The Remote Sensing and GIS Software Library (RSGISLib)

Peter Bunting[a], Daniel Clewley[a,b], Richard M. Lucas[a], Sam Gillingham[c]

[a]*Institute of Geography and Earth Sciences, Aberystwyth University, Aberystwyth, Ceredigion, SY23 3DB, UK.*

[b]*Microwave Systems, Sensors and Imaging Lab, Viterbi School of Engineering, The University of Southern California, Los Angeles, CA, USA*

[c]*Landcare Research, PO Box 40, Lincoln 7640, New Zealand*

Abstract

Key to the successful application of remotely sensed data to real world problems is software that is capable of performing commonly used functions efficiently over large datasets, whilst being adaptable to new techniques. This paper presents an open source software library that was developed through research undertaken at Aberystwyth University for environmental remote sensing, particularly in relation to vegetation science. The software was designed to fill the gaps within existing software packages and to provide a platform to ease the implementation of new and innovative algorithms and data processing techniques. Users interact with the software through an XML script, where XML tags and attributes are used to parameterise the available commands, which have now grown to more than 300. A key feature of the XML interface is that command options are easily recognisable to the user because of their logical and descriptive names. Through the XML interface, processing chains and batch processing are supported. More recently a Python binding has been added to RSGISLib allowing individual XML commands to be called as Python functions. To date the Python binding has over 90 available functions, mainly concentrating on image utilities, segmentation, calibration and raster GIS. The software has been released under a GPL3 license and makes use of a number of other open source software libraries (e.g., GDAL/OGR), a user guide and the source code are available at http://www.rsgislib.org.

Keywords: Software, Remote Sensing, GIS, Raster, Vector, Open Source

1. Introduction

Within the field of remote sensing, the availability and functionality of software for undertaking the required processing of datasets is of particular importance. One of the key requirements for the processing of remote sensing data is the ability to include geospatial information, both when reading data and outputting finished products. Often this processing is undertaken using specialist commercial packages such as Erdas Imagine, ITT ENVI, ESRI ArcGIS, PCI Geomatica and eCognition. With the increasing popularity of open source software and licensing, academics and other individuals are making use of, and contributing to, the increasing amount of software available. Examples of such software are the Orfeo Toolbox (OTB; Inglada and Christophe, 2009, http://www.orfeo-toolbox.org), the OSSIM tools (http://www.ossim.org), GRASS GIS (http://grass.osgeo.org), Geospatial Data Abstraction Library (GDAL; www.gdal.org), SAGA GIS (http://www.saga-gis.org), Opticks (http://opticks.org), SEXTANTE (http://www.sextantegis.com), Terrain Analysis using Digital Elevation Models (TAUDEM; http://hydrology.usu.edu/taudem/taudem5.0/), the Raster Input/Output Simplification (RIOS; https://bitbucket.org/chchrsc/rios) Python library, the Sorted Pulse Data software library (SPD; Bunting et al., 2013a,b, http://www.spdlib.org), PostGIS (http://www.postgis.org), PolSARPro (Pottier et al., 2009, http://earth.eo.esa.int/polsarpro/) and QGIS(http://www.qgis.org).

Email address: petebunting@mac.com (Peter Bunting)

The OTB, from the French Centre National d'Études Spatiales (CNES), is built on the Insight Segmentation and Registration Toolkit (ITK; `http://www.itk.org/`), which provides a large number of image processing algorithms for use within the medical imaging community, specifically image segmentation and image-to-image registration techniques. OTB has also provided a common interface to a number of other tools, including the OSSIM ortho-rectification algorithms and the 6S codes (Vermote et al., 1997) for atmospheric correction. OTB uses the GDAL/OGR library (GDAL, 2010) to access image data, which provides a C++ and Python programming interface, command line and a graphical user interface (GUI; Monteverdi). The OSSIM image processing software primarily provides tools for registering and mosaicing image data but also includes capacity for visualising remotely sensed imagery (including in 3D). GRASS GIS provides extensive tools for a large range of spatial data processing including image, point cloud, vector and DEM processing and analysis, as well as tools for visualisation. GDAL provides support for accessing a wide range of image file formats and includes utilities for data management tasks, such as conversion between different formats and projections. TAUDEM provides tools for hydrologic processing of DEMs to create watersheds and stream networks. The RIOS library gives an easy to use interface to the GDAL library through Python, enabling the user to easily develop their own algorithms and provides access to the large number of algorithms available within Python libraries such as numpy and scipy. The SPDLib software is specifically written for the processing of airborne and terrestrial LiDAR data, including waveform data, and includes commands for the management and analysis of data. Additionally, there are a large number of specific tools and applications, such as proj.4, for managing coordinate systems and projections, Geometry Engine Open Source (GEOS) for processing geometries, libLAS for reading and writing the LAS file format used for LiDAR data, and PostGIS, which provides spatial extensions to the postgresql relational database.

The importance of Free and Open Source Software (FOSS) in remote sensing is recognised by initiatives such as the Open Source Geospatial Foundation (`http://www.osgeo.org`). Even manufacturers of commercial packages, such as ESRI, recognise the role of open source software in remote sensing (Kouyoumjian, 2011). For users in the research community, access to the cutting-edge algorithms available within open source packages is seen as a key benefit (Christophe and Inglada, 2009) as is the ability to view source code, providing better understanding and options to modify algorithms as required. Within the wider research community, the availability of source code to verify and recreate published results is seen as a standard researchers should be aiming for (Sonnenburg et al., 2007; Ince et al., 2012). However, producing software that is sufficiently generalist and user friendly to be of use to end-users can be a difficult and time consuming task. A decision must also be made about whether the software should be released as a stand-alone project or incorporated as an extension or plug-in to an existing project. These overheads often lead to software not being released, despite good intentions.

The motivation behind the development of RSGISLib was to support the requirements of research being undertaken by the authors, with particular emphasis on providing a framework for developing new and innovative solutions and making these algorithms available to the wider community. A key requirement was the ability to combine a number of commands using a common interface and batch process data. The ability to combine RSGISlib with the large amount of software and tools already available was an additional consideration. The paper outlines the software library structure (Section 2.1), highlights some of the available user commands (Section 2.2), and provides some examples of research that has benefited from using RSGISLib (Section 3). Finally, a discussion of the software and conclusions is provided.

2. The Remote Sensing and GIS Software Library

RSGISLib consists of sixteen C++ libraries and over 300 user commands. The user either interacts with RSGISLib through an XML or Python script, where each of the user commands is defined and parameterised using XML tags or, more recently, as a Python function. Rather than re-developing existing functionality independently, RSGISLib has aimed to interface with other open source software where appropriate. A list of pre-requisite software libraries used by RSGISLib is given in Table 1. The GDAL/OGR library provides a common interface to read and write image and vector files in the formats commonly used within the remote sensing community. The HDF5 and KEA (Bunting and Gillingham, 2013) libraries are also required to make direct use of specific features, not available through GDAL (e.g., additional attribute table features in the KEA format). Many mathematical algorithms in RSGISLib utilise the functions available in the GNU Scientific Library (GSL), with the Fastest Fourier Transform in the West (FFTW) library used to perform fast Fourier transforms. The GEOS and CGAL libraries provide a number of algorithms for geometry operations. The Xerces-C parser is used to parse the XML input file used within RSGISLib and muparser is used to parse the mathematical expressions used within the image and band maths tools.

Table 1: The pre-requisite software libraries of RSGISLib.

Pre-requisite	License	Website	Usage
		Input / Output	
GDAL/OGR	MIT-X	`http://www.gdal.org`	Reading and writing raster and vector datasets.
HDF5	BSD-style	`http://www.hdfgroup.org`	Reading and writing HDF5 files.
KEALib	MIT-X	`http://www.kealib.org`	Accessing extra features of the KEA format which are available through GDAL.
Xerces-C	Apache 2.0	`http://xerces.apache.org/xerces-c`	Parsing XML files.
		Mathematical Operations	
GNU Scientific Library (GSL)	GPL 3	`http://www.gnu.org/software/gsl`	Vector and Matrix operations and other common mathematical tools.
FFTW	GPL 2 or later	`http://www.fftw.org`	Fast Fourier transformation.
MuParser	MIT-X	`http://muparser.beltoforion.de`	Parsing and evaluating mathematical expressions.
		Geometry	
GEOS	LGPL 2.1	`http://trac.osgeo.org/geos`	Spatial geometry operations (e.g., intersect, buffer)
CGAL	GPL 3	`http://www.cgal.org`	Triangulations and interpolation.
Proj.4	MIT-X	`https://trac.osgeo.org/proj`	Presentation and transformation of projections.
		Other	
Boost	Boost	`http://www.boost.org`	Various utilities (e.g., text processing and save casting) and data structures (e.g., graphs).

2.1. Software Structure

The sixteen C++ libraries that make up RSGISLib partition the functionality into classification, data structures, geometry, image processing, raster GIS, image segmentation, mathematics, modelling, radar tools, image registration, image calibration, image filtering, vector processing, general utilities, a commands interface and common utilties (LiDAR processing functionality has been moved into the separate SPDLib software; Bunting et al., 2013a). The libraries within RSGISLib have a number of inter-dependencies (Figure 1), which govern the functionality that can be accessed between the libraries as there can be no cyclic dependencies for compilation.

2.1.1. Common

The common library provides some base classes, such as RSGISException from which the RSGISLib exception hierarchy originates. Additional functionality includes, a class to parser command line arguments (e.g., for the rsgisexe executable) and a class and abstract factory for parsing the individual commands within the XML scripts.

2.1.2. Data Structures

The data structures library currently provides just an implementation of a sorted list/queue but is the location where any other generic data structures could be implemented in the future.

2.1.3. Maths

The maths library provides classes and interfaces for mathematical operation used throughout the other libraries. These include polynomial fitting (based on the GSL library), random sampling from distributions and optimisation. Functions for reading/writing matrices and vectors to a text file are also provided in the maths library.

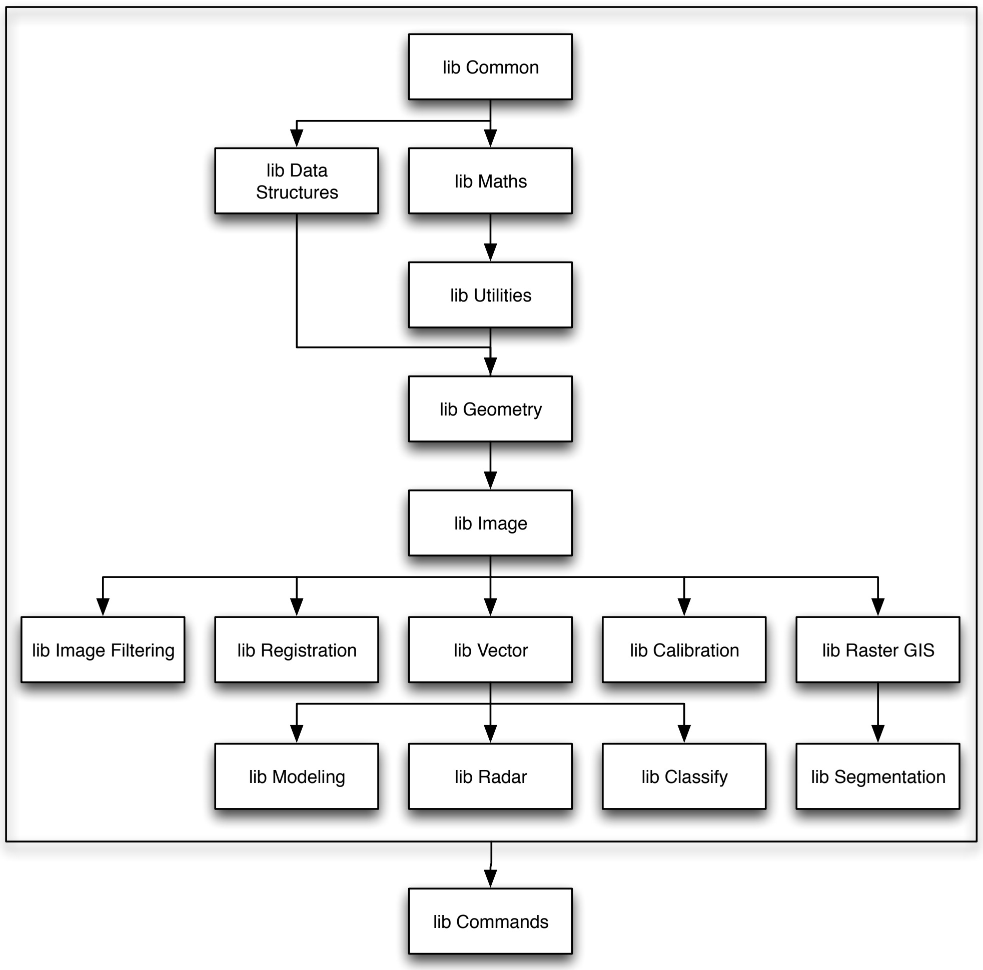

Figure 1: Dependency graph between the libraries which make up RSGISLib.

2.1.4. Utilities

The utilities library provides classes for some generic functionality such as parsing text file, manipulating strings and file system operations. There are also classes for importing and exporting data from the other libraries within RSGISLib for operations such as plotting and debugging.

2.1.5. Geometry

Functionality within the geometry class is largely built on the GEOS library but, in some cases, extends GEOS to provide further functionality, such as polygon overlap removal and calculating the length of any shared border between polygons. The geometry library also contains an implementation of the algorithms used within Bunting et al. (2010b) to find non-convex polygons from a group of smaller polygons/geometries.

2.1.6. Image

The image library contains tools and algorithms for image processing. One of the key components of this library is an abstract interface defined with the RSGISImageCalc class that provides an easy to use interface for looping through input images as blocks (the block size is read from the input data). This enables the processing of large datasets with a small memory footprint. Many of the image processing functions within RSGISLib are derived from the RSGISImageCalc class. All image I/O is undertaken through the GDAL library and therefore all image formats available through the specific GDAL installation are supported.

2.1.7. Calibration

The calibration library includes classes for converting optical remote sensing data from digital numbers (DN) to surface and top-of-atmosphere reflectance using published calibration coefficients and parameters derived from atmospheric modelling using 6S (Vermote et al., 1997).

2.1.8. Filtering

Within the filtering library is an interface for applying individual filters, or a bank of filters (e.g., the Leung-Malik filter bank; Leung and Malik, 2001) to images. Functions for applying morphological operations to images are also provided within this library.

2.1.9. Raster GIS

The raster GIS library contains functions for creating and manipulating raster attribute tables (RAT). The library also links to the KEA library (Bunting and Gillingham, 2013) to allow clump neighbours to be stored and used within processing. This is functionality which is not supported via GDAL. Functionality available within the raster GIS library allows RATs to be used for object-orientated classifications when interfaced with tools such as the Python RIOS library.

2.1.10. Segmentation

The segmentation library contains classes for performing image segmentation. The primary method available is that of Shepherd et al. (2013), which was implemented through a series of separate commands such as clumping and elimination. Whilst used in the segmentation, these independently provide tools that can be used within other contexts, thereby creating a more generic library.

2.1.11. Registration

The registration library contains classes for generating tie points between images and warping images based on tie points. The algorithm for tie point generation is a version of the method presented in Bunting et al. (2010a). The algorithm assumes an initial alignment and uses the correlation coefficient to match windows from the floating image to that of the base image. A network can be used to constrain tie point locations to reduce outliners and improve tie point coverage.

2.1.12. Vector

The vector library contains classes for processing OGR supported vectors, primarily shapefiles. Functions are also provided for performing pixel-in-polygon analysis (e.g., zonal statistics). Functionality includes conversion between OGR and GEOS geometries, buffering, merging and morphological operations.

2.1.13. Classify

The classification library contains algorithms for supervised (e.g., spectral angle mapper and minimum distance) and unsupervised (e.g., K-Means) image classification. A command for applying a rule-based classification to a set of vectors using SQL statements is also available. There are also functions for removing single classified pixels and collapsing RATs based on a classification (i.e., turning a classified segmentation into a classification).

2.1.14. Modelling

The modelling library contains classes for the generation of a voxel based vegetation transect and functions to estimate Foliage Projective Cover (FPC; Armston et al., 2009) and canopy cover (CC). These estimates are derived from an implementation of the FPC model proposed in Clewley (2012).

2.1.15. Radar

The radar library contains algorithms for processing Synthetic Aperture Radar (SAR) data. These include an implementation of the above ground biomass estimation algorithm of Saatchi et al. (2007). Additionally, more advanced techniques such as the non-linear estimation algorithm of Moghaddam and Saatchi (1999), including the object-based adaptations proposed in Clewley (2012), have also been implemented to provide an estimation of height and cover.

2.1.16. Commands

The commands library contains a simplified interface of C++ functions which correspond to the XML commands and Python functions. These functions provide an easy method for new interfaces to the RSGISLib software to be developed, or for RSGISLib commands to be included in other projects. For example, the Python binding for RSGISLib is being developed using this library. The library has been specifically developed to provide a clean interface and does not require knowledge of the RSGISLib pre-requisites to compile against and include in other C++ projects.

2.2. Users Commands

A single command line program is used for RSGISLib (`rsgisexe`). The commands specifying the algorithms and options to be executed are embedded within an XML file with the following structure.

```
<?xml version="1.0" encoding="UTF-8" ?>
<rsgis:commands xmlns:rsgis="http://www.rsgislib.org/xml/">
    <rsgis:command algor="XXXXX" option="XXXXX" ... ... />
    <rsgis:command algor="XXXXX" option="XXXXX" ... ... />
    <rsgis:command algor="XXXXX" option="XXXXX" ... ... />
</rsgis:commands>
```

Commands within the XML file are defined within a `rsgis:commands` tag and listed as individual `rsgis:command` tags where the '`algor`' and '`option`' attributes specify the command within RSGISLib to be executed. The parameters for the command are provided as attributes or child tags to the `rsgis:command` tag.

Alternatively, and where available, commands can be executed as Python functions where the options provided within the XML tags are provided as options to the Python functions. To date, over 100 commands are available within the Python binding.

To provide some structure to the user and to help them find commands, the XML commands have been broken down into 23 subsections (`algor`; Table 2). The Python binding uses the same structure where each of the required subsections are represented as Python modules.

Table 2: List of user command sections

Section	Description
Biomass	Implementation of published algorithms for the estimation of biomass from SAR data.
Classification	Classification related commands (e.g., Spectral Angle Mapper).
Command Line	Allows any command line tool to be executed.
Elevation	Tools for deriving products from a DEM.
Estimation	Estimation of parameters from SAR data.
Fitting	Fits models to data.
Image Calculations	Tools such as band and image maths, also includes other commands that calculate values between (e.g., correlation) or from images (e.g., speed of movement between range images).
Image Calibration	Calibrate optical data to reflectance.
Image Conversion	Convert image between formats.
Image Filtering	Apply filters to image data.
Image Morphology	Apply morphological filters to image data.
Image Registration	Generate and warp to tie points.
Image Segmentation	Segment images to generate a set of clumps.
Image Utilities	General image utilities, such as populating statistics and pyramids, stretch image values, create tiling and mosaic images together.
Maths Utilities	Apply some matrix operations and calculate a PCA transformation.
Post Classification Vector	Combine and build large regions from classified polygons.
Raster GIS	Create and populate information into a raster attribute table.
Stack Bands	Tools for stacking images.
Test Images	Tools for creating a test images.
Vector Utilities	Apply operations on the geometries and attribute tables of the vector datasets. For example, buffering the geometries or performing a ‘find and replace’ on text in the attribute table columns.
Visualisation	A set of commands to export data, as a text file, for plotting using rsgislib-plot.py.
Zonal Statistics	Tools for extracting pixel statistics within a polygon.

2.3. Additional Tools

Included as part of RSGISLib are several Python scripts to aid batch processing and export of plot data for visualisation through the utilities library. The batch processing script allows a user to define a template with the commands they wish to use. As an example, a batch script might be used to generate a new XML file to run these commands populated with the appropriate file names for each image in a directory.

3. Software Highlights

There are currently over 300 commands within RSGISLib to process raster and vector data. Whilst some of these tools replicate similar functionality to those available in other packages (e.g., image stacking, band maths) albeit with improvements/specialisations, a number of tools are unique to RSGISLib and are the product of research by the authors and their collaborators. A selection of tools, which the authors feel offer significant improvements over existing software packages, are detailed below as examples of the type of functionality and capability available.

3.1. Zonal Statistics

For developing and validating algorithms using remotely sensing data, statistics from pixels falling within a zone (e.g., study site, field plot) are often required. These 'zones' do not always line up exactly with the pixels in the remote sensing data. Therefore, a criteria must be established for determining which pixels to include. To reduce random noise, it is desirable to take the average from a larger number of pixels. However, when zones are not in homogeneous areas, including pixels from different cover types, the average may be skewed. Currently there

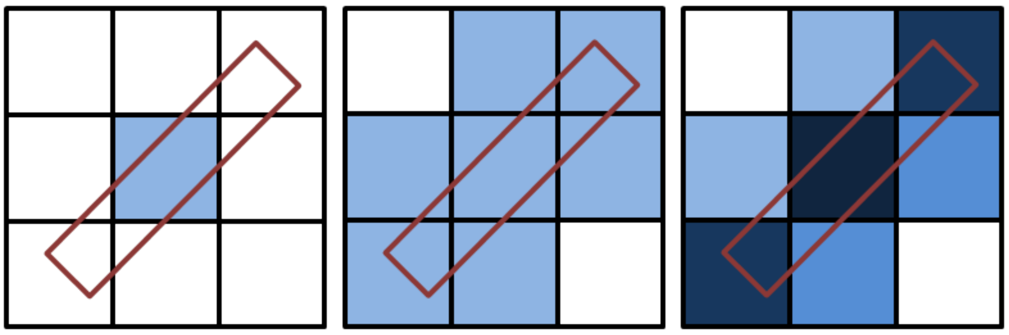
(a) Polygon contains pixel centre (b) Polygon intersects pixel (c) Weighted mean

Figure 2: Different methods used for determining pixel inclusion within a polygon when calculating zonal statistics. In a and b blue pixels represent those that would be included when considering a) only pixels with the centre contained within the polygon, b) pixels that intersect the polygons. An alternative method, the pixel weighted mean, is shown in c, the shade represents the relative weighing of each pixel, with darker colours representing higher weighting.

are tools available for extracting zonal statistics within a number of software packages. These use different criteria to determine pixel inclusion, such as pixel centre within polygon (Figure 2.a; this is the default in ArcMap; ESRI, 2007) or percent of pixel within polygon (Figure 2.b), as used in StarSpan (Rueda et al., 2005), with variable percentages. This is an option in ArcMap, with the percentage fixed at 50 %.

The zonal statistics commands in RSGISLib use a number of methods for determining pixel inclusion within a polygon. The default is that the pixel centre is within the polygon, as this takes the least time to run. In addition to boolean inclusion criteria, a fuzzy approach is also available (Figure 2.c) in which the percent of a pixel contained within each polygon is used as the weighting for calculating the mean:

$$\bar{x} = \frac{\sum x_i \kappa}{\sum \kappa}, \tag{1}$$

where κ is the percent of the pixel that intersects the polygon (e.g., when the polygon is completely contained within the pixel, $\kappa = 1$).

A polygon representation of the image pixels is used, allowing RSGISLib to build on the geometry operations available within GEOS to determine pixel inclusion. When using the standard zonal statistics algorithm, a number of statistics are available (mean, min, max, standard deviation and mode, for categorical data). Pixel values may also be exported as a text file for more detailed statistical analysis. The XML interface allows statistics to be exported from all or selected bands in an image using generic band names (b1, b2...,etc.,), the band names from the image, or user specified names. When specifying bands, the statistics for each band and minimum/maximum thresholds for pixel inclusion may be defined. Statistics may be exported to a ESRI Shapefile (optionally retaining the attributes of the input file) or CSV file. To improve speed, in particular when repeatedly calculating statistics from the same set of polygons, a rasterised version of polygons may also be supplied.

3.2. Image Texture Classification / Segmentation

Within the image processing community, approaches utilising image texture alongside colour (reflectance) for classification and/or segmentation have commonly been sought (e.g., Haralick et al., 1973) as the use of textural information more closely corresponds to the human vision system. More recently, approaches such as textons (Leung and Malik, 2001) have been explored. These methods aim to extract the constitute parts of the texture, similar to the unmixing of reflectance data. Within RSGISLib, a new method of applying a textons-based approach for image analysis has been implemented (Bunting et al., 2009), where a Principle Component Analysis (PCA) has been used to defined the linear combination of filter responses.

The textons-based approach is based on the application of a number of image filters, such as Laplacian edge filters, Gaussian derivative filters (i.e., first and second derivatives) and smoothing filters. When these filters are used at a number of angles and filter sizes, a bank of filters is defined. Leung and Malik (2001) defined a filter bank, which has been used for many studies (Figure 3) consisting of 48 filters, including eight Laplacian of Gaussian, four Gaussian smoothing filters and six Gaussian first and second derivative filters at three scales. For the Laplacian of Gaussian filters scales of 1, $\sqrt{2}$, 2, $2\sqrt{2}$, 3, $3\sqrt{2}$, 6 and $6\sqrt{2}$ are used while the Gaussian smoothing use scales of 1, $\sqrt{2}$, 2 and $2\sqrt{2}$. For the Gaussian first and second derivative filters scales of $\left(\sigma_x, \sigma_y\right)$, $(1, 3)$, $\left(\sqrt{2}, 3\sqrt{2}\right)$ and $(2, 6)$ are used, where each scale is rotated by 0, 30, 60, 90, 120 and 150 degrees.

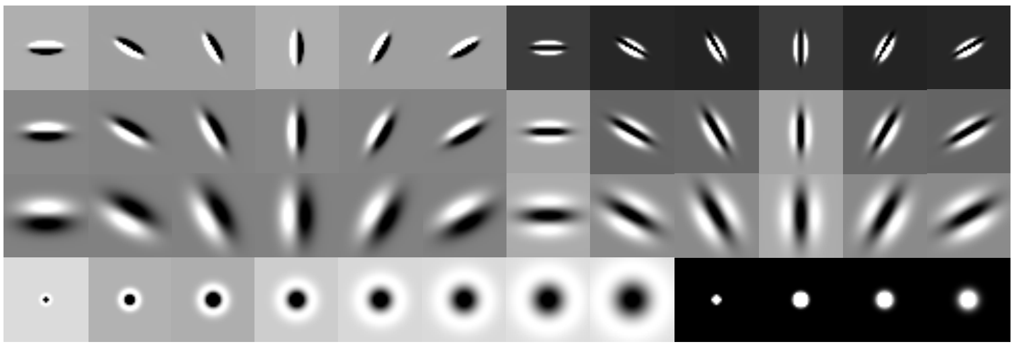

Figure 3: The Leung and Malik (2001) filter bank used extract textural features from the images.

Within RSGISLib, a generic command for defining and applying filter banks has been developed, where a list of filters within the filter bank is specified using the command's XML script. Additionally, published filters banks, such as Leung and Malik (2001), have been hard coded and can be executed using a single command.

Following the application of the filter bank to the image, training regions are identified, with each corresponding to a textural region of interest. PCA is performed on each of these regions, and the eigenvectors for each region are applied back to the whole dataset, retaining only the highest three principle components. This results in a three band image for each texture, where each band is the product of a linear texton. Using a nearest neighbour classifier (also within RSGISLib) and the training regions previously defined, the image is classified. An example result is shown in Figure 4, where a 2.6 m resolution hyperspectral HyMap data, flown over the Injune (Queensland, Australia) study site in 2000 (Lucas et al., 2008) has been classified. To reduce the number of generated filter responses, only three (446.1, 716.2 and 891.2 nm) of the 126 wavelengths available in the HyMap data were used, with these selected as they demonstrated the best visual differentiation of the classes of interest. The result demonstrates a good correspondence between the textures identified and those picked out by a human operator.

3.3. Objected Oriented Image Classification

Classification has traditionally been performed at a pixel by pixel level but over the last 10 years there has been a significant movement to embrace context and segment-based classifications (Lucas et al., 2007, 2011) because of observed improvements in classification accuracy (Lobo, 1997; Stuckens et al., 2000; Fuller et al., 2002; Tarabalka et al., 2009; Blaschke, 2010) and region boundaries. A significant driver for this adoption has been the availability of the eCognition software (Definiens, 2005). Therefore the development of tools, which allow these techniques to be applied within a flexible open source environment, is highly desirable.

The segmentation algorithm of Shepherd et al. (2013) has been implemented in RSGISLib and can be used to segment data to create image objects. The algorithm (Figure 5) uses an implementation of K-Means clustering to generate seeds for the segmentation where, following the assignment of the pixels to the associated cluster centre, the clumps are iteratively eliminated if they are below the minimum mapping unit threshold to the neighbouring clump that is spectrally closest. Following elimination, the final clumps are relabelled to ensure they are consecutively

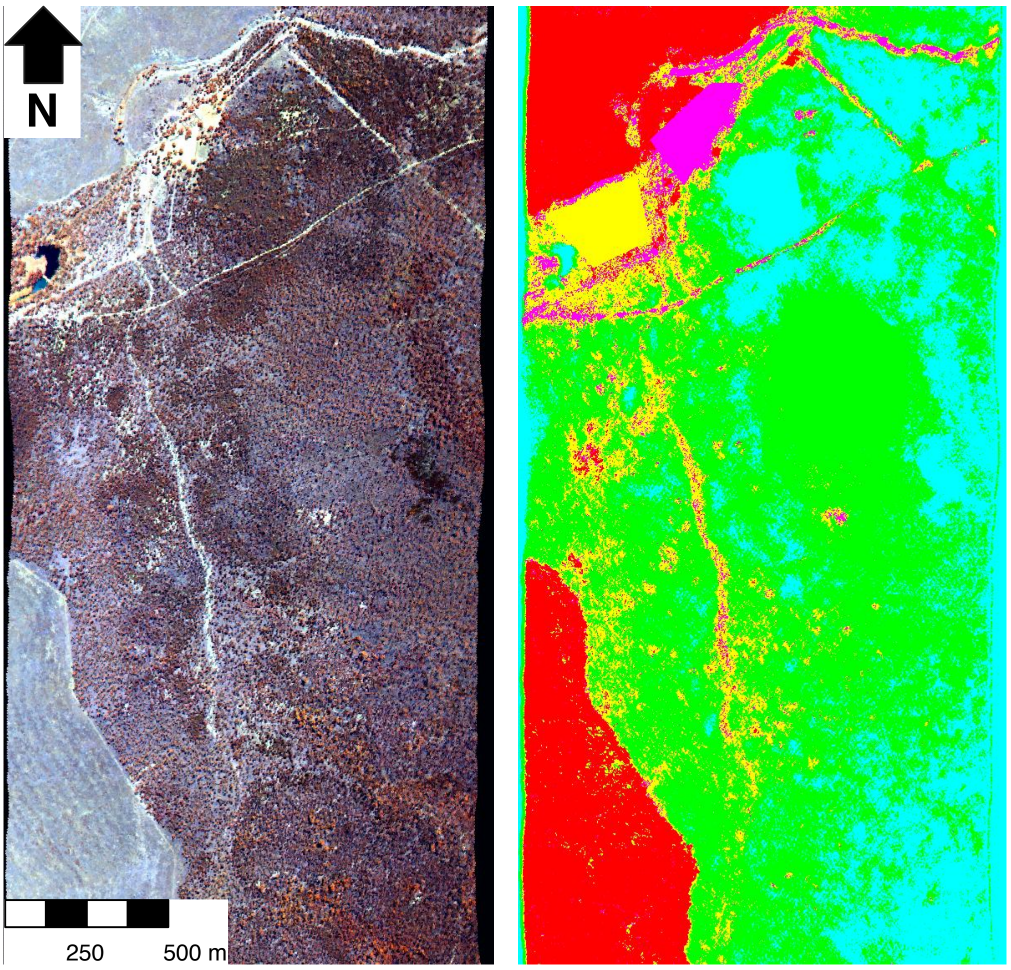

Figure 4: A classification result from the supervised texton classification process.

numbered, where a value of zero defines no data regions.

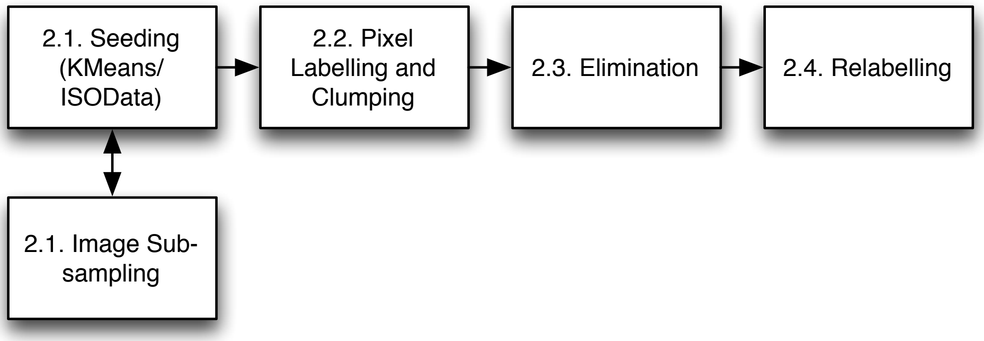
2.1. Seeding (KMeans/ ISOData)
2.2. Pixel Labelling and Clumping
2.3. Elimination
2.4. Relabelling
2.1. Image Sub-sampling

Figure 5: A flowchart of the Shepherd et al. (2013) segmentation algorithm.

The current implementation allows for large multi-gigabyte images to be processed at the same time as the (K-Means) clustering uses a sampled version of the input image. For example, a mosaic of 4 band SPOT data with 36500 × 35600 (1.3 billion) pixels covering a region of South Island New Zealand was segmented using just 12 GB of RAM in 3 hours (Shepherd et al., 2013). Being able to perform segmentation on the entire dataset overcomes the problem of edge effects introduced when the dataset is split into smaller tiles for processing.

Following segmentation and identification of individual clumps, those clumps need to be processed to produce a result (e.g., classification). Within RSGISLib, raster clump files with attribute tables (RAT) are used to represent the segments and their attributes (e.g., mean reflectance, topographic elevation, etc.). With the development of the KEA image file format (Bunting and Gillingham, 2013), attribute tables are compressed providing a manageable and GDAL compatible method of storing the data.

RSGISLib provides commands to populate objects with statistics from image bands and shape information (e.g., shape index). Objects may be attributed with additional indices or classified using tools and libraries available within Python such as scipy (Jones et al., 2001–), scikit-learn (Pedregosa et al., 2011), machine learning Python (mlp; Albanese et al., 2012) and Pandas (Python Data Analysis Library; `http://pandas.pydata.org`). To access the RAT columns, the RIOS (`https://bitbucket.org/chchrsc/rios`) Python library is used where the attribute table columns are presented as Numpy arrays (`http://www.numpy.org`). The approach is particularly effective when combined with the TuiView image viewer (`https://bitbucket.org/chchrsc/tuiview`) which allows the attribute table to be viewed, expressions evaluated and results visualised. When the functionality of RSGISLib is combined with RIOS and other Python libraries, a highly flexible system is created, which is comparable to and is, in some cases, an advancement on the functionality available within commercially available software.

Figure 6 presents results of a rule-based classification of the growth stages within a native Australian forest ecosystem dominated by Brigalow in Southern Queensland (*Acacia harpophylla* Lucas et al., 2013). The classification was undertaken using a combination of 30 m Landsat derived Foliage Projective Cover (FPC; Armston et al., 2009) and Advanced Land Observing Satellite (ALOS) Phased-array L-band SAR (PALSAR) data. The scene was 25200×44200 (1.1 billion) pixels in size.

3.4. Image Registration

When combining data from multiple sources, it is important that all datasets are accurately co-registered. Within RSGISLib, the image registration algorithm of Bunting et al. (2010a) is available to automatically generate tie points between images. One image is assumed to be correctly geolocated (the reference image) and the other image (the floating image) is registered to this. The algorithm generates a connected grid of tie points where the inverse weighted distance of the

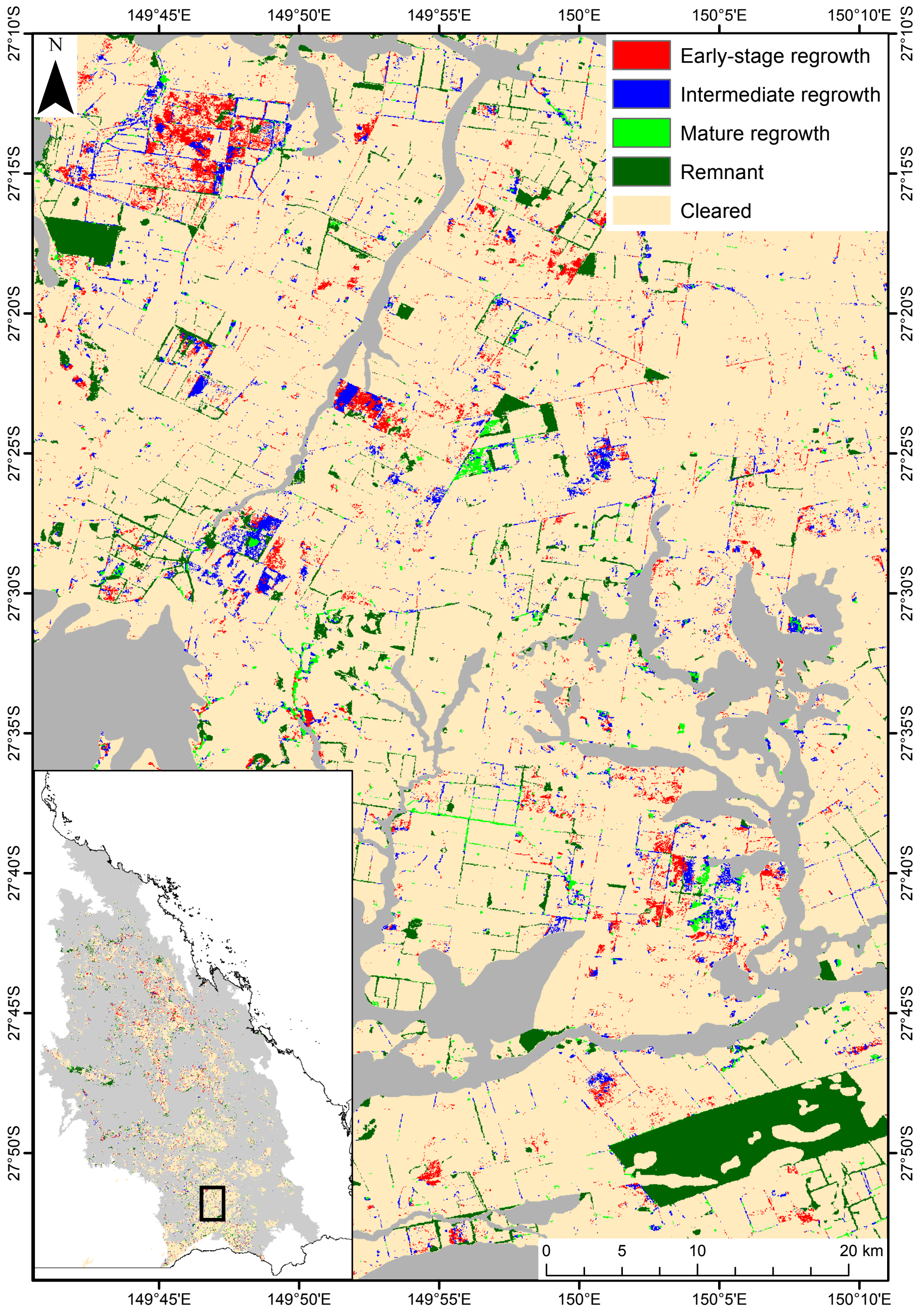

Figure 6: A object oriented classification of regrowth Brigalow in Queensland, Australia (Lucas et al., 2013) using RSGISLib and Python.

movement of a current tie point is applied to the connected tie points (within a given distance). The approach has demonstrated improved performance over existing algorithms, such as those found in ITK, in particular with multi-modal remote sensing data (Bunting et al., 2010a). A number of metrics are available to compare images; correlation has been shown to perform best for multi-model data but Euclidian distance, Manhattan distance or square difference are also included with RSGISLib. A threshold is used to exclude points with low correlation, either in areas without distinct features or where there has been change between the two images. Images may be warped from the generated tie points using nearest neighbour, triangular or polynomial warping. Tie points may also be added to a GDAL dataset for warping using the `gdalwarp` command, allowing warping using thin plate splines.

Tie points may also be used to evaluate the co-registration between two datasets. This is particularly important when evaluating products from different providers. When the tie point generation algorithm is combined with the image tiling commands, tie points may be generated over very large areas. In Figure 7, an example is presented showing tie points generated for evaluating national level remote sensing data in Australia. The datasets had a spatial resolution of 30 m, and the entire mosaic comprised over 143000 $\times$ 164000 (23 billion) pixels. Using an 8-core 3.6 GHz Intel Xeon machine running Ubuntu 12.04, over 40,000 tie points were generated in under eight hours by splitting the image up into tiles (using RSGISLib) and processing a separate tile on each core. Tie points in areas without distinct features (such as desert areas in the centre of the country) were automatically removed as part of the algorithm because of their low correlation.

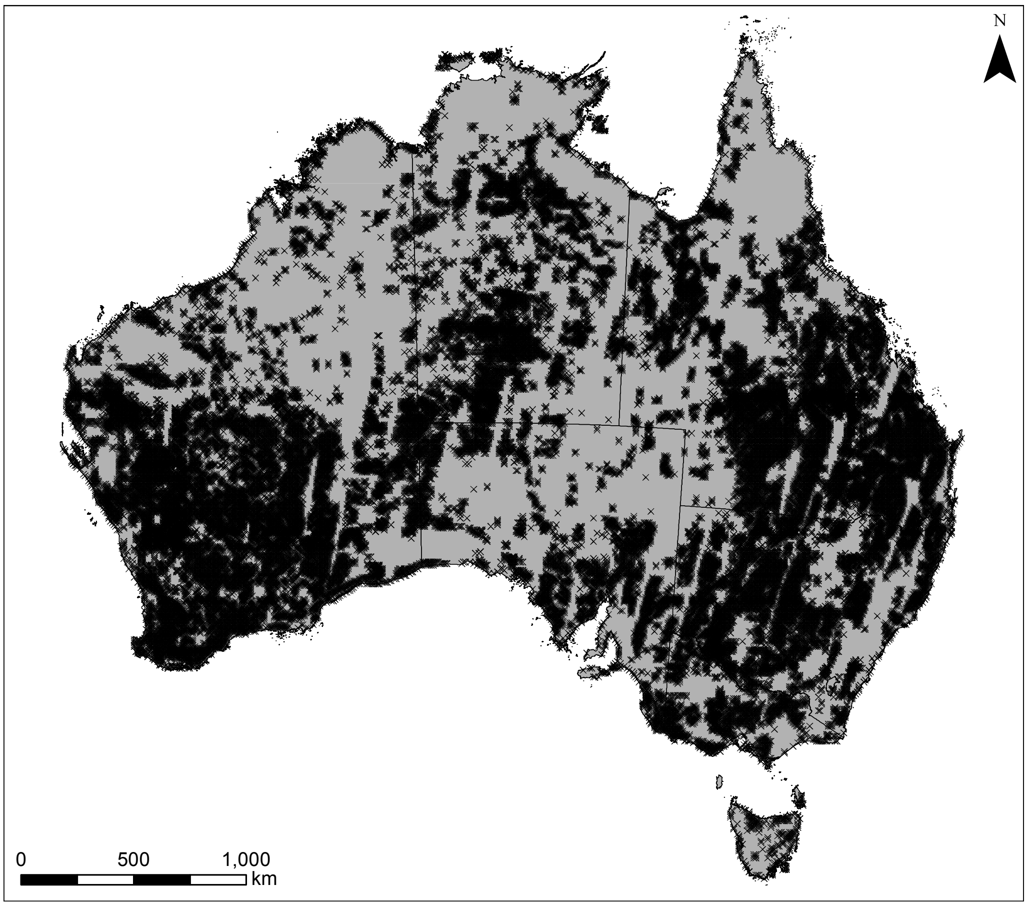

Figure 7: Tie points generated across Australia to evaluate the registration accuracy of national datasets.

4. Discussion

4.1. Supported Research

RSGISLib has been used to support a number of research projects. An overview of these, the functions they have used and publications arising are provided in Table 3. Some used existing functionality within RSGSLib (e.g., Clewley et al., 2012a; Lucas et al., 2013), whilst others exploited existing libraries in RSGISLib (such as the image library) to implement new algorithms (e.g., Bunting et al., 2010b). It should be noted that, for some papers, functionality that could have been used was not available at the time the articles were published. For example in Clewley et al. (2012a), eCognition was used for image segmentation as that feature was yet to be added into RSGISLib. However, for application of the technique to a larger area in (Lucas et al., 2013), the segmentation and object-oriented image classification in RSGISLib was used as it avoided the requirement for splitting the data into tiles and allowed the approach to be more automated. In addition to the selection of work presented in Table 3, RSGISLib has been used as part of a number of smaller projects and dissertations but is being progressively adopted for use in regional studies relating to, for example, mangrove change detection and biomass estimation.

Table 3: A selection of research using RSGISLib and the functions utilised.

		RSGISLib Functions																						
Research	**Publications**	Biomass	Classification	Command Line	Elevation	Estimation	Fitting	Image Calculations	Image Calibration	Image Conversion	Image Filtering	Image Morphology	Image Registration	Image Segmentation	Image Utilities	Maths Utilities	Modelling	Post Classification	Raster GIS	Stack Bands	Test Images	Vector Utilities	Visualisation	Zonal Statistics
Regrowth stage mapping	Clewley et al. (2012a)			x				x							x					x		x		x
	Lucas et al. (2013)							x					x	x	x				x	x		x		x
Forest structure estimation from SAR data	Clewley et al. (2010)					x	x	x			x				x					x				x
	Clewley (2012)	x		x		x	x	x			x				x		x			x	x	x	x	x
	Clewley et al. (2012b)					x	x																	
Segmentation of New Zealand SPOT mosaics for change detection								x						x	x				x					
BioSOS	`http://www.biosos.wur.nl/UK`		x	x	x			x	x		x			x	x				x					
Textons	Bunting et al. (2009)		x								x					x				x				x
Tree Crown Clustering	Bunting et al. (2010b)		x															x				x		
Habitat mapping	Lucas et al. (2011)		x												x							x		x
	Breyer (2010)		x												x							x		x

4.2. Abstract Frameworks

The libraries were all developed using object-orientated abstract class design frameworks, simplifying the addition of new and advanced functionality. Moreover, the frameworks were implemented to provide support for large dataset handling and to use memory efficiently. These capabilities are therefore inherited by all new commands that use these frameworks, allowing additional functions (e.g., new image filters, image warping) to be implemented efficiently and integrated within the wider project. A standard user interface familiar to users is provided and consistent comparison between algorithms is facilitated.

4.3. Support for High Performance Computing environments

Although RSGISLib runs single threaded, it is being actively used within High Performance Computing (HPC) environments and particularly where a large number of images are processed simultaneously. Within the field of remote sensing, processing of multiscene datasets over large areas is a common requirement. When processing a single, large scene, the image can be split into tiles (with optional overlaps) for processing on separate nodes and then re-mosaiced.

4.4. Comparison with other software

A significant difference between RSGISLib and other packages (e.g., OTB) is the use of an XML interface rather than providing separate command line tools for different options. This gives greater flexibility in the parameterisation of algorithms than is possible using command line options. However, the advantage of command line tools is they are easier to include within scripts (e.g., bash, Python) and to loop through parameters. Scripts are made available within RSGISLib to create XML files for different variables based on a template. For those users who prefer to specify command line options, many (but not all) commands are accessible using an included Python utility (`rsgiscmd`) which generates and executes a temporary XML file.

There are currently no plans to develop a GUI interface for RSGISLib. A number of existing open source tools (e.g., QGIS) have intuitive GUIs that provide the user with an easy way to access a number of tools for processing and visualising data. However, GUI tools are not ideal for batch processing or creating processing chains. Therefore, the development of RSGISLib has focused on these areas. The software can be used in combination with visualisation tools such as TuiView (`http://tuiview.org`) or QGIS (`http://qgis.org`) to visualise the results and input data.

4.5. Software License

The libraries are released under a General Public License (GPL) version 3 license (GNU, 2007). Whilst the software is provided completely free of charge, it is without a warranty or promise of support. The GPL license is a so called 'viral license' meaning that any works derived from this software must be released under a compatible license. Therefore, the source code of any improvements or changes must be made available without any charge.

4.6. Future Developments

In the short term, a Python binding to complement the XML interface is under development with many of the most used commands already made available as Python functions. In the longer term, new functionality and improvements to existing functions will be added as and when needed for our ongoing research. Expected improvements include further methods for image segmentation and feature extraction, change detection and classification with particular focus on environmental remote sensing application and vegetation studies.

5. Conclusions

An open source software library for use with Remote Sensing and GIS data has been developed and made publicly available under a GPL license. Key features of the library are:

- A common XML interface providing access to over 300 commands and enabling the development of processing chains and batch processing.
- A number of specialised algorithms for image registration, zonal statistics and object based image classification that represent improvements on those available in other packages.
- An object-based software design allowing new algorithms to be quickly added.

The library has been used in a number of publications and continues to be actively developed supporting ongoing research across a wide range of topics including vegetation science. New functionality is frequently added and made publicly available through a publicly-accessible code repository.

Acknowledgements

The authors would like to thank the developers of the software libraries which RSGISLib depends on. Landcare Research, New Zealand, are thanked for their support for Pete Bunting during his period of employment (2011–2012) in New Zealand during which time RSGISLib was further developed. Mahta Moghaddam is thanked for supporting Dan Clewley (2012 –) and the Natural Environment Research Council (NERC) for providing funding for his PhD (2009 – 2012). John Armston and James Shepherd are thanked for providing ideas, improvements and bugs fixes. Additionally, all the students and collaborators who have used, tested and reported bugs in RSGISLib are also thanked.

References

Albanese, D., Visintainer, R., Merler, S., Riccadonna, S., Jurman, G., Furlanello, C., 2012. mlpy: Machine learning python.

Armston, J. D., Denham, R. J., Danaher, T. J., Scarth, P. F., Moffiet, T. N., 2009. Prediction and validation of foliage projective cover from Landsat-5 TM and Landsat-7 ETM+ imagery. Journal of Applied Remote Sensing 3 (1), 1–28.

Blaschke, T., 2010. Object based image analysis for remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing 65 (1), 2–16.

Breyer, J., 2010. Habitat classification using airborne and spaceborne remote sensing for biodiversity assessment in Wales. Ph.D. thesis, University of Wales, Aberystwyth.

Bunting, P., Armston, J., Clewley, D., Lucas, R. M., 2013a. Sorted pulse data (SPD) library – Part II: A processing framework for LiDAR data from pulsed laser systems in terrestrial environments. Computers & Geosciences 56, 207–215.

Bunting, P., Armston, J., Lucas, R. M., Clewley, D., 2013b. Sorted pulse data (SPD) library – Part I: A generic file format for LiDAR data from pulsed laser systems in terrestrial environments. Computers & Geosciences 56, 197–206.

Bunting, P., He, W., Zwiggelaar, R., Lucas, R., 2009. Combining textural and hyperspectral information for the classification of tree species in Australian savanna woodlands. In: Lecture Notes in Geoinformation and Cartography: Innovations in Remote Sensing and Photogrammetry, S. Jones and K. Reinke (Eds.), Springer, 19–26.

Bunting, P. J., Gillingham, S., 2013. The KEA image file format. Computers & Geosciences 57 (C), 54–58.

Bunting, P. J., Labrosse, F., Lucas, R. M., 2010a. A multi-resolution area-based technique for automatic multi-modal image registration. Image and Vision Computing 28 (8), 1203–1219.

Bunting, P. J., Lucas, R. M., Jones, K., Bean, A. R., 2010b. Characterisation and mapping of forest communities by clustering individual tree crowns. Remote Sensing of Environment 114 (11), 2536–2547.

Christophe, E., Inglada, J., 2009. Open source remote sensing: Increasing the usability of cutting-edge algorithms. IEEE Geoscience Remote Sensing Society Newsletter March.

Clewley, D., 2012. Retrieval of Forest Structure and Biomass From Radar Data Using Backscatter Modelling and Inversion. Ph.D. thesis, Aberystwyth University.

Clewley, D., Lucas, R. M., Accad, A., Armston, J., Bowen, M. E., Dwyer, J., Pollock, S., Bunting, P. J., McAlpine, C., Eyre, T., Kelly, A., Carreiras, J., Moghaddam, M., Aug. 2012a. An Approach to Mapping Forest Growth Stages in Queensland, Australia through Integration of ALOS PALSAR and Landsat Sensor Data. Remote Sensing 4 (8), 2236–2255.

Clewley, D., Lucas, R. M., Moghaddam, M., Bunting, P. J., 2012b. The effects of noise on model inversion for the retrieval of forest structure from SAR data. Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International.

Clewley, D., Lucas, R. M., Moghaddam, M., Bunting, P. J., Dwyer, J., Carreiras, J., 2010. Forest parameter retrieval from SAR data using an estimation algorithm applied to regrowing forest stands in Queensland, Australia. Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International, 1238–1241.

Definiens, 2005. eCognition Version 5 Object Oriented Image Analysus User Guide. Tech. rep., Munich, Germany: Definiens AG.

ESRI, 2007. ArcGIS Version 9.3. Environmental Systems Research Institute, Redlands, California.

Fuller, R. M., Smith, G. M., Sanderson, J., Hill, R., Thomson, A., 2002. The UK Land Cover Map 2000: construction of a parcel-based vector map from satellite images. Cartographic Journal, The 39 (1), 15–25.

GDAL, 2010. Geospatial Data Abstraction Library (GDAL).
URL `http://www.gdal.org`

GNU, 2007. GNU General Public License (GPL) Version 3. (http://www.gnu.org/copyleft/gpl.html).

Haralick, R. M., Shaummugam, K., Dinstein, I., 1973. Texture features for image classification. IEEE Transactions on Systems, Man and Cybernetics 3, 610–621.

Ince, D. C., Hatton, L., Graham-Cumming, J., Feb. 2012. The case for open computer programs. Nature 482 (7386), 485–488.

Inglada, J., Christophe, E., 2009. The Orfeo Toolbox remote sensing image processing software. In: 2009 IEEE International Geoscience and Remote Sensing Symposium. pp. IV–733–IV–736.

Jones, E., Oliphant, T., Peterson, P., others, 2001–. SciPy: Open source scientific tools for Python.
URL `http://www.scipy.org/`

Kouyoumjian, V., October 2011.
URL `http://blogs.esri.com/esri/esri-insider/2011/10/24/open-source-closed-source-moving-to-the-middle/`

Leung, T., Malik, J., 2001. Representing and recognizing the visual appearance of materials using three-dimensional textons. International Journal of Computer Vision 43 (1), 29–44.

Lobo, A., 1997. Image segmentation and discriminant analysis for the identification of land cover units in ecology. Geoscience and Remote Sensing, IEEE Transactions on 35 (5), 1136–1145.

Lucas, R., Bunting, P., Paterson, M., Chisholm, L., Jan. 2008. Classification of Australian forest communities using aerial photography, CASI and HyMap data. Remote Sensing Of Environment 112 (5), 2088–2103.

Lucas, R., Medcalf, K., Brown, A., Bunting, P., Breyer, J., Clewley, D., Keyworth, S., Blackmore, P., Jan. 2011. Updating the Phase 1 habitat map of Wales, UK, using satellite sensor data. ISPRS Journal of Photogrammetry and Remote Sensing 66 (1), 81–102.

Lucas, R., Rowlands, A., Brown, A., Keyworth, S., Bunting, P., Jan. 2007. Rule-based classification of multi-temporal satellite imagery for habitat and agricultural land cover mapping. ISPRS Journal of Photogrammetry and Remote Sensing 62 (3), 165–185.

Lucas, R. M., Clewley, D., Accad, A., Butler, D., Armston, J., Bowen, M., Bunting, P., Carreiras, J., Dwyer, J., Eyre, T., Kelly, A., McAlpine, C., Pollock, S., Seabrook, L., 2013. Mapping Forest Growth Stage in the Brigalow Belt Bioregion of Australia through integration of ALOS PALSAR and Landsat-derived Foliage Projective Cover (FPC) data. Accepted for Publication in Remote Sensing of Environment.

Moghaddam, M., Saatchi, S. S., 1999. Monitoring tree moisture using an estimation algorithm applied to SAR data from BOREAS. IEEE Transactions on Geoscience and Remote Sensing 37 (2), 901–916.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Pottier, E., Ferro-Famil, L., Allain, S., Cloude, S., Hajnsek, I., Papathanassiou, K., Moreira, A., Williams, M., Minchella, A., Lavalle, M., Desnos, Y.-L., 2009. Overview of the PolSARpro V4.0 software. the open source toolbox for polarimetric and interferometric polarimetric SAR data processing. Geoscience and Remote Sensing Symposium,2009 IEEE International,IGARSS 2009 4.

Rueda, C., Greenberg, J., Ustin, S., 2005. StarSpan: A Tool for Fast Selective Pixel Extraction from Remotely Sensed Data. Center for Spatial Technologies and Remote Sensing (CSTARS), University of California at Davis, Davis, CA.

Saatchi, S. S., Halligan, K., Despain, D., Crabtree, R., 2007. Estimation of Forest Fuel Load From Radar Remote Sensing. IEEE Transactions on Geoscience and Remote Sensing 45 (6), 1726–1740.

Shepherd, J., Bunting, P., Dymond, J., 2013. Segmentation of imagery based on iterative elimination. Remote Sensing Submitted.

Sonnenburg, S., Braun, M. L., Ong, C. S., Bengio, S., Bottou, L., Holmes, G., LeCun, Y., Müller, K.-R., Pereira, F., Rasmussen, C. E., 2007. The need for open source software in machine learning. Journal of Machine Learning Research 8, 2443–2466.

Stuckens, J., Coppin, P., Bauer, M., 2000. Integrating contextual information with per-pixel classification for improved land cover classification. Remote Sensing Of Environment 71 (3), 282–296.

Tarabalka, Y., Benediktsson, J., Chanussot, J., 2009. Spectral-Spatial Classification of Hyperspectral Imagery Based on Partitional Clustering Techniques. IEEE Transactions of Geoscience and Remote Sensing 47 (8), 2973–2987.

Vermote, E., Tanre, D., Deuze, J., Herman, M., Morcrette, J., Jan. 1997. Second Simulation of the Satellite Signal in the Solar Spectrum, 6S: An overview. IEEE Transactions on Geoscience and Remote Sensing 35 (3), 675–686.