



MG400 QUICKSTART GUIDE

By In-Position Technologies





Contents

Getting Started	3
Hardware Required	3
Downloading the Software	4
Connecting to the MG400	5
Configuring Your Hardware	5
Configuring Your IP-Address	7
Connecting to the MG400 with DobotStudio2020	11
Enabling the Robot	
Using the Unlock Button to move the MG400	13
Using the Jog Panel to Move the MG400	14
MG400 IO	17
MG400 IO Wiring Example	
Programming the MG400	19
Basics	19
Saving Points	20
Different Move Types	21
Blockly	22
Porting Blockly Program to Script	25
Blockly Example 1: Basic Moves	27
Blockly Example 2: Using IO #1	28
Blockly Example 3: Modified Parameters in Move	29
Script	
Script Example 1: Basic Moves	
Script Example 2: Basic IO Operations	
Script Example 3: Custom Function	34
Script Example 4: TCP/IP Client	35
Script Example 5: Variable palletizing	
Troubleshooting & Documents	
Dobot Connectivity Guide for MG400	
Hardware User Guide	40
DobotStudio2020 Dobot User Guide	
MG400 Spec Sheet	40
Clearing MG400 Errors	40
LED Functionality	41
Using Sync with IO Commands	41
Contacts	
Phone	
Email	





Getting Started

Hardware Required

To operate the robot, you'll need:

• MG400 Robot



MG400 E-Stop



• A PC running windows



MG400 power adapter



• Ethernet cable (included cable preferred)



 If your PC doesn't have an Ethernet port, you'll need a USB to Ethernet adapter



See <u>Connecting to the MG400</u> for details on cable connections.



Downloading the Software

You'll need the MG400 software before getting connected. Download it off the <u>Dobot official</u> <u>manufacturer's</u> site or contact your distributor for the most recent version. The software you need is called "DobotStudio2020"

	ort/ MG400 Downloa	a Centers							
S	oftware	User manual		SDK		3D Model			
	DobotSCStudio DobotSCStudio-wir	v2.1.8 ndows-offline-installer-2.1	.8exe 2	021. 09. 10				228 MB	🛃 Download
	DobotSCStudio is a M1Pro). Friendly in structures and inte	n industrial robot progra ierface, innovative interac grated virtual simulation e	mming platfor ctive programi environment to	m launched by ming, supportin) realize rapid d	Yuejiang, wl g user seco eployment c	ich is suitable for idary developmer f various process	the whole series of it. It also provides I applications on site	f industrial robots (S kinematics algorithr e. V	SA / SR / CR / MG400 / n of various mechanical ew historical versions
	DobotStudio2020S	20 v1.3.0 etup-1.3.0-stable.202106	exe 202	21. 06. 23				491 MB	🛃 Download
	VeVersion Descript DobotStudio New u - New features: Poi - Various programn - New Ul and intere - design: devices, - Automatic update More features to be * The DobotStudio2 * Thank you for tryi pm@dobot.cc. We	ion: ipgrade, better experience this list, I/O alias, User co- ining methods to choose: ction provide a pleasure applications and plug-ins s keep your software up e explored! 2020 of V1.1.0 version su- ng out this stable version will continue to improve	ce ordinate syste Teach&Playba user experier can be exten to date. pports MG400 . If you find ar the user expe	rm & Tool coordi ack, Graphical p ice. ded. More devi) and M1, pleas ny bugs or have rience of the so	inate system rogramming ces and fund e try it with t any sugges ftware.	, and Pallet. , and Script progr tions will be supp ne right device. tions, please subr	ramming. orted in the future. nit them on the Mer	nu > Help > Feedb	ack page or email to
	If you have trouble	downloading directly, clic	ok here to dow	nload on Goog	le Drive.			V	ew historical versions

While DobotSCStudio functions with the MG400, we recommend using Dobotstudio2020.





Connecting to the MG400

Configuring Your Hardware

Before making any connections, be sure your MG400 power switch is set to "off" (O), like shown here:



Make the following connections to the MG400:

- Plug the power connector into a standard US wall outlet
- Plug the DC output of the power connector into the robot port labeled "POWER"
- Plug the E-Stop connector into the port labeled "E-STOP"
- Plug your Ethernet cable into the port labeled "LAN 1" and plug the other end into your PC

The result should look like this:







Next, flip the ON/OFF switch to the ON position. The switch LED should turn on, as shown here:



The LED on the front side of the robot base will now be blinking white to signify that the robot is booting up. Once the robot is ready for a connection, the LED will be a solid blue:



Robot Booting





Robot Ready







Configuring Your IP-Address

You'll need to configure the IP-Address of the MG400 to be able to connect to it with your computer.¹

Go to the control panel:

Best match		
Control Panel		
Apps Settings UCC Control Contor	>	Control Panel
Intel® Graphics Control Panel Intel® Control Panel	>	☐ Open
Run Windows Administrative Tools	>	Recent Network and Sharing Center
Settings Change User Account Control settings	>	 Connect to a network Allowed apps Programs and Features
 Controlled folder access App & browser control 	>	System Devices and Printers Device Manager
Set up USB game controllers System	>	 View network computers and devices View network status and tasks
Search work and web Control - See work and web results	>	

Click "Network and Internet"



¹ For more information on why IP-Addresses are necessary, see here: https://www.paessler.com/it-explained/ip-address



Click "Network and Sharing Center":

🗅 v 🛦 🚰 🔪 Cambra	Da > Naturark and Internet > > > 71 Search Control Danel		
- T	Pa > Network and internet > • • • •		
Control Panel Home	Network and Sharing Center View network status and tasks Connect a network View network computers and device	vices	
Network and Internet	Internet Option Check network status, change		
Hardware and Sound	Change your home network settings and set preferences Delete browsing history and cooking	es	
Programs	for sharing files and printers.		
User Accounts			
Appearance and Personalization			
Clock and Region			
Ease of Access			

Click "Change adapter settings":

Network and Sharing Center			—	×
	> Network and Sharing Center v	ල Sear	rch Control Panel	Q
Control Panel Home	View your basic network infor	mation and	set up connections	
Change adapter settings	View your active networks			
	IPTech 3		Access type: Internet	
settings	Public network		Connections: 🔐 Wi-Fi (IPTech)	
Media streaming options				
	Unidentified network		Access type: No network access	
	Public network		Connections: 📱 Ethernet	
	Change your networking settings —			
	🛬 Set up a new connection or r	network		
	Set up a broadband, dial-up,	or VPN connec	tion; or set up a router or access point.	
	Troubleshoot problems			
	Diagnose and repair network	problems, or g	et troubleshooting information.	
See also				
Internet Options				



Double click the adapter you'd like to use:

^						
	😰 « Network and > Netv	vork Connections 🗸 🗸	Ō	Search Network Connections		م
Organize 🔻	Disable this network device	Diagnose this connection	Rena	me this connection »		?
		Ethernet Unidentified ne ASIX AX88772 U	swork SB2.0 to	Fast		

Click "Properties" and then double-click "Internet Protocol Version 4 (TCP/IPv4)"

🖗 Ethernet Status	X 🖣 Ethernet Properties X
General	Networking Sharing
Connection IPv4 Connectivity: No network access IPv6 Connectivity: No network access Media State: Enabled Duration: 00:49:05 Speed: 100.0 Mbps Details	Connect using: ASIX AX88772 USB2.0 to Fast Ethernet Adapter Configure This connection uses the following items: This connection uses the following items: Client for Microsoft Networks Client for Microsoft Networks File and Printer Sharing for Microsoft Networks File and Printer Sharing for Microsoft Networks Npcap Packet Driver (NPCAP) Client Protocol Version 4 (TCP/IPv4)
Sent — Received	 ✓ ✓
Packets: 2,463 0	Description Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.
Close	e OK Cancel



Dobot.us

The LAN1 port of the MG400 is always set to the IP address 192.168.1.6 with subnet 255.255.255.0. Set your IP-Address to 192.168.1.x where x is any number between 0 and 255 that is not 6. As an example, I used this one:

You can get IP settings assigned au this capability. Otherwise, you need for the appropriate IP settings.	tomatically if your network supports to ask your network administrator
	ically
	Carry
IP address:	192 . 168 . 1 . 26
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	
Obtain DNS server address au	tomatically
• Use the following DNS server a	addresses:
Preferred DNS server:	
Alternate DNS server:	• • •
	Adumend
Validate settings upon exit	AUVAULEU

Leave the rest of the information blank and then hit okay to exit the IPv4 prompt. Hit okay a second time to exit the Ethernet properties prompt.

		🖣 Ethernet	Properties		
nternet Protocol Version 4 (TCP/IPv4)	Properties	× Networking	Sharing		
General		Connect us	sing:		
You can get IP settings assigned auton this capability. Otherwise, you need to for the appropriate IP settings.	natically if your network supports ask your network administrator	ASI>	(AX88772 USB2.0 to Fas	t Ethernet Adapte	r onfigure
Obtain an IP address automatical	у		ient for Microsoft Networks	3 3	^
Output Use the following IP address:			Mware Bridge Protocol	_	
IP address:	192.168.1.26	Fi 🖳 🗹 📜 Fi	e and Printer Sharing for N	licrosoft Network	s
Subnet mask:	255 . 255 . 255 . 0		pcap Packet Driver (NPC# oS Packet Scheduler	λP)	
Default gateway:			ternet Protocol Version 4 (icrosoft Network Adapter I	TCP/IPv4) Multiplexor Protoc	ol 🗸
Obtain DNS server address autom	atically	<			>
• Use the following DNS server add	esses:	Insta	all Uninsta	ll Pr	operties
Preferred DNS server:		Descriptio	on		
Alternate DNS server:		Allows y	our computer to access re	sources on a Micr	rosoft
Validate settings upon exit	Advanced				
	OK Cancel			ОК	Cancel

You are now ready to connect to the MG400.



Dobot.us

Connecting to the MG400 with DobotStudio2020

Open DobotStudio2020. The screen should look like this:

O DobotStudio2020 1.3.0-stable.202106191640			- 🗆 X
🗿 Dobot Studio 🚊 🏠		MG400 V 192.168.1.6 MG400 Disconnected Connect	Image: Clobal Speed(50%) Image:
			Control
Welcome to DobotStudio2020			
- Brand new user interface - Easy to use and user friendly - More Dobot products will be supported VERSION: 1.3.0-stable.202106191640 Please email to pm@dobot.cc if you have any suggestions.		+2	
Recent Projects	Teach & Playback	DobotBlockly	Script
Apple_USBC 2021-08-23 16:00:41			
ECIPLATX_DEMO 2021-08-11 10:23:45			
circleTest 2021-08-02 13:36:12			
CamcraftPalletPr., 2021-07-28 15:15:43 script_CamcraftPalletProg	Remote Control		

The search bar should have the MG400 pre-populated:



If this is not the case:

- Check that IP-Address is correct (See <u>Configuring Your IP-Address</u>)
- Check that the robot is powered on and shows a blue solid light (See <u>Configuring Your</u> <u>Hardware</u>)
- Try the Dobot connectivity guide (See <u>Dobot Connectivity Guide for MG400</u>)
- If the above options all fail, get in touch with your Dobot distributor for additional support

If you are able to connect, hit connect. The robot should now be online.





Enabling the Robot

In robotics, the robot is unable to move until it has been enabled. Before enabling the MG400, Joints 2 and 3 are mechanically locked and Joints 1 and 4 can be rotated freely. This state is considered the "disabled state." After enabling the MG400 by left-clicking the button shown below, the MG400 is enabled. The robot's joints are now active and cannot be moved by hand.



Upon starting up and trying to enable for the first time, the MG400 will ask you for the payload information.



The definition of "Payload" is the net weight that you have attached to the robot. The MG400 can handle a maximum payload of 750g.

If you have not attached anything to the end of arm tool (like in the picture below), ensure that the payload is set to 0g. Knowing this weight helps the robot determine when it has collided with something while moving. The offset values are the x and y offsets of the center of gravity of your tool. You may leave these offsets blank most of the time.







If you attach something to the robot in the future, be sure to change the payload in the MG400 settings.

After enabling the robot, you are ready to begin motion. The front LED should now be green. If an error appears and the robot remains disabled, that means something is preventing the robot from booting. See <u>Clearing MG400 Errors</u>.

Using the Unlock Button to move the MG400

A great perk of the MG400 is the built-in unlock button. The unlock button allows the user to "drag and drop" the robot. This allows for fast programming and development compared to traditional methods.

To use the unlock function, simply click the button on the GM400 arm. The green light will change to blinking blue. You can now re-position the robot freely. Be sure to click the "unlock" button again before commanding motion.







Using the Jog Panel to Move the MG400

For a more repeatable way to move the robot, the user can use the jog window to position the robot electronically. To open the jog window, click "control" on the right side of DobotStudio2020:



From there, you will be able to jog the robot by either cartesian or joint moves. Here are the cartesian/linear coordinates: **7**





Here are the joint coordinates:



In "Jog" mode, the robot will jog in the selected direction until the button is released. The speed of the robot is controlled by the speed slider on the top of the screen.

Mode	Jog	Step
(r	Global Speed(100%)	бор



In "Step" mode, the robot will increment the selected distance in mm whenever the user taps a button. For example, the following button configuration would drive the robot 1mm in the x-direction when clicked:



To get rid of the jog window, hit the "fold" button:





Dobot.us



MG400 IO

The MG400 has 16 digital inputs and 16 digital outputs that function on 24VDC that are visible on the back of the robot base. The MG400 also has 2 digital inputs and 2 digital outputs on the arm of the robot for easy access to robot tools. Dobot provides the connector for this IO with the MG400.



To use the rear-IO, compress the orange tab and insert a wire. Release the tab and the wire should be firmly clamped in the IO slot. We recommend zip-tieing your automation setup so the IO wires don't have any strain.

To view IO status and toggle IO easily, use the "I/O" tab:



Deline Global Speed(100%)



Dobot.us

MG400 IO Wiring Example

Here is an example on how to wire two 24VDC sensors with the following pinouts:





Note: your sensors may have different wiring. Incorrect wiring could damage the MG400.



Programming the MG400

Basics

There are many ways to program the MG400. In this guide, we will cover <u>Blockly</u> and <u>Script</u>. Blockly is a visual, block-based programming method. Script is programming in Lua, a high-level programming language similar to Python. Generally, Blockly is friendlier to new users and quick applications while Script is preferred by more experienced users and is suited towards

Blockly

- Visual block-building
- Also known as "Scratch"
- Suited towards simple applications

Script

- Simple structured-text programming in Lua

 Similar to Python
- Suited towards more advanced applications/more experienced users

For a first-time application, it is recommended to start with Blockly. You may port your project to a script project (see <u>Porting Blockly Program to Script</u>).



DOROT



Saving Points

Saving points is the same in Blockly and in Script and is usually done before writing any code.

Open the points list in the top right of the screen. Hitting "add" will take the robot's current cartesian position and save it as a point.

You can rename points by double-clicking the name P1, P2, etc.

You can tweak a point by double-clicking an X,Y,Z, or R value.

By clicking a point, you are given the options to "Cover", "RunTo", or "Delete". Cover will overwrite the selected point with the current position of the robot. RunTo will move the robot to the selected point. Delete will remove the point.

> Points						-	Control
Name	User	Tool	х	Y	Z	R	
InitialPose	0	0	350	0	0	o	Points
P1	0	0	247.43	-34.55	-1.239	-18.15	1/0
P2	0	0	164.72	273.38	-106.5	53.343	
				Cov	er RunTo	Delete	Pallet
P3	0	0	179.07	-236.4	41.123	11.452	
P4	0	0	250.83	-25.67	161.77	-2.928	
			Add				





Different Move Types

In Dobot Studio, you'll find several move types. Here is a list of their functions and where to use them:

- MovJ
 - Joint-based trajectory move to point P
 - \circ Fastest, most efficient move
- MovL
 - Linear Trajectory to point P
 - Good for insertions, picking parts, paths
- Jump
 - Builds in a routine that moves linearly up a height and down to point P

List of parameters in settings

---- z_limit | h2 |---P

Settings panel X									
2 Motion ty	pe								
MovJ	MovL	Jump	JointMovJ						
RelMovJ	RelMovL	Arc	Circle						
👯 Parameter	r configuration								
Coordinates of p P:	oint InitialP	ose 🗸	Custom						
Advanced sett	ing		~						
	C	ancel	Save						

- JointMovJ
 Mo
 - Move to joint position instead of cartesian position
 - Generally not used
- RelMovJ
 - MovJ to an offset X,Y,Z,R position instead of to a fixed point
 - Joint Trajectory
- RelMovL
 - MovL to an offset X,Y,Z,R position instead of to a fixed point
 - o Linear Trajectory
- Arc
- Partial circular trajectory
- Circle
 - Full circle trajectory







Blockly

Open up a new Blockly program by clicking the "Homepage" button and then clicking the "DobotBlockly" button



blocks



Start your program by adding some points in your points list (See <u>Saving Points</u>). To add a point, navigate to "Move" and then hit "Advanced configuration":

🗁 🛛 File	e 🖾 Save 🖓 Undo	🖒 R	edo	,		D I	Deb	bug] [D	S	tart	0	Sto	op				
	Move																		
events	Advanced configuration			-				i.											
Control	Nove in MovJ 👻 mode to point			S	start			:											
perators	Move in RelMovJ mode to point	Ax																	
Move	Move in Jump mode to point	Raise																	
Move	Move in Jump mode to point	urch p																	
	Exercise in circle mode: middle point																		
1/0	Move in arc mode: middle point	еп																	
Program anagemer																			
Posture	Move Arguments					- 14 - 14 - 14													

This will open the move panel. Select your type of move and hit "Save" (for more info, see <u>Different</u> <u>Move Types</u>)

Settings panel X										
Motion type										
MovJ	MovL	Jump	JointMovJ							
RelMovJ	RelMovL	Arc	Circle							
📢 Parameter	configuration									
Coordinates of p P:	P1	~	Custom							
Advanced sett	ing		~							
	С	ancel	Save							



Dobot.us

Drag the block into a position in your blockly workspace. Note that the blocks need to "click" together.

To copy blocks, select a block, hit "ctrl + x", and "ctrl + v" to paste. From there, you can edit the point names. You can also right click and hit "duplicate" to make copies. From there, you can make simple programs. For example, the program shown here will move to points 1, 2, 3, and 4.





Porting Blockly Program to Script

Often times, it's easier to start in Blockly and transfer to script once the basic program is there.



Start by opening your blockly program. Be sure it is saved. Navigate to File -> Export Project

Name the file and save it somewhere you will remember. Then hit "Export"

Export project						×	:
← → ∽ ↑ 🕹 > This PC > Downloads		~	Ū	Search Downlo	bads	Q	
Organize 🔻 New folder						- ?	
↑ Name	Date modified	Туре		Size			~
File name: myBlocklyExample						~	Ì
Save as type: Project File (*.*)						~	,
∧ Hide Folders				Export		Cancel	





After that, open up the script interface.





Go to File -> Import project and select your exported project

ObotStudio2020 1.3.0-stable.202106191640	
🗿 Dobot Studio 🚊 🎧	
🗁 File 🖾 Save 🛛 🏠 Undo	💫 Redo
New (Ctrl+N)	Q
Open (Ctrl+O)	
Save As (F12)	-
Import project a point	I I I I I I I I I I I I I I I I I I I
Export project	
C Dinear Movement	[@
MOVL	
Point to point, the target point is Joint point	<mark>له</mark>

Import from:					×
← → ∽ ↑ 🕂 > This PC	C > Downloads >		✓ Ö Sear	rch Downloads	Q
Organize 🔻 New folder					?
	 Name ✓ Today (1) 	Date modified	Туре	Size	^
	myBlocklyExample	9/13/2021 4:11 PM	File folder		
					-
					~
Folder: r	myBlocklyExample				
				Open Cano	el

Your blockly program will now be displayed as script.



Blockly Example 1: Basic Moves

This program will run to points 1, 2, 3, and 4. At each point, the MG400 will wait 1 second to proceed. This process will repeat 10 times before ending.

		-					2		2				
- 6							2	÷.,	\sim		$\widehat{a}(\cdot)$		
S	Start					5	÷.	÷.					
						2	×.				13		
	eneat	10											
										•			
										-		ŀ	
	IN M	ovJ n	IOV	e m	100	e t	0 0)OII	nt (P	1	1	
												2	
		1) :	sec	on	ds	ε.	÷.	\sim		42		
	_	-		12000				1		21			
	in M	ov I m		a m	od	lo t		oir		P		Ŀ	
		UVJII	IUV	e n	100	eı	υμ			E			
		-					÷		1.	1		1	
	wait	1		sec	on	ds	÷	,		÷			
		_			-	-	-				_	r.	
	in M	n Lvo	IOV	e m	lod	le t	0 0	oir	nt (P	3		
· •										-		Ŀ	
	wait	4	٦.					*	-	(\mathbf{r})			
			1	sec	om			1		*	•		
										-		r.	
	in M	n Lvo	IOV	e m	lod	e t	o p	oir	nt (P	4)		
							-						
	wait	1		sec	on	ds	1	*			*	•	
		-					٠.						
					3	1		*	1		÷2		-
	-						. *		۰.		1		
	1.1	1.1	* :			2		1		2	• :		
0.000								1.0					1.0





Blockly Example 2: Using IO #1

This program will execute a loop 10 times. Each time, it will wait until Digital Input 1 is high. After the signal goes high, it will execute 10 times.

		1						1				1				1						
-		2								2												
	1. 1.	2																2				
tart																						
pear 10																						
Stop at th	is poir	nt																				
		-																		_	_	
report up		-	_					dia	ital				DI.	04				1	7			
repeat un		R	ea	u s	Idi	us	UI	aig	itai	m	JUL		UI_	_01				1	2			
					r															-	-	
wait	0.01		cor																			
Stop of																						
Stop at	this p	oint						2	2	1					1	2	2	2				
Stop at	this p	oint		2		1	1	2	ŝ.	1	į.	1	1		1	2	4	-	į.	-		-
Stop at	this p	oint		2			1			1	i.	1	i	-		1	1	-	Ì		,	
in Moy Lo	this p	oint						1	1						1	1	1	1			,	
in MovJ n	this pendot	nod	e te	o p	ioir	nt (P	1	i I					:	1	2		1) >	
in MovJ n	this p	nod	e to) o p	, ooir	nt (P	1													,	
in MovJ n	nove n	nod	e to ts	o p	oir	nt (P	1													· •	
in MovJ n	nove n	oint nod	e to ts	o p	oir	nt (P	1														
in MovJ n wait 1 in MovJ n	nove n sec	oint nod	le to ds le to	o p	oir	nt (P	1								NAMES OF A DESCRIPTION OF A DESCRIPTION OF A DESCRIPTIONO					· · · · · · · · · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n	nove n sec	oint nod	ie to ds	o b	ooir	nt (P	1													· · · · · · · · · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n	nove n sec	nod nod	ie to ds ie to	o p	oir	nt (P	1													· · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n wait 1	nove n sec nove n sec	nod nod nod	le to ds le to ds	o p	ooir	nt (P	1													· · · · · · · · · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n wait 1	nove n sec nove n sec	nod conc nod	ie to ds ds ds	o p	ooir	nt (nt (P	1													· · · · · · · · · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n wait 1	nove n sec nove n sec	nod nod nod	ie to ds ds	o p	oir	nt (22) nt (23)	P	1										A A A ACADAGE A A A A	1.11 一日 日日 日日 日日 日日 日日 日日		*	
in MovJ n wait 1 in MovJ n wait 1	nove n sec nove n sec	nod ond ond ond	le to ds ds	o b	oir	nt (24) 24) 24) 24) 24) 24) 24) 24) 24) 24)	P	1													· · · · · · · · · · · · · · · · · · ·	
in MovJ n wait 1 in MovJ n wait 1	nove n sec nove n sec	nod nod nod	le to ds ds	o p	ooir	nt (P	1								1999 日本市民的主要的主要的		A DE LE REPORT DE LE			· · · · · · · · · · · · · · · · · · ·	

Without the "Stop at this point" command, the program will not work. For more information on why the Sync function is necessary, see <u>Using Sync with IO Commands</u>.

Body I/O			
Digital input name	e		
DI_01	•	DI_09	
D1_02		DI_10	0

Remember you can monitor this with the IO panel to help debug: (right side of studio)



Blockly Example 3: Modified Parameters in Move

By right-clicking a move and hitting "edit", you may make special configurations for that move. For example, you can have slower velocity and acceleration on an insertion move. Or you can Have an IO toggle a certain distance into a move.

		4						
Start								
~								
in MovJ mov	e mode to point	P2						
	× ×							
in Moy.I moy	e mode to point	P4						
	o modo to point					6.2		
in Movil mov	a mode to point		duana	d oor	figuro	tion	(Cnood In:	101
III WOVJ IIIOV	e mode to point		uvance	eu con	ingura	uon	{Speed]=	10}
		1.1				1.1		
repeat 10	and a second second							
	an an anaith		-	28.0		1.135	-	
in MovJ m	ove mode to point	P4	Advar	iced c	onfigu	ration	{Speed.	J=10}
in MovJ m	ove mode to point	P2						
		_	·					
in MovJ m	ove mode to point	P3	1.1.1					
		0	1.100					
in Mov I m	ove mode to point	P4						
III WOVS III	ove mode to point	14						
the between the second		-					(Onered	1 403
in Movj m	ove mode to point	PT	Advar	iced c	onngu	ration	{Speed.	J=10}
	and the second							
wait 5	seconds							
	J							
Coordinates of point	P4 ~	Custor	n					
Advanced setting			^					
Speed Speed	••							
Acceleration	0							
CP	0							
Process I / O set	tings ?							
DO_01		~	•					
Trigger mode	Distance V							
Distance	2	mm						
	+							
	Cancel	Course						
	Cancel	Save						



Dobot.us



Script

Open up a new Script program by clicking the "Homepage" button and then clicking the "Script" button



This scripting language is called Lua. It is similar to Python and should be

familiar to those who know structured text programming. Aside from the Dobot functions, everything in Lua is supported here.²

There are two files that open on bootup: scr0.lua and global.lua.

- scr0.lua
 - Used for motion commands
 - Used for sequential code
- global.lua
 - Used for global variables
 - Used for user functions
- scr1.lua, scr2.lua, etc.
 - User may create as many parallel threads as they desire by hitting the +



- Recommended for monitoring IO, TCP/IP communications, etc.
- It is recommended to avoid motion commands in parallel threads. This could lead to erratic behavior



² If you are unfamiliar with structured-text programming, there is a great Lua guide here: https://www.tutorialspoint.com/lua/index.htm

Points Manager, IO, Jog window **Robot Functions** Start/Stop Workspace DobotStudio2020 0-stable.202106191640 х 3 Ø Dobot tudio 📃 🖹 untitled Save | 🔇 Undo 🖉 Redo Debug 🗁 - File ٢ 🕟 Start src0.lua × global.lua × + (iii) Points Version: Lua 5.4.1 + .M. Motion 8 + Param + :0: I/O Dallat 主 淰 Util + Pose + TCP + UDP + Modbus + Conveyer Belt + Stack





Users may use the built-in robot functions on the left to quickly deploy code without worrying about Syntax:



"--" signifies a comment in Lua





Script Example 1: Basic Moves

This script runs to P1 and P2 with normal parameters. Afterwards, it goes to P3 with 50% speed and 20% velocity with a CP = 1.

```
src0.lua
MovJ(P1) --move to P1 with a joint trajectory
MovL(P2) --move to P2 with a linear trajectory
MovJ(P3, {CP=1, SpeedJ=50, AccJ=20}) --move to P1 with a joint trajectory at 50% speed
```

global.lua

--empty

Script Example 2: Basic IO Operations

This script iterates 10 times in a loop. It starts by moving to P3 and then will move to P1 is DI1 is high or P2 if DI1 is not high.

```
src0.lua
for count = 1, 10 do
   MovJ(P3)
   Sync()
   if (DI(1))==1 then
        MovJ(P1)
   else
        MovJ(P2)
   end
   Wait(1000)
end
```

global.lua

--empty





Script Example 3: Custom Function

If you have repetitive code, use a custom function to save on time and programming efficiency. The below example uses a custom function named "partInsertion()". That function is defined in global.lua and called in scr0.lua during execution.

src0.lua

```
while true do
  MovJ(P1) --clearpoint
  MovJ(P2) --process point 1
  partInsertion()
  MovJ(P3) --process point 2
  partInsertion()
  MovJ(P4) --process point 3
  partInsertion()
  Wait(1500)
end
```

global.lua

```
function partInsertion()
RelMovL({0, 0, -10,0}, {CP=0, SpeedL=20, AccL=20}) --move 10mm downwards at 20% speed
Sync() --Sync() before IO commmand
DO(1,ON) --toggle DIO on
Wait(1500) -- wait 1500ms (1.5s)
Sync() --Sync() before IO commmand
DO(1,OFF) --toggle DIO off
RelMovL({0,0,10,0}) --move back up
end
```





Script Example 4: TCP/IP Client

You can have the MG400 run a program on-board

```
src0.lua
-- Version: Lua 5.4.1
--Robot code:
local ip="192.168.1.6" -- IP address of the robot as a server
local port=6001 -- Server port
local err=0
local socket=0
--- PROGRAM
err, socket = TCPCreate(true, ip, port)
if err == 0 then
       err = TCPStart(socket, 0)
       if err == 0 then
               local RecBuf
               while true do
                       TCPWrite(socket, "tcp server test") -- Server sends data
                       err, RecBuf = TCPRead(socket, 10, "string") -- Server receives
                       if err == 0 then --if you received a message ...
                               Go(P1) --Start to run motion commands
                               Go(P2)
                               print(RecBuf.buf)
                       else
                               print("Read error ".. err)
                               break
                       end
                       Wait(100)
               end
               else
               print("Create failed ".. err)
       end
       TCPDestroy(socket)
else
       print("Create failed ".. err)
end
```

global.lua

--empty

This may be run with the python script shown here:







Script Example 5: Variable palletizing

This script takes in variables for X,Y,Z, and R positions of two pallets sitting in any X-Y plane. This example program is of a pick + camera inspect + place sequence.

src0.lua --IO items cameraSignal in = 5 --set this to the digital input of the camera cameraSignal out = 3 --set this to the digital output trigger from the robot to the gripperSignal out = 2 --set this to the digital output for the gripper --input pallet parameters (this is where you pick up parts) x input startingPoint = 300 --x coordinate of robot when above the first pallet slot y input startingPoint = 100 --y coordinate of robot when above the first pallet slot x input pallet = 10 --x distance between spots in pallet, defined in mm y input pallet = -25 --y distance between spots in pallet, defined in mm z input pallet = 10 --defined in mm, same for all input pallet points r input pallet = 90 --defined in degrees, same for all input pallet positions x input num iterations = 4 y input num iterations = 3 --this is an 8x3 pallet where we have 8 rows (x shifts), 3 columns (y shifts) num parts input pallet = x input num iterations * y input num iterations --calculate total input items --output pallet parameters (this is where you put good parts) x output startingPoint = 250 --x coordinate of robot when above the first pallet slot y output startingPoint = -70 --y coordinate of robot when above the first pallet slot x output pallet = -15 --x distance between spots in pallet, defined in mm y output pallet = 26 --y distance between spots in pallet, defined in mm z output pallet = -10 --defined in mm, same for all output pallet points r output pallet = 0 --defined in degrees, same for all output pallet positions x output num iterations = 3y output num iterations = 4 --this is an 8x3 pallet where we have 8 rows (x shifts), 3 columns (y shifts) num parts output pallet = x output num iterations * y output num iterations --calculate total input items --reset counters x input iter = 0y input iter = 0x output iter = 0y output iter = 0input pallet finished = 0output pallet finished = 0-- PROGRAM VARIABLES (DO NOT MODIFY) --set inital pick and place point pickPoint = {armOrientation="right", coordinate={x input startingPoint+x input iter*x input pallet,



```
y_input_startingPoint+y_input_iter*y_input_pallet, z input pallet, r input pallet,
0.000000, 0.0000000},tool=0, user=0}
placePoint={armOrientation="right",
coordinate={x output startingPoint+x output iter*x output pallet,
y output startingPoint+y output iter*y output pallet, z output pallet,
r output pallet, 0.000000, 0.0000000},tool=0, user=0}
-- PROGRAM (USER MAY MODIFY)
Go(clearPoint) --this is my clear point
print('Starting Program')
while (input pallet finished == 0) and (output pallet finished == 0) do
 --pick part
 MovJ(pickPoint)
 inputPalletIter()
 pickPart()
  --INSPECT PART
 MovJ(cameraPoint)
 DO (cameraSignal out, ON)
 Sync()
 Wait (500)
 DO(cameraSignal out,OFF)
  Sync()
 --place part if good
 if DI(cameraSignal in) == OFF then -- put your camera's "good part" digital input here
   MovJ(placePoint)
   outputPalletIter()
   placePart()
 else
   MovJ(dumpPoint)
   placePart()
 end
end
--clearpoint
Go(clearPoint)
```

global.lua

```
-- Version: Lua 5.4.1
function pickPart()
  local Offset = {0, 0, -10, 0}
  local Option={CP=1, SpeedL=50, AccL=20}
  RelMovL(Offset, Option)
  Sync()
  DO(gripperSignal_out, ON)
  Wait(500)
  Offset = {0, 0, 10, 0}
  RelMovL(Offset, Option)
  Sync()
```



```
end
function placePart()
 local Offset = \{0, 0, -10, 0\}
 local Option={CP=1, SpeedL=50, AccL=20}
 RelMovL(Offset, Option)
 Sync()
 DO (gripperSignal out, OFF)
 Wait (500)
 Option={CP=1, SpeedL=100, AccL=75}
 Offset = \{0, 0, 10, 0\}
 RelMovL(Offset, Option)
 Sync()
end
function inputPalletIter()
  --this funciton iterates the input pallet
 if (x input iter+1) < x input num iterations then
    --if x row isn't finished, keep 1 incriment in the x
    x input iter = x input iter + 1
  elseif (y_input_iter+1) < y_input_num iterations then</pre>
    --else if y column isn't finished, x iter set to 0, incriment 1 in the y
    print('GOT HERE')
    x input iter = 0
    y input_iter = y_input_iter + 1
 else
    --else input pallet is done
   print('INPUT PALLET DONE')
   x input iter = 0
    y input iter = 0
    input pallet finished = 1
  end
 pickPoint = {armOrientation="right",
coordinate={x input startingPoint+x input iter*x input pallet,
y input startingPoint+y input iter*y input pallet, z input pallet, r input pallet,
0.000000, 0.0000000}, tool=0, user=0}
 Sync()
end
function outputPalletIter()
  --this funciton iterates the output pallet
  if (x output iter+1) < x output num iterations then
    --if x row isn't finished, keep 1 incriment in the x
    x output iter = x output iter + 1
  elseif (y output iter+1) < y output num iterations then</pre>
    --else if y column isn't finished, x iter set to 0, incriment 1 in the y
    print('GOT HERE')
    x_output_iter = 0
    y output iter = y output iter + 1
  else
    --else input pallet is done
```



```
print('OUTPUT PALLET DONE')
x_output_iter = 0
y_output_iter = 0
output_pallet_finished = 1
end
placePoint={armOrientation="right",
coordinate={x_output_startingPoint+x_output_iter*x_output_pallet,
y_output_startingPoint+y_output_iter*y_output_pallet, z_output_pallet,
r_output_pallet, 0.000000, 0.0000000},tool=0, user=0}
Sync()
end
```



Troubleshooting & Documents

Dobot Connectivity Guide for MG400

How to connect MG400 using software

Hardware User Guide



Dobot MG400 Hardware User Guide

DobotStudio2020

Dobot User Guide



DobotStudio2020 User Guide V1.1.1(2020)



Clearing MG400 Errors

The MG400 will error out when the user tries to perform one of many forbidden actions or the robot cannot perform a given task. You can tell the robot is in an error state without a PC because the front LED will be blinking red. Common cases where a user can see an error include:

- Attempting to enable in an invalid position
- A collision was detected
- The robot is violating the parallelogram limits (too close to the base)
- The user attempted to command the robot to hit itself or move outside the max reach

An error is indicated by the log in the top right of DobotStudio2020:

 						-	٥	×
Га MG400	MG400 192.168.1.6 <u>Connected</u>	✓ Online		Global Speed(51%)	•	-	S	op
								Control
								l/O

To clear the error, press the log and view what the error was. It may require you to resolve the issue before hitting "clear alarm."





LED Functionality

The LED on the front right of the MG400 base can glow several colors. This tells us what state the robot is in without needing a computer.

Color	State	Meaning
White	Blinking	Booting
Blue	Solid	Ready, disabled
Blue	Blinking	Unlock button pressed – free to drag robot
Green	Solid	Enabled, Ready
Green	Blinking	Executing Program
Red	Blinking	Error



Using Sync with IO Commands

The Sync() function ("Stop at this point" in blockly) is essential for the use of MG400 DIO.

The reason for this is queuing. Similar to other robots, the MG400 compiles all commands very quickly and then executes them in real time. The reading of inputs is generally queued immediately and not in sync with the real-time execution, like we generally want IO to be.

When reading from an IO or writing to an IO, use a Sync() or Stop at this point block before the IO command, like shown below:



Reading from an IO

Writing to an IO







Contacts

Don't hesitate to reach out for help, questions, and sales. You may reach IP-Tech Dobot support in any of the following ways:

Phone

Customer Service/Sales: 877-478-3241

Email

Help email: <u>help@iptech1.com</u>



