Z-Arm SDK instruction

I C#

Operation Environment: vs2010 runtime libraries, net framework 4.0 or above installed

Development Environment: vs2010 or above installed

1、Preparation

1，Set a new C# program, copy the appropriate libraries to the relevant file, included ClassLibrary_ControlBean.dll，share.dll，server.exe，small_scara_interface.dll, and copy the 32bit application to bin\debug, the 64bit to bin\x64\Debug.

2，Add ClassLibrary_ControlBean.dll to references with the program.

Add using TcpserverExDll；using ControlBeanExDll to the main program.

3，First, call the TcpserverEx.net_port_initial() to initialize the network.

4，Call TcpserverEx.card_number_connect(int card_number) to check the equipment connected or not.

5，Call ControlBeanEx robot= TcpserverEx.get_robot(int card_number) to select the Z-arm.

6，Call robot.initial(int generation,float z_travel) to initialize the Z-Arm.

7，Call robot. set_arm_length(float l1,float l2) to set the first joint and second joint with Z-Arm, the default is l1=200,l2=200；

8，After initialization, call robot.unlock_position() to unlock Z-Arm.

9，Z-Arm could be controlled with other control libraries when it was unlock.

2、Libraries instruction

1）TcpserverEx Type：

Member function：

No.	Function declaration	Incoming Parameter	Return Value
1	static void net_port_initial()		

	Explain: initialize the server		
2	static int card_number_connect(int card_number); Explain: Check the Z-Arm connected or not	1)card_number (The number of Z-Arm, the fourth bit of the IP address)	=0 disconnected =1 connected =2 incoming parameter error =101 incoming parameter NAN
3	static ControlBeanEx get_robot(int card_number); Explain: Select the Z-Arm with number	1)card_number (The number of Z-Arm, the fourth bit of the IP address)	Callback from the selected Z-Arm
4	static void close_tcpserver(); Explain: close the tcpserver		

2）ControlBeanEx Type

Member function：

float x;//Set the coordinate with X axes of Z-Arm（mm）

float y; // Set the coordinate with Y axes of Z-Arm (mm)

float z; // Set the coordinate with Z axes of Z-Arm (mm)

float angle1; // Set the coordinate witeh first joint of Z-Arm(deg)

float angle2; // Set the coordinate with second joint of Z-Arm (deg)

float rotation;// // Set the coordinate with fourth joint (deg)

bool communicate_success;//state of host computer and Z-Arm，true (connected)，false(disconnected)

bool initial_finish;//state of initialization，true(initialized)，false(not initialized)

bool move_flag;//state of Z-Arm，true(running)，false(stop)

bool servo_off_flag;//state of Z-Arm，true(servo on)，false，(servo off)

-- The above variants should be update after calling get_scara_param()

bool isReach_after_judge;//

-- The above variants should call judge_position_gesture() and get callback with true before update.

float angle1_after_judge;// With judge_position_gesture() libraries, x and y are the relative coordinates

of first joint.

float angle2_after_judge; // With judge_position_gesture() libraries, x and y are the relative coordinates of second joint.

-- The above variants should call judge_position_gesture() and angle2_after_judge get callback with true before update.

int efg_type;

float efg_distance;

-- The above variants should call get_efg_state() and get callback with 1 before update.

Function libraries：

No.	Function declaration	Incoming Parameter	Return Value
~~1~~	int initial(int generation, float z_travel) Initial parameters corresponding to the model Default J1 joint arm length is 200mm Default J2 joint arm length is 200mm.	1) int generation. =1 Z-Arm low configuration =2 Z-Arm high configuration 2) float z_travel Set the up and down stroke to be 210 or 310 according to the actual model, and you need to pass positive value;	=0 communication has not yet been established,this initialization is unsuccessful; =1 initializing; =2 generation parameter error; =3 encoder value error; =11 controlled by the mobile terminal, this initialization is not successful =12 z_travel transmission error
~~2~~	void get_scara_param(float *x,float *y,float*z, float *angle1, float *angle2, float *rotation, bool *communicate_success, bool *initial_finish, bool *servo_off_flag, bool *move_flag)	1) float *x, coordinate value of x (mm) 2) float *y, coordinate value of y (mm) 3) float *z, coordinate value of z (mm) 4) float *angle1, angle value of joint 1 (deg) 5) float *angle2, angle value of joint 2	No Return Value

		(deg) 6) float * rotation, angle value of joint 4 (deg) 7) bool *communicate_success, =0 communication has not been connected =1 communication has been established 8) bool *initial_finish, =0 initialization successful =1 initialization unsuccessful 9) bool *servo_off_flag, =0 servo not closed =1 servo closed 10) bool *move_flag, =0 the robot arm is in standby state =1 the robot arm is in motion state	
~~3~~	void set_arm_length(float l1, float l2) Set J1 J2 joint arm length	float l1 J1 joint length, reserved parameters, you must introduce J1 joint length, introduce 160 or 200 generally for those with J4 rotation joint, and set according to the actual situation for those with no J4 rotation joint	No Return Value
4	int unlock_position(int n); Unlock function, before you control the movement of the robotic arm, you must unlock first	1) int n The fourth bit of the robotic arm's IP address	=0 not connected =1 connected =2 parameter n error
5	bool is_connected();		=true connected

	Explain: check the Z-Arm connected or not		=false disconnected
6	int get_card_num(); Explain: Get the number of Z-Arm, and get invoked after initialization		Callback the number of Z-Arm
7	int get_joint_state(int joint_num); Explain: Get state of Z-Arm, available after initialization	1)joint_num Joint number	=0 reset the joint and need to be initialized again =1 joint could move regular =2 incoming parameter error =3 joint is not initialized =4 fail to get joint state（Only new version support） =5 in collision; =6 Drag-teaching mode;
8	bool set_drag_teach(bool state); Explain: Only Z-Arm support, drag-teaching mode on, the other joint could be drag except joint 3, available after initialization	1)state True on False off	=true setting success =false setting fail
9	bool get_drag_teach(); Explain: Only Z-Arm support, to check drag-teaching mode is on or off, available after initialization		=false drag-teaching mode off =true drag-teaching mode on
10	bool set_cooperation_fun_state(bool state); Explain: Only Z-Arm support, to check collision protect mode is on or off, available after initialization	1)state true collision protect mode on false collision protect mode off	=true setting success =false setting fail
~~11~~	bool get_cooperation_fun_state() Query whether the coordination function is on	No input parameter	=false off =true on
~~12~~	bool is_collision() Query whether the coordination function is triggered (collided)	No input parameter	=false no =true yes

Movement libraries

No.	Function declaration	Incoming Parameter	Return Value
1	int set_position_move(float goal_x, float goal_y, float goal_z, float rotation, float speed, float acceleration, int interpolation, intmove_mode); Move to the target point from the current position attitude	float goal_x X coordinate value of the target point, unit is mm float goal_y Y coordinate value of the target point, unit is mm float goal_z Z coordinate value of the target point, unit is mm float goal rotation z J4 angle value of the target point, unit is deg float speed running speed mm/s float acceleration acceleration value in T shape interpolation, valid only when interpolation=2; int interpolation 1 is s curve interpolation, and 2 is T curve interpolation intmove_mode =1 is MOVEJ The trajectory from the current position to the target position is a straight line (if it can arrive) =2 is MOVEL Each joint moves from the current position to the target position, and the intermediate	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robot arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 in the MOVEL movement, the intermediate process points go out of bounds and it cannot arrive, and the robotic arm will stop moving =6 robotic arm servo not opened =7 in the MOVEL movement, any intermediate process point cannot arrive by the robotic arm's current attitude (attitude), and the robotic arm will stop moving =8 setting acceleration is less than or equal to zero =9 interpolation mode

		movement trajectory is generally not a straight line	parameter error =10 move_mode move mode error =11 mobile terminal is controlling =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
~~2~~	int set_angle_move(float angle1, float angle2, float z, float rotation, float speed);	float angle1 The absolute angle of the target point joint 1, unit is deg float angle2 The absolute angle of the target point joint 2, unit is deg float z The absolute coordinate of target point joint 3, unit is mm float rotation The absolute coordinate of target point joint 4, unit is deg float speed Running speed unit Judge the difference of the joint angles between the current position and the target point, divided by speed at the same time, to get the movement time of each joint, and take the longer time as the final movement time, and then inversely calculate the actual running speed of each joint	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 the position point goes beyond bounds =6 the robotic arm servo not opened =11 mobile terminal is controlling =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
~~3~~	int xyz_move(int direction, float distance, float speed) Motion of x, y, z single axis	int direction =1 x axis direction motion =2 y axis direction motion =3 z axis direction motion float distance Offset in the direction of direction relative to	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move

		the current position float speed Unit is mm/s	=2 setting speed is less than or equal to zero =3 not initialized yet =4 process point cannot reach =5 direction parameter error =6 robotic arm servo not opened =7 any intermediate process point cannot arrive by the robotic arm's current attitude (attitude), and the robotic arm will stop moving =11 mobile terminal is controlling =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
4	int single_joint_move(int axis, float distance, float speed);	1)axis Input 1 to 4, matching with joint 1 to joint 4 2)distance Moving distance from the present position， When axis=3, unit of distance is mm, when axis=1 or 2 or 3, unit of distance is deg 3)speed	=0 Z-Arm is in state of other implementation, the present implementation is unavailable. =1 present implementation is available, Z-Arm begin moving

		Moving speed， When axis=3, unit of speed is mm/s When axis=1 or 2 or 4, unit of speed is deg/s	=2 set the speed less or equal to 0 =3 Not initialized yet =4 Could not reach the position =5 parameter error of the number of output axis =6 Z-Arm servo off =11 Controlling by APP =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
5	int trail_move(intpoint_number, float *x, float *y, float *z, float *r, float speed); Represent four degrees of freedom x(mm),y(mm),z(mm),r(deg) of all the point coordinates in a section of trajectory with four float arrays, and indicate the total number of points and running speed, introduce into the trail_move function; Note: the linear distance between two	intpoint_number Number of points to be executed float *x The first address of x coordinate array, and the unit of data in the array is mm float *y The first address of y coordinate array, and the unit of data in the array is mm float *z The first address of z coordinate array, and the unit of data in the array is mm float *r The first address of r coordinate array, and the unit of data in the array is deg	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 the first point in the trajectory goes beyond bounds =6 the robotic arm servo not opened

	adjacent points in the trajectory should be equal to 1mm	float speed Running speed	=11 mobile terminal in controlling =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
~~6~~	int change_attitude(float speed) Change attitude	float speed The joint speed (deg/s) when transforming attitude, and the difference of the joint angles between the two attitudes will be judged. At the same time, divided by speed to get the motion time of each joint, and the longer time is the final movement time	=0 the robot arm is running other instructions, this command is invalid =1 this command goes into effect, and the robot arm begins to move =2 the incoming speed is less than or equal to 0 =3 not initialized yet =4 can't reach by the other attitude =6 servo not opened =11 mobile terminal is controlling =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
~~7~~	void stop_move()	No Incoming Parameter	Cannot Return Value

	Stop the robot arm and stop all movement		
~~8~~	void servo_off() Turn off the servo	No Incoming Parameter	Cannot Return Value
~~9~~	bool servo_on() Trun on the servo	No Incoming Parameter	=0 not initialized or initialization not completed =1 settings successful
10	bool is_robot_goto_target(); Explain: Check the four incoming parameters whether the arm arrived the position or not		=true arrived =false not arrived
11	void set_allow_offset_at_target_position(float x_distance, float y_distance, float z_distance, float r_distance);	1)x_distance X axes coordinate deviation 2)y_distance Y axes coordinate deviation 3)z_distance Y axes coordinate deviation 4)r_distance R axes coordinate deviation	
12	void set_catch_or_release_accuracy(float accuracy); Explain: The allowed error to get the position when the Z-Arm moving with Y axes.	1)accuracy allowed error	
~~13~~	bool judge_in_range(float x, float y, float z, float ratation) Judge whether the output position point	float x X axis coordinate value mm float y Y axis coordinate value mm float z Z axis coordinate value mm	=false it cannot arrive =true it can arrive

	can arrive	float rotation J4 joint angle deg	
14	bool judge_position_gesture(float x, float y); Explain: Callback bool judge_position_gesture(float x, float y) before callback set_position_move(), if it could callback true that means available, check member variants isReach_after_judge, if callback true, position could be reached, if not, it could not arrived.	1)x X axes coordinate of position 2)y Y axes coordinate of position	=ture success =false fail, Z-Arm is running.
15	int joint_home(int joint_num); Explain: Reset the Z-Arm when it was connected but haven't been initialized. After callback, Z-Arm will be back to not initialized.	1)joint_num Joint number	=0 connected =1 success =2 incoming parameter error =3 Z-Arm is initializing.

IO libraries

No.	Function declaration	Incoming Parameter	Return Value
~~1~~	bool set_digital_out(intio_number, bool value) Set io output	intio_number Output io port number, value range is 0-2 containing 0 and 2, which can be 0-2 currently; others are reserved bool value Set the output value of io_number	=0 io_number parameter error =1 settings successful =3 not initialized yet

		=0 corresponds to the disconnect state of the two pins of io =1 corresponds to the conducting state of the two pins of io The connection of IO port output pin is shown in Appendix 1.2	
~~2~~	int get_digital_out(int io_out_num); Obtain the state of io output interface	1) int io_out_number the serial number of io interfaces.	=-1 io_out_num parameter error =0 output state of io interface is off =1 output state of io interface is on =3 not initialized yet
~~3~~	int get_digital_in(intio_in_number) Get the state value of the output IO	int io_in_number Input the io port number 0-2, including 0 and 2; Specific pin connection mode is shown in Appendix 1.3;	=0 24 v signal input =1 not connected, or no signal input =2 parameter io_in_number error =3 not initialized yet
~~4~~	int set_efg_state(int type, float distance) Aims to control efg-20 motor-driven gripper(effective stroke is 20mm, which is unadjustable) and efg-8 motor-driven gripper(effective stroke is 8mm, which is unadjustable) Notice: every time after the mechanical arm is powered up, the controlling type can't be changed.	1） int type Type of motor-driven grippers: 20 for efg-20 and 8 for efg-8 2） float distance If type=20, distance for gripping position, data range (0,20), accurate to 0.1 If type=8, distance=0, stretch,	=1 Controls parameter changed =0 Type parameter error =1 Set ok =3 not initialized yet

		Distance=1, clamp.	
5	int get_efg_state(int *type, float *distance) Acquire the controlling type and the actual postion of the motor-driven gripper	1） int *type Shift to int pointer type. Assigning after function reference. type=0 controlling type unselected type=8 controlling type is efg-8 type=20 controlling type is efg-20 2) Shift to float pointer type. Assigning after function reference.	=1 Function reference ends =3 not initialized yet

II CPP version。

For now, only support the Windows x86 or x 64 application developed by CPP.

1、Preparation

1 New CPP program, copy hitbot_interface.h，ControlBeanEx.h，hitbot_interface.lib to the source program files, and INCLUDE the first 2 files with the program.

2 Copy the libraries of relative version to the debug files, included hitbot_interface.dll，share.dll，server.exe，small_scara_interface.dll.

3 Callback net_port_initial() to initialized the network.

4 Callback card_number_connect(int card_number) to check whether the equipment connected or not.

5 Callback ControlBeanEx* robot=get_robot(int card_number) to select the Z-Arm.

6 Callback robot->initial(int generation,float z_travel) to initialized the Z-Arm.

7 Callback robot->set_arm_length(float l1,float l2) to set joint 1 and joint 2 with Z-Arm, the default is l1=200,l2=200.

8 After initialization, callback robot->unlock_position() to unlock the Z-Arm

9 Z-Arm could be controlled with other libraries after unlock.

2、Libraries instruction

1）Export libraries of hitbot_interface with dynamic-link library：

Function libraries：

No.	Function declaration	Incoming Parameter	Return Value
1	extern "C" __declspec(dllexport) void net_port_initial(); Explain: initialize the server		
2	extern "C" __declspec(dllexport) int card_number_connect(int num); Explain: Check the Z-Arm connected or not	1)card_number (The number of Z-Arm, the fourth bit of the IP address)	=0 disconnected =1 connected =2 incoming parameter error =101 incoming parameter NAN
3	extern "C" __declspec(dllexport) ControlBeanEx * get_robot(int card_number); Explain: get the Z-Arm pointer	1)card_number (The number of Z-Arm, the fourth bit of the IP address)	Callback to the Z-Arm number
4	extern "C" __declspec(dllexport) void close_tcpserver(); Explain: close the tcpserver		

2）ControlBeanEx Type

Member function：

float x;//Set the coordinate with X axes of Z-Arm（mm）

float y; // Set the coordinate with Y axes of Z-Arm (mm)

float z; // Set the coordinate with Z axes of Z-Arm (mm)

float angle1; // Set the coordinate witeh first joint of Z-Arm(deg)

float angle2; // Set the coordinate with second joint of Z-Arm (deg)

float rotation;// // Set the coordinate with fourth joint (deg)

bool communicate_success;//state of host computer and Z-Arm， true (connected)， false(disconnected)

bool initial_finish;//state of initialization，true(initialized)，false(not initialized)

bool move_flag;//state of Z-Arm，true(running)，false(stop)

bool servo_off_flag;//state of Z-Arm，true(servo on)，false，(servo off)

-- The above variants should be update after calling get_scara_param()

bool isReach_after_judge;//

-- The above variants should call judge_position_gesture() and get callback with true before update.

float angle1_after_judge;// With judge_position_gesture() libraries, x and y are the relative coordinates of first joint.

float angle2_after_judge; // With judge_position_gesture() libraries, x and y are the relative coordinates of second joint.

-- The above variants should call judge_position_gesture() and angle2_after_judge get callback with true before update.

int efg_type;

float efg_distance;

-- The above variants should call get_efg_state() and get callback with 1 before update.

Functional member libraries：

No.	Function declaration	Incoming Parameter	Return Value
1	int initial(int generation, float z_travel) Initial parameters corresponding to the model Default J1 joint arm length is 200mm	1) int generation. =1 Z-Arm low configuration =2 Z-Arm high configuration	=0 communication has not yet been established,this initialization is unsuccessful; =1 initializing; =2 generation parameter error;

	Default J2 joint arm length is 200mm.	2) float z_travel Set the up and down stroke to be 210 or 310 according to the actual model, and you need to pass positive value;	=3 encoder value error; =11 controlled by the mobile terminal, this initialization is not successful =12 z_travel transmission error
~~2~~	void get_scara_param(float *x,float *y,float*z, float *angle1, float *angle2, float *rotation, bool *communicate_success, bool *initial_finish, bool *servo_off_flag, bool *move_flag)	1) float *x, coordinate value of x (mm) 2) float *y, coordinate value of y (mm) 3) float *z, coordinate value of z (mm) 4) float *angle1, angle value of joint 1 (deg) 5) float *angle2, angle value of joint 2 (deg) 6) float * rotation, angle value of joint 4 (deg) 7) bool *communicate_success, =0 communication has not been connected =1 communication has been established 8) bool *initial_finish, =0 initialization successful =1 initialization unsuccessful 9) bool *servo_off_flag, =0 servo not closed =1 servo closed 10) bool *move_flag, =0 the robot arm is in standby state =1 the robot arm is in motion state	No Return Value
~~3~~	void set_arm_length(float l1, float l2)	float l1 J1 joint length, reserved parameters,	No Return Value

	Set J1 J2 joint arm length	you must introduce 200, float l2 J1 joint length, introduce 200 generally for those with J4 rotation joint, and set according to the actual situation for those with no J4 rotation joint	
4	int unlock_position(int n); Unlock function, before you control the movement of the robotic arm, you must unlock first	1) int n The fourth bit of the robotic arm's IP address	=0 not connected =1 connected =2 parameter n error
5	bool is_connected(); Explain: check the Z-Arm connected or not		=true connected =false disconnected
6	int get_card_num(); Explain: Get the number of Z-Arm, and get invoked after initialization		Callback the number of Z-Ar
7	int get_joint_state(int joint_num); Explain: Get state of Z-Arm, available after initialization	1)joint_num Joint number	=0 reset the joint and need to be initialized again =1 joint could move regular =2 incoming parameter error =3 joint is not initialized =4 fail to get joint state（Only new version support） =5 in collision; =6 Drag-teaching mode;
8	bool set_drag_teach(bool state); Explain: Only Z-Arm support, drag-teaching mode on, the	1)state True on	=true setting success =false setting fail

	other joint could be drag except joint 3, available after initialization	False off	
9	bool get_drag_teach(); Explain: Only Z-Arm support, to check drag-teaching mode is on or off, available after initialization		=false drag-teaching mode off =true drag-teaching mode on
10	bool set_cooperation_fun_state(bool state); Explain: Only Z-Arm support, to check collision protect mode is on or off, available after initialization	1)state true collision protect mode on false collision protect mode off	=true setting success =false setting fail
~~11~~	bool get_cooperation_fun_state() Query whether the coordination function is on	No input parameter	=false off =true on
~~12~~	bool is_collision() Query whether the coordination function is triggered (collided)	No input parameter	=false no =true yes

Movement libraries

No.	Function declaration	Incoming Parameter	Return Value
~~1~~	int set_position_move(float goal_x, float goal_y, float goal_z, float rotation, float speed, float acceleration, int interpolation, intmove_mode); Move to the target point from the current position attitude	float goal_x X coordinate value of the target point, unit is mm float goal_y Y coordinate value of the target point, unit is mm float goal_z Z coordinate value of the target point, unit is mm float goal rotation z J4 angle value of the target point, unit is deg float speed running speed mm/s	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robot arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 in the MOVEL movement, the intermediate process points go out of bounds and it cannot arrive,

		float acceleration acceleration value in T shape interpolation, valid only when interpolation=2; int interpolation 1 is s curve interpolation, and 2 is T curve interpolation intmove_mode =1 is MOVEJ The trajectory from the current position to the target position is a straight line (if it can arrive) =2 is MOVEL Each joint moves from the current position to the target position, and the intermediate movement trajectory is generally not a straight line	and the robotic arm will stop moving =6 robotic arm servo not opened =7 in the MOVEL movement, any intermediate process point cannot arrive by the robotic arm's current attitude (attitude), and the robotic arm will stop moving =8 setting acceleration is less than or equal to zero =9 interpolation mode parameter error =10 move_mode move mode error =11 mobile terminal is controlling
~~2~~	int set_angle_move(float angle1, float angle2, float z, float rotation, float speed);	float angle1 The absolute angle of the target point joint 1, unit is deg float angle2 The absolute angle of the target point joint 2, unit is deg float z The absolute coordinate of target point joint 3, unit is mm float rotation The absolute coordinate of target point joint 4, unit is deg float speed Running speed unit Judge the difference of the joint angles between the current position and the target point, divided by speed at the same time, to get the movement time of each joint, and take the longer time as the final movement time, and then inversely calculate the actual	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 the position point goes beyond bounds =6 the robotic arm servo not opened =11 mobile terminal is controlling

		running speed of each joint	
3	int xyz_move(int direction, float distance, float speed) Motion of x, y, z single axis	int direction =1 x axis direction motion =2 y axis direction motion =3 z axis direction motion float distance Offset in the direction of direction relative to the current position float speed Unit is mm/s	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 process point cannot reach =5 direction parameter error =6 robotic arm servo not opened =7 any intermediate process point cannot arrive by the robotic arm's current attitude (attitude), and the robotic arm will stop moving =11 mobile terminal is controlling
4	int single_joint_move(int axis, float distance, float speed);	1)axis Input 1 to 4, matching with joint 1 to joint 4 2)distance Moving distance from the present position， When axis=3, unit of distance is mm, when axis=1 or 2 or 3, unit of distance is deg 3)speed	=0 Z-Arm is in state of other implementation, the present implementation is unavailable. =1 present implementation is available, Z-Arm begin moving

		Moving speed， When axis=3, unit of speed is mm/s When axis=1 or 2 or 4, unit of speed is deg/s	=2 set the speed less or equal to 0 =3 Not initialized yet =4 Could not reach the position =5 parameter error of the number of output axis =6 Z-Arm servo off =11 Controlling by APP =101 Incoming parameter NAN =102 In collision, could not move =103 joint was reset and need to be initialized again
5	int trail_move(intpoint_number, float *x, float *y, float *z, float *r, float speed); Represent four degrees of freedom x(mm),y(mm),z(mm),r(deg) of all the point coordinates in a section of trajectory with four float arrays, and indicate the total number of points and running speed, introduce into the trail_move function; Note: the linear distance between two	intpoint_number Number of points to be executed float *x The first address of x coordinate array, and the unit of data in the array is mm float *y The first address of y coordinate array, and the unit of data in the array is mm float *z The first address of z coordinate array, and the unit of data in the array is mm float *r The first address of r coordinate array, and the unit of data in the array is deg	=0 the robotic arm is running other instructions, this command is invalid =1 this command goes into effect, and the robotic arm begins to move =2 setting speed is less than or equal to zero =3 not initialized yet =4 the first point in the trajectory goes beyond bounds =6 the robotic arm servo not opened

	adjacent points in the trajectory should be equal to 1mm	float speed Running speed	=11 mobile terminal in controlling
6	int change_attitude(float speed) Change attitude	float speed The joint speed (deg/s) when transforming attitude, and the difference of the joint angles between the two attitudes will be judged. At the same time, divided by speed to get the motion time of each joint, and the longer time is the final movement time	=0 the robot arm is running other instructions, this command is invalid =1 this command goes into effect, and the robot arm begins to move =2 the incoming speed is less than or equal to 0 =3 not initialized yet =4 can't reach by the other attitude =6 servo not opened =11 mobile terminal is controlling
7	void stop_move() Stop the robot arm and stop all movement	No Incoming Parameter	Cannot Return Value
8	void servo_off() Turn off the servo	No Incoming Parameter	Cannot Return Value
9	bool servo_on() Trun on the servo	No Incoming Parameter	=0 not initialized or initialization not completed =1 settings successful
10	bool is_robot_goto_target(); Explain: Check the four incoming parameters whether the arm arrived the position or not		=true arrived =false not arrived
11	void set_allow_offset_at_target_position(float	1)x_distance X axes coordinate deviation	

	x_distance, float y_distance, float z_distance, float r_distance);	2)y_distance Y axes coordinate deviation 3)z_distance Y axes coordinate deviation 4)r_distance R axes coordinate deviation	
12	void set_catch_or_release_accuracy(float accuracy); Explain: The allowed error to get the position when the Z-Arm moving with Y axes.	1)accuracy allowed error	
~~13~~	bool judge_in_range(float x, float y, float z, float ratation) Judge whether the output position point can arrive	float x X axis coordinate value mm float y Y axis coordinate value mm float z Z axis coordinate value mm float rotation J4 joint angle deg	=0 it cannot arrive =1 it can arrive
14	bool judge_position_gesture(float x, float y); Explain: Callback bool judge_position_gesture(float x, float y) before callback set_position_move(), if it could callback true that means available, check member variants isReach_after_judge, if callback true, position could be reached, if not, it could not arrived.	1)x X axes coordinate of position 2)y Y axes coordinate of position	=ture success =false fail, Z-Arm is running.

15	int joint_home(int joint_num); Explain: Reset the Z-Arm when it was connected but haven't been initialized. After callback, Z-Arm will be back to not initialized.	1)joint_num Joint number	=0 connected =1 success =2 incoming parameter error =3 Z-Arm is initializing.

IO libraries

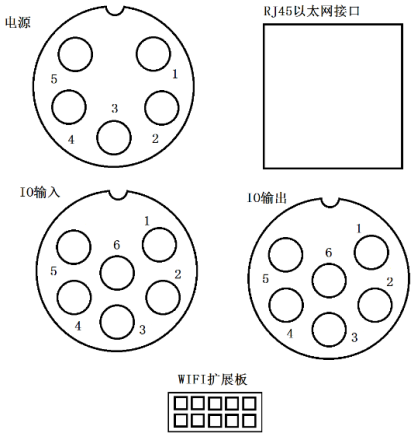
No.	Function declaration	Incoming Parameter	Return Value
~~1~~	bool set_digital_out(intio_number, bool value) Set io output	intio_number Output io port number, value range is 0-2 containing 0 and 2, which can be 0-2 currently; others are reserved bool value Set the output value of io_number =0 corresponds to the disconnect state of the two pins of io =1 corresponds to the conducting state of the two pins of io The connection of IO port output pin is shown in Appendix 1.2	=0 io_number parameter error =1 settings successful
~~2~~	int get_digital_out(int io_out_num); Obtain the state of io output interface	1) int io_out_number the serial number of io interfaces.	=-1 io_out_num parameter error =0 output state of io interface is off =1 output state of io interface is on

3	int get_digital_in(intio_in_number) Get the state value of the output IO	int io_in_number Input the io port number 0-2, including 0 and 2; Specific pin connection mode is shown in Appendix 1.3;	=0 24 v signal input =1 not connected, or no signal input =2 parameter io_in_number error
4	int set_efg_state(int type, float distance) Aims to control efg-20 motor-driven gripper(effective stroke is 20mm, which is unadjustable) and efg-8 motor-driven gripper(effective stroke is 8mm, which is unadjustable) Notice: every time after the mechanical arm is powered up, the controlling type can't be changed.	3） int type Type of motor-driven grippers: 20 for efg-20 and 8 for efg-8 4） float distance If type=20, distance for gripping position, data range (0,20), accurate to 0.1 If type=8, distance=0, stretch, Distance=1, clamp.	=1 Controls parameter changed =0 Type parameter error =1 Set ok
5	int get_efg_state (int* type, float* distance); Explain: Get the electric gripper model and position, after callback, value of type,distance will be assigned. Type=0 means gripper could not be identified, type=8 means EFG-8 electric gripper has been identified, type=20 means EFG-20 electric gripper has been identified.	1)type Int pointer 2)distance Float pointer	=1 Callback success =3 Not initialized

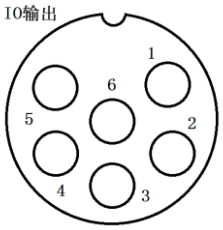
III IO interface instruction

Appendice

1. Interface instruction (old version):



2. IO output interface instruction (Old version):

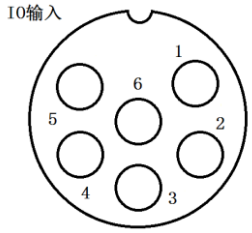


1-2 are IO0 output interfaces, 1 for high level, 2 for low level.

3-4 are IO0 output interfaces, 3 for high level, 4 for low level.

5-6 are IO0 output interfaces, 5 for high level, 6 for low level.

3.IO input interface instruction (Old version):

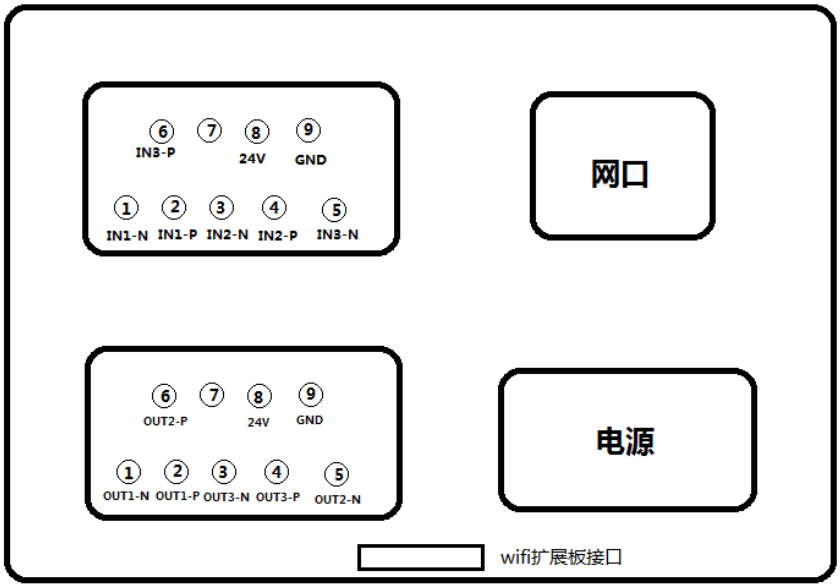


1-2 are IO0 input interfaces，1 and 2 connecting with signal cable at both ends.

3-4 are IO0 input interfaces，3 and 4 connecting with signal cable at both ends.

5-6 are IO0 input interfaces，5 and 6 connecting with signal cable at both ends.

4. Base interface instruction (New Version):



5. Joint 2 interface instruction (new version):