KBSW190925 Win32 - Rotate a particular angle/Rotate to a particular angle

This document introduces the demo project of "rotation_action_demo", including how to rotate clockwise, anticlockwise and turn to a particular angle.

Content

IDE Preperation

- ### Software
  - Visual Studio 2010 SP1
  - Slamware Windows SDK:Slamware Windows SDK
  - RoboStudio(for map display):Robostudio installer
  - Sample Code:

> Higher version of Visual Studio will cause errors. sometime you will need to upgrade SP1 package to make your VS compatable with .Net Framework.

- ### Hardware

Either one of following

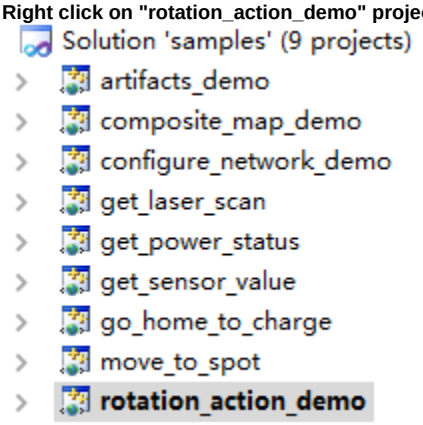
  - Slamware SDP mini
  - Slamware SDP
  - Slamware Kit
  - Zeus/Apollo robot base

Download

Win32-Demo

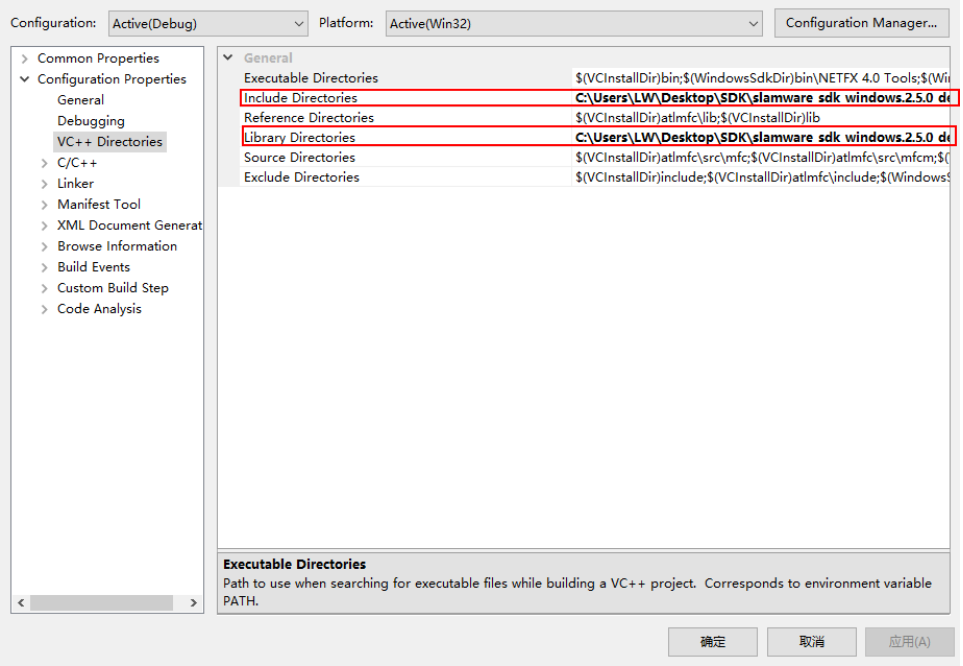
Compiling

1. **Right click on "rotation_action_demo" project, set as StartUp project.**

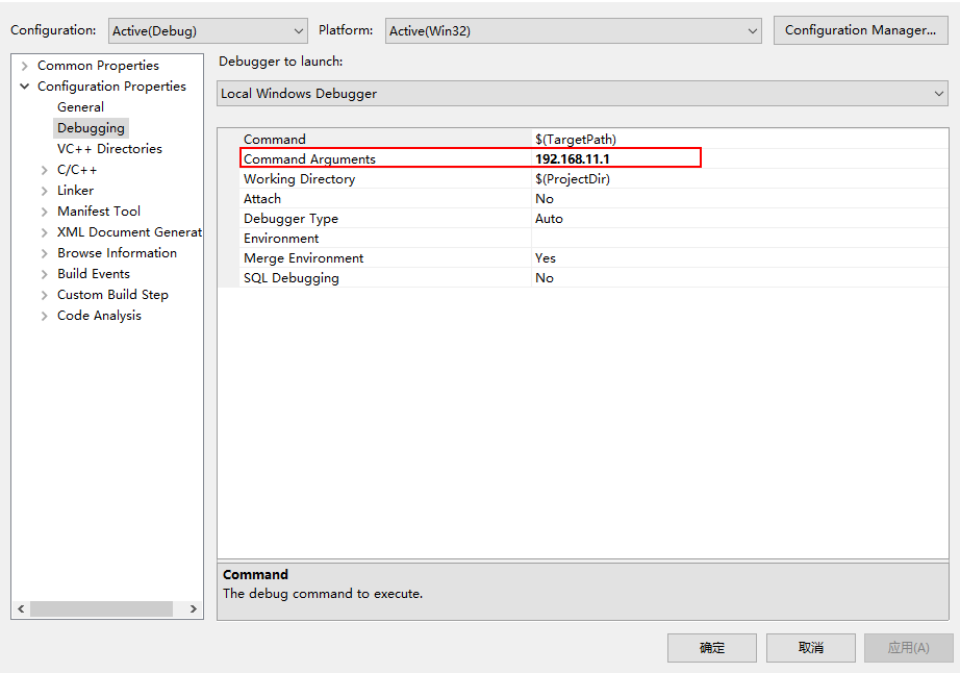


2. **Right click on "rotation_action_demo", then " Properties"configure "include" and "lib" directories to the corresponding folder path of Slamware SDK.**

> It's not necessary to copy files to the project directory, user will only need to configure the path of SDK.



3. **Right click on "rotation_action_demo", then "properties"set "Command Arguments" as follows: Syntax rotation_action_demo <IP address>**



4. **Click " F5" to execute.**
5. **Robot's motion could be seen in Robostudio.**
   Your browser does not support the HTML5 video element

Code

- The robot will turn anticlockwise, then clockwise, and finally turn to the position where the value of yaw is pi.

Rotate a particular angle/Rotate to a particular angle

```
SlamwareCorePlatform sdp = SlamwareCorePlatform::connect(argv[1], 1445);
std::cout <<"SDK Version: " << sdp.getSDKVersion() << std::endl;
std::cout <<"SDP Version: " << sdp.getSDPVersion() << std::endl;
rpos::actions::MoveAction action = sdp.getCurrentAction();
if (action)
    action.cancel();
//anticlockwise rotation
rpos::core::Rotation rotation(pi*2, 0, 0);
action = sdp.rotate(rotation);
action.waitUntilDone();
std::cout << "Action Status: " << action.getStatus() << std::endl;
//clockwise rotation
rotation.yaw() = pi * (-2);
action = sdp.rotate(rotation);
action.waitUntilDone();
std::cout << "Action Status: " << action.getStatus() << std::endl;
//rotate to a certain orientation
rpos::core::Rotation orientation(pi, 0, 0);
action = sdp.rotateTo(orientation);
action.waitUntilDone();
std::cout << "Action Status: " << action.getStatus() << std::endl;
```