Data-Efficient Process Monitoring and Failure Detection for Robust Robotic Screwdriving

Xianyi Cheng, Zhenzhong Jia, and Matthew T. Mason *Fellow, IEEE*

***Abstract*— Screwdriving is one of the most prevalent assembly methods, yet its full automation remains challenging, especially for small screws. A critical reason is that existing techniques perform poorly in process monitoring and failure prediction. In addition, most solutions are essentially data-driven, thereby requiring lots of training data and laborious labeling. Moreover, they are not robust against varying environment conditions and suffer from generalization issues. To this end, we propose a framework that combines knowledge-based process models with a hidden Markov model. The novelty of this work is the incorporation of operation-invariant characteristics such as screwdriving mechanics and stage transition graph. The developed algorithm performs online stage and result predictions with very few training data and labeling. It can also quickly adapt to variant conditions, as validated by the robotic experiments.**

I. INTRODUCTION

Threaded fastening is one of the most commonly used assembly methods [1]. Take the consumer electronics industry for example, hundreds of billions of small screws are assembled every year. Despite the substantial research in this field, fully automatic insertion of these screws remains extremely challenging [1], [2], especially for smart-phones. There are several reasons: incomplete understanding of the underlying process [3], especially the initial mating step; tighter tolerance and higher alignment accuracy requirements for small screws [4]; very limited study on failure prediction and recovery [1], [2]. In this paper, we focus on screwdriving process monitoring and failure prediction.

People have developed many failure detection methods [1] to mitigate the problems [5] encountered in screwdriving. However, there are some limitations. First, all algorithms except [2] can only perform result classification. However, some irreversible process failures [3] cannot be detected through result classification alone. Second, failure prediction and recovery, a must-have technology for very large volume production, has rarely been studied [1]. Third, many algorithms cannot incorporate robot information for improved performance. Fourth, some algorithms are not robust against varying environment conditions. Thus, a robust screwdriving system should have the following functions: 1) real-time process monitoring; 2) failure recovery; 3) fast adaptation to new assembly needs; 4) affordable cost. This paper focuses on realizing function 1), 3) and 4); function 2), i.e., failure recovery, is left as future work.

The authors are with the Robotics Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA. Email: xianyic@cmu.edu, zhenzhong.jia@gmail.com, matt.mason@cs.cmu.edu

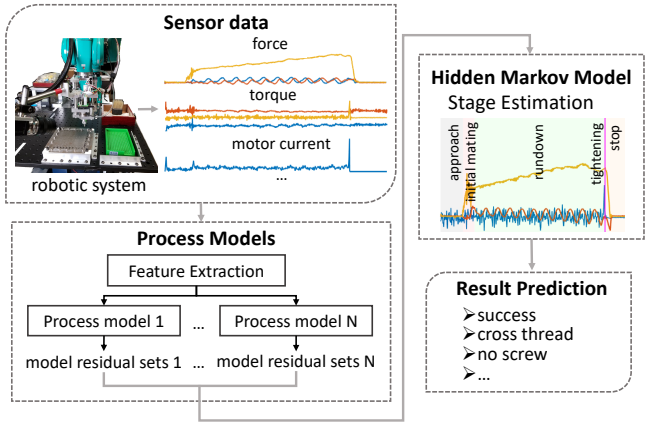

Fig. 1. An overview of our data-efficient approach.

While most of the previous work focus on result detection, fast adaptation/generalization has not been explored in the literature. In particular, many failure detection systems for intelligent screwdriving are based on data-driven approaches, including our previous work [3] and [2]. Directly learning from data yields good results. However, data-driven methods often require lots of training data and laborious labeling that requires expert knowledge [3]. These methods are not robust against environment changes (screw sizes, motor setups, force magnitude, friction, and sensor noise), thereby making generalization difficult.

To mitigate these problems, we propose a system framework shown in Fig. 1. In this system, from the collected data, we construct knowledge-based process models and feed them into a hidden Markov model (HMM), through which we can do stage estimation and later rule-based result prediction. This framework is developed based on the following fact: the mechanics (see Sec. IV-A) that dominates individual stages (e.g., *rundown*) and the structure of the stage transition graph (see Fig. 3) do not change with varying experiment conditions (e.g., screw sizes). For operation-invariants, process models (see Sec. IV-A) encapsulate local invariant characteristics (e.g., mechanics for each stage) in their constraint equations, while an HMM incorporate global invariant state transition graph in its transition model. For environment-dependents, process models treat them as identifiable model parameters, while an HMM adapts to these variations by updating its observation models during training. Under this framework, we only need very few labeled data for the environment-dependent process model parameter estimation during system initialization. As unlabeled data accumulates, the unsupervised learning HMM enables continuous update

and adaptation without human intervention. In conclusion, our system : 1) performs online stage and result prediction; 2) quickly adapts to new experiments with similar structures; 3) requires very little data labeling; 4) reduces cost.

To summarize, our contributions are:

- The first attempt to perform unsupervised learning in screwdriving, to our best knowledge.
- A framework of quick generalization for screwdriving assembly, which offers a feasible way to solve real-industrial problems.
- Significant reduction in data/labeling requirements, by taking full advantage of screwdriving mechanics.

II. RELATED WORK

A. Analysis of the Screwdriving Process

A thorough understanding of threaded fastening process is critical for reliable and accurate fault detection. A typical screwdriving process consists of several steps: *initial mating*, *rundown*, and *tightening*, according to the *torque-angle curve* [6]. Recently, [3] presents a more complete stage transition graph, covering successful cases and multiple failure modes. According to [3], a standard operation often consists of stages including *approach*, *initial thread mating*, *rundown* and *tightening*, while other stages like *hole finding* or *no screw spinning* occur when there is alignment error or pick-up failure. Each stage has different sensor signal signatures and process models. In the *initial mating* stage, contact models are studied by [7], [8] and [9]. According to [9], a force spike is an indicator of this stage. In the *rundown* stage, quasi-static analysis shows that the oscillation phenomenon is an important signature [10]. In the *tightening* stage,torque, rotation angle and torque-angle gradient [11] are commonly used for failure detection in industry. By taking full advantage of previous researches, we combine these features to develop process models in Section IV-A.

B. Fault Detection for Threaded Fastening

In industrial threaded fastening, the most common fault detection approach is the *teach method* [1]. In this method, correct torque-angle fastening signatures are first collected as reference, and then compare thme with actual signals using limit check or trend check [12]. The *teach method* is simple and easy to implement; however, it lacks flexibility and generality. To overcome these problems, intelligent threaded fastening systems are developed, most of which fall into two categories: model-based or data-driven. Model-based approaches [13] require analytic models and accurate system parameters. On the other hand, data-driven method directly learn from labeled data, such as ANNs [14], SVMs [10], GTC-DF [3], and decision trees [2]. The model-based approaches are flexible, but accurate system models are hard to obtain. The data-driven methods do not require prior knowledge, but they are difficult to adapt to variations.

C. Multi-State Learning in Manipulation

A robot task often includes a series of discrete states [15]. Identification of states is critical during task operations. This widely studied problem can be formulated by two methods, Markov model or hybrid systems. Our approach is a combination of these two methods. Hybrid models are incorporated to augment the HMM, while the HMM keeps learning and modifying hybrid model parameters.

The Markov model view this problem in a stochastic way. Most previous work focus on HMM, which assumes latent states. The HMMs are used to perform automatic action segmentation from demonstrations in [16] and [17]. In [18], which is most close to our work, contact states are predicted by combining contact model estimation with an HMM. Contact states are modeled with unspecified parameters, which are estimated from observations, while the HMM performs as a acceptance test to predict the most likely state. The major differences between our work and [18] is the way of using HMMs. In [18], the HMM is used as a probability observer. In contrast, we train the HMM to learn and refine the process models. The HMMs are also used to detect transitions after actions in [19] and [20]

In contrast, the hybrid system way is more deterministic. Hybrid system involves both continuous models and discrete states [21]. A discrete state can be identified when the states and inputs fall into the corresponding domain [22]. Hybrid systems can incorporate simplified mechanical models to approximate complex robot behaviors, as well as to reason mechanisms and to perform feedback control [23]. In [24], an unsupervised learning framework for nonlinear hybrid system is developed, where state models are learned through Gaussian process and linear classifiers are trained to specify states.

III. DATA COLLECTION

In this paper, the screwdriving data is collected by a robotic screwdriving system shown in Fig. 2. By using a "floating structure" [2] design, the forces and torques exerted on the screwdriver tip can be measured by the 6-axis force/torque (F/T) sensor. Following a procedure similar to the one outlined in [3], we collected more than 7000 experiment runs. During each run, the system records signals from the robot arm, the F/T sensor, motor current and encoder, a linear potentiometer that is used to measure the compliant spring displacement (so that we can trace the screwdriver tip trajectory), and a high-speed camera (used as ground truth). The system is an upgrade of those used in [3] [25], where more detailed descriptions can be found.

We performed experiments on different sized screws with various operation force profiles, motor velocities and alignment errors. In this paper, we will focus on the datasets of M1.4x4 screws (for training use), M1.2x3 screws and M2.5x5 screws (for generalization use).

IV. METHOD

This section describes our failure detection system framework and generalization approach in details. Our failure detection system is composed of three parts:

1) Modeling screwdriving process : develop process models from expert knowledge

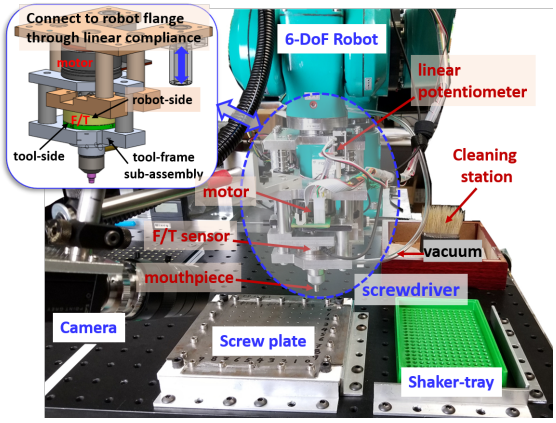

Fig. 2. Intelligent robotic screwdriving system for data collection.

2) Learning an HMM: train a unsupervised HMM on the unlabeled data
3) Prediction of result using rules: use process rules of stages to predict the result of a operation

The generalization is performed only when our well-trained system is given a new experiment data — it will quickly adapt to new experiment setups.

An overview of our system is illustrated in Fig. 1. Firstly, all the possible stages during the screwdriving process are specified and modeled as constraint equations. Secondly, the model residuals of the constraint equations are passed to a hidden Markov model, which predicts the stages during the operations. The hidden Markov model is then trained on our dataset using expectation-maximization algorithm. Then, once the stages can be predicted, rules are constructed to infer results from stages. Finally, given new data from similar experimental setups, a well-trained system is generalized to new data by parameter estimation.

A. Modeling of the Screwdriving Process

Modeling of screwdriving process includes three steps: building a stage transition graph, designing a feature set, and modeling stages as constraint equations.

First, a stage transition graph is concluded through expert knowledge of screwdriving process, see Fig. 3. The status of screwdriving process can be modeled as discrete states, which is defined as *stage* in our paper. A screwdriving operation can be seen as a sequence of stages. The stage transition graph describes all possible transition among the stages of our system. All the screwdriving operations begin at *approach*. A standard successful operation is then followed by *initial mating*, *rundown*, *tightening* and *stop*, while *hole finding* stage could appear when alignment errors exist. Different kind of failures are indicated by the corresponding stages, such as *no screw spinning* and *cross-tightening*. Their corresponding failures are *no screw* and *cross thread*. At last, all operations end at *stop*. A reasonable and well-defined graph enables us to clearly and separately model the screwdriving process, because each stage is defined to have only one dynamic model. Moreover, in Section IV-B, this graph is used as the state transition model in

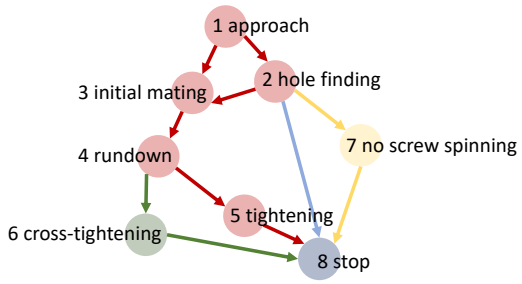

Fig. 3. The stage transition graph in the screwdriving process

the HMM. For reference, this graph is modified from our previous stage transition graph in [3]. Compared to the previous graph, stages and results related to *stripped* are eliminated by exerting appropriate insertion force during the experiments. More importantly, we add a *stop* stage as the termination mode, instead of different result types. The design of the *stop* mode is for simplification of process model and generalization of HMM. For simplification, a general termination at *stop* can be very easily modeled as in Section IV-A. For generalization, all path ending at one mode reduces the stages and parameters to learn in an HMM.

Second, a feature set is designed from the signatures of sensor signals in screwdriving stages. At time t, a set of features can be extracted inside a fix-sized window from signal data at and before t. Besides the values of the data itself, our feature set includes:

ν_x, ν_y, ν_z: the frequency of maximum amplitude of forces by discrete Fourier transform. The oscillation phenomenon of the forces on x and y axis is only found in the rundown stage of robotic screwdriving task [10]. With small alignment error, the frequency is approximately equal to thread fastener motor velocity.

d_{tip}: screwdriver tip distance to the screw hole plane, a very strong indicator of the process status. It is computed by signals from robot positions, orientations, linear potentiometer and hole location.

$\Delta_{d_{tip}}$: the change of screwdriver tip distance.

k_{tip}: the gradient of screwdriver tip distance.

k_{ta}: the torque-angle gradient. This feature is widely used in torque control threaded fastening for indicating the status of tightening, whether in elastic zone or yield zone.

k_t: the z-axis torque-time gradient.

k_f: the z-axis force-time gradient.

k_c: the current-time gradient. Our screwdriver employs a direct drive motor, thus the torque on the motor is proportional to the current in the motor. Since the torque sensing have relatively large noise, motor current is used as a redundant feature to provide more robust information.

Δ_{f_z}: the change of force on z-axis. A drop of z-axis force often indicates the alignment or mating of screw threads [9].

$\boldsymbol{\mu}_\mathbf{f}$, $\boldsymbol{\sigma}_\mathbf{f}$: the means and variances of forces (both are 3 by 1 vectors).

Third, each stage is modeled as one or multiple constraint equation sets through physics or process knowledge. The parameters of stage models can either be predefined or estimated from data. In our system, we preset the some

parameters: V_m (motor velocity), H_1 (height of screw head), and H_2 (cross thread height). Other parameters are learned from labeled data. Our stage models are constructed as follows:

approach: no contact on the screwdriver, thus zero forces are assumed:

$$\mu_{\mathbf{f}} = 0$$
$$\sigma_{\mathbf{f}} = 0$$

hole finding: the screw moves around without insertion, while screwdriver tip distance remains unchanged:

$$\Delta_{fz} = 0$$

initial mating: two possible models exist in this stage. The first model is a smooth mating, where screw is inserted smoothly with a constant velocity $C_{1k_{tip}}$ and a constant gradient of force C_{1k_f}:

$$k_f - C_{1k_f} = 0$$
$$k_{tip} - C_{1k_{tip}} = 0$$

The second model is an abrupt mating, which often follows a *hole finding* stage. This model is featured by a sudden drop of screw tip and z-axis force:

$$1/k_f = 0$$
$$1/k_{tip} = 0$$

rundown: vibration phenomenon occurs in this stage , indicating the rotation of the center axis of screw [10]. the vibration frequency can be approximated by motor velocity V_m. At the same time, the screw goes down with a constant velocity $C_{2k_{tip}}$ and a constant gradient of force C_{2k_f}:

$$v_x - V_m = 0$$
$$v_y - V_m = 0$$
$$k_f - C_{2k_f} = 0$$
$$k_{tip} - C_{2k_{tip}} = 0$$

tightening: for a successful tightening, the torque angle gradient must satisfy a preset value C_{1kta} to ensure correct tension. Since the motor current is proportional to screwdriver torque, the gradient of current should also approximately equal to a constant C_{1kc}. The distance of screwdriver tip to the screw hole surface is the thickness of screw driver head H_1:

$$k_{ta} - C_{1kta} = 0$$
$$k_c - C_{1kc} = 0$$
$$d_{tip} - H_1 = 0$$

cross tightening: has the same model with *tightening* but different model parameters ,C_{2kta}, C_{2kc} and H_2. During *cross tightening*, the torque angle gradient is much less than that in *tightening* due to angular errors. For a cross thread condition, only the first external thread is mated at the cross thread angle [7], thus the screwdriver tip distance is approximately computed as $H_2 = l \cdot cos(\theta)$, where l is the screw length and θ is the minimum cross thread angle.

no screw spinning: a vibration of z-axis force at frequency of V_m happens after contact begins, because no screw head is matched with screwdriver tip:

$$v_z - V_m = 0$$

stop: the screwdriver motor stop, thus the signal values motor current m_c and motor velocity m_v are equal to zero:

$$m_c = 0$$
$$m_v = 0$$

Given data from an operation, features are first extracted. For each stage, the residuals of its constraint equations are computed. Then these stage model residuals are passed to a hidden Markov model in Section IV-B.

B. Stage Prediction by Hidden Markov Model

In our system, a hidden Markov model is used to perform stage prediction. Given a set of observations, an HMM can be used to compute the hidden states which generate these observations. The key elements of a HMM are hidden states, initial state distributions, state transition probability distribution, and observation probability distributions [26]. In our system, the hidden states $S = \{S_1, S_2, \ldots, S_N\}$ are the stages defined in II-A, where the order is the same as in Fig. 3. The state at time t is denoted as q_t. The initial state distribution is denoted as $\pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$, where $\pi_i = P(q_t = S_i)$.

The state transition probability matrix is $A = \{a_{ij}\}$, where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$. If no edge goes from S_i to S_j in the stage transition graph, we have $a_{ij} = 0$; otherwise $a_{ij} > 0$.

The observation probability distribution $P(O_t|q_t = S_i)$ provides the probability density that the observation at time t (O_t) emitted by hidden state S_i. The $P(O_t|q_t = S_i)$, written as $b_i(O_t)$, can be represented in the form of Gaussian mixture models:

$$b_i(O_t) = \sum_{m=1}^{M} c_{im} \phi_{\mu_{im}, \Sigma_{im}}(x_{imt})$$

where M is number of models in S_i, c_{im} is the weight of model m; x_{imt} is the residuals of the constraint equations in model m computed from O_t; $\phi_{\mu,\Sigma}(x)$ represent a multivariate Gaussian function with mean μ and covariance Σ. The Gaussian model $\phi_{\mu_{im}, \Sigma_{im}}(x_{imt})$ gives the probability of O_t generated by the mth model in S_i.

The HMM is trained by EM algorithm, for which the details can be found in [26]. The initialization and constraint of HMM parameters is critical. Since the start states can only be S_1,*approach*, we have fixed $\pi = \{1,0,0,0,0,0,0,0\}$. The state transition matrix A is empirically initialized through expert knowledge. If there exists an edge from S_i to S_j, a_{ij} should be constrained as nonzero during training. As for observation probability $b_i(O_t)$, the weight c_{im} is initialized as $\frac{1}{M}$ and constrained as nonzero. The Gaussian mean μ_{im} is initialized as zeros and covariance Σ_{im} is a diagonal matrix initialized from estimation error in Section II-A. The mean and covariance are constrained so that our observation models not deviate too much from the stage models.

C. Rule-Based Result Prediction

The result of an operation can be inferred from stages. Five types of results exist in our system, *success*, *cross thread*, *no screw*, *no hole found* and *partial*. Our system checks the result at each timestep. The rules of result determination are constructed to prediction result of screwdriving operations as follows:

1) If certain failure stages are detected, such as *cross tightening* and *no screw spinning*, the system can

TABLE I

DATA COLLECTION SETTINGS

	dataset 1	*dataset 2*	*dataset 3*
screw size	M1.4x4	M1.2x3	M2.5x5
tightening current	1600 mA	1300 mA	3200 mA
motor velocity	80 rpm	96 rpm	80 rpm
insertion force	10 N	6 N	18 N
tightening force	20 N	14 N	30 N

stop in advance and predict their corresponding failure types, *cross thread* and *no screw*.

2) If the time of *hole finding* stage exceeds certain threshold, this indicates large non-correctable alignment error and potential damage of the screw plate. In this case, the system will stop and predict *no hole found*.
3) If the operation follow the red path on Fig. 3, then the system proceeds to success condition check. The condition check examines the final insertion length, tightening torque and tightening torque-angle gradient. The result is predicted as *success* if condition check is passed, or *partial* if otherwise.

D. Generalization

When the operation condition of screwdriving is changed, the previously trained HMM cannot be applied to the new experimental setups. But with process models and a new operation data sample, our system can easily generalize to new experiment conditions. The generalization is performed by parameter estimation given labeled data of one sample from new experiments. First, corresponding labeled data are used to estimate parameters of stage models in II-A. This is simply a linear least square problem. Then, the relationship of parameters among different stages is estimated from previous data, which is used to update unseen stage models. For example, in *cross tightening* and *tightening*, we often have $5C_{2kf} = C_{1kf}$. This relationship is estimated from old process models and can be used as a good initialization for the new dataset to generalize. Finally, the covariance in the observation model in HMM is adjusted by the newly estimated process models.

V. EXPERIMENTS

A. Dataset

Our experiments are performed on three datasets: *dataset 1*, *dataset 2* and *dataset 3*. Each dataset is collected by our robotic system under different operation conditions. The major difference of operation settings is shown in Table I. The *tightening current* is the motor stop treshold. The *insertion force* is the z-axis force exert on screw when screwdriving operation start. The *tightening force* is the z-axis force at the time when screw tightening is finished. Each dataset includes 200 operations. As described in Section III, each operation contains real-time sensor data, robot data and an operation video (as ground truth). In our experiments, we uniformly sample all the data at 100 Hz.

We labeled a few samples for process modeling and performance evaluation. For *dataset 1*, we labeled 10 samples covered all stages (training data) for process modeling, while we labeled 86 samples (test data) for performance evaluation. For *dataset 2* and *dataset 3*, one success sample in each dataset is labeled for generalization training, 98 samples in *dataset 2* and 140 in *dataset 3* are labeled for generalization performance evaluation.

To further evaluate the robustness of our system, during the collection of test data, random alignment errors (in the range of 40% of screw diameters) are added. On the other hand, all the training samples are collected under standard high precision operations. This means that, our system is trained on data with high precision alignment, but will be tested on the data with large alignment errors.

B. Implementation

The implementation of our system follows the four parts as described in Section IV: process modeling, HMM learning, result prediction, and generalization.

In process modeling, the predefined parameters are first manually setup according to our operation settings. Then the learnable parameters of all stage models are estimated from the 10 labeled samples from our original dataset, *dataset 1*. The parameter estimation is solved as the least square problem.

Given well estimated process models, the HMM is trained on 50 unlabeled samples from *dataset 1*. The training sample set composes of 30 *success* samples, 10 *crossthread* samples, and 10 *no screw* samples. Stages in HMM are numbered as in Fig.3. For initialization, the setup is described in Section IV-B. During training, to ensure convergence, the mean and covariance of the Gaussians are constrained so that the observation models just vary within $\pm 15\%$ of the stage models. When the change of log likelihood of observations is less than a set threshold, the system is considered as converged. Our HMM converges after 6-7 batches within a training time of less than 3 min.

The rules of result prediction are constructed as *if-else* statements. Three simple threshold checks form the success condition check.

When generalize to new settings, our system only need one labeled success sample from the new dataset. New stage model parameters are estimated from the one labeled sample. For other stages which do not appear in the labeled sample, their model parameters are infered from the relationship among the models in the old system. As the process models change, the covariances of the Gaussians in the HMM change proportionally.

C. Results and Evaluation

For system performance evaluation, three performance metrics are used to evaluate our methods, the overlap rate (OR, the proportion of correct stage prediction duration in total duration), result prediction accuracy (RAC), and *cross thread* detection rate (CDR). The *cross thread* detection rate is the true positive rate for the result type *cross thread*. In the industry, to identify *cross thread* is the one of the most important parts in failure detection for threaded fastening. We compare our system with the determined assignment

TABLE II

PERFORMANCE EVALUATION

Method	OR	RAC	CDR
our method	98.21%	96.51%	96.67% (29/30)
decision trees	97.34%	66.27%	73.33% (22/30)
determined assignment	87.82%	/	/

TABLE III

GERALIZATION EVALUATION

Dataset	OR	RAC	CDR
dataset 2	95.75%	92.86%	100% (18/18)
dataset 3	98.81%	97.14%	100% (10/10)

OR: overlap rate, RAC: result prediction accuracy,
CDR: *cross thread* detection accuracy.

baseline(each stage is assigned at a fixed time window) and the decision tree method proposed in [2]. The test results on 86 labeled samples from *dataset 1* are shown in Table II. Fig. 4 visualizes the stage prediction for a *sucess* sample and a *cross thread* sample by our method and the decision tree. As shown in Table II, our method produces reasonable stage classification and high result detection accuracy. As a comparison, although the decision tree has high overlap rate, it has poor result prediction performance. The decision tree method often fails to detection short but critical stages like 'tightening'. Moreover, decision trees method can produce unreasonable prediction as in Fig. 4(b), because it does not make predictions based on previous stage information.

As shown in Fig. 5 and Table III, with just one standard sample from the new dataset,*dataset 2* and *dataset 3*, our method can generalize with good performance. The new datasets has different sensor signal scales, noise, and oscillations. For instance, *dataset 2* has much larger signal-to-noise ratio than that of *dataset 1*. But as can be shown in Fig. 5(b), the scale of motor torque noise is very large, and the increase of torque is smooth. This pattern of *cross thread* failure has never been seen during training. Still, our system can adapt to these unseen failure samples.

Our system shows robustness in three aspects. First, our system can adapt to data with much larger errors it has never seen during training. Our system is trained on high precision operation data, but is tested with error infused data. Still, we get good test performance. Second, our system can generalize with very limited data and produce promising results on the unseen dataset. Third, our system almost successfully predict all the *cross thread* failures, which is crucial in the industrial assembly. Especially for *dataset 2* and *dataset 3*, the detection rate is 100% when our system has never seen their *cross thread* data.

Still, there are two major improvements to be made in our system. First, our system needs a good initialization to converge to the correct prediction. Otherwise, the HMM will be stuck in the local minima. For example, with bad initialization, our system can predict most *tightening* stages as *cross tightening* and reinforce this mistake during unsupervised training. Second, misjudgments occur due to similar features in different stages. Like in *no screw spinning*, the features v_x and v_y sometimes have the same values as in *rundown*, because oscillations can also happen without

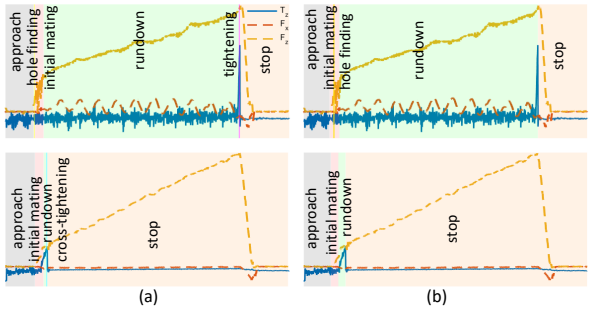

Fig. 4. The stage prediction results from: (a) our method (b) the decision tree method. The upper figures are prediction for a success operation. The lower figures are prediction for a cross thread operation.

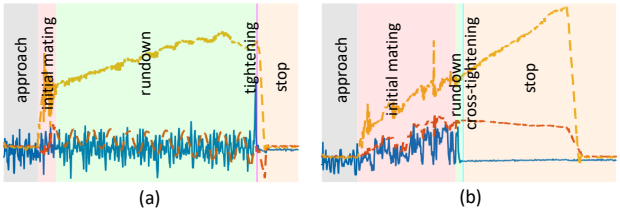

Fig. 5. Generalization to *dataset 2* with only one labeled sample. (a) is the stage prediction for a success operation, and (b) is for a cross thread operation.

screws. For both problems, a possible solution is to add more features and constraints in process models. This solution requires more human effort for observing special patterns in each stage. Another solution is to learn the process models, like fitting Gaussian process models, at the expense of generalization ability.

VI. CONCLUSION AND FUTURE WORK

In this paper, we develop a failure detection system which can perform stage and result prediction. This system combines known process models with a HMM. With a good estimation of process models, our system can perform unsupervised learning as unlabeled data accumulate, as well as generalize to similar but different screwdriving operations. We show that with prior process knowledge about a specific system, we can develop a robust failure detection system with very limited data. Large amount of labeled data are actually unnecessary in this task. Our system provides a robust solution to process monitoring tasks with limited resources of data.

One of the future improvement of our system is to decrease the required labeled generalization sample to zero. This can be achieved by data adaptation. On the other hand, our future work includes two new elements: alignment error estimation and recovery strategy development. Predicting real-time stages is not enough for a robust screwdriving system. Recoveries are needed, for which the system should specify not only the current failure, but also the magnitude of failures, like alignment error values.

ACKNOWLEDGMENTS

The authors would like to thank Foxconn for providing the financial support. The views and opinions expressed in this paper do not necessarily state or reflect those of Foxconn.

REFERENCES

[1] Z. Jia, A. Bhatia, R. M. Aronson, D. Bourne, and M. T. Mason, "A survey of automated threaded fastening," *IEEE Transactions on Automation Science and Engineering*, no. 99, pp. 1–13, 2018.

[2] X. Cheng, Z. Jia, A. Bhatia, R. M. Aronson, and M. T. Mason, "Sensor selection and stage & result classifications for automated miniature screwdriving," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2018, p. Accepted.

[3] R. M. Aronson, A. Bhatia, Z. Jia, M. Guillame-Bert, D. Bourne, A. Dubrawski, and M. T. Mason, "Data-driven classification of screwdriving operations," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 244–253.

[4] D. E. Whitney, *Mechanical assemblies: their design, manufacture, and role in product development*. Oxford university press, 2004, vol. 1.

[5] U. S. C. P. S. Commission. (2018) Lenovo recalls thinkpad laptops due to fire hazard. [Online]. Available: https://www.cpsc.gov/Recalls/2018/lenovo-recalls-thinkpad-laptops-due-to-fire-hazard

[6] J. H. Bickford, *Introduction to the design and behavior of bolted joints: non-gasketed joints*. CRC Press, 2007.

[7] E. J. Nicolson, "Grasp stiffness solutions for threaded insertion," Master's thesis, University of California, Berkeley, 1990.

[8] S. Wiedmann and B. Sturges, "Spatial kinematic analysis of threaded fastener assembly," *Journal of Mechanical Design*, vol. 128, no. 1, pp. 116–127, 2006.

[9] M. A. Diftler, "Alignment of threaded parts using a robot hand: Theory and experiments," Ph.D. dissertation, Rice University, 1998.

[10] T. Matsuno, J. Huang, and T. Fukuda, "Fault detection algorithm for external thread fastening by robotic manipulator using linear support vector machine classifier," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3443–3450.

[11] S. Smith, "Use of microprocessor in the control and monitoring of air tools while tightening thread fasteners," *Eaton Corporation, Autofact West, Proc. Society of Manufacturing Engineers, Dearborn, MI*, vol. 2, 1980.

[12] R. S. Shoberg, "Engineering fundamentals of threaded fastener design and analysis. i," *Fastening*, vol. 6, no. 2, pp. 26–29, 2000.

[13] R. Isermann, *Fault-diagnosis applications: model-based condition monitoring: actuators, drives, machinery, plants, sensors, and fault-tolerant systems*. Springer Science & Business Media, 2011.

[14] K. Althoefer, B. Lara, Y. Zweiri, and L. Seneviratne, "Automated failure classification for assembly with self-tapping threaded fastenings using artificial neural networks," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 222, no. 6, pp. 1081–1095, 2008.

[15] J. R. Flanagan, M. C. Bowman, and R. S. Johansson, "Control strategies in object manipulation tasks," *Current opinion in neurobiology*, vol. 16, no. 6, pp. 650–659, 2006.

[16] K. Ogawara, J. Takamatsu, H. Kimura, and K. Ikeuchi, "Modeling manipulation interactions by hidden markov models," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 2. IEEE, 2002, pp. 1096–1101.

[17] L. Rozo, P. Jiménez, and C. Torras, "A robot learning from demonstration framework to perform force-based manipulation tasks," *Intelligent service robotics*, vol. 6, no. 1, pp. 33–51, 2013.

[18] T. J. Debus, P. E. Dupont, and R. D. Howe, "Contact state estimation using multiple model estimation and hidden markov models," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 399–413, 2004.

[19] O. Kroemer, H. Van Hoof, G. Neumann, and J. Peters, "Learning to predict phases of manipulation tasks as hidden states," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4009–4014.

[20] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1503–1510.

[21] W. Heemels, D. Lehmann, J. Lunze, and B. De Schutter, "Introduction to hybrid systems," *Handbook of Hybrid Systems Control–Theory, Tools, Applications*, pp. 3–30, 2009.

[22] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal, "Identification of hybrid systems a tutorial," *European journal of control*, vol. 13, no. 2-3, pp. 242–260, 2007.

[23] A. M. Johnson, S. A. Burden, and D. E. Koditschek, "A hybrid systems model for simple manipulation and self-manipulation systems," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1354–1392, 2016.

[24] G. Lee, Z. Marinho, A. M. Johnson, G. J. Gordon, S. S. Srinivasa, and M. T. Mason, "Unsupervised learning for nonlinear piecewise smooth hybrid systems," *arXiv preprint arXiv:1710.00440*, 2017.

[25] R. M. Aronson, A. Bhatia, Z. Jia, and M. T. Mason, "Data collection for screwdriving," in *Robotics Science and Systems, Workshop on (Empirically) Data-Driven Manipulation*, 2017.

[26] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.