A File Structure and Reference Data Set for High Resolution Hyperspectral 3D Point Clouds

Thomas Wiemann * **Felix Igelbrink** * **Sebastian Pütz** * **Joachim Hertzberg**[*,**]

* *Knowledge Based Systems Group, University of Osnabrück, Osnabrück, Germany (firstname.lastname@uni-osnabrueck.de)*
** *DFKI Robotics Innovation Center, Osnabrück Branch, Osnabrück, Germany (e-mail: ric-osnabrueck@dfki.de)*

Abstract: Hyperspectral imaging has been extensively studied in remote sensing. In this community, several approaches exist for classifying different organic and an-organic materials. However, this data is usually collected from large distances (flight or satellite data) and hence lacks geometric precision, which is required for robotic applications like mapping and navigation. In this paper, we present a reference data set that maps hyperspectral intensity data to a terrestrial 3D laser scanner to generate what we call hyperspectral point clouds (HPCs). To organize and distribute the resulting massive data, we designed an HDF5 file structure that is the basis to feed information derived from the raw data into robot control frameworks like ROS.

Keywords: 3D Mapping, Hyperspectral Imaging, Data Storage, Perception and Sensing, Information and Sensor Fusion

1. INTRODUCTION

Over recent years, the rapid development of laser scanning technology has allowed to generate highly precise 3D models of large scale environments. Via co-calibration of different kinds of cameras, additional modalities can be added to the measured distance and reflectivity values. Adding color information from RGB cameras is state-of-the-art in commercial laser scanners. Adding other information than color is seldom seen but desirable to ease typical problems like segmentation and classification. One example is the addition of infrared data to detect thermal leaks (Borrmann et al., 2016).

In contrast to these thermal cameras, which capture only a narrow part of the electromagnetic spectrum, hyperspectral cameras allow to capture intensity information over several hundreds of narrow wavelength intervals. The analysis of such hyperspectral distributions for classification is a well-known technique in geo sciences and remote sensing. However, freely available data from remote sensing has a poor spatial resolution, which makes this data more or less useless for robotic applications, where a high local resolution is necessary to safely operate a robot.

In this paper, we close this gap by presenting a freely available reference data set of high resolution 3D point clouds, where each 3D point carries a spectral intensity distribution over 150 spectral channels. With this work, we initiate a basis for identifying new methods for typical problems that are hard to solve by relying on RGB information alone. One obvious example is the classification of vegetation to distinguish between crops and weed in precision farming, where both classes of objects are more or less green in RGB images. Other possible applications include classification between materials in urban areas (concrete, asphalt, stones) to distinguish between roads and sidewalks or assessing building damages by analyzing the integrity of, e.g., walls. Our data set was collected in the Botanical Garden at the University of Osnabrück. The garden is located in a former stone quarry and therefore features a large number of different materials from the aforementioned application domains in a relatively small area and is, therefore, an excellent environment to generate a reference data set of hyperspectral point clouds.

In the remainder of this paper, we present our measurement systems and characteristics of the collected data as well as a proposal to organize and distribute the huge amount of data that arises when dealing with such high resolution spectral data. We present reference images in different spectral channels and a preliminary application in segmentation, where we computed the well-known NDVI index to recognize vegetation in the collected reference data. All collected data is stored in a HDF5 file structure that can be accessed with a number of open source libraries and software packages. The presented reference data set and our open source processing software "Las Vegas Surface Reconstruction Toolkit" are available on the project's website [1] .

⋆ DFKI in Osnabrück is generally supported by the state of Niedersachsen. The 3D scanner and hyperspectral camera used for acquiring the data set was financed using a grant by DFG. Thomas Wiemann is currently financed by a grant of BMBF in the project SoilAssist-2. Sebastian Pütz is financed by a doctoral grant by Friedrich-Ebert-Stiftung. All this sponsoring is gratefully acknowledged.

[1] www.las-vegas.uni-osnabrueck.de

2. RELATED WORK

Most work that describes the use of 3D hyperspectral point cloud data is based on image data collected by UAVs. Although the calibration of hyperspectral cameras against 3D point clouds generated from UAVs is technically feasible, the actual generation of 3D point clouds with hyperspectral annotations is seldom seen. In Nevalainen et al. (2017), photogrammetically generated point clouds were fused with hyperspectral images from an UAV for tree classification. Similar results were shown in Näsi et al. (2015). However, the reported point density in these studies is relatively low with about 500 points per square meters with significant noise in the range of several centimeters. Using terrestrial laser scanners, point density can be increased by several orders of magnitude, while reducing the noise level to millimeters.

Fusing point clouds from terrestrial scanners and color data from RGB camera images is state of the art in commercial systems. The most common solution to compute the extrinsic calibration is to use specific reference patterns with detectable feature points in both domains, e.g., a checkerboard (Zhang and Pless, 2004; Buckley et al., 2013). This idea was extended to thermal camera data in Borrmann et al. (2016). In Nieto et al. (2010), a calibrated RGB camera on top of a laser scanner is used to create an RGB-colored point cloud, which is registered to the hyperspectral image using SIFT features and a piecewise linear transformation. However, this approach requires an additional camera for calibration. In Buckley et al. (2013), a terrestrial laser scanner was calibrated against a laser scanner to generate hyperspectral point clouds. In contrast to our system, an external hyperspectral camera was used, presumably resulting in a large parallax error in near range. Furthermore, the approach required manually placed markers for registration. In our setup, we used a hyperspectral line camera that was mounted on top of the laser scanner and rotated with it during scanning. To register point cloud and camera data, we used a similar camera model as in Buckley et al. (2013), but implemented a GPU-based mutual information optimization method to allow for markerless and featureless ad-hoc calibration (Igelbrink et al., 2018).

In the last years, several different data formats have become popular that allow for efficient storage and fast access of large amounts of block-shaped data coupled together with attached metadata. Those formats include, but are not limited, to the ROOT (Antcheva et al., 2011) and HDF5- Formats (Folk et al., 1999), which are among the most widely supported and mature. For this data set, we decided to use HDF5 for storing our multi-modal data (laser scanner, RGB camera, hyperspectral camera, pose information, and more) in a single file per data set.

3. GENERATING AND STORING HYPERSPRECTRAL POINT CLOUDS

In this section we present the mobile mapping system that was used for generating hyperspectral point clouds, as well as the HDF5 file structure for storing the data from the different sensors.

Fig. 1. The Pluto robot with Riegl VZ400i laser scanner and Resonon Pika L hyperspectral camera in the Botanical Garden at the University of Osnabrück.

3.1 System Setup

We installed our hyperspectral scanning system on the mobile robot *Pluto* that is based on the VolksBot XT platform. It features a *Riegl VZ400i* high resolution 3D laser scanner and a *Resonon Pika L* hyperspectral line camera, mounted on top of the rotating laser scanner. The camera records up to 297 independent spectral channels between 400 nm 1000 nm, which are accumulated to a hyperspectral panorama image, while the scanner is rotating and recording the point cloud data. In addition, we installed a regular RGB camera on the scanner system to capture high resolution images. For precise pose estimation, the robot is equipped with 3 IMU units and a differential GPS sensor. The robot with laser scanner and hyperspectral camera is shown in Fig. 1.

3.2 Data Acquisition

The data was collected in a stop-and-go fashion: The robot was manually steered to a desired scanning position. Here, we took a 360° laser scan and simultaneously recorded the hyperspectral data. After each scan, we switched the Resonon camera with a Nikon D610 camera to capture RGB images. This manual switching had to be done due to spatial restrictions. Since switching the sensors resulted in small but noticeable changes in the extrinsic orientation of the sensors, we had to re-calibrate the system every time we switched the cameras. Since our hyperspectral calibration procedure (Igelbrink et al., 2018) requires no external features and the calibration of the RGB camera can be easily re-adjusted after scanning using the software provided by the scanner manufacturer, the switching did not significantly affect the quality of the final data set.

The data collected by the sensors were stored into dedicated directories of the on-board computer. This kind of data organization has several drawbacks. Fist, this implementation is generating many files. For hyperspectral registration, we need to fuse the single lines of the Resonon camera into a consistent panorama image. Since the reference driver does not provide time-stamped data and does not deliver data at a fixed frame rate, we need to save all received frames and fuse them according to the average rotation speed of the laser scanner to generate consistent panorama images for re-projection of the points. Currently, we save each frame into a time-stamped file. In the generation process, frames are duplicated or dropped

depending on the rotation speed of the laser scanner, which we currently do offline after a scan was collected. Similarly, we have to store all RGB images and pose information.

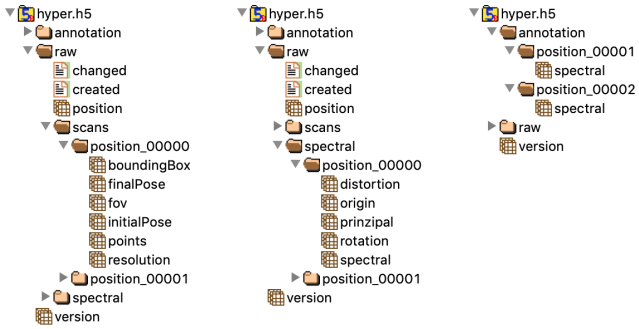
Our initial idea was to use ROS bags as a container for all data, but practically it showed that the hyperspectral data alone blew the 1GB default limit of ROS bags. Increasing this size resulted in other side effects like missing frames that lead us to the current solution. Distributing the data in a file system structure is generally not desirable. Compressing the data in a Zip file would solve the problem of scattered data, but uncompressing only parts is ineffective. From our point of view, HDF5 files overcome these drawbacks. Hence, we have implemented an HDF5 structure that fits the requirements for our data set. We implemented an interface that is compatible with common exchange formats and puts the data into the proposed structured within the HDF5 file. This structure is presented in the next section.

3.3 Data Storage and Structure

The HDF5 file format has several advantages over using directory structures. An HDF5 file has an internal structure similar to a file system consisting of groups (directories) and data sets (files), allowing for flexible hierarchical storage of data. Additionally, the format supports attaching metadata to each object. This makes it easy to store data of different modalities in a self-explaining format, as required for our data sets. More so, it allows for keeping related data closely together, e.g., the metadata related to a data set is attached directly to it, rather than being kept in a separate file. The HDF5 format is very flexible regarding data storage, access and memory layout. As an alternative to storing a data set as a single continuous block of memory, a chunked layout, where a data set is split into several, equally sized, chunks, which are stored separately in the file, is available. This significantly improves i/o performance while working on subsets of the data, because not all data has to be loaded into memory. Storing the data in a chunked layout allows for additional compression of a data set with multiple available algorithms. The different memory layouts and the compression can be freely mixed within one file, which allows for great flexibility.

The reference implementation of HDF5 is developed by the HDF5-Foundation in the C programming language. Bindings into nearly all popular programming languages are available. This makes it easy to switch between languages in a clean manner and using, e.g., Python for exploration and analysis, and C++ for fast processing of the same files without having to rely on multiple different libraries to parse and load a complex directory structure.

Data Organization in HDF5 Fig. 2 presents the general structure to organize our spectral and point cloud data. All collected sensor data is organized in a collection called `raw`. It contains meta information of where and when the data was acquired. The data collected by the individual sensors of the system is stored separately for each scanning position in sub-collections referring to the different sensors. For laser scan data, we store the initial pose estimation, the final registration in the global reference frame, field of view and angular resolution, the axis aligned bounding



(a) Scan Data (b) Spectral Data (c) Annotations

Fig. 2. Proposed organization of captured data in HDF5.

box as well as the single points in a sub-collection called `scans`.

The corresponding spectral data is stored in the `spectral` sub-collection. It contains the raw spectral panorama images and the calibration parameters for the cylindrical calibration model. The spectral data is organized in an image cuboid where each layer refers to an intensity image of a spectral channel. Information about the mapping of layers to spectral channels in encoded in the meta attributes of this `spectral` data set. Similarly, we store the acquired RGB images in a sub-collection `images` for each position.

Besides the raw data that comes from the system, we store derived data from the original data in an additional group called `annotations`. Here, we store the annotated point clouds with hyperspectral data directly as an array of consisting of 150 unsigned characters per point as well as RGB data in a separate array.

In addition to annotated point clouds, the user can store all other representations in specific groups and data sets that can be derived from the raw data. This can also include spatial indices like octrees and kd-trees that represent the stored point clouds. Our intention here is to allow the user to keep all information that was derived from the original data closely together with the original data. The representations stored in these additional sections are not limited to structures related to point cloud data. It may also include meshes with textures computed from the raw data like the ones presented in Wiemann et al. (2018).

In future work, we intend to store there map representations that can be used in robotic frameworks. Current ideas are Octomaps (Hornung et al., 2013), GridMaps (Kohlbrecher et al., 2011) or polygonal navigation maps (Pütz et al., 2016). The idea is to use the HDF5 data format as the basic storage for all static data about an environment and the can be re-used for different applications. The aim is to reduce the scattering of processed data from the original raw data that usually happens over time, when a data set is used for different experiments and application scenarios. We see this kind of organization within the HDF5 files as a persistence layer on file servers or other long term data storage. To make the data accessible in robotic contexts, an API layer and interface nodes to the used robot control architecture have to be implemented.

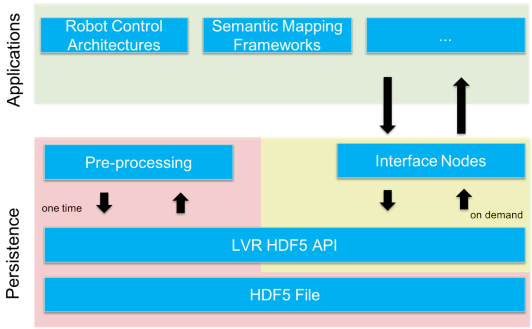


Fig. 3. Components to make high resolution data available for robotic applications.

The desired interaction between static storage and robot control architectures is sketched in Fig. 3. We mainly distinguish between a persistence layer, where all high resolution data is stored on a dedicated file storage with high bandwidth and high capacity. The raw and derived data are stored in the HDF5 file. The computations to generate the derived data are done by pre-processing modules that add new representations to the HDF5 file base. This includes map generation, scan registration, computation of the annotated point clouds, filtering, computation of indices, and generally all computations that are done on static environment data.

To make the computed information available, interface nodes need to be implemented that access the file base and generate representations that can be used in different applications. For example, stored grid maps from the HDF5 files could be converted to ROS messages and then published via an existing ROS master. Another possible application is feeding of static information into semantic mapping applications like our SEMAP framework (Deeken et al., 2018) for qualitative spatial reasoning about the environment. The interface nodes could also be used to send relevant data *on-demand*, since a robot usually does not need to keep all information about an environment in working memory, but only the information that is needed the current task, e.g., the navigation map of a certain area.

Currently, we have implemented interface nodes to ROS that allow to query polygonal meshes and cost maps from such an HDF5 file that are used for local navigation. This so called *Mesh Map Server* is able to perform several range queries like and deliver it in form of dedicated ROS messages. For this kind of interaction between persistence layer and application it is important that the relevant data can be extracted without high latencies from the data storage. Hence, we have evaluated to performance of HDF5 files for our application.

4. THE BOTANICAL GARDEN DATA SET

In this section we describe the main features of the generated reference data set. It consists of 16 hyperspectral laser scans and covers an area of approximates 70×75,m. An aerial photo of the scanned area and the rough layout of the scan positions and covered area is shown in Fig. 4. Each 3D laser scan was taken with a field of view of $360 \times 100°$ with an horizontal and vertical angular resolution of $0.05°$, resulting in approximately 70 million points per scan. For each scan position, we collected the full spectral data in

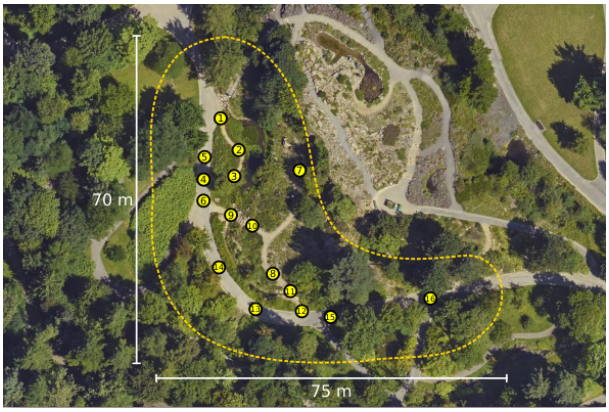


Fig. 4. Aerial view of a part of the Botanical Garden at the University of Osnabrück with an indication of the scanned area (Image provided by the City of Osnabrück, geo.osnabrueck.de).

150 buckets between 400 nm and 1000 nm. Additionally, we took 5 24 megapixel RGB images per scan for RGB annotation. The total amount of collected raw data sums up to 45 GB for this relatively small area. The single scans were automatically registered based on the GPS pose estimations provided by the Riegl laser scanner and the robots odometry using slam6d [2].

Fig. 5 shows exemplary 3D views on the data set with different modalities. The top row shows renderings of the same scenes with annotated spectral intensities at wavelengths of 400, 600 and 800 nm respectively. For rendering, the measured intensities were normalized and mapped to an blue to red color gradient, visualizing the different intensity distributions at different wavelengths. The pictures in the bottom row show overview renderings of the whole data set with RGB annotation (left) and mapped NDVI values. Human-made an-organic structures like pathways are clearly distinguishable in this representation. We added this picture to demonstrate the potential of the combination of hyperspectral and spatial information for robotic applications. Especially for semantic classification the use of such sensor combinations seem to be extremely beneficial, especially in combination with methods from remote sensing, where classification of hyperspectral images is an established discipline.

Note, that the spectral intensities of the different scans have not been normalized yet. This task would require a reference spectrum. Due to the rotating camera in our setup, acquiring such a spectrum is difficult, since it is only valid for small variations of the camera's field of view. We plan to resolve this issue in future work.

5. HDF STORAGE PERFORMANCE

So far, we presented the organization and features of the collected data. In this section we analyze the benefits of using HDF5 files for such data in terms of data compression and access speed. All experiments were performed on an Intel Core i7-4930K processor with 6 physical cores and 32 GB RAM. For accessing HDF5 files, we used libhdf5 in version 1.10.

[2] www.slam6d.sourceforge.net

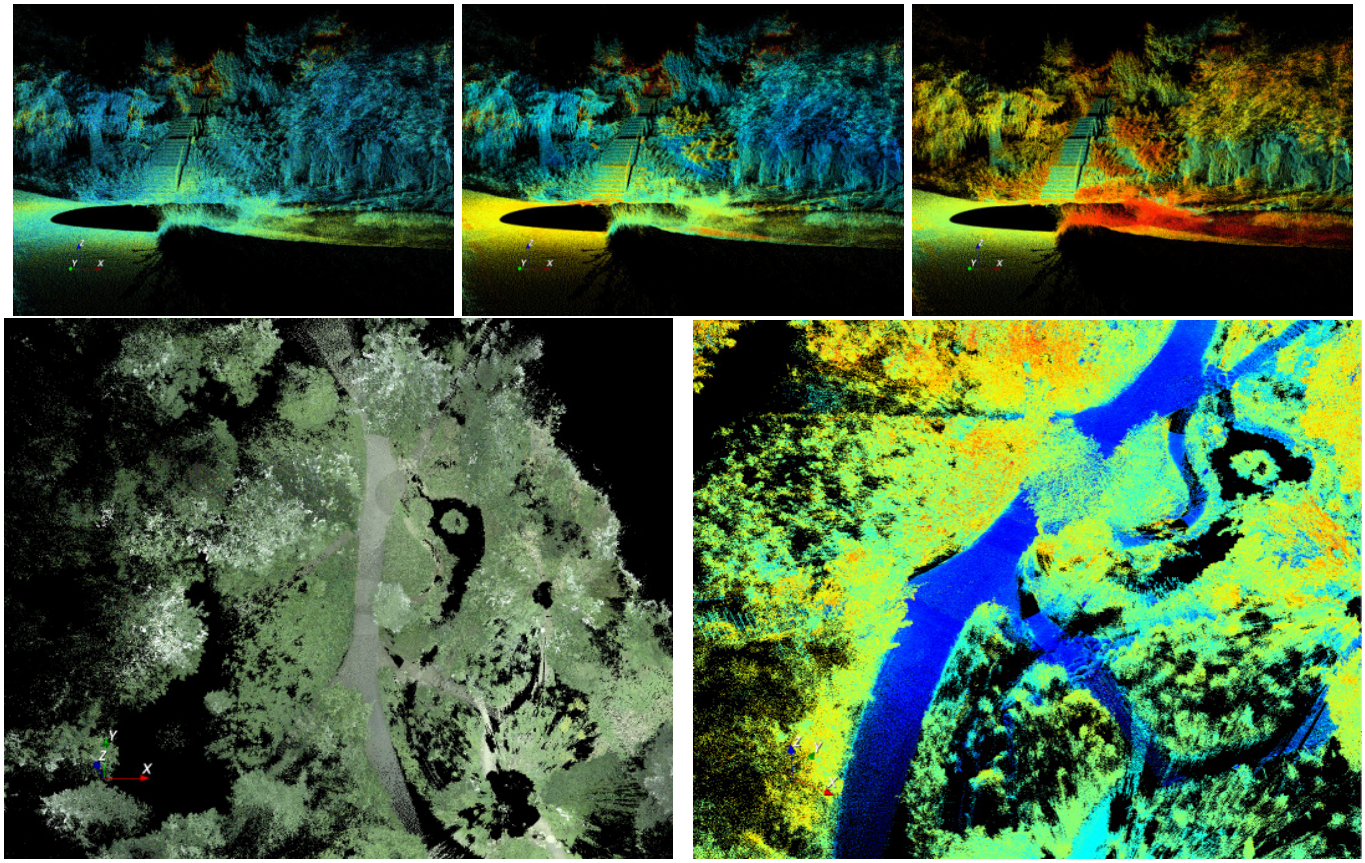


Fig. 5. Visualizations of the Botanical Garden data set. The top row shows renderings of the spectral intensities at 400 mn (left), 600 nm (middle) and 800 nm (right). The lower row shows an overview of the whole data set with RGB annotations (left) and normalized NDVI values (right); the gravel pathways clearly stick out in the NDVI representation.

Table 1. Maximum compression rates and HDF5 file generation times for different storage modes.

Method	File size	Generation Time
No compression	39.1 GB	27:26 min
1 channel image aligned	21.3 GB	47:26 min
5 channel image aligned	21.3 GB	47:30 min
Chunked 50	21.0 GB	49:56 min

In a first experiment, we evaluated the resulting file sizes using different compression methods available in libhdf5. The maximum achievable compression rates and average file creation times for different compression methods are shown in Tab. 1.

The first row shows the stats without compression enabled. With compression enabled, the HDF5 library splits the data into chunks which are organized in a binary search tree. The size of the chunks has an significant impact on the achievable compression ratios and influences the file generation time. When experimenting with the data, we found that the chunking of the float arrays of the point cloud data did initially not effect the file size noticeably, probably due to the encoding of floating point values. However, the chunking of the hyperspectral data proved to deliver higher compression rates. Therefore, we evaluated different chunk sizes as displayed in the remainder of Tab. 1.

First, we stored one chunk per hyperspectral channel in the panorama images ("1 channel image aligned") as well as chunks consisting of 5 adjacent hyperspectral channels ("5 channel image aligned"). This reduced the initial file size, but showed no measurable differences between 1 or 5 channeled data access. However, creating chunks of 50 elements over all dimensions in the image cuboid ("Chunked 50") reduced the file size further at cost of higher generation time. To find out how this kind of chunking effected the files, we varied the chunks sizes from 10 to 1000 as displayed in Fig. 6. Here, a minimum at the tested chunk size of 50 is clearly visible with the default compression enabled.

In addition, we activated the data shuffle algorithm implemented in the HDF5 library. Here, the file size initially increased. To test our hypotheses that the internal floating point representation prevents further compression, we converted the scan data to integer values by converting from meter values to rounded units of 0.1 millimeters, which is well below the distance accuracy of the laser scanner. In this integer representation, the compression ratio increased, but combining the integer representation with shuffling in the end lead to very high compression rates. In this representation, we were able to achieve compression rates of about 64% compared to the initial raw data.

To evaluate the time to access our data stored in HDF5 files we benchmarked the time needed to load a single

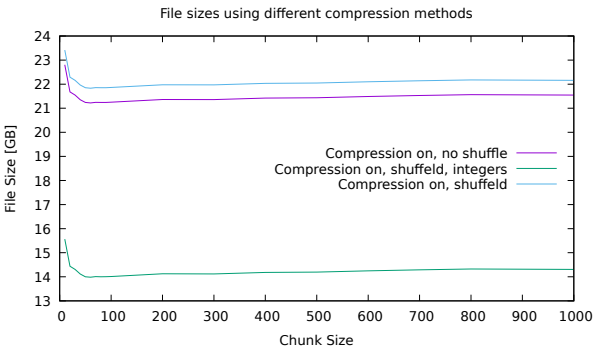


Fig. 6. HDF5 file sizes of the Botanical Garden data set with different compression methods and chunk sizes.

Table 2. Comparison of creation and access times for a single scan position.

Representation	Creation Time	Access Time	Size
Directory	n/a	52 s	2.1 GB
gzip	4:33 min	68 s	1.5 GB
HDF5	4:41 min	32 s	1.6 GB

scan position from a HDF5 file and compared it to direct loading of raw data from a directory and a gzip-compressed directory. For the gzip comparison, we extracted the compressed data to a separate directory and loaded the extracted raw data. The results of these experiments are shown in Tab. 2. As expected, the access time for gziped data is significantly slower than direct data access. Surprisingly, the HDF5 access is about 45% faster than direct reading of the individual files. The high HDF5 reading performance can be explained with the internal chunking which is optimized for fast data access. In directory access we store the singe hyperspectral channel images in PNG files, which may produce reading overhead. Generally, this evaluation proves that the use of HDF5 files is a good choice for permanent storage and fast access. The most significant drawback here is the comparatively long time that is needed to generate the HDF5 files from the initial directory representation of the raw data.

6. CONCLUSION

In this paper we presented a reference data set consisting of 16 hyperspectral laser scans. In addition, we presented a HDF5 based organization of the acquired information that allows for compact storage and fast access to recorded data, as shown in the evaluation. Besides these features, the use of HDF5 allows to index the data and make it self-consistent. Currently, we are creating the files from a intermediate directory structure on a hard drive. In future work, we plan to implement a solutions that writes the acquired data directly into HDF5 files. The challenge here will be to deal with the relatively high latencies when creating HDF5 files. Furthermore, we plan to formalize the storage of map representations and other relevant data in HDF5 files, for data distribution scenarios as indicated in Fig. 3.

REFERENCES

Antcheva, I., Ballintijn, M., Bellenot, B., Biskup, M., Brun, R., Buncic, N., Canal, P., Casadei, D., Couet, O., Fine, V., et al. (2011). Root—a c++ framework for petabyte data storage, statistical analysis and visualization. *Computer Physics Communications*, 182(6), 1384–1385.

Borrmann, D., Leutert, F., Schilling, K., and Nüchter, A. (2016). Spatial projection of thermal data for visual inspection. In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 1–6. IEEE.

Buckley, S.J., Kurz, T.H., Howell, J.A., and Schneider, D. (2013). Terrestrial lidar and hyperspectral data fusion products for geological outcrop analysis. *Computers & Geosciences*, 54, 249–258.

Deeken, H., Wiemann, T., and Hertzberg, J. (2018). Grounding semantic maps in spatial databases. *Robotics and Autonomous Systems*, 105, 146–165.

Folk, M., Cheng, A., and Yates, K. (1999). Hdf5: A file format and i/o library for high performance computing applications. In *Proceedings of supercomputing*, volume 99, 5–33.

Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*.

Igelbrink, F., Wiemann, T., Pütz, S., and Hertzberg, J. (2018). Markerless ad-hoc calibration of a hyperspectral camera and a 3d laser scanner. In *Proc. 15th International Conference on Intelligent Autonomous Systems (IAS-15)*. Springer.

Kohlbrecher, S., von Stryk, O., Meyer, J., and Klingauf, U. (2011). A flexible and scalable slam system with full 3d motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, 155–160.

Näsi, R., Honkavaara, E., Lyytikäinen-Saarenmaa, P., Blomqvist, M., Litkey, P., Hakala, T., Viljanen, N., Kantola, T., Tanhuanpää, T., and Holopainen, M. (2015). Using uav-based photogrammetry and hyperspectral imaging for mapping bark beetle damage at tree-level. *Remote Sensing*, 7(11), 15467–15493.

Nevalainen, O., Honkavaara, E., Tuominen, S., Viljanen, N., Hakala, T., Yu, X., Hyyppä, J., Saari, H., Pölönen, I., Imai, N.N., and Tommaselli, A.M.G. (2017). Individual tree detection and classification with uav-based photogrammetric point clouds and hyperspectral imaging. *Remote Sensing*, 9(3).

Nieto, J., Monteiro, S., and Viejo, D. (2010). 3D geological modelling using laser and hyperspectral data. In *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*, 4568–4571. IEEE.

Pütz, S., Wiemann, T., Sprickerhof, J., and Hertzberg, J. (2016). 3d navigation mesh generation for path planning in uneven terrain. *IFAC-PapersOnLine*, 49(15), 212 – 217. 9th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2016.

Wiemann, T., Mitschke, I., Mock, A., and Hertzberg, J. (2018). Surface reconstruction from arbitrarily large point clouds. In *Robotic Computing (IRC), 2018 Second IEEE International Conference on*, 278–281. IEEE.

Zhang, Q. and Pless, R. (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *Proc. Intelligent Robots and Systems, (IROS 2004)*, 2301–2306. IEEE.