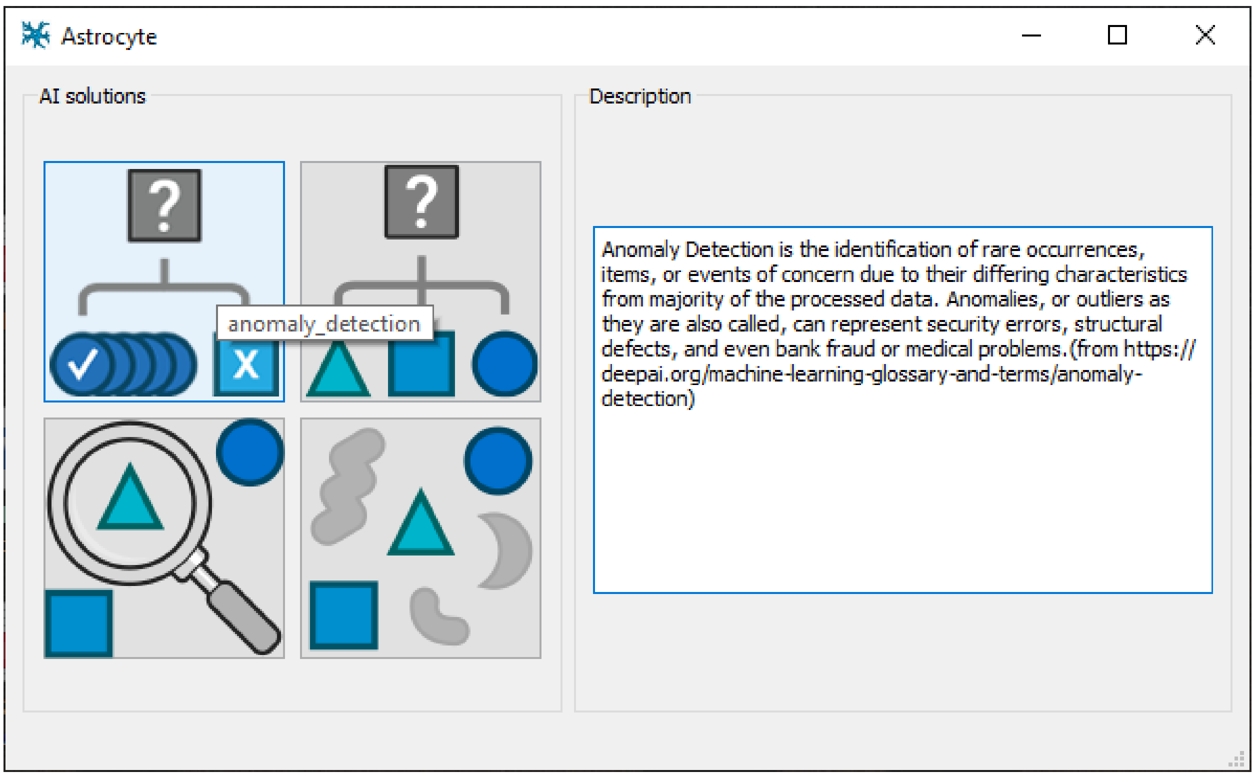
Astrocyte^{TM} Datasheet

Overview

Teledyne DALSA Astrocyte^{TM} is a code-free AI training tool to quickly deploy AI models for machine vision solutions. Astrocyte empowers users to harness their own images of products, samples, and defects to train neural networks to perform a variety of tasks such as anomaly detection, classification, object detection, segmentation. With its highly flexible graphical user interface, Astrocyte allows visualizing and interpreting models for performance/accuracy as well as exporting these models to files that are ready for runtime in Teledyne DALSA Sapera and Sherlock vision software platforms.



Key Features

- Graphical User Interface for rapid machine vision application development.
- Automatic tuning of training hyperparameters for maximum ease-of-use for non-experts in AI.
- Automatic generation of annotations via pre-trained models or semi-supervised training.
- Continual Learning (aka Lifelong Learning) in classification for further learning at runtime.
- Location of small defects in high-resolution images via tiling mechanism.

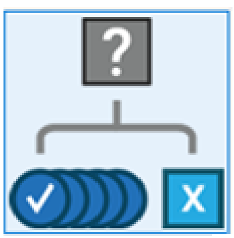
- Masking of regions to exclude from inspection via ROI markers of multiple shapes.
- Highly accelerated inference engine for optimal runtime speed on either GPU or CPU.
- Easy integration with Sapera Processing and Sherlock vision software for runtime inference.

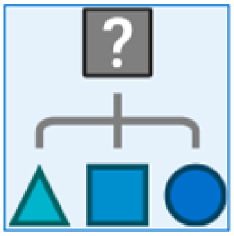
Benefits

- Train AI models quickly (within a few minutes with good data).
- Save time labelling images with automatically generated annotations.
- Decrease training effort with automatic tuning of hyperparameters.
- Assess AI models via visual tools (such as heatmaps, loss function curves, confusion matrix).
- Preserve full image data privacy by training and deploying AI models on local PC.
- Acquire live video from Teledyne and third-party cameras directly into Astrocyte.
- Run inference at a high frame rate with speed-optimized inference engine.

Deep Learning Architectures

Astrocyte supports the following deep learning architectures.

Module	Description
Anomaly Detection 	**Short Description** A binary classifier (good/bad) trained on "good" images only. **Typical Usage** Use in defect inspection where simply finding defects is sufficient (no need to classify defects). Useful on imbalanced datasets where many "good" images and a few "bad" images are available. Does not require manual graphical annotations, hence very practical on large datasets. **Detailed Description** Creates a model trained on "normal" or "good" images which can detect instances that deviate from this norm, identifying anomalies or outliers compared to the training set. It is a binary classifier (yes/no, either/or) such that images presented to the model are labelled as "normal" or "anomaly". When performing inference operations, it can generate heatmaps to assess what image details activated the model neurons; this information can help determine the validity of predictions (that is, if the image features that prompted the determination are what is expected).

Classification 	**Short Description** A generic classifier to identify the class of an image. **Typical Usage** Use in applications where multiple class identification is required. For example, it can be used to identify several classes of defects in industrial inspection. It can train in the field using continual learning. **Detailed Description** Creates a model trained on a dataset of labelled images, with each label describing a category (class). Typically, dataset images are sorted using separate folders per class (no manual annotation required). When executing the inference operation, the model returns the class label with the highest calculated probability from among the set of classes in the training dataset. If more than one class appears in the image, the class with the highest match score is returned. In addition to the regular classifier, a Continual Learning model type is available. Using Sapera Processing, existing classes in the model can be updated or new classes added based on new image data in the field without requiring retraining in Astrocyte.
Object Detection 	**Short Description** An all-in-one localizer and classifier. Object detection finds the location of an object in an image and classifies it. **Typical Usage** Use in applications where the position of objects is important. For example, it can be used to provide the location and class of defects in industrial inspection. **Detailed Description** Training object detection requires adding bounding boxes around objects, identified with class labels, in the training image dataset. The bounding boxes and associated class labels are referred to as dataset annotations. Object detection can locate multiple objects and classify them in an image. Object detection also includes a Semi-Supervised Object Detection (SSOD) Self-Training option that automatically generates bounding boxes and corresponding labels for datasets that contain a majority of unlabeled images.
Segmentation	**Short Description** A pixel-wise classifier. Segmentation associates each image pixel with a class. Connected pixels of the same class create identifiable regions in the image. **Typical Usage** Use in applications where the size and/or shape of objects are

	required. For example, it can be used to provide location, class and shape of defects in industrial inspection. **Detailed Description** Creates a semantic segmentation model that predicts which pixels within an image are of a certain class. The dataset identifies the regions in the image that are assigned to a particular class; unlabeled regions are considered background. Typically, at inference time Blob Analysis operators analyze the segmented image to obtain logical objects from which to extract features such as area, elongation, perimeter, etc.

Astrocyte Graphical User Interface

Creating Dataset

Generating image samples

- Connect to a camera (Teledyne or 3rd party) or a frame-grabber to acquire live video
- Save images while acquiring live video stream (manually from click or automatic)

Importing image samples

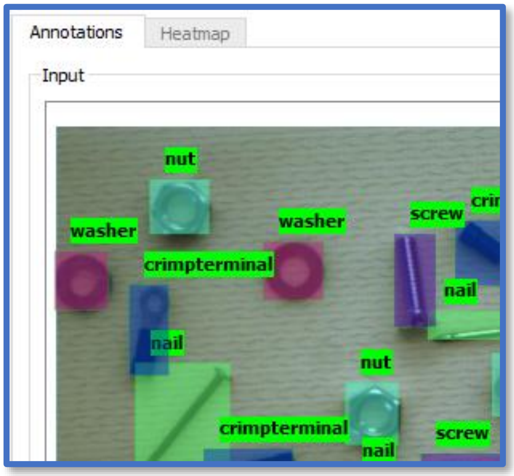
- File selection based on folder layout, prefix/suffix and regular expressions.
- Image file formats: PNG, JPG, BMP, GIF and TIFF.
- Automatic (random) or manual distribution of images into training and validation datasets.
- Adjustable image size via resizing or cropping for optimal memory usage.
- Creation of a mask via visual editing tools to mark portions of the image to be excluded.

Creating/importing annotations (ground truth)

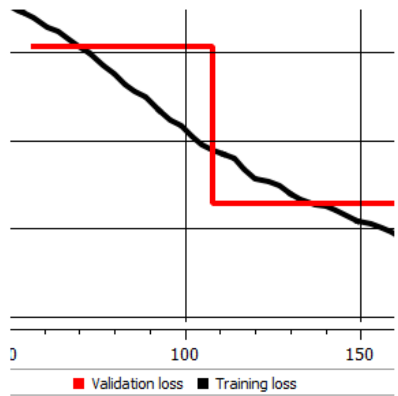
- Manually create annotations with built-in visual editing tools: rectangle, circle, polygon, brush, …
- Automatically create annotations using pre-built models.
- Automatically create annotations using Semi-Supervised Object Detection (SSOD) applied to a partially annotated dataset.
- Import annotations from user-defined text files with customizable parsing scheme.
- Import annotations from common database formats such as Pascal VOC, MS COCO and KITTI.

Visualizing/editing dataset

- Image display and zoom.
- Annotation display as overlay graphics on image.
- Annotation selection, deletion and editing.
- Manual editing of annotations on individual samples.
- Merging of dataset via saving as TAR file.



Training Model



- Training on system GPU. See minimum requirements below.
- Selection of device (when multiple devices available)
- Choice of deep learning models for optimal accuracy.
- Selection of preprocessing level: native, scaling or tiling.
- Support of rectangular input images (preserving aspect ratio)
- Automatic or manual setting of hyperparameters.
- Access to hyperparameters such as learning rate, number of epochs, batch size, etc., for customization of training execution.
- Hyperparameters pre-set with default values commonly used.
- Image augmentation available for artificially increasing the number of training samples via transformations such as rotation, warping, lighting, zoom, etc.
- Training session control: save while training and cancel if needed.
- Progress bar with training duration estimation.
- Graph display of progress including accuracy and training loss at each iteration (epoch).

Validating Model

1187	0	0	0	1189	99.83%
0	**1257**	0	1	1260	99.76%
0	0	**1153**	0	1154	99.91%
0	2	1	**1166**	1171	99.57%
1187	1263	1155	1167	**12000**	99.83%
100.00%	99.52%	99.83%	99.91%	99.83%	**99.83%**

- Statistics on model training.
- Metrics on model performance: accuracy, recall, mean average precision (mAP), intersection over union (IoU).
- Model testing interface to perform validation of the model on either training, validation, entire, or user-defined dataset with possibility of reshuffling samples.
- Display of confusion matrix (graph showing intersection between prediction and ground truth). Interactive selection of individual images.
- Display of heatmaps for visualization of hot regions in classification.

Running Inference

- Run inference on sample images for testing validity and speed inside Astrocyte.
- Select which GPU/CPU to run inference on.
- Adjust inference parameters like thresholds and options.

Exporting Model

- Proprietary model format compatible with Sapera Processing and Sherlock.
- Model contains all information required for performing inference: model architecture, trained weights, metadata such as image size and format.
- Multiple model management. Models stored in Astrocyte internal storage.
- Model can be imported into user application via Sapera Processing or Sherlock.

Integration with Sapera Processing and Sherlock

- Both Sapera Processing and Sherlock include an inference tool for each of the supported model architecture.
- Model files are imported into the inference tool and ready for execution on live video stream.
- The inference tool can be coupled with other image processing tools such as blob analysis, pattern matching, barcode reading, etc.
- To be used in conjunction with Sapera LT for acquiring images from Teledyne DALSA cameras and frame-grabbers.
- Examples available with source code.

Licensing

- Astrocyte Licensing
  - Requires either **Sapera AI SDK** or **Sherlock AI SDK** to operate.
  - If no license is present Astrocyte will run in evaluation mode for 60 days.
- Sapera Processing Licensing
  - AI inference in Sapera Processing is enabled by the **Sapera Group 4 Runtime** license.
  - If no license is present Sapera Processing will run in evaluation mode for 60 days.
- Sherlock Licensing
  - AI inference in Sherlock is enabled by the **Sherlock AI Runtime** license.
  - If no license is present Sherlock will operate with no video acquisition.

System Requirements

- Microsoft Windows 10 or 11 (64-bit)

- Intel® Processor supporting EM64T technology
- Minimum 16GB of RAM (32 GB recommended)
- An NVIDIA GPU
  - With minimum 8GB of RAM
  - With graphics driver version 516.31 or later
  - Recommendations:
    - Minimum: RTX 3070 or any other card with 8GB RAM
    - Very good: RTX 3080 or any other card with 12GB RAM
    - Best: RTX 3090 or any other card with 24GB RAM