Table of Contents

Introduction 1.1

1 Motion Commands 1.2

- 1.1 Point to point, the target point is Cartesian point 1.2.1
- 1.2 Linear Movement 1.2.2
- 1.3 Point to point, the target point is Joint point 1.2.3
- 1.4 Jump Movement, Jump parameters can be set in this command 1.2.4
- 1.5 Jump Movement, Jump parameters are called by Arch index 1.2.5
- 1.6 Move to the Cartesian offset position in a point to point mode 1.2.6
- 1.7 Move to the Cartesian offset position in a straight line 1.2.7
- 1.8 Linear movement in parallel with output 1.2.8
- 1.9 Point to point movement in parallel with output 1.2.9
- 1.10 Arc Movement 1.2.10
- 1.11 Circle Movement 1.2.11

2 Motion Parameters 1.3

- 2.1 Joint Acceleration 1.3.1
- 2.2 Cartesian Acceleration 1.3.2
- 2.3 Joint Speed 1.3.3
- 2.4 Cartesian Speed 1.3.4
- 2.5 CP 1.3.5
- 2.6 Synchronization 1.3.6
- 2.7 Set Load Parameters 1.3.7

3 IO 1.4

- 3.1 DI 1.4.1
- 3.2 DO 1.4.2
- 3.3 DOInstant 1.4.3

4 Program Managing Commands 1.5

- 4.1 Motion command waiting 1.5.1
- 4.2 Blocking instruction issuance 1.5.2
- 4.3 Pause program operation 1.5.3
- 4.4 Start timing 1.5.4
- 4.5 Stop timing 1.5.5
- 4.6 Get current time 1.5.6

5 Pose 1.6

- 5.1 Get Cartesian coordinates 1.6.1
- 5.2 Get Joint coordinates 1.6.2

5.3 Cartesian point offset 1.6.3
5.4 Joint point offset 1.6.4
5.5 Cartesian point 1.6.5
5.6 Joint point 1.6.6
6 TCP 1.7
6.1 Create TCP 1.7.1
6.2 Establish TCP connection 1.7.2
6.3 Receive TCP data 1.7.3
6.4 Send TCP data 1.7.4
6.5 Close TCP 1.7.5
7 UDP 1.8
7.1 Create UDP 1.8.1
7.2 Receive UDP data 1.8.2
7.3 Send UDP data 1.8.3
8 Modbus 1.9
8.1 Read the value from Modbus slave coil register address 1.9.1
8.2 Set the coil register in the Modbus slave 1.9.2
8.3 Read the value from the Modbus slave discrete register address 1.9.3
8.4 Read the value from the Modbus slave input register address 1.9.4
8.5 Read the value from the Modbus slave holding register address 1.9.5
8.6 Set the holding register in the Modbus slave 1.9.6
9 Conveyor Tracking 1.10
9.1 Set conveyor number to create a tracing queue 1.10.1
9.2 Obtain status of the object 1.10.2
9.3 Set X,Y axes offset under the set User coordinate system 1.10.3
9.4 Set time compensation 1.10.4
9.5 Synchronize the specified conveyor 1.10.5
9.6 Stop synchronous conveyor 1.10.6
10 Pallet 1.11
10.1 Instantiate matrix pallet 1.11.1
10.2 Instantiate teaching pallet 1.11.2
10.3 Set the next stack index which is to be operated 1.11.3
10.4 Get the current operated stack index 1.11.4
10.5 Set the next pallet layer index which is to be operated 1.11.5
10.6 Get the current pallet layer index 1.11.6
10.7 Reset pallet 1.11.7
10.8 Check whether the stack assembly or dismantling is complete 1.11.8
10.9 Release palletizing instance 1.11.9
10.10 The robot moves from the current position to the first stack position as the configured stack

assembly path 1.11.10

10.11 The robot moves from the current position to the transition point as the configured stack dismantling path 1.11.11

Program Guide

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 15:03:14

1 Motion Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Point to Point, the target point is Cartesian point

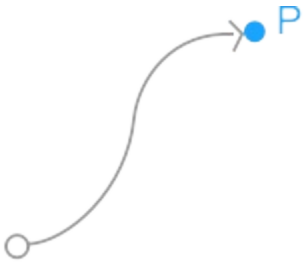
- Function:

```
MovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}
MovJ(P, Option)
```

- Description: Point to Point, the target point is Cartesian point.
- Required parameter: P，Indicate target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Linear Movement

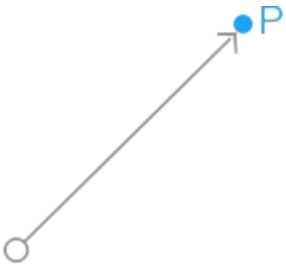
- Function:

```
MovL(P)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}
MovL(P, Option)
```

- Description: Linear Movement, the target point is Cartesian point.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Point to point, the target point is Joint point

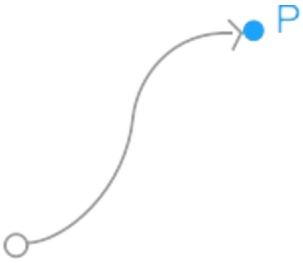
- Function:

```
JointMovJ(P)
```

Or:

```
local Option={CP=1, SpeedJ=50, AccJ=20}
local P={joint={J1,J2,J3,J4}}
JointMovJ(P, Option)
```

- Description: Point to point, the target point is Joint point.
- Required parameter: P,Indicate the target point, which is user-defined or obtained from the points list. Only joint point is supported.
- Optional parameter:{CP=1, SpeedJ=50, AccJ=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100



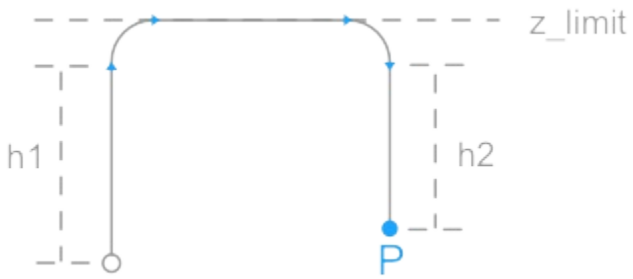
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-03-03 17:41:10

Jump Movement, Jump parameters can be set in this command

- Function:

```
local Option={SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters can be set in this command.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, ZLimit=100, End=20}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Start: Lifting height(h1).
  - ZLimit: Maximum lifting height(z_limit).
  - End: Dropping height(h2).



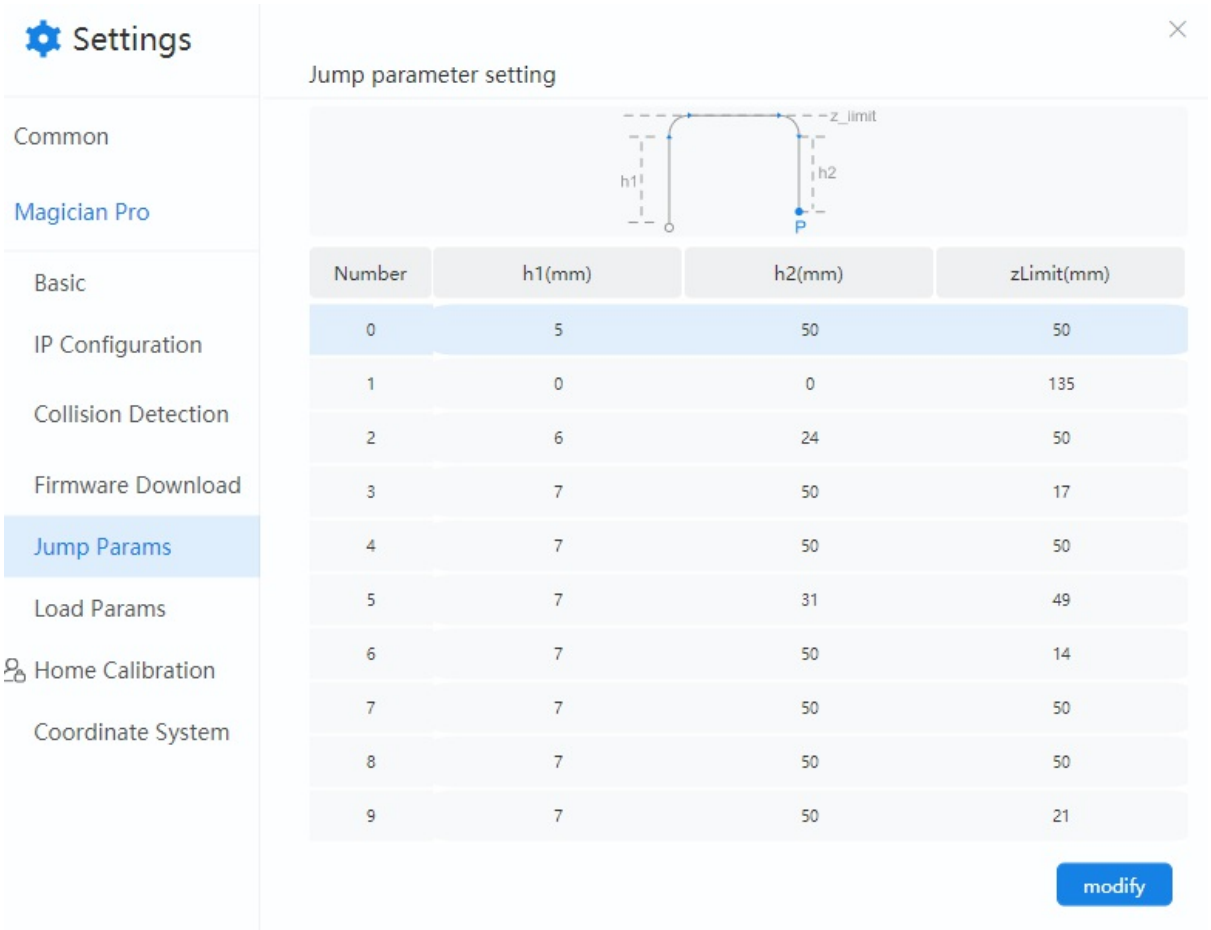
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:21:39

Jump Movement, Jump parameters are called by Arch index

- Function:

```
local Option={SpeedL=50, AccL=20, Arch=1}
Jump(P, Option)
```

- Description: Jump Movement. The jump parameters are called by Arch index.
- Required parameter: P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter: {SpeedL=50, AccL=20, Start=10, Arch=1}
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100
  - Arch: Arch index. Value range: 0 - 9. Please set Jump parameters on the **Settings > Jump Params** page.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:21:14

Move to the Cartesian offset position in a point to point mode

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
RelMovJ(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
local Option={CP=1, SpeedJ=50, AccJ=20}
RelMovJ(Offset, Option)
```

- Description: Move to the Cartesian offset position in a point to point mode.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR}, X, Y, Z, R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedJ=50, AccJ=20}. You can double-click to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Move to the Cartesian offset position in a straight line

- Function:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
RelMovL(Offset)
```

Or:

```
local Offset = {OffsetX, OffsetY, OffsetZ, OffsetR}
local Option={CP=1, SpeedL=50, AccL=20}
RelMovL(Offset, Option)
```

- Description: Move to the Cartesian offset position in a straight line.
- Required parameter: {OffsetX, OffsetY, OffsetZ, OffsetR} , X, Y, Z ,R axes offset in the Cartesian coordinate system.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click to inset the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Linear movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}
MovLIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}
local Option={CP=1, SpeedL=50, AccL=20}
MovLIO(P, IO, Option)
```

- Description: Linear movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode：Set Distance mode. 0: Distance is a percentage；1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status：Status of the digital output port.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Point to point movement in parallel with output

- Function:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}
MovJIO(P, IO)
```

Or:

```
local IO={{Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...}
local Option={CP=1, SpeedL=50, AccL=20}
MovJIO(P, IO, Option)
```

- Description: Point to point movement in parallel with output . Multiple digital output ports can be set.
- Required parameter:
  - P, Indicate the target point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {Mode, Distance, Index, Status},{Mode, Distance, Index, Status},...: Multiple digital output ports can be set.
    - Mode：Set Distance mode. 0: Distance is a percentage；1: Distance from the starting point, or from the target point.
    - Distance: If the Mode is a percentage, it represents the percentage of the distance between the starting point and the target point. If the Mode is a distance, it represents the distance from the starting point, or from the target point. If the distance is set to positive, it indicates the distance from the starting point; if set to negative, it indicates the distance from the target point.
    - Index: Digital output port. Value range: 1- 18
    - Status：Status of the digital output port.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedJ: Velocity rate. Value range: 1 - 100
  - AccJ: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Arc Movement

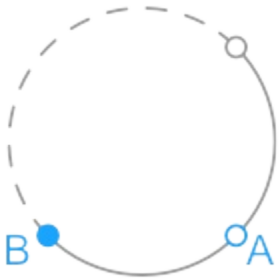
- Function:

```
Arc(P1, P2)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}
Arc(P1, P2, Option)
```

- Description: Arc movement. This command needs to combine with other motion commands, to obtain the starting point of an arc trajectory.
- Required parameter：
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, End point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
- Optional parameter：{CP=1, SpeedL=50, AccL=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Circle Movement

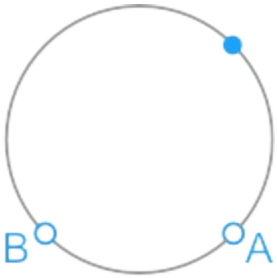
Function:

```
Circle(P1, P2, Count)
```

Or:

```
local Option={CP=1, SpeedL=50, AccL=20}
Circle(P1, P2, Count, Option)
```

- Description: Circle movement. This command needs to combine with other motion commands, to obtain the starting point of a circle trajectory
- Required parameter:
  - P1, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - P2, Middle point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - Count, Number of circles.
- Optional parameter: {CP=1, SpeedL=50, AccL=20}. You can double-click to insert the command with optional parameters.
  - CP: Continuous path rate. Value range: 0-100
  - SpeedL: Velocity rate. Value range: 1 - 100
  - AccL: Acceleration rate. Value range: 1 -100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

2 Motion Parameters

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Joint Acceleration

- Function:

```
AccJ(R)
```

- Description: Set the joint acceleration rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:28:42

Cartesian Acceleration

- Function:

```
AccL(R)
```

- Description: Set the Cartesian acceleration rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Acceleration rate. Value range: 1 -100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:28:53

Joint Speed

- Function:

```
SpeedJ(R)
```

- Description: Set the joint velocity rate . This command is valid only when the motion mode is MovJ, MovJIO, MovJR, or JointMovJ .
- Required parameter: Velocity rate. Value range: 1 - 100

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:20:30

Cartesian Speed

- Function:

```
SpeedL(R)
```

- Description: Set the Cartesian velocity rate. This command is valid only when the motion mode is MovL, MovLIO, MovLR, Jump, Arc, Circle.
- Required parameter: Velocity rate. Value range: 1 -100

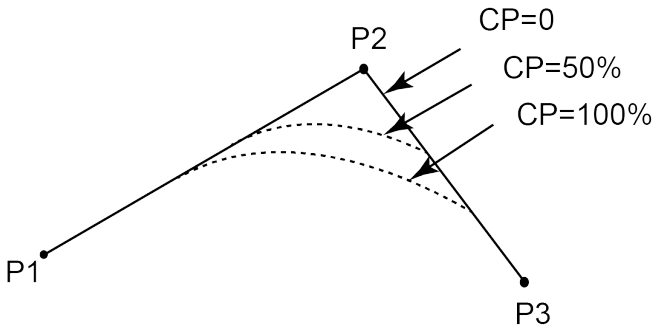
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:19:55

CP

- Function:

```
CP(R)
```

- Description: Set the continuous path rate. This command is invalid when the motion mode is Jump.
- Required parameter: Continuous path rate. Value range: 0-100



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 11:01:34

Synchronization

- Function:

```
Sync()
```

- Description: Whether to stop at this point.
- Required parameter: None.

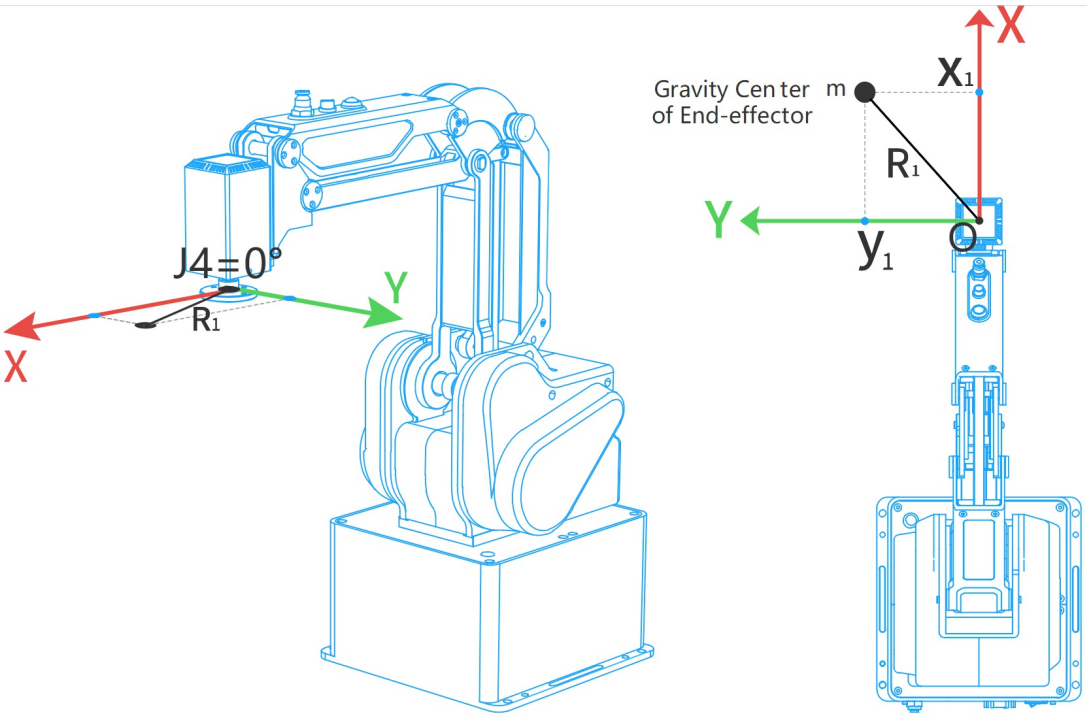
Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Set Load Parameters

- Function：

```
SetPayload(payload, {x, y}, index)
```

- Description: Set payload, X-axis offset, Y-axis offset and servo index.
- Required parameter:
  - payload: Payload. Value range: 0- 750. Unit: g
  - {x,y}: Offset in X-axis and Y-axis
- Optional parameter: index, servo parameter index。The default value range is 1 - 10.



Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

3 IO

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-25 09:35:43

DI

- Function:

```
DI(Index)
```

- Description: Get the status of the digital input port.
- Required parameter: Index，Digital input port. Value range: 1-18
- Return:
  - When an port is set in the DI function, **DI(index)** returns the status (ON/OFF) of this specified input port.
  - When there is no port in the DI function, **DI()** returns the status of all the input ports, which are saved in a table. For example, local di=(), the saving format is **{num = 24 value = {0x55, 0xAA, 0x52}}**, you can obtain the status of the specified input port with **di.num** and **di.value[n]**.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 11:01:33

DO

- Function:

```
DO(Index,ON/OFF)
```

- Description: Set the status of digital output port (Queue command).
- Required parameter：
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Queue command: When the robot system receives a command, this command will be pressed into the internal command queue. The robot system will execute commands in the order in which the commands were pressed into the queue.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 17:24:07

DOInstant

- Function:

```
DOInstant(Index,ON/OFF)
```

- Description: Set the status of digital output port (Immediate command).
- Required parameter：
  - Index: Digital output port. Value range: 1 - 18
  - ON/OFF: Status of the digital output port.

Immediate command: The robot system will process the command once received regardless of whether there is the rest commands processing or not in the current controller;

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-03-03 17:41:10

4 Program Managing Commands

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Motion command waiting

- Function:

```
Wait(time)
```

- Description: Set the delay time for robot motion commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Blocking instruction issuance

- Function:

```
Sleep(time)
```

- Description: Set the delay time for all commands.
- Required parameter: time, Delay time. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Pause program operation

- Function:

```
Pause()
```

- Description: Pause the running program. When the program runs to this command, robot pauses running and you need to click **Resume** on the DobotStudio2020 to recover the running.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Start timing

- Function:

```
ResetElapsedTime()
```

- Description: Start timing after all commands before this command are executed completely. Use in conjunction with ElapsedTime() command.
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Stop timing

- Function:

```
ElapsedTime()
```

- Description: Stop timing and return the time difference. Use in conjunction with ResetElapsedTime() command
- Required parameter: None
- Return: Time difference. Unit: ms

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Get current time

- Function:

```
Systime()
```

- Description: Get the current time
- Required parameter: None

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

5 Pose

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Get Cartesian coordinates

- Function:

```
GetPose()
```

- Description: Get the current pose of the robot under the Cartesian coordinate system. If you have set the User or Tool coordinate system, the current pose is under the current User or Tool coordinate system.
- Required parameter: None
- Return: Cartesian coordinate of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Get Joint coordinates

- Function:

```
GetAngle()
```

- Description: Get the current pose of the robot under the Joint coordinate system.
- Required parameter: None
- Return: Joint coordinates of the current pose.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Cartesian point offset

- Function:

```
local Offset={OffsetX, OffsetY, OffsetZ, OffsetR}
RelPoint(P, Offset)
```

- Description: Set the X, Y, Z.R axes offset under the Cartesian coordinate system to return a new Cartesian coordinate point. he robot can move to this point in all motion commands except JointMovJ.
- Required parameter:
  - P, Indicate the current Cartesian point, which is user-defined or obtained from the points list. Only Cartesian point is supported.
  - {OffsetX, OffsetY, OffsetZ, OffsetR}: X, Y, Z, R axes offset in the Cartesian coordinate system.
- Return: Cartesian point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Joint point offset

- Function:

```
local Offset={Offset1, Offset2, Offset3, Offset4}
RelJoint(P, Offset)
```

- Description: Set the joint offset in the Joint coordinate system to return a new joint point. The robot can move to this point only in JointMovJ command .
- Required parameter:
  - P, Indicate the current joint point, which is user-defined or obtained from the points list. Only joint point is supported.
  - {Offset1, Offset2, Offset3, Offset4}: J1 - J4 axes offset.
- Return: Joint point.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Cartesian point

- Function:

```
local P={coordinate = {x,y,z,r}, tool = 0, user = 0}
```

- Description: User-define a Cartesian point.
- Required parameter:
  - {x,y,z,r}: X, Y, Z, R axes coordinates.
  - tool: Tool coordinate system index. Value range: 0-9
  - user: User coordinate system index. Value range: 0-9

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Joint point

- Function:

```
local P={joint= {j1,j2,j3,j4}}
```

- Description: User-define a joint point.
- Required parameter: {j1,j2,j3,j4}，J1-J4 axes coordinates.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

6 TCP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-25 09:35:43

Create TCP

- Function:

```
Err, Socket = TCPCreate(IsServer, IP, Port)
```

- Description: Create a TCP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port：Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: TCP network is created successfully.
    - 1: TCP network is created failed.
  - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Establish TCP connection

- Function:

```
TCPStart(Socket, Timeout)
```

- Description: Establish TCP connection.
- Required parameter:
  - Socket: Socket object.
  - Timeout: Wait timeout. Unit: s. If Timeout is 0, the connection is still waiting. If not, after exceeding the timeout, the connection is exited.
- Return:
  - 0: TCP connection is successful.
  - 1: Input parameters are incorrect.
  - 2: Socket object is not found.
  - 3: Timeout setting is incorrect.
  - 4: If the robot is set as a client, it indicates that the connection is wrong. If the robot is set as a server, it indicates that receiving data is wrong.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Receive TCP data

- Function:

```
Err, RecBuf = TCPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or

  as a server receives data from a client .

- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Send TCP data

- Function:

```
TCPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Close TCP

- Function:

```
TCPDestroy(Socket)
```

- Description: Release a TCP network.
- Required parameter: Socket, Socket object.
- Return:
  - 0: Releasing TCP is successful.
  - 1: Releasing TCP is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

7 UDP

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-25 09:35:43

Create UDP

- Function:

```
Err, Socket = UDPCreate(IsServer, IP, Port)
```

- Description: Create a UDP network. Only support a single connection.
- Required parameter:
  - IsServer: Whether to create a server. false: Create a client; true: Create a server.
  - IP: IP address of the server, which is in the same network segment of the client without conflict.
  - Port：Server port. When the robot is set as a server, **port** cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail.
- Return:
  - Err:
    - 0: UDP network is created successfully.
    - 1: UDP network is created failed.
    - Socket: Socket object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Receive UDP data

- Function:

```
Err, RecBuf = UDPRead(Socket, Timeout, Type)
```

- Description: Robot as a client receives data from a server or

  as a server receives data from a client .

- Required parameter:
  - Socket: Socket object.
  - Timeout: Receiving timeout. Unit: s. If Timeout is 0 or is not set, this command is a block reading. Namely, the program will not continue to run until receiving data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether receiving data is complete.
  - Type: Buffer type. If Type is not set, the buffer format of RecBuf is a table. If Type is set to string, the buffer format is a string.
- Return:
  - Err:
    - 0: Receiving data is successful.
    - 1: Receiving data is failed.
  - Recbuf: Data buffer.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Send UDP data

- Function:

```
UDPWrite(Socket, Buf, Timeout)
```

- Description: Robot as a client sends data to a server or as a server sends data to a client.
- Required parameter:
  - Socket: Socket object.
  - Buf: Data sent by the robot.
  - Timeout: Timeout. Unit: s. If Timeout is 0 or not set, this command is a block reading. Namely, the program will not continue to run until sending data is complete. If not, after exceeding the timeout, the program will continue to run regardless of whether sending data is complete.
- Return:
  - 0: Sending data is successful.
  - 1: Sending data is failed.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

8 Modbus

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-25 09:35:43

Read the value from Modbus slave coil register address

- Function:

```
GetCoils(Addr, Count)
```

- Description: Read the value from Modbus slave coil register address.
- Required parameter:
  - Addr: Starting address of the coils. Value range: 0-4095
  - Count: Number of the coils to read. Value range: 0 to 4096- Addr
- Return: Return a table, each with the value 1 or 0, where the first value in the table corresponds to the coil value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Set the coil register in the Modbus slave

- Function:

```
SetCoils(Addr, Count, Table)
```

- Description: Set the coil register in the Modbus slave. This command is not supported when the coil register address is from 0 to 5.
- Required parameter:
  - Addr: Starting address of the coils to set. Value range: 6 - 4095
  - Count: Number of the coils to set. Value range: 0 to 4096- Addr

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Read the value from the Modbus slave discrete register address

- Function:

```
GetInBits(Addr, Count)
```

- Description: Read the value from the Modbus slave discrete register address.
- Required parameter:
  - Addr: Starting address of the discrete inputs to read. Value range: 0-4095
  - Count: Number of the discrete inputs to read. Value range: 0 to 4096- Addr
- Return: Return a table, each with the value 1 or 0, where the first value in the table corresponds to the discrete value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Read the value from the Modbus slave input register address

- Function:

```
GetInRegs(Addr, Count, Type)
```

- Description: Read the input register value with the specified data type from the Modbus slave.
- Required parameter:
  - Addr: Starting address of the input registers. Value range: 0 - 4095
  - Count: Number of the input registers to read. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer (two bytes, occupy one register) .
    - “U16”: Read 16-bit unsigned integer (two bytes, occupy one register).
    - “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers).
    - “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: Return a table, the first value in the table corresponds to the input register value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-03-03 17:41:10

Read the value from the Modbus slave holding register address

- Function:

```
GetHoldRegs(Addr, Count, Type)
```

- Description: Read the holding register value from the Modbus slave according to the specified data type
- Required parameter:
  - Addr: Starting address of the holding registers. Value range: 0 - 4095
  - Count: Number of the holding registers to read. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer (two bytes, occupy one register).
    - “U16”: Read 16-bit unsigned integer (two bytes, occupy one register).
    - “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers).
    - “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers).
- Return: Return a table, the first value in the table corresponds to the input register value at the starting address.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Set the holding register in the Modbus slave

- Function:

```
SetHoldRegs(Addr, Count, Table, Type)
```

- Description: Set the holding register in the Modbus slave
- Required parameter:
  - Addr: Starting address of the holding registers to set. Value range: 0 - 4095
  - Count：Number of the holding registers to set. Value range: 0 to 4096- Addr
  - Type: Data type.
    - Empty: Read 16-bit unsigned integer (two bytes, occupy one register) .
    - “U16”: Set 16-bit unsigned integer (two bytes, occupy one register).
    - “U32”: Set 32-bit unsigned integer (four bytes, occupy two registers).
    - “F32”: Set 32-bit single-precision floating-point number (four bytes, occupy two registers).
    - “F64”: Set 64-bit double-precision floating-point number (eight bytes, occupy four registers).

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

9 Conveyor Tracking

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Set conveyor number to create a tracing queue

- Function:

```
CnvVison(CnvID)
```

- Description: Set conveyor number to create a tracing queue.
- Required parameter: CnvID, Conveyor number. Only support single conveyor.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

Obtain status of the object

- Function:

```
GetCnvObject(CnvID, ObjID)
```

- Description: Obtain the information of the part on the conveyor to check whether the part is in the pickup area .
- Required parameter:
  - CnvID: Conveyor index.
  - ObjID: Part index.
- Return:
  - Part status: Whether there is a part. Value range: true or false
  - Part type
  - Part coordinate (x,y,r)

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Set X,Y axes offset under the set User coordinate system

- Function:

```
SetCnvPointOffset(OffsetX,OffsetY)
```

- Description: Set X,Y axes offset under the set User coordinate system.
- Required parameter:
  - OffsetX: X-axis offset.
  - OffsetY: Y-axis offset.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 17:27:19

Set time compensation

- Function:

```
SetCnvTimeCompensation(Time)
```

- Description: Set time compensation. This command is used for compensating the pick-up position offset in the moving direction of the conveyor which is caused by taking photos with a time delay.
- Required parameter: Time, time-offset. Unit: ms
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Synchronize the specified conveyor

- Function:

```
SyncCnv(CnvID)
```

- Description: Synchronize the specified conveyor. The motion commands used between SyncCnv(*CnvID*) and StopSyncCnv(*CnvID*) only support MovL command.
- Required parameter: CnvID, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 17:14:42

Stop synchronous conveyor

- Function:

```
StopSyncCnv(CnvID)
```

- Descriptioon: Stop synchronizing the conveyor. The other commands following this command will not be executed until this command running is completed.
- Required parameter: CnvID, Conveyor index.
- Return:
  - 0: No error
  - 1: Error

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54

10 Pallet

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Instantiate matrix pallet

- Function:

```
Pallet = MatrixPallet (Index,ID)
```

  Or:

```
local Option={IsUnstack= true, User= 1}
Pallet = MatrixPallet (Index,ID, Option)
```

- Description: Instantiate matrix pallet.
- Required parameter:
  - Index: Matrix pallet index.
  - ID: Unique identification of pallet
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Matrix pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Instantiate teaching pallet

- Function:

```
Pallet = TeachPallet (Index,ID, Option)
```

  Or:

```
local Option={IsUnstack= true, User= 1}
Pallet = TeachPallet (Index,ID, Option)
```

- Description: Instantiate teaching pallet.
- Required parameter:
  - Index: Teaching pallet index.
  - ID: Unique identification of pallet
- Optional parameter: {IsUnstack= true, User= 1}. You can double-click to insert the command with optional parameters.
  - IsUnstack: Stack mode. Value range: true or false. true: Dismantling mode . false: Assembly mode. If not set, the default is assembly mode
  - User: User coordinate system index. If not set, the default is User 0 coordinate system.
- Return: Teaching pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

Set the next stack index which is to be operated

- Function:

```
SetPartIndex(Pallet, Index)
```

- Description: Set the next stack index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next stack index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Get the current operated stack index

- Function

```
GetPartIndex(Pallet)
```

- Description: Get the current operated stack index.
- Required parameter: Pallet, Pallet object.
- Return: The current operated stack index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Set the next pallet layer index which is to be operated

- Function:

```
SetLayerIndex(Pallet, Index)
```

- Description: Set the next pallet layer index which is to be operated.
- Required parameter:
  - Pallet: Pallet object.
  - Index: The next pallet layer index. Initial value: 0

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Get the current pallet layer index

- Function:

```
GetLayerIndex(Pallet)
```

- Description: Get the current pallet layer index.
- Required parameter: Pallet, Pallet object.
- Return: The current pallet layer index.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Reset pallet

- Function:

```
Restet(Pallet)
```

- Description: Reset pallet.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:53

Check whether the stack assembly or dismantling is complete

- Function:

```
IsDone(Pallet)
```

- Description: Check whether the stack assembly or dismantling is complete.
- Required parameter: Pallet, Pallet object.
- Return:
  - true: Finished.
  - false: Un-finished.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

Release palletizing instance

- Function:

```
Release(Pallet)
```

- Description: Release palletizing instance.
- Required parameter: Pallet, Pallet object.

Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved，powered by GitbookRevision: 2021-01-26 16:58:54

The robot moves from the current position to the first stack position as the configured stack assembly path

- Function:

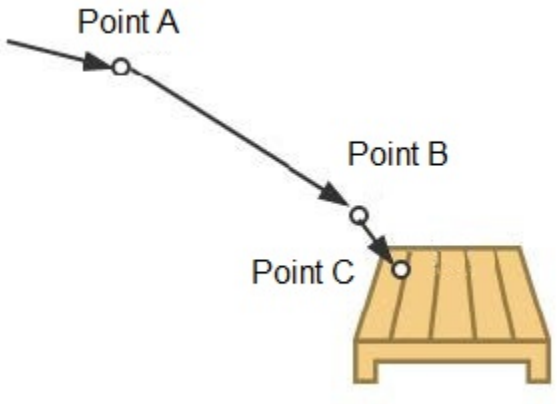
```
PalletMoveIn(Pallet)
```

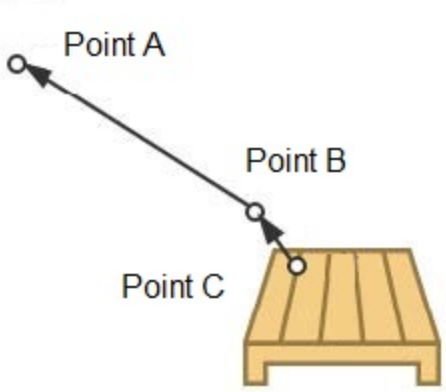
Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}
PalletMoveIn(Pallet, Option)
```

- Description: The robot moves from the current position to the first stack position as the configured stack assembly path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click to insert the command with optional parameters.
  - SpeedAB: Velocity rate when the robot moves from the transition point to the preparation point. Value range: 1-100
  - SpeedBC: Velocity rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
  - AccAB: Acceleration rate when the robot moves from the transition point to the preparation point. Value range: 1-100
  - AccBC: Acceleration rate when the robot moves from the preparation point to the first stack point. Value range: 1-100
  - CP: Continuous path rate. Value range: 0-100

  The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:53

The robot moves from the current position to the transition point as the configured stack dismantling path

- Function:

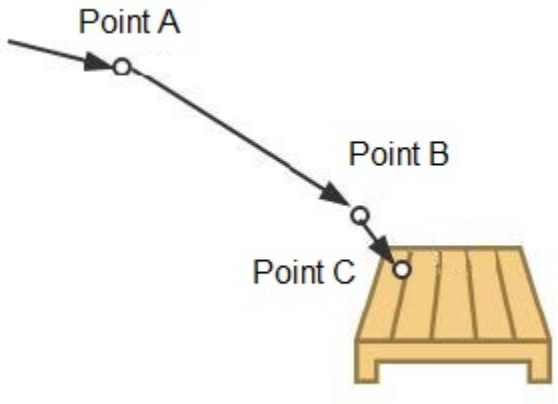
```
PalletMoveOut(Pallet)
```

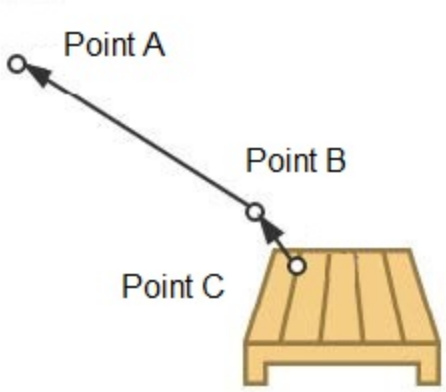
Or:

```
local Option={SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}
PalletMoveOut(Pallet, Option)
```

- Description: The robot moves from the current position to the transition point as the configured stack dismantling path.
- Required parameter: Pallet, Pallet object.
- Optional parameter: {SpeedAB=20, SpeedBC=30, AccAB=20, AccBC=10, CP=20}. You can double-click to insert the command with optional parameters.
  - SpeedAB: Velocity rate when the robot moves from the preparation point to the transition point. Value range: 1-100
  - SpeedBC: Velocity rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
  - AccAB: Acceleration rate when the robot moves from the preparation point to the transition point. Value range: 1-100
  - AccBC: Acceleration rate when the robot moves from the first stack point to the preparation point. Value range: 1-100
  - CP: Continuous path rate. Value range: 0-100

  The stack assembly path and dismantling path are shown as follows. Point A is the transition point, which is fixed or varies with the pallet layer. Point B is the preparation point which is calculated by the target point and the set offset. Point C is the first stack point.





Copyright © ShenZhen Yuejiang Technology Co., Ltd 2020 all right reserved, powered by GitbookRevision: 2021-01-26 16:58:54