Proficiency Self-Assessment without Breaking the Robot: Anomaly Detection using Assumption-Alignment Tracking from Safe Experiments

Xuan Cao[1], Jacob W. Crandall[1], Ethan Pedersen[1], Alvika Gautam[2], Michael A. Goodrich[1]

Abstract— **Proficiency self-assessment (PSA), the ability to assess how well one can carry out a task, is a desirable capability of autonomous robot systems. Prior work has proposed *assumption-alignment tracking* (AAT) for performing PSA, and has shown that it can accurately predict robot performance in real-time given a dataset obtained from both *normal* and *abnormal* training runs. Obtaining data in abnormal conditions (i.e., conditions in which the robot is not prepared to operate) is difficult and is often not possible. As a result, many realistic datasets contain very few data points for abnormal conditions, making it difficult to apply AAT. This paper hypothesizes that a *one-class classifier* can be built to detect anomalies using only data collected under normal conditions. Two metrics, *difference* and *separation*, are proposed and used to demonstrate that AAT feature vectors from different running conditions tend to form distinct clusters that are identifiable by mainstream *one-class classification* algorithms. Thus, one-class classifiers trained on AAT feature vectors from normal data can detect anomalous conditions. Furthermore, preliminary results suggest that a few abnormal data points, if available, can be used to classify the abnormality type and, in turn, the degree to which the anomalies will likely impact robot performance. Empirical results from both a simulated navigation robot and a Sawyer robot manipulating blocks show the efficacy of the approach.**

I. Introduction

Proficiency self-assessment (PSA) has been defined as "*the ability to detect or predict success (or failure) towards a goal in a particular environment given an agent's sensors, computational reasoning resources, and effectors*" [1]. PSA is a desirable property for autonomous robot systems, especially for task-critical robots [2] or robots in a team [3], [4].

Prior work [5] introduced a novel PSA method called *assumption-alignment tracking* (AAT). The basic idea of AAT is that a robot's proficiency is sensitive to how well its decision-making algorithms (or generators) align with the environment, its hardware, and the task, which can be determined by tracking the veracity of assumptions upon which the robot's generators rely. The veracity of these assumptions is tested by generator-specific functions called *assumption checkers* based on the robot's observations (from sensory data). Feature vectors are created from veracity assessments using exponential blending. A kNN (k-nearest neighbor) model is then used to predict robot performance in real-time based on AAT data. The kNN model relies on AAT data obtained from both normal conditions (the robot's assumptions are generally met) and abnormal conditions (one or more of the robot's assumptions are not met).

Obtaining *abnormal data* for a real robot is costly and risky. Indeed, for some robot systems, only normal AAT data can safely be obtained, coupled perhaps with a few carefully obtained abnormal data points. This paper asks: *Can AAT still be useful when designed using normal data points coupled with just a few abnormal data points?*

This paper hypothesizes that a *one-class classifier* that detects anomalous operating conditions can be built based on AAT data obtained under normal conditions. Two metrics are proposed. The *difference* metric measures how different operating conditions cause different subsets of assumptions to be violated at different frequencies, while the *separation* metric measures how feature vectors for different operating conditions (normal and abnormal) form distinct clusters that are generally distinguishable from each other. Clusters for normal conditions are identifiable by mainstream one-class classification algorithms. As such, one-class classifiers, built on AAT normal data, can identify anomalies that occur after the robot has started its task. Importantly, an ensemble of assumption checkers performs more robustly for anomaly detection than any individual assumption checker, because checker output can be noisy and some assumptions can be violated sometimes even under normal conditions. Preliminary results further allow us to hypothesize that, when available, a few abnormal data points can be used to classify the abnormality type and, in turn, the degree to which the anomalies will likely impact robot performance.

This paper makes three contributions. First, two metrics are developed that effectively characterize how well AAT assumption checkers evaluate the alignment between the robot's generators and the environment, its hardware, and tasks. Second, one-class classifiers built from AAT data from normal conditions alone are shown to effectively detect anomalies that impact robot performance. Finally, results reveal opportunities to utilize a few abnormal data points, when available, to further classify anomalies and to estimate their impact on robot performance. The contributions are demonstrated using data from (1) a simulated robot navigating a maze and (2) a Sawyer robot [6] manipulating blocks.

II. Related Work

Machine self-confidence, which is a robot's "self-trust in its functional abilities to accomplish assigned tasks", is similar to PSA [7]. Several metrics have been proposed to measure machine self-confidence [8]–[13]. An integrated system of assessing self-confidence, known as *Factorized*

*This work was supported by the U.S. Office of Naval Research (grants N00014-18-1-2503 and N00014-16-1-3025). The authors thank Tim Whiting for collecting the Sawyer robot dataset used in this work.

[1]Computer Science Department, Brigham Young University, Provo, UT, USA. Contact: caoxuan8872@gmail.com

[2]Department of Mechanical Engineering, Texas A&M University, College Station, TX, USA. alvikag@tamu.edu

Machine Self-confidence, characterizes the overall confidence of an autonomous system using factors that score different parts of the system's decision-making process [7], [14]–[17].

PSA also relates to robotic introspection, introduced as the ability for a robot to differentiate between normal and abnormal modes of operation [18]. Introspection has sometimes been viewed as the capability of a model to adjust confidence in its output depending on how representative the training data are of the test case [2], [19], [20]. Daftry *et al.* [21] and Kuhn *et al.* [22] build introspective models for failure prediction in vision systems and for autonomous driving, respectively. Other ad hoc approaches to PSA have also been proposed (e.g., [23]–[25]).

Prior work formalizes robot fault/anomaly detection as a traditional binary or multi-class classification problem (e.g., [26]–[29]). In contrast to these approaches, which require sufficient training data for both normal and abnormal conditions, we formalize robot fault/anomaly detection as a one-class classification problem where data from only the normal condition is available and needed [30]. One-class classification is important for robots since behavior under abnormal conditions can be difficult to obtain.

III. Methodology and Formalism

A. Assumption-Alignment Tracking (AAT)

In AAT [5], system designers identify the assumptions made in the design of the robot's decision-making algorithms (i.e., generators). Generator-specific functions, or *assumption checkers*, are created to track the veracity of these assumptions over time. Feature vectors are constructed from veracity assessments using exponential blending. Veracity assessments and feature vectors are recorded at each time step.

Formally, suppose there are M assumptions on which the robot's behavior-generating and sensor algorithms rely. Suppose further that each assumption has a checker algorithm, implemented by the robot designer, which evaluates the veracity of the assumption. Let $\mathbf{v}(t) = [v_1(t), v_2(t), \ldots, v_M(t)]$ denote the *veracity assessment vector* of the M checkers at time t, and let $\mathbf{f}(t)$ denote a *feature vector* created from $\mathbf{v}(t)$ using exponential smoothing,

$$\mathbf{f}(t) = \lambda \mathbf{f}(t-1) + (1-\lambda)\mathbf{v}(t),$$

where λ (subjectively set to 0.3) determines how much $\mathbf{v}(t)$ and $\mathbf{f}(t-1)$ contribute to $\mathbf{f}(t)$, and $\mathbf{f}(-1) = \mathbf{0}$. The time series of assessments $\mathbf{v}(t)$ and features $\mathbf{f}(t)$ generated in a single trial estimate how well the robot's decision-making algorithms align with its environment and hardware systems in the trial. These data are used to assess robot proficiency.

B. Data Collection in Various Robot Running Conditions

AAT data consists of time series of veracity assessments and feature vectors collected under different conditions. Conditions are denoted by $\{\mathbb{C}_0, \mathbb{C}_1, ..., \mathbb{C}_N\}$ where $\mathbb{C}_0$ represents the normal condition and all others are various abnormal conditions. For each condition $\mathbb{C}_i \in \{\mathbb{C}_0, \mathbb{C}_1, ..., \mathbb{C}_N\}$, let $\mathbf{V}_i = \{\mathbf{v} : \mathbf{v} \text{ is labeled as } \mathbb{C}_i\}$ and $\mathbf{F}_i = \{\mathbf{f} : \mathbf{f} \text{ is labeled as } \mathbb{C}_i\}$ denote the veracity assessment vector set and feature vector set for that condition, respectively.

C. Two Metrics for Characterizing AAT Data

Ideally, different running conditions produce (1) different patterns in the veracity assessment vectors and (2) distinct clusters in the feature vector space. We propose two metrics to characterize AAT data: *difference* and *separation*.

For each running condition $\mathbb{C}_i \in \{\mathbb{C}_0, \mathbb{C}_1, ..., \mathbb{C}_N\}$, let α_{ij} denote the *violation rate* for the jth assumption. The violation rate and *violation rate vector* are defined as

$$\alpha_{ij} = \frac{\#\text{violations for the } j\text{th assumption in } \mathbf{V}_i}{|\mathbf{V}_i|} \quad (1)$$

$$\boldsymbol{\alpha}_i = [\alpha_{i1}, ..., \alpha_{iM}] \quad (2)$$

where $|\mathbf{V}_i|$ is the total number of veracity assessment vectors contained in $\mathbf{V}_i$. The *difference* between condition $\mathbb{C}_i$ and $\mathbb{C}_j$, denoted by $\delta_D(\mathbf{V}_i, \mathbf{V}_j)$, is an aggregate measure of dissimilarity and is defined using the $L1$ distance as

$$\delta_D(\mathbf{V}_i, \mathbf{V}_j) = \|\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j\|_1 = \sum_{n=1}^{M} |\alpha_{in} - \alpha_{jn}|. \quad (3)$$

The *separation* metric is inspired by the Silhouette score [31] and evaluates how far apart sets of feature vectors are from each other. Formally, let $\mathbf{F}_i$ and $\mathbf{F}_j$ denote two feature vector sets created from $\mathbb{C}_i$ and $\mathbb{C}_j$. Define $\overline{D}(\mathbf{F}_i, \mathbf{F}_j)$ as the mean of distances between any feature vector $\mathbf{f}_i \in \mathbf{F}_i$ and $\mathbf{f}_j \in \mathbf{F}_j$, yielding

$$\overline{D}(\mathbf{F}_i, \mathbf{F}_j) = \frac{\sum_{\mathbf{f}_i \in \mathbf{F}_i} \sum_{\mathbf{f}_j \in \mathbf{F}_j} D(\mathbf{f}_i, \mathbf{f}_j)}{|\mathbf{F}_i| * |\mathbf{F}_j|}, \quad (4)$$

where $|\mathbf{F}_i|$, $|\mathbf{F}_j|$ are the total number of feature vectors contained in $\mathbf{F}_i$ and $\mathbf{F}_j$, and where $D(\mathbf{f}_i, \mathbf{f}_j)$ is a vector-based distance operator. The *separation* metric between $\mathbf{F}_i$, $\mathbf{F}_j$, denoted by $\delta_S(\mathbf{F}_i, \mathbf{F}_j)$, is given by

$$\delta_S(\mathbf{F}_i, \mathbf{F}_j) = \frac{\overline{D}(\mathbf{F}_i, \mathbf{F}_j)}{2\overline{D}(\mathbf{F}_i, \mathbf{F}_i)} + \frac{\overline{D}(\mathbf{F}_i, \mathbf{F}_j)}{2\overline{D}(\mathbf{F}_j, \mathbf{F}_j)}, \quad (5)$$

Euclidean distance and cosine distance are used for the operator D in Eq. (4), and the corresponding separation metrics are denoted by δ_S^E and δ_S^c, respectively.

D. One-class Classification Algorithms

One-class classifier models are trained on a specific class of data. When feature vectors created in different running conditions form distinct clusters, as discussed in Section III-C, a one-class classifier that is trained on one specific feature vector set can distinguish that feature vector set from others.

Various one-class classification algorithms can be trained on some $\mathbf{F}_i \in \{\mathbf{F}_1, ..., \mathbf{F}_N\}$ and then used to classify a test vector as $\mathbb{C}_i$ or $\neg\mathbb{C}_i$. Four algorithms are considered: CH (convex hull) [32], one-class SVM (Support Vector Machine) [33], DBSCAN (density-based spatial clustering of applications with noise) [34], and deep SVDD (Support Vector Data Description) [35]. Each algorithm builds a subspace (e.g. n-sphere or n-polytope) in the feature vector space that contains most of the training data.

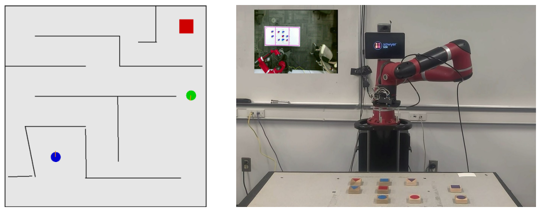

(a) (b)

Fig. 1: (a) A simulated blue robot navigating to a red charger. (b) A Sawyer robot setting up a table with various blocks.

The classification decision rule used in this paper first computes the Euclidean distance from the test vector to the surface of the sub-space, which we call the *anomaly score*. If the anomaly score is less than a threshold, then the test vector is classified as $\mathbb{C}_i$ and otherwise as $\neg\mathbb{C}_i$. By varying the threshold, a classifier's performance can be measured using the AUC-ROC (Area Under the ROC), which is computed using true and false positive rates for all thresholds.

E. Anomaly Detection and Classification

Let $\mathbf{F}_0$ denote the feature vector set for the normal operating condition $\mathbb{C}_0$. Section V-C shows that each classifier trained on $\mathbf{F}_0$ can discriminate between $\mathbb{C}_0$ and $\neg\mathbb{C}_0$ well. Suppose that there exist feature vectors $\mathbf{F}_i \neq \mathbf{F}_0$ obtained from experiments in which a known set of assumptions are violated. Each $\mathbf{F}_i$ reflects a *type* of abnormal condition. Section V-F provides evidence it might be possible to classify the abnormality type of detected anomalies based on the proximity of test vectors to each $\mathbf{F}_i$.

IV. Experiments

AAT data are collected from a simulated robot navigating a maze and a Sawyer robot performing a manipulation task. This section describes each task domain, the assumption checkers, the data collection and processing.

A. Robot Systems

Fig 1a shows a simulated robot (blue circle) that must navigate to its charger (red square) within a certain amount of time. The black line segments and the green circle in Fig 1a are obstacles. The robot can spin in place in either direction or move forward (straight). The robot is equipped with a camera that looks down on the world from above and a sensor that detects whether or not the robot is on its charger. AAT data for the navigation task is collected in the environment shown in Fig 1a, as well as three other configurations of robot position, charger position, and obstacle positions.

Fig 1b shows a Sawyer robot that must arrange nine unique (color, shape) blocks in the center of the table in a desired order and desired positions within a certain amount of time.

B. Assumptions and Checkers

The assumption checkers for the navigation task and Sawyer robot can be roughly divided into four categories: sensor, environment, actuator and generator output. The specific checkers are listed in Table I.

C. Datasets

Veracity assessment vectors and feature vectors are collected for the *navigation task* under normal conditions. Normal conditions assume that the camera has low noise, the camera has low distortion, the robot moves straight (without bias), and the robot moves at the desired speed. Besides, data are collected under eight abnormal conditions by violating one or two of these assumptions: camera noise (N), camera distortion (D), robot bias (B), robot speed (S), camera noise-robot bias (NB), camera noise-robot speed (NS), camera distortion-robot bias (DB), and camera distortion-robot speed (DS). There are 3864 samples (veracity assessment vectors and feature vectors) under the normal condition, as well as 9232, 7509, 7285, 4157, 10575, 9321, 8412, and 6471 samples in the eight abnormal conditions, respectively.

The *Sawyer task* has the normal condition and one abnormal condition where the vision system can fail to detect and localize blocks. There are 3041 samples and 10131 samples in the normal and abnormal conditions, respectively.

An additional dataset is generated by starting the navigation robot and Sawyer robot under normal conditions and then introducing assumption violations part way through the execution. Camera noise is added to the navigation task, and the vision system is damaged in the Sawyer task. This data set is called the *change task*.

D. Data Processing

The data are processed as follows. (a) Violation rate vectors are computed using Eqs. (1)-(2) for each condition. (b) The difference metric $\delta_D(\mathbf{V}_i, \mathbf{V}_j)$ for each pair of conditions is computed using Eq. (3). (c) The separation metrics δ_S^E and δ_S^C for each pair of feature vector sets are computed using Eq. (5). (d) For each condition, each of the four one-class classifiers described in Section III-D is trained on 75% of the feature vector set for that condition, then tested by the 25% hold-out set as well as by each feature vector set for every other condition. Hyper-parameter tuning was performed and only the best AUC-ROC score is reported. Appendix I describes the hyper-parameters and other implementation details. (e) The anomaly score is computed for the data from the *change task* by computing the Euclidean distance between the feature vector at each time and the decision boundary of the CH classifier trained on normal data. (f) A single feature vector is sampled from the N, S, and NB navigation feature vector sets, as well as one from the damaged vision Sawyer feature vector set. The Euclidean distance is computed between these sampled vectors and the feature vectors from the change task condition.

V. Results and Discussions

A. Violation Rates

Fig. 2 shows violation rate vectors for three conditions (normal, N, and NB) in the navigation task. These vectors are typical of other results. All but one assumption is rarely violated for the normal condition. The `ApproachingGoal`

TABLE I: Alignment checkers for the navigation task and Sawyer robot.

Navigation Task	
Checker type	Checkers
Sensor	camera is updated
	camera sees expected colors
	camera is with expected resolution
	camera is with low distortion
	camera is with low noise[1]
Environment	there is one charger
	there is one robot
	obstacles are visible
	charger is visible
	robot is visible
	charger has expected size
	open space has uniform cost
	robot is with expected size
	there is a path to charger
	world is stationary
Acuator	robot spins at expected speed
	robot moves straight when going foward
	robot moves at expected speed
	actuators are engaged
Output	map of world is consistent and accurate
	selected path is desirable
	wheel movements are in expected manner
	robot approaches charger

Sawyer Robot	
Checker type	Checkers
Sensor	camera is updated
	camera sees expected brightness
	camera sees expected colors
Environment	table is flat
	blocks have sufficient spacing
	table is rectangle
	world is stationary
	blocks are visible
	table is initially visible
	table is currently visible
	there are no duplicate blocks
	there are no missing blocks
	there are no foreign objects
	table is in expected state
Acuator	robot's arm moves at expected speed
	robot gripper works properly
	robot is not in error state
	no collision is detected
	robot state is available
	joint moves as expected
Output	map of table is consistent and accurate
	planner output is as expected
	controller output is as expected

[1]This checker compares how much the obstacle map has changed over the last n images taken of environment.

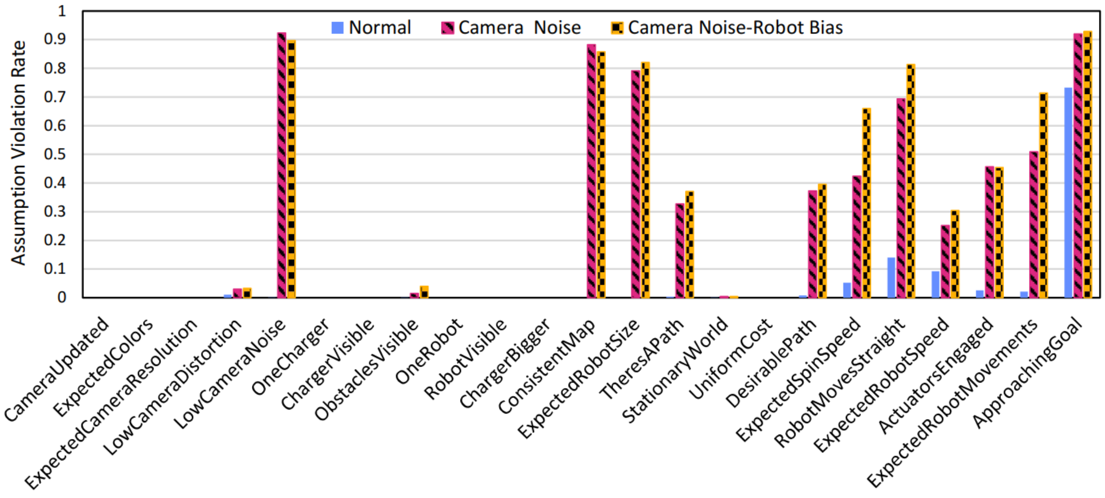


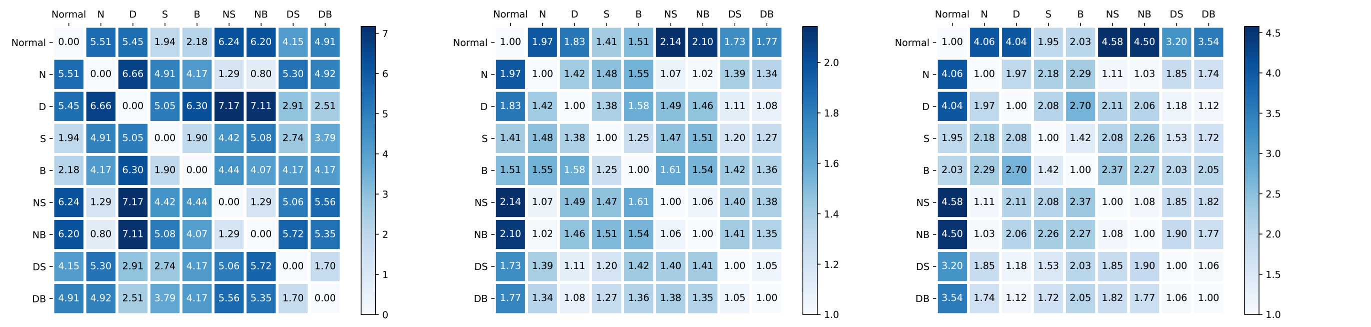
Fig. 2: Assumption violation rates for three running conditions of the simulated navigation task.

assumption, which encodes an assumption that the robot consistently progresses toward the goal, is violated in the normal condition because not all paths are ideal even for successful tasks. By contrast, eleven assumptions are frequently violated for the N and NB conditions, which significantly differs from the normal condition. Since both N and NB have camera noise, their violation rates are similar. However, the NB condition more often violates bias-sensitive assumptions: `ExpectedSpinSpeed`, `RobotMovesStraight`, and `ExpectedRobotMovements`. The veracity assessment vectors from different conditions exhibit unique patterns, supporting the hypothesis that AAT yields distinct clusters.

B. *Difference and Separation Metrics*

Fig. 3a shows the difference metric between all running conditions for the navigation task. Each abnormal condition differs from the normal condition. Additionally, each abnormal condition differs from every other. Importantly, any pair of different conditions that share an abnormal factor (e.g. N-NB) show smaller δ_D than pairs that do not share common factors. For the Sawyer robot, δ_D between the normal and vision condition is 3.14, roughly in the middle of the values for the navigation task. These results suggest that AAT generates veracity assessment vectors that tend to be far apart from each other if they come from different conditions.

Figs. 3b–3c show that patterns for δ_S^E and δ_S^c are very similar to those of δ_D for the navigation task. For the Sawyer robot, δ_S^E and δ_S^c between the normal and vision condition are 1.20 and 1.23, respectively. These results suggest that AAT feature vectors form distinct clusters under different running conditions. Furthermore, the separation between two clusters is sensitive to the overlap between the abnormal conditions.



(a) δ_D (b) δ_S^E (c) δ_S^c

Fig. 3: δ_D, δ_S^E and δ_S^c between any pair of running conditions for the navigation task.

TABLE II: AUC-ROC scores for the one-class classifiers for different training conditions. Each classifier is trained on 75% of the data from the training condition and tested on both the remaining 25% as well as the vectors from every other condition. Bold numbers indicate the best algorithm for that condition.

Classification Algorithm	**Navigation Task**									**Sawyer Robot**	
	Normal	N	D	S	B	NS	NB	DS	DB	Normal	Vision
CH	0.959	0.839	0.908	**0.934**	**0.953**	**0.881**	0.810	0.875	0.822	0.928	0.809
One-class SVM	0.960	**0.844**	0.895	0.932	0.941	0.865	**0.834**	0.878	0.802	0.893	**0.892**
DBSCAN	**0.968**	0.791	**0.913**	0.930	0.919	0.873	0.812	**0.889**	**0.837**	0.919	0.706
Deep SVDD	0.963	0.781	0.903	0.929	0.943	0.837	0.758	0.875	0.804	**0.932**	0.705

C. *AUC-ROC Score*

The AUC-ROC scores for all one-class classifiers are summarized in Table II. Each classifier has high accuracy, and each condition has a classifier that performs very well. Since one-class classifiers are capable of identifying a cluster for each condition that differentiates between vectors in the class and those outside the class, the AAT assumption checkers appear to yield distinct clusters.

D. *Detecting the Onset of an Anomaly*

Fig. 4a demonstrates how the CH-based classifier trained on the normal feature vector set works in real-time for anomaly detection for the navigation task. During the first 60 seconds, the robot is running in the normal condition and the classifier keeps producing low anomaly scores indicating that the feature vector is within or very near the decision boundary. When noise is added to the robot's camera, the feature vector jumps farther away from the decision boundary. Fig. 4b shows a similar pattern for the Sawyer robot. These results, which are consistent across the datasets, suggest that a one-class classifier trained on only normal data can detect when an abnormal condition occurs.

E. *Discussion*

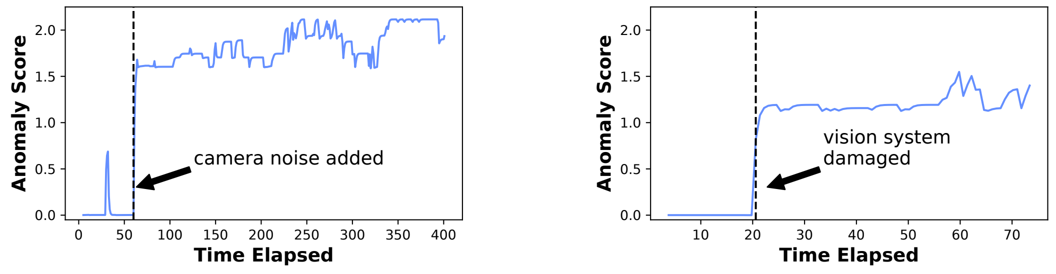
Each of the results above supports the assertion that properly designed assumption checkers yield data that form distinct clusters between various conditions. This is important because feature vectors for normal conditions are in a different cluster than vectors from abnormal conditions, making it possible to train a one-class classifier using only normal data. The one-class classifier is able to detect abnormal conditions, suggesting that it is possible to use data obtained on safe conditions to create a classifier that detects anomalies.

F. *Toward Anomaly Classification and Evaluation*

Ideally, small amounts of data from abnormal conditions could be obtained and used (1) to determine the type of anomaly occurring and (2) to assess the proficiency loss in terms of the negative impact on performance. There is reason for confidence that this might be possible.

Figure 5 shows the Euclidean distance between the feature vector time series in the *change task* data set and a randomly chosen vector from the N, S, and NB abnormal condition feature vector set. The onset of abnormal condition is indicated. The data is noisy but shows that the distance between the vector selected from the N and NB data sets and the run-time data tends to get small when the camera noise is introduced, but no such decrease is observed for a vector from the S data set. Combined with Fig. 2, there is reason to believe that the assumption violation patterns between different conditions are unique enough that a small amount of abnormal data could be sufficient to allow the type of anomaly to be identified. Thus, a lot of safe training data and a little bit of costly training data might be sufficient to both detect and classify anomalies.

The ability to determine the anomaly type is important because the type of anomaly has a strong effect on a



(a) navigation task (b) Sawyer robot

Fig. 4: Euclidean distance between the decision boundary for the CH classifier trained on normal data and the feature vector in a specific trial. The condition changes from normal to abnormal in the middle of the task. The specific changes are noted.

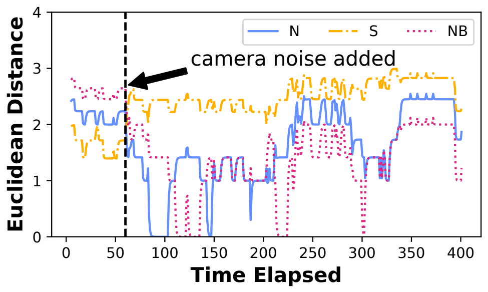


Fig. 5: Euclidean distance between feature vectors of a random sample of abnormal data and run-time robot data.

TABLE III: The average task completion time in each condition for both robot systems.

Running Condition	Average Task Completion Time (s)	Running Condition	Average Task Completion Time (s)
Normal-Navigation	96	NB	300
N	254	DS	168
D	191	DB	215
S	105	Normal-Sawyer	121
B	184	Vision	215
NS	253		

robot's ability to perform a task. Sometimes, a task can be completed even when assumptions are violated. Table III shows how different anomaly types correlate with changes in task completion time. The table indicates that noisy or broken sensors can have a substantially worse impact on performance than some other types of anomalies.

VI. Summary and Future Work

This paper demonstrates a simple yet crucial property of AAT: it produces data that effectively captures the alignment between the robot's generators and the environment, its hardware and tasks, using two proposed metrics, *difference* and *separation*. Feature vectors from different running conditions tend to form distinct clusters that are identifiable by one-class classification algorithms. Therefore, in the case where only sufficient normal data is available, a one-class classifier can be trained solely on feature vectors for the normal condition to accurately detect robot anomalies. Preliminary results also suggest that, if a small amount of abnormal training data is available for various forms of anomalies, AAT could also be used to determine the types of anomalies as well as how these anomalies might impact robot performance.

Future work should look deeper into the relation between robot anomaly and proficiency in the context of AAT. Future work should also explore more systematical ways of classifying anomaly types using a small amount of abnormal data. Another interesting direction of future work is combining robot anomaly detection and robot performance estimation [5] together to provide more comprehensive information for PSA.

TABLE IV: Hyper-parameter(s) tuned for each one-class classification algorithm for each robot system.

Algorithm	**Hyper-parameter and Tuning Range**
CH	inflation factor that controls the size of CH: {-0.05, 0, 0.05}

One-class SVM	gamma: {2e-10, 2e-9, 2e-8, 2e-7, 2e-6, 2e-5, 2e-4, 2e-3, 2e-2, 2e-1, 'scale'}
	nu for navigation task: {0.001, 0.005, 0.01, 0.025, 0.05, 0.1}
	nu for Sawyer robot: {0.001, 0.005, 0.01, 0.025, 0.05, 0.1, 0.125, 0.15, 0.175, 0.2}

DBSCAN	eps for navigation task: {0.01, 0.025, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5}
	eps for Sawyer robot: {0.05, 0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0}
	min_samples: {3, 5, 7, 10, 13, 16, 19, 22, 24, 27, 30, 33, 36, 40, 43, 46}

Deep SVDD	nu: {0.01, 0.05, 0.1}
	representation dimension: {16, 32, 64, 128}
	k: {5, 10, 15, 20}

Appendix I
Classifier Implementation Details

All classifiers tested in this paper are implemented using Python. CH classifiers are implemented based on the algorithm described in [32]. Classifiers based on one-class SVM and DBSCAN are implemented using `scikit-learn` [36].

Deep SVDD classifiers are implemented using `TensorFlow` [37]. For the sake of simplicity, we used the same network architecture the original authors used in [35] when evaluating their network on the MNIST dataset. Note that, in order to use convolutional layers, we converted the feature vectors into image data via a Grammian angular summation field; see [38] for more details. For training, we used the Adam optimizer [39] with a learning rate of 0.001 and trained for 50 epochs.

The tuned hyper-parameters for each algorithm are summarized in Table IV.

References

[1] A. Gautam, J. W. Crandall, and M. A. Goodrich, “Self-assessment of proficiency of intelligent systems: Challenges and opportunities,” in *International Conference on Applied Human Factors and Ergonomics*, pp. 108–113, Springer, 2020.

[2] H. Grimmett, R. Paul, R. Triebel, and I. Posner, “Knowing when we don’t know: Introspective classification for mission-critical decision making,” in *2013 IEEE International Conference on Robotics and Automation*, pp. 4531–4538, IEEE, 2013.

[3] D. Schreckenghost, T. Fong, T. Milam, E. Pacis, and H. Utz, “Real-time assessment of robot performance during remote exploration operations,” in *2009 IEEE Aerospace conference*, pp. 1–13, IEEE, 2009.

[4] A. Ramesh, M. Chiou, and R. Stolkin, “Robot vitals and robot health: An intuitive approach to quantifying and communicating predicted robot performance degradation in human-robot teams,” in *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 303–307, 2021.

[5] A. Gautam, T. Whiting, X. Cao, M. A. Goodrich, and J. W. Crandall, “A method for designing autonomous robots that know their limits,” in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 121–127, IEEE, 2022.

[6] “Rethink robotics.” https://www.rethinkrobotics.com/sawyer, 2019.

[7] B. W. Israelsen and N. Ahmed, “A factor-based framework for decision-making competency self-assessment.” Appears in Proceedings of the AAAI SSS-22 Symposium “Closing the Assessment Loop: Communicating Proficiency and Intent in Human-Robot Teaming”, 2022.

[8] A. R. Hutchins, M. Cummings, M. Draper, and T. Hughes, “Representing autonomous systems’ self-confidence through competency boundaries,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 59, pp. 279–283, SAGE Publications Sage CA: Los Angeles, CA, 2015.

[9] U. Kuter and C. Miller, “Computational mechanisms to support reporting of self confidence of automated/autonomous systems,” in *2015 AAAI Fall Symposium Series*, 2015.

[10] N. Sweet, N. R. Ahmed, U. Kuter, and C. Miller, “Towards self-confidence in autonomous systems,” in *AIAA Infotech@ Aerospace*, p. 1651, 2016.

[11] A. Zagorecki, M. Kozniewski, and M. Druzdzel, “An approximation of surprise index as a measure of confidence,” in *2015 AAAI Fall Symposium Series*, 2015.

[12] J. Habbema, “Models for diagnosis and detection of combinations of diseases,” *Decision Making and Medical Care*, pp. 399–411, 1976.

[13] K. N. Kaipa, A. S. Kankanhalli-Nagendra, and S. K. Gupta, “Toward estimating task execution confidence for robotic bin-picking applications,” in *2015 AAAI Fall Symposium Series*, 2015.

[14] M. Aitken, “Assured human-autonomy interaction through machine self-confidence,” Master’s thesis, University of Colorado at Boulder, 2016.

[15] M. Aitken, N. Ahmed, D. Lawrence, B. Argrow, and E. Frew, “Assurances and machine self-confidence for enhanced trust in autonomous systems,” in *RSS 2016 Workshop on Social Trust in Autonomous Systems*, 2016.

[16] B. Israelsen, N. Ahmed, E. Frew, D. Lawrence, and B. Argrow, “Machine self-confidence in autonomous systems via meta-analysis of decision processes,” in *International Conference on Applied Human Factors and Ergonomics*, pp. 213–223, Springer, 2019.

[17] B. W. Israelsen, *Algorithmic Assurances and Self-Assessment of Competency Boundaries in Autonomous Systems*. PhD thesis, University of Colorado at Boulder, 2019.

[18] A. C. Morris, *Robotic introspection for exploration and mapping of subterranean environments*. PhD thesis, 2007.

[19] H. Grimmett, R. Triebel, R. Paul, and I. Posner, “Introspective classification for robot perception,” *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 743–762, 2016.

[20] R. Triebel, H. Grimmett, R. Paul, and I. Posner, “Driven learning for driving: How introspection improves semantic mapping,” in *Robotics Research*, pp. 449–465, Springer, 2016.

[21] S. Daftry, S. Zeng, J. A. Bagnell, and M. Hebert, “Introspective perception: Learning to predict failures in vision systems,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1743–1750, IEEE, 2016.

[22] C. B. Kuhn, M. Hofbauer, G. Petrovic, and E. Steinbach, “Introspective black box failure prediction for autonomous driving,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1907–1913, IEEE, 2020.

[23] A. Dutta, P. Dasgupta, J. Baca, and C. Nelson, “Towards autonomously predicting and learning a robot’s efficiency in performing tasks,” in *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 3, pp. 92–95, IEEE, 2013.

[24] G. J. Burghouts, A. Huizing, and M. A. Neerincx, “Robotic self-assessment of competence,” in *Proceedings of the HRI 2020 Workshop on Assessing, Explaining, and Conveying Robot Proficiency for Human-Robot Teaming)*, 2020.

[25] T. Frasca, E. Krause, R. Thielstrom, and M. Scheutz, “‘‘can you do this?” self-assessment dialogues with autonomous robots before, during, and after a mission,” in *Proceedings of the HRI 2020 Workshop on Assessing, Explaining, and Conveying Robot Proficiency for Human-Robot Teaming)*, 2020.

[26] P. Goel, G. Dedeoglu, S. I. Roumeliotis, and G. S. Sukhatme, “Fault detection and identification in a mobile robot using multiple model estimation and neural network,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 3, pp. 2302–2309, IEEE, 2000.

[27] A. L. Christensen, R. O’Grady, M. Birattari, and M. Dorigo, “Fault detection in autonomous robots based on fault injection and learning,” *Autonomous Robots*, vol. 24, no. 1, pp. 49–67, 2008.

[28] E. Khalastchi, M. Kalech, and L. Rokach, “A hybrid approach for fault detection in autonomous physical agents,” tech. rep., BEN-GURION UNIV OF THE NEGEV BEERSHEBA (ISRAEL), 2014.

[29] E. Khalastchi, M. Kalech, and L. Rokach, “A hybrid approach for improving unsupervised fault detection for robotic systems,” *Expert Systems with Applications*, vol. 81, pp. 372–383, 2017.

[30] M. Zeng, Y. Yang, S. Luo, and J. Cheng, “One-class classification based on the convex hull for bearing fault detection,” *Mechanical Systems and Signal Processing*, vol. 81, pp. 274–293, 2016.

[31] P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[32] X. Zhou and Y. Shi, “Nearest neighbor convex hull classification method for face recognition,” in *International Conference on Computational Science*, pp. 570–577, Springer, 2009.

[33] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[34] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, “A density-based algorithm for discovering clusters in large spatial databases with noise.,” in *kdd*, vol. 96, pp. 226–231, 1996.

[35] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, “Deep one-class classification,” in *International conference on machine learning*, pp. 4393–4402, PMLR, 2018.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. Software available from tensorflow.org.

[38] Z. Wang and T. Oates, “Imaging time-series to improve classification and imputation,” in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[39] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.