Discrete N-Dimensional Entropy of Behavior: DNDEB

Michael Young [1,3], Mahdieh Nejati Javaremi [1,3], and Brenna D. Argall [1,2,3]

Abstract— **Shared control for human-robot teams—where both the human and the robot's autonomy provide commands to the hardware—offers advantages over fully teleoperated or fully autonomous systems by utilizing the unique skill sets of both the human and robot's autonomy simultaneously. However, the mechanism by which control is shared is often static and many teams could benefit from adjusting this mechanism, such that the human or autonomy alternatively receive more control authority in different scenarios. The question then is: how do we know when these scenarios occur? In this paper, we present a method to estimate the performance of human-robot teams using a novel metric called Discrete N-Dimensional Entropy of Behavior (DNDEB). DNDEB utilizes knowledge of a high-performing human-robot team to build a model of how the team should operate. The model is used to predict the human's command. The error between the prediction and actual command is tracked and after a certain number of samples, entropy is estimated. A higher level of entropy corresponds to deviations from the high-performance model, which can be interpreted as poor performance by the human-robot team (e.g., long task time or a collision). Our formulation offers several advantages: it (1) accepts discrete inputs of any size, (2) does not require additional sensors, and (3) is tunable to the specific application. To validate this, we conduct a 15-person study where subjects operated a powered wheelchair under three different shared-control paradigms. We find that entropy is higher for cases with longer task durations and cases where there is a collision. Moreover, we use DNDEB thresholds as a mechanism to predict the performance of the human-robot team online and find an average accuracy of 91% with a prescience rate of 72%.**

I. Introduction

Human-robot teams often take advantage of shared-control paradigms where the human and robot both work together to complete the same task. In shared control, the robotic hardware is equipped with autonomy to intelligently offload some of the control from the user. Often the aim in sharing control is to increase the safety of the user while capitalizing on benefits of a human-in-the-loop. The advantages of this collaborative scheme over either using solely a human or robotic system can be significant.

These teams, however, often perform suboptimally when their relationship is static and the human interacts with the robot in the same manner throughout the mission. For example, the robot autonomy may underperform due to environmental factors, sensor failure, or computational issues, to name a few. In such scenarios, the team may benefit by allocating more control to the human who could override faulty autonomy behavior. Conversely, human operators may become distracted, fatigued, or have varying skills when interacting with the robot. Now the autonomy could increase its authority to protect itself from harm. Thus, it can be beneficial to dynamically adjust the shared-control paradigm.

While examples of shifting the shared-control paradigm do exist in today's technologies, most focus on adding autonomous assistance when the user is performing the task unsafely (e.g., lane assist technology in automobiles). However, little work investigates automatically adjusting the shared-control paradigm due to inadequacies in the *autonomy's* performance—instead, often the human is relied upon to detect the inadequacy and take over. Moreover, autonomous shifts between discrete formulations of shared-control is understudied within the literature. In order to autonomously switch between paradigms, it is necessary to have a reliable metric to assess real-time performance of the human-robot team as a whole. Ideally, this metric would not require the addition of costly sensors and would be adaptable to a variety of human-robot team combinations.

To fill this gap, we present a novel approach and metric to predict the performance of human-robot teams using only the human operator's input to the system. The metric stems from behavioral entropy, a metric developed to estimate automobile driver distraction by looking strictly at the angle of the steering wheel [1]. Our intent is to assess team performance when operator input also might be discrete or higher than a single dimension. Towards this end, we have reformulated behavioral entropy to include multi-dimensional and discrete signals while improving the prediction mechanisms using state-of-the-art techniques from machine learning. We also present an approach to utilize this metric for human-robot teams and for dynamically shifting control-sharing paradigms. Our contribution is threefold:

1) We introduce a discrete N-dimensional variant of behavioral entropy, named DNDEB.
2) We develop an environment-aware machine learning algorithm to predict the user's command.
3) We experimentally validate our approach with 15 human subjects and three shared-control paradigms.

We first discuss the research that motivated this work in Section II. Section III covers the formulation of DNDEB and examples for intuitive understanding. Next, we present the experimental validation of the algorithm in Section V and the results of validation in Section VI. Lastly, we discuss the implications of the results for human-robot teams and dynamic shifting of autonomy in Section VII.

[1]Department of Mechanical Engineering, Northwestern University, Evanston, IL, USA
[2]Department of Computer Science, Northwestern University, Evanston, IL, USA
[3]Shirley Ryan AbilityLab, Chicago, IL, USA

II. Related Work

Shared control balances the benefits of both a human and robot partner, and multiple factors play a role in choosing the correct shared-control paradigm for a given situation [2]. Fortunately, many control sharing strategies exist from which to choose, each with their own set of advantages [3], and altering an existing paradigm or shifting between different paradigms in real-time can leverage the advantages of multiple paradigms [4]. To adjust parameters within a single control sharing paradigm, some works show success using metrics such as distance to obstacles [5] or a notion of trust-based performance [6]. However to the best of our knowledge, no works use a metric to shift between multiple shared-control paradigms.

A dynamic autonomy allocation framework could utilize a plethora of metrics to shift between control sharing paradigms. Human workload, for example, is an important measure of operator performance and state in many fields, including human-computer interaction (HCI), human-robot interaction (HRI), and human factors research. In the HCI domain, secondary tasks have been used to infer the operators' spare mental capacity and therefore their mental workload [7]. However, the secondary task itself affects the workload of the human operator and can impede the human's capacity to perform the primary task at hand. The need for non-intrusive and online measures of workload has led to a large body of work looking at using varying psychophysiological sensors (including, but not limited to, heart rate variability, electrodermal activity, eye movements, and skin temperature) [8]. For this type of workload measurement, additional sensors need to be attached to the operator. Although technological advances have given way to wireless sensors that are mostly non-intrusive, these sensors are still limited by battery power, communication channel, bandwidth, latency, and jitter [9]. Performance is another metric of use but it typically requires concept, situation, and task awareness by the human [10], or alternatively is estimated from implicit cues, such as trust or disagreement between the human and the autonomy [11].

Removing the requirement of additional sensors or situational awareness, Nakayama *et al.* [1] develop behavioral entropy as a metric to detect (post-hoc) increased levels of workload for the driver of an automobile. Their formulation utilizes a 1-dimensional continuous input over a small window to indicate whether a driver is distracted. More specifically, they use a prediction of steering wheel angle and calculate the entropy of the prediction errors over a window of time. A higher value compared to an undistracted baseline indicates distraction or increased workload. Later work further validates the method on a larger dataset with different forms of workload [12]. Behavioral entropy is validated further [13] within the human-robot interaction (HRI) domain using 1-dimensional joystick angle (not magnitude), with positive results. An extension to behavioral entropy uses an autoregressive model to predict steering wheel angle [14], and explores other hyperparameters such as sampling frequency, model parameters, and entropy estimation parameters, to show that fine tuning can increase prediction accuracy. There are many scenarios, however, where the input from the user is not continuous or 1-dimensional. For example, tetraplegic powered wheelchair users typically operate a non-proportional (discrete) 1- or 2-dimensional interface, such as a series of buttons, rather than a 1-d continuous joystick. To employ behavioral entropy in such domains, a reformulation is thus required.

The aforementioned works lay the necessary foundation for the development of a discrete and higher-dimensional variant of behavioral entropy and speak to the potential utility of behavioral entropy within expanded domains. In this paper, we contribute Discrete N-Dimensional Entropy of Behavior (DNDEB),[1] a method able to provide behavioral entropy predictions within discrete N-dimensional spaces. We further contribute a state-of-the-art prediction mechanism and an evaluation within a compelling use-case domain: powered wheelchair operation using a limited control interface.

III. Discrete N-Dimensional Entropy of Behavior

As discussed in Section II, behavioral entropy is a metric with demonstrated effectiveness in detecting human workload by monitoring 1-D, continuous-valued control signals from the human input [1], [12], [13]. However, there are many situations where the interface a human uses to interact with a robot or system is discrete and/or multi-dimensional. In this paper, we fill this gap with the introduction of Discrete N-Dimensional Entropy of Behavior (DNDEB).

At a high level, we calculate DNDEB using a window of the human's inputs and predict the next input at time t. The prediction mechanism is designed such that it accurately predicts the human's inputs when the team is performing well. We then compute the error in the prediction and store it. After a certain number of predictions, we estimate the entropy of the distribution of the stored errors. Since the prediction is accurate for high performance, a higher entropy value indicates poorer performance of the predictor which correlates with poor performance of the team [13]. For intuition, we can analogize DNDEB to entropy in information theory, where it represents uncertainty in a signal. In the case of DNDEB, this is the uncertainty in the prediction mechanism. The mechanism is trained to accurately predict the user's command when performing the task well. The idea is that the prediction mechanism will be less accurate at predicting the command when the user is performing poorly, for example due to distraction or high workload. Higher uncertainty in the prediction mechanism thus suggests the team to be operating in a suboptimal state.

Now looking at the specific details of our DNDEB formulation, we start with the discrete inputs and how to characterize them. The human provides binary input $\mathbf{u}$ within any number $(1, .., N)$ of N control dimensions at a time, depending on the limitations of the control interface in use.

[1]We call the metric entropy of behavior instead of behavioral entropy to highlight the fact that our work does not reformulate Shannon information entropy, but rather applies entropy to behavior cues to assess performance.

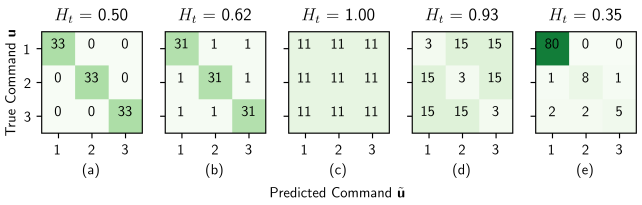


Fig. 1. Example DNDEB values (H_t) for confusion matrices of a 3-dimensional discrete input signal (e.g., *forward*, *left*, and *right*). In all 5 examples, $N = 3$, $w_e = 99$. Entropy values increase as the inaccuracy of the prediction increases (a $\rightarrow$ c), with one exception: if the predictor guesses wrong more often than chance (d). In a-d, the commands are equally likely to appear (33.3% likely). However, if the one command is more probable than the others and the model is still accurate (e), the entropy will decrease below 0.5 which could lead to false positive flags of poor performance.

Mathematically, the input is an N-dimensional vector where the element corresponding to the active input is one and other elements are zero. Simultaneous (multidimensional) control inputs are represented as additional unique DNDEB vector dimensions. That is, if the potential inputs are *left, right,* and *forward*, it is possible to provide inputs in one dimension (*left, right, forward*) or two dimensions (*left-forward, right-forward*). This results in three unique inputs ($N = 3$) for the 1-D case and five unique inputs ($N = 5$) for the 2-D case.

To calculate the error of the prediction $\tilde{\mathbf{u}}$ (covered in Section IV), we create a $N \times N$ confusion matrix $\mathbf{C}$ and initialize all elements to zero. We then update $\mathbf{C}$ using a window w_e of predicted $\tilde{\mathbf{u}}$ and true user commands $\mathbf{u}$, where w_e is a tunable parameter set based on the domain. Each prediction is recorded by incrementing a counter in cell $\mathbf{C}[i, j]$, where i is the index of the dimension of the provided control $\mathbf{u}_t$, and j is the index of the dimension of the predicted control $\tilde{\mathbf{u}}_t$. If $\mathbf{u}_t = \tilde{\mathbf{u}}_t$ (a correct prediction), then $i = j$ which corresponds to the diagonal elements of $\mathbf{C}$. Similarly, the non-diagonal elements of a row indicate the count of false positives for each of the other possible commands. This confusion matrix encodes how well the discrete predictor has performed over the window. If the elements of the matrix are large in the non-diagonal components, the prediction is poor. Once predictions are aggregated over the window, we estimate entropy H using the following formula:

$$H_t(\mathbf{C_t}) = -\sum_{i=1}^{N}\sum_{j=1}^{N} P(c_{ij}) \log_b P(c_{ij}) \tag{1}$$

where c_{ij} are elements of C and $P(c_{ij})$ is the probability of c_{ij} that we estimate as $P(c_{ij}) = c_{ij}/w_e$.[2] A summary of the calculation is presented in Algorithm 1, where Equation 1 corresponds to Line 7.

Figure 1 presents examples of multiple confusion matrices $\mathbf{C}$ at time t and each is labeled with the instantaneous DNDEB value. In addition to demonstrating how entropy as a function of prediction accuracy, these examples also highlight the necessity to monitor the accuracy of the predictor to correctly interpret the resulting entropy value.

[2]Equation 1 could be rewritten as a single summation to N^2 if the matrix C is flattened to a $[N^2 \times 1]$ vector. This formulation more closely resembles the tradition entropy formula.

Algorithm 1 Discrete N-D Entropy of Behavior

1: **Given:** prediction window w_p, estimation window w_e, and number of inputs N
2: $\mathbf{C} \leftarrow \mathbf{0}[N \times N]$
3: **while** running **do**
4: $\quad \mathbf{u}_t \leftarrow$ user command $\quad \triangleright$ receive the user command
5: $\quad \tilde{\mathbf{u}}_t$ = predictCommand($\mathbf{u}_{t-w_p}, ..., \mathbf{u}_{t-1}$)
6: $\quad \mathbf{C}_t$ = getConfusionMatrix($\mathbf{u}_{t-w_e+1}, \tilde{\mathbf{u}}_{t-w_e+1}, ..., \mathbf{u}_t, \tilde{\mathbf{u}}_t$)
7: $\quad H_t$ = estimateEntropy($\mathbf{C}_t$)

In practice, the DNDEB is calculated over multiple windows and the mean value is used as the metric. The entropy calculation windows w_e can be overlapping or independent of the rate of commands/predictions, depending on the implementation domain requirements.

IV. Prediction Mechanism

As discussed in Section III, DNDEB requires a mechanism to accurately predict the human operator's command input. Here we contribute a prediction mechanism that employs only the user input in its base formulation, but which also is able to additionally incorporate sensor information, if available. The sensor input is simply concatenated to the end of the N-dimensional DNDEB input vector. Sensor information can prove useful for interfaces with discrete inputs because the user's command is often sparse and might not encode much information about the next command, and the sensor readings can contain information about which commands may be valid. For example, if the sensor detects an obstacle in front of the human-robot team, a forward command should not be predicted.

The output of the predictor is a vector of zeros except for the index corresponding the predicted active control input is equal to one. A discrete classifier therefore is better suited to this prediction than a Taylor-series prediction or regression, as used in prior works [14]. For the predictor itself, we choose to leverage neural networks (NN) and long short-term memory networks (LSTM) due to extensive success in temporal predictions and robotics [15].

The general architecture of our network is shown in Figure 2. This model utilizes a LSTM to handle the discrete

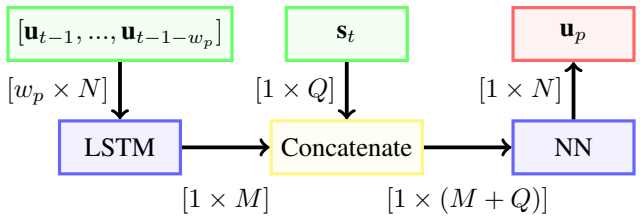
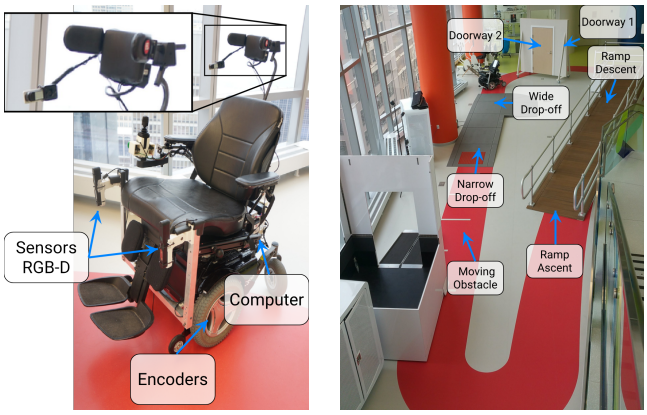


Fig. 2. Neural network architecture. Inputs are green, neural network layers are blue, operators are yellow, and the output is red. Size of the LSTM output is M, sensor data is Q ($Q = 0$ if no sensor data is available), DNDEB input is N, prediction window is w_p.



(a) (b)

Fig. 3. Experimental set-up: (a) wheelchair with electronic head array system called out (b) task track with 7 tasks.

time-series data (our human inputs **u**) and a fully connected feedforward NN to incorporate the sensor information **s**. LSTMs are shown to perform well in tasks where the inputs are discrete vectors with a similar output [16]. Moreover, researchers have found success in combining LSTMs with other information, such as sensor or camera information, to increase prediction accuracy [17]. In our implementation, the additional sensor information provided to the network is a series of distance measurements to obstacles (or lack thereof) surrounding the human-robot team.

V. Experimental Design

The domain of assistive robotics is particularly well-suited to validate DNDEB because human inputs are often discrete, and the degree to which a user's motor impairment affects their ability to control their device often changes based on the task, fatigue, pain, and/or the progression/digression of their impairment. Therefore, users may benefit from varying amounts of autonomy assistance, and thus shifting formulations of shared control. To that end, we test DNDEB for different shared-control paradigms on a common assistive device: a powered wheelchair.

A. Robotic Platform

We conduct the study using a Permobil wheelchair retrofitted with an on-board computer and two ASUS Xtion RGB-D sensors on each arm (Fig. 3-a).The control interface is an ASL 105 electronic head array system (ASL, TX, USA), which is configured as a 2-dimensional discrete controller. This is a commonly used interface for controlling commercial powered wheelchairs. The head array consists of three headrest pads with embedded proximity sensors, and a chin switch. The lateral pads are used for turning laterally left or right, and the back pad is used for moving longitudinally. The chin button is used to switch between whether the back pad commands indicate forward or reverse motion. The back pad and lateral pads can be triggered simultaneously, which results in a total of 9 possible commands: (1) forward, (2) backward, (3) left, (4) right, (5) forward-left, (6) backward-left, (7) forward-right, (8) backward-right, and (9) stop, if no command is being issued.

B. Shared-Control Paradigms

We hypothesize that DNDEB will correlate with the performance of human-robot teams for a variety of shared-control paradigms. Thus, we develop three different shared-control paradigms to assess the versatility of DNDEB:

- **Full teleoperation (A0):** The executed command is a direct mapping of the user input—there is no assistance from autonomy. We include this paradigm because studies show that users desire more control when they can control their device effectively [18].
- **Obstacle avoidance (A1):** The user input is blended with the input from the autonomous planner solely to avoid obstacles. We use the obstacle avoidance paradigm proposed in [19].
- **Waypoint planning (A2):** The user issues one of the 9 possible commands via the head array. The autonomy projects the direction of the command based on visibility (up to 2m) and plans a collision-free trajectory to that goal. The user may cancel the goal at any time by pressing the chin button.

C. Experimental Protocol

As a first evaluation of DNDEB, we recruited 15 (9 female and 6 male) subjects with an average age of 31 ± 9 years old. The subjects had varying degrees of experience with robotic systems and powered wheelchairs, but were all naive to powered wheelchair control with a head array interface.

The experimental protocol consisted of one session with both a training phase and testing phase. The training phase was designed for the participants to become familiar with using the head array interface and the dynamics of the wheelchair. The test phase included seven tasks: two doorway traversals, ramp ascent and descent, avoiding a dynamic obstacle, and traversing a sidewalk with drop-off from wide and narrow ends. For each shared-control paradigm (A0, A1, A2), the participants executed all seven tasks in a circuit for a total of three circuits (Fig. 3-b). The order of tasks within a circuit and the order of shared-control paradigms were randomly counterbalanced across all subjects. In total 315 tasks were performed and 105 per shared-control paradigm.

VI. Results

We begin this section by defining high and low performance using our metrics of interest. Next, we evaluate the prediction mechanism on the high-performing cases to ensure a consistent measure of entropy. Last, we evaluate DNDEB by looking at averages across trials and test its ability to predict collisions online.

For the statistical analyses, we perform non-parametric Kruskal-Wallis tests to check for significance within groups. If significance exists, we test pairwise using the non-parametric Mann-Whitney-Wilcoxin test with Bonferonni correction m. Throughout, we denote statistical significance of $p < \frac{0.05}{m}$ as *, $p < \frac{0.01}{m}$ as **, and $p < \frac{0.001}{m}$ as ***.

A. Experiment Performance

Using the number of collisions and average completion time per task, we quantify the safety and performance, respectively, of the human-robot team. This allows us to categorize tasks as high performance, low performance, and unsafe for use in the DNDEB evaluation.

We define *high performance* as task trials (1) without collisions and (2) with a task time lower than the median over all subjects. Similarly, we define *low performance* as task trials (1) without collisions and (2) with task times higher than the median. Finally, we designate trials as *unsafe* if a collision occurred. The counts of trials in each quality category are provided in Table I. The distribution of data is relatively equal with slightly more samples in the *high-performance* classification. This is due to the the fact that trials with task times above the median also see a collision more often than those with task times below the median.

B. Discrete Predictor Analysis

Before evaluating DNDEB, we need to determine the accuracy of the prediction mechanism and ensure the primary weight of the confusion matrix remains on the diagonal elements. In our implementation, we choose a prediction window of 50 samples ($w_p = 50$) as input to the LSTM. For sensor input $\mathbf{s}$ we utilize 10 ($Q = 10$) distance readings equally spaced between 0-180^o, centered on the wheelchair facing forward, and ranging from 0-4 meters. We extract these distance readings from the costmap populated by the RGB-D sensors. The distance measurement input is directly concatenated with the output of the LSTM (a vector of size 32) which is then fed into 3 fully connected neural network layers of size 64 (Fig. 2). We train each model for 50 epochs with a binary cross entropy loss function. Finally, the result is the predicted user command $\tilde{\mathbf{u}}_t$.

To validate the network architecture, we perform leave-one-out cross validation (LOOCV) for each subject. That is, the network is trained using 14 subjects and we check its accuracy using the remaining subject's data. For training and assessment we only use high-performance trials, because we are building a predictor of good performance (Sec. IV) and then detecting deviations from good performance via model prediction errors. We find that the model accurately predicts the user's next command $91.7 \pm 6.7\%$ of the time. Though there is room for improvement, this range results in informative DNDEB calculations (that is, more weight in the diagonal elements of the confusion matrix, Figure 1).

TABLE I

TRIAL COUNT FOR EACH QUALITY CATEGORY BY SHARED-CONTROL PARADIGM

Paradigm	Unsafe	High Performance	Low Performance
A0	27	41	37
A1	21	46	38
A2	28	42	35

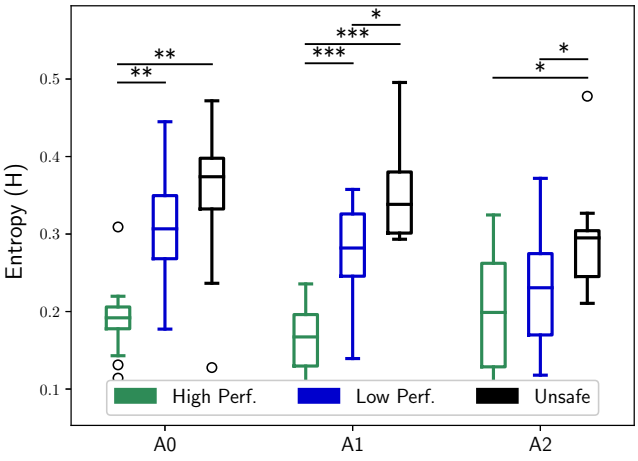


Fig. 4. The average values of all task trials on the one-left-out subject for DNDEB. For all assistance levels, the entropy value of the high performance tasks is significantly lower than the unsafe cases. Except for assistance level 2, significant differences also exist between high and low performance and between low performance and unsafe.

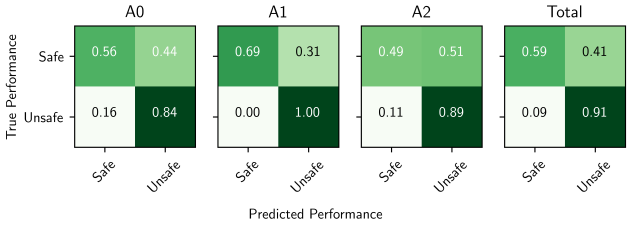
C. Discrete N-D Entropy of Behavior Evaluation

The benefits of DNDEB appear in both post-hoc and real-time analyses, and we present each in this section.

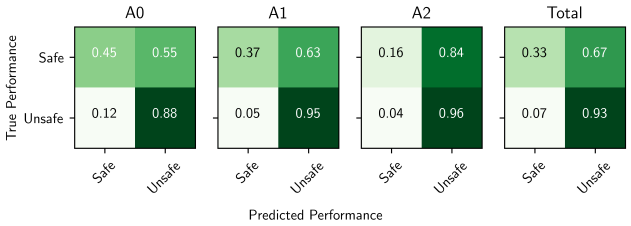
1) DNDEB Correlation with Performance: We perform leave-one-out cross validation to assess the accuracy of DNDEB as an indicator of performance. Using a model trained on 14 subjects, we calculate the DNDEB metric on predictions made on the remaining subject's data. For this preliminary analysis, we set $w_p = 50$ and $w_e = 30$ and average all the DNDEB values calculated for each window. Figure 4 shows the resulting entropy values for each label under each shared-control paradigm, averaged over all subjects. The statistically significant differences suggest that DNDEB is a useful metric in differentiating between the quality categories. Referring to the information theory analogy, the model is less uncertain when making predictions for high-performance trials—due to the fact that it was trained on data for high-performance tasks.

2) Real-time DNDEB Analysis: Having established that DNDEB correlates well with performance for our domain, we now investigate the evolution of DNDEB over time to assess its usefulness as an online indicator for shared-control allocation. This allocation could be handled in a variety of ways. For example, one possible mechanism is to allow the user to continue to operate the robot using the current shared-control paradigm while their DNDEB metric indicates high performance, and to shift shared-control paradigm or warn the user if the metric indicates any other quality category.

For an online indicator, we choose a thresholding mechanism where the DNDEB metric is monitored, and if it surpasses a set threshold, we trigger an alert. We also evaluate how the DNDEB threshold compares to minimum distance to obstacles (DTO) as an online indicator. While an intuitive metric for collisions (human-robot teams need to be close to an obstacle in order to collide with it), DTO



(a) DNDEB Thresholds



(b) Minimum DTO Thresholds

Fig. 5. Normalized confusion matrix of each prediction mechanism for the shared-control paradigms (A0, A1, A2) and all three combined (Total).

does not necessarily correlate with human operator workload or distraction, as does behavioral entropy [13]. Minimum DTO is measured by calculating the euclidean distance to all obstacles and choosing the minimum value.

To set thresholds for DNDEB, we utilize the data in Figure 4 and tune empirically. The threshold values for DNDEB are set to 0.33, 0.33, and 0.28 for shared-control paradigms A0, A1, and A2, respectively. Using a buffer of 5cm from the wheelchair footprint, we set the threshold of distance to obstacles to 52cm for all three shared-control paradigms to achieve the best prediction performance. For both DNDEB and DTO, we smooth the prediction by using a mean value over a window of 30 samples.

The results of both collision prediction mechanisms are presented in Figure 5. Combining over all three shared-control paradigms, we see the DNDEB threshold method predicts a collision correctly 91% (67 total) of the time there is a collision, missing only 9% (7 total). It is worth noting that there is a large number of false positives at 38% (91) of tasks without a collision. By comparison, the DTO threshold predicts true positives with similar accuracy (93%) but an even larger number of false positives (67%). This is likely due to the fact that many tasks (such as doorway traversal) require close proximity to obstacles.

Depending on the domain, a cautious autonomy allocation framework—one that predicts a large number of false positives—may be desirable to improve safety. Conversely, a large number of false positives might prove unacceptable to the human partner.

In order to be useful for online autonomy shifting with the aim of improved performance and increased safety, whether the prediction occurs before an actual collision (prescience) and how early the prediction takes places (timeliness) is important. The results for both metrics are presented in Table II. We find that on average both metrics provide advance detection of a collision with a more than sufficient amount of time for a safety intervention: their advance detection range (timeliness) is 32.0-63.6s, while the wheelchair itself is able to cold stop from maximum velocity in under 2s. In other experimental work, we find an average time of approximately 4s is necessary for a human operator to attain situational awareness after a shared-control paradigm shift, so even the lower bound of 32s is quite sufficient to safely perform a shift (a more elegant response than a hard stop). We do see that DTO is more successful than DNDEB (88% versus 71%, averaging the three shared-control paradigms) in predicting a collision before it occurs (prescience). This aligns with the earlier findings of DTO being a more cautious metric, with a higher number of false positives (Fig. 5-b).

VII. Discussion

The results demonstrate the effectiveness of DNDEB as a post-hoc metric to classify performance and as an online early indicator of collisions during a task. In this section, we discuss the likely reasons for the results, limitations of the work, other potential use cases, and ideas for improvement.

In many scenarios, engineering teams do not have true labels of performance or collisions for tasks performed by human-robot teams. Since DNDEB correlates well with performance labels on average (Fig. 4), researchers can use this metric to assess their shared-control paradigms post-hoc. If DNDEB increases over a long period of time, for example, the team can launch an investigation as to the causes of a likely suboptimal performance by the human-robot team.

Similarly, it is useful to know beforehand if unsafe behavior or suboptimal performance may occur. Averaging the three shared-control paradigms, DNDEB predicts online a collision before it occurs 71% of the time with sufficient time to intervene (40.4s). Though not perfect, this drastically increases the likelihood of changing operational parameters to address the situation. The reason prediction does not occur 100% of the time may be the relatively short duration of the tasks in this study. DNDEB evolves at a slower rate than other metrics such as distance to obstacles. If DNDEB is tracked over longer periods of time, it is likely that the accuracy would increase and changes be more evident. Other features calculated from the DNDEB metric, such as rate of change, also might prove useful.

As this is a first analysis of DNDEB, we see many areas for potential improvement and to deepen understanding of the metric and technique as a whole. First, the user

TABLE II
Prescience and Timeliness of Predictions for DNDEB and DTO

Metric	Paradigm	Prescience (%)	Timeliness(s)
DNDEB	A0	77	32.0
	A1	68	41.7
	A2	69	45.1
DTO	A0	90	40.1
	A1	91	57.9
	A2	85	63.6

command prediction mechanism might utilize any number of architectures and techniques. Depending on the domain, more information—such as task information, sensor data, or a user profile—might improve, or confound, the model prediction. DNDEB has a variety of hyperparameters that can be tuned by the designer (w_p, w_e, model architecture, DNDEB thresholds, smoothing window size). This work did not aim to fully optimize these parameters, but future work will investigate patterns in tuning the parameters to edge out extra gains in prediction accuracy.

The labeling of data for this work consisted of only one type of collision; however in reality, the collisions varied in severity. Some collisions had high risk of injury (driving over a drop-off edge) while others were minor such as scraping the doorway with the arm rest. Differentiating between the severity of collision would be an interesting extension.

Despite the promise of DNDEB, limitations do exist. First, the approach requires a baseline high-performance dataset to train the prediction mechanism. Depending on the complexity of the system and the environment in which it operates, large amounts of data may be required to train the model that can accurately and reliably predict a high-performance user command. Fortunately, simulation might serve as an efficient method for data collection and should be investigated further. Similarly, the rates of real-time change in DNDEB presumably vary between use cases, for example depending on characteristics of the user interface (such as sparsity). The experimental results presented here are for a specific domain and interface, and further validation is required to establish the ubiquity of this metric.

VIII. Conclusion

In this work, we presented a metric that correlates with performance for human-robot teams where the human provides discrete, N-dimensional inputs to the system called Discrete N-Dimensional Entropy of Behavior. Compared to previous formulations, it offers the ability to use limited discrete interfaces with multiple dimension inputs, and it also leverages environment-aware user command prediction. We validated the metric through a 15-person study where users operated a powered wheelchair using a limited interface and under three formulations of shared control. We also demonstrated its potential as a real-time performance detection mechanism for use in autonomy allocation.

Overall, DNDEB shows promise as a metric for classifying and predicting performance of a human-robot team. Future work will aim at addressing generalization by testing with other interfaces (sip-and-puff), other user populations (spinal cord injured), and different shared-control paradigms. Additionally, we will evaluate the effect of different hyperparameters of DNDEB on accuracy and timeliness.

ACKNOWLEDGMENT

We would like to gratefully acknowledge support by grant from U.S. Office of Naval Research under the Award Number N00014-16-1-2247. This material is also based upon work supported by the National Science Foundation under Grant No. 1552706. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] O. Nakayama, T. Futami, T. Nakamura, and E. R. Boer, "Development of a steering entropy method for evaluating driver workload," *SAE Technical Paper*, pp. 1686–1695, 1999.

[2] D. A. Abbink, T. Carlson, M. Mulder, J. C. de Winter, F. Aminravan, T. L. Gibo, and E. R. Boer, "A topology of shared control systems-finding common ground in diversity," *IEEE Transactions on Human-Machine Systems*, vol. 48, pp. 509–525, 2018.

[3] S. Musić and S. Hirche, "Control sharing in human-robot team interaction," *Annual Reviews in Control*, vol. 44, pp. 342–354, 2017.

[4] M. Chiou, R. Stolkin, G. Bieksaite, N. Hawes, K. L. Shapiro, and T. S. Harrison, "Experimental analysis of a variable autonomy framework for controlling a remotely operating mobile robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 3581–3588.

[5] Q. Li, W. Chen, and J. Wang, "Dynamic shared control for human-wheelchair cooperation," 2011, pp. 4278–4283.

[6] H. Saeidi and Y. Wang, "Trust and self-confidence based autonomy allocation for robotic systems," *Proc. IEEE Conference on Decision and Control (CDC)*, pp. 6052–6057, 2015.

[7] C. D. Wickens, *Measures of Workload, Stress and Secondary Tasks*. Springer US, 1979, pp. 79–99.

[8] R. Rajan, T. Selker, and I. Lane, "Task load estimation and mediation using psycho-physiological measures," *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pp. 48–59, 2016.

[9] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich, "Common metrics for human-robot interaction," in *Proceedings of the ACM SIGCHI/SIGART Conference on Human-robot Interaction*, 2006, pp. 33–40.

[10] A. Lampe and R. Chatila, "Performance measure for the evaluation of mobile robot autonomy," in *Proc. IEEE International Conference on Robotics and Automation*, 2006, pp. 4057–4062.

[11] S. Nikolaidis, Y. X. Zhu, D. Hsu, and S. Srinivasa, "Human-robot mutual adaptation in shared autonomy," *Proceedings of the ACM/IEEE Conference on Human-Robot Interaction*, pp. 294–302, 2017.

[12] E. R. Boer, "Behavioral entropy as an index of workload," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 2000, pp. 125–128.

[13] M. A. Goodrich, E. R. Boer, J. W. Crandall, R. W. Ricks, and M. L. Quigley, "Behavioral entropy in human-robot interaction," Defense Technical Information Center, Tech. Rep., 2004.

[14] E. R. Boer, M. E. Rakauskas, N. J. Ward, and M. A. Goodrich, "Steering entropy revisited," *Proceedings of the Third International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*, pp. 25–32, 2005.

[15] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[16] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[17] I. Kostavelis, K. Charalampous, A. Gasteratos, and J. K. Tsotsos, "Robot navigation via spatial and temporal coherent semantic maps," *Engineering Applications of Artificial Intelligence*, vol. 48, pp. 173–187, 2016.

[18] D. Gopinath, S. Jain, and B. D. Argall, "Human-in-the-loop optimization of shared autonomy in assistive robotics," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 247–254, 2017.

[19] A. Erdogan and B. D. Argall, "The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: an experimental assessment," *Robotics and Autonomous Systems*, vol. 94, pp. 282–297, 2017.