1 Overview

This release contains a co-registered in-situ and ex-situ *Peregrine* dataset from five Concept Laser M2 Laser Powder Bed Fusion (L-PBF) stainless steel 316L builds containing 6,299 SS-J3 individually tracked tensile coupons. These data were collected at the Manufacturing Demonstration Facility (MDF) located at Oak Ridge National Laboratory (ORNL). The dataset includes layer-wise visible-light in-situ imaging data, the laser scan paths and parameters, in-situ temporal sensor data, room-temperature static tensile test results, and the target part geometries. Additionally, anomaly detections produced by a modified Dynamic Segmentation Convolutional Neural Network (DSCNN) [1] are provided. Figure 1 shows example slices of each of these data modalities for a single print layer. Download instructions and a full description of the dataset structure are provided in the following sections.

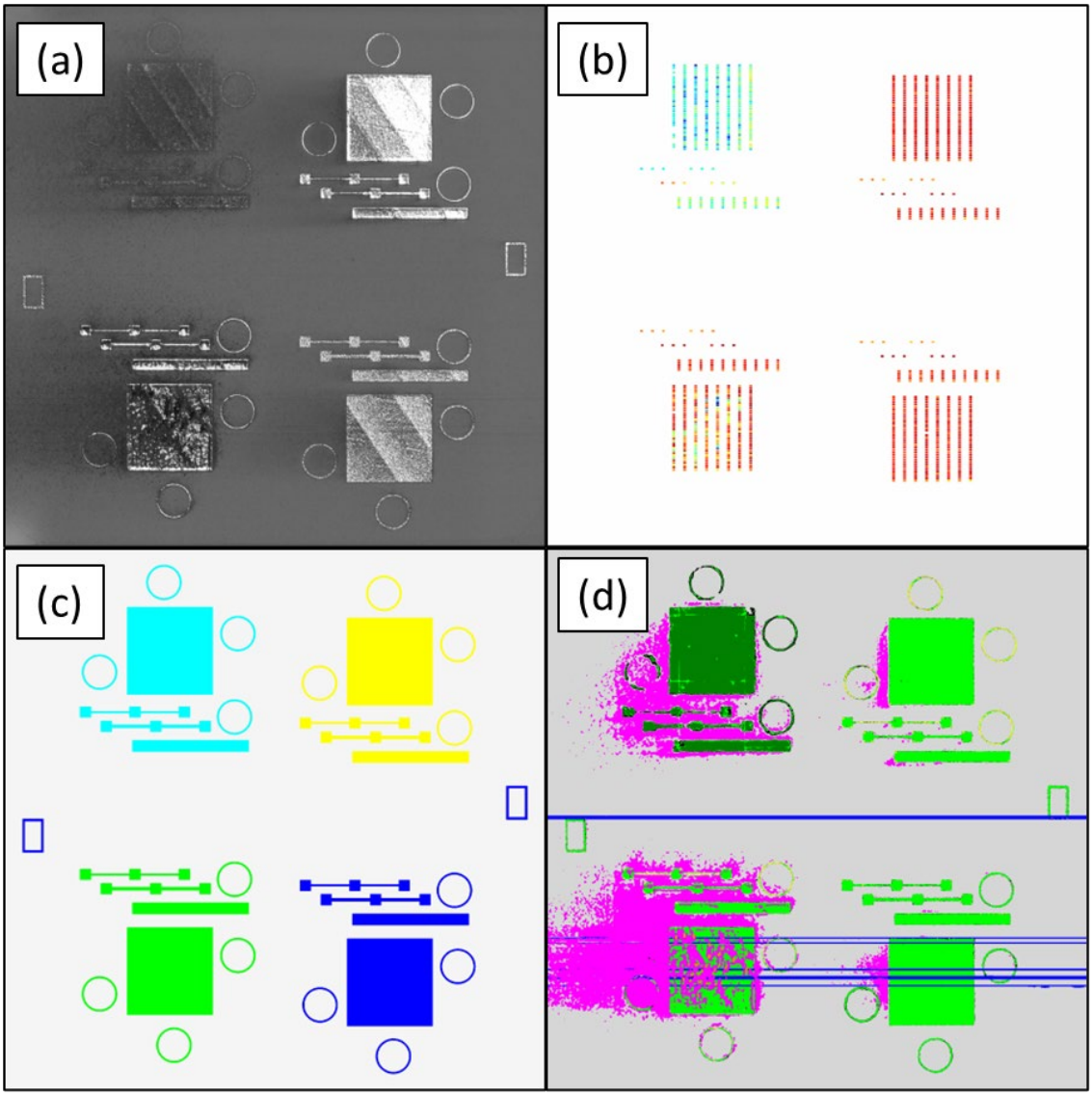

Figure 1. Co-registered data slices from layer 651, approximately 32.55 mm into the build height. (a) An in-situ visible-light image captured after the powder has been melted. (b) The ultimate tensile strength measured using SS-J3 coupons extracted from the printed parts. (c) The intended STL geometries for each printed part, colored by their process parameter set. (d) Anomaly segmentations produced by a trained DSCNN with each color representing a different anomaly class.

2 Dataset Download Instructions

1. Create a Globus account.
2. Create a Globus Endpoint on your computer. You may need to create an exception for Globus in your antivirus software so that it can create an Endpoint.
3. Transfer the dataset from the OLCF DOI-DOWNLOADS Collection to your Collection. Be sure to confirm that the transfer is going from OLCF DOI-DOWNLOADS to your Collection.
4. Sometimes users will need to manually create a Globus access directory (where the data will be downloaded) by going to the **Preferences > Access** tab before the download will begin.
5. Please direct any questions about accessing the dataset to Dr. Luke Scime scimelr@ornl.gov.

3 Dataset Organization

Unlike previous *Peregrine* dataset releases [2,3], this dataset is stored as a single HDF5 file. HDF5 is a common data format [4] optimized for working with large volumes of data and will be used by the *Peregrine* team for all planned future dataset releases. HDF5 files are hierarchical and self-describing, i.e., the data can be navigated similarly to a folder directory and metadata can be attached to each datum. The released *Peregrine* data can be grouped into seven categories based on how they are intended to be retrieved from the dataset. Users can explore the entire dataset using the *ExampleHDF5Open.py* Python 3.9 script provided alongside this dataset. In-line code snippets are also provided throughout this section for easy reference.

3.1 Build Metadata

Metadata applying to the entire build are stored as their native data type and can be accessed using their attribute key. The code snippet below accesses and prints out the **core/build_name** value, while the following list describes the contents of each high-level metadata grouping. If available, units for a given piece of metadata can be accessed by appending **/units** to the attribute key.

```
import h5py
with h5py.File(path_to_hdf5_file,'r') as build:
    print(build.attrs['core/build_name'])
```

- **core/:** Contains information uniquely identifying this build including the name and any notes.
- **layer_notes/#:** Contains user-notes specific to particular print layers, replace **#** with the layer.
- **log_file/:** This blob of text is extracted directly from the log file produced by the printer.
- **material/:** Contains information about the material composition and layer thickness.
- **people/:** Contains information about the people who requested and manufactured the parts.
- **printer/:** Describes the printer used to manufacture the build including its physical dimensions.
- **specimens/:** Contains information about the printed parts and any excised samples.
- **user_defined/:** Contains user-defined metadata fields that are typically printer-specific.

3.2 Reference Images

Reference images captured by the printer operator before, during, and after the build are stored as two-dimensional arrays and can be accessed by name and displayed using Matplotlib. The code snippet below displays the automatically generated **reference_images/thumbnail** which shows a two-dimensional projection of the three-dimensional part layout for this build.

```
import h5py
import matplotlib.pyplot as plt
with h5py.File(path_to_hdf5_file,'r') as build:
    plt.imshow(build['reference_images/thumbnail'][...],interpolation='none')
```

3.3 Slice Data

Slice data include the in-situ visible-light images, the connectivity maps for the intended part geometries and extracted samples, and the DSCNN anomaly predictions. Each sliced data modality is stored as a two-dimensional array and indexed by the layer number. All of the sliced data are in the same global coordinate system and have been resized to the pixel resolution of the in-situ visible-light data. Information about the coordinate system origin and axes are saved as metadata in the format **slices/origin**, **slices/x-axis**, etc. The code snippet below displays an in-situ visible-light image **slices/camera_data/visible/0** captured immediately after the layer has been melted. The remaining slice-type data are described in the following list.

```
import h5py
import matplotlib.pyplot as plt
with h5py.File(path_to_hdf5_file,'r') as build:
    plt.imshow(build['slices/camera_data/visible/0'][layer_number,...],
               cmap='jet',interpolation='none')
```

- **camera_data/visible/0:** Thes are in-situ images captured using a 5 MP visible-light camera immediately after each layer is melted by the laser beam.
- **camera_data/visible/1:** These are in-situ images captured using a 5 MP visible-light camera immediately after a new layer of powder is spread across the powder bed.
- **slices/part_ids:** Each printed part is automatically assigned a unique ID based on its location in the print volume. The corresponding part ID is stored for each pixel within the build volume, with an ID of zero representing regions outside of the intended print geometries.
- **slices/sample_ids:** Each sample extracted from a printed part, typically using wire electrical discharge machining, is assigned a unique Id based on its location within the volume of its parent printed part. The corresponding sample ID is stored for each pixel within the build volume, with an ID of zero representing regions not belonging to any of the samples.
- **slices/segmentation_results/#:** The pixel-wise anomaly predictions produced by the trained DSCNN model are based on the in-situ visible-light images and are stored as a set of binary arrays, with one array per anomaly class per print layer. Each array can be accessed by replacing the **#** in **slices/segmentation_results/#** with the class ID of interest. The corresponding enumerated list of classes can be retrieved using the **slices/segmentation_results/class_names** key. Brief descriptions of each class are provided in the following list.
  - **[0] Powder:** Areas of the powder bed which contain no detected anomalies or printed parts.
  - **[1] Printed:** Printed regions with no detected anomalies.
  - **[2] Recoater Hopping:** Ripples which occur if the recoater impacts parts below the surface.
  - **[3] Recoater Streaking:** Occurs if the recoater is damaged or dragging a large particle.

- **[4] Incomplete Spreading:** Occurs if insufficient powder is spread across the powder bed.
- **[5] Swelling:** A distortion or warping of the printed material that protrudes above the powder.
- **[6] Debris:** A catch-all class for small-to-medium disturbances in the powder bed.
- **[7] Super-Elevation:** A lack of powder coverage over top of a printed region.
- **[8] Spatter:** Ejecta from the weld pool which has landed back on the powder bed.
- **[9] Misprint:** Printed material detected outside of the intended part geometries.
- **[10] Over Melting:** Regions melted with high energy density process parameters.
- **[11] Under Melting:** Regions melted with low energy density process parameters.

3.4 Temporal Data

Temporal data are sourced from the log file produced by Concept Laser M2 printers and generally contain information regarding the health of the printer itself. Each temporal data modality is stored as a one-dimensional array and indexed by the layer number. If multiple measurements for the same sensor were captured during a single layer, they have been averaged together. The code snippet below displays a scatter plot of the **temporal/top_flow_rate** sensor measurement. Units for each temporal data modality can be retrieved by appending **/units** to the corresponding attribute key. The following list briefly describes each temporal data modality.

```
import h5py
import matplotlib.pyplot as plt
with h5py.File(path_to_hdf5_file,'r') as build:
    plt.scatter(np.arange(build['temporal/top_flow_rate'].shape[0]),
                build['temporal/top_flow_rate'])
```

- **temporal/absolute_image_capture_timestamp:** UTC time stamp for each printed layer.
- **temporal/actual_ventilator_flow_rate:** Measured argon flow through the ventilator.
- **temporal/bottom_chamber_temperature:** Temperature at the bottom of the build chamber.
- **temporal/bottom_flow_rate:** Measured argon flow across the bottom of the build chamber.
- **temporal/bottom_flow_temperature:** Temperature of the lower argon flow.
- **temporal/build_chamber_position:** Height of the build platform.
- **temporal/build_plate_temperature:** Temperature measured at the build plate.
- **temporal/build_time:** Cumulative time since the start of the build.
- **temporal/gas_loop_oxygen:** Oxygen concentration measured in the gas loop.
- **temporal/glass_scale_temperature:** Temperature measured in the laser optics.
- **temporal/laser_rail_temperature:** Temperature measured in the laser optics.
- **temporal/layer_times:** Time required to print each individual layer.
- **temporal/module_oxygen:** Oxygen concentration measured in the build chamber.
- **temporal/powder_chamber_position:** Height of the powder dosing platform.
- **temporal/target_ventilator_flow_rate:** Setpoint for the ventilator flow rate.
- **temporal/top_chamber_temperature:** Temperature at the top of the build chamber.
- **temporal/top_flow_rate:** Measured argon flow across the top of the build chamber.
- **temporal/top_flow_temperature:** Temperature of the upper argon flow.
- **temporal/ventilator_speed:** Measured ventilator speed.

3.5 Scan Path Data

Laser scan paths for Concept Laser M2 printers can be recovered from the QM Meltpool system [5] as described by Halsey et al. [6]. For each print layer, the scan paths are stored in a five column array with a number of rows equal to the number of scan vectors in the layer. To reduce the file size, only raster and contour scan paths are included in this dataset, i.e., the beam turnarounds and laser jump lines have been excluded. For each scan vector, columns 0 – 1 store the start and end *x* coordinates, columns 2 – 3 store the start and end *y* coordinates, and column 4 stores the relative time at which the vector was printed. Units for these values can be retrieved using the **scans/units** key. The code snipped below displays a line plot of the scan path data for a specified layer, coloring each vector based on its relative timing.

```
import h5py
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import matplotlib.cm as cm
import matplotlib.collections as collections
with h5py.File(path_to_hdf5_file,'r') as build:
    x = build['scans/%i' %(layer_number)][:,0:2]
    y = build['scans/%i' %(layer_number)][:,2:4]
    t = build['scans/%i' %(layer_number)][:,4]
    colorizer = cm.ScalarMappable(norm=mcolors.Normalize(np.min(t),np.max(t)),
                                  cmap='jet')
    line_collection = collections.LineCollection(np.stack([x,y],axis=-1),
                                                 colors=colorizer.to_rgba(t))
    fig = plt.figure('scan paths')
    ax = fig.add_subplot()
    plt.axis('scaled')
    ax.set_xlim(x.min(),x.max())
    ax.set_ylim(y.min(),y.max())
    ax.add_collection(line_collection)
```

3.6 Part and Sample Data

Data associated with specific printed parts or excised samples are stored as one dimensional arrays and indexed using the part or sample IDs contained in **slices/part_ids** or **slices/sample_ids**. Recall that indices of 0 should be ignored because the part and sample IDs begin at 1. The code snippet below prints a subset of the **samples/test_results/ultimate_tensile_strength** measurements. Units for each data modality can be retrieved by appending **/units** to the corresponding attribute key. Alternatively, some data may be stored as enumerated values, the mapping between the integers and the true values can then be retrieved by appending **/enums** to the corresponding attribute key. The following list briefly describes each set of part- or sample-specific data.

```
import h5py
with h5py.File(path_to_hdf5_file,'r') as build:
    key = 'samples/test_results/ultimate_tensile_strength'
    for i in range(1,min(build[key].shape[0],10),1):
        print('  %i ' %(i),build[key][i])
```

- **parts/process_parameters/hatch_spacing:** Spacing between each laser raster melt track.
- **parts/process_parameters/laser_beam_power:** Commanded raster laser beam power.
- **parts/process_parameters/laser_beam_speed:** Commanded raster laser beam velocity.
- **parts/process_parameters/laser_module:** Indicates which of two laser modules were used.
- **parts/process_parameters/laser_spot_size:** Commanded raster laser beam spot size.

- **parts/process_parameters/parameter_set:** User-defined names for each parameter set.
- **parts/process_parameters/scan_rotation:** Layer-wise scan strategy rotation.
- **parts/process_parameters/stripe_width:** Width of the raster stripes.
- **parts/test_results/burst_pressure:** Measured internal pressure at which the tube burst.
- **parts/test_results/burst_temperature:** Measured elevated temperature at which the tube burst.
- **parts/test_results/total_elongation:** Measured total elongation.
- **parts/test_results/ultimate_tensile_strength:** Measured ultimate tensile strength.
- **parts/test_results/uniform_elongation:** Measured uniform elongation.
- **parts/test_results/yield_strength:** Measured yield strength.

4 References

[1] L. Scime, D. Siddel, S. Baird, V. Paquit, Layer-wise anomaly detection and classification for powder bed additive manufacturing processes: A machine-agnostic algorithm for real-time pixel-wise semantic segmentation, Addit. Manuf. 36 (2020) 101453. https://doi.org/10.1016/j.addma.2020.101453.

[2] L. Scime, V. Paquit, C. Joslin, D. Richardson, D. Goldsby, L. Lowe, Layer-wise Imaging Dataset from Powder Bed Additive Manufacturing Processes for Machine Learning Applications (Peregrine v2021-03), (2021). https://doi.org/10.13139/ORNLNCCS/1779073.

[3] L. Scime, C. Joslin, R. Duncan, F. Brinkley, C. Ledford, D. Siddel, V. Paquit, Layer-wise Imaging Dataset from Powder Bed Additive Manufacturing Processes for Machine Learning Applications (Peregrine v2022-10), (2022). https://doi.org/10.13139/ORNLNCCS/1896716.

[4] A. Collette, HDF5 for Python, (2014). https://docs.h5py.org/en/stable/ (accessed August 25, 2023).

[5] ConceptLaser, Quality Management, (2020). https://www.ge.com/additive/sites/default/files/2018-02/1708_QM-pm_EN_update_1__lowres_einzel.pdf (accessed December 14, 2020).

[6] W. Halsey, D. Rose, L. Scime, R. Dehoff, V. Paquit, Localized Defect Detection from Spatially Mapped, In-Situ Process Data With Machine Learning, Front. Mech. Eng. 7 (2021) 1–14. https://doi.org/10.3389/fmech.2021.767444.