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 is still 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 stage and result prediction 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, enabling our system to generalize across different experimental settings and largely reduce the required data and labeling. In our experiments, a system trained on M1.4x4 screws adapted with very little non-labeled data to three other screws (M1.2x3, M2.5x5, and M1.4x4) with widely varying tightening current, motor velocity, insertion force, and tightening force.**

I. INTRODUCTION

Screwdriving is one of the most commonly used assembly methods [1]. In the consumer electronics industry for example, hundreds of billions of tiny screws are assembled every year; however, fully automating this huge-volume assembly remains very challenging, especially for smartphones [1], [2], [3]. Smaller screws require tighter tolerance and higher alignment accuracy [4]. Moreover, compared to the well-studied peg-in-hole problem, screwdriving has more process stages and failure modes; most stages involve multiple contacts with highly variant discontinuous surfaces, making the mechanics extremely complicated [5]. A system capable of online process monitoring, failure prediction and recovery is necessary for highly automated solutions [3]. However, existing work, including ours [3], are still preliminary. This paper aims to address the robustness and data-efficiency issues in process monitoring and failure prediction for screwdriving.

Previous screwdriving failure detection methods have robustness issues. First, all algorithms except [3] can only do result classification, which alone cannot detect irreversible process failures [6]. Second, the screwdriving process has different stages that lead to different results. Most methods only use global features for result prediction while ignoring the state-machine-like nature of the screwdriving process. Third, most existing methods cannot distinguish unseen anomalies or unknown failures. Fourth, previous methods are not guaranteed to work when experimental conditions (e.g., screw sizes) change. We believe that a robust failure detection system should: 1) accurately predict different failures; 2) perform real-time stage classification; 3) distinguish unknown anomalies; 4) quickly adapt to new assembly needs.

Due to difficulties in screwdriving process modeling, most failure detection systems are essentially data-driven. Directly learning from data yields good results. However, this approach often needs lots of training data and laborious labeling that requires expert knowledge [6]. Collecting a large dataset that covers various failure modes is extremely time-consuming and requires extra engineering effort, mainly due to the "long tail" effect [7]: most failure rates are less than $1\% - 2\%$. Hence, can we minimize the data labeling and avoid recollecting data when experiment setup changes?

To address the robustness and data-efficiency issues, we propose a system (Fig. 1) that combines process knowledge with learning method for stage and result classifications in an unsupervised and data-adapting manner. From the sensor signals, we construct knowledge-based process models and feed them into a hidden Markov model (HMM), through which we do stage estimation and rule-based result prediction. The HMM model automatically adapts and improves as data feed in. This framework is built on the following facts: the mechanics (Section IV-A) that dominates individual stages and the structure of stage transition graph (Fig. 4) do not change over experiment conditions. For operation-invariants, process models (Section IV-A) encapsulate local invariant characteristics (e.g., mechanics for each stage) in their constraint equations, while an HMM incorporates globally invariant stage 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.

Our system shows robustness and data-efficiency in the following ways. First, it performs good online stage classification and result prediction. Second, with very simple and limited prior knowledge, our system automatically labels a large amount of data, thus freeing people from tedious labeling. As new data accumulate, the system can adapt to minor changes in data without human intervention. Third, our system can generalize and adapt to new experiment setups with little new data. This fast adaptation helps with the problem of getting an entire new dataset covering all stages and results. Fourth, our system can distinguish some unseen situations.

To summarize, our main contributions are:

- The first attempt of unsupervised learning and automatic labeling in screwdriving, to our best knowledge.

The authors are (were) 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

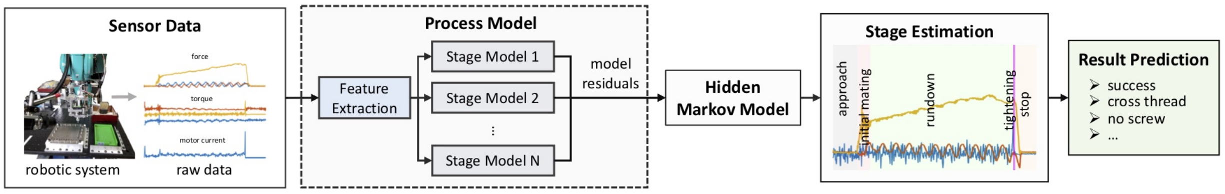

Fig. 1. An overview of our stage estimation and result prediction pipeline.

- A fast generalization framework to solve industrial problems.
- Significant reduction in data collection and labeling, by taking full advantage of screwdriving mechanics.

II. RELATED WORK

A. Analysis of the Screwdriving Process

A thorough understanding of screwdriving process is critical for reliable and accurate fault detection. Typically, the process can be segmented into several stages: *initial mating*, *rundown*, and *tightening*, according to its *torque-angle curve* [8]. Recently, [6] presents a more complete stage transition graph, covering multiple process stages and result classes. A standard operation often include stages like *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 [9], [5] and [10]. A force spike can be used as an indicator [10] [6]. In the *rundown* stage, quasi-static analysis shows that the oscillation phenomenon is an important signature [11]. In the *tightening* stage, torque, rotation angle and torque-angle gradient [12] are commonly used for failure detection. In Section IV-A, we combine these features with our own analysis to develop screwdriving process models.

B. Fault Detection for Threaded Fastening

In industrial screwdriving, the most common fault detection approach is the *teach method* [1]. Correct torque-angle fastening signatures are collected as reference, which is then compared with actual signals using limit check or trend check [13]. The *teach method* is simple and easy to implement, but it lacks flexibility and generality. To overcome these problems, intelligent screwdriving systems are developed, most of which fall into two categories: model-based or data-driven. Model-based methods [14] require analytic models and accurate system parameters. While data-driven methods, such as ANNs [15], SVMs [11], GTC-DF [6], and decision trees [3], directly learn from labeled data. Model-based approaches are flexible, but accurate system models are hard to obtain. Data-driven methods do not require prior knowledge, but they are difficult to adapt to variations.

C. Discrete State Estimation in Manipulation

A typical robot task (including screwdriving) often includes a series of discrete states [16]. Identification of states is critical during task operations, which can be formulated by 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 the hybrid model parameters.

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

In contrast, the hybrid system approach is more deterministic. Hybrid systems involve both continuous models and discrete states [20]. A discrete state can be identified when the states and inputs fall into the corresponding domain [21]. Hybrid systems can incorporate simplified mechanical models to approximate complex robot behaviors, as well as to reason mechanisms and to perform feedback control [22].

III. DATA COLLECTION

Our data is collected with the robotic screwdriving system shown in Fig. 2. By using a “floating structure” [3] design, the forces and torques exerted on the screwdriver tip can be measured by a 6-axis force/torque (F/T) sensor. This system is an upgrade of our previous system used in [6] [7], where more detailed descriptions can be found. In each operation, a screw from the tray is first picked up by the screwdriver using vacuum suction. Then the robot moves to the calibrated screw holes on a plate at a specified height. The screwdriver motor turns on and starts collecting data as the screw insertion starts. Each run terminates when the motor current or motor angle reaches the specified limit. The system records robot positions, 6-axis F/T data, motor current, motor encoder readings, linear potentiometer values that measure the spring displacement (to trace the screwdriver tip), and videos (as the ground truth) from a high-speed camera. A typical successful run would look like Fig. 3.

Following the above procedure, we collected more than 7000 experiment runs on different sized small screws (from M1.2 to M2.5) with various insertion force profiles, motor speeds and alignment errors. In this paper, we select several subsets of different setups to verify our method. The detailed information is described in Section V.

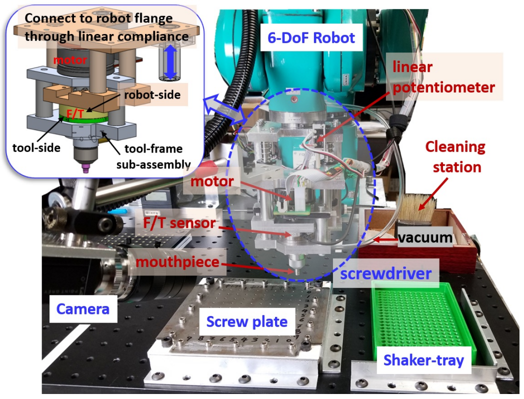

Fig. 2. Intelligent robotic screwdriving system for data collection.

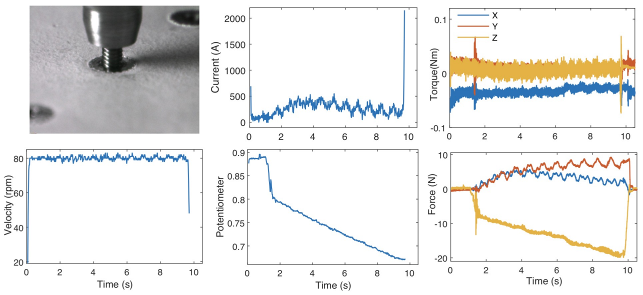

Fig. 3. An example of collected raw sensor data.

IV. METHODS

This section describes our system framework and generalization approach in detail. Our system framework for failure detection and process monitoring consists of three parts:

1) Modeling the screwdriving process: develop process models using expert knowledge;
2) Learning an HMM: train an unsupervised HMM on the unlabeled data;
3) Predicting the result using rules: use known process rules to predict the result of an operation (success or different failure types).

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

Fig. 1 shows an overview of our framework. First, all possible stages during the screwdriving process are specified and modeled as constraint equations. Second, the model residuals of constraint equations are used as inputs of HMM, which can be trained using Expectation-Maximization (EM) algorithm. Once the stages are predicted by HMM, results are inferred from stages using rules constructed from the stage transition graph. Third, given new data from similar experimental setups, a well-trained system generalizes to new data by parameter specification and HMM learning.

A. Modeling of the Screwdriving Process

From previous research work and our collected data, the screwdriving process can be modeled through mechanical knowledge and empirical observations. Our modeling includes three parts: building a stage transition graph, designing a feature set, and modeling stages as constraint equations.

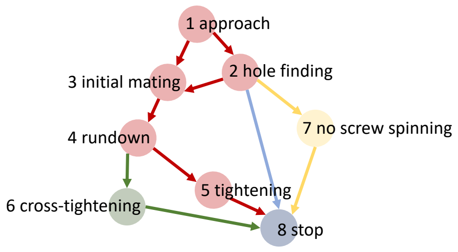

Fig. 4. The stage transition graph in the screwdriving process

1) State Transition Graph: A screwdriving operation can be modeled as a sequence of discrete states, which are defined as *stages*. The stage transition graph (Fig. 4) describes all possible transitions among the stages of our system. All 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 there are alignment errors. Different types of failures are indicated by their process stages, such as *no screw spinning* and *cross-tightening*, with corresponding results as *no screw* and *cross thread*. All operations end at *stop*. A well-defined graph enables us to clearly and separately model the screwdriving process, because each stage has only one dynamic model. Moreover, this graph is used as the HMM state transition model. Note that this graph is modified from our work in [6]. Compared to the previous graph, by exerting proper insertion force during operation, stages and results related to *stripped* happen very rarely, thus can be removed from the stage transition graph and better to be treated as anomaly. More importantly, instead of different result types, we use a *stop* stage as the uniform termination mode, to reduce the stages and parameters to learn.

2) Feature Extraction: Feature extraction pre-processes the raw sensor data before stage modeling. We design a feature set with mechanical meanings concluded from the previous work and our observations. These features enable complicated process models to be transformed into mostly linear equations of features and parameters, making parameter estimation and generalization clearer and easier.

At timestep t, a set of features is extracted inside a fix-sized window w from signal data before t:

v_x, v_y, v_z: frequency of the 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 [11]. Under small alignment error assumption, the frequency approximately equals the screwdriver motor velocity.

d_{tip}: the distance from screwdriver tip to the screw hole. It is a strong indicator of the process status. It is calculated as: $d_{tip} = r_z - h_z - l_z$, where r_z is the screwdriver end position on z axis in the whole time window, computed from the robot positions. l_z is the compression of linear potentiometer, and h_z is the z-value of the calibrated hole location.

$\Delta_{d_{tip}}$: the change of screwdriver tip distance in the time window, $\Delta_{d_{tip}} = max(d_{tip}) - min(d_{tip})$.

k_{tip}: the gradient of screwdriver tip distance. It is computed by least square estimation for $d_{tip} = k_{tip}t$. The other gradients in the following are also computed in the same way.

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 motor current-time gradient. Our screwdriver employs a direct-drive motor; therefore, the motor torque is proportional to the motor current. It is used as a redundant feature to provide more robust information against relatively large noise on the torque sensor.

Δ_{fz}: the change of force on the z-axis inside the time window, $\Delta_{fz} = max(f_z) - min(f_z)$, where f_z is the z axis force in the time window. A drop of z-axis force often indicates the alignment or mating of screw threads [10].

$\mu_{\mathbf{f}}$, $\sigma_{\mathbf{f}}$: the means and variances of forces.

3) Stage Models: Each stage is modeled as a set of constraint equations using physics or process knowledge. After stage models (constraint equations) are obtained, the model errors are passed to the HMM to perform stage estimation. The parameters of stage models can be specified by user or estimated from data. Some parameters can be specified from the experiment setup: V_m (motor velocity), H_1 (height of screw head), H_2 (cross thread height), C_{1kta} (tightening torque-angle gradient), C_{2kta} (cross-tightening torque-angle gradient), C_{1kc} (tightening current-time gradient), and C_{2kc} (cross-tightening current-time gradient). Other parameters are learned from the labeled data before HMM training. After stage models (constraint equations) are obtained, their errors are passed to HMM to perform further estimation.

Our stage models are constructed as follows:

approach: no contact between the screw and target hole, 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 constant: $\Delta_{fz} = 0$.

initial mating: two possible models exist in this stage. The first model is a smooth mating, where the 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 screwdriver 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 [11]. 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 a constant C_{1kc}. The distance of screwdriver tip to the screw hole surface is the thickness of screw head H_1: $k_{ta} - C_{1kta} = 0$; $k_c - C_{1kc} = 0$; $d_{tip} - H_1 = 0$.

cross tightening: the same model as *tightening* but with different model parameters C_{2kta}, C_{2kc} and H_2. During *cross tightening*, the torque-angle gradient is much smaller than that of *tightening* due to angular errors. For a cross-thread case, only the first external thread is mated at the cross-thread angle [9], 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 the frequency of V_m occurs after contact starts, because no screw head is mated with screwdriver tip: $v_z - V_m = 0$.

stop: the screwdriver motor stops, thus the signal values of motor current m_c and motor velocity m_v are equal to zero: $m_c = 0$; $m_v = 0$.

B. Stage Prediction by Hidden Markov Model

Given data from a screwdriving 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 to make stage prediction. Given a time series of observations, an HMM can estimate the sequence of states which generate these observations.

1) HMM Representation: An HMM λ is composed of states S, initial state distributions π, state transition distribution A, and observation probability distributions B [23]. A compact notation of the HMM parameters is $\lambda = (A, B, \pi)$. In our system, the hidden states $S = \{S_1, S_2, \dots, S_N\}$ are the stages in Fig. 4. The state at time t is denoted as q_t. The initial state distribution is the probability of each state that appears at the start, denoted as $\pi = \{\pi_1, \pi_2, \dots, \pi_N\}$, where $\pi_i = P(q_1 = S_i)$.

The state transition probability matrix $A = \{a_{ij}\}$ represents the probability of state S_j occurs after S_i, where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i), 1 \le i, j \le N$. If there is 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. In our method, the $P(O_t | q_t = S_i)$, written as $b_i(O_t)$, is represented in the form of multivariate Gaussian distribution:

$$b_i(O_t) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp(-\frac{1}{2}(x_{it} - \mu_i)^T \Sigma^{-1} (x_{it} - \mu_i)$$

where x_t is the residuals of all constraint equations computed from O_t; μ_i, Σ_i are the mean vector and covariance matrix of the model residuals for state S_i.

2) Stage Classification: Stage classification is to find the state sequence that maximize its joint probability $P(q_1, q_2 \dots q_t \wedge O_1, O_2 \dots O_t | \lambda)$ with the observation. Viterbi algorithm, a dynamic programming algorithm, can solve this problem efficiently [23]. The forward-backward procedure is performed firstly. The forward variable is defined as $\alpha_t(i) = P(O_1, O_2 \dots O_t, q_t = S_i | \lambda)$. It can be solved inductively as: $\alpha_1(i) = \pi_i b_i(O_1)$; $\alpha_{t+1}(i) = \sum_{j=1}^{N} \alpha_t(j) a_{ji} b_i(O_{t+1})$. Similarly, the backward variable $\beta_t(i) = P(O_{t+1}, O_{t+2} \dots O_T | q_t = S_i, \lambda)$ can also be inductively solved as: $\beta_T(i) = 1$; $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$.

We define the most probable state sequence ended with S_i at timestep t as $\text{mpp}_i(t)$, and its probability $\delta_i(t) =$

$\max_{q_1 \ldots q_{t-1}} P(q_1, q_2 \ldots q_{t-1} \wedge q_t = S_i \wedge O_1 \ldots O_t)$. The most probable path computation is as follows:

$$\delta_1(i) = \pi_i b_i(O_1)$$

$$\delta_{t+1}(j) = \delta_t(i^*) a_{i^*j} b_j(O_{t+1}))$$

$$\mathrm{mpp}_j(t+1) = [\mathrm{mpp}_{i^*}(t), S_{i^*}]$$

where $i^* = \arg\max_i \delta_t(i) a_{ij} b_j(O_{t+1})$.

During the screwdriving process, at each timestep t, the current state sequence is predicted as $mmp^*_j(t+1)$, where $j^* = \arg\max_j \delta_t(j)$.

3) Data Adaptation: HMM can iteratively update and adjust to a better model as data accumulates using Expectation-Maximization (EM) algorithm [23]. In our case, data adaptation is achieved by updating the mean μ and variance Σ of the multivariate Gaussian observation model.

The probability of observing state S_i for all $i = 1, \ldots, N$ can be computed as follows:

$$\gamma_t(i) = P(q_t = S_i | O_1 \ldots O_t, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

Thus the mean and variance can be updated similar to the weighted sum:

$$\hat{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) x_t}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\hat{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(x_t - \hat{\mu}_i)(x_t - \hat{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}$$

Our HMM training efficiently incorporates process knowledge through the initialization and constraints of HMM parameters. Since the starting states can only be S_1, *approach*, we have $\pi = \{1,0,0,0,0,0,0,0\}$. 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. For observation probability $b_i(O_t)$, all the Gaussian means are initialized as zeros. The initialization of covariance matrix, as a diagonal matrix, is the key to differentiate states. For state S_i, the elements in Σ_i that correspond to its own constraint equations in all equations are initialized to be small, while the variances corresponding to other state constraint equations are set to be very large. During training, the mean and covariance are constrained so that our observation models will not deviate too much from the stage models.

4) Anomaly Detection: We detect anomaly in two cases, unseen/unknown stages or impossible path patterns. Unseen or unknown stages are mathematically defined as those have observation probability very close to zero for each state. Impossible path pattern is defined as for all possible states that give observations, there is no possible transition that gives nonzero probability. These two cases can be unified into one, that the probability of the observations under our current HMM model is zero, written as $P(O_1 \ldots O_T | \lambda) = \sum_{i=1}^{N} \alpha_T(i) = 0$. When this probability is extremely close to zero, our system determines there is a unexpected problem.

C. Rule-Based Result Prediction

The result of an operation can be inferred from its stage history [6]. There are five types of results in this paper: *success*, *cross thread*, *no screw*, *no hole found* and *partial*. The rules for result prediction are constructed as follows:

1) If certain failure stages are detected, such as *cross tightening* and *no screw spinning*, the system will stop in advance and predict the corresponding failure types, *cross thread* and *no screw*.

2) If the time of *hole finding* or *initial mating* (with large forces) stage exceeds certain threshold, this indicates large non-correctable alignment error that might damage the screw plate. The system will stop and predict *no hole found*.

3) If the operation follows the red path on Fig. 4, then the system will proceed to condition check, which will examine the final insertion length, tightening torque and tightening torque-angle gradient. The result is predicted as *success* if condition check is passed, otherwise *partial*.

D. Generalization

Previously trained HMM cannot be directly applied when the operation conditions change. Knowing the new operation settings, our system can quickly generalize to new experiment conditions by changing process model parameters for the new data. First, the change of experiment setup is specified and modified in the process model as in Section.II-A.3. Then, the relationship of parameters among different stages is estimated from previous data, which is used to update unspecified stage parameter. 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. These estimated parameters do not need to be accurate because we can always use data adaptation to refine. Finally, the means and covariances in the HMM observation model is adjusted by the newly estimated process models. The new HMM keeps improving as new experiment data feeds in.

V. EXPERIMENTS

A. Dataset

Our experiments are performed on the four datasets shown in Table I, where each dataset is collected by our robotic system under different operation conditions. The *tightening current* is the motor stop threshold. The *insertion force* is the z-axis force exerted on the screw when screwdriving operation starts. The *tightening force* is the z-axis force at the time when screw tightening finishes. In our experiments, we uniformly sample all data at 100 Hz.

To further evaluate the robustness of our system, for all the dataset, we collect data with both high precision alignment (1/4 of total) and random alignment errors (3/4 of total). For high precision alignment data, accurate positions of screw holes are given to the robot. To generate data errors, random alignment errors are added to the screw hole positions. The alignment errors include translational errors (in the range of 0 to 40% of screw diameters) and angular errors (0 to

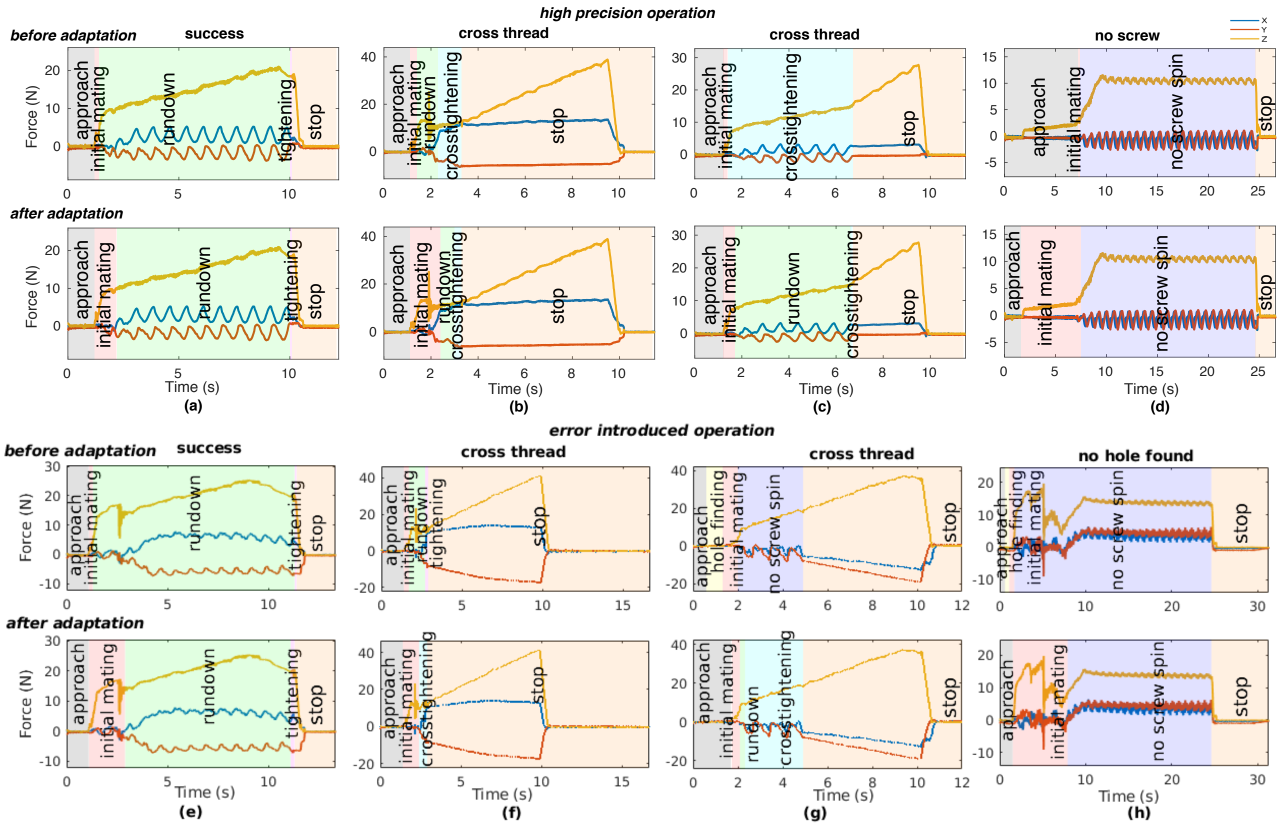

Fig. 5. Comparison of the stage predictions before and after data adaptation on high precision operation data and error introduced operation data. Different color blocks correspond to different stages. Only the force sensor data are plotted due to space limitations. The actual result for each operation is shown on the top of the figure.

TABLE I
DATA COLLECTION SETTINGS

dataset	*1*	*2*	*3*	*4*
screw size	M1.4x4	M1.2x3	M2.5x5	M1.4x4
tightening current (mA)	1600	1300	3200	2400
motor velocity (rpm)	80	96	80	320
insertion force (N)	10	6	18	10
tightening force (N)	20	14	30	20

6 degrees of the screwdriver axis to the screw hole). Since errors are injected, the average rate of successful operations is around 75%. Compared to the industrial data, our data have much more unexpected and deviated patterns, which largely increase the difficulty for prediction.

B. Implementation

We implemented our system using MATLAB. 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 set up according to our operation settings. Then the learnable parameters of all stage models are estimated from the 10 labeled aligned samples from *dataset 1*. The parameter estimation is solved as the least square problem.

We implement the HMM based on [23]. The proper initial values of covariances corresponding to their state constraint equations can be set as $1/10$ of corresponding constant terms, if not available in parameter estimation. The other large error variances can be initialized as 100 for small screws; these values are empirical, but they can be properly scaled for other sizes. During training, to ensure convergence, the mean and covariance in observation models are constrained so that the variations of the initial stage models are within $\pm 15\%$. When the change of log likelihood of observations is less than a set threshold of 30, the system is considered as converged. For the training with 50 samples, our HMM often converges after 8-10 batches in less than 3 minutes.

The rules of result prediction are constructed as *if-else* statements. The values of condition check will vary with respect to the assembly requirements.

When generalizing to new settings, the model parameters of process constraint equations are modified directly based on the new setups. The parameters not designated in the experiment settings are directly inferred from the relationship among the models in the old system. As the process models change, the initial covariances and means of the HMM Gaussians change proportionally.

C. Results

Some selected operations are firstly visualized. The actual performances of our method are then evaluated by failure prediction accuracy.

1) Stage and Result Classifications: Fig. 5 shows the stage classification of our HMM on both high accurate alignment data and random alignment error data before and after training. The HMM before training is initialized on

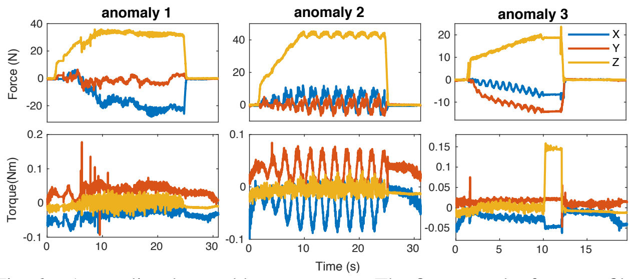

Fig. 6. Anomalies detected by our system. The first row: the force profiles. The second row: the torque profiles.

10 accurately aligned samples without any training, while the HMM after training makes adaptation on 50 samples of both alignment error types. Both HMMs are tested on new samples that have never been seen by our system. This comparison emulates the process of our HMM adapting to new data as the system gradually changes. Our HMM before training can make good prediction on accurate alignment data for all types of failures. With data adaptation, the HMM after training learns richer representations of stages and performs more accurate stage classification. For example, in Fig. 5 (a), the HMM after training predicts to extend the period of *initial mating* to exactly where an expert label would be. In Fig. 5 (c) and (d), the predictions for *crossthightening* are narrowed down to the right period that cross-thread occurs. As errors are introduced, the HMM before training makes more mistakes because of the deviation of the signal data. With data adaptation, the HMM after training adjusts to small changes while maintains performance on accurate alignment data.

Fig. 5 covers most of the result types that can occur in our system. The operation *partial* has the same signal profiles as *success*, it is the quality check based on actual assembly requirements that distinguishes them. The profiles of *no hole found* vary largely; Fig. 5(h) visualizes one example: the screw tip fails to match the screw hole during *initial mating*, when the screwdriver goes down, the screw is pressed away and thus results in stage *no screw spinning*.

2) Anomaly Detection: Fig. 6 shows three anomaly cases that our system detected. It is even hard for human to tell what exactly happened simply from the signals. By checking the videos, we found that: for *anomaly 1*, the screws failed matching the screw hole, and the screwdriver was stuck by the head of the fallen screw. For *anomaly 2*, the screw was cross-threaded and accidentally stripped due to misalignment, thus it did not result in the usual cross-thread profile and spun for longer. In *anomaly 3*, the data connection was lost for two seconds.

3) Generalization: As shown in Fig. 7, without any label, our method generalize to *dataset 2*, *dataset 3* and *dataset 4* with different screw sizes, insertion forces, tightening torques and forces, motor velocities and even noise-signal ratios,.

In *dataset 2*, we use tiny screws, for which the tightening torque is much smaller. In this case, the noise-signal ratio of the torque sensor can be as high as 50%, as shown in Fig. 7, *dataset 2*. But our method can reduce its influence and generalize to different noise-signal ratios by exploiting

TABLE II

RESULT ACCURACY

dataset		*1*(accurate)	*1*(error)	*2*	*3*	*4*
HMM type		original		generalized		
adapt	before	97.47%	77.27%	89.90%	91.33%	88.57%
	after	98.48 %	84.85%	91.71%	95.92%	91.43%

known meaningful models.

In *dataset 3*, the screw size is almost 2 times of that in our original dataset. This verified that our model can be adapted to much larger screw sizes and tightening torques.

In *dataset 4*, the system generalizes to a motor speed over four times faster than the original dataset. In real factory settings, the screwdriving motor speed varies and sometimes can be really fast.

D. Evaluation

We evaluate our method using the accuracy of result prediction (Table.II), the proportion of correctly predicted *success* and different failures in all the samples. We do not directly measure the performance of stage classification because it is hard and extremely time consuming to manually label the stages. We only manually labeled the results for all datasets. The result prediction, inferred by stage classification, can indirectly show the effectiveness of stage classification.

Our models obtained good result prediction accuracy, both for the original ones with parameter estimation from labeled samples and for generalized ones without any labeling. By data adaptation, our model can automatically fit to the actual data and make more accurate predictions. Note that accuracy of *dataset 1* with injected errors and *dataset 2* are not as good as the others. For *dataset 1*, to verify the robustness of our method, we use very large random alignment errors, which produced many largely deviated signal patterns. The lower result accuracy is because that the HMM is trained with limited data that cannot cover these deviations. This is not a problem for industrial use, where operations are highly accurate and repeatable. For *dataset 2*, the noise-to-signal ratio is too high (50%) for z-axis torque. A solution is to filter the noise when necessary. Note that because we injected large alignment errors in our dataset, introducing many unexpected and deviated signal patterns, our result accuracy is not comparable to the desirable 99.9% for industry use.

E. Limitations

When tuning the models for data adaptation, we found that one stage model might fit to another stage, because the prediction errors can accumulate and reinforce during training without labels. To prevent this, we add constraints on the mean and covariance. However, to truly solve this problem, we need observation models with more representative capacity, but complicated models require more data and labeling to train. One possible solution to avoid this trade-off might be introducing human correction. If an expert can tell the model whether it is making the right predictions, more complicated models can be deployed while still avoiding tedious manual labeling.

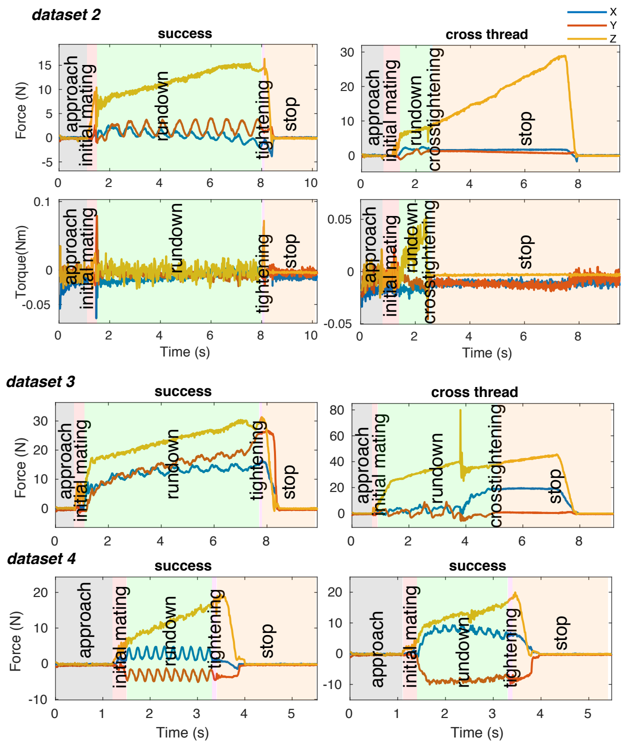

Fig. 7. The visualization of the generalization of our system on *dataset 2* (force and torque profiles), *dataset 3* and *dataset 4* (force profiles only).

VI. CONCLUSION AND FUTURE WORK

In this paper, we develop a failure detection system which can perform stage and result prediction under limited data resource. 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 even with very simple prior process knowledge about a specific system, we can develop a robust failure detection system with very limited data. This method can be extended to processes similar to screwdriving (with complex underlying models but limited human knowledge).

Our future work includes accuracy improvement through expert correction, alignment error estimation and recovery strategy development. The accuracy can be largely improved and the deviation during data adaptation can be corrected if an expert can spend a little effort telling whether the system is making correct prediction. Moreover, 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, such as 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] Z. Li. Robotics research for 3c assembly automation. [Online]. Available: https://app.box.com/s/zcg8qqxt6fw6v4xz22h6

[3] 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.

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

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

[6] 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.

[7] 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.

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

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

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

[11] 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.

[12] 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.

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

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.