Journal of Manufacturing Systems 58 (2021) 306–315

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Multi-task learning for data-efficient spatiotemporal modeling of tool surface progression in ultrasonic metal welding

Haotian Chen, Yuhang Yang, Chenhui Shao *

Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, United States

ARTICLE INFO

Keywords:
Spatiotemporal process
Multi-task learning
Transfer learning
Kernel method
Surface monitoring
Quality control
Ultrasonic metal welding
Manufacturing

ABSTRACT

Spatiotemporal processes commonly exist in manufacturing. Modeling and monitoring such processes are crucial for ensuring high-quality production. For example, ultrasonic metal welding is an important industrial-scale joining technique with wide applications. The surfaces of ultrasonic welding tools evolve in both spatial and temporal domains, resulting in a spatiotemporal process. Close monitoring of tool surface progression is imperative since degraded tools often lead to low-quality joints. However, it is generally expensive and time-consuming to acquire fine-scale surface measurement data, which is not economically viable. This paper develops a multi-task learning method to enable data-efficient spatiotemporal modeling. A Gaussian process-based hierarchical Bayesian inference structure is constructed to transfer knowledge among multiple similar-but-not-identical measurement tasks. Meanwhile, a spatiotemporal kernel is developed based on squared sine exponential damping (SSED) function to characterize the periodic trend of anvil surfaces. The proposed method is able to improve interpolation accuracy using limited measurement data compared with state-of-the-art techniques. Data collected from lithium-ion battery production are employed to demonstrate the effectiveness of the proposed method. Additionally, the influence of training data size and hyperparameter selection on the modeling performance is systematically investigated.

1. Introduction

Spatiotemporal processes widely exist in manufacturing. Modeling and monitoring spatiotemporal processes are of great interest to manufacturers and industry practitioners. For instance, in ultrasonic metal welding, which is an important industrial-scale solid-state joining technique, the surfaces of welding tools change both spatially and temporally, as shown in Fig. 1, and the surface degradation leads to low-quality joints [1–6]. Modeling and monitoring the tool surface degradation are crucial for improving the process robustness [7] and online monitoring of product quality [8,9,1]. In automotive machining processes, spatial and temporal changes in tool geometry result in machined parts with different surface patterns and quality [10,11]. In addition, the geometry modeling of surface spatial variation [12] and time-varying deviations [13] have been conducted for monitoring the quality of machined parts.

Fine-scale measurement data of spatiotemporal processes is crucial in manufacturing applications to enable effective modeling, monitoring, and control. The acquisition of fine-scale measurement data, however, is often expensive and time-consuming [2,4]. For instance, it may take a three-dimensional (3D) microscope around 8 hours to scan an anvil surface with a dimension of 43 mm × 8 mm in ultrasonic metal welding [2], which brings prohibitive production downtime. Additionally, multiple factors may limit the availability of measurement data. For example, the surface measurement processes are subject to the disturbances including measurement table vibration, dissipated heat, and surface contamination [14].

In light of the challenges brought by the high cost of the measurement process and limited measurement data, researchers and practitioners have developed various interpolation techniques to predict the values at unmeasured locations using available data. The interpolation methods can be generally categorized into deterministic and stochastic methods. The former includes inverse distance weighted interpolation [15], B-spline methods [16], and artificial neural networks [17]. The representative stochastic models include ordinary kriging [18] and its variants such as co-kriging [19] and kriging with external drift [20]. Different kernels, such as Bessel additive variogram [21], have been developed to extend the capability of kriging methods to model different

* Corresponding author.
E-mail addresses: hc19@illinois.edu (H. Chen), yang221@illinois.edu (Y. Yang), chshao@illinois.edu (C. Shao).
URL: http://mechanical.illinois.edu/directory/profile/chshao (C. Shao).

https://doi.org/10.1016/j.jmsy.2020.12.009
Received 19 August 2020; Received in revised form 10 December 2020; Accepted 11 December 2020

spatial variation patterns. These methods have been widely adopted in manufacturing applications, including wafer profile monitoring in semiconductor industry [22], quality control in additive manufacturing [23], and tool condition monitoring in ultrasonic metal welding [2,4,3]. Because these methods estimate missing values from nearby locations, their effectiveness relies on adequate measurement data. As the data of nearby locations become limited, their performance quickly degrades.

Spatiotemporal modeling overcomes such limitation by leveraging both spatial and temporal correlations to infer on unmeasured locations. The addition of temporal information often leads to improvements in modeling performance. Recent methods on spatiotemporal modeling and prediction include shapelet based spatial-temporal feature extraction [24] and evolutionary algorithm based spatiotemporal prediction [25]. As an example, in automotive machining processes, Babu et al. adopted a state-space spatiotemporal modeling to improve the quality inspection by predicting the deviations of the entire part from partial measurements [26]. However, when data scarcity is more severe, the effectiveness of spatiotemporal modeling methods is impaired, because the consistent data deficiency across all time stages prohibits transferring information among time stages.

To cope with this challenge, this paper develops a multi-task learning method for data-efficient spatiotemporal modeling. It is motivated by the fact that in factories, manufacturing tasks are often performed by multiple machines in parallel, which share much similarity. High standardization of modern manufacturing further enforces this similarity [27]. Therefore, it is potentially more cost-effective if spatiotemporal processes can be jointly learned by transferring information among them.

To realize the joint learning, this paper develops a method called spatiotemporal kernel based multi-task learning (STK-MTL). Each spatiotemporal process is modeled from a kernel perspective, and domain knowledge can be integrated into customized kernels. This kernel view also provides an access to the variogram method, which has been extensively studied in the geostatistics community [28]. Finally, we note that the proposed method is readily applicable to a wide range of scenarios in manufacturing.

The main contributions of this paper can be summarized as follows:

1. A new spatiotemporal modeling approach is developed by combing a spatiotemporal kernel and hierarchical multi-task learning. It provides a solution for cost-effective spatiotemporal modeling and monitoring in data scarce situations. This method is demonstrated to be more effective than learning each process separately.
2. A framework is developed to formulate a customized kernel function that can account for the periodic spatial patterns of the tool surfaces in ultrasonic metal welding. Compared with the conventional kernel used in the multi-task learning for Gaussian process, our kernel captures both periodic spatial variations and temporal correlations well.
3. The characteristics of the proposed STK-MTL approach are systematically studied, including its applicability and the effects of hyperparameters. Practical suggestions are also presented based on the experimental results.

The rest of the paper is organized as follows. Section 2 presents the STK-MTL model and its implementation. In Section 3, a case study is reported on the anvil surface to verify the effectiveness of proposed method. In Section 4, the applicability of the proposed method and the effects of hyperparameters are discussed. Finally, Section 5 concludes the paper.

2. Method

In this section, the spatiotemporal modeling approach from the kernel point of view and the kernel function construction are first reviewed, and then, the multi-task learning algorithm and its implementation are introduced.

2.1. Spatiotemporal modeling

We use spatiotemporal coordinates to represent measurement data. Each measurement can be denoted as (t, x, y, z), where t is the time when the measurement happens, and x, y, and z specify its location in the 3D space. The task of surface measurement/modeling is essentially to obtain height z in a given spatiotemporal location p, where

$$p = (t, x, y). \tag{1}$$

These spatiotemporal locations are not independent, but instead correlate with each other [28]. One way to explore these correlations is using kernel functions. The inner product $\kappa(p_i, p_j) = \left\langle \phi(p_i), \phi(p_j) \right\rangle$ provides a valid positive definite kernel, where $\phi(p_i)$ is the projection of p_i in a Hilbert space. According to the closure property of kernel functions, a spatiotemporal kernel can be constructed as the product of a spatial kernel and a temporal kernel [29] as shown below:

$$\kappa_{\text{st}}(p_i, p_j) = \kappa_s(p_i, p_j) \cdot \kappa_t(p_i, p_j). \tag{2}$$

As a prime form of spatial kernel κ_s, Gaussian radial basis function (RBF) kernel is popular in spatial statistics. It is given by

$$\kappa_s(p_i, p_j) = exp\left(- \frac{\left\| (x_i, y_i) - (x_j, y_j) \right\|^2}{\delta_s^2} \right), \tag{3}$$

where δ_s^2 is scaling factor. The intuitive interpretation for Gaussian RBF kernel is that the nearest neighbors share the most similarities.

However, the Gaussian RBF kernel fails to characterize the periodic pattern of anvil surfaces in ultrasonic metal welding, which is documented as “hole effect” [21]. To capture the “hole effect”, we can consider sinusoidal-function-based kernels, such as “waving model” and squared sine exponential (SSE) model:

$$\kappa_{\text{wav}}(p_i, p_j) = 1 - \frac{sin\left(\left\| (x_i, y_i) - (x_j, y_j) \right\| / w \right)}{\left\| (x_i, y_i) - (x_j, y_j) \right\| / w}, \tag{4}$$

$$\kappa_{\text{SSE}}(p_i, p_j) = exp\left(- \frac{sin^2(\left\| (x_i, y_i) - (x_j, y_j) \right\| / w)}{\delta_s^2} \right), \tag{5}$$

where w is the wavelength of a periodic pattern. “Wave model” fits the damping periodic trend but leads to nonzero kernel value among irrelevant locations. In fact, the affinity in these locations is expected to be zero instead. The SSE model characterizes this attribute by coating an exponential function out of the sinusoidal term, so the kernel value can

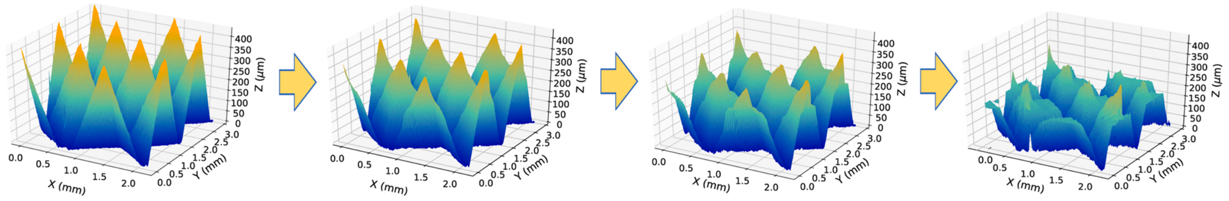


Fig. 1. The spatiotemporal progression of anvil surface topography in an ultrasonic welding process.

be kept ideally small in “inactivated” area.

Strictly speaking, the SSE model still fails to depict the damped trend for different periods, meaning that compared with distant periods, adjacent periods share more similar surface conditions with the interesting location. One possible solution is to use radial basis to characterize this trend. Popular radial basis functions with the damped trend include inverse quadratics and inverse multiquadrics [30]. To overcome this drawback, we propose an squared sine exponential damping (SSED) model, which is shown by Eq. (6), by adding inverse quadratics to a standard SSE model:

$$\kappa_{\mathrm{SSED}}(p_i, p_j) = exp[-\frac{1}{\delta_s^2}(\frac{sin^2(||x_i - x_j||/w_x)}{||x_i - x_j||^2 \Big/ w_x^2 + 1} + \frac{sin^2(||y_i - y_j||/w_y)}{||y_i - y_j||^2 \Big/ w_y^2 + 1})], \tag{6}$$

where w_x and w_y represent the periods in x and y directions, respectively. In practice, w_x and w_y can be either obtained from the provided tool geometry specifications or estimated from surface measurement data using frequency domain analysis such as fast Fourier transform.

Following the work on multi-task time-series prediction [31], we choose Gaussian RBF kernel for the temporal kernel:

$$\kappa_t(p_i, p_j) = exp\left(-\frac{||t_i - t_j||^2}{\delta_t^2}\right) \tag{7}$$

The spatiotemporal kernel $\kappa_{st}(p_i, p_j)$ is then constructed by obtaining the product of the spatial and temporal kernels, as shown in Eq. (2).

It is worth mentioning that modeling spatiotemporal processes from the kernel perspective builds a connection to the variogram model. The transformation from a variogram function to a kernel function is illustrated by:

$$\kappa_s(p_i, p_j) = C(\parallel p_i - p_j \parallel) = \gamma(\infty) - \gamma(\parallel p_i - p_j \parallel), \tag{8}$$

where $C(\parallel p_i - p_j \parallel)$ is the covariance function and $\gamma(\parallel p_i - p_j \parallel)$is the variogram function. Practically, the variogram function can be estimated using the following equation:

$$\widehat{\gamma}(h) = \frac{1}{2|N(h)|}\sum_{N(h)}\left\{[Z(p_i) - Z(p_j)]^2\right\}, \tag{9}$$

where $N(h) = \{(s_i, s_j) : ||s_i - s_j|| = h\}$ is the set of all pairs of locations grouped by lag distance h. $Z(p_i)$ is the surface height in our context of anvil surface monitoring. Expert knowledge is often incorporated in the process of choosing variogram functions. Using Eq. (8), we can convert an existing variogram function to a kernel function and use it in our STK-MTL model.

2.2. Multi-task learning

Various approaches have been developed for multi-task learning, such as transferring similar features, sharing hidden layers of neural network, and introducing regularization terms as constraints [32]. Here, we develop a hierarchical multi-task learning structure on top of the methods of [33].

Given m similar tasks, each of them is denoted as task l, where $l = 1$, 2, 3, …, m. The target of multi-task learning is to estimate latent functions f_l for each task based on training data $D_l = (P_l, z_l)$, where $P_l \in \mathbb{R}^{n_l \times d}$ are the spatiotemporal locations of interest; $z_l \in \mathbb{R}^{n_l}$ are the corresponding measurement values, which is the surface height in anvil surface monitoring; n_l is the size of training data. In our application, dimension d is 3, because a spatiotemporal location is characterized by (t, x, y). $\cup P$ denotes the set of distinguished p_i in $\{D_l\}$, and $\cup P \in \mathbb{R}^{n \times d}$, where n is the size of distinctive training data for all tasks.

Fig. 2 illustrates the hierarchical structure for the Gaussian process multi-task learning. The estimated values in task l are obtained by the following steps:

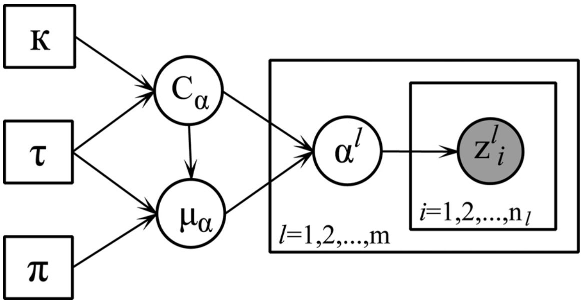


Fig. 2. Graphic model for hierarchical multi-task learning.

(10) μ_α, C_α are generated from

$$p(\mu_\alpha, C_\alpha) = N\left(\mu_\alpha \middle| 0, \frac{1}{\pi}C_\alpha\right)\mathrm{IW}\left(C_\alpha \middle| \tau, K^{-1}\right) \tag{10}$$

(11) For each task f_l

$$\alpha^l \sim N(\mu_\alpha, C_\alpha). \tag{11}$$

(12) The estimated function in task l is given by

$$Z_l(p) = \sum_{i=1}^{n} \alpha_i^l \kappa(p, p_i) + \varepsilon, \tag{12}$$

where K is an $\mathbb{R}^{n \times n}$ kernel matrix (also called Gram matrix), containing kernel $\kappa(p_i, p_j)$ of every input pair from $\cup P$, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$. In anvil surface monitoring, $\kappa(p_i, p_j)$ is replaced with the spatialtemporal kernel $\kappa_{st}(p_i, p_j)$ obtained in Section 2.1. ε is the output noise, following $\varepsilon \sim N(0, \sigma^2)$.

The rationale of sharing similarity among tasks is to assume that the α_l for each task l is sampled from the same multivariate Gaussian distribution as shown by Eq. (11), whose parameters are sampled from an upper layer of normal-inverse-Wishart distribution, which is given by Eq. (10).

In the above-mentioned framework, the model parameters are $\theta = \{\mu_\alpha, \mathrm{C}_\alpha, \sigma^2\}$, whose estimation can be achieved with an Expectation Maximization (EM) algorithm. Details of the EM algorithm are provided in Appendix I. Then a series of estimated α^l are plugged into Eq. (13) and the estimated measurement value at p_u is given by:

$$\widehat{Z}_l(p_u) = \sum_{i=1}^{n} \widehat{\alpha}_i^l \kappa_{\mathrm{st}}(p_u, p_i), \tag{13}$$

where p_u denotes the spatiotemporal location that we are interested in. $\kappa_{st}(p_u, p_i)$ is the kernel value between p_u and each p_i in $\cup P$.

The implementation of the above procedure is illustrated by Fig. 3. The hyperparameters δ_s^2, δ_t^2, τ, and π are predetermined and can be used to tune the STK-MTL model. Then, spatialtemporal kernels $\kappa_{st}(p_i, p_j)$ are formulated for every pair of training data. These kernels $\kappa_{st}(p_i, p_j)$ constitute the kernel matrices K and K_l, which are subsequently used in the EM algorithm. The mathematical formulations on kernel matrices and EM algorithm are detailed in the Appendix. The EM algorithm is used to find maximum-likelihood estimates for model parameters μ_α, C_α. Finally, the estimated μ_α is plugged in the Eq. (13) for predicting test data.

The complexity of the EM algorithm is O(kmn^3), where k is the number of iterations in EM. The computation could be more time-consuming with the increase of k, m, and n. Here, we suggest two ways to accelerate the algorithm.

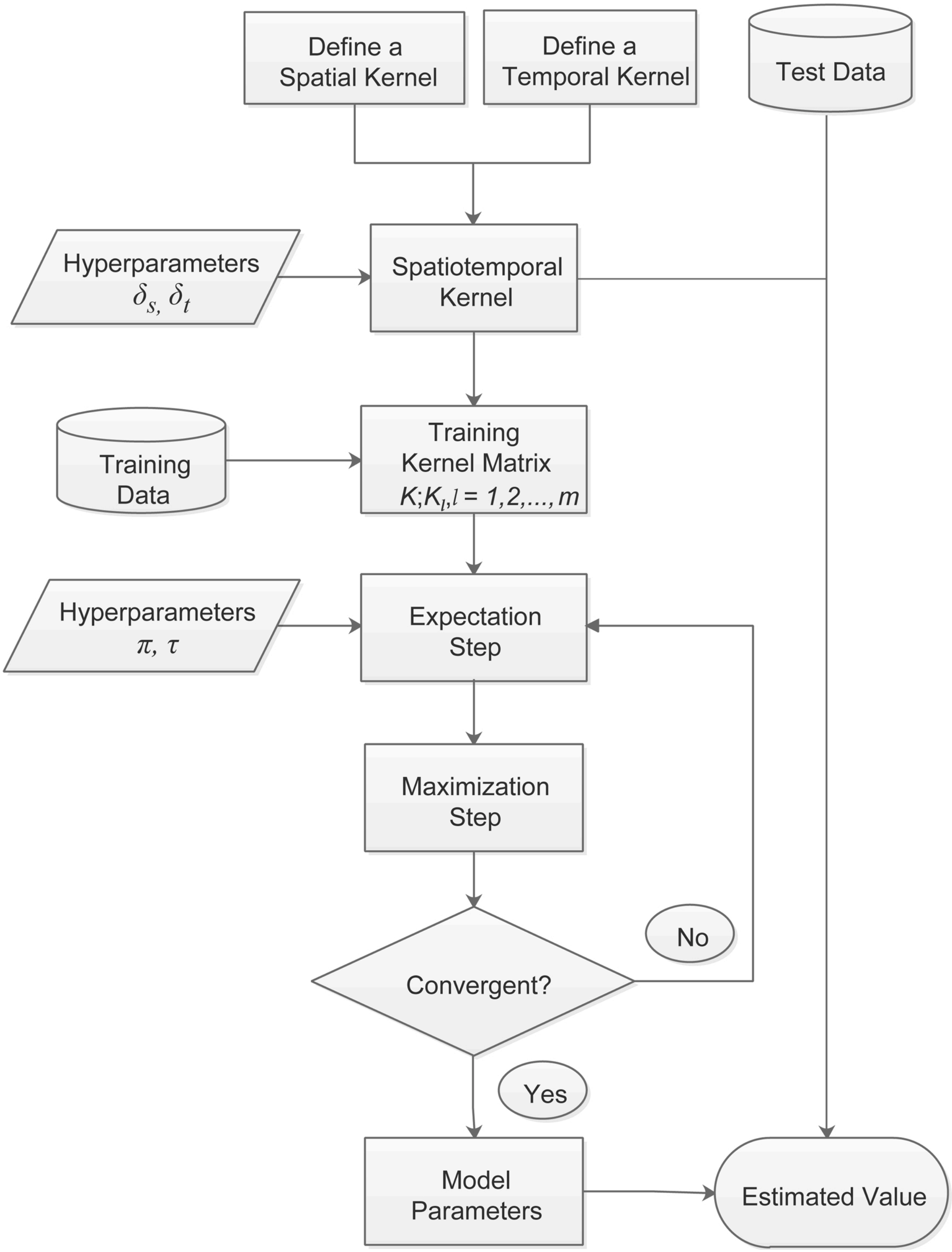


Fig. 3. Flowchart for the spatiotemporal kernel based multi-task learning.

- Use parallel LU decomposition in graphics processing unit or other multi-cores processors to accelerate matrix inversion [34] which contributes most complexity O(n^3) in the EM algorithm.
- Check the log-likelihood trace and reset initial conditions, when convergence cannot be achieved after k reaches a predetermined threshold.

There are four hyperparameters in the STK-MTL model, namely, δ_s^2, δ_t^2 from spatiotemporal kernel, and τ, π from the normal-inverse-Wishart distribution. The selection of these hyperparameters affects the prediction accuracy, which will be discussed in Section 4.

3. Case study

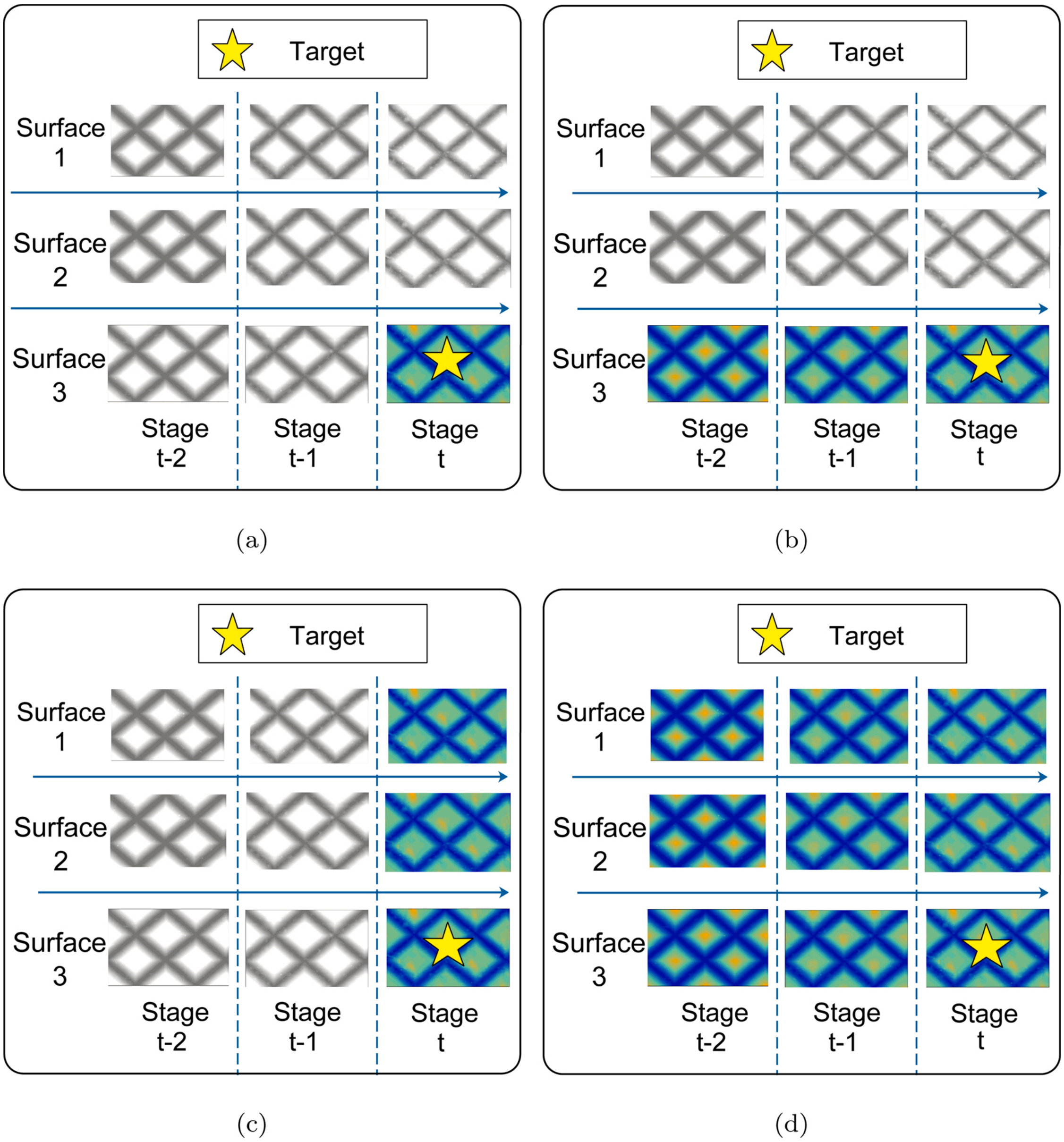
3.1. Data acquisition and experimental setup

Ultrasonic metal welding is a solid-state joining technique. A bonding between thin metal sheets clamped under pressure is created with oscillating shears generated by ultrasonic vibration [35]. It is well suited for various applications such as lithium-ion battery assembly [35, 3] and joining of hybrid heat exchangers [36–38]. However, ultrasonic metal welding has relatively large process variability because its quality is influenced by a variety of uncontrollable process conditions such as surface contamination [35,7] and tool degradation [1,2,4–6]. As such, quality monitoring for ultrasonic welding has been widely investigated [3,8,9,38]. Ultrasonic welding tools, including horn and anvil, are directly involved in the bonding formation mechanism and its geometry significantly affects the joining quality [5,1,6,7]. Therefore, the tool surface degradation is a major concern in the quality control of ultrasonic metal welding.

This case study aims to model the spatiotemporal progression of anvil surfaces in ultrasonic metal welding and compares the performance of our method with state-of-the-art modeling approaches. The anvil surface measurement data was obtained using Keyence VK-9700 laser scanning confocal microscope. The original dataset contains fine-scale measurement of three similar-but-not-identical anvil surfaces in consecutive time stages.

The missing measurement values at the target task are selected by random sampling, while data sampled out are used as testing data. In the case study, the sampling rate is set as 25%. For an anvil surface, this sampling strategy results in 405 training data points and 135 testing data points at each time stage. A detailed discussion about the applicability of the proposed method at different sampling rates can be found in Section 4. Considering that the prediction performance can fluctuate due to the randomness of sampling, the sampling-and-prediction process is repeated 10 times to obtain a range for prediction accuracy.

The prediction for missing measurement values is conducted over time, which means that STK-MTL is progressively applied, each time with previous stages as already measured. For example, when we are predicting surface height in stage t, the measurement data in earlier stages $1, 2, \ldots, t-2, t-1$ are available, while later stages $t+1, t+2, \ldots$ are treated as unknown. After finishing the stage t, we continue to predict stage $t+1$ with the same premise. In total, 16 consecutive stages are predicted in the case study.



(a)

(b)

(c)

(d)

Fig. 4. Comparison of four candidate methods with regard to fused data: (a) SK-STL, (b) STK-STL, (c) SK-MTL, (d) STK-MTL. The color and grey-scale images represent anvil surfaces that are used and unused for predicting the target surface, respectively. Target surfaces are indicated by yellow stars. Different colors in target surfaces indicate different height values and a similar color scale with that of Fig. 1 is used.

3.2. Performance comparison

The performance of the proposed STK-MTL method is compared with three other methods. The differences of these methods are summarized in Fig. 4 and Table 1. The setting of these methods aims to test the effectiveness of introducing temporal and similar-task information. A highlight of the differences is given below:

1. Spatial kernel based single-task learning (SK-STL) uses data from the target surface at the current time stage as shown in Fig. 4(a). In other words, neither spatiotemporal kernel nor multi-task learning are involved. This method is essentially the same as simple kriging, a common algorithm in geostatistics which models spatial variation using Gaussian process [28].
2. Spatiotemporal kernel based single-task learning (STK-STL) uses data from the target surface at recent three time stages as shown in Fig. 4(b). The key difference from SK-STL is that a spatiotemporal kernel is formulated by extending the spatial kernel with time domain. This method can also be referred as spatiotemporal Gaussian process.
3. Spatial kernel based multi-task learning (SK-MTL) introduced in [14] leverages data from multiple surfaces, but uses data from the current time stage only. This method is also known as multi-task Gaussian process learning. In our scenario, it essentially uses three surfaces at a certain time stage, as shown in Fig. 4(c).

Root mean squared error (RMSE), the definition of which is given by Eq. (14), is selected as an evaluation metric for prediction performance. Because RMSE varies in each run due to random sampling, we calculate the mean RMSE to measure the accuracy. Another important metric is standard deviation (STD) of RMSE during all repeated runs, which is given by Eq. (15). STD indicates the robustness of the prediction performance. A lower STD value indicates better robustness).

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{p\in P}\left(\widehat{Z}(p) - Z(p)\right)^2}{N}}, \tag{14}$$

where N denotes the size of P. Recall that P is the set of spatiotemporal location.

$$\mathrm{STD} = \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(\mathrm{RMSE}_i - \overline{\mathrm{RMSE}}\right)^2}, \tag{15}$$

where K is the total number of experiments repeated for each prediction method. Since we have repeated sampling-and-prediction process 10 times, K is 10 in this case.

Fig. 5 displays the comparison of four methods. In general, it is observed that the RMSEs of multi-task learning (including SK-MTL and STK-MTL) stay lower and more stable. In contrast, there are some outliers with significantly higher RMSEs in single-task learning (including SK-STL and STK-STL), which indicates their poor robustness and that they are influenced by sampling much more significantly. This evidence proves sharing information among similar tasks can be beneficial.

Fig. 6 compares the RMSE mean and standard deviation of the four methods. By comparing spatiotemporal-kernel methods with spatial-kernel ones, we can see from Fig. 6(a) that incorporating temporal information helps to reduce error of multi-task learning in every time stage, and it also benefits single-task learning in 14 of 16 time stages. Fig. 6(b) shows that including temporal information can reduce standard deviation of RMSE thus improving the modeling robustness. Among four methods, the proposed STK-MTL outperforms others in every time stage, with highest prediction accuracy and robustness.

Table 1
Methods summary for the case study.

Method	Transfer knowledge from other surfaces?	Use knowledge from historical measurement?
SK-STL	No	No
STK-STL	No	Yes
SK-MTL	Yes	No
STK-MTL	Yes	Yes

4. Discussion

This section discusses the following two issues: (A). Model applicability (with respect to data availability) and (B). Effects of hyperparameters.

4.1. Model applicability w.r.t. data availability

In order to reveal the applicable scope of STK-MTL, its performance is tested on different data availability. 40%–10% of original data are randomly sampled to simulate the missing data. Thus, the available training data in the target task vary from 60% to 90%. 10 runs are repeated for sampling and prediction. The prediction is made to estimate the missing data in the last time stage, which is stage 16, with the data from previous two stages and other two surfaces available for STK-MTL. The performance of STK-MTL is compared with other three methods, as introduced in Section 3.

The results are presented by Fig. 7, where Fig. 7(a) shows the average RMSE and Fig. 7(b) shows the STD of RMSE. It is seen that when available (measurement) data are limited, the performance of single-task learning (SK-STL and STK-STL) rapidly degrades, indicating that traditional STL becomes unreliable with limited measurement. Meanwhile, multi-task learning methods (SK-MTL and STK-MTL) outperform STL methods substantially. As more data in the target task are available, the prediction accuracy of single-task learning methods continuously improves, while the accuracy of multi-task learning methods is considerably stable. When the data availability in the target task exceeds 80%, the STK-STL method starts to outperform the STK-MTL method. In such cases, the training data is sufficient in the target task and STK-STL can better capture the specificity of the target task. This phenomenon indicates that multi-task learning is more advantageous in data-scarce scenarios. Model selection should be conducted in order to select the best-performing method.

Fig. 7(b) shows the standard deviation of RMSE for each setting. It is observed that spatiotemporal-kernel methods keep showing higher robust than spatial-kernel methods, and multi-task learning shows higher robust than single-task learning. Specifically, STK-MTL has the lowest standard deviation in all levels of data loss.

In short, the above experiment results lead to the following conclusions:

1. STK-MTL is applicable to a wide range of data loss scenarios and it brings many benefits especially in a severely data-scarce environment.
2. Bringing information from similar tasks can help to improve prediction accuracy and robustness, especially when limited data is available. However, multi-task learning should be carefully used when data loss is minor because the dissimilarity among tasks can manifest.

4.2. Effects of hyperparameters

There are two groups of hyperparameters in STK-MTL model: $\{\delta_s^2, \delta_t^2\}$ come from spatiotemporal kernel and $\{\pi, \tau\}$ come from normal-inverse-Wishart distribution in multi-task learning structure. In this section, experiments are conducted to study the effects of hyperparameters in these two groups.

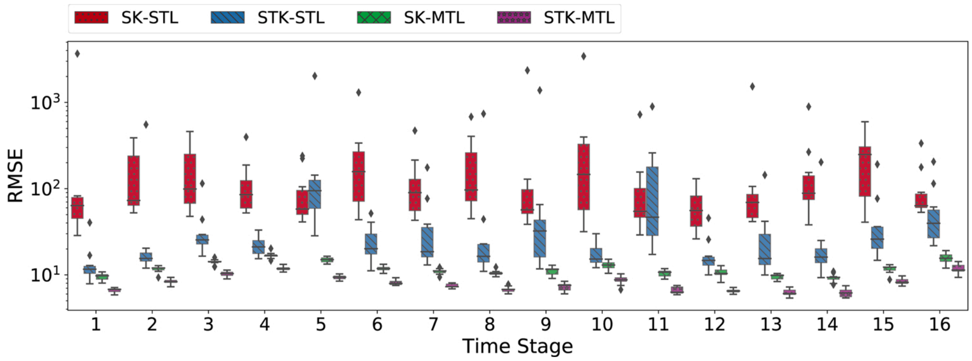
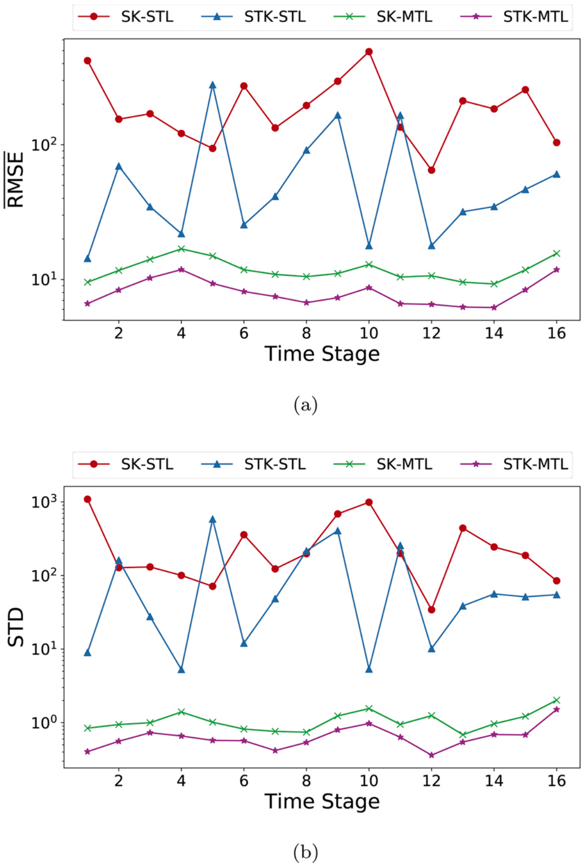


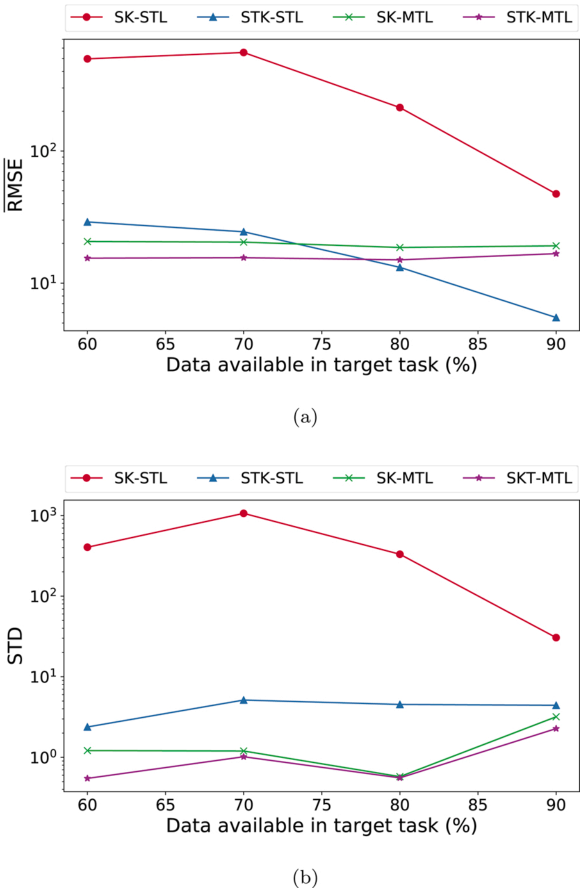
Fig. 5. RMSEs of four candidate methods in all time stages.



(a)

(b)

Fig. 6. Performance of four models in each time stage.



(a)

(b)

Fig. 7. Performance of the four models on different amount of training data.

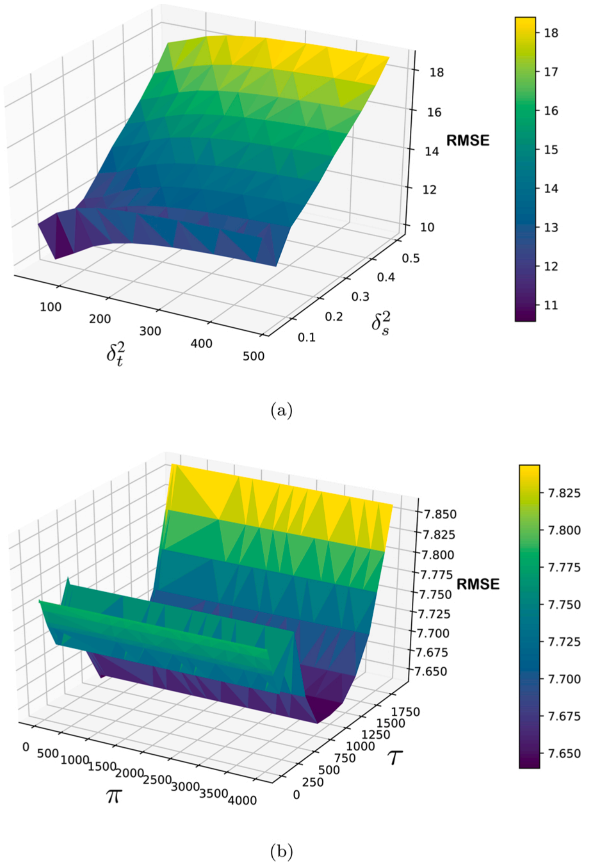
(1) Hyperparameters from spatiotemporal kernel: During the experiment, δ_s^2 varies from 0.01 to 0.5, and δ_t^2 varies from 0.01 to 500, while two other hyperparameters are set to be constant at $\pi = 1000$, $\tau = 0.01$. Fig. 8(a) displays the trend of RMSE with respect to δ_s^2 and δ_t^2. It is observed that with the increase of δ_s^2, the prediction error first sharply decreases and then increases. It can be explained by the fact that hyperparameter δ_s^2 scales the correlation for each data pair, thus learning too much nor too little from relevant locations could be harmful. Based on the experiment result, it is suggested to set δ_s^2 to be around 0.1, and δ_t^2 around 40 for ultrasonic anvil surface.

(2) Hyperparameters from multi-task learning structure: When investigating the effect of π, τ, the hyperparameters from kernel are fixed to be $\delta_s^2 = 0.1$, $\delta_t^2 = 40$. π varies from 0.01 to 4000, and τ varies from 0.01 to 1750. The corresponding RMSE is shown in Fig. 8(b). It is observed that with the increase of τ, prediction error first decreases and then increases, having minimum error at around $\tau = 1000$. Meanwhile, with the increase of π, prediction error first increases and then keeps stable. With τ settled as optimum at 1000, prediction error becomes lower when $\pi < 50$.

By comparing the two groups above, we can see that prediction accuracy is much more sensitive to the hyperparameters from the kernel function. This indicates that hyperparameters from kernel function should be more carefully chosen in order to fit a particular spatiotemporal process. In practice, they can be tuned based on training data with cross-validation.

4.3. Accelerated computing

To pinpoint the bottleneck of the STK-MTL algorithm, we first perform program profiling to obtain the breakdown of runtime, which is



(a)

(b)

Fig. 8. Effects of hyperparameters on prediction accuracy: (a) effects of (δ_s^2,δ_t^2) from kernel function, and (b) effects of (π,τ) from multi-task learning structure.

visualized in Fig. 9. Profiling is a common approach to locating the hotspot in a program [39]. In this experiment, the program is implemented in Python and evaluated on a laptop with Intel i7-9750H @ 2.60 GHz CPU and Nvidia GeForce GTX 1650 4GB GPU. To facilitate accurate analysis, trivial data pre-processing before the inference process is striped for runtime analysis.

As shown in Fig. 9, an inference on a target surface costs 986.9 seconds. Notably, the matrix inversion operation alone costs 804.5 seconds, accounting for over 81.5% the total computation time. This is because the matrix inversion is operated in every iteration of EM algorithm, and its complexity $O(n^3)$ grows polynomially as the matrix size increases. In our case study, the kernel matrix in a ~3400 × 3400 size leads to extensive computation. Particularly, the baseline matrix inversion using Gauss-Jordan Elimination is prohibitively expensive, as shown in Fig. 10. To achieve improved computing efficiency, we recommend using the LU decomposition and parallelizing the computation with pivoting on hypercubes [40]. These solutions can be realized in certain Python libraries such as NumPy and SciPy using multi-thread CPU programming. The runtime shown in Fig. 9 is based on NumPy library. The comparison between baseline and CPU parallel matrix inversion is presented in Fig. 10.

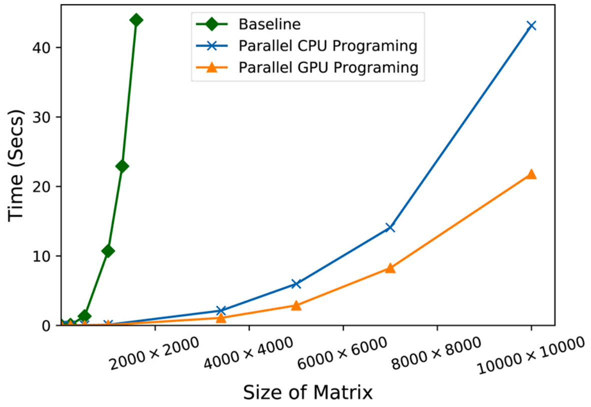


Fig. 10. Runtime of matrix inversion as the matrix size growing.

Apart from multi-thread CPU parallel programming, further improvements in computational efficiency can be accomplished by (1) using GPU with a large number of processors and (2) optimizing the number of iterations needed to achieve convergence. First, it is documented that when employing a large number of processors, GPU can be more efficient than CPU in large matrix operations [41]. We experiment the matrix inversion on GPU via PyTorch with CUDA platform, and the result is shown in Fig. 10. Compared with CPU-based parallel computing, the GPU programming achieves 1.7 times speedup for a 3400 × 3400 size matrix, and 2.0 times speedup for a 10000 × 10000 matrix. Second, the convergence of EM can be significantly slow in some instances, although it has a desirable monotonicity property [42]. One possible reason is that the likelihood may be stuck at ridge regions. To avoid these regions, we suggest reinitializing the parameters with random seeds when EM fails to converge. Fig. 11 demonstrates the effectiveness of these two acceleration approaches. The combination of GPU-based parallel programming and reinitialization achieves 1.84

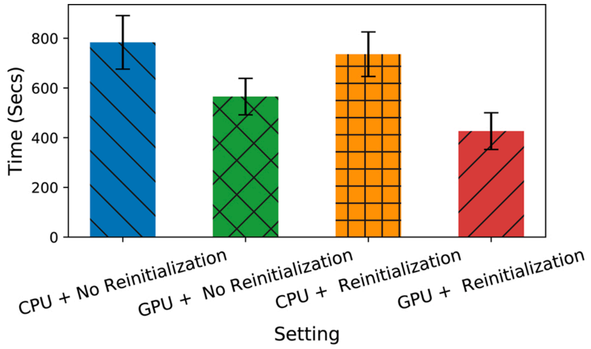


Fig. 11. Comparison of runtime on different acceleration settings. 20 experiments are repeated.

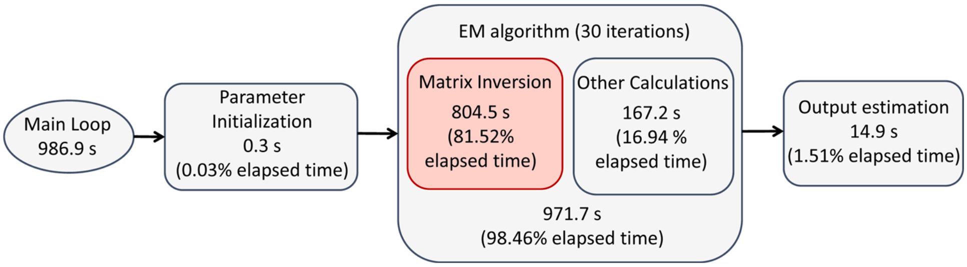


Fig. 9. Breakdown of the program’s runtime in a sample execution.

times speedup.

5. Conclusion

In order to cope with challenges brought by the high cost associated with fine-scale surface measurement, this paper develops a new spatiotemporal modeling method using multi-task learning. This allows a joint learning over multiple similar spatiotemporal processes. Moreover, a systematic framework is developed for the construction of the spatiotemporal kernel. In the case study, the proposed STK-MTL approach is applied to model the spatiotemporal progression of anvil surfaces in ultrasonic metal welding, and its performance is thoroughly tested and compared with three state-of-the-art approaches. Results show that the STK-MTL significantly outperforms others in terms of prediction accuracy and robustness. As such, the proposed approach is expected to improve the prediction accuracy under limited measurement, which can enable cost-effective measurement, modeling, and monitoring of spatiotemporal processes in industry. In addition, the effects of hyperparameters were systematically investigated. It is found that kernel parameters have a substantial effect, so they should be carefully selected prior to applying the STK-MTL method.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgment

This research has been supported by the National Science Foundation under Grant No. 1944345. We thank two anonymous reviewers whose comments helped clarify and improve this paper.

Appendix

Here, we provide the EM algorithm for estimating the parameters in hierarchical multi-task learning model. Detailed derivation is available in [33].

(16) Expectation (E-step): Estimate the expectation and covariance of α^l, $l = 1, 2, \ldots, m$, given the current Θ.

$$\widehat{\alpha}^l = \left(\frac{1}{\sigma^2}K_l^T K_l + C_\alpha^{-1}\right)^{-1}\left(\frac{1}{\sigma^2}K_l^T\eta_l + C_\alpha^{-1}\mu_\alpha\right) \tag{16}$$

$$C_{\alpha^l} = \left(\frac{1}{\sigma^2}K_l^T K_l + C_\alpha^{-1}\right)^{-1} \tag{17}$$

$$\text{where } K_l = \begin{bmatrix} \kappa(p_1,p_1) & \kappa(p_1,p_2) & \ldots & \kappa(p_1,p_n) \\ \kappa(p_2,p_1) & \kappa(p_2,p_2) & \ldots & \kappa(p_2,p_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(p_{n_l},p_1) & \kappa(p_{n_l},p_2) & \ldots & \kappa(p_{n_l},p_n) \end{bmatrix}.$$

(18) Maximization (M-step): optimize $\Theta = \{\mu_\alpha, C_\alpha, \sigma^2\}$

$$\mu_\alpha = \frac{1}{\pi + m}\sum_{l=1}^{m}\widehat{\alpha}^l \tag{18}$$

$$C_\alpha = \frac{1}{\tau + m} \times \left\{\pi\mu_\alpha\mu_\alpha^T + \tau K^{-1} + \sum_{l=1}^{m} C_{\alpha^l} + \sum_{l=1}^{m}[\widehat{\alpha}^l - \mu_\alpha][\widehat{\alpha} - \mu_\alpha]^T\right\} \tag{19}$$

$$\sigma^2 = \frac{1}{\sum_{l=1}^{m} n_l}\sum_{l=1}^{m}\|\eta_l - K_l\widehat{\alpha}_l\|^2 + \mathrm{tr}[K_l C_{\alpha^l} K_l^T] \tag{20}$$

$$\text{where } K = \begin{bmatrix} \kappa(p_1,p_1) & \kappa(p_1,p_2) & \ldots & \kappa(p_1,p_n) \\ \kappa(p_2,p_1) & \kappa(p_2,p_2) & \ldots & \kappa(p_2,p_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(p_n,p_1) & \kappa(p_n,p_2) & \ldots & \kappa(p_n,p_n) \end{bmatrix}, \text{ and } tr(\cdot) \text{ is the trace operator.}$$

As an approximation, Kronecker product may be used to simplify the construction of kernel matrix K. [43] models a kernel matrix K as the Kronecker product between two covariance matrices:

$$K = K^0 \otimes K^f, \tag{21}$$

where K^0 is the internal covariance matrix, which is the same for all tasks; and K^f is the external covariance matrix over all tasks, whose element measures correlation between each pair of tasks. However, because this approximation oversimplified each task and lose their specificities, we do not adopt this approximation in this paper.

References

[1] Shao C, Guo W, Kim TH, Jin JJ, Hu SJ, Spicer JP, Abell JA. Characterization and monitoring of tool wear in ultrasonic metal welding. 9th international workshop on microfactories 2014:161–9.

[2] Shao C, Jin JJ, Hu SJ. Dynamic sampling design for characterizing spatiotemporal processes in manufacturing. J Manuf Sci Eng 2017;139(10):101002.

[3] Wang B, Hu SJ, Sun L, Freiheit T. Intelligent welding system technologies: state-of-the-art review and perspectives. J Manuf Syst 2020;56:373–91.

[4] Yang Y, Zhang Y, Cai YD, Lu Q, Koric S, Shao C. Hierarchical measurement strategy for cost-effective interpolation of spatiotemporal data in manufacturing. J Manuf Syst 2019;53:159–68.

[5] Shao C, Kim TH, Hu SJ, Jin JJ, Abell JA, Spicer JP. Tool wear monitoring for ultrasonic metal welding of lithium-ion batteries. J Manuf Sci Eng 2016;138(5): 051005.

[6] Zerehsaz Y, Shao C, Jin J. Tool wear monitoring in ultrasonic welding using high-order decomposition. J Intell Manuf 2019;30(2):657–69.

[7] Nong L, Shao C, Kim TH, Hu SJ. Improving process robustness in ultrasonic metal welding of lithium-ion batteries. J Manuf Syst 2018;48:45–54.

[8] Shao C, Paynabar K, Kim TH, Jin JJ, Hu SJ, Spicer JP, Wang H, Abell JA. Feature selection for manufacturing process monitoring using cross-validation. J Manuf Syst 2013;32(4):550–5.

[9] Guo W, Shao C, Kim TH, Hu SJ, Jin JJ, Spicer JP, Wang H. Online process monitoring with near-zero misdetection for ultrasonic welding of lithium-ion batteries: an integration of univariate and multivariate methods. J Manuf Syst 2016;38:141–50.

[10] Suriano S, Wang H, Shao C, Hu SJ, Sekhar P. Progressive measurement and monitoring for multi-resolution data in surface manufacturing considering spatial and cross correlations. IIE Trans 2015;47(10):1033–52.

[11] Zhao C, Du S, Deng Y, Li G, Huang D. Circular and cylindrical profile monitoring considering spatial correlations. J Manuf Syst 2020;54:35–49.

[12] Suriano S, Wang H, Hu SJ. Sequential monitoring of surface spatial variation in automotive machining processes based on high definition metrology. J Manuf Syst 2012;31(1):8–14.

[13] Hao X, Li Y, Cheng Y, Liu C, Xu K, Tang K. A time-varying geometry modeling method for parts with deformation during machining process. J Manuf Syst 2020; 55:15–29.

[14] Shao C, Ren J, Wang H, Jin JJ, Hu SJ. Improving machined surface shape prediction by integrating multi-task learning with cutting force variation modeling. J Manuf Sci Eng 2017;139(1):011014.

[15] Zhou M, Guan H, Li C, Teng G, Ma L. An improved idw method for linear array 3d imaging sensor. 2017 ieee international geoscience and remote sensing symposium (IGARSS) 2017:3397–400.

[16] Gavriil K, Schiftner A, Pottmann H. Optimizing b-spline surfaces for developability and paneling architectural freeform surfaces. Comput Aided Des 2019;111:29–43.

[17] Noack S, Knobloch A, Etzold S, Barth A, Kallmeier E. Spatial predictive mapping using artificial neural networks. Int Arch Photogramm Remote Sens Spatial Inf Sci 2014;XL-2:79–86.

[18] Klauberg C, Hudak A, Bright B, Boschetti L, Dickinson M, Kremens R, Silva C. Use of ordinary kriging and gaussian conditional simulation to interpolate airborne fire radiative energy density estimates. IJWF 2018;27(4):228–40.

[19] Du S, Fei L. Co-kriging method for form error estimation incorporating condition variable measurements. J Manuf Sci Eng 2016;138(4):041003.

[20] Sales MHR, Souza CM, Kyriakidis PC. Fusion of modis images using kriging with external drift. IEEE Trans Geosci Remote Sens 2012;51(4):2250–9.

[21] Yang Y, Shao C. Spatial interpolation for periodic surfaces in manufacturing using a bessel additive variogram model. J Manuf Sci Eng 2018;140(6):061001.

[22] Liu JP, Jin R, Kong ZJ. Wafer quality monitoring using spatial dirichlet process based mixed-effect profile modeling scheme. J Manuf Syst 2018;48:21–32.

[23] Xia C, Pan Z, Polden J, Li H, Xu Y, Chen S, Zhang Y. A review on wire arc additive manufacturing: monitoring, control and a framework of automated system. J Manuf Syst 2020;57:31–45.

[24] Zhu L, Lu C, Kamwa I, Zeng H. Spatial-temporal feature learning in smart grids: A case study on short-term voltage stability assessment. IEEE Trans Ind Inf 2018.

[25] Meng XB, Li HX, Yang HD. Evolutionary design of spatio–temporal learning model for thermal distribution in lithium-ion batteries. IEEE Trans Ind Inf 2018;15(5): 2838–48.

[26] Babu M, Franciosa P, Ceglarek D. Spatio-temporal adaptive sampling for effective coverage measurement planning during quality inspection of free form surfaces using robotic 3d optical scanner. J Manuf Syst 2019;53:93–108.

[27] Li Q, Jiang H, Tang Q, Chen Y, Li J, Zhou J. Smart manufacturing standardization: reference model and standards framework. In: OTM confederated international conferences “on the move to meaningful internet systems”; 2016. p. 16–25.

[28] Christakos G. Modern spatiotemporal geostatistics, Vol. 6. Oxford University Press; 2000.

[29] Ak Ç, Ergönül Ö, Şencan İ, Torunoğlu MA, Gönen M. Spatiotemporal prediction of infectious diseases using structured gaussian processes with application to crimean-congo hemorrhagic fever. PLOS Negl Trop Dis 2018;12(8):e0006737.

[30] Schaback R. Limit problems for interpolation by analytic radial basis functions. J Comput Appl Math 2008;212(2):127–49.

[31] Shireen T, Shao C, Wang H, Li J, Zhang X, Li M. Iterative multi-task learning for time-series modeling of solar panel pv outputs. Appl Energy 2018;212:654–62.

[32] Evgeniou T, Pontil M. Regularized multi-task learning. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining; 2004. p. 109–17.

[33] Yu K, Tresp V, Schwaighofer A. Learning gaussian processes from multiple tasks. Proceedings of the 22nd international conference on machine learning (ICML-05) 2005:1012–9.

[34] Ezzatti P, Quintana-Orti ES, Remon A. High performance matrix inversion on a multi-core platform with several gpus. In: 2011 19th international euromicro conference on parallel, distributed and network-based processing; 2011. p. 87–93.

[35] Lee S Shawn, Shao C, Kim T Hyung, Hu S Jack, Kannatey-Asibu E, Cai WW, et al. Characterization of ultrasonic metal welding by correlating online sensor signals with weld attributes. J Manuf Sci Eng 2014;136(5).

[36] Meng Y, Peng D, Nazir Q, Kuntumalla G, Rajagopal MC, Chang HC, Zhao H, Sundar S, Ferreira PM, Sinha S, Miljkovic N, Salapaka SM, Shao C. Ultrasonic welding of soft polymer and metal: a preliminary study. International manufacturing science and engineering conference, Vol. 58752 2019. p. V002T03A083.

[37] Kuntumalla G, Meng Y, Rajagopal M, Toro R, Zhao H, Chang HC, Sundar S, Salapaka S, Miljkovic N, Shao C, Ferreira P, Sinha S. Joining techniques for novel metal polymer hybrid heat exchangers. In: ASME international mechanical engineering congress and exposition, Vol. 59384; 2019. p. V02BT02A018.

[38] Meng Y, Rajagopal M, Kuntumalla G, Toro R, Zhao H, Chang HC, Sundar S, Salapaka S, Miljkovic N, Ferreira P, Sinha S, Shao C. Multi-objective optimization of peel and shear strengths in ultrasonic metal welding using machine learning-based response surface methodology. Math Biosci Eng 2020;17(6):7411–27.

[39] Yang Y, Cai YD, Lu Q, Zhang Y, Koric S, Shao C. High-performance computing based big data analytics for smart manufacturing. In: ASME 2018 13th international manufacturing science and engineering conference; 2018. p. V003T02A013.

[40] Liu Z, Cheung D. Efficient parallel algorithm for dense matrix lu decomposition with pivoting on hypercubes. Computers & Mathematics with Appl 1997;33(8): 39–50. https://doi.org/10.1016/S0898-1221(97)00052-7. URL http://www.sciencedirect.com/science/article/pii/S0898122197000527.

[41] Ezzatti P, Quintana-Ortí ES, Remón A. Using graphics processors to accelerate the computation of the matrix inverse. J Supercomput 2011;58(3):429–37.

[42] L. E. Ortiz, L.P. Kaelbling, Accelerating em: An empirical study, arXiv preprint 2013. arXiv:1301.6730.

[43] Bonilla EV, Chai KM, Williams C. Multi-task gaussian process prediction. Advances in neural information processing systems. 2008. p. 153–60.