机器人接口

/\*system initialize

\* robot\_flag: select robot ,0: get from configure file;1:sia; 2:ur

\* path: configure file path

\*

\* return 0:ritgth; other: wrong

\* \*/

**int** **rob\_initialize**(**int** robot\_flag, **char**\* path);

/\*move robot by joint space

\* rjoint: target position(joint)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\* \*/

**void** **moveA**(robjoint\* rjoint, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj);

/\*move dual-robot by joint space

\* robot1---

\* rjoint1: target position(joint)

\* rspeed1: move speed

\* rzone1: area of turning

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* rjoint2: target position(joint)

\* rspeed2: move speed

\* rzone2: area of turning

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveA**(robjoint\* rjoint1,robjoint\* rjoint2, speed\* rspeed1,speed\* rspeed2, zone\* rzone1, zone\* rzone2, tool\* rtool1, tool\* rtool2, wobj\* rwobj1, wobj\* rwobj2);

/\*move robot by joint space

\* rjoint: target position(joint)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\*\_index: robot index

\* \*/

**void** **multi\_moveA**(robjoint\* rjoint, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*move robot by joint space

\* rpose: target position(pose)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\* \*/

**void** **moveJ**(robpose\* rpose, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj);

/\*move dual-robot by joint space

\* robot1---

\* rpose1: target position(pose)

\* rspeed1: move speed

\* rzone1: area of turning

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* rpose2: target position(pose)

\* rspeed2: move speed

\* rzone2: area of turning

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveJ**(robpose\* rpose1,robpose\* rpose2, speed\* rspeed1,speed\* rspeed2, zone\* rzone1, zone\* rzone2, tool\* rtool1, tool\* rtool2, wobj\* rwobj1, wobj\* rwobj2);

/\*move robot by joint space

\* rpose: target position(pose)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\*\_index: robot index

\* \*/

**void** **multi\_moveJ**(robpose\* rpose, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*move robot in a straight line

\* rpose: target position(pose)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\* \*/

**void** **moveL**(robpose\* rpose, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj);

/\*move dual-robot in a straight line

\* robot1---

\* rpose1: target position(pose)

\* rspeed1: move speed

\* rzone1: area of turning

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* rpose2: target position(pose)

\* rspeed2: move speed

\* rzone2: area of turning

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveL**(robpose\* rpose1,robpose\* rpose2, speed\* rspeed1,speed\* rspeed2, zone\* rzone1, zone\* rzone2, tool\* rtool1, tool\* rtool2, wobj\* rwobj1, wobj\* rwobj2);

/\*move robot in a straight line

\* rpose: target position(pose)

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\*\_index: robot index

\* \*/

**void** **multi\_moveL**(robpose\* rpose, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*move robot in a circular arc

\* rpose: target position(pose)

\* rpose\_mid: middle position

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\* \*/

**void** **moveC**(robpose\* rpose, robpose\* rpose\_mid, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj);

/\*move dual-robot in a straight line

\* robot1---

\* rpose1: target position(pose)

\* rpose\_mid1: middle position

\* rspeed1: move speed

\* rzone1: area of turning

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* rpose2: target position(pose)

\* rpose\_mid2: middle position

\* rspeed2: move speed

\* rzone2: area of turning

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveC**(robpose\* rpose1,robpose\* rpose2, robpose\* rpose\_mid1, robpose\* rpose\_mid2, speed\* rspeed1,speed\* rspeed2, zone\* rzone1, zone\* rzone2, tool\* rtool1, tool\* rtool2, wobj\* rwobj1, wobj\* rwobj2);

/\*move robot in a circular arc

\* rpose: target position(pose)

\* rpose\_mid: middle position

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\*\_index: robot index

\* \*/

**void** **multi\_moveC**(robpose\* rpose, robpose\* rpose\_mid, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*move robot in a spiral line

\* rpose: target position(pose)

\* rpose\_mid: middle position

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\* \*/

**void** **moveT**(robpose\* rpose, robpose\* rpose\_mid, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj);

/\*move dual-robot in a spiral line

\* robot1---

\* rpose1: target position(pose)

\* rpose\_mid1: middle position

\* rspeed1: move speed

\* rzone1: area of turning

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* rpose2: target position(pose)

\* rpose\_mid2: middle position

\* rspeed2: move speed

\* rzone2: area of turning

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveT**(robpose\* rpose1,robpose\* rpose2, robpose\* rpose\_mid1, robpose\* rpose\_mid2, speed\* rspeed1,speed\* rspeed2, zone\* rzone1, zone\* rzone2, tool\* rtool1, tool\* rtool2, wobj\* rwobj1, wobj\* rwobj2);

/\*move robot in a spiral line

\* rpose: target position(pose)

\* rpose\_mid: middle position

\* rspeed: move speed

\* rzone: area of turning

\* rtool: tool

\* rwobj: coordinate system

\*\_index: robot index

\* \*/

**void** **multi\_moveT**(robpose\* rpose, robpose\* rpose\_mid, speed\* rspeed, zone\* rzone, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*B spline move in joint space

\* filename Necessary points sequence data filename

\* rspeed velocity (time)

\* rtool tool

\* rwobj coordinate system

\* \*/

**void** **moveAJBS**(**char**\* filename, speed\* rspeed, tool\* rtool, wobj\* rwobj);

/\*dual-robot B spline move in joint space

\* robot1---

\* filename1: data file

\* rspeed1: move speed

\* rtool1: tool

\* rwobj1: coordinate system

\* robot2---

\* filename2: data file

\* rspeed2: move speed

\* rtool2: tool

\* rwobj2: coordinate system

\*

\* \*/

**void** **dual\_moveAJBS**(filename1, filename2, rspeed1, rspeed2, rtool1, rtool2, rwobj1, rwobj2);

/\*B spline move in joint space

\* filename Necessary points sequence data filename

\* rspeed velocity (time)

\* rtool tool

\* rwobj coordinate system

\* \_index robot index

\* \*/

**void** **multi\_moveAJBS**(**char**\* filename, speed\* rspeed, tool\* rtool, wobj\* rwobj, **int** \_index);

/\*start move initialize or power

\* \*/

**void** **move\_start**();

/\*stop move or poweroff

\* \*/

**void** **move\_stop**();

/////////////////////////////////////////////////////////////////////////////////////////

/\*get joint data by name of the data

\*J: data name

\*rjoint: joint data

\*\_index: robot index

\*

\* return 1:right other:wrong

\* \*/

**int** **getrobjoint2**(**char**\* J, robjoint\* rjoint, **int** \_index);

/\*get joint data by name of the data

\*J: data name

\*rjoint: joint data

\*

\* return 1:right other:wrong

\* \*/

**int** **getrobjoint**(**char**\* J, robjoint\* rjoint);

/\*get pose data by name of the data

\*P: data name

\*rpose: pose data

\*

\* return 1:right other:wrong

\* \*/

**int** **getrobpose**(**char**\* P, robpose\* rpose);

/\*get speed data by name of the data

\* S: data name

\* sp: speed data

\* \_index: robot index

\*

\* return 1:right other:wrong

\* \*/

**int** **getspeed2**(**char**\* S, speed\* sp, **int** \_index);

/\*get speed data by name of the data

\* S: data name

\* sp: speed data

\*

\* return 1:right other:wrong

\* \*/

**extern** **int** **getspeed**(**char**\* S, speed\* sp);

/\*get zone data by name of the data

\* Z: data name

\* zo: zone data

\*

\* return 1:right other:wrong

\* \*/

**int** **getzone**(**char**\* Z, zone\* zo);

/\*get tool data by name of the data

\*T: data name

\*to: tool data

\*

\* return 1:right other:wrong

\* \*/

**int** **gettool**(**char**\* T, tool\* to);

/\*get wobj data by name of the data

\* w: data name

\* wo: wobj data

\*

\* return 1:right other:wrong

\* \*/

**int** **getwobj**(**char**\* W, wobj\* wo);

/\*get robot current joint

\* joint: current joint data

\* \_index: robot index

\* \*/

**void** **getRobotJoint**(**double**\* joint, **int** \_index);

/\*get robot current pose

\* pospose: current pose data

\* \_index: robot index

\* \*/

**void** **getPosAndPose**(**double**\* pospose, **int** \_index);

/\*sleep

\* \_time time (s) resolution ratio ms

\* \*/

**extern** **void** **RSleep**(**double** \_time);

/\*set robot move the percentage of the acceleration and jerk

\* a: acceleration

\* aa: jerk

\* \*/

**void** **AccSet**(**int** a, **int** aa);

/\*set do

\* id: do index

\* flag: 0 or 1

\* \*/

**void** **SetDo**(**int** id,**int** flag);

/\*get do

\* id: do index

\* flag: return 0 or 1

\* \*/

**void** **GetDi**(**int** id,**int**\* flag);

/\*wait di

\* di :di

\* value: 0 or 1

\* \*/

**void** **WaitDi**(**int** di, **int** value);

/\*set ao

\* id: ao index

\* flag: ao value

\* \*/

**void** **SetAo**(**int** id,**double** flag);

/\*get ao

\* id: ao index

\* flag: return ao value

\* \*/

**void** **GetAi**(**int** id,**double**\* flag);

/\*get offset position

\* rpose: positon

\* x: offset x value

\* y: offset y value

\* z: offset z value

\* k: offset roll

\* p: offset pitch

\* s: offset yaw

\*

\* return offset position

\* \*/

robpose **Offs**(**const** robpose\* rpose, **double** x, **double** y, **double** z, **double** k, **double** p, **double** s);

//----------------------------------------------------------------------------------------communication interface--------------------------------------------------------------------------

/\*create socket server

\* ip: ip

\* port: port

\* sName: server name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketCreate**(**char**\* ip, **int** port, **char**\* sName);

/\*create socket client

\* ip: ip

\* port: port

\* sName: client name

\*

\* return 1:right other:wrong

\* \*/

**int** **ClientCreate**(**char**\* ip, **int** port, **char**\* sName);

/\*close socket server or client

\* sName: server or client name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketClose**(**char**\* sName);

/\*sent byte

\* data: byte data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketSendByte**(**int** data, **char**\* sName);

/\*receive byte

\* data: return byte data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketRecvByte**(**int**\* data, **char**\* sName);

/\*sent string

\* data: string data

\* sName: socket name

\*

\* return >=0string lenth other:wrong

\* \*/

**int** **SocketSendString**(**char**\* data, **char**\* sName);

/\*receive string

\* data: retrun string data

\* sName: socket name

\*

\* return >=0string lenth other:wrong

\* \*/

**int** **SocketRecvString**(**char**\* data, **char**\* sName);

/\*sent double

\* data: double data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketSendDouble**(**double** data, **char**\* sName);

/\*receive double

\* data: retrun double data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketRecvDouble**(**double**\* data, **char**\* sName);

/\*sent int

\* data: int data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketSendInt**(**int** data, **char**\* sName);

/\*receive int

\* data: retrun int data

\* sName: socket name

\*

\* return 1:right other:wrong

\* \*/

**int** **SocketRecvInt**(**int**\* data, **char**\* sName);

/\*sent byte array

\* data: byte array data

\* n: data number

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketSendByteArray**(**int**\* data,**int** n,**char**\* sName);

/\*receive byte array

\* data: return byte array data

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketRecvByteArray**(**int**\* data, **char**\* sName);

/\*sent double array

\* data: doube array data

\* n: data number

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketSendDoubleArray**(**double**\* data, **int** n,**char**\* sName);

/\*receive double array

\* data: return double array data

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketRecvDoubleArray**(**double**\* data, **char**\* sName);

/\*sent int array

\* data: int array data

\* n: data number

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketSendIntArray**(**int**\* data, **int** n, **char**\* sName);

/\*receive int array

\* data: return int array data

\* sName: socket name

\*

\* return >0: number of the data; other: wrong

\* \*/

**int** **SocketRecvIntArray**(**int**\* data, **char**\* sName);

//----------------thread interface---------------------------------

/\*create thread

\* fun: thread execute function

\* arg: execute function

\* name: thread name

\* \*/

**int** **ThreadCreat**(**void**\* (\*fun)(**void**\*), **void**\* arg, **char**\* name, **int** detached\_flag);

/\*thread data free

\* name: thread name

\* \*/

**int** **ThreadDataFree**(**char**\* name);

/\*thread join (It works when thread attribute is PTHREAD\_CREATE\_JOINABLE)

\* name thread name

\* return 0:right; other: wrong

\* \*/

**int** **ThreadWait**(**char**\* name);