ARL-TN-0918 ● SEP 2018

US Army Research Laboratory

Continuous Object Learning Interface Specification

by Philip Osteen, Jason Owens, Troy Kelley, Sean Mcghee, and Jonathan Milton

NOTICES

Disclaimers

ARL-TN-0918 ● SEP 2018

US Army Research Laboratory

Continuous Object Learning Interface Specification

by Philip Osteen, Jason Owens, and Troy Kelley
Vehicle Technology Directorate, ARL

Sean Mcghee
SOS International, Reston, VA

Jonathan Milton
University of Illinois Springfield, Springfield, IL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
September 2018	Technical Note	14 February–20 March 2018

4. TITLE AND SUBTITLE
Continuous Object Learning Interface Specification

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Philip Osteen, Jason Owens, Troy Kelley, Sean Mcghee, and Jonathan Milton

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-VTA
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TN-0918

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Future Army robots will be expected to operate in highly dynamic, uncertain environments, making use of incoming data to make intelligent and independent decisions. While many autonomy applications can leverage known structure in the environment and a finite (closed) set of situations that may be encountered, the singularly uncertain domain of a battlefield demands that autonomous agents are as flexible and adaptive as possible. Within the larger scope of general adaptive systems, we seek to identify capabilities that will lead to continuous object learning (COL), to continuously learn to identify object instances and entire object categories that have never been seen before (i.e., perform open-set learning). In this report, we identify a set of core component capabilities we hypothesize to be sufficient for COL. The goal of the report is to provide a specification for the interfaces to the individual capabilities that is specific enough to derive an implementation of such an architecture, yet general enough to allow flexibility in the implementation itself. We see this specification definition as a way to ensure that well-defined, testable, and repeatable experiments can be performed when developing future autonomous systems.

15. SUBJECT TERMS
continuous object learning, autonomous systems, learning architecture, interface specification, online learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Philip Osteen
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (Include area code)
Unclassified	Unclassified	Unclassified	UU	23	410-278-2057

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

Future Army robots will be expected to operate in highly dynamic, uncertain environments, making use of incoming data to make intelligent and independent decisions. While many autonomy applications can leverage known structure in the environment and a finite (closed) set of situations that may be encountered, the singularly uncertain domain of a battlefield demands that autonomous agents are as flexible and adaptive as possible. Within the larger scope of general adaptive systems, we seek to identify capabilities that will lead to continuous object learning (COL), to continuously learn to identify object instances and entire object categories that have never been seen before (i.e., perform open-set learning).

To this end, we have identified a set of core component capabilities we hypothesize to be sufficient for COL. The components are outlined in Fig. 1, which shows a conceptual pipeline that could yield a COL system. Note that while we have defined data flow and direction in the figure, we expect that this may change and that bidirectional component-wise data flow may eventually be required (for example, using the output of an object detection algorithm to improve object tracking performance).

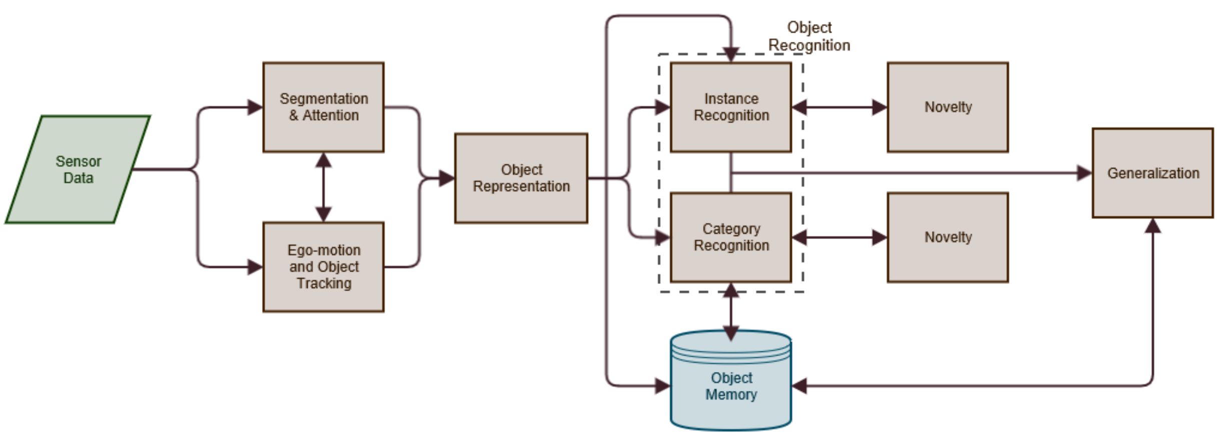

Fig. 1 We hypothesize that, given known sensor input and an object memory mechanism, a COL system requires processes to 1) track ego- and object-motion, 2) segment potential objects of interest, 3) convert raw input to rich, flexible representations, 4) recognize previously observed instances and categories, 5) detect novel objects, and 6) generalize novel instance representations to categories over time

This document details the interface specification that will be used to test various components of the pipeline and establish baseline performance of algorithms on a benchmark data set. The interface is a critical step toward rigorously characterizing the performance of individual algorithms as well as integrated COL systems. The interface is intended to be generic with minimal assumptions about the environment or any particular algorithm's implementation details. We briefly describe each component of the interface, as well as the interface specification. The US Army Research Laboratory's (ARL) Symbolic and Subsymbolic Robotics Intelligence Control System (SS-RICS) architecture, which will be migrated to integrate with this interface specification, is outlined in Appendix A. The mapping to this COL interface specification, which describes the integration path, is defined in Appendix B.

2. Intermediate Type Definitions

For the following interface, the intermediate data types that are not self-evident are defined. For those items needing a unique identification, the combination of tag + description must be unique. In some cases, it may be easier to use only the tag or description as a unique ID. Tables 1 through 7 provide the intermediate data type details.

Table 1 Header

Field	Type	Description
Stamp	Uint64	Time stamp
Frame_id	String	Coordinate frame name

Table 2 Boundary

Field	Type	Description
Header	Header	…
Type	Constant	One of: {BOUNDING_BOX, POLYGON, POLYTOPE}
Points	List of vec3	xyz boundary points (z = 0 for 2-D boundary)
Faces	List of list of indices	Each list of indices corresponds to the points from pts for one face

Table 3 Tracked object

Field	Type	Description
Boundary	Boundary	Object boundary
Tag	Positive integer	Unique track ID (or other number-based tag)
Description	String	Unique track ID (or other description)

Table 4 Object segment

Field	Type	Description
Boundary	Boundary	Object boundary
Tag	Positive integer	Unique segment ID (or other number-based tag)
Description	String	Unique segment ID (can be assigned to an existing track ID) or other description
Confidence	Double	Normalized measure of how likely it is this segment is an object
In-motion?	Yes, no, unknown (ternary logic)	Whether this segment is moving
Deformable?	Yes, no, unknown	Whether this segment is deformable

Table 5 Novelty

Field	Type	Description
Novelty	Probability/confidence	Information signal provided as output by the recognition algorithms

Table 6 Instances

Field	Type	Description
Instance_id	String	Unique instance ID
Category_id	String	Most unique known category ID
Confidence	Float	Confidence in assigning instance_id to object
Novelty	Novelty	Novelty signal

Table 7 Category

Field	Type	Description
Category_id	String	Unique category ID
Confidence	Float	Confidence in assigning category_id to object
Novelty	Novelty	Novelty signal

3. Sensor Data

We assume that our sensor input is in the form of red, green, blue, depth (RGB-D) frames, plus any information from additional sensors or algorithms that may make the tasks easier. Specifically, we identify inertial sensors as an optional but recommended additional input that will significantly aid with ego-motion estimation and subsequent processing (Table 8).

Table 8 Sensor input

Field	Type	Description
Image	Image (RGB)	Rectified color image
Depth	Image (16-bit mono)	Rectified depth image
Inertial measurement unit (IMU)	Orientation (quaternion) + covariance(mat3 × 3) Angular velocity (vector3) + covariance(mat3 × 3) Linear acceleration (vector3) + covariance(mat3 × 3)	Optional, recommended

4. Ego-motion

Ego-motion estimation, or the ability to estimate one's own motion, is important for an embodied agent (whether animal or robot) to know where it is and how it may interact with the environment. It facilitates detecting dynamic objects from static background, as well as building an internal map (fine or coarse) of one's environment. Within the interface, the ego-motion component is responsible for maintaining the six degrees-of-freedom pose of the sensor in the local environment.

4.1 Input

In addition to the raw sensor data, the ego-motion interface provides optional but recommended input from a dynamic-object tracking component, which may be used to reduce the negative impact dynamic objects in the environment have on ego-motion estimates.

4.2 Output

Table 9 provides a list of the output types.

Table 9 Ego-motion output

Field	Type	Description
Header	Header	…
Position	Vector3	The position of the sensor relative to the local environment coordinate frame (does not necessarily have to be a globally consistent pose)
Orientation	Quaternion	The (x,y,z,w) orientation of the sensor relative to the local environment coordinate frame
Gravity	Vector3	Direction of gravity (if known, should be opposite z-up); zero vector otherwise
Velocity	Vector3	Estimated velocity
Angular velocity	Vector3	Estimated angular velocity

5. Tracking

Object tracking complements ego-motion awareness by estimating the independent motion of objects in the environment. Together, tracking and ego-motion provide continuity to our perception and actions that can enable us as humans to build models of the world and the objects within it. From a more utilitarian standpoint, tracking algorithms allow other processes to either incorporate (e.g., instance recognition) or ignore (e.g., ego-motion) dynamic data for improved performance.

5.1 Input

In addition to the raw sensor data, the tracking interface supports optional but recommended ego-motion estimates to facilitate distinguishing static background from moving objects.

5.2 Output

Table 10 provides the output types.

Table 10 Tracking output

Field	Type	Description
Header	Header	…
Tracked objects	List of tracked objects	Set of tracked objects

6. Segmentation

While motion is an excellent cue for segmenting objects from the background, objects of interest may also be stationary. Clearly, *of interest* is context-dependent, and implies that segmentation may be in the eye of the beholder (as opposed to, for example, a concrete ego-motion ground-truth value). While this is true, we expect an agent to produce segments that largely agree with those produced by humans.

6.1 Input

In addition to raw sensor data, the segmentation interface supports optional but recommended inputs from both ego-motion and tracking. Ego-motion may be used to associate current background data with prior data, while tracking effectively supplies a direct object segmentation.

6.2 Output

Table 11 provides the output types.

Table 11 Segmentation output

Field	Type	Description
Header	Header	…
Header	List of object segments	Set of object segments

7. Representation

Perhaps the most critical component to the success of any learning system is the representation, derived from raw sensor data, that is provided as input to learning algorithms. Unfortunately, the value of a representation is also one of the most difficult to quantify; its impact is intrinsically tied to the algorithm using it. Therefore, the best way to assess the relative value of a representation is through the relative performance of a given learning algorithm. Also, while ideally we wish to employ learning algorithms that are general enough to support any type of representation (e.g., hierarchical vs. flat, variable vs. fixed length, and so on), in practice, the algorithms using a representation must know its form ahead of time. Subsequently, while we strive to maintain a general interface definition, subsequent algorithms must know and support the representation format for a given system.

7.1 Input

In addition to raw sensor data, the representation interface requires input from the segmentation component.

7.2 Output

Table 12 provides the output types.

Table 12 Representation output

Field	Type	Description
Header	Header	…
Repr_type	String	Representation type specification
Repr	List of floats	Representation

8. Novelty

Instead of defining a single standalone novelty component tasked with determining whether the object identified was a new/old instance or new/old category based on the system's training and experience, we allow recognition algorithms to determine how to produce a novelty *signal* as part of its output. This decision does not, however, prevent one from composing a more-or-less standalone novelty algorithm for use with a recognition algorithm in order to provide the novelty signal.

9. Instance Recognition

The goal of instance recognition is to recognize a *physical object* that has been seen before and apply the appropriate object-specific label. It is not straightforward to determine whether an object that has gone out of view and then returned is the same exact object or instead a similar-looking object (even humans can be fooled here); therefore, the label confidence must be adjusted accordingly. The recognition of a previously seen object implies that there is also a given category label for the object, even if the category is novel itself. In such a case, the category ID will be a non-semantic but still unique label that applies to that particular physical object and perhaps other similar-looking instances.

9.1 Input

The instance recognition interface requires input from the representation component.

9.2 Output

Table 13 provides the output types.

Table 13 Instance recognition output

Field	Type	Description
Header	Header	…
Instance	List of instances	Set of instances

10. Category Recognition

In contrast to instance recognition, category recognition is not interested in whether any specific object has been seen before, but rather what *type* of object is being seen. As with instance recognition, in an open-set learning framework, there is a likelihood that eventually an entirely new type of object will be encountered. This object must be integrated into existing object models and learned to enable the adaptive intelligence we seek.

10.1 Input

The category recognition interface requires input from the representation and instance recognition components.

10.2 Output

Table 14 provides the output types.

Table 14 Category recognition output

Field	Type	Description
Objects	List of categories	Category ID of presumably new instance. **If low confidence, leave category unassigned**
Novelty	…	See above

11. Generalization

Generalization is a key part of incorporating new data into existing concepts and object relationships. In the general case of what we call *deep generalization*, objects exist with type hierarchy (often an ontology) that ideally can be updated as a result of new inferences. In the more narrow case of what we call *shallow generalization*, a collection of N instances of N unknown categories can be determined to be similar enough (based on representation) to be combined to form N instances of one still-unknown category.

11.1 Input

The generalization interface requires input from the representation, instance recognition, and category recognition components.

11.2 Output

Table 15 provides the output types.

Table 15 Generalization output

Field	Type	Description
Updates	Map of category_ids {*old_id : new_id*}	For all unique unassigned instances, determine if generalization can/should occur. If so, update a persistent map of category_ids, such that any future instances that were previously labeled with *old_id* will now be assigned *new_id*.

12. Conclusion

In this report, we have identified a set of core component capabilities we hypothesize to be sufficient for COL, as well as the conceptual interface to combine these capabilities into a single architecture. The goal of the report is to provide a specification for the interfaces to the individual capabilities that is specific enough to derive an implementation of such an architecture, yet general enough to allow flexibility in the implementation itself. We see this specification definition as a way to ensure that well-defined, testable, and repeatable experiments can be performed when developing future autonomous systems. As more experiments are performed and autonomy becomes better understood, details of this specification are bound to change; still, rigorous progress in the field of fully autonomous learning systems demands that specifications such as this are developed and maintained.

Appendix A. Symbolic and Subsymbolic Robotics Intelligence Control System (SS-RICS) Architecture Specification

Table A-1 SS-RICS architecture specification

Item	Type	Description
Input images		
Instances	IplImage	RGB (3-channel, 24-bit packed pixel) regions of interest (ROIs) obtained from motion detection from a video stream using a stereo camera.
Depth	IplImage	Signed/unsigned integer (16-bit pixel) corresponding ROIs obtained from the source image's accompanying disparity image.
Intermediate categories		
Resized images	IplImages	Instances and depth image/data files resized to a uniform shape to facilitate correlation.
Categories	IplImages	Folders of images and YAML (containing corresponding depth data) files as sorted after resizing (unsupervised learning step).
Vector files	Hue vectors and depth vectors	Text files containing extracted hue-value vectors and depth value vectors.
Correlation tables	Confidences	Tables of categories that correlate to each other and which do not. Used to pick trainable categories and negative exemplars.
Output classifications		
Trained categories	Novelties	Trained neural nets that will provide a classification when presented a vector of IplImage pixels with hue and depth values.

Appendix B. Symbolic and Subsymbolic Robotics Intelligence Control System (SS-RICS) Mapping to Continuous Object Learning (COL) Interface Specification

Table B-1 SS-RICS mapping to COL interface specification

Object	VTD type	SS-RICS type
Header		
Centroid	Pair of Int's	Formatted x,y coordinate of center of region of interest (ROI) (Embedded in file name)
Sequence	Int	Int (unique ID embedded in filename)
Boundary	(Object delineation)	
Type	From enum (BOUNDING BOX, POLYGON, POLYTYPE)	BOUNDING_BOX
Points	List of Int's(x,y,w,h)	Ints (x,y)
Faces	List of list of indices	Vector of pixels contained within the bounding box
Tracked objects	(Object)	
Header	Header	Filename (String)
Boundary	Boundary	BOUNDING_BOX
Object segment	(Same as object)	*Assume this is referring to subsegments WITHIN the object*
Boundary	Boundary	Single segment
Tag	Int	NA
Description	String	NA
Confidences	Double	Assumed to be an object
In motion	Yes, no, unknown	NA
Deformable	Yes, no, unknown	NA
Novelty		
Novelty	Probability	Confidence level of a classification (float)
Instances		
Instance_id	Header	Unique id for snippet (string)
Category_id	Int	Category assigned after sorting (int)
Confidence	Float	Correlation with initiating category (float)
Novelty	Novelty	Delta between correlations (float)
Resize	Pair of Int's (W,H)	Added - the dimensions of the newly resized snippets

Table B-1 SS-RICS mapping to COL interface specification (continued)

Object	VTD type	SS-RICS type
Category		
Category_id	Int	Category after sort (int)
Confidence	Float	Delta between correlations (float)
Novelty	Novelty	Confidence level of a classification (float)
Trained category		
Neural Net		Collection of category classifications
Image		
IplImage		Image as formatted by openCV after capture
Video data	Video sequence	A sequence of 24-bit pixel, 3-channel RGB IplImages obtained from live video stereo camera
Depth data	Video sequence	A sequence of 16-bit bit unsigned integer 1-channel disparity IplImages from live video stereo camera
Vector files		
Hue vectors		Hue value (converted from RGB) pixel data obtained from sorted snippets
Depth vectors		Depth value pixel data obtained from associated YAML files

List of Symbols, Abbreviations, and Acronyms

2-D	two-dimensional
ARL	US Army Research Laboratory
COL	continuous object learning
ID	identification
RGB-D	red, green, blue, depth
ROI	region of interest
SS-RICS	Symbolic and Subsymbolic Robotics Intelligence Control System

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIR ARL IMAL HRA RECORDS MGMT RDRL DCL TECH LIB
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
5 (PDF)	ARL RDRL VTA P OSTEEN J OWENS MA FIELDS G SLIPHER RDRL HRF-D T KELLEY