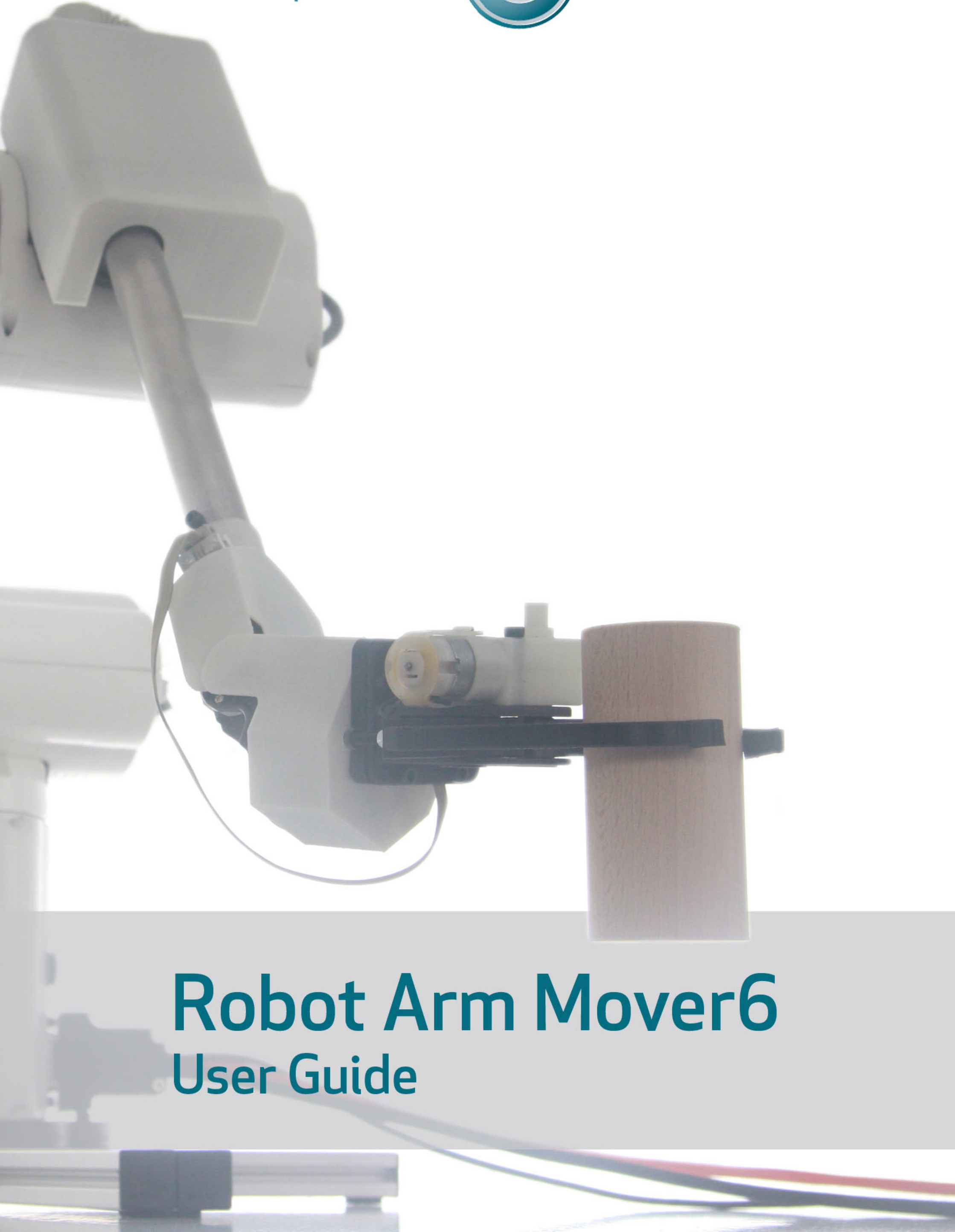
commonplace robotics GmbH



Robot Arm Mover6

User Guide

User Guide Mover6
Version 2014/07 (SW V902-06-016, HWE 2MV11 HWM V02 DOC V01)

Commonplace Robotics GmbH
Osterfeldstr. 1
D-49326 Melle
Germany
++49 5429 / 374983-4
info@cpr-robots.com
www.cpr-robots.com

The CE sign confirms that this product meets the requirements of the directive 2004/108/EC (EMC) and 2002/95/EC (RohS). The according documentation is deposited at the manufacturer.

Help to save our environment! When reaching the end of livetime, do not throw the device into the garbage, but bring it to a public recycling place.

Inhalt

1. Safety Instructions

- The Mover6 is designed to be used by adolescents and adults in edutainment and research areas.

- The robot must not be used in industrial production facilities of in other continuous operation scenarios.

- Do not operate the robot when it is unattended.

- The gripping area exposes the risk of bruises to the fingers, especially when using the parallel gripper. Keep off during operation! The robot must not operate unattended!

- The robot must not be used by children without the supervision by adults. Not suited for children below 3 years. Small parts pose choking hazards.

- Sharp or otherwise dangerous tools or parts must not be mounted on the robot without according safety measures, e.g. housings. This especially comprises milling applications or laser sources.

- Take care for a stable stand of the robot.

- The robot has been designed for indoor use. They do have to be protected of humidity, dust or excessive solar radiation.

- Do not open the product.

- It is necessary to backup important data before the installation of the CPRog software.

- Switch of the power supply when the robot is not in use or unattended.

2. Introduction

2.1 Product

The Commonplace Robotics Mover6 is a six axis robot for the use in education, entertainment and research environments. The robot can lift a payload up to 500 g with a reach of 550 mm plus gripper.

Pic. 1: Robot arm and the necessary components (here with the Mover4 robot arm)

The modular set up of the robot connects four servo motor joint modules and two smart servos. The structure consists of engineering plastic and aluminum.

The front end of the robot, the flange, allows mounting a gripper or another tool. The robot base has to be fastened on a stand or a table. At the robot base one plug connect with power and the PC, another provides digital IO access.

The cable loom is connected with a 12V/5A DC power supply. The USB adapter connects the control PC with the robots internal communication bus. The CPRog software allows to control and program the robot.

2.2 Specifications

Nr of Joints	6 (4 servo joints and 2 smart servos)
Power Supply	12V DC at max 5 A
Power Consumption	Paused: 0,5 A / in motion: < 2.5 A. Fuse with 2.5 A in base
Communication	CAN at 500 kBit/s
Reach	455 mm plus gripper
Payload	200 g
Inputs / Outputs	At the base: 3 digital outputs (5V / 25 mA) and 4 digital Inputs (5V) via D-Sub 9 poles female. or, with 24V-DIO Extension: 3 relay out (24V/1A) and 4 digital in (24V) via D-Sub 9 poles male. At the flange: 2 digital Outputs (5V/ 25 mA) und 12V/0,5A supply via Harting SEK 6 poles.
Communication	Definition of position setpoints for all four joints with 20 Hz cycle. Reading of the current position and the motor current.

2.3 System Requirements

The control the robot with the CPRog programming environment a PC with Windows operating system is necessary (minimum values):

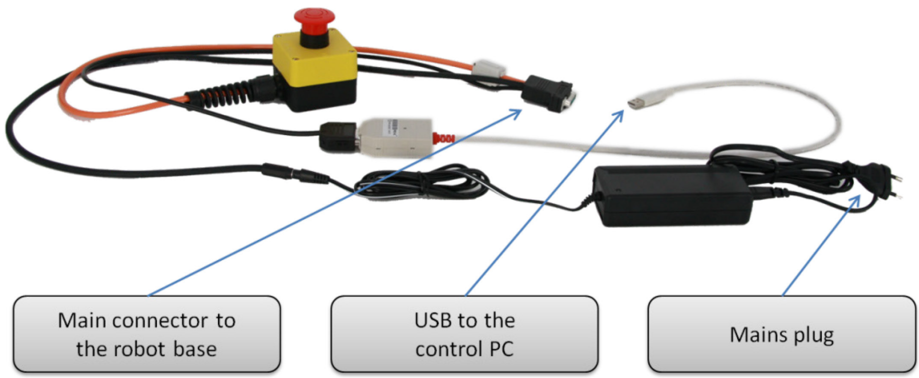
- Operating System: Windows 7 (32 or 64 Bit) or higher.
- .NET-Framework 3.5 or higher and DirectX in Version 9.0c
- Processor: 1.6-GHz-Pentium
- RAM: 1 GB
- Up to 50 MB disk space
- 1 free USB port for the communication with the robot, another one for joypad operation

3. Installation

3.1 Setting up the Robot

The robot has to be placed on a table or stand so that he cannot tumble down.

3.2 Connecting the Power Supply and the USB Adapter

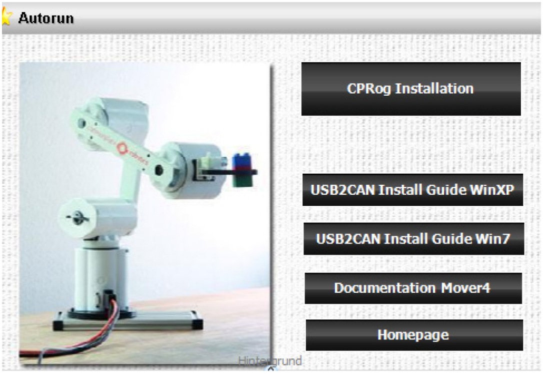


Pic. 2: Cable loom and USB adapter

The cable loom connects the power supply, the USB adapter and the robot as depicted above. All connectors fit only one way. The USB adapter may be white or black, depending on the current type.

When plugging in the USB2CAN adapter, a driver installation assistant may appear. Please refer to section 3.4 for installation guidelines.

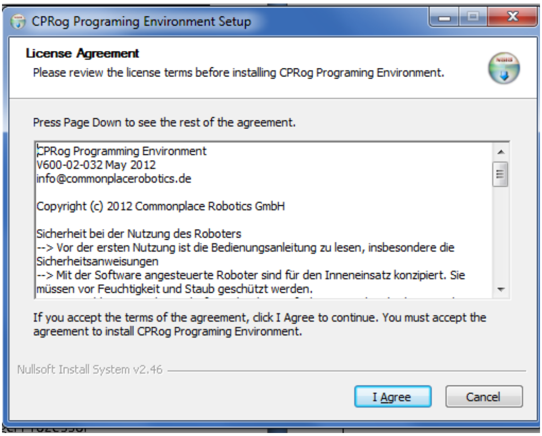
3.3 Installation of the CPRog Software



Insert the CPRog CD into the drive.

Depending on your systems configuration the CD menu will open automatically, or you have to start it manually:
D:\.autorun\autorun.exe

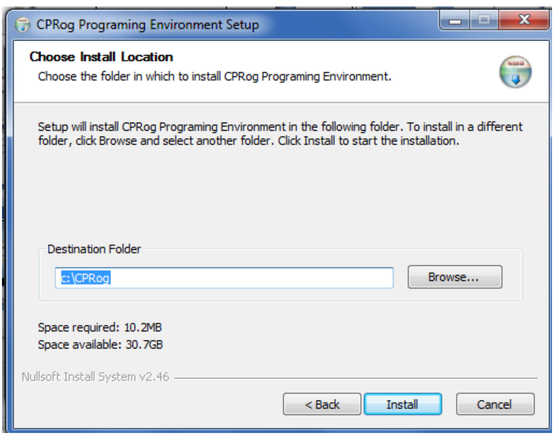
Choose the first button „CPRog Installation"



Maybe you have to allow changes to be made on your system.

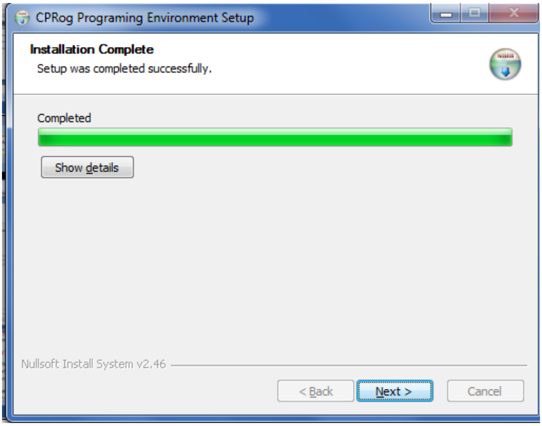
After the installer has started you need to choose between English and German as language.

Then you need to confirm the license agreement.



In the next step you can choose where to install CPRog. The recommended directory is `c:\CPRog`.

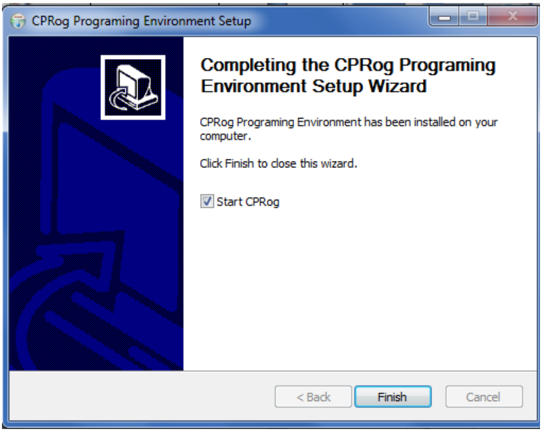
When installing in a Windows program directory like `c:\Programme` it is possible that CPRog can only be started as administrator.



The installation normally takes only a few seconds.

The installer checks, if DirectX 9.5 is installed. If not it will install theses libraries from the CD or via download. This will take some minutes.

The installation of DirectX 9.5 is necessary even if e.g. DirectX 11 is already installed.



When finishing the installation you can choose to directly start CPRog.

Now you can start CPRog using the link on the desktop or via the start menu.

Installation Failure: The installation assistant checks, if all necessary extensions are available, especially the .NET-Framework and DirectX 9.5. If this is not the case an error message will appear. DirectX is installed automatically, but the .NET-Framework has to be installed manually:

Search in the net for “Microsoft .NET download” and install

3.4 Installation of the Driver for the USB Adapter

The robot is delivered with the PCAN USB driver from www.peak-system.com. To run the adapter the according driver has to be installed. This can be done from the CPRog installation CD (button "Install USB-CAN adapter"), or using the manufacturers installation CD.

After starting the installation you need to

- accept the license agreement and
- set the installation folder.

In the next step please check the PCAN-USB device and the PCAN-View CAN-Bus Monitor as shown on the following picture for installation.

Choose for installation:

- PCAN-USB
- PCAN-View

The PCAN-View monitor allows to check if the adapter is connected correctly.

3.5 Licensing

The CPRog software needs a license key to start. This key is in most cases already integrated into the installation version of CPRog. Nothing has to be done.

If it is not, e.g . because you have installed a demo version, please see the following information. The key, an XML file, has to be stored in the directory

```
\CPRog\Data\License
```

The CPRog installer copies the key Demo.xml to the directory. With this key CPRog starts in demo mode. The runtime is limited to 10 minutes; afterwards you need to restart CPRog.

When you do need a license key, please send a short mail with the following information to licensing@cpr-robots.com:

- Your Name or the name of the responsible contact person
- Your companies / organizations name

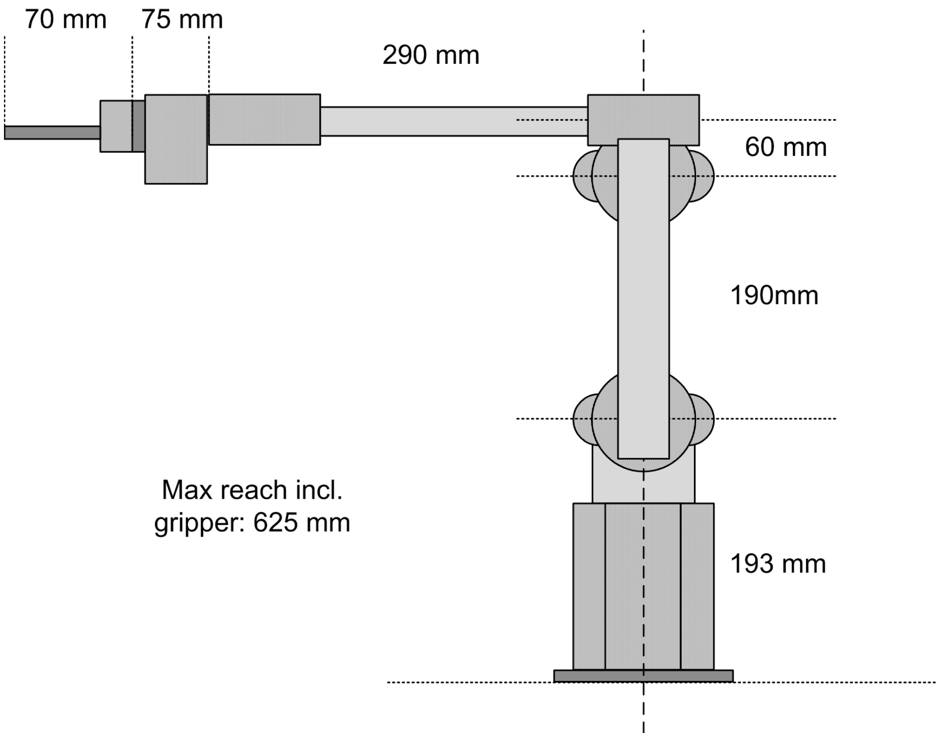
We will send you your key via mail. After starting CPRog your licensing data are shown in the main frame and can be shown complete with the Programbutton/License.

Please do not change the content of the license file, this will render it invalid!

A standard license allows the installation and use of CPRog on an arbitrary number of computers in the company or organization of the license holder.

4. Robot Arm Mover6

4.1 Geometry



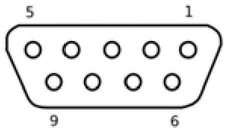
Pic. 3: Mover6 simplified side view

The picture shows the joints length of the Mover6 robot arm. Including the gripper the arm reaches 625 mm.

CPRog offers functionalities to limit the workspace, see section 7.2.

4.2 Digital Inputs and Outputs

The robot base shows a D-Sub plug with several digital inputs and outputs. The pin assignment for the versions „Standard“ and „24V DIO“ are shown below.



D-Sub Female: View from the outside on the pins. Male according mirrored.

Standard: 5V DIO	Extension: 24V DIO
Type: Female	Type: Male
Pin 1: 5V	Pin 1: Out1 24V
Pin 2: In 3	Pin 2: Out2 24V
Pin 3: In 1	Pin 3: Out3 24V
Pin 4: Out 3	Pin 4: In1
Pin 5: Out 1	Pin 5: In2
Pin 6: GND	Pin 6: In3
Pin 7: In 4	Pin 7: In4
Pin 8: In 2	Pin 8: GND
Pin 9: Out4	Pin 9: 24V Supply

Pic. 4: Pin assignment at the base digital IO plug

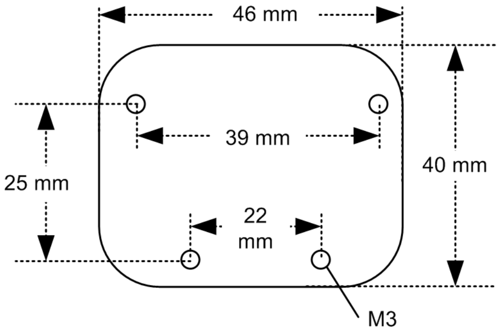
The Standard robot is set up for communication on TTL level (5V).

The 24V-DIO-Extension provides relays and 24V digital inputs, allowing to communicate with a PLC on 24V level.The outputs are able to provide 1A each, forwarding the external supply voltage from pin 9. The inputs are 24V.

Two further digital outputs can be found on the flange of the robot, see the next section.

4.3 Mounting Flange for the Gripper

The robot flange offers 10 M3 threads to connect gripper or other devices. A plug allows electrical connectivity.
Six of the threads are placed on a 15 mm reference circle; the remaining 4 M3 threads are located as shown in the drawing. The plug is a Harting product SEK male 6 poles:

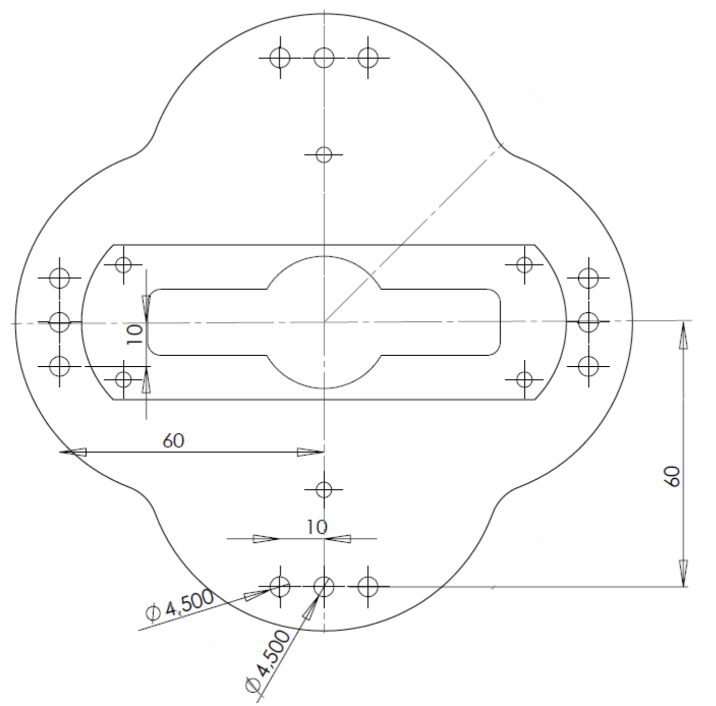


Pin1:	12V, max 0.5A
Pin2	GND
Pin3:	Out 11
Pin4:	DOut 12
Pin5:	not connected
Pin 6:	not connected

Pic. 5: Drawing of the flange

4.4 Drawing of the Robot Base

The robot can be mounted using the M4 through holes.

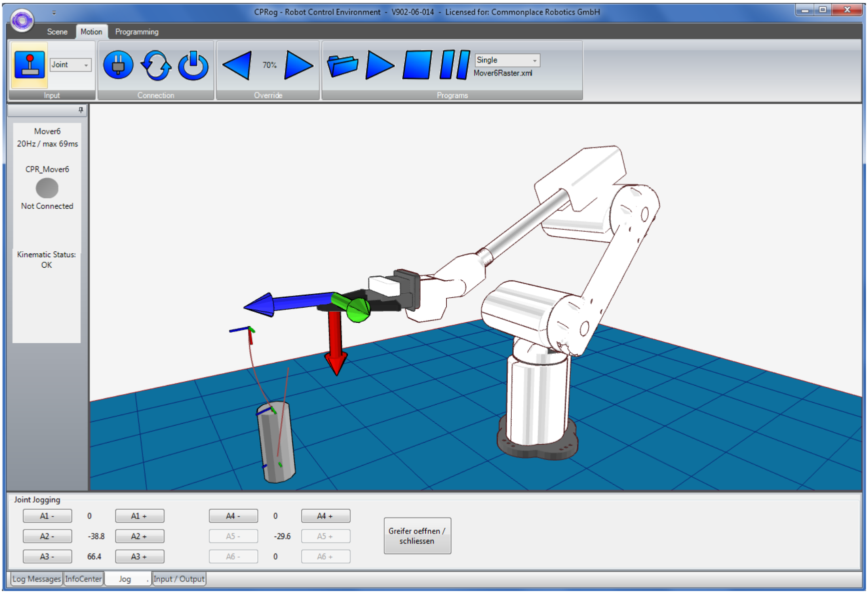


Pic. 6: Drawing of the robot base plate

5. Moving the Robot Arm with CPRog

5.1 Introduction

The CPRog programming environment allows to control and program the Mover robot. Both online and offline working is possible.



Pic. 7: CPRog User Interface

In the upper area the three ribbons “Scene”, “Motion” and “Programming” provide access to the main functionalities.

On the left corner information on the robots current state are provided.
On the lower side four sections are found:

- „Log Message“: Messages regarding the programs state
- „Info Center“: Shows joint values, the Cartesian position and the motor currents
- „Jog“: Buttons to move the robot
- „Input / Output“: View and set digital IOs

5.2 Navigation using the Mouse

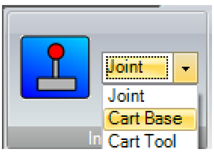
A 3 button mouse is recommended to navigate in the CPRog 3D environment:

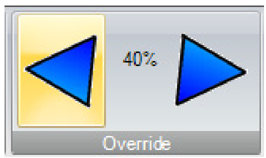
- Left button: Selection of robots and other objects
- Middle button: Navigation in the scene
    - Rotate: drag the mouse while holding the middle button
    - Pan: drag the mouse while holding the middle button and pressing down the CTRL-key
    - Zoom: Drag the mouse while holding the middle button and pressing down the SHIFT-key (zooms to the center of the scene), or turn the mouse wheel (zooms to the current cursor position)
- Right button: Opens the context menu

Alternatively the function of the left mouse button can be changed in the upper menu area at Scene/Navigation. Possibilities are Selection, Rotation, Panning or Zooming.

5.3 Moving the Robot with Joypad and Buttons

The robot can be moved (or "jogged") manually, as long as no program is running. The main elements are the button to connect the joypad, the combobox to choose the motion type and the override.



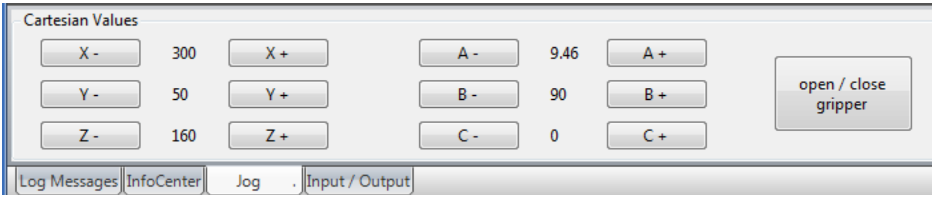


Pic. 8: Elements to jog the robot

When pressing the joystick button CPRog connects with a joypad. Nevertheless, CPRog tries to connect on start up. When the connection succeeded a green ok sign is shown on the joystick button.
The device must have the type "Joystick" or "Gamepad". Further information on the connection trial is provided in the log window.

The „Joint“ mode allows to turn the single robot axis from A1 to A4. In “Cartesian” mode the robot moves in straight lines following the x, y and z coordinate axis. The rotation is defined with the B commands. In „Cartesian Tool” mode the robot moves aligned to the current tool coordinate system.

The override scales the motion speed between 0 and 100%.



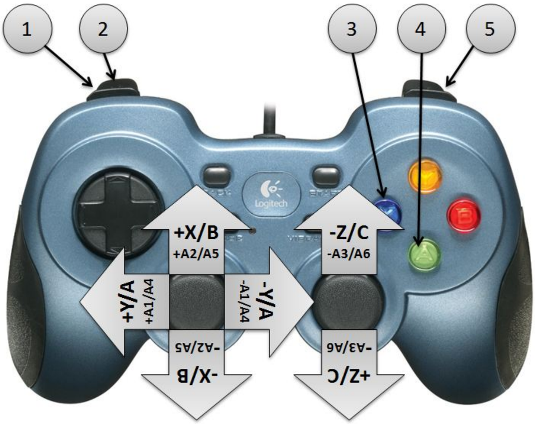
Pic. 9: Buttons to jog the robot in Cartesian mode. In joint mode the buttons change to the axis A1 to A4.

Software and Hardware End Stops: The software limits the joints rotation when it reaches the software stops defined in the robot config file. In joint 1 and joint 4 there are additional hardware end stops.

While jogging in Cartesian mode virtual walls can be switched on to limit the motion and thereby avoiding e.g. collisions. See section 7 for details.

When the virtual walls are active the robot blocks further motions and generates warning sounds when leaving the allowed area. These restrictions do not hold in joint mode.

The most convenient way to move the robot is with a connected joypad, the picture below shows the assignment of keys.



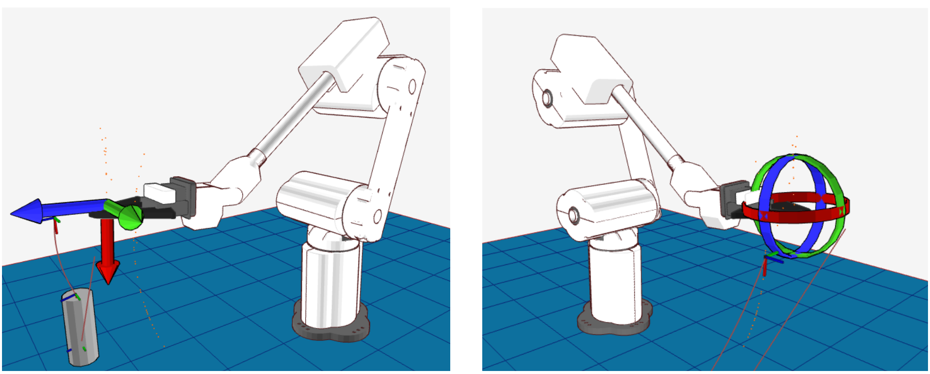
Reference:

1. Change motion mode
2. Change active robot
3. Open / close gripper
4. Record a motion point
5. Change button assignment:
   when pressed it is not +X, but +B

Pic. 10: Assignment of keys for the joypad. Upper markings for Cartesian mode, lower for joint mode.

5.4 Moving the Robot using the Graphics

An alternative to the joypad is to drag the robot in the graphical 3D environment. When selecting a joint of the robot with the left mouse button, the joint outlines blink red. When selecting the joint and moving the mouse with pressed left mouse button this joint will rotate forward or backward, depending on the mouse motion.



Pic. 10: Joint and coordinate system outlines after selecting with the mouse.

The motions are possible in simulation and with the real robot. They are also scaled with the override buttons.

5.5 Connect with the Hardware

The real robot can be controlled in the same way as the simulated one, only the hardware has to be connected before: connect, reset errors and enable motors.

Prerequisites are that the robot is connected via the USB-CAN adapter and the robots supply is running (plugged in, switched on and emergency button released).

Pic. 11: Buttons to connect to the hardware, reset the errors and enable the motors

Step 1: Connect with the hardware. This step initializes the USB-CAN interface.
The LED on the left side of CPRog changes color from grey to red. Below the LED several error messages are displayed.

Step 2: Reset the errors. This button resets the error memory of the joint module controller in the robot. The joint values are copied from the real robot to the simulation environment.

The 3D visualization of the robot has to match the current pose of the real robot now. This has to be verified every time the errors are reset!
The LED stays red. The error messages get cleared, only "Motors not enabled" is remaining.
If further error messages remain visible try again and see section 10.3 for possible approaches.

Step 3: Enable the motors. Now the robot can be jogged as described above.
The LED is green now.

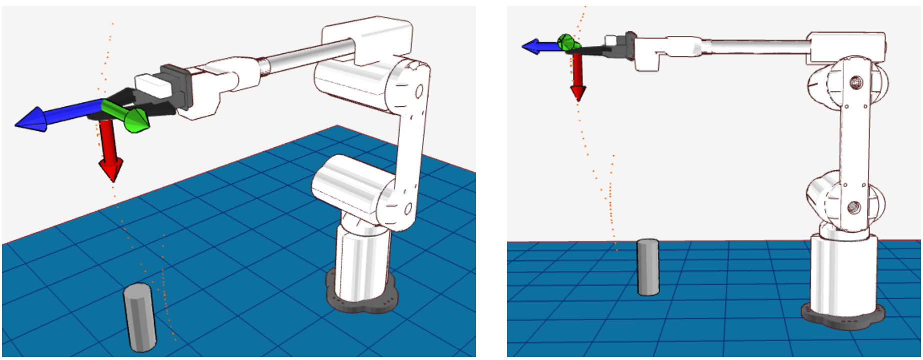
5.6 Reset the Zero Points of the Joints

The joint modules 1 to 4 keep track of their position using incremental encoder. The joint electronics stores these positions in their non volatile memory, so that the robot is aware of his current position after being switched on. Joints 5 and 6 provide absolute position encoder, there is no need to store the current position.

It may happen that the stored values are not correct, then the joints need to be calibrated. Reasons may be:

- Turning the joints while the robot is without power supply
- Stopping the robot with the emergency stop during a motion
- ...

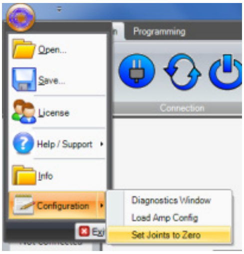
To calibrate the robot it has to be jogged into the zero position using the joint mode. The sticker on the axis provide assistance, their triangles should be aligned with the structures marks.



Pic. 11: Zero Position of the Mover6

If it is, e.g. due to the software end stops, not possible to reach the zero position, it helps to reset the joints to zero in an intermediate position.

The calibration defines the current hardware joint values of joint 1 to 4 as zero position.



When the robot is in the correct position press “LogoCircle/Configuration/SetJointsToZero” in the upper left menu. Confirm the following dialog with „OK“.

This function takes back the motor enabling, so the error reset and enable motor buttons have to be pressed.

The precision of the calibration is vital for replaying old programs. If theses old programs (recorded before the calibration) show precise motions close to the environment they have to be replayed slowly the first time. It may be necessary to re-teach positions.

6. Programming the Robot Arm with CPRog

6.1 Program Elements

A Mover6 robot program is build out of the following commands:

	Lin	The robot moves on a straight line from the current position to the target position. The velocity is defined in mm/s.
	Joint	Interpolates the axis from the current position to the target position in joint coordinates. Velocity is defined as percentage of the maximum joint rotational velocity.
	Relative	A linear or joint motion with a defined distance starting at the current position. Velocity in mm/s or percent.
	Wait	Waits for a time span defined in ms.
	DigitalOut	Sets a digital output. The button on the left side records the current gripper state.
	Loop	Iterates the included commands until a digital input is set or the defined number of iterations is reached.
	If-Then-Else	Executes two different command lines depending on the state of a digital input.
	ExternalMotion	Passes control to an external program connected via sockets, e.g. a vision system. See section 8.1 for details.
	Sub	Allows to call sub programs to structure longer tasks.

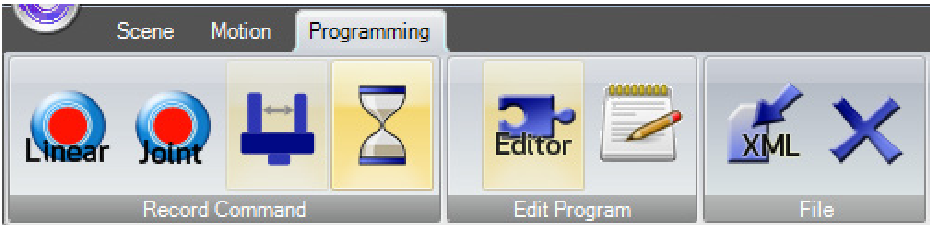
The complete language specification is found in section 11.

6.2 Recording a Program using the 3D Interface

Using the 3D screen a program can be recorded using the joypad or the buttons.

Joypad: Every time the A button (or button 2) on the joypad is pressed a linear command to the current robot position is recorded. If necessary a command to open or close the gripper is added, together with a short break. A sound indicates that the point was recorded.

Program Buttons: If there are still commands in the memory, they can be deleted using the Delete button. Afterwards linear, joint, gripper / DOut and break commands can be recorded.



Pic. 12: Buttons for program generation

6.3 Saving and Loading of a Program

With the button „Save Program" the commands are saved in an XML file in the directory

```
c:\CPRog\Data\Programs\
```

or in any other user defined folder. The button to load a program is found in the Motion/Programs area.

6.4 Replay of a Program

The buttons in the area Motion/Programs allow to replay, pause and stop a program. The replay can be done in single, repeat or step mode.

Hitting the space bar also starts and stops a robot program.

6.5 Editing a Program with GraphEdit

The button Programming/Editor opens a graphical program editor. This editor allows in a puzzle visualization to adapt and create programs.

Pic. 13: Graphical program editor

In the upper left area common elements are found, Undo/Redo and Load/Save. Two further buttons need explanation:

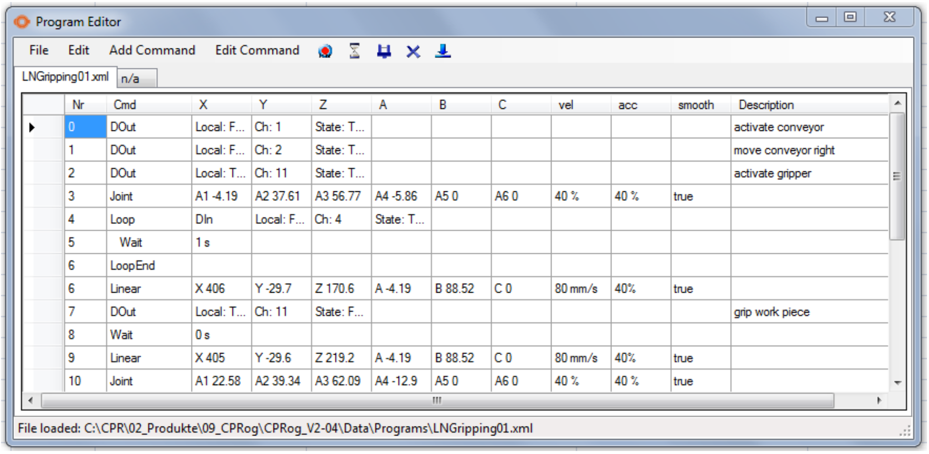
- The dust bin deletes the selected puzzle piece
- The light bulb updates the program on the current robot to the updated version. It also saves the program.

New commands can be dragged from the shelf on the upper right corner. They are initialized with the current robot position.

Commands can be copied by dragging them with the CTRL key pressed.

6.6 Editing a Program with TextEdit

Using the button beneath GraphEdit opens the TextEdit Program editor. This editor is more usable for bigger programs and provides more information.



Nr	Cmd	X	Y	Z	A	B	C	vel	acc	smooth	Description
0	DOut	Local: F...	Ch: 1	State: T...							activate conveyor
1	DOut	Local: F...	Ch: 2	State: T...							move conveyor right
2	DOut	Local: T...	Ch: 11	State: T...							activate gripper
3	Joint	A1 -4.19	A2 37.61	A3 56.77	A4 -5.86	A5 0	A6 0	40 %	40 %	true	
4	Loop	DIn	Local: F...	Ch: 4	State: T...						
5	Wait	1 s									
6	LoopEnd										
6	Linear	X 406	Y -29.7	Z 170.6	A -4.19	B 88.52	C 0	80 mm/s	40%	true	
7	DOut	Local: T...	Ch: 11	State: F...							grip work piece
8	Wait	0 s									
9	Linear	X 405	Y -29.6	Z 219.2	A -4.19	B 88.52	C 0	80 mm/s	40%	true	
10	Joint	A1 22.58	A2 39.34	A3 62.09	A4 -12.9	A5 0	A6 0	40 %	40 %	true	

Pic. 14: TextEdit program editor

The five picture buttons allow direct interaction:

- Recording a motion command (lin or joint according to the robots current motion type) with the current robot position
- Recording a break
- Recording a DOut-command
- Deleting the selected line
- Setting the selected line as starting point for the next program start

All fields of the commands can be edited directly to e.g. change the velocity. Changes are accepted when clicking on another line or field. A line can be copied with Ctrl-C (or in the Edit menu), Ctrl-V pastes the command again.

All new commands are inserted in the line above the selected line.

When choosing Sync from the File menu the current program is written and loaded by the robot. So the robot is again in synch with the text editor.

7. Configuration

7.1 Application Configuration

The application configuration can be done by adapting the parameters in different XML files.

7.1.1 Startup Configuration

The file c:\CPRog\CPRFrontend.xml allows to set the language and the start project.
<Program Language="DE"/> Possible choices are DE and EN.
<StartUp Project="..\..\Data\Projects\Mover6.prj" />

7.1.2 Project File

The project file can be found in c:\CPRog\Data\Projects\
The initial position of the view camera is defined in the line:
<Camera xPos="683" yPos="826" zPos="306" yRot="-10.1" zRot="-161"/>

Offset and Tool of the robot can be changed:
<Robot Name="Mover6" Type="CPR_Mover6" Homepos = "0.0 -11.0 46.0 0.0 -37.0 0.0" OffsetX="0" OffsetY="0" OffsetZ ="10" OffsetRX="0" OffsetRY="0" OffsetRZ="0" Parent="-1" Tool="TwoFingerGripper.xml" FlipMesh="True"/>

The initial robot program is specified:
<Program File="\Programs\6AxisTest.xml" Override="70" Repeat="true"/>

Objects in the Virtual Environment can be added, static or active:
<StaticSceneObject Name="WorkPiece" Geometry="zylinder.obj" Gravity="true" Sensable="false" Process="None" OffsetX="350" OffsetY="-150" OffsetZ ="0" OffsetRX="0" OffsetRY="0" OffsetRZ="0.0" FlipMesh="True"/>

Please refer to the examples to get to know different combinations. The CAD files have to be in .stl (ASCII and binary) or AliasWavefront .obj format, units are mm.

7.2 Robot Configuration

For each robot in the CPRog simulation a XML config file exists, e.g.

C:\CPRog\Data\Robots\CPRMover6\CPRMover6.xml

Definition for the velocities in Jog mode:
<Velocities JogCart="10.0" JogOri="1.5" JogJoint="1.0"/>

A virtual cell can be defined as a safety space the robot cannot leave in Cartesian mode:
<VirtualBox active="false" xMin="450.0" xMax="1700.0" yMin="-1300.0" yMax ="1300.0" zMin ="800.0" zMax="1700.0" bMin="45.0" bMax="135.0"/>
The cell walls are not active in joint mode.

Also the control parameter of the joint controller can be adapted, e.g. the position and velocity PID settings. This can be done with the button „Load Amp Config" in the program start menu. This button reads a parameter file and transfers the data to the connected robot arm.

Further information on the joint controller and the CAN protocol can be found in the servo controller documentation.
Get in touch with CPR to get this documentation: info@cpr-robots.com

Changing the joint controller configuration requires knowledge of the system. The robot may move in unexpected ways if ill-fitted parameters are uploaded. The robot may damage itself and its environment. Any warranty is void when uploading parameter files not generated by CPR!

8. Interfacing

8.1 Command „External Motion“

Using the „External Motion“ command is an easy way to integrate external control, e.g. a vision system.

<ExMotion Nr="1" IP="127.0.0.1" Port="1234" Scale="10" Descr="" />

When the robot reaches this command in program replay he hands control over to the external program. He gets connected defined by IP address and port. The scale parameter adapts the velocity.

The steps are:

- When starting the robot program CPRog connects as client at the server at IP:Port. CPRog needs to have the necessary rights to do so!
- When the ExternalMotion command is reached CPRog sends the current position in 20 Hz cycles to the server and expects the XYZABC set point velocity as answer.
  - From the robot: `„Pos 234.0 123.1 987.3 0.0 90.0 0.0 8\n“`
    - `Pos:` Keyword
    - `234.0 123.1 …` Position as xyzabc
    - `8` An int with the binary coded Din at the base
  - Answer: `„Dir 95.0 14.1 -30.0 0.0 0.0 0.0 open 8\n“`
    - `Dir` Keyword
    - `95.0 14.1 …` Relative velocites [-100.0 .. 100.0]
    - `open` Gripper state, „open“ or „closed“
    - `8` An int with the binary coded Dout at the robot base

  Both messages have to end with the character Newline („\n“ or 0x0A).

- When finished the external control sends `„Done \n“` (with blank). Then the robot proceeds in replaying the program.

The parameter `Scale` adapts the velocity. When the last answer from the server is older then 500 ms the robot does not move.

A Java/Eclipse example program can be found at

https://www.github.com/CPR-Robots

It demonstrates the set up of the communication an approaching a position.

8.2 ROS – Robot Operating System

The Willow Garage Robot Operating System (see www.ros.org) is wide spread in the research community, especially when dealing with service robots. For the Mover6 packages are available to connect with the hardware and to command the robot by joint and position messages. Also a teleoperation node is available for demonstration purposes.

The packages can be downloaded at

https://www.github.com/CPR-Robots

8.3 Direct Access using the CAN Protocol

It is also possible to directly access the Mover6 on CAN field bus level, using a custom control software. The necessary protocol specification is found in the following section.

9. CAN Protocol Specification

Please refer also to the C++ implementations found on www.githu.com/CPR-Robots for code examples. The Mover6 has the standard CAN IDs 0x10, 0x20, 0x30, 0x40, 0x50 and 0x60 for the four joint modules.

9.1 Commands

It is important to send the commands with the correct length, otherwise the board controller will ignore them!

Command	CAN Command	Comments
Reset Error	0x01 0x06	Sets Error Code to 0x04 (Motor not enabled)
SetPosition	0x01 0x08	Send posH in byte 3, posL in byte 4 Typical: 0x01 0x08 0x7D 0x00 To ensure data integrity this command has to be send twice within the time of 0.5 s to take effect.
Enable Motor	0x01 0x09	Also resets errors
Disable Motor	0x01 0x0A	
Set DOut 1	0x01 0x20	Third byte 0x01 for true, 0x00 for false
Set DOut 2	0x01 0x21	See above. DOut 3 and 4 via 0x22 und 0x23
Set parameter	0x02 …	See following table for parameter and standard values
Get Parameter	0x03 0x50	Answer: 8 bytes with 0x50, maxMissedCom-H, maxMissedCom-L, maxLag-H, maxLag-L, maxCurrent, maxTorque, torqueBias
	0x03 0x51	Answer: parameter of the position control loop 0x51, (P*1000)-H, (P*1000)-L, (I*10000)-H, (I*10000)-L, (D*1000)-H, ('D*1000)-L, antiWindUp
	0x03 0x52	Answer: parameter of the velocity control loop 0x51, (P*1000)-H, (P*1000)-L, (I*10000)-H, (I*10000)-L, (D*1000)-H, ('D*1000)-L, antiWindUp
	0x03 0x54	Answer: Working hours. 0x54, workinghours-H, workinghours-L, workingMinutes, workingSeconds, firmwareVersion1, firmwareVersion2, 0x00
Set Joint Pos	0x04	See section 9.3
Set Velocity	0x05	See section 9.4

9.2 Parameter

The Operation can be adopted by a variety of parameters listed below. These parameters can be set using the 0x02 command, see section before. An exemplary command can be to set the maximal current is: 0x02 0x32 0x70 sets the allowed current to 127.

The parameter are set for the board. A two motor board does have only one set of parameter.

Some parameter changes require a reboot of the board to take place.

Parameter	CAN Command	Standard Value	Comment
maxMissedCom	0x02 0x30 data-H data-L	1000	
maxLag	0x02 0x31 data-H data-L	1200	Distance in encoder tics
maxCurrent	0x02 0x32 data	0x80	0-255
Position control – proportional	0x02 0x40 pos-H pos-L	1.0	
Position control – integral	0x02 0x41 data-H data-L	0.0	
Position control – differential	0x02 0x42 data-H data-L	0.0	
Position control – anti-windUp	0x02 0x43 data	50.0	
Velocity control – proportional	0x02 0x44 pos-H pos-L	0.5	
Velocity control – integral	0x02 0x45 data-H data-L	0.0	
Velocity control – differential	0x02 0x46 data-H data-L	0.0	
Velocity control – anti-windUp	0x02 0x47 data	20	

9.3 SetJoints Command

To set a new joint value messages with the first byte 0x04 are used:

Message ID:	board id
Protocol:	Command Velocity PositionH PositionL TimeStamp
Command:	0x04
Velocity:	not used
Position:	16 bit unsigned long from 0 to 65535.
TimeStamp:	Arbitrary number, the module will copy this code in the answer
Example:	0x20 - 0x04 0x13 0x7D 0x68 0x51

The boards answer provides information on the actual position, the moment and motor current. The message ID used is ID+1

Protocol:	ErrorCode Velocity PositionH PositionL Shunt TimeStamp SupplyVoltag Din
Example:	0x21 - 0x04 0x00 0x7D 0x6A 0x05 0x51 0xB5 0x06

The supply voltage byte sends the current voltage on the controller board (only available on the Slider mobile Platforms)
The last byte of the answer, Din, encodes the digital inputs of the according joint module in binary format. The base digital inputs are found on board 0x10.

9.4 SetVelocity Command

To set a velocity to the motor the first byte 0x05 is used:

Message ID:	board id
Protocol:	Command Velocity TimeStamp
Command:	0x05
Velocity:	Velocity value from -max (0) to +max (255). 127 is zero.
TimeStamp:	Arbitrary number, the module will answer with this code
Example:	0x20 - 0x05 0x90 0x51

The boards answer follows the SetJoints answer.

9.5 Error Codes

The board provides two means of error indications, via the inner LED and via the CAN bus.

9.5.1 Red LED

Off:	No error
One fast blink every second:	Restart after brown-out reset. Supply voltage dropped below minimum value. Increase stability of supply.
Two fast blinks every second:	Restart after watch-dog reset. Microcontroller got stuck. Get in contract with Commonplace Robotics.
Three fast blinks every second:	CAN error has occurred

9.5.2 CAN-bus

Error	Bit in error byte	Meaning	Possible action
Brown Out or Watch Dog	Bit 1	Microcontroller restarted after a brown out. Supply voltage was too low or µC got stuck.	Increase stability of supply voltage. Reset errors.
Velocity Lag	Bit 2	Velocity changes too fast	Reset errors, enable again. Slower acceleration.
Motor not enabled	Bit 3	Not an error. Motor needs to be enabled by explicit command	Enable motor when appropriate.
Comm Watch Dog	Bit 4	Interval without command was too long	Provide the position or velocity commands in a reliable and short enough time interval. Increase maxMissedCom.
Position Lag	Bit 5	Position is too far away from the setpoint position	Provide setpoint positions reachable to the current motor

			position. Increase maxLag.
Encoder Error	Bit 6	The sequence of the quadrature encoder pulses did not fit.	Check connection cable motor – motor controller
Over Current	Bit 7	Current value too high	Decrease applied load on motor. Increase maxCurrent.
CAN Error	Bit8	CAN error occured	CAN bus to crowded? All connectrions ok?

A normal state after start-up of the board is an error code of 0x1C (Motor not enabled, CommWatchDog, PositionLag)
After an error reset the normal state is 0x04 (Motor not enabled).
After enabling the motor the status is 0x00, now the motor is ready to move.

To reach this state, the communication has to provide values in a fast and reliable way and, when in position control, in the reach of the motor. These restrictions are taken to prevent unwanted motion due to e.g. a blue screen on the control PC, a broken communication, or programming bugs.

10. Troubleshooting

This section offers approaches to solve problems and errors in the areas Installation, Software and Hardware. If these measures do not solve your problem please get in contact with us, we are happy to help:

- Mail: service@cpr-robots.com
  Please add a description of the problem, the robots serial number (found at the base) and the three files „install.log", „startUpLog.txt" and „logMessages.log". They are found in c:\CPRog\.
- Phone: ++49 5429 / 37983-4 in Germany, GMT+1

10.1 Installation and Program Start

Error	Possible Cause	Measures
Error message: „Windows Version older than XP. Installation stops."	Is an older Windows version running on the computer? CPRog needs Windows XP or higher.	Installation on a computer with Windows XP or higher.
Program does not start, error message e.g. „The Application could not be initialized correctly (0xc0000135). "	Is the Microsoft .NET-Framework missing?	Install the current .NET framework from the Microsoft homepage
Program does not start, error message: "CPRog has encountered a problem and needs to close".	Is and DirectX file missing? These files are necessary for joypad and audio functionalities.	Install DirectX Version 9.0c for your system. Links can be found using any search engine.
Program does not start, error message: „CPRog does not work any more"	If running on Windows 7: If CPRog is installed at c:\Programs or c:\Programs (x86) it can be started only with administrator rights.	Right-click on the CPRog link on the desktop, choose „start as administrator"

10.2 Software CPRog

Error	Possible Cause	Measures
Connection loss	The robot expects periodically a message from the CPRog computer. If this does not have sufficient free computing power, the break between two	Reset the errors and free the motors, restart the program, see section 5.4. Avoid these problems by closing further programs, eventually disabling the network connection and

	messages may be too long.	the virus scanner.
Does not move any more: Message "Virtual Box violated" and warning sound.	When moving in „Cartesian Mode" the robot cannot get out of a virtual box. In "Joint Mode" this is possible, changing back in CartMode may lead to these warnings.	Change to JointMode and move the robot back into the virtual Box, e.g. into a standard position.
The robot does not connect, the connect button does not work.	Is the USB-CAN adapter plugged in, are the driver installed? Is the mains adapter plugged and switched on? Is the EmergencyStop button released? Do the green status LEDs in the joint modules blink?	You can find further information in the Log-Messages window. The LED of the USB interface provides information: • LED slow blinking: power on • LED fast blinking: communication on the bus, ok • LED on for longer phases: transmission losses, traffic too high?
The robot does not react to the joypad	Has it been activated? Is the green ok mark on the joystick button?	The Log-Messages window shows if the joypad has been connected correctly. On the joypad: When the red light beneath the mode button of the joypad is on, you have to press the mode button once again.

10.3 Hardware Mover6

Error	Possible Cause	Measures
Position error: the robot does not proceed to the same positions as before.	The robot can lose its joint values due to unexpected power loss or motions of the joints when the robot is switched off.	Reset the joint zero values according to section 5.6. Do not disconnect the robot from energy while in motion. Do not move the joints manually.
The gripper does not open any more	The gripper is controlled via the digital outputs 11 (open/close) and 12 (activated?). Is DOut 12 set to true?	Set DOut 12 manually or via program to true;
USB-Interface is not recognized	Is the USB interface plugged into a port where it has not been in before?	With WindowsXP you have to install the interface driver once for every port it is plugged in. Use the assistant that opens automatically.
No motion, no connection possible	Burnt-out fuse in the robot base?	At the base of the robot there is a small fuse cabinet for 5x20 mm microfuses, 3.1 A. The reason for the failure should be known and removed before changing the fuse!

11. Language Specification

The following nine commands can be combined to form a robot program. A program is a XML file that starts with the lines

```
<?xml version="1.0" encoding="utf-8"?>
<!-- values in mm and degree -->
<Program>
<Header ProgramName ="CPRog recording" Author="nn" SetUpDate=""
              LastChangeDate="" Kinematic="CPRFour"/>
```

Then the commands follow, the program closes with the line

```
</Program>
```

Command Reference
`<Linear Nr="2" x="232.93" y="12.2" z="494.01" a="0.05" b="84.25" c="0" vel="70" acc="40" smooth="true" Descr=""/>`
`<Joint Nr="1" a1="5.19" a2="-26.08" a3="124.72" a4="-14.39" a5="0" a6="0" velPercent="35" acc="40" smooth="true" Descr="" />`
`<Relative Nr="17" x="0" y="0" z="-20.0" a="0" b="0" c="0" vel="50" acc="40" smooth="true" Descr=""/>`
`<Output Nr="6" Local="True" DIO="5" State="True" Descr="" />`
`<Wait Nr="4" Seconds="1" Descr="" />`

`<Loop Nr="1" Mode="Count" Times="3" Descr=""/>`
`… place the commands to be repeated here`
`<EndLoop Nr="1" />`
Alternative:
`<Loop Nr="1" Mode="DIn" Local="True" Channel="0" State="True" Descr=""/>`

`<If Nr="15" Local="True" Channel="0" Descr=""/>`
`… place the then-commands here`
`<Else/>`
`… place the else-commands here`
`<EndIf/>`

`<Sub Nr="6" File="up.xml" Descr=""/>`
`<ExMotion Nr="14" IP="127.0.0.1" Port="1234" Scale="1" Descr="" />`