www.cad-journal.net
Computer-Aided Design AND APPLICATIONS

Storing Optic Nerve Head Geometry in a Category-Major PLY Format

John K. Johnstone[1]

[1]Computer Science and Ophthalmology, The University of Alabama at Birmingham, jkj@uab.edu

Corresponding author: John K. Johnstone, jkj@uab.edu

Abstract. Shape modeling and measurement of the optic nerve head is important for a better understanding of the progress of diseases of the eye associated with deformation of the optic nerve head. The dataset for this optic morphometry is a special point cloud, gathered from certain structures in the optic nerve head, called an ONH dataset. This paper proposes a new data format for ONH datasets, adopting a variant of the widely used PLY format and organizing the points by category, a more natural organization for computation. The paper discusses the present data format for an ONH dataset, the new format and its advantages, the adaptations of the PLY format necessary for the new format, and a comparison of the new format with other data formats.

Keywords: data management, data format, PLY format, morphometry, optic nerve head
DOI: https://doi.org/10.14733/cadaps.2025.555-565

1 INTRODUCTION

Reconstruction of the optic nerve head (ONH) is important for an improved understanding of the evolution of ONH shape, especially under the influence of increased intraocular pressure, in diseases like glaucoma. The underlying shape models are reconstructed from point samples recorded in ONH datasets, the raw data of optic morphometry. This paper introduces a new data format for ONH datasets. The new format is organized by category rather than by slice, which more naturally reflects computation with the point clouds. A data format that reflects the desired internal data structure is clearer and more transparent, less prone to misinterpretation, and streamlines use of the data format. The new format replaces a bespoke private format by a familiar format for point clouds, adapting the widely used Stanford PLY format [28, 29]. This encourages swift adoption by other research groups and sharing of software that works with the format. Finally, the new format is concise and self-documenting.

A good data format is an important catalyst to computation and to software sharing. A good data format should be clear, concise, well documented (optimally as part of the format itself), should reflect the desired internal structure of the data, and should ideally adopt a familiar and widely accepted standard format, to encourage simple adoption by the research community. In this paper, we develop a new data format for ONH datasets with these characteristics, adapting the widely accepted Stanford PLY format.

The rest of the paper is structured as follows. Section 2 provides the minimal necessary background on optic morphometry for an understanding of the ONH dataset and its preferred structure. Section 3 discusses the present data format for ONH datasets. Section 4 defines the new data format for ONH datasets, as an adaptation of the PLY format, and the associated data structure. Section 5 discusses the advantages of the new format, analogies to other formats, and the C++ implementation. Section 6 discusses other data formats and datasets of point clouds and ophthalmological datasets. Section 7 ends with conclusions and future work.

2 OPTIC MORPHOMETRY

We start by explaining the ONH dataset and its use in optic morphometry. To build an ONH dataset, the optic nerve head is imaged into a volume using spectral-domain optical coherence tomography (OCT) [16, 24]. This volume is then sliced into images by planes rotating around an axis through the middle of the optic nerve head. It makes sense for the planar slices to rotate around an axis through the middle of the optic nerve head, rather than to lie in parallel planes as is typical in contour datasets, to focus the gathering of data to the cylinder of the optic nerve head. Finally, each of the images is segmented into the anatomical categories of interest. Since this segmentation is subtle and challenging, it is done by hand. This defines a point cloud organized both into categories and slices, which we call an **ONH dataset**. Fig. 1 shows an ONH dataset and its organization into categories and slices.

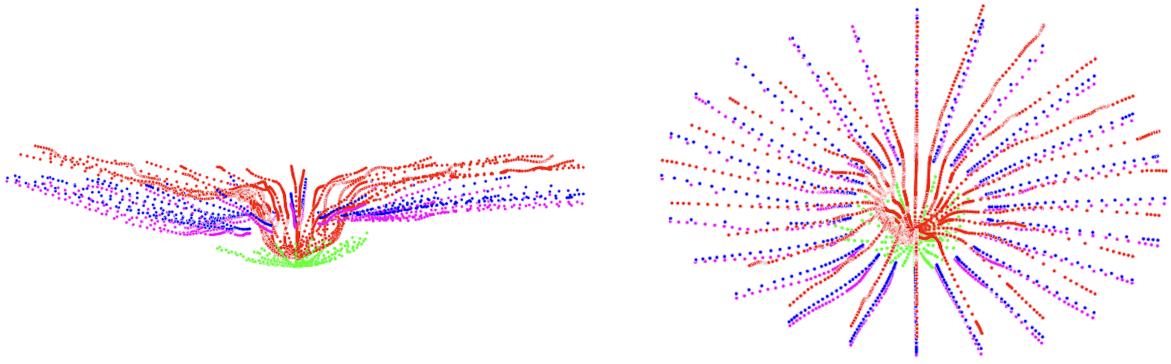

Figure 1: An ONH dataset with four categories. Left: The categories: ILM (red), BM (blue), AS (magenta), and AL (green) Right: A different viewpoint of this dataset, showing its organization into radial slices.

The four most important categories of the optic nerve head are the inner limiting membrane (ILM), anterior lamina (AL), Bruch's membrane (BM), and anterior sclera (AS). All datasets will contain these four main categories, since they are visible in all images. However, a data format must have the flexibility to allow a variable number of categories, since other categories of lesser interest may also be included, such as the anterior scleral canal opening, while histological datasets (where more is visible) will contain more categories, such as the posterior lamina.

ONH datasets are the raw data for optic morphometry. Key morphometric measurements are defined from ILM and AL, such as cup depth (signed distance of ILM from a reference plane), laminar cup depth (distance of AL from a reference plane), cup volume (volume within a reference cylinder from a reference plane to ILM on the laminar side), and laminar cup volume (volume within a reference cylinder from a reference plane to AL). These measurements are defined with respect to a reference plane and a reference cylinder, defined either from the opening of Bruch's membrane [13] or anterior sclera [12].

3 THE PRESENT SLICE-MAJOR FORMAT FOR ONH DATASETS

Since an ONH dataset (Fig. 1) consists of categories and slices, the storage of its point cloud may be organized either by category or by slice. That is, each point has two main attributes, its category and its slice, and the point cloud will be first grouped by one attribute, and then within each of these groups, by the second attribute. The choice of the primary attribute dictates the storage order. Since this is analogous to row-major vs column-major storage of a matrix, we call these two storage orders category-major and slice-major.

In **slice-major order**, the primary attribute is slice: the point cloud is stored slice by slice, then within each slice, category by category. In **category-major order**, the primary attribute is category: the point cloud is stored category by category, then within each category, slice by slice. In either slice-major or category-major order, the points of a slice's category (or a category's slice) are sorted along the slice.

The present data format for ONH datasets is slice-major. This is an artefact of the acquisition of the ONH dataset's point cloud by image segmentation, slice by slice. However, a slice-major format is awkward, since computation with the optic nerve head is category-centric: for example, the extraction of a reference plane and ellipse from the opening of Bruch's membrane, or surface reconstruction of ILM for computation of cup depth. The present format is also bespoke, in a local format that is not standard for point clouds. We now describe it in more detail.

The present format is organized as follows. Each line of the file represents a point and has 6 entries. Here is a typical line:

```
5011.16 4816.23 1096.28 30 4 0
```

The first three entries of each line are the Cartesian coordinates of the point; each is a floating point scalar. The coordinate frame of these points is not important since, in an early stage of computation, the dataset will be shifted to another coordinate frame, defined by the reference structures.

The fourth entry of each line is the category code of the point. The codes of some important categories are given in Table 1. Certain categories are divided into left and right parts, since the structure has a gap as it crosses the optic nerve head. For example, BM (blue in Fig. 1, codes 30 and 31) is divided into left and right categories since it has a gap, while ILM (red in Fig. 1, code 120) is not divided into left and right.

Bruch's Membrane Left	30
Bruch's Membrane Right	31
Anterior Lamina Surface	45
Anterior Sclera Left	50
Anterior Sclera Right	51
Inner Limiting Membrane	120

Table 1: Category codes of some important ONH structures in datasets segmented from OCT

The fifth entry of each line is the 0-based slice index of the point. Although the increment between consecutive slice indices is consistent, the typical increment is 4, not 1, and could be a float like 4.5.

The sixth entry of each line is a boolean mark, associated with a special property of the point during segmentation. It is ignored by all morphometric computations.

As mentioned, the present format is slice-major: all points of the first slice are recorded, then all points of the second slice, and so on. Within a slice, the points are sorted by category code: for example, the first points of the dataset are the points of the first slice from the category with the lowest category code in the dataset, sorted along the slice.

4 THE NEW CATEGORY-MAJOR PLY FORMAT FOR ONH DATASETS

We now describe the new format for ONH datasets, which pivots to a category-major format. The new format adopts a standard data format for point clouds and polygon meshes, Stanford's PLY format [28], that is widely used in computer graphics and shape modeling, with existing tools such as readers [11] and familiary within the research community. Another nice feature of the PLY format is that it allows for natural documentation of the semantics of its data entries within the dataset itself, in a lightweight fashion. This adds to the clarity of the data format. In short, we shift from a bespoke acquisition-based slice-major format to a standardized computation-based category-major format.

It turns out that both the fifth and sixth entries (slice index and boolean mark) of the old format are vestigial, since they are not used in any of the standard morphometric computations and do not influence any geometry. Like the organization of the dataset into slices, they are another aspect of the present format that reflects the segmentation process. Therefore, these components of the dataset are stripped from the new format.

4.1 PLY Format

The new data format is an adaptation of the PLY format, a format developed by the Stanford Computer Graphics Laboratory [28], originally designed for polygon meshes but, anticipating unknown future needs, made flexible enough to also represent point clouds and essentially any geometric data. PLY is a standard in shape modeling and computer graphics, widely adopted because of its flexibility, its rigour, and its ability to document a format clearly. This has also led to software for handling PLY files, such as Sharp's happly [11]. Another nice feature is that data can be stored in plain text or in binary.

We now give a concise description of PLY format; more details are available through various sources including Turk's official documentation [29, 25]. Then we discuss how we have adapted the PLY format for ONH datasets.

A PLY file is organized into elements. The file begins with a **header**, describing each element in the data, including its name, a count of the number of these elements, and a list of the various properties associated with the element, with their types. Arbitrary comments are also allowed in the header. The header is followed by the **body**, containing the actual data of the shape.

```
ply
format ascii 1.0
comment zipper output
comment modified by flipply
element vertex 35947
property float32 x
property float32 y
property float32 z
property float32 confidence
property float32 intensity
element face 69451
property list uint8 int32 vertex_indices
end_header
-0.0378297 0.12794 0.00447467 0.850855 0.5
...
3 20399 21215 21216
```

Figure 2: Header and first vertex/face of the PLY file for the Stanford Bunny

For example, Fig. 2 is the header of the famous Stanford Bunny [28], a triangle mesh whose PLY file has two elements (vertex and face), with the vertex element having five properties (three Cartesian coordinates, confidence, and intensity, each a 32-bit float) and the face element having one property (an integer list, of the 0-based indices of the vertices of this face). The header is followed by the data itself in the body, in this case 35,947 vertices and 69,451 faces. We have shown the first vertex entry and the first face entry.

4.2 ONH PLY Format

The new data format for ONH datasets adapts the PLY format (Fig. 3).

```
ply
format ascii 1.0
comment 30  BML   Bruch's Membrane - Left
comment 31  BMR   Bruch's Membrane - Right
comment 45  AL    Anterior Lamina Surface
comment 50  ASL   Anterior Sclera Surface - Left
comment 51  ASR   Anterior Sclera Surface - Right
comment 120 ILM   Inner limiting membrane
comment 200 NFLL  Posterior RGC/NFL - Left
comment 201 NFLR  Posterior RGC/NFL - Right
element category 1
property list uchar int code
element section 8
property list uchar int start
element precision 1
property int p
element vertex 4322
property double x
property double y
property double z
end_header
8 30 31 45 50 51 120 200 201
12 0 34 65 93 124 164 207 247 293 348 391 428
12 459 508 542 567 584 603 640 677 723 754 794 822
12 859 874 907 923 940 956 967 979 991 1000 1010 1020
12 1034 1054 1079 1101 1126 1151 1173 1190 1215 1233 1246 1271
12 1286 1308 1333 1358 1383 1408 1433 1455 1477 1496 1521 1549
12 1574 1863 2089 2306 2454 2623 2762 2916 3055 3254 3393 3586
12 3821 3837 3859 3893 3921 3943 3965 3990 4015 4043 4068 4090
12 4112 4140 4162 4181 4194 4210 4229 4248 4258 4268 4287 4303
3
4745.190 4612.820 1103.990
```

Figure 3: The header and first few entries of an ONH dataset in the proposed PLY format

Like the PLY polygon mesh, it defines a point cloud using the vertex element. Like the PLY polygon mesh, it imposes a structure on the point cloud using other elements, but instead of using a face element to define polygons, it uses a section element to define sections (a **section** is the point cloud of a certain category in a certain slice), and a category element to define categories. By leveraging the added structure

of a category-major point cloud, the definition of sections and categories is streamlined, reducing to a single integer per section and a single integer per category.

We now define the PLY elements of an ONH dataset more precisely (Table 2). It may be useful to follow along with the example of an ONH dataset in Fig. 3; this dataset contains 8 categories, each with 12 sections, and 4322 points in the entire point cloud. Note that, like most ONH datasets, it has some additional categories (posterior RGC/NFL) beyond the standard ones (ILM, AL, BM, AS).

Element	Property(s)	Type
category	code	integer list
vertex	x, y, z	double, double, double
section	start	integer list
precision	p	integer

Table 2: The elements and properties of an ONH dataset in PLY format

The **category element** represents the categories in the point cloud. It has one property (code), which is a list of category codes (Table 1). This element specifies the order of the categories in the category-major point cloud (see vertex element), which is also the order of the categories in the section lists (see section element). For example, the dataset of Fig. 3 has 8 categories, with codes 30, 31, 45, 50, 51, 120, 200, and 201, in that order.

The capacity to add comments in the PLY header is leveraged to document the categories associated with these category codes. For example, the first category, with code 30, is the left half of Bruch's membrane. Note that the old format has no documentation within the dataset of the interpretation of the category codes.

The **vertex element** represents the entire point cloud of samples (all categories, all slices) in category-major order. It has three properties x, y, and z, representing the Cartesian coordinates. This element is analogous to the vertex element of a polygon mesh. The dataset of Fig. 3 has 4322 vertices. and the first vertex has x-coordinate 4745.190.

The **section element** represents the sections of a single category. Each section is described by a single integer, the 0-based index of the first point of the section. This leverages the category-major structure of the point cloud. Only the start index is necessary, since the end index is implied by the start index of the next section, since the sections of a category are consecutive. If there is no next section, then the end index of the present section is implied by the size of the point cloud, which is specified in the vertex header element. The section element has one property (start), which is a list of the start indices of the sections of the category in question. Therefore, each line of the body associated with the section element is a list of integers of length n_s, where n_s is the number of rotational slices through the ONH volume (often 12), representing the sections of a single category. Integer lists in PLY format begin with the length of the list. For example, in Fig. 3, the third section entry (the fourth line of the body) lists the 12 sections of the anterior lamina (45, the third category), and the first of these 12 sections runs from the point with index 859 to the point with index 873.

The **precision element** represents the number of digits recorded after the decimal point in point cloud floats. It is useful for preserving the same precision when writing the format in plaintext. Ideally, the read/write machinery of a dataset f satisfies write(read(f)) = f, and recording the precision enables this property. Notice that floats in Fig. 3 are recorded to 3 decimal places.

An important characteristic of a good data format is that it reflects the desired internal structure of the data. To illustrate this characteristic of the new format, we outline below the internal data structure of an ONH dataset, which has proved computationally successful for morphometry and rendering. We have pointed out the mirroring of the format in each entry:

1. the point cloud of the entire optic nerve head, which is stored in a column-major data matrix of shape (3, nPt), with the columns/points sorted by category, then by slice within each category;
   this mirrors the vertex element of the format (Table 2)
2. an integer vector of the category codes (Table 1), in order of their appearance in the point cloud;
   this mirrors the category element of the format
3. an associative map, mapping between integer category codes and their category names (strings);
   this mirrors the comments in the PLY header
4. the number of slices per category, an integer;
   this mirrors the length of each integer list associated with the section element in the format (e.g., 12 slices in the example of Fig. 3)
5. a data matrix of shape (2, nCat * nSlice), whose $j * \text{nCat} + i$th column stores the start and end indices of the ith section of the jth category (indexing into the global point cloud);
   this mirrors the section element of the format; the end index is added, although redundant since extractable from the start indices, for simplicity of computation
6. the precision of the point cloud's floats, to be used in writing;
   this mirrors the length of each point coordinate of the format

5 DISCUSSION AND IMPLEMENTATION

What are the main improvements of the new data format for an ONH dataset?

5.1 PLY Format

The new format adapts a widely accepted data format, the PLY format. This means that most researchers with a background in computer graphics or shape modeling will be immediately comfortable with the data. A widely used format is more transparent and more clear, since it is familiar. A widely used format speeds adoption. A widely used format implies the presence of existing tools for handling the data, such as Sharp's hapPLY [11] (an open-source C++ parser for the PLY format), a Julia parser for PLY files [26], tools for handling PLY in MeshLab [4], tools for handling PLY in the Wolfram language [31], and so on.

The new format is self-documenting. In contrast to the old format, which is simply a collection of scalars, the header of the new format documents the structure of the data clearly, especially for someone familiar with PLY format. This even includes documentation of the categories associated with the abstract category codes. It is also useful to have point counts for the entire dataset, each category, and each section, readily available from the header (see Fig. 3). Since the new format is an adaptation of a widely used standard format and self-documenting, its use is aligned with new NIH/NSF mandates on data management and sharing of funded research [19, 20].

5.2 Category-major

The new format is category-major, which mirrors the computational framework that works with categories. Consider the category-major structure of morphometric computation (see end of Section 2). Cup volume requires reconstruction of the inner limiting membrane. Laminar cup volume requires reconstruction of the anterior lamina. The reference plane requires principal component analysis with Bruch's membrane or with the anterior sclera. Cup depth works with ILM, and laminar cup depth with AL. All of these work with a single category, not a single slice. A data format that is organized by category adds transparency by matching the underlying data structure, which is organized by category.

The advantages of a category-major storage of an optic nerve head point cloud for optic morphometry are analogous to the advantages of a column-major storage of a point cloud for OpenGL/C++ rendering. Consider a C++ implementation of a point cloud renderer using Eigen [6], the preferred numerical library in C++, and OpenGL [23], the preferred graphics library in C++. An example is the rendering of a single category, or all categories, in our viewer of ONH datasets. A **column-major data matrix for a point cloud** is a matrix where each column stores a point of the point cloud. For example, the shape of this matrix would be (3,n) if the point cloud has n points in 3-space. Note that the default, and preferred, storage order of a matrix in Eigen is column-major. Indeed the Eigen documentation states that there are no guarantees for a matrix in row-major storage. Using a column-major data matrix has significant benefits when rendering the point cloud using OpenGL and Eigen. When binding an OpenGL vertex array object in preparation for rendering the point cloud, glBufferData can reference the entire block of data with a single pointer (in the data parameter of the glBufferData function [9]), since the points are a contiguous block (consider the matrix flattened in memory). Then only a single glDrawArrays call is necessary in the display loop, since the point cloud is a contiguous block of data. This makes the column-major data matrix for a point cloud the preferred representation for a point cloud, in analogy to the preference for a category-major representation for the optic point cloud.

5.3 Concision

The new format is concise: a single integer is sufficient to encode each section and each category. For example, the overhead of the category and section elements in the dataset of Fig. 3 is the first 8 lines of the body. This efficiency is thanks to the category-major order of the point cloud, which gives it an implied hierarchical tree structure. Every point of the old format (Section 3) redundantly includes the category code and slice index (as well as a vestigial mark). By laying on a hierarchical structure of section starts, the PLY format does not need to explicitly store the category and slice of a point: it is implicit. Only one additional scalar is required for an entire section of points, recording the start index of the section. In particular, the space complexity of the new format is approximately half that of the old format: 3*nPt + nCat + nSec, rather than 6*nPt. For example, if a dataset contains 10,000 points, 6 categories, and 72 sections (12 per category), then the new dataset require 30,078 entries rather than 60,000.

The concision of the new format for ONH datasets is analogous to the concision of a sparse matrix format, where unnecessary information is not stored explicitly. In compressed row storage (CRS) of a sparse matrix [5], only nonzero values are stored, with some additional structural elements: nonzero values are stored in row-major order, along with the (1-based) column indices of these values, and the indices of values that start a row. Rather than storing n^2 scalars from a dense $n \times n$ matrix, only n_{nz} scalars and $n_{nz} + n + 1$ integer indices must be stored, where n_{nz} is the number of nonzero entries. Just as, in a sparse matrix format, zero elements are not stored, in our new category-major format, category codes, section indices, and point marks are not stored.

5.4 Implementation

To allow the new data format to be easily incorporated into ONH morphometry research, C++ code has been written for several major components: 1) reader/writer machinery for both the old and new formats; 2) translation code from the old format to the new format, and from the new format to the old format; and 3) an OpenGL viewer of ONH datasets in the new format. All of this code uses Eigen [6], a C++ template library for linear algebra, for matrix and vector storage.

This code benefits from the use of Nick Sharp's hapPLY, an open-source C++ parser of the PLY format [11]. However, minor adaptations of hapPLY need to be made to accommodate two elements of our data format: the precision element and the embedded comments in our PLY header (Fig. 3) that document the category names associated with codes. In particular, an ability to control precision (number of digits after

decimal place) in writing is added, by adding an integer precision parameter to the writeDataASCII, write, and writePLY functions; and an ability to write extra comments is added, by adding a comment parameter (vector of strings) to the write, writePLY, and writeHeader functions. A getter of the comments member variable is also added, so that our reader can use the embedded comments in our PLY header to build an associative map between category codes and category names, which can be used, for example, in the viewer to document categories.

6 DATASETS FROM COMPARABLE LITERATURE

We have discussed a new data format for point clouds of the optic nerve head. That is, a format for a point cloud dataset (with an unusual organization) and an ophthalmological dataset. We now consider other point cloud datasets, and other ophthalmological datasets, along with their formats.

First, point cloud datasets. A common source of point clouds is LIDAR, such as KITTI [8], SemanticKITTI [1], and KITTI-360 [17]. KITTI-360, for example, includes both Velodyne scans and point clouds in PLY format. Another common source of point clouds is GIS data in the form of topographic maps (contour datasets). Many of these datasets are released by the U.S. Geological Survey [30], such as the National Elevation Dataset. Contour datasets are also used to store point clouds from biomedical data (CT/MR), again organized in parallel planes (e.g., Geiger in Nuages [21]). Both LIDAR datasets and contour datasets are structured point clouds, although with a substantially different organization than ONH datasets. Point clouds are often sampled from polygon meshes (e.g., the point clouds in Hanocka [10] and Qi [27]), using mesh datasets such as Thingi10k [33], ModelNet40 [32] (over 10K CAD models in OFF format), and Choi [3] – which includes both scans as RGB-D sequences (JPG color images and PNG depth images) and reconstructed meshes in PLY format.

Various formats are used for the point clouds. LIDAR may use raw PLY [17]. GIS datasets use special formats such as FileGDB (file geodatabases). Geiger uses a proprietary NUAGES format [21]; Hanocka and Qi [10, 27] use raw point clouds with no structure, while the underlying meshes are often in PLY format. Note that if there is additional structure added to the point clouds, it will typically be in parallel planes, rather than the rotational slices of ONH datasets, and there will be no organization into categories.

Now consider ophthalmological datasets. These are dominated by image datasets, such as OCT images (Section 2). A collection of public-domain image datasets related to ophthalmology has been collected by Khan [15], such as OCT image datasets from Duke [2, 7], and OCT volumes [18, 22] from NYU and IBM. The Duke datasets are stored in Matlab's MAT format, while the NYU datasets are stored in NumPy format. We are not aware of any public domain datasets of optic nerve head point clouds, although many major ophthalmology labs maintain their own proprietary datasets.

7 CONCLUSIONS AND FUTURE WORK

This paper has presented a new data format for optic nerve head datasets, a fundamental research tool in ophthalmology. The new data format adapts the PLY format for polygon meshes, a widely accepted format, and switches from a slice-major to a category-major structure, which better matches computational needs. The new format is more space efficient, about half the space of the earlier format, offers binary options for even further compression, and is self-documenting, improving on the bespoke earlier format.

There are several directions for future work. We have developed mechanisms to guarantee correct I/O behaviour in reading and writing a dataset [14], and tests of robust translation between two data formats for the same data, guaranteeing no loss or corruption of data, where these ONH datasets are a natural testbed. It would be interesting to consider augmenting PLY format with a string type: presently the only data types in PLY are scalar [29] (char, uchar, short, ushort, int, uint, float, double). The addition of a non-numeric string type would allow a PLY element to record the category names, rather than the present hack of recording the

category names in the comments of the PLY header. As the ONH datasets of this paper are used to build shape models in the morphometric pipeline, there is the potential to extend the format with new PLY elements to record intermediate structures, such as reference planes and cylinders, B-spline reconstructions of sections, mesh reconstructions of a category, and so on. The addition of an RGB element for rendering ONH models is also possible.

ACKNOWLEDGEMENTS

We appreciate discussions with colleagues in UAB's Department of Ophthalmology about optic morphometry, and access to the ONH datasets used in this study.

John K. Johnstone, https://orcid.org/0000-0003-4033-0066

REFERENCES

[1] Behley, J.; Garbade M.; Milioto, A.; Quenzel, J.; Behnke, S.; Gall, J.; Stachniss, C.: Towards 3D LiDAR-based semantic scene understanding of 3D point cloud sequences: The SemanticKITTI Dataset. International Journal on Robotics Research, 40(8-9), 2021, 959–967. https://doi.org/10.1177/02783649211006735

[2] Chiu, S.; Izatt, J.; O'Connell, R.; Winter, K.; Toth, C.; Farsiu, S.: Validated Automatic Segmentation of AMD Pathology including Drusen and Geographic Atrophy in SDOCT Images. Investigative Ophthalmology & Visual Science, 53(1), 2012, 53–61. https://doi.org/10.1167/iovs.11-7640

[3] Choi, S.; Zhou, Q.-Y.; Miller, S.; Koltun, V.: A Large Dataset of Object Scans. arXiv:1602.02481 (2016).

[4] Cignoni, P.; Callieri, M.; Corsini, M.; Dellepiane, M.; Ganovelli, F.; Ranzuglia, G.: MeshLab: an Open-Source Mesh Processing Tool. Sixth Eurographics Italian Chapter Conference, 2008, 129–136. https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136

[5] Compressed Row Storage: https:://netlib.org/utk/people/JackDongarra/etemplates/node373.html

[6] Eigen: https://eigen.tuxfamily.org

[7] Farsiu, S.; Chiu, S.; O'Connell, R.; Folgar, F.; Yuan, E.; Izatt, J.; Toth, C.: Quantitative Classification of Eyes with and without Intermediate Age-related Macular Degeneration Using Optical Coherence Tomography. Ophthalmology, 121(1), 2014, 162–172. http://doi.org/10.1016/j.ophtha.2013.07.013

[8] Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R.: Vision meets Robotics: The KITTI Dataset. International Journal of Robotics Research, 32(11), 2013, 1231–1237. https://doi.org/10.1177/0278364913491297

[9] glBufferData: https://registry.khronos.org/OpenGL-Refpages/gl4/html/glBufferData.xhtml

[10] Hanocka, R.; Metzer, G.; Giryes, R.; Cohen-Or, D.: Point2Mesh: a self-prior for deformable meshes. ACM Transactions on Graphics, 39(4), 2020. http://doi.org/10.1145/3386569.3392415

[11] hapPLY: a C++ header-only parser for the PLY file format. https://github.com/nmwsharp/happly

[12] Johnstone, J.; Fazio, M.; Rojananuangnit, K.; Smith, B.; Clark, M.; Downs, J.C.; Owsley, C.; Girard, M.; Mari, J.-M.; Girkin, C.: Variation of the Axial Location of Bruch's Membrane Opening with Age, Choroidal Thickness and Race. Investigative Ophthalmology and Visual Science, 55(3), 2014, 2004–2009. https://doi.org/10.1167/iovs.13-12937

[13] Johnstone, J.; Rhodes, L.; Fazio, M.; Smith, B.; Wang, L.; Downs, J.C.; Owsley, C.; Girkin, C.: Measuring Mean Cup Depth in the Optic Nerve Head. Computer Aided Design and Applications, 13(5), 2016, 693–700. https://doi.org/10.1080/16864360.2016.1150716

[14] Johnstone, J.: Lost in Translation: Evaluating the Robustness of a Data Format and its Read/Write Machinery. CAD 2024 (Eger, Hungary), 2024, 227–231. https://doi.org/10.14733/cadconfP.2024.227-231

[15] Khan, S.; Liu, X.; Nath, S.; Korot, E.; Faes, L.; Wagner, S.; Keane, P.; Sebire, N.; Burton, M.; Denniston, A.: A global review of publicly available datasets for ophthalmological imaging: barriers to access, usability, and generalisability. Lancet Digit Health, 3(1), 2021, 51-66. http://doi.org/10.1016/S2589-7500(20)30240-5

[16] Lee, E.; Kim, T.; Weinreb, R.; Park, K.; Kim, S.; Kim, D.: Visualization of the lamina cribrosa using enhanced depth imaging spectral-domain optical coherence tomography. Am J Ophthalmol, 152(1), 2011, 87–95. https://doi.org/10.1016/j.ajo.2011.01.024

[17] Liao, Y.; Xie, J.; Geiger, A.: KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(3), 2023, 3292-3310. https://doi.org/10.1109/tpami.2022.3179507

[18] Maetschke, S.; Antony, B.; Ishikawa, H.; Wollstein, G.; Schuman, J.; Garnavi, R.: A feature agnostic approach for glaucoma detection in OCT volumes. PLOS ONE, 14(7), 2019. http://dx.doi.org/10.1371/journal.pone.0219126

[19] NIH Data Management and Sharing Policy: https://grants.nih.gov/grants/guide/notice-files/NOT-OD-21-013.html

[20] NSF Proposal and Award Policies and Procedures Guide: Dissemination and Sharing of Research Results: https://new.nsf.gov/policies/pappg/23-1/ch-11-other-post-award-requirements#11D4

[21] Nuages: https://www-sop.inria.fr/prisme/logiciel/nuages.html.en

[22] OCT volumes for glaucoma detection: https://zenodo.org/records/1481223#.%20Xr06Q2gzbIU

[23] OpenGL: https://www.opengl.org/

[24] Park, S.; De Moraes, C.; Teng, C.; Tello, C.; Liebmann, J.; Ritch, R.: Enhanced Depth Imaging Optical Coherence Tomography of Deep Optic Nerve Complex Structures in Glaucoma. Ophthalmology 119(1), 2012, 3–9. https://doi.org/10.1016/j.ophtha.2011.07.012

[25] PLY (file format). https://en.wikipedia.org/wiki/PLY_(file_format)

[26] PlyIO.jl, Ply polygon file IO: https://github.com/JuliaGeometry/PlyIO.jl

[27] Qi, C.; Su, H.; Mo, K.; Guibas, L.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR, 2017, 77–85. http://doi.org/10.1109/CVPR.2017.16

[28] The Stanford 3D Scanning Repository. https://graphics.stanford.edu/data/3Dscanrep

[29] Turk, G.: The PLY Polygon File Format. https://gamma.cs.unc.edu/POWERPLANT/papers/ply.pdf

[30] US Geological Survey GIS Data Download: https://www.usgs.gov/the-national-map-data-delivery/gis-data-download

[31] Wolfram PLY: https://reference.wolfram.com/language/ref/format/PLY.html

[32] Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J.: 3D ShapeNets: A Deep Representation for Volumetric Shapes. CVPR, 2015, 1912–1920. http://doi.org/10.1109/cvpr.2015.7298801

[33] Zhou, Q.; Jacobson, A.: Thingi10K: A Dataset of 10,000 3D-Printing Models. arXiv:1605.04797 (2016).