RXP2LAZ - DOCUMENTATION

CONTENTS

1. INITIAL DESCRIPTION 1
2. SETUP PROJECT AND CODE COMPILATION 2
2.1 WINDOWS USING CODE BLOCKS 2
2.2 LINUX USING UBUNTU 4

1. INITIAL DESCRIPTION

The source code rxp2laz was developed in C/C++ language to converts .rxp Riegl format to LAS format (1.4) or - its compressed, but identical twin - the LAZ format, using RIVLIB and LASzip dynamic libraries (.dll). **RIVLIB (http://www.riegl.com/index.php?id=224) and LASzip (https://laszip.org/) dynamic libraries are not provided, which must be obtained directly from the owners company with the appropriate copyright**.

!!! ATTENTION !!!

RIVLIB is just available for Riegl clients with access to the RIEGL support area.

The source code can be compile at Windows or Linux platform. Instructions to user compilation at Windows or Linux platform are presented in section 2.1 and section 2.2, respectively. This code is distributed according to MIT License.

Input description: .rpx file (dir\name.rxp), output name (dir\.laz or dir\.las) and optional filtering based on positional coordinates (X, Y, Z) or/and angular coordinates (Theta and Phi) and range. The user can include optional boundary box coordinates (X^{min} Y^{min} Z^{min} X^{max} Y^{max} Z^{max}), separated by space or press enter to ignore this option. Next the user can include angles (degrees) and range (meters) thresholds also separated by space ($Theta^{min}$ Phi^{min} $Range^{min}$ $Theta^{max}$ Phi^{max} $Range^{max}$) or press enter again to ignore this option.

Output description: Laz or Las Version 1.4 and data format 1.0. Fields: X, Y, Z, Intensity, GPS Time, Return Number, Number of returns and five extra parameters (Reflectance, Range, Deviation, Theta, Phi). The output fields can be changed according to the users' need before compilation. For instance, classification, edge of flight line, RGB, user data, scan angle rank and scan direction flag fields can be enabled at line 620.

Limitation: Do not resolve multiple time-around (MTA) echoes (Dec, 2020).

Authors:

Mariana Batista Campos has contributed to code writing, testing and documentation.

Eetu Puttonen has contributed to code debug, development and documentation.

Acknowledgments:

Martin Isenburg, for creating, developing, and maintaining the LAStools and LASzip (http://lastools.org)

Riegl developers for RIVLIB. Available only for Riegl clients (http://www.riegl.com/index.php?id=224).

This work received support from **Academy of Finland project no.316096/320075**, "Upscaling of carbon intake and water balance models of individual trees to wider areas with short interval laser scanning time series" and from the Strategic Research Council at the **Academy of Finland project no. 293389/314312**, "Competence Based Growth Through Integrated Disruptive Technologies of 3D Digitalization, Robotics, Geospatial Information and Image Processing/Computing - Point Cloud Ecosystem (COMBAT) "

2. SETUP PROJECT AND CODE COMPILATION

2.1 WINDOWS USING CODE BLOCKS

ATTENTION: !!! GCC VERSION AND RVLIB VERSION MUST BE COMPATIBLE!!!

This project was build using release mode. Requirements:

- ✓ CODEBLOCKS 16.01 - mingw 4.9 (GCC 4.9) – [http://www.codeblocks.org/]
- ✓ RIVLIB: rivlib-2_5_7-x86-windows-mgw49 (Riegl DLL v7.1) - [http://www.riegl.com/]
- ✓ LASzip DLL v3.4 r1 (build 190728) – [https://rapidlasso.com/laszip/]

PROJECT CONFIGURATION

PROJECT (CLICK ON THE PROJECT NAME)
PROJECT → Build options → TO RUN IN RELEASE MODE (CLICK ON RELEASE)

PROJECT → Build options → Compiler settings → Compiler Flags [-std=c++11]
*Compatibility with RIVLIB used

PROJECT → Build options → Compiler settings → OtherCompilerOptions (if need):

```
-std=gnu++11
-std=gnu++0x
-fexceptions
```

PROJECT → Build options → Linker settings → Linker options →include standard C/C++ libs:
*See LASZIP example configuration

```
kernel32
user32
gdi32
winspool
comdlg32
advapi32
shell32
ole32
oleaut32
```

uuid
odbc32
obdc32

PROJECT → Build options → Linker settings → Other Linker Options
-Wl,--allow-multiple-definition

PROJECT →Build options → Search directory → compiler/linker → point to
Lastools/Laslib/src
Lastools/Laslib/inc
Lastools/Laszip/src
Lastools/Laszip/dll
RIVLIB/include
RIVLIB/lib

PROJECT → Build options → Linker settings → Linker options →include RIVLIB libraries (s.lib)
Find: RIVLIB FOLDER→ LIB → *_s.lib (All)

INPUT FILES

On the workspace → Right click on the project name → add files

HEADERS:

laszip_api.h (LAStools\LASzip\dll)
ctrlifc.h (RIVLIB\include\riegl)
scanifc.h (RIVLIB\include\riegl)

SOURCES:

laszip_api.c (LAStools\LASzip\dll)
Rxp2Lasdll.cpp

EXECUTABLE

By default, an executable (.exe) will be built at bin\Debug or bin\Release
To change this path set Project → Properties → Build Targets
To run the executable, copy to the same folder of the executable the dynamic libraries:
LASzip.dll
scanifc-mt-s.dll

A python script (CommandLineParallel.py) that call the rxp2laz executable and run a maximum number of four files (4 cores) in parallel is available in the folder as additional option.
To run this python script, please inform the path to:

1. Executable path
2. RXP files folder
3. Output folder for Laz files
4. Output folder to configuration files

Please check this configuration in lines 35 to 45.

2.2 LINUX USING UBUNTU

ATTENTION: !!! GCC VERSION AND RVLIB VERSION MUST BE COMPATIBLE!!!

This project was build using:

✓ GCC VERSION 8.3

✓ RIVLIB: rivlib-2_5_7-x86_64-linux-gcc55 (Riegl DLL v7.1) - [http://www.riegl.com/]

✓ LASzip DLL v3.4 r1 (build 190728) – [https://rapidlasso.com/laszip/]

✓ Make installation ubuntu:

[sudo apt-get update]
[sudo apt-get install cmake]
[sudo apt-get upgrade]

✓ **IMPORTANT**: To run the code in **LINUX** please change Rxp2Lasdll.cpp file in the lines:
Line 101: #include "laszip_api.h" to #include <laszip/laszip_api.h>
Lines 226 to 271: Comment load LASzip VIA DLL
Lines 756 to 761: Comment unload LASzip DLL

PROJECT CONFIGURATION

Open the project folder.Example: Cprojects/ubuntu_rxp2las
mkdir Cprojects → cd Cprojects → mkdir ubuntu_rxp2las →cd ubuntu_rxp2las

Create a make file (cmakelists.txt) and save at project folder

CMakeLists.txt

```
---------------------------------------------------------------------------------------------------
cmake_minimum_required(VERSION 3.0)
project(rivlib-test2 C CXX)

find_package(RiVLib COMPONENTS scanlib scanifc)
set(executable Rxp2Las)
add_executable(${executable} src/Rxp2Lasdll.cpp)
target_link_libraries(${executable} PRIVATE ${RiVLib_SCANLIB_LIBRARY} PRIVATE
${RiVLib_SCANIFC_LIBRARY} laszip)
target_include_directories(${executable} PRIVATE ${RiVLib_INCLUDE_DIRS} laszip)
target_compile_options(${executable} PRIVATE -std=c++11)
install(TARGETS ${executable} RUNTIME DESTINATION bin)
---------------------------------------------------------------------------------------------------
```

Open a src folder and include the cpp file there (Rxp2Lasdll.cpp).
Please modified Rxp2Lasdll.cpp according to the instruction above.
cd Cprojects/ubuntu_rxp2las
mkdir src
Cprojects/ubuntu_rxp2las/src >> Rxp2Lasdll.cpp (LINUX VERSION)

SETUP LIBRARIES

A. LASZip

git clone https://github.com/LASzip/LASzip.git
go to LASzip folder
cd LASzip
mkdir build
cd build
cmake ..
sudo make install
* Copy the lib files inside the lib folder of rivlib

B. RIVLIB

cd Cprojects/ubuntu_rxp2las
mkdir build
cd build
-- !!check first the absolute path to rivlib library! note that cmake subfolder is needed here!! --
sudo cmake [cmake subfolder] -G "Unix Makefiles" -DRiVLib_DIR=rivlib-2_5_7-x86_64-linux-gcc55/cmake

In my case: [cmake subfolder] = Cprojects/ubuntu_rxp2las--
cd Cprojects/ubuntu_rxp2las
sudo cmake -G "Unix Makefiles" -DRiVLib_DIR=rivlib-2_5_7-x86_64-linux-gcc55/cmake
sudo make

USE THE PROGRAM

Cprojects/ubuntu_rxp2las sudo ./Rxp2Las
LASzip DLL v3.4 r1 (build 190411)
Riegl DLL v7.1 build 0
Running via command line:./Rxp2Las