KBSW210217 bmp

bmpbmp

-
  -
  -
-
-
-

-
  - Visual Studio 2010 SP1
  - Slamware Windows SDK:Slamware Windows SDK
  - RoboStudio():Robostudio installer
  - Sample Code:

    > Visual Studio
    >
    > Visual Studio 2010SP1.Net FrameworkSP1

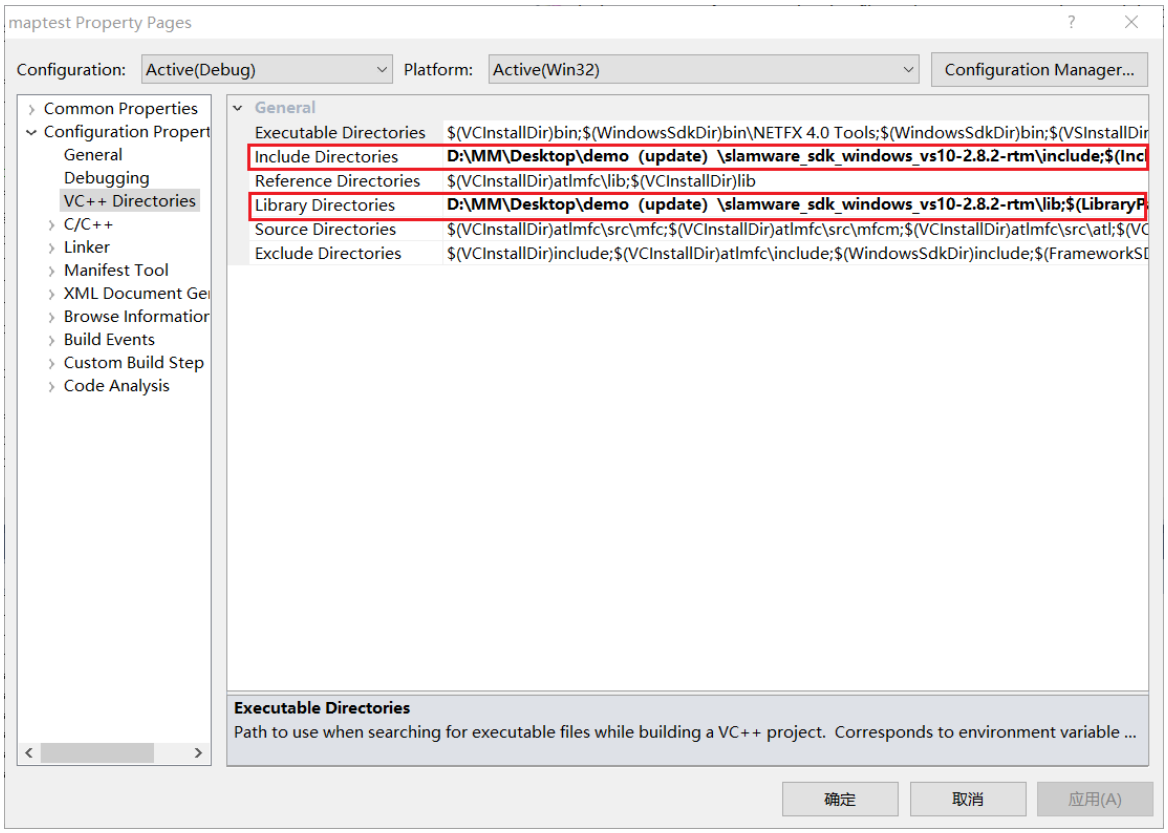
-

    - Slamware SDP mini
    - Slamware Slamware
    - Apollo/Ares/Athena

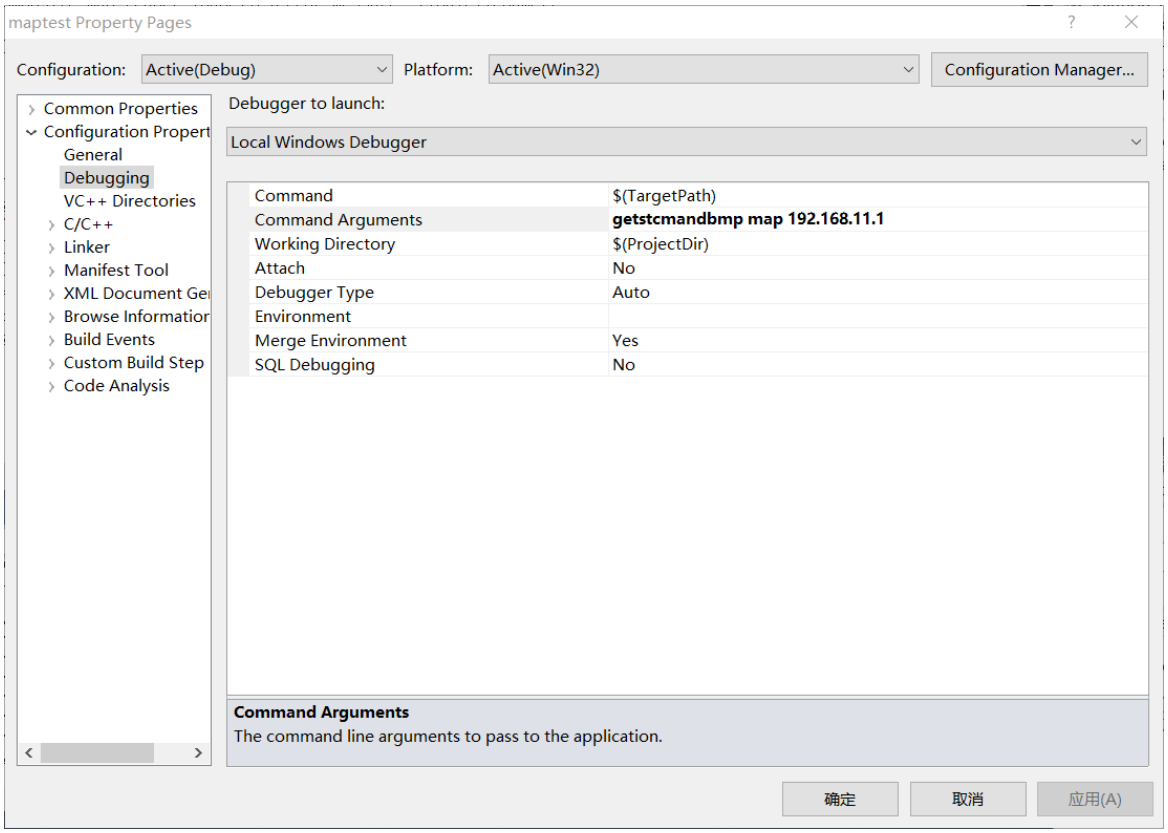
---

maptest.rar

1.maptestmaptest, Slamware SDK includelib



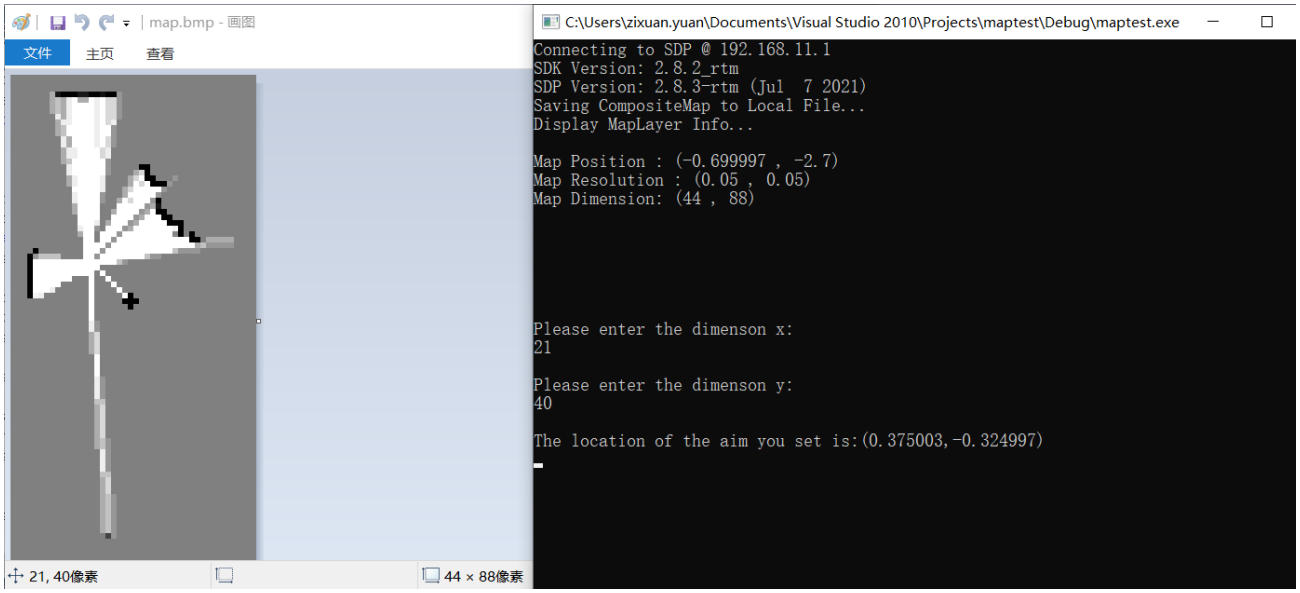
2.Debuggingcommand Arguments



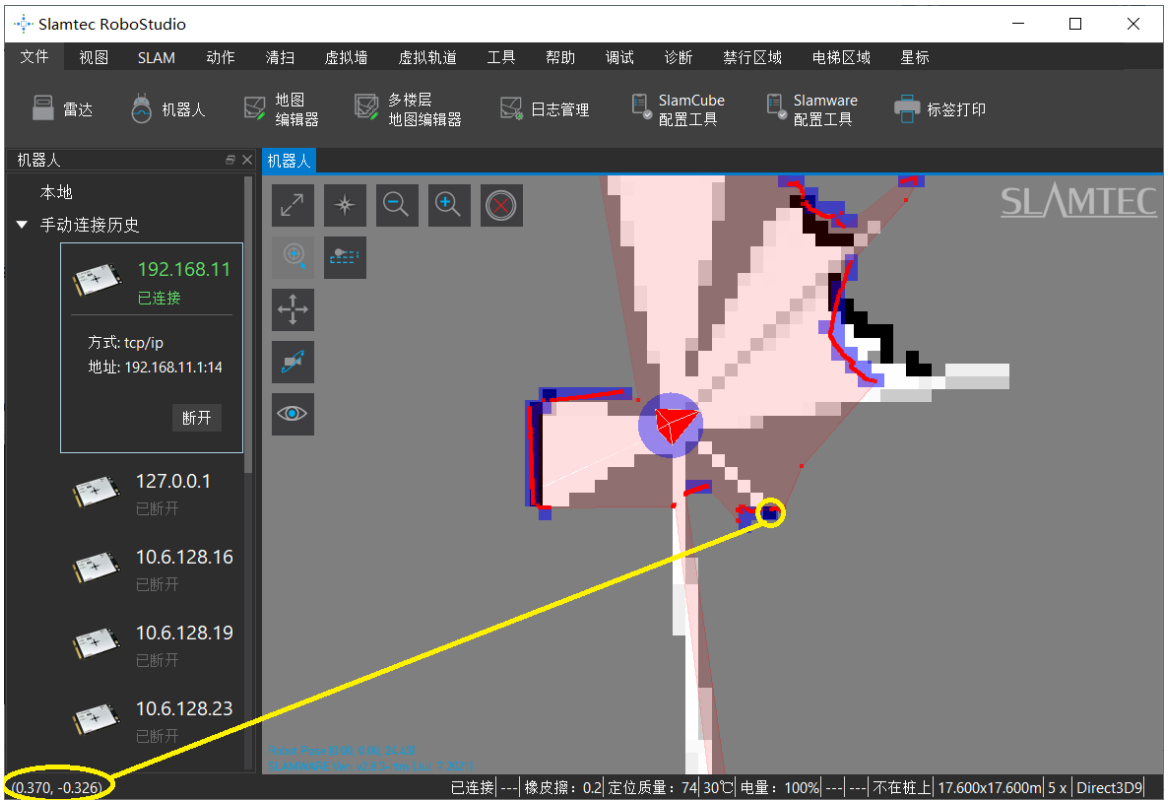
composite_map_demo [OPTS] [filename] <SDP IP Address>
SDP IP Address   The ip address string of the SLAMWARE SDP
getstcmandbmp filename   download compositeMap and bmp map
setstcm filename   upload compositeMap
-h   Show this message

3.F5

4.map.stcmmap.bmpbmp”“console1



5.RoboStudiobmp221



6.composite_map_demo setstcm map 192.168.11.1 stcm

- compositemapstcm

compositemap

```
bool StcmMapWriter(const std::string file_name, SlamwareCorePlatform platform) {
        std::string finalFilename = file_name;
        finalFilename += ".stcm";
        CompositeMap composite_map = platform.getCompositeMap();
        CompositeMapWriter composite_map_writer;
        std::string error_message;
        bool result = composite_map_writer.saveFile(error_message, finalFilename, composite_map);
        return result;
}
```

- stcm

compositemap

```
bool StcmMapReader(const std::string file_path, rpos::core::Pose pose, SlamwareCorePlatform platform)
{
        std::string finalFilename = file_path;
        finalFilename += ".stcm";
        CompositeMapReader composite_map_reader;
        std::string error_message;
        boost::shared_ptr<CompositeMap> composite_map(composite_map_reader.loadFile(error_message,
finalFilename));
        if (composite_map) {
                platform.setCompositeMap((*composite_map), pose);
                return true;
        }
        return false;
}
```

- bmp

bmp

```
bool SaveToBmp(const char * filename, SlamwareCorePlatform platform)
{
        std::string finalFilename = filename;
        finalFilename += ".bmp";

        rpos::core::RectangleF knownArea = platform.getKnownArea(MapTypeBitmap8Bit, rpos::features::
location_provider::EXPLORERMAP);
        Map map = platform.getMap(MapTypeBitmap8Bit, knownArea, rpos::features::location_provider::
EXPLORERMAP);

        bitmap_image mapBitmap(map.getMapDimension().x(), map.getMapDimension().y());

        for (size_t posY = 0; posY < map.getMapDimension().y(); ++posY)
        {
                for (size_t posX = 0; posX < map.getMapDimension().x(); ++posX)
                {
                        rpos::system::types::_u8 cellValue = ((rpos::system::types::_u8)128U) + map.
getMapData()[posX + (map.getMapDimension().y()-posY-1) * map.getMapDimension().x()];
                        mapBitmap.set_pixel(posX, posY, cellValue, cellValue, cellValue);
                }
        }
        mapBitmap.save_image(finalFilename);
        return true;
}
```

- bmp

```
void convertdemension(rpos::core::Location mapposition,double height)
{
        std::cout << "Please enter the dimenson x:" << std::endl;
        int dx;
        std::cin >> dx ;
        std::cout <<  std::endl;
        std::cout << "Please enter the dimenson y:" << std::endl;
        int dy;
        std::cin >> dy;
        std::cout <<  std::endl;

        rpos::core::Location loc;
        loc.x()=mapposition.x()+(dx+0.5)*map_resolution;
        loc.y()=mapposition.y()+(height-dy-0.5)*map_resolution;

        std::cout << "The location of the aim you set is:("<<loc.x()<<","<<loc.y()<<")"<< std::endl;
}
```