A Data-Driven Approach to Prediction and Optimal Bucket-Filling Control for Autonomous Excavators

Ryan J. Sandzimier and H. Harry Asada, Member, IEEE

Abstract—We develop a data-driven, statistical control method for autonomous excavators. Interactions between soil and an excavator bucket are highly complex and nonlinear, making traditional physical modeling difficult to use for real-time control. Here, we propose a data-driven method, exploiting data obtained from laboratory tests. We use the data to construct a nonlinear, non-parametric statistical model for predicting the behavior of soil scooped by an excavator bucket. The prediction model is built for controlling the amount of soil collected with a bucket. An excavator collects soil by dragging the bucket along the soil surface and scooping the soil by rotating the bucket. It is important to switch from the drag phase to the scoop phase with the correct timing to ensure an appropriate amount of soil has accumulated in front of the bucket. We model the process as a heteroscedastic Gaussian process based on the observation that the variance of the collected soil mass depends on the scooping trajectory, i.e., the input, as well as the shape of the soil surface immediately prior to scooping. We develop an optimal control algorithm for switching from the drag phase to the scoop phase at an appropriate time and for generating a scoop trajectory to capture a desired amount of soil with high confidence. We implement the method on a robotic excavator. Experiments show promising results.

Index Terms—Data-driven control, Gaussian process, robotics in construction, mining robotics, field robots, model learning for control, optimization and optimal control, probability and statistical methods

I. INTRODUCTION

There is a growing need in the construction and mining industries for autonomous excavators that can operate excavators autonomously with increased productivity and fuel efficiency [1]. The worldwide shortage of skilled workers to operate these excavators is a major driver behind the development of intelligent excavators for performing various earthmoving tasks.

While loading a dump truck with soil, an experienced operator can fully load the truck in just a few excavation cycles without overloading the truck. This requires the operator to collect a desired amount of soil in the bucket in a limited amount of time. Unskilled operators are unable to precisely fill the bucket, resulting in additional loading cycles and underloading or overloading of the truck. Thus, the amount of soil collected in the bucket, referred to as the bucket fill factor, is a fundamental metric for quantifying productivity and operator skill level [2]. Cycle time is another important factor for maximizing productivity. To reduce cycle time and increase fuel efficiency, experienced operators often only fill the bucket to 80% of capacity [1]. Scooping a desired amount of soil is thus a critical requirement for autonomous excavators.

Early work by Bernold [3] proposed force feedback and impedance control as an effective method for controlling the path of the bucket as it drags through soil. His work focuses mainly on control of the bucket during the drag phase. It does not consider directly controlling the bucket fill factor, where the scoop phase plays a crucial role. Singh and Simmons [4] approached the problem from a task planning perspective. Placing constraints on the allowable actions of the excavator and forces at the bucket, they choose actions that satisfy a bound on a cost criterion. For our approach, the proposed cost criterion to be maximized is related to the bucket fill factor.

Bucket-soil interactions are very complex and difficult to model. Singh and Simmons [4] used a simple heuristic soil model proposed by Homma et. al [5]. Attempts at modeling these interactions using discrete-element method [6] are met with computational challenges that make these models infeasible for real-time applications. Singh [7] developed a method for predicting resistive forces using the Fundamental Equation of Earthmoving [8]. Luengo et. al [9] built on this by determining soil parameters on-line using force measurements.

Terramechanic modeling based on fundamental physical principles is, in general, difficult to use for real-time control due to computational complexity. Model validity is also limited due to unmodeled dynamics. Furthermore, the use
of complex feedback control with sophisticated sensors and instrumentation technology is not practical, considering the harsh environment where excavators have to work. In exploring an alternative approach and a new methodology, we exploit the data. Excavation consists mostly of repetitive operations. Although the operations are performed under diverse conditions, we can obtain a large amount of data from both laboratory tests and field operations to handle these conditions. This allows us to use the data for intelligent control of excavators. In a statistical modeling framework, we can deal with highly nonlinear, distributed behaviors of soil without going through terramechanics-based parametric representations. We obtain a non-parametric, nonlinear model directly from the data. It is possible to derive novel control methods from the statistical model.

We aim to apply this data-driven, statistical model and control method to bucket fill factor control. The new method takes into account both the expected bucket fill factor as well as the associated variance so we can control the excavator to collect a desired amount of soil with high confidence. Through the estimation of the variance, the data-driven approach can produce a reliable solution to autonomous operation in an unstructured environments under diverse conditions.

II. PROBLEM STATEMENT

A single excavation cycle consists of three sequential phases: penetrate, drag, and scoop. Structuring the problem in this way is common practice and breaks the excavation cycle into manageable sub-tasks. First, the bucket penetrates the soil to reach a desirable depth. Next, the bucket moves forward in a dragging motion to accumulate soil inside and in front of the bucket. While operators do make depth adjustments during the drag phase, the bucket generally moves at a constant depth forward toward the base of the excavator. When a desirable amount of soil has accumulated, the excavator scoops the soil by rotating and translating the bucket to collect the desired amount of soil. Fig. 2 illustrates these phases.

The final conditions after the penetrate and drag phases serve as the initial conditions for the scoop phase. While the scoop phase occurs last chronologically, we choose to focus on it first to better motivate the discussion about the preceding phases later. An excavator can scoop a certain amount of soil by controlling the bucket motion in the scoop phase. However, the amount of scooped soil highly depends on the accumulated soil within the bucket and in front of the bucket just before the scooping action begins. Therefore, the shape of the accumulated soil both inside and in front of the bucket is a critical factor in predicting the bucket fill factor. In the current work, we describe the soil profile with a set of soil state variables, \( s \in \mathbb{R}^{n_s} \). As detailed later, we measure the accumulated soil with a depth camera and compress the image data to a representation using a small number of variables.

The input command to the bucket is described with a vector \( u \in \mathbb{R}^{n_u} \) consisting of the rotational as well as translational movements of the bucket. In the current work, a single input command defines the entire trajectory of the bucket throughout the scoop phase. The soil profile at the beginning of the scoop phase, \( s \), and the bucket input command, \( u \), are the two sets of variables that determine the bucket fill factor at the end of the scooping phase. Collectively, these variables are combined as \( x = [s, u] \in \mathbb{R}^n \), where \( n = n_s + n_u \). The resultant bucket fill factor at the end of scooping is denoted by output \( y \).

The bucket fill factor \( y \) is stochastic. We treat the scoop phase as a stochastic process where the inputs are the combined soil state and bucket command variables, \( x \), and the output is random variable \( y \). The bucket fill factor \( y \) has a probability density conditioned on the soil profile and bucket command, \( x \).

Fig. 3 illustrates three examples of typical soil states. In the top case, not enough soil has accumulated. The bucket fill factor \( y \) is likely to be too low if we initiate the scoop phase immediately. The bottom case is likely to overfill the bucket. In each case, the soil profile immediately prior to
scooping serves as the initial conditions for the scoop phase. As the excavator drags the bucket through the soil, the soil profile continually changes. The excavator controller makes a decision when to initiate the scoop phase by observing the soil profile throughout the drag phase.

To make this control decision, we must construct a prediction model that can predict the bucket fill factor $y_s$ in response to the current soil profile $s_s$ and a bucket control command $u_s$. If the predicted bucket fill factor is significantly lower than its desired value, the drag phase should continue in hopes of achieving a more desirable $s_s$ in the future. To this end, we must be able to make predictions about how the soil state will change if the drag action continues. We represent the soil state during the drag phase with a vector $\xi$ using the same soil state parameterization as before. We represent the drag action as a vector $v$ where the elements are the parameter values of the drag action. To condense our notation, we refer to $\boldsymbol{\xi} = [\xi', v]$ as the input vector for the drag phase. Using $\xi_* = [\xi', v]$ to represent a new test input, we want to make predictions about the corresponding test output $s_*$. Using such a prediction model, we can make decisions about whether it is more desirable to initiate the scoop phase using the current soil state or to continue with another drag action and wait to initiate the scoop phase in the future.

III. PREDICTION MODEL

We use a data-driven approach to learn a prediction model. That is, we collect a training data set $\mathcal{D}_s = \{(x_i, y_i)\}_{i=1}^{N}$ consisting of pairs of inputs and outputs for $N$ scoop trials. We model $y$ as the sum of some unknown latent function $f(x)$ and additive independent noise.

$$y(x_i) = f(x_i) + \epsilon_i$$

Note that the additive noise $\epsilon_i$ depends on input $x_i$ because of the heteroscedastic nature of scooping soil, as discussed previously. We will examine and verify this assumption of heteroscedasticity later in Section V-C.

Given the current test input $x_*$ and the training data set $\mathcal{D}_s$, we must predict the probability distribution $P(y_* | x_*, \mathcal{D}_s)$ for the corresponding bucket fill factor $y_*$. Gaussian processes are an effective framework for making this type of prediction. Unlike traditional Gaussian processes [10], which assume homoscedasticity, our process is heteroscedastic. Goldberg et al. [11] dealt with heteroscedastic Gaussian process regression (HGPR) using Markov chain Monte Carlo (MCMC) approximate inference. While effective, MCMC is very slow compared to traditional Gaussian process regression. Lázaro-Gredilla and Titsias [12] developed an effective method called variational HGPR (VHGPR) using a variational approximation which maximizes an analytically tractable lower bound on the maximum likelihood. Lázaro-Gredilla and Titsias have shown that the performance of VHGP is similar to that of HGPR using MCMC and is comparable in speed to homoscedastic Gaussian process regression. We find VHGP used for other robotics applications, such as planar pushing [13].

Under the VHGP framework, we place a Gaussian process prior on the latent function $f(x)$ and Gaussian priors on the noise terms $\epsilon_i$.

$$f(x) \sim \mathcal{GP}(0, k_f(x, x'))$$

$$\epsilon_i \sim \mathcal{N}(0, e^\Theta(x_i))$$

where $g(x_i)$ is the log-variance at every input $x_i$. We also place a Gaussian process prior on $g(x)$.

$$g(x) \sim \mathcal{GP}(\mu_0, k_g(x, x'))$$

where $k_f(x, x')$ and $k_g(x, x')$ are covariance functions and $\mu_0$ is the noise mean hyperparameter. In practice, we use the automatic relevance determination squared exponential (ARD-SE) kernel for both $k_f(x, x')$ and $k_g(x, x')$ have hyperparameters $\Theta_f$ and $\Theta_g$, respectively. The predictive distribution for $y_*$ according to the VHGP [12] framework is:

$$P(y_* | x_*, \mathcal{D}_s) = \int \mathcal{N}(y_* | a_s, c^2_s + \sigma^2_g) \times \mathcal{N}(y_0 | \mu_s, \sigma^2_s) dg_*$$

$$a_s = k_f^T_s(K_f + R)^{-1} y$$

$$c^2_s = k_{fss}^T - k_{fs}^T(K_f + R)^{-1} k_{fs}$$

$$\mu_s = k_{gs}^T(\Lambda - \frac{1}{2} I) + \mu_0$$

$$\sigma^2_s = k_{gss} - k_{gs}^T(K_g + \Lambda^{-1})^{-1} k_{gs}$$

$$\mu = K_g^T(\Lambda - \frac{1}{2} I) + \mu_0$$

$$\Sigma^{-1} = K_g^{-1} + \Lambda$$

where $y$ is a column vector with elements $[y_i] = y_i$. $K_f$ is a matrix with elements $[K_f]_{ij} = k_f(x_i, x_j)$, $k_{fss}$ is a column vector with elements $[k_{fss}] = k_f(x_s, x_*)$, and $k_{fss} = k_f(x_s, x_s)$ for covariance function $k_f(x, x')$. We construct $K_g, k_{gs}, k_{gss}$ in the same way using covariance function $k_g(x, x')$. $R$ is a diagonal matrix with elements $[R]_{ii} = e^g(x_i)$ where $g(x_i) = |\mu_i| + |\Sigma_i|/2$. $\Lambda$ is a positive semidefinite diagonal matrix whose elements represent the free parameters in $\mu$ and $\Sigma$. We optimize the hyperparameters $\Lambda, \mu_0, \Theta_f$, and $\Theta_g$ simultaneously to maximize the variational bound.

The predictive distribution in (6) is not analytically tractable. However, its mean and variance are computable analytically.

$$E[y_* | x_*, \mathcal{D}_s] = a_s$$

$$\mathbb{V}[y_* | x_*, \mathcal{D}_s] = c^2_s + e^{\mu_s} + \sigma^2_g/2$$
We can use this same framework for making predictions about the soil state \( s \) throughout the drag phase. Specifically, given a drag test input \( \xi \), we want to make predictions about each component \( \{s_i\}_j \) of the corresponding output \( s \). In general, the soil state \( s \) is vector-valued. Gaussian process regression is traditionally formulated only for scalar outputs except for some work [14] that formulates homoscedastic Gaussian processes with multiple dependent outputs. To our knowledge, there is no prior work successfully handling Gaussian processes with multiple dependent outputs. To make accurate predictions, we do lose information about potential correlation among outputs. However, we show in Section V that even with this loss of information we can still make appropriate predictions.

We can make use of (1)-(14) by substituting \( \xi \) for \( x \), \( D_d = \{\{\xi_i, s_i\}\}_{i=1}^N \) for \( D_s \), and each \( \{s_i\}_j \) for \( y \) and independently optimizing the hyperparameters for each VHGPR model.

IV. OPTIMAL CONTROL

Next, we use the prediction models to optimize our choice of action. First, we focus on the scoop phase. Given the current soil state \( s \) that we want to test, we choose a trajectory \( u_{opt} \) that minimizes a cost function \( C_s(s_*, u_*) \).

\[
u_{opt} = \arg \min_{u_*} C_s(s_*, u_*) \tag{15}\]

The goal of the scoop phase is to achieve a specified bucket fill factor. A natural choice of cost function is the expected squared error between the bucket fill factor \( y \) and some desired bucket fill factor \( y_d \).

\[
C_s(s_*, u_*) = \mathbb{E}[(y_zaucshp) - y_d]^2 | x_*, D_s] \tag{16}
\]

Expanding (16) and using the relationship between the second moment, variance, and mean: \( \mathbb{E}[y_*^2 | x_*, D_s] = \mathbb{V}[y_* | x_*, D_s] + \mathbb{E}[y_* | x_*, D_s]^2 \) yields:

\[
C_s(s_*, u_*) = \mathbb{V}[y_* | x_*, D_s] + (\mathbb{E}[y_* | x_*, D_s] - y_d)^2 \tag{17}
\]

From (13) and (14), we have closed-form expressions for the mean and variance that we can plug into (17).

\[
C_s(s_*, u_*) = c_*^2 + e^{\mu_* + \sigma_*^2/2} + (a_* - y_d)^2 \tag{18}
\]

To minimize the cost function, we take the derivative with respect to each of the components of the action and use gradient descent methods to search for a minimum. Taking the derivative with respect to \( [x_*]_j \), the \( j \)th component of \( x_* \):

\[
\frac{\partial C_s(s_*, u_*)}{\partial [x_*]_j} = \frac{\partial c_*^2}{\partial [x_*]_j} + 2\frac{\partial a_*}{\partial [x_*]_j}(a_* - y_d) + \left( \frac{\partial \mu_*}{\partial [x_*]_j} + \frac{\partial \sigma_*^2}{2} \right) \frac{e^{\mu_* + \sigma_*^2/2}}{\partial [x_*]_j} \tag{19}
\]

Using gradient descent methods, we search for the optimal scoop action \( u_{opt} \) and the corresponding cost. Given a current test soil state \( s_* \) and a desired bucket fill factor \( y_d \), we choose an optimal scoop action. However, if the soil state at the start of the scoop phase is undesirable, the cost associated with the optimal scoop may still be high. Therefore, we next consider how to decide if it is better to initiate the scoop phase with the current soil state or to continue the drag action in hopes of making the soil state more desirable prior to scooping.

Given a drag input \( \xi \), we can predict the distribution of the corresponding soil state output \( s_* \). Rewriting (6) in terms of the of the drag phase inputs and outputs:

\[
P(\{s_*\}_j | \xi_*, D_d) = \int \mathcal{N}(\{s_*\}_j | a_*c_*^2 + e^{\mu_*}) \times \mathcal{N}(a_*|c_*^2, \sigma_*^2)dg_* \tag{26}
\]

which is analytically intractable. However, we can approximate the distribution using Gauss-Hermite quadrature [12], which is computationally inexpensive.

Using the assumption that the components of the predicted soil state \( s_* \) are statistically independent, we can calculate the predicted distribution of the full soil state

\[
P(s_* | \xi_*, D_d) = \prod_{j=1}^{n_s} P(\{s_*\}_j | \xi_*, D_d) \tag{27}
\]

where \( s_* \in \mathbb{R}^{n_s} \).

In order to make a decision about whether to initiate the scoop phase or continue with another drag action, we must
compare the cost to initiate the scoop phase now to the expected cost to initiate the scoop phase in the future. The cost to continue dragging for some drag input $\xi_s$ is:

$$C_d(\xi_s) = \mathbb{E}[C_s(s_s, u_{opt})|\xi_s, D_d]$$

$$= \int C_s(s_s, u_{opt})P(s_s|\xi_s, D_d)ds_s$$

(28)

(29)

This integral is not analytically tractable. However, we can approximate this expected cost using importance sampling. While it is difficult to sample directly from $P(s_s|\xi_s, D_d)$, we can approximate the probability density at a given sample $s_s$. By sampling from some distribution $q(s_s)$ that is easy to sample from, we can approximate the expectation as follows:

$$C_d(\xi_s) \approx \sum_{i=1}^{k} \frac{w(s_{s,i})C_s(s_{s,i}; u_{opt})}{\sum_{i'=1}^{k} w(s_{s,i'})}$$

(30)

where

$$w(s_{s,i}) = \frac{P(s_{s,i}|\xi_s, D_d)}{q(s_{s,i})}$$

(31)

By taking this $k$-sample weighted average of the optimal scoop cost, we approximate the cost to continue dragging. We compare the cost to continue dragging for every possible drag action $v$ and compare it to the cost to initiate the scoop phase now. If the cost to initiate scooping is smaller for every choice of $v$, we decide to initiate the scoop phase now. Otherwise, we continue dragging by performing one step of a drag action. We then repeat these steps until the scoop phase is initiated.

V. PERFORMANCE AND EXPERIMENTAL RESULTS

In this section, we discuss our experimental implementation of the proposed method and present the experimental results. In addition, we address the assumption of heteroscedasticity in more detail.

A. Experimental Setup

The experimental setup is shown in Fig. 4. We use a Universal Robotics UR10e to control the excavator bucket. The robot is equipped with a 6-axis force/torque sensor. We use this sensor to measure the mass of soil collected in the bucket after scooping (bucket fill factor). Throughout the dragging phase and immediately prior to scooping, a stereo camera captures a depth image which we use to represent the soil state. The soil medium is homogeneous, low-density, and has particles between 1-3mm in size.

B. Parameterization of Soil State and Actions

To represent the soil state, we capture a depth image using a stereo camera. After isolating pixels corresponding to the soil inside and in front of the bucket, we project the depth image onto an orthogonal plane (top). This new depth map contains pixels representing the depth of the soil relative to the bucket center. Although the image is high-dimensional due to the large number of pixels, we use principal component analysis and find that the first two principal components explain 96% of the variance in the depth images. We then represent the soil state $s \in \mathbb{R}^2$ as a vector with the projection of the depth image onto the first two principal component directions.

To represent the scoop actions, we parameterize the allowable scoop trajectories the bucket follows during scooping. This parameterization is illustrated in Fig. 5. During the scoop phase, the bucket rotates about the bucket center to collect the soil. While rotating, we allow the bucket center to follow a parabolic trajectory, which takes three parameters to represent. We find that this parameterization is simple enough to keep the input space relatively small, while also rich enough to allow a sufficient range of scoop actions for controlling the bucket fill factor. We control the bucket along this reference trajectory using impedance control to account for high reaction forces at the bucket that may not allow the robot to follow the reference trajectory exactly.

While operators sometimes vary the depth of the bucket throughout the drag, these variations are typically small relative to the forward motion of the bucket. Therefore, we prescribe a constant depth during the drag phase to reduce the dimensionality of the drag action parameterization. The only
parameter necessary to represent a drag action is the drag length, or the distance to move the bucket in the forward direction. The maximum allowable drag length is limited by the workspace of the excavator. When optimizing the transition point between the drag phase and scoop phase, we search over the remaining possible drag lengths.

C. Demonstration of Heteroscedasticity

We observe that the variance in the bucket fill factor is dependent on the initial soil state and the chosen scoop action. Fig. 6 illustrates this heteroscedastic property. By measuring the bucket fill factor for repeated trials using the same initial soil state and scoop action, we can test for homoscedasticity. While setting up the exact same initial soil state for repeated trials is not practical, we can achieve approximately the same initial soil state by repeatedly performing a drag trajectory. This is an approximation because repeating a drag trajectory does not guarantee the same soil state. However, we find the approximation is appropriate.

We measure the bucket fill factor for a combination of two separate drag actions and two separate scoop actions. For one of the drag actions, the bucket penetrates deep into the soil and drags for a long length. We refer to this as the long drag. For the other drag action, which we refer to as the short drag, the bucket penetrates to a shallow depth and drags for a short length. During a long drag, a lot of soil accumulates inside and in front of the bucket. On the other hand, for a short drag, the bucket barely moves any soil. We refer to the two choices of scoop action as the deep scoop and the shallow scoop. For the deep scoop, the bucket moves deep into the soil while also moving forward to collect soil. For the shallow scoop, the bucket only rotates. There is no forward or downward movement during the shallow scoop. Referring to the notation from Fig. 5, the deep scoop has maximum values for $u_1$ and $u_2$ and $u_3$ set to zero, while the shallow scoop has parameters $u_1$, $u_2$, and $u_3$ all set to zero.

We measure the bucket fill factor for four groups: long drag and deep scoop (LD-DS), long drag and shallow scoop (LD-SS), short drag and deep scoop (SD-DS), and short drag and shallow scoop (SD-SS). The null hypothesis is that the bucket fill factor samples came from populations which each have the same variance. We use an F-test to test this hypothesis for each pair of groups. Table I shows the \( p \)-values for each of these tests. We determine with statistical significance that the bucket fill factor is heteroscedastic.

D. Training the Model

We collect training data sets $D_s$ and $D_d$ by performing random drag and scoop actions, measuring the soil state throughout the drag phase and immediately prior to the scoop phase, and measuring the resulting bucket fill factor. The use of random drag and scoop actions provides a rich set of soil states, ensuring the training data set spans the entire input space.

The prediction models depend on the training data sets and the hyperparameters $\Theta_f$ and $\Theta_g$. In general, we do not know the best choice of hyperparameters, so we must infer them from the training data. We find the hyperparameters that maximize the variational bound on the log-likelihood [12] using the L-BFGS-B optimization algorithm [15].

We set aside a testing data set to cross-validate the trained prediction models. We use the normalized mean squared error (NMSE) and normalized log-probability density (NLPD) as performance measures. The NMSE and NLPD are defined as follows:

\[
NMSE = \frac{\sum_{j=1}^{M}(y_{*,j} - \bar{a}_{*,j})^2}{\sum_{j=1}^{M}(\bar{y} - \bar{y}_*)^2} \tag{32}
\]

\[
NLPD = -\frac{1}{M} \sum_{j=1}^{M} \log(p(y_{*,j}|x_{*,j}, D_s)) \tag{33}
\]

where $x_{*,j}$ and $y_{*,j}$ are the \( j \)th test input and output, respectively, $M$ is the number of test inputs and outputs, $a_{*,j}$ is the mean of the predictive distribution corresponding to the \( j \)th test input, and $\bar{y}$ is the mean of the training data outputs.

Fig. 7 and 8 show how the performance measures change with varying training data set sizes for the bucket fill factor prediction model and soil state prediction models, respectively. We find that the performance saturates around 250 training data points for the bucket fill factor prediction model and around 1000 training data points for the soil state prediction models. We observe that the second component of the soil state requires more training data points to saturate than the first component and has a higher NMSE. Since the components of the soil state represent the principal
Fig. 7. Normalized mean squared error (NMSE) and normalized log-probability density (NLPD) for the bucket fill factor prediction model.

Fig. 8. Normalized mean squared error (NMSE) and normalized log-probability density (NLPD) for the soil state prediction models.

Fig. 9. Predicted cost and actual cost throughout a single drag phase. In this example, the desired bucket fill factor was 1.0kg. The plots show the predicted cost at the beginning of the drag phase (top), after executing one drag action (middle), and after executing a second drag action (bottom). In addition to the predicted cost, each plot also shows the actual cost we measured retroactively (circles) and the actual costs from previous time steps (diamonds). After the second drag action, the scoop phase initiates because the model predicts the cost will increase for all remaining choices of drag action.

E. Performance

To evaluate the performance of the drag transition decision-making, we measure the cost to initiate the scoop phase throughout the drag phase. Using the prediction model, we decide the optimal transition point. We can then compare this transition point to the actual optimal transition point, which we determine retroactively. Fig. 9 shows these results for a single drag phase. We find that the predicted cost is relatively close to the actual measured cost. For this example, we decide to initiate the scoop phase after performing two drag actions because the predicted cost increases for all remaining choices of drag actions. We illustrate a single drag phase as an example, however, we find these results to be components, it makes sense that the second component would be more difficult to train.
consistent and typical for any choice of target bucket fill factor.

Next, we discuss the performance while controlling the bucket fill factor. We run repeated trials using different target bucket fill factors. We vary the drag depth and target bucket fill factor for each trial and the controller must decide when to transition to the scoop phase and which scoop action to perform to minimize the cost. The results of this experiment are shown in Fig. 10. As we increase the desired bucket fill factor, the experimentally measured bucket fill factor increases and has relatively low variance.

VI. CONCLUSION

The experimental results are promising and demonstrate that a data-driven model can be suitable for making predictions about soil-bucket interactions and controlling the bucket fill factor. We present a specific parameterization of the problem, however, the approach extends to other parameterizations if provided an appropriately sized training data set.

There are some key areas for future work on this problem. For the experiment, we used soil of uniform grain size. For real world applications, it is important to consider various soil grain sizes and properties. This requires a large amount of data collected for many different conditions. The proposed Gaussian process model is an effective framework. It is flexible enough to apply to the diverse conditions. It is also efficient since it converges with a relatively small training data set size. We use a simple method for parameterizing drag actions, however, considering more complex trajectories may increase the performance.

REFERENCES

[1] F. Ng, J. A. Harding, and J. Glass, “An eco-approach to optimise efficiency and productivity of a hydraulic excavator,” Journal of Cleaner Production, vol. 112, pp. 3966–3976, 2016.
[2] S. Dadhich, U. Bodin, and U. Andersson, “Key challenges in automation of earth-moving machines,” Automation in Construction, vol. 68, pp. 212–222, 2016.
[3] L. E. Bersold, “Motion and path control for robotic excavation,” Journal of Aerospace Engineering, vol. 6, no. 1, pp. 1–18, 1993.
[4] S. Singh and R. G. Simmons, “Task planning for robotic excavation,” in IROS, vol. 92, 1992, pp. 1284–1291.
[5] K. Homma, T. Nakamura, T. Arai, and H. Adachi, “Spatial image model for manipulation of shape variable objects and application to excavation,” in IEEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications. IEEE, 1990, pp. 645–650.
[6] E. G. Nezami, Y. M. Hashash, D. Zhao, and J. Ghaboussi, “Simulation of front end loader bucket–soil interaction using discrete element method,” International journal for numerical and analytical methods in geomechanics, vol. 31, no. 9, pp. 1147–1162, 2007.
[7] S. Singh, “Learning to predict resistive forces during robotic excavation,” in Proceedings of 1995 IEEE International Conference on Robotics and Automation, vol. 2. IEEE, 1995, pp. 2102–2107.
[8] A. R. Reece, “Paper 2: The fundamental equation of earth-moving mechanics,” Proc. Inst. Mech. Eng., vol. 179, no. 6, pp. 16–22, 1964.
[9] O. Luengo, S. Singh, and H. Cannon, “Modeling and identification of soil-tool interaction in automated excavation,” in Proceedings, 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190), vol. 3. IEEE, 1998, pp. 1900–1906.
[10] C. E. Rasmussen and C. K. I. Williams, Gaussian Processes for Machine Learning. MIT Press, 2006.
[11] P. W. Goldberg, C. K. Williams, and C. M. Bishop, “Regression with input-dependent noise: A gaussian process treatment,” in Advances in neural information processing systems, 1998, pp. 493–499.
[12] M. Lázaro-Gredilla and M. K. Titsias, “Variational heteroscedastic gaussian process regression.” in ICML, 2011, pp. 841–848.
[13] M. Bauza and A. Rodriguez, “A probabilistic data-driven model for planar pushing,” in 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 3008–3015.
[14] P. Boyle and M. Frean, “Dependent gaussian processes,” in Advances in neural information processing systems, 2005, pp. 217–224.
[15] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, “Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization,” ACM Transactions on Mathematical Software (TOMS), vol. 23, no. 4, pp. 550–560, 1997.