Cartesian coordinate

A coordinate system that mathematically describes the position of a point in 3D space, consisting of the x-, y-, and z-axes.

Communication Period

Time to send and receive data between devices.

COMPORT

The input and output ports on the computer used for serial communication.

D

Developer Mode

A mode that uses a user-developed controller, which receives information from each joint of the gripper and allows you to control each joint directly.

© 2026 Tesollo Inc. All rights reserved

E

Encoder

A sensor that outputs information about the position and velocity of an object in rotational or linear motion as an electrical signal.

F

Fix tilt option (Fix tilt grasp option)

An option that performs a grasping motion while maintaining the tilt of the 4th joint of the finger based on the gripper's current posture.

I

Inpose

When moving from position control to a target joint position value, the range to verify that the target position value has been reached. This is often used to perform other motions before reaching the target position due to the inpose value.

J

Joint

A drive unit with a 1 degree of freedom rotation axis. For the Delto Gripper series, the DG-3F-M and DG-3F-B have 12 joints, the DG-4F-M has 18 joints, and the DG-5F-R and DG-5F-L have 20 joints.

L

Low Pass Filter

A filter that passes below a certain frequency, reducing noise in the received digital signal and smoothing out abrupt changes to produce a smoother result.

M

Modbus protocol (Modbus protocol)

An industry-standard communication protocol developed for communication between automation devices that uses a master-slave structure and supports a variety of communication methods, including RTU, ASCII, and TCP/IP. Connected devices can be identified through Slave IDs.

Modbus RTU (Modbus RTU)

© 2026 Tesollo Inc. All rights reserved

It is a protocol based on serial communication, such as RS-232 or RS-485, and uses a binary format for efficient and fast communication.

Modbus TCP (Modbus TCP)

A Modbus protocol that operates on top of TCP/IP networks. It transmits data in the form of packets and allows for fast and flexible communication over a network.

Module

Minimum actuation unit divided into independent parts. The Delto Gripper series consists of one independent module for each joint, allowing for independent operation, scalability and ease of maintenance. Each module consists of a motor, encoder, motor drive, reducer, etc.

O

Operator Mode

Mode controlled using a controller inside the product. Mode that supports the Modbus Protocol and allows you to easily control the gripper by simply changing the register map, or by using the UI operations provided.

Overshoot

A temporary exceedance of a target value in a control system as it converges to the target value.

P

Parallel option (Parallel grasp option)

Unlike the tilt-fixed grasp option, this option always grasps with the planes of the fingertips touching, regardless of the gripper's current pose, to the contact area with the object when grasping.

PD control (PD control)

A feedback control system that combines proportional and derivative control. By combining a response proportional to the error with a response based on the rate of change of the error, fast and stable control is possible, but perfect error compensation is difficult.

PID control (PID control)

A feedback control system that combines proportional, integral, and derivative control. It is used in automation and robotics systems to achieve desired target values, allowing for stable and precise control.

Position control

A control method that adjusts the position of a motor or machine part. Moving to a target position based on the current position is called position control, and it can be precisely controlled with PID control.

© 2026 Tesollo Inc. All rights reserved

R

Recipe

A set of pre-saved data with key settings for easy and efficient control of a robot (gripper). Recipes can be saved in advance to facilitate control through stored recipes.

S

Smooth grasping (Smooth grasping mode)

A mode that allows you to grasp objects gently to prevent damage; once contact is made, the grasping force changes to the pre-set level.

Socket communication

A method by which two devices send and receive data over a network. IP addresses, PORT numbers identify the devices to send and receive.

Software Development Kit (SDK)

A collection of tools and libraries for developing software applications. Users can utilize the SDK to easily implement functionality for gripper control.

T

TCP/IP

A core protocol for internet and network communications that allows data to be reliably delivered. LAN ports allow for wired network connectivity and efficient connection of multiple devices.

W

Window Socket API (WSA)

An API that supports socket programming on Windows operating systems, enabling data communication between a client and a server.

© 2026 Tesollo Inc. All rights reserved

Appendix 2. Error list

Value	Name	Description
0	DG_RESULT_NONE	Normal
1	DG_RESULT_SYSTEM_SETTING_ NOT_PERFORMED	System setup is incomplete; perform the system setup function after performing the following
100	DG_RESULT_VALUE_IS_NEGATIVE	The input data is negative.
101	DG_RESULT_OVERFLOW_ JOINT_COUNT	Exceeded maximum motor number. MAX_JOINT_COUNT = 20
102	DG_RESULT_OVERFLOW_ FINGER_COUNT	You have exceeded the maximum number of fingers you can enter. MAX_FINGER_COUNT = 5
103	DG_RESULT_OVERFLOW_TCP_ COUNT	You have exceeded the maximum number of coordinate systems you can enter. CARTESIAN_COORDINATE_POSE_COUNT = 6
105	DG_RESULT_OVERFLOW_ADD_ BLEND_SIZE	The list of poses is maxed out. No more blend poses can be added. MAX_BLEND_ADD_POSE_COUNT = 50
106	DG_RESULT_OVERFLOW_RECIPE_ BLEND_SIZE	You have exceeded the maximum number of blend motions you can enter. MAX_BLEND_COUNT = 10
107	DG_RESULT_BLEND_SIZE_ZERO	There are no saved blend motions.
108	DG_RESULT_NOT_SUPPORTED_MODEL	The model does not support that feature.
109	DG_RESULT_DATA_IS_ZERO	The input data is zero.
110	DG_RESULT_DATA_IS_NOT_BOOLEAN	Only 0 and 1 can be entered.
111	DG_RESULT_NOT_FOUND_MODEL	Model number not found.
112	DG_RESULT_OVERFLOW_RECIPE_ POSE_COUNT	You have exceeded the maximum number of recipe poses that can be entered. MAX_RECIPE_POSE_COUNT = 100
113	DG_RESULT_OVERFLOW_RECIPE_ GAIN_COUNT	You have exceeded the maximum recipe gain number you can enter. MAX_RECIPE_GAIN_COUNT = 20

© 2026 Tesollo Inc. All rights reserved

114	DG_RESULT_OVERFLOW_RECIPE_GRASP_COUNT	You have exceeded the maximum number of recipe grasps that can be entered. MAX_RECIPE_GRASP_COUNT = 20
115	DG_RESULT_INVALID_CONTROL_MODE	Invalid system control mode.
116	DG_RESULT_INVALID_GRASP_MODE_DATA	You have exceeded the maximum number of grasp modes that can be entered. 9.2.6 enum GAIN_MODE
118	DG_RESULT_OVERFLOW_MAX_BYTE_DATA	The maximum number of bytes that can be entered (255) has been exceeded.
119	DG_RESULT_OVERFLOW_CURRENT_LIMIT	The maximum current value that can be entered has been exceeded. M model: MAX_INPUT_CURRENT = 150 S model: MAX_INPUT_CURRENT = 360
120	DG_RESULT_IS_NOT_TORQUE_CONTROL_MODE	Not in torque control mode.
121	DG_RESULT_NOT_SUPPORTED_DATA_TYPE	This is an unsupported data receive command.
122	DG_RESULT_BACKUP_START	Data is being backed up.
123	DG_RESULT_INVALID_PASSWORD	Invalid password entered.
124	DG_RESULT_OVERFLOW_GPIO_COUNT	The maximum allowable GPIO number has been exceeded. MAX_GRIPPER_GPIO_SIZE = 6
125	DG_RESULT_OVERFLOW_GRASP_FORCE	The input value for grip strength has been exceeded. MAX_GRASP_FORCE = 20
126	DG_RESULT_OVERFLOW_BLEND_WAIT_TIME	The input value for blend wait time has been exceeded. MAX_BLEND_WAIT_TIME = 30000
127	DG_RESULT_NOT_FOUND_RESTORE_DATA	No file to restore along the path.
128	DG_RESULT_RESTORE_START	The system is currently in the process of restoration.
200	DG_RESULT_NOT_ARRIVED	The target location was not reached.
201	DG_RESULT_NOT_START_BLEND_MOVE	Not in blend Motion mode.

© 2026 Tesollo Inc. All rights reserved

202	DG_RESULT_ALREADY_ BLEND_MOVE_STATE	Blend is in action.
203	DG_RESULT_ALREADY_TCP_MOVE	TCP location is moving.
204	DG_RESULT_RECIPE_IS_NOT_ JOINT_MODE	The pose recipe is not in Joint Control Mode.
205	DG_RESULT_ACTIVATE_CURRENT_ CONTROL_MODE	The current control mode is active.
206	DG_RESULT_ACTIVATE_ GRASP_MOTION	The gripper is currently performing a grasp.
207	DG_RESULT_ACTIVATE_MANUAL_ CONTROL_MODE	Manual control mode is active.
401	DG_RESULT_ MODULE _FAULT_1	Module 1 fault detected
402	DG_RESULT_ MODULE _FAULT_2	Module 2 fault detected
403	DG_RESULT_ MODULE _FAULT_3	Module 3 fault detected
404	DG_RESULT_ MODULE _FAULT_4	Module 4 fault detected
405	DG_RESULT_ MODULE _FAULT_5	Module 5 fault detected
406	DG_RESULT_ MODULE _FAULT_6	Module 6 fault detected
407	DG_RESULT_ MODULE _FAULT_7	Module 7 fault detected
408	DG_RESULT_ MODULE _FAULT_8	Module 8 fault detected
409	DG_RESULT_ MODULE _FAULT_9	Module 9 fault detected
410	DG_RESULT_ MODULE _FAULT_10	Module 10 fault detected
411	DG_RESULT_ MODULE _FAULT_11	Module 11 fault detected
412	DG_RESULT_ MODULE _FAULT_12	Module 12 fault detected
413	DG_RESULT_ MODULE _FAULT_13	Module 13 fault detected
414	DG_RESULT_ MODULE _FAULT_14	Module 14 fault detected
415	DG_RESULT_ MODULE _FAULT_15	Module 15 fault detected
416	DG_RESULT_ MODULE _FAULT_16	Module 16 fault detected
417	DG_RESULT_ MODULE _FAULT_17	Module 17 fault detected
418	DG_RESULT_ MODULE _FAULT_18	Module 18 fault detected
419	DG_RESULT_ MODULE _FAULT_19	Module 19 fault detected
420	DG_RESULT_ MODULE _FAULT_20	Module 20 fault detected
421	DG_RESULT_ MODULE _FAULT_21	First fingertip module fault detected
422	DG_RESULT_ MODULE _FAULT_22	Second fingertip module fault detected

© 2026 Tesollo Inc. All rights reserved

423	DG_RESULT_ MODULE _FAULT_23	Third fingertip module fault detected
424	DG_RESULT_ MODULE _FAULT_24	Fourth fingertip module fault detected
425	DG_RESULT_ MODULE _FAULT_25	Fifth fingertip module fault detected
500	DG_RESULT_SOCK_EXCEPTION	An exception occurred in the socket communication.
501	DG_RESULT_SOCK_FAILED_ WSA_START_UP	The socket communication connection failed.
1002	DG_RESULT_NO_GRASP_OBJECT	You didn't grasp the object.
1003	DG_RESULT_ONLY_SUPPORTED_ 3FINGER	For DG-3F-B and DG-3F-M models only.
1004	DG_RESULT_NOT_SUPPORTED_ CONTROL_MODE_OPERATOR	Operator Modes are not a supported feature.
1005	DG_RESULT_NOT_SUPPORTED_ CONTROL_MODE_DEVELOPER	Develop Modes art not supported feature.
1006	DG_RESULT_NOT_SUPPORTED_PORT_NUM	Port number is not supported
2000	DG_RESULT_PORT_EXCEPTION	An exception occurred in serial communication.
2001	DG_RESULT_PORT_FAILED_START_UP	Failed to establish a serial communication connection.
2002	DG_RESULT_PORT_SET_CONFIG_ ERROR	Setting the serial port failed.
2003	DG_RESULT_PORT_SET_TIMEOUT_ ERROR	Failed to set serial port timeout.
2004	DG_RESULT_PORT_SET_COMMASK_ ERROR	Failed to set serial communication mask.
2009	DG_RESULT_DIAGNOSING_SYSTEM	The diagnostic function is in progress.

© 2026 Tesollo Inc. All rights reserved


TESOLLO