



HORIZON EUROPE

Digital and emerging technologies for competitiveness and fit for the Green Deal

HYPERIMAGE

**A universal spectral imaging sensor platform for industry,
agriculture, and autonomous driving.**

Starting date of the project: 01/12/2023

Duration: 42 months

= Deliverable D4.1 =

Reference data formats and structure description

Due date of deliverable: 30/06/2025

Actual submission date: 02/07/2025

Responsible WP4: Netcompany Intrasoft

Responsible TL: Jonas Golde, Fraunhofer IWS

Version: V1.0

Dissemination level		
PU	Public	x
SE	SENSITIVE – limited under the conditions of the Grant Agreement	



AUTHOR

Author	Institution	Contact (e-mail, phone)
Jonas Golde	FhG IWS	jonas.golde@iws.fraunhofer.de
Stephan Becker	FhG IWS	stephan.becker@iws.fraunhofer.de

DOCUMENT CONTROL

Document version	Date	Change
V0.1	20/05/2025	Initial setup
V0.2	03/06/2025	Added Specification
V0.3	18/06/2025	Removed unnecessary paragraphs
V1.0	30/06/2025	Finalization of all sections

VALIDATION

Reviewers		Validation date
Work Package Leader	Netcompany-Intrasoft (INTRA)	01/07/2025
Project Manager	Marina de Souza Faria	02/07/2025
Project Coordinator	Alexander Kabardiadi-Virkovski	02/07/2025

DOCUMENT DATA

Keywords	HSI, spectral image, data format
Point of Contact	Name: Jonas Golde Partner: IWS Address: Winterbergstraße 28, 01277 Dresden, Germany Phone: +49 351 83391 3886 E-mail: jonas.golde@iws.fraunhofer.de
Delivery date	02/07/2025

DISCLAIMER

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Health and Digital Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Executive Summary

Hyperspectral imaging (HSI) is moving from laboratory studies to networked industrial and agricultural workflows, but no single data standard currently satisfies both legacy desktop tools and cloud-native processing. We introduce an open container specification that extends the widely adopted ENVI format while retaining full human readability. The proposed format introduces an enriched ENVI header that functions as a centralized metadata hub, referencing auxiliary files essential for comprehensive data interpretation and visualization. These include calibration and processing directives, environmental and camera-specific metadata and machine learning (ML) ground truth annotations for semantic segmentation and classification. The format is encapsulated in an uncompressed ZIP container to ensure robust and transparent data exchange across diverse applications and platforms.

This specification supports multiple metadata types, including binary masks for semantic segmentation, CSV files for bounding boxes, and direct header integration for classification labels. While the spectral data structure remains largely ENVI-compliant, optional support for advanced compression methods such as JPEG-XL and a forthcoming PCA-based algorithm is included. The format is designed to accommodate a wide range of use cases, including the fields within the HyperImage project which is the prediction of void formation in soldering processes for high-energy semiconductor modules; phenotypic trait estimation and stress detection in vertical farming, the use in UAV / drone systems for remote surveillance as well autonomous driving in off-road scenarios. Furthermore, integration in a cloud-based spectral data processing infrastructure is envisioned and prepared.

Designed with extensibility in mind, the format offers a flexible foundation that surpasses the limitations of the traditional ENVI standard, enabling seamless integration of diverse data types and supporting a broad spectrum of current and future hyperspectral imaging applications.

Table of Contents

1. Introduction	5
2. Data Format Specification.....	6
2.1. Mandatory and 'optional' information	6
2.2. Container Structure.....	7
2.3. ENVI Header Specification.....	8
2.4. Mandatory fields following the original ENVI standard.....	8
2.5. Mandatory fields of the proposed HSI format	9
2.6. Optional fields for visualization.....	9
2.7. Optional fields for searching and grouping	10
2.8. Further optional metadata and configuration fields	11
2.9. Optional classification and labeling fields	12
3. Discussion	13
3.1. Harmonization with the new IEEE 4001 standard	13
4. Conclusions	14
5. Dissemination level	14

1. Introduction

The deliverable “Reference data formats and structure description” is part of the work package 4 (Digital Platform Data Architecture) and corresponds to the task 4.1 “Definition of reference data formats and overall data structure”. Therefore, it specifies the spectral data format, that has been developed within the task and which will be used for storage and subsequent machine-learning/visualisation in the future.

The main motivation for specifying a new data format is the large volume of hyperspectral images, which typically presents a significant challenge in multiple application scenarios. Figure 1 shows the starting point of the considerations based on the previous data handling in the software developed and used by Fraunhofer IWS and DIVE. In the former approach (a) all data is processed directly after the acquisitions and therefore needs to be stored in floating point format, which increases the amount of stored data substantially, e.g. by a factor of almost 2 when assuming 4 byte per pixel floats instead of 2 byte / 16 bit integers of the raw data. A common strategy to address this issue would be the use of compression techniques, which can substantially reduce data volume and thereby enable the storage and transmission of large hyperspectral datasets. However, already storing the raw data (b) with a subsequent and computationally efficient processing, which includes the spectral background and envelope correction, can reduce the amount of stored data.

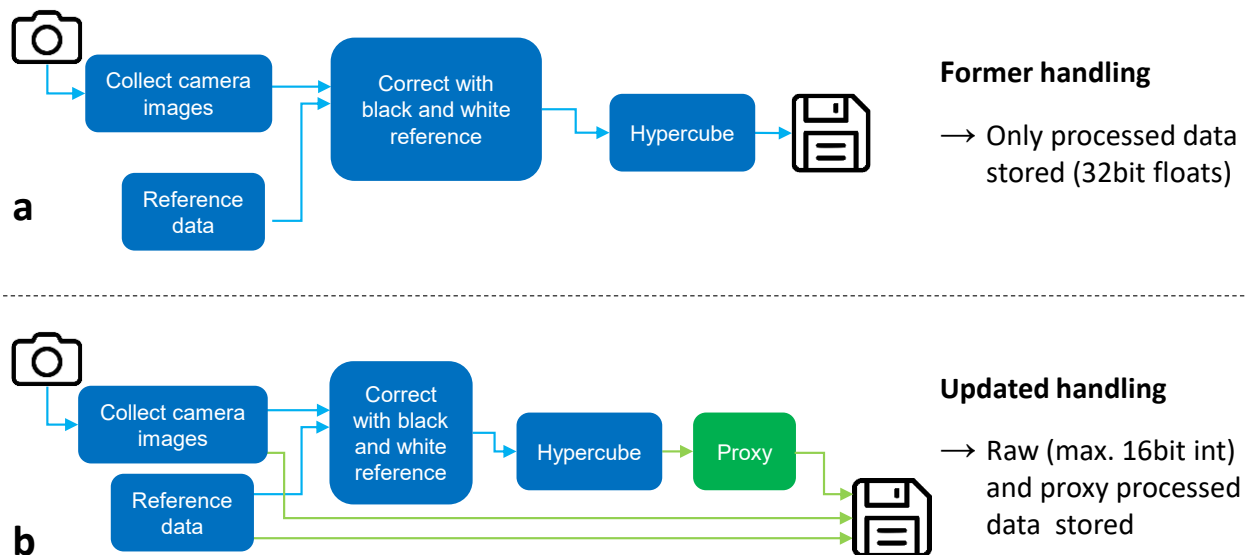


Figure 1: Previous (a) and new (b) envisioned processing scheme for the hyperspectral imaging solutions, both in stand-alone / desktop applications and in the cloud platform.

To enable this approach, a new data format based on the ENVI standard and a container concept is introduced within this work. The described solution includes the possibility to store ground-truth data in various formats for each use case including an extendable metadata standard, and is suitable to be combined with different methods for efficient handling of large spectral volumes, e.g. based on compression. Due to the high flexibility of the container design, future concepts for heterogeneous spectral data integration beyond the scope of this project can be envisioned.

2. Data Format Specification

The need for a robust sharing standard between partners of the HyperImage project and later usage encouraged a new specification of a HSI container format. It can also serve as simple archive format, where no database is available. As a data format mainly used for research and development, the simple container solution with its easy access to all metadata is preferred over similar hierarchical standards like HDF5 that require additional software. The easily human readable ENVI header serves as the main information hub and enables a standardized and thorough understanding of the HSI container file without additional software. Compared to a similar ENVI standard folder structure that is for instance used also commercially by SPECIM in their HSI cameras, the new format tries to specify the minimal common set over all HyperImage use cases while still enabling the storage of any additional information that will be needed for the specialized cases.

Figure 2 shows an overview sketch of the container including all relevant information about a certain measurement, the corresponding parameters, conditions and additional ground truth. While its contents are described in more detail in the following, the idea of the container is a high flexibility and robustness at the same time. By using the uncompressed ZIP format, data can be surveyed and modified easily by an experienced user as most operating systems natively provide direct memory access to this container format. Storing all data in a folder would provide the same flexibility, however, the chance for unintended modifications during data transfer and exchange is higher in this case.

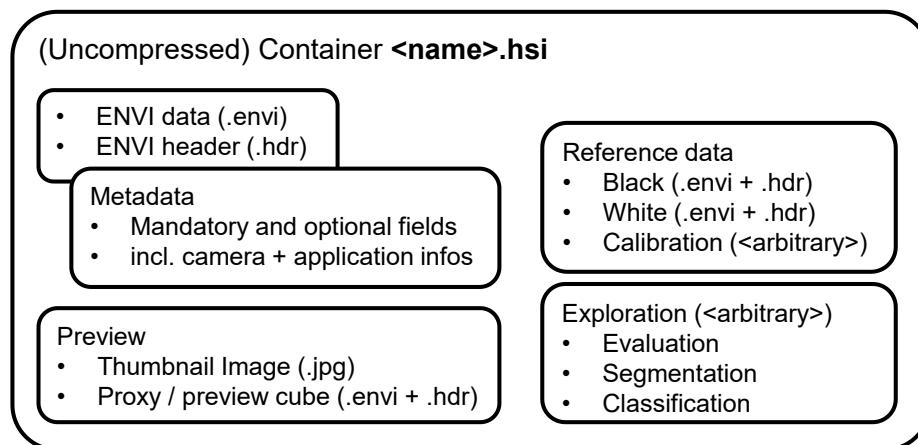


Figure 2: Sketch of the data format concept.














2.1. Mandatory and 'optional' information

The specified container file must contain certain information to allow importing software solutions to reconstruct a meaningful data representation. While reading the container file, optional information and metadata cannot be expected. However, if optional information is missing it may be interpreted as not available at all.

On the other hand, software that writes this format must include all information for a meaningful data visualization, but at minimum the mandatory fields. **If the optional information of the specification is available at all, it must be included.** This leads to only a small overhead compared to the typically huge HSI data by duplicating information for every container file that may be stored only once for multiple measurements. The main advantage is that the container file gains completeness and becomes independent of additional information, thus enabling easy sharing and archiving properties, which is the main purpose of this container file format.

2.2. Container Structure

The container consists of several files that have a fixed file name. Arbitrary name components are marked with <...>. Mandatory files are additionally marked with bold names.

	Name	Description	Explanation
	<name>.hsi	HSI Container	This is the root container in ZIP format. Currently it is defined as an uncompressed zip container to bundle all information and still allow direct memory access.
	data.envi	ENVI data (Hypercube)	Actual measured data. It represents images of different spectral 'bands'. The image consists of horizontal 'lines' and each line is comprised by 'samples'. The format is specified in the .hdr file.
	data.hdr	ENVI header (metadata hub)	This serves as the central information hub and stores many non-standard fields. Please see section 2.3 for a detailed header specification.
	thumbnail.jpg	Thumbnail of data	This serves as a preview image, showing the data with an arbitrary resolution and bands to color transfer function.
	preview.envi	Preview ENVI data	This is an optional preview version of the ENVI data. This is especially useful if the original file is too big to load and extractions are unwanted or computationally expensive. Can be used to generate a preview visualization of the data. More proxy files with higher levels of details (LODs) may be present. The naming convention of the larger proxy data files is <ul style="list-style-type: none"> • preview_<LOD identifier>.envi
	preview.hdr	Preview ENVI header	Header file of the preview ENVI data. This should outline the minimal information for calculating a meaningful preview independently of the main data header file. Please see sections Error! Reference source not found. and 2.5 for detailed header specifications. More proxy files with higher levels of details (LODs) may be present. The naming convention of the larger proxy header files is <ul style="list-style-type: none"> • preview_<LOD identifier>.hdr
	Reference	Reference data subdirectory	Subdirectory under the root container bundling the reference files. These are normally used to calibrate the raw sensor data under different measurement conditions (sensor, lighting, aperture, exposure, etc.). All files in this directory need to be available for the custom micro-services specified in the <i>read procedures</i> (see section 2.5).
	white.envi	White reference ENVI data	Calibration data for the white reference, representing the maximum raw intensities measured by each sensor pixel. It at least contains multiple <i>samples</i> and <i>bands</i> . Additionally it either consists of a single (possibly aggregated) <i>line</i> for push broom setups or multiple lines for multi-spectral snap shot setups.
	white.hdr	White reference ENVI header	Calibration header for the white reference. This needs to specify at least the mandatory fields (see section 2.3).
	black.envi	Black reference ENVI data	Calibration data for the black reference, representing the minimum raw intensities measured by each sensor pixel. Otherwise similar to the white reference ENVI data.
	black.hdr	Black reference ENVI header	Calibration header for the black reference. This needs to specify at least the mandatory fields (see section 2.3).
	<any format>	QE-, WL-Calibration, etc.	Further calibration files containing sensor response information (quantum efficiency (QE), wavelength calibrations, etc.). These may be needed to compute comparable units from the raw data.
	Exploration	Data exploration subdirectory	Subdirectory under the root container bundling the exploration related files (e. g. results for evaluation, segmentation, classification). Further specifications may follow in the future.

2.3. ENVI Header Specification

The header specification and the following header field tables mainly follows the specification of [NV5 Geospatial Solutions, Inc.](#). The header follows some simple rules to ensure human and machine readability:

- The envy header file starts with a line containing only the string `ENVI`.
- The metadata then is stored in the following line separated entries of field-value pairs. Field and value are separated by space-equal-space `=`. The field names are read case-insensitive and written in lower case only. Special characters for the values are braces and commas.
- Braces `{ }` encapsulate a value list and may span over multiple lines. Values inside a value list are separated by commas. Whitespace after opening and before closing braces is ignored.
- Commas `,` inside the braces are used to separate the different values of a value list. Inside braces, any additional whitespace before and after commas is ignored. While writing, a comma should be followed by a space to increase readability. Commas outside of braces are interpreted as part of the value string.
- Comments may be added by inserting a new line with a semicolon `;` as the first non-whitespace character, but must not appear inside braces of a value list.

Fields, values and their description are listed in the following sections. Entries that are not in the original ENVI standard are marked in **green**. Square brackets and asterisk in the value syntax specification are only meant substitutional: Square brackets `[]` surround optional parts of a value and asterisks `*` indicate optional repeating syntax.

2.4. Mandatory fields following the original ENVI standard

All contained ENVI header files need to at least specify the fields *bands*, *lines*, *samples*, *byte order*, *data type*, *file type* and *interleave*.

Field name	Value syntax	Description
bands	Integer	The number of (spectral) <i>bands</i> per image file.
lines	Integer	The number of <i>lines</i> per image for each band.
samples	Integer	The number of <i>samples</i> (pixels) per image line for each band.
byte order	0 or 1	The order of the bytes in integer, long integer, 64-bit integer, unsigned 64-bit integer, floating point, double precision, and complex data types. Use one of the following: <ul style="list-style-type: none"> • Byte order=0 (Host (Intel) in the Header Info dialog) is least significant byte first (LSF) data (DEC and MS-DOS systems). • Byte order=1 (Network (IEEE) in the Header Info dialog) is most significant byte first (MSF) data (all other platforms).
data type	1 to 15	The type of data representation: <ul style="list-style-type: none"> • 1 = Byte: 8-bit unsigned integer • 2 = Integer: 16-bit signed integer • 3 = Long: 32-bit signed integer • 4 = Floating-point: 32-bit single-precision • 5 = Double-precision: 64-bit double-precision floating-point • 6 = Complex: Real-imaginary pair of single-precision floating-point • 9 = Double-precision complex: Real-imaginary pair of double precision floating-point • 12 = Unsigned integer: 16-bit • 13 = Unsigned long integer: 32-bit • 14 = 64-bit long integer (signed) • 15 = 64-bit unsigned long integer (unsigned)

HyperImage

file type	ENVI Standard or ENVI Classification	The ENVI-defined file type, such as a certain data format and processing result. ENVI files in the root, including proxy files, must set the file type to <i>ENVI Standard</i> . Other file types may be supported in the future (see ENVI FILE TYPE for more definitions). Future supported values may include compression formats: <ul style="list-style-type: none"> • PCA • JPEGXL • JPEG2000 Additional ENVI files in the Exploration directory may be set to <i>ENVI Classification</i> , <i>ENVI FFT</i> , PCA or other file types mentioned in the original definition.
interleave	BSQ, BIL or BIP	Refers to whether the data interleave is Band Sequential (BSQ), Band-interleaved-by-Line (BIL) or Band-interleaved-by-Pixel (BIP). See ENVI Image Files for detailed explanation.
header offset	Integer (normally 0)	The number of bytes of embedded header information present in the file. ENVI skips these bytes when reading the file. The default value is 0 bytes.

2.5. Mandatory fields of the proposed HSI format

Field name	Value syntax	Description
wavelength	{Float[, Float]*}	Lists the center wavelength values of each band in an image. Units should be the same as those used for the <i>fwhm</i> field and set in the <i>wavelength units</i> value.

2.6. Optional fields for visualization

Optional fields must be written when the data is available. Absence of the fields can be interpreted as absence of the corresponding information.

Field name	Value syntax	Description
band names	{String[, String]*}	Allows entry of specific names for each band of an image. Number of values should match the <i>bands</i> field.
default bands	{Integer, Integer, Integer}	Indicates which band numbers to automatically load greyscale or RGB fields every time the file is opened. By default, a new image is automatically loaded when a file that has default bands defined in its header is opened. If only one (repeating) band number is used, then ENVI loads a greyscale image.
pixel size	{Float, Float}	Indicates x and y pixel size in meters for non-georeferenced files.
reflectance scale factor	Float	The value that, when divided into your data, would scale it from 0-1 reflectance. For example, if the value of 10,000 in your data represents a reflectance value of 1.0, enter a reflectance scale factor of 10,000.
data offset values	Float	Offset values for each band.
description	{String}	A string describing the image or the performed processing.
read procedures	{String[, String]*}	This field is only used in ENVI Classic, but will be hijacked to include processing steps that should be reenacted in the given order on the given data before visualization. This is not the place for a history of processing steps. Possible string values may include:

HyperImage

		<ul style="list-style-type: none"> • microservice:String starts a (python) micro-service routine with given name. This may be used to encapsulate all non-standard procedures, like e. g. special calibrations or preprocessing of certain camera devices. • resample:{Integer, Integer, Integer} resamples the data to given size in <i>sample</i>, <i>line</i> and <i>band</i> axis • crop:{Integer:Integer, Integer:Integer, Integer:Integer} for cropping to given (inclusive) index range in <i>sample</i>, <i>line</i> and <i>band</i> axis • rotate:String with String being either <i>clockwise</i>, <i>counterclockwise</i>, <i>180</i>, <i>swap-left-right</i>, <i>swap-up-down</i> • merge:{String:Integer:Integer[, String:Integer:Integer]*} with String as an ID (e.g. relative full path) for the other hypercube to merge with, Integer as an offset in x direction • other procedure definitions will be added in the future
--	--	--

2.7. Optional fields for searching and grouping

The following fields provide the information that will be used to search the database for reusability and future referencing. The same fields may be used to group different HSI-measurements into training and/or validation data sets for different machine learning models. IDs should be descriptive strings, since most software solutions will display them directly to the user.

Field name	Value syntax	Description
acquisition time	YYYY-MM-DD or YYYY-MM-DDTHH:MM:SS.DZ or YYYY-MM-DDTHH:MM:SS:Dooo:mm	<ul style="list-style-type: none"> • YYYY is the four-digit year • MM is the two-digit month • DD is the two-digit day • T separates the date and time • HH is the two-digit hour • MM is the two-digit minute • SS is the two-digit second • D is the decimal fraction of a second with up to double-precision • Z indicates that the time is in Coordinate Universal Time (UTC) • ooo is a two-digit offset in hours from UTC time. If the offset is negative, a minus symbol (-) precedes the value. • mm is an optional partial-hour offset (in minutes) from UTC time.
problem	{String[, String]*}	A problem keyword that describes the reason of the measurement. This serves as a tag collection for filtering and will also be used to find suitable data sets for a given machine learning problem. Examples may be layer-thickness, contamination, delamination, crop-disease or impassability.
series	String	Groups multiple measurement targets or scenes with the same problem into one (preferably human readable) series ID.

HyperImage

target	String	Groups multiple measurements of a target or scene into one (preferably human readable) ID. (different measuring devices, spectral ranges or other differing conditions)
target info	{String[, String]*}	This field can be used to store additional searchable information about the measurement target, like customer, manufacturer or scene location.
measurement	String	A preferably human readable ID of the single measurement that meaningfully distinguishes the measurement from the others. This is the smallest unit and identifies the current HSI container.
sensor type	String	Link to a camera file of the same name under the Metadata directory that describes the sensor or camera and its characteristics in more detail.
device	String	A preferably human readable ID of the whole HSI camera setup or other measuring device. (More specific device descriptions may be stored in arbitrary file format in the Metadata directory.)
optics	{String[, String]*}	This specifies the used optics (e. g. microscope, macro, pixel-shifted). This field may have multiple Strings as its value to allow for multiple tags at the same time.

2.8. Further optional metadata and configuration fields

Additional metadata fields, for detailed inspections and documentation. Especially all configuration parameters for the used HSI camera setup should be listed as an additional field to ensure complete documentation and reproducibility.

Field name	Value syntax	Description
fwhm	{Float[, Float]*}	Lists full-width-half-maximum (FWHM) values of each <i>band</i> in an image. Units should be the same as those used for <i>wavelength</i> and set in the <i>wavelength units</i> parameter.
temperature	Float	Temperature of the HSI camera in °C.
String	String or {String[, String]*}	Any metadata or configuration parameter that was not mentioned before or cannot fit into the previously specified fields should define their own fields using the general syntax. Typical additional values may include <ul style="list-style-type: none"> • start time = HH:MM:SS • stop time = HH:MM:SS • errors = {String[, String]*} • fps = Float • exposure = Float • aperture = Float • binning = {Integer, Integer[, Integer]} • hroi = {Integer, Integer} • vroi = {Integer, Integer} • humidity = Float • light condition = String • pressure = Float All of these unspecified fields should be readable in the visualization software.

HyperImage

		Any metadata that cannot be handled with a simple keyword or set of keywords may be stored in additional files under the Metadata directory.
--	--	--

2.9. Optional classification and labeling fields

This section is subject to change and will be expanded during the duration of the project. There are currently three types of classification problems supported:

- *Image classification*, which classifies whole images. This may be simply stored in the class field of the ENVI header.

Field name	Value syntax	Description
class	String	Class for the whole image. This is useful for image classification problems.

- *Semantic segmentation*, which stores pixel accurate segmentations for each class either in multiple binary files or a single ENVI classification file plus a corresponding minimal ENVI header in the Exploration directory.

Field name	Value syntax	Description
classification file	String.envi	File name of the ENVI classification file in the <i>Exploration</i> directory. Serves as an indicator of a single ENVI classification file that stores the semantic segmentation masks as bands.
classes	Integer	Defines the number of classes, including unclassified regions, for classification files.
class lookup	{Integer, Integer, Integer[, Integer, Integer]*}	Lists class colors using (uint8) RGB color definitions for classification files. For example, black is 0,0,0.
class names	{String[, String]*}	For classification files, this lists the names of all classes, including background or unclassified. For the main ENVI file, this links to multiple binary files with the same name. Each of them stores the semantic segmentation as a binary mask for the corresponding class. Note that each pixel may have more than one class, supporting more complex instance segmentation tasks. Class colors (either from the field class lookup or arbitrarily chosen) may be arbitrarily blended for visualization.

- *Bounding box based object detection*, marks instances of specified classes in the spectral image with a separate CSV file under the Exploration directory.

Field name	Value syntax	Description
bounding boxes	bounding boxes.csv	This links to a CSV file, containing the detected bounding boxes. Each row defines a single bounding box in the format <i>name, left, top, width, height</i> . The name depicts a possibly non-unique bounding box class name. Left and top are integer coordinates of the bounding boxes top left corner. Width and height specify the size of the bounding box in pixel.

3. Discussion

The specified data format structure is closely based on the well-established ENVI format, which ensures a high adaptability to existing solutions in hyperspectral or multispectral imaging. By additionally introducing the container concept, both the flexibility of a hierarchical structure, e.g. nested folders, and the robustness of a single file for data transfer and handling can be achieved.

An intrinsic data reduction in comparison with the existing DIVE / Fraunhofer software solutions (VESolve / imanto pro) is thereby achieved through storing up to 16 bit integer raw data instead of 32 bit processed floating point processed data as shown in Figure 1. While the latter implementation requires the storage of some additional information and especially all reference data, the advantage becomes more efficient the larger a single acquisition is. In addition, multiple scenarios are possible in the future to address certain needs in precision, e.g. a white reference before and after the actual measurement can be used to create a time-dependent interpolated reference for measurement conditions with a drift.

The amount of data could be further reduced for some camera models, which natively only record raw data with a 12 bit depth instead of 2 byte / 16 bit. While this is not compatible with the ENVI standard, sophisticated bit packaging concept might be useful. Another approach is the compression of the container, which would reduce the flexibility but also the amount of stored data. Several compression algorithms, e.g. DEFLATE or LZW are available in the ZIP standard and natively supported by most operating systems.

Based on this format specifications, the next steps towards the platform integration (work package 6) include:

- Development of a corresponding software module, e.g. Python package, for storing and handling of the new format data as well as convenient conversion from existing solutions and formats
- Conceptualization of preview or proxy files, e.g. via binning/undersampling, for convenient and fast visualization for large data sets
- Specification of exploration information, e.g. segmentation or label data based on currently used solutions by the partners (e.g. [V7 Darwin | AI Data Labeling & ML Training Data Platform](#) used at WUR)
- Implementation of the data format in the DIVE / Fraunhofer software solutions VESolve / imanto pro based on the existing ENVI interface

3.1. Harmonization with the new IEEE 4001 standard

Since May 2018, a working group under the IEEE Geoscience and Remote Sensing Society Standards Committee, involving a mayor part of the hyperspectral imaging community, has been developing the IEEE 4001 standard, which was just recently finalized (see doi.org/10.1117/12.3055117). The standard specifies a complete set of characteristics for hyperspectral camera performance and an extensive set of camera-related metadata.

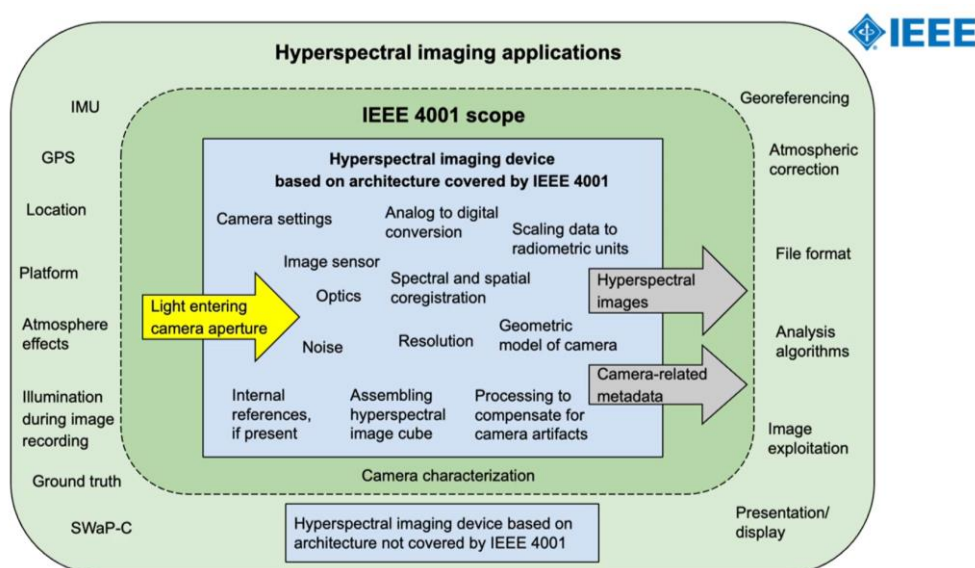


Figure 3: Sketch of the relevant characteristics included in the IEEE 4001 standard, reused from Skauli et al. "The IEEE P4001 standard for hyperspectral imaging," Proc. SPIE 13455 (28 May 2025).

HyperImage

While this standard does not include the specification of file formats or data flow, data processing or exploitation, the relevant and mentioned characteristics are to some extent already incorporated in the ENVI metadata. However, a future alignment and harmonization of the metadata and stored reference data with the new standard is needed for a most broad compatibility.

4. Conclusions

The frame of a new hyperspectral imaging data format based on the ENVI standard is proposed. The format centralizes metadata in an extended ENVI header and supports diverse auxiliary files for future machine-learning, calibration, and visualization tasks. It is packaged in an uncompressed ZIP container for robust data exchange and internally supports optional advanced compression algorithms. The next steps include demonstrating the advantages of the format within the use cases of the HyperImage project, e.g. in semiconductor inspection and vertical farming, and the further integration in a cloud-based spectral data processing infrastructure.

5. Dissemination level

Public